HOTEL ROOM BOOKING MANAGEMENT SYSTEM

Project submitted to the

Thiruvalluvar University, Vellore.

In partial fulfilment for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted by

KARTHI KAYAN R (40120U09020)

Under the Guidance of

Mr. S. MADHANMOHAN, MCA., MBA., M.Phil.,(ph.D)

Assistant Professor in Computer Applications

Arignar Anna Government Arts College

Villupuram – 605 602.

DEPARTMENT OF COMPUTER APPLICATIONS

ARIGNAR ANNA GOVERNMENT ARTS COLLEGE

VILLUPURAM- 605 602

MAY 2023.

HOTEL ROOM BOOKING MANAGEMENT SYSTEM

Project submitted to the

Thiruvalluvar University, Vellore.

In partial fulfilment for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted by

KARTHI KAYAN R (40120U09020)

Under the Guidance of

Mr. S. MADHANMOHAN, MCA., MBA., M.Phil., (Ph.D.)

Assistant Professor in Computer Applications

Arignar Anna Government Arts College

Villupuram – 605 602.

DEPARTMENT OF COMPUTER APPLICATIONS

ARIGNAR ANNA GOVERNMENT ARTS COLLEGE

VILLUPURAM- 605 602

MAY 2023.

ARIGNAR ANNA GOVT. ARTS COLLEGE

VILLUPURAM- 605 602

DEPARTMENT OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the report entitled HOTEL ROOM BOOKING MANAGEMENT SYSTEM Submitted in partial fulfilment of requirement for the award of degree of BACHELOR OF COMPUTER APPLICATIONS is a bonafide record of the original work done by

KARTHI KAYAN R (40120U09020)

Department of Computer Applications,

Arignar Anna Government Arts College,

Villupuram-605 602

During the period December 2022 – April 2023

Signature of the HOD **Signature of the Guide**

Date:

VIVA VOCE Examination for this project Report was held on_____

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

First and foremost we would like to thank God Almighty for making it possible to complete this project successfully. We am indented to extend my heartfelt Gratitude and sincere thanks to my beloved principal Dr. R. SHIVAKUMAR, who is the guiding light behind every student in our college.

We would like to Mr. S. MADHANMOHAN, Assistant Professor, Head of the Department, Department of Computer Applications who give this opportunity to complete this project.

We are extremely thankful to my internal project guide Mr. S. MADHANMOHAN, Assistant Professor, Head of the Department, Department of Computer Applications for their encouragement and support to finish the project successfully.

Lastly, but most importantly we extend my sincere thanks to other faculty member of Computer Applications Department, my parents, friends, well-wisher and everyone who have helped in the least possible way, for their kind help guidance and suggestions to make this project par excellence.

KARTHI KAYAN R

TABLE OF CONTENTS

ABSTRACT

The hotel room booking management abstract describes the system for managing hotel room bookings. The system includes a user interface that allows customers to search for available rooms, make reservations, and view their booking details. The system also includes a back-end interface for hotel staff to manage room inventory, check-in and check-out guests, and generate reports on occupancy and revenue. The system utilizes a database to store and retrieve booking information and integrates with payment gateways to process customer payments. The hotel room booking management system is designed to streamline the booking process, improve customer experience, and increase operational efficiency for hotel staff.

Overall this project of ours is being developed to help the staff of hotel to maintain the hotel in the best way possible and also reduce the human efforts.

CHAPTER 1

INTRODUCTION

This chapter gives an overview about the aims, objectives, features, background and operation environment of the system.

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The main aim and objectives are as follows:

- Online book issue
- Request to change room's count and increase or decrease cost per hour for hotel staff.
- customer login page where customer can book rooms issued by hotel and date and time of check out.
- Customer can search availability of rooms
- A admin login page can see the total profit, total rooms booked, live customer details, repeated customers detail, view report of hotel and download it.
- Repeated customer can get 10% cashback of old payment

1.2 FEATURES

Administrator

1. ROOMS DETAILS
2. LIVE CUSTOMER DETAILS
3. REPEATED CUSTOMERS LIST
4. ALL CUSTOMERS mail id and count LIST

5. TOTAL PROFIT
6. REPORT DISPLAY
7. DOWNLOAD REPORT TEXT FILE
8. FOR CUSTOM SETTINGS

Customer

1. FOR ROOM BOOKING
2. FOR VACATE A ROOM

1.3 BACKGROUND OF PROJECT

- ✓ Hotel Room Booking Management System is an application which refers to hotel systems which are generally small or medium in size. It is used by hotel staff to manage the hotel rooms using a computerized system where he/she can record various transactions like issue of rooms, check out of rooms, addition of new customers, addition of new rooms etc.

- ✓ rooms and customer maintenance modules are also included in this system which would keep track of the customers using the rooms and also a detailed description about the rooms a hotel contains. With this computerized system there will be no loss of rooms charge record or member record which generally happens when a non-computerized system is used.

- ✓ In addition, report module is also included in Hotel Room Booking Management System. If user's position is admin, the user is able to generate different kinds of reports like lists of customers registered, list of rooms, issue and return reports.

- ✓ All these modules are able to help hotel staff to manage the hotel with more convenience and in a more efficient way as compared to hotel systems which are not computerized.

1.4 OPERATION ENVIRONMENT

PROCESSOR	INTEL CORE PROCESSOR OR BETTER PERFORMANCE
OPERATING SYSTEM	WINDOWS VISTA ,WINDOWS10, UBUNTU
MEMORY	1GB RAM OR MORE
HARD DISK SPACE	MINIMUM 3 GB FOR DATABASE USAGE FOR FUTURE
DATABASE	MY SQL

CHAPTER 2

SYSTEM ANALYSIS

In this chapter, we will discuss and analyse about the developing process of Hotel Room Booking Management System including software requirement specification (SRS) and comparison between existing and proposed system. The functional and non-functional requirements are included in SRS part to provide complete description and overview of system requirement before the developing process is carried out. Besides that, existing vs proposed provides a view of how the proposed system will be more efficient than the existing one.

2.1 SOFTWARE REQUIREMENT SPECIFICATION

2.1.1 GENERAL DESCRIPTION

PRODUCT DESCRIPTION:

Hotel Room Booking Management System is a computerized system which helps user (Hotel staff) to manage the hotel daily activity in electronic format. It reduces the risk of paper work such as file lost, file damaged and time consuming. It can help user to manage the transaction or record more effectively and timesaving.

PROBLEM STATEMENT:

The problem occurred before having computerized system includes:

- File lost

  When computerized system is not implemented file is always lost because of human environment. Sometimes due to some human error there may be a loss of records.

- File damaged

  When a computerized system is not there file is always lost due to some accident like spilling of water by some member on file accidentally. Besides some natural disaster like floods or fires may also damage the files

- Difficult to search record

  When there is no computerized system there is always a difficulty in searching of records if the records are large in number.

- Space consuming

  After the number of records become large the space for physical storage of file and records also increase if no computerized system is implemented.

- Cost consuming

  As there is no computerized system to add each record paper will be needed which will increase the cost for the management of hotel.

2.1.2 SYSTEM OBJECTIVES

- Improvement in control and performance

The system is developed to cope up with the current issues and problems of hotel. The system can add user, validate user and is also bug free.

- Save cost

After computerized system is implemented less human force will be required to maintain the hotel thus reducing the overall cost.

- Save time

Hotel staff is able to search record by using few clicks of mouse and few search keywords thus saving his valuable time.

2.1.3 SYSTEM REQUIREMENTS

2.1.3.1 NON FUNCTIONAL REQUIREMENTS

- Product Requirements

EFFICIENCY REQUIREMENT

When a hotel room booking management system will be implemented hotel staff and customers will easily access hotels as searching and rooms booking transaction will be very faster.

RELIABILITY REQUIREMENT

The system should accurately performs member registration, member validation, report generation, room transaction and search

USABILITY REQUIREMENT

The system is designed for a user friendly environment so that customers and staff of hotel can perform the various tasks easily and in an effective way.

ORGANIZATIONAL REQUIREMENT

- IMPLEMENTATION REQUIREMNTS

In implementing whole system uses java in front end with jdbc and java as server side language which will be used for database connectivity and the backend, i.e. the database part is developed using MySQL.

❖ DELIVERY REQUIREMENTS

The whole system is expected to be delivered in six months of time with a weekly evaluation by the project guide.

CHAPTER 3

SYSTEM SPECIFICATIONS

SOFTWARE AND HARDWARE REQUIREMENTS

This section describes the software and hardware requirements of the system

3.1 SOFTWARE REQUIREMENTS

MY SQL

Languages used:

Front End: java

Connectivity: jdbc

Back End: MySQL

- Operating system - Windows 10 is used as the operating system as it is stable and supports more features and is more user friendly

- Database MYSQL - MYSQL is used as database as it easy to maintain and retrieve records by simple queries which are in English language which are easy to understand and easy to write.

- Development tools and Programming language - java is used to write the whole code and develop the system.

3.2 HARDWARE REQUIREMENTS

Processor: intel core i5 2nd generation

Ram: 1gb

- Intel core i5 2nd generation is used as a processor because it is fast than other processors a provide reliable and stable and we can run our pc for long time. By using this processor we can keep on developing our project without any worries.

- Ram 1 gb is used as it will provide fast reading and writing capabilities and will in turn support in processing.

CHAPTER 4

SOFTWARE DESCRIPTION

The whole Project is divided in two parts the front end and the back end.

4.1 FRONT END

The front end is designed using of Java

- JAVA – PROGRAMMING LANGUAGE

Java is a high-level, object-oriented programming language developed by Sun Microsystems (now owned by Oracle Corporation) in the mid-1990s. It was designed to be platform-independent, meaning that Java programs can run on any hardware or operating system that has a Java Virtual Machine (JVM) installed.

Java is known for its simplicity, readability, and robustness, making it a popular choice for developing enterprise-level applications, mobile apps, and web applications. Java's syntax is similar to that of C++ and C#, making it relatively easy for developers familiar with these languages to learn.

Java's object-oriented programming features include encapsulation, inheritance, and polymorphism. It also supports multithreading, allowing multiple threads to run simultaneously, which is useful for developing applications that require high concurrency.

Java has a vast standard library that provides many pre-built classes and methods, including support for networking, database connectivity, and graphical user interfaces. It also has a large community of developers who contribute to open-source libraries and frameworks, such as Spring, Hibernate, and Apache Struts.

To run Java programs, a JVM is required, which interprets Java bytecode and executes it on the target hardware. Java programs can be compiled into bytecode using a Java compiler, which produces .class files that can be run on any JVM.

4.2 CONNECTIVITY

We use jdbc to connect the java code and MySQL database.

JDBC – JAVA DATABASE CONNECTIVITY

JDBC stands for "Java Database Connectivity" and is a Java API that provides a standard way for Java programs to access and manipulate data stored in databases. It allows developers to write database-independent code that can connect to any database that supports JDBC.

JDBC consists of two main components: the JDBC API and the JDBC Driver API. The JDBC API defines a set of classes and interfaces that provide a uniform interface for connecting to and interacting with a database. The JDBC Driver API defines a set of interfaces that driver developers must implement to provide connectivity to specific databases.

To use JDBC, a developer must first load the appropriate JDBC driver for the target database. The driver is responsible for communicating with the database server and translating the JDBC API calls into the database-specific protocol.

