CS 432: Databases

Assignment 4: DEPLOYING THE DBMS

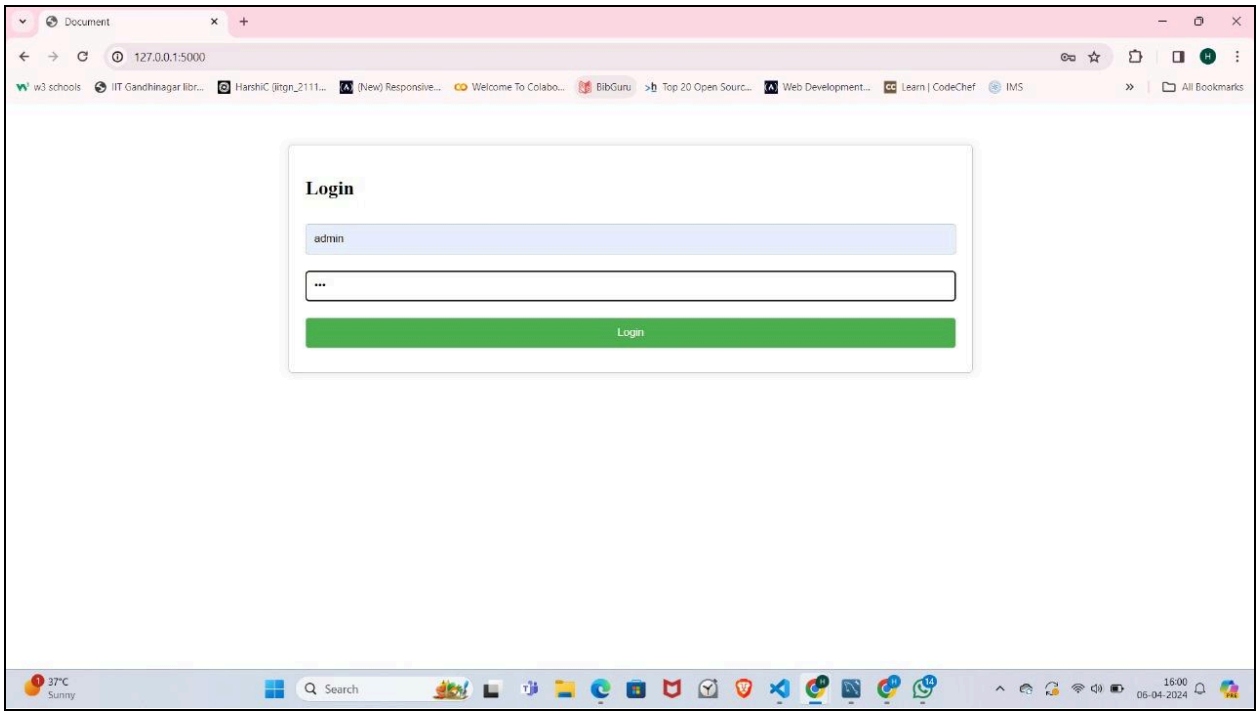
3.1 Responsibility of G1:

1. **The G1 takes two feedbacks from the stakeholders, one initial feedback (on or before 8th April 11:59 PM), and then makes relevant changes as suggested per the first feedback, then final feedback (on or before 11th April 11:59 PM) post changes. The write-up/documentation should have screenshots before the first feedback, after the first feedback, and after the second feedback. If a team discusses with multiple stakeholders, please fill out the forms again (Initial and final feedback forms).**

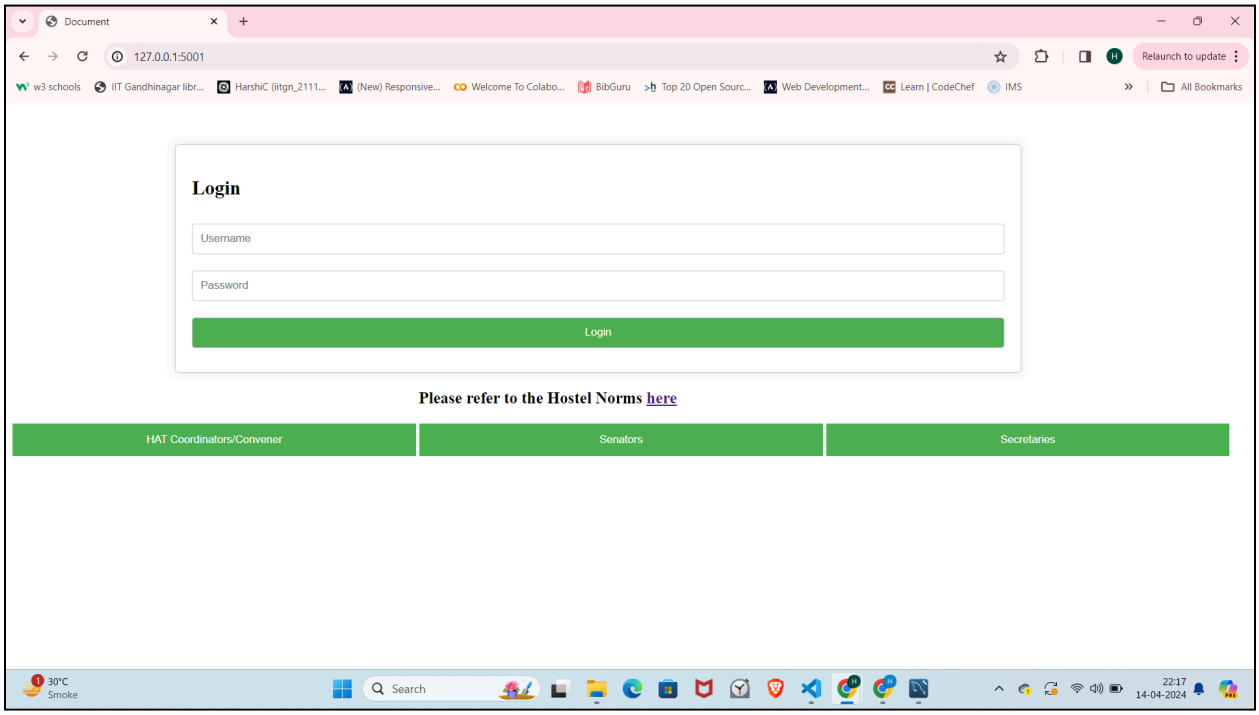
Feedback:
Shweta Bichawat and Barkha Govind from the hostel office provided valuable feedback on our website. They appreciate its utility but suggested incorporating additional features like hostel norms. They raised concerns about the website's performance with increased data volumes, emphasising its need to remain user-friendly while handling large datasets. They also recommended consolidating similar data columns, such as those for stay rooms and store rooms, into one column and proposed including information about secretarial roles and the hostel in charge.

- It should be able to accommodate large datasets.
  One of the stakeholders' concerns was whether this website would work for larger datasets. For this, the student's table in our database has around 10,000 entries and works properly. If we reduce the entries to 2000 (the approximate number of students in IITGN), the website will work more smoothly compared to before.

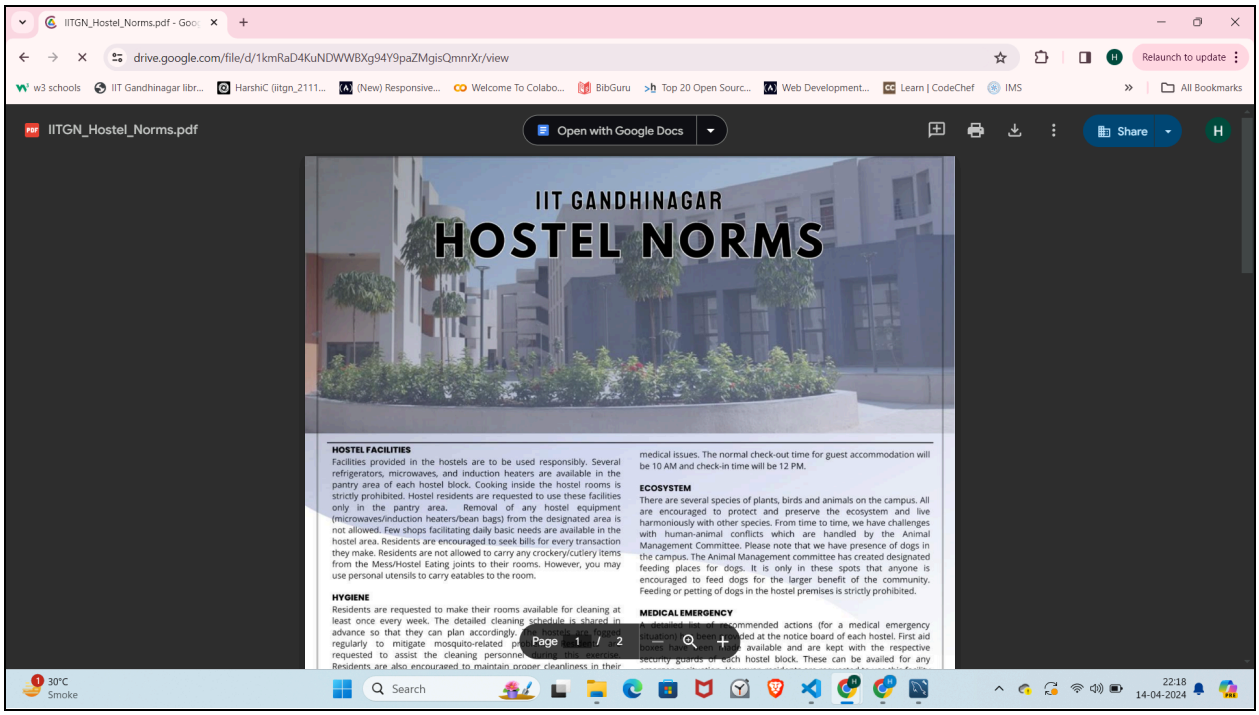
- Should contain information about Hostel norms
  - Before Feedback:



- After feedback:

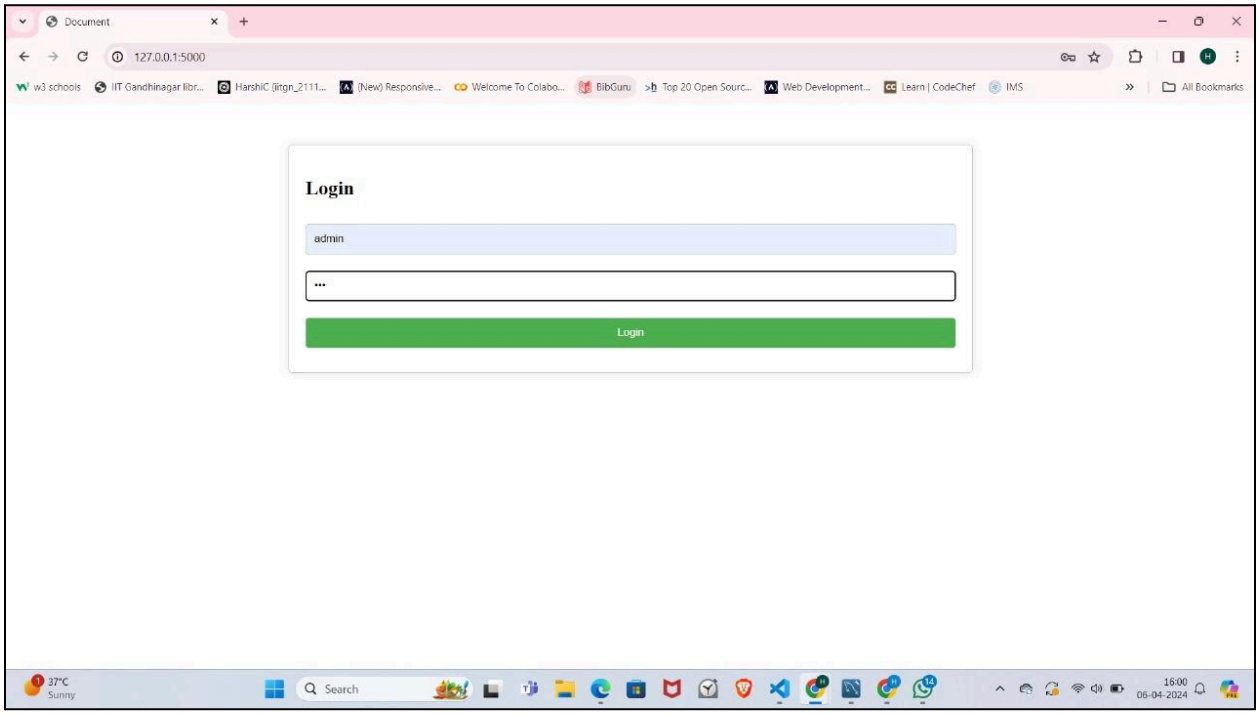


In the above image, you can see that we have added a link to show hostel norms - "Please refer to the Hostel Norms here."

