React Project Requirement Document

1. Project Objective

Each student must develop a **complete React application** that demonstrates the following key concepts:

- **Authentication** – Register, Login, and Protected Routes
- **Routing** – Parent, Child, and Dynamic Routes using `react-router-dom`
- **CRUD Operations** – Create, Read, Update, Delete using **JSON Server**
- **Global State Management** – Using **Context API** or **Redux Toolkit**
- **Form Validation** – Proper field validation using controlled components
- **Reusable Components** – Modular and maintainable UI design
- **Data Persistence** – Using **Local Storage**
- **Responsive Layout** – Implemented using **React-Bootstrap** or **Material UI**
- **Version Control** – Git repository for all code commits
- **Optional Bonus:** Integrate **EmailJS** for sending emails through a contact form

---

2. Suggested Project Ideas

You can choose any practical project, for example:

- **Book Library Manager**
- **Movie Watchlist**
- **Fitness Tracker**
- **Ticket Booking Platform**
- Event Planner
- E-Commerce Storefront
- Blog Platform

3. Core Functional Requirements

Feature	Description
Authentication	Implement Register, Login, and Logout functionality. Secure certain routes using Protected Routes.
Routing	Use `react-router-dom` for navigation. Include Parent, Child, and Dynamic Routes (e.g., `/products/:id`).
CRUD Operations	Use **JSON Server** as a fake REST API. Implement Create, Read, Update, and Delete functionality.
Global State Management	Use **Context API** or **Redux** to manage global data such as user authentication and CRUD state.
Form Validation	Validate inputs (email, password, required fields) using React's controlled components or libraries like Formik/Yup.
Data Persistence	Store tokens, login status, or preferences in **Local Storage**.
Responsive UI	Use **React-Bootstrap** or **Material UI** to ensure responsive design.
EmailJS (Optional)	Integrate EmailJS for sending contact form data directly to an email inbox.

---

4. Technical Stack

Category	Technology
Frontend Framework	React (Vite or Create React App)
Routing	React Router DOM
State Management	Context API or Redux Toolkit
UI Framework	React-Bootstrap / Material UI
Backend (Mock)	JSON Server
Data Persistence	Local Storage
Form Handling	Controlled Components
Version Control	Git & GitHub
Optional	EmailJS for email functionality

---

5. React Concepts to Implement

- Functional Components & Props
- Hooks: `useState`, `useEffect`, `useContext` / `useSelector`, `useDispatch`
- Conditional Rendering
- Reusable Components (e.g., Button, Input, Card)
- API calls using `fetch` or `axios`
- JSON Server setup for CRUD
- Context Provider or Redux Store configuration
- ProtectedRoute component for authentication
- Local Storage integration for persistence

---

8. Expected Output

Your React App should:

- Allow user registration and login (authentication)
- Navigate across pages using React Router
- Perform CRUD operations via JSON Server
- Manage global state (Context API or Redux)
- Validate forms with clear error messages
- Store and retrieve data from Local Storage
- Be visually appealing and responsive
- Have clean commits in a GitHub repository