Hostel Room Allocation System

Documentation

Mohith Kumar Thummaluru - B180299CS
Avinash Samudrala - B180409CS
Rahul Kumawat - B180635CS
Tushar Kumar Patni - B180122CS
Ritik Gautam - B180630CS

Final Draft **Date : 26/12/2020**

Table of Contents

1.Modules **3**

2.Code Description : **4**

Signup.php : 4
Index.php : 5
Student_main.php : 5
Notifications.php : 7
Profile.php : 7
Edit.php : 8
Admin_login.php : 8
Admin_land.php : 8
Add_admins.php : 9
Admin_list.php : 9
Admin_feed.php : 9
Update_hostels.php : 9

3. Steps to execute : **10**

1.Modules

This section gives functional requirements that apply to the HRAS.
These are sub modules in this phase.

- Administrative Module
- User Module
- Application Module

The Functionalities are as follows:

- Administrative Module

  The Administrator can:
  1. Allot different students in different hostels.
  2. Vacate the students from the hostels.
  3. Add new employees.
  4. Can clear student’s complaints.
  5. Can add new hostels and allocate admin to it.
  6. Can post announcements.

- User Module

  The user can:
  1. Can submit the application form.
  2. Can view the notice board.
  3. Can choose roommates.
  4. Can raise complaints.
  5. Get notices regarding his/her complaints and room requests.

2. Code Description :

Connect to Database using the command :

```
$connection=new mysqli('localhost','root','','hostel');
```

Execute the insert,update,delete operations(Data Manipulation Language) using the commands :

```
$sql=$connection->prepare("query");
$sql->execute();
```

Execute the query operations (Data Query Language) using the command :

```
mysqli_query("query");
```

Data between the php pages is transferred using the Session variables.

A. Signup.php :

Fresh Users(students) can register to web-app using this page.
After fetching all the details from the user using the "POST" method of Form we check for the given mail and student roll against the existing mail's and student roll's in our database so that duplicate users are prevented.
SQL's used:

`SELECT * FROM STUDENTS WHERE Student_ID='$roll'` to check for existing roll numbers.
`SELECT * FROM STUDENTS WHERE mail='$mail'` to check for existing emails.

If these two sql's return an empty table then we can proceed assuming that he is a fresh user who is signing up for the first time.And we will insert his details into our student's table of databases.
SQL used to insert student details into database:

```
INSERT INTO STUDENTS
(full_name,PhoneNo,Student_ID,DOB,mail,password,gender,State
)VALUES('$name','$phn','$roll','$dob','$email','$password','
$gender','$state')'
```

After this you will be redirected to index.php where you can login.
"PHP header" is used to redirect to another page.

B. Index.php :

Here you can login either as a student or admin each having their own credentials.Students after signing up for the webapp can login into their account from here using their email and password they choose during the process of registering.

The email and password entered by the user were matched against the existing emails and passwords in our database. If the email and password pair matches then the user gets redirected to student_main.php which is the landing/welcome page for all users.

SQL used : `SELECT * FROM STUDENTS WHERE mail='$mail' AND password='$password'`

C. Student_main.php :

If the admin has posted an new hostel update then all students will get an option to choose their hostels and roommates for upcoming year.We will check in notices table if there is an hostel update or not(hostel update is identified uniquely when student_id=0 in notices table which means this notice is common to all students in all hostels)

SQL used : `SELECT * FROM NOTICES WHERE student_id='0'`

Here hostels available for users depends on his batch which is year of study(which is calculated using (current year)-roll[1:3])
Eg: for student with roll b180409cs : current year=20 ,roll[1:3]=18,
So batch =20-18=2
If the current month >7 (july) then batch =batch+1,so batch=3 which is 3rd year.
Here the batch of the hostel is decided by the main admin.

SQL used : `SELECT * FROM HOSTELS WHERE Batch='$batch'`

User can send a request to his friend who is studying in the same year as the user.
The list of available students is made visible to the user.

SQL’s used :

a. `SELECT * FROM STUDENTS WHERE Student_ID LIKE '_$stu_year%'`
b. `"INSERT INTO room_requests(sender_id,receiver_id) VALUES('$student_id','$roomie_id')`

Before inserting the request into the room_requests table we will check whether the receiver to which the user is sending the request is paired up or not. If paired up then we can’t send a request else we can proceed to insert the request into room_requests table.

Sql used: `SELECT * FROM STUDENTS WHERE Student_ID='$receiver_id'"`

If the above sql returns a table with null “Roommate-ID” column then the receiver is not paired else we can consider him as paired.

