ATM SIMULATION

ATM SIMULATION

DEPARTMENT OF ARTIFICIAL INTELIGANCE AND MACHINE LEARING

BONAM VENKATA CHALAMAYYA INSTITUTE OF TECHNOLOGY AND SCIENCE

A MINI PROJECT BASED ON

ATM SIMULATION

Submitted by

1. PALAGUMMI SRI SAI VINAY
2. Raj
3. TRIPURA PUTREVU
4. KARUNA PREETHI

Under the esteemed guidance of
Mr. AKSHAY KUMAR, M.Tech

ACKNOWLEDGEMENT

It gives us immense pleasure to acknowledge all those who helped us throughout in making this project a great success.

We would like to express our heartfelt gratitude to our guide **AKSHAY KUMAR SIR** for his continuous guidance, valuable suggestions, and constant encouragement throughout the completion of this project. His expertise and support have been instrumental in helping us understand the concepts involved in programming and system design.

We also extend our sincere thanks to the **Department of Artificial Intelligence and Machine Learning (AIML)** and the management of **BONAM VENKATA CHALAMAYYA INSTITUTE OF TECHNOLOGY AND SCIENCE** for providing us with the facilities and an excellent learning environment.

Finally, we are pleased to acknowledge our indebtedness to all those who devoted themselves directly to make this project work a total success.

1. PALAGUMMI SRI SAI VINAY
2. Raj
3. TRIPURA PUTREVU
4. KARUNA PREETHI

ABSTRACT

The **ATM Simulation Project** replicates the fundamental operations of a real-world Automated Teller Machine. It allows users to perform basic transactions like balance inquiry, withdrawal, and deposit after proper authentication. This project is developed using the **C programming language**, relying only on **conditional statements** (`if`, `else if`, `else`). It helps students understand logic building, input handling, and transaction flow control — all in a secure and interactive way.

TABLE OF CONTENTS

CHAPTER-1

INTRODUCTION

INTRODUCTION

An Automated Teller Machine (ATM) is a device that allows customers to perform a variety of financial transactions without the need for direct interaction with bank staff. These transactions typically include withdrawing money, depositing funds, and checking account balances. ATMs provide convenience and accessibility, making banking services available 24/7.

This project aims to simulate the core operations of an ATM using the C programming language. The simulation focuses on creating a console-based system that mimics real-world ATM functionality, enabling users to carry out transactions in a structured and interactive manner.

To ensure the system is secure and reliable, the project incorporates user authentication, input validation, and error handling. These features help prevent unauthorized access, handle incorrect inputs gracefully, and provide a realistic and robust experience for users interacting with the simulated ATM**.**

CHAPTER-2

PROBLEM STATEMENT

PROBLEM STATEMENT

The objective of this project is to design and develop a console-based C program that simulates an Automated Teller Machine (ATM) system. The system should enable users to securely log in using either a card with a PIN or an account number with OTP verification.

Once authenticated, the program should allow users to perform core banking operations such as withdrawing funds, depositing money, checking account balances, and exiting the system. All operations must be handled accurately, ensuring proper validation of inputs and maintaining the integrity of account data.

The program's logic must be implemented solely using conditional statements (if, else if, and else), demonstrating control flow management in C programming while maintaining a secure, reliable, and user-friendly interface.

CHAPTER-3

OBJECTIVES OF THE PROJECT

Objectives of the Project:

- To design and implement an ATM simulation that demonstrates core banking operations using structured programming logic.
- To strengthen understanding and practical application of conditional statements in the C programming language.
- To develop a secure, user-interactive, and reliable console-based system for performing financial transactions.
- To provide insight into the workflow of banking operations, including transaction validation and error handling.

