

PHARMACY MANAGEMENT SYSTEM



A PROJECT REPORT

Submitted by

INIYAN L (2303811710421065)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

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

SAMAYAPURAM – 621 112

NOVEMBER- 2024

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

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**PHARMACY MANAGEMENT SYSTEM**” is the bonafide work of **INIYAN L(2303811710421065)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.,)
SUPERVISOR
ASSISTANT PROFESSOR

SIGNATURE

Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.,),

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held On.....03/12/2024.....

CGB1201-JAVA PROGRAMMING
Mr.MANJUNATH A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

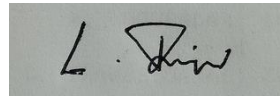
CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8104-DSEC, PERAMBALUR.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**PHARMACY MANAGEMENT SYSTEM**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING**.

Signature

A rectangular box containing a handwritten signature in black ink. The signature appears to be 'L. Iniyan'.

INIYAN L

Place: Samayapuram

Date: 03/12/2024

ACKNOWLEDGEMENT

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

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for 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 to our project and offering adequate duration in completing our project.

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

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs. K. VALLI PRIYADHARSHINI, M.E., (Ph.D.)**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to 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 global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

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 member or 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.

ABSTRACT

The Pharmacy Management System is a simple Java application designed to manage drugs in a pharmacy using an AWT-based GUI. The system allows users to add new drugs to the inventory, sell drugs, and display the current inventory. It uses a Drug class to represent individual drugs with properties such as name, price, and quantity, and provides methods to manipulate. The PharmacyManagementSystemAWT class serves as the main controller, handling user interactions and managing the drug inventory. The application supports basic functionalities like adding drugs with a specified name, price, and quantity, as well as selling drugs and updating their quantities. Dialog boxes are used to show success or error messages to the user. The program also demonstrates key Object-Oriented Programming concepts such as encapsulation, abstraction, and event-driven programming through listeners attached to buttons for user actions. Exception handling is implemented to manage invalid user inputs, ensuring the system remains robust. The ArrayList class is utilized to dynamically store and manage the inventory of drugs. This system can be extended to include additional features like drug search, updating drug details, and reporting, making it scalable for more complex pharmacy management tasks.

Dialog boxes offer feedback for each action, confirming drug addition or notifying the user of errors such as insufficient stock. This system also showcases Java's event-driven programming model, where user actions trigger specific responses through the use of action listeners. The ArrayList collection is leveraged to store the inventory dynamically, allowing the program to handle a flexible number of drugs. The modular structure of the code makes it easily extensible, and future enhancements could include features like sorting the inventory, searching for specific drugs, generating sales reports, and implementing a more advanced database backend for persistent storage.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Pharmacy Management System simplifies drug inventory and sales management through a user-friendly Java-based application. It enables efficient stock tracking, handles invalid inputs gracefully, and ensures accurate updates for real-time operations. Designed with modularity and scalability, the system optimizes resources and supports collaborative enhancements while promoting ethical practices and clear communication. Its development encourages continuous learning and application of advanced programming techniques to create impactful community-oriented solutions.	PO1 -3 PO2 -3 PO3 -3 PO4 -3 PO5 -3 PO6 -3 PO7 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3	PSO1 -3 PSO2 -3 PSO3 -3

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
1	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
2	PROJECT METHODOLOGY	4
	2.1 Proposed Work	4
	2.2 Block Diagram	4
3	MODULE DESCRIPTION	5
	3.1 Drug Class	5
	3.2 Inventory Management	5
	3.3 User Interface (GUI)	5
	3.4 Event Handling	6
	3.5 Message Display	6
4	CONCLUSION & FUTURE SCOPE	7
	4.1 Conclusion	7
	4.2 Future Scope	7
	APPENDIX A (SOURCE CODE)	8
	APPENDIX B (SCREENSHOTS)	13
	REFERENCES	15

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of the Pharmacy Management System is to develop a simple and efficient software application that enables pharmacy staff to manage their drug inventory through a graphical user interface (GUI). To allow the addition of new drugs into the inventory by specifying the drug's name, price, and quantity. This ensures an up-to-date record of available drugs. To provide an intuitive and easy-to-use interface using AWT components like TextField, Button, and TextArea, making it accessible for non-technical users to operate effectively. The system aims to streamline pharmacy inventory management tasks, reduce human error, and enhance efficiency in managing drug stocks.

