Build-Documentation.md

Table of Contents

Production Build

Preparations

- Docker Installation
- Environment for .sh scripts

How-To

1. Clone git repository
2. Navigate to the src-folder
3. Copy the `.env.template` file to `.env.local`
4. Edit the settings in `.env.local` if necessary.

The frontend should now be reachable on `http://localhost:3000`. After that you should be able to register, login and use the software. If you need further informations, please look at the build process video (https://github.com/amosproj/amos2022ws05-shared-desk-mgmt/blob/develop/Documentation/BuildProcessVideo.mp4) or contact the developers.

5. OPTIONAL: If you want to fill the database with our dummy-data you can use the following command:

```
docker exec -i deskstar_postgres psql -U postgres deskstar < ./deskstar-db/dummy-data.sql
```

The dummy data already contain some companies you can use and also some test users. You can find further information in the user documentation (https://github.com/amosproj/amos2022ws05-shared-desk-mgmt/wiki/User-Documentation).

This explains how to configure the local devcontainer for development.

Local Dev Setup (Containered)

Requirements

- Docker Installation
- Visual Studio Code
- Dev Container Extension (https://marketplace.visualstudio.com/items?itemName=ms-vscode-remote.remote-containers)

How-To

First you need to checkout the repository.

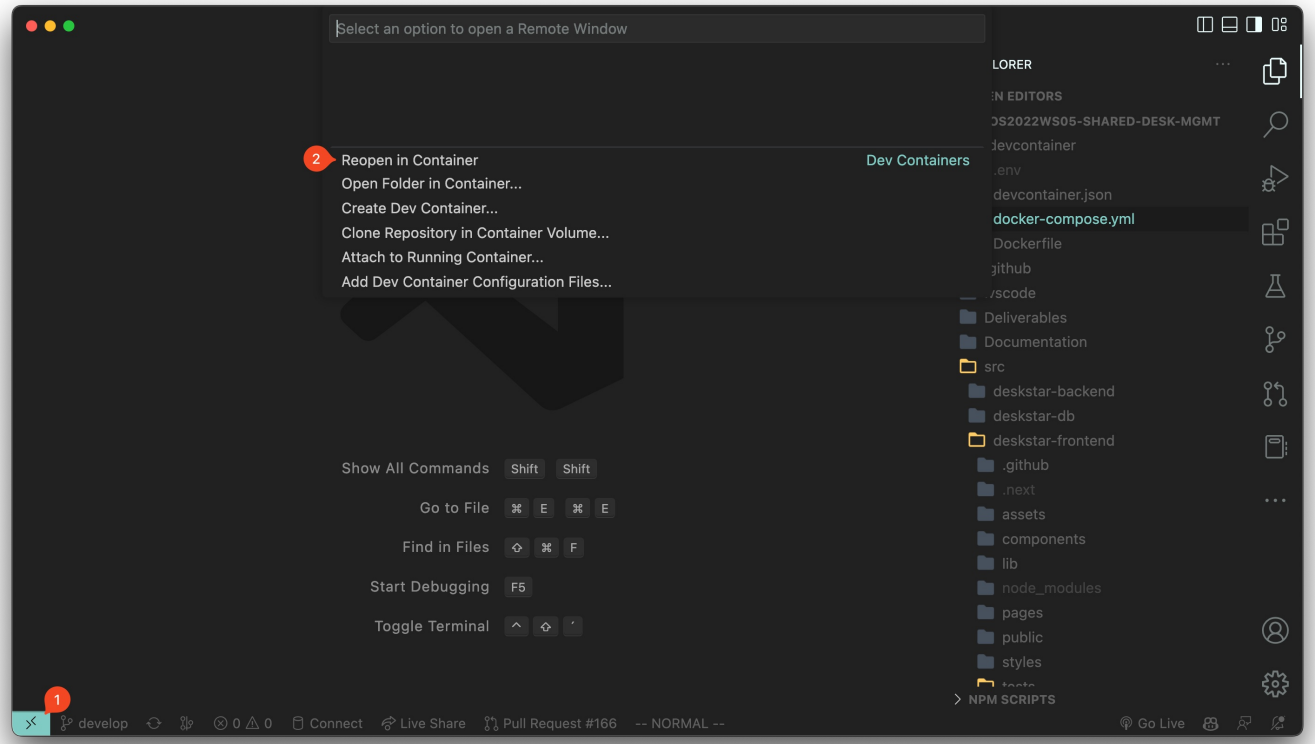
Then open it in Visual Studio Code.

In the folder `.devcontainer` you need to create a file with the name `.env` and the following content:

```
EMAIL__PSW=Your email password
```

For additional smtp settings take a look into the `docker-compose.yml` inside the `.devcontainer` folder.

Click on the Icon on the bottom left. (1) And then klick on "Reopen in Container" (2)



Now the Dev Container should build and Visual Studio Code should open again inside the container. Now you can start the frontend by navigating into `src/deskstar-frontend` using the Visual Studio Code Terminal and running

```
yarn
```

to install the dependencies and

```
yarn dev
```

to start the frontend dev server on port 3000.

For the backend you need to navigate into `src/deskstar-backend/Deskstar` and run `./run.sh` in a separate terminal.

Local Database Setup

The database schema is completely defined in the backend. The entities folder inside the backend defines all Entities and Relations between them. Using dotnet-ef we are now able to update the database regarding the schema in the backend. The backend is the single point of truth.

After adding, deleting or modifying an entity inside the entities folder, a new Migrations must be created. For that the following command needs to be executed:

```
dotnet ef migrations add [NAME OF THE MIGRATIONS]
```

The name of the migrations should be a short label for the changes made. For example: AddUserRoles

Now a new migrations is added to the Migrations folder. Every migrations needs to be commited to the git, so everyone else can also update their database.

> Its important to not delete the previous migrations! A migration is always depending on their previous migration. So you need to commit all migrations and not change one of them.

To update the database now, we need to execute the following command:

```
dotnet ef database update
```

That updates the local database by comparing the current state the database is in with the migrations inside the Migrations folder.