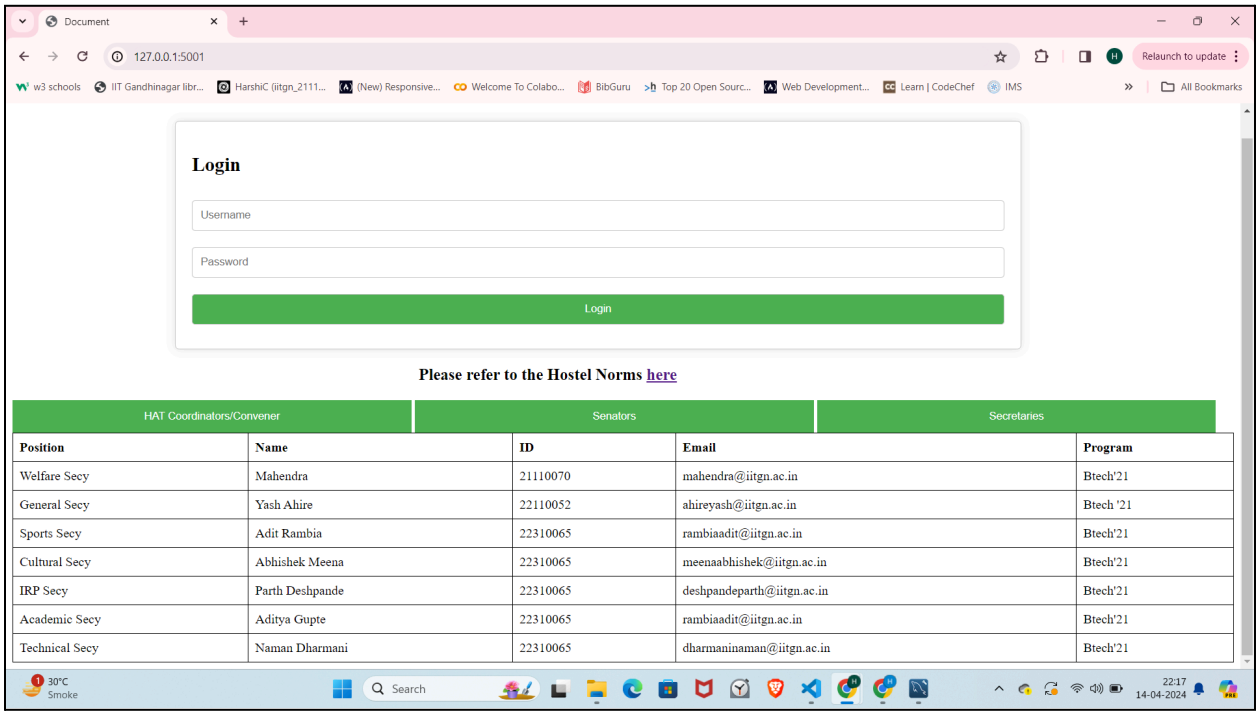


When you click on that link, this page opens up.

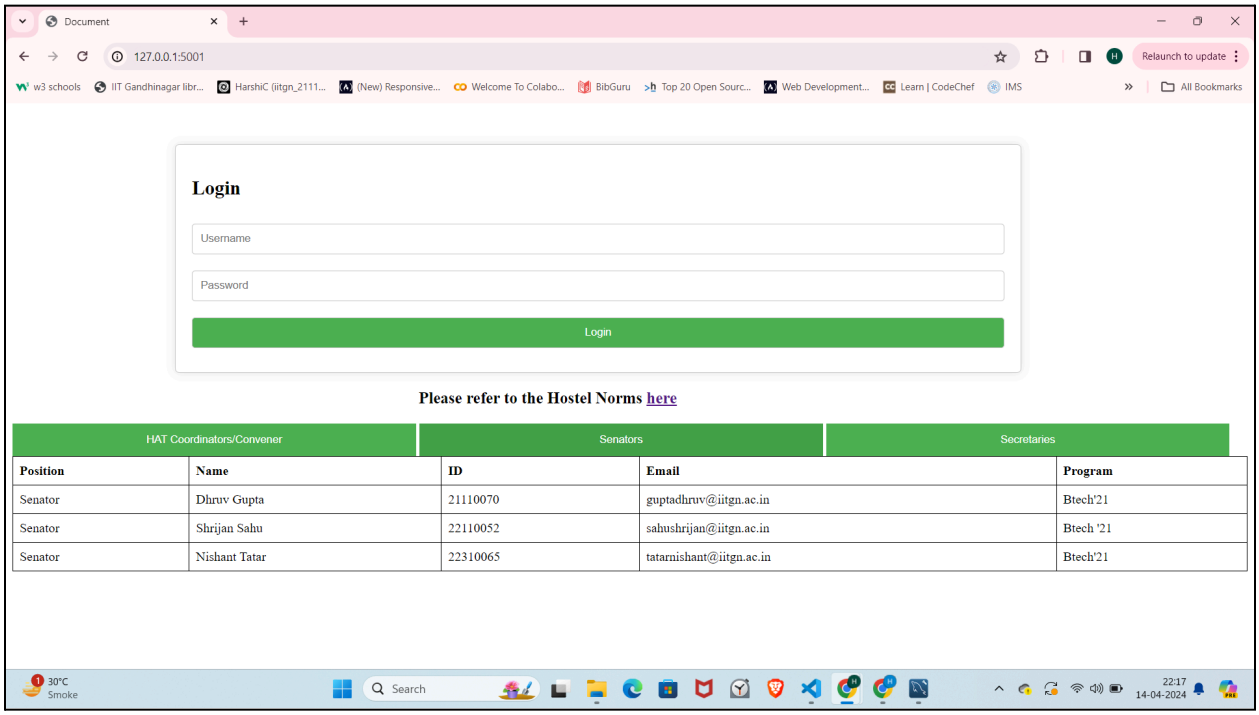
- <u>Should contain information about HAT Coordinators, Senators and Secretaries</u>
  - Before Feedback:



- After feedback:

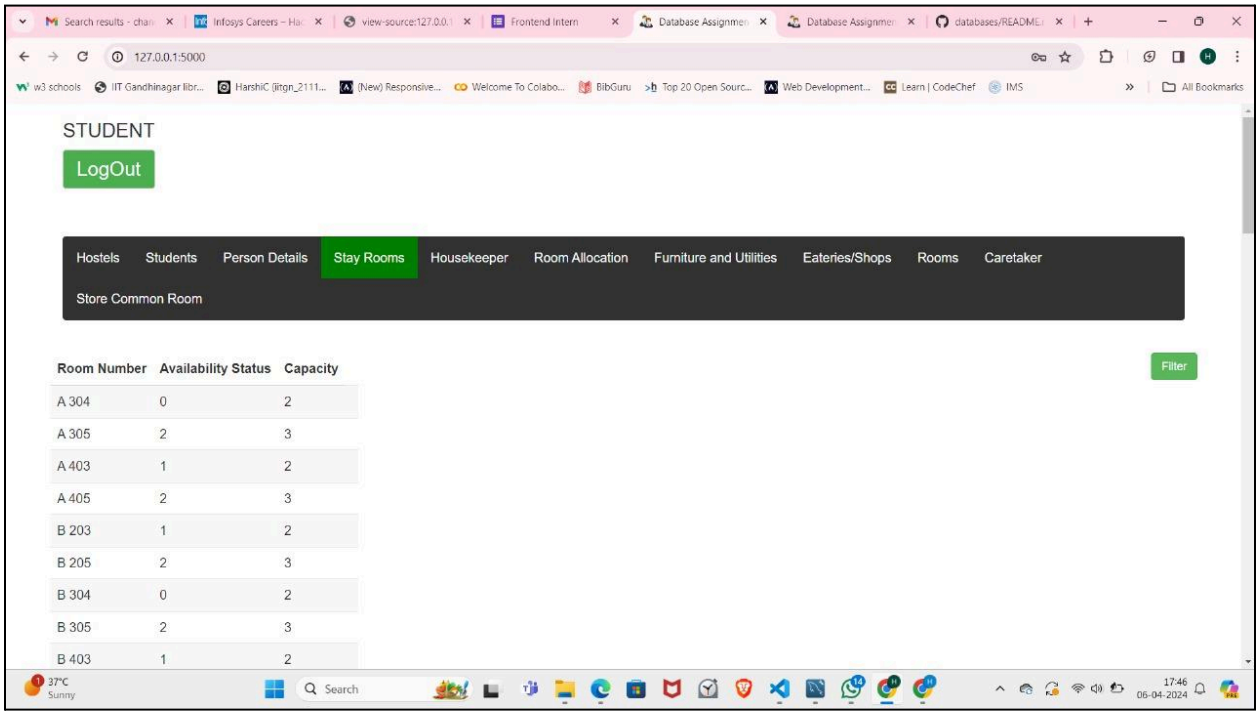


When you click on the secretaries button, this information opens up.

