Full Stack Development with MERN

Project Documentation

I. Introduction

• **Project Title:** **Online Complaint Registration and Management System**

• **Team Members:**

1.V Srikanth– Project Manager & Testing: Responsible for testing, overseeing the entire project, ensuring timelines are met, and leading the team.

2.G K Hariharan – Database Administrator: Manages and optimizes the database, ensuring data integrity and efficient querying.

3.B Jayaprakash– Backend Developer: In charge of designing and implementing the server-side logic, APIs, and database interactions

4.N Afzal Hameeth – Frontend Developer: Develops and styles the user interface, ensuring a seamless and responsive user experience.

II. Project Overview

• **Purpose:** The Online Complaint Registration and Management System is designed to enhance the process of submitting, managing, and resolving customer complaints. This platform aims to optimize the complaint-handling workflow and improve customer satisfaction while ensuring compliance with industry and regulatory standards.

The system serves as a centralized hub for tracking and resolving issues, streamlining communication between users and the organization, and fostering transparency in the complaint resolution process.

• Features:

• User Registration:

- Allows users to create accounts for submitting and tracking complaints.

• Complaint Submission:

- Users can file complaints, providing details such as their name, complaint description, and address.

• Tracking and Notifications:

- Users can monitor the status of their complaints.
- Notifications are sent via email or SMS regarding updates or resolutions.

• User-Agent Interaction:

- Users can communicate directly with the assigned agent to discuss the complaint.

• Assigning and Routing Complaints:

- The system automatically assigns complaints to the appropriate department or personnel based on predefined criteria or intelligent routing algorithms.

• Security and Confidentiality:

- Ensures data protection with secure authentication, encryption, and access controls.
- Complies with data protection regulations.

III. Architecture

• Frontend:

- Developed using React, the app utilizes functional components and React Hooks for state management.

- The design is component-based, making it scalable and easy to maintain. React-Bootstrap is used for a sleek and responsive UI design.

• **Backend:**

- The server is built with Node.js and Express.js. It handles CRUD operations, user authentication, and integrates with a MongoDB database.

- RESTful API structure is implemented for clean and efficient communication between the frontend and backend.

• **Database:**

- MongoDB is used as the database. The schema is designed using Mongoose to handle data efficiently, with collections for users, properties, and bookings.

IV. Setup Instructions

• **Prerequisites:**

- Node.js: Ensure you have the latest version installed.
- MongoDB: A running instance of MongoDB is required for database operations.
- npm (Node Package Manager): Comes with Node.js.

• **Installation:**

1. Clone the repository:

```
https://github.com/srikanth230/Online-Complaint-Registration
```

2. Navigate to the project directory:

```
cd ResolveX
```

3. Install dependencies for both frontend and backend:

- For frontend:

```
npm install
```

- For backend:

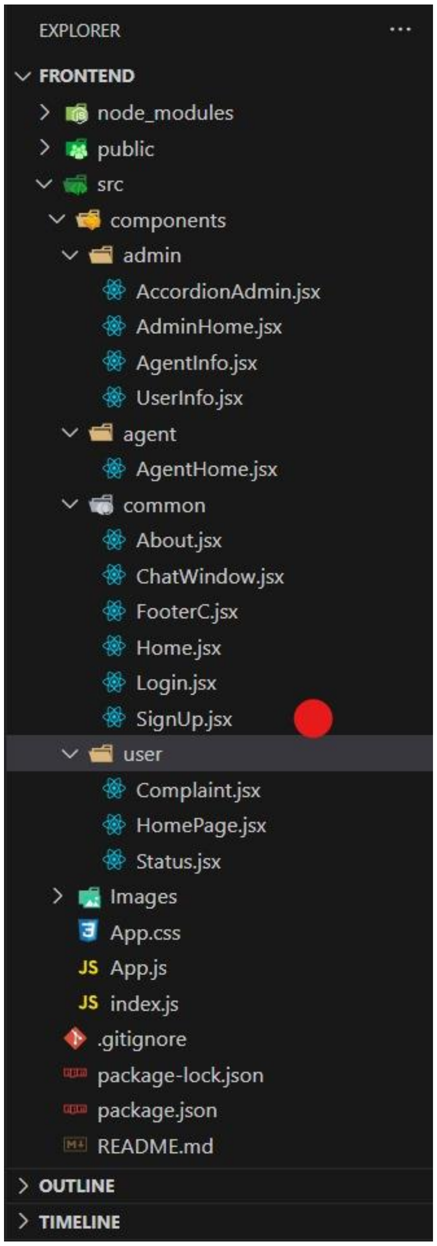
```
npm install
```

4. Set up environment variables:

- Create a .env file in the server directory and define variables such as MongoDB URI, JWT Key & Port number.

