Secure Bank Management System Using Python and AI-Based Iris Recognition

Mr.M.Rama Raju[1], Ganiya Naaz[2], K.Pradeep Reddy[3], K.Uma Devi[4]
[1]Assistant Professor, Department of CSE
[2,3,4,] UG Students, Department of CSE

mrraju808 @gmail.com, ganiyanaaz@gmail.com, karlapradeepreddy6@gmail.com, kondaboinaumadevi@2gmail.com

Christu Jyothi Institute of Technology & Science, Jangaon, Telangana, India

Abstract: This project presents a Python-based desktop application that simulates core banking functionalities without relying on a traditional database system. Utilizing the Tkinter library for graphical user interface (GUI) development and text files for data storage, the system provides a lightweight and offline solution for managing bank accounts. It supports essential operations such as account creation, deposits, withdrawals, fund transfers, balance inquiries, and transaction history management.Through structured file handling and object-oriented programming principles, the system ensures persistent data storage and organized account management. The intuitive interface guides users through each task with built-in input validation, promoting error-free interactions and enhancing usability for individuals with minimal technical background. This user-centric design makes the application particularly suited for educational purposes and small-scale banking simulations.The project showcases Python's versatility in developing full-featured desktop applications by integrating GUI design with backend logic using file-based storage. It demonstrates practical applications of software engineering concepts such as modularity, data validation, and interactive system design. This system serves as a foundational model for building more complex, database-integrated banking solutions in the future.

Keywords*:* Bank Management System ,Python Programming , Tkinter GUI File Handling, Offline Banking Application, Desktop Application Development, Object-Oriented Programming,Transaction History Tracking ,Account Management , Secure Data Storage,Financial Software Simulation ,Educational Banking Model

1. INTRODUCTION

In today's digital era, banking systems must be efficient, accurate, and secure to manage increasing volumes of customers and transactions. Traditional methods such as paper-based records or outdated software often lead to inefficiencies, data loss, and operational delays. To overcome these challenges, a Bank Management System (BMS) developed using Python offers a practical, offline solution for simulating and managing core banking operations.The proposed system simplifies tasks like customer registration, account creation, deposits, withdrawals, fund transfers, and balance inquiries. Built with Python and Tkinter for the graphical interface, the application uses text files for data storage, ensuring ease of access without relying on databases. The interface is user-friendly and supports built-in validation to reduce errors, prevent fraud, and automate repetitive processes. This project not only demonstrates the capabilities of Python in building real-world applications but also serves as a useful educational tool for learning file handling, object-oriented programming, and GUI design.

2. LITERATURE SURVEY

- MD. Faizan (2012) discusses how ICT drives global competition and highlights the use of Service-Oriented Architecture (SOA) in enhancing service scalability and reliability. The paper cites case studies from Scandinavian and Swiss banks, demonstrating how SOA enables greater organizational agility and competitiveness.

- MD. Aquil Amwar (2012) discusses transaction issues in banking systems, focusing on how failures can be avoided and fixed. It highlights that weak legal enforcement increases default risks, particularly in banks with high past losses. The study emphasizes the importance of security in banking systems, proposing security questions for fraud prevention. Additionally, it uses the Technology Acceptance Model to show thatperceived service quality, credibility, and risk influence customer satisfaction and continued usage.
- Tran Duc Loi's paper discusses creating desktop applications using Python and TkInter, focusing on UI development, packaging, and deployment across platforms. It highlights Python's ease of use and TkInter's cross-platform capabilities for desktop apps.

- Python iS a Nutshell" (3rd Edition) by Alex Martelli, Anna Ravenscroft, and Steve Holden is a concise reference guide, offering quick access to key Python concepts, libraries, and practical examples for efficient programming.

3. PROPOSED SYSTEM

The proposed **Bank Management System** is a Python-based desktop application with a Tkinter graphical user interface. It automates key banking operations such as account creation, deposits, withdrawals, balance checks, and fund transfers. The system stores data using text files, ensuring easy access and lightweight storage without the need for a complex database. It offers a simple and intuitive interface, making it user-friendly for both customers and staff. Transaction history and account modifications are also supported for transparency and flexibility. By reducing manual work and errors, the system improves efficiency and accuracy. This solution is ideal for small-scale banking simulations or educational purposes.

MODULES USED

- **1. Account Management Module**
- Handles account creation with details like name, address, phone, and initial deposit
- Validates inputs (e.g., 6-digit account number, 10-digit phone number)
- Stores account data in a text file for persistence
- **2. Transaction Module**
- Supports deposit, withdrawal, and balance inquiry operations
- Updates balance and logs each transaction with a timestamp
- Ensures input validation and error handling for each transaction
- **3. Transfer Module**
- Enables fund transfer between two accounts
- Validates sufficient balance and existence of both accounts
- Records the transaction for future reference
- **4. Account Modification & Deletion Module**
- Allows users to update account details such as name, address, and phone
- Deletes an account after confirmation and updates the file storage

- **5. Transaction History Module**
- Reads and displays past transactions from a log file
- Filters transactions based on account number
- Helps in reviewing and auditing account activities
- **6. GUI Module (Tkinter Interface)**
- Provides a user-friendly interface using Tkinter
- Contains buttons for each action like deposit, withdraw, create account, etc.
- Uses pop-up dialogs for input and feedback messages

TECHNOLOGIES USED

Programming Language: Python

Framework: Tkinter (for GUI development)

Tools: PyCharm, Visual Studio Code

Database: File handling (using .txt files for data storage)

