**PROJECT REPORT**

**OBJECT ORIENTED ANALYSIS AND**

**DESIGN USING JAVA UE21CS352B**

**Hotel Management System**

**Anika Mishra PES2UG21CS069**

**Arya Tapikar PES2UG21CS095**

**Aryan Deo PES2UG21CS097**

**Ankush Nagar PES2UG21CS905**

**Hotel Management System**

**Problem Statement**

The project focuses on building a hotel management system to streamline hotel operations. The key aspects include managing room availability, booking guest reservations, and overseeing hotel facilities such as food services and other amenities.

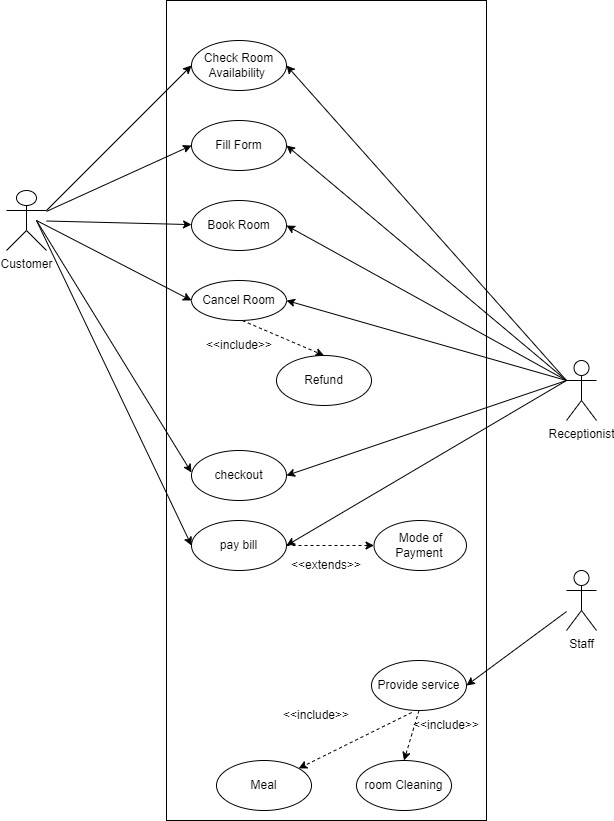
**The system will handle:**

* **Room Management**: Tracking room availability, types, rates, and booking status.
* **Guest Reservations**: Booking rooms, handling guest information, and managing check-in/ check-out processes.
* **Hotel Facilities**: Integrating various amenities and services into booking and billing.
* **Reporting and Analytics**: Providing insights into occupancy rates and revenue for better decision making.

