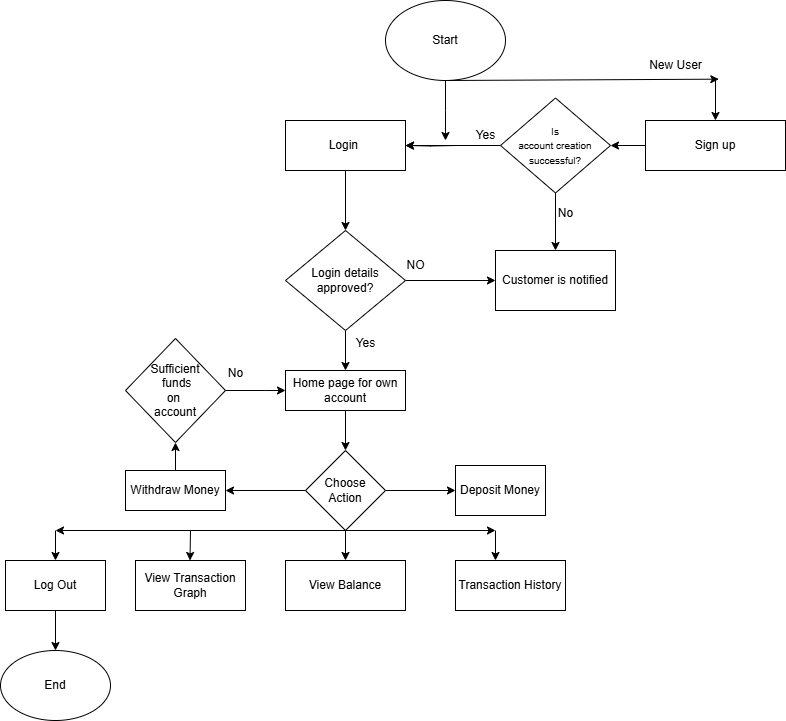
**Project report: Bank Account system using Python**

**Project description:**

The goal of this project is to develop simple banking system that includes all the basic operations such as Account creation, login, deposit and withdraw, view balance and transaction history. This application uses python as backend logic, Tkinter library for Graphical user interface, dictionary and csv files for data storage

**Flowchart**



**Flow of the Program**

**Account Creation**

* This bank account system allows user to create an account using Account name, Account number and Email id(optional). Account number must be alphanumeric characters and it should be between 6 and 10 characters. Account number is case sensitive
* The system validates the input and check whether it is unique account number
* New user information validating using dictionary, account number as a key, after validation account information write in to csv file and user receives the confirmation of successful creation

**Login**

* Once user creates the account, user can login using registered username and account number, it would take the user to their account dash board.
* The program reads the csv file in to dictionary, where account number as a key and account details as a value. This allows the validation of login credential
* Up on successful login users are redirect in to account dashboard

**Account Dashboard**

* In the Account dashboard account holder can see their current account balance, and perform all transactions like deposit and withdrawal. Account holder is able to see their transaction history during each login and have an option to view transaction graph where account holder can visualize their balance after each transaction.

**Deposit**

* User can enter the amount to deposit, system validate the input, it allows maximum 6-digit numbers (999,999) and allows only valid numbers
* After successful deposit, system will update the current balance and transaction history

**Withdraw**

* Users enter the amount to withdraw, system will check whether account have enough balance to withdraw specified funds
* If the balance is sufficient, system will update current balance and transaction history, otherwise system will throw the error message

**Transaction History**

* Users can view all the transactions they performed after logged in, including withdraw and deposit

**View transaction Graph**

* User can visualize their transaction and see the balance after each transaction

**Log out**

* Ends the session and redirects the user to the login screen.

**Link to GitHub**

<https://github.com/Rachanamannangari1/Bank_account_system>

**Programming Concept used in this project**

**Object oriented programming**

* Use the concept of classes that encapsulate methods and attributes.
* Created three classes Account, Bank Action, BankAccountGUI

**Data Management with CSV and Dictionary:**

* Reading and writing data from/to a CSV file (read\_accounts, save\_accounts).
* Loading data in to dictionaries to do all manipulation.

**GUI Development with Tkinter:**

* Creating graphical user interfaces (GUI) using Tkinter.
* Widgets like Tk, Toplevel, Frame, Label, Entry, Button.
* Event handling and callbacks (e.g., command=self. deposit).

**Error Handling:**

* Using try...except blocks to handle exceptions, such as invalid input or file not found errors.

**Data visualization using Matplotlib**

* Plotting graphs to visualize account balance changes over time.

**New techniques I learnt during this project**

* from PIL import Image, ImageTk

Used PIL\Pillow which stands for python imaging library. Image and ImageTk are two modules within PIL library. It is useful for working with images in python

* self.enter\_number.bind("<Button-1>", self.show\_message)

Bind method is used when certain event occur, corresponding function should be called. When I click on the text box it will call the show\_message method

"<Button-1>" refers to a mouse button click event, specifically the left mouse button.

* "\n".join(history)

Used join is a string method used to concatenate the list of transactions into a single string with each transaction on a new line.

**Screenshots of sample output**

