Administrator Manual

requisis_**ReqIF Manager**

# Instructions for the Installation and Configuration of requisis_ReqIF Manager Version 2.11

# requisis_ReqIF Manager

## Table of Contents

# requisis_ReqIF Manager

# 1 Introduction

*requisis_ReqIF Manager* is part of the requisis_Toolbox for DOORS Next Generation (DNG) to exchange modules using the Requirements Interchange Format (ReqIF respectively RIF) between different requirement management databases and your **IBM DOORS Next Generation (DNG)** database. This document provides detailed information for administrators and experienced users about the installation and configuration of the tool.

# 2 Requirements

To run the *requisis_ ReqIF Manager* the following conditions are required:

- IBM DOORS Next Generation 6.0.6 or later
- Local Installation of Docker Engine or Docker on a server and Docker compose
- Delivery Package of *requisis_ ReqIF Manager* provided by REQUISIS

# 3 Installation

To use the *requisis_Toolbox* and its individual components, you need to install the Docker Desktop or Docker on your server. You can download Docker Desktop for free at https://www.docker.com/products/docker-desktop.

The installation of *requisis_ReqIF Manager* is divided into the following steps:

- Get your individual *requisis_Toolbox* zip-package from REQUISIS
- Configuration of an oAuth-Consumer in JTS
- Configuration of various settings like DNG Server Configuration, LDAP configuration
- Creation of SSL certficates and placement in docker storage folder
- Starting the Docker Container

## 3.1 Download the requisis_Toolbox with the requisis_ReqIF Manger

Download the zip file from the REQUISIS download portal (https://requisis.com/de/apps/file-download) with your personal download code and unzip it. If you need a new personal download code, please contact the product support at product-support@requisis.com .

In case this is an Update please unzip the package to a new location first and update your existing files as described in chapter “Update”!

## 3.2 Description of Package and Container

### 3.2.1 Required Containers

Mysql

The mysql-container offers the database services for the requisis_Toolbox.

The container is downloaded from dockerhub.

Requisis-toolbox

The requisis-toolbox container offers the functionality of requisis_Toolbox.

The container will be downloaded from a private repository of requisis. In order to download the container, you need a login that will be provide by REQUISIS as long as you have a valid support contract.

### 3.2.2 Mounts

mysql

| Mount-Point in Container | Mount target | Description |
|---|---|---|
| /docker-entrypoint-initdb.d | ./dist/conf/mysql/sql | Initial SQL-DB content |
| /var/lib/mysql | ./store/mysql/data | MySQL Database content |
| /var/log/mysql | ./store/mysql/log | MySQL Logfiles |

toolbox-server

| Mount-Point in Container | Mount target | Description |
|---|---|---|
| /store | ./store/toolbox/config | Toolbox Server configuration |
| /app/web/uploads | ./store/toolbox/uploads | Content that has been uploaded by user and is being processed by requisis_Toolbox |
| /app/var/logs | ./store/toolbox/logs | Logfiles of requisis_Toolbox |

### 3.2.3 File and Folder Structure of deployment package

| Path and File | Description |
|---|---|
| docker-compose.yml | Docker-compose file.<br>Used to bring up all relevant docker containers |
| .env | Contains Names/URIs for Docker Images and mysql password. |
| startDocker.cmd | Windows Shell Script to pull the latest images and start docker container. |
| createMysqlBackup.cmd | Windows Shell Script to create a mysql database dump. |
| dist\conf\mysql\sql\init.sql | Initial content for the mysql database.<br>Will be imported on first start of mysql docker container. |
| store\mysql\data | Will contain the content of the mysql database.<br>Is empty when delivered. |
| store\mysql\log | Storage of mysql logs. Is empty by default. |
| store\toolbox\config | Contains the configuration of the requisis_Toolbox |

