IBM Safer Payments

Version 6 Release 1

Application Programming Interface Reference

General Information about IBM Safer Payments Application Programming Interface: click here

1 ActivateKey 1

2 AddCaseActionAttachment 2

3 AddDefinedRiskEntry 3

4 AddTickerEntry 5

5 AssignElementToInstance 6

6 BulkDeleteDefinedRiskEntries 7

7 BulkEnableDisableDefinedRiskEntries 9

8 CancelGolive 11

9 CancelMasterKeyChange 12

10 CancelSimulationReport 13

11 CanChangeJobEncryptionSettings 14

12 ChangeAuthAddress 15

13 ChangeMasterKey 16

14 ChangePassword 17

15 CheckCalendarComputations 18

16 CheckDataSelection 20

17 CheckIndex 22

18 CheckUserId 23

19 CleanoutRevisions 24

20 CloneSandboxRecord 25

21 CommitInternalModel 26

22 CommitRule 27

23 CommitRuleCondition 28

24 CommitRules 29

25 CommitRuleScore 30

26 CommitRuleset 31

27 CompareRevisions 32

28 ComputeRuleGenerationDataSelection 34

29 ComputeSandboxRecords 35

30 ConfirmGolive 36

31 CopyMapping 37

32 CopyRevision 38

33 CountQuery 39

34 CreateCaseManually 41

35 CreateCasesFromQuery 43

36 CreateDefinedRiskListEntryFromQuery 45

37 CreateDefinedRiskListImportSettings 47

38 CreateRulesScoring 48

39 DeactivatePrivateKeyPasswordSafe 50

40 Delete 51

41 DeleteAnalyses 52

42 DeleteAnalysis 53

43 DeleteDefinedRiskEntries 54

44 DeleteIndexEntry 56

45 DeleteInstance 57

46 DeleteRulegenerationRules 58

47 DeleteTickerEntry 59

48 Detach 60

49 DisableAmi 61

50 DisableApi 62

51 DisableBdi 63

52 DisableEci 64

53 DisableFli 65

54 DisableMci 66

55 DisableMqi 67

56 DisableSimulation 68

57 DiscardCurrentFliMessage 69

58 EnableAmi 70

59 EnableApi 71

60 EnableBdi 72

61 EnableDisableCaseGroup 73

62 EnableDisableComplianceList 74

63 EnableDisableCpps 75

64 EnableDisableDefinedRiskEntries 76

65 EnableDisableDefinedRiskList 77

66 EnableEci 78

67 EnableFli 79

68 EnableMci 80

69 EnableMqi 81

70 EnableSimulationCheckAndReport 82

71 EnableSimulationConfirmed 84

72 ExecuteBulkCaseTransition 85

73 ExecuteCaseTransition 87

74 ExecuteExternalQuery 88

75 ExecuteGroupByQuery 89

76 ExecuteJob 90

77 ExecuteQuery 91

78 ExecuteReport 94

79 ExecuteReportingQuery 95

80 ExecuteRuleQuery 96

81 ExecuteRuleReportQuery 97

82 ExportMultipleRules 98

83 ExportOptimizationAnalysis 99

84 ExportQueryResults 100

85 ExportRuleStatistics 101

86 FixMillisecondsMapping 102

87 GenerateRulesUndoRule 103

88 Get 104

89 GetActivatedComplianceLists 105

90 GetAdHocCheckComplianceListEntriesTable 106

91 GetAggregatedAuditTrails 108

92 GetAllProfilingsTable 110

93 GetAllRulesTable 111

94 GetAnalysedAttributes 113

95 GetAnalysisDataSelection 115

96 GetAnalysisProgress 117

97 GetAnalysisTable 119

98 GetAttributes 120

99 GetAuditLogTable 123

100 GetCaseAction 125

101 GetCaseActionPreview 127

102 GetCaseActions 128

103 GetCaseActionsFromCaseQueue 129

104 GetCaseActionsTable 130

105 GetCaseCloseCodes 131

106 GetCaseCloseCodesTable 133

107 GetCaseData 134

108 GetCaseGroupDefinitions 136

109 GetCaseGroupsTable 137

110 GetCaseHistory 138

111 GetCaseQueueReportsTable 140

112 GetCaseQueues 141

113 GetCaseQueuesTable 142

114 GetCasesTable 143

115 GetCaseStates 146

116 GetCaseStatesTable 148

117 GetCaseTransitions 149

118 GetCaseVariablesAsConditions 151

119 GetCaseWorkflows 152

120 GetCaseWorkflowsTable 153

121 GetChartColorSchema 154

122 GetCharts 155

123 GetChartsTable 157

124 GetCollusions 158

125 GetCollusionSimulationResults 159

126 GetCollusionsTable 160

127 GetCommonPointQueriesTable 162

128 GetCommonQuerySubsetResults 163

129 GetComplianceDataSourceExists 166

130 GetComplianceDefinitionTable 167

131 GetComplianceListDefinitions 168

132 GetConclusionExpressionValuePairs 169

133 GetCountersTable 170

134 GetCpp 171

135 GetCpps 172

136 GetCppsToHighlight 174

137 GetCppTable 175

138 GetDashboardCharts 178

139 GetDashboardKeyPerformanceIndicatorData 179

140 GetDashboardKeyPerformanceIndicators 181

141 GetDefaultCapacities 183

142 GetDefaults 184

143 GetDefinableCaseTransitions 186

144 GetDefinedRiskDefinitions 187

145 GetDefinedRiskListAuditTrailTable 189

146 GetDefinedRiskListDefinitionTable 191

147 GetDefinedRiskListEntriesTable 193

148 GetDefinedRiskListEntry 196

149 GetDefinedRiskListEntryAttribute 198

150 GetDefinedRiskListPrivileges 199

151 GetDefinedRiskListSettings 200

152 GetDeviceIdentificationTable 201

153 GetDonorInstances 203

154 GetElementPrintView 204

155 GetEnabledAnalyses 206

156 GetEnabledRulesetsOfAnalysis 207

157 GetEventLogMessage 208

158 GetEventLogMessagesTable 210

159 GetEventsTable 211

160 GetExecutables 212

161 GetExternalQueries 213

162 GetExternalQueriesTable 214

163 GetFastLinkStatusTable 215

164 GetFinalRulesets 216

165 GetFinalRulesetsTable 217

166 GetFittingRulesets 219

167 GetFollowupUsers 220

168 GetFormulasTable 222

169 GetFraudValues 223

170 GetGroupByQueriesTable 224

171 GetGroupByQueryAccountResults 225

172 GetGroupByQueryResult 227

173 GetGroupByQueryResultMetaInfo 229

174 GetGroupByQueryResults 231

175 GetIndexAspects 232

176 GetIndexAttribute 233

177 GetIndexBasedEvaluations 235

178 GetIndexBasedEvaluationsTable 236

179 GetIndexedAttributes 237

180 GetIndexes 238

181 GetIndexesTable 240

182 GetInputsTable 241

183 GetInstances 242

184 GetInstancesTable 243

185 GetInternalModelGenerationDataSelection 244

186 GetInternalModelGenerationSettings 247

187 GetInternalModelGenerationStatus 248

188 GetInvestigationReportsTable 249

189 GetIsRulesetEnabled 250

190 GetJobsTable 251

191 GetKeyPerformanceIndicatorsTable 252

192 GetKeysTable 253

193 GetLastUrid 254

194 GetLastUserAction 255

195 GetListsTable 256

196 GetLoadBalancingInstances 257

197 GetMandators 258

198 GetMandatorSelectionTable 261

199 GetMandatorSelectionTree 262

200 GetMandatorsTable 264

201 GetMappingsTable 265

202 GetMasterdataDeepInformation 267

203 GetMasterdatas 269

204 GetMasterdatasForIndex 271

205 GetMasterdatasTable 272

206 GetMasterKeysTable 273

207 GetMemoryManagementTable 274

208 GetMerchantMonitoringRulesTable 276

209 GetMerchantMonitoringTemplate 277

210 GetMerchantMonitoringTemplates 279

211 GetMergingsTable 280

212 GetMessageReport 281

213 GetMessages 282

214 GetMessagesReport 283

215 GetMessagesTable 284

216 GetMetaAttributeAmount 285

217 GetMetaAttributeMaximum 286

218 GetMissedCasesReportsTable 287

219 GetModelComponents 288

220 GetModelComponentsTable 289

221 GetModelingWorkflowsTable 290

222 GetModelling 291

223 GetModellingsTable 293

224 GetMultipleRulesFiringStatistics 295

225 GetMyWorkingQueues 297

226 GetNextCase 298

227 GetNotifications 299

228 GetNotificationsTable 300

229 GetOfflineComplianceListDefinitions 301

230 GetOptimizationAnalysis 302

231 GetOutgoingChannelConfigurations 303

232 GetOutgoingChannelConfigurationTable 304

233 GetOutputsTable 305

234 GetPasswordSafes 306

235 GetPasswordSafeTable 307

236 GetPatternsTable 308

237 GetPciDssReport 309

238 GetPossibleAmountAttributes 310

239 GetPrecedentsTable 312

240 GetPreference 313

241 GetPreProcessingRulesets 314

242 GetPreProcessingRulesetsTable 315

243 GetPrintView 316

244 GetProcessPriorities 317

245 GetProfiles 318

246 GetProfilesTable 319

247 GetPythonFunctions 320

248 GetQueries 321

249 GetQueriesTable 322

250 GetQueryOptions 323

251 GetQueryResults 324

252 GetQuickSearchCasesTable 327

253 GetRealtimeInterceptionCodes 329

254 GetRealtimeInterceptionCodesTable 330

255 GetReferencesOfAttribute 331

256 GetReferencesOfCaseAction 332

257 GetReferencesOfCaseCloseCode 333

258 GetReferencesOfCaseQueue 334

259 GetReferencesOfCaseState 335

260 GetReferencesOfCaseWorkflow 336

261 GetReferencesOfChart 337

262 GetReferencesOfComplianceList 338

263 GetReferencesOfDefinedRiskList 339

264 GetReferencesOfEvent 340

265 GetReferencesOfExternalQuery 341

266 GetReferencesOfIndex 342

267 GetReferencesOfMasterdata 343

268 GetReferencesOfMessage 344

269 GetReferencesOfNotification 345

270 GetReferencesOfOutgoingChannelConfiguration 346

271 GetReferencesOfPasswordSafe 347

272 GetReferencesOfProfile 348

273 GetReferencesOfPythonModule 349

274 GetReferencesOfQuery 350

275 GetReferencesOfReminder 351

276 GetReferencesOfUserGroup 352

277 GetReminders 353

278 GetReminderTable 354

279 GetReportingAttributes 355

280 GetReportingQueriesTable 357

281 GetReportingQueryAccountResults 358

282 GetReportingQueryResult 360

283 GetReportingQueryResultMetaInfo 362

284 GetReportingQueryResults 364

285 GetRevisionAuditTrail 365

286 GetRevisionAuditTrailTable 367

287 GetRevisionGeneral 368

288 GetRevisions 371

289 GetRevisionSelectionTable 372

290 GetRevisionStatus 374

291 GetRoles 375

292 GetRolesTable 376

293 GetRuleGenerationDataSelection 377

294 GetRuleGenerationDesignTable 380

295 GetRuleGenerationExplore 382

296 GetRuleGenerationPerformance 384

297 GetRuleGenerationRules 387

298 GetRuleGenerationSettings 388

299 GetRuleGenerationStatus 390

300 GetRuleGenerationTargets 391

301 GetRulePerformance 392

302 GetRulePerformanceRulesets 394

303 GetRuleReport 396

304 GetRulesetsForAnalysis 399

305 GetRulesStatistics 400

306 GetRulesTable 402

307 GetRuleStatistics 404

308 GetRunningSimulations 406

309 GetSandboxTable 407

310 GetSearchComplianceListEntriesTable 408

311 GetSessionCountdownData 410

312 GetSettings 411

313 GetSimulationDataSelection 412

314 GetSimulationProgress 414

315 GetSimulationTable 416

316 GetStatistics 418

317 GetStatusAlarmIndicatorsDisplay 419

318 GetStatusAlarmIndicatorsTable 421

319 GetSystemInformation 422

320 GetSystemLogTable 423

321 GetTargetRevisions 425

322 GetTextModules 426

323 GetTextModulesTable 427

324 GetThreadPriorities 428

325 GetTickerEntries 429

326 GetTimestampAttributes 431

327 GetToAddressMeta 433

328 GetTransactionMessageReportTable 434

329 GetUserGroups 436

330 GetUserGroupsTable 437

331 GetUsers 438

332 GetUsersAndMandatorsOfRoles 440

333 GetUsersOfRole 441

334 GetUsersReportsTable 442

335 GetUsersTable 443

336 GetWorkingQueueCasesTable 445

337 GetWorkingQueues 447

338 GetWorkingQueuesTable 448

339 GetWorkingQueueStatistics 449

340 GoliveCheckAndReport 451

341 HasEntry 452

342 InsertPin 453

343 InterruptCase 454

344 InvestigateCase 456

345 IsCasesTableEncrypted 457

346 IsCppTableEncrypted 460

347 IsDdcEnabled 462

348 IsDdcEnabledForManualCaseCreation 463

349 IsDefinedRiskListEncrypted 464

350 IsGoliveRunning 465

351 IsGroupByQueryResultEncrypted 466

352 IsIndexSequenceMdcOutsideSimulation 467

353 IsMetaAttributeInChampions 468

354 IsOffline 469

355 IsPasswordValid 470

356 IsQueryResultEncrypted 471

357 IsReportingQueryResultEncrypted 472

358 IsRevisionDeletionEnabled 473

359 IsSamplingEnabled 474

360 IsSimulationValid 475

361 IsValidInstanceId 476

362 IsWorkingQueueCasesTableEncrypted 477

363 LoadCaseAuditTrailsFromDisk 478

364 Login 479

365 Logout 480

366 MarkFraud 481

367 MoveRules 482

368 MultiDelete 483

369 ParentPathExists 484

370 PathExists 485

371 ReAttach 486

372 RebuildAllIndexes 487

373 RebuildIndex 488

374 RebuildIndexSequence 489

375 RecomputeAttributes 490

376 RefreshRandomForestClassificationThreshold 491

377 ReGoliveRevision 492

378 ReleaseFreeMemory 493

379 ReloadComplianceLists 494

380 ReloadPrivateKeys 495

381 ReloadPrivateKeysPasswordSafe 496

382 Reserve 497

383 ReserveEventLogMessage 498

384 ResetAllUserPreferences 499

385 ResetIndex 500

386 ResetOutgoingFli 501

387 ResetPreference 502

388 ResetRuleGeneration 503

389 ResetSandboxRecords 504

390 ResetUserPreferences 505

391 RestartInstance 506

392 Restore 507

393 RetireChampion 508

394 RevertToChallenger 509

395 RevisionElementExists 510

396 RevokeKey 512

397 RewindFliBuffer 513

398 RewriteRiskList 514

399 RuleReportQueryExists 515

400 Save 516

401 SaveCluster 519

402 SaveCpp 524

403 SaveEventLogMessage 526

404 SaveSandboxEntry 528

405 SaveSettings 529

406 SendCaseActionFromPreview 552

407 SendCaseActions 554

408 Serialize 555

409 SetAllMyPreferencesAsDefault 556

410 SetAnalysis 557

411 SetAttachmentData 559

412 SetCrashWorkflow 560

413 SetFinalRulesetsEnabled 561

414 SetInternalModelGenerationDataSelection 562

415 SetInternalModelGenerationSettings 565

416 SetKey 567

417 SetMasterdata 569

418 SetModelComponentsEnabled 570

419 SetModelling 571

420 SetMultiInterfaces 572

421 SetMultiModelling 574

422 SetOffline 575

423 SetOnline 576

424 SetPreference 577

425 SetQuickSearchCasesTablePreference 578

426 SetRuleGenerationCategoricIndicators 579

427 SetRuleGenerationDataSelection 580

428 SetRuleGenerationFollowingIndicators 583

429 SetRuleGenerationIndicator 584

430 SetRuleGenerationPrecedingIndicators 585

431 SetRuleGenerationSettings 586

432 SetRulesEnabled 588

433 SetSimulationDataSelection 589

434 SetUserExportPassword 591

435 SetXdcSizes 592

436 Shutdown 593

437 SimulateCollusion 594

438 StartAnalysis 595

439 StartAutomaticRuleGeneration 596

440 StartInternalModelGeneration 597

441 StopAnalysis 598

442 StopAutomaticRuleGeneration 599

443 StopCollusionSimulation 600

444 StopInternalModelGeneration 601

445 StopJob 602

446 StopQuery 603

447 StopRuleGeneration 604

448 TakeoverCase 605

449 TakeoverRevision 606

450 Test 607

451 TestLdapConnection 608

452 UnmarkFraud 610

453 Unreserve 611

454 UnreserveCase 612

455 UnreserveDefinedRiskListEntry 613

456 UnreserveEventLogMessage 614

457 UpdateRiskListEntry 615

1 ActivateKey

Parameters

httpRequest: Holds the key uid and User object that have been passed to the server.

```
{
    "request": "activateKey",
    "uid": <uid>
}
```

- "uid": Uid of the encryption key to be activated.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

2 AddCaseActionAttachment

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "addCaseActionAttachment",
    "uid": <caseUid>
}
```

- "uid" (mandatory): Uid of the investigation case.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

3 AddDefinedRiskEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "addDefinedRiskEntry",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "value": "<inputAttributeValue>",
        "outputAttributeCategoryValue": "<outputAttributeCategoryValue>" | "undefined",
        "expiresAt": <expiresAt>,
        "startsAt": <startsAt>,
        "id": <entryId>,
        "comment": "<comment>",
        "label": "<label>",
        "active": true|false,
        "conditions": [<conditions>]
    }
}
```

- "uid": Uid of the risk list, in which the entry should be added.
- "mandator": Uid of mandator, for which the risk list is defined.
- "data:value": The value of the entry.
- "data:outputAttributeCategoryValue": If a category is defined for the attribute the category value has to be send.
- "data:expiresAt": If the entry should expire, the date can be assigned here.
- "data:startsAt": If the entry should only be computed after a defined date, the date can be assigned here.

- "data:id": Id of the entry in the list.
- "data:comment": A comment for this entry can be assigned.
- "data:label": Id of the entry in the list.
- "data:active": Defines if the entry should be active or inactive.
- "data:conditions": Additional conditions which have to be met for this entry to be applied in computation.

Returns

A HTTP response with the following content:

```
{
    "comparisonResult": "Unchanged" | "ChangedNoImpact" | "ChangedComputationImpact",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

4 AddTickerEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "addTickerEntry",
    "data": {
        "userUid": <userUid>,
        "title": "<tickerTitle>",
        "note": "<tickerNote>",
        "mandatorUid": <mandatorUid>,
        "mandatorUids": [<mandatorUid>,..]
    }
}
```

- "data:userUid": Uid of the user, who adds the entry.
- "data:title": Title of the entry.
- "data:note": Text of the entry.
- "data:mandatorUid": Uid of the mandator that the ticker entry to should added to.
- "data.mandatorUids": Uids of a list of mandators that the ticket entry should be displayed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

5 AssignElementToInstance

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "assignElementToInstance",
    "uid": <elementUid>,
    "data": {
        "instanceUid": <instanceUid>
    }
}
```

- "uid" (mandatory): Uid of the element to assign to an instance. Right now only revisions are supported.
- "data:instanceUid" (mandatory): Uid of the instance to assign the element to.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

6 BulkDeleteDefinedRiskEntries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "bulkDeleteDefinedRiskEntries",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "items": [
        {
            "attributeValue": "<attributeValue>",
            "id": <entryId>
        },
        ... ...
        ]
    }
}
```

- "uid": Uid of the risk list, from which the entries should be deleted.
- "mandator": Uid of the mandator in which the risk list is defined.
- "data:items": The list of items to delete.
- "data:items:attributeValue": The attribute value of the entry (being used to identify entry).
- "data:items:id": Id of the entry (used to identify entry).

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

7 BulkEnableDisableDefinedRiskEntries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "bulkEnableDisableDefinedRiskEntries",
    "uid": <defineRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "active": true|false,
        "items": [
        {
            "attributeValue": "<inputAttributeValue>",
            "id": <entryId>
        },
        ... ...
        ]
    }
}
```

- "uid": Uid of the risk list, from which the entries should be enabled/disabled.
- "mandator": Uid of the mandator in which the risk list is defined.
- "data:active": Set to true if the entries should be enabled, set false to disable.
- "data:items": List of items, to enable or disable.
- "data:items:attributeValue": The attribute value of the entry (used to identify entry).
- "data:items:id": Id of the entry (used to identify entry).

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

8 CancelGolive

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "cancelGolive",
    "revision": <revisionUid>
}
```

- "revision": Uid of the revision whose golive report should be canceled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

9 CancelMasterKeyChange

Parameters

httpRequest: Holds the parameter invalidateInstance uid and user object.

```
{
    "request": "cancelMasterKeyChange",
    "data": {
        "invalidateInstance": <true/false>
    }
}
```

- "data:invalidateInstance": Indicates if the instance status is "INVALIDATED" or "OK" after cancellation.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

10 CancelSimulationReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "cancelSimulationReport",
    "revision": <revisionUid>
}
```

- "revision": Uid of the revision whose simulation report should be canceled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

11 CanChangeJobEncryptionSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "canChangeJobEncryptionSettings"
}
```

Returns

A HTTP response with the following content:

```
{
    "canChangeJobEncryptionSettings": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

12 ChangeAuthAddress

13 ChangeMasterKey

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "changeMasterKey",
    "uid": <masterKeyUid>
}
```

- "uid" (mandatory): Uid of the new master key.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

14 ChangePassword

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "changePassword",
    "data": {
        "oldPassword": "<oldPassword>",
        "newPassword": "<newPassword>"
    }
}
```

- "data:oldPassword" (mandatory): The old password which should be changed.
- "data:newPassword" (mandatory): The new password to set to the user.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

15 CheckCalendarComputations

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "checkCalendarComputations",
    "mandator": <mandatorUid>,
    "data": {
        "indexUid": <indexUid>,
        "calendarComputations": [
            ...
        ]
    }
}
```

- "mandator" (mandatory): Uid of the mandator that the calendar computations belong to.
- "data" (mandatory): The calendar computations to check.
- "data:indexUid" (mandatory): Uid of the index that the selected calendar uses.
- "data:calendarComputations" (mandatory): Calendar computation data streams.

Returns

A HTTP response with the following content:

```
{
    "attributes": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

- "attributes": The calendar computations written like attributes to be usable by the conditions client-code.

16 CheckDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "checkDataSelection",
    "type": "<selectionType>",
    "revision": <revisionUid>,
    "data": {
        "mandators": [<mandatorUid>],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "desired": {
            "fromUrid": <fromUrid>,
            "toUrid": <toUrid>,
            "fromRecord": <fromRecord>,
            "toRecord": <toRecord>,
            "fromTimestamp": <fromTimestamp>,
            "toTimestamp": <toTimestamp>,
            "fromDays": <fromDays>,
            "toDays": <toDays>
        }
    }
}
```

- "type": The type of the date selection.
- "revision": Uid of the revision in which the data selection is defined.
- "data:mandators": Uids of a list of mandators whose records should be included in the selection.
- "data:periodType": Choose one of these four period types for how the records are selected (by time, recordId or relative by count).
- "data:desired": The desired selection, defined by following params.

- "data:desired:fromUrid": Starting Urid of selection (for RecordsAbsolute period type).
- "data:desired:toUrid": Last Urid of selection (for RecordsAbsolute period type).
- "data:desired:fromRecord": Starting number of selection (for RecordsRelative period type).
- "data:desired:toRecord": Last record number of selection (for RecordsRelative period type).
- "data:desired:fromTimestamp": Starting timestamp of selection (for TimeAbsolute period type).
- "data:desired:toTimestamp": End timestamp of selection (for TimeAbsolute period type).
- "data:desired:fromDays": Number of days back of records are included for selection (for TimeRelative period type).
- "data:desired:toDays": Number of days back of records are excluded for selection (for TimeRelative period type).

Returns

A HTTP response with the following content:

```
{
    "dataSelection": {
        <dataSelection>
    },
    "minimumUrid": <minimum urid>,
    "maximumUrid": <maximum urid>,
    "minimumSystemtime": <minimum systemtime>,
    "maximumSystemtime": <maximum systemtime>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

17 CheckIndex

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "checkIndex",
    "uid": <indexUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the index to be checked.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

18 CheckUserId

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "checkUserId",
    "data": {
        "userId": <userUid>
    }
}
```

- "data:userId" (mandatory): Uid of the user that needs to be checked.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

19 CleanoutRevisions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "cleanoutRevisions"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

20 CloneSandboxRecord

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "cloneSandboxRecord",
    "uid": <sourceSandboxRecordUid>,
    "revision": <revisionUid>,
    "data": {
        "name": "<targetSandboxRecordName>"
    }
}
```

- "uid": Uid of record to be cloned.
- "revision": Uid of the revision in which the record will be created.
- "data:name": The name of the new sandbox record.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

21 CommitInternalModel

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitInternalModel",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the internal model will be created.
- "data": All data pertaining to the generated model.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>",
    "reloadUserProfile": true|false
}
```

22 CommitRule

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitRule",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the rule will be created.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

23 CommitRuleCondition

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitRuleCondition",
    "uid": <attributeUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the attribute in the proposed rule condition.
- "revision" (mandatory): Uid of the revision in which the rule will be created.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

24 CommitRules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitRules",
    "revision": <revisionUid>,
    "ruleset": <rulesetUid>,
    "data": {
        "rules": [<ruleUid>, ...],
        "rulesetData": {
                "uid": number,
                "enabled": boolean,
                "priority": number,
                "name": string,
        }
    }
}
```

- "revision" (mandatory): Uid of the revision in which the rule will be created.
- "ruleset" (optional): Uid of the ruleset in which the rules should be added.
- "data:rules" (mandatory): A list of generated rule uids to be added to the ruleset.
- "data:rulesetData" (optional, only needed if param "ruleset" is set to -1): All data necessary to create or update a ruleset, where rules should be added.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

25 CommitRuleScore

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitRuleScore",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>,
        "allRules": true|false,
        "ruleUids": [<ruleUid>, ...]
    }
}
```

- "revision" (mandatory): Uid of the revision in which the rules are defined in.
- "data:analysisId" (mandatory): Id of the analysis.
- "data:allRules" (mandatory): Indicates whether all rules that were scored should be committed or only a subset (see data:rules).
- "data:rules" (optional): List of uids of rules to be committed, if not all rules that were scored should be committed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

26 CommitRuleset

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "commitRuleset",
    "revision": <revisionUid>,
    "ruleset": <rulesetUid>
}
```

- "revision" (mandatory): Uid of the revision in which the ruleset will be created.
- "ruleset" (optional): Uid of the ruleset where the rules should be added.
- "data" (optional if param "ruleset" is set to -1): All data necessary to create a new ruleset.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>",
    "reloadUserProfile": true|false
}
```

27 CompareRevisions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "compareRevisions",
    "uid": <sourceRevisionUid>,
    "revision": <targetRevisionUid>
}
```

- "uid" (mandatory): Uid of the source revision for the comparison.
- "revision" (mandatory): Uid of the target revision for the comparison.

Returns

A HTTP response with the following content:

```
{
    "showUnchangedElements": "<showUnchangedElements>",
    "source": "[<sourceRevisionId] '<sourceRevisionName>'",
    "target": "[<targetRevisionId] '<targetRevisionName>'",
    "sourceRevisionUid": <sourceRevisionUid>,
    "targetRevisionUid": <targetRevisionUid>,
    "sourceStatus": "<sourceRevisionStatus>",
    "targetStatus": "<targetRevisionStatus>",
    "sourceUser": "<sourceRevisionCurrentlyBeingEditedByUserName>",
    "targetUser": "<targetRevisionCurrentlyBeingEditedByUserName>",
    "comparisonResult": [{
```

```
        "listValueInAudittrailEncrypted": true|false,
        "notificationEntry": true|false,
        "userName": "<userName>",
        "userUid": <userUid>,
        "utc": <utcTimestamp>,
        "changedElementName": "<changedElementName>",
        "changedElementUid": <changedElementUid>,
        "revisionNumber": <revisionNumber>,
        "revisionName": "<revisionName>",
        "uid": <uid>,
        "comparisonResult": "<comparisonResult>",
        "type": "<type>",
        "name": "<name>",
        "origin": "<origin>",
        "changes": [{
            "type": "name",
            "status": "Unchanged",
            "needsEncryption": false,
            "sourceValue": "Primary Record ID",
            "targetValue": "Primary Record ID"
        },
        ... ...
        ]
    },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

28 ComputeRuleGenerationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "computeRuleGenerationDataSelection",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the rule generation should run.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

29 ComputeSandboxRecords

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "computeSandboxRecords",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the sandbox computation should run.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

30 ConfirmGolive

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "confirmGolive",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision to execute golive.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

31 CopyMapping

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "copyMapping",
    "revision": <revisionUid>,
    "data": {
        "messageUid": <sourceMessageUid>,
        "targetMessageUid": <targetMessageUid>,
        "attributeUid": <attributeUid>
    }
}
```

- "revision" (mandatory): Uid of the revision in which the mapping should be copied.
- "data:messageUid" (mandatory): Uid of the source message that the mapping should be copied from.
- "data:targetMessageUid" (mandatory): Uid of the target message that the mapping should be copied to.
- "data:attributeUid" (mandatory): Uid the attribute that the mapping should be copied for.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

32 CopyRevision

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "copyRevision",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision that should be copied to create a new challenger.

Returns

A HTTP response with the following content:

```
{
    "revisionUid": <newRevisionUid>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false,
    "uid": <newRevisionUid>
}
```

33 CountQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "countQuery",
    "uid": <queryUid>,
    "type": "query" | "simulationQuery",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "parameter: "<parameter>"
    }
}
```

- "uid" (mandatory): Uid of the query which should be executed.
- "type" (mandatory): The type of query, which is either "query" or "simulationQuery".
- "mandator" (mandatory): Uid of the mandator for which the query is defined.
- "revision" (optional): Uid of the revision that the simulation query belongs to.
- "data:parameter" (optional): Index value for hyperlink or index queries to query for.

Returns

A HTTP response with the following content:

```
{
    "amountDecimals": "<amountDecimals>",
    "amountDimension": "<amountDimension>",
    "statistics": {
        "transaction": {
            "genuine": {
                "records": <genuineRecords>,
                "amount": <amount>
            },
            "fraud": {
                "records": <fraudRecords>,
                "amount": <amount>
            }
        }
    },
    "queryUid": <queryUid>,
    "recordsHit": <recordsHit>,
    "parameter": "<parameter>",
    "duration": <duration>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

- "statistics:transactions": It shows how many records are genuine/fraud and how large the amount is for each.

34 CreateCaseManually

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "createCaseManually",
    "mandator": <mandatorUid>,
    "data": {
        "caseQueue": <caseClassUid>,
        "caseScore": "<caseScore>",
        "memo": "<comment>",
        "cpp": "-1"|"<cppId>",
        "reportingAttributes": [
            {
                "attributeId": <attributeUid>,
                "value": "<value>"
            },
            ... ...
        ]
    }
}
```

- "mandator": Uid of the mandator.
- "data:caseQueue": Uid of the case class.
- "data:caseScore": Score of the case to manually create.
- "data:memo": Comment of the case to manually create.
- "data:cpp": The value is a combination of the cpp id and case group id. For example: 1_1234, where '1' is the cpp id and '1234' is the case group id. '-1' means no cpp.
- "data:reportingAttributes:attributeId": Uid of the reporting attribute.
- "data:reportingAttributes:value": Value of the reporting attribute.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

35 CreateCasesFromQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "createCasesFromQuery",
    "mandator": <mandatorUid>,
    "data": {
        "caseQueue": <caseQueueUid>,
        "urids": [<urid>,...],
        "caseScore": <caseScore>,
        "memo": "<memo>",
        "cpp": "<cpp>" | "-1",
        "includeDdc": true|false
    }
}
```

- "mandator" (mandatory): Uid of the owning mandator for the newly created case.
- "data:caseQueue": Cases will be created with this case class.
- "data:urids": URIDs of a list of records for which cases should be created.
- "data:caseScore": A numeric value indicating the score of a case.
- "data:memo": An short explanation or comment on creating cases.
- "data:cpp": If selected, cases will be assigned to this CPP.
- "data:includeDdc": If enabled, the DDC data will be available for the case creation.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

36 CreateDefinedRiskListEntryFromQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "createDefinedRiskListEntryFromQuery",
    "uid": <definedRiskListUid>,
    "data": {
        "urid": [<queryResultRecordUrid>],
        "outputAttributeCategoryValue": "<outputAttributeCategoryValue>" | "undefined",
        "expiresAt": <expiresAtTimestamp>,
        "startsAt": <startsAtTimestamp>,
        "comment": "<comment>",
        "label": "<label>",
        "active": true|false,
        "conditions": [ ... ]
    }
}
```

- "uid": Uid of a risk list for which a new entry should be created.
- "data:urid": Unique record id for which risk list entry will be created.
- "data:outputAttributeCategoryValue": The value of output attribute.
- "data:expiresAt": Expiry date for the risk list entry.
- "data:startsAt": Can be used to define risk list entry for future use; only records past this date are checked against this risk list.
- "data:comment": A comment to be added to newly created risk list.
- "data:label": A label to be added to newly created risk list.
- "data:active": If active, risk list will be enabled for rule actions.
- "data:conditions": The risk list will be applied to transaction records satisfying these conditions.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

37 CreateDefinedRiskListImportSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "createDefinedRiskListImportSettings",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "overwrite": true|false,
        "value": "-1",
        "outputAttributeCategoryValue": "<outputAttributeCategoryValue>" | "undefined",
        "expirsAt": <expiresAtTimestamp>,
        "startsAt": <startsAtTimestamp>,
        "id": -1,
        "comment": "<comment>",
        "label": "<label>",
        "active": true|false,
        "conditions": [...]
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "id": <id>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

38 CreateRulesScoring

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "createRulesScoring",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>,
        "rulesets": [<rulesetUid>,...],
        "scoreType": "Standard" | "SAPFA",
        "scoreConclusionAttribute": <scoreConclusionAttributeUid>
    }
}
```

- "revision" (mandatory): Uid of the revision for which rule scoring should be created.
- "data:analysisId" (mandatory): Id of the analysis for which rule scoring should be created.
- "data:rulesets" (mandatory): For each ruleset supplied a rule scoring will be created.
- "data:scoreType" (optional): The type of the rule scoring, by default "Standard" is used.
- "data:scoreConclusionAttribute": Uid of the conclusion attribute.

Returns

A HTTP response with the following content:

```
{
    "ruleScoring": [
        [
            <ruleUid>,
            <rulesetUid>,
            "<ruleName> (<rulesetName>)",
            <hitRate>,
            <falseAlarmRatio>,
            <savedAmountPerFalseAlarm>,
            <fraudAmountIntercepted>,
            <fraudTransactionsIntercepted>,
            <genuineAmountIntercepted>,
            <genuineTransactionsIntercepted>,
            "<rulesetName>",
            "<ruleName>",
            <scoring>
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

39 DeactivatePrivateKeyPasswordSafe

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deactivatePrivateKeyPasswordSafe",
    "uid": <privateKeyPasswordSafeUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

40 Delete

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "delete",
    "uid": <elementUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the element to be deleted.
- "revision": Uid of the revision whose element is to be deleted. This variable is required if the element to delete is a revision element.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

41 DeleteAnalyses

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deleteAnalyses",
    "revision": <revisionUid>,
    "data": {
        "analysisId": [<analysisId>, ...]
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analyses should be deleted.
- "data:analysisId" (mandatory): A list of uids of analyses to be deleted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

42 DeleteAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deleteAnalysis",
    "uid": <analysisUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the analysis to be deleted.
- "revision" (mandatory): Uid of the revision from which the analysis should be deleted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

43 DeleteDefinedRiskEntries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deleteDefinedRiskEntries",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "items": [
        {
            "attributeValue": "<attributeValue>",
            "id": <entryId>
        },
        ... ...
        ]
    }
}
```

- "uid": Uid of the risk list, from which the entries should be deleted.
- "mandator": Uid of the mandator in which the risk list is defined.
- "data:items": The list of items to delete.
- "data:items:attributeValue": The attribute value of the entry (being used to identify entry).
- "data:items:id": Id of the entry (being used to identify entry).

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

44 DeleteIndexEntry

Parameters

httpRequest: Holds the key uid and User object that have been passed to the server.

