ATM SIMULATION SYSTEM

A PROJECT REPORT

Submitted by

KEERTHANA R(2303811710422077)

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 **"ATM SIMULATION SYSTEM"** is the bonafide work of **KEERTHANA R(2303811710422077)** 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
ASSISTANTP ROFESSOR

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.MALARMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Ms.R.ARUNA PRIYA, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8104-DSEC, PERAMBALUR.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **“ATM SIMULATION 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

R. Keerthana.

KEERTHANA R

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 ATM Simulation program is a Java-based application designed to emulate core banking operations such as withdrawals, deposits, balance inquiries, and user authentication through a graphical user interface (GUI) built using Java AWT. It demonstrates essential programming concepts like event-driven programming and object-oriented principles, including encapsulation and abstraction. The application ensures secure transactions through PIN-based authentication and provides real-time feedback to users. It is a scalable and modular system, serving as a prototype for developing more advanced financial software. The program aims to provide a practical learning platform for students and developers while paving the way for future enhancements like database integration, fund transfers, and advanced security measures.

ABSTRACT WITH POs AND PSOs MAPPING

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

ABSTRACT	POs MAPPED	PSOs MAPPED
The ATM Simulation program is a Java-based application designed to emulate core banking operations such as withdrawals, deposits, balance inquiries, and user authentication through a graphical user interface (GUI) built using Java AWT.	**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 1

INTRODUCTION

1.1 Objective:

The objective of the ATM Simulation program is to emulate a secure and user-friendly banking system that allows users to log in, perform withdrawals, deposits, and balance inquiries while demonstrating core Object-Oriented Programming (OOP) principles such as encapsulation, abstraction, and inheritance. It aims to provide a practical understanding of GUI programming using Java's AWT library, emphasizing modularity, error handling, and real-time feedback to simulate real-world scenarios like login authentication and insufficient funds. Additionally, the program is designed to be scalable, serving as a foundation for more complex financial systems, and highlights the importance of cross-platform compatibility, making it a valuable tool for learning, testing, and further development.

1.2 Overview:

The ATM Simulation program is a Java-based application designed to replicate essential banking functionalities in a secure and interactive manner. It provides users with a graphical user interface (GUI) built using Java AWT, enabling them to log in with an account number and PIN, perform deposits, withdrawals, and check their account balance. The system employs robust error handling to manage invalid inputs and ensures secure transactions through authentication. With features such as modular design, real-time feedback, and platform independence, the program not only mimics the behavior of a real ATM but also demonstrates key programming concepts like event-driven programming and object-oriented principles. It serves as a practical example for educational purposes and a foundation for developing more sophisticated financial software.

1.3 Java Programming Concepts:

- **Encapsulation:** The program encapsulates data like balance, accountNumber, and pin within the class, ensuring these attributes are not directly accessible and can only be manipulated through methods like validateLogin(), withdraw(), deposit(), and checkBalance().

- **Abstraction:** The program abstracts the implementation details of banking operations, such as login validation and balance updates, providing only the essential interfaces (methods) for interaction with the user.

- **Inheritance:** The program uses inheritance as the ATMSimulation class extends the Frame class from AWT, inheriting its properties and methods to create a graphical user interface.

- **Polymorphism:** Polymorphism is evident in the use of event handling through the ActionListener interface, where different actions (e.g., deposit, withdraw, login) are implemented uniquely for specific buttons.

