Online Hostel Complaint Management System

Name : Gourav R. Patil | Roll No. : 205223022 | M.Tech. Data Analytics

Introduction

The Online Hostel Complaint Management System serves as a comprehensive platform designed to streamline the process of handling and resolving hostel-related concerns within our college community. In response to the growing need for an efficient and user-friendly system, this project aims to provide students, hostel staff, and administrators with a centralized platform to manage and address various hostel-related matters.

Objectives

1. **Efficient Complaint Resolution:**

   Create a seamless process for students to submit and track their hostel-related complaints.

   Enable hostel staff and administrators to promptly address and resolve reported issues.

2. **User-Friendly Interface:**

   Develop an intuitive and visually appealing user interface accessible to users of varying technical expertise.

3. **Role-Based Access:**

   Implement a secure user authentication system with role-based access control, distinguishing between students, hostel staff, and administrators.

Technologies Used

1. HTML, CSS, Bootstrap:

For the front-end development, we utilized HTML for structuring the web pages, CSS for styling and layout, and Bootstrap to expedite the development process while ensuring a responsive and consistent user interface across devices.

2. Flask (Backend Framework):

The backend of our system is powered by Flask, a Python web framework. Flask enables us to handle server-side operations, manage routing, and interact with the database. Its simplicity and flexibility make it an ideal choice for our application.

3. SQLite (User Authentication):

For user authentication, we opted for SQLite, a lightweight relational database. SQLite provides a secure and efficient way to manage user credentials, ensuring a reliable authentication process for our system.

4. CouchDB (Content Storage):

To manage and store content related to the website, we implemented CouchDB as our NoSQL database. CouchDB's ability to handle semi-structured data and scalability makes it a suitable choice for our dynamic content storage needs.

User Roles

1. Student Role:

As users of the system, students have the ability to submit complaints regarding hostel-related issues. They can track the status of their complaints, view updates, and provide feedback on the resolution process. The student role is focused on ensuring a straightforward and responsive experience for those lodging complaints.

2. Hostel Staff Role:

Hostel staff play a crucial role in managing and resolving complaints. They have the ability to view all lodged complaints, update their statuses, and communicate with students to gather additional information. The hostel staff role is geared towards efficient complaint resolution and effective communication with the student body.

3. Hostel Admin Role:

The hostel admin has a comprehensive role in overseeing the entire complaint management system. They are responsible for user management, ensuring the system's proper functioning, and configuring system settings. The hostel admin has the highest level of access and is pivotal in maintaining the integrity and security of the entire system.

4. Role-Based Access Control:

Role-based access control is implemented to ensure data security and privacy. Access permissions are granted based on user roles, restricting users to actions and information relevant to their responsibilities. This approach enhances system security and maintains a clear separation of duties.

5. User Interface for Each Role:

The user interface is designed with role-specific features to cater to the unique needs of each user type. Students have an intuitive dashboard for submitting and tracking complaints, while hostel staff have additional tools for managing and resolving complaints. The hostel admin interface provides a comprehensive view of the entire system.

