ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM

INTRODUCTION

The **Online Complaint Registration and Management System** is a web-based software solution designed to enable individuals or organizations to lodge complaints and monitor their resolution in an efficient and transparent manner. It provides a centralized digital platform for managing grievances, automating complaint handling, and ensuring timely responses — all while maintaining compliance with industry standards and regulatory requirements.
This system empowers organizations to build an efficient and secure complaint handling workflow, improving overall accountability and enhancing customer satisfaction.

It consists of some key features which include:

1. **User Registration**:Users can sign up to create personal accounts, allowing them to log complaints and monitor their status securely.
2. **Complaint Submission**:Users can fill out and submit complaint forms including essential details such as name, issue description, contact information, and location.
3. **Real-Time Tracking & Notifications**:The system provides live status updates and sends automated notifications (e.g., via email or SMS) whenever progress is made on a complaint.
4. **User-Agent Interaction**:Users can communicate directly with the assigned agent to discuss issues, share updates, or seek clarification.
5. **Complaint Assignment & Routing**:Complaints are automatically assigned to appropriate agents or departments using smart routing logic to optimize workload and resolution speed.
6. **Data Security & Confidentiality**:Built-in security features such as user authentication, access controls, and data encryption help safeguard user data and ensure compliance with data privacy regulations.

Description

This platform offers a comprehensive and intuitive interface for managing complaint workflows from start to finish. By integrating submission, tracking, communication, and resolution into a single system, it minimizes delays and reduces manual errors. The system also fosters better engagement between users and agents through real-time chat and updates, ensuring that complaints are resolved promptly. Designed with scalability and security in mind, it can be adapted to suit various organizational needs while keeping user data protected.

SCENARIO

Peter Kavinskey, a customer, recently discovered a defect in a product he purchased online. To resolve the issue, he decides to use the Online Complaint Registration and Management System to file and track his complaint.

1. User Registration and Login

- Peter visits the complaint management system's website and clicks the "Sign Up" button to create a new user account.
- He fills out the registration form, providing his full name, email address, and a secure password.
- Upon submitting the form, Peter receives a verification email and completes his account setup.
- He logs in using his email and password and is redirected to his personal dashboard.

2. Complaint Submission

- After logging in, Peter sees the option to "Submit a Complaint" on the dashboard.
- He clicks the button and fills out the form with relevant information:
  A detailed description of the defect
  Supporting images/documents
  Product purchase date and his contact details
- Once satisfied, he submits the complaint through the system.

3. Complaint Tracking and Notifications

- Peter immediately receives a confirmation message that his complaint has been registered.
- He navigates to the "My Complaints" section to view real-time status updates.
- As his complaint progresses (e.g., assigned, in review, resolved), Peter receives automated email notifications with status updates.

4. Agent Interaction

- The complaint is assigned to Will Jacks, a customer service agent.
- Will reviews the submitted information and reaches out to Peter via the system's built-in chat feature.
- Peter receives a notification and opens the chat to discuss the issue further.

- Will assures Peter that the defect will be reviewed promptly and provides guidance on the next steps.

