HOTEL MANAGEMENT SYSTEM

A JAVA PROJECT BY-

Vivan Jain (16010122071)

Kunj Gohil (16010122060)

Eeshanya Joshi (16010122074)

INTRODUCTION

"Modern Hotel Management System" – a powerful and efficient software solution designed to streamline and optimize the operations of a hotel. In today's dynamic hospitality industry, managing a hotel's daily tasks, guest reservations, and resources efficiently is critical to delivering exceptional customer experiences and maintaining a competitive edge.

Our project represents a comprehensive and user-friendly hotel management system developed in Java, which offers a wide range of features and functionalities tailored to meet the specific needs of hotels and their guests.

PROJECT OBJECTIVES

1. Create a menu-driven program for hotel management.
2. Implement a system to store and manage customer details.
3. Allow customers to book rooms of different types and handle unbooking
4. Enable customers to place food orders.
5. Generate bills for customers based on their room bookings and food orders.
6. Provide information on room availability and features.
7. Use file handling to store and retrieve data, ensuring data persistence.
8. Implement user-defined exceptions to handle specific error scenario appropriately.

Problem Statement

Design and implement a comprehensive hotel management system that allows hotel staff to efficiently manage various aspects of their business. The system should provide functionalities for managing customer information, booking and unbooking rooms of different types, ordering food, and generating bills. It should also allow staff to check room availability and view room features. The system should use file handling to persist data across sessions, ensuring that customer details, booked rooms, and food orders are retained even after the program is restarted. Implement proper exception handling to deal with unexpected errors and ensure smooth operation.

OOPM FEATURES USED

1. **INHERITANCE:**

In Java, inheritance is a fundamental concept of object-oriented programming that enables the creation of a hierarchy of classes. It allows one class (the subclass or child class) to inherit attributes and behaviors from another class (the superclass or parent class). This mechanism encourages code reuse and promotes a structured approach to designing software. Subclasses can access and utilize the attributes and methods of the superclass, helping to avoid redundancy and simplify code maintenance.

2. POLYMORPHISM:

Polymorphism in Java is a powerful object-oriented programming concept that allows different classes to be treated as instances of a common superclass. It enables flexibility and extensibility in code by allowing objects of diverse types to be used interchangeably, as long as they adhere to a shared interface or inheritance hierarchy. Polymorphism is achieved through method overriding and interface implementation, and it enables dynamic method dispatch, where the actual method to be executed is determined at runtime based on the specific object's type.

3. Abstraction

Abstraction in programming is a fundamental concept that allows developers to simplify complex systems by focusing on essential features while hiding intricate implementation details. It involves creating a clear separation between what an object does and how it accomplishes those tasks. In Java, abstraction is often realized through abstract classes and interfaces, which define a set of methods that must be implemented by concrete classes. By abstracting away implementation specifics, abstraction makes code more modular and understandable.

4. Packages

Packages in Java are a powerful organizational mechanism that help structure and manage your codebase. They serve as containers for related classes, interfaces, and other code elements, enabling you to group and categorize components in a logical and hierarchical manner. By using packages, you can prevent naming conflicts, as each package provides its own namespace. This not only ensures code clarity but also makes it easier to locate and reuse specific pieces of code.