| Path and File | Description |
|---|---|
| store\toolbox\config\parameters.yml | Contains the configuration parameters of requisis_Toobox |
| store\toolbox\config\mix-config.yml | Contains the configuration of DNG-Servers.<br>(Configuration of the requisis_MiX library which is used by requisis_Toolbox to communicate with DNG) |
| store\toolbox\config\features.yml | Contains the licensed and enabled features. **This file will be provided by REQUISIS and must not me modified.** |
| store\toolbox\config\routing.yml | Contains URL path parameters. **This file will be provided by REQUISIS and must not me modified.** |
| store\toolbox\config\.init | This file indicates that the store has been initialized. **This file will be provided by REQUISIS and must not me modified.** |
| store\toolbox\config\mix-certs | This folder contains certificates that might be needed to access the DNG Server. |
| store\toolbox\config\ssl | This folder contains the ssl certificates that are used by the webserver. |
| store\toolbox\config\ssl\server.crt | Contains the server certificate. |
| store\toolbox\config\ssl\server.key | Contains the server certificate private key. **The key file must not be password protected!** |
| store\toolbox\logs | Contains the logs of the requisis_Toolbox |
| store\toolbox\uploads | Contains all the data that is stored by the requisis_Toolbox (besides the data that is stored in the database) |

### 3.2.4 Start Docker using docker-compose

The default `docker-compose.yml` defines how the docker-containers shall be started on your machine / server.

It can be customized to your needs (e.g. changing port numbers or binding the docker to a different network interface)

When the default settings in `docker-compose.yml` are meeting your requirements, you can start the docker containers by navigating into the folder where the `docker-compose.yml` is stored and type:

```
docker-compose up
```

if you want to run the container in foreground

or

```
docker-compose up -d
```

if you want the docker container to be run in background.

To stop the container, you can use

```
docker-compose stop
```

if you only want to stop the container

or

```
docker-compose down
```

if you want to stop and destroy the container. All data is placed outside the container in the /store folder, so it is safe to dispose of the container.

### 3.2.5 Using own docker deployment frameworks

If you want to deploy the docker container in other deployment frameworks like on AWS you can use the docker-compose.yml as a kind of description how the docker containers are interconnected and which data stores need to be mounted.

# 4 Configuration

Before the *requisis_Toolbox* and the *requisis_ReqIF Manager* can be started, you must make several settings according to your infrastructure on a few files. You also need to access the DNG server configuration.

## 4.1 Configuration of DOORS Next Generation

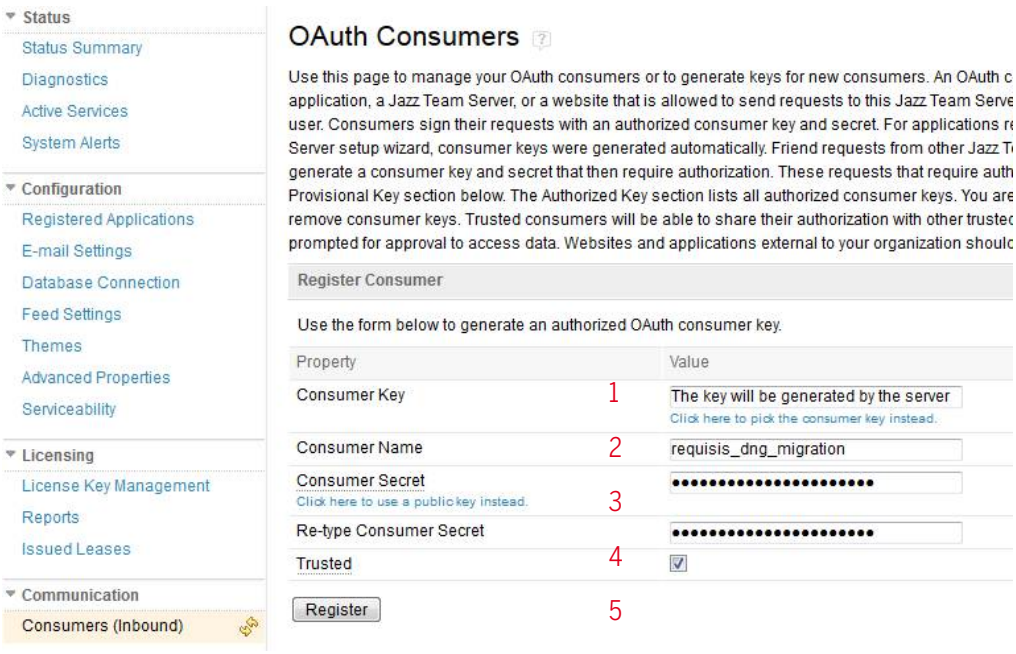
***IMPORTANT:*** To make *requisis_ReqIF Manager* working with DNG you need to setup an OAuth Consumer Key and Secret in the JTS Application of the Jazz Platform.

