**Corporate safety system codeBase**

**Frontend:**

**main.jsx:**

import { createRoot } from 'react-dom/client'

import '@mantine/core/styles.css'

import App from './App.jsx'

import { MantineProvider } from '@mantine/core'

createRoot(document.getElementById('root')).render(

<MantineProvider>

<App/>

</MantineProvider>

)

**Index.css:**

:root {

font-family: system-ui, Avenir, Helvetica, Arial, sans-serif;

line-height: 1.5;

font-weight: 400;

color-scheme: light dark;

color: rgba(255, 255, 255, 0.87);

background-color: #242424;

font-synthesis: none;

text-rendering: optimizeLegibility;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

a {

font-weight: 500;

color: #646cff;

text-decoration: inherit;

}

a:hover {

color: #535bf2;

}

body {

margin: 0;

display: flex;

place-items: center;

min-width: 320px;

min-height: 100vh;

}

h1 {

font-size: 3.2em;

line-height: 1.1;

}

button {

border-radius: 8px;

border: 1px solid transparent;

padding: 0.6em 1.2em;

font-size: 1em;

font-weight: 500;

font-family: inherit;

background-color: #1a1a1a;

cursor: pointer;

transition: border-color 0.25s;

}

button:hover {

border-color: #646cff;

}

button:focus,

button:focus-visible {

outline: 4px auto -webkit-focus-ring-color;

}

@media (prefers-color-scheme: light) {

:root {

color: #213547;

background-color: #ffffff;

}

a:hover {

color: #747bff;

}

button {

background-color: #f9f9f9;

}

}

**App.jsx:**

import Login from './pages/Login'

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

import UserManagement from './pages/UserManagement';

import UserForm from './pages/UserForm';

import ProfilePage from './pages/ProfilePage';

import Manager from './pages/Manager';

import Associate from './pages/Associate';

import Executive from './pages/Executive';

import Admin from './pages/Admin';

import WeatherAlert from './pages/WeatherAlert';

import Userdetail from './pages/Userdetail';

function App() {

return (

<Router>

<Routes>

<Route path='/' element={<Login/>} />

<Route path='/user' element={<UserManagement/>} />

<Route path='/userform' element={<UserForm/>} />

<Route path='/profile' element={<ProfilePage/>} />

<Route path='/manager' element={<Manager/>} />

<Route path='/associate' element={<Associate/>} />

<Route path='/executive' element={<Executive/>} />

<Route path='/admin' element={<Admin/>} />

<Route path='/admin/:id' element={<Userdetail/>} />

<Route path='/alert' element={<WeatherAlert/>} />

</Routes>

</Router>

)

}

export default App

**App.css:**

#root {

max-width: 1280px;

margin: 0 auto;

padding: 2rem;

text-align: center;

}

.logo {

height: 6em;

padding: 1.5em;

will-change: filter;

transition: filter 300ms;

}

.logo:hover {

filter: drop-shadow(0 0 2em #646cffaa);

}

.logo.react:hover {

filter: drop-shadow(0 0 2em #61dafbaa);

}

@keyframes logo-spin {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

@media (prefers-reduced-motion: no-preference) {

a:nth-of-type(2) .logo {

animation: logo-spin infinite 20s linear;

}

}

.card {

padding: 2em;

}

.read-the-docs {

color: #888;

}

**Admin.jsx:**

import React, { useEffect, useState } from 'react'

import {Box, Button, Card, Flex, Table, Text} from '@mantine/core'

import axios from 'axios';

import { useNavigate } from 'react-router-dom';

import Incident from '../../components/Incident';

const Admin = () => {

const navigate = useNavigate();

const adduser = ()=>{

navigate('/userform')

}

const [data,setdata] = useState([]);

useEffect(()=>{

const userapi = async()=>{

try{

const response = await axios.get("http://localhost:3000/userdata");

setdata(response.data);

}

catch(error){

console.log(error.message);

}

}

userapi();

},[])

const row = data.map((item)=>(

<Table.Tr key={item.id} >

<Table.Td>{item.id}</Table.Td>

<Table.Td>{item.username}</Table.Td>

<Table.Td>{item.password}</Table.Td>

<Table.Td>{item.role}</Table.Td>

<Table.Td><Button onClick={()=>navigate(`/admin/${item.id}`)} >View</Button></Table.Td>

