



VIT[®]
BHOPAL

VIT Bhopal University

**School of Computer Science and
Engineering(SCOPE)**

Mini Project Report on

“Mini ATM System using Python”

Submitted by:

NACHIKET PRASHANT GADGE

Registration No: 25BCE10852

Slot: B11+B12+B13

ABSTRACT: -

This project, Mini ATM System using Python, demonstrates the essential operations of an Automated Teller Machine. The system allows users to create accounts, log in using a PIN, deposit or withdraw funds, check balances, and view transaction history.

The project focuses on simplicity and clarity so beginners can understand how banking systems work internally. It uses modular Python code split into multiple .py files, making the project clean and maintainable. This mini-project helps students learn programming logic, problem-solving, and system flow through a real-world example.

INTRODUCTION: -

Banking systems are part of our daily lives, and an ATM (Automated Teller Machine) is one of the most commonly used banking services. Understanding the internal workflow of an ATM is important for computer science students.

This project explains core ATM functionalities through Python programming. It includes modules for account management, ATM operations, and transaction tracking. The system is designed to be simple, educational, and easy to execute on any computer.

PROBLEM STATEMENT: -

Many students find real banking systems too complex to understand. This project solves that problem by creating a simplified ATM system that demonstrates:

Account creation

Secure login using PIN

Basic banking functions

Balance updates

Transaction history

The goal is to show the flow of operations that happen inside an ATM in a beginner-friendly way.

OBJECTIVES: -

1. To simulate ATM operations using Python.
2. To understand modular programming by separating logic into multiple files.
3. To implement basic banking features such as deposit, withdraw, and balance inquiry.
4. To provide visual output that can be easily tested and verified.
5. To help students develop problem-solving and code-organization skills.

SYSTEM ARCHITECTURE (SIMPLE): -

The system contains 4 main modules:

main.py → Controls the entire flow

atm.py → Handles login & ATM menu

account.py → Manages account details

transaction.py → Records and displays transactions

These files work together to simulate an ATM environment.

USE CASE DIAGRAM EXPLANATION: -

Actors:

User

Use Cases:

Create Account

Login

Deposit

Withdraw

Check Balance

View Transaction History

Logout

The user interacts with the ATM system to perform basic banking tasks.

SEQUENCE DIAGRAM EXPLANATION: -

1. User runs main.py
2. Main program shows welcome screen
3. User logs in
4. ATM verifies account and PIN
5. User selects an option
6. ATM triggers the respective function
7. Account updates balance
8. Transaction module stores the history
9. Updated balance is shown

WORKFLOW DIAGRAM EXPLANATION: -

1. Start
2. Display main menu
3. Create account or Login
4. If login successful → Show ATM menu
5. Choose operation (deposit, withdraw, check balance, history)
6. Perform operation

7. Ask user if they want to continue

8. Logout

9. End

CLASS DIAGRAM EXPLANATION: -

Classes:

Account

Attributes: name, account number, pin, balance

Methods: deposit (), withdraw(), check balance(), change pin()

ATM

Methods: login (), show menu (), process choice ()

Transaction

Methods: add transaction (), view history ()

METHODOLOGY: -

1. Understanding ATM operations

2. Designing simple class structure
3. Splitting logic into separate Python modules
4. Using lists to store accounts and transactions
5. Implementing menu-driven navigation
6. Testing every module individually
7. Integrating all modules through main.py