Once the driver is loaded, a developer can create a Connection object to establish a connection to the database. From there, a Statement object can be created to execute SQL statements and retrieve data from the database. JDBC supports a wide range of SQL statements, including queries, updates, and stored procedures.

JDBC also supports transaction management, allowing developers to group related SQL statements into a single transaction that can be committed or rolled back as a unit.

JDBC is widely used in enterprise-level applications that require access to large databases. It is supported by most relational database systems, including Oracle, MySQL, PostgreSQL, and Microsoft SQL Server.

4.3 BACK END

The back end is designed using MySQL which is used to design the Databases.

MYSQL

MySQL ("My S-Q-L", officially, but also called "My Sequel") is (as of July 2013) the world's second most widely used open-source relational database management system (RDBMS). It is named after co-founder Michael Widenius daughter, MySQL phrase stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for profit firm, the Swedish

company MySQL AB, now owned by Oracle Corporation. MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl / PHP/ Python". Free-software-open source projects that require a full-featured database management system often use MySQL. For commercial use, several paid editions are available, and offer additional functionality.
Applications which use MySQL databases include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, Drupal and other software. MySQL is also used in many high-profile, large-scale websites, including Wikipedia, Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 SOURCE CODE

Main.java

```
package residency_project_files;       // my project's user-defined package

import java.io.IOException;
import java.sql.SQLException;
import java.util.InputMismatchException;
import java.util.Scanner;

public class Main {
        public static void main(String[] args) {

    int firstOption = 0;
    Scanner in = new Scanner(System.in);
    Utils.starTitle('#');
```

```
        System.out.println("\nENTER 1 IF YOU ARE CUSTOMER");
        System.out.println("ENTER 2 IF YOU ARE ADMINISTATOR");
        System.out.print("\nENTER YOUR VALUE HERE : ");

        try{
          firstOption = in.nextInt();
          if(firstOption > 2 || firstOption < 1) {
                throw new InputMismatchException();
          }
        }
        catch (InputMismatchException e) {
                System.err.println("! enter your value rightly");
                System.err.println("! run the program from first");
                return;
        }
        switch(firstOption){
          case 1:
          {
            CustomerBlock cb = new CustomerBlock();
            break;
          }
          case 2:
          {
            try {
                                                AdministrationBlock ab = new AdministrationBlock();
                                        } catch (SQLException | IOException e) {
                                                System.err.println("problem at Administration Block");
                                        }
            break;
          }
        }
    }
}
```

AdministratorBlock.java

```
package residency_project_files;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.InputMismatchException;
import java.util.Scanner;

public class AdministrationBlock {
	AdministrationBlock() throws SQLException, IOException{

    Scanner in = new Scanner(System.in);
    int adminBlOption = 1;
    lable :
    while(true) {
    Utils.blockTitle("ADMINISTRATION BLOCK");
    System.out.println("\nENTER 1 FOR ROOMS DETAILS");
    System.out.println("ENTER 2 FOR LIVE CUSTOMER DETAILS");
    System.out.println("ENTER 3 FOR REPEATED CUSTOMERS LIST");
    System.out.println("ENTER 4 FOR ALL CUSTOMERS mail id and count LIST");
    System.out.println("ENTER 5 FOR TOTAL PROFIT");
    System.out.println("ENTER 6 FOR REPORT DISPLAY");
    System.out.println("ENTER 7 FOR DOWNLOAD REPORT TEXT FILE");
    System.out.println("ENTER 8 FOR CUSTOM SETTINGS");
    System.out.println("ENTER NUMBERS EXCEPT OPTION TO EXIT");
    System.out.println("\n\t\t//must once it will use before any transaction");
    System.out.print("\nENTER YOUR VALUE HERE : ");

    try {
       adminBlOption = in.nextInt();
    }
    catch(InputMismatchException e) {
    	System.err.println("! enter option rightly");
    	AdministrationBlock ab = new AdministrationBlock();
    }
    switch(adminBlOption) {
    //-------------------------------------------------------------------------------------
    case 1:{

    	int[] aofr = AllMethodsBlock.arrayOfRooms();
    	int totalRooms = aofr.length;
    	Utils.boxTitle("ROOMS DETAILS");
    	System.out.println("\n");
    	Utils.subTitle("TOTAL BOOKED ROOMS");
    	System.out.println("TOTAL BOOKED ROOMS : "+totalRooms);
    	System.out.println("\n");
    	Utils.subTitle("BOOKED ROOMS LIST");
    	System.out.print("BOOKED ROOMS LIST : ");
    	for(int i = 0; i < totalRooms; i++) {
    		System.out.print(aofr[i]);
    		if(i == totalRooms-1) {
    			System.out.println('.');
    		}
    		else {
    			System.out.print(", ");
    		}
    	}
    	System.out.println("\n");
    	Utils.subTitle("BUILDING VIEW");
    	System.out.println("BUILDING VIEW OF ROOMS : ");
    	AllMethodsBlock.printRooms('a', aofr);
       break;
    }
    //-------------------------------------------------------------------------------------
```

```
case 2:{
	Utils.boxTitle("LIVE CUSTOMER DETAILS");
	System.out.println("\n");
	ResultSet rst = DAO.getLiveCustomers();
	while(rst.next()) {
		System.out.printf("%2d  %-20s %s\n",rst.getInt(1),rst.getString(2),rst.getString(3));
	}
   break;
}
//-------------------------------------------------------------------------------------
case 3:{
	Utils.boxTitle("REPEATED CUSTOMERS LIST");
	System.out.println("\n");
	ResultSet rst = DAO.getStatistics();
	while(rst.next()) {
		if(rst.getInt(3) > 1) {
		System.out.printf("%35s %2d\n",rst.getString(1),rst.getInt(3));
		}
	}
	break;
}
//-------------------------------------------------------------------------------------
case 4:{
	Utils.boxTitle("ALL CUSTOMERS mail id and count LIST");
	System.out.println("\n");
	int count = 0;
	ResultSet rst = DAO.getStatistics();
	while(rst.next()) {
		System.out.println(rst.getString(1));
		count++;
	}
	ResultSet rst1 = DAO.getLiveCustomers();
	while(rst1.next()) {
		if(!DAO.checkMail(rst1.getString(3))) {
			System.out.println(rst1.getString(3));
			count++;
		}
	}
	System.out.println("\n");
	Utils.boxTitle("total customers");
	System.out.println("\n");
	System.out.println("total customers : "+count);
	break;
}
//-------------------------------------------------------------------------------------
case 5:{
	Utils.boxTitle("TOTAL PROFIT");
	System.out.println("\n");
	System.out.println("total profit is : "+DAO.getProfit());
	break;
}
//-------------------------------------------------------------------------------------
case 6:{

	Utils.starTitle('#');

	System.out.println("\n");
	Utils.boxTitle("ROOMS DETAILS");
	System.out.println("\n");
	Utils.subTitle("TOTAL BOOKED ROOMS");
	int[] aofr = AllMethodsBlock.arrayOfRooms();
	int totalRooms = aofr.length;
	System.out.println("TOTAL BOOKED ROOMS : "+totalRooms);
	System.out.println("\n");
	Utils.subTitle("BOOKED ROOMS LIST");
	System.out.print("BOOKED ROOMS LIST : ");
	for(int i = 0; i < totalRooms; i++) {
		System.out.print(aofr[i]);
		if(i == totalRooms-1) {
			System.out.println('.');
		}
```

```
                else {
                        System.out.print(", ");
                }
        }
        System.out.println("\n");
        Utils.subTitle("BUILDING VIEW");
        System.out.println("BUILDING VIEW OF ROOMS : ");
        AllMethodsBlock.printRooms('a', aofr);
        System.out.println("\n");
        //-------------------------------------------------------------------------------

        Utils.boxTitle("LIVE CUSTOMER DETAILS");
        System.out.println("\n");
        ResultSet rst = DAO.getLiveCustomers();
        while(rst.next()) {
                System.out.printf("%2d  %-20s %s\n",rst.getInt(1),rst.getString(2),rst.getString(3));
        }
        System.out.println("\n");

        //-------------------------------------------------------------------------------

        Utils.boxTitle("REPEATED CUSTOMERS LIST");
        System.out.println("\n");
        ResultSet rst1 = DAO.getStatistics();
        while(rst1.next()) {
                if(rst1.getInt(3) > 1) {
                System.out.printf("%35s %2d\n",rst1.getString(1),rst1.getInt(3));
                }
        }
        System.out.println("\n");



        //-------------------------------------------------------------------------------
        Utils.boxTitle("ALL CUSTOMERS mail id and count LIST");
        System.out.println("\n");
        int count = 0;
        ResultSet rst11 = DAO.getStatistics();
        while(rst11.next()) {
                System.out.println(rst11.getString(1));
                count++;
        }
        ResultSet rst111 = DAO.getLiveCustomers();
        while(rst111.next()) {
                if(!DAO.checkMail(rst111.getString(3))) {
                        System.out.println(rst111.getString(3));
                        count++;
                }
        }

        System.out.println("\n");
        Utils.boxTitle("total customers");
        System.out.println("\n");
        System.out.println("total customers : "+count);
        System.out.println("\n");

        //-------------------------------------------------------------------------------


        Utils.boxTitle("TOTAL PROFIT");
        System.out.println("\n");
        System.out.println("total profit is : "+DAO.getProfit());
        break;
}
//-------------------------------------------------------------------------------
case 7:{
        File folder = new File("C:\\Users\\HP\\Desktop\\Residency_project_folder");
        File file = new File("C:\\Users\\HP\\Desktop\\Residency_project_folder\\Report.txt");
        if(!folder.exists()) {
                folder.mkdir();
        }
```

```
            if(!file.exists()) {
                        file.createNewFile();
                        FileWriter writer = new FileWriter(file);
                        writer.write( AllMethodsBlock.forTXT());
                        writer.flush();
                        writer.close();
            }else {
                        file.delete();
                        file.createNewFile();
                        FileWriter writer = new FileWriter(file);
                        writer.write( AllMethodsBlock.forTXT());
                        writer.flush();
                        writer.close();
            }

            break;
      }
      //-------------------------------------------------------------------------------------
      case 8:{
         Utils.blockTitle("CUSTOM Settings BLOCK");

         Connection con = DriverManager.getConnection("jdbc:mysql://localHost:3306/hotel_tables","root","tiger");
         Statement smt = con.createStatement();

         String query = "select * from hotel_settings";
         ResultSet rst = smt.executeQuery(query);
         rst.next();
         boolean state = rst.getBoolean(4);
         if(state == false) {
              System.out.println("\n//must once it will use before any transaction");
              System.out.println("so you can't change it");
              break;
         }
         else if(state == true) {
              String query1 = "select * from statistics;";
              ResultSet rst1 = smt.executeQuery(query1);
              boolean a = rst1.next();
              String query2 = "select * from customers;";
              ResultSet rst2 = smt.executeQuery(query2);
              boolean b = rst2.next();
              if((a == false)&&(b == false)) {

              int hourCost = 50;
              int roomCount = 50;
              System.out.print("enter how many number of rooms you need to set : ");
              try {
                        roomCount = in.nextInt();
              } catch (InputMismatchException e) {
                        System.err.println("! enter your rooms count rightly");
                        System.err.println("! run the program from first");
                        return;
              }

              System.out.print("enter the cost per hour you need to set : ");
              try {
                        hourCost = in.nextInt();
              } catch (InputMismatchException e) {
                        System.err.println("! enter your rooms count rightly");
                        System.err.println("! run the program from first");
                        return;
              }
              String queryToSet = "update hotel_settings set rooms_count = "+roomCount+", hour_cost = "+hourCost+", modify_state =
"+false+" where hotel_id = 1";
              smt.execute(queryToSet);
              System.out.println("updated");
              }
              else {
                        System.out.println("\n//must once it will use before any transaction");
              System.out.println("so you can't change it");
              String query3 = "update hotel_settings set modify_state = "+false+ " where hotel_id = 1";
              smt.execute(query3);
```

```
            }
        }
    }
     //-------------------------------------------------------------------------------------------
     default :{
            System.out.println("thank you");
            break lable;
     }
     //-------------------------------------------------------------------------------------------
     }
  }

//   ResultSet rst1 = smt.executeQuery(Query);
//   if(rst1.next()) {
//           state = false;
//   }
//   con.close();
//   break;
   }
}
```

