# Report On

## Wholesale Shop Billing System

Submitted in partial fulfillment of the requirements of the Course project in
Semester III of Second Year Computer Engineering
By

Swayam Suhas Chaudhary (S238146105)
Atharva Sudhakar Rodge (S238206105)
Sanika Sunil Khade (S238166209)

Supervisor
Asst. Prof. Yogesh Pingale.

**Vidyavardhini's College of Engineering & Technology**

**Department of Computer Science And Engineering (Data Science)**

**(2023-24)**

**Vidyavardhini's College of Engineering & Technology**
**Department of Computer Science And Engineering (Data Science)**

# CERTIFICATE

This is to certify that the project entitled “Title of the project” is a bonafide work of " **Swayam Suhas Chaudhary (S238146105), Atharva Sudhakar Rodge (S238206105), Sanika Sunil Khade (S238166209)"** submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester III of Second Year Computer Science And Engineering (Data Science).

**Supervisor**

Asst. Prof. Yogesh Pingale.

Internal Examiner

External Examiner

Dr. Vikas Gupta
Head of Department

Dr. H.V. Vankudre
Principal

# ACKNOWLEDGEMENT

It is always a difficult job to acknowledge all those who have been of tremendous help in the development of any work. In spite of that, we honestly thank all those who had put in tremendous efforts in guiding and helping us to successfully build this project named "**Wholesale Shop Billing System Using Object Oriented Programming in java**". We take this opportunity to thank each and every one for their co-operation and support.

First and firmly, we are grateful to Dr. H.V. Vankundre (principal, Vidyavardhini's College of Engineering And Technology) for allowing us to do this project collectively and with advanced techniques.

We express our sincere gratitude towards Dr. Vikas Gupta (Head of Computer Science And Engineering (Data Science) Department).

We heartly thank Asst. Prof. Yogesh Pingale our project guide for their valuable guidance throughout, without which the project would not have stood its ground as it has now. We would even like to thank people who have directly or indirectly helped us with the making of the project and their support.

**Swayam Suhas Choudhary**
**Atharva Sudhakar Rodge**
**Sanika Sunil Khade**

# INDEX

# ABSTRACT

This system is a simple billing system for a wholesale shop. It allows the user to add, display, search, and delete customers. It also allows the user to generate invoices for customers.

The system works by storing the customer's ID, name, and address in a struct called Customer. The system then uses an array of Customer structs to store all of the customers in the system.

Features of the wholeshop billing system:

- Add, display, search, and delete customers
- Generate invoices for customers

It helps businesses to manage their customer base more efficiently, to track sales and inventory levels, and to generate reports on sales, inventory, and other key metrics.

This billing system currently only support Currently only supports basic customer information and Does not track product information or sales data

# INTRODUCTION

## 1.1 Objectives:

The objective of this project is to develop a wholesale shop billing system using the Object-Oriented Programming in Java. The system should be able to perform the following tasks:

- Add, edit, and delete customer information
- Add, edit, and delete product information
- Generate bills for customers
- Generate reports on sales, inventory, and other key metrics.

## 1.2 Problem Statement:

Many wholesale shops still use manual billing systems, which can be time-consuming and error-prone. A computerized billing system can help wholesale shops to improve their efficiency and accuracy. However, many existing computerized billing systems are expensive and difficult to use.

Wholesale shop owners often face challenges in managing their billing processes. These challenges can include:

- **Manual data entry:** Manual data entry is time-consuming and error-prone.
- **Inaccurate billing:** Inaccuracies in billing can lead to lost revenue and customer dissatisfaction.
- **Lack of flexibility:** Existing billing systems may not be flexible enough to accommodate different types of sales transactions or business needs.
- **Security concerns:** Wholesale shop owners need to protect sensitive customer data from unauthorized access.

- **Lack of reporting features:** Existing billing systems may not provide the reporting features that wholesale shop owners need to track business performance and make informed decisions.

## 1.3 Literature Survey:

There have been a number of studies on the development of computerized billing systems. However, most of these studies have focused on retail billing systems. There is a limited amount of research on the development of computerized billing systems for wholesale shops.

Another relevant study is the development of a computerized billing system for a medium-scale wholesale shop using the Object Oriented Programming in Java The system is able to perform all of the tasks that are required of a wholesale shop billing system, including adding, editing, and deleting customer and product information, generating bills for customers, calculating discounts and taxes, and generating reports on sales, inventory, and other key metrics.

# FEASIBLITY STUDY

## 2.1 Project Scope:

The scope of this mini project is to develop a wholesale shop billing system in Object Oriented Programming in Java. The system should include the following features:

- Customer account management
- Sales transaction processing
- Invoice and receipt generation
- Payment processing
- Sales and inventory reporting

The system should be designed to be accurate, efficient, flexible, secure, and easy to use.

