COMPLAINT CARE

Complaint registration and management

1. Introduction

- **Project Title:** ResolveNow- Your Platform for Online Complaints
- **Team Members:**
  - *Chinthalapudi Karthik Sai*–Full Stack Developer
  - *Katta Rohi*– Database Administrator
  - *Kavuri Efefra*– Deployment/DevOps Engineer
  - *Katuri Tulasi Krishna* – Quality Analyst

2. Project Overview

- **Purpose:**

  The primary purpose of **ResolveNow** is to **streamline and digitize the complaint registration and resolution process** for users, organizations, and service providers. By providing a centralized, secure, and user-friendly platform, ResolveNow empowers individuals to submit complaints, track their status in real time, and interact directly with support agents—ensuring faster resolution, improved transparency, and enhanced customer satisfaction. The system also helps organizations manage and route complaints efficiently, reducing manual effort, response delays, and operational bottlenecks.

  Features:

  - User registration and secure login/logout
  - Role-based access (User, Agent, Admin)
  - Email verification on sign-up
  - Submit complaints with title, description, category, location, and attachments
  - Track complaint status (e.g., Submitted, In Progress, Resolved)
  - View complaint history in dashboard
  - Assign and reassign complaints to agents
  - Monitor complaint volume, resolution time, and agent activity
  - View analytics and generate system reports

Architecture:

Frontend:

- **React.js** – Component-based UI rendering
- **HTML5 & CSS3** – Structure and styling
- **Bootstrap & Material UI** – Responsive design and styled components

Backend:

- **Node.js** – JavaScript runtime environment for server-side code
- **Express.js** – Web application framework for handling routes and middleware

Database:

- **MongoDB** (NoSQLdatabase) with **Mongoose** for schema modeling

3. Setup Instructions

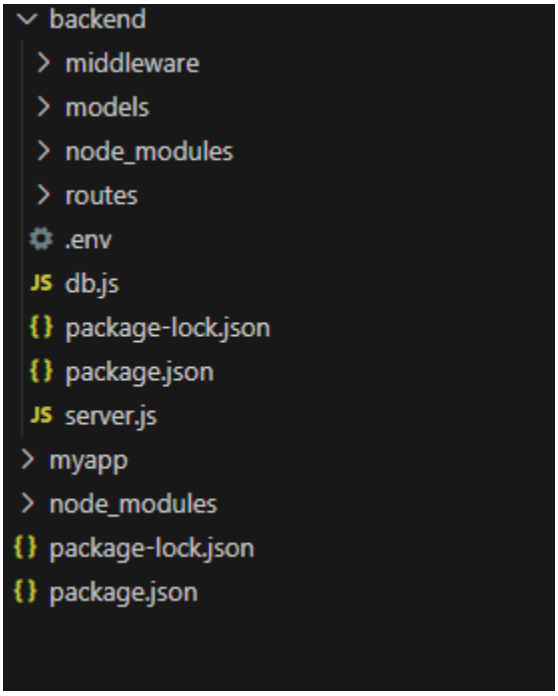
- **Prerequisites:**
  - Node.jsv18+
  - MongoDB
  - Git
  - npmoryarn
- **Installation:**

```
S:\>cd House Hund
The system cannot find the path specified.

S:\>cd House hunt

S:\House hunt> cd frontend

S:\House hunt\frontend> npm install

up to date, audited 1628 packages in 10s

252 packages are looking for funding
  run `npm fund` for details

34 vulnerabilities (4 low, 13 moderate, 16 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

S:\House hunt\frontend>npm start

> frontend@0.1.0 start
> react-scripts start
```

