ATM TRANSACTION SIMULATION

A Mini Project report submitted in partial fulfillment of the requirements for the course

C PROGRAMMING

Submitted by:

Name	Register no	Class
Shashmitha Sivakumar	732925ITR097	25IT1B
Ramya V	732925ITR081	25IT1B
Sonasri G	732925ITR098	25IT1B
Sathyarooba S	732925ITR093	25IT1B
Sameera	732925ITR091	25IT1B
Tejasri	732925ITR117	25IT1B

Submitted to:

Maheshwari R

DEPARTMENT OF INFORMATION TECHNOLOGY

VELALAR COLLEGE OF ENGINEERING AND TECHNOLOGY

ACADEMIC YEAR:2025-2026

PROBLEM STATEMENT

In real-world banking systems, ATM machines are used to perform basic financial transactions such as account creation, cash deposit, withdrawal, and balance enquiry. Managing these operations manually is inefficient and error-prone.

The problem is to design and develop a simple ATM Transaction Simulation system using C programming that allows users to perform basic banking operations in a secure and structured manner. The system should store account and transaction details permanently using file handling.

The system also incorporates basic security using PIN authentication and maintains transaction history with date and time to simulate real ATM operations

ALGORITHM

STEP 1: Start the program

STEP 2: Display ATM menu

STEP 3: Read user choice

STEP 4: If choice = 1

Create a new account and store details in file

STEP 5: If choice = 2

Deposit amount and update balance

STEP 6: If choice = 3

Withdraw amount after checking balance

STEP 7: If choice = 4

Display account balance

STEP 8: If choice = 5

Display recent transactions (mini statement)

STEP 9: If choice = 0

Exit the program

STEP 10: Repeat steps 2–9 until exit

STEP 11: Stop the program.

C SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

/* Structure to store account details */
struct Account {
    int accNo;
    char name[50];
    int pin;
    float balance;
};

/* Function declarations */
void createAccount();
void deposit();
void withdraw();
void balanceEnquiry();
void miniStatement();

int main() {
    int choice;

    do {
```

```
        printf("\n===== ATM TRANSACTION SIMULATION
=====\n");
        printf("1. Create Account\n");
        printf("2. Deposit\n");
        printf("3. Withdraw\n");
        printf("4. Balance Enquiry\n");
        printf("5. Mini Statement\n");
        printf("0. Exit\n");

printf("=====================================\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1: createAccount(); break;
            case 2: deposit(); break;
            case 3: withdraw(); break;
            case 4: balanceEnquiry(); break;
            case 5: miniStatement(); break;
            case 0: printf("Thank you for using ATM!\n"); break;
            default: printf("Invalid choice!\n");
        }
    } while(choice != 0);

    return 0;
}
```

```c
/* Create new account */
void createAccount() {
    struct Account a;
    FILE *fp = fopen("accounts.txt", "ab");

    if(fp == NULL) {
        printf("File error!\n");
        return;
    }

    printf("Enter Account Number: ");
    scanf("%d", &a.accNo);

    getchar(); // clear buffer
    printf("Enter Name: ");
    fgets(a.name, sizeof(a.name), stdin);
    a.name[strcspn(a.name, "\n")] = '\0';

    do {
        printf("Set 4-digit PIN: ");
        scanf("%d", &a.pin);
    } while(a.pin < 1000 || a.pin > 9999);

    do {
        printf("Enter Initial Deposit: ");
        scanf("%f", &a.balance);
```

```
    } while(a.balance < 0);

    fwrite(&a, sizeof(a), 1, fp);
    fclose(fp);

    printf("Account created successfully!\n");
}

/* Deposit amount */
void deposit() {
    struct Account a;
    int acc, enteredPin, found = 0;
    float amt;
    time_t t;

    FILE *fp = fopen("accounts.txt", "rb");
    FILE *temp = fopen("temp.txt", "wb");

    if(fp == NULL || temp == NULL) {
        printf("File error!\n");
        return;
    }

    printf("Enter Account Number: ");
    scanf("%d", &acc);
```

```
    printf("Enter Amount to Deposit: ");
    scanf("%f", &amt);

    if(amt <= 0) {
        printf("Invalid amount!\n");
        return;
    }

    while(fread(&a, sizeof(a), 1, fp)) {
        if(a.accNo == acc) {
            printf("Enter PIN: ");
            scanf("%d", &enteredPin);

            if(enteredPin == a.pin) {
                a.balance += amt;
                time(&t);

