MERN STACK POWERED BY MONGODB

Online Complaint Registration and Management System

A PROJECT REPORT

Submitted by

BACHU MANEESH	**113321104007**
JAVVAJI VAMSIKRISHNA	**113321104032**
K VENKATA KOUSHIK REDDY	**113321104037**
YANDODU VIVEK REDDY	**113321104118**

BACHELOR OF ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

VELAMMAL INSTITUTE OF TECHNOLOGY CHENNAI

601 204

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"Online Complaint Registration and Management System"** is the Bonafide work of **"BACHU MANEESH-113321104007, J VAMSIKRISHNA- 113321104032, K.VENKATA KOUSHIK REDDY-113321104037, YANDODU VIVEK REDDY-113321104118"** who carried out the project work under my supervision

SIGNATURE

Dr.V.P.Gladis Pushaparathi,

Professor,

Head of the Department,

Computer Science and Engineering,
Velammal Institute of Technology,
Velammal Knowlegde Park,
Panchetti, Chennai-601 204.

SIGNATURE

Mrs.Joice Ruby J,

Assistant Professor,

Naan Mudhalvan Coordinator,

Computer Science and Engineering,
Velammal Institute of Technology,
Velammal Knowledge Park,
Panchetti, Chennai-601 204

ACKNOWLEDGEMENT

We are personally indebted to many who had helped us during the course of this project work. Our deepest gratitude to the **God Almighty.**

We are greatly and profoundly thankful to our beloved Chairman **Thiru.M.V.Muthuramalingam** for facilitating us with this opportunity. Our sincere thanks to our respected Director **Thiru.M.V.M Sasi Kumar** for his consent to take up this project work and make it a great success.

We are also thankful to our Advisors **Shri.K.Razak**, **Shri.M.Vaasu**, our Principal **Dr.N.Balaji** and our Vice Principal **Dr.S.Soundararajan** for their never ending encouragement that drives us towards innovation.

We are extremely thankful to our Head of the Department **Dr. V.P.Gladis Pushaparathi and** Naan Mudhalvan Coordinator **Mrs. Joice Ruby J,** for their valuable teachings and suggestions.

The Acknowledgment would be incomplete if we would not mention word of thanks to our Parents. Teaching and Non-Teaching Staffs, Administrative Staffs and Friends for their motivation and support throughout the project. Thank you one and all

TABLE OF CONTENTS

PROJECT OVERVIEW

The Online Complaint Registration and Management System (OCRMS) is an advanced, web-based platform designed to streamline the complaint filing, tracking, and resolution process. It aims to replace traditional, inefficient systems with a more organized and automated approach, empowering users—such as citizens, customers, employees, and others—to submit complaints easily and efficiently.

OCRMS allows users to file complaints through multiple channels, including web forms, mobile apps, emails, and social media. Once submitted, complaints are automatically categorized and routed to the appropriate department for resolution. The platform provides users with a real-time complaint tracking system, enabling them to follow the progress of their issue through various stages, from submission to resolution, with automatic status updates along the way. This transparency reduces uncertainty and enhances user satisfaction.

The system incorporates automated workflows to ensure efficiency, including reminders, notifications, and escalation procedures when complaints are not resolved in a timely manner. Additionally, data analytics is used to analyze complaint trends and performance metrics, helping organizations identify patterns, improve service delivery, and address recurring issues proactively.

OCRMS also supports multi-platform integration, seamlessly connecting with existing systems like CRM, ERP, and communication tools, ensuring that complaint data is synchronized across all departments. The platform is customizable, allowing organizations to tailor the interface to their branding and needs, and offers multi-language support to cater to a diverse user base.

Security is a top priority, with encrypted communications, secure data storage, and compliance with data protection regulations, ensuring that sensitive information remains protected. The system is designed to be scalable, handling increasing volumes of complaints without compromising performance.

In summary, OCRMS is a robust, user-friendly solution that enhances the complaint management process by automating tasks, improving transparency, and providing valuable insights. It offers a scalable and customizable platform suitable for businesses, government agencies, and service providers, meeting the growing demands of modern complaint management.

OBJECTIVES

The primary objectives of the **Online Complaint Registration and Management System (OCRMS)** are centered around enhancing the efficiency, transparency, and accessibility of the complaint handling process for both users and organizations. The following objectives outline the key goals of the system:

1. **Ease of Complaint Submission from Any Location**
   OCRMS aims to provide users with the ability to submit complaints remotely from any location at their convenience. By leveraging modern technology, users can file complaints via various channels, including web forms, mobile apps, emails, or social media, eliminating the need for physical visits or paperwork. This anytime, anywhere approach is particularly beneficial for citizens, customers, or employees who may not have the time or resources to visit a physical office. Whether a complaint is about a product defect, service disruption, or other issues, users can quickly document and submit their concerns with ease. This increases accessibility, saves time, and streamlines the complaint registration process.

2. **Real-time Complaint Tracking and Progress Monitoring**
   Transparency is a core component of OCRMS. Once a complaint is submitted, the system assigns it a unique tracking number and provides users with real-time tracking of their complaint's progress through various stages—acknowledgment, investigation, resolution, and closure. Regular updates are sent automatically to the user, keeping them informed about the actions being taken on their complaint. This level of transparency not only ensures that users know exactly where their complaint stands, but it also enhances accountability and user satisfaction. The ability to track complaints in real time reduces frustration caused by uncertainty, fostering a sense of trust and confidence in the system. Additionally, users can raise follow-up queries, ensuring the process is clear and that the complaint is actively being addressed.

3. **Differentiated Access Levels for Users, Complaint Handlers, and Administrators**
   Data security and privacy are paramount in any complaint management system. OCRMS establishes different access levels for various user types—complaint submitters (users), complaint handlers (employees or departments), and administrators (system managers or supervisors)—to ensure the confidentiality and integrity of sensitive information.

   - **Users**: Have access to submit complaints, track the status of their complaints, and provide feedback upon resolution. Their access is restricted to viewing only their own complaint history and statuses.

   - **Complaint Handlers**: Authorized personnel or departments responsible for investigating, managing, and resolving complaints have access to view complaints assigned to them, update their status, and communicate with users when necessary. They may also escalate complaints if required.

   - **Administrators**: Have the highest level of access, overseeing the system's functionality, assigning complaints, managing user accounts, generating reports, and ensuring smooth operation. They can view all complaints and monitor system performance, providing oversight to maintain efficiency and accuracy. Administrators are also responsible for ensuring that security protocols are adhered to, and user data is properly protected.