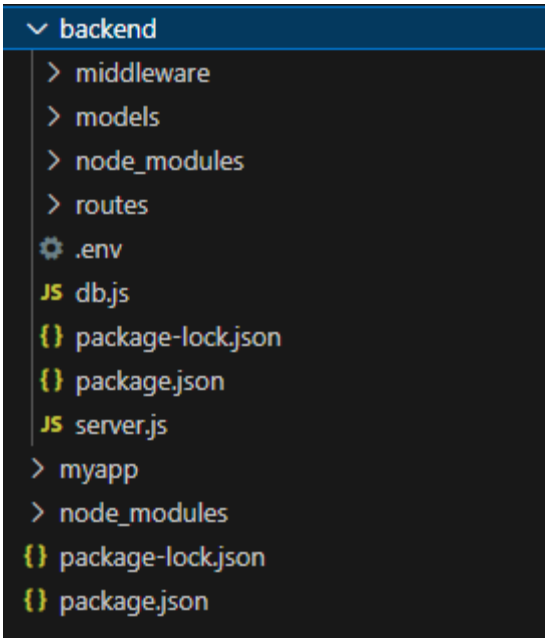
S

4. Folder Structure

- **Client(React):**



- **Server(Node.js):**



5. Running the Application

- **Frontend:**

```
cd frontend
npm start
```

- **Backend:**

```
cd backend
npm run dev
```

Sample Request:

POST/api/auth/register

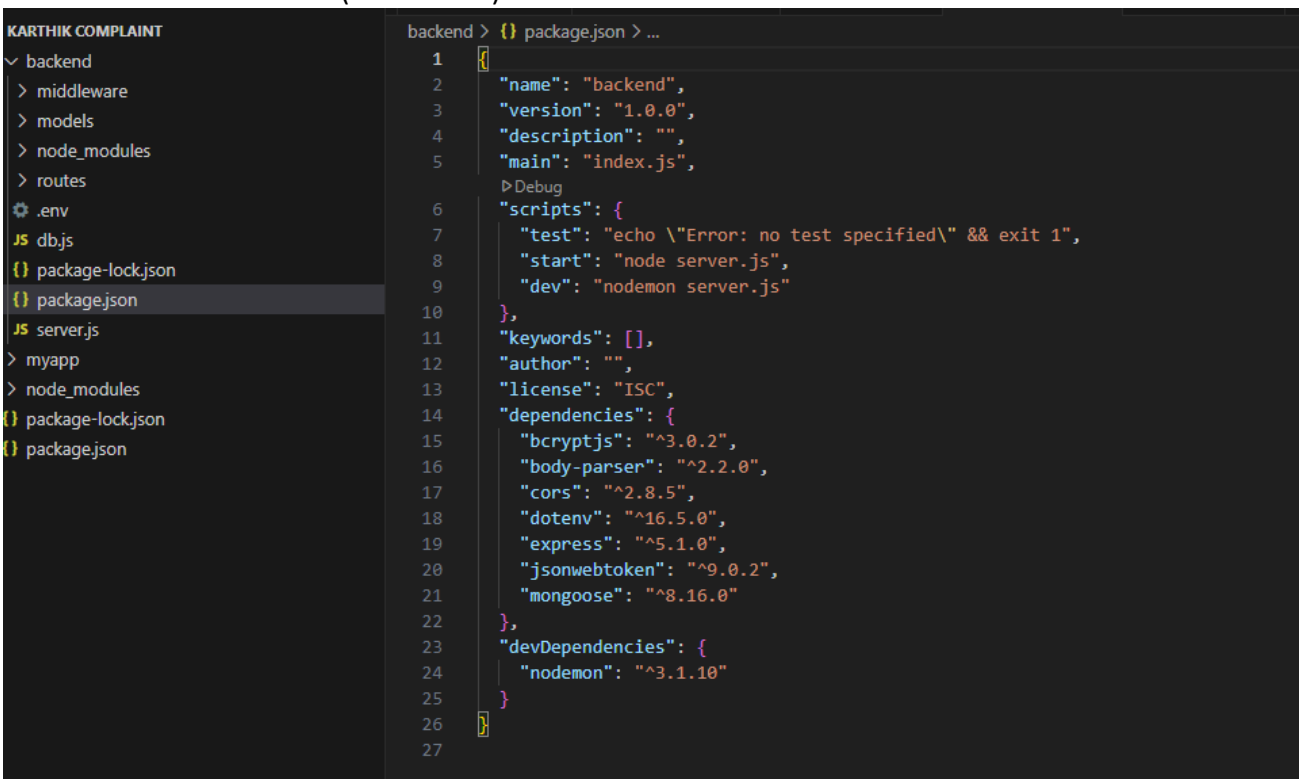
```
JS const User = require("../models/User"); Untitled-1 ●
1   const User = require("../models/User");
2   const bcrypt = require("bcryptjs");
3
4   exports.register = async (req, res) => {
5     const { name, email, password, type } = req.body;
6
7     try {
8       const existingUser = await User.findOne({ email });
9       if (existingUser)
10        return res.status(400).json({ message: "User already exists" });
11
12      const hashedPassword = await bcrypt.hash(password, 10);
13
14      const newUser = new User({
15        name,
16        email,
17        password: hashedPassword,
18        type
19      });
20
21      await newUser.save();
22      res.status(201).json({ message: "User registered successfully" });
23
24    } catch (err) {
25      res.status(500).json({ message: "Registration failed", error: err.message });
26    }
27  };
28
29
```

