# Project Code Compilation

📌 Total Code Files: 33

## 1. Postcss.Config

📂 File Path: C:\Users\Admin\React\Academix\postcss.config.js

// postcss.config.js  
module.exports = {  
 plugins: {  
 tailwindcss: {},  
 autoprefixer: {},  
 },  
 }

================================================================================

## 2. Tailwind.Config

📂 File Path: C:\Users\Admin\React\Academix\tailwind.config.js

// tailwind.config.js  
module.exports = {  
 content: [  
 "./src/\*\*/\*.{js,jsx,ts,tsx}", // Add this line to your config  
 ],  
 theme: {  
 extend: {},  
 },  
 plugins: [],  
}

================================================================================

## 3. Database

📂 File Path: C:\Users\Admin\React\Academix\database\database.js

const mongoose = require("mongoose");  
  
const connectDB = async () => {  
 try {  
 await mongoose.connect("mongodb://localhost:27017/", {  
 useNewUrlParser: true,  
 useUnifiedTopology: true,  
 });  
 console.log("✅ MongoDB Connected Successfully (Localhost)");  
 } catch (error) {  
 console.error("❌ MongoDB Connection Failed:", error);  
 process.exit(1);  
 }  
};  
  
module.exports = connectDB;

================================================================================

## 4. Server

📂 File Path: C:\Users\Admin\React\Academix\database\server.js

const express = require("express");  
const mongoose = require("mongoose");  
const cors = require("cors")  
require("dotenv").config();  
  
const app = express();  
const PORT = process.env.PORT || 5000;  
  
  
app.use(express.json());   
const corsOptions = {  
 origin: true, // Reflect request origin  
 credentials: true,  
};  
app.use(cors(corsOptions));  
app.options("\*", cors(corsOptions));  
  
// MongoDB Connection (use 127.0.0.1 to avoid IPv6/localhost issues)  
const mongoURI = "mongodb://127.0.0.1:27017/academixDB";  
mongoose.connect(mongoURI, {  
 useNewUrlParser: true,  
 useUnifiedTopology: true,  
})  
.then(() => console.log("✅ MongoDB Connected!"))  
.catch(err => console.error("❌ MongoDB Connection Error:", err));  
  
// Cleanup: Drop stray 'projects' collection if it exists (we use 'project')  
mongoose.connection.once("open", async () => {  
 try {  
 const collections = await mongoose.connection.db.listCollections().toArray();  
 const hasPluralProjects = collections.some((c) => c.name === "projects");  
 if (hasPluralProjects) {  
 await mongoose.connection.db.dropCollection("projects");  
 console.log("🧹 Dropped stray 'projects' collection");  
 }  
 } catch (cleanupErr) {  
 console.warn("Cleanup check failed:", cleanupErr?.message || cleanupErr);  
 }  
});  
  
// Mongoose Schemas  
const userSchema = new mongoose.Schema(  
 {  
 name: { type: String, required: true, trim: true },  
 email: { type: String, required: true, unique: true, lowercase: true, trim: true },  
 password: { type: String, required: true },  
 phone: { type: String, default: null },  
 restricted: { type: Boolean, default: false },  
 displayName: { type: String, trim: true },  
 profilePicture: { type: String, trim: true },  
 skills: { type: [String], default: [] },  
 interests: { type: [String], default: [] },  
 },  
 { timestamps: true }  
);  
  
const messageSchema = new mongoose.Schema(  
 {  
 text: { type: String, required: true, trim: true },  
 senderId: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },  
 recipientId: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },  
 participants: { type: [mongoose.Schema.Types.ObjectId], ref: "User", index: true },  
 timestamp: { type: Date, default: Date.now },  
 },  
 { timestamps: true }  
);  
  
const User = mongoose.models.User || mongoose.model("User", userSchema);  
const Message = mongoose.models.Message || mongoose.model("Message", messageSchema);  
  
// Query (Q&A) schema  
const querySchema = new mongoose.Schema(  
 {  
 authorId: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },  
 content: { type: String, required: true, trim: true },  
 comments: [  
 {  
 authorId: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },  
 content: { type: String, required: true, trim: true },  
 createdAt: { type: Date, default: Date.now },  
 },  
 ],  
 },  
 { timestamps: true }  
);  
  
const Query = mongoose.models.Query || mongoose.model("Query", querySchema);  
  
// Event schema  
const eventSchema = new mongoose.Schema(  
 {  
 title: { type: String, required: true },  
 description: { type: String },  
 date: { type: String },  
 time: { type: String },  
 category: { type: String },  
 speakers: { type: String },  
 location: { type: String },  
 audienceLevel: { type: String, default: "General" },  
 registrationLink: { type: String },  
 },  
 { timestamps: true }  
);  
const Event = mongoose.models.Event || mongoose.model("Event", eventSchema);  
  
// Project schema  
const projectSchema = new mongoose.Schema(  
 {  
 title: { type: String, required: true },  
 description: { type: String, required: true },  
 technologies: { type: String },  
 startDate: { type: String },  
 endDate: { type: String },  
 level: { type: String },  
 contactEmail: { type: String },  
 },  
 { timestamps: true, collection: "project" }  
);  
const Project = mongoose.models.Project || mongoose.model("Project", projectSchema);  
  
// Study Group schema  
const studyGroupSchema = new mongoose.Schema(  
 {  
 title: { type: String, required: true },  
 description: { type: String, required: true },  
 prerequisites: { type: String },  
 skills: { type: String },  
 level: { type: String },  
 duration: { type: String },  
 mode: { type: String },  
 contactEmail: { type: String },  
 },  
 { timestamps: true }  
);  
const StudyGroup = mongoose.models.StudyGroup || mongoose.model("StudyGroup", studyGroupSchema);  
  
// Test Route  
app.get("/", (req, res) => {  
 res.send("Server is running!");  
});  
  
// Health check  
app.get("/api/health", async (req, res) => {  
 const state = mongoose.connection.readyState; // 1 = connected  
 res.json({ ok: true, mongoState: state });  
});  
  
// Signup Route  
app.post("/api/signup", async (req, res) => {  
 try {  
 const { name, email, password, phone, displayName } = req.body || {};  
  
 if (!name || !email || !password) {  
 return res.status(400).json({ error: "All fields are required" });  
 }  
  
 const existingUser = await User.findOne({ email });  
 if (existingUser) {  
 return res.status(409).json({ error: "Email already registered" });  
 }  
  
 console.log("➡️ Signup request:", { name, email });  
 const newUser = await User.create({ name, email, password, phone: phone || null, displayName: displayName || name });  
 console.log("✅ User saved:", newUser.\_id.toString());  
 return res.status(201).json({  
 message: "User created successfully",  
 user: { id: newUser.\_id, name: newUser.name, email: newUser.email },  
 });  
 } catch (error) {  
 console.error("Signup Error:", error);  
 // Handle duplicate key error explicitly  
 if (error && error.code === 11000) {  
 return res.status(409).json({ error: "Email already registered" });  
 }  
 return res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// List users (debug)  
app.get("/api/users", async (req, res) => {  
 const users = await User.find({}, { name: 1, email: 1, displayName: 1, profilePicture: 1, skills: 1, interests: 1, restricted: 1, phone: 1 }).sort({ createdAt: -1 }).limit(100);  
 res.json(users);  
});  
  
// Update user fields  
app.patch("/api/users/:id", async (req, res) => {  
 try {  
 const { id } = req.params;  
 const update = req.body || {};  
 const updated = await User.findByIdAndUpdate(id, update, { new: true });  
 if (!updated) return res.status(404).json({ error: "User not found" });  
 res.json(updated);  
 } catch (err) {  
 console.error("Update user error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Delete user  
app.delete("/api/users/:id", async (req, res) => {  
 try {  
 const { id } = req.params;  
 await User.findByIdAndDelete(id);  
 res.json({ ok: true });  
 } catch (err) {  
 console.error("Delete user error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Create message  
app.post("/api/messages", async (req, res) => {  
 try {  
 const { text, senderId, recipientId } = req.body || {};  
 if (!text || !senderId || !recipientId) {  
 return res.status(400).json({ error: "text, senderId, recipientId are required" });  
 }  
 const message = await Message.create({  
 text,  
 senderId,  
 recipientId,  
 participants: [senderId, recipientId],  
 timestamp: new Date(),  
 });  
 res.status(201).json(message);  
 } catch (err) {  
 console.error("Create message error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Get messages between two users  
app.get("/api/messages", async (req, res) => {  
 try {  
 const { a, b } = req.query;  
 if (!a || !b) return res.status(400).json({ error: "Query params 'a' and 'b' are required" });  
 const messages = await Message.find({  
 $or: [  
 { senderId: a, recipientId: b },  
 { senderId: b, recipientId: a },  
 ],  
 })  
 .sort({ timestamp: 1 });  
 res.json(messages);  
 } catch (err) {  
 console.error("Get messages error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Create a query  
app.post("/api/queries", async (req, res) => {  
 try {  
 const { authorId, content } = req.body || {};  
 if (!authorId || !content) return res.status(400).json({ error: "authorId and content are required" });  
 const created = await Query.create({ authorId, content });  
 res.status(201).json(created);  
 } catch (err) {  
 console.error("Create query error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// List queries  
app.get("/api/queries", async (\_req, res) => {  
 try {  
 const items = await Query.find({}).sort({ createdAt: -1 }).limit(200);  
 res.json(items);  
 } catch (err) {  
 console.error("List queries error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Update a query content  
app.patch("/api/queries/:id", async (req, res) => {  
 try {  
 const { id } = req.params;  
 const { content } = req.body || {};  
 const updated = await Query.findByIdAndUpdate(id, { content }, { new: true });  
 if (!updated) return res.status(404).json({ error: "Query not found" });  
 res.json(updated);  
 } catch (err) {  
 console.error("Update query error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Delete a query  
app.delete("/api/queries/:id", async (req, res) => {  
 try {  
 const { id } = req.params;  
 await Query.findByIdAndDelete(id);  
 res.json({ ok: true });  
 } catch (err) {  
 console.error("Delete query error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Add a comment to a query  
app.post("/api/queries/:id/comments", async (req, res) => {  
 try {  
 const { id } = req.params;  
 const { authorId, content } = req.body || {};  
 if (!authorId || !content) return res.status(400).json({ error: "authorId and content are required" });  
 const updated = await Query.findByIdAndUpdate(  
 id,  
 { $push: { comments: { authorId, content, createdAt: new Date() } } },  
 { new: true }  
 );  
 if (!updated) return res.status(404).json({ error: "Query not found" });  
 res.json(updated);  
 } catch (err) {  
 console.error("Add comment error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Events routes  
app.post("/api/events", async (req, res) => {  
 try {  
 const created = await Event.create(req.body || {});  
 res.status(201).json(created);  
 } catch (err) {  
 console.error("Create event error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.get("/api/events", async (\_req, res) => {  
 try {  
 const items = await Event.find({}).sort({ createdAt: -1 }).limit(200);  
 res.json(items);  
 } catch (err) {  
 console.error("List events error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.delete("/api/events/:id", async (req, res) => {  
 try {  
 await Event.findByIdAndDelete(req.params.id);  
 res.json({ ok: true });  
 } catch (err) {  
 console.error("Delete event error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Projects routes  
app.post("/api/projects", async (req, res) => {  
 try {  
 const payload = req.body || {};  
 const { title, description, contactEmail, startDate, endDate } = payload;  
 if (!title || !description || !contactEmail || !startDate || !endDate) {  
 return res.status(400).json({ error: "title, description, contactEmail, startDate, endDate are required" });  
 }  
 console.log("➡️ Create Project:", { title });  
 const created = await Project.create(payload);  
 console.log("✅ Project saved:", created.\_id.toString());  
 res.status(201).json(created);  
 } catch (err) {  
 console.error("Create project error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.get("/api/projects", async (\_req, res) => {  
 try {  
 const items = await Project.find({}).sort({ createdAt: -1 }).limit(200);  
 res.json(items);  
 } catch (err) {  
 console.error("List projects error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.delete("/api/projects/:id", async (req, res) => {  
 try {  
 await Project.findByIdAndDelete(req.params.id);  
 res.json({ ok: true });  
 } catch (err) {  
 console.error("Delete project error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Study Groups routes  
app.post("/api/study-groups", async (req, res) => {  
 try {  
 const created = await StudyGroup.create(req.body || {});  
 res.status(201).json(created);  
 } catch (err) {  
 console.error("Create study group error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.get("/api/study-groups", async (\_req, res) => {  
 try {  
 const items = await StudyGroup.find({}).sort({ createdAt: -1 }).limit(200);  
 res.json(items);  
 } catch (err) {  
 console.error("List study groups error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
app.delete("/api/study-groups/:id", async (req, res) => {  
 try {  
 await StudyGroup.findByIdAndDelete(req.params.id);  
 res.json({ ok: true });  
 } catch (err) {  
 console.error("Delete study group error:", err);  
 res.status(500).json({ error: "Internal server error" });  
 }  
});  
  
// Start Server  
app.listen(PORT, () => {  
 console.log(`🚀 Server running on http://localhost:${PORT}`);  
});

================================================================================

## 5. Index

📂 File Path: C:\Users\Admin\React\Academix\public\index.html

<!DOCTYPE html>  
<html lang="en">  
 <head>  
 <link  
 href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css"  
 rel="stylesheet"  
/>  
 <meta charset="utf-8" />  
 <link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />  
 <meta name="viewport" content="width=device-width, initial-scale=1" />  
   
 <meta  
 name="description"  
 content="Web site created using create-react-app"  
 />  
 <link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />  
 <!--  
 manifest.json provides metadata used when your web app is installed on a  
 user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/  
 -->  
 <link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />  
 <!--  
 Notice the use of %PUBLIC\_URL% in the tags above.  
 It will be replaced with the URL of the `public` folder during the build.  
 Only files inside the `public` folder can be referenced from the HTML.  
  
 Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will  
 work correctly both with client-side routing and a non-root public URL.  
 Learn how to configure a non-root public URL by running `npm run build`.  
 -->  
 <title>React App</title>  
 </head>  
 <body>  
 <noscript>You need to enable JavaScript to run this app.</noscript>  
 <div id="root"></div>  
 <!--  
 This HTML file is a template.  
 If you open it directly in the browser, you will see an empty page.  
  
 You can add webfonts, meta tags, or analytics to this file.  
 The build step will place the bundled scripts into the <body> tag.  
  
 To begin the development, run `npm start` or `yarn start`.  
 To create a production bundle, use `npm run build` or `yarn build`.  
 -->  
 </body>  
</html>

================================================================================

## 6. App

📂 File Path: C:\Users\Admin\React\Academix\src\App.css

.App {  
 text-align: center;  
}  
  
.App-logo {  
 height: 40vmin;  
 pointer-events: none;  
}  
  
@media (prefers-reduced-motion: no-preference) {  
 .App-logo {  
 animation: App-logo-spin infinite 20s linear;  
 }  
}  
  
.App-header {  
 background-color: #282c34;  
 min-height: 100vh;  
 display: flex;  
 flex-direction: column;  
 align-items: center;  
 justify-content: center;  
 font-size: calc(10px + 2vmin);  
 color: white;  
}  
  
.App-link {  
 color: #61dafb;  
}  
  
@keyframes App-logo-spin {  
 from {  
 transform: rotate(0deg);  
 }  
 to {  
 transform: rotate(360deg);  
 }  
}

================================================================================

## 7. App

📂 File Path: C:\Users\Admin\React\Academix\src\App.js

import React from "react";  
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";  
import LandingPage from "./components/LandingPage";  
import LoginPage from "./components/loginpage";  
import SignupPage from "./components/SignupPage";  
import Dashboard from "./components/Dashboard";  
import AdminDashboard from "./components/AdminDashboard";   
import StudentDetails from "./components/StudentDetails";  
import Project from "./dashboard/project";  
import ActiveCollaborations from "./dashboard/activecollabrations";  
import StudyGroup from "./dashboard/studygroup";  
import QueriesWritten from "./dashboard/QueriesWritten";  
import ProfileSettings from "./dashboard/profilesettings";  
import AccountManagement from "./dashboard/accountmanagement";  
import EventManagement from "./dashboard/EventManagement"; // Ensure this path is correct  
import Chat from "./components/Chat";   
import { AuthProvider } from "./components/authContext";   
import './App.css';  
  
function App() {  
 return (  
<React.Fragment>  
 <AuthProvider>   
 <Router>  
 <Routes>  
 <Route path="/" element={<LandingPage />} />  
 <Route path="/landing" element={<LandingPage />} />  
 <Route path="/login" element={<LoginPage />} />  
 <Route path="/signup" element={<SignupPage />} />  
 <Route path="/dashboard" element={<Dashboard />} />  
 <Route path="/project" element={<Project />} />  
 <Route path="/collaborations" element={<ActiveCollaborations />} />  
 <Route path="/studygroup" element={<StudyGroup />} />  
 <Route path="/queries-written" element={<QueriesWritten />} />  
 <Route path="/profile-settings" element={<ProfileSettings />} />  
 <Route path="/account-management" element={<AccountManagement />} />  
 <Route path="/signout" element={<LandingPage />} />  
 <Route path="/event" element={<EventManagement />} />   
 <Route path="/chat" element={<Chat />} />  
 <Route path="/admin-dashboard" element={<AdminDashboard />} />  
 <Route path="/student-details" element={<StudentDetails />} />  
 <Route path="\*" element={<LandingPage />} />  
 </Routes>  
 </Router>  
 </AuthProvider>  
   
</React.Fragment>  
  
 );  
}  
  
export default App;

================================================================================

## 8. App.Test

📂 File Path: C:\Users\Admin\React\Academix\src\App.test.js

import { render, screen } from '@testing-library/react';  
import App from './App';  
  
test('renders learn react link', () => {  
 render(<App />);  
 const linkElement = screen.getByText(/learn react/i);  
 expect(linkElement).toBeInTheDocument();  
});

================================================================================

## 9. Index

📂 File Path: C:\Users\Admin\React\Academix\src\index.css

@import "tailwindcss/base";  
@import "tailwindcss/components";  
@import "tailwindcss/utilities";  
  
/\* Custom scrollbar styles for chat \*/  
.chat-scrollbar::-webkit-scrollbar {  
 width: 6px;  
}  
  
.chat-scrollbar::-webkit-scrollbar-track {  
 background: #1F2937;  
 border-radius: 3px;  
}  
  
.chat-scrollbar::-webkit-scrollbar-thumb {  
 background: #4B5563;  
 border-radius: 3px;  
}  
  
.chat-scrollbar::-webkit-scrollbar-thumb:hover {  
 background: #6B7280;  
}

================================================================================

## 10. Index

📂 File Path: C:\Users\Admin\React\Academix\src\index.js

import React from 'react';  
import ReactDOM from 'react-dom/client';  
import './index.css';  
import App from './App';  
import reportWebVitals from './reportWebVitals';  
  
const root = ReactDOM.createRoot(document.getElementById('root'));  
root.render(  
 <React.StrictMode>  
 <App />  
 </React.StrictMode>  
);  
  
reportWebVitals();

================================================================================

## 11. Reportwebvitals

📂 File Path: C:\Users\Admin\React\Academix\src\reportWebVitals.js

const reportWebVitals = onPerfEntry => {  
 if (onPerfEntry && onPerfEntry instanceof Function) {  
 import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {  
 getCLS(onPerfEntry);  
 getFID(onPerfEntry);  
 getFCP(onPerfEntry);  
 getLCP(onPerfEntry);  
 getTTFB(onPerfEntry);  
 });  
 }  
};  
  
export default reportWebVitals;

================================================================================

## 12. Setuptests

📂 File Path: C:\Users\Admin\React\Academix\src\setupTests.js

// jest-dom adds custom jest matchers for asserting on DOM nodes.  
// allows you to do things like:  
// expect(element).toHaveTextContent(/react/i)  
// learn more: https://github.com/testing-library/jest-dom  
import '@testing-library/jest-dom';

================================================================================

## 13. Admindashboard

📂 File Path: C:\Users\Admin\React\Academix\src\components\AdminDashboard.js

import React, { useState, useEffect } from "react";  
import { useNavigate } from "react-router-dom";  
import StudentDetails from "./StudentDetails";  
import ProfileSettings from "../dashboard/profilesettings"; // Reusing Student Profile Settings  
import AccountManagement from "../dashboard/accountmanagement"; // Reusing Student Account Management  
  
const AdminDashboard = () => {  
 const navigate = useNavigate();  
 const [activeTab, setActiveTab] = useState("dashboard");  
 const [activities, setActivities] = useState([]);  
  
 // Fetch student activities (placeholder)  
 useEffect(() => {  
 setActivities([  
 { id: 1, name: "John Doe", action: "Joined Study Group", date: "23-Mar-2025" },  
 { id: 2, name: "Jane Smith", action: "Posted a Query", date: "22-Mar-2025" },  
 ]);  
 }, []);  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-gray-900 via-gray-800 to-gray-700 text-white">  
   
 {/\* Header \*/}  
 <div className="bg-gray-900 text-white p-4 flex justify-between items-center shadow-lg">  
 <h2 className="text-2xl font-extrabold text-blue-400">Admin Dashboard</h2>  
 <button  
 onClick={() => navigate("/login")}  
 className="bg-red-600 text-white p-2 rounded-lg hover:bg-red-700 transition"  
 >  
 🚪 Logout  
 </button>  
 </div>  
  
 {/\* Navigation Tabs \*/}  
 <div className="flex justify-center gap-4 p-4">  
 <button onClick={() => setActiveTab("dashboard")} className={`p-2 rounded-lg ${activeTab === "dashboard" ? "bg-blue-500" : "bg-gray-700"} text-white hover:bg-blue-600 transition`}>  
 🎓 Student Management  
 </button>  
 <button onClick={() => setActiveTab("profile")} className={`p-2 rounded-lg ${activeTab === "profile" ? "bg-blue-500" : "bg-gray-700"} text-white hover:bg-blue-600 transition`}>  
 🛠 Profile Settings  
 </button>  
 <button onClick={() => setActiveTab("account")} className={`p-2 rounded-lg ${activeTab === "account" ? "bg-blue-500" : "bg-gray-700"} text-white hover:bg-blue-600 transition`}>  
 ⚙ Account Management  
 </button>  
 </div>  
  
 {/\* Display Selected Section \*/}  
 <div className="p-6">  
 {activeTab === "dashboard" && (  
 <>  
 <div className="p-6">  
 <button  
 onClick={() => navigate("/student-details")}  
 className="bg-blue-500 text-white p-3 w-[200px] rounded-lg shadow-lg hover:bg-blue-600 transition"  
 >  
 👨‍🎓 View All Students  
 </button>  
 </div>  
  
 {/\* Student Activity Logs \*/}  
 <div className="bg-gray-800 p-6 rounded-lg shadow-lg mt-6 mx-4">  
 <h3 className="text-xl font-bold mb-4 text-blue-400">📊 Student Activity Logs</h3>  
 <table className="w-full border-collapse border border-gray-700">  
 <thead>  
 <tr className="bg-gray-900 text-white">  
 <th className="p-2 border">Student</th>  
 <th className="p-2 border">Activity</th>  
 <th className="p-2 border">Date</th>  
 </tr>  
 </thead>  
 <tbody>  
 {activities.length > 0 ? (  
 activities.map((activity) => (  
 <tr key={activity.id} className="text-center border border-gray-700">  
 <td className="p-2 border">{activity.name}</td>  
 <td className="p-2 border">{activity.action}</td>  
 <td className="p-2 border">{activity.date}</td>  
 </tr>  
 ))  
 ) : (  
 <tr>  
 <td colSpan="3" className="p-4 text-gray-400">No activities recorded.</td>  
 </tr>  
 )}  
 </tbody>  
 </table>  
 </div>  
 </>  
 )}  
 {activeTab === "profile" && <ProfileSettings />}  
 {activeTab === "account" && <AccountManagement />}  
 </div>  
  
 {/\* Footer \*/}  
 <div className="text-center text-gray-400 text-sm mt-6 p-4">  
 Ⓒ {new Date().getFullYear()} EduTech Admin Panel  
 </div>  
 </div>  
 );  
};  
  
export default AdminDashboard;

