RMG

Senior Front-End Developer – Technical Assignment

Build a sample single-page application (SPA) that includes user authentication, product management, and invoice creation features—all without a real backend. Instead, you should use a fake API (or mock server) to simulate data interactions.

Application Requirements

1. **Core Feature**
   - **Login Page**
     - Create a clean, modern login interface.
     - Simulate user authentication. (It is acceptable to use a hardcoded user/password pair.)
     - Redirect successfully authenticated users to the Home page.
   - **Home Page**
     - A dashboard or landing area that provides navigation to product management and invoice creation.
     - Include clear calls-to-action and an intuitive layout.
   - **Product Management**
     - **Create Product:** Allow users to input product details (e.g., name, description, price, etc.) and add a new product.
     - **List Products:** Display a list of products fetched from a fake API.
     - **Edit/Delete Product:** Extra points for additional CRUD operations.
2. **Technical Specifications**
   - **Framework/Libraries:** Angular, Angular Material.
   - **Styling:** Ensure the application has a polished and professional design. You may use CSS frameworks (e.g., Tailwind, Bootstrap, Material UI) or write custom styles.
   - **Data Handling:** Simulate API calls using a fake API tool (e.g., json-server, MirageJS, or similar). The application should mimic real-world API interactions.
   - **Responsiveness:** The UI should be responsive and work well on both desktop and mobile devices.
   - **Code Quality:** Follow best practices in coding, including clean code, modular design, and proper documentation/comments.

3. **Bonus Points**
    - **Performance:** Optimize for performance and best practices (e.g., lazy loading, code splitting).
    - **Deliverables**
    - **Source Code:** Provide a link to a public Git repository (e.g., GitHub, GitLab) containing your source code.
    - **ReadMe File:** Include instructions on how to set up and run the application locally. Describe any architectural decisions, libraries used, and any additional features implemented.
    - **Demo:** (Optional) If possible, deploy the application (e.g., on Vercel, Netlify) and provide a demo link.

Evaluation Criteria

- **Code Quality:** Readability, maintainability, and adherence to best practices.
- **Technical Competence:** Demonstration of a strong understanding of modern front-end development techniques and technologies.
- **Design & UX:** The aesthetic quality and usability of the application.
- **Problem-Solving:** Effective use of a fake API to simulate backend functionality and handling of asynchronous data.
- **Documentation:** Clarity and thoroughness of your README and inline code comments.

Timeline

- **Assignment Duration:** You have up to 5 days from the time you receive this document to complete the assignment.

Submission Instructions

1. Fork the repository or create your own repository with your code.
2. Share the link to your repository (and the demo link if available).
3. Include any notes or instructions in your README file.