**Hotel Management System**



Session: 2021 – 2024

**Submitted by:**

Yasir Mahmood 2021-CS-124

**Supervised by:**

Prof.Dr.Awais

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Short Description about project:**

This is actually a business application which mainly consist of the two users (Admin , Customer).Admin can control the cost of per night stays , discounts , data of the customer , keeps the record of the customer and most important he can generate the total bill of the customer. The customer can see the cost of per night stays, type of rooms available, select the number of stays, and see the bill generated by the admin.

**Users of Application:**

Mainly there are two users in this project:

1. Admin
2. Customer

**Functional Requirements (Admin):**

1. As an admin, he can log in.
2. As an admin, he can add cost of rooms for per night stay.
3. As an admin, he can give discount,
4. As an admin, he can data of the customer.
5. As an admin, he can view the data of the customer.
6. As an admin, he can delete the data of the customer.
7. As an admin, he can generate total bill.
8. As an admin, he can exit.

**Functional Requirements (User):**

1. As a user, he can log in.
2. As a user, he can see the cost of per night stay.
3. As a user, he can see the discount on per night stay.
4. As a user, he can select the number of persons.
5. As a user, he can select the type of room.
6. As a user, he can select the number of stays.
7. As a user, he can see the total bill generated by the Admin.
8. As a user, he can exit.

**Data Structures:**

*//.................Data Structure Start*

int user\_count = 0;

const int TOTAL\_USER = 10;

string usernameA[TOTAL\_USER];

string passwordA[TOTAL\_USER];

string roleA[TOTAL\_USER];

const int TOTAL\_CUSTOMER = 10;

int customer\_data\_count = 0;

string nameA[TOTAL\_CUSTOMER];

string mobile\_numberA[TOTAL\_CUSTOMER];

string emailA[TOTAL\_CUSTOMER];

string cnicA[TOTAL\_CUSTOMER];

int staysA[TOTAL\_CUSTOMER];

int customer\_bill\_count = 0;

string type[TOTAL\_CUSTOMER];

int type\_count = 0;

float actual\_bill[TOTAL\_CUSTOMER];

bool customer\_bill = false;

bool room\_price = false;

bool discount\_price = false;

bool customer\_data = false;

bool customer\_stays = false;

bool billing = false;

int discounted, five\_star, four\_star, three\_star, two\_star;

int stays;

*//.................Data Structure End*

**Functions Prototypes:**

string who();

void add\_user(string username, string password, string role);

char menu2(char op2);

char menu1(char op1);

void exit();

void header();

void clear\_screen();

void price\_of\_rooms();

void discount();

void data\_of\_customer();

void view\_data();

float final\_bill(int stays , int cost\_per\_night);

void exit();

**Complete Code:**

*#include* <iostream>

*#include* <fstream>

*#include* <conio.h>

*#include* <stdlib.h>

using namespace std;

*//...Funtions Prototypes...*

string who();

void add\_user(string username, string password, string role);

char menu2(char op2);

char menu1(char op1);

void exit();

void header();

void clear\_screen();

void price\_of\_rooms();

void discount();

void data\_of\_customer();

void view\_data();

float final\_bill(int stays, int cost\_per\_night);

void exit();

*//...Prototypes End...*

*//.................Data Structure Start*

int user\_count = 0;

const int TOTAL\_USER = 10;

string usernameA[TOTAL\_USER];

string passwordA[TOTAL\_USER];

string roleA[TOTAL\_USER];

const int TOTAL\_CUSTOMER = 10;

int customer\_data\_count = 0;

string nameA[TOTAL\_CUSTOMER];

string mobile\_numberA[TOTAL\_CUSTOMER];

string emailA[TOTAL\_CUSTOMER];

string cnicA[TOTAL\_CUSTOMER];

int staysA[TOTAL\_CUSTOMER];

int customer\_bill\_count = 0;

string type[TOTAL\_CUSTOMER];

int type\_count = 0;

float actual\_bill[TOTAL\_CUSTOMER];

bool customer\_bill = false;

bool room\_price = false;

bool discount\_price = false;

bool customer\_data = false;

bool customer\_stays = false;

bool billing = false;

int discounted, five\_star, four\_star, three\_star, two\_star;

int stays;

*//.................Data Structure End*

*//...Main Starts...*

main()

{

    add\_user("admin", "123", "ADMIN");

    add\_user("customer", "123", "CUSTOMER");

*while* (true)

    {

        string find = who();

        clear\_screen();

        char admin\_option = ' ';

*if* (find == "ADMIN")

        {

*while* (true)

            {

                admin\_option = menu1(admin\_option);

*if* (admin\_option == '1')

                {

                    price\_of\_rooms();

                    system("cls");

                }

*else* *if* (admin\_option == '2')

                {

                    discount();

                    system("cls");

                }

*else* *if* (admin\_option == '3')

                {

                    data\_of\_customer();

                    system("cls");

                }

*else* *if* (admin\_option == '4')

                {

*if* (customer\_data == true)

                    {

                        view\_data();

                    }

*else*

                    {

                        cout << "Customer data has not entered yet." << endl;

                    }

                }

*else* *if* (admin\_option == '5')

                {

                    string asking;

                    cout << "Are you sure want to delete the data of the customer(yes or no):";

                    cin >> asking;

*if* (asking == "yes")

                    {

                        cout << "Successfully deleted." << endl;

                    }

*else* *if* (asking == "no")

                    {

                        cout << "Data is not deleteed." << endl;

                    }

*else*

                    {

                        cout << "Invalid option!" << endl;

                        cout << "##### Validation #####" << endl;

                        cout << "Write yes or no." << endl;

                    }

                }

*else* *if* (admin\_option == '6')

                {

*if* (customer\_stays == true && room\_price == true)

                    {

                        float actual\_bill;

                        string type2;

                        cout << "Enter the type of room(fivestar,fourstar,threestar,twostar):";

                        cin >> type2;

*if* (type2 == "fivestar")

                        {

                                actual\_bill = final\_bill(stays, five\_star);

                                cout << "So the bill of the customer is:" << actual\_bill;

                                billing = true;

                        }

*else* *if* (type2 == "fourstar")

                        {

                                actual\_bill = final\_bill(stays, four\_star);

                                cout << "So the bill of the customer is:" << actual\_bill;

                                billing = true;

                        }

*else* *if* (type2 == "threestar")

                        {

                                actual\_bill= final\_bill(stays, three\_star);

                                cout << "So the bill of the customer is:" << actual\_bill;

                                billing = true;

                        }

*else* *if* (type2 == "twostar")

                        {

                                actual\_bill = final\_bill(stays, two\_star);

                                cout << "So the bill of the customer is:" << actual\_bill;

                                billing = true;

                        }

*else*

                        {

                            cout << "Invalid room." << endl;

                            cout << "#### Validation ####" << endl;

                            cout << "Please select the right type of room." << endl;

                        }

                    }

*else*

                    {

                        cout << "Customer has not entered the number of stays yet.";

                        cout << "#### Validation ####" << endl;

                        cout << "First enter the number of stays." << endl;

                    }

                }

*else* *if* (admin\_option == '7')

                {

                    cout << "No need for the employe." << endl;

                }

*else* *if* (admin\_option == '8')

                {

                    exit();

*break*;

                }

*else*

                {

                    cout << "Invalid option." << endl;

                    cout << "#### Validation ####" << endl;

                    cout << "Please try again! You have entered wrong option." << endl;

                }

                clear\_screen();

            }

        }

*else* *if* (find == "CUSTOMER")

        {

            char customer\_option = ' ';

*while* (true)

            {

                customer\_option = menu2(customer\_option);

*if* (customer\_option == '1')

                {

*if* (room\_price == false)

