

# Final Report (Hostel Management System)

**Course Code:** CS254

**Semester:** B. Tech 4<sup>th</sup> Sem

**Academic Year:** 2020-21

Mr. Sharath Yaji

**Course Title:** DBMS Lab

**Section:** S1

**Course Instructor:** Dr. Annappa B and

## **Team Members:**

1. Akanksha More, 191CS106, +91 9920694982, akanksha.191cs106@nitk.edu.in
2. Mansi Singh, 191CS135, +91 9993063395, mansi.191cs135@nitk.edu.in
3. Deepta Devkota, 191CS117, +91 7349641338, deeptadevkota.191cs117@nitk.edu.in

## 1 Abstract

### **Brief Description:**

We will be building a self-sufficient hostel management system. The hostel's structure would be quite similar to that of NITK, we will have separate blocks for boys and girls, the students belonging to the same batch will be sharing a room if needed. Each hostel block will be having four sub-blocks

- 1st sub-block: Triple sharing rooms for 1st years.
- 2nd sub-block: Double sharing rooms for 2nd year.
- 3rd and 4th sub-blocks: Single occupancy rooms for 3rd, 4th years students respectively

When a newly enrolled student register and logs into the hostel portal they will be able to go for room allocation, the allocation will happen serially for the 1st year students i.e they won't be given the option to choose their roommates or rooms, the rooms will be filled unless the maximum capacity of three is reached.

As the 2nd years have a double occupancy room in their block hence, they have to register in pairs. Those without a partner will be kept in the waiting table and pairs will be formed with the ones in the waiting table in that order.

The 3rd, 4th year students will get to have single occupancy rooms on a first come first serve

basis.

The hostel management system will also have a provision to register the wardens. Wardens will have the following authorities:

1. Enable/Disable allocation and thus set the allocation period.
2. To view the hostel related complaints from the students.

Thus, the hostel management system will help to run the room allotment and maintenance issues effectively by saving time and making the process more efficient.

## 2 Introduction

A student has to first register in the portal. A student cannot register multiple times. Once the registration is successfully completed, he/she can log in to the portal. As per the student's year, they will be given an option for room booking. The student cannot go for room allocation if they are already allocated a room. Second year students who currently do not have a room-mate yet, will opt for a room individually such student data will be kept at the waiting table and they will be paired in that order. A student can generate several complaints. However, they won't be able to see the complaints generated by other students. Once the student has accessed the portal, he/she can log out if needed.

A second year student who applies with a partner where the partner is already has name in waiting table, then such registration will not be valid. A student can get the contact details of the warden of his/her respective block. Also, every student can get contact details of all the wardens from the main hostel dashboard page( in contact us page).

Every warden has to be allocated exactly one block, and each block will have at most one warden. We have created a registration form for the wardens where the warden has to fill in the required fields. A warden cannot register multiple times. Once the registration is complete, the warden can log in to the portal and view the complaints generated by the hostel students and the data of the student residing in the warden's hostel block. The warden can log out if needed.

There are separate blocks for both boys and girls. Also, students from same year reside in the same hostel block. Hence, total 8 blocks are available in the hostel.

A superuser will only have the privilege, to add data in the buildings and block tables. The superuser manages this from the admin portal.

We are giving the wardens the authority to set the room allocation period for the hostel portal. This is done by changing the changing the value of is\_room\_allocation\_set in the room\_allocation table by the wardens.

