PROJECT TITLE

RESOLVENOW: YOUR PLATFORM FOR ONLINE COMPLAINTS

Team Members:

Here List team members and their roles:

1. B. Vijay Babu (Full Stack Developer): Combines both frontend and backend responsibilities, ensuring smooth communication between the two. This role also handles bug fixing, feature integration, and overall system performance.

2. A. Dileep Kumar (Frontend Developer): Responsible for designing the user interface using React.js. This role focuses on ensuring a responsive, user-friendly design, as well as integrating the frontend with backend APIs.

3. A. Srija (Backend Developer): Develops the backend server using Node.js and Express.js, ensuring the creation of secure, scalable RESTful APIs, as well as handling authentication, data processing, and business logic.

4. A. Sathwik (Database Administration): Manages the MongoDB database, focusing on schema design, data integrity, and database optimization to ensure efficient data storage and retrieval.

Introduction:

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 main goal of an online complaint system is to make it easier and faster to send and handle complaints. With the help of this online tool, people can send their complaints from anywhere, without needing to visit in person or use paper forms.

This system helps save time by letting users see the progress of their complaints right away. It also keeps things clear and gives updates quickly. Complaints are sent to the right departments automatically, which helps fix problems faster.

The system also aims to solve complaints faster by using automatic steps. It sends each complaint to the right team or person based on set rules. This helps avoid delays and gets the problem to the right hands quickly. Because of this, users are more likely to get their issues fixed on time.

The platform also makes it easy for users and staff to talk to each other. It sends updates, messages, and notices so everyone knows what's going on.

Problem Statement:

The problem with old-style complaint systems is that they are slow, hard to manage, and not very clear. This often causes delays, unhappy users, and many complaints being left unsolved. In

most offices or places, people still send complaints using paper, phone calls, or emails. This makes it hard to check what is happening with each complaint.

Because there is no proper system, it takes a long time to reply and fix problems. Mistakes can happen, and no one knows for sure who is handling what. This causes confusion and makes the system weak. People don't get updates, so they stop trusting the service. For the staff, doing everything by hand is tiring and mistakes are common. It is also very hard to keep track of complaints or see how well the system is working. Without a proper tool, making the process better becomes nearly impossible

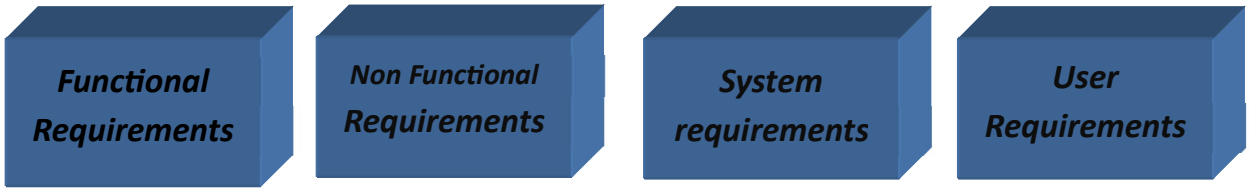
Solution:

The goal of an online complaint system is to fix the problems found in older ways of handling complaints. It gives users a simple website where they can send their complaints, check what's happening, and get quick updates. This makes the process faster and easier for everyone.

For the staff, the system brings all the complaints into one place. It helps organize the work, save time, and reduce mistakes by using automatic tools. This also makes it easier to manage and use the data.
As a result, users feel more heard, and this become dependent.

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.**



Below are the key requirements for the system:

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**:

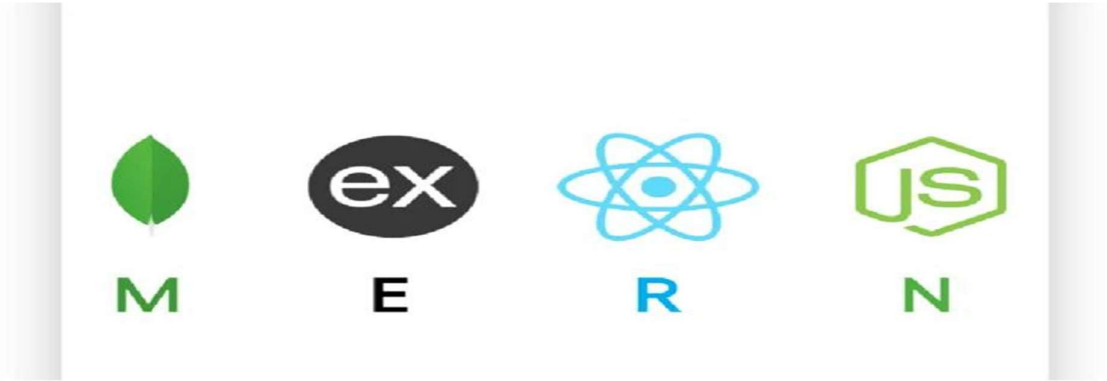
  - Ability to register, log in, and submit complaints.
  - Ability to track and check the status of complaints.
  - Receive notifications about complaint updates.