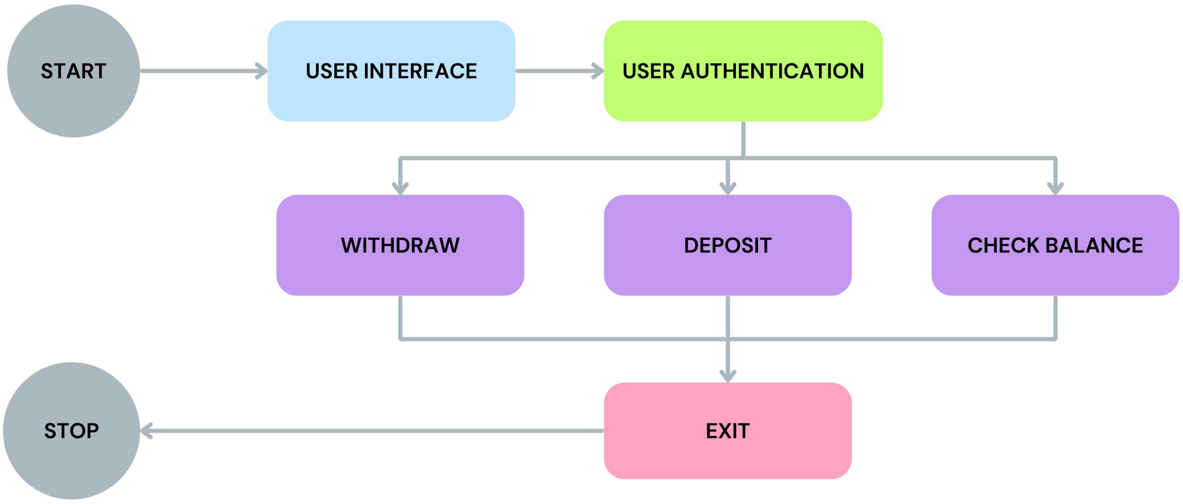
- **Classes and Objects:** The program is designed around the ATMSimulation class, and an object of this class (atm) is created in the main method to initiate the application.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work:

The proposed work for the ATM Simulation program aims to enhance its functionality and user experience by integrating advanced features and improving existing capabilities. Key enhancements include implementing multi-factor authentication for improved security, adding transaction history to track user activities, and enabling fund transfer functionality between accounts. The graphical interface will be modernized for better usability, with color-coded alerts and tooltips. Additionally, the system will support multiple accounts, provide localization options for a diverse user base, and integrate a backend database for dynamic data management. Future scalability will allow for advanced features such as utility bill payments and investment management. These upgrades aim to make the program a more robust, realistic, and educational tool for understanding banking systems and software development.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 User Authentication Module:

Validates the account number and PIN entered by the user. Enables access to other functionalities upon successful login.

3.2 Withdrawal Module:

Allows users to withdraw money, checks for sufficient balance, and updates the account balance. Ensures the entered amount is valid and non-negative.

3.3 Deposit Module:

Facilitates depositing money into the account. Validates the input amount and updates the balance accordingly.

3.4 Balance Inquiry Module:

Displays the current account balance to the user. Provides feedback in the display area after successful login.

3.5 Graphical User Interface (GUI) Module:

Manages the graphical user interface using AWT components like buttons, text fields, labels, and text areas. Handles user interaction and provides feedback for each operation.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

In conclusion, the ATM Simulation program serves as a practical and interactive application that demonstrates core banking functionalities and key programming concepts in Java. With its secure authentication, user-friendly interface, and modular design, it provides an excellent foundation for understanding GUI programming and object-oriented principles. While the current system effectively mimics basic ATM operations, the proposed enhancements such as advanced security features, transaction history, and database integration pave the way for transforming it into a more robust and realistic banking application. This program not only serves as a valuable learning tool but also has the potential to be scaled and adapted for real-world financial systems.

4.2 FUTURE SCOPE

Future implementations for the ATM Simulation program focus on enhancing its functionality, security, and user experience. Integrating a database to manage dynamic user data and enabling multi-factor authentication can improve scalability and security. Adding features such as transaction history, fund transfers, and bill payments will expand the program's utility, making it more versatile. Upgrading the user interface with modern frameworks and supporting multiple languages will enhance usability and accessibility. Advanced functionalities like personalized profiles, AI-driven virtual assistants, and fraud detection systems can bring a touch of innovation. Additionally, supporting multiple currencies, integrating real-time currency conversion, and extending the application to mobile and web platforms will make the system more practical and aligned with contemporary banking solutions. These advancements will transform the simulation into a robust, real-world-ready application while maintaining its educational value.

