# Food Ordering System

**A PROJECT REPORT**

***Submitted by***

**SAI NITHICK ROSHAAN S 2303811724321094**

***in partial fulfillment of requirements for the award of the course***

**CGB1201 – JAVA PROGRAMMING**

***in***

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

**SAMAYAPURAM – 621 112**

**DECEMBER, 2024**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)**

**SAMAYAPURAM – 621 112**

## BONAFIDE CERTIFICATE

Certified that this project report on **“FOOD ORDERING SYSTEM”** is the bonafide work of **SAI NITHICK ROSHAAN S 2303811724321094** who carried out the project work during the academic year 2024 - 2025 under my supervision.

[illegible]

**Signature**

Dr. T. AVUDAIAPPAN M.E.,Ph.D.,

**HEAD OF THE DEPARTMENT,**

Department of Artificial Intelligence,

K. Ramakrishnan College of Technology,

Samayapuram, Trichy -621 112.

[illegible]

**Signature**

Mrs. S. GEETHA M.E.,

**SUPERVISOR,**

Department of Artificial Intelligence,

K. Ramakrishnan College of Technology,

Samayapuram, Trichy -621 112.

Submitted for the viva-voce examination held on 3.12.24 .

[illegible]

**INTERNAL EXAMINER**

# DECLARATION

I declare that the project report on **"FOOD ORDERING SYSTEM"** is the result of original work done by me and best of my knowledge, similar work has not been submitted to "**ANNA UNIVERSITY CHENNAI**" for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING.**

[illegible]

**Signature**

**Place:** Samayapuram

**Date:** 3/12/2024

## ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution,**"K. Ramakrishnan College of Technology (Autonomous)",** for providing us with the opportunity to do this project.

I extend our sincere acknowledgement and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.,** for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.,** Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D**., Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E**., Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

## VISION OF THE INSTITUTION

**To serve the society by offering top-notch technical education on par with globalstandards.**

## MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies byexceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transformthem as all- round personalities respecting moral and ethical values.

## VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Scienceto fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conductive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

## PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

**PEO 1:** Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

**PEO 2:** Provide industry-specific solutions for the society with effective communication and ethics.

**PEO 3:** Hone their professional skills through research and lifelong learning initiatives.

## PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a memberor leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

# ABSTRACT

The Food Ordering System project is designed to streamline the process of placing food orders through a simple and interactive graphical user interface (GUI). Developed using Javaand AWT, the application offers an intuitive platform where users can select menu items, input quantities, and view an order summary with total cost calculations. The system is organized into key modules, including User Interface, Order Management, and Notification & Error Handling, ensuring a structured approach to data handling and user interaction. Exception handling and input validation mechanisms are incorporated to maintain the reliability and stability of the application. The final result is a user-friendly tool that enhances the efficiency of food ordering, reduces the chances of errors, and provides clear feedback to the user.

# TABLE OF CONTENTS

# CHAPTER 1
# INTRODUCTION

## 1. INTRODUCTION

The Food Ordering System project is a desktop application developed using Java and AWT to simplify and enhance the process of ordering food. This project aims to create an intuitive user interface that allows customers to easily select items, specify quantities, and view a detailed order summary with real-time cost calculations. The system is structured into key modules such as User Interface, Order Management, and Notification & Error Handling, each designed to handle different aspects of the application seamlessly. With built-in input validation and error handling, this project ensures a smooth user experience by preventing incorrect data entry and providing immediate feedback.

## 2. OBJECTIVE

The primary objective of the Food Ordering System is to develop a robust, scalable, and efficient order management, input validation, and notification features to ensure a seamless user experience and accurate order processing.. The system aims to:

1. To create a user-friendly desktop application for placing food orders efficiently.
2. To design a clear and interactive graphical user interface (GUI) using Java AWT.
3. To implement input validation and error handling to ensure data accuracy andapplication stability.
4. To develop an order management system capable of calculating the total cost based onitem quantities.
5. To provide users with an organized order summary displaying the details and total price.
6. To integrate modular programming for better code organization, maintenance, and scalability.
7. To enable the application to handle user interactions smoothly with event-driven programming.

