**ALIGARH COLLEGE OF**

**ENGINEERING AND TECHNOLOGY**

(Affiliated to AKTU, Lucknow - College Code: 109)

Aligarh, Uttar Pradesh – 202001



A MINI PROJECT REPORT

ON

**“BANK MANAGEMENT SYSTEM”**

**Submitted by: Submitted to:**

Name: AMIT SHARMA **Mr. S.K. Sharma**

Roll Number: 1810910006

Sec: A

Semester: 3

**Head of Dept: Dr. Anand Sharma**

**Department of Computer Science & Engineering**

Aligarh College of Engineering and Technology

2019-20

ACKNOWLEDGEMENT

I express my sincere gratitude and thanks to Google and udemy for providing me the excellent opportunity to do a project on **Bank Management System** and providing me with all the essential elements required for the completion and enhancement of this project.

I express my sincere thanks to the **Mr. S.K. Sharma** **and Dr. Anand** **sharma** ,Head of Department , Computer Science and Engineering for their kind co-operation.

I would like to thank those respondents who have taken pain in successful completion of my project work.

DECLARATION

I here declare that submitted is my own work and that , to the best of my knowledge and belief , it contains no material previously published or written by any another preson material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learing ,except where due acknowledgment has been made in the text.

Name: Amit sharma

Roll Number : 181091006

Date :

Signature :

## CERTIFICATE-

Certified that Amit sharma have carried out the mini project entitled “Bank management system “ for the award of Bachelor of Technology from Dr. A.P.J.Abdul Kalam Technical University ,Lucknow under my supervision .The project embodies result of original work and studies carried out by Student them self and the content of the project do not from the basis for the award of any other degree to the candidate or to any boby else.

Amit sharma

Date :

Signature :

## **TABLE OF CONTENTS**

* Introduction
* Objective
* System requirement specification
* Feasibility Analysis
* Module Use
* Project Development
* Design
* Coding

**INTRODUCTION: -**

During the past several decades’ personnel function has been transformed from a relatively obscure record keeping staff to central and top level management function. There are many factors that have influenced this transformation like technological advances, professionalism, and general recognition of human beings as most important resources.

A computer based management system is designed to handle all the primary information required to calculate monthly statements of customer account which include monthly statement of any month. Separate database is maintained to handle all the details required for the correct statement calculation and generation.

This project intends to introduce more user friendliness in the various activities such as record updating, maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updating can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The entire information has maintained in the database or Files and whoever wants to retrieve can’t retrieve, only authorization user can retrieve the necessary information which can be easily be accessible from the file.

OBJECTIVE: -

A computer based management system is designed to handle all the primary information required to calculate monthly statements of customer account which include monthly statement of any month. Separate database is maintained to handle all the details required for the correct statement calculation and generation.

This project intends to introduce more user friendliness in the various activities such as record maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updating can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The main objective of our project is providing the different typed of customers’ facility, the main objective of this system is to find out the actual customer service. Etc.

This system is very easy to use, so that any user can use without getting pre-knowledge about this. It’s very much user friendly and meet almost all daily working process requirements. This system is completely GUI.

FEASIBILITY ANALYSIS: -

Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. A feasibility study is a test of a system proposal. According to its workability, impact on the organization, ability to meet user’s needs and effective use of the resources its main task done during the feasibility study are: -

TECHNICAL FEASIBILITY: -

The proposed system is technically feasible as it can be developed easily with the help of available technology. The proposed system requires LINUX – red hat enterprises using VM –ware as Interface for Programming & FILE handling for storing/maintaining database.

OPERATIONAL FEASIBILITY: -

Automation makes our life easy. The proposed system is highly user friendly and is much easily able to interact with the system. Therefore, the users will readily accept the system as data entry and making queries can be easily done.

**MODULE USED:-**

***Tkinter*** - The [Tkinter](https://docs.python.org/2/library/tkinter.html" \l "module-Tkinter" \o "Tkinter: Interface to Tcl/Tk for graphical user interfaces) module (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and [Tkinter](https://docs.python.org/2/library/tkinter.html" \l "module-Tkinter" \o "Tkinter: Interface to Tcl/Tk for graphical user interfaces) are available on most Unix platforms, as well as on Windows systems.(Tk itself is not part of Python; it is maintained at ActiveState.)

Running python -m Tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that **[Tkinter](https://docs.python.org/2/library/tkinter.html" \l "module-Tkinter" \o "Tkinter: Interface to Tcl/Tk for graphical user interfaces)** is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documention specific to that version.

