Web developpement : Homework 4

Esada Licina – BICS

The first steps that I did was to install knex, express and curl (so that we can use crud operations) in ubuntu terminal:

- Sudo npm install -g knex
- Sudo apt install node-express-generator
- Sudo apt-get install curl

Then I imported the given file (db.sql) in mysql into the created database WD_HW4.

- Mysql -u root -p WD_HW4 < db.sql

Then I created a index.js file:

- Touch index.js
- Open: nano index.js
- Run: node index.js

Then it was time to start with the main code in my index.js file. It's needed to declare (const) some variables at first so that we can use express and knex. The user, password, and database from knex are personal. To set the password use this code lines in mysql (where red is, there you need to enter your personal password):

- sudo mysql -u root
- ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
- flush privileges;

```
const express = require("express")
const app = express()
const knex = require("knex")({
        client: "mysql",
        connection: {
                host : "127.0.0.1",
                port : 3306,
                user : "root",
                password : "password",
                database : "WD_HW4"}})

app.use(express.json())
```

1)

```
app.post("/test1", (request, response) => {

        knex("city").insert(request.body).then(city => {response.send("Successfully created")}).
                                                        catch(error => {response.send("Creation failed")})
})
```

In the first exercice we needed to take the post request to insert new data in the database.

Then we have two parameters (request and response) which are there to receive and send data.

Request.body : It takes the information that is given from the user (here it is the new city .

Then : This is the main operation so that it is even possible to send something.

There is also a message code which will be output when the creation success or failed.

Before running the code, it is important to go in mysql and to put an auto increment in the ID column that every time the id goes+1 higher when we add new data :

ALTER TABLE CITY MODIFY COLUMN ID int(11) UNIQUE AUTO_INCREMENT PRIMARY_KEY

Run code : curl -X POST -H "Content-Type: application/json" -d '{"Name":"Luxemburg","CountryCode":"LUX","District":"Bissen","Population":12376}' http://localhost:9000/test1

2)

```
app.get("/test2", (request, response) => {

        knex("city").select("*").where("Name", "=" ,request.body["Name"]).then(city => {response.json(city)})

})
```

This code is used to get the row with the given city name.
We choose at first the table city of the database and take every column out, which is defined by with the "*" operation. With the where operation, we specify from which city the information will be taken, where the name of the city is the same as the input name.

Run code: curl -X GET -H "Content-Type: application/json" -d '{"Name":"Luxemburg"}' http://localhost:9000/test2

3)

```
app.put("/test3", (request, response) => {

        knex("city").select("*").where("Name", "=",request.body["Name"]).update({"Population":request.body["Population"]}).
                                                                                then(city => {response.json(city)})
})
```

This part will update the population number of a chosen city. The code is similar as above just that we added the update operation to change the value.

Run code: curl -X PUT http://localhost:9000/test2 -H "Content-Type: application/json" -d '{"Name":"Luxemburg", "Population":22222}' "Content-Type: application/json"

4)

```
app.delete("/test4", (request, response) => {

        knex("city").where("Name", "=",request.body["Name"]).delete("*").then(city => {response.send(true)}).
                                                                    catch(error => {response.send(false)})

})
```

The last code part will delete a row with by the input name. Where we look at the request body which is the input of the user and delete the row with the same city name. Here we get a Boolean value back, if it was successful deleted than it outputs "true", if not then "false".

Run code: curl -X DELETE -H "Content-Type: application/json" -d '{"Name":"Luxemburg"}' http://localhost:9000/test4