1.2 Overview

The Pharmacy Management System is a Java-based desktop application designed to help manage pharmaceutical inventories in a simple and intuitive way. Built using AWT (Abstract Window Toolkit), the application provides an interface to add new drugs, process sales transactions, and display the current inventory. It is structured around basic inventory management functionalities, with an emphasis on maintaining stock levels, tracking drug details, and providing users with immediate feedback on actions such as drug sales. The application handles invalid inputs (e.g., non-numeric values for price or quantity) by displaying appropriate error messages in the TextArea. This ensures that only valid data is processed, reducing the chance of system errors.

1.3 Java Programming Concepts

Encapsulation:

- The Drug class uses private fields (e.g., name, price, quantity) to encapsulate data. Public getter and setter methods are provided to access and modify the private data.

Abstraction:

- The Abstraction is achieved in the Drug class by defining its structure and behaviors (methods) without exposing the internal workings. The user interacts with the Drug class through public methods like `getPrice()` and `setQuantity()`, but the actual implementation is hidden.

Inheritance:

- Although not explicitly implemented in the code, the concept of inheritance is utilized through Object class inheritance, as all Java classes implicitly extend the Object class.

Polymorphism:

- Polymorphism is evident in the way Java can treat different types of objects through the same interface, for example methods like `showMessage()` could be overloaded with different parameter types, providing different functionalities for displaying message

- **Swing and AWT (Abstract Window Toolkit) :** The application uses AWT and Swing to build the graphical user interface (GUI).
- **Exception Handling:** The code handles potential runtime errors (such as invalid number formats) using a try-catch block. This is an example of exception handling.
- **Event Handling:** The code makes extensive use of Java's event handling mechanism. The ActionListener interface is used to handle events triggered by UI components (e.g. addButton, sellButton, and displayButton listen for user interactions (clicks), and when clicked, the corresponding actionPerformed() method is executed).
- **String Handling:** The code uses string concatenation to build messages, particularly when displaying inventory or success/error messages. The showMessage() method creates a message string to show the user the drug details.

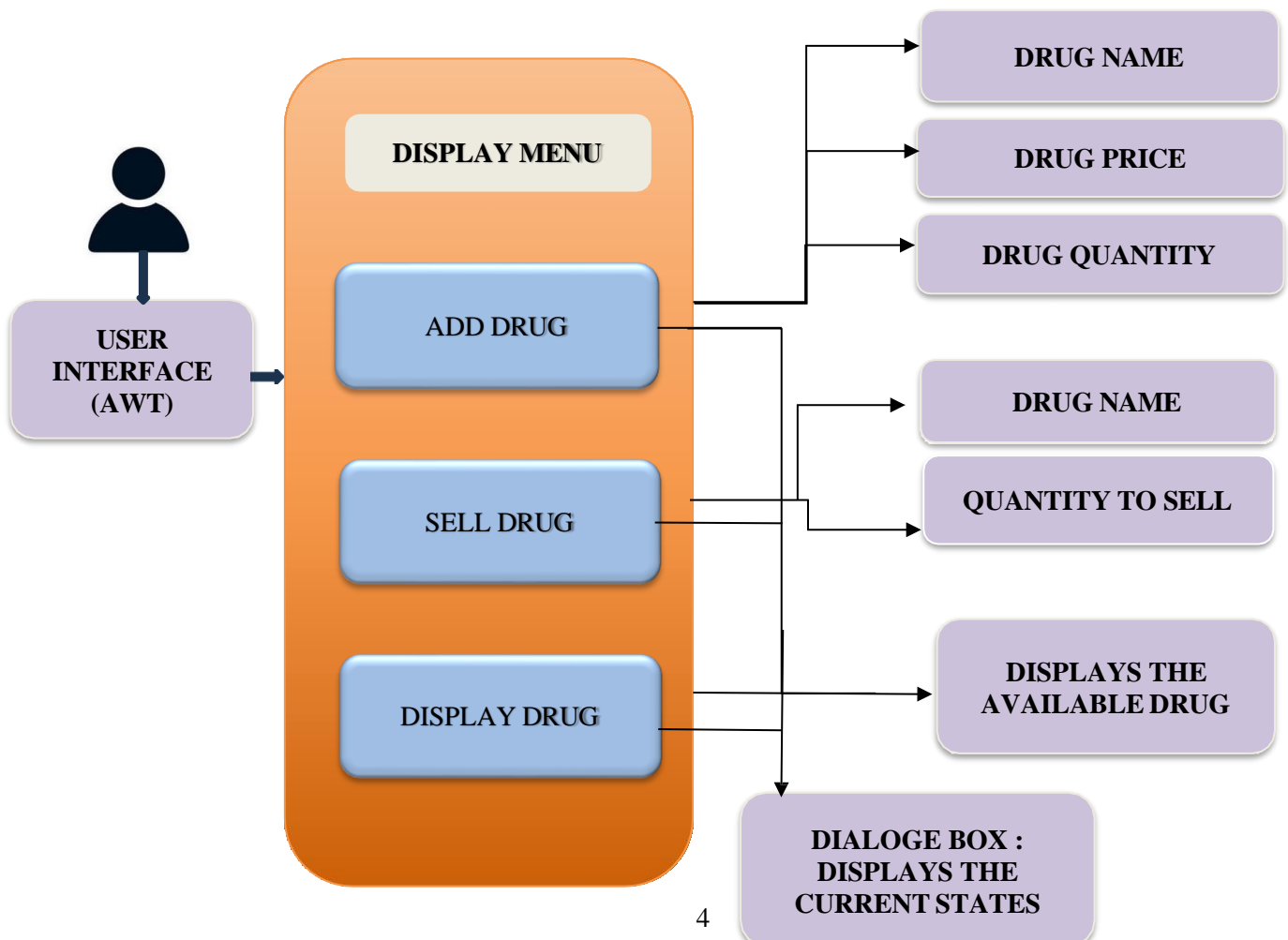
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The Pharmacy Management System is a simple Java application designed to manage drugs in a pharmacy using an AWT-based GUI. The system allows users to add new drugs to the inventory, sell drugs, and display the current inventory. It uses a Drug class to represent individual drugs with properties such as name, price, and quantity, and provides methods to manipulate these properties. The system makes use of dialog boxes (via Dialog class from AWT) to display messages to the user. Dialog boxes are used to show confirmation messages, error messages, or other notifications. The application uses AWT and Swing to build the graphical user interface (GUI).