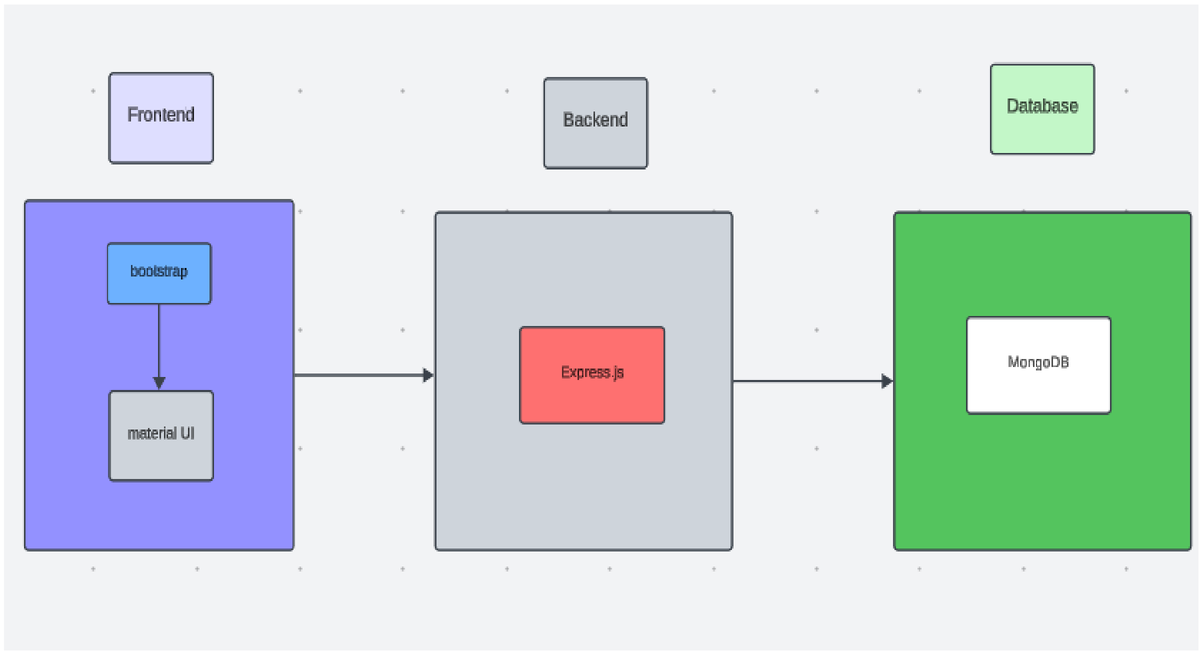
5. Resolution and Feedback

- After reviewing the complaint, the company confirms the defect and offers Peter a replacement or a refund.
- Peter is notified about the resolution outcome and provided with the necessary instructions.
- He shares his feedback, expressing satisfaction with the efficient complaint handling and Will's polite communication.

6. Admin Oversight and Management

- Lara Gean, the system administrator, oversees all complaints submitted through the platform.
- Lara uses the admin panel to assign new complaints to agents like Will, based on their workload and expertise.
- She ensures smooth operation, data security, and compliance with all platform policies and standards.

TECHNICAL ARCHITECTURE



The technical architecture of our online complaint registration and management app 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.

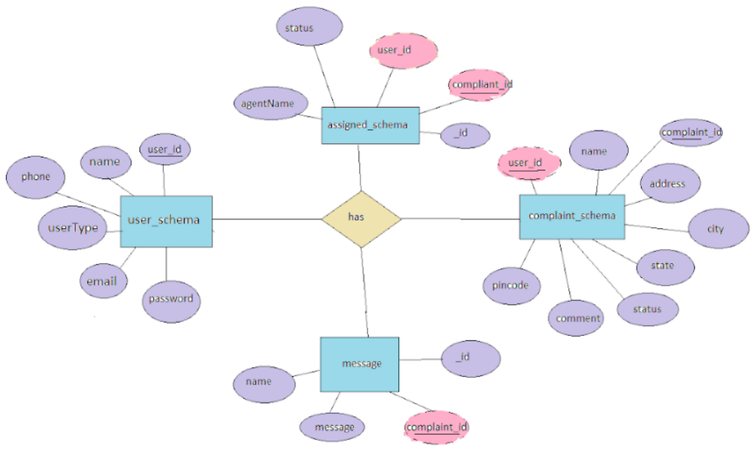
The frontend utilizes the bootstrap and material UI library to establish real-time and better UI experience for any user whether it is agent, admin or ordinary user working on it.

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.

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:

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

To run the existing Video Conference App project downloaded from GitHub:
Follow below steps:
Clone the Repository:

- 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: https://github.com/awdhesh-student/complaint-registery.git

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:

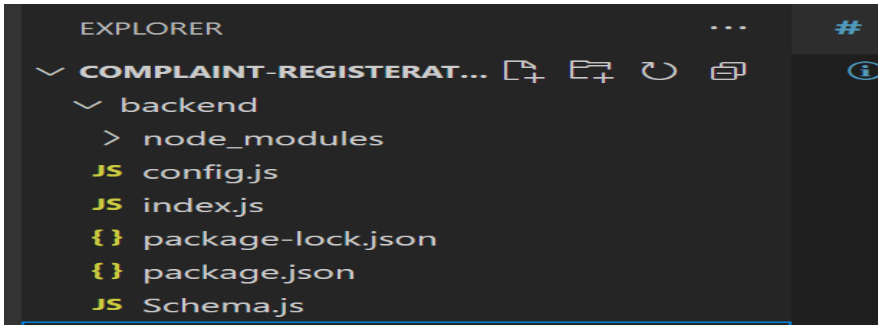
• To start the development server, execute the following command:

```
npm start
```

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

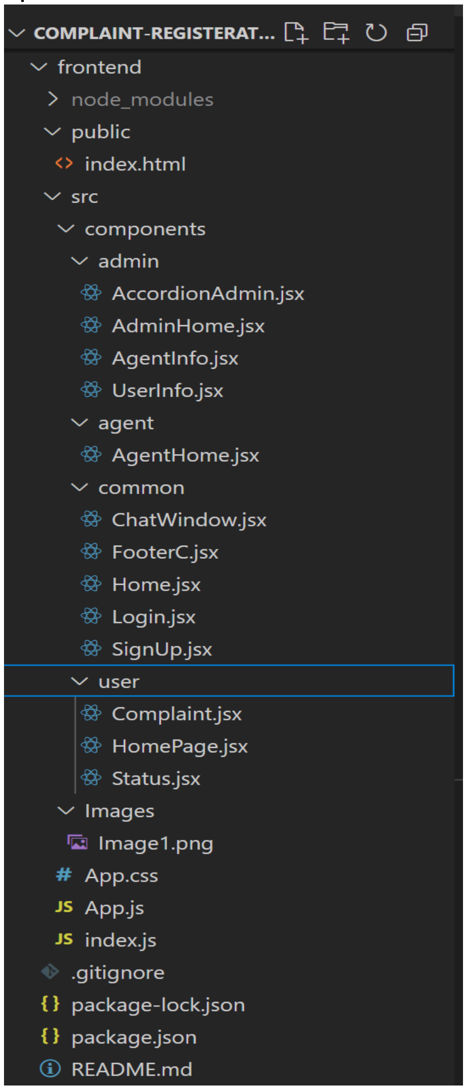
You have successfully installed and set up the online complaint registration and management app on your local machine. You can now proceed with further customization, development, and testing as needed.