================================================================================

## 14. Chat

📂 File Path: C:\Users\Admin\React\Academix\src\components\Chat.js

import { useState, useEffect, useRef, useContext } from "react";  
import axios from "axios";  
import { AuthContext } from "./authContext";  
  
const Chat = () => {  
 const { currentUser } = useContext(AuthContext);  
 const [messages, setMessages] = useState([]);  
 const [newMessage, setNewMessage] = useState("");  
 const [recipientId, setRecipientId] = useState("");   
 const [messageView, setMessageView] = useState("received");   
 const [allUsers, setAllUsers] = useState([]);   
 const chatEndRef = useRef(null);  
  
 useEffect(() => {  
 if (!currentUser) return;  
 const fetchUsers = async () => {  
 try {  
 const { data } = await axios.get("http://localhost:5000/api/users");  
 setAllUsers(  
 data.map((u) => ({  
 id: u.\_id || u.id,  
 displayName: u.displayName || u.name,  
 }))  
 );  
 } catch (err) {  
 console.error("Failed to load users", err);  
 }  
 };  
 fetchUsers();  
 }, [currentUser]);  
  
 useEffect(() => {  
 if (!currentUser || !recipientId) return;  
 const fetchMessages = async () => {  
 try {  
 const selfId = currentUser.id || currentUser.\_id;  
 const { data } = await axios.get(  
 `http://localhost:5000/api/messages?a=${selfId}&b=${recipientId}`  
 );  
 setMessages(data.map((m) => ({ id: m.\_id, ...m })));  
 } catch (err) {  
 console.error("Failed to load messages", err);  
 }  
 };  
 fetchMessages();  
 }, [recipientId, currentUser]);  
  
 const sendMessage = async () => {  
 if (!newMessage.trim() || !recipientId.trim() || !currentUser) return;  
  
 try {  
 const selfId = currentUser.id || currentUser.\_id;  
 const messageData = {  
 text: newMessage,  
 senderId: selfId,  
 recipientId,  
 createdAt: new Date().toISOString()  
 };  
   
 await axios.post("http://localhost:5000/api/messages", messageData);  
   
 // Clear input immediately  
 setNewMessage("");  
   
 // Add message to local state immediately for instant feedback  
 const newMsg = {  
 id: Date.now(), // temporary ID  
 ...messageData,  
 \_id: Date.now()  
 };  
 setMessages(prev => [...prev, newMsg]);  
   
 // Scroll to bottom  
 setTimeout(() => {  
 chatEndRef.current?.scrollIntoView({ behavior: "smooth" });  
 }, 100);  
   
 // Refresh messages from server  
 const { data } = await axios.get(  
 `http://localhost:5000/api/messages?a=${selfId}&b=${recipientId}`  
 );  
 setMessages(data.map((m) => ({ id: m.\_id, ...m })));  
 } catch (error) {  
 console.error("Error sending message:", error);  
 // Remove the temporary message if sending failed  
 setMessages(prev => prev.filter(msg => msg.id !== Date.now()));  
 }  
 };  
  
 const handleInputKeyDown = (e) => {  
 if (e.key === "Enter") {  
 e.preventDefault();  
 sendMessage();  
 }  
 };  
  
 return (  
 <div className="bg-gray-800 p-4 rounded-lg shadow-lg flex flex-col h-full">  
   
  
 {!currentUser ? (  
 <p className="text-center text-gray-400">Loading chat...</p>  
 ) : (  
 <>  
 {/\* Select Recipient \*/}  
 <div className="mb-2">  
 <label className="block text-xs text-gray-300 mb-1">  
 Select a user to chat with:  
 </label>  
 <select  
 value={recipientId}  
 onChange={(e) => setRecipientId(e.target.value)}  
 className="w-full p-2 text-sm rounded-lg bg-gray-700 text-white border border-gray-600 focus:border-blue-500 focus:outline-none"  
 >  
 <option value="">Choose a user...</option>  
 {allUsers  
 .filter(  
 (user) =>  
 user.id !== (currentUser.id || currentUser.\_id)  
 )  
 .map((user) => (  
 <option key={user.id} value={user.id}>  
 {user.displayName || "Anonymous"} ({user.id})  
 </option>  
 ))}  
 </select>  
 </div>  
  
 {/\* Toggle Sent/Received Messages \*/}  
 <div className="flex justify-center mb-2">  
 <div className="inline-flex rounded-full bg-gray-700 p-1 shadow-inner">  
 <button  
 onClick={() => setMessageView("received")}  
 className={`px-4 py-1 rounded-full text-xs font-medium transition-all duration-200 ${  
 messageView === "received"  
 ? "bg-blue-600 text-white shadow-md"  
 : "text-gray-300 hover:text-white hover:bg-gray-600"  
 }`}  
 >  
 Received  
 </button>  
 <button  
 onClick={() => setMessageView("sent")}  
 className={`px-4 py-1 rounded-full text-xs font-medium transition-all duration-200 ${  
 messageView === "sent"  
 ? "bg-blue-600 text-white shadow-md"  
 : "text-gray-300 hover:text-white hover:bg-gray-600"  
 }`}  
 >  
 Sent  
 </button>  
 </div>  
 </div>  
  
 {/\* Chat Messages \*/}  
 <div className="overflow-y-auto flex flex-col space-y-1 mb-2 p-2 bg-gray-900 rounded-lg flex-grow min-h-[100px] max-h-[400px] chat-scrollbar">  
 {recipientId ? (  
 (() => {  
 const filteredMessages = messages.filter((msg) =>  
 messageView === "sent"  
 ? String(msg.senderId) === String(currentUser.id || currentUser.\_id) &&  
 String(msg.recipientId) === String(recipientId)  
 : String(msg.senderId) === String(recipientId) &&  
 String(msg.recipientId) === String(currentUser.id || currentUser.\_id)  
 );  
   
 return filteredMessages.length > 0 ? (  
 filteredMessages.map((msg) => (  
 <div  
 key={msg.id}  
 className={`p-2 text-sm max-w-[80%] rounded-lg ${  
 String(msg.senderId) === String(currentUser.id || currentUser.\_id)  
 ? "ml-auto bg-blue-500 text-white text-right"  
 : "bg-gray-600 text-white"  
 }`}  
 >  
 <p>{msg.text}</p>  
 <p className="text-xs opacity-70 mt-1">  
 {new Date(msg.createdAt || Date.now()).toLocaleTimeString()}  
 </p>  
 </div>  
 ))  
 ) : (  
 <div className="flex items-center justify-center h-full">  
 <p className="text-gray-400 text-sm">No messages yet. Start the conversation!</p>  
 </div>  
 );  
 })()  
 ) : (  
 <div className="flex items-center justify-center h-full">  
 <p className="text-gray-400 text-sm">Select a user to start chatting</p>  
 </div>  
 )}  
 <div ref={chatEndRef} />  
 </div>  
  
 {/\* Message Input Field \*/}  
 <div className="border-t border-gray-600 pt-3 flex-shrink-0 bg-gray-800 rounded-lg">  
 <div className="flex items-center space-x-2">  
 <input  
 type="text"  
 value={newMessage}  
 onChange={(e) => setNewMessage(e.target.value)}  
 onKeyDown={handleInputKeyDown}  
 placeholder={  
 recipientId  
 ? "Type your message here..."  
 : "Select a user first to start chatting"  
 }  
 disabled={!recipientId}  
 className="flex-grow p-3 text-sm border-2 border-blue-500 rounded-lg bg-gray-700 text-white focus:border-blue-400 focus:outline-none focus:ring-2 focus:ring-blue-500 disabled:bg-gray-800 disabled:text-gray-500 disabled:cursor-not-allowed disabled:border-gray-600"  
 />  
 <button  
 onClick={sendMessage}  
 disabled={!recipientId || !newMessage.trim()}  
 className="bg-blue-600 disabled:bg-gray-500 disabled:cursor-not-allowed text-white px-6 py-3 text-sm rounded-lg hover:bg-blue-700 transition-colors font-medium shadow-lg"  
 >  
 Send  
 </button>  
 </div>  
 {!recipientId && (  
 <p className="text-xs text-gray-400 mt-2 text-center">  
 Please select a user from the dropdown above to start chatting  
 </p>  
 )}  
 </div>  
 </>  
 )}  
 </div>  
 );  
};  
  
export default Chat;

================================================================================

## 15. Dashboard

📂 File Path: C:\Users\Admin\React\Academix\src\components\Dashboard.js

import React, { useState } from "react";  
import { useNavigate } from "react-router-dom";  
import Activity from "../dashboard/Activity";  
import Peers from "../dashboard/Peers";  
import Chat from "./Chat";   
  
const Dashboard = () => {  
 const navigate = useNavigate();  
 const [sidebarVisible, setSidebarVisible] = useState(false);  
  
 const toggleSidebar = () => setSidebarVisible(!sidebarVisible);  
  
 const handleSignOut = () => {  
 localStorage.removeItem("authToken");  
 navigate("/");  
 };  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-gray-900 to-gray-800 text-white font-sans">  
   
 {/\* Header \*/}  
 <div className="bg-gray-900 text-white p-4 flex justify-between items-center shadow-lg">  
 <h2 className="text-2xl font-extrabold text-blue-400">Academix</h2>  
 <button  
 onClick={toggleSidebar}  
 className="bg-blue-600 hover:bg-blue-700 text-white p-2 rounded-full shadow-lg transition duration-300"  
 >  
 👤  
 </button>  
 </div>  
  
 {/\* Sidebar \*/}  
 <div  
 className={`fixed top-0 left-0 h-full w-56 bg-gray-800 p-4 transform ${  
 sidebarVisible ? "translate-x-0" : "-translate-x-64"  
 } transition-transform duration-300 ease-in-out`}  
 >  
 <ul className="space-y-3">  
 <li>  
 <button  
 onClick={() => navigate("/profile-settings")}  
 className="w-full p-2 bg-blue-600 rounded-md hover:bg-blue-700 transition"  
 >  
 Profile Settings  
 </button>  
 </li>  
 <li>  
 <button  
 onClick={() => navigate("/account-management")}  
 className="w-full p-2 bg-blue-600 rounded-md hover:bg-blue-700 transition"  
 >  
 Account Management  
 </button>  
 </li>  
 <li>  
 <button  
 onClick={handleSignOut}  
 className="w-full p-2 bg-red-600 rounded-md hover:bg-red-700 transition"  
 >  
 Sign Out  
 </button>  
 </li>  
 </ul>  
 </div>  
  
 {/\* Stats Section \*/}  
 <div className="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-5 gap-4 p-4">  
 {[   
 { path: "/collaborations", icon: "👥", title: "Collaborations", count: 3, color: "blue" },  
 { path: "/studygroup", icon: "📖", title: "Study Groups", count: 2, color: "purple" },  
 { path: "/project", icon: "🎯", title: "Projects", count: 4, color: "green" },  
 { path: "/queries-written", icon: "📅", title: "Queries", count: 5, color: "red" },  
 { path: "/event", icon: "📅", title: "Events", count: "Manage", color: "blue" }, // Corrected path  
 ].map((item, index) => (  
 <button  
 key={index}  
 onClick={() => navigate(item.path)}  
 className={`flex flex-col items-center justify-center bg-${item.color}-500 text-white p-4 w-full max-w-[180px] h-auto rounded-lg shadow-lg hover:bg-${item.color}-600 transition duration-300 mx-auto`}  
 >  
 <span className="text-3xl">{item.icon}</span>  
 <span className="font-semibold text-sm">{item.title}</span>  
 <span className="text-lg font-bold">{item.count}</span>  
 </button>  
 ))}  
 </div>  
  
 {/\* Second Row: Activity, Peers, and Chat \*/}  
 <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4 p-4">  
   
 {/\* Activity Section \*/}  
 <div className="bg-gray-800 p-4 rounded-lg shadow-lg min-h-[224px] max-h-[500px] flex flex-col">  
 <h2 className="text-lg font-bold mb-2">Recent Activity</h2>  
 <div className="text-sm flex-grow overflow-hidden">  
 <Activity />  
 </div>  
 </div>  
  
 {/\* Peers Section \*/}  
 <div className="bg-gray-800 p-4 rounded-lg shadow-lg min-h-[224px] max-h-[500px] flex flex-col">  
 <h2 className="text-lg font-bold mb-2">Your Peers</h2>  
 <div className="text-sm flex-grow overflow-hidden">  
 <Peers />  
 </div>  
 </div>  
  
 {/\* Chat Section (Aligned and Fully Functional) \*/}  
 <div className="bg-gray-800 p-4 rounded-lg shadow-lg min-h-[320px] max-h-[600px] flex flex-col">  
 <h2 className="text-lg font-bold mb-2">Chat</h2>  
 <div className="text-sm flex-grow overflow-hidden">  
 <Chat />  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
};  
  
export default Dashboard;

