Final Project: Building a Production-Ready Full Stack JS App

Project Overview

This project involves creating a comprehensive JavaScript application that encompasses both client-side and server-side functionalities. The project topic is flexible, allowing you to explore areas of personal interest or market demand. The aim is to develop a full stack application that is ready for production, emphasizing the integration of frontend, backend, and database components, and managing environments.

Key Requirements

- **Topic Flexibility**: you can choose any topic for your application. Make sure it's a topic that interests you, and that it might be an idea you're willing to develop further in the future.
- **Technology Stack**: Any JavaScript framework or runtime (e.g., React, Angular, Vue.js for frontend; Node.js, Deno for backend; npm, bun for runtime) can be utilized. Choice of database (SQL or NoSQL) is also flexible.
- **Architecture Components**:
    - **Frontend**: A user interface that is interactive and user-friendly.
    - **Backend**: Server-side logic that handles business processes.
    - **Database**: Persistent storage for application data.
- **Production Readiness**:
    - The application must be built and compiled for production.
    - It should handle both development and production environments.
    - Deployment must be online, accessible via a URL.
    - The application should be containerized using Docker for ease of deployment and scalability.
- **Communication**:
    - The frontend and backend must communicate using the backend's API to fulfill the application's functionality.

    - Communication between frontend and backend should be authenticated to ensure security.

- **Authentication**:

    - Implement login functionality with Google for user authentication.
    - Ensure that access to certain URLs is restricted to logged-in or authorized users only. Unauthorized access should redirect users to an appropriate screen, such as the login page.

- **API as a Service**:

    - Provide an API (added to the backend) that external users can interact with by passing an authentication token (JWT). The API should return responses based on the services offered by the application.
    - Implement proper error handling for unauthenticated requests.

- **API Paradigms**:

    - The application should utilize at least two API paradigms, ex: REST and WebSockets.

- **Testing**:

    - The application should have at leaset 3 unit tests, and 3 end-to-end tests.

- **Team Collaboration**:

    - you may work in groups of up to 3 members. Using a Task manager (clickup is a really nice tool for example, and this is purely optional) might help make sure that everyone has a clear path for what needs to be developed.

Submission Guidelines

- Submit your source code (without the **node_modules** folders).
- Include a README file with detailed instructions on how to set up and run the application, highlighting any environment-specific configurations.
- Provide a link to the deployed application.

This project is an opportunity for you to demonstrate your ability to create a comprehensive, real-world application using modern JavaScript technologies. Best of luck!