DPES
Dhruv Patil Education Society

DHOLE PATIL COLLEGE OF ENGINEERING
WAGHOLI,PUNE-412207

A

PROJECT REPORT ON

BANKING MANAGEMENT SYSTEM

Submitted by:

1. Rohit Saxena PRN- 72154098J Roll no. 41

2. Elisha Elanjikal PRN- 72153979D Roll no.09

3. Ketki Tatarkar PRN- 72154136E Roll no.47

in partial fulfilment for the award of the degree
OF

BACHELOR OF ENGINEERING IN
INFORMATION TECHNOLOGY AT

Savitribai Phule Pune University, Pune

Under The Guidance of Prof.

NEHALI SHINDE

DEPARTMENT OF INFORMATION TECHNOLOGY

DHOLE PATIL COLLEGE OF ENGINEERING WAGHOLI,PUNE-412207

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that,

Rohit Saxena PRN-72154098J

Elisha Elanjikal PRN-72153979D

Ketki Tatarkar PRN-72154136E

Have successfully completed the project entitle "BANKING MANAGEMENT SYSTEM" in partial fulfilment for the award of the degree of Bachelors of Engineering in Information Technology of Savitribai Phule Pune University during the Academic year 2021-2022. The project report has been approved as it satisfies the academic requirement in respect to project work prescribes by the said degree.

Date:- 24/May/2022

Place:-DPCOE

Prof. Nehali Shinde
(Project Guide)

Prof. Rahul Ghode
(Head of the department)

(External Examiner)

Dr. Nihar Walimbe
(Principal)

DPES
Dhruv Patil Education Society

DPES
Dhruv Patil Education Society

TABLE OF CONTENTS

1. ***SYNOPSIS***

1.1 Project title.....1
1.2 Project Option.....1
1.3 Internal Guide.....1
1.4 Technical Keywords.....1
1.5 Abstract.....1
1.6 Goals 1
1.7 Accessibility.....1
1.8 Software Requirement.....1

2. ***Introduction***

2.1 Module Description.....2
2.2 Working Model.....2
2.3 ER Diagram.....2
2.4 Method.....2
2.5 User Module.....2

3. ***System & Database Design***

3.1 System Design.....3
3.2 Logical Design.....3
3.3 Physical Design.....3
3.4 Database Design.....3
3.4 Data Flow Diagram.....3

4. ***Terms of Service & Security***

4.1 General Information.....4
4.2 Security Terms.....4
4.3 Bank Term's.....4
4.4 Customer Obligation.....4
4.5 Safe online Banking Tips.....4
4.6 Beware of phishing Attack.....4

5. ***Code & Snapshots of Project***

5.1 Code
5.2 Tkinter Imported Code
5.3 Output

Project Title

The **bank management system project proposal** states the solution and the problems faced by bank management. It should contain the project objectives, scope, and description. The Bank Management System (BMS) is a web-based tool that is used to reimburse financial institutions for services rendered to the Bureau of the Fiscal Service. In addition, BMS provides analytical tools for reviewing and approving salaries, budgets, and outflows.
The goal of the bank management system project is to create an organic and optimal software of interaction between the various banking components. This is to maximize the profit of the banking mechanism. The implementation of competent bank management procedures is significantly responsible for the successful optimization of the bank's productivity and activities.

The project's main goal is to create an online banking system for banks. All banking work is done manually in the current system. To withdraw or deposit money, the user must go to the bank. Today, it is also hard to find account information for people who have accounts in the banking system. The bank management system project is a program that keeps track of a client's bank account. This project demonstrates the operation of a banking account system and covers the essential functions of bank management software. It develops a project for resolving a customer's financial applications in a banking environment to meet the needs of an end banking user by providing multiple ways to complete banking chores. Additionally, this project is to provide additional features to the user's workspace that are not available in a traditional banking project. The project's bank management system is built on cutting-edge technologies. This project's main goal is to create software for a bank account management system. This project was designed to make it simple and quick to complete previously impossible processes with manual systems which are now possible with this software.

Project Options-

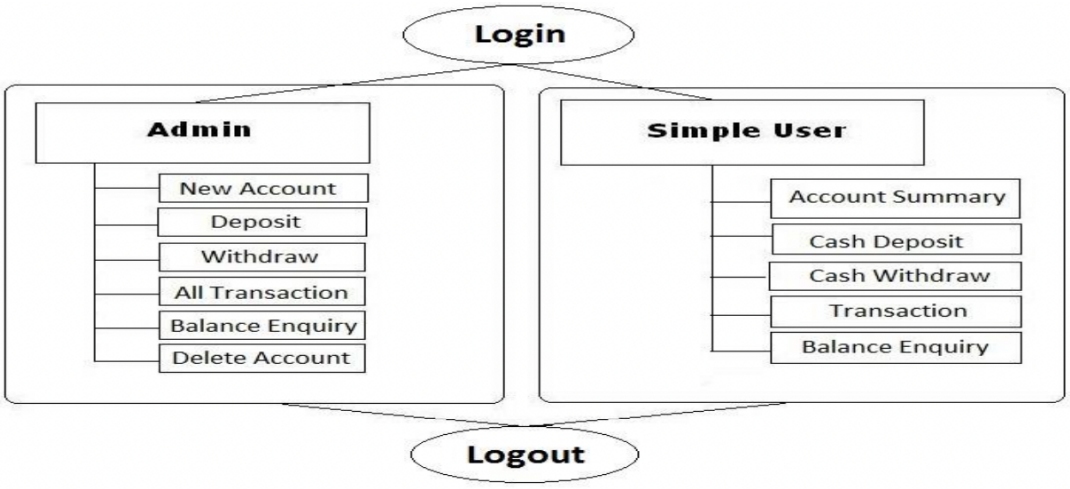
The “Bank Account Management System” project is a model Internet Banking Site. This site enables the customers to perform the basic banking transactions by sitting at their office or at homes through PC or laptop. The system provides the access to the customer to create an account, deposit/withdraw the cash from his account, also to view reports of all accounts present. The customers can access the banks website for viewing their Account details and perform the transactions on account as per their requirements. With Internet Banking, the brick and mortar structure of the traditional banking gets converted into a click and portal model, thereby giving a concept of virtual banking a real shape. Thus today's banking is no longer confined to branches. E-banking facilitates banking transactions by customers round the clock globally. Bank is a place where customers feel a sense of safety for their property. In the bank, customers deposit and withdraw their money. Transaction of money also is a part where the customer takes shelter in the bank. Now to keep the belief and trust of customers, there is the positive need for management of the bank, which can handle all this with comfort and ease. Smooth and efficient management affects the satisfaction of the customers and staff members, indirectly. And of course, it encourages management committee in taking some needed decision for future enhancement of the bank. The Bank Account Management System keeps the day by day tally record as a complete banking system. It can keep the information of Account type, account opening form, Deposit fund, Withdrawal, and Searching the transaction, Transaction reports, Individual account opening form, Group Account.

Internal Guide

The purpose of Online Banking System is to automate the existing manual system by the help of computerized equipments and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Online Banking System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus it will help organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipments and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically the project describes how to manage for good performance and better services for the clients.



DPES
Dhruv Patil Education Society

ABSTRACT

The Bank Account Management System is an application for maintaining a person's account in a bank. In this project I tried to show the working of a banking account system and cover the basic functionality of a Bank Account Management System. To develop a project for solving financial applications of a customer in banking environment in order to nurture the needs of an end banking user by providing various ways to perform banking tasks. Also to enable the user's workspace to have additional functionalities which are not provided under a conventional banking project.