```
{
  "request": "deleteIndexEntry",
  "uid": <indexUid>,
  "mandator": <mandatorUid>,
  "data": {
  "toBeDeletedEntry": "<indexEntryValue>"
  }
}
```

- "uid": Uid of the index from which the value should be deleted.
- "mandator": Uid of the mandator for which the index is defined.
- "data:toBeDeletedEntry": The value that should be deleted from the index.

Returns

A HTTP response with the following content:

```
{
  "responseStatus": ["<status>", "<feedbacktext>"],
  "reloadUserProfile": true|false
}
```

45 DeleteInstance

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "deleteInstance",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance that is to be deleted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

46 DeleteRulegenerationRules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deleteRulegenerationRules",
    "revision": <revisionUid>,
    "data": {
        "rules": [<ruleUid>,...]
    }
}
```

- "revision": Uid of the revision in which the rule generation is executed.
- "data:rules": Uids of a list of rules to delete from the generator.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

47 DeleteTickerEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "deleteTickerEntry",
    "data": {
        "entryId": <entryId>
    }
}
```

- "data:entryId": Uid of the ticket entry that should be deleted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

48 Detach

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "detach",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance, which should be detached from the cluster.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

49 DisableAmi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableAmi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the AMI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

50 DisableApi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableApi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the API should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

51 DisableBdi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableBdi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the BDI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

52 DisableEci

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableEci",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the ECI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

53 DisableFli

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableFli",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the FLI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

54 DisableMci

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableMci",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the MCI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

55 DisableMqi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "disableMqi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the MQI should be disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

56 DisableSimulation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "disableSimulation",
    "data":{
        "revisions": [<revisionUids>]
    }
}

or

{
    "request": "disableSimulation",
    "revision": <revisionUid>
}
```

- "revision" (optional): Uid of revision in which simulation should be stopped.
- "data:revisions" (optional): Uids of a list of revisions where simulation should be stopped.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

57 DiscardCurrentFliMessage

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "discardCurrentFliMessage",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance to which the outgoing FLI buffer should discard currently blocking message.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

58 EnableAmi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableAmi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the AMI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

59 EnableApi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableApi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the API should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

60 EnableBdi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableBdi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the BDI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

61 EnableDisableCaseGroup

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableDisableCaseGroup",
    "uid": <caseGroupUid>,
    "mandator": <mandatorUid>,
    "data": {
        "activate": true|false
    }
}
```

- "uid" (mandatory): Uid of the case group that should be enabled / disabled.
- "mandator" (mandatory): Uid of the mandator of the case group.
- "data:activate" (mandatory): Set to true to enable, false to disable.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

62 EnableDisableComplianceList

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableDisableComplianceList",
    "uid":<complianceListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "activate": true|false
    }
}
```

- "uid" (mandatory): Uid of the compliance list that should be enabled / disabled.
- "mandator" (mandatory): Uid of the mandator of the case group.
- "data:activate" (mandatory): Set to true to enable, false to disable.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

63 EnableDisableCpps

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableDisableCpps",
    "uid": <caseGroupUid>,
    "mandator": <mandatorUid>,
    "data": {
        "activate": true|false,
        "id": <cppId>
    }
}
```

- "uid" (mandatory): Uid of the case group that owns the CPP.
- "mandator" (mandatory): Uid of the mandator of the case group.
- "data:activate" (mandatory): Set to true if enable, false if disable.
- "data:id" (mandatory): Id of the CPP that should be enabled / disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

64 EnableDisableDefinedRiskEntries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableDisableDefinedRiskEntries",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "activate": true|false,
        "items":[<attributeValue>,<id>]
    }
}
```

- "uid" (mandatory): Uid of the defined risk list that owns the entries.
- "mandator" (mandatory): Uid of the mandator of the case group.
- "data:activate" (mandatory): Set to true to enable, false to disable.
- "data:items" (mandatory): Holds attribute value and id of defined risk list entry that should be enabled / disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

65 EnableDisableDefinedRiskList

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableDisableDefinedRiskList",
    "uid":<definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "activate":true|false
    }
}
```

- "uid" (mandatory): Uid of the defined risk list that should be enabled / disabled.
- "mandator" (mandatory): Uid of the mandator of the case group.
- "data:activate" (mandatory): Set to true to enable, false to disable.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

66 EnableEci

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableEci",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the ECI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

67 EnableFli

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableFli",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the FLI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

68 EnableMci

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableMci",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the MCI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

69 EnableMqi

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "enableMqi",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance on which the MQI should be enabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

70 EnableSimulationCheckAndReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableSimulationCheckAndReport",
    "revision": <revisionUid>,
    "data": {
        "ruleUid": <ruleUid>
    }
}
```

- "revision" (mandatory): Uid of challenger revision for which the simulation report should be generated.
- "data:ruleUid" (optional): Uid of rule for which a rule report should be started (only when started from rule).

Returns

A HTTP response with the following content:

```
{
    "simulationReport": [{
        "type": "simulationReportSimulateAttribute",
        "variables": ["<elementType>", "<elementName>", <value>]
    },
    {
        "type": "simulationReportSimulate",
        "variables": ["<elementType", "<elementName>", <value>]
    }, {
```

```
        "type": "simulationReportJustUseInput",
        "variables": ["<elementType>", "<elementName>"]
    }, {
        "type": "simulationReportJustUse",
        "variables": ["<elementType>", "<elementName>", <value>]
    }
    ],
    "memoryRequired": <value>,
    "memoryAvailable": <value>,
    "userMemoryAvailable": true|false,
    "mandatorMemoryAvailable": true|false,
    "systemMemoryAvailable": true|false,
    "simulatedRecords": <value>,
    "userMemoryDefined": <value>,
    "canStartSimulation": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

71 EnableSimulationConfirmed

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "enableSimulationConfirmed",
    "revision": <revisionUid>,
    "data": {
        "ruleUid": <ruleUid>
    }
}
```

- "revision" (mandatory): Uid of the challenger revision for which the simulation should be started.
- "data:ruleUid" (optional): Uid of rule which should be simulated (only when started from rule).

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

72 ExecuteBulkCaseTransition

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeBulkCaseTransition",
    "data": {
        "cases": [<caseUid>,<caseUid>,...],
        "caseTransition": <caseTransitionUid>,
        "justification": "<justificationCodes>",
        "comment": "<comment>",
        "caseCloseCode": <caseCloseCodeUid>
    },
    "type": "<type>"
}
```

- "cases" (optional): Uids of a list of cases upon which the transition should be performed.
- "caseTransition" (mandatory): Uid of the transition that should be performed on all the cases.
- "justification" (optional): The justification (as a string) to be used for the case transition. It is mandatory if required by the case transition.
- "comment" (optional): A description to be used for the transition. It is mandatory if required by the case transition.
- "caseCloseCode" (optional): Uid of the case close code to be used for the cases. It is mandatory if the cases transition to a closed state.
- "type" (optional): If the user wants to interrupt/intercept a case from another user(s), then the type needs to be set to "interrupt" in which case none of the other parameters except the cases array are used.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

73 ExecuteCaseTransition

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeCaseTransition",
    "uid": <caseUid>,
    "data": {
        "caseTransition": <caseTransitionUid>,
        "justification": "<justificationCodes>",
        "comment": "<comment>",
        "caseCloseCode": <caseCloseCodeUid>
    }
}
```

- "uid" (mandatory): Uid of the case that should be transitioned.
- "caseTransition" (mandatory): Uid of the transition that should be performed on the case.
- "justification" (optional): The justification (as a string) to be used for the case transition. It is mandatory if required by the case transition.
- "comment" (optional): A description to be used for the transition. It is mandatory if required by the case transition.
- "caseCloseCode" (optional): Uid of the case close code to be used. It is mandatory if the case transitions to a closed state.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

74 ExecuteExternalQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeExternalQuery",
    "uid": <externalQueryUid>,
    "data": {
        "caseUid": <caseUid>
    }
}
```

- "uid" (mandatory): Uid of the external query to be executed.
- "data:caseUid" (mandatory): Uid of the case, from which the external query is executed.

Returns

A HTTP response with the following content:

```
{
    "externalQuery": "<externalQueryName>",
    "values": [ ... ],
    "overwriteValues": 0|1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

75 ExecuteGroupByQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeGroupByQuery",
    "uid": <groupByQueryUid>
}
```

- "uid" (mandatory): Uid of the groupByQuery to be executed.

Returns

A HTTP response with the following content:

```
{
    "resultId": <resultId>,
    "groupByQueryName": "<groupByQueryName>",
    "queryUid": <groupByQueryUid>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

76 ExecuteJob

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeJob",
    "uid": <uid>
}
```

- "uid" (mandatory): Uid of the job which should be started.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

77 ExecuteQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeQuery",
    "uid": <queryUid>,
    "type": <type>,
    "revision": <revisionUid>
    "data": {
        "parameter": "<queryIndexValue>",
        "urid": <urid>,
        "indexValues": ["<queryIndexValue>"],
        "dataExport": true|false,
        "ignoreNumberOfRecordsInDataExport": true|false,
        "dataExportType": "DataExportTypeCsv",
        "exportPath": "<occUid>",
        "exportFileName": "<exportFileName>",
        "salt": "<salt>",
        "encryptedAttributesExportOptions": [
          {
            "uid": <attributeUid>,
            "values": {
              "plain": true|false,
              "mask": true|false,
              "hash": true|false
            }
          },
          ...
        ]
    }
}
```

- "uid" (mandatory): Uid of the query to be executed.
- "type" (mandatory): The type of the Query to execute. Available values for the "type" are: "query", "commonPointQuery", "simulationQuery".
- "revision" (optional): Used by simulation queries. Uid of the revision running a simulation in which to execute the simulation query.
- "data" (mandatory): Depending on "type" different values have to be set.
- "data:parameter" (optional): Used by investigation and simulation queries. It holds the index value to run the query on.
- "data:urid" (optional): Used by investigation queries.
- "data:indexValues" (optional): Used by common point queries.
- "data:dataExport" (optional): Specifies whether or not the query should run in export mode. The default value is "false".
- "data:ignoreNumberOfRecordsInDataExport" (optional): Only relevant for data exports. If set to true, record limits of query will be ignored.
- "data:dataExportType" (optional): Only relevant for data exports. Currently only "DataExportTypeCsv" is supported.
- "data:exportPath" (optional): Only relevant for data exports. Specify uid of an OCC with target type "File".
- "data:exportFileName" (optional): Only relevant for data exports. Name of export file. Default is {dateIso}.csv.
- "data:salt" (optional): Only relevant for data exports. At least 32 characters long salt which will be used for hashing encrypted attributes (if any).
- "data:encryptedAttributesExportOptions" (optional): Only relevant for data exports. An array of export settings per attribute specifying which of plain, masked or hashed versions should be written to disk.

Returns

A HTTP response with the following content:

```
{
    "resultId": <resultId>,
    "queryUid": <queryUid>,
    "attributes": <attributes>,
```

```
    "parameter": "<parameter>",
    "format": "<format>"|"None",
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

- "attributes": Only present after executing a common point query.
- "parameter": For simulation queries contains the parameter the query was called with, for investigation or common point queries contains the index value on which the query was run.

78 ExecuteReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeReport",
    "uid": <reportUid>
}
```

- "uid" (mandatory): The Report to execute.

Returns

A HTTP response with the following content:

```
{
    "reportResult": [
        ...
    ],
    "type": <type>,
    "timeFrom": <fromTimestamp>,
    "timeTo": <toTimestamp>,
    "generatedOn": <generatedOnTimestamp>,
    "generatedBy": "<generatedByUserName>",
    "usedMandators": ["<mandatorName>", ...],
    "name": "<reportName>",
    "comment": "<comment>",
    "uid": <reportUid>,
    "reportingAttributesCondtions": "<conditions>",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

79 ExecuteReportingQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeReportingQuery",
    "uid": <reportingQueryUid>
}
```

- "uid" (mandatory): Uid of the reportingQuery to be executed.

Returns

A HTTP response with the following content:

```
{
    "resultId": <resultId>,
    "reportingQueryName": "<reportingQueryName>",
    "queryUid": <reportingQueryUid>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

80 ExecuteRuleQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeRuleQuery",
    "revision": <revisionUid>,
    "data": {
        "parameter": "<ruleName>",
        "ruleUid": <ruleUid>
    }
}
```

- "revision" (mandatory): Uid of the revision running a simulation in which to execute the simulation query.
- "data" (mandatory): Contains either "ruleUid" or "parameter" to identify the simulation query by its uid or name respectively.

Returns

A HTTP response with the following content:

```
{
    "resultId": <resultId>,
    "queryUid": <queryUid>,
    "parameter": "<ruleName>",
    "format": "None",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

81 ExecuteRuleReportQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "executeRuleReportQuery",
    "revision": <revisionUid>,
    "data": {
        "parameter": "<ruleName>",
        "ruleUid": <ruleUid>
    }
}
```

- "revision" (mandatory): Uid of the revision that runs a simulation in which to execute the simulation query.
- "data" (mandatory): Either "ruleUid" or "parameter" have to be set.
- "data:parameter" (optional): The name of the rule the SimulationQuery should run on.
- "data:ruleUid" (optional): Uid of the rule the simulation query should run on.

Returns

A HTTP response with the following content:

```
{
    "resultId": <resultId>,
    "queryUid": <queryUid>,
    "parameter": "<ruleName>",
    "format": "<format>"|"None",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

82 ExportMultipleRules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "exportMultipleRules",
    "type": "firing" | "intercept" | "alarm" | "notification" | "reminder",
    "revision": <revisionUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "analysisId": <analysisId>
    }
}
```

- "type" (mandatory): The type of rule overlap table to export.
- "revision" (mandatory): Uid of the revision running an analysis.
- "data:downloadToken" (mandatory): The downoadToken of the user. Sent by the user interface automatically.
- "data:analysisId" (mandatory): Id of the analysis to retrieve data from.

Returns

A csv file with the requested data

83 ExportOptimizationAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "exportOptimizationAnalysis",
    "revision": <revisionUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision that runs an analysis.
- "data:downloadToken" (mandatory): The downoadToken of the user. Sent by the user interface automatically.
- "data:analysisId" (mandatory): Id of the analysis to retrieve data from.

Returns

A csv file with the requested data

84 ExportQueryResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "exportQueryResults",
    "uid": <queryUid>,
    "revision": <revisionUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the query from which to export results.
- "revision" (optional): Uid of the revision that runs a simulation. Only needed for simulation queries.
- "data:downloadToken" (mandatory): The downoadToken of the user. Sent by the user interface automatically.
- "data:resultId" (mandatory): Id of the query result to export.

Returns

A csv file with the requested data

85 ExportRuleStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "exportRuleStatistics",
    "revision": <revisionUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "analysisId": <analysisId>,
        "rulesets": [<rulesetId1>, <rulesetId2>, ...]
    }
}
```

- "revision" (mandatory): Uid of the revision that runs an analysis.
- "data:downloadToken" (mandatory): The downoadToken of the user. Sent by the user interface automatically.
- "data:analysisId" (mandatory): Id of the analysis to retrieve data from.
- "data:rulesets" (mandatory): Ids of the rulesets to include in the export.

Returns

A csv file with the requested data

86 FixMillisecondsMapping

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "fixMillisecondsMapping",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "uid": <attributeUid>
}
```

- "mandator" (optional): Uid of the mandator of the revision that owns the attribute.
- "revision" (mandatory): Uid of the revision that owns the attribute.
- "uid" (mandatory): Uid of the attribute that should be changed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

87 GenerateRulesUndoRule

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "generateRulesUndoRule",
    "revision: " <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision running a rule generation in which the conditions should be discarded.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

88 Get

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "get",
    "uid": <elementUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the element to retrieve.
- "mandator" (optional): Uid of the mandator to look in. Usage depends on type of element.
- "revision" (optional): Uid of the revision to look in. Usage depends on type of element.

Returns

A HTTP response with the following content:

```
{
    "data": {
        ...
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

89 GetActivatedComplianceLists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getActivatedComplianceLists",
    "mandator": <mandatorUid>
}
```

- "mandator" (optional): Uid of the mandator being used to check the role privilege if the requesting user does not have the global one.

Returns

A HTTP response with the following content:

```
{
    "activatedLists": {
        "PoliticalExposedPersonList": true|false,
        "OfacSdn": true|false,
        "UnitedNationsList": true|false,
        "GlobalWatchList": true|false,
        "EuropeanTerrorList": true|false,
        "RussianTerrorList": true|false
    },
    "currentlyReloading": true|false,
    "readonly": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

90 GetAdHocCheckComplianceListEntriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAdHocCheckComplianceListEntriesTable",
    "data": {
        "selectedLists": [<complianceListUid>],
        "name": "<name>",
        "street": "<street>",
        "city": "<city>",
        "country": "<country>",
        "passport": "<passport>",
        "birthdate": "<birthDate>"
    }
}
```

- "data:selectedLists": The selected compliance lists.
- "data:name": The name of the user.
- "data:street": The street of the user.
- "data:city": The city of the user.
- "data:country": The country of the user.
- "data:passport": The passport number of the user.
- "data:birthdate": The birthdate of the user.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        {
            "id": <listId>,
            "type": "<typeOfHit>",
            "entityType": "<entityType>",
            "entryUniqueId": <entryUniqueId>,
            "name": "<name>",
            "streetAddress": "<street>",
            "city": "<city>",
            "country": "<country>",
            "blacklistType": "<blackListType>",
            "passportNumber": "<passportNumber>",
            "birthdate": "<birthDate>",
            "title": "<title>",
            "otherInformation": "<otherInformation>",
            "negativeNewsAlert": true|false,
            "computedScore": <computedScore>,
            "blacklistHitType": "<blackListHitType>"
        },
        ... ...
    ],
    "resultId": <resultId>,
    "numberOfEntriesShown": <numberOfEntriesShown>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

91 GetAggregatedAuditTrails

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAggregatedAuditTrails",
    "uid": <caseUid>,
    "data": {
        "attributeUid": <reportingAttributeUid>
    }
}
```

- "uid" (mandatory): Uid of the investigation case whose audit trail is requested.
- "data:attributeUid" (mandatory): Uid of the reporting attribute for which the aggregated audit trail is requested.

Returns

A HTTP response with the following content:

```
{
    "auditTrails": [
        {
            "UTC": <timestamp>,
            "caseQueueId": <caseQueueId>,
            "caseQueueName": "<caseQueueName>",
            "caseUid": <caseUid>,
            "hits": <numberOfHits>,
            "score": <score>,
```

```
            "status": "<status>",
            "caseActionName": "<caseActionName>",
            "attachmentName": "<attachmentName>",
            "user": "<userName>",
            "userUid": <userUid>,
            "followUpUser": "<followUpUser>",
            "followUpTimestamp": <followUpTimestamp>,
            "comment": "<comment>",
            "cppName": "<cppName>",
            "caseCloseCode": "<caseCloseCode>",
            "fraudStatus": "<fraudStatus>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

92 GetAllProfilingsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAllProfilingsTable",
    "revision":<revisionUid>
}
```

- "revision" (mandatory): Uid of the revision to retrieve the profilings from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

93 GetAllRulesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAllRulesTable",
    "type": "rule" | "preprocessing" | "final",
    "revision": <revisionUid>,
    "data": {
        "filterRulesActive": "Both" | "Active" | "Inactive",
        "filterRulesInherited": "Both" | "Inherited" | "Own"
    }
}
```

- "type" (mandatory): The type of rules to retrieve.
- "revision" (mandatory): Uid of the revision where the all rules table is requested.
- "data:filterRulesActive" (optional): Retrieve only enabled or disabled rules, or both. (A rule is considered disabled, if its ruleset is disabled.)
- "data:filterRulesInherited" (optional): Retrieve only inherited rules or only rules owned by the given revision, or both.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <rulesetUid>,
            "<rulesetName>",
            <rulesetPriority>,
            <rulesetEnabled>"Yes" | "No",
            "<rulesetConditions>",
            <ruleUid>,
            <revisionUid>,
            <rulePriority>,
            "<ruleName>",
            "<ruleComment>",
            <ruleEnabled>"Yes" | "No",
            "<ruleConditions>",
            "<ruleConclusion>",
            "<actions>",
            "<mandatorName>",
            <mandatorUid>,
            <inherited>"Yes" | "No",
            <finalRuleset>"Yes" | "No"
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

94 GetAnalysedAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAnalysedAttributes",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision of the attribute analysis.

Returns

A HTTP response with the following content:

```
{
    "attributes": [
        {
            "uid": <attributeUid>,
            "name": "<attributeName>",
            "mandator": <mandatorUid>,
            "revision": <revisionUid>,
            "mdcStored": true|false,
            "origin": "<attributOrigin>",
            "type": "<attributeType>",
            "encrypted": true|false,
            "length": <attributeLength>,
            "decimals": <decimals>,
            "dimension": "<dimension>",
```

```
            "categoriesEnabled": true|false,
            "categories": [...],
            "usedInMergingConclusion": true|false,
            "metaAttribute": "<metaAttributeName>",
            "format": "<format>" | "None"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

95 GetAnalysisDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAnalysisDataSelection",
    "uid": <analysisId>,
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the analysis is defined.
- "data:analysisId" (mandatory): Uid of the analysis for which the data selection is requested.

Returns

A HTTP response with the following content:

```
{
    "analysis": {
        "enabled": true|false,
        "name": "<analysisName>",
        "comment": "<comment>",
        "optimization": true|false,
        "adoptSimulationDataSelection": true|false,
        "dataSelection": {
            "mandators": [<mandatorUid>],
            "periodType": "<periodType>",
            "actual": {
```

```
                "fromUrid": <fromUrid>,
                "fromRecord": <fromRecord>,
                "fromTimestamp": <fromTimestamp>,
                "fromDays": <fromDays>,
                "toUrid": <toUrid>,
                "toRecord": <toRecord>,
                "toTimestamp": <toTimestamp>,
                "toDays": <toDays>
            },
            "desired": {
                "fromUrid": <fromUrid>,
                "fromRecord": <fromRecord>,
                "fromTimestamp": <fromTimestamp>,
                "fromDays": <fromDays>,
                "toUrid": <toUrid>,
                "toRecord": <toRecord>,
                "toTimestamp": <toTimestamp>,
                "toDays": <toDays>
            },
            "incDdc": true|false,
            "conditions": [<conditions>]
        }
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

96 GetAnalysisProgress

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAnalysisProgress",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analysis is defined.
- "data:analysisId" (mandatory): Id of the analysis for which the progress is requested.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "simulationValid": true|false,
        "simulationEnabled": true|false,
        "isDataInIris": true|false,
        "simulationStatus": "<simulationStatus>",
        "simulationProgress": <simulationProgress>,
        "progress": <progress>,
        "recordsProcessed": <recordsProcessed>,
```

```
        "optimizationProgress": <optimizationProgress>,
        "manRecordsProcessed": <manRecordsProcessed>,
        "finished": true|false,
        "optimizing": true|false,
        "uridFrom": <uridFrom>,
        "uridTo": <uridTo>,
        "refresh": <autoRefreshSetting>,
        "analysisResultsValid": true|false,
        "analysisStatus": "<analysisStatus>"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

97 GetAnalysisTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAnalysisTable",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision for which the analyses table is requested.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <analysisId>,
            <enabled>true|false,
            "<name>",
            "<comment>",
            <ruleOptimization>true|false,
            "<currentStatus>"
        ],
        ... ...
    ],
    "refresh": <autoRefreshSetting>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

98 GetAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAttributes",
    "uid": <uid>,
    "type": <type>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": { "index": <indexUid> }
}
```

- "uid" (optional): Uid of an monitoring list, index based evaluation or attribute, depending on the type.
- "type" (optional): Type of the request that filters the attribute for some specific use case. The type defines, which other elements of the requests are mandatory and which internal logic is used to filter the attributes. Possible types are:

```
"compromisedConditionsCollusions"
"complianceList"
"fraudFlaggingTimestamp"
"doubletDetection"
"mandator"
"query"
"groupByQuery"
"definedRiskListImport"
"definedRiskList"
"definedRiskListRuleAction"
"definedRiskListImport"
"aPrioriConditions"
"counterEvaluation"
"counterEvaluationIncludingCurrent"
```

```
"counterReference"
"counterReferenceIncludingCurrent"
"mergingConclusion"
"mergingTarget"
"mergingTermination"
"deviceIdentification"
"deviceIdentificationInputs"
"eventIncludingCurrent"
"indexComputation"
"indexInsertion"
"complianclisteList"
"mergingSource"
"pattern"
"profilePre"
"ruleConclusion"
"event"
"ruleset"
"ruleConclusionExpression"
"ruleCondition"
"ruleAction"
"profilePost"
"modelTrainingDataSelection"
"modelVerificationDataSelection"
"connivanceConditionsCollusions"
"counterpartyConditionsCollusions"
"stencil"
"precedent"
"precedentsReference"
"queryDataSelection"
"groupByQueryDataSelection"
"analysisDataSelection"
"simulationQuery"
"simulationQueryDataSelection"
"simulation"
"stencil"
"precedent"
"precedentsReference"
"caseQueue"
```

```
        "createRulesScoring"
        "definedRiskInput"
        "definedRiskOutput"
        "reporting"
        "formula"
        "transaction"
        "mergingConclusionExpression"
        "reportingQuery"
```

- "mandator" (optional): Uid of the mandator.
- "revision" (optional): Uid of the revision.
- "data: index" (optional): Uid of the index that is used for specific request types.

Returns

A HTTP response with the following content:

```
{
    "attributes": [{
        "uid": <uid>,
        "name": <name>,
        "type": "Numeric" | "Text" | "Hexadecimal" | "Boolean" | "IP" | ...,
        "metaAttribute": <metaAttributeType>,
        "mdcStored": true|false},
        ....
        }, ...],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

99 GetAuditLogTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getAuditLogTable",
    "data": {
        "from": <fromTimestamp>,
        "to": <toTimestamp>,
        "filter": ["A","C","D","X","E","F","I","N","W"],
        "instances": [<instanceUids>],
        "users": [<userUids>],
        "lastEntry": -1|<maxTimestamp>
    }
}
```

- "data:from" (mandatory): The start time to filter the audit log entries.
- "data:to" (mandatory): The end time to filter the audit log entries.
- "data:filter" (mandatory): A concatenated string containing the log levels to filter the audit log entries.
- "data:instances" (mandatory): The IrisInstances (identified by their uids) to filter the audit log entries.
- "data:users" (mandatory): The Users (identified by their uids) to filter the audit log entries.
- "data:lastEntry" (mandatory): The maximum timestamp of an audit log entry to be included or -1 for all entries.

Returns

A HTTP response with the following content:

```
{
    "recordLimit": <recordLimitSetting>,
    "tableContent": [...],
    "from": <fromTimestamp>,
    "to": <toTimestamp>,
    "synced": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

100 GetCaseAction

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseAction",
    "uid": <caseActionUid>,
    "mandator": <mandatorUid>,
    "data": {
        "caseUid": <caseUid>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "caseAction": {
    "uid": <caseActionUid>,
    "enabled": true|false,
    "mandator": {
        "uid": <mandatorUid>,
        "name": "<mandatorName>",
        "maxMemoryForSimulationGB": <maxMemoryForSimulationInGB>
    },
    "name": "<caseActionName>",
    "comment": "<comment>",
    "targetType": "<targetType>",
    "messageAddress": "<messageAddress>",
```

```
    "messagePort": <messagePort>,
    "userName": "<userName>",
    "contentType": "<contentType>",
    "targetPath": "<targetPath>",
    "useSsl": true|false,
    "certificatePath": "<certificatePath>",
    "useClientCertificate": true|false,
    "clientCertificatePath": "<clientCertificatePath>",
    "clientCertificateKeyPath": "<clientCertificateKeyPath>",
    "clientCertificateKeyType": "<clientCertificateKeyType>",
    "clientCertificateKeyPasswordPath": "<clientCertificateKeyPasswordPath>",
    "basicAuthUsername": "<userName>",
    "basicAuth": true|false,
    "fromAddress": "<fromAddress>",
    "addressType": "<addressType>",
    "constantToAddress": "<constantToAddress>",
    "formatAttributeValues": true|false,
    "maskAttributeValues": true|false,
    "subjectTemplate": "<template>",
    "bodyTemplate": "<template>",
    "messageTemplate": "<template>",
    "attachment": {
        "name": "<name>",
        "comment": "<comment>",
        "size": <size>,
        "lastChangedBy": "<lastChangedByUserName>",
        "lastChangedOn": <lastChangedOnTimestamp>,
        "data": "<data>"
    },
    "lastChangedBy": "<lastChangedByUserName>",
    "lastChangedTimestamp": <lastChangedTimestamp>,
    "emailTo": "<emailTo>"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

101 GetCaseActionPreview

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseActionPreview",
    "uid": <caseUid>,
    "mandator": <caseMandatorUid>,
    "data": {
        "caseActionUid": <caseActionUid>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "preview": {
        ... ...
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

102 GetCaseActions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseActions",
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "caseActions": [
        {...},
        {...},
        ... ...
    ],
    "responseStatus": ["<status>", "<feedback>"],
    "reloadUserProfile": true|false
}
```

103 GetCaseActionsFromCaseQueue

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseActionsFromCaseQueue",
    "uid": <caseUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "caseActions": [
    {
        "uid": <caseActionUid>,
        "enabled": true|false,
        "name": "<caseActionName>",
        "targetType": "<targetType>"
    },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

104 GetCaseActionsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseActionsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [... ...],
        [... ...],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

105 GetCaseCloseCodes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseCloseCodes",
    "type": "bulkClose" | "",
    "data": {
        "transitionUid": <transitionUid>,
        "mandators": [<mandatorUid>,...]
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "caseCloseCodes": [
        {
            "uid": <caseCloseCodeUid>,
            "name": "<name>",
            "comment": "<comment>",
            "caseActions": [
            ... ...
        ],
            "fraudStatus": <fraudStatus>,
            "lastChangedBy": "<lastChangedByUserName>",
            "lastChangedTimestamp": <lastChangedTimeStamp>
        },
```

```
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

106 GetCaseCloseCodesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseCloseCodesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            ... ....
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

107 GetCaseData

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseData",
    "uid": <caseUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "caseData": [
        {
            "uid": <caseUid>,
            "Queue": "<caseQueueName>",
            "Generated": <generatedAtTimestamp>,
            "Score": <score>,
            "Hits": <numberOfHits>,
            "Status": "<status>",
            "FraudStatus": "<fraudStatus>",
            "CloseCodeName": "<caseCloseCodeName>",
            "User": <userUid>,
            "UserName": "<userName>",
            "LastUserAction": <lastUserActionTimestamp>,
            "MandatorName": "<mandatorName>",
            "Mandator": <mandatorUid>,
            "CppName": "<cppName>",
```

```
            "Cpp": "<cpp>",
            "<attributeUid>": "<attributeValue>",
            ... ...
        }
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

108 GetCaseGroupDefinitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseGroupDefinitions"
}
```

Returns

A HTTP response with the following content:

```
{
    "casaeGroups": [
        {
            "streamtype": "caseGroup",
            "uid": <caseGroupUid>,
            "name": "<caseGroupName>",
            "comment": "<comment>",
            "mandatorUid": <mandatorUid>,
            "inherited": true|false,
            "enabled": true|false,
            "evaluationAttributeUid": <evaluationAttributeUid>,
            "lastUsedId": <lastUsedId>,
            "cpps": [ ... ]
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

109 GetCaseGroupsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseGroupsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <caseGroupUid>,
            <mandatorUid>,
            "<mandatorName>",
            "<caseGroupName>",
            "<comment>",
            <enabled>"Yes" | "No",
            <inheritToSubMandators>"Yes" | "No",
            "<evaluationAttributeName>"
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

110 GetCaseHistory

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseHistory",
    "uid": <caseUid>,
    "data": {
        "attributeUid": <reportingAttributeUid>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "header": [
        {
            "uid": "<uid>",
            "name": "<name>",
            "align": "<align>",
            "type": "<type>",
            "format": "<format>"
        },
        ... ...
    ],
    "tableContent": [
        {
            "uid": <caseUid>,
```

```
            "Queue": "<caseQueueName>",
            "Generated": <generatedOnTimestamp>,
            "Score": <score>,
            "Hits": <numberOfHits>,
            "Status": "<status>",
            "FraudStatus": "<fraudStatus>",
            "CloseCodeName": "<closeCodeName>",
            "User": <userUid>,
            "UserName": "<userName>",
            "LastUserAction": <lastUserActionTimestamp>,
            "MandatorName": "<mandatorName>",
            "Mandator": <mandatorUid>,
            "CppName": "<cppName>",
            "Cpp": "<cpp>",
            "<attributeUid>": "<attributeValue>",
            ... ...
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

111 GetCaseQueueReportsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseQueueReportsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <caseQueueReportUid>,
            "<mandatorName>",
            "<reportName>",
            "<comment>",
            "<lastChangedByUserName>",
            "<lastChangedTimestamp>"
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

112 GetCaseQueues

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseQueues",
    "type": "<type>",
    "mandator": <mandatorUid>
}
```

- "type": Available values for this parameter are: "caseFilter", "kpiFilter", "conclusion", "query", "collusion" (requires "revision"), "caseCreation", "report", "user↩ Report", "ruleAction".
- "mandator": Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "caseQueues": [
        {
            "uid": <caseQueueUid>,
            "id": <id>,
            "name": "<caseQueueName>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

113 GetCaseQueuesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseQueuesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            ... ...
        ],
        ... ...
    ],
    "responseStatus": ["<status">, "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

114 GetCasesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCasesTable",
    "data": {
        "mandators": [
            <mandatorUid>,
            ... ...
        ],
        "caseQueues": [
            <caseQueueUid>,
            ... ...
        ],
        "dateRestriction": {
            "from": <fromTimestamp>,
            "to": <toTimestamp>
        },
        "status": {
            "New": true|false,
            "Investigated": true|false,
            "Followup": true|false,
            "Due": true|false,
            "Closed": true|false,
            "Reopened": true|false
        },
        "users": [<userUid>],
        "scoreFrom": <caseScoreFromValue>,
        "scoreTo": <caseScoreToValue>,
        "cpps": ["<cppUid_caseGroupUid>", ...]
    }
```

Returns

A HTTP response with the following content:

```
{
    "header": [
        {
            "uid": "<uid>",
            "name": "<name>",
            "align": "<align>",
            "type": "<type>",
            "format": "<format>"
            "dimension": "<dimension>"
        },
        ... ...
    ],
    "tableContent": [
        {
            "uid":<caseId>,
            "Queue": "<caseQueueName>",
            "Generated": <generatedTimestamp>,
            "Score": <scoreValue>,
            "Hits": <numberOfHits>,
            "Status": "<status>",
            "FraudStatus": "<fraudStatus>",
            "CloseCodeName": "<caseCloseCodeName>",
            "User": <userUid>,
            "UserName": "<userName>",
            "LastUserAction": <lastUserActionTimestamp>,
            "MandatorName": "<mandatorName>",
            "Mandator": <mandatorUid>,
            "CppName": "<cppName>",
            "Cpp": "<cppCode>",
            ... ...
        },
        ... ...
    ],
```

```
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

115 GetCaseStates

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseStates",
    "uid": <targetCaseQueueUid>,
    "type": "caseQueue"|"kpiFilter"|"sai"|"caseFilter"|"caseWorkflow",
    "mandator": <mandatorUid>,
    "data": {
        "mandators": [<mandatorUid>,...]
    }
}
```

- "uid": This is required only if the "type" is "caseQueue".
- "mandator": This is required only if the "type" is "caseWorkflow".
- "data:mandators": This is required only if the "type" is "kpiFilter" or "sai".

Returns

A HTTP response with the following content:

```
{
    "caseStates": [
        {
            "uid": <caseStateUid>,
            "name": "<caseStateName>",
            "exclusiveState": true|false,
```

```
            "meta": "<meta>"
        },
        ... ...
    ],
    "responseStatus": ["<status">, "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

116 GetCaseStatesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseStatesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <caseStateUid>,
            <mandatorUid>,
            "<mandatorName>",
            "<caseStateName>",
            "<comment>",
            "<lastChangedByUserName>",
            <lastChangedOnTimestamp>,
            <defaultState>"Yes"|"No"
        ],
        ... ...
    ],
    "responseStatus": ["<status">, "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

117 GetCaseTransitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
There are two ways to run this request.

Option 1:
    {
        "request": "getCaseTransitions",
        "uid": <caseUid>
    }

Option 2:
    {
        "request": "getCaseTransitions",
        "type": "caseSelection",
        "data": {
            "cases": [<caseUid>,...]
        }
    }
```

Returns

A HTTP response with the following content:

```
{
    "caseTransitions": [
        {
            "uid": <caseTransitionUid>,
            "name": "<caseTransitionName>",
```

```
            "justificationCodes": [<justificationCode>,...],
            "justificationMandatory": true|false,
            "commentAllowed": true|false,
            "commentMandatory": true|false,
            "targetStateExclusive": true|false,
            "targetStateMeta": "<targetStateMeta>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

118 GetCaseVariablesAsConditions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseVariablesAsConditions"
}
```

Returns

A HTTP response with the following content:

```
{
    "caseVariables": [
        {
            "caseVariable": true|false,
            "uid": "<uid>",
            "name": "[<name>]",
            "type": "<type>",
            "metaAttribute": "None"|"<metaAttribute>",
            "categories": [ ... ]
        },
        ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

119 GetCaseWorkflows

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseWorkflows",
    "type": "caseQueue",
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "caseWorkflows": [
        {
            "uid": <caseWorkflowUid>,
            "name": "<caseWorkflowName>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

120 GetCaseWorkflowsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCaseWorkflowsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <caseWorkflowUid>,
            <mandatorUid>,
            "<mandatorName>",
            <enabled>"Yes"|"No",
            "<caseWorkflowName>",
            "<comment>",
            "<lastChangedByUserName>",
            <lastChangedOnTimestamp>,
            <defaultWorkflow>"Yes"|"No",
            <numberOfStates>,
            <numberOfTransitions>
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

121 GetChartColorSchema

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getChartColorSchema"
}
```

Returns

A HTTP response with the following content:

```
{
    "data": {
        "chartColorSchema": {
            "colorSchemaFraudulentValues": "<colorHex>,<colorHex>,<colorHex>,<colorHex>,<colorHex>",
            "colorSchemaGenuineValues": "<colorHex>,<colorHex>,<colorHex>,<colorHex>,<colorHex>",
            "colorSchemaOtherValues": "<colorHex>,<colorHex>,<colorHex>,<colorHex>,<colorHex>"
        }
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

122 GetCharts

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCharts"
}
```

Returns

A HTTP response with the following content:

```
{
    "charts": [
        {
            "streamType": "chart",
            "uid": <chartUid>,
            "position": <position>,
            "name": "<chartName>",
            "comment": "<comment>",
            "mandator": {
            "uid": <mandatorUid>,
            "name": "<mandatorName>",
            "maxMemoryForSimulationGB": <maximumMemoryForSimulationInGB>
        },
        "explanation": "<explanation>",
        "tooltip": "<tooltip>",
        "onlineHelpType": "NoOnlineHelp" | "DefaultOnlineHelp" | "CustomAndDefaultOnlineHelp" | "CustomOnlineHelp",
        "onlineHelp": "<onlineHelp>"
        },
```

```
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

123 GetChartsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getChartsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [... ...],
        ... ...
    ],
    "nextPosition": <nextPosition>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

124 GetCollusions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCollusions",
    "revision": <revisionUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "collusions": [
        {
            "uid": <collusionUid>,
            "name": "<collusionName>",
            "comment": "<comment>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

125 GetCollusionSimulationResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCollusionSimulationResults",
    "uid": <collusionUid>,
    "revision": <revisionUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "simulationResults": [<simulationResults>],
    "progress": <progress>,
    "simulationFinished": true|false,
    "simulationRunning": true|false,
    "refresh": <autoRefreshSetting>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

126 GetCollusionsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCollusionsTable",
    "type": "own" | "inherited",
    "revision": <revisionUid>
}
```

- "type" (mandatory): Decide whether to get collusions of own revision or inherited mandators.
- "revision": Uid of the revision to get collusions for.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <revisionUid>,
            <collusionUid>,
            "<collusionName>",
            "<comment>",
            "<mandatorName>",
            "<firstPartyName>",
            "<counterPartyName>",
            <counterPartyTimeFrom>,
```

```
            <counterPartyTimeTo>,
            "<counterPartyTimeUnit>",
            <connivanceTimeBefore>,
            <connivanceTimeAfter>,
            "<connivanceTimeUnit>"
            "<caseQueueName>",
            <thresholdFirstParties>,
            "<counterPartyConditions>",
            "<connivanceConditions>",
            "<compromisedConditions>",
            []
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

127 GetCommonPointQueriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCommonPointQueriesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <commonPointQueryUid>,
            <mandatorUid>,
            "<queryType>",
            <indexUid>,
            "<indexName>",
            "<mandatorName>",
            "<commonPointQueryName>",
            "<comment>",
            <enabled>"Yes" | "No",
            <columnsForCommonPoint>,
            "<lastChangedByUserName>",
            <lastChangedTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

128 GetCommonQuerySubsetResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCommonQuerySubsetResults",
    "uid": <commonQueryUid>,
    "data": {
        "resultId": <resultId>,
        "urid" : [<urid>]
    }
}
```

- "uid" (mandatory): Uid of the common points query that runs.