MODULES: -

1. account.py

Stores user information

Handles deposit, withdraw, and balance operations

2. atm_menu.py

Validates login

Displays ATM menu

Directs user to the selected ATM action

3. transaction.py

Records each transaction

Stores timestamps

Displays history

4. Mainstream.py

Entry point

Runs the ATM system

Shows welcome screen and options

DATA FLOW (EXPLANATION): -

1. User inputs → main.py

2. Login → atm.py

3. Account data → account.py

4. Transaction data → transaction.py

5. Output displayed back to the user

```
File Edit Selection View Go Run ...
CSEPROJECT
Mini-Bank-ATM-
  _pycache_
  account.py
  atm_menu.py
  bank.py
  Mainstream.py
  README.md1
  transaction.py

Mainstream.py
1 from bank import create_account, login
2 from atm_menu import atm_menu
3
4
5 def main():
6     while True:
7         print("\n===== MINI BANK ATM =====")
8         print("1. Create Account")
9         print("2. Login")
10        print("3. Exit")
11
12        choice = input("Choose an option: ")

TERMINAL
===== MINI BANK ATM =====
1. Create Account
2. Login
3. Exit
Choose an option: 1

--- Create New Account ---
Enter new account number: 2708529421
Enter your name: NACHIKET
Set a 4-digit PIN: 0852
Account created successfully!

===== MINI BANK ATM =====
1. Create Account
2. Login
3. Exit
Choose an option: []
```

```
File Edit Selection View Go Run ...
CSEPROJECT
Mini-Bank-ATM-
  _pycache_
  account.py
  atm_menu.py
  bank.py
  Mainstream.py
  README.md1
  transaction.py

Mainstream.py
1 from bank import create_account, login
2 from atm_menu import atm_menu
3
4
5 def main():
6     while True:
7         print("\n===== MINI BANK ATM =====")
8         print("1. Create Account")
9         print("2. Login")
10        print("3. Exit")
11
12        choice = input("Choose an option: ")

TERMINAL
2. Deposit
3. Withdraw
4. Mini Statement
5. Change PIN
6. Logout
Choose an option: 2
Enter amount to deposit: 2400000
Deposit successful!

--- ATM Menu ---
1. Check Balance
2. Deposit
3. Withdraw
4. Mini Statement
5. Change PIN
6. Logout
Choose an option: 1
Your balance is: ₹2400000

--- ATM Menu ---
1. Check Balance
```

```
File Edit Selection View Go Run ...
CSEPROJECT
Mini-Bank-ATM-
  _pycache_
  account.py
  atm_menu.py
  bank.py
  Mainstream.py
  README.md1
  transaction.py

Mainstream.py
2 from bank import create_account, login
3 from atm_menu import atm_menu
