**ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM**

# INTRODUCTION

An online complaint registration and management system is a software application or platform that allows individuals or organizations to submit and track complaints or issues they have encountered. It can help optimize the complaint handling process and empower organizations to develop a safety management system to efficiently resolve customer complaints, while staying in line with industry guidelines and regulatory compliance obligations. This project was built by a team of dedicated professionals, each specializing in different areas.

The Project manager [Priyanka V] leads the project with a focus on end-to-end development and integration.

The Backend Developer, [Sherlin Stefina Catherine S ] focuses on server-side operations, ensuring smooth data management and security.

The Frontend Developer, [Suganthi S] implements and refines the user interface, while the UI/UX Designer

The Database engineer, [Thirisha M] focus on designing, implementing and maintaining the data base.

# PROJECT OVERVIEW

Purpose:

The Online Complaint Registration and Management System is designed to streamline the process of submitting, managing, and resolving complaints in a centralized platform. This system aims to simplify the way users report issues and track their resolutions, ensuring transparency and efficiency in complaint handling. By automating and digitizing this process, the system enhances communication between users and administrative departments, ultimately improving service quality and response times.

Features:

* User Registration and Login: Users can register and log in securely to access the complaint system.
* Complaint Submission: Users can submit complaints, specifying categories (e.g., technical, maintenance) and priority levels to facilitate effective sorting and handling.
* Complaint Tracking: Users receive updates on the status of their complaints, from submission to resolution.
* Admin Dashboard: Admins can view, prioritize, and manage complaints, with filters for sorting by category, status, or priority.
* Notifications: Automated notifications alert users to any updates or changes in complaint status.

# TECHNICAL ARCHITECTURE



The technical architecture of our online complaint registration and management system follows a client-server model, where the frontend serves as the client and the backend acts as the server. The frontend encompasses not only the user interface and presentation but also incorporates the axios library to connect with backend easily by using RESTful Apis.

The frontend utilizes the bootstrap and material UI library to establish real-time and better UI experience for any user whether it is agent, admin or ordinary user working on it.

On the backend side, we employ Express.js frameworks to handle the server-side logic and communication.

For data storage and retrieval, our backend relies on MongoDB. MongoDB allows for efficient and scalable storage of user data, including user profiles, for complaints registration, etc. It ensures reliable and quick access to the necessary information during registration of user or any complaints.

Together, the frontend and backend components, along with socket.io, Express.js, WebRTC API, and MongoDB, form a comprehensive technical architecture for our video conference app. This architecture enables real-time communication, efficient data exchange, and seamless integration, ensuring a smooth and immersive video conferencing experience for all users.

# SETUP INSTRUCTIONS

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

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.

# FOLDER STRUCTURE:





The first image is of frontend part which is showing all the files and folders that have been used in UI development

The second image is of Backend part which is showing all the files and folders that have been used in backend development

RUNNING THE APPLICATION:

To run the application locally:

Start the frontend server by navigating to the client directory and executing npm start.

Start the backend server by navigating to the server directory and executing npm start.

# API DOCUMENTATION**:**

1. **Customer/Ordinary User:**
   * **Role:** Create and manage complaints, interact with agents, and manage profile information.
   * **Flow:**

**Registration and Login:**

* + - * Create an account by providing necessary information such as email and password.
      * Log in using the registered credentials.

**Complaint Submission:**

* + - * Fill out the complaint form with details of the issue, including description, contact information, and relevant attachments.
      * Submit the complaint for processing.

**Status Tracking:**

* + - * View the status of submitted complaints in the dashboard or status section.
      * Receive real-time updates on the progress of complaints.

**Interaction with Agents:**

* + - * Connect with assigned agents directly using the built-in messaging feature.
      * Discuss complaints further and provide additional information or clarification.

**Profile Management:**

Manage personal profile information, including details and addresses.

1. **Agent:**
   * **Role:** Manage complaints assigned by the admin, communicate with customers, and update complaint statuses.
   * **Flow:**

**Registration and Login:**

* + - * Create an account using email and password.
      * Log in using the registered credentials.

**Complaint Management:**

* + - * Access the dashboard to view and manage complaints assigned by the admin.
      * Communicate with customers regarding their complaints through the chat window.

**Status Update:**

* + - * Change the status of complaints based on resolution or progress.
      * Provide updates to customers regarding the status of their complaints.

**Customer Interaction:**

* + - * Respond to inquiries, resolve issues, and address feedback from customers.

1. **Admin:**
   * **Role:** Oversee the overall operation of the complaint registration platform, manage complaints, users, and agents, and enforce platform policies.
   * **Flow:**

**Management and Monitoring:**

* + - * Monitor and moderate all complaints submitted by users.
      * Assign complaints to agents based on workload and expertise.

**Complaint Assignment:**

* + - * Assign complaints to the desired agents for resolution.
      * Ensure timely and efficient handling of complaints.

**User and Agent Management:**

* + - * Manage user and agent accounts, including registration, login, and profile information.
      * Enforce platform policies, terms of service, and privacy regulations.

