SSN COLLEGE OF ENGINEERING
KALAVAKKAM-603110

Department of Computer Science and Engineering

Hostel Management System for Universities

Team Name: The Scanners

V S Pranav - Reg No: 3122 21 5001 **069**
Pooja Premnath - Reg No: 3122 21 5001 **066**
Prajit R - Reg No: 3122 21 5001 **068**

Problem Statement:

The maintenance of digital hostel records is key to ensuring efficient management. Factors such as security and accessibility are greatly enhanced by replacing the current system, which requires extensive paperwork and is cumbersome. Implementation of functionalities such as a system to apply and validate student outpasses streamlines the process, reducing hassle. The maintenance of a system to manage the entry and exit of visitors, for the students with reasons mentioned as well, is necessary to aid tracking. The assignment of rooms, considering requirements and availability, as well as a provision to update the allotment as necessary, would allow for greater clarity, than maintaining records on paper. This proposed hostel management system aims to generate outpasses, maintain visitor logs and streamline the room allocation process.

Motivation:

Universities have a massive number of hostellers and it is extremely essential to create a system that enables them to keep track of information that involves their whereabouts and their activities on a day-to-day basis. Currently, since all records are stored on paper, it involves a lot of time and effort to keep track of them. Storing these records, maintaining as well as updating them is very difficult and time-consuming. Keeping such essential information on paper, a source that is easily misplaced and could cause a whole swarm of problems makes retrieval of such records extremely tedious in nature.

The whereabouts of students in hostels are key to ensuring their security. For this purpose, universities have an outpass system wherein each student fills out a form that gives them permission to leave campus

during the specified time. However, most of the time, there is a large divide between the issue and the authentication of passes. Since there is no proper system for validation in place, forgery and the utilization of passes for unnecessary causes are also quite commonplace. The inefficiency and unreliability in terms of getting outpasses as and when necessary causes unnecessary tension and problems. When it comes to leaving college for a multitude of reasons that include sickness, family functions, and other personal reasons, the outpass isn't as accessible and the procurement seems harder even with valid details.

The allocation of rooms, in the current state, seems very random and insipid. The records that hold this information aren't maintained or updated on a reliable system or perhaps not even on a regular basis it would seem. This allocation of rooms to students is dynamic and changes after every academic year requiring frequent updation. There is a multitude of room types available in the hostel, ranging from shared accommodation to single rooms. It is necessary that rooms be allotted systematically based on demand and availability. Appropriate fees also have to be collected for different room types.

A hostel management system would enable records to be maintained electronically. The implementation of various functionalities within the system would largely automate tasks of updating records. Furthermore, records about every single student would be maintained in a systematic manner. The development of a system that facilitates double verification during the application of an outpass, would ensure greater reliability and security. Permission to obtain an outpass would have to be obtained from both the warden and the parent. Hostel room allocation can be dynamically updated based on the requirement and availability and on students moving in and out of the hostel. Visitor logs can be maintained, to keep track of people visiting the hostel for various purposes- be it, parents or maintenance.

Centralized control with greater access privileges is given to wardens and supervisors to change and update records as needed. In all, a hostel management system would largely simplify and ease out the hectic processes making them more accessible to students and massively improving record maintenance and updates.

Proposed Functionalities:

The following functionalities are proposed, to develop the hostel management system: outpass application and verification, visitor logs, and hostel room allocation.

Outpass Verification:
The student applies for the outpass with the necessary details of times(in and out) along with the reasons for leaving the campus. The notice of the application is then sent to both student's parents as well as the warden for verification and pending approval. This creates a sense of security due to the setup with double verification. Upon successful verification and cross-checking of the number of outpasses obtained by the student in a given month, the outpass is generated and then sent to the student.

Visitor logs:
The visitor logs keep track of all the visitors that a student gets. The entry and exit times for the visitor are noted along with the type of visitor, whether personal as family members or for the purpose of maintenance. The reasons for the visit are identically noted for both the above cases, and for maintenance,

the person is also mentioned, say, the electrician, the plumber, etc. The above details are kept as part of an organized record structure that is verifiable later on.

Hostel room allocation:
Taking into account the preference of the student for the room type and the availability of the preferred room, hostel rooms are allocated. For the new incumbents as well, the rooms are appropriately allotted. Vacancies are also updated, once students move out.

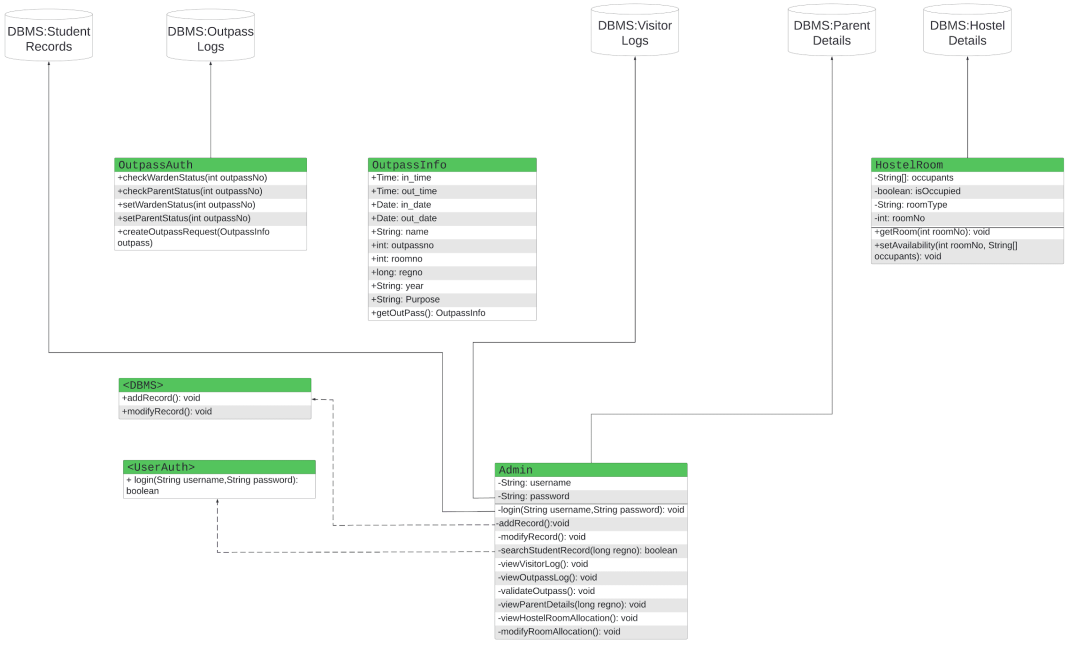
Scope and Limitations:

Scope:

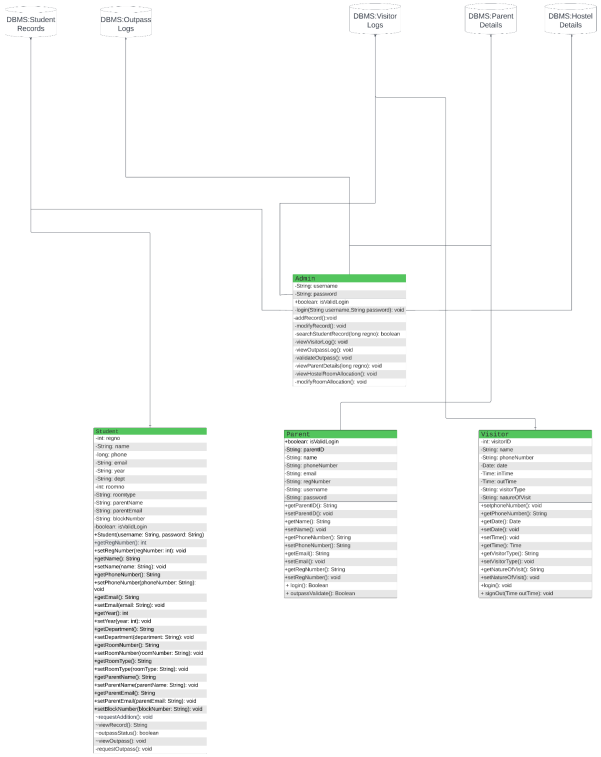
The project is a basic Java program that simulates a hostel management system. The program allows users to login as an administrator, student or parent and provides different functionality to each user type.
The administrator can add, modify and view student records, visitor logs, outpass logs, room allocation and parent details. The student can view their own record, request and view outpasses and view their outpass status. The parent can validate the outpasses of their child.
In terms of scope, the program provides basic functionality for a hostel management system as described above, but it does not function as an extensive and complete system that can handle all the features and operations of a real-world hostel management.

Limitations:
Security of passwords is not implemented.
There is only a command line interface, no GUI or Web interface.
The system does not have the capability to manage staff or other non-student users.
The system does not handle room allocation dynamically and does not have a provision for handling room changes, room swaps, or vacating rooms.

Class Diagram:



Updated Class Diagram:



Modules

1. **DBMS**

   Admin Table:
   The Admin record table contains the details of the admin and verifies if the person is indeed the admin based on username and password. In this table, the admin_id is labeled the primary key. There is no limit on the number of admins and the id gets auto-incremented as the other admins are added in.

```
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| admin_id | int          | NO   | PRI | NULL    | auto_increment |
| username | varchar(255) | NO   |     | NULL    |                |
| password | varchar(255) | NO   |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
```

Student Record Table:

The student record table contains details of all the students, including their names, register numbers, departments, phone numbers, room numbers, room types, etc. It also contains basic details about their parent, like the parents' names and the parent's email address. The register number of the student is taken to be the Primary Key. Pertinent information like the register number, name, phone number, room number, and room type are not allowed to have Null values.

Field	Type	Null	Key	Default	Extra
RegisterNumber	varchar(100)	NO	PRI	NULL	
Name	varchar(255)	NO		NULL	
PhoneNo	mediumtext	NO		NULL	
EmailID	varchar(200)	YES		NULL	
Year	int	YES		NULL	
Department	varchar(255)	NO		NULL	
RoomNo	int	NO		NULL	
RoomType	varchar(255)	NO		NULL	
ParentName	varchar(255)	NO		NULL	
ParentEmail	varchar(255)	NO		NULL	
StudentPassword	varchar(100)	NO		NULL	
BlockNo	varchar(3)	NO		NULL	

Parent Record Table

The Parent Record Table facilitates the parents of the students to have access to the hostel management system as well. Parent details such as the name, phone number, and email address are stored in the table. Mandatory fields are not allowed to have NULL values. This table is linked to the student table, using the Register Number that uniquely identifies every student. The ParentID is the Primary Key of the Parent Record table, while the Register Number is the Foreign Key that references the Register Number of the Student table.

Field	Type	Null	Key	Default	Extra
ParentID	varchar(100)	NO	PRI	NULL	
ParentName	varchar(255)	NO		NULL	
ParentPhoneNumber	mediumtext	NO		NULL	
ParentEmail	varchar(100)	YES		NULL	
RegisterNumber	varchar(100)	NO	MUL	NULL	
ParentPassword	varchar(100)	NO		NULL	

Hostel Rooms Table

The hostel rooms table maintains a record of all the rooms, which are occupied and unoccupied. This table contains details like the names of the occupant, whether the room is occupied or not, as well as the room number and the block number. The combination of the block number and the room number is taken to be the primary key.