Operating System: Windows 10

Frontend: Tkinter (UI components like buttons, text entries, popups)

SYSTEM ADVANTAGES

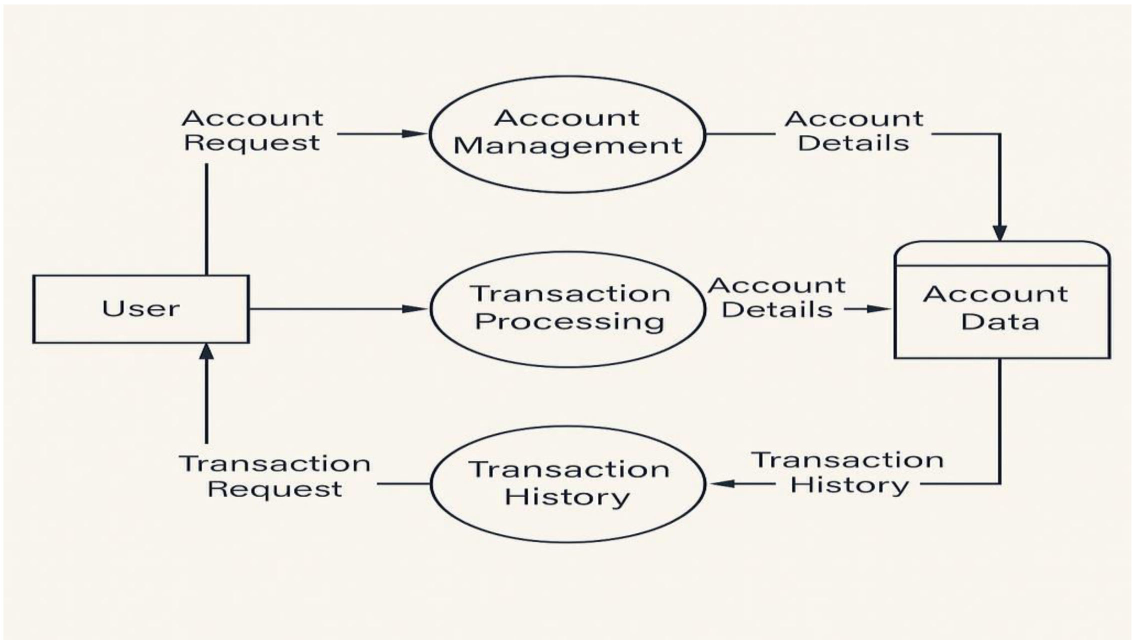
- Simplifies banking operations through automation of account management and transactions
- User-friendly Tkinter-based graphical interface for easy interaction
- Accurate and real-time transaction processing with balance updates
- Transaction history logging for easy auditing and transparency Lightweight file-based data storage system, no need for complex database setup
- Enhanced security with input validation for account and transaction details

Advantages Of Proposed System

- Efficient Account Management: Automates account creation, updates, and deletions, reducing manual workload and errors.
- User-Friendly Interface: A simple and intuitive Tkinter-based GUI ensures easy navigation for both customers and staff.
- Real-Time Transaction Processing: Supports deposits, withdrawals, and transfers with immediate balance updates and transaction logging.

4. ARCHITECTURE

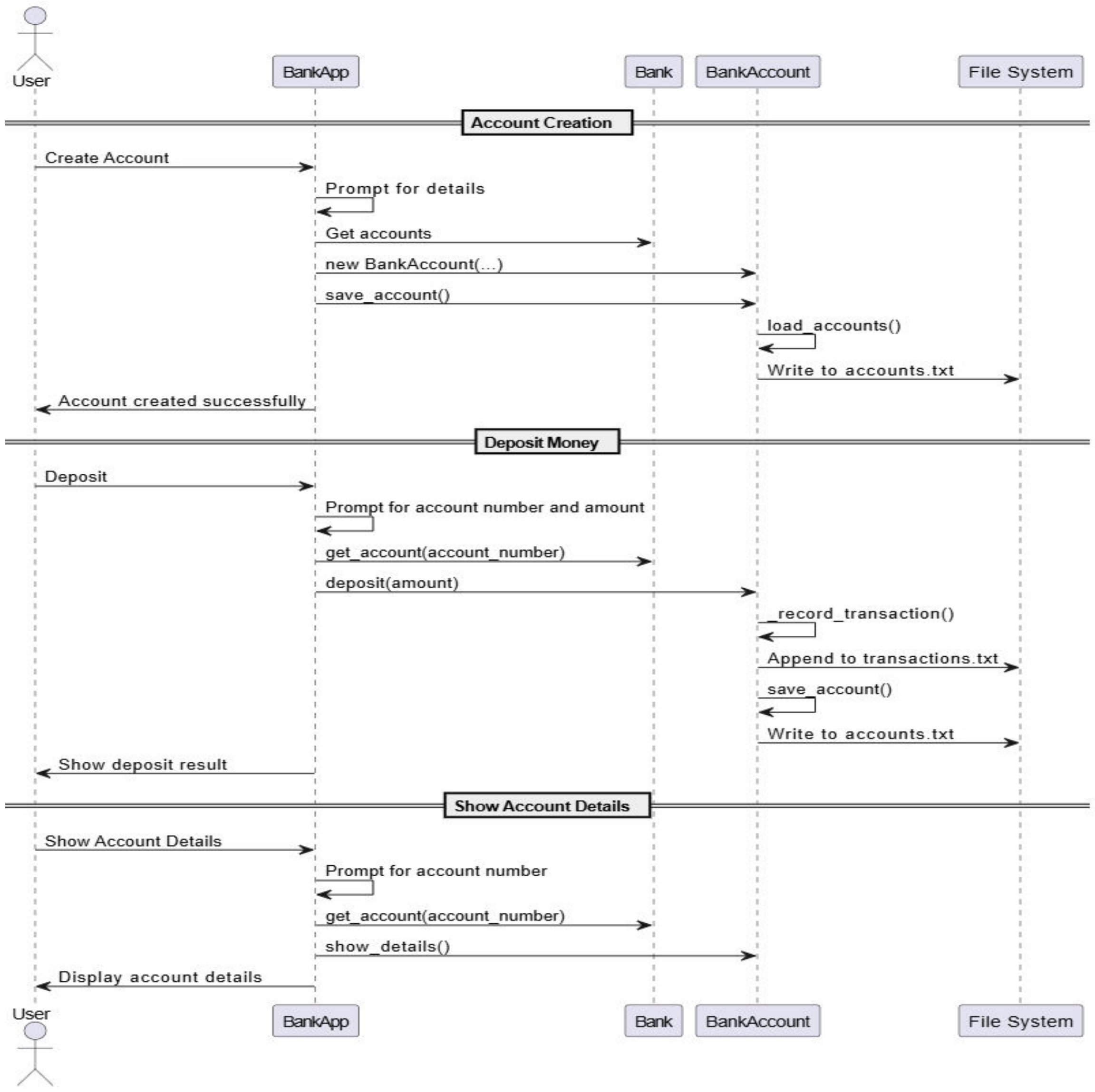
The system architecture consists of three main modules: **User Module**, **Transaction Module**, and **Account Module**. The **User Module** allows users to register, view, and modify their account details. The **Transaction Module** handles deposits, withdrawals, transfers, and transaction history, ensuring real-time updates of balances. The **Account Module** manages the storage of account data and transaction logs in text files. The system's user interface, built using **Tkinter**, connects these modules seamlessly, offering a simple and intuitive experience for both users and bank staff. This architecture ensures smoothbanking operations and efficient data management.