## A layout of the structure of the hostel:

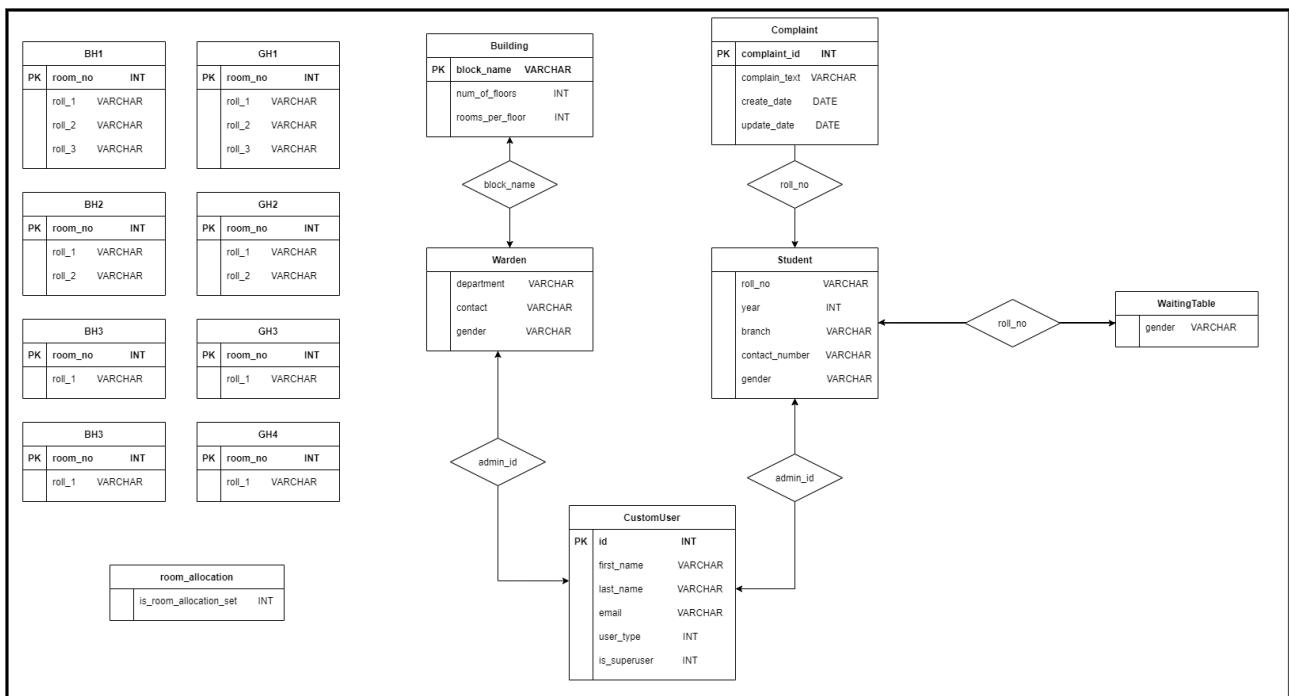
### GIRLS HOSTEL:

Block Name	No. of Floors	No. of Rooms/per	No. of students per room	Total students
GH1	4	25	3	300
GH2	3	50	2	300
GH3	4	75	1	300
GH4	4	75	1	300

### BOYS HOSTEL:

Block Name	No. of Floors	No. of Rooms/per	No. of students per room	Total students
BH1	4	50	3	600
BH2	4	75	2	600
BH3	6	100	1	600
BH4	6	100	1	600