Field	Type	Null	Key	Default	Extra
OccupantName	varchar(255)	NO		NULL	
RegisterNumber	varchar(100)	NO	PRI	NULL	
OccupancyStatus	char(1)	YES		E	
RoomType	varchar(100)	NO		NULL	
RoomNumber	int	NO	PRI	NULL	
BlockNumber	varchar(5)	NO	PRI	NULL	

Visitor Log Table

The visitor log table keeps track of all the visitors entering and leaving the hostel premises. Visitors are required to enter their phone number and the nature of the visit. The log table also maintains the reason for their visit, be it a parent, or part of the maintenance personnel. The date of visit and time of entry and exit are also maintained. A visitor ID, which is automatically generated, and auto-incremented in the log helps to uniquely identify each visitor.

Field	Type	Null	Key	Default	Extra
VisitorID	int	NO	PRI	NULL	auto_increment
VisitorName	varchar(255)	NO		NULL	
PhoneNo	mediumtext	NO		NULL	
DateofVisit	date	YES		NULL	
InTime	time	YES		NULL	
OutTime	time	YES		NULL	
VisitorType	varchar(255)	NO		NULL	
NatureOfVisit	varchar(255)	NO		NULL	

8 rows in set (0.02 sec)

Outpass Log Table

The outpass log table maintains a record of all the outpasses that have been applied for. Each outpass is identified by a unique outpass number. The outpass also contains information like the name and registration number of the student, along with the block and room they reside in. It also specifies the date and time of exit and entry, along with the purpose for requesting the outpass. When the warden and parent approve of the outpass request, suitable updates are made to the table.

```
+------------------+--------------+------+-----+---------+----------------+
| Field            | Type         | Null | Key | Default | Extra          |
+------------------+--------------+------+-----+---------+----------------+
| OutpassNumber    | int          | NO   | PRI | NULL    | auto_increment |
| Name             | varchar(255) | NO   |     | NULL    |                |
| RegisterNumber   | varchar(255) | NO   |     | NULL    |                |
| BlockNumber      | varchar(5)   | NO   |     | NULL    |                |
| RoomNumber       | int          | NO   |     | NULL    |                |
| Purpose          | varchar(500) | NO   |     | NULL    |                |
| Year             | int          | NO   |     | NULL    |                |
| OutDate          | date         | NO   |     | NULL    |                |
| InDate           | date         | NO   |     | NULL    |                |
| OutTime          | time         | NO   |     | NULL    |                |
| InTime           | time         | NO   |     | NULL    |                |
| WardenValidation | char(1)      | YES  |     | NULL    |                |
| ParentValidation | char(1)      | YES  |     | NULL    |                |
+------------------+--------------+------+-----+---------+----------------+
13 rows in set (0.02 sec)
```

2. **Developing the interface for DBMS Functionality:**

The common functionality between multiple classes exists in the form of the addition and modification of records, so an interface DBMS is used. It is implemented in the Admin, Student, and Parent classes. These functions are common for the storage and modification of the records of students, parent details to contact them as well as the details of the students in the hostel such as room type, room no, email address, etc.

Functions defined:

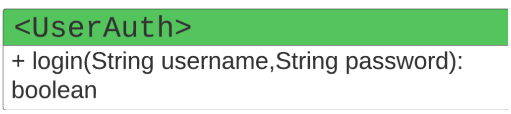
1. addRecord() [Access: Public]: Used to add records in the classes Admin, Student, and Parent.
2. modifyRecord() [Access: Public]: Used to modify the records in the classes Admin, Student and Parent.

3. **Developing the interface for Authentication of Users:**

The interface UserAuth is used to authenticate the user when their login details are entered. The interface method takes in the Username and the Password as parameters and returns a boolean value that allows the user to take the respective roles of Admin, Student, and Parent respectively. The respective user can access the respective functions as specified in the class diagram.

Functions defined:

1. login(String username, String password)[Access: Public]: Used to check login access and validate the user entering details.

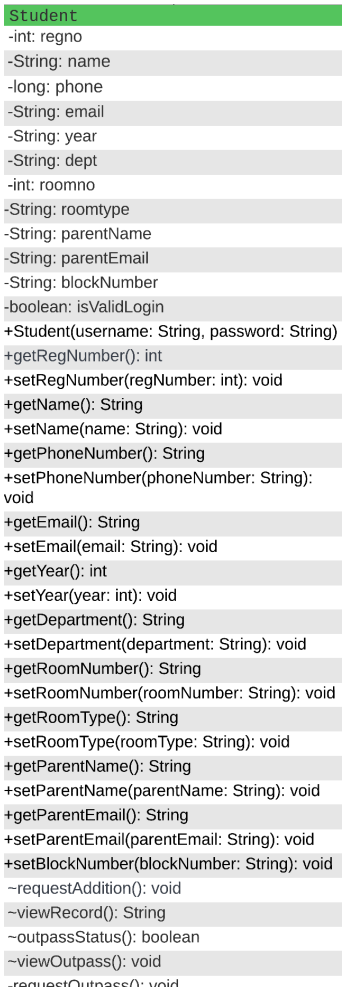


4. Student Class

Each instance of the class has the following member variables about the particular student: the student's registration number, name, phone number, email address, current year of study, department, hostel room number, the type of the room the student is currently staying in, the student's parent's name, email address.

Function Details:

It also has functions namely requestAddition() that is used to register a new student by sending the details to the admin pending approval after which it is added to the database and the viewRecord() function is used to view the student's aforementioned details. requestOutpass() is a function using which a student can request an outpass from the hostel office which will be granted after the warden and the parent approve it. The current status of the outpass can be tracked using the outpassStatus() function and once outpass has been approved, the student can view the outpass details using the viewOutpass() function.

