Object Oriented Development Using JAVA

Contents

Abstract

The ‘Courier Service System’ is a platform with which an individual can convey his/her goods from one geographical location to other. This system acts as a bridge between sender and receiver by safely and convincingly delivering the packages. Furthermore, it also provides facilities to book a shipment, manage employee information as well as to keep hub and system details up to date. The entire system is built, based on JAVA programming language, reinforced with **Object Oriented Programming (OOPs)** concepts. With prime objective of making system effectual and flexible, the OOPs concept is justified and implemented throughout the system. A number of OOPs concepts such as objects, classes inheritance, polymorphism and abstraction of data are brought into execution during the system construction. The system facilitate users with handful of functionalities such as creating, reporting, shipping and managing orders. With the intention of allowing system security to end users, the system requires obligatory authorization from every end users. Likewise, data and other requisite details about users and transactions are accumulated in a text files (.txt).

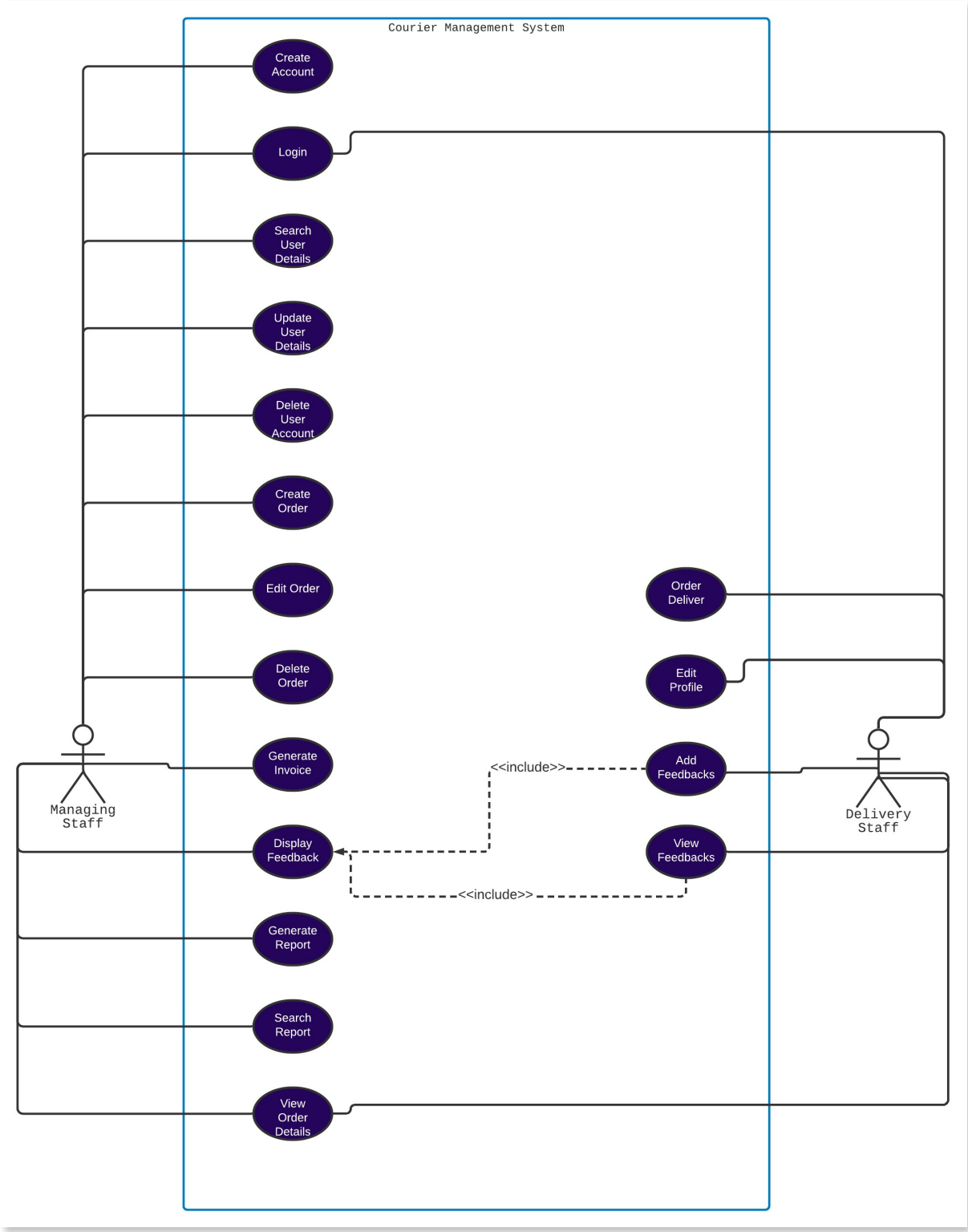
Acknowledgement

The encouragement and positive motivation we were given throughout the completion of this project is gigantic. First of all, we manifest our heartfelt appreciation to our subject teacher **Mr. Sushil Adhikari** sir, who guided us to the triumph of project from the very first day. Without his assistance, we might not be able to complete the project with detailed understanding and lucidity.

Likewise, we also convey thankfulness to all our colleagues who helped us during project. Your visualization and perception about the project aided us to learn new techniques and to implement them in the task. Winding up, we would like to make use of this opportunity to express our deepest gratefulness to our parents and the college who are facilitating us to learn professorial technical courses.

