Sukkur Institute of Business Administration University
Department of Computer Science

Object Oriented Programming

BS – II (CS/SE/AI)
Spring 2025

Lab # 07: Team-Based OOP Development: Ride-Sharing & Event Ticketing System

Instructor: Moona Solangi

Lab Report Rubrics (Add the points in each column, then add across the bottom row to find the total score)					**Total Marks**
S.No	**Criterion**	**0.5**	**0.25**	**0.125**	
1	Accuracy	☐ Desired output	☐ Minor mistakes	☐ Critical mistakes	
2	Timing	☐ Submitted within the given time	☐ 1 day late	☐ More than 1 day late	

Submission Profile

Name: Submission date (dd/mm/yy):
Enrollment ID: Receiving authority name and signature:
Comments:

Instructor Signature

Note: Submit this lab hand-out before the next lab with attached solved activities and exercises

Objectives

After performing this lab, students will be able to understand,

- ✅ Understand and implement **Classes & Objects** in Java.
- ✅ Learn **Constructor Initialization** and **Encapsulation**.
- ✅ Use **Method Overloading** to handle different cases.
- ✅ Work with **Object Passing & Aggregation**.
- ✅ Apply **Decision Making** and **Logical Flow Control** in Java.
- ✅ Gain experience in **Group Collaboration** on OOP Projects.

📌 Lab Instructions

1 Students should read the problem statements carefully and **discuss their approach** within the group **(5-10 mins)**.
2 Each group **develops their solution** collaboratively **(40-45 mins)**.
3 Each group **presents** their code **(10-15 mins)**:

- Explains the **logic and OOP concepts used**.
- Demonstrates **working output**.
- Discusses **challenges faced**.

📝 Task 1: Ride-Sharing System (Intermediate-Hard Level)

🚗 Scenario

A ride-sharing company wants to develop a **basic ride-booking system** where:

- **Drivers** have different car types (Economy, Business, Luxury).
- **Customers** request a ride and select their preferred car type.
- **The system assigns** a driver based on customer preference.
- Customers can **view the estimated price** and **confirm booking**.

📌 Task Requirements

1 **Create a class Driver** with:

- Attributes: name, carType, availability (boolean), baseFarePerKM.
- Constructor to initialize driver details.

2 **Create a class Customer** with:

- Attributes: name, pickupLocation, destination, rideDistance, preferredCarType.
- Constructor to initialize customer details.

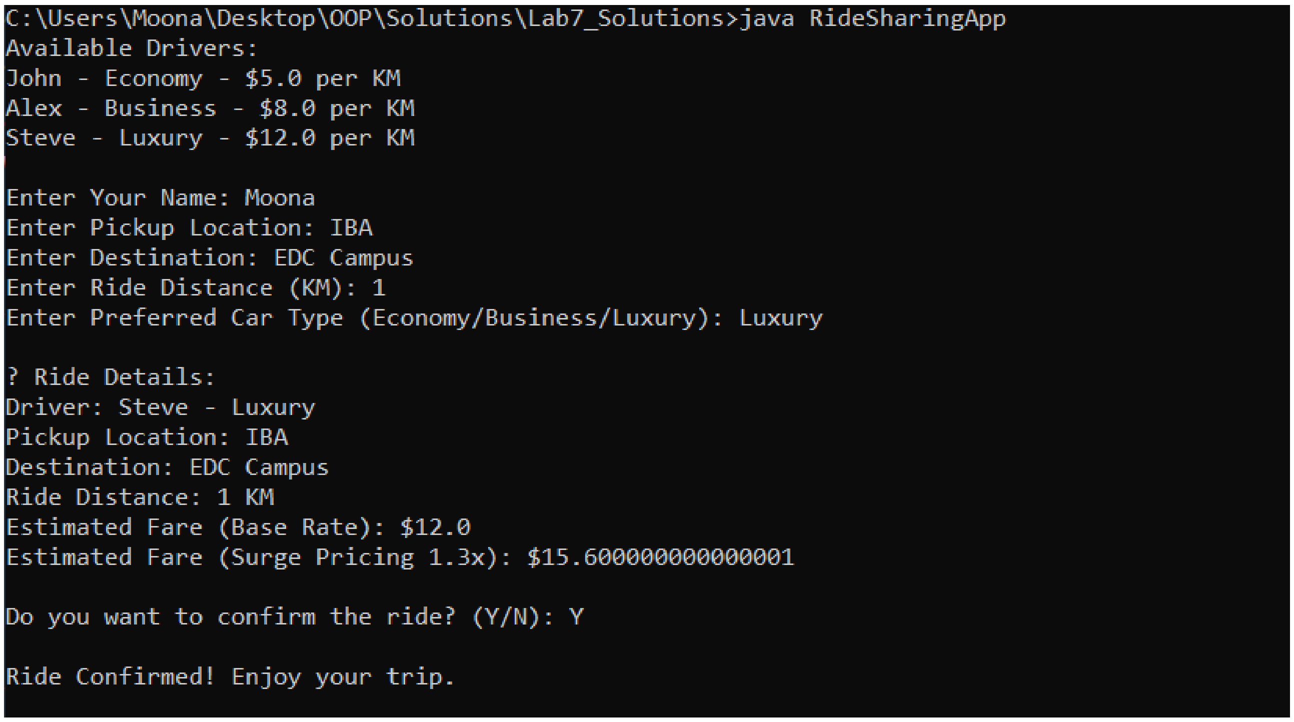
3 **Create a class Ride** with:

- Attributes: Driver, Customer, totalFare.
- **Method Overloading**:
    - calculateFare(int distance) – Uses **default base fare**.
    - calculateFare(int distance, double surgeMultiplier) – Applies **surge pricing**.

4 **In main(), implement:**

- Create **3 drivers** with different car types (Economy, Business, Luxury).
- Allow the **customer to request a ride and select a preferred car type**.
- Assign the **first available driver** of that car type.
- If no preferred car type is available, **assign the next available driver.**
- Calculate **fare with/without surge pricing**.
- Display **ride details** and ask the customer if they want to confirm.

📌 Expected Output Example



```
C:\Users\Moona\Desktop\OOP\Solutions\Lab7_Solutions>java RideSharingApp
Available Drivers:
John - Economy - $5.0 per KM
Alex - Business - $8.0 per KM
Steve - Luxury - $12.0 per KM

Enter Your Name: Moona
Enter Pickup Location: IBA
Enter Destination: EDC Campus
Enter Ride Distance (KM): 1
Enter Preferred Car Type (Economy/Business/Luxury): Luxury

? Ride Details:
Driver: Steve - Luxury
Pickup Location: IBA
Destination: EDC Campus
Ride Distance: 1 KM
Estimated Fare (Base Rate): $12.0
Estimated Fare (Surge Pricing 1.3x): $15.600000000000001

Do you want to confirm the ride? (Y/N): Y

Ride Confirmed! Enjoy your trip.
```

✅ OOP Concepts Covered in Task 1:

✓ **Classes & Objects** (Driver, Customer, Ride).
✓ **Constructors** (Initializing driver/customer details).
✓ **Method Overloading** (calculateFare() with and without surge pricing).
✓ **Object Passing** (Passing Driver and Customer objects to Ride).
✓ **Decision Making** (Choosing the first available driver).

📝 Task 2: Event Ticketing System 🎟

🏢 Scenario

This system will allow users to **book event tickets** for different event categories with dynamic pricing.

📌 Task Requirements

1 **Create a class Event** with:

- Attributes: eventName, eventType (Concert, Sports, Theatre), ticketPrice, availableSeats.
- Constructor to initialize event details.
- Method bookTicket(int quantity):
    - Reduces available seats and calculates total price.
    - If seats are unavailable, displays a message.

2 **Create a class Customer** with:

- Attributes: name, email, selectedEvent.
- Constructor to initialize customer details.

3 **Create a class Booking** with:

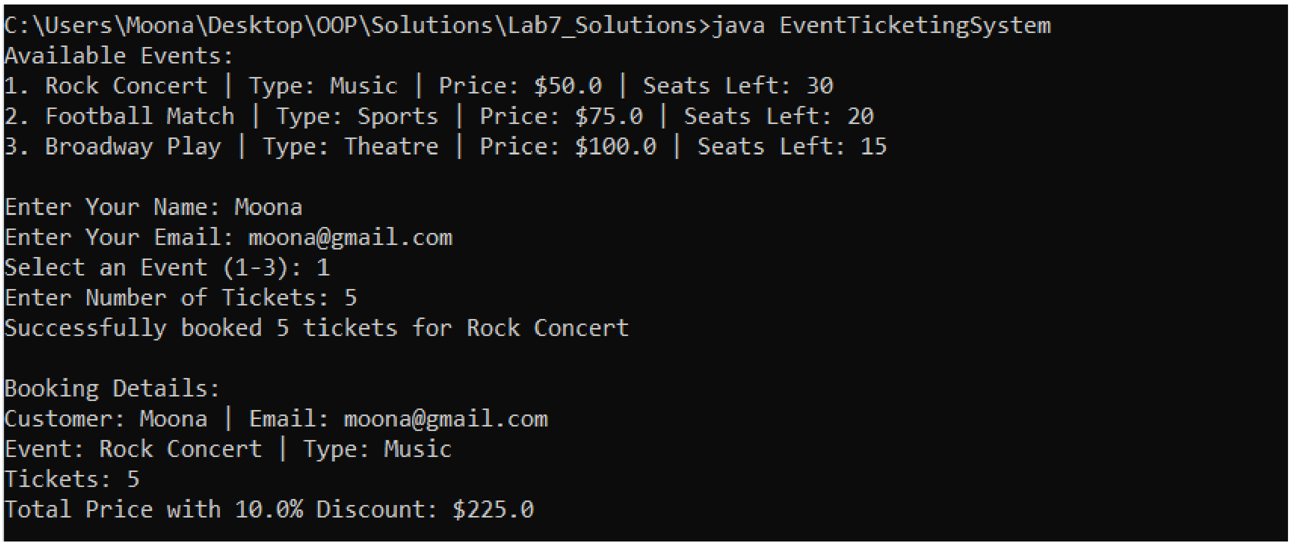
- Attributes: Customer, Event, numTickets, totalPrice.
- Method Overloading:
    - calculateTotal(int numTickets) – Standard booking price.
    - calculateTotal(int numTickets, double discountRate) – Applies a discount.