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1: Drug Management

- This module defines the Drug class, which represents a pharmaceutical product. It contains attributes such as the drug's name, price, and quantity. It provides methods to retrieve these attributes and modify the quantity of the drug, as well as a toString method to display the drug's information in a readable format.

3.2 Module 2: Inventory Management

- This module manages the drug inventory within the system. It allows drugs to be added to the inventory, checked for availability, and sold. It includes methods for adding new drugs, displaying the inventory, and updating drug quantities upon sales, ensuring that the system accurately reflects current stock levels.

3.3 Module 3: User Interface (GUI)

- This module handles the graphical user interface of the Pharmacy Management System using Java AWT components. It creates a window with text fields, buttons, and labels for user input and interaction. Users can add drugs, sell drugs, and view the inventory through a simple GUI, making the system accessible and easy to use.

3.4 Module 4: Event Handling

- This module is responsible for handling user actions like button clicks. It uses event listeners to trigger specific actions when users interact with the interface, such as adding a drug to the inventory, selling a drug, or displaying the current inventory. The module ensures that appropriate methods are called based on user input and provides feedback messages when needed.

3.5 Module 5: Message Display

- This module is responsible for displaying feedback messages to the user. It uses a dialog box to show messages such as successful drug additions, sales confirmations, error notifications, and inventory status. The dialog box ensures that users are informed of the system's actions, errors, or updates in an intuitive manner.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The Pharmacy Management System implemented using Java and AWT successfully demonstrates a robust and user-friendly solution for managing pharmacy operations. The system provides essential functionalities such as inventory management, sales processing, and real-time user feedback through an intuitive graphical user interface. Key strengths of the system include its modular design, which ensures that each functionality is encapsulated within a specific module, enhancing maintainability and scalability. In conclusion, this project demonstrates a practical approach to automating pharmacy management, improving efficiency, reducing errors, and providing a scalable platform for managing drug inventories and sales. It serves as a stepping stone toward more comprehensive and dynamic pharmacy management systems.

4.2 FUTURE SCOPE

- **QR Code Generation:** Automatically generate QR codes for each drug for quick access and management.
- **Login System:** Implement user authentication with usernames and passwords for different types of users (e.g., admin, pharmacist, cashier).
- **Monthly/Annual Reports:** Generate sales reports showing total revenue, top-selling drugs, etc.
- **Expiry Date Tracking:** Add an expiration date field for drugs and alert users when a drug is about to expire or has expired.

