**Full Stack Development with MERN**

## 1. Introduction

**Project Title:** ResolveNow - Online Complaint Registration and Management System

**Team ID:** LTVIP2025TMID54506

**Team Members and Roles:**

* **K Chandrika** - Team Leader & Backend Developer
* **Kadiyala Madhuri** - Frontend Developer
* **Kalakoti Ushasri** - UI/UX Designer & Tester
* **Kanala Jaya Trisha** - Full Stack Developer

## 2. Project Overview

**Purpose:** ResolveNow is a robust web-based application developed to streamline the complaint management lifecycle for both users and service providers. It centralizes complaint registration, intelligent routing, real-time communication, and resolution tracking into a single efficient interface. By leveraging modern technologies, ResolveNow enhances transparency, customer satisfaction, and compliance with industry standards.

**Key Goals:**

* Enable users to easily register complaints with all relevant data.
* Ensure timely response and resolution through efficient workflow automation.
* Foster communication between customers and agents.
* Maintain a secure and scalable system architecture.

**Features:**

* **User Registration & Login:** Secure onboarding using JWT authentication.
* **Complaint Submission:** Users can file complaints with descriptions, categories, attachments, and location details.
* **Real-time Tracking & Notifications:** Users are updated via email or SMS about status changes or agent responses.
* **Agent Interaction:** Users and agents can chat in real-time within the system interface.
* **Smart Assignment:** Complaints are routed to the appropriate department/agent based on priority, type, and location.
* **Role-based Access:** Admins, Agents, and Users have distinct interfaces and permissions.
* **Security Measures:** Data encryption, authentication middleware, and access control mechanisms.

## 3. Architecture

**Frontend:**

* **Technology:** React.js
* **Libraries:** Axios, Bootstrap, Material UI
* **Features:**
  + Modular component structure
  + Role-specific dashboards (User, Agent, Admin)
  + Real-time updates using Socket.IO
  + Mobile responsive design for accessibility across devices

**Backend:**

* **Technology:** Node.js with Express.js
* **Design:** RESTful API-based service layer
* **Functions:**
  + Handles all business logic
  + Integrates with MongoDB for data persistence
  + Manages authentication and session control
  + Includes WebSocket handling for chat feature

**Database:**

* **Type:** NoSQL
* **Technology:** MongoDB using Mongoose ODM
* **Schema Includes:**
  + **Users:** Personal and login info with roles
  + **Complaints:** Description, user ID, timestamps, status
  + **Messages:** Real-time message exchange between agent and user
  + **Agents:** Availability, assigned complaints, roles

## 4. Setup Instructions

**Prerequisites:**

* Node.js (v16 or higher)
* MongoDB (local or cloud-hosted, e.g., Atlas)
* Git

**Installation Steps:**

1. **Clone the repository:**

* git clone https://github.com/yourusername/resolvenow.git  
  cd resolvenow

1. **Install dependencies:**

* cd frontend  
  npm install  
  cd ../backend  
  npm install

1. **Environment configuration:** Create a .env file in the backend directory:

* PORT=5000  
  MONGO\_URI=mongodb://localhost:27017/resolvenow  
  JWT\_SECRET=your\_jwt\_secret

1. **Start servers:**

* # Terminal 1 - Backend  
  cd backend  
  npm start  
    
  # Terminal 2 - Frontend  
  cd frontend  
  npm run dev

## 5. Folder Structure

**Frontend Directory Structure:**

frontend/  
├── public/  
├── src/  
│ ├── components/  
│ │ ├── user/  
│ │ ├── agent/  
│ │ └── admin/  
│ ├── pages/  
│ ├── utils/  
│ ├── App.jsx  
│ └── index.js

**Backend Directory Structure:**

backend/  
├── controllers/ # Business logic  
├── models/ # Mongoose schemas  
├── routes/ # Express routes  
├── middlewares/ # Auth & error handlers  
├── config/ # Database connection  
├── sockets/ # WebSocket logic  
├── index.js # Entry point  
├── .env # Environment variables

## 6. Running the Application

# Backend Setup  
cd backend  
npm install  
npm start  
  
# Frontend Setup  
cd frontend  
npm install  
npm run dev

## 7. API Documentation

**Authentication:**

* POST /api/auth/register – Register user
* POST /api/auth/login – Login user

**Complaints:**

* POST /api/complaints – File new complaint
* GET /api/complaints/my – Fetch user’s complaints
* PUT /api/complaints/:id – Update complaint (status/comments)

**Messaging:**

* POST /api/messages/:complaintId – Post message
* GET /api/messages/:complaintId – Retrieve messages

## 8. Authentication

ResolveNow uses JWT-based secure login for all users. Role-based access is implemented through custom middleware:

* **Token Storage:** Stored securely in localStorage on the frontend.
* **Token Usage:** Included in request headers to access protected endpoints.
* **Middleware:** Validates token, checks user role (admin/agent/user), and grants or denies access.

## 9. User Interface

**User View:**

* Register/Login
* Submit new complaint
* View and track previous complaints
* Chat with assigned agent

**Agent View:**

* View assigned complaints
* Respond to user via chat
* Update status and resolution notes

**Admin View:**

* Dashboard overview (user counts, open/closed complaints)
* Manage agents and users
* View all system-wide complaints

## 10. Testing

**Manual Testing:**

* Registration/Login with various roles
* Complaint submission and resolution lifecycle
* Chat interaction and message persistence

**Automated Tools:**

* **Postman:** Test API endpoints and token-based access
* **Browser DevTools & React DevTools:** Validate frontend flows

## 11. Screenshots or Demo

## 

## 12. Known Issues

* No built-in password recovery or reset line
* Image upload support limited to specific formats
* Form validation requires improvement
* Admin dashboard lacks downloadable reports

## 13. Future Enhancements

* Implement password reset via email/token
* Enable advanced complaint categorization and search
* Integrate email/SMS alert systems for updates
* Introduce video call support via WebRTC
* Build a mobile-friendly Progressive Web App (PWA)
* Add multilingual support and accessibility features
* Generate analytical reports for admin users