LAB RECORD

23CSE101 – Computational Problem Solving

Submitted by

CH.SC.U4CSE24039 – **SAHIL PAREEK**

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING

CHENNAI

November - 2024

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING, CHENNAI

BONAFIDE CERTIFICATE

This is to certify that the Lab Record work for 23CSE101-Computational Problem Solving Subject submitted by *CH.SC.U4CSE24039 – SAHIL PAREEK* in **"Computer Science and Engineering"** is a bonafide record of the work carried out under my guidance and supervision at Amrita School of Computing, Chennai.

This Lab examination held on 11/03/2025

Internal Examiner 1 Internal Examiner 2

Index

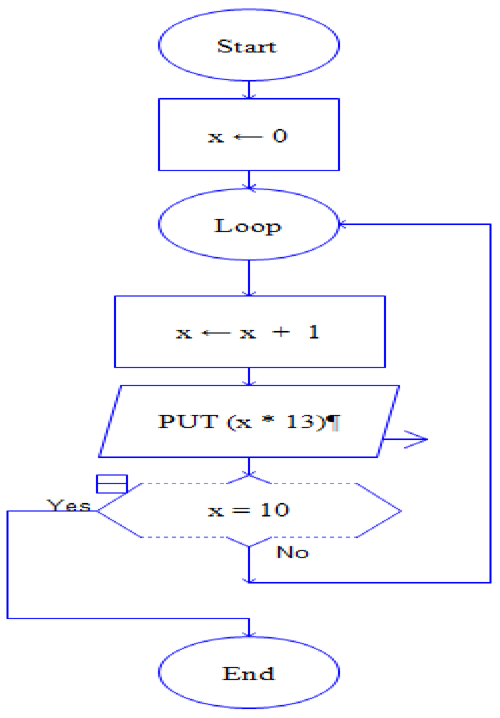
RAPTOR

1) Multiplying a Number Till 10:

Aim: To generate and display the multiplication table of a given number up to 10.

Algorithm:

1. **Input a number from the user.**
2. **Loop from 1 to 10.**
    - **For each iteration, multiply the input number by the loop index.**
    - **Print the result in a formatted manner.**
3. **End the program.**

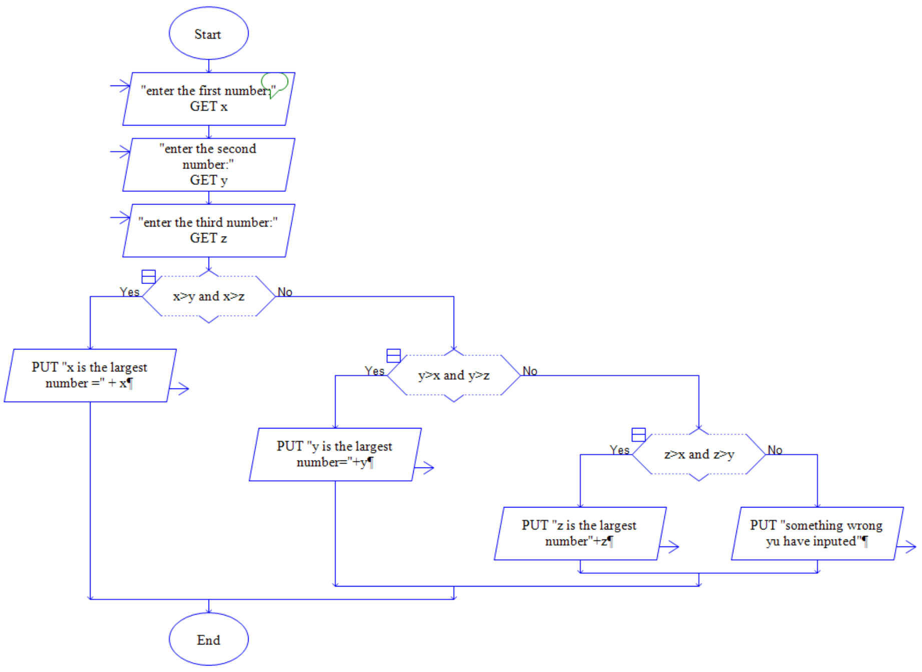


2) Checking Which is the Largest Number from Three:

Aim: To determine and display the largest number among three user-provided numbers.

Algorithm:

1. **Input three numbers from the user.**
2. **Initialize a variable to hold the largest number.**
3. **Compare the first number with the second and third numbers.**
    - **If the first number is greater, update the largest number.**
    - **If the second number is greater, update the largest number.**
    - **If the third number is greater, update the largest number.**
4. **Print the largest number.**
5. **End the program.**



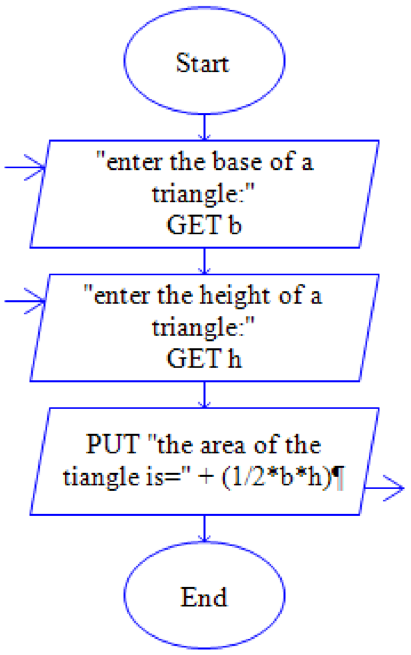
Start
"enter the first number:"
GET x
"enter the second number:"
GET y
"enter the third number:"
GET z
x>y and x>z
Yes
No
PUT "x is the largest number =" + x¶
y>x and y>z
Yes
No
PUT "y is the largest number="+y¶
z>x and z>y
Yes
No
PUT "z is the largest number"+z¶
PUT "something wrong yu have inputed"¶
End

3) Calculating the Area of a Triangle

Aim: To calculate and display the area of a triangle using the base and height provided by the user.

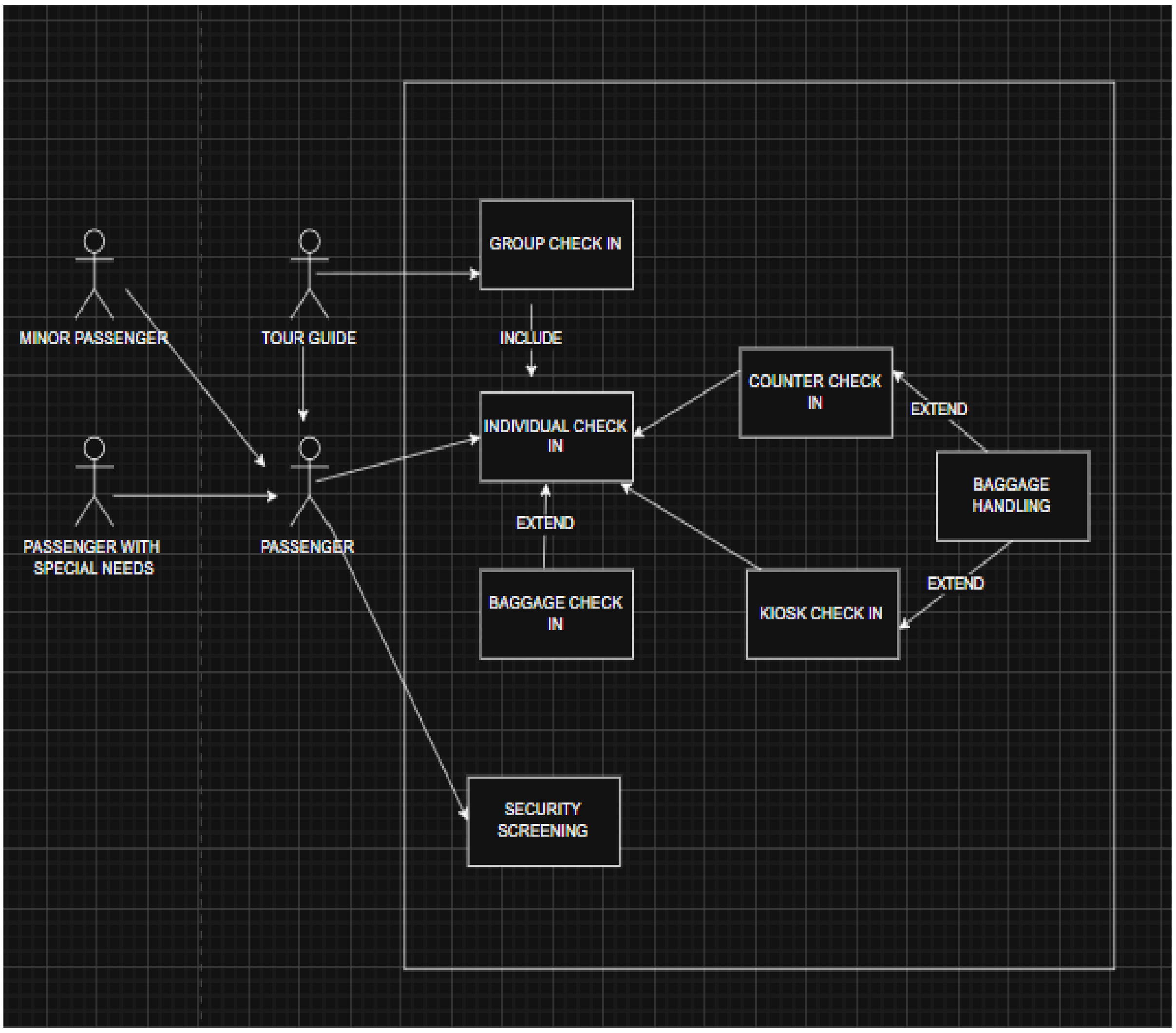
Algorithm:

1. **Input the base and height of the triangle from the user.**
2. **Use the formula for the area of a triangle: Area = (base * height) / 2.**
3. **Calculate the area using the input values.**
4. **Print the calculated area.**
5. **End the program.**


Start
"enter the base of a triangle:" GET b
"enter the height of a triangle:" GET h
PUT "the area of the tiangle is=" + (1/2*b*h)¶
End

USE CASE DIAGRAMS(UML)

1. **Airport Check-In:**
    - **Aim: To illustrate the process of passengers checking in for flights, including interactions with airline staff and kiosks.**
    - **Algorithm:**
        1. **Identify actors: passengers, airline staff, check-in kiosks.**
        2. **Define use cases: check-in, baggage drop, seat selection.**
        3. **Establish relationships between actors and use cases.**
        4. **Create the diagram with actors and use cases.**
        5. **Review for clarity.**

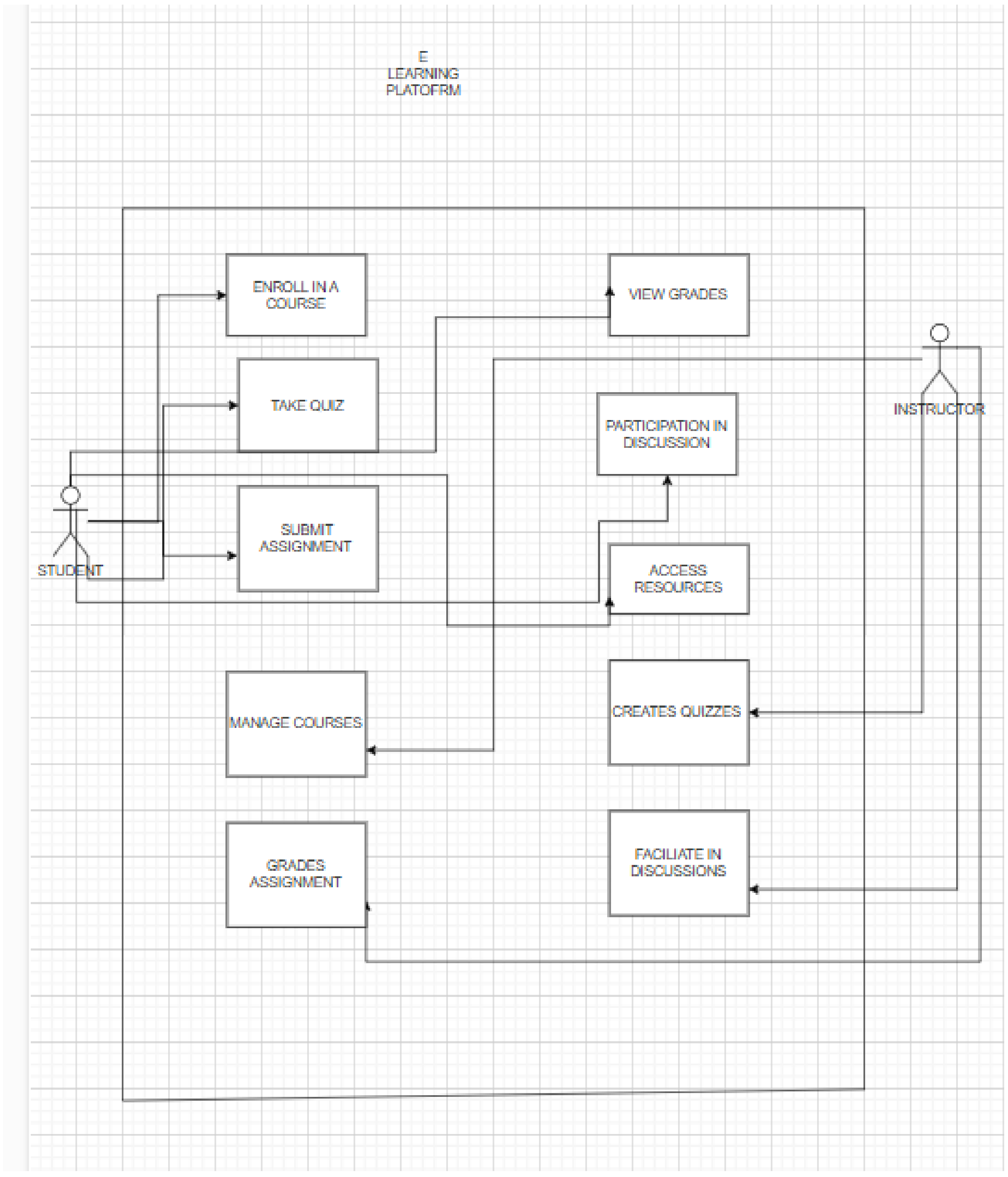


2. <u>E-Learning Platform:</u>

- **<u>Aim: To represent user interactions with the platform, including students, instructors, and administrators.</u>**
- **<u>Algorithm:</u>**
  1. **<u>Identify actors: students, instructors, administrators.</u>**
  2. **<u>Define use cases: enroll in courses, submit assignments, grade submissions.</u>**
  3. **<u>Establish relationships between actors and use cases.</u>**

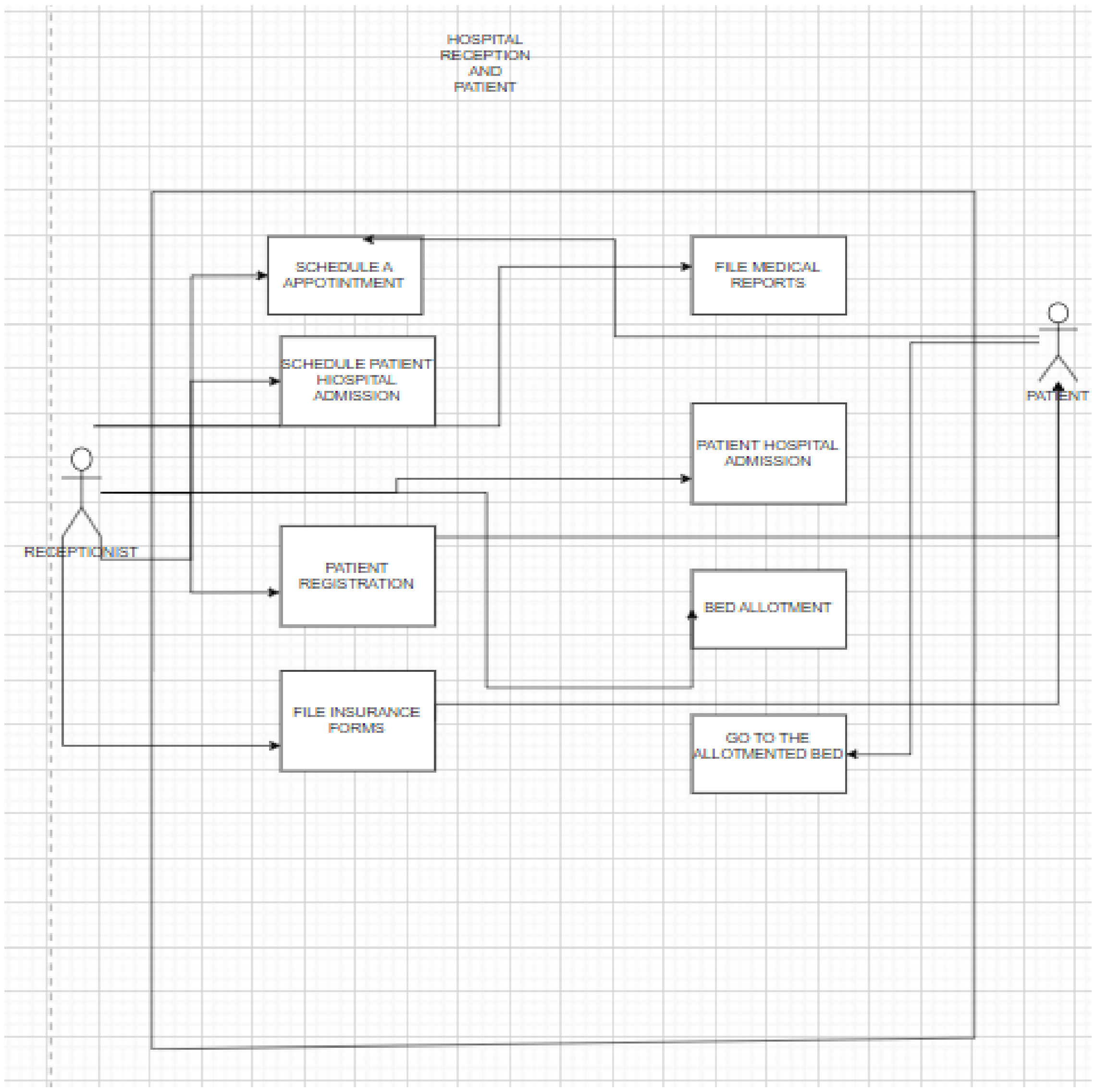
4. Create the diagram with actors and use cases.