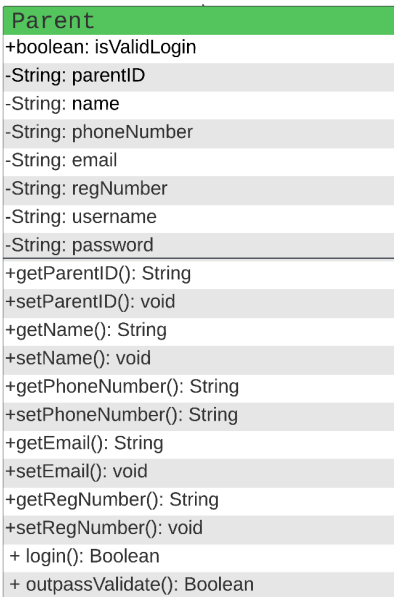


3. **Parent Class**

The class Parent is used for the storage of various details such as ParentID, i.e, a unique number used for identification, their name, phone number, email address and their ward's registration number. These details are pertinent towards making the process of obtaining the outpass easy and effortless in times of emergencies or important events.

Function Details:

Parents have the ability to login with the login() function enabling them to approve or deny any pending outpass requests which in turn can be accessed using the outpassValidate() function.



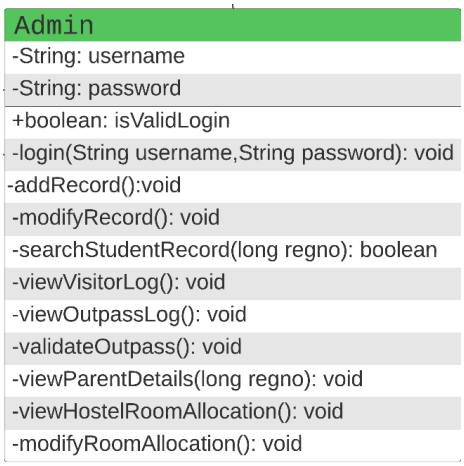
4. **Admin**

Admin is the class that contains the major functions. It takes in variables of login and password and authenticates the login of the Admin. It has the ability to access the various databases of VisitorLogs, Student Records, Hostel details along with Parent Details.

Functions defined:

1. login(String username,String Password)[Access: Private]: It authenticates the Admin login and gives access.
2. addRecord()[Access:Private]:It adds the respective student record to the database with necessary details mentioned in Student Class.
3. modifyRecord()[Access: Private]: It allows the Admin to modify records when errors creep in as well as to delete redundant records.
4. searchStudentRecord(long regno)[Access: Private]: It enables searching for a student's details when Regno of student is known and to verify details and update if necessary.

5. viewVisitorLog()[Access: Private]: The VisitorLogs DBMS can be accessed through this function and enables the admin to view the entry and exit details of visitors to respective rooms and purposes later on whenever required.
6. viewOutpassLog()[Access: Private]: This facilitates the Admin to view outpass data i.e details as well as the status of validation from parent and warden.
7. validateOutpass()[Access: Private]: This allows the outpass to be sent for validation to the warden and parent.
8. viewParentDetails(long Regno)[Access: Private]: This facilitates viewing the respective Parent DBMS and accessing the respective details for outpass verification.
9. viewHostelRoomAllocation()[Access: Private]: This enables the Admin to see the respective details for the rooms such as the type of room as well as the other details.
10. modifyRoomAllocation()[Access: Private]: This leads to changing of the students year after year in the respective rooms and regular updation as it occurs.

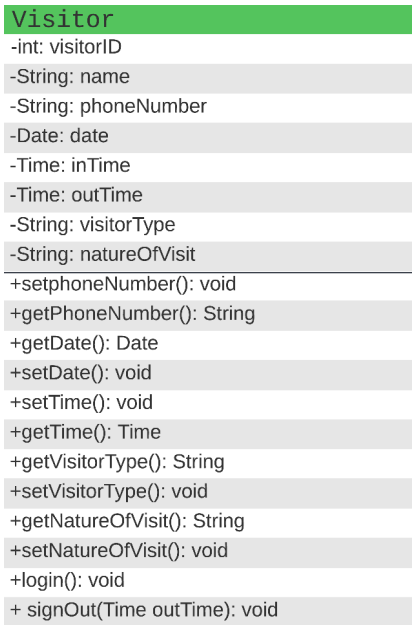


5. Visitor Class

This class contains the information of visitors who have to enter details such as VisitorID, name, phone number, date,inTime, Nature of visit and nature of visit on entry. For example: If a parent visits, the nature of the visit can be listed as personal with other details, or maintenance workers like plumbers or electricians may state the nature as maintenance, etc. The outTime can be entered on the way out.

Functions defined:

1. Variables listed are with access Private.
2. register()[Access: Public]: It takes the above-mentioned details and stores them in the Visitor Logs DBMS and can be accessed by the admin.
3. signOut(outTime)[Access: Public]: It takes the outTime details on exit and updates them in Visitor logs.



Implementation

Establishment of MySQL Connectivity

The backend of the project is set up with mySQL. Tables for maintaining student, parent and visitor logs are made. There are also tables for maintaining records of the rooms allocated, as well as the the outpasses that students have applied for. After the initial creation of these tables, it is necessary to establish connectivity between MySQL and Java.
Java applications can connect to a MySQL database using the MySQL Connector/J, a driver that implements the Java Database Connectivity (JDBC) API. To establish a connection, the application needs to load the driver and provide the URL, username, and password of the MySQL server. The application can then use standard JDBC methods to execute queries and update the database.