***Shelve-***The shelve module in Python’s standard library is a simple yet effective tool for persistent data storage when using a relational database solution is not required. The shelf object defined in this module is dictionary-like object which is persistently stored in a disk file. This creates afile similar to dbm database on UNIX like systems. Only string data type can be used as key in this special dictionary object, whereas any picklable object can serve as value.

***OS-***The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The \*os\* and \*os.path\* modules include many functions to interact with the file system.

***Datetime -***The [datetime](https://docs.python.org/3/library/datetime.html#module-datetime) module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. For related functionality, see also the [time](https://docs.python.org/3/library/time.html#module-time) and [calendar](https://docs.python.org/3/library/calendar.html#module-calendar) modules.

SYSYTEM REQUIREMENTS:

Software Requirements**: -**

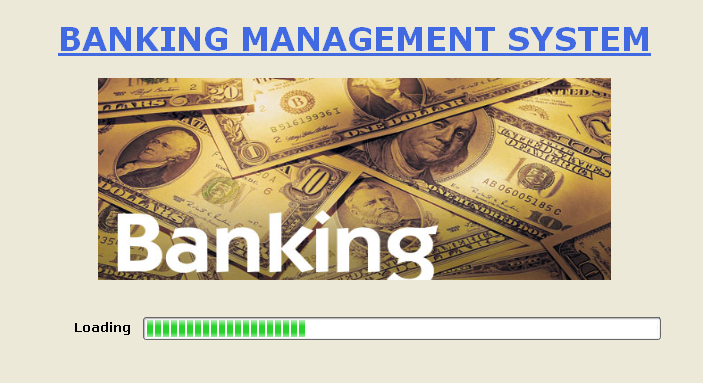
The software is a set of procedures of coded information or a program which when fed into the computer hardware, enables the computer to perform the various tasks. Software is like a current inside the wire, which cannot be seen but its effect can be felt.