Data flow diagram

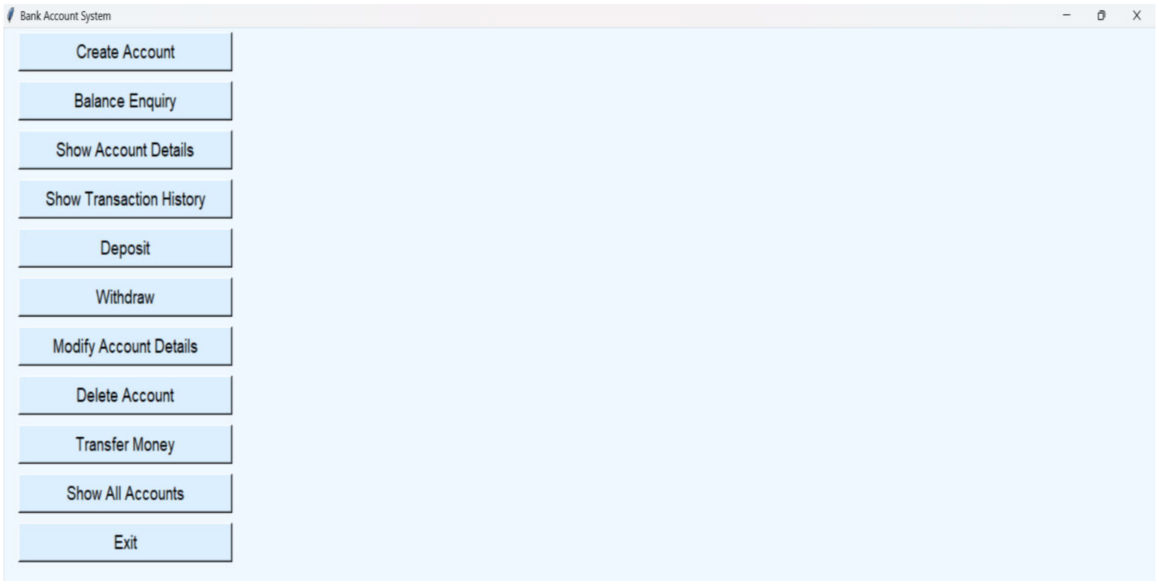
SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) can be an affordable interaction diagram that shows but processes operate with one another and in what order and at a specific time. it is a construct of a Message Sequence Chart. Sequence diagrams area unit usually called event diagrams, event scenarios, and temporal arrangement diagrams

