

NAAN MUDHALVAN PROJECT

MERN stack powered by MongoDB



Smart
Internz



PROJECT TITLE

Online Complaint Registration And Management System Using Mern Stack

Submitted by the team members of Final Year IT

Mohammed Azeem R 110121205033

Mohamed Fahim I 110121205022

Mohamed Asif M 110121205021

Mohamed Fawzan S 110121205024

DEPARTMENT OF INFORMATION TECHNOLOGY AALIM MUHAMMED
SALEGH COLLEGE OF ENGINEERING, MUTHAPUDUPET, AVADI,
CHENNAI - 600 055, TAMILNADU. (AFFILIATED TO ANNA
UNIVERSITY)

Introduction:

An online complaint registration system is a digital tool that simplifies and improves the process of filing and resolving complaints. An online complaint registration and management system is a web-based application designed to simplify the process of submitting, tracking, and resolving complaints. By leveraging modern technologies, this system aims to provide users with an efficient and transparent platform for lodging complaints, while ensuring that administrators can manage and resolve issues in a timely manner.

The system allows users to create complaints through an intuitive web interface, attach relevant documents or details, and monitor the status of their complaints in real time. For administrators, the system offers an easy-to-use dashboard to view and manage incoming complaints, assign tasks, and provide updates.



Objectives:

The primary objective of an online complaint registration and management system is to streamline and simplify the process of filing and addressing complaints. By offering a digital platform, users can register complaints from any location, removing the need for in-person or paper-based submissions.

This system improves efficiency by enabling users to track the status of their complaints in real time, enhancing transparency and providing timely updates. Complaints can be automatically assigned to the appropriate departments, reducing delays and ensuring faster response times.



The system also seeks to expedite complaint resolution through automation, intelligently assigning complaints to relevant departments or staff based on pre-established rules. This feature reduces processing time and increases the likelihood of timely resolution. Moreover, the platform facilitates effective communication between users and administrators by enabling direct feedback, notifications, and status updates, ensuring all parties remain well-informed.

Problem Statement:

The problem with traditional complaint management systems is that they are often manual, inefficient, and lack transparency, leading to delays, user frustration, and unresolved complaints. In many organizations and institutions, complaints are typically filed through paper forms, phone calls, or email, which makes it difficult to track the status and progress of individual cases.

Furthermore, users have limited access to updates on their complaint status, reducing trust in the system and satisfaction with the service. For administrators, the manual handling of complaints is labor-intensive and prone

to errors, making it challenging to analyze data for performance improvements. Without a structured digital system, it is difficult to maintain records, assign complaints efficiently, and gather insights for process optimization.

Solution:

The aim of an online complaint registration and management system is to address these issues by providing a digital platform that allows users to easily submit complaints, track their progress, and receive timely updates. For administrators, the system centralizes complaint handling, streamlines workflow, and improves efficiency through automated assignment and data management, ultimately enhancing transparency, accountability, and user satisfaction.

Requirements:

The requirements for an **Online Complaint Registration and Management System** can be broadly divided into **functional** and **non-functional requirements**, as well as **system** and **user requirements**.

Functional Requirements

System Requirements

Non-Functional Requirements

User Requirements

Functional Requirements:

➤ **User Registration and Authentication:**

Users must be able to create an account with a username/email and password..

➤ **Complaint Registration:**

Users can submit complaints with relevant details and attachments

➤ **Complaint Tracking:**

Users can track the status of their submitted complaints.

Notifications should be sent to users when their complaint status is updated.

➤ **Admin Dashboard:**

Admin can manage complaints, assign them, and monitor progress.

➤ **Agent Dashboard:**

Different access levels (User, Admin, Agent) with restrictions based on roles.

Admin has full access to all features, whereas agents only access complaints assigned to them.

➤ **Role-Based Access:**

Different access levels (User, Admin, Agent) with restrictions based on roles.

Admin has full access to all features, whereas agents only access complaints assigned to them.

Non-Functional Requirements:

➤ **Performance:**

The system must handle a large volume of complaints without significant slowdowns.

The response time for complaint registration and status updates should be minimal (within a few seconds).

➤ **Security:**

- Sensitive data, such as user information and complaint details, must be securely stored (using encryption).

Use of secure login mechanisms (e.g., JWT for session management, password hashing).

Role-based access control (RBAC) to ensure only authorized personnel can access certain data.

➤ **Usability:**

The system must have an intuitive and user-friendly interface.

The system should be accessible across multiple devices (desktop, tablet, mobile) and responsive to different screen sizes.

➤ **Reliability:**

The system must be highly available and reliable, with minimal downtime.

Data backup mechanisms should be in place to prevent data loss.

➤ **Compliance:**

The system should comply with applicable data protection regulations (e.g., GDPR) when handling user information.

System Requirements:

➤ **Software:**

Frontend: React (for building the user interface).

Backend: Node.js with Express.js (for building RESTful APIs).

Database: Mongo DB (for storing complaints and user data).

Hosting: Cloud hosting platform (e.g., AWS, Heroku, or local server).

➤ **Hardware:**

Server with adequate CPU, RAM, and storage to handle the system's load.

A reliable internet connection for cloud or on-premise hosting.

➤ **Compatibility:**

The system should work across major browsers like Chrome, Firefox, Safari, and Edge.

User Requirements:

➤ **End Users:**

Ability to register, log in, and submit complaints.

Ability to track and check the status of complaints.

Receive notifications about complaint updates.

➤ **Admin**

- Ability to manage user roles, assign complaints, and monitor complaint resolutions.

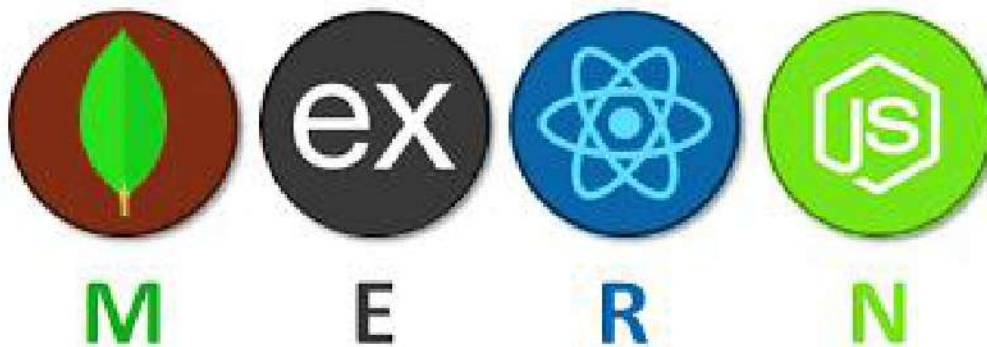
Ability to generate and view reports on complaint statistics and trends.

➤ **Agents:**

Ability to view and manage complaints assigned to them.

Ability to update complaint status and add comments or solutions.

Technical Specifications:



MERN is a popular technology stack that consists of four powerful technologies used to build dynamic, full-stack web applications. It stands for **M**ongoDB, **E**xpress.js, **R**ead, and **N**ode.js, and is widely used for developing scalable, efficient, and high-performance web applications.

Here's how each component of the MERN stack contributes to the development of an **Online Complaint Registration and Management System**:

MongoDB (Database):

MongoDB is a NoSQL database used to store and manage the system's data in a flexible, JSON-like format. In the case of the complaint management system, MongoDB stores user data, complaint details, status updates, and any

other relevant information.

MongoDB's schema-less structure allows the system to easily handle varied types of data (such as different complaint categories or user preferences) and scale as the number of complaints grows.

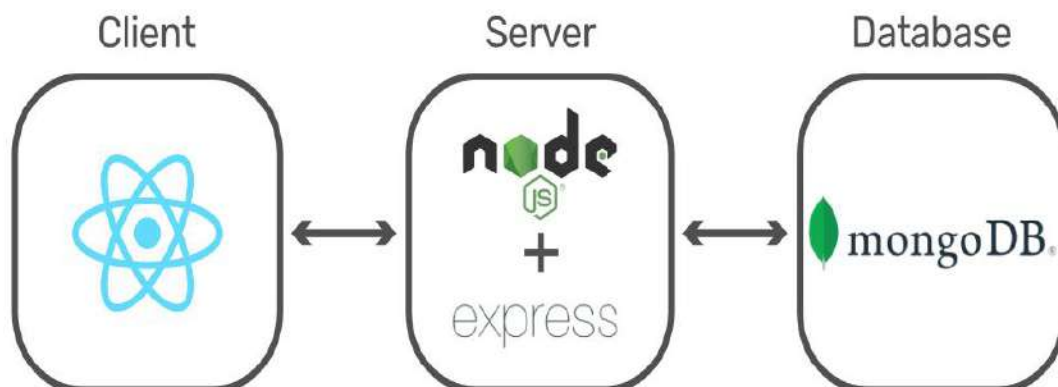
Express.js (Backend Framework):

Express.js is a web application framework for Node.js that simplifies the creation of APIs and backend functionality. It is used to handle HTTP requests, define routes, and process data between the frontend and backend.