CustomerBlock.java

```
package residency_project_files;

import java.sql.SQLException;
import java.util.InputMismatchException;
import java.util.Scanner;

public class CustomerBlock {
   CustomerBlock(){

      int custBlOption = 0;
      Scanner in = new Scanner(System.in);

      Utils.blockTitle("CUSTOMER BLOCK");

      System.out.println("\nENTER 1 FOR ROOM BOOKING");
      System.out.println("ENTER 2 FOR VACATE A ROOM");
      System.out.print("\nENTER YOUR VALUE HERE : ");

      try{
           custBlOption = in.nextInt();
        if(custBlOption > 2 || custBlOption < 1) {
            throw new InputMismatchException();
        }
      }
      catch (InputMismatchException e) {
            System.err.println("! enter your option rightly");
            CustomerBlock cb = new CustomerBlock();
      }

      switch(custBlOption){
        case 1:{
          try {
                                                RoomBookingBlock rbb = new RoomBookingBlock();
                                            } catch (SQLException e) {
                                                System.err.println("! problem at RoomBookingBlock");
                                            }
          break;
        }
        case 2:{
            try {
                                                RoomVacate rv = new RoomVacate();
                                            } catch (SQLException e) {
                                                System.err.println("! problem at RoomVacate");
```

```
                                        }
                    break;
                }
            }
        }
    }
```

AllMethodsBlock.java

```
package residency_project_files;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Arrays;
import java.util.Map;
import java.util.TreeMap;

public class AllMethodsBlock {
        static int totalRooms = DAO.getRoomsCount();
        //1-------------------------------------------------------------------------------------------
        static int roomsCount() {
                TreeMap<Integer, Integer> tm = new TreeMap<>();
                tm = DAO.getRoomId();
                int count = 0;
                for(Map.Entry<Integer, Integer> entry:tm.entrySet()) {
                        count++;
                }
                return count;
        }
        //2-------------------------------------------------------------------------------------------

        static int[] arrayOfRooms(){
                TreeMap<Integer, Integer> tm = new TreeMap<>();
                tm = DAO.getRoomId();
                int[] array = new int[roomsCount()];
                int ptr = 0;
                for(Map.Entry<Integer, Integer> entry:tm.entrySet()) {
                        array[ptr] = entry.getKey();
                        ptr++;
                }
                Arrays.sort(array);
                return array;
        }
        //3-------------------------------------------------------------------------------------------

        static int[] roomChooser(int roomCount){
                TreeMap<Integer, Integer> tm = new TreeMap<>();
                tm = DAO.getRoomId();
                int[] array = new int[roomCount];
                int ptr = 0;
                for(int i = 1; i <= totalRooms && ptr < roomCount; i++) {
                        if(!tm.containsKey(i)) {
                                array[ptr] = i;
                                ptr++;
                        }
                }
                return array;
        }

        //4-------------------------------------------------------------------------------------------

        static void printRooms(char AdminOrCust, int[] ar) {
                Arrays.sort(ar);
```

```
		int ptr = ar.length-1;
		for(int i = totalRooms; i> 0; i--) {

			System.out.println(" ---  ---  ---  ---  --- ");

			for(int j = 1; j <= 5; j++) {

				boolean b = false;
				if(ptr >= 0) {
					if(i == ar[ptr]) {
						b = true;
						if(ptr>0) {
							ptr--;
						}
					}
				}
				if(AdminOrCust == 'c') {
					if(i<10) {
						if(b) {
							System.out.print("|0"+i+'Y'+"| ");
						}
						else {
							System.out.print("|0"+i+" | ");
						}
					}
					else {
						if(b) {
							System.out.print("|"+i+'Y'+"| ");
						}
						else {
							System.out.print("|"+i+" | ");
						}
					}
				}
				else if(AdminOrCust == 'a') {
					if(i<10) {
						if(b) {
							System.out.print("|0"+i+'B'+"| ");
						}
						else {
							System.out.print("|0"+i+'e'+"| ");
						}
					}
					else {
						if(b) {
							System.out.print("|"+i+'B'+"| ");
						}
						else {
							System.out.print("|"+i+'e'+"| ");
						}
					}
				}
				i--;
			}
			System.out.println("\n ---  ---  ---  ---  --- ");
			i++;
		}

}
//5-------------------------------------------------------------------------------------

static String printRoomsTXT(char AdminOrCust, int[] ar) {
		Arrays.sort(ar);
		int ptr = ar.length-1;
		String totalStr = "";
		for(int i = totalRooms; i> 0; i--) {

			totalStr = totalStr.concat(" ---  ---  ---  ---  --- \n");

			for(int j = 1; j <= 5; j++) {
```

```
                                        boolean b = false;
                                        if(ptr >= 0) {
                                                if(i == ar[ptr]) {
                                                        b = true;
                                                        if(ptr>0) {
                                                                ptr--;
                                                        }
                                                }
                                        }
                                        if(AdminOrCust == 'c') {
                                                if(i<10) {
                                                        if(b) {
                                                                totalStr = totalStr.concat("|0"+i+'Y'+"| ");
                                                        }
                                                        else {
                                                                totalStr = totalStr.concat("|0"+i+" | ");
                                                        }
                                                }
                                                else {
                                                        if(b) {
                                                                totalStr = totalStr.concat("|"+i+'Y'+"| ");
                                                        }
                                                        else {
                                                                totalStr = totalStr.concat("|"+i+" | ");
                                                        }
                                                }
                                        }
                                        else if(AdminOrCust == 'a') {
                                                if(i<10) {
                                                        if(b) {
                                                                totalStr = totalStr.concat("|0"+i+'B'+"| ");
                                                        }
                                                        else {
                                                                totalStr = totalStr.concat("|0"+i+'e'+"| ");
                                                        }
                                                }
                                                else {
                                                        if(b) {
                                                                totalStr = totalStr.concat("|"+i+'B'+"| ");
                                                        }
                                                        else {
                                                                totalStr = totalStr.concat("|"+i+'e'+"| ");
                                                        }
                                                }
                                        }
                                        i--;
                                }
                                totalStr = totalStr.concat("\n ---  ---  ---  ---  --- \n");
                                i++;
                        }
                        return totalStr;
}
//6-------------------------------------------------------------------------------------
static String forTXT() throws SQLException {

                String totalStr = Utils.starTitleTXT('#');

                totalStr = totalStr.concat("\n\n");
                totalStr = totalStr.concat(Utils.boxTitleTXT("ROOMS DETAILS"));
                totalStr = totalStr.concat("\n\n");
                totalStr = totalStr.concat(Utils.subTitleTXT("TOTAL BOOKED ROOMS"));
int[] aofr = AllMethodsBlock.arrayOfRooms();
int totalRooms = aofr.length;
totalStr = totalStr.concat("TOTAL BOOKED ROOMS : "+totalRooms);
totalStr = totalStr.concat("\n\n");
totalStr = totalStr.concat(Utils.subTitleTXT("BOOKED ROOMS LIST"));
totalStr = totalStr.concat("BOOKED ROOMS LIST : ");
for(int i = 0; i < totalRooms; i++) {
                totalStr = totalStr.concat(String.format("%d", aofr[i]));
                if(i == totalRooms-1) {
                                totalStr = totalStr.concat(".\n");
```

```
                }
                else {
                        totalStr = totalStr.concat(", ");
                }
        }
        totalStr = totalStr.concat("\n\n");
        totalStr = totalStr.concat(Utils.subTitleTXT("BUILDING VIEW"));
        totalStr = totalStr.concat("BUILDING VIEW OF ROOMS : \n");
        totalStr = totalStr.concat(AllMethodsBlock.printRoomsTXT('a', aofr));
        totalStr = totalStr.concat("\n\n");
        //-------------------------------------------------------------------------------

        totalStr = totalStr.concat(Utils.boxTitleTXT("LIVE CUSTOMER DETAILS"));
        totalStr = totalStr.concat("\n\n");
        ResultSet rst = DAO.getLiveCustomers();
        while(rst.next()) {
                totalStr = totalStr.concat(String.format("%2d  %-20s %s\n",rst.getInt(1),rst.getString(2),rst.getString(3)));
        }
        totalStr = totalStr.concat("\n\n");

        //-------------------------------------------------------------------------------

        totalStr = totalStr.concat(Utils.boxTitleTXT("REPEATED CUSTOMERS LIST"));
        totalStr = totalStr.concat("\n\n");
        ResultSet rst1 = DAO.getStatistics();
        while(rst1.next()) {
                if(rst1.getInt(3) > 1) {
                        totalStr = totalStr.concat(String.format("%35s %2d\n",rst1.getString(1),rst1.getInt(3)));
                }
        }
        totalStr = totalStr.concat("\n\n");



        //-------------------------------------------------------------------------------
        totalStr = totalStr.concat(Utils.boxTitleTXT("ALL CUSTOMERS mail id and count LIST"));
        totalStr = totalStr.concat("\n\n");
        int count = 0;
        ResultSet rst11 = DAO.getStatistics();
        while(rst11.next()) {
                totalStr = totalStr.concat(rst11.getString(1)+"\n");
                count++;
        }
        ResultSet rst111 = DAO.getLiveCustomers();
        while(rst111.next()) {
                if(!DAO.checkMail(rst111.getString(3))) {
                        totalStr = totalStr.concat(rst111.getString(3)+"\n");
                        count++;
                }
        }

        totalStr = totalStr.concat("\n\n");

        totalStr = totalStr.concat(Utils.boxTitleTXT("total customers"));
        totalStr = totalStr.concat("\n\n");
        totalStr = totalStr.concat("total customers : "+count);
        totalStr = totalStr.concat("\n\n");

        //-------------------------------------------------------------------------------


        totalStr = totalStr.concat(Utils.boxTitleTXT("TOTAL PROFIT"));
        totalStr = totalStr.concat("\n\n");
        totalStr = totalStr.concat("total profit is : "+DAO.getProfit());
        return totalStr;
        }
}
```