CHAPTER 5 APPENDIX A

(SOURCE CODE)

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

// Class representing a drug
class Drug {
    private String name;
    private double price;
    private int quantity;

    public Drug(String name, double price, int quantity) {
        this.name = name;
        this.price = price;
        this.quantity = quantity;
    }

    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }

    public int getQuantity() {
        return quantity;
    }

    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }

    @Override
    public String toString() {
        return "Drug Name: " + name + ", Price: $" + price + ", Quantity: " + quantity;
    }
}

// Main class representing the pharmacy management system
```

```

public class PharmacyManagementSystemAWT {
    private ArrayList<Drug> inventory;
    private TextField nameField, priceField, quantityField, sellNameField,
sellQuantityField;

    public PharmacyManagementSystemAWT() {
        inventory = new ArrayList<>();
    }

    // Method to add a drug to the inventory
    public void addDrug(String name, double price, int quantity, Frame parentFrame) {
        inventory.add(new Drug(name, price, quantity));
        showMessage("Drug added: " + name, parentFrame);
    }

    // Method to display the inventory in a dialog box
    public void displayInventory(Frame parentFrame) {
        if (inventory.isEmpty()) {
            showMessage("No drugs available in the inventory.", parentFrame);
        } else {
            StringBuilder inventoryList = new StringBuilder();
            for (Drug drug : inventory) {
                inventoryList.append(drug).append("\n");
            }
            showMessage(inventoryList.toString(), parentFrame);
        }
    }

    // Method to sell a drug
    public void sellDrug(String name, int quantity, Frame parentFrame) {
        for (Drug drug : inventory) {
            if (drug.getName().equalsIgnoreCase(name)) {
                if (drug.getQuantity() >= quantity) {
                    drug.setQuantity(drug.getQuantity() - quantity);
                    showMessage("Sold " + quantity + " units of " + name, parentFrame);
                    return;
                } else {
                    showMessage("Insufficient stock of " + name, parentFrame);
                    return;
                }
            }
        }
        showMessage("Drug not found in the inventory.", parentFrame);
    }
}

```

```

// Method to show message dialog
public void showMessage(String message, Frame parentFrame) {
    Dialog dialog = new Dialog(parentFrame, "Message", true);
    dialog.setLayout(new FlowLayout());
    dialog.setSize(300, 150);

    Label messageLabel = new Label(message);
    Button okButton = new Button("OK");

    okButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            dialog.setVisible(false);
        }
    });

    dialog.add(messageLabel);
    dialog.add(okButton);
    dialog.setVisible(true);
}

// GUI setup
public void createGUI() {
    Frame frame = new Frame("Pharmacy Management System");

    // Labels
    Label nameLabel = new Label("Drug Name:");
    Label priceLabel = new Label("Drug Price:");
    Label quantityLabel = new Label("Drug Quantity:");
    Label sellNameLabel = new Label("Drug Name to Sell:");
    Label sellQuantityLabel = new Label("Quantity to Sell:");

    // Enlarged TextFields for better visibility
    nameField = new TextField(20); // 20 columns wide
    priceField = new TextField(20); // 20 columns wide
    quantityField = new TextField(20); // 20 columns wide
    sellNameField = new TextField(20); // 20 columns wide
    sellQuantityField = new TextField(20); // 20 columns wide

    // Buttons
    Button addButton = new Button("Add Drug");
    Button sellButton = new Button("Sell Drug");
    Button displayButton = new Button("Display Inventory");

```

```

// Set Layout
frame.setLayout(new FlowLayout());

// Add components to the frame
frame.add(nameLabel);
frame.add(nameField);
frame.add(priceLabel);
frame.add(priceField);
frame.add(quantityLabel);
frame.add(quantityField);
frame.add(addButton);

frame.add(sellNameLabel);
frame.add(sellNameField);
frame.add(sellQuantityLabel);
frame.add(sellQuantityField);
frame.add(sellButton);

frame.add(displayButton);

// Add event listeners
addButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            String name = nameField.getText();
            double price = Double.parseDouble(priceField.getText());
            int quantity = Integer.parseInt(quantityField.getText());
            addDrug(name, price, quantity, frame);
        } catch (NumberFormatException ex) {
            showMessage("Invalid price or quantity input.", frame);
        }
    }
});

sellButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            String name = sellNameField.getText();
            int quantity = Integer.parseInt(sellQuantityField.getText());
            sellDrug(name, quantity, frame);
        } catch (NumberFormatException ex) {
            showMessage("Invalid quantity input.", frame);
        }
    }
});

```

```

});

displayButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        displayInventory(frame);
    }
});