Database Design

1. Introduction:

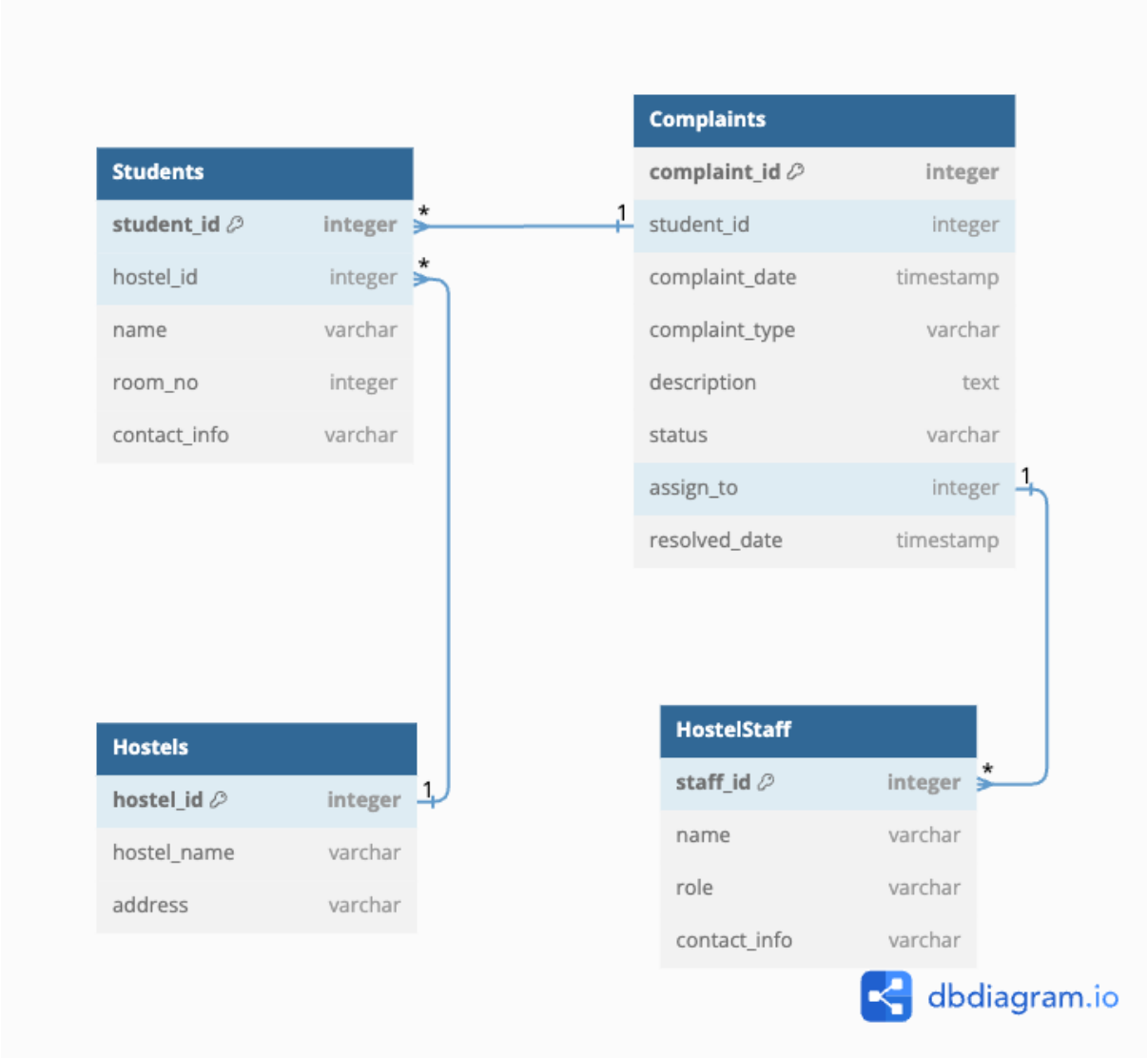
Database design is a critical component of our online hostel complaint management system, influencing how data is stored, organized, and retrieved. We employ two databases, SQLite for user authentication and CouchDB for content storage, to ensure a robust and scalable system.

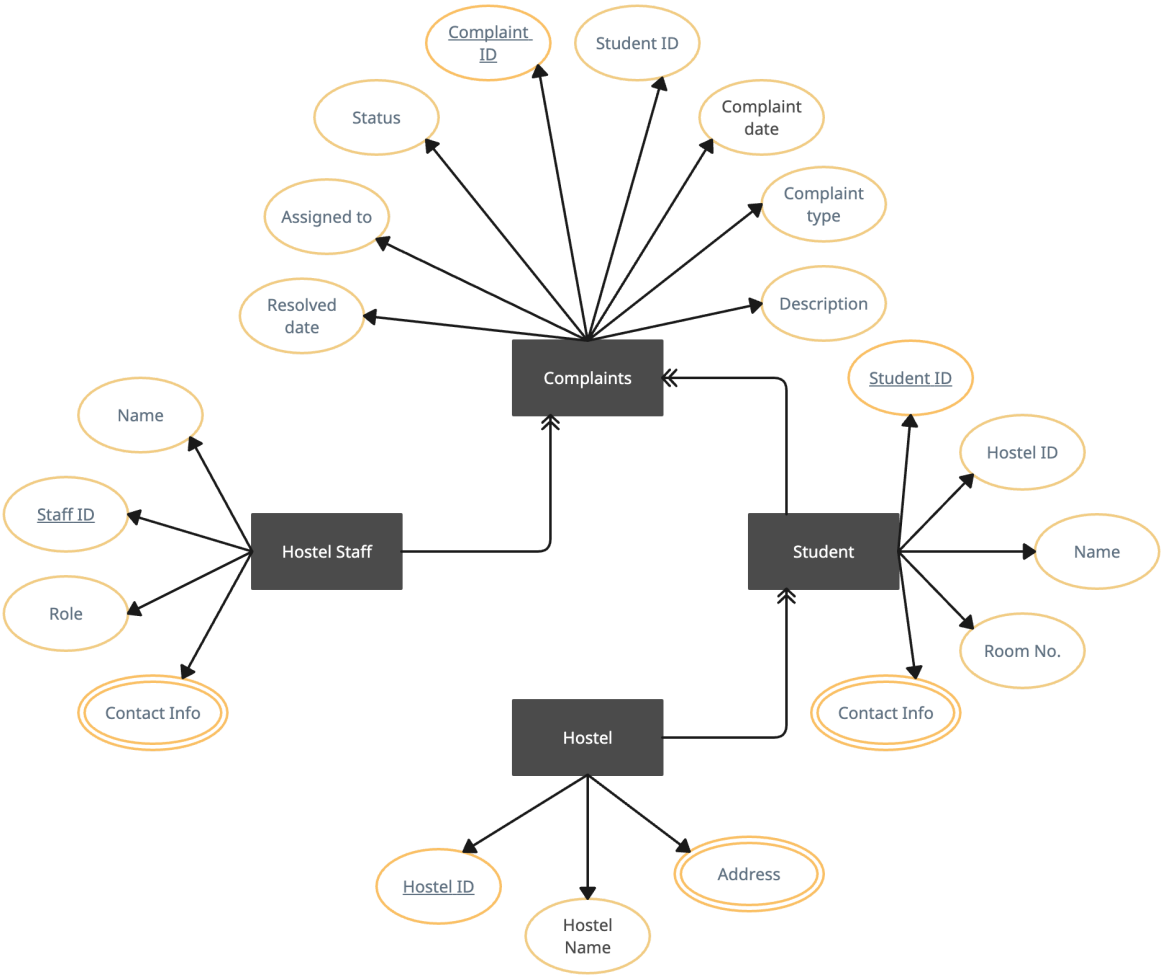
2. SQLite Database Design (User Authentication):