8. To ensure the system can display notifications and alert messages for successful operations and errors.
9. To create a reliable and responsive application that handles multiple user inputs simultaneously.
10. To provide a scalable solution that can be easily updated or expanded with additional features.
11. To enhance user engagement through a simple and intuitive interface that simplifiesthe ordering process.

# CHAPTER 2
# PROJECT METHODOLOGY

## 1 PROPOSED WORK

The Food Ordering System aims to provide a user-friendly and efficient platform for customers to place food orders, simplifying the ordering process and ensuring a seamless experience. The methodology focuses on:

1.User Interface Design

- Utilize Java AWT to create a responsive and interactive interface.
- Incorporate clear labels and input fields for easy data entry.

2.Order Processing Logic

- Implement input validation to ensure valid and non-negative quantities.
- Calculate the total cost dynamically based on the selected items and quantities.

3. Error Handling and Notifications

- Use try-catch blocks to manage potential input errors and maintain stability.
- Display user-friendly error and success messages for a better user experience.

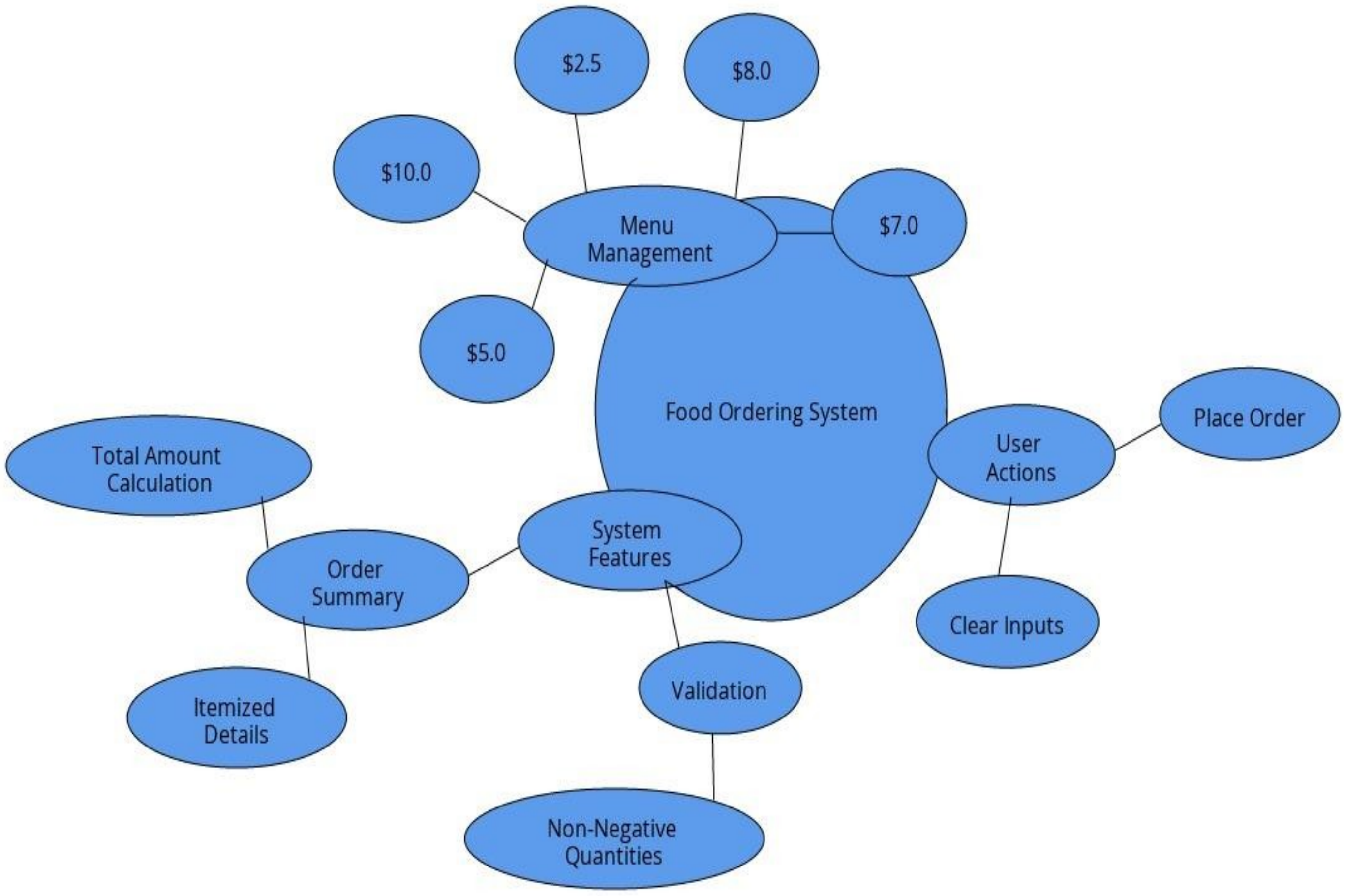
4. Modular Architecture

- Divide the application into modules for better organization and maintenance.
- Ensure each module handles specific tasks, such as UI management and orderprocessing.

**5. Scalability and Future Enhancements**

- Design the system to be scalable, allowing easy updates and feature addition.

## 2 BLOCK DIAGRAM



# CHAPTER 3
# JAVA PROGRAMMING CONCEPTS

## 1. Event-Driven Programming:

The Food Ordering System utilizes event-driven programming, a core concept in Java that allows the application to respond to user actions, such as clicking buttons or entering data. This is implemented through action listeners that detect user interactions and trigger specific methods to handle events like placing an order or clearing inputs. This approach helps create a responsive and interactive interface, enhancing the user experience.

- Utilizes action listeners to detect user interactions, such as clicking buttons andentering data.
- Creates a responsive interface that reacts to user actions in real-time.
- Enables event handling methods to execute specific actions based on user inputs.
- Enhances the overall user experience by making the application more interactive and dynamic.

## 1.1.Exception Handling:

The system uses exception handling to manage and respond to potential errors during runtime. By using try-catchblocks, the application can catch and handle exceptions, suchas invalid input formats or negative values entered by the user.