The Bank Account Management System undertaken as a project is based on relevant technologies. The main aim of this project is to develop software for Bank Account Management System. This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software. This project is developed using PHP, HTML language and MYSQL use for database connection. Creating and managing requirements is a challenge of IT, systems and product development projects or indeed for any activity where you have to manage a contractual relationship. Organization need to effectively define and manage requirements to ensure they are meeting needs of the customer, while proving compliance and staying on the schedule and within budget. The impact of a poorly expressed requirement can bring a business out of compliance or even cause injury or death. Requirements definition and management is an activity that can deliver a high, fast return on investment.

Bank is the place where customers feel the sense of safety for their property. In the bank, customers deposit and withdraw their money. Transaction of money also is a part where customer takes shelter of the bank. Now to keep the belief and trust of customers, there is the positive need for management of the bank, which can handle all this with comfort and ease. Smooth and efficient management affects the satisfaction of the customers and staff members, indirectly. And of course, it encourages management committee in taking some needed decision for future enhancement of the bank.

Goals for this Project Proposal-

The Bank Management System (BMS) is a web-based tool that is used to reimburse financial institutions for services rendered to the Bureau of the Fiscal Service. In addition, BMS provides analytical tools for reviewing and approving salaries, budgets, and outflows.

The bank management system(BMS) project is a program that keeps track of a client's bank account. This project demonstrates the operation of a banking account system and covers the essential functions of bank management software. It develops a project for resolving a customer's financial applications in a banking environment to meet the needs of an end banking user by providing multiple ways to complete banking chores. Additionally, this project is to provide additional features to the user's workspace that are not available in a traditional banking project. The project's bank management system is built on cutting-edge technologies. This project's main goal is to create software for a bank account management system. This project was designed to make it simple and quick to complete previously impossible processes with manual systems which are now possible with this software.

Accessibility of the Software-

- Access specifies the access right for the statements or functions that follow it until another access specifier or till the end of a class. The client of the bank can perform various tasks that are given below-
- Client has a unique customer id and password.
- Clients can do various transactions.
- Bank has its unique bank id, branch-city,branch phone.
    - Create a Bank Account.
    - Deposit & Withdraw Money

- Bank Account Type Support (e.g. Current Account, Savings Account)
- Interest calculation depending on the Bank Account type
- Transaction report with a date range filter
- See balance after every transaction in the Transaction Report
- Calculate Monthly Interest Using Celery Scheduled tasks
- More efficient and accurate interest calculation and balance update
- Ability to add Minimum and Maximum Transaction amount restriction.
-

Hardware and Software Requirement-

Hardware requirement-

- *Pentium IV or higher, (PIV-300GHz recommended)*
- *256 MB RAM*
- *1 Gb hard free drive space*

Processor	AMD Ryzen 5 5500U with Radeon Graphics, 2.10 GHz
Installed RAM	8.00 GB (7.34 GB usable)
Device ID	4D0BFD4C-DD2C-4651-82B7-0357D42ABF83
Product ID	00327-36334-59796-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Software Requirement-

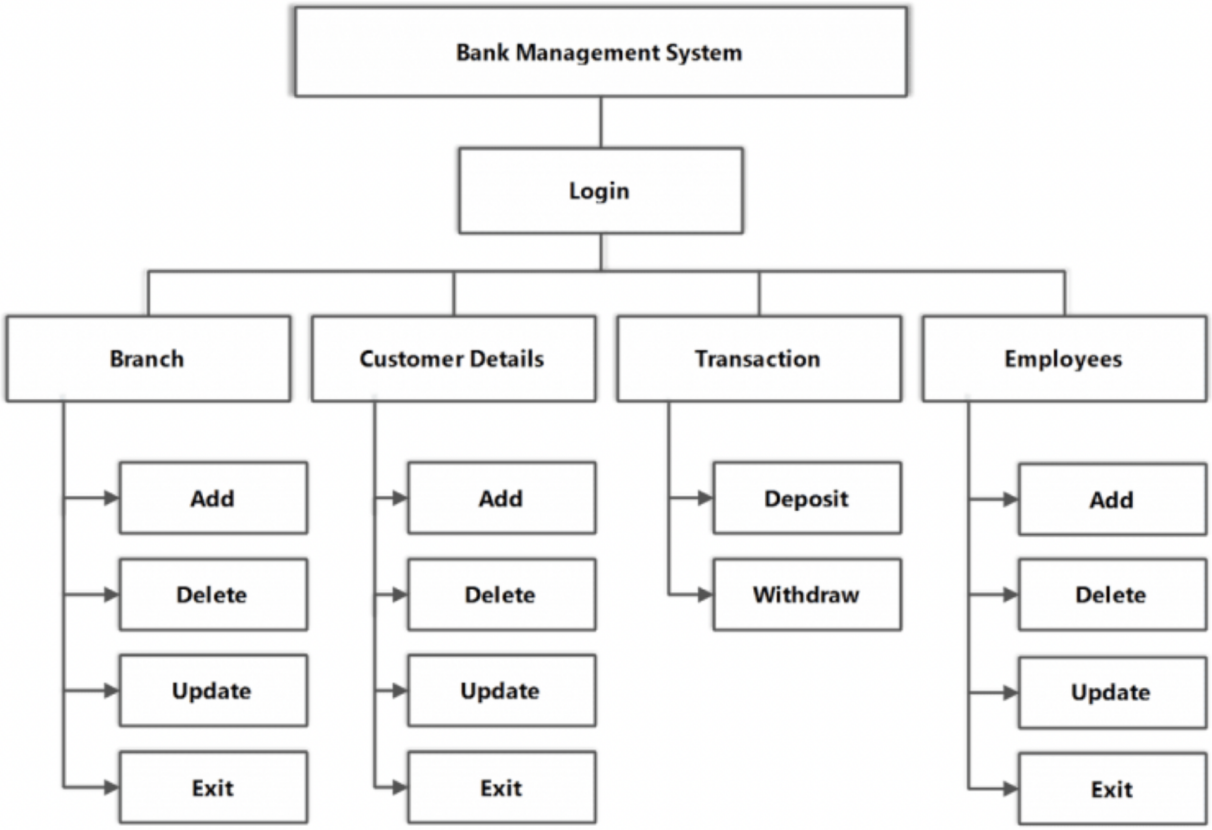
- Python (front end)
- Frameworks-Django
- Libraries-Tkinters,Pyside.
- MS Word 97 or later
- Web Browser: Microsoft Internet Explorer, Mozilla, Google Chrome and Safari.
- MySQL Server (back-end)
- Operating System: Windows 7,8,10 / Windows11/ MAC-OS.

