NUM-CODEX– Installation guide – EDC System – DIS – Docker

Florian Kohlmayer, 06.07.2021, Version: 0.2

Background

For the local data collection in the NUM-CODEX project an EDC system will be needed. The EDC system can be integrated with the local data integration centers and provides the capability to enter data manually, which can not be imported automatically. The EDC system will export the data in CDISC-ODM format. A transformation pipeline will convert the ODM format into the specified GECCO-FHIR format. As EDC systems REDCap and DIS will be provided. The data dictionaries of both systems will be harmonized. This document describes the installation of the DIS-system preconfigured for the collection of the GECCO dataset. The installation will be based on docker. Basic knowledge of docker is required to execute the installation procedure.

Installation

Pre-requisites

The installation requires an installed docker engine (https://www.docker.com/) with an working docker-compose environment (https://docs.docker.com/compose/). It is recommended to install DIS-docker on a Linux environment.

Clone git repository containing the docker-compose file

The first step is to clone the git repository containing the docker-compose file, the seed database and the configuration files for the DIS-application:

```
git clone https://github.com/kohlmayer/gecco-num.git
```

Please change the working directory to the checkout location and change to the “compose” subdirectory.

Configure the domain names of the different modules

The DIS-application consists of eight modules (m4app, m4mdat, m4pdat, m4idat, m4psns, m4export, m4import, m4search) which can be distributed to different servers, according to the data protection concept (e.g. server1: m4idat, m4import; server2: m4psns, m4search; server3: m4app, m4mdat, m4pdat, m4export). For this simple installation guide, it is assumed that all modules run on the same instance. The configuration of the domain names for the modules is done in the .env file. A template ENV file (base.env) is provided in the repository. The base.env file is located in the repsotory under the *compose* subdirectory. Copy the base.env file to the .env file:

```
cp base.env .env
```

Afterwards edit the .env file with your favorite editor, e.g.:

```
vi .env
```

In the .env file you need to change the domainnames/ip-adresses of the eight modules:

```
hostnameapp=192.168.178.33
hostnameidat=192.168.178.33
hostnamemdat=192.168.178.33
hostnamepsns=192.168.178.33
hostnamepdat=192.168.178.33
hostnameimport=
hostnameexport=192.168.178.33
hostnamesearch=192.168.178.33
```

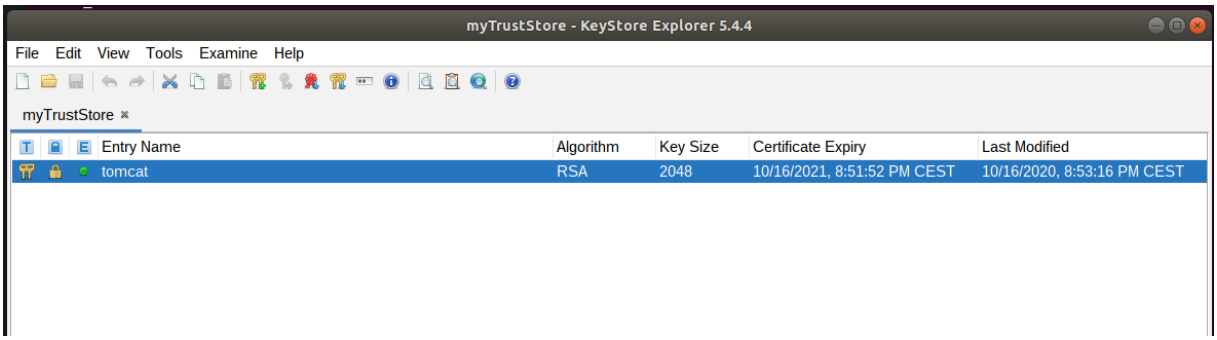
If you leave one module domainname value empty (hostnameimport in the example) the respective module will not be available in the application. If you also like to disable the module from loading in tomat, you need to specify the module in the "DEPLOY_IGNORE" attribute in the .env file. If you use a different port than the standard SSL 443 port, you have to append the port, separated by colon, here as well. If you like to use a different folder than *compose* you can adapt the configuration value "COMPOSE_PROJECT_NAME".

Adapt docker-compose.yml if needed

E.g. in case you would like to run the tomcat on a different port than the standard SSL port 443, you can edit the docker-compose.yml file and change the ports section.