*Do not setup the oAuth Consumer Key and Secret in the RM Application!*

Instructions:

Open "**https://(your-server:your-port)/jts/admin#action=com.ibm.team.repository.admin.configureOAuth**".

The following page should open:




1) Set *Consumer Key*: Either choose one or note down the key generated by the server.
2) Set *Consumer Name*: Set to "requisis_ReqIF Manger" (or whatever you like)
3) Set *Secret Key*: Choose one
4) Set *Trusted*: Activate checkbox
5) Click on *Register*

## 4.2 Configuring DNG-Servers in mix-config.json file

Location: store\toolbox\config

**Important Note:** You need to restart the docker container to allow the docker container to pull the new configuration.

Place your DNG server URL and OAuth consumer key here. You can configure multiple servers. The file should then look like the image below. In this example two servers were configured.

```
{
  "serverTargets": [
    {
      "targetUrl": "https:\/\/your-server1:9443",
      "oauth_autoLogin": true,
      "oauth_clientId": "tbd",
      "oauth_clientSecret": "tbd",
      "maxSynchronousCurlRequests": 5,
      "databaseCreatedPriorToVersion7": true
    },
    {
      "targetUrl": "https:\/\/your-server2:9443",
      "oauth_autoLogin": true,
      "oauth_clientId": "tbd",
      "oauth_clientSecret": "tbd",
      "maxSynchronousCurlRequests": 10,
      "databaseCreatedPriorToVersion7": false
    }
  ],
  "maxSynchronousCurlRequestsDefault": 5,
  "proxy": "",
  "indexerCheckInterval": 100,
  "preferPublicApi": true,
  "curlTimeout": 120,
  "curlConnectionTimeout": 60
}
```

Please note that slash `/` needs to be escaped with a backslash `\` in the json format and don't forget to save the file for your changes to take effect. For DNG databases created in version 7 or later, no Correlator service is required. With the parameter `databaseCreatedPriorToVersion7` you enable the Correlator service for your databases for version 6 or older. Set the parameter to *false* if the database used was created with version 7 or later.

After the serverTargets section are the following options that apply to any connection to the above targets.

- `maxSynchronousCurlRequestsDefault:`
  Number of synchronously executed requests to the DNG server. Tweek for performance optimization.
  Default value: 5
  **Only change if advised by REQUISIS to do so.**
- `proxy:`
  If necessary, an IP address and port number (syntax: host:port) for a proxy server can be entered here for the connections to the JAZZ server.
  Default value: (empty)

- `indexerCheckInterval:`
  Interval in seconds while waiting for tasks on the JAZZ server
  Default value: 100
  **Only change if advised by REQUISIS to do so.**
- `preferPublicApi:`
  Prefers to use the DNG public API before the private API
  Default value: true
  **Only change if advised by REQUISIS to do so.**
- `curlTimeout:`
  Timeout while waiting for a response from the JAZZ server in seconds.
  Default value: 120
  **Only change if advised by REQUISIS to do so.**
- `curlConnectionTimeout:`
  Timeout while waiting for the connection to the JAZZ server in seconds.
  Default value: 60
  **Only change if advised by REQUISIS to do so.**

### 4.2.1 SSL-Certificates for DNG Server Connections

Depending on the configuration, your DNG server may require an SSL connection to communicate. To set up SSL certificates for a specific server you need to edit the *mix-config.json* file.

In the relevant section of the server concerned, insert the lines `ssl_client_cert_file`, `ssl_client_cert_keyfile` and `ssl_client_cert_keyfile_password`.

