

# **RESOLVENOW - ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM**

## **INTRODUCTION**

ResolveNow is an online complaint registration and management platform built to simplify and digitize the way individuals or organizations report, manage, and track complaints or issues they face.

## **DESCRIPTION**

ResolveNow offers an intuitive and secure software system that centralizes the complaint submission and resolution process. It allows users to raise complaints, track them in real-time, and interact directly with the responsible agents for prompt resolutions. By incorporating features such as intelligent routing, real-time updates, and data protection, ResolveNow enhances efficiency and customer satisfaction.

The system enables structured and effective complaint handling in line with regulatory and organizational standards. It ensures systematic follow-ups and resolution timelines, helping build trust with users.

Key features include:

- **User Registration** : Individuals create accounts to file and monitor their complaints.
- **Complaint Filing** : Users can lodge complaints with supporting details such as issue description, location, and related media.
- **Live Tracking and Notifications** : Real-time status tracking and automated alerts via SMS or email keep users informed.
- **Agent Interaction** : Customers can directly message the agent handling their issue.
- **Complaint Routing** : Smart logic assigns complaints to the most suitable personnel.
- **Data Security** : Robust protocols including authentication, access controls, and encryption secure user information.

## **SCENARIO-BASED CASE STUDY**

**Scenario:** Sreenath, a customer, receives a defective item through an online purchase and decides to report the issue via ResolveNow.

### **User Registration and Login :**

- Sreenath visits the ResolveNow website and selects “Sign Up.”
- He provides his full name, email, and password to register.
- After verifying his email, he logs in with his credentials.

### **Complaint Filing :**

- After logging in, Sreenath accesses the dashboard and selects “Submit Complaint.”
- He fills in the form with details about the defective product, attaches an image, and enters relevant purchase information.
- Sreenath submits the complaint.

### **Status Tracking and Alerts :**

- A confirmation message indicates successful registration of the complaint.
- In the “My Complaints” section, Sreenath can view real-time status updates.
- Email notifications are sent for each status change.

### **Agent Communication :**

- Manoj, a service agent, is assigned the complaint.
- Manoj reaches out via ResolveNow’s built-in chat.
- Sreenath receives a notification and replies through the messaging interface.

### **Resolution and Feedback :**

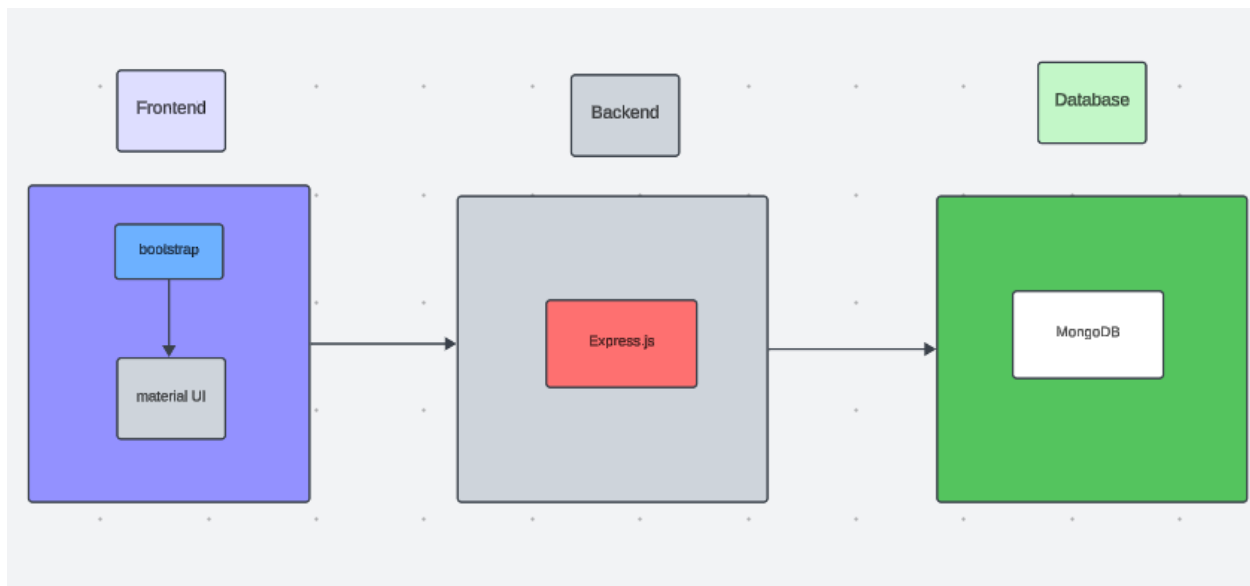
- After reviewing the issue, Manoj offers a replacement product or refund.
- Sreenath is notified and proceeds accordingly.
- He then provides feedback, appreciating the professional support.