================================================================================

## 16. Landingpage

📂 File Path: C:\Users\Admin\React\Academix\src\components\LandingPage.js

import React from "react";  
import { useNavigate } from "react-router-dom";  
  
const LandingPage = () => {  
 const navigate = useNavigate();  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-gray-900 via-gray-800 to-gray-700 text-white font-sans">  
 {/\* Navbar \*/}  
 <header className="bg-gray-900 shadow-md py-5">  
 <div className="container mx-auto flex justify-between items-center px-8">  
 <h1 className="text-2xl font-extrabold text-blue-400">EduTech</h1>  
 <div>  
 <button  
 className="px-5 py-2 text-white border border-gray-500 rounded-lg mr-4 hover:bg-gray-700 transition duration-300"  
 onClick={() => navigate("/login")}  
 >  
 Sign In  
 </button>  
 <button  
 className="px-5 py-2 bg-blue-600 rounded-lg hover:bg-blue-700 transition duration-300"  
 onClick={() => navigate("/signup")}  
 >  
 Get Started  
 </button>  
 </div>  
 </div>  
 </header>  
  
 {/\* Hero Section \*/}  
 <section className="text-center py-24 px-6">  
 <div className="container mx-auto">  
 <h1 className="text-5xl font-extrabold leading-tight">  
 Find Your Perfect <span className="text-blue-400">EduTech Partner</span>  
 </h1>  
 <p className="text-lg text-gray-300 mt-4">  
 Connect with like-minded students, collaborate on projects, and excel in your academic journey through AI-powered peer matching.  
 </p>  
 <div className="mt-8">  
 <button  
 className="px-6 py-3 bg-blue-600 rounded-lg text-lg hover:bg-blue-700 transition duration-300 mr-4"  
 onClick={() => navigate("/signup")}  
 >  
 Get Started Now  
 </button>  
 <button  
 className="px-6 py-3 bg-gray-700 text-white border border-gray-500 rounded-lg text-lg hover:bg-gray-600 transition duration-300"  
 onClick={() => navigate("/login")}  
 >  
 Sign In  
 </button>  
 </div>  
 </div>  
 </section>  
  
 {/\* Features Section \*/}  
 <section className="py-20 bg-gray-800">  
 <div className="container mx-auto px-6">  
 <h2 className="text-3xl font-bold text-center text-blue-400">  
 Why Choose EduTech?  
 </h2>  
 <p className="text-gray-300 text-center mt-4 mb-10">  
 Discover the features that make our platform unique and effective.  
 </p>  
  
 <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 gap-8">  
 {/\* Feature 1 \*/}  
 <div className="bg-gray-900 p-6 rounded-xl shadow-lg text-center hover:scale-105 transition-transform duration-300">  
 <div className="text-blue-400 text-5xl mb-4">👥</div>  
 <h3 className="text-xl font-semibold">Smart Matching</h3>  
 <p className="text-gray-300 mt-2">  
 Our AI-powered system connects you with peers who complement your skills and learning style.  
 </p>  
 </div>  
  
 {/\* Feature 2 \*/}  
 <div className="bg-gray-900 p-6 rounded-xl shadow-lg text-center hover:scale-105 transition-transform duration-300">  
 <div className="text-blue-400 text-5xl mb-4">📚</div>  
 <h3 className="text-xl font-semibold">Study Groups</h3>  
 <p className="text-gray-300 mt-2">  
 Form or join study groups based on your courses and academic interests.  
 </p>  
 </div>  
  
 {/\* Feature 3 \*/}  
 <div className="bg-gray-900 p-6 rounded-xl shadow-lg text-center hover:scale-105 transition-transform duration-300">  
 <div className="text-blue-400 text-5xl mb-4">🛠️</div>  
 <h3 className="text-xl font-semibold">Project Collaboration</h3>  
 <p className="text-gray-300 mt-2">  
 Find teammates for academic projects and research initiatives.  
 </p>  
 </div>  
  
 {/\* Feature 4 \*/}  
 <div className="bg-gray-900 p-6 rounded-xl shadow-lg text-center hover:scale-105 transition-transform duration-300">  
 <div className="text-blue-400 text-5xl mb-4">💡</div>  
 <h3 className="text-xl font-semibold">Skill Enhancement</h3>  
 <p className="text-gray-300 mt-2">  
 Learn from peers and share your knowledge in a collaborative environment.  
 </p>  
 </div>  
 </div>  
 </div>  
 </section>  
 </div>  
 );  
};  
  
export default LandingPage;

================================================================================

## 17. Signuppage

📂 File Path: C:\Users\Admin\React\Academix\src\components\SignupPage.js

import React, { useState } from "react";  
import { useNavigate } from "react-router-dom";  
import axios from "axios";  
  
const SignupPage = () => {  
 const [formData, setFormData] = useState({  
 name: "",  
 email: "",  
 password: ""  
 });  
 const navigate = useNavigate();  
  
 const validateForm = () => {  
 // Simple validation  
 return (  
 formData.name.trim() !== "" &&  
 formData.email.trim() !== "" &&  
 formData.password.trim() !== ""  
 );  
 };  
  
 const handleChange = (e) => {  
 const { name, value } = e.currentTarget || e.target;  
 setFormData((prev) => ({  
 ...prev,  
 [name]: value,  
 }));  
 };  
  
 const handleSubmit = async (e) => {  
 e.preventDefault();  
 if (validateForm()) {  
 try {  
 const payload = {  
 name: formData.name.trim(),  
 email: formData.email.trim(),  
 password: formData.password.trim(),  
 displayName: formData.name.trim(),  
 };  
 // Determine API hosts to try  
 const currentHost = typeof window !== "undefined" ? window.location.hostname : "localhost";  
 const candidates = [  
 `http://${currentHost}:5000`,  
 "http://localhost:5000",  
 "http://127.0.0.1:5000",  
 ];  
 let data;  
 let lastError;  
 for (const base of candidates) {  
 try {  
 ({ data } = await axios.post(`${base}/api/signup`, payload, {  
 headers: { "Content-Type": "application/json" },  
 timeout: 8000,  
 }));  
 break;  
 } catch (errTry) {  
 lastError = errTry;  
 }  
 }  
 if (!data) throw lastError || new Error("Network Error");  
 localStorage.setItem("currentUser", JSON.stringify({ id: data.user.id, name: data.user.name, email: data.user.email }));  
 alert("Signup successful! 📌");  
 navigate("/dashboard");  
 } catch (err) {  
 const status = err?.response?.status;  
 const message = err?.response?.data?.error || (status === 409 ? "Email already registered" : (err?.message || "Something went wrong. Please try again."));  
 alert(message);  
 }  
 } else {  
 alert("Please fill in all fields.");  
 }  
 };  
  
 return (  
 <div className="signup-container">  
 <h2>Sign Up</h2>  
 <form onSubmit={handleSubmit}>  
 <input  
 type="text"  
 name="name"  
 placeholder="Name"  
 value={formData.name}  
 onChange={handleChange}  
 />  
 <input  
 type="email"  
 name="email"  
 placeholder="Email"  
 value={formData.email}  
 onChange={handleChange}  
 />  
 <input  
 type="password"  
 name="password"  
 placeholder="Password"  
 value={formData.password}  
 onChange={handleChange}  
 onInput={handleChange}  
 autoComplete="new-password"  
 required  
 />  
 <button type="submit">Sign Up</button>  
 </form>  
 </div>  
 );  
};  
  
export default SignupPage;

================================================================================

## 18. Studentdetails

📂 File Path: C:\Users\Admin\React\Academix\src\components\StudentDetails.js

import React, { useState, useEffect } from "react";  
import { useNavigate } from "react-router-dom";  
import axios from "axios";  
  
const StudentDetails = () => {  
 const navigate = useNavigate();  
 const [students, setStudents] = useState([]);  
  
 // Fetch students from MongoDB  
 useEffect(() => {  
 const fetchStudents = async () => {  
 try {  
 const { data } = await axios.get("http://localhost:5000/api/users");  
 setStudents(data.map((u) => ({  
 id: u.\_id || u.id,  
 name: u.name,  
 email: u.email,  
 phone: u.phone,  
 restricted: u.restricted,  
 })));  
 } catch (err) {  
 console.error("Failed to load students", err);  
 }  
 };  
  
 fetchStudents();  
 }, []);  
  
 // Restrict/Unrestrict a student  
 const toggleRestriction = async (id, restricted) => {  
 const updatedStudents = students.map((student) =>  
 student.id === id ? { ...student, restricted: !restricted } : student  
 );  
 setStudents(updatedStudents);  
  
 try {  
 await axios.patch(`http://localhost:5000/api/users/${id}`, { restricted: !restricted });  
 } catch (error) {  
 console.error("Error updating user restriction:", error);  
 }  
 };  
  
 // Delete a student  
 const handleDeleteStudent = async (id) => {  
 if (window.confirm("Are you sure you want to delete this student?")) {  
 await axios.delete(`http://localhost:5000/api/users/${id}`);  
 setStudents(students.filter((student) => student.id !== id));  
 }  
 };  
  
 return (  
 <div className="min-h-screen bg-gray-900 text-white p-6">  
 {/\* Header \*/}  
 <h2 className="text-3xl font-bold text-blue-400">👨‍🎓 Student Details</h2>  
  
 {/\* Table \*/}  
 <div className="bg-gray-800 p-6 rounded-lg shadow-lg mt-4">  
 <table className="w-full border-collapse border border-gray-700">  
 <thead>  
 <tr className="bg-gray-800 text-white">  
 <th className="p-2 border">ID</th>  
 <th className="p-2 border">Name</th>  
 <th className="p-2 border">Email</th>  
 <th className="p-2 border">Phone</th>  
 <th className="p-2 border">Actions</th>  
 </tr>  
 </thead>  
 <tbody>  
 {students.length > 0 ? (  
 students.map((student) => (  
 <tr key={student.id} className="text-center border border-gray-700">  
 <td className="p-2 border">{student.id}</td>  
 <td className="p-2 border">{student.name || "N/A"}</td>  
 <td className="p-2 border">{student.email}</td>  
 <td className="p-2 border">{student.phone || "N/A"}</td>  
 <td className="p-2 border flex justify-center gap-2">  
 <button  
 onClick={() => toggleRestriction(student.id, student.restricted)}  
 className={`px-3 py-1 rounded-lg text-white ${  
 student.restricted ? "bg-green-600 hover:bg-green-700" : "bg-yellow-600 hover:bg-yellow-700"  
 } transition`}  
 >  
 {student.restricted ? "✅ Unrestrict" : "🚫 Restrict"}  
 </button>  
 <button  
 onClick={() => handleDeleteStudent(student.id)}  
 className="bg-red-600 text-white px-4 py-1 rounded-lg hover:bg-red-700 transition"  
 >  
 ❌ Delete  
 </button>  
 </td>  
 </tr>  
 ))  
 ) : (  
 <tr>  
 <td colSpan="5" className="p-4 text-gray-400">No students found.</td>  
 </tr>  
 )}  
 </tbody>  
 </table>  
 </div>  
 </div>  
 );  
};  
  
export default StudentDetails;

================================================================================

## 19. Authcontext

📂 File Path: C:\Users\Admin\React\Academix\src\components\authContext.js

import { createContext, useState, useEffect } from "react";  
  
export const AuthContext = createContext();  
  
export const AuthProvider = ({ children }) => {  
 const [currentUser, setCurrentUser] = useState(null);  
 const [role, setRole] = useState(null);  
  
 useEffect(() => {  
 try {  
 const raw = localStorage.getItem("currentUser");  
 if (raw) {  
 const parsed = JSON.parse(raw);  
 setCurrentUser(parsed);  
 setRole(parsed.role || "student");  
 }  
 } catch (\_) {}  
 }, []);  
  
 return (  
 <AuthContext.Provider value={{ currentUser, role }}>  
 {children}  
 </AuthContext.Provider>  
 );  
};

================================================================================

## 20. Loginpage

📂 File Path: C:\Users\Admin\React\Academix\src\components\loginpage.js

import React, { useState } from "react";  
import { useNavigate } from "react-router-dom";  
import axios from "axios";  
  
const LoginPage = () => {  
 const navigate = useNavigate();  
 const [email, setEmail] = useState("");  
 const [password, setPassword] = useState("");  
 const [error, setError] = useState("");  
  
 const handleLogin = async (e) => {  
 e.preventDefault();  
 setError("");  
 try {  
 // Simple email/password login: verify user exists; in real app add hashing + JWT  
 const { data } = await axios.get("http://localhost:5000/api/users");  
 const found = data.find((u) => u.email === email);  
 if (!found) {  
 setError("User not found. Please sign up.");  
 return;  
 }  
 // NOTE: Backend currently stores plaintext password; for demo, fetch user by email and compare via another endpoint ideally  
 // For now, simulate success if email matches. Extend backend with /api/login for real validation.  
 localStorage.setItem("currentUser", JSON.stringify({ id: found.\_id || found.id, name: found.name, email: found.email }));  
 navigate("/dashboard");  
 } catch (err) {  
 setError("Login failed. Try again.");  
 }  
 };  
  
 return (  
 <div className="min-h-screen flex items-center justify-center bg-gradient-to-br from-gray-900 via-gray-800 to-gray-700 text-white">  
 <div className="bg-gray-900 p-8 rounded-2xl shadow-xl w-96 backdrop-blur-md">  
 <h2 className="text-3xl font-extrabold text-center text-blue-400">Login</h2>  
 {error && <p className="text-red-500 text-center mt-2">{error}</p>}  
  