### 3 ER Diagram



## 4 Source Code

```
def room_register_2(request):  
    roll_1 = request.POST.get('roll_1'), roll_2 = request.POST.get('roll_2')  
    name_1 = request.POST.get('name_1'), name_2 = request.POST.get('name_2')  
    year = request.POST.get('year'), gender = request.POST.get('gender')  
    if gender == "Female":  
        if GH2.objects.filter(roll_1=roll_1).exists() or  
            GH2.objects.filter(roll_2=roll_1).exists():  
                return render(request, 'studentDashboard.html', {})  
        if GH2.objects.filter(roll_1=roll_2).exists() or  
            GH2.objects.filter(roll_2=roll_2).exists():  
                return render(request, 'studentDashboard.html', {})  
        latest_count = GH2.objects.count()  
        mod = latest_count % 50  
        quotient = int(latest_count / 50)  
        room_no = 100*(quotient+1) + mod + 1  
        room = GH2(room_no=room_no, roll_1=roll_1, roll_2=roll_2), room.save()  
    else:  
        if BH2.objects.filter(roll_1=roll_1).exists() or  
            BH2.objects.filter(roll_2=roll_1).exists():  
                return render(request, 'studentDashboard.html', {})  
        if BH2.objects.filter(roll_1=roll_2).exists() or  
            BH2.objects.filter(roll_2=roll_2).exists():  
                return render(request, 'studentDashboard.html', {})  
        latest_count = BH2.objects.count()  
        mod = latest_count % 50  
        quotient = int(latest_count / 50)  
        room_no = 100*(quotient+1) + mod + 1  
        room = BH2(room_no=room_no, roll_1=roll_1, roll_2=roll_2), room.save()  
    return render(request, 'studentDashboard.html', {})  
  
def room_register_1_3_4(request):  
    roll_no = request.POST.get('roll_no'), name = request.POST.get('name')  
    year = request.POST.get('year'), gender = request.POST.get('gender')  
    contact_number = request.POST.get('contact')  
    if Student.objects.filter(roll_no=roll_no).exists() == False:  
        return render(request, 'studentDashboard.html', {})  
    if int(year) == 1:  
        if gender == "Female":
```

```

if GH1.objects.filter(roll_1=roll_no).exists() or GH1.objects.
    filter(roll_2=roll_no).exists() or GH1.objects.filter(roll_3=
    roll_no).exists():
    return render(request, 'studentDashboard.html', {})

latest_room = GH1.objects.order_by('room_no').last()
print(latest_room)

if latest_room == None:
    new_room = GH1(room_no=101, roll_1=roll_no),new_room.save()

elif latest_room.roll_2 == None:
    latest_room.roll_2 = roll_no,latest_room.save()

elif latest_room.roll_3 == None:
    latest_room.roll_3 = roll_no,latest_room.save()

else:
    latest_count = GH1.objects.count(),mod = latest_count % 25
    quotient = int(latest_count / 25)
    room_no = 100*(quotient+1) + mod + 1
    room = GH1(room_no=room_no, roll_1=roll_no)
    room.save()

else:
    if BH1.objects.filter(roll_1=roll_no).exists() or BH1.objects.
        filter(roll_2=roll_no).exists() or BH1.objects.filter(roll_3=
        roll_no).exists():
        return render(request, 'studentDashboard.html', {})

    latest_room = BH1.objects.order_by('room_no').last()

    if latest_room == None:
        new_room = BH1(room_no=101, roll_1=roll_no),new_room.save()

    elif latest_room.roll_2 == None:
        latest_room.roll_2 = roll_no, latest_room.save()

    elif latest_room.roll_3 == None:
        latest_room.roll_3 = roll_no,latest_room.save()

    else:
        latest_count = BH1.objects.count()
        mod = latest_count % 25
        quotient = int(latest_count / 25)
        room_no = 100*(quotient+1) + mod + 1
        room = BH1(room_no=room_no, roll_1=roll_no),room.save()

    elif int(year) == 3 or int(year) == 4:
        if gender == "Female":
            if int(year) == 3 and GH3.objects.filter(roll_1=roll_no).exists
                ():


```

```

        return render(request , 'studentDashboard.html' , {})

    elif int(year) == 4 and GH4.objects.filter(roll_1=roll_no).
        exists():

        return render(request , 'studentDashboard.html' , {})

    else:

        if int(year) == 3 and BH3.objects.filter(roll_1=roll_no).exists
           ():

            return render(request , 'studentDashboard.html' , {})

        elif int(year) == 4 and BH4.objects.filter(roll_1=roll_no).
            exists():

            return render(request , 'studentDashboard.html' , {})

    if gender == "Female":

        if int(year) == 3:
            latest_count = GH3.objects.count()

        else:
            latest_count = GH4.objects.count()

        mod = latest_count % 75

        quotient = int(latest_count / 75)

        room_no = 100*(quotient+1) + mod + 1

        if int(year) == 3:
            room = GH3(room_no=room_no , roll_1=roll_no)

        else:
            room = GH4(room_no=room_no , roll_1=roll_no)

        room.save()

    else:

        if int(year) == 3:
            latest_count = BH3.objects.count()

        else:
            latest_count = BH4.objects.count()

        mod = latest_count % 75

        quotient = int(latest_count / 75)

        room_no = 100*(quotient+1) + mod + 1

        if student.year == 3:
            room = BH3(room_no=room_no , roll_1=roll_no)

        else:
            room = BH4(room_no=room_no , roll_1=roll_no)

    return render(request , 'studentDashboard.html' , {})

```

[Link to complete source code in GitHub](#)