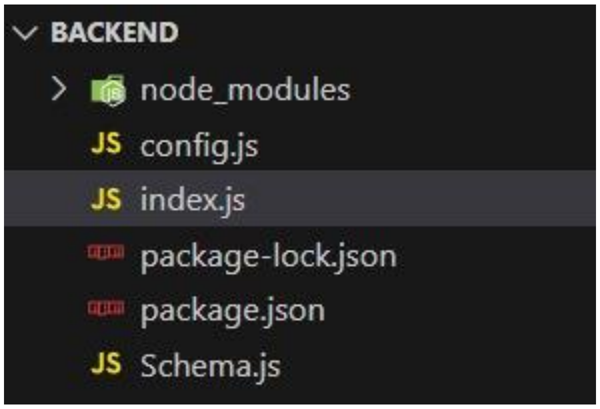
V. Folder Structure

• **Frontend:**



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

• **Backend:**



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

VI. Running the Application

• Frontend:

```
npm start
```

• Backend:

```
node index.js
```

• Access the app at: http://localhost:3000

VII. API Documentation

Base URL: All endpoints are prefixed with /api.

1. User Routes (/api/user)

a. Register User

Endpoint: POST /api/user/register

Request Body:

email (string, required)

password (string, required)

role (string, required; values: "User", "Admin", "Agent")

Example Request:

json

Copy code

```
{
  "email": "example@example.com",
  "password": "securepassword123",
  "role": "User"
}
```

Responses:

201: { "message": "Registered Successfully", "success": true }

400: { "message": "User already exists", "success": false }

500: { "message": "Error details", "success": false }

b. Login User

Endpoint: POST /api/user/login

Request Body:

email (string, required)

password (string, required)

Example Request:

json

Copy code

```
{
  "email": "example@example.com",
  "password": "securepassword123"
}
```

Responses:

200: { "message": "Login successful", "success": true, "token": "jwt_token_here", "user": { "email": "example@example.com", "role": "User" } }

404: { "message": "User not found", "success": false }

401: { "message": "Invalid email or password", "success": false }

500: { "message": "Error details", "success": false }

c. File Complaint

Endpoint: POST /api/user/complaint

Request Body:

userId (string, required)

name (string, required)

description (string, required)

attachments (array, optional)

Example Request:

json

Copy code

```
{
  "userId": "user123",
  "name": "John Doe",
  "description": "The product I purchased has a defect.",
  "attachments": ["image1.jpg"]
}
```

Responses:

201: { "message": "Complaint filed successfully", "success": true }

500: { "message": "Error filing complaint", "success": false }

d. Get User Complaints

Endpoint: GET /api/user/complaints

Query Parameters:

userId (string, required)

Example Request:

sql

Copy code

GET /api/user/complaints?userId=user123

Responses:

200: { "success": true, "data": [{ "complaintId": "comp123", "status": "Pending", "details": "Description of the issue" }] }

404: { "message": "No complaints found", "success": false }

500: { "message": "Error fetching complaints", "success": false }

2. Admin Routes (/api/admin)

a. Get All Complaints

Endpoint: GET /api/admin/complaints

Example Request:

GET /api/admin/complaints

Responses:

200: { "success": true, "data": [{ "complaintId": "comp123", "userId": "user123", "status": "Pending" }] }

404: { "message": "No complaints found", "success": false }

500: { "message": "Error retrieving complaints", "success": false }

b. Assign Complaint to Agent

Endpoint: PUT /api/admin/assign/:complaintId

Request Body:

agentId (string, required)

Example Request:

json

Copy code

{

  "agentId": "agent123"

}

Responses:

200: { "message": "Complaint assigned successfully", "success": true }

500: { "message": "Error assigning complaint", "success": false }

c. Get All Users

Endpoint: GET /api/admin/users

Example Request:

Copy code

GET /api/admin/users

Responses:

200: { "success": true, "data": [{ "userId": "user123", "email": "user@example.com" }] }

500: { "message": "Error fetching users", "success": false }

d. Handle Complaint Status

Endpoint: PUT /api/admin/status/:complaintId

Request Body:

status (string, required; e.g., "Resolved")

Example Request:

json

Copy code

{

  "status": "Resolved"

}

Responses:

200: { "message": "Status updated successfully", "success": true }

500: { "message": "Error updating status", "success": false }

3. Agent Routes (/api/agent)

a. Get Assigned Complaints

Endpoint: GET /api/agent/complaints/:agentId

Example Request:

Copy code

GET /api/agent/complaints/agent123

Responses:

200: { "success": true, "data": [{ "complaintId": "comp123", "status": "Pending" }] }

404: { "message": "No complaints assigned", "success": false }

500: { "message": "Error fetching complaints", "success": false }

b. Update Complaint Status

Endpoint: PUT /api/agent/status/:complaintId

Request Body:

status (string, required; e.g., "In Progress", "Resolved")

Example Request:

json

Copy code

```
{
  "status": "In Progress"
}
```

Responses:

200: { "message": "Status updated successfully", "success": true }

500: { "message": "Error updating status", "success": false }

XII. Authentication

Role-based authentication:

- Role-based authentication using JSON Web Tokens (JWT).
- Secure storage of tokens in HTTP-only cookies or local storage.
- Middleware in Express.js to validate tokens and protect routes.