In the complaint system, Express.js enables features such as submitting complaints, updating complaint statuses, and managing user authentication by connecting the frontend React components to the MongoDB database.

Its middleware capabilities simplify complex tasks like error handling, request processing, and user authentication, making it easier to build a secure, well-structured API for managing complaints, user roles, and statuses.

The MERN stack



React (Frontend Library):

React is a JavaScript library used to build dynamic and interactive user interfaces. It enables the creation of single-page applications (SPA) by updating only the components that change without reloading the entire page.

For the complaint system, React provides a responsive and user-friendly interface where users can register complaints, track their progress, and administrators/agents can view and resolve complaints. React's component-based architecture makes it easier to manage the user interface and improve performance.

React allows developers to reuse UI components, such as login forms and dashboards, which enhances maintainability and reduces development time.

Node.js (Runtime Environment):

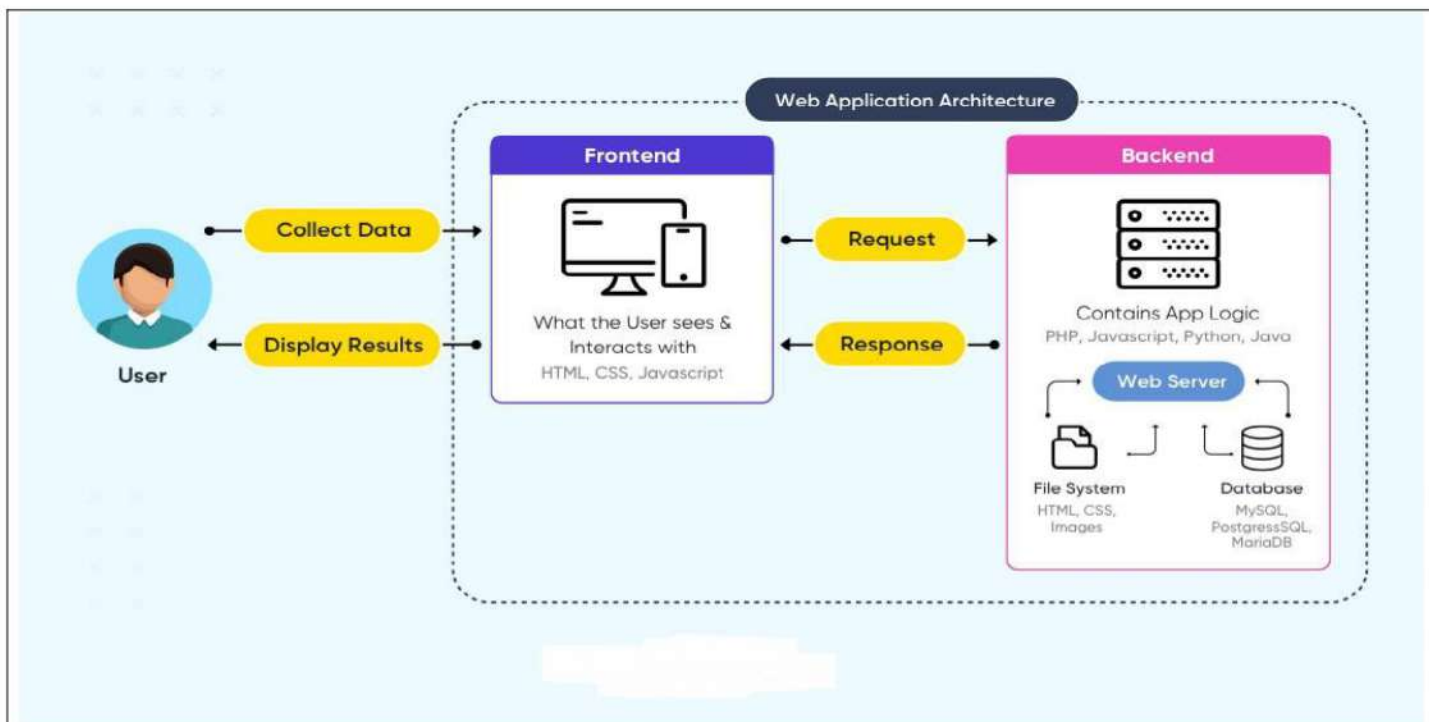
Node.js is a JavaScript runtime that runs on the server-side. It allows developers to write both the client-side and server-side code in JavaScript, making the development process more efficient.

In the complaint system, Node.js serves as the backend runtime environment, handling business logic, processing incoming requests, and communicating with the MongoDB database through Express.js.

Known for its event-driven architecture, Node.js can handle multiple requests simultaneously, an essential feature for a system where complaint data and status updates are frequently accessed by users, administrators, and agents.

Technical Architecture:

The **System Architecture** of an **Online Complaint Registration and Management System** built with the MERN stack typically includes the following layers and components:



Client-Side (Frontend):

1.React Application: This is the main interface for users, admins, and agents, where they interact with the system. The frontend uses React to create a dynamic, responsive single-page application (SPA).

2.Component-Based UI: The UI is divided into components like login, registration, complaint form, user dashboard, admin dashboard, and agent dashboard.

3.API Calls: The frontend communicates with the backend server via HTTP requests (using Axios or Fetch API) to perform actions such as submitting complaints, tracking statuses, and fetching user details.

Server-Side (Backend):

1.Node.js: Node.js runs the backend server, executing the business logic and processing incoming requests.

2.Express.js Framework: Express is used to create RESTful APIs, defining endpoints for various functionalities like complaint registration, status updates, user authentication, and data retrieval.

3.Middleware: Express middleware is used for request parsing, logging, authentication (using JSON Web Tokens), and error handling, which helps

maintain security and efficient processing.

Database Layer:

#MongoDB Database: MongoDB serves as the primary database, storing data in a flexible, document-based format. Collections within MongoDB include:

1.Users: Stores user profiles, roles (e.g., user, admin, agent), and authentication credentials.

2.Complaints: Contains complaint data, including type, description, status, priority, and timestamps.

3.Logs/Notifications: (Optional) Stores logs or notifications for tracking changes or updates to complaint statuses.

Authentication and Security:

JWT (JSON Web Tokens): JWTs are used for secure user authentication. After login, a JWT is issued, allowing the user to access their dashboard or other functionalities securely. The JWT is included in HTTP headers for protected API requests.

Cloud Hosting and Deployment

Hosting Services: The application can be hosted on cloud platforms like **AWS, Heroku, or DigitalOcean**. The frontend React app, backend Node/Express server, and MongoDB database are hosted and connected through the cloud.

Load Balancing and Scalability: The hosting platform provides load balancing and scalability options to manage high traffic and increased demand, ensuring consistent performance.

Application Workflow:

Customer/Ordinary User Flow:

The **Customer/Ordinary User** in the **Online Complaint Registration and Management System** has a key role in creating, managing, and tracking complaints. Below is the detailed flow of the user's interactions within system,

1. Registration and Login:

Account Creation: The user starts by creating an account by providing basic details such as **email address**, **password**, and sometimes additional information like **contact number** or **address**.

Login: After successful registration, the user can log in using their credentials (email and password). If the user forgets their password, they can use a **password recovery** feature to reset it.

Authentication: The system verifies the credentials and issues a **JWT token** for secure session management, ensuring the user has access to their dashboard and related features.

2. Complaint Submission:

Complaint Form: Once logged in, the user accesses the **complaint submission page**. They fill out a form with all the required details, including:

Complaint type (e.g., service issue, technical problem, etc.)

Description of the issue in detail.

Contact information for follow-up (if required).

Attachments such as images or documents to support the complaint.

Submission: After completing the form, the user submits the complaint. Upon successful submission, the system assigns a unique **complaint ID** and stores the complaint in the database.

3. Status Tracking:

Complaint Dashboard: The user can view all submitted complaints on their **dashboard**, where each complaint is listed with essential details such as **complaint ID**, **status**, **date submitted**, and **priority**.