                FILE *tr = fopen("transactions.txt", "a");
                if(tr != NULL) {
                    fprintf(tr, "%d Deposit %.2f %s", acc, amt,
ctime(&t));
                    fclose(tr);
                }

                printf("Deposit successful!\n");
                printf("New Balance: %.2f\n", a.balance);
            } else {
```

```
            printf("Incorrect PIN!\n");
        }
        found = 1;
    }
    fwrite(&a, sizeof(a), 1, temp);
}

fclose(fp);
fclose(temp);
remove("accounts.txt");
rename("temp.txt", "accounts.txt");

if(!found)
    printf("Account not found!\n");
}

/* Withdraw amount */
void withdraw() {
    struct Account a;
    int acc, enteredPin, found = 0;
    float amt;
    time_t t;

    FILE *fp = fopen("accounts.txt", "rb");
    FILE *temp = fopen("temp.txt", "wb");
```

```
if(fp == NULL || temp == NULL) {
    printf("File error!\n");
    return;
}

printf("Enter Account Number: ");
scanf("%d", &acc);

printf("Enter Amount to Withdraw: ");
scanf("%f", &amt);

if(amt <= 0) {
    printf("Invalid amount!\n");
    return;
}

while(fread(&a, sizeof(a), 1, fp)) {
    if(a.accNo == acc) {
        printf("Enter PIN: ");
        scanf("%d", &enteredPin);

        if(enteredPin == a.pin) {
            if(a.balance >= amt) {
                a.balance -= amt;
                time(&t);
```

```
                FILE *tr = fopen("transactions.txt", "a");
                if(tr != NULL) {
                    fprintf(tr, "%d Withdraw %.2f %s", acc, amt,
ctime(&t));
                    fclose(tr);
                }

                printf("Withdrawal successful!\n");
                printf("New Balance: %.2f\n", a.balance);
            } else {
                printf("Insufficient balance!\n");
            }
        } else {
            printf("Incorrect PIN!\n");
        }
        found = 1;
    }
    fwrite(&a, sizeof(a), 1, temp);
}

fclose(fp);
fclose(temp);
remove("accounts.txt");
rename("temp.txt", "accounts.txt");

if(!found)
    printf("Account not found!\n");
```

```
}

/* Balance enquiry */
void balanceEnquiry() {
    struct Account a;
    int acc, enteredPin, found = 0;
    FILE *fp = fopen("accounts.txt", "rb");

    if(fp == NULL) {
        printf("File error!\n");
        return;
    }

    printf("Enter Account Number: ");
    scanf("%d", &acc);

    while(fread(&a, sizeof(a), 1, fp)) {
        if(a.accNo == acc) {
            printf("Enter PIN: ");
            scanf("%d", &enteredPin);

            if(enteredPin == a.pin) {
                printf("\nAccount No: %d", a.accNo);
                printf("\nName      : %s", a.name);
                printf("\nBalance   : %.2f\n", a.balance);
            } else {
```

```
            printf("Incorrect PIN!\n");
        }
        found = 1;
        break;
    }
}

fclose(fp);

if(!found)
    printf("Account not found!\n");
}

/* Mini statement */
void miniStatement() {
    int acc, accNo;
    char line[200];
    int found = 0;

    FILE *fp = fopen("transactions.txt", "r");

    if(fp == NULL) {
        printf("No transactions available!\n");
        return;
    }
```

```
    printf("Enter Account Number: ");
    scanf("%d", &acc);

    printf("\n--- Mini Statement ---\n");
    while(fgets(line, sizeof(line), fp)) {
        sscanf(line, "%d", &accNo);
        if(accNo == acc) {
            printf("%s", line);
            found = 1;
        }
    }

    fclose(fp);

    if(!found)
        printf("No transactions found for this account!\n");
}
```

SAMPLE INPUT AND OUTPUT

```
===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
0. Exit
=======================================
Enter your choice: 1
Enter Account Number: 0112345678
Enter Name: John
Set 4-digit PIN: 1308
Enter Initial Deposit: 20000
Account created successfully!

