



HOSTEL MANAGEMENT SYSTEM

Lecturer: Dr. Noureen Talpur



GROUP MEMBERS

NAME	ID
YAP CLEMENT	22008723
ADIB THAQIFY BIN SUHAIMI	22011594
ZENG KAI MING	24001855
AIMAN NASRULLAH BIN MAS NASRULLAH	22009315



Project Idea

The goal of the Hostel Management System (HMS) is to manage hostel lodgings more efficiently by streamlining the administrative procedures. Hostel administrators may more easily oversee everyday operations with greater accuracy and efficiency thanks to this software system, which automates billing, cleaning service requests, and room distribution.

Objectives:

- To automate the process of room allocation and changes.
- To provide an easy-to-use interface for requesting cleaning services.
- To calculate and manage billing based on room type and additional services.
- To maintain data integrity by saving and loading room details from files.

Introduction & Background

The hostel management system is a comprehensive software solution designed to streamline various aspects of hostel accommodation. The program allows users to choose their type of room village, floor, room and cleaning service. This program also provides transparency, accountability and flexibility for student to select or change room and calculate the accommodation bill.



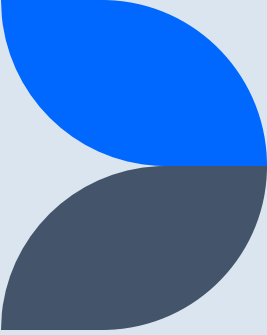
Room Management

Functionality: The system allows users to change their rooms by specifying details such as room type (shared/single), village (V1 to V6), block, floor, unit, and room number.

Implementation: The *changeRoom* function, along with *promptRoomDetails*, collects and updates room details based on user input. The room data is stored in a Room structure that includes attributes like type, village, block, floor, unit, room number, air conditioning status, cleaning service status, and price.

```
101  void changeRoom() {  
102      char type[10];  
103      char village[3];  
104      char block[4];  
105      int floor;  
106      int unit;  
107      int room;  
108  
109      promptRoomDetails(type, village, block, floor, unit, room);  
110      cout << "Room change logic here\n";  
111      cout << "Changed to room: " << type << " in " << village << " " << block  
112          << " on floor " << floor << ", unit " << unit << ", room " << room  
113          << endl;  
114  }  
115
```

Sample output



```
Hostel Management System
1. Change Room
2. Cleaning Service
3. Calculate Bill
4. Save Room Details to File
5. Load Room Details from File
0. Exit
Enter your choice: 1
Enter room type (Shared/Single): shared
Enter village (V1/V2/V3/V4/V5/V6): v2
Enter block (e.g., V1a, V1b): v2b
Enter floor number: 1
Enter unit number: 4
Enter room number: 2
Room change logic here
Changed to room: shared in v2 v2b on floor 1, unit 4, room 2
```

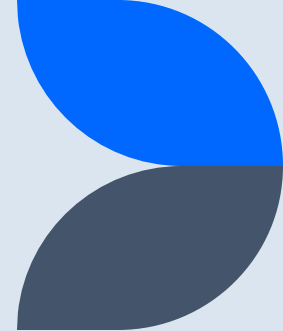
Cleaning Service Management

Functionality: Users can request cleaning services for their rooms. The system tracks the status of these services and includes the associated costs in the final bill.

Implementation: The *selectCleaningService* function updates the cleaning service status for the specified room. This information is integrated into the overall room data managed by the system.

```
116  void selectCleaningService() {  
117      char type[10];  
118      char village[3];  
119      char block[4];  
120      int floor;  
121      int unit;  
122      int room;  
123  
124      promptRoomDetails(type, village, block, floor, unit, room);  
125      cout << "Do you want cleaning service? (1 for Yes, 0 for No): ";  
126      int cleaning;  
127      cin >> cleaning;  
128      bool cleaningService = (cleaning == 1);  
129      cout << "Cleaning service: " << (cleaningService ? "Yes" : "No") << endl;  
130  }
```