Configure certificates and CAs

The DIS-application requires the communication over TLS. To this end the certificates and CAs must be configured. In *compose/keystore* a JAVA keystore file is located which is be used for the tomcat as well as for the communication of the modules among themselves. The certificate (and corresponding private key) under the alias "tomcat" will be used by the tomcat to secure the HTTPS request to the server. If a common CA is used, this CA can also be imported, to allow the secure communication of the modules. If you get an error like "PKIX path building failed in Java application", the probable reason is the missing certificate hierarchy in the keystore file. To edit the keystore file, either the java command line tool "keytool" can be used or the GUI tool KeyStore Explorer (https://keystore-explorer.org/). Currently the passphrase for the keystore and the certificates must be "*myTrustStore*". This will be made configurable in a future release. The keystore in the repository contains a self signed certificate for the ip address 192.168.178.33.



Resetting the database

In case the initial seeding of the database does not work or you would like to reset the databases to their initial state you could execute:

```
docker exec compose-mysql bash /docker-entrypoint-initdb.d/importDBs.sh
```

This command will execute the importDBs.sh script again, which drops the database (CAUTION) and imports the seed database again.

Testing the installation

To start the application, change to the compose folder and execute:

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

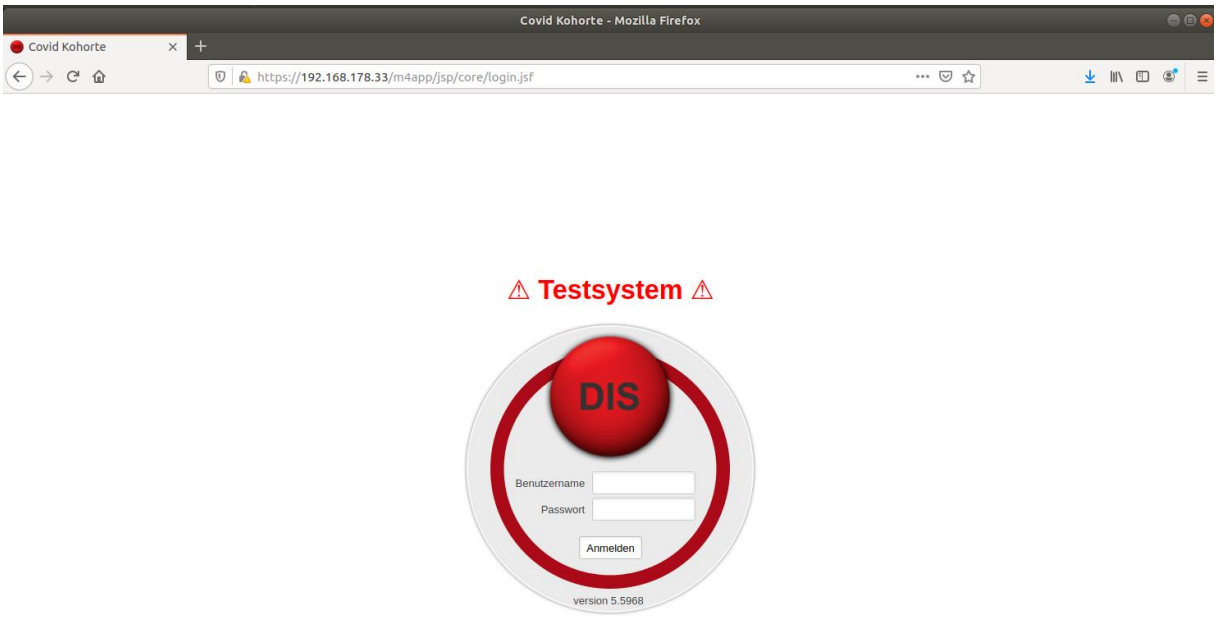
To see the logs you can execute:

```
docker-compose logs -f
```

If no errors occur, the application is available under the domainname/ip address configured under the “hostnameapp” configuration parameter above. In this example the app will be available unter:

https://192.168.178.33/m4app

You should see the login screen:



The default username/password is: *gecco/gecco*

The default username/password for the admin is: *admin/admin*

If you have any problems please contact florian.kohlmayer@tum.de.