# DBMS Project

## Facility Management System

**Team-78**
**Team Members:**
Anjali
Avinash
Nandesh
Sarthak Arora
Sandeep Kumar

**Git Link:** https://github.com/itissandeep98/Facility-Management-System

## WEEK 1

In the first meeting of our group, the team met and got familiar with each other. We discussed various aspects of our projects.
We started brainstorming our ideas and came up with various vivid ones. Some of these ideas included creating a social network for college, criminal records handling, protein database management, etc.
While discussing these projects, all the team members participated actively and gave intellectual inputs to the ideation. Our group maintained the enthusiasm of passionately working on this project and aimed to make it an actual implementable application.
In the second meeting of the first week, we started with shortlisting the ideas and finally decided upon one. We decided upon redesigning an application for “Facility Management System(FMS)”

The scope of our project is restricted to the services provided by FMS in our college. These services include housekeeping, plumbing, AC repairing, various other electrical problems, etc. All these issues are centrally controlled by contacting a particular help center that can be contacted via various means like calling, texting, WhatsApp, and an online portal.

After deciding upon our topics, we started recognizing the stakeholders.
The following were the recognized stakeholders after the first week's discussion:

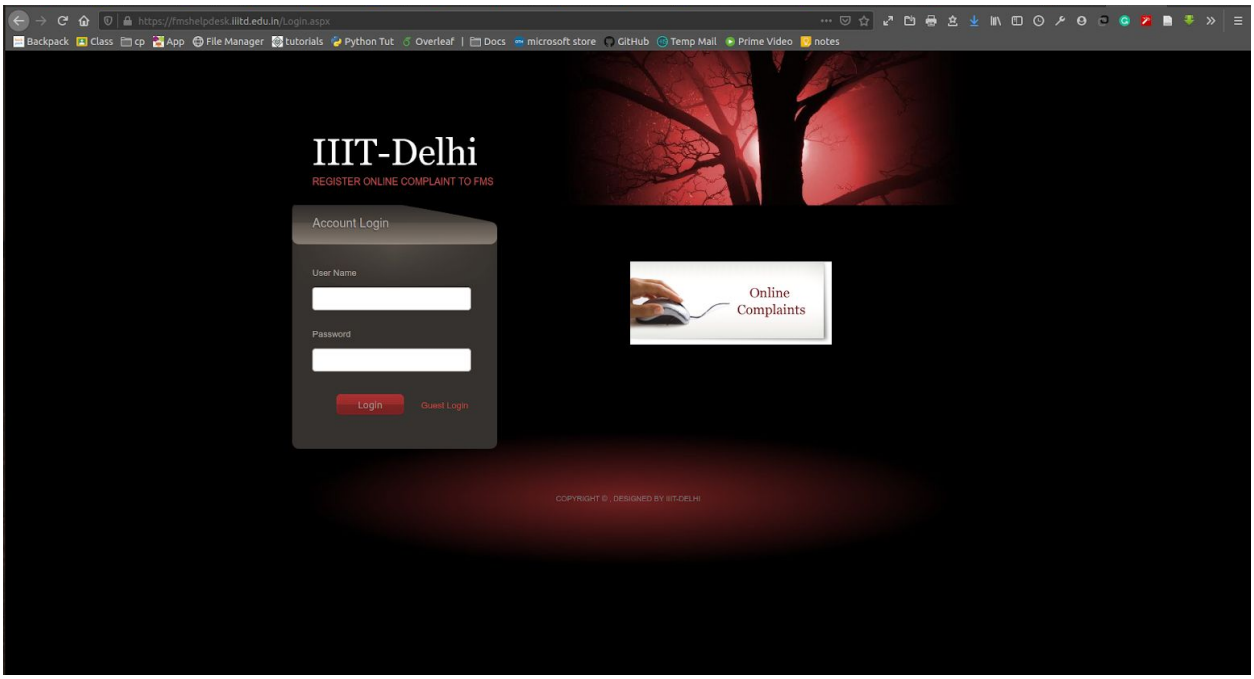
- Students
- Faculty

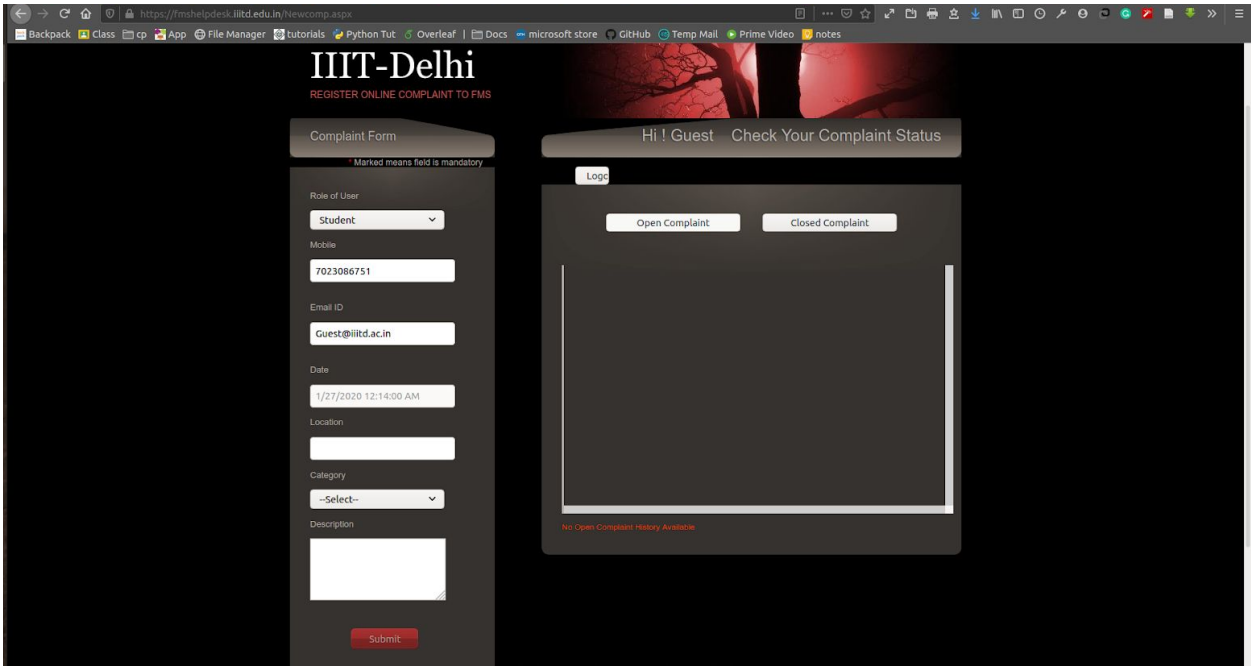
- Individuals from different eateries.
- Cleaners, electricians, mechanics, carpenters, and other skilled laborers.
- Heads at FMS

# WEEK 2

We started conducting surveys and interviews. We even had informal discussions with various people involved.

We started by contacting Mr. Tinku, an FMS in charge of our college. Mr. Tinku manages all the requests and complaints about the FMS by the students, faculty, and eateries. Mr. Tinku told us that most of the people don't use the portal made explicitly for this purpose. During our 20-25 minute visit to Mr. Tinku, he attended more than 6-7 phone calls and responded to a minimum of 5 WhatsApp messages regarding FMS services.






He even told us various drawbacks of the present portal service. Certain drawbacks he listed were:

1. Supervisors don't get a direct SMS after assigning.
2. They have to maintain a separate offline excel database, which is entered manually.
3. There are pretty limited searching options on their platform.
4. There is no facility for editing the details of an employee.

Tinku Sir even made us aware that there was a stakeholder that we were missing all together till now, the supervisors of workers. The FMS heads assign various tasks to the block supervisors, who further assigns the tasks to the workers.