APPENDIX A
(SOURCE CODE)

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

public class ATMSimulation extends Frame {
    private TextField accountField, amountField, pinField;
    private TextArea displayArea;
    private Button withdrawButton, depositButton, balanceButton, exitButton;
    private double balance = 1000.00; // Initial balance
    private int accountNumber = 12345;
    private int pin = 1234;
    private boolean isLoggedIn = false;

    public ATMSimulation() {
        setTitle("ATM Simulation");
        setSize(400, 500);
        setLayout(new FlowLayout());
        setBackground(Color.LIGHT_GRAY);

        // Initialize components
        Label accountLabel = new Label("Account Number: ");
        accountField = new TextField(20);
        Label pinLabel = new Label("PIN: ");
        pinField = new TextField(20);
        pinField.setEchoChar('*');

        Label amountLabel = new Label("Amount: ");
```

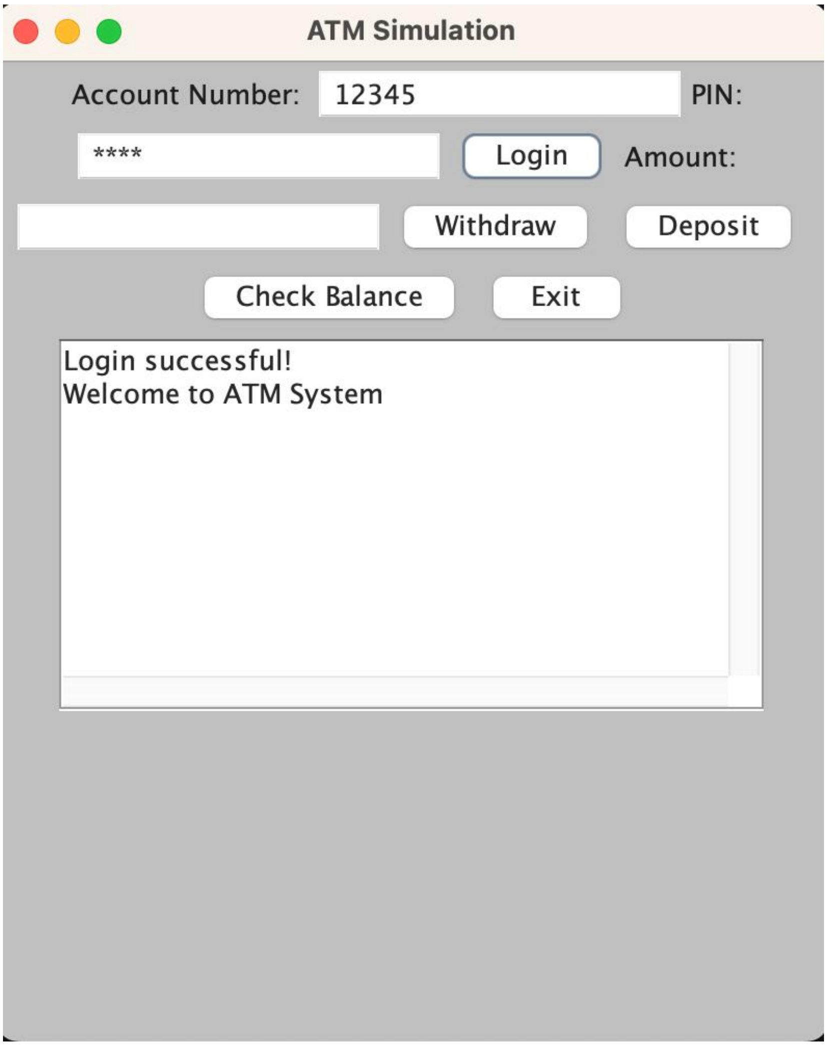
```
amountField = new TextField(20);

displayArea = new TextArea(10, 40);
displayArea.setEditable(false);

Button loginButton = new Button("Login");
withdrawButton = new Button("Withdraw");
depositButton = new Button("Deposit");
balanceButton = new Button("Check Balance");
exitButton = new Button("Exit");

// Initially disable transaction buttons
withdrawButton.setEnabled(false);
depositButton.setEnabled(false);
balanceButton.setEnabled(false);