 <form onSubmit={handleLogin} className="space-y-4 mt-4">  
 <div>  
 <label className="block text-gray-300 font-semibold">Email</label>  
 <input  
 type="email"  
 value={email}  
 onChange={(e) => setEmail(e.target.value)}  
 required  
 className="w-full p-3 bg-gray-800 border border-gray-600 rounded-lg focus:ring-2 focus:ring-blue-500 text-white placeholder-gray-400"  
 placeholder="Enter your email"  
 />  
 </div>  
  
 <div>  
 <label className="block text-gray-300 font-semibold">Password</label>  
 <input  
 type="password"  
 value={password}  
 onChange={(e) => setPassword(e.target.value)}  
 required  
 className="w-full p-3 bg-gray-800 border border-gray-600 rounded-lg focus:ring-2 focus:ring-blue-500 text-white placeholder-gray-400"  
 placeholder="Enter your password"  
 />  
 </div>  
  
 <button  
 type="submit"  
 className="w-full bg-blue-600 text-white py-3 rounded-lg font-semibold hover:bg-blue-700 transition-transform transform hover:scale-105"  
 >  
 Login  
 </button>  
 </form>  
  
 {/\* Display Test Credentials for Easy Access \*/}  
   
 </div>  
 </div>  
 );  
};  
  
export default LoginPage;

================================================================================

## 21. Button

📂 File Path: C:\Users\Admin\React\Academix\src\components\ui\button.js

import React from "react";  
  
export const Button = ({ children, onClick }) => {  
 return (  
 <button onClick={onClick} className="px-4 py-2 bg-blue-500 text-white rounded">  
 {children}  
 </button>  
 );  
};

================================================================================

## 22. Card

📂 File Path: C:\Users\Admin\React\Academix\src\components\ui\card.js

import React from "react";  
  
export const Card = ({ children }) => {  
 return (  
 <div className="bg-white p-4 rounded-lg shadow-md">  
 {children}  
 </div>  
 );  
};  
  
export const CardContent = ({ children }) => {  
 return (  
 <div className="mt-2">  
 {children}  
 </div>  
 );  
};

================================================================================

## 23. Input

📂 File Path: C:\Users\Admin\React\Academix\src\components\ui\input.js

import React from "react";  
  
export const Input = ({ name, value, onChange, placeholder }) => {  
 return (  
 <input  
 name={name}  
 value={value}  
 onChange={onChange}  
 placeholder={placeholder}  
 className="p-2 border rounded w-full"  
 />  
 );  
};

================================================================================

## 24. Activity

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\Activity.js

import React, { useState } from "react";  
  
const Activity = () => {  
 const [collaborationRequests, setCollaborationRequests] = useState([  
 {  
 id: 1,  
 from: "Alex Johnson",  
 project: "Data Structures Project",  
 status: "pending",  
 timestamp: "2 hours ago"  
 },  
 {  
 id: 2,  
 from: "Sarah Wilson",  
 project: "Machine Learning Study Group",  
 status: "pending",  
 timestamp: "1 day ago"  
 }  
 ]);  
  
 const handleAcceptRequest = (requestId) => {  
 setCollaborationRequests(prev =>   
 prev.map(req =>   
 req.id === requestId   
 ? { ...req, status: "accepted" }  
 : req  
 )  
 );  
 };  
  
 const handleDeclineRequest = (requestId) => {  
 setCollaborationRequests(prev =>   
 prev.map(req =>   
 req.id === requestId   
 ? { ...req, status: "declined" }  
 : req  
 )  
 );  
 };  
  
 const getStatusBadge = (status) => {  
 switch (status) {  
 case "pending":  
 return "bg-gradient-to-r from-yellow-500 to-yellow-400 text-white text-xs px-3 py-1 rounded-full font-medium";  
 case "accepted":  
 return "bg-gradient-to-r from-green-500 to-green-400 text-white text-xs px-3 py-1 rounded-full font-medium";  
 case "declined":  
 return "bg-gradient-to-r from-red-500 to-red-400 text-white text-xs px-3 py-1 rounded-full font-medium";  
 default:  
 return "bg-gray-500 text-white text-xs px-3 py-1 rounded-full font-medium";  
 }  
 };  
  
 const getStatusText = (status) => {  
 switch (status) {  
 case "pending":  
 return "Pending";  
 case "accepted":  
 return "Accepted";  
 case "declined":  
 return "Declined";  
 default:  
 return "Unknown";  
 }  
 };  
  
 return (  
 <div className="bg-gray-900 bg-opacity-90 p-4 rounded-2xl shadow-lg backdrop-blur-md flex-1 w-full max-w-md overflow-y-auto">  
 <div className="space-y-2">  
 {collaborationRequests.map((request) => (  
 <div key={request.id} className="flex items-center gap-4 p-3 bg-gray-800 rounded-lg shadow-sm hover:bg-gray-700 transition duration-300">  
 <span className="text-2xl">👥</span>  
 <div className="flex-1">  
 <p className="font-semibold text-gray-200">Collaboration request</p>  
 <p className="text-sm text-gray-400">From {request.from}</p>  
 <p className="text-xs text-gray-500">{request.project}</p>  
 <p className="text-xs text-gray-500">{request.timestamp}</p>  
 </div>  
 <div className="flex flex-col items-end gap-2">  
 <span className={getStatusBadge(request.status)}>  
 {getStatusText(request.status)}  
 </span>  
 {request.status === "pending" && (  
 <div className="flex gap-1">  
 <button  
 onClick={() => handleAcceptRequest(request.id)}  
 className="bg-green-600 hover:bg-green-700 text-white text-xs px-2 py-1 rounded transition-colors"  
 >  
 Accept  
 </button>  
 <button  
 onClick={() => handleDeclineRequest(request.id)}  
 className="bg-red-600 hover:bg-red-700 text-white text-xs px-2 py-1 rounded transition-colors"  
 >  
 Decline  
 </button>  
 </div>  
 )}  
 </div>  
 </div>  
 ))}  
 </div>  
 </div>  
 );  
};  
  
export default Activity;

================================================================================

## 25. Card

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\Card.js

import React from "react";  
  
const Card = ({ title, count, icon }) => {  
 return (  
 <div className="bg-gradient-to-r from-gray-800 to-gray-900 text-white p-5 rounded-xl shadow-lg flex items-center gap-4 transition-transform transform hover:scale-105 duration-300">  
 <span className="text-3xl bg-gray-700 p-3 rounded-full">{icon}</span>  
 <div>  
 <h4 className="font-semibold text-lg">{title}</h4>  
 <p className="text-gray-300 text-sm">{count}</p>  
 </div>  
 </div>  
 );  
};  
  
export default Card;

================================================================================

## 26. Eventmanagement

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\EventManagement.js

import React, { useEffect, useState } from "react";  
import { Button } from "../components/ui/button";  
import { Input } from "../components/ui/input";  
import { toast } from "react-toastify";  
import "react-toastify/dist/ReactToastify.css";  
import axios from "axios";  
  
const formatDate = (inputDate) => {  
 const date = new Date(inputDate);  
 const day = String(date.getDate()).padStart(2, "0");  
 const month = String(date.getMonth() + 1).padStart(2, "0");  
 const year = date.getFullYear();  
 return `${day}-${month}-${year}`; // Standard Date Format: DD-MM-YYYY  
};  
  
const EventManagement = () => {  
 const [isFormVisible, setIsFormVisible] = useState(false);  
 const [events, setEvents] = useState([]);  
  
 const [newEvent, setNewEvent] = useState({  
 title: "",  
 description: "",  
 date: "",  
 time: "",  
 category: "",  
 speakers: "",  
 location: "",  
 audienceLevel: "General",  
 registrationLink: "",  
 });  
  
 const handleInputChange = (e) => {  
 const { name, value } = e.target;  
 setNewEvent({ ...newEvent, [name]: value });  
 };  
  
 const addEvent = () => {  
 if (!newEvent.title || !newEvent.date || !newEvent.time || !newEvent.location) {  
 toast.error("Please fill in all required fields!", { position: "top-right" });  
 return;  
 }  
  
 axios.post("http://localhost:5000/api/events", { ...newEvent, date: formatDate(newEvent.date) })  
 .then(({ data }) => {  
 setEvents([data, ...events]);  
 setNewEvent({  
 title: "",  
 description: "",  
 date: "",  
 time: "",  
 category: "",  
 speakers: "",  
 location: "",  
 audienceLevel: "General",  
 registrationLink: "",  
 });  
 setIsFormVisible(false);  
 toast.success("Event posted successfully! 🎉", { position: "top-right" });  
 })  
 .catch(() => toast.error("Failed to post event", { position: "top-right" }));  
 };  
  
 useEffect(() => {  
 axios.get("http://localhost:5000/api/events").then(({ data }) => setEvents(data)).catch(() => {});  
 }, []);  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-indigo-50 to-indigo-100 flex flex-col items-center p-8">  
 {/\* Header Section \*/}  
 <div className="w-full max-w-4xl flex justify-between items-center bg-white p-6 rounded-xl shadow-lg">  
 <h2 className="text-3xl font-bold text-indigo-700">📅 Events</h2>  
 <button  
 onClick={() => setIsFormVisible(!isFormVisible)}  
 className="px-5 py-3 bg-indigo-600 text-white font-semibold rounded-xl shadow-md hover:bg-indigo-700 transition"  
 >  
 {isFormVisible ? "Close Form" : "➕ Post an Event"}  
 </button>  
 </div>  
  
 {/\* Display Available Events \*/}  
 <div className="w-full max-w-4xl mt-8">  
 <h3 className="text-2xl font-semibold text-indigo-700 mb-4">Upcoming Events</h3>  
 <div className="grid grid-cols-1 md:grid-cols-2 gap-6">  
 {events.map((event) => (  
 <div  
 key={event.\_id}  
 className="bg-white p-6 rounded-xl shadow-lg border-l-4 border-indigo-500 transform hover:scale-105 transition"  
 >  
 <h4 className="text-xl font-semibold text-gray-800">{event.title}</h4>  
 <p className="text-gray-600 mt-2">{event.description}</p>  
 <p className="text-gray-700"><strong>📅 Date:</strong> {event.date}</p>  
 <p className="text-gray-700"><strong>⏰ Time:</strong> {event.time}</p>  
 <p className="text-gray-700"><strong>📌 Category:</strong> {event.category || "N/A"}</p>  
 <p className="text-gray-700"><strong>🎤 Speakers:</strong> {event.speakers || "N/A"}</p>  
 <p className="text-gray-700"><strong>📍 Location:</strong> {event.location}</p>  
 <p className="text-gray-700"><strong>🎯 Audience Level:</strong> {event.audienceLevel}</p>  
 {event.registrationLink && (  
 <p className="text-blue-600 font-semibold">  
 <a href={event.registrationLink} target="\_blank" rel="noopener noreferrer">  
 🔗 Register Here  
 </a>  
 </p>  
 )}  
 <button  
 onClick={() => toast.info(`Organizer details sent!`, { position: "top-right" })}  
 className="mt-3 px-4 py-2 bg-green-600 text-white font-semibold rounded-lg hover:bg-green-700 transition"  
 >  
 🤝 Connect with Organizer  
 </button>  
 <button  
 onClick={() => axios.delete(`http://localhost:5000/api/events/${event.\_id}`).then(() => setEvents(events.filter(e => e.\_id !== event.\_id)))}  
 className="mt-3 ml-2 px-4 py-2 bg-red-600 text-white font-semibold rounded-lg hover:bg-red-700 transition"  
 >  
 🗑 Delete  
 </button>  
 </div>  
 ))}  
 </div>  
 </div>  
  
 {/\* Event Posting Form \*/}  
 {isFormVisible && (  
 <div className="w-full max-w-3xl bg-white p-6 mt-8 rounded-xl shadow-lg animate-fadeIn">  
 <h3 className="text-2xl font-semibold text-indigo-700 mb-4">📌 Post a New Event</h3>  
  