</Table.Tr>

))

return (

<Box align="center" justify="center" >

<Text mt={20} fw={700} align="center" >Admin Panel</Text>

<Incident />

<Box mx={50} mt={30}>

<Table>

<Table.Thead>

<Table.Tr >

<Table.Th>ID</Table.Th>

<Table.Th>Username</Table.Th>

<Table.Th>Password</Table.Th>

<Table.Th>Role</Table.Th>

<Table.Th>View</Table.Th>

</Table.Tr>

</Table.Thead>

<Table.Tbody>{row}</Table.Tbody>

</Table>

</Box>

<Flex align="center" justify="space-around" >

<Button mt={30} align="center" justify='center' onClick={adduser} >Add User</Button>

</Flex>

</Box>

)

}

export default Admin

**Associate.jsx:**

import React from 'react'

import {Box, Table} from '@mantine/core'

const Associate = () => {

return (

<Box mt={30} mx={50} >

<Table>

<Table.Thead>

<Table.Th>S.no</Table.Th>

<Table.Th>project Name</Table.Th>

<Table.Th>Duration</Table.Th>

<Table.Th>Team name</Table.Th>

<Table.Th>No of members</Table.Th>

<Table.Th>Status</Table.Th>

</Table.Thead>

<Table.Tr>

<Table.Td>1</Table.Td>

<Table.Td>Scorpio-x</Table.Td>

<Table.Td>3Months</Table.Td>

<Table.Td>Challenger Deep</Table.Td>

<Table.Td>8People</Table.Td>

<Table.Td>In Progress</Table.Td>

</Table.Tr>

</Table>

</Box>

)

}

export default Associate

**Executive.jsx:**

import React from 'react'

import {Box,Table} from '@mantine/core'

const Executive = () => {

return (

<Box>

<Table>

<Table.Thead>

<Table.Tr>

<Table.Th>S.no</Table.Th>

<Table.Th>Name</Table.Th>

<Table.Th>Role</Table.Th>

<Table.Th>Contact No</Table.Th>

<Table.Th>No of Teams</Table.Th>

</Table.Tr>

</Table.Thead>

<Table.Tbody>

<Table.Tr >

<Table.Td>1</Table.Td>

<Table.Td>Ramesh</Table.Td>

<Table.Td>General manager</Table.Td>

<Table.Td>9876543210</Table.Td>

<Table.Td>5</Table.Td>

</Table.Tr>

</Table.Tbody>

</Table>

</Box>

)

}

export default Executive

**Login.jsx:**

import { useState } from "react";

import { TextInput, PasswordInput, Button, Card, Title, Text, Select } from "@mantine/core";

import axios from "axios";

import { useNavigate } from "react-router-dom";

export default function Login() {

const [isAuthenticated,setisAuthenticated] = useState(false);

const [username,setusername] = useState("");

const [password,setpassword] = useState("");

const [role,setrole] = useState("admin");

const [error,seterror] = useState("");

const [responseData, setResponseData] = useState(""); // To store API response

const navigate = useNavigate();

const handleLogin = async (e) => {

e.preventDefault();

seterror("");

setResponseData(""); // Clear previous response

try {

const response = await axios.post("http://localhost:3000/login", {

username,

password,

role,

});

navigate(`/${response.data.role}`);

console.log("Login successful", response.data.role);

setResponseData(response.data.message); // Store success message

} catch (err) {

console.error("Login failed:", err.response?.data?.error || "Server error");

seterror(err.response?.data?.error || "Invalid credentials");

}

};

return (

<div style={{ display: "flex", justifyContent: "center", alignItems: "center", height: "100vh", background: "#f5f5f5" }}>

<Card shadow="sm" padding="lg" radius="md" withBorder style={{ width: 350 }}>

<Title order={2} align="center" mb="md">

Login

</Title>

<TextInput label='username' value={username} onChange={(event)=>setusername(event.target.value)} placeholder="enter username" />

<TextInput label='password' value={password} onChange={(event)=>setpassword(event.target.value)} placeholder="enter passoword" type="password" />

<Select label="role" value={role} onChange={(value) => setrole(value)} placeholder="select role" data={['associate','manager','executive','admin']} />

<Button my={20} onClick={handleLogin} >Login</Button>

</Card>

</div>

);

}

**Manager.jsx:**

import { Card, Table } from '@mantine/core'

import React from 'react'

const Manager = () => {

const data = [

{ sno: 1, project: 'AI Chatbot', teamname: 'Alpha Squad', teamlead: 'John Doe', status: 'In Progress' },