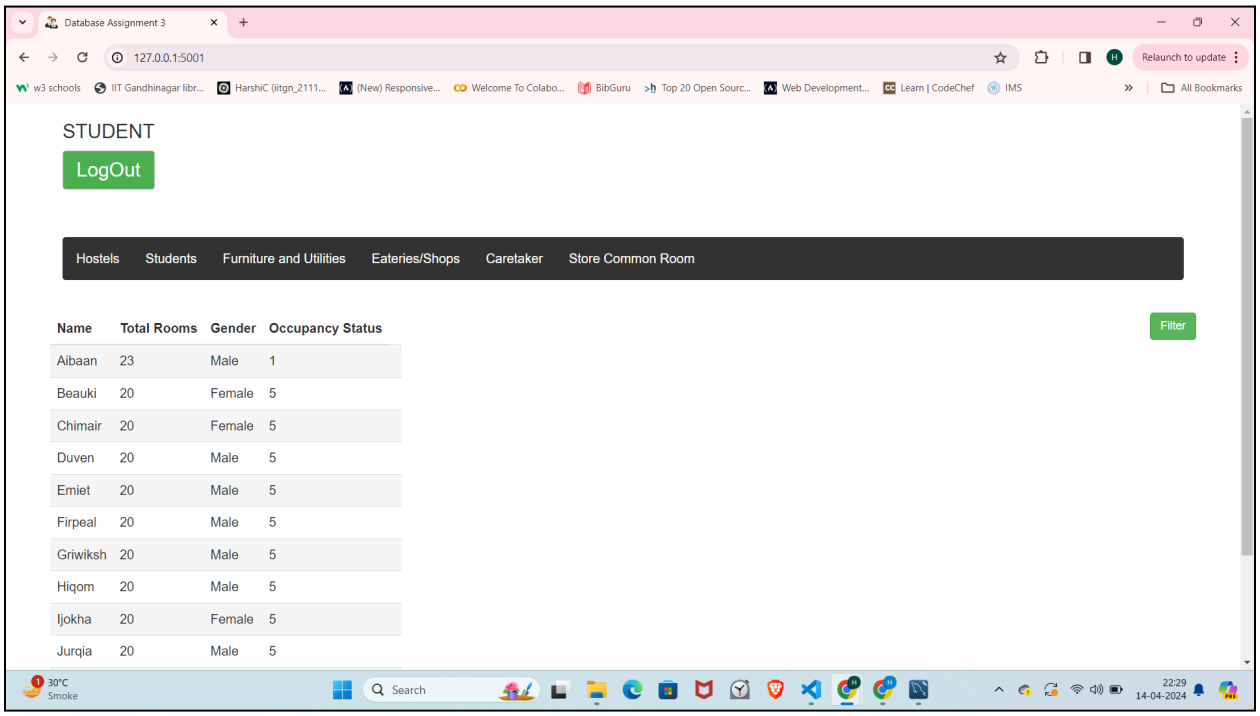


When you click on the senator's button, this table opens up.

- <u>Students should not be given the privilege to see room allocation tables.</u>
  - Before feedback:



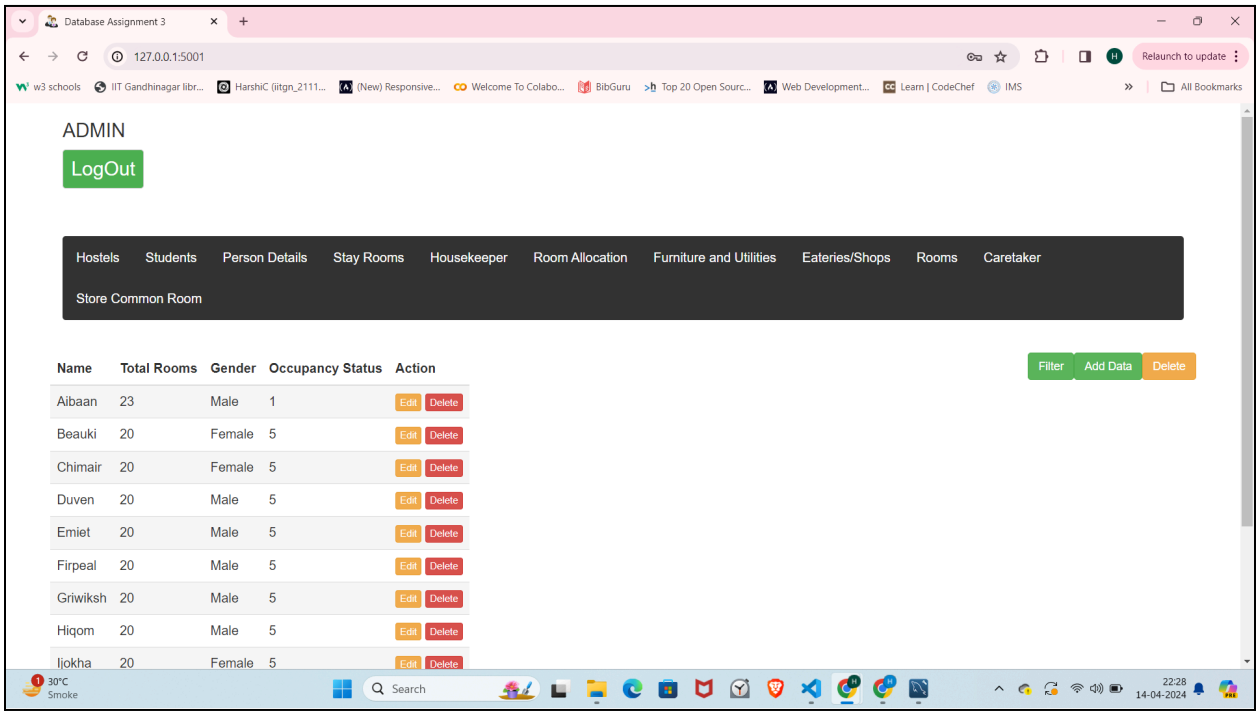
- After Feedback:



After the feedback tables related to the rooms had been removed from the students' page,

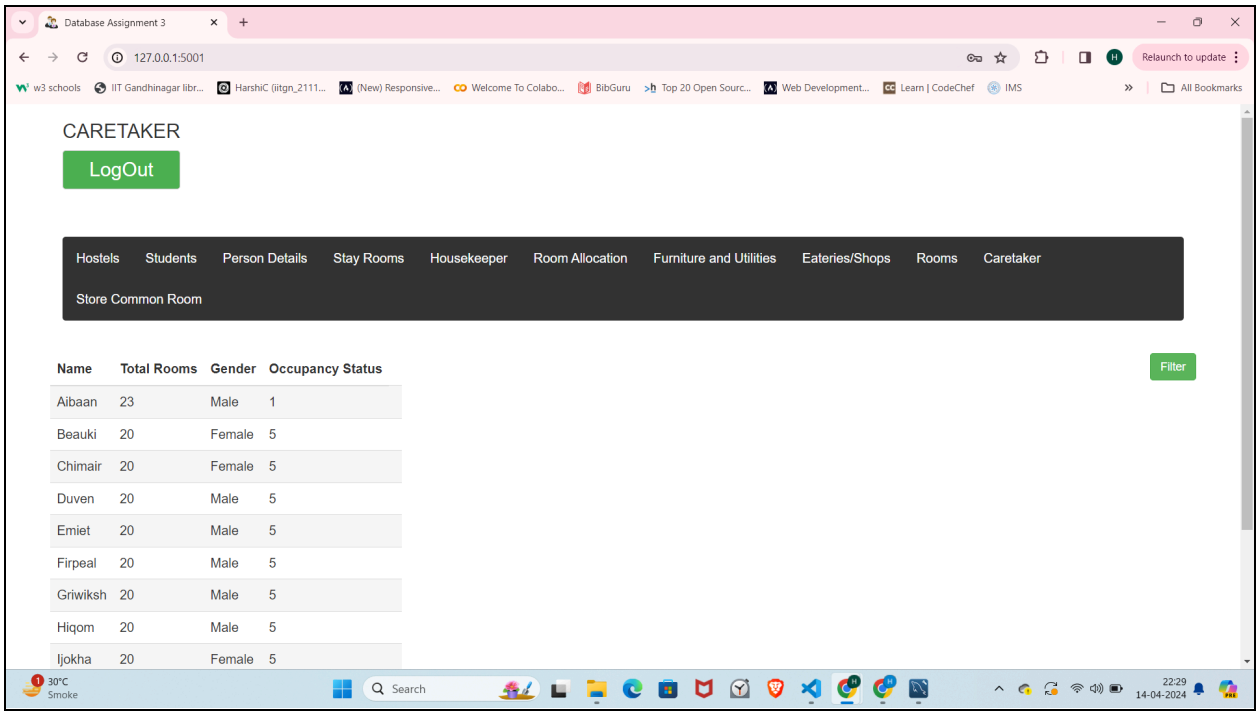
2. **Attach screenshots of different views [along with a write-up on their privileges] of the database as seen by different classes of users.**
   Different views for different users:

1. Admin
   - The admin can edit, delete, and filter all the tables and all their entries.

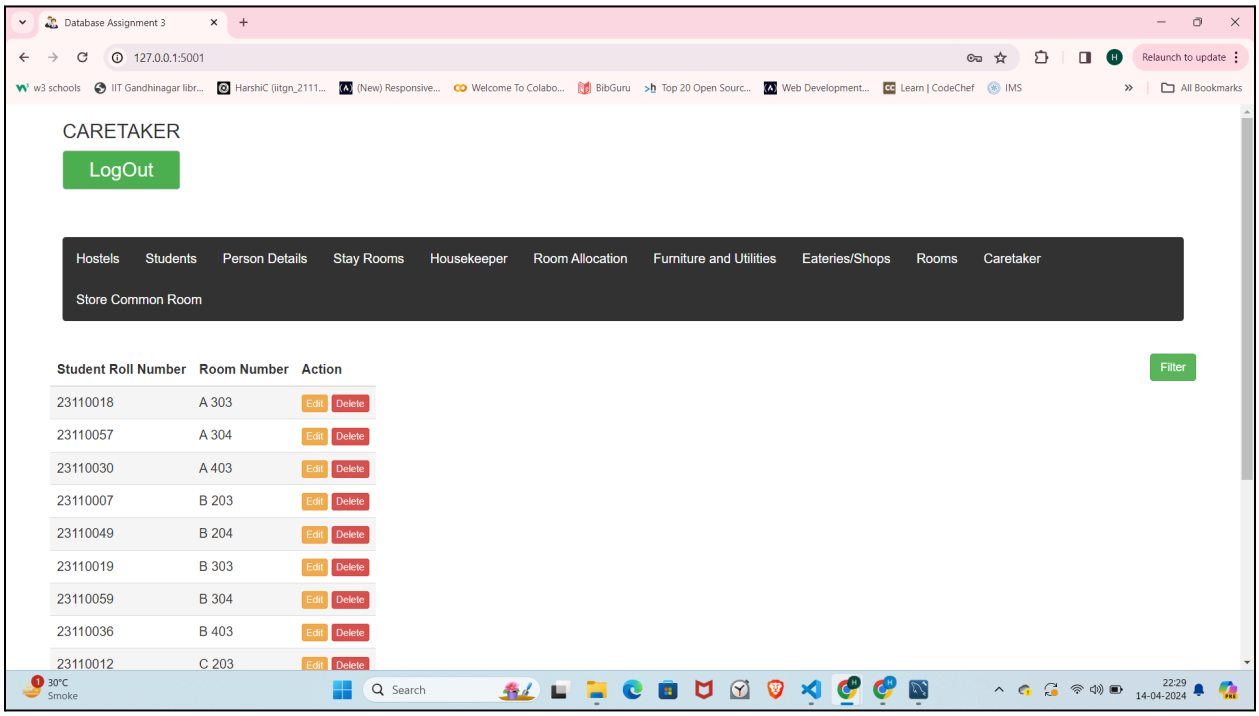


This page opens up when the admin logs in.

2. Caretaker
   - The caretaker can edit and delete entries in a few tables not all. However, they can view all the tables that the admin can view.