4 **In main()**, implement:

- Display **3 different events** (Concert, Sports, Theatre).
- Allow the **customer to choose an event** and the number of tickets.
- **Check availability** and calculate total cost.
- **Apply discounts** for bulk bookings (e.g., 10% off for 5+ tickets).

- Confirm or cancel the booking.

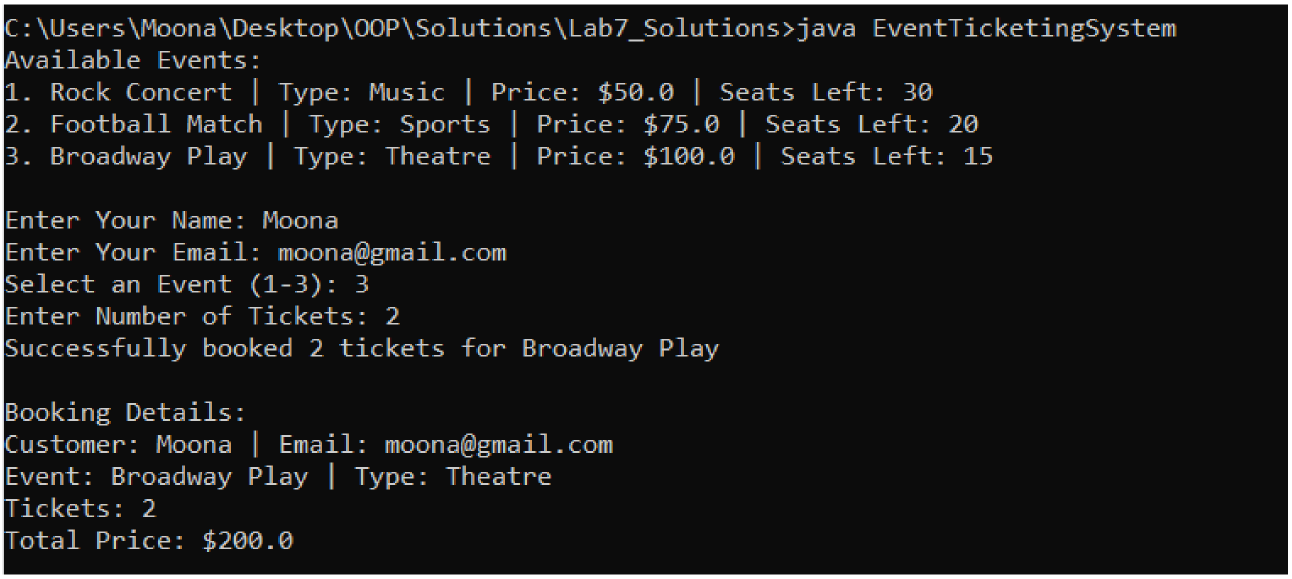
📌 **Expected Output Example**



```
C:\Users\Moona\Desktop\OOP\Solutions\Lab7_Solutions>java EventTicketingSystem
Available Events:
1. Rock Concert | Type: Music | Price: $50.0 | Seats Left: 30
2. Football Match | Type: Sports | Price: $75.0 | Seats Left: 20
3. Broadway Play | Type: Theatre | Price: $100.0 | Seats Left: 15

Enter Your Name: Moona
Enter Your Email: moona@gmail.com
Select an Event (1-3): 1
Enter Number of Tickets: 5
Successfully booked 5 tickets for Rock Concert

Booking Details:
Customer: Moona | Email: moona@gmail.com
Event: Rock Concert | Type: Music
Tickets: 5
Total Price with 10.0% Discount: $225.0
```



```
C:\Users\Moona\Desktop\OOP\Solutions\Lab7_Solutions>java EventTicketingSystem
Available Events:
1. Rock Concert | Type: Music | Price: $50.0 | Seats Left: 30
2. Football Match | Type: Sports | Price: $75.0 | Seats Left: 20
3. Broadway Play | Type: Theatre | Price: $100.0 | Seats Left: 15

Enter Your Name: Moona
Enter Your Email: moona@gmail.com
Select an Event (1-3): 3
Enter Number of Tickets: 2
Successfully booked 2 tickets for Broadway Play

Booking Details:
Customer: Moona | Email: moona@gmail.com
Event: Broadway Play | Type: Theatre
Tickets: 2
Total Price: $200.0
```

✅ **OOP Concepts Covered in Task 2**

✔ **Classes & Objects** (Event, Customer, Booking).
✔ **Encapsulation** (customer/event details stored privately).
✔ **Method Overloading** (calculateTotal() for pricing with and without discounts).
✔ **Decision Making** (ticket availability & discount application).
✔ **Real-World Ticket Booking Experience.**