Status Updates: The system provides real-time updates about the complaint's status, such as "In Progress", "Resolved", "Under Review", or "Closed".

Notification Alerts: Users receive notifications (via email, SMS, or in-app) whenever there is an update on their complaint, ensuring they are informed at every stage.

4. Interaction with Agents:

Assigned Agent: Each complaint is assigned to an **agent** or **admin** responsible for resolution. The user can see the agent's details in their complaint dashboard.

Messaging Feature: The system includes a **messaging** or **chat feature** where users can directly communicate with the assigned agent, discuss further details of the complaint, ask questions, or provide additional information to help resolve the issue.

Follow-up: Users can keep the conversation ongoing, asking for updates or clarifications about the resolution process.

5. Profile Management:

Profile Information: The user can update and manage their personal details, including contact information, address, and other preferences, from their **profile page**.

Change Password: The user can also change their password if necessary, ensuring account security.

Logout: After completing their tasks, the user can securely **log out** from the application to maintain privacy and session security.

Agent Flow:

The **Agent** in the **Online Complaint Registration and Management System** plays a crucial role in managing complaints, interacting with customers, and updating complaint statuses. Below is the detailed flow of the agent's interactions within the system:

1. Registration and Login:

Account Creation: The agent first creates an account by providing required details such as **email address** and **password**. This allows the agent to access the system securely.

Login: Once registered, the agent logs in using the credentials they provided (email and password). Authentication is handled securely, and the system generates a **JWT token** for session management to ensure that only authenticated agents can access the system's dashboard.

2. Complaint Management:

Assigned Complaints: Once logged in, the agent accesses their **dashboard**, which lists all the complaints assigned to them by the **admin**. These complaints include relevant details like **complaint ID**, **user information**, **issue description**, **priority**, and **status**.

Complaint View: The agent can click on any complaint to view its detailed information and history, including past interactions with the customer and current status.

Complaint Categorization: The agent can categorize complaints based on urgency or issue type, allowing them to prioritize and address the most pressing cases first.

3. Customer Interaction:

Chat/Message Communication: The agent communicates directly with customers through the **inbuilt messaging/chat feature** in the system. They can ask for more information, clarify issues, or request additional documents if needed.

Responding to Feedback: The agent can also respond to any inquiries or feedback provided by the customers regarding the resolution process, keeping them updated on the complaint's progress.

Resolution: As the agent works to resolve the complaint, they can ask for any further information, close communication loops with the customer, and ensure that the complaint is fully addressed.

4. Status Update:

Progress Updates: As the agent works on the complaint, they update its status (e.g., "In Progress", "Under Review", "Resolved") to reflect the current stage of the resolution.

Final Resolution: Once the complaint is resolved, the agent can mark the complaint as "Resolved", triggering a final update to the customer. This update is often accompanied by a message explaining the resolution steps taken and any further actions required.

Notifications: After updating the complaint status, the agent can send **notifications** (via in-app alerts or email) to customers to inform them of the progress or resolution.

5. Customer Feedback and Issue Resolution:

Follow-up: Agents can follow up with customers to ensure satisfaction with the resolution. If any issues persist, the agent can reopen the complaint or escalate it to a higher authority (like an admin).

Resolving Disputes: If the customer is not satisfied with the resolution, the agent can engage in further discussions, resolve the issue, or escalate the complaint to the admin for further investigation.

Documentation: Throughout the process, the agent may document the steps taken, comments, or notes related to the complaint to provide a detailed record for future reference.

Admin Flow:

The **Admin** in the **Online Complaint Registration and Management System** has a critical role in overseeing and managing the entire complaint lifecycle, from submission to resolution. Below is the detailed flow of the admin's interactions within the system:

1. Management and Monitoring:

Complaint Monitoring: The admin has an overarching view of all complaints submitted by users. They can monitor the status and details of each

complaint in real-time, ensuring that no complaints go unresolved or are overlooked.

Complaint Moderation: The admin can moderate complaints to ensure they meet the system's standards and policies before assigning them to agents. This includes checking for completeness and appropriateness of the information provided by users.

Analytics and Reporting: The admin can access analytics on complaint volume, resolution times, user satisfaction, and agent performance. These insights help in monitoring the overall system health and in making data-driven decisions.

2. Complaint Assignment:

Assigning Complaints: Admin are responsible for assigning complaints to the most appropriate agent based on expertise, availability, and workload. The system allows the admin to easily allocate complaints and track the progress of their resolution.

Priority Management: The admin can prioritize complaints, ensuring that critical or urgent issues are addressed first. This ensures that the platform operates efficiently and meets users' needs promptly.

Monitoring Timeliness: Admin ensure that complaints are handled in a timely manner by setting deadlines or alerts if an agent does not update the status within a set timeframe.

3. User and Agent Management:

User Management: The admin is responsible for managing the accounts of both **users** and **agents**. They can register new users, activate accounts, or deactivate profiles when needed. They may also edit or update user information as necessary.

Agent Management: Admin oversee the registration and management of **agents**. They ensure agents have appropriate permissions and are assigned to the correct complaint types. Admin can monitor agents' performance and handle any disciplinary actions if needed.

Role-Based Access Control (RBAC): Admin enforce role-based access, ensuring that users, agents, and admin have the correct level of access to data and platform features.

4. Enforcement of Policies:

Platform Policies: Admin are responsible for enforcing the platform's **terms of service, privacy regulations**, and other relevant policies. This includes monitoring the use of the platform for abuse, fraud, or violations of terms.

Security and Compliance: Admin ensure that user data and complaint information are stored securely and that the platform complies with relevant regulations (e.g., data protection laws like GDPR).

Conflict Resolution: If there is a dispute between a user and an agent, or if there are issues with a complaint's resolution, the admin acts as a mediator, resolving conflicts and ensuring fairness.

5. Continuous Improvement:

Feature Enhancements: Admin monitor feedback from users and agents to identify areas for improvement in the platform. They may implement new features or modify existing ones to enhance the user experience, such as adding a more intuitive interface, improving reporting capabilities, or integrating new communication tools.

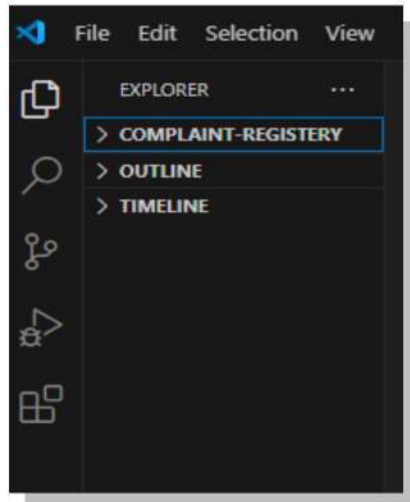
System Optimization: Admin are involved in continuous monitoring and optimization of the platform's performance. This includes overseeing system security, managing performance bottlenecks, and making sure the platform can scale as usage grows.

User and Agent Support: Admin address concerns or issues raised by users or agents. This could involve troubleshooting system bugs, handling user complaints about system functionality, and ensuring that the platform remains efficient and effective for all stakeholders.

Security Audits: Admin conduct regular security audits to ensure the platform's security measures are robust and that any vulnerabilities are addressed promptly.

Project configuration:

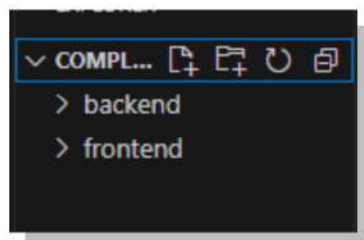
Now, for project configuration we used the visual studio code editor. In visual studio code editor, we created a folder called complaint-registry.