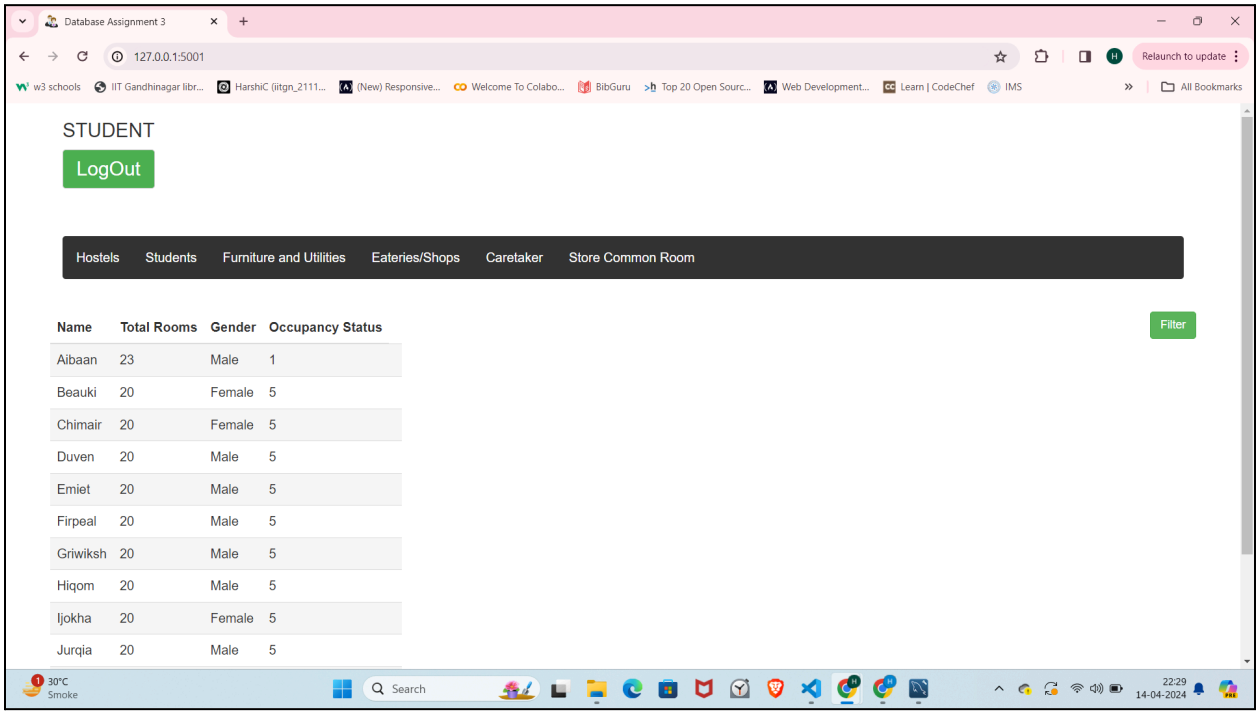


This page opens up when the caretaker logs in. The caretaker can edit/delete only a few tables. It cannot edit the above table of hostels.



The caretaker can edit the above table of students.

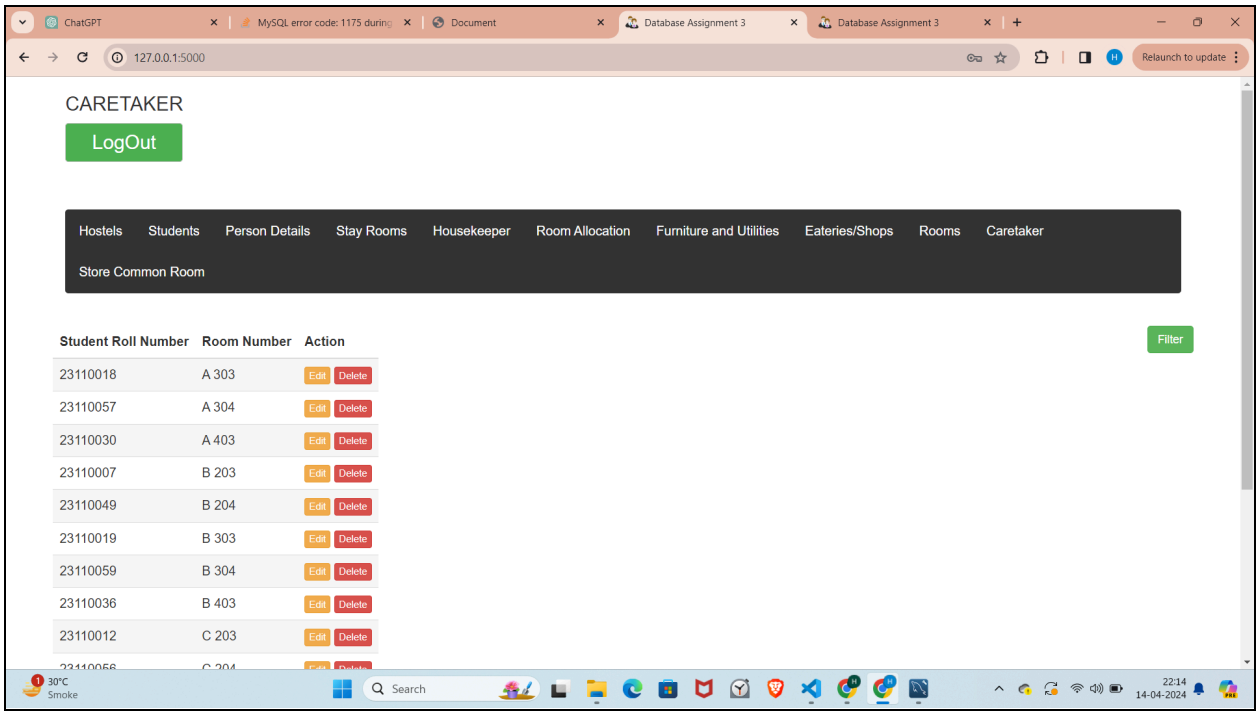
3. Students
    - Students only have the privilege to view and filter data. Also, as suggested by the hostel office, students should not be able to view data regarding room allocations; therefore, those tables have been removed.



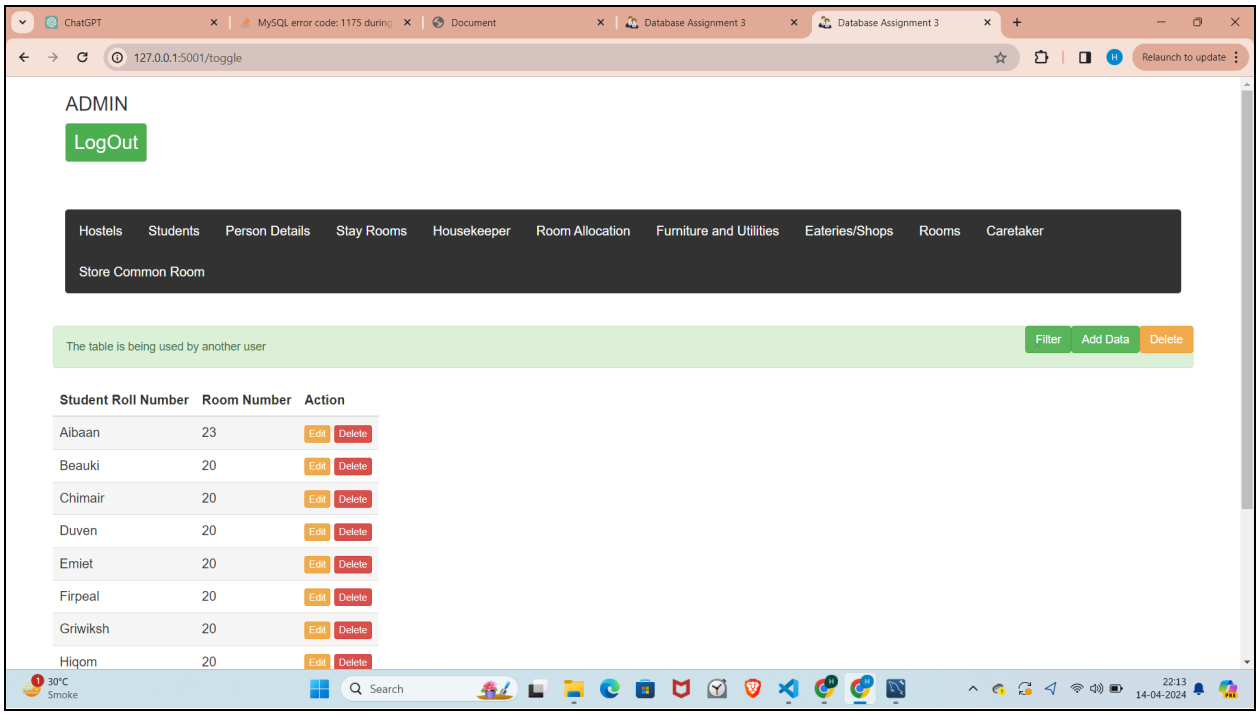
This page opens up when a student logs in.

3.2 Responsibility of G2: 40 Pts.

1. **Concurrent multi-user access: Multiple users with different roles can access and update the database concurrently. In such a scenario, two different users cannot update the same item. For example, locks can be applied to tables in MySQL. (10 Points)**



We have implemented locks on our system. We logged in to our website using two portals. One is from the caretaker ID, and the other is from the admin ID. If the admin tries to edit/ delete any of the data, he will be unable to do so because the first user locks the table. It will pop up a message that another user is currently using the table.



Since the caretaker has logged in and is editing the same page, another user(admin in this case) cannot make changes.

2 Implement the changes in the database as per the feedback received from stakeholders.

-According to the feedback from the stakeholders, not many changes were required in the backend. The front end only required changes, like adding hostel norms, HAT Coordinators, and Secretaries. We merged similar columns and removed columns that were not required. Their primary concern was the effectiveness of the website and its speed, but since our website already has large datasets with 10,000 entries, it will easily be able to accommodate data from IIT Gn.

3 Add Google authentication for login and registration. (only IITGN users can log in and register)

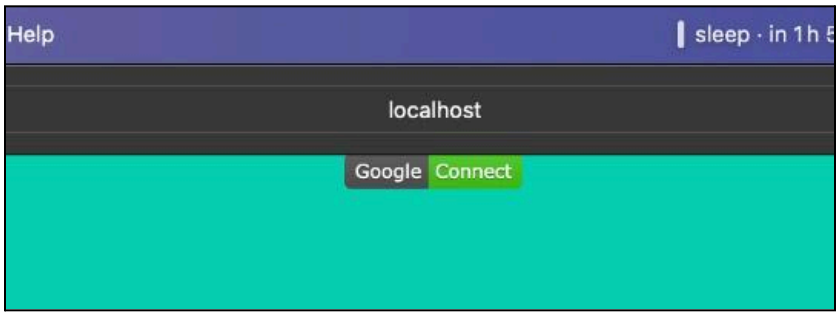
We have implemented Google authentication to allow only IITGN users to log in and added an option to sign in if you are a new user. For new users, the student page is rendered. For this, we used the reference provided in Assignment 4.

