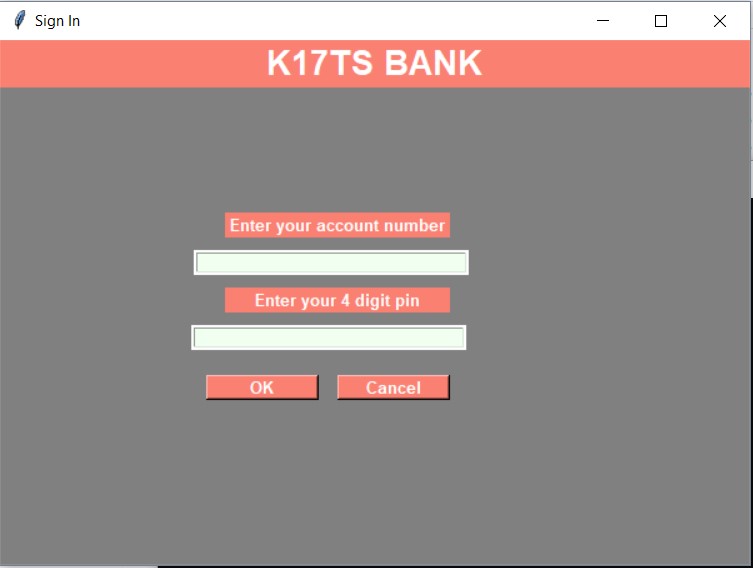


ATM Management System



**Prepared by -**

Nishant Yadav

Index

1.Introduction…………………………………………………………………………….……….…1

1.1. History….………………………………………………………………………….……2

1.2.Uses of ATM………………...…………………………………………………..……3

1. 3.Types of ATM……………………….……………………………………….……….4
2. Objective………………………………..…………………………………………………….….6
3. Methodologies Used…………………………………………………………………….....7

4.Modules……………………………………………………………………………………………9

5. Source Code……………..………………………………………………………………….…10

6.Screen Shots………………….………………………………………………………………..16

7.References………………………………………………………………………………….…..19

## Methodologies Used

1. Tkinter- The Tkinter module (“Tk interface”) is the standard Python interface to the Tk GUI toolkit from Scriptics (formerly developed by Sun Labs).

Tkinter consists of a number of modules. The Tk interface is provided by a binary extension module named \_tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.The public interface is provided through a number of Python modules. The most important interface module is the Tkinter module itself. To use Tkinter, all you need to do is to import the Tkinter module:

import Tkinter

1. SQLite- SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

1. import tkinter.messagebox - The tkMessageBox module is used to display message boxes in your applications. This module provides a number of functions that you can use to display an appropriate message.Some of these functions are showinfo, showwarning, showerror, askquestion, askokcancel, askyesno, and askretryignore.

1. from PIL import ImageTk - This module is for use with the graphical user interface functions of the Tkinter package. It is used to import images from the database or a directory.

1. root = Tk() - To initialize Tkinter, we have to create a Tk root widget. This is an ordinary window, with a title bar and other decoration provided by your window manager. We should only create one root widget for each program, and it must be created before any other widgets.
2. Var1=IntVar() - To create a Tkinter variable, the corresponding constructor is called upon. Note that the constructor takes an optional widget argument, but no value argument; to set the value, the set method is called.

1. w=Label(..) - This widget implements a display box where you can place text or images. The text displayed by this widget can be updated at any time you want.

1. sqlite3.connect(..) - This API opens a connection to the SQLite database file. You can use ":memory:" to open a database connection to a database that resides in RAM instead of on disk. If database is opened successfully, it returns a connection object.

1. Button(..) - The Button widget is used to add buttons in a Python application. These buttons can display text or images that convey the purpose of the buttons. You can attach a function or a method to a button which is called automatically when you click the button.
2. Frame(..) - The Frame widget is very important for the process of grouping and organizing other widgets in a somehow friendly way. It works like a container, which is responsible for arranging the position of other widgets.It uses rectangular areas in the screen to organize the layout and to provide padding of these widgets.