Returns

A HTTP response with the following content:

```
{
    "queryResult": {
    "name": "<commonPointQueryName>",
    "comment": "<commonPointQueryComment>",
    "header": [{
        "name": "<attributeName>",
        "attributeUid": <attributeUid>,
        "align": "<align>",
        "decimals": <decimals>,
        "hyperlinkQueryUid": [{
```

```
    "name": "<hyperlinkQueryName>",
    "comment": "<comment>",
    "uid": <hyperlinkQueryUid>
    }
],
"commonPointQueryUid": [{
    "name": "<commonPointQueryName>",
    "comment": "<commonPointQueryComment>",
    "uid": <commonPointQueryUid>
    }
],
"type": "<attributeType>",
"format": "<attributeFormat>"
},
... ...
],
"tableContent": [
... ...
]
},
"indexValueUrid": <indexValueUrid>|-1,
"running": true|false,
"recordsTotal": <recordsTotal>,
"recordsFinished": <recordsFinished>,
"duration": <duration>,
"amountDimension": "<amountDimension>",
"statistics": {
    "transaction": {
        "genuine": {
            "records": <numberOfRecords>,
            "amount": <amount>
        },
        "fraud": {
            "records": <numberOfRecords>,
            "amount": <amount>
        }
    }
},
```

```
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

129 GetComplianceDataSourceExists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getComplianceDataSourceExists"
}
```

Returns

A HTTP response with the following content:

```
{
    ["euslPath" | "ruslPath" | "ofacPath" | "unPath" | "pepPath"]: true|false,
    "allDataSourcesAvailable": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

130 GetComplianceDefinitionTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getComplianceDefinitionTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <complianceListUid>,
            <mandatorUid>,
            "<mandatorName>",
            "<complianceListName>",
            "<comment>",
            <priority>,
            "<outputAttribute>",
            "<typeOfList>",
            <searchTreeSize>,
            "<lastChangedByUserName>",
            <lastChangedOnTimestamp>,
            <enabled>["Yes" | "No"]
        ],
    ... ...
    ],
    "currentlyReloading": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

131 GetComplianceListDefinitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getComplianceListDefinitions"
}
```

Returns

A HTTP response with the following content:

```
{
    "complianceLists": [
        { ... ... },
        ... ...
    ],
    "currentlyReloading": true|false,
    "readonly": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

132 GetConclusionExpressionValuePairs

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getConclusionExpressionValuePairs",
    "uid": <uid>,
    "data": {
        "expressionAttribute": <attributeUid>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "values": [<values>],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

133 GetCountersTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCountersTable",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision the counters are to be fetched for.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
    [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

134 GetCpp

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCpp",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision the CPP is fetched from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

135 GetCpps

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCpps",
    "uid": <caseUid>,
    "type": "caseCreation" | "caseInvestigation" | "caseFilter",
    "mandator": <mandatorUid>
}
```

- "uid" (optional): Uid of the investigation case.
- "type" (mandatory): Type of the section that the reqeust shall be sent.
- "mandator" (optional): Uid of mandator, if the request is sent from case creation or case investigation.

Returns

A HTTP response with the following content:

```
{
    "selectedCpp": <selectedCpp>,
    "selectedCaseGroup": <selectedCaseGroup>,
    "cpps": [{
        "streamType": "cpp",
        "caseGroupUid": <caseGroupUid>,
        "mandatorUid": <mandatorUid>,
        "id": <cppId>,
        "name": "<cppName>",
```

```
        "status": "<status>",
        "comment": "<comment>",
        "reportingAttributes": [{
            "name": "<attributeName>",
            "type": "<type>",
            "format": "<format>",
            "attributeId": <attributeUid>,
            "encrypted": true|false,
            "value": <value>,
            "decimals": <decimals>,
            "category": [<category>]
            },
        ... ...
        ],
        "createdBy": "<createdByUserName>",
        "createdTimestamp": <createdTimestamp>,
        "lastChangedBy": "<lastChangedByUserName>",
        "lastChangedTimestamp": <lastChangedTimestamp>,
        "active": true|false,
        "inherit": true|false
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

136 GetCppsToHighlight

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCppsToHighlight",
    "uid": <queryUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the query to be highlighted.
- "mandator" (mandatory): Uid of the mandator the CPPs are requested for. Head mandator CPPs will also be returned in case their case group inherits.

Returns

A HTTP response with the following content:

```
{
    "reportingAttributeValues" : [
    {
        <attributeUid>: [
            <reportingAttributeValues>
        ],
        ...
    },
    ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

137 GetCppTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getCppTable",
    "data": {
        "showAllEntries": true|false,
        "caseGroups": [<caseGroupUid>],
        "dateRestriction": {
            "all": true|false,
            "from": <fromTimestamp>,
            "to": <toTimestamp>
        },
        "status": {
            "New": true|false,
            "Investigated": true|false,
            "Followup": true|false,
            "Due": true|false,
            "Closed": true|false,
            "Reopened": true|false
        },
        "active": {
            "Active": true|false,
            "Inactive": true|false
        }
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "header" : [
        {
            "uid" : "<uid>",
            "name" : "<name>",
            "align" : "<align>",
            "type" : "<type>",
            "decimals" : <decimals>,
            "format" : "<format>"
        },
    ... ...
    ],
    "tableContent" : [
        {
            "id" : <id>,
            "caseGroupUid" : <caseGroupUid>,
            "caseGroup" : "<caseGroupName>",
            "mandatorUid" : <mandatorUid>,
            "mandator" : "<mandatorName>",
            "name" : "<cppName>",
            "status" : "<status>",
            "comment" : "<comment>",
            "active" : "Yes" | "No",
            "inheritable" : "Yes" | "No",
            "createdBy" : "<createdByUserName>",
            "createdAt" : <createdAtTimestamp>,
            "lastChangedBy" : "<lastChangedByUserName>",
            "lastChangedAt" : <lastChangedAtTimestamp>,
            "numberOfAssociatedCases" : <numberOfAssociatedCases>,
            "numberOfAssociatedFraudulentCases" : <numberOfAssociatedFraudulentCases>,
            "numberOfAssociatedGenuineCases" : <numberOfAssociatedGenuineCases>,
            "rateOfFalseAlarmCases" : <rateOfFalseAlarmCases>,
            "evaluationAttribute" : {
                "value" : "<value>"
            },
            "<attributeUid>" : "<attributeValue>",
            ... ...
        },
```

```
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

138 GetDashboardCharts

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDashboardCharts"
}
```

Returns

A HTTP response with the following content:

```
{
    "dashboardCharts": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

139 GetDashboardKeyPerformanceIndicatorData

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDashboardKeyPerformanceIndicatorData",
    "uid": "<kpiUid>,
    "data": {
        "from": <fromTimestamp>,
        "to": <toTimestamp>,
        "instance": <instanceId>,
        "exportAll": true|false
    }
}
```

- "uid" (mandatory): Uid of the KPI for which data has to be retrieved.
- "data" (mandatory):
- "data:from": Data points that have been recorded after this Unix timestamp value will be returned.
- "data:to": Data points that have been recorded before this Unix timestamp value will be returned.
- "data:instance": Id of the instance.
- "data:exportAll": If set to "true", all data points in the data range will be exported. Otherwise, the maximum number of data points is 5000. Data points will be interpolated if set to "false" (default).

Returns

A HTTP response with the following content:

```
{
    "keyPerformanceIndicatorData": {
        "instance": <instanceId>,
        "timestamps": [ <x1>, <x2>, <x3>, ... , <xn> ],
        "values": [ <y1>, <y2>, <y3>, ... , <yn> ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

140 GetDashboardKeyPerformanceIndicators

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDashboardKeyPerformanceIndicators",
    "uid": <chartUid>
}
```

- "uid" (mandatory): Uid of the chart.

Returns

A HTTP response with the following content:

```
{
    "keyPerformanceIndicators": [
        {
        "uid": <kpiUid>,
        "position": <position>,
        "name": <kpiName>,
        "unit": "<unit>",
        "tooltip": "<tooltip>",
        "instance": <instanceId>,
        "autoScale": true|false,
        "minimum": <minimum>,
        "maximum": <maximum>,
        "showLine": true|false,
        "showBars": true|false,
```

```
        "showPoints": true|false
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

141 GetDefaultCapacities

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefaultCapacities"
}
```

Returns

A HTTP response with the following content:

```
{
    "ddcCapacity": <ddcCapacity>,
    "mdcCapacity": <mdcCapacity>,
    "mdcStored": <mdcStored>true|false,
    "ddcStored": <ddcStored>true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

142 GetDefaults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefaults"
}
```

Returns

A HTTP response with the following content:

```
{
    "data": {
        "streamType": "defaultSettings",
        "settings": {
            "application": {
                "name": "<applicationName>"
            },
            "pageAutoRefresh": {
                "dashboard": <valueInSeconds>,
                "investigation": <valueInSeconds>,
                "analysis": <valueInSeconds>,
                "jobSchedule": <valueInSeconds>,
                "elementGeneration": <valueInSeconds>,
                "golive": <valueInSeconds>,
                "cluster": <valueInSeconds>,
                "keys": <valueInSeconds>,
                "query": <valueInSeconds>,
                "memoryManagement": <valueInSeconds>,
```

```
                "sessionCountdown": <valueInSeconds>
            },
            "interfaces": {
                "applicationProgramming": {
                    "lockTimeoutSeconds": <lockTimeoutInSeconds>,
                    "lockTimeoutCountdownThresholdSeconds": <lockTimeoutCountdownThresholdInSeconds>,
                    "enableCsrf": true|false,
                    "useGzip": true|false,
                    "useKeepAlive": true|false,
                    "downloadRequests": true|false,
                    "defaultLanguage": "<defaultLanguage>",
                    "defaultDateTimeFormat": "ISO" | "US" | "EUR",
                    "defaultDecimal": "<defaultDecimalSeparator>",
                    "defaultDigitGroup": "<defaultDigitGroupSeparator>",
                    "defaultFieldSeparator": "<defaultFieldSeparator>",
                    "defaultExtendedSelectDialog": true|false,
                    "defaultSearchInSelections": true|false,
                    "defaultShowExplanation": true|false,
                    "defaultShowUids": true|false
                }
            }
        }
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

143 GetDefinableCaseTransitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinableCaseTransitions",
    "uid": <caseWorkflowUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "definableCaseTransitions": [
        {
            "uid": <caseTransitionUid>,
            "name": "<caseTransitionName>",
            "targetState": <caseStateUid>,
            "targetStateExclusive": true|false,
            "targetStateMeta": "<caseStateMeta>",
            "targetCaseQueue": "Current"|"Definable"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

144 GetDefinedRiskDefinitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskDefinitions",
    "type": "ruleAction" | "createEntryFromQuery",
    "mandator": <mandatorUid>
}
```

- "type" (optional): For "ruleAction" type, the "mandator" is required.

Returns

A HTTP response with the following content:

```
{
    "definedRisks": [
        {
            "streamType": "definedRiskList",
            "reencryptAuditTrailValues": true|false,
            "uid": <definedRiskListUid>,
            "name": "<definedRiskListName>",
            "comment": "<comment>",
            "priority": <priority>,
            "explanation": "<explanation>",
            "starts": true|false,
            "expires": true|false,
            "expiresByDefault": true|false,
```

```
            "defaultLifeTime": <defaultLifeTime>,
            "label": "<label>",
            "active": true|false,
            "lastUsedId": <lastUsedId>,
            "mandator": <mandatorUid>,
            "enableForRulAction": true|false,
            "ruleAction": {
                "comment": "<comment>",
                "label": "<label>",
                "active": true|false,
                "overwrite": true|false,
                "conditions": [<conditions>]
            }
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

145 GetDefinedRiskListAuditTrailTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListAuditTrailTable",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "definedRiskListName": "<definedRiskListName>",
    "auditTrail": [{
        "reencryptAuditTrailValues": true|false,
        "notificationEntry": true|false,
        "userName": "<userName>",
        "userUid": <userUid>,
        "createdByRuleAction": true|false,
        "utc": <timestamp>,
        "action": "<action>",
        "encrypted": true|false,
        "changedElementName": "<changedElementName>",
        "riskListName": "<riskListName>",
        "riskListUid": <riskListUid>,
        "comparisonResult": "<comparisonResult>",
        "type": "<riskListType>",
```

```
        "changes": [{
            ... ...
            }]
        }
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

146 GetDefinedRiskListDefinitionTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListDefinitionTable",
    "data": {
        "selectFilter": "all" | "active" | "inactive"
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "allowDelete": true|false,
    "tableContent": [[
        <definedRiskListUid>,
        <mandatorUid>,
        "<mandatorName>",
        "<definedRiskListName>",
        "<comment>",
        <priority>,
        "<inputAttributeName>",
        "<outputAttributeName>",
        "<outputAttributeValue>",
        <numberOfEntries>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>,
```

```
        <enabled>"Yes"|"No",
        <enabledForRuleActions>"Yes"|"No"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

The value for "allowDelete" is decided by the system setting "Permanent deletion of defined risk lists".

147 GetDefinedRiskListEntriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListEntriesTable",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "label": "<labelFilterValue>",
        "comment": "<commentFilterValue>",
        "value": "<riskListInputAttributeValue>",
        "showEnabled": true|false,
        "showDisabled": true|false,
        "showAllEntries": true|false,
        "lastChangeRestriction": {
            "all": false,
            "from": <lastChangedOnFrom>,
            "to": <lastChangedOnTo>
        },
        "validRestriction": {
            "all": false,
            "from": <startsAtFrom>,
            "to": <startsAtTo>
        },
        "expiresRestriction": {
            "all": false,
            "from": <expiresFrom>,
            "to": <expiresTo>
        },
        "users": [<userUid>,...],
        "ruleAction": true|false
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        "<inputAttributeValue>",
        "<entryId>",
        "<outputCategoryAttributeValue>",
        "<startsAt>",
        "<expiresAt>",
        "<comment>",
        "<lastChangedBy>",
        "<lastChangedOn>",
        "<filterCriteria>",
        "<label>",
        <enabled>"Yes" | "No"
        ],
    ... ...
    ],
    "name": "<definedRiskListName>",
    "attributeName": "<inputAttributeName>",
    "expiresEnabled": true|false,
    "expiredByDefault": true|false,
    "defaultLifeTime": <defaultLifeTime>,
    "startsAtEnabled": true|false,
    "explanation": "<explanation>",
    "inputAttributeIsEncrypted": true|false,
    "deleteEnabled":  true|false,
    "activateEnabled": true|false,
    "showAuditTrail": true|false,
    "numberOfEntriesShown": <numberOfEntriesShown>,
    "outputAttributeCategoriesEnabled": true|false,
    "outputAttributeIsEncrypted": true|false,
    "attributeFormat": "<attributeFormat>",
    "attributeType": "<attributeType>",
    "responseStatus": ["<status>", "<feedbacktext>"],
```

```
    "reloadUserProfile": true|false
}
```

148 GetDefinedRiskListEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListEntry",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "attributeValue": "<entryInputAttributeValue>",
        "id": <entryId>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "entry": {
        "value": "<value>",
        "attributeName": "<attributeName>",
        "attribute": {
            "type": "<type>",
            "format": "<format>" | "None",
            "decimals": <decimals>,
            "length": <length>
        },
        "outputAttributeCategoryValue": "<value>",
        "expiresAt": <expiresAtTimestamp>,
```

```
        "startsAt": <startsAtTimestamp>,
        "id": <entryId>,
        "comment": "<comment>",
        "label": "<label>",
        "active": true|false,
        "conditions": [ <conditions> ],
        "expires": true|false,
        "starts": true|false,
        "lastChangedBy": "<lastChangedByUserName>",
        "lastChangedTimestamp": <lastChangedTimestamp>
    },
    "canEdit": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

149 GetDefinedRiskListEntryAttribute

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListEntryAttribute",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "attributeName": "<inputAttributeName>",
    "attribute": {
        "type": "<attributeType>",
        "format": "<attributeFormat>",
        "decimals": <decimals>,
        "encrypted": true|false,
        "length": <attributeLength>
    },
    "outputAttributeCategoriesEnabled": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

150 GetDefinedRiskListPrivileges

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListPrivileges",
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "viewAuditTrail": true|false,
    "allowDelete": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

The value for "viewAuditTrail" is decided by "view defined risk list audit trail" privilege setting in user's role. The value for "allowDelete" is decided by "Permanent deletion of defined risk list" setting in System Configuration.

151 GetDefinedRiskListSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDefinedRiskListSettings",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "starts": true|false,
    "expires": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

152 GetDeviceIdentificationTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDeviceIdentificationTable",
    "type": "own" | "inherited",
    "revision": <revisionUid>
}
```

- "type" (mandatory): Choose "Own" to retrieve Device Identifations in the given revision. Choose "Inherited" to retrieve Device Identifications defined in head mandators.
- "revision" (mandatory): Uid of the revision to retrieve the Device Identifications from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <revisionUid>,
            <deviceIdentificationUid>,
            "<deviceIdentificationName>",
            "<comment>",
            "<mandatorName>",
            "<indexName>",
            "<timestampAttribute>",
```

```
            "<cookie>",
            "<fingerPrint>",
            "<conditions>",
            [
                "<attributeNameA>",
                "<attributeNameB>"
            ... ...
            ],
            [
                "<computationForAttributeA>",
                "<computationForAttributeB>",
                ... ...
            ],
            <numberOfAttributes>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

153 GetDonorInstances

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getDonorInstances",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the iris instance to restore.

Returns

A HTTP response with the following content:

```
{
    "donorInstances": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

154 GetElementPrintView

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getElementPrintView",
    "uid": <revisionElementUid>,
    "revision": <revisionUid>
}
```

- "uid": Uid of the revision element.
- "revision": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "uid": <revisionUid>,
        "mandator": <mandatorUid>,
        "mandatorName": "<mandatorName>",
        "general": {
            "revision": <revisionId>,
            "name": "<revisionName>",
            "comment": "<comment>",
            "editedByUser": <editedByUserUid>,
            "editedSince": <editedSinceTimestamp>,
```

```
            "lastChangedBy": "<lastChangedByUserName>",
            "lastChangedTimestamp": <lastChangedTimestamp>,
            "goliveInitiationUser": <goliveInitiationUserUid>|-1,
            "setLifeBy": "<setLifeBy>",
            "setLifeTimestamp": <setLifeTimestamp>,
            "retiredBy": "<retiredBy>",
            "retiredTimestamp": <retiredTimestamp>,
            "status": "<revisionStatus>"
        },
        "elementType": "<elementType>",
        "<elementType>": [
            {
                ... ...
            }
        ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

155 GetEnabledAnalyses

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getEnabledAnalyses",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision for which the analyses should be listed.

Returns

A HTTP response with the following content:

```
{
    "enabledAnalyses": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

156 GetEnabledRulesetsOfAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getEnabledRulesetsOfAnalysis",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>,
        "own": true|false
    }
}
```

- "revision": Uid of the revision.
- "data:analysisId": Uid of the analysis.
- "data:own": True to retrieve rulesets enabled for the analysis only from the selected revision. False to retrieve all rulesets enabled for the analysis.

Returns

A HTTP response with the following content:

```
{
    "analysisEnabledRulesets": [
        {
            "uid": <rulesetUid>,
            "name": "<rulesetName>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

157 GetEventLogMessage

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getEventLogMessage",
    "uid": <eventLogId>
}
```

- "uid": Id of the event log.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "number": <eventLogId>,
        "system": true|false,
        "audit": true|false,
        "external": true|false,
        "console": true|false,
        "pciDssMandated": true|false,
        "comment": "<comment>",
        "level": "<logLevel>",
        "systemChanged": true|false,
        "auditChanged": true|false,
        "externalChanged": true|false,
        "consoleChanged": true|false,
        "commentChanged": true|false,
        "levelChanged": true|false,
```

```
        "systemDefault": true|false,
        "auditDefault": true|false,
        "externalDefault": true|false,
        "consoleDefault": true|false,
        "commentDefault": "<defaultComment>",
        "levelDefault": "<defaultLogLevel>"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

158 GetEventLogMessagesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getEventLogMessagesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

159 GetEventsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getEventsTable",
    "type": "inherited" | "own",
    "revision": <revisionUid>
}
```

- "type" (mandatory): Choose "own" to retrieve the events defined in the given revision. Choose "inherited" to retrieve all events from the head mandators.
- "revision" (mandatory): Uid of the revision for which the events should be shown.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

160 GetExecutables

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getExecutables",
    "data": {
        "mandators": [<mandatorUid>]
    }
}
```

- "data:mandators" (mandatory): Uids of a list of mandators that the executables belong to.

Returns

A HTTP response with the following content:

```
{
    "executables": [
        {
            "streamType": "<streamType>",
            "uid": <uid>,
            "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

161 GetExternalQueries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getExternalQueries",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator to check privileges and available external queries.

Returns

A HTTP response with the following content:

```
{
    "externalQueries": [...],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

162 GetExternalQueriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getExternalQueriesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [{
        "uid": <uid>,
        "enabled": true | false,
        "comment": "<comment>",
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

163 GetFastLinkStatusTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFastLinkStatusTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        {
            "instanceName": "<instanceName>",
            "instanceStatus": [
                {
                    ... ...
                }
            ]
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

164 GetFinalRulesets

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFinalRulesets",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision from which the final rulesets are retrieved.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [
        {
            "uid": <rulesetUid>,
            "name": "<rulesetName>",
            "comment": "<comment>",
            "revision": <revisionUid>
        },
        ... ...
    ],
    "mayDefineFinalRulesets": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

165 GetFinalRulesetsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFinalRulesetsTable",
    "type": "own" | "inherited",
    "revision": <revisionUid>
}
```

- "type": Choose "own" to only retrieve final rulesets defined in the give revision. Choose "inherited" to retrieved all final rulesets inherited from the head mandators.
- "revision" (mandatory): Uid of the revision for which the table should be returned.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <rulesetUid>,
            <revisionUid>,
            <priority>,
            "<rulesetName>",
            "<comment>",
            <enabled>"Yes" | "No",
            <enabledRules>,
            <definedRules>,
```

```
            "<mandatorName>",
            "<conditions>"
        ],
        ... ...
    ],
    "mayDefineFinalRulesets": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

166 GetFittingRulesets

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFittingRulesets",
    "uid": <ruleUid>,
    "revision": <revisionUid>
}
```

- "uid": Uid of the rule to be moved.
- "revision" (mandatory): Uid of the revision in which the rulesets should be searched.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [
        {
            "uid": <rulesetUid>,
            "name": "<rulesetName>",
            "comment": "<comment>",
            "revision": <revisionUid>
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

167 GetFollowupUsers

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFollowupUsers",
    "type": <type>,
    "uid": <caseUid>,
    "data": {
        "mandators": [<mandatorUid>]
    }
}
```

- "type" (optional): Describes the context of the request. Available values are: "caseSelection" and "caseQueue". It is not used when a valid "uid" is provided.
- "uid" (optional): Uid of the case to search potential followup users for.
- "data" (optional): Only used when "type" is "caseSelection". -"data:mandators" (optional): Uids of a list of mandators from which the users should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "users": [
        {
            "streamType": "user",
            "uid": <userUid>,
            "enabled": true|false,
            "login": <userLoginName>,
```

```
            "name": "<userName>"
        },
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

168 GetFormulasTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFormulasTable",
    "type": <type>,
    "revision": <revisionUid>
}
```

- "type" (optional): If set to "inherited", formulas inherited from the head mandators are retrieved. Otherwise, only formulas from the given revision are retrieved.
- "revision" (mandatory): Uid of the revision to retrieve formulas from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

169 GetFraudValues

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getFraudValues",
    "mandator":<mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator to get the fraud values from.

Returns

A HTTP response with the following content:

```
{
    "categories": [<categoriesForMetaAttributeFraud>],
    "manualFraudValue": <manualFraudValue>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

170 GetGroupByQueriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getGroupByQueriesTable",
    "user": <userUid>
}
```

- "user": Uid of the user who sends the request.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <groupByQueriesUid>,
            <mandatorUid>,
            "<mandatorName>",
            "<groupByQueriesName>",
            "<comment>",
            "<attributeName>",
            <attributeUid>,
            <lastChangedTimestamp>,
            "<lastChangedByUserName>"
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

171 GetGroupByQueryAccountResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getGroupByQueryAccountResults",
    "uid": <groupByQueryUid>,
    "data": {
        "resultId": <resultId>,
        "period": <period>,
        "groupingValue": "<groupByAttributeValue>"
    }
}
```

- "uid" (mandatory): Uid of the group by query for which the results should be returned.
- "data:resultId" (mandatory): The resultId, which was returned by the request "executeGroupByQuery", to get the exact result of this execution.
- "data:period" (optional): The time period if timing analysis is enabled.
- "data:groupingValue" (mandatory): The grouping value that should be in the transactions.

Returns

A HTTP response with the following content:

```
{
    "accountAttribute": {
    "name": "<attributeName>",
    "format": "<format>" | "None",
    "decimals": <decimals>,
    "dimension": "<dimension>",
    "type": "<type>"
    },
    "accounts": [
        "<accountName>",
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

172 GetGroupByQueryResult

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getGroupByQueryResult",
    "uid": <groupByQueryUid>,
    "data": {
        "resultId": <resultId>,
        "maxRecords": <maximumNumberOfRecords>,
        "sortedBy": "none" | "falseAlarms" | "fraudAmount" | "fraudRatio" | "genuineAmount" | "totalAmount"
    }
}
```

- "uid" (mandatory): Uid to identify the group by query, to get the result from.
- "data:resultId" (mandatory): Id of the result.
- "data:maxRecords": Maximum number of records to print.
- "data:sortedBy": If sorting should be applied, the kind of sorting can be chosen here.

Returns

A HTTP response with the following content:

```
{
    "dimension": "<dimension>",
    "decimals": <decimals>,
    "groupByQueryName": "<groupByQueryName>",
    "running": true|false,
    "queryResult": [
        [
            "<groupByAttributeValue>",
            "groupByAttributeValue_CATEGORY",
            <groupingUID>,
            <timePeriod>,
            "<timePeriodDisplay>",
            <totalNumber>,
            <totalAmount>,
            <totalAverage>,
            <genuineNumber>,
            <genuineAmount>,
            <genuineAverage>,
            <fraudNumber>,
            <fraudAmount>,
            <fraudAverage>,
            <falseAlarms>,
            <fraudRatio>,
            <accounts>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

173 GetGroupByQueryResultMetaInfo

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getGroupByQueryResultMetaInfo",
    "uid": <groupByQueryUid>,
    "data": {
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the group by query to execute function in.
- "resultId" (mandatory): The result id to get meta information about.

Returns

A HTTP response with the following content:

```
{
    "groupByQueryName": "<groupByQueryName>",
    "groupingAttribute": {
        "name": "<groupByAttributeName>",
        "format": "<format>" | "None",
        "dataType": "<dataType>",
        "masked": 0|1
    },
    "timestampAnalysisEnabled", 0|1,
```

```
    "timestampAttribute": {
        ... ...
    },
    "accountAnalysisEnabled": 0|1,
    "rulePerformanceEnabled": 0|1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

174 GetGroupByQueryResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getGroupByQueryResults",
    "uid": <groupByQueryUid>
}
```

- "uid" (mandatory): Uid to identify the group by query, to get the results from.

Returns

A HTTP response with the following content:

```
{
    "availableResults": [
        [
            <resultId>,
            <startedOnTimestamp>,
            <finished>"Yes" | "No",
            "<dataSelectionConditions>",
            <dataRangeFromUrid>,
            <dataRangeToUrid>,
            <timingAnalysis>true|false,
            <accountAnalysis>true|false
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

175 GetIndexAspects

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexAspects",
    "uid": <indexUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of index whose aspects should be returned.
- "revision" (mandatory): Uid of the revision where the index is defined.

Returns

A HTTP response with the following content:

```
{
    "indexAspects": [
        {
            "uid": <attributeUid>,
            "name": "<attributeName>"
        }
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

176 GetIndexAttribute

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexAttribute",
    "uid": <indexUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the index for which the index attribute should be shown.
- "mandator" (mandatory if no revision): Uid of the mandator in which the index is defined.
- "revision" (mandatory if no mandator): Uid of the champion revision in which the index is defined.

Returns

A HTTP response with the following content:

```
{
    "attribute": {
        "uid": <attributeUid>,
        "name": "<attributeName>",
        "mandator": <mandatorUid>,
        "revision": <revisionUid>,
        "mdcStored": true|false,
        "origin": "<origin>",
        "type": "<type>",
```

```
        "encrypted": true|false,
        "length": <attributeLength>,
        "decimals": <decimals>,
        "dimension": "<dimension>",
        "categoriesEnabled": true|false,
        "categories": [ ... ],
        "usedInMergingConclusion": true|false,
        "metaAttribute": "<metaAttribute>" | "None",
        "format": "<format>" | "None"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

177 GetIndexBasedEvaluations

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexBasedEvaluations",
    "data": {
        "mandators": [<mandatorUid>]
    }
}
```

- "data" (mandatory): See below.
- "data:mandators" (mandatory): Uids of a list of mandators to retrieve index based evaluations from.

Returns

A HTTP response with the following content:

```
{
    "indexBasedEvaluations": [
    ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

178 GetIndexBasedEvaluationsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexBasedEvaluationsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

179 GetIndexedAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexedAttributes"
}
```

Returns

A HTTP response with the following content:

```
{
    "indexAttributes": [
        {
            "uid": <indexUid>,
            "revision": <revisionUid>,
            "mandator": <mandatorUid>,
            "name": "<indexName>",
            "attribute": {
                "type": "<attributeType>",
                "format": "<format>" | "None",
                "decimals": <decimals>,
                "length": <length>
            },
        ... ...
        }
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

180 GetIndexes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIndexes",
    "type": <type>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "selectedMandators": [<mandatorUid>]
    }
}
```

- "type" (mandatory): Controls the general behaviour of the retrieval. Available values for the "type" are:

```
"caseQueue"
"checkIndex"
"collusion",
"complianceList"
"counter"
"counterGeneration"
"doubletDetection"
"fraudFlag"
"indexBasedEvaluation"
"job"
"masterdataAssociated"
"merchantMonitoringTemplate"
"merging"
"pattern"
"precedent"
"query"
```

```
        "rebuildIndex"
        "resetIndex"
        "sai"
        "setXdcSizes"
```

- "mandator" (optional): Uid of the mandator from which indexes should be retrieved. Usage depends on "type".
- "revision" (optional): Uid of the revision from which indexes should be retrieved. Usage depends on "type".
- "data" (optional): Only used for type "job".
- "data:selectedMandators" (optional): Uids of a list of mandators to retrieve the indexes from.

Returns

A HTTP response with the following content:

```
{
    "indexes": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

181 GetIndexesTable

Parameters

httpRequest: Holds the request variables that have been passed to the server

```
{
    "request": "getIndexesTable",
    "type": <type>,
    "revision": <revisionUid>
}
```

- "type" (optional): If not set or set to a value different from "inherited", only indexes owned by the revision are retrieved. Otherwise, only inherited indexes are retrieved.
- "revision" (mandatory): Uid of the revision to retrieve the indexes from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

182 GetInputsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInputsTable",
    "type": <type>,
    "revision":<revision>
}
```

- "type" (optional): If not set or set to a value different from "inherited", only input attributes owned by the revision are retrieved. Otherwise, only inherited input attributes are retrieved.
- "revision" (mandatory): Uid of the revision to retrieve input attributes from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

183 GetInstances

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInstances",
    "data": {
        "checkInstanceStatus": true|false
    }
}
```

- "checkInstanceStatus": limit the retrieved instances to only those that are reachable.

Returns

A HTTP response with the following content:

```
{
    "instances": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

184 GetInstancesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInstancesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "autoRefresh": <autoRefreshInterval>,
    "tableContent": [
    ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

185 GetInternalModelGenerationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInternalModelGenerationDataSelection",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which model generation data selection should be fetched.

Returns

A HTTP response with the following content:

```
{
    "simulationAttributesAvailable": true|false,
    "missingAttributes": [ ... ... ],
    "verify": true|false,
    "generationScenario": "FromScratch" | "CurrentRevisionOnly" | "CurrentRevisionAndHistory",
    "trainingDataSelectionStream": {
        "mandators": [ ... ... ],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
```

```
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "incDdc": true|false,
        "conditions": [ ... ... ]
    },
    "verificationDataSelectionStream": {
        "mandators": [<mandatorUid>],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
```

```
            "toDays": <toDays>
        },
        "incDdc": true|false,
        "conditions": [ ... ... ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

186 GetInternalModelGenerationSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInternalModelGenerationSettings",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which model generation settings should be returned.

Returns

A HTTP response with the following content:

```
{
    ...will vary for different models
    "modeName": "<rulesName>",
    "modelComment": "<rulesComment>",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

187 GetInternalModelGenerationStatus

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInternalModelGenerationStatus",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which model generation settings should be returned.