{ sno: 2, project: 'E-commerce Platform', teamname: 'Beta Team', teamlead: 'Jane Smith', status: 'Completed' },

{ sno: 3, project: 'Corporate Safety System', teamname: 'Gamma Force', teamlead: 'Alice Johnson', status: 'Pending' }

]

const row = data.map((item)=>(

<Table.Tr key={item.id} >

<Table.Td>{item.sno}</Table.Td>

<Table.Td>{item.project}</Table.Td>

<Table.Td>{item.teamname}</Table.Td>

<Table.Td>{item.teamlead}</Table.Td>

<Table.Td>{item.status}</Table.Td>

</Table.Tr>

))

return (

<Card>

<Table>

<Table.Thead>

<Table.Th>s.no</Table.Th>

<Table.Th>Project</Table.Th>

<Table.Th>Team name</Table.Th>

<Table.Th>Team Lead</Table.Th>

<Table.Th>Status</Table.Th>

</Table.Thead>

<Table.Tbody>

{row}

</Table.Tbody>

</Table>

</Card>

)

}

export default Manager

**Profilepage.jsx:**

import { useState } from "react";

import { TextInput, Button, Card } from "@mantine/core";

export default function ProfilePage() {

const [name, setName] = useState("John Doe");

const [email, setEmail] = useState("john@example.com");

return (

<Card shadow="sm" padding="lg">

<h2 className="text-lg font-bold mb-2">My Profile</h2>

<TextInput label="Name" value={name} onChange={(e) => setName(e.target.value)} required />

<TextInput label="Email" value={email} onChange={(e) => setEmail(e.target.value)} required mt="md" />

<Button fullWidth mt="md">Update Profile</Button>

</Card>

);

}

**Userdetail.jsx:**

import React, { useEffect, useState } from 'react';

import { Box, Button, Card, Center, Flex, Loader, PasswordInput, Select, TextInput, Notification } from '@mantine/core';

import { useNavigate, useParams } from "react-router-dom";

import axios from 'axios';

const Userdetail = () => {

const navigate = useNavigate();

const { id } = useParams();

const [employee, setEmployee] = useState(null);

const [loading, setLoading] = useState(true);

const [updating, setUpdating] = useState(false);

const [username, setUsername] = useState("");

const [password, setPassword] = useState("");

const [role, setRole] = useState("");

const [message, setMessage] = useState("");

const [error, setError] = useState(""); // Added error message state

useEffect(() => {

const fetchUser = async () => {

try {

const response = await axios.get(`http://localhost:3000/user/${id}`);

setEmployee(response.data);

setUsername(response.data.username);

setPassword(response.data.password);

setRole(response.data.role);

setLoading(false);

} catch (error) {

console.error("Error fetching user:", error);

setError("Failed to fetch user data.");

setLoading(false);

}

};

fetchUser();

}, [id]);

const deleteUser = async () => {

const confirmDelete = window.confirm("Are you sure you want to delete this user?");

if (!confirmDelete) return;

try {

await axios.delete(`http://localhost:3000/delete/${id}`);

alert("User deleted successfully!");

navigate('/admin'); // Redirect to admin page after deletion

} catch (error) {

console.error("Failed to delete user:", error);

alert("Failed to delete user. Please try again.");

}

};

const editUser = async () => {

if (!username || !password || !role) {

alert("Please fill in all fields!");

return;

}

const userData = {

id: parseInt(id),

username,

password,

role

};

setUpdating(true);

try {

const response = await axios.put(`http://localhost:3000/updateuser/`, userData);

console.log("User updated:", response.data);

setMessage("User updated successfully!");

alert("Updated successfully!");

setTimeout(() => setMessage(""), 3000); // Clear success message after 3s

} catch (error) {

console.error("Failed to update user:", error);

setError("Failed to update user. Please try again.");

} finally {

setUpdating(false);

}

};

if (loading) return <Loader />;

return (

<Card mx={250} mt={100} shadow="sm" padding="lg">

<Center><b>Employee ID:</b> {employee?.id}</Center>

{message && (

<Notification color="green" title="Success" onClose={() => setMessage('')}>

{message}

</Notification>

)}

{error && (

<Notification color="red" title="Error" onClose={() => setError('')}>

{error}

</Notification>

)}

<Box>

<p style={{ fontSize: '15px', fontWeight: '600' }}>Username: <span style={{ fontWeight: '400' }}>{employee?.username}</span></p>