- Uses try-catchblocks to capture and handle runtime errors, ensuring the programruns smoothly.
- Validates user input to catch exceptions like non-numeric or negative values.
- Prevents the application from crashing due to unexpected input errors.
- Provides informative feedback to users, guiding them to correct their input for a betterexperience.

# CHAPTER 4
# MODULEDESCRIPTION

## 1. MAIN APPLICATION MODULE

### 1.1. Objective

The objective of the Order Management System module is to handle user inputs for selecting menu items and quantities while ensuring accurate cost calculations. It aims to facilitate smooth communication between the user interface and backend processing, providing users with a detailed and correct order summary.

### 1.2. Design

- The module is structured using a clear, modular approach to separate user inputhandling, calculations, and order summary generation.
- Java classes and methods are organized to handle different aspects of the module,allowing for scalability and easy maintenance.

### 1.3. Navigation Workflow:

- Users input their desired quantities for each menu item in the provided fields andsubmit the order.
- The system processes the input, performs validation, and displays the calculated ordersummary, including the total price.

### 1.4. Usage

- Users interact with the module through a graphical user interface where theycan easily input quantities and view results.
- The system provides real-time feedback, showing error messages for invalid data anddisplaying a summary once the order is complete.

1. **User Interface Module:**

1.1. Objective

This module provides users with an intuitive and interactive platform for inputtingdata, navigating the application, and viewing the order summary.

1.2. Submodules

1. Input Handling Submodule:
2. Objective: Captures user inputs for menu item selection and quantity entry.

Features:

- Accepts quantities for various food items.
- Provides real-time validation to prevent invalid or negative entries.

Workflow:

- Users type in the desired quantities for each item.
- The input is validated and sent for processing.

2. **Order Management Module**

2.1. Objective

- Handles user inputs, calculates total order costs, and generates order summaries.

2.2. Submodules

1. Input Processing Submodule:
    - Objective: Captures company details for new employers.
    - Work flow:
        - User input is checked and validated before proceeding.

Features:

- Ensures input is valid and non-negative.
- Sends validated data for cost calculation.