Returns

A HTTP response with the following content:

```
{
    "generationCompleted": true|false,
    "accuracy": <accuracy>,
    "refresh": <refresh>,
    "trainingDataSelectionError": "<errortext>",
    "verificationDataSelectionError": "<errortext>",
    "modelGenerationError": "<errortext>",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

188 GetInvestigationReportsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getInvestigationReportsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <investigatoinReportUid>,
            "<mandatorName>",
            "<reportName>",
            "<comment>",
            "<lastChangedByUserName>",
            <lastChangedTimestamp>
        ],
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

189 GetIsRulesetEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getIsRulesetEnabled",
    "uid": <rulesetUid>,
    "revision": <revisionUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "rulesetEnabled": 0|1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

190 GetJobsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getJobsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
        ... ...
    ],
    "refresh": <autoRefereshPeriodInSeconds>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

191 GetKeyPerformanceIndicatorsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getKeyPerformanceIndicatorsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

192 GetKeysTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getKeysTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "autoRefresh": <autoRefreshSetting>,
    "changeMasterKeyStatus": <changeMasterKeyStatus>,
    "masterKeyChangable": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

193 GetLastUrid

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getLastUrid"
}
```

Returns

A HTTP response with the following content:

```
{
    "urid": [<urid>],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

194 GetLastUserAction

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getLastUserAction"
}
```

Returns

A HTTP response with the following content:

```
{
    "lastUserAction": <timestamp>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

195 GetListsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getListsTable",
    "type": "inherited" | "own",
    "revision": <revisionUid>
}
```

- "type" (mandatory): For inherited or revision's own lists table.
- "revision" (mandatory): Uid of revision containing the lists.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [{
        "uid": <uid>,
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

196 GetLoadBalancingInstances

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getLoadBalancingInstances",
    "uid": <elementUid>
}
```

- "uid": Uid of the element planned to run on another instance. Right now only revisions are supported.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [ ... ],
    "recommendedInstanceUid": <instanceUid>
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

197 GetMandators

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMandators",
    "type": "<type>",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "type": The type of request. Available values for the "type" are:

```
"analysis" (requires "mandator")
"groupByQueryDataSelection" (requires "mandator")
"queryDataSelection" (requires "mandator")
"simulation" (requires "mandator")
"simulationQuery" (requires "mandator")
"ruleGeneration" (requires "mandator" and "revision")
"counterGeneration" (requires "mandator" and "revision")
"user"
"reportGeneration"
"grants"
"reportUseMandatorData"
"chartSubMandator"
"statusAlarmIndicatorSubMandator"
"reportGenerationJob"
"commonPointQuery"
"ticker"
"caseromQuery"
"mandator"
"headMandators"
```

```
"query"
"groupByQuery"
"report"
"caseQueue"
"roles"
"notification"
"textModule"
"caseAction"
"externalQuery"
"reminder"
"caseCloseCode"
"statusAlarmIndicator"
"chart"
"caseFilter"
"caseCreation"
"caseGroup"
"definedRisk"
"compliance"
"merchantMonitoringRule"
"memoryManagement"
"rebuildIndex"
"resetIndex"
"checkIndex"
"setXdcSizes"
"fixMillisecondsMapping"
"reportingQuery"
```

- "mandator": Members of the selected mandator will be returned.
- "revision": The revision should be supplied for revision components.

Returns

A HTTP response with the following content:

```
{
    "mandators": [{
        "uid": <mandatorUid>,
        "name": "<name>",
        "maxMemoryForSimulationGB": <simulationMemorySize>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

198 GetMandatorSelectionTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMandatorSelectionTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

199 GetMandatorSelectionTree

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMandatorSelectionTree",
    "type": "adminMandators" | ""
}
```

- "type" (optional): If not specified, this request returns the mandators tree for the Model tab in the user interface. If set to "adminMandators", this request returns the mandators tree for the Administration tab in the user interface.

Returns

A HTTP response with the following content:

```
{
    "tree": [{
        "uid": <mandatorUid>,
        "label": "<mandatorLabel>",
        "champion": true|false,
        "revisions": <totalNumberOfRevisions>,
        "title": "<mandatorTitle>",
        "conditions": "<mandatorConditions>",
        "expanded": true|false,
        "depth": <mandatorDepth>,
        "selectable": 0|1,
        "children": [
            ... ...
```

```
        ]
    },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

200 GetMandatorsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMandatorsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <mandatorUid>,
        "<headMandatorName>" | "ROOT",
        "<mandatorName>",
        "<comment>",
        <numberOfRevisions>,
        <hasChampion>"Yes" | "No",
        <simulationMemoryInGB>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>,
        "<conditions>"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

201 GetMappingsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMappingsTable",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "mandator" (mandatory): Uid of mandator that owns the revision. Used to get the champion to access attributes.
- "revision" (mandatory): Uid of the revision that owns the mappings.

Returns

A HTTP response with the following content:

```
{
    "header": [{
        "name": "<name>",
        "type": "<type>",
        "messageUid": -1
        },
    ... ...
    ],
    "dataSource": [{
        "key": "<value>"
        },
```

```
    ... ...
    ],
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "mappingsUids": [{
        "attributeUid": <attributeUid>,
        "messageUid": <messageTypeUid>,
        "mappingUid": <mappingUid>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

202 GetMasterdataDeepInformation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMasterdataDeepInformation",
    "uid": <uid>,
    "mandator": <mandatorUid>,
    "data": {
        "parameter": "<indexValue>",
        "urid": <recordOfIndexValue>,
        "isTargetValue": true | false
    }
}
```

- "uid" (mandatory): Uid of the index.
- "mandator" (mandatory): Uid of the mandator.
- "data: parameter" (optional): The value of the index as text.
- "data: urid" (optional): Urid of the record that contains the value.
- "data: isTargetVale" (optional): Search on target or source of the index.

Returns

A HTTP response with the following content:

```
{
    "uid": <indexUid>,
    "mandator": <mandatorUid>,
    "parameter": "<parameter>",
    "indexAttribute": <indexAttribute>,
    "type": "<indexAttributeType>",
    "format": <indexAttributeFormat>,
    "tableHeader": [ ... ... ],
    "tableContent": [ ... ... ],
    "multipleValues": [ ... ... ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

203 GetMasterdatas

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMasterdatas",
    "type": <type>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "type" (mandatory): Choose one of these types: "caseQueue" | "caseSelection" | "ruleAction".
- "mandator" (optional): Uid of the mandator.
- "revision" (optional): Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "uid": <uid>,
    "name": "<name>",
    "comment": "<comment>",
    "index": <index uid>,
    "conditions": [<Condition>, ...],
    "deletionConditions": [<Condition>, ...],
    "insertTransaction": true | false,
    "multipleValues": true | false,
```

```
    "multipleValuesCapacity": <multipleValuesCapacity>,
    "associatedIndexUid": <associatedIndexUid>,
    "relationshipAttributes": [<Attribute>, ...],
    "attribute": <attribute uid>,
    "attributeData": {<attribute data>},
    "targetAttribute": <attribute uid>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

204 GetMasterdatasForIndex

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMasterdatasForIndex",
    "mandator": <mandatorUid>,
    "index": <indexUid>
}
```

- "mandator": Uid of the mandator.
- "index": Uid of the index.

Returns

A HTTP response with the following content:

```
{
    "masterdata": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

205 GetMasterdatasTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMasterdatasTable",
    "type": "inherited",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "type" (optional): If set to 'inherited', retrieve masterdatas from parent mandator(s). If not set, only retrieve masterdatas from the selected revision.
- "mandator": Uid of mandator.
- "revision": Uid of revision.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [ ... ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

206 GetMasterKeysTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMasterKeysTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <id>,
        "<status>",
        <numberOfKeys>,
        <activationTimestamp>,
        "<enteredByUserName>"
        ],
    ... ...
    ],
    "autoRefresh": <autoRefreshSetting>,
    "lastActiveMasterKey": <lastActiveMasterKeyId>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

207 GetMemoryManagementTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMemoryManagementTable",
    "data": {
        "mandators": [<mandatorUid>]
    }
}
```

- "data" (optional): Contains mandators and users for which to retrieve information about memory consumption.
- "data:mandators" (optional): A list of Mandator objects for which to retrieve information about memory consumption.

Returns

A HTTP response with the following content:

```
{
    "memory"Consumption": [[
        "<mandatorName>",
        <memoryUsedInGB>,
        <memoryLimitInGB>
        ],
    ... ...
    ],
    "totalUsedMemoryMandators": <totalUsedMemoryMandators>,
    "tableContent": [
```

```
        ... ...
    ],
    "totalUsedMemoryUsers": <totalUsedMemoryUsers>,
    "refresh": <autoRefreshSetting>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

208 GetMerchantMonitoringRulesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMerchantMonitoringRulesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ],
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

209 GetMerchantMonitoringTemplate

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMerchantMonitoringTemplate",
    "data": {
        "templateKey": "<templateKey>"
    }
}
```

- "data" (mandatory): Contains the key of the MerchantMonitoringTemplate.
- "data:templateKey" (mandatory): Identifies the MerchantMonitoringTemplate to retrieve.

Returns

A HTTP response with the following content:

```
{
    "merchantMonitoringTemplate": {
        "name": "<templateName>",
        "index": <indexUid>|-1,
        "mandator": <mandatorUid>|-1,
        "periodsBack": "<numberOfPeriodsBack>",
        "chargebackProfile": <chargebackProfile>,
        "totalSalesProfile": <totalSalesProfile>,
        "minimumNumberOfChargebacks": <minimumNumberOfChargebacks>,
        "minimumNumberOfTotalSales": <minimumNumberOfTotalSales>,
```

```
        "minimumRatioChargebackTotalBp": <minimumRatioChargebackTotalBp>,
        "numberOfConsecutivePeriodsToMeet": <numberOfConsecutivePeriodsToMeet>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

210 GetMerchantMonitoringTemplates

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMerchantMonitoringTemplates"
}
```

Returns

A HTTP response with the following content:

```
{
    "merchantMonitoringTemplates": [{
        "templateKey": "<templateKey>",
        "name": "<templateName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

211 GetMergingsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMergingsTable",
    "type": <type>,
    "revision":<revisionUid>
}
```

- "type" (optional): If not set or set to a value different from "inherited", only mergings owned by the revision are retrieved. Otherwise, only inherited mergings are retrieved.
- "revision" (mandatory): Uid of the revision to retrieve mergings from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

212 GetMessageReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMessageReport",
    "uid": <messageUid>
}
```

- "uid" (mandatory): Uid of the message to retrieve the report for.

Returns

A HTTP response with the following content:

```
{
    "messageReport": {
        "name": "<messageTypeName>",
        "comment": "<comment>",
        "mtid": <messageTypeId>,
        "messageType": "<messageType>",
        ["offline" | "online"]: {
            ...
        }
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

213 GetMessages

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMessages",
    "type": "Revision" | "Job" | "Queue" | "Reminder",
    "revision": <revisionUid>
}
```

- "type" (mandatory): The type of request.
- "revision" (optional): For request type "Revision" the revision has to be supplied.

Returns

A HTTP response with the following content:

```
{
    "messages": [{
        "uid": <messageUid>,
        "name": "<messageName>",
        "messageType": "<messageType>"
        "mtid": <mtidValue>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

214 GetMessagesReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMessagesReport"
}
```

Returns

A HTTP response with the following content:

```
{
    "messagesReport": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

215 GetMessagesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMessagesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <messageUid>,
        "<messageName>",
        "<comment>",
        <MTID>,
        <createTransactionRecords>true|false,
        "<type>",
        "<lastChangedByUserName>",
        "<lastChangedOnTimestamp>"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

216 GetMetaAttributeAmount

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMetaAttributeAmount"
}
```

Returns

A HTTP response with the following content:

```
{
    "metaAttributeAmount": {
        "dimension": "<unit>",
        "decimals": <decimals>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

217 GetMetaAttributeMaximum

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMetaAttributeMaximum",
    "type": <type>,
    "mandator":<mandatorUid>,
    "data": {
        "metaAttribute": <metaAttribute>
    }
}
```

- "type" (optional): Can be left out or set to one of ["", "caseQueue", "intercept"].
- "mandator" (optional): Uid of the mandator to retrieve the meta attribute's maximum value from. It is only used when the param "type" is left out or empty.
- "data" (mandatory): Contains the name of the meta attribute.
- "data:metaAttribute" (mandatory): Name of the meta attribute e.g. "Account", "Amount", "Fraud" etc.

Returns

A HTTP response with the following content:

```
{
    "metaAttributeMax": <maxValue>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

218 GetMissedCasesReportsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMissedCasesReportsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <missedCasesReportUid>,
        "<mandatorName>",
        "<reportName>",
        "<comment>",
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

219 GetModelComponents

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getModelComponents",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which model components should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [{
        "uid": <rulesetUid>,
        "name": "<rulesetName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

220 GetModelComponentsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getModelComponentsTable",
    "type": "inherited|"",
    "revision": <revisionUid>,
    "data": {
        "componentTypes": [...]
    }
}
```

- "type": Indicates whether or not inherited components should be returned.
- "revision": The revision for which model components should be returned.
- "data:componentTypes": List of model component types to retrieve. Possible values are: ["Ruleset", "RandomForest", "NeuralNetwork", "DecisionTree", "↩ BoostedTrees", "InternalRandomForest"]

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

221 GetModelingWorkflowsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getModelingWorkflowsTable",
    "mandator": <mandatorUid>
    "type": "inherited|"",
    "revision": <revisionUid>
}
```

- "mandator" (mandatory): Uid of the mandator that the modeling workflows belong to.
- "revision" (mandatory): Uid of the revision that the modeling workflows belong to.
- "type": Indicates whether or not inherited modeling workflows should be returned.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <modelingWorkflowUid>,
            "<modelingWorkflowName>",
            "<modelingWorkflowComment>"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

222 GetModelling

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getModelling",
    "uid": <attributeUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the attribute to retrieve modelling information from.
- "revision" (mandatory): Uid of the revision to retrieve the attribute's modelling information from.

Returns

A HTTP response with the following content:

```
{
    "modelling": {
        "attribute": <attributeUid>,
        "name": "<attributeName>",
        "origin": "<origin>",
        "type": "<type>",
        "dimension": "<dimension>",
        "decimals": <decimals>,
        "simulationRequired": true|false,
        "testEnabled": true|false,
        "simulationEnabled": true|false,
```

```
        "analysisEnabled": true|false,
        "counterGenerationEnabled": true|false,
        "modelGenerationEnabled": true|false,
        "attributeUsage": "<attributeUsage>",
        "maxComputedIndicators": <maxComputedIndicators>,
        "maxSelectedIndicators": <maxSelectedIndicators>,
        "linearFrom": <linearFrom>,
        "linearStep": <linearStep>,
        "linearTo": <linearTo>,
        "customCategories": [
            ... ...
        ],
        "customIntervalFrom": [
            ... ...
        ],
        "customIntervalTo": [
            ... ...
        ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

223 GetModellingsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getModellingsTable",
    "revision": <revisionUid>,
    "data": {
        "showInheritedAttributes": true|false,
        "selectedOrigins": [<origin>],
        "selectedMessages": [<messageUid>]
    }
}
```

- "revision" (mandatory): Uid of the revision to retrieve the modellings table from.
- "data" (optional): Additional values to filter the displayed attributes.
- "data:showInheritedAttributes" (optional): When set to "false", only attributes owned by the revision are retrieved. Otherwise, the inherited attributes are retrieved as well.
- "data:selectedOrigins" (optional): A list of attribute origins. Possible values are: ["List", "DeviceIdentification", "Precedent", "Profile", "Pattern", "Event", "Counter", "Collusion", "Formula", "Output", "Input"]. This is to limit the attributes to those that have at least one of the defined origins.
- "data:selectedMessages" (optional): Uids of a list of messages. This is only used when the respective origin was not selected. It is to limit the input and output attributes to those that have a mapping in at least one of the messages.

Returns

A HTTP response with the following content:

```
{
    "tableContent" :[
        ... ...
    ],
    "totalAttributes": <totalNumberOfAttributes>,
    "testEnbled": <totalNumberOfAttributesWithTestEnabled>,
    "simulationEnabled": <totalNumberOfAttributesWithSimulationEnabled>,
    "analysisEnabled": <totalNumberOfAttributesWithAnalysisEnabled>,
    "modelGenerationEnabled": <totalNumberOfAttributesWithModelGenerationEnabled>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

224 GetMultipleRulesFiringStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMultipleRulesFiringStatistics",
    "revision": <revisionUid>,
    "data": {
        "maxNumberOfRules": <maxNumberOfRules>,
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision to retrieve Rule firing statistics from.
- "data" (mandatory): Additional values to limit the maximum number of displayed rules and select an Analysis to use.
- "data:maxNumberOfRules" (mandatory): The maximum number of rules for which to retrieve firing statistics.
- "data:analysisId" (mandatory): Id of the analysis to use as a base for the Rule firing statistics.

Returns

A HTTP response with the following content:

```
{
    "multipleRulesFiringNames": [
        ...
    ],
    "multipleRulesFiring": [
```

```
        ...
    ],
    "multipleRulesInterceptNames: [
        ...
    ],
    "multipleRulesIntercept: [
        ...
    ],
    "multipleRulesGeneratingAlarmsNames": [
        ...
    ],
    "multipleRulesGeneratingAlarms: [
        ...
    ],
    "multipleRulesGeneratingNotficationsNames": [
        ...
    ],
    "multipleRulesGeneratingNotifications: [
        ...
    ],
    "multipleRulesGeneratingRemindersNames": [
        ...
    ],
    "multipleRulesGeneratingReminders: [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

225 GetMyWorkingQueues

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getMyWorkingQueues"
}
```

Returns

A HTTP response with the following content:

```
{
    "workingQueues": {
        "public": [{
            "uid": <workingQueueUid>,
            "name": "<workingQueueName>"
            },
        ... ...
        ],
        "private": [{
            "streamType": "user",
            "uid": <userUid>,
            "enabled": true|false,
            "login": "<loginName>",
            "name": "<userName>"
            },
        ... ...
        ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

226 GetNextCase

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getNextCase"
}
```

Returns

A HTTP response with the following content:

```
{
    "uid": <caseUid>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

227 GetNotifications

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getNotifications",
    "type": <type>,
    "mandator": <mandatorUid>
}
```

- "type" (mandatory): Select one of these types: "query" | "caseSelection" | "ruleAction".
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "notifications": [{
        "uid": <uid>,
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

228 GetNotificationsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getNotificationsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [{
        "uid": <uid>,
        "id": <id>,
        "enabled": true | false,
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

229 GetOfflineComplianceListDefinitions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getOfflineComplianceListDefinitions",
    "data": {
        "mandators": [mandatorUid, mandatorUid,...]
    }
}
```

- "data:mandators": An array of mandator uids.

Returns

A HTTP response with the following content:

```
{
    "complianceLists": [
        { ... },
        ...
    ],
    "currentlyReloading": true|false,
    "readonly": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

230 GetOptimizationAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getOptimizationAnalysis",
    "revision": <revisionUid>,
    "data":  {
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision from which to retrieve the rule optimization analysis.
- "data" (optional): Identifies the analysis to use.
- "data:analysisId" (optional): Id of the analysis to use as a base for the Rule optimization analysis.

Returns

A HTTP response with the following content:

```
{
    "statistics": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

231 GetOutgoingChannelConfigurations

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getOutgoingChannelConfigurations",
    "type": "<type>"
}
```

- "type" (mandatory): Choose one of these types: "caseAction" | "externalQuery" | "reportGenerationJob" | "notification" | "forwardingModule".

Returns

A HTTP response with the following content:

```
{
    "outgoingChannelConfigurations": [{
        "uid": <uid>,
        ... ...
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

232 GetOutgoingChannelConfigurationTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getOutgoingChannelConfigurationTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [{
        "uid": <uid>,
        ... ...
        "name": "<name>"
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

233 GetOutputsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getOutputsTable",
    "type": "inherited" | "own",
    "revision": <revisionUid>
}
```

- "type" (optional): Defines if inherited or own output attribute shall be returned. The default value is "own".
- "revision" (mandatory): Uid of the revision from which the outputs should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
            ... ...
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

234 GetPasswordSafes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPasswordSafes"
}
```

Returns

A HTTP response with the following content:

```
{
    "passwordSafes": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

235 GetPasswordSafeTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPasswordSafeTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

236 GetPatternsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPatternsTable",
    "type": "own" | "inherited",
    "revision": <revisionUid>
}
```

- "type" (optional): Defines if inherited or own patterns are returned. The default value is "own".
- "revision" (mandatory): Uid of the revision from which the patterns should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <revisionUid>,
        <patternUid>,
        "<patternName>",
        "<comment>",
        "<mandatorName>",
        "<indexName>",
        "<computationConditions>",
        <numberOfStencils>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

237 GetPciDssReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPciDssReport"
}
```

Returns

A HTTP response with the following content:

```
{
    "pciDssReport":[ {"type": "<type">, "variables": [ ... ...] }, ... ... ],
    "generatedOn": <now as timestamp>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

238 GetPossibleAmountAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPossibleAmountAttributes",
    "revision": <revisionUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "passibleAmountAttributes": [{
        "uid": <attributeUid",
        "name": "<attributeName>",
        "mandator": <mandatorUid>,
        "revision": <revisionUid>,
        "mdcStored": true|false,
        "origin": "Input" | "Output",
        "type": "Numeric",
        "encrypted": true|false,
        "length": <attributeLength>,
        "decimals": <decimals>,
        "dimension": "<dimension>",
        "categoriesEnabled": true|false,
        "categories": [
            ... ...
        ],
```

```
        "usedInMergingConclusion": true|false,
        "metaAttribute": "None" | "<metaAttributeType>",
        "format": "None" | "<format>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

239 GetPrecedentsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPrecedentsTable",
    "type": <type>,
    "revision": <revisionUid>
}
```

- "type" (optional): If set to "inherited", only inherited precedents are retrieved. Otherwise, only precedents owned by the given revision are retrieved.
- "revision" (mandatory): Uid of the revision to retrieve precedents from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

240 GetPreference

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPreference",
    "type": "<type>"
}
```

- "type" (mandatory): lookup key for preference value.

Returns

A HTTP response with the following content:

```
{
    "preference": ... ...,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

241 GetPreProcessingRulesets

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPreProcessingRulesets",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the rulesets are listed.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [{
        "uid": <rulesetUid>,
        "name": "<rulesetName>",
        "comment": "<comment>",
        "revision": <revisionUid>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

242 GetPreProcessingRulesetsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPreProcessingRulesetsTable",
    "type": "inherited|"",
    "revision": <revisionUid>
}
```

- "type": Indicates whether or not inherited components should be returned.
- "revision": The revision for which model components should be returned.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

243 GetPrintView

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPrintView",
    "uid": <revisionUid>
}
```

- "uid": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "data": {
        ... ...
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

244 GetProcessPriorities

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getProcessPriorities"
}
```

Returns

A HTTP response with the following content:

```
{
    "data": [
        "<priority1>",
        "<priority2>",
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

245 GetProfiles

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getProfiles",
    "uid": <indexUid>,
    "revision": <revisionUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (optional): Uid of the index in order to retrieve all calendar profiles that are referencing it. If not specified, all calendar profiles from the revision are retrieved.
- "revision" (optional): Uid of the revision that the calendar profiles belong to. If not specified, the mandator's champion revision is taken.
- "mandator" (mandatory): Uid of the mandator whose champion revision is used, if no revision is specified.

Returns

A HTTP response with the following content:

```
{
    "profiles": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

246 GetProfilesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getProfilesTable",
    "type": <type>,
    "revision":<revisionUid>
}
```

- "type" (optional): If set to "inherited", only inherited calendar profiles are returned.Otherwise, only calendar profiles owned by the revision are returned.
- "revision" (mandatory): Uid of the revision to retrieve calendar profiles from.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

247 GetPythonFunctions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getPythonFunctions",
    "type": "<type>",
    "revision": <revisionUid>
}
```

- "type" (mandatory): The available values for "type" are: "ruleConclusion", "ruleConclusionPreProcessing", "formula".
- "revision" (mandatory): Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "pythonFunctions": [{
        "name": "<pythonFunctionName>",
        "helpText": "<pythonFunctionHelpText>",
        "arguments": ["argument1", "argument2", ...]
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

248 GetQueries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getQueries",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "queries": [
        {
            "streamType": "investigationQuery",
            "uid": <queryUid>,
            "name": <queryName>
        },
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

249 GetQueriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getQueriesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [ ... ],
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

250 GetQueryOptions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getQueryOptions"
}
```

Returns

A HTTP response with the following content:

```
{
    "queryOptions": {
        "maxRecordsWarning": <maxRecordsWarning>,
        "maxRecordsLimit": <maxRecordsLimit>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

251 GetQueryResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getQueryResults",
    "uid": <queryUid>,
    "data": {
        "resultId": <resultId>
    }
}
```

- "uid": Uid of the query.
- "data:resultId": The resultId generated by the request "executeQuery".

Returns

A HTTP response with the following content:

```
{
    "queryResult": {
        "name": "<queryName>",
        "comment": "<comment>",
        "header": [
            {
                "name": "<attributeName>",
                "attributeUid": <attributeUid>,
                "align": "<align>",
```

```
            "decimals": <decimals>,
            "dimension": <dimension>,
            "hyperlinkQueryUid": [],
            "commonPointQueryUid": [],
            "type": "<attributeType>",
            "format": "<format>"
        },
        ... ...
    ],
    "tableContent": [
        ... ...
    ]
},
"indexValueUrid": <urid>,
"running": true|false,
"recordsTotal": <recordsTotal>,
"recordsFinished": true|false,
"duration": <duration>,
"refresh": true|false,
"highlightCppAttributes": <highlightCppAttributes>,
"amountDimension": "<amountDimension>",
"amountDecimals": "<amountDecimals>",
"statistics": {
    "transaction": {
        "genuine": {
            "records": <numberOfRecords>,
            "amount": <amount>
        },
        "fraud": {
            "records": <numberOfRecords>,
            "amount": <amount>
        }
    }
},
"falsePositives": <falsePositiveRate>,
"numberOfAccounts": <numberOfAccounts>,
"extractTemplate": "<extractTemplate>",
"responseStatus": ["<status>", "<feedbacktext>"],
```

```
    "reloadUserProfile": true|false
}
```

252 GetQuickSearchCasesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getQuickSearchCasesTable",
    "data": {
        "caseSearch": true|false
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "header": [
        {
            "uid": "<uid>",
            "name": "<name>",
            "align": "<align>",
            "type": "<type>",
            "decimals": <decimals>,
            "format": "<format>"
        },
        ... ...
    ],
    "tableContent": [
        {
            "uid":<caseId>,
```

```
            "Queue": <caseQueueName>,
            "Generated": <generatedTimestamp>,
            "Score": <scoreValue>,
            "Hits": <numberOfHits>,
            "Status": "<status>",
            "FraudStatus": "<fraudStatus>",
            "CloseCodeName": "<caseCloseCodeName>",
            "User": <userUid>,
            "UserName": "<userName>",
            "LastUserAction": <lastUserActionTimestamp>,
            "MandatorName": "<mandatorName>",
            "Mandator": <mandatorUid>,
            "CppName": "<cppName>",
            "Cpp": "cppCode",
            ... ...
        },
        ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

253 GetRealtimeInterceptionCodes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRealtimeInterceptionCodes"
}
```

Returns

A HTTP response with the following content:

```
{
    "realtimeInterceptionCodes": [{
        "uid": <realtimeInterceptionCodeUid>,
        "name": "<realtimeInterceptionCodeName>,
        "from": <fromValue>,
        "to": <toValue>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

254 GetRealtimeInterceptionCodesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRealtimeInterceptionCodesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
        ... ...
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

255 GetReferencesOfAttribute

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfAttribute",
    "uid": <attributeUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the attribute to retrieve references of.
- "revision" (mandatory): Uid of the revision in which to search the attribute.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "foreignReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

256 GetReferencesOfCaseAction

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfCaseAction",
    "uid": <uid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the case action.
- "mandator" (mandatory): Uid of the mandator of the (outgoing channel configuration) case action.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [ ... ... ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

257 GetReferencesOfCaseCloseCode

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfCaseCloseCode",
    "uid": <caseCloseCodeUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the case close code.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "hasCases": true (optional),
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

258 GetReferencesOfCaseQueue

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfCaseQueue",
    "uid": <caseQueueUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the case class.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "hasCases": true (optional),
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

259 GetReferencesOfCaseState

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfCaseState",
    "uid": <caseStateUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the case state.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "hasCases": true (optional),
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

260 GetReferencesOfCaseWorkflow

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfCaseWorkflow",
    "uid": <caseWorflowUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the case workflow.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

261 GetReferencesOfChart

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfChart",
    "uid": <chartUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the chart to retrieve references of.
- "mandator" (mandatory): Uid of the mandator the chart belongs to. This is used to check the user's privilege.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

262 GetReferencesOfComplianceList

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfComplianceList"
    "uid": <ComplianceListUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the compliance list to retrieve references of.
- "mandator" (mandatory): Uid of the mandator the compliance list belongs to.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

263 GetReferencesOfDefinedRiskList

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfDefinedRiskList"
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the defined risk list to retrieve references of.
- "mandator" (mandatory): Uid of the mandator the defined risk list belongs to.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

264 GetReferencesOfEvent

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfEvent",
    "uid": <eventUid>,
    "revision": <revisionUid>
}
```

- "uid": Uid of the event.
- "revision": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

265 GetReferencesOfExternalQuery

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfExternalQuery",
    "uid": <externalQueryUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the external query.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

266 GetReferencesOfIndex

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfIndex",
    "uid": <indexUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the index to retrieve references of.
- "revision" (mandatory): Uid of the revision to search the index in.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ... ...
    ],
    "foreignReferences": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

267 GetReferencesOfMasterdata

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfMasterdata",
    "uid": <masterdataUid>,
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision that is used as base for the masterdata reference search.
- "uid" (mandatory): Uid of the masterdata whose references are requested.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [<reference>],
    "foreignReferences": [<reference>],
    "responseStatus": [<status>, <message>],
    "reloadUserProfile": true | false
}
```

268 GetReferencesOfMessage

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfMessage",
    "uid": <messageUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the message whose references are requested.
- "revision" (mandatory): Uid of the revision that is used as base for the message reference search.

Returns

A HTTP response with the following content:

```
{
    "ownReferences":[<reference>],
    "responseStatus":[<status>, <message>],
    "reloadUserProfile": true | false
}
```

269 GetReferencesOfNotification

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfNotification",
    "uid": <notificationUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the notification whose references are requested.
- "revision" (mandatory): Uid of the revision that the notification belongs to.

Returns

A HTTP response with the following content:

```
{
    "ownReferences":[<reference>],
    "responseStatus":[<status>, <message>],
    "reloadUserProfile": true | false
}
```

270 GetReferencesOfOutgoingChannelConfiguration

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfOutgoingChannelConfiguration",
    "uid": <outgoingChannelConfigurationUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the outgoing channel configuration.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

271 GetReferencesOfPasswordSafe

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfPasswordSafe",
    "uid": <passwordSafeUid>
}
```

- "uid" (mandatory): Uid of the password safe.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [ <references> ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

272 GetReferencesOfProfile

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfProfile",
    "uid": <profileUid>,
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision that is used as base for the Profile reference search.
- "uid" (mandatory): Uid of the profile whose references are requested.

Returns

A HTTP response with the following content:

```
{
    "ownReferences":[<reference>],
    "responseStatus":[<status>, <message>],
    "reloadUserProfile": true|false
}
```

273 GetReferencesOfPythonModule

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getReferencesOfPythonModule",
    "uid": <pythonModuleUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the PythonModule.
- "mandator" (mandatory): Uid of the Mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

274 GetReferencesOfQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfQuery",
    "uid": <queryUid>,
    "mandator":<mandatorUid>
}
```

- "uid" (mandatory): Uid of the query.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [<reference>],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

275 GetReferencesOfReminder

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfReminder",
    "uid": <reminderUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the reminder to request the references for.
- "mandator" (mandatory): Uid of the mandator of the reminder.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

276 GetReferencesOfUserGroup

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReferencesOfUserGroup",
    "uid": <userGroupUid>,
    "mandator":<mandatorUid>
}
```

- "uid" (mandatory): Uid of the user group.
- "mandator" (mandatory): Uid of the mandator.

Returns

A HTTP response with the following content:

```
{
    "ownReferences": [<reference>],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

277 GetReminders

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReminders",
    "type": "query" | "ruleAction" | "caseSelection",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "type" (optional): The type to identify the origin of the request.
- "mandator" (optional): Uid of the mandator that the specified type belongs to.
- "revision" (optional): Uid of the revision that the specified type belongs to.

Returns

A HTTP response with the following content:

```
{
    "reminders": [{
        "uid": <reminderUid>,
        "id": <reminderId>,
        "name": "<reminderName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

278 GetReminderTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReminderTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

279 GetReportingAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingAttributes",
    "uid": <caseUid>
}
```

- "uid" (mandatory): Uid of a case for which reporting attributes has to be retrieved.

Returns

A HTTP response with the following content:

```
{
    "reportingAttributes": [{
        "uid": <attributeUid>,
        "name": "<attributeName>",
        "mandator": <mandatorUid>,
        "revision": <revisionUid>,
        "mdcStored": true|false,
        "origin": "<attributeOrigin>",
        "type": "<type>",
        "encrypted": true|false,
        "length": <length>,
        "decimals": <decimals>,
        "dimension": "<dimension>",
        "categoriesEnabled": true|false,
```

```
        "categories": [ ... ],
        "usedInMergingConclusion": true|false,
        "metaAttribute": "<metaAttribute>" | "None",
        "format": "<format>" | "None"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

280 GetReportingQueriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingQueriesTable"
    "user": <userUid>
}
```

- "user": Uid of the user who sends the request.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        [
            <reportingQueriesUid>,
            <mandatorUid>,
            "<mandatorName>",
            "<reportingQueriesName>",
            "<comment>",
            "<attributeName>",
            <attributeUid>,
            <lastChangedTimestamp>,
            "<lastChangedByUserName>"
        ],
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

281 GetReportingQueryAccountResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingQueryAccountResults",
    "uid": <ReportingQueryUid>,
    "data": {
        "resultId": <resultId>,
        "period": <period>,
        "groupingValue": "<ReportingAttributeValue>"
    }
}
```

- "uid" (mandatory): Uid of the reporting query for which the results should be returned.
- "data:resultId" (mandatory): The resultId that was returned from "executeReportingQuery", in order to retrieve the exact result of this execution.
- "data:period" (optional): The time period if timing analysis is enabled.
- "data:groupingValue" (mandatory): The reporting attribute value that should be in the transaction.

Returns

A HTTP response with the following content:

```
{
    "accountAttribute": {
    "name": "<attributeName>",
    "format": "<format>" | "None",
    "decimals": <decimals>,
    "dimension": "<dimension>",
    "type": "<type>"
    },
    "accounts": [
        "<accountName>",
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

282 GetReportingQueryResult

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingQueryResult",
    "uid": <reportingQueryUid>,
    "data": {
        "resultId": <resultId>,
        "maxRecords": <maximumNumberOfRecords>,
        "sortedBy": {
            "piId: <performanceIndicatorId>,
            "column": "number", "amount", "average"
        }
    }
}
```

- "uid" (mandatory): Uid to identify the reporting query, to get the result from.
- "data:resultId" (mandatory): Id of the result.
- "data:maxRecords": Maximum number of records to print.
- "data:sortedBy:piId" (mandatory): Set to -1 if no sorting should be applied.
- "data:sortedBy:column": Mandatory only if piId is larger than 0, indicates by which column of performance indicator that the result should be sorted.

Returns

A HTTP response with the following content:

```
{
    "dimension": "<dimension>",
    "decimals":  <decimals> ,
    "reportingQueryName": "<reportingQueryName>",
    "running": true | false,
    "queryResult": {
        "tableHeader": {
            "genericColumns": ["<groupingAttribute>", "comment", "timestampAnalysis", "accounts"],
            "indicatorColumns": ["number", "amount", "average"],
            "indicators": {
                 < piId > : {
                    "name": "<name>",
                    "selectedColumns": ["number", "amount", "average"],
                }
            }
        },
        "tableContent": [
            "comment": "<comment>",
            "groupingValue": "<groupingValue>",
            "indicators": {
                 < piId > : {
                    "number": <number> ,
                    "amount": <amount> ,
                    "average": <average>
                },
                ...,
            },
            "period": <period>
        ],
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true | false
}
```

283 GetReportingQueryResultMetaInfo

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingQueryResultMetaInfo",
    "uid": <ReportingQueryUid>,
    "data": {
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the reporting query.
- "resultId" (mandatory): The query result id to get meta information from.

Returns

A HTTP response with the following content:

```
{
    "ReportingQueryName": "<ReportingQueryName>",
    "groupingAttribute": {
        "name": "<groupByAttributeName>",
        "format": "<format>" | "None",
        "dataType": "<dataType>",
        "masked": 0|1
    },
    "timestampAnalysisEnabled", 0|1,
```

```
    "timestampAttribute": {
        ... ...
    },
    "accountAnalysisEnabled": 0|1,
    "rulePerformanceEnabled": 0|1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

284 GetReportingQueryResults

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getReportingQueryResults",
    "uid": <ReportingQueryUid>
}
```

- "uid" (mandatory): Uid to identify the group by query, to get the results from.

