ATM Interface Simulation

Presented by Rupesh Saini (24CSU259)

Maulik Chopra (24CSU302)

Monish yadav (24CSU321)

Project Overview: Building a Robust ATM System

Ourprojectdeliversaconsole-basedATMsystemdesignedforreliabilityandeaseofuse,leveragingfundamentalprogramming concepts.

Console-Based

An intuitive text-based interface for user interaction.

Secure Login

Robust authentication for user accounts and transactions.

Persistent Storage

Account balances and transaction history are safely stored.

Core Features: Essential ATM Functionality

ThisATMsimulation includesallcriticaloperationsexpectedfromareal-worldmachine, ensuring a comprehensive user experience.

PIN Verification

Secure personal identification number (PIN) entry and validation to protect user accounts.

Deposit & Withdraw

Seamless processing of cash deposits into accounts and withdrawals from available balances.

File I/O: Ensuring Data Persistence

Data Stored in CSV Format

Account details and transaction records are efficiently managed in Comma Separated Values (CSV) files.

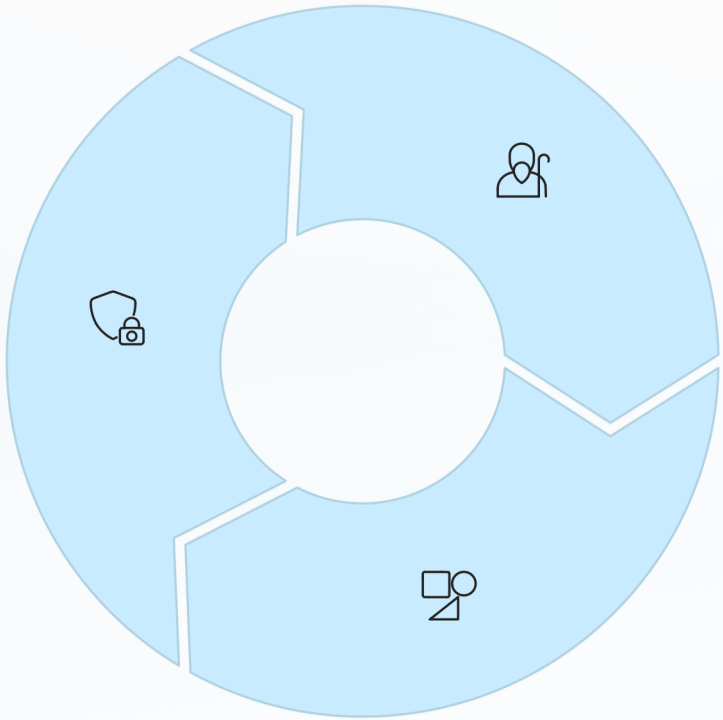
Survives Session Closures

Data is written to disk immediately, ensuring that no information is lost even if the application is closed unexpectedly.

OOP Concepts: Building a Structured Foundation

Encapsulation

Secures data by bundling it with the methods that operate on it, restricting direct access.



Inheritance

Enables classes to inherit properties and promoting code reusability and a logical hierarchy.

Polymorphism

Allows objects to take on many forms, facilitating flexible and adaptable transaction handling.

System Architecture: Modular and Scalable

Multiple P ackages

Organizes related classes into distinct packages for better project structure.

Modular Classes

Each class has a single, well-defined responsibility, making the system easier to understand and maintain.

Object Passing

Objects are passed between different layers, facilitating interaction and data flow.

Flow of Program: A User's Journey

.

01

1. Start Application

Initiates the ATM program and loads necessary data.

02

2. User Login

Prompts for credentials; validates PIN against stored records.

03

3. Display Main Menu

Presents options like deposit, withdraw, mini-statement, and exit.

04

4. Process Transaction

Executes user-selected operations, updating account balances.

05

5. Save to File

Writes updated account data and transaction history to CSV files.

06

6. Exit

Gracefully terminates the program, with all data persistently stored.

Why This Project Stands Out

Real-Life B ehavior

Accurately simulates the core functionalities and user experience of an actual ATM.

Clean OOP Design

Demonstrates best practices in object-oriented programming for maintainabilit y.

Reliable & Extendable

Built for stability with a structure that easily accommodates future enhancements.

Thank You