                    {

                        cout << "Prices of the rooms are not entered yet." << endl;

                    }

*else*

                    {

                        fstream file2;

                        string test;

                        file2.open("RoomPrice.txt", ios::in);

*while*(getline(file2, test))

                        { *//read data from file object and put it into string.*

                            cout << test << "\n"; *//print the data of the string*

                        }

                        file2.close();

                    }

                }

*else* *if* (customer\_option == '2')

                {

*if* (discount\_price == true)

                    {

                        fstream file4;

                        file4.open("Discount.txt", ios::in);

                        file4 >> discounted;

                        cout << "Discount on per night stay is:" << discounted;

                    }

*else*

                    {

                        cout << "Discount has not been entered yet." << endl;

                    }

                }

*else* *if* (customer\_option == '3')

                {

                    int persons;

                    cout << "Enter the number of persons:";

                    cin >> persons;

                    system("cls");

                }

*else* *if* (customer\_option == '4')

                {

*if* (type\_count < 10)

                    {

                        cout << "You can select the type of room:(five star,four star,three star,two star):";

                        cin >> type[type\_count];

                        type\_count = type\_count + 1;

                    }

*else*

                    {

                        cout << "No more rooms." << endl;

                    }

                }

*else* *if* (customer\_option == '5')

                {

*if* (customer\_bill\_count < 10)

                    {

                        cout << "Enter the number of stays:";

                        cin >> stays;

                        customer\_stays = true;

                        customer\_bill\_count = customer\_bill\_count + 1;

                    }

*else*

                    {

                        cout << "No more space!" << endl;

                    }

                }

*else* *if* (customer\_option == '6')

                {

*if* (billing == true)

                    {

                        cout << "Your Total bill is:" << actual\_bill;

                    }

*else*

                    {

                        cout << "Bill has not been generated by the admin." << endl;

                    }

                }

*else* *if* (customer\_option == '7')

                {

                    exit();

*break*;

                }

*else*

                {

                    cout << "Please try again! You have entered wrong option." << endl;

                }

                clear\_screen();

            }

            clear\_screen();

        }

*else*

        {

            cout << "Wrong username or password.";

        }

    }

}

*//...Main End...*

*//...Funtions Implementation...*

float final\_bill(int stays, int cost\_per\_night)

{

    float total\_bill , bill , total\_discount;

    bill = stays\*cost\_per\_night;

    total\_bill = (bill \* discounted)/100;

    total\_discount = bill - total\_bill;

*return* total\_bill;

}

void view\_data()

{

    fstream file6;

    string test2;

    file6.open("Customer\_Data.txt", ios::in);

*while*(getline(file6,test2))

    {

        cout<<test2;

    }

        cout<<endl;

}

void data\_of\_customer()

{

    fstream file5;

    file5.open("Customer\_Data.txt", ios::app);

*if* (customer\_data\_count < 10)

    {

        cout << "Enter the name of the "<< customer\_data\_count + 1 << " customer:";

        cin >> nameA[customer\_data\_count];

        cout << "Enter the mobile number of the "<<customer\_data\_count + 1<<" customer:";

        cin >> mobile\_numberA[customer\_data\_count];

        cout << "Enter the email of the "<<customer\_data\_count + 1<<" customer:";

        cin >> emailA[customer\_data\_count];

        cout << "Enter the CNIC of the "<<customer\_data\_count + 1<<" customer:";

        cin >> cnicA[customer\_data\_count];

        file5 << nameA[customer\_data\_count] << "\t" << mobile\_numberA[customer\_data\_count] << "\t" << emailA[customer\_data\_count] << "\t" << cnicA[customer\_data\_count] << endl;

        customer\_data\_count = customer\_data\_count + 1;

        cout<<endl;

    }

    customer\_data = true;

}

void discount()

{

    fstream file3;

    file3.open("Discount.txt", ios::out);

    cout << "Enter your discount percentage for per night star:";

    cin >> discounted;

    file3 << discounted;

    file3.close();

    room\_price = true;

}

void price\_of\_rooms()

{

    fstream file1;

    file1.open("RoomPrice.txt", ios::out);

    cout << "Enter the price of five star room:";

    cin >> five\_star;

    cout << "Enter the price of four star room:";

    cin >> four\_star;

    cout << "Enter the price of three star room:";

    cin >> three\_star;

    cout << "Enter the price of the two star room:";

    cin >> two\_star;

    file1 << "Fivestar"<< "\t"<<"Fourstar"<< "\t"<< "Threestar"<< "\t"<< "Twostar" << endl;

    file1 << five\_star << "\t"<< four\_star << "\t"<< three\_star << "\t" << two\_star;

    file1.close();

    discount\_price = true;

}

void header()

{

  cout<<"   \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

  cout<<"   \*                  HOTEL MANAGEMENT SYSTEM                             \*"<<endl;

  cout<<"   \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

}

void exit()

{

    cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

    cout << "\*\*\*\*\*    Thanks for using Hotel Management System            \*\*\*\*\*" << endl;

    cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

string who()

{

    string password, username;

    cout << "Enter the username (admin or customer):";

    cin >> username;

    cout << "Enter the password (123):";

    cin >> password;

*for* (int a = 0; a < TOTAL\_USER; a++)

    {

*if* (usernameA[a] == username && passwordA[a] == password)

        {

*return* roleA[a];

        }

    }

*return* "Wrong.";

}

void add\_user(string username, string password, string role)

{

*if* (user\_count < TOTAL\_USER)

    {

        usernameA[user\_count] = username;

        passwordA[user\_count] = password;

        roleA[user\_count] = role;

        user\_count = user\_count + 1;

    }

*else*

    {

        cout << "No more space.";

    }

}

char menu1(char op1)

{

    header();

    cout<<"WELCOME ADMIN."<<endl;

    cout << "--------------------------------------------------------------------------------" << endl;

    cout << "MAIN MENU:" << endl;

    cout << "1) You can add the cost of rooms for per night stay." << endl;

    cout << "2) You can give discount." << endl;

    cout << "3) You can add the data of the customer." << endl;

    cout << "4) You can see the data of the customer." << endl;

    cout << "5) You can delete the data of the customer." << endl;

    cout << "6) You can generate the total bill of the customer." << endl;

    cout << "7) You can hire employee for hotel." << endl;

    cout << "8) Exit." << endl;

    cout << "--------------------------------------------------------------------------------" <<endl;

    cout << "Enter your option:" << endl;

    cin >> op1;

*return* op1;

}

char menu2(char op2)

{

    header();

    cout<<"WELCOME CUSTOMER."<<endl;

    cout << "--------------------------------------------------------------------------------" << endl;

    cout << "MAIN MENU:" << endl;

    cout << "1) You can see the cost of per night stay." << endl;

    cout << "2) You can see the discount on rooms." << endl;

    cout << "3) You can select the number of persons." << endl;

    cout << "4) You can select the type of Room." << endl;

    cout << "5) You can select the number of stays." << endl;

    cout << "6) You can see the total bill." << endl;

    cout << "7) Exit..." << endl;

    cout << "--------------------------------------------------------------------------------" <<endl;

    cout << "Enter your option:" << endl;

    cin >> op2;

*return* op2;

}

void clear\_screen()

{

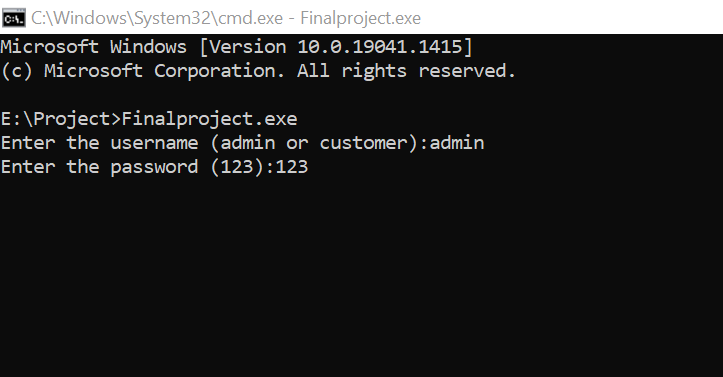
    cout << "Press any key to continue..." << endl;