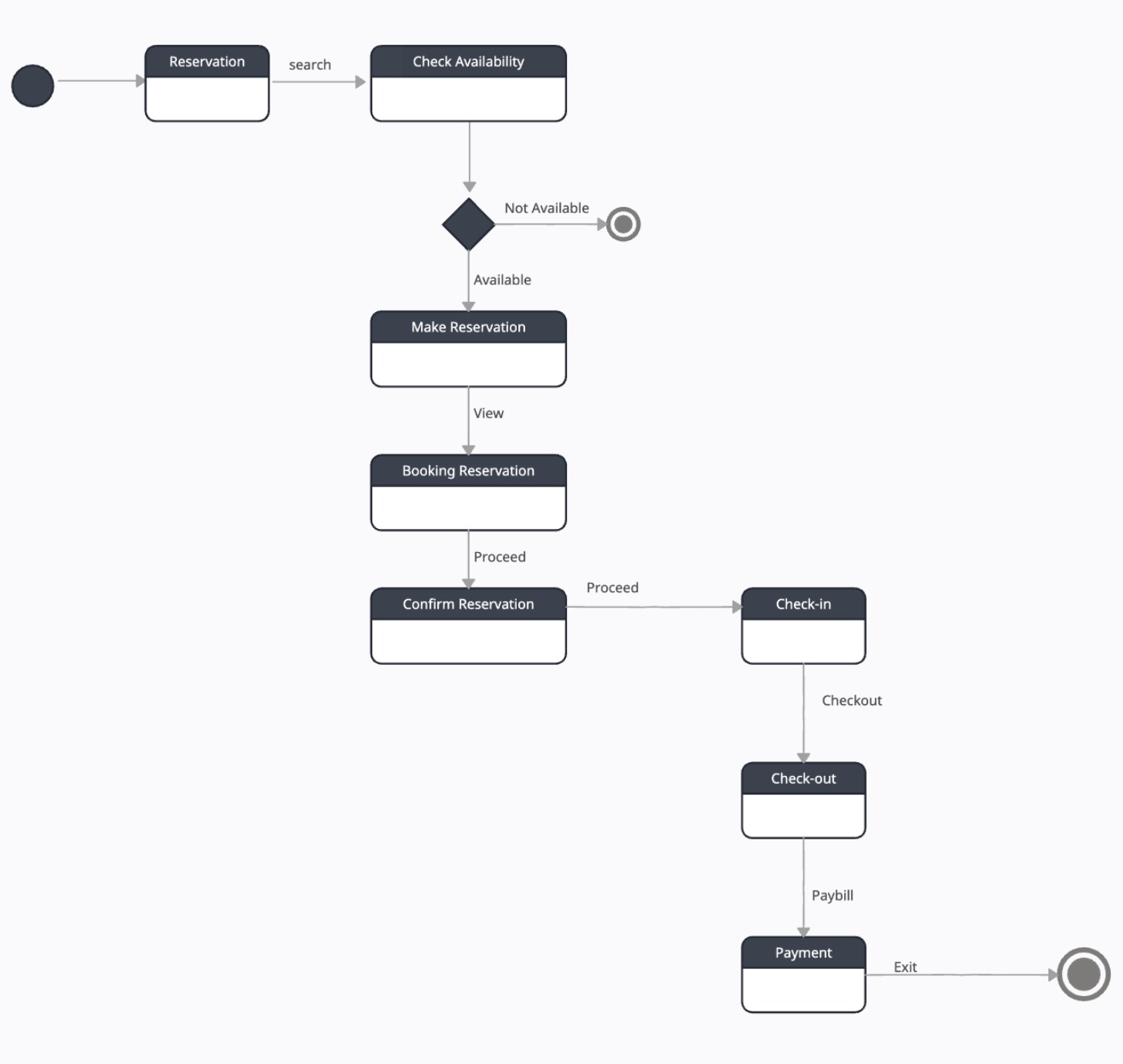
The system should be easy to use, flexible to scale with the hotel's needs, and secure to protect guest and hotel data.