**Continuous Improvement:**

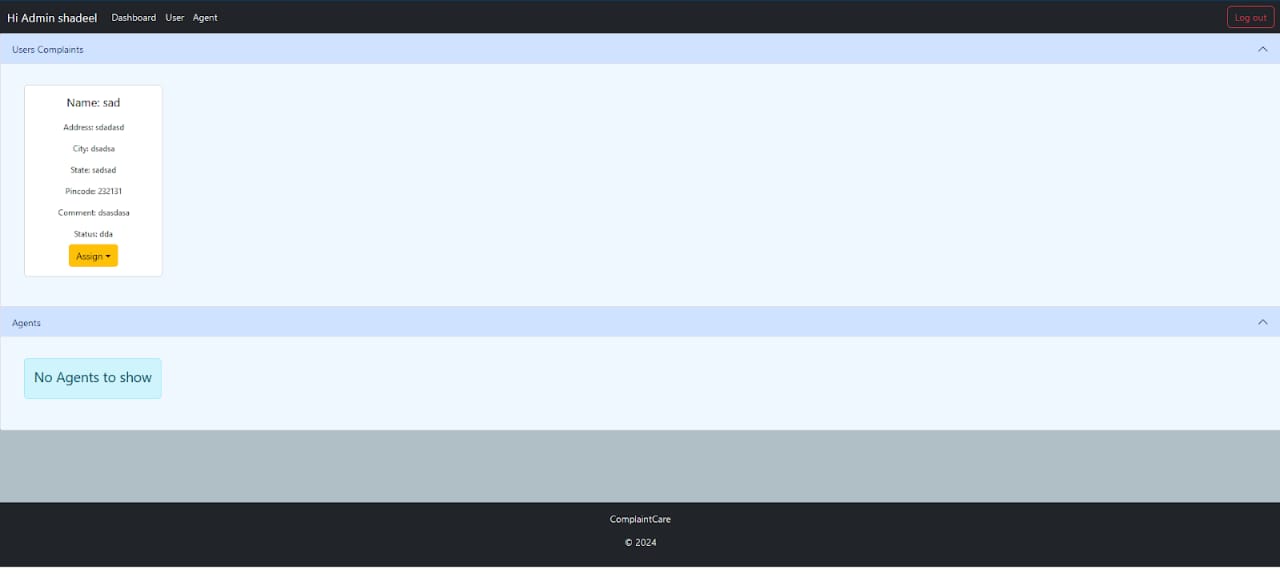
* + - * Implement measures to improve the platform's functionality, user experience, and security measures.
      * Address any issues or concerns raised by users or agents for better service delivery.

AUTHENTICATION:

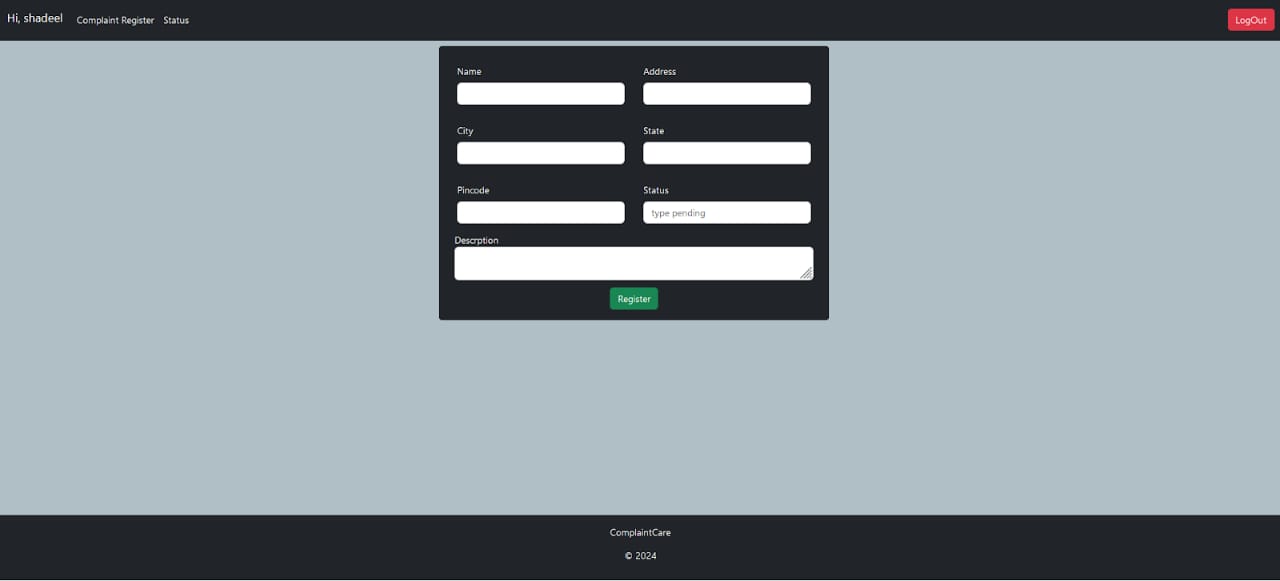
Authentication is implemented using JSON Web Tokens (JWT). Upon login, a JWT token is issued, which is stored on the client side and used to authenticate subsequent requests. Role-based authorization is implemented to distinguish user and admin functionalities.