- **Admin**

  - Ability to manage user roles, assign complaints, and monitor complaint resolutions.
  - Ability to generate and view reports on complaint statistics and trends.

- **Agent**s:
  - Ability to view and manage complaints assigned to them.
  - Ability to update complaint status and add comments or solutions.

Technical Specifications:



MERN is a well-known set of tools used to build complete and powerful web applications. It includes MongoDB, Express.js, React, and Node.js. These four tools work together to help developers make fast, easy-to-use, and strong web systems. MERN is often used to create websites that can grow and handle many users.

Each part of the MERN stack plays an important role in building an Online Complaint Registration and Management System. These tools help store data, build the server, create the user interface, and connect everything smoothly. Using the MERN stack makes the system faster, safer, and more user-friendly.

MongoDB (Database):

MongoDB is a type of database that doesn't use tables like traditional ones. Instead, it stores data in a flexible format that looks like JSON. In the complaint system, MongoDB keeps user information, complaint details, status changes, and other needed data. It helps the system store everything in one place.

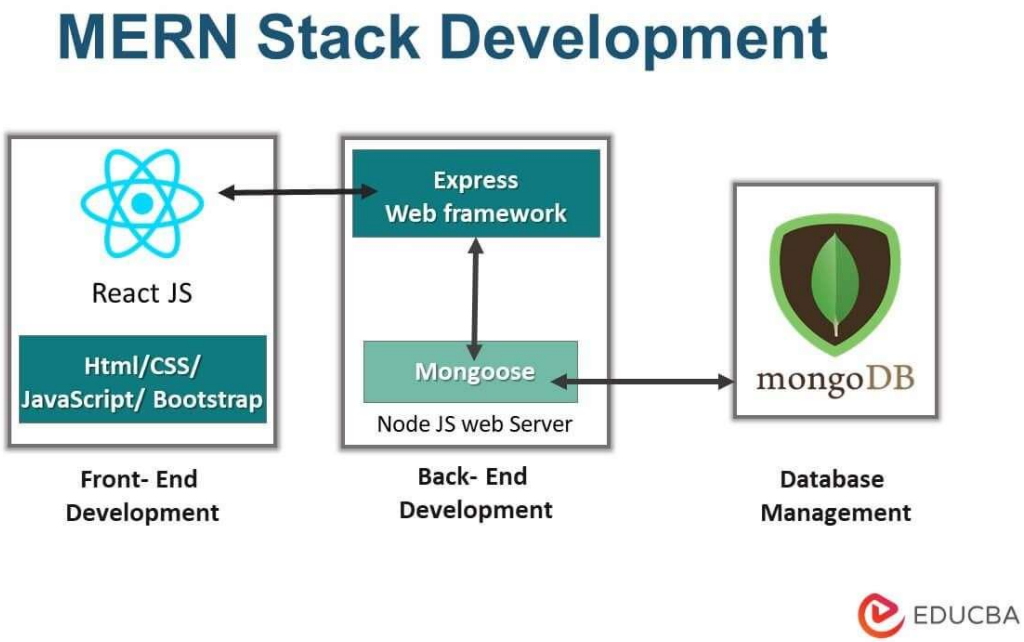
Since MongoDB doesn't need a fixed layout, it can easily manage different kinds of data, like various types of complaints or user needs.

Express.js (Backend Framework):

Express.js is a tool that works with Node.js to make backend work easier. It helps create APIs and manage how data moves between the frontend and backend. It takes care of things like sending and receiving data and setting up the right paths for different actions.

In the complaint system, Express.js handles tasks like sending a complaint, changing its status, and checking who the user is. It

connects the React frontend to the MongoDB database so everything works together. Express.js also makes hard jobs like handling errors and checking users much simpler. This helps the system stay safe and organized. With Express.js, building features for users, admins, and tracking complaints becomes smooth and faster.



React (Frontend Library):

React is a JavaScript tool used to build fast and interactive web pages. It helps create single-page applications, where only the parts that change are updated without reloading the whole page. This makes the website work faster and feel smoother for users.

In the complaint system, React gives users a clean and easy-to-use interface. People can send complaints, check their progress, and staff can view and solve them. React is built with small parts called components, which makes it easier to build and manage the site.

Developers can use the same parts, like login forms or dashboards, again and again.