## Modules

1. Class Bank- It is a class which comprises of different functions and data members.
2. Database\_fetch()- Function used to fetch data from the database file.
3. Verify()- This functions validates the user by taking account number and pin from the user.
4. MainMenu()- This function comprises of the main screen which is shown to the user once it validates himself.
5. Account\_detail()- This function shows all the details of the user.
6. Balance()- This function returns balance of the account.
7. Deposit\_money()- This function accepts the value from user and adds it to the balance of the user.
8. Withdrawl\_money()- It accepts the amount of money user needs to withdraw and also updates it to the database.
9. Deposit\_trans()/Withdrawl\_trans()- It returns the prompt to the user that the respective task is completed.
10. Pin\_change()- This function allows the user to change the pin.

## Source Code

from tkinter import \* from tkinter import messagebox import sqlite3 class Bank: def \_\_init\_\_(self,root):

self.conn = sqlite3.connect("atm\_databse.db", timeout=100)

self.login = False self.root = root

self.header = Label(self.root,text="K17TS BANK",bg="salmon",fg="white",font="arial 20 bold")

self.header.pack(fill=X)

self.frame = Frame(self.root,bg="gray",width=600,height=400)

self.userlabel =Label(self.frame,text="Enter your account number",bg="salmon",fg="white",font="arial 10 bold")

self.uentry =

Entry(self.frame,bg="honeydew",highlightcolor="white",highlightthickness=2,highlightbackgr ound="white")

self.plabel = Label(self.frame, text="Enter your 4 digit pin",bg="salmon",fg="white",font="arial 10 bold")

self.pentry = Entry(self.frame,bg="honeydew",show="\*",highlightthickness=2, highlightbackground="white")

self.button = Button(self.frame,text="OK",bg="salmon",fg="white",font="arial 10 bold",command=self.verify)

self.q = Button(self.frame,text="Cancel",bg="salmon",fg="white",font="arial 10 bold",command = self.root.destroy) self.userlabel.place(x=180,y=100,width=180,height=20) self.plabel.place(x=180,y=160,width=180,height=20) self.uentry.place(x=155,y=130,width=220,height=20) self.pentry.place(x=153,y=190,width=220,height=20) self.button.place(x=165,y=230,width=90,height=20) self.q.place(x=270,y=230,width=90,height=20)

self.frame.pack()

def database\_fetch(self):#Fetching Account data from database

self.acc\_list = []

self.temp = self.conn.execute("select name,pass,acc\_no,acc\_type,bal from atm where acc\_no = ? ",(self.ac,))

for i in self.temp:

self.acc\_list.append("Name = {}".format(i[0])) self.acc\_list.append("Account no = {}".format(i[2])) self.acc\_list.append("Account type = {}".format(i[3]))

self.ac = i[2]

self.acc\_list.append("Balance = {}".format(i[4]))

def verify(self):#verifying of authorised user

ac = False

self.temp = self.conn.execute("select name,pass,acc\_no,acc\_type,bal from atm where acc\_no = ? ", (self.uentry.get()),)

for i in self.temp: self.ac = i[2] if i[2] == self.uentry.get():

ac = True elif i[1] == self.pentry.get():

ac = True

m = "{} Login SucessFull".format(i[0]) self.database\_fetch() messagebox.\_show("Login Info", m)

self.frame.destroy() self.MainMenu()

else:

ac = True

m = " Login UnSucessFull ! Wrong Password" messagebox.\_show("Login Info!", m)

if not ac:

m = " Wrong Acoount Number !" messagebox.\_show("Login Info!", m)

def MainMenu(self):#Main App Appears after logined !

self.frame = Frame(self.root,bg="gray",width=800,height=400) root.geometry("800x400")

self.detail = Button(self.frame,text="Account Details",bg="salmon",fg="white",font="arial