5. Review for completeness.



3. **<u>Hospital Receptionist and Patient:</u>**

- **<u>Aim: To depict the interactions between patients and hospital receptionists during the appointment process.</u>**
- **<u>Algorithm:</u>**
    1. **<u>Identify actors: patients, receptionists.</u>**
    2. **<u>Define use cases: schedule appointment, check-in, update patient information.</u>**
    3. **<u>Establish relationships between actors and use cases.</u>**
    4. **<u>Create the diagram with actors and use cases.</u>**
    5. **<u>Review for accuracy.</u>**

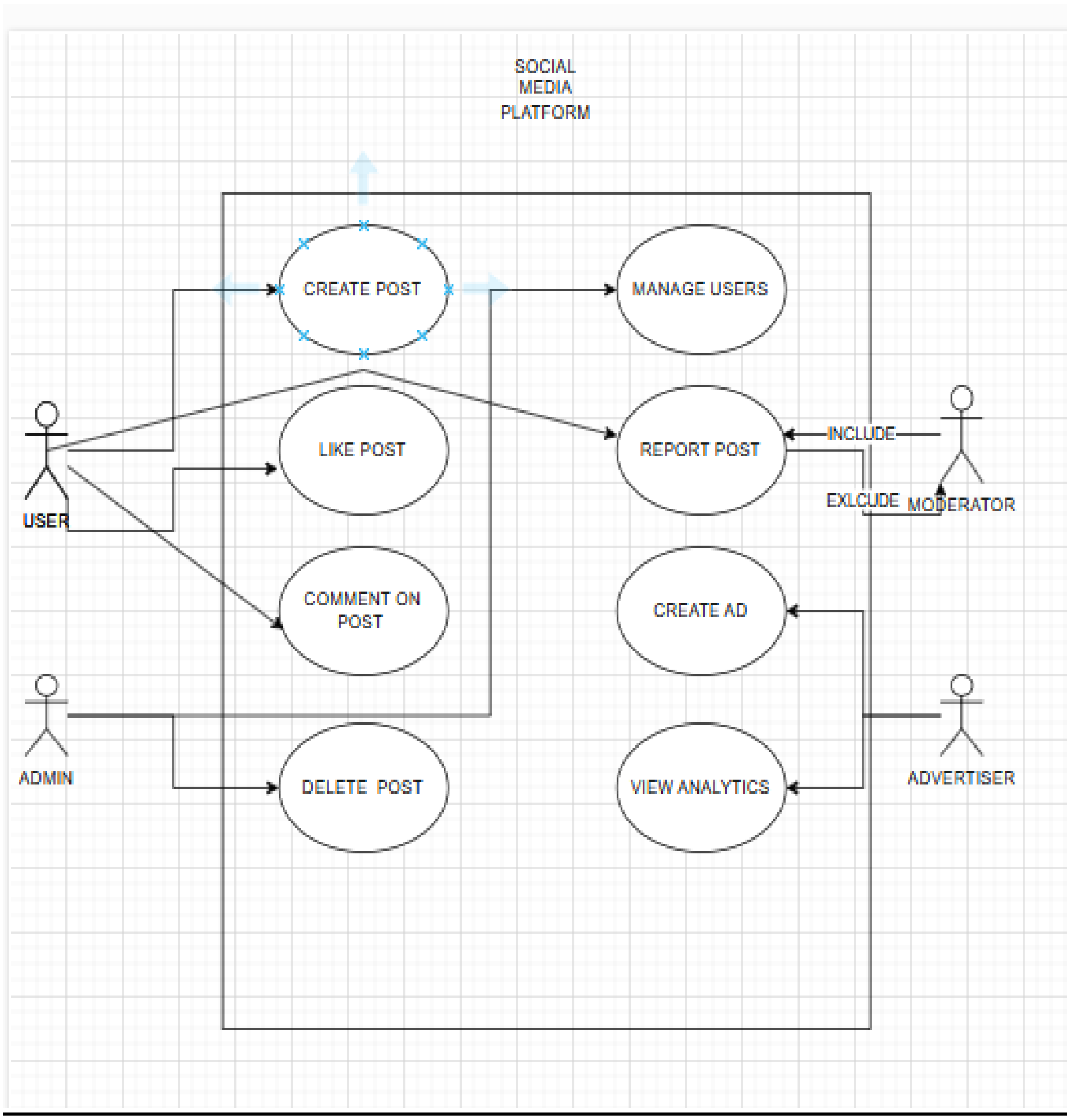

HOSPITAL RECEPTION AND PATIENT
SCHEDULE A APPOTINTMENT
SCHEDULE PATIENT HIOSPITAL ADMISSION
FILE MEDICAL REPORTS
PATIENT
PATIENT HOSPITAL ADMISSION
RECEPTIONIST
PATIENT REGISTRATION
BED ALLOTMENT
FILE INSURANCE FORMS
GO TO THE ALLOTMENTED BED

4. Social Media Platform:

- **Aim: To illustrate user interactions on a social media platform, including posting content and managing profiles.**
- **Algorithm:**
  1. **Identify actors: users, administrators.**
  2. **Define use cases: create post, like post, follow user.**
  3. **Establish relationships between actors and use cases.**

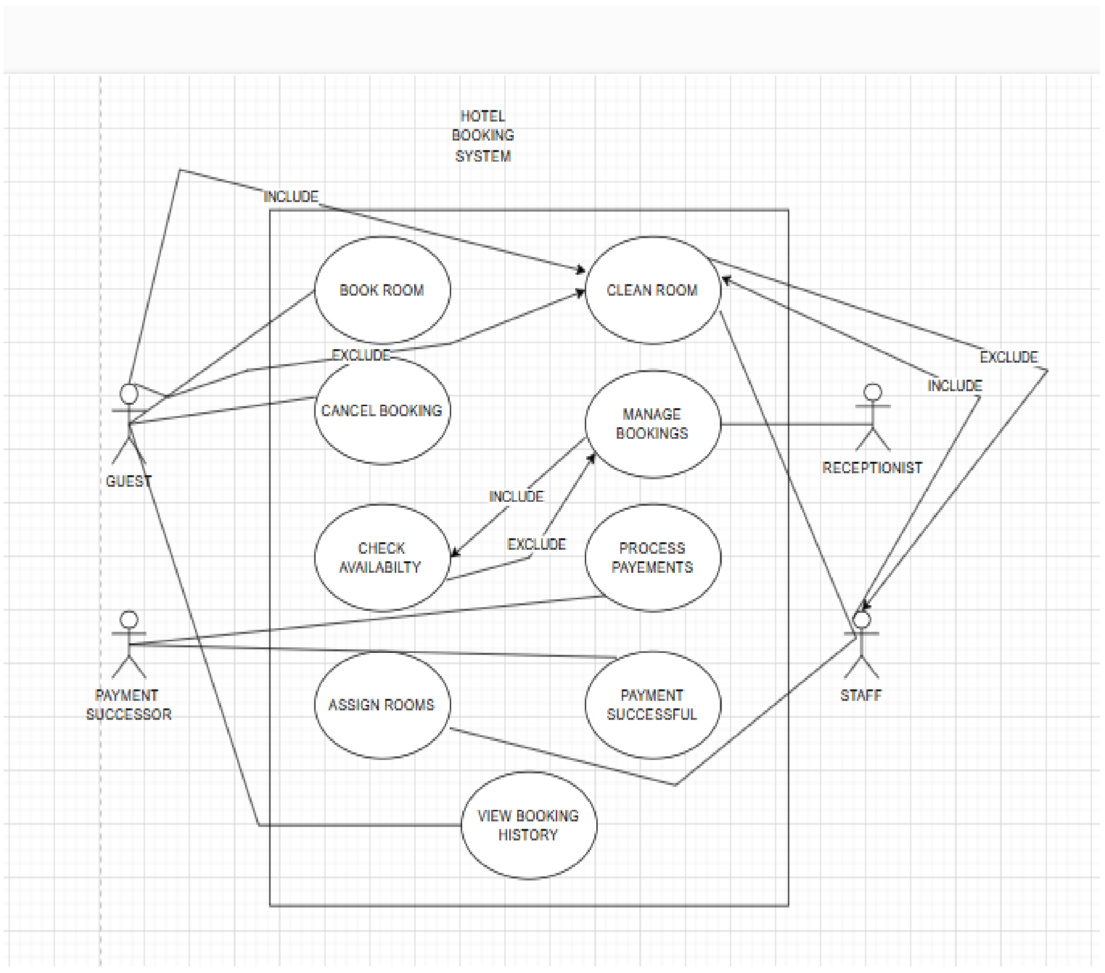
4. Create the diagram with actors and use cases.

5. Review for clarity.



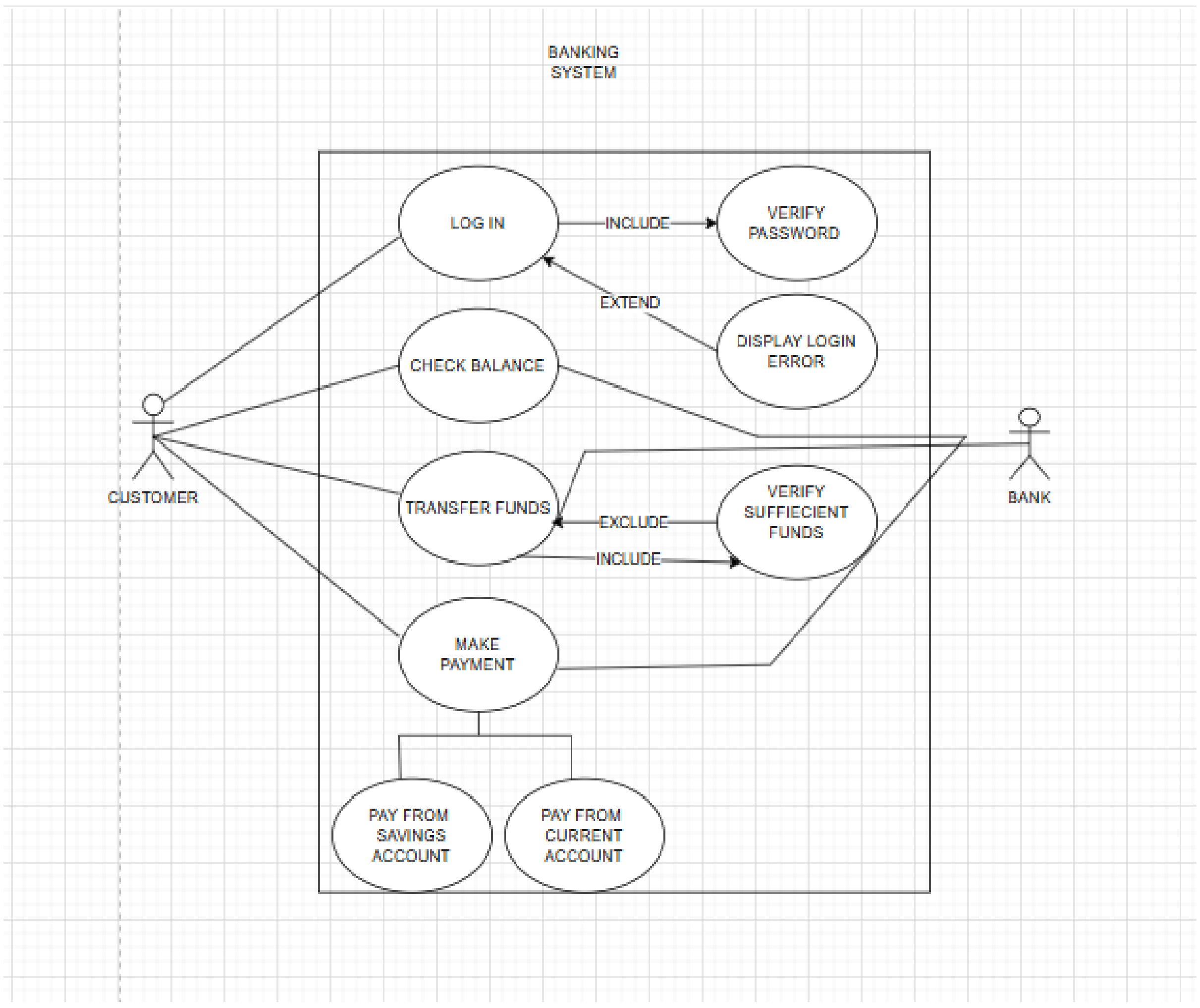
5. **Hotel Booking System:**

- **Aim: To represent the interactions between customers and the hotel booking system for reservations.**
- **Algorithm:**
    1. **Identify actors: customers, hotel staff.**
    2. **Define use cases: search for hotels, make a reservation, cancel a booking.**
    3. **Establish relationships between actors and use cases.**
    4. **Create the diagram with actors and use cases.**
    5. **Review for completeness.**



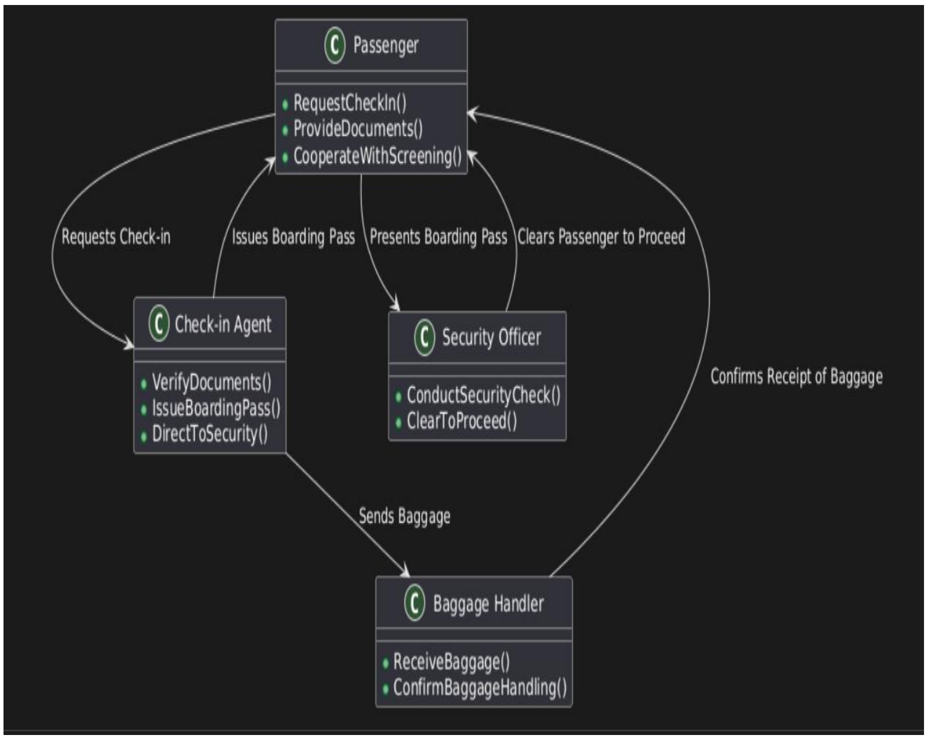
6. **<u>Banking System:</u>**

- **<u>Aim: To depict user interactions with a banking system, including account management and transactions.</u>**
- **<u>Algorithm:</u>**
    1. **<u>Identify actors: customers, bank staff.</u>**
    2. **<u>Define use cases: deposit funds, withdraw funds, check balance.</u>**
    3. **<u>Establish relationships between actors and use cases.</u>**
    4. **<u>Create the diagram with actors and use cases.</u>**
    5. **<u>Review for accuracy.</u>**