## 5 Results

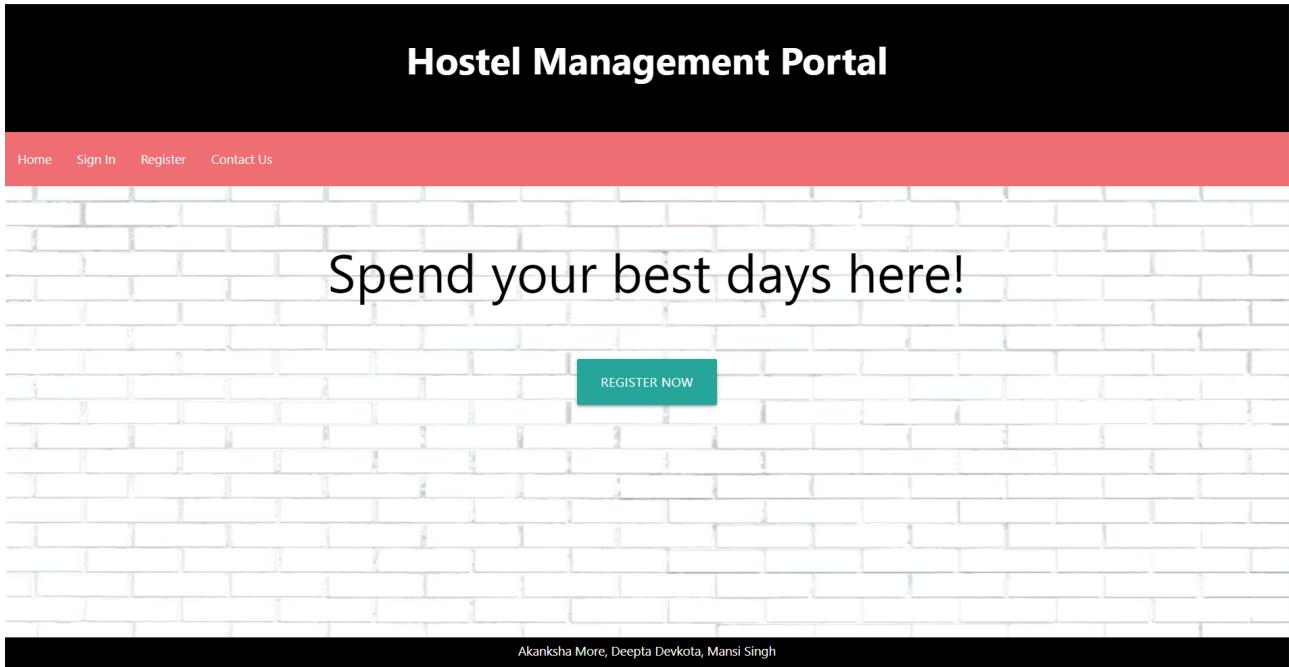


Figure 1: Home page of the hostel portal

The image shows the student registration form page of the Hostel Management Portal. At the top, there is a black header bar with the text "Hostel Management Portal" in white. Below the header is a yellow navigation bar containing links for "Home", "Sign In", "Register", and "Contact Us". The main content area has two orange buttons at the top: "STUDENT REGISTRATION" on the left and "WARDEN REGISTRATION" on the right. The form consists of several input fields with placeholder text and a "SUBMIT" button at the bottom. The fields are as follows:

First Name Deeptha	Last Name Devkota
Roll number 191CS117	Branch CSE
Contact Number 7349641338	Year 2
Password deeptha	Gender Female
Email deepthadevkota21@gmail.com	
<b>SUBMIT</b>	