CHAPTER-4

SYSTEM REQURMENTS

System Requirements

Hardware Requirements:

- **Processor**: Intel i3 or higher
- **RAM**: 2 GB or more
- **Hard Disk:** Minimum 100 MB of free space

Software Requirements:

- **Operating System**: Windows or Linux
- **Compiler:** Turbo C or GCC
- **Programming Language**: C

CHAPTER-5

SYSTEM DESIGN AND WORKING

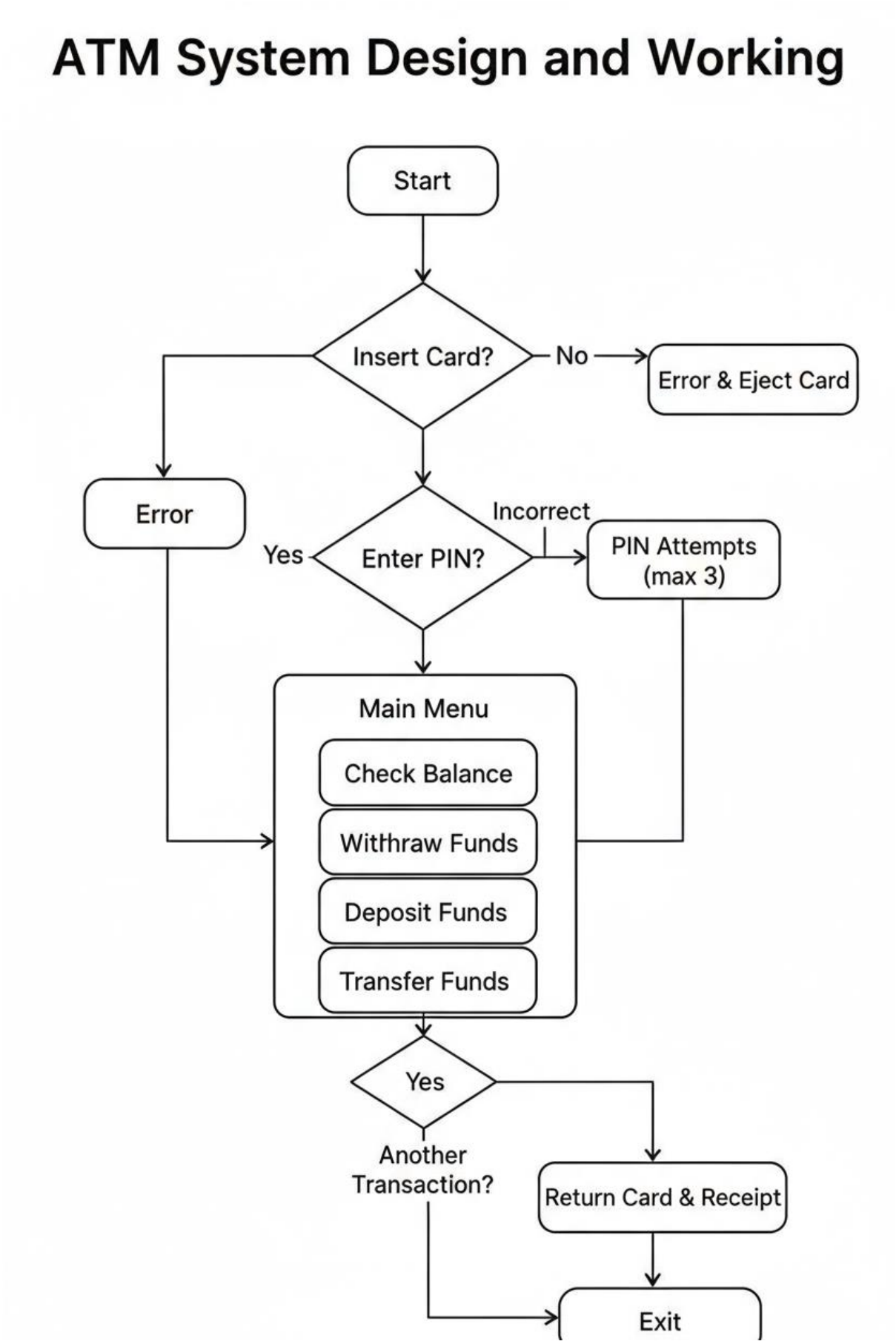
System Design and Working

The ATM simulation starts by prompting the user to indicate whether they possess a card. Based on the user's response:

- **No card:** The system requests the user to enter a 16-digit account number followed by OTP verification for authentication.
- **Card present:** The card is inserted virtually, and the user is prompted to enter their PIN for authentication.