### Admin Oversight :

- Raghu, the admin, oversees complaint flow and assigns tasks to agents.
- He ensures workload balance and policy adherence.

## TECHNICAL ARCHITECTURE

ResolveNow follows a client-server architecture. The front end (React.js) interfaces with the back end (Node.js + Express.js) via REST APIs using Axios. Real-time features utilize Socket.IO.



The technical architecture of ResolveNow follows a client-server model. The frontend acts as the client, responsible for presenting the user interface, while the backend serves as the server, managing logic and data operations.

### Frontend :

- Built using React.js, enhanced with Bootstrap and Material UI to deliver a responsive and dynamic user experience for all users—admins (Raghu), agents (Manoj), and customers (Sreenath).
- Axios is used to establish communication with the backend through RESTful APIs.

## Backend :

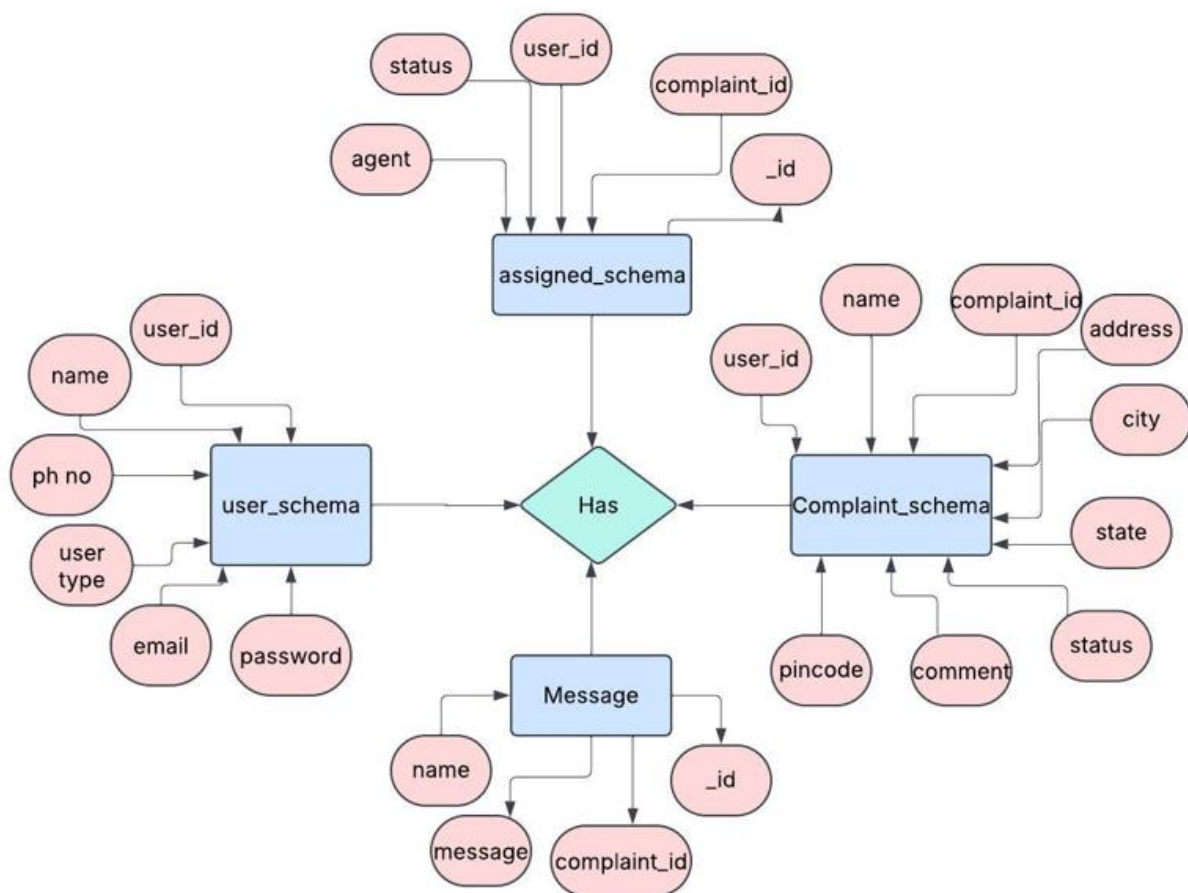
- The server-side is managed using the Express.js framework, which handles API routing, business logic, and server responses.
- MongoDB is used as the database to store user profiles, complaints, assignment details, and chat messages. Its flexibility and scalability make it ideal for this application.