1. Requirement Analysis

1.1 Use Case Diagram



Use Case	Actors	Functions
Create User Account	Managing Staff	Creates a new account
Login	Managing Staff	Authorizes users to run the system
Search Details	Managing Staff	Finds customer information
Update Details	Managing Staff	Updates credentials
Delete Account	Managing Staff	Removes existing account
Create Order	Managing Staff	Creates an order through the system
Edit Order	Managing Staff	Changes order information
Delete Order	Managing Staff	Deletes or cancels order
Generate Invoice	Managing Staff	Creates bill according to the order
Display Feedback	Managing Staff	View feedbacks provided by customers
Generate Report	Managing Staff	Generates a report of provided services
Search Report	Managing Staff	Studies transactions and services done within certain period of time
Create Delivery Staff Account	Managing Staff	Opens a new account for delivery staff
Search Delivery Staff	Managing Staff	Displays information about delivery staffs / drivers
Delete Delivery Staff	Managing Staff	To remove or fire delivery staff
View Order Details	Managing Staff	To display details of a particular order
Edit Profile	Delivery Staff	To make certain changes in individual profile of delivery staff
Add Feedbacks	Delivery Staff	To add customer's feedbacks about the provided service
View Feedbacks	Delivery Staff	To check list of feedbacks received

1.2 Class Diagram

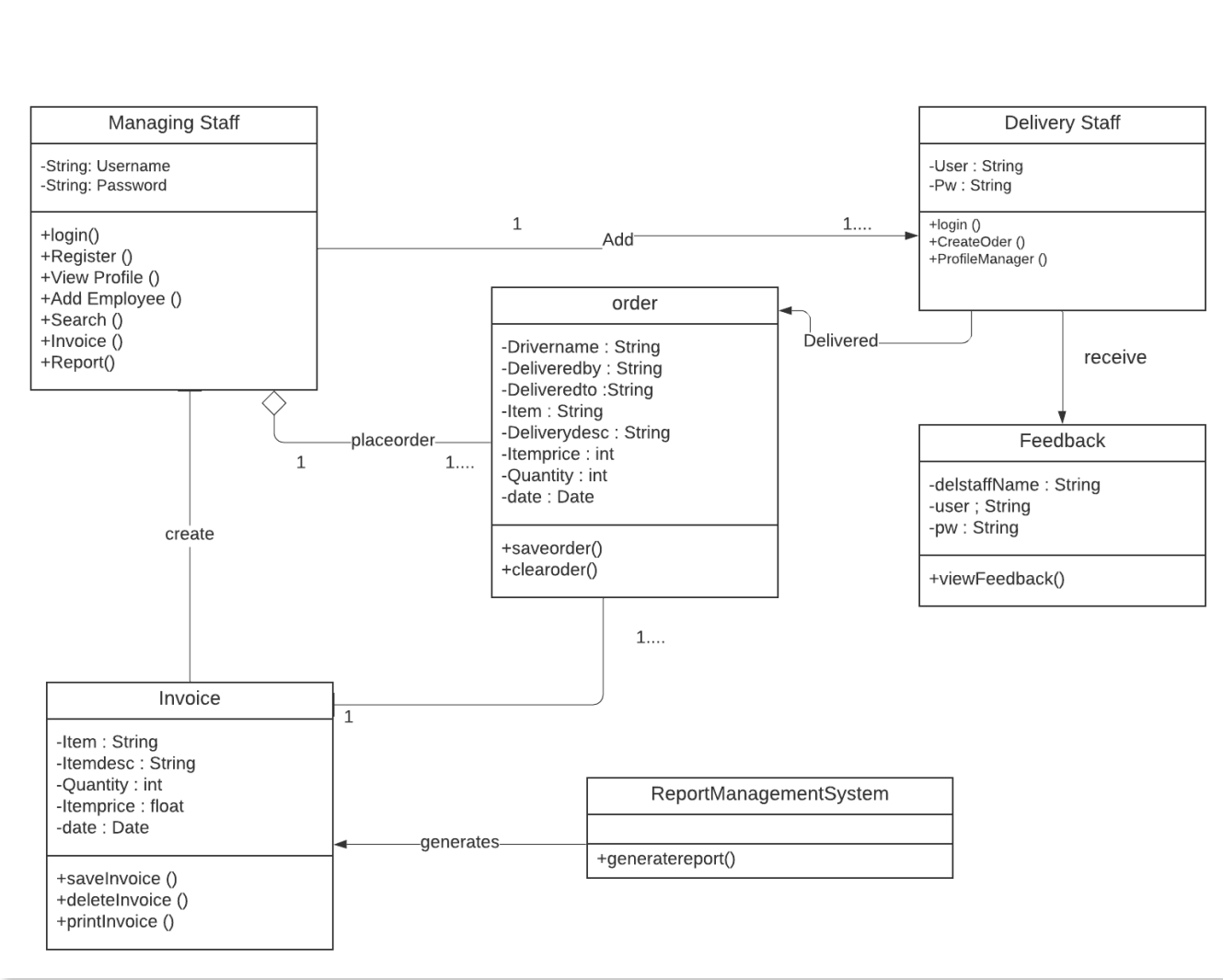


Figure : Class Diagram of Courier Service System

2. User Roles

2.1 Managing Staff

The Managing Staff, also recognized as ‘System Admin’ is supreme end user of the system. The function and features of the system are developed in such a way that they are accessible to management staff to make necessary modifications. Some of the key features of managing staff is, to be in charge of user accounts, superintend existing orders and also to control records and customer feedbacks. He/She can make alterations within user credentials such as insertion, updating deletion, etc. whenever it is necessary. Furthermore, a managing staff will also assign a delivery staff to receive an order from a specific location. As per project requirements, the system admin requires authorization, every time he attempts to access the system.

2.2 Delivery Staff

Delivery staffs are end users whose task is to drop-ship goods from a specific arrival station to its final destination. In courier system, delivery staffs are under the jurisdiction of admin staff, who assigns a delivery person for order delivery. A delivery staff requires his credentials to login and access system functionalities. Once the access is gained, he can view orders assigned to him, approve and update the status once delivery finalized and also manage his individual profile to update his credentials.