At line 58 in __init__.py, add this piece of code:

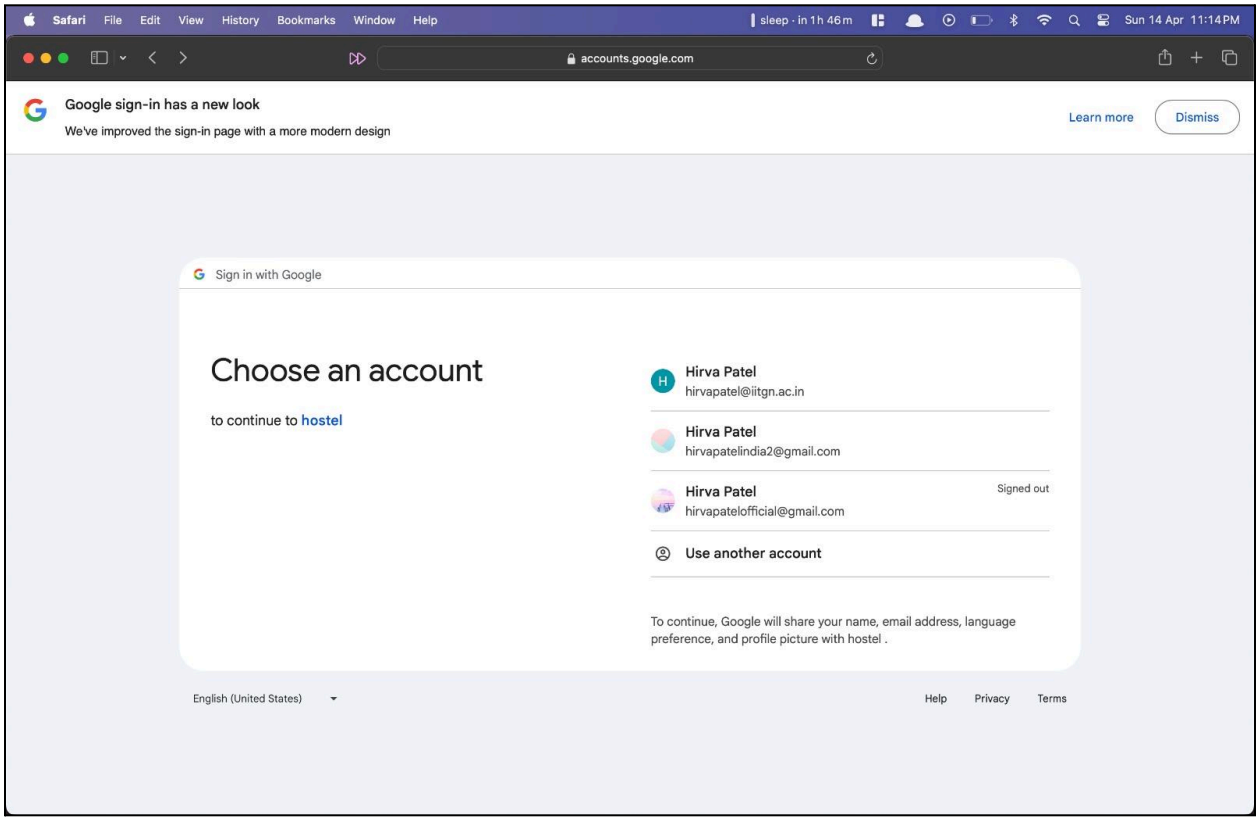
```
GOOGLE_CLIENT_ID =
"640414763215-saog82ngnv8ljkhimv9midh1u517kpaj.apps.googleusercontent.com"
    GOOGLE_CLIENT_SECRET = "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"

    CONF_URL= 'https://accounts.google.com/.well-known/openid-configuration'
    oauth.register(
        name='google',
        client_id=GOOGLE_CLIENT_ID,
        client_secret=GOOGLE_CLIENT_SECRET,
```

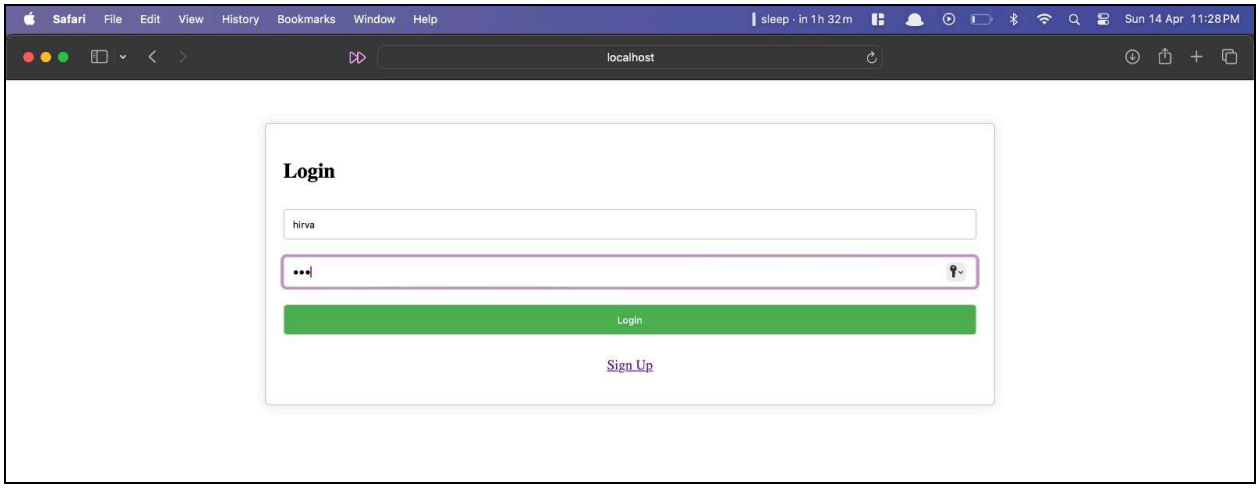
```
server_metadata_url=CONF_URL,
client_kwargs={
    'scope': 'openid email profile'
    }
)
```



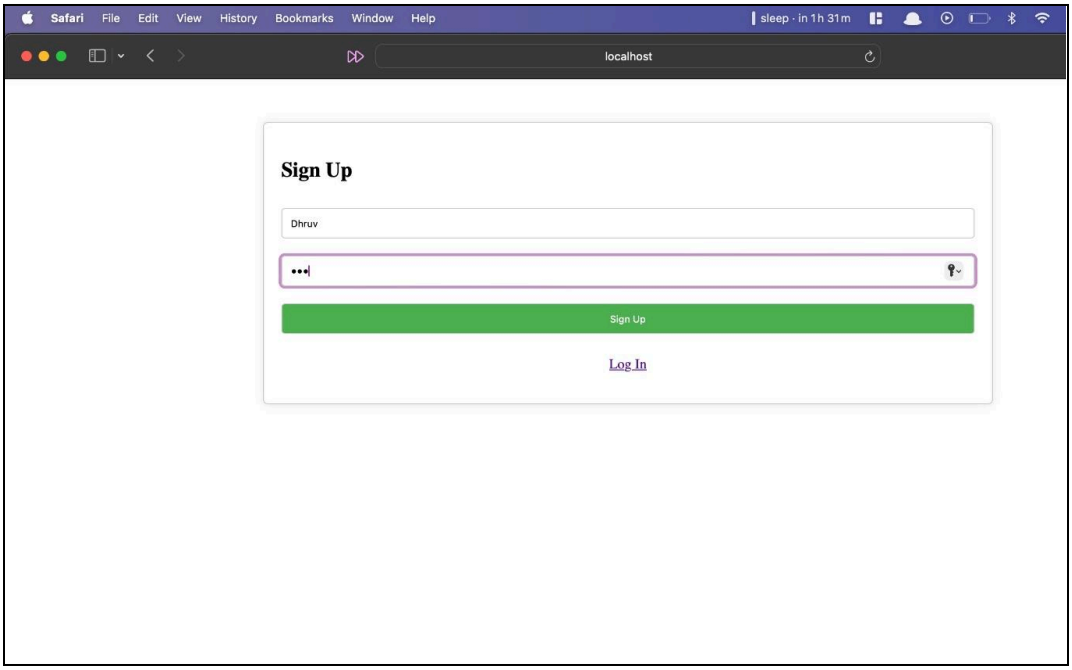
Button was added for Google authentication.



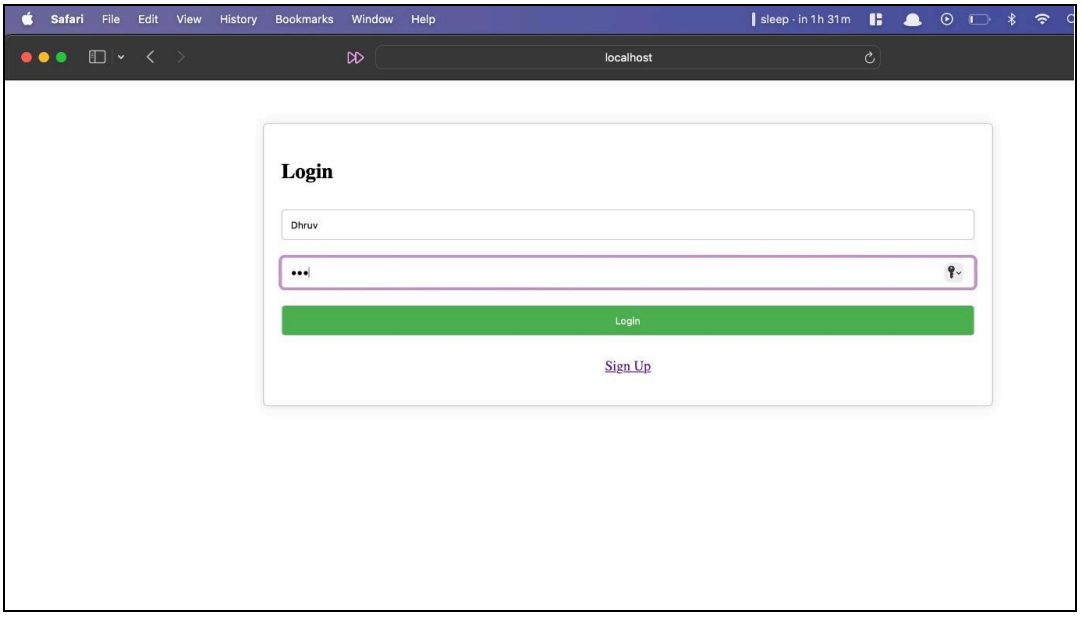
Pop-up After you click on button



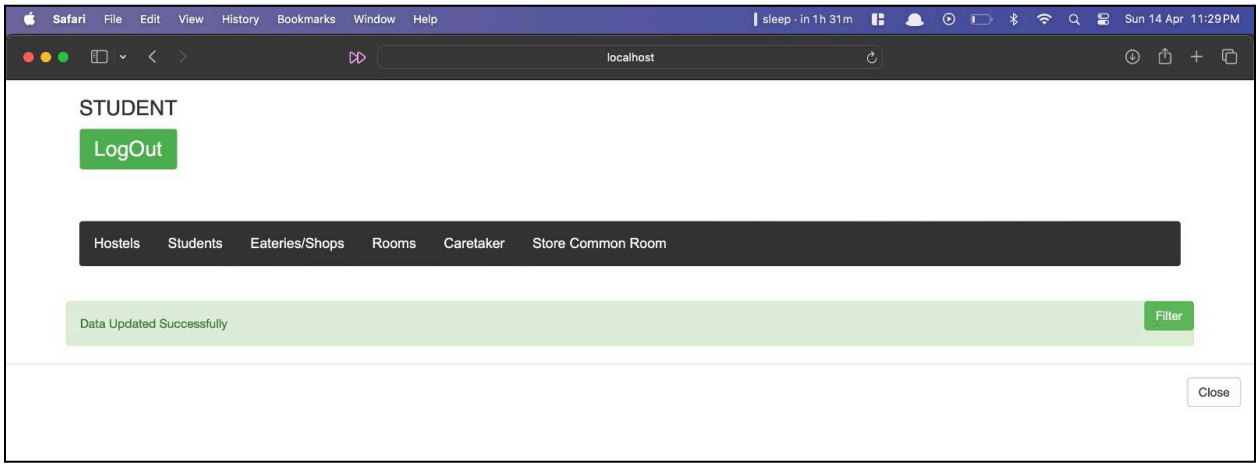
Already existing users log in using the IITGN email.