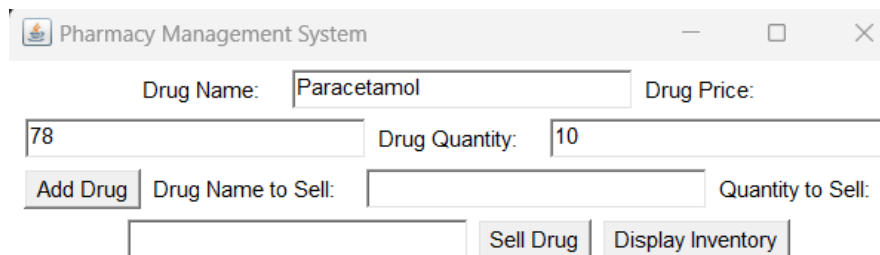
// Frame settings
frame.setSize(500, 400);
frame.setVisible(true);
frame.addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        System.exit(0);
    }
});
}

// Main method to run the pharmacy management system
public static void main(String[] args) {
    PharmacyManagementSystemAWT system = new
PharmacyManagementSystemAWT();
    system.createGUI();
}
}

```

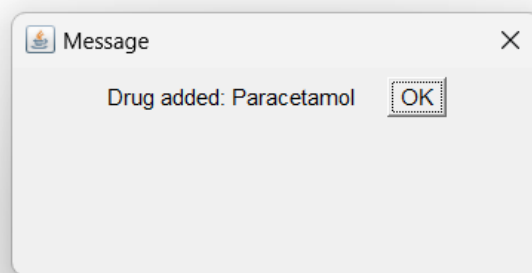
APPENDIX B

(SCREENSHOTS)

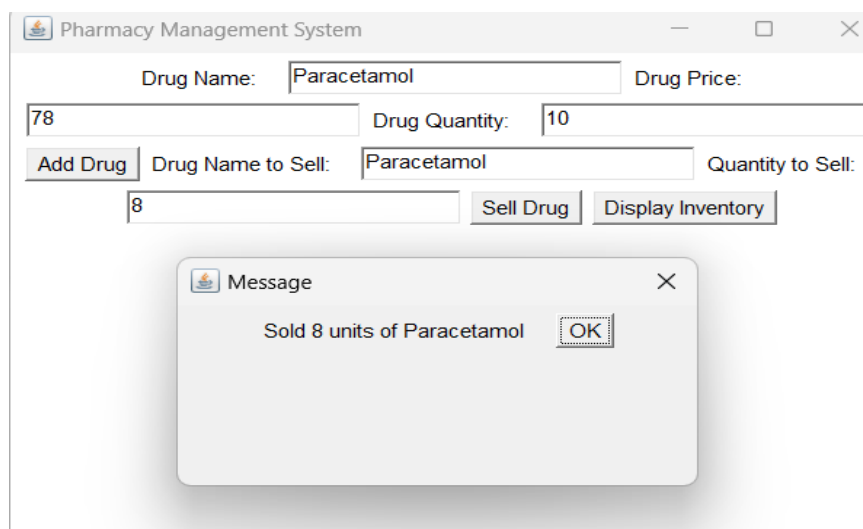


The screenshot shows the 'Pharmacy Management System' window. It contains the following fields and buttons:

- Drug Name:** A text box containing 'Paracetamol'.
- Drug Price:** A text box containing '78'.
- Drug Quantity:** A text box containing '10'.
- Add Drug:** A button.
- Drug Name to Sell:** A text box (empty).
- Quantity to Sell:** A text box (empty).
- Sell Drug:** A button.
- Display Inventory:** A button.



ADD DRUG



The screenshot shows the 'Pharmacy Management System' window. It contains the following fields and buttons:

- Drug Name:** A text box containing 'Paracetamol'.
- Drug Price:** A text box containing '78'.
- Drug Quantity:** A text box containing '10'.
- Add Drug:** A button.
- Drug Name to Sell:** A text box containing 'Paracetamol'.
- Quantity to Sell:** A text box containing '8'.
- Sell Drug:** A button.
- Display Inventory:** A button.

Below the main form, there is a 'Message' dialog box with the text 'Sold 8 units of Paracetamol' and an 'OK' button.

SELL DRUG

Pharmacy Management System

Drug Name: Drug Price: Drug Quantity: Drug Name to Sell: Quantity to Sell:

Message

Drug Name: Paracetamol, Price: \$78.0, Quantity: 2Drug Name: Paracetamol, Price: \$78.0, Quantity: 10Drug Name: Paracetamol, Price: \$78.0, Quantity: 10Drug Name: Cold act, Price: \$99.0, Quantity: 13

DISPLAY INVENTORY

REFERENCES

1. Java Documentation Official Java Platform Documentation by Oracle
<https://docs.oracle.com/javase/>
2. OOP Principles and Design Patterns *Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software.*
3. Schildt, H. (2018). *Java: The Complete Reference* (11th Edition, Vol. 1, pp. 326–410). McGraw-Hill Education.