XIII. User Interface

• Screenshots or Walkthroughs:

Admin Dashboard: Manage complaints, agents, and users.

Agent Dashboard: View assigned complaints and update statuses.

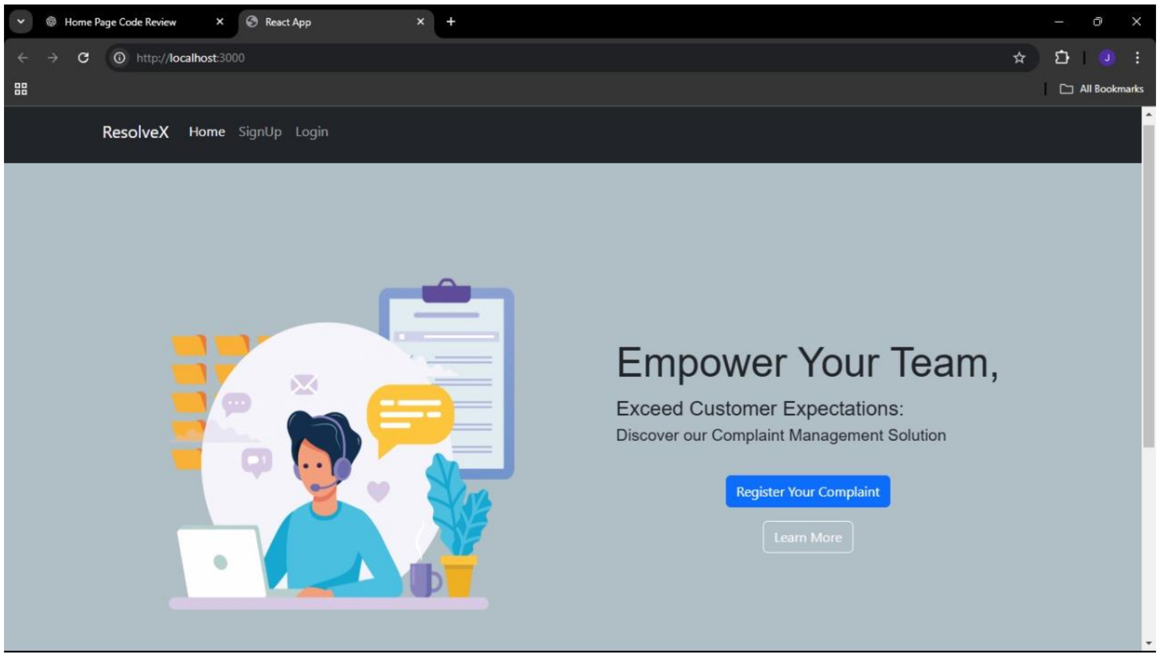
User Dashboard: Register complaints and track progress.