Returns

A HTTP response with the following content:

```
{
    "availableResults": [
        [
            <resultId>,
            <startedOnTimestamp>,
            <finished>"Yes" | "No",
            "<dataSelectionConditions>",
            <dataRangeFromUrid>,
            <dataRangeToUrid>,
            <timingAnalysis>true|false,
            <accountAnalysis>true|false
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

285 GetRevisionAuditTrail

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRevisionAuditTrail",
    "uid": <elementUid>,
    "type": "element" | "filter",
    "revision": <revisionUid>,
    "data": {
        "value": "<value>",
        "comparisonResult": "Added" | "all" | "ChangedComputationImpact" | "ChangedNoImpact" | "Removed",
        "entryType": "Attribute"| "Collusion"| "Counter"| "DeviceIdentification"| "Event"| "Formula"| "Index"| "List"|
"Mapping"| "Masterdata"| "Merging"| "Notification"| "NumericCategory"| "Pattern"| "Precedent"| "Profile"| "Rule"| "Ruleset"|
"PmmlModel"| "Stencil"| "TextCategory"| "UnknownType"| "Value" | "all", "changeRestriction":
            {
                "from": <fromTimestamp>,
                "to": <toTimestamp>
            }
        "users": [<userUid>, ...]
    }
}
```

- "uid": Uid of the element the audit trail belongs to.
- "type": "element" to request an audit trail for a single element specified by uid or "filter" for the general revision audit trail.
- "revision": Uid of the revision of the audit trail.
- "data:value" (optional): A search value to filter the general revision audit trail when using type "filter".
- "data:comparisonResult" (optional): The type of change to be retrieved when using type "filter".

- "data:entryType" (optional): The entry types to be retrieved when using type "filter".
- "data:changeRestriction:from" (optional): A start timestamp to restrict the audit trail entries by time when using type "filter".
- "data:changeRestriction:to" (optional): An end timestamp to restrict the audit trail entries by time when using type "filter".
- "data:users" (optional): A list of users to filter the audit trail when using type "filter".

Returns

A HTTP response with the following content:

```
{
    "revisionNumber": "<revisionNumber>",
    "editingUser": <userUid>,
    "auditTrail": <auditTrail>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

286 GetRevisionAuditTrailTable

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getRevisionAuditTrailTable",
    "revision": <revisionUid>
}
```

- "revision": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "revisionNumber": "[<reivsionNumber] '<revisionName>'",
    "tableContent": [
        [ ... ],
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

287 GetRevisionGeneral

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRevisionGeneral",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which information should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "revisionGeneral": {
        "name": "<revisionName>,
        "comment": "<revisionComment>",
        "status": "<status>",
        "editedBy": <editedByUserUid>|-1,
        "editedByName": "<editedByUserName>" | "",
        "editedSince": <editedSinceTimestamp>,
        "lastChangedBy": <lastChangedByUserUid>|-1,
        "lastChangedTimestamp": <lastChangedTimestamp>,
        "setLifeBy": "<setLifeByUserName>" | "",
        "setLifeTimestamp": <setLifeTimestamp>,
        "retiredBy": "<retiredByUserName>" | "",
        "retiredTimestamp": <retiredTimestamp>
```

```
},
"ddcStatistics": {
    "inputsMemory": "<inputsMemory>",
    "numberOfInputs": "<numberOfInputs>",
    "outputsMemory": "<outputsMemory>",
    "numberOfOutputs": "<numberOfOutputs>",
    "listsMemory": "<listsMemory>",
    "numberOfLists": "<numberOfLists>",
    "indexesMemory": "<indexesMemory>",
    "numberOfIndexes": "<numberOfIndexes>",
    "masterdataMemory": "<masterdataMemory>",
    "numberOfMasterdatas": "<numberOfMasterdatas>",
    "deviceIdentificationsMemory": "<deviceIdentificationsMemory>",
    "numberOfDeviceIdentifications": "<numberOfDeviceIdentifications>",
    "precedentsMemory": "<precedentsMemory>",
    "numberOfPrecedents": "<numberOfPrecedents>",
    "profilesMemory": "<profilesMemory>",
    "numberOfProfiles": "<numberOfProfiles>",
    "patternsMemory": "<patternsMemory>",
    "numberOfEvents": "<numberOfPatterns>",
    "eventsMemory": "<eventsMemory>",
    "numberOfEvents": "<numberOfEvents>",
    "countersMemory": "<countersMemory>",
    "numberOfCounters": "<numberOfCounters>",
    "collusionsMemory": "<collusionsMemory>",
    "numberOfCollusions": "<numberOfCollusions>",
    "formulasMemory": "<formulasMemory>",
    "numberOfFormulas": "<numberOfFormulas>",
    "totalMemory": "<totalMemory>"
},
"mdcStatistics":{
    "inputsMemory": "<inputsMemory>",
    "numberOfInputs": "<numberOfInputs>",
    "outputsMemory": "<outputsMemory>",
    "numberOfOutputs": "<numberOfOutputs>",
    "listsMemory": "<listsMemory>",
    "numberOfLists": "<numberOfLists>",
    "indexesMemory": "<indexesMemory>",
```

```
        "numberOfIndexes": "<numberOfIndexes>",
        "masterdatasMemory": "<masterdataMemory>",
        "numberOfMasterdatas": "<numberOfMasterdatas>",
        "deviceIdentificationsMemory": "<deviceIdentificationsMemory>",
        "numberOfDeviceIdentifications": "<numberOfDeviceIdentifications>",
        "precedentsMemory": "<precedentsMemory>",
        "numberOfPrecedents": "<numberOfPrecedents>",
        "profilesMemory": "<profilesMemory>",
        "numberOfProfiles": "<numberOfProfiles>",
        "patternsMemory": "<patternsMemory>",
        "numberOfEvents": "<numberOfPatterns>",
        "eventsMemory": "<eventsMemory>",
        "numberOfEvents": "<numberOfEvents>",
        "countersMemory": "<countersMemory>",
        "numberOfCounters": "<numberOfCounters>",
        "collusionsMemory": "<collusionsMemory>",
        "numberOfCollusions": "<numberOfCollusions>",
        "formulasMemory": "<formulasMemory>",
        "numberOfFormulas": "<numberOfFormulas>",
        "totalMemory": "<totalMemory>"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

288 GetRevisions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRevisions",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): The mandator for which revisions should be fetched.

Returns

A HTTP response with the following content:

```
{
    "revisions": [{
        "name": "[<revisionId>] \'<revisionName>\'",
        "uid": <revisionUid>,
        "status": <status>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

Revision status value mapping:

```
  0 means revision status of "Challenger"
  1 means revision status of "Golive"
  2 means revision status of "Champion"
  3 means revision status of "Retired"
  4 means revision status of "Invalidated"
  5 means revision status of "Golive waiting for confirmation"
```

289 GetRevisionSelectionTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRevisionSelectionTable",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator for which revisions should be fetched.

Returns

A HTTP response with the following content:

```
{
    "mandatorName": "<mandatorName>",
    "tableContent": [[
        <revisionUid>,
        <editingUserUid>,
        <revisionNumber>,
        "<revisionName>",
        "<revisionComment>",
        "<revisionStatus>",
        "<editingUserName>",
        <editingSinceTimestamp>,
        "<lastChangedBy>",
        <lastChangedTimestamp>,
        "<setLifeBy>",
```

```
        <setLifeTimestamp>,
        "<retiredBy>",
        <retiredTimestamp>,
        <ifNotContainSubMandator)>true|false
        ],
    ... ...
    ],
    "refresh": <autoRefreshSettingInSecondsForModelGolive>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

- If the revision is not edited by any user, the "editingUserUid" is set to -1.
- Available values for "revisionStatus" are: "Challenger", "Golive", "Champion", "Retired", "Invalidated", "Waiting".

290 GetRevisionStatus

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRevisionStatus",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which the status should be checked.

Returns

A HTTP response with the following content:

```
{
    "revisionStatus": {
        "reserved": true|false,
        "isChallenger": true|false,
        "challengerStatus": "Simulate" | "Analyze" | "Ready" | "GenerateRules",
        "status": "Challenger" | "Golive" | "Champion" | "Retired" | "Invalidated" | "Waiting",
        "name": "<revisionName>",
        "revision": <revisionNumber>,
        "mandatorName": "<mandatorName>",
        "mayDefineFinalRulesets": true|false,
        "autoRefreshAnalysis": <autoRefreshSetting>,
        "simulationValid": true|false,
        "averageSimulationPerformance": <averageSimulationPerformance>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

291 GetRoles

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRoles",
    "mandator": <mandatorUid>
}
```

- "mandator": Uid of the mandator for which the roles are retrieved.

Returns

A HTTP response with the following content:

```
{
    "roles": [<role>, ...],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

292 GetRolesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRolesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <roleUid>,
        <mandatorUid>,
        "<mandatorName>",
        "<roleName>",
        "<roleComment>",
        <numberOfPrivileges>,
        <numberOfUsers>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

293 GetRuleGenerationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationDataSelection",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which rule generation data selection should be fetched.

Returns

A HTTP response with the following content:

```
{
    "simulationAttributesAvailable": true|false,
    "missingAttributes": [ ... ... ],
    "performanceTableProgress": "<performanceTableProgress>",
    "verify": true|false,
    "generationScenario": "FromScratch" | "CurrentRevisionOnly" | "CurrentRevisionAndHistory",
    "trainingDataSelectionStream": {
        "mandators": [ ... ... ],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
```

```
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "incDdc": true|false,
        "conditions": [ ... ... ]
    },
    "verificationDataSelectionStream": {
        "mandators": [<mandatorUid>],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
```

```
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "incDdc": true|false,
        "conditions": [ ... ... ]
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

294 GetRuleGenerationDesignTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationDesignTable",
    "revision": <revisionUid>,
    "data": {
        "showNonSelectedIndicatorAttributes": true|false
    }
}
```

- "revision" (mandatory): Uid of the revision that the rule generation design table should be fetched from.
- "data" (optional):
- "data:showNonSelectedIndicatorAttributes" (optional): A boolean that determins if non-selected indicator attributes should be retrieved. The default value is false.

Returns

A HTTP response with the following content:

```
{
    "stopCondition": "<stopCondition>",
    "noConditionProposedReason: "<noConditionProposedReason>",
    "decimalsAmount": <decimalsAmount>,
    "ignoredConditionContainingEncryptedAttribute": true|false,
    "numberOfGeneratedConditions": <numberOfGeneratedConditions>,
    "numberOfUsedAttributes": <numberOfUsedAttributes>,
    "numberOfGeneratedRules": <numberOfGeneratedRules>,
```

```
    "tableContent": [[
        ... ...
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

295 GetRuleGenerationExplore

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationExplore",
    "uid": <attributeUid>,
    "type": "all" | "",
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of an attribute for which indicators should be returned.
- "type" (mandatory): Filter type: if "all", all indicators are returned. If not set, only indicators for which training results contain fraudulent records are returned.
- "revision" (mandatory): Uid of the revision of the attribute.

Returns

A HTTP response with the following content:

```
{
    "exploreAttribute": {
        "name": "<attributeName>",
        "modelUsage": "NoUsage" | "Categoric" | "Quantise" | "Logarithmic" | "Interval" | "Integer" | "CustomInterval" |
"CustomCategoric", "type": "attributeType", "dimension": "attributeDimension", "decimals": <decimals>
    },
    "amountAttribute": {
        "dimension": "<dimension>",
        "decimals": <decimals>
```

```
    },
    tableContent: [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

296 GetRuleGenerationPerformance

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationPerformance",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which the generation performance should be returned.

Returns

A HTTP response with the following content:

```
{
    "generationScenario": "FromScratch" | "CurrentRevisionOnly" | "CurrentRevisionAndHistory",
    "training": {
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
```

```
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "minimumUrid": <minimumUrid>,
        "maximumUrid": <maximumUrid>,
        "minimumSystemtime": <minimumSystemTimestamp>,
        "maximumSystemtime": <maximumSystemTimestamp>
    },
    "verification": {
        "actual": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "desired": {
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "minimumUrid": <minimumUrid>,
        "maximumUrid": <maximumUrid>,
        "minimumSystemtime": <minimumSystemTimestamp>,
```

```
        "maximumSystemtime": <maximumSystemTimestamp>
    },
    "numberOfGeneratedRules": <numberOfGeneratedRules>,
    "numberOfGeneratedConditions": <numberOfGeneratedConditions>,
    "tableContent": [
        [ ... ... ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

297 GetRuleGenerationRules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationRules",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which a table of generated rules should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "amountDimension": "<amountDimension>",
    "verify": "true|false",
    "tableContent": [
        [ ... ... ],
        .. ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

298 GetRuleGenerationSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationSettings",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which rule generation settings should be returned.

Returns

A HTTP response with the following content:

```
{
    "performanceTableProgress": "<performanceTableProgress>",
    "balanceFraudFalsePositives": <balanceFraudFalsePositives>,
    "lowerBoundFalsePositives": <lowerBoundFalsePositives>,
    "upperBoundFraudHitRate": <upperBoundFraudHitRate>,
    "stopRuleFalsePositives": <stopRuleFalsePositives>,
    "maxNumberConditions": <maxNumberConditions>,
    "minPopulationQuantise": <minPopulationQuantise>,
    "minSampleSizePercent": <minSampleSizePercent>,
    "minSelectHitRate": <minSelectHitRate>,
    "minConditionImprovement": <minConditionImprovement>,
    "relaxAllUpAutoThreshold": <relaxAllUpAutoThreshold>,
    "maxNumberRules": <maxNumberRules>,
    "maxFraudHit": <maxFraudHit>,
```

```
    "rulesName": "<rulesName>",
    "rulesComment": "<rulesComment>",
    "aPrioriConditions": [ <conditions> ],
    "ruleConclusion": {
        "interceptValue": <interceptValue>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

299 GetRuleGenerationStatus

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationStatus",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which the status should be retrieved.

Returns

A HTTP response with the following content:

```
{
    "simulationAttributesAvailable": true|false,
    "missingAttributes": [
        ... ...
    ],
    "verify": true|false,
    "performanceTableProgress": true|false,
    "designerTableProgress": true|false,
    "automaticRulegenerationRunning: true|false,
    "refresh": <autoRefreshSettingInSecondsForModelElementGeneration>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

300 GetRuleGenerationTargets

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleGenerationTargets",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision for which target rulesets should be returned.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [{
        "uid": <rulesetUid>,
        "name": "<rulesetName>",
        "comment": "<comment>",
        "revision": <revisionUid>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

301 GetRulePerformance

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRulePerformance",
    "uid": <ruleUid>,
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>
    }
}
```

- "uid" (mandatory): Uid of the rule that the rule performance result is from.
- "revision" (mandatory): Uid of the revision in which the analysis has been executed.
- "data:analysisId" (mandatory): Id of the analysis that the rule performance results should be retrieved from.

Returns

A HTTP response with the following content:

```
{
    "metaAmountDimension": "<dimension>",
    "metaAmountDecimals": <decimal>,
    "conditionsShort": "<conditions>",
    "conclusionsShort": "<conclusions>",
    "graphDataDaily": {
        "totalFraud": [<value>],
```

```
        "numberFraud": [<value>],
        "amountFraud": [<value>],
        "numberGenuine": [<value>],
        "amountGenuine": [<value>],
        "timestamp": [<timestamp>]
    },
    "graphDataWeekly": {
        "totalFraud": [<value>],
        "numberFraud": [<value>],
        "amountFraud": [<value>],
        "numberGenuine": [<value>],
        "amountGenuine": [<value>],
        "timestamp": [timestamp]
    },
    "data": {
        "simulationValid": true|false,
        "simulationEnabled": true|false,
        "isDataInIris": true|false,
        "simulationStatus": "<status>",
        "simulationProgress": <progress>,
        "progress": <progress>,
        "recordsProcessed": <numberOfRecordsProcessed>,
        "optimizationProgress": <progress>,
        "manRecordsProcessed": <manRecordsProcessed>,
        "finished": true|false,
        "optimizing": true|false,
        "uridFrom": <fromUrid>,
        "uridTo": <toUrid>,
        "refresh": <autoRefreshSetting>,
        "analysisResultsValid": true|false,
        "analysisStatus": "<status>"
    },
    "name": "<ruleName>",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

302 GetRulePerformanceRulesets

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRulePerformanceRulesets",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>,
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analysis has been executed.
- "data:analysisId" (mandatory): Id of the analysis that the enabled rulesets are retrieved from.

Returns

A HTTP response with the following content:

```
{
    "rulePerformanceRulesets": [{
        "name": "<rulesetName>",
        "uid": <rulesetUid>,
        "rules": [{
            "uid": <ruleUid>,
            "name": "<ruleName>"
            },
        ... ...
```

```
        ]},
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

303 GetRuleReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleReport",
    "uid": <ruleUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "withoutContext": true|false
    }
}
```

- "uid" (mandatory): Uid of the rule for which the rule report result is requested.
- "mandator" (mandatory): Uid of the mandator in which the rule report is executed.
- "revision" (mandatory): Uid of the revision in which the rule report is executed.
- "data:withoutContext" (mandatory): A flag of whether or not the rule report uses the current simulation or starts a new simulation for the relevant elements.

Returns

A HTTP response with the following content:

```
{
    "ruleName": "<ruleName>",
    "ruleComment": "<ruleComment>",
    "rulePriority": <rulePriority>,
    "ruleEnabled": <enabled>1|0,
    "rulesetName": "<rulesetName>",
    "rulesetComment": "<rulesetComment>",
    "rulesetPriority": <rulesetPriority>,
    "rulesetEnabled": <enabled>1|0,
    "rulesetConditions": [
        <rulesetConditions>
    ],
    "mandator": "<mandatorName>",
    "revisionName": "[<revisionId] \'<revisionName>\'",
    "ruleConditions": [
        <ruleConditions>
    ],
    "ruleActions": [
        <ruleActions>
    ],
    "userName": "<userName>",
    "generationTimestamp": "<timestamp>",
    "amountDimension": "<amountDimension>",
    "amountDecimals": <decimals>,
    "interceptAttributeName": "<interceptAttributeName>",
    "simulationMethod": "<simulationMethod>",
    "simulationPeriodType": "<simulationPeriodType>",
    "simulationNumberOfTrx": <simulationNumberOfTransactions>,
    "simulationTotalAmount": <simulationTotalAmount>,
    "simulationDatarangeFrom": <simulationDataRangeFrom>,
    "simulationDatarangeTo": <simualtionDataRangeTo>,
    "simulationConditions ": [
        <simulationConditions>
    ],
    "summary":{
        <summary>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
```

```
    "reloadUserProfile": true|false
}
```

304 GetRulesetsForAnalysis

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "getRulesetsForAnalysis",
    "revision": <revisionUid>
}
```

- "revision": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "rulesets": [
        {
            "uid": <rulesetUid>,
            "name": "<rulesetName>",
            "simulated": true|false
        },
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

305 GetRulesStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRulesStatistics",
    "revision": <revisionUid>,
    "data": {
        "rulesets": [<rulesetUid>],
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision where the analysis has been executed.
- "data:rulesets" (mandatory): Uids of a list of rulesets for which the result of rule analysis is requested.
- "data:analysisId" (mandatory): Id of the analysis for which the results of rule analysis are requested.

Returns

A HTTP response with the following content:

```
{
    "ruleStatistics": [[
        <ruleUid>,
        <rulesetUid>,
        <revisionUid>,
        <hitRate>,
        <falseAlarmRatio>,
```

```
        <savedAmountPerFalseAlarm>,
        <fraudAmountIntercepted>,
        <fraudTransactionsIntercepted>,
        <genuineAmountIntercepted>,
        <genuineTransactionsIntercepted>,
        "<rulesetName>",
        "<ruleName>"
        ],
    ... ...
    ],
    "refresh": <autoRefreshSetting>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

306 GetRulesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRulesTable",
    "revision": <revisionUid>,
    "ruleset": <rulesetUid>
}
```

- "revision" (mandatory): Uid of the revision where the rules table is requested.
- "ruleset" (mandatory): Uid of the ruleset that all the rules belong to.

Returns

A HTTP response with the following content:

```
{
    "ruleSetName": "<rulesetName>",
    "ruleSetConditions": [
        { <condition> },
        ... ...
    ],
    "enabled":0|1,
    "tableContent": [[
        <ruleUid>,
        <revisionUid>,
        <priority>,
```

```
        "<name>",
        "<comment>",
        "Yes" | "No",
        "<conditionsShortString>",
        "<conclusionsShortString>"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

307 GetRuleStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRuleStatistics",
    "revision": <revisionUid>,
    "data": {
        "rule": "ruleUid",
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision where the single rule analysis has been executed.
- "data:rule" (mandatory): Uid of the rule for which the result of single rule analysis is requested.
- "data:analysisId": Id of the analysis for which the result of single rule analysis is requested.

Returns

A HTTP response with the following content:

```
{
    "ruleStatistics": {
        "dimension": "<dimension>",
        "fraudHit": <fraudHitRate>,
        "falseAlarms": <falseAlarms>,
        "SAPFA": <savedAmountPerFalseAlarm>,
        "fraudAmountIntercepted": <fraudAmountIntercepted>,
```

```
        "fraudTrxIntercepted": <fraudTransactionsIntercepted>,
        "genuineAmountIntercepted": <genuineAmountIntercepted>,
        "genuineTrxIntercepted": <genuineTransactionsIntercepted>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

308 GetRunningSimulations

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getRunningSimulations",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision for which the request is performed.

Returns

A HTTP response with the following content:

```
{
    "runningSimulations": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

309 GetSandboxTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSandboxTable",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision where the sandbox table is requested.

Returns

A HTTP response with the following content:

```
{
    "header": [{
        "key": "attributeUid"
        },{
        "key": "Attributes"
        },
    ... ...
    ],
    "tableContent": [[
        ... ...
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

310 GetSearchComplianceListEntriesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSearchComplianceListEntriesTable",
    "data": {
        "selectedTypes": <selectedTypes>,
        "entityType": "Individual"|"Legal",
        "listId": <listId>,
        "name": "<name>",
        "street": "<street>",
        "city": "<city>",
        "country": "<country>",
        "passport": "<passport>",
        "birthdate": "<birthdate>"
    }
}
```

- "data" (mandatory): Contains a compliance list type filter and the values to search for.
- "data:selectedTypes" (mandatory): Filters the type of compliance lists to search through. Available values for the "selectedTypes" are: "OfacSdn", "United↩ NationsList", "PoliticalExposedPersonList", "EuropeanTerrorList", "RussianTerrorList".
- "data:entityType" (mandatory): Determines if individual or legal entries should be searched.
- "data:listId": The list id to search for.
- "data:name": The name to search for.
- "data:street": The street to search for.
- "data:city": The city to search for.
- "data:passport": The passport number to search for.
- "data:birthdate": The date of birth to search for.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [
        ... ...
    ],
    "resultId": <resultId>,
    "numberOfEntriesShown": <numberOfEntriesInTable>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

311 GetSessionCountdownData

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSessionCountdownData"
}
```

Returns

A HTTP response with the following content:

```
{
    "sessionCountdownThreshold": <sessionCountdownThreshold>,
    "sessionCountdownRefresh": <sessionCountdownRefresh>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

312 GetSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSettings"
}
```

Returns

A HTTP response with the following content:

```
{
    "data": {
        "streamType": "settings",
        "settings": {
            "application": {
                "name": "<applicationName>",
                "serverTimeZone": <serverTimezoneSetting>
            },
            "memory": {
                "limit": <mainMemoryUsageLimit>
            },
            ... ...
        },
        ... ...
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

313 GetSimulationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSimulationDataSelection",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the simulation is defined.

Returns

A HTTP response with the following content:

```
{
    "simulationDataSelection": {
        "mandators": [<mandatorUid>,...],
        "periodType": "RecordsAbsolute" | "RecordsRelative" | "TimeAbsolute" | "TimeRelative",
        "actual":{
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
```

```
        "desired":{
            "fromUrid": <fromUrid>,
            "fromRecord": <fromRecord>,
            "fromTimestamp": <fromTimestamp>,
            "fromDays": <fromDays>,
            "toUrid": <toUrid>,
            "toRecord": <toRecord>,
            "toTimestamp": <toTimestamp>,
            "toDays": <toDays>
        },
        "incDdc": true|false,
        "conditions": [ ... ]
    },
    "simulationMethod": "PerRecord" | "PerChunk",
    "metaSystemTime": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

314 GetSimulationProgress

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSimulationProgress",
    "revision": <revisionUid>
}
```

- "revision": The revision for which simulation status should be returned.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "simulationValid": true|false,
        "simulationEnabled": true|false,
        "isDataInIris": true|false,
        "simulationStatus": "SimDisabled" | "SimInitializing" | "SimComputing" | "SimReady" | "SimStopping",
        "simulationProgress": <simulationProgress>,
        "refresh": <autoRefreshSetting>,
        "singleRuleSimulation": true|false,
        "simulationInitiatedAt": <simulationInitiatedAtTimestamp>,
        "simulationProcessingStartedAt": <simulationProcessingStartedAtTimestamp>,
        "simulationProcessingFinishedAt": <simulationProcessingFinishedAtTimestamp>,
        "simulationProcessedRecords": <numberOfRecordsProcessedBySimulation>,
        "transactionsPerSecondAverage": <averageTransactionsPerSecond>,
```

```
        "averageProcessingTime": <averageProcessingTime>,
        "estimatedFinishingTimestamp": <estimateFinishingTimestamp>
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

315 GetSimulationTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSimulationTable",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision from which the simulation query table is requested.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <simulationQueryUid>,
        <mandatorUid>,
        "<simulationQueryType>",
        <indexUid>|-1,
        "<indexName" | "",
        "<mandatorName>",
        "<simulationQueryName>",
        "<comment>",
        "Yes" | "No",
        <numberOfColumns>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>
```

```
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

316 GetStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getStatistics",
    "revision": <revisionUid>,
    "data": {
        "analysisId": <analysisId>
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analysis was executed.
- "data:analysisId" (mandatory): Id of the analysis for which the statistics are requested.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

317 GetStatusAlarmIndicatorsDisplay

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getStatusAlarmIndicatorsDisplay"
}
```

Returns

A HTTP response with the following content:

```
{
    "statusAlarmIndicators": [{
        "streamType": "statusAlarmIndicator",
        "uid": <statusAlarmIndicatorUid>,
        "showOnDashboard": true|false,
        "position": <position>,
        "instance": <instanceId>,
        "type": <alarmType>,
        "status": true|false,
        "replacements": {
            "{alarmTimestamp}": "",
            "{comment}": "",
            "{externalMessage": "",
            "{instanceId}": "<instanceId>",
            "{instanceName": "<instanceName>",
            "{integerVAlue}": "<integerValue>",
            "{lastUpdateTimestamp}": "<lastUpdateTimestamp>",
            "{mandator}": "<mandatorName>",
```

```
            "{name}": "<statusAlarmIndicatorName>",
            "{thresholdAbove}": "<thresholdAboveValue>",
            "{thresholdBelow}": "<thresholdBelow>",
            "{value}": "<value>"
            },
        "displayText": "<displayText>",
        "displayTooltip": "<displayTooltip>"
        },
    ... ...
    ],
    "saiConfig": {
        "explanation": "<explanation>",
        "tooltip": "<toolTip>",
        "onlineHelpType": "<onlineHelpType>",
        "onlineHelp": "<onlineHelp>"
    },
    "refresh": <autoRefreshSettingInSecondsForDashboard>",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

318 GetStatusAlarmIndicatorsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getStatusAlarmIndicatorsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <statusAlarmIndicatorUid>,
        <enabled>"Yes" | "No",
        <mandatorUid>,
        "<statusAlarmIndicatorName>",
        "<statusAlarmIndicatorComment>",
        "<alarmType>",
        "<alarmStatus>",
        <position>,
        "<lastChangedByUserName>",
        <lastchangedOnTimestamp>
        ],
    ... ...
    ],
    "nextPosition": <nextPosition>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

319 GetSystemInformation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSystemInformation"
}
```

Returns

A HTTP response with the following content:

```
{
    "systemInternals": [
        ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

320 GetSystemLogTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getSystemLogTable",
    "data": {
        "from": <fromTimestamp>,
        "to": <toTimestamp>,
        "instances": [<instanceUid>],
        "users": [<userUid>]
    }
}
```

- "data" (optional): Contains values to filter the retrieved log messages.
- "data:from" (optional): If set, only log messages that occurred after this timestamp are retrieved.
- "data:to" (optional): If set, only log messages that occurred before this timstamp are retrieved.
- "data:instances" (optional): A list of IrisInstance objects (identified by their uids) to retrieve log messages for.
- "data:users" (optional): A list of User objects (identified by their uids) to retrieve log messages for.

Returns

A HTTP response with the following content:

```
{
    "recordLimit": <maximumNumberOfRecords>,
    "tableContent": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

321 GetTargetRevisions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTargetRevisions",
    "uid: ": <revisionElementUid>,
    "type": <revisionElementType>["modelling" | "input" | "output" | "list" | "index" | "merging" | "masterdata" |
"deviceIdentification" | "precedent" | "profile" | "pattern" | "event" | "counter" | "collusion" | "formula" | "ruleset" |
"finalRuleset"],
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the revision element that is copied.
- "type" (mandatory): The type of the revision element.
- "revision" (mandatory): Uid of the revision where the element is copied from.

Returns

A HTTP response with the following content:

```
{
    "revisions": [[
        "<mandatorName>",
        <mandatorUid>,
        "<[revisionId> \'<revisionName>\'",
        <revisionUid>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

322 GetTextModules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTextModules",
    "mandator": <mandatorUid>
}
```

- "mandator" (optional): The Mandator (identified by his uid) TextModules are requested for.

Returns

A HTTP response with the following content:

```
{
    "textModules": [{
        "uid": <textModuleUid>,
        "name": "<textModuleName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

323 GetTextModulesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTextModulesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <textModuleUid>,
        <mandatorUid>,
        "<mandatorName>",
        <enabled>"Yes" | "No",
        "<textModuleName>",
        "<comment>",
        "<textTemplate>",
        "<lastChangedByUserName>",
        <lastChangedTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

324 GetThreadPriorities

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getThreadPriorities"
}
```

Returns

A HTTP response with the following content:

```
{
    "data": [
        "<priority_1>",
        "<priority_2>",
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

325 GetTickerEntries

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTickerEntries"
}
```

Returns

A HTTP response with the following content:

```
{
    "streamType": "tickerData",
    "maxEntries": <maxTickerEntriesSetting>,
    "tickerHighlightInterval": <tickerHighLightInterval>,
    "tickerRefreshInterval": <tickerRefreshInterval>,
    "lastLogOut": <lastLogOutTimestamp>,
    "lastLogIn": <lastLogInTimestamp>,
    "tickerEntriesList": [{
        "streamType": "tickerEntry",
        "uid": <entryId>,
        "name": "<userName>",
        "date": <timestamp>,
        "title": "<tickerTitle>",
        "note": "<tickerNote>",
        "mandatorUid": <mandatorUid>,
        "mandatorUids": [<mandatorUid>],
        "fromCppTable": true|false,
        "cppId": <cppId>,
```

```
        "cppCaseGroupUid": <cppCaseGroupUid>,
        "cppMandatorUid": <cppMandatorUid>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

326 GetTimestampAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTimestampAttributes",
    "type": <type>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "includeDdc": true|false
    }
}
```

- "type" (optional): Describes the context in which the request was sent. Available values for the "type" are: "groupByQuery", "deviceIdentification", "event", "merchantMonitoringTemplate", "profile", "sequence", "reportingQuery".
- "mandator" (optional): Uid of the mandator to retrieve attributes from. Only used for the types of "groupByQuery", "reportingQuery" and "merchantMonitoring↩ Template".
- "revision" (optional): Uid of the revision to retrieve attributes from. Only used when the type is not "groupByQuery", "reportingQuery" or "merchantMonitoring↩ Template".
- "data" (optional): Only used when the type is "groupByQuery" or "reportingQuery".
- "data:includeDdc" (optional): Normally only attributes stored in MDC are retrieved. If set to true, attributes stored in DDC are also retrieved.

Returns

A HTTP response with the following content:

```
{
    "timestampAttributes": [
        ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

327 GetToAddressMeta

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getToAddressMeta",
    "mandator": <mandatorUid>,
    "data": {
        "caseUid": <caseUid>
    }
}
```

- "mandator" (mandatory): Uid of the mandator to retrieve the meta attribute from.
- "data:caseUid" (mandatory): Uid of the investigation case to retrieve the meta attribute value from.

Returns

A HTTP response with the following content:

```
{
    "toAddress": <toAddress>,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

328 GetTransactionMessageReportTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getTransactionMessageReportTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "header": [
        <MessageTypeId>,
        ... ...
    ],
    "tableContent": [
        [
            <reportingPeriod1>,
            <year>,
            <totalMessagesForMessageTypeId>,
            ... ...
            <totalMessagesInReportingPeriod1>
        ],[
            <reportingPeriod2>,
            <year>,
            <totalMessagesForMessageTypeId>,
            ... ...
            <totalMessagesInReportingPeriod2>
```

```
        ],
        ... ...
        [
            -1,
            -1,
            <totalMessagesForMessageTypeId>,
            ... ...
            <totalMessages>
        ]
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

329 GetUserGroups

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUserGroups",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator for which user groups shall be queried.

Returns

A HTTP response with the following content:

```
{
    "userGroups": [{
        "uid": <userGroupUid>,
        "name": "<userGroupName>",
        "comment": "<comment>",
        "users": [<userUid>,...],
        "lastChangedBy": "<lastChangedByUserName>",
        "lastChangedTimestamp": <lastChangedTimestamp>
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

330 GetUserGroupsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUserGroupsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <userGroupUid>,
        <mandatorUid>,
        "<mandatorName>",
        "<userGroupName>",
        "<comment>",
        <numberOfUsers>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

331 GetUsers

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUsers",
    "type": "<type>",
    "mandator": <mandatorUid>
}
```

- "type" (mandatory): The type of request. The available values for "type" are:

  ```
  "missedCasesReport" (requires "mandator")
  "audit_log"
  "system_log"
  "caseFilter"
  "risklistFilter"
  "definedRiskList" (requires "mandator")
  "userReport" (requires "mandator")
  ```

- "mandator": Uid of the mandator of an element.

Returns

A HTTP response with the following content:

```
{
    "users": [{
        "streamType": "user",
        "uid": <userUid>,
        "enabled": true|false,
        "login": "<loginName>",
        "name": "<userName>",
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

332 GetUsersAndMandatorsOfRoles

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUsersAndMandatorsOfRoles",
    "data": {
        "roles": [<roleUid>]
    }
}
```

- "data:roles" (optional): A list of role uids which shall be included.

Returns

A HTTP response with the following content:

```
{
    "roles": [{
        "role": "<roleName>",
        "user": "<userName>",
        "mandator": "<mandatorName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

333 GetUsersOfRole

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUsersOfRole",
    "uid": <roleUid>
}
```

- "uid" (mandatory): Uid of the role that is queried.

Returns

A HTTP response with the following content:

```
{
    "users": [{
        "streamType": "user",
        "uid": <userUid>,
        "enabled": true|false,
        "login": "<loginName>",
        "name": "<userName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

334 GetUsersReportsTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUsersReportsTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <reportUid>,
        "<mandatorName>",
        "<reportName>",
        "<comment>",
        "<lastChangedByUserName>",
        <lastChangedTimestamp>
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

335 GetUsersTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getUsersTable",
    "data": {
        "showDisabled": true|false
    }
}
```

- "data:showDisabled" (optional): Boolean value to include or exclude disabled users in the response.

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <userUid>,
        "<loginName>",
        "<userName>",
        "<comment>",
        "<enabled>true|false",
        "<associatedMandatorName>",
        <numberOfRoles>,
        <lastPasswordChangeTimestamp>,
        "<lastchangedByUserName>"
        <numberOfFailedLogin>,
```

```
        <lastChangedOn>,
        <lastLoginTimestamp>,
        true|false
        ],
    ... ...
    ],
    "mayDeleteUser": 0|1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

336 GetWorkingQueueCasesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getWorkingQueueCasesTable",
    "uid": <workingQueueUid>,
    "type": "public"|"private"
}
```

- "uid" (optional): Uid of a working queue. This is used when querying for one certain working queue.
- "type" (optional): "public" or "private". This is used when querying for all public/private working queues. Neither "uid" nor "type" have to be set. If both are omitted, all accessible working queues are queried.

