**Online complain registration and management system**

Team members:

| DHANALAKSHMI S | au410121243006 |
| --- | --- |
| GAYATHRI M | au410121243009 |
| HEMASRI R | au410121243013 |
| KOKILA M | au410121243021 |

ABSTRACT:

The Online Complaint Registration and Management System is a web application developed using the MERN stack (MongoDB, Express.js, React.js, Node.js) that allows users to register, track, and manage complaints in an efficient manner. The system enables users to sub complaints, view the status of their complaints, update them, and delete them if necessary. The backend manages all the data using MongoDB, while the frontend provides an interactive interface for the users. This system automates the complaint process and helps organizations improve their response times and customer satisfaction.

TECHNOLOGIES USED:

FRONTEND: React.js

BACKEND: Node.js, Express

DATABASE: MongoDB

OBJECTIVES:

The primary objective of this system is to provide an efficient and user-friendly platform for registering and managing complaints. The key goals include:

**Complaint Registration:** Allow users to submit complaints via a simple form.

**Complaint Tracking:** Enable users to track the progress and status of their complaints.

**Complaint Management:** Allow administrators or relevant users to update, resolve, and delete complaints.

**Efficiency:** Provide a quick and easy way to address complaints, improving the communication between users and service providers.

**Data Handling:** Store all complaint data securely and manage it effectively using a centralized database.

DATABASE DESIGN:

The database schema in MongoDB consists of collections for complaints and users. Each complaint document stores essential details such as title, description, category, status, and timestamps.

Implementation:

Frontend: React.js components are created for the complaint registration form and a dynamic list of complaints. Axios is used to handle requests and responses between the frontend and backend.

Backend: The backend APIs are built to manage complaints (create, update, retrieve, and delete), and data is stored in MongoDB for persistence.

METHODOLOGY:

The development of the Online Complaint Registration and Management System follows these key steps:

System Design:

Frontend Design: The UI is built using React.js, ensuring that users can easily interact with the system. It includes features like complaint submission forms, complaint tracking, and an admin panel to manage complaints.

Backend Design: The backend is built using Node.js and Express.js to handle all the business logic and interact with the MongoDB database. RESTful APIs are created to perform CRUD operations for managing complaints.

Database Design: The database schema in MongoDB consists of collections for complaints and users. Each complaint document stores essential details such as title, description, category, status, and timestamps.

Implementation:

Frontend: React.js components are created for the complaint registration form and a dynamic list of complaints. Axios is used to handle requests and responses between the frontend and backend.

Backend: The backend APIs are built to manage complaints (create, update, retrieve, and delete), and data is stored in MongoDB for persistence.

SOURCE CODE:

mkdir complaint-system-backend

cd complaint-system-backend

npm init -y

npm install express mongoose body-parser cors

const express = require('express');

const mongoose = require('mongoose');

const bodyParser = require('body-parser');

const cors = require('cors');

const app = express();

app.use(bodyParser.json());

app.use(cors());

mongoose.connect('mongodb://localhost:27017/complaints\_db', { useNewUrlParser: true, useUnifiedTopology: true })

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

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

const PORT = process.env.PORT || 5000;

app. listen (PORT, () => console.log(Server running on port ${PORT}));

const mongoose = require('mongoose');

const complaintSchema = new mongoose.Schema({

complaintId: { type: String, unique: true },

title: String,

description: String,

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

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

dateUpdated: Date,

category: String,

priority: String,

assignedTo: String

});

module.exports = mongoose.model('Complaint', complaintSchema);

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

// POST: Create a new complaint

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

try {

consnt newComplaint = new Complaint(req.body);

await newComplaint.save();

res.status(201).json(newComplaint);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

// GET: Fetch all complaints

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

try {

const complaints = await Complaint.find();

res.json(complaints);

} catch (err) {

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

}

});

// PUT: Update a complaint

app.put('/complaints/:id', async (req, res) => {

try {

const updatedComplaint = await Complaint.findOneAndUpdate(

{ complaintId: req.params.id },

req.body,

{ new: true }

);

if (!updatedComplaint) {

return res.status(404).json({ error: 'Complaint not found' });

}

res.json(updatedComplaint);

} catch (err) {

res.status(400).json({ error: err.message });

}

});

// DELETE: Delete a complaint

app.delete('/complaints/:id', async (req, res) => {

try {

await Complaint.deleteOne({ complaintId: req.params.id });

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

} catch (err) {

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

  }

});

node app.js

npx create-react-app complaint-system-frontend

cd complaint-system-frontend

npm install axios

import axios from 'axios';

const api = axios.create({

baseURL: 'http://localhost:5000'

});

export default api;

import React, { useState } from 'react';

import api from '../api';

function ComplaintForm() {

const [title, setTitle] = useState('');

const [description, setDescription] = useState('');

const [category, setCategory] = useState('');

const [priority, setPriority] = useState('');

const handleSubmit = async (e) => {

e.preventDefault();

await api.post('/complaints', { title, description, category, priority });

setTitle('');

setDescription('');

setCategory('');

setPriority('');

};

return (

<form onSubmit={handleSubmit}>

<input

type="text"

placeholder="Complaint Title"

value={title}

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

required

/>

<textarea

placeholder="Complaint Description"

value={description}

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

required

/>

<input

type="text"

placeholder="Category"

value={category}

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

required

/>

<input

type="text"

placeholder="Priority"

value={priority}

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

required

/>

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

</form>

);

}

export default ComplaintForm;

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

import api from '../api';

function ComplaintsList() {

const [complaints, setComplaints] = useState([]);

useEffect(() => {

const fetchComplaints = async () => {

const response = await api.get('/complaints');

setComplaints(response.data);

};

fetchComplaints();

}, []);

const handleUpdate = async (id, status) => {

await api.put(/complaints/${id}, { status });

const updatedComplaints = complaints.map((complaint) =>

complaint.complaintId === id

? { ...complaint, status }

: complaint

);

setComplaints(updatedComplaints);

};

const handleDelete = async (id) => {

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

setComplaints(complaints.filter(complaint => complaint.complaintId !== id));

};

return (

<div>

<h2>Complaints List</h2>

{complaints.map((complaint) => (

<div key={complaint.complaintId}>

<h3>{complaint. title}</h3>

<p>{complaint.description}</p>

<p>Status: {complaint.status}</p>

<button onClick={() => handleUpdate(complaint.complaintId, 'Resolved')}>Mark as Resolved</button>

<button onClick={() => handleDelete(complaint.complaintId)}>Delete</button>

</div>

))}

</div>

);

}

import React from 'react';

import ComplaintForm from './components/ComplaintForm';

import ComplaintsList from './components/ComplaintsList';

function App() {

return (

<div>

<h1>Complaint Management System</h1>

<ComplaintForm />

<ComplaintsList />

</div>

);

}

SAMPLE OUTPUT:

Complaint Management System

[Complaint Title] []

[Complaint Description] [\_]

[Category] [\_]

[Priority] [\_]

[Submit Complaint Button]

Complaints List:

Complaint 1:

Title: "Broken Street Light"

Description: "The street light near the park is broken."

Status: Pending

Complaint 2:

Title: "Water Leakage"

Description: "Water leakage in my bathroom."

Status: Resolved

POST /complaints 201

GET /complaints 200

PUT /complaints/:id 200

DELETE /complaints/:id 200

CONCLUSION:

The Online Complaint Registration and Management System improves efficiency and user satisfaction by streamlining the complaint process. Built on the MERN stack, it provides a centralized, scalable platform for managing complaints, reducing response time, and enhancing transparency. Future enhancements could include features like notifications and analytics for further operational improvements