Reference

// 🔙 BACKEND: Add this route in your Express backend (groupMessages.js or messages.js)

import express from 'express';

import db from '../models/db.js';

const router = express.Router();

// DELETE a group message

router.delete('/groups/group-messages/:g\_m\_id', async (req, res) => {

const { g\_m\_id } = req.params;

try {

const [result] = await db.query('DELETE FROM group\_message WHERE g\_m\_id = ?', [g\_m\_id]);

if (result.affectedRows === 0) {

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

}

res.json({ message: 'Message deleted successfully' });

} catch (err) {

console.error('Error deleting message:', err);

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

}

});

export default router;

// ✅ FRONTEND (React): Inside Chat component

const handleDeleteGroupMessage = async (g\_m\_id) => {

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

if (!confirmDelete) return;

try {

await api.delete(`/api/groups/group-messages/${g\_m\_id}`);

setGroupMessages(prev => prev.filter(msg => msg.g\_m\_id !== g\_m\_id));

} catch (err) {

console.error('Failed to delete message:', err);

}

};

// ✅ Inside return, map over groupMessages:

<div

key={i}

className={`flex ${msg.sender\_name === myName ? 'justify-end' : 'justify-start'}`}

>

<div

className={`group relative max-w-xs lg:max-w-md px-4 py-2 rounded-2xl shadow-sm ${

msg.sender\_name === myName

? 'bg-blue-600 text-white rounded-br-none'

: 'bg-white text-gray-800 border border-gray-200 rounded-bl-none'

}`}

>

{/\* message content \*/}

{msg.sender\_name !== myName && (

<p className="text-xs font-semibold text-blue-600 mb-1">{msg.sender\_name}</p>

)}

<p className="text-sm">{msg.content}</p>

<p className={`text-xs mt-1 ${

msg.sender\_name === myName ? 'text-blue-100' : 'text-gray-500'

}`}>

{new Date(msg.created\_at).toLocaleTimeString([], { hour: '2-digit', minute: '2-digit' })}

</p>

{/\* Delete button, visible on hover \*/}

{msg.sender\_name === myName && (

<button

onClick={() => handleDeleteGroupMessage(msg.g\_m\_id)}

className="hidden group-hover:flex absolute top-2 right-2 ring-2 text-red-500 hover:text-red-700 bg-white p-1 rounded-full shadow"

aria-label="Delete message"

>

<svg xmlns="http://www.w3.org/2000/svg" className="h-4 w-4" fill="none" viewBox="0 0 24 24" stroke="currentColor">

<path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M6 18L18 6M6 6l12 12" />

</svg>

</button>

)}

</div>

</div>

For private message

// 🔙 BACKEND: Add this route in your Express backend (messages.js)

import express from 'express';

import db from '../models/db.js';

const router = express.Router();

// DELETE a private message

router.delete('/messages/:message\_id', async (req, res) => {

const { message\_id } = req.params;

try {

const [result] = await db.query('DELETE FROM messages WHERE id = ?', [message\_id]);

if (result.affectedRows === 0) {

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

}

res.json({ message: 'Message deleted successfully' });

} catch (err) {

console.error('Error deleting message:', err);

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

}

});

export default router;

// ✅ FRONTEND (React): Inside Chat component

const handleDeletePrivateMessage = async (id) => {

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

if (!confirmDelete) return;

try {

await api.delete(`/api/messages/${id}`);

setMessages(prev => prev.filter(msg => msg.id !== id));

} catch (err) {

console.error('Failed to delete message:', err);

}

};

// ✅ Inside return, map over private messages:

{messages.map((msg) => (

<div

key={msg.id}

className={`group flex ${msg.sender\_name === myName ? 'justify-end' : 'justify-start'} relative`}

>

<div

className={`max-w-xs lg:max-w-md px-4 py-2 rounded-2xl shadow-sm relative ${

msg.sender\_name === myName

? 'bg-blue-600 text-white rounded-br-none'

: 'bg-white text-gray-800 border border-gray-200 rounded-bl-none'

}`}

>

{msg.sender\_name !== myName && (

<p className="text-xs font-semibold text-blue-600 mb-1">{msg.sender\_name}</p>

)}

<p className="text-sm">{msg.content}</p>

<p className={`text-xs mt-1 ${msg.sender\_name === myName ? 'text-blue-100' : 'text-gray-500'}`}>

{new Date(msg.created\_at).toLocaleTimeString([], { hour: '2-digit', minute: '2-digit' })}