3. Object Oriented Programming Concept (OOPs)

Object-oriented programming is a language, that functions on the basis of fundamental idea of objects (V. MEHTRE and NIGAM, 2019). Java is considered as object-oriented programming language. This means, everything in Java is well ordered as synergy of objects, with the aim of assimilating actions as well as information. Using OOPS concepts in Java, one can comfortably build variety of software programs (Simmel, 1992).

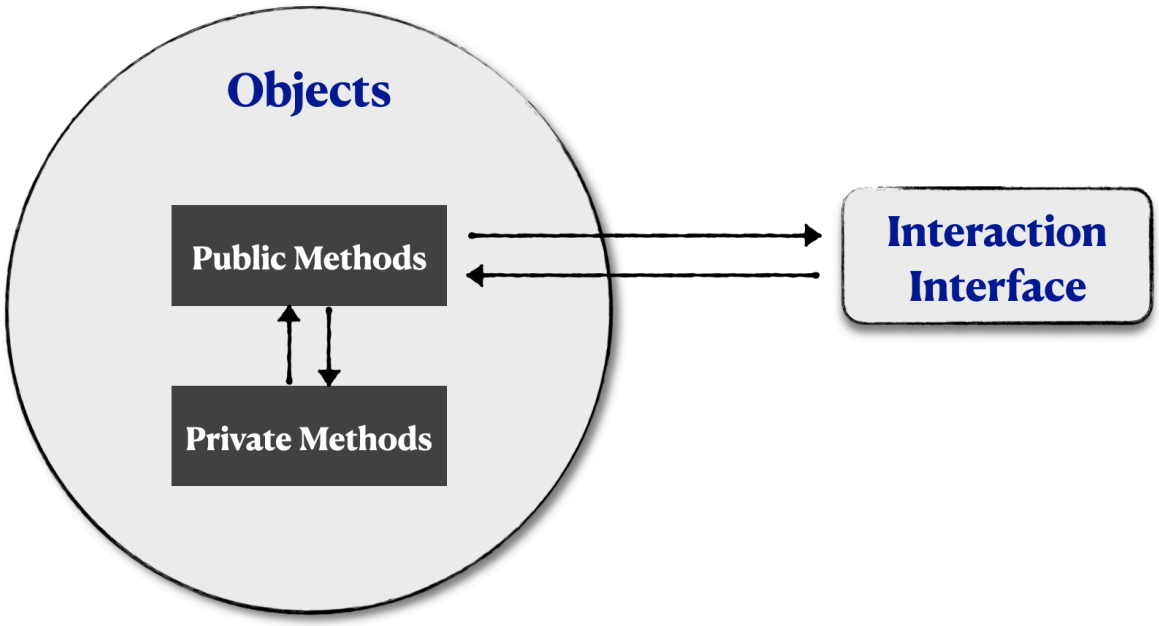


Figure : Picture showing Object interaction

3.1 Advantages of OOPs

Following are some of the advantages of Java OOPs concept :

- Object-oriented programming is modular and extensible. This gives advanced efficiency in software development process.

- The natural and practical technique in OOP makes problem solving practice effectual.

- Reusability of objects in OOP authorizes high quality lower cost and swift development.

- OOP concept makes system enhancement effortless with bottom-up approach and also minimize software complexity with inheritance.

3.2 Disadvantages of OOPS

Following are some of the disadvantages of Java OOPs methodology :

- The object oriented methodology of constructing programs with objects interaction might be complicated for some people.

- Since a lot of commands need to be performed, OOP methodology can also be sluggish in comparison to procedure-based programming.

3.3 Implemented Concepts : Demonstration & Justification

3.3.1 Objects and Classes

An object is real-world entity that has particular state, behavior and identity. Objects can be both physical or logical. Anything around us having certain state and behaviors can be considered as an object (Javatpoint, n.d.). For example, a soccer ball is an object. The name of the ball is Nike and color is blue (recognized as its state). Likewise, the ball is used for playing, which is its behavior.

In OOP paradigm, object is an instance of a class. This means, class is a design that allows us to create as many objects as we like. A Java class is a logical logical entity that comprise of constituents like methods, constructors, fields and blocks.

Example of Objects and Class Concept

```
public void assignData()
{
    try
        {
        BufferedReader br = new BufferedReader(new FileReader("/Users/sandeshkey/Desktop/Courier Service System/Files/Accounts.txt"));
        String s;
        b:
        while((s=br.readLine()) != null)
        {
            if(s.equals(userID))
            {
                txtID.setText(s);
                txtFirstName.setText(br.readLine());
                txtLastName.setText(br.readLine());
                txtMail.setText(br.readLine());
                txtPass.setText(br.readLine());
                txtCur.setText(br.readLine());
                txtPer.setText(br.readLine());
                break b;
            }
        }
        }catch(Exception ex)
        {
            ex.printStackTrace();
        }
}
```

Figure : Screenshot demonstrating usage of Objects and Class concept

3.3.3 Constructors

Constructor in java are codes having block which is similar to the method that is defined whenever class instance is being created.

```
public class DeliveryStaff {
    private String username;
    private String password;

    @Override
    public String toString() {
        return "DeliveryStaff{" + "username=" + username + ", password=" + password + '}';
    }

    public DeliveryStaff(String user, String pw) {
        this.username = user;
        this.password = pw;
    }

    public void setUsername(String user) { this.username = user; }

    public void setPassword(String pw) { this.password = pw; }


    public String getUsername() { return username; }

    public String getPassword() { return password; }
```

Figure : Screenshot demonstrating usage of Constructors

During the time of defining a constructor, object needs memory which is assigned in the memory. Constructor is mainly used to initialize object in Java. New () keyword is used while creating an object and in that time, default constructor is defined (How to Use a Constructor in Java, 2016). There is no necessity of writing a class constructor as java creates default constructor for users without mentioning class constructor.

