UE21CS352B - Object Oriented Analysis & Design using Java

Mini Project Report

“Online Courier Management”

Submitted by:

Gajendra S	**PES1UG21CS197**
Ganta Srilalitha	**PES1UG21CS200**
Gowtham S	**PES1UG21CS210**
H M Bharath	**PES1UG21CS213**

6th Semester ‘D’ Section

Team Number : 6

Prof. Vinay Joshi
Assistant Professor

January - May 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING

PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

Project Abstract:

In today's fast-paced world, the efficient management of courier services is critical for businesses and individuals alike. The "Online Courier Management System," developed using Java and MySQL, represents a comprehensive solution to address the challenges and demands of the logistics industry. This project offers a wide range of features designed to streamline the entire courier process, benefiting users, administrators, and staff members.

One of the central components of this system is its user-centric approach. Users can easily book courier services online, simplifying the process of sending parcels to their desired destinations. This feature provides a user-friendly interface where customers can input essential details, such as sender and recipient information, parcel specifications, and delivery preferences, ensuring a seamless booking experience.

Upon booking, users receive digital receipts that serve as a proof of transaction. This receipt management feature enhances transparency and accountability, offering users peace of mind as they track the status and location of their parcels in real-time. The User Courier Tracking function is instrumental in providing this vital information, creating a sense of trust and reliability in the courier service.

Efficient workforce management is another key aspect of this system. New staff members can easily register, streamlining the onboarding process. However, the system incorporates an essential layer of administrative control, where only staff members approved by the admin gain access to the system. This ensures that only qualified personnel join the courier network, maintaining the quality and professionalism of the service.

The "Online Courier Management System" empowers administrators with tools for dynamic cost management. Admins can update costs based on market conditions and business requirements, ensuring competitive pricing that aligns with industry standards. Additionally, admins have access to comprehensive courier tracking capabilities, allowing them to monitor parcels, delivery timelines, and overall performance, thereby ensuring efficient and accountable operations.

To further optimise courier routing, the system incorporates Staff Pickup and Delivery Assignment based on location. This feature leverages geographical proximity to assign pickups and deliveries to staff members, leading to timely and efficient service.

In conclusion, this project represents a significant advancement in the courier management industry. Its user-friendly interface, real-time tracking capabilities, streamlined staff management, and dynamic cost control make it a valuable asset for

businesses and individuals alike. By providing complete details tracking for both administrators and users, this system promotes transparency, accountability, and overall customer satisfaction, ultimately redefining the courier management landscape.

FUNCTIONALITIES:

Home

- **Admin home:** land to admin homepage
- **Staff home:** land to staff homepage
- **User home:** land to user homepage

Admin

- **Admin Authentication:** Verifying the identity of admins trying to access the system.
- **Admin Management:** Allowing admins to add, edit, or delete other admins, staff, or users.
- **Staff Management:** Enabling admins to manage staff details, schedules, and deliveries.
- **Update Cost**: update cost per KG of the package to be delivered.

Staff

- **Staff Authentication:** Verifying the identity of admins trying to access the system.
- **PickUp:** track & manage courier pickup schedules
- **Delivery:** track & manage delivery schedules
- **WorkLog:** track the remaining or uncompleted deliveries

User

- **User Authentication:** Verifying the identity of users trying to access the system.
- **User Registration :**register the user if the user is new.
- **Send Courier**: user gives courier details & gets the courier receipt
- **Package Tracking:** Allowing users to track their packages in real-time.
- **User Management:** Managing user profiles, orders, and feedback
-

Use case modelling:

Actors

User:

- Description: The end-users of the system, typically individuals or businesses looking to send parcels or goods using courier services.
- Roles:
  - Can register and log in to the system.
  - Can book courier services by providing necessary details (pickup address, destination, weight, etc.).
  - Can view the status and tracking details of booked couriers.
  - Can update their profile information.

Admin:

- Description: Administrative users responsible for managing and overseeing the entire Courier Management System.
- Roles:
  - Can log in to the system with admin credentials.
  - Manages courier cost details.
  - Approves or rejects staff registrations.
  - Monitors and tracks all couriers through the system.
  - Manages staff assignments for pickups and deliveries.

Staff:

- Description: Employees responsible for executing courier pickups and deliveries assigned to them.
- Roles:
  - Can log in to the system with staff credentials.
  - Receives approvals from the admin.
  - Can view assigned pickups and deliveries based on their location.
  - Marks pickups as completed.
  - Marks deliveries as completed.

Use cases Description:

Courier Booking:

- Description: Allows users to book courier services by providing necessary details.
- Actors: User
- Basic Flow:

  User logs in.
  User enters details for the courier (pickup address, destination, weight, etc.).
  System calculates the cost.
  User confirms the booking.
  System generates a digital receipt.
- Extensions:

  If the user provides incorrect information, the system prompts for corrections.

Courier Tracking:

- Description: Enables users and admin to track the status and details of booked couriers.
- Actors: User, Admin
- Basic Flow:

  User or Admin logs in.
  User enters the tracking number.
  System displays the current status and details of the courier.
- Extensions:

  If the tracking number is not found, the system provides an appropriate message.

Admin Functions:

- Description: Provides admin functionalities for managing the Courier Management System.
- Actors: Admin
- Basic Flow:

  Admin logs in.
  Admin manages courier cost details.
  Admin approves or rejects staff registrations.
  Admin monitors and tracks all couriers.

Admin assigns staff for pickups and deliveries.

- Extensions:

  If there are no staff registrations to approve, the system provides a notification.

Staff Pickup Assignment based on location:

- Description: Allows the admin to assign staff for courier pickups based on their location.
- Actors: Admin
- Basic Flow:

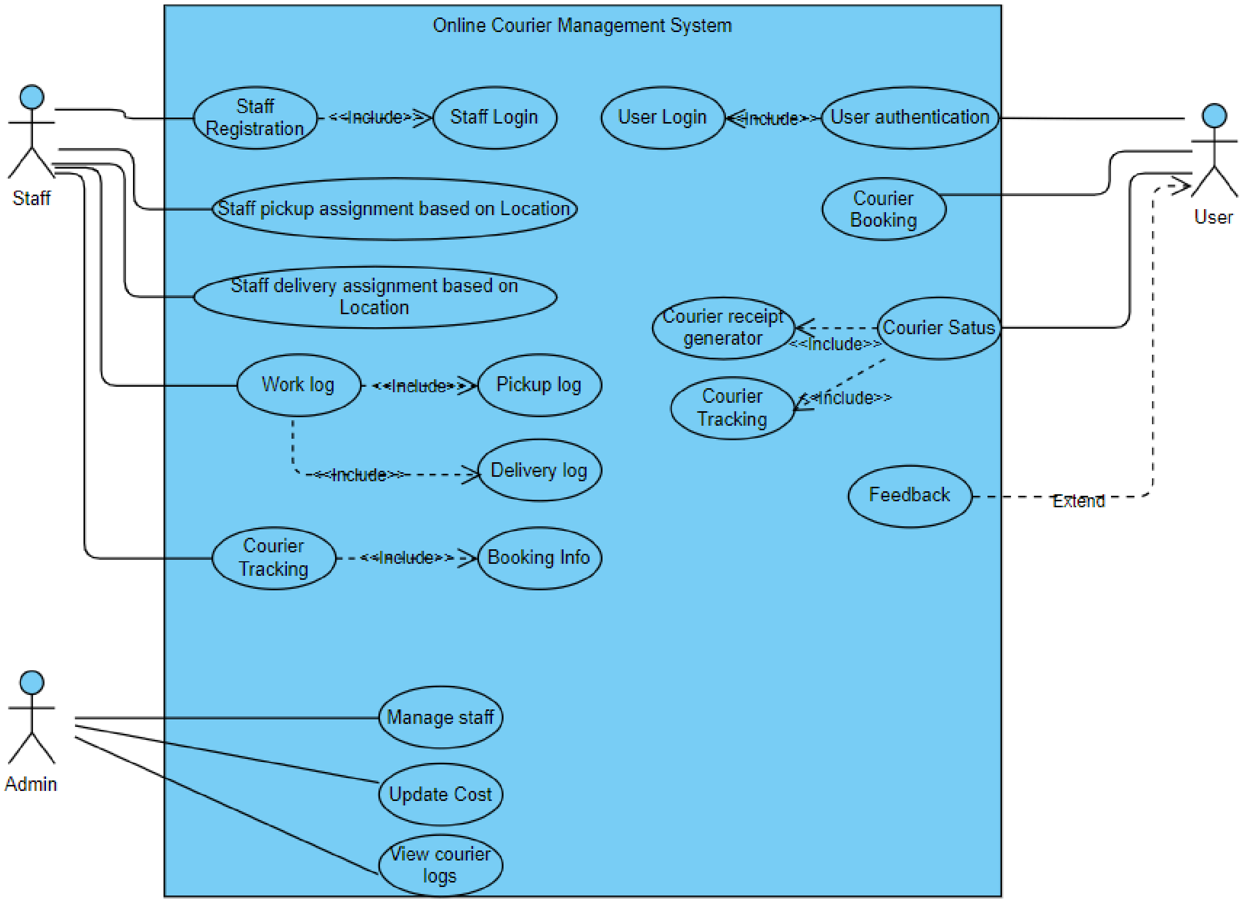
  Admin logs in.
  Admin views available staff and their locations.
  Admin assigns staff for pickups based on their location.

Staff Delivery Assignment based on Location:

- Description: Allows the admin to assign staff for courier deliveries based on their location.
- Actors: Admin
- Basic Flow:

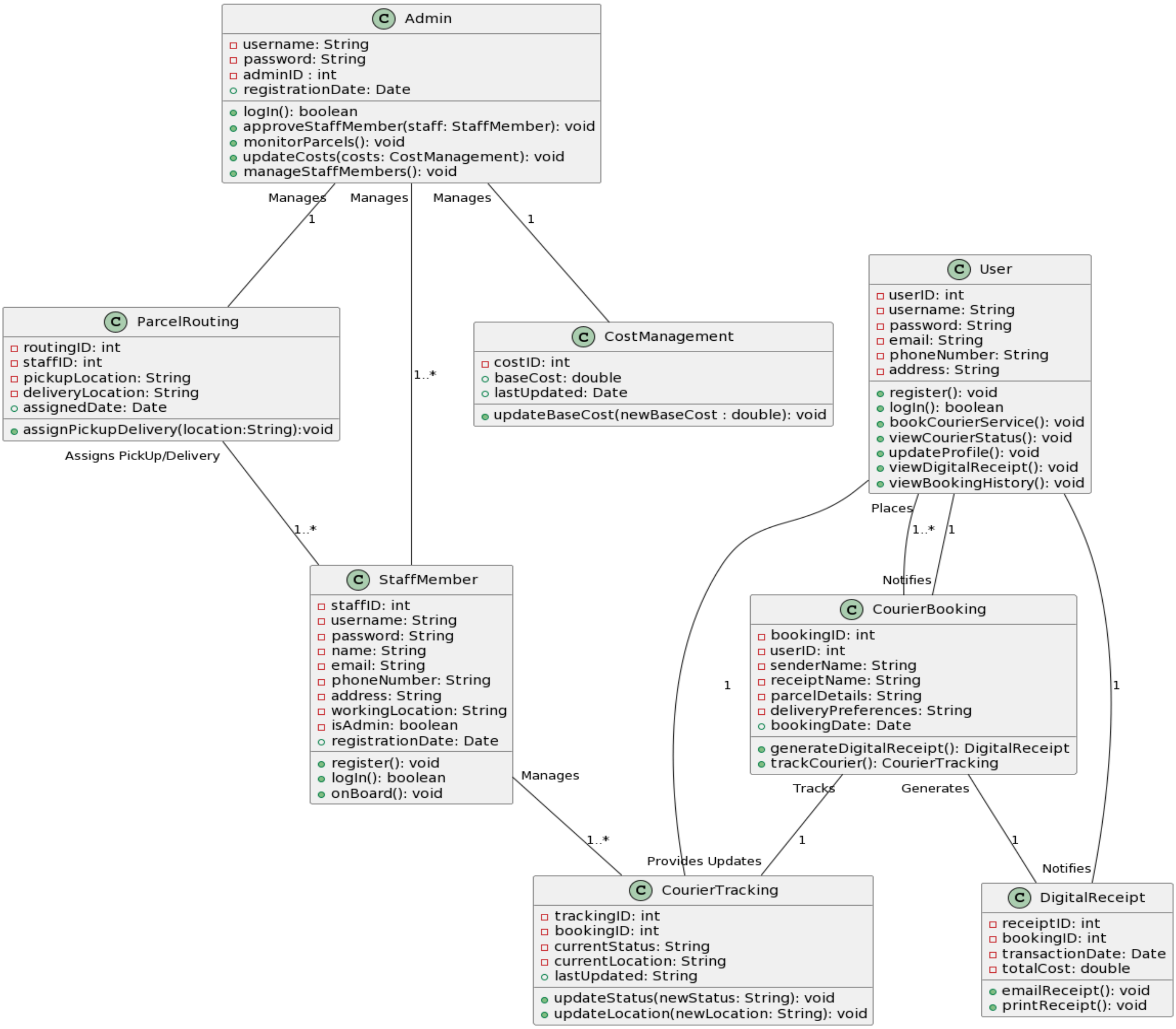
  Admin logs in.
  Admin views available staff and their locations.
  Admin assigns staff for deliveries based on their location.

USE-CASE DIAGRAM



CLASS DIAGRAM

(BEFORE MODIFICATION)



MODIFIED CLASS DIAGRAM

1. Separation of Controllers and Services:

- The classes related to controllers (CourierController) are grouped under the "Controllers" package, while the classes related to services (AdminService, CourierService, UserService) are grouped under the "Services" package. This separation ensures that classes responsible for handling web requests and view rendering are distinct from classes responsible for implementing business logic and operations.

2. Single Responsibility Principle (SRP) in Services:

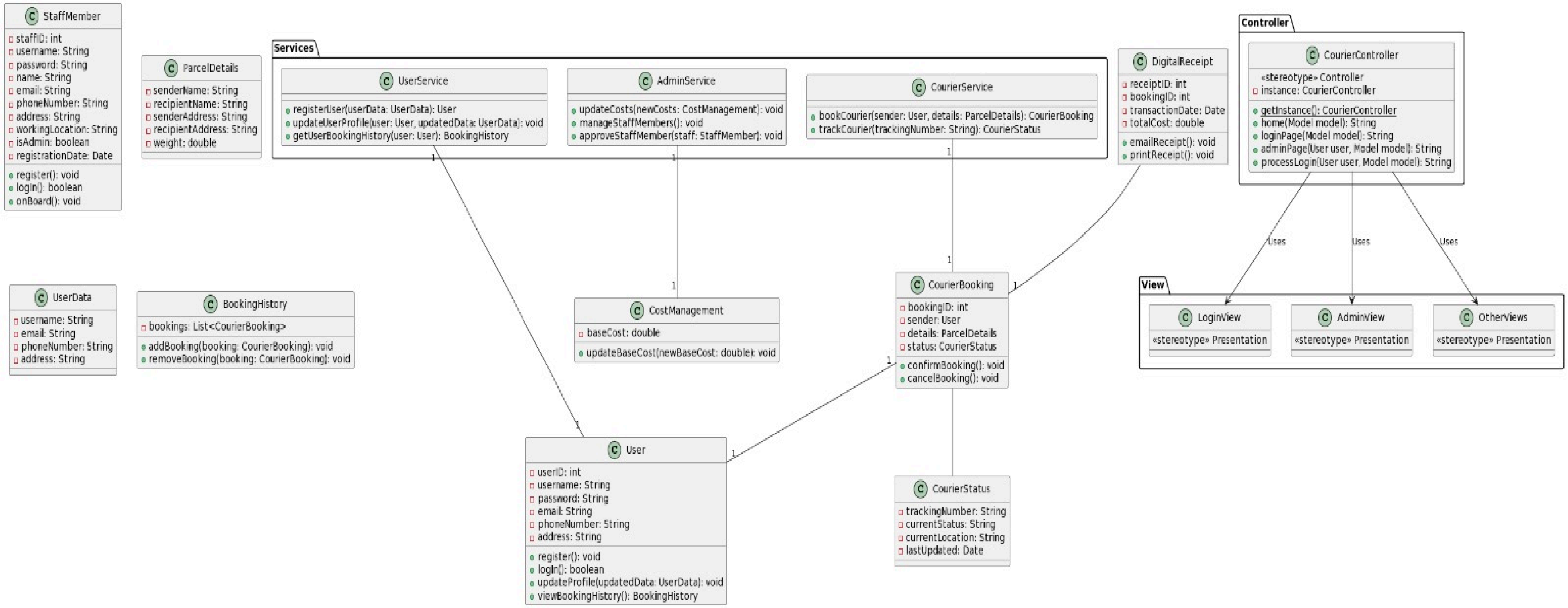
- Each service class (AdminService, CourierService, UserService) has a single responsibility related to a specific domain area. Eg: AdminService manages admin-related operations, CourierService handles courier-related operations, and UserService deals with user-related operations. This adherence to SRP ensures that each service class focuses on a specific aspect of the system's functionality.

3. MVC Pattern:

Model: Represents the data entities (Admin and User classes).
View: Represents the presentation layer (UI).
Controller: Handles the incoming requests, processes them using services, and determines the appropriate view to render (CourierController)

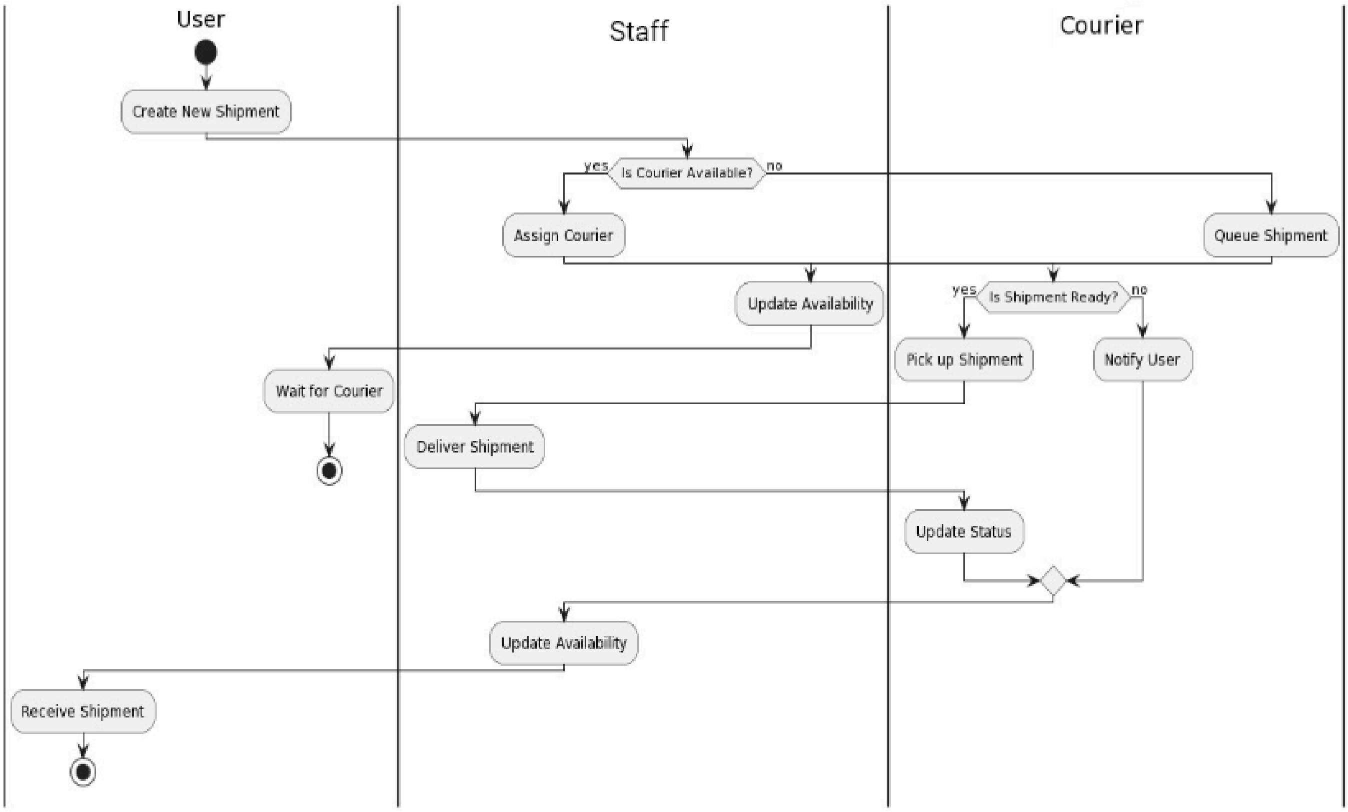
4.Singleton Pattern:

To ensure that there is only one instance of the CourierController class throughout the application . By enforcing the Singleton Pattern, multiple instances of CourierController class cannot be created, and all parts of the application will interact with the same instance of the class.

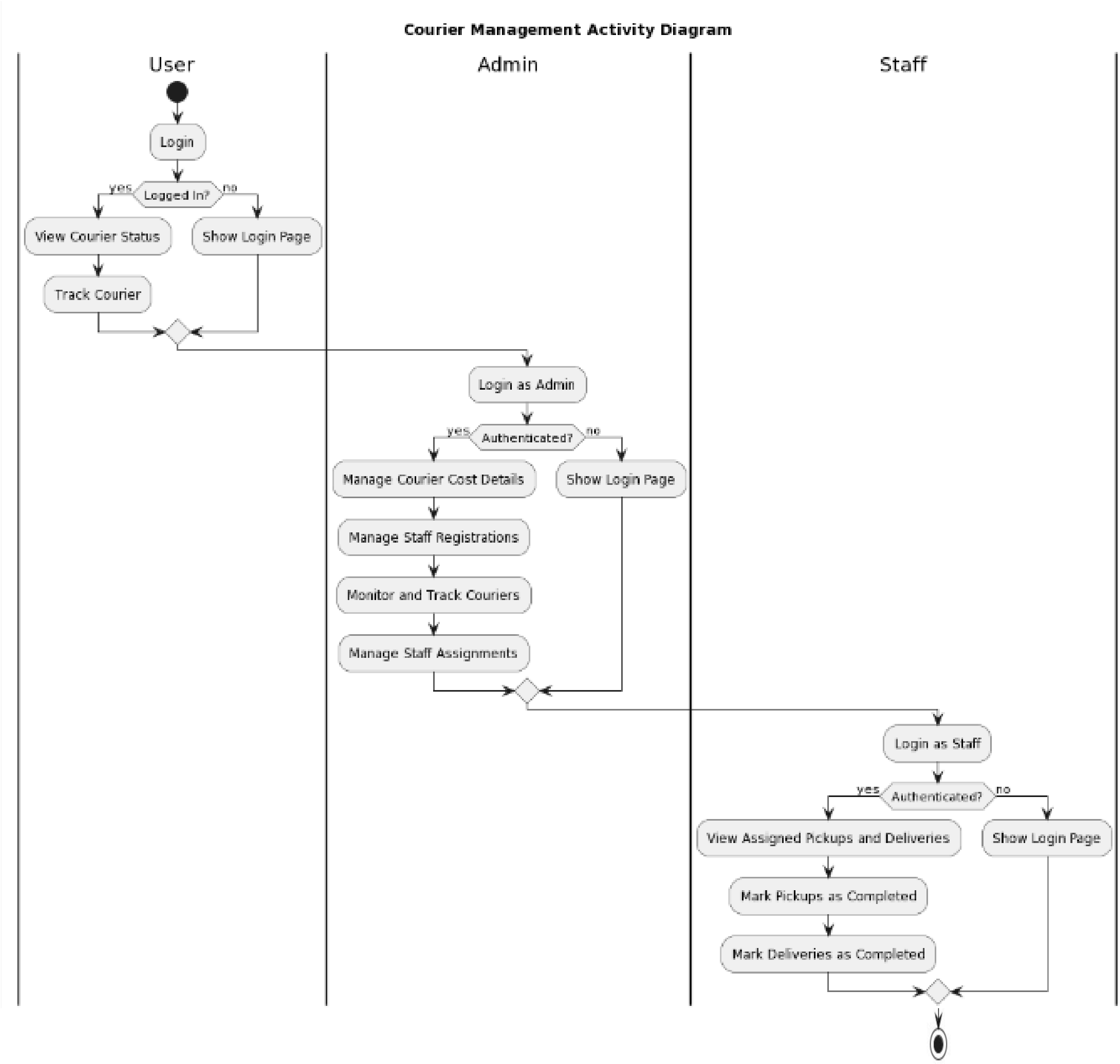

StaffMember
staffID: int
username: String
password: String
name: String
email: String
phoneNumber: String
address: String
workingLocation: String
isAdmin: boolean
registrationDate: Date
register(): void
login(): boolean
onBoard(): void
ParcelDetails
senderName: String
recipientName: String
senderAddress: String
recipientAddress: String
weight: double
Services
UserService
registerUser(userData: UserData): User
updateUserProfile(user: User, updatedData: UserData): void
getUserBookingHistory(user: User): BookingHistory
AdminService
updateCosts(newCosts: CostManagement): void
manageStaffMembers(): void
approveStaffMember(staff: StaffMember): void
CourierService
bookCourier(sender: User, details: ParcelDetails): CourierBooking
trackCourier(trackingNumber: String): CourierStatus
DigitalReceipt
receiptID: int
bookingID: int
transactionDate: Date
totalCost: double
emailReceipt(): void
printReceipt(): void
Controller
CourierController
«stereotype» Controller
instance: CourierController
getInstance(): CourierController
home(Model model): String
loginPage(Model model): String
adminPage(User user, Model model): String
processLogin(User user, Model model): String
Uses
Uses
Uses
View
LoginView
«stereotype» Presentation
AdminView
«stereotype» Presentation
OtherViews
«stereotype» Presentation
UserData
username: String
email: String
phoneNumber: String
address: String
BookingHistory
bookings: List<CourierBooking>
addBooking(booking: CourierBooking): void
removeBooking(booking: CourierBooking): void
CostManagement
baseCost: double
updateBaseCost(newBaseCost: double): void
CourierBooking
bookingID: int
sender: User
details: ParcelDetails
status: CourierStatus
confirmBooking(): void
cancelBooking(): void
User
userID: int
username: String
password: String
email: String
phoneNumber: String
address: String
register(): void
login(): boolean
updateProfile(updatedData: UserData): void
viewBookingHistory(): BookingHistory
CourierStatus
trackingNumber: String
currentStatus: String
currentLocation: String
lastUpdated: Date
1
1
1
1
1
1
1
1
1
1

Activity diagram

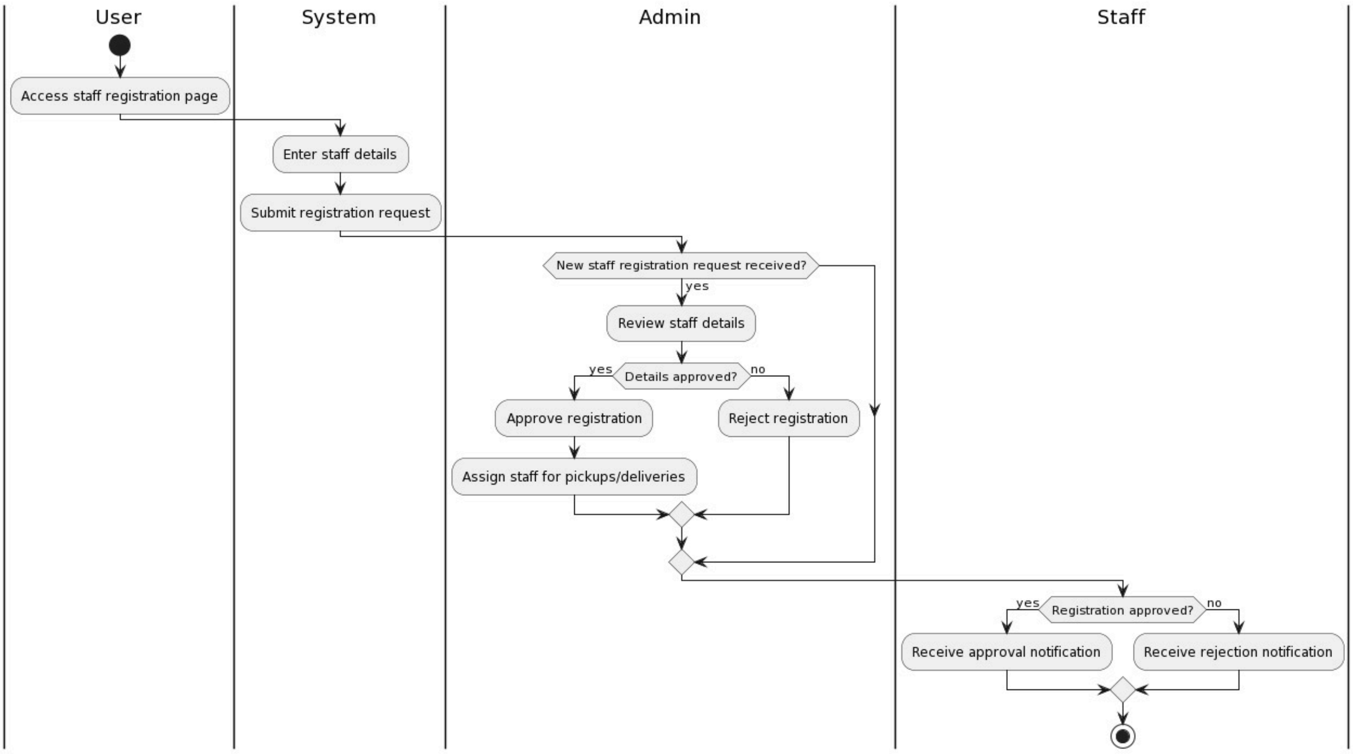
Courier Assignment



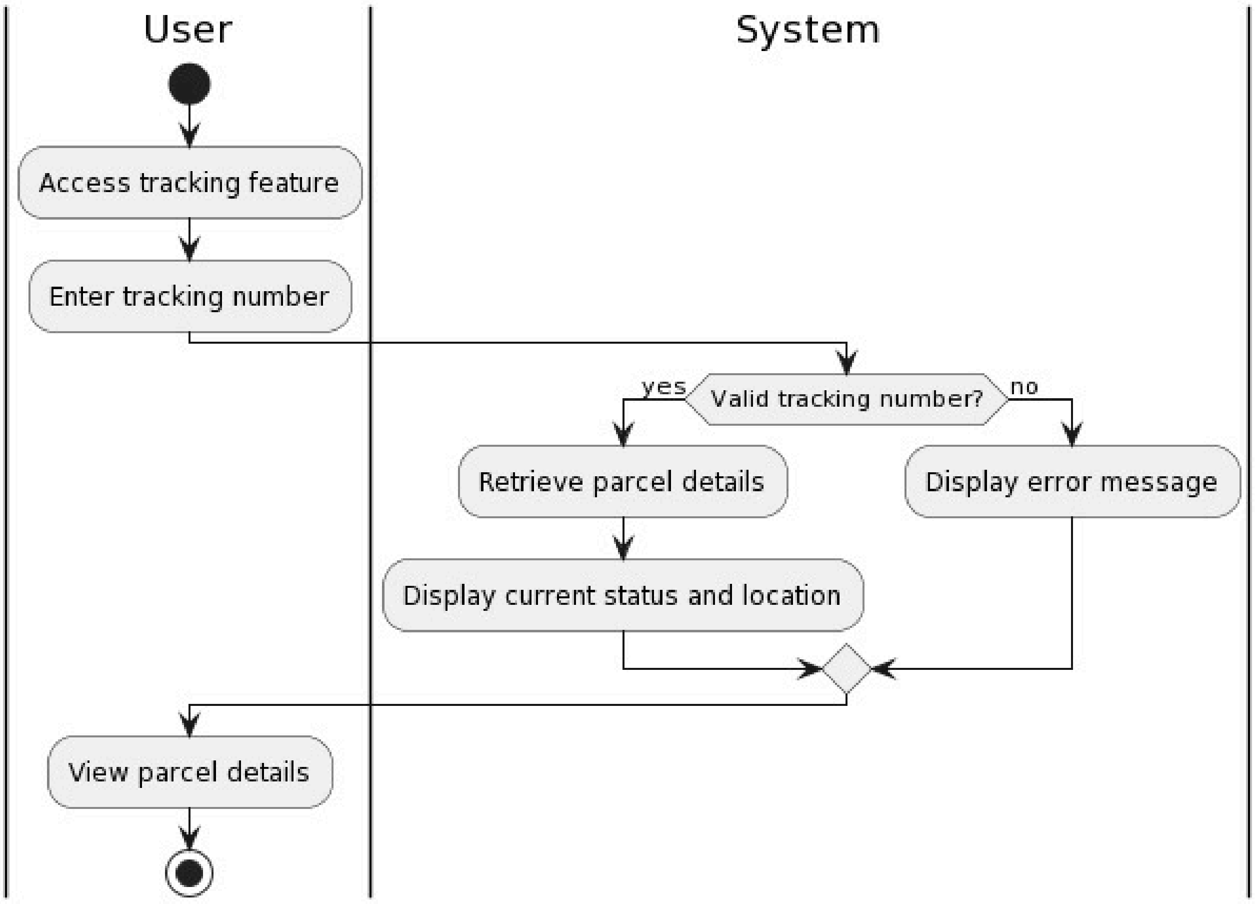
Actor Registration and Courier Management


Courier Management Activity Diagram
User
Admin
Staff
Login
Logged In?
yes
no
View Courier Status
Show Login Page
Track Courier
Login as Admin
Authenticated?
yes
no
Manage Courier Cost Details
Show Login Page
Manage Staff Registrations
Monitor and Track Couriers
Manage Staff Assignments
Login as Staff
Authenticated?
yes
no
View Assigned Pickups and Deliveries
Show Login Page
Mark Pickups as Completed
Mark Deliveries as Completed

Staff Registration


User
System
Admin
Staff
Access staff registration page
Enter staff details
Submit registration request
New staff registration request received?
yes
Review staff details
yes
Details approved?
no
Approve registration
Reject registration
Assign staff for pickups/deliveries
yes
Registration approved?
no
Receive approval notification
Receive rejection notification

Parcel Tracking

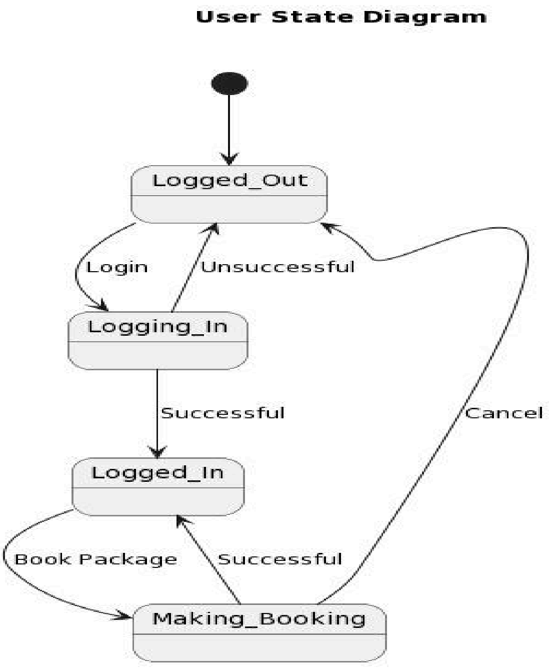


STATE DIAGRAMS

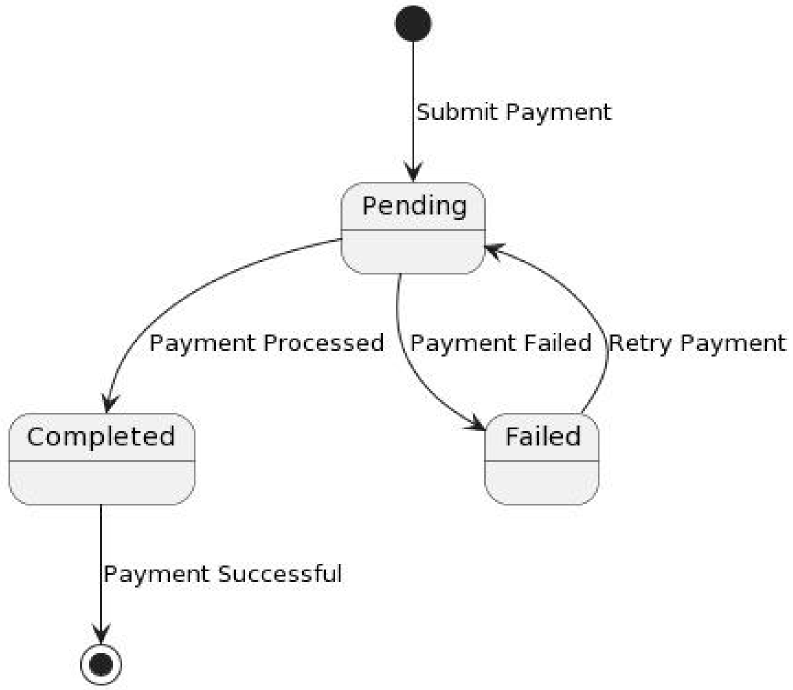
Here are some entities for which you can create state diagrams:

1. **User:**
   - States: Logged In, Logged Out, Registration, Profile Update
2. **Package:**
   - States:Picked,Shipped, In Transit, Delivered
3. **Payment:**
   - States: Pending, Authorised, Completed, Failed, Refunded
4. **Logs:**
   - States: To Do, In Progress, Completed, Blocked

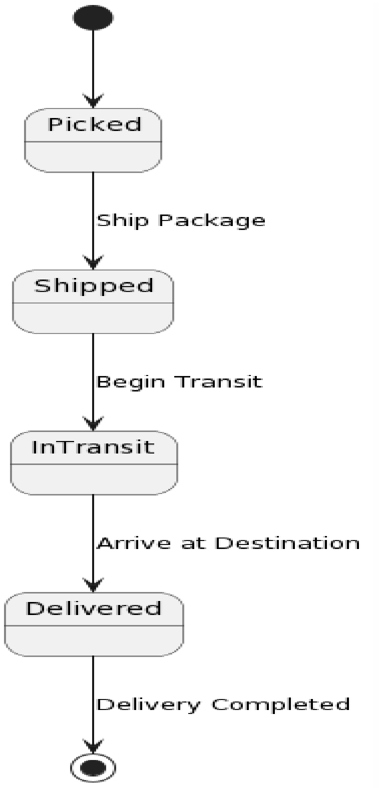
State diagram for user:



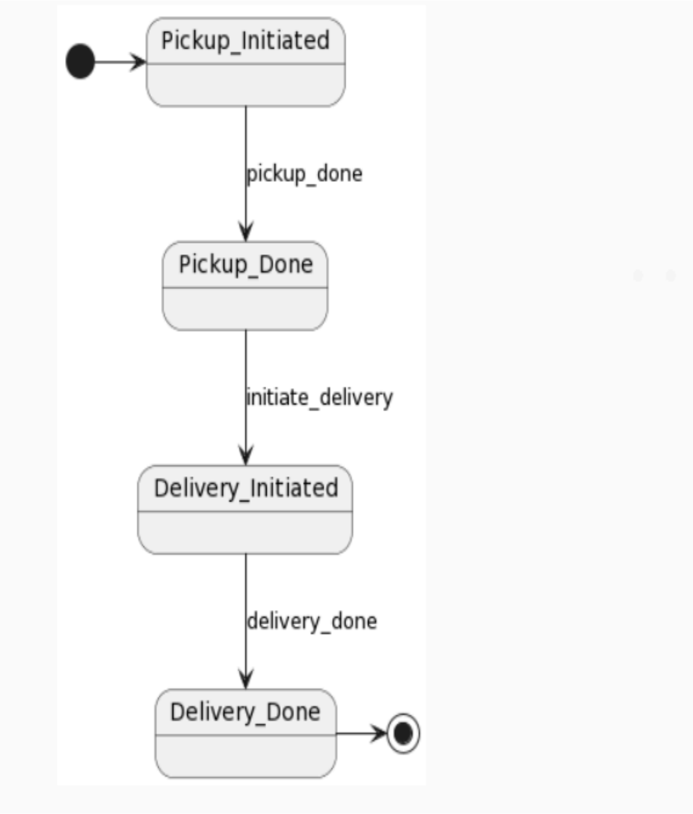
State diagram for payment:



State diagram for courier package:



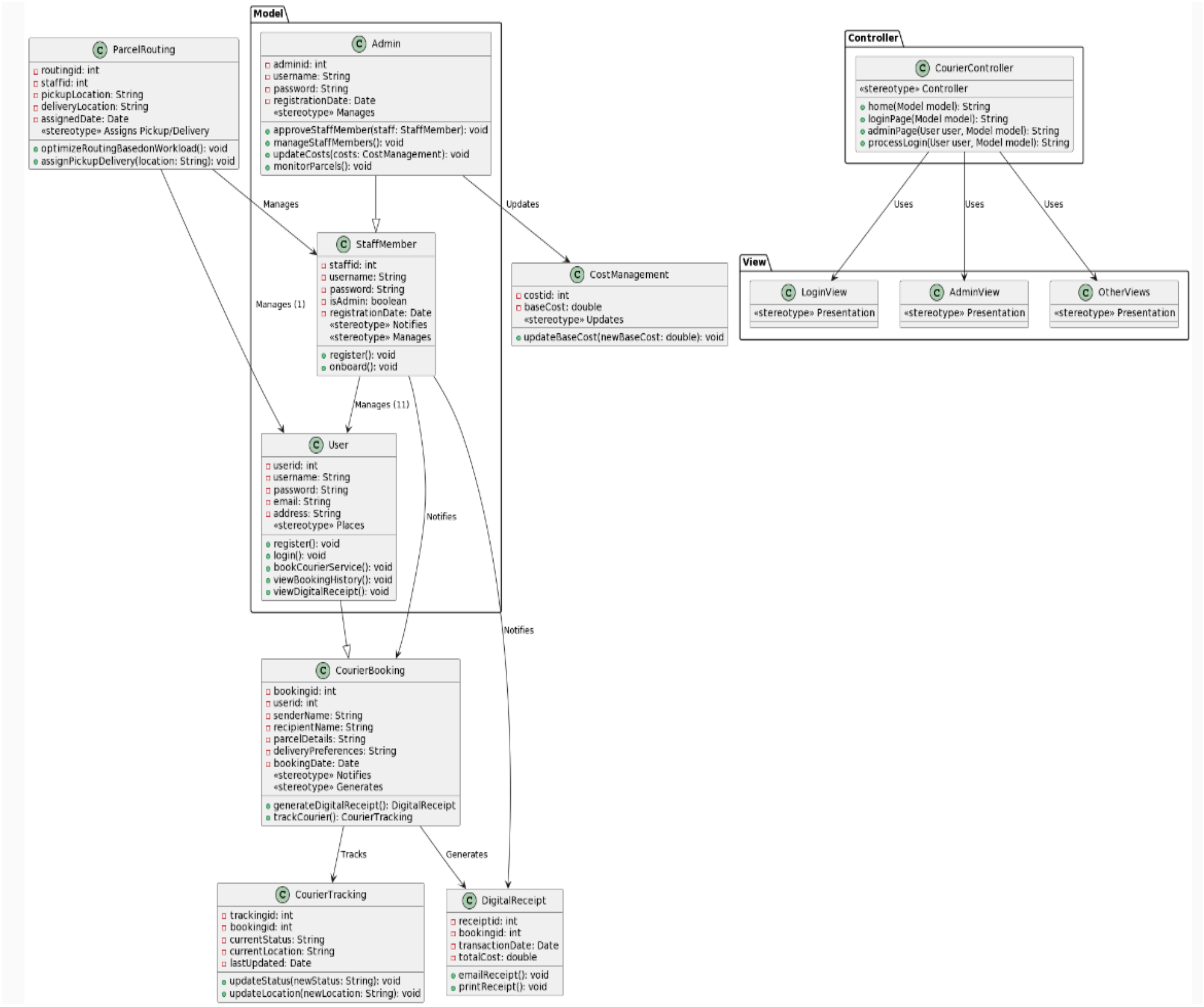
State diagram for Logs:



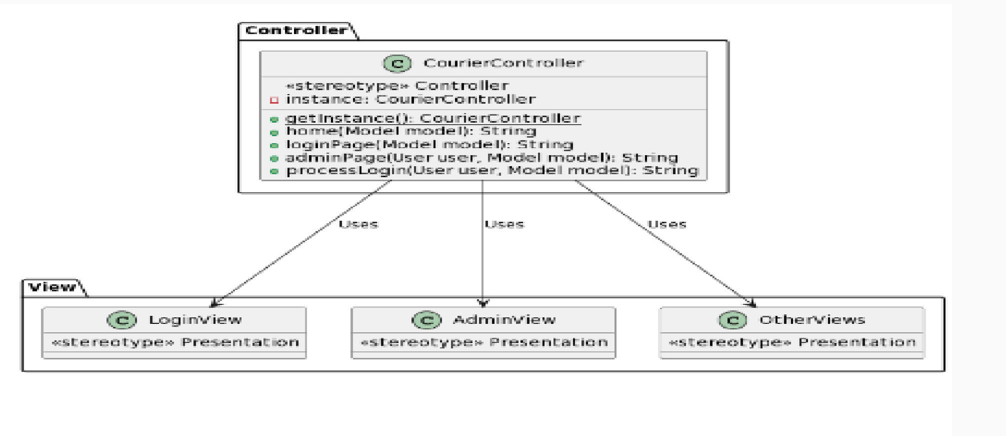
4 Design Patterns

- **MVC-Model-View-Controller (MVC) Pattern:**

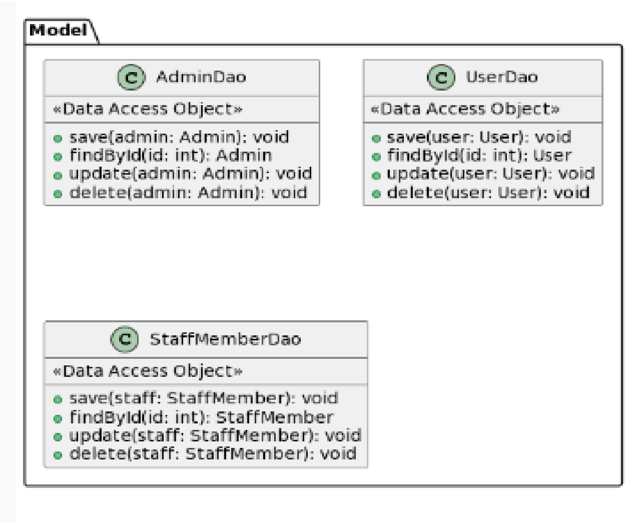
- Model: Represents the data entities (Admin and User classes).
- View: Represents the presentation layer (UI).
- Controller: Handles the incoming requests, processes them using services, and determines the appropriate view to render (CourierController).