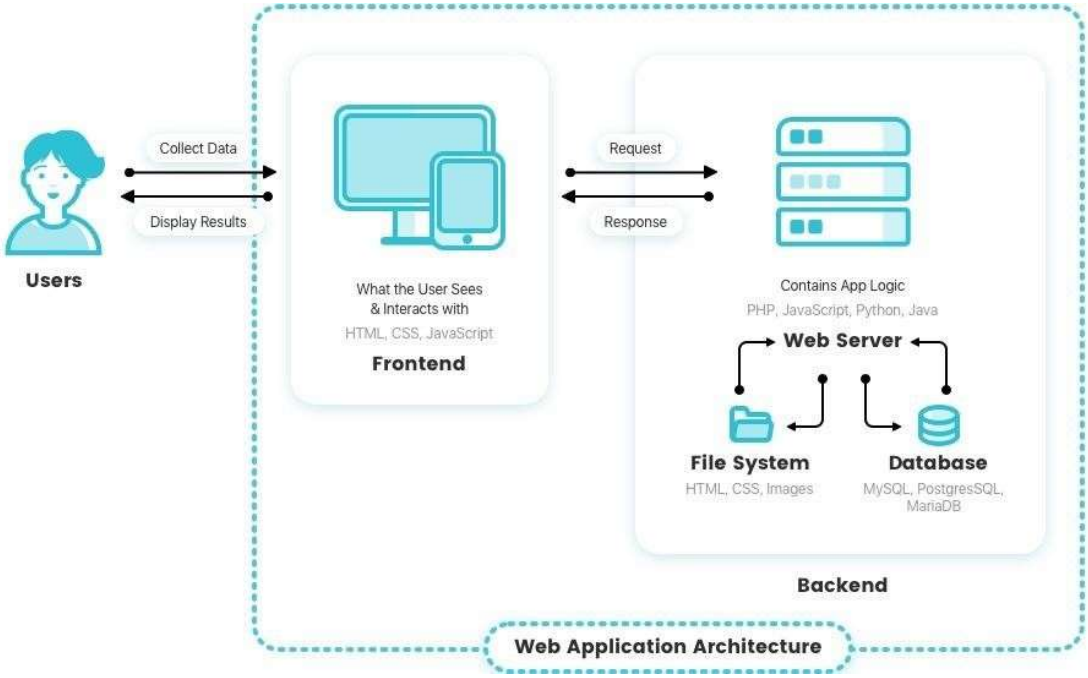
Node.js (Runtime Environment):

Node.js is a tool that lets JavaScript run on the server side, not just in the browser. This means developers can use the same language— JavaScript—for both the front and back parts of a website. This makes building the system easier and faster.

In the complaint system, Node.js runs the backend. It takes care of main tasks like handling user requests, running logic, and talking to the MongoDB database using Express.js. Node.js is great at handling many users at the same time, which is important for a system used by both users and staff.

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)

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).

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

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)

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

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.

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:

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

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

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 the 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.

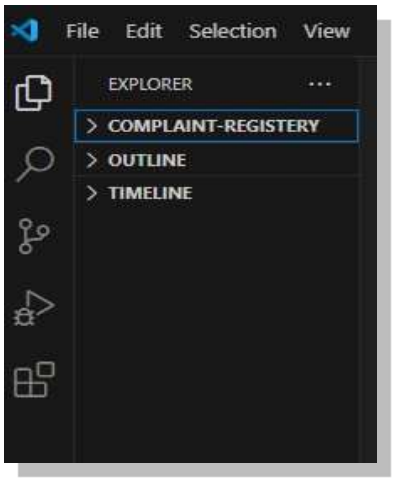
System Optimization: Admin are involved in continuous monitoring and optimization of the platform's performance. This includes overseeing system security, managing performance bottlenecks, and making sure the platform can scale as usage grows.

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:

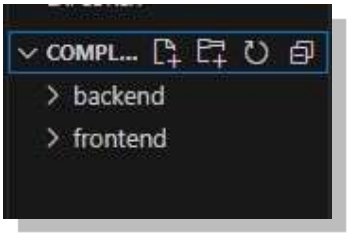
Now, for project configuration we used the visual studio code editor. In visual studio code editor, we created a folder called complaint-registery.



Create project folders and files:

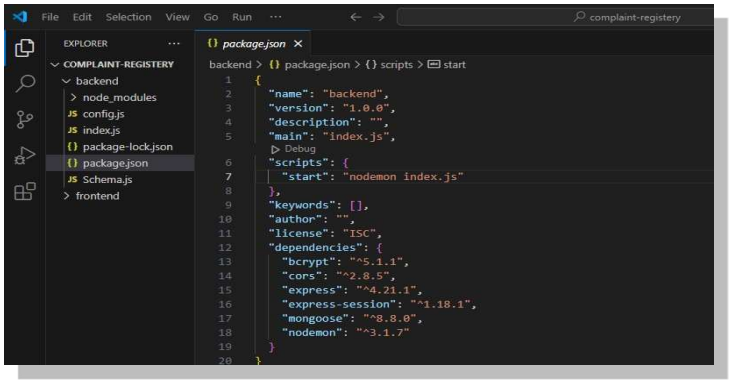
Now, firstly create the folders for frontend and backend to write the respective code and install the essential libraries.