10 bold",command=self.account\_detail)

self.enquiry = Button(self.frame, text="Balance

Enquiry",bg="salmon",fg="white",font="arial 10 bold",command= self.Balance)

self.deposit = Button(self.frame, text="Deposit

Money",bg="salmon",fg="white",font="arial 10 bold",command=self.deposit\_money)

self.withdrawl = Button(self.frame, text="Withdrawl

Money",bg="salmon",fg="white",font="arial 10 bold",command=self.withdrawl\_money)

self.change = Button(self.frame, text="Pin Change",bg="salmon",fg="white",font="arial

10 bold",command=self.pin\_change)

self.q = Button(self.frame, text="Quit", bg="salmon", fg="white", font="arial 10 bold", command=self.root.destroy)

self.detail.place(x=0,y=120,width=200,height=50) self.enquiry.place(x=0, y=215, width=200, height=50) self.deposit.place(x=600, y=120, width=200, height=50) self.withdrawl.place(x=600, y=215, width=200, height=50) self.q.place(x=340, y=340, width=120, height=20) self.change.place(x=300,y=260,width=200,height=50)

self.frame.pack()

def account\_detail(self): self.database\_fetch() text = self.acc\_list[0]+"\n"+self.acc\_list[1]+"\n"+self.acc\_list[2] self.label = Label(self.frame,text=text,font="arial 10 bold") self.label.place(x=250,y=135,width=300,height=100)

def Balance(self):

self.database\_fetch() self.label = Label(self.frame, text=self.acc\_list[3],font="arial 10 bold") self.label.place(x=235, y=135, width=300, height=100)

def deposit\_money(self):

self.money\_box =

Entry(self.frame,bg="honeydew",highlightcolor="white",highlightthickness=2,highlightbackgr ound="white")

self.submitButton = Button(self.frame,text="Submit",bg="salmon",fg="white",font="arial

10 bold") self.money\_box.place(x=235,y=135,width=200,height=20) self.submitButton.place(x=445,y=135,width=55,height=20) self.submitButton.bind("<Button-1>",self.deposit\_trans)

def deposit\_trans(self,flag):

self.label = Label(self.frame, text="Transaction Completed !", font="arial 10 bold") self.label.place(x=235, y=135, width=300, height=100)

self.conn.execute("update atm set bal = bal + ? where acc\_no =

?",(self.money\_box.get(),self.ac)) self.conn.commit()

def withdrawl\_money(self):

self.money\_box =

Entry(self.frame,bg="honeydew",highlightcolor="white",highlightthickness=2,highlightbackgr ound="white")

self.submitButton = Button(self.frame,text="Submit",bg="salmon",fg="white",font="arial

10 bold") self.money\_box.place(x=235,y=135,width=200,height=20) self.submitButton.place(x=445,y=135,width=55,height=20) self.submitButton.bind("<Button-1>",self.withdrawl\_trans)

def withdrawl\_trans(self,flag):

self.label = Label(self.frame, text="Money Withdrawl !", font="arial 10 bold") self.label.place(x=235, y=135, width=300, height=100)

self.conn.execute("update atm set bal = bal - ? where acc\_no =

?",(self.money\_box.get(),self.ac)) self.conn.commit() def pin\_change(self):

self.pin\_box =

Entry(self.frame,bg="honeydew",highlightcolor="white",highlightthickness=2,highlightbackgr ound="white")

self.submitButton = Button(self.frame,text="Submit",bg="salmon",fg="white",font="arial

10 bold") self.pin\_box.place(x=235,y=135,width=200,height=20) self.submitButton.place(x=445,y=135,width=55,height=20) self.submitButton.bind("<Button-1>",self.pin\_trans)

def pin\_trans(self,flag):