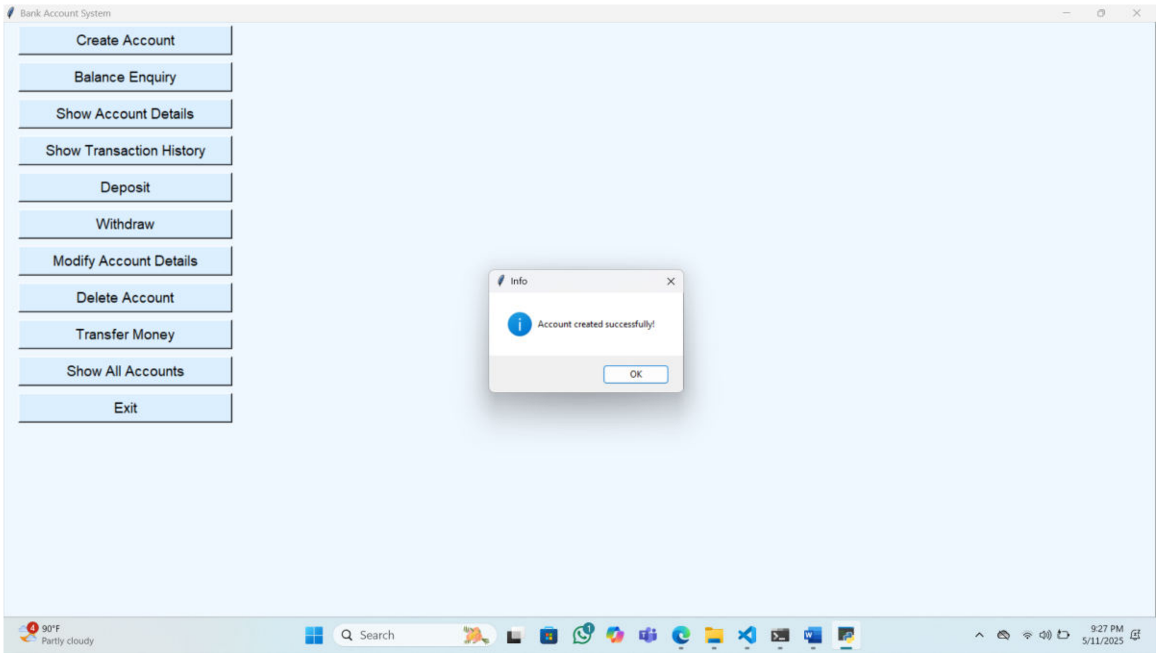


5. OUTPUT SCREENS

The system features the following UI screens

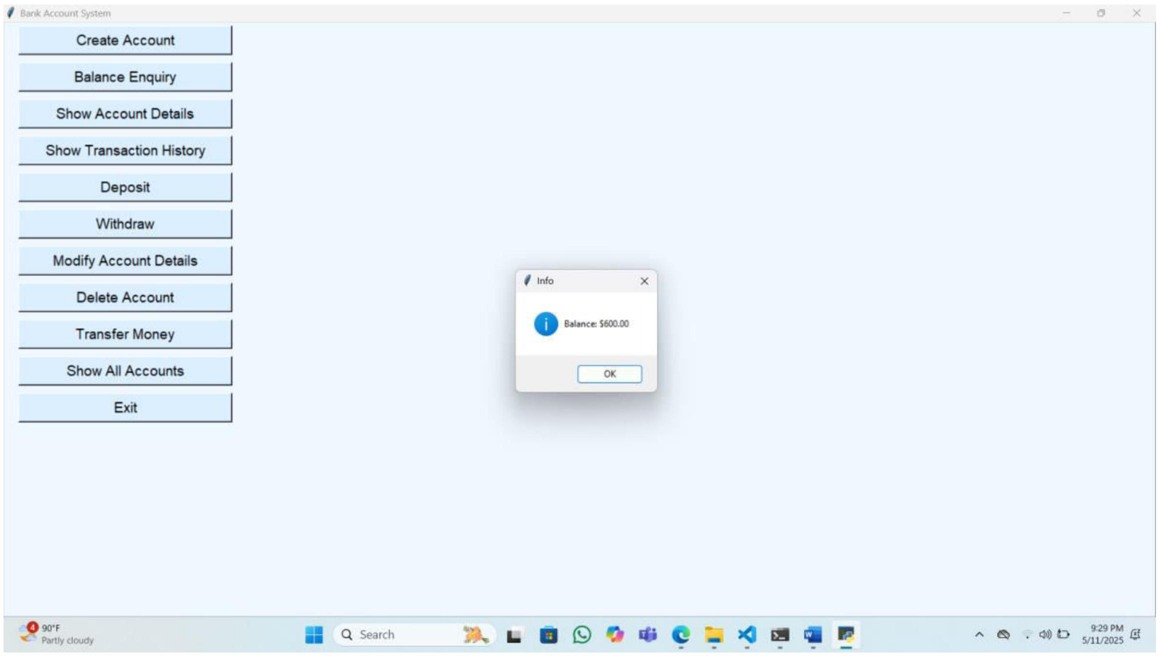


• Home Page



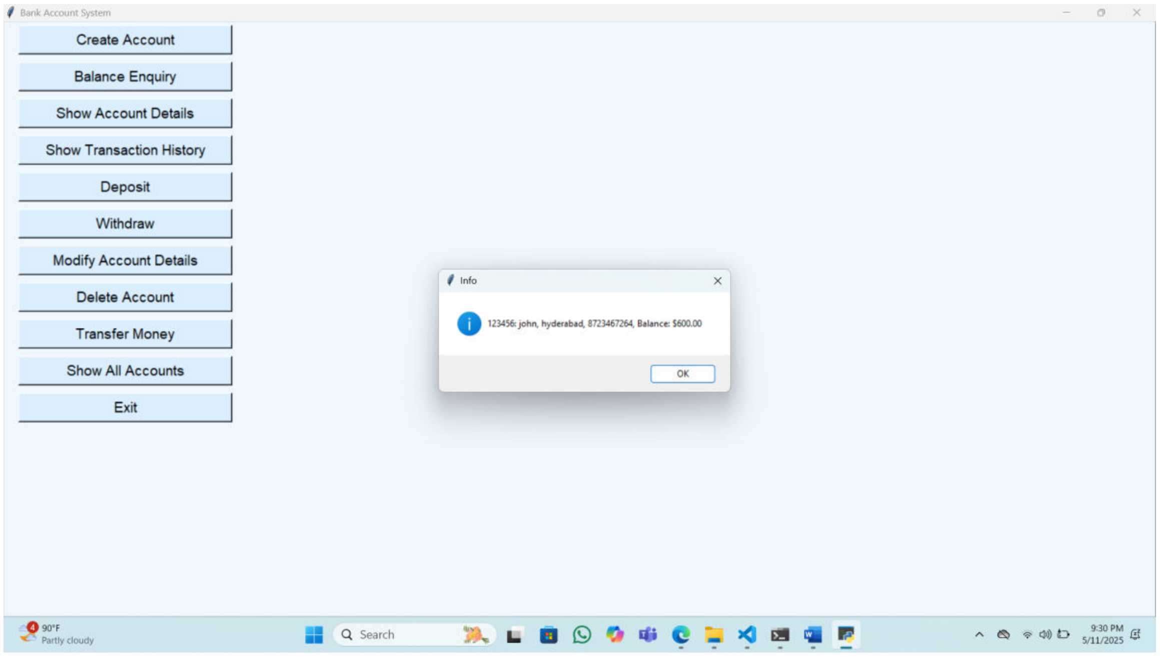
Account created successfully!