Sample Response:

```
{
  "message": "User registered successfully",
  "token": "JWT_TOKEN_HERE"
}
```

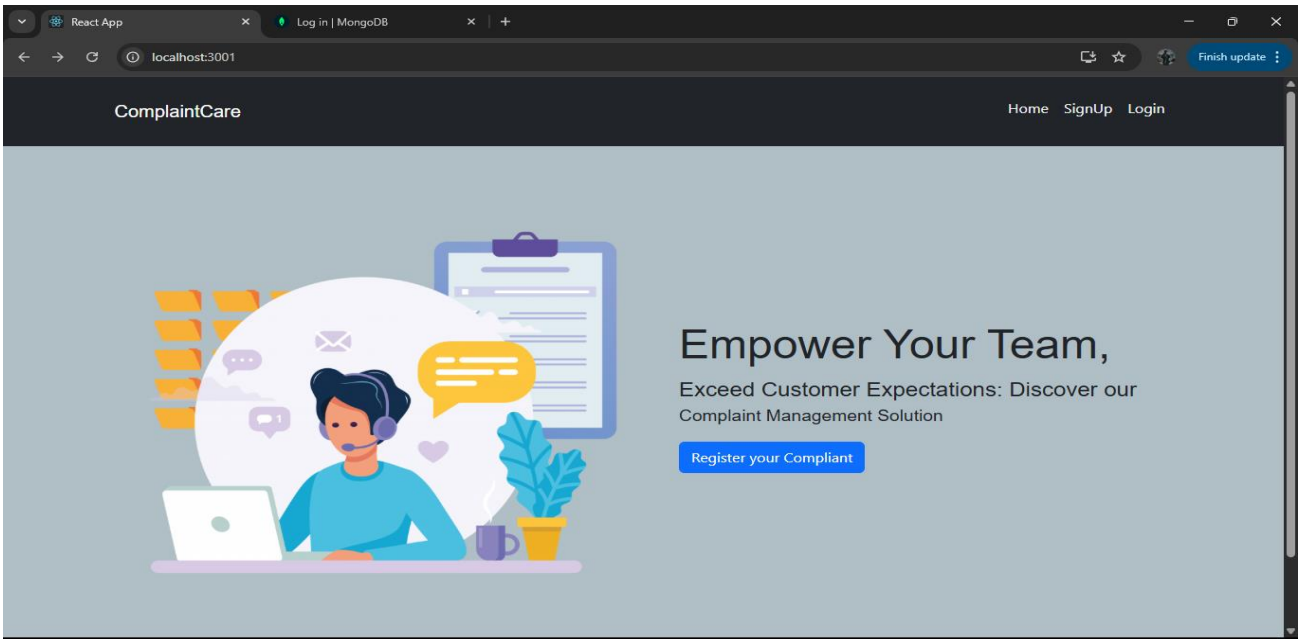
6. Authentication

- **Method Used :**(JSON Web)



- **Workflow:**

User logs in → receivesUser Interface



- My Complaint Dashboard - View current complaint status: `pending`, `approved`, `rejected`.