BANKING SYSTEM
LOG IN
INCLUDE
VERIFY PASSWORD
EXTEND
DISPLAY LOGIN ERROR
CHECK BALANCE
CUSTOMER
BANK
TRANSFER FUNDS
EXCLUDE
VERIFY SUFFIECIENT FUNDS
INCLUDE
MAKE PAYMENT
PAY FROM SAVINGS ACCOUNT
PAY FROM CURRENT ACCOUNT

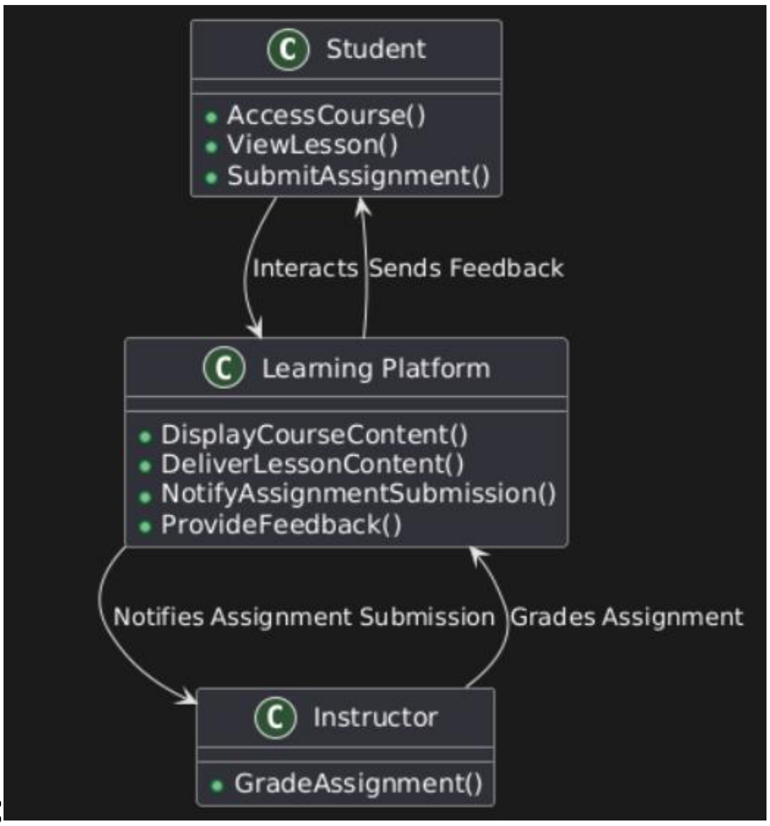
CLASS CASE DIAGRAMS(UML)

1. **Airport Check-In:**
   - **Aim: To represent the structure of the check-in system, including classes for passengers, flights, and check-in kiosks.**
   - **Algorithm:**
     1. **Identify key classes: Passenger, Flight, CheckInKiosk.**
     2. **Define attributes and methods for each class.**
     3. **Determine relationships (e.g., Passenger has a Flight).**
     4. **Create the diagram with classes and their relationships.**
     5. **Validate for accuracy.**



2. **E-Learning Platform:**

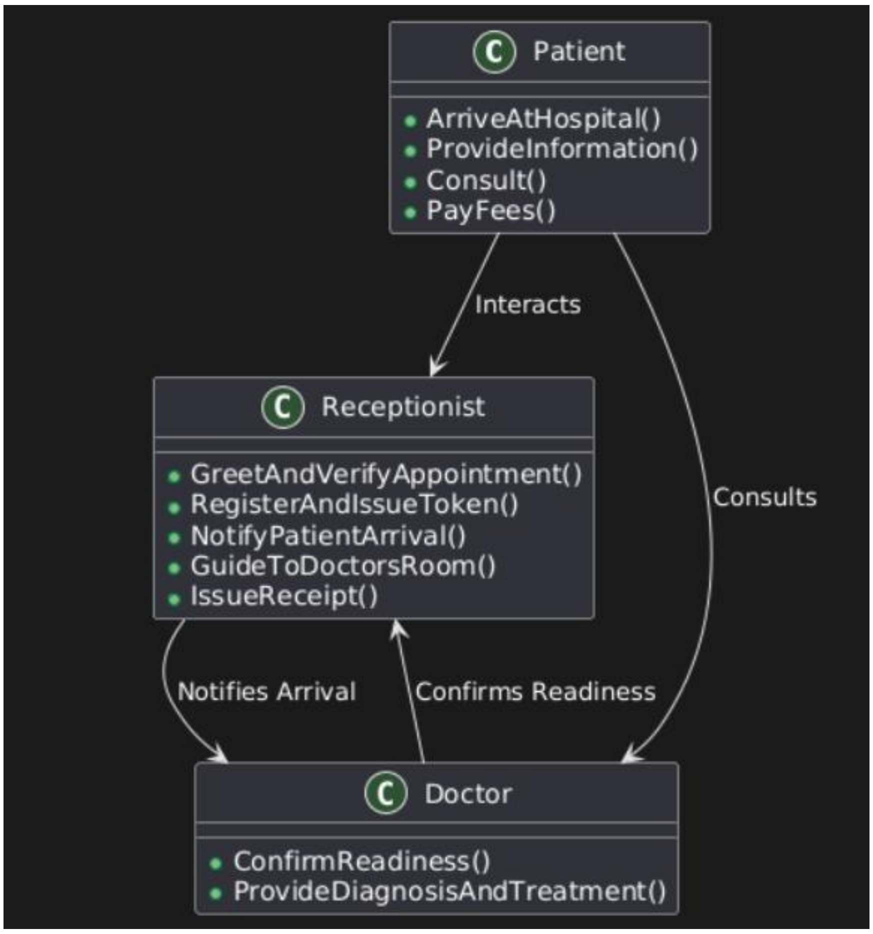
- **Aim: To illustrate the structure of the e-learning system, including classes for users, courses, and assignments.**
- **Algorithm:**
    1. **Identify key classes: User, Course, Assignment.**
    2. **Define attributes and methods for each class.**
    3. **Determine relationships (e.g., User enrolls in Course).**
    4. **Create the diagram with classes and their relationships.**
    5. **Validate for completeness.**



:

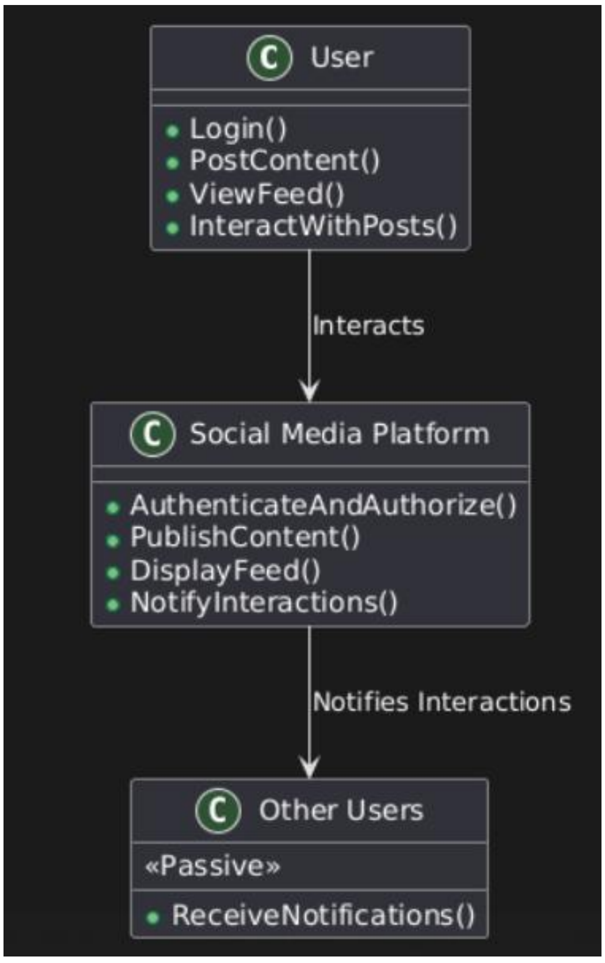
3. Hospital Receptionist and Patient:

- **Aim: To depict the structure of the hospital management system, including classes for patients, appointments, and receptionists.**
- **Algorithm:**
    1. **Identify key classes: Patient, Appointment, Receptionist.**
    2. **Define attributes and methods for each class.**
    3. **Determine relationships (e.g., Patient has an Appointment).**
    4. **Create the diagram with classes and their relationships.**
    5. **Validate for accuracy.**



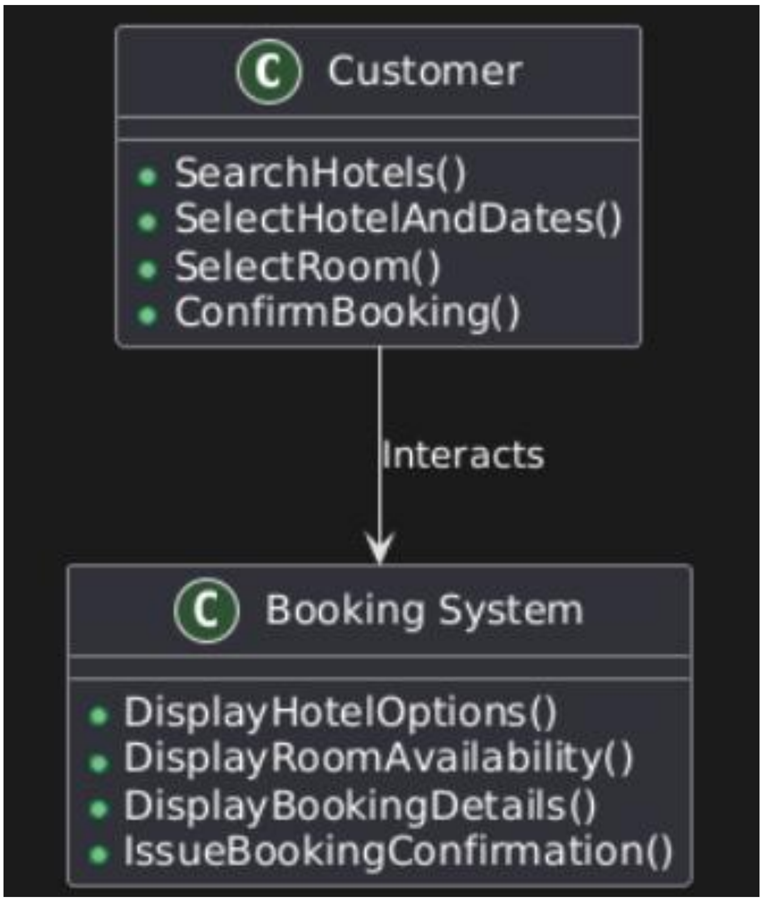
4. **Social Media Platform:**

- **Aim: To represent the structure of the social media system, including classes for users, posts, and comments.**
- **Algorithm:**
    1. **Identify key classes: User, Post, Comment.**
    2. **Define attributes and methods for each class.**
    3. **Determine relationships (e.g., User creates Post).**
    4. **Create the diagram with classes and their relationships.**
    5. **Validate for completeness.**



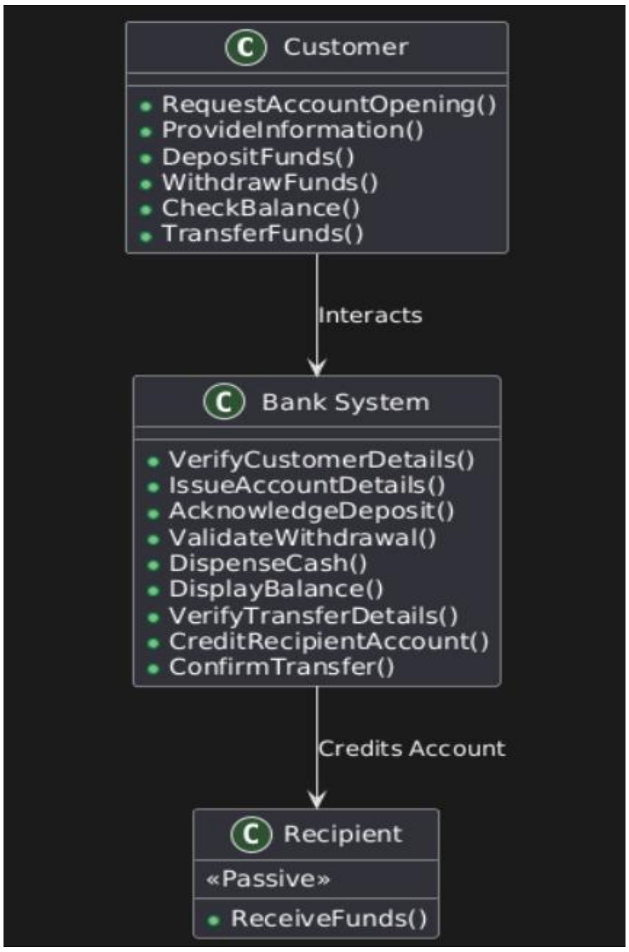
5. Hotel Booking System:

- **Aim: To illustrate the structure of the hotel booking system, including classes for customers, reservations, and rooms.**
- **Algorithm:**
  1. **Identify key classes: Customer, Reservation, Room.**
  2. **Define attributes and methods for each class.**
  3. **Determine relationships (e.g., Customer makes Reservation).**
  4. **Create the diagram with classes and their relationships.**
  5. **Validate for accuracy.**



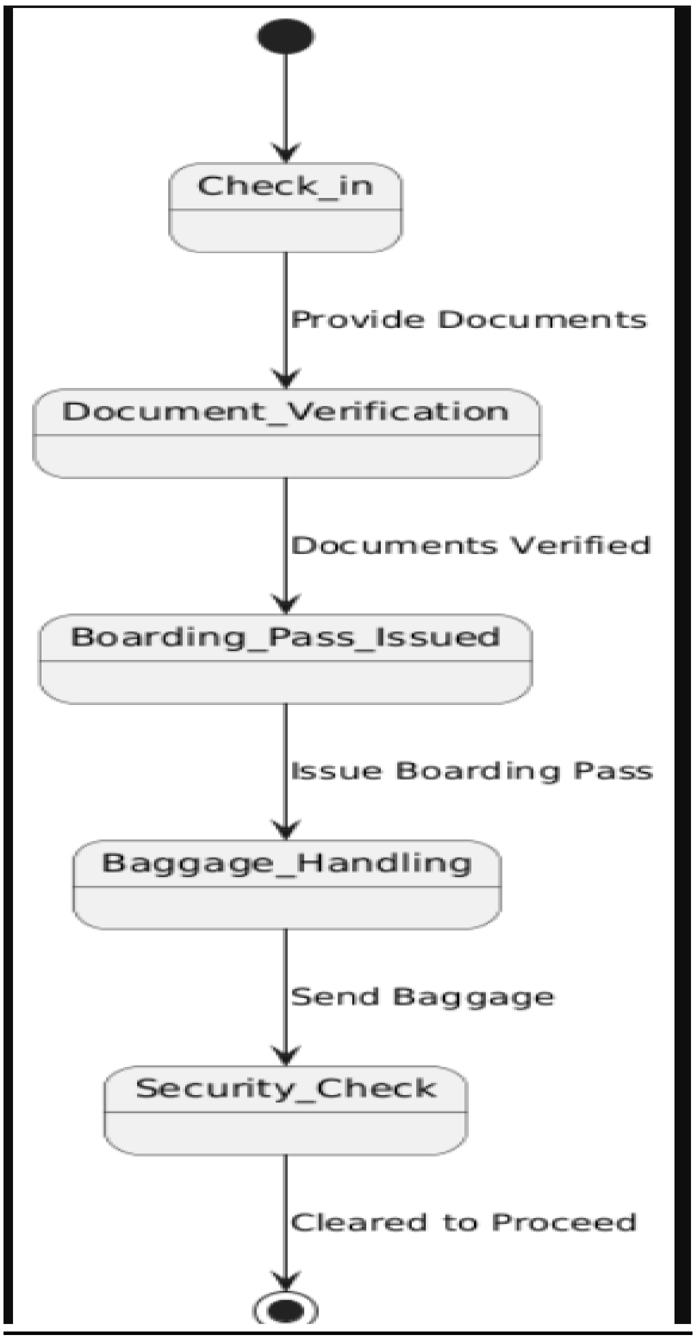
6. Banking System:

- **Aim: To depict the structure of the banking system, including classes for customers, accounts, and transactions.**
- **Algorithm:**
  1. **Identify key classes: Customer, Account, Transaction.**
  2. **Define attributes and methods for each class.**
  3. **Determine relationships (e.g., Customer owns Account).**
  4. **Create the diagram with classes and their relationships.**
  5. **Validate for completeness.**

: 

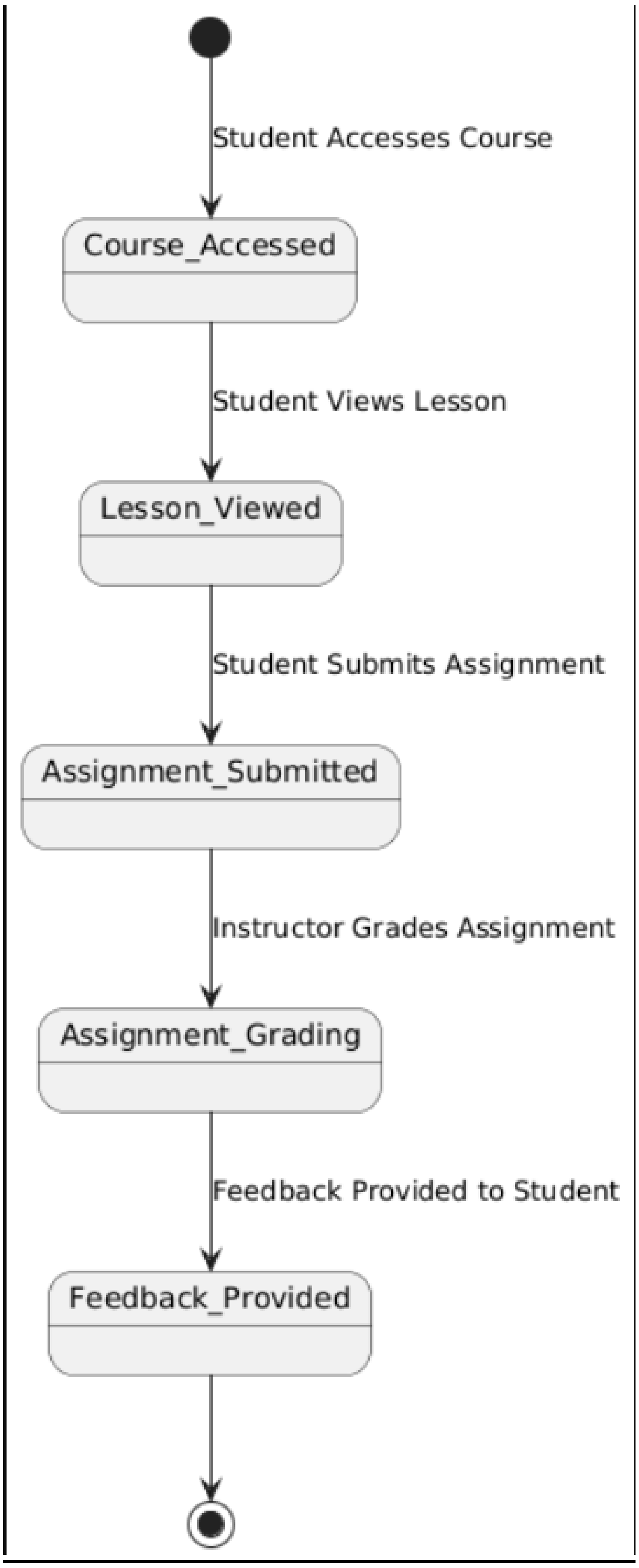
STATE CASE DIAGRAMS(UML)

1. Airport Check-In
   - **Aim: To represent the different states of a passenger during the check-in process at the airport.**
   - **Algorithm:**
     1. **Identify the states: At Airport, Checked In, Baggage Dropped, Security Cleared, Boarded.**
     2. **Define events that cause transitions between states (e.g., Check In, Drop Baggage, Pass Security, Board Flight).**
     3. **Create the diagram with states represented as circles or rounded rectangles.**
     4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
     5. **Validate for completeness and accuracy.**


Check_in
Provide Documents
Document_Verification
Documents Verified
Boarding_Pass_Issued
Issue Boarding Pass
Baggage_Handling
Send Baggage
Security_Check
Cleared to Proceed

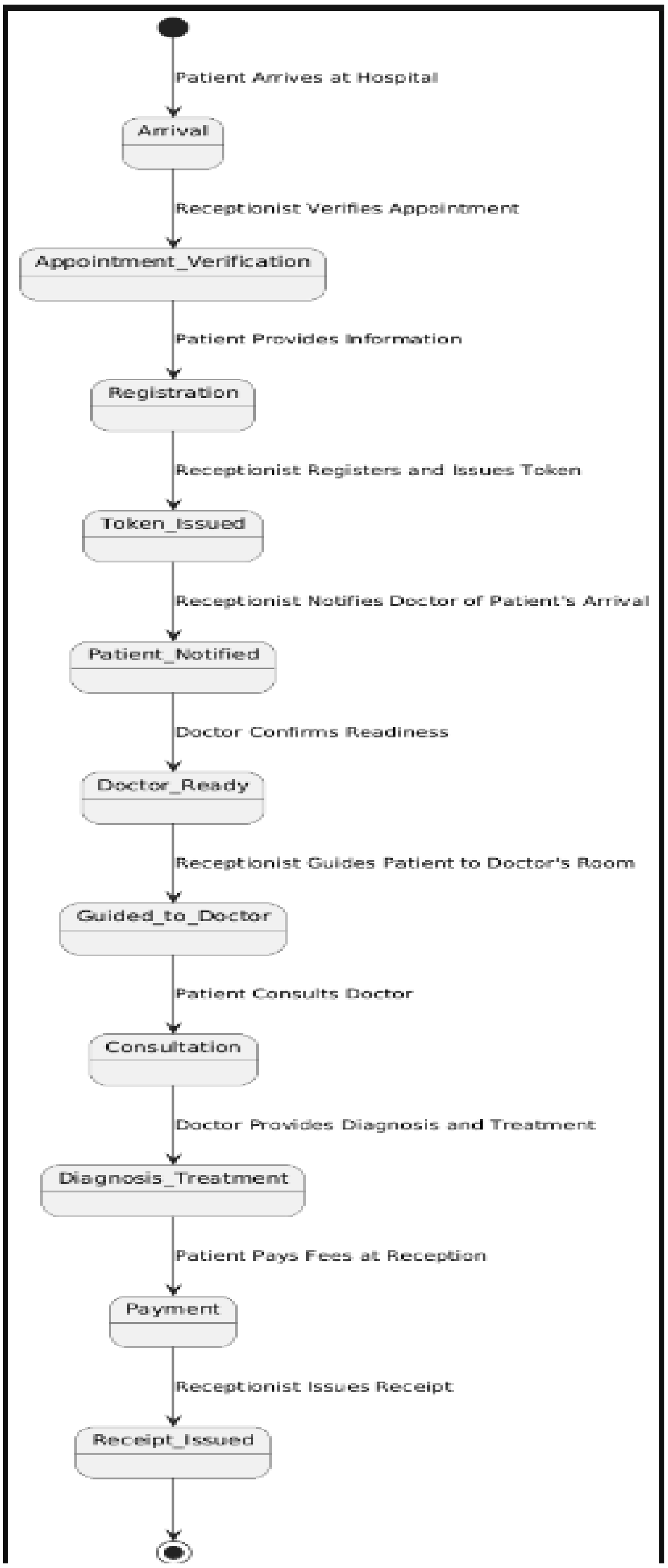
2. **E-Learning Platform**

- **Aim: To illustrate the states of a course enrollment process in an e-learning platform.**
- **Algorithm:**
    1. **Identify the states: Not Enrolled, Enrolled, In Progress, Completed, Graded.**
    2. **Define events that cause transitions (e.g., Enroll, Start Course, Submit Assignment, Receive Grade).**
    3. **Create the diagram with states represented as circles or rounded rectangles.**
    4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
    5. **Validate for clarity and completeness.**


Student Accesses Course
Course_Accessed
Student Views Lesson
Lesson_Viewed
Student Submits Assignment
Assignment_Submitted
Instructor Grades Assignment
Assignment_Grading
Feedback Provided to Student
Feedback_Provided

3. Hospital Receptionist and Patient

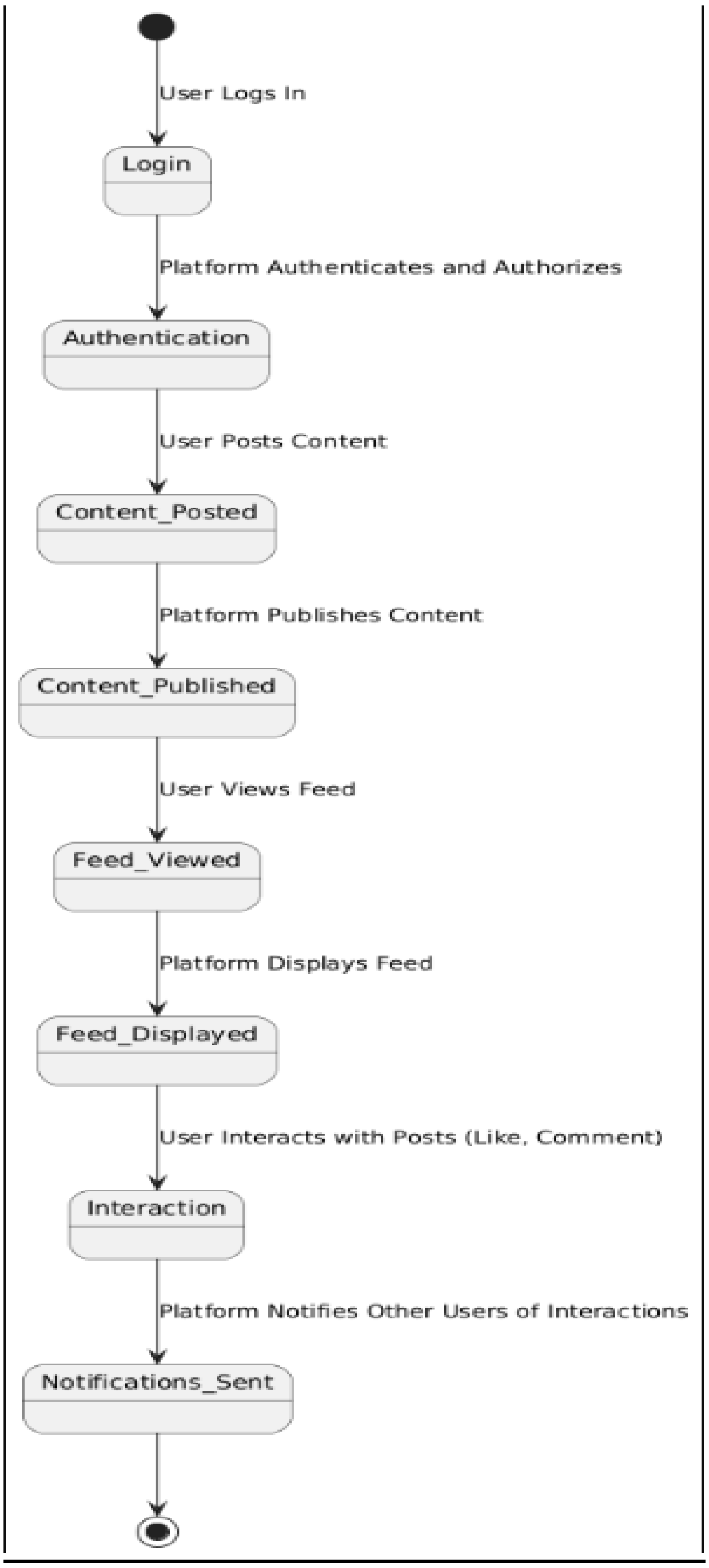
- **Aim: To depict the states of a patient during the appointment process at a hospital.**
- **Algorithm:**
    1. **Identify the states: Registered, Checked In, In Consultation, Completed Consultation, Discharged.**
    2. **Define events that cause transitions (e.g., Register, Check In, Start Consultation, Complete Consultation).**
    3. **Create the diagram with states represented as circles or rounded rectangles.**
    4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
    5. **Validate for accuracy and completeness.**



Patient Arrives at Hospital
Arrival
Receptionist Verifies Appointment
Appointment_Verification
Patient Provides Information
Registration
Receptionist Registers and Issues Token
Token_Issued
Receptionist Notifies Doctor of Patient's Arrival
Patient_Notified
Doctor Confirms Readiness
Doctor_Ready
Receptionist Guides Patient to Doctor's Room
Guided_to_Doctor
Patient Consults Doctor
Consultation
Doctor Provides Diagnosis and Treatment
Diagnosis_Treatment
Patient Pays Fees at Reception
Payment
Receptionist Issues Receipt
Receipt_Issued

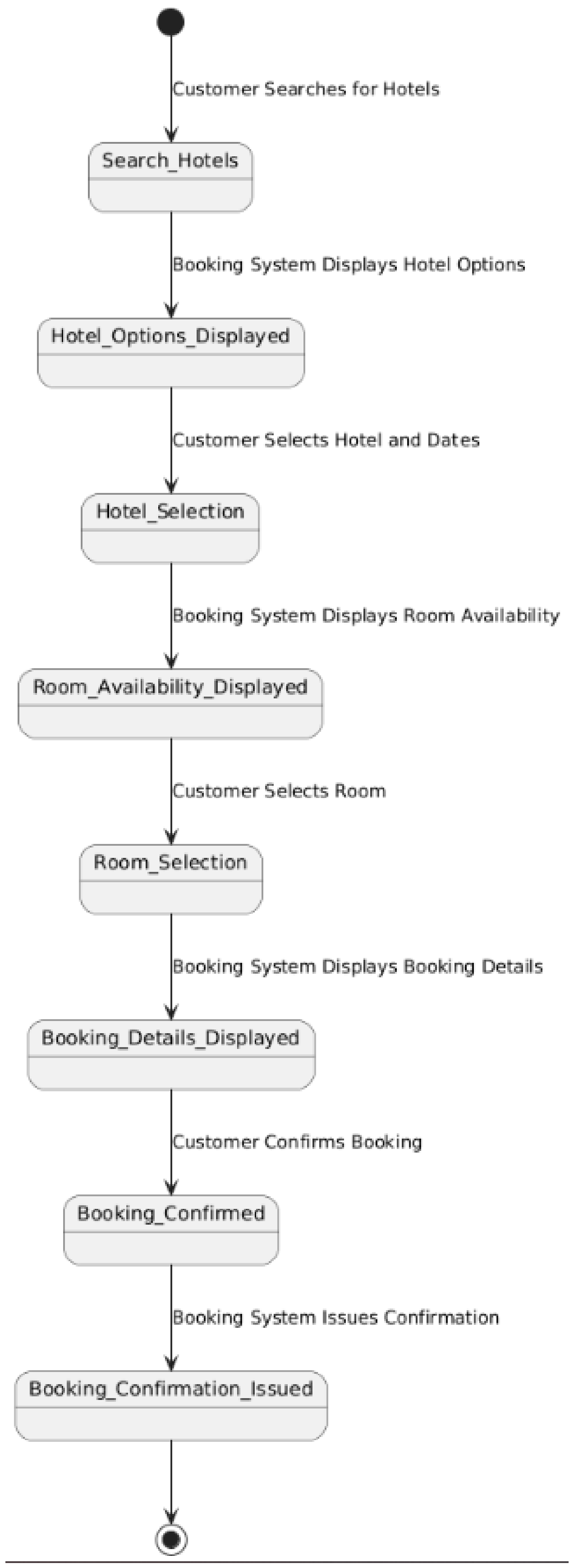
4. **Social Media Platform**

- **Aim: To represent the states of a user’s post on a social media platform.**
- **Algorithm:**
    1. **Identify the states: Draft, Published, Liked, Commented, Archived.**
    2. **Define events that cause transitions (e.g., Publish Post, Like Post, Comment on Post, Archive Post).**
    3. **Create the diagram with states represented as circles or rounded rectangles.**
    4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
    5. **Validate for clarity and completeness.**


User Logs In
Login
Platform Authenticates and Authorizes
Authentication
User Posts Content
Content_Posted
Platform Publishes Content
Content_Published
User Views Feed
Feed_Viewed
Platform Displays Feed
Feed_Displayed
User Interacts with Posts (Like, Comment)
Interaction
Platform Notifies Other Users of Interactions
Notifications_Sent