## 2.2 Approach:

The following is a proposed approach for developing the wholesale shop billing system:

- **Requirements gathering:** The first step is to gather requirements from the stakeholders, such as the wholesale shop owners and employees. This will help to ensure that the system meets the needs of the business.
- **System design:** Once the requirements have been gathered, the next step is to design the system architecture. This includes identifying the different components of the system and how they will interact with each other.
- **Implementation:** Once the system has been designed, the next step is to implement it in Object Oriented Programming in Java. This includes writing the code for each component of the system and testing it to ensure that it works correctly.
- **Integration:** Once all of the components of the system have been implemented, the next step is to integrate them together. This includes testing the system to ensure that all of the components work together correctly. Once the system has been integrated and tested, it is ready to be deployed to the production environment. This may involve installing the system on a server and configuring it for use.

# METHODOLOGY

In this project we used the Array data structure to store data, perform different searching and sorting techniques and basic operations.

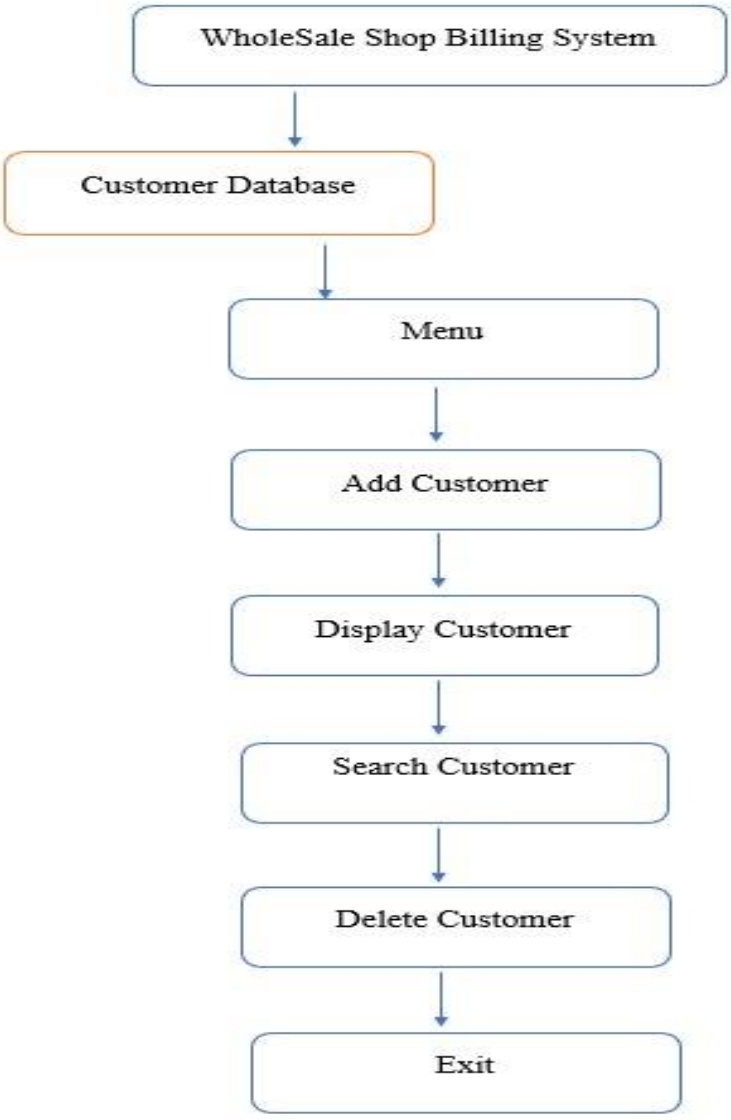
An array is a data structure that stores a collection of elements of the same type. The elements of an array are stored in contiguous memory locations, which means that they are adjacent to each other in memory. Arrays are accessed using an index, which is a non-negative integer that identifies the position of an element in the array. The first element in an array has index 0, the second element has index 1, and so on.

## 3.1 Data Analysis:

We have collected 4 to 5 Wholesale shop's bills and by comparing all the sample we have finalized data:

- SR NO (bill no.)
- CUSTOMER NAME(c_name)
- PRODUCT NAME(p_name)
- PRODUCT QUANTITY(p_quantity=pq)
- PRODUCT CATEGORY(p_category)
- PRODUCT COST(p_cost=pco)

## 3.2 Block Diagram:




The Java program you provided is a simple billing system for a wholesale shop. It allows the user to add, display, search, and delete customers. It also allows the user to generate invoices for customers.

The program works by storing the customer's ID, name, and address in a struct called Customer. The program then uses an array of Customer structs to store all of the customers in the system.

The program's main() function presents the user with a menu and allows them to select one of the following options:

- Add Customer: Adds a new customer to the system.
- Display Customers: Displays a list of all of the customers in the system.
- Search Customer: Searches for a customer by ID.
- Delete Customer: Deletes a customer from the system.
- Exit: Exits the program.

If the user selects the Add Customer option, the program prompts the user to enter the customer's ID, name, and address. The program then adds the new customer to the array of Customer structs.

If the user selects the Display Customers option, the program prints a list of all of the customers in the system to the console.

