

**School of Engineering & Technology**

**Computer Science Department**

A Project report on

“ATM WITH CHATBOT INTERFACE USING GUI”

in ‘Python’ Language

**Submitted By:**

Dhyanesh Sharma (23CSU100)

Viraj Yadav (23CSU346)

Vedant Malhotra (23CSU339)

**Section:** AI & ML Sec - D

**Date**: 25 November 2024

GitHub Link-

<https://github.com/Malhotra7890/Atm_with_chatbotinterface>

**Acknowledgment**

We extend our sincere gratitude to our teacher Shraddha Arora ma’am, for providing us with this opportunity and guiding us every step of the way. Her support and mentorship have been invaluable to the success of our Python programming project.

**Index**

1. **Introduction**
   * **Overview**
   * **Purpose**
   * **Scope**
2. **Project Architecture**
   * **System Overview**
3. **Project Implementation**
   * **Details Validation**
   * **User Input Handling**
   * **Basic Operations**
   * **File Handling**
4. **Project Components**

* **Details Validation**
* **User Input Handling**
* **Transaction History and File Handling (JSON)**
* **GUI Implementation (Tkinter)**
* **Chatbot Interaction**

1. **Test Cases**
2. **Tools Used**
3. **Conclusion**
4. **Bibliography**
5. **Introduction**
   * Overview

This ATM interface project is designed using Python's Tkinter library to provide a graphical user interface (GUI). It allows users to log in, perform transactions such as deposits and withdrawals, view transaction history, and interact with a simple chatbot for customer service. The project implements persistent data storage for user accounts using JSON files and transaction history in text files.

* + Purpose

The purpose of this project is to create a basic ATM simulation where users can securely log in, check their balance, deposit or withdraw money, view transaction history, and get assistance via a chatbot.

* + Scope

This project simulates an ATM system with basic features like login validation, balance checking, transaction history, and simple chatbot interaction. data persistence through JSON files and text-based transaction logs. The system is built without external databases, making it lightweight and portable.

1. **Project Architecture**
   * System Overview

The project is a Tkinter-based GUI that allows users to perform various operations after logging in. It includes features like balance check, withdraw and deposit options, transaction history, and a chatbot for assistance. The system simulates an ATM interface, processing transactions ,updating balance information in a local session  and JSON files for persistent account data storage.

1. **Project Implementation**

* **Details Validation**
* User details, including name, account number, and PIN, are checked against a JSON file (account\_database.json).
* Validation ensures the account exists and the PIN matches. Invalid inputs prompt error messages.
* **User Input Handling**
* Inputs are captured through Tkinter’s entry widgets.
* Numeric values for transactions are validated to ensure they are positive and within available balance limits.
* **Basic Operations**
* Withdrawal: Validates the entered amount and updates the user's account balance. Logs the transaction in a user-specific text file.
* Deposit: Ensures the deposited amount is valid and updates the account balance. Records the transaction in a user-specific file.
* Balance Inquiry: Displays the current account balance to the user.
* Transaction History: Users can view their past transactions stored in a text file associated with their account.
* Chatbot Interaction: Provides predefined responses to user queries related to ATM operations and general assistance.
* Sign Out: Logs the user out of the system and returns to the login screen for security.
* **File Handling**

1. **JSON Files**:
   * Store user credentials and account balances, updated after each transaction.
2. **Text Files**:
   * Transaction history is logged in user-specific files ({account\_number}\_history.txt).
   * These files track deposits, withdrawals, and updated balances.
3. **Project Components**
   * **Login and Validation System**

The login system ensures that only valid users can access the ATM. It checks the account name, number, and PIN against the predefined data in the account\_database.

* + **Withdraw and Deposit Features**

Users can withdraw or deposit money from their account. The system ensures that sufficient balance is available for withdrawals, and it updates the account balance accordingly after each transaction.