DAO.java

```
package residency_project_files;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
//import java.sql.Timestamp;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.util.Date;
//import java.util.Map;
import java.util.TreeMap;

public class DAO { //		database access object



	//1-------------------------------------------------------------------------------------------

	static TreeMap<Integer, Integer> getRoomId() {

		TreeMap<Integer, Integer> bookedRooms = new TreeMap<>();

		Connection con;
		try {
			con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables","root","tiger");
			Statement smt = con.createStatement();
			String query = "select * from rooms";
			ResultSet rs = smt.executeQuery(query);

			while(rs.next()) {
				int roomno = rs.getInt(1);
				int customerid = rs.getInt(2);
				bookedRooms.put(roomno, customerid);
			}
		} catch (SQLException e) {
			System.err.println("problem at DAO.static TreeMap<Integer, Integer> setHashMap()'s
database connection");
			return null;
		}

		return bookedRooms;
	}
	//2-------------------------------------------------------------------------------------------
	// id ,name ,mail_id ,members ,checkin ,time_by_customer
	static void booking(String name, String mailid, int members, double days, int[] arrayOfRooms) {
//	SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");
	SimpleDateFormat s0 = new SimpleDateFormat("aa");
	SimpleDateFormat s1 = new SimpleDateFormat("yyyy-MM-dd");
	SimpleDateFormat s2 = new SimpleDateFormat("hh:mm:ss");
	try {
		Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables","root","tiger");
		Statement smt = con.createStatement();
  //insert into customers(id, name, mail_id, members, check_in, time_by_cust) values (12,'karthi', 'karthikayan@gmail.com', 9, date, 2.5);
		Date date = new Date();
		java.sql.Timestamp sqldate = new java.sql.Timestamp(date.getTime());
		String query1 = "insert into customers values ("+arrayOfRooms[0]+", '"+name+"', '"+mailid+"', "+members+",
'"+sqldate+"' , "+days+")";
		smt.execute(query1);
	DAO.bookRooms(arrayOfRooms);
		System.out.println("\nConfermation of your details : \n\n");
		ResultSet rst = getCustomer(mailid);
		rst.next();
			int id1 = rst.getInt(1);
			String name1 = rst.getString(2);
			String mail_id1 = rst.getString(3);
```

```
                                int members1 = rst.getInt(4);
                                Date check_in1 = rst.getTimestamp(5);
                                float time_by_cust1 = rst.getFloat(6);
                                System.out.println("your booking id is : "+id1);
                                System.out.println("your name is : "+name1);
                                System.out.println("your mail id is : "+mail_id1);
                                System.out.println("count of members : "+members1+" and rooms : "+ DAO.getRoomsCount(id1));
                                System.out.println("your needed stay time : "+time_by_cust1);
                                System.out.println("your checkin time  : "+check_in1);

                        String a = s0.format(check_in1);
                        String b = "pm";

                        LocalDateTime lt1 = LocalDateTime.parse(s1.format(check_in1)+'T'+s2.format(check_in1));
                        double t1 = 0.0;
                        if(a.equals(b)){
                           t1 = (time_by_cust1*24)+12;
                        }
                        else{
                           t1 = time_by_cust1*24;
                        }
                        int t2 = (int)t1;
                        LocalDateTime lt = lt1.plusHours(t2);
                                System.out.println("your checkout time : "+lt);

                                con.close();
            } catch (SQLException e) {
                    System.err.println("problem at DAO.static void booking(String name, String mailid, int members, double days, int[]
arrayOfRooms)'s database connection");
                                return;
                                }


                    }
                    //3-----------------------------------------------------------------------------------------
                    static ResultSet getCustomer(String mail_id){
                                try {
                                        Connection con =
DriverManager.getConnection("jdbc:mysql://localHost:3306/hotel_tables","root","tiger");
                                        Statement smt = con.createStatement();
                                        String query = "select * from customers where mail_id = '"+mail_id+"'";
                                        return smt.executeQuery(query);
                                } catch (SQLException e) {
                                        System.err.println("problem at DAO.static ResultSet getCustomer(String mail_id)'s database
connection");
                                        return null;
                                }
                    }
                    //4-----------------------------------------------------------------------------------------
                    static ResultSet getLiveCustomers() {
                                try {
                                        Connection con =
DriverManager.getConnection("jdbc:mysql://localHost:3306/hotel_tables","root","tiger");
                                        Statement smt = con.createStatement();
                                        String query = "select * from customers";
                                        return smt.executeQuery(query);
                                } catch (SQLException e) {
                                        System.err.println("problem at DAO.static ResultSet getLiveCustomers()'s database connection");
                                        return null;
                                }
                    }
                    //-----------------------------------------------------------------------------------------
//                  static ResultSet getCustomersIdNameMail () throws SQLException {
//                          Connection con = DriverManager.getConnection("jdbc:mysql://localHost:3306/hotel_tables","root","tiger");
//                          Statement smt = con.createStatement();
//                          String query = "select id, name, mail_id from customers";
//                          return smt.executeQuery(query);
//                  }
                    //5-----------------------------------------------------------------------------------------
                    static void bookRooms(int[] array){
                                try {
```

```
                        Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables","root","tiger");
                        Statement smt = con.createStatement();
                        for(int a:array) {
                                String query = "insert into rooms values ("+a+", "+array[0]+")";
                                smt.execute(query);
                        }
                        con.close();
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static void bookRooms(int[] array)'s database connection");
                        return;
                }
        }
        //6-------------------------------------------------------------------------------------
        static boolean checkMail (String mail){
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select true from statistics where mail_id = '"+mail+"'";
                        ResultSet rst = smt.executeQuery(query);
                        if(!rst.next()) {
                                return false;
                        }
                        return rst.getBoolean(1);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static boolean checkMail (String mail)'s database connection");
                        return false;
                }
        }
        //7-------------------------------------------------------------------------------------
        static ResultSet getStatistics(String mail){
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select * from statistics where mail_id = '"+mail+"'";
                        return smt.executeQuery(query);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static ResultSet getStatistics(String mail)'s database connection");
                        return null;
                }
        }
        //8-------------------------------------------------------------------------------------
        static ResultSet getmail(int id) {
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select mail_id from customers where id = "+id;
                        return smt.executeQuery(query);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static ResultSet getmail(int id)'s database connection");
                        return null;
                }
        }
        //9-------------------------------------------------------------------------------------
        static int getRoomsCount(int id) {
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select count(room_no) from rooms where customer_id = "+id;
                        ResultSet rst = smt.executeQuery(query);
                        rst.next();
                        return rst.getInt(1);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static int getRoomsCount(int id)'s database connection");
                        return -1;
                }
        }
```

```
	//10----------------------------------------------------------------------------------------
	static void vacate(int id){
		try {
			Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
			Statement smt = con.createStatement();
			String query = "delete from customers where id = "+id;
			smt.execute(query);
			con.close();
		} catch (SQLException e) {
			System.err.println("problem at DAO.static TreeMap<Integer, Integer> setHashMap()'s database
connection");
			return;
		}
	}
	//----------------------------------------------------------------------------------------
/*	static void vacate(String mail) throws SQLException {
		Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root", "tiger");
		Statement smt = con.createStatement();
		String query = "delete from customers where mail_id = '"+mail+"'";
		smt.execute(query);
		con.close();
	}
*/
	//11----------------------------------------------------------------------------------------
	static void statisticsUpdate(String mail, int repeat, int newhours){
		try {
			Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
			Statement smt = con.createStatement();
			String query = "update statistics set repeatation = "+repeat+" ,total_time = "+newhours+" where mail_id
= '"+mail+"'";
			smt.execute(query);
			con.close();
		} catch (SQLException e) {
			System.err.println("problem at DAO.static void statisticsUpdate(String mail, int repeat, int newhours)'s
database connection");
			return;
		}
	}
	//12----------------------------------------------------------------------------------------
	static void statisticsInsert(String mail, int newhours) {
		try {
			Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
			Statement smt = con.createStatement();
			String query = "insert into statistics values ('"+mail+"',"+newhours+",1)";
			smt.execute(query);
			con.close();
		}  catch (SQLException e) {
			System.err.println("problem at DAO.static void statisticsInsert(String mail, int newhours)'s database
connection");
			return;
		}
	}
	//13----------------------------------------------------------------------------------------
	static ResultSet getStatistics(){
		try {
			Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
			Statement smt = con.createStatement();
			String query = "select * from statistics";
			return smt.executeQuery(query);
		} catch (SQLException e) {
			System.err.println("problem at DAO.static ResultSet getStatistics()'s database connection");
			return null;
		}
	}
	//14----------------------------------------------------------------------------------------
	static int getProfit(){
		try {
```

```
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select profit from hotel_settings";
                        ResultSet rst = smt.executeQuery(query);
                        rst.next();
                        return rst.getInt(1);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static int getProfit()'s database connection");
                        return -1;
                }
        }
        //15-------------------------------------------------------------------------------------
        static void setProfit(int total){
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "update hotel_settings set profit = "+total+" where hotel_id = 1";
                        smt.execute(query);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static void setProfit(int total)'s database connection");
                        return;
                }
        }
        //16-------------------------------------------------------------------------------------
        static int getRoomsCount() {
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select rooms_count from hotel_settings";
                        ResultSet rst = smt.executeQuery(query);
                        rst.next();
                        return rst.getInt(1);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static int getRoomsCount()'s database connection");
                        return -1;
                }
        }
        //17-------------------------------------------------------------------------------------
        static int gethourCost() {
                try {
                        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/hotel_tables", "root",
"tiger");
                        Statement smt = con.createStatement();
                        String query = "select hour_cost from hotel_settings";
                        ResultSet rst = smt.executeQuery(query);
                        rst.next();
                        return rst.getInt(1);
                } catch (SQLException e) {
                        System.err.println("problem at DAO.static int getHourCost()'s database connection");
                        return -1;
                }
        }

}
```