Figure 2: Registration form page for the students



Figure 3: Student Login

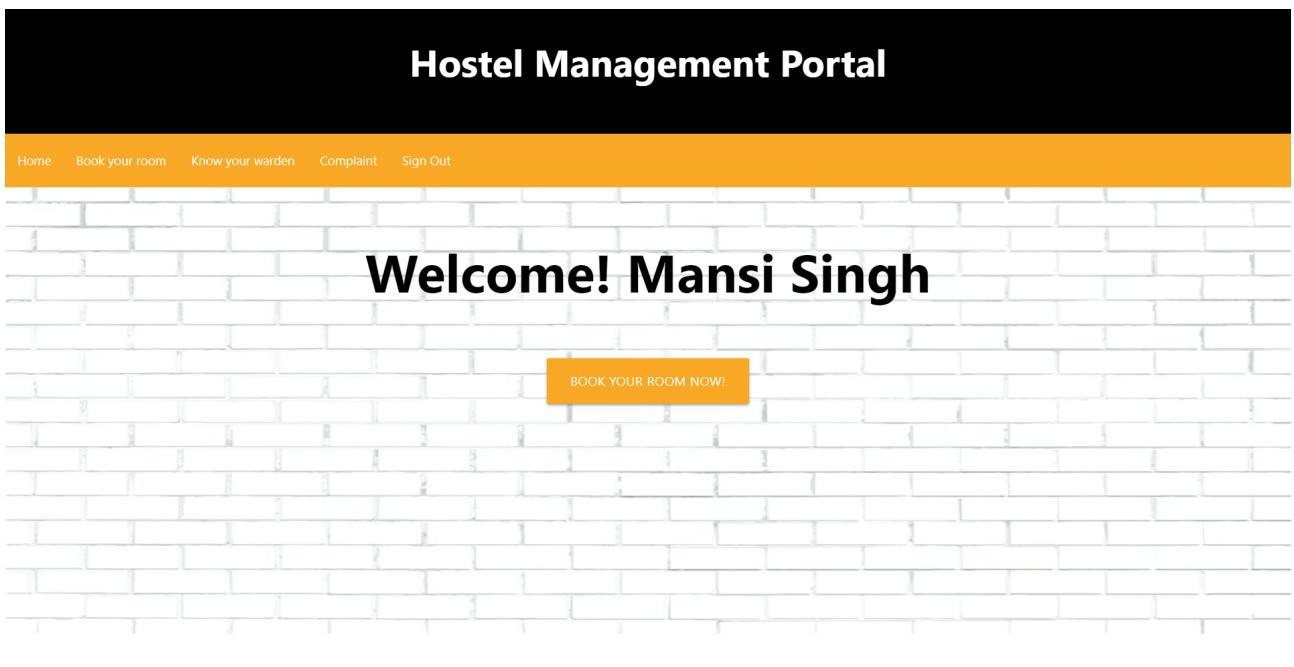


Figure 4: Student Dashboard

# Hostel Management Portal

Home Book your room Know your warden Complaint Sign Out

---

Name  
Mansi Singh

Roll Number  
191CS135

Contact Number  
9993063395

Gender  
Female

**SUBMIT**

Figure 5: Room registration for 1st, 3rd and 4th year students

# Hostel Management Portal

Home Sign In Register Staff Contact Us

---

Name of Student 1  
Laksh Sethi

Roll Number of Student 1  
9876

Name of Student 2  
Priyansh

Roll Number of Student 2  
1234

Year  
2

Gender  
Male

**SUBMIT**

Figure 6: Room registration for 2nd year students

# Hostel Management Portal

Home Book your room Know your warden Complaint Sign Out

Roll No  
191CS135

Complain  
Leakage water from taps in GH1, first floor

SUBMIT

Akanksha More, Deeptha Devkota, Mansi Singh

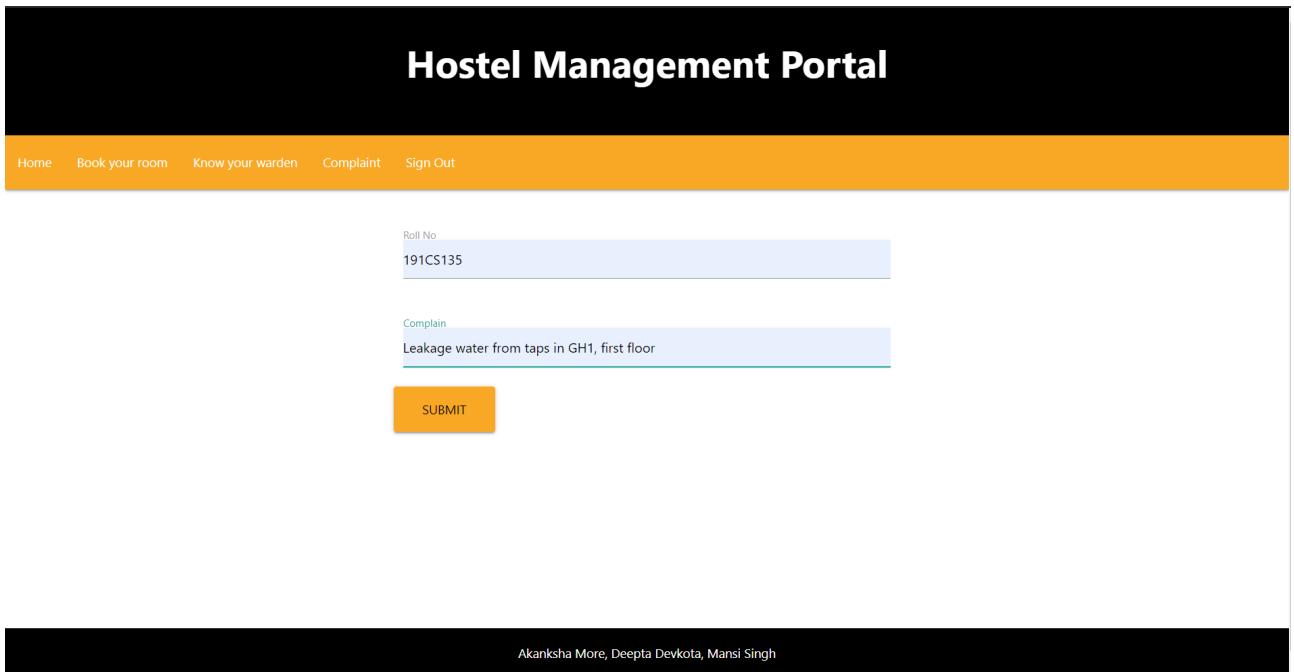


Figure 7: Complain Form for students

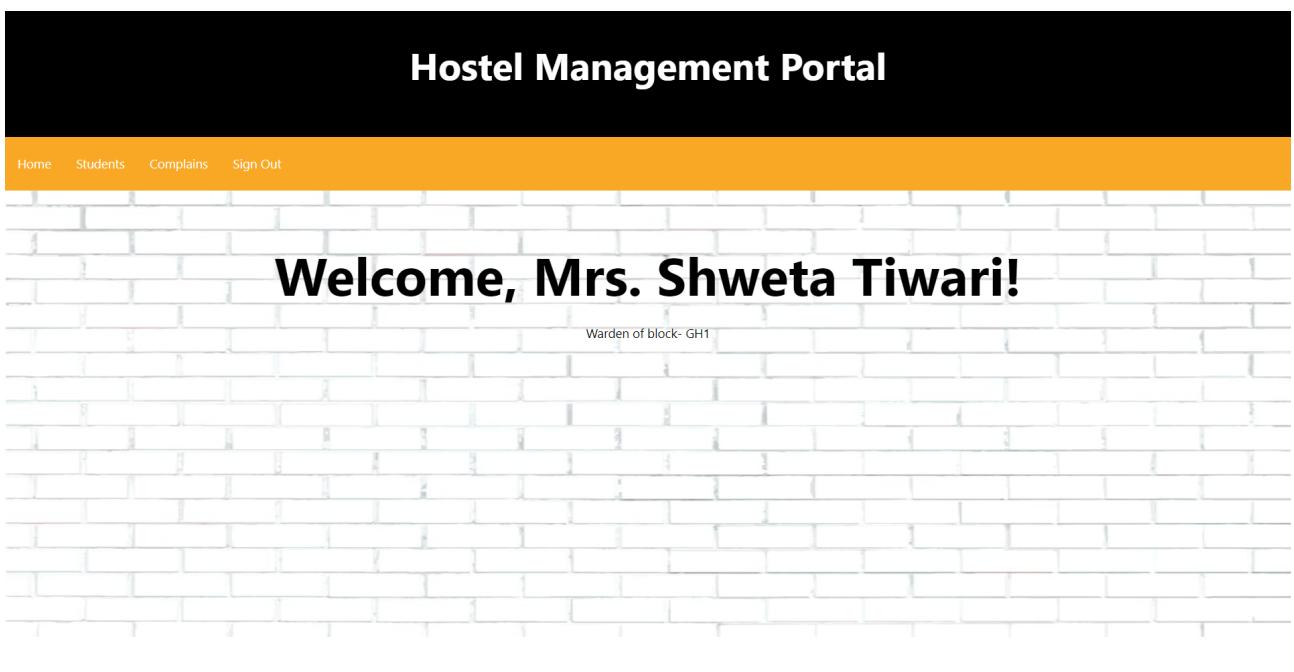


Figure 8: Warden Dashboard

Hostel Management Portal				
Home	Students	Complains	Sign Out	
<b>List of students in block- GH1</b>				
<hr/>				
Roll no	Branch	Gender	Year	Contact Number
191CS135	Computer Science	Female	1	09993063395
191cs111	Computer Science	Female	1	9876543210

Figure 9: Student list viewed by the warden

Hostel Management Portal - Complain List				
