

## **Courier Management System (Ashutosh Aswani)**

### **Index :**

<b>Topic</b>	<b>Page Number</b>
Index	1
Abstract	2
Software Used	-
Programming Language Used	-
OOPs concepts used	3
Issues faced	4
Screenshots of our work	5
UML Diagram	10

**Abstract:**

Our project is basically a courier system and we have two entities in this courier system. When we see the login panel, we see there is an option for admin and another one for the customer in the login panel. The admin can actually change the details of the delivery executives and also update the delivery related details like delivery person name and expected delivery date for each and every order ID and in the customer panel, we have different options like track order, wallet, my orders, help, and pick up also. In the track order panel we can enter the courier ID and track our courier, in my order section we can enter our name and see all the courier orders placed by us, then in the wallet, we can add or withdraw money and when we place an order for pick up of a courier the money is directly withdrawn from the wallet itself and in the pickup panel, we can place a request to pick up the parcel from a particular address and deliver it to a particular address.

**Software Used:**

MySQL , Netbeans IDE

**Programming Language Used:**

Java

## **OOPs concepts used:**

### **1. Method Overloading**

Method overloading has been used while taking the wallet initial balance when a new customer registers and thus there are two methods which are overloaded and one of them takes 4 parameters and the other takes 5 parameters. If the wallet balance is given then 5 parameters would be passed or else only 4 and default value of wallet would be 0 in the sql query executed.

### **2. Constructor Overloading**

Constructor overloading has been used in the pickup panel where the mobile number field has been kept optional and so thus the number of parameters passed varies depending upon if the mobile was entered.

### **3. Multiple Inheritance using interface implementation**

Multiple inheritance has been implemented by creating an fetchMobileNumber interface and then in the pickup class it has been inherited and the fetchMobileNumber1 method to fetch customer mobile has been overridden.

### **4. this keyword**

this keyword has been used to refer to the current instance of the class and thus close it by using setVisible as false and create an object of other class to open and make it visible by using setVisible - true.

### **5. Exception handling using try-catch block**

Exception handling has been used at several places for ensuring that if any error occurs while establishing a connection with the database then the exception is caught and to ensure that the wallet balance and the mobile number are always entered numeric so they have been parsed into integer and if any exception occurs in parsing that means some other character instead of an integer has been entered into a mobile number or wallet balance and there is a number format exception and that exception would be caught by the catch block.

### **6. Method Overriding for methods in interface**

Method overriding has been used while giving the definition to the fetchMobileNumber1 method of the fetchMobileNumber interface and it is used for to autofill mobile number of customer by name in the pickup panel.

### **7. Final keyword**

The final keyword has been used in the customer registered panel in the sqlQuery method of the query class so as to prevent it from being overridden in some other class.

#### **8. Static keyword**

Static keyword has been used in the sqlQuery method of the query class in customerregister panel so that it can be called without the creation of an object as it belongs to the whole class and no meaning for it to be specific for every instance of the class.

#### **9. Protected keyword**

Protected keyword has been used in the sqlQuery method of the query class so that it is not accessible from other packages.

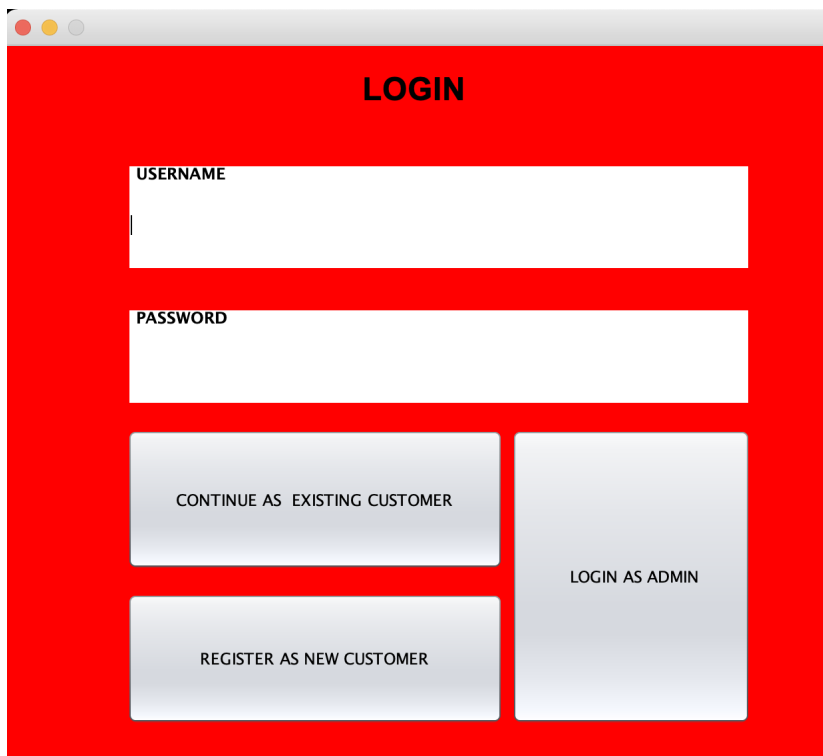
#### **10. Encapsulation**

Encapsulation has been used while overriding the fetchMobileNumber1 method as it takes the input parameter of name and returns the mobile number and is thus similar to get and set methods as we use while encapsulation for read and write operations.

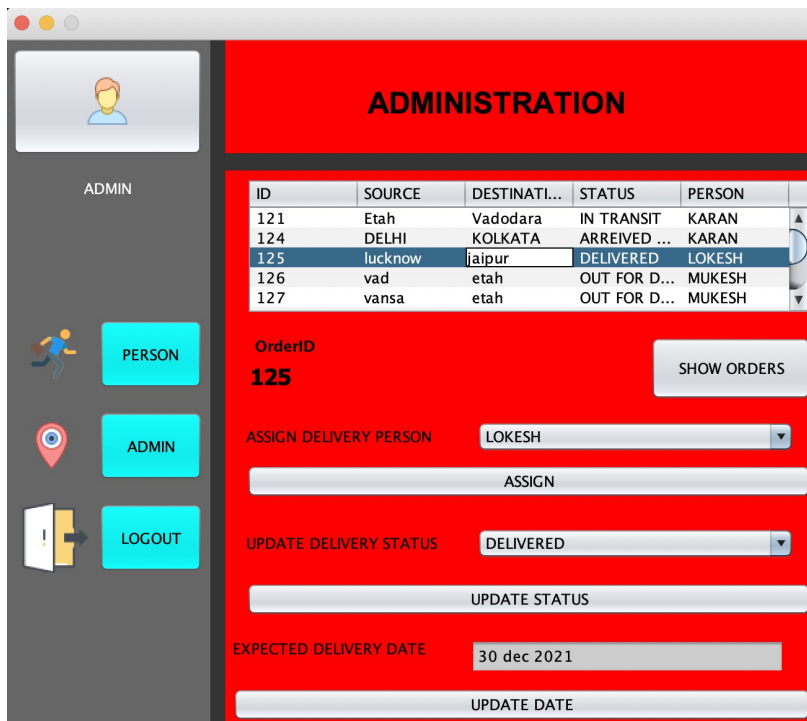
### **Issues faced**

1. Another problem we faced was the simultaneous working of multiple team members on the project as some had installed the Oracle NetBeans and after sometime it was removed from the web and only Apache netbeans was available which was very similar but had a few changes like we were not getting the Libraries option in it and we had to import a MySQL jar file in the Libraries folder to establish connection with the database and so working was getting difficult at initial stage though later the issue was solved.

## Screenshots:



A screenshot of a web application's login page. The page has a solid red background. At the top center, the word "LOGIN" is written in bold black capital letters. Below it, there are two white rectangular input fields. The first field is labeled "USERNAME" in black capital letters. The second field is labeled "PASSWORD" in black capital letters. Below the password field, there are three white rectangular buttons with black text. On the left side, there are two buttons: "CONTINUE AS EXISTING CUSTOMER" (top) and "REGISTER AS NEW CUSTOMER" (bottom). On the right side, there is a single button labeled "LOGIN AS ADMIN".



A screenshot of a web application's administration page. The page has a red background. On the left side, there is a dark grey sidebar with a user profile icon at the top, labeled "ADMIN". Below the profile, there are three buttons: "PERSON" (with a person icon), "ADMIN" (with a location pin icon), and "LOGOUT" (with a door icon). The main content area has a red header with the word "ADMINISTRATION" in bold black capital letters. Below the header, there is a table with 5 columns: ID, SOURCE, DESTINATI..., STATUS, and PERSON. The table contains 5 rows of data. The third row is highlighted. Below the table, there is a section for "OrderID 125" with a "SHOW ORDERS" button. Below that, there are three sections for updating delivery information: "ASSIGN DELIVERY PERSON" with a dropdown menu showing "LOKESH" and an "ASSIGN" button; "UPDATE DELIVERY STATUS" with a dropdown menu showing "DELIVERED" and an "UPDATE STATUS" button; and "EXPECTED DELIVERY DATE" with a text input showing "30 dec 2021" and an "UPDATE DATE" button.

ID	SOURCE	DESTINATI...	STATUS	PERSON
121	Etah	Vadodara	IN TRANSIT	KARAN
124	DELHI	KOLKATA	ARREIVED ...	KARAN
125	lucknow	jaipur	DELIVERED	LOKESH
126	vad	etah	OUT FOR D...	MUKESH
127	vansa	etah	OUT FOR D...	MUKESH

OrderID  
**125**

SHOW ORDERS

ASSIGN DELIVERY PERSON: LOKESH

ASSIGN

UPDATE DELIVERY STATUS: DELIVERED

UPDATE STATUS

EXPECTED DELIVERY DATE: 30 dec 2021

UPDATE DATE



ADMIN

 PERSON

 ADMIN

 LOGOUT

PERSON

ID	NAME	MOBILE	AADHAR NUMB...
11	Lokesh	92351712	324875
12	Karan	92351731	324836
13	Suresh	91351734	224822
14	Vedant	91354731	223836
15	Mukesh	91314731	223736
16	Revant	91312331	228836

EmployeeID  
**13**

SHOW

UPDATE MOBILE NUMBER

91351734

UPDATE MOBILE

UPDATE AADHAR NUMBER

224822

UPDATE AADHAR

NEW CUSTOMER REGISTRATION

Name

Username

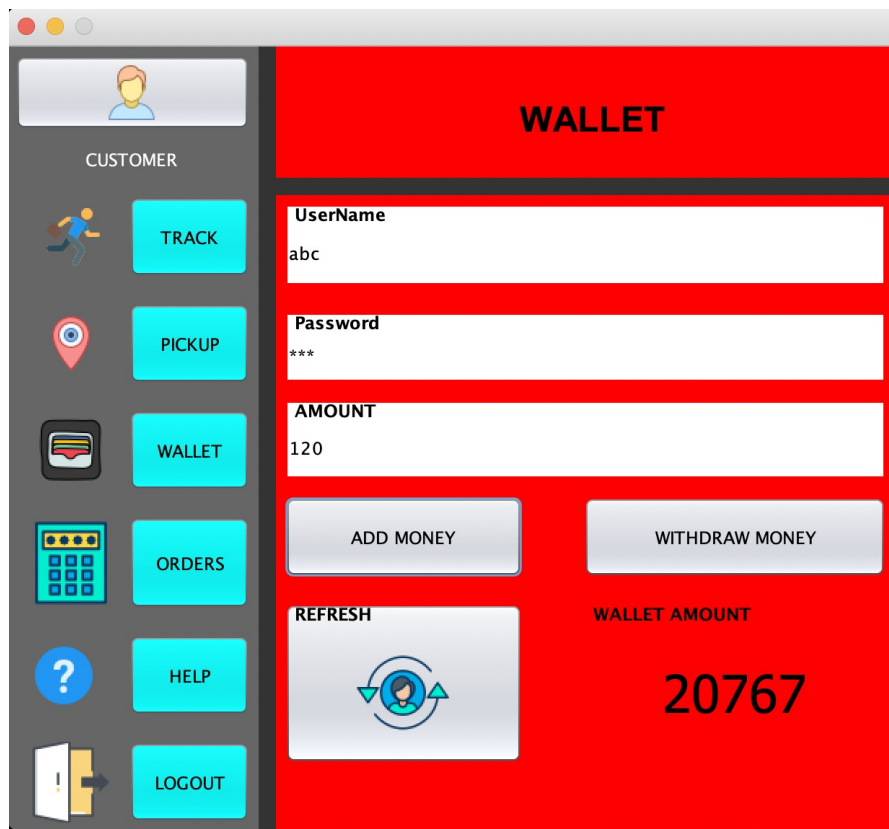
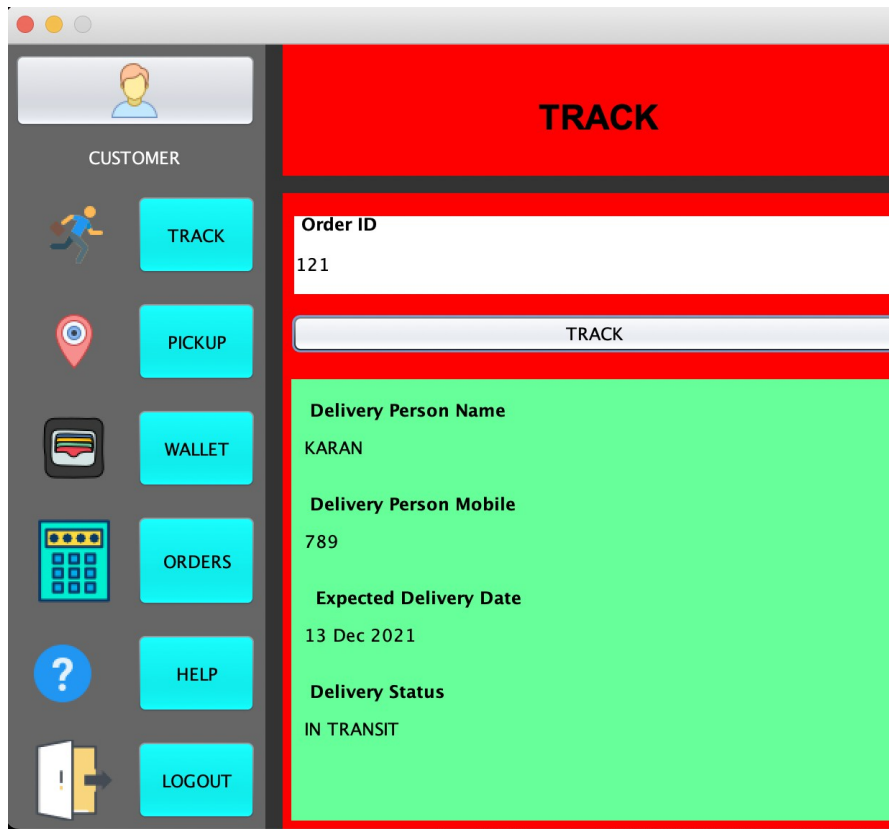
Password

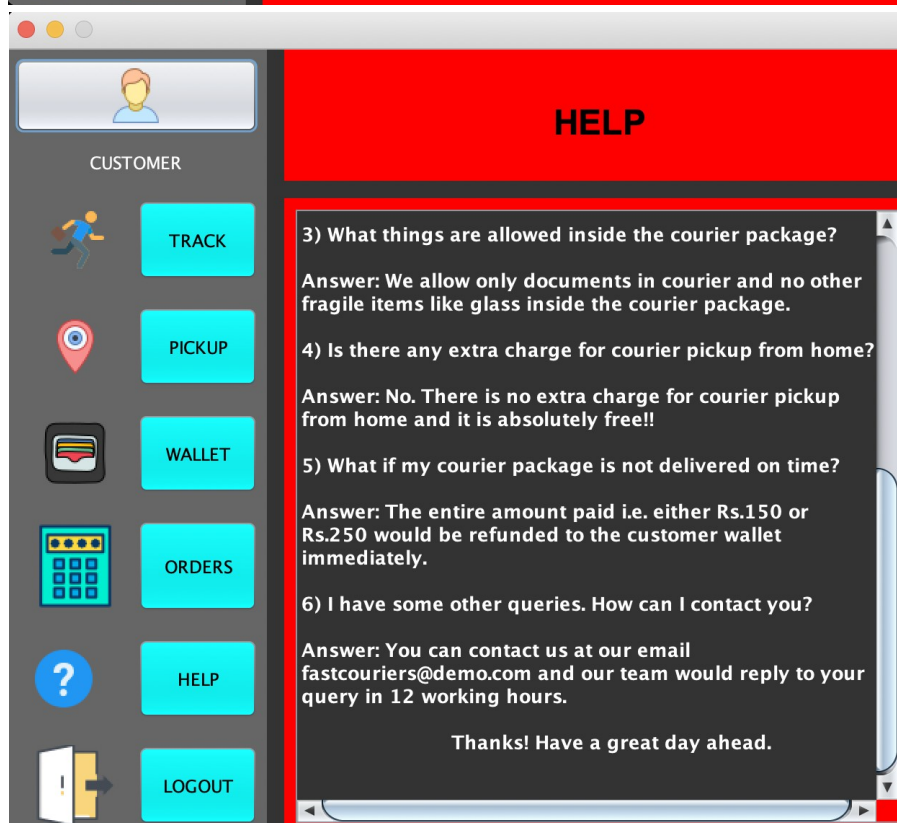
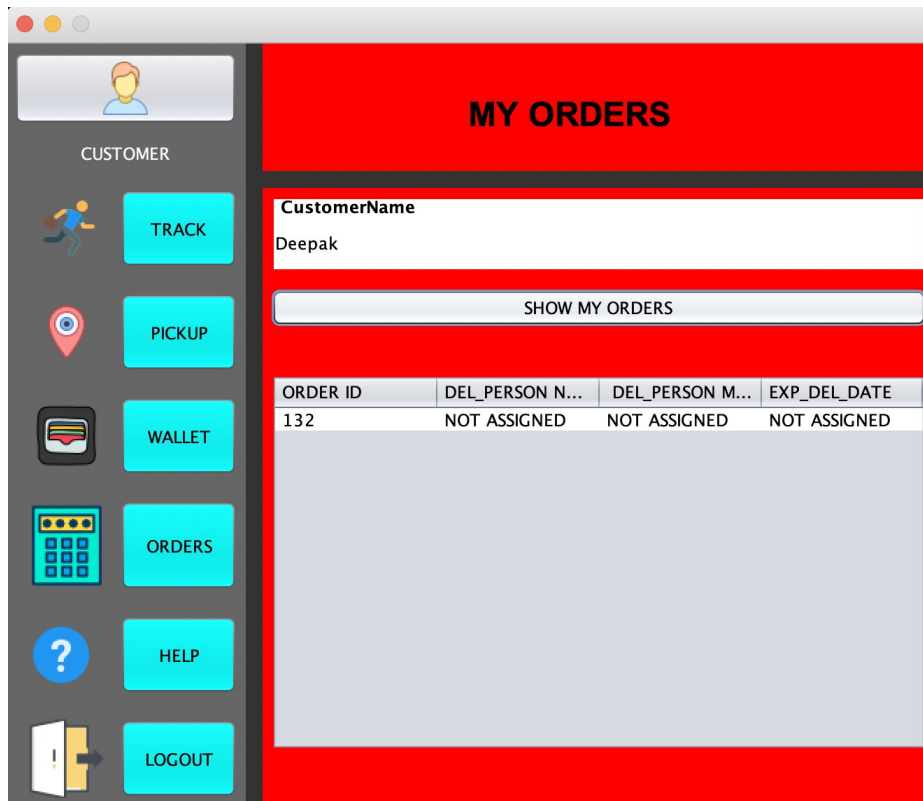
Mobile

Wallet (Optional)


SAVE

CONTINUE AFTER REGISTER











CUSTOMER




TRACK




PICKUP




WALLET



ORDERS



HELP



LOGOUT

PICKUP

PICKUP

NAME

Aman

ADDRESS

Etah

MOBILE

9234567891

FETCH MOBILE

DESTINATION

NAME

Ashutosh

ADDRESS

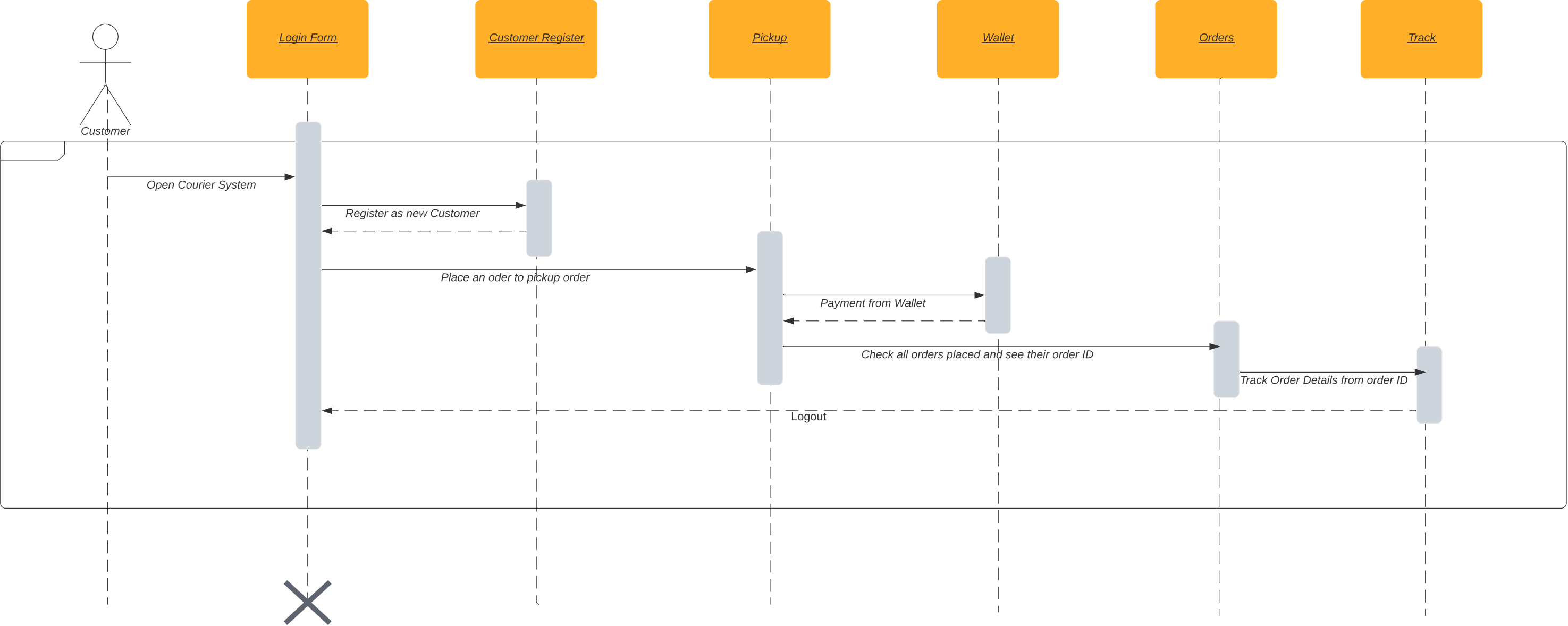
Vadodara

MOBILE (OPTIONAL)