3.3.2 Inheritance

A conception about the properties obtained from one class to field of another classes is known as Inheritance in java. Hierarchical order is seen with the use of inheritance while managing information. In case of java, methods and attributes of class can inherit another class (Vitek, 2008). Child class or derived class or sub – class is class which inherits the properties of another class. Likewise, parent class or superclass is the class that inherited from properties of another class. In order to create the child class, 'extends' keyword is used in Java.

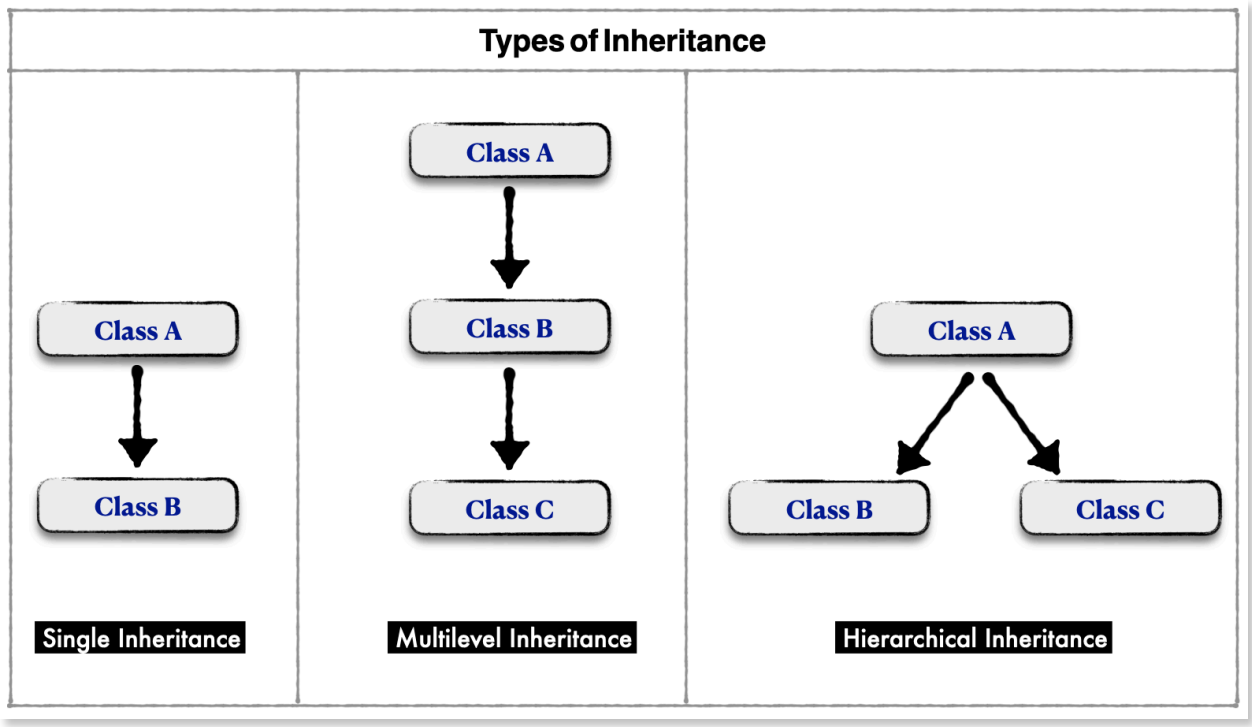


Figure : Types of Inheritance

```java
public class UserProfile extends javax.swing.JFrame {

    String userID;
    public UserProfile() {
        initComponents();
    }

    public UserProfile(String id)
    {
        userID = id;
        initComponents();

        assignData();
    }
```

Figure : Screenshot demonstrating usage of Inheritance concept

3.3.3 Encapsulation

Among four fundamental concepts of object oriented programming, encapsulation is one of them. Encapsulation in java is a technique where both data and code are wrapped up under a single unit. In other word, it is the prevention shield of data that prevents data to be accessed by the code (GeeksforGeeks, 2021). *Figure* : Screenshot demonstrating usage of getters and setters

```java
public class DeliveryStaff {
    private String username;
    private String password;

    @Override
    public String toString() {
        return "DeliveryStaff{" + "username=" + username + ", password=" + password + '}';
    }

    public DeliveryStaff(String user, String pw) {
        this.username = user;
        this.password = pw;
    }

    public void setUsername(String user) { this.username = user; }

    public void setPassword(String pw) { this.password = pw; }


    public String getUsername() { return username; }

    public String getPassword() { return password; }
```

Figure : Screenshot demonstrating usage of getters and setters

Variable class will be secured from other class in encapsulation, method of current class only permits to access class of variables. This is the reason that encapsulation also known for data hiding.

Following points should be considered in order to achieve encapsulation in java:

- Class variables should declared as private.
- In order to do any modifications, public setter and getter should be provided and variables values are demonstrated.

3.3.4 Abstraction

Abstraction in java is the mechanism of demonstrating and hiding fundamental details and information from users. Basically, it is used to display necessary information and hide private details. Moreover, it is also a technique to deal with ideas rather than functions. Abstract classes and interface are the two way of achieving abstraction in java (Schoeberl et al., 2011). A class uses 'abstract' keyword for its declaration, which is known as abstract class. With infract, our project achieve 100% abstraction that will deny users from viewing inner working mechanism of the system.