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



Install required tools and software:

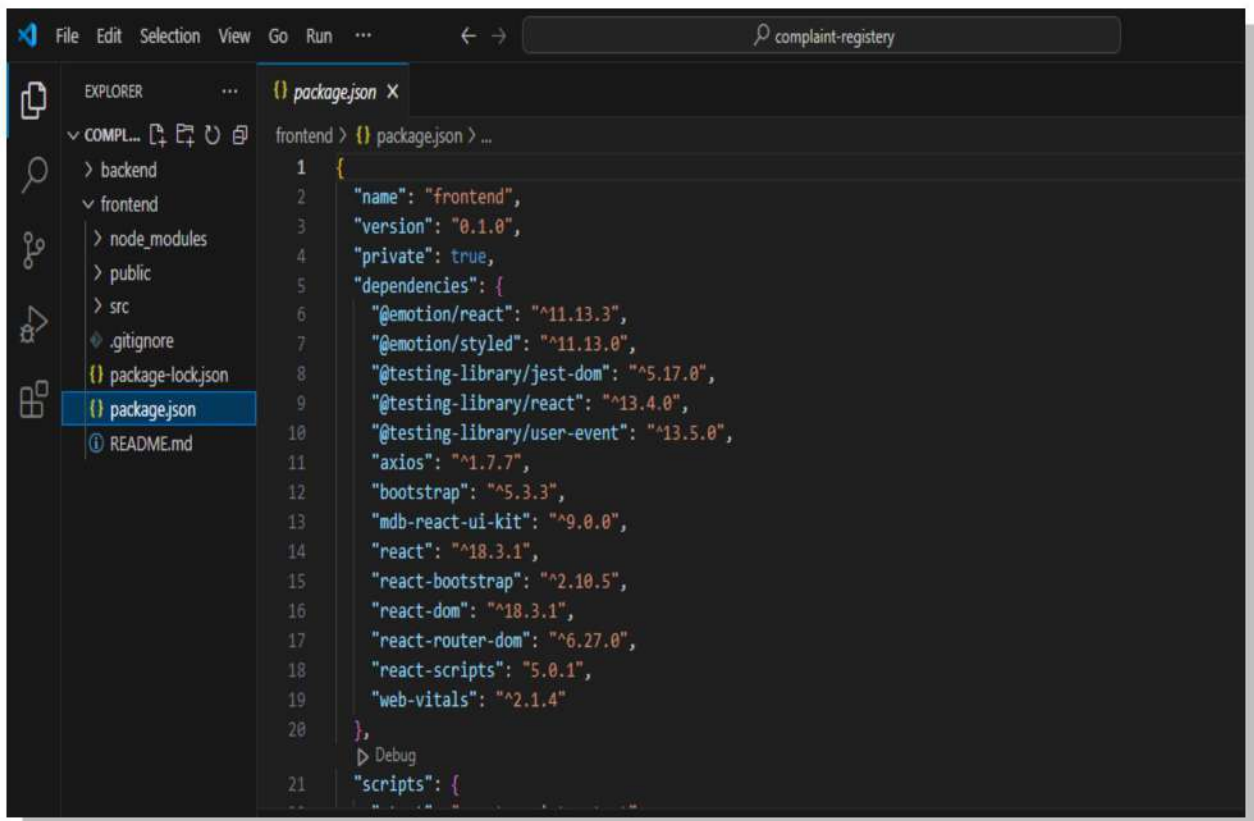
For the backend to function well, we use the libraries mentioned in the prerequisites. Those libraries include

- Node.js.
- MongoDB.
- Bcrypt
- Cors

- Express
- Express-session
- Mongoose
- Nodemon

Also, for the frontend we use the libraries such as

- React
- Bootstrap
- Axios
- React-dom
- React-bootstrap
- React-router-dom
- React scripts



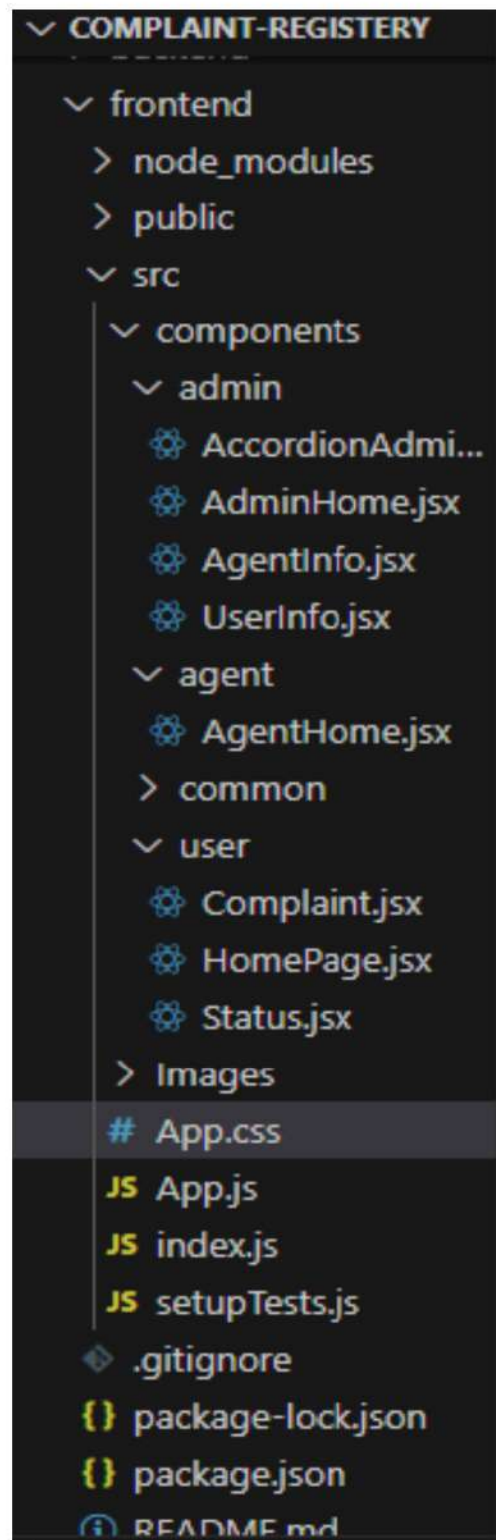
The screenshot shows a Visual Studio Code editor window with a project named 'complaint-registry'. The Explorer sidebar on the left shows a file tree with 'frontend' selected, containing 'node_modules', 'public', 'src', '.gitignore', 'package-lock.json', 'package.json', and 'README.md'. The main editor area displays the 'package.json' file for the 'frontend' directory. The file contains the following JSON structure:

```
1 {
2   "name": "frontend",
3   "version": "0.1.0",
4   "private": true,
5   "dependencies": {
6     "@emotion/react": "^11.13.3",
7     "@emotion/styled": "^11.13.0",
8     "@testing-library/jest-dom": "^5.17.0",
9     "@testing-library/react": "^13.4.0",
10    "@testing-library/user-event": "^13.5.0",
11    "axios": "^1.7.7",
12    "bootstrap": "^5.3.3",
13    "mdb-react-ui-kit": "^9.0.0",
14    "react": "^18.3.1",
15    "react-bootstrap": "^2.10.5",
16    "react-dom": "^18.3.1",
17    "react-router-dom": "^6.27.0",
18    "react-scripts": "5.0.1",
19    "web-vitals": "^2.1.4"
20  },
21  "scripts": {
```

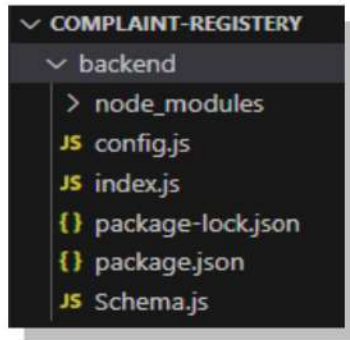
Project Implementation:

Add necessary backend and frontend folders and files

Frontend:

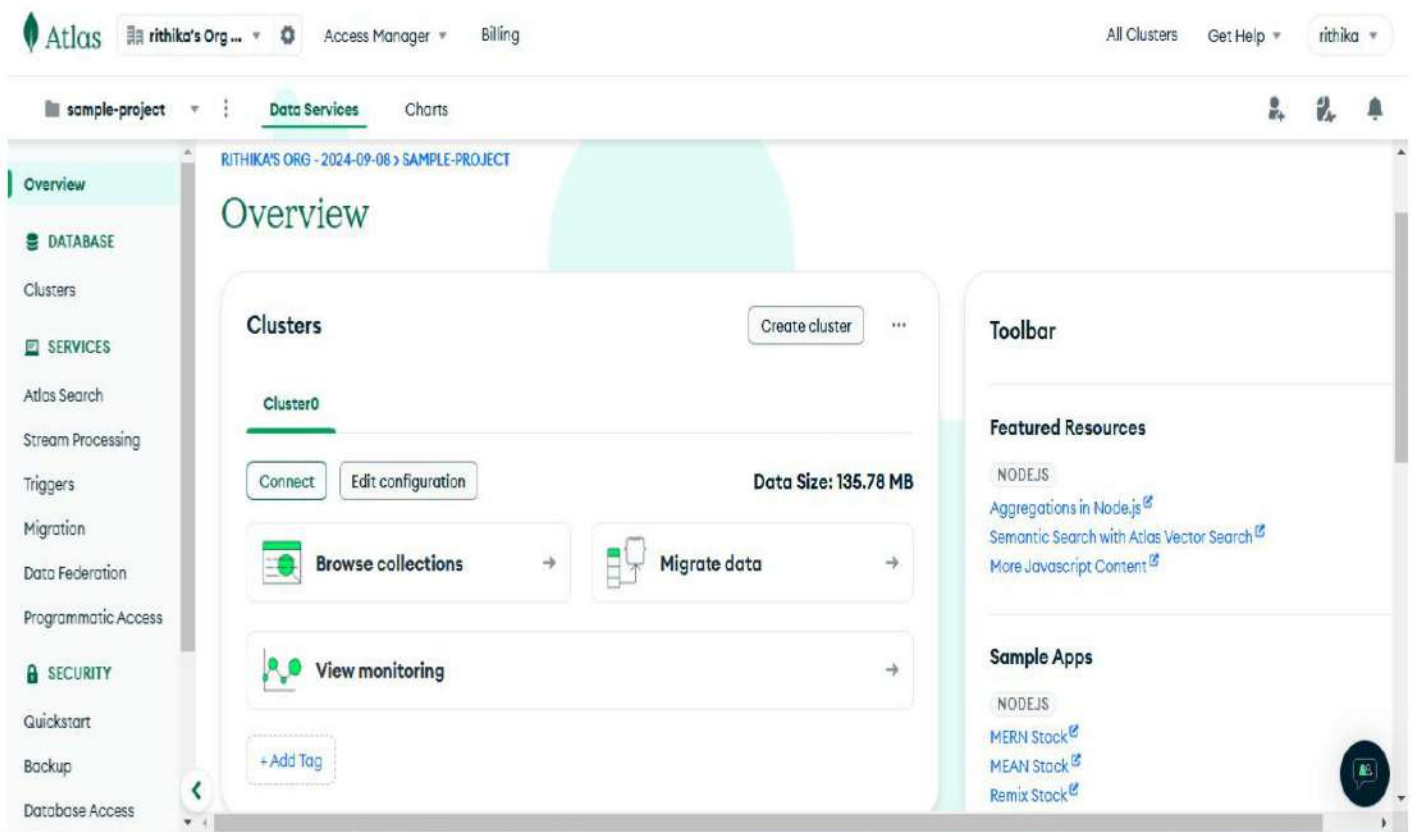


Backend

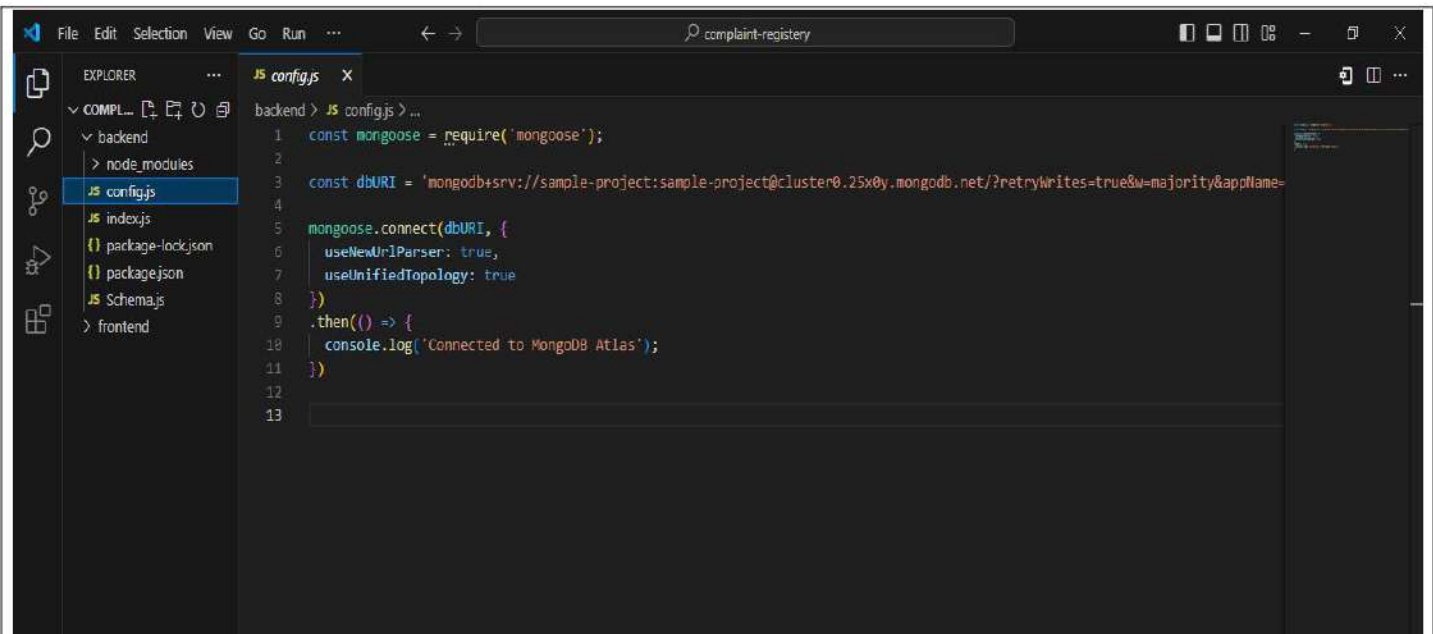


DATABASE:

For database creation , we used mongo db atlas, the mongo db pictures of online complaint registration and management system are shown below



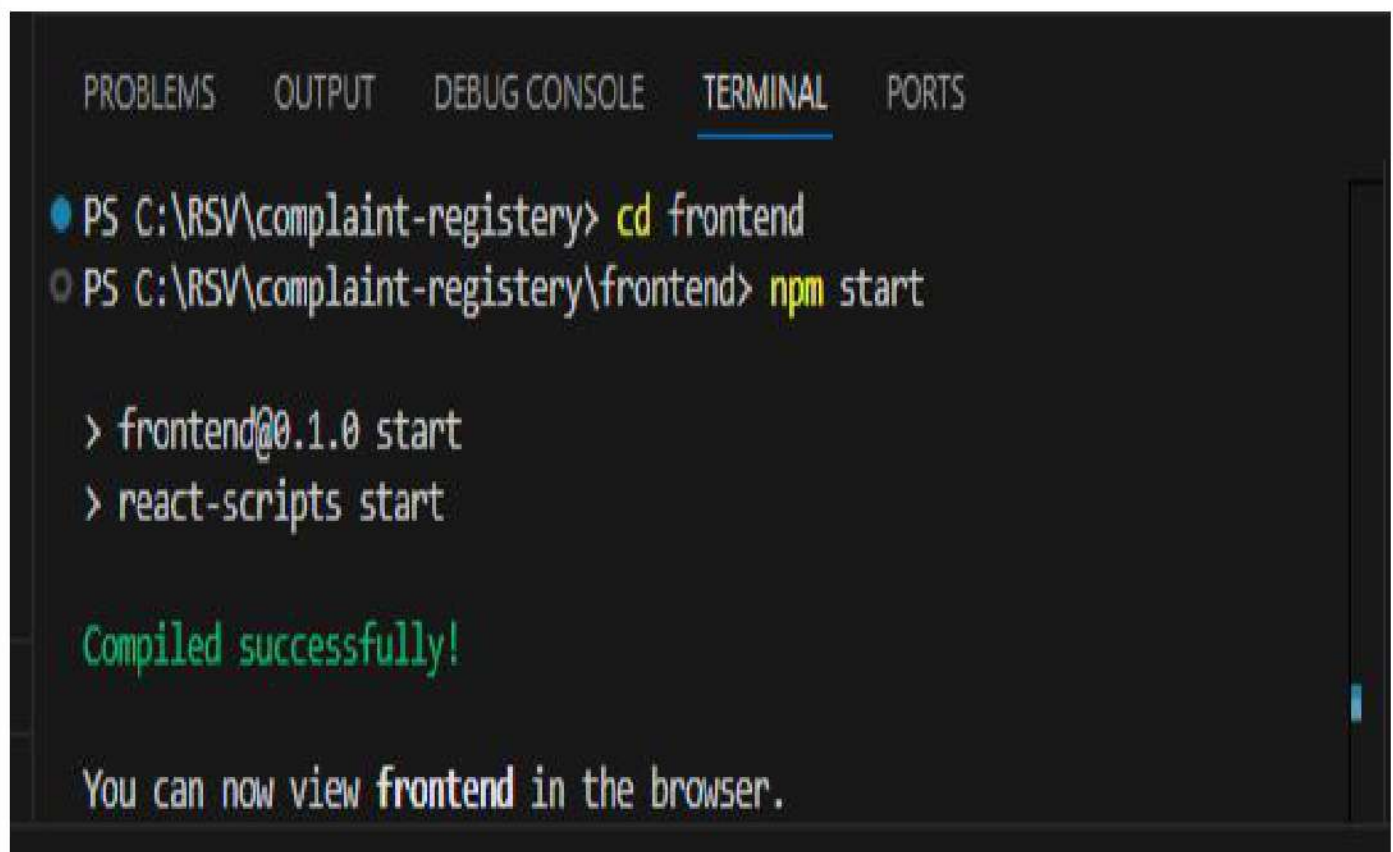
We used the mongo db string to connect the database to the backend folder in config.js file



Project Execution:

To execute the frontend folder and backend folder we need to give the “**npm start**” command in the terminal for each separately.

