Vision

Provide skilled professionals in Computer Engineering to contribute towards the advancement of technology useful for society and industrial environment.

Mission

M1. Impart need based and value based education by providing exposure of latest tools and technologies in the area of computer engineering to satisfy the stakeholders.

M2. Upgrade and maintain facilities for quality technical education with continuous effort for excellence in Computer Engineering.

M3. Train students with Computer Engineering knowledge to apply it in the general disciplines of design, deployment of software and integration of existing technologies for E-governance and for benefit of society.

M4. Provide a learning ambience to enhance innovations, problem solving skills, leadership qualities, team spirit and ethical responsibilities.

M5. Provide an academic environment and consultancy services to the industry and society in the area of Computer Engineering.

MICRO-PROJECT REPORT

ON

Hostel Management System.

In Partial fulfilment of Diploma in Computer Engineering

In the subject of

Java Programming (CM3408)

By

Mrs. Bhagyashree Tekade (19CM003)

Mr. Ayush Bulbule (19CM007)

Mr. Pratham Gaur (19CM020)

Mr. Malhar Joshi (19CM033)

Mrs. Akansha Shewatkar (19CM057)

Submitted To

Government Polytechnic, Amravati

(An Autonomous Institute of Govt. of Maharashtra)

Under the guidance of

Mrs. P. R. Satav Sir

Head of Dep. Computer Science and Engineering

Department of Computer Science & Engg.

Government Polytechnic, Amravati

(2020-2021)

Government Polytechnic, Amravati.

(An Autonomous Institute of Govt. of Maharashtra)

Department of Computer Science & Engg.

Certificate

This is to certify that **Mrs. Bhagyashree Tekade (19CM003), Mr. Ayush Bulbule (19CM007), Mr. Pratham Gaur (19CM020), Mr. Malhar Joshi (19CM033), Mrs. Akansha Shewatkar (19CM057)** of Third Semester Diploma in Computer Engineering has satisfactorily completed the micro project entitled "**Hostel Management System**." in **(CM3408) Programming in Java** the academic year 2020-21 as prescribed in curriculum.

Place: Amravati

Date: 01 /06 /2021

Mrs. P. R. Satav

H.O.D Dept. of Computer Science and Engineering

Annexure-I

Title of Micro-Project

Hostel Management System

1.0 Brief Introduction

The project entitled "Develop a 'Java' program for Hostel management System. This project aims to develop a hostel management system in java. The system could be able to add room data to the database (i.e. files).Then it should have functionality to add Student's info, allot room to student, display available rooms, display enrolled students etc.

2.0 Aim of the Micro-Project

This Micro-Project aims at: Developing Hostel Management System

1. Learn Basic Java Programming Concepts
2. Use various concepts of java like exception handling, file handling etc.
3. Develop problem solving program.

3.0 Action Plan (Sequence and time required for major activities for 8 weeks)

S. N.	Details of activity	Planned Start date	Planned Finish date	I. Code &Name of Team Members
1	Gathering Information	04-05-2020	08-05-2021	Bhagyashree Tekade (19CM003)
2	Making report and file	14-05-2021	17-05-2021	Pratham Gaur (19CM020)
3	Preparing the Code of project in Java Lang.	26-05-2021	30-06-2021	Ayush Bulbule (19CM007)
4	Planning proposal submission	12-05-2021	13-05-2021	Akanksha Shewatkar (19CM057)
5	Gathering content	8-05-2020	11-05-2020	Malhar Joshi (19CM033)

4.0 Resources Required (major resources such as raw material, some machining facility, software etc.)

S.N.	Name of Resource/material	Specifications	Remarks
1	Computer System (System with basic configuration)	__	
2.	Java JDK	V 15	
3	VS Code		
4	Command Prompt		
5	M S Word		
6	Windows 10 OS		

5.0 Names of Team Members with Identity Codes :

i. Bhagyashree Tekade (19CM003)
ii. Ayush Bulbule (19CM007)
iii. Pratham Gaur (19CM020)
iv. Malhar Joshi (19CM033)
v. Akanksha Shewatkar (19CM057)

Guideline for Assessment of Micro-Project

Evaluation as per suggested Rubric for Assessment of Micro-Project

Assessment Parameter	Characteristic to be assessed	Average (1 mark)	Good (1.5 mark)	Excellent (2 mark)
Process Assessment (06)	Relevance of the courses & proposals			
	Literature survey/market survey/information collection			
	Analysis of data & completion of the target as per proposal/			
Product Assessment (04)	Report Preparation/Quality of Prototype/model			

Annexure-II

Title of Micro Project

Develop Hostel Management System

1.0 Brief Introduction

The project entitled "Develop a 'Java' program for Hostel management System. This project aims to develop a hostel management system in java. The system could be able to add room data to the database (i.e. files).Then it should have functionality to add Student's info, allot room to student, display available rooms, display enrolled students etc.

2.0 Aim of the Micro-Project

This Micro-Project aims at : 'Developing Hostel management System'

1. Learn concepts Java Programming
2. Develop Real-world Program.

2.0 Course Outcomes Integrated

1. Develop Java Programs to solve different preliminary problems.
2. Develop class in Java Program with the Implementation of Inheritance.
3. Use Array, Strings, Vector, and Interface in Java program to solve associated problems.
4. Create threads to control execution of program in java.
5. Develop Java program to perform file I/O operation.

4.0 Actual Procedure Followed

1) Bhagyashree Tekade**:** Gathered information about the Hostel Management System.

2) Ayush Bulbule: Prepared the Code for the Hostel Management System.

3) Pratham Gaur: Prepared word file related the project with synopsis also.

4) Malhar Joshi: Tested the system and gathered other related info.

5) Akanksha Shewatkar: Planned about and managed submission.

5.0 Actual Resources used (Mention the actual resources used)

S.N.	Name of Resource/material	Specifications	Remarks

1	Computer System (System with basic configuration)	__	
2	Java JDK 14	***version*** 14	
3	CMD	Version	
4	VS Code		
5	M S Word	2019	
6	Windows 10	--	

6.0 Output of the Micro-Project

Output of this Micro-Project is attached to this file.

7.0 Skill Developed / Learning outcomes of this Micro-Project

1. Learned Java Programming.
2. Logic Building in Java Programming.
3. Developing Real-world Programs.

8.0 Assessment by Faculty as per Rubrics

Process Assessment (06)	Product Assessment (04)	Total Marks (10)	Signature of Faculty

****

Develop Hostel Management System.

About Project

The project entitled “Develop a ‘Java’ program for Hostel management System. This project aims to develop a hostel management system in java. The system could be able to add room data to the database (i.e. files).Then it should have functionality to add Student’s info, allot room to student, display available rooms, display enrolled students etc.

Functionalities in Program:

1. Manage Rooms:

This Option is used to Manage Room In hostel. This functionality has three mre options:

1) Add Room:
   This option will add a new room to the database. Room will have characters like: room_number, allotted and student.
2) Search Room:
   This will find room for given room no in the database.
3) Back:
   This option will take user back to the main Menu.

2. Manage Students:

This option will manage all student data related info. Such as enrol new student, edit existing student and deleting student.

1) Add Room:
   This option will add a new student to the database (i.e. file) and it will allot a room to the student.
2) Edit Student:
   This will find the student in the database using student id and will update the student by taking input from user.
3) Delete Student:

This option will take user student' id as input and will delete that student from the database. This will also deallocate the room.

3. **Available Rooms:**

   This will first filnd that room are initialized in database or not and if they are, then it will show the room which status is available. (i.e. rooms which are not allotted yet)

4. **Search Student:**

   This option will ask user to enter student's roll no. Then it will search for the entered roll no the database. If the student is found then it will display the student details and if the student is not found then it will show such message.

