Namal University, Mianwali

Department of Electrical Engineering

CSC-101L- Object Oriented Programming (Lab)

OOP Design Project
Namal Hostel Management System using OOP

Group Members: Hassan Ali (NUM-BSEE-2023-03)
Muskan Aman Khan (NUM-BSEE-2022-04)
Kishwar Raza (NUM-BSEE-2022-39)

Submission Date: 1/16/2024

Course Instructor: **Naureen Shaukat**

Lab Instructor: **Majid Ali**

Contents

1 Abstract

This project shows an efficient hostel management system for Namal University. Using object-oriented programming principles, the system manages student, faculty, and guest accommodations. Room allocation, dues management, migration, and data storage via CSV files are among the features. The system offers a user-friendly interface and proves effective in hostel operations.

2 Introduction

For colleges and universities, managing hostel housing is important. The difficulties in allocating rooms, collecting fees, and guaranteeing proper use of resources are all dealt with in this project. Applying object-oriented programming techniques, the C++-implemented system offers an expandable and effective solution.

3 Methodology

3.1 System Design

The system employs modular programming with the following components:

1. **Object-Oriented Design (OOP):**
    - Key classes: Room, Person, student _hostel, facultystaff hostel, and guest rooms.
    - Inheritance and polymorphism enable feature reuse, such as room allocation and status checks.
    - Encapsulation ensures data security.
2. **Room Management:**
    - Arrays (rooms[], room[], g room[]) manage room availability for different categories.
    - Core tasks include allocation, freeing, and availability checks.
3. **File Management:**
    - Data is stored securely in CSV files.
    - Functions ensure consistency through data loading, saving, and updating.

3.2 Team Contributions

- **Hassan Ali:**

  - Focused on the student section: room allocation, migration, and dues management.
  - Key functions: addStudent(), change room(), displayAvailableRooms().

- **Muskan Aman:**
  - Managed faculty sections: room allocation and hostel exit processes.
  - Key functions: add _faculty or staff(),leave hostel().

- **Kishwar Raza:**
  - Managed Guest sections: room allocation and hostel exit processes.
  - Key functions: add guest(), leave hostel().

3.3 Workflow

3.3.1 Simple Menu System

o Users can manage rooms and data through an easy-to-use menu with separate options for students, faculty, and guests.

3.3.2 Error Checks and Validation

o Ensures only valid inputs are accepted, and errors like incorrect details or file issues are flagged.

3.3.3 Billing and Payments

o Users can pay dues immediately or later, with the system keeping track of pending amounts.

3.3.4 Room Changes and Moves

o Allows users to change rooms or move while managing any pending dues.

4 Implementation Details

The implementation is divided into interfaces for students, faculty/staff, and guests, each offering specific functionality.

4.1 Main Interface Code

```
int main() {
        system("color
        f1");
        student_hostel stdt_hostel("0", "0", "0", "0", "0", "0"); facultystaff_hostel
        fclty_hostel("0", "0", "0", "0", "0"); guest_rooms gst_room("0", "0", "0",
        "0");

        stdt_hostel.load_student_hostel();
        fclty_hostel.load_Faculty_and_staff_hostel();
        gst_room.load_guest_room(); system("CLS");

        int choice; do { cout << "\nEnter ’1’ for student" << endl;
        cout << "Enter ’2’ for Faculty/staff" << endl; cout << "Enter
        ’3’ for guest house" << endl; cout << "Enter ’0’ to exit from
        program" << endl; cout << "Enter choice: "; cin >> choice;
        system("CLS");

                switch (choice) { case 1: /* Student Menu */ break; case 2: /*
                        Faculty/Staff Menu */ break; case 3: /* Guest Menu */ break;
                        case 0: cout << "Thank you for using our services." << endl;
                        break;
                        default:
                                cout << "Invalid choice. Please try again." << endl; }
        } while (choice != 0);

        return 0;
```

4.2 Adding a Student

```
void student_hostel::addStudent() { string name, id,
        contact, roll_no, department, year; int room_no;

        cout << "Enter student details:" <<
        endl; cout << "Name : "; cin.ignore();
        getline(cin, name); cout << "ID : ";
        cin >> id;
        cout <<
        "Contact : ";
        cin >>
        contact; cout
        << "Roll No :
        "; cin >>
        roll_no; cout
        <<
        "Department :
        "; cin >>
        department;
        cout << "Year
        : "; cin >>
        year;

        do { cout << "Room No
               (1-200) : "; cin >>
               room_no;
        } while (room_no > 200 || room_no <= 0);

        float bill = 12000, total; cout << "Do you want to take the inverter
        facility? (yes/no): "; string option; cin >> option; total = (option ==
        "yes") ? bill + 5000 : bill;

        cout << "Total charges for hostel: " << total << endl; save_student_hostel(name, id,
        contact, roll_no, department, year, room_no,
         t
}
```