Logging into the System

Any user who wishes to use the hostel management system must initially log in with their credentials. Users can log in as the administrator, a student or a parent. Logging in makes use of the user's unique username and password or in the case of a student, their registration number and password. Checking is done with the corresponding MySQL table, to validate the credentials. The tables which are checked for each login attempt are as follows:

```
mysql> select* from admin;
+----------+----------+-----------+
| admin_id | username | password  |
+----------+----------+-----------+
|        1 | admin1   | password1 |
|        2 | admin2   | password2 |
+----------+----------+-----------+
2 rows in set (0.04 sec)
```

```
mysql> select* from student;
+---------------+---------------+------------+-------------------------+------+--------------------------------+--------+-------------------------+----
-------------+-----------------------------------+----------------+---------+
| RegisterNumber | Name          | PhoneNo    | EmailID                 | Year | Department                     | RoomNo | RoomType                | Par
entName      | ParentEmail                       | StudentPassword | BlockNo |
+---------------+---------------+------------+-------------------------+------+--------------------------------+--------+-------------------------+----
-------------+-----------------------------------+----------------+---------+
| 2110152       | Pooja Premnath | 2244668800 | pooja2110152@ssn.edu.in |    2 | Computer Science and Engineering |     19 | Single_AttachedBathroom | Pre
etha Premnath | tpreetha@gmail.com                | pooja@2110152   | 5A      |
| 2110272       | Ojus          | 9876512340 | ojuice2110272@ssn.edu.in |    2 | Computer Science and Engineering |      5 | Single_AttachedBathroom | Tej
uice          | tejuicedadofojuice1969@gmail.com | ojuice@2110272  | LH8     |
| 2110567       | Rita Verma    | 1234567899 | rita2110567@ssn.edu.in   |    1 | Mechanical Engineering           |     14 | Shared_AttachedBathroom | Min
u Verma       | minu@gmail.com                    | rita@2110567    | 5C      |
+---------------+---------------+------------+-------------------------+------+--------------------------------+--------+-------------------------+----
-------------+-----------------------------------+----------------+---------+
3 rows in set (0.00 sec)
```

```
mysql> select* from parent;
+------------------+------------------+-------------------+-----------------------------------+----------------+------------------+
| ParentID         | ParentName       | ParentPhoneNumber | ParentEmail                       | RegisterNumber | ParentPassword   |
+------------------+------------------+-------------------+-----------------------------------+----------------+------------------+
| minu@2110567     | Minu Verma       | 9876543211        | minu@gmail.com                    | 2110567        | 2110567@minu     |
| prajitha@2110568 | Prajitha VS      | 2244668899        | prajithavs@gmail.com              | 2110568        | 2110568@prajitha |
| preetha@2110152  | Preetha Premnath | 9003166462        | tpreetha@gmail.com                | 2110152        | 2110152@preetha  |
| tejuice          | Tejuice          | 0987645321        | tejuicedadofojuice1969@gmail.com | 2110272        | Tejuice@2110272  |
+------------------+------------------+-------------------+-----------------------------------+----------------+------------------+
4 rows in set (0.00 sec)
```

Administrator

Addition of records

The administrator is responsible for adding new students to the database. Student details like the register number, name, phone number, email id, year of study, department, room number, room type and block number are all collected and entered into the student table. For every student record that is created, a corresponding parent record is also created in the parent table, with the details of the parent as well. The register number of the student links both tables together.

Modification of records

The administrator has the ability to change any details pertaining to a student's record in the database, be it their year of study, room number, room number, room type, block etc. This is done by initially obtaining the column that has to be modified as input, and then using an update statement to change the corresponding details. The changes are seen in the student table.

Looking up student records

This function is mainly to search the student table in the database. Upon finding the needed record, the program displays the information from the database.

Viewing Visitor Logs

The function enables the administrator to view the information such as the visitor's name, phone number, nature of visit, in time, out time, date of visit, of all visitors that has been stored in the database.

Viewing Outpass Logs

The administrator has a very important role of approving or denying outpass requests by students. This function enables the administrator to view all the outpasses that the students have applied for, in the past and their status, i.e if they have been approved or not.

Validating outpasses

The process of validating outpasses involves the administrator accepting or denying the student's request. The administrator is shown the details of the outpass that the student has applied for, including the time and date the outpass request is for. Accordingly, the status is set to "A" for Accepted, or "D" for denied. When a student initially applies for an outpass, the warden validation feature is by default set to "P" for Pending.

Viewing parent details

Parent details such as their name, phone number, email address and their ward's registration number can be obtained by the administrator from the database in case it is needed, especially in case of emergencies.

Viewing room allocation

The warden can view the entire list of all the rooms available in the hostel and all the rooms' information such as room number, block number, room type, occupant(s) names.

Modifying room allocation

Student

Student Details
The details of a student like name, reg no, email address, Parent ID, year, room no, room type etc are collected from the student and then inputted into the database by the admin.

Outpass Status

The outpass status shows whether a given outpass is accepted or denied. It requires both parent and warden validation.

Viewing Outpass
An outpass is generated only after it gets approval from both parties. The subsequent outpass with validation is printed and then reviewed for approval.

Requesting for an outpass
This requires the student to enter the details such as when, inTime, outTime, inDate, outDate etc for the outpass to be generated.