By implementing role-based access controls, OCRMS ensures that sensitive complaint data is only accessible to authorized individuals, minimizing the risk of unauthorized data breaches or misuse. Additionally, these distinct roles contribute to an organized and efficient workflow, allowing the system to scale easily across different departments and user groups.

In summary, the key objectives of OCRMS are designed to improve the complaint management process by providing ease of access, real-time transparency, and enhanced security through role-based access. These objectives not only streamline the process for users and organizations but also improve overall user engagement, accountability, and efficiency in addressing complaints.

TECHNOLOGY STACK

The **Hostel Complaint Registration System** utilizes a modern, scalable, and secure technology stack designed to provide a seamless and efficient experience for users and administrators. The system employs a variety of tools for frontend development, backend functionality, database management, authentication, and deployment. Below is a detailed breakdown of the technology stack used in this system:

FRONTEND TECHNOLOGIES

- **React.js:** A popular JavaScript library used to build dynamic, component-based user interfaces. React.js provides a responsive, interactive frontend experience for users, allowing them to submit complaints, track progress, and interact with the system.
- **HTML5, CSS3, and JavaScript**: These are the foundational technologies for structuring and styling the user interface. They ensure a responsive design and compatibility across different screen sizes, ensuring the system is accessible on desktops, tablets, and smartphones.
- **Bootstrap / Material UI / Ant Design**: These are CSS frameworks and UI component libraries used to design visually appealing, consistent, and mobile-responsive interfaces. They provide pre-designed UI components like buttons, forms, and modals to enhance the user experience.

BACKEND TECHNOLOGIES

- **Node.js**: A JavaScript runtime environment used for handling backend logic and server-side operations. Node.js is chosen for its non-blocking, event-driven architecture, making it well-suited for handling multiple concurrent requests.
- **Express.js**: A web application framework built on top of Node.js, Express.js simplifies routing and API development, making it easier to build scalable backend services for handling user requests and managing complaints.

- **RESTful API**: The backend communicates with the frontend through a RESTful API. This API allows for creating, reading, updating, and deleting complaints, managing user roles, and interacting with the database.

DATABASE TECHNOLOGIES

- **MongoDB (NoSQL Database):** MongoDB is used for storing user data, complaint information, and related records. Its flexibility allows for a dynamic data structure, making it easier to adapt to the evolving needs of the system.
- **Mongoose (ODM for MongoDB):** Mongoose is used for modeling data and ensuring smooth interactions with the MongoDB database. It simplifies data validation and query management, ensuring the integrity of stored data.

AUTHENTICATION AND SECURITY

- **JWT (JSON Web Token):** JWT is used for secure user authentication and session management. It allows users to remain authenticated across sessions without re-entering credentials, while maintaining a stateless communication between the frontend and backend.
- **Bcrypt.js:** Bcrypt.js is employed to hash user passwords before storing them in the database, ensuring that sensitive user data remains secure.
- **HTTPS:** To ensure secure data transmission between the client and server, HTTPS is used to encrypt communications and prevent potential man-in-the-middle attacks.

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

Machine Specification:

- **Operating System:** Windows 8 or higher (recommended for compatibility with modern software).
- **Processor**: Multi-core processor (Intel i5 or higher) for efficient performance.
- **RAM**: Minimum of 8 GB (16 GB preferred) for smooth multitasking.
- **Storage**: At least 50 GB of free space (SSD preferred for faster performance).
- **Internet**: Stable internet connection with 30 Mbps download speed for seamless development and remote deployment.

SOFTWARE REQUIREMENTS

1. **Operating System:**
   - **Windows 8 or higher** (also compatible with macOS and Linux for development).
2. **Node.js**:
   - Latest stable version of **Node.js** for backend development. Enables running JavaScript on the server side and facilitates efficient request handling.
3. **MongoDB**:
   - **MongoDB Community Server** for NoSQL database management. Used to store user data, complaints, and other system records.
   - MongoDB provides scalability, flexibility, and high availability.

4. **VWeb Browsers for Testing**:
    - **Google Chrome** and **Mozilla Firefox** for cross-browser testing.
    - Ensures the system works across different browsers and platforms, improving user accessibility.
5. **Text Editor/IDE**:
    - **Visual Studio Code (VS Code)** or **Sublime Text** for writing and editing the system code.
    - Offers integrated tools for debugging, Git support, and code management.
6. **Version Control**:
    - **Git** for version control, enabling code tracking and team collaboration.
7. **Postman**:
    - Tool for API testing to verify backend functionalities and ensure correct data handling.

ADDITIONAL TOOLS:

1. **GitHub**:
    - For storing and managing code repositories, enabling version control and collaboration.
2. **Docker**:
    - Used for containerizing the application, ensuring consistent deployment across different environments.
3. **Cloud Platforms**:
    - **Heroku / AWS / Google Cloud / Microsoft Azure** for hosting and deploying the application with scalability in mind.

FEATURES

1. USER REGISTRATION AND AUTHENTICATION

- **User Accounts**: The system allows users (whether customers, employees, or citizens) to create personal accounts, enabling them to manage and track their complaints easily. Registration involves providing essential details like name, email, contact number, and password. Once registered, users can log in to their accounts to submit new complaints, track existing ones, and interact with support personnel.

- **Authentication:** To ensure secure access, the system incorporates multi-factor authentication (MFA), a security feature that adds an extra layer of protection to user accounts. With MFA enabled, users are required to verify their identity using a second method of authentication, such as an OTP (One-Time Password) sent to their mobile device or email, alongside their regular login credentials. This dual authentication process significantly reduces the risk of unauthorized access, as it ensures that even if someone gains access to a user's password, they cannot log in without the second verification step.

- **Role-based Access Control:** The system supports role-based access control (RBAC), which defines different permission levels for various users. The three primary roles include:

    - **End Users** (Customers, Citizens, or Employees) who can file and track complaints.

    - **Admins** who have full control over the system, including the ability to resolve complaints, assign priorities, and access reports.

    - **Support Staff or Customer Service** Agents who can view, manage, and respond to complaints based on their assigned roles and permissions.
      This ensures that users have appropriate access to the system's functionalities based on their roles, maintaining data security and efficient workflow.

2. COMPLAINT SUBMISSION

- **Complaint Form**:
  The system features a simple, intuitive form for submitting complaints. This form is designed to be user-friendly, allowing users to easily input the required information such as:
    - **Complaint Category**: Type of issue (e.g., service, product, maintenance).
    - **Description**: A brief description of the complaint.
    - **Urgency**: The level of urgency (e.g., low, medium, high, or urgent).

The design ensures that users can quickly and accurately submit their complaints with all relevant details, reducing the chances of missing important information.