PROJECT STRUCTURE:



```
EXPLORER
COMPLAINT-REGISTERAT...
  backend
    > node_modules
    config.js
    index.js
    package-lock.json
    package.json
    Schema.js
```

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



```
COMPLAINT-REGISTERAT...
  frontend
    > node_modules
    public
      index.html
    src
      components
        admin
          AccordionAdmin.jsx
          AdminHome.jsx
          AgentInfo.jsx
          UserInfo.jsx
        agent
          AgentHome.jsx
        common
          ChatWindow.jsx
          FooterC.jsx
          Home.jsx
          Login.jsx
          SignUp.jsx
        user
          Complaint.jsx
          HomePage.jsx
          Status.jsx
      Images
        Image1.png
      App.css
      App.js
      index.js
    .gitignore
    package-lock.json
    package.json
    README.md
```

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

APPLICATION FLOW:

Online Complaint Registration and Management System

1. **Customer / Ordinary User**

- **Role:**
  Submit and track complaints, communicate with agents, and manage personal profile details.
- **Flow:**

1. **Registration & Login**
   - Create an account by entering details such as name, email, and password.
   - Login securely using registered credentials.
2. **Complaint Submission**
   - Access the complaint form and input details including issue description, contact information, and attachments.
   - Submit the complaint for processing.
3. **Status Tracking**
   - View all submitted complaints via the dashboard.
   - Receive real-time notifications as complaint status changes (e.g., Assigned, In Progress, Resolved).
4. **Interaction with Agents**
   - Communicate with the assigned agent through the in-app messaging/chat feature.
   - Clarify issues and provide additional context if needed.
5. **Profile Management**
   - Update and manage personal information such as address, phone number, and account password

2. Agent:

- **Role:**
  Address and resolve complaints assigned by the admin, maintain customer communication, and provide complaint status updates.

- **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. **System Monitoring**
   View and moderate all complaints submitted across the platform.
   Ensure that no complaint is left unresolved.
2. **Complaint Assignment**
   Allocate complaints to agents based on availability, workload, and expertise.
   Reassign complaints when needed for efficient resolution.
3. **User & Agent Management**
   Add, remove, or update user and agent profiles.
   Monitor login activity and profile integrity.
4. **Policy Enforcement**
   Ensure compliance with platform guidelines, terms of service, and data privacy regulations.
5. **Platform Improvement**
   Collect feedback to enhance system functionality and security.
   Address system-level issues or suggestions from users and agents.

Project Flow:

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
Use the code in: https://github.com/awdhesh-student/complaint-registery.git
or follow the videos below for better understanding.

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.

