GIS – Tutorial 3 – Frontend Development

University of Konstanz | Due Date: 25.11.2022 08:00

1 Submission Instructions

Please provide **one** well-formatted **PDF** file with all your submissions on schedule (i.e. before the deadline). Otherwise your submission will not be graded!

2 Background

TypeScript is an open-source language which builds on JavaScript, one of the world's most used tools, by adding static type definitions.
For more information visit: `https://www.typescriptlang.org/`
For a tutorial visit: `https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html`

Angular is an application design framework and development platform for creating efficient and sophisticated single-page apps.
For more information visit: `https://angular.io/docs`
For a tutorial visit: `https://angular.io/tutorial`

2.1 Installation

This tutorial assumes that Angular will be run inside a Docker container. If you want to manually install it please visit the following URLs:

- Typescript: `https://www.typescriptlang.org/download`
- Angular: `https://angular.io/guide/setup-local`

2.2 Useful commands for this assignment

During this tutorial we will often start & stop Docker containers, the following commands will be often used or might be helpful:

- *docker run [OPTIONS] IMAGE*
  `https://docs.docker.com/engine/reference/run/`
  Runs the *IMAGE* Docker container with the specified *[OPTIONS]*

- *docker ps*
  `https://docs.docker.com/engine/reference/commandline/ps/`
  Shows a list of currently running Docker containers

- *docker stop [OPTIONS] CONTAINER*
  `https://docs.docker.com/engine/reference/run/`
  Stops the *[CONTAINER]*. You can identify the container id or name using *docker ps*.

- *docker system prune -a*
  `https://docs.docker.com/engine/reference/commandline/system_prune/` Remove all unused containers, networks, images, etc.

Later in the tutorial we will also use docker-compose, here the following commands are the most used:

- *docker-compose up*
  `https://docs.docker.com/compose/reference/up/`
  Builds, (re)creates, starts, and attaches to containers for a service.

- *docker-compose down*
  `https://docs.docker.com/compose/reference/down/`
  Stops containers and removes containers, networks, volumes, and images created by up.

3 Homework

Task 1: Setup **0 Points**

This tutorial you will be provided with a full stack containing a database, backend as well as the frontend.

1. Download the GIS.zip file from assignment 3 in ILIAS and unzip it. The directory structure should look like this:

```
GIS
├── backend
│   ├── code
│   │   └── ...
│   └── Dockerfile
├── database
│   ├── data
│   │   └── ...
│   └── Dockerfile
├── frontend
│   ├── gis
│   │   └── ...
│   └── Dockerfile
└── docker-compose.yml
```

2. Run the project using the command *docker-compose up*

3. Visit the webpage `localhost:4200` to check if the project is running correctly.

Submission: State the tag, image id and size of your modified Angular container. You can check this information with the command *docker images*.

Task 2: Bringing it all together 15 Points

Your task is to extend the given framework to load amenities of a user-specified type dynamically and show the location, as well as additional information, as markers in the frontend. For that, please work on the following tasks:

Database

1. Write a SQL query that, given an amenity type, returns the name, latitude, and longitude of all amenities of the given type in the city of Konstanz.

Backend

1. Create a new endpoint in the backend, that takes an amenity type as a POST request in JSON format (see: `https://flask.palletsprojects.com/en/1.1.x/api/#flask.Request.json` on how to get this information) and uses the query from 1.) to return all amenities of that type in the city of Konstanz.

Frontend

1. Add a text input field and a button to the Settings component, in which the user can enter an amenity type and submit the query.
2. Propagate the button click and the entered amenity type to the App compoment
3. Write a new method in the DataService to request the information about the specified amenities from the backend.
4. Push the retrieved data into the Map component and add a marker for each requested amenity.

Submission: Screenshot of your website after you added a marker for a specific amenity with at least 5 entries (e.g. *pharmacy, bank, drinking_water*).