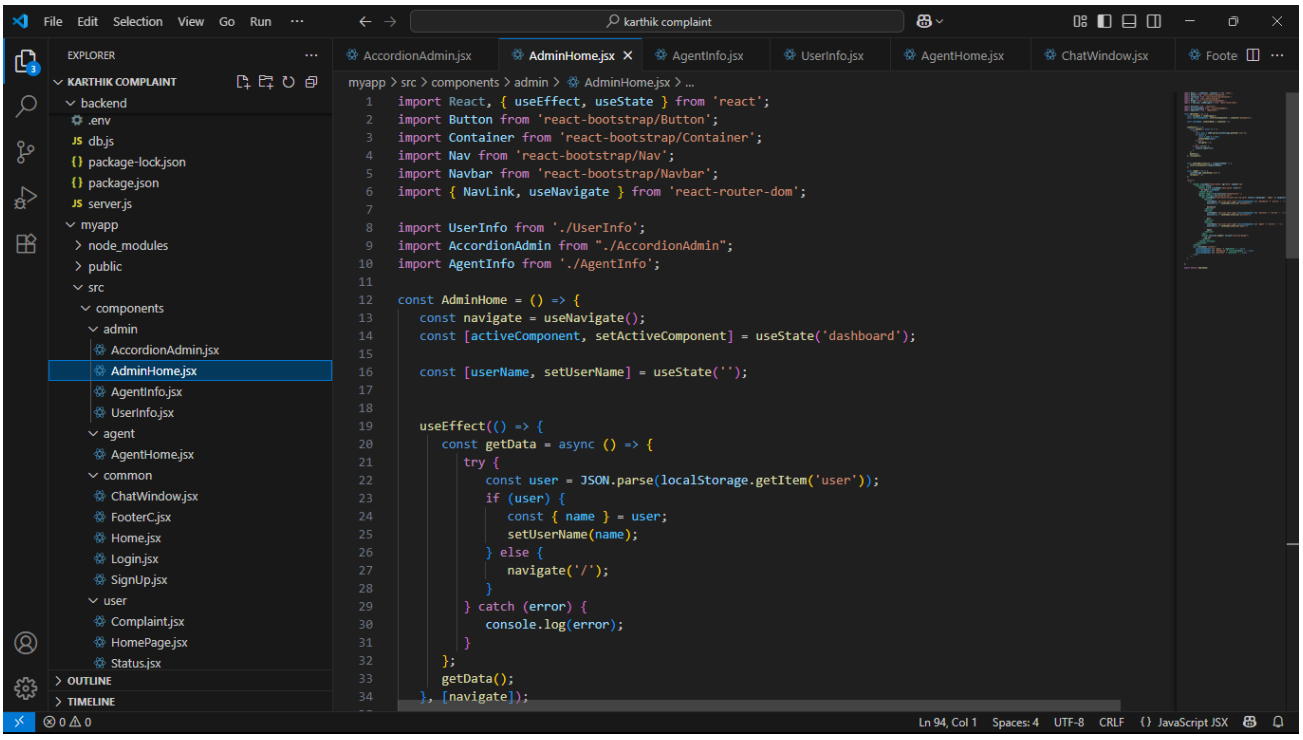
7. Testing

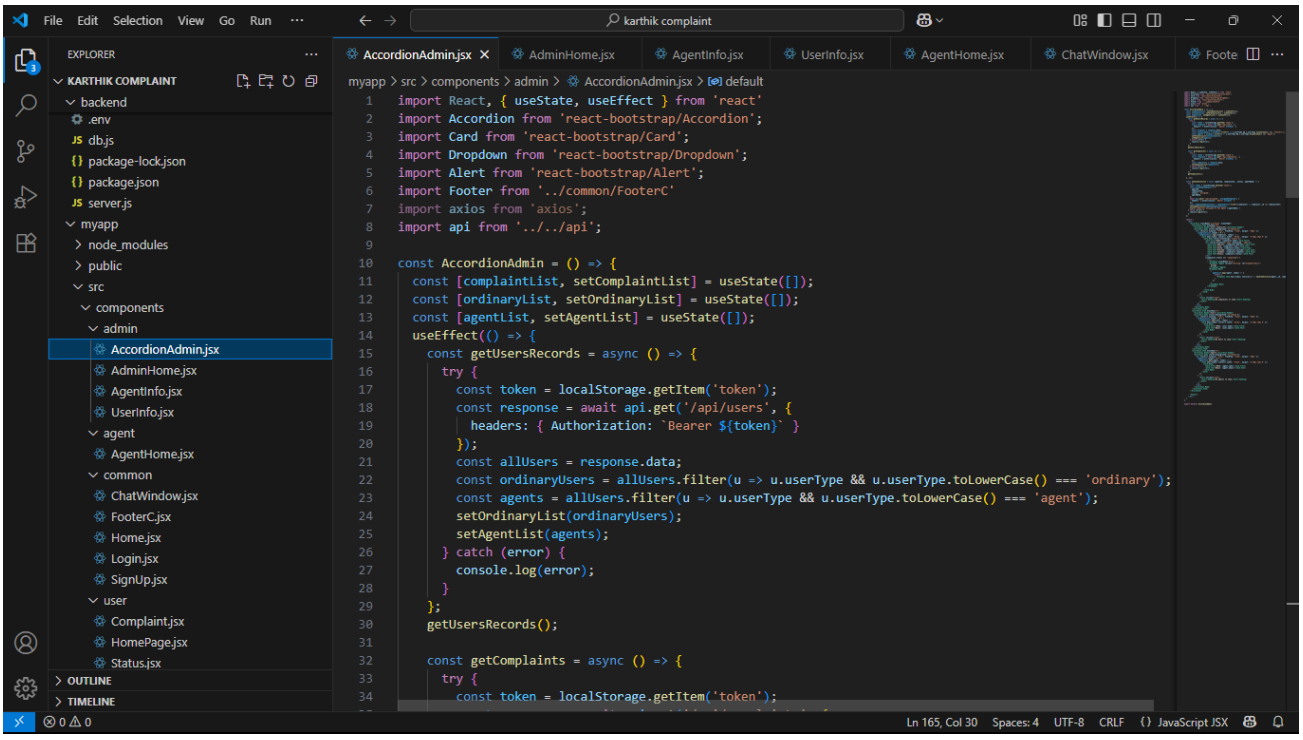
Tools Used: Postman , MongoDB Compass ,Console Logs & DevTools , JWT Debugger

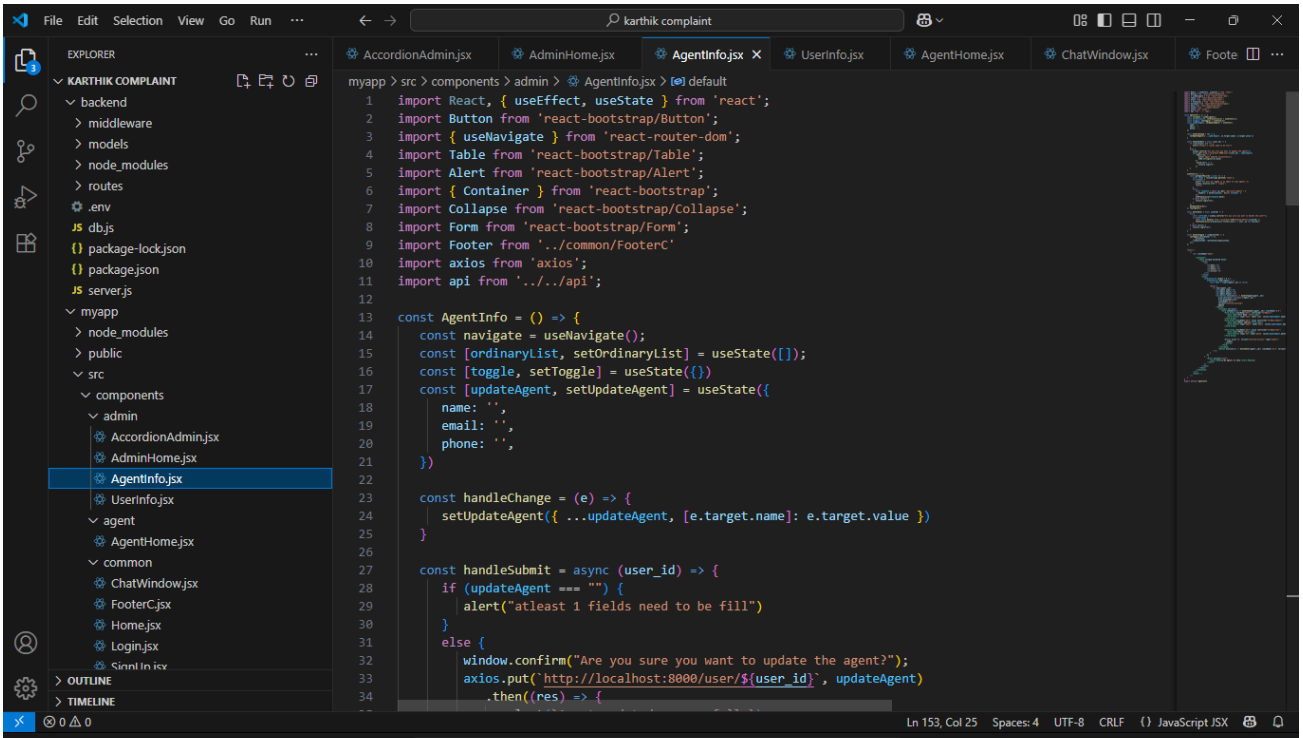
- **Strategy:**
    - Unit tests for backend logic
    - Integration tests for routes and middleware
    - Manual testing for frontend components
    - API testing with Postman