Upon successful login, the user is presented with a menu of operations:

- **View Balance** – Check the current account balance.
- **Withdraw Money** – Withdraw funds with validation for available balance and transaction limits.
- **Deposit Money** – Deposit funds into the account with validation of input amounts.
- **Exit** – Terminate the session securely.


ATM System Design and Working
Start
Insert Card?
No
Error & Eject Card
Error
Yes
Enter PIN?
Incorrect
PIN Attempts
(max 3)
Main Menu
Check Balance
Withdraw Funds
Deposit Funds
Transfer Funds
Yes
Another
Transaction?
Return Card & Receipt
Exit

Each transaction is processed by updating the account balance accordingly, while the system ensures input validation, adherence to transaction limits, and correct handling of denominations to provide a realistic and secure ATM experience.

CHAPTER-6

MODULES DESCRIPTION

Modules Description

1. Authentication Module

This module is responsible for verifying the identity of the user. Authentication is performed using either a Personal Identification Number (PIN) or a One-Time Password (OTP) to ensure secure access to the system.

2. Account Module

The Account Module manages the user's account details, including current balance. It also ensures that all transactions comply with minimum balance requirements and account rules.

3. Transaction Module

This module handles all transactional operations such as withdrawals, deposits, and balance inquiries. It ensures that each transaction is executed accurately and securely.

4. Menu and Exit Module

The Menu and Exit Module provides the user with a clear and interactive transaction menu. It also allows users to exit the system safely, ensuring that all session data is properly managed.

CHAPTER-7

IMPLEMENTATION

Implementation

The ATM program is structured entirely around decision-making constructs, relying exclusively on if, else if, and else statements to control the application's logic and direct user interactions. By carefully layering these conditional statements, the system provides a seamless and linear process that guides users through card authentication, account type selection, and a variety of transaction operations, such as balance inquiry, deposits, and withdrawals. Each decision point dynamically adapts to user inputs, allowing only legitimate operations and providing immediate feedback for invalid actions or data entries, thus maintaining both functionality and security in the absence of iterative structures or switch-case statements.

Input validation and error-handling have been meticulously implemented at every critical step to ensure safe, accurate processing. For instance, the program checks the completeness of the 16-digit account number, validates OTP and PIN formats, and enforces positive transaction values for deposits and withdrawals. Further protective measures include balance sufficiency checks and maintenance of minimum account requirements as appropriate. Should users supply incorrect or out-of-range values, the program gracefully alerts them and, when necessary, terminates further actions to prevent unauthorized or erroneous transactions, keeping user accounts secure and recordkeeping consistent.

The overall design delivers a clear, modular execution flow where each branch precisely defines the permitted path based on successive inputs, without relying on repetition constructs or case-based navigation. This approach highlights the power and versatility of conditional logic in managing a full-featured ATM simulation, showcasing how careful control flow and thoughtful validation routines can replace more complex or repetitive programming techniques, ultimately yielding a secure and transparent user experience.

