E4 - Introduction to Microservices

It is a students' info system, built of 3 microservices:
- students + H2 DB
- assignments + H2 DB
- reports + mongoDB in a separate container

1. Start Docker

2. Recycle P5-RESTful-Level-3 from last time into a `student` microservice with a H2 database

3. Create an `assignment` microservice with a H2 database

Another microservice, which uses MongoDB from a Docker container

4. Create new Spring Boot project with Web and MongoDB dependency

5. In the project folder (the terminal), create an empty Mongo Docker container
```
docker run -P -d --name mongodb mongo
```

6. Check if mongo is up and running
```
docker exec -it mongodb sh
```

7. In the shell, to get info and confirmation, type
```
mongo
```

8. Try to create a database and a collection in it
```
use mongoreport
db.createCollection ('report')
```

9. Finish the test with
```
Ctrl/C
exit
```

10. Check which port has been given to `mongodb` in the container
```
docker ps -a
```

11. Try to access it from out of the container on the localhost port (32768?).

You may decide to download a GUI for MongoDB, e.g. Compass from
https://www.mongodb.com/try/download/compass

12. Now create the microservice `reports`, which will store data in a new mongo database

Test the service creating records in the database from Postman and accessing them from HAL (the browser). Remember the Content-Type!

Note: for auto-incrementing the MongoDB records id, see https://www.baeldung.com/spring-boot-mongodb-auto-generated-field

Deploying the Microservices

1. For each microservice, create an executable (`jar`) file as an *artefact* stored in a `/out` subfolder. Use the *Project Structure* menu to arrange the location of the folder.

2.. For each microservice create a `Dockerfile` as a text file in the root directory. Use CTRL+space to see all Dockerfile commands.

Here are some examples of proper Dockerfile content:

- for Student

```
FROM java:8
VOLUME /out
ADD out/ms-student.jar app1.jar
EXPOSE 8080
RUN bash -c 'touch /app1.jar'
ENTRYPOINT ["java","-
Djava.security.egd=file:/dev/./urandom","-jar","/app1.jar"]
```

- for Assignment

```
FROM java:8
VOLUME /out
ADD out/ms-assignment.jar app2.jar
EXPOSE 8080
RUN bash -c 'touch /app2.jar'
ENTRYPOINT ["java","-
Djava.security.egd=file:/dev/./urandom","-jar","/app2.jar"]
```

- for Report

```
FROM java:8
VOLUME /out
ADD out/msa-report.jar app3.jar
EXPOSE 8080
RUN bash -c 'touch /app3.jar'
ENTRYPOINT ["java","-
Dspring.data.mongodb.uri=mongodb://mongo/reports", "-
Djava.security.egd=file:/dev/./urandom","-jar","/app3.jar"]
```

3. Create a Docker image of each microservice

```
docker build -t msdemo/students .
docker build -t msdemo/assignments .
docker build -t msdemo/reports .
```

4. Check with

```
docker images
```

5. Build containers from the images

```
docker run -p 6060:8080  -d --name students msdemo/students
docker run -p 7070:8080 -d --name assignments msdemo/assignments
```

For Report, add a link to the mongodb container

```
docker run -p 8080:8080 -d --name reports --link mongodb
msdemo/reports
```

Integrate the Services

1. Create `docker-compose.yml`

```
version: "3.7"
services:
  ...
volumes:
  ...
```

```
                networks:
                  ...

version: "3.7"
services:
      reports:
        build: reports
        ports:
          - "8080:8080"
        links:
          - mongodb

      assignments:
          build: assignments
          ports:
            - "7070:8080"
          links:
            - mongodb

      students:
          build: students
          ports:
            - "6060:8080"
          links:
            - mongodb

      mongodb:
        image: mongo
```

Before we continue, we'll check our build-file for syntax-errors:

```
docker-compose config
```

2. Run

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

3. Run to test

```
docker-compose ps
```

4. Scale a microservice

```
docker-compose scale students=3
```

5. To stop the containers, remove them from *Docker* and remove the connected *networks* from it. To do this, we can use the opposite command:

```
docker-compose down
```