Module Description

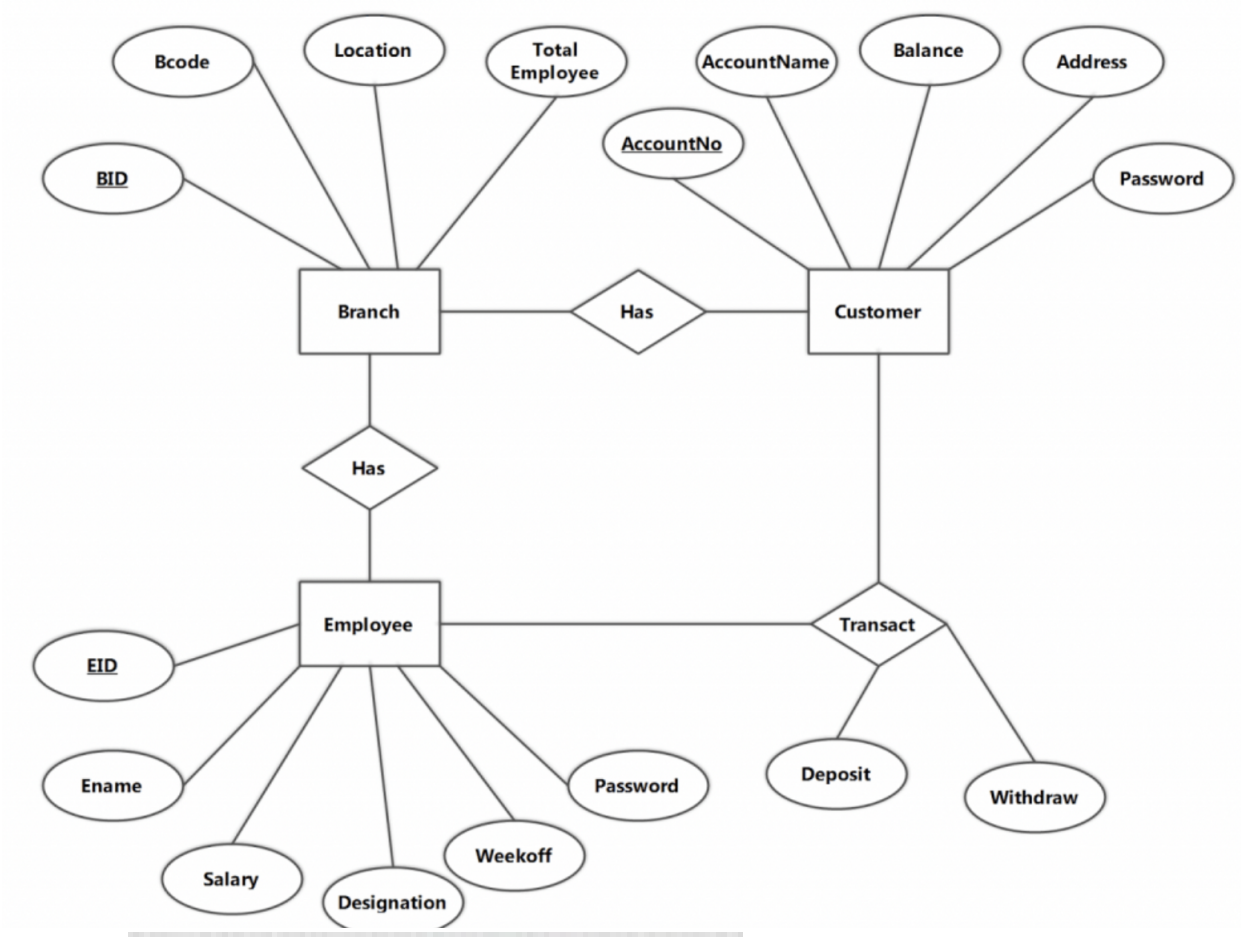
The module description of the Bank Management System project . This Module is developed by Python source code and My SQL as a database.

1. **Create New Account:** A customer who having the account in the world can create a virtual account through this module. This module receives the customer profile details and the bank account details with the proof of the ownership of the bank account.
2. **Login:** Virtual account holders can login in to the system using this module. Thus this is the secured login page for the customers in the website.
3. **Virtual Account:** After the approval of new virtual account creation, the customer assigned a unique virtual account number to make the online money transactions. This module views the details of the logged customer's virtual account.
4. **Bank Accounts:** A customer may have more than one bank account in various banks, in this case, the customer prompted to decide which bank account should reflect in the account debit or amount credit. For these operations customers can add their owned bank accounts here and it will be approved by the administrations of the system.
5. **Fund Transfer:** This is the module to make fund transfer to the virtual bank account holders or the usual bank account holders from the customer's specified bank account.
6. **Beneficiary:** Beneficiary is a person who receives money. Here the customer can add the beneficiaries to make fund transfer in the future.
7. **Transactions:** This module displays the transactions made by the customer in the Particular date with transaction ID.

Working Model-



ER Diagram-



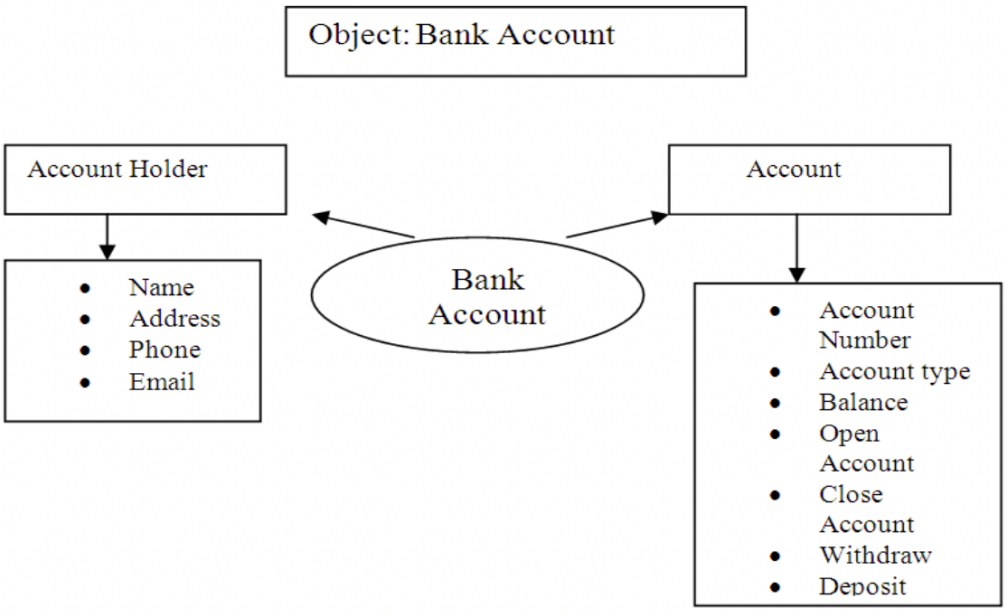
There are other features and actions that can be performed on a back account but we are not going to look at bank accounts in their entirety only the basics, this way we avoid over complicating the exercise. The purpose of this whole exercise is to show the usefulness of object oriented programming as opposed to really wanting to create a banking system.

Method

- We need to be able to generate an account number
- Account types: Savings or Current Account
- Maintain/update Balance
- Open/Close Account
- Withdraw/Deposit

The next thing we need to look at is where to store the information about the account. Obviously, the best place to store information relating to bank accounts is in a database. To work with a database (from an OOP point of view) will require the following methods:

- Connecting to the database
- Inserting account details
- Updating the balance on any withdrawal or deposits made



2.3.Administrative Modules

Here in my project there are two types of modules. This module is the main module which performs all the main operations in the system. The major operations in the system are:

2.4.Admin Module

Admin can access this project there is an authorization process. If you login as an Admin then you will be redirected to the Admin Home Page and if you are a simple user you will be redirected to your Account Home Page. This performs the following functions: Create Individual Accounts, Manage existing accounts, View all transactions, Balance enquiry, Delete/close account etc.

- ✓ Admin login
- ✓ Add/delete/update account
- ✓ Withdrawal/deposit/statements transaction
- ✓ Account Information
- ✓ User details list
- ✓ Active/Inactive account
- ✓ View transaction histories

2.5.User Module

A simple user can access their account and can deposit/withdraw money from their account. User can also transfer money from their account to any other bank account. User can see their transaction report and balance enquiry too.

- ➢ User login, use PIN system
- ➢ Creating/open new account registration
- ➢ Funds transfer (local/international/domestic)
- ➢ View statements transaction
- ➢ User account details
- ➢ Change Password and PIN

3.1.System Design

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term "design" is defined as "the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization". It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel.

System design goes through two phases of development:

- ✓ Logical Design and
- ✓ Physical Design.