5. Exception Handling

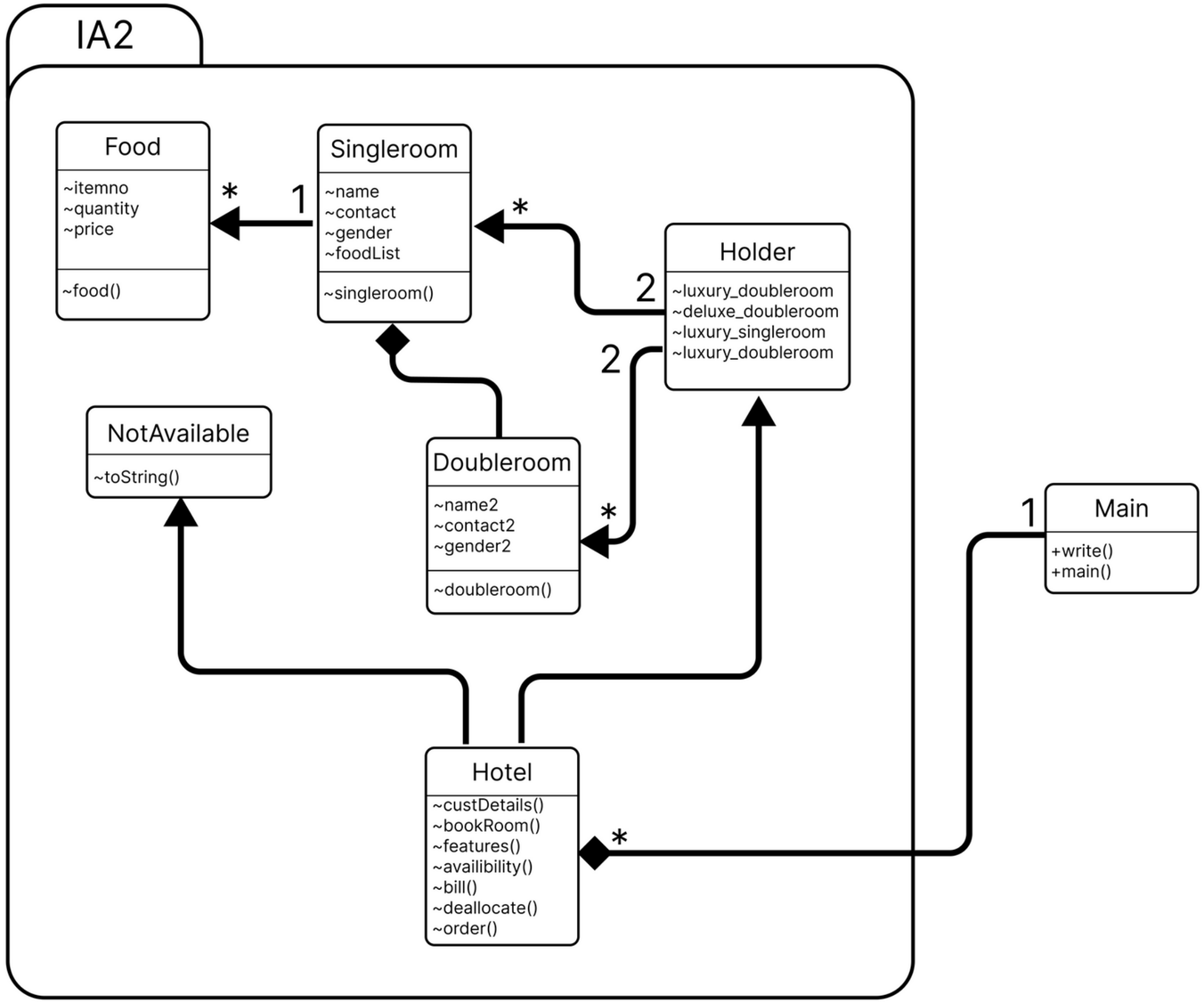
Exception handling in Java is a vital mechanism for managing unexpected runtime errors and exceptional conditions that may disrupt the normal flow of a program. It allows developers to anticipate, capture, and gracefully handle these errors, preventing abrupt program termination. Java's exception handling is structured around a set of keywords and constructs, including ‘ try ’, ‘ catch ’, ‘ throw ’, and ‘ finally ’. It enables the separation of error-handling code from the main program logic, promoting code clarity and maintainability.