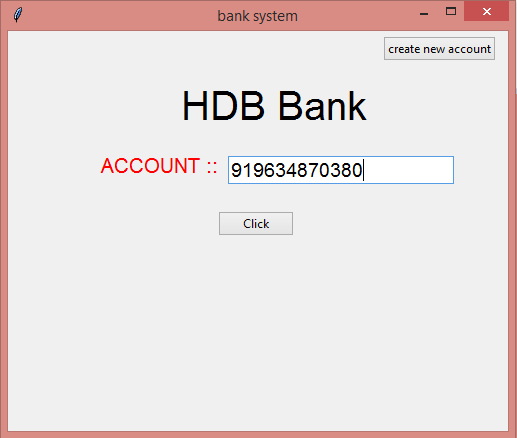
1. Operating System-Windows

2. Application Software: -Python 3.7.2 64bit

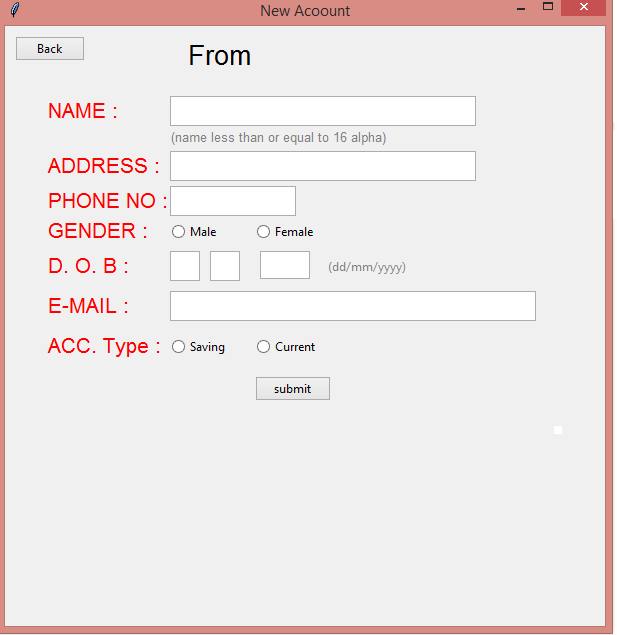
3. Editor: - IDLE

**DESIGN AND SNAPSHOT: -**

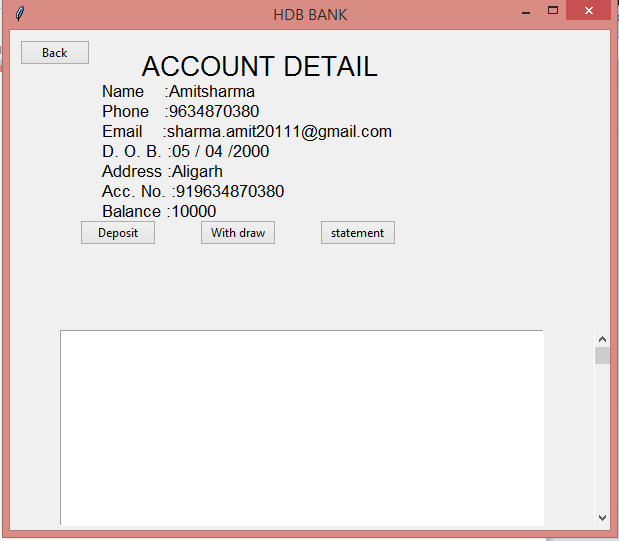
* SPLASH FORM
* 
* **Opening Account**



* ACCOUNT OPENING FORM



**Account operation**



**CODING: -**

import datetime

import tkinter as t

from tkinter import messagebox

from tkinter import \*

from tkinter import ttk

import os

import shelve

#chanding directory

os.chdir('C:\\Users\\charm\\Desktop\\db')

r=t.Tk()

r.title('bank system')

def call\_main():

#disable window m

m.state(['withdraw'])

main()

def check\_value():

# creating new account

if ent\_phone.get().isdigit() ==False or len(ent\_phone.get())!=10:

messagebox.showinfo('Error','Please enter correct phone no. value')

elif ent\_file.get().isalpha()==False:

messagebox.showinfo('Error','Please enter correct name value')

try:

if (int(e\_d.get())>31 or int(e\_d.get())<1) or( int(e\_m.get())<1 or int(e\_m.get())>12) or len(e\_y.get())!=4 :

messagebox.showinfo('Error','Please enter correct Date of birth value')

else:

creating\_acc()

except ValueError:

messagebox.showinfo('Error','Please fill entrys enter')

def creating\_acc():

a=shelve.open('acclist')

file\_name='91'+ent\_phone.get()

k=a['list']

k.append(file\_name)

a['list']=k

a.close()

file\_name='91'+ent\_phone.get()

new\_file=shelve.open(file\_name)

new\_file['acc\_num']='91'+ent\_phone.get()

new\_file['user\_name']=ent\_file.get()

new\_file['address']=ent\_add.get()

new\_file['phone']=ent\_phone.get()

new\_file['acc\_type']=acctype.get()

new\_file['email']=ent\_mail.get()

new\_file['balance']=0

new\_file['statement']=[ ]

new\_file['index']=1

new\_file['dob']=e\_d.get()+' / ' +e\_m.get()+' /'+e\_y.get()

new\_file.close()

messagebox.showinfo('MESSAGE','''YOUR ACCOUNT IS SUCCESSFUL CREATE’’’+file\_name)

AND YOUR ACC. NO. '''+file\_name)

def new():

# creating new window for new account

global ent\_file, m , ent\_add, ent\_phone,acctype,gen,ent\_mail,e\_d,e\_m,e\_y

r.state(['withdraw'])

m=t.Tk()

m.title('New Acoount')

lab\_=Label(m,text="From ",font=("",20))

lab\_.place(x=180,y=10)

lab\_name=Label(m,text='NAME :',font=("",16),fg='red')

lab\_name.place(x=40,y=70)

lab\_messege=Label(m,text=" (name less than or equal to 16 alpha)",font=("",10),fg='grey')

lab\_messege.place(x=155,y=100)

lab\_add=Label(m,text='ADDRESS :',font=("",16),fg='red')

lab\_add.place(x=40,y=125)

lab\_phone=Label(m,text='PHONE NO :',font=("",16),fg='red')

lab\_phone.place(x=40,y=160)

lab\_gender=Label(m,text='GENDER :',font=("",16),fg='red')

lab\_gender.place(x=40,y=190)

lab\_dob=Label(m,text='D. O. B :',font=("",16),fg='red')

lab\_dob.place(x=40,y=225)

lab\_mail=Label(m,text='E-MAIL :',font=("",16),fg='red')

lab\_mail.place(x=40,y=265)

lab\_acctype=Label(m,text='ACC. Type :',font=("",16),fg='red')

