

Project Report: Online Complaint Registration

1. INTRODUCTION

1.1 Project Overview

The **Online Complaint Registration** and Management System is a full-stack web application designed to streamline complaint handling. It enables users to submit and track complaints, allows agents to respond to issues, and empowers administrators to oversee operations and maintain regulatory compliance.

1.2 Purpose

The purpose of this project is to modernize traditional complaint management practices by providing a centralized, transparent, and efficient digital platform. It ensures faster resolution, role-based access, and real-time updates.

2. IDEATION PHASE

2.1 Problem Statement

Traditional complaint handling processes are inefficient, opaque, and time-consuming. Users often face delays, lack feedback, and experience poor communication. This system addresses those gaps with real-time, automated workflows.

2.2 Empathy Map Canvas

Role	Says	Thinks	Feels	Does
User	"Why is no one responding?"	"I hope this gets resolved fast"	Frustrated, anxious	Submits complaint
Agent	"Too many complaints!"	"I must prioritize cases"	Pressured, responsible	Reviews and resolves complaints
Admin	"I need to balance workload"	"The system must stay organized"	Accountable, focused	Assigns roles, monitors activity

2.3 Brainstorming

Initial feature ideas:

- Complaint submission with attachments
 - Real-time updates and notifications
 - Role-based dashboards (User, Agent, Admin)
 - Built-in chat feature
 - Complaint assignment logic
-

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

User:

- Registers and logs in
- Submits complaint with details and evidence
- Tracks status
- Interacts with assigned agent
- Gives feedback

Agent:

- Logs in
- Views assigned complaints
- Communicates with users
- Updates complaint status
- Resolves and closes complaint

Admin:

- Logs in
- Monitors complaints
- Assigns complaints to agents
- Manages user and agent roles
- Accesses dashboard analytics

3.2 Solution Requirement

The system

- Secure authentication
- Complaint registration & status tracking
- Real-time messaging
- Role-based access

- Responsive UI

3.3 Data Flow Diagram

- User → [React UI] → [Express Backend] → [MongoDB]
- Chat: WebSocket → Backend → Socket.io Server → Client Chat Window

3.4 Technology Stack

- Frontend: React.js, Bootstrap, Material UI
- Backend: Node.js, Express.js
- Database: MongoDB
- Authentication: JWT, Bcrypt
- Real-Time: Socket.io
- Tools: Postman, Git, VS Code

4. PROJECT DESIGN

4.1 Problem Solution Fit

The system enhances visibility, reduces complaint resolution time, and ensures accountability through structured access and communication.

4.2 Proposed Solution

A single-page React app connected to a REST API built with Express. MongoDB stores structured complaint and user data. Socket.io enables live chat.

4.3 Solution Architecture

User ↔ React UI ↔ REST APIs (Express) ↔ MongoDB (via Mongoose)

- JWT for secure access
- REST API routes handle core CRUD operations
- WebSockets handle real-time messaging

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Milestone 1: Environment Setup and Project Initialization

Objective: Establish the foundational structure and development environment for both backend and frontend components.

Key Activities:

- Set up project directory structure (`/backend`, `/frontend`)
- Initialize Git repository with proper
 - `.gitignore`
- Install Node.js, MongoDB, and supporting tools (VS Code, Postman)
- Create initial configuration files:
 - `.env` for environment variables
 - `package.json` for dependency tracking

Deliverables:

- Working dev environment with separate backend and frontend folders
 - Dependencies installed and linked
 - GitHub repository with initial commit
-

Milestone 2: Backend Development

Objective: Implement the server-side logic for user authentication, complaint handling, messaging, and role-based access.

Key Activities:

- Create Express server and configure middleware (CORS, body-parser, Helmet)
- Design MongoDB schemas using Mongoose:
 - `User`, `Complaint`, `Message`
- Set up JWT-based authentication and Bcrypt for password hashing
- Develop REST API routes for:
 - `auth`, `users`, `agents`, `admin`, `complaints`, `messages`

Apply route-level middleware for authorization

Deliverables:

- Secure, modular REST API
- Connected MongoDB database
- JWT-protected route access
- Schema-based data validation

Milestone 3: Frontend Development

Objective: Develop a responsive user interface with React and integrate it with the backend API.

Key Activities:

- Create reusable UI components: Login, Dashboard, ComplaintForm, ChatWindow
- Configure routing using React Router
- Integrate Axios for API communication
- Apply styling with Tailwind CSS and Bootstrap
- Implement conditional rendering for user roles (User, Agent, Admin)

Deliverables:

- Fully functional SPA for all roles
- Dynamic dashboards based on role
- Complaint submission and listing integrated with backend
- Chat component UI setup

Milestone 4: Real-Time Messaging Integration

Objective: Enable live chat functionality between users and agents using WebSockets via Socket.io.

Key Activities:

- Integrate Socket.io server in Express backend
- Connect Socket.io client in React frontend
- Build real-time chat interface tied to specific complaint IDs
- Store messages in MongoDB using `Message` schema
- Ensure secure and scoped messaging per user and complaint

Deliverables:

- Bi-directional chat system
 - Persistent message storage
 - Realtime updates and notifications within dashboard
-

Milestone 5: Admin Features and Role Management

Objective: Implement the administrator's dashboard and functionality for managing users, agents, and complaints.