Advantages of Hotel Management System

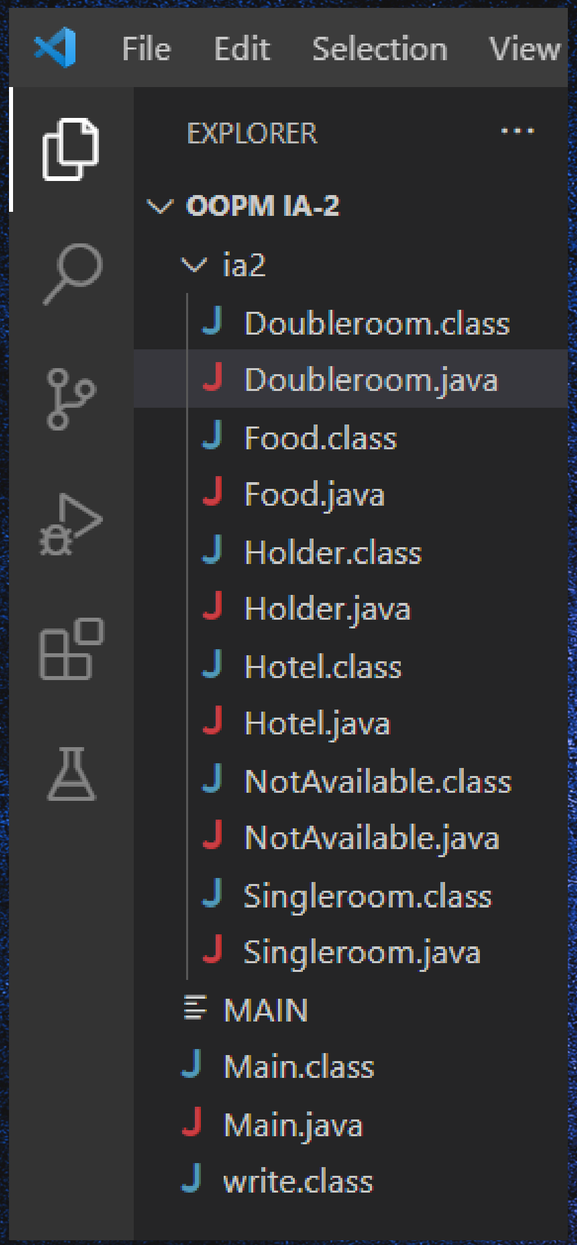
A Hotel Management System offers several advantages, both for hotel owners and guests. Here are some of the key benefits of implementing a Hotel Management System:

1. Efficient Reservation Management: Hotel Management system streamlines the booking process, allowing guests to make reservations online or through a hotel's front desk. It helps prevent overbooking and double-booking errors, ensuring optimal room utilization.
2. Improved Guest Experience: Guests can enjoy a seamless check-in and check-out process, as well as personalized services. HMS stores guest preferences, making it easier to provide a comfortable and customized stay.
3. Real-time Room Availability: Hotel staff can access real-time information about room availability, helping them allocate rooms efficiently. This reduces guest waiting times and enhances overall satisfaction.

CLASS DIAGRAM



PACKAGE DIRECTORY



ia2 > Doubleroom.java > Doubleroom

```java
package ia2;
public class Doubleroom extends Singleroom
{
    public String name2;
    public String contact2;
    public String gender2;

    public Doubleroom()
    {
        this.name="";
        this.name2="";
    }

    public Doubleroom(String name,String contact,
    String gender,String name2,String contact2,String gender2)
    {
        this.name=name;
        this.contact=contact;
        this.gender=gender;
        this.name2=name2;
        this.contact2=contact2;
        this.gender2=gender2;
    }
}
```

CODE SCREENSHOTS

ia2 > Food.java > Food

```java
package ia2;

public class Food{
    public int itemno;
    public int quantity;
    public float price;

    public Food(int itemno,int quantity)
    {
        this.itemno=itemno;
        this.quantity=quantity;

        switch(itemno)
        {
            case 1:price=quantity*50;
            break;
            case 2:price=quantity*60;
            break;
            case 3:price=quantity*70;
            break;
            case 4:price=quantity*30;
            break;
        }
    }
}
```

ia2 > Singleroom.java > Singleroom

```java
package ia2;

import java.util.ArrayList;
public class Singleroom
{
    public String name;
    public String contact;
    public String gender;
    public ArrayList<Food>
    food=new ArrayList<>();

    public Singleroom()
    {
        this.name="";
    }

    public Singleroom (String name,
    String contact,String gender)
    {
        this.name=name;
        this.contact=contact;
        this.gender=gender;
    }
}
```