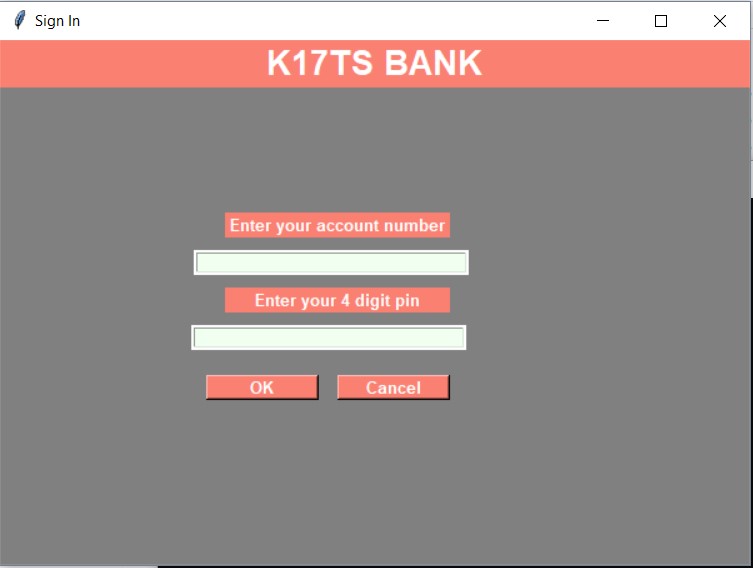
self.label = Label(self.frame, text="Successfully Updated! !",font="arial 10 bold") self.label.place(x=235, y=135, width=300, height=100) self.conn.execute("update atm set pin = ? where acc\_no = ?",(self.pin\_box.get(),self.ac)) self.conn.commit()

root = Tk()

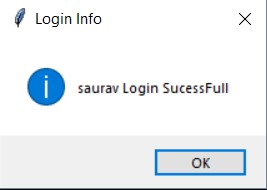
root.title("Sign In")

root.geometry("600x420") obj = Bank(root) root.mainloop()