After understanding the whole procedure of how the FMS department is working, we re-finalized our stakeholders:

- Students
- Faculty
- Individuals from different eateries.
- Supervisors
- Cleaners, electricians, mechanics, carpenters, and other skilled laborers. (Indirect stakeholders, they don't use the platform)
- Heads at FMS

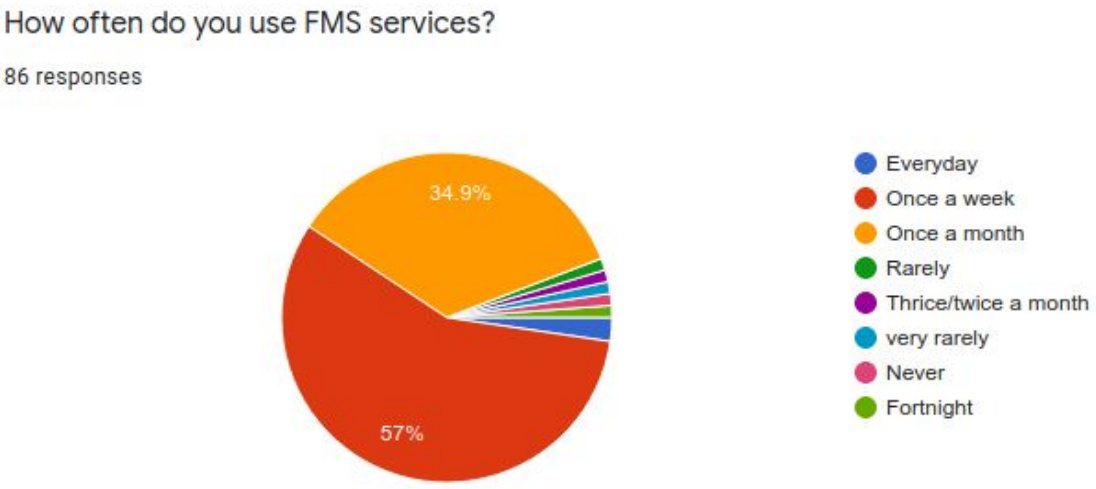
By incorporating the suggested improvements in our portal, we can improve the time utility of the FMS heads in the following manner:

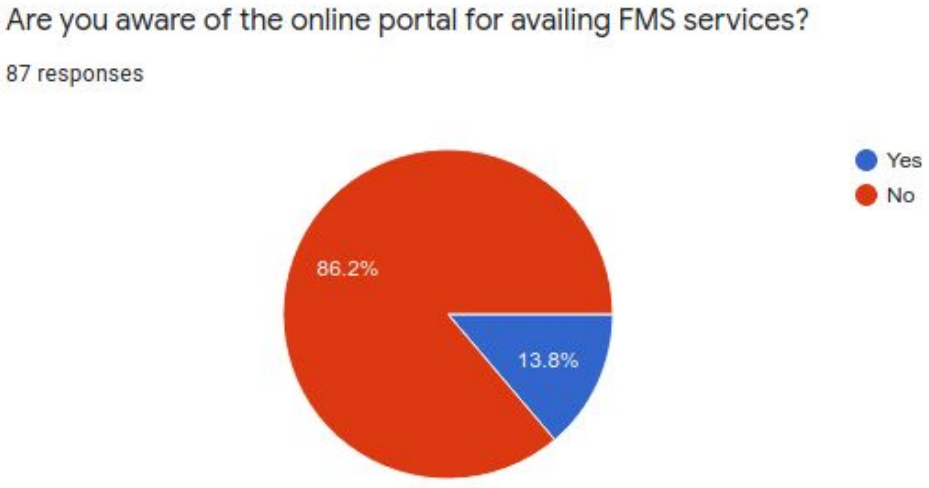
1. They would have to attend fewer phone calls as the majority of requests will be shifted to the mobile application/ portal.
2. They wouldn't have to maintain a separate database.
3. There would be more searching options on the portal.
4. There would be a facility for editing the details of an employee.

Then, we started by floating a survey among students and faculty of our college. The study had a simple questionnaire:

1. How often do you use FMS services?
2. Are you aware of the online portal for availing FMS Services?

The results of this study are available below:




Our study clearly showed that even though the services provided by FMS are used pretty often, most of the students are unaware of the existence of the FMS portal.

By providing a dedicated and improved portal, we'll help the students and faculty in the following ways:

1. They can schedule FMS services.
2. They can keep track of their past usage.
3. They can give a review of the service.
4. They'll get one-touch access to all the services provided by FMS.

5. They can also request for special events that happen around the campus.

Next, we contacted different eateries around the campus, even they were unaware of the existence of the online portal for FMS services.

By providing a dedicated and improved portal, we'll help different eateries in the following ways:

1. They can schedule dustbin pickups.
2. They can ask for emergency pickups.
3. They can give a review of the service.
4. They'll get one-touch access to all the services provided by FMS.
5. They can keep track of their past usage.

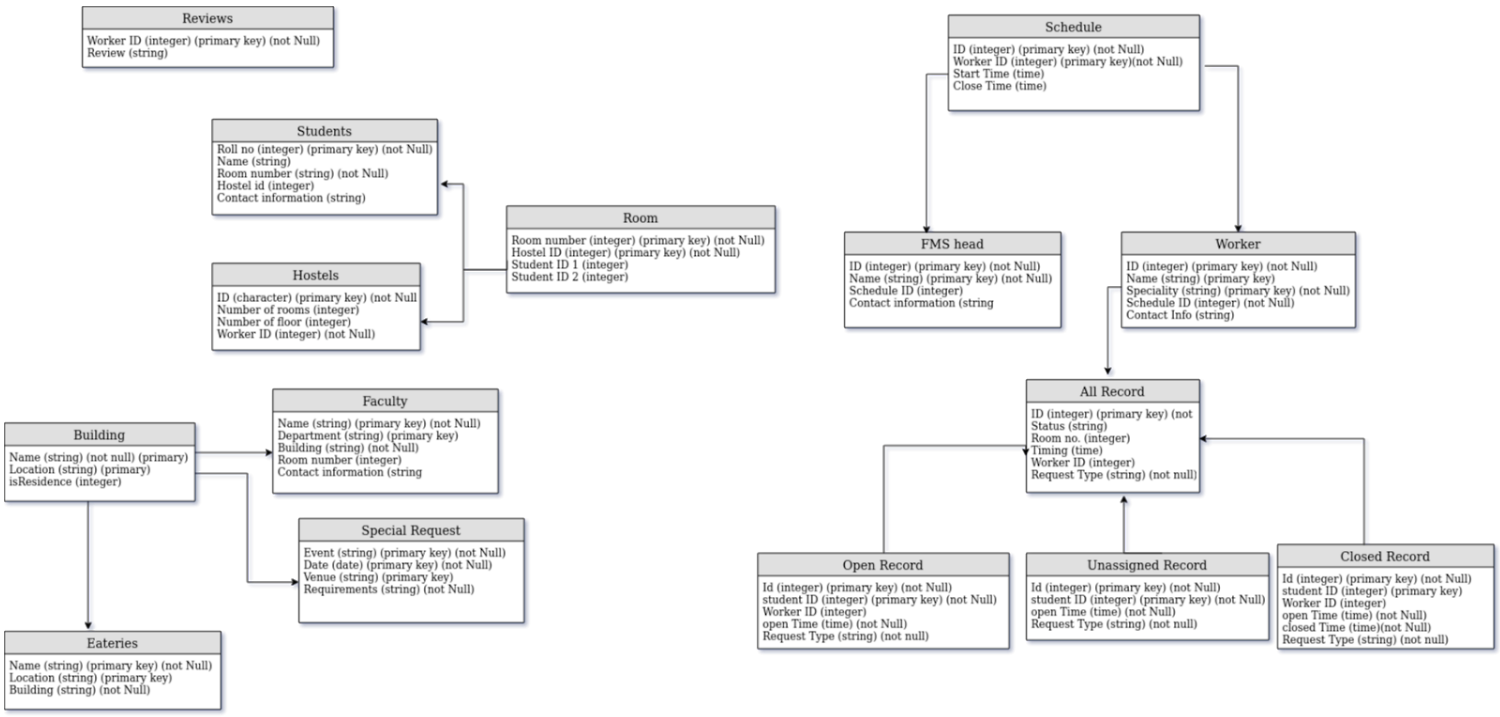
At last, we contacted the supervisors and got to know how they were involved in the whole system. We got to know that yet now, all their communication was on a phone call and WhatsApp. They assigned all the tasks to random workers and, at times, lost contact with attendance and the working schedule of workers.

Through their input in the discussion, we'll help the students and faculty in the following ways:

1. They can keep track of assigned work to workers in real-time.
2. They can keep track of the attendance of the workers.
3. The app will automatically show the workers that are available for work.
4. It will lead to more coordination between the FMS head and the supervisors.
5. They would have to attend fewer phone calls as the majority of the work would be shifted to the portal/app.

# Week 3

### Schema Diagram:




Some of the constraints that were recognized were:

- NOT NULL:

  “Students” table we have used not null in “ID”,” Room no”.,” Name”

- PRIMARY KEY

  “Schedule” table has two columns “ID”, “WorkerID” are primary keys

- FOREIGN KEY

  “WorkerID” in table “Worker” is a foreign key for table “AllRecord”

- CHECK

  In “Worker” table we have used CHECK to make sure that “Speciality” of a worker can only be “cleaner”,” carpenter”,” electrician”,” painter”,” plumber”

  In “Room” table we have used CHECK constraint to make sure that “HostelID” can only be (1,2,3)

# WEEK-4

We started this week by creating tables in our database. All the members were involved in this process and this was done using MySQL Workbench.