ia2 > Holder.java > Holder

```java
package ia2;
public class Holder
{
    public Doubleroom luxury_doublerrom[]=new Doubleroom[10]; //Luxury
    public Doubleroom deluxe_doublerrom[]=new Doubleroom[20]; //Deluxe
    public Singleroom luxury_singleerrom[]=new Singleroom[10]; //Luxury
    public Singleroom deluxe_singleerrom[]=new Singleroom[20]; //Deluxe
}
```

ia2 > NotAvailable.java > NotAvailable

```java
package ia2;

public class NotAvailable extends Exception
{
    @Override
    public String toString()
    {
        return "Not Available !";
    }
}
```

HOTEL . JAVA

Hotel.java

ia2 > Hotel.java > {} ia2

```java
package ia2;
import java.util.*;
public class Hotel
{
public static Holder hotel_ob=new Holder();
public static Scanner sc = new Scanner(System.in);
public static void CustDetails(int i,int rn)
{
String name, contact, gender;
String name2 = null, contact2 = null;
String gender2="";
System.out.print(s:"InEnter customer name: ");
name=sc.next();
System.out.print(s:"Enter contact number: ");
contact=sc.next();
System.out.print(s:"Enter gender: ");
gender=sc.next();
if(i<3)
{
System.out.print(s:"Enter second customer name: ");
name2=sc.next();System.out.print(s:"Enter contact number: ");
contact2=sc.next();
System.out.print(s:"Enter gender: ");
gender2 =sc.next();
}
switch (i)
{
    case 1:hotel_ob.luxury_doublerrom[rn]=new Doubleroom(name, contact, gender, name2,contact2, gender2);
break;
case 2:hotel_ob.deluxe_doublerrom[rn]=new Doubleroom(name, contact,gender,name2,contact2,gender2);
break;
case 3:hotel_ob. luxury_singleerrom[rn]=new Singleroom(name, contact,gender);
break;
case 4:hotel_ob.deluxe_singleerrom[rn]=new Singleroom(name, contact,gender);
break;
default:System.out.println(x:"Wrong option");
break;
 }
}
public static void bookroom(int i)
{
    int j;
    int rn;
    System.out.println(x:"\nChoose room number from : ");
    switch (i)
    {
        case 1:
            for(j=0;j<hotel_ob.luxury_doublerrom.length;j++)
            {
                if(hotel_ob.luxury_doublerrom[j]==null)
                {
                    System.out.print(j+1+",");
                }
            }
            System.out.print(s:"\nEnter room number: ");
            try
            {
                rn=sc.nextInt();
                rn--;
                if(hotel_ob.luxury_doublerrom[rn]!=null)
                throw new NotAvailable();
                CustDetails(i,rn);
            }
            catch(Exception e)
            {
                System.out.println(x:"Invalid Option");
                return;
            }
            break;
        case 2:
            for(j=0;j<hotel_ob.deluxe_doublerrom.length;j++)
            {
                if(hotel_ob.deluxe_doublerrom[j]==null)
                {
```

```
                    System.out.print(j+11+",");
                }
            }
            System.out.print(s:"InEnter room number: ");
            try
            {
                rn=sc.nextInt();
                rn=rn-11;
                if(hotel_ob.deluxe_doublerrom[rn]!=null)
                throw new NotAvailable();
                CustDetails(i,rn);
            }
            catch(Exception e)
            {
                System.out.println(x:"Invalid Option");
                return;
            }
            break;
        case 3:
            for(j=0;j<hotel_ob.luxury_singleerrom.length;j++)
            {
                if(hotel_ob. luxury_singleerrom[j]==null)
                {
                    System.out.print(j+31+",");
                }
            }
            System.out.print(s:"InEnter room number:");
            try
            {
                rn=sc.nextInt();
                rn=rn-31;
                if(hotel_ob.luxury_singleerrom[rn]!=null)
                throw new NotAvailable();
                CustDetails(i,rn);
            }
            catch(Exception e)
            {
                System.out.println(x:"Invalid Option");
                return;
            }
            break;
        case 4:
            for(j=0;j<hotel_ob.deluxe_singleerrom.length;j++)
            {
                if(hotel_ob.deluxe_singleerrom[j]==null)
                {
                    System.out.print(j+41+",");
                }
            }
            System.out.print(s:"InEnter room number: ");
            try
            {
                rn=sc.nextInt();
                rn=rn-41;
                if(hotel_ob.deluxe_singleerrom[rn]!=null)
                throw new NotAvailable();
                CustDetails(i,rn);
            }
            catch(Exception e)