Hostel and Room-allocation were done based on the vacancies of the room. If the hostel has vacant room then we allot a room to the user in the hostel with room_no=(total_rooms-vacant_rooms)+1

If a user has accepted your request then he will be allocated the same room and hostel as your’s.

After allocating the room and hostel to a user we need to update the vacancies in ROOM and HOSTELS tables in the database.

After the hostel allocation is done the admin_ID in the student’s table gets updated by the admin-id of the admin who is in charge of the user’s hostel.

SQL’s used :

a. `UPDATE STUDENTS SET Hostel_ID='$hos_id',Room_No='$room_no' WHERE Student_ID='$your_id';`
b. `$temp=$hostel['Vacant_rooms']-1;`
   `UPDATE HOSTELS SET Vacant_rooms='$temp' WHERE Hostel_ID='$hos_id'`

Here all the room-requests which a user received were also made visible so that they can accept or reject a room request. Initially when a room request is sent then the flag of the room_requests table is made ‘0’.When the receiver accepts the request then the flag is updated to ‘1’ and if the user rejects the flag will be updated to ‘-1’. Notifications are sent to the sender whether the request is accepted or rejected is done by using the flags.

SQL’s used:

a. `UPDATE room_requests SET flag='1' WHERE sender_id='$mate_id' AND receiver_id='$rol_no';` (accept)
b. `UPDATE room_requests SET flag='-1' WHERE sender_id='$mate_id' AND receiver_id='$rol_no'";`(reject)

Updating Admin_id for students after allocating a hostel.

c. `UPDATE STUDENTS SET Admin_ID=(SELECT Admin_ID FROM HOSTELS WHERE Hostel_ID='$hostel_id') WHERE Student_ID='$rol_no' ;`

Here users can also send queries or related complaints to the admins who are supervising his hostel using the complaint-box.

SQL used:`"INSERT INTO COMPLAINTS(Complaint_Status,Student_ID,Subject,Admin_ID,Hostel_ID,room_no) VALUES('0','$roll','$msg','$Admin','$h_id','$room_no')"`

D. Notifications.php :

All the notices from the NOTICES table which are sent to the user,complaint status of the complaint claimed by the user to admin ,feed posted by the admin,and the room requests status are made visible in this page . Users can clear all their notifications to free space in the database except the feed notification sent by admin which can be deleted only by the admins.

SQL's used:

a. `"SELECT * FROM room_requests WHERE sender_id='$rol_no' AND flag!='0'"`
b. `"SELECT * FROM NOTICES WHERE student_id='$rol_no' OR student_id='1'`
c. `"DELETE FROM NOTICES WHERE Notice_ID='$n_id'";`
   `(student_id='0' notice from admin to all the user ,student_id='1' notice from admin to the users who are under his supervision,student_id='roll' notice to user whose id='roll' )`
d. `"DELETE FROM room_requests WHERE sender_id='$rol_no' AND receiver_id='$s_id'";`

E. Profile.php :

All the details regarding the user are available in this page.

SQL used:`SELECT * FROM STUDENTS WHERE Student_ID='$r_id'`

($r_id is being fetched from the session variable from previous page)

F. Edit.php :

All the personal details of the user except mail and student_id can be edited in this page.

SQL used:`"UPDATE STUDENTS SET gender='$gender',full_name='$name',DOB='$dob',PhoneNo='$phn',State='$state' WHERE Student_ID='$roll'"`

G. Admin_login.php :

Admins can login into their account using their credentials which were given my the main admin. The mail and password entered by the user were matched against mail and password pair in Admin's table of database. If there is a match then you will have access to admin_land.php

SQL's used : `SELECT * FROM ADMIN WHERE mail='$mail' AND password='$password'`

H. Admin_land.php :

This is the landing page of admins. Here admins can see the details of their peer admins and the hostels in which they are in-charge.Also all the complaints from the students of the hostel which is under his supervision are also visible here.

Admins can Accept,clear and reject the complaints in the complaint box so the unnecessary complaints can be rejected.

SQL's used :

a. `"SELECT * FROM COMPLAINTS WHERE  Admin_ID='$ad_id' AND Complaint_Status>'-1'"` (complaint_status='-1' for rejected complaints,complaint_status='1' for accepted complaints and complaint_status='0' for pending complaints)
b. `"UPDATE  COMPLAINTS SET Complaint_Status='1' WHERE Complaint_No='$c_no'`(accepted complaint)
c. `"DELETE FROM COMPLAINTS WHEREComplaint_No='$c_no'"`(cleared complaint);
d. `"SELECT * FROM Admin"` (to show details of all admins)

I. Add_admins.php :

This page is available only for main admin where he can add new admins to this the web-app.