• creating account page



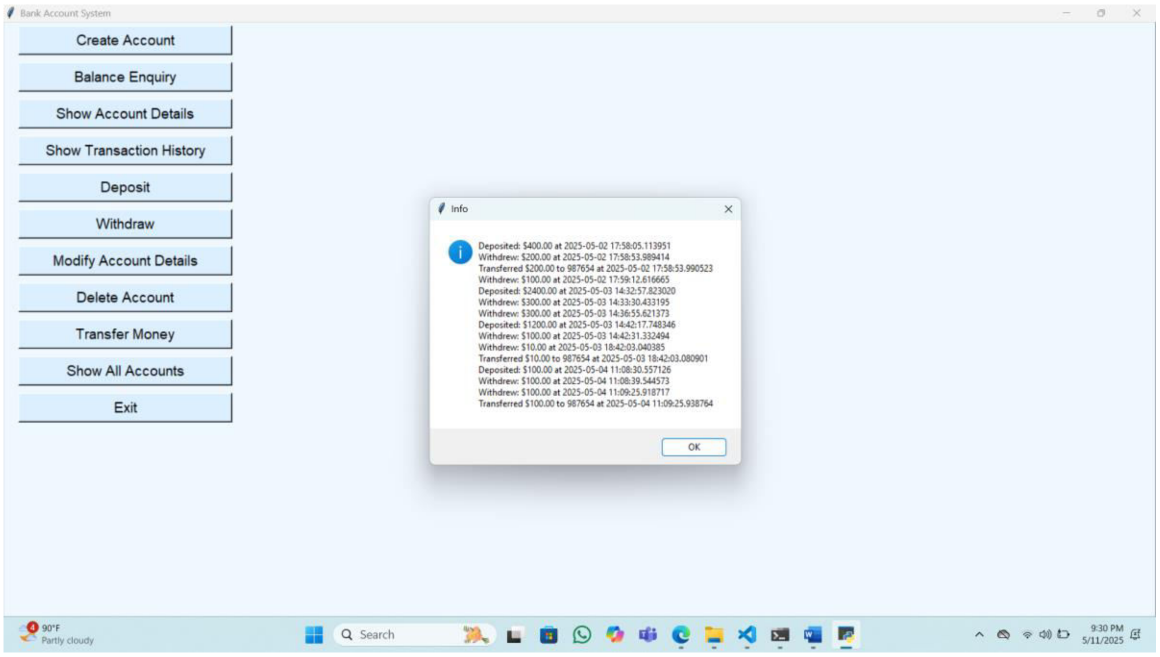
Balance: $600.00

• Balance enquiry page



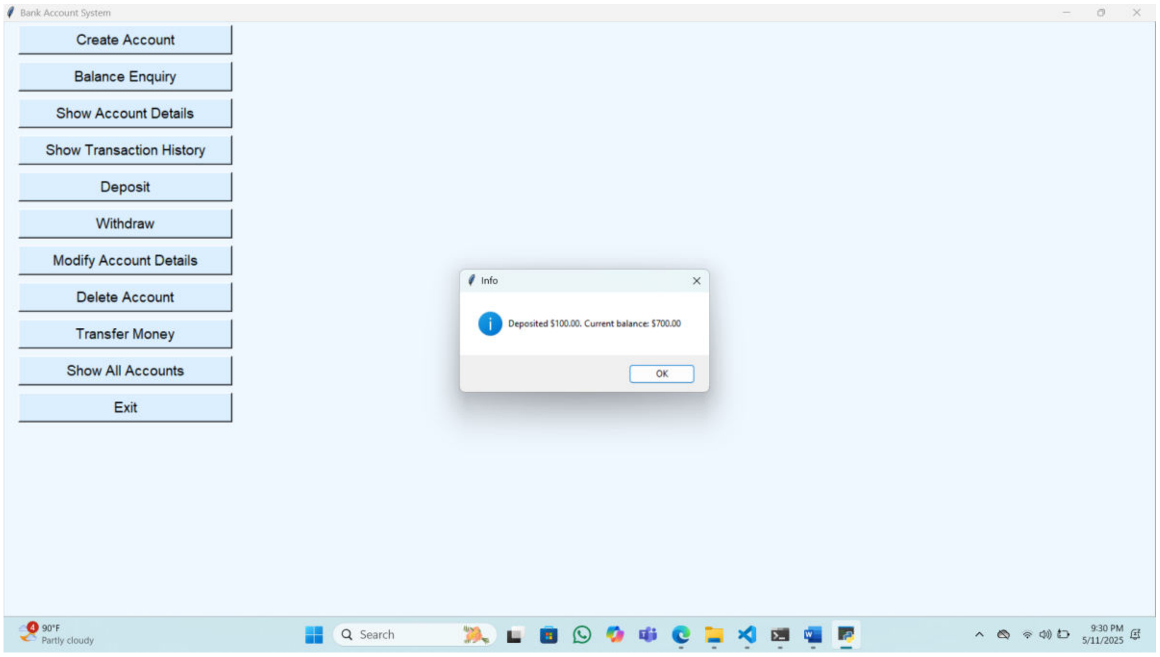
123456: john, hyderabad, 8723467264, Balance: $600.00