            {
                System.out.println(x:"Invalid Option");
                return;
            }
            break;
        default:
            System.out.println(x:"Enter valid option");
            break;
    }
        System.out.println(x:"Room Booked");
    }

    public static void features(int i)
    {
switch (i)
{
case 1:
System.out.println(x:"Number of double beds : 1 \nAC : Yes\nFree breakfast: Yes\nCharge per day 4000 ");
break;
case 2:
System.out.println(x:"Number of double beds : 1 \nAC : No\nFree breakfast: Yes\nCharge per day 3000 ");
break;
case 3:
System.out.println(x:"Number of single beds : 1 \nAC : Yes\nFree breakfast: Yes\nCharge per day 2200 ");
break;
case 4:
System.out.println(x:"Number of single beds : 1 \nAC : No\nFree breakfast: Yes\nCharge per day 1200 ");
break;
default:
System.out.println(x:"Enter valid option");
break;
```

```
}
}

    public static void availability(int i)
    {
        int j,count=0;
        switch (1)
        {
            case 1:
                for(j=0;j<10;j++)
                {
                    if(hotel_ob.luxury_doublerrom[j]==null)
                    count++;
                }
                break;
            case 2:
                for(j=0;j<hotel_ob.deluxe_doublerrom.length;j++)
                {
                    if(hotel_ob.deluxe_doublerrom[j]==null)
                    count++;
                }
                break;
            case 3:
                for(j=0;j<hotel_ob.luxury_singleerrom.length;j++)
                {
                    if(hotel_ob.luxury_singleerrom[j]==null)
                    count++;
                }
                break;
                case 4:
                for(j=0;j<hotel_ob.deluxe_singleerrom.length;j++)
                {
                    if(hotel_ob.deluxe_singleerrom[j]==null)
                    count++;
                }
                break;
            default:
                System.out.println(x:"Enter valld option");
                break;
        }
        System.out.println("Number of rooms available : "+count);
    }
    public static void bill(int rn,int rtype)
    {
        double amount=0;
        String list[]={"Sandwich","Pasta","Noodles","Coke"};
        System.out.println(x:"\n*******");
        System.out.println(x:"Bill:-");
        System.out.println(x:"*******");
        switch(rtype)
        {
            case 1:
                amount+=4000;
                System.out.println("\nRoom Charge - "+4000);
                System.out.println(x:"\n***************");
                System.out.println(x:"Food Charges:-");
                System.out.println(x:"**************");
                System.out.println(x:"Item Quantity Price");
                System.out.println(x:".........................");
                for(Food obb:hotel_ob.luxury_doublerrom[rn].food)
                {
                    amount+=obb.price;
                    System.out.printf(list[obb.itemno-1],obb.quantity,obb.price);
                }
                break;
            case 2:
                amount+=3000;
                System.out.println("Room Charge- "+3000);
                System.out.println(x:"\nFood Charges:- ");
                System.out.println(x:"**************");
                System.out.println(x:"Item Quantity Price");
                System.out.println(x:".........................");
                for(Food obb:hotel_ob.deluxe_doublerrom[rn].food)
                {
                    amount+=obb.price;
                    System.out.printf(list[obb.itemno-1],obb.quantity,obb.price );
                }
                break;
            case 3:
                amount+=2200;
                System.out.println("Room Charge- "+2200);
                System.out.println(x:"\nFood Charges:- ");
                System.out.println(x:"**************");
                System.out.println(x:"Item Quantity Price");
                System.out.println(x:".........................");
                System.out.println(x:"Item Quantity Price");
                System.out.println(x:"------------------ ");
                for(Food obb:hotel_ob.luxury_singleerrom[rn].food)
                {
                    amount+=obb.price;
```

```
                System.out.printf(list[obb.itemno-1],obb.quantity,obb.price);
            }
            break;
        case 4:
            amount+=1200;
            System.out.println("Room Charge- "+1200);
            System.out.println(x:"\nFood Charges:- ");
            System.out.println(x:"**************");
            System.out.println(x:"Item Quantity Price");