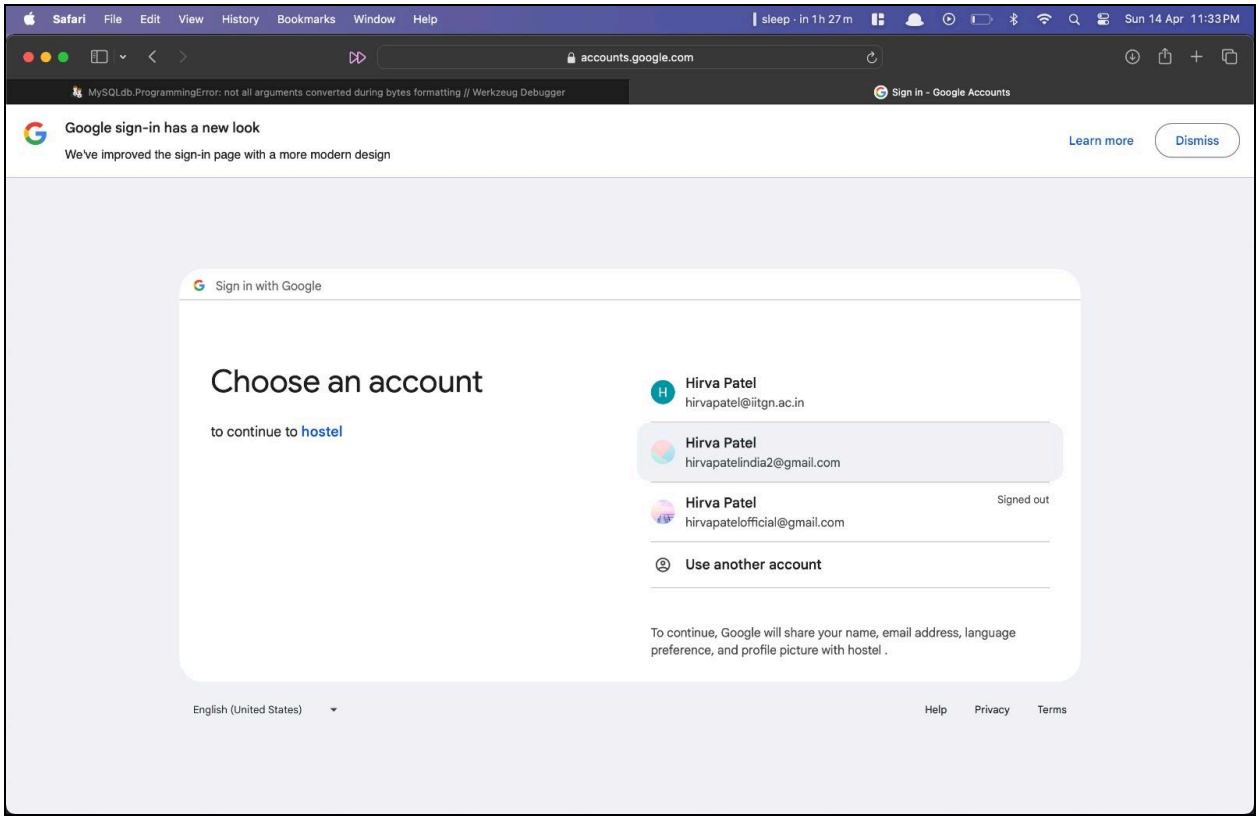
A new user signed in.



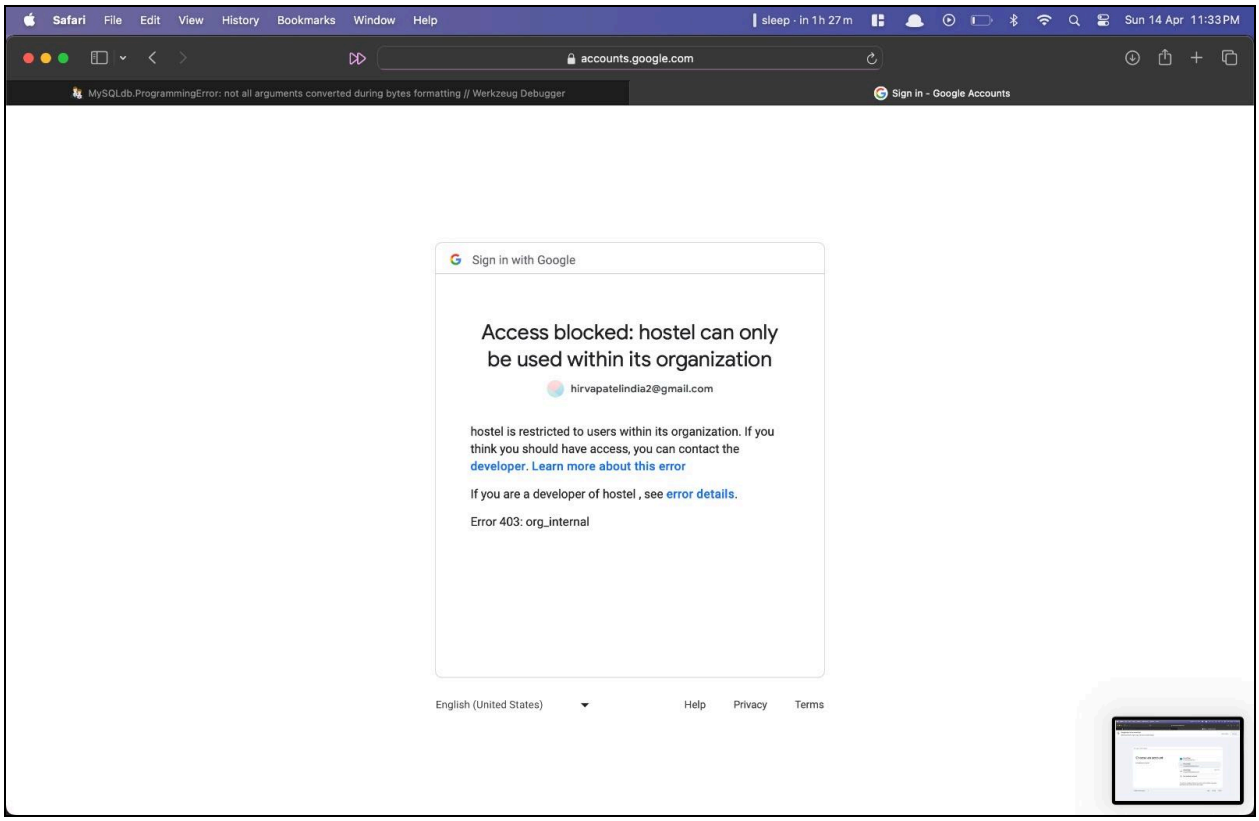
Logging in after signing up.



Data of the new user was inserted successfully.



Trying to log in use personal email ID.

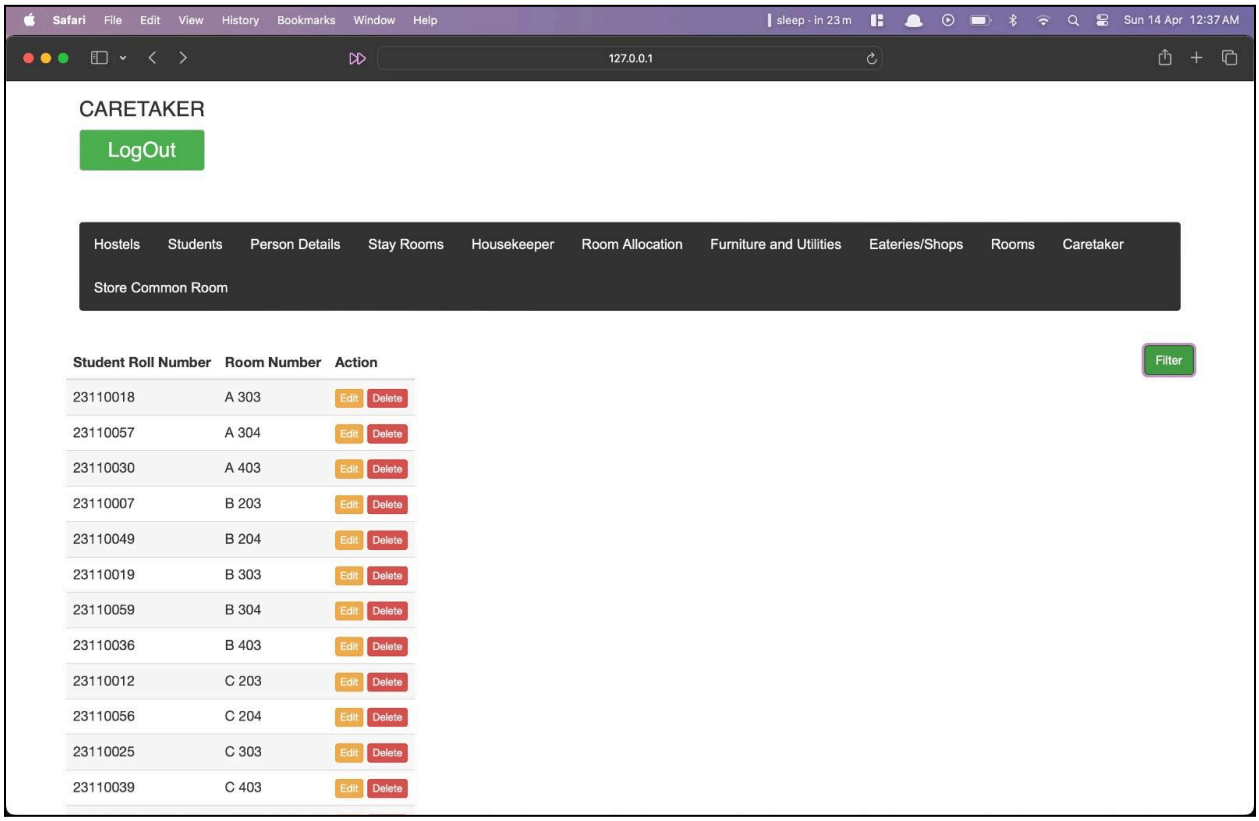


Cannot log-in using personal email ID. (IIT GN's ID required)

