PROJECT REPORT
Console-Based Secure ATM Banking System

Submitted By: PRADEEP NAIN

REG No: 25BAI11576

Branch : CSE AI/ML

1. Introduction

An Automated Teller Machine allows customers to perform banking transactions without going to the bank.

This is a console-based Python ATM Banking System where users can log in using their card number and PIN. withdraw cash and deposit cash, perform fund transfers, balance inquiries, and change PINs. It also includes an admin module for account management, like adding and deleting users and unlocking accounts.

2. Problem Statement

Most of the available educational ATM examples are rather limited and do not secure login, multi-user support, transaction tracking. PIN attempts or admin controls, for example. The aim is to create a real-life simulation of ATM banking using JSON for storage. multiple users, and modular programming.

3.Objectives

• To simulate real ATM banking using Python.

• To enable login securely using a card number and PIN.

• To enable customers to carry out banking operations using a menu.

• To permanently store all account data.

• To incorporate an administration system to manage accounts.

4. Functional Requirements

• Customer Login using card number and PIN

• Withdraw cash

• Deposit money

• Check balance

• Change PIN

• Transfer money to another account

• Close own account

• Admin login

• Admin can add user

• Admin can delete user

• Admin can unlock user

4. Functional Requirements

- Customer Login using card number and PIN
- Withdraw money
- Deposit money
- Check balance
- Change PIN
- Transfer money to another account
- Close own account
- Admin login
- Admin can add user
- Admin can delete user
- Admin can unlock user

5. Non-Functional Requirements

- Data stored securely in JSON file
- PIN encryption for security
- User-friendly menu system
- Fast read/write operations
- Lock account after 3 wrong PIN attempts
- Modular and maintainable Python architecture

6. System Architecture

```
User (Customer/Admin)
        ↓
main.py (User Interface)
        ↓
Account / Admin Classes (Application Layer)
        ↓
Storage.py (Data Access Layer)
        ↓
accounts.json & admin.json (Data Storage)
```

7. UML Design Diagrams

Use Case Diagram

```
      +----------------------+
      |     ATM System       |
      +----------------------+
     /        |         \
    /         |          \
[Customer]   [Admin]       (System)

Customer:
  - Login
  - Withdraw Money
  - Deposit Money
  - Check Balance
  - Change PIN
  - Transfer Money
  - Close Account
  - Logout

Admin:
  - Admin Login
  - Add User
  - Delete User
  - Unlock User
```

Class Diagram

Admin: add_user(), delete_user(), unlock_user()

Account: withdraw(), deposit(), transfer_money(), change_pin(), close_account()

Storage: load_accounts(), save_accounts(), load_admin(), save_admin()

8. Implementation Details

Language: Python

Storage: JSON file (accounts.json / admin.json)

Program Structure: Multiple Python files (main.py, account.py, admin.py, storage.py)

Input: Console-based keyboard input

Output: Console display messages and menus

9. Testing Results

Test Case: Wrong PIN 3 times → Account locks ✅

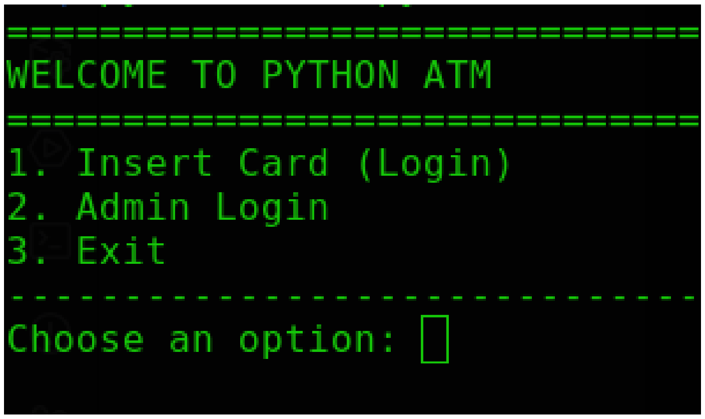
Test Case: Withdraw more than balance → Shows error ✅

Test Case: Deposit money → Balance updated ✅

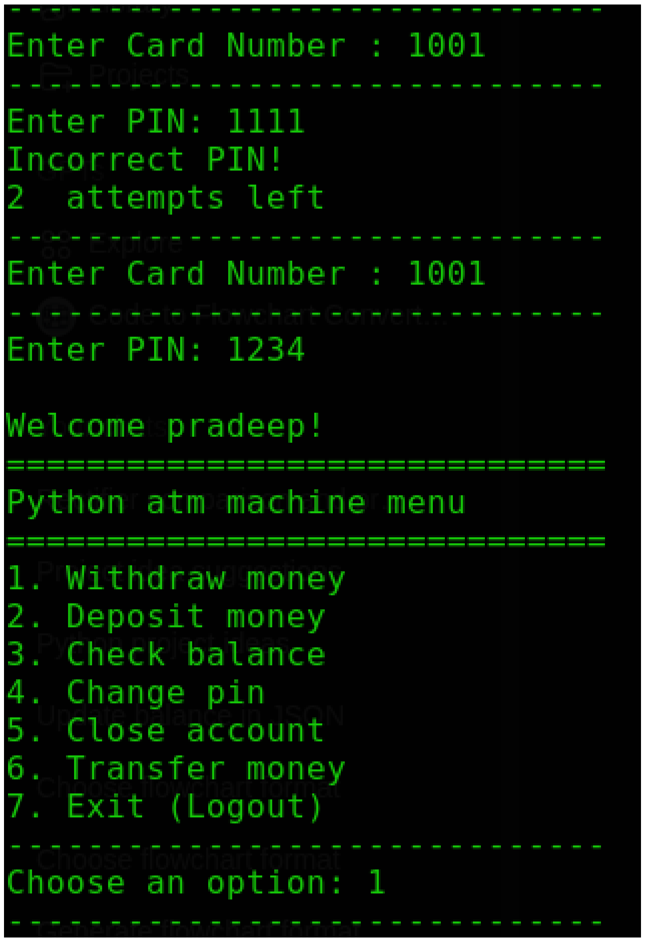
Test Case: Transfer → Both accounts updated ✅