For frontend



For Backend

```
node - backend + v [ ] [ ] ... ^ X

● PS C:\RSV\complaint-registery> cd backend
○ PS C:\RSV\complaint-registery\backend> npm start

> backend@1.0.0 start
> nodemon index.js

[nodemon] 3.1.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*

Col 1 Spaces: 2 UTF-8 CRLF { } JavaScript @ Go Live ✓ Prettier [ ]
```

```
node - backend + v [ ] [ ] ... ^ X

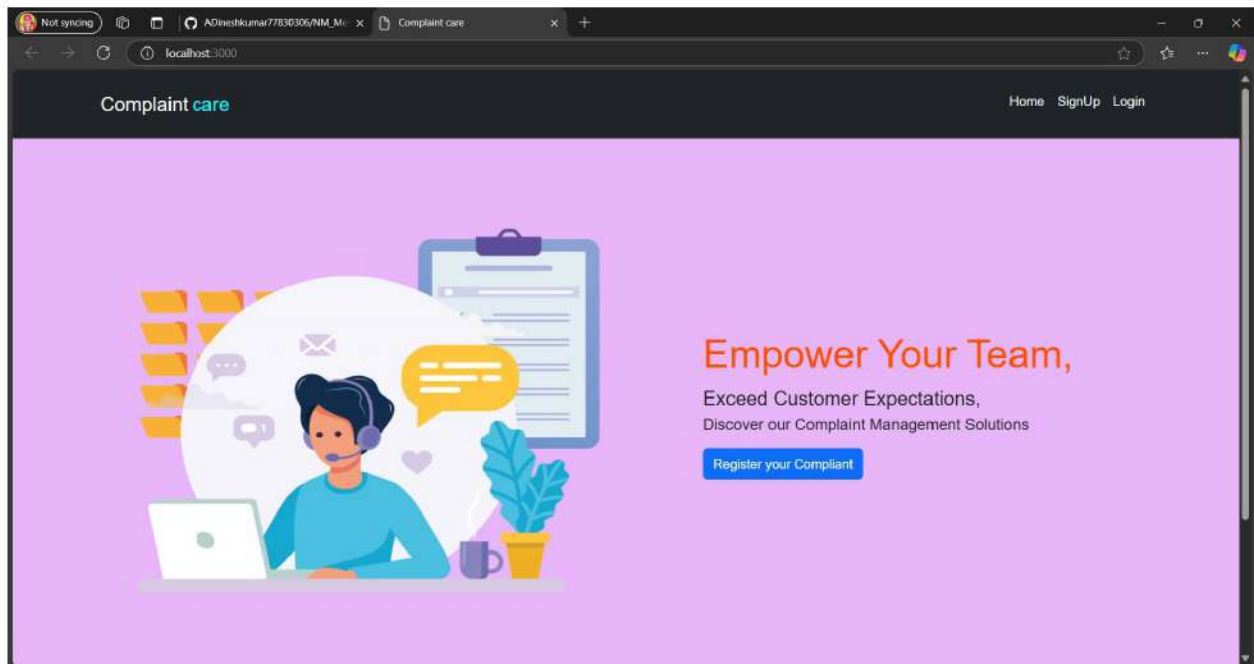
(Use `node --trace-warnings ...` to show where the warning was created)
(node:13248) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
server started at 8000
Connected to MongoDB Atlas

[ ]
```

Project output

The output of the online complaint registration and management system was executed in the browser.