Sample output



```
Hostel Management System
1. Change Room
2. Cleaning Service
3. Calculate Bill
4. Save Room Details to File
5. Load Room Details from File
0. Exit
Enter your choice: 2
Enter room type (Shared/Single): 2
Enter village (V1/V2/V3/V4/V5/V6): v6
Enter block (e.g., V1a, V1b): v6b
Enter floor number: 2
Enter unit number: 4
Enter room number: 3
[ Do you want cleaning service? (1 for Yes, 0 for No): 1 ]
[ Cleaning service: Yes ]
```


Billing System

Functionality: The system calculates the total bill for the user, considering factors such as room type, air conditioning, and cleaning services. It provides a detailed breakdown of the charges.

Implementation: The *calculateBill* function computes the bill by iterating through the room data and summing up the costs based on the selected services and room type.

```
132  void calculateBill() {  
133      totalBill = 0;  
134      for (int i = 0; i < roomCount; ++i) {  
135          totalBill += rooms[i].price;  
136      }  
137      cout << "Total bill: RM " << totalBill << endl;  
138  }
```

Sample output

Hostel Management System

1. Change Room
2. Cleaning Service
3. Calculate Bill
4. Save Room Details to File
5. Load Room Details from File
0. Exit

Enter your choice: 3

Enter room type (Shared/Single): shared

Does the room have air conditioning? (1 for Yes, 0 for No): 1

Total bill for the semester: RM 5400

File Handling

Functionality: The system can save room details to a file and load them back, ensuring data persistence and easy recovery.

Implementation: The *saveRoomDetailsToFile* function writes the room details to a specified file, while the *LoadRoomDetailsFromFile* function reads room details from the file.

```
207 void saveRoomDetailsToFile(const char *filename) {
208     ofstream outFile(filename);
209     if (outFile.is_open()) {
210         outFile << roomCount << endl;
211         for (int i = 0; i < roomCount; ++i) {
212             outFile << rooms[i].type << " " << rooms[i].village << " "
213                 << rooms[i].block << " " << rooms[i].floor << " " << rooms[i].unit
214                 << " " << rooms[i].room << " " << rooms[i].airCond << " "
215                 << rooms[i].cleaningService << " " << rooms[i].price << endl;
216         }
217         outFile.close();
218         cout << "Room details saved to " << filename << endl;
219     } else {
220         cout << "Unable to open file for writing." << endl;
221     }
222 }
223
224 void loadRoomDetailsFromFile(const char *filename) {
225     ifstream inFile(filename);
226     if (inFile.is_open()) {
227         inFile >> roomCount;
228         for (int i = 0; i < roomCount; ++i) {
229             inFile >> rooms[i].type >> rooms[i].village >> rooms[i].block >>
230                 rooms[i].floor >> rooms[i].unit >> rooms[i].room >>
231                 rooms[i].airCond >> rooms[i].cleaningService >> rooms[i].price;
232         }
233         inFile.close();
234         cout << "Room details loaded from " << filename << endl;
235     } else {
236         cout << "Unable to open file for reading." << endl;
237     }
238 }
```

