Todo App Documentation

The app supports 2 features in order to run the database. You can either use Docker or not.

Install and run with Docker:

First time installation:

- npm install
- npm run migration

//run migration scripts for the database

Run commands:

- docker-compose up
- npm run server
- npm start

Install and run without Docker:

Credentials of the database must be changed in the file /src/conif/config.json

First time installation:

- npm install
- npm run migration

//run migration scripts for the database

Run commands:

- npm run server
- npm start

Technologies used:

For the implementation of the app I used NodeJs , Webpack , Babel , React , Express , Sequelize and Postgres.

For the user authentication I used PassportJs.

App flow:

On the home page the user can either Sign In with his username and password or he can create a new account.

When a new account is created the user can then Sign In through the form to the app.

Then he gets redericted to his TodoList.

Tasks can be added , edited , deleted or reordered.

The user can Log Out any time from the app with the Log out button.

Models and their relationships:

Two models have been used for the implementation of the app and their definitions can be found in the folder /model.

1.

Posts
Id
Text
Post_date
User_id
order

Id: primary key , identification number of every todo item

Text : description of the todo item

Post_date: date of creation

User_id: id of the user who made the todo item (references Users(id))

Order: order of the todo item in the list , when the items are reordered with the drag and drop feature their new order gets updated.

2.

Users
Id
Username
Password

Id: primary key , identification number of every user

Username: username of user , unique , no users can have the same username

Password: password of user

For the storage of passwords bcrypt has been used.

Application's structure and architecture:

/models: Defenition of models

/passport: Strategy of passport and functions for serializing and deserializing the user.

Simple authentication using username and password has been used.

/src:

/components: Components rendered by app.js

/config: configuration for the connection of the database

/db/migrate: migration scripts for the database

/server.js:

-server configuration

-connection with the database

- post and get requests to the database