**3. Notification and Error Handling Module**

3.1. Objective

Ensures users are informed of order processing status and alerts them about any errors or issues.

Features

- Utilizes try-catchblocks for exception handling.
- Detects invalid input such as non-numeric or negative values.Workflow:

Errors are caught during user input or processing, triggering feedback to the user.

# CHAPTER 5
# CONCLUSION

## SUMMARY OF ACHIEVEMENTS

The Food Ordering System successfully meets the needs of usersthrough the following:

**For Users:**

- A user-friendly interface for placing and managing orders.
- Real-time validation of inputs to ensure correct data entry.

**For the System:**

- Efficient input processing and accurate cost calculation.
- Clear order summary display with detailed itemization.

**Overall System:**

- A modular design that supports scalability and futureupdates.

## LIMITATIONS

- Limited to basic functionalities (no payment gatewayintegration).
- GUI may appear basic due to AWT's limitations.

## FUTURE SCOPE

- **Database Integration**: Connect the system to a database fororder history and user data storage.
- **Enhanced Features**: Implement payment processing andorder tracking.
- **Mobile Compatibility**: Adapt the system for mobile platforms using frameworks like JavaFX or a cross-platformsolution.
- **User Feedback System**: Add a feedback mechanism forimproving user experience and service quality.

## REFERENCES:

The development of this project involved the use of various resources to understand Java programming principles and implement best practices. The references include:

**Books**

1. *Core Java Volume I: Fundamentals* by Cay S. Horstmann and Gary Cornell
   - Comprehensive guide to Java fundamentals, including OOP concepts and collections.
2. *Effective Java* by Joshua Bloch
   - Focus on practical programming techniques and Java best practices.

**Online Tutorials**

1. **W3Schools Java Tutorials**: https://www.w3schools.com/java/
   - For understanding basic to advanced Java programming concepts.
2. **GeeksforGeeks Java Programming**:https://www.geeksforgeeks.org/java/
   - Detailed explanations of Java collections, object-oriented principles,and practical examples.

**Documentation**

1. **Oracle Java Documentation**: https://docs.oracle.com/javase/tutorial/
   - Official documentation for Java programming and the Collections
   - Framework.
2. **Java SE 17 API Documentation**: https://docs.oracle.com/en/java/javase/17/docs/ap

**Videos**

1. **Programming with Mosh – Java Full Course** (YouTube):https://www.youtube.com/watch?v=grEKMHGYyns
   - Detailed video tutorials covering Java basics to advanced topics.
2. **CodeWithHarry – Java for Beginners** (YouTube):https://www.youtube.com/CodeWithHarry
   - Simplified explanation of Java programming concepts.

## APPENDICES

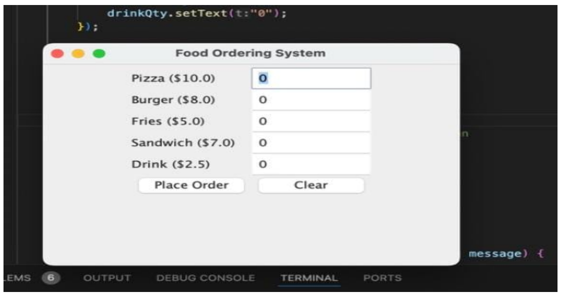
## APPENDIX A – SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
public class FoodOrderingApplication {
    public static void main(String[] args) {
        Frame frame = new Frame("Food Ordering System");
        // Create labels and text fields for menu items
        Label pizzaLabel = new Label("Pizza ($10.0)");
        TextField pizzaQty = new TextField("0", 5);
        Label burgerLabel = new Label("Burger ($8.0)");
        TextField burgerQty = new TextField("0", 5);
         Label friesLabel = new Label("Fries ($5.0)");
        TextField friesQty = new TextField("0", 5);
        Label sandwichLabel = new Label("Sandwich ($7.0)");
        TextField sandwichQty = new TextField("0", 5);
        Label drinkLabel = new Label("Drink ($2.5)");
        TextField drinkQty = new TextField("0", 5);
        // Create buttons
        Button orderButton = new Button("Place Order");
        Button clearButton = new Button("Clear");
        // Create a panel for the form
        Panel panel = new Panel(new GridLayout(6, 2));
        panel.add(pizzaLabel);
        panel.add(pizzaQty);
        panel.add(burgerLabel);
```