In the SQLite database, we have a 'users' table that stores essential information such as usernames, hashed passwords, and role assignments. This table is central to user authentication, providing a secure and efficient means of verifying user identities during the login process.

3. CouchDB Database Design (Content Storage):

CouchDB, being a NoSQL database, stores data in a document-oriented format. For content related to the website, we use CouchDB to store documents representing complaints, feedback, and other dynamic data. This design choice allows for flexibility in handling semi-structured data and supports the dynamic nature of the content.





View for Stakeholders

1. Views for Student User

a. View to Retrieve Student Information by Username:

Purpose: Retrieve detailed information about a student based on their username.

Example Map Function:

```
function (doc) {
    if (doc.type === 'student' && doc.username) {
        emit(doc.username, doc);
    }
}
```

b. **View to Retrieve Student Complaints:**

Purpose: Retrieve a list of complaints submitted by a specific student.

Example Map Function:

```
function (doc) {
     if (doc.type === 'complaint' && doc.student_username) {
          emit(doc.student_username, doc);
     }
}
```

c. **View to Retrieve Student's Hostel Information:**

Purpose: Retrieve information about the hostel to which a student belongs.

Example Map Function:

```
function (doc) {
     if (doc.type === 'student' && doc.hostel_id) {
          emit(doc.hostel_id, doc);
     }
}
```

2. Views for Hostel Staff User

a. **View to Retrieve Hostel Staff Information by Username:**

Purpose: Retrieve detailed information about a hostel staff member based on their username.

Example Map Function:

```
function (doc) {
     if (doc.type === 'hostel_staff' && doc.username) {
          emit(doc.username, doc);
     }
}
```

b. View to Retrieve Complaints Assigned to Hostel Staff:

Purpose: Retrieve a list of complaints assigned to a specific hostel staff member.

Example Map Function:

```
function (doc) {
     if (doc.type === 'complaint' &&
doc.assigned_staff_username){
          emit(doc.assigned_staff_username, doc);
     }
}
```

c. View to Retrieve Hostel Information for Hostel Staff:

Purpose: Retrieve information about the hostel for which a staff member is responsible.

Example Map Function:

```
function (doc) {
     if (doc.type === 'hostel_staff' && doc.hostel_id) {
          emit(doc.hostel_id, doc);
     }
}
```

3. Views for Administrator User

a. View to Retrieve All User Information:

Purpose: Retrieve detailed information about all users in the system.

Example Map Function:

```
function (doc) {
     if (doc.type === 'user') {
          emit(doc.username, doc);
     }
}
```

b. **View to Retrieve All Complaints in the System:**

Purpose: Retrieve a list of all complaints submitted by students.

Example Map Function:

```
function (doc) {
    if (doc.type === 'complaint') {
        emit(doc._id, doc);
    }
}
```

c. **View to Retrieve Hostel Staff Information:**

Purpose: Retrieve detailed information about all hostel staff members.