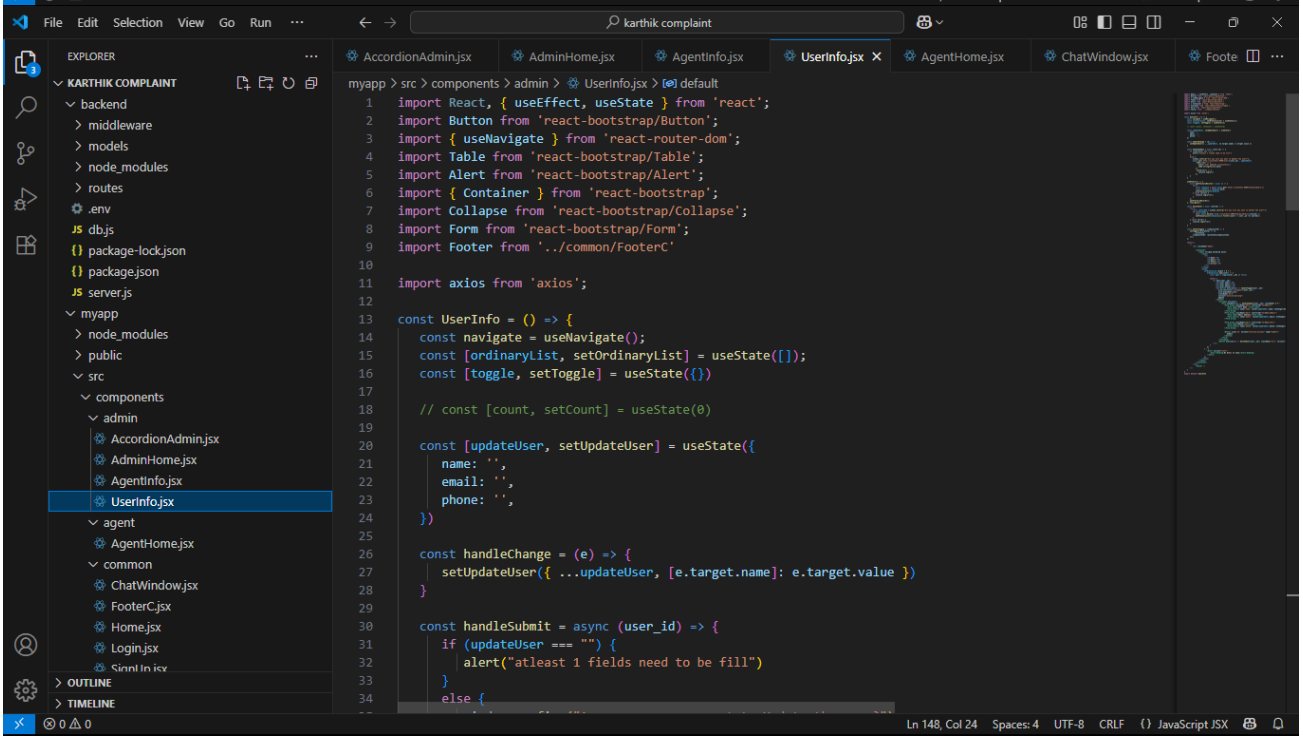
8. Screenshots or Demo

- **Demo:** Live Site
- **Screenshots:**


File Edit Selection View Go Run
karthik complaint
EXPLORER
KARTHIK COMPLAINT
backend
.env
db.js
package-lock.json
package.json
server.js
myapp
node_modules
public
src
components
admin
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
ChatWindow.jsx
FooterC.jsx
Home.jsx
Login.jsx
SignUp.jsx
user
Complaint.jsx
HomePage.jsx
Status.jsx
OUTLINE
TIMELINE
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
AgentHome.jsx
ChatWindow.jsx
Foote
myapp > src > components > admin > AdminHome.jsx > ...
import React, { useEffect, useState } from 'react';
import Button from 'react-bootstrap/Button';
import Container from 'react-bootstrap/Container';
import Nav from 'react-bootstrap/Nav';
import Navbar from 'react-bootstrap/Navbar';
import { NavLink, useNavigate } from 'react-router-dom';
import UserInfo from './UserInfo';
import AccordionAdmin from "./AccordionAdmin";
import AgentInfo from './AgentInfo';
const AdminHome = () => {
const navigate = useNavigate();
const [activeComponent, setActiveComponent] = useState('dashboard');
const [userName, setUserName] = useState('');
useEffect(() => {
const getData = async () => {
try {
const user = JSON.parse(localStorage.getItem('user'));
if (user) {
const { name } = user;
setUserName(name);
} else {
navigate('/');
}
} catch (error) {
console.log(error);
}
};
getData();
}, [navigate]);
Ln 94, Col 1 Spaces: 4 UTF-8 CRLF JavaScript JSX


File Edit Selection View Go Run
karthik complaint
EXPLORER
KARTHIK COMPLAINT
backend
.env
db.js