- **Attachment Upload**:
  Users can attach supporting documents, such as images, videos, or documents, to their complaints. This feature is useful when users need to provide evidence or additional context for their complaint, helping to expedite the resolution process. For instance, a photo of a damaged product or a screenshot of an issue with a service can be directly uploaded, enhancing the clarity of the complaint.

- **Complaint ID Generation**:
  Each complaint submitted through the system automatically generates a unique Complaint ID. This ID acts as a reference number for tracking purposes, ensuring that users can easily monitor the status of their complaints at any time. The generated ID allows both users and administrators to quickly search for and retrieve specific complaints within the system.

3. COMPLAINT CATEGORIZATION AND PRIORITIZATION

- **Category Selection**:
  When filing a complaint, users can select from various pre-defined categories such as:

- **Service Issues**: Problems related to customer service or support.
- **Product Issues**: Complaints regarding defective or damaged products.
- **Support Issues**: Difficulties with technical support or assistance.

The categorization ensures that complaints are directed to the appropriate department or team, allowing for faster and more effective resolution.

- **Priority Levels**:
  To further streamline the complaint resolution process, complaints can be assigned different **priority levels**. Users or admins can select from predefined priority levels like:
  - **Low**: Non-urgent issues that can be resolved in the normal course of time.
  - **Medium**: Issues that require attention but are not urgent.
  - **High**: Critical issues that need quick attention.
  - **Urgent**: Extremely critical issues that must be addressed immediately.

This helps prioritize complaints based on their severity, ensuring that more pressing issues are handled first, improving overall customer satisfaction and operational efficiency.

- **Automated Categorization**:
  The system incorporates **AI-based automated categorization**. Leveraging machine learning algorithms, the system can analyze the content of the complaint and automatically classify it into the appropriate category, without requiring manual input. This helps streamline the process, reduce human error, and route complaints to the correct department or team without delay. This automation increases efficiency, especially for high-volume complaints, and ensures faster response times.

PROJECT OVERVIEW

FRONTEND (CLIENT-SIDE)

- **React.js**:
    - Dynamic and interactive user interface (UI).
    - Single-page application (SPA) for a seamless user experience.
- **CSS Frameworks**:
    - Bootstrap, Material UI, and Ant Design:
        - Responsive and aesthetic UI components.
        - Ensures consistent design across different devices (mobile, tablet, desktop).
- **Communication with Backend**:
    - HTTP requests (via Axios or Fetch API) to interact with backend API for submitting and retrieving data.

BACKEND(SERVER-SIDE):

- **Express.js**:
    - Handles routing and processing of API requests.
    - Defines server-side logic for business operations (e.g., complaint submission, user management).
- **MongoDB**:
    - NoSQL database to store user information, complaints, feedback, etc.
    - Document-based format for flexible, scalable data storage.

- Provides fast data retrieval and efficient querying.

- **JWT (JSON Web Tokens)**:
  - Secure user authentication and session management.
  - Authenticates user identity and ensures authorized access to data.

AUTHENTICATION & SECURITY

- **Multi-factor Authentication (MFA)**:
  - Additional layer of security (e.g., OTP sent to mobile/email).
  - Prevents unauthorized access to sensitive user data.

- **JWT**:
  - Token-based authentication system for secure access to user data.
  - Ensures that only authorized users can perform specific actions (e.g., submitting complaints, viewing personal data).

DATABASE:

- **MongoDB** (Primary Database):
  - Stores data in a flexible, document-based format (e.g., user details, complaint info).
  - Scalable, with high performance for dynamic data.
  - Supports replication, sharding, and indexing for high availability and reliability.

- **Optional Relational Database** (if required):
  - PostgreSQL, MySQL, or Oracle for structured data with relationships.
  - Ideal for complex, tabular data relationships (e.g., multiple tables for complaints, feedback, and user info).

INSTALLATION AND SET UP

PREREQUISITES:

Before starting the setup, make sure the following software is installed on your machine:

1. **Node.js and npm**: Node.js is required for running the server-side application, and npm (Node Package Manager) will help you manage dependencies.
    - Download and install Node.js from the official website: Node.js
    - Verify installation by running the following commands in your terminal

```
node -v
npm -v
```

2. **MongoDB Community Server**:
   MongoDB is the database used for this project to store user data, complaints,and feedback.

- Download and install MongoDB from: MongoDB Download Center
- Start MongoDB by running the following command in your terminal:

```
mongod
```

STEP-BY-STEP SETUP

1. Clone the Repository

To begin the setup, you need to clone the project repository to your local machine. You can do this by using the following Git command:

```
git clone https://github.com/Techwoof/NaanMudhalavan_Complaint-registration-Backend
```

3. **Backend Setup**

Once you have cloned the repository, navigate to the backend directory to set up the backend part of the application**.**

Open your terminal and navigate to the backend folder:

```
cd online-complain-registration/code/backend
```

Install all required dependencies:

```
npm install
```

Create a **.env** file in the backend directory to store sensitive environment variables

```
MONGODB_URI=mongodb://localhost:27017/complaintdb
JWT_SECRET=your_jwt_secret_key
```

Modify the MongoDB connection string in the connect.js file located in the config folder to match the string in your **.env** file:

- Open connect.js and replace the connection string with the one specified in your **.env** file.

```
const mongoose = require('mongoose');
const { MONGODB_URI } = process.env;

mongoose.connect(MONGODB_URI, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
}).then(() => console.log('MongoDB connected'))
.catch((err) => console.log('Error connecting to MongoDB: ', err));
```

Start the backend server:

```
npm start
```

The backend server will now be running and accessible via **http://localhost:8000**.

3. Frontend Setup

Next, set up the frontend part of the application, which provides the user interface for submitting complaints and tracking their status.

Navigate to the frontend directory:

```
cd hostel-complaint-app/code/frontend
```

Install all required frontend dependencies:

```
npm install
```

Start the frontend server:

```
npm start
```

This will start the frontend server, which will run on **http://localhost:3000**. The React application will be accessible in your browser.

4. Access the App

Once both the frontend and backend servers are up and running, you can access the application by navigating to the following URLs:

- **Frontend**: http://localhost:3000
    - This is the user-facing interface where complaints can be submitted and tracked.
- **Backend**: http://localhost:8000
    - This is the backend API that processes the complaint data and manages user interactions.