4
5 def main():
6     while True:
7         print("\n===== MINI BANK ATM =====")
8         print("1. Create Account")
9         print("2. Login")
10        print("3. Exit")
11
12        choice = input("Choose an option: ")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\PRASHANT\Desktop\CSEPROJECT> & C:/Users/PRASHANT/AppData/Local/Programs/Python/Python313/python.exe c:/Users/PRASHANT/Desktop/CSEPROJECT/Mini-Bank-ATM-/Mainstream.py

===== MINI BANK ATM =====
1. Create Account
2. Login
3. Exit
Choose an option: 2

--- Login ---
Enter account number: 2708529421
Enter PIN: 8852
Invalid account number or PIN.

===== MINI BANK ATM =====
1. Create Account
2. Login
3. Exit
Choose an option: 
```

```
File Edit Selection View Go Run ...
CSEPROJECT
Mini-Bank-ATM-
  _pycache_
  account.py
  atm_menu.py
  bank.py
  Mainstream.py
  README.md1
  transaction.py

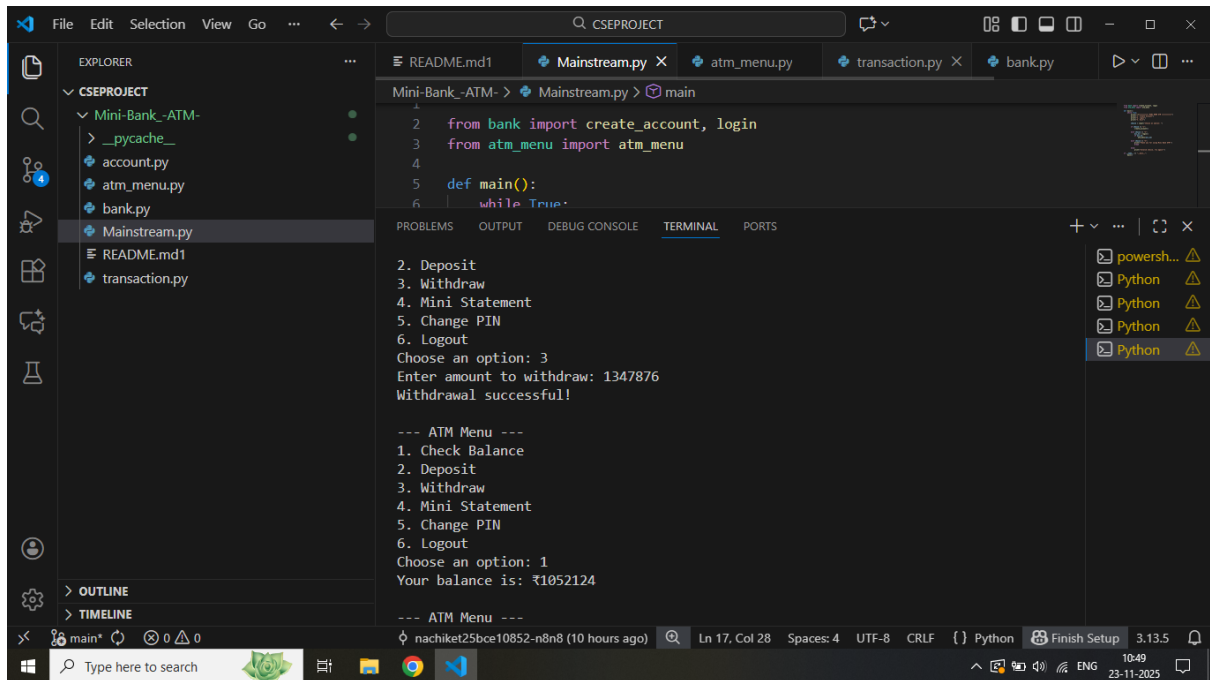
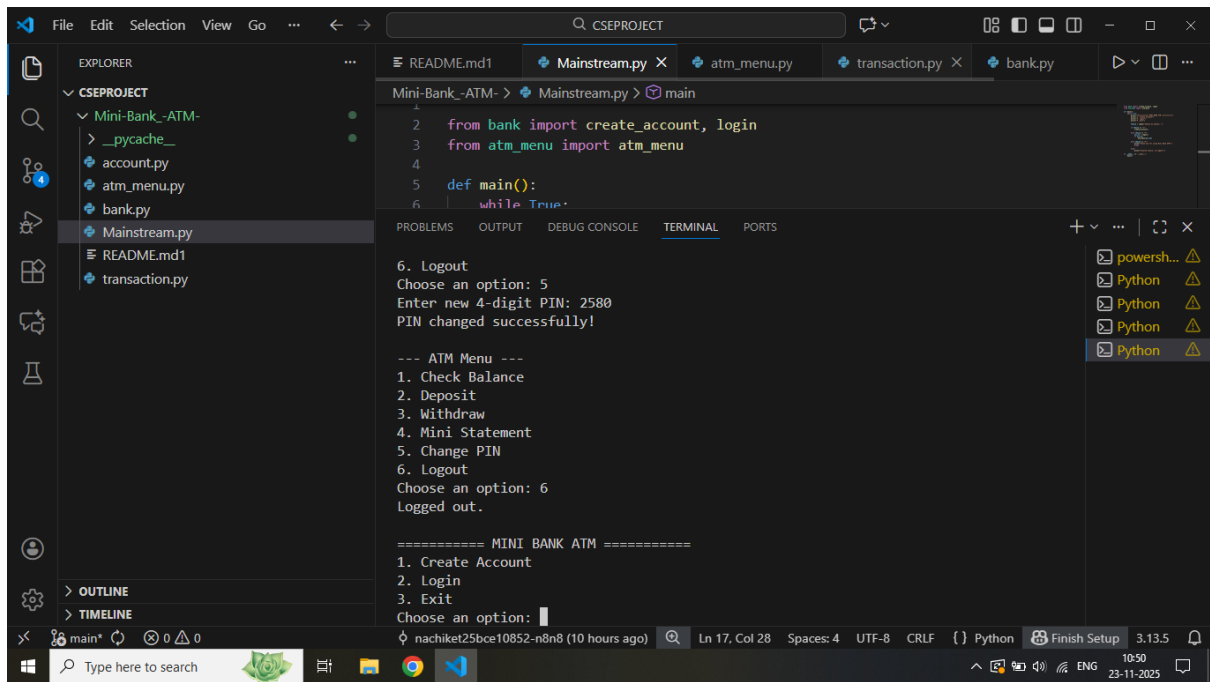
Mainstream.py
2 from bank import create_account, login
3 from atm_menu import atm_menu
4
5 def main():
6     while True:
7         print("\n===== MINI BANK ATM =====")
8         print("1. Create Account")
9         print("2. Login")
10        print("3. Exit")
11
12        choice = input("Choose an option: ")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Account created successfully!

===== MINI BANK ATM =====
1. Create Account
2. Login
3. Exit
Choose an option: 2

--- Login ---
Enter account number: 2708529421
Enter PIN: 8852
Welcome, NACHIKET!

--- ATM Menu ---
1. Check Balance
2. Deposit
3. Withdraw
4. Mini Statement
5. Change PIN
6. Logout
Choose an option: 
```



OUTPUT: -

The system successfully performs:

Account creation

Login authentication

Balance updates

Safe withdrawal

Storing transaction history

Clean menu-based operations

FUTURE SCOPE: -

Adding real database support

ATM card simulation

User-friendly GUI

Online banking integration

Advanced security features

CONCLUSION: -

This Mini ATM System demonstrates the fundamentals of Python programming through a practical banking application. It improves understanding of modular code, problem-solving, and system design. The project meets all academic requirements for a mini-project and provides a strong foundation for future learning.

REFERENCES: -

Python documentation

Class lecture notes

Basic OOP concepts

Vityarthi modules