lab\_acctype.place(x=40,y=305)

lab\_d=Label(m,text='(dd/mm/yyyy)',fg='grey')

lab\_d.place(x=320,y=230)

e\_d=ttk.Entry(m,width=2,font=("",16))

e\_d.place(x=165,y=225)

e\_m=ttk.Entry(m,width=2,font=("",16))

e\_m.place(x=205,y=225)

e\_y=ttk.Entry(m,width=4,font=("",14))

e\_y.place(x=255,y=225)

ent\_phone=ttk.Entry(m,width=10,font=("",16))

ent\_phone.place(x=165,y=160)

ent\_file=ttk.Entry(m,width=25,font=("",16))

ent\_file.place(x=165,y=70)

ent\_add=ttk.Entry(m,width=25,font=("",16))

ent\_add.place(x=165,y=125)

ent\_mail=ttk.Entry(m,width=30,font=("",16))

ent\_mail.place(x=165,y=265)

acctype=StringVar()

ttk.Radiobutton(m,text='Saving',value='Male',variable=acctype).place(x=165,y=310)

ttk.Radiobutton(m,text='Current',value='Female',variable=acctype).place(x=250,y=310)

gen=StringVar()

aaa=ttk.Radiobutton(m,text='Male',value='0',variable=gen)

aaa.place(x=165,y=195)

a=ttk.Radiobutton(m,text='Female',value='1',variable=gen)

a.place(x=250,y=195)

but\_submit=ttk.Button(m,text="submit" ,command=check\_value)

but\_submit.place(x=250,y=350)

#ent\_.insert(0,"less then or equal to 16 letters")

but\_back=ttk.Button(m,text="Back",width=10,command=call\_main)

#but\_back.place(x=10,y=10)

# but\_back=ttk.Button(m,text="verifi email",width=10,command=call\_main)

#but\_back.place(x=10,y=10)

but\_back.grid(row=300,column=10,padx=10,pady=10)

#check\_value

m.geometry('600x600')

m.mainloop()

def open(a):

global s

op=shelve.open('acclist')

o=op['list']

name=ent.get()

if name in op['list']:

r.state(['withdraw'])

s=t.Tk()

s.title('HDB BANK')

def withdraw():

global e\_withdraw

l\_withdraw=ttk.Label(s,text='Enter Amount to withdraw :',font=('',16)).place(x=45,y=230)

e\_withdraw=ttk.Entry(s,width=10,font=('',14))

e\_withdraw.focus\_set()

e\_withdraw.bind('<Return>',sub\_wid)

e\_withdraw.place(x=300,y=230)

b\_withdraw=ttk.Button(s,text='Click',command=sub\_wid).place(x=150,y=270)

def deposit():

global e\_deposit

l\_deposit=ttk.Label(s,text='Enter Amount to Deposit :',font=('',16)).place(x=45,y=230)

e\_deposit=ttk.Entry(s,width=10,font=('',14))

e\_deposit.focus\_set()

e\_deposit.bind('<Return>',add\_dep)

e\_deposit.place(x=300,y=230)

b\_deposit=ttk.Button(s,text='Click',command=add\_dep).place(x=150,y=270)

def sub\_wid(a):

if e\_withdraw.get().isdigit()== True:

detail=shelve.open(name)

if detail['balance']>2000:

detail['balance']-=int(e\_withdraw.get())

lab\_balance.config(text='Balance :'+str(detail['balance']))

k=detail['statement']

tj.config(state='normal')

d=datetime.datetime.now()

tj.delete('1.0', END)

tj.insert(INSERT,'Your '+e\_withdraw.get()+'\trupees successfull withdraw\n')

tj.config(state='disabled')

k.append(str(str(detail['index'])+' '+str(d.year)+ ','+str(d.month)+','+str(d.day)+' ' + e\_withdraw.get()+'dr. '+str(detail['balance'])))

detail['index']+=1

#print(detail['statement'])

detail['statement']=k

#print(str(d))

detail.close()

e\_withdraw.delete(0,t.END)