<p style={{ fontSize: '15px', fontWeight: '600' }}>Password: <span style={{ fontWeight: '400' }}>{employee?.password}</span></p>

<p style={{ fontSize: '15px', fontWeight: '600' }}>Role: <span style={{ fontWeight: '400' }}>{employee?.role}</span></p>

</Box>

<TextInput

label="Username"

value={username}

onChange={(e) => setUsername(e.target.value)}

required

mt="md"

/>

<PasswordInput

label="Password"

placeholder="Enter password"

value={password}

onChange={(event) => setPassword(event.target.value)}

required

mt="md"

/>

<Select

label="Role"

value={role}

onChange={setRole}

data={["Admin", "Executive", "Manager", "Associate"]}

required

mt="md"

/>

<Flex align='center' gap={5} justify='space-around' mt="md">

<Button onClick={editUser} w={300} fullWidth disabled={updating}>

{updating ? <Loader size="xs" color="white" /> : "Edit User"}

</Button>

<Button onClick={deleteUser} w={300} fullWidth mt="md" color="red">

Delete User

</Button>

<Button w={300} onClick={() => navigate('/admin')} fullWidth mt="md">

Back to Admin

</Button>

</Flex>

</Card>

);

};

export default Userdetail;

**Userform.jsx:**

import { useState } from "react";

import { TextInput, Select, Button, Card, PasswordInput, Box, Flex } from "@mantine/core";

import { useNavigate } from "react-router-dom";

export default function UserForm() {

const [username, setusername] = useState("");

const [password, setpassword] = useState("");

const [role, setRole] = useState("");

const navigate = useNavigate();

const backtoadmin = ()=>{

navigate('/admin');

}

const handleSubmit = async() => {

const userData = {username,password,role};

try {

const response = await fetch("http://localhost:3000/api/signup",{

method : "POST",

headers :{"Content-Type":"application/json"},

body:JSON.stringify(userData)

});

if(response.ok) {

const result = await response.json();

console.log("User registered:", result);

setusername("");

setpassword("");

setRole("");

alert("user added succesfully");

}

else{

console.error("signup failed")

}

}

catch(error){

console.log(error);

}

}

return (

<Card mx={250} mt={100} shadow="sm" padding="lg">

<h2 className="text-lg font-bold mb-2">Add user</h2>

<TextInput label="Name" value={username} onChange={(e) => setusername(e.target.value)} required />

<PasswordInput

label="password"

placeholder="enter password"

value = {password}

onChange={(event)=>setpassword(event.target.value)}

/>

<Select

label="Role"

value={role}

onChange={setRole}

data={["Admin", "Executive", "Manager","Associate"]}

mt="md"

/>

<p>{username}{password}{role}</p>

<Flex align='center' justify='space-around' >

<Button w={300} onClick={handleSubmit} fullWidth mt="md">

Add user

</Button>

<Button w={300} onClick={backtoadmin} >Back to Admin</Button>

</Flex>

</Card>

);

}

**Usermanagement.jsx:**

import { useState } from "react";

import { Table, Button, TextInput, Select } from "@mantine/core";

const usersData = [

{ id: 1, name: "John Doe", email: "john@example.com", role: "Admin" },

{ id: 2, name: "Alice Smith", email: "alice@example.com", role: "Employee" },

];

export default function UserManagement() {

const [users, setUsers] = useState(usersData);

const [search, setSearch] = useState("");

const filteredUsers = users.filter(user =>

user.name.toLowerCase().includes(search.toLowerCase())

);

return (

<div className="p-8" style={{display:'flex',flexDirection:'column',width:'50%',alignItems:'center',justifyContent:'center'}} >

<h1 className="text-2xl font-bold mb-4">User Management</h1>

<TextInput

placeholder="Search users..."

value={search}

onChange={(e) => setSearch(e.target.value)}

mb="md"

/>

<Table striped highlightOnHover>

<thead>

<tr>

<th>Name</th>

<th>Email</th>

<th>Role</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{filteredUsers.map((user) => (

<tr key={user.id}>

<td>{user.name}</td>

<td>{user.email}</td>

<td>{user.role}</td>

<td>

<Button color="blue" size="xs" mr="sm">Edit</Button>

<Button color="red" size="xs">Delete</Button>

</td>

</tr>

))}

</tbody>

</Table>

</div>

);

