

---

# BANKING MANAGEMENT SYSTEM

---

By

Hamim Sheikh,	ID:20215103049
Rabeya Basri Sumona,	ID:20215103015
John Pritom Biswas,	ID:20215103032
Azizur Rahman Ankon,	ID:20215103104
Zarin Akter,	ID:20215103016

Submitted in partial fulfillment of the requirements of the degree of

**Bachelor of Science in**

**Computer Science and Engineering**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BANGLADESH UNIVERSITY OF BUSINESS AND TECHNOLOGY

October 2022

## Banking Management System