- Frontend folders.
- backend folders



Install required tools and software:

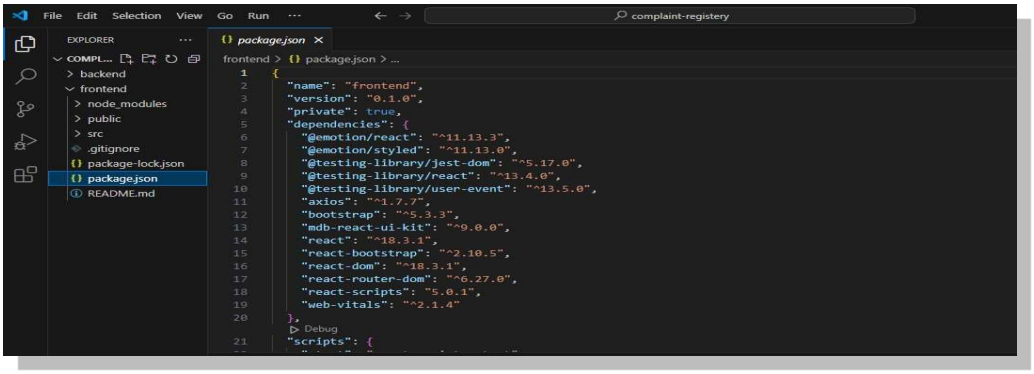
For the backend to function well, we use the libraries mentioned in the prerequisites. Those libraries include



- Node.js.
- MongoDB.
- Bcrypt
- Cors
- Express
- Express-session
- Mongoose
- Nodemon

Also, for the frontend we use the libraries such as

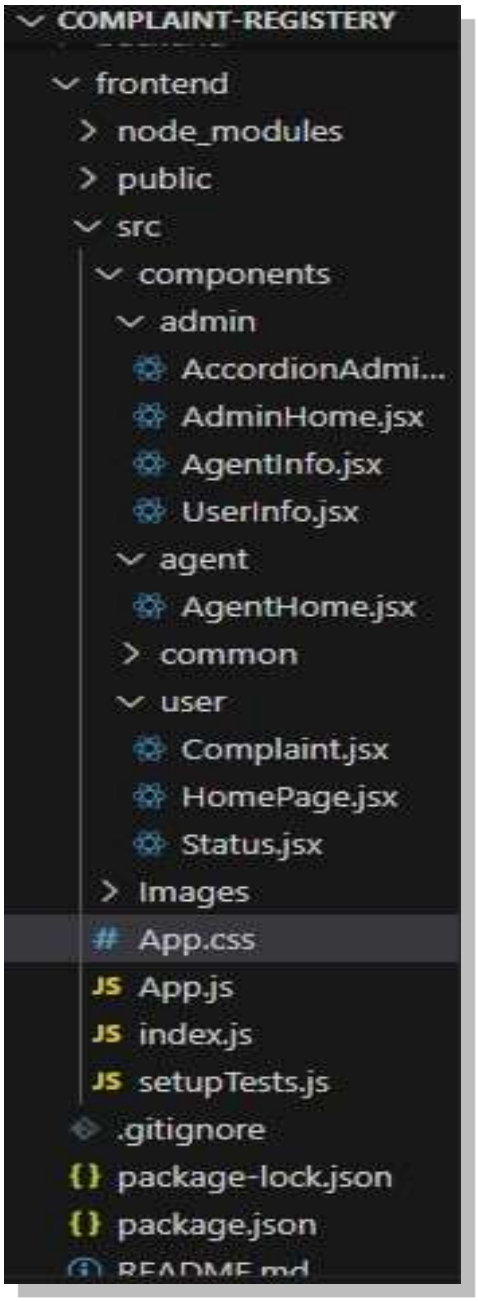
- React
- Bootstrap
- Axios
- React-dom
- React-bootstrap
- React-router-dom
- React scripts



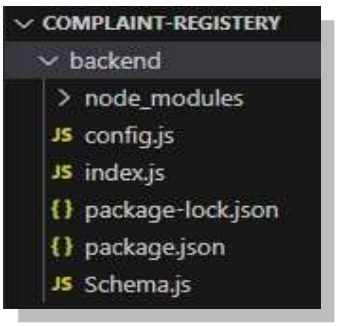
Project Implementation:

Add necessary backend and frontend folders and files

<u>Frontend:</u>

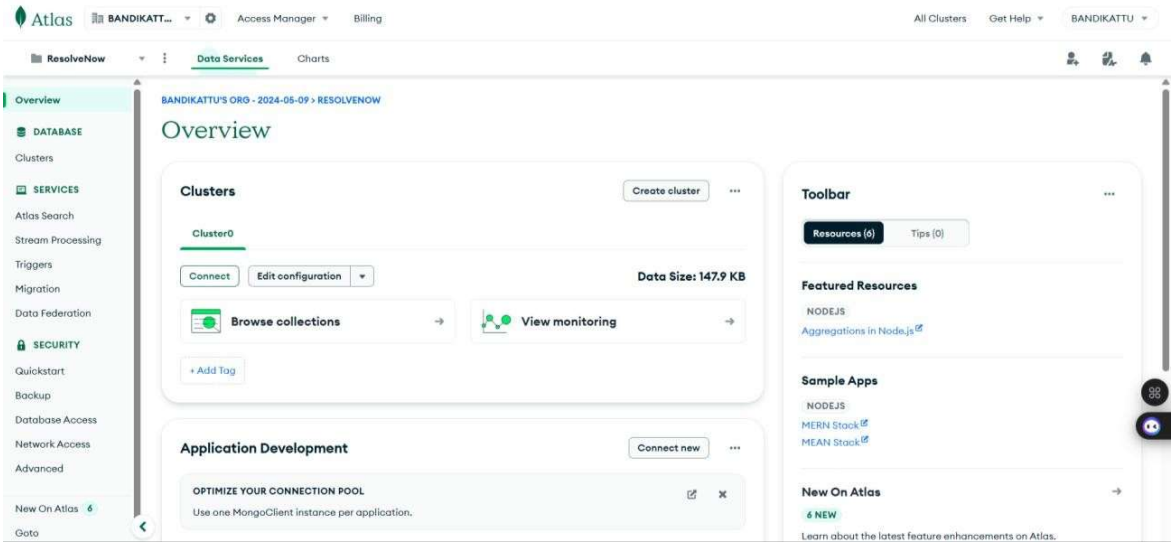

COMPLAINT-REGISTERY
frontend
node_modules
public
src
components
admin
AccordionAdmi...
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
user
Complaint.jsx
HomePage.jsx
Status.jsx
Images
App.css
App.js
index.js
setupTests.js
.gitignore
package-lock.json
package.json

Backend

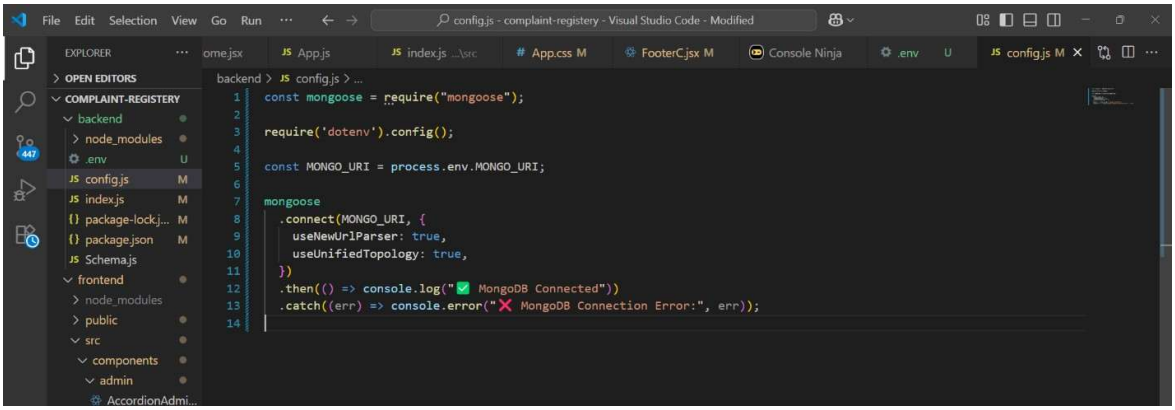

COMPLAINT-REGISTERY
backend
node_modules
config.js
index.js
package-lock.json
package.json
Schema.js

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

```
PROBLEMS 1   OUTPUT   TERMINAL   PORTS

PS C:\Users\bprat\complaint-registery> cd frontend
PS C:\Users\bprat\complaint-registery\frontend> npm start

> task1@0.1.0 start
> react-scripts start

✓ Console Ninja extension is connected to Webpack, see http
s://tinyurl.com/2vt8jxzw
Browserslist: caniuse-lite is outdated. Please run:
```

For Backend

```
PS C:\Users\bprat\complaint-registery> cd backend
PS C:\Users\bprat\complaint-registery\backend> npm start

> backend@1.0.0 start
> nodemon index.js

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
```

```
[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
server started at 8000
✅ MongoDB Connected
```

Testing:

Testing an Online Complaint Registration System in the MERN stack ensures that the backend (API), frontend (React components), and overall user flow work correctly. It includes testing API endpoints, form validations, user interactions, and performance to ensure everything functions smoothly.

Backend Testing (Node.js + Express)

- Unit Tests: Use Jest or Mocha with Supertest to test API endpoints (e.g., complaint submission).

Frontend Testing (React)

- Component Tests: Use Jest and React Testing Library to test individual components, like form validation and submission.

End-to-End Testing

- Use Cypress to test the full flow (register complaint, view status).
- Example: Simulate user actions, like form submission and success message.

Authentication Testing (if applicable)

- Test login and access control (using JWT or OAuth).