#messagebox.showinfo('With draw',e\_withdraw.get()+' rupess successfully withdraw')

else:

messagebox.showinfo('Error','Low Balance')

else:

messagebox.showinfo('Error','Enter correct value')

def add\_dep(a):

if e\_deposit.get().isdigit()== True:

detail=shelve.open(name)

detail['balance']+=int(e\_deposit.get())

k=detail['statement']

tj.config(state='normal')

d=datetime.datetime.now()

tj.delete('1.0', END)

lab\_balance.config(text='Balance :'+str(detail['balance']))

tj.insert(INSERT,'Your '+e\_deposit.get()+'\trupees successfull deposit\n')

tj.config(state='disabled')

k.append(str(str(detail['index'])+' '+str(d.year)+ ','+str(d.month)+','+str(d.day)+' '+e\_deposit.get()+'cr. '+' ' +str(detail['balance'])))

detail['index']+=1

#print(detail['statement'])

detail['statement']=k

#print(str(d))

detail.close()

detail.close()

e\_deposit.delete(0,t.END)

#messagebox.showinfo('Deposit',e\_deposit.get()+' rupess successfully deposit')

else:

messagebox.showinfo('Error','Enter correct value')

def sat():

detail=shelve.open(name)

tj.config(state='normal')

tj.delete('1.0', END)

tj.insert(INSERT,'So.no. Date Deposit Withdraw Balance \n')

#t.insert(INSERT,str(str(detail['index'])+'\t' + e\_withdraw.get()+'dr.\t'+str(detail['balance'])))

for i in detail['statement']:

#print(i,end='\n')

tj.insert(INSERT,i+'\n')

tj.config(yscrollcommand=scroll.set)

tj.config(state='disabled')

tj=Text(s,width=60,height=12,wrap='word')

tj.grid(row=1,column=0,pady=300,padx=50)

scroll=Scrollbar(s,orient=VERTICAL,command=tj.yview)

scroll.grid(row=1,column=1,pady=300,sticky=N+S)

# scroll.place(x=460,y=230)

#tj.place(x=30,y=300)

tj.config(state='disabled')

#tj.config(state='disabled')

global lab\_balance

detail=shelve.open(name)

lab\_=ttk.Label(s,text='ACCOUNT DETAIL',font=('',20)).place(x=130,y=18)

lab\_name3=ttk.Label(s, text='Name :'+ detail['user\_name'],font=('',13)).place(x=90,y=50)

lab\_phone2=ttk.Label(s, text='Phone :'+ detail['phone'],font=('',13)).place(x=90,y=70)

lab\_email3=ttk.Label(s, text='Email :'+ detail['email'],font=('',13)).place(x=90,y=90)

lab\_dob=ttk.Label(s,text='D. O. B. :'+ detail['dob'],font=('',13)).place(x=90,y=110)

lab\_phone25=ttk.Label(s, text='Address :'+ detail['address'],font=('',13)).place(x=90,y=130)

lab\_email33=ttk.Label(s, text='Acc. No. :'+ detail['acc\_num'],font=('',13)).place(x=90,y=150)

lab\_balance=ttk.Label(s,text='Balance :'+ str(detail['balance']),font=('',13))

lab\_balance.place(x=90,y=170)

b=ttk.Button(s,text='Deposit',command=deposit).place(x=70,y=190)

b1=ttk.Button(s,text='With draw',command=withdraw).place(x=190,y=190)

b11=ttk.Button(s,text='statement',command=sat).place(x=310,y=190)

b\_back=ttk.Button(s,text='Back',width=10,command=back\_main).place(x=10,y=10)

s.geometry('600x500')

s.mainloop()

else:

messagebox.showinfo('ERROR','Acc. No. Not Found')

def back\_main():

s.state(['withdraw'])

r.state(['normal'])

def main():

# main starting of program

r.state(['normal'])

global ent

lab=Label(r,text='HDB Bank',font=("",30))

lab1=Label(r,text='ACCOUNT ::',font=('',16),fg='red')

ent=ttk.Entry(r,width=20,font=("",14))

but =ttk.Button(r,text='Click',command=open)

but1=ttk.Button(r,text='create new account',command=new)

lab.place(x=170,y=50)

lab1.place(x=90,y=120)

ent.place(x=220,y=125)

but.place(x=210,y=180)

but1.place(x=375,y=5)

ent.focus\_set()

ent.bind('<Return>',open)

r.geometry('500x400+100+50')

main()

r.mainloop()

BIBLIOGRAPHY AND REFERENCES:

Book :- AUTOMATE THE BORING STUFF WITH PYTHON

WEBSITES: - [www.udemy.com](http://www.udemy.com) ,[www.geeksforgeeks.com](http://www.geeksforgeeks.com)

SEARCH ENGINES: - GOOGLE

**“Thank you “**