            System.out.println(x:".......................");
            for(Food obb: hotel_ob.deluxe_singleerrom[rn].food)
            {
                amount+=obb.price;
                System.out.printf(list[obb.itemno-1],obb.quantity,obb.price );
            }
            break;
        default:
            System.out.println(x:"Not valid");
    }
    System.out.println("InTotal Amount- "+amount);
}
public static void deallocate(int rn,int rtype)
{
    char w;
    switch (rtype)
    {
        case 1:
            if(hotel_ob.luxury_doublerrom[rn]!=null)
            System.out.println("Room used by "+hotel_ob.luxury_doublerrom[rn].name);
            else
            {
                System.out.println(x:"Empty Aready");
                return;
            }
            System.out.println(x:"Do you want to checkout ?(y/n)");
            w=sc.next().charAt(index:0);
            if(w=='y'||w=='Y')
            {
                bill(rn,rtype);
                hotel_ob.luxury_doublerrom[rn]=null;
                System.out.println(x:"Deallocated succesfully");
            }
            break;
        case 2:
            if(hotel_ob.deluxe_doublerrom[rn]!=null)
                System.out.println("Room used by "+hotel_ob.deluxe_doublerrom[rn].name);
            else
            {
                System.out.println(x:"Empty Already");
                return;
            }
            System.out.println(x:" Do you want to checkout ?(y/n)");
            w=sc.next().charAt(index:0);
            if(w=='y'||w=='Y')
            {
                bill(rn,rtype);
                hotel_ob.deluxe_doublerrom[rn]=null;
                System.out.println(x:"Deallocated succesfully");
            }
            break;
        case 3:
            if(hotel_ob.luxury_singleerrom[rn]!=null)
            System.out.println("Room used by "+hotel_ob.luxury_singleerrom[rn].name);
            else
            {
                System.out.println(x:"Empty Already");
                return;
            }
            System.out.println(x:" Do you want to checkout ? (y/n)");
            w=sc.next().charAt(index:0);
            if(w=='y'||w=='Y')
            {
                bill(rn,rtype);
                hotel_ob.luxury_singleerrom[rn]=null;
                System.out.println(x:"Deallocated succesfully");
            }
            break;
        case 4:
            if(hotel_ob.deluxe_singleerrom[rn]!=null)
            System.out.println("Room used by "+hotel_ob.deluxe_singleerrom[rn].name);
            else
            {
                System.out.println(x:"Empty Already");
                return;
            }
            System.out.println(x:" Do you want to checkout ? (y/n)");
            w=sc.next().charAt(index:0);
            if(w=='y'||w=='Y')
            {
                bill(rn,rtype);
```

```
                    hotel_ob.deluxe_singleerrom[rn]=null;
                    System.out.println(x:"Deallocated succesfully");
                }
                break;
            default:
                System.out.println(x:"\nEnter valid option : ");
                break;
        }
    }
    public static void order(int rn,int rtype)
    {
        int i,q;
        char wish;
        try
        {
System.out.println("\n=======\n Menu: \n=========\n\n1.Sandwich"+
"\tRs.50\n2.Pasta\t\tRs.60\n3.Noodles\tRs.70\n4.Coke\t\tRs.30\n");
            do
            {
                i=sc.nextInt();
                System.out.print(s:"Quantity- ");
                q=sc.nextInt();
                switch(rtype)
                {
                    case 1:
                        hotel_ob.luxury_doublerrom[rn].food.add(new Food(i,q));
                        break;
                    case 2:
                        hotel_ob.deluxe_doublerrom[rn].food.add(new Food(i,q));
                        break;
                    case 3:
                        hotel_ob.luxury_singleerrom[rn].food.add(new Food(itemno:1,q));
                        break;
                    case 4:
                        hotel_ob.deluxe_singleerrom[rn].food.add(new Food(i,q));
                        break;
                }
                System.out.println(x:"Do you want to order anything else?  (y/n)");
                wish=sc.next().charAt(index:0);
            }while(wish=='y'||wish=='Y');
        }
        catch(NullPointerException e)
        {
            System.out.println(x:"InRoom not booked");
        }
        catch(Exception e)
        {
            System.out.println(x:"Cannot be done");
        }
    }
}
```