WORKFLOW AND USAGE

1. USER/COMPLAINANT

- **Role Summary**: The user (or complainant) is the individual who submits complaints within the system. This role includes customers, citizens, or employees depending on the organization or context. Users are the primary source of complaints and their interaction with the system forms the starting point for the entire process.
- **Functionalities**:
    - **Complaint Submission**: Users are able to easily submit complaints through an intuitive form on the platform. They provide details such as the type of issue, description, urgency, and any supporting documents (e.g., images or videos). This submission process is designed to be straightforward to encourage participation from all types of users.
    - **Complaint Tracking**: After submitting a complaint, users receive a unique complaint ID that allows them to track the status of their complaint at any time. The status updates are provided in real-time, helping users stay informed on the progress of their issue.

- **Feedback and Rating**: Once a complaint is resolved, users can rate the service they received and provide feedback on the process. This feedback is crucial for identifying areas of improvement and ensuring the continued improvement of the complaint handling process.

2. AGENT/CUSTOMER SERVICE REPRESENTATIVES

- **Role Summary**: Agents or customer service representatives are responsible for managing the complaints submitted by users. They are the main point of contact between the complainant and the organization, ensuring that complaints are resolved within the stipulated time frame.

- **Functionalities**:

  - **Complaint Management**: Agents have access to a dashboard where they can view all complaints assigned to them. They can see the complaint details, user information, and the current status of the complaint. Their main task is to assess the complaint and take necessary action to resolve it.

  - **Communication with Users**: Agents may need to communicate with users to clarify the complaint or request additional information. This communication can take place through the platform's messaging system, ensuring that all interactions are logged and traceable.

  - **Resolution of Complaints**: Agents are responsible for offering solutions to the issues raised by users. This could include providing technical support, offering compensation, or arranging a service to resolve the issue.

  - **Escalation**: If an agent is unable to resolve a complaint or if the complaint requires higher-level intervention, they can escalate the issue to a supervisor or manager.

3. SUPERVISOR/TEAM LEADER

- **Role Summary**: Supervisors or team leaders oversee the performance of agents and ensure that complaints are being handled efficiently. They are responsible for ensuring that the complaints assigned to their team are being addressed in a timely manner and that any escalated issues are appropriately resolved.

- **Functionalities**:

    - **Team Monitoring**: Supervisors have access to the system's reporting tools and can monitor the progress of all complaints within their team. This includes tracking response times, resolution rates, and customer satisfaction.

    - **Complaint Escalation Management**: If an agent encounters a complex issue or if a complaint is not resolved in a timely manner, supervisors have the authority to escalate the complaint further to the management or administrative level. They ensure that critical issues are addressed promptly.

    - **Team Performance Reporting**: Supervisors can generate reports showing the performance of their team members. These reports include metrics such as the number of complaints handled, average resolution time, and customer satisfaction ratings.

    - **Workload Distribution**: Supervisors are responsible for managing the workload within their team. They ensure that complaints are evenly distributed among agents, based on the severity and complexity of the issues.

4. ADMIN

- **Role Summary**: The **admin** role is the most privileged user in the **Hostel Complaint Registration System** and has the highest level of access and control within the platform. Admins are responsible for ensuring the smooth and efficient functioning of the entire system, from configuring the platform to overseeing the complaint management processes. Their primary responsibility is to ensure that the platform runs according to organizational standards and that all

users (agents, supervisors, and complainants) have the tools and support they need to perform their roles effectively.

Functionalities:

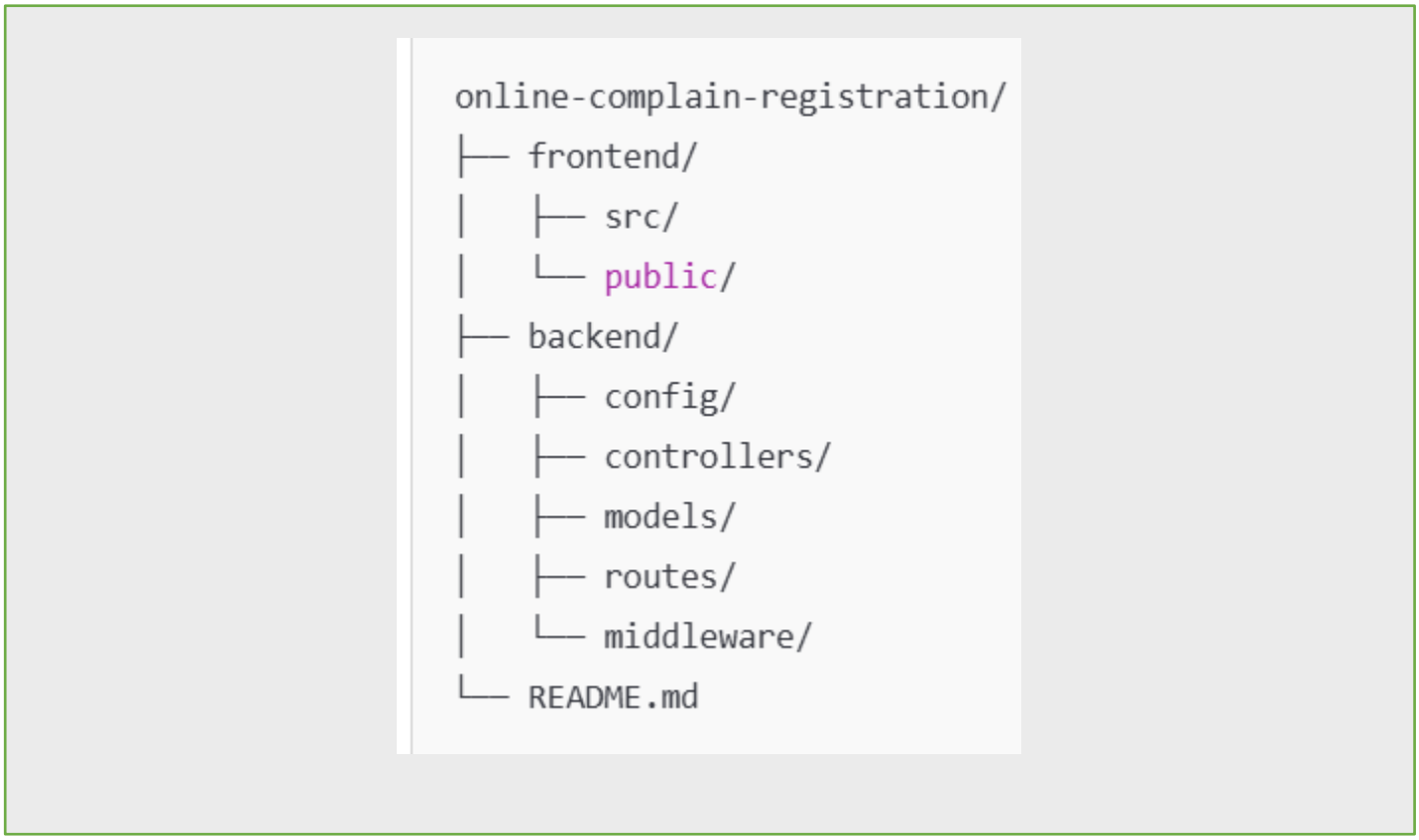
- **System Configuration**: Admins are responsible for configuring key settings within the platform. This includes setting up complaint categories, defining urgency levels, and customizing workflows to suit the organization's processes. Admins can also manage notification settings, ensuring users and agents receive relevant updates on complaints.