}

**Weatheralert.jsx:**

import React, { useEffect, useState } from "react";

import axios from "axios";

const WeatherAlert = () => {

const [weather, setWeather] = useState(null);

useEffect(() => {

const fetchWeather = async () => {

try {

const response = await fetch("http://localhost:3000/api/weather");

const data = await response.json();

setWeather(data);

checkWeatherConditions(data);

} catch (error) {

console.error("Error fetching weather data:", error);

}

};

fetchWeather();

const interval = setInterval(fetchWeather, 60000); // Refresh every 1 minute

return () => clearInterval(interval);

}, []);

const checkWeatherConditions = (data) => {

if (!data || !data.current) return;

const severeConditions = ["Thunderstorm", "Heavy Rain", "Storm", "Cyclone"];

if (severeConditions.includes(data.current.condition.text)) {

sendNotification("Severe Weather Alert!", `Current weather: ${data.current.condition.text}`);

}

};

const sendNotification = (title, message) => {

if (Notification.permission === "granted") {

new Notification(title, { body: message });

} else if (Notification.permission !== "denied") {

Notification.requestPermission().then((permission) => {

if (permission === "granted") {

new Notification(title, { body: message });

}

});

}

};

return (

<div>

<h2>Chennai Weather</h2>

{weather && weather.current ? (

<div>

<img src={`https:${weather.current.condition.icon}`} alt={weather.current.condition.text} />

<p><strong>Condition:</strong> {weather.current.condition.text}</p>

<p><strong>Temperature:</strong> {weather.current.temp\_c}°C / {weather.current.temp\_f}°F</p>

<p><strong>Feels Like:</strong> {weather.current.feelslike\_c}°C / {weather.current.feelslike\_f}°F</p>

<p><strong>Wind:</strong> {weather.current.wind\_kph} kph ({weather.current.wind\_dir})</p>

<p><strong>Humidity:</strong> {weather.current.humidity}%</p>

<p><strong>Visibility:</strong> {weather.current.vis\_km} km</p>

</div>

) : (

<p>Loading weather data...</p>

)}

</div>

);

};

export default WeatherAlert;

**Incident.jsx:**

import { Box, Button, Flex, Textarea, TextInput } from '@mantine/core'

import React, { useState } from 'react'

const Incident = () => {

const [value,setvalue] = useState("")

const hadlereport = ()=>{

}

return (

<Box px={50} bd="1px solid black" >

<Flex>

<Textarea value={value} onChange={(event) => setvalue(event.currentTarget.value)} />

<Button>Report</Button>

</Flex>

</Box>

)

}

export default Incident

**Package.json:**

{

"name": "frontend",

"private": true,

"version": "0.0.0",

"type": "module",

"scripts": {

"dev": "vite",

"build": "vite build",

"lint": "eslint .",

"preview": "vite preview"

},

"dependencies": {

"@emailjs/browser": "^4.4.1",

"@mantine/core": "^7.17.1",

"@mantine/hooks": "^7.17.1",

"axios": "^1.8.2",

"react": "^19.0.0",

"react-dom": "^19.0.0",

"react-router-dom": "^7.2.0"

},

"devDependencies": {

"@eslint/js": "^9.21.0",

"@types/react": "^19.0.10",

"@types/react-dom": "^19.0.4",

"@vitejs/plugin-react": "^4.3.4",

"eslint": "^9.21.0",

"eslint-plugin-react-hooks": "^5.1.0",

"eslint-plugin-react-refresh": "^0.4.19",

"globals": "^15.15.0",

"postcss": "^8.5.3",

"postcss-preset-mantine": "^1.17.0",

"postcss-simple-vars": "^7.0.1",

"vite": "^6.2.0"

}

}

**Index.html:**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<link rel="icon" type="image/svg+xml" href="/vite.svg" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Vite + React</title>

</head>

<body>

<div id="root"></div>

<script type="module" src="/src/main.jsx"></script>

<script type="text/javascript">

(function(d, t) {

var v = d.createElement(t), s = d.getElementsByTagName(t)[0];

v.onload = function() {

window.voiceflow.chat.load({

verify: { projectID: '67d1a4c75c5916e15e7ce59a' },

url: 'https://general-runtime.voiceflow.com',

versionID: 'production',

voice: {

url: "https://runtime-api.voiceflow.com"

}

});

}

v.src = "https://cdn.voiceflow.com/widget-next/bundle.mjs"; v.type = "text/javascript"; s.parentNode.insertBefore(v, s);

})(document, 'script');

</script>

</body>

</html>

**eslint.config.js:**

import js from '@eslint/js'

import globals from 'globals'

import reactHooks from 'eslint-plugin-react-hooks'

import reactRefresh from 'eslint-plugin-react-refresh'

export default [

{ ignores: ['dist'] },

{

files: ['\*\*/\*.{js,jsx}'],

languageOptions: {

ecmaVersion: 2020,

globals: globals.browser,

parserOptions: {

ecmaVersion: 'latest',

ecmaFeatures: { jsx: true },

sourceType: 'module',

},

},

plugins: {

'react-hooks': reactHooks,

'react-refresh': reactRefresh,

},

rules: {

...js.configs.recommended.rules,

...reactHooks.configs.recommended.rules,

'no-unused-vars': ['error', { varsIgnorePattern: '^[A-Z\_]' }],

'react-refresh/only-export-components': [

'warn',

{ allowConstantExport: true },

],

},

},

]

**Backend**

**server.js:**

const express = require("express");

const sqlite3 = require("sqlite3").verbose();

const signupROute = require('./routes/signupRoute');

const login = require('./routes/login')

const cors = require('cors');

const weatherapi = require('./routes/weatherApi')

const userdata = require('./routes/userdata');

const updateuser = require('./routes/updateuser');

const user = require('./routes/specifiuser');

const deleteuser = require('./routes/deleteuser');

const app = express();

const PORT = 3000;

app.use(cors({

origin: 'http://localhost:5173', // React frontend origin

methods: ['GET', 'POST', 'PUT', 'DELETE'],

credentials: true

}))

app.use(express.json());

// Connect to SQLite database

const db = new sqlite3.Database("./corporatedatabase.db", (err) => {

if (err) {

console.error("Error connecting to SQLite database:", err.message);

} else {

console.log("Connected to the SQLite database successfully!");

console.log("Database file path:", require("path").resolve("./corporatedatabase.db"));

}

});

// Test route

app.get("/", (req, res) => {

db.all("SELECT name FROM sqlite\_master WHERE type='table'", [], (err, tables) => {

if (err) {

return res.status(500).json({ error: err.message });

}

res.json({ message: "Connection successful!", tables });

});

});

//creating the user table

const createusertable = `CREATE TABLE IF NOT EXISTS user (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT NOT NULL UNIQUE,

password TEXT NOT NULL,

role TEXT NOT NULL

);`

db.run(createusertable, (err) => {

if (err) {

console.error("Error creating 'user' table:", err.message);

} else {

console.log("'user' table created successfully.");

}

});

app.use('/api',signupROute);

app.use('/login',login);

app.use('/userdata',userdata);

app.use("/api", weatherapi);

app.use("/",updateuser);

app.use("/",user);

app.use("/",deleteuser);

// Close the database connection on exit

process.on("SIGINT", () => {

db.close(() => {

console.log("SQLite database connection closed.");

process.exit(0);

});

});

app.listen(PORT, () => {

console.log(`Server is running on http://localhost:${PORT}`);

});

**Login.jsx:**

const express = require("express");

const sqlite3 = require("sqlite3").verbose();

const router = express.Router();

router.use(express.json());

const db = new sqlite3.Database("./corporatedatabase.db");

// Allowed roles and their respective dashboard pages

const rolePages = {

associate: "associate",

manager: "manager",

executive: "executive",

admin: "admin"

};

// Login Route

router.post("/", (req, res) => {

const { username, password, role } = req.body;

if (!username || !password || !role) {

return res.status(400).json({ error: "Username, password, and role are required" });

}

// Convert role to lowercase for case-insensitive comparison

const normalizedRole = role.toLowerCase();

// Check if role is valid

if (!rolePages[normalizedRole]) {

return res.status(400).json({ error: "Invalid role selected" });

}

// Query to check if user exists with the given username, password, and role

const query = "SELECT \* FROM user WHERE username = ? AND password = ? AND LOWER(role) = LOWER(?)";

db.get(query, [username, password, role], (err, user) => {

if (err) {

return res.status(500).json({ error: "Database error" });

}

if (!user) {

return res.status(401).json({ error: "Invalid username, password, or role" });

}

// If user exists, authenticate and redirect to appropriate dashboard

res.json({

message: "Login successful",

role: normalizedRole,

redirectTo: rolePages[normalizedRole]