3.2.Logical Design

The logical flow of a system and define the boundaries of a system. It includes the following steps:

- ✓ Reviews the current physical system – its data flows, file content, volumes, frequencies etc.
- ✓ Prepares output specifications – that is, determines the format, content and frequency of reports.
- ✓ Prepares input specifications – format, content and most of the input functions.
- ✓ Prepares edit, security and control specifications.
- ✓ Specifies the implementation plan.
- ✓ Prepares a logical design walk through of the information flow, output, input, controls and implementation plan.
- ✓ Reviews benefits, costs, target dates and system constraints.

3.3.Physical Design

Physical system produces the working systems by define the design specifications that tell the programmers exactly what the candidate system must do. It includes the following steps.

- ✓ Design the physical system.
- ✓ Specify input and output media.
- ✓ Design the database and specify backup procedures.
- ✓ Design physical information flow through the system and a physical design Walk through.
- ✓ Plan system implementation.
- ✓ Prepare a conversion schedule and target date.
- ✓ Determine training procedures, courses and timetable.
- ✓ Devise a test and implementation plan and specify any new hardware/software.
- ✓ Update benefits, costs, and conversion date and system constraints.

3.4.Database design

The database, called a bank, will have two tables, one called accounts and the other called customer. Each will hold information about either the account or the customer. The two tables will be linked through a foreign key. The customer table has the following fields:

Account User Table-3.1

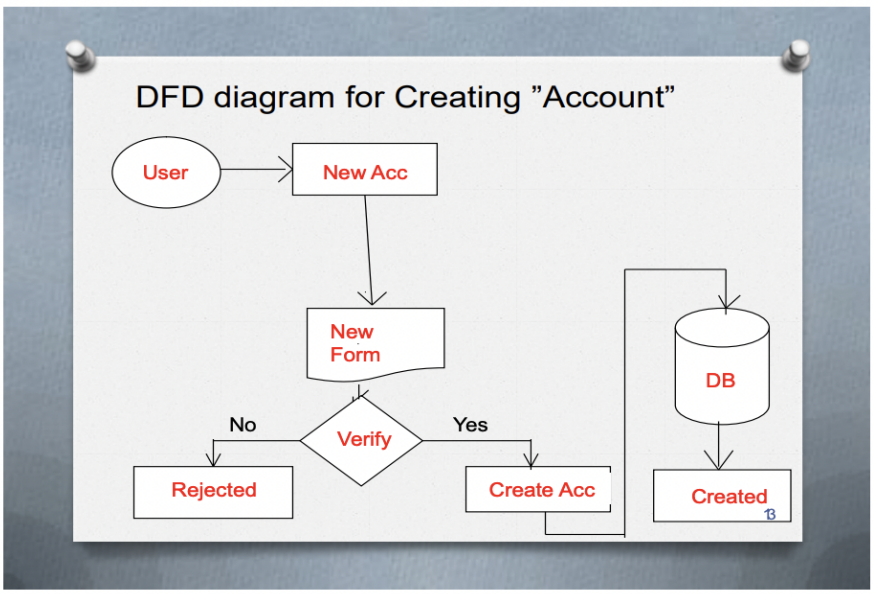
Field	Description
cusid	Creates a unique customer id for each new customer
name	Stores the customer name
address	Stores the customer address
acc_id	Links the customer to a account in the accounts table

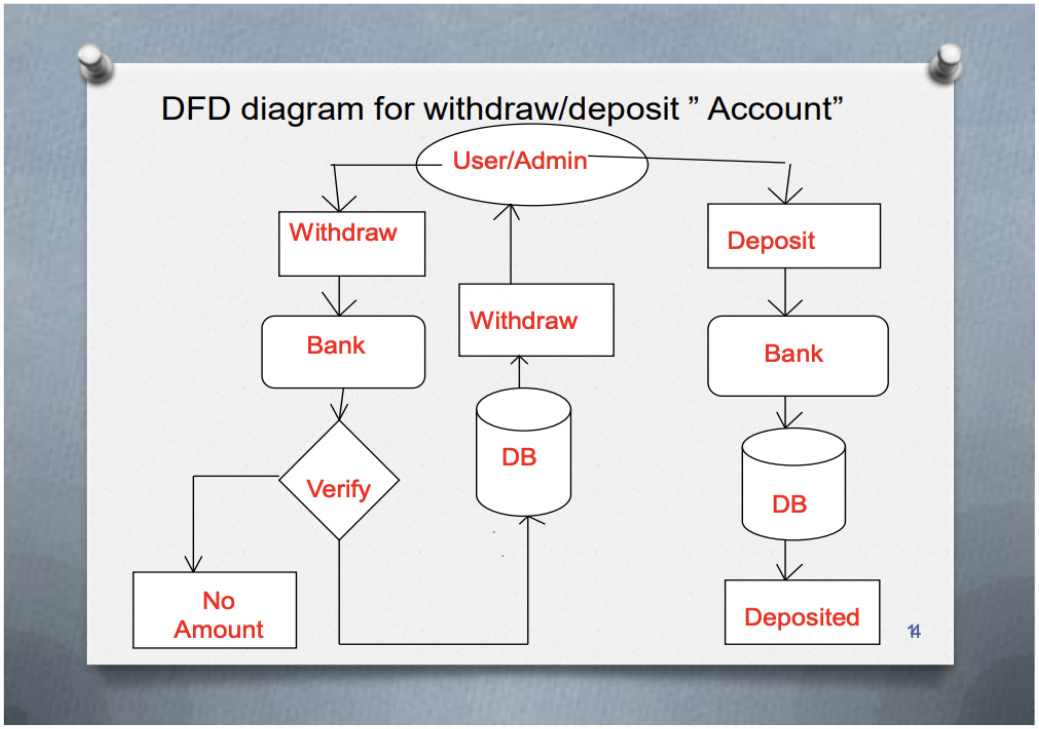
Accounts Table-3.2

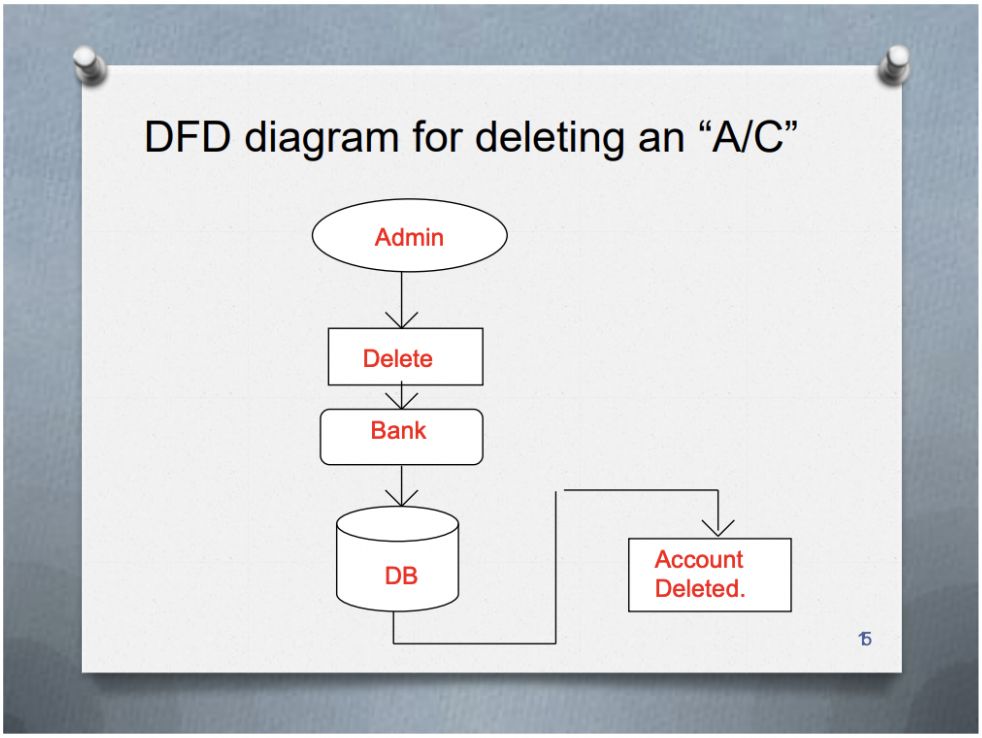
Field	Description
accid	Creates a unique account number for each new account
accno	Stores the account number
type	Stores the account type
balance	Stores the account balance
active	Shows the account status

Since one customer can have many accounts, I thought it only right to insert a foreign key acc_id into the customer table. In addition, instead of having fields such as date created and date closed, I simply use the active field to check if the account is active or not. This will enable us to focus more on the programming than on particulars of the database.

3.5.Data flow diagram




DFD diagram for withdraw/deposit " Account"
User/Admin
Withdraw
Bank
Verify
No Amount
Withdraw
DB
Deposit
Bank
DB
Deposited


DFD diagram for deleting an "A/C"
Admin
Delete
Bank
DB
Account Deleted.

4.1.General Information:

1. You should register for BAMS bank with the branch where you maintain the account.
2. If you maintain accounts at more than one branch, you need to register at each branch separately.
3. Normally BAMS Bank services will be open to the customer only after he/she acknowledges the receipt of password.
4. We invite you to visit your account on the site frequently for transacting business or viewing account balances. If you believe that any information relating to your account has a discrepancy, please bring it to the notice of the branch by e-mail or letter.
5. In a joint account, all account holders are entitled to register, as users of BAMS Bank, but transactions would be permitted based on the account operation rights recorded at the branch. (To begin with the services will be extended only to single or Joint "E or S" accounts only).
6. All accounts at the branch whether or not listed in the registration form, will be available on the BAMS Bank. However the applicant has the option to selectively view the accounts on the BAMS Bank.

4.2.Security terms:

1. The Branch where the customer maintains his/her account will assign:
    a) User Account Number &
    b) Password
2. The User-id and Password given by the branch must be replaced by User Name and Password of customer's choice at the time of first log-on. This is mandatory.
3. Bank will make reasonable use of available technology to ensure security and to prevent unauthorized access to any of these services. The BAMS Bank service is VERISIGN certified which guarantees, that it is a secure site. It means that
    - You are dealing with RR at that moment.
    - The two-way communication is secured with 128-bit SSL encryption technology, which ensures the confidentiality of the data during transmission.
4. You are welcome to access BAMS Bank from anywhere anytime. However, as a matter of precaution, customers may avoid using PCs with public access.

4.3.Banks terms:

1. All requests received from customers are logged for backend fulfillment and are effective from the time they are recorded at the branch.
2. Rules and regulations applicable to normal banking transactions in India will be applicable mutatis mutandis for the transactions executed through this site.
3. The BAMS Bank service cannot be claimed as a right. The bank may also convert this into a discretionary service anytime.
4. Dispute between the customer and the Bank in this service is subject to the jurisdiction of the courts in the Republic of India and governed by the laws prevailing in India.
5. The Bank reserves the right to modify the services offered or the Terms of service of BAMS Bank. The changes will be notified to the customers through a notification on the Site.

4.4.Customer's obligations

1. The customer has an obligation to maintain secrecy in regard to Username & Password registered with the Bank. The bank presupposes that login using valid Username and Password is a valid session initiated by none other than the customer.
2. Transaction executed through a valid session will be construed by RR to have emanated from the registered customer and will be binding on him/her.
3. The customer will not attempt or permit others to attempt accessing the BAMS Bank through any unlawful means.

4.6.Safe Online Banking Tips

- URL address on the address bar of your internet browser begins with "https"; the letter's at the end of "https" means 'secured'.
- Look for the padlock symbol either in the address bar or the status bar (mostly in the address bar) but not within the web page display area. Verify the security certificate by clicking on the padlock.
- Do not enter login or other sensitive information in any pop up window.
- The address bar has turned to green indicating that the site is secured with an SSL Certificate.

4.7.Beware of Phishing Attacks

- Phishing is a fraudulent attempt, usually made through email, phone calls, SMS etc seeking your personal and confidential information.
- State Bank or any of its representatives never sends you email/SMS or calls you over phone to get your personal information, password or one time SMS (high security) password.
- Any such e-mail/SMS or phone call is an attempt to fraudulently withdraw money from your account through Internet Banking. Never respond to such email/SMS or phone call.
- Change your Internet Banking password at periodical intervals.
- Always check the last log-in date and time in the post login page.