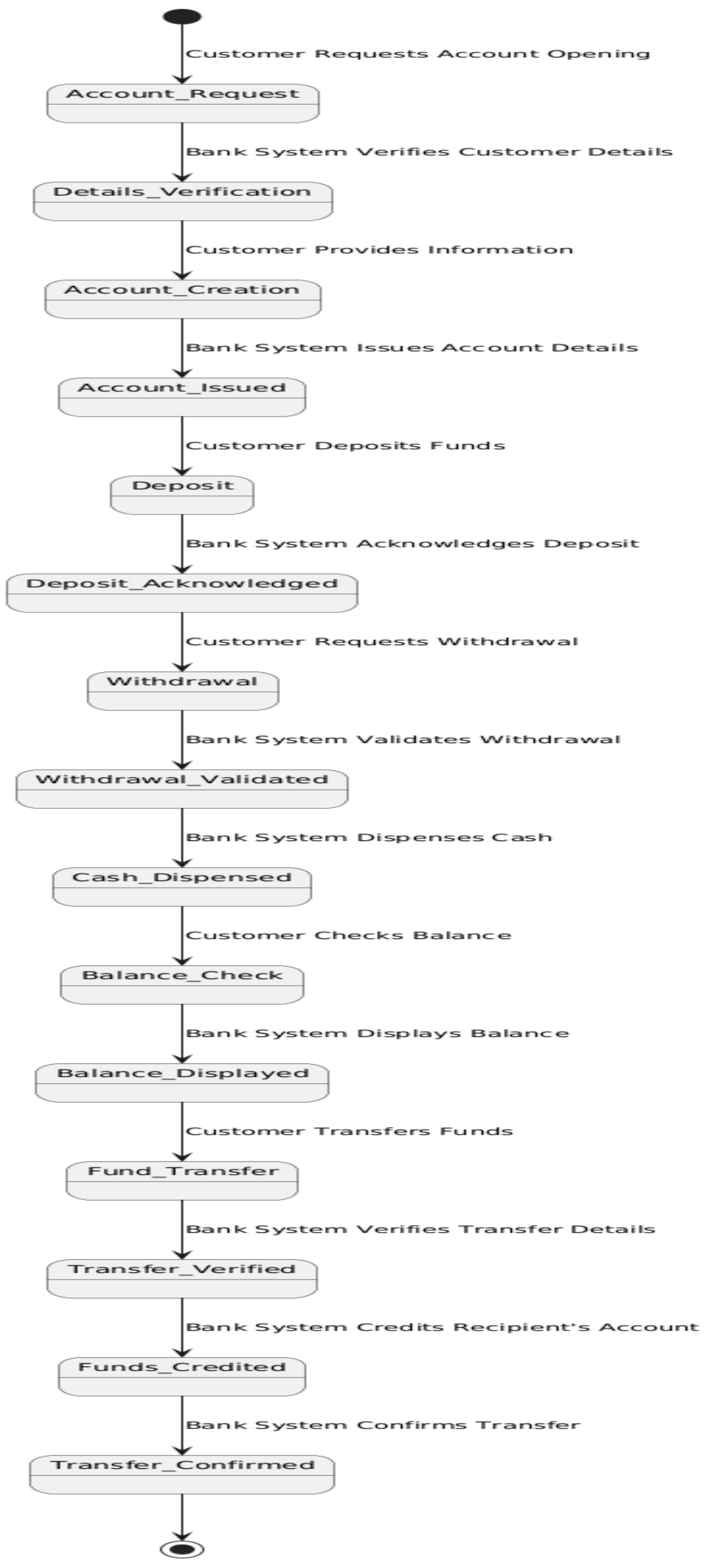
5. **Hotel Booking System**

- **Aim: To illustrate the states of a hotel booking process.**
- **Algorithm:**
    1. **Identify the states: Searching, Booked, Checked In, Checked Out, Canceled.**
    2. **Define events that cause transitions (e.g., Search Hotels, Confirm Booking, Check In, Check Out, Cancel Booking).**
    3. **Create the diagram with states represented as circles or rounded rectangles.**
    4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
    5. **Validate for accuracy and completeness.**


Customer Searches for Hotels
Search_Hotels
Booking System Displays Hotel Options
Hotel_Options_Displayed
Customer Selects Hotel and Dates
Hotel_Selection
Booking System Displays Room Availability
Room_Availability_Displayed
Customer Selects Room
Room_Selection
Booking System Displays Booking Details
Booking_Details_Displayed
Customer Confirms Booking
Booking_Confirmed
Booking System Issues Confirmation
Booking_Confirmation_Issued

6. **Banking System**

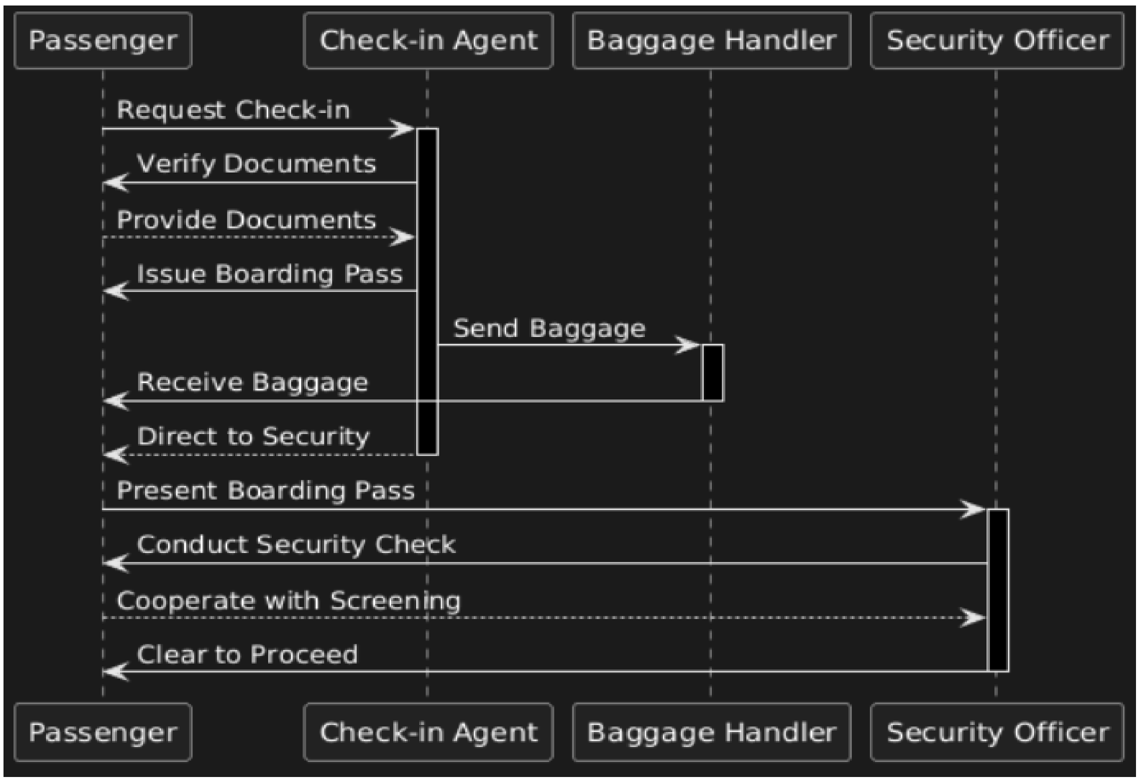
- **Aim: To depict the states of a banking transaction process.**
- **Algorithm:**
    1. **Identify the states: Logged Out, Logged In, Transaction Initiated, Transaction Completed, Transaction Failed.**
    2. **Define events that cause transitions (e.g., Log In, Initiate Transaction, Complete Transaction, Fail Transaction).**
    3. **Create the diagram with states represented as circles or rounded rectangles.**
    4. **Draw arrows to indicate transitions between states, labeling them with the events that trigger the transitions.**
    5. **Validate for clarity and completeness.**


Customer Requests Account Opening
Account_Request
Bank System Verifies Customer Details
Details_Verification
Customer Provides Information
Account_Creation
Bank System Issues Account Details
Account_Issued
Customer Deposits Funds
Deposit
Bank System Acknowledges Deposit
Deposit_Acknowledged
Customer Requests Withdrawal
Withdrawal
Bank System Validates Withdrawal
Withdrawal_Validated
Bank System Dispenses Cash
Cash_Dispensed
Customer Checks Balance
Balance_Check
Bank System Displays Balance
Balance_Displayed
Customer Transfers Funds
Fund_Transfer
Bank System Verifies Transfer Details
Transfer_Verified
Bank System Credits Recipient's Account
Funds_Credited
Bank System Confirms Transfer
Transfer_Confirmed

SEQUENCE CASE DIAGRAMS(UML)

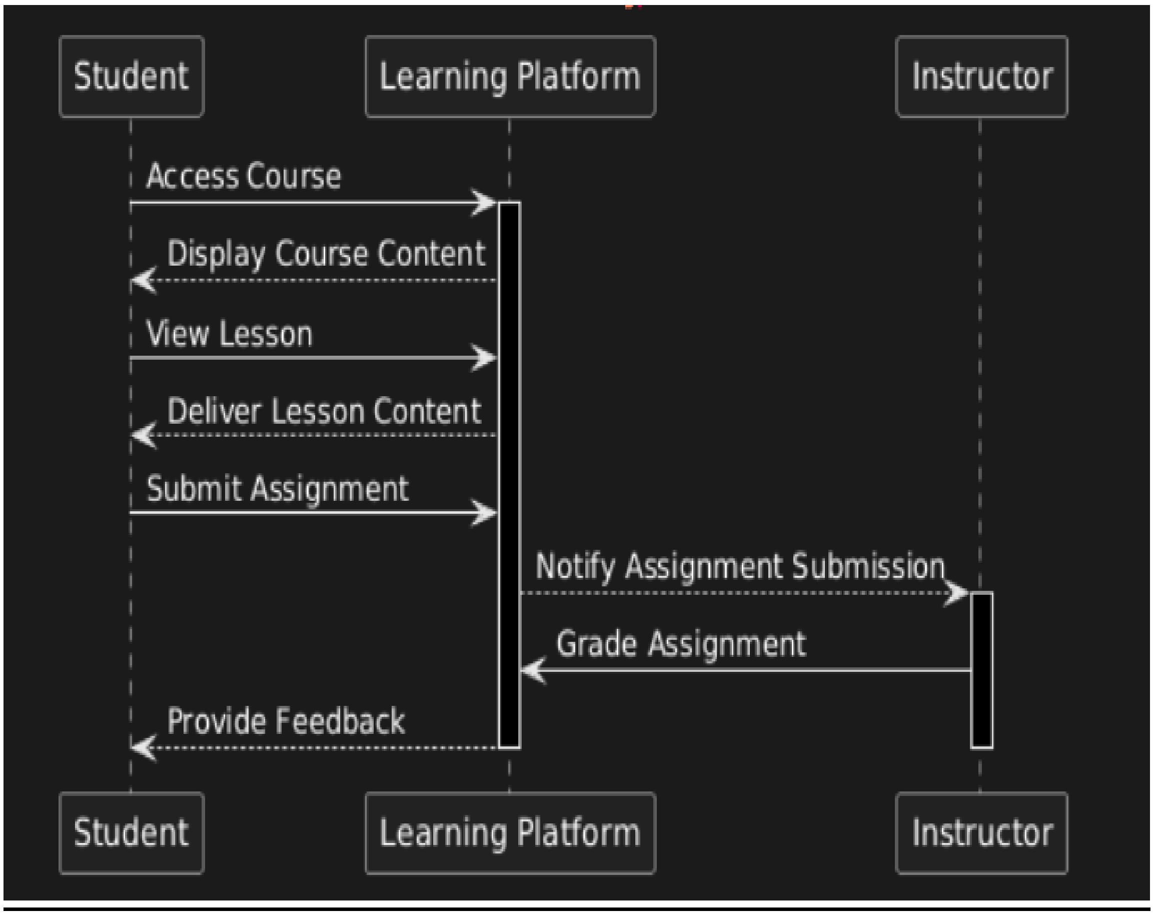
1)Airport Check-In and Security Screening:

- **Aim: To illustrate the sequence of interactions during the check-in process.**
- **Algorithm:**
  1. **Identify objects: Passenger, CheckInKiosk, AirlineStaff.**
  2. **Define the sequence of messages exchanged.**
  3. **Arrange objects horizontally and time sequence vertically.**
  4. **Draw arrows to represent messages.**
  5. **Review for accuracy.**



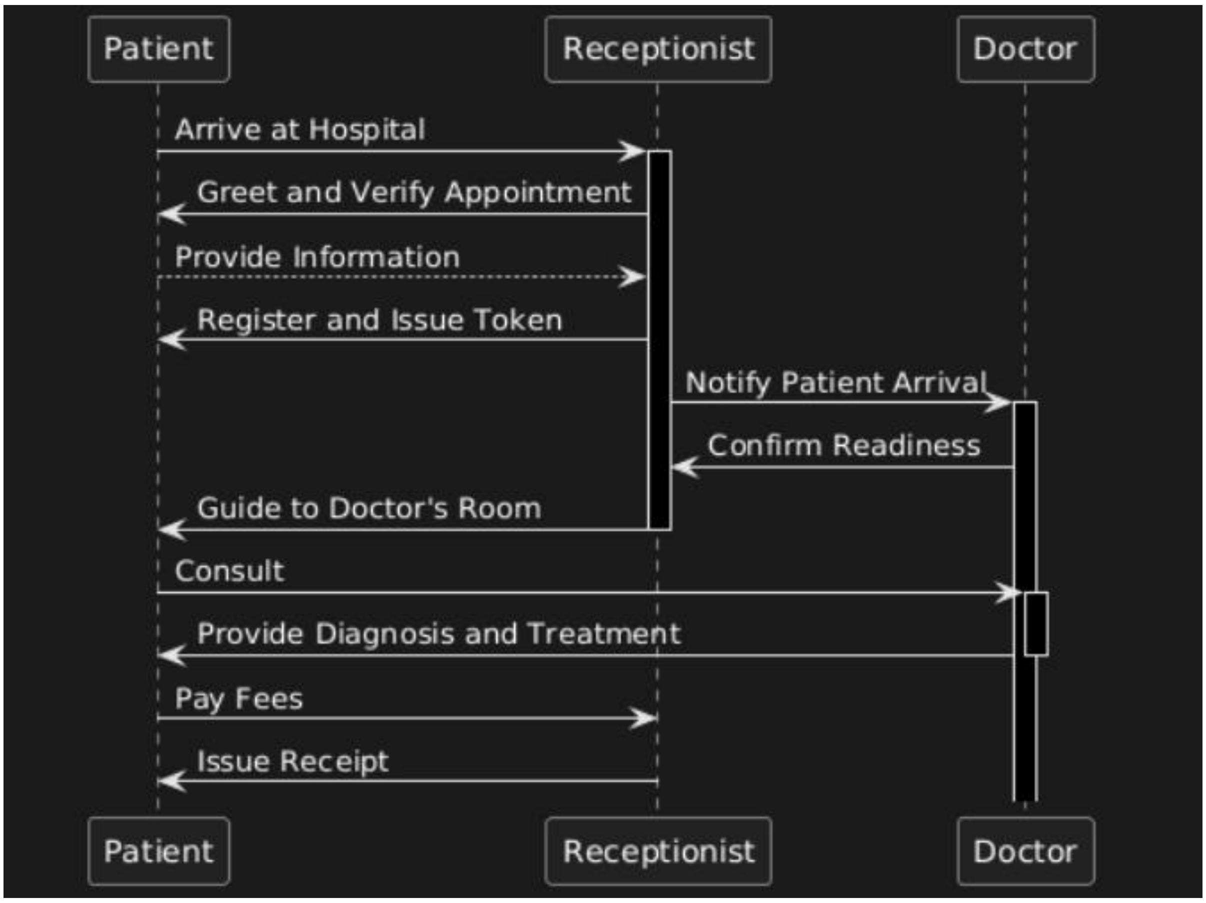
2)E-Learning Platform:

- **Aim: To depict the sequence of interactions for course enrollment.**
- **Algorithm:**
    1. **Identify objects: Student, Course, Instructor.**
    2. **Define the sequence of messages exchanged.**
    3. **Arrange objects horizontally and time sequence vertically.**
    4. **Draw arrows to represent messages.**
    5. **Review for completeness.**