All the team members collectively sat to make the tables and populated them.

The following were some commands that were used to create the tables:

```
CREATE TABLE `AllRecord` (
  `ID` int(11) NOT NULL,
  `Status` varchar(10) COLLATE utf8_unicode_ci DEFAULT NULL,
  `RoomNum` int(11) DEFAULT NULL,
  `Timing` time DEFAULT NULL,
  `WorkerID` int(11) DEFAULT NULL,
  `RequestType` varchar(15) COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`ID`),
  UNIQUE KEY `ID_UNIQUE` (`ID`),
  CONSTRAINT FOREIGN KEY (`ID`) REFERENCES `ClosedRecord` (`id`),
  CONSTRAINT FOREIGN KEY (`ID`) REFERENCES `OpenRecord` (`ID`),
  CONSTRAINT FOREIGN KEY (`ID`) REFERENCES `UnassignedRecord` (`id`),
  CONSTRAINT CHECK ((`RequestType` in ('Cleaner', 'Plumber', 'Electrician', 'Carpenter',
'Painter'))),
  CONSTRAINT CHECK ((`Status` in ('Open','Close','Unassigned'))))
```

---

```
CREATE TABLE `Hostels` (
  `ID` int(11) NOT NULL,
  `NumOfRoom` int(11) DEFAULT NULL,
  `NumOfFloor` int(11) DEFAULT NULL,
  `SupervisorID` int(11) NOT NULL,
  PRIMARY KEY (`ID`),
  UNIQUE KEY `ID_UNIQUE` (`ID`),
  KEY `SupervisorID` (`SupervisorID`),
  CONSTRAINT FOREIGN KEY (`SupervisorID`) REFERENCES `Supervisor` (`id`))
```

---

Queries for creating other tables are attached to this document.

The following were some commands that were used to insert data into the tables:

```
INSERT INTO `Worker` VALUES
(1,'Swastik','Cleaner','12345',1),
(2,'Navya','Electrician','23456',2),
```

```
(3,'Nitin','Plumber','34567',3),
(4,'Arka','Carpenter','45678',4),
(5,'Kinshuk','Cleaner','56789',5),
(6,'Rishabh','Painter','98990',2),
(7,'Dhruv','Plumber','88605',4),
(8,'Pratham','Carpenter','93129',1),
(9,'Yash','Electrician','93503',1),
(10,'Naman','Cleaner','55500',3);
```

---

```
INSERT INTO `UnassignedRecord` VALUES
(1,3,'03:00:23','Cleaner'),
(2,4,'12:00:59','Carpenter'),
(3,9,'04:12:36','Electrician'),
(4,5,'05:35:56','Plumber');
```

---

Queries for inserting data into other tables are attached to this document.

**WEEK 6**

This week started by emphasizing on looking at the reviews given by our TA. We looked ahead and wrote the following relational algebraic queries related to our database

Relational algebraic queries

1) To select vacant rooms

```
σStudent1ID=NULL and Student2ID=NULL(room)
σ - select
```

2) Date and venue for an events special request

```
ΠDate,venue(σEvent="Odyssey"(specialrequest))
Π  - project operation for viewing
```

3) Status of a particular request (ID)

```
ΠStatus(σid=17(allrecord))
```

4) Name of the worker servicing a request

```
Πworker.Name(σallrecord.id=17 and allrecord.WorkerID=worker.id(allrecord x worker))
X - cartesian product
```

5) All reviews for a particular worker

```
∏Reviews(σWorkerID=3(reviews))
```

6) Schedule for a particular request

```
∏starttime,closetime(σid=6(schedule))
```

7) All Requests closed on a particular day

```
∏id(σStatus="Closed" and closetime=2020-04-23(allrecord))
```

8) Workers who are currently busy

```
∏worker(σschedule.starttime < currentTime and schedule.closetime > currentTime and
worker.id=schedule.id(worker x schedule)
```

9) All workers under a supervisor

```
∏worker(σSupervisorID=3(worker))
```

10) Supervisor info of a particular worker

```
∏supervisor.id,supervisor.Name(σworker.id=10 and worker.SupervisorID = supervisor.id
(worker x supervisor))
```

## WEEK 7

As embedded SQL is not supported by MySql, Our group went further with using Dynamic SQL as directed by Md. Saad Akhtar Sir. Following dynamic queries were written in Java and the corresponding files are mentioned alongside the query.

1) To Decrypt the password retrieved from the database (File: Main.java)

```
ResultSet rs;
String query = String.format(
   "SELECT ar.ID,ar.Name FROM allusers ar WHERE ar.Username = \"%s\" and
substring(ar.Password,2,CHAR_LENGTH(ar.Password)-2) = \"%s\" and ar.Type = \"%s\"",
```

```
   username, password, );
rs = Main.con.createStatement().executeQuery(query);
```

2) To close an open record (File : Employee.java)

```
String query = String.format(
   "UPDATE allrecord SET status = \"Close\", closedtime =\"%s\" WHERE status =
\"Open\" and id= %d", date, workid)Main.con.createStatement().executeUpdate(query);
```

3) To change the phone number of an employee (File: Employee.java)

```
query = String.format("UPDATE worker set contactinfo=\"%s\" where id =%d", value,
       user.getID());
```

4) To show all records which are tagged Unassigned (File : FMS.java)

```
tablequery = "Select * From allrecord where status=\"Unassigned\"";
recordmenu.setText(i.getText());
filltable();
```

5) To show all records (File : FMS.java)

```
tablequery = "Select * From allrecord ";
recordmenu.setText(i.getText());
filltable();
```

6) To show all records which are tagged Open (File : FMS.java)

```
tablequery = "Select * From allrecord where status =\"Open\"";
recordmenu.setText(i.getText());
typefilltable();
```

7) To remove a particular record (File : User.java)

```
String query = String.format("Delete From allrecord where id=%d",
   row.getTableView().getSelectionModel().getSelectedItem().getId());
```

8) To show all previous tasks performed for a particular user (File : User.java)

```
rs = Main.con.createStatement().executeQuery("Select * FROM allrecord WHERE
```

```
studentid=" + user.getID());
while (rs.next()) {
 list.add(
     new Record(rs.getInt("ID"), rs.getString("workerid"), rs.getInt("studentid"),
         rs.getString("roomnum"), rs.getString("Status"),
rs.getString("requesttype"), rs.getTimestamp("starttime"),
         rs.getTimestamp("closedtime"), rs.getString("hostel"),
         rs.getString("comment")));
}
```

9) To show all previous tasks performed by a particular worker (File : EmployeeHistory.java)

