ADIT Engineering Projekt Infrastructure

Documentations to allow the setup of the infrastructure.

Please note that this documentation only covers the setup of a production instance, although the repository also contains configuration files for `-develop` instances. These can be setup just like the production system (except having `-develop` in names everywhere).

This documentation assumes that your current PWD ist the repository root, e.g.:

```
git clone https://github.com/fabianhauser/engineering-projekt-infrastructure.git /opt/engineering-projekt-infrastructure
cd /opt/engineering-projekt-infrastructure
```

Adit-Application

This section covers the installation of a full adit stack.

Prerequisities

- Ubuntu Linux (https://www.ubuntu.com/) (currently 16.04.2 LTS)
- Systemd (https://www.freedesktop.org/wiki/Software/systemd/) Version >=229
- Docker (https://www.docker.com/) Version >= 17.05
- git (https://www.git-scm.org/)
- Recommended: netfilter/iptables (https://netfilter.org/) based firewall

The installation of these dependencies is not in the scope of this document.

Installation engineering-projekt-client

This is a nginx container, hosting the adit frontend/client application.

```
docker pull fabianhauser/engineering-projekt-client
systemctl enable `pwd`/services/engineering-projekt-client.service
systemctl start engineering-projekt-client.service
```

Rollator

The Rollator utility allows our Travis-CI instance to update the server application.

Please follow the base installation instructions on the project website https://github.com/fabianhauser/rollator/ (https://github.com/fabianhauser/rollator/)

```
# This only works after the installation of Rollator
systemctl link `pwd`/engineering-projekt-client/engineering-projekt-client-rollator
.service
```

Installation engineering-projekt-server

The `engineering-projekt-server` container is a java/jetty environment, containing the adit backend/server application. This setup also includes a postgres database container.

```
cp services/engineering-projekt-server/engineering-projekt-server.env /etc/
# Replace ###POSTGRES_PASSWORD### with a random password string

docker pull fabianhauser/engineering-projekt-server
systemctl enable `pwd`/services/engineering-projekt-server-postgres.service
systemctl enable `pwd`/services/engineering-projekt-server.service
systemctl start engineering-projekt-server.service
```

Rollator

The Rollator utility allows our Travis-CI instance to update the server application.

Please follow the base installation instructions on the project website https://github.com/fabianhauser/rollator/ (https://github.com/fabianhauser/rollator/)

```
# This only works after the installation of Rollator
systemctl link `pwd`/engineering-projekt-server/engineering-projekt-server-rollator
.service
```

Installation nginx

The nginx container functions as load balancer and TLS termination. It is not strictly required to run the application (although neccessary and sensible with this deployment)

```
docker pull fabianhauser/nginx-dehydrated
systemctl enable `pwd`/services/nginx/nginx.service
systemctl start nginx.service
# Note that the start command may fail, if no certificates are available. See secti
on Dehydrated
```

Dehydrated

Dehydrated is a Let's encrypt TLS certification ACME client.

```
systemctl enable `pwd`/services/nginx/status-email-root@.service
systemctl enable `pwd`/services/nginx/nginx-dehydrated.service
systemctl enable `pwd`/services/nginx/nginx-dehydrated.timer

# Create certificates with this oneshot-execution:
```

```
/usr/bin/docker exec -ti nginx dehydrated --register --accept-terms
systemctl start nginx-dehydrated.service
```

Building the application

This section covers how to build the whole application and correspondant containers locally on your machine

Prerequisities

- Docker (https://www.docker.com/) Version >= 1.12
- GNU Make (https://www.gnu.org/software/make/) and Bash (https://www.gnu.org/software/bash/)
- git (https://www.git-scm.org/)

Client: `engineering-projekt-client`

```
git clone https://github.com/fabianhauser/engineering-projekt-client.git engineerin
g-projekt-client
cd engineering-projekt-client

# Build angular building docker container
make build-container-testing

# Build application and docker container
make build

# For a release, the docker container could be uploaded now.
```

Server: `engineering-projekt-server`

```
git clone https://github.com/fabianhauser/engineering-projekt-server.git engineerin
g-projekt-server
cd engineering-projekt-server

# Build java building docker container
make build-container-testing

# Build application and docker container
make postgres-start
make build
make postgres-stop

# For a release, the docker container could be uploaded now.
```

CI: Continous Integration

Travis

The Travis-configuration is contained in the respective repository. Note that the Travisfiles require the definition of following (private) ENV variables:

- `DOCKER_USERNAME`: https://hub.docker.com/ (https://hub.docker.com/) username
- `DOCKER_PASSWORD`: https://hub.docker.com/ (https://hub.docker.com/) password
- `SONAR_KEY`: API-KEY for the sonar update
- `SSH_KEY`: SSH private key for server deployment (see Rollator setup)

Sonarqube

This is the setup of our sonarqube instance used to push testing results from our CI.

```
cp sonarqube/sonarqube.env /etc/
# Replace ###POSTGRES_PASSWORD### with a random password string
systemctl enable `pwd`/services/sonarqube-postgres.service
systemctl start sonarqube-postgres.service

systemctl enable `pwd`/services/sonarqube.service
systemctl start sonarqube.service
```