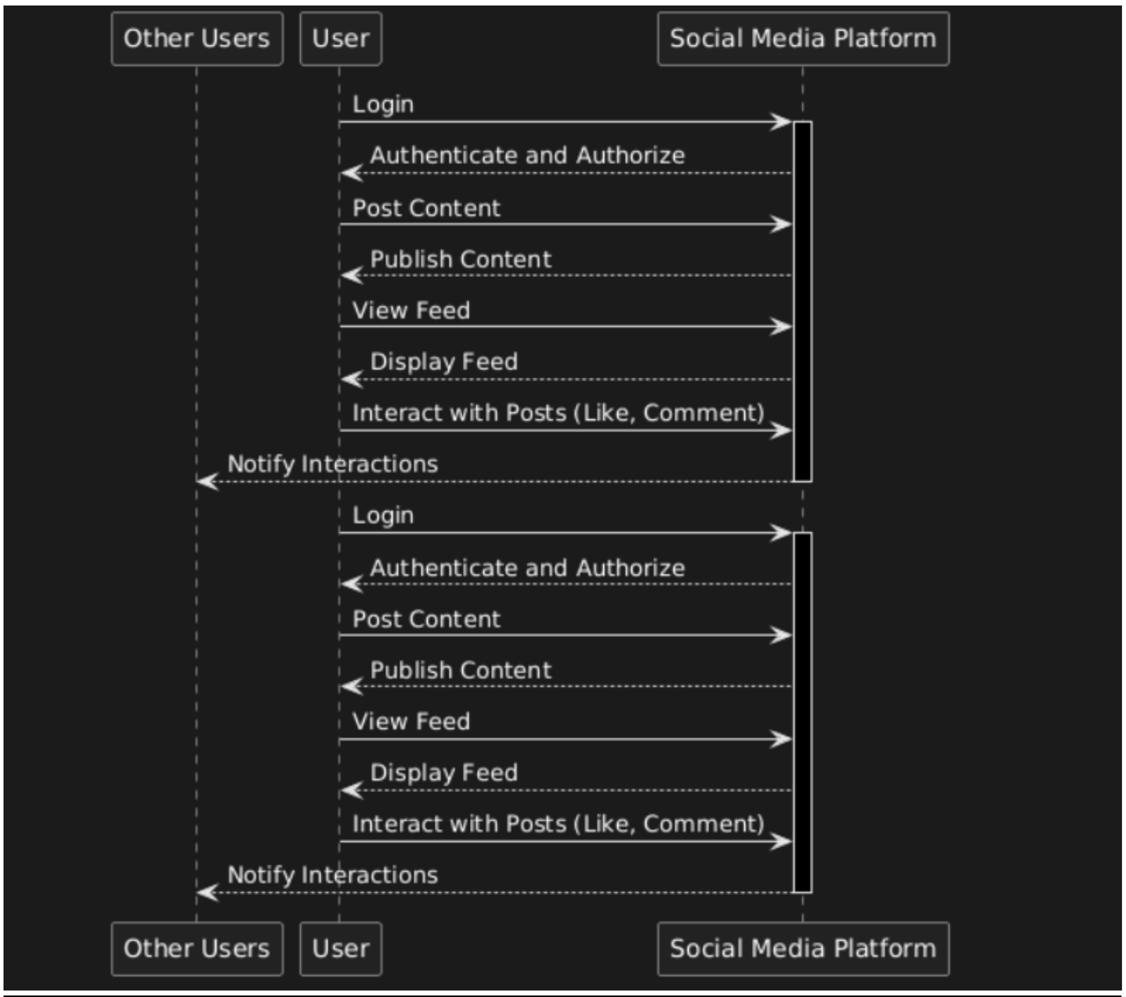
3)HOSPITAL RECIEPTIONIST AND PATIENT:

- **Aim: To illustrate the sequence of interactions during patient check-in.**
- **Algorithm:**
    1. **Identify objects: Patient, Receptionist, AppointmentSystem.**
    2. **Define the sequence of messages exchanged.**
    3. **Arrange objects horizontally and time sequence vertically.**
    4. **Draw arrows to represent messages.**
    5. **Review for accuracy.**



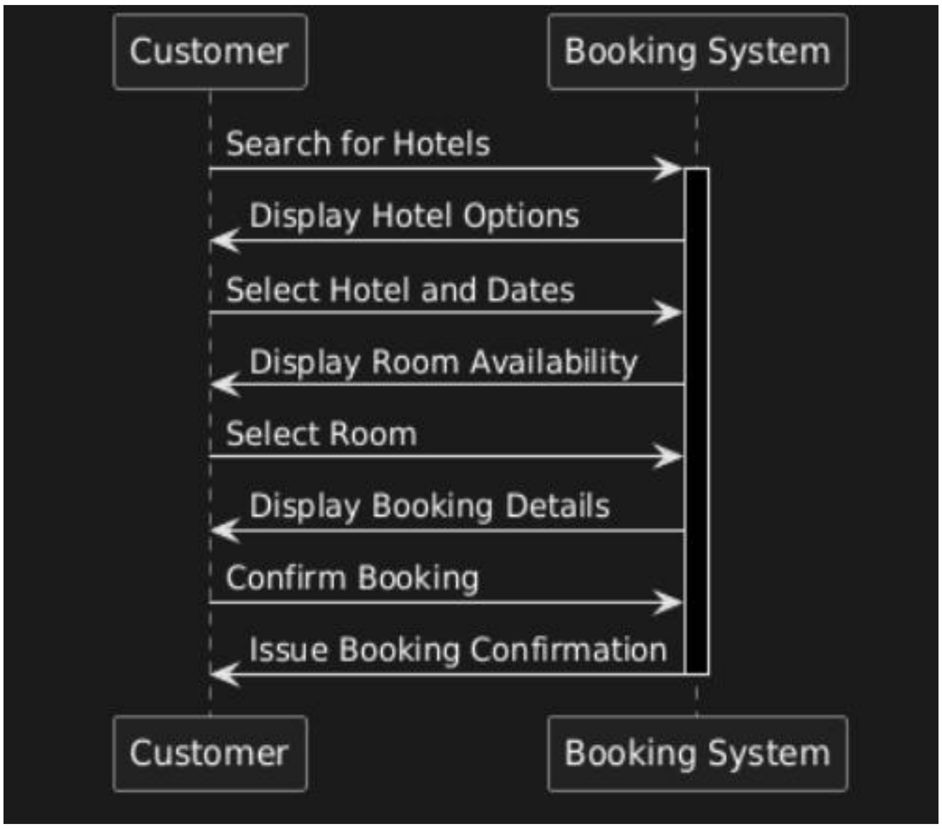
4)SOCIAL MEDIA PLATFROM:

- **Aim: To depict the sequence of interactions for posting content.**
- **Algorithm:**
  1. **Identify objects: User, Post, NotificationSystem.**
  2. **Define the sequence of messages exchanged.**
  3. **Arrange objects horizontally and time sequence vertically.**
  4. **Draw arrows to represent messages.**
  5. **Review for completeness.**



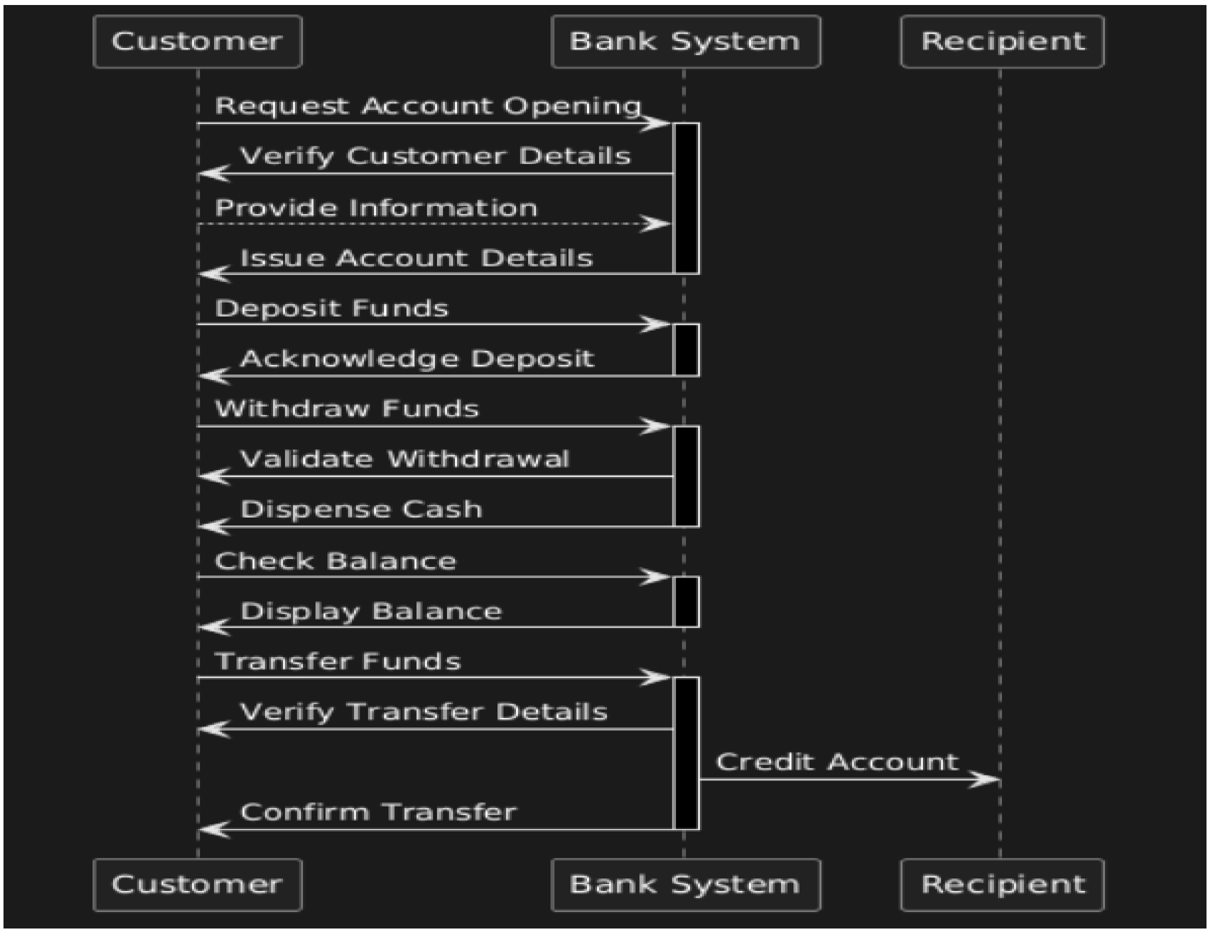
5)HOTEL BOOKING SYSTEM:

- **Aim: To illustrate the sequence of interactions during a hotel reservation.**
- **Algorithm:**
  1. **Identify objects: Customer, BookingSystem, PaymentGateway.**
  2. **Define the sequence of messages exchanged (e.g., search hotels, select hotel, make payment).**
  3. **Arrange objects horizontally and time sequence vertically.**
  4. **Draw arrows to represent messages (e.g., Customer sends a request to BookingSystem).**
  5. **Review for completeness and accuracy.**



6)BANKING SYSTEM:

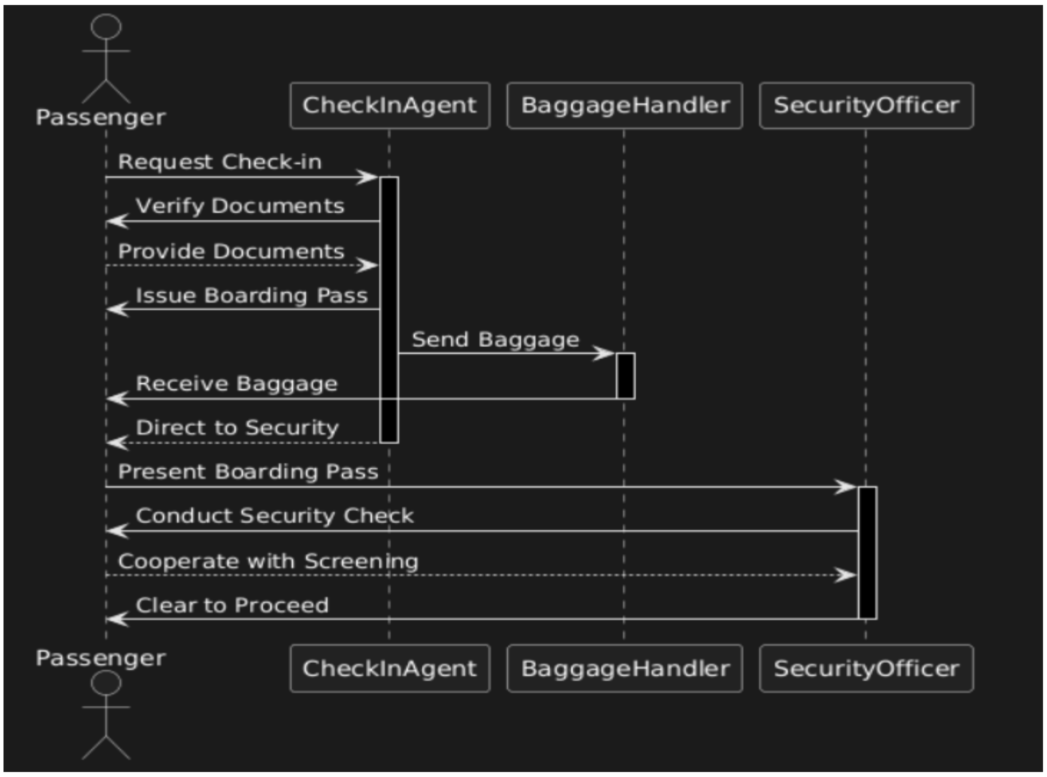
- **Aim: To depict the sequence of interactions for a banking transaction.**
- **Algorithm:**
  1. **Identify objects: Customer, BankAccount, ATM.**
  2. **Define the sequence of messages exchanged (e.g., check balance, withdraw funds).**
  3. **Arrange objects horizontally and time sequence vertically.**
  4. **Draw arrows to represent messages (e.g., Customer requests balance from BankAccount).**
  5. **Review for accuracy.**



OBJECT CASE DIAGRAMS(UML)

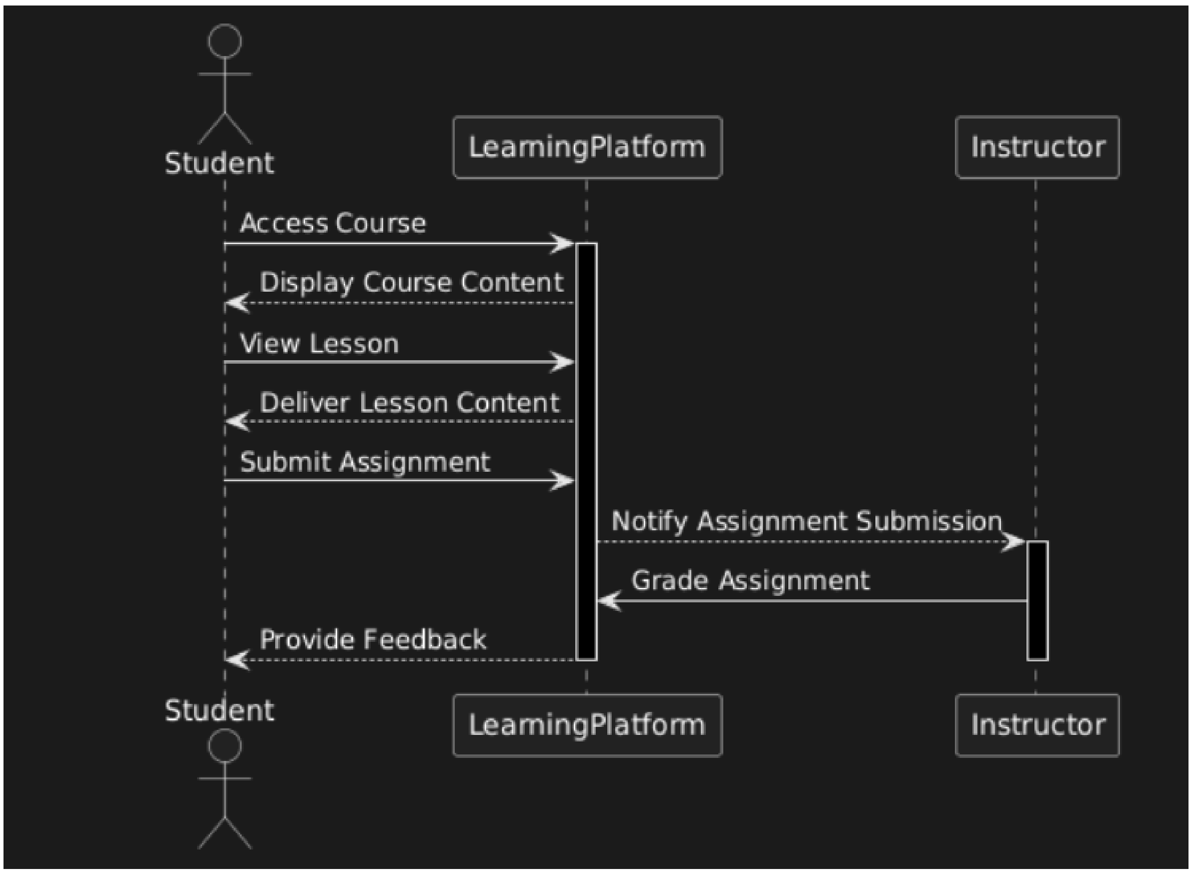
1)Airport Check-In and Security Screening:

- **Aim: To model the flow of activities during the airport check-in process.**
- **Algorithm:**
  1. **Identify main activities: arrive at airport, check in, drop baggage, go through security.**
  2. **Determine flow paths and decision points (e.g., is baggage checked?).**
  3. **Represent activities using rounded rectangles and decisions using diamonds.**
  4. **Draw arrows to indicate the flow of activities.**
  5. **Validate for clarity and completeness.**



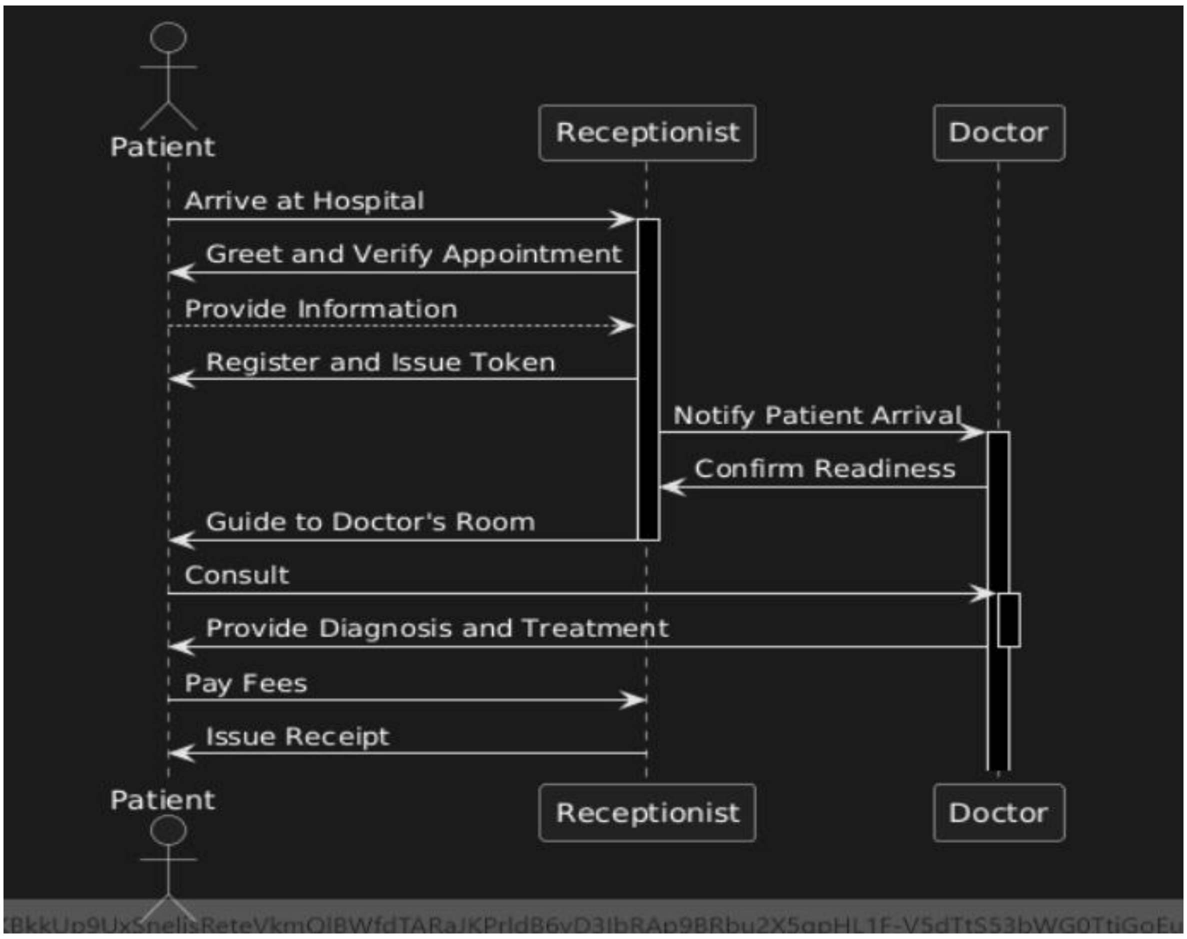
2)E-Learning Platform:

- **Aim: To illustrate the flow of activities in the e-learning platform.**
- **Algorithm:**
    1. **Identify main activities: log in, browse courses, enroll, submit assignments.**
    2. **Determine flow paths and decision points (e.g., is the assignment submitted on time?).**
    3. **Represent activities using rounded rectangles and decisions using diamonds.**
    4. **Draw arrows to indicate the flow of activities.**
    5. **Validate for accuracy.**



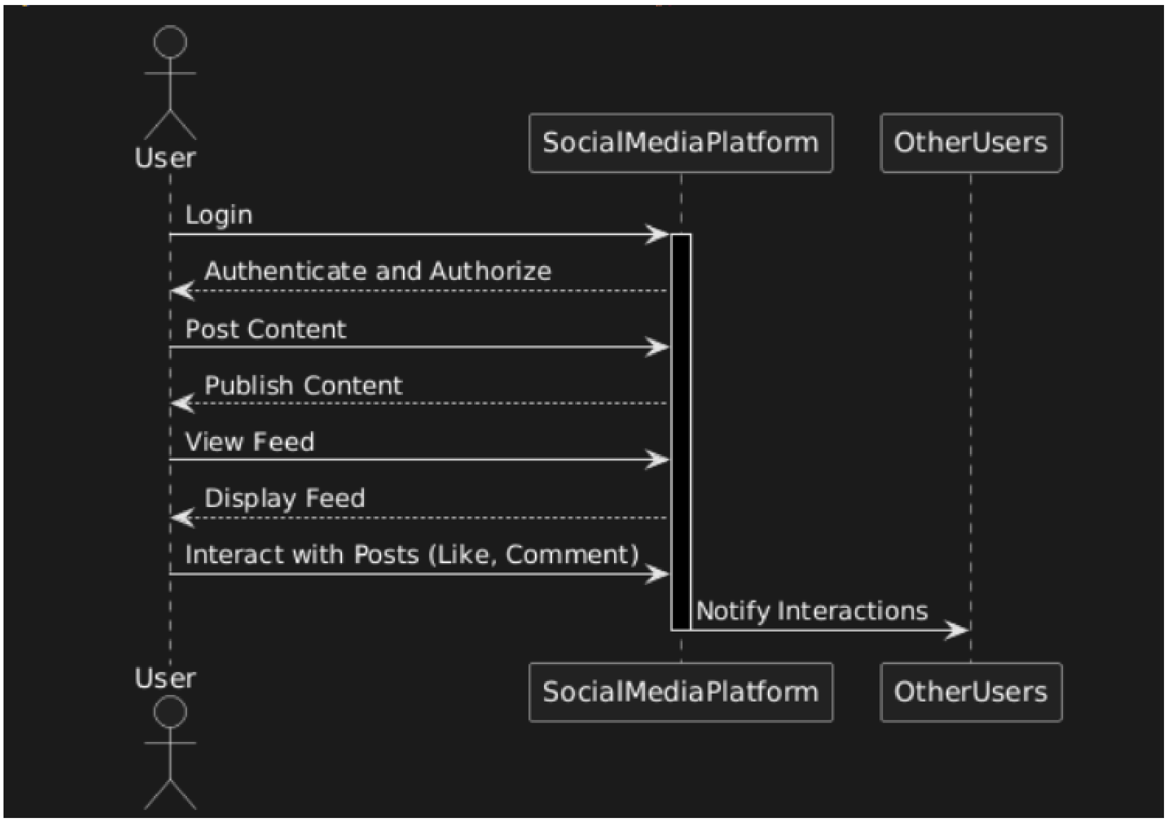
3)HOSPITAL RECIEPTIONIST AND PATIENT:

- **Aim: To model the flow of activities during patient check-in at a hospital.**
- **Algorithm:**
    1. **Identify main activities: patient arrives, check-in, verify information, schedule appointment.**
    2. **Determine flow paths and decision points (e.g., is the patient a new or returning patient?).**
    3. **Represent activities using rounded rectangles and decisions using diamonds.**
    4. **Draw arrows to indicate the flow of activities.**
    5. **Validate for clarity.**



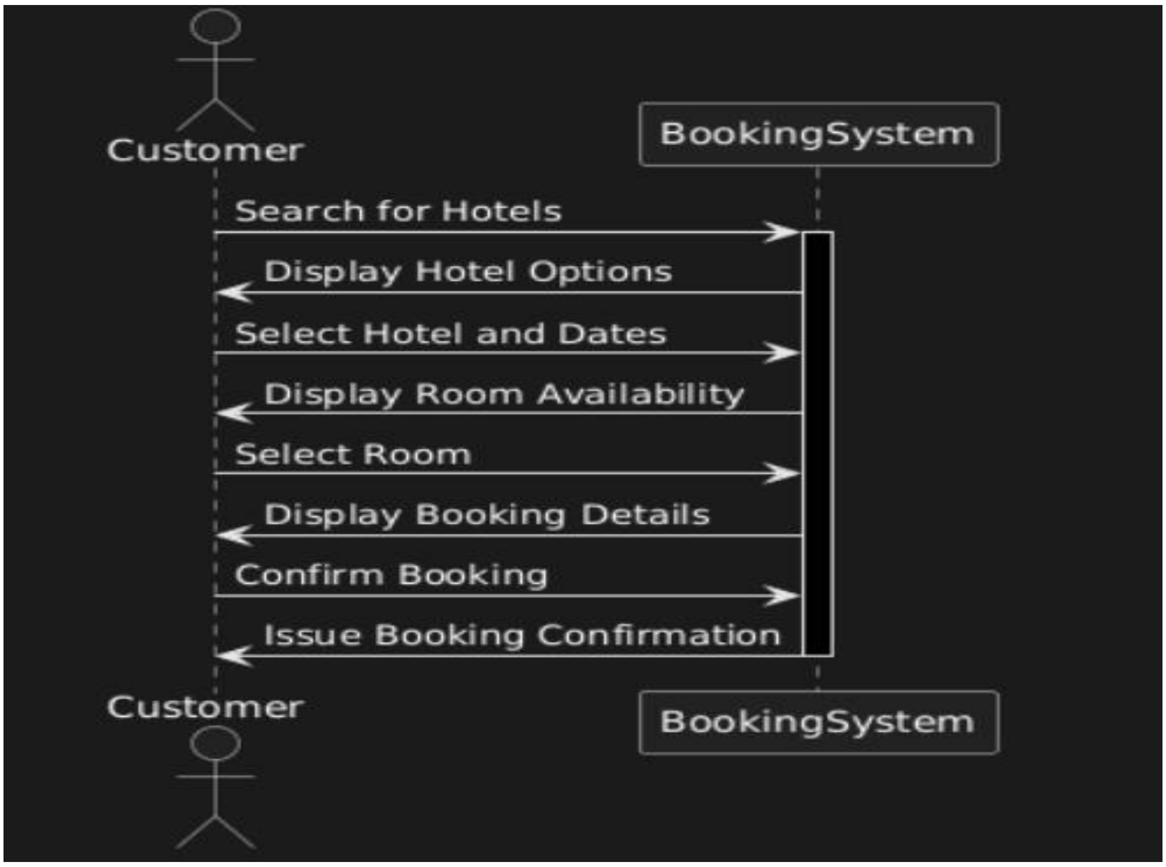
4)SOCIAL MEDIA PLATFROM:

- **Aim: To illustrate the flow of activities on a social media platform.**
- **Algorithm:**
    1. **Identify main activities: log in, create post, like post, comment on post.**
    2. **Determine flow paths and decision points (e.g., is the post public or private?).**
    3. **Represent activities using rounded rectangles and decisions using diamonds.**
    4. **Draw arrows to indicate the flow of activities.**
    5. **Validate for completeness.**



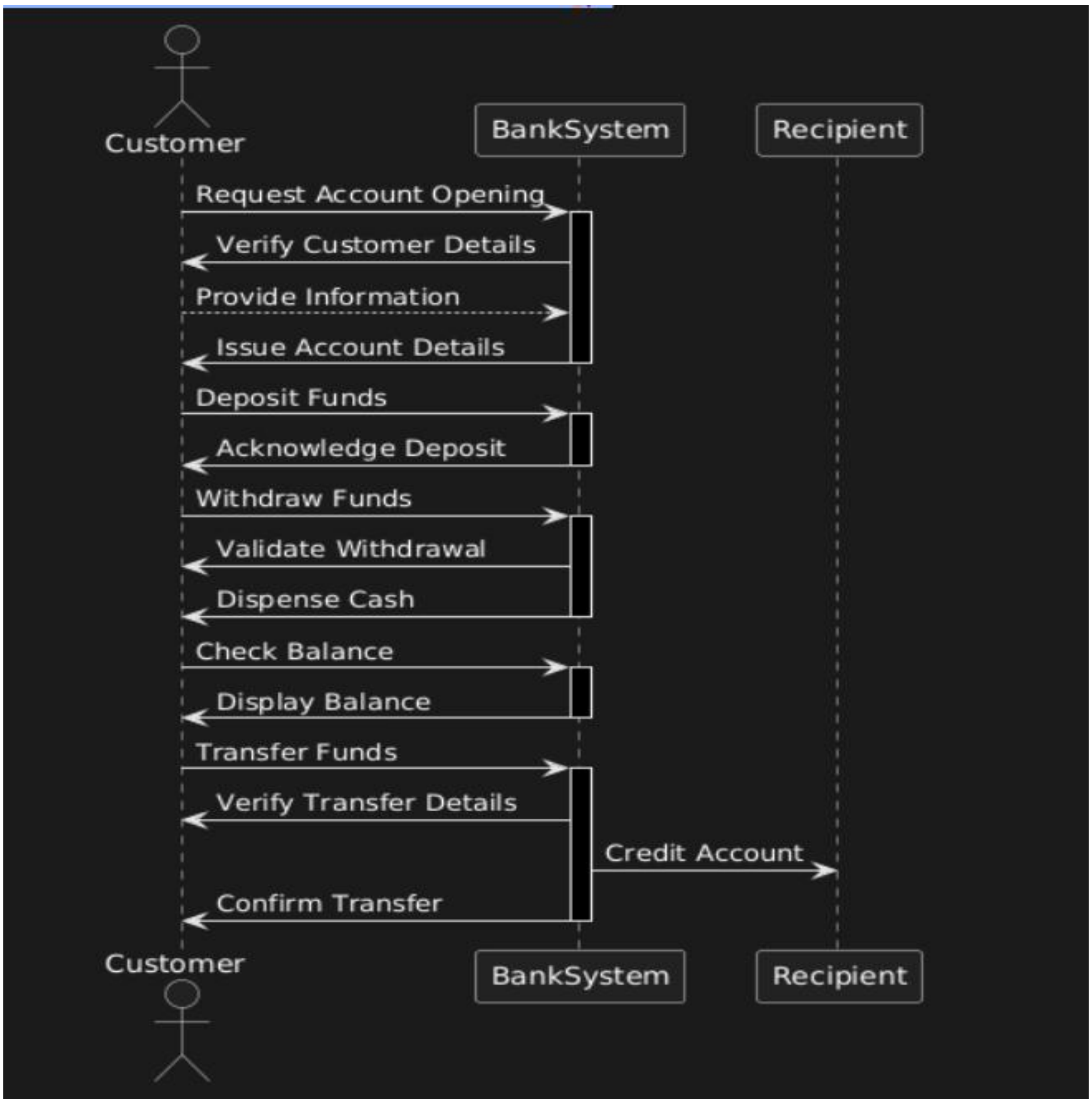
5)HOTEL BOOKING SYSTEM:

- **Aim: To model the flow of activities during the hotel booking process.**
- **Algorithm:**
  1. **Identify main activities: search for hotels, select hotel, enter guest information, confirm booking.**
  2. **Determine flow paths and decision points (e.g., is payment successful?).**
  3. **Represent activities using rounded rectangles and decisions using diamonds.**
  4. **Draw arrows to indicate the flow of activities.**
  5. **Validate for accuracy.**



6)BANKING SYSTEM:

- **Aim: To illustrate the flow of activities in a banking transaction.**
- **Algorithm:**
  1. **Identify main activities: log in, check balance, withdraw funds, confirm transaction.**
  2. **Determine flow paths and decision points (e.g., is there sufficient balance?).**
  3. **Represent activities using rounded rectangles and decisions using diamonds.**
  4. **Draw arrows to indicate the flow of activities.**
  5. **Validate for clarity and completeness.**