// Add components
add(accountLabel);
add(accountField);
add(pinLabel);
add(pinField);
add(loginButton);
add(amountLabel);
add(amountField);
add(withdrawButton);
add(depositButton);
add(balanceButton);
add(exitButton);
add(displayArea);
```

```
// Login button action
loginButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        validateLogin();
    }
});

// Withdraw button action
withdrawButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        withdraw();
    }
});

// Deposit button action
depositButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        deposit();
    }
});

// Balance button action
balanceButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        checkBalance();
    }
});
```

```
        // Exit button action
        exitButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                System.exit(0);
            }
        });

        // Window closing event
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    private void validateLogin() {
        try {
            int inputAccount = Integer.parseInt(accountField.getText());
            int inputPin = Integer.parseInt(pinField.getText());

            if (inputAccount == accountNumber && inputPin == pin) {
                isLoggedIn = true;
                withdrawButton.setEnabled(true);
                depositButton.setEnabled(true);
                balanceButton.setEnabled(true);
                displayArea.setText("Login successful!\nWelcome to ATM System");
            } else {
                displayArea.setText("Invalid account number or PIN!");
            }
        } catch (NumberFormatException e) {
            displayArea.setText("Please enter valid numbers!");
        }
    }

    private void withdraw() {
        if (!isLoggedIn) {
            displayArea.setText("Please login first!");
            return;
        }

        try {
            double amount = Double.parseDouble(amountField.getText());
            if (amount > 0 && amount <= balance) {
                balance -= amount;
                displayArea.setText(String.format("Withdrawal successful!\nAmount:
$%.2f\nNew Balance: $%.2f",
                    amount, balance));
            } else {
                displayArea.setText("Invalid amount or insufficient funds!");
            }
        } catch (NumberFormatException e) {
            displayArea.setText("Please enter a valid amount!");
        }
        amountField.setText("");
    }

    private void deposit() {
        if (!isLoggedIn) {
        }

            displayArea.setText("Please login first!");
            return;
        }

        try {
            double amount = Double.parseDouble(amountField.getText());
            if (amount > 0) {
                balance += amount;
                displayArea.setText(String.format("Deposit successful!\nAmount:
$%.2f\nNew Balance: $%.2f",
                    amount, balance));
            } else {
                displayArea.setText("Please enter a positive amount!");
            }
        } catch (NumberFormatException e) {
            displayArea.setText("Please enter a valid amount!");
        }
        amountField.setText("");
    }

    private void checkBalance() {
        if (!isLoggedIn) {
            displayArea.setText("Please login first!");
            return;
        }
        displayArea.setText(String.format("Current Balance: $%.2f", balance));
    }