Sample output

```
Hostel Management System
1. Change Room
2. Cleaning Service
3. Calculate Bill
4. Save Room Details to File
5. Load Room Details from File
0. Exit
Enter your choice: 4
Room details saved to rooms.txt
```

```
hostel_management.cpp  rooms.txt x
Hostel_management > rooms.txt
1 1000
2 Shared V1 V1a 0 1 1 0 1 998
3 Single V1 V1a 0 1 2 0 1 1100
4 Shared V1 V1a 0 1 3 0 1 998
5 Single V1 V1a 0 1 4 0 1 1100
6 Shared V1 V1a 0 1 5 0 1 998
7 Single V1 V1a 0 1 6 0 1 1100
8 Shared V1 V1a 0 2 1 0 1 998
9 Single V1 V1a 0 2 2 0 1 1100
10 Shared V1 V1a 0 2 3 0 1 998
11 Single V1 V1a 0 2 4 0 1 1100
12 Shared V1 V1a 0 2 5 0 1 998
13 Single V1 V1a 0 2 6 0 1 1100
14 Shared V1 V1a 0 3 1 0 1 998
15 Single V1 V1a 0 3 2 0 1 1100
16 Shared V1 V1a 0 3 3 0 1 998
17 Single V1 V1a 0 3 4 0 1 1100
18 Shared V1 V1a 0 3 5 0 1 998
19 Single V1 V1a 0 3 6 0 1 1100
20 Shared V1 V1a 0 4 1 0 1 998
21 Single V1 V1a 0 4 2 0 1 1100
22 Shared V1 V1a 0 4 3 0 1 998
23 Single V1 V1a 0 4 4 0 1 1100
24 Shared V1 V1a 0 4 5 0 1 998
25 Single V1 V1a 0 4 6 0 1 1100
26 Shared V1 V1a 1 1 1 0 1 998
27 Single V1 V1a 1 1 2 0 1 1100
28 Shared V1 V1a 1 1 3 0 1 998
29 Single V1 V1a 1 1 4 0 1 1100
30 Shared V1 V1a 1 1 5 0 1 998
31 Single V1 V1a 1 1 6 0 1 1100
32 Shared V1 V1a 1 2 1 0 1 998
33 Single V1 V1a 1 2 2 0 1 1100
34 Shared V1 V1a 1 2 3 0 1 998
35 Single V1 V1a 1 2 4 0 1 1100
36 Shared V1 V1a 1 2 5 0 1 998
37 Single V1 V1a 1 2 6 0 1 1100
38 Shared V1 V1a 1 3 1 0 1 998
39 Single V1 V1a 1 3 2 0 1 1100
40 Shared V1 V1a 1 3 3 0 1 998
```

User Interaction & Menu System

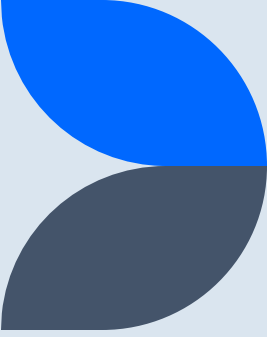
```
74 void showInitialMenu() {  
75     cout << "\nHostel Management System\n";  
76     cout << "1. Change Room\n";  
77     cout << "2. Cleaning Service\n";  
78     cout << "3. Calculate Bill\n";  
79     cout << "4. Save Room Details to File\n";  
80     cout << "5. Load Room Details from File\n";  
81     cout << "0. Exit\n";  
82     cout << "Enter your choice: ";  
83 }
```

Functionality: The system provides a user-friendly menu that guides users through various tasks, such as changing rooms, selecting cleaning services, calculating bills, and saving/loading room details to/from a file.

Implementation: The *showInitialMenu* function displays the menu options, and the *main* function handles user input, directing the flow based on the user's choices.

```
39 int main() {  
40     initializeRooms();  
41  
42     bool exit = false;  
43     while (!exit) {  
44         showInitialMenu();  
45         int choice;  
46         cin >> choice;  
47         switch (choice) {  
48             case 1:  
49                 changeRoom();  
50                 break;  
51             case 2:  
52                 selectCleaningService();  
53                 break;  
54             case 3:  
55                 calculateBill();  
56                 break;  
57             case 4:  
58                 saveRoomDetailsToFile("rooms.txt");  
59                 break;  
60             case 5:  
61                 loadRoomDetailsFromFile("rooms.txt");  
62                 break;  
63             case 0:  
64                 exit = true;  
65                 break;  
66             default:  
67                 cout << "Invalid choice. Try again." << endl;  
68             }  
69         }  
70  
71         return 0;  
72     }
```

Sample output



```
Hostel Management System
1. Change Room
2. Cleaning Service
3. Calculate Bill
4. Save Room Details to File
5. Load Room Details from File
0. Exit
Enter your choice: █
```

Source code

https://github.com/Canvas-Learning-Hub/Hostel_Management/tree/main



Thank you