XIV. Testing

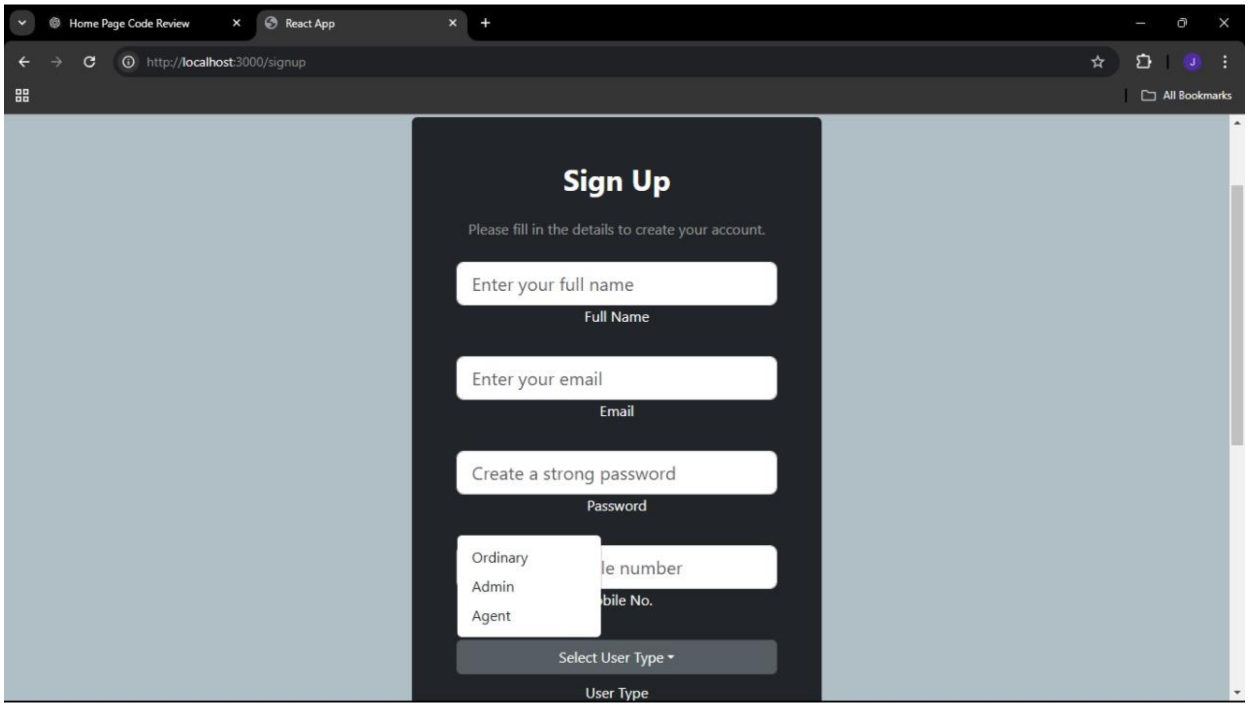
• Testing: Unit and integration tests are implemented using tools like Jest for the frontend and Mocha for the backend.

XV. Screenshots or Demo

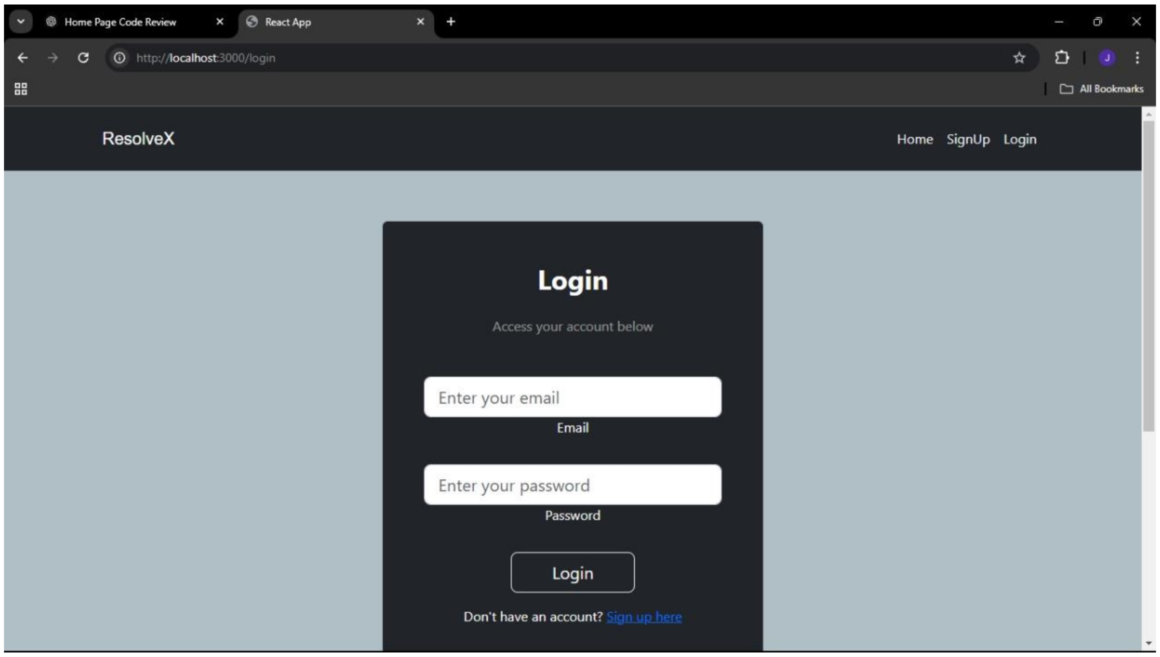
Home page:



User Sign-up page:



Login page:



Login

Access your account below

Enter your email

Email

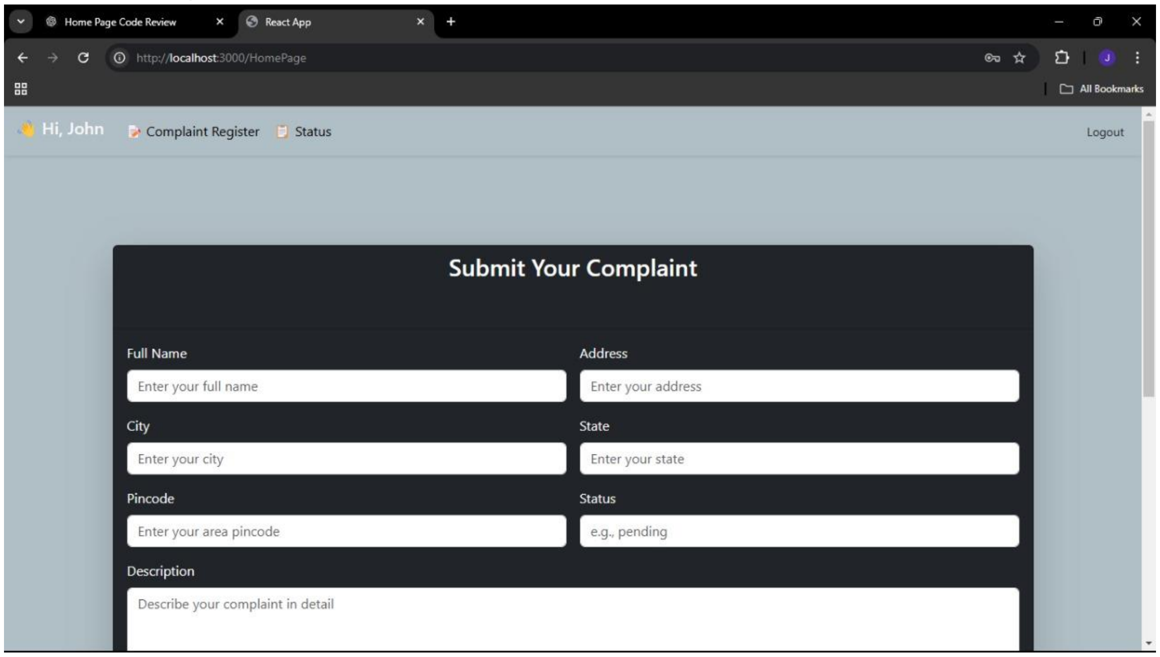
Enter your password

Password

Login

Don't have an account? Sign up here

User Main page:



Submit Your Complaint

Full Name: Enter your full name

Address: Enter your address

City: Enter your city

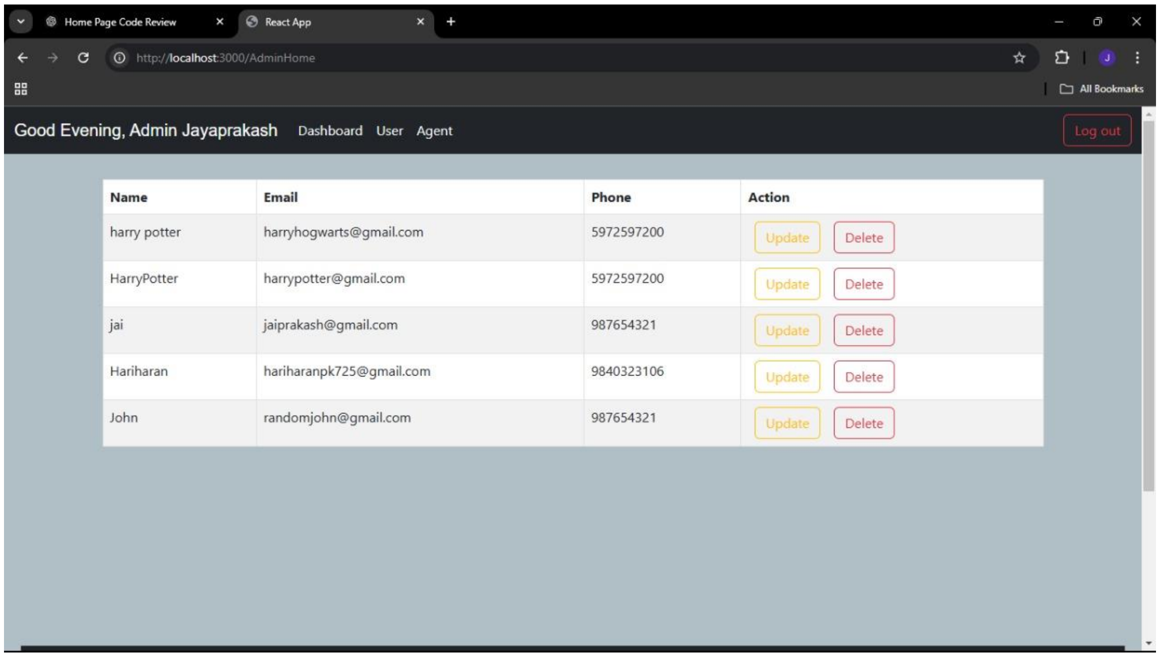
State: Enter your state

Pincode: Enter your area pincode

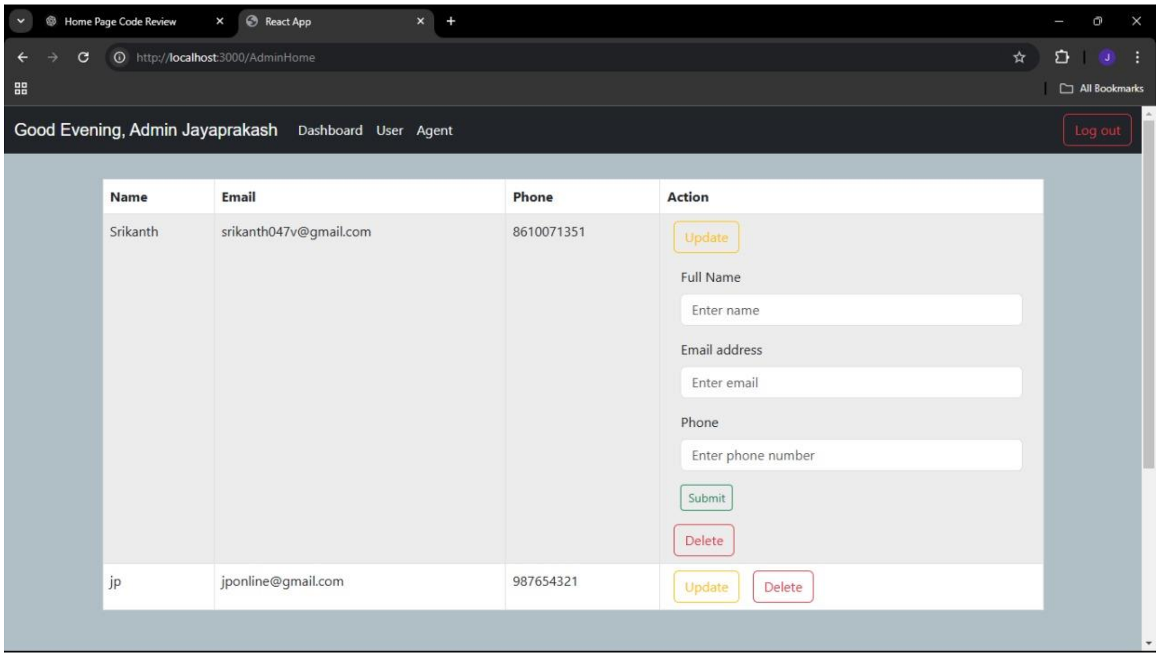
Status: e.g., pending

Description: Describe your complaint in detail

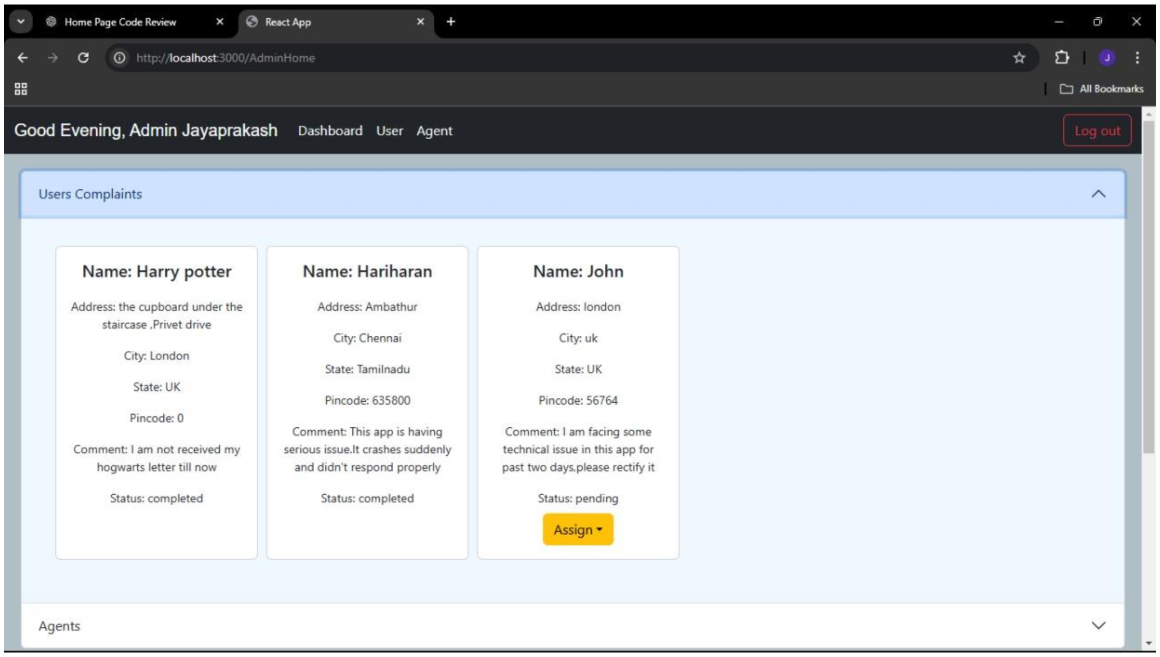
Admin pages:



Name	Email	Phone	Action
harry potter	harryhogwarts@gmail.com	5972597200	Update Delete
HarryPotter	harrypotter@gmail.com	5972597200	Update Delete
jai	jaiprakash@gmail.com	987654321	Update Delete
Hariharan	hariharanpk725@gmail.com	9840323106	Update Delete
John	randomjohn@gmail.com	987654321	Update Delete



Name	Email	Phone	Action
Srikanth	srikanth047v@gmail.com	8610071351	Update Full Name: Enter name Email address: Enter email Phone: Enter phone number Submit Delete
jp	jponline@gmail.com	987654321	Update Delete



Good Evening, Admin Jayaprakash

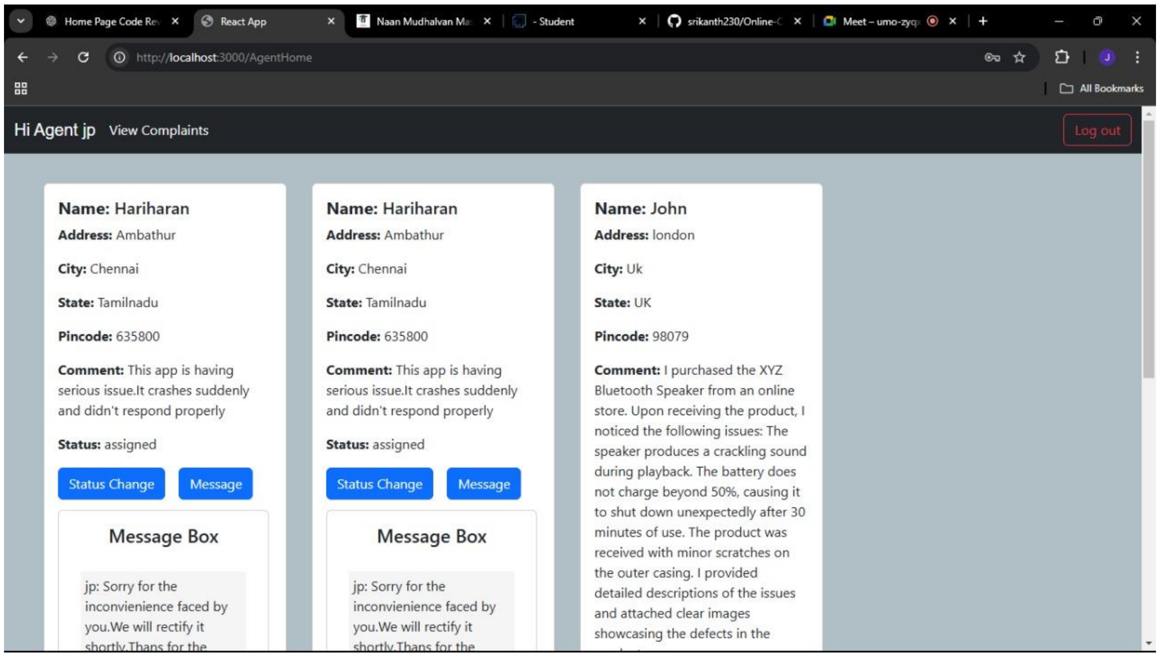
Users Complaints

Name: Harry potter
Address: the cupboard under the staircase ,Privet drive
City: London
State: UK
Pincode: 0
Comment: I am not received my hogwarts letter till now
Status: completed

Name: Hariharan
Address: Ambathur
City: Chennai
State: Tamilnadu
Pincode: 635800
Comment: This app is having serious issue.It crashes suddenly and didn't respond properly
Status: completed

Name: John
Address: london
City: uk
State: UK
Pincode: 56764
Comment: I am facing some technical issue in this app for past two days,please rectify it
Status: pending

Agent page:



Hi Agent jp

Name: Hariharan
Address: Ambathur
City: Chennai
State: Tamilnadu
Pincode: 635800
Comment: This app is having serious issue.It crashes suddenly and didn't respond properly
Status: assigned

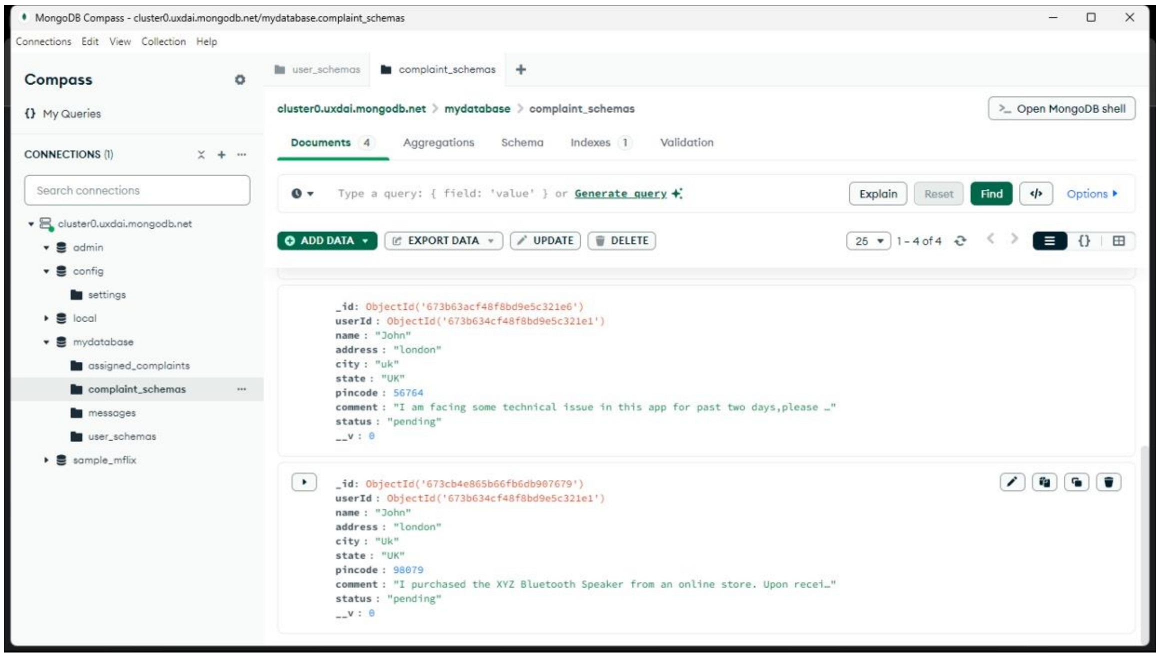
Message Box
jp: Sorry for the inconvienience faced by you.We will rectify it shortly.Thans for the

Name: Hariharan
Address: Ambathur
City: Chennai
State: Tamilnadu
Pincode: 635800
Comment: This app is having serious issue.It crashes suddenly and didn't respond properly
Status: assigned

Message Box
jp: Sorry for the inconvienience faced by you.We will rectify it shortly.Thans for the

Name: John
Address: london
City: Uk
State: UK
Pincode: 98079
Comment: I purchased the XYZ Bluetooth Speaker from an online store. Upon receiving the product, I noticed the following issues: The speaker produces a crackling sound during playback. The battery does not charge beyond 50%, causing it to shut down unexpectedly after 30 minutes of use. The product was received with minor scratches on the outer casing. I provided detailed descriptions of the issues and attached clear images showcasing the defects in the

Database:



```
_id: ObjectId('673b63acf48f8bd9e5c321e6')
userId : ObjectId('673b634cf48f8bd9e5c321e1')
name : "John"
address : "london"
city : "uk"
state : "UK"
pincode : 56764
comment : "I am facing some technical issue in this app for past two days,please …"
status : "pending"
__v : 0

_id: ObjectId('673cb4e865b66fb6db907679')
userId : ObjectId('673b634cf48f8bd9e5c321e1')
name : "John"
address : "london"
city : "Uk"
state : "UK"
pincode : 98079
comment : "I purchased the XYZ Bluetooth Speaker from an online store. Upon recei…"
status : "pending"
__v : 0
```

XVI. Known Issues

- Complaint Assignment Not Reflecting in Agent Dashboard
- Inconsistent Status Updates
- Error Handling Gaps
- Authentication Token Expiry Not Handled Gracefully
- Slow Complaint Loading Times for Admins

XVII. Future Enhancements

- ✓ Real-time notifications for users and agents via WebSockets.
- ✓ Enhanced analytics and reporting dashboards.
- ✓ Integration with third-party services like Twilio for SMS updates

XVIII. Video link:

Demo Video