NAAN MDUHALVAN PROJECT REPORT

TITLE: ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM

COLLEGE NAME: AALIM MUHAMMED SALEGH COLLEGE OF ENGINEERING
COLLEGE CODE: 1101

MEMBERS:

1. MOHAMED FAHIM (11021205022)
2. MOHAMED ASIF (110121205021)
3. MOHAMED AZEEM (110121205033)
4. MOHAMED FAWZAN (11012120523)

INTRODUCTION

An online complaint registration and management system is a dynamic software solution designed for individuals or organizations to submit, track, and resolve complaints or issues seamlessly. This system enhances the efficiency of complaint handling, empowering organizations to establish a robust safety management framework that not only accelerates complaint resolution but also aligns with industry standards and regulatory compliance requirements. By centralizing complaint handling, it optimizes processes, facilitates transparent communication, and bolsters customer satisfaction through timely resolutions.

Key features of an online complaint registration and management system include:

1. **User Registration and Account Management**: New users can create secure accounts, providing access to complaint submission and real-time tracking capabilities. Returning users can easily log in to manage existing complaints and review resolution histories.
2. **Intuitive Complaint Submission**: Users can lodge complaints with ease, entering essential details such as their name, contact information, location, and a thorough description of the issue. This functionality ensures that all necessary data is collected from the start, supporting a quick and accurate resolution process.
3. **Real-Time Tracking and Notifications**: Users can monitor the progress of their complaints at every stage. Automated notifications, sent via email or SMS, inform users of key status updates, from complaint acknowledgment to resolution, ensuring they stay informed throughout the process.
4. **Direct User-Agent Interaction**: To enhance communication, users have the option to engage directly with the assigned support agent, enabling them to provide additional information or clarification on the complaint. This feature fosters a collaborative approach to problem-solving and improves response times.
5. **Intelligent Complaint Routing and Assignment**: Complaints are automatically routed to the appropriate department or personnel based on the nature of the issue. Advanced algorithms may be employed to streamline this process, ensuring that complaints are assigned to the most suitable team members, maximizing efficiency and resource allocation.
6. **Enhanced Security and Confidentiality**: The system prioritizes data protection by implementing robust security measures, including user authentication, encryption, and access controls. Compliance with data protection regulations guarantees that user information and complaint details remain confidential, enhancing user trust.

DESCRIPTION

The **Online Complaint Registration and Management System** is an intuitive software platform built to simplify and enhance the experience of lodging, monitoring, and resolving complaints for both individuals and organizations. This centralized solution enables users to effortlessly register issues, view real-time progress updates, and communicate directly with designated agents to resolve their concerns. Key features, including automated notifications, smart complaint routing, and advanced security protocols, ensure that each complaint is managed promptly and securely, with a strong focus on safeguarding user information and delivering a smooth, reliable experience for all users.

SCENARIO

1. User Registration and Login

- John, facing an issue with a product he recently bought, decides to report it through the complaint management system.
- Account Creation: He visits the system's website, clicks on "Sign Up," and completes the registration form with his name, email, and a secure password.
- Email Verification: After submitting, he receives a verification email to confirm his account.
- Logging In: Once verified, he logs into the system using his credentials, gaining access to his dashboard.

2. Complaint Submission

- Navigating the Dashboard: On logging in, John sees an option to submit a complaint.
- Filing Details: He selects "Submit Complaint," then describes the product defect in detail, attaching images of the defect and providing his contact info and purchase date.
- Submission: After reviewing, John submits the complaint, formally registering his issue in the system.

3. Tracking and Notifications

- Confirmation and Status Updates: John immediately receives confirmation that his complaint is registered.
- Real-Time Tracking: Through the "My Complaints" section, John can monitor his complaint status, from acknowledgment to resolution.
- Email Notifications: At each stage, from assignment to progress updates, John receives email alerts, keeping him informed throughout.