As shown here:

```
{
      "targetUrl": "https:\/\/carcompany.com",
      "oauth_autoLogin": true,
      "oauth_clientId": "123ab12a",
      "oauth_clientSecret": "123ab12a",
      "ssl_client_cert_file": "conf/certs/ssl-client-cert.pem",
      "ssl_client_cert_keyfile": "conf/certs/ssl-client-cert.key.pem",
      "ssl_client_cert_keyfile_password": "yourSecretPassword",
      "maxSynchronousCurlRequests": 5
},
```

Put your SSL public and private certificates for the connection to your DNG server here:

`store\toolbox\config\mix-certs`

The certificates will then be copied to the docker container during the next start of the docker container.

## 4.3 Settings in parameters.yml

Location: store\toolbox\config

**Important Note:** You need to restart the docker container to allow the docker container to pull the new configuration.

**Important Note:** Some parameter names and syntax have been changed from Version 1.8 to 1.9!

- `database_host`:
  Host address of your Database. Default value: mysql if the MySQL instance in the docker_compose.yml is also named like this.
- `database_port`:
  Port of your used database. Default value: 3306
- `database_name`:
  Database mysql table for working. Default value: requisis_toolbox
- `database_user`:
  Username for the database login. Default value: toolbox
- `database_password`:
  Password for the username to login in your database. Default value: mysql
- `secret:`
  Salt for encryption / decryption, do not change after installation
- `logger.name`
  Name of logfile. Default value: 'requisis_toolbox.log'
- `logger.level`
  loglevel for logging possible values are: debug, error, warning, critical, info Default value: 'error'
- `directory.logs`
  webserver path to logfiles. Default value: ./web/logs
- `directory.files.temp:`
  webserver path for temporary file uploads of requisis_Toolbox. Default value: './web/uploads/temp/[USER]/'
- `directory.files.limit.uploaded`:
  maximum file uploads for requisis_Toolbox. Default value: 20
- `directory.files.user_uploaded:`
  webserver path for file uploads of requisis_Toolbox. Default value: './web/uploads/uploaded/[USER]/'
- `directory.files.user_storage:`
  webserver path for stored files of requisis_Toolbox. Default value: '/web/uploads/storage/[USER]/'
- `auth.session.timeout:`
  number of seconds of inactivity after which the session becomes invalid. Default value: 3600
- `auth.ldap.enable: 'true'`
  Switch the LDAP authentication.
- `auth.ldap.debug: 'false'`
  Enables the LDAP debug mode for more detailed error messages.
- `auth.ldap.server: 'ldap://ldap-server.company.net'`
  The address of your LDAP server / Active Directory.
- `auth.ldap.port: 389`
  Port for LDAP.
- `auth.ldap.domain: carcompany`
  The Domain of your LDAP server. Will be added to username in order to authenticate.

- `auth.ldap.dc: 'DC=carcompany,DC=com'`
  LDAP path where users are located.
- `auth.ldap.company: 'CarCompany'`
  Default company name for user if no company name is set in LDAP.
- `auth.ladp.disabled.users.parameter: 'userAccountControl':`
  Checks if a user account is inactive. Comment out to disable
- `auth.ldap.disabled.users.parameter.mask: '2'`
  Defines the value of the parameter which marks a user as inactive. Comment out to disable
- `auth.ldap.supporter.dn: 'admin'`
  LDAP group to grant supporter role to *ReqIF Manager* and access to administration area.
- `auth.ldap.binduser.enable: true`
  Activates the login via a technical user in LDAP. **Required if API is being used!**
- `auth.ldap.binduser.login.dn 'uid=global,ou=users,dc=carcompany,dc=com'`
  Full LDAP DN of the technical user.
- `auth.ldap.binduser.login.pw: 'TechUserPassWord'`
  The password of the technical user.
- `auth.ldap.binduser.searchloginnameby: 'sAMAccountName'`
  Search pattern for the name of the technical user. UID or custom fields would also be possible.
- `auth.dngserver.enable`
  Is authentication via dng server enabled Default value false. **Not implemented yet!**
- `auth.dngserver.prefix`
  DNG server prefix. Default value 'dng/' **Not implemented yet!**
- `auth.dngserver.jtsuri`
  Address of your jazz team server. **Not implemented yet!**
- `auth.oidc.enable: true`
  Switches Authentication via OIDC on/off.
- `auth.oidc.disable_classic_login: false`
  If true: switches off the classic login page and goes directly to the OIDC Auth page.
- `auth.oidc.provider: 'https://login.microsoftonline.com/common/v2.0'`
  The provider URL of the OpenID Server.
- `auth.oidc.client.id: abc123`
  The OIDC client ID. Generated by Provider
- `auth.oidc.client.secret: ''`
  The OIDC client secret.
- `auth.oidc.allow_implicit_flow: true`
  Optional! To use the implicit OIDC flow.
- `auth.oidc.provider_config_params: "issuer": 'https://login.microsoftonline.com/2bf7b4d7-2407-4e42-a8bc-0def1235a78c/v2.0'`
  Optional! Additional provider config parameter as array. Syntax as YAML Spec.
- `auth.oidc.login.url: 'https://login.microsoftonline.com/`
  Optional! The login URLs endpoint - overwrites the value from the well known URL.
- `auth.oidc.login.response_type: id_token%20token`
  Response type for the login action (valid values: 'id_token%20token', 'code')