RoomBookingBlock.java

```
package residency_project_files;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.InputMismatchException;
import java.util.Scanner;
```

```
public class RoomBookingBlock {
		RoomBookingBlock() throws SQLException{

	int totalMembers = 0,totalRooms = 0, roomsCount = DAO.getRoomsCount(),  costPerHour = DAO.gethourCost();
	Scanner in = new Scanner(System.in);

	Utils.blockTitle("ROOM BOOKING BLOCK");
	Utils.subTitle("room allocation");
	System.out.print("\tHOW MANY MEMBERS TO STAY : ");

	//
	try {
	   totalMembers = in.nextInt();
	}
	catch(InputMismatchException e) {
		System.err.println("! enter right members count");
		RoomBookingBlock rbb = new RoomBookingBlock();
	}


	if(totalMembers > 2){
	   if(totalMembers%2 == 0){
	      totalRooms = totalMembers / 2;
	   }
	   else{
	      totalRooms = (totalMembers / 2) + 1;
	   }
	}
	else{
	   totalRooms = 1;
	}
	String s = totalRooms == 1 ? " room" : " rooms";
	System.out.println("\t\t// you need minimum " + totalRooms + s);


	System.out.print("\tHOW MANY ROOMS IF YOU NEED : ");

	try {
	   totalRooms = in.nextInt();
	}
	catch(InputMismatchException e) {
		System.err.println("! enter right rooms count");
		RoomBookingBlock rbb = new RoomBookingBlock();
	}

	if((roomsCount-AllMethodsBlock.roomsCount()) >= totalRooms) {
		Utils.boxTitle("rooms are avilable.");
	}
	else {
		Utils.boxTitle("rooms are not avilable.");
		System.out.println("\t\t//only "+(roomsCount-AllMethodsBlock.roomsCount())+"rooms are avilable.");
		RoomBookingBlock rbb = new RoomBookingBlock();
	}

			int[] a = AllMethodsBlock.roomChooser(totalRooms);
			System.out.print("\nyour booked rooms : ");
			for(int b : a) {
				System.out.print(b+" ");
			}
			System.out.println("\n\nfloor view : ");
	AllMethodsBlock.printRooms('c', a);

	System.out.println("\n");



	String name,mailid;
	double days = 0.0;
	Utils.subTitle("for rigistration");
```

```
		System.out.print("\nENTER YOUR NAME\n\t: ");
		in.nextLine();
		name = in.nextLine();
		System.out.print("ENTER YOUR MAIL ID \t ! without mistake ! without space !\n\t: ");
		mailid = in.nextLine();
		System.out.print("ENTER HOW MANY DAYS YOU NEED TO STAY \t ! minimum helf day, enter 0.5 to helf day\n\t: ");


		try {
			days = in.nextDouble();
		}
		catch(InputMismatchException e) {
				System.err.println("! enter days rightly\n\t! one and half day means enter like 1.5");
				RoomBookingBlock rbb = new RoomBookingBlock();
		}


		System.out.println("\t// "+costPerHour+" rs per hour");
		double actualCost = 24*days*costPerHour*totalRooms;


		if(DAO.checkMail(mailid)) {
				ResultSet rst = DAO.getStatistics(mailid);
				rst.next();
				System.out.println("your old lived hours : "+rst.getInt(2));
					int totalCost = rst.getInt(2)*costPerHour;
				System.out.println("your payed cost per room : "+totalCost);
					double discount = ((double)10/100*totalCost*totalRooms);
					double costAfterDiscount = actualCost-discount;
				System.out.println("your actual payment is : "+actualCost+" but your our regular customer so we discount 10% of your old
payment on this payment.");
				System.out.println("your total discount amount is "+discount);
			System.out.println("press enter to pay after discount prise rs "+costAfterDiscount+" for your booking");
			in.nextLine();
			in.nextLine();

			DAO.setProfit((int)(DAO.getProfit()+costAfterDiscount));

		}
		else {
			System.out.println("press enter to pay rs "+actualCost+" for your booking");
			in.nextLine();
			in.nextLine();

			DAO.setProfit((int)(DAO.getProfit()+actualCost));

		}
		System.out.println("thank, you are paid successfully");
		DAO.booking(name, mailid, totalMembers, days,a);
	}
}
```

RoomVacate.java

```
package residency_project_files;

import java.sql.ResultSet;
import java.sql.SQLException;
//import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.InputMismatchException;
import java.util.Scanner;

public class RoomVacate {
		RoomVacate() throws SQLException {
```

```
                        int roomId = 0, costPerHour = DAO.gethourCost();
                        Scanner in = new Scanner(System.in);
//      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");

                        Utils.blockTitle("ROOM VACATING BLOCK");
                        System.out.print("you can vacate rooms using your room id : ");
/*                      System.out.println("enter 1 for id : ");
                        System.out.println("enter 2 for mail id : ");
    System.out.print("\nenter your value here : ");
*/


    try {
      roomId = in.nextInt();
    }
    catch(InputMismatchException e) {
            System.err.println("! enter your id rightly");
            RoomVacate rv = new RoomVacate();
    }


    ResultSet rst = DAO.getmail(roomId);

    if(!rst.next()) {                   //check id is valid or not
            System.err.println("! enter right id");
            RoomVacate rv = new RoomVacate();
    }

    String mail = rst.getString(1);

    ResultSet rst2 = DAO.getCustomer(mail);
    rst2.next();
    int totalRooms = DAO.getRoomsCount(roomId);
    double days = rst2.getDouble(6);
    double oldCost = 24*days*costPerHour*totalRooms;                    // old cost without discount

    Date dNow = new Date();
    Date dOld = rst2.getTimestamp(5);
                        long diff = dNow.getTime() - dOld.getTime();
                        long diffhours = diff / (60*60*1000);
                        double newCost = diffhours*costPerHour*totalRooms;                  // new cost without discount

    if(DAO.checkMail(mail)) {
            ResultSet rst1 = DAO.getStatistics(mail);
            rst1.next();
                        int totalCost = rst1.getInt(2)*costPerHour;
                        double discount = (double)10/100*totalCost;
                        System.out.println("hours you spand : "+diffhours);
            System.out.println("your payed cost : "+(oldCost-discount));
            System.out.println("your new cost : "+(newCost+discount));
            if((oldCost-discount) == (newCost+discount)) {
                        System.out.println("thank you");
            }
            else if((oldCost-discount) < (newCost+discount)) {
                        System.out.println("press enter to pay balence : "+((newCost+discount)-(oldCost-discount)));
        in.nextLine();
        in.nextLine();

        DAO.setProfit((int)(DAO.getProfit()+((newCost+discount)-(oldCost-discount))));

                        System.out.println("thank you");
            }
            else {
                        System.out.println("press enter to get balence : "+((oldCost-discount) - (newCost+discount)));
        in.nextLine();
        in.nextLine();

        DAO.setProfit((int)(DAO.getProfit()-((oldCost-discount)-(newCost+discount))));

                        System.out.println("thank you");
```

```java
            }
            int repeatation = rst1.getInt(3);
            DAO.statisticsUpdate(mail, repeatation+1, (int)(diffhours));
        }
        else {
            DAO.statisticsInsert(mail, (int)diffhours);
                    System.out.println("hours you spand : "+diffhours);
            System.out.println("your payed cost : "+oldCost);
            System.out.println("your new cost : "+newCost);
            if(oldCost == newCost) {
                    System.out.println("thank you");
            }
            else if(oldCost < newCost) {
                    System.out.println("press enter to pay balence : "+(newCost-oldCost));
          in.nextLine();
          in.nextLine();

          DAO.setProfit((int)(DAO.getProfit()+((newCost)-(oldCost))));

                    System.out.println("thank you");
            }
            else {
                    System.out.println("press enter to get balence"+(oldCost-newCost));
          in.nextLine();
          in.nextLine();

          DAO.setProfit((int)(DAO.getProfit()-((oldCost)-(newCost))));

                    System.out.println("thank you");
            }
        }
        DAO.vacate(roomId);
            }
}
```

Utils.java

```java
package residency_project_files;

public class Utils {

            //------------------------------------------------------------------------------------------

   static void starTitle(char i){

       System.out.println("--------------------------------------------------------------------------------------------------------------------------");
       System.out.printf("%c%c%c%c%c  %c%c%c%c%c%c %c%c%c%c%c%c %c%c %c%c%c%c%c  %c%c%c%c%c%c %c    %c
%c%c%c%c%c %c    %c    %c%c%c%c%c%c%c    %c%c    %c       %c %c%c%c%c%c%c %c%c%c%c%c
\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c   %c  %c     %c       %c   %c   %c     %c %c    %c  %c     %c    %c        %c     %c    %c   %c    %c   %c  %c
%c   %c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c   %c  %c     %c     %c%c  %c    %c  %c    %c  %c   %c  %c     %c  %c        %c     %c     %c  %c   %c %c   %c
%c     %c   %c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c%c%c%c%c  %c%c%c%c%c%c %c%c%c%c%c%c %c%c %c    %c %c%c%c%c%c%c %c  %c  %c %c       %c %c
%c    %c       %c  %c   %c   %c   %c %c%c%c%c%c%c %c%c%c%c%c
\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c %c   %c        %c %c%c %c    %c  %c     %c   %c %c %c        %c          %c     %c      %c  %c %c    %c %c
%c     %c %c \n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c  %c   %c        %c %c%c %c   %c   %c     %c    %c %c %c       %c          %c     %c     %c  %c %c     %c %c
%c     %c  %c \n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.printf("%c   %c %c%c%c%c%c%c %c%c%c%c%c%c %c%c %c%c%c%c%c   %c%c%c%c%c%c %c     %c  %c%c%c%c%c
%c           %c       %c%c     %c        %c %c%c%c%c%c%c %c
%c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i);
       System.out.println("--------------------------------------------------------------------------------------------------------------------------");
```

```java
    // * -> 42 # 35, $ 36, % 37, & 38, @ 64, ^ 94, ~ 126, £ -> 163, ¥ -> 165, Ö -> 214 , Ü -> 220, ß -> 223 , ö -> 246
    }
            //-----------------------------------------------------------------------------------------

    static void blockTitle(String str){
        System.out.print("\n--------------------------------------------------------------------------------------\n|
|\n|");
        int totalLength = 88;
        int firstLength = (totalLength/2) - (str.length()/2);
        int lastLength = totalLength - (firstLength+str.length());
        for (int i = 0; i < firstLength; i++) {
            System.out.print(' ');
        }
        System.out.print(str.toUpperCase());
        for (int i = 0; i < lastLength; i++) {
            System.out.print(' ');
        }
        System.out.println("|\n|                                                                    |\n--------------------------------------------------------------------------
---------------\n");
    }
            //-----------------------------------------------------------------------------------------

    static void subTitle(String str){
        System.out.println(str.toUpperCase());
        for (int i = 0; i < str.length(); i++) {
            System.out.print('-');
        }
        System.out.println("\n");
    }
            //-----------------------------------------------------------------------------------------
    static void boxTitle(String str) {
            System.out.print("\t");
            for(int i = 0; i < str.length()+4; i++) {
                    System.out.print('-');
            }
            System.out.print("\n\t| "+str+" |\n\t");
            for(int i = 0; i < str.length()+4; i++) {
                    System.out.print('-');
            }
            System.out.println();
    }
            //-----------------------------------------------------------------------------------------
            //-----------------------------------------------------------------------------------------
            //-----------------------------------------------------------------------------------------
    static String starTitleTXT(char i){
            String totalStr;
        totalStr = "--------------------------------------------------------------------------------------------------------------------------\n";
        totalStr = totalStr.concat(String.format("%c%c%c%c%c  %c%c%c%c%c%c  %c%c%c%c%c%c  %c%c  %c%c%c%c%c   %c%c%c%c%c%c
%c    %c  %c%c%c%c%c  %c     %c      %c%c%c%c%c%c%c     %c%c     %c        %c  %c%c%c%c%c%c%c  %c%c%c%c%c%c
\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c   %c  %c     %c        %c   %c  %c     %c %c    %c  %c       %c    %c         %c     %c    %c
%c    %c    %c  %c     %c   %c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c   %c  %c     %c     %c%c  %c    %c  %c     %c  %c   %c  %c       %c  %c          %c     %c
%c  %c   %c %c   %c  %c      %c   %c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c%c%c%c%c  %c%c%c%c%c%c  %c%c%c%c%c%c  %c%c  %c    %c  %c%c%c%c%c%c%c  %c   %c
%c  %c       %c %c          %c    %c       %c  %c   %c   %c   %c  %c%c%c%c%c%c%c  %c%c%c%c%c%c
\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c  %c   %c        %c  %c%c  %c    %c  %c      %c   %c  %c  %c         %c            %c     %c
%c   %c  %c    %c  %c  %c      %c  %c  \n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c   %c   %c        %c  %c%c  %c    %c   %c      %c    %c %c  %c        %c            %c      %c     %c
%c %c     %c %c  %c      %c   %c \n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat(String.format("%c   %c  %c%c%c%c%c%c  %c%c%c%c%c%c  %c%c  %c%c%c%c%c   %c%c%c%c%c%c  %c
%c  %c%c%c%c%c   %c             %c        %c%c      %c        %c  %c%c%c%c%c%c%c  %c
%c\n",i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i,i));
        totalStr = totalStr.concat("--------------------------------------------------------------------------------------------------------------------------\n");
        return totalStr;
    }
            //-----------------------------------------------------------------------------------------

    static String blockTitleTXT(String str){
```

```
        String totalStr = "\n-----------------------------------------------------------------------------------------\n|
|\n|";
        int totalLength = 88;
        int firstLength = (totalLength/2) - (str.length()/2);
        int lastLength = totalLength - (firstLength+str.length());
        for (int i = 0; i < firstLength; i++) {
                totalStr = totalStr.concat(" ");
//          System.out.print(' ');
        }
        totalStr = totalStr.concat(str.toUpperCase());
        for (int i = 0; i < lastLength; i++) {
                totalStr = totalStr.concat(" ");
//          System.out.print(' ');
        }
        totalStr = totalStr.concat("|\n|                                                                                    |\n-----------------------------------------------------------------------
----------\n");
        return totalStr;
    }
                //-----------------------------------------------------------------------------------------

    static String subTitleTXT(String str){
        String totalStr = str.toUpperCase();
        totalStr = totalStr.concat("\n");
        for (int i = 0; i < str.length(); i++) {
          totalStr = totalStr.concat("-");
        }
        totalStr = totalStr.concat("\n");
        return totalStr;
    }
                //-----------------------------------------------------------------------------------------
    static String boxTitleTXT(String str) {
                String totalStr = "\t";
                for(int i = 0; i < str.length()+4; i++) {
                        totalStr = totalStr.concat("-");
                }
                totalStr = totalStr.concat("\n\t| "+str+" |\n\t");
                for(int i = 0; i < str.length()+4; i++) {
                        totalStr = totalStr.concat("-");
                }
                totalStr = totalStr.concat("\n");
                return totalStr;
    }
                //-----------------------------------------------------------------------------------------
}
```

