Shreyansh Padarha

Write-up I Original Idea

Automate the Boring Stuff

Application : ICICI bank database management system (with additional calculative tools) (completely self coded)

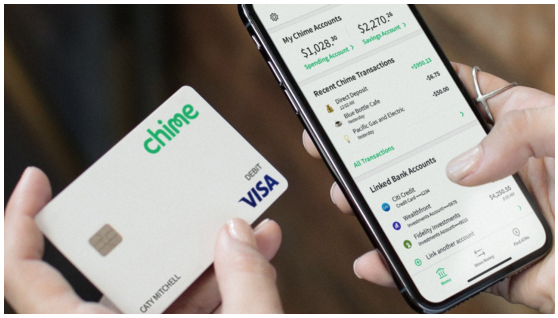
Introduction

The program that I intend to create is in the field of **banking, finance and managing records of bank accounts**.

The main aim is to facilitate banking executives with an **interactive app**, which can be dealt with easily.

Idea behind the development of the application :

In today's day and age majority of transactions happen remotely and digitally through various gadgets whether it be mobile phones, tablets or NFC smart watches.



In such a scenario, **banks need to maintain databases** which store personal information of various customers, and the basic necessary monetary details like the net balance and type of account.

What the program will pertain :

This program will try to combine all that in one space, with letting the user (manager or clerk at a bank's branch) chose from the following nominations :

- Display all existing records in the db
- Create a new account
- Edit personal details of a specific account
- Display all (personal and financial) details of an account.
- Delete an account.
- Withdraw money from a specific account. (**Eg**. can be used when a senior citizen comes to the bank and the clerk needs to help the customer withdraw money)
- Deposit money into an account.

All the details (personal details, account type, net balance) will be stored in a database which can be accessed easily with the help of the same program.

There will be additional tools which are relevant in the banking realm such as :

- Mathematical calculator
- Simple interest calculator

- Compound interest calculator.

Reason for choosing this specific problem and domain :

Personally, whenever I go to my **ICICI bank branch (that's the reason I created an ICICI themed app, with same colour patterns)** for depositing money, I feel that the clerks sitting and accessing their systems with plethora of records, must surely find it difficult to do so.

And I felt that the systems they use are very bland, hence the idea popped up to create a very user friendly bank management program.

Approach :

I am planning on using Tkinter module along with **Sqllite3** for the completion of the GUI eminent program.

Tkinter is a user friendly library which can let the developer create a GUI, with full customisability. The methods in it are highly modifiable and easy to use.

Sqlite3 is a relational database management system, which has similar commands to MYSQL, but is lighter on the system to boot. Its library is imported for updating, creating, deleting tables in a database.

I have already watched a 6hour video/course on YouTube for it, as I have previously not used Tkinter. I have made running notes while watching the video for further referencing.

I will create a rough blueprint on how I want to design my main page of the app, then I'll use the newly gained knowledge to create the program. I will also take help of knowledgable stackoverflow users for debugging bugs, errors and glitches.

The algorithms and logic used in the app will be extensively written and added in the final coding script, please check it out for a detailed guide to the approach I took

Shreyansh Padarha

MINI REPORT
[Automate the Boring Stuff]

Application : ICICI bank database management system (with additional calculative tools) (completely self coded)

I have created a **GUI based bank database management application**, for ease of accessing records of banking records and then further treating them.

The program for starters took me 46 hours [1300 lines of script] to code, but each second spent, felicitated new learnings, and I enjoyed the challenge thoroughly.

Whenever I used go to my **ICICI bank branch** for monetary work, I felt that the clerks sitting and accessing their systems with plethora of records, must find it difficult to do so with the kind of laggy and bland applications and systems they use. Hence, the idea popped up to create a **very** user friendly banking database management program. (I created an ICICI bank themed app, with same colour patterns, as that's what I've seen since childhood)

I planned on using **Tkinter** library along with S**qllite3** library for the completion of the GUI eminent program. This turned out to be a smart choice as Tkinter let me highly customise each widget (button, label, frame, entry, etc.) with whatever attribute (size, stretch, colour, fill and placement, etc.) I wanted; and Sqllite3 surprisingly had almost same commands as to Mysql, which I've previously self-learnt.