Complain ID	Complain by	Complain	Created On	
1	191CS135	Leakage water from taps in GH1, first floor	March 28, 2021	

Akanksha More, Deepa Devkota, Mansi Singh

Figure 10: Complaint List of the students viewed by the wardens

<b>Hostel Management Portal</b>				
<a href="#">Home</a> <a href="#">Sign In</a> <a href="#">Register</a> <a href="#">Contact Us</a>				
Name	Email	Contact	Gender	Department
Mrs. Shweta Tiwari	shweta@gmail.com	1234567890	Female	Computer Science
Mrs. Anjali Diwedi	anjali@gmail.com	2345678901	Female	Civil
Mrs. Priyanka Jain	priyanka@gmail.com	3456789012	Female	Mining
Mrs. Komal Pandey	komal@gmail.com	4567890123	Female	Metallurgy
Mr. Ankit Sethi	ankit@gmail.com	5678901234	Male	Chemical
Mr. Jay Soni	jay@gmail.com	6789012345	Male	Mining
Mr. Amit Kumar	amit@gmail.com	7890123456	Male	Computer Science
Mr. Krishna Prasad	krishna@gmail.com	8901234567	Male	Chemical

Akanksha More, Deeptha Devkota, Mansi Singh

Figure 11: Contact Us page

<b>Hostel Management Portal</b>				
<a href="#">Home</a> <a href="#">Book your room</a> <a href="#">Know your warden</a> <a href="#">Complaint</a> <a href="#">Sign Out</a>				
<b>Warden of block GH1</b>				
Name	Email	Contact	Gender	Department
Mrs. Shweta Tiwari	shweta@gmail.com	1234567890	Female	Computer Science

Figure 12: Know your warden Page for the student

MySQL Workbench