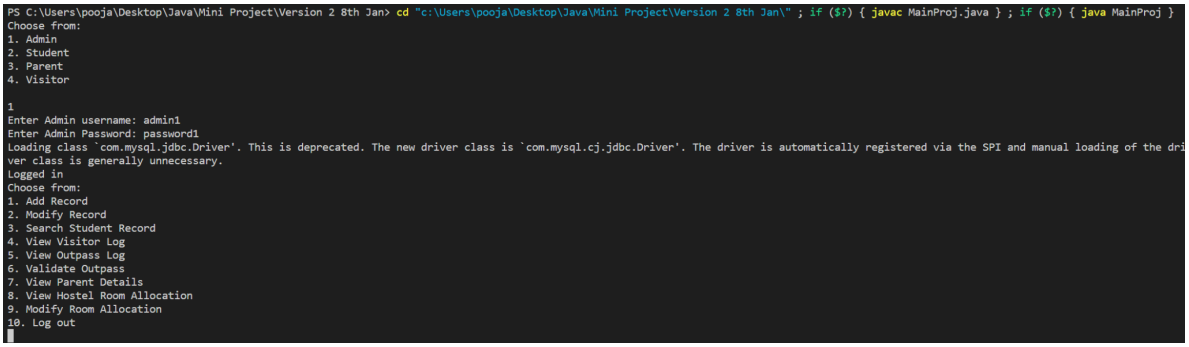
Visitor

The visitor class has been implemented to add new entries to the visitor logs. The class takes in the necessary information such as date of entry, in time, out time, phone number of visitor, name of the visitor, nature of their visit and the type of visitor, through a function and these details are then inserted into a MySQL database.

Output

- **Logging into the system**

    1. **Administrator**




```
mysql> select* from admin;
+----------+----------+-----------+
| admin_id | username | password  |
+----------+----------+-----------+
|        1 | admin1   | password1 |
|        2 | admin2   | password2 |
+----------+----------+-----------+
2 rows in set (0.00 sec)
```

2. Student

```
PS C:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan> cd "c:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan\" ; if ($?) { javac MainProj.java } ; if ($?) { java MainProj }
Choose from:
1. Admin
2. Student
3. Parent
4. Visitor

2
Enter Student Reg No: 2110152
Enter Student Password: pooja@2110152
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of the dri
ver class is generally unnecessary.
Logged in
Choose from:
1. View Details
2. Outpass Status
3. View Outpass
4. Request Outpass
5. Log out
```

3. Parent

```
PS C:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan> cd "c:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan\" ; if ($?) { javac MainProj.java } ; if ($?) { java MainProj }
Choose from:
1. Admin
2. Student
3. Parent
4. Visitor

3
Enter Parent ID: preetha@2110152
Enter Parent Password: 2110152@preetha
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of the dri
ver class is generally unnecessary.
Logged in
Choose from:
1.Outpass Validation
2.Log out
```

4. Visitor

```
PS C:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan> cd "c:\Users\pooja\Desktop\Java\Mini Project\Version 2 8th Jan\" ; if ($?) { javac MainProj.java } ; if ($?) { java MainProj }
Choose from:
1. Admin
2. Student
3. Parent
4. Visitor

4
Choose from:
1.Add Visitor Entry
2. Log out
```

- **Administrator Functionalities**

1. Addition of records

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
1
Enter student's name: Pranavi VS
Enter student's regno: 2110568
Enter student's phone number: 1133557788
Enter student's email: pranavi2110568@ssn.edu.in
Enter student's year: 2
Enter student's department: Computer Science and Engineering
Enter student's room number: 11
Enter student's room type: Single_AttachedBathroom
Enter student's parent's name: Prajitha VS
Enter student's parent's email: prajithavs@gmail.com
Enter student's password: pranavi@2110568
Enter student's block: 5A
Enter parent's ID: prajitha@2110568
Enter parents's phone number: 2244668899
Enter parent's password: 2110568@prajitha
```

```
mysql> select* from student;
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------------+---
---------------+----------------------------------------+-----------------+---------+
| RegisterNumber | Name      | PhoneNo     | EmailID                     | Year | Department                     | RoomNo | RoomType
| ParentName     | ParentEmail                            | StudentPassword | BlockNo |
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------------+---
---------------+----------------------------------------+-----------------+---------+
| 2110152        | Pooja Premnath | 2244668800 | pooja2110152@ssn.edu.in     |    2 | Computer Science and Engineering |     19 | Single_AttachedBathroom | Pr
eetha Premnath | tpreetha@gmail.com                     | pooja@2110152   | 5A      |
| 2110272        | Ojus           | 9876512340 | ojuice2110272@ssn.edu.in    |    2 | Computer Science and Engineering |      5 | Single_AttachedBathroom | Te
juice          | tejuicedadofojuice1969@gmail.com       | ojuice@2110272  | LH8     |
| 2110567        | Rita Verma     | 1234567899 | rita2110567@ssn.edu.in      |    1 | Mechanical Engineering           |     14 | Shared_AttachedBathroom | Mi
nu Verma       | minu@gmail.com                         | rita@2110567    | 5C      |
| 2110568        | Pranavi VS     | 1133557788 | pranavi2110568@ssn.edu.in   |    2 | Computer Science and Engineering |     11 | Single_AttachedBathroom | Pr
ajitha VS      | prajithavs@gmail.com                   | pranavi@2110568 | 5A      |
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------------+---
---------------+----------------------------------------+-----------------+---------+
4 rows in set (0.00 sec)