CHAPTER-8

PROGRAM CODE

8.1 SOURCE CODE

SOURCE CODE

```
#include <stdio.h>

int main() {
   int has, acc, choice, pin;
   long long num;
   float bal = 50000.0, a, b;
   int otp, ver = 0;

   printf("Do you have a card? (1 for Yes / 0 for No): ");
   scanf("%d", &has);

   if (has == 0) {
      printf("Please enter your 16-digit account number: ");
      scanf("%lld", &num);
      if (num >= 1000000000000000LL && num <= 9999999999999999LL) {
         printf("otp has been sent to your registered mobile number\n");
      } else {
         printf("Invalid number. It must be exactly 16 digits.\n");
         printf("Enter your 16-digit account number again: ");
         scanf("%lld", &num);
         if (num >= 1000000000000000LL && num <= 9999999999999999LL) {
            printf("otp has been sent to your registered mobile number\n");
         } else {
            printf("Invalid number. It must be exactly 16 digits.\n");
            printf("Enter your 16-digit account number again: ");
            scanf("%lld", &num);
```

```
            if (num >= 1000000000000000LL && num <= 9999999999999999LL) {
                printf("An OTP has been sent to your registered mobile number.\n");
            } else {
                printf("Maximum attempts reached. Exiting...\n");
                return 0;
            }
        }
    }

    printf("Enter 6 digit OTP: ");
    scanf("%d", &otp);

    if (otp >= 100000 && otp <= 999999) {
        ver = 1;
    } else {
        printf("Invalid OTP. Please try again.\n");
        printf("Enter 6-digit OTP: ");
        scanf("%d", &otp);

        if (otp >= 100000 && otp <= 999999) {
            ver = 1;
        } else {
            printf("Invalid OTP. Please try again.\n");
            printf("Enter 6-digit OTP: ");
            scanf("%d", &otp);

            if (otp >= 100000 && otp <= 999999) {
                ver = 1;
            } else {
```

```
                printf("Maximum OTP attempts reached. Exiting...\n");
                return 0;
            }
        }
    }
} else if (has == 1) {
    printf("Please insert your card.");
    char ch;
    scanf("%c%c", &ch, &ch);
    printf("Card inserted successfully.\n");
} else {
    printf("Invalid input. Please restart and select 1 or 0 only.\n");
    return 0;
}

if (has == 1 || ver == 1) {
    printf("\nWelcome to the ATM.\n");
    printf("Select account type:\n");
    printf("1. Savings\n2. Current\n");
    printf("Enter your choice: ");
    scanf("%d", &acc);

    if (acc == 1 || acc == 2) {
        printf("\nATM Menu:\n");
        printf("1. Balance Inquiry\n");
        printf("2. Withdraw Cash\n");
        printf("3. Deposit Money\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
```

```
        scanf("%d", &choice);

        if (choice == 1) {
            printf("Enter your 4-digit PIN: ");
            scanf("%d", &pin);

            if (pin >= 1000 && pin <= 9999) {
                printf("Your current balance is ₹%g\n", bal);
            } else {
                printf("Invalid PIN.\n");
            }
        } else if (choice == 2) {
            printf("Enter amount to withdraw: ₹");
            scanf("%f", &a);

            if (a <= 0) {
                printf("Invalid amount. Please enter a positive value.\n");
            } else {
                printf("Enter your 4-digit PIN: ");
                scanf("%d", &pin);

                if (pin >= 1000 && pin <= 9999) {
                    if (a > 20000)
                        printf("Maximum withdrawal limit is ₹20,000 per transaction.\n");
                    else if (a > bal)
                        printf("Insufficient balance.\n");
                    else if (acc == 2 && (bal - a) < 5000)
                        printf("You must maintain a minimum balance of ₹5,000 in a Current
account.\n");
                    else if ((int)a % 100 != 0)
```

```
                printf("Please enter amount in ₹100, ₹200, or ₹500 denominations.\n");
            else {
                bal -= a;
                printf("Please collect your cash: ₹%g\n", a);
                printf("Remaining balance: ₹%g\n", bal);
            }
        } else {
            printf("Invalid PIN.\n");
        }
    }
} else if (choice == 3) {
    printf("Enter amount to deposit: ₹");
    scanf("%f", &b);

    if (b <= 0) {
        printf("Invalid amount. Please enter a positive value.\n");
    } else {
        printf("Enter your 4-digit PIN: ");
        scanf("%d", &pin);

        if (pin >= 1000 && pin <= 9999) {
            if (b > 200000)
                printf("Maximum deposit allowed is ₹2,00,000 per transaction\n");
            else if ((int)b % 100 != 0)
                printf("Deposits must be in ₹100, ₹200, or ₹500 notes.\n");
            else {
                bal += b;
                printf("Amount ₹%g deposited successfully.\n", b);
                printf("Updated balance: ₹%g\n", bal);
```

```
                }
            } else {
                printf("Invalid pin.\n");
            }
        }
    } else if (choice == 4) {
        printf("Thank you for using the atm. Please collect your card.\n");
    } else {
        printf("Invalid option selected.\n");
    }
} else {
    printf("Authentication failed. Exiting...\n");
}

return 0;
}
```

