How To Run?

* Frontend – React – There is a docker compose file at the root. Simply run docker-compose up to start and open localhost:80
* Frontend – Angular – The angular solution also has a docker compose file. Run docker-compose up and open localhost:80
* Python – this assumes that you do not have postgres. Run docker-compose up and open your sql editor. Username and password is admin and admin and database is postgres. There is an init.sql in ./data which initializes the database. If you already have postgres, then run ./data/init.sql to create the schema. Then, change the connection settings in ./src/database/database.py to connect to your personal instance.

Solution files

1. Database diagram – at the root of the python code there is a .png image called er\_diagram.png which contains a visual representation. It was made using dbdiagram.io. The link to the project is - <https://dbdiagram.io/d/64a0bb1402bd1c4a5e5a78be> . This is publicly available.
2. SQL – at the root of the python code, solution.sql has solutions to the 3 tasks.
3. Python – After creating your DB, the main.py file at the root has some commented code. You can uncomment them to test the functions. /data/init.sql has the sql inserts and create necessary for setting up the schema.
4. JavaScript – Run docker-compose up to start the server. That should open localhost:80 and the UI should render. I initially coded it in angular but challenged myself by trying to learn react and porting the angular code to react.