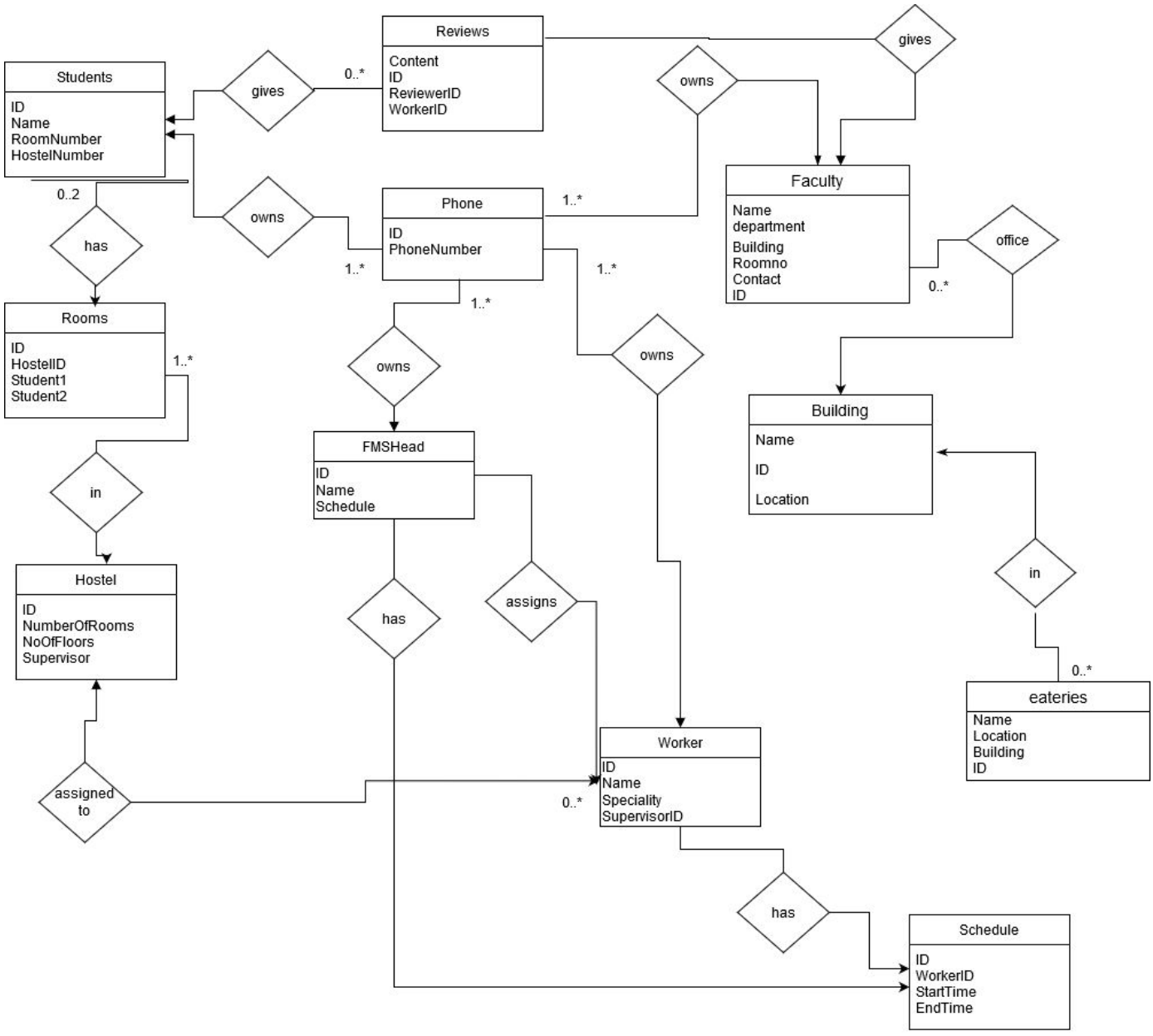
```
String query =   String.format(
       "SELECT ar.id,starttime,requesttype,roomno,ar.hostel "
           + "FROM allrecord ar,students s "
           + "WHERE s.id=ar.studentid and ar.workerid=%d ", Workerid);
ResultSet rs = Main.con.createStatement().executeQuery(query);
```

10) To add a new employee (File : AddEmployee.java)

```
Main.con.createStatement().executeUpdate(String.format("Insert into worker
(name,speciality,contactinfo) VALUES (\"%s\", \"%s\", \"%s\")", empname,
empspeciality, empphone));
```

## WEEK 8

Our group began in week 8 by making certain important modifications. Majorly **we *Normalised* our database and removed the inclusion of Supervisor stakeholder which posed to be a redundant step in the whole process.** With the current version of our Project, a person can directly file a request to certain services that can be directly assigned to the workers by the FMS heads. This accounted for the present system where we had 3 types of accounts in our application: Employees, Users, and FMS Heads.

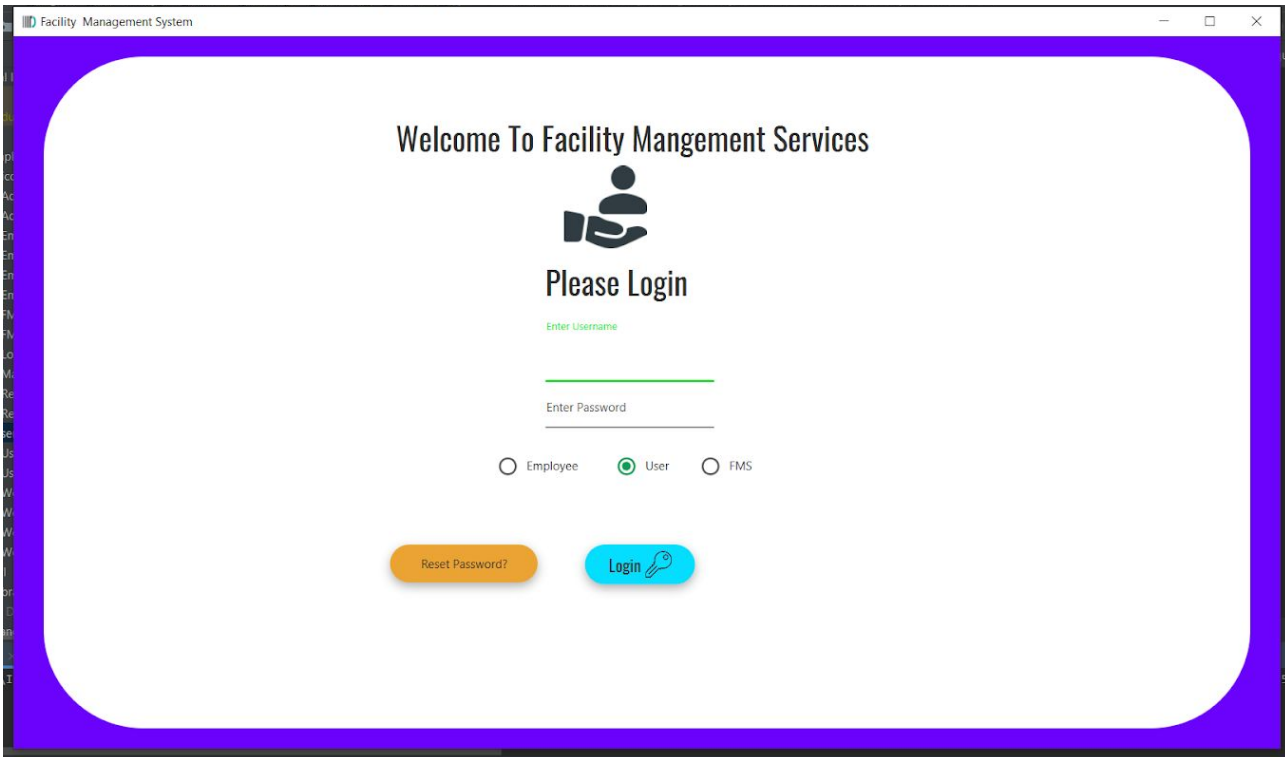


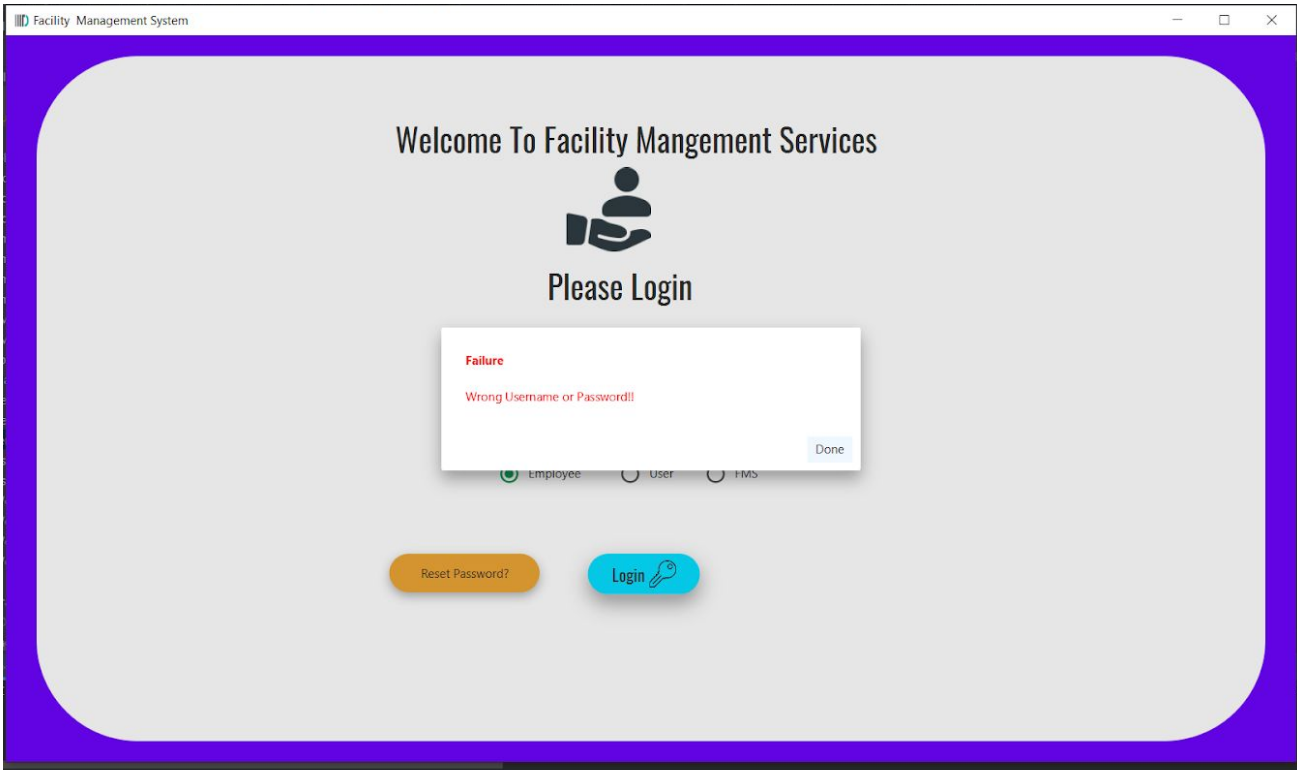

This week we also implemented ***triggers in our project***. This ensured that if a particular value is changed in some table, its effect is reflected in other tables as well. This also helped us maintain the uniqueness of ID of the user across the 3 identified stakeholders.