USER INTERFACE:

ADMIN DASHBOARD



USER DASHBOARD



Testing:-

**Testing Strategy:**

Unit and integration tests are written using Jest and Mocha for both frontend components and backend APIs. Postman is used for manual API testing.

# SCREENSHOTS AND DEMO:

Before starting to work on this project, let’s see the demo.

**Project demo:** [**https://drive.google.com/file/d/1YwXaHRBZJL\_V7dcEK8SOmtPWZasAxccm/view?usp=drive\_link**](https://drive.google.com/file/d/1YwXaHRBZJL_V7dcEK8SOmtPWZasAxccm/view?usp=drive_link)

Use the code in: <https://github.com/awdhesh-student/complaint-registery.git>

or follow the videos below for better understanding.



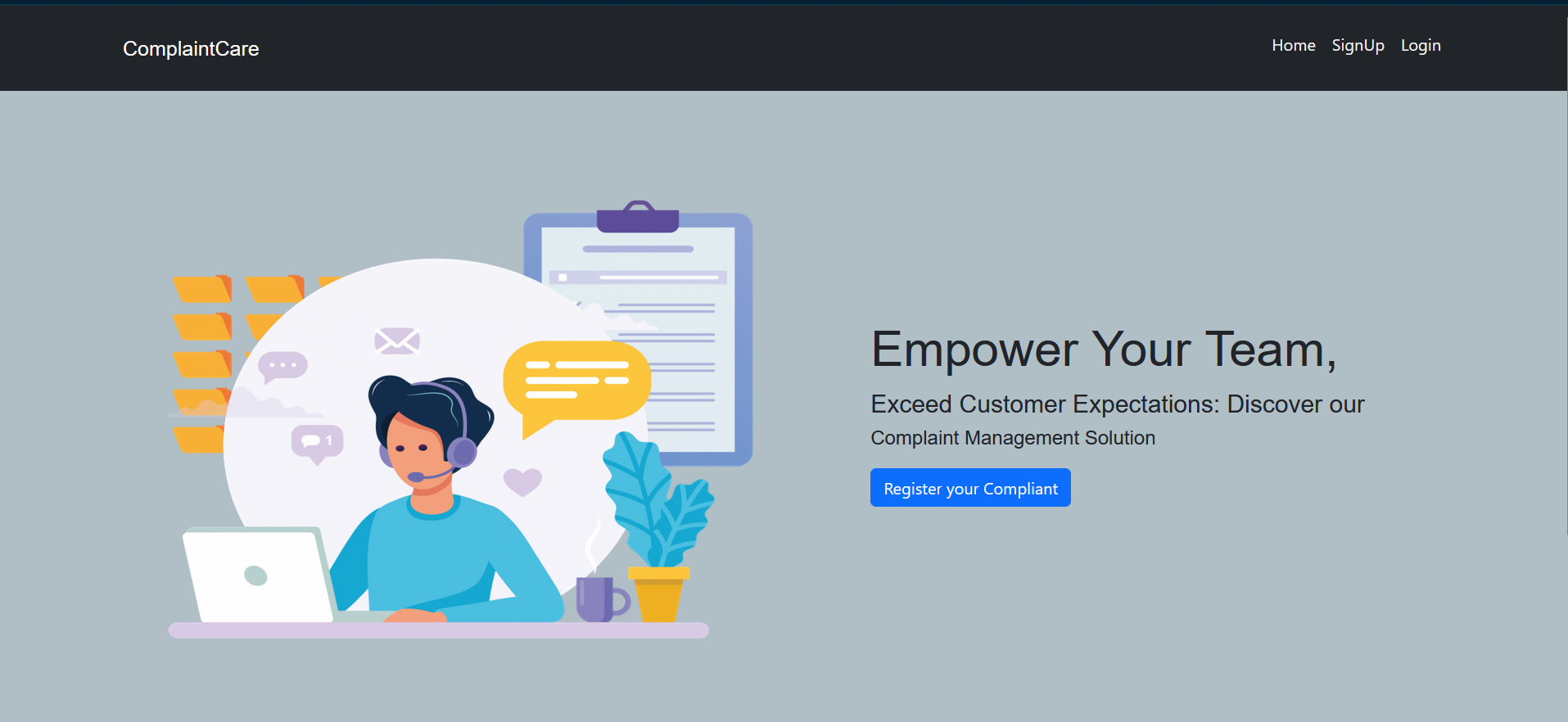
After the installation of all the libraries, the package.json files for the backend looks like the one mentioned below.



**Project Implementation:**

On completing the development part, we then run the application one last time to verify all the functionalities and look for any bugs in it. The user interface of the application looks a bit like the one’s provided below

* Landing Page



* Login Page



* Registration Page



Known Issues:

Issue 1: Delay in notification delivery under high traffic.

Issue 2: Limited analytics features in the admin dashboard.

Future Enhancements:

Implement AI-based complaint categorization.

Add mobile support for Android and iOS.

Improve real-time notifications using WebSockets.