I started off with watching a 6 hour Tkinter course video on YouTube, along with watching the video, I made **running notes** on jupyter notebook for practice and further reference (submitted on Google classroom). The course also helped me get started with using widgets, the stepping stone of Tkinter. Along with this I started reading a bit on GitHub on how people have created whole fooding apps with Tkinter, this encouraged me to start as soon as possible.

I started off by making a blueprint of the main bank window, I had a main objective to make it look as similar as to the I-MOBILE ICICI bank app (available on Google Play store and App Store), so I further went on and researched on how to place pictures perfectly on a window, in the process I got to know about **Pillow** module of Tkinter, which helped me place the ICICI logo perfectly, which makes the program more relatable and realistic. I also figured background colour and resizing images. I fetched the ICICI American red, Deep Saffron orange and Blue sapphire colour's hexadecimal codes, for using it as my app's **theme.**

I decided to **create a main tkinter window** which would hold all my buttons, and tools. These buttons (tkinter widgets) had an attribute called command, which helped me to recall any user-defined function to it for execution. I made use of this to the utmost. I made **23 user defined functions**, with various **additional tkinter windows** opening inside them, which meant as soon as a button was clicked a new pop-up window would be made. The creation of functions helped me to traceback bugs, and helped while debigging. Another new thing I learnt, which might be the most useful and biggest takeaway in-built function for me from this project are **global variables**, which I kept using as inputs from entry boxes from one function needed to be recalled in another, and global variables helped globalise it for further use in any function.

For **Sqllite3** database linking it was very simple, a **connection** needed to be established with the database, a **cursor** needed to be created, the cursor would then be used for **executing** any sql based command (UPDATE, INSERT, CREATE,SET), then the database needed to be **committed**, and the connection **closed**.

I kept learning more intricacies within tkinter and sql3lite. Like for example, I got to know about scrollbar and frames that can be used in Tkinter windows, and also font customisations was a new finding. While a big revelation in databases realm was that, the individual rows are stored in forms of tuples.

There were many hurdles while creating the application, the biggest ones were small bugs and Mac specific glitches, and designing and placing each button specifically at a spot. The majority of time was taken to debug the bugs and find loopholes or other methods to get around problems. This was again fun to do as it involved wracking my brains and using logical reasoning.

The biggest helper by far was **Stackoverflow**, I used forums, threads and questions posted by other users to my advantage while looking to debug specific bugs. Each problem I faced could easily solved with a bit of effort, as there were other people facing the same issue on Stackoverflow and there were people answering .

I was able to achieve each one of the functionality from my program which I set out with,
The program has **financial tools** such as :

- Mathematical calculator
- Simple interest calculator
- Compound interest calculator.

Other **banking features** include :

- Display all existing records/bank accounts in the database
- Create a new account (prompt the user to input : personal details - name, address, city, state, zip code and banking details - account type (drop down menu), initial balance)
- Edit personal details of a specific account
- Display all (personal and financial) details of a specific account.
- Withdraw money from a specific account. (**Eg**. can be used when a senior citizen comes to the bank and the clerk needs to help the customer withdraw money quickly, the program can be used at that point)
- Deposit money into an account.
- Delete an account.

IMPROVEMENTS THAT CAN BE MADE :

There are some improvements I would make if given the opportunity for future use. The improvements include using a **visual editor IDLE** like ms studio, so that while placing the widgets, I could simultaneously check its positioning, another major upgrade I'd bring to my app is using **exceptions**, to create many more popups, in case the user types vague information. There are many other things that can be added, like **loans** and **credit card** facilities could be added or maybe **date time module** can be used to add clock in the app home page.

CONCLUSION :

There are many learnings to take from this project, this is by far the best thing I've created and only within a span of 5 days. I've learnt that programming is easy only and only if I'm willing to put in the hours, there are many more modules and libraries that I'd be interested to explore and learn in the near future.

NOTE : The output screenshots couldn't be attached here as there are a total of 25 permutations in my program, so I will be presenting them individually during my presentation. An output (screenshots) pdf file has been uploaded on the google classroom portal separately as well.