**DESIGN MODELS**

# USE CASE DIAGRAM



# STATE DIAGRAM



# ACTIVITY DIAGRAM

FOR CANCELLATION



AVAILABLE



E

N

Q

U

I

R

Y

F

O

R

R

O

O

M

OPEN

V

I

S

I

T

H

O

T

E

L

NOT

AVAILABLE

F

I

L

L

F

O

R

M

B

O

O

K

R

O

O

M

SUBMIT

GET KEY

U

S

E

R

O

O

M

A

V

A

I

L

S

E

R

V

I

C

E

S

AVAILABLE

C

H

E

C

K

O

U

T

LEAVE ROOM

LOCK ROOM

P

A

Y

B

I

L

L

EXIT



AVAILABLE

B

O

O

K

R

O

O

M

C

A

N

C

L

E

R

O

O

M

LEAVE ROOM



C

H

E

C

K

O

U

T

LOCK ROOM



P

A

Y

R

E

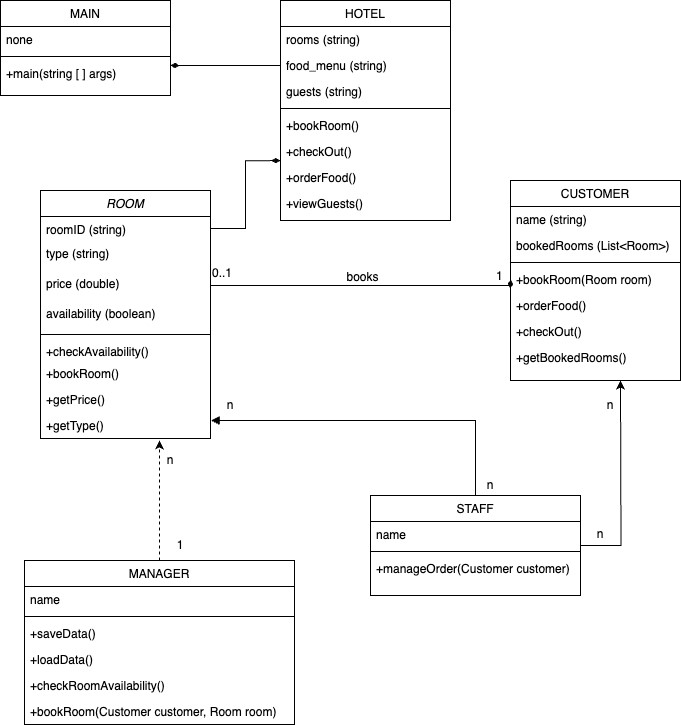
N

T

EXIT

FOR BOOKING

# CLASS DIAGRAM



# MVC ARCHITECTURE

**DESIGN PRINCIPLES, DESIGN PATTERNS AND ARCHITECTURE PATTERNS**

**Design Principles**

* Single Responsibility Principle (SRP): Classes like Hotel, Food, Singleroom, Doubleroom, NotAvailable, and holder each have their own distinct responsibilities.
* Open/Closed Principle (OCP): The code is designed to allow extension without modification. For instance, adding new room types or features can be done without changing existing code.
* Dependency Inversion Principle (DIP): The code relies on interfaces (Serializable) rather than concrete implementations, promoting flexibility and easier maintenance.

**Design Patterns**

* Factory Method Pattern: In the bookroom method, this method dynamically creates instances of different rooms based on the user’s input. It encapsulates the object creation logic, allowing the Hotel class to delegate the responsibility of creating rooms to subclasses (Doubleroom and Singleroom).
* Singleton Pattern: Only one instance of the hotel class is allowed throughout the program execution, by utilizing a private constructor to prevent creation of instances of objects from outside the class. This prevents multiple hotel instances from being created, and ensuring global access to the hotel object.

**Architectural Patterns**

* Layered Architecture: The code follows a clear separation of concerns, with distinct layers for data manipulation, business logic (Hotel class), and presentation (main method).
* Model-View-Controller: The code structure has a separation between data models (Food, Singleroom, Doubleroom, etc.), business logic (Hotel class), and the user interface (main method).

**GITHUB REPO LINK**

**https://github.com/aryatapikar/hotel-Management-Project**

**INDIVIDUAL CONTRIBUTIONS**

Aryan Deo:

**Serialization and Multithreading Module:**

Responsibilities:

* Persisting room booking data using serialization.
* Implementing a separate thread for the serialization process to avoid blocking the main program execution.

Classes Involved:

write, holder.

Main Contributions:

* Designing the serialization mechanism for storing room booking data.
* Developing the multithreading functionality for efficient data persistence.

Ankush Nagar:

**Room Management Module:**

Responsibilities:

* Handling the booking and de-allocation of different types of rooms.
* Displaying room details and availability.

Classes Involved:

Hotel, Singleroom, Doubleroom.

Main Contributions:

* Implementing the methods for booking and de-allocating rooms.
* Developing functions to display room details and availability.

Arya Tapikar:

**Billing and Payment Module:**

Responsibilities:

* Generating bills for guests based on room type and food ordered.
* Handling payments and checkout processes.

Classes Involved:

Food, Hotel.

Main Contributions:

* Implementing bill generation based on room type and food orders.
* Developing functions for payment processing and checkout procedures.

Anika Mishra:

**Food Ordering Module:**

Responsibilities:

* Allowing guests to order food items and associating them with their respective rooms.