4.3 Changing the Room (Student)

```
int student_hostel::change_room()
        { string student_id; cout <<
        "Enter your student ID: "; cin
        >> student_id;

        int old_room_no = -1, new_room_no; bool
        room_found = false; for (int i = 0; i <
        student_capacity; i++) { if
        (rooms[i].getOccupantId() == student_id) {
        old_room_no = rooms[i].getRoomNo();
        rooms[i].freeRoom(); room_found = true; break;
               }
        }

        if (!room_found) { cout << "Student not found or room already
               freed." << endl; return 1;
        }

        for (int i = 0; i < student_capacity; i++) { if
               (rooms[i].isAvailable()) {
               new_room_no =
               rooms[i].getRoomNo();
               rooms[i].allocateRoom(student_id);
               break;
```

```
                }
        }

        update_data(student_id, new_room_no, false);
        cout << "Room changed successfully!" << endl;
}
```

4.4 Migrating (Student)

```
int student_hostel::migration() { string
        student_id; cout << "Enter your student
        ID: "; cin >> student_id;

        for (int i = 0; i < student_capacity; i++) { if (rooms[i].getOccupantId() ==
                student_id) { if (rooms[i].get_dues() == 0) { rooms[i].freeRoom();
                update_data(student_id, rooms[i].getRoomNo(), true); cout << "Migration
                successful!" << endl;
                        } else { cout << "You have pending dues." << endl;
                        }
                        bre
                        ak;
                }
        }
}
```

4.5 Display Data of Student

```
void student_hostel::load_student_hostel() { ifstream
        load_stdt("save_student_hostel.csv"); if
        (load_stdt.is_open()) { while (getline(load_stdt, name,
        ’,’)) { getline(load_stdt, id, ’,’);
                        // Further processing...
                } load_stdt.close();
        }
}
```

4.6 Displaying Available Rooms (Student)

```
void student_hostel::displayAvailableRooms() { for (int i
        = 0; i < student_capacity; i++) { if
        (rooms[i].isAvailable()) {
        rooms[i].displayRoom("student");
                }
        }
}
```

4.7 Adding a Faculty/Staff

```
void facultystaff_hostel::add_faculty_or_staff() { string name,
        id, contact, department, position; int room_no;

        cout << "Enter faculty/staff details:" << endl; cout <<
        "Name: "; cin.ignore(); getline(cin, name); cout <<
        "ID: "; cin >> id; cout << "Contact: "; cin >> contact;
        cout << "Department: "; cin >> department; cout <<
        "Position: "; cin >> position;

        do { cout << "Room No (1-20):
                "; cin >> room_no;
        } while (room_no > 20 || room_no <= 0);

        save_faculty_hostel(name, id, contact, department, position, room_no, 12000.
        cout << "Faculty/staff added successfully!" << endl;
}
```

4.8 Changing the Room (Faculty/Staff)

```
int facultystaff_hostel::change_room() { string
        ID; cout << "Enter your ID: "; cin >> ID;

        int old_room_no = -1, new_room_no; bool
        room_found = false; for (int i = 0; i < capacity;
        i++) { if (room[i].getOccupantId() == ID) {
        old_room_no = room[i].getRoomNo();
        room[i].freeRoom(); room_found = true; break;
                }
        }

        if (!room_found) { cout << "Faculty/staff not found or room already freed." << endl;
                return 1;
        }

        for (int i = 0; i < capacity; i++) { if
                (room[i].isAvailable()) { new_room_no =
                room[i].getRoomNo();
                room[i].allocateRoom(ID); break;
                }
        }

        update_data(ID, new_room_no, false); cout << "Room
        changed successfully!" << endl;
}
```

4.9 Leaving the Hostel (Faculty/Staff)

```
int facultystaff_hostel::leave_hostel() {
        string id; cout << "Enter your ID: ";
        cin >> id;
        for (int i = 0; i < capacity; i++) { if
                (room[i].getOccupantId() == id) {
                room[i].freeRoom(); update_data(id,
                room[i].getRoomNo(), true); cout << "Room freed
                successfully!" << endl; return 0;
                } } cout << "Faculty/staff not found."
        << endl; return 1;
}
```

4.10 Display Data of Faculty

```
void facultystaff_hostel::load_Faculty_and_staff_hostel() { ifstream
        load_fclty("save_faculty_and_staff_hostel.csv"); if
        (load_fclty.is_open()) { while (getline(load_fclty, name, ’,’)) {
        getline(load_fclty, id, ’,’); // Further processing...
                }
                load_fclty.close();
        }
}
```

4.11 Displaying Available Rooms (Faculty/Staff)

```
void facultystaff_hostel::displayAvailableRooms()
        { for (int i = 0; i < capacity; i++) { if
        (room[i].isAvailable()) {
        room[i].displayRoom("faculty/staff");
                }
        }
}
```

4.12 Adding a Guest

```
void guest_rooms::add_guest() { string
        name, id, contact, reason_of_stay; int
        room_no; cout << "Enter guest
        details:" << endl;
        cout << "Name: ";
        cin.ignore();
        getline(cin, name);
        cout << "ID: "; cin
        >> id; cout <<
        "Contact: "; cin >>
        contact; cout <<
        "Reason of stay: ";
        cin >>
        reason_of_stay;

        do { cout << "Room No
                (1-3): "; cin >>
                room_no;
        } while (room_no > 3 || room_no <= 0);

        save_guest_room(name, id, contact, reason_of_stay, room_no); cout <<
        "Guest added successfully!" << endl;
}
```

4.13 Changing the Room (Guest)

```
int guest_rooms::change_room() {
        string ID; cout << "Enter your
        ID: "; cin >> ID;
        int old_room_no = -1, new_room_no; bool
        room_found = false; for (int i = 0; i < capacity; i++) {
        if (g_room[i].getOccupantId() == ID) { old_room_no =
        g_room[i].getRoomNo(); g_room[i].freeRoom();
        room_found = true; break;
                }
        }

        if (!room_found) { cout << "Guest not found or room already freed." << endl;
                return 1;
        }

        for (int i = 0; i < capacity; i++) { if
                (g_room[i].isAvailable()) {
                new_room_no =
                g_room[i].getRoomNo();
                g_room[i].allocateRoom(ID); break;
                }
        }

        update_data(ID, new_room_no, false); cout << "Room
        changed successfully!" << endl;
}
```

4.14 Leaving the Hostel (Guest)

```
int guest_rooms::leave_hostel() { string
        id; cout << "Enter your ID: "; cin
        >> id;

        for (int i = 0; i < capacity; i++) { if (g_room[i].getOccupantId() ==
                id) { g_room[i].freeRoom(); update_data(id,
                g_room[i].getRoomNo(), true); cout << "Room freed
                successfully!" << endl; return 0;
                }
        } cout << "Guest not found." << endl;
        return 1;
}
```

4.15 Displaying List of Guests

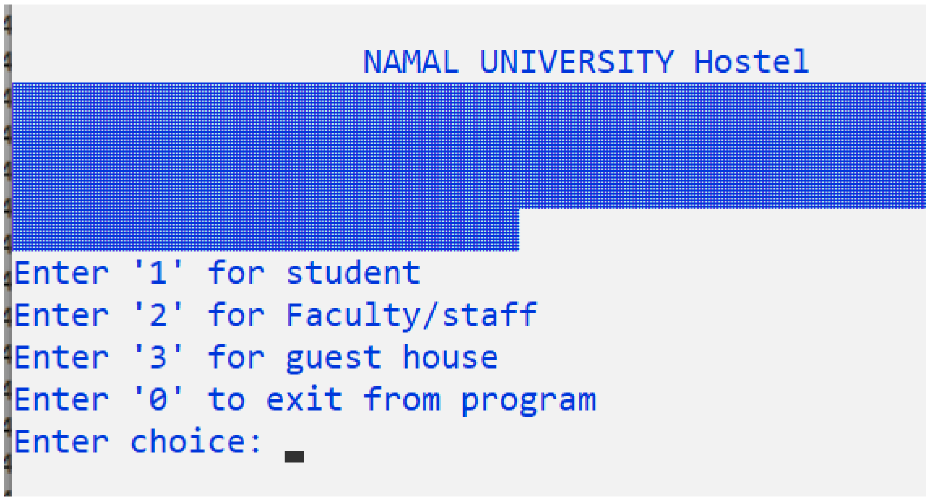
```
void guest_rooms::load_guest_room() { ifstream
        load_gst("save_guest_rooms.csv"); if
        (load_gst.is_open()) { while (getline(load_gst,
        name, ’,’)) { getline(load_gst, id, ’,’); // Further
        processing...
                }
                load_gst.close();
        }
}
```

4.16 Displaying Available Rooms (Guest)

```
void
        guest_rooms::displayAvailableRo
        oms() { for (int i = 0; i < capacity;
        i++) { if (g_room[i].isAvailable())
        {
        g_room[i].displayRoom("guest");
                }
        }
}
```

5 Result

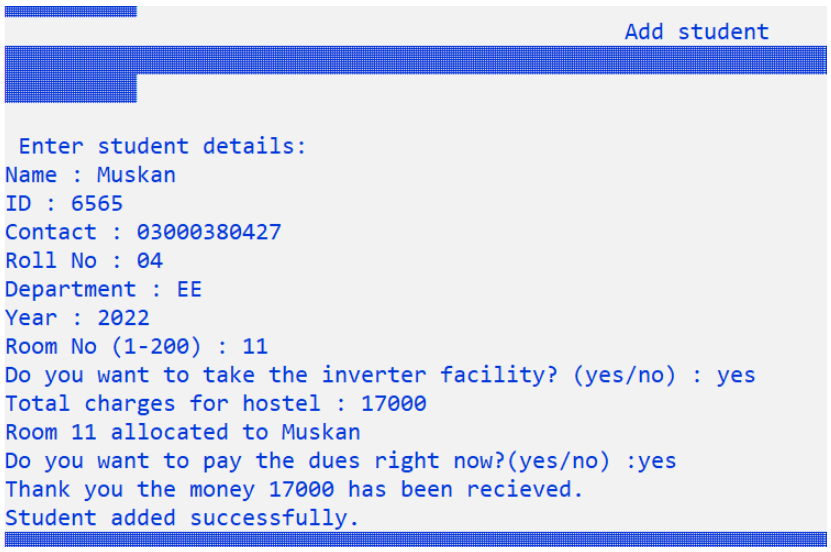
5.1 Main Interface



```
                    NAMAL UNIVERSITY Hostel

Enter '1' for student
Enter '2' for Faculty/staff
Enter '3' for guest house
Enter '0' to exit from program
Enter choice: _
```

Figure 1: Main Interface

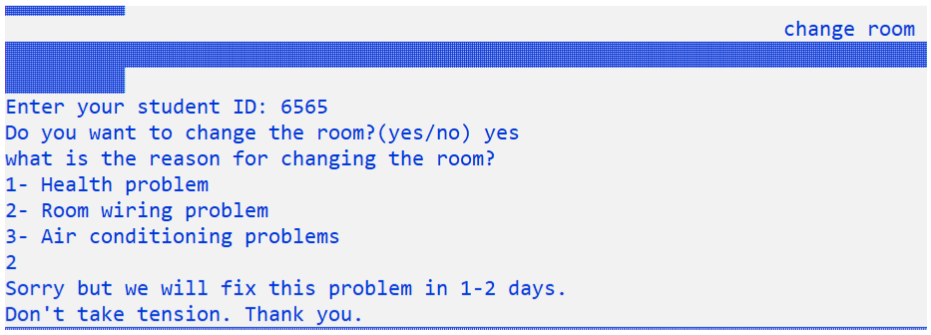
5.2 Adding a Student



```
Add student

 Enter student details:
Name : Muskan
ID : 6565
Contact : 03000380427
Roll No : 04
Department : EE
Year : 2022
Room No (1-200) : 11
Do you want to take the inverter facility? (yes/no) : yes
Total charges for hostel : 17000
Room 11 allocated to Muskan
Do you want to pay the dues right now?(yes/no) :yes
Thank you the money 17000 has been recieved.
Student added successfully.
```

Figure 2: Output

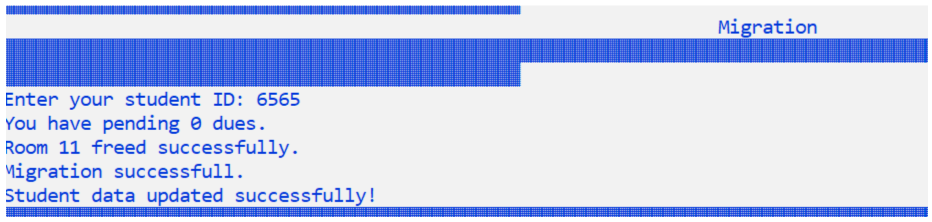
5.2.1 Changing the Room (Student)



```
change room

Enter your student ID: 6565
Do you want to change the room?(yes/no) yes
what is the reason for changing the room?
1- Health problem
2- Room wiring problem
3- Air conditioning problems
2
Sorry but we will fix this problem in 1-2 days.
Don't take tension. Thank you.
```

Figure 3: Output

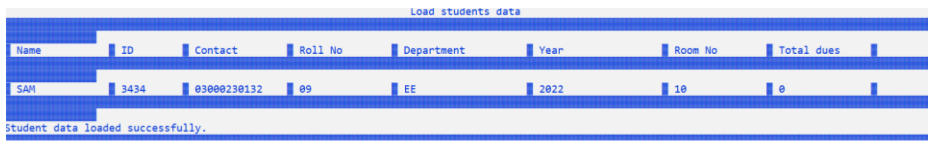
5.2.2 Migrating (Student)



```
Migration

Enter your student ID: 6565
You have pending 0 dues.
Room 11 freed successfully.
Migration successfull.
Student data updated successfully!
```