5. **Display Students:**

   This option will display all the enrolled students in the hostel.

6. **Student with Unpaid Fees**

   This option will find all enrolled student's and filter the students who have not paid complete fees yet and display the student whose fees is Incomplete.

7. **EXIT**

   This functionality will Save all the data to database and Altert the user that he choose to exit and will terminate the Program.

Working Diagram:

Program Structure:

```
String id;
String name;
String dep;
int sem;
int age;
String dob;
double paidFees;
```

Class Student

```
int roomNumber;

boolean allotte;

Student student;
```

class Room

```
private static final int MAX = 80;
public static final int FEES = 3000;

public static Vector<Room> roomsVector = new Vector(MAX);
```

roomsVector

Class HostelManagement

Program in Java:

```
import java.io.*;
import java.util.Scanner;
import java.util.Vector;

@SuppressWarnings("serial")
class Student implements Serializable {
    public String id;
    public String name;
    public String dep;
    public int sem;
    public int age;
    public String dob;
    public double paidFees;
}

@SuppressWarnings("serial")
class Room implements Serializable {
    public int roomNumber;
    public boolean allotted;
    Student student = new Student();

    public Room(int roomNumber, boolean allotted) {
        this.roomNumber = roomNumber;
        this.allotted = allotted;
```

```
    }

    public void displayRoom() {
        System.out.println("Room No: " + roomNumber);
        System.out.println("Alloted? :" + allotted);
    }

    public void displayDetails() {
        System.out.println("Student Id: " + student.id);
        System.out.println("Student Name: " + student.name);
        System.out.println("Department: " + student.dep);
        System.out.println("Semester: " + student.sem);
        System.out.println("Age: " + student.age);
        System.out.println("Room No: " + roomNumber);
    }

}

public class HostelManagement {
    private static final int MAX = 80;
    public static final int FEES = 3000;
    public static Vector<Room> roomsVector = new
Vector<>(MAX);

    @SuppressWarnings("unchecked")
```

```
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int opt, i;

        File f = null;
        FileInputStream fis = null;
        ObjectInputStream ois = null;
        FileOutputStream fos = null;
        ObjectOutputStream oos = null;

        try {
            f = new
File("C:/HostelManagement/HostelData.dat");

            if (f.exists()) {
                fis = new FileInputStream(f);
                ois = new ObjectInputStream(fis);
                roomsVector = (Vector<Room>)
ois.readObject();
            } else {
                f.createNewFile();
            }
        } catch (Exception ex) {
```

```
            ex.printStackTrace();
        }

        do {
            cls();
            System.out.println("\nProject By -
19CM003,19CM007,19CM020,19CM033 and 19CM057\n");
            System.out.println("########## HOSTEL MANAGEMENT
SYSTEM ##########");
            System.out.println("1. Manage Rooms");
            System.out.println("2. Manage Students");
            System.out.println("3. Available Rooms");
            System.out.println("4. Search Student");
            System.out.println("5. Display Students");
            System.out.println("6. Student with Unpaid
Fees");
            System.out.println("7. EXIT");

            System.out.println("\nEnter Your Option: ");
            opt = sc.nextInt();

            switch (opt) {
                case 1:
                    manageRooms();
                    sc.nextLine();
                    break;
```

```
                case 2:
                    if (roomsVector.isEmpty()) {
                        System.out.println("Room Data Not
Available!!");
                        System.out.println("Please Add Rooms
First!");
                    } else {
                        manageStudents();
                        break;
                    }
                    sc.nextLine();
                    break;

                case 3:
                    i = 1;
                    if (roomsVector.isEmpty()) {
                        System.out.println("No Student
added!!");
                        System.out.println("Please Add Rooms
First!");
                    } else {
                        int found = 0;
                        for (Room r : roomsVector) {
                            if (!r.allotted) {
```

```
                                System.out.println(i + ")
Room No: " + r.roomNumber);
                                System.out.println();
                                found++;
                                i++;
                            }
                        }
                        if (found == 0) {
                            System.out.println("All Rooms are
Allotted!");
                            sc.next();
                        }
                        sc.nextLine();
                    }
                    sc.nextLine();
                    break;

                case 4:
                    if (roomsVector.isEmpty()) {
                        System.out.println("No Student
added!!");
                        System.out.println("Please Add Rooms
Frist!");
                    } else {
                        System.out.println("Enter Student Id:
");
```

```
                        sc.nextLine();
                        String id = sc.nextLine();
                        int found = 0;
                        if (roomsVector.isEmpty()) {
                            System.out.println("Student
Details Not Available");
                        } else {
                            for (Room r : roomsVector) {
                                if (id.equals(r.student.id))
{

System.out.println("\n\nSearch Results: \n");
                                    r.displayDetails();
                                    found++;
                                }
                            }
                            sc.nextLine();
                        }
                        if (found == 0) {
                            System.out.println("Student Not
Found!!");
                        }
                        sc.nextLine();
                    }
                    sc.nextLine();
                    break;
```

```
                case 5:
                    i = 1;
                    if (roomsVector.isEmpty()) {
                        System.out.println("Student Details
Not Available");
                    } else {
                        for (Room r : roomsVector) {
                            if (r.allotted) {

System.out.println("\n\nSerial No: " + i);
                                r.displayDetails();
                                i++;
                            }
                        }
                        sc.nextLine();
                    }
                    sc.nextLine();
                    break;

                case 6:
                    i = 1;

                    if (roomsVector.isEmpty()) {
                        System.out.println("Student Details
Not Available");
```

```
                    } else {
                        for (Room r : roomsVector) {
                            if (r.student.paidFees != FEES &&
r.allotted) {

System.out.println("\n\nSerial No: " + i);
                                System.out.println("Student
Id: " + r.student.id);
                                System.out.println("Student
Name: " + r.student.name);
                                System.out.println("Room Id:
" + r.roomNumber);
                                System.out.println("REMAINING
FEES: " + (FEES - r.student.paidFees));
                                i++;
                            }


                        }
                        sc.nextLine();
                    }
                    sc.nextLine();
                    break;


                case 7:
                    try {
                        fos = new FileOutputStream(f);
                        oos = new ObjectOutputStream(fos);
```

```
                    oos.writeObject(roomsVector);
                } catch (Exception ex) {
                    ex.printStackTrace();
                } finally {
                    try {
                        fis.close();
                        ois.close();
                        fos.close();
                        oos.close();
                        sc.close();
                    } catch (Exception e1) {
                        e1.printStackTrace();
                    }
                }

                System.out.println("\n You Choose To
EXIT!! All are saved!");
                System.out.println("########## Thanks for
Using ############");
                System.exit(0);
                break;
            default:
                System.out.println("Please Enter correct
Option: ");
                break;
```

```
            }
        } while (opt != 7);
    }

    private static void manageRooms() {
        Scanner sc = new Scanner(System.in);
        int ch;
        int rno;

        do {
            cls();
            System.out.println("####### Manage Rooms
#######");
            System.out.println("Enter 1: Add Room");
            System.out.println("Enter 2: Search Room");
            System.out.println("Enter 3: Back To Main menu");

            System.out.println("\nEnter Your Choise: ");
            ch = sc.nextInt();

            switch (ch) {
                case 1:
                    int found = 0;
                    System.out.println("Enter room no: ");
                    rno = sc.nextInt();
```

```
                    for (Room r : roomsVector) {
                        if (rno == r.roomNumber) {
                            found++;
                        }
                    }
                    if (found != 0) {
                        System.out.println("Room is Already
Added!!");
                        sc.nextLine();
                        sc.nextLine();
                    } else {
                        roomsVector.add(new Room(rno,
false));
                        System.out.println("Room Added
Successfully!!");
                        sc.nextLine();
                        System.out.println(roomsVector);
                        sc.nextLine();
                    }
                    break;
                case 2:
                    found = 0;
                    System.out.println("Enter room no: ");
                    rno = sc.nextInt();
                    for (Room r : roomsVector) {
                        if (rno == r.roomNumber) {
```

```
                            System.out.println("Room Details:
");
                            if (r.allotted) {
                                System.out.println("Room is
reserved!!");
                                sc.nextLine();

                            } else {
                                System.out.println("Room is
Available");
                                sc.nextLine();

                            }
                            r.displayRoom();
                            found++;
                        }
                    }
                    sc.nextLine();
                    if (found == 0) {
                        System.out.println("Room Not
found!!");
                        sc.nextLine();
                    }
                    sc.nextLine();
                    break;
                case 3:
```

```
                    break;
                default:
                    System.out.println("Select Correct
Option!!");
                    sc.nextLine();
            }

        } while (ch != 3);
    }

    private static void manageStudents() {
        Scanner sc = new Scanner(System.in);

        int ch;
        int rno;

        do {
            cls();
            System.out.println("####### Manage Students
#######");
            System.out.println("Enter 1: Enroll Student");
            System.out.println("Enter 2: Edit Student");
            System.out.println("Enter 3: Delete Student");
            System.out.println("Enter 4: Back To Main menu");
```

```
            System.out.println("\nEnter Your Choise: ");
            ch = sc.nextInt();
            String id;

            switch (ch) {
                case 1:
                    int found = 0;
                    System.out.println("Enter room no: ");
                    rno = sc.nextInt();
                    for (Room r : roomsVector) {
                        if (r.roomNumber == rno) {
                            if (r.allotted) {
                                System.out.println("Room not
Available! Already Allotted!");
                                sc.nextLine();
                            } else {
                                System.out.println("Room
Available ");
                                sc.nextLine();
                                System.out.println("\nEnter
Student Details: \n");
                                System.out.println("Enter
Student Id: ");
                                r.student.id = sc.nextLine();
```

```
                                    System.out.println("Enter
Student Name: ");
                                    r.student.name =
sc.nextLine();
                                    System.out.println("Enter
Student age: ");
                                    r.student.age = sc.nextInt();
                                    sc.nextLine();
                                    System.out.println("Enter
Department: ");
                                    r.student.dep =
sc.nextLine();
                                    System.out.println("Enter
Semester: ");
                                    r.student.sem = sc.nextInt();
                                    sc.nextLine();
                                    System.out.println("Enter
Student DOB: ");
                                    r.student.dob =
sc.nextLine();
                                    System.out.println("Enter
Fees Paid: ");
                                    r.student.paidFees =
sc.nextDouble();
                                    r.allotted = true;

                                    System.out.println("Student
added Success!");
```

```
                                sc.nextLine();
                                found++;
                            }
                        }
                    }

                    if (found == 0) {
                        System.out.println("No such Room! Add
Room First!");
                        sc.nextLine();
                    }
                    break;
                case 2:
                    found = 0;
                    sc.nextLine();
                    System.out.println("Enter Student Id: ");
                    id = sc.nextLine();
                    for (Room r : roomsVector) {
                        if (id.equals(r.student.id)) {
                            System.out.println("\nCurrent
Student Data:\n");
                            r.displayDetails();
                            System.out.println("Enter new
Data to edit: ");
```

