# **Full Stack Development with MERN**

# **Project Documentation format**

## 1. Introduction

- **Project Title:** ResolveNow: Complaint Registration and Management
- Team Members: Kiran Kumar Madam Karanam Dattasai Khaja Bande Nawaz Katika Pamidi Sandeep Kumar

# 2. Project Overview

• **Purpose:** ResolveNow aims to streamline the complaint management process by allowing users to register issues, track their status, and interact with agents. It provides a centralized, secure, and user-friendly web application for efficient complaint resolution.

#### • Features:

- User Registration/Login
- Complaint Submission
- Status Tracking
- Real-time Messaging
- Role-Based Access for Admins, Agents, and Users
- Complaint Assignment by Admin
- Agent Dashboard for Complaint Handling

## 3. Architecture

- **Frontend:** Built using React.js with React Router for navigation. Styled using Bootstrap and MDB React UI Kit. Axios is used for HTTP communication.
- **Backend:** Built using Node.js and Express.js. Contains RESTful API endpoints for users, complaints, messages, and assignments.
- **Database:** MongoDB with Mongoose ODM is used to store user details, complaints, assigned complaints, and messages. Schemas are defined for each collection.

# 4. Setup Instructions

- Prerequisites:
  - Node.js (v14+)
  - npm
  - MongoDB installed locally or MongoDB Atlas

## • Installation:

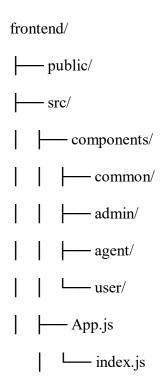
```
# Clone the project
git clone https://github.com/your-username/ResolveNow.git
cd ResolveNow

# Setup Backend
cd backend
npm install
node index.js # or use nodemon

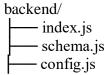
# Setup Frontend
cd ../frontend
npm install
npm start
```

# 5. Folder Structure

• **Client:** Describe the structure of the React frontend.



• **Server:** Explain the organization of the Node.js backend.



# 6. Running the Application

- Provide commands to start the frontend and backend servers locally.
  - o Frontend:

```
cd frontend
npm start
```

#### o Backend:

cd backend
node index.js

#### 7. API Documentation

- **POST** /SignUp Register a user
- **POST** /Login Login user
- POST /Complaint/:id Register a complaint by user ID
- **GET** /status/:id **Get** complaints by user ID
- POST /messages Send a message
- **GET** /messages/:complaintId Get messages by complaint ID
- **GET** /AgentUsers Fetch all agents
- **GET** /OrdinaryUsers Fetch all ordinary users
- **DELETE** /OrdinaryUsers/:id **Delete** a user and their complaint
- **PUT** /user/:id Update user details
- PUT /complaint/:complaintId Update complaint status
- POST /assignedComplaints Assign complaint to agent

# 8. Authentication

- Basic authentication using email and password.
- Session data is stored in localStorage on the client side.
- No tokens used yet could be enhanced using JWT for better security.

## 9. User Interface

- Responsive Bootstrap design
- Role-based dashboards: Admin, Agent, User
- Complaint form, status tracker, and chat interface
- Clean navigation and feedback messages
- Dark-themed navbar and cards

# 10. Testing

# **Frontend Testing:**

- Manual testing with React Developer Tools
- Form validation using Bootstrap

# **Backend Testing:**

- API testing using Postman or Thunder Client
- MongoDB testing with MongoDB Compass

## 11. Screenshots or Demo

• <a href="https://drive.google.com/file/d/1EbZcS3YRYe03D6UeuFjGZNG4rVRG-Nla/view?usp=drivesdk">https://drive.google.com/file/d/1EbZcS3YRYe03D6UeuFjGZNG4rVRG-Nla/view?usp=drivesdk</a>

# 12. Known Issues

- No email/SMS integration yet for notifications
- No password hashing (authentication security could be improved)
- No file uploads for complaint proofs

# 13. Future Enhancements

- Add JWT-based authentication
- Implement email/SMS notifications
- Add document/image upload for complaints
- Integrate pagination and search
- Build a dashboard analytics for Admin
- Role-based access middleware in backend