Figure 4: Output

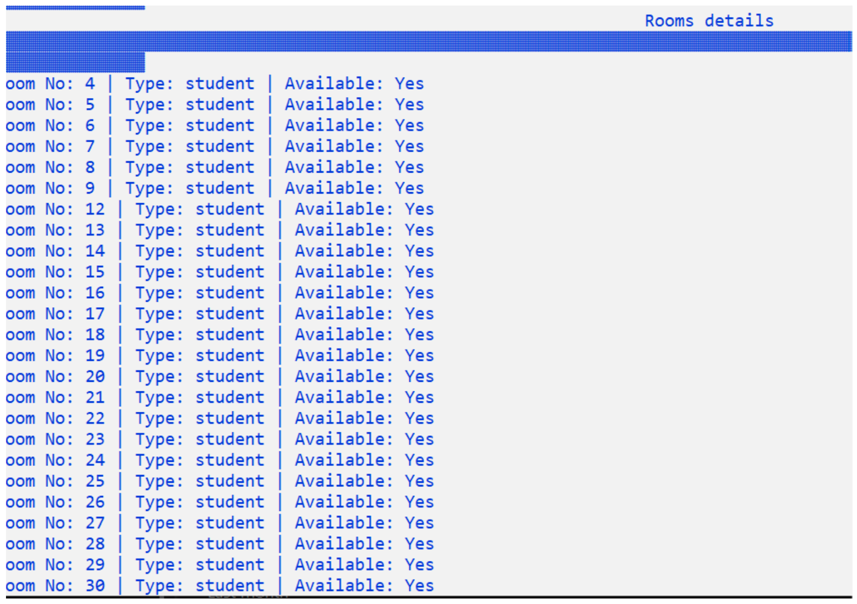
5.2.3 Display Data of Student



```
                                   Load students data

 Name        ID        Contact        Roll No     Department      Year        Room No     Total dues
 SAM         3434      03000230132    09          EE              2022        10          0

Student data loaded successfully.
```

Figure 5: Output

5.2.4 Displaying Available Rooms (Student)

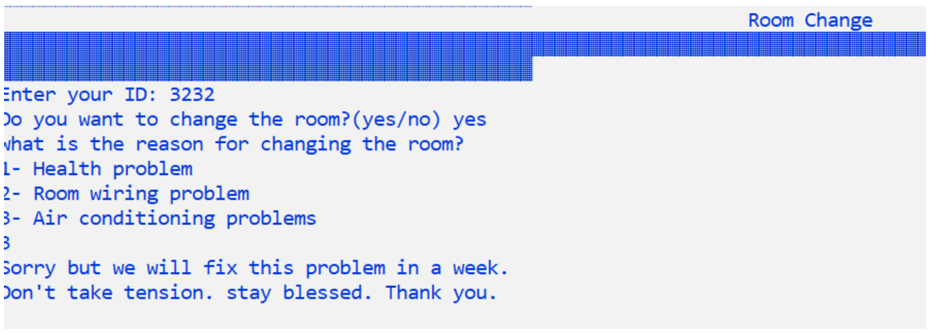


```
                                                 Rooms details

oom No: 4 | Type: student | Available: Yes
oom No: 5 | Type: student | Available: Yes
oom No: 6 | Type: student | Available: Yes
oom No: 7 | Type: student | Available: Yes
oom No: 8 | Type: student | Available: Yes
oom No: 9 | Type: student | Available: Yes
oom No: 12 | Type: student | Available: Yes
oom No: 13 | Type: student | Available: Yes
oom No: 14 | Type: student | Available: Yes
oom No: 15 | Type: student | Available: Yes
oom No: 16 | Type: student | Available: Yes
oom No: 17 | Type: student | Available: Yes
oom No: 18 | Type: student | Available: Yes
oom No: 19 | Type: student | Available: Yes
oom No: 20 | Type: student | Available: Yes
oom No: 21 | Type: student | Available: Yes
oom No: 22 | Type: student | Available: Yes
oom No: 23 | Type: student | Available: Yes
oom No: 24 | Type: student | Available: Yes
oom No: 25 | Type: student | Available: Yes
oom No: 26 | Type: student | Available: Yes
oom No: 27 | Type: student | Available: Yes
oom No: 28 | Type: student | Available: Yes
oom No: 29 | Type: student | Available: Yes
oom No: 30 | Type: student | Available: Yes
```