 {["title", "description", "category", "speakers", "location", "registrationLink"].map((field, index) => (  
 <div key={index} className="mb-4">  
 <label className="block text-gray-700 font-semibold capitalize">{field.replace(/([A-Z])/g, " $1")}</label>  
 <Input name={field} value={newEvent[field]} onChange={handleInputChange} placeholder={`Enter ${field}...`} />  
 </div>  
 ))}  
  
 {/\* Date & Time Inputs \*/}  
 <div className="grid grid-cols-2 gap-4 mb-4">  
 <div>  
 <label className="block text-gray-700 font-semibold">📅 Event Date:</label>  
 <Input name="date" type="date" value={newEvent.date} onChange={handleInputChange} />  
 </div>  
 <div>  
 <label className="block text-gray-700 font-semibold">⏰ Event Time:</label>  
 <Input name="time" type="time" value={newEvent.time} onChange={handleInputChange} />  
 </div>  
 </div>  
  
 {/\* Audience Level \*/}  
 <label className="block text-gray-700 font-semibold">🎯 Audience Level:</label>  
 <select name="audienceLevel" value={newEvent.audienceLevel} onChange={handleInputChange} className="w-full p-3 border border-gray-300 rounded-lg mb-4">  
 <option value="General">General</option>  
 <option value="Beginners">Beginners</option>  
 <option value="Intermediate">Intermediate</option>  
 <option value="Advanced">Advanced</option>  
 </select>  
  
 <Button onClick={addEvent} className="w-full py-3 bg-indigo-600 text-white font-semibold rounded-lg hover:bg-indigo-700 transition">  
 🚀 Post Event  
 </Button>  
 </div>  
 )}  
 </div>  
 );  
};  
  
export default EventManagement;

================================================================================

## 27. Peers

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\Peers.js

import React, { useEffect, useState } from "react";  
import \* as tf from "@tensorflow/tfjs";  
import axios from "axios";  
  
// Encode user skills and interests into a feature vector  
const encodeUser = (user, allSkills, allInterests) => {  
 const skillVector = allSkills.map((skill) =>  
 user.skills.includes(skill) ? 1 : 0  
 );  
 const interestVector = allInterests.map((interest) =>  
 user.interests.includes(interest) ? 1 : 0  
 );  
 return tf.tensor1d([...skillVector, ...interestVector]);  
};  
  
// Compute cosine similarity between two vectors  
const cosineSimilarity = (vecA, vecB) => {  
 const dotProduct = tf.sum(tf.mul(vecA, vecB)).dataSync()[0];  
 const normA = tf.norm(vecA).dataSync()[0];  
 const normB = tf.norm(vecB).dataSync()[0];  
 return dotProduct / (normA \* normB);  
};  
  
// Compute Jaccard similarity for interests  
const jaccardSimilarity = (setA, setB) => {  
 const intersection = setA.filter((value) => setB.includes(value)).length;  
 const union = new Set([...setA, ...setB]).size;  
 return intersection / union;  
};  
  
// Recommend peers based on similarity  
const recommendPeers = (currentUser, users) => {  
 const allSkills = [...new Set(users.flatMap((user) => user.skills))];  
 const allInterests = [...new Set(users.flatMap((user) => user.interests))];  
 const currentUserVector = encodeUser(currentUser, allSkills, allInterests);  
  
 return users  
 .filter((user) => user.id !== currentUser.id) // Exclude self  
 .map((user) => {  
 const userVector = encodeUser(user, allSkills, allInterests);  
 const skillSimilarity = cosineSimilarity(currentUserVector, userVector);  
 const interestSimilarity = jaccardSimilarity(  
 currentUser.interests,  
 user.interests  
 );  
  
 // Weighted scoring: 70% Skills + 30% Interests  
 const finalScore = 0.7 \* skillSimilarity + 0.3 \* interestSimilarity;  
 return { ...user, similarity: finalScore };  
 })  
 .sort((a, b) => b.similarity - a.similarity) // Sort by highest similarity  
 .slice(0, 5); // Get top 5 recommendations  
};  
  
const Peers = ({ currentUserId }) => {  
 const [users, setUsers] = useState([]);  
 const [currentUser, setCurrentUser] = useState(null);  
 const [recommendedPeers, setRecommendedPeers] = useState([]);  
 const [loading, setLoading] = useState(true);  
  
 // Fetch users and current user from the backend  
 useEffect(() => {  
 const fetchUsers = async () => {  
 try {  
 const response = await axios.get("/api/users"); // Replace with your API  
 setUsers(response.data);  
 const user = response.data.find((u) => u.id === currentUserId);  
 setCurrentUser(user);  
 } catch (error) {  
 console.error("Error fetching users:", error);  
 } finally {  
 setLoading(false);  
 }  
 };  
 fetchUsers();  
 }, [currentUserId]);  
  
 // Compute recommendations when data is loaded  
 useEffect(() => {  
 if (currentUser && users.length > 0) {  
 const recommendations = recommendPeers(currentUser, users);  
 setRecommendedPeers(recommendations);  
 }  
 }, [currentUser, users]);  
  
 return (  
 <div className="bg-white p-4 rounded-lg shadow-md flex-1 overflow-y-auto">  
 <h3 className="text-lg font-bold mb-3 text-gray-800">Recommended Connections</h3>  
 {loading ? (  
 <p className="text-gray-500">Loading recommendations...</p>  
 ) : recommendedPeers.length > 0 ? (  
 recommendedPeers.map((peer) => (  
 <div key={peer.id} className="flex items-center gap-4 border-b p-3">  
 <img  
 src={peer.profilePicture || "https://via.placeholder.com/50"}  
 alt="Profile"  
 className="rounded-full w-12 h-12 object-cover border-2 border-gray-300"  
 />  
 <div className="flex-1">  
 <p className="font-semibold text-gray-700">{peer.name}</p>  
 <p className="text-sm text-gray-500">  
 Skills: {peer.skills.join(", ")}  
 </p>  
 <p className="text-sm text-gray-500">  
 Interests: {peer.interests.join(", ")}  
 </p>  
 </div>  
 <button className="bg-blue-600 text-white px-4 py-2 rounded-lg hover:bg-blue-700 transition">  
 Connect  
 </button>  
 </div>  
 ))  
 ) : (  
 <p className="text-gray-500">No recommendations available</p>  
 )}  
 </div>  
 );  
};  
  
export default Peers;

================================================================================

## 28. Querieswritten

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\QueriesWritten.js

import React, { useEffect, useState } from 'react';  
import axios from 'axios';  
  
const QueriesWritten = () => {  
 // State to store queries and comments  
 const [queries, setQueries] = useState([]);  
 const [newQuery, setNewQuery] = useState('');  
 const [editingId, setEditingId] = useState(null);  
 const [commentInputs, setCommentInputs] = useState({});  
 const currentUser = (() => {  
 try { return JSON.parse(localStorage.getItem('currentUser')) } catch (\_) { return null }  
 })();  
  
 useEffect(() => {  
 const load = async () => {  
 try {  
 const { data } = await axios.get('http://localhost:5000/api/queries');  
 setQueries(data);  
 } catch (e) {}  
 };  
 load();  
 }, []);  
  
 // Function to add or update a query  
 const handleSubmit = async (e) => {  
 e.preventDefault();  
  
 if (newQuery.trim() === '' || !currentUser?.id) return;  
  
 try {  
 if (editingId) {  
 const { data } = await axios.patch(`http://localhost:5000/api/queries/${editingId}`, { content: newQuery });  
 setQueries(queries.map(q => q.\_id === editingId ? data : q));  
 setEditingId(null);  
 } else {  
 const { data } = await axios.post('http://localhost:5000/api/queries', { authorId: currentUser.id, content: newQuery });  
 setQueries([data, ...queries]);  
 }  
 setNewQuery('');  
 } catch (\_) {}  
 };  
  
 // Function to delete a query  
 const handleDelete = async (id) => {  
 try {  
 await axios.delete(`http://localhost:5000/api/queries/${id}`);  
 setQueries(queries.filter(q => q.\_id !== id));  
 } catch (\_) {}  
 };  
  
 // Function to set query for editing  
 const handleEdit = (q) => {  
 setNewQuery(q.content);  
 setEditingId(q.\_id);  
 };  
  
 // Function to add a comment to a query  
 const handleAddComment = async (id) => {  
 const value = (commentInputs[id] || '').trim();  
 if (!value || !currentUser?.id) return;  
 try {  
 const { data } = await axios.post(`http://localhost:5000/api/queries/${id}/comments`, { authorId: currentUser.id, content: value });  
 setQueries(queries.map(q => q.\_id === id ? data : q));  
 setCommentInputs({ ...commentInputs, [id]: '' });  
 } catch (\_) {}  
 };  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-blue-50 to-blue-100 flex justify-center items-center p-6">  
 <div className="w-full max-w-2xl bg-white shadow-lg rounded-2xl p-6">  
 <h2 className="text-3xl font-extrabold text-blue-700 text-center">Queries Hub</h2>  
 <p className="text-center text-gray-500 mb-6">Submit, edit, and manage your queries.</p>  
  
 {/\* Form for submitting/updating queries \*/}  
 <form onSubmit={handleSubmit} className="flex flex-col items-center space-y-4">  
 <input  
 type="text"  
 value={newQuery}  
 onChange={(e) => setNewQuery(e.target.value)}  
 placeholder="Write your query..."  
 className="w-full p-3 border border-gray-300 rounded-xl focus:outline-none focus:ring-2 focus:ring-blue-500 text-gray-700"  
 />  
 <button  
 type="submit"  
 className={`w-full px-5 py-3 font-semibold rounded-xl transition shadow-md ${  
 editingId !== null ? 'bg-yellow-500 hover:bg-yellow-600' : 'bg-blue-600 hover:bg-blue-700'  
 } text-white`}  
 >  
 {editingId !== null ? 'Update Query' : 'Post Query'}  
 </button>  
 </form>  
  
 {/\* Display the list of queries \*/}  
 <div className="mt-6 space-y-4">  
 {queries.length > 0 ? (  
 queries.map((query, index) => (  
 <div  
 key={index}  
 className="p-4 bg-white border-l-4 border-blue-600 shadow-md rounded-lg"  
 >  
 {/\* Query Section \*/}  
 <div className="flex items-center gap-4">  
 <div className="w-10 h-10 flex items-center justify-center bg-blue-500 text-white font-bold rounded-full">  
 {index + 1}  
 </div>  
 <p className="text-gray-800 flex-1">{query.content}</p>  
 <button  
 onClick={() => handleEdit(query)}  
 className="px-3 py-1 bg-yellow-500 text-white rounded-md hover:bg-yellow-600 transition"  
 >  
 ✏️ Edit  
 </button>  
 <button  
 onClick={() => handleDelete(query.\_id)}  
 className="px-3 py-1 bg-red-600 text-white rounded-md hover:bg-red-700 transition"  
 >  
 🗑 Delete  
 </button>  
 </div>  
  
 {/\* Comments Section \*/}  
 <div className="mt-4">  
 <h3 className="text-gray-700 font-semibold">💬 Answers:</h3>  
 <div className="space-y-2">  
 {query.comments?.length > 0 ? (  
 query.comments.map((c, ci) => (  
 <p key={ci} className="text-gray-600 bg-gray-100 p-2 rounded-md">  
 {c.content}  
 </p>  
 ))  
 ) : (  
 <p className="text-gray-400 text-sm">No answers yet. Be the first to reply!</p>  
 )}  
 </div>  
  
 {/\* Comment Input \*/}  
 <div className="mt-2 flex items-center gap-2">  
 <input  
 type="text"  
 placeholder="Write an answer..."  
 className="flex-1 p-2 border border-gray-300 rounded-md focus:outline-none focus:ring-2 focus:ring-blue-400"  
 value={commentInputs[query.\_id] || ''}  
 onChange={(e) => setCommentInputs({ ...commentInputs, [query.\_id]: e.target.value })}  
 onKeyDown={(e) => { if (e.key === 'Enter') handleAddComment(query.\_id) }}  
 />  
 <button  
 onClick={() => handleAddComment(query.\_id)}  
 className="px-3 py-1 bg-green-600 text-white rounded-md hover:bg-green-700 transition"  
 >  
 ➕ Reply  
 </button>  
 </div>  
 </div>  
 </div>  
 ))  
 ) : (  
 <p className="text-center text-gray-400 text-lg">  
 No queries posted yet. Start the discussion!  
 </p>  
 )}  
 </div>  
 </div>  
 </div>  
 );  
};  
  
export default QueriesWritten;