- `auth.oidc.login.additional_parameter: 'response_mode=form_post&scope=openid+profile+email'`
  Additional parameters which are needed for the login URL.
- `auth.oidc.login.callbackurl: 'http://localhost:8080/auth/oidccallback'`
  The callback URL that is called after login.
- `support.mail: support@company.com'`
  Support email address to be shown at dashboard.
- `Support.text: ''`
  If filled, the default help text in the dashboard will be replaced.
- `testsystem.hosts: ['localhost', '127.0.0.1']`
  Hostnames where the Message "Testsystem" shall be displayed.
- `job.search.maximum.results: 25`
  Default value of results for job search, allowed are: 25, 50, 75, 100, 125, 150
- `parallel.jobs: 4`
  The maximum number of concurrent jobs.
- `parallel.jobs.maximum.modules: 10`
  Run parallel jobs only for jobs with less modules than this value. Used to avoid the server to be blocked by one user exporting huge amount of data for multiple partners at the same time.
- `public.api.key: 123456`
  Shared key for the public API.
- `dng.reqif.check_log_for_error: true`
  Check DNG log for errors.
- `job.notification.uri: 'https://requisis.com/generic_webhook.php'`
  Specify a push URL to send a JSON to as a POST to receive job status changes.
  Parameters: userID, jobId, jobtype, jobStatus, reqifSetId, partnerId
- `xhtml.container.artifact.type: ' '`
  DNG rdf artifact type uri for xhtml container, if empty feature is inactive.
- `xhtml.container.link.type: ' '`
  DNG rdf link type uri used for linking xhtml container with artifact, if empty feature is inactive.
- `xhtml.container.link.direction: artifact2xhtmlContainer`
  direction for linking xhtml container with artifact.
  Possible values are: xhtmlContainer2artifact and artifact2xhtmlContainer, artifact2xhtmlContainer is default.
- `close_exportsession.create_baseline: true`
  Show create baseline dialog on close export session. Default setting is true.
- `close_exportsession.remember_baseline: true`
  Show remember baseline dialog for on export session. Default setting is true. Exist only if create baseline dialog is true.
- `rm.administration.usermanagement.visible: true`
  Specifies whether the User Management tab should be displayed in the administration area.
- `rm.customer.export.force_view: true`
  In the ReqIF Set role Customer you have to select an explicit view to be able to export a module. Default setting is true.
- `rm.allow.importwithoutmapping: true`
  Allows to select the option *Import without Mapping* when mapping in the role of the Suppliers.

- `rm.reqifset.owner.domain.blacklist:`
  `- myDomain.de`
  Users with email addresses of the specified domain cannot become owner of a ReqIF Set.

### 4.3.1 Optional email registration (does not work if ldap is enabled, will be removed in later versions!):

If you want to set up a local email registration for your *requisis_Toolbox*, set this value to true otherwise to false. The following lines must also be configured for logging on by email.