Key Activities:

- Build admin panel to view all complaints, users, and agents □ Add interfaces for:
 - Role assignment
 - Complaint reassignment
 - User deactivation
- Implement filters and status grouping for complaint views

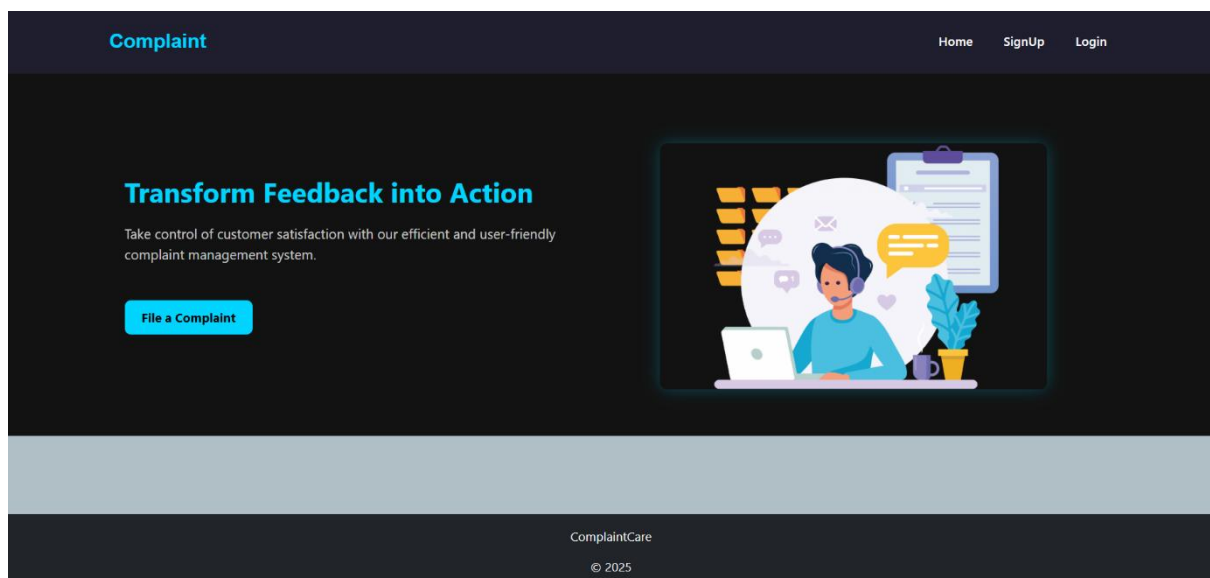
Deliverables:

- Fully featured admin dashboard
- Functional role and complaint management tools
- Complaint assignment logic with validation

6. RESULTS

6.1 Output Screenshots

- Landing Page




- Signup Page



ComplaintCare

HomeSignUpLogin

SignUp

Register for ComplaintCare

Please fill in the details to create an account

Full Name

Email

Password

Phone

User Type

Select User ▾

- Login Page

ComplaintCare

HomeSignUpLogin

Login to Continue

Please enter your credentials

Email

Password

Login

Don't have an account? [SignUp](#)

- User Complaint Dashboard

Hi, admin 1

Complaint Register

Status

LogOut

Name

Address

City

State

Pincode

Status

type pending

Description

Register

- User Dashboard

Hi, admin 1

Complaint Register

Status

LogOut

Name: charmika reddy

Address: bhimavaram

City: bhimavaram

State: ap

Pincode: 522601

Comment: water

Status: pending

Message

Name: charmika reddy

Address: bhimavaram

City: bhimavaram

State: ap

Pincode: 522601

Comment: website problem

Status: pending

Message

Agent Dashboard

Hi Agent Prasanna

View Complaints

Name: Harsha

Address: fgdtr34ertf

City: ghfgv

State: andhra pradesh

Pincode: 34564657

Comment: dsdfsfgfhgkjh

Status: completed

Status Change

Message

Message Box

Harsha: gfhdgfdhg
03:24 PM, 6/24/2025

Prasanna: hi harsha
03:28 PM, 6/24/2025

Message

Send

- Admin Dashboard

Hi Admin admin 2 Dashboard User Agent Log out				
Name	Email	Phone	Action	
admin 1	admin@gmailcom	1234567891	Update	Delete

Hi Admin admin 2DashboardUserAgentLog out

Name	Email	Phone	Action
No Agents to show			

7. ADVANTAGES & DISADVANTAGES

Advantages

- Real-time communication with agents
- Centralized dashboard for all roles
- Secure, scalable backend with REST APIs
- Responsive UI across devices
- Modular and maintainable architecture

Disadvantages

- Initial setup requires tech stack familiarity
- MongoDB must be continuously available
- Lacks offline mode and mobile app (yet)

8. CONCLUSION

The Service Desk Application meets the core requirements of modern complaint management systems. It empowers users with visibility and agents with accountability. The use of modern web technologies ensures speed, security, and scalability. The project is ready for institutional deployment and further enhancements.

9. FUTURE SCOPE

- Email and SMS notifications

- Mobile App (React Native / Flutter)
 - Multi-language support
 - AI-powered chatbot for complaint intake
 - Advanced analytics and charts for admins
 - Integration with CRM or ticketing tools
-

10. APPENDIX

Source Code

- **Backend Folder:**
 - `./backend/`
 - Express API, Mongoose Models, JWT Middleware
 - **Frontend Folder Source Code:**
 - `./frontend/`
 - React SPA with role-based UI and Axios for API communication
-

GitHub & Project Demo Link

- **Source Code GitHub Link :** https://github.com/charmika-reddy/online_complaint.git
- **Demo Video Link:** <https://youtu.be/xfFbtcBX9Cg?si=P-yqe0JoEMIMbId3>