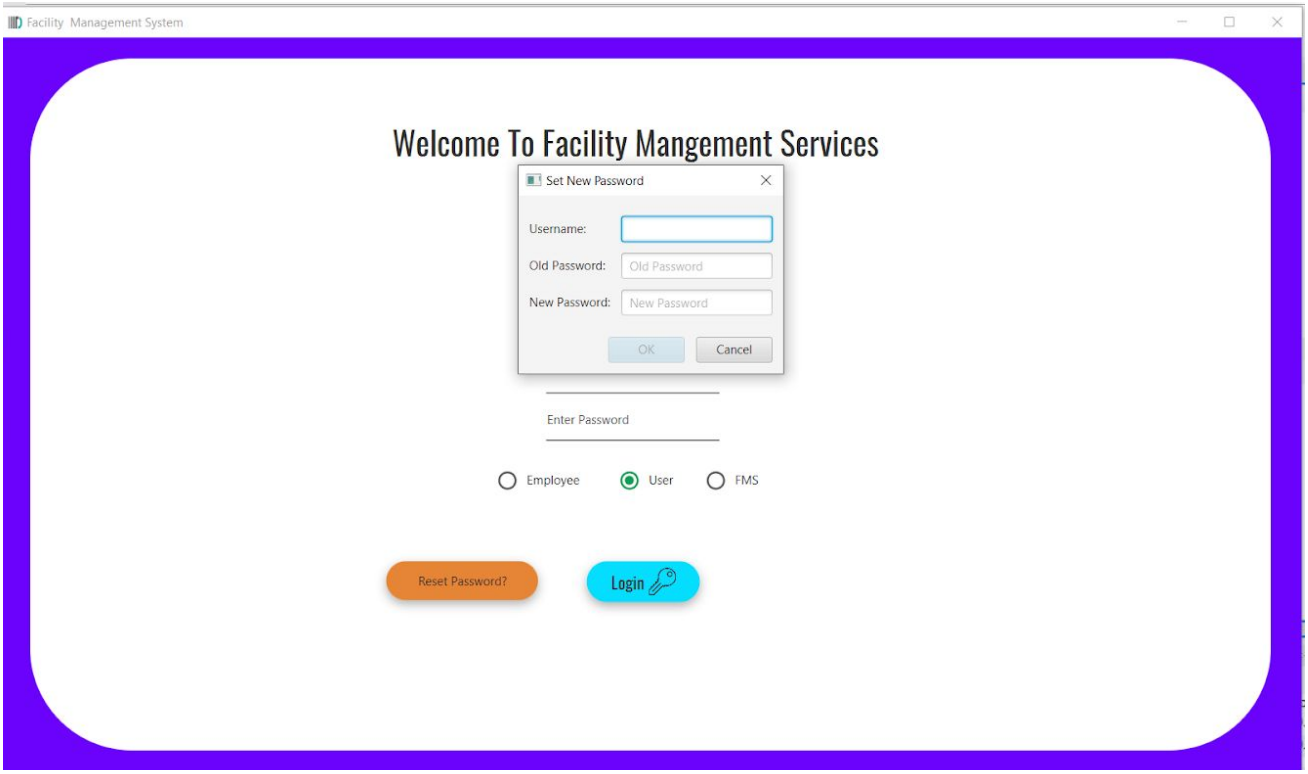
## WEEK 9 & 10

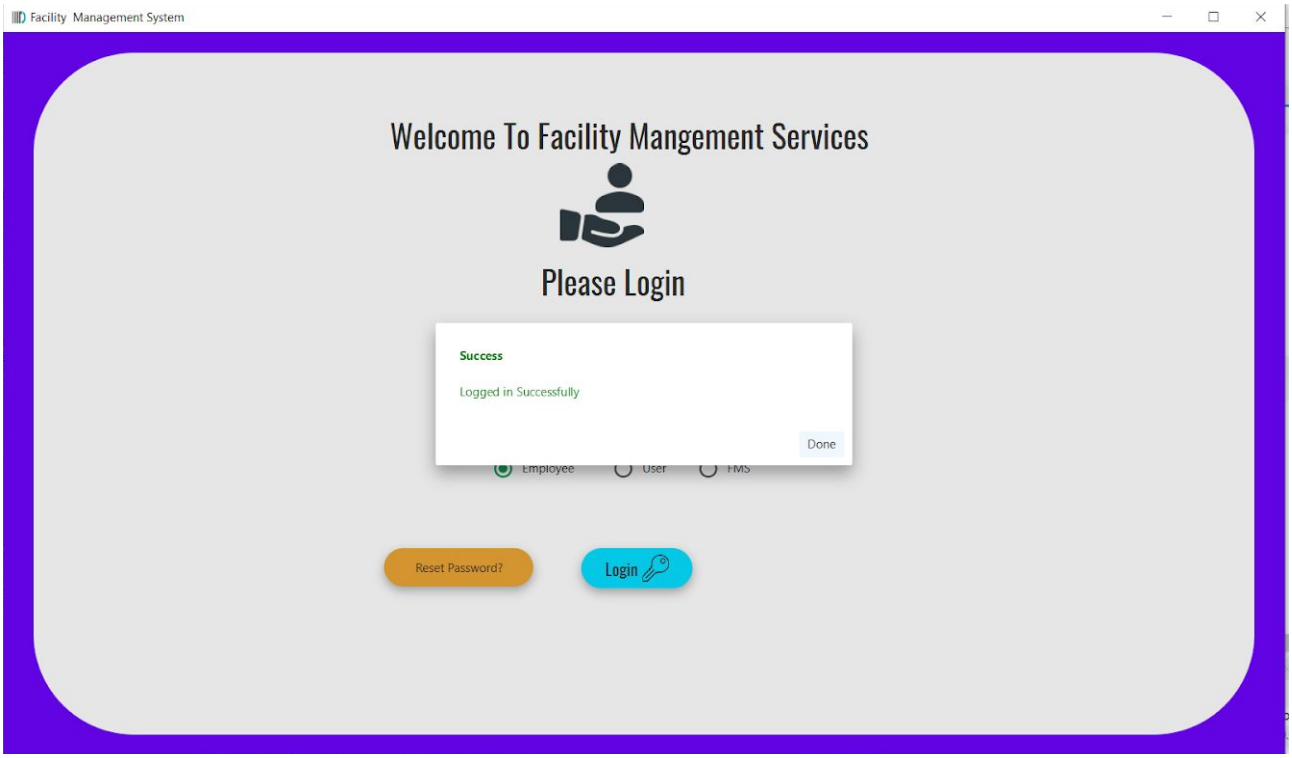
We started working on the GUI aspect of our project. The whole UI was decided to be implemented in JavaFX and later the web versions if we get a chance to actually implement it in the college. **The GUI was meant to be Database centric with modifiable attributes in the GUI itself.**

