Project 4DWR

Project Description

In this project, you are tasked with developing the frontend for a LinkedIn clone. This frontend application should interact seamlessly with the backend services designed using a microservice architecture. Your primary technology stack will include ReactJS, along with additional libraries and tools like Redux for state management, React Router for navigation, and Axios for HTTP requests. This project aims to deliver a dynamic and responsive web application that employs modern development practices and design principles covered in the 4AMS course.

The backend will be provided using project files and a docker compose file which allows to run the project easily.

Backend Github repository: https://github.com/bn3t/4ams-linkedin-clone.

Key Requirements:

1. **Page Components (8 points):** Develop the following main components:
   - **Home Page**: Displays a list of posts with Author Name, Post Title, and a snippet of the content. It should also allow navigation to the post's detailed page and the user's profile.
   - **Post Page**: Show the details of a single post, including full content, author details, and comments. Ensure navigation to the author's profile is possible.
   - **New Post Page**: Allow the user to create a new Post.
   - **User Profile Page**: Displays user information, including education, skills, experience, and connections. Offer functionality to edit profile details.
2. **State Management (4 points):** Utilize Redux, React Router Loaders or React Query for managing application state efficiently.
3. **Routing (2 points):** Implement React Router for navigating between different components/pages in your application. It should be possible to bookmark a link to a Post or to a User profile page.

4. **API Integration (2 point):** Use Axios or the Fetch API for making HTTP requests to your backend services. Implement CRUD operations for users, posts, profiles, and comments. Ensure proper error handling and feedback to the user when loading data.
5. **UI/UX Design (1 point):** Use advanced libraries or framework to improve the user experience. Advice is to use tailwindcss, ant-design (or another similar library). It should be easy to use the application and navigate between the different features. Use the principles presented in the course like module css to modularize your css.
6. **Documentation (1 point)**: Document your code extensively and provide a README that includes setup instructions, a description of the project structure.
7. **Error Handling (2 point):** Implement user-friendly error messages and input validations to improve the user experience.

Bonus Points:

- **Dockerization (2 points):** Provide a Dockerfile for the frontend application and integrate it with the backend docker compose setup to allow easy setup and deployment.

Deliverables:

- Complete source code, buildable and runnable on Github. Create a tag named **v1** with the version that should be evaluated, this tag must be done before the deadline of the project. When the project is done in a group of 2 the history of commits should be indicative of the work everybody contributing to the project.
- Dockerfile and integration with the backend docker compose setup (if attempting bonus points). This means that your frontend must be integrated in the same docker compose and everything should run together.
- Comprehensive documentation, including deployment instructions, application documentation and integration points with the backend services.
- A report detailing your approach, challenges faced, and how you addressed them.

- When documentation is inside the source code (github), it should be in the Markdown format (.md). For documentation provided outside of the source code, it should be in the PDF format.
- The project should be runnable using maximum 2 commands: 1 for the backend, 1 for the frontend. Ideally, it should be one command with a docker compose integrating backend and frontend (if attempting the bonus points).

This project is designed to test your abilities in creating complex, interactive web applications using ReactJS and integrating them with a microservices architecture. It will challenge you but also help solidify your understanding of modern web development practices.