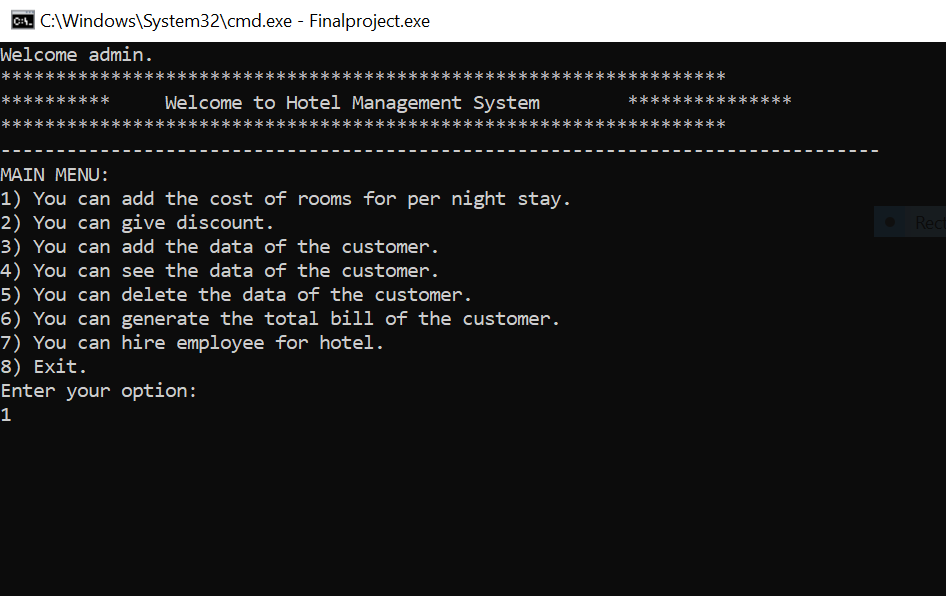
    getch();

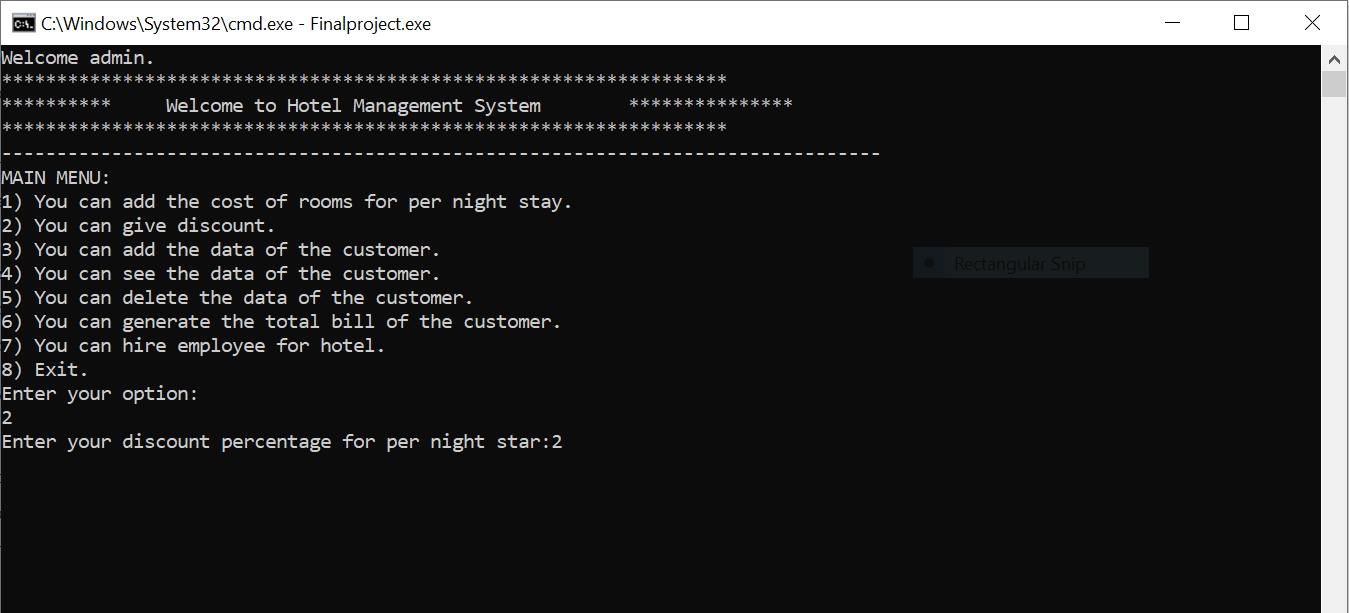
    system("CLS");

