Installation and deployment

Production Optimiser

Politecnico di Milano, Information Engineering School, Italy
University of Zagreb, Faculty of Electrical Engineering and Computing, Croatia
Distributed Software Development

17th of January 2024

Installation – Backend

1. Using Docker:

   Position current directory:

```
cd backend/production-optimiser-api
```

   Run docker compose:

```
docker compose -f docker-compose.integration.yml up -d
```

This will start backend with database + API + services all configured

2. Manually:

   Manually setup postgres database (16+) with params:

```
POSTGRES_USER: production_optimiser
POSTGRES_PASSWORD: production_optimiser
POSTGRES_DB: production_optimiser
```

   Position current directory:

```
cd backend/production-optimiser-api
```

   Run maven task:

```
maven spring-boot:run
```

   Run python services using unicorn (optional):

```
unicorn main:app --host 0.0.0.0 --port 9083
```

In both cases spring JPA will take care or database tables creating and initial data filling, for different behavior configure application.properties in backend/production-optimiser-api/src/main/resources. API will run on localhost:8080

Required python dependencies are listed in requirements.txt folder inside each service.

Installation – Frontend

1. Manually:

   Position current directory:

```
cd frontend
```

   Install required dependencies:

```
npm install
```

   Run the application:

```
npm run dev
```

By default application will run at localhost:5173 and backend url will be localhost:8080. If you would like to change backend url it can be done by configuring environment variable.

```
VITE_API_URL = „backend-url.com“
```

Deployment – Backend

Deployment can be done in any docker environment or virtual machine. Once the docker image or jar is built.

1. Docker environment

   Position current directory:

```
cd backend/production-optimiser-api
```

   Build application jar:

```
maven package -DskipTests
```

   Build docker image:

```
docker build -t production-optimiser-api:1.0.0 -f Dockerfile /target
```

   Run docker image as docker image or as Kubernetes deployment.

In this setup you need to manually configure database and when running the docker image configure environment variables for database connection parameters.

2. Virtual machine

   Position current directory:

```
cd backend/production-optimiser-api
```

   Build application jar:

```
maven package -DskipTests
```

   After jar file is built in /target folder transfer it to virtual machine and run it:

```
java -jar production-optimiser-api-1.0.0.jar
```

Deployment – Frontend

Frontend can be served as a static page or as server. Many online services offer static page serving including GitHub and GitLab.

More about static page deployment can be found here:

https://github.com/gitname/react-gh-pages

https://docs.gitlab.com/ee/user/project/pages/

Server deployment can be done with Nginx (https://www.nginx.com/).

Build optimized application:

```
npm run build
```

Copy generated /build or /dist folder to the server and point already configured Nginx proxy to the copied folder.