```
        panel.add(burgerQty);
        panel.add(friesLabel);
        panel.add(friesQty);
        panel.add(sandwichLabel);
        panel.add(sandwichQty);
        panel.add(drinkLabel);
        panel.add(drinkQty);
        panel.add(orderButton);
        panel.add(clearButton);
        // Action listener for placing order
        orderButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    int pizzaCount = Integer.parseInt(pizzaQty.getText());
                    int burgerCount = Integer.parseInt(burgerQty.getText());
                    int friesCount = Integer.parseInt(friesQty.getText());
                    int sandwichCount = Integer.parseInt(sandwichQty.getText());
                    int drinkCount = Integer.parseInt(drinkQty.getText());

                    if (pizzaCount < 0 || burgerCount < 0 || friesCount < 0 || sandwichCount < 0 ||
drinkCount < 0) {
                        throw new NumberFormatException("Negative value");
                    }

                    double totalAmount = (pizzaCount * 10.0) + (burgerCount * 8.0) +
                                         (friesCount * 5.0) + (sandwichCount * 7.0) +
                                         (drinkCount * 2.5);
                    // Prepare the order summary
                    String summary = "Order Summary:\n"
                        + "Pizza: " + pizzaCount + " x $10.0\n"
                        + "Burger: " + burgerCount + " x $8.0\n"
                        + "Fries: " + friesCount + " x $5.0\n"
                        + "Sandwich: " + sandwichCount + " x $7.0\n"
                        + "Drink: " + drinkCount + " x $2.5\n"
                        + "Total: $" + totalAmount;

                    // Display the order summary
                    displayMessage("Order Summary", summary);
                } catch (NumberFormatException ex) {
                    displayMessage("Error", "Please enter valid non-negative quantities.");
                }
            }
        });

        // Action listener for clearing inputs
        clearButton.addActionListener(e -> {
            pizzaQty.setText("0");
            burgerQty.setText("0");
            friesQty.setText("0");
            sandwichQty.setText("0");
            drinkQty.setText("0");
        });

        // Add the panel to the frame
        frame.add(panel);
        frame.setSize(400, 300);
        frame.setLayout(new FlowLayout());
```

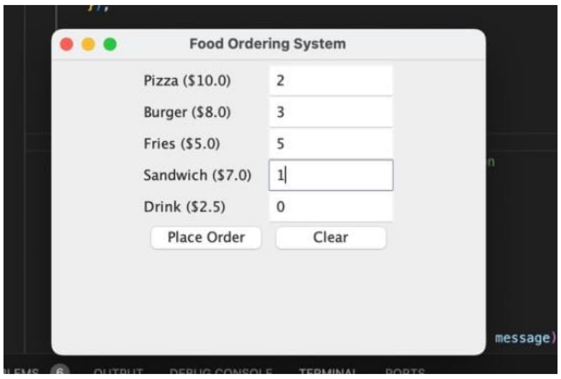
```
        frame.setVisible(true);

        // Add window closing event to close the application
        frame.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                frame.dispose();
            }
        });
    }

    // Utility method to display a message in a new frame
    private static void displayMessage(String title, String message) {
        Frame messageFrame = new Frame(title);
        messageFrame.setSize(350, 300);
        messageFrame.setLayout(new FlowLayout());

        TextArea messageArea = new TextArea(message, 10, 30,
TextArea.SCROLLBARS_VERTICAL_ONLY);
```

# APPENDIX B - SCREENSHOTS

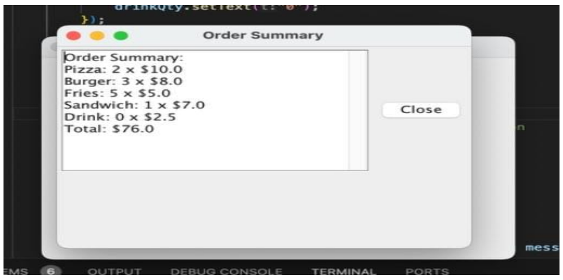