4. Interaction with Assigned Agent

- Agent Assignment: Sarah, a customer service agent, is assigned to manage John's complaint.
- Direct Communication: Sarah reviews John's complaint details and contacts him through the platform's built-in messaging feature.
- Discussion: John receives a message notification and engages in a chat with Sarah to provide more context. Sarah reassures John that the company will thoroughly investigate the issue for a prompt resolution.

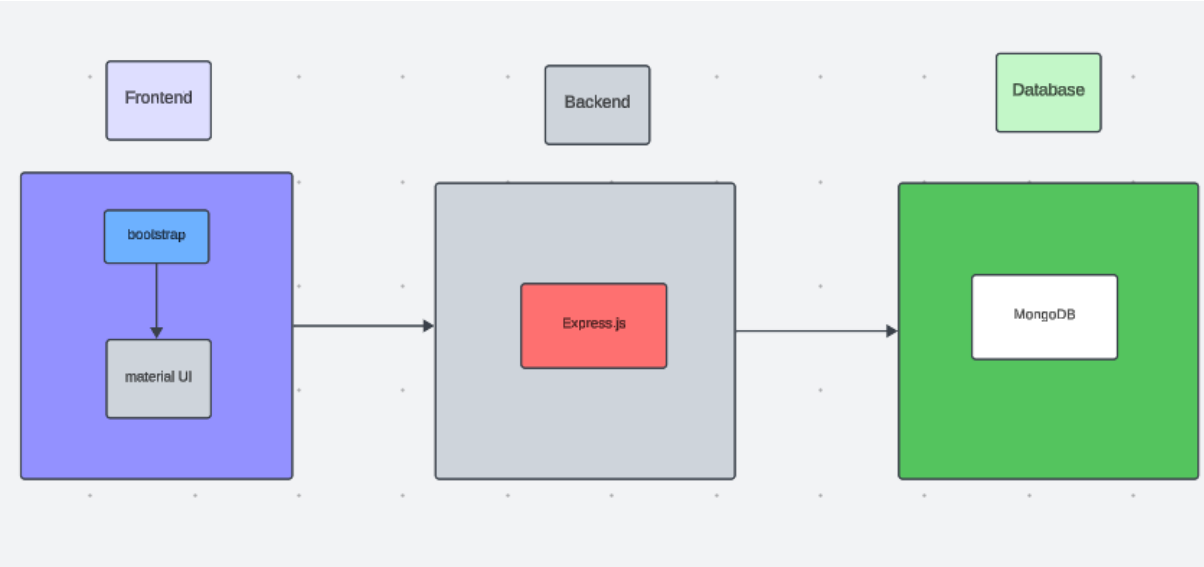
5. Resolution and Feedback

- Investigation and Solution: The company identifies the defect and offers John a choice of a replacement or a refund.
- Resolution Notification: John receives a notification with resolution details and next steps.
- Feedback: After resolving his complaint, John shares positive feedback, appreciating the quick response and Sarah's professionalism.

6. Admin Management

- Monitoring: The system administrator oversees all registered complaints, ensuring that each complaint moves efficiently through the resolution process.
- Assignment Control: The admin assigns complaints based on agents' expertise and availability.
- Compliance Assurance: They maintain compliance with platform policies, ensuring data security, confidentiality, and regulatory standards are upheld.

TECHNICAL ARCHITECTURE



The **technical architecture** of our **Online Complaint Registration and Management System** adopts a **client-server model**, with a clear separation between frontend and backend functionalities to provide a seamless user experience.

1. Frontend Architecture

The **frontend** acts as the client, offering an intuitive user interface for agents, admins, and general users.

UI Libraries: We use **Bootstrap** and **Material UI** to create a responsive, user-friendly interface that ensures a consistent experience across devices.

RESTful API Integration: The frontend leverages the Axios library to facilitate communication with the backend, allowing users to submit complaints, track progress, and interact with agents through RESTful API calls.

2. Backend Architecture

The **backend** serves as the server, powered by the **Express.js framework**. This enables efficient handling of server-side logic, complaint processing, and communication between users and agents.