## Real-Time Features :

- Socket.IO is integrated to enable real-time updates and chat functionality between users and agents.

Together, these technologies create a robust infrastructure that supports smooth, secure, and responsive complaint registration and resolution through the ResolveNow platform.

## ER DIAGRAM :



This ER diagram represents the data model of the **ResolveNow** platform, highlighting the relationships between key entities such as **users**, **agents**, **complaints**, and **messages**.

It illustrates how a **user (e.g., Sreenath)** can file a complaint by submitting required fields like name, contact information, issue details, and location. Once submitted, the complaint is stored with a reference to the user's unique `userId`.

Each complaint can then be **assigned to an agent (e.g., Manoj)** by the admin (Raghu), creating a link between `agentId` and `complaintId`. The assignment structure ensures that each complaint is directed to the appropriate support agent for resolution.

Additionally, the ER diagram includes a **message schema**, which facilitates two-way communication between users and agents. This schema references both `userId` and `complaintId`, ensuring that chat messages are tied to the correct complaint context.

The ER diagram provides a clear understanding of how data flows within ResolveNow, ensuring smooth complaint registration, tracking, agent assignment, and real-time communication.

## PRE-REQUISITES

Here are the key prerequisites for developing a full-stack application using Node.js, Express.js, MongoDB, and React.js:

### Node.js and npm :

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the server side. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server side.

Download : <https://nodejs.org/en/download/>

Installation instructions : <https://nodejs.org/en/download/package-manager/>

### Express.js :

Express.js is a fast and minimalist web application framework for Node.js. It simplifies the process of creating robust APIs and web applications, offering features like routing, middleware support, and modular architecture.

Install Express.js, a web application framework for Node.js, which handles server-side routing, middleware, and API development.

Installation: Open your command prompt or terminal and run the following command:

**npm install express**

## **MongoDB :**

MongoDB is a flexible and scalable NoSQL database that stores data in a JSON-like format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

Set up a MongoDB database to store your application's data.

Download: [https : //www.mongodb.com/try/download/community](https://www.mongodb.com/try/download/community)

Installation instructions : <https://docs.mongodb.com/manual/installation/>

## **React.js :**

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

Follow the installation guide : <https://reactjs.org/docs/create-a-new-react-app.html>

## **HTML, CSS, and JavaScript :**

Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

## **Database Connectivity :**

Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD

(Create, Read, Update, Delete) operations. To Connect the Database with Node JS go through the below provided link:

<https://www.section.io/engineering-education/nodejs-mongoosejs-mongodb/>

### **Front-end Framework :**

Utilize Reactjs to build the user-facing part of the application, including entering complaints, the status of the complaints, and user interfaces for the admin dashboard.

To make better UI we have also used some libraries like Material UI and Bootstrap.

### **Version Control :**

Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

**Git :** Download and installation instructions can be found at : <https://git-scm.com/downloads>

### **Development Environment :**

Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

- Visual Studio Code: Download from <https://code.visualstudio.com/download>

To run the existing Video Conference App project downloaded from GitHub:

Follow the below steps:

Clone the Repository : <https://github.com/SaikumarReddy504/ResolveNow.git>

- Open your terminal or command prompt.
- Navigate to the directory where you want to store the e-commerce app.
- Execute the following command to clone the repository:

git clone:

## Install Dependencies:

- Navigate into the cloned repository directory:

**cd complaint-register**

- Install the required dependencies by running the following commands:

**cd frontend**

**npm install**

**cd ../backend**

**npm install**

## Start the Development Server :

- To start the development server, execute the following command:

**npm start**

- The online complaint registration and management app will be accessible at <http://localhost:3000>