- **User and Role Management**: Admins can create, update, or delete user accounts for both internal users (agents, supervisors) and external users (complainants). They assign roles and access levels to ensure that each user has the appropriate level of access to the system. They can assign specific roles and permissions to each user, ensuring that each individual has access to the appropriate tools and data based on their responsibilities. For example, agents may only have access to complaints they are assigned, while supervisors can monitor the progress of all complaints in their team, and admins can access everything in the system.

- **Reporting and Analytics**: Admins have access to comprehensive reporting tools that provide insights into the system's performance. These reports can help track key metrics such as the total number of complaints, average resolution time, and customer satisfaction levels. This data is critical for making data-driven decisions to improve the complaint management process.

- **Security and Permissions**: Admins ensure the security of the platform by managing access controls. They can define who has permission to view or edit specific data and ensure that the system is compliant with security and privacy regulations. Admins also monitor and audit system usage to identify any unusual activities.

FOLDER STRUCTURE



1. frontend/

This directory contains all the files related to the frontend of the application. It holds all the UI components, styles, and client-side logic necessary for users to interact with the system.

- **src/**: This folder contains the main logic and components of the React-based frontend.
    - **Components**: This is where reusable UI components like forms, buttons, tables, etc., are stored.
    - **App.js**: The root file where the core React components are defined and routed.
    - **State Management**: Depending on the architecture, you may find Redux, Context API, or other state management logic here.

    - **Services**: API calls and external communication logic are stored here to interact with the backend.

- **public/**: This folder contains static assets that will be served directly to the client, such as:
    - **index.html**: The main HTML file where the React app is injected.
    - **Images, fonts, icons**: Any static files required for the frontend are stored here.

2. backend/

This directory holds all the files related to the backend logic, which handles business processes, interactions with the database, user authentication, and more. It is built with Node.js and Express.js.

- **config/**: This folder contains configuration files and environment variables required to run the backend.
    - **Database Configuration**: Files that define how the application connects to the database (in this case, MongoDB).
    - **Environment Variables**: Files such as .env where sensitive information like API keys, JWT secrets, and database credentials are stored securely.
- **controllers/**: This folder holds the logic for handling various API requests and operations related to complaints.
    - **ComplaintController.js**: Manages the logic of complaint submission, updates, and retrieval.
    - **UserController.js**: Handles user registration, authentication, and profile management.
    - Controllers are responsible for interacting with the database models to create, read, update, and delete records.

- **models/**: This folder contains the data models for MongoDB, which define the structure of the collections in the database.
    - **Complaint.js**: Defines the schema for a complaint, including fields such as description, category, priority, etc.
    - **User.js**: Defines the schema for users, which includes their login credentials, role (complainant, agent, admin), and personal details.
    - These models are used to interact with the MongoDB database, such as saving new complaints or fetching complaints for a specific user.

- **routes/**: This folder defines the API routes used by the frontend to interact with the backend.
    - **ComplaintRoutes.js**: Handles routes related to complaints, such as /submitComplaint, /getAllComplaints, /updateComplaintStatus, etc.
    - **UserRoutes.js**: Handles user-related routes, such as /register, /login, /updateProfile, etc.
    - **Authentication Routes**: These routes handle login, logout, and token generation processes for secure access to the application.

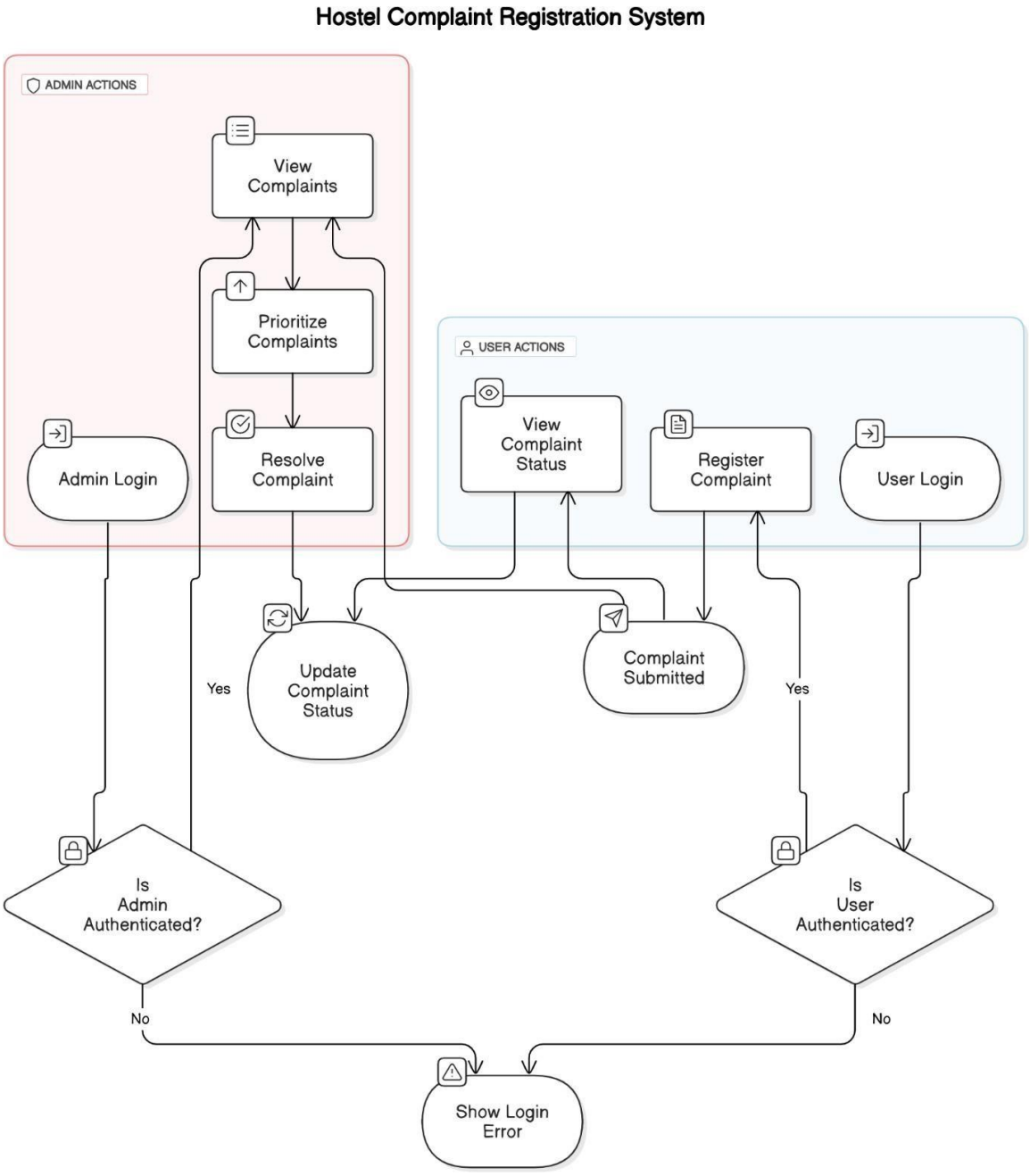
- **middleware/**: This folder contains the middleware functions that sit between the request and the response cycle, allowing for actions such as validation, authentication, and authorization.
    - **authMiddleware.js**: Ensures that a user is authenticated (e.g., checking JWT tokens).
    - **validationMiddleware.js**: Ensures that incoming requests have valid data (e.g., checking that required fields are provided in a complaint submission).