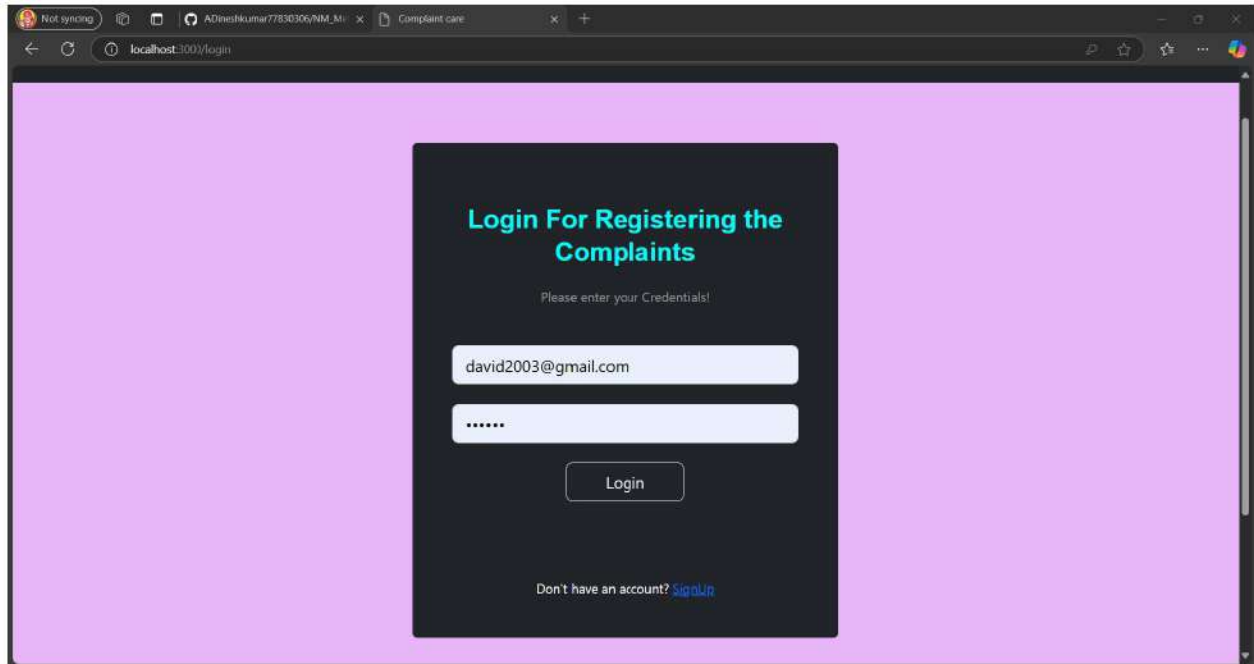
The home page



sign up page

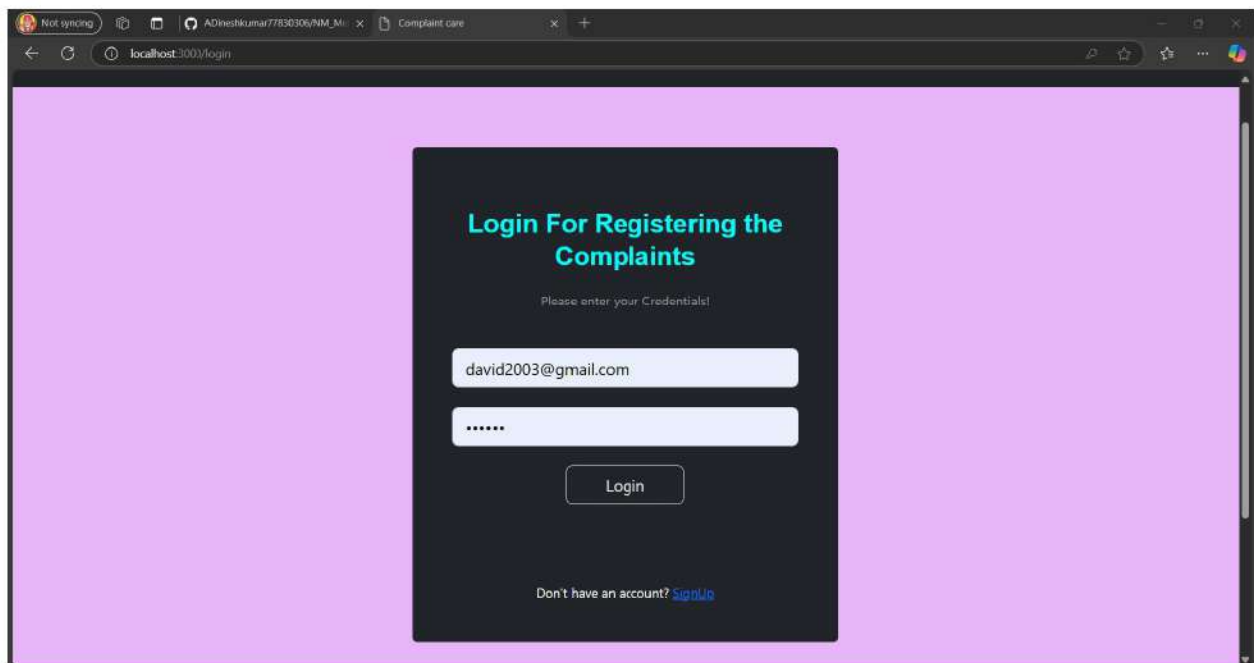
A screenshot of the "sign up" page of the "Complaint care" system. The browser address bar shows "localhost:3000/signup". The page features a dark central form area on a light purple background. The form is titled "SignUp For Registering the Complaint" in teal. Below the title, it says "Please enter your Details". The form contains four input fields: "Enter Your Full Name", "Enter Your e-mail", "Enter Your Password", and "Enter Your Mob:". To the right of the "Enter Your Mob:" field is a dropdown menu with options "Ordinary", "Admin", and "Agent". Below the dropdown is a "Select User" button. At the bottom of the form is a "Register" button. A link "Select User Type" is also visible below the dropdown menu.

Login Page



The screenshot shows a web browser window with the address bar displaying 'localhost:100/login'. The page has a light purple background. In the center, there is a dark gray rectangular box containing the following elements:

- Title:** 'Login For Registering the Complaints' in bold teal text.
- Instruction:** 'Please enter your Credentials!' in small gray text.
- Email Input:** A white text input field containing 'david2003@gmail.com'.
- Password Input:** A white text input field with masked characters '*****'.
- Login Button:** A gray button with the text 'Login'.
- Link:** A link that says 'Don't have an account? [signUp](#)' in blue text.



This screenshot is identical to the one above, showing the same login page layout with the email 'david2003@gmail.com' and a masked password.

Complaint register page

Hi, david Complaint Register Status [LogOut](#)

Name Address

City State

Pincode Status
type pending

Description

[Register](#)

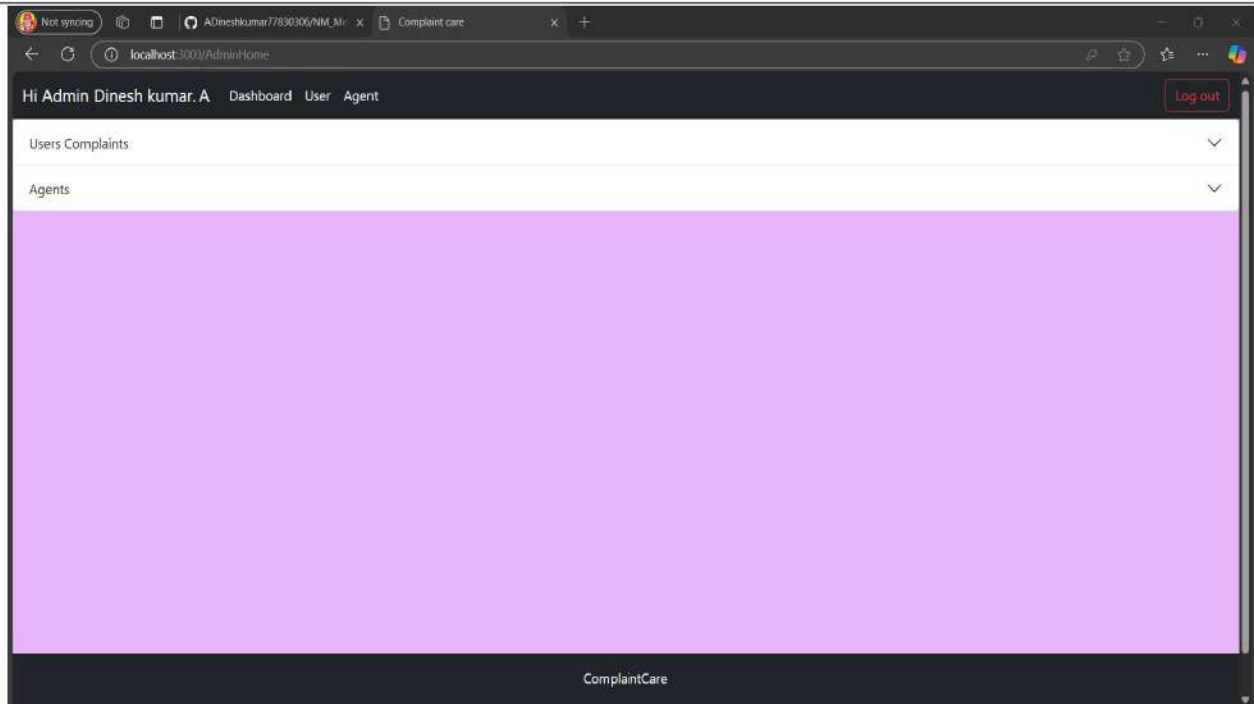
Status page

Hi, david Complaint Register Status [LogOut](#)

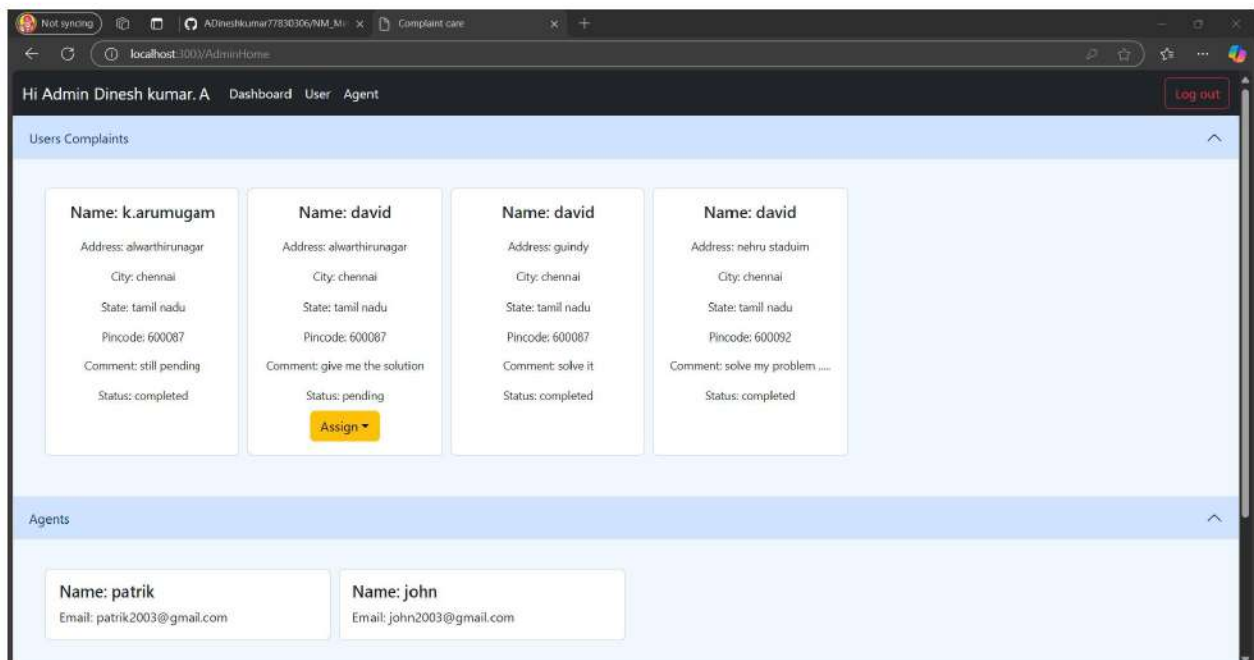
Name: david
Address: gundy
City: chennai
State: tamil nadu
Pincode: 600087
Comment: solve it
Status: completed
[Message](#)