8.2 PSEUDO CODE

PSEUDO CODE

```
START

DECLARE variables:
  has, acc, choice, pin as integer
  num as long integer
  bal as float = 50000.0
  a, b as float
  otp, ver as integer = 0

PRINT "Do you have a card? (1 for Yes / 0 for No):"
INPUT has

IF has == 0 THEN
  REPEAT up to 3 times:
    PRINT "Please enter your 16-digit account number:"
    INPUT num
    IF num is a valid 16-digit number THEN
      PRINT "OTP has been sent to your registered mobile number"
      BREAK
    ELSE
      PRINT "Invalid number. It must be exactly 16 digits."
  IF no valid number after 3 attempts THEN
    PRINT "Maximum attempts reached. Exiting..."
    STOP

  REPEAT up to 3 times:
```

ATM SIMULATION

```
    PRINT "Enter 6 digit OTP:"
    INPUT otp
    IF otp is a valid 6-digit number THEN
      ver = 1
      BREAK
    ELSE
      PRINT "Invalid OTP. Please try again."
  IF no valid OTP after 3 attempts THEN
    PRINT "Maximum OTP attempts reached. Exiting..."
    STOP

ELSE IF has == 1 THEN
  PRINT "Please insert your card"
  WAIT for card insertion input
  PRINT "Card inserted successfully."
ELSE
  PRINT "Invalid input. Please restart and select 1 or 0 only."
  STOP

IF has == 1 OR ver == 1 THEN
  PRINT "Welcome to the ATM."
  PRINT "Select account type:\n1. Savings\n2. Current"
  INPUT acc

  IF acc == 1 OR acc == 2 THEN
    PRINT ATM menu options:
      1. Balance Inquiry
      2. Withdraw Cash
      3. Deposit Money
```

```
    4. Exit
  INPUT choice


  IF choice == 1 THEN
    PRINT "Enter your 4-digit PIN:"
    INPUT pin
    IF pin is valid 4-digit THEN
      PRINT current balance bal
    ELSE
      PRINT "Invalid PIN."


  ELSE IF choice == 2 THEN
    PRINT "Enter amount to withdraw:"
    INPUT a
    PRINT "Enter your 4-digit PIN:"
    INPUT pin
    IF pin is valid 4-digit THEN
      IF a > 20000 THEN
        PRINT "Maximum withdrawal limit is 20,000 per transaction."
      ELSE IF a > bal THEN
        PRINT "Insufficient balance."
      ELSE IF acc == Current account AND bal - a < 5000 THEN
        PRINT "Minimum balance of 5,000 must be maintained in Current account."
      ELSE IF a not multiple of 100 THEN
        PRINT "Please enter amount in 100, 200, or 500 denominations."
      ELSE
        bal = bal - a
        PRINT "Please collect your cash:", a
        PRINT "Remaining balance:", bal
```

```
    ELSE
      PRINT "Invalid PIN."

  ELSE IF choice == 3 THEN
    PRINT "Enter amount to deposit:"
    INPUT b
    PRINT "Enter your 4-digit PIN:"
    INPUT pin
    IF pin is valid 4-digit THEN
      IF b > 200000 THEN
        PRINT "Maximum deposit allowed is 2,00,000 per transaction."
      ELSE IF b not multiple of 100 THEN
        PRINT "Deposits must be in 100, 200, or 500 notes."
      ELSE
        bal = bal + b
        PRINT "Amount deposited successfully:", b
        PRINT "Updated balance:", bal
    ELSE
      PRINT "Invalid PIN."

  ELSE IF choice == 4 THEN
    PRINT "Thank you for using the ATM. Please collect your card."
  ELSE
    PRINT "Invalid option selected."
 ELSE
  PRINT "Authentication failed. Exiting..."
```