===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
```

0. Exit

==========

Enter your choice: 1

Enter Account Number: 0118765432

Enter Name: Jennie

Set 4-digit PIN: 1407

Enter Initial Deposit: 30000

Account created successfully!

===== ATM TRANSACTION SIMULATION =====

1. Create Account

2. Deposit

3. Withdraw

4. Balance Enquiry

5. Mini Statement

0. Exit

==========

Enter your choice: 1

Enter Account Number: 0119876543

Enter Name: Peter

Set 4-digit PIN: 1108

Enter Initial Deposit: 50000

Account created successfully!

```
===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
0. Exit
=======================================
Enter your choice: 2
Enter Account Number: 0112345678
Enter Amount to Deposit: 2450
Enter PIN: 1308
Deposit successful!
New Balance: 22450.00

===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
0. Exit
=======================================
Enter your choice: 3
```

Enter Account Number: 0118765432
Enter Amount to Withdraw: 3240
Enter PIN: 1407
Withdrawal successful!
New Balance: 26760.00

===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
0. Exit
==========
Enter your choice: 4
Enter Account Number: 0119876543
Enter PIN: 1108

Account No: 119876543
Name : Peter
Balance : 50000.00

===== ATM TRANSACTION SIMULATION =====
1. Create Account

2. Deposit

3. Withdraw

4. Balance Enquiry

5. Mini Statement

0. Exit

==========

Enter your choice: 5

Enter Account Number: 0118765432

--- Mini Statement ---

118765432 Withdraw 3240.00 Thu Dec 25 19:54:05 2025

118765432 Withdraw 3240.00 Thu Dec 25 19:54:09 2025

===== ATM TRANSACTION SIMULATION =====

1. Create Account

2. Deposit

3. Withdraw

4. Balance Enquiry

5. Mini Statement

0. Exit

==========

Enter your choice: 5

Enter Account Number: 0112345678

--- Mini Statement ---
112345678 Deposit 2450.00 Thu Dec 25 19:53:38 2025
112345678 Deposit 2450.00 Thu Dec 25 19:53:41 2025

===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry
5. Mini Statement
0. Exit
==
Enter your choice: 5
Enter Account Number: 0119876543

--- Mini Statement ---
No transactions found for this account!

===== ATM TRANSACTION SIMULATION =====
1. Create Account
2. Deposit
3. Withdraw
4. Balance Enquiry

5. Mini Statement

0. Exit

==========

Enter your choice: 0

Thank you for using ATM!

LEARNING OUTCOME

After completing the ATM Transaction Simulation project, the learner will be able to:

- Understand ATM Working Principles

Gain a clear understanding of how ATM systems perform basic banking operations such as PIN verification, balance enquiry, withdrawal, and deposit.

- Apply Programming Concepts

Use core programming concepts like variables, data types, conditional statements, loops, and functions in a real-world application.

- Implement Decision Making

Learn how to use logical conditions to validate user input, check account balance, and control transaction flow.

- Improve Problem-Solving Skills

Analyze real-life problems and convert them into step-by-step program logic.

- Handle User Input and Output

Develop skills in taking user input, displaying meaningful messages, and creating a user-friendly interaction.

- Understand Security Basics

Learn the importance of PIN authentication and secure handling of financial transactions.

- Error Handling Awareness

Identify and handle common errors such as invalid PINs and insufficient balance.

- Build Confidence in Project Development

Gain confidence in designing, coding, and explaining a complete mini-project suitable for academic evaluation.

These learning outcomes clearly show the knowledge and skills gained from the ATM Transaction Simulation project.

Enhancements Implemented:

- PIN-based authentication for secure access
- Transaction logging with date and time

CONCLUSION

The ATM Transaction Simulation project clearly explains how an ATM system works in real life. In this project, basic ATM functions such as card verification (PIN check), balance enquiry, cash withdrawal, and cash deposit are simulated using a simple program.

The project shows how user input is processed and how conditions are used to allow or deny transactions. It also ensures that withdrawals are made only when sufficient balance is available, which helps avoid errors. This improves understanding of security, accuracy, and control in banking transactions.

Overall, this project is useful for learning programming concepts and understanding banking operations. It gives a clear idea of how automated systems handle financial transactions. The project can be further improved by adding features like transaction history, database storage, and a graphical interface in the future.

Future Enhancement (GUI)

The current system is console-based. In the future, this project can be enhanced by implementing a graphical user interface (GUI) to improve user interaction and usability, making it closer to a real ATM system.