Main.java > write > run()

```java
import java.io.*;
import java.util.*;
import ia2.*;

class write implements Runnable
{
    Holder hotel_ob;
    write(Holder hotel_ob)
    {
        this.hotel_ob=hotel_ob;
    }
    @Override
    public void run()
    {
        try
        {
            FileOutputStream fout=new FileOutputStream(name:"MAIN");
            ObjectOutputStream oos=new ObjectOutputStream(fout);
            oos.close();
            oos.writeObject(hotel_ob);
        }
        catch(Exception e)
        {
            System.out.println("Error in writing "+e);
        }
    }
}

class Main
{
    Run | Debug
    public static void main(String[] args)
    {
        try
        {
            File f= new File(pathname:"MAIN");
            if(f.exists())
```

MAIN.JAVA STARTS HERE

Main.java > Main > main(String[])

```
37      {
38          FileInputStream fin=new FileInputStream(f);
39          ObjectInputStream ois=new ObjectInputStream(fin);
40          ois.close();
41          Hotel.hotel_ob=(Holder)ois.readObject();
42      }
43      Scanner sc = new Scanner(System.in);
44      int ch,ch2;
45      char wish;
46      x:
47      do
48      {
49      System.out.println(x:"\nEnter your choice :\n1.Display room details\n2.Display room availability \n3.Book\n4.Order food\n5.Checkout\n6.Exit\n");
50      ch=sc.nextInt();
51      switch(ch)
52      {
53          case 1:
54          System.out.println(x:"\nChoose room type :\n1.Luxury Double Room\n2.Deluxe Double Room \n3.Luxury Single Room \n4.Deluxe Single Room\n");
55          ch2=sc.nextInt();
56          Hotel.features(ch2);
57          break;
58
59          case 2:
60          System.out.println(x:"\nChoose room type :\n1.Luxury Double Room\n2.Deluxe Double Room \n3.Luxury Single Room \n4.Deluxe Single Room\n");
61          ch2=sc.nextInt();
62          Hotel.availability(ch2);
63          break;
64
65          case 3:
66          System.out.println(x:"\nChoose room type :\n1.Luxury Double Room\n2.Deluxe Double Room \n3.Luxury Single Room \n4.Deluxe Single Room\n");
67          ch2=sc.nextInt();
68          Hotel.bookroom(ch2);
69          break;
70
71          case 4:
72          System.out.println(x:"Room Number -");
73          ch2=sc.nextInt();
```