   - frontend folders.
   - Backend folders

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

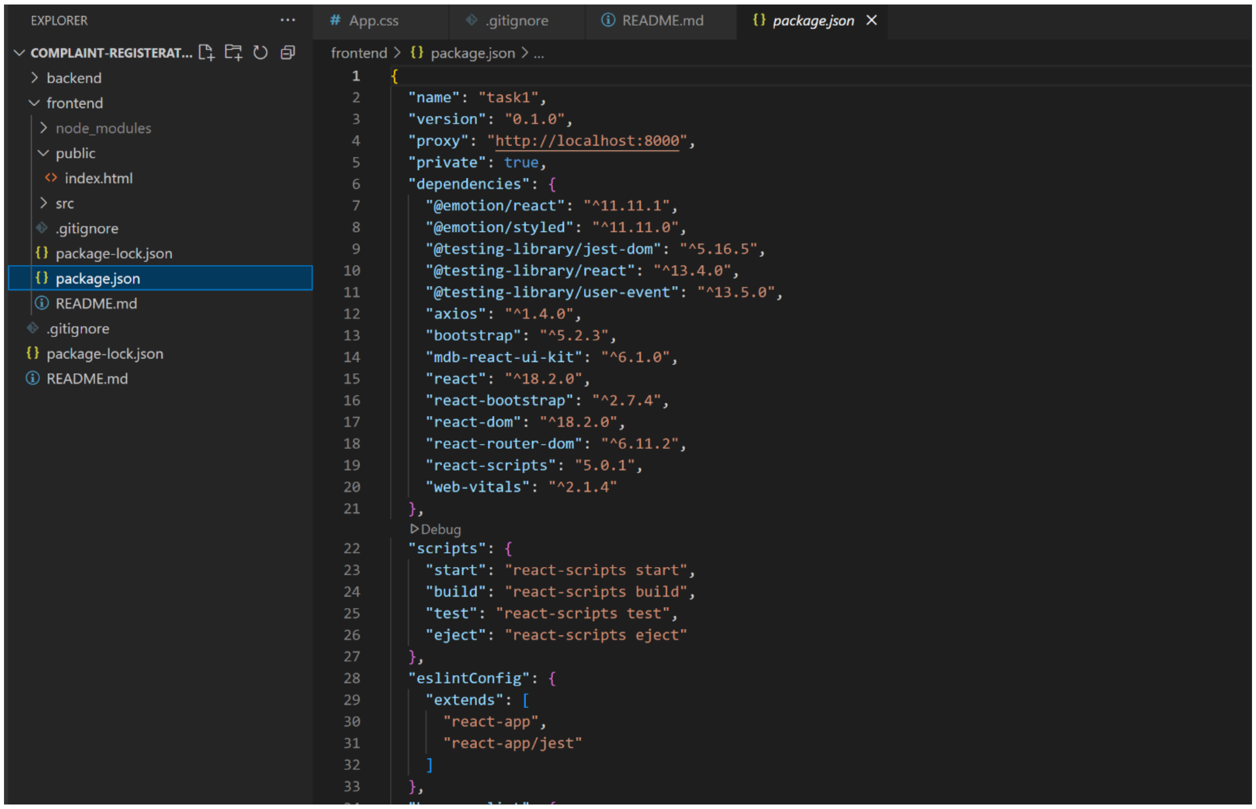
   For the backend to function well, we use the libraries mentioned in the prerequisites. Those libraries includes

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

   Also, for the frontend we use the libraries such as

   - React Js.
   - Material UI
   - Bootstrap
   - Axios

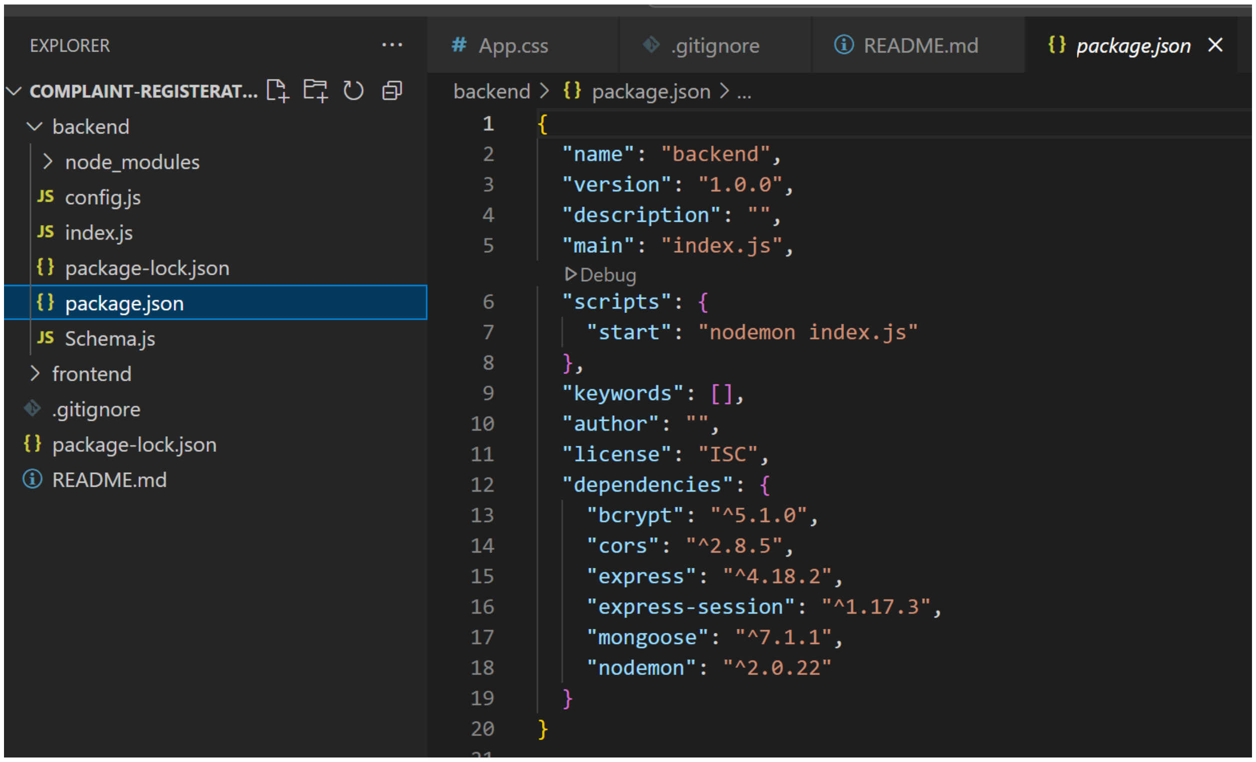
After the installation of all the libraries, the package.json files for the frontend looks like the one mentioned below.