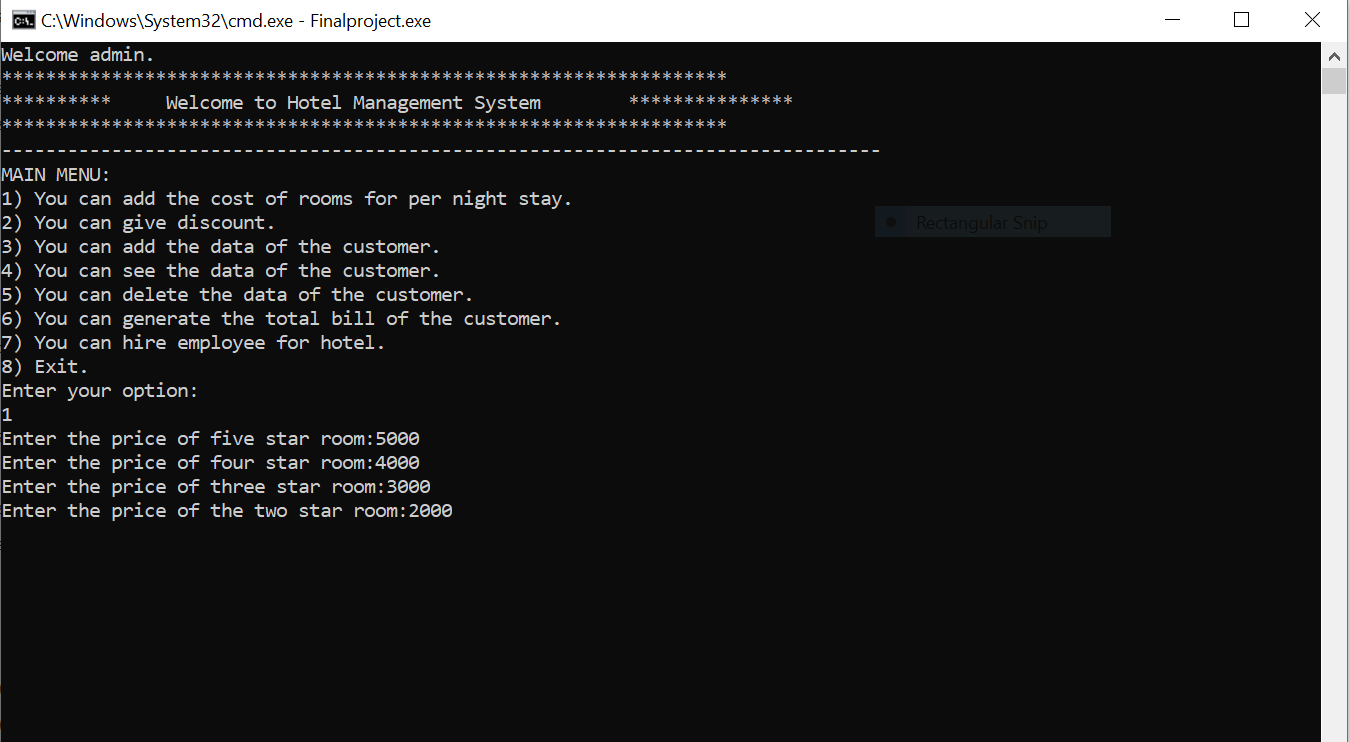
}

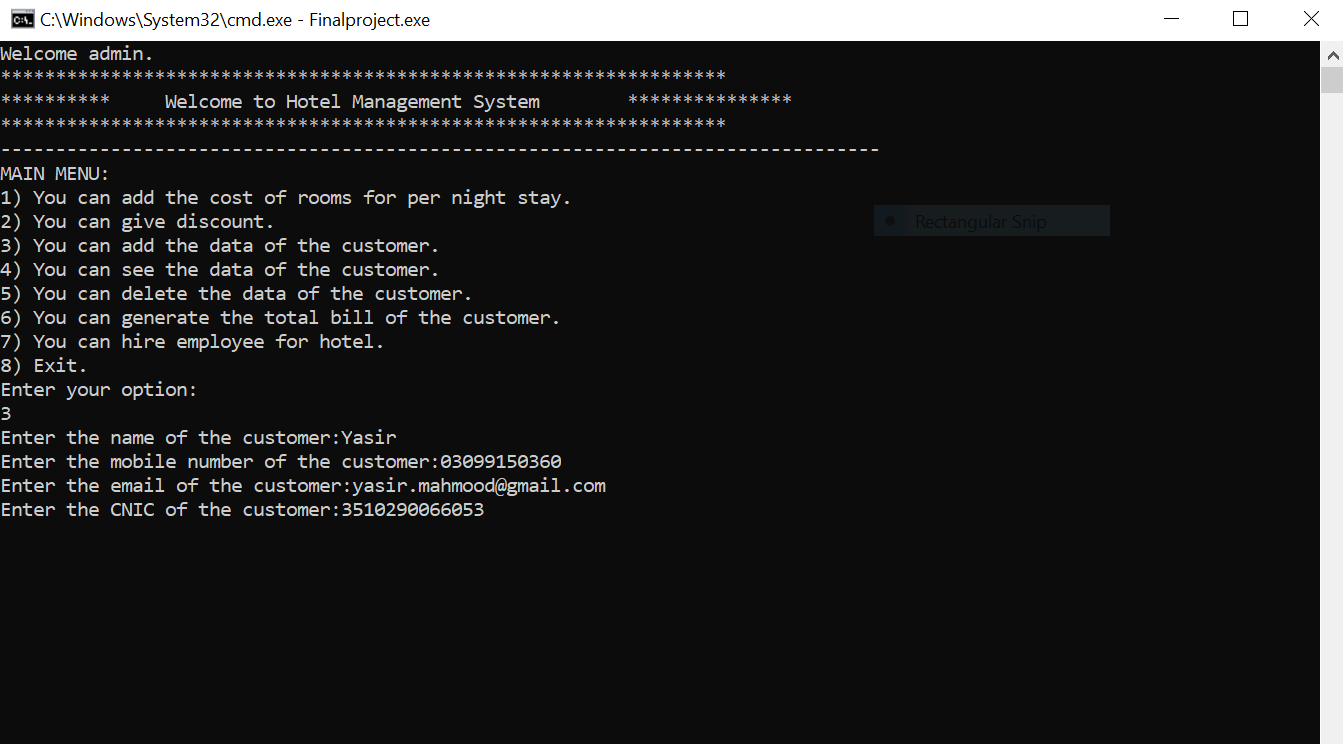
**Wireframes and Test Cases:**

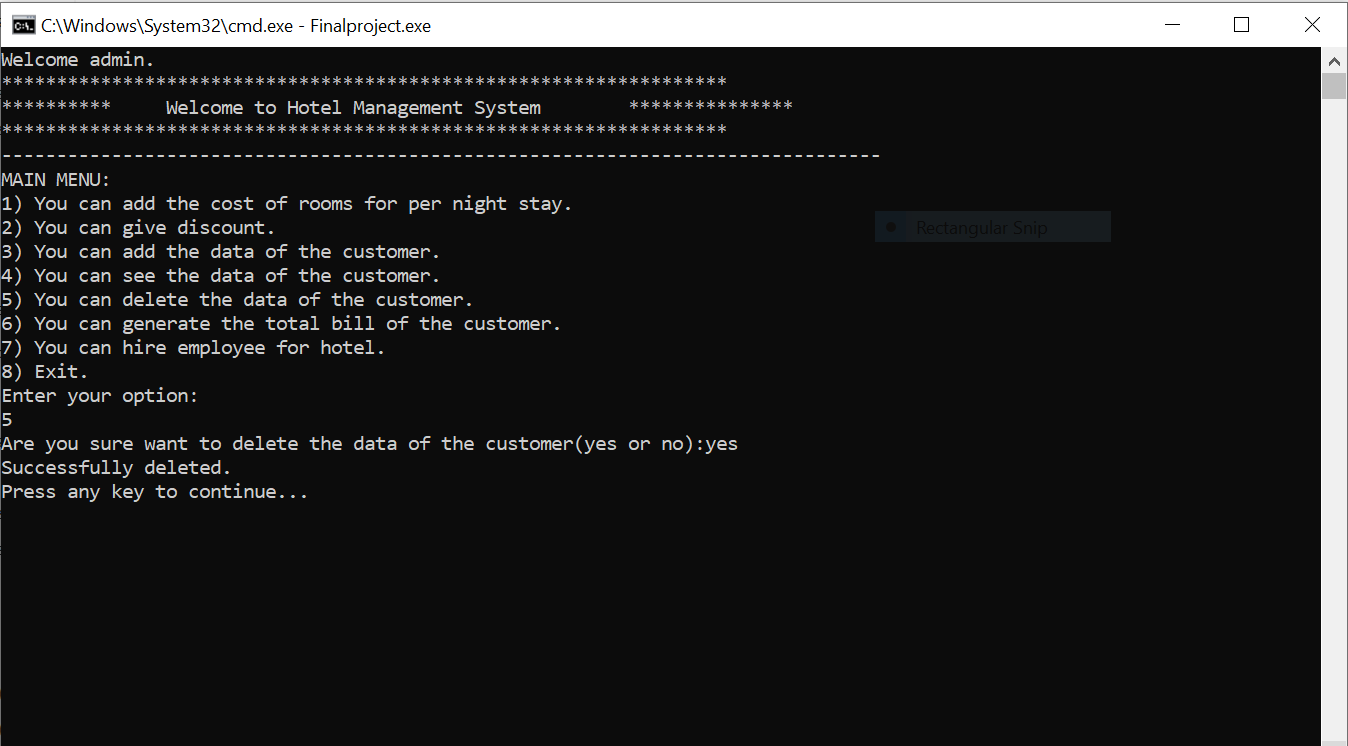
****

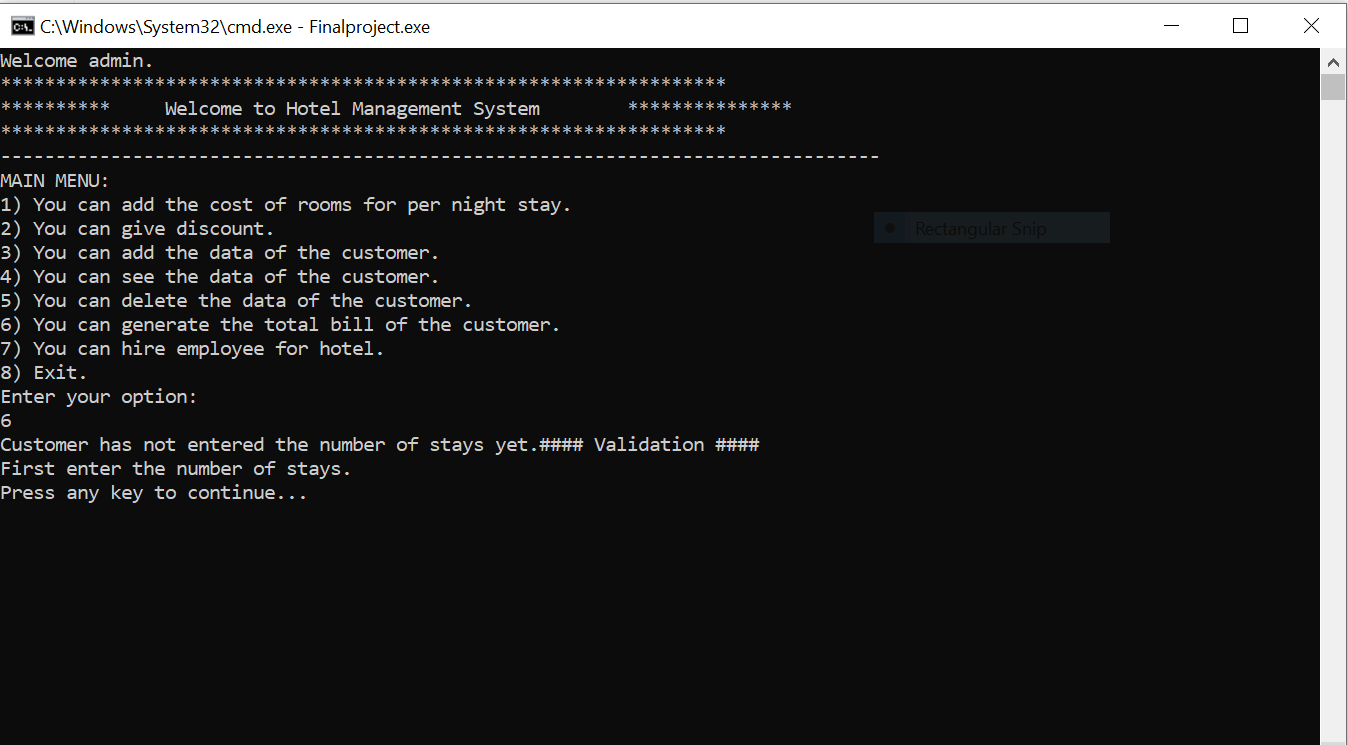
****

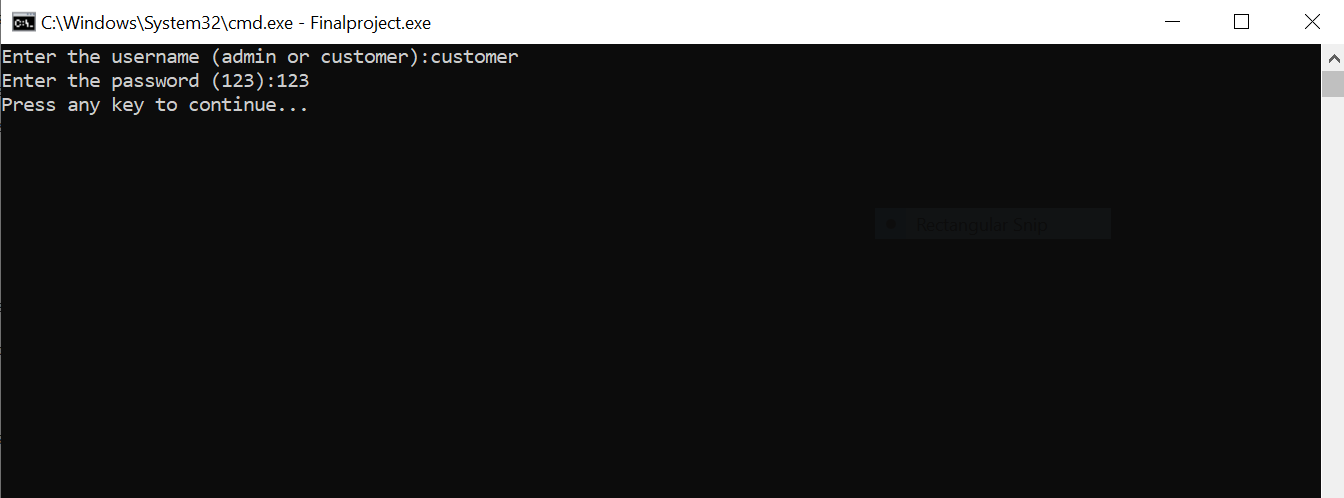
****

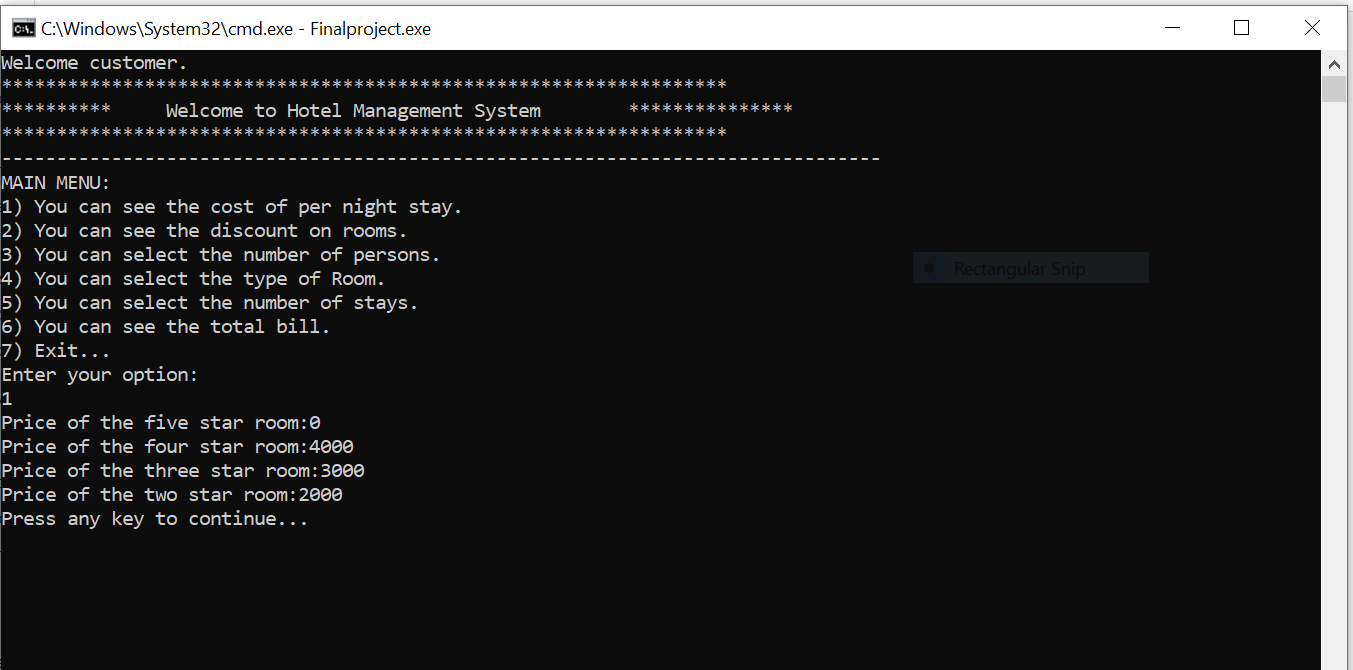
****

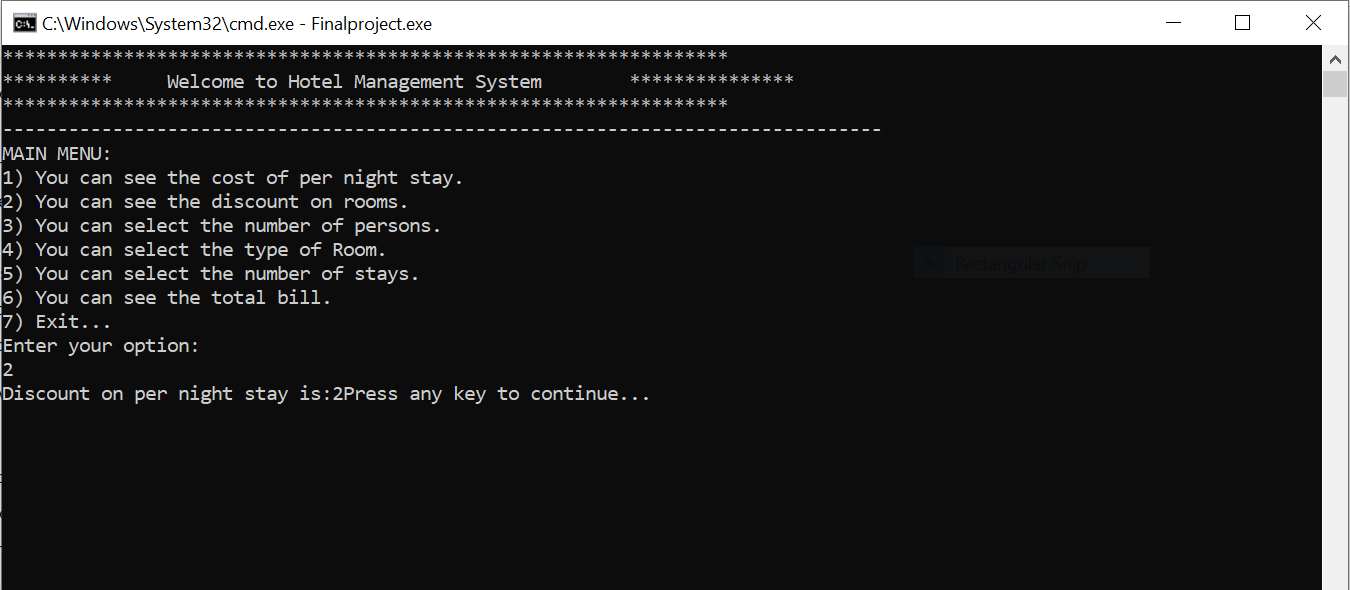
****

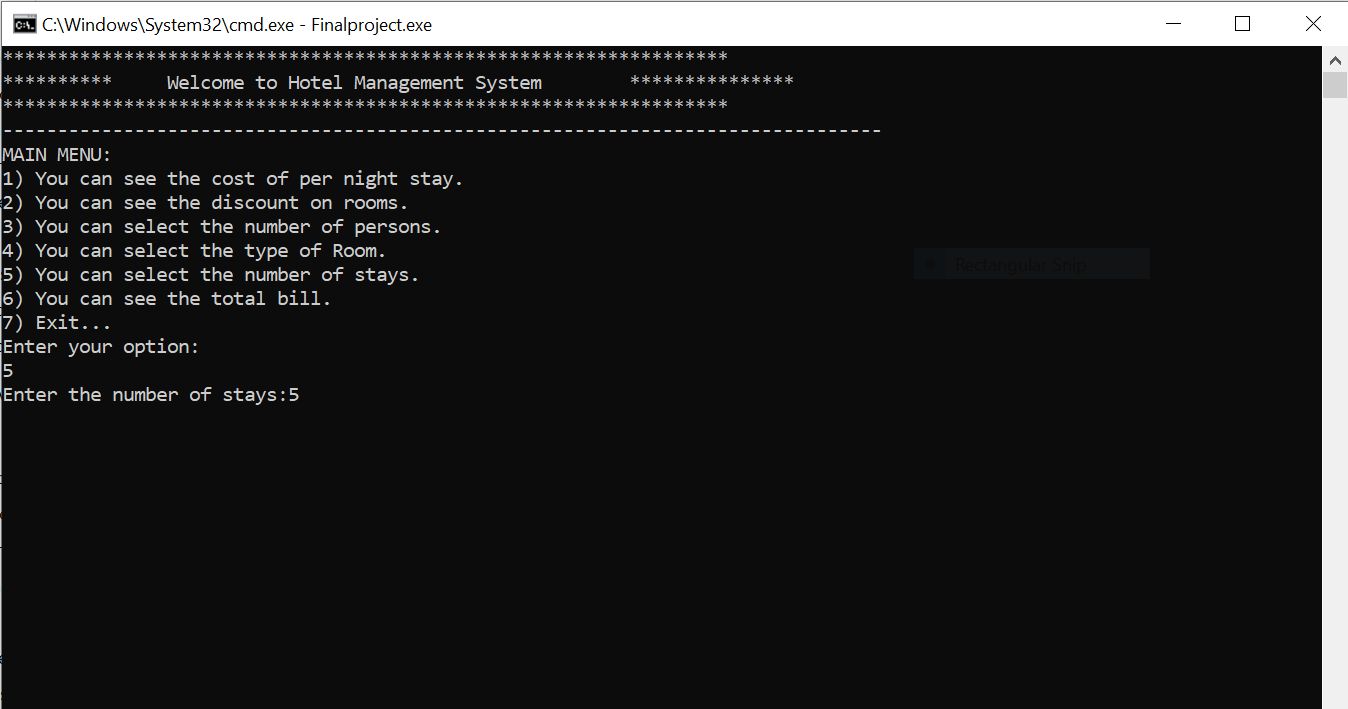
****

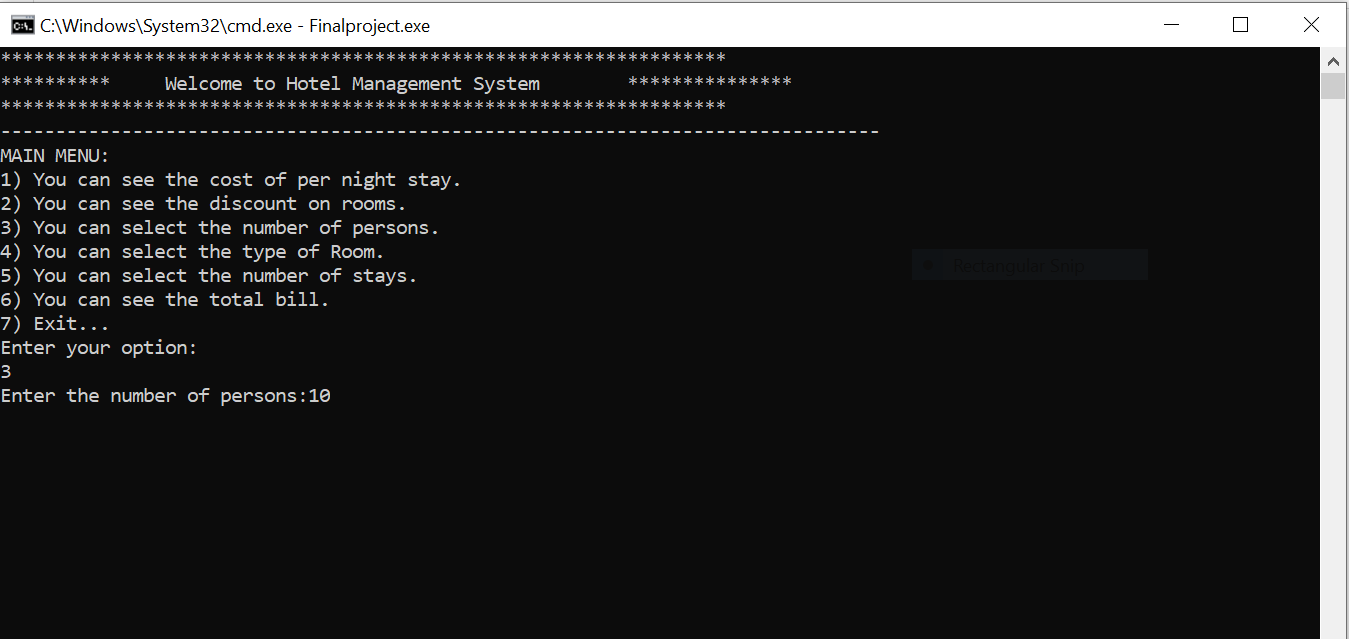
****

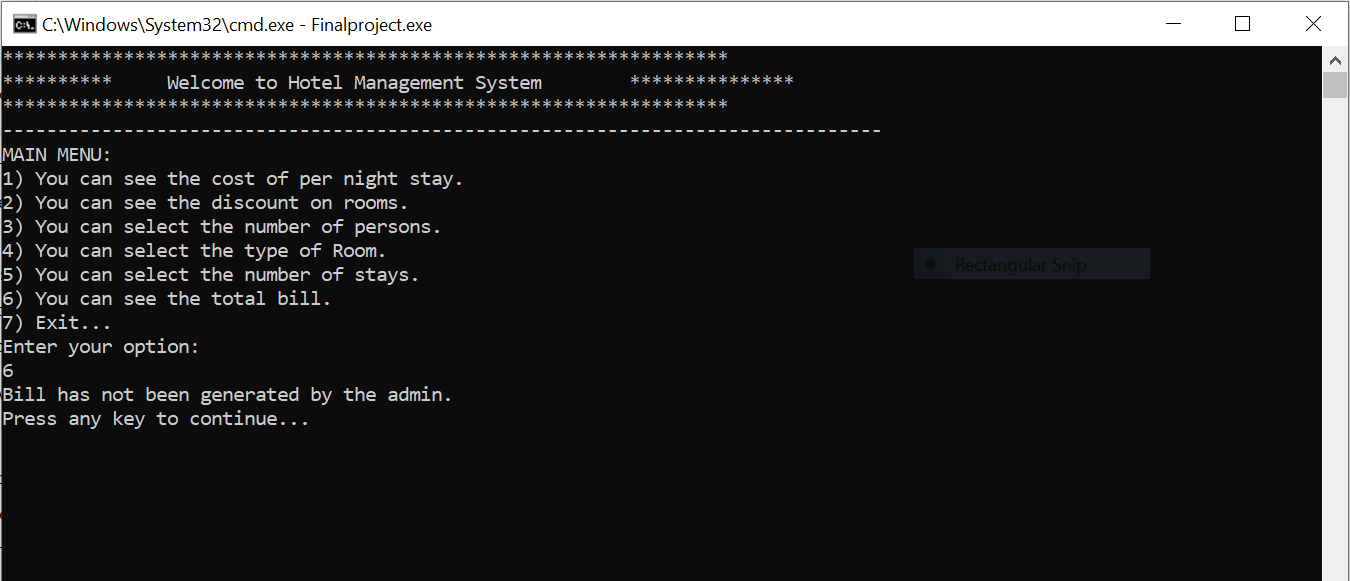
****

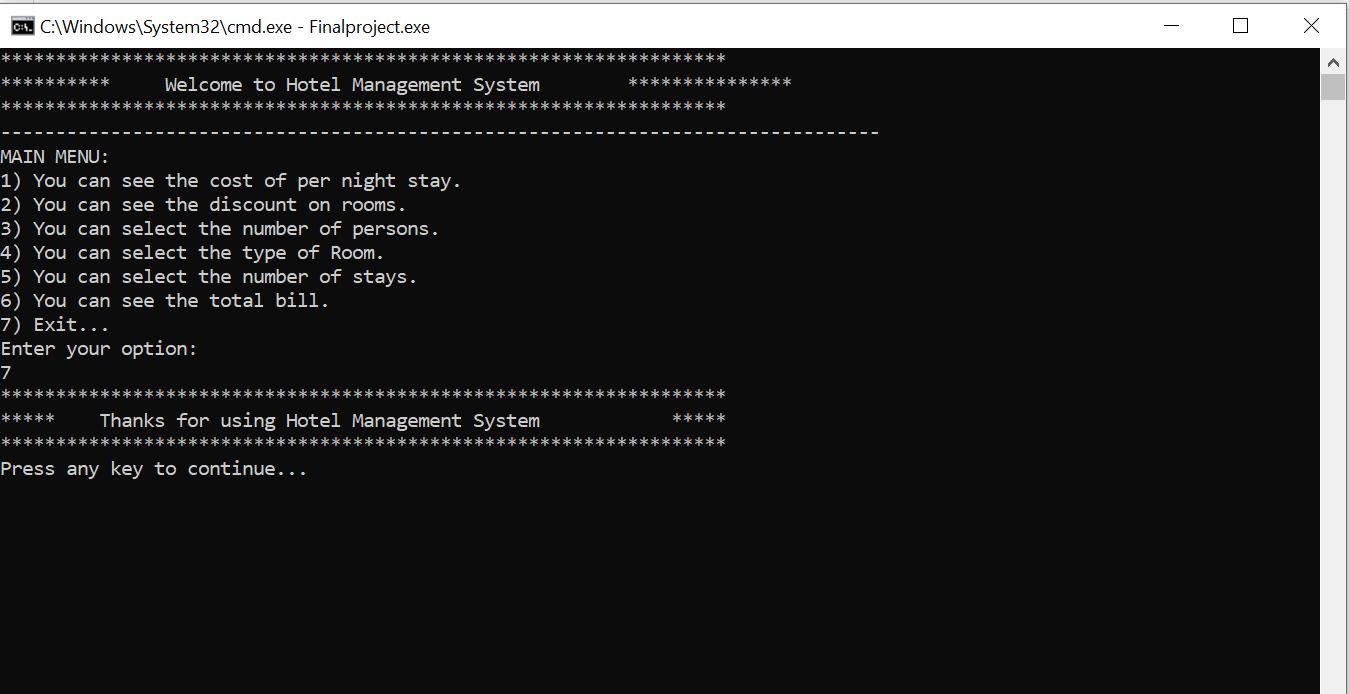
****

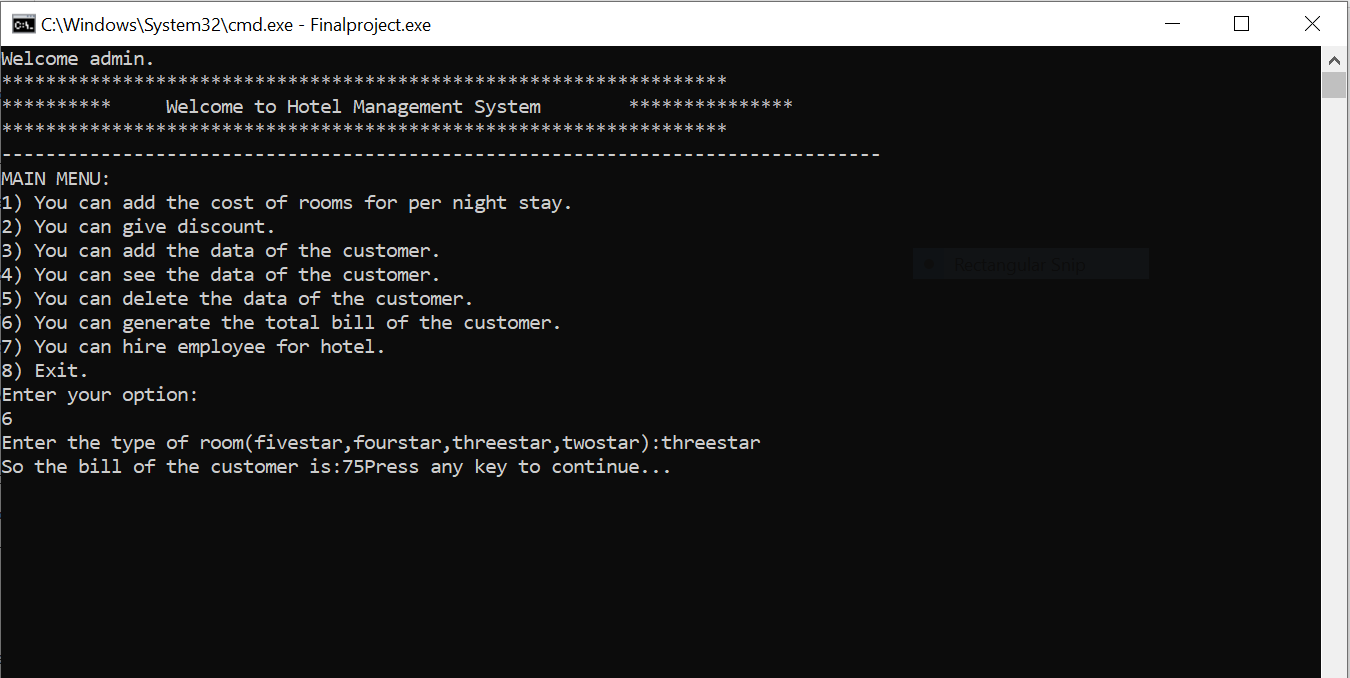
****

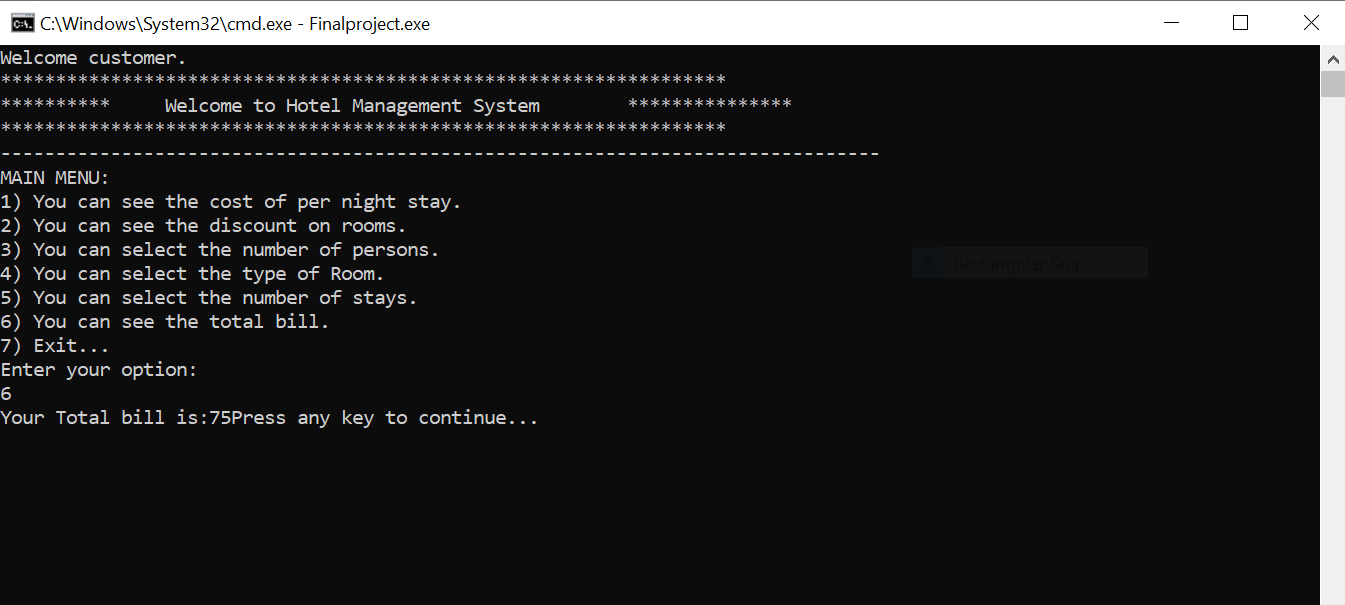
****

****

****

****

****

****

**Functions Working Flow:**

main

menu2(customer\_option)

menu1(admin\_option)

who()

add\_user()

exit()

final\_bill(stays,type)

data\_of\_customer()

discount()

price\_of\_rooms()

**Student Reg. No. :**  2021-CS-124  **Student Name:** YasirMahmood

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Checked by**: | **A-Extensive Evidence** | **B- Convincing Evidence** | **C-Limited Evidence** | **D-No Evidence** |