================================================================================

## 29. Accountmanagement

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\accountmanagement.js

import React, { useState } from "react";  
  
const AccountManagement = () => {  
 const [twoStepVerification, setTwoStepVerification] = useState(false);  
 const [multiStepVerification, setMultiStepVerification] = useState(false);  
   
 const [loginSessions, setLoginSessions] = useState([  
 { id: 1, device: "Windows PC", location: "New York, USA", lastLogin: "2025-02-01 10:30 AM" },  
 { id: 2, device: "Android Phone", location: "Los Angeles, USA", lastLogin: "2025-02-03 8:15 PM" },  
 { id: 3, device: "MacBook Pro", location: "London, UK", lastLogin: "2025-02-02 5:00 PM" },  
 ]);  
  
 const handleLogout = (id) => {  
 setLoginSessions(loginSessions.filter(session => session.id !== id));  
 alert("Logged out from the selected device.");  
 };  
  
 const handleLogoutAll = () => {  
 setLoginSessions([]);  
 alert("Logged out from all other devices.");  
 };  
  
 return (  
 <div className="p-6 bg-gray-100 min-h-screen">  
 <h2 className="text-2xl font-bold mb-4">Account Management</h2>  
  
 {/\* Login Sessions \*/}  
 <div className="bg-white p-4 rounded-lg shadow-md mb-6">  
 <h3 className="text-lg font-semibold mb-2">Active Login Sessions</h3>  
 <ul>  
 {loginSessions.length > 0 ? (  
 loginSessions.map((session) => (  
 <li key={session.id} className="border-b py-3 flex justify-between items-center">  
 <div>  
 <p className="text-gray-700"><strong>Device:</strong> {session.device}</p>  
 <p className="text-gray-500"><strong>Location:</strong> {session.location}</p>  
 <p className="text-gray-500"><strong>Last Login:</strong> {session.lastLogin}</p>  
 </div>  
 <button   
 onClick={() => handleLogout(session.id)}   
 className="bg-red-500 text-white px-3 py-1 rounded-lg hover:bg-red-600 transition"  
 >  
 Logout  
 </button>  
 </li>  
 ))  
 ) : (  
 <p className="text-gray-500">No active sessions.</p>  
 )}  
 </ul>  
 {loginSessions.length > 1 && (  
 <button   
 onClick={handleLogoutAll}   
 className="mt-4 w-full bg-red-600 text-white px-4 py-2 rounded-lg hover:bg-red-700 transition"  
 >  
 Logout from All Devices  
 </button>  
 )}  
 </div>  
  
 {/\* Security Settings \*/}  
 <div className="bg-white p-4 rounded-lg shadow-md">  
 <h3 className="text-lg font-semibold mb-2">Security Settings</h3>  
  
 {/\* Two-Step Verification Toggle \*/}  
 <div className="flex justify-between items-center border-b py-3">  
 <p className="text-gray-700">Enable 2-Step Verification</p>  
 <label className="relative inline-flex items-center cursor-pointer">  
 <input   
 type="checkbox"   
 className="sr-only peer"   
 checked={twoStepVerification}   
 onChange={() => setTwoStepVerification(!twoStepVerification)}  
 />  
 <div className="w-11 h-6 bg-gray-300 peer-focus:ring-4 peer-focus:ring-blue-300 rounded-full peer dark:bg-gray-700 peer-checked:after:translate-x-5 peer-checked:after:border-white after:content-[''] after:absolute after:top-1 after:left-1 after:bg-white after:border-gray-300 after:border after:rounded-full after:h-4 after:w-4 after:transition-all peer-checked:bg-blue-600"></div>  
 </label>  
 </div>  
  
 {/\* Multi-Step Verification Toggle \*/}  
 <div className="flex justify-between items-center border-b py-3">  
 <p className="text-gray-700">Enable Multi-Step Verification</p>  
 <label className="relative inline-flex items-center cursor-pointer">  
 <input   
 type="checkbox"   
 className="sr-only peer"   
 checked={multiStepVerification}   
 onChange={() => setMultiStepVerification(!multiStepVerification)}  
 />  
 <div className="w-11 h-6 bg-gray-300 peer-focus:ring-4 peer-focus:ring-blue-300 rounded-full peer dark:bg-gray-700 peer-checked:after:translate-x-5 peer-checked:after:border-white after:content-[''] after:absolute after:top-1 after:left-1 after:bg-white after:border-gray-300 after:border after:rounded-full after:h-4 after:w-4 after:transition-all peer-checked:bg-blue-600"></div>  
 </label>  
 </div>  
 </div>  
 </div>  
 );  
};  
  
export default AccountManagement;

================================================================================

## 30. Activecollabrations

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\activecollabrations.js

import React, { useState, useEffect } from "react";  
import { useNavigate } from "react-router-dom";  
  
const ActiveCollaborations = () => {  
 const navigate = useNavigate(); // Hook for navigation  
  
 // Sample data for Study Groups and Projects (You may fetch this from an API)  
 const [studyGroups, setStudyGroups] = useState([  
 {  
 id: 1,  
 title: "Math Study Group",  
 description: "Collaborate with peers to solve math problems.",  
 level: "Beginner",  
 contactEmail: "mathgroup@example.com",  
 },  
 {  
 id: 2,  
 title: "Science Study Group",  
 description: "Learn and discuss scientific concepts.",  
 level: "Intermediate",  
 contactEmail: "sciencegroup@example.com",  
 },  
 ]);  
  
 const [projects, setProjects] = useState([  
 {  
 id: 1,  
 title: "React App Development",  
 description: "Develop a simple web app using React.",  
 prerequisites: "Basic HTML, CSS, and JavaScript",  
 skills: "React, JavaScript",  
 level: "Beginner",  
 contactEmail: "reactproject@example.com",  
 },  
 {  
 id: 2,  
 title: "Data Science Project",  
 description: "Analyze data and build machine learning models.",  
 prerequisites: "Python, Statistics",  
 skills: "Python, Machine Learning",  
 level: "Hard",  
 contactEmail: "datascienceproject@example.com",  
 },  
 ]);  
  
 // Combine Study Groups and Projects into a single list of active collaborations  
 const activeCollaborations = [...studyGroups, ...projects];  
  
 // Handle delete collaboration (both from Study Groups and Projects)  
 const handleDelete = (id) => {  
 setStudyGroups(studyGroups.filter(group => group.id !== id)); // Delete from Study Groups  
 setProjects(projects.filter(project => project.id !== id)); // Delete from Projects  
 };  
  
 return (  
 <div className="min-h-screen bg-gray-100 flex flex-col items-center p-6">  
 {/\* Header \*/}  
 <h2 className="text-2xl font-bold text-gray-800 mb-6">Active Collaborations</h2>  
  
 {/\* Active Collaborations List \*/}  
 <div className="w-full max-w-3xl bg-white p-6 rounded-lg shadow-md">  
 <h3 className="text-xl font-semibold mb-4">Collaboration List</h3>  
 <ul>  
 {activeCollaborations.length > 0 ? (  
 activeCollaborations.map((collaboration) => (  
 <li key={collaboration.id} className="border-b py-4">  
 <h4 className="font-semibold">{collaboration.title}</h4>  
 <p>{collaboration.description}</p>  
 <p><strong>Level:</strong> {collaboration.level}</p>  
 <p><strong>Email:</strong> {collaboration.contactEmail}</p>  
 <button  
 onClick={() => handleDelete(collaboration.id)}  
 className="bg-red-600 text-white px-4 py-2 rounded-lg mt-2 hover:bg-red-700"  
 >  
 Delete Collaboration  
 </button>  
 </li>  
 ))  
 ) : (  
 <p>No active collaborations available at the moment.</p>  
 )}  
 </ul>  
 </div>  
  
 {/\* Button to add new collaboration (optional) \*/}  
 <button  
 onClick={() => navigate("/dashboard")}  
 className="bg-blue-600 text-white px-4 py-2 rounded-lg mt-6 hover:bg-blue-700"  
 >  
 Go Back to Dashboard  
 </button>  
 </div>  
 );  
};  
  
export default ActiveCollaborations;

================================================================================

## 31. Profilesettings

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\profilesettings.js

import React, { useState } from "react";  
  
const ProfileSettings = () => {  
 const [profile, setProfile] = useState({  
 name: "",  
 email: "",  
 username: "",  
 bio: "",  
 profilePic: null, // For storing the uploaded profile picture  
 });  
  
 const handleChange = (e) => {  
 const { name, value } = e.target;  
 setProfile({ ...profile, [name]: value });  
 };  
  
 const handleFileChange = (e) => {  
 const file = e.target.files[0];  
 setProfile({ ...profile, profilePic: file });  
 };  
  
 const handleSubmit = (e) => {  
 e.preventDefault();  
 console.log("Updated Profile:", profile);  
 alert("Profile updated successfully!");  
 };  
  
 return (  
 <div className="max-w-lg mx-auto p-6 bg-white shadow-lg rounded-lg">  
 <h2 className="text-2xl font-bold text-gray-800 mb-6 text-center">Profile Settings</h2>  
  
 {/\* Profile Picture Upload \*/}  
 <div className="mb-6 text-center">  
 <label className="block font-semibold text-gray-700 mb-2">Profile Picture</label>  
 <input  
 type="file"  
 accept="image/\*"  
 onChange={handleFileChange}  
 className="block w-full text-sm text-gray-700 file:mr-4 file:py-2 file:px-4 file:rounded-lg file:border file:border-gray-300 file:bg-gray-100 file:text-blue-600 hover:file:bg-gray-200 transition"  
 />  
 {profile.profilePic && (  
 <img  
 src={URL.createObjectURL(profile.profilePic)}  
 alt="Profile Preview"  
 className="w-24 h-24 rounded-full mt-3 mx-auto border-2 border-gray-300 object-cover"  
 />  
 )}  
 </div>  
  
 <form onSubmit={handleSubmit} className="space-y-4">  
 {/\* Full Name \*/}  
 <div>  
 <label htmlFor="name" className="block font-semibold text-gray-700">Full Name</label>  
 <input  
 type="text"  
 id="name"  
 name="name"  
 value={profile.name}  
 onChange={handleChange}  
 className="w-full border border-gray-300 p-3 rounded-lg focus:ring-2 focus:ring-blue-400 focus:outline-none"  
 placeholder="Enter your full name"  
 />  
 </div>  
  
 {/\* Email \*/}  
 <div>  
 <label htmlFor="email" className="block font-semibold text-gray-700">Email</label>  
 <input  
 type="email"  
 id="email"  
 name="email"  
 value={profile.email}  
 onChange={handleChange}  
 className="w-full border border-gray-300 p-3 rounded-lg focus:ring-2 focus:ring-blue-400 focus:outline-none"  
 placeholder="Enter your email"  
 />  
 </div>  
  
 {/\* Username \*/}  
 <div>  
 <label htmlFor="username" className="block font-semibold text-gray-700">Username</label>  
 <input  
 type="text"  
 id="username"  
 name="username"  
 value={profile.username}  
 onChange={handleChange}  
 className="w-full border border-gray-300 p-3 rounded-lg focus:ring-2 focus:ring-blue-400 focus:outline-none"  
 placeholder="Enter your username"  
 />  
 </div>  
  
 {/\* Bio \*/}  
 <div>  
 <label htmlFor="bio" className="block font-semibold text-gray-700">Bio</label>  
 <textarea  
 id="bio"  
 name="bio"  
 value={profile.bio}  
 onChange={handleChange}  
 className="w-full border border-gray-300 p-3 rounded-lg h-24 resize-none focus:ring-2 focus:ring-blue-400 focus:outline-none"  
 placeholder="Tell something about yourself"  
 />  
 </div>  
  
 {/\* Submit Button \*/}  
 <button  
 type="submit"  
 className="w-full bg-blue-600 text-white py-3 rounded-lg font-semibold hover:bg-blue-700 transition"  
 >  
 Update Profile  
 </button>  
 </form>  
 </div>  
 );  
};  
  
export default ProfileSettings;

================================================================================

## 32. Project

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\project.js

import React, { useEffect, useState } from "react";  
import { toast } from "react-toastify";  
import "react-toastify/dist/ReactToastify.css";  
import axios from "axios";  
  
const ProjectsHub = () => {  
 const [formData, setFormData] = useState({  
 title: "",  
 description: "",  
 technologies: "",  
 startDate: "",  
 endDate: "",  
 level: "",  
 contactEmail: "",  
 });  
  
 const [isFormVisible, setIsFormVisible] = useState(false);  
 const [projects, setProjects] = useState([]);  
  