4 DAYS : Rs. 150

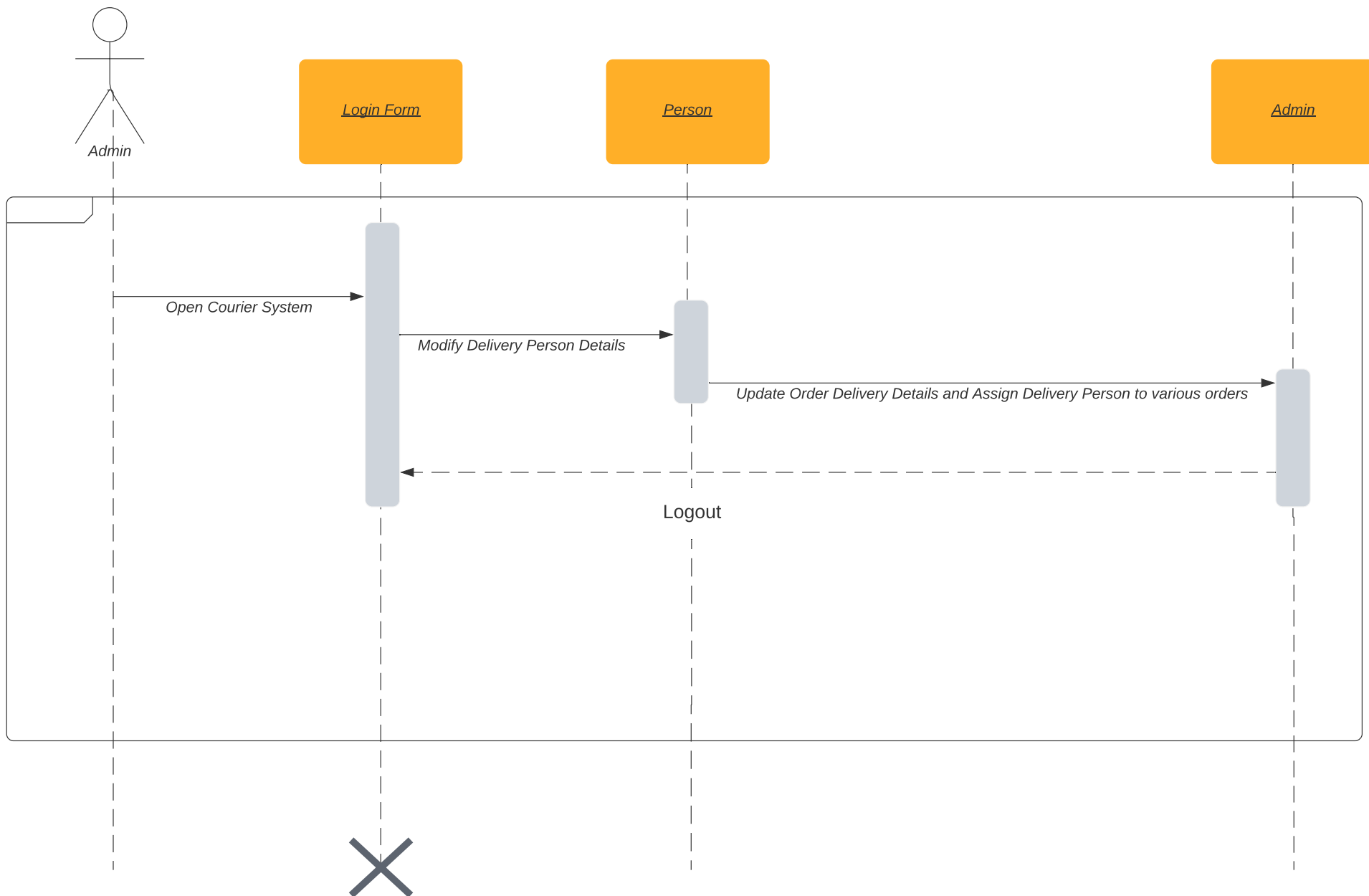
2 DAYS : Rs.250

Sequence diagram  
Customer Side Panel