| Documentation  Formatting **Grade:** | All the documentation meets all the criteria. |  |  |  |
| Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. | Documentation is not Available |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation Contents **Grade:** | Documentation includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria:** **Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |
| Project  Complexity  **Grade:** | Project has at least 2 user’s types and each user has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive evidence | Project complexity meet 50%  criteria given in extensive evidence | Project complexity meet less than 50% criteria given in extensive evidence |
| Code Style **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code style, |
| **Code Style Criteria:**  Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure  (Arrays) **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Sorting Features **Grade:** | Sort working 100% and generating useful report | Sorting Feature is working but sorted data is not useful for project. | Sorting feature is partial implemented | Project do not contain sorting |
| Modularity **Grade:** | Meet all Modularity criteria | Meet all Modularity criteria  but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality Added-At least Two Unit Tests are defined. | | | | |
| Validations **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| Recommendatio n Feature | Proper meaning full recommendation is present into system | Partial Recommendation is implemented | Implemented but not meaning full. | Not implemented |
| Presentation and  Demo  **Grade:** | Presentation and Demo was 100% working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student Understanding with the Code.  **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not  know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |

**Student Reg. No. :**  2021-CS-124  **Student Name:** YasirMahmood

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Checked by**: | **A-Extensive Evidence** | **B- Convincing Evidence** | **C-Limited Evidence** | **D-No Evidence** |
| Documentation  Formatting **Grade:** | All the documentation meets all the criteria. |  |  |  |
| Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. | Documentation is not Available |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation Contents **Grade:** | Documentation includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria:** **Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |
| Project  Complexity  **Grade:** | Project has at least 2 user’s types and each user has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive evidence | Project complexity meet 50%  criteria given in extensive evidence | Project complexity meet less than 50% criteria given in extensive evidence |
| Code Style **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code style, |
| **Code Style Criteria:**  Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure  (Arrays) **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Sorting Features **Grade:** | Sort working 100% and generating useful report | Sorting Feature is working but sorted data is not useful for project. | Sorting feature is partial implemented | Project do not contain sorting |
| Modularity **Grade:** | Meet all Modularity criteria | Meet all Modularity criteria  but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality Added-At least Two Unit Tests are defined. | | | | |
| Validations **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| Recommendatio n Feature | Proper meaning full recommendation is present into system | Partial Recommendation is implemented | Implemented but not meaning full. | Not implemented |
| Presentation and  Demo  **Grade:** | Presentation and Demo was 100% working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student Understanding with the Code.  **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not  know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |