**PROJECT REPORT**

***Submitted by***

**Shrey Sharma [RA2211051010010]**

**&**

**Ashish Arya [RA2211051010041]**

***Under the Guidance of***

**Dr.Madhumita. K**

**ASSISTANT PROFESSOR**

1. **TECH. DEPARTMENT**

**Associate Professor, Department of Computing Technologies**

***In partial satisfaction of the requirements for the degree of***

## **BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE ENGINEERING**

**with specialization in GAMING TECHNOLOGY**

## Logo, company name Description automatically generated

**SCHOOL OF COMPUTING**

# **COLLEGE OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR - 603203**

**MAY 2023**

1

Logo, company name

Description automatically generated

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

**BONAFIDE CERTIFICATE**

Certified that this Project Report titled **“Online Room Booking System”** is the bonafide work done by Shrey Sharma [RA2211051010010] & Ashish Arya [RA2211051010041] who completed the project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other work.

| **SIGNATURE**  **Dr.Madhumita. K**  **OODP – Course Faculty**  Associate Professor  Department of Computing Technologies  SRMIST | **SIGNATURE**  **Dr.Lakshmi**  Professor & Head  Department of Computing Technologies  School of Computing  SRMIST |
| --- | --- |

2

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **CONTENTS** | **PAGE NO** |
| 1. | Abstract |  |
| 2. | Modules of Project |  |
| 3. | Diagrams |  |
|  | 1. Use case Diagram |  |
|  | 1. Class Diagram |  |
|  | 1. Sequence Diagram |  |
|  | 1. Collaboration Diagram |  |
|  | 1. State Chart Diagram |  |
|  | 1. Activity Diagram |  |
|  | 1. Package Diagram |  |
|  | 1. Component Diagram |  |
|  | 1. Deployment Diagram |  |
| 4. | Code/Output Screenshots |  |
| 5. | Conclusion and Results |  |
| 6. | References |  |

3

**ABSTRACT**

The Room Booking project is an excellent software tool for the related industries that can be used in hostels, resorts, lodgings, motels, lodges, hostels.

The aim of this Online Room Booking System project is to build a system that will be able to automate many operations in a hotel. Modern hotels aim to create a user-friendly atmosphere with the availability of concierges who remember frequent visitors and make it possible to call and make reservations. While such hotels are extremely expensive, such a service can also be provided in a cost – effective manner with the use of computers.

This software is designed for luxury full-service lodge, beds, breakfast and

motels. It emphasizes the best stage of services for character visitors through

our full capabilities that integrate rooms, restaurants and retail stores with

records contained in the book reserves for the fashion book at the bottom of the

office. The process begins when a customer books a room. The booking can be placed

online, once the room is booked, the visitor only needs to turn up and present

identification.

At check out, the user can enter a review which will be uploaded to the

websites, the system will collect user reviews from other sites and from past

guests, making them available to future visitors. The Online Room Booking System will make it easy for higher level management to easily review the operation of the hotel.

Online Room Booking System is much more efficient in terms of human and monetary resources. The number of staff to ensure successful implementation is reduced. Lead time of service is reduced and it is easy to predict room availabilities. Electronic documentation of hotel operations help management in keeping the hotel in check.

4

**MODULE DESCRIPTION**

***<iostream>:*** iostream stands for standard input-output stream. This header file contains definitions of objects like cin, cout, cerr, etc.You use operators or iostream member functions to insert data into a stream (output) or extract data from a stream (input), and to control the format of data that you insert or extract.

***<stlib.h>:*** It is a header file to access a set of C++ template classes to provide common programming data structures and functions such as lists, stacks, arrays, etc. It is a library of container classes, algorithms, and iterators. It is a generalized library and so, its components are parameterized. Working knowledge of template classes is a prerequisite for working with STL.

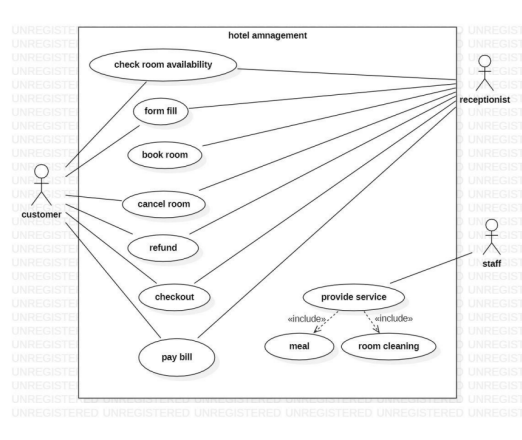
***<fstream>:*** <fstream> library provides functions for files, and we should simply add #include <fstream> directives at the start of our program. To open a file, a file stream object should first be created. This is either a stream object for writing, or an if stream object for reading.

***<direct.h>:*** <direct.h> is a [C](https://en.wikipedia.org/wiki/C_(programming_language)" \o "C (programming language))/[C++](https://en.wikipedia.org/wiki/C++" \o "C++) [header file](https://en.wikipedia.org/wiki/Header_file" \o "Header file) provided by Microsoft Windows, which contains functions for manipulating file system [directories](https://en.wikipedia.org/wiki/Directory_(computing)" \o "Directory (computing)). Some [POSIX](https://en.wikipedia.org/wiki/POSIX" \o "POSIX) functions that do similar things are in [unistd.h](https://en.wikipedia.org/wiki/Unistd.h" \o "Unistd.h).

***<stdio.h>:*** The <stdio.h> header file declares functions that deal with standard input and output. One of these functions, fdopen(), is supported only in a POSIX program.

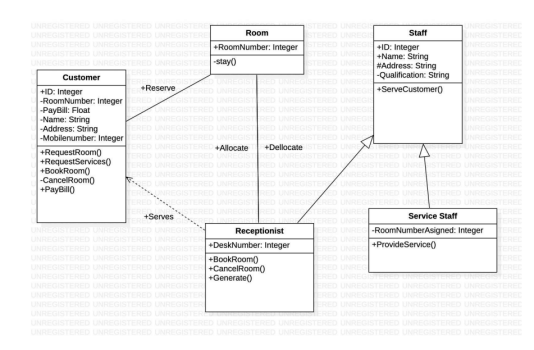
5

**CASE DIAGRAM**

****A case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application.

6

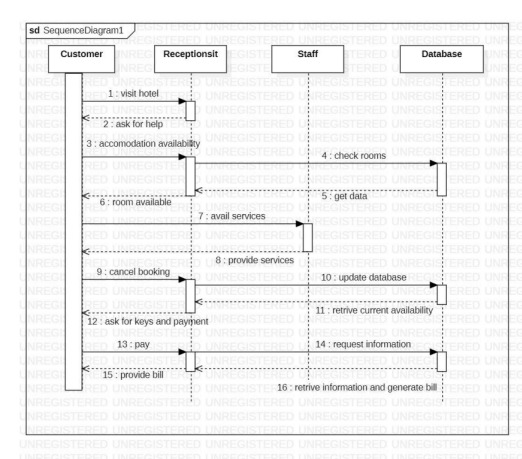
**CLASS DIAGRAM**

****The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code.

It shows the attributes, classes, functions, and relationships to give an overview of the software system. It includes class names, attributes, and functions in a separate compartment that helps in software development.

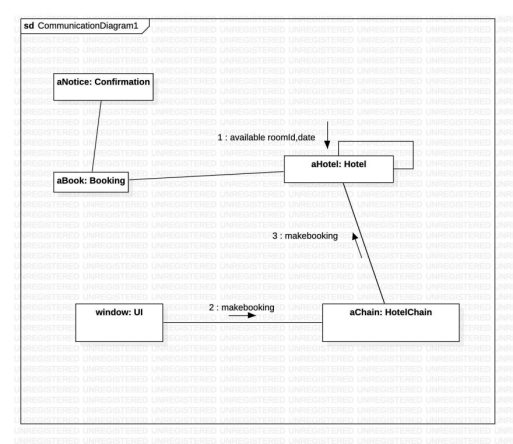
7

**SEQUENCE DIAGRAM**

****The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that  these lifelines took part at the same time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates iterations as well as branching.

8

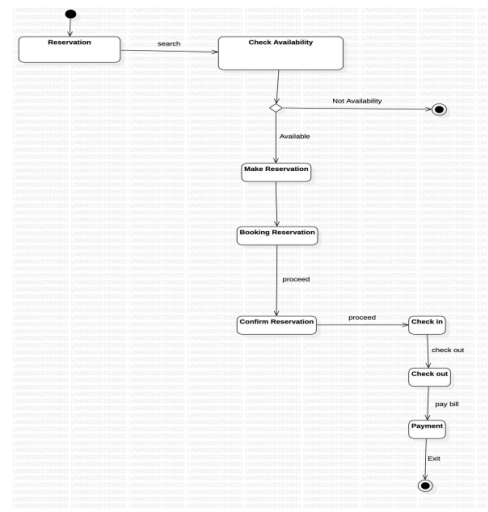
**COLLABORATION DIAGRAM**

****

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

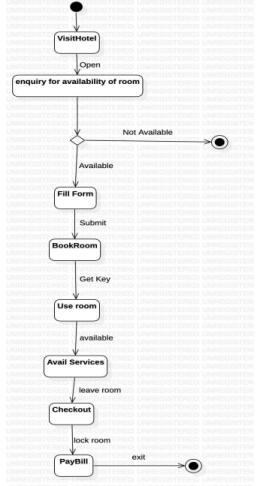
9

**STATE CHART DIAGRAM**

****A Statechart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

10

**ACTIVITY DIAGRAM**

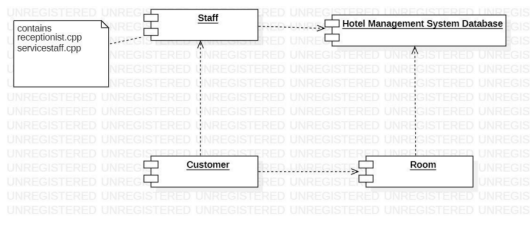


Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all types of flow control by using different elements such as fork, join, etc

11

**COMPONENT DIAGRAM**

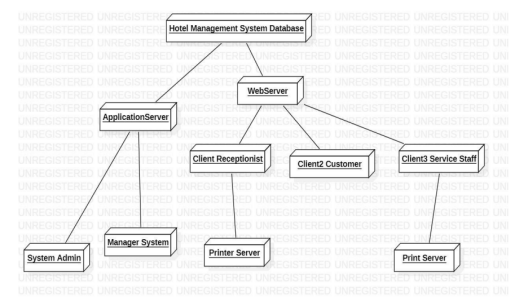
****

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems. it describes the components used to make those functionalities

12

12

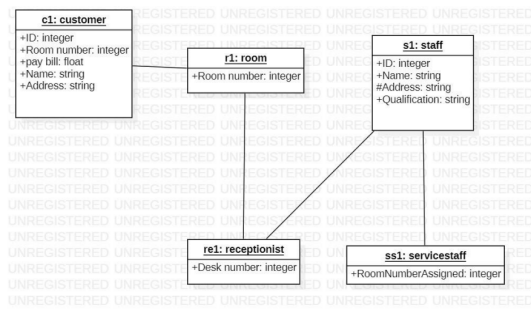
**DEPLOYMENT DIAGRAM**

****

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system.It involves the nodes and their relationships

13

**OBJECT DIAGRAM**

****

Object diagrams represent an instance of a class diagram. The basic concepts are similar to class diagrams and object diagrams. Object diagrams also represent the static view of a system, but this static view is a snapshot of the system at a particular moment.

14

**CODE**

#include<iostream>

#include<fstream>

#include<stdio.h>

#include<direct.h>

#include<stdlib.h>

using namespace std;

class hotel//main class

{

int room\_no;

char name[30];

char address[50];

char phone[10];

public:

void main\_menu(); //to display the main menu

void add(); //to book a room

void display(); //to display the customer record

void rooms(); //to display allotted rooms

void edit(); //to edit the customer record

int check(int); //to check room status

void modify(int); //to modify the record

void delete\_rec(int); //to delete the record

};

void hotel::main\_menu()

{

int choice;

while(choice!=5)

15

{

cout<<"\n\t\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\t\t\t\t\* MAIN MENU \*";

cout<<"\n\t\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\n\n\t\t\t1.Book A Room";

cout<<"\n\t\t\t2.Customer Record";

cout<<"\n\t\t\t3.Rooms Allotted";

cout<<"\n\t\t\t4.Edit Record";

cout<<"\n\t\t\t5.Exit";

cout<<"\n\n\t\t\tEnter Your Choice: ";

cin>>choice;

switch(choice)

{

case 1: add();

break;

case 2: display();

break;

case 3: rooms();

break;

case 4: edit();

break;

case 5: break;

default:

{

cout<<"\n\n\t\t\tWrong choice!!!";

cout<<"\n\t\t\tPress any key to continue!!";