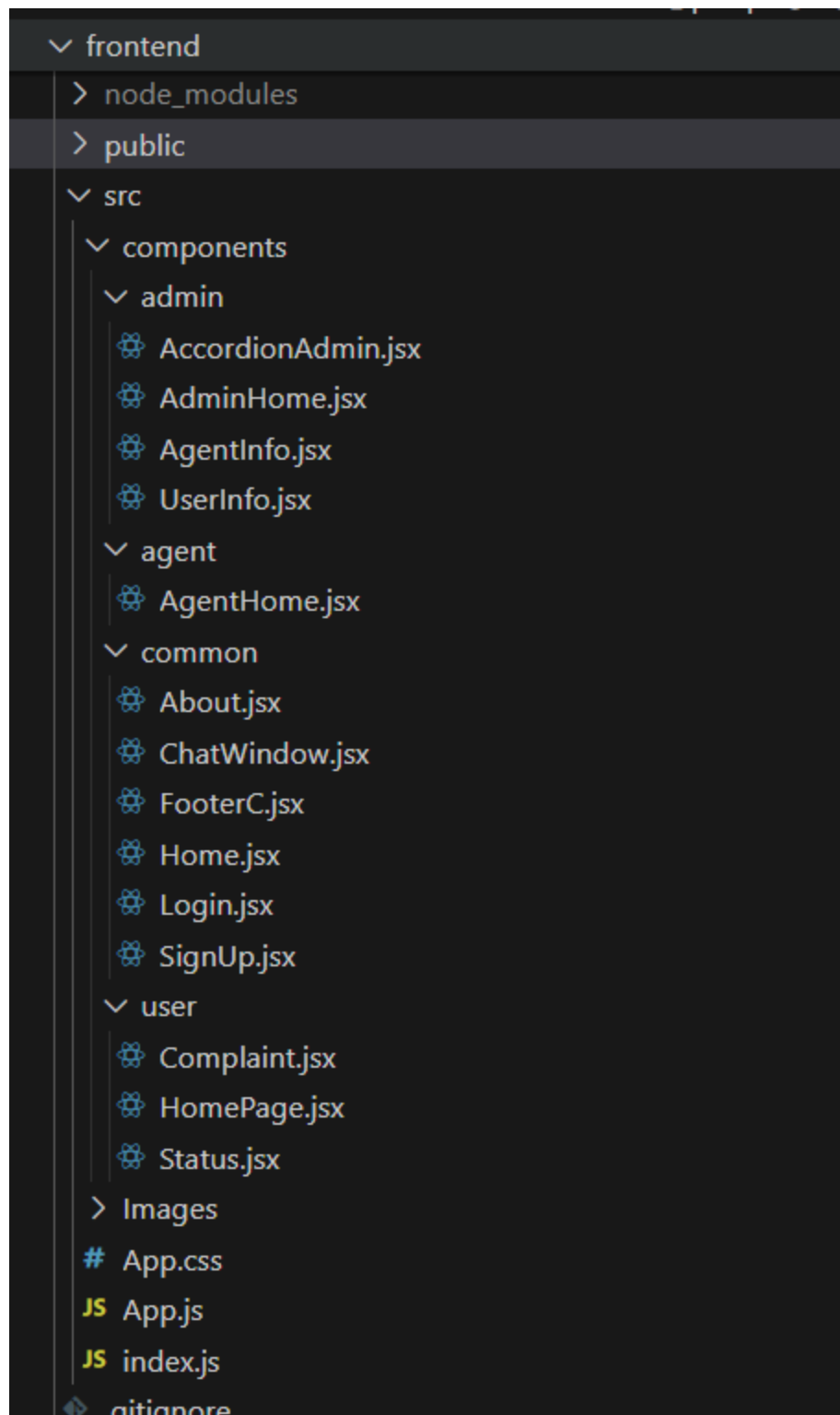
You have successfully installed and set up the online complaint registration and management app on your local machine. You can now proceed with further customization, development, and testing as needed

## PROJECT STRUCTURE

The first image illustrates the frontend directory structure, showcasing all the essential files and folders used for developing the user interface of ResolveNow. This includes components for registration, login, dashboards, complaint forms, and styling resources using Material UI and Bootstrap.

The second image displays the backend directory structure, highlighting the files and folders responsible for the server-side logic, including API routes, controllers, models, middleware, and configuration files. This structure supports functionalities such as user authentication, complaint handling, agent assignment, and real-time chat features.





```
✓ backend
  > node_modules
  JS config.js
  JS index.js
  {} package-lock.json
  {} package.json
  JS Schema.js
```

## APPLICATION FLOW :

### RESOLVENOW - ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM

#### 1. Customer (Sreenath)

**Role:** Register complaints, track progress, interact with agents, and manage personal profile.

#### Flow:

- **Registration & Login**  
Sreenath creates an account using his email and password. After verification, he logs in to access his dashboard.
- **Complaint Submission**  
He fills out a complaint form with details of the issue, contact info, and optional attachments (images, documents), and submits it.
- **Status Tracking**  
Sreenath can view the status of his complaints in real-time via his dashboard and receives automated email or SMS updates.
- **Agent Interaction**  
If assigned, Sreenath can directly chat with Manoj (or any assigned agent) through ResolveNow's internal messaging system to clarify or update the complaint.

- **Profile Management**

He can update personal details such as address, contact number, and change password through the profile section.