});

});

});

module.exports = router;

**deleteuser.js:**

const express = require('express');

const sqlite3 = require('sqlite3').verbose();

const router = express.Router();

const db = new sqlite3.Database("./corporatedatabase.db");

router.use(express.json());

router.delete('/delete/:id', (req, res) => {

const { id } = req.params;

if (!id) {

return res.status(400).json({ message: "User ID is required" });

}

const query = `DELETE FROM user WHERE id = ?`;

db.run(query, [id], function (err) {

if (err) {

return res.status(500).json({ message: "Failed to delete user", error: err.message });

}

if (this.changes === 0) {

return res.status(404).json({ message: "User not found" });

}

res.json({ message: "User deleted successfully" });

});

});

module.exports = router;

**signuproute.js:**

const express = require("express");

const sqlite3 = require("sqlite3").verbose();

const router = express.Router();

const db = new sqlite3.Database("corporatedatabase.db");

// Create users table if not exists

db.run(

`CREATE TABLE IF NOT EXISTS user (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT UNIQUE NOT NULL,

password TEXT NOT NULL,

role TEXT NOT NULL

)`

);

// Signup API (without hashing)

router.post("/signup", (req, res) => {

const { username, password, role } = req.body;

if (!username || !password || !role) {

return res.status(400).json({ error: "All fields are required" });

}

db.run(

"INSERT INTO user (username, password, role) VALUES (?, ?, ?)",

[username, password, role],

function (err) {

if (err) {

return res.status(500).json({ error: "User already exists or DB error" });

}

res.status(201).json({ message: "User registered successfully", userId: this.lastID });

}

);

});

module.exports = router;

**Specificuser.jsx:**

const express = require("express");

const sqlite3 = require("sqlite3").verbose();

const router = express.Router();

router.use(express.json());

const db = new sqlite3.Database("./corporatedatabase.db");

router.get('/user/:id', (req, res) => {

const employeeId = req.params.id;

const query = `SELECT \* FROM user WHERE id = ?`;

db.get(query, [employeeId], (err, row) => {

if (err) {

return res.status(500).json({ error: err.message });

}

if (!row) {

return res.status(404).json({ message: 'Employee not found' });

}

res.json(row);

});

});

module.exports = router

**Updateuser.jsx:**

const express = require('express');

const sqlite3 = require('sqlite3').verbose();

const router = express.Router();

const db = new sqlite3.Database("./corporatedatabase.db");

router.use(express.json()); // Ensure JSON parsing

router.put("/updateuser", (req, res) => {

const { id, username, password, role } = req.body;

if (!id || !username || !password || !role) {

return res.status(400).json({ message: "Must provide all credentials" });

}

const query = `UPDATE user SET username=?, password=?, role=? WHERE id=?`;

db.run(query, [username, password, role, id], function (err) {

if (err) {

return res.status(500).json({ message: err.message });

}

if (this.changes === 0) {

return res.status(404).json({ message: "User not found" });

}

res.json({ message: "User updated successfully" });

});

});

module.exports = router;

**Userdate.jsx:**

const express = require("express");

const sqlite3 = require("sqlite3").verbose();

const router = express.Router();

const db = new sqlite3.Database("./corporatedatabase.db");

router.get("/",(req,res)=>{

const sql = 'SELECT \* FROM USER';

db.all(sql,[],(err,row) =>{

if(err){

res.status(500).json({error:err.message});

return;

}

res.json(row);

})

})

module.exports = router;

**Weatherapi.jsx:**

const express = require("express");

const axios = require("axios");

const router = express.Router();

const API\_KEY = "214a09fd9e484e339c1154925251303";

const LOCATION = "Chennai,IN";

router.get("/weather", async (req, res) => {

try {

const response = await axios.get(

"https://api.weatherapi.com/v1/current.json?key=214a09fd9e484e339c1154925251303&q=chennai&aqi=no",

{

headers: {

"User-Agent": "Mozilla/5.0",

},

}

);

console.log(response);

res.json(response.data);

} catch (error) {

console.log("Weather API Error:", error.response?.data || error.message);

res.status(500).json({ error: "Failed to fetch weather data" });

}

});

module.exports = router;

**Package.json:**

{

"dependencies": {

"axios": "^1.8.3",

"cors": "^2.8.5",

"express": "^4.21.2",

"nodemon": "^3.1.9",

"sqlite3": "^5.1.7"

}

}