Figure 6: Output

5.3 Adding a Faculty/Staff

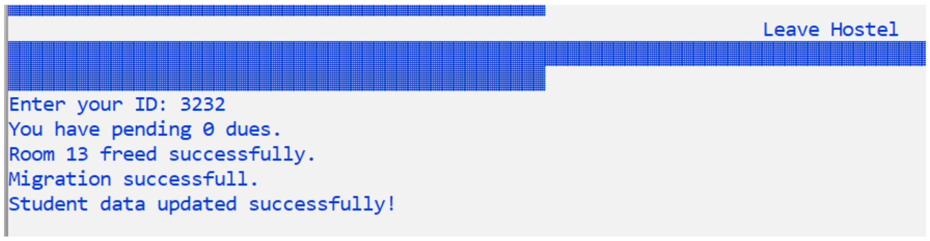
5.3.1 Changing the Room (Faculty/Staff)



```
                                                              Room Change
Enter your ID: 3232
Do you want to change the room?(yes/no) yes
what is the reason for changing the room?
1- Health problem
2- Room wiring problem
3- Air conditioning problems
3
Sorry but we will fix this problem in a week.
Don't take tension. stay blessed. Thank you.
```

Figure 7: Output

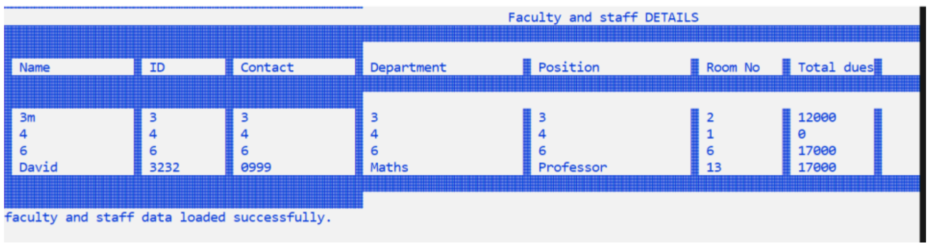
5.3.2 Leaving the Hostel (Faculty/Staff)



```
                                                              Leave Hostel
Enter your ID: 3232
You have pending 0 dues.
Room 13 freed successfully.
Migration successfull.
Student data updated successfully!
```

Figure 8: Output

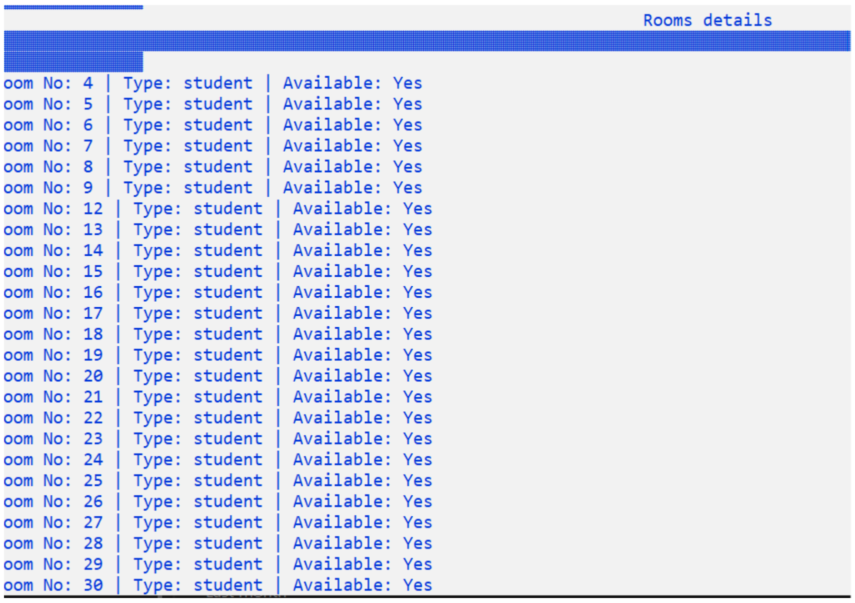
5.3.3 Display Data of Faculty



```
                                          Faculty and staff DETAILS

 Name          ID         Contact       Department          Position            Room No     Total dues

 3m            3          3             3                   3                   2           12000
 4             4          4             4                   4                   1           0
 6             6          6             6                   6                   6           17000
 David         3232       0999          Maths               Professor           13          17000

faculty and staff data loaded successfully.
```