- `auth.registration.allowed: true`
  To enable this feature set the value to *true* otherwise set to *false*.
- `mailer_transport: smtp`
  Protocol to be used by the local mail server.
- `mailer_host: 127.0.0.1`
  Host address of your mail server.
- `mailer_user: null`
  Username to login on your mail server.
- `mailer_password: null`
  Password of the specified user to login on your mail server.

# 5 Cron Commands

## 5.1 Job Cleanup Script

You have the option to use a cleanup script that deletes old jobs to save disk space. To do this, open a console on the *Requisis toolbox server* Docker container. You run the script with the following command:

```
php bin/console requisis:jc:job-cleanup <fileDays> <jobDays>
```

| | | |
|---|---|---|
| > | `fileDays:` | Deletes all files of all jobs (user independent) that are older than 30 days (default). |
| > | `jobDays:` | Deletes all jobs of all users of the last 60 days (default). |

Depending on the number of files and jobs to be deleted, the execution may take several minutes. As soon as the script is finished it reports with the message: *finished jobcleanup script*.

## 5.2 User Cleanup Script

It is possible to query the Active Directory via LDAP with a cron job for inactive/locked users. To start this cron job, open a console on the *Requisis toolbox server* Docker container. The command to start the job is:

```
php bin/console requisis:rm:users-cleanup
```

If a user wants to log in to the requisis_Toolbox who was locked via the cron job and is no longer inactive/locked via LDAP, the account is automatically unlocked again at login.

# 6 Update procedure for new release

Deployment of new versions of ReqIF Manager is done via our Docker container distribution system Harbor. When you licensed ReqIF Manager, you received an account that logs into Harbor to update your ReqIF Manager.

## 6.1 Create mySQL Backup

See backup chapter in this manual.

## 6.2 Shutdown requisis-toolbox Docker Container

To shut down the requisis-toolbox container by open a command line / shell and type:

```
docker-compose stop
```

## 6.3 Check the .env file

You find this file on the top level of the Docker container. In the .env file settings for updating the ReqIF Manager are managed. As well as the used time zones of the ReqIF Manager are determined.

You can adjust the following parameters:

- REQIF_MANAGER_IMAGE=harbor.requisis.com:443/requisis/requisis-toolbox/reqif-manager:2.10.00-stable
  Here you can determine which version should be pulled from the harbor. You can specify an explicit version such as `2.9.03-stable`. Or if you want automatic updates for your version, use the tag `2.9-stable` to get the latest version of the reqif manager 2.9.
- LDAP_IMAGE=harbor.requisis.com:443/requisis_public/requisis-toolbox/ldap-docker:latest-stable
  Automatic updating of the LDAP Docker Container.
- MYSQL_ROOT_PASSWORD=start.root
  Adjust the MySQL password.
- TZ=Europe/Berlin
  Adjust the time zone for OS.
- PHP_DATE_TIMEZONE=Europe/Berlin
  Adjust the time zone for PHP.
- HARBOR_USER=<yourHarborRobotUser>
  The username of your Harbor account to update the reqif Manager.
- PW=<veryLongPassword>
  The password for your harbor account to login.

## 6.4 Start new Docker Container

To start the Reqif Manager open a command line / shell and run the script startDocker.cmd it will check if you are currently using the latest version configured in your .env file and update it if not. Afterwards, the Docker containers are started up again. This can take some time due to the update. This depends on how many DNG modules are used. As soon as you can see the interface of the Reqif Manager in the browser again, the update is finished.

With the command docker-compose logs -f server you can follow the process of update and migration.

## 6.5 Check migration logs

After the container is running again, check the log files to see if there were any issues during database migration.

Please check that your database was updated correctly by checking the content of:

migraions-schema-update-needed.log in folder store/toolbox/logs/app/

If there were no issues, you will find the following text in the log file:

```
[OK] Nothing to update - your database is already in sync with the current entity metadata.
```

In case that differences are shown here. Please contact support and provide the content of

migrations.log in folder store/toolbox/logs/app and the database backup that has been created before.

Do not use the ReqIF-Manager before

# 7 API

The *requisis ReqIF Manager* has an API. The API functions can be studied best with a swagger editor like https://editor.swagger.io/.