mysql>
```

2. Modification of records

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
2
Enter the regno of the student whose record you want to modify: 2110568
Enter the field you want to update (name, phone_number, email, year, department, room_number, room_type, parent_name, parent_email, parent_password): year
Enter the student's new year: 4
Do you want to update more fields (y/n)? y
Enter the field you want to update (name, phone_number, email, year, department, room_number, room_type, parent_name, parent_email, parent_password): department
Enter the student's new department: Electronics and Communication Engineering
Do you want to update more fields (y/n)? n
```

```
mysql> select* from student;
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------
-----+----------------+----------------------------------------+-----------------+---------+
| RegisterNumber | Name      | PhoneNo     | EmailID                     | Year | Department                     | RoomNo | RoomType
     | ParentName     | ParentEmail                            | StudentPassword | BlockNo |
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------
-----+----------------+----------------------------------------+-----------------+---------+
| 2110152        | Pooja Premnath | 2244668800 | pooja2110152@ssn.edu.in     |    2 | Computer Science and Engineering |     19 | Single_AttachedBath
room | Preetha Premnath | tpreetha@gmail.com                     | pooja@2110152   | 5A      |
| 2110272        | Ojus           | 9876512340 | ojuice2110272@ssn.edu.in    |    2 | Computer Science and Engineering |      5 | Single_AttachedBath
room | Tejuice          | tejuicedadofojuice1969@gmail.com       | ojuice@2110272  | LH8     |
| 2110567        | Rita Verma     | 1234567899 | rita2110567@ssn.edu.in      |    1 | Mechanical Engineering           |     14 | Shared_AttachedBath
room | Minu Verma       | minu@gmail.com                         | rita@2110567    | 5C      |
| 2110568        | Pranavi VS     | 1133557788 | pranavi2110568@ssn.edu.in   |    4 | Electronics and Communication Engineering |     11 | Single_AttachedBath
room | Prajitha VS      | prajithavs@gmail.com                   | pranavi@2110568 | 5A      |
+----------------+-----------+-------------+-----------------------------+------+--------------------------------+--------+--------------------
-----+----------------+----------------------------------------+-----------------+---------+
4 rows in set (0.00 sec)
```

3. Looking up student records

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
3
Enter student register number: 2110568
Name: Pranavi VS       Regno: 2110568  Phone number: 1133557788        Email: pranavi2110568@ssn.edu.in        Year: 4 Department: Electronics and Communication Engineering   Room number: 1R
oom type: Single_AttachedBathroom       Parent's name: Prajitha VS      Parent's email: prajithavs@gmail.com
```

4. Viewing visitor logs

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
4
Visitor's name: Latha   Visitor's phone number: 1122334455      Visitor's date of visit: 2022-01-03     Nature of visit: Visit Ward     Entry time: 1970-01-01 11:30:00.0       Exit time: 1970
-01-01 12:30:00.0       Type of visitor: Parent

Visitor's name: John    Visitor's phone number: 9988776655      Visitor's date of visit: 2022-01-04     Nature of visit: Maintenance    Entry time: 1970-01-01 09:00:00.0       Exit time: 1970
-01-01 10:00:00.0       Type of visitor: Contractor
```

5. Viewing Outpass Logs

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
5
Student's name: Pooja Premnath  Student's regno: 2110152       Student's block number: 5A      Student's room number: 19       Student year: 2 Purpose: Going home     Warden's approval: P  P
arent's approval: P     Out Date: 2023-09-01 00:00:00.0 Out Time: 1970-01-01 11:30:00.0 In Date: 2023-10-01 00:00:00.0  In Time: 1970-01-01 10:30:00.0
```

6. Validating outpasses

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
6
OutpassNumber: 3        Name: Pooja Premnath    RegisterNumber: 2110152 BlockNumber: 5A RoomNumber: 19  Purpose: Going home     Year: 2 OutDate: 2023-09-01     InDate: 2023-10-01      OutTime
: 11:30:00      InTime: 10:30:00        WardenValidation: P     ParentValidation: P
Enter the new status (approved(A) or rejected(D)): A
```

```
mysql> select* from outpasslogs;
+---------------+---------------+----------------+-------------+------------+------------+------+------------+------------+----------+----------+----------
--------+------------------+
| OutpassNumber | Name          | RegisterNumber | BlockNumber | RoomNumber | Purpose    | Year | OutDate    | InDate     | OutTime  | InTime   | WardenVal
idation | ParentValidation |
+---------------+---------------+----------------+-------------+------------+------------+------+------------+------------+----------+----------+----------
--------+------------------+
|             3 | Pooja Premnath | 2110152        | 5A          |         19 | Going home |    2 | 2023-09-01 | 2023-10-01 | 11:30:00 | 10:30:00 | A
       | P                |
+---------------+---------------+----------------+-------------+------------+------------+------+------------+------------+----------+----------+----------
--------+------------------+
1 row in set (0.00 sec)
```

7. Viewing parent details

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
7
Enter student register number: 2110568
Student's regno: 2110568        Parent's ID: prajitha@2110568   Parent's name: Prajitha VS      Parent's phone number: 2244668899       Parent's email: prajithavs@gmail.com
```

8. Viewing room allocation

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
8
Student's Name: Pranavi VS      Student's Register Number: 2110568      Student's Room Number: 11       Room type: Single_AttachedBathroom      Block Number: 5A

Student's Name: Rita Verma      Student's Register Number: 2110567      Student's Room Number: 14       Room type: Shared_AttachedBathroom      Block Number: 5C

Student's Name: Pooja Premnath  Student's Register Number: 2110152      Student's Room Number: 30       Room type: Single_SharedBathroom        Block Number: 3B

Student's Name: Ojuice  Student's Register Number: 2110272      Student's Room Number: 42       Room type: Single_AttachedBathroom      Block Number: LH7
```