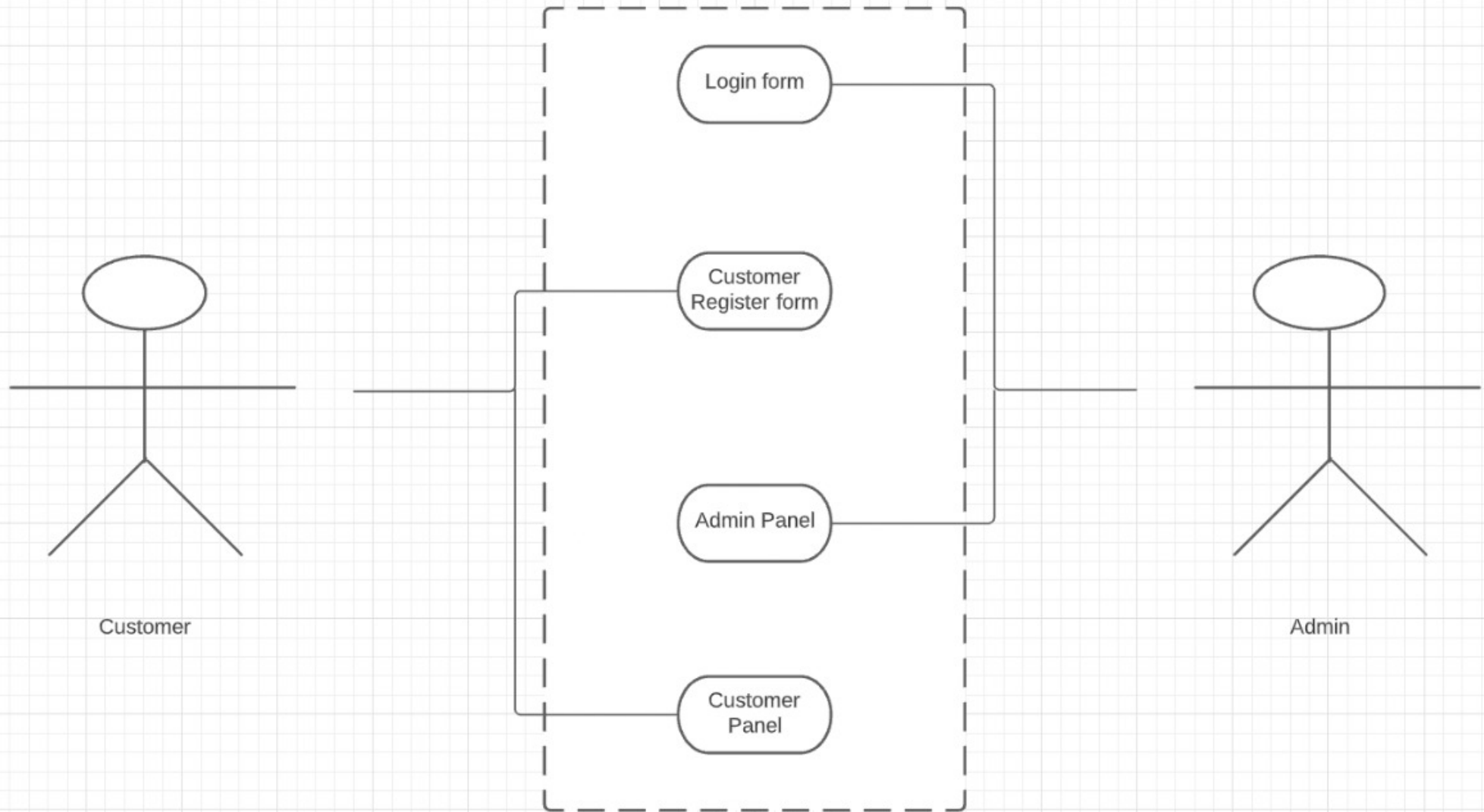


# Sequence Diagram

## Admin Side Panel



# Courier System



## Admin Panel