• Displaying account details page



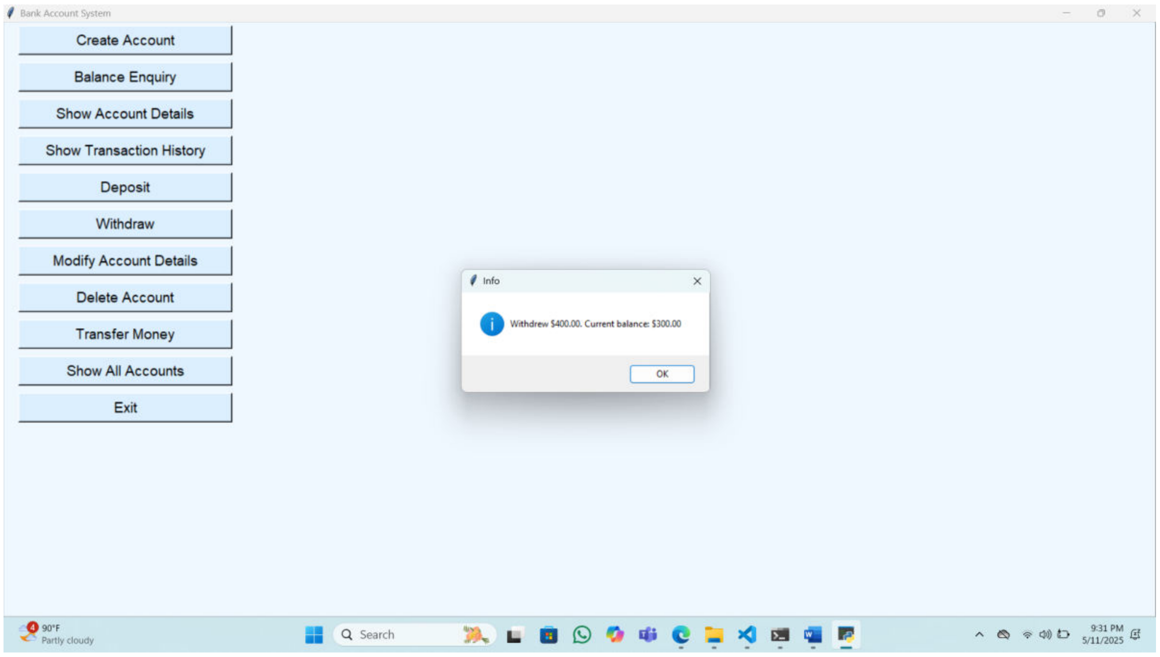
```
Deposited: $400.00 at 2025-05-02 17:58:05.113951
Withdrew: $200.00 at 2025-05-02 17:58:53.989414
Transferred $200.00 to 987654 at 2025-05-02 17:58:53.990523
Withdrew: $100.00 at 2025-05-02 17:59:12.616665
Deposited: $2400.00 at 2025-05-03 14:32:57.823020
Withdrew: $300.00 at 2025-05-03 14:33:30.433195
Withdrew: $300.00 at 2025-05-03 14:36:55.621373
Deposited: $1200.00 at 2025-05-03 14:42:17.748346
Withdrew: $100.00 at 2025-05-03 14:42:31.332494
Withdrew: $10.00 at 2025-05-03 18:42:03.040385
Transferred $10.00 to 987654 at 2025-05-03 18:42:03.080901
Deposited: $100.00 at 2025-05-04 11:08:30.557126
Withdrew: $100.00 at 2025-05-04 11:08:39.544573
Withdrew: $100.00 at 2025-05-04 11:09:25.918717
Transferred $100.00 to 987654 at 2025-05-04 11:09:25.938764
```