3. README.md

The **README.md** file is located at the root of the project and provides important details about the project, such as:

- **Project Overview**: A brief explanation of the project, its purpose, and how it works.
- **Installation Instructions**: Step-by-step guidelines for setting up the development environment, including installing dependencies and running the application.
- **Usage Instructions**: Information about how to use the application, how to register complaints, track progress, and more.
- **Technologies Used**: A list of technologies and frameworks used in the project, such as Node.js, React, MongoDB, etc.
- **Contributions**: Guidelines for contributing to the project, reporting issues, or submitting pull requests.

ER DIAGRAM


Hostel Complaint Registration System
ADMIN ACTIONS
View Complaints
Prioritize Complaints
Resolve Complaint
Admin Login
USER ACTIONS
View Complaint Status
Register Complaint
User Login
Update Complaint Status
Yes
Complaint Submitted
Yes
Is Admin Authenticated?
Is User Authenticated?
No
No
Show Login Error

CHALLENGES FACED

Overwhelmed System:

- High complaint volumes, particularly during peak periods, can strain the system, causing slow response times or even system crashes.
- Users may experience delays in complaint resolution, leading to dissatisfaction and frustration.
- To manage such volumes effectively, the system must be designed for scalability, with robust infrastructure that can handle increased traffic. This includes implementing load balancing techniques, optimizing database queries, and using asynchronous processing to ensure the system remains responsive during high-demand times.

User Privacy:

- Managing sensitive user information, such as personal details and complaint records, poses a significant privacy risk. Any breach of this data could have serious legal and reputational consequences.
- Users expect their information to be securely handled, and failure to do so can result in loss of trust and legal repercussions.
- To address this, the system must employ strong encryption protocols for data storage and transmission, implement secure authentication mechanisms (such as multi-factor authentication), and comply with relevant privacy regulations (e.g., GDPR) to protect user data from unauthorized access or leaks.

Service Level Agreement (SLA) Compliance:

- Ensuring timely resolution of complaints according to established SLAs can be challenging, especially when there is an unexpected surge in complaints or limited resources available to address them.
- If SLAs are not met, it can result in penalties, damage to reputation, and a decline in user satisfaction.
- To maintain SLA compliance, it’s essential to have a well-organized workflow with clear complaint prioritization, resource allocation, and automation for follow-ups and reminders. Additionally, contingency plans

should be in place to address spikes in complaint volumes, ensuring that all complaints are handled within the specified timeframes.

4. Integration with Existing Systems:

- Integrating the complaint registration system with existing organizational tools and platforms (such as CRM, ERP, or communication systems) can be complex and time-consuming.
- Data synchronization across multiple platforms may face compatibility issues, leading to inconsistencies and errors in complaint records.
- To overcome this challenge, the system needs to support seamless API integrations and provide real-time data syncing. Proper planning and testing are essential to ensure that all systems communicate effectively without data loss or discrepancies.

5. User Adoption and Training:

- Encouraging users to adopt the new complaint registration system can be challenging, especially if they are accustomed to traditional complaint handling methods.
- Users may struggle with understanding how to navigate the system, leading to delays in complaint submission or improper usage of features.
- To address this, comprehensive training materials, user guides, and a support system should be provided. Additionally, the system's interface should be intuitive and user-friendly, ensuring that users of all technical skill levels can easily use it. User feedback should also be incorporated to continuously improve the platform and enhance user experience.

6. System Downtime and Reliability:

- Unexpected system downtimes or technical failures can disrupt complaint registration and resolution, affecting user experience.
- Ensuring high availability and minimizing downtime through regular maintenance and backups is essential.
- Implementing failover mechanisms and monitoring tools can help detect and resolve issues before they impact users.

FUTURE ENHANCEMENTS

Immutable Complaint Logs:

- Storing complaint history on a blockchain will provide an immutable, tamper-proof record of all complaints, ensuring that the information is secure and transparent. Blockchain technology can guarantee that once a complaint is logged, it cannot be altered or deleted, making it particularly useful in industries that require strict regulatory compliance and auditability, such as finance, healthcare, and government services.
- This enhancement would in still greater trust in the system, as users and organizations can rely on an indisputable history of their complaints and resolutions. Additionally, it would make it easier for regulatory bodies or auditors to review the complaint data without concerns about manipulation or errors.

Smart Contracts for SLA Compliance:

- Smart contracts, powered by blockchain technology, can automatically enforce Service Level Agreements (SLAs) by defining the rules and conditions for complaint resolution. If a complaint is not resolved within the specified time frame, the smart contract could automatically trigger penalties or escalations, such as notifying higher management or issuing compensation to the affected user.
- This will ensure more reliable SLA compliance, reduce human error, and eliminate the possibility of negligence. Additionally, smart contracts can streamline the dispute resolution process by automating some of the decision-making, ensuring that SLAs are adhered to strictly and in a transparent manner.

Anomaly Detection:

- Implementing real-time anomaly detection algorithms powered by machine learning and artificial intelligence can help the system detect unusual patterns in complaint data. For example, if there is a sudden surge in complaints related to a particular product or service, the system can flag this and notify the relevant teams immediately.

- By spotting these patterns early, the organization can take proactive steps to address the root cause of the issue before it escalates further. This could lead to quicker resolutions, better customer satisfaction, and a more effective approach to quality control and continuous improvement. The ability to predict and prevent major issues will improve organizational efficiency and reduce the volume of complaints overall.