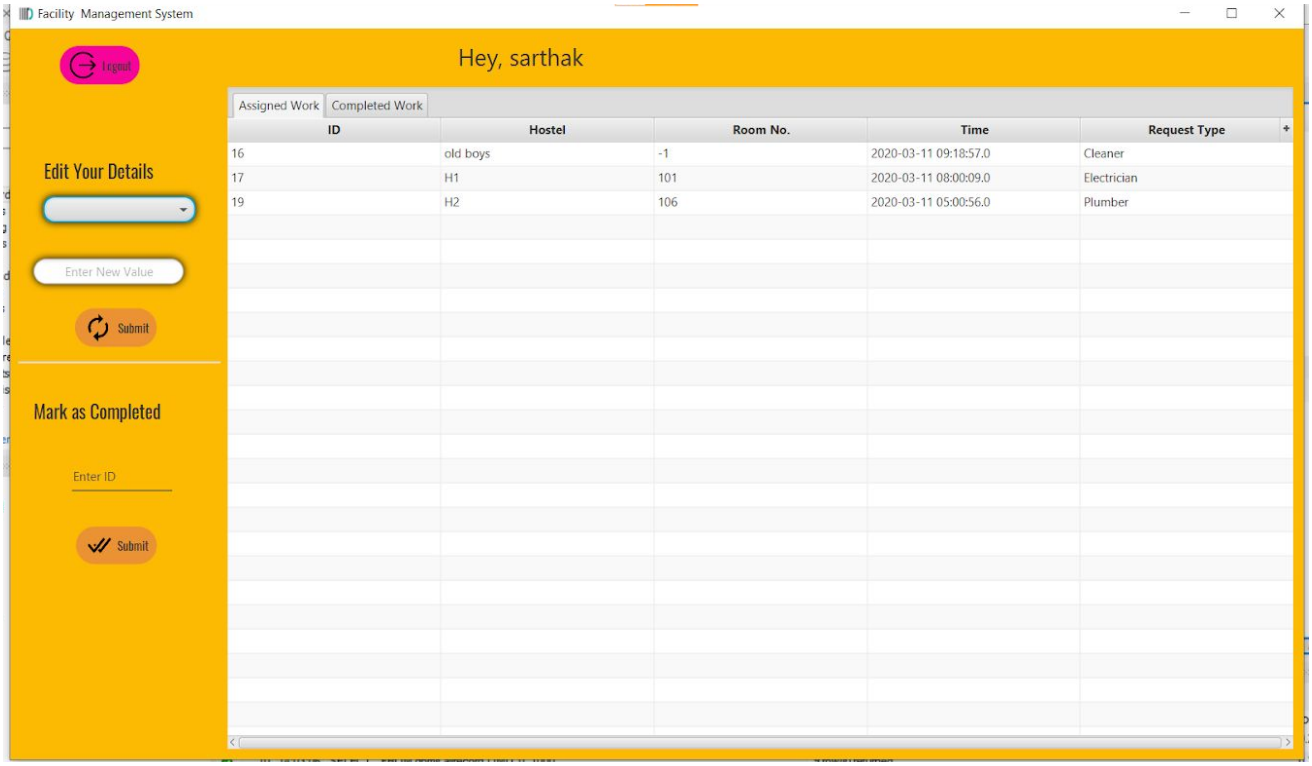
The following are are the screenshots from the app :