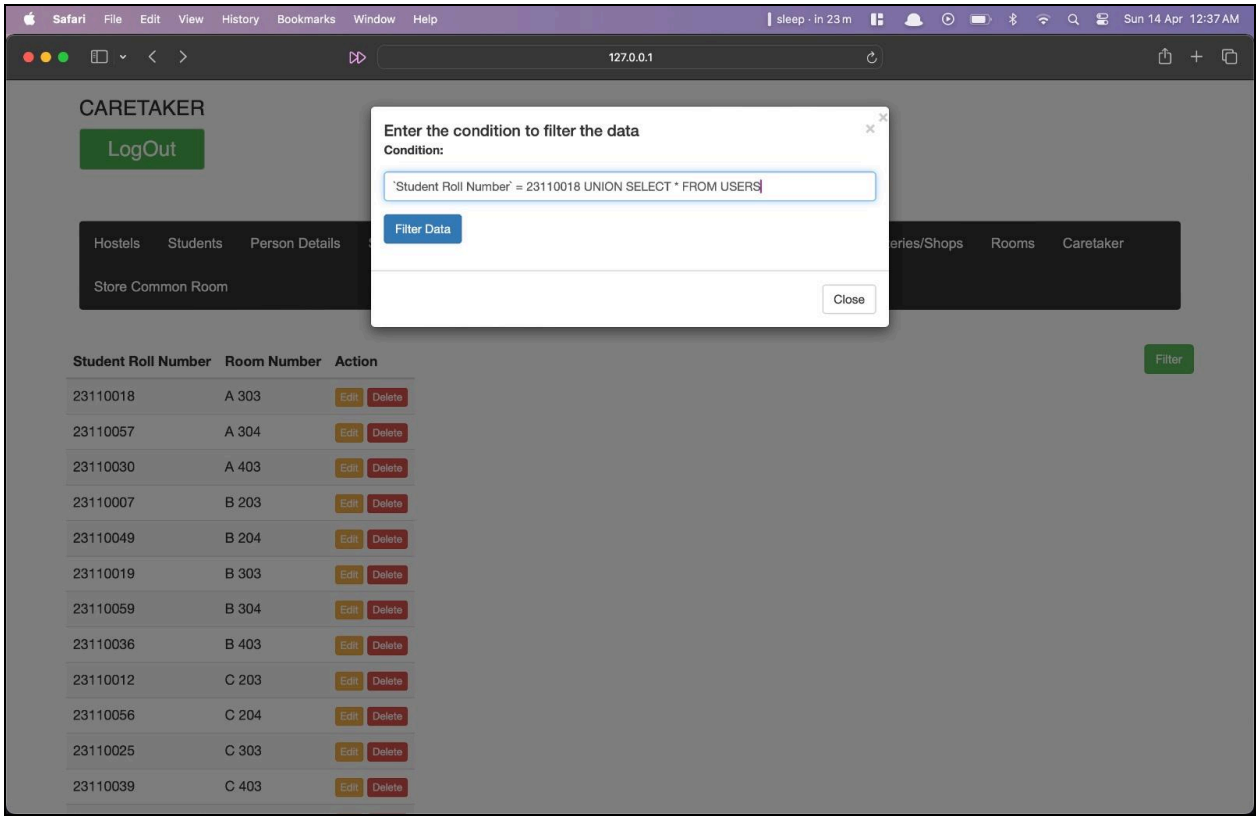
3.3 Responsibility of G1 & G2: 20 Pts

1. **Documentation and screenshots of a total of 2 attacks [SQL Injection and XSS] performed and the defences against those attacks. (15 points)**

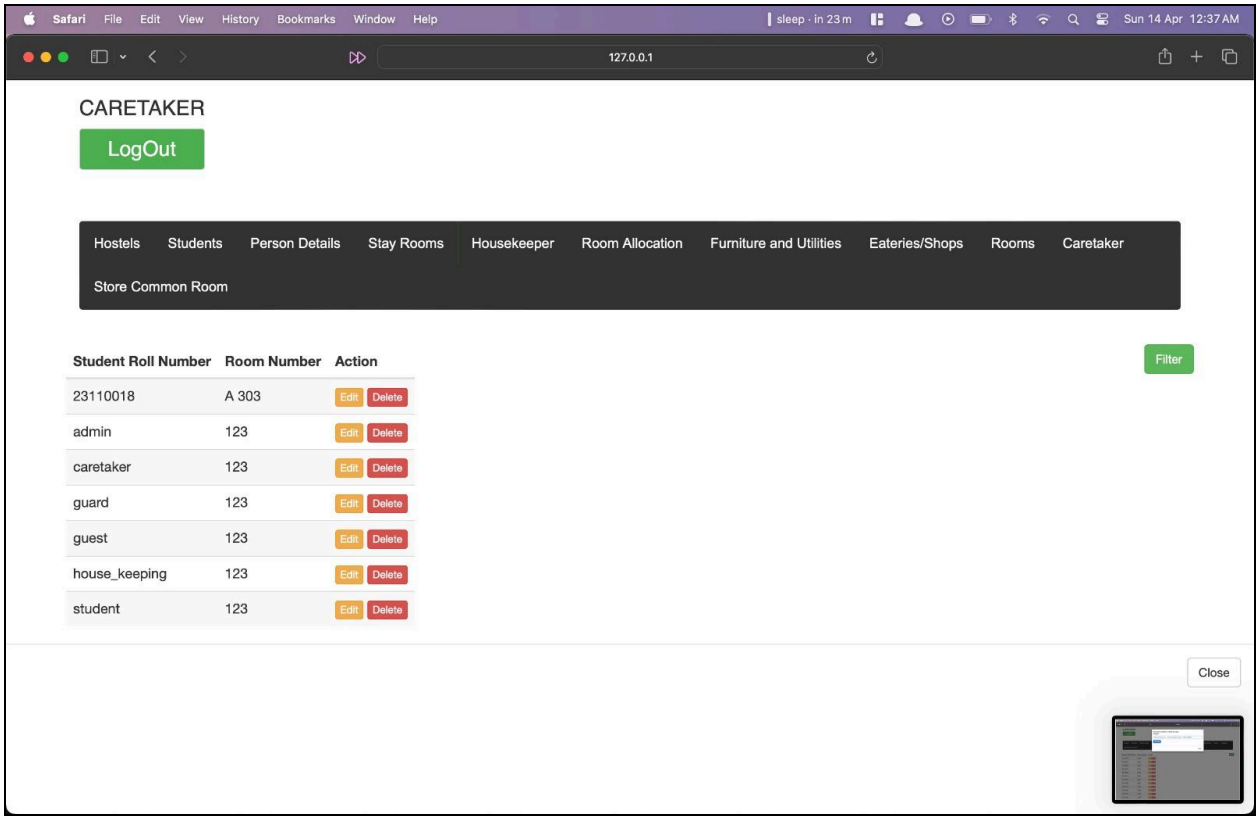
   We have made Four attacks in total.
   SQL injection:



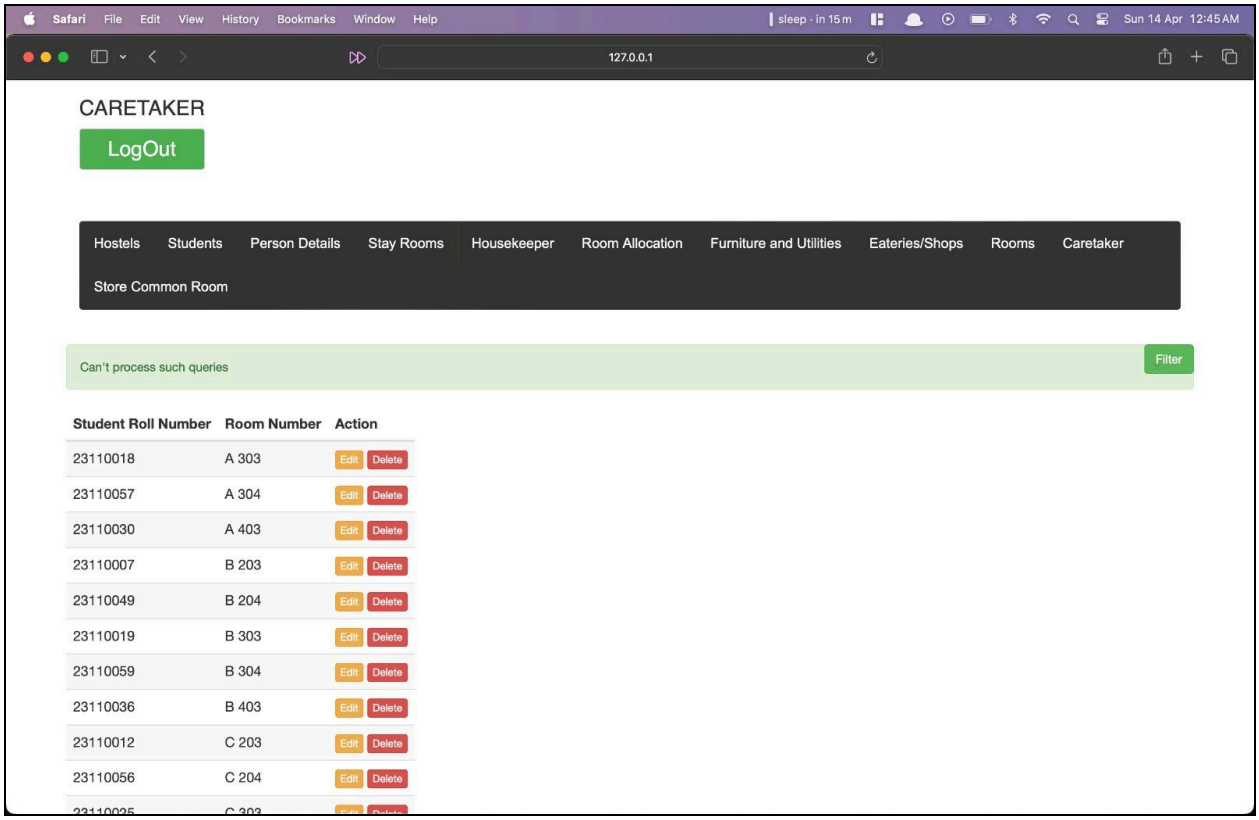
In the above image, we have opened the Room allocation table.



We open the filter section for the room allocation table and write a query to display the user table.