**Important:** In order to use the API auth.ldap.bind.user must be configured and enabled.

From version 2.0.1 on the API-Documentation is also accessible usint the following URI:
http://<servername>/requisis/reqifmanager/apiDocumenation

## 7.1 Shared secret calculation

The API uses an API key which is generated as a time-based hash.

The client must build the hash and send it to the server. The shared secret consists of a public API key and salt build by timestamp that changes every 5 minutes.

The server checks this hash accordingly. Accepted are the keys of the current salt as well as the previous and following salt. Here an example how the hash of the shared secret is created:

PseudoCode:

```
salt = floor(current_unix_timestamp/300)    //timestamp for every 5 minutes
api-key = sha256(publicApiKey+salt)         //string concatenation
```

Example:

```
SharedSecret = 123456
The Current Unix Timestamp
1605600862 seconds since Jan 01 1970. (UTC)
salt = roundDown(5352002,8733..)
salt = 5352002
api-key = sha256(SharedSecret+salt)
api-key = sha256(1234565352002)
api-key = dd8a093c01dfef50d02eec4393ecdbf1da1cb27edc1f2dc4f2e602d7c1b10d21
```

Header information for API request:

```
"api-key: dd8a093c01dfef50d02eec4393ecdbf1da1cb27edc1f2dc4f2e602d7c1b10d21"
```

## 7.2 Impersonation User

The impersonation User is the user in whose name the API requests are executed. If the user is not already existing the user information will be taken from LDAP.

Header information for API request:

```
impersonation-user: <userNameOfUserToPerformActionsAs>
```

# 8 Backup and Restore Scripts

We recommend backing up the Docker images so that you can fall back on a specific version instead of always using the latest version. Copy the following two scripts into an editor of your choice and save them as .sh files. Add the necessary information for:

```
<PLACEHOLDER_REQIF-MANAGER-CONTAINERNAME>

<PLACEHOLDER_MYSQL-CONTAINERNAME>

<PLACEHOLDER_MSQL-ROOT-PW
```

Give yourself with the command chmod 777 <filename>.sh execution rights for the scripts. Then start the backup script with the command: ./backupReqifMGR.sh (if your filename is "backupReqifMGR.sh") This takes some time depending on the size of the data to be backed up. The backed-up data is in the file backup.tar.gz (in your current folder from where you started the script).

After that you can use the backups for restoring in your new environment.

## 8.1 Backup script

```
#!/bin/bash

export REQIF_MGR_CONTAINER=<PLACEHOLDER_REQIF-MANAGER-CONTAINERNAME>
export MYSQL_CONTAINER=<PLACEHOLDER_MYSQL-CONTAINERNAME>
export MYSQL_ROOT_PASSWORD=<PLACEHOLDER_MSQL-ROOT-PW>

#backup mysql
docker exec -ti $MYSQL_CONTAINER mysqldump -A -C -x --max-allowed-packet=2G
-u root -p$MYSQL_ROOT_PASSWORD --result-file=dump.sql
docker exec -ti $MYSQL_CONTAINER gzip dump.sql
rm dump.sql
docker cp $MYSQL_CONTAINER:dump.sql.gz dump.sql.gz
docker exec -ti $MYSQL_CONTAINER rm dump.sql.gz

#backup reqif files
docker exec -ti $REQIF_MGR_CONTAINER tar -zcvf backup.tar.gz
/app/web/uploads
rm backup.tar.gz
docker cp $REQIF_MGR_CONTAINER:/app/backup.tar.gz backup.tar.gz
docker exec -ti $REQIF_MGR_CONTAINER rm /app/backup.tar.gz
```

## 8.2 Restore script

The restore script must be located in the same folder as your backup.tar.gz file with the data you want to restore. To restore the backed-up data use the following bash script:

```
#!/bin/bash

export REQIF_MGR_CONTAINER=<PLACEHOLDER_REQIF-MANAGER-CONTAINERNAME>
export MYSQL_CONTAINER=<PLACEHOLDER_MYSQL-CONTAINERNAME>
export MYSQL_ROOT_PASSWORD=<PLACEHOLDER_MSQL-ROOT-PW>

#restore mysql
docker cp dump.sql.gz $MYSQL_CONTAINER:dump.sql.gz
docker exec -ti $MYSQL_CONTAINER gunzip dump.sql.gz
docker exec -ti $MYSQL_CONTAINER bash -c 'mysql --max-allowed-packet=2G -u
root -p$MYSQL_ROOT_PASSWORD <dump.sql'
docker exec -ti $MYSQL_CONTAINER rm -f dump.sql

#restore reqif files
docker cp backup.tar.gz $REQIF_MGR_CONTAINER:/app/backup.tar.gz
docker exec -ti $REQIF_MGR_CONTAINER tar -xvf /app/backup.tar.gz -C /
docker exec -ti $REQIF_MGR_CONTAINER rm -f /app/backup.tar.gz
```

# 9 Glossary

| Concept | Meaning |
|---|---|
| Exchange partner | An exchange partner is a company or a department from which you receive exports from a different Requirements Management System, which you then import with the ReqIF Manager. In return, you export data from DNG using the ReqIF Manager, which your exchange partner then imports. |
| Baseline | A baseline is a frozen snapshot of a stream that can no longer be changed. |
| Change set | A change set is the smallest change unit in DNG. The ReqIF Manager automatically creates change sets for carrying out the import. These can either be delivered to the stream or discarded by the user after the import. |
| Customer (role) | In the customer role, you export requirements and only import the values of the harmonization attributes of your harmonization process that you have received from the supplier. |
| Job | When you start an import or export, a job with the included modules is created in the DNG server queue. You can track the progress while the job is being carried out, and then examine the logs; for export jobs, you can then download the modules as a ReqIFz file. |
| Mapping | During mapping, you map the data model from your exchange partner onto your own. This occurs in ReqIF sets with the supplier role when you import modules for the first time. The mapping is subdivided into the sub-stages type mapping, attribute mapping and link mapping. In these sub-stages, you decide which data you would like to import and which types or attributes you would like to use for this. |
| Module (module artifact) | In DNG, artifacts can be consolidated into module artifacts. For simplicity, this manual uses the word "module" from the classic Doors for the module artifacts from DNG. |
| ReqIF/Reqifz (file format) | ReqIF stands for Requirements Interchange Format. This is an XML file format that assists in the exchange of all related metadata between software tools such as the ReqIF Manager and other manufacturers. |

| Concept | Meaning |
|---|---|
| ReqIF set | In a ReqIF set, you compile various module quantities from various DNG projects and components. In the ReqIF set, you also define which modules and attributes an exchange partner may receive and edit. They initiate imports to and exports from the DNG database. |
| (Export/Import) session | Before you can configure an export or import for an exchange partner, you must start a session. During import sessions, all imports are imported into the same change set. At the end of export sessions, you can create a baseline for all the involved DNG components and have the ReqIF Manager “remember” this baseline for all successfully exported modules |
| Stream | A stream can be compared with a branch from software development. The artifacts can be edited in a stream. In a stream, you can create baselines that take a snapshot of the stream. |
| Super set | A super set are modules that are included for all Exchange Partners of a ReqIF Set. Super set modules can be configured centrally for all exchange partners of the ReqIF set |
| Supplier (role) | As a supplier, you import all the data from the customer and map it onto your data model. You only export the attributes defined in the harmonization process. |

# requisis_ReqIF Manager

requisis *Outstanding in Engineering*

**REQUISIS** GmbH
Walter-Benjamin-Platz 8, D-10629 Berlin, Germany
**Telefon +49 30 / 53 65 06 - 700;** Fax +49 30 / 53 65 06 - 300

**REQUISIS** GmbH
Willy-Brandt-Str. 54, D-70173 Stuttgart, Germany
**Telefon +49 711 / 528 529 - 700;** Fax +49 711 / 528 529 - 300

**REQUISIS** GmbH Product Support
DE-Hotline +49 30 536 506 - 666
US-Hotline 1-844-REQUISI, ext. 666

product-support@requisis.com, **https://requisis.com**