**Full Stack Development with MERN**

**Project Documentation**

**1. Introduction**

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

**Team ID :** LTVIP2025TMID32193

**Team Size :** 4

**Team Leader :** G Rachitha

**Team member :** Datla Deepthi

**Team member :** Gummadi Reshmee Deepika

**Team member :** Pavan Kumar Gulla

**2. Project Overview**

**Purpose:**

To provide a centralized online platform where users can submit complaints, track progress, interact with agents, and receive timely resolutions, improving transparency and efficiency in complaint management.

**Features:**

\* User registration & authentication

\* Complaint submission with attachments

\* Real-time status tracking and notifications

\* Agent and admin dashboards

\* Chat interface between user and agent

\* Complaint assignment & routing

\* Security via authentication, authorization, and encryption

**3. Architecture**

**Frontend:**

Developed using React.js with Material UI and Bootstrap for a responsive user interface. Axios is used for API communication.

**Backend:**

Node.js with Express.js framework handles API routes, middleware, and server-side logic.

**Database:**

MongoDB is used for storing user profiles, complaints, chats, and assigned complaints. Mongoose ODM is used for schema management.

**4. Setup Instructions**

**Prerequisites:**

\* Node.js and npm

\* MongoDB Community Server

\* Git

\* Visual Studio Code

**Installation Steps:**

1. Clone the repository:

git clone https://github.com/ReshmeeDeepika/Complaint\_Registration.git

2. Navigate to the project folder:

cd Complaint\_Registration

3. Install dependencies:

cd frontend

npm install

cd ../backend

npm install

4. Set up environment variables (.env file) for database URI and JWT secrets.

**5. Folder Structure**

**Client (Frontend):**

\* `components/` - Reusable UI components

\* `pages/` - User, Admin, and Agent pages

\* `services/` - Axios service files for API calls

**Server (Backend):**

\* `routes/` - Express route handlers

\* `models/` - Mongoose schemas

\* `controllers/` - Business logic functions

\* `middleware/` - Auth and error-handling middlewares

**6. Running the Application**

**Frontend:**

cd frontend

npm start

**Backend:**

cd backend

npm start

Application runs on `http://localhost:3000`.

**7. API Documentation**

Base URL:`http://localhost:5000/api`

\* `POST /auth/register` - Register a new user

\* `POST /auth/login` - Login and receive JWT

\* `POST /complaints` - Submit a complaint

\* `GET /complaints/:userId` - Fetch user's complaints

\* `POST /assignments` - Assign complaint to agent

\* `GET /messages/:complaintId` - Fetch chat messages

\* `POST /messages` - Send chat message

**8. Authentication**

Method: JSON Web Tokens (JWT)