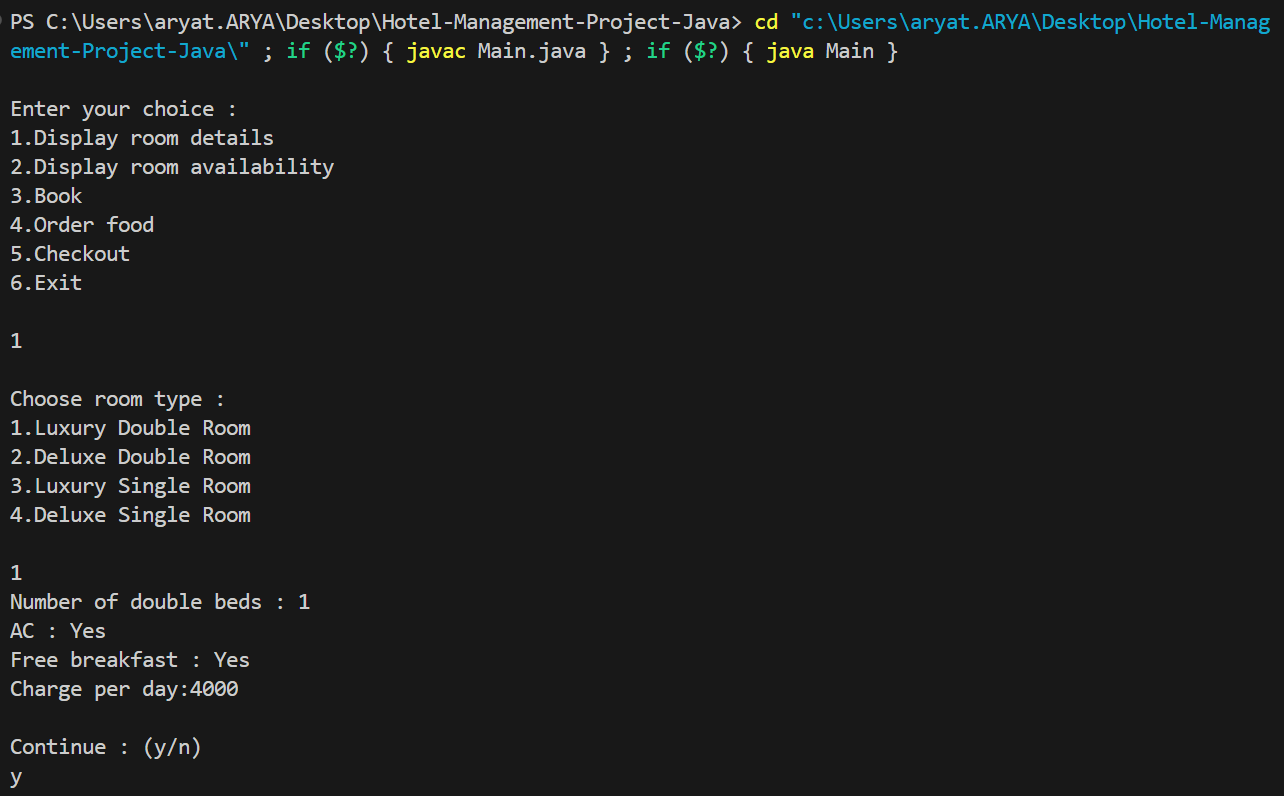
Classes Involved:

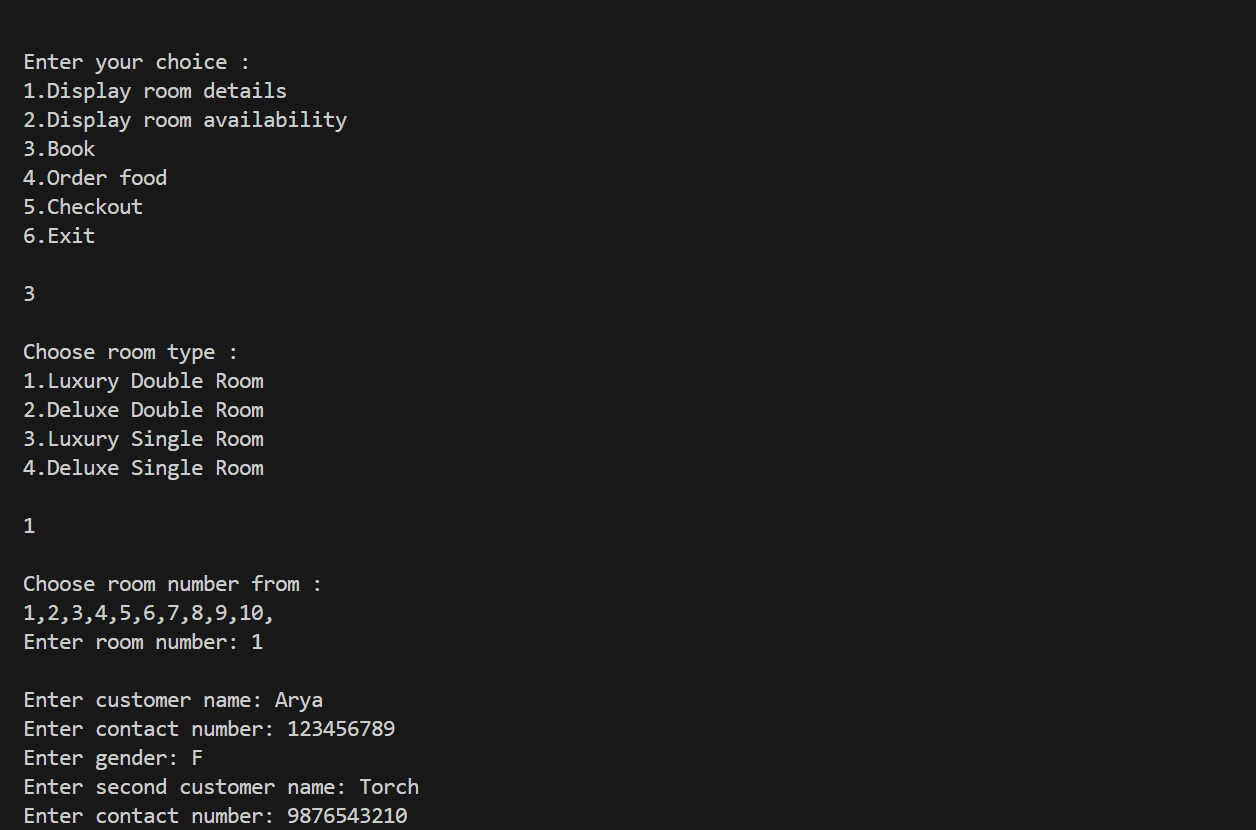
Food.

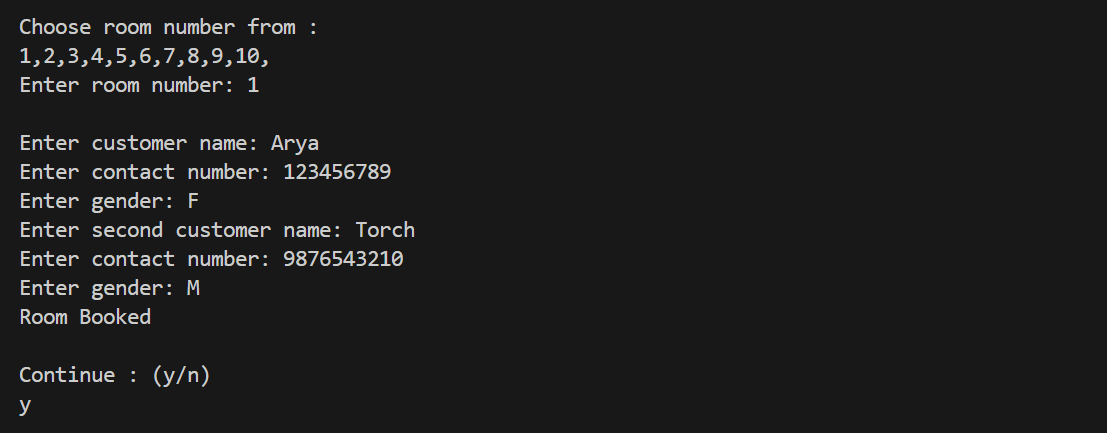
Main Contributions:

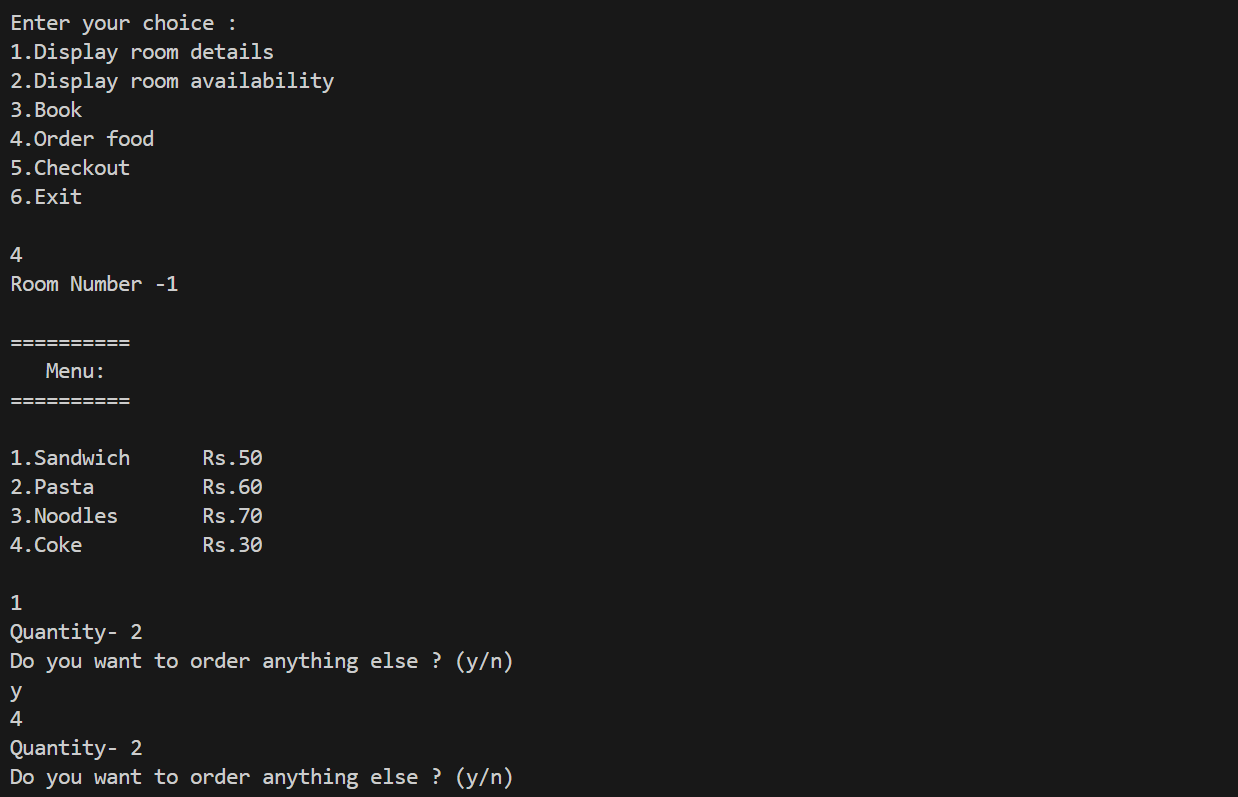
* Creating methods to handle food orders and linking them to room bookings.
* Designing a menu system for users to select food items.