Example Map Function:

```
function (doc) {
    if (doc.type === 'hostel_staff') {
        emit(doc.username, doc);
    }
}
```

d. **View to Monitor System Activity and Updates:**

Purpose: Retrieve a log of system activities, such as user registrations, complaint submissions, and feedback.

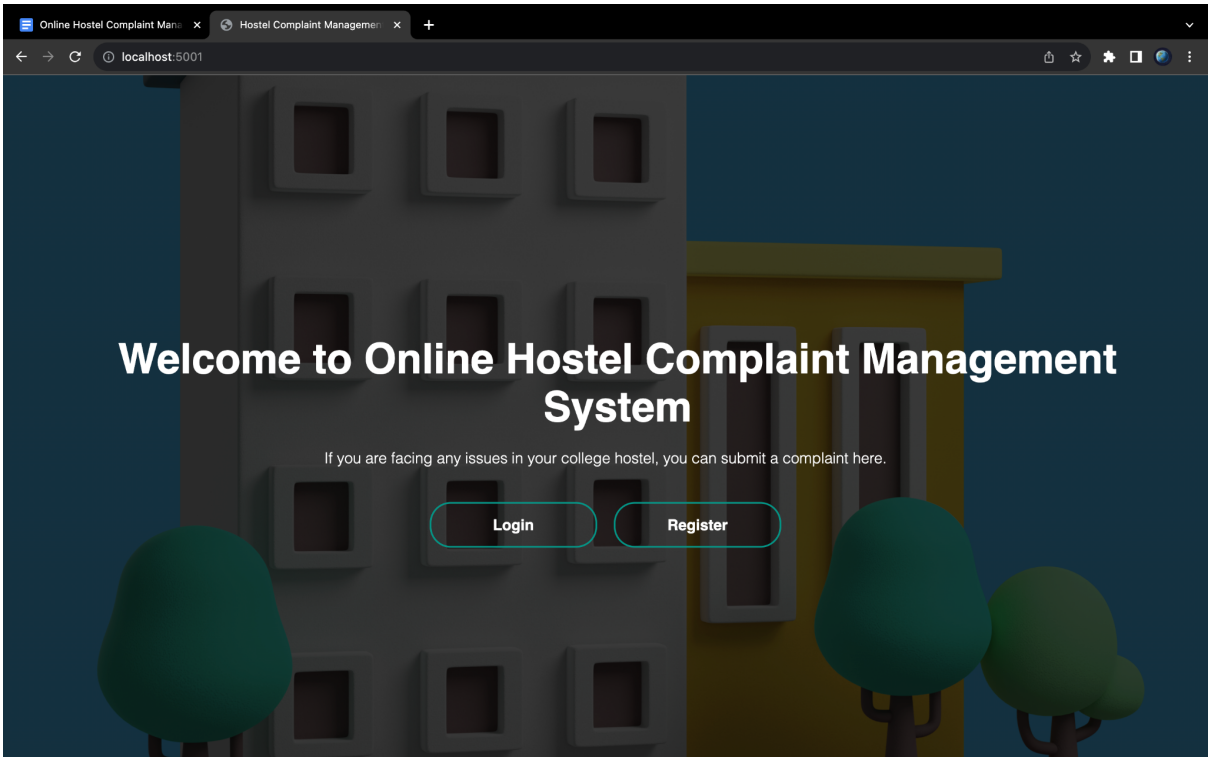
Example Map Function:

```
function (doc) {
    emit(doc.timestamp, { type: doc.type, username:
doc.username, action: doc.action });
}
```