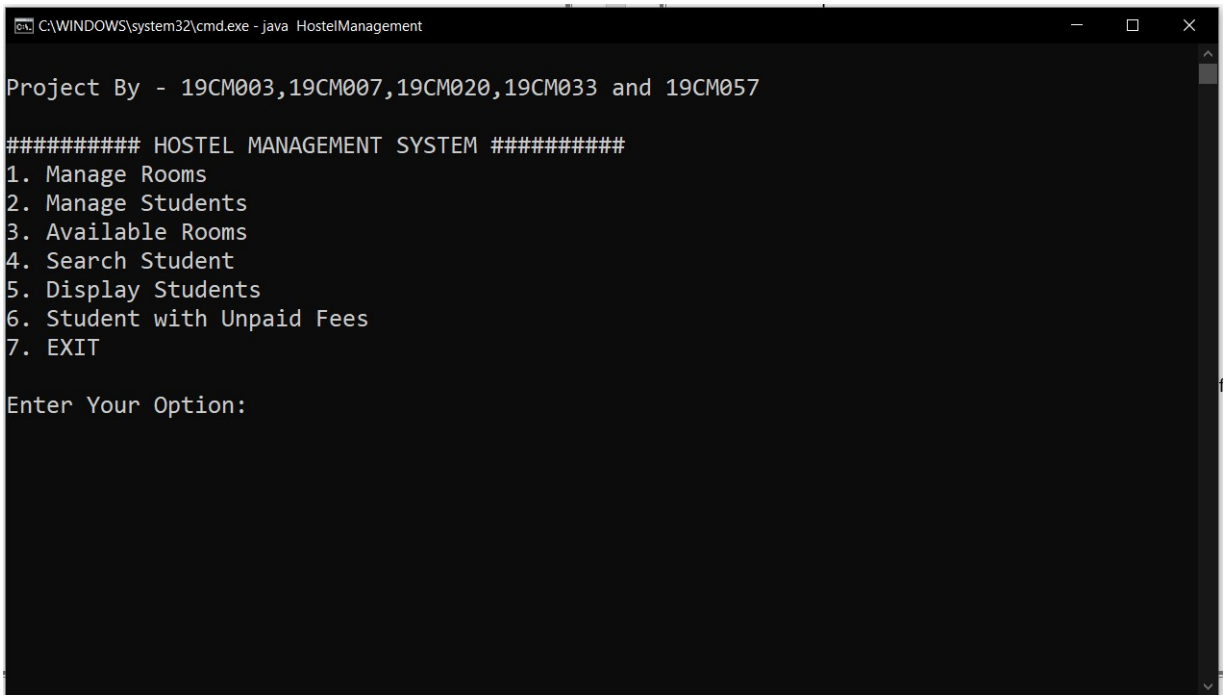
```
                            System.out.println("Enter Student
Id: ");
                            r.student.id = sc.nextLine();
                            System.out.println("Enter Student
Name: ");
                            r.student.name = sc.nextLine();
                            System.out.println("Enter Student
age: ");
                            r.student.age = sc.nextInt();
                            sc.nextLine();
                            System.out.println("Enter
Department: ");
                            r.student.dep = sc.nextLine();
                            System.out.println("Enter
Semester: ");
                            r.student.sem = sc.nextInt();
                            sc.nextLine();
                            System.out.println("Enter Student
DOB: ");
                            r.student.dob = sc.nextLine();
                            System.out.println("Enter Fees
Paid: ");
                            r.student.paidFees =
sc.nextDouble();
                            r.allotted = true;

                            System.out.println("Student
Edited Successfully!");
```

```
                        sc.nextLine();
                        found++;
                        break;
                    }
                }
                if(found==0){
                    System.out.println("Student not
found!");
                    sc.nextLine();
                }
                sc.nextLine();
                break;
            case 3:
                found = 0;
                System.out.println("Enter Student Id to
Delete: ");
                sc.nextLine();
                id = sc.nextLine();
                for (Room r : roomsVector) {
                    if (id.equals(r.student.id)) {
                        r.student.id = null;
                        r.student.name = null;
                        r.student.age = 0;
                        r.student.sem = 0;
                        r.student.paidFees = 0;
```

```
                            r.student.dep = null;
                            r.student.dob = null;
                            r.allotted=false;
                            found++;
                            break;
                        }
                    }
                    if(found==0){
                        System.out.println("No such
Student");
                    }
                    System.out.println("Student Removed!!");
                    sc.nextLine();
                    break;
                case 4:

                    break;
                default:
                    System.out.println("Select Correct
Option!!");
                    sc.nextLine();
            }

        } while (ch != 4);
    }
```