EXPLORER
COMPLAINT-REGISTERAT...
backend
frontend
node_modules
public
index.html
src
.gitignore
package-lock.json
package.json
README.md
.gitignore
package-lock.json
README.md

frontend > {} package.json > ...

```
{
  "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"
  },
  ▷Debug
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
```

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


EXPLORER
COMPLAINT-REGISTERAT...
backend
node_modules
config.js
index.js
package-lock.json
package.json
Schema.js
frontend
.gitignore
package-lock.json
README.md

backend > {} package.json > ...

```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷Debug
  "scripts": {
    "start": "nodemon index.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.1.0",
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "express-session": "^1.17.3",
    "mongoose": "^7.1.1",
    "nodemon": "^2.0.22"
  }
}
```

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.

Reference video for backend code:

https://drive.google.com/file/d/113g0LQe5AsYBzgBmZNESdmFleq32PaNG/view?usp=sharing

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.

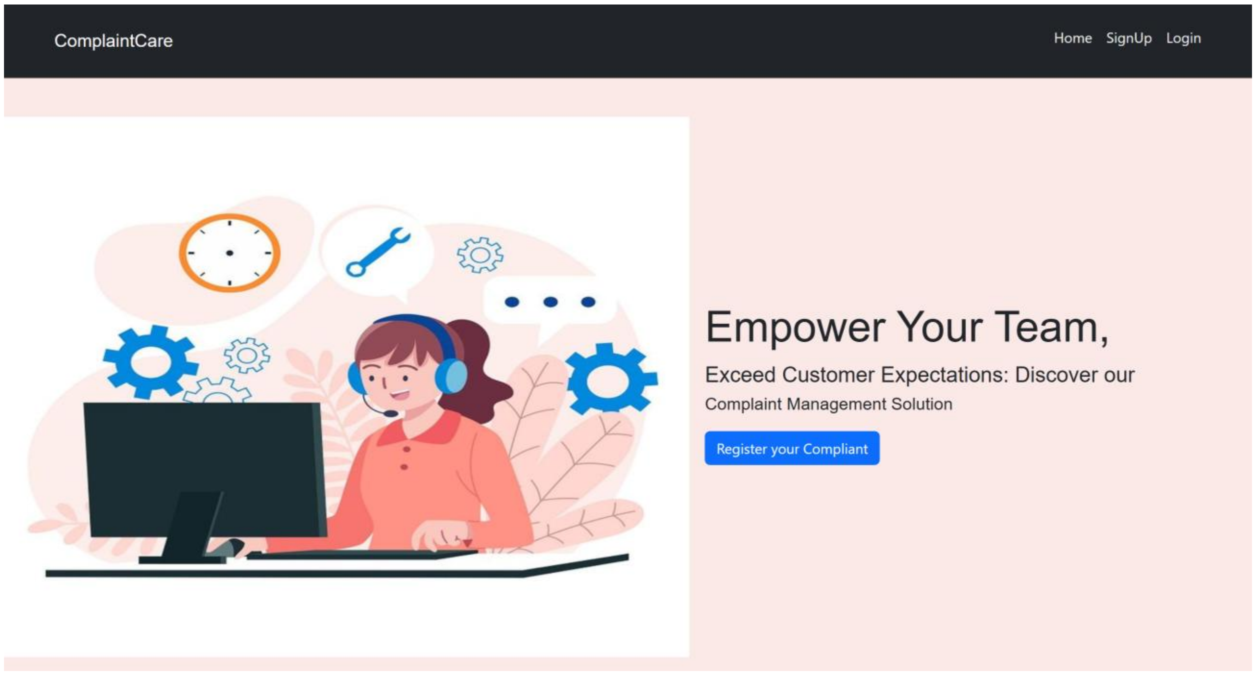
Reference video for frontend code:
https://drive.google.com/file/d/1Tj0TPrJK9c7R5C8RgGwao2gXDpqKbkNX/view?usp=sharing

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.