Data Storage: We use **MongoDB** for data storage, which provides scalable and efficient storage for user data, complaint records, and communication logs, enabling quick access to user profiles and complaint information.

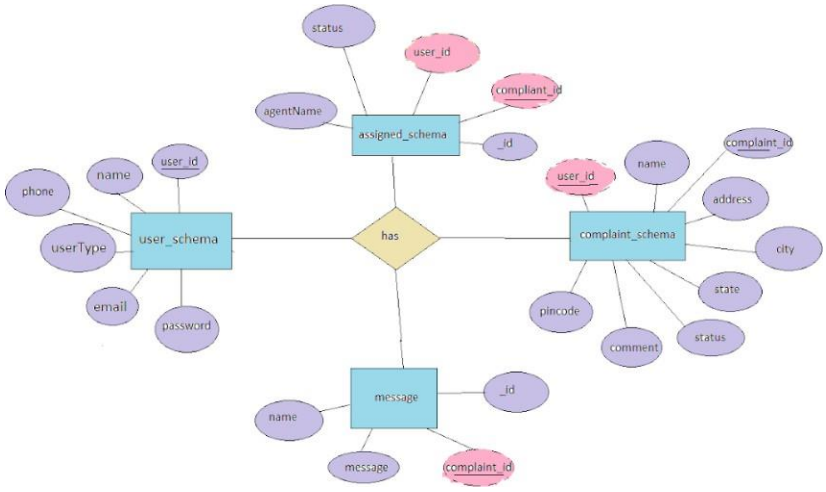
3. Real-Time Communication

Socket.io is integrated into the system, enabling real-time updates, notifications, and messaging between users and agents, enhancing responsiveness in complaint handling.

This allows for real-time status updates on complaints and facilitates immediate interactions within the system's built-in messaging feature.

Together, the frontend and backend components form a robust, scalable technical architecture for the **Online Complaint Registration and Management System**. This architecture enables real-time complaint tracking, seamless data exchange, and an efficient, interactive user experience for all users.

ER DIAGRAM



This is the er diagram of the project which shows the relationship between user and agent
It shows how user which have required fields can raise a complaint by fillings required fields. It illustrates how these entities relate to each other, helping us understand the underlying database structure and the flow of information within the app. He / She can also communicate with the agent with chat window which follows the message schema which uses **USERID** and **COMPLAINTID** from other schemas.

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

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.

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, status of the complaints, and user interfaces for the admin dashboard.
For making better UI we have also used some libraries like material UI and boostrap.

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

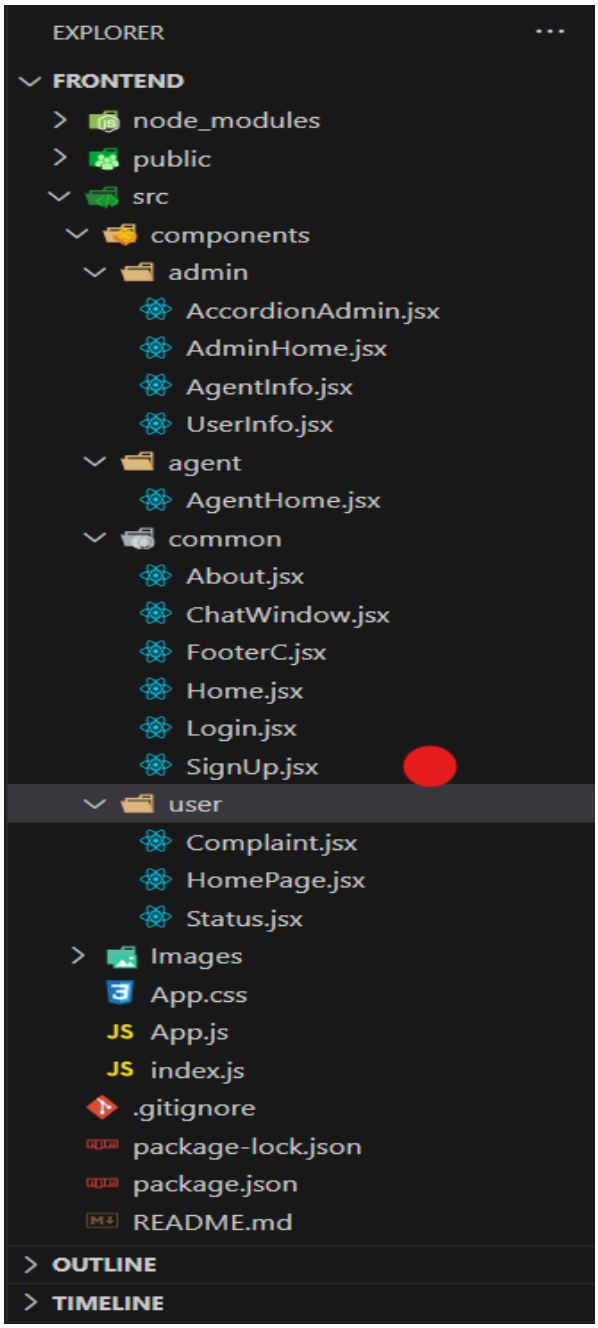
Install Dependencies:

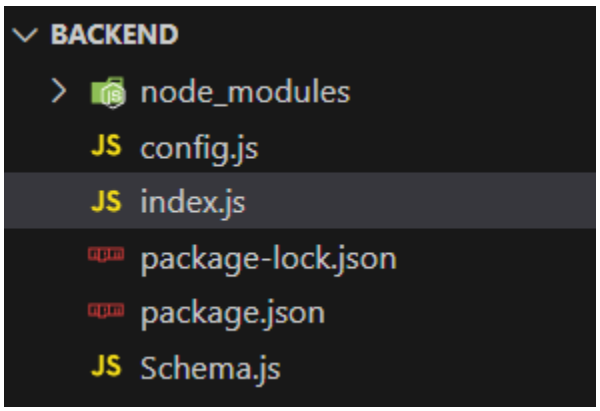
• Navigate into the cloned repository directory:
    cd complaint-registery
• Install the required dependencies by running the following commands:
    cd frontend
    npm install
    cd ../backend
    npm install
Start the Development Server:
    • To start the development server, execute the following 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


EXPLORER
FRONTEND
node_modules
public
src
components
admin
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
About.jsx
ChatWindow.jsx
FooterC.jsx
Home.jsx
Login.jsx
SignUp.jsx
user
Complaint.jsx
HomePage.jsx
Status.jsx
Images
App.css
App.js
index.js
.gitignore
package-lock.json
package.json
README.md
OUTLINE
TIMELINE


BACKEND
node_modules
config.js
index.js
package-lock.json
package.json
Schema.js

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

APPLICATION FLOW

Online Complaint Registration and Management System

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

1. **Registration and Login:**
    - Create an account by providing necessary information such as email and password.
    - Log in using the registered credentials.
2. **Complaint Submission:**
    - Fill out the complaint form with details of the issue, including description, contact information, and relevant attachments.
    - Submit the complaint for processing.
3. **Status Tracking:**
    - View the status of submitted complaints in the dashboard or status section.
    - Receive real-time updates on the progress of complaints.
4. **Interaction with Agents:**
    - Connect with assigned agents directly using the built-in messaging feature.
    - Discuss complaints further and provide additional information or clarification.
5. **Profile Management:**
    - Manage personal profile information, including details and addresses.

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

1. **Registration and Login:**
    - Create an account using email and password.
    - Log in using the registered credentials.
2. **Complaint Management:**
    - Access the dashboard to view and manage complaints assigned by the admin.
    - Communicate with customers regarding their complaints through the chat window.
3. **Status Update:**
    - Change the status of complaints based on resolution or progress.
    - Provide updates to customers regarding the status of their complaints.
4. **Customer Interaction:**
    - Respond to inquiries, resolve issues, and address feedback from customers.

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

1. **Management and Monitoring:**
    - Monitor and moderate all complaints submitted by users.
    - Assign complaints to agents based on workload and expertise.
2. **Complaint Assignment:**
    - Assign complaints to the desired agents for resolution.
    - Ensure timely and efficient handling of complaints.
3. **User and Agent Management:**
    - Manage user and agent accounts, including registration, login, and profile information.
    - Enforce platform policies, terms of service, and privacy regulations.
4. **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.

PROJECT FLOW

Milestone 1:

Project Setup and Configuration:

1. **Create project folders and files:**

   Now, firstly create the folders for frontend and backend to write the respective code and install the essential libraries.

   - Client folders.
   - Server folders

2. **Install required tools and software:**

   For the backend to function well, we use the libraries mentioned in the prerequisites. Those libraries includes

   - Node.js.
   - MongoDB.
   - Bcrypt
   - Body-parser

   Also, for the frontend we use the libraries such as

   - React Js.
   - Material UI

- Bootstrap
- Axios

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

```
{
  "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"
    ]
  }
}
```

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

Milestone 2:

Backend Development:

- **Set Up Project Structure:**
  - Create a new directory for your project and set up a package.json file using npm init command.
  - Install necessary dependencies such as Express.js, Mongoose, and other requiredpackages.

- **Set Up Project Structure:**
  - Create a new directory for your project and set up a package.json file using npm init command.
  - Install necessary dependencies such as Express.js, Mongoose, and other requiredpackages.
- **Create Express.js Server:**
  - Set up an Express.js server to handle HTTP requests and serve API endpoints.
  - Configure middleware such as body-parser for parsing request bodies and cors forhandling cross-origin requests.

- **Define API Routes:**
  - Create separate route files for different API functionalities such as authentication, stock actions, and transactions.
  - Implement route handlers using Express.js to handle requests and interact with thedatabase.

- **Implement Data Models:**
  - Define Mongoose schemas for the different data entities like Bank, users, transactions, deposits and loans.
  - Create corresponding Mongoose models to interact with the MongoDB database.
  - Implement CRUD operations (Create, Read, Update, Delete) for each model toperform database operations.
- **User Authentication:**
  - Implement user authentication using strategies like JSON Web Tokens (JWT) orsession-based authentication.
  - Create routes and middleware for user registration, login, and logout.
  - Set up authentication middleware to protect routes that require user authentication.

- **Handle new transactions:**
  - Allow users to make transactions to other users using the user's account

id.
- Update the transactions and account balance dynamically in real-time.

- **Admin Functionality:**
  - Implement routes and controllers specific to admin functionalities such as fetching all the data regarding users, transactions, stocks and orders.

- **Error Handling:**
  - Implement error handling middleware to catch and handle any errors that occurduring the API requests.
  - Return appropriate error responses with relevant error messages and HTTP statuscodes.

Milestone 3:

Database Development

1. **User Schema:**
   - The user schema defines the structure of user data stored in the database. It includes fields such as name, email, password, phone, and userType.
   - Each user must provide a name, email, password, phone number, and userType (e.g., customer, agent, admin).
   - User data is stored in the "user_Schema" collection in the MongoDB database.
2. **Complaint Schema:**
   - The complaint schema specifies the format of complaint data registered by users.
   - It contains fields like userId, name, address, city, state, pincode, comment, and status.
   - Complaints are associated with users through the userId field, and each complaint must have a name, address, city, state, pincode, comment, and status.
   - Complaint data is stored in the "complaint_schema" collection in the MongoDB database.
3. **Assigned Complaint Schema:**
   - The assigned complaint schema defines how complaints are assigned to agents for resolution.

- It includes fields such as agentId, complaintId, status, and agentName.
- Each assigned complaint is linked to a specific agent (identified by agentId) and complaint (identified by complaintId).
- The status field indicates the current status of the assigned complaint.
- Assigned complaint data is stored in the "assigned_complaint" collection in the MongoDB database.

4. **Chat Window Schema:**
   - The chat window schema governs the structure of messages exchanged between users and agents regarding specific complaints.
   - It comprises fields like name, message, and complaintId.
   - Messages are associated with a complaint through the complaintId field, allowing for easy tracking and retrieval of chat history for each complaint.
   - Message data is stored in the "message" collection in the MongoDB database.

Milestone 4:

Frontend Development:

1. **Setup React Application:**

   Bringing Customer Care Registry to life involves a three-step development process. First, a solid foundation is built using React.js. This includes creating the initial application structure, installing necessary libraries, and organizing the project files for efficient development. Next, the user interface (UI) comes to life. To start the development process for the frontend, follow the below steps.

   - Install required libraries.
   - Create the structure directories.

2. **Design UI components:**

   Reusable components will be created for all the interactive elements you'll see on screen, from stock listings and charts to buttons and user profiles. Next, we'll implement a layout and styling scheme to define the overall look and feel of the application. This ensures a visually-appealing and intuitive interface. Finally, a navigation system will be integrated, allowing you to effortlessly explore different sections of Customer Care Registry, like making specific complaints or managing your Product complaints.

3. **Implement frontend logic:**

   In the final leg of the frontend development, we'll bridge the gap between the visual interface and the underlying data. It involves the below stages.

   - Integration with API endpoints.
   - Implement data binding.

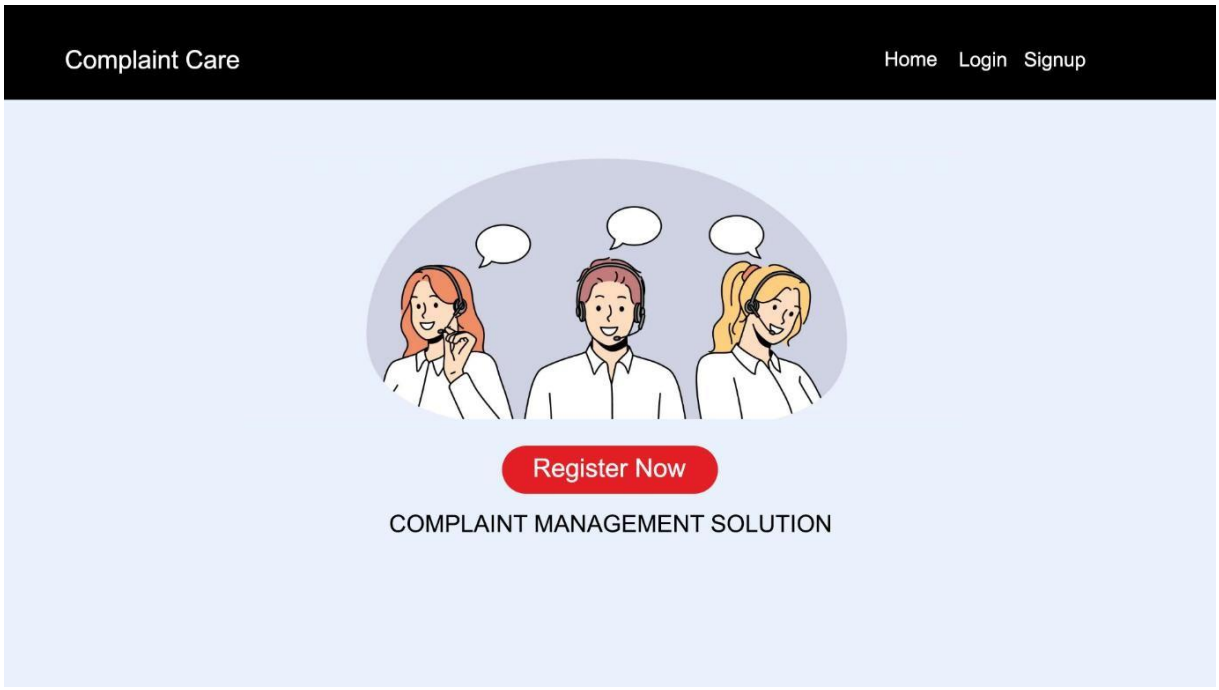
Reference video for frontend code:

Milestone 5:

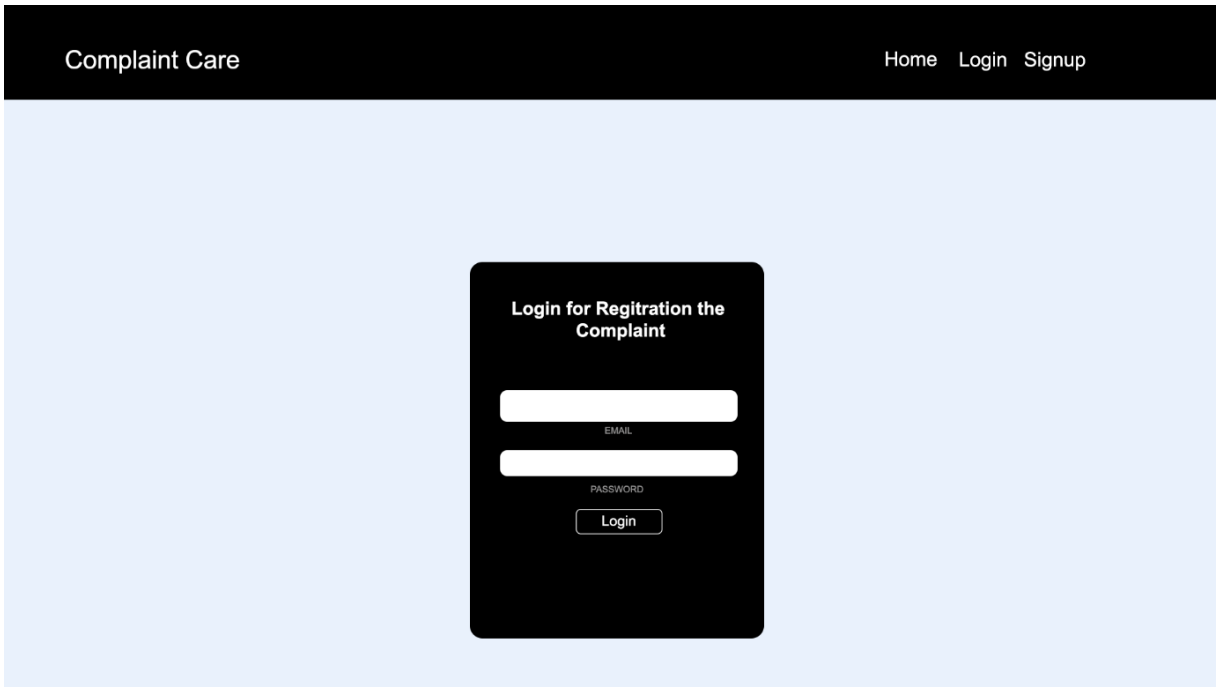
Project Implementation:

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.