MySQL source code:

create table hotel settings(hotel_id int default 1 not null, rooms_count int default 50 not null, hour_cost int default 50 not null, modify_state boolean default true not null, profit int default 0 not null);

insert into hotel_settings values (1, 50, 50, true, 0);

create table statistics (mail_id varchar(200) primary key, total_time int, repeatation int default 0);

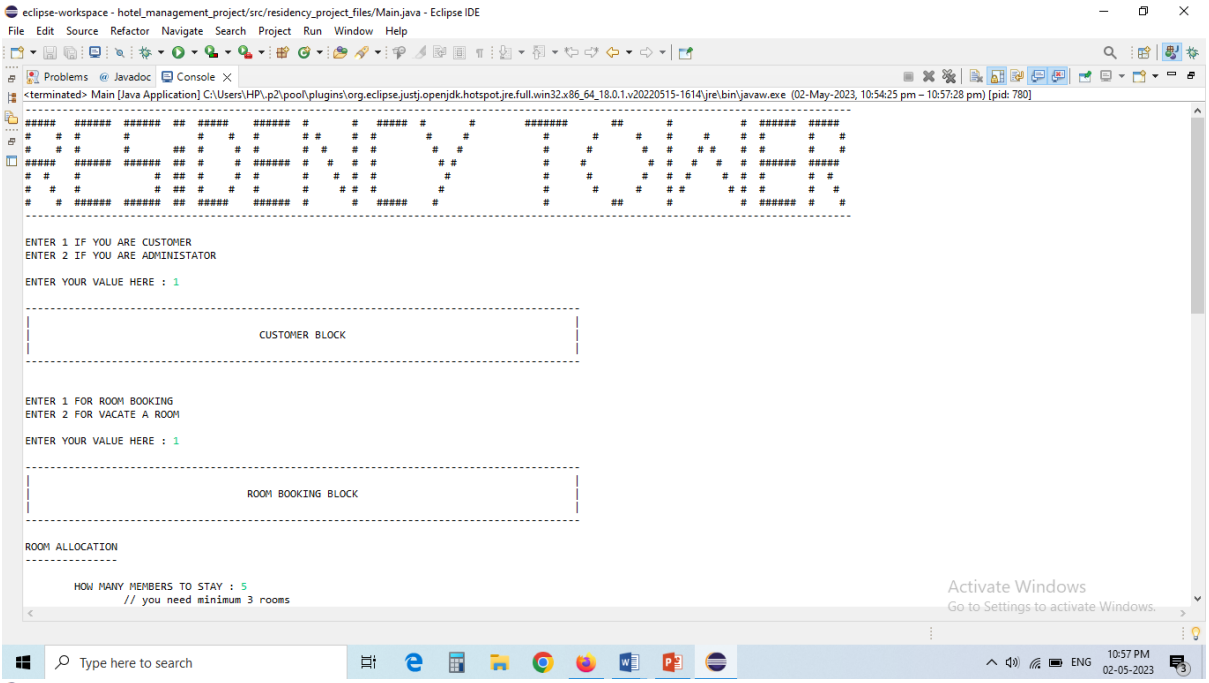
create table rooms (room_no int primary key, customer_id int not null, foreign key(customer_id) references customers(id) on delete cascade);

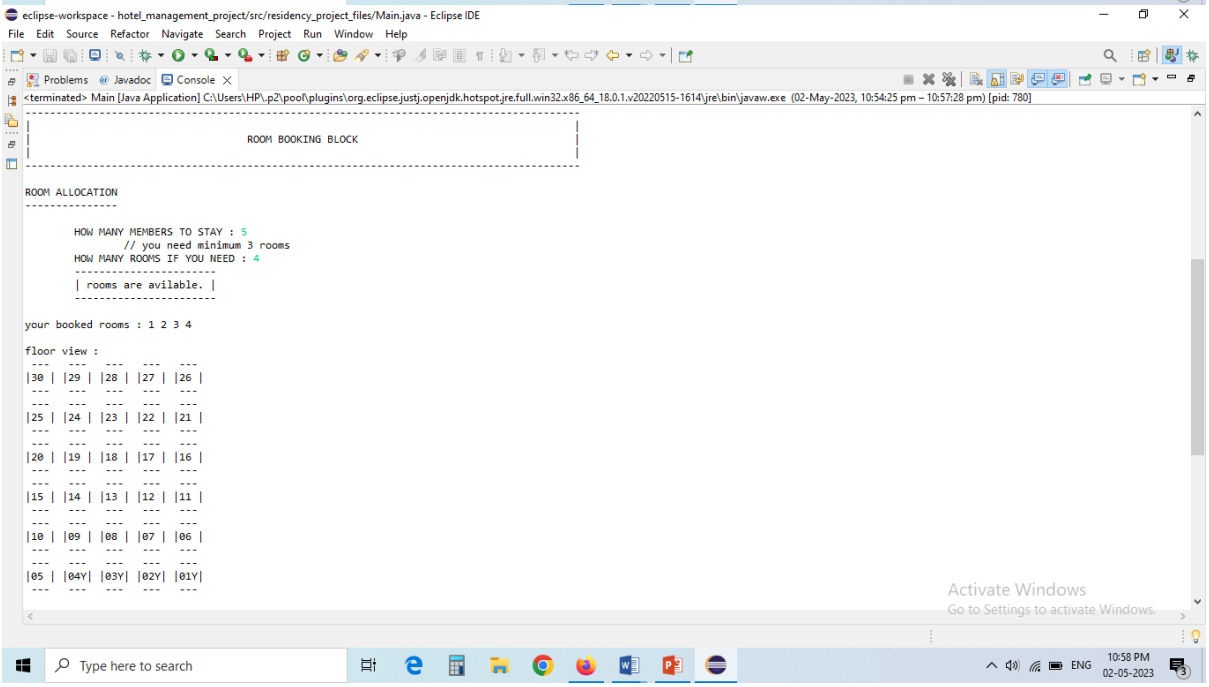
create table customers (id int primary key, name varchar(100) not null, mail_id varchar(200) unique key, members int not null, checkin timestamp not null, time_by_customer int not null);