Returns

A HTTP response with the following content:

```
{
    "header": [{
        "uid": "<uid>",
        "name": "<name>",
        "align": "<align>",
        "type": "<type>",
        "format": "<format>"
        },
    ... ...
    ],
```

```
    "tableContent": [{
        "uid":<caseId>,
        "Queue": "<caseQueueName>",
        "Generated": <generatedTimestamp>,
        "Score": <scoreValue>,
        "Hits": <numberOfHits>,
        "Status": "<status>",
        "FraudStatus": "<fraudStatus>",
        "CloseCodeName": "<caseCloseCodeName>",
        "User": <userUid>,
        "UserName": "<userName>",
        "LastUserAction": <lastUserActionTimestamp>,
        "MandatorName": "<mandatorName>",
        "Mandator": <mandatorUid>,
        "CppName": "<cppName>",
        "Cpp": "<cppCode>",
        ... ...
        },
    ... ...
    ],
    "responseStatus": ["<status>", "feedbacktext"],
    "reloadProfile": true|false
}
```

337 GetWorkingQueues

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getWorkingQueues",
    "type": "caseFilter"
}
```

- "revision" (mandatory): Uid of the revision that runs a simulation in which to execute the simulation query.
- "data" (mandatory): Either "ruleUid" or "parameter" have to be set.
- "data:parameter" (optional): The name of the simulation query.
- "data:ruleUid" (optional): Uid of the simulation query.

Returns

A HTTP response with the following content:

```
{
    "workingQueues": [{
        "uid": <workingQueueUid>,
        "name": "<workingQueueName>"
        },
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

338 GetWorkingQueuesTable

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getWorkingQueuesTable"
}
```

Returns

A HTTP response with the following content:

```
{
    "tableContent": [[
        <workingQueueUid>,
        <mandatorUid>,
        "<mandatorName>",
        <enabled>"Yes"|"No",
        "<workingQueueName>",
        "<comment>",
        <priority>,
        <numberOfUsers>,
        <numberOfQueueManagers>,
        "<lastChangedByUserName>",
        <lastChangedOnTimestamp>,
        "<conditions>"
        ],
    ... ...
    ],
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

339 GetWorkingQueueStatistics

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "getWorkingQueueStatistics",
    "type": "private"|"public"
}
```

- "type" (optional): Compute values for all "private" or "public" working queues. If "type" is omitted, values for all working queues are computed.

Returns

A HTTP response with the following content:

```
{
    "caseQueues": [{
        "name": "<caseClassName>",
        "value": "<numberofCases>"
        },
    ... ...
    ],
    "caseStates": [{
        "name": "<caseStateName>",
        "value": "<numberofCases>"
        },
    ... ...
    ],
    "general": {
```

```
        "numberOfCases": "<totalNumberOfCases>",
        "lead": "<lead>",
        "oldestOpen": "<ageOfOldestOpenCaseInDays>",
        "averageAge": "<averageAgeInDays>"
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

340 GoliveCheckAndReport

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "goliveCheckAndReport",
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "mandator" (mandatory): The mandator to which the challenger revision belongs.
- "revision" (mandatory): The challenger revision to be checked.

Returns

A HTTP response with the following content:

```
{
    "goliveReport": [
        ... ...
    ],
    "goliveEnabled": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

341 HasEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "hasEntry",
    "uid": <definedRiskListUid>,
    "data": {
        "value": "<entryInputAttributeValue>"
    }
}
```

- "uid" (mandatory): Uid of the defined risk list that is checked for the entry.
- "data:value" (mandatory): The value which is searched in the given defined risk list.

Returns

A HTTP response with the following content:

```
{
    "valueExists": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

342 InsertPin

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "insertPin",
    "data": {
        "login": <userLoginName>,
        "password": <userPassword>,
        "pin": <pin>
    }
}
```

- "data:login": User's login name.
- "data:password": User's login password.
- "data:pin": The one time password for the current user login session. The one time password is sent to the email address defined in the user's "Two factor authentication address" setting.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

343 InterruptCase

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "interruptCase",
    "uid": <caseUid>
}
```

- "uid" (mandatory): Uid of the case that is to be interrupted.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "streamType": "case",
        "case": {
            "uid": <caseUid>,
            "createdOnInstance": <instanceId>,
            "localCaseUid": <caseUid>,
            "mandator": {
                "uid": <mandatorUid>,
                "name": "<mandatorName>"
            },
            "caseQueue": {
                ... ...
            },
```

```
            ... ...
        },
        ... ...
    },
    "editable": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

344 InvestigateCase

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "investigateCase",
    "uid": <caseUid>,
    "data": {
        "asyncQueryResult": true|false
    }
}
```

- "uid" (mandatory): Uid of the case which is to be investigated.
- "data:asyncQueryResult" (optional): If true, only the query resultId will be returned, without waiting for the query to be completed. The query result needs to be queried with API request "getQueryResult" later. The default value is false.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "streamType": "case",
        ... ...
    },
    "editable": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

345 IsCasesTableEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isCasesTableEncrypted",
    "data": {
        "mandators": [
            <mandatorUid>,
            ... ...
        ],
        "caseQueues": [
            <caseQueueUid>,
            ... ...
        ],
        "dateRestriction": {
            "all": true|false,
            "from": <fromTimestamp>,
            "to": <toTimestamp>
        },
        "status": {
            "New": true|false,
            "Investigated": true|false,
            "Followup": true|false,
            "Due": true|false,
            "Closed": true|false,
            "Reopened": true|false
        },
        "users": [<userUid>],
        "scoreFrom": <caseScoreFromValue>,
        "scoreTo": <caseScoreToValue>,
        "cpps": ["<cppUid_caseGroupUid>"],
        "conditions: [...]
```

```
        }
}
```

- "data" (mandatory): Contains the filter criteria for the cases table.
- "data:mandators" (optional): Include only cases from these mandators.
- "data:caseQueues" (optional): Include only cases from these case queues.
- "data:dateRestriction:all" (optional): Boolean value to indicate whether cases of any age should be included.
- "data:dateRestriction:from" (optional): If not dateRestriction:all, include only cases younger than this value.
- "data:dateRestriction:to" (optional): If not dateRestriction:all, include only cases older than this value.
- "data:status:New" (optional): Boolean value to indicate whether cases of status New shall be included.
- "data:status:Investigated" (optional): Boolean value to indicate whether cases of status Investigated shall be included.
- "data:status:Followup" (optional): Boolean value to indicate whether cases of status Followup shall be included.
- "data:status:Closed" (optional): Boolean value to indicate whether cases of status Closed shall be included.
- "data:status:Reopened" (optional): Boolean value to indicate whether cases of status Reopened shall be included.
- "data:users" (optional): Include only cases which are investigated/on followup by these users.
- "data:scoreFrom" (optional): Include only cases that have a score higher than this value.
- "data:scoreTo" (optional): Include only cases that have a score lower than this value.
- "data:cpps" (optional): Include only cases which are assigned to these cpps.
- "data:conditions" (optional): Include only cases which satisfy these conditions.

If any of the lists of mandators, caseQueues and working queues is empty or if all status flags are false, no cases can be found.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

346 IsCppTableEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isCppTableEncrypted",
    "data": {
        "showAllEntries": true|false,
        "caseGroups": [<caseGroupUid>,...],
        "dateRestriction": {
            "all": true|false,
            "from": <fromTimestamp>,
            "to": <toTimestamp>
        },
        "status": {
            "New": true|false,
            "Investigated": true|false,
            "Followup": true|false,
            "Due": true|false,
            "Closed": true|false,
            "Reopened": true|false
        },
        "active": {
            "Active": true|false,
            "Inactive": true|false
        }
    }
}
```

- "data" (mandatory): Contains the filter criteria.
- "showAllEntries" (mandatory): Boolean flag to decide whether to include all accessable cpps.

- "caseGroups" (optional): Include only cpps which belong to these case groups.
- "dateRestriction:all" (optional): If set, include cpps of all age.
- "dateRestriction:from" (optional): If not dateRestriction:all, include only cpps younger than this value.
- "dateRestriction:to" (optional): If not dateRestriction:to, include only CPPs older than this value.
- "status:New" (optional): Include only cpps with status New.
- "status:Investigated" (optional): Include only cpps with status Investigated.
- "status:Followup" (optional): Include only cpps with status Followup.
- "status:Due" (optional): Include only cpps with status Due.
- "status:Closed" (optional): Include only cpps with status Closed.
- "status:Reopened" (optional): Include only cpps with status Reopened.
- "active:Active" (optional): Include active cpps.
- "active:Inactive" (optional): Include inactive cpps.

If "showAllEntries" is true, all other values can be omitted.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

347 IsDdcEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isDdcEnabled"
}
```

Returns

A HTTP response with the following content:

```
{
    "ddcEnabled": {
        "mergings": true|false,
        "merchantMonitoringRules": true|false,
        "preprocessings": true|false,
        "simulation": true|false,
        "query": true|false
        },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

348 IsDdcEnabledForManualCaseCreation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isDdcEnabledForManualCaseCreation"
}
```

Returns

A HTTP response with the following content:

```
{
    "caseCreationDdc": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

349 IsDefinedRiskListEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isDefinedRiskListEncrypted",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the checked defined risk list.
- "mandator" (mandatory): Uid of the mandator that the defined risk list belongs to.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

350 IsGoliveRunning

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isGoliveRunning"
}
```

Returns

A HTTP response with the following content:

```
{
    "goliveRunning": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

351 IsGroupByQueryResultEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isGroupByQueryResultEncrypted",
    "uid": <groupByQueryUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the asserted group by query.
- "data:downloadToken" (mandatory): The downloadToken for the asserted result.
- "data:resultId" (mandatory): Id of the group by query result to be checked.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

352 IsIndexSequenceMdcOutsideSimulation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isIndexSequenceMdcOutsideSimulation",
    "uid": <indexUid>,
    "revision": <revisionUid>,
    "data": {
        "sequenceType": "Source"|"Target"|"Both"
    }
}
```

- "uid": Uid of the index whose sequence shall be checked.
- "revision": Uid of the revision.
- "data:sequenceType" (optional): For peer indexes, determines whether "source" or "target" or "both" sequences shall be asserted.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "outsideSimulation": true|false
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

353 IsMetaAttributeInChampions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isMetaAttributeInChampions",
    "data": {
        "metaAttributeType": "<metaAttributeType>"
    }
}
```

- "data:metaAttributeType" (mandatory): The checked meta attribute type. The available values for the "metaAttributeType" are:

```
"Amount" "Timestamp" "Account" "Fraud" "Intercept" "Email" "CaseScore" "CaseQueue" "MessageTypeId" "Sequence"
    "SystemTime"
    "Notification"
    "PrimaryInstanceId"
    "PrimaryUrid"
    "URID"
    "UridComputationComplete"
    "Reminder"
    "Score"
```

Returns

A HTTP response with the following content:

```
{
    "metaAttributeExists": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

354 IsOffline

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isOffline",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance to be checked.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": "OK", "INSTANCE_IS_ONLINE" | "INSTANCE_IS_FINISHING_DEFERRED_WRITING" | "INSTANCE_IS_OFFLINE",
    "reloadUserProfile": true|false
}
```

355 IsPasswordValid

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isPasswordValid",
    "data": {
        "password": "<newPassword>"
    }
}
```

- "password" (mandatory): The password which is to be checked against policies.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

356 IsQueryResultEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isQueryResultEncrypted",
    "uid": <queryUid>,
    "revision": <revisionUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the query to be checked.
- "revision" (optional): Uid of the revision. It is only required when asserting a simulation query.
- "data:downloadToken" (mandatory): The downloadToken for the asserted query result.
- "data:resultId" (mandatory): Id of the query result from the given query that is to be checked.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

357 IsReportingQueryResultEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isReportingQueryResultEncrypted",
    "uid": <ReportingQueryUid>,
    "data": {
        "downloadToken": "<downloadToken>",
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the reporting query to be checked.
- "data:downloadToken" (mandatory): The downloadToken for the asserted result.
- "data:resultId" (optional): Id of the result from the given reporting query to be checked.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

358 IsRevisionDeletionEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isRevisionDeletionEnabled"
}
```

Returns

A HTTP response with the following content:

```
{
    "deletionEnabled": {
        "Challenger": true|false,
        "Invalidated": true|false,
        "Retired": true|false
    },
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

359 IsSamplingEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isSamplingEnabled"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

360 IsSimulationValid

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isSimulationValid",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision identifies the simulation which is checked.

Returns

A HTTP response with the following content:

```
{
    "simulationValid": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

361 IsValidInstanceId

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isValidInstanceId",
    "uid": <instanceId>
}
```

- "uid" (mandatory): Id of an instance to be checked.

Returns

A HTTP response with the following content:

```
{
    "isValid": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

362 IsWorkingQueueCasesTableEncrypted

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "isWorkingQueueCasesTableEncrypted",
    "uid": <workingQueueUid>,
    "type": "public"|"private"
}
```

- "uid" (optional): Uid of a working queue. This is used when querying for one certain working queue.
- "type" (optional): "public" or "private". This is used when querying for all public/private working queues. Neither "uid" nor "type" has to be set. If both are omitted, all accessible working queues are queried.

Returns

A HTTP response with the following content:

```
{
    "isEncrypted": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

363 LoadCaseAuditTrailsFromDisk

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "loadCaseAuditTrailsFromDisk",
    "uid": <caseUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

364 Login

Detailed Description

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "login",
    "data": {
        "login": <userLoginName>,
        "password": <userPassword>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

365 Logout

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "logout"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

366 MarkFraud

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "markFraud",
    "data": {
        "urid": [<urid>],
        "fraudValue": <manualFraudValue>,
        "fraudTimestampAttribute": <timestampAttributeUid>|-1
    }
}
```

- "data:urid" (mandatory): A list of URIDs which shall be marked as fraud.
- "data:fraudValue" (mandatory): The fraud value which shall be assigned to the selected URIDs.
- "fraudTimestampAttribute" (optional): Provide a fraud timestamp attribute to save the time of fraud marking.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

367 MoveRules

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "moveRules",
    "revision": <revisionUid>,
    "ruleset": <targetRulesetUid>,
    "data": {
        "rules": [<ruleUid>]
    }
}
```

- "revision" (mandatory): The revision from which the rules are copied.
- "ruleset" (mandatory): The target ruleset to which the rules shall be copied.
- "data:rules" (mandatory): A list of rule uids that shall be copied.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

368 MultiDelete

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "multiDelete",
    "revision": <revisionUid>,
    "data": {
        "items": [<elementUid>,...]
    }
}
```

- "revision" (optional): Uid of a revision if deleting revision elements.
- "data:items" (mandatory): Uids of a list of elements to be deleted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

369 ParentPathExists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "parentPathExists",
    "data": {
        "path": "<path>"
    }
}
```

- "data:path": The path which is to be checked.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

370 PathExists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "pathExists",
    "data": {
        "path": "<path>"
    }
}
```

- "data:path": The path which is to be checked.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

371 ReAttach

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reAttach",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): Uid of the instance to reattach.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

372 RebuildAllIndexes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "rebuildAllIndexes"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

373 RebuildIndex

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "rebuildIndex",
    "uid": <indexUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the index to be rebuilt.
- "mandator" (mandatory): Uid of the mandator to search the index in.
- "revision" (optional): Uid of the revision to search the index in. If not provided, the mandator's champion revision is used.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

374 RebuildIndexSequence

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "rebuildIndexSequence",
    "uid": <indexUid>,
    "mandator": <mandatorUid>,
    "data": {
        "threadNumberSequenceRebuild": <noOfThreads>
    }
}
```

- "uid" (mandatory): Uid of the index whose sequence will be rebuilt.
- "mandator" (mandatory): Uid of the mandator to search the index in.
- "data:threadNumberSequenceRebuild" (optional): Number of threads to be used to rebuild index sequence. The default is 1.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

375 RecomputeAttributes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "recomputeAttributes",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Revision uid to identify the rule generation of which you want to recompute attributes.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

376 RefreshRandomForestClassificationThreshold

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "refreshRandomForestClassificationThreshold",
    "revision": <revisionUid>,
    "data": {
        "classificationThreshold": <classificationThreshold>
    }
}
```

- "revision" (mandatory): Uid of the revision whose model generation settings shall be changed.
- "data:classificationThreshold" (mandatory): Determines the threshold to classify a record as fraud or genuine.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

377 ReGoliveRevision

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reGoliveRevision",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the retired revision that shall re-golive.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

378 ReleaseFreeMemory

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "releaseFreeMemory"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

379 ReloadComplianceLists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reloadComplianceLists"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

380 ReloadPrivateKeys

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reloadPrivateKeys"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

381 ReloadPrivateKeysPasswordSafe

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reloadPrivateKeysPasswordSafe"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

382 Reserve

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "reserve",
    "uid": <elementUid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the element which is to be reserved.
- "mandator" (mandatory): Uid of the mandator that the element belongs to.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

383 ReserveEventLogMessage

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "reserveEventLogMessage",
    "uid": <eventLogMessageId>
}
```

- "uid": Id of the event log message.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

384 ResetAllUserPreferences

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetAllUserPreferences"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

385 ResetIndex

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetIndex",
    "uid": <indexUid>,
    "mandator": <mandatorUid>
}
```

- "uid": Uid of the index to be reset.
- "mandator": Uid of the mandator that the index belongs to.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

386 ResetOutgoingFli

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetOutgoingFli"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

387 ResetPreference

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetPreference",
    "type": <type>
}
```

- "type" (mandatory): Controls which preferences should be reset.

Returns

A HTTP response with the following content:

```
{
    "preference": {
        ... ...
    },
    "responseStatus":["<status>", "<feedbackText>"],
    "reloadUserProfile":true|false
}
```

388 ResetRuleGeneration

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "resetRuleGeneration",
    "revision": <revisionUid>
}
```

- "revision": Uid of the revision.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

389 ResetSandboxRecords

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetSandboxRecords",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision of which sandbox records table shall be cleared.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

390 ResetUserPreferences

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "resetUserPreferences"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

391 RestartInstance

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "restartInstance",
    "uid": <instanceUid>
}
```

- "uid": Uid of the instance to be restarted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

392 Restore

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "restore",
    "uid": <instanceUid>,
    "data": {
        "donor": <donorUid>
    }
}
```

- "uid": Uid of the instance to be restored.
- "data:donor": Uid of the donor instance.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

393 RetireChampion

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "retireChampion",
    "mandator": <mandatorUid>
}
```

- "mandator" (mandatory): Uid of the mandator whose champion revision shall be retired.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

394 RevertToChallenger

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "revertToChallenger",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision which is to be reverted to challenger.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

395 RevisionElementExists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "revisionElementExists",
    "uid": <elementUid>,
    "type": "<elementType>",
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the checked revision element.
- "type" (mandatory): The type of element that is expected. Available values for "elementType" are:

```
"Attribute"
"Counter"
"Collusion"
"Event"
"DeviceIdentification"
"Formula"
"Index"
"Masterdata"
"Merging"
"Pattern"
"Profile"
"Rule"
"Ruleset"
"FinalRuleset"
```

- "revision" (mandatory): Uid of the revision in which the element should be found.

Returns

A HTTP response with the following content:

```
{
    "status": 1,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

- The "status" only shows in the response if the element exists.

396 RevokeKey

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "revokeKey",
    "uid": <keyUid>
}
```

- "uid" (mandatory): Uid of the key to revoke.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

397 RewindFliBuffer

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "rewindFliBuffer"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

398 RewriteRiskList

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "rewriteRiskList",
    "uid": <risk list uid>,
    "mandator": <mandatorUid>
}
```

- "uid" (mandatory): Uid of the risk list to synchronize.
- "mandator" (mandatory): Uid of the mandator to search the risk list in.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

399 RuleReportQueryExists

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "ruleReportQueryExists",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): The revision in which a rule report query is searched.

Returns

A HTTP response with the following content:

```
{
    "ruleReportQueryExists": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

400 Save

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "save",
    "uid": <elementUid>|-1,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "streamType": "<streamType>",
        "uid": <elementUid>|-1,
        "name" : "<name>",
        "comment" : "<comment>",
        ... ...
    }
}
```

- "uid" (optional): If no uid is provided, a new element is created; if an uid is provided, an existing element is changed.
- "mandator" (mandatory/optional): Uid of the mandator to which the element belongs/shall belong.
- "revision" (mandatory/optional): Uid of the revision to which the element belongs/shall belong.
- "data" (mandatory): Contains the settings for the new element. This dictionary is very different for every element.
- "data:streamType" (mandatory): Determines which type of element is to be created. Available values for "streamType" are:

```
"caseAction"
"caseCloseCode"
"caseGroup"
"caseQueue"
"chart"
```

```
"eventLogMessages"
"externalQuery"
"loadFileJob"
"reportGenerationJob"
"keyPerformanceIndicator"
"mandator"
"message"
"notification"
"passwordSafe"
"realtimeInterceptionCode"
"reminders"
"role"
"statusAlarmIndicator"
"textModule"
"user"
"input"
"output"
"collusion"
"counter"
"deviceIdentification"
"event"
"formula"
"index"
"list"
"mapping"
"masterdata"
"merging"
"pattern"
"precedent"
"profile"
"rule"
"ruleset"
"finalRuleset"
"sandboxEntry"
"complianceList"
"definedRiskList"
"merchantMonitoringRule"
"caseQueueReport"
```

```
        "investigationReport"
        "missedCasesReport"
        "userReport"
```

- "data:name" (mandatory): Sets the name of the new element.
- "data:comment" (optional): Sets a descriptional comment of the new element.
- "data:..." (mandatory): Various additional parameters for all different element types.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

401 SaveCluster

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "saveCluster",
    "uid": <instanceUid>,
    "data": {
        "streamType": "irisInstance",
        "id": <instanceId>,
        "uid": <instanceUid>,
        "enabled": true|false,
        "name": "<instanceName>",
        "comment": "<comment>",
        "directories": {
            "arcPath": "<archivePath>",
            "cfgPath": "<configurationPath>",
            "ddcPath": "<ddcPath>",
            "emlPath": "<emailPath>",
            "fliPath": "<fliBufferPath>",
            "incPath": "<apiIncludesPath>",
            "invPath": "<investigationPath>",
            "keyPath": "<keyPath>",
            "kpiPath": "<kpiPath>",
            "logPath": "<logPath>",
            "rdiPath": "<rdiPath>",
            "repPath": "<messageReportPath>",
            "saiPath": "<saiPath>",
            "usrPath": "<userPath>",
            "pwsPath": "<passwordSafePath>"
        },
        "interfaces": [{
            "type": "MessageCommandInterface",
```

```
"address": "<ipAddress>",
"port": "<port>",
"enabled": true|false,
"acceptAll": true|false,
"acceptFrom": ["<ipAddress",...],
"enabledSsl": true|false,
"certificateFile": "<certificateFilePath>",
"privateKeyFile": "<privateKeyFilePath>",
"diffieHellmanFile": "<diffieHellamFilePath>",
"privateKeyType": "ConsolePassword" | "PasswordFile" | "Unencrypted",
"privateKeyPasswordFile": "<privateKeyPasswordFilePath>",
"rejectTls10": true|false,
"rejectTls11": true|false,
"clientCertificateFile": "<clientCertificateFilePath>",
"clientAuthorizationCertificateFile": "<clientAuthorizationCertificateFilePath>",
"requireValidClientCertificate": true|false,
"requireFittingCommonName": true|false,
"requireValidServerCertificate": true|false,
"revokedClientsCrlFile": "<revokedClientsCrlFilePath>"
}, {
"type": "ApplicationProgrammingInterface",
"address": "<ipAddress>",
"port": "<port>",
"enabled": true|false,
"acceptAll": true|false,
"acceptFrom": ["<ipAddress>"],
"enabledSsl": true|false,
"certificateFile": "<certificateFilePath>",
"privateKeyFile": "<privateKeyFilePath>",
"diffieHellmanFile": "<diffieHellamFilePath>",
"privateKeyType": "ConsolePassword" | "PasswordFile" | "Unencrypted",
"privateKeyPasswordFile": "<privateKeyPasswordFilePath>",
"rejectTls10": true|false,
"rejectTls11": true|false,
"clientCertificateFile": "<clientCertificateFilePath>",
"clientAuthorizationCertificateFile": "<clientAuthorizationCertificateFilePath>",
"requireValidClientCertificate": true|false,
"requireFittingCommonName": true|false,
```

```
"requireValidServerCertificate": true|false,
"revokedClientsCrlFile": "<revokedClientsCrlFilePath>"
}, {
"type": "BatchDataInterface",
"enabled": true|false
}, {
"type": "FastLinkInterface",
"address": "<ipAddress>",
"port": "<port>",
"enabled": true|false
}, {
"type": "StatusControlInterface",
"address": "<ipAddress>",
"port": "<port>"
}, {
"type": "EncryptedCommunicationInterface",
"address": "<ipAddress>",
"port": "<port>",
"enabled": true|false,
"enabledSsl": true|false,
"certificateFile": "<certificateFilePath>",
"privateKeyFile": "<privateKeyFilePath>",
"diffieHellmanFile": "<diffieHellamFilePath>",
"privateKeyType": "ConsolePassword" | "PasswordFile" | "Unencrypted",
"privateKeyPasswordFile": "<privateKeyPasswordFilePath>",
"rejectTls10": true|false,
"rejectTls11": true|false,
"clientCertificateFile": "<clientCertificateFilePath>",
"clientAuthorizationCertificateFile": "<clientAuthorizationCertificateFilePath>",
"privateClientKeyFile": "<privateClientKeyFilePath>",
"privateClientKeyType": "ConsolePassword" | "PasswordFile" | "Unencrypted",
"privateClientKeyPasswordFile": "<privateClientKeyPasswordFilePath>",
"serverAuthorizationCertificateFile": "<serverAuthrizationCertificationFilePath>",
"requireValidClientCertificate": true|false,
"requireFittingCommonName": true|false,
"requireValidServerCertificate": true|false,
"revokedServersCrlFile": "<revokedServersCrlFilePath>",
"revokedClientsCrlFile": "<revokedClientsCrlFilePath>"
```

```
    }, {
    "type": "AlertMessageInterface",
    "enabled": true|false
    }, {
    "type": "WebSphereMQInterface",
    "enabled": true|false,
    "queueManagers": [{
        "uid": <uid>|-1,
        "queueManager": "<queueManagerName>",
        "enabled": true|false,
        "address": "<ipAddress>",
        "port": <port>,
        "useFallback": true|false,
        "fallbackAddress": "<ipAddress>",
        "fallbackPort": <port>,
        "channel": "<channel>",
        "protocol": "TCP",
        "reconnectInterval": <reconnectInterval>,
        "useSsl": true|false,
        "keyRepoStem": "<keyRepositoryPath>",
        "cipherSpec": "<sslCipherSpecification>",
        "responderUrl": "<responderUrl>",
        "queues": [{
            "uid": <uid>|-1,
            "queue": "<queueName>",
            "enabled": true|false,
            "message": <messageTypeUid>,
            "queueParameter": "<parameter>",
            "useRespondQueue": true|false,
            "respondQueue": "<responseQueueName>",
            "mqGetTimeout": "mqgetTimeoutSetting"
            },
        ...
        ]},
    ...
    ]}
],
"monitoringDirectories": {
```

```
            "euslPath": "<europeanSanctionListFilePath>",
            "ofacPath": "<ofacSdnListFilePath>",
            "unPath": "<unitedNationsSactionListPath>",
            "pepPath": "<politicalExposedPersonsListFilePath>",
            "ruslPath": "<russianSactionListFilePath>"
        }
    }
}
```

- "uid" (mandatory): The IrisInstance to change, add or remove.
- "data" (mandatory): The IrisInstance's various settings.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

402 SaveCpp

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "saveCpp",
    "uid": <caseGroupUid>,
    "mandator": <mandatorUid>,
    "data": {
        "streamType: "cpp",
        "caseGroupUid": <caseGroupUid>,
        "mandatorUid": <mandatorUid>,
        "id": <cppId> | -1,
        "name": "<name>",
        "status": "New"| "Investigated"| "Followup"| "Due"| "Closed"| "Reopened",
        "comment": "<comment>",
        "reportingAttributes": [
            {
            "name": "<reportingAttributeName>",
            "type": "Numeric" | "Text" | "Timestamp" | "Boolean" | "Hex" | "IPv4" | "TimeInterval",
            "format": <reportingAttributeFormat>,
            "attributeId": <reportingAttributeUid>,
            "encrypted": true | false,
            "value": <reportingAttributeValue>,
            "decimals": <reportingAttributeDecimals>,
            "length": <reportingAttributeLength>,
            "category": "",
            },
            ...],
        "active": true|false
    }
}
```

- "uid": Uid of the case group the CPP belongs to.
- "mandator": Uid of mandator, for which the case group is defined.
- "data:streamType": The stream type.
- "data:caseGroupUid": Uid of the case group the CPP belongs to.
- "data:mandatorUid": Uid of mandator that the case group is defined for.
- "data:id": Id of an existing CPP, or -1 to create a new CPP.
- "data:name": Name of the CPP.
- "data:status": Status of the CPP.
- "data:comment" (optional): Comment for the CPP.
- "data:reportingAttributes": A list of reporting attributes with values for the CPP.
- "data:active": Defines if the CPP should be active or inactive.

Returns

A HTTP response with the following content:

```
{
    "comparisonResult": "Unchanged" | "ChangedNoImpact" | "ChangedComputationImpact",
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

403 SaveEventLogMessage

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "saveEventLogMessage",
    "uid": <eventLogMessageId>,
    "data": {
        "streamType": "eventLogMessages",
        "uid": <eventLogMessageId>,
        "number": <number>,
        "name": "<name>",
        "system": true|false,
        "audit": true|false,
        "external": true|false,
        "console": true|false,
        "pciDssMandated": true|false,
        "comment": "<comment>",
        "level": "<logLevel>",
        "systemChanged": true|false,
        "auditChanged": true|false,
        "externalChanged": true|false,
        "consoleChanged": true|false,
        "commentChanged": true|false,
        "levelChanged": true|false
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