 // Calculate total duration in days  
 const calculateDuration = (startDate, endDate) => {  
 if (!startDate || !endDate) return "";  
 const start = new Date(startDate);  
 const end = new Date(endDate);  
 const diffTime = Math.abs(end - start);  
 return Math.ceil(diffTime / (1000 \* 60 \* 60 \* 24)) + " days";  
 };  
  
 // Handle input changes  
 const handleChange = (e) => {  
 setFormData({ ...formData, [e.target.name]: e.target.value });  
 };  
  
 // Handle form submission  
 const handleSubmit = (e) => {  
 e.preventDefault();  
  
 if (!formData.title || !formData.description || !formData.contactEmail || !formData.startDate || !formData.endDate) {  
 toast.error("Please fill in all required fields!", { position: "top-right" });  
 return;  
 }  
  
 axios.post("http://localhost:5000/api/projects", formData)  
 .then(({ data }) => {  
 setProjects([data, ...projects]);  
 toast.success("Project posted successfully! 🎉", { position: "top-right" });  
 setFormData({  
 title: "",  
 description: "",  
 technologies: "",  
 startDate: "",  
 endDate: "",  
 level: "",  
 contactEmail: "",  
 });  
 setIsFormVisible(false);  
 // Re-fetch to reflect any server-side defaults/transformations  
 axios.get("http://localhost:5000/api/projects").then(({ data: list }) => setProjects(list)).catch(() => {});  
 })  
 .catch(() => toast.error("Failed to post project", { position: "top-right" }));  
 };  
  
 // Handle deleting a project  
 const handleDelete = (id) => {  
 axios.delete(`http://localhost:5000/api/projects/${id}`)  
 .then(() => {  
 setProjects(projects.filter((project) => project.\_id !== id));  
 toast.info("Project deleted!", { position: "top-right" });  
 })  
 .catch(() => toast.error("Failed to delete", { position: "top-right" }));  
 };  
  
 useEffect(() => {  
 axios.get("http://localhost:5000/api/projects").then(({ data }) => setProjects(data)).catch(() => {});  
 }, []);  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-blue-50 to-blue-100 flex flex-col items-center p-8">  
 {/\* Header Section \*/}  
 <div className="w-full max-w-4xl flex justify-between items-center bg-white p-6 rounded-xl shadow-lg">  
 <h2 className="text-3xl font-bold text-blue-700">🚀 Projects Hub</h2>  
 <button  
 onClick={() => setIsFormVisible(!isFormVisible)}  
 className="px-5 py-3 bg-blue-600 text-white font-semibold rounded-xl shadow-md hover:bg-blue-700 transition"  
 >  
 {isFormVisible ? "Close Form" : "➕ Post a Project"}  
 </button>  
 </div>  
  
 {/\* Display Projects \*/}  
 <div className="w-full max-w-4xl mt-8">  
 <h3 className="text-2xl font-semibold text-blue-700 mb-4">Available Projects</h3>  
 <div className="grid grid-cols-1 md:grid-cols-2 gap-6">  
 {projects.map((project) => (  
 <div  
 key={project.\_id}  
 className="bg-white p-6 rounded-xl shadow-lg border-l-4 border-blue-500 transform hover:scale-105 transition"  
 >  
 <h4 className="text-xl font-semibold text-gray-800">{project.title}</h4>  
 <p className="text-gray-600 mt-2">{project.description}</p>  
 <p className="text-gray-700"><strong>Technologies:</strong> {project.technologies}</p>  
 <p className="text-gray-700"><strong>Level:</strong> {project.level}</p>  
 <p className="text-gray-700"><strong>Start Date:</strong> {project.startDate}</p>  
 <p className="text-gray-700"><strong>End Date:</strong> {project.endDate}</p>  
 <p className="text-gray-700 font-bold"><strong>Duration:</strong> {calculateDuration(project.startDate, project.endDate)}</p>  
 <button  
 onClick={() => handleDelete(project.\_id)}  
 className="mt-3 px-4 py-2 bg-red-600 text-white font-semibold rounded-lg hover:bg-red-700 transition"  
 >  
 🗑️ Delete  
 </button>  
 </div>  
 ))}  
 </div>  
 </div>  
  
 {/\* Project Form \*/}  
 {isFormVisible && (  
 <form  
 onSubmit={handleSubmit}  
 className="w-full max-w-3xl bg-white p-6 mt-8 rounded-xl shadow-lg transform transition-all animate-fadeIn"  
 >  
 <h3 className="text-2xl font-semibold text-blue-700 mb-4">📌 Post a New Project</h3>  
  
 {["title", "description", "technologies", "contactEmail"].map((field, index) => (  
 <div key={index} className="mb-4">  
 <label className="block text-gray-700 font-semibold capitalize">{field}</label>  
 <input  
 type={field === "contactEmail" ? "email" : "text"}  
 name={field}  
 value={formData[field]}  
 onChange={handleChange}  
 required  
 className="w-full p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 text-gray-700"  
 placeholder={`Enter ${field}...`}  
 />  
 </div>  
 ))}  
  
 {/\* Date Selection \*/}  
 {["startDate", "endDate"].map((dateField, index) => (  
 <div key={index} className="mb-4">  
 <label className="block text-gray-700 font-semibold">{dateField === "startDate" ? "Start Date" : "End Date"}</label>  
 <input  
 type="date"  
 name={dateField}  
 value={formData[dateField]}  
 onChange={handleChange}  
 required  
 className="w-full p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 text-gray-700"  
 />  
 </div>  
 ))}  
  
 {/\* Display Total Duration \*/}  
 {formData.startDate && formData.endDate && (  
 <p className="text-lg font-bold text-blue-700 mb-4">  
 ⏳ Total Duration: {calculateDuration(formData.startDate, formData.endDate)}  
 </p>  
 )}  
  
 <button  
 type="submit"  
 className="w-full px-5 py-3 bg-blue-600 text-white font-semibold rounded-xl shadow-md hover:bg-blue-700 transition"  
 >  
 🚀 Submit Project  
 </button>  
 </form>  
 )}  
 </div>  
 );  
};  
  
export default ProjectsHub;

================================================================================

## 33. Studygroup

📂 File Path: C:\Users\Admin\React\Academix\src\dashboard\studygroup.js

import React, { useEffect, useState } from "react";  
import { toast } from "react-toastify";  
import "react-toastify/dist/ReactToastify.css";  
import axios from "axios";  
  
const StudyGroupPage = () => {  
 const [formData, setFormData] = useState({  
 title: "",  
 description: "",  
 prerequisites: "",  
 skills: "",  
 level: "",  
 duration: "",  
 mode: "",  
 contactEmail: "",  
 });  
  
 const [isFormVisible, setIsFormVisible] = useState(false);  
 const [studyGroups, setStudyGroups] = useState([]);  
  
 // Handle input changes for form  
 const handleChange = (e) => {  
 setFormData({ ...formData, [e.target.name]: e.target.value });  
 };  
  
 // Handle form submission  
 const handleSubmit = (e) => {  
 e.preventDefault();  
  
 if (!formData.title || !formData.description || !formData.contactEmail) {  
 toast.error("Please fill in the required fields!", { position: "top-right" });  
 return;  
 }  
  
 axios.post("http://localhost:5000/api/study-groups", formData)  
 .then(({ data }) => {  
 setStudyGroups([data, ...studyGroups]);  
 toast.success("Study Group posted successfully! 🎉", { position: "top-right" });  
 setFormData({  
 title: "",  
 description: "",  
 prerequisites: "",  
 skills: "",  
 level: "",  
 duration: "",  
 mode: "",  
 contactEmail: "",  
 });  
 setIsFormVisible(false);  
 })  
 .catch(() => toast.error("Failed to post study group", { position: "top-right" }));  
 };  
  
 useEffect(() => {  
 axios.get("http://localhost:5000/api/study-groups").then(({ data }) => setStudyGroups(data)).catch(() => {});  
 }, []);  
  
 return (  
 <div className="min-h-screen bg-gradient-to-br from-indigo-50 to-indigo-100 flex flex-col items-center p-8">  
 {/\* Header Section \*/}  
 <div className="w-full max-w-4xl flex justify-between items-center bg-white p-6 rounded-xl shadow-lg">  
 <h2 className="text-3xl font-bold text-indigo-700">📚 Study Groups</h2>  
 <button  
 onClick={() => setIsFormVisible(!isFormVisible)}  
 className="px-5 py-3 bg-indigo-600 text-white font-semibold rounded-xl shadow-md hover:bg-indigo-700 transition"  
 >  
 {isFormVisible ? "Close Form" : "➕ Post a Study Group"}  
 </button>  
 </div>  
  
 {/\* Display Study Groups \*/}  
 <div className="w-full max-w-4xl mt-8">  
 <h3 className="text-2xl font-semibold text-indigo-700 mb-4">Available Study Groups</h3>  
 <div className="grid grid-cols-1 md:grid-cols-2 gap-6">  
 {studyGroups.map((studyGroup) => (  
 <div  
 key={studyGroup.\_id}  
 className="bg-white p-6 rounded-xl shadow-lg border-l-4 border-indigo-500 transform hover:scale-105 transition"  
 >  
 <h4 className="text-xl font-semibold text-gray-800">{studyGroup.title}</h4>  
 <p className="text-gray-600 mt-2">{studyGroup.description}</p>  
 <p className="text-gray-700 mt-2"><strong>Skills:</strong> {studyGroup.skills}</p>  
 <p className="text-gray-700"><strong>Level:</strong> {studyGroup.level}</p>  
 <p className="text-gray-700"><strong>Duration:</strong> {studyGroup.duration}</p>  
 <p className="text-gray-700"><strong>Mode:</strong> {studyGroup.mode}</p>  
 <button  
 onClick={() => toast.info(`Form sent to ${studyGroup.contactEmail}`, { position: "top-right" })}  
 className="mt-3 px-4 py-2 bg-indigo-600 text-white font-semibold rounded-lg hover:bg-indigo-700 transition"  
 >  
 🤝 Connect  
 </button>  
 <button  
 onClick={() => axios.delete(`http://localhost:5000/api/study-groups/${studyGroup.\_id}`).then(() => setStudyGroups(studyGroups.filter(s => s.\_id !== studyGroup.\_id)))}  
 className="mt-3 ml-2 px-4 py-2 bg-red-600 text-white font-semibold rounded-lg hover:bg-red-700 transition"  
 >  
 🗑 Delete  
 </button>  
 </div>  
 ))}  
 </div>  
 </div>  
  
 {/\* Study Group Form \*/}  
 {isFormVisible && (  
 <form  
 onSubmit={handleSubmit}  
 className="w-full max-w-3xl bg-white p-6 mt-8 rounded-xl shadow-lg transform transition-all animate-fadeIn"  
 >  
 <h3 className="text-2xl font-semibold text-indigo-700 mb-4">📌 Create a Study Group</h3>  
  
 {["title", "description", "prerequisites", "skills", "contactEmail"].map((field, index) => (  
 <div key={index} className="mb-4">  
 <label className="block text-gray-700 font-semibold capitalize">{field}</label>  
 <input  
 type={field === "contactEmail" ? "email" : "text"}  
 name={field}  
 value={formData[field]}  
 onChange={handleChange}  
 required  
 className="w-full p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 text-gray-700"  
 placeholder={`Enter ${field}...`}  
 />  
 </div>  
 ))}  
  
 {/\* Dropdowns \*/}  
 {[   
 { name: "level", options: ["Beginner", "Moderate", "Hard"] },  
 { name: "duration", options: ["4 weeks", "2 months"] },  
 { name: "mode", options: ["Online", "Offline"] },  
 ].map((selectField, index) => (  
 <div key={index} className="mb-4">  
 <label className="block text-gray-700 font-semibold capitalize">{selectField.name}</label>  
 <select  
 name={selectField.name}  
 value={formData[selectField.name]}  
 onChange={handleChange}  
 required  
 className="w-full p-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 text-gray-700"  
 >  
 <option value="">Select {selectField.name}</option>  
 {selectField.options.map((option, idx) => (  
 <option key={idx} value={option}>  
 {option}  
 </option>  
 ))}  
 </select>  
 </div>  
 ))}  
  
 <button  
 type="submit"  
 className="w-full px-5 py-3 bg-indigo-600 text-white font-semibold rounded-xl shadow-md hover:bg-indigo-700 transition"  
 >  
 🚀 Submit Study Group  
 </button>  
 </form>  
 )}  
 </div>  
 );  
};  
  
export default StudyGroupPage;

================================================================================