SQL used : `INSERT INTO Admin(Fname,Lname,Mail,password) VALUES ('$fname','$lname','$mail','$password')";`

J. Admin_list.php :

All the students details who were allocated to the hostel which is being supervised by corresponding admin are visible here.

SQL used: `$query="SELECT * FROM STUDENTS WHERE Admin_ID='$ad_id'";($ad_id is the admin id of currently logged in admin)`

K. Admin_feed.php :

Admin can send daily updates and feeds to the corresponding hostlers. Students cannot clear the feed sent by the admin.Only admin who sent that feed has privileges to delete a particular feed .

All the previously posted updates and feeds are also visible in this page .

SQL's used :

```
a. "INSERT INTO NOTICES(Subject,Admin_ID,student_id)
   VALUES('$feed','$ad_id','1')"
b. "SELECT * FROM NOTICES WHERE Admin_ID='$ad_id' AND
   student_id='1'"
c. "DELETE FROM NOTICES WHERE Notice_ID='$notice_id'"
   ($ad_id is the admin id of currently logged in admin)
```

L. Update_hostels.php :

This page is accessible only to main admin where he can add new hostels and allocate an admin to that hostel.

SQL used :`INSERT INTO HOSTELS(Hostel_name,No_of_rooms,Vacant_rooms,Landmark,Gender_flag,Admin_ID,Batch)VALUES('$h_name','$h_rooms','$h_rooms','$h_landmark','$h_gender','$h_admin','$h_batch')"`

Here the main admin will also have an option to send new hostel updates so that students can choose their hostels and roomies for the upcoming year. He will also have an option to remove hostel updates so that he can close the allocation portal after a certain deadline before which all students have to choose their hostels and roommates.

After sending the room requests all the hostels,room notices,complaints,roommates,allocated rooms information present in our current database should be cleared so that it again updates the hostel vacancies and also students can choose their new hostel and the whole process repeats again.

SQL's used :

A. After sending room update :

a. `”UPDATE STUDENTS SET Roomate_ID=NULL,Hostel_ID=NULL,Admin_ID=NULL,Room_No=NULL WHERE 1";`
b. `$q="UPDATE ROOM SET Vacancies='2' WHERE 1";`
c. `$q="DELETE FROM room_requests WHERE 1";`
d. `$q="DELETE FROM NOTICES WHERE 1";`
e. `$q="DELETE FROM COMPLAINTS WHERE 1";`
f. `"ALTER TABLE room_requests AUTO_INCREMENT=1";`
g. `"ALTER TABLE NOTICES AUTO_INCREMENT=1";`
h. `"ALTER TABLE COMPLAINTS AUTO_INCREMENT=1");`
i. `("UPDATE HOSTELS SET Vacant_rooms=No_of_rooms");`
j. `"INSERT INTO NOTICES(Subject,Admin_ID,student_id) VALUES('New Hostel Update','1','0')";`

B. After Removing room update :

a. `DELETE FROM NOTICES WHERE student_id='0'";`

3. Steps to execute :

I. Firstly,install XAMPP software on PC.
II. Then open XAMPP control-panel and start apache and mysql servers.
III. Open your web browser and go to http://localhost/phpmyadmin/
IV. Create a new database with the name “hostel” by clicking the New on the left panel on the phpmyadmin page and navigate to the hostel database where you can find an import option.
V. Now unzip dbms.zip file
VI. There you can find hostel.sql, now import this hostel.sql into your hostel database which you have created earlier
VII. Now copy the paste the unzipped dbms folder into the htdocs folder in XAMPP folder.This is the folder when you have installed your XAMPP app. (You can probably find this folder in Windows C)
VIII. Since you have already started an apache server from XAMPP control-panel directly, open the browser and go to http://localhost/dbms/pages/ ,this directly brings up index.php from our files.
IX. There you have an option to login as a student or else login as admin.If you want to login as a student you can register to this web-app by clicking “New here? Create one” link which is located just above the login button.
X. You can also use credentials of pre-existing users which you can find in the STUDENTS table of the hostel database.
XI. To login as admin you can use credentials of pre-existing admins in Admin table of hostel database (or) you can the credentials of main admin which are: Email: admin1@gmail.com Password: admin1
XII. You can add new admins into the database when you login as main admin.

This is how you can start the web-application. For further guidelines refer to the user manual in report.