OOP

by Kapil Pokhrel

Submission date: 13-Aug-2021 02:15AM (UTC+0800)

Submission ID: 1630685397

File name: Java_Assignment_V1.docx (904.63K)

Word count: 2324

Character count: 12073

Acknowledgment

This project has taken a lot of time and effort to complete. This initiative would not have been possible without the collaboration and generous assistance of many people, as well as the **Infomax College of IT and Management**. We'd want to extend our gratitude to everyone who has been so generous to us.

Mr. Sushil Adhikari deserves a special mention for his guidance and regular monitoring, as well as for supplying important project information and his assistance in finishing the project.

Without the unwavering support and encouragement of our parents and friends , we would not have been able to complete this project.

Sincerely,

Shivam Ranabhat(NPI000047)

Kapil Pokhrel(NPI000030)

Contents

1. Introduction

The whole process of developing a Courier Service System is described in this report. For users, a Courier Service System maintains real-world things like as orders and items. It simplifies and streamlines the whole distribution and tracking of items. Easier in the sense that everything from placing an order to processing it is done digitally, reducing the amount of physical labour. This application allows customers to customize their delivery and manage their orders from anywhere. This will save time by reducing the amount of paper used for printing and entering manual entries.This project is completed in NetBeans IDE with GUI. All the data are stored using text files.

2. Assumptions

There are two categories of users who can utilize this system:

I. Managing Staff

The system's most powerful users are the management staff. Access is granted to Managing Staff via a unique user ID and password. They have complete access to the whole system. Managing Staff has the ability to add delivery staff, edit them, and handle their orders. The order, profiles, reports, and feedbacks may be added, viewed, modified, and deleted by the managing staff. The management personnel may also monitor or follow the order, as well as accept or reject it.

II. Delivery Staff

The system is accessible to delivery staff for registering and making deliveries. Every Delivery Staff is given an order detail by the managing staff. Every member of the delivery team has a name, address, and phone number that must be entered into the system. Employees that work in delivery can change their profiles. They have access to order and delivery information. The delivery staff may update the status of orders and view feedback.

3. Objectives

- To Reduce data redundancy by simplifying the ordering and delivery procedures.
- To decrease the time spent by staff writing orders when it isn't necessary.

4. Design Diagrams

a. Use Case Diagram

Use Case Diagram is one of the UML diagrams that has been utilized for this system. It provides a realistic overview of the system's actors, as well as the different functions that the actors demand and how these functions interact.

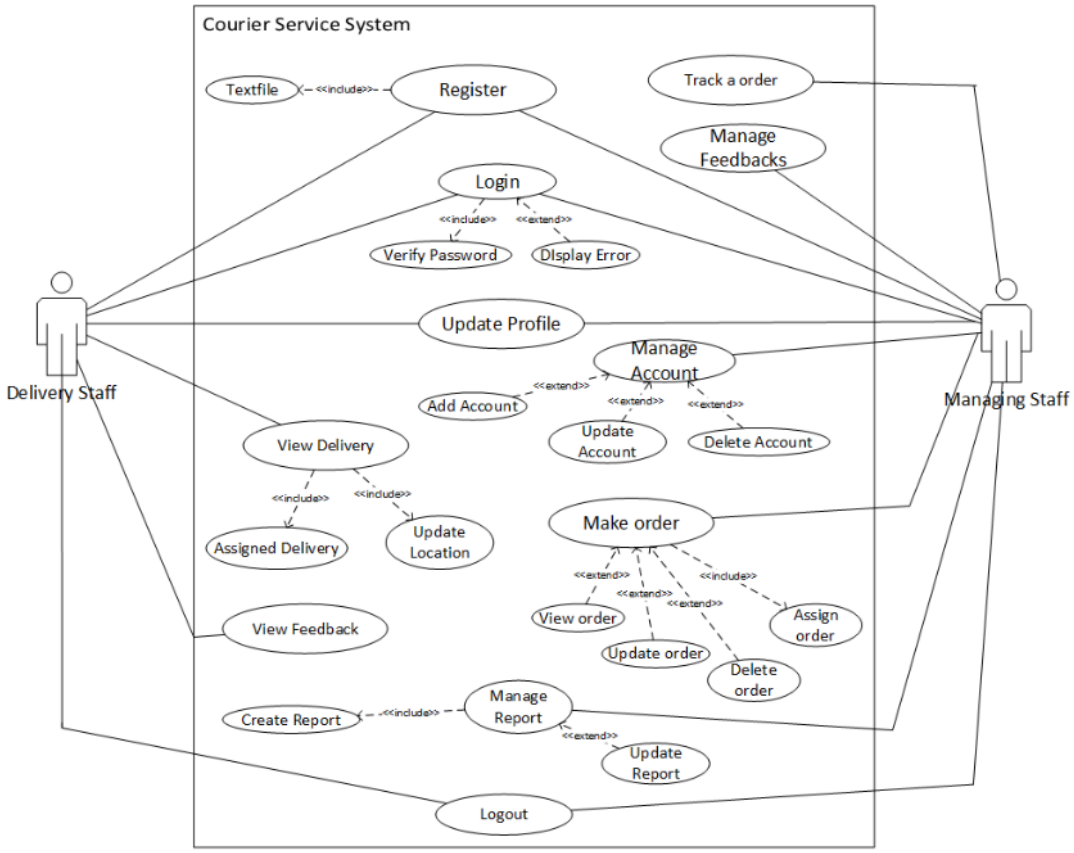


Fig: Use case diagram

Description

1. **Use case: Register**

   Description: If the user has no account, then according to their user type, they can register their account.

   Actors: Managing staff, Delivery staff

   Main flow:

   a. Users select login menu and select their user type.
   b. User select the No account register here.
   c. User provides all the information they are asked to provide.
   d. The system will store the information into text file as a database.

   Post conditions: User registered successfully

2. **User case:** Login

   Description: Only registered users are allowed to enter into the system

   Actors: Managing staff, Delivery staff

   Main flow:

   a. Users select login menu and select the type of user.
   b. Then according to their type of login page will display.
   c. User enters their username and password.
   d. Then the system will verify the user from the information stored in text file

   Post conditions: Login successfully

3. **Use case:** Update profile

   Brief description: Both users are allowed to update their profile.

   Actors: Managing staff, Delivery staff

   Main flow:

   a. After login successfully, user click the profile button.
   b. All the details about the profile are displayed.
   c. User change/update the profile.

Post conditions: User data updated successfully

4. **Use case:** Manage account

   Brief description: Only managing staff are allowed to manage the account of delivery staff i.e., they can add user, update user's information, delete user.

   Actors: Managing staff

   Main flow:

   a. User Account page is displayed.
   b. Managing staff enter the id of delivery staff.
   c. Information is displayed.
   d. Managing staff add, update or delete user.

   Post conditions: User account created; User removed successfully

5. **Use case:** Make order

   Brief description: Only managing staff will create order, and update, delete and assign the created order.

   Actors: Managing staff

   Main flow:

   a. Order menu is displayed.
   b. Managing staff create order.
   c. Managing staff select order management menu
   d. Managing staff view order to update, delete and assign order

      .

   Post conditions: Order created successfully, Order updated successfully, Order deleted successfully, Order assigned successfully.

6. **Use case:** View delivery

   Brief description: Delivery staff can view delivery details assigned to them.

   Actors: Delivery staff

   Main flow:

   a. Delivery staff selects view delivery and GUI is displayed.

   b. Delivery staff enters id to view assigned delivery details.

   c. Delivery staff update location for tracking.

   Post conditions: Delivery found for you; Location updated for tracking.

7. **Use case:** Manage report

   Brief description: Only managing staff can create report based upon the delivery details which have been delivered. Managing staff can update, delete report.

   Actors: Managing staff

   Main flow:

   a. Create report GUI is displayed.

   b. Managing staff create report from the table which contains details of delivered items.

   c. Managing staff selects report management GUI is displayed.

   d. Managing staff update, delete reports.

   Post conditions: Report created successfully, Report updated successfully, Report deleted successfully.

8. **Use case:** View feedback

   Brief description: Delivery staff can also view feedback provided by customer.

   Actors: Delivery staff

   Main flow:

   a. View feedback GUI is displayed.

   b. Delivery staff clicks view button.

   c. Feedback is displayed in a table.

   Alternative

a. No feedback is added.

Post conditions: Feedback found.

9. **Use case:** Manage feedback

   Brief description: Managing staff can create feedback according to the customer's opinion, and update/delete the created feedback.

   Actors: Managing staff

   Main flow:

   a. Managing staff selects the create feedback section.

   b. Create feedback GUI is displayed.

   c. Managing staff enter all the details to create feedback.

   d. Managing staff select back option and go to feedback management GUI.

   e. Now, managing staff clicks view button and add, update and delete feedback.

   Post conditions: Feedback created successfully, Feedback updated successfully, Feedback deleted successfully.

10. **Use case:** Track order

    Description: Only managing staffs can tack the location using order id.

    Actors: Managing staffs

    Main flow:

    a. Managing staff select tracking option.

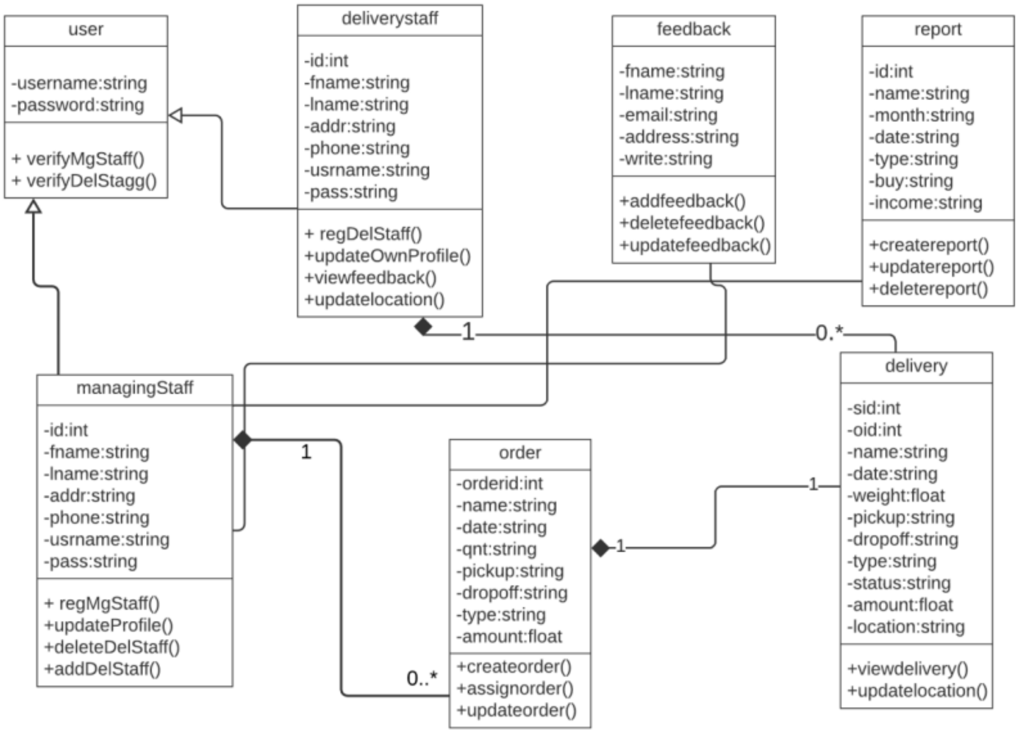
    b. Managing staff enters order id.

    c. Details of order along with current location is displayed.

    Post condition: Invalid id

b. Class Diagram

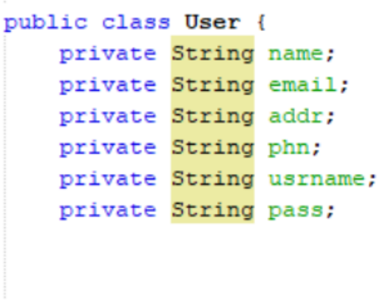
The basic building block of every object-oriented programming design is the class diagram. It shows the framework's classes, as well as the features and tasks of each one, as well as the relationships between them. The following is the class diagram that was created for the system:



5. Object Oriented Programming Concept

a. Class

In java a class is a group of objects with common characteristics and also called a blueprint or template using which objects are created. Similarly, a class in java contains different fields, methods, constructor and so on (Nayan, 2021).



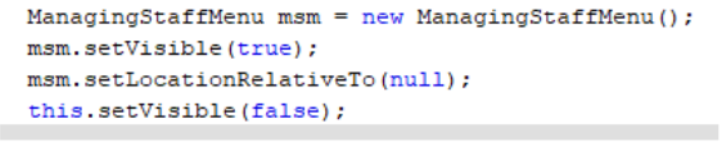
```
public class User {
    private String name;
    private String email;
    private String addr;
    private String phn;
    private String usrname;
    private String pass;
```

Fig: User class with private access specifiers

In above class we have we used private data members which can be assessed inside the same class which makes the personal information more secure like username and password.

b. Object

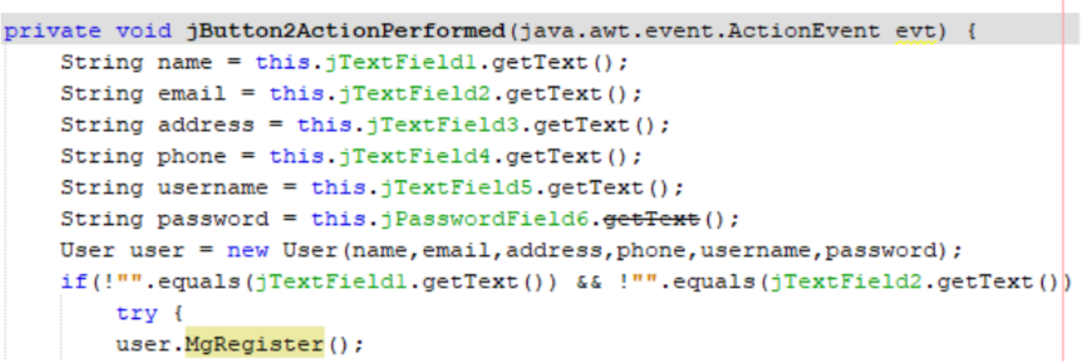
An object in java is an instance of a class which includes data structures, variable or functions (Rabelo,2021). While developing courier service system we can an object to call another class.



```
ManagingStaffMenu msm = new ManagingStaffMenu();
msm.setVisible(true);
msm.setLocationRelativeTo(null);
this.setVisible(false);
```

Fig: Object used to call another class and display the GUI

In above class we created an object called “msm” using “ManagingStaffMenu” class to display the menu for managing staff. Similarly, we created different object to different class and display the desired GUI.

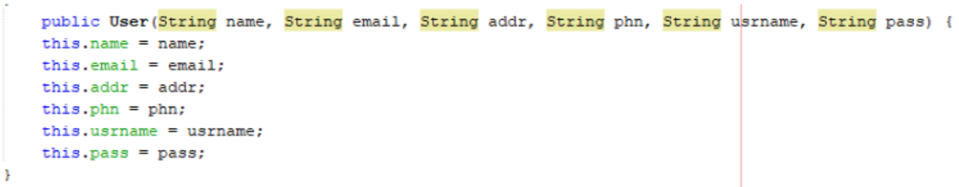


```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    String name = this.jTextField1.getText();
    String email = this.jTextField2.getText();
    String address = this.jTextField3.getText();
    String phone = this.jTextField4.getText();
    String username = this.jTextField5.getText();
    String password = this.jPasswordField6.~~getText~~();
    User user = new User(name,email,address,phone,username,password);
    if(!"".equals(jTextField1.getText()) && !"".equals(jTextField2.getText())
        try {
        user.MgRegister();
```

Fig: Object to call the constructor of class user

c. Constructor

A constructor in java is like method and it is call when an object is created. A constructor should have same name as that of the class where it is created. Similarly, a constructor does not have return type (Jenkov,2021). Similarly, we have created a parameterized constructor to create the object of class user as shown below:

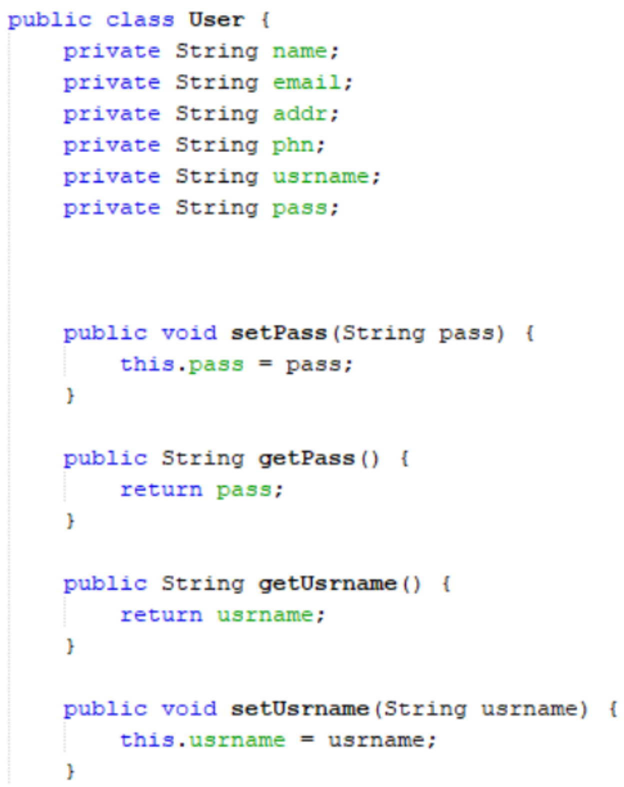


```
    public User(String name, String email, String addr, String phn, String usrname, String pass) {
    this.name = name;
    this.email = email;
    this.addr = addr;
    this.phn = phn;
    this.usrname = usrname;
    this.pass = pass;
}
```

Fig: Parameterized constructor of class User

d. Encapsulation

In encapsulation, the variables of class are hidden from other classes and can be assessed only by using methods of current class which is called data hiding. Similarly, we have used the concept of encapsulation by creating private variables and access them using get method which provides read only access and set method which provide write only access as shown below:

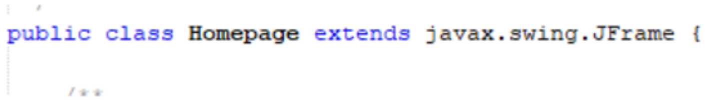


```
public class User {
    private String name;
    private String email;
    private String addr;
    private String phn;
    private String usrname;
    private String pass;



    public void setPass(String pass) {
        this.pass = pass;
    }

    public String getPass() {
        return pass;
    }

    public String getUsrname() {
        return usrname;
    }

    public void setUsrname(String usrname) {
        this.usrname = usrname;
    }
```

Figure: Accessing the private variables using getter and setter method

e. Inheritance

In object-oriented concept, inheritance represents a IS-A relationship between parent class and child class. It generally uses word “extends”.



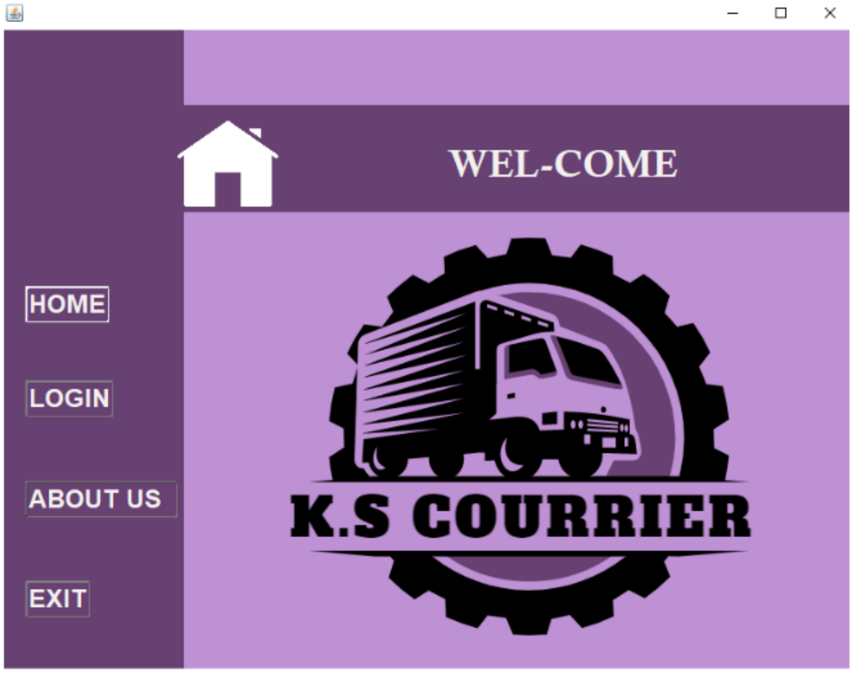
```
public class Homepage extends javax.swing.JFrame {

    /**
```

Fig: Inheritance

In above class, homepage is inherit javax.swing.JFrame class. It inherits some methods of parent class.

6. Sample Output

a. Main Menu



\

Figure: Main Interface

b. Users Menu

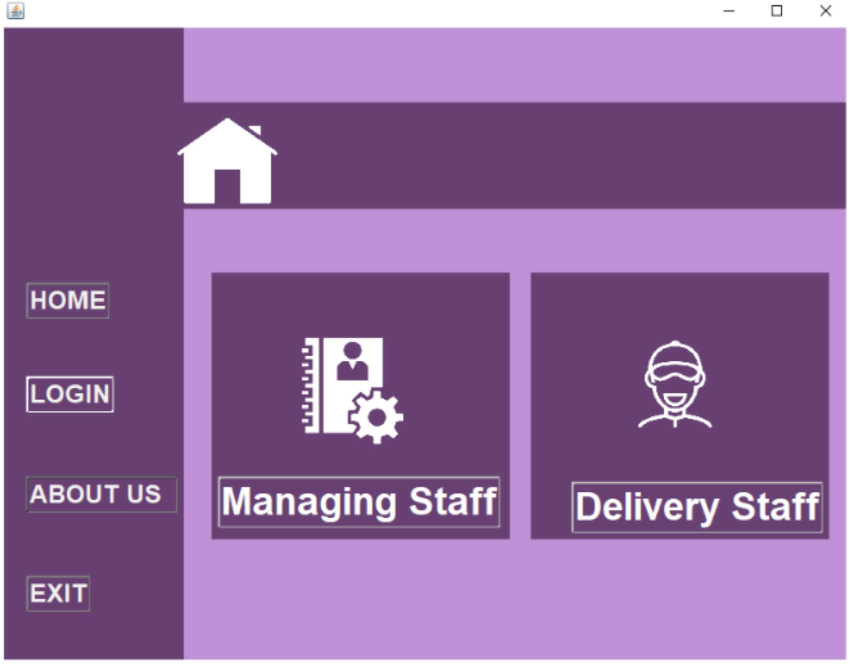


Fig: Menu for selecting user type

In the above page the user needs to select their type of user so that they can use their functionalities.

c. Managing Staff Login

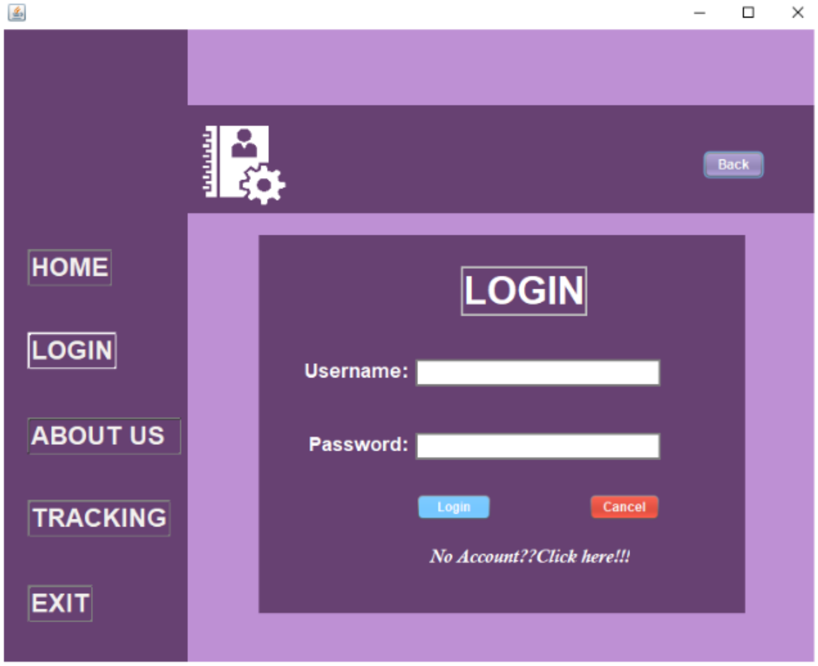


Fig: Managing staff login

This is the login GUI of managing staff where a managing staff should enter their username and password to use the managing functionalities.

d. Delivery Staff Login

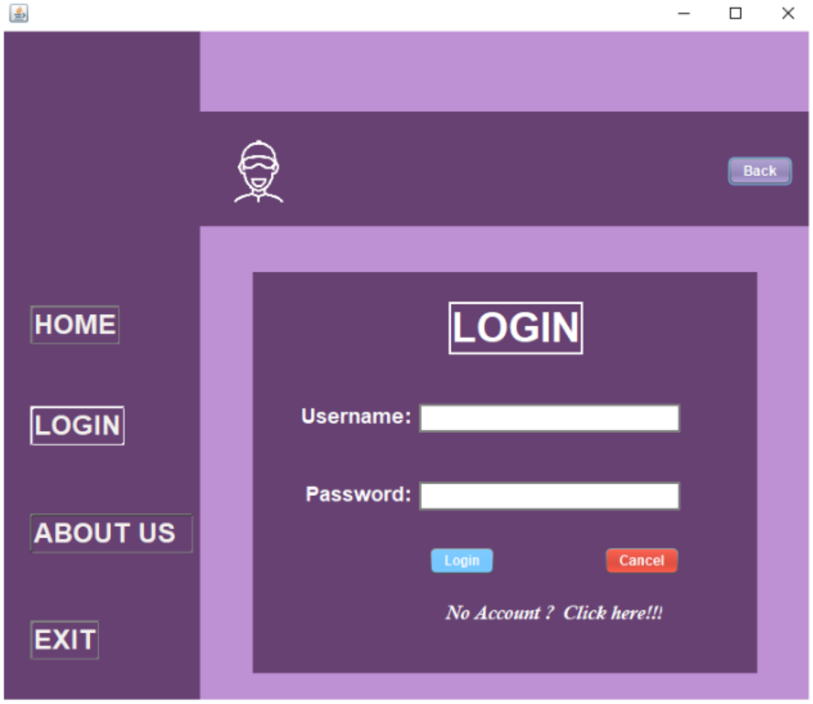


Fig: Delivery staff login

Similarly, this is the login GUI of delivery staff where the delivery staff enters their username and password to enter delivery staff menu.

e. Managing Staff Register

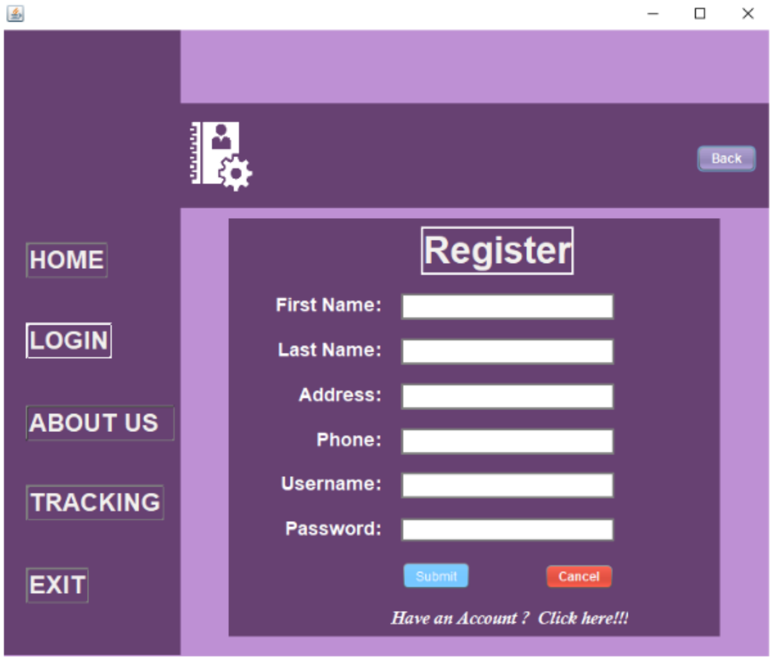


Fig: Managing staff register

This is the register GUI of managing staff where a managing staff provide his/her information to create a managing staff account.

f. Managing Staff Profile

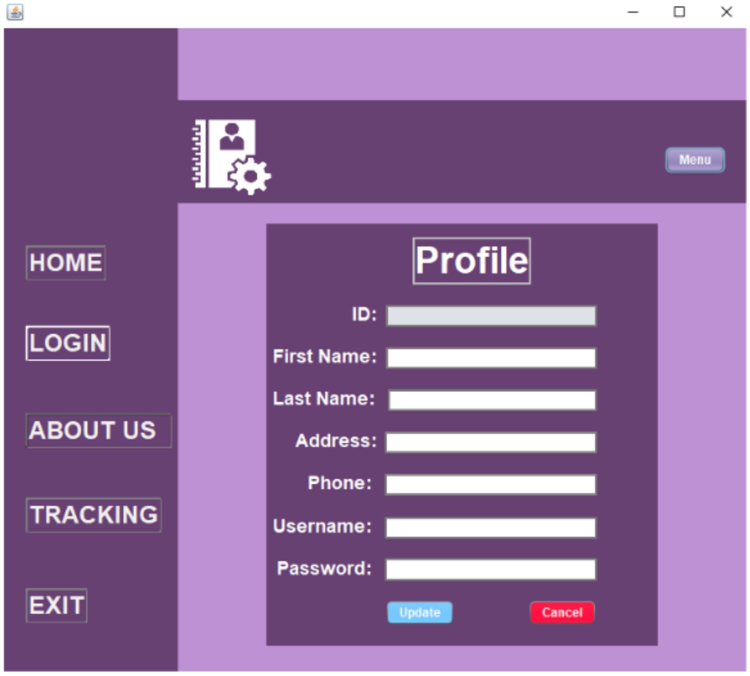


Fig: Managing staff profile

This is profile of a user where a user can update his/her information.

g. Managing Staff Menu

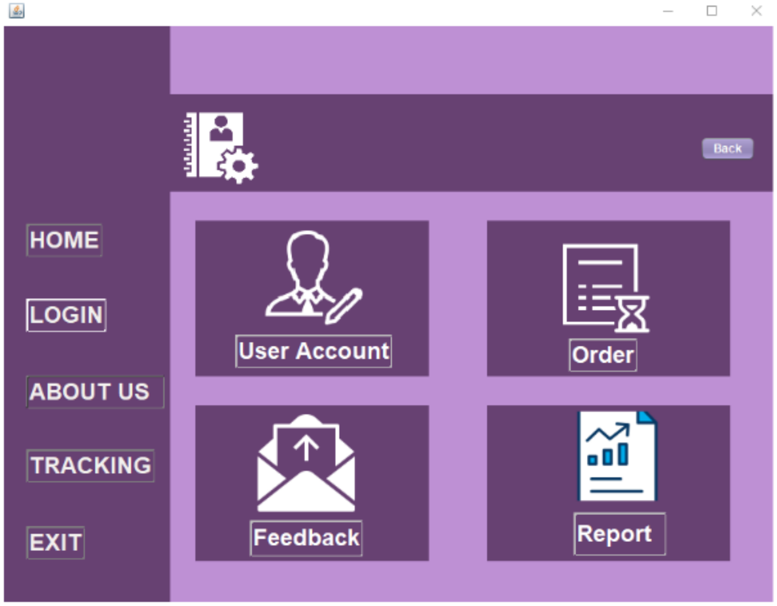


Fig: Managing staff menu

This GUI is managing staff menu where a managing staff can handle all the available functionalities.

h. Order Menu

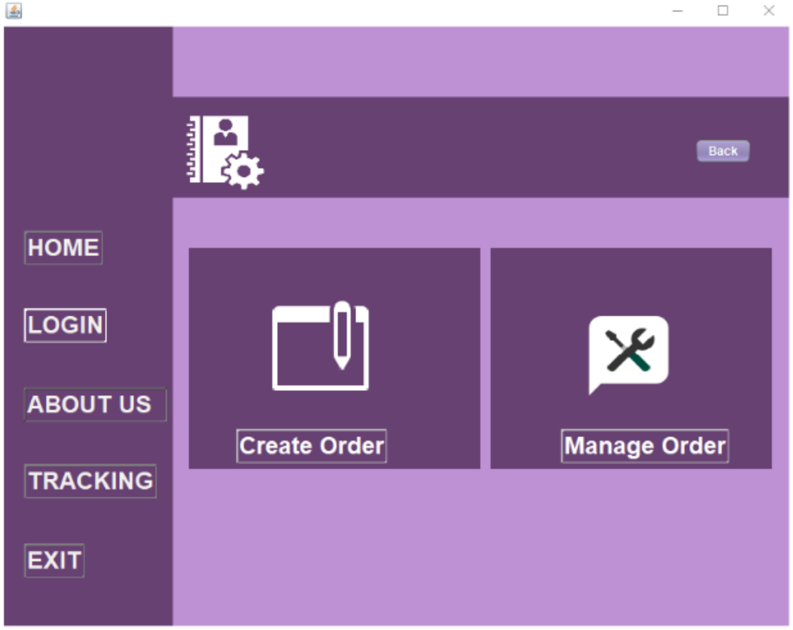
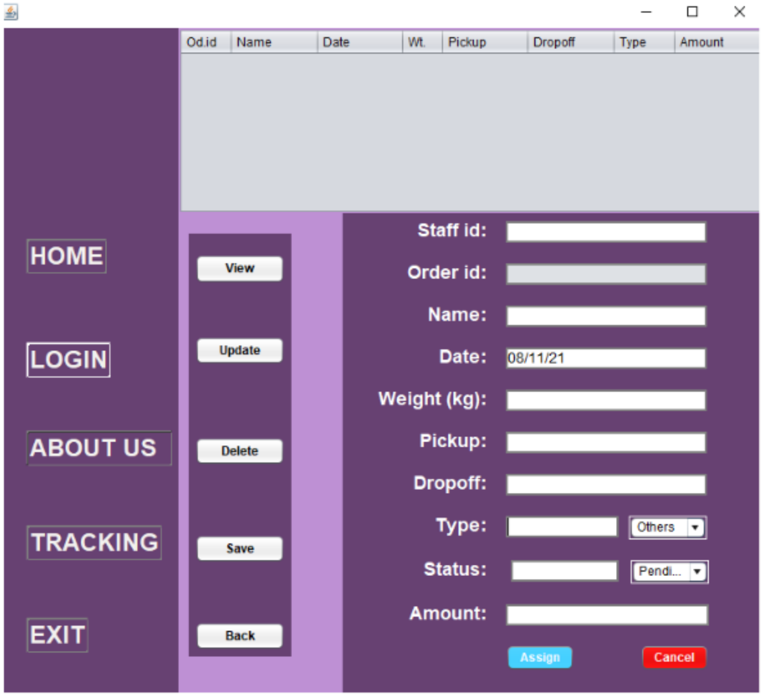


Fig: Order Menu

In this GUI a managing staff can select the functionalities available for order management.

i. Order Management Menu



Od.id	Name	Date	Wt.	Pickup	Dropoff	Type	Amount

Staff id:

Order id:

Name:

Date: 08/11/21

Weight (kg):

Pickup:

Dropoff:

Type: Others

Status: Pendi...

Amount:

Fig: Order management menu

In this GUI a managing staff can manage the order details like update, delete and assign the order to the delivery staff.

j. Feedback Menu

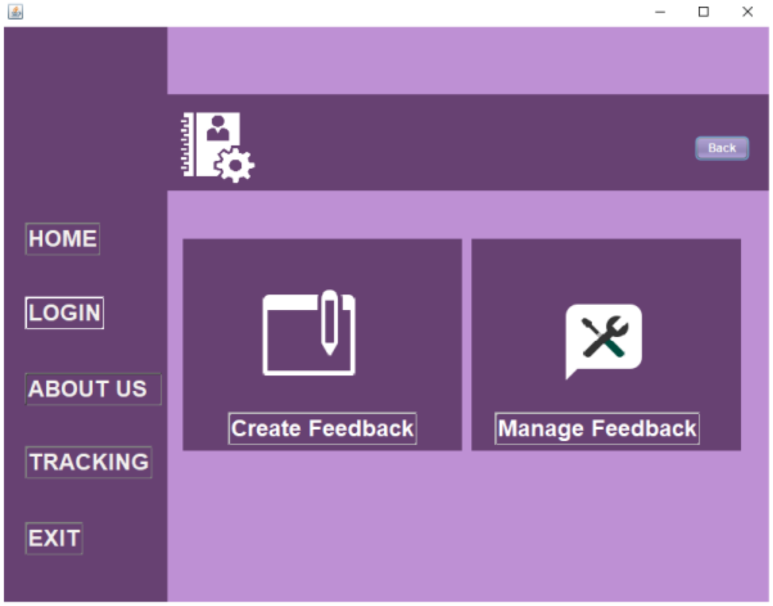


Fig: Feedback Menu

This is a GUI for feedback management where a managing staff can select whether to create feedback or to manage the created feedback.

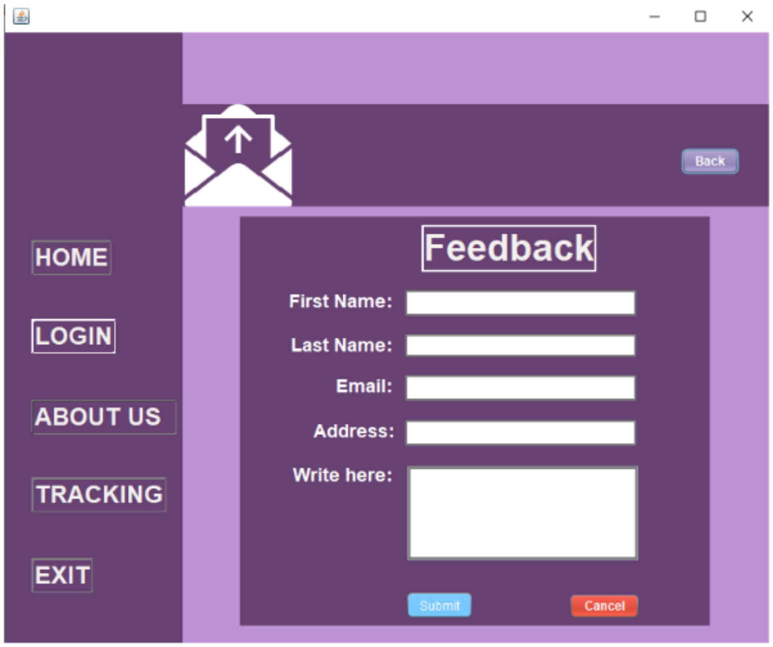
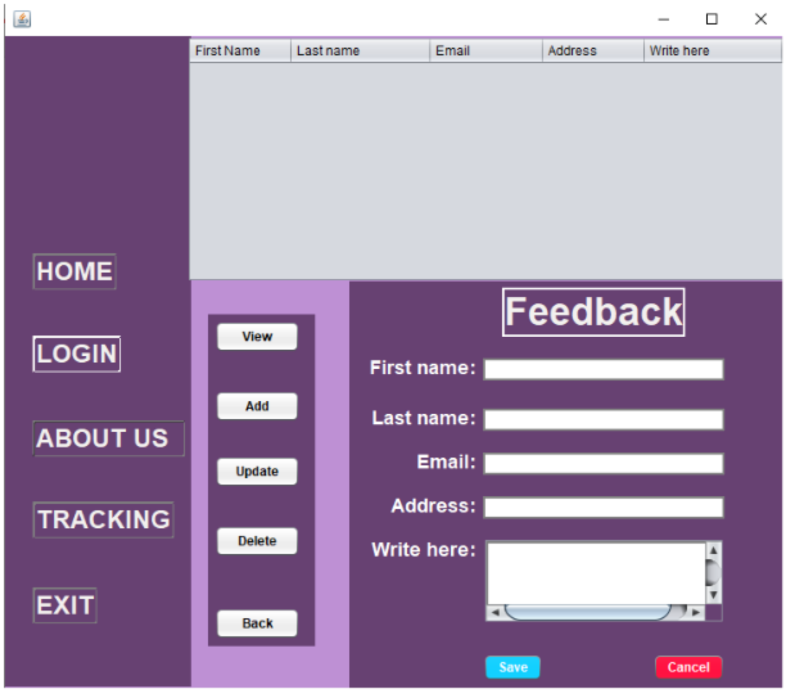


Fig: Create feedback

In this GUI a managing staff can create feedback as per the customer’s response.



First Name	Last name	Email	Address	Write here

Fig: Feedback management

In this GUI managing staff can add, update and delete the feedback.

k. Report Menu

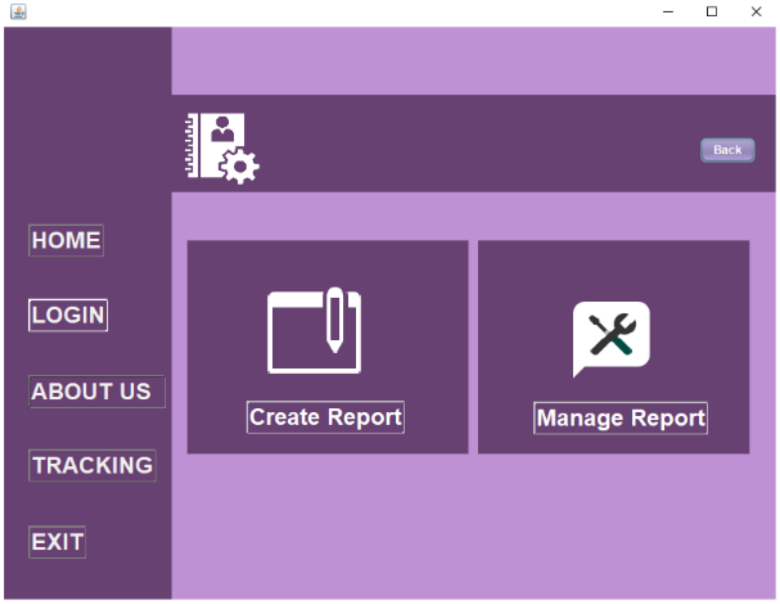
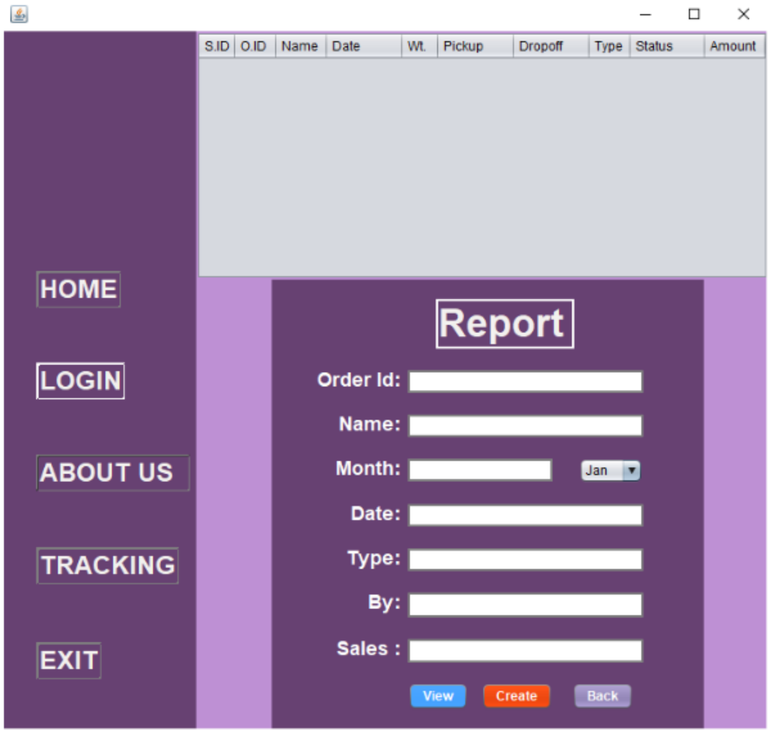


Fig: Report Menu

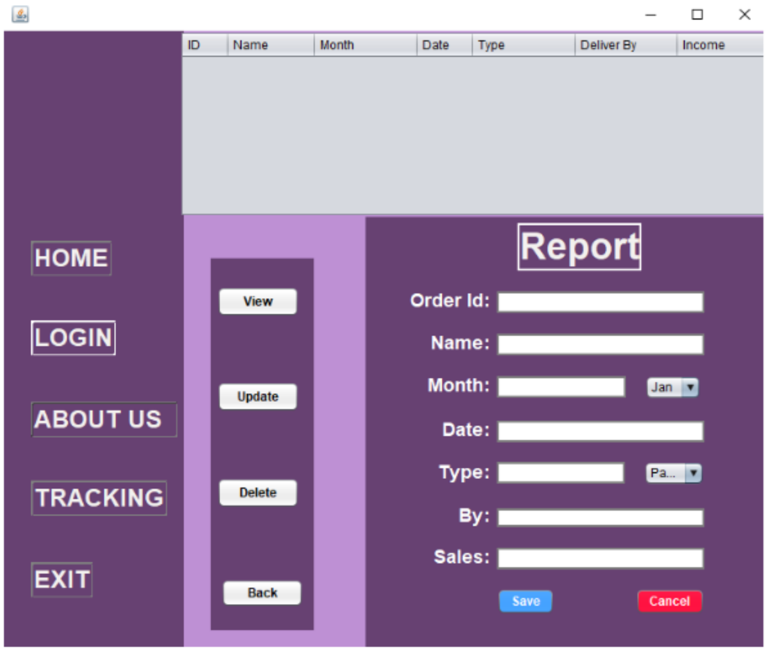
In this GUI, a managing staff can create and manage the report.



S.ID	O.ID	Name	Date	Wt.	Pickup	Dropoff	Type	Status	Amount

Fig: Create Report using delivered details

Similarly, a managing staff can create a report from the details of delivered courier which display in the table.



ID	Name	Month	Date	Type	Deliver By	Income

Fig: Report Management

In this GUI, a managing staff can manage the report details like update and delete.

l. Delivery Staff Menu

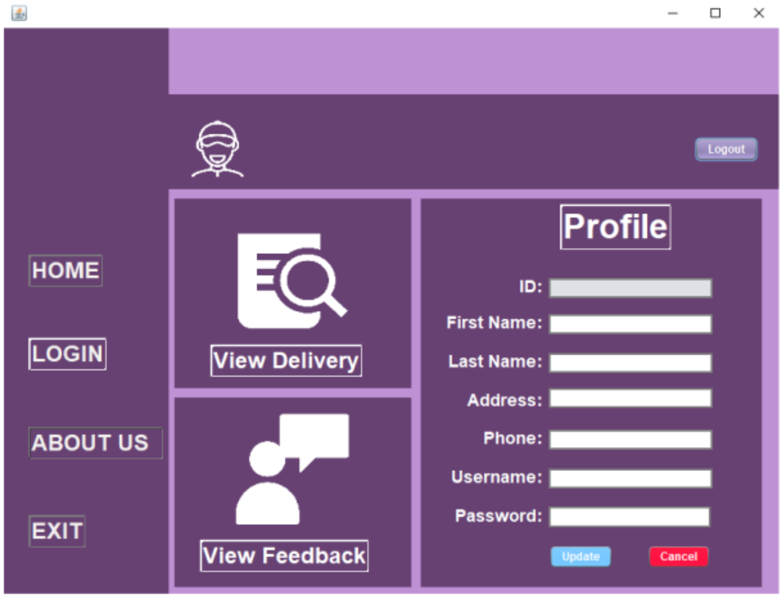
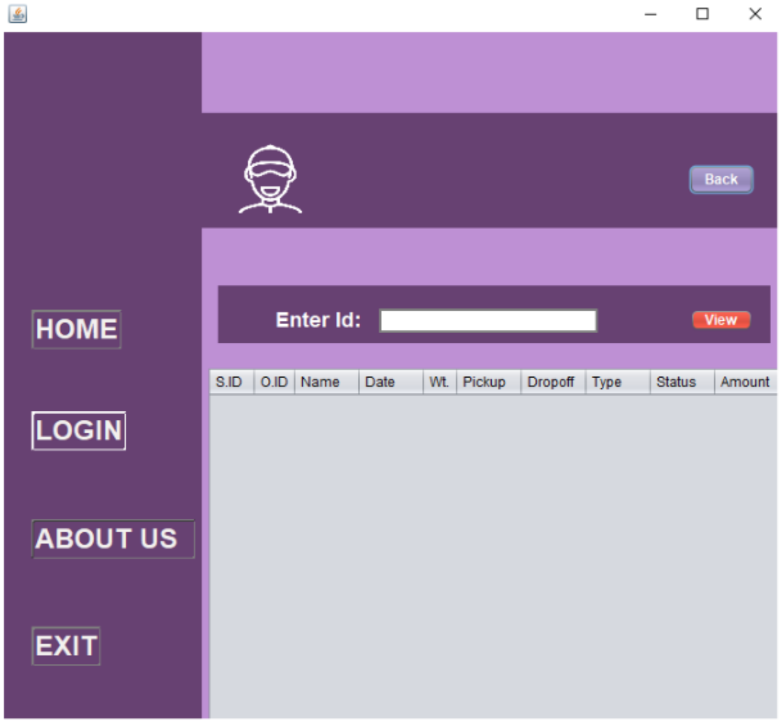


Fig: Delivery staff menu

In this GUI a delivery staff can update his/her profile and select the available functionalities like view delivery and view feedback.



S.ID	O.ID	Name	Date	Wt.	Pickup	Dropoff	Type	Status	Amount

Fig: Assigned delivery details

In this GUI, a delivery staff can view the delivery details by entering his/her staff id.

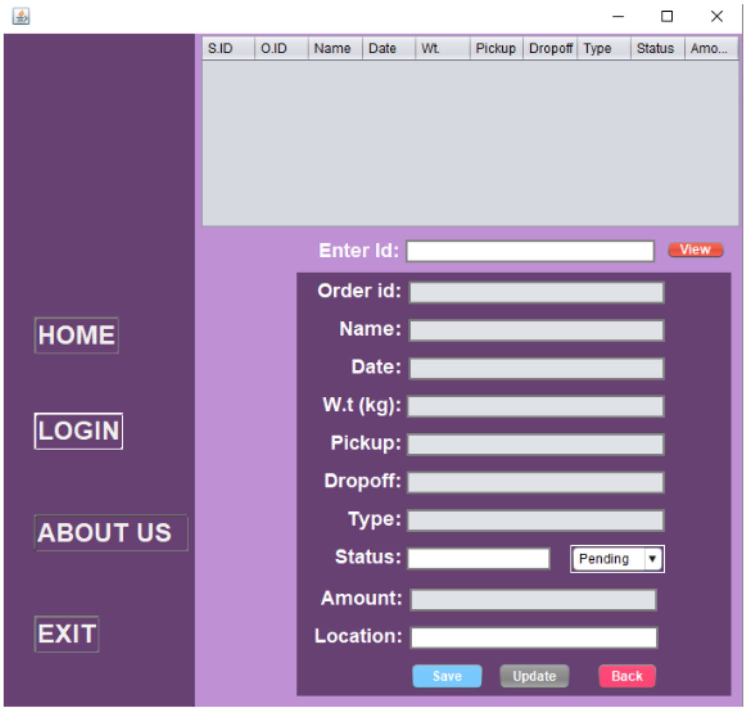
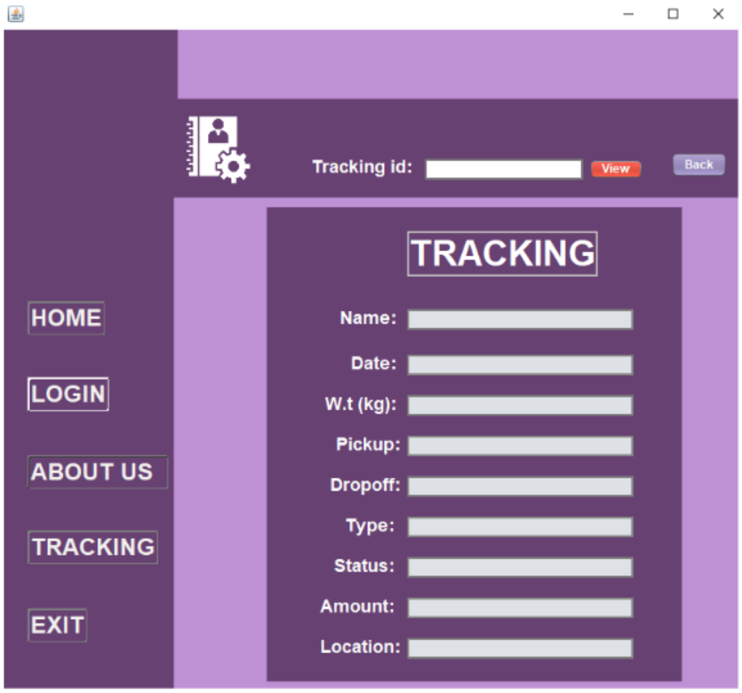


Fig: Update Location for tracking

In this GUI a delivery staff can update location by which managing staff can track the order using order id.



Tracking id:

TRACKING

Name:

Date:

W.t (kg):

Pickup:

Dropoff:

Type:

Status:

Amount:

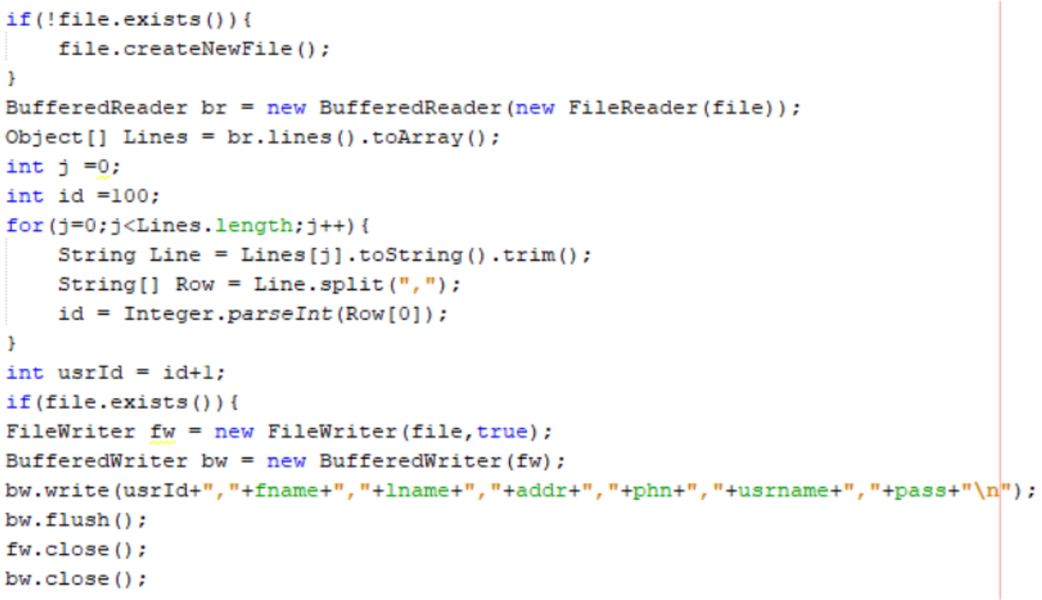
Location:

Fig: Tracking location

Similarly, this functionality is only for managing staff where managing staff can track the order using order id.

7. Additional Features

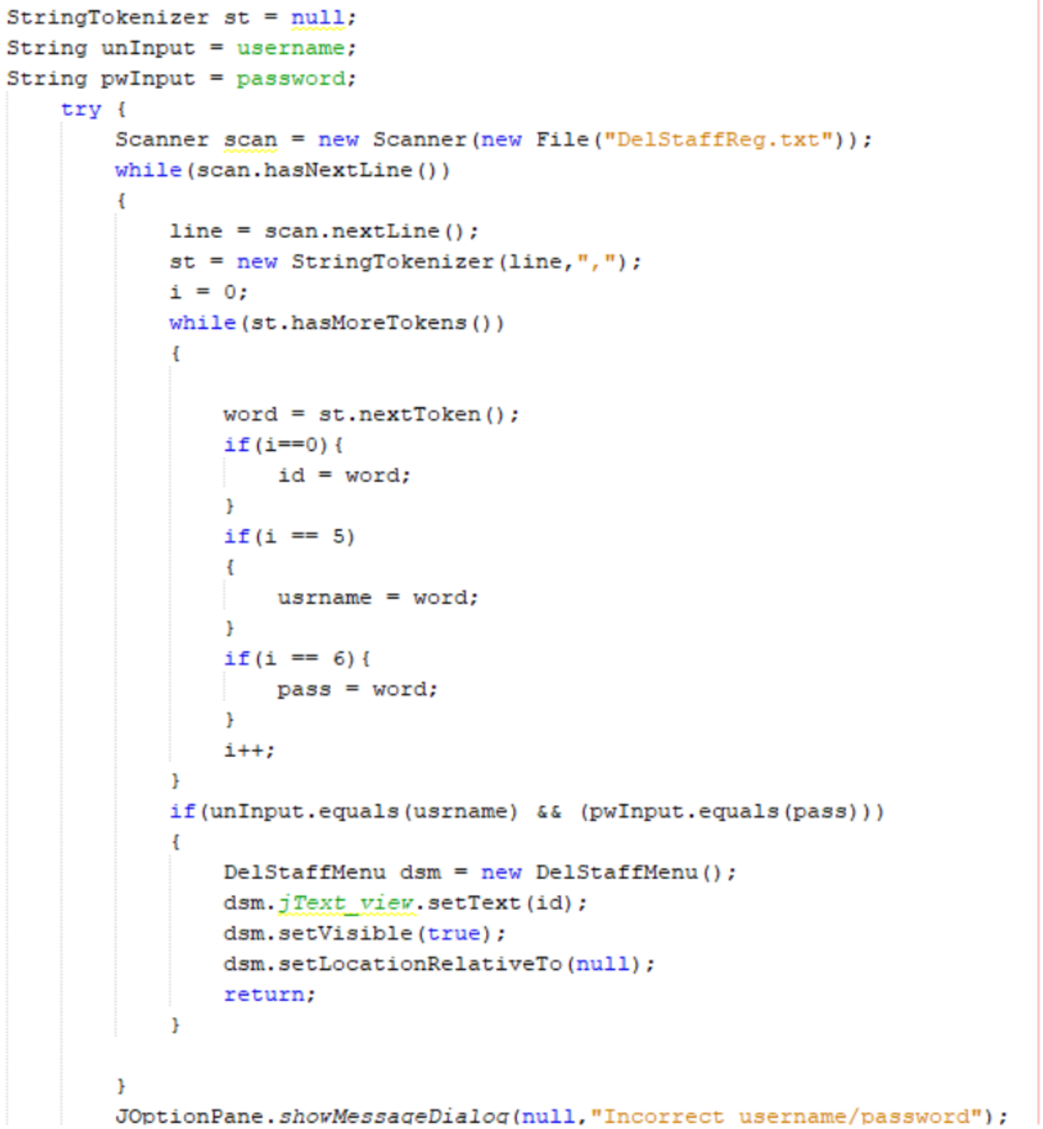
1. Writing in text file



```
if(!file.exists()){
    file.createNewFile();
}
BufferedReader br = new BufferedReader(new FileReader(file));
Object[] Lines = br.lines().toArray();
int j =0;
int id =100;
for(j=0;j<Lines.length;j++){
    String Line = Lines[j].toString().trim();
    String[] Row = Line.split(",");
    id = Integer.parseInt(Row[0]);
}
int usrId = id+1;
if(file.exists()){
FileWriter fw = new FileWriter(file,true);
BufferedWriter bw = new BufferedWriter(fw);
bw.write(usrId+","+fname+","+lname+","+addr+","+phn+","+usrname+","+pass+"\n");
bw.flush();
fw.close();
bw.close();
```

By creating object for buffered writer bw.write method have been use to write the data into file which is used as database. There are 7 data fields as shown above where getText() have been used to get the entered value from the GUI and buffered writer will write the data into the text file.

2. Reading from text file



```
StringTokenizer st = null;
String unInput = username;
String pwInput = password;
    try {
        Scanner scan = new Scanner(new File("DelStaffReg.txt"));
        while(scan.hasNextLine())
        {
            line = scan.nextLine();
            st = new StringTokenizer(line,",");
            i = 0;
            while(st.hasMoreTokens())
            {

                word = st.nextToken();
                if(i==0){
                    id = word;
                }
                if(i == 5)
                {
                    usrname = word;
                }
                if(i == 6){
                    pass = word;
                }
                i++;
            }
            if(unInput.equals(usrname) && (pwInput.equals(pass)))
            {
                DelStaffMenu dsm = new DelStaffMenu();
                dsm.jText_view.setText(id);
                dsm.setVisible(true);
                dsm.setLocationRelativeTo(null);
                return;
            }

        }
        JOptionPane.showMessageDialog(null,"Incorrect username/password");
```

Fig: Reading with text file

For reading purpose we have declared a file path and used scanner which is used to read a file and input which a user has entered. Similarly, Scanner will capture the entered data and search one by one in the declared file path. If the entered data is found in the file, then it will save in tokens by string tokenizer.

8. Conclusion

This Courier Management System requires a lot of functionality, which makes it difficult to build in Java using object-oriented concepts. To provide an overview of the system, use case diagrams, class diagrams, and activity diagrams are utilized. From this project, we have a better understanding of how Java works. How to develop a Java GUI has been taught.

This project focuses on object-oriented programming, which is necessary to utilize or apply OOP principles in the system. Encapsulation and inheritance are two OOP principles that have been used in this system. We are having some issues with the system's development procedures, particularly in the order section. We learned more about problem solving and object-oriented development as a result of this assignment.

OOP

ORIGINALITY REPORT

3%	2%	0%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	2%
2	mafiadoc.com Internet Source	1%
3	www.scirp.org Internet Source	1%
4	www.arbitration.ie Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		