React Task 2 - React router

This is the second part of the application.
In this assignment, you should continue to work with your project with a slight modification.
This module is designed to teach how to work with routing in React.

Prerequisite:

Firstly, clone and run repo with a server:

1. Clone repo with a server (link)
2. Go to server project folder, install dependencies and run server:

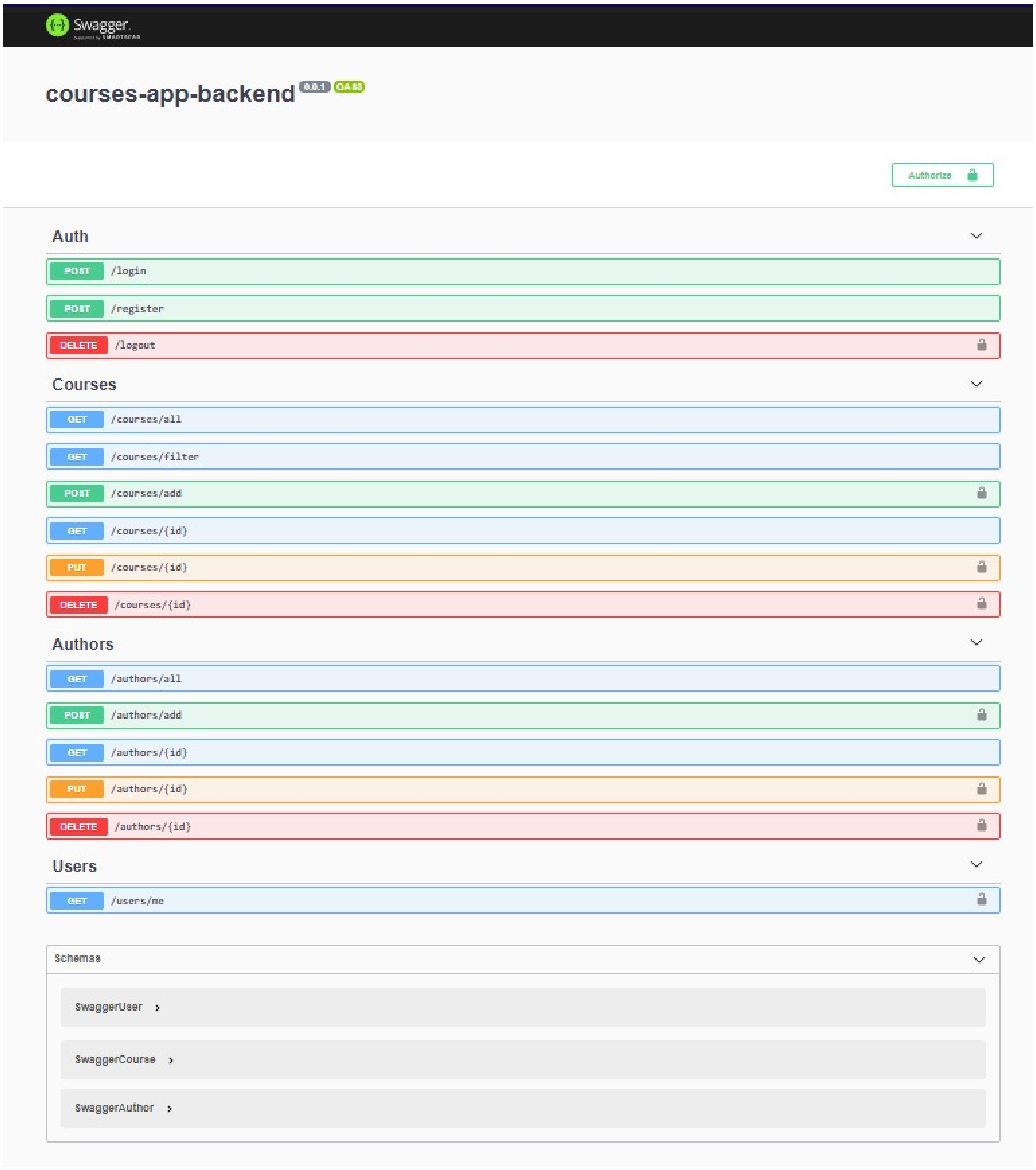
```
npm install

npm run start
```

3. Now you have a backend for your application.

   Go to the url: http://localhost:3000/api

You should see page with all APIs



Swagger

courses-app-backend 0.0.1 OAS3

Authorize

Auth

POST /login

POST /register

DELETE /logout

Courses

GET /courses/all

GET /courses/filter

POST /courses/add

GET /courses/{id}

PUT /courses/{id}

DELETE /courses/{id}

Authors

GET /authors/all

POST /authors/add

GET /authors/{id}

PUT /authors/{id}

DELETE /authors/{id}

Users

GET /users/me

Schemas

SwaggerUser

SwaggerCourse

SwaggerAuthor

> ❗
> There are all APIs for the app, BUT **you should use only those specified in the task**.

Secondly, you need to install the `react-router-dom` module in your project using npm:

1. Go to your project directory
2. Install `react-router-dom`

```
npm i react-router-dom --save
```

3. Run your project

```
npm run start
```

> If you have already run server on localhost:3000 you will see message
>
> ```
> √ Something is already running on port 3000.
> ```
>
> when you try to run the React project.
>
> You just need to enter 'yes' and React project will be run on localhost:3001

Project structure requirements

1. Create new folders and `jsx` files for each component:

```
src
|-- components
|    |-- Login
|    |     |__ Login.jsx
|    |
|    |-- Registration
|    |    |__ Registration.jsx
|    |
|    |-- CourseInfo
|    |    |__ CourseInfo.jsx
|    |
|    |__ ...
|__ ...
```

2. For sending requests to API you should use fetch or axios.

3. APIs from SWAGGER for Module 2:

- `/login` [POST]
- `/register` [POST]

Criteria (25 points max)

- [2 points] - Use `react-router-dom` hooks: `useParams`, `useHistory` etc.

Courses Component

- [1 point] - Component **Courses** should be open by route `/courses`

CourseInfo Component