\* Login: Returns JWT stored in localStorage

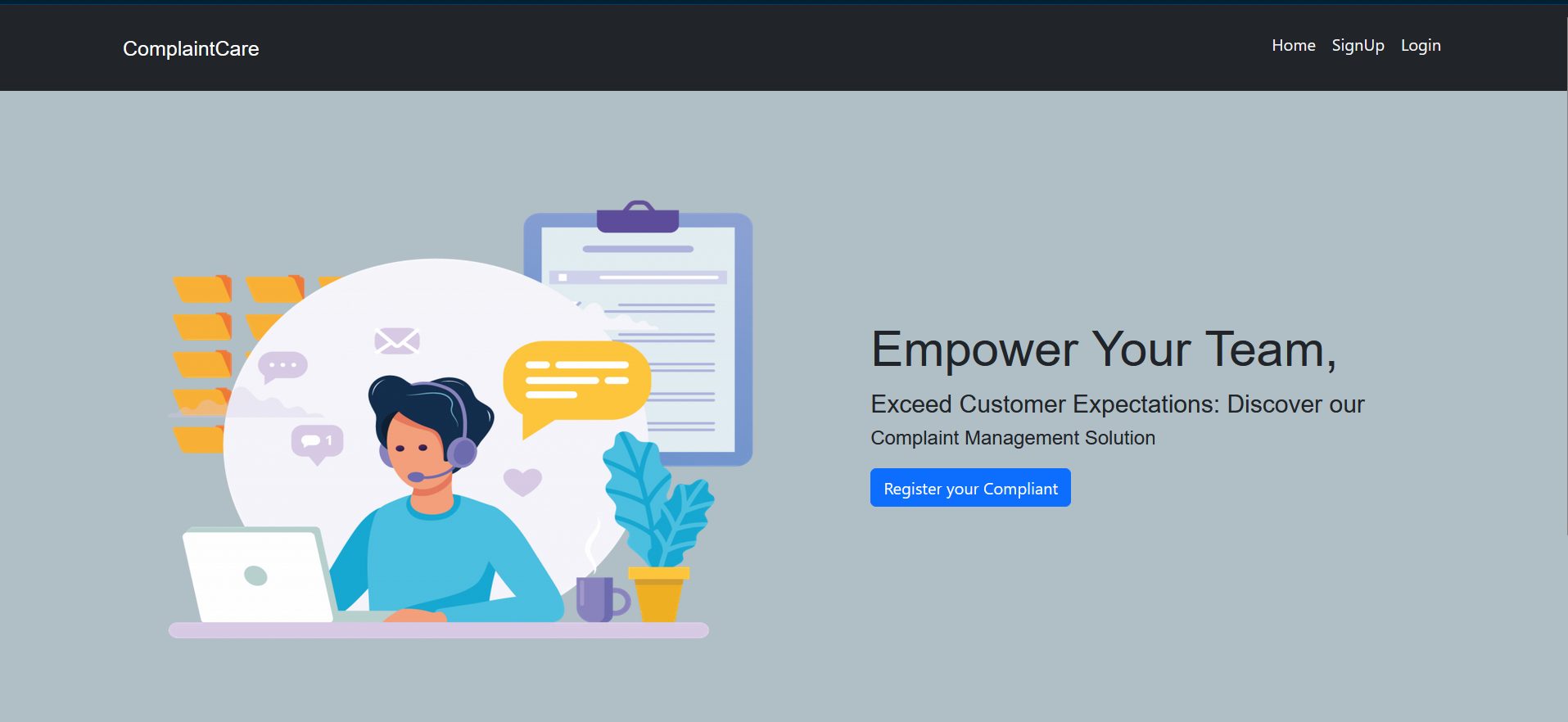
\* Protected Routes: Middleware verifies JWT