4. AI-Powered Chatbot Integration:

- Integrating an AI-powered chatbot into the complaint registration system would provide users with instant support 24/7, allowing them to submit complaints, track status, and get answers to common questions without needing to wait for human agents. The chatbot could help guide users through the complaint submission process, ensuring that all required details are provided, and provide real-time updates on the status of their complaints.
- Additionally, the chatbot could analyze recurring complaints or frequently asked questions to identify areas where the system or service could be improved. Over time, as the AI learns from user interactions, it would become more efficient at resolving simple complaints, freeing up human agents to handle more complex issues. This enhancement would improve user experience, reduce response times, and increase overall system efficiency.

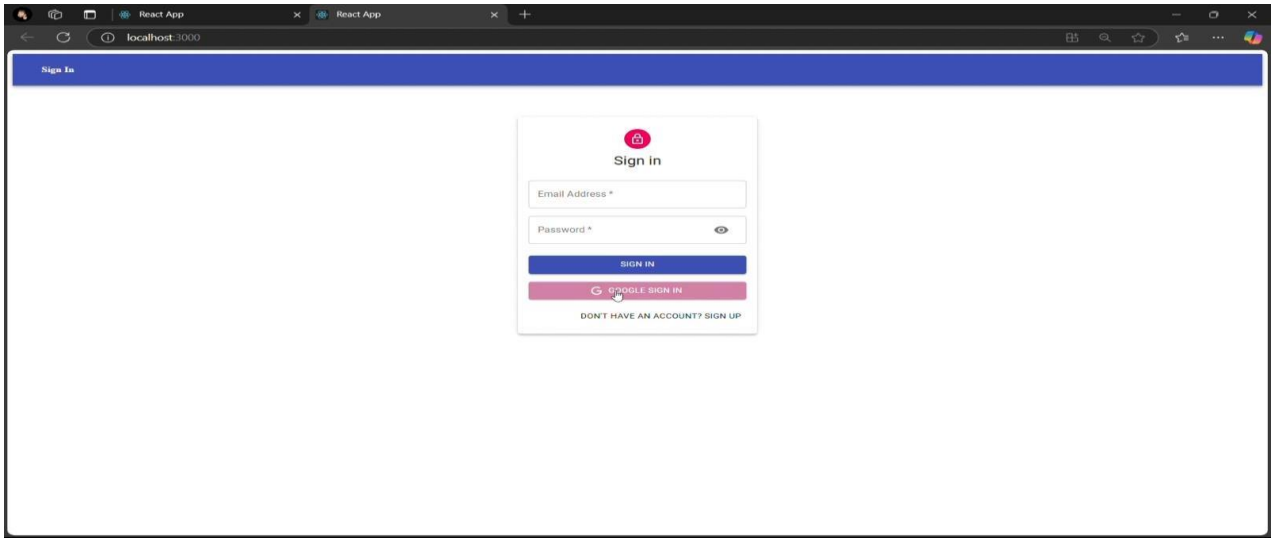
5. Integration with Social Media Platforms:

- Future enhancements could include integrating the complaint registration system with popular social media platforms like Twitter, Facebook, and Instagram, enabling users to submit complaints directly through these channels. This would offer a seamless and convenient way for users to register complaints without needing to access the system separately, increasing accessibility and engagement.
- Social media integration could also allow for real-time monitoring of public sentiment, enabling organizations to quickly identify and respond to complaints shared on social platforms. Furthermore, it could help improve customer satisfaction by providing users with more touchpoints to voice their concerns and receive timely updates, making the system more inclusive and adaptable to various user preferences.

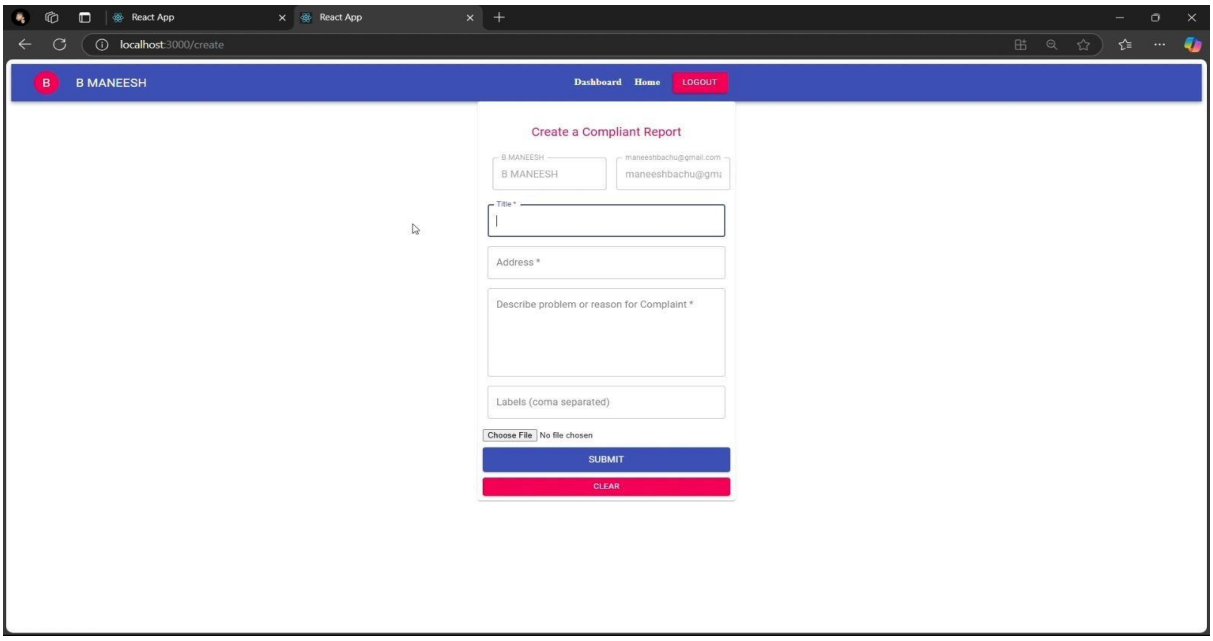
PROJECTION IMPLEMENTATION & EXECUTION

Our project implements an efficient Hostel Complaint Registration System, streamlining complaint submission, tracking, and resolution. Below are key features and the system workflow illustrated.