404 SaveSandboxEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "saveSandboxEntry",
    "uid": <entryUid>,
    "revision": <revisionUid>,
    "data": {
        "attributeUid": <attributeUid>,
        "value": <value>
    }
}
```

- "uid" (mandatory): Uid of the sandbox record.
- "revision" (mandatory): Uid of the revision in which the sandbox record is found.
- "data:attributeUid" (mandatory): Uid of the attribute for which the value is to be set.
- "data:value" (mandatory): The value to be set for the given attribute.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

405 SaveSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "saveSettings",
    "data": {
        "settings": {
            "application": {
                "name": "<applicationName>",
                "serverTimeZone": <serverTimezone>
            },
            "memory": {
                "limit": <mainMemoryUsageLimitInGB>
            },
            "userAccounts": {
                "maxFailedLogin": <maximumFailedLogins>,
                "numberOldPasswordRejected": <numberOfOldPasswordsRejected>,
                "passwordValiditySeconds": <passwordValidityInSeconds>,
                "minPasswordLength": <passwordMinimumLength>,
                "passwordMustContainLowerCase": true|false,
                "passwordMustContainUpperCase": true|false,
                "passwordMustContainDigit": true|false,
                "passwordMustContainSpecialCharacter": true|false,
                "userAccountDeletion": true|false,
                "enableExtendedAuth": true|false,
                "enableSystemUser": true|false,
                "maximumUserInactivityPeriodDays": <maximumUserInactivityPeriodDays>,
                "restrictSystemUserIP": ["<ipAddressThatAllowSystemUsersToAccessAPIFrom>"]
            },
            "ldap": {
                "login": "local" | "ldap" | "both",
                "useActiveDirectory": true|false,
```

```
                "host": "<activeDirectoryServerIpAddress",
                "port": <activeDirectoryServerPort>,
                "baseDn": "<baseDN>",
                "domain": "<domain>",
                "ssl": true|false,
                "timeout": <timeoutSetting>,
                "sso": true|false,
                "ssoAutoRelogin": true|false
            },
            "priorisation": {
                "enabled": true|false,
                "irisProcess": "Idle" | "BelowNormal" | "Normal" | "AboveNormal" | "High" | "Realtime",
                "threads": {
                    "interfaces": {
                        "messageCommand": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "statusControl": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                        "applicationProgramming": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "batchData": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                        "fastLink": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "alertMessage": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical"
                    },
                    "caseManagement": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "deferredWriter": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "analysis": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "elementGeneration": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "statusAlarmIndicators": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "keyPerformanceIndicators": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "endOfDay": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "golive": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "simulation": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "query": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "reminder": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" | "TimeCritical",
                    "instanceStatusFileService": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "definedRiskListEntry": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical"
                }
            },
            "deferredWriting": {
```

```
        "enabled": true|false,
        "resetChunkingForShutdownEnabled": true|false,
        "timeGap": <timeGap>,
        "minimumMdcSize": <minimumMdcSize>,
        "safetyMargin": <safetyMargin>,
        "pauseAfterElement": <pauseAfterElement>,
        "diskChunkDelay": <diskChuckDelay>,
        "diskChunkSize": <diskChuckSize>,
        "shadowCommitChunkSize": <shadowCommitChunkSize>,
        "shadowCommitChunkDelay": <shadowCommitChunkDelay>
    },
    "numberOfThreads": {
        "messageCommandInterface": <numberOfThreadsForMessageCommandInterface>,
        "batchDataInterface": <numberOfThreadsForBatchDataInterface>,
        "fastLinkInterface": <numberOfThreadsForFastLinkInterface>,
        "analysis": <numberOfThreadsForAnalyses>,
        "elementGeneration": <numberOfThreadsForModelElementGeneration>,
        "simulation": <mumberOfThreadsForSimulation>,
        "query": <numberOfThreadsForQuery>,
        "file": <numberOfThreadsForFileNotifications>,
        "message": <numberOfThreadsForMessageNotifications>,
        "http": <numberOfThreadsForHTTPNotifications>,
        "odbcSql": <numberOfThreadsForSqlNotifications>,
        "smtp": <numberOfThreadsForSmtpNotifications>,
        "templateFile": <numberOfThreadsForDocxNotifications>
    },
    "pageAutoRefresh": {
        "dashboard": <pageAutoRefreshInSecondsForDashboard>,
        "investigation": <pageAutoRefreshInSecondsForInvestigation>,
        "analysis": <pageAutoRefreshInSecondsForAnalyses>,
        "jobSchedule": <pageAutoRefreshInSecondsForJobSchedule>,
        "elementGeneration": <pageAutoRefreshInSecondsForModelElementGeneration>,
        "golive": <pageAutoRefreshInSecondsForModelGolive>,
        "cluster": <pageAutoRefreshInSecondsForClusterAdministration>,
        "keys": <pageAutoRefreshInSecondsForEncryptionKeys>,
        "query": <pageAutoRefreshInSecondsForQueryResult>,
        "memoryManagement": <pageAutoRefreshInSecondsForMemoryManagement>,
        "sessionCountdown": <pageAutoRefreshInSecondsForSessionCountdown>,
```

```
        "log": <pageAutoRefreshInSecondsLogTables>
    },
    "investigation": {
        "caseGenerationEnabled": true|false,
        "manualFraudValue": <manualFraudValue>,
        "updateProfilesAndEventsByManualFraudMark": true|false,
        "maxCasesSelectionTable": <maximumCasesShownOnSelectionTable>,
        "maxRecCaseHistory": <maximumCasesShownInHistory>,
        "caseConsolidationRepeat": <caseConsolicationStartsEveryNumberOfSeconds>,
        "archiveCasesAfterSeconds": <archiveCasesAfterNumberOfSeconds>,
        "enableCaseAttachments": true|false,
        "maxCaseAttachmentSizeMB": <maximumCaseAttachmentSizeInMB>,
        "caseCreationDdc": true|false
    },
    "monitoring": {
        "monitoringResultLifetime": <monitoringResultLifetimeInSeconds>,
        "monitoringMaxTableEntries": <maximumNumberOfDisplayedEntries>,
        "monitoringComplianceEnabled": true|false,
        "monitoringComplianceEuslEnabled": true|false,
        "monitoringComplianceOfacEnabled": true|false,
        "monitoringComplianceUNEnabled": true|false,
        "monitoringComplianceGlobalWatchListEnabled": true|false,
        "monitoringCompliancePepEnabled": true|false,
        "monitoringComplianceRuslEnabled": true|false,
        "monitoringComplianceWaitFactor": <reloadWaitFactor>,
        "definedRiskListDeleteEnabled": true|false,
        "definedRiskListMassDeleteEnabled": true|false,
        "definedRiskListMassDisableEnabled": true|false,
        "definedRiskListEntryConsolidationRepeat": <consolidationStartsEveryNumberOfSeconds>
    },
    "model": {
        "enforcePeerConfirmation": true|false,
        "rebuildIndexOnSizeChange": true|false,
        "enableDdcInModel": true|false,
        "enableDdcInMerchantMonitoringRules": true|false,
        "enableDdcInMergings": true|false,
        "numberOfCategoriesInAuditTrail": <maximumNumberOfCategoryChanges>,
        "numberOfStoredChampionAuditTrailEntries": <storedChampionAuditTrailEntries>,
```

```
        "numberOfDisplayedChampionAuditTrailEntries": <displayedChampionAuditTrailEntries>,
        "numberOfDisplayedCollusionSimulationResults": <displayedCollusionSimulationResults>,
        "goliveRelativeWaitFactor": <remoteGoliveWaitFactor>,
        "treatGoliveWarningsAsErrors": true|false,
        "defaultModelling": {
            "simulationMayUseDdc": true|false,
            "categoryLimitModelling": <maximumComputedIndicators>
        },
        "defaultCapacities": {
            "mdc": <mdcCapacity>,
            "ddc": <ddcCapacity>,
            "ddcStored": true|false,
            "mdcStored": true|false
        },
        "deletion": {
            "challenger": true|false,
            "invalidated": true|false,
            "retired": true|false
        }
    },
    "interfaces": {
        "serializeComputation": {
            "enabled": true|false,
            "mergingProtectionEnabled: true|false,
            "maxMergingProtectionTries": <maxMergingProtectionTries>,
            "mergingProtectionWaitMsec": <mergingProtectionWaitMsec>,
            "mergingProtectionFailOnTimeout": true|false
        },
        "statusControl": {
            "receiveTimeoutSeconds": <receiveTimeoutInSeconds>
        },
        "messageCommand": {
            "mciFormat": "ISO20022" | "ISO8583",
            "mciLatencyViolationThreshold": <latencyViolationThresholdInMilliseconds>,
            "mciLatencyViolationLoggingEnabled": true|false,
            "useOnlyPrintableAscii": true|false,
            "closeDuringGolive": true|false,
            "responseMetaInformation": {
```

```
            "respondInstanceName": true|false,
            "respondMtid": true|false,
            "respondSystemTime": true|false,
            "respondUrid": true|false,
            "respondMessageId": true|false,
            "respondModels": true|false,
            "respondMergingSource": true|false,
            "respondStatus": true|false,
            "respondLatency": true|false,
            "respondErrorText": true|false,
            "maxDocumentedRulesInResponse": <respondMaxFiredRules>
        },
        "messageTracing": {
            "showIpData": true|false,
            "dumpMalformattedMessages": <dumpMalformattedMessages>
        },
        "threadBufferSize": {
            "incomingBuffer": <incomingBufferInBytes>,
            "responseBuffer": <responseBufferInBytes>
        }
    },
    "applicationProgramming": {
        "lockTimeoutSeconds": <sessionTimeoutInSeconds>,
        "lockTimeoutCountdownThresholdSeconds": <sessionTimeoutCountdownThresholdInSeconds>,
        "enableCsrf": true|false,
        "useDefaultHttpHeaders": true|false,
        "useHstsHttpHeaders": true|false,
        "customHttpHeaders": "<customHttpHeaders>",
        "useGzip": true|false,
        "useKeepAlive": true|false,
        "downloadRequests": true|false,
        "serverNameInHttpResponses": "IRIS",
        "maxPostRequestSizeMB": <maximumPostRequestSizeInMB>,
        "defaultLanguage": "en",
        "defaultDateTimeFormat": "US" | "EUR" | "ISO",
        "defaultDecimal": "." | ",",
        "defaultDigitGroup": "," | "'" | ".",
        "defaultFieldSeparator": "tab" | "," | ";",
```

```
                    "defaultExtendedSelectDialog": true|false,
                    "defaultSearchInSelections": true|false,
                    "defaultShowExplanation": true|false,
                    "defaultShowUids": true|false,
                    "colorSchemaFraudulentValues": "#<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>",
                    "colorSchemaGenuineValues": "#<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>",
                    "colorSchemaOtherValues": "#<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>, #<colorHex>"
                },
                "batchData": {
                    "jobRepeat": <jobSchedulerStartsEveryNumberOfSeconds>,
                    "prioMaxJobThread": "Idle" | "Lowest" | "BelowNormal" | "Normal" | "AboveNormal" | "Highest" |
"TimeCritical", "jobEncryptionEnabled": true|false
                },
                "fastLink": {
                    "receiveTimeoutSeconds": <receiveTimeoutInSeconds>,
                    "retryConnectionSeconds": <retryConnectionInSeconds>,
                    "numMessagesToleratedInSync": <synchronisationThreshold>,
                    "outgoingBufferSize": <bufferSizeInGB>,
                    "jobProcessingBrakeAt": <jobProcessingFreezeAtNumberOfPercentageOfBufferSize, valueShouldBeInDecimal>,
                    "transactionMessageProcessingHaltAt": <closeMCIAtNumberOfPercentageOfBufferSize, valueShouldBeInDecimal>
                },
                "alertMessage": {
                    "ipAddress": "<ipAddress>",
                    "ipPort": "<port>",
                    "receiveTimeoutSeconds": "<receiveTimeoutInSeconds>",
                    "authentication": "AuthLogin" | "NoAuth",
                    "username": "<userName>",
                    "password": "<password>",
                    "from": "<fromEmailAddress>",
                    "sendInterval": <sendIntervalInSeconds>,
                    "retryInterval": <retryIntervalInSeconds>,
                    "archive": true|false,
                    "useSsl": true|false,
                    "verificationCertificate": "<serverCACertificateFile>"
                }
            },
            "query": {
                "maxRecordsWarning": <maximumNumberOfRecordsWarningAt>,
```

```
        "maxRecordsLimit": <maximumNumberOfRecordsLimitAt>,
        "groupingQueryAccountsLimit": <groupByQueryAccountsLimit>,
        "groupingQueryResultCategoriesLimit": <groupByQueryCategoriesLimit>,
        "queryEnableDdc": "Available" | "Never" | "Default",
        "queryResultLivetime": <queryResultLifetimeInSeconds>,
        "commonPointQueryResultLivetimeSeconds": <commonPointResultLifetimeInSeconds>,
        "maxQueryResults": <maximumQueryResults>,
        "apiQueryAccess": "Disabled" | "EnabledAllUsers" | "EnabledSystemUsers"
    },
    "encryption": {
        "enabled": true|false,
        "keyReuseEnabled": true|false,
        "keyRetryTimeoutSeconds": <retryKeyretrivalEveryNumberOfSeconds>,
        "maximumKeyLifeBeforeShutdownDays": <maximumKeyLifeInDays>,
        "maximumMasterKeyLifeBeforeShutdownDays": <maximumMasterKeyLifeInDays>,
        "wipeDeletedConfigurationAndUserFiles": true|false,
        "changeMasterKeyRelativeWaitFactor": <waitFactor>,
        "encryptSensitiveExports": true|false
    },
    "eventLogMessages": {
        "defaultEventLogMessageViewInterval": <defaultViewPeriodInSeconds>,
        "maxRecEventLogMessage": <maximumRecordsShown>,
        "enableAllLogMessages": true|false,
        "eventLogMessagesEnabled": true|false,
        "syslogTemplate": "{instanceId}  {dateIso}.{nanoSeconds}  {level}  {number}  {user}  {message}  {ipData}"
    },
    "ticker": {
        "enabled": true|false,
        "maxTickerEntries": <maximumNumberOfSavedTickerEntries>
    },
    "other": {
        "eodDayTimeString": "<startEndOfDayJobAt[hh:mm]",
        "shutdownGracePeriod": <shutdownGracePeriodInSeconds>,
        "updateInstanceStatusFileSec": <updateIntervalInSecondsForInstanceStatusFile>,
        "maxReportLimit": <reportLimitAt>,
        "checkIndexesOnStartup": true|false,
        "fileCreateZeros": "AlwaysFill" | "NoFill" | "FillInitialOnly",
        "wipeDdcPosition": true|false,
```

```
                "computationThreadBufferElementSize": <computationThreadBufferElementSize>,
                "multipleRelationsIndexSearch": true|false,
                "sslCipherList": "<sslCipherList>",
                "sslUseCertificateChain": true|false
            }
        }
    }
}
```

- "data:settings" (mandatory): Contains the settings dictionary.
- "data:settings:applications" (optional): Contains general application settings.
- "data:settings:applications:name" (optional): Sets the application name which is e.g. displayed as browser tab name.
- "data:settings:applications:serverTimeZone" (optional): Sets the server time zone in UTC +/- seconds.
- "data:settings:memory" (optional): Contains memory settings.
- "data:settings:memory:limit" (optional): Sets a maximum memory border (in GB) a SP instance may use.
- "data:settings:userAccounts" (optional): Contains settings for user accounts, including password policies.
- "data:settings:userAccounts:maxFailedLogin" (optional): A user account is blocked after these many failed login attempts.
- "data:settings:userAccounts:numberOldPasswordRejected" (optional): Length of password history which is rejected as new password.
- "data:settings:userAccounts:passwordValiditySeconds" (optional): Time period in seconds that a password is valid until it must be changed.
- "data:settings:userAccounts:minPasswordLength" (optional): Minimum password length.
- "data:settings:userAccounts:passwordMustContainLowerCase" (optional): Enforce using lower case characters in passwords.
- "data:settings:userAccounts:passwordMustContainUpperCase" (optional): Enforce using upper case characters in passwords.
- "data:settings:userAccounts:passwordMustContainDigit" (optional): Enforce using digits in passwords.
- "data:settings:userAccounts:passwordMustContainSpecialCharacter" (optional): Enforce using special characters in passwords.
- "data:settings:userAccounts:userAccountDeletion" (optional): Enable functionality to permanently delete user accounts.

- "data:settings:userAccounts:enableExtendedAuth" (optional): Enable two factor authentification.
- "data:settings:userAccounts:enableSystemUser" (optional): Enables definition of system accounts that may be used by API scripts.
- "data:settings:userAccounts:maximumUserInactivityPeriodDays" (optional): Users are disabled automatically when not logged in for this amount of dates.
- "data:settings:userAccounts:restrictSystemUserIP" (optional): If system users are enabled, this is a list of IPs from which they may connect.
- "data:settings:ldap" (optional): Allows to configure ldap.
- "data:settings:ldap:login" (optional): Choose whether to use normal SP login, ldap or both.
- "data:settings:ldap:useActiveDirectory" (optional): If enabled, active directory authentication will be used.
- "data:settings:ldap:host" (optional): IP address or domain name of LDAP server.
- "data:settings:ldap:port" (optional): IP port of LDAP server.
- "data:settings:ldap:baseDn" (optional): Base of distinguished name.
- "data:settings:ldap:domain" (optional): Active Directory domain for login.
- "data:settings:ldap:ssl" (optional): Enables encrypted communication between Safer Payments and LDAP host.
- "data:settings:ldap:timeout" (optional): Time to wait to connect to LDAP server before timeout.
- "data:settings:ldap:sso" (optional): If enabled, users may automatically login using their windows credentials.
- "data:settings:ldap:ssoAutoRelogin" (optional): Allow a user to automatically sign in after their session has expired.
- "data:settings:priorisation" (optional): Contains settings for thread priorisations to set priorities for CPU resources.
- "data:settings:priorisation:enabled" (optional): Enables priorisations between threads.
- "data:settings:priorisation:irisProcess" (optional): Sets priority of the SP process in general.
- "data:settings:priorisation:threads:interfaces" (optional): Contains priority settings for the several interfaces.
- "data:settings:priorisation:threads:interfaces:messageCommand" (optional): Sets priority of MCI.
- "data:settings:priorisation:threads:interfaces:statusControl" (optional): Sets priority of SCI.

- "data:settings:priorisation:threads:interfaces:applicationProgramming" (optional): Sets priority of API.
- "data:settings:priorisation:threads:interfaces:batchData" (optional): Sets priority of BDI.
- "data:settings:priorisation:threads:interfaces:fastLink" (optional): Sets priority of FLI.
- "data:settings:priorisation:threads:interfaces:alertMessage" (optional): Sets priority of AMI.
- "data:settings:priorisation:threads:caseManagement" (optional): Sets priority of activities in case management.
- "data:settings:priorisation:threads:deferredWriter" (optional): Sets priority of deferred writing thread.
- "data:settings:priorisation:threads:analysis" (optional): Sets priority of analysis computations.
- "data:settings:priorisation:threads:elementGeneration" (optional): Sets priority of element creations.
- "data:settings:priorisation:threads:statusAlarmIndicators" (optional): Sets priority of status alarm indicators.
- "data:settings:priorisation:threads:keyPerformanceIndicators" (optional): Sets priority of key performance indicators.
- "data:settings:priorisation:threads:endOfDay" (optional): Sets priority of end of day job.
- "data:settings:priorisation:threads:golive" (optional): Sets priority of golives.
- "data:settings:priorisation:threads:simulation" (optional): Sets priority of simulation computation.
- "data:settings:priorisation:threads:query" (optional): Sets priority of query execution.
- "data:settings:priorisation:threads:reminder" (optional): Sets priority of reminder computation.
- "data:settings:priorisation:threads:instanceStatusFileService" (optional): Sets priority of instance status file write thread.
- "data:settings:priorisation:threads:definedRiskListEntry" (optional): Sets priority of defined risk list entry generation.
- "data:settings:deferredWriting" (optional): Contains settings for deferred writing.
- "data:settings:deferredWriting:enabled" (optional): Enables deferred writing.
- "data:settings:deferredWriting:resetChunkingForShutdownEnabled" (optional): If enabled, resets all deferred writing chunking parameters during shutdown to achieve maximum writing speed.
- "data:settings:deferredWriting:timeGap" (optional): Time in seconds that MDC will never be written to DDC.

- "data:settings:deferredWriting:minimumMdcSize" (optional): Defined minimum MDC size that Safer Payments uses for sanity checks during golives.
- "data:settings:deferredWriting:safetyMargin" (optional): Number of records that within the MDC are not used for caching but rather to account for the new transactions to be stored in MDC during commitment of MDC data to DDC.
- "data:settings:deferredWriting:pauseAfterElement" (optional): Time in milliseconds to wait after each element is written to disk.
- "data:settings:deferredWriting:diskChunkDelay" (optional): Time in milliseconds to wait before writing the next chunk.
- "data:settings:deferredWriting:diskChunkSize" (optional): Bytesize of data chunk which is written to disk before Safer Payments waits the defined time.
- "data:settings:deferredWriting:shadowCommitChunkSize" (optional): If greater than 0, transaction computation will get smoother as between chunks, computation continues.
- "data:settings:deferredWriting:shadowCommitChunkDelay" (optional): Time in milliseconds to wait in between chunks for transaction computation.
- "data:settings:numberOfThreds" (optional): Defines how many threads different SP computations may acquire.
- "data:settings:numberOfThreads:messageCommandInterface" (optional): Number of threads for MCI.
- "data:settings:numberOfThreads:batchDataInterface" (optional): Number of threads for BDI.
- "data:settings:numberOfThreads:fastLinkInterface" (optional): Number of threads for FLI.
- "data:settings:numberOfThreads:analysis" (optional): Number of threads analysis services computations spin off in each computational step.
- "data:settings:numberOfThreads:elementGeneration" (optional): Number of threads model element generation spins off in each computational step.
- "data:settings:numberOfThreads:simulation" (optional): Number of threads simulation spins off in each simulation run.
- "data:settings:numberOfThreads:query" (optional): Number of threads queries spin off.
- "data:settings:numberOfThreads:file" (optional): Number of file notification threads defines maximum number of file notifications that may be processed in parallel.
- "data:settings:numberOfThreads:message" (optional): Number of message notification threads defines maximum number of message notifications that may be processed in parallel.
- "data:settings:numberOfThreads:http" (optional): Number of http notification threads defines maximum number of http notifications that may be processed in parallel.

- "data:settings:numberOfThreads:odbcSql" (optional): Number of odbcSql notification threads defines maximum number of odbcSql notifications that may be processed in parallel.
- "data:settings:numberOfThreads:smtp" (optional): Number of smtp notification threads defines maximum number of smtp notifications that may be processed in parallel.
- "data:settings:numberOfThreads:templateFile" (optional): Number of templateFile notification threads defines maximum number of docx notifications that may be processed in parallel.
- "data:settings:pageAutoRefresh" (optional): Allows to define the intervals (in seconds) in which different pages are automatically refreshed.
- "data:settings:pageAutoRefresh:dashboard" (optional): Automatically refresh the dashboard page after x seconds.
- "data:settings:pageAutoRefresh:investigation" (optional): Automatically refresh cases table after x seconds.
- "data:settings:pageAutoRefresh:workCase" (optional): Automatically refresh work case page after x seconds.
- "data:settings:pageAutoRefresh:analysis" (optional): Automatically refresh analysis results during computation after x seconds.
- "data:settings:pageAutoRefresh:jobSchedule" (optional): Automatically refresh job schedule page after x seconds.
- "data:settings:pageAutoRefresh:elementGeneration" (optional): Automatically refresh a model element generation page after x seconds.
- "data:settings:pageAutoRefresh:golive" (optional): Automatically refresh model revision selection page during a golive after x seconds.
- "data:settings:pageAutoRefresh:cluster" (optional): Automatically refresh cluster page after x seconds.
- "data:settings:pageAutoRefresh:keys" (optional): Automatically refresh encryption keys page after x seconds.
- "data:settings:pageAutoRefresh:query" (optional): Automatically refresh query results during computation after x seconds.
- "data:settings:pageAutoRefresh:memoryManagement" (optional): Automatically refresh memory management page after x seconds.
- "data:settings:pageAutoRefresh:sessionCountdown" (optional): Auto-refresh period in seconds for synchronization of session countdowns in different tabs.
- "data:settings:pageAutoRefresh:log" (optional, new user interface only): Defines the interval in seconds after which system and audit log tables are automatically refreshed, if auto refresh is enabled in log page.
- "data:settings:investigation" (optional): Configures handling of investigation cases.
- "data:settings:investigation:caseGenerationEnabled" (optional): Enables to work with cases in Safer Payments.

- "data:settings:investigation:manualFraudValue" (optional): If no categories are defined for meta attribute "fraud", this value is assigned to records that are manually marked fraudulent.
- "data:settings:investigation:updateProfilesAndEventsByManualFraudMark" (optional): If enabled, a manual fraud marking will cause an update of profile and event values.
- "data:settings:investigation:maxCasesSelectionTable" (optional): Case selection table will not show more entries than defined here.
- "data:settings:investigation:maxRecCaseHistory" (optional): Case history table will not show more entries than defined here.
- "data:settings:investigation:caseConsolidationRepeat" (optional): Time period in seconds in which case consolidation is periodically started.
- "data:settings:investigation:caseEscalationRepeat" (optional): Time period in seconds in which case escalation is periodically started.
- "data:settings:investigation:caseDispatchingRepeat" (optional): Time period in seconds in which case dispatching is periodically started.
- "data:settings:investigation:archiveCasesAfterSeconds" (optional): Time period in days after which cases are automatically archived.
- "data:settings:investigation:enableCaseAttachments" (optional): If enabled, files may be uploaded to a case.
- "data:settings:investigation:maxCaseAttachmentSizeMB" (optional): Defines a maximum size allowed for case attachments.
- "data:settings:investigation:caseCreationDdc" (optional): If enabled, cases manually created from a query result may include data from DDC.
- "data:settings:monitoring" (optional): Configures usage of defined risk lists and compliance lists.
- "data:settings:monitoring:monitoringResultLifetime" (optional): Time in seconds for how long a monitoring search result is stored.
- "data:settings:monitoring:monitoringMaxTableEntries" (optional): Maximum number of defined risk list entries that are shown in a table.
- "data:settings:monitoring:monitoringComplianceEnabled" (optional): Enables monitoring of compliance lists.
- "data:settings:monitoring:monitoringComplianceEuslEnabled" (optional): Enables usage of European sanction list.
- "data:settings:monitoring:monitoringComplianceOfacEnabled" (optional): Enables usage of OFAC list.
- "data:settings:monitoring:monitoringComplianceUNEnabled" (optional): Enables usage of United Nations list.
- "data:settings:monitoring:monitoringComplianceGlobalWatchListEnabled" (optional): Enables usage of Global watch list.
- "data:settings:monitoring:monitoringCompliancePepEnabled" (optional): Enables usage of politically exposed persons list (PEP).

- "data:settings:monitoring:monitoringComplianceRuslEnabled" (optional): Enables usage of Russian sanction list (PEP).
- "data:settings:monitoring:monitoringComplianceWaitFactor" (optional): The relative wait factor for a compliance list reload request to be transmitted to remote instances.
- "data:settings:monitoring:definedRiskListDeleteEnabled" (optional): Allows to permanently delete defined risk lists.
- "data:settings:monitoring:definedRiskListMassDeleteEnabled" (optional): Allows bulk deletion of defined risk list entries.
- "data:settings:monitoring:definedRiskListMassDisableEnabled" (optional): Allows bulk enabling/disabling of defined risk list entries.
- "data:settings:monitoring:definedRiskListEntryConsolidationRepeat" (optional): Time period in seconds in which defined risk list entry consolidation is periodically started.
- "data:settings:model" (optional): Configures settings related to decision models. "data:settings:model:enforcePeerConfirmation" (optional): If enabled, the user that has last edited a model revision may not be the one who confirms a golive.
- "data:settings:model:rebuildIndexOnSizeChange" (optional): If active, indexes will be recomputed from scratch during golive, if their sequence settings have changed and they now can access more records.
- "data:settings:model:enableDdcInModel" (optional): If enabled, counters and index sequencings may be defined to also use data from DDC.
- "data:settings:model:enableDdcInMerchantMonitoringRules" (optional): If enabled, merchant monitoring rules may be defined to also use data from DDC.
- "data:settings:model:enableDdcInMergings" (optional): If enabled, mergings may be defined to also evaluate data from DDC.
- "data:settings:model:numberOfCategoriesInAuditTrail" (optional): The maximum number of changed categories which will be stored in detail in the revision audit trail or shown in the revision compare function.
- "data:settings:model:numberOfStoredChampionAuditTrailEntries" (optional): The maximum number of audit trail entries which will be stored in the champion audit trail.
- "data:settings:model:numberOfDisplayedChampionAuditTrailEntries" (optional): The maximum number of displayed audit trail entries out of the champion audit trail.
- "data:settings:model:numberOfDisplayedCollusionSimulationResults" (optional): The maximum number of collusion simulation results that will be displayed.
- "data:settings:model:goliveRelativeWaitFactor" (optional): The relative wait factor for a golive to be transmitted to remote instances; "0" for immediate transmission after successful golive; "1" for a cascaded golive.

- "data:settings:model:treatGoliveWarningsAsErrors" (optional): If enabled, all warnings in golive report will be treated as errors and thus prevent the golive.
- "data:settings:model:defaultModelling" (optional): Allows to configure settings for modelling.
- "data:settings:model:defaultModelling:simulationMayUseDdc" (optional): If enabled, simulation may access data from DDC.
- "data:settings:model:defaultModelling:categoryLimitModelling" (optional): Defines how many indicators shall be considered in computation.
- "data:settings:model:defaultCapacities" (optional): Defines the default mdc/ddc settings that should be pre-chosen when creating a new element.
- "data:settings:model:defaultCapacities:mdc" (optional): Default MDC capacity.
- "data:settings:model:defaultCapacities:ddc" (optional): Default DDC capacity.
- "data:settings:model:defaultCapacities:ddcStored" (optional): If enabled, new elements will have MDC enabled as default.
- "data:settings:model:defaultCapacities:mdcStored" (optional): If enabled, new elements will have DDC enabled as default.
- "data:settings:model:deletion" (optional): Defines which types of revisions may be deleted.
- "data:settings:model:deletion:challenger" (optional): If enabled, challenger revisions may be deleted.
- "data:settings:model:deletion:invalidated" (optional): If enabled, invalidated revisions may be deleted.
- "data:settings:model:deletion:retired" (optional): If enabled, retired revisions may be deleted.
- "data:settings:interfaces" (optional): Contains settings for the various interfaces of Safer Payments.
- "data:settings:interfaces:serializeComputation" (optional): Contains settings for serializing computations.
- "data:settings:interfaces:serializeComputation:enabled" (optional): If enabled, computation of critical areas will be slower; if not enabled, doublet detection and mergings will not work correctly for parallely incoming transactions.
- "data:settings:interfaces:serializeComputation:mergingProtectionEnabled" (optional): If enabled, records that still are in computation may not be overwritten by merging sources, retrying the merging source until the computation is complete.
- "data:settings:interfaces:serializeComputation:maxMergingProtectionTries" (optional): Defines the maximum amount of retries for mergings with access protection.
- "data:settings:interfaces:serializeComputation:mergingProtectionWaitMsec" (optional): Defines the time in miliseconds in between access protection retries.

- "data:settings:interfaces:serializeComputation:mergingProtectionFailOnTimeout" (optional): Define whether to abort a transaction or continuing when maximum retries are exceeded.
- "data:settings:interfaces:statusControl" (optional): Configures settings of Status Control Interface (SCI).
- "data:settings:interfaces:statusControl:receiveTimeoutSeconds" (optional): When this timeout expires, the receiving socket is closed and the status message is dumped.
- "data:settings:interfaces:messageCommand" (optional):
- "data:settings:interfaces:messageCommand:mciFormat" (optional): Select which ISO format this interface shall support.
- "data:settings:interfaces:messageCommand:mciLatencyViolationThreshold" (optional): Latency threshold which is used to define what is considered as a latency violation.
- "data:settings:interfaces:messageCommand:mciLatencyViolationLoggingEnabled" (optional): If enabled, detailed processing times of transactions are captured when latency violations occur.
- "data:settings:interfaces:messageCommand:useOnlyPrintableAscii" (optional): If enabled, MCI response will only contain printable ASCII characters, others are escaped with _.
- "data:settings:interfaces:messageCommand:closeDuringGolive" (optional): If enabled, all MCI connections will close during golive.
- "data:settings:interfaces:messageCommand:responseMetaInformation" (optional): If enabled, users can define a custom response message for MCI messages. Only available in association with a custom parser.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondInstanceName" (optional): If enabled, a MCI response will contain the instance name of the processing instance.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondMtid" (optional): If enabled, a MCI response will contain the MessageTypeID that has processed the message.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondSystemTime" (optional): If enabled, a MCI response will contain the servers system time.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondUrid" (optional): If enabled, a MCI response will contain the URID that was assigned to an incoming message.

- "data:settings:interfaces:messageCommand:responseMetaInformation:respondMessageId" (optional): If enabled, a MCI response will echo the MessageId value that was sent with the incoming message.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondModels" (optional): If enabled, a MCI response will contain the model revisions that were used to compute the incoming message.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondMergingSource" (optional): If enabled, a MCI response will respond whether the incoming message was used as a merging source.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondStatus" (optional): If enabled, a MCI response will include Safer Payments server status.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondLatency" (optional): If enabled, a MCI response will include processing latency of the incoming transaction.
- "data:settings:interfaces:messageCommand:responseMetaInformation:respondErrorText" (optional): If enabled, a MCI response will contain textual descriptions next to error codes.
- "data:settings:interfaces:messageCommand:responseMetaInformation:maxDocumentedRulesInResponse" (optional): If set to a non-zero value, a MCI response will contain rules that had been fired during processing of the incoming transaction.
- "data:settings:interfaces:messageCommand:messageTracing" (optional): These settings allow for the tracing of XML message requests within Safer Payments to test the MCI.
- "data:settings:interfaces:messageCommand:messageTracing:showIpData" (optional): Causes Safer Payments to echo all data exchanged between Safer Payments and connected systems.
- "data:settings:interfaces:messageCommand:messageTracing:dumpMalformattedMessages" (optional): If value is not zero, Safer Payments dumps this maximum number of malformatted messages it receives in the "log" subdirectory as individual files.
- "data:settings:interfaces:messageCommand:threadBufferSize" (optional): Configures sizes of MCI buffers.
- "data:settings:interfaces:messageCommand:threadBufferSize:incomingBuffer" (optional): Bytesize of MCI incoming buffer.
- "data:settings:interfaces:messageCommand:threadBufferSize:responseBuffer" (optional): Bytesize of MCI outgoing buffer.
- "data:settings:interfaces:applicationProgramming" (optional):

- "data:settings:interfaces:applicationProgramming:lockTimeoutSeconds" (optional): User will automatically log out when there is no action within the defined amount of seconds.
- "data:settings:interfaces:applicationProgramming:lockTimeoutCountdownThresholdSeconds" (optional): Timeout countdown in UI will be highlighted starting from this threshold.
- "data:settings:interfaces:applicationProgramming:enableCsrf" (optional): Use a cookie-stored session variable to avoid session hijacking via CSRF.
- "data:settings:interfaces:applicationProgramming:useDefaultHttpHeaders" (optional): Use the default and secure HTTP headers for API requests.
- "data:settings:interfaces:applicationProgramming:useHstsHttpHeaders" (optional): If enabled, use standard secure set of HTTP headers.
- "data:settings:interfaces:applicationProgramming:customHttpHeaders" (optional): If useDefaultHttpHeaders is disabled, you can provide your custom headers here.
- "data:settings:interfaces:applicationProgramming:useGzip" (optional): Use HTTP gzip compression for API requests.
- "data:settings:interfaces:applicationProgramming:useKeepAlive" (optional): Use HTTP 1.1 keep-alive for API requests.
- "data:settings:interfaces:applicationProgramming:downloadRequests" (optional): Allows the user to download all his or her previous requests.
- "data:settings:interfaces:applicationProgramming:serverNameInHttpResponses" (optional): Custom server name in header of HTTP responses.
- "data:settings:interfaces:applicationProgramming:maxPostRequestSizeMB" (optional): Maximum size of a HTTP POST request.
- "data:settings:interfaces:applicationProgramming:defaultLanguage" (optional): The default language with which new users are created.
- "data:settings:interfaces:applicationProgramming:defaultDateTimeFormat" (optional): The default date time format with which new users are created.
- "data:settings:interfaces:applicationProgramming:defaultDecimal" (optional): The default decimal separator with which new users are created.
- "data:settings:interfaces:applicationProgramming:defaultDigitGroup" (optional): The default digit group separator with which new users are created.
- "data:settings:interfaces:applicationProgramming:defaultFieldSeparator" (optional): The default field separator with which new users are created.
- "data:settings:interfaces:applicationProgramming:defaultExtendedSelectDialog" (optional): The default setting whether to enable extended select dialogs for newly created users.
- "data:settings:interfaces:applicationProgramming:defaultSearchInSelections" (optional): The default setting whether to enable search functionality for newly created users.

- "data:settings:interfaces:applicationProgramming:defaultShowExplanation" (optional): The default setting whether to show explanations for newly created users.
- "data:settings:interfaces:applicationProgramming:defaultShowUids" (optional): The default setting whether to show UIDs for newly created users.
- "data:settings:interfaces:applicationProgramming:colorSchemaFraudulentValues" (optional): These colors are used to display fraudulent values in charts.
- "data:settings:interfaces:applicationProgramming:colorSchemaGenuineValues" (optional): These colors are used to display genuine values in charts.
- "data:settings:interfaces:applicationProgramming:colorSchemaOtherValues" (optional): These colors are used to display other fraud values in charts.
- "data:settings:interfaces:batchData" (optional):
- "data:settings:interfaces:batchData:jobRepeat" (optional): Time period in seconds in which the job scheduler is periodically started.
- "data:settings:interfaces:batchData:prioMaxJobThread" (optional): Maximum possible priority of a job.
- "data:settings:interfaces:batchData:jobEncryptionEnabled" (optional): If enabled, encrypted job files can be imported.
- "data:settings:interfaces:fastLink" (optional):
- "data:settings:interfaces:fastLink:receiveTimeoutSeconds" (optional): Safer Payments closes its receiving sockets and restarts its incoming interface if the timeout expires.
- "data:settings:interfaces:fastLink:retryConnectionSeconds" (optional): Time in seconds after a sending IBM Safer Payments instance retries to connect to another instance that was considered unresponsive before.
- "data:settings:interfaces:fastLink:numMessagesToleratedInSync" (optional): Number of queued FastLink messages to still be considered synchronized.
- "data:settings:interfaces:fastLink:outgoingBufferSize" (optional): Size in GB that Safer Payments allocates for the buffering of FastLink messages.
- "data:settings:interfaces:fastLink:jobProcessingBrakeAt" (optional): If the largest filling degree of all outgoing FastLink interface buffers exceeds this percentage, all data loading jobs will be freezed until it is not exceeded anymore.
- "data:settings:interfaces:fastLink:transactionMessageProcessingHaltAt" (optional): If the largest filling degree of all FastLink interface buffers exceeds this percentage, all MCI connections are closed until it is not exceeded anymore.
- "data:settings:interfaces:alertMessage" (optional):
- "data:settings:interfaces:alertMessage:ipAddress" (optional): Address of the SMTP server/gateway the alert message interface shall use.
- "data:settings:interfaces:alertMessage:ipPort" (optional): Port of the SMTP server/gateway the alert message interface shall use.

- "data:settings:interfaces:alertMessage:receiveTimeoutSeconds" (optional): Safer Payments closes its receiving sockets and restarts its incoming interface if the timeout expires.
- "data:settings:interfaces:alertMessage:authentication" (optional): SMTP authentication method according to RFC 4954.
- "data:settings:interfaces:alertMessage:username" (optional): SMTP user name to be used for login type SMTP authentication.
- "data:settings:interfaces:alertMessage:password" (optional): SMTP user password to be used for login type SMTP authentication.
- "data:settings:interfaces:alertMessage:from" (optional): This email address will be used as sender and reply-to address with all outgoing email/SMS messages.
- "data:settings:interfaces:alertMessage:sendInterval" (optional): Time period in seconds in which Safer Payments sends out email/SMS.
- "data:settings:interfaces:alertMessage:retryInterval" (optional): Time period in seconds Safer Payments waits before it resends messages that could not be delivered in a previous attempt.
- "data:settings:interfaces:alertMessage:archive" (optional): If checked, successfully sent Email/SMS messages are archived on disk.
- "data:settings:interfaces:alertMessage:useSsl" (optional): Use SSL for communication to the SMTP server.
- "data:settings:interfaces:alertMessage:verificationCertificate" (optional): Load the certificates for one or more trusted certification authorities from this PEM file.
- "data:settings:query" (optional): Contains settings for the different type of queries in Safer Payments.
- "data:settings:query:maxRecordsWarning" (optional): If a user defines a query with a maximum number of records larger than this value, a warning will be generated.
- "data:settings:query:maxRecordsLimit" (optional): Safer Payments will prohibit defining a query with a maximum number of records larger than this value.
- "data:settings:query:groupingQueryAccountsLimit" (optional): Defines the maximum number of displayed accounts in a group by query result.
- "data:settings:query:groupingQueryResultCategoriesLimit" (optional): Defines the maximum number of categories that will be evaluated in a group by query.
- "data:settings:query:queryEnableDdc" (optional): If checked, users may define queries that also use data from DDC.
- "data:settings:query:queryResultLivetime" (optional): Defines the time a query result is stored by Safer Payments.
- "data:settings:query:commonPointQueryResultLivetimeSeconds" (optional): Defines the time a common point query result is stored by Safer Payments.
- "data:settings:query:maxQueryResults" (optional): Maximum number of results that are being stored for a group by query.

- "data:settings:query:apiQueryAccess" (optional): Restricts the access to single API query requests.
- "data:settings:encryption" (optional): Contains settings for encryption and encryption keys.
- "data:settings:encryption:enabled" (optional): Enables functions for key management and encryption of elements.
- "data:settings:encryption:keyReuseEnabled" (optional): If enabled, a previous key that is now inactive can be reactivated in the future.
- "data:settings:encryption:keyRetryTimeoutSeconds" (optional): Time interval in seconds in which Safer Payments attempts to retrieve keys from other cluster instances.
- "data:settings:encryption:maximumKeyLifeBeforeShutdownDays" (optional): After the current key was active for the specified number of days, Safer Payments will automatically shut down.
- "data:settings:encryption:maximumMasterKeyLifeBeforeShutdownDays" (optional): After the current master key was active for the specified number of days, Safer Payments will automatically shut down.
- "data:settings:encryption:wipeDeletedConfigurationAndUserFiles" (optional): If enabled, all configuration and user files are permanently and securely deleted when the respective element stored in it is deleted; if disabled, the respective file is renamed to ".deleted" suffix but not physically removed.
- "data:settings:encryption:changeMasterKeyRelativeWaitFactor" (optional): The relative wait factor for a master key to get transmitted to other Safer Payments instances; "0" for immediate transmission after successful change, "1" for a cascaded master key change.
- "data:settings:encryption:encryptSensitiveExports" (optional): If enabled, CSV exports that contain encrypted values will be wrapped in an AES encrypted zip file.
- "data:settings:eventLogMessages" (optional): Contains various settings for handling of event log messages.
- "data:settings:eventLogMessages:defaultEventLogMessageViewInterval" (optional): Default time period from now into the past for which the UI integrated event log viewer shows log events.
- "data:settings:eventLogMessages:maxRecEventLogMessage" (optional): Maximum number of entries shown in an event log table.
- "data:settings:eventLogMessages:enableAllLogMessages" (optional): If enabled, the individual configuration of event log messages is ignored and all available log messages are always printed.
- "data:settings:eventLogMessages:eventLogMessagesEnabled" (optional): If enabled, messages will be sent to the operating system event log.
- "data:settings:eventLogMessages:syslogTemplate" (optional): If log messages are sent to the operating system event log, a custom format template can be defined here.