CODE

```
import tkinter as tk
from tkinter import messagebox
from time import gmtime, strftime


def is_number(s):
   try:
       float(s)
       return 1
   except ValueError:
       return 0

def check_acc_nmb(num):
   try:
       fpin=open(num+".txt",'r')
   except FileNotFoundError:
       messagebox.showinfo("Error","Invalid Credentials!\nTry Again!")
       return 0
   fpin.close()
   return

def home_return(master):
   master.destroy()
   Main_Menu()

def write(master,name,oc,pin):

   if( (is_number(name)) or (is_number(oc)==0) or (is_number(pin)==0)or name==""):
       messagebox.showinfo("Error","Invalid Credentials\nPlease try again.")
       master.destroy()
       return

   f1=open("Accnt_Record.txt",'r')
   accnt_no=int(f1.readline())
   accnt_no+=1
   f1.close()
```

```
    f1=open("Accnt_Record.txt",'w')
    f1.write(str(accnt_no))
    f1.close()

    fdet=open(str(accnt_no)+".txt","w")
    fdet.write(pin+"\n")
    fdet.write(oc+"\n")
    fdet.write(str(accnt_no)+"\n")
    fdet.write(name+"\n")
    fdet.close()

    frec=open(str(accnt_no)+"-rec.txt",'w')
    frec.write("Date                                Credit      Debit     Balance\n")
    frec.write(str(strftime("[%Y-%m-%d] [%H:%M:%S]  ",gmtime()))+"     "+oc+"
"+oc+"\n")
    frec.close()

    messagebox.showinfo("Details","Your Account Number is:"+str(accnt_no))
    master.destroy()
    return

def crdt_write(master,amt,accnt,name):

    if(is_number(amt)==0):
        messagebox.showinfo("Error","Invalid Credentials\nPlease try again.")
        master.destroy()
        return

    fdet=open(accnt+".txt",'r')
    pin=fdet.readline()
    camt=int(fdet.readline())
    fdet.close()
    amti=int(amt)
    cb=amti+camt
    fdet=open(accnt+".txt",'w')
    fdet.write(pin)
    fdet.write(str(cb)+"\n")
    fdet.write(accnt+"\n")
    fdet.write(name+"\n")
    fdet.close()
    frec=open(str(accnt)+"-rec.txt",'a+')
```

```
    frec.write(str(strftime("[%Y-%m-%d] [%H:%M:%S]   ",gmtime()))+"      "+str(amti)+"
"+str(cb)+"\n")
    frec.close()
    messagebox.showinfo("Operation Successfull!!","Amount Credited Successfully!!")
    master.destroy()
    return

def debit_write(master,amt,accnt,name):

    if(is_number(amt)==0):
        messagebox.showinfo("Error","Invalid Credentials\nPlease try again.")
        master.destroy()
        return

    fdet=open(accnt+".txt",'r')
    pin=fdet.readline()
    camt=int(fdet.readline())
    fdet.close()
    if(int(amt)>camt):
        messagebox.showinfo("Error!!","You dont have that amount left in your
account\nPlease try again.")
    else:
        amti=int(amt)
        cb=camt-amti
        fdet=open(accnt+".txt",'w')
        fdet.write(pin)
        fdet.write(str(cb)+"\n")
        fdet.write(accnt+"\n")
        fdet.write(name+"\n")
        fdet.close()
        frec=open(str(accnt)+"-rec.txt",'a+')
        frec.write(str(strftime("[%Y-%m-%d] [%H:%M:%S]   ",gmtime()))+"      "+"
"+str(amti)+"                "+str(cb)+"\n")
        frec.close()
        messagebox.showinfo("Operation Successfull!!","Amount Debited Successfully!!")
        master.destroy()
        return

def Cr_Amt(accnt,name):
    creditwn=tk.Tk()
    creditwn.geometry("600x300")
    creditwn.title("Credit Amount")
```

```
    creditwn.configure(bg="orange")
    fr1=tk.Frame(creditwn,bg="blue")
    l_title=tk.Message(creditwn,text="UNITED
BANK",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="center
",anchor="center")
    l_title.config(font=("Courier","50","bold"))
    l_title.pack(side="top")
    l1=tk.Label(creditwn,relief="raised",text="Enter Amount to be credited: ")
    e1=tk.Entry(creditwn,relief="raised")
    l1.pack(side="top")
    e1.pack(side="top")

b=tk.Button(creditwn,text="Credit",relief="raised",command=lambda:crdt_write(creditwn,
e1.get(),accnt,name))
    b.pack(side="top")
    creditwn.bind("<Return>",lambda x:crdt_write(creditwn,e1.get(),accnt,name))


def De_Amt(accnt,name):
    debitwn=tk.Tk()
    debitwn.geometry("600x300")
    debitwn.title("Debit Amount")
    debitwn.configure(bg="orange")
    fr1=tk.Frame(debitwn,bg="blue")
    l_title=tk.Message(debitwn,text="UNITED
BANK",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="center
",anchor="center")
    l_title.config(font=("Courier","50","bold"))
    l_title.pack(side="top")
    l1=tk.Label(debitwn,relief="raised",text="Enter Amount to be debited: ")
    e1=tk.Entry(debitwn,relief="raised")
    l1.pack(side="top")
    e1.pack(side="top")

b=tk.Button(debitwn,text="Debit",relief="raised",command=lambda:debit_write(debitwn,e1
.get(),accnt,name))
    b.pack(side="top")
    debitwn.bind("<Return>",lambda x:debit_write(debitwn,e1.get(),accnt,name))
```

```
def disp_bal(accnt):
    fdet=open(accnt+".txt",'r')
    fdet.readline()
    bal=fdet.readline()
    fdet.close()
    messagebox.showinfo("Balance",bal)




def disp_tr_hist(accnt):
    disp_wn=tk.Tk()
    disp_wn.geometry("900x600")
    disp_wn.title("Transaction History")
    disp_wn.configure(bg="orange")
    fr1=tk.Frame(disp_wn,bg="blue")
    l_title=tk.Message(disp_wn,text="UNITED
BANK",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="center
",anchor="center")
    l_title.config(font=("Courier","50","bold"))
    l_title.pack(side="top")
    fr1=tk.Frame(disp_wn)
    fr1.pack(side="top")
    l1=tk.Message(disp_wn,text="Your Transaction
History:",padx=100,pady=20,width=1000,bg="blue",fg="orange",relief="raised")
    l1.pack(side="top")
    fr2=tk.Frame(disp_wn)
    fr2.pack(side="top")
    frec=open(accnt+"-rec.txt",'r')
    for line in frec:
        l=tk.Message(disp_wn,anchor="w",text=line,relief="raised",width=2000)
        l.pack(side="top")
    b=tk.Button(disp_wn,text="Quit",relief="raised",command=disp_wn.destroy)
    b.pack(side="top")
    frec.close()

def logged_in_menu(accnt,name):
    rootwn=tk.Tk()
    rootwn.geometry("1600x500")
    rootwn.title("UNITED BANK-"+name)
    rootwn.configure(background='orange')
    fr1=tk.Frame(rootwn)
```

```
   fr1.pack(side="top")
   l_title=tk.Message(rootwn,text="SIMPLE BANKING\n
SYSTEM",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="cent
er",anchor="center")
   l_title.config(font=("Courier","50","bold"))
   l_title.pack(side="top")
   label=tk.Label(text="Logged in as:
"+name,relief="raised",bg="black",fg="white",anchor="center",justify="center")
   label.pack(side="top")
   img2=tk.PhotoImage(file="credit.gif")
   myimg2=img2.subsample(2,2)
   img3=tk.PhotoImage(file="debit.gif")
   myimg3=img3.subsample(2,2)
   img4=tk.PhotoImage(file="balance1.gif")
   myimg4=img4.subsample(2,2)
   img5=tk.PhotoImage(file="transaction.gif")
   myimg5=img5.subsample(2,2)
   b2=tk.Button(image=myimg2,command=lambda: Cr_Amt(accnt,name))
   b2.image=myimg2
   b3=tk.Button(image=myimg3,command=lambda: De_Amt(accnt,name))
   b3.image=myimg3
   b4=tk.Button(image=myimg4,command=lambda: disp_bal(accnt))
   b4.image=myimg4
   b5=tk.Button(image=myimg5,command=lambda: disp_tr_hist(accnt))
   b5.image=myimg5

   img6=tk.PhotoImage(file="logout.gif")
   myimg6=img6.subsample(2,2)
   b6=tk.Button(image=myimg6,relief="raised",command=lambda: logout(rootwn))
   b6.image=myimg6



   b2.place(x=100,y=150)
   b3.place(x=100,y=220)
   b4.place(x=900,y=150)
   b5.place(x=900,y=220)
   b6.place(x=500,y=400)


def logout(master):

   messagebox.showinfo("Logged Out","You Have Been Successfully Logged Out!!")
```

```
    master.destroy()
    Main_Menu()

def check_log_in(master,name,acc_num,pin):
    if(check_acc_nmb(acc_num)==0):
        master.destroy()
        Main_Menu()
        return

    if( (is_number(name))  or (is_number(pin)==0) ):
        messagebox.showinfo("Error","Invalid Credentials\nPlease try again.")
        master.destroy()
        Main_Menu()
    else:
        master.destroy()
        logged_in_menu(acc_num,name)



def log_in(master):
    master.destroy()
    loginwn=tk.Tk()
    loginwn.geometry("600x300")
    loginwn.title("Log in")
    loginwn.configure(bg="orange")
    fr1=tk.Frame(loginwn,bg="blue")
    l_title=tk.Message(loginwn,text="UNITED
BANK",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="center
",anchor="center")
    l_title.config(font=("Courier","50","bold"))
    l_title.pack(side="top")
    l1=tk.Label(loginwn,text="Enter Name:",relief="raised")
    l1.pack(side="top")
    e1=tk.Entry(loginwn)
    e1.pack(side="top")
    l2=tk.Label(loginwn,text="Enter account number:",relief="raised")
    l2.pack(side="top")
    e2=tk.Entry(loginwn)
    e2.pack(side="top")
    l3=tk.Label(loginwn,text="Enter your PIN:",relief="raised")
    l3.pack(side="top")
    e3=tk.Entry(loginwn,show="*")
    e3.pack(side="top")
```

```
   b=tk.Button(loginwn,text="Submit",command=lambda:
check_log_in(loginwn,e1.get().strip(),e2.get().strip(),e3.get().strip()))
   b.pack(side="top")
   b1=tk.Button(text="HOME",relief="raised",bg="black",fg="white",command=lambda:
home_return(loginwn))
   b1.pack(side="top")
   loginwn.bind("<Return>",lambda
x:check_log_in(loginwn,e1.get().strip(),e2.get().strip(),e3.get().strip()))


def Create():

   crwn=tk.Tk()
   crwn.geometry("600x300")
   crwn.title("Create Account")
   crwn.configure(bg="orange")
   fr1=tk.Frame(crwn,bg="blue")
   l_title=tk.Message(crwn,text="UNITED
BANK",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="center
",anchor="center")
   l_title.config(font=("Courier","50","bold"))
   l_title.pack(side="top")
   l1=tk.Label(crwn,text="Enter Name:",relief="raised")
   l1.pack(side="top")
   e1=tk.Entry(crwn)
   e1.pack(side="top")
   l2=tk.Label(crwn,text="Enter opening credit:",relief="raised")
   l2.pack(side="top")
   e2=tk.Entry(crwn)
   e2.pack(side="top")
   l3=tk.Label(crwn,text="Enter desired PIN:",relief="raised")
   l3.pack(side="top")
   e3=tk.Entry(crwn,show="*")
   e3.pack(side="top")
   b=tk.Button(crwn,text="Submit",command=lambda:
write(crwn,e1.get().strip(),e2.get().strip(),e3.get().strip()))
   b.pack(side="top")
   crwn.bind("<Return>",lambda
x:write(crwn,e1.get().strip(),e2.get().strip(),e3.get().strip()))
   return
```

```
def Main_Menu():



   rootwn=tk.Tk()
   rootwn.geometry("1600x500")
   rootwn.title("UNITED Bank")
   rootwn.configure(background='orange')
   fr1=tk.Frame(rootwn)
   fr1.pack(side="top")
   bg_image = tk.PhotoImage(file ="pile1.gif")
   x = tk.Label  (image = bg_image)
   x.place(y=-400)
   l_title=tk.Message(text="SIMPLE BANKING\n
SYSTEM",relief="raised",width=2000,padx=600,pady=0,fg="white",bg="black",justify="cent
er",anchor="center")
   l_title.config(font=("Courier","50","bold"))
   l_title.pack(side="top")
   imgc1=tk.PhotoImage(file="new.gif")
   imglo=tk.PhotoImage(file="login.gif")
   imgc=imgc1.subsample(2,2)
   imglog=imglo.subsample(2,2)

   b1=tk.Button(image=imgc,command=Create)
   b1.image=imgc
   b2=tk.Button(image=imglog,command=lambda: log_in(rootwn))
   b2.image=imglog
   img6=tk.PhotoImage(file="quit.gif")
   myimg6=img6.subsample(2,2)

   b6=tk.Button(image=myimg6,command=rootwn.destroy)
   b6.image=myimg6
   b1.place(x=800,y=300)
   b2.place(x=800,y=200)
   b6.place(x=920,y=400)

   rootwn.mainloop()

Main_Menu()
```