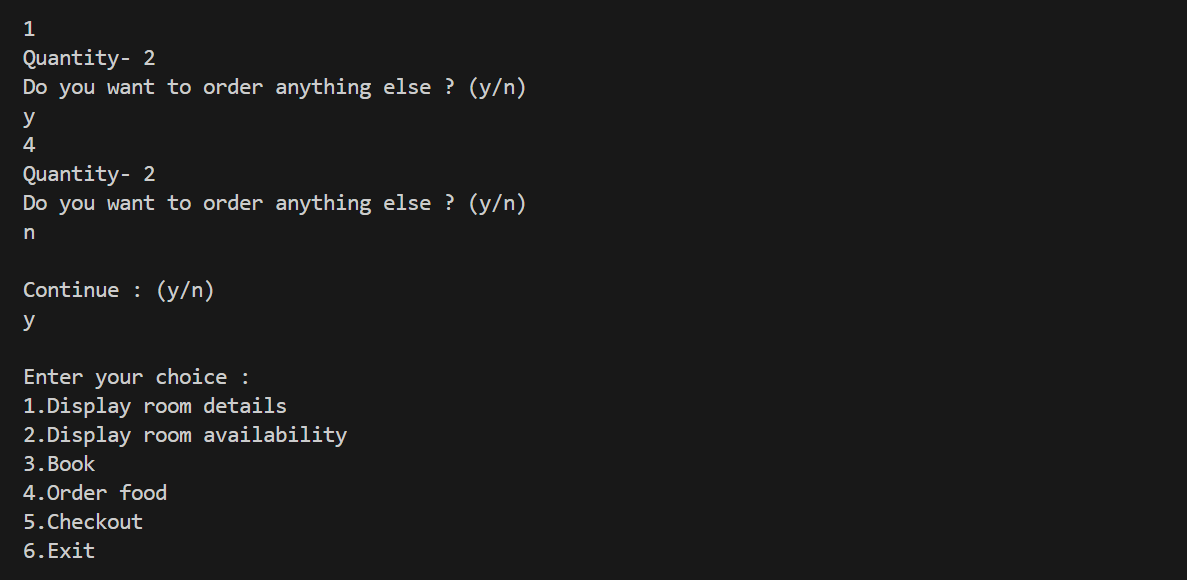
**SCREENSHOTS OF OUTPUT**

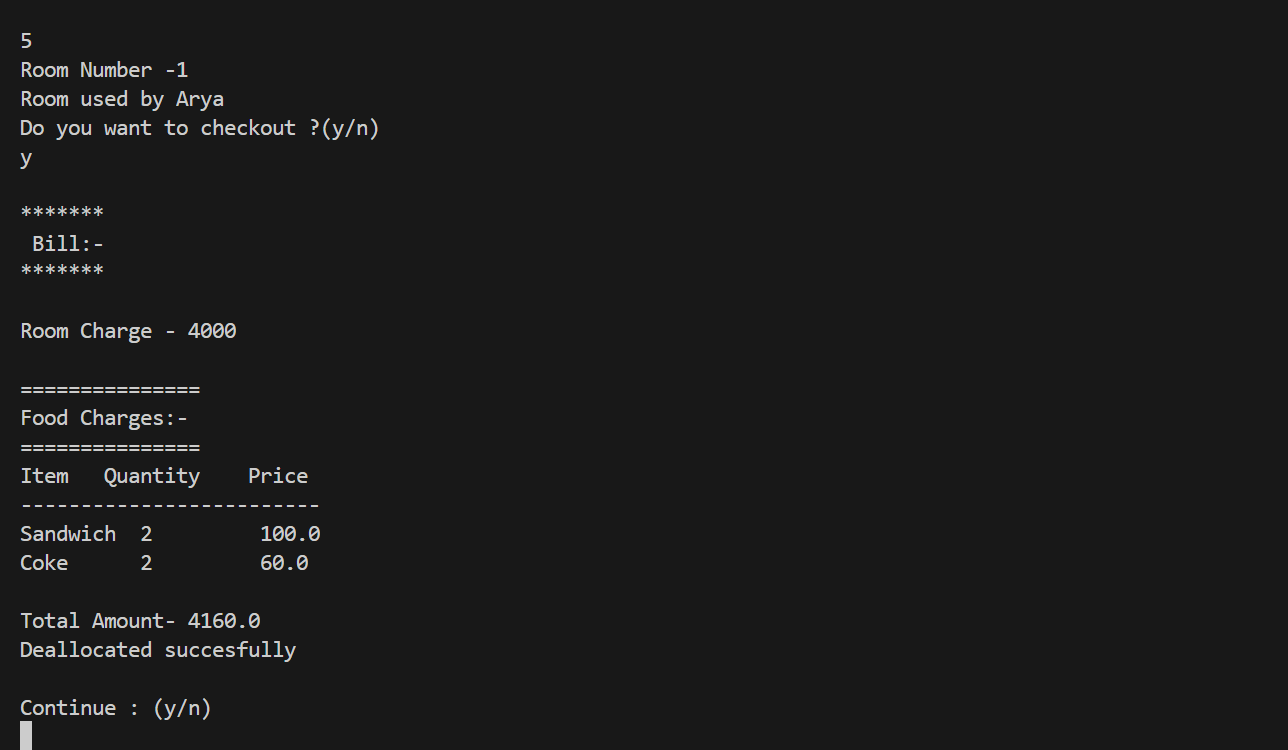