</p>

{msg.sender\_name === myName && (

<button

onClick={() => handleDeletePrivateMessage(msg.id)}

className="hidden group-hover:flex absolute -top-2 -right-2 text-red-500 hover:text-red-700 bg-white p-1 rounded-full shadow"

>

<svg xmlns="http://www.w3.org/2000/svg" className="h-4 w-4" fill="none" viewBox="0 0 24 24" stroke="currentColor">

<path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M6 18L18 6M6 6l12 12" />

</svg>

</button>

)}

</div>

</div>

))}

Handle search on bashboard

// ✅ BACKEND: Add this to your posts route file (e.g., posts.js)

import express from 'express';

import db from '../models/db.js';

const router = express.Router();

// GET /api/posts (with optional name query)

router.get('/posts', async (req, res) => {

const { name } = req.query;

try {

let sql = 'SELECT \* FROM posts';

let values = [];

if (name) {

sql += ' WHERE name LIKE ?';

values.push(`%${name}%`);

}

sql += ' ORDER BY created\_at DESC';

const [rows] = await db.query(sql, values);

res.json(rows);

} catch (err) {

console.error('Error fetching posts:', err);

res.status(500).json({ error: 'Server error' });

}

});

export default router;

// ✅ FRONTEND (React) update

import api from "../api";

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

import { useState, useEffect } from "react";

import { FaHome, FaEnvelope, FaBell, FaUserCircle, FaSearch } from "react-icons/fa";

export default function Dashboard() {

const [user, setUser] = useState("");

const [allUsers, setAllUsers] = useState([]);

const [content, setContent] = useState("");

const [image, setImage] = useState(null);

const [post, setPost] = useState([]);

const [searchTerm, setSearchTerm] = useState("");

const navigate = useNavigate();

const handlePostSubmit = async (e) => {

e.preventDefault();

const formData = new FormData();

formData.append("content", content);

if (image) formData.append("image", image);

try {

await api.post("/api/posts", formData);

setContent("");

setImage(null);

fetchPosts();

} catch (err) {

alert("Post failed.");

}

};

const fetchPosts = async (name = "") => {

const res = await api.get(`/api/posts${name ? `?name=${name}` : ""}`);

setPost(res.data);

};

useEffect(() => {

fetchPosts();

}, []);

useEffect(() => {

api.get('/api/auth/profile')

.then((res) => {

setUser(res.data);

return api.get('/api/users');

})

.then((res) => {

setAllUsers(res.data);

})

.catch(() => {

alert("Please login first");

navigate('/');

});

}, []);

if (!user) return <p className="text-center mt-10">Loading.....</p>;

return (

<div className="min-h-screen flex overflow-hidden">

<aside className="w-64 bg-gray-100 p-4 border-r shadow-sm overflow-y-auto fixed top-16 left-0 h-full z-10">

{/\* Profile & Post Form \*/}

<h2 className="text-lg font-bold mb-4">Profile</h2>

<div className="space-y-1 text-sm">

<p><strong>Name:</strong> {user.name}</p>

<p><strong>Phone:</strong> {user.phone}</p>

<p><strong>Role:</strong> {user.role}</p>

</div>

{['director', 'dos', 'patron', 'matron', 'dod'].includes(user.role) && (

<form className="mt-4 p-4 flex flex-col gap-4 bg-white shadow-md border" onSubmit={handlePostSubmit} encType="multipart/form-data">

<label className="text-sm font-semibold text-gray-700">Write a post:</label>

<textarea

value={content}

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

placeholder="Post something..."

className="border border-gray-300 p-3 rounded-md resize-none min-h-[100px] focus:outline-none focus:ring-2 focus:ring-blue-400"

></textarea>

<label className="text-sm font-semibold text-gray-700 block mb-1">Upload image (optional):</label>

<input type="file" onChange={(e) => setImage(e.target.files[0])} className="block w-full text-sm text-gray-600 file:mr-4 file:py-2 file::px-4 file:rounded file:border-0 file:text-sm file:semibold file:bg-blue-200 file:text-blue-700 hover:file:bg-gray-200" />

<button className="mt-6 bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700">Post</button>

</form>

)}

</aside>

<main className="flex-1 ml-64 overflow-y-auto p-6 bg-white h-screen">

<div className="flex items-center justify-between mb-4">

<h2 className="text-xl font-bold">Posts</h2>

<form

onSubmit={(e) => {

e.preventDefault();

fetchPosts(searchTerm);

}}

className="flex items-center border border-gray-300 rounded-lg overflow-hidden shadow-sm"

>

<input

type="text"

placeholder="Search by name..."

value={searchTerm}

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

className="px-3 py-2 focus:outline-none w-64"

/>

<button type="submit" className="px-3 py-2 bg-blue-600 text-white hover:bg-blue-700">

<FaSearch />

</button>

</form>

</div>

{post.map((post) => (

<div key={post.post\_id} className="p-4 border rounded shadow mb-4 bg-gray-50">

<div className="flex items-center gap-2 mb-2">

<FaUserCircle className="text-xl text-blue-500" />

<span className="font-semibold">{post.name} ({post.role})</span>

</div>

<span className="text-xs text-gray-500 ml-auto bg-yellow-100 py-1 px-2">

{new Date(post.created\_at).toLocaleString()}

</span>

{post.content && <p className="mb-2">{post.content}</p>}

{post.image && (

<img

src={`http://localhost:4000/uploads/${post.image}`}

alt="Post"

className="w-64 h-auto rounded shadow-md object-cover"

/>

)}

</div>

))}

</main>

<aside className="w-64 bg-gray-50 p-4 border-1 shadow-sm overflow-y-auto fixed top-16 right-0 h-full z-10">

<h2 className="text-lg font-bold mb-4">All Users</h2>

<ul className="space-y-2">

{allUsers.map((u) => (

<li key={u.phone}>

<button

onClick={() => navigate(`/chat?user=${u.name}`)}

className="w-full p-2 bg-white hover:bg-blue-100 rounded flex justify-between items-center text-left"

>

<div>

<strong>{u.name}</strong><br />

<span className="text-xs text-gray-600">{u.role}</span>

</div>

<FaUserCircle className="text-2xl text-blue-500" />

</button>

</li>

))}

</ul>

</aside>

</div>

);

}

Searching user

import api from "../api";

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

import { useState, useEffect } from "react";

import { FaHome, FaEnvelope, FaBell, FaUserCircle } from "react-icons/fa";

export default function Dashboard() {

const [user, setUser] = useState("");

const [allUsers, setAllUsers] = useState([]);

const [content, setContent] = useState("");

const [image, setImage] = useState(null);

const [post, setPost] = useState([]);

const [searchTerm, setSearchTerm] = useState("");

const [userSearch, setUserSearch] = useState("");

const navigate = useNavigate();

const handlePostSubmit = async (e) => {

e.preventDefault();

const formData = new FormData();

formData.append("content", content);

if (image) formData.append("image", image);

try {

await api.post("/api/posts", formData);

setContent("");

setImage(null);

fetchPosts();

} catch (err) {

alert("Post failed.");

}

};

const fetchPosts = async () => {

const res = await api.get(`/api/posts?name=${searchTerm}`);

setPost(res.data);

};

useEffect(() => {

fetchPosts();

}, [searchTerm]);

useEffect(() => {

api.get('/api/auth/profile')

.then((res) => {

setUser(res.data);

return api.get('/api/users');

})

.then((res) => {

setAllUsers(res.data);

})

.catch((err) => {

console.error("Error fetching profile or users:", err);

alert("Please login first");

navigate('/');

});

}, []);

const filteredUsers = allUsers.filter((u) =>

u.name.toLowerCase().includes(userSearch.toLowerCase())

);

if (!user) return <p className="text-center mt-10">Loading.....</p>;

return (

<div className="min-h-screen flex overflow-hidden">

{/\* Left side profile \*/}

<aside className="w-64 bg-gray-100 p-4 border-r shadow-sm overflow-y-auto fixed top-16 left-0 h-full z-10">

<h2 className="text-lg font-bold mb-4">Profile</h2>

<div className="space-y-1 text-sm">

<p><strong>Name:</strong> {user.name} </p>

<p><strong>Phone:</strong> {user.phone} </p>

<p><strong>Role:</strong> {user.role} </p>

</div>

{['director', 'dos', 'patron', 'matron', 'dod'].includes(user.role) && (

<form

className="mt-4 p-4 flex flex-col gap-4 bg-white shadow-md border"

onSubmit={handlePostSubmit}

encType="multipart/form-data"

>

<label className="text-sm font-semibold text-gray-700">Write a post:</label>

<textarea

value={content}

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

placeholder="Post something..."

className="border border-gray-300 p-3 rounded-md resize-none min-h-[100px] focus:outline-none focus:ring-2 focus:ring-blue-400"

></textarea>

<label className="text-sm font-semibold text-gray-700 block mb-1">Upload image (optional):</label>

<input type="file"

onChange={(e) => setImage(e.target.files[0])}

className="block w-full text-sm text-gray-600 file:mr-4 file:py-2 file:px-4 file:rounded file:border-0 file:text-sm file:semibold file:bg-blue-200 file:text-blue-700 hover:file:bg-gray-200" />

<button className="mt-6 bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700 ">Post</button>

</form>

)}

</aside>

<main className="flex-1 ml-64 overflow-y-auto p-6 bg-white h-screen">

<h2 className="text-xl font-bold mb-4">Posts</h2>

<input

type="text"

placeholder="Search posts by name..."

value={searchTerm}

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

className="mb-4 p-2 border border-gray-300 rounded w-full focus:outline-none focus:ring-2 focus:ring-blue-400"

/>

{post.map((post) => (

<div key={post.post\_id} className="p-4 border rounded shadow mb-4 bg-gray-50">

<div className="flex items-center gap-2 mb-2">

<FaUserCircle className="text-xl text-blue-500" />

<span className="font-semibold">{post.name} ({post.role})</span>

</div>

<span className="text-xs text-gray-500 ml-auto bg-yellow-100 py-1 px-2">

{new Date(post.created\_at).toLocaleString()}

</span>

{post.content && <p className="mb-2">{post.content}</p>}

{post.image && (

<img

src={`http://localhost:4000/uploads/${post.image}`}

alt="Post"

className="w-64 h-auto rounded shadow-md object-cover"

/>

)}

</div>

))}

</main>

<aside className="w-64 bg-gray-50 p-4 border-1 shadow-sm overflow-y-auto fixed top-16 right-0 h-full z-10">

<h2 className="text-lg font-bold mb-4">All Users</h2>

<input

type="text"

placeholder="Search users by name..."

value={userSearch}

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

className="mb-4 p-2 border border-gray-300 rounded w-full focus:outline-none focus:ring-2 focus:ring-blue-400"

/>

<ul className="space-y-2">

{filteredUsers.map((u) => (

<li key={u.phone}>

<button

onClick={() => navigate(`/chat?user=${u.name}`)}

className="w-full p-2 bg-white hover:bg-blue-100 rounded flex justify-between items-center text-left"

>

<div>

<strong>{u.name}</strong> <br />

<span className="text-xs text-gray-600">{u.role}</span>

</div>

<FaUserCircle className="text-2xl text-blue-500" />

</button>

</li>

))}

</ul>

</aside>

</div>

);

}

Backend

UserController

import db from '../models/db.js';

export const getAllUsers = async (req, res) => {

const { name } = req.query;

try {

let query = `SELECT user\_id, name, phone, role FROM users`;

const params = [];

if (name) {

query += ` WHERE name LIKE ?`;

params.push(`%${name}%`);

}

const [rows] = await db.query(query, params);

res.json(rows);

} catch (err) {

console.error("Error fetching users:", err);

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

}

};

Frotned

useEffect(() => {

const fetchUsers = async () => {

try {

const res = await api.get(`/api/users?name=${userSearch}`);

setAllUsers(res.data);

} catch (err) {

console.error("Failed to search users", err);

}

};

fetchUsers();

}, [userSearch]);

Frotned full

import api from "../api";

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

import { useState, useEffect } from "react";

import { FaUserCircle } from "react-icons/fa";

export default function Dashboard() {

const [user, setUser] = useState("");

const [allUsers, setAllUsers] = useState([]);

const [content, setContent] = useState("");

const [image, setImage] = useState(null);

const [post, setPost] = useState([]);

const [searchTerm, setSearchTerm] = useState(""); // for posts search

const [userSearch, setUserSearch] = useState(""); // for users search

const navigate = useNavigate();

const handlePostSubmit = async (e) => {

e.preventDefault();

const formData = new FormData();

formData.append("content", content);

if (image) formData.append("image", image);

try {

await api.post("/api/posts", formData);

setContent("");

setImage(null);

fetchPosts(searchTerm);

} catch (err) {

alert("Post failed.");

}

};

const fetchPosts = async (name = "") => {

try {

const res = await api.get(`/api/posts?name=${encodeURIComponent(name)}`);

setPost(res.data);

} catch (err) {

console.error("Failed to fetch posts:", err);

}

};

const fetchUsers = async (name = "") => {

try {

const res = await api.get(`/api/users?name=${encodeURIComponent(name)}`);

setAllUsers(res.data);

} catch (err) {

console.error("Failed to fetch users:", err);

}

};

useEffect(() => {

fetchPosts(searchTerm);

}, [searchTerm]);

useEffect(() => {

fetchUsers(userSearch);

}, [userSearch]);

useEffect(() => {

api

.get("/api/auth/profile")

.then((res) => {

setUser(res.data);

})

.catch(() => {

alert("Please login first");

navigate("/");

});

}, [navigate]);

if (!user) return <p className="text-center mt-10">Loading.....</p>;

return (

<div className="min-h-screen flex overflow-hidden">

{/\* Left side profile and post form \*/}

<aside className="w-64 bg-gray-100 p-4 border-r shadow-sm overflow-y-auto fixed top-16 left-0 h-full z-10">

<h2 className="text-lg font-bold mb-4">Profile</h2>

<div className="space-y-1 text-sm">

<p>

<strong>Name:</strong> {user.name}{" "}

</p>

<p>

<strong>Phone:</strong> {user.phone}{" "}

</p>

<p>

<strong>Role:</strong> {user.role}{" "}

</p>

</div>

{["director", "dos", "patron", "matron", "dod"].includes(user.role) && (

<form

className="mt-4 p-4 flex flex-col gap-4 bg-white shadow-md border"

onSubmit={handlePostSubmit}

encType="multipart/form-data"

>

<label className="text-sm font-semibold text-gray-700">

Write a post:

</label>

<textarea

value={content}

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

placeholder="Post something..."

className="border border-gray-300 p-3 rounded-md resize-none min-h-[100px] focus:outline-none focus:ring-2 focus:ring-blue-400"

></textarea>

<label className="text-sm font-semibold text-gray-700 block mb-1">

Upload image (optional):

</label>

<input

type="file"

onChange={(e) => setImage(e.target.files[0])}

className="block w-full text-sm text-gray-600

file:mr-4 file:py-2 file:px-4

file:rounded file:border-0

file:text-sm file:semibold

file:bg-blue-200 file:text-blue-700

hover:file:bg-gray-200"

/>

<button className="mt-6 bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700 ">

Post

</button>

</form>

)}

</aside>

{/\* Main posts section \*/}

<main className="flex-1 ml-64 overflow-y-auto p-6 bg-white h-screen">

<h2 className="text-xl font-bold mb-4">Posts</h2>

<input

type="text"

placeholder="Search posts by name..."

value={searchTerm}

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

className="mb-4 p-2 border border-gray-300 rounded w-full focus:outline-none focus:ring-2 focus:ring-blue-400"

/>

{post.map((post) => (

<div

key={post.post\_id}

className="p-4 border rounded shadow mb-4 bg-gray-50"

>

<div className="flex items-center gap-2 mb-2">

<FaUserCircle className="text-xl text-blue-500" />

<span className="font-semibold">

{post.name} ({post.role})

</span>

</div>

<span className="text-xs text-gray-500 ml-auto bg-yellow-100 py-1 px-2">

{new Date(post.created\_at).toLocaleString()}

</span>

{post.content && <p className="mb-2">{post.content}</p>}

{post.image && (

<img

src={`http://localhost:4000/uploads/${post.image}`}

alt="Post"

className="w-64 h-auto rounded shadow-md object-cover"

/>

)}

</div>

))}

</main>

{/\* Right side user list and search \*/}

<aside className="w-64 bg-gray-50 p-4 border-1 shadow-sm overflow-y-auto fixed top-16 right-0 h-full z-10">

<h2 className="text-lg font-bold mb-4">All Users</h2>

<input

type="text"

placeholder="Search users by name..."

value={userSearch}

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

className="mb-4 p-2 border border-gray-300 rounded w-full focus:outline-none focus:ring-2 focus:ring-blue-400"

/>

<ul className="space-y-2">

{allUsers.map((u) => (

<li key={u.phone}>

<button

onClick={() => navigate(`/chat?user=${u.name}`)}

className="w-full p-2 bg-white hover:bg-blue-100 rounded flex justify-between items-center text-left"

>

<div>

<strong>{u.name}</strong> <br />

<span className="text-xs text-gray-600">{u.role}</span>

</div>

<FaUserCircle className="text-2xl text-blue-500" />

</button>

</li>

))}

</ul>

</aside>

</div>

);

}

Change profile image

FetchUser

const res = await api.get('/api/auth/profile');

setMyName(res.data.name);

setMyUserImage(res.data.image); // add this line

in jsx

<img

src={`http://localhost:4000/uploads/${user.image || 'default.png'}`}

alt={user.name}

className="w-10 h-10 rounded-full object-cover shadow"

/>