Imported Library Code-

```
import pickle
import os
import pathlib
class Account :
   accNo = 0
   name = ''
   deposit=0
   type = ''

   def createAccount(self):
       self.accNo= int(input("Enter the account no : "))
       self.name = input("Enter the account holder name : ")
       self.type = input("Ente the type of account [C/S] : ")
       self.deposit = int(input("Enter The Initial amount(>=500 for Saving and >=1000
for current"))
       print("\n\n\nAccount Created")

   def showAccount(self):
       print("Account Number : ",self.accNo)
       print("Account Holder Name : ", self.name)
       print("Type of Account",self.type)
       print("Balance : ",self.deposit)

   def modifyAccount(self):
       print("Account Number : ",self.accNo)
       self.name = input("Modify Account Holder Name :")
       self.type = input("Modify type of Account :")
       self.deposit = int(input("Modify Balance :"))

   def depositAmount(self,amount):
       self.deposit += amount

   def withdrawAmount(self,amount):
       self.deposit -= amount

   def report(self):
       print(self.accNo, " ",self.name ," ",self.type," ", self.deposit)

   def getAccountNo(self):
       return self.accNo
```

```
    def getAcccountHolderName(self):
        return self.name
    def getAccountType(self):
        return self.type
    def getDeposit(self):
        return self.deposit


def intro():
    print("\t\t\t\t**********************")
    print("\t\t\t\tBANK MANAGEMENT SYSTEM")
    print("\t\t\t\t**********************")

    print("\t\t\t\tBrought To You By:")
    print("\t\t\t\tprojectworlds.in")
    input()



def writeAccount():
    account = Account()
    account.createAccount()
    writeAccountsFile(account)

def displayAll():
    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
        mylist = pickle.load(infile)
        for item in mylist :
            print(item.accNo," ", item.name, " ",item.type, " ",item.deposit )
        infile.close()
    else :
        print("No records to display")


def displaySp(num):
    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
        mylist = pickle.load(infile)
        infile.close()
```