package-lock.json
package.json
server.js
myapp
node_modules
public
src
components
admin
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
ChatWindow.jsx
FooterC.jsx
Home.jsx
Login.jsx
SignUp.jsx
user
Complaint.jsx
HomePage.jsx
Status.jsx
OUTLINE
TIMELINE
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
AgentHome.jsx
ChatWindow.jsx
Foote
myapp > src > components > admin > AccordionAdmin.jsx > default
import React, { useState, useEffect } from 'react'
import Accordion from 'react-bootstrap/Accordion';
import Card from 'react-bootstrap/Card';
import Dropdown from 'react-bootstrap/Dropdown';
import Alert from 'react-bootstrap/Alert';
import Footer from '../common/FooterC'
import axios from 'axios';
import api from '../../api';
const AccordionAdmin = () => {
const [complaintList, setComplaintList] = useState([]);
const [ordinaryList, setOrdinaryList] = useState([]);
const [agentList, setAgentList] = useState([]);
useEffect(() => {
const getUsersRecords = async () => {
try {
const token = localStorage.getItem('token');
const response = await api.get('/api/users', {
headers: { Authorization: `Bearer ${token}` }
});
const allUsers = response.data;
const ordinaryUsers = allUsers.filter(u => u.userType && u.userType.toLowerCase() === 'ordinary');
const agents = allUsers.filter(u => u.userType && u.userType.toLowerCase() === 'agent');
setOrdinaryList(ordinaryUsers);
setAgentList(agents);
} catch (error) {
console.log(error);
}
};
getUsersRecords();
const getComplaints = async () => {
try {
const token = localStorage.getItem('token');
Ln 165, Col 30 Spaces: 4 UTF-8 CRLF JavaScript JSX