Name: david
Address: nehru staduim
City: chennai
State: tamil nadu
Pincode: 600092
Comment: solve my problem
Status: completed
[Message](#)

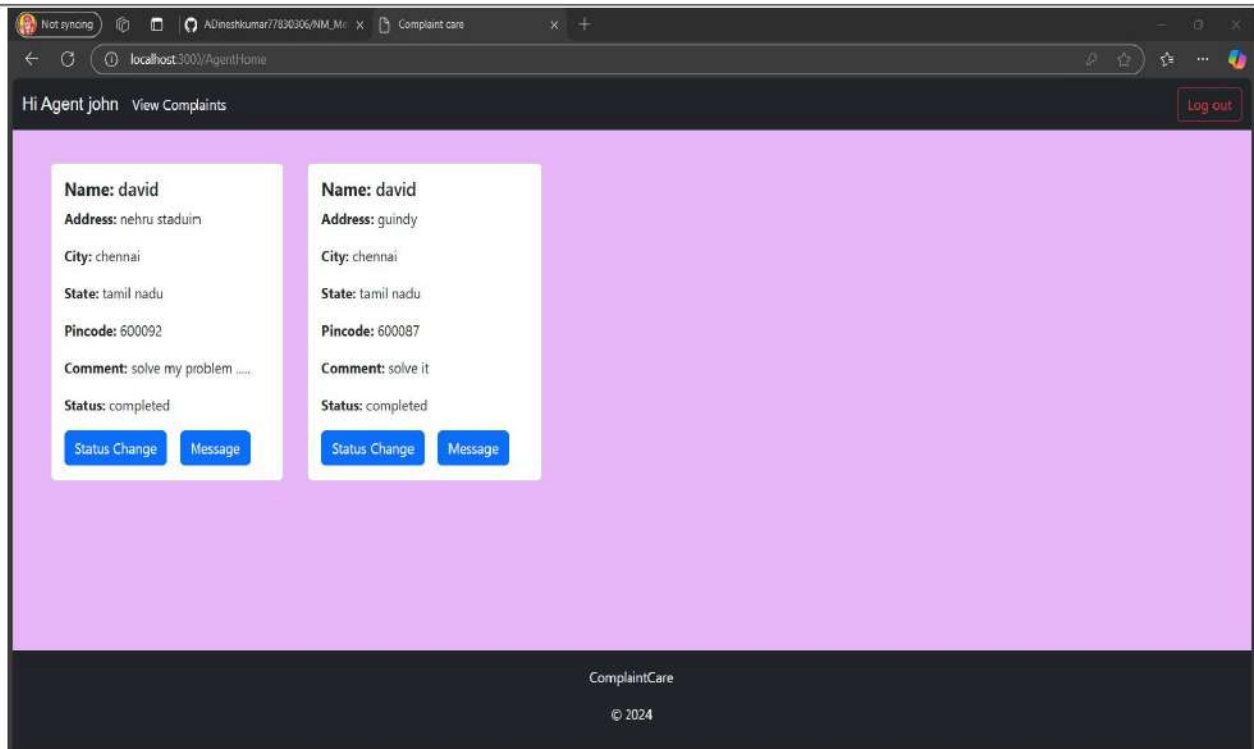
Admin login



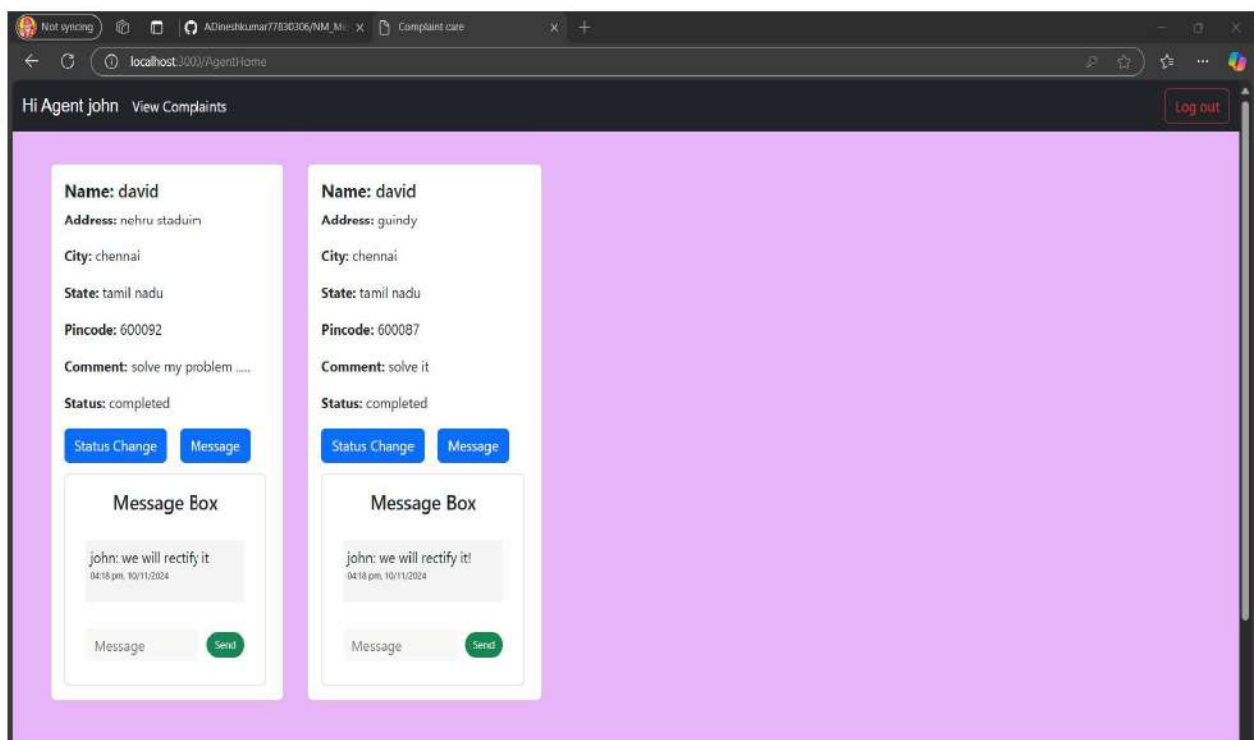
Admin Dashboard



Agent login



Agent dashboard



Advantages of online complaint registration and management system:

The Online Complaint Registration and Management System brings multiple advantages, optimizing the complaint-handling process for both users and organizations. Here are the primary benefits:

Customer Benefits:

1. Convenience: 24/7 accessibility
2. Easy registration: Quick and simple complaint submission
3. Transparency: Real-time status updates
4. Faster resolution: Expedited complaint resolution process
5. Increased satisfaction: Timely and effective issue resolution

Organization Benefits:

1. Improved efficiency: Automated complaint processing
2. Enhanced transparency: Clear complaint tracking and monitoring
3. Reduced workload: Streamlined complaint management
4. Data analysis: Insights from complaint data for quality improvement
5. Cost savings: Reduced manual processing and paperwork

Operational Benefits:

1. Centralized database: All complaints in one place
2. Standardized process: Consistent complaint handling
3. Automated notifications: Timely updates to customers and staff
4. Escalation mechanism: Efficient issue escalation and resolution

5. Performance metrics: Tracking and measurement of complaint resolution rates

Strategic Benefits:

1. Enhanced reputation: Demonstrated commitment to customer satisfaction
2. Competitive advantage: Differentiation through effective complaint management
3. Regulatory compliance: Adherence to industry regulations and standards
4. Continuous improvement: Identification of areas for improvement

Future enhancements:

some **future enhancements** that could be made to an Online Complaint Registration and Management System:

- ✓ **AI-Powered Complaint Categorization and Prioritization:** Implement AI algorithms to automatically categorize complaints based on content and prioritize them by urgency, ensuring critical issues are addressed promptly.
- ✓ **Automated Response System:** Use chatbots or automated response systems to provide instant answers to common questions, acknowledge complaint submissions, and offer updates, enhancing user satisfaction and reducing agent workload.

Conclusion:

In conclusion, an **Online Complaint Registration and Management System** modernizes the complaint process by making it efficient, transparent, and easy to use. The system allows users to submit complaints, monitor their status in real-time, and communicate with agents, which helps speed up resolutions and enhances user satisfaction.

Additionally, this platform provides organizations with insightful data for understanding recurring issues and improving their services. It's an essential tool for organizations committed to building user trust, enhancing operational efficiency, and managing complaints in a