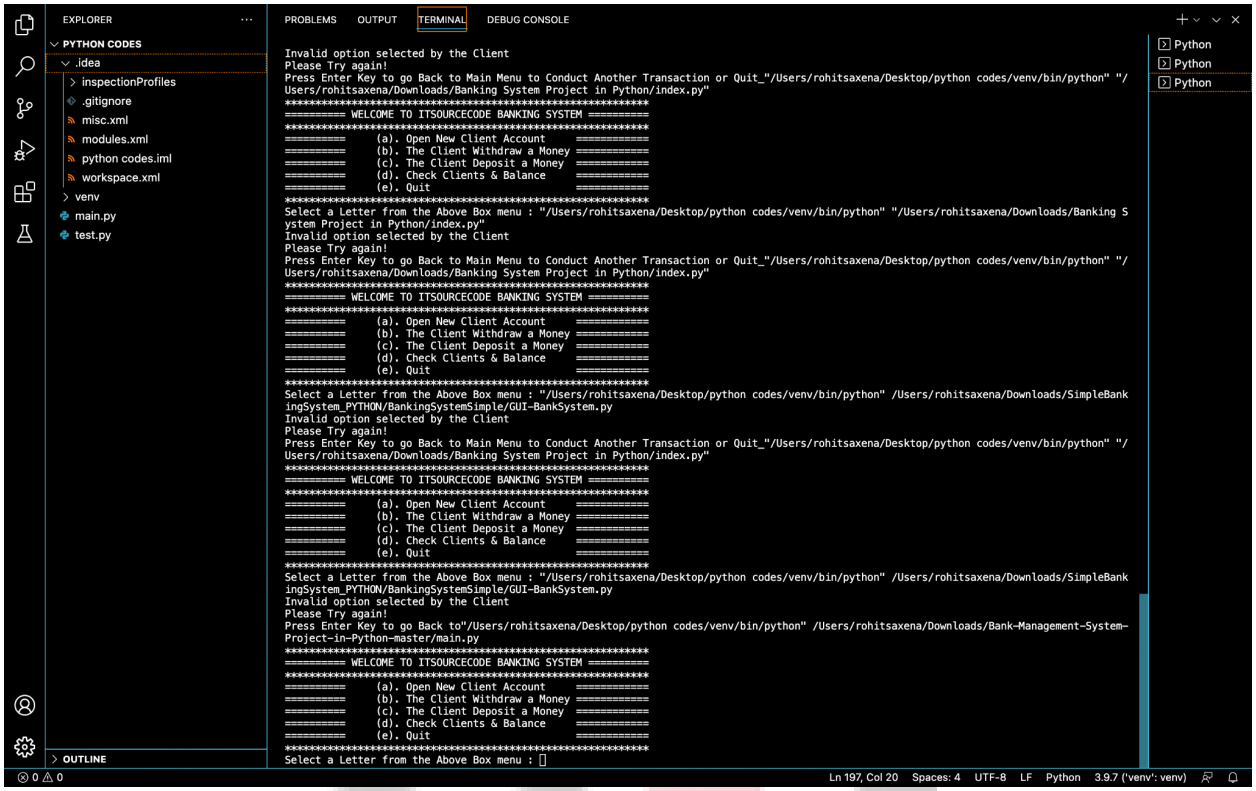
```
        found = False
        for item in mylist :
            if item.accNo == num :
                print("Your account Balance is = ",item.deposit)
                found = True
    else :
        print("No records to Search")
    if not found :
        print("No existing record with this number")

def depositAndWithdraw(num1,num2):
    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
        mylist = pickle.load(infile)
        infile.close()
        os.remove('accounts.data')
        for item in mylist :
            if item.accNo == num1 :
                if num2 == 1 :
                    amount = int(input("Enter the amount to deposit : "))
                    item.deposit += amount
                    print("Your account is updted")
                elif num2 == 2 :
                    amount = int(input("Enter the amount to withdraw : "))
                    if amount <= item.deposit :
                        item.deposit -=amount
                    else :
                        print("You cannot withdraw larger amount")

    else :
        print("No records to Search")
    outfile = open('newaccounts.data','wb')
    pickle.dump(mylist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')


def deleteAccount(num):
    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
```

```
        oldlist = pickle.load(infile)
        infile.close()
        newlist = []
        for item in oldlist :
            if item.accNo != num :
                newlist.append(item)
        os.remove('accounts.data')
        outfile = open('newaccounts.data','wb')
        pickle.dump(newlist, outfile)
        outfile.close()
        os.rename('newaccounts.data', 'accounts.data')

def modifyAccount(num):
    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
        oldlist = pickle.load(infile)
        infile.close()
        os.remove('accounts.data')
        for item in oldlist :
            if item.accNo == num :
                item.name = input("Enter the account holder name : ")
                item.type = input("Enter the account Type : ")
                item.deposit = int(input("Enter the Amount : "))

        outfile = open('newaccounts.data','wb')
        pickle.dump(oldlist, outfile)
        outfile.close()
        os.rename('newaccounts.data', 'accounts.data')


def writeAccountsFile(account) :

    file = pathlib.Path("accounts.data")
    if file.exists ():
        infile = open('accounts.data','rb')
        oldlist = pickle.load(infile)
        oldlist.append(account)
        infile.close()
        os.remove('accounts.data')
    else :
        oldlist = [account]
```

```
    outfile = open('newaccounts.data','wb')
    pickle.dump(oldlist, outfile)
    outfile.close()
    os.rename('newaccounts.data', 'accounts.data')


# start of the program
ch=''
num=0
intro()

while ch != 8:
    #system("cls");
    print("\tMAIN MENU")
    print("\t1. NEW ACCOUNT")
    print("\t2. DEPOSIT AMOUNT")
    print("\t3. WITHDRAW AMOUNT")
    print("\t4. BALANCE ENQUIRY")
    print("\t5. ALL ACCOUNT HOLDER LIST")
    print("\t6. CLOSE AN ACCOUNT")
    print("\t7. MODIFY AN ACCOUNT")
    print("\t8. EXIT")
    print("\tSelect Your Option (1-8) ")
    ch = input()
    #system("cls");

    if ch == '1':
        writeAccount()
    elif ch =='2':
        num = int(input("\tEnter The account No. : "))
        depositAndWithdraw(num, 1)
    elif ch == '3':
        num = int(input("\tEnter The account No. : "))
        depositAndWithdraw(num, 2)
    elif ch == '4':
        num = int(input("\tEnter The account No. : "))
        displaySp(num)
    elif ch == '5':
        displayAll();
    elif ch == '6':
        num =int(input("\tEnter The account No. : "))
        deleteAccount(num)
```

```
elif ch == '7':
    num = int(input("\tEnter The account No. : "))
    modifyAccount(num)
elif ch == '8':
    print("\tThanks for using bank managemnt system")
    break
else :
    print("Invalid choice")

ch = input("Enter your choice : ")
```

OUTPUT-


EXPLORER
PYTHON CODES
.idea
inspectionProfiles
.gitignore
misc.xml
modules.xml
python codes.iml
workspace.xml
venv
main.py
test.py
OUTLINE
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
Python
Python
Python
Invalid option selected by the Client
Please Try again!
Press Enter Key to go Back to Main Menu to Conduct Another Transaction or Quit_"/Users/rohitsaxena/Desktop/python codes/venv/bin/python" "/Users/rohitsaxena/Downloads/Banking System Project in Python/index.py"
********** WELCOME TO ITSOURCECODE BANKING SYSTEM **********
(a). Open New Client Account
(b). The Client Withdraw a Money
(c). The Client Deposit a Money
(d). Check Clients & Balance
(e). Quit
Select a Letter from the Above Box menu : "/Users/rohitsaxena/Desktop/python codes/venv/bin/python" "/Users/rohitsaxena/Downloads/Banking System Project in Python/index.py"
Invalid option selected by the Client
Please Try again!
Press Enter Key to go Back to Main Menu to Conduct Another Transaction or Quit_"/Users/rohitsaxena/Desktop/python codes/venv/bin/python" "/Users/rohitsaxena/Downloads/Banking System Project in Python/index.py"
********** WELCOME TO ITSOURCECODE BANKING SYSTEM **********
(a). Open New Client Account
(b). The Client Withdraw a Money
(c). The Client Deposit a Money
(d). Check Clients & Balance
(e). Quit
Select a Letter from the Above Box menu : "/Users/rohitsaxena/Desktop/python codes/venv/bin/python" /Users/rohitsaxena/Downloads/SimpleBankingSystem_PYTHON/BankingSystemSimple/GUI-BankSystem.py
Invalid option selected by the Client
Please Try again!
Press Enter Key to go Back to Main Menu to Conduct Another Transaction or Quit_"/Users/rohitsaxena/Desktop/python codes/venv/bin/python" "/Users/rohitsaxena/Downloads/Banking System Project in Python/index.py"
********** WELCOME TO ITSOURCECODE BANKING SYSTEM **********
(a). Open New Client Account
(b). The Client Withdraw a Money
(c). The Client Deposit a Money
(d). Check Clients & Balance
(e). Quit
Select a Letter from the Above Box menu : "/Users/rohitsaxena/Desktop/python codes/venv/bin/python" /Users/rohitsaxena/Downloads/SimpleBankingSystem_PYTHON/BankingSystemSimple/GUI-BankSystem.py
Invalid option selected by the Client
Please Try again!
Press Enter Key to go Back to"/Users/rohitsaxena/Desktop/python codes/venv/bin/python" /Users/rohitsaxena/Downloads/Bank-Management-System-Project-in-Python-master/main.py
********** WELCOME TO ITSOURCECODE BANKING SYSTEM **********
(a). Open New Client Account
(b). The Client Withdraw a Money
(c). The Client Deposit a Money
(d). Check Clients & Balance
(e). Quit
Select a Letter from the Above Box menu :
Ln 197, Col 20
Spaces: 4
UTF-8
LF
Python
3.9.7 ('venv': venv)


Deshboard
Account No :
37098100000016
Enter
Clear
Name :
Danish Ali
Amount :
10000
Mobile No :
8534867764
Source Of Funds :
Salary
Vahical Sale
Property Sale
Others
Others :
Students
Scan
Cancel

DPES
Dhruv Patil Education Society