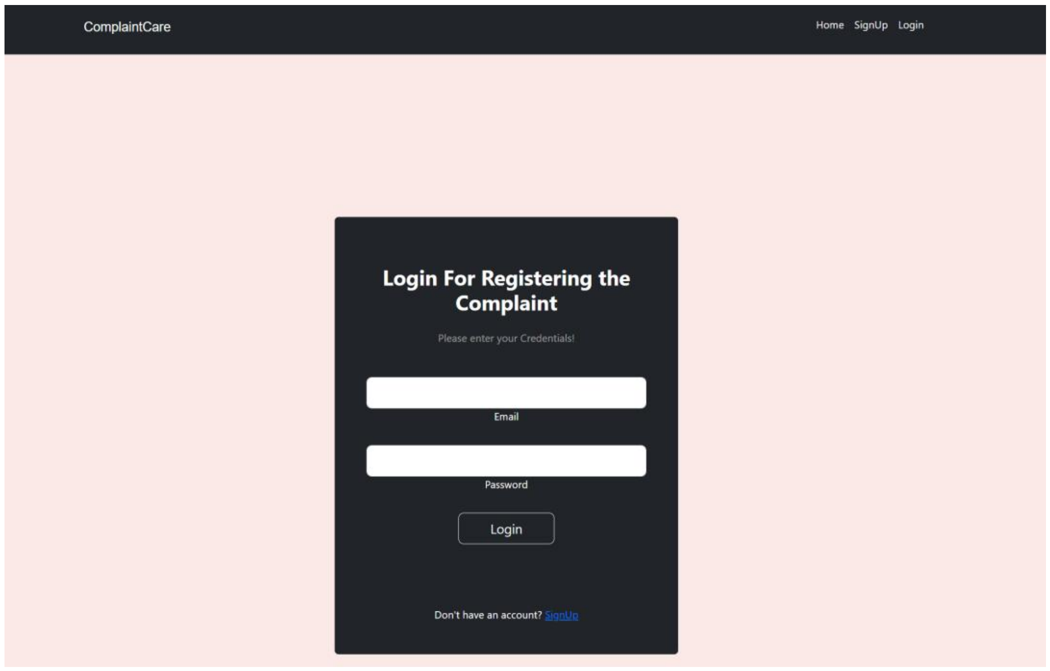
- Landing Page



Empower Your Team,

Exceed Customer Expectations: Discover our Complaint Management Solution

- Login Page

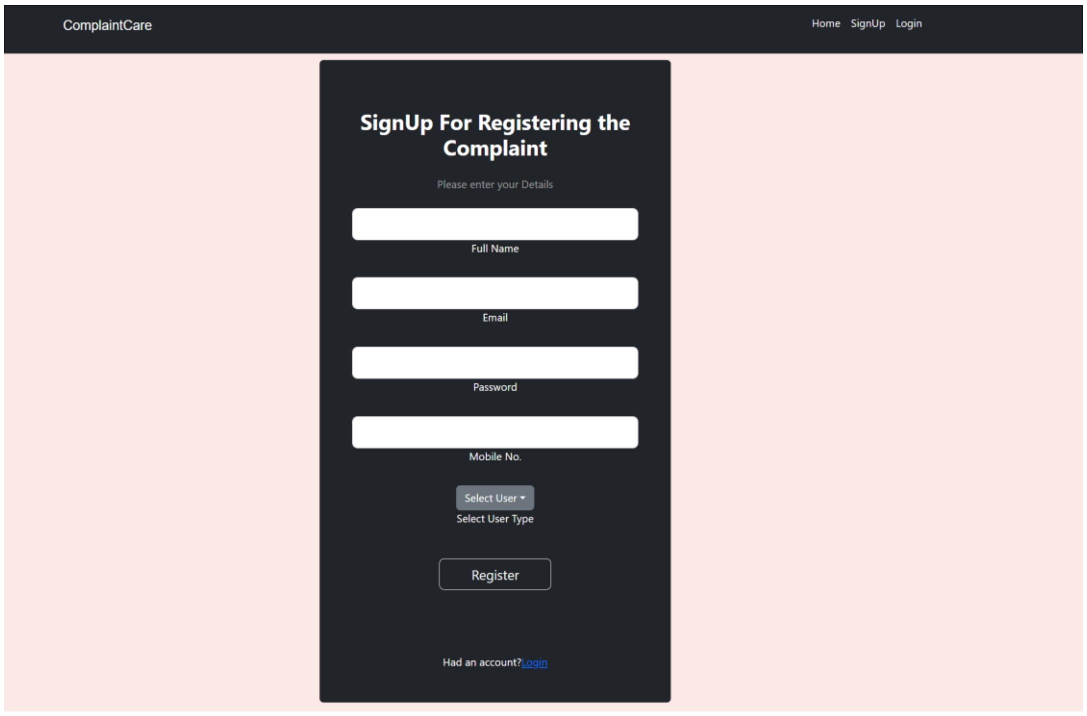


Login For Registering the Complaint

Please enter your Credentials!