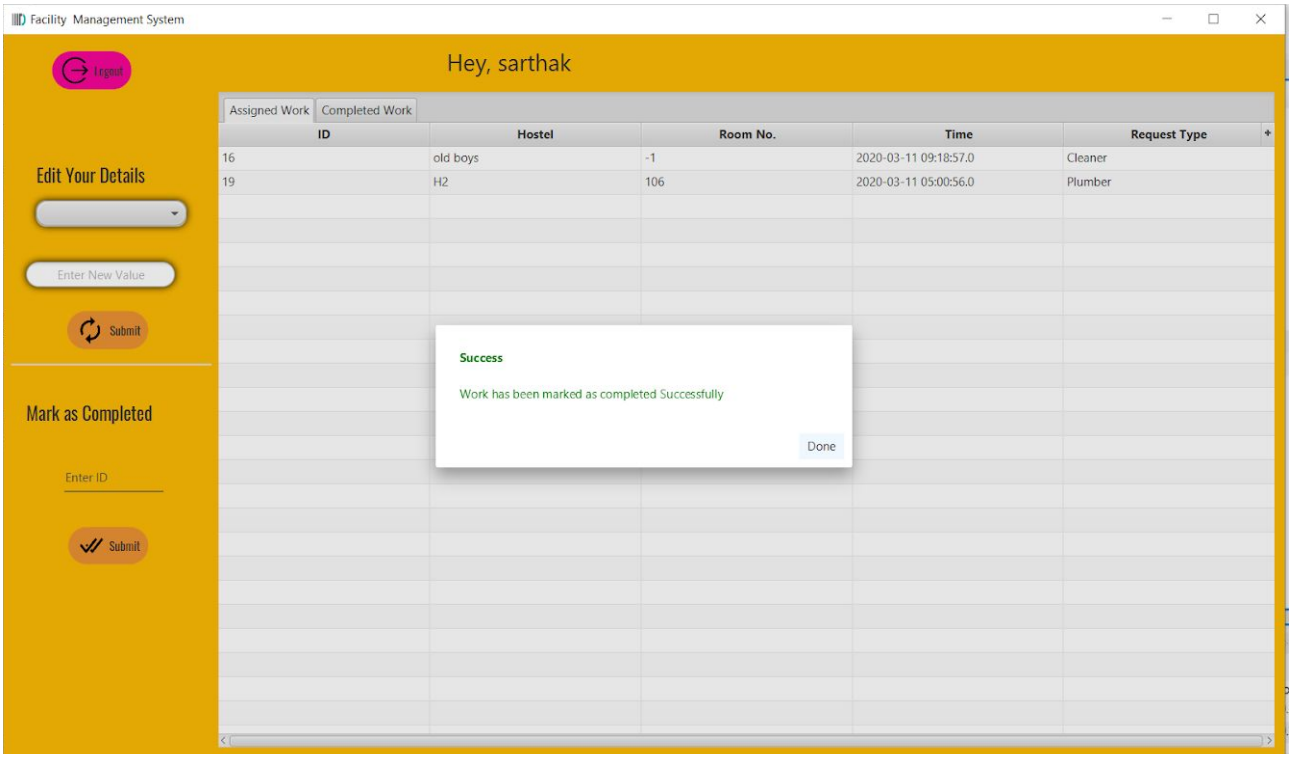


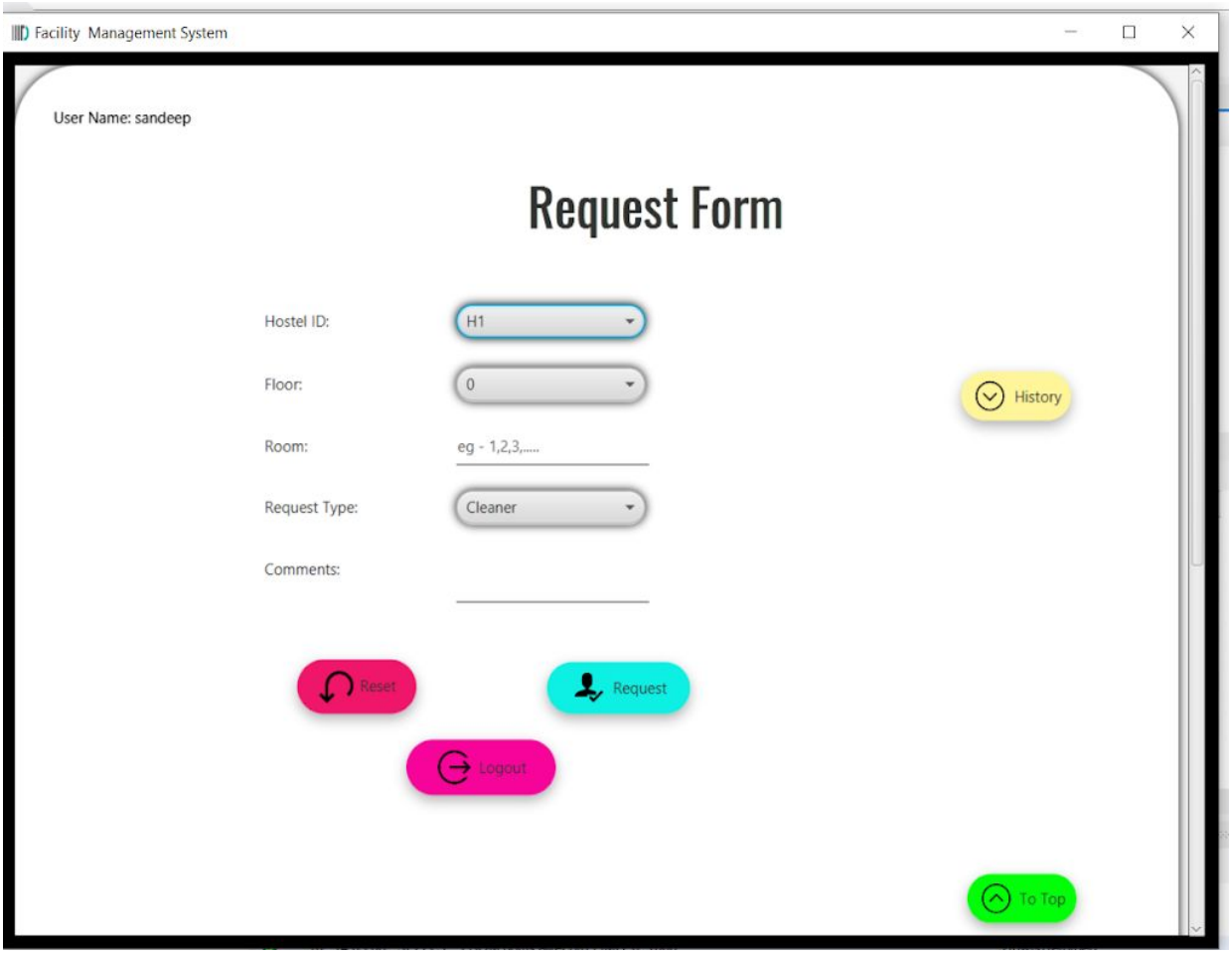



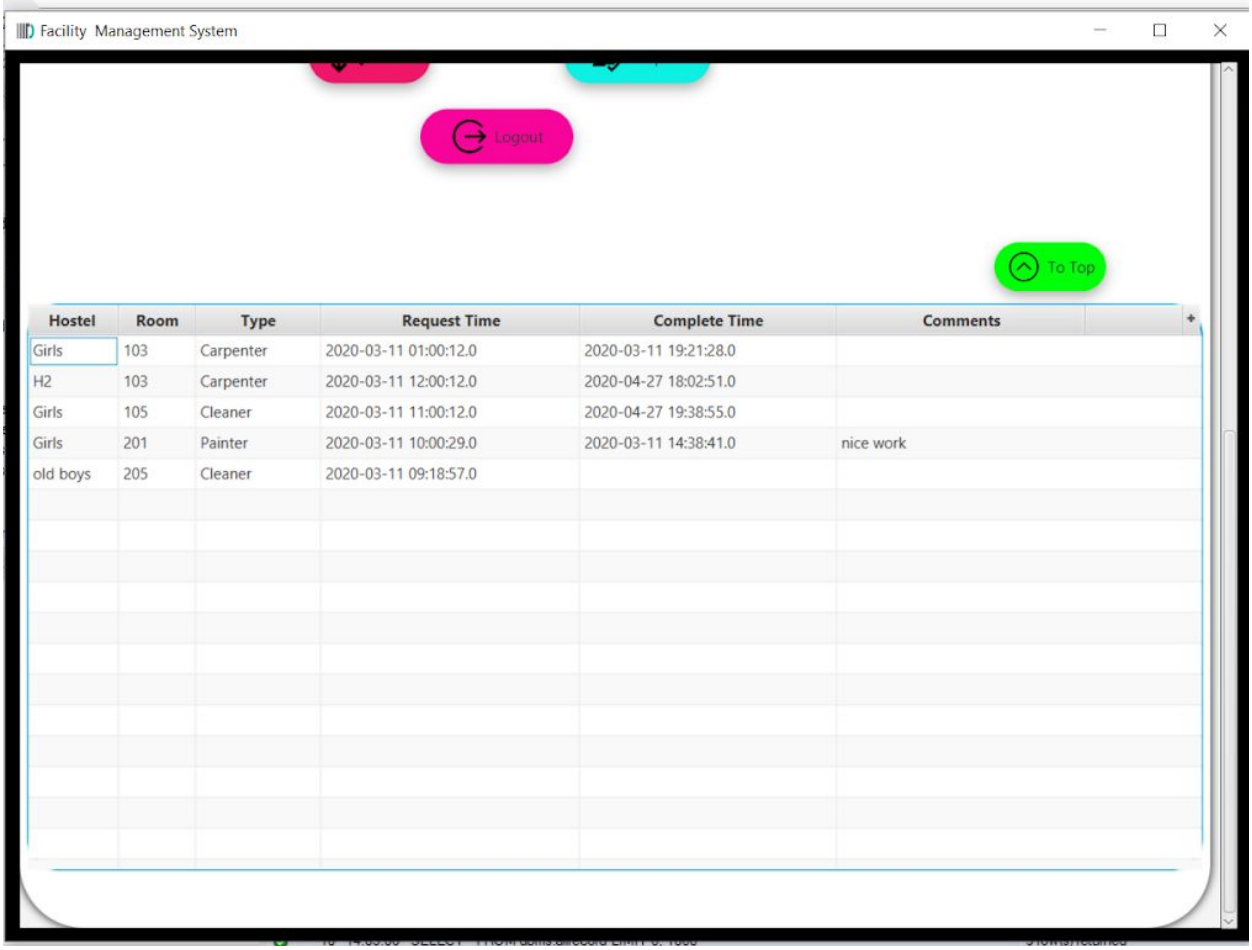

Facility Management System
Welcome To Facility Mangement Services
Set New Password
Username:
Old Password:
Old Password
New Password:
New Password
OK
Cancel
Enter Password
Employee
User
FMS
Reset Password?
Login

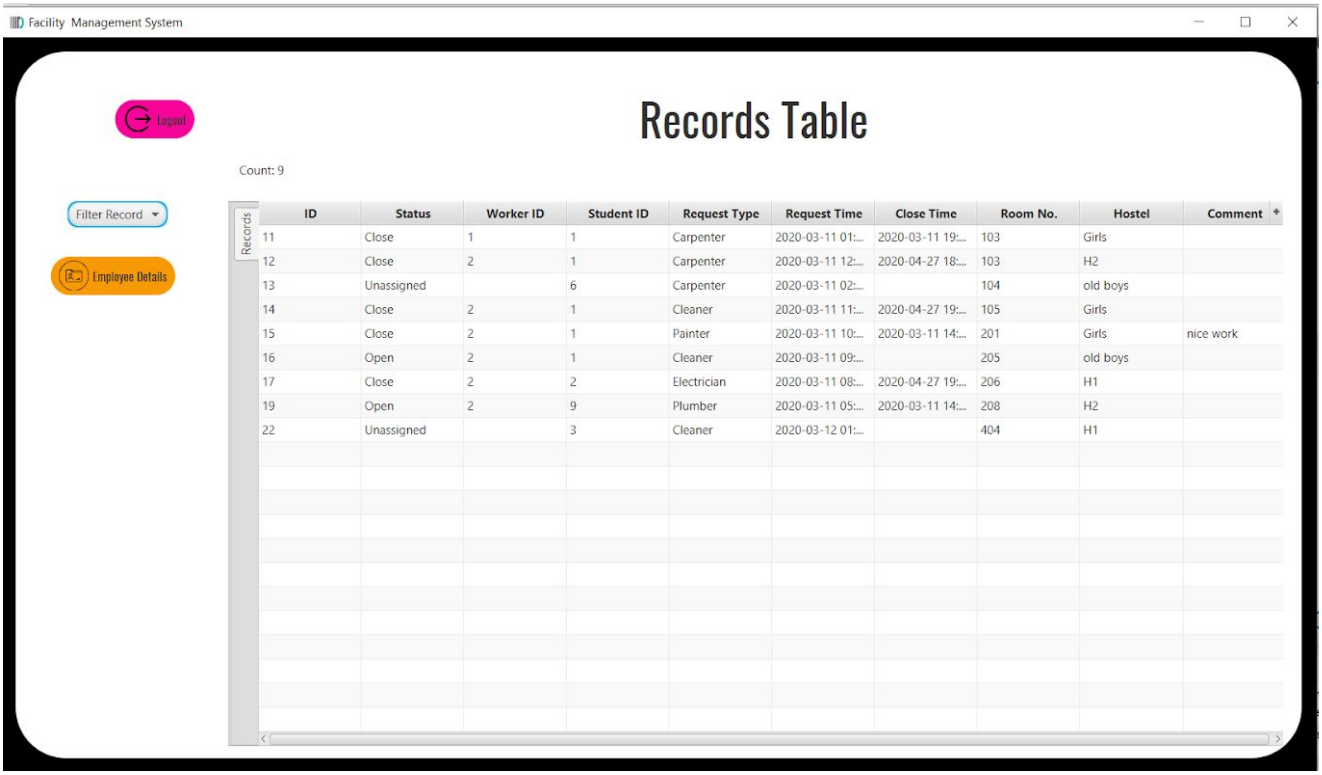

Facility Management System
Welcome To Facility Mangement Services
Please Login
Success
Logged in Successfully
Done
Reset Password?
Login


Facility Management System
Logout
Hey, sarthak
Assigned Work
Completed Work
ID
Hostel
Room No.
Time
Request Type
16
old boys
-1
2020-03-11 09:18:57.0
Cleaner
17
H1
101
2020-03-11 08:00:09.0
Electrician
19
H2
106
2020-03-11 05:00:56.0
Plumber
Edit Your Details
Enter New Value
Submit
Mark as Completed
Enter ID
Submit