File Edit Selection View Go Run
karthik complaint
EXPLORER
KARTHIK COMPLAINT
backend
middleware
models
node_modules
routes
.env
db.js
package-lock.json
package.json
server.js
myapp
node_modules
public
src
components
admin
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
ChatWindow.jsx
FooterC.jsx
Home.jsx
Login.jsx
SignUp.jsx
OUTLINE
TIMELINE
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
AgentHome.jsx
ChatWindow.jsx
Foote
myapp > src > components > admin > AgentInfo.jsx > default
import React, { useEffect, useState } from 'react';
import Button from 'react-bootstrap/Button';
import { useNavigate } from 'react-router-dom';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container } from 'react-bootstrap';
import Collapse from 'react-bootstrap/Collapse';
import Form from 'react-bootstrap/Form';
import Footer from '../common/FooterC'
import axios from 'axios';
import api from '../../api';
const AgentInfo = () => {
const navigate = useNavigate();
const [ordinaryList, setOrdinaryList] = useState([]);
const [toggle, setToggle] = useState({})
const [updateAgent, setUpdateAgent] = useState({
name: '',
email: '',
phone: '',
})
const handleChange = (e) => {
setUpdateAgent({ ...updateAgent, [e.target.name]: e.target.value })
}
const handleSubmit = async (user_id) => {
if (updateAgent === "") {
alert("atleast 1 fields need to be fill")
}
else {
window.confirm("Are you sure you want to update the agent?");
axios.put(`http://localhost:8000/user/${user_id}`, updateAgent)
.then((res) => {
Ln 153, Col 25 Spaces: 4 UTF-8 CRLF JavaScript JSX


File Edit Selection View Go Run
karthik complaint
EXPLORER
KARTHIK COMPLAINT
backend
middleware
models
node_modules
routes
.env
db.js
package-lock.json
package.json
server.js
myapp
node_modules
public
src
components
admin
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
agent
AgentHome.jsx
common
ChatWindow.jsx
FooterC.jsx
Home.jsx
Login.jsx
SignUp.jsx
OUTLINE
TIMELINE
AccordionAdmin.jsx
AdminHome.jsx
AgentInfo.jsx
UserInfo.jsx
AgentHome.jsx
ChatWindow.jsx
Foote
myapp > src > components > admin > UserInfo.jsx > default
import React, { useEffect, useState } from 'react';
import Button from 'react-bootstrap/Button';
import { useNavigate } from 'react-router-dom';
import Table from 'react-bootstrap/Table';
import Alert from 'react-bootstrap/Alert';
import { Container } from 'react-bootstrap';
import Collapse from 'react-bootstrap/Collapse';
import Form from 'react-bootstrap/Form';
import Footer from '../common/FooterC'
import axios from 'axios';
const UserInfo = () => {
const navigate = useNavigate();
const [ordinaryList, setOrdinaryList] = useState([]);
const [toggle, setToggle] = useState({})
// const [count, setCount] = useState(0)
const [updateUser, setUpdateUser] = useState({
name: '',
email: '',
phone: '',
})
const handleChange = (e) => {
setUpdateUser({ ...updateUser, [e.target.name]: e.target.value })
}
const handleSubmit = async (user_id) => {
if (updateUser === "") {
alert("atleast 1 fields need to be fill")
}
else {
Ln 148, Col 24 Spaces: 4 UTF-8 CRLF JavaScript JSX

9. Future Enhancements

- Create more dynamic dashboards tailored to the specific needs of users, agents, and admins (e.g., analytics for admin, task queues for agents).

- Add graphs and charts for:
    - Complaint resolution time
    - Agent performance metrics
    - Monthly/weekly complaint volume
- Exportable PDF/CSV reports for audits and reviews.

- Develop a cross-platform mobile app using **React Native** or **Flutter** for on-the-go access to complaint tracking and resolution. 4. **Image and Video Upload Support**

- Allow users to attach multiple images or videos to better demonstrate their issue.
- Use **cloud storage** (e.g., Cloudinary or AWS S3) for file management.

- Integrate real-time **voice/video calling** between user and agent using **WebRTC** for more complex issue resolution.

- Implement **Natural Language Processing (NLP)** to auto-detect complaint categories and suggest resolutions or routes based on complaint content.

- Enable localization and translation of the UI to support regional languages for broader accessibility.

- Add extra security during login using OTP or app-based 2FA systems (Google Authenticator, Authy).