## **2. Agent (Manoj)**

**Role:** Resolve complaints assigned by the admin, communicate with users, and update complaint statuses.

**Flow:**

- **Registration & Login**

Manoj signs up and logs into his agent dashboard using valid credentials.

- **Complaint Management**

He reviews complaints assigned by Raghu (admin), checks complaint details, and initiates resolution steps.

- **Status Update**

Manoj changes the status of each complaint (e.g., pending, in progress, resolved) and adds notes as needed.

- **Customer Communication**

He uses the chat feature to communicate with users like Sreenath to request additional information or provide updates.

## **3. Admin (Raghu)**

**Role:** Oversee the entire system, assign complaints, manage users and agents, and ensure system efficiency and compliance.

**Flow:**

- **Monitoring & Oversight**

Raghu has full access to all complaints, user activities, and agent performance. He ensures smooth platform operations.

- **Complaint Assignment**

Based on agent workload and specialization, Raghu assigns complaints to agents like Manoj for prompt resolution.

- **User & Agent Management**

He manages registration requests, profile edits, and can deactivate or flag accounts that violate terms.

- **System Optimization**

Raghu is responsible for implementing platform improvements, updating policies, and maintaining compliance with data regulations.

## PROJECT FLOW

The **ResolveNow** project is designed to deliver an intuitive, secure, and scalable platform for managing user complaints in a structured manner. The complete development lifecycle is divided into **Six milestones**, each representing a logical stage in the project—from setup and development to testing and deployment. This structured approach ensures efficient team coordination, better debugging, and smoother feature enhancements.

### Milestone 1 : Initial Setup and Project Configuration

Before diving into development, a well-organized structure for both frontend and backend is crucial. This milestone involves preparing the workspace, installing dependencies, and configuring environment files.

#### • Steps:

1. **Create Project Directories:**

- frontend/ for the React application.
- backend/ for Express.js and database logic.

2. **Initialize Projects:**

- Use `npx create-react-app` for frontend setup.
- Run `npm init -y` in the backend folder to initialize Node.js.

3. **Install Required Packages:**

- **Frontend:** axios, bootstrap, @mui/material, react-router-dom
- **Backend:** express, mongoose, cors, bcryptjs, dotenv, jsonwebtoken, nodemon

```
{
  "name": "task1",
  "version": "0.1.0",
  "proxy": "http://localhost:8000",
  "private": true,
  "dependencies": {
    "@emotion/react": "^11.11.1",
    "@emotion/styled": "^11.11.0",
    "@testing-library/jest-dom": "^5.16.5",
    "@testing-library/react": "^13.4.0",
    "@testing-library/user-event": "^13.5.0",
    "axios": "^1.4.0",
    "bootstrap": "^5.2.3",
    "mdb-react-ui-kit": "^6.1.0",
    "react": "^18.2.0",
    "react-bootstrap": "^2.7.4",
    "react-dom": "^18.2.0",
    "react-router-dom": "^6.11.2",
    "react-scripts": "5.0.1",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