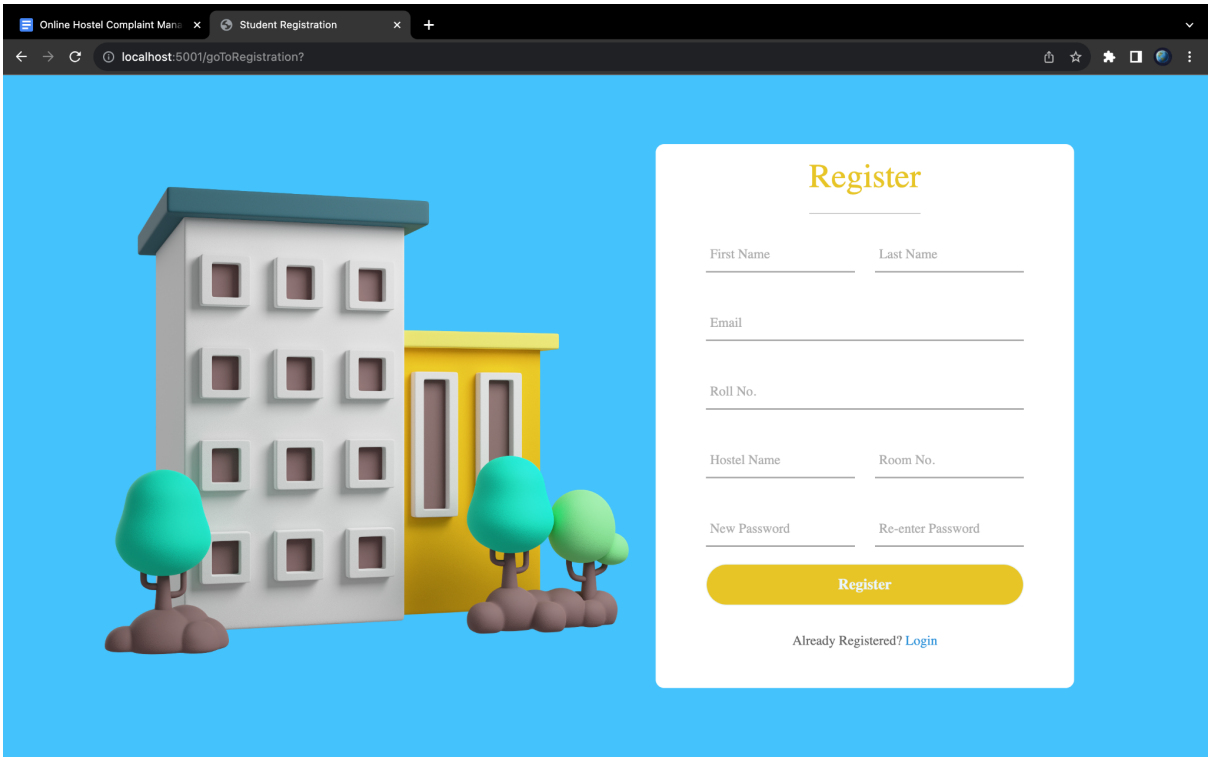
FrontEnd Design

The frontend design of our online hostel complaint management system is a critical component that directly influences the user experience. It focuses on creating an intuitive, user-friendly interface to ensure ease of navigation and efficient interaction with the system.

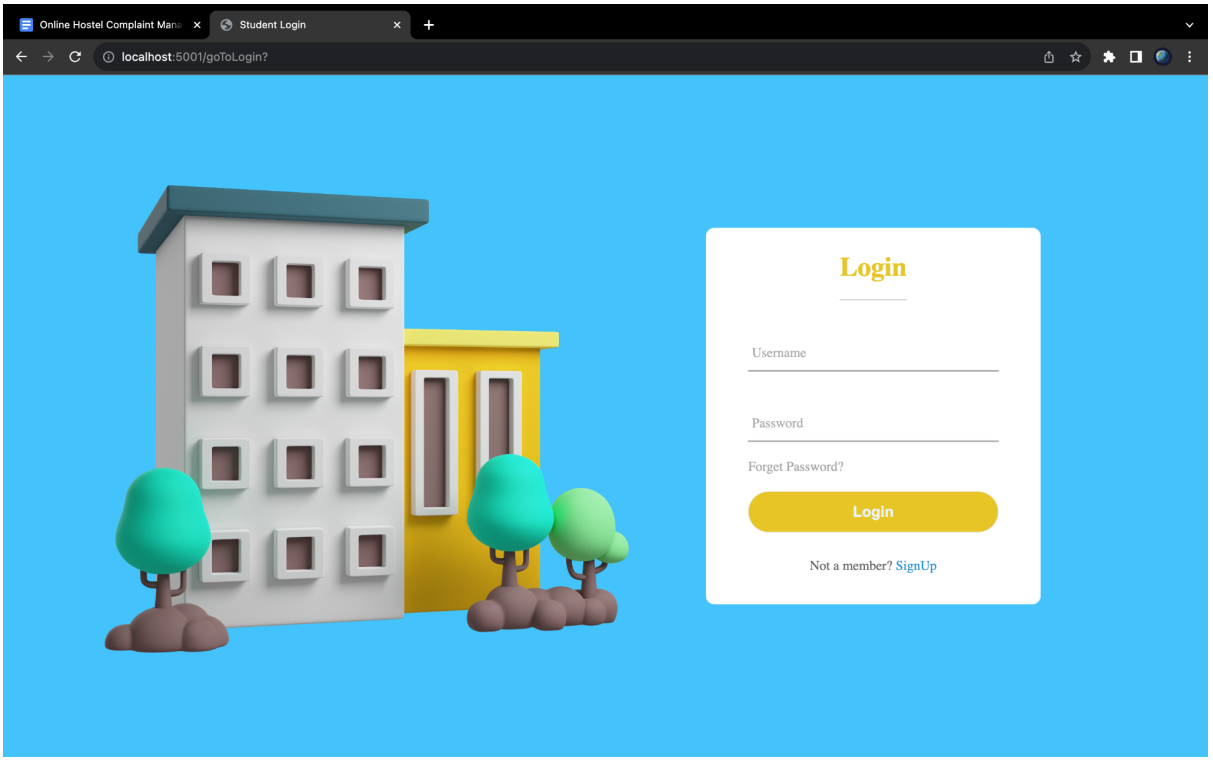
1. Home Page:



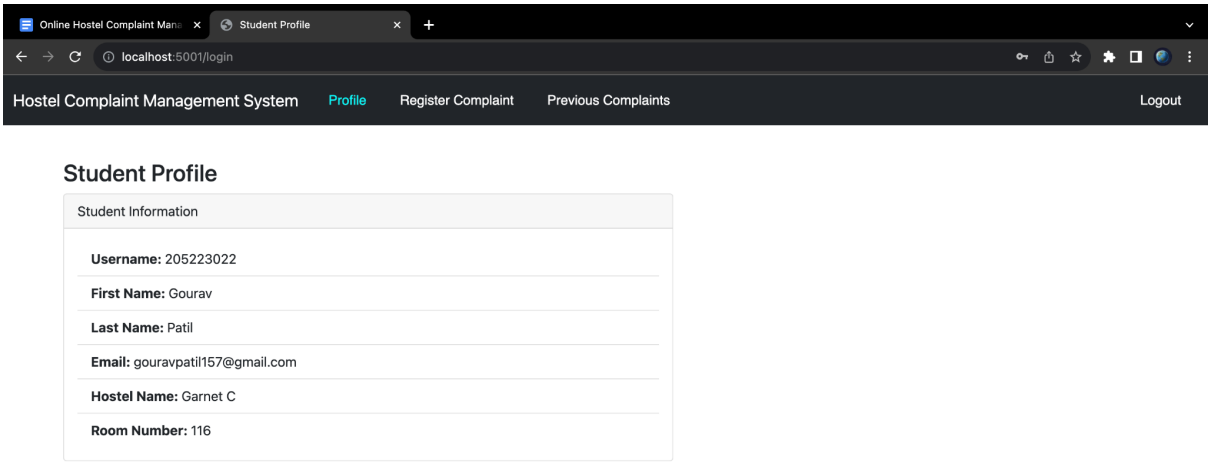
2. SignUp Page:



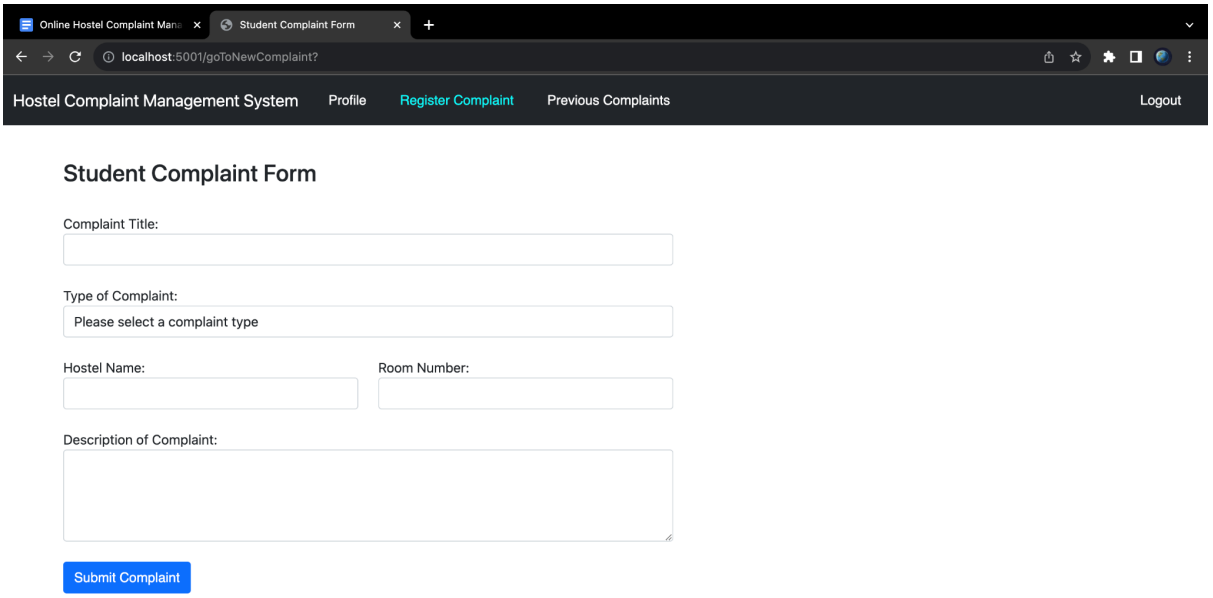
3. Login Page:



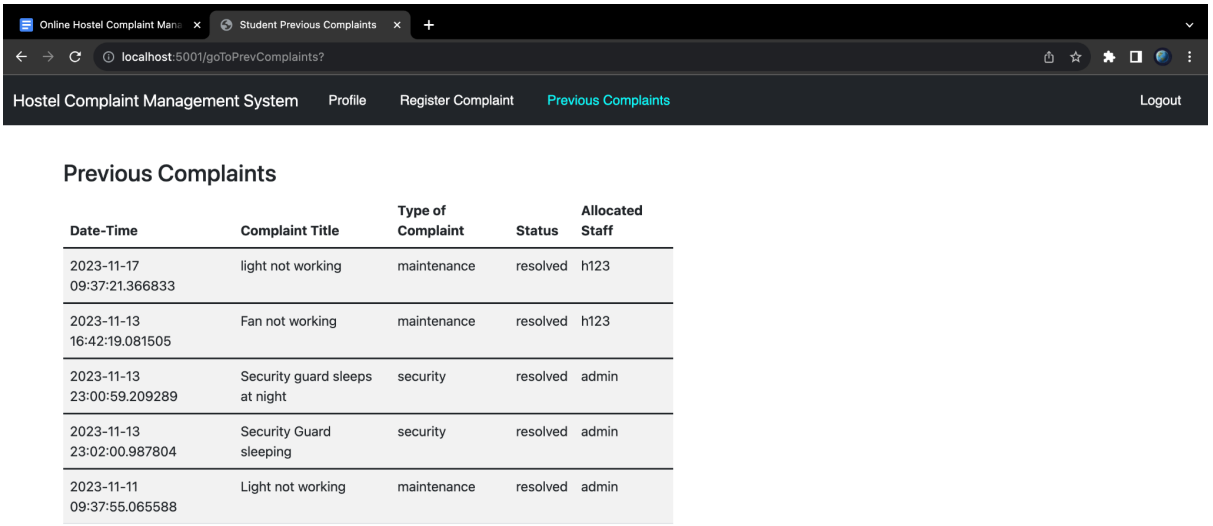
4. Student Profile Page:



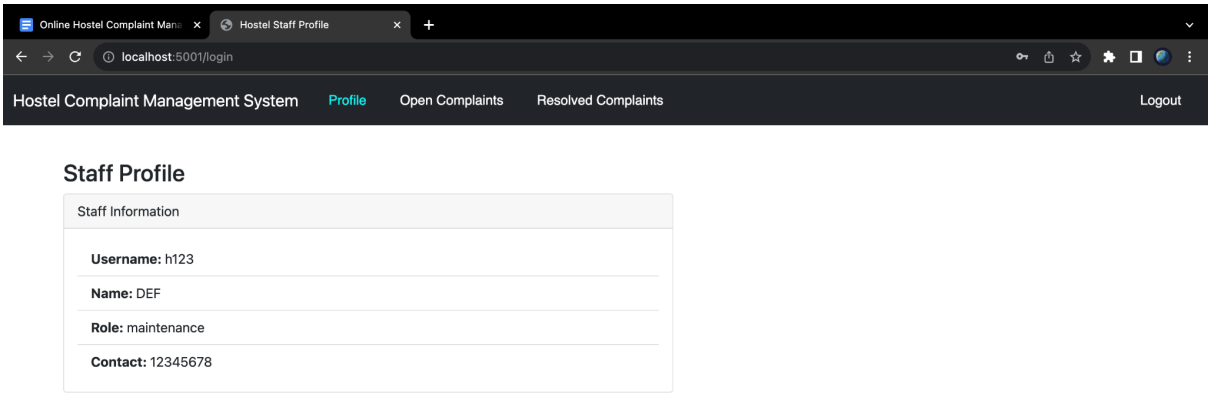
5. New Complaint Registration page:



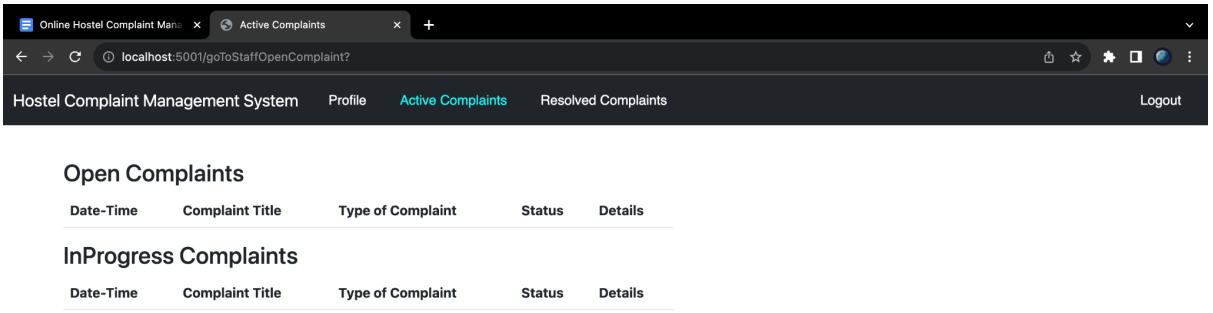
6. Previous Complaints Status Page:



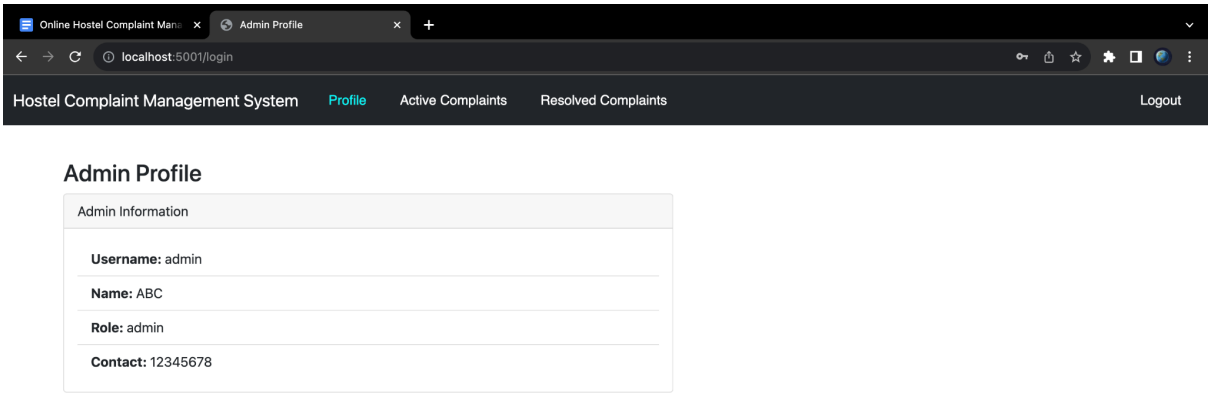
7. Hostel Staff Profile Page:



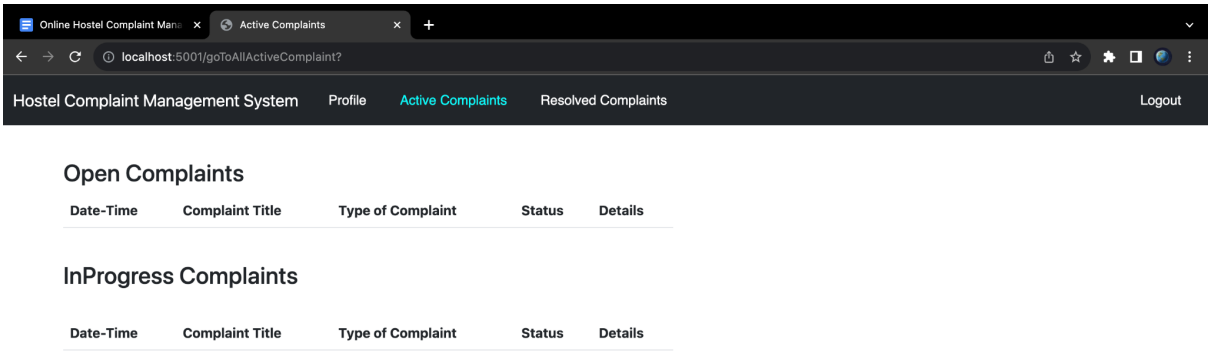
8. Hostel Staff New and InProgress Complaints View Page:



9. Administrative profile page:



10. All the complaints Page:



Conclusion

In the course of developing our online hostel complaint management system, we successfully implemented several key features, including real-time complaint updates, a user-friendly interface, and robust security measures. These achievements represent significant milestones in delivering a system that meets the needs of both students and hostel staff.

Looking ahead, we envision future enhancements to further elevate the system's capabilities. This may include additional analytics features for administrators, expanded user roles, and integration with emerging technologies. By keeping an eye on evolving user needs, we aim to continuously improve and adapt the system to remain effective in the long term.