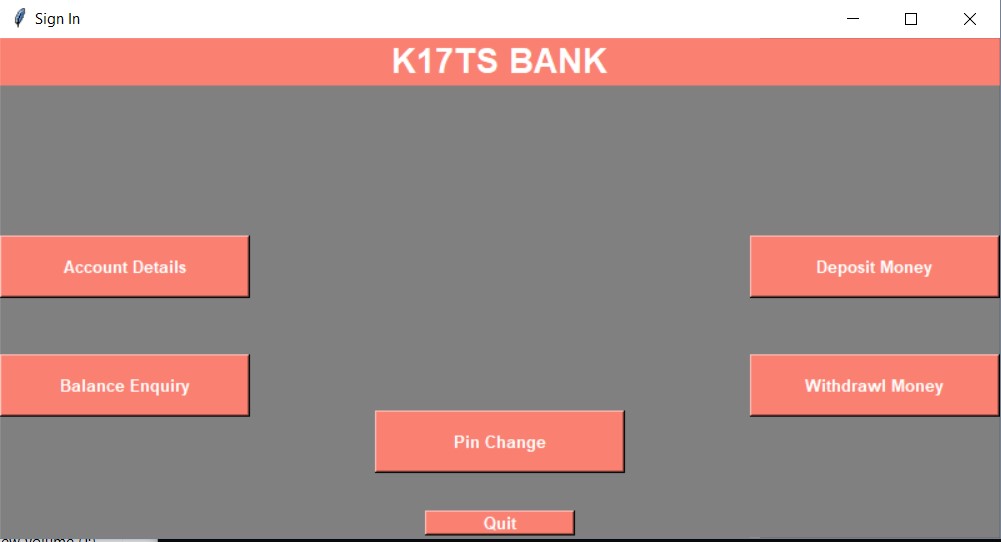
## Screen Shots



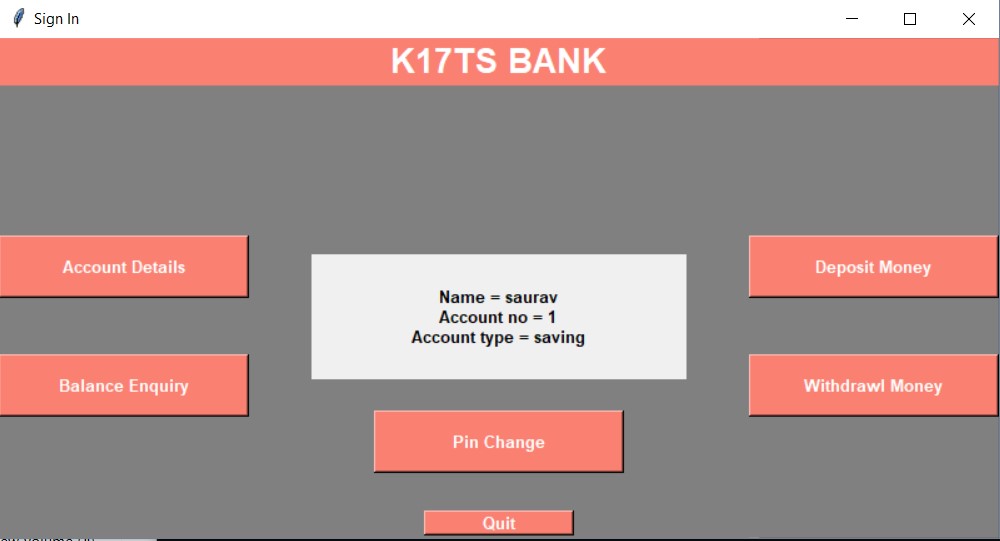
## Homescreen



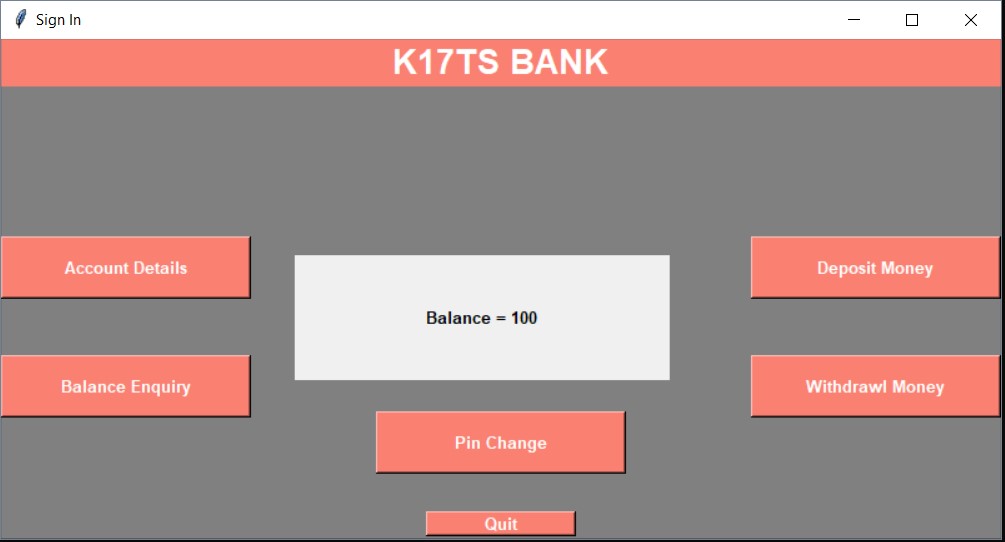
## Message Box



Welcome Screen



Account Details



## Balance Enquiry



## Pin Change

**References**

1. StackOverflow
2. http://effbot.org/tkinterbook/tkinter-hello-tkinter.htm
3. Tutorial Point
4. Python.org
5. PIL.org
6. https://pillow.readthedocs.io/en/5.3.x/
7. http://pythonware.com/products/pil/

thank you very much