****

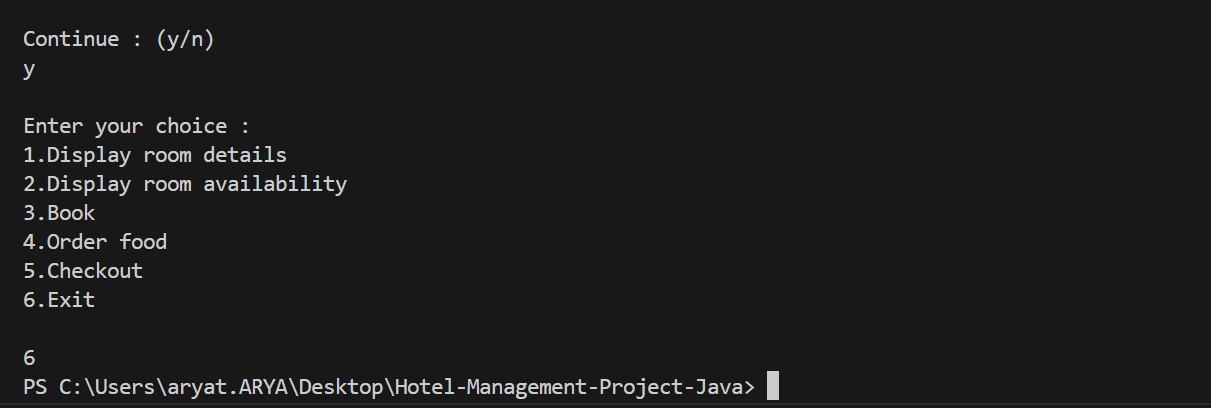
****

****

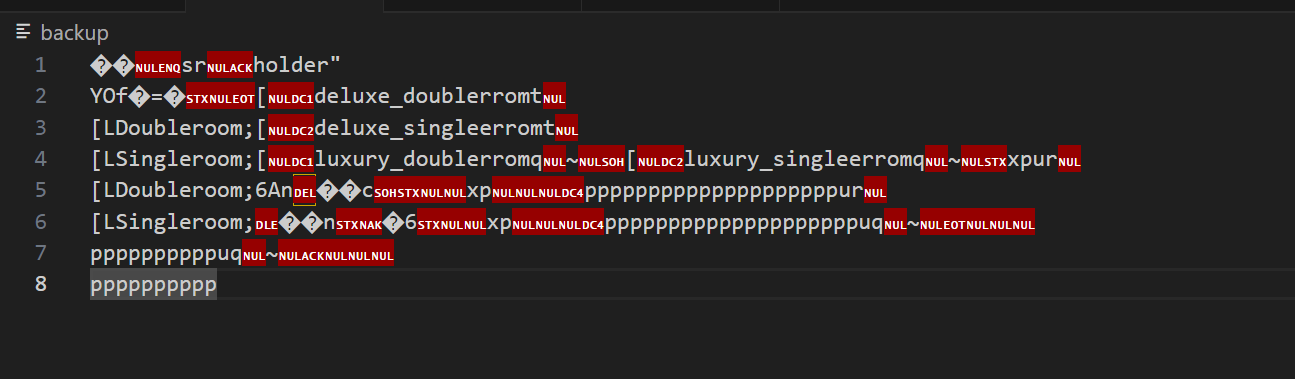
****

****

****

****

Backup:

****