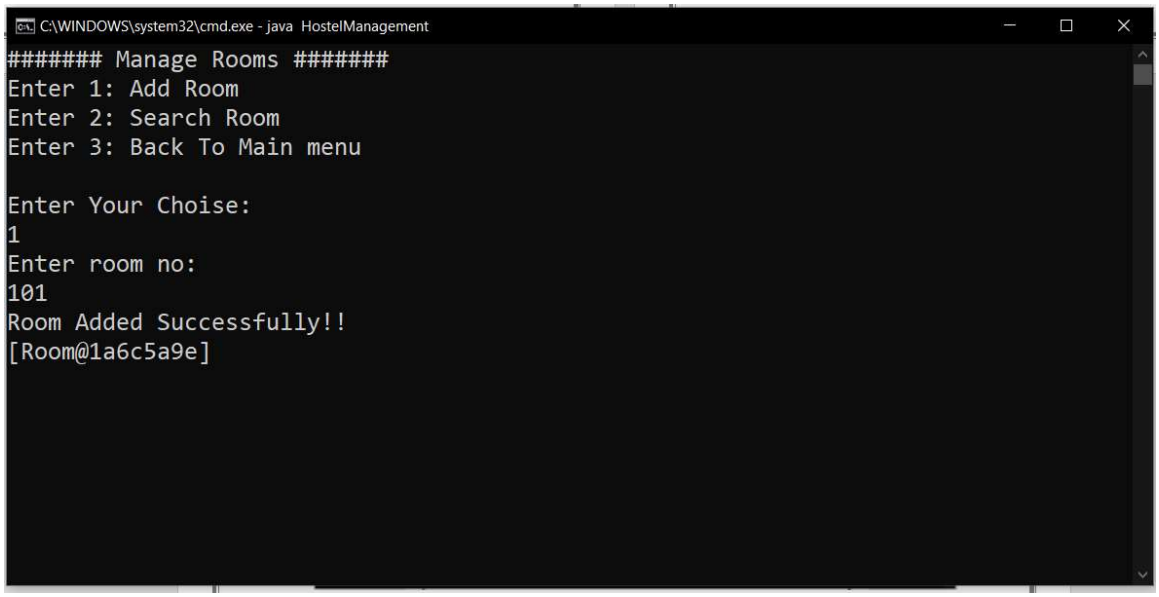
```
    public static void cls() {
        try {
            if
(System.getProperty("os.name").contains("Windows"))
                new ProcessBuilder("cmd", "/c",
"cls").inheritIO().start().waitFor();
        } catch (IOException | InterruptedException ex) {
            ex.printStackTrace();
        }
    }

}
```

Output:

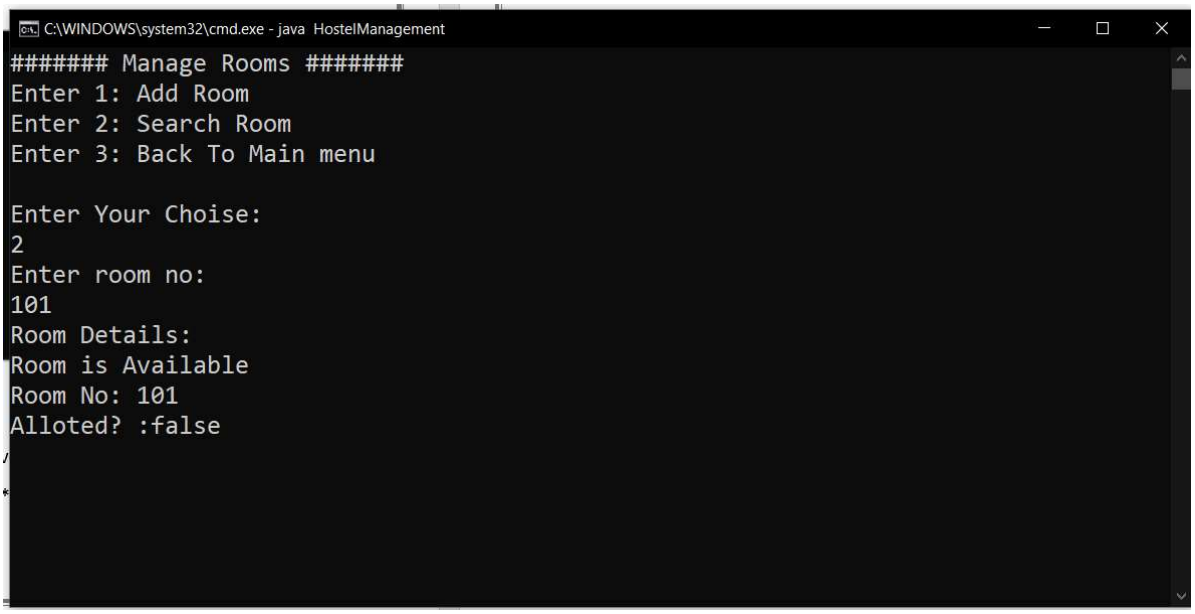


1.1 Manage Rooms

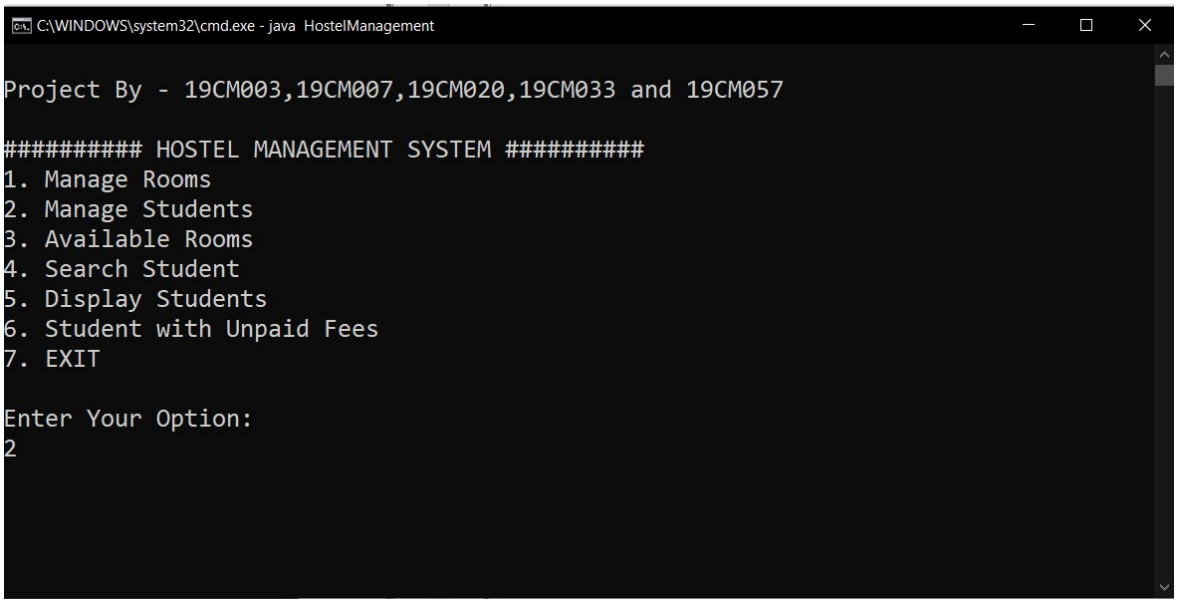
1.2 Add Room:



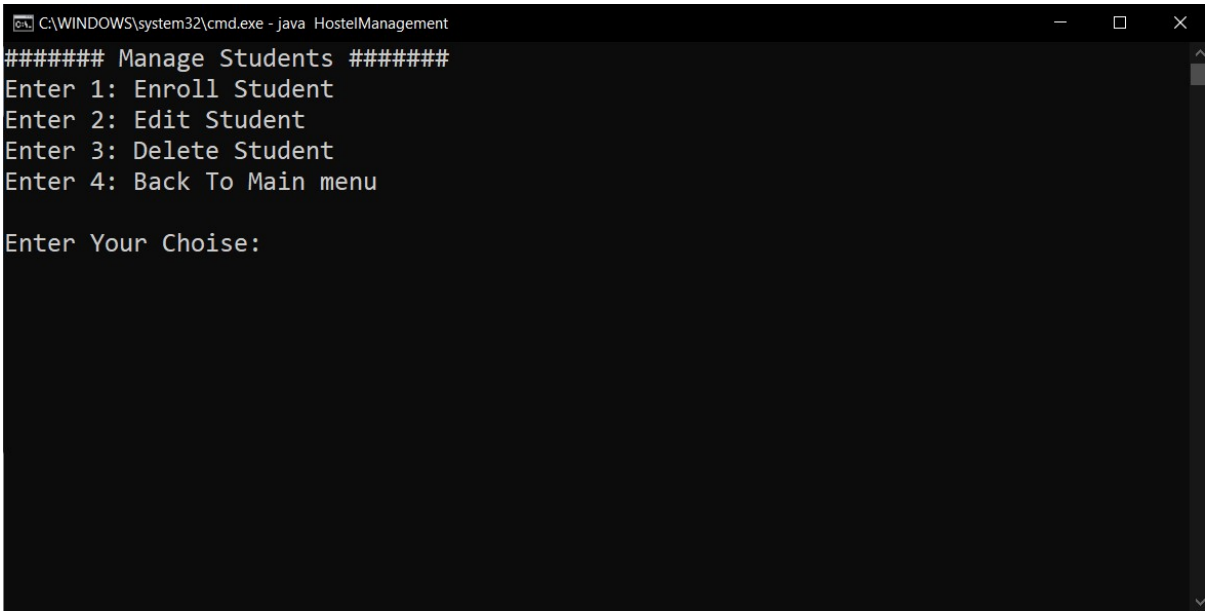
1.3



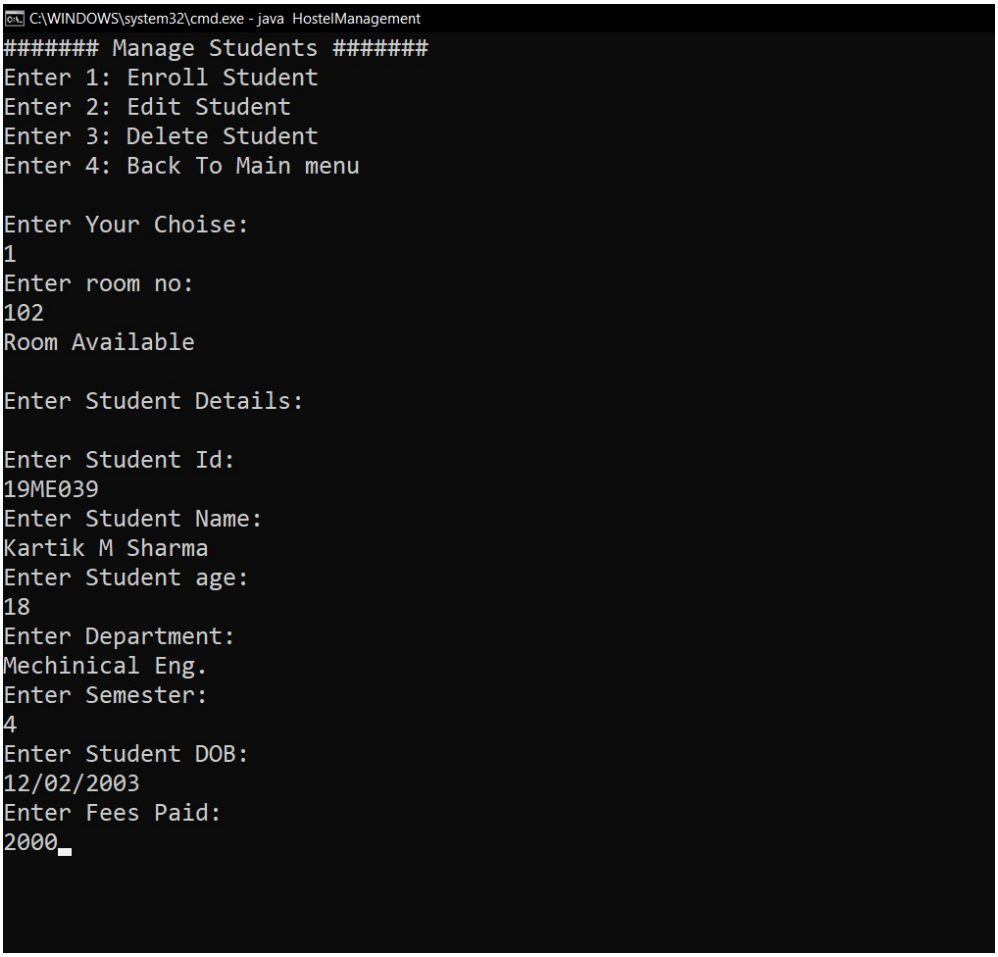
2 a



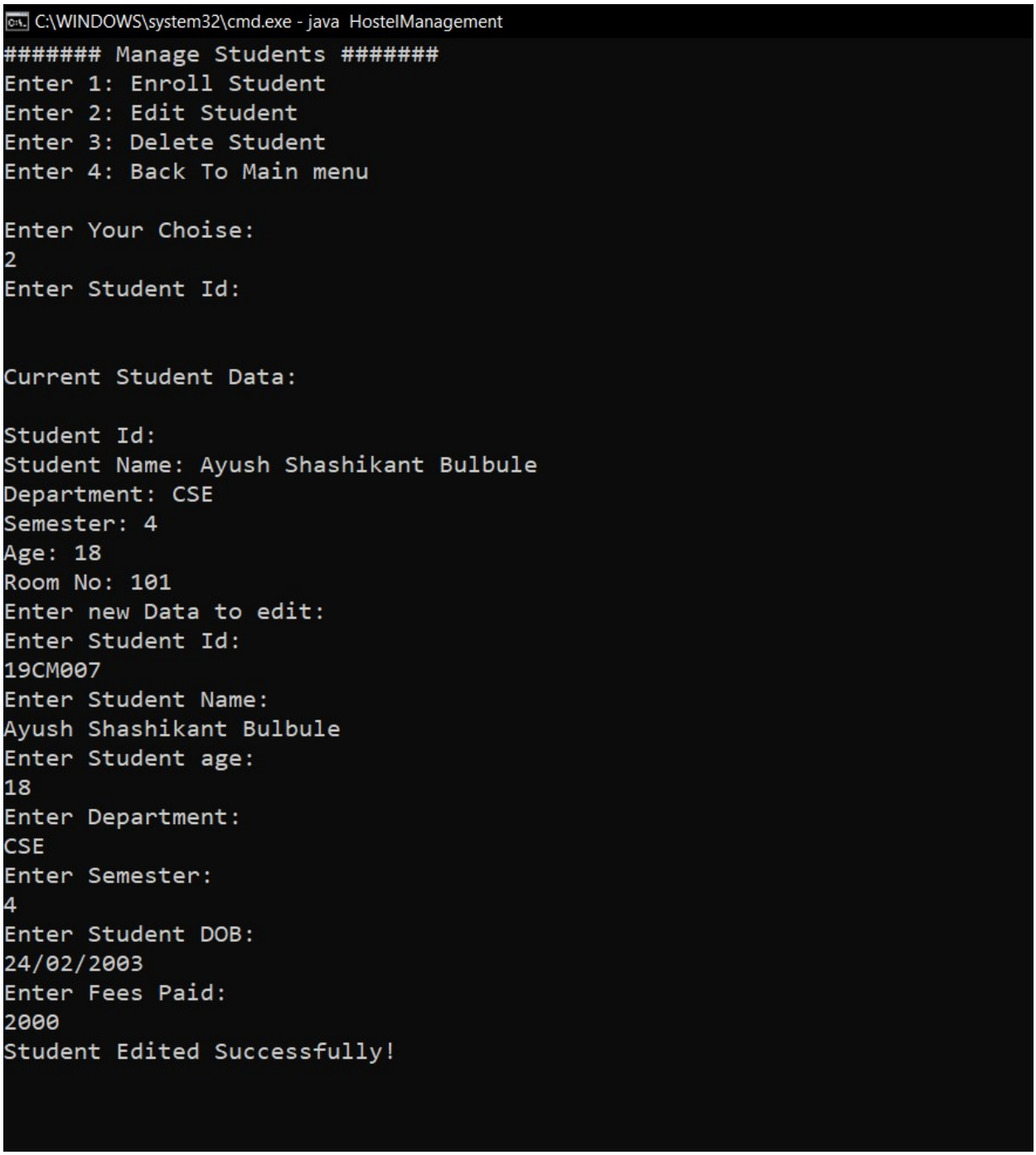
2.1 Manage Students



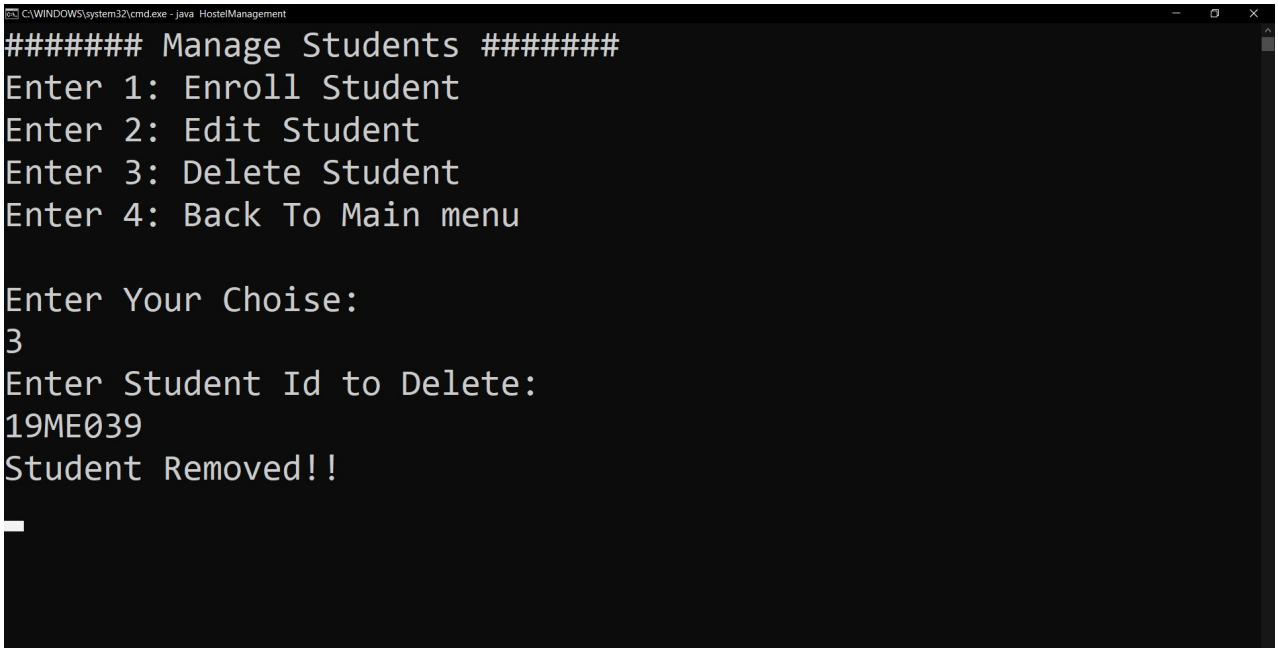
2.2 Enrol Student



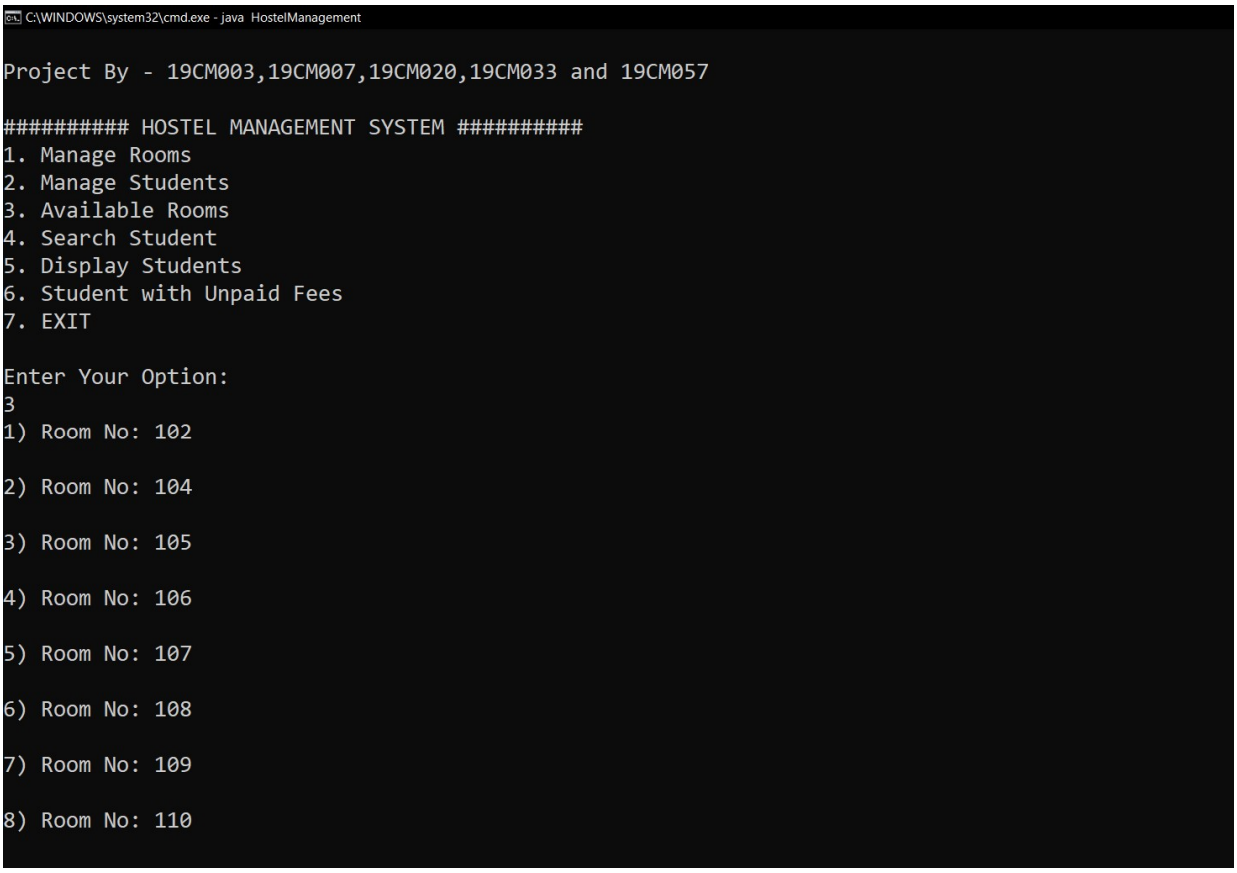
2.3 Edit Student