Local instance MySQL80 x Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS: Filter objects

hostel\_management\_warden

1 • SELECT \* FROM hostel\_management\_system.hostel\_management\_warden;

	id	department	contact	gender	admin_id	block_name_id
▶	1	Computer Science	1234567890	Female	2	GH1
▶	2	Civil	2345678901	Female	3	GH2
▶	3	Mining	3456789012	Female	4	GH3
▶	4	Metallurgy	4567890123	Female	5	GH4
▶	5	Chemical	5678901234	Male	6	BH1
▶	6	Mining	6789012345	Male	7	BH2
▶	7	Computer Science	7890123456	Male	8	BH3
▶	8	Chemical	8901234567	Male	9	BH4
▶	NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types |

Information: hostel\_management\_warden 1

Schema: hostel\_management\_system

Action Output:

#	Time	Action
1	21:20:04	SELECT * FROM hostel_management_system.hostel_management_warden LIMIT 0, 1000

Message: 8 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

Figure 13: Database scheme of warden

MySQL Workbench

Local instance MySQL80 x Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS: Filter objects

hostel\_management\_complaint

1 • SELECT \* FROM hostel\_management\_system.hostel\_management\_complaint;

	complaint_id	complainText	create_date	update_date	roll_no_id
▶	1	Leakage water from taps in GH1, first floor	2021-03-28	2021-03-28	1
▶	NULL	NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types |

Information: hostel\_management\_complaint 1

Schema: hostel\_management\_system

Action Output:

#	Time	Action
1	21:20:04	SELECT * FROM hostel_management_system.hostel_management_warden LIMIT 0, 1000
2	21:20:14	SELECT * FROM hostel_management_system.hostel_management_building LIMIT 0, 1000
3	21:23:14	SELECT * FROM hostel_management_system.hostel_management_complaint LIMIT 0, 1000

Message: 8 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

8 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

1 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

Figure 14: Database scheme of complaint

MySQL Workbench - Local instance MySQL80

**Query 1:** hostel\_management\_building

```
1 • SELECT * FROM hostel_management_system.hostel_management_building;
```

block_name	num_of_floors	rooms_per_floor
BH1	4	50
BH2	4	75
BH3	6	100
BH4	6	100
GH1	4	25
GH2	3	50
GH3	4	75
GH4	4	75
• NULL	NULL	NULL

**Action Output:**

#	Time	Action	Message	Duration / Fetch
1	21:20:04	SELECT * FROM hostel_management_system.hostel_warden LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
2	21:20:14	SELECT * FROM hostel_management_system.hostel_management_building LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec

Figure 15: Database scheme of buildings

MySQL Workbench - Local instance MySQL80

**Query 1:** hostel\_management\_waitingtable

```
1 • SELECT * FROM hostel_management_system.hostel_management_bh1;
```

room_no	roll_1	roll_2
101	9876	1234
• NULL	NULL	NULL

**Action Output:**

#	Time	Action	Message	Duration / Fetch
2	21:20:14	SELECT * FROM hostel_management_system.hostel_warden LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
3	21:23:14	SELECT * FROM hostel_management_system.hostel_management_complaint LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
4	21:45:36	SELECT * FROM hostel_management_system.hostel_management_complaint LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
5	21:48:56	SELECT * FROM hostel_management_system.hostel_management_student LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
6	21:49:24	SELECT * FROM hostel_management_system.hostel_management_waitingtable LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
7	21:52:27	SELECT * FROM hostel_management_system.hostel_management_student LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
8	21:52:41	SELECT * FROM hostel_management_system.hostel_management_customuser LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
9	21:53:16	SELECT * FROM hostel_management_system.hostel_management_bh1 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Figure 16: Database scheme of BH1

MySQL Workbench

Local instance MySQL80 x Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS: Filter objects

hostel\_management\_għ1

SELECT \* FROM hostel\_management\_system.hostel\_management\_għ1;

room_no	roll_1	roll_2	roll_3
101	191CS135	191cs111	NULL
NULL	NULL	NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

għement\_għ1 x

Schema: hostel\_management\_system

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:20:04	SELECT * FROM hostel_management_system.hostel_management_warden LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
2	21:20:14	SELECT * FROM hostel_management_system.hostel_management_building LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
3	21:23:14	SELECT * FROM hostel_management_system.hostel_management_complaint LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
4	21:45:36	SELECT * FROM hostel_management_system.hostel_management_għ1 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Figure 17: Database scheme of GH1

MySQL Workbench

Local instance MySQL80 x Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS: Filter objects

hostel\_management\_waitingtable

SELECT \* FROM hostel\_management\_system.hostel\_management\_waitingtable;

roll_no	gender
191451	Male
NULL	NULL

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

t\_waitingtab... x

Schema: hostel\_management\_system

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:20:04	SELECT * FROM hostel_management_system.hostel_management_warden LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
2	21:20:14	SELECT * FROM hostel_management_system.hostel_management_building LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
3	21:23:14	SELECT * FROM hostel_management_system.hostel_management_complaint LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
4	21:45:36	SELECT * FROM hostel_management_system.hostel_management_għ1 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
5	21:48:56	SELECT * FROM hostel_management_system.hostel_management_student LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
6	21:49:24	SELECT * FROM hostel_management_system.hostel_management_waitingtable LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Figure 18: Database scheme of Waiting Table

## **6 References:**

1. <https://medium.com/nybles/building-a-hostel-managing-system-with-django-d2dc85d9dec2>
2. <https://medium.com/@omaraamir19966/connect-django-with-mysql-database-f946d0f6f9e3>

**\*\*\*\* END \*\*\*\***