}

}

}

}

16

void hotel::add()

{

int r,flag;

ofstream fout("Record.dat",ios::app);

cout<<"\n Enter Customer Details";

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\n Room no: ";

cin>>r;

flag=check(r);

if(flag)

cout<<"\n Sorry..!!!Room is already booked";

else

{

room\_no=r;

cout<<" Name: ";

scanf("%s",name);

cout<<" Address: ";

scanf("%s",address);

cout<<" Phone No: ";

scanf("%s",phone);

fout.write((char\*)this,sizeof(hotel));

cout<<"\n Room is booked!!!";

}

cout<<"\n Press any key to continue!!";

fout.close();

}

17

void hotel::display()

{

ifstream fin("Record.dat",ios::in);

int r,flag;

cout<<"\n Enter room no: ";

cin>>r;

while(!fin.eof())

{

fin.read((char\*)this,sizeof(hotel));

if(room\_no==r)

{

cout<<"\n Customer Details";

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\n Room no: "<<room\_no;

cout<<"\n Name: "<<name;

cout<<"\n Address: "<<address;

cout<<"\n Phone no: "<<phone;

flag=1;

break;

}

}

if(flag==0)

cout<<"\n Sorry Room no. not found or vacant!!";

cout<<"\n\n Press any key to continue!!";

fin.close();

}

18

void hotel::rooms()

{

ifstream fin("Record.dat",ios::in);

cout<<"\n\t\t\tList Of Rooms Allotted";

cout<<"\n\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\n Room No.\tName\t\tAddress\t\tPhone No.\n";

while(!fin.eof())

{

fin.read((char\*)this,sizeof(hotel));

cout<<"\n\n "<<room\_no<<"\t\t"<<name;

cout<<"\t\t"<<address<<"\t\t"<<phone;

}

cout<<"\n\n\n\t\t\tPress any key to continue!!";

fin.close();

}

void hotel::edit()

{

int choice,r;

cout<<"\n EDIT MENU";

cout<<"\n \*\*\*\*\*\*\*\*\*";

cout<<"\n\n 1.Modify Customer Record";

cout<<"\n 2.Delete Customer Record";

cout<<"\n Enter your choice: ";

cin>>choice;

19

cout<<"\n Enter room no: ";

cin>>r;

switch(choice)

{

case 1: modify(r);

break;

case 2: delete\_rec(r);

break;

default: cout<<"\n Wrong Choice!!";

}

cout<<"\n Press any key to continue!!!";

}

int hotel::check(int r)

{

int flag=0;

ifstream fin("Record.dat",ios::in);

while(!fin.eof())

{

fin.read((char\*)this,sizeof(hotel));

if(room\_no==r)

{

flag=1;

break;

}

}

fin.close();

20

return(flag);

}

void hotel::modify(int r)

{

long pos,flag=0;

fstream file("Record.dat",ios::in|ios::out|ios::binary);

while(!file.eof())

{

pos=file.tellg();

file.read((char\*)this,sizeof(hotel));

if(room\_no==r)

{

cout<<"\n Enter New Details";

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n Name: ";

gets(name);

cout<<" Address: ";

gets(address);

cout<<" Phone no: ";

gets(phone);

file.seekg(pos);

file.write((char\*)this,sizeof(hotel));

cout<<"\n Record is modified!!";

flag=1;

break;

}

}

if(flag==0)

21

cout<<"\n Sorry Room no. not found or vacant!!";

file.close();

}

void hotel::delete\_rec(int r)

{

int flag=0;

char ch;

ifstream fin("Record.dat",ios::in);

ofstream fout("temp.dat",ios::out);

while(!fin.eof())

{

fin.read((char\*)this,sizeof(hotel));

if(room\_no==r)

{

cout<<"\n Name: "<<name;

cout<<"\n Address: "<<address;

cout<<"\n Pone No: "<<phone;

cout<<"\n\n Do you want to delete this record(y/n): ";

cin>>ch;

if(ch=='n')

fout.write((char\*)this,sizeof(hotel));

flag=1;

}

else

fout.write((char\*)this,sizeof(hotel));

}

22

fin.close();

fout.close();

if(flag==0)

cout<<"\n Sorry room no. not found or vacant!!";

else

{

remove("Record.dat");

rename("temp.dat","Record.dat");

}

}

int main()

