Full-Stack Intern Assignment

Objective

Your task is to build a full-stack application using a provided Figma design for the frontend and specified technologies for both frontend and backend. The application will consist of a React-based frontend with TypeScript and a Node.js backend with Prisma, adhering to modern development practices.

Frontend Requirements

- **Figma Design Conversion:**
  - Convert the provided Figma design link into a fully functional React application.
  - Use **TypeScript** for all code to ensure type safety.
- **Project Structure:**
  - Organize the project into three main areas:
    - **UI Components:** Reusable and modular components reflecting the Figma design.
    - **Business Logic:** Separate logic from UI for reusability and maintainability (e.g., using hooks or utility functions).
    - **API Handling:** Manage API interactions efficiently.
- **Required Libraries:**
  - **Zod:** Implement schema validation for data integrity.
  - **React Hook Form:** Use for form management and validation.
  - **React Query:** Handle data fetching, caching, and state management.
- **Type Safety:**
  - Ensure all API responses are typed using TypeScript interfaces or types for full type safety.
- **Error Handling:**
  - Implement proper error handling in the frontend to manage API errors and display meaningful messages to the user.

Backend Requirements

- **Technology Stack:**
  - Build a Node.js server using **TypeScript**.
  - Use **Prisma** for database management (schema definition and querying).
- **Prisma Schema:**
  - Create **only a user schema** in Prisma with the following fields:
    - email (string, unique)

        - password (string)
    - Ensure the schema is properly defined and migrations are handled.
- **Project Structure:**
    - Organize the backend into:
        - **Controllers:** Contain the business logic for each endpoint.
        - **Routes:** Define API endpoints (e.g., using Express.js or a similar router).
- **Error Handling:**
    - Implement robust error handling (e.g., custom error classes or middleware) to manage and respond to errors gracefully.

General Instructions

- **README Files:**
    - Include a detailed README in both the frontend and backend projects.
    - The README should cover:
        - Setup instructions (e.g., installing dependencies, configuring environment variables).
        - How to run the project locally.
        - A brief overview of the tech stack and project structure.
- **Video Demonstration:**
    - Record a video showing the working application (both frontend and backend in action).
    - The video must:
        - Demonstrate the full functionality of the application.
        - Show **proper error handling from the frontend** (e.g., invalid inputs, API errors).
        - Confirm that the **project is working** as expected.
    - Share the video via a **Google Drive link**.
    - Ensure the video clearly demonstrates functionality; failure to show a working project will result in immediate disqualification.
- **GitHub Repository:**
    - Provide a single GitHub repository containing both frontend and backend projects.
    - Use separate folders (e.g., /frontend and /backend) for clarity.
    - Ensure the repository is public or accessible to reviewers.
- **Deployment:**
    - Deployment is **not required**; focus on local functionality.

Evaluation Criteria

Your submission will be assessed based on the following:

- Accurate and complete implementation of the Figma design in the frontend.
- Proper use of TypeScript for type safety across both frontend and backend.
- Effective integration of Zod, React Hook Form, and React Query in the frontend.
- Correct implementation of the user schema in Prisma (with only email and password) and its integration with the backend.
- Clean, organized, and modular code structure in both projects.
- Robust error handling in both frontend and backend.
- Quality and clarity of the README files and video demonstration.

Additional Notes

- The application must be fully functional with no critical bugs.
- Focus on code quality, readability, and adherence to best practices.
- This is an individual assignment; collaboration is not allowed.

Submission

- Submit the following by: 31, March, 2025
    - The GitHub repository link containing both frontend and backend projects.
    - The Google Drive link to your video demonstration.
- Ensure both links are accessible to the reviewer.