9. Modifying room allocation

```
Choose from:
1. Add Record
2. Modify Record
3. Search Student Record
4. View Visitor Log
5. View Outpass Log
6. Validate Outpass
7. View Parent Details
8. View Hostel Room Allocation
9. Modify Room Allocation
10. Log out
9
Enter the regno of the student whose hostel record you want to modify: 2110568
Enter the student's new room number: 33
Enter the student's new block number: 5A
Enter the student's new room type: Single_AttachedBathroom
```

```
mysql> select* from hostelrooms;
+----------------+----------------+-----------------+-------------------------+------------+-------------+
| OccupantName   | RegisterNumber | OccupancyStatus | RoomType                | RoomNumber | BlockNumber |
+----------------+----------------+-----------------+-------------------------+------------+-------------+
| Rita Verma     | 2110567        | 0               | Shared_AttachedBathroom |         14 | 5C          |
| Pooja Premnath | 2110152        | 0               | Single_SharedBathroom   |         30 | 3B          |
| Pranavi VS     | 2110568        | 0               | Single_AttachedBathroom |         33 | 5A          |
| Ojuice         | 2110272        | 0               | Single_AttachedBathroom |         42 | LH7         |
+----------------+----------------+-----------------+-------------------------+------------+-------------+
4 rows in set (0.00 sec)
```

● Student User Functionalities

1. Student Details

```
Choose from:
1. View Details
2. Outpass Status
3. View Outpass
4. Request Outpass
5. Log out
1
Register Number: 2110152        Name: Pooja Premnath    Phone number: 2244668800        Email: pooja2110152@ssn.edu.in  Year: 2 Department: Computer Science and Engineering    Room number: 1R
oom type: Single_AttachedBathroom       Parent name: Preetha Premnath   Parent email: tpreetha@gmail.com        Block number: 5A
```

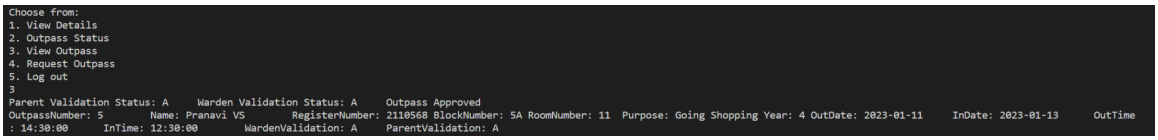
2. Outpass Status

```
Choose from:
1. View Details
2. Outpass Status
3. View Outpass
4. Request Outpass
5. Log out
2
Parent Validation Status: P     Warden Validation Status: A     Outpass Not Approved Yet
```

3. Requesting for an outpass

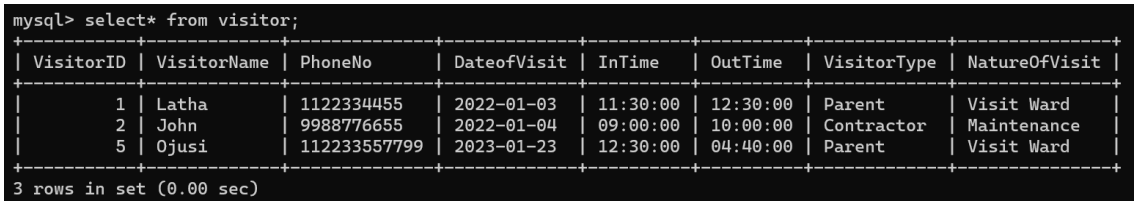
```
Enter Student Reg No: 2110568
Enter Student Password: pranavi@2110568
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of the dri
ver class is generally unnecessary.
Logged in
Choose from:
1. View Details
2. Outpass Status
3. View Outpass
4. Request Outpass
5. Log out
4
Enter the purpose of the outpass: Going Shopping
Enter the out date (yyyy-MM-dd): 2023-01-11
Enter In Date (yyyy-MM-dd): 2023-01-13
Enter In Time (HH:mm): 12:30
Enter Out Time (HH:mm): 14:30
```

4. **Viewing Outpass**



- **Visitor Functionalities**

Adding a visitor entry



Object Oriented Features:

1. **Exceptions:**

Inbuilt exceptions are handled in the code. SQL exceptions like “TableNotFound”are handled when calling the SQL tables and using it for purposes like inserting, viewing or modifying records.Certain other exceptions like “ClassNotFound” when calling SQL tables are for the interface between java and SQL to reference the links between the classes and the information they give to the respective tables.

2. **Interfaces:**

An interface is used to reference certain methods and functions common to multiple classes. The “UserAuth” interface is initialized for Parent and is also referenced in the Student and Admin where they enter the respective login details for logging into the database.

3. **SQL Connectivity:**

The connection of both the java and the sql interface takes place through JDBC Driver and is called for the access of information from the database for various purposes such as the visitor log and the student record etc.

4. **Encapsulation:**

Encapsulation is used in the initial program by defining member variables as private, and providing public methods to access and modify their values. For example, in the Administrator class, the member variables username and password are defined as private, and the login() method is used to access and verify their values against the data stored in the database. Similarly, in the Student class, the member variable isValidLogin is defined as private and is modified using the public login() method.

The class Administrator, Student and Parent also have their own functions for login, add record, search student record, view outpass log, and so on. Additionally, the main class MainProj uses encapsulation by creating objects of these classes and calling their methods to execute specific actions based on user input.

Inference and Future Extension:

Inference:
The Java program simulates a student management system for a college hostel. The program has three main user roles: Administrator, Student, and Parent. Each role has a different set of functionality that the user can access.
The login functionality is defined in the Administrator class and is reused in Student and Parent classes.Encapsulation is used in the program to protect the data members of the classes from unauthorized access. The data members of the Administrator, Student, and Parent classes are declared private, meaning that they cannot be accessed directly from outside the class. Instead, the program uses getter and setter methods to access and modify the data members. This allows the program to maintain the integrity of the data and ensure that it is only modified in a controlled manner.

Future Extension:
In terms of future extensions, the program could be extended in a number of ways. Some possible ideas include adding more functionality for the administrator, such as the ability to generate reports on student attendance and performance. Additionally, the program could be extended to include additional user roles, such as teachers. The program also has the potential to be integrated with other systems, such as an attendance tracking system or a grade book. Finally, the program could be made more visually appealing by implementing a GUI using Java's Swing library.