• Transaction history page



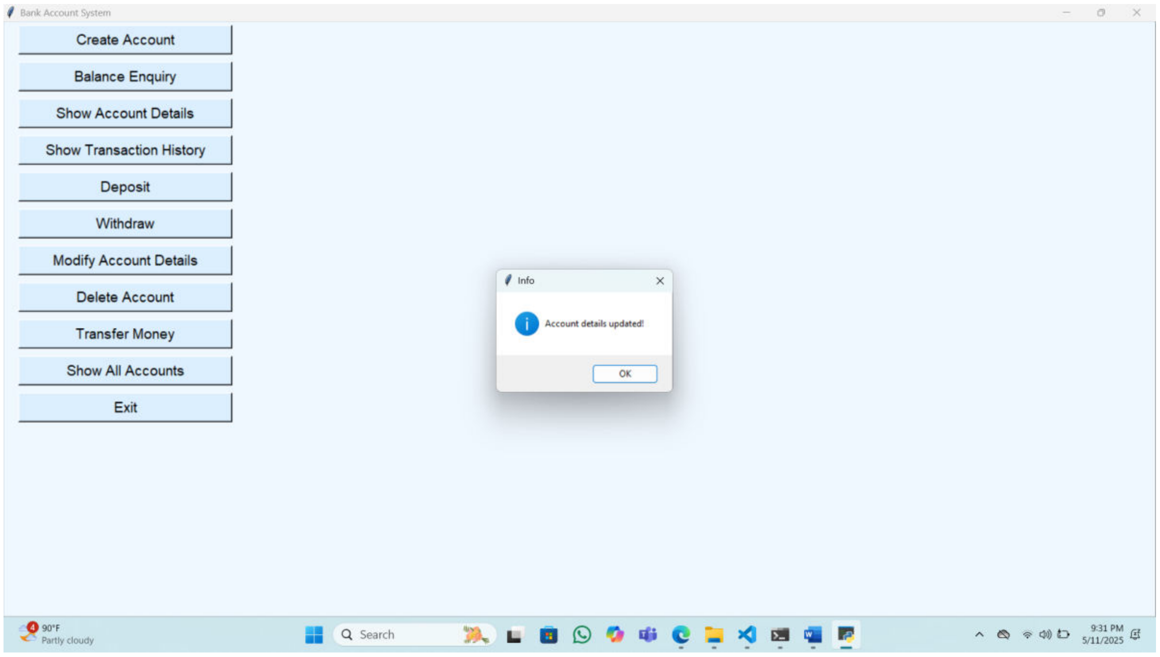
```
Deposited $100.00. Current balance: $700.00
```

Deposit sucessful page



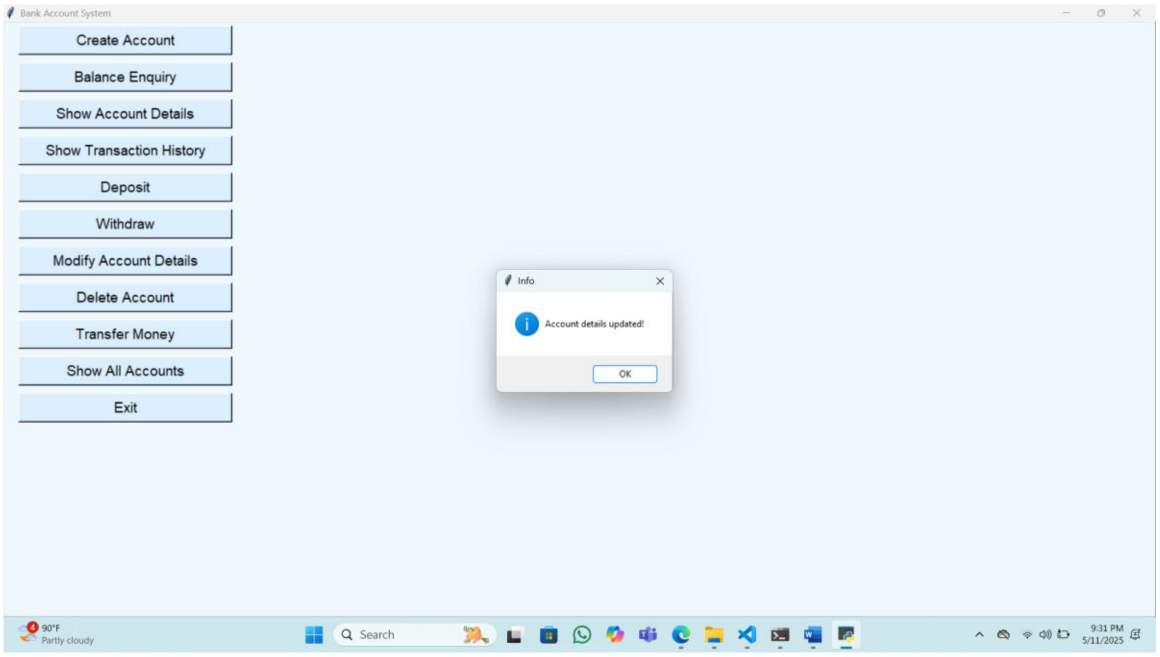
Withdrew $400.00. Current balance: $300.00

