IA03 – User Registration API with React Frontend

🎯 Objective

Implement a complete **User Registration System** that includes:

- A **backend API** (NestJS recommended) to handle user registration and data management.
  A **React frontend** that allows users to **sign up and log in** through a user-friendly interface.

🧩 Requirements

Backend Implementation (NestJS recommended)

Database Setup

- Create a `User` schema with the following fields:
  - `email` (String, required, unique)
  - `password` (String, required)
  - `createdAt` (Date, default to current date)

API Endpoints

- **POST** `/user/register`
  - Handle new user registration.
  - Validate input data (`email`, `password`).
  - Check for existing email before creating a new user.
  - Hash passwords before saving to the database.
  - Return appropriate success or error responses.

Error Handling

- Provide meaningful error messages for validation and database operations.

Security

- Use environment variables for sensitive configuration (e.g., database URI).
- Enable CORS for requests from the React frontend.

Frontend Implementation (React)

Styled using shadcn/ui, Material UI, Chakra UI, or Tailwind CSS.
Validation using React Hook Form (recommended).

Pages & Routing

- Implement routing for the following pages:
    - **Home**
    - **Login**
    - **Sign Up (Register)**

Sign Up Screen

- Create a **Sign Up** page that includes:
    - Registration form with **email** and **password** fields.
    - On submit, send a **POST** request to `/user/register` and display feedback messages.

Login Screen

- Create a **Login** page UI (no backend logic required).
    - Include **email** and **password** input fields.
    - Simulate login feedback (e.g., mock success message or redirection).

API Integration

- Use **React Query** (or equivalent) for:
    - Managing API requests (mutation for registration).
    - Handling loading and error states.
    - Improving UX with caching and automatic refetch when needed.

User Experience

- Provide clear validation messages (required fields, valid email format).
- Show visual feedback for success and error states.
- Ensure a responsive and accessible interface.

🧮 Rubric (10 points total)

Criteria	Description	Points
Backend Implementation	API Endpoint (`/register`)	2
	Error Handling	2
Frontend Implementation	Routing (Home, Login, Sign Up)	1
	Sign Up Page (Form, Validation, API Integration with React Query)	2
	Login Page (Form, Validation, UI with shadcn/ui or equivalent)	2
Deployment	Public host deployment	1

🚀 Deliverables

- Source code for both backend and frontend.
- Live deployed application (frontend + backend accessible publicly).
- Short README explaining how to install and run the project locally.