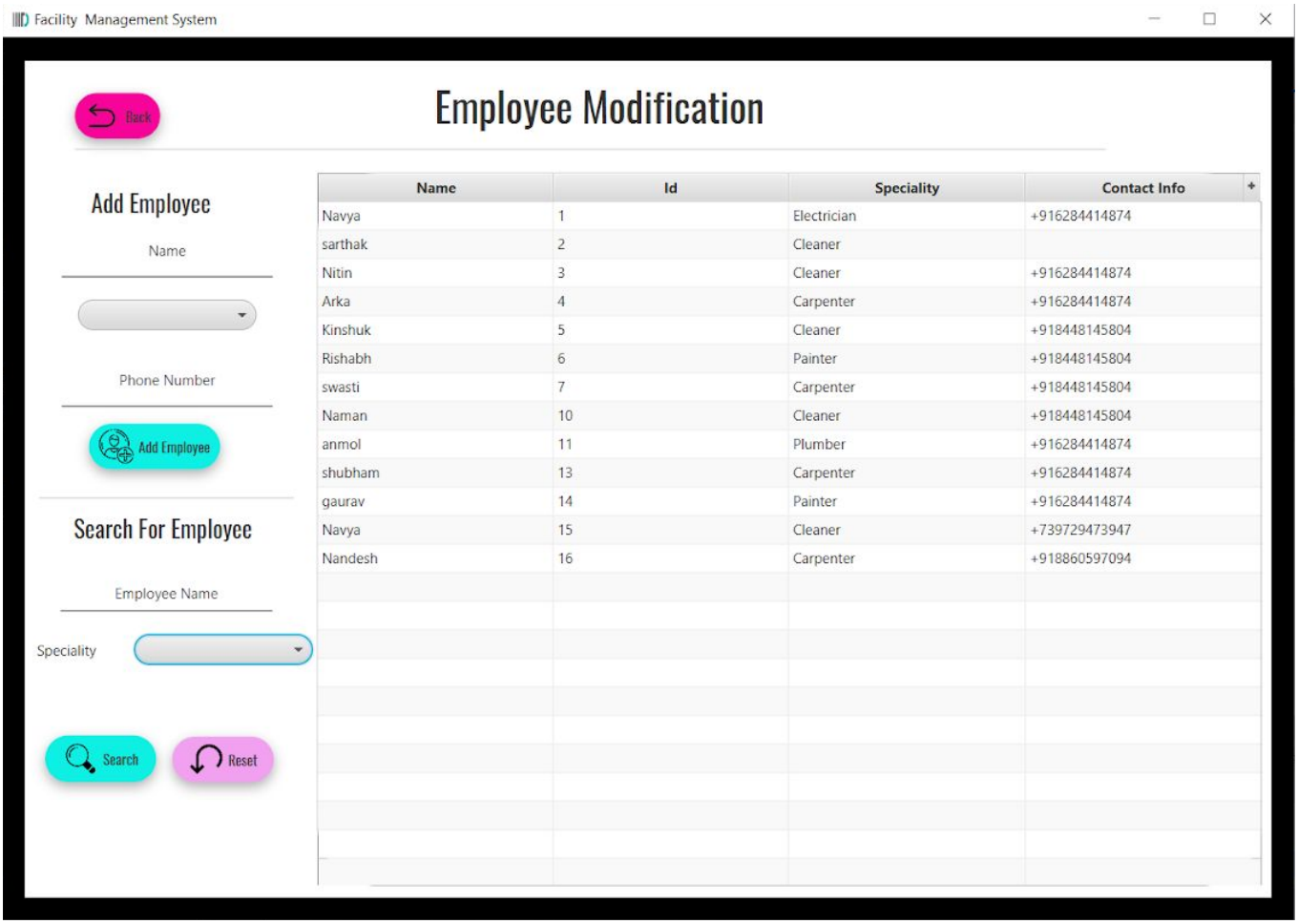

Facility Management System
Logout
Hey, sarthak
Assigned Work
Completed Work
ID
Hostel
Room No.
Time
Request Type
16
old boys
-1
2020-03-11 09:18:57.0
Cleaner
19
H2
106
2020-03-11 05:00:56.0
Plumber
Edit Your Details
Enter New Value
Submit
Mark as Completed
Enter ID
Submit
Success
Work has been marked as completed Successfully
Done


Facility Management System
User Name: sandeep
Request Form
Hostel ID:
H1
Floor:
0
History
Room:
eg - 1,2,3,.....
Request Type:
Cleaner
Comments:
Reset
Request
Logout
To Top


Facility Management System
Logout
To Top
Hostel
Room
Type
Request Time
Complete Time
Comments
Girls 103 Carpenter 2020-03-11 01:00:12.0 2020-03-11 19:21:28.0
H2 103 Carpenter 2020-03-11 12:00:12.0 2020-04-27 18:02:51.0
Girls 105 Cleaner 2020-03-11 11:00:12.0 2020-04-27 19:38:55.0
Girls 201 Painter 2020-03-11 10:00:29.0 2020-03-11 14:38:41.0 nice work
old boys 205 Cleaner 2020-03-11 09:18:57.0




| ID | Status | Worker ID | Student ID | Request Type | Request Time | Close Time | Room No. | Hostel | Comment |
|---|---|---|---|---|---|---|---|---|---|
| 11 | Close | 1 | 1 | Carpenter | 2020-03-11 01:... | 2020-03-11 19:... | 103 | Girls | |
| 12 | Close | 2 | 1 | Carpenter | 2020-03-11 12:... | 2020-04-27 18:... | 103 | H2 | |
| 13 | Unassigned | | 6 | Carpenter | 2020-03-11 02:... | | 104 | old boys | |
| 14 | Close | 2 | 1 | Cleaner | 2020-03-11 11:... | 2020-04-27 19:... | 105 | Girls | |
| 15 | Close | 2 | 1 | Painter | 2020-03-11 10:... | 2020-03-11 14:... | 201 | Girls | nice work |
| 16 | Open | 2 | 1 | Cleaner | 2020-03-11 09:... | | 205 | old boys | |
| 17 | Close | 2 | 2 | Electrician | 2020-03-11 08:... | 2020-04-27 19:... | 206 | H1 | |
| 19 | Open | 2 | 9 | Plumber | 2020-03-11 05:... | 2020-03-11 14:... | 208 | H2 | |
| 22 | Unassigned | | 3 | Cleaner | 2020-03-12 01:... | | 404 | H1 | |




| Name | Id | Speciality | Contact Info |
|---|---|---|---|
| Navya | 1 | Electrician | +916284414874 |
| sarthak | 2 | Cleaner | |
| Nitin | 3 | Cleaner | +916284414874 |
| Arka | 4 | Carpenter | +916284414874 |
| Kinshuk | 5 | Cleaner | +918448145804 |
| Rishabh | 6 | Painter | +918448145804 |
| swasti | 7 | Carpenter | +918448145804 |
| Naman | 10 | Cleaner | +918448145804 |
| anmol | 11 | Plumber | +916284414874 |
| shubham | 13 | Carpenter | +916284414874 |
| gaurav | 14 | Painter | +916284414874 |
| Navya | 15 | Cleaner | +739729473947 |
| Nandesh | 16 | Carpenter | +918860597094 |

## WEEK 11 & 12

In the last weeks, we started majorly working on our Bonus Elements of the Project.

There 3 major implementations for which our group seeks Bonus:

1. **Encryption Protected Passwords:** Passwords are of utter importance to everyone. Thus, it becomes very important for us to maintain security for them. For this, we have an encryption and decryption system enabled in our Application. All the passwords stored in the database are encrypted and can't be directly accessed by just looking at the database. The encryption and decryption mechanism is original and totally implemented by the team members and none of the existing sources are used.

2. **SMS Service for Worker:** This is a prime feature that was requested by an FMS Head who was interviewed during the research phase of our project. In any time when a worker is assigned work or his work is completed, he will receive an SMS that is automatically sent by the system. This maintains the feedback mechanism and no worker is left in doubt of their work. This also helps in the case a worker is not able to access the internet at any given time.

3. **Export to Excel:** Yet again a feature that was requested by the FMS Heads. This solves the problem of maintaining an offline record for various purposes. Now we can simply click on a button and the application will export the all records database to an excel sheet making the task easy for the FMS Heads.

4. **Database centered features in UI:** The UI of the application is not just a regular UI. It focuses on certain database features like adding and removing columns. This enhances the user experience by just providing an adequate amount of information so that the user is not subjected to any redundant information which is not useful to him.