Figure 9: Output

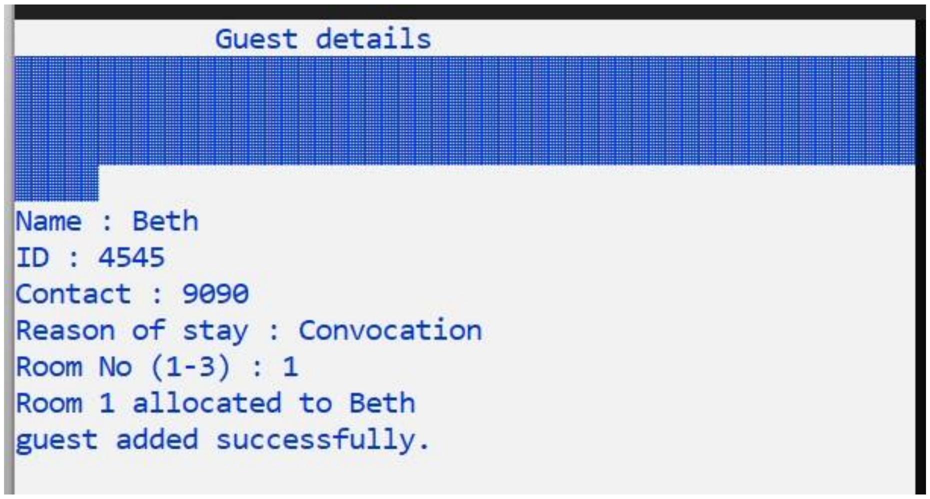
5.3.4 Displaying Available Rooms (Faculty/Staff)



```
                                                        Rooms details
oom No: 4 | Type: student | Available: Yes
oom No: 5 | Type: student | Available: Yes
oom No: 6 | Type: student | Available: Yes
oom No: 7 | Type: student | Available: Yes
oom No: 8 | Type: student | Available: Yes
oom No: 9 | Type: student | Available: Yes
oom No: 12 | Type: student | Available: Yes
oom No: 13 | Type: student | Available: Yes
oom No: 14 | Type: student | Available: Yes
oom No: 15 | Type: student | Available: Yes
oom No: 16 | Type: student | Available: Yes
oom No: 17 | Type: student | Available: Yes
oom No: 18 | Type: student | Available: Yes
oom No: 19 | Type: student | Available: Yes
oom No: 20 | Type: student | Available: Yes
oom No: 21 | Type: student | Available: Yes
oom No: 22 | Type: student | Available: Yes
oom No: 23 | Type: student | Available: Yes
oom No: 24 | Type: student | Available: Yes
oom No: 25 | Type: student | Available: Yes
oom No: 26 | Type: student | Available: Yes
oom No: 27 | Type: student | Available: Yes
oom No: 28 | Type: student | Available: Yes
oom No: 29 | Type: student | Available: Yes
oom No: 30 | Type: student | Available: Yes
```

Figure 10: Output

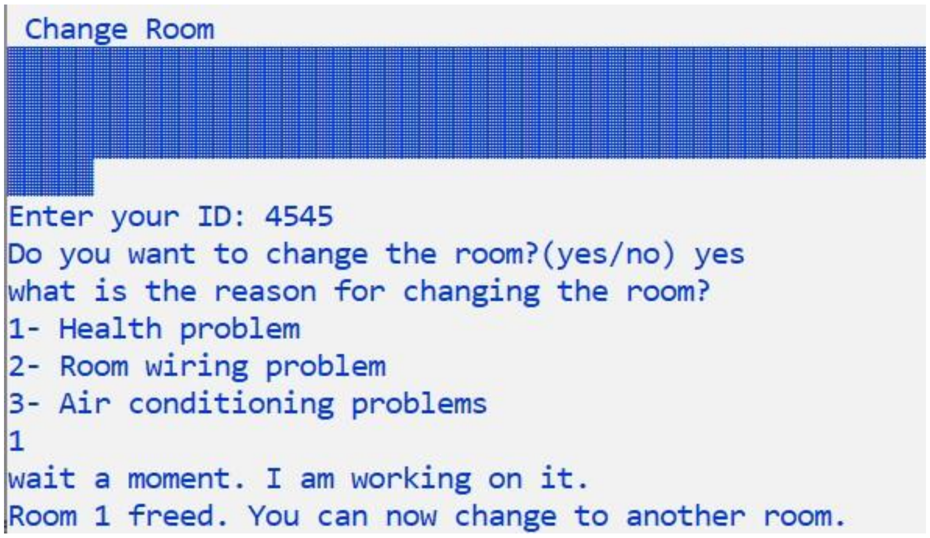
5.4 Adding a Guest



```
             Guest details
Name : Beth
ID : 4545
Contact : 9090
Reason of stay : Convocation
Room No (1-3) : 1
Room 1 allocated to Beth
guest added successfully.
```

Figure 11: Output

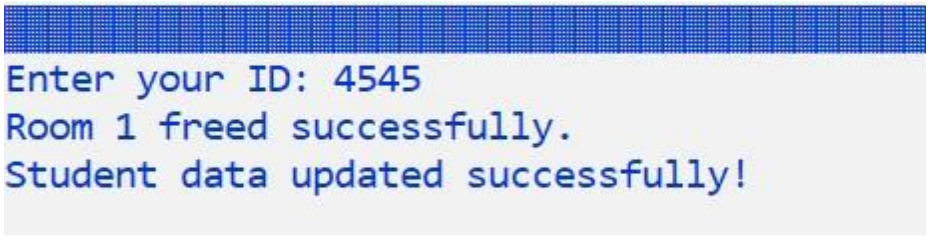
5.4.1 Changing the Room (Guest)



```
Change Room

Enter your ID: 4545
Do you want to change the room?(yes/no) yes
what is the reason for changing the room?
1- Health problem
2- Room wiring problem
3- Air conditioning problems
1
wait a moment. I am working on it.
Room 1 freed. You can now change to another room.
```