* + **Transaction History and File Handling**

The transaction history feature allows users to view a list of all previous deposits and withdrawals. The history is stored in text files, where each transaction is logged with the date and time. The system uses a JSON file (account\_database.json) to store user details, including their account numbers, PINs, and balances.When the user logs in, the entered details (name, account number, and PIN) are validated by comparing them against the data stored in this JSON file.

* + **Sign-Out Option**

Users can sign out from the ATM interface, which logs them out and brings them back to the login screen.

* + **Chatbot Interface**

The chatbot provides users with instant assistance. Users can ask predefined questions like "How can I apply for a credit card?" or "What is the interest rate on savings accounts?" The bot responds with relevant information.

* + **Graphical User Interface (GUI)**

Tkinter is used to design the user interface, allowing users to interact with the ATM system in an easy-to-use manner. The GUI consists of frames for login, transaction options, withdrawal and deposit screens, and a chatbot.

1. **Test Cases**
   * **Valid Login:**

A screenshot of a computer screen

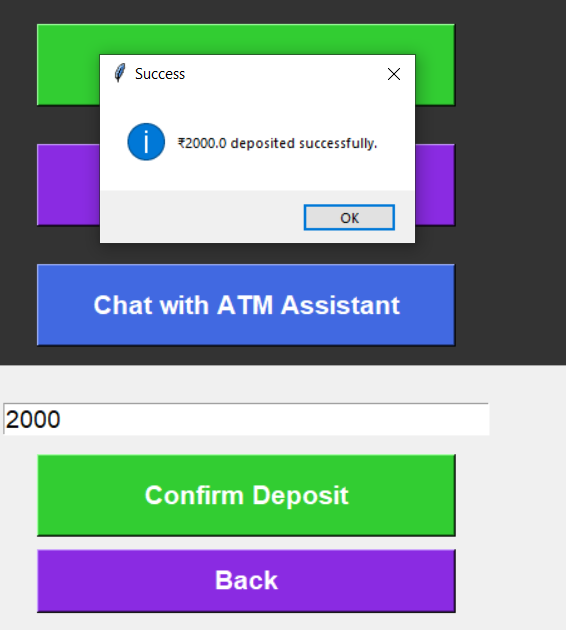
Description automatically generated

* + **Invalid Login:**

A screenshot of a computer screen

Description automatically generated

* + **Deposit:.**



* + **Withdraw:**

A screenshot of a computer

Description automatically generated

* + **Check Balance:**

A screenshot of a computer screen

Description automatically generated

* + **Transaction History:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

* + **Chatbot Interaction:**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

* + **Sign – Out:**

A screenshot of a computer

Description automatically generated

1. **Tools Used** 
   * Tkinter (Graphical User Interface):

Tkinter is used for building the user interface (UI). It allows the creation of interactive windows and widgets such as buttons, entry fields, and labels, which are essential for the ATM system's operation.

* + Python (Programming Language):

Python is the main programming language used to implement the logic behind the ATM system. It handles user input, transaction processing, data validation, and file handling.

* + File Handling (Text Files):

Data related to transactions (deposits and withdrawals) is stored in text files, simulating how an ATM would maintain a record of user activities. The files are stored locally with each user’s account number as the filename.

* + Date and Time Handling:

Python’s datetime module is used to record timestamps for each transaction, ensuring that all activities are logged with the exact date and time.

1. **Conclusion**

This ATM simulation project provides a simple, functional interface that mimics real-world ATM operations, such as login, balance checking, withdrawals, deposits, transaction history, and a basic chatbot assistant. It demonstrates the use of Tkinter for creating GUI-based applications, as well as  Data persistence through JSON files which enhances reliability, while text-based transaction history ensures transparency.

1. **Bibliography**
   * "Tkinter Documentation." Python.org. <https://docs.python.org/3/library/tkinter.html>
   * "Python File Handling." W3Schools. <https://www.w3schools.com/python/python_file_handling.asp>
   * "Python DateTime Module." W3Schools. https://www.w3schools.com/python/python\_datetime.asp