{

hotel h;

cout<<"\n\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\t\t\t\* HOTEL ROOM MANAGEMENT SYSTEM \*";

cout<<"\n\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

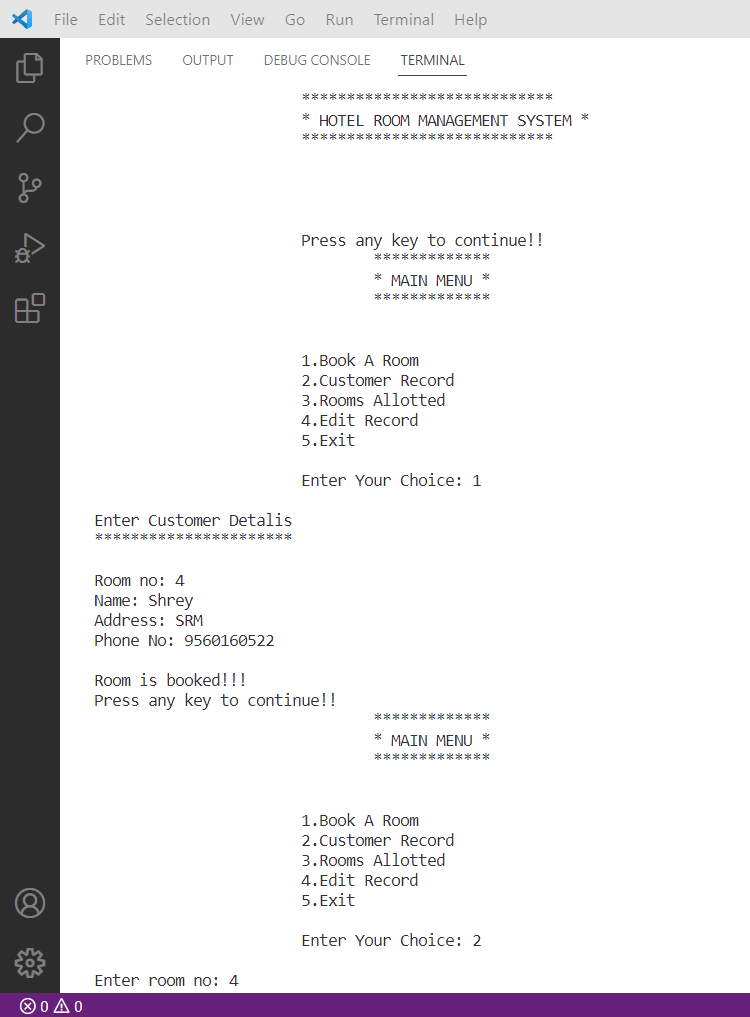
cout<<"\n\n\n\n\n\t\t\tPress any key to continue!!";

h.main\_menu();

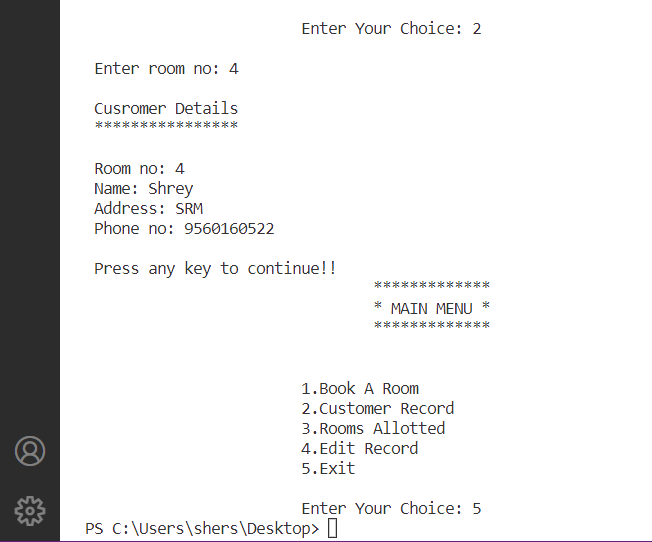
}

23

**OUTPUT**

****

24

****

25

**CONCLUSION**

Finally, in the Online Room Room Booking System, we have developed a secure, user-friendly Hotel Management System. This system can take care of each member, whether its owner or customer. This system will help them to properly manage their hotel and help in growth without creating any hassle.

26

**REFERENCES**

* Online Hotel Management System Project - ProjectsGeek
* https://www.javatpoint.com/
* <https://hackr.io/>
* https://www.programiz.

27