```

1  {
2    "name": "backend",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "start": "nodemon index.js"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "dependencies": {
13     "bcrypt": "^5.1.0",
14     "cors": "^2.8.5",
15     "express": "^4.18.2",
16     "express-session": "^1.17.3",
17     "mongoose": "^7.1.1",
18     "nodemon": "^2.0.22"
19   }
20 }

```

4. **Create .env files** in both frontend and backend to store environment-specific configurations securely (e.g., DB URI, JWT secret).

This milestone ensures that your application has a clean separation of concerns between UI, business logic, and data handling.

## Milestone 2 : Backend API and Authentication Development

The backend handles the logic of complaint submission, user management, and chat messaging. Express.js powers the RESTful APIs, while MongoDB serves as the data layer.

### • Steps:

1. **Create Express Server** (index.js or server.js):
  - Setup middleware: body-parser, cors, dotenv
  - Define routes for users, complaints, assignments, and messages
2. **Implement Mongoose Models:**
  - User: with fields like name, email, password, phone, userType
  - Complaint: contains userId, description, address, status

- Assignment: links agentId and complaintId
- Message: includes name, complaintId, and message text

### 3. **JWT-based Authentication:**

- Secure login and protected routes using JSON Web Tokens
- Middleware to verify token and extract user details

### 4. **Role-Based Access Control:**

- Different access for admin (Raghu), agent (Manoj), and customer (Sreenath)

### 5. **Error Handling and Validation:**

- Centralized error handlers for clean API responses
- Use express-validator for input validation (optional)

This phase ensures that all backend logic, including user roles and security, is implemented and tested through tools like Postman.

**For Reference :** <https://drive.google.com/drive/folders/1Lh1hNjgYzn-AwVPFlobyvxVwXzU6Gp37?usp=sharing>

## **Milestone 3 : Database Schema and Real-Time Messaging Integration**

This phase strengthens the backend by defining precise schema relationships and enabling real-time communication between users and agents.

### **• Steps:**

#### 1. **Database Setup:**

- Connect MongoDB Atlas to backend via Mongoose
- Ensure indexed collections and reference keys (userId, complaintId) are correctly linked