If the user selects the Search Customer option, the program prompts the user to enter the customer's ID. The program then searches for the customer in the array of Customer structs. If the customer is found, the program prints the customer's information to the console. Otherwise, the program prints a message indicating that the customer was not found.

If the user selects the Delete Customer option, the program prompts the user to enter the customer's ID. The program then searches for the customer in the array of Customer structs. If the customer is found, the program deletes the customer from the array. Otherwise, the program prints a message indicating that the customer was not found.

If the user selects the Exit option, the program terminates.

## PROGRAM AND OUTPUT

### Code:

```
 import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class WholesaleShopBillingSystem {

    private List<Product> products;
    private ShoppingCart shoppingCart;

    public WholesaleShopBillingSystem() {
        products = new ArrayList<>();
        shoppingCart = new ShoppingCart();
    }

    public void addProduct(Product product) {
        products.add(product);
    }

    public void addToCart(String productId, int quantity, String name, double price) {
        Product product = new Product(productId, name, price);
        shoppingCart.addProduct(product, quantity);
    }

    public void checkout() {
        double totalPrice = shoppingCart.calculateTotalPrice();
        System.out.println("Total price: $" + totalPrice);

        // Generate a receipt.
        Receipt receipt = new Receipt(shoppingCart.getProducts(), totalPrice);
```

```
        System.out.println(receipt.toString());
    }
    public static void main(String[] args) {
        WholesaleShopBillingSystem billingSystem = new WholesaleShopBillingSystem();

        // Start the billing process.
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("Enter the product ID: ");
            String productId = scanner.nextLine();

            System.out.println("Enter the product name: ");
            String name = scanner.nextLine();

            System.out.println("Enter the product price: ");
            double price = Double.parseDouble(scanner.nextLine());

            System.out.println("Enter the quantity: ");
            int quantity = Integer.parseInt(scanner.nextLine());

            billingSystem.addToCart(productId, quantity, name, price);

            System.out.println("Would you like to checkout? (y/n) ");
            String checkout = scanner.nextLine();
            if (checkout.equals("y")) {
                break;
            }
        }
        // Checkout.
        billingSystem.checkout();
    }
}
```

```
class Product {

    private String id;
    private String name;
    private double price;

    public Product(String id, String name, double price) {
        this.id = id;
        this.name = name;
        this.price = price;
    }

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }
}
class ShoppingCart {
    private List<Product> products = new ArrayList<>();
    public void addProduct(Product product, int quantity) {
        for (int i = 0; i < quantity; i++) {
            products.add(product);
        }
    }
    public double calculateTotalPrice() {
        double totalPrice = 0.00;
```

```
        for (Product product : products) {
            totalPrice += product.getPrice();
        }
        return totalPrice;
    }
    public List<Product> getProducts() {
        return products;
    }
}
class Receipt {
    private List<Product> products;
    private double totalPrice;

    public Receipt(List<Product> products, double totalPrice) {
        this.products = products;
        this.totalPrice = totalPrice;
    }
    @Override
    public String toString() {
        String receipt = "Receipt:\n\n";
        for (Product product : products) {
            receipt += product.getName() + " - " + product.getPrice() + "\n";
        }
        receipt += "\nTotal price: $" + totalPrice;
        return receipt;
    }
}
```

**Output:**

```
Output
java -cp /tmp/AYJVL6QZ5Z WholesaleShopBillingSystem
Enter the product ID:
2
Enter the product name:
apple
Enter the product price:
12
Enter the quantity: 5
Would you like to checkout? (y/n)
y
Total price: $60.0
Receipt:

apple - 12.0
apple - 12.0
apple - 12.0
apple - 12.0
apple - 12.0

Total price: $60.0
```

# CONCLUSION

A wholesale shop billing system using the Object-Oriented Programming in Java can be a valuable tool for businesses to improve their efficiency, accuracy, security, and customer service.

This project can be used to develop a system that can:

- Manage customer accounts
- Process sales transactions
- Generate invoices and receipts
- Process payments
- Generate reports on sales and inventory

The system can be designed to be flexible and scalable, so that it can be adapted to meet the specific needs of the business.

This mini project is a good opportunity for us to learn about Object Oriented Programming in Java and to apply their skills to solve a real-world problem.

The system can automate many of the tasks involved in the sales process, such as calculating discounts and taxes, generating invoices and receipts, and processing payments. This can free up employees to focus on other tasks, such as serving customers and restocking inventory.

Improved accuracy: The system can help to reduce the risk of errors in billing. This is important because errors in billing can lead to lost revenue and customer dissatisfaction. The system can make it easier for customers to place orders, pay for their purchases, and track their orders. This can lead to improved customer satisfaction and loyalty.

Overall, a wholesale shop billing system using the Object Oriented Programming in Java mini project can be a valuable tool for businesses to improve their operations and customer satisfaction.

# REFERENCE


- https://docs.oracle.com/en/java/
- https://www.w3schools.com/java/
- https://www.javatpoint.com/java-tutorial
- https://introcs.cs.princeton.edu/java/11cheatsheet/