• Withdraw



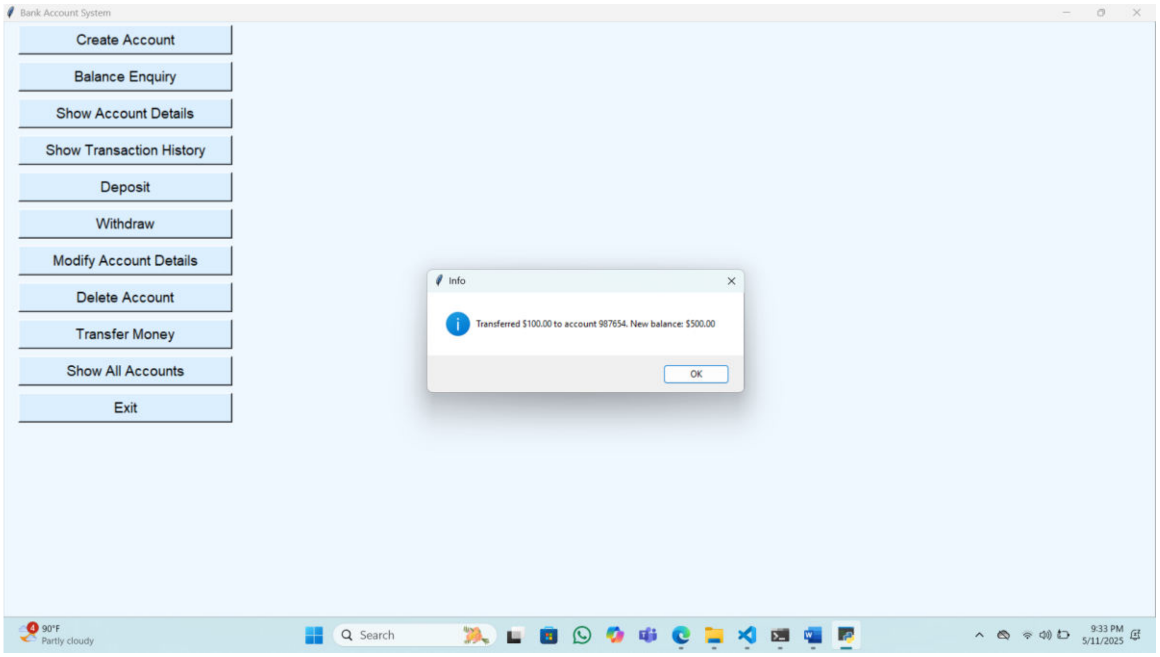
Account details updated!

• Account deletion



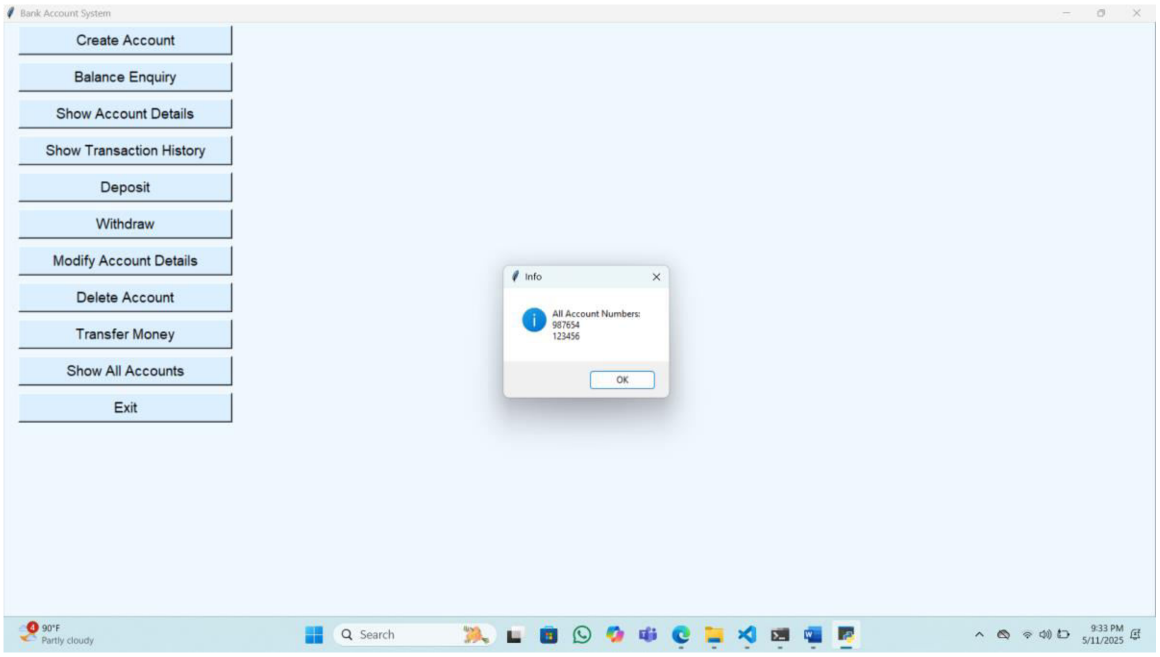
Account details updated!

• Modification of details



Transferred $100.00 to account 987654. New balance: $500.00

Transferring amount



```
All Account Numbers:
987654
123456
```

Displaying all accounts

6. CONCLUSION

The Bank Management System software is designed to streamline the management of customer accounts and banking transactions, ensuring a faster, more efficient banking experience. In today's fast-paced world, manual banking processes can be time-consuming and prone to errors, making automated solutions essential for maintaining smooth operations.This software will handle all essential banking tasks, including account creation, balance inquiries, deposits, withdrawals, transfers, and transaction history, making it easier for customers to manage their finances. It offers a user-friendly interface, reducing the workload of bank staff and improving overall customer satisfaction.

7. FUTURE SCOPE

For any system, present satisfaction is important, but is also necessary to see and visualizes the future scope. It is necessary for any system as the limitations that cannot be denied by anybody. These limitations, can be overcome by better technologies. In my project, records of the customers are transactions are maintained.

8. REFERENCES

- Faizan, M. (2012). Information and communication technology (ICT) and its role in driving global competition: A case study on SOA in Scandinavian and Swiss banks.

- Amwar, M. A. (2012). Transaction issues and security in banking systems: Addressing failures and fraud prevention through technology acceptance.

- Python For Desktop Applications: How to develop, pack and deliver Python applications with TkInter by Tran Duc Loi

- Python in A Nutshell: A Desktop Quick Reference, Third Edition by Alex Martelli , Anna Ravenscroft , Steve Holden

- The Python Language Reference Manual (version 3.2) by Guido van Rossum, and Fred L. Drake, Jr. (Editor)