#### 2. **Schema Relations:**

- Complaints are tied to users (userId)
- Assigned complaints are tied to agents (agentId)
- Messages are tied to both complaints and users

#### 3. **Real-Time Communication:**

- Integrate **Socket.IO** to allow users to chat with agents in real-time
- Use `socket.on()` and `socket.emit()` to broadcast messages

#### 4. **Store Messages:**

- Every message sent is also stored in the message collection for record-keeping

This milestone ensures that the app supports asynchronous messaging and a real-time experience for users needing quick issue resolution.

### **Milestone 4 : Frontend Development with React.js**

The frontend acts as the bridge between the user and backend services. It includes intuitive forms, dashboards, chat interfaces, and role-based views.

#### **Steps:**

##### 1. **UI Design:**

- Use **Material UI** and **Bootstrap** to create consistent and responsive design
- Structure components: Header, Sidebar, ComplaintForm, ComplaintList, ChatBox, AdminPanel

##### 2. **Routing and Navigation:**

- Use `react-router-dom` for navigating between pages like Login, Register, Dashboard, Chat, etc.

##### 3. **Axios Integration:**

- Use `Axios` to send HTTP requests to backend APIs
- Configure global error handlers and interceptors for token management

##### 4. **Authentication & Role Handling:**

- Manage user sessions with JWT stored in `localStorage`
- Redirect users to their respective dashboards based on role

##### 5. **Dynamic Dashboards:**

- **Customer (Sreenath):** Submit complaints, track status, chat with agents
- **Agent (Manoj):** View assigned complaints, resolve, chat with customers
- **Admin (Raghu):** Assign complaints, manage users, oversee system operations



**For Reference :**

<https://drive.google.com/drive/folders/1eEaV1wtfpOFRmVjiUTfR4XfMI9Z3mp1b?usp=sharing>

## **Milestone 5 : Testing, Debugging**

Once both backend and frontend are developed, rigorous testing is performed to ensure all components work seamlessly together.

### **• Steps:**

#### **1. Manual Testing:**

- Use different test accounts for each role
- File, assign, resolve complaints and simulate chat flows

#### **2. Bug Fixing:**

- Use browser dev tools and console logs for debugging
- Handle edge cases like invalid logins, unauthorized access, missing fields

#### **3. UI Polishing:**

- Improve responsiveness for mobile/tablet views
- Add loaders, success messages, and error toasts