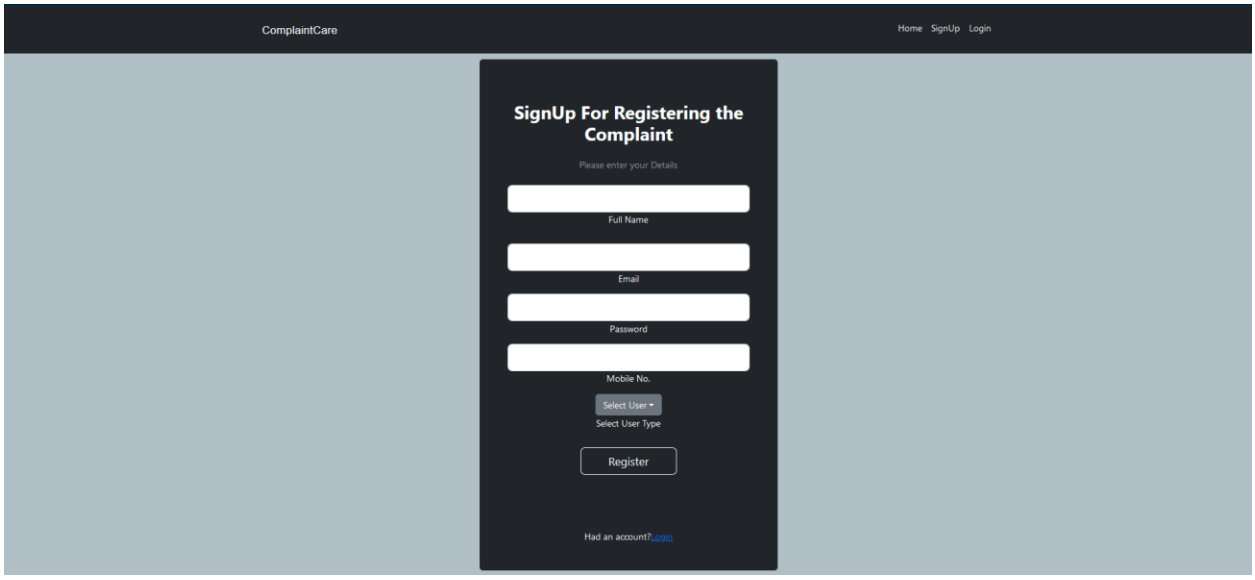
- Landing Page



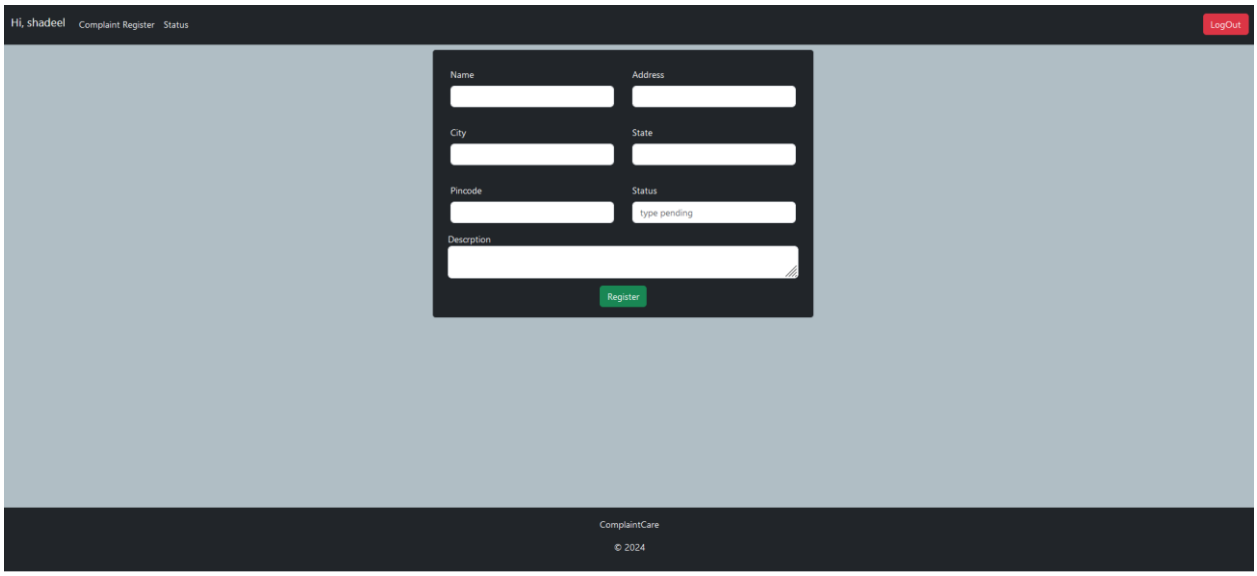
- Login Page



- Registration Page



- Common Dashboard For Complaint



- Admin Dashboard

- Agent Dashboard