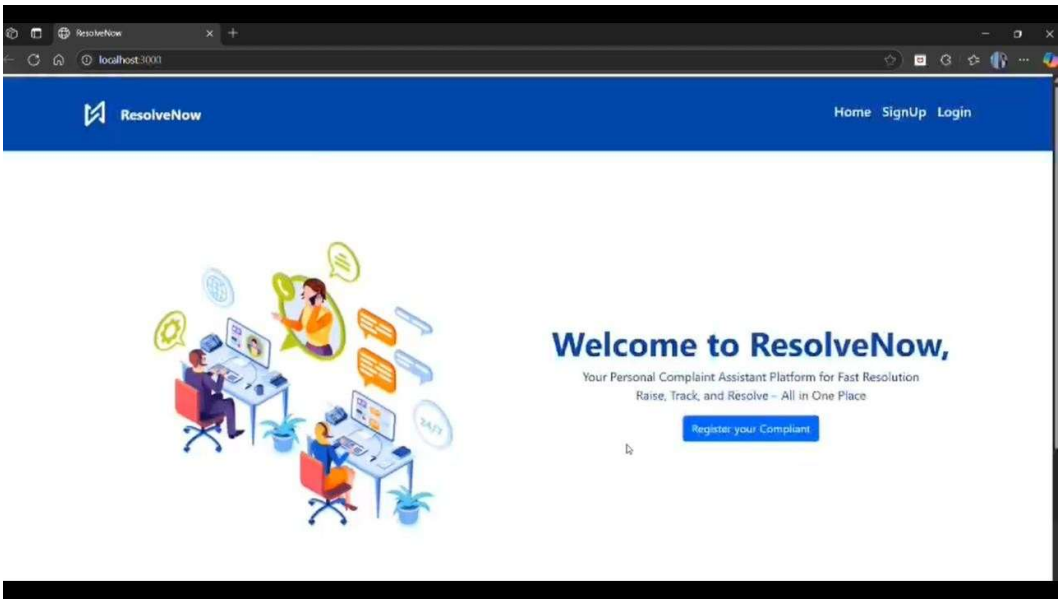
Performance Testing

- Use Artillery or JMeter to simulate high traffic and test API performance.