- "data:settings:other" (optional): Contains miscellaneous settings which to not fit into other sections.
- "data:settings:other:eodDayTimeString" (optional): The time when the daily end of day job to clean out unused data is started.
- "data:settings:other:shutdownGracePeriod" (optional): Time period in seconds Safer Payments waits for service threads to end after a shutdown was initiated; after this period, the threads are forced to end.
- "data:settings:other:updateInstanceStatusFileSec" (optional): Time interval after which the instance status file is updated; 0 disables the instance status file.
- "data:settings:other:maxReportLimit" (optional): If a user defines a report with a maximum number of records larger than this value, Safer Payments will automatically limit the result.
- "data:settings:other:checkIndexesOnStartup" (optional): If enabled, Safer Payments will check structural consistency of indexes during startup.
- "data:settings:other:fileCreateZeros" (optional): Define when Safer Payments enforces to fill the end of files with zeros on resize.
- "data:settings:other:wipeDdcPosition" (optional): If enabled, the DDC write position of an encrypted attribute will be wiped before writing a new value.
- "data:settings:other:computationThreadBufferElementSize" (optional): Allocated size (in bytes) of a buffer element in each computation thread.
- "data:settings:other:multipleRelationsIndexSearch" (optional): If enabled, it is possible to automatically search for potential indexes when querying multiple relations masterdata.
- "data:settings:other:sslCipherList" (optional): Defines which ciphers are allowed for all SSL-encrypted interfaces, similarly to the Apache setting "SSLCipherSuite".
- "data:settings:other:sslUseCertificateChain" (optional): Enables whether SSL certificates are verified against certificate chains if given.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

406 SendCaseActionFromPreview

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "sendCaseActionFromPreview",
    "uid": <caseUid>,
    "mandator": <mandatorUid>,
    "data": {
        "toAddress": "<targetEmailAddress>",
        "caseActionUid": <caseActionUid>,
        "subjectTemplate": "<subjectTemplate>",
        "textModuleUid": <textModuleUid>,
        "bodyTemplate": "<bodyTemplate>",
        "bodyTemplateHtml": "<bodyTemplateHtml>",
        "comment": "<comment>",
        "table": <transactionTableData>
    }
}
```

- "uid" (mandatory): Uid of the case to fill case action data from.
- "mandator" (mandatory): Uid of the mandator to which the case belongs.
- "data:toAddress": The address to which the SMTP message shall be delivered (SMTP only).
- "data:caseActionUid": Uid of the case action which is to be sent (SMTP or SQL).
- "data:subjectTemplate": The text which is used for the subject of an SMTP message (SMTP only).
- "data:textModuleUid" (optional): Uid of a text module.
- "data:bodyTemplate" (mandatory): The content of the email or SQL statement.

- "data:bodyTemplateHtml" (optional): The content which supports HTML formatted text (SMTP only).
- "comment" (optional): Comment to be added to the case audit trail.
- "table" (optional): Transaction data may be added in form of a transaction table.
- "table:header" (optional): Transaction data table header.
- "table:data" (optional): Transaction data table content.

All templates may include placeholders for {reporting attributes}, {{transaction data attribute values}}, [[masterdata values]], [CaseVariables], [TextModules]

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

407 SendCaseActions

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "sendCaseActions",
    "uid": <caseUid>,
    "mandator": <mandatorUid>,
    "data": {
        "caseActions": [<caseActionUid>],
        "comment": "<comment>"
    }
}
```

- "uid" (mandatory): Uid of the case that shall be sent via case action.
- "mandator" (mandatory): Uid of the mandator that the case belongs to.
- "data:caseActions" (mandatory): A list of case action uids which shall be sent.
- "data:comment" (optional): Comment that shall be added to the case audit trail.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

408 Serialize

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "serialize",
    "uid": <elementUid>
}
```

- "uid" (mandatory): Uid of the element to write to disk.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

409 SetAllMyPreferencesAsDefault

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setAllMyPreferencesAsDefault"
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

410 SetAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setAnalysis",
    "uid": -1|<analysisId>,
    "revision": <revisionUid>,
    "data": {
        "enabled": true|false,
        "name": "<analysisName>",
        "comment": "<comment>",
        "optimization": true|false,
        "adoptSimulationDataSelection": true|false,
        "dataSelection": {
            "mandators": [<mandatorUid>],
            "periodType": "TimeAbsolute" | "RecordsAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>,
            },
            "conditions": [...]
        },
        "execute": true|false
    }
}
```

- "uid" (mandatory): Use -1 for new analysis or the ID of an existing analysis. The "analysisId" can be obtained from the request of "getAnalysisTable".
- "revision" (mandatory): Revision uid to identify the revision to which the analysis belongs.

- "data:enabled" (mandatory): Enable or disable the analysis.
- "data:name" (mandatory): Name for the analysis.
- "data:comment" (optional): Provide a description or additional info for this analysis.
- "data:optimization" (optional): Enable or disable analysis optimization.
- "data:adoptSimulationDataSelection" (optional): Take data selection from simulation, otherwise use custom values as follows.
- "data:dataSelection" (optional): Provide a custom data selection if "adoptSimulationDataSelection" is not enabled.
- "data:dataSelection:mandators" (optional): Analyze transactions from these mandators.
- "data:dataSelection:periodType" (optional): Provide data selection in form of time or records.
- "data:dataSelection:desired" (optional): Contains the from-to values for data selection. If periodType is TimeAbsolute, the timestamp keys must be filled. If it is RecordsAbsolute, the urid keys must be filled.
- "data:dataSelection:desired:fromUrid" (optional): Analyze transactions starting from this URID.
- "data:dataSelection:desired:fromTimestamp" (optional): Analyze transactions that are younger than this time.
- "data:dataSelection:desired:toUrid" (optional): Analyze transactions up to this URID.
- "data:dataSelection:desired:toTimestamp" (optional): Analyze transactions that are older than this time.
- "data:dataSelection:conditions" (optional): Analyze only transactions which meet these conditions.
- "data:execute" (optional): Execute the analysis immediately after saving.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

411 SetAttachmentData

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setAttachmentData",
    "uid": <caseUid>,
    "data": {
        "id": <attachmentId>,
        "comment": "<comment>"
    }
}
```

- "uid" (mandatory): Uid of the investigation case to which the attachment belongs.
- "data:id" (mandatory): Id of the attachment.
- "data:comment" (optional): Information to be saved to the attachment.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

412 SetCrashWorkflow

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setCrashWorkflow",
    "uid": <instanceUid>,
    "data": {
        "workflow": "start"|"forceHealthy"|"invalidate"|"ignore"|"restore",
        "donorUid": <donorInstanceUid>
    }
}
```

- "uid": Uid of the instance to apply the workflow after it is crashed.
- "workflow": The status to set to a crashed instance. The workflow of "start" means to continue the default startup. The workflow of "forceHealthy" forcefully sets the instance to Healthy status, which may result in data loss or further synchronization problems. The workflow of "invalidate" invalidates the instance. The workflow of "ignore" resets the crash workflow. The workflow of "restore" restores the instance from a donor instance. By default, the instance is invalidated after a crash.
- "donorUid": Uid of the donor instance. Only required if "workflow" is set to "restore".

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

413 SetFinalRulesetsEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setFinalRulesetsEnabled",
    "revision": <revisionUid>,
    "data": {
        "enabled": true|false,
        "rulesets": [<rulesetUid>]
    }
}
```

- "revision": The revision to which final rulesets belong.
- "data:enabled": Flag indicating enabled/disabled operation.
- "data:rulesets": The final rulesets that should be enabled/disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

414 SetInternalModelGenerationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setInternalModelGenerationDataSelection",
    "revision": <revisionUid>,
    "data": {
        "trainingDataSelectionStream": {
            "mandators": [<mandatorUid>],
            "periodType": "RecordsAbsolute" | "TimeAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>
            },
            "conditions": [ <conditions> ]
        },
        "verificationDataSelectionStream": {
            "mandators": [<mandatorUid>],
            "periodType": "RecordsAbsolute" | "TimeAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>
            },
            "conditions": [ <conditions> ]
        }
    }
}
```

- "revision" (mandatory): Uid of the revision in which the rule generation data selection is to be set.
- "data:verify" (optional): Enables/disables verification of data selection.
- "data:generationScenario" (optional):
- "data:trainingDataSelectionStream" (mandatory): Defines which data is used for rule generation.
- "data:trainingDataSelectionStream:mandators" (optional): Uids of a list of mandators whose transactions shall be included.
- "data:trainingDataSelectionStream:periodType" (optional): Choose whether to provide data selection in form of time or records.
- "data:trainingDataSelectionStream:desired" (optional): Contains the from-to values for data selection. If "periodType" is "TimeAbsolute", the timestamp keys must be filled. If it is "RecordsAbsolute", the urid keys must be filled.
- "data:trainingDataSelectionStream:desired:fromUrid" (optional): Transactions whose urids are greater than this URID shall be included.
- "data:trainingDataSelectionStream:desired:fromTimestamp" (optional): Transactions created after this Unix timestamp shall be included.
- "data:trainingDataSelectionStream:desired:toUrid" (optional): Transactions whosed urids are snaller than this URID shall be included.
- "data:trainingDataSelectionStream:desired:toTimestamp" (optional): Transactions created before this Unix timestamp shall be included.
- "data:trainingDataSelectionStream:conditions" (optional): Include only transactions that meet these conditions.
- "data:verificationDataSelectionStream" (optional): Defines which data is used for verification of rule generation.
- "data:verificationDataSelectionStream:mandators" (optional): Verify the generated rules against transactions from these mandators.
- "data:verificationDataSelectionStream:periodType" (optional): Choose whether to provide data selection in form of time or records.
- "data:verificationDataSelectionStream:desired" (optional): Contains the from-to values for data selection. If periodType is TimeAbsolute, the timestamp keys must be filled. If it is RecordsAbsolute, the urid keys must be filled.
- "data:verificationDataSelectionStream:desired:fromUrid" (optional): Transactions whose urids are greater than this URID shall be included for verification.
- "data:verificationDataSelectionStream:desired:fromTimestamp" (optional): Transactions created after this Unix timestamp shall be included for verification.
- "data:verificationDataSelectionStream:desired:toUrid" (optional): Transactions whosed urids are snaller than this URID shall be included for verification.
- "data:verificationDataSelectionStream:desired:toTimestamp" (optional): Transactions created before this Unix timestamp shall be included for verification.
- "data:verificationDataSelectionStream:conditions" (optional): Verify the generated rules only against transactions that meet these conditions.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

415 SetInternalModelGenerationSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setInternalModelGenerationSettings",
    "revision": <revisionUid>,
    "data": {
        "maxDepthOfTree": <depthOfTree>,
        "minSamplesSplit": <minSamplesSplit>,
        "treeCount": <treeCount>,
        "maxFeatures": <maxFeatures>,
        "truncatePrunedTree": <truncatePrunedTree>,
        "genuineLabelWeight": <genuineLabelWeight>,
        "fraudLabelWeight": <fraudLabelWeight>,
        "predictionFlag": <attributeUid>,
        "probabilityFraud": <attributeUid>,
        "probabilityGenuine": <attributeUid>
        "name": "<name>",
        "comment": "<comment>",
    }
}
```

- "revision" (mandatory): Uid of the revision whose model generation settings shall be changed.
- "data:maxDepthOfTree" (optional): Determines how much Safer Payments favors hit rate over false positives.
- "data:minSamplesSplit" (optional): Any false positive value below this value is considered to be indifferent to this value.
- "data:treeCount" (optional): Hit rate in percent at which automatic rule generation stops.
- "data:maxFeatures" (optional): Stop rule generation if the last generated rule has false positives equal to or greater than this setting.

- "data:truncatePrunedTree" (optional): Maximum number of conditions that are generated for each rule.
- "data:genuineLabelWeight" (optional): Weight of genuine label prediction.
- "data:fraudLabelWeight" (optional): Weight of fraud label prediction.
- "data:predictionFlag": Uid of the attribute that will be used to store prediction class of the transaction by the generated model.
- "data:probabilityFraud" (optional): Uid of the attribute that will be used to store the percentage of trees indicating transaction to be fraud.
- "data:probabilityGenuine" (optional): Uid of the attribute that will be used to store the percentage of trees indicating transaction to be genuine.
- "data:name" (optional): A template for the names of the generated model. Use {n} for model number.
- "data:comment" (optional): A template for the comments of the generated model. Available placeholders: {n}, {timestamp}, {revision}, {user}.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

416 SetKey

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setKey",
    "uid": <keyUid>,
    "data": {
        "streamType": "key",
        "uid": <keyUid>,
        "public": {
            "right"|"left": {
                "passphrase": "<passphrase>",
                "enteredComment": "<comment>"
            }
        }
    }
}
```

- "uid" (mandatory): Uid of the public encryption key for which to set the left or right key.
- "data" (mandatory): Contains the entered passphrase.
- "data:uid" (mandatory): Uid of the public encryption key.
- "data:public" (mandatory): Contains the left or right passphrase.
- "data:right|left" (mandatory): Usage depends on which key was entered.
- "data:right|left:phassphrase": The entered passphrase.
- "data:right|left:enteredComment": The entered comment.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

417 SetMasterdata

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setMasterdata",
    "uid": <masterdataUid>,
    "mandator": <mandatorUid>,
    "data": {
        "parameter": "<parameter>",
        "value": "<value>"
    }
}
```

- "uid" (mandatory): Uid of the masterdata which is to be assigned the new value.
- "mandator" (mandatory): Uid of the mandator in which the masterdata is defined.
- "data:parameter" (mandatory): The index entry that is to be assigned with the new masterdata value.
- "data:value" (mandatory): The value that shall be assigned to the given index entry.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

418 SetModelComponentsEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setModelComponentsEnabled",
    "revision": <revisionUid>,
    "data": {
        "enabled": true|false,
        "rulesets": [<rulesetUid>]
    }
}
```

- "revision": Uid of the revision that the model components belong to.
- "data:enabled": Flag indicating enabled/disabled operation.
- "data:modelComponents": Uids of the model components that should be enabled/disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

419 SetModelling

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setModelling",
    "revision": <revisionUid>,
    "data": {
        "attribute": <attributeUid>,
        "testEnabled": true|false,
        "simulationEnabled": true|false,
        "analysisEnabled": true|false,
        "modelGenerationEnabled": true|false,
    }
}
```

- "revision" (mandatory): Uid of the revision in which the modelling setting shall be changed.
- "data:attribute" (mandatory): Uid of the attribute.
- "data:testEnabled" (optional): Enables the attribute for sandbox computations.
- "data:simulationEnabled" (optional): Enables the attribute for simulation.
- "data:analysisEnabled" (optional): Enables the attribute for analyses.
- "data:modelGenerationEnabled" (optional): Enables the attribute for model generation.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

420 SetMultiInterfaces

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setMultiInterfaces",
    "data": {
        "interfaceType": "<interfaceType>",
        "instances": [<instanceId>],
        "enable": true|false
    }
}
```

- "data" (mandatory): Values to identify the interface to enable/disable and the instances on which to perform the action.
- "data:interfaceType" (mandatory): Type of the interface to enable/disable. Available values for the "interfaceType" are:

```
"MessageCommandInterface"
"ApplicationProgrammingInterface"
"BatchDataInterface"
"FastLinkInterface"
"StatusControlInterface"
"EncryptedCommunicationInterface"
"AlertMessageInterface"
"WebSphereMQInterface"
```

- "data:instances" (mandatory): A list of IrisInstance objects (identified by their uids) to change interface settings of.
- "data:enable" (mandatory): Indicates if interface should be enabled or disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

421 SetMultiModelling

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setMultiModelling",
    "revision": <revisionUid>,
    "data": {
        "modellingType": "Test" | "Simulation" | "Analysis" | "RuleGeneration",
        "enable": true|false,
        "attributes": [<attributeUid>]
    }
}
```

- "revision" (mandatory): Uid of the revision in which modelling settings shall be changed.
- "data:modellingType" (mandatory): The modelling setting that shall be changed for all the given attributes.
- "data:enable" (mandatory): Enable or disable the given modelling setting for all given attributes.
- "data:attributes" (mandatory): Uids of a list of attributes for which the modelling setting shall be changed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

422 SetOffline

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setOffline",
    "uid": <instanceUid>
}
```

- "uid": Uid of the instance to take offline.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

423 SetOnline

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setOnline",
    "uid": <instanceUid>
}
```

- "uid": Uid of the instance to go online.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

424 SetPreference

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setPreference",
    "type": <type>,
    "data": {
        ... ...
    }
}
```

- "type" (mandatory): Controls which preferences should be set.
- "data" (mandatory): Depending on "type" contains the new preferences.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

425 SetQuickSearchCasesTablePreference

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setQuickSearchCasesTablePreference",
    "data": {
        "caseSearch": {
            "attribute": <attributeUid>,
            "value": "<value>"
        }
    }
}
```

- "data:caseSearch:attribute" (mandatory): Uid of the attribute that is used to search for cases.
- "data:caseSearch:value" (mandatory): Value of the given attribute that is used to search for cases.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

426 SetRuleGenerationCategoricIndicators

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationCategoricIndicators",
    "uid": <attributeUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "value": true|false,
        "indicators": [<indicatorId>]
    }
}
```

- "uid" (mandatory): Uid of the attribute whose indicators shall be set.
- "mandator" (mandatory): Uid of the mandator in which the rule generation is taking place.
- "revision" (mandatory): Uid of the revision in which rule generation is taking place.
- "data:value" (mandatory): Decides whether to enable or disable the indicators.
- "data:indicators" (mandatory): A list of indicators that shall be enabled or disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

427 SetRuleGenerationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationDataSelection",
    "revision": <revisionUid>,
    "data": {
        "verify": true|false,
        "generationScenario": "FromScratch" | "CurrentRevisionOnly" | "CurrentRevisionAndHistory",
        "trainingDataSelectionStream": {
            "mandators": [<mandatorUid>],
            "periodType": "RecordsAbsolute" | "TimeAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>
            },
            "conditions": [ <conditions> ]
        },
        "verificationDataSelectionStream": {
            "mandators": [<mandatorUid>],
            "periodType": "RecordsAbsolute" | "TimeAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>
            },
            "conditions": [ <conditions> ]
        }
    }
}
```

- "revision" (mandatory): Uid of the revision in which the rule generation data selection is to be set.
- "data:verify" (optional): Enables/disables verification of data selection.
- "data:generationScenario" (optional): Choose one of the rule generation scenarios.
- "data:trainingDataSelectionStream" (mandatory): Defines which data is used for rule generation.
- "data:trainingDataSelectionStream:mandators" (optional): Uids of mandators that the transactions are from.
- "data:trainingDataSelectionStream:periodType" (optional): Choose whether to provide data selection in form of time or records.
- "data:trainingDataSelectionStream:desired" (optional): Contains the from-to values for data selection. If periodType is "TimeAbsolute", the timestamp keys must be filled. If it is "RecordsAbsolute", the urid keys must be filled.
- "data:trainingDataSelectionStream:desired:fromUrid" (optional): Required by "RecordsAbsolute" period type, transactions whose urid is larger than this URID value shall be included.
- "data:trainingDataSelectionStream:desired:fromTimestamp" (optional): Required by "TimeAbsolute" period type, transactions created after this Unix timestamp value shall be included.
- "data:trainingDataSelectionStream:desired:toUrid" (optional): Required by "RecordsAbsolute" period type, transactions whose urid is smaller than this URID value shall be included.
- "data:trainingDataSelectionStream:desired:toTimestamp" (optional): Required by "TimeAbsolute" period type, transactions created before this Unix timestamp value shall be included.
- "data:trainingDataSelectionStream:conditions" (optional): Include only transactions that meet these conditions.
- "data:verificationDataSelectionStream" (optional): Defines transaction data that shall be used for verification of rule generation.
- "data:verificationDataSelectionStream:mandators" (optional): Uids of mandators whose transactions shall be included for verification.
- "data:verificationDataSelectionStream:periodType" (optional): Choose whether to provide data selection in form of time or records.
- "data:verificationDataSelectionStream:desired" (optional): Contains the from-to values for data selection. If periodType is "TimeAbsolute, the timestamp keys must be filled. If it is "RecordsAbsolute", the urid keys must be filled. - "data:verificationDataSelectionStream:desired:fromUrid" (optional): Required by "Records↩ Absolute" period type, transactions whose urid is larger than this URID value shall be used for verification. - "data:verificationDataSelectionStream:desired↩ :fromTimestamp" (optional): Required by "TimeAbsolute" period type, transactions created after this Unix timestamp value shall be used for verification. - "data↩ :verificationDataSelectionStream:desired:toUrid" (optional): Required by "RecordsAbsolute" period type, transactions whose urid is smaller than this URID value

shall be used for verification. - "data:verificationDataSelectionStream:desired:toTimestamp" (optional): Required by "TimeAbsolute" period type, transactions created before this Unix timestamp value shall be used for verification. - "data:verificationDataSelectionStream:conditions" (optional): Include only transactions that meet these conditions for verification.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

428 SetRuleGenerationFollowingIndicators

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationFollowingIndicators",
    "uid": <attributeUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "value": true|false,
        "fromIndicator": <fromIndicatorId>,
        "toIndicator": <toIndicatorId>
    }
}
```

- "uid" (mandatory): Uid of the attribute whose indicators shall be set.
- "mandator" (mandatory): Uid of the mandator in which the rule generation is taking place.
- "revision" (mandatory): Uid of the revision in which rule generation is taking place.
- "data:value" (mandatory): Decides whether to enable or disable the indicators.
- "data:fromIndicator" (mandatory): Start from this indicator, value is given as row number in indicators table. Must be smaller than "toIndicator".
- "data:toIndicator" (mandatory) End with this indicator, value is given as row number in indicators table. Must be greater than "fromIndicator".

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

429 SetRuleGenerationIndicator

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationIndicator",
    "uid": <attributeUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "indicator": <indicatorId>,
        "value": true|false
    }
}
```

- "uid" (mandatory): Uid of the attribute whose indicator shall be set.
- "mandator" (mandatory): Uid of the mandator in which the rule generation is taking place.
- "revision" (mandatory): Uid of the revision in which rule generation is taking place.
- "data:indicator" (mandatory): Identifies the indicator by row number in the indicator table.
- "data:value" (mandatory): Decides whether to enable or disable the indicator.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

430 SetRuleGenerationPrecedingIndicators

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationPrecedingIndicators",
    "uid": <attributeUid>,
    "mandator": <mandatorUid>,
    "revision": <revisionUid>,
    "data": {
        "value": true|false,
        "fromIndicator": <fromIndicatorId>,
        "toIndicator": <toIndicatorId>
    }
}
```

- "uid" (mandatory): Uid of the attribute whose indicators shall be set.
- "mandator" (mandatory): Uid of the mandator in which the rule generation is taking place.
- "revision" (mandatory): Uid of the revision in which rule generation is taking place.
- "data:value" (mandatory): Decides whether to enable or disable the indicators.
- "data:fromIndicator" (mandatory): Start from this indicator, value is given as row number in indicators table. Must be greater than toIndicator.
- "data:toIndicator" (mandatory) End with this indicator, value is given as row number in indicators table. Must be smaller than fromIndicator.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

431 SetRuleGenerationSettings

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRuleGenerationSettings",
    "revision": <revisionUid>,
    "data": {
        "balanceFraudFalsePositives": <weightingSetting>,
        "lowerBoundFalsePositives": <minFalsePositives>,
        "upperBoundFraudHitRate": <upperBoundHitRate>,
        "stopRuleFalsePositives": <stopFalsePositives>,
        "maxNumberConditions": <maxNumberOfConditions>,
        "minPopulationQuantise": <minPopulationQuantise>,
        "minSampleSizePercent": <minSampleSizeInPercent>,
        "minSelectHitRate": <minSelectHitRate>,
        "minConditionImprovement": <minConditionImprovement>,
        "relaxAllUpAutoThreshold": <relaxAllUpThreshold>,
        "maxNumberRules": <maxNumberOfRules>,
        "maxFraudHit": <maxHitRate>,
        "rulesName": "<ruleName>",
        "rulesComment": "<comment>",
        "aPrioriConditions": [ <predefinedConditions> ],
        "ruleConclusion": {
            "interceptValue": <interceptValue>
        }
    }
}
```

- "revision" (mandatory): Uid of the revision whose rule generation settings shall be changed.
- "data:balanceFraudFalsePositives" (optional): Determines how much Safer Payments favors hit rate over false positives.

- "data:lowerBoundFalsePositives" (optional): Any false positive value below this value is considered to be indifferent to this value.
- "data:upperBoundFraudHitRate" (optional): Hit rate in percent at which automatic rule generation stops.
- "data:stopRuleFalsePositives" (optional): Stop rule generation if the last generated rule has false positives equal to or greater than this setting.
- "data:maxNumberConditions" (optional): Maximum number of conditions that are generated for each rule.
- "data:minPopulationQuantise" (optional): Minimum number of records that a quantile must have.
- "data:minSampleSizePercent" (optional): Relative number of records covered by a new condition that must be met to be suggested as a condition.
- "data:minSelectHitRate" (optional): Minimum hit rate for automated selection of an indicator.
- "data:minConditionImprovement" (optional): Minimum quality improvement that adding a new condition must provide to be suggested.
- "data:relaxAllUpAutoThreshold" (optional): Threshold in quality for complete relaxation of indicators up from current.
- "data:maxNumberRules" (optional): Maximum number of rules that shall be generated during fully automated rule generation.
- "data:maxFraudHit" (optional): Any hit rate value above this setting is considered to be indifferent to the value.
- "data:rulesName" (optional): A template for the names of the generated rules. Use {n} for rule number.
- "data:rulesComment" (optional): A template for the comments of the generated rules. Available placeholders: {n}, {timestamp}, {revision}, {user}.
- "data:aPrioriConditions" (optional): Conditions that a record must meet to be included in rule generation.
- "data:ruleConclusion:interceptValue" (optional): The intercept value which will be set by all generated rules as a rule conclusion.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

432 SetRulesEnabled

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setRulesEnabled",
    "revision": <revisionUid>,
    "data": {
        "enabled": true|false,
        "rules": [<ruleUid>]
    }
}
```

- "revision" (mandatory): Uid of the revision that the rules to be enabled/disabled belong to.
- "data:enabled" (mandatory): A flag to decide whether the rules should be enabled or disabled.
- "data:rules" (mandatory): Uids of a list of rules to be enabled/disabled.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

433 SetSimulationDataSelection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setSimulationDataSelection",
    "revision": <revisionUid>,
    "data": {
        "simulationDataSelection": {
            "mandators": [<mandatorUid>],
            "periodType": "RecordsAbsolute" | "TimeAbsolute",
            "desired": {
                "fromUrid": <fromUrid>,
                "fromTimestamp": <fromTimestamp>,
                "toUrid": <toUrid>,
                "toTimestamp": <toTimestamp>,
            },
            "incDdc": true|false,
            "conditions": [ <conditions> ]
        },
        "simulationMethod": "PerChunk" | "PerRecord",
        "targetConditions": [ <conditions> ]
    }
}
```

- "revision" (mandatory): Uid of the revision that the simulation belongs to.
- "data:simulationDataSelection:mandators" (mandatory): Uids of a list of the mandators whose data should be included in the simulation.
- "data:simulationDataSelection:periodType" (mandatory): A string that indicates whether the data selection refers to urids or a time span.
- "data:simulationDataSelection:desired:fromUrid" (optional): If period type "RecordsAbsolute", the desired first urid of the data selection.

- "data:simulationDataSelection:desired:fromTimestamp" (optional): If period type "TimeAbsolute", the desired start timestamp of the data selection.
- "data:simulationDataSelection:desired:toUrid" (optional): If period type "RecordsAbsolute", the desired last urid of the data selection.
- "data:simulationDataSelection:desired:toTimestamp" (optional): If period type "TimeAbsolute", the desired end timestamp of the data selection.
- "data:simulationDataSelection:incDdc" (mandatory): A flag decides whether or not data from the DDC may be accessed during simulation.
- "data:simulationDataSelection:incDdc" (optional): An array of conditions to furhter restrict the data selection.
- "data:simulationMethod" (mandatory): A string that indicates whether the simulation is performed per chunk or per record.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

434 SetUserExportPassword

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setUserExportPassword",
    "uid": <queryUid>|<groupByQueryUid>|<definedRiskListUid>,
    "revision": <revisionUid>,
    "mandator": <mandator>,
    "data": {
        "streamType": "casesExport" | "cppExport" | "queryResultExport" | "definedRiskListExport",
        "exportPassword": "<password>"
    }
}
```

- "uid" (optional): Uid of the Query/GroupByQuery/DefinedRiskList, if the streamType is "queryResultExport" or "definedRiskListExport".
- "revision" (optional): Uid of the revision, if the streamType is "queryResultExport" and the export is for a simulation query.
- "mandator" (optional): Uid the mandator of the exported element, if the streampType is "definedRiskListExport".
- "data:streamType" (mandatory): A string to identify which kind of table is exported.
- "data:exportPassword" (mandatory): The password to be set.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

435 SetXdcSizes

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "setXdcSizes",
    "type": "DDC" | "MDC" | "MdcWhereDdc",
    "revision": <revisionUid>,
    "data": {
        "newSize": <newSize>
    }
}
```

- "type" (mandatory): Decides which storage size is set to the new size.
- "revision" (mandatory): Uid of the challenger revision to change sizes.
- "data:newSize" (mandatory): The new size to be set to.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

436 Shutdown

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "shutdown",
    "uid": <instanceUid>
}
```

- "uid" (mandatory): The IrisInstance to shutdown.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

437 SimulateCollusion

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "simulateCollusion",
    "uid": <collusionUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the collusion to simulate.
- "revision" (mandatory): Uid of the revision in which the collusion simulation is excuted.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

438 StartAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "startAnalysis",
    "revision": <revisionUid>,
    "data": {
        "analysisId": [<analysisId>]
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analyses are executed.
- "data:analysisId" (mandatory): Ids of analyses to execute.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

439 StartAutomaticRuleGeneration

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "startAutomaticRuleGeneration",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the rule generation is executed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

440 StartInternalModelGeneration

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "startInternalModelGeneration",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the model generation is executed.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

441 StopAnalysis

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopAnalysis",
    "revision": <revisionUid>,
    "data": {
        "analysisId": [<analysisId>]
    }
}
```

- "revision" (mandatory): Uid of the revision in which the analysis is running.
- "data:analysisId" (mandatory): Ids of the analyses to stop.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

442 StopAutomaticRuleGeneration

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopAutomaticRuleGeneration",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision in which the rule generation is running.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

443 StopCollusionSimulation

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopCollusionSimulation",
    "uid": <collusionUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): Uid of the collusion that is currently simulated.
- "revision" (mandatory): Uid of the revision in which the collusion simulation takes place.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

444 StopInternalModelGeneration

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopInternalModelGeneration",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision whose model generation shall be stopped.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

445 StopJob

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopJob",
    "uid": <jobUid>
}
```

- "uid": Uid of the job to be stopped.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

446 StopQuery

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopQuery",
    "uid": <queryUid>,
    "revision": <revisionUid>,
    "data": {
        "resultId": <resultId>
    }
}
```

- "uid" (mandatory): Uid of the query to be stopped.
- "revision" (optional): Uid of the revision, if the query is a simulation query.
- "data:resultId" (mandatory): Id of the query result whose computation should be stopped.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

447 StopRuleGeneration

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "stopRuleGeneration",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision whose rule generation shall be stopped.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

448 TakeoverCase

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "takeoverCase",
    "uid": <caseUid>
}
```

- "uid" (mandatory): Uid of the investigation case which should be taken over.

Returns

A HTTP response with the following content:

```
{
    "data": {
        "streamType": "case",
        "case": {
            "uid": <caseUid>,
            ... ...
        },
        ... ...
    },
    "editable": true|false,
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

449 TakeoverRevision

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "takeoverRevision",
    "revision": <revisionUid>
}
```

- "revision" (mandatory): Uid of the revision to be taken over.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

450 Test

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "test",
    "uid": <caseActionUid>,
    "mandator": <mandatorUid>,
    "data": {
        "caseUid": <caseUid>|-1,
        "textModuleUid": <textModuleUid>|-1
    }
}
```

- "uid" (mandatory): Uid of the case action/notification/external query to be tested.
- "mandator" (mandatory): Uid of the mandator that the element belongs to.
- "data:caseUid" (optional): If a case uid is provided, placeholders will be filled from that case.
- "data:textModuleUid" (optional): If a text module uid is provided, [TextModule] placeholders will be filled from it.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

451 TestLdapConnection

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "testLdapConnection",
    "data": {
        "ldapLogin": "<userName>",
        "useActiveDirectory": true|false,
        "ldapPassword": "<ldapPassword>",
        "hostName": "<ldapServerHostName>",
        "baseDn": "<baseDn>",
        "domain": "<domain>",
        "ldapPort": "<ldapPort>",
        "ssl": true|false
    }
}
```

- "data:ldapLogin" (mandatory): Login name for the LDAP server.
- "data:useActiveDirectory" (optional): If enabled, active directory authentication will be used.
- "data:ldapPassword" (optional): Password for the user login to the LDAP server.
- "data:hostName" (optional): IP address or domain name of LDAP server.
- "data:baseDn" (optional): Base of distinguished name.
- "data:domain" (optional): Active Directory domain for login.
- "data:ldapPort" (optional): IP port of LDAP server.
- "data:ssl" (optional): Enables encrypted communication between IBM Safer Payments and LDAP host.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

452 UnmarkFraud

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "unmarkFraud",
    "data": {
        "urid": [<urid>],
        "fraudValue": <genuineValue>,
        "fraudTimestampAttribute": <timestampAttributeUid>|-1
    }
}
```

- "data:urid" (mandatory): A list of URIDs which shall be marked as genuine.
- "data:fraudValue" (mandatory): The fraud value which shall be assigned to the selected URIDs.
- "fraudTimestampAttribute" (optional): Uid of a fraud timestamp attribute to save the time of marking.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

453 Unreserve

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "unreserve",
    "uid": <elementUid>,
    "revision": <revisionUid>
}
```

- "uid" (mandatory): The element to unreserve.
- "revision" (optional): The Revision to look in. Usage depends on type of element.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbackText>"],
    "reloadUserProfile": true|false
}
```

454 UnreserveCase

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "unreserveCase",
    "uid": <caseUid>
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

455 UnreserveDefinedRiskListEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "unreserveDefinedRiskListEntry",
    "uid": <definedRiskListUid>,
    "mandator": <mandatorUid>,
    "data": {
        "attributeValue": <attributeValue>,
        "id": <definedRiskListEntryId>
    }
}
```

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

456 UnreserveEventLogMessage

Parameters

httpRequest: Holds the request variables that have been passed to the server:

```
{
    "request": "unreserveEventLogMessage",
    "uid": <eventLogMessageId>
}
```

- "uid": Id of the event log message.

Returns

A HTTP response with the following content:

```
{
    "responseStatus": ["<status>", "<feedbacktext>"],
    "reloadUserProfile": true|false
}
```

457 UpdateRiskListEntry

Parameters

httpRequest: Holds the request parameters that have been passed to the server:

```
{
    "request": "updateRiskListEntry",
    "uid" : <definedRiskListUid>,
    "mandator" : <mandatorUid>,
    "data" : {
        "value": "<inputAttributeValue>",
        "outputAttributeCategoryValue" : "<outputAttributeCategoryValue>" | "undefined",
        "expiresAt" : <expiresAt>,
        "startsAt" : <startsAt>,
        "id" : <entryId>,
        "comment" : "<comment>",
        "label" : "<label>",
        "active" : true | false,
        "conditions" : [<conditions>]
    }
}
```

- "uid": Uid of the risk list, in which the entry should be updated.
- "mandator": Uid of mandator, for which the risk list is defined.
- "data:value": The value of the entry.
- "data:outputAttributeCategoryValue" : If a category is defined for the attribute, the category value has to be set.
- "data:expiresAt": If the entry should expire, the date can be assigned here.
- "data:startsAt": If the entry should only be computed after a defined date, the date can be assigned here.

- "data:id": Id of the entry in the list.
- "data:comment" : A comment for this entry can be assigned/changed.
- "data:label" : Label of the entry in the list.
- "data:active" : Defines if the entry should be active or inactive.
- "data:conditions" : Additional conditions that have to be met for this entry to be applied in computation.

Returns

A HTTP response with the following content :

```
{
    "comparisonResult": "Unchanged" | "ChangedNoImpact" | "ChangedComputationImpact",
    "responseStatus" : ["<status>", "<feedbacktext>"],
    "reloadUserProfile" : true | false
}
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.
Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Users' Comments - We'd Like to Hear from You

We appreciate your comments about this publication. Please contact IBM Technical Support if you have question or wish to make comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this reference book. The comment you send should pertain to only the information in this manual or product and the way in which the information is presented.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.