Figure 12: Output

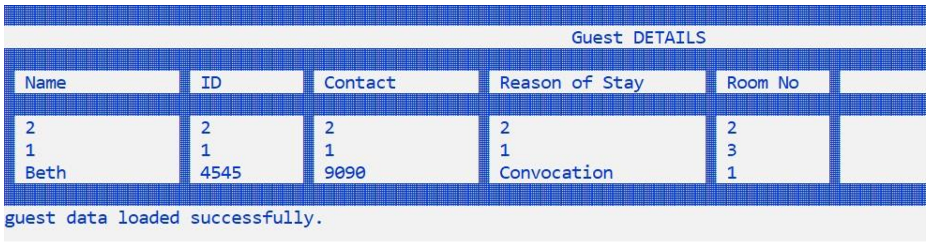
5.4.2 Leaving the Hostel (Guest)



```
Enter your ID: 4545
Room 1 freed successfully.
Student data updated successfully!
```

Figure 13: Output

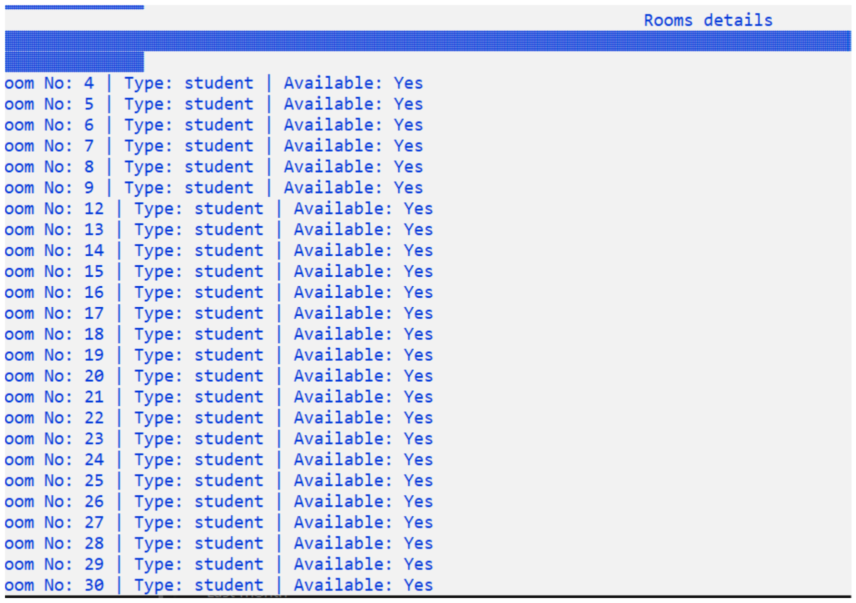
5.4.3 Displaying List of Guests



```
                                   Guest DETAILS
Name             ID          Contact         Reason of Stay        Room No
2                2           2               2                     2
1                1           1               1                     3
Beth             4545        9090            Convocation           1
guest data loaded successfully.
```

Figure 14: Output

5.4.4 Displaying Available Rooms (Guest)



```
                                                        Rooms details
oom No: 4 | Type: student | Available: Yes
oom No: 5 | Type: student | Available: Yes
oom No: 6 | Type: student | Available: Yes
oom No: 7 | Type: student | Available: Yes
oom No: 8 | Type: student | Available: Yes
oom No: 9 | Type: student | Available: Yes
oom No: 12 | Type: student | Available: Yes
oom No: 13 | Type: student | Available: Yes
oom No: 14 | Type: student | Available: Yes
oom No: 15 | Type: student | Available: Yes
oom No: 16 | Type: student | Available: Yes
oom No: 17 | Type: student | Available: Yes
oom No: 18 | Type: student | Available: Yes
oom No: 19 | Type: student | Available: Yes
oom No: 20 | Type: student | Available: Yes
oom No: 21 | Type: student | Available: Yes
oom No: 22 | Type: student | Available: Yes
oom No: 23 | Type: student | Available: Yes
oom No: 24 | Type: student | Available: Yes
oom No: 25 | Type: student | Available: Yes
oom No: 26 | Type: student | Available: Yes
oom No: 27 | Type: student | Available: Yes
oom No: 28 | Type: student | Available: Yes
oom No: 29 | Type: student | Available: Yes
oom No: 30 | Type: student | Available: Yes
```

Figure 15: Output

6 Conclusion

The Namal University Hostel Management System helps make hostel work easier. It is simple to use and works well. In the future, it can be improved by making the interface better, adding reports, and allowing online payments.