Don't have an account? SignUp

- Registration Page

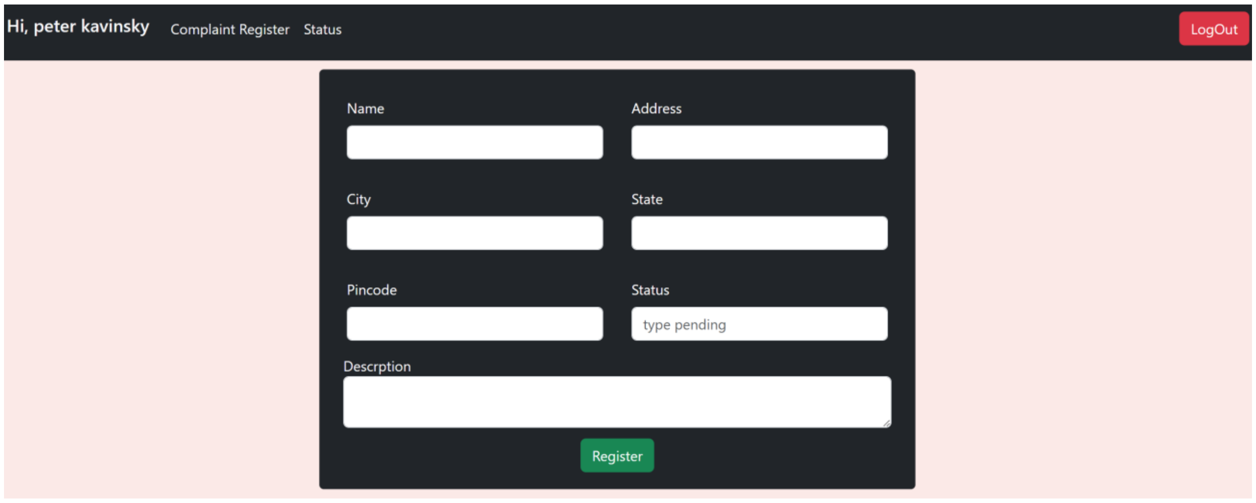


SignUp For Registering the Complaint

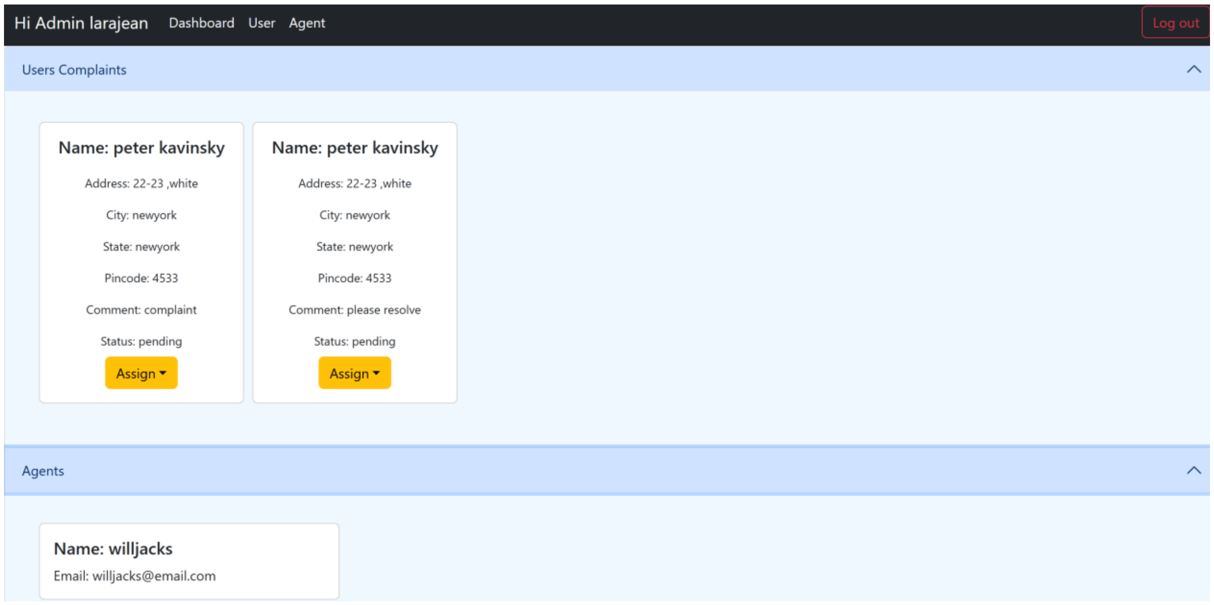
Please enter your Details

Had an account?Login

- Common Dashboard For Complaint



- Admin Dashboard

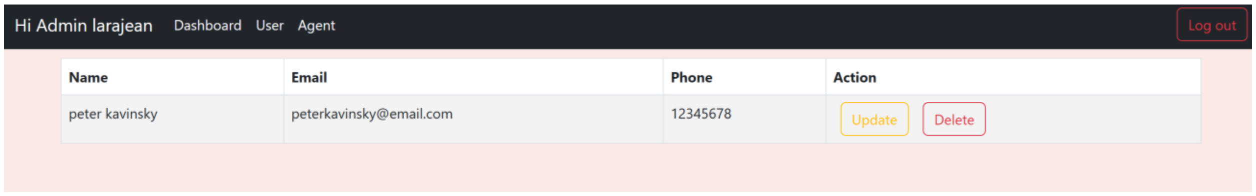


Name: peter kavinsky
Address: 22-23 ,white
City: newyork
State: newyork
Pincode: 4533
Comment: complaint
Status: pending