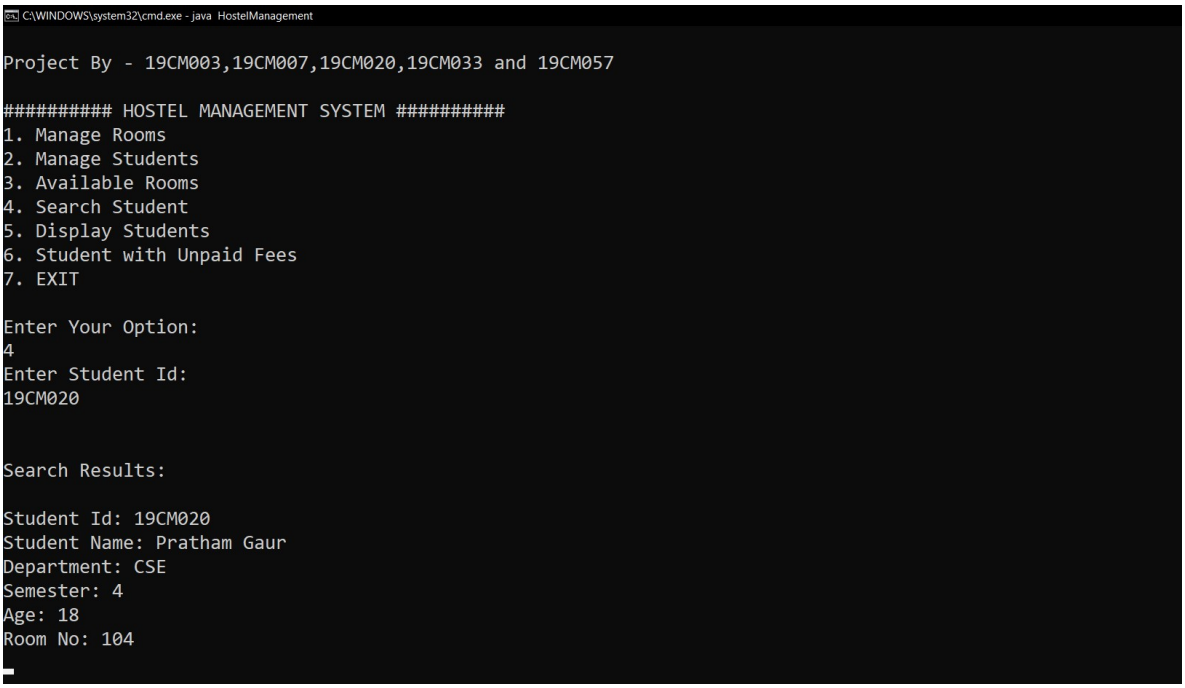
2.4 Delete Student



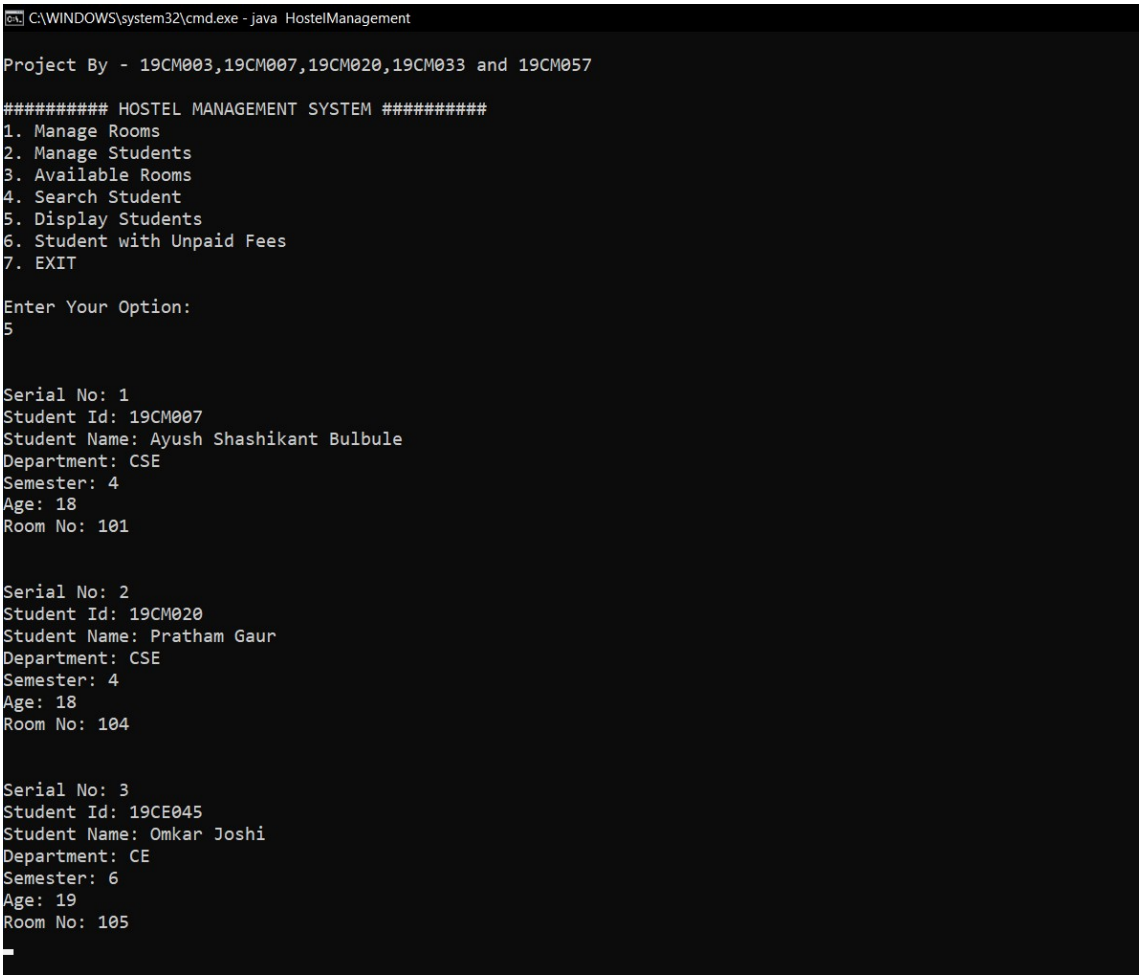
3 Available Rooms



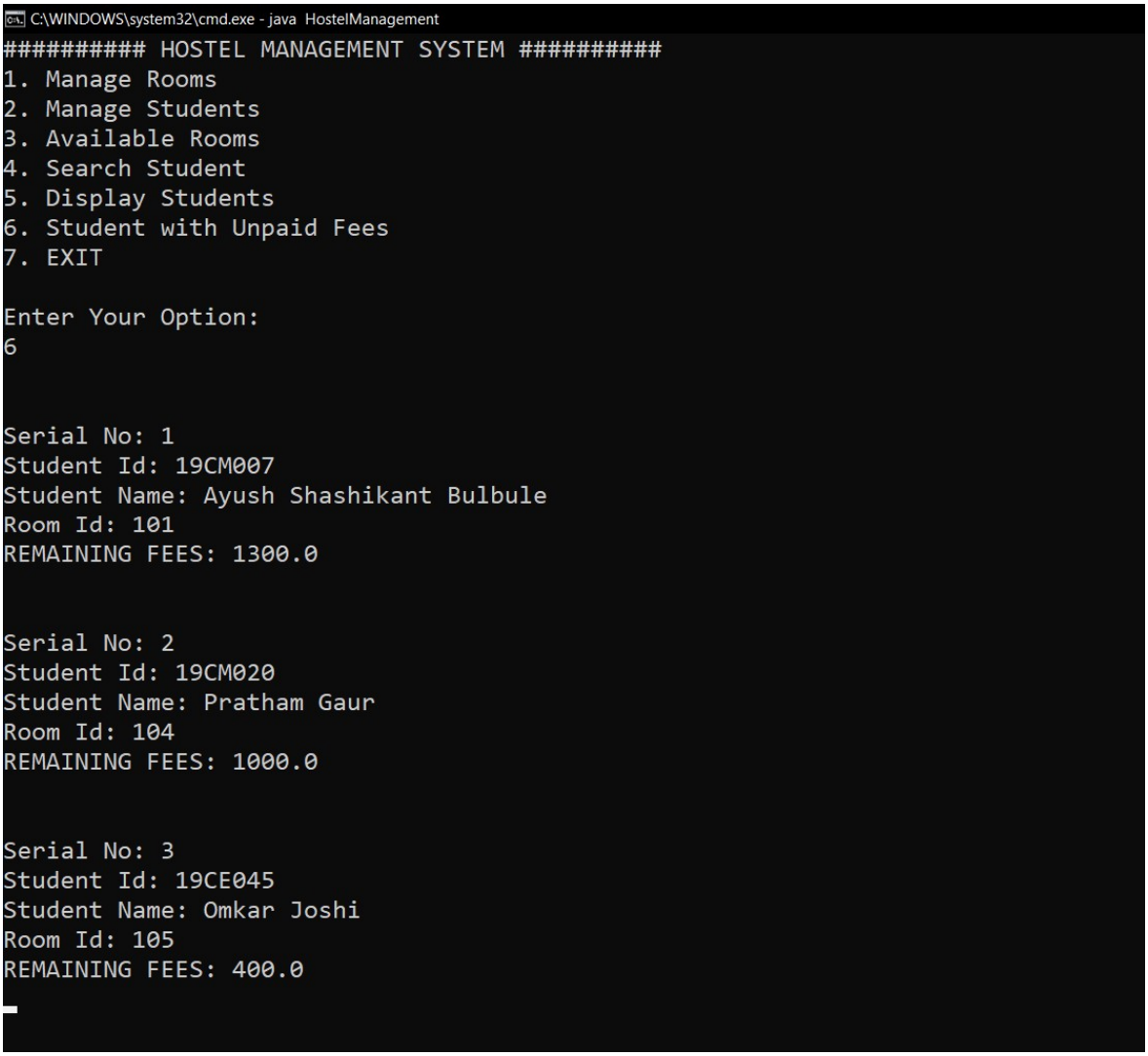
4 Search Student



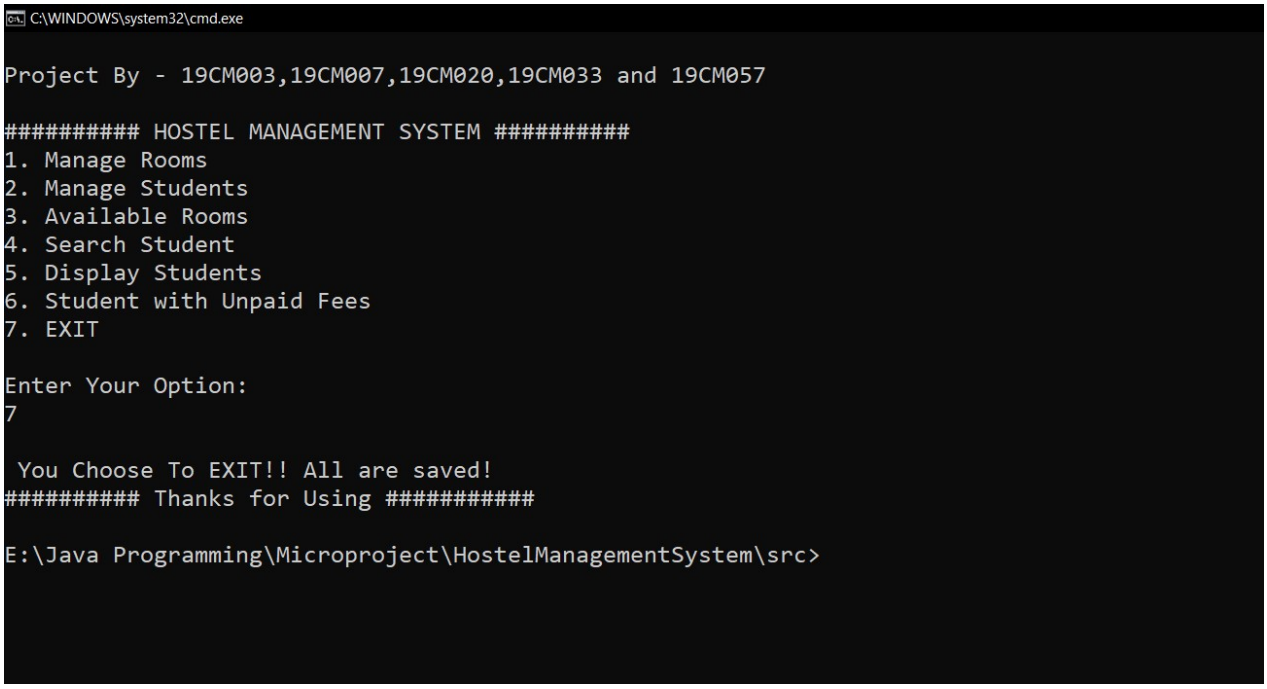
5 Display Students



6 Student with Unpaid Fees


C:\WINDOWS\system32\cmd.exe - java HostelManagement
########## HOSTEL MANAGEMENT SYSTEM ##########
1. Manage Rooms
2. Manage Students
3. Available Rooms
4. Search Student
5. Display Students
6. Student with Unpaid Fees
7. EXIT
Enter Your Option:
6
Serial No: 1
Student Id: 19CM007
Student Name: Ayush Shashikant Bulbule
Room Id: 101
REMAINING FEES: 1300.0
Serial No: 2
Student Id: 19CM020
Student Name: Pratham Gaur
Room Id: 104
REMAINING FEES: 1000.0
Serial No: 3
Student Id: 19CE045
Student Name: Omkar Joshi
Room Id: 105
REMAINING FEES: 400.0

7 EXIT



Conclusion: Hostel Management System developed Successfully.

****