    public static void main(String[] args) {
        ATMSimulation atm = new ATMSimulation();
        atm.setVisible(true);
    }
}
```

APPENDIX B

(SCREENSHOTS)

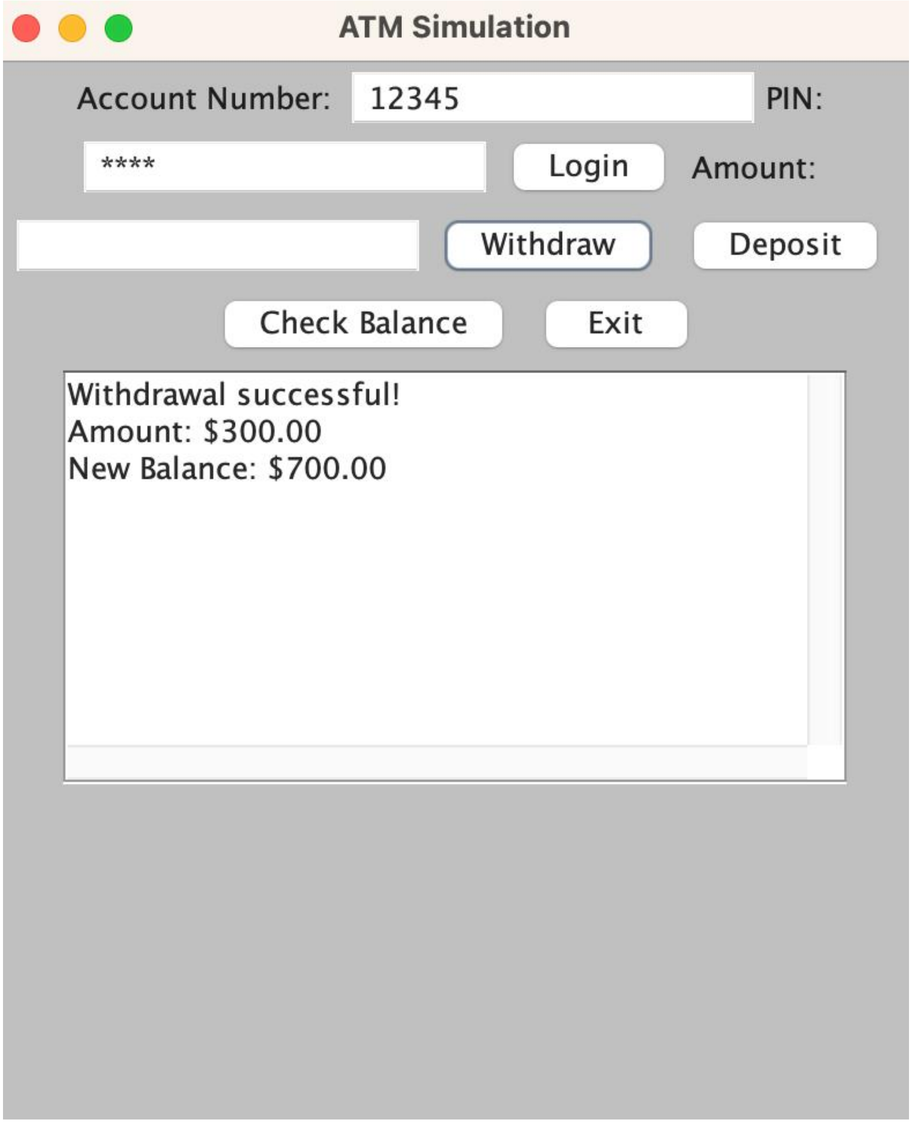


Account Number: 12345

PIN: ****

Amount:

```
Login successful!
Welcome to ATM System
```

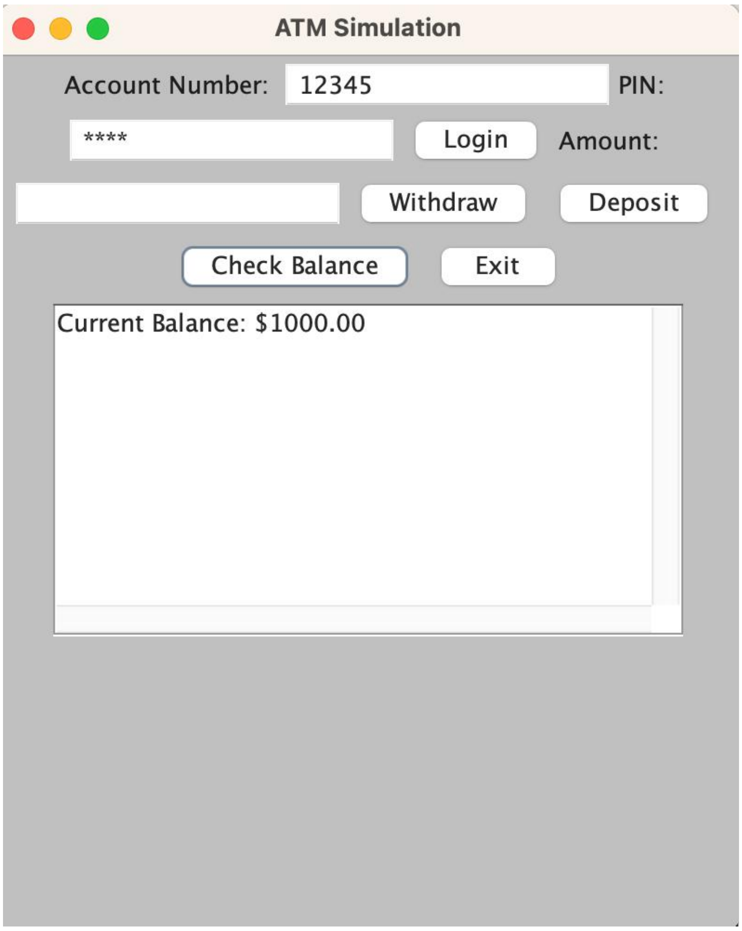


Account Number: 12345

PIN: ****

Amount:

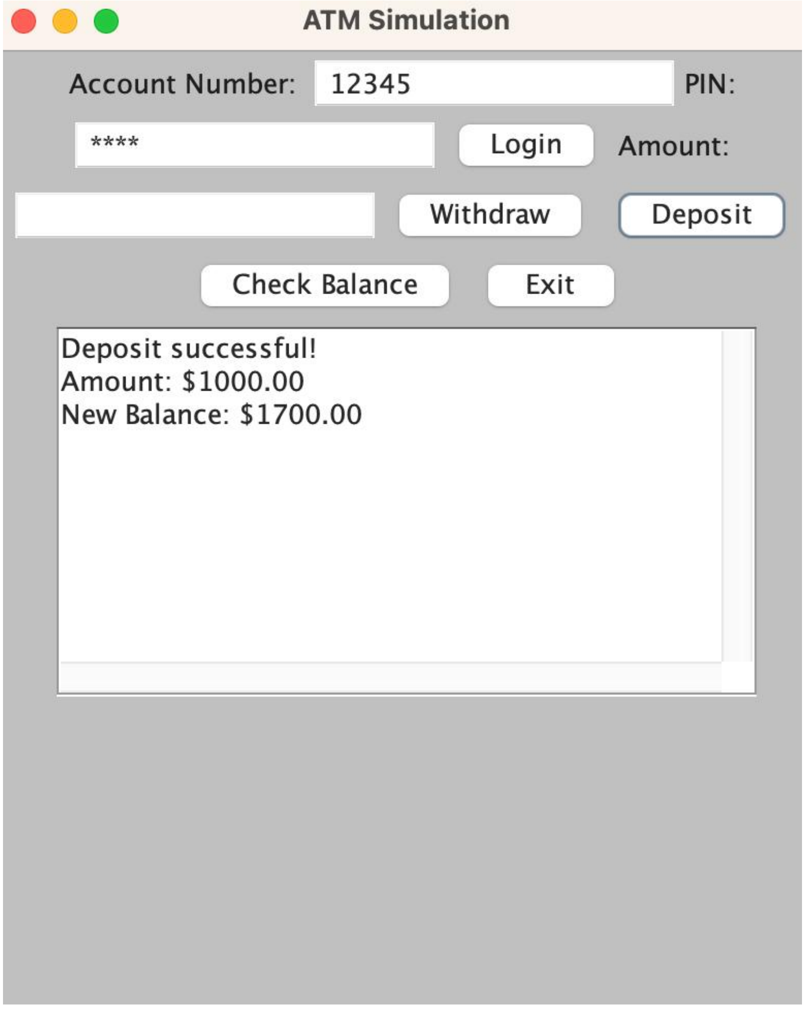
```
Withdrawal successful!
Amount: $300.00
New Balance: $700.00
```



Account Number: 12345

PIN: ****

```
Current Balance: $1000.00
```



Account Number: 12345

PIN: ****

```
Deposit successful!
Amount: $1000.00
New Balance: $1700.00
```

REFERENCES

Website:

Reference link: https://chatgpt.com/c/674d7de6-a490-8005-9bd3-c8c1c15eb74a

Youtube link: https://www.youtube.com/watch?v=K2Tqubig4Eo

Github Reference link:

https://github.com/shivamverma26/ATM_Simulator/tree/main/ATM-Simulator-System/src/ASimulatorSystem