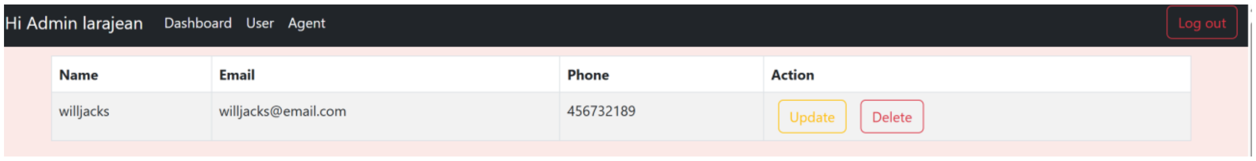
Name: peter kavinsky
Address: 22-23 ,white
City: newyork
State: newyork
Pincode: 4533
Comment: please resolve
Status: pending

Name: willjacks
Email: willjacks@email.com

- Admin User and Agent Dashboards

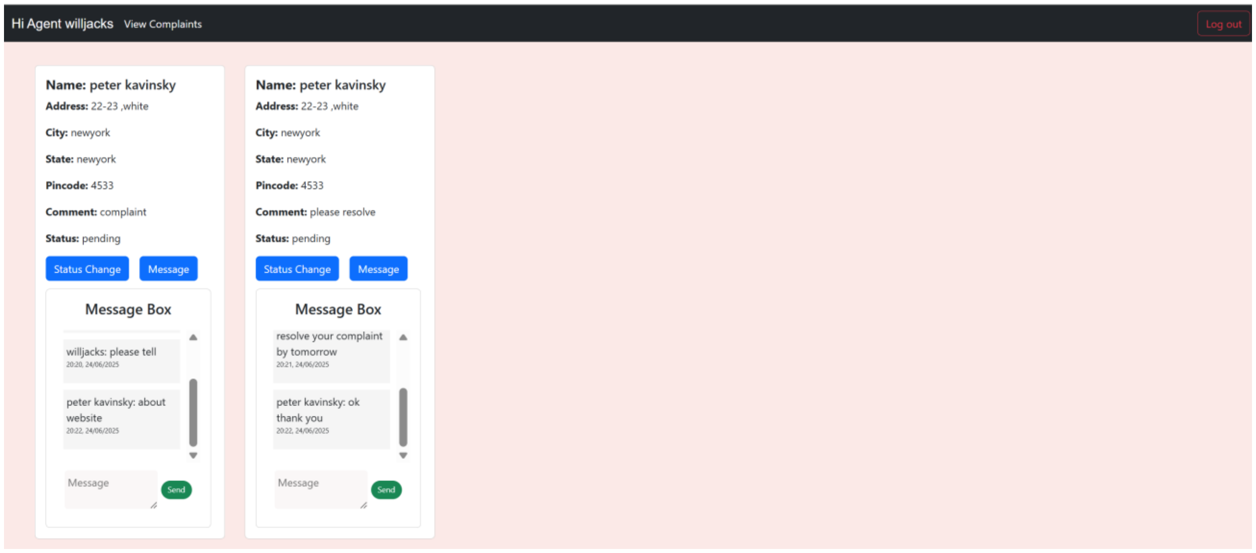


Name	Email	Phone	Action
peter kavinsky	peterkavinsky@email.com	12345678	Update Delete



Name	Email	Phone	Action
willjacks	willjacks@email.com	456732189	Update Delete

- Agent Dashboard to view Complaints



Name: peter kavinsky
Address: 22-23 ,white
City: newyork
State: newyork
Pincode: 4533
Comment: complaint
Status: pending

willjacks: please tell
20:20, 24/06/2025

peter kavinsky: about website
20:22, 24/06/2025

Name: peter kavinsky
Address: 22-23 ,white
City: newyork
State: newyork
Pincode: 4533
Comment: please resolve
Status: pending

resolve your complaint by tomorrow
20:21, 24/06/2025

peter kavinsky: ok thank you
20:22, 24/06/2025

For any further doubts or help, please consider the code from the google drive,

https://drive.google.com/drive/folders/1nGZmQVYUZLoQ_fRjaZFau-3wz5_uZKqD?usp=sharing

The demo of the app is available at:

https://drive.google.com/file/d/1bSHBUpfhuTbNxFnqB7V_604QZ-Vd2SeT/view?usp=sharing

For code the GitHub link is:

https://github.com/Prajitha-Katikolu/complaint-registration