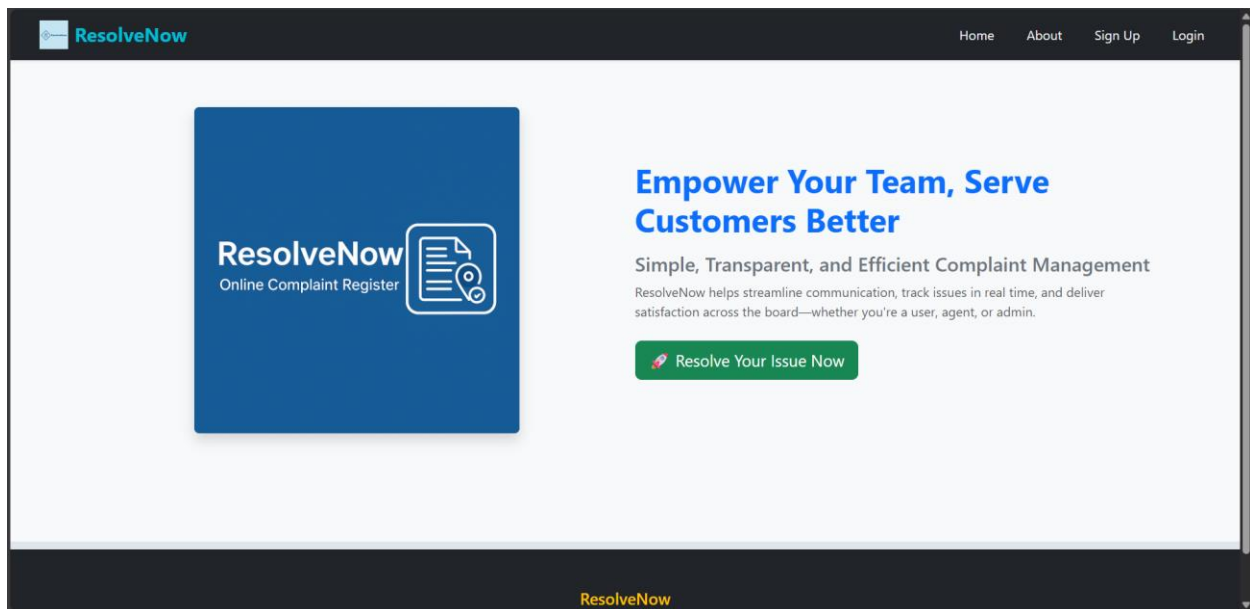
#### **4. Deployment Preparation:**

- Prepare build/ directory for frontend
- Set environment variables for production
- Use services like **Render**, **Vercel**, or **Heroku** for hosting

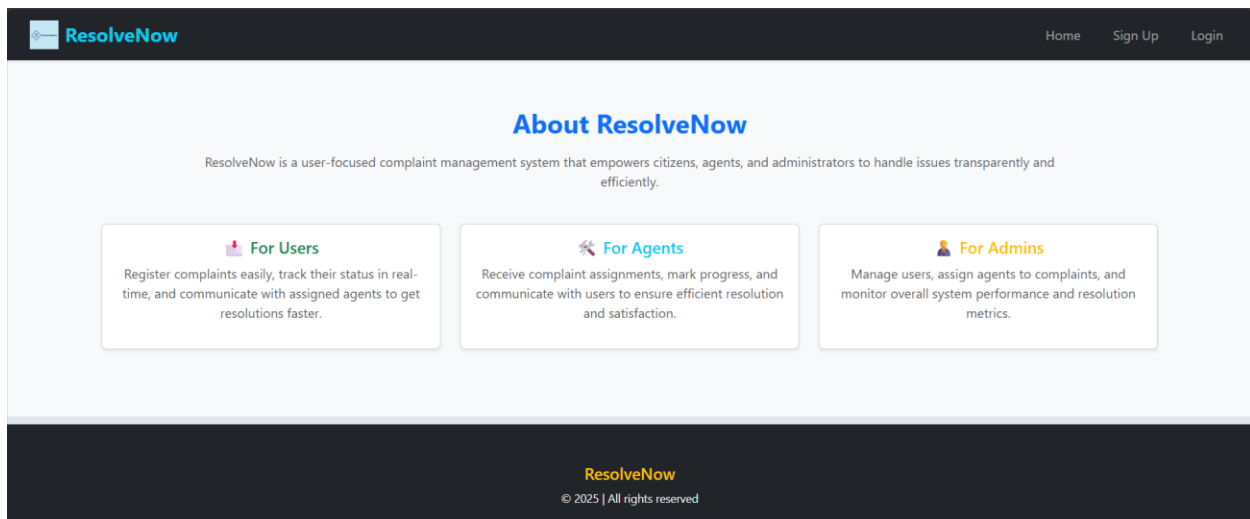
## **Milestone 6 : Project Implementation**

Once the development cycle for **ResolveNow – Online Complaint Registration and Management System** was completed, we proceeded with the **final testing and implementation phase**. This stage ensured that all integrated modules worked together flawlessly and that the user experience was consistent across different user roles.

- Home Page



- About Page



- Login Page

**ResolveNow** Home Sign Up Login

### Login to File a Complaint


Enter your email and password below.

Email

reddysaikumar254@gmail.com

Password

.....

 Login

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

- Signup Page



# Sign Up to File Complaints

Fill in your details below

Full Name

Email

Password

Phone

User Type



Register

Already have an account? [Login](#)

- Common Page For Complaint

Hi, Sake Sreenath

Register Complaint

View Status

Log Out

Name

Address

City

State

Pincode

Status

Pending

Description

Describe your complaint in detail...

Submit

Clear

- Status page

Hi, Sake Sreenath

Register Complaint

View Status

Log Out

Sreenath

Address: 554-1

City: Kadapa

State: AP

Pincode: 516316

Comment: I had lost my Wallet at New Bus Stand

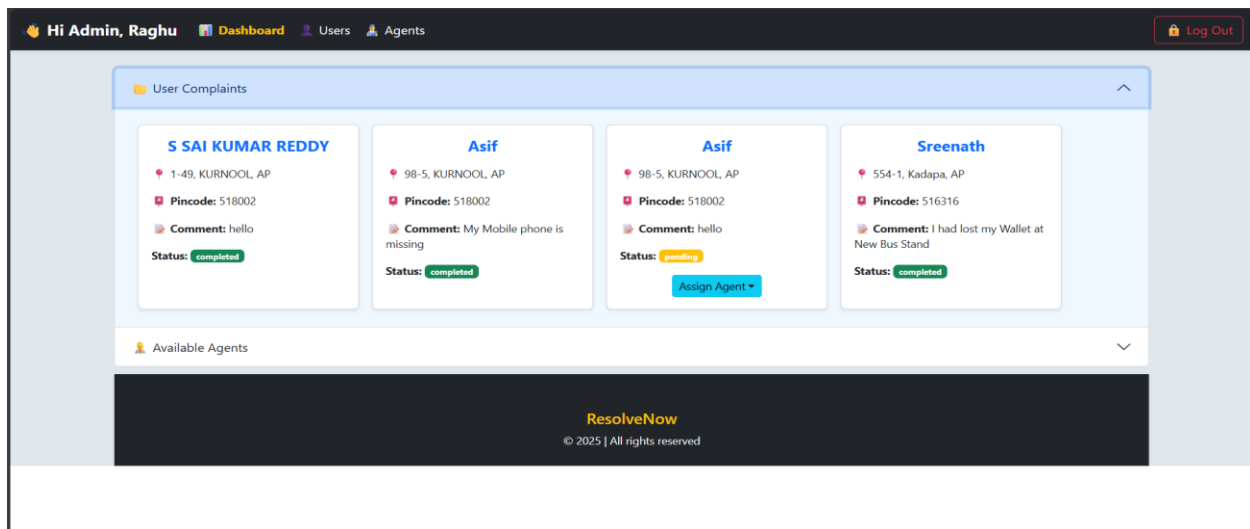
Status: completed

Message

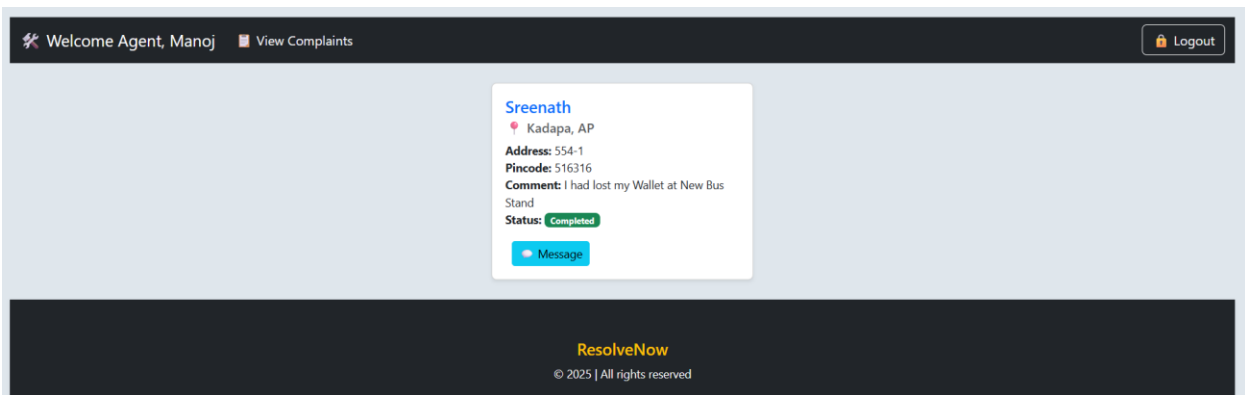
ResolveNow

© 2025 | All rights reserved

- Admin Page



- Agent Page



For any further doubts or assistance regarding the **ResolveNow – Online Complaint Registration and Management System**, you can refer to the complete source code and implementation files available in the following Google Drive folder:

- **Source Code & Resources :**

[https://drive.google.com/drive/folders/1FWvLoHL\\_mV5m70bbVIJXnSjL3\\_bSwaER?usp=sharing](https://drive.google.com/drive/folders/1FWvLoHL_mV5m70bbVIJXnSjL3_bSwaER?usp=sharing)

Additionally, you can watch the full application walkthrough and working demo at:

- **Project Demo Video :**

[https://drive.google.com/file/d/13\\_4Fitepf2IIUsbPo1TaUw8JXfLrFYWX/view?usp=sharing](https://drive.google.com/file/d/13_4Fitepf2IIUsbPo1TaUw8JXfLrFYWX/view?usp=sharing)