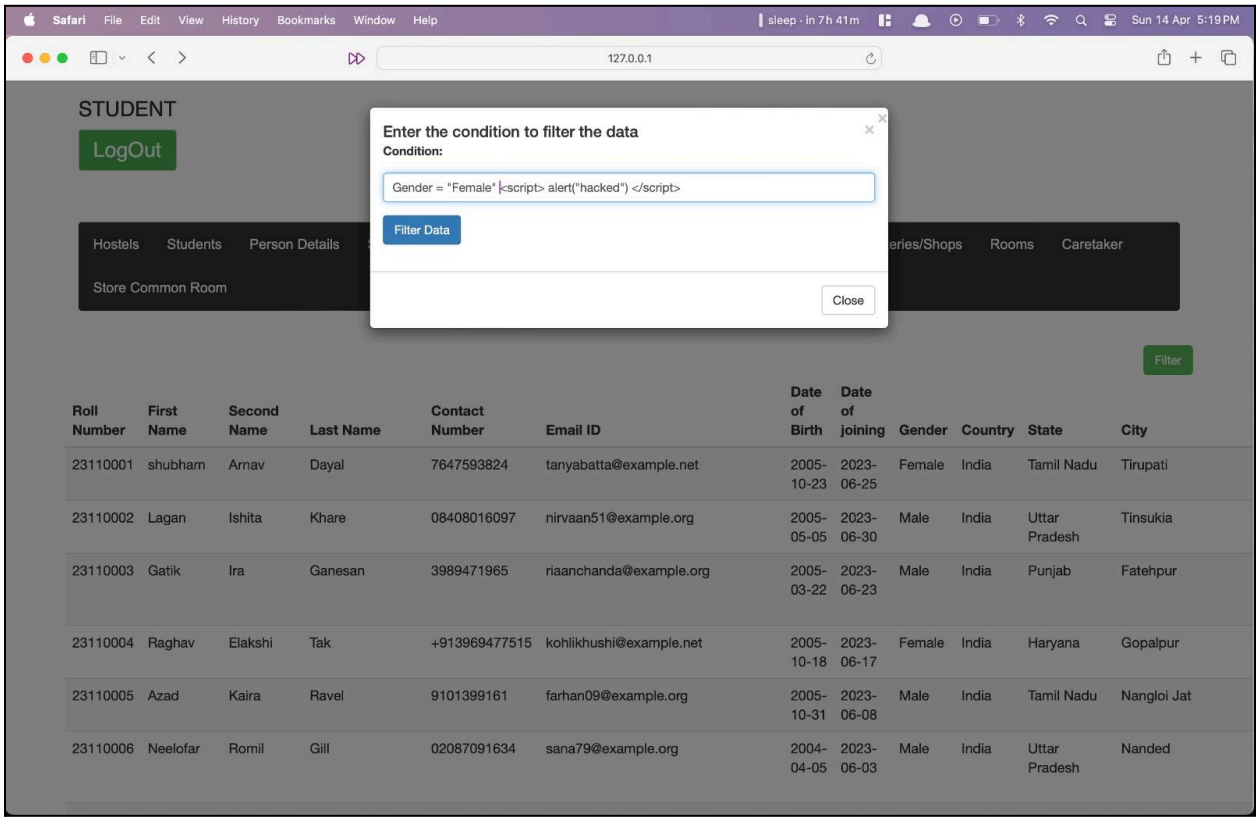


Before developing the defence mechanism, after filtering, we could also see usernames and passwords.



Implementing the defence mechanism shows that such queries cannot be processed.

XSS:



XSS attack is not valid in our case. All the conditions entered in filter boxes are directly used in the SQL Query statement and are not run or used in the HTML files.

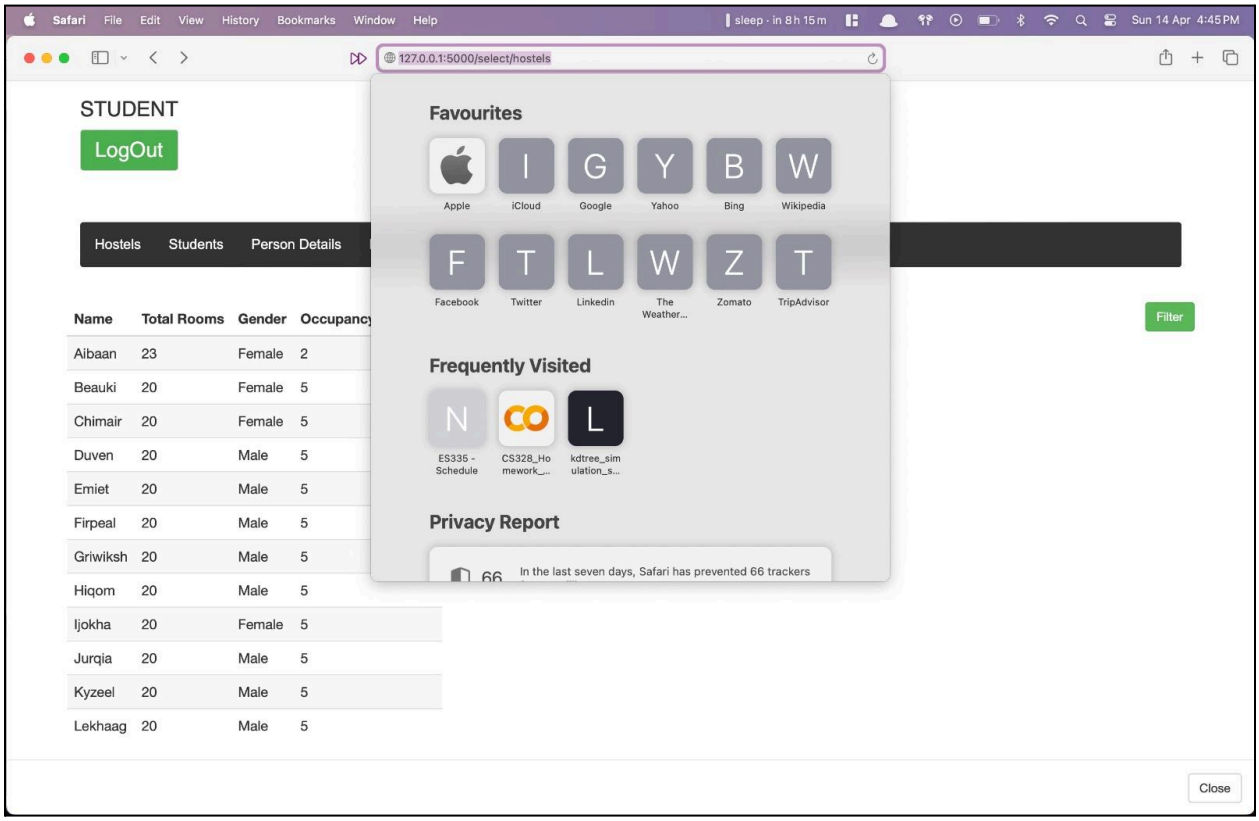
Brute Force:

Someone might be able to log in by trying different passwords many times. To prevent this attack, we have implemented the website so that after three unsuccessful login trials, the time taken to log in will increase exponentially (Two power number of times).

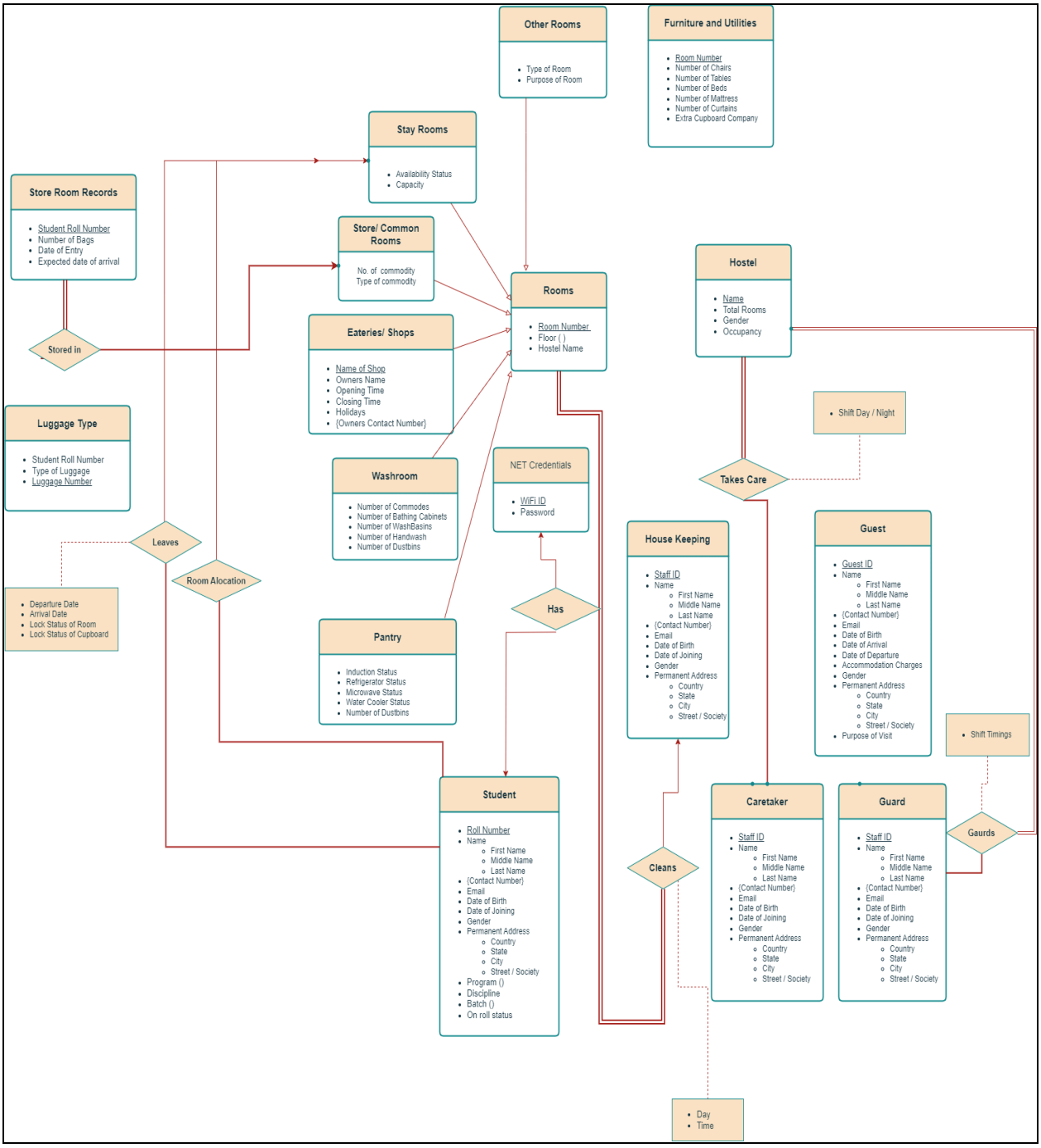
```
    if user:
        session['logged_in'] = True
        session['user_type'] = username
    else:
        if 'attempts' not in session:
            session['attempts'] = 1
        else:
            session['attempts']+=1
        if session['attempts']>=3:
            time.sleep(2**session['attempts'])
        flash("Incorrect Credentials")
        session['logged_in'] = False
    cur.close()
    return redirect(url_for('Index'))
```

Path Traversal:

Here as we can see, when we try to input any path that the student does not have access to, it does not get followed up. Thus, our website is already defended.



2 Show that all the relations and their constraints, finalized after the second feedback, are present and valid as per the ER diagram constructed in Assignment 1.



After the two feedbacks, all the relations of the tables on our website are valid as per the updated ER diagram in assignment 2.

D. Contributions

Name	Contribution
Divyanshu Pandey	-Took the hostel office's feedback and suggested website changes accordingly.
Hirva Patel	Worked on attack and defence system and created 4 such attacks with its defence mechanism. Created the signup page and helped implement google authentication.
Harshi Chandrafari	Worked on the website's frontend and made changes after the first and second feedback. Contributed in the write-up of the document.
Disha Chopra	-Worked on implementing the Google authentication system and rectified errors and bugs throughout.
Siddhi Rajpurohit	-Helped make changes in the front end by adding tables and norms. Contributed in the final documentation. ‘
Shubham More	-Contributed to the final documentation and helped with the feedback.
Dhruv Gupta	-Worked on the locking mechanism and helped implement changes after receiving feedback.