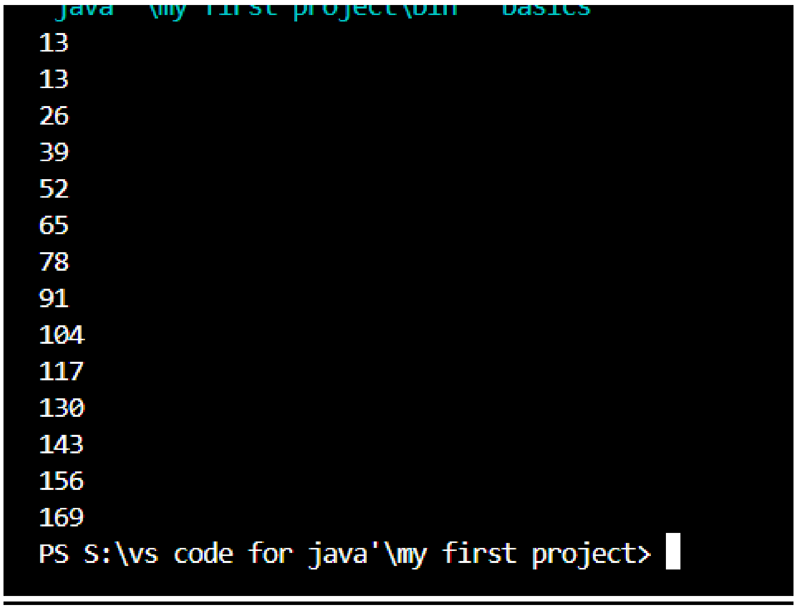
JAVA BASICS PROGRAMS:

1)Multiplying a number till 10:

PROGRAM:

```
import java.util.*;
public class basics {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int i = 1;
        while (i <= a) {
            System.out.println(i * a);
            i++;
        }
    }
}
```

OUTPUT:



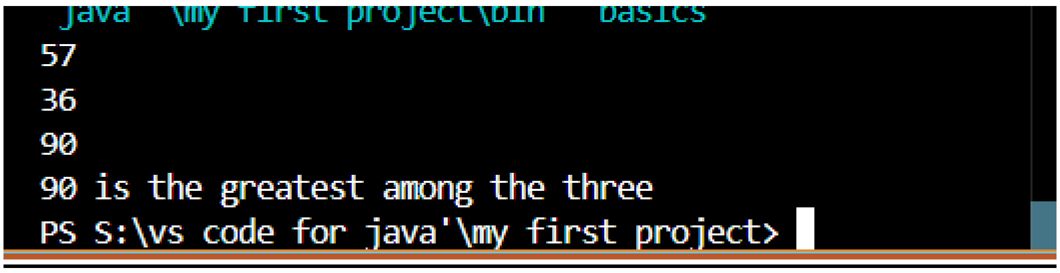
```
  java' '\my first project\bin' basics
13
13
26
39
52
65
78
91
104
117
130
143
156
169
PS S:\vs code for java'\my first project>
```

2)Checking which the largest number from three:

PROGRAM:

```
import java.util.*;
public class basics {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();
        if(a > b && a > c){
            System.out.println(a + " is the greatest among the three");
        } else if(b > a && b > c){
            System.out.println(b + " is the greatest among the three ");
        } else {
            System.out.println(c + " is the greatest among the three");
        }
    }
}
```

OUTPUT:



```
 java  \my first project\bin  basics
57
36
90
90 is the greatest among the three
PS S:\vs code for java'\my first project>
```

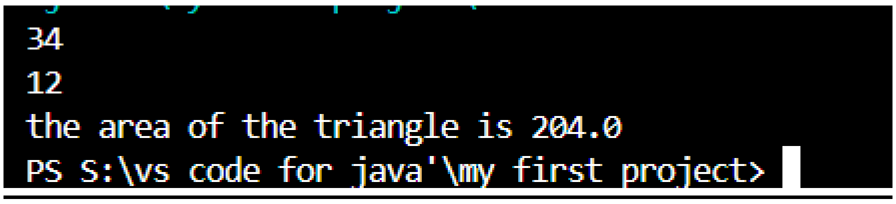
3)Calculating the area of a triangle:

PROGRAM:

```
import java.util.*;
```

```
public class basics{
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      int b = sc.nextInt();
      int h = sc.nextInt();
      double c = (0.5 * b * h);
      System.out.println("the area of the triangle is " + c);
      }
   }
```

OUTPUT:



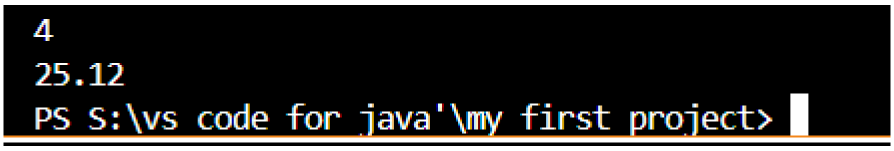
```
34
12
the area of the triangle is 204.0
PS S:\vs code for java'\my first project>
```

4)Calculating the area of a circle:

PROGRAM:

```
import java.util.*;
public class basics{
   public static Double getCircumference(Double r){
      return 2 * 3.14 * r;
   }
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      Double r = sc.nextDouble();
      System.out.println(getCircumference(r));
   }
}
```

OUTPUT:



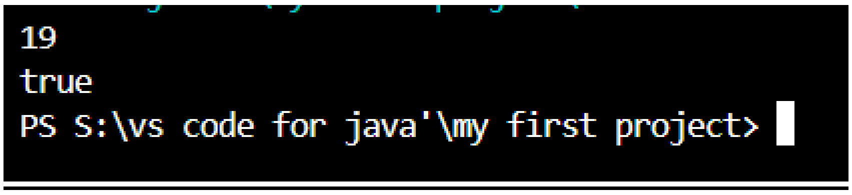
```
4
25.12
PS S:\vs code for java'\my first project>
```

5)Checking whether the person is eligible for voting or not:

PROGRAM:

```
import java.util.*;
public class basics{
   public static boolean isElligible(int age){
      if(age >= 18){
         return true;
      } else {
         return false;
      }
   }
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      int age = sc.nextInt();
      System.out.println(isElligible(age));

   }
}
```

OUTPUT:



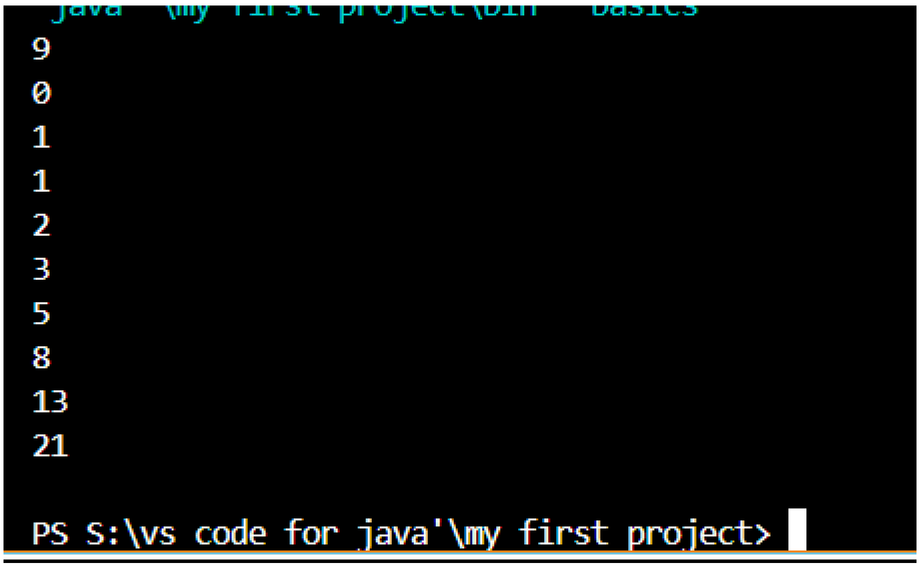
```
19
true
PS S:\vs code for java'\my first project>
```

6)Fibonacci series:

PROGRAM:

```
import java.util.*;
public class basics{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int a = 0;
        int b = 1;
        System.out.println(a + " ");
        if(n > 1){
            for(int i = 2; i <= n; i++){
                System.out.println(b + " ");
                int temp = b;
                b = a + b;
                a = temp;
            }
            System.out.println();
        }
    }
}
```

OUTPUT:



```
 java'   \my first project\bin'  basics
9
0
1
1
2
3
5
8
13
21

PS S:\vs code for java'\my first project>
```

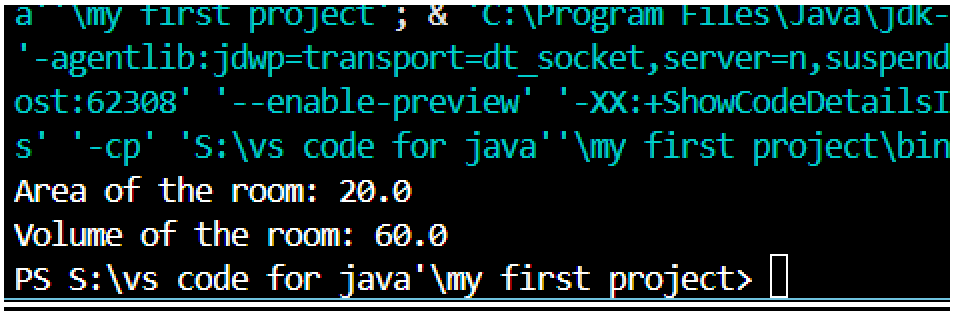
7)Area and Volume of a Room (Single Inheritance)

PROGRAM:

```
class Room {
   double length;
   double width;
   double height;

   double calculateArea() {
      return length * width;
   }
}

class VolumeRoom extends Room {
   double calculateVolume() {
      return length * width * height;
   }
}
```

```
public class college {
    public static void main(String[] args) {
        VolumeRoom room = new VolumeRoom();
        room.length = 5;
        room.width = 4;
        room.height = 3;

        System.out.println("Area of the room: " + room.calculateArea());
        System.out.println("Volume of the room: " + room.calculateVolume());
    }
}
```

OUTPUT:



```
a''\my first project'; & 'C:\Program Files\Java\jdk-
'-agentlib:jdwp=transport=dt_socket,server=n,suspend
ost:62308' '--enable-preview' '-XX:+ShowCodeDetailsI
s' '-cp' 'S:\vs code for java''\my first project\bin
Area of the room: 20.0
Volume of the room: 60.0
PS S:\vs code for java'\my first project> 
```

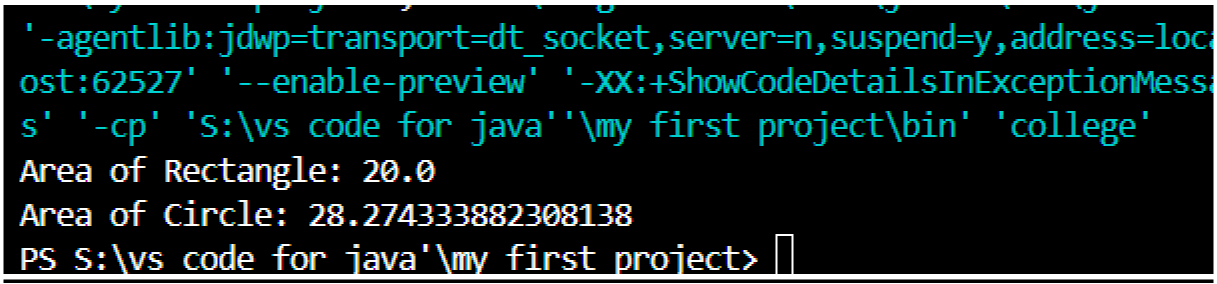
8)Hierarchical Inheritance (Shape, Rectangle, Circle)

PROGRAM:

```
class Shape {
    double area;

    void calculateArea() {
        // Default implementation
    }
```

```
}

class Rectangle extends Shape {
    double length;
    double width;

    @Override
    void calculateArea() {
        area = length * width;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Circle extends Shape {
    double radius;

    @Override
    void calculateArea() {
        area = Math.PI * radius * radius;
        System.out.println("Area of Circle: " + area);
    }
}

public class college {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle();
        rectangle.length = 5;
        rectangle.width = 4;
        rectangle.calculateArea();

        Circle circle = new Circle();
        circle.radius = 3;
        circle.calculateArea();
    }
}
```

OUTPUT:



```
'-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=loca
ost:62527' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMess
s' '-cp' 'S:\vs code for java''\my first project\bin' 'college'
Area of Rectangle: 20.0
Area of Circle: 28.274333882308138
PS S:\vs code for java'\my first project>
```

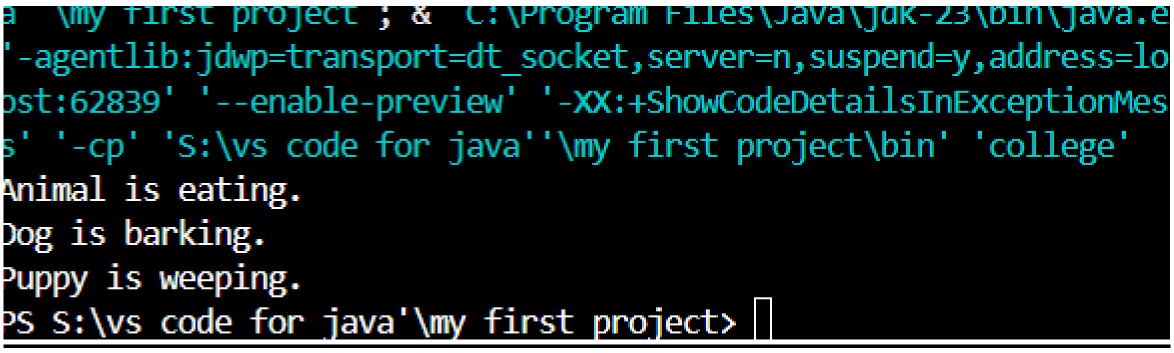
9)Multilevel Inheritance Example

PROGRAM:

```
class Animal {
    void eat() {
        System.out.println("Animal is eating.");
    }
}

class Dog extends Animal {
    void bark() {
        System.out.println("Dog is barking.");
    }
}
```

```
class Puppy extends Dog {
    void weep() {
        System.out.println("Puppy is weeping.");
    }
}

public class Main {
    public static void main(String[] args) {
        Puppy puppy = new Puppy();
        puppy.eat();
        puppy.bark();
        puppy.weep();
    }
}
```

OUTPUT:



```
a  \my first project'; & 'C:\Program Files\Java\jdk-23\bin\java.e
'-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=lo
ost:62839' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMes
s' '-cp' 'S:\vs code for java''\my first project\bin' 'college'
Animal is eating.
Dog is barking.
Puppy is weeping.
PS S:\vs code for java'\my first project>
```

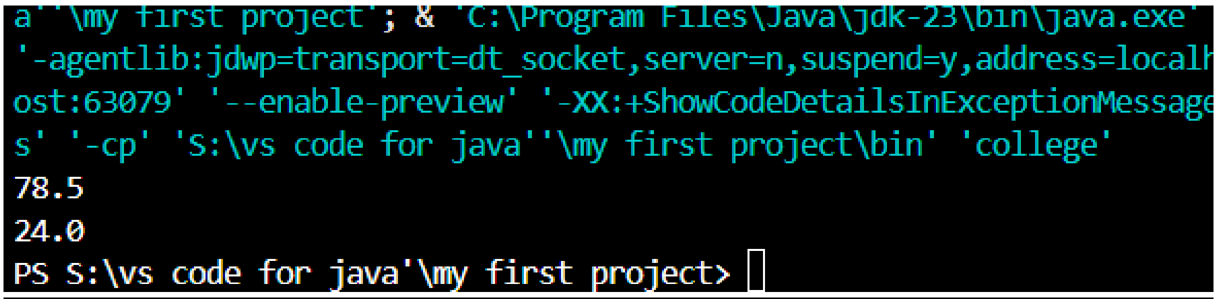
10)Shape Class with CalculateArea Method

PROGRAM:

```
abstract class shape{
   abstract double calculatearea();
}

class circle extends shape{
   double radius;
   circle(double radius){
      this.radius = radius;
   }
   @Override
   double calculatearea(){
      return 3.14 * radius * radius;
   }
}

class rectangle extends shape{
   double length;
   double breadth;
   rectangle(double length,double breadth){
      this.length = length;
      this.breadth = breadth;
   }
   @Override
   double calculatearea(){
      return length * breadth;
```

```
    }
}
public class college{
    public static void main(String[] args){
        shape mycircle = new circle(5);
        shape myrectangle = new rectangle(4,6);
        System.out.println(mycircle.calculatearea());
        System.out.println(myrectangle.calculatearea());
    }
}
```

OUTPUT:



```
a''\my first project'; & 'C:\Program Files\Java\jdk-23\bin\java.exe'
'-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localh
ost:63079' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessage
s' '-cp' 'S:\vs code for java''\my first project\bin' 'college'
78.5
24.0
PS S:\vs code for java'\my first project>
```

CH.SC.U4CSE24039

SAHIL PAREEK