ATM SIMULATION

END

8.3 SAMPLE OUTPUTS

SAMPLE OUTPUTS

1.

```
Do you have a card? (1 for Yes / 0 for No): 1

Please insert your card
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 1

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 1
Enter your 4-digit PIN: 2256
Your current balance is ₹50000
```

2.

```
Do you have a card? (1 for Yes / 0 for No): 0
Please enter your 16-digit account number:
4527862145243 25
Invalid number. It must be exactly 16 digits.
Enter your 16-digit account number again: 12343435468
Invalid number. It must be exactly 16 digits.
Enter your 16-digit account number again: 54321314334
Maximum attempts reached. Exiting...
```

3.

```
Do you have a card? (1 for Yes / 0 for No): 1

Please insert your card
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 1

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 2
Enter amount to withdraw: ₹30000
Enter your 4-digit PIN: 2256
Maximum withdrawal limit is ₹20,000 per transaction.
```

4.

```
Do you have a card? (1 for Yes / 0 for No): 1
Please insert your card
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 1

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 2
Enter amount to withdraw: ₹12354
Enter your 4-digit PIN: 2256
Please enter amount in ₹100, ₹200, or ₹500 denominations.
```

5.

```
Do you have a card? (1 for Yes / 0 for No): 1

Please insert your card
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 1

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 3
Enter amount to deposit: ₹2545
Enter your 4-digit PIN: 2256
Deposits must be in ₹100, ₹200, or ₹500 notes.
```

6.

```
Do you have a card? (1 for Yes / 0 for No): 0
Please enter your 16-digit account number: 5245621452645879
otp has been sent to your registered mobile number
Enter 6 digit OTP: 2345
Invalid OTP. Please try again.
Enter 6-digit OTP: 2159
Invalid OTP. Please try again.
Enter 6-digit OTP: 3154
Maximum OTP attempts reached. Exiting...
```

7.

```
Do you have a card? (1 for Yes / 0 for No): 1
Please insert your card.
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 2

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 4
Thank you for using the atm. Please collect your card.
```

8.

```
Do you have a card? (1 for Yes / 0 for No): 5
Invalid input. Please restart and select 1 or 0 only.
```

9.

```
Do you have a card? (1 for Yes / 0 for No): 5
Invalid input. Please restart and select 1 or 0 only.
```

10.

```
Do you have a card? (1 for Yes / 0 for No): 1
Please insert your card.
Card inserted successfully.

Welcome to the ATM.
Select account type:
1. Savings
2. Current
Enter your choice: 1

ATM Menu:
1. Balance Inquiry
2. Withdraw Cash
3. Deposit Money
4. Exit
Enter your choice: 2
Enter amount to withdraw: ₹-1000
Invalid amount. Please enter a positive value.
```

CHAPTER-9

CONCLUSION

Conclusion

The ATM Simulation Project successfully illustrates the application of decision-making logic in C to model a real-world banking system. Through this project, users gain a thorough understanding of input validation, error handling, and program flow management using only conditional statements. It emphasizes the importance of logical structuring in software development and demonstrates how complex operations can be handled efficiently without relying on loops or switch-case constructs.