Main.java > Main > main(String[])

```
74
75              if(ch2>60)
76              System.out.println(x:"Room doesn't exist");
77              else if (ch2>40)
78              Hotel.order(ch2-41,rtype:4);
79              else if (ch2>30)
80              Hotel.order(ch2-31,rtype:3);
81              else if (ch2>10)
82              Hotel.order(ch2-11,rtype:2);
83              else if (ch2>0)
84              Hotel.order(ch2-1,rtype:1);
85              else
86              System.out.println(x:"Room doesn't exist");
87              break;
88              case 5:
89                  System.out.println(x:"Room Number-");
90                  ch2=sc.nextInt();
91                  if(ch2>60)
92                  System.out.println(x:"Room doesn't exist");
93                  else if(ch2>40)
94                  Hotel.deallocate(ch2-41,rtype:4);
95                  else if(ch2>30)
96                  Hotel.deallocate(ch2-31,rtype:3);
97                  else if(ch2>10)
98                  Hotel.deallocate(ch2-11,rtype:2);
99                  else if(ch2>0)
100                 Hotel.deallocate(ch2-1,rtype:1);
101                 else
102                 System.out.println(x:"Room doesn't exist");
103                 break;
104             case 6:
105                 break x;
106     }
107     System.out.println(x:"\nContinue : (y/n)");
108     wish=sc.next().charAt(index:0);
109     if(!(wish=='y'||wish=='Y'||wish=='n'||wish=='N'))
```

```
110         {
111             System.out.println(x:"Invalid Option");
112             System.out.println(x:"\nContinue : (y/n)");
113             wish=sc.next().charAt(index:0);
114         }
115     }while(wish=='y'||wish=='Y');
116     Thread t=new Thread(new write(Hotel.hotel_ob));
117     t.start();
118
119 tch(Exception e)
120
121     System.out.println(x:"Not a valid input");
```

MAIN.JAVA

ENDS HERE

SAMPLE OUTUT

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Run: Main

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ag181\.1-VS CODE PROGRAMS\eesh>  & 'C:\Program Files\Java\jdk-20\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ag181\AppData\Roaming\Code\User\workspaceStorage\535
796d4621170a04153ae6ec1558068\redhat.java\jdt_ws\eesh_491d4dc5\bin' 'Main'

Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit

1

Choose room type :
1.Luxury Double Room
2.Deluxe Double Room
3.Luxury Single Room
4.Deluxe Single Room

2
Number of double beds : 1
AC : No
Free breakfast: Yes
Charge per day 3000

Continue : (y/n)
y

Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit


2

Choose room type :
```

```
Choose room type :
1.Luxury Double Room
2.Deluxe Double Room
3.Luxury Single Room
4.Deluxe Single Room

1
Number of rooms available : 10

Continue : (y/n)
y

Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit

3

Choose room type :
1.Luxury Double Room
2.Deluxe Double Room
3.Luxury Single Room
4.Deluxe Single Room


1

 Choose room number from :
1,2,3,4,5,6,7,8,9,10,/nEnter room number: 8
InEnter customer name: Cust1
Enter contact number: 1234
Enter gender: Male
Enter second customer name: Cust2
Enter contact number: 5678
Enter gender: Female
Room Booked

Continue : (y/n)
y

Enter your choice :
```

```
Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit

4
Room Number -
8

=======
 Menu:
4.Coke          Rs.30

2
Quantity- 2
Do you want to order anything else?  (y/n)
y
n
Cannot be done

Continue : (y/n)
y

Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit

5
Room Number-
8
Room used by Cust1
Do you want to checkout ?(y/n)

Continue : (y/n)
y

Enter your choice :
```

```
Enter your choice :
1.Display room details
2.Display room availability
3.Book
4.Order food
5.Checkout
6.Exit

6
Error in writing java.io.IOException: Stream Closed
PS C:\Users\ag181\.1-VS CODE PROGRAMS\eesh>
```

OUTPUT

ENDS HERE

LEARNING OUTCOMES

1 Improved Programming Skills: Developing a hotel management system in Java required us to apply and enhance our programming knowledge. We gained experience in object-oriented programming, data structures, and algorithm implementation.

2.Enhanced Teamwork and Collaboration: Working on this group project taught us the importance of effective communication and teamwork. We had to coordinate tasks, integrate our code, and resolve conflicts, which improved our ability to collaborate on complex software projects.

3.Problem Solving and Troubleshooting: Throughout the project, we encountered various challenges, from debugging code to optimizing performance. This experience improved our problem-solving skills, critical thinking, and the ability to find efficient solutions to technical issues