Test Case: Admin deletes user → User login disabled ✅

10.Screenshots

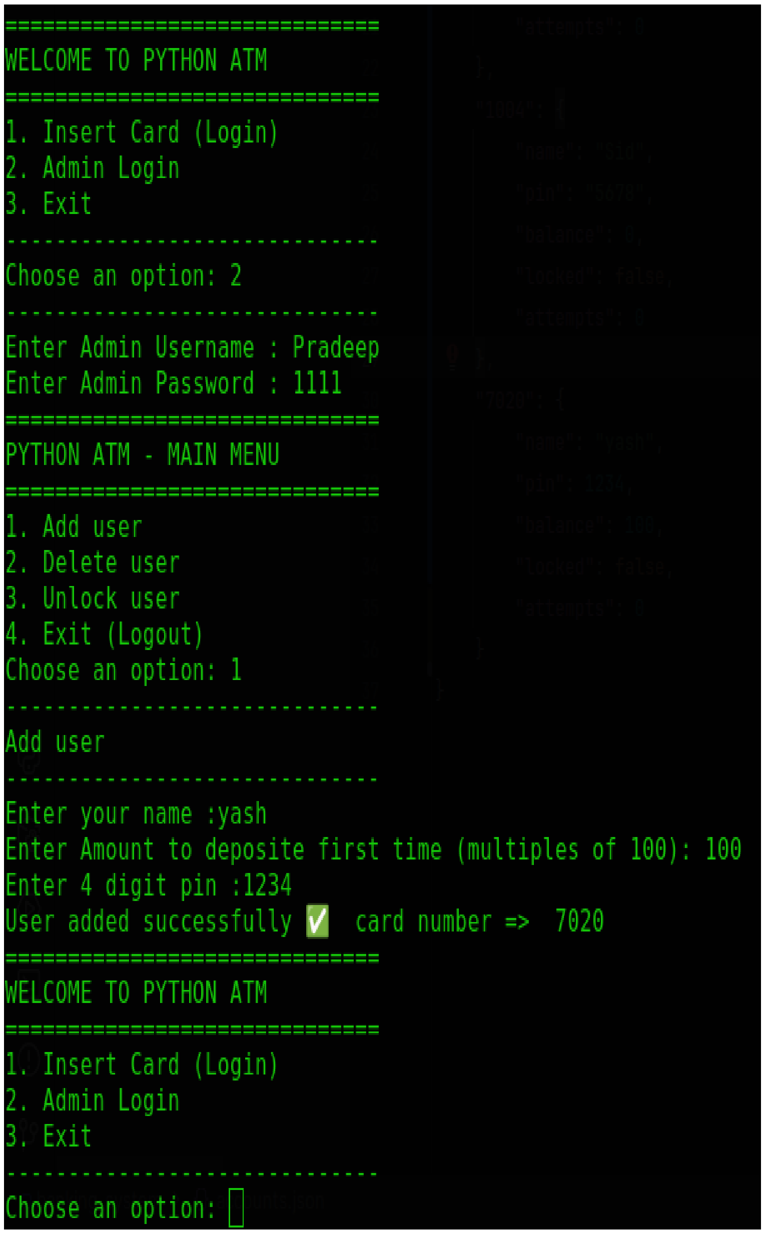


```
=================================
WELCOME TO PYTHON ATM
=================================
1. Insert Card (Login)
2. Admin Login
3. Exit
---------------------------------
Choose an option: 
```



```
--------------------------------
Enter Card Number : 1001
--------------------------------
Enter PIN: 1111
Incorrect PIN!
2  attempts left
--------------------------------
Enter Card Number : 1001
--------------------------------
Enter PIN: 1234

Welcome pradeep!
================================
Python atm machine menu
================================
1. Withdraw money
2. Deposit money
3. Check balance
4. Change pin
5. Close account
6. Transfer money
7. Exit (Logout)
--------------------------------
Choose an option: 1
--------------------------------
```



```
==============================
WELCOME TO PYTHON ATM
==============================
1. Insert Card (Login)
2. Admin Login
3. Exit
------------------------------
Choose an option: 2
------------------------------
Enter Admin Username : Pradeep
Enter Admin Password : 1111
==============================
PYTHON ATM - MAIN MENU
==============================
1. Add user
2. Delete user
3. Unlock user
4. Exit (Logout)
Choose an option: 1
------------------------------
Add user
------------------------------
Enter your name :yash
Enter Amount to deposite first time (multiples of 100): 100
Enter 4 digit pin :1234
User added successfully ✅  card number =>  7020
==============================
WELCOME TO PYTHON ATM
==============================
1. Insert Card (Login)
2. Admin Login
3. Exit
------------------------------
Choose an option: 
```

11. Challenges Faced

- Handling invalid user input
- Designing JSON structure for multiple users
- Implementing secure PIN verification logic
- Managing multiple menus and modules

12. Learning Outcomes

- Learned file handling and JSON storage
- Understood modular programming in Python
- Implemented authentication and error handling
- Gained knowledge of real banking workflow

13. Conclusion

The Secure ATM Banking System successfully simulates banking operations through a console-based application.

It enables safe, fast, and realistic ATM usage and demonstrates modular programming, secure authentication,

and persistent storage using JSON.

14. References

- Python Documentation – json library
- Stack Overflow (error-handling examples)
- W3Schools Python Tutorials