- [3 points] - Create component **CourseInfo**. *(**Course info** in COMPONENTS)*
- [1 point] - Show information about course. Use route `/courses/:courseId` (`courseId` - id of current course)
- [1 point] - To find out which course info you should render on `CourseInfo` page, you should use id of course from path-parameters.

CourseCard Component

- [1 point] - Add functionality for button `Show course` on `CourseCard` component.
When user clicks `Show course` button he should be navigated to the page with course information.

CreateCourse Component

- [1 point] - Open **CreateCourse** component by route `/courses/add`.

Registration Component

- [2 points] - Create component **Registration**:
*(**Registration** in COMPONENTS)*
- [1 point] - Registration form should appear after clicking on the link *Registration* on the `Login` form.
- [1 point] - Registration form should appear by route `/registration`.
- [2 points] - Registration should have an auth functionality.
User enters email, name and password, presses the `Registration` button then application send request to API.
See `/register` endpoint in API Swagger.
After success registration application navigates you to `Login` page.

Login Component

- [2 points] - Create component **Login**.
*(**Login** in COMPONENTS)*.
- [1 point] - Login should be showed after first open application by route `/login`.
- [1 point] - Login form should appear after clicking on the link *Login* on the `Registration` form
- [2 points] - Login should have an auth functionality.
When you entered email and password application sends request to API.
See `/login` endpoint in API Swagger.
After success login application navigates to `Courses` page.
- [2 points] - Save token from API after login.
Add functionality that check if token in `localStorage`. If token is in the `localStorage` app automatically navigates to `/courses` route.

Header Component

- [1 point] - Add logout functionality.
  When user click `Logout` button in `Header` component, token should be removed from `localStorage` and user is navigated to `Login` page.

Extra Task

- Add data type checking for props to all components using PropTypes or TypeScript

> Please find detailed description of components and functionality in the file COMPONENTS