```
drinkQty.setText(t:"0");
});
```

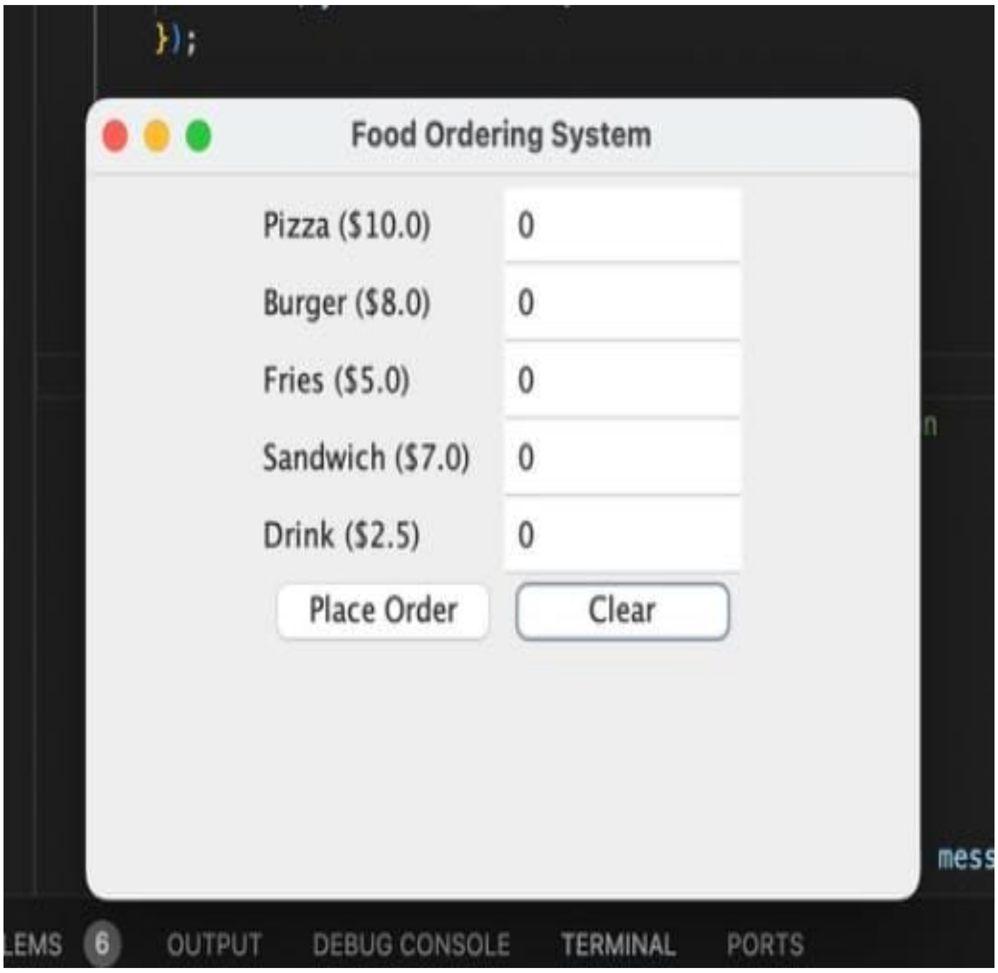
```
Pizza ($10.0)       0
Burger ($8.0)       0
Fries ($5.0)        0
Sandwich ($7.0)     0
Drink ($2.5)        0
```



```
Pizza ($10.0)       2
Burger ($8.0)       3
Fries ($5.0)        5
Sandwich ($7.0)     1
Drink ($2.5)        0
```



```
Order Summary:
Pizza: 2 x $10.0
Burger: 3 x $8.0
Fries: 5 x $5.0
Sandwich: 1 x $7.0
Drink: 0 x $2.5
Total: $76.0
```



```
});
```

Food Ordering System

| | |
|---|---|
| Pizza ($10.0) | 0 |
| Burger ($8.0) | 0 |
| Fries ($5.0) | 0 |
| Sandwich ($7.0) | 0 |
| Drink ($2.5) | 0 |