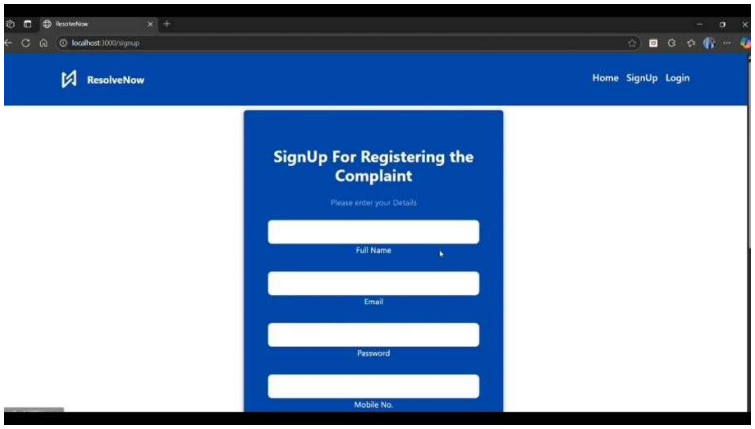
Project output

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

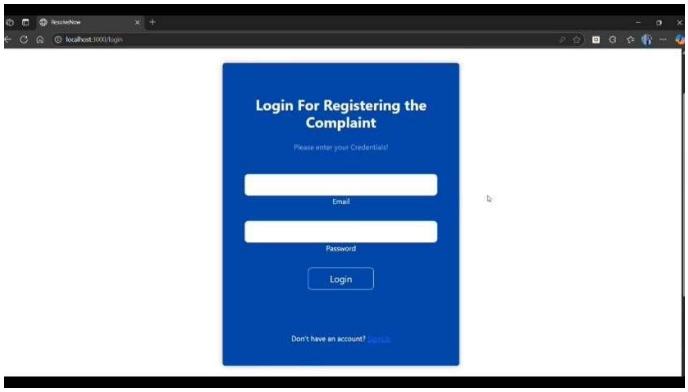
The home page

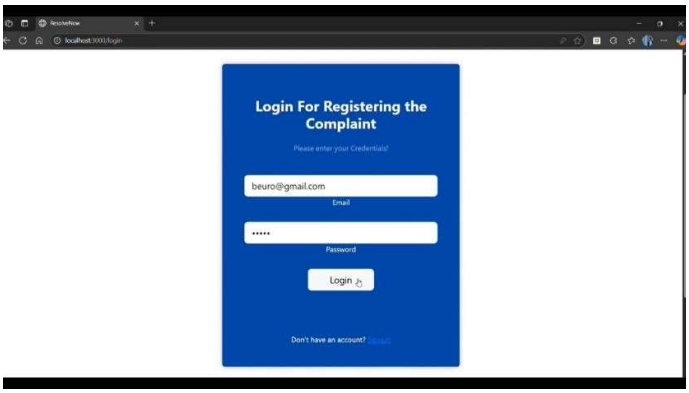


signup page

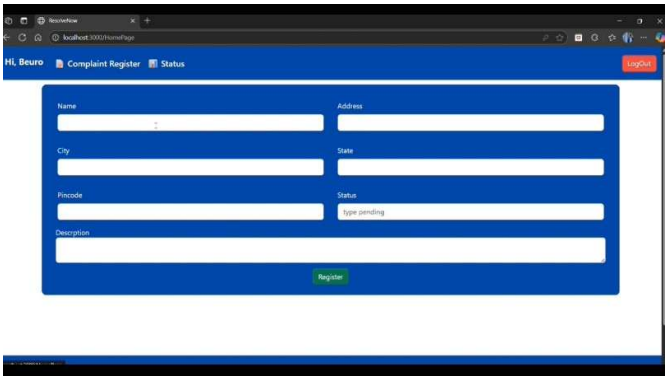


Login page

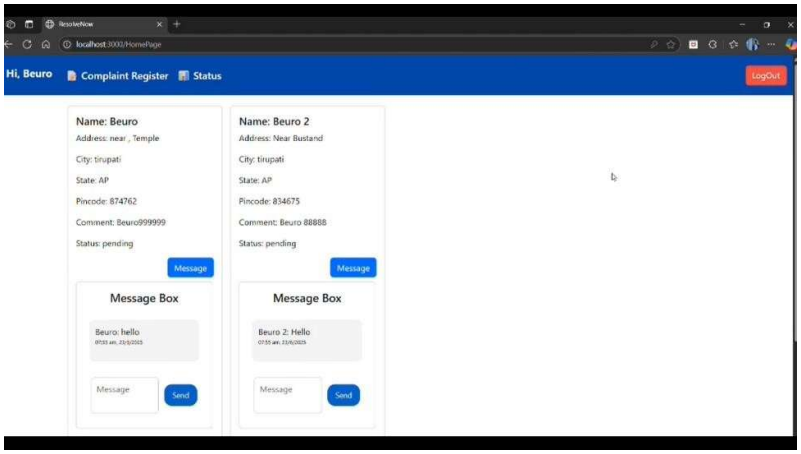




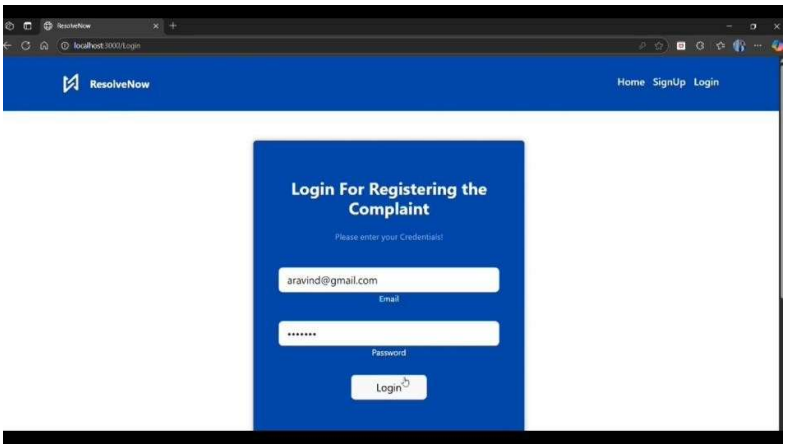
Complaint register page



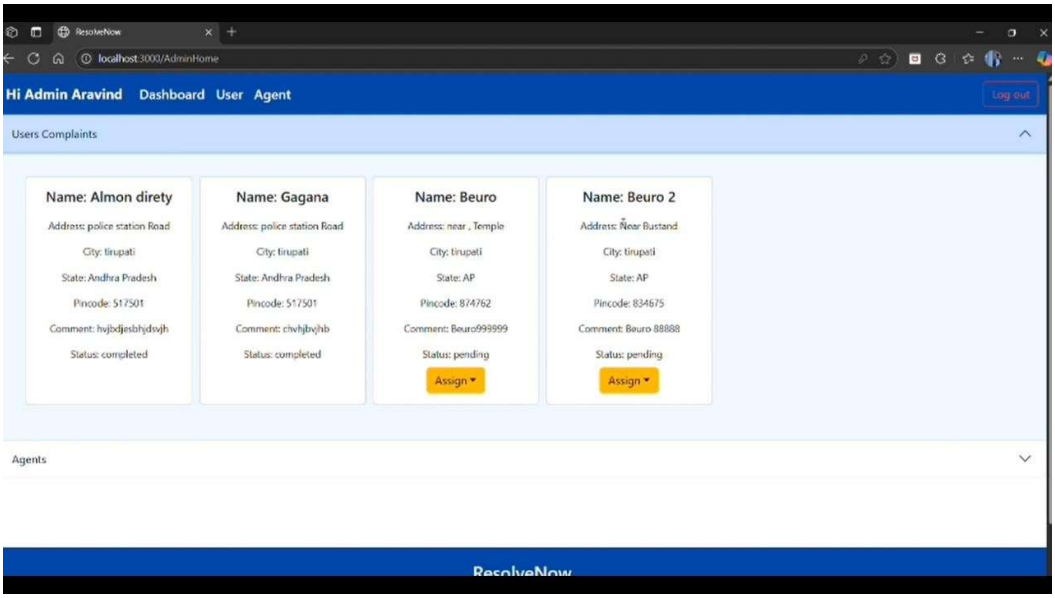
Status page



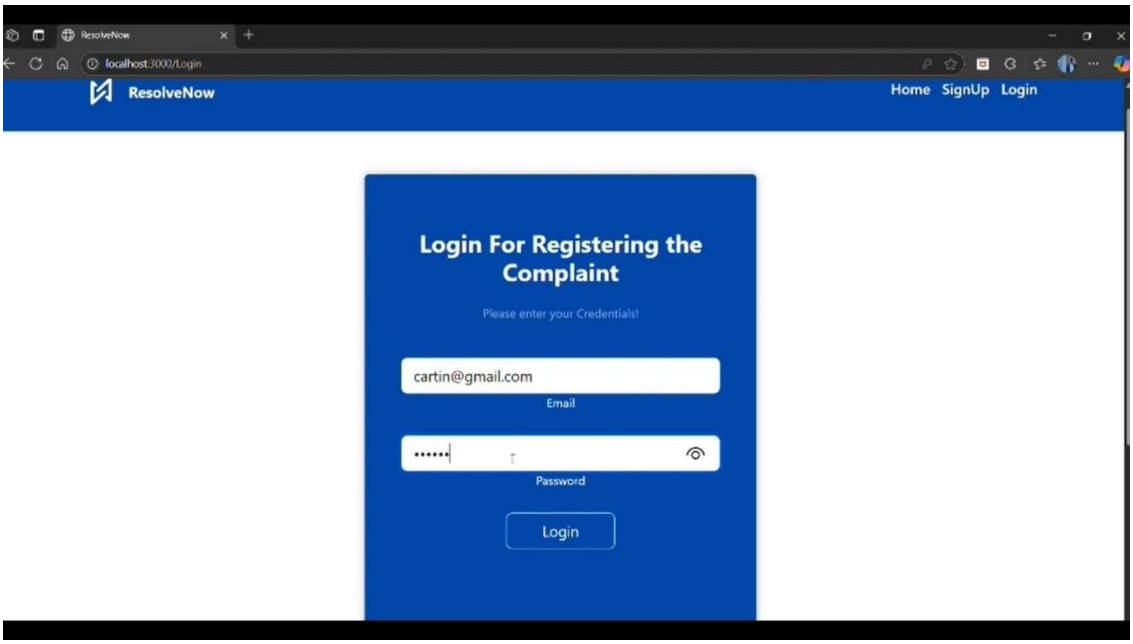
Admin login



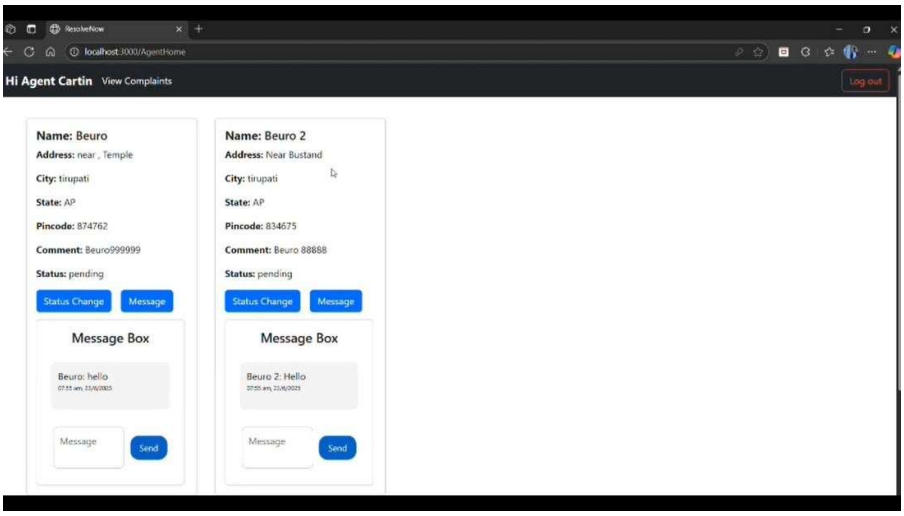
Admin dashboard



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 makes the complaint process faster, clearer, and easier for everyone. Users can send complaints, check their progress anytime, and talk to support staff, which helps solve problems quickly and keeps users happy.

This system also gives useful data to organizations to spot common problems and make their services better. In the future, features like smart complaint sorting, mobile use, and better reports can be added. The system is built to grow and improve over time. It is a helpful tool for any group that wants to build trust, work better, and handle complaints in a smooth and quick way.