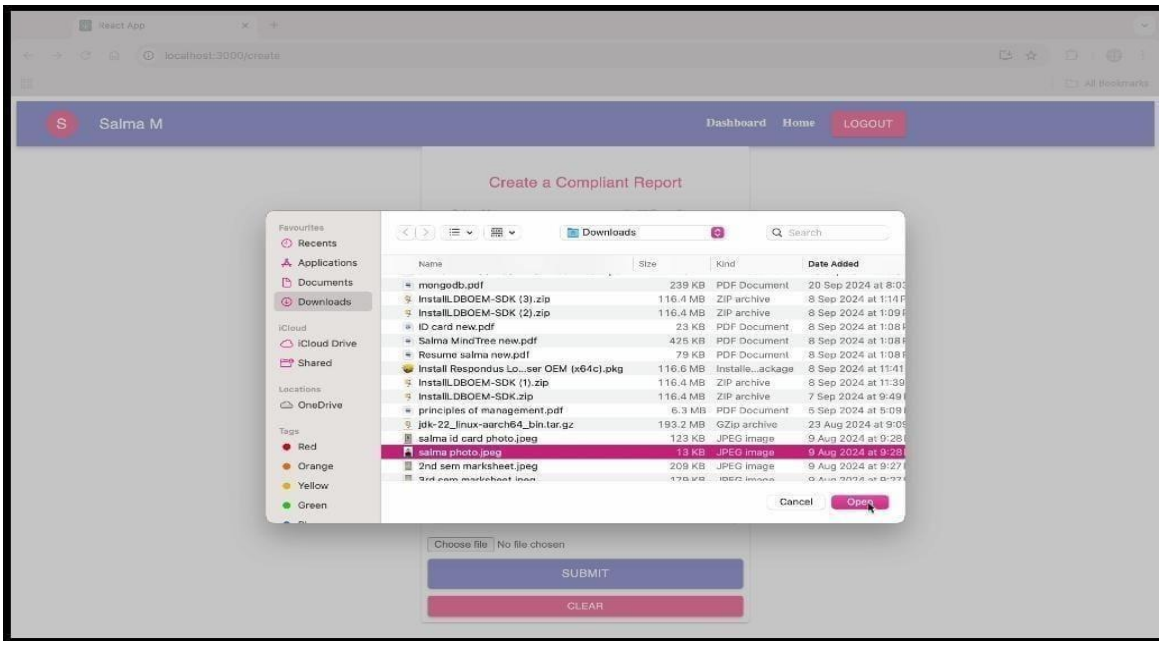
LOGIN PAGE:



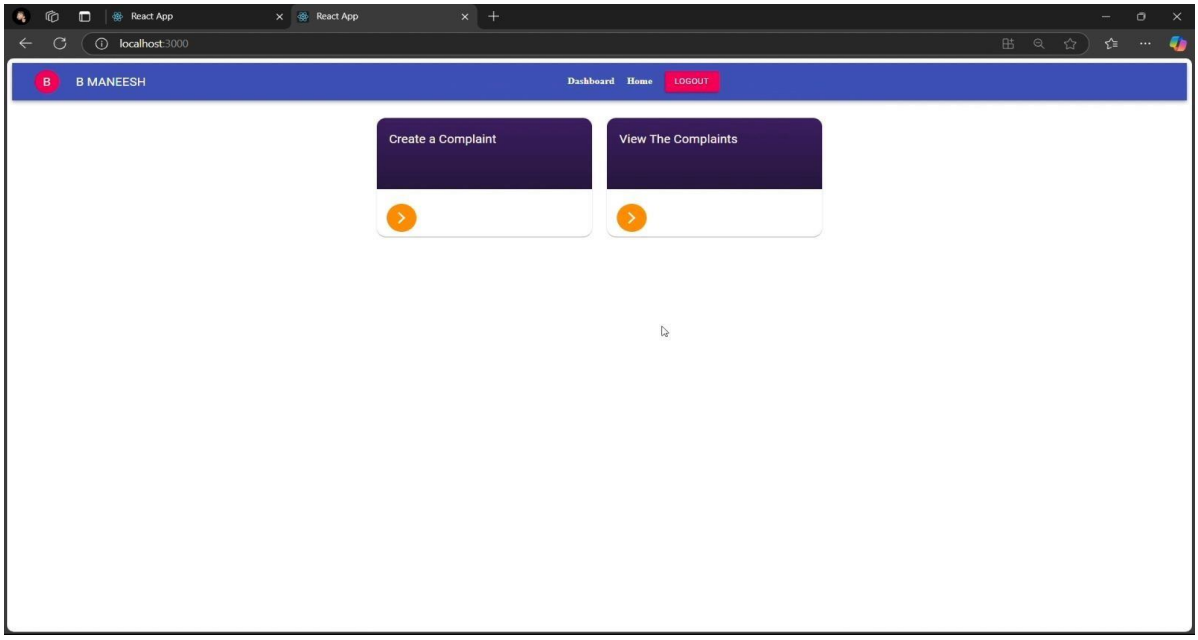
CREATING REPORT:



CHOOSING FILE:



VIEWING COMPLAINTS



CONCLUSION

An online complaint registration and management system plays a crucial role in enhancing an organization's ability to handle customer concerns efficiently, leading to improved customer satisfaction, quicker issue resolution, and stronger accountability. In today's fast-paced world, where customers and users expect prompt responses, such a system ensures that complaints are received, categorized, and tracked in a streamlined, automated manner. This reduces the chances of complaints being overlooked or delayed, which often leads to dissatisfaction.

By automating processes such as complaint intake, categorization, and prioritization, the system significantly reduces the time and manual effort involved in handling issues. Users can submit their complaints online, and once received, complaints are automatically categorized according to the type of issue and the urgency level, ensuring that they are routed to the appropriate team or individual for immediate action.

The system’s automated workflows further enhance this efficiency by managing routine tasks, such as sending follow-up notifications, reminders, or updates to both users and complaint handlers. These workflows ensure that complaints are resolved in a timely manner, preventing delays or missed deadlines. Additionally, the system provides a real-time tracking mechanism that allows users to monitor the progress of their complaints, bringing transparency and reducing uncertainty about the status of their issues.

One of the key benefits of such a system is its ability to provide detailed performance analytics. By collecting data on complaint types, response times, resolution rates, and user satisfaction, organizations can gain valuable insights into their operations. These insights enable businesses to identify recurring issues, spot trends, and take proactive steps to address root causes, which ultimately helps improve overall service delivery and customer satisfaction.

In conclusion, implementing an online complaint registration and management system transforms customer service by providing users with an accessible, transparent platform to file complaints. It equips organizations with tools to manage, analyze, and resolve complaints efficiently, reducing response times and enhancing communication. This system fosters trust and loyalty by enabling data-driven decision-making and improving overall customer satisfaction.