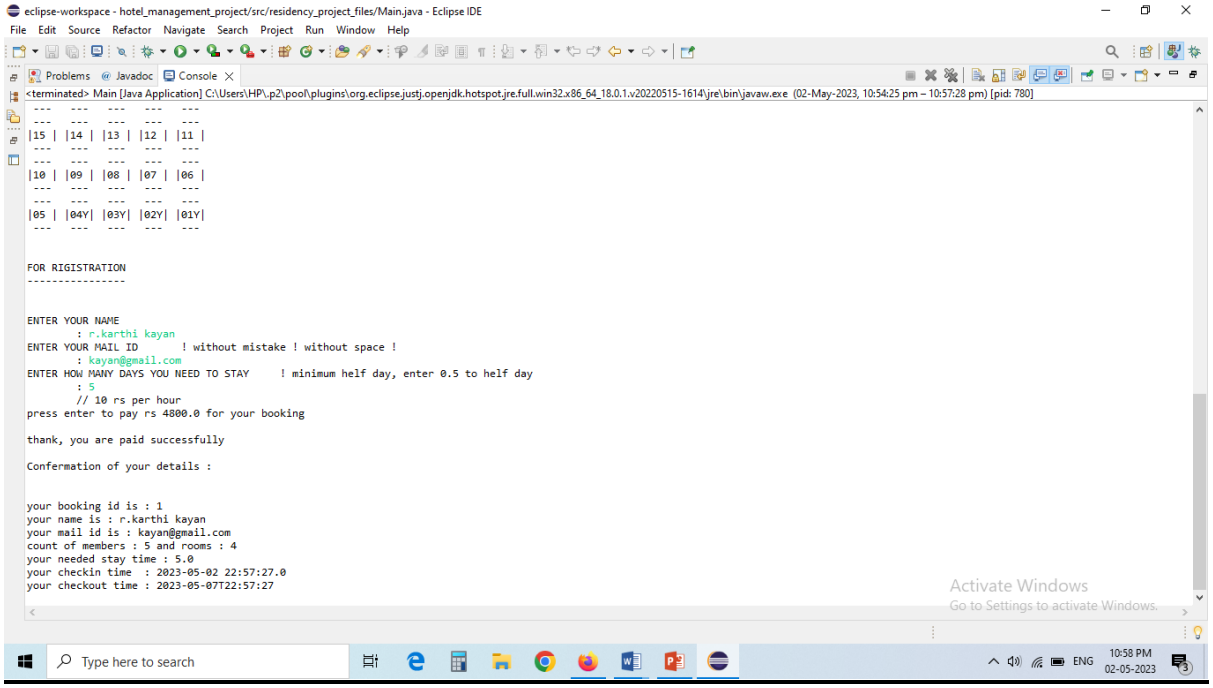
CHAPTER 6

SCREENSHOTS

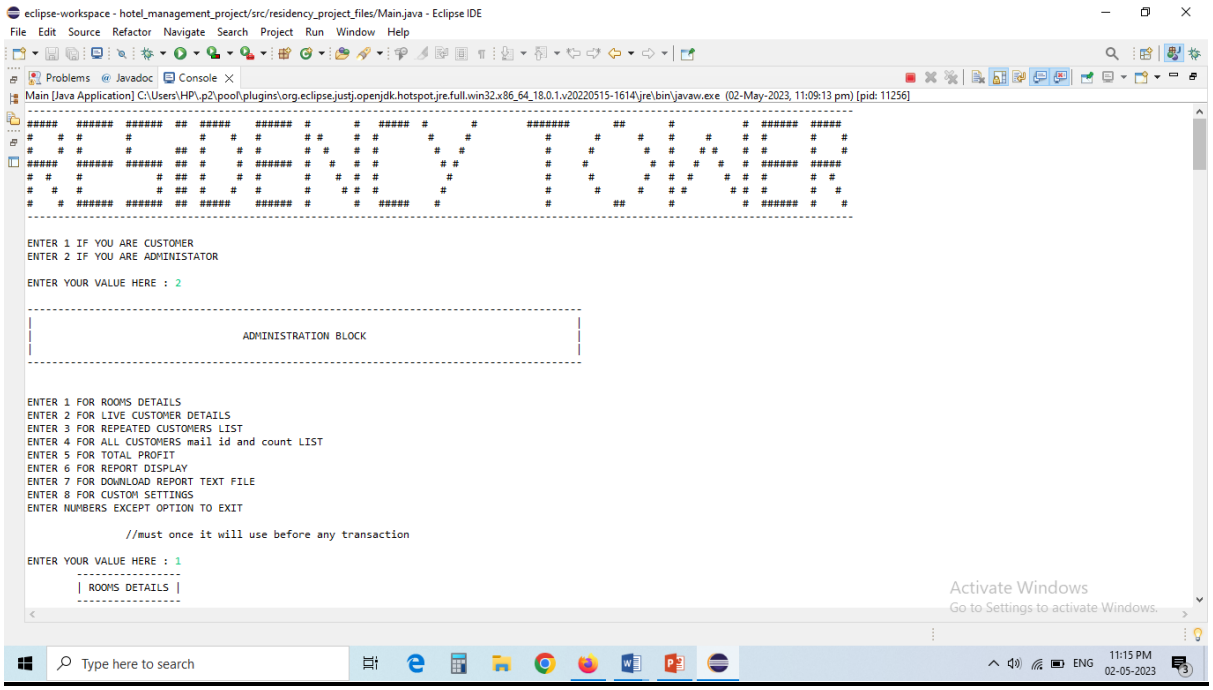
Customer block:

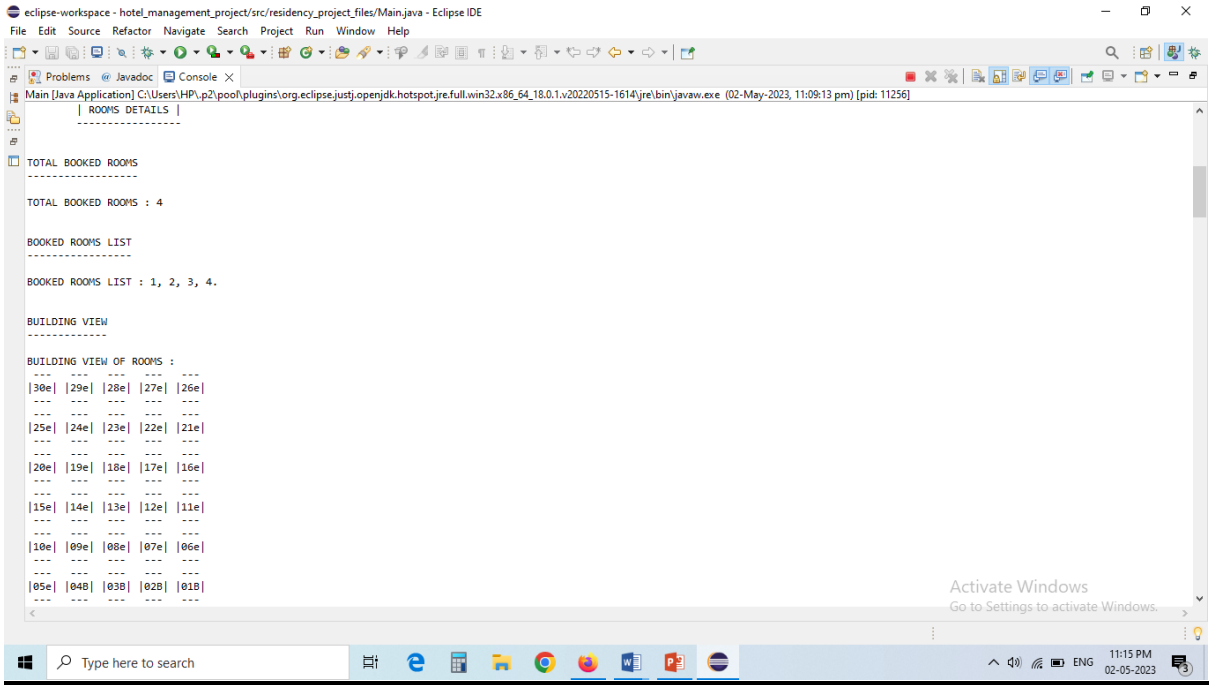


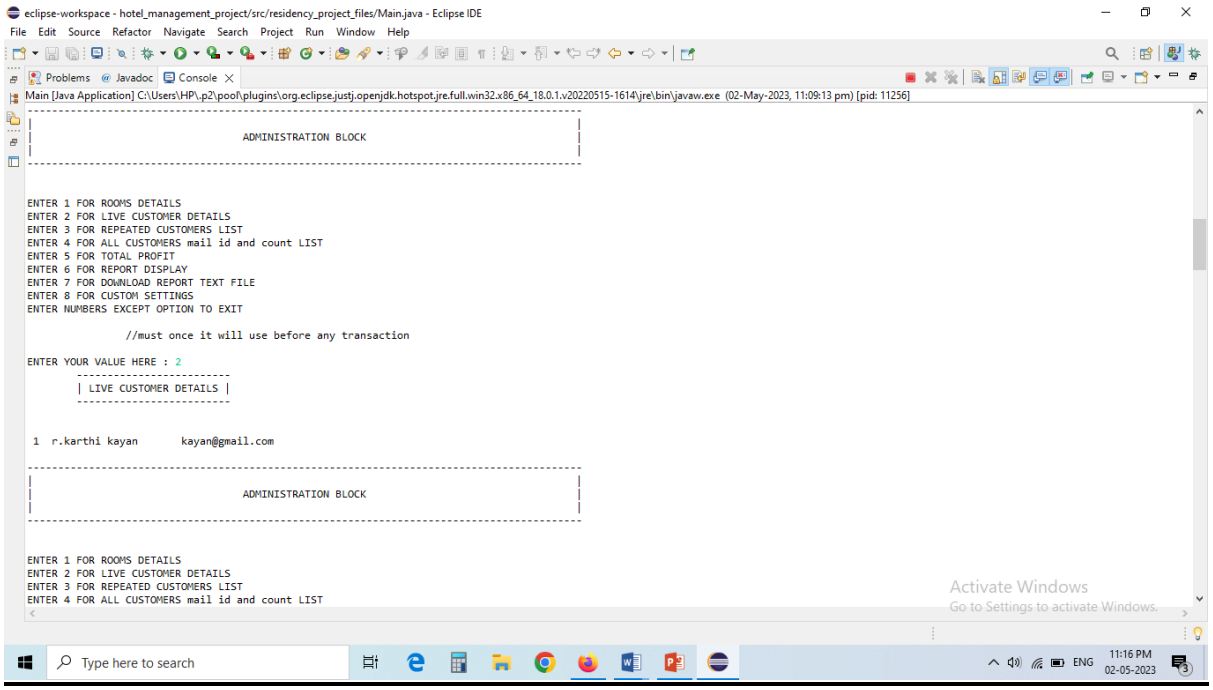


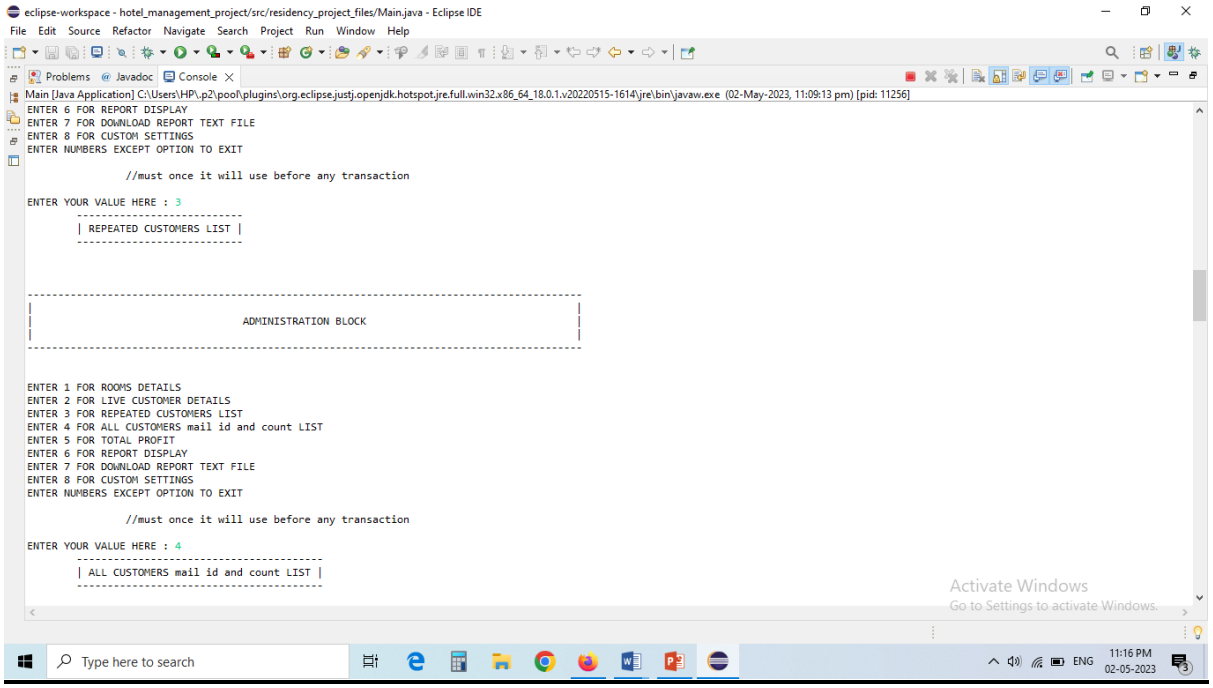


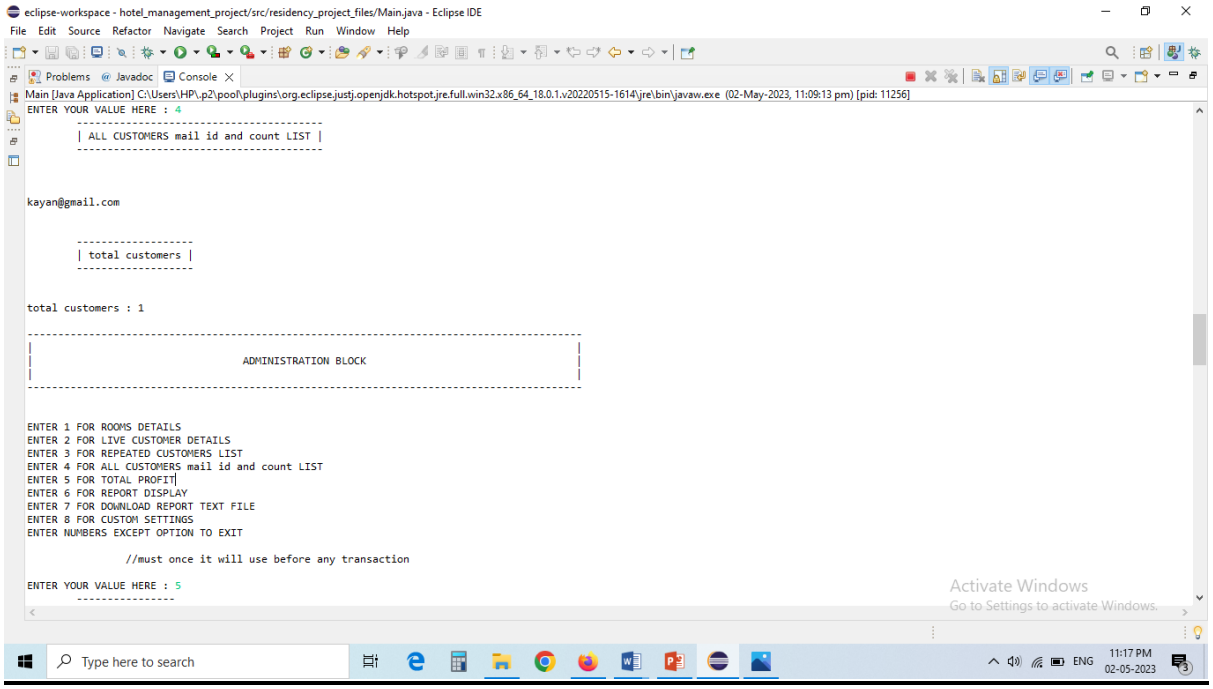
Administrator block:

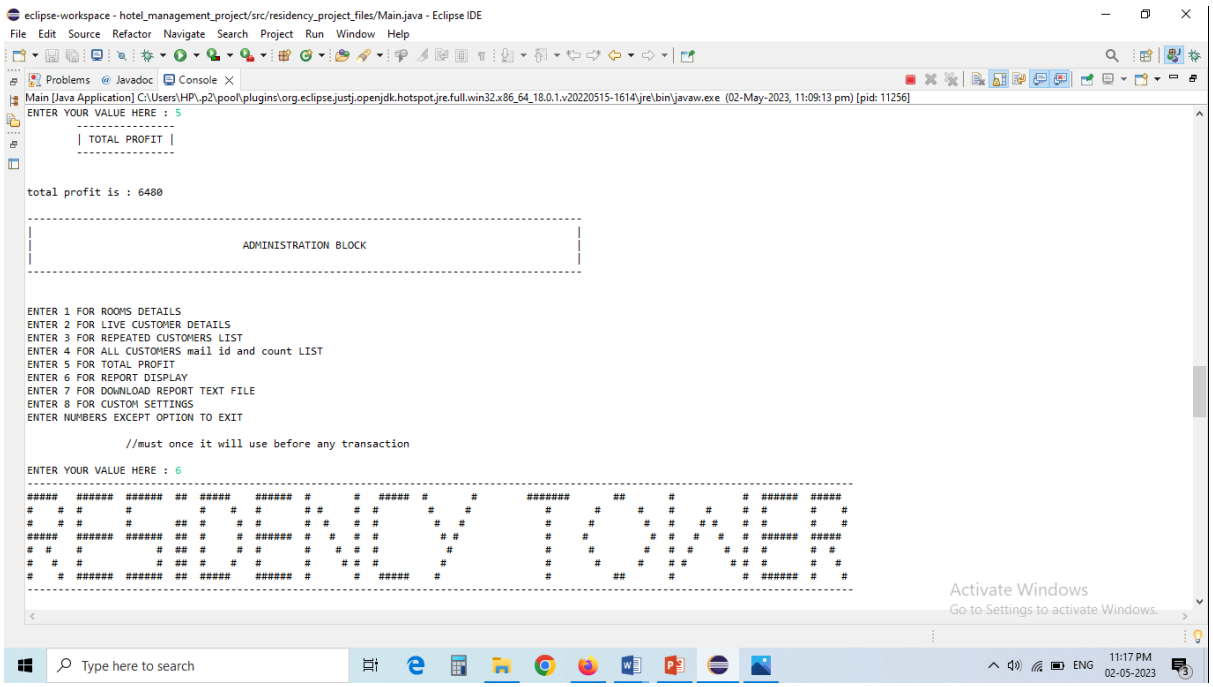



eclipse-workspace - hotel_management_project/src/residency_project_files/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Console
Main [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_18.0.1.v20220515-1614\jre\bin\javaw.exe (02-May-2023, 11:09:13 pm) [pid: 11256]
| ROOMS DETAILS |
TOTAL BOOKED ROOMS
TOTAL BOOKED ROOMS : 4
BOOKED ROOMS LIST
BOOKED ROOMS LIST : 1, 2, 3, 4.
BUILDING VIEW
BUILDING VIEW OF ROOMS :
|30e| |29e| |28e| |27e| |26e|
|25e| |24e| |23e| |22e| |21e|
|20e| |19e| |18e| |17e| |16e|
|15e| |14e| |13e| |12e| |11e|
|10e| |09e| |08e| |07e| |06e|
|05e| |04B| |03B| |02B| |01B|
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
11:15 PM
02-05-2023


eclipse-workspace - hotel_management_project/src/residency_project_files/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Console
Main [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_18.0.1.v20220515-1614\jre\bin\javaw.exe (02-May-2023, 11:09:13 pm) [pid: 11256]
ADMINISTRATION BLOCK
ENTER 1 FOR ROOMS DETAILS
ENTER 2 FOR LIVE CUSTOMER DETAILS
ENTER 3 FOR REPEATED CUSTOMERS LIST
ENTER 4 FOR ALL CUSTOMERS mail id and count LIST
ENTER 5 FOR TOTAL PROFIT
ENTER 6 FOR REPORT DISPLAY
ENTER 7 FOR DOWNLOAD REPORT TEXT FILE
ENTER 8 FOR CUSTOM SETTINGS
ENTER NUMBERS EXCEPT OPTION TO EXIT
//must once it will use before any transaction
ENTER YOUR VALUE HERE : 2
| LIVE CUSTOMER DETAILS |
1 r.karthi kayan kayan@gmail.com
ADMINISTRATION BLOCK
ENTER 1 FOR ROOMS DETAILS
ENTER 2 FOR LIVE CUSTOMER DETAILS
ENTER 3 FOR REPEATED CUSTOMERS LIST
ENTER 4 FOR ALL CUSTOMERS mail id and count LIST
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
11:16 PM
02-05-2023


eclipse-workspace - hotel_management_project/src/residency_project_files/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Console
Main [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_18.0.1.v20220515-1614\jre\bin\javaw.exe (02-May-2023, 11:09:13 pm) [pid: 11256]
ENTER 6 FOR REPORT DISPLAY
ENTER 7 FOR DOWNLOAD REPORT TEXT FILE
ENTER 8 FOR CUSTOM SETTINGS
ENTER NUMBERS EXCEPT OPTION TO EXIT
//must once it will use before any transaction
ENTER YOUR VALUE HERE : 3
| REPEATED CUSTOMERS LIST |
ADMINISTRATION BLOCK
ENTER 1 FOR ROOMS DETAILS
ENTER 2 FOR LIVE CUSTOMER DETAILS
ENTER 3 FOR REPEATED CUSTOMERS LIST
ENTER 4 FOR ALL CUSTOMERS mail id and count LIST
ENTER 5 FOR TOTAL PROFIT
ENTER 6 FOR REPORT DISPLAY
ENTER 7 FOR DOWNLOAD REPORT TEXT FILE
ENTER 8 FOR CUSTOM SETTINGS
ENTER NUMBERS EXCEPT OPTION TO EXIT
//must once it will use before any transaction
ENTER YOUR VALUE HERE : 4
| ALL CUSTOMERS mail id and count LIST |
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
11:16 PM
02-05-2023


eclipse-workspace - hotel_management_project/src/residency_project_files/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Console
Main [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_18.0.1.v20220515-1614\jre\bin\javaw.exe (02-May-2023, 11:09:13 pm) [pid: 11256]
ENTER YOUR VALUE HERE : 4
| ALL CUSTOMERS mail id and count LIST |
kayan@gmail.com
| total customers |
total customers : 1
ADMINISTRATION BLOCK
ENTER 1 FOR ROOMS DETAILS
ENTER 2 FOR LIVE CUSTOMER DETAILS
ENTER 3 FOR REPEATED CUSTOMERS LIST
ENTER 4 FOR ALL CUSTOMERS mail id and count LIST
ENTER 5 FOR TOTAL PROFIT
ENTER 6 FOR REPORT DISPLAY
ENTER 7 FOR DOWNLOAD REPORT TEXT FILE
ENTER 8 FOR CUSTOM SETTINGS
ENTER NUMBERS EXCEPT OPTION TO EXIT
//must once it will use before any transaction
ENTER YOUR VALUE HERE : 5
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
11:17 PM
02-05-2023



CHAPTER 7

CONCLUSION & FUTURE SCOPE

7.1 CONCLUSION

A hotel management system is a crucial tool for hotels to streamline their operations, improve guest experience, and increase revenue. The system can automate many tasks, such as reservations, check-in and check-out, inventory management, and billing, reducing the workload on hotel staff and minimizing errors. It can also provide valuable insights into hotel performance, such as occupancy rates, revenue, and guest feedback, helping hotels make data-driven decisions.

Hotel management systems have evolved over the years, with newer systems incorporating advanced features such as mobile apps, contactless services, and personalization. The future of hotel management systems is promising, with further integration with other hospitality and travel systems, as well as increased focus on sustainability and environmental impact.

7.2 FUTURE SCOPE

The future scope of hotel management systems is promising and will continue to evolve with advances in technology and changing customer demands. Here are some possible future directions for hotel management systems:

1. Personalization: As hotels seek to provide more personalized experiences to their guests, hotel management systems may incorporate machine learning and AI to analyse guest data and preferences. This could lead to more tailored recommendations, personalized marketing, and customized services.
2. Mobile-first approach: With more guests using mobile devices to book hotels and manage their reservations, hotel management systems may prioritize mobile interfaces and incorporate features like mobile check-in and keyless entry.
3. Contactless services: The COVID-19 pandemic has accelerated the adoption of contactless technologies in the hospitality industry. Hotel management systems may incorporate more contactless features like online payments, digital menus, and virtual concierge services.
4. Integration with other systems: Hotel management systems may increasingly integrate with other hospitality and travel systems, such as airline reservation systems, travel booking platforms, and ride-sharing apps, to provide a seamless end-to-end travel experience for guests.

CHAPTER 8

REFERENCES

1. Oracle Corporation, “Commercial License for OEM, ISVs and VARs”. [www-document] available at: http://www.mysql.com/about/legal/licensing/oem/ Retrieved 07.2010

2. The MySQL Benchmark Suite. [www-document]. Available at: http://dev.mysql.com/doc/refman/5.1/en/mysql-benchmarks.html. Retrieved 2011

3. http://www.w3schools.com/sql/sql_insert.asp

4. http://www.w3schools.com/sql/sql_update.asp

5. Fundamentals of software engineering by Rajib mall, PHIlearning