const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const requestRoutes = require('./routes/requests');

const app = express();

app.use(cors());

app.use(express.json());

mongoose.connect('mongodb://localhost:27017/fixit', {

useNewUrlParser: true,

useUnifiedTopology: true

}).then(() => console.log('MongoDB connected'))

.catch(err => console.log(err));

app.use('/api/requests', requestRoutes);

app.listen(5000, () => console.log('Server running on port 5000'));

const mongoose = require('mongoose');

const RequestSchema = new mongoose.Schema({

roomNumber: String,

description: String,

status: { type: String, default: 'Pending' },

worker: { type: String, default: null },

createdAt: { type: Date, default: Date.now }

});

module.exports = mongoose.model('Request', RequestSchema);

const express = require('express');

const router = express.Router();

const Request = require('../models/Request');

// Submit a new request

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

const { roomNumber, description } = req.body;

const newRequest = new Request({ roomNumber, description });

await newRequest.save();

res.json(newRequest);

});

// Get all pending requests

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

const requests = await Request.find({ status: 'Pending' });

res.json(requests);

});

// Worker accepts request

router.put('/:id/accept', async (req, res) => {

const { worker } = req.body;

const request = await Request.findByIdAndUpdate(

req.params.id,

{ status: 'In Progress', worker },

{ new: true }

);

res.json(request);

});

// Mark request as completed

router.put('/:id/complete', async (req, res) => {

const request = await Request.findByIdAndUpdate(

req.params.id,

{ status: 'Completed' },

{ new: true }

);

res.json(request);

});

module.exports = router;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<title>FixIt - Request Help</title>

</head>

<body>

<h2>Submit a Maintenance Request</h2>

<form id="requestForm">

<input type="text" id="roomNumber" placeholder="Room Number" required /><br>

<textarea id="description" placeholder="Describe your issue..." required></textarea><br>

<button type="submit">Submit Request</button>

</form>

<h2>Available Requests (For Workers)</h2>

<ul id="requestsList"></ul>

<script src="script.js"></script>

</body>

</html>

const API\_URL = 'http://localhost:5000/api/requests';

// Submit form

document.getElementById('requestForm').addEventListener('submit', async (e) => {

e.preventDefault();

const roomNumber = document.getElementById('roomNumber').value;

const description = document.getElementById('description').value;

const res = await fetch(API\_URL, {

method: 'POST',

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

body: JSON.stringify({ roomNumber, description })

});

const data = await res.json();

alert('Request submitted!');

loadRequests();

});

// Fetch and show available requests for workers

async function loadRequests() {

const res = await fetch(API\_URL);

const requests = await res.json();

const list = document.getElementById('requestsList');

list.innerHTML = '';

requests.forEach(req => {

const li = document.createElement('li');

li.innerHTML = `

Room ${req.roomNumber}: ${req.description}

<button onclick="acceptRequest('${req.\_id}')">Accept</button>

`;

list.appendChild(li);

});

}

async function acceptRequest(id) {

const workerName = prompt("Enter your name (worker):");

if (!workerName) return;

await fetch(`${API\_URL}/${id}/accept`, {

method: 'PUT',

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

body: JSON.stringify({ worker: workerName })

});

alert('Request accepted!');

loadRequests();

}

loadRequests();