- **Singleton-**only 1 controller instance

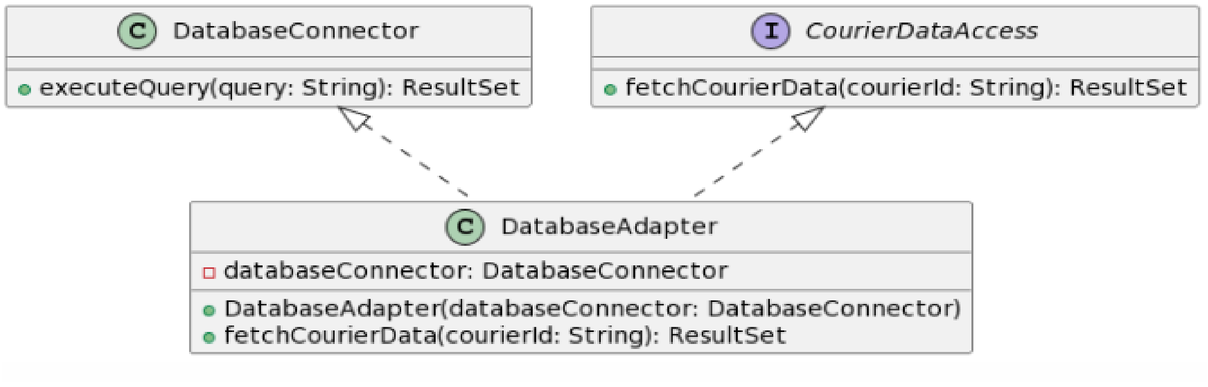

Controller
C CourierController
«stereotype» Controller
instance: CourierController
getInstance(): CourierController
home(Model model): String
loginPage(Model model): String
adminPage(User user, Model model): String
processLogin(User user, Model model): String
Uses
Uses
Uses
View
C LoginView
«stereotype» Presentation
C AdminView
«stereotype» Presentation
C OtherViews
«stereotype» Presentation

- **Data Access Object (DAO) pattern** : DAO classes (AdminDao, UserDao, StaffMemberDao) are introduced to handle database operations for their respective entities (Admin, User, StaffMember).



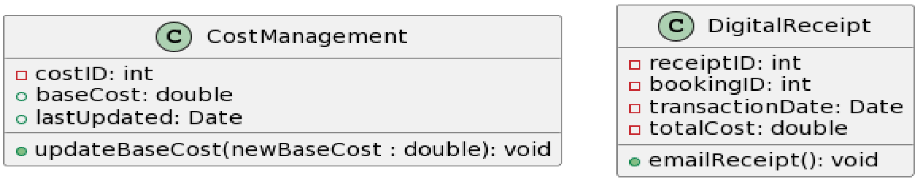
- **Adapter Pattern:**

Fetch data from an existing backend database that uses a specific database access interface, such as JDBC (Java Database Connectivity).


C DatabaseConnector
executeQuery(query: String): ResultSet
I CourierDataAccess
fetchCourierData(courierId: String): ResultSet
C DatabaseAdapter
databaseConnector: DatabaseConnector
DatabaseAdapter(databaseConnector: DatabaseConnector)
fetchCourierData(courierId: String): ResultSet

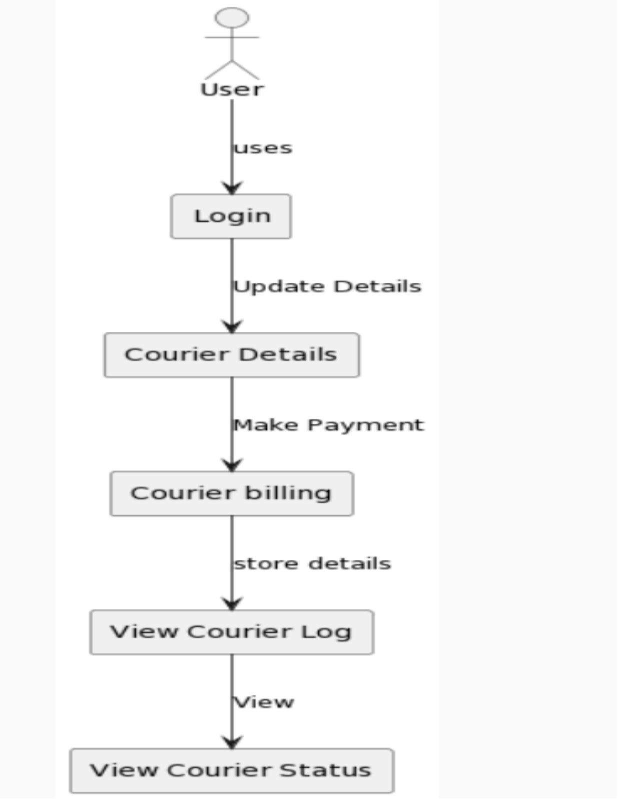
4 Design Principles

- **Single Responsibility Principle(SRP):**
    - The **"CostManagement"** class has a single responsibility, which is managing cost-related operations. It encapsulates the functionality related to managing and updating costs, adhering to the SRP.
    - The **"DigitalReceipt"**class has a single responsibility, which is managing digital receipts. It encapsulates the functionality of notifying/sending the digital receipt to the user.



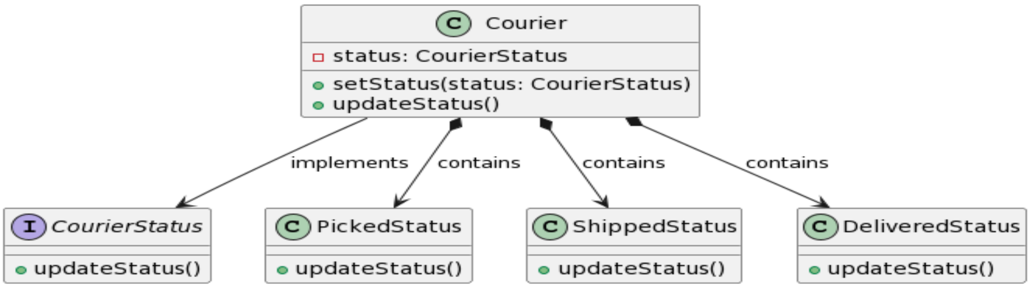
- **Separation Of Concerns:**

The process of booking a courier and viewing its details is decoupled into smaller modules, reflecting low coupling & high cohesion.



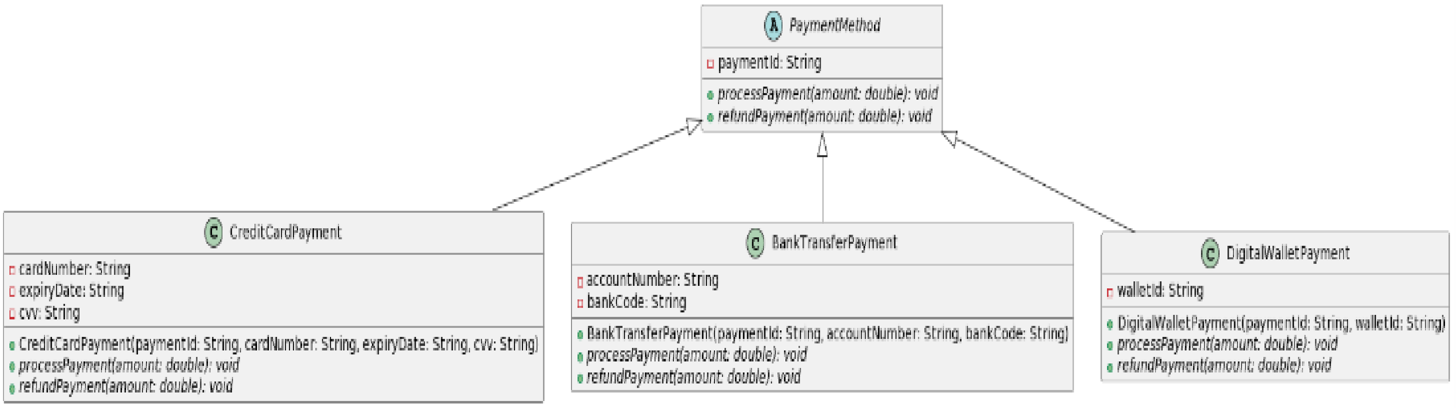
- **Open Closed Principle:**

  Instead of directly modifying the code for handling different courier statuses (e.g., "Picked," "Shipped," "Delivered"), the system can be designed to allow for the addition of new status types without changing the existing code.New status classes can be added to the system without modifying the existing codebase, allowing for easy extension of the system's behaviour.



- **Liskov Substitution Principle:**

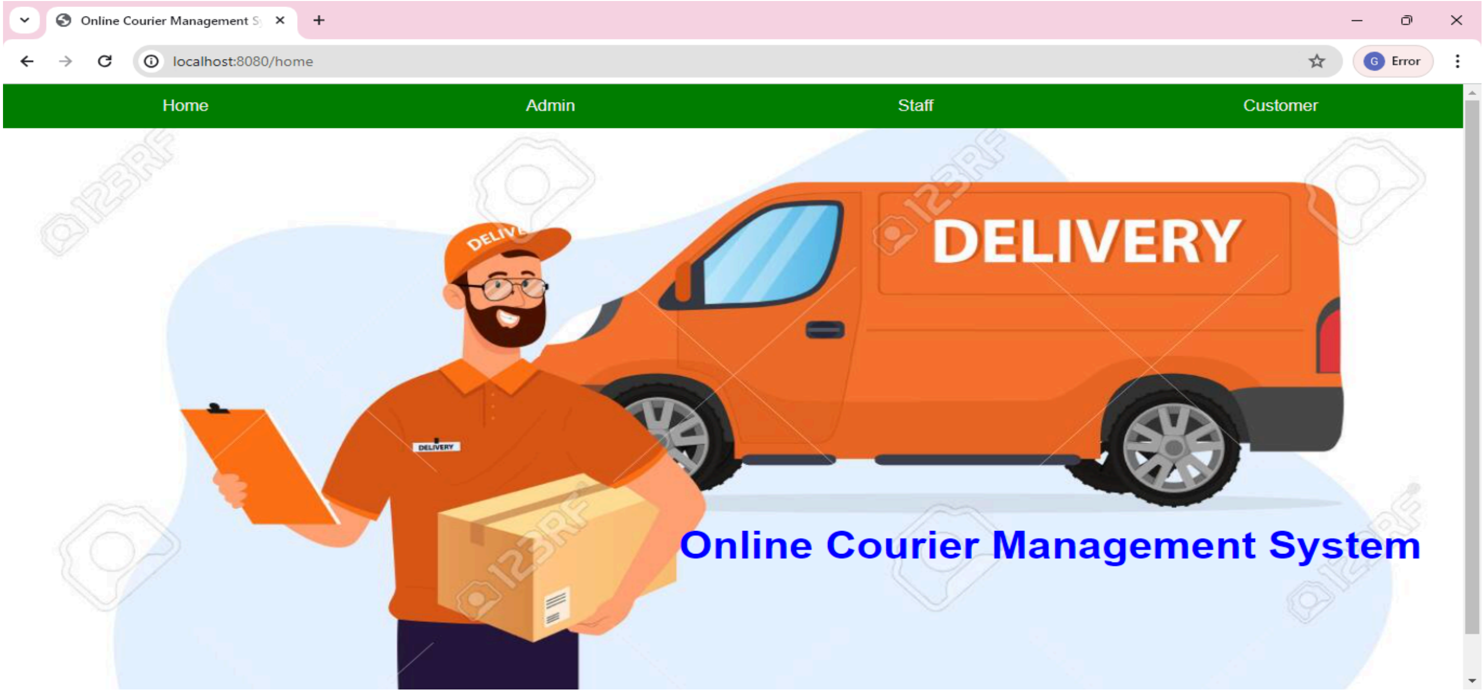
process payments using any type of payment method without needing to know the specific subclass. For example, if there's a method to process a payment using a PaymentMethod object, it should work correctly for instances of CreditCardPayment, BankTransferPayment, or DigitalWalletPayment



A PaymentMethod
paymentId: String
processPayment(amount: double): void
refundPayment(amount: double): void
C CreditCardPayment
cardNumber: String
expiryDate: String
cvv: String
CreditCardPayment(paymentId: String, cardNumber: String, expiryDate: String, cvv: String)
processPayment(amount: double): void
refundPayment(amount: double): void
C BankTransferPayment
accountNumber: String
bankCode: String
BankTransferPayment(paymentId: String, accountNumber: String, bankCode: String)
processPayment(amount: double): void
refundPayment(amount: double): void
C DigitalWalletPayment
walletId: String
DigitalWalletPayment(paymentId: String, walletId: String)
processPayment(amount: double): void
refundPayment(amount: double): void

<u>SCREENSHOTS OF INPUT & OUTPUT VALUES</u>

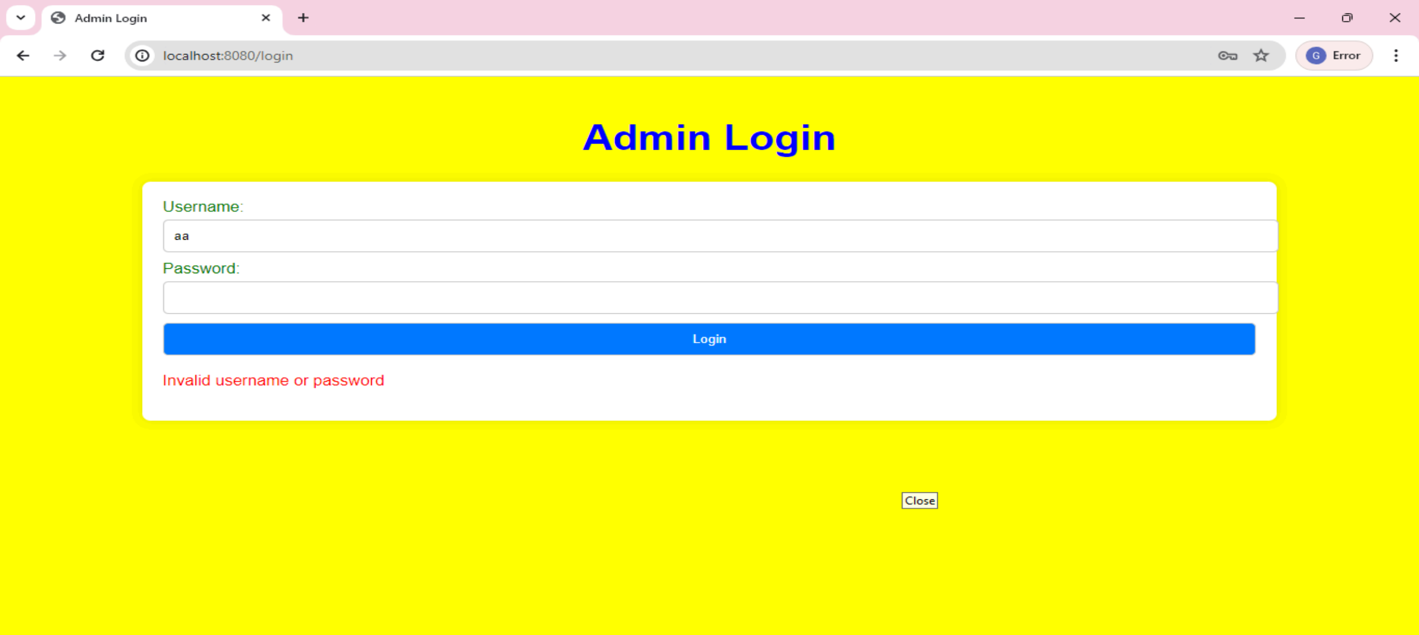
Home Page:



On clicking **Admin:**

a) **Admin Login:**

- Alert if invalid credentials given



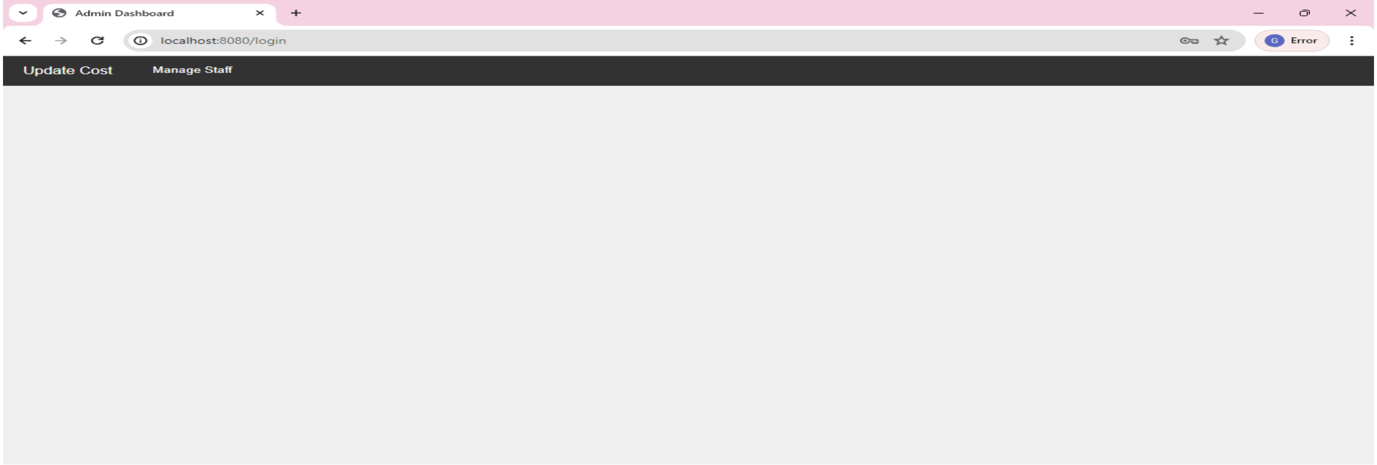
Admin Login

Username: aa

Password:

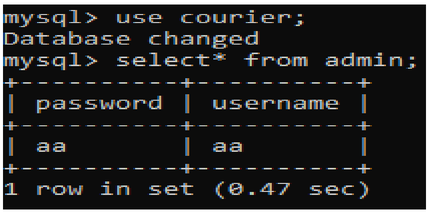
Invalid username or password

Land to admin dashboard on successful login



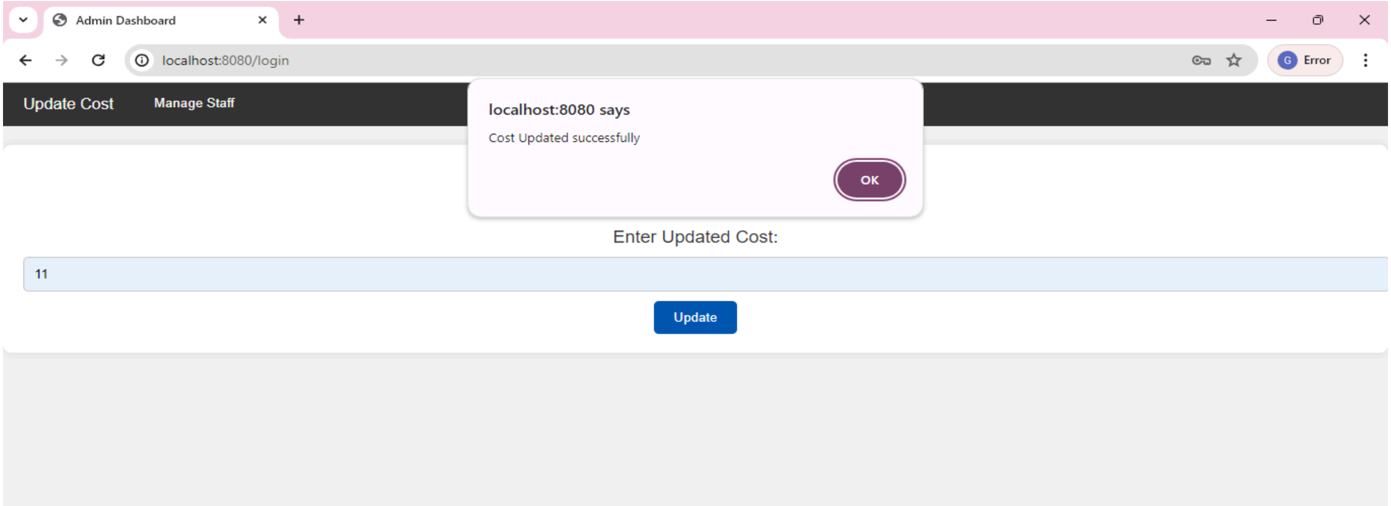
- Admin database, that stores admin credentials.

Assumption: admin is already registered and their credentials exist in the database.



```
mysql> use courier;
Database changed
mysql> select* from admin;
+----------+----------+
| password | username |
+----------+----------+
| aa       | aa       |
+----------+----------+
1 row in set (0.47 sec)
```

b)Update Cost functionality- admin enters the new cost per kg of the courier and the cost gets updated.



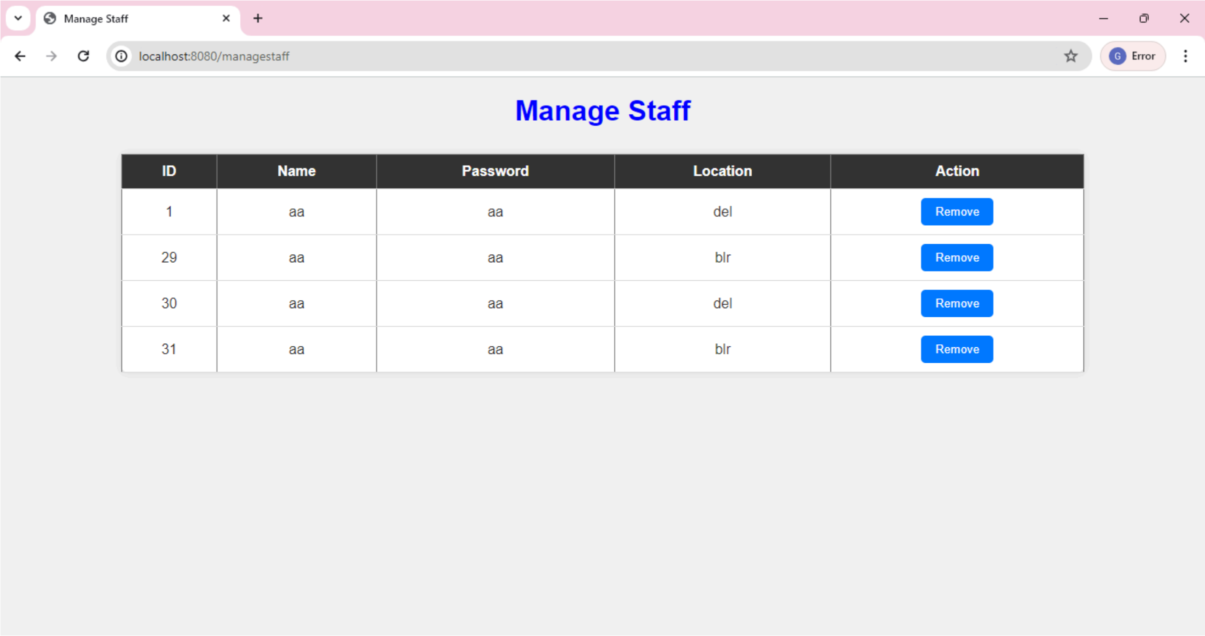
localhost:8080 says

Cost Updated successfully

Enter Updated Cost:

11

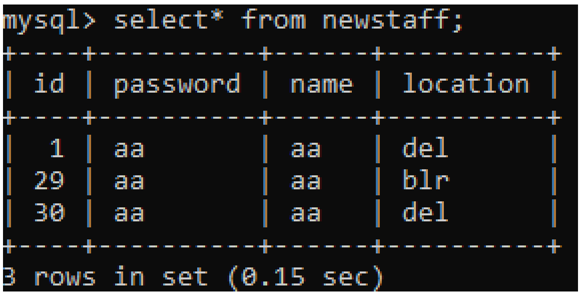
c)Manage Staff- admin can view the registered and active staffs at any time and can remove any unauthorised staff who can no longer login into the system and hence, maintain system security.



Manage Staff

ID	Name	Password	Location	Action
1	aa	aa	del	Remove
29	aa	aa	blr	Remove
30	aa	aa	del	Remove
31	aa	aa	blr	Remove

- On clicking on the “Remove” button against any staff credentials which the admins feels as unauthorised, that particular staff credentials are deleted from the staff database and that staff can no longer login.



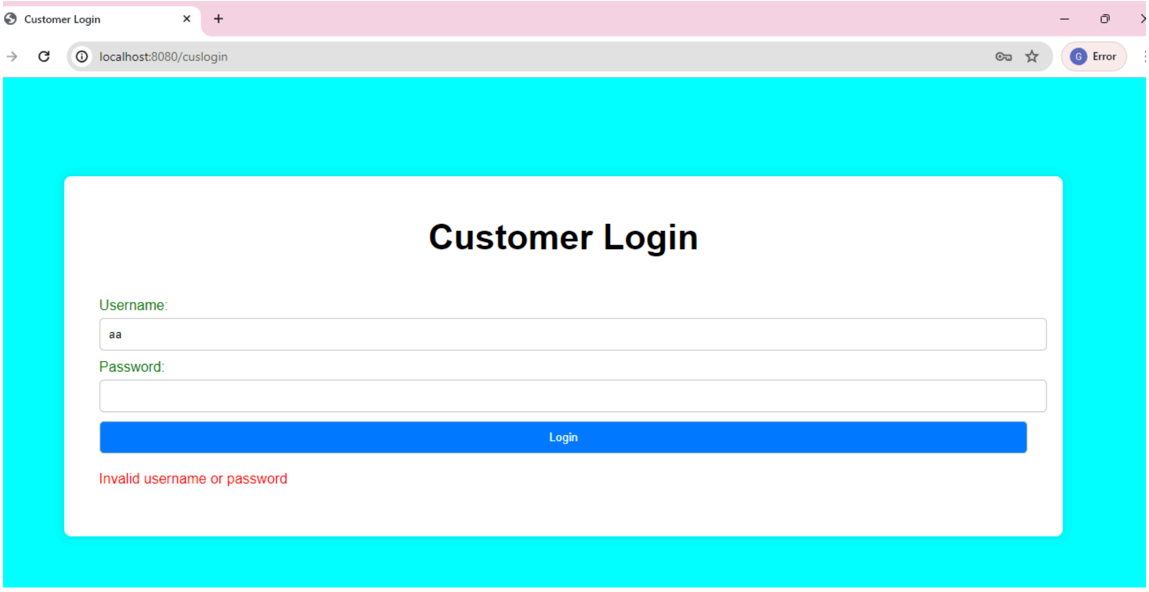
```
mysql> select* from newstaff;
+----+----------+------+----------+
| id | password | name | location |
+----+----------+------+----------+
|  1 | aa       | aa   | del      |
| 29 | aa       | aa   | blr      |
| 30 | aa       | aa   | del      |
+----+----------+------+----------+
3 rows in set (0.15 sec)
```

Modified database of staff

On clicking on **Customer**

a)Customer Login

- Alert if invalid credentials given



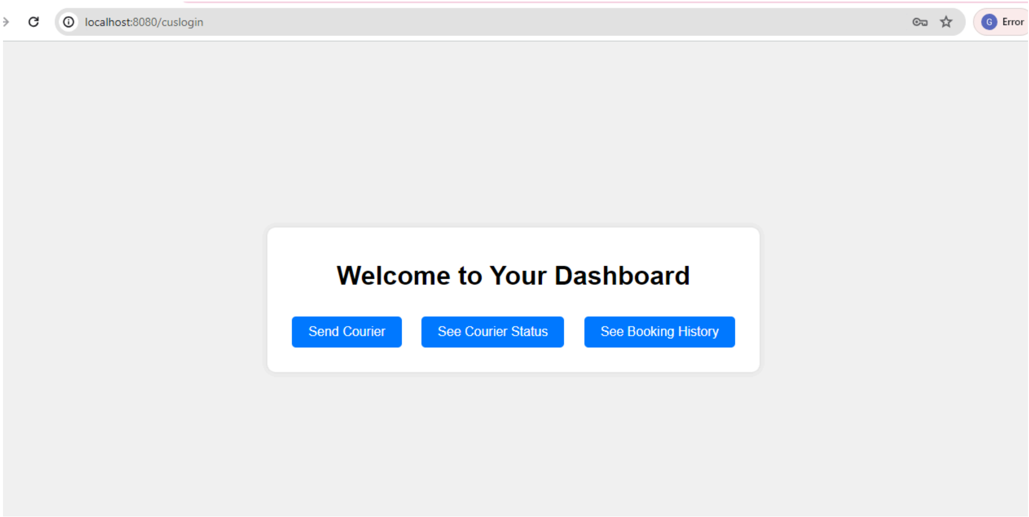
Customer Login

Username: aa

Password:

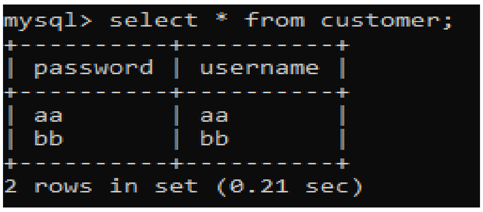
Invalid username or password

- Land to customer dashboard on successful login



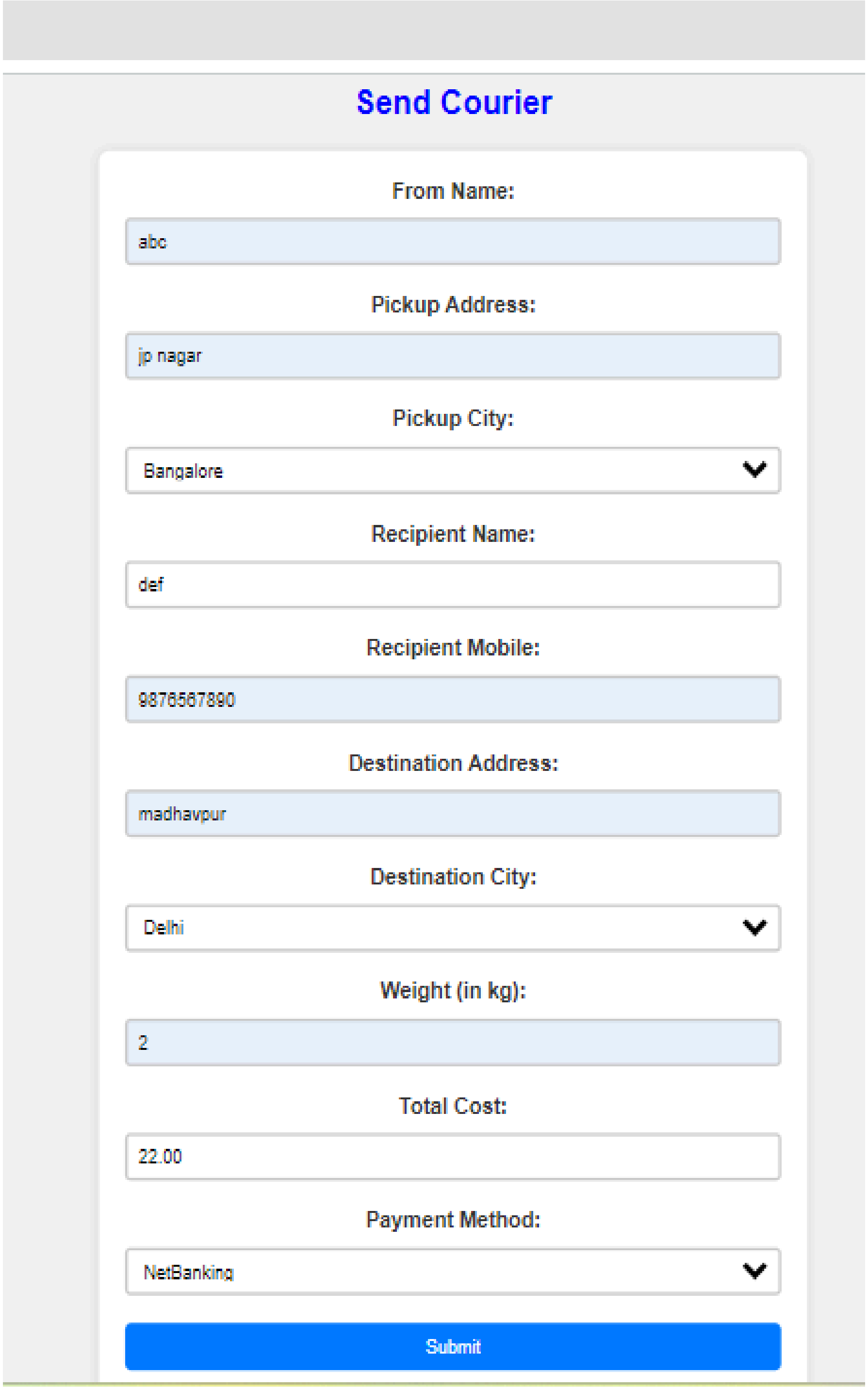
Welcome to Your Dashboard

- Customer database that stores customer credentials. **Assumption:** admin is already registered and their credentials exist in the database.



```
mysql> select * from customer;
+----------+----------+
| password | username |
+----------+----------+
| aa       | aa       |
| bb       | bb       |
+----------+----------+
2 rows in set (0.21 sec)
```

b)Send Courier



Send Courier

From Name: abc

Pickup Address: jp nagar

Pickup City: Bangalore

Recipient Name: def

Recipient Mobile: 9876567890

Destination Address: madhavpur

Destination City: Delhi

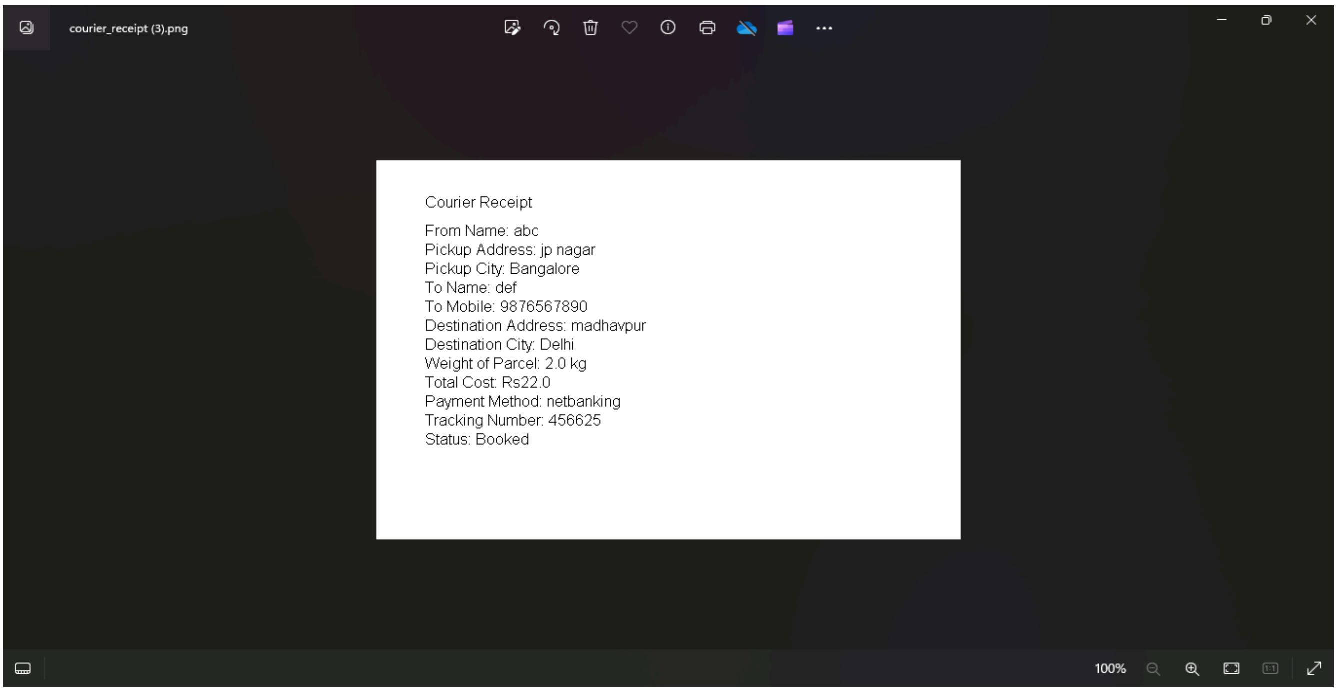
Weight (in kg): 2

Total Cost: 22.00

Payment Method: NetBanking

Customer enters all the required details for placing a courier.On entering the weight, the total cost is calculated automatically wrt to the updated cost set by admin .

- Courier receipt generated as an image,when clicked on submit button

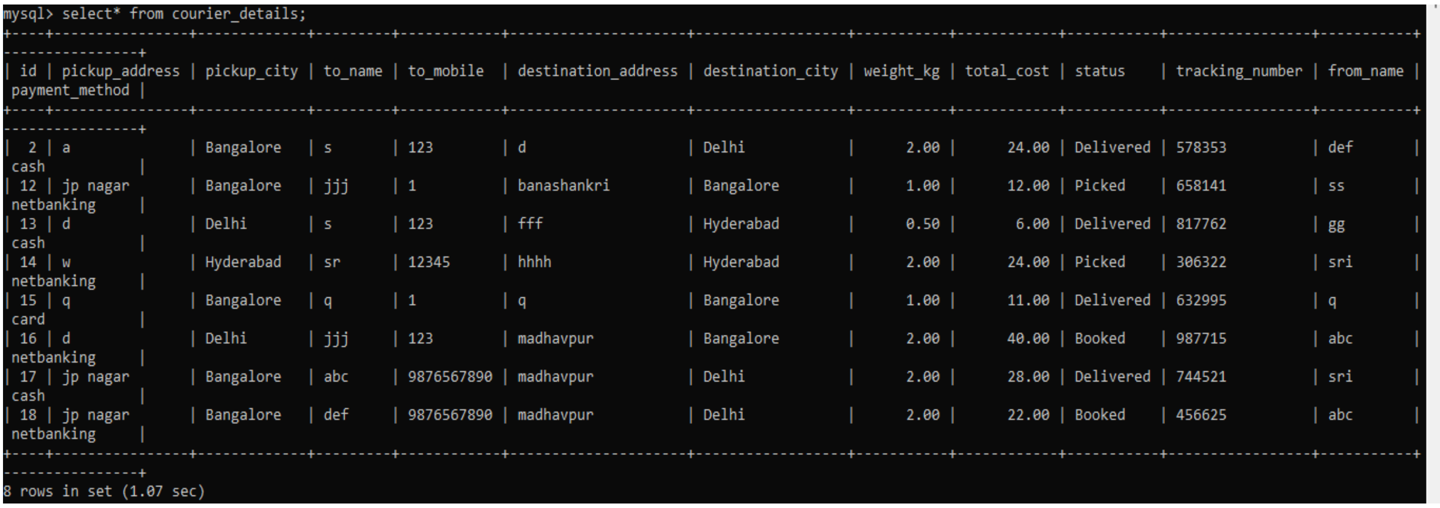


Courier Receipt

From Name: abc
Pickup Address: jp nagar
Pickup City: Bangalore
To Name: def
To Mobile: 9876567890
Destination Address: madhavpur
Destination City: Delhi
Weight of Parcel: 2.0 kg
Total Cost: Rs22.0
Payment Method: netbanking
Tracking Number: 456625
Status: Booked

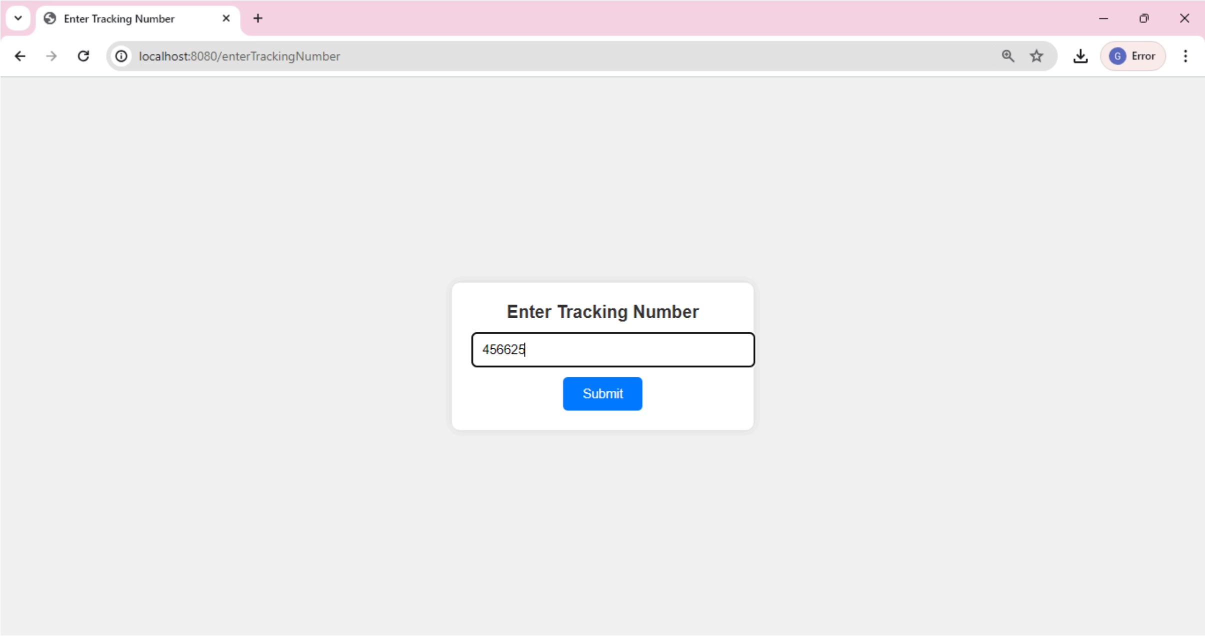
The courier status is booked.
A unique tracking number also generated .

- The courier details gets added into courier_detials database



```
mysql> select* from courier_details;
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
| id | pickup_address | pickup_city | to_name | to_mobile  | destination_address | destination_city | weight_kg | total_cost | status    | tracking_number | from_name |
 payment_method |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
|  2 | a              | Bangalore   | s       | 123        | d                   | Delhi            |      2.00 |      24.00 | Delivered | 578353          | def       |
 cash           |
| 12 | jp nagar       | Bangalore   | jjj     | 1          | banashankri         | Bangalore        |      1.00 |      12.00 | Picked    | 658141          | ss        |
 netbanking     |
| 13 | d              | Delhi       | s       | 123        | fff                 | Hyderabad        |      0.50 |       6.00 | Delivered | 817762          | gg        |
 cash           |
| 14 | w              | Hyderabad   | sr      | 12345      | hhhh                | Hyderabad        |      2.00 |      24.00 | Picked    | 306322          | sri       |
 netbanking     |
| 15 | q              | Bangalore   | q       | 1          | q                   | Bangalore        |      1.00 |      11.00 | Delivered | 632995          | q         |
 card           |
| 16 | d              | Delhi       | jjj     | 123        | madhavpur           | Bangalore        |      2.00 |      40.00 | Booked    | 987715          | abc       |
 netbanking     |
| 17 | jp nagar       | Bangalore   | abc     | 9876567890 | madhavpur           | Delhi            |      2.00 |      28.00 | Delivered | 744521          | sri       |
 cash           |
| 18 | jp nagar       | Bangalore   | def     | 9876567890 | madhavpur           | Delhi            |      2.00 |      22.00 | Booked    | 456625          | abc       |
 netbanking     |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
8 rows in set (1.07 sec)
```

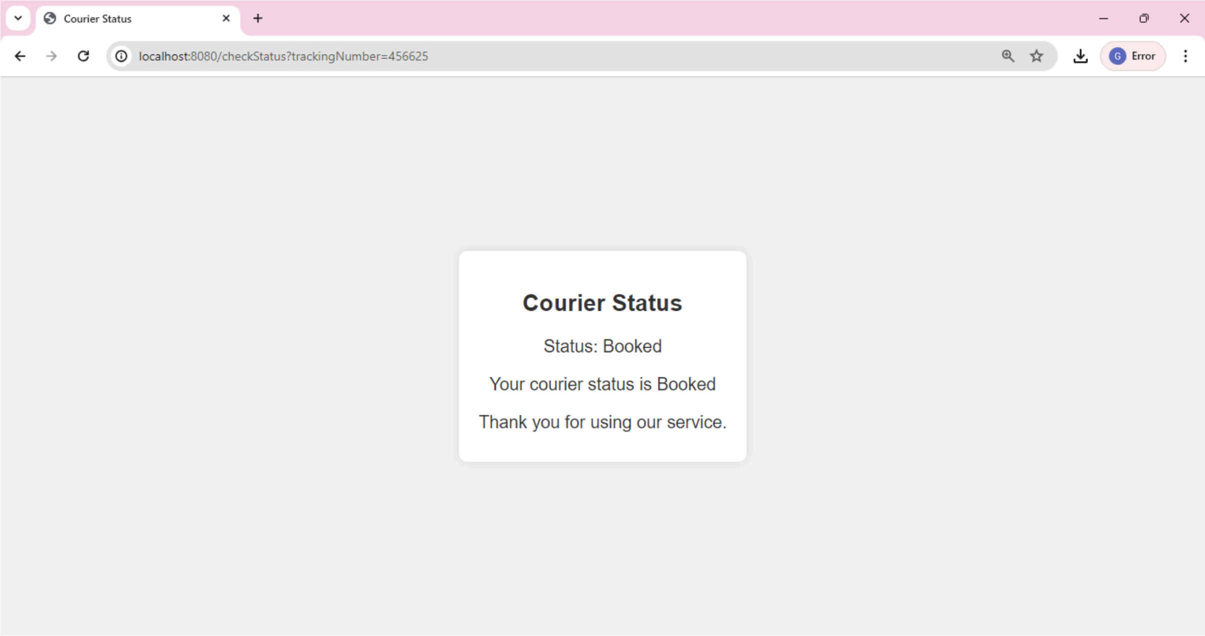
c) See courier Status- customer enters tracking number to view courier status



Enter Tracking Number

456625

On clicking submit,courier status is displayed.



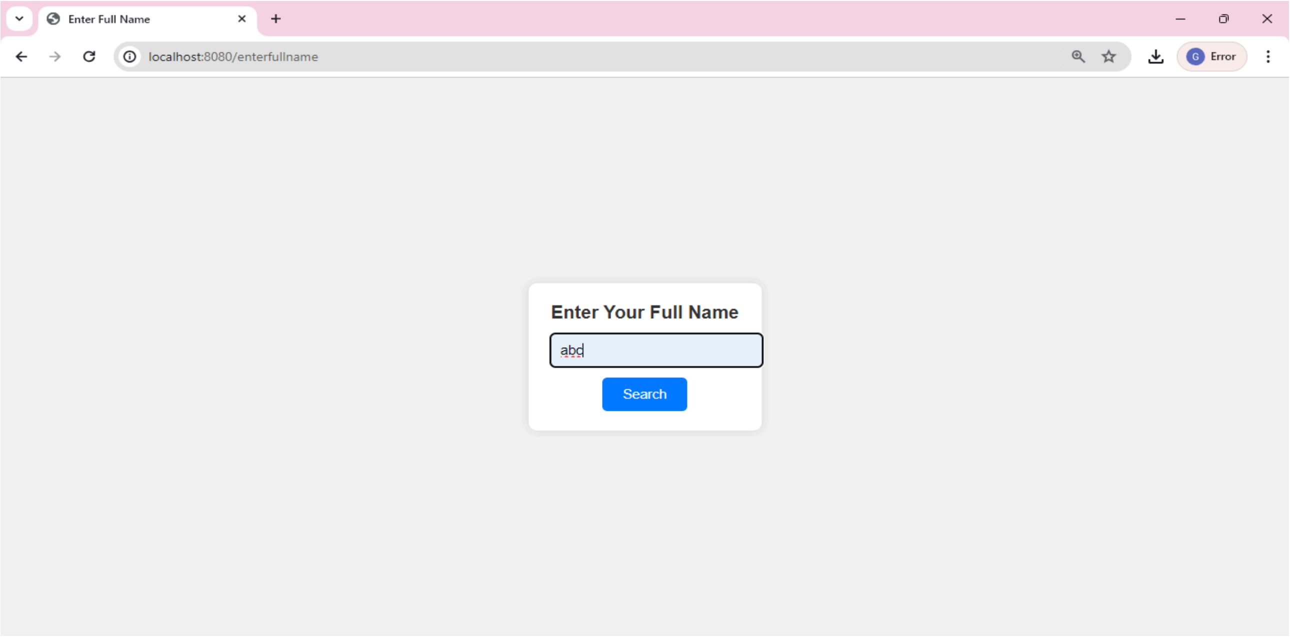
Courier Status

Status: Booked

Your courier status is Booked

Thank you for using our service.

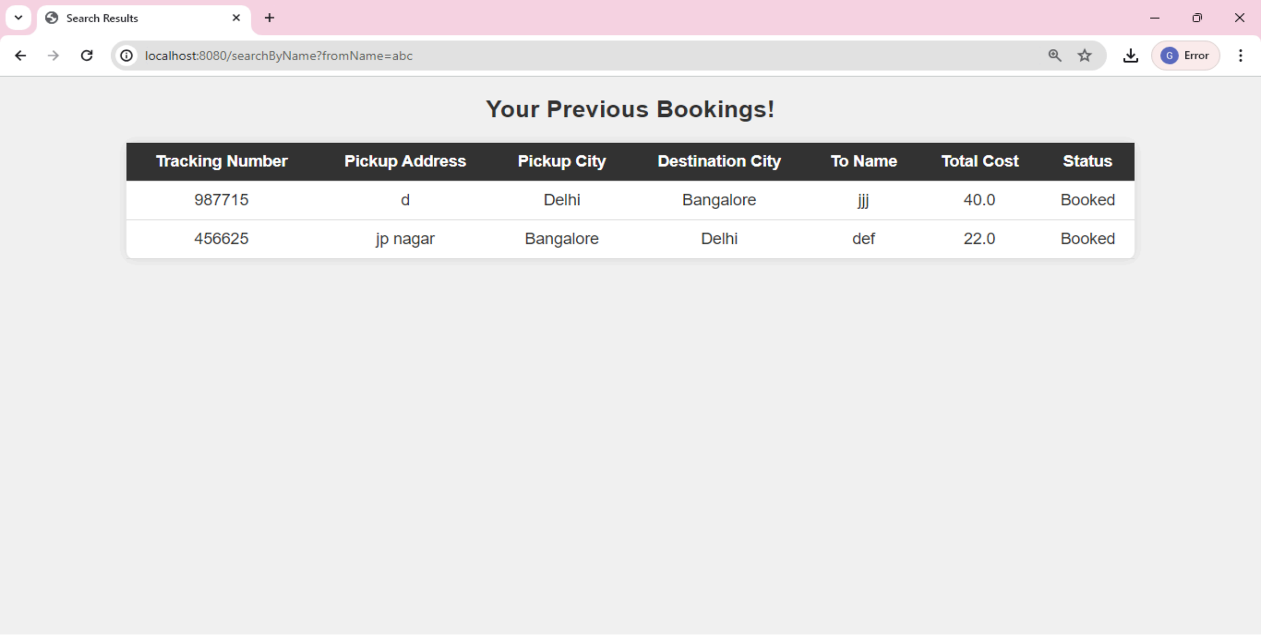
d)See Booking History- on clicking on ”see booking history” button on dashboard, customer enters their full name and can view all their placed couriers along with their current status.



Enter Your Full Name

abc

Search

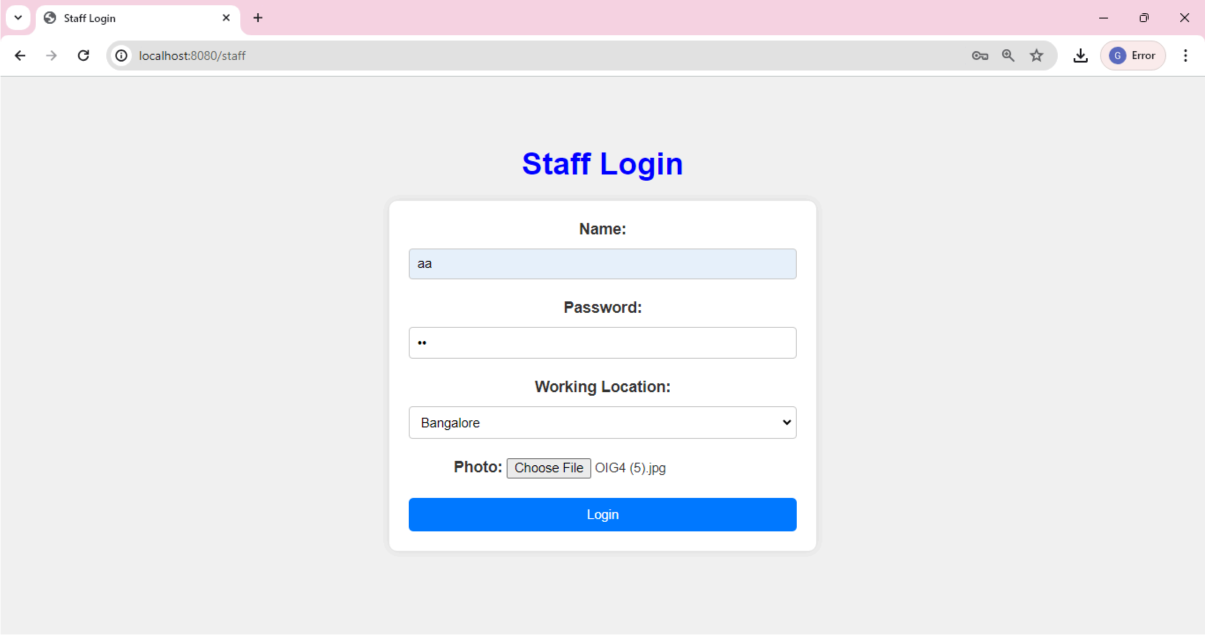


Your Previous Bookings!

Tracking Number	Pickup Address	Pickup City	Destination City	To Name	Total Cost	Status
987715	d	Delhi	Bangalore	jjj	40.0	Booked
456625	jp nagar	Bangalore	Delhi	def	22.0	Booked

On clicking on **Staff**

a)Staff login- staff enters their credentials & chooses their working location accordingly which they can pickup or deliver the courier.



Staff Login

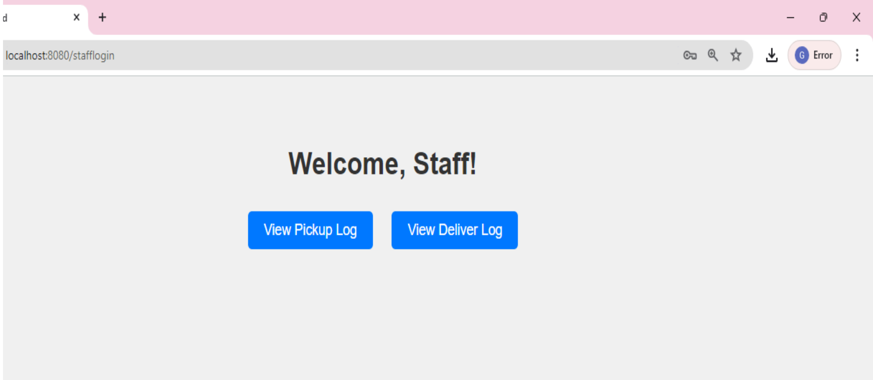
Name: aa

Password: ••

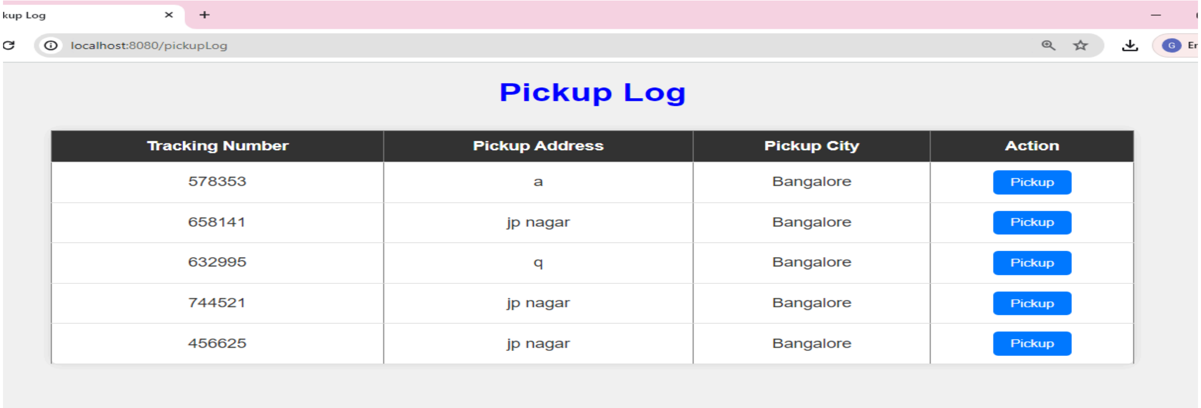
Working Location: Bangalore

Photo: Choose File OIG4 (5).jpg

On login, they can view pickup & delivery logs



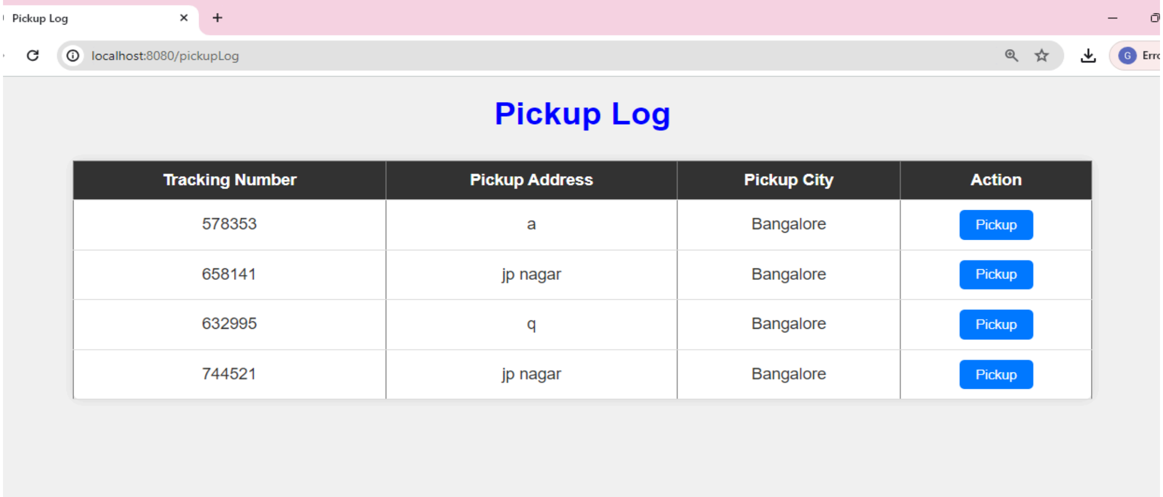
b) Pickup- On clicking on “view pickup log” button, all the couriers with pickup city that matches the staff’s working location are displayed.



Pickup Log

Tracking Number	Pickup Address	Pickup City	Action
578353	a	Bangalore	Pickup
658141	jp nagar	Bangalore	Pickup
632995	q	Bangalore	Pickup
744521	jp nagar	Bangalore	Pickup
456625	jp nagar	Bangalore	Pickup

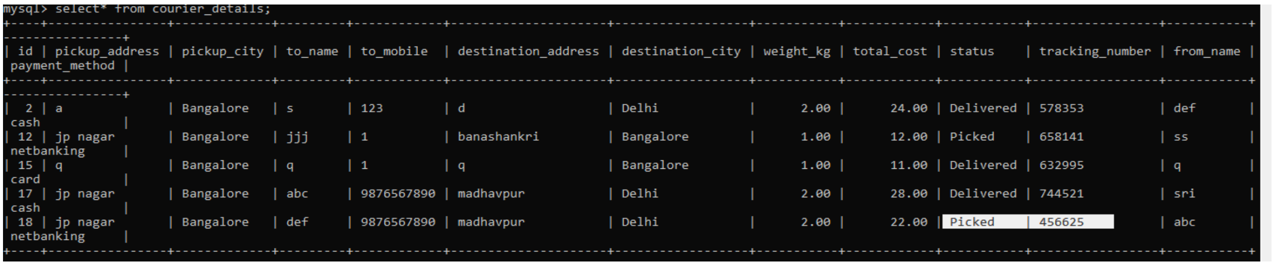
- On clicking on pickup button against any courier, its status gets changed to “Pickup”, removed from pickup log and the courier is added to deliver log.



Pickup Log

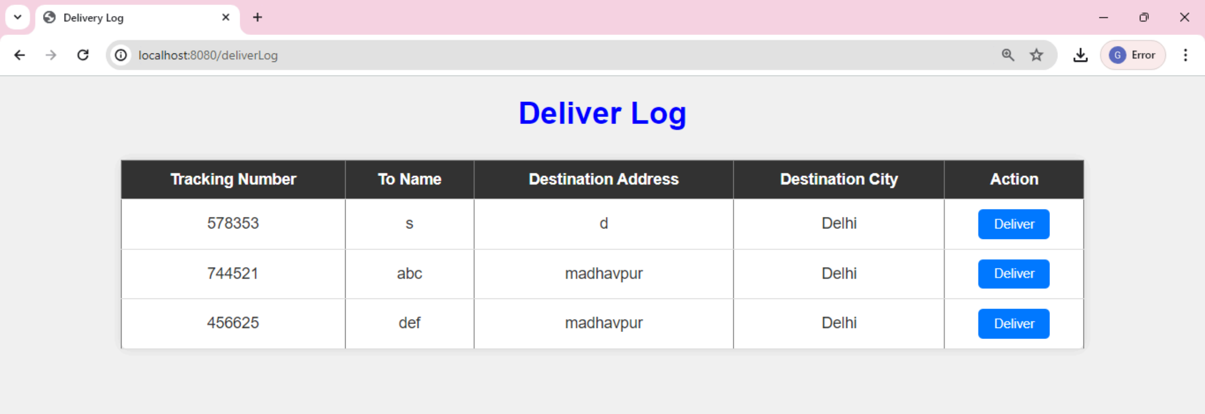
Tracking Number	Pickup Address	Pickup City	Action
578353	a	Bangalore	Pickup
658141	jp nagar	Bangalore	Pickup
632995	q	Bangalore	Pickup
744521	jp nagar	Bangalore	Pickup

The new courier placed, has been removed, with its status updated in the database too



```
mysql> select* from courier_details;
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
| id | pickup_address | pickup_city | to_name | to_mobile  | destination_address | destination_city | weight_kg | total_cost | status    | tracking_number | from_name |
 payment_method |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
|  2 | a              | Bangalore   | s       | 123        | d                   | Delhi            |      2.00 |      24.00 | Delivered | 578353          | def       |
 cash           |
| 12 | jp nagar       | Bangalore   | jjj     | 1          | banashankri         | Bangalore        |      1.00 |      12.00 | Picked    | 658141          | ss        |
 netbanking     |
| 15 | q              | Bangalore   | q       | 1          | q                   | Bangalore        |      1.00 |      11.00 | Delivered | 632995          | q         |
 card           |
| 17 | jp nagar       | Bangalore   | abc     | 9876567890 | madhavpur           | Delhi            |      2.00 |      28.00 | Delivered | 744521          | sri       |
 cash           |
| 18 | jp nagar       | Bangalore   | def     | 9876567890 | madhavpur           | Delhi            |      2.00 |      22.00 | Picked    | 456625          | abc       |
 netbanking     |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
```

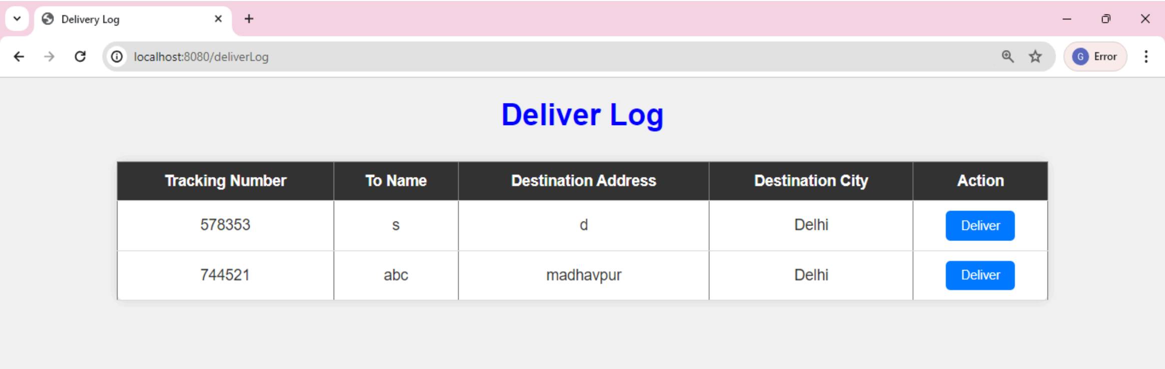
c)Deliver- on clicking on “view deliver log” button, all the couriers that need to be delivered are shown that match the staff’s working location.



Deliver Log

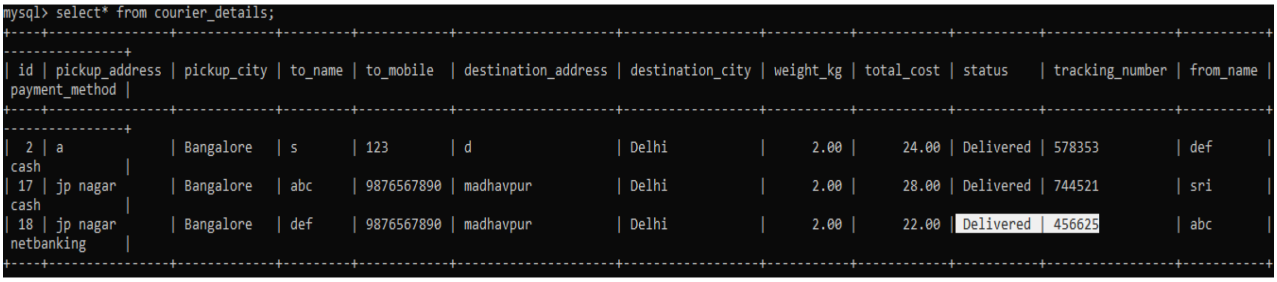
Tracking Number	To Name	Destination Address	Destination City	Action
578353	s	d	Delhi	Deliver
744521	abc	madhavpur	Delhi	Deliver
456625	def	madhavpur	Delhi	Deliver

- On clicking on “Deliver” button against any courier, its removed from delivery log & its status is updated in the database too.



Deliver Log

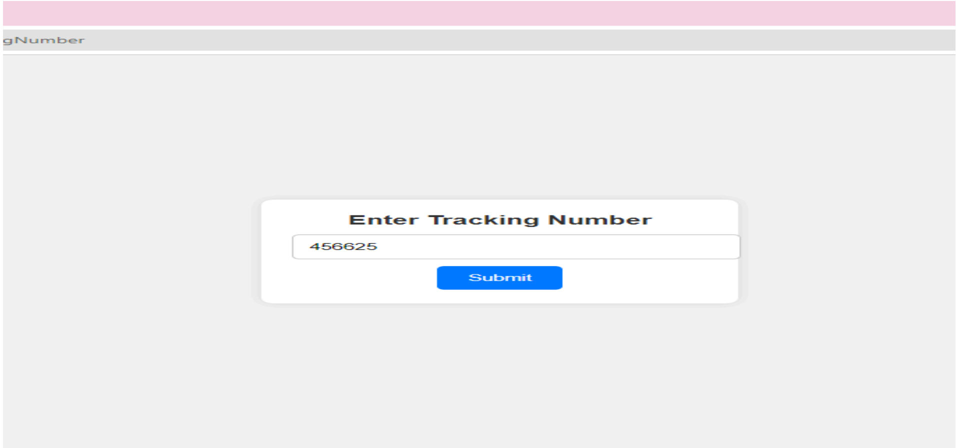
Tracking Number	To Name	Destination Address	Destination City	Action
578353	s	d	Delhi	Deliver
744521	abc	madhavpur	Delhi	Deliver



```
mysql> select* from courier_details;
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
| id | pickup_address | pickup_city | to_name | to_mobile  | destination_address | destination_city | weight_kg | total_cost | status    | tracking_number | from_name |
 payment_method |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
----------------+
|  2 | a              | Bangalore   | s       | 123        | d                   | Delhi            |      2.00 |      24.00 | Delivered | 578353          | def       |
 cash           |
| 17 | jp nagar       | Bangalore   | abc     | 9876567890 | madhavpur           | Delhi            |      2.00 |      28.00 | Delivered | 744521          | sri       |
 cash           |
| 18 | jp nagar       | Bangalore   | def     | 9876567890 | madhavpur           | Delhi            |      2.00 |      22.00 | Delivered | 456625          | abc       |
 netbanking     |
+----+----------------+-------------+---------+------------+---------------------+------------------+-----------+------------+-----------+-----------------+-----------+
```

Status updated in database too.

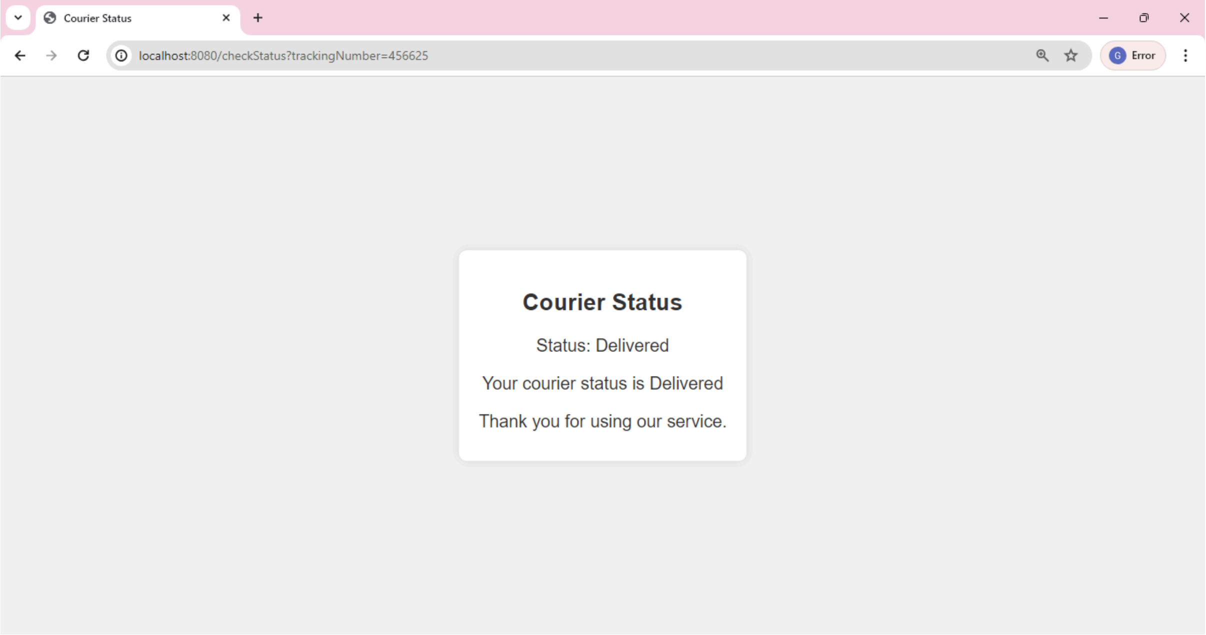
- The customer can enter the tracking number of courier again to view its status



Enter Tracking Number

456625

Submit



Courier Status

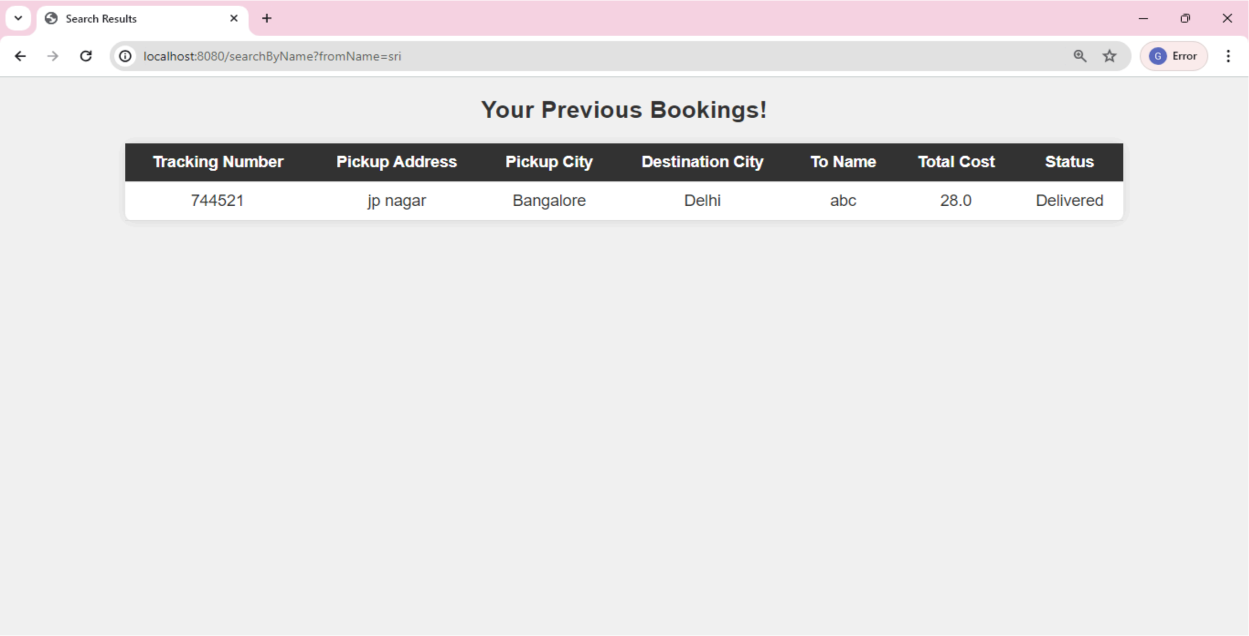
Status: Delivered

Your courier status is Delivered

Thank you for using our service.

The status shown is “Delivered”

- The customer can view their previous booking and the current status of the courier is shown



Your Previous Bookings!

Tracking Number	Pickup Address	Pickup City	Destination City	To Name	Total Cost	Status
744521	jp nagar	Bangalore	Delhi	abc	28.0	Delivered

GITHUB LINK OF PROJECT

https://github.com/gantasrilaitha/courier-management-java

Individual contributions of the team members

Gajendra S PES1UG21CS197
State Diagrams, Activity Diagram, Design Principles, and Design Pattern, Front-end of the Project and Java code

Ganta Srilalitha PES1UG21CS200
Class diagram, Modified class Diagram, Architecture Patterns, Design Principles, and Design Pattern, Back-End and Java code of the Project

Gowtham S PES1UG21CS210
Activity Diagrams, Class diagram, Modified class Diagram, Design Principles, and Design Pattern, Back-End and Java code of the Project

H M Bharath PES1UG21CS213
Use Case diagram , Activity Diagrams, Design Principles, Front-end of the Project