\* Roles: User, Agent, Admin

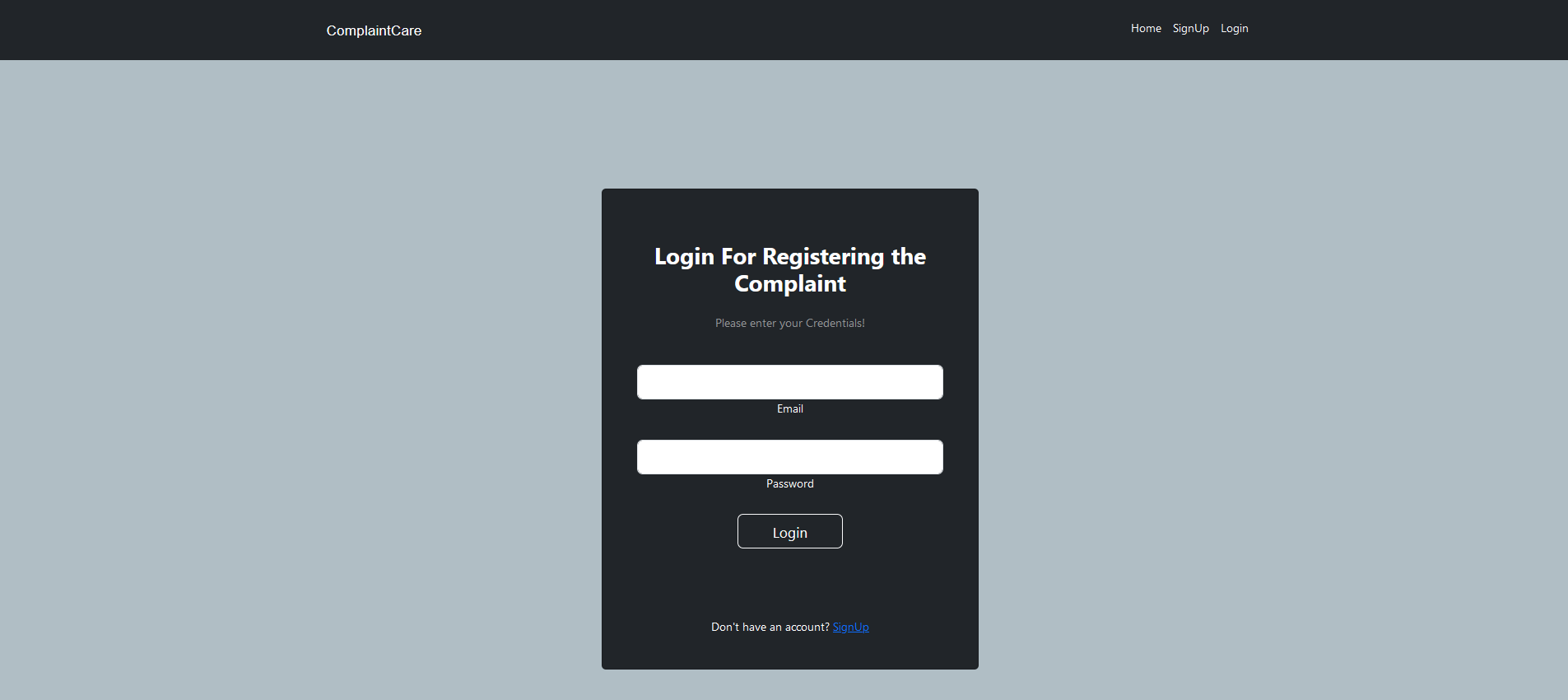
**9. User Interface**

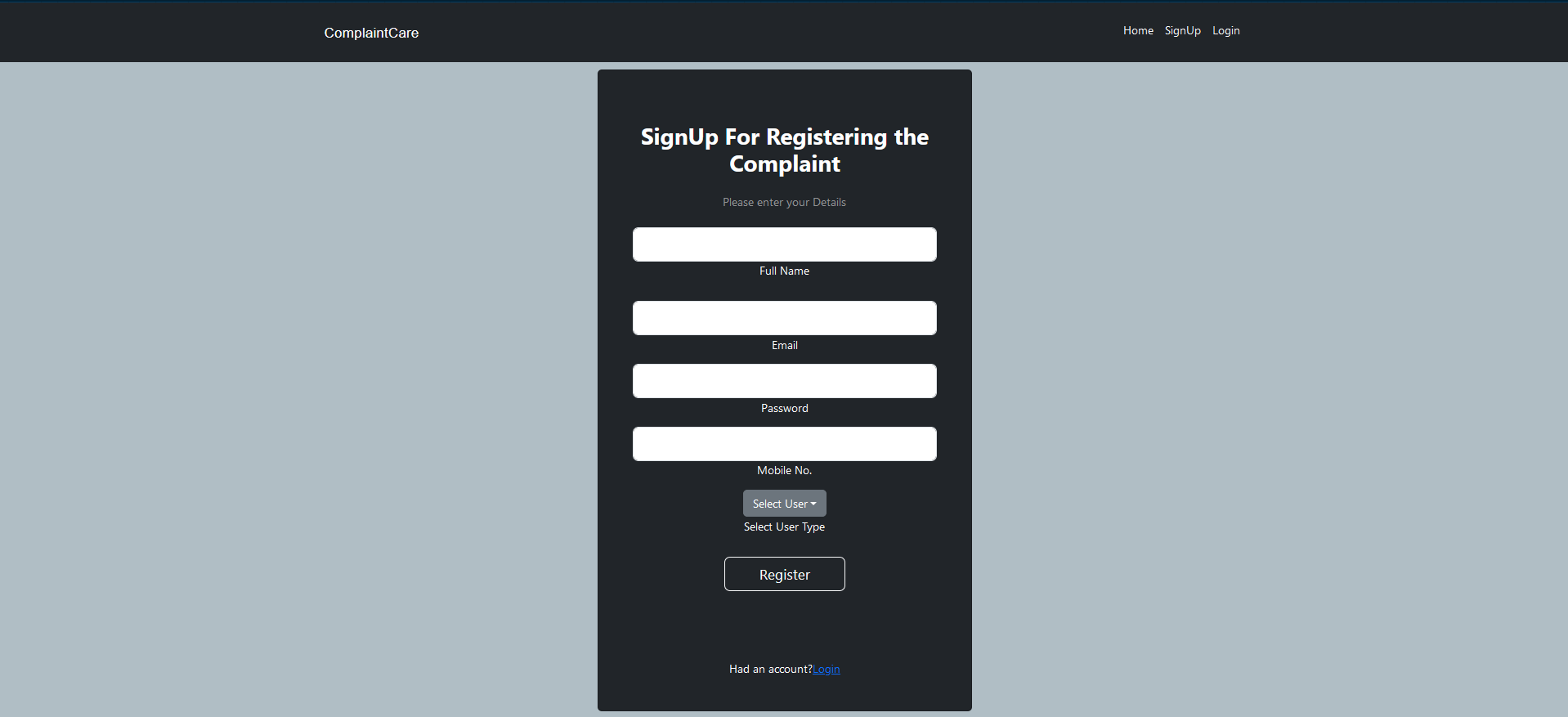
**Screenshots**:

\* Landing Page

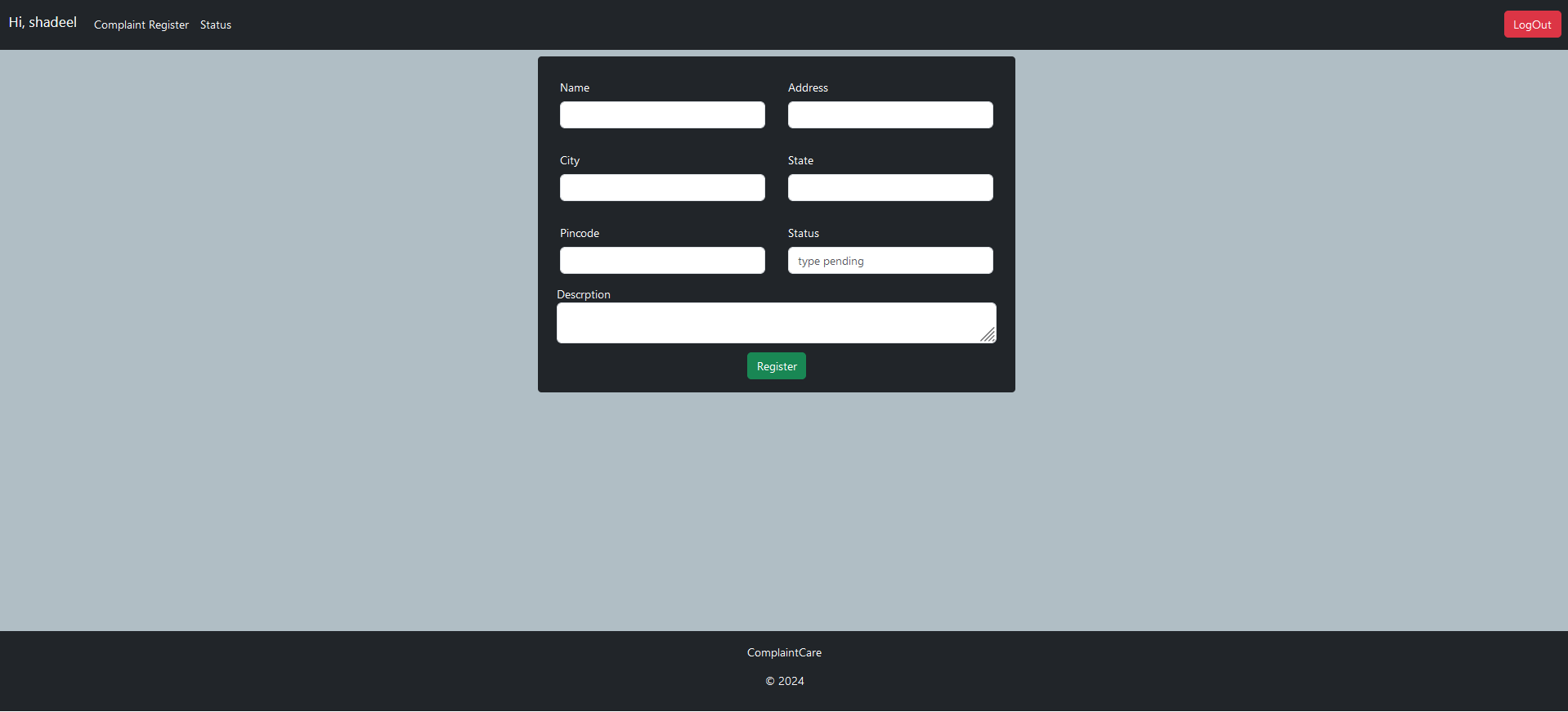


\* Login & Registration Pages

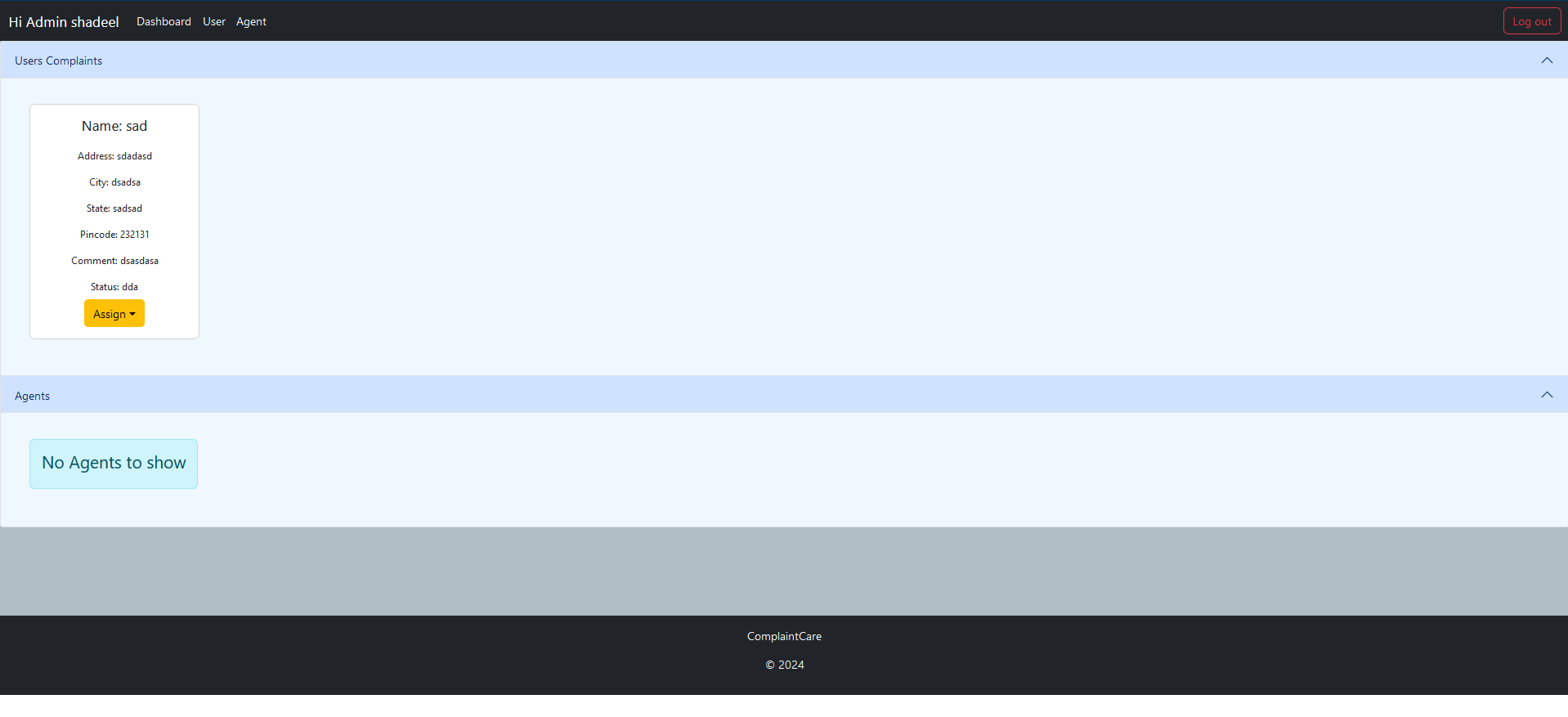




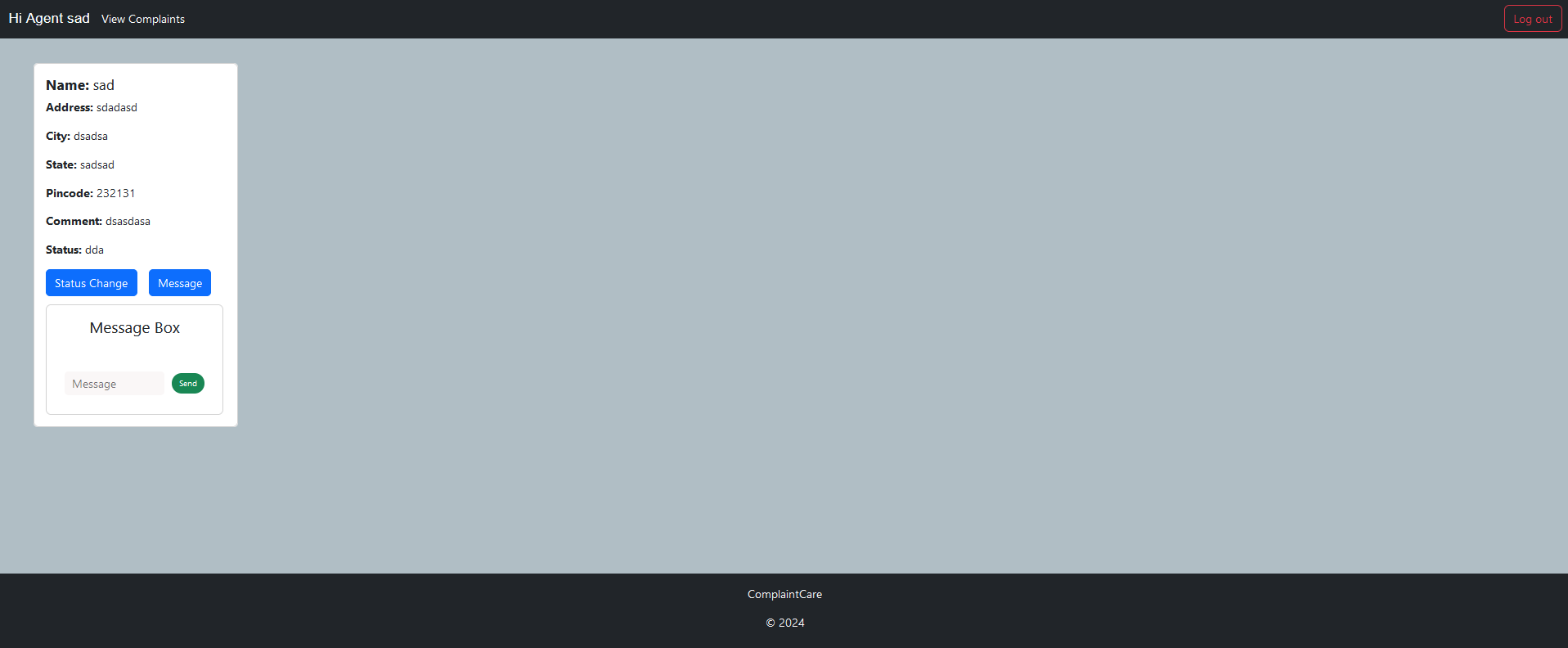
\* Complaint Dashboard



\* Admin Panel



\* Agent Dashboard



**10. Testing**

**Strategy**:

\* Manual testing of features

\* Postman for API endpoint validation

\* Console logging for debugging

**Tools:**

\* Postman, Browser DevTools, MongoDB Compass

**11. Demo**

\* GitHub Repo:

(https://github.com/ReshmeeDeepika/Complaint\_Registration.git)

**12. Known Issues**

\* Limited input validation on complaint forms

\* Lack of unit testing for backend logic

\* Chat window does not auto-scroll

**13. Future Enhancements**

\* Add file type validation for complaint attachments

\* Add pagination and filtering to admin dashboard

\* Enable push notifications

\* Integrate email service for communication

\* Enhance mobile responsiveness