- In abstract classes, there is restriction to create an object.
- Abstract method does not contain any body in which body are provided as subclasses and are only allowed in abstract class

3.3.5 Polymorphism

An object can perform similar action in a number of different ways which is allowed by polymorphism in java. An object in java passes through IS–A test that is considered as polymorphism in java. Compile–time polymorphism and runtime polymorphism are the two types of polymorphism in java. Polymorphism in OOPs permits one to perform one task in multiple ways. Likewise, in technical terms, polymorphism in java permits defining a single interface and to implement a single task to do multiple tasks.

```java
public class DeliveryStaff {
    private String username;
    private String password;

    @Override
    public String toString() {
        return "DeliveryStaff{" + "username=" + username + ", password=" + password + '}';
    }

    public DeliveryStaff(String user, String pw) {
        this.username = user;
        this.password = pw;
    }

    public void setUsername(String user) { this.username = user; }

    public void setPassword(String pw) { this.password = pw; }


    public String getUsername() { return username; }

    public String getPassword() { return password; }
```

Figure : Screenshot demonstrating usage of Polymorphism

4. Project Design

4.1 Main Page

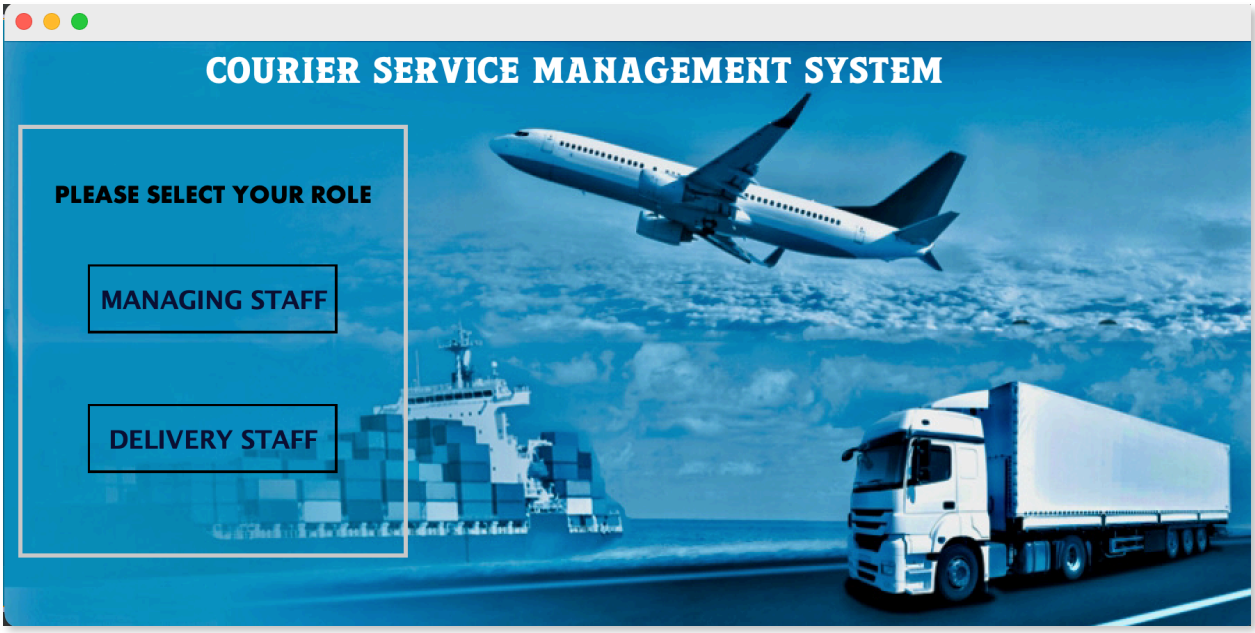


Figure : Main Page of Courier Service System

4.2 Login Page

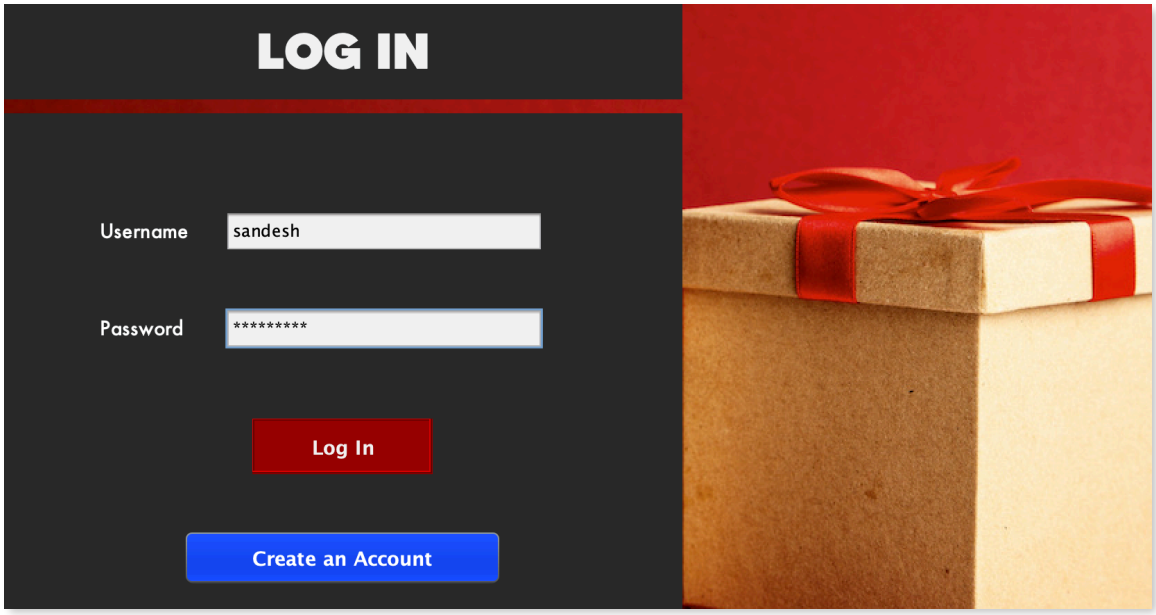


Figure : Login Page of Courier Service System

4.3 User Profile Management

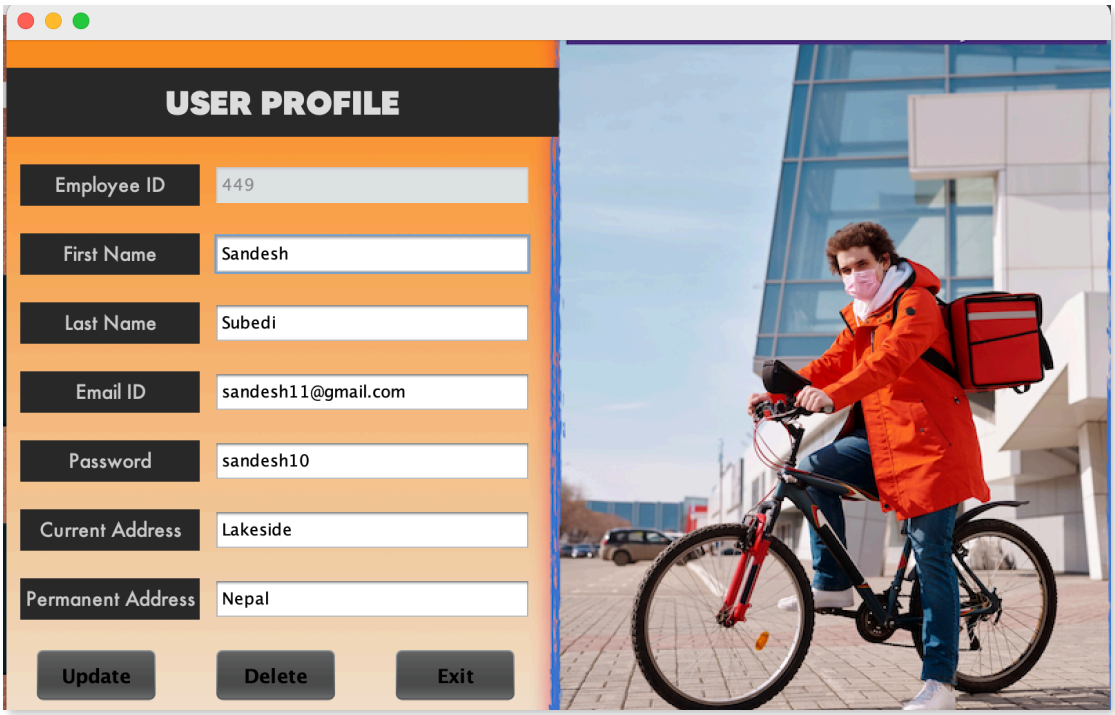


Figure : User Profile Management

4.4 Employee Management

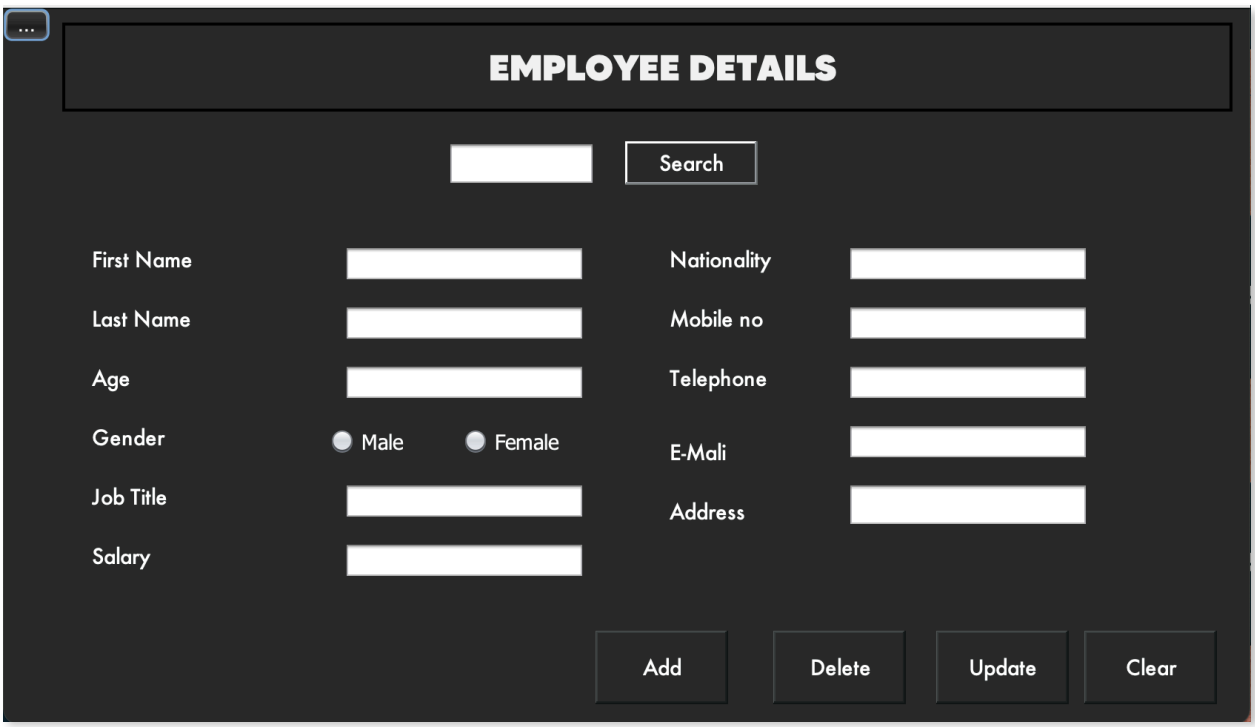


Figure : Employee Management

4.5 Invoice

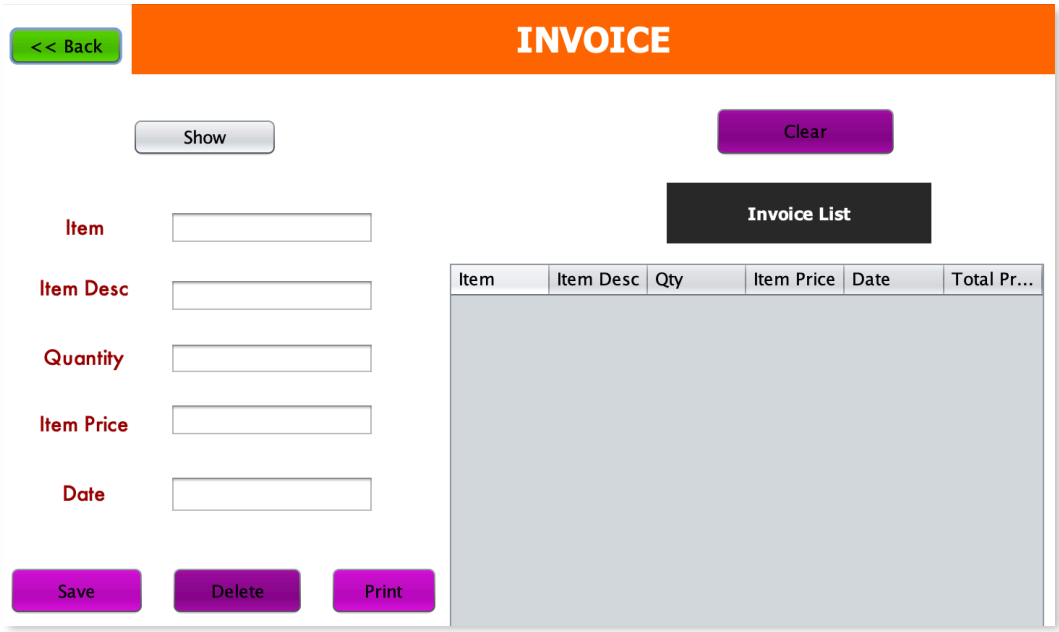


Figure : Employee Management

4.6 Order Management

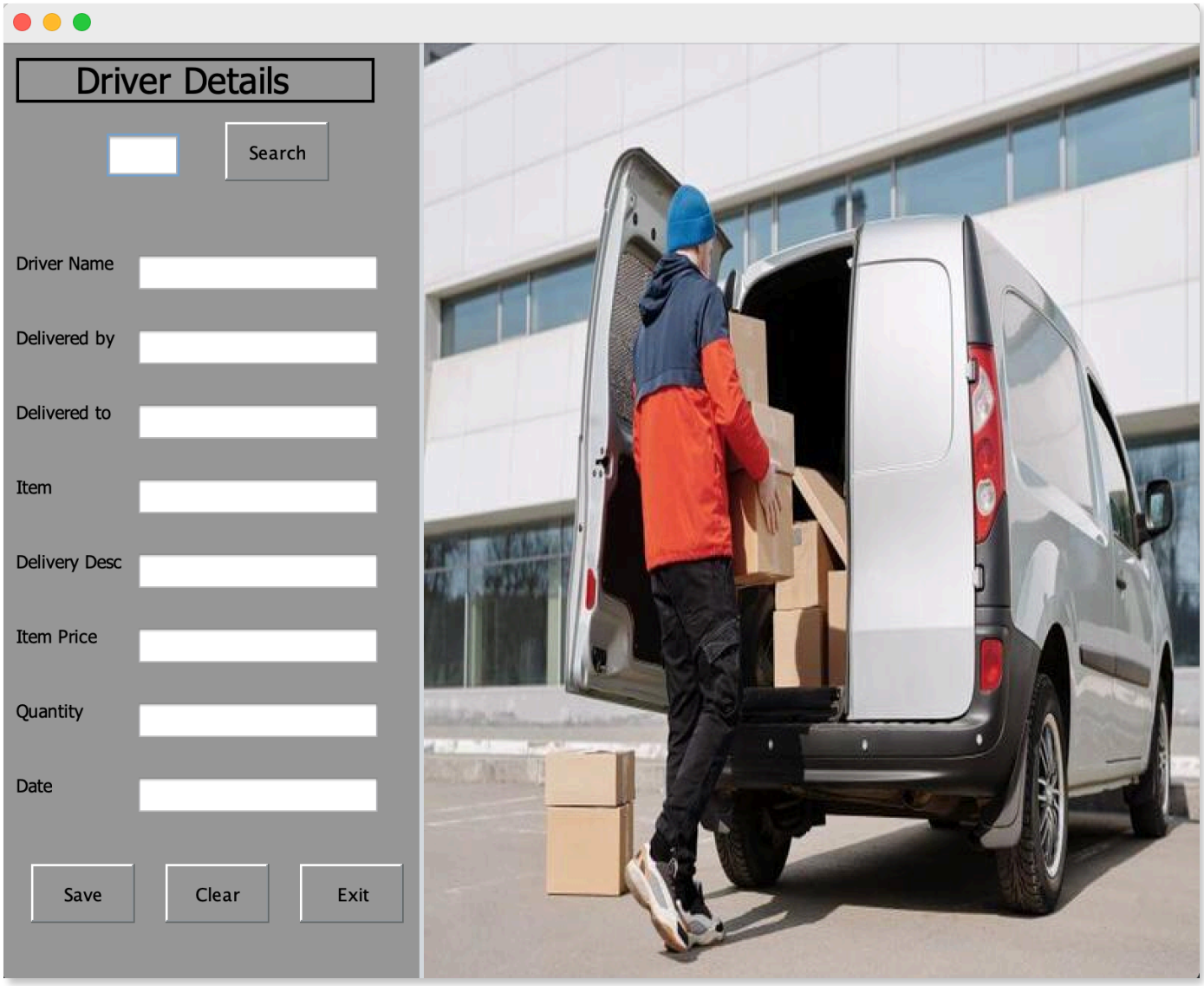


Figure : Order Management

4.7 Report Details

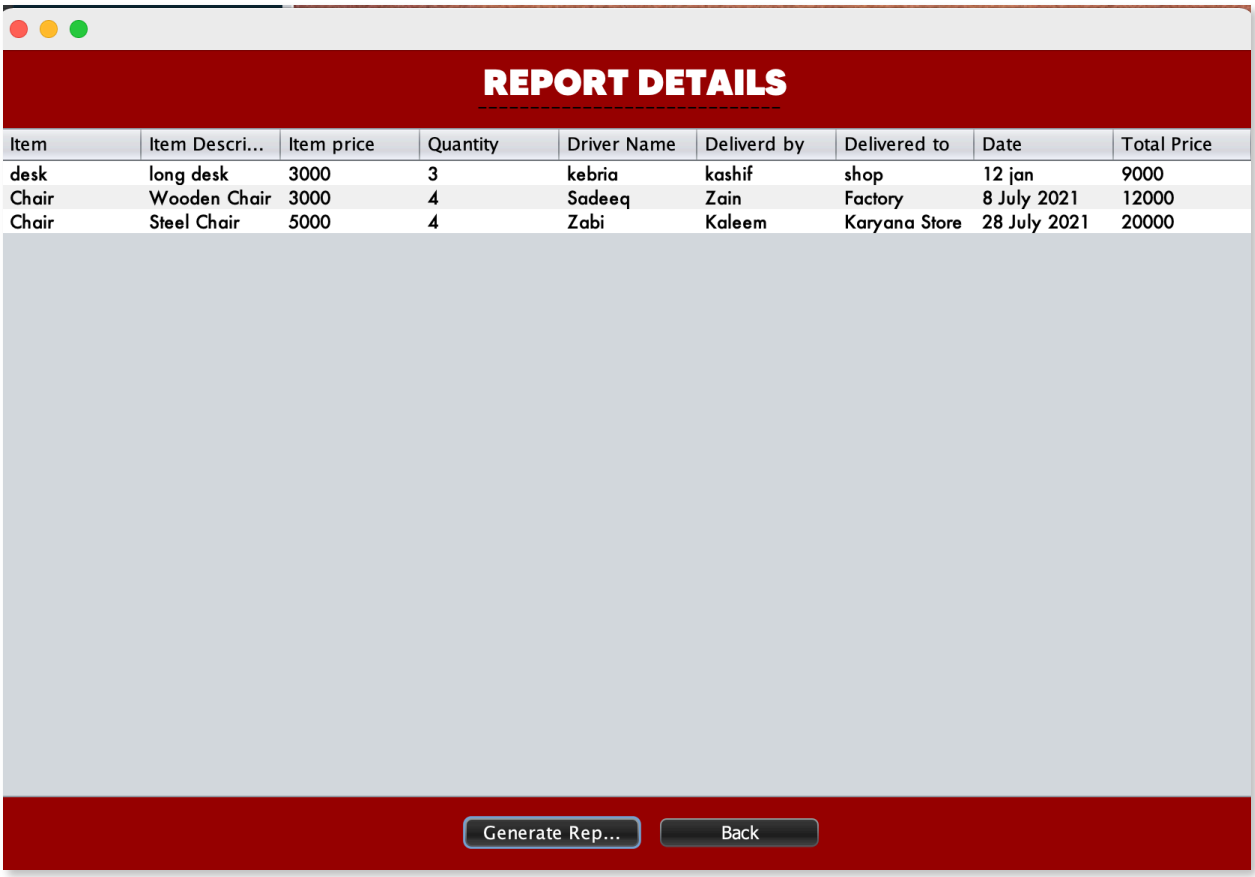


Figure : Report Details and Management

6. Conclusion

