Full stack Development

Application Structure Overview:

1. Frontend (React)
   1. Framework: React, which allows us to create reuseable UI components.
   2. Component Architecture: (Login, Register, Admin, Forgot Password, Reset Password)
   3. State Management: React (useState, useEffect) used to managing handling input value and fetching the data.
   4. Routing: React Router navigation between different pages.
   5. API integration: Frontend interacts with the backend through API by using Axios and organized in service layer call Api.js.
2. Backend (C# with ASP.NET core)
   1. Framework: ASP.NET Core
   2. Controller-Based Architecture: The backend follows the MVC (Model-View-Controller) pattern. AuthContorller handle HTTP requests, logic, and return response.
   3. Service layer: logic is encapsulated in AuthService. Which handle tasks like user authentication, registration, password reset, and managing user data.
   4. Data model: User.cs and Request.cs represent the data structures for the project.
   5. In-Memory Data Storage: Currently, user data is stored in an in-memory list within the AuthService. This means that the data has not persisted across application restarts, but this structure allows for easy testing and development without setting up a database. (Will be use database future)
   6. Email integration: using SmtpClient class in C# for sending email.
3. Communication between frontend and backend
   1. The frontend and backend communicate via RESTful APIs. The frontend makes HTTP requests to the backend’s API endpoints (e.g., /api/auth/login, /api/auth/register), and the backend responds with JSON data.
4. Summary
   1. Your application is a full-stack web application with a React-based frontend and a C# ASP.NET Core backend. The React frontend handles the user interface, making HTTP requests to the ASP.NET Core backend, which manages business logic and data. The application is organized in a modular way, with clear separation of concerns between different components and layers, making it easy to manage, develop, and scale.
5. Tools & Technologies
   1. React, JavaScript, ASP.NET Core, C#, HTML/CSS, Axios, RESTful APIs, SmtpClient (for email).