Java provides users a wonderful platform to demonstrate programming skills and construct something extraordinary. This particular project was to develop a Courier Service System using object-oriented programming concepts and the guidelines are effectively followed. The system comprise of two major departments (managing and delivery) including features like registration, modification, deletion, along with order management and other courier features. The designed system can help courier companies ship or deliver items from one destination to another. This overall system depicts the importance of technology and use of it in contemporary world.

7. References

GeeksforGeeks. (2021, August 2). *Encapsulation in Java*. https://www.geeksforgeeks.org/encapsulation-in-java/

How to Use a Constructor in Java. (2016, March 26). Dummies. https://www.dummies.com/programming/java/how-to-use-a-constructor-in-+java/#:%7E:text=A%20constructor%20in%20Java%20is,of%20an%20object%20is%20created.&text=A%+20constructor%20doesn't%20have,considered%20members%20of%20a%20class.

Javatpoint. (n.d.). *Object in Java | Class in Java - javatpoint*. Www.Javatpoint.Com. Retrieved July 9, 2021, from https://www.javatpoint.com/object-and-class-in-java

Schoeberl, M., Korsholm, S., Kalibera, T., & Ravn, A. P. (2011). A Hardware Abstraction Layer in Java. *ACM Transactions on Embedded Computing Systems*, *10*(4), 1–40. https://doi.org/10.1145/2043662.2043666

Simmel, S. (1992). Object-based visual programming languages. *OOPS*, *3*(4), 99. https://doi.org/10.1145/143776.270578

Sinicki, A. (2017, March 13). *What is Object Oriented Programming?* Android Authority. https://www.androidauthority.com/object-oriented-programming-755216/

V. MEHTRE, V. I. S. H. A. L., & NIGAM, Y. A. S. H. (2019b). Review on Concepts Related to Object Oriented Programming System. ICONIC RESEARCH AND ENGINEERING JOURNALS, 3(6), 56–58. https://www.irejournals.com/formatedpaper/1701801.pdf

Vitek, J. (2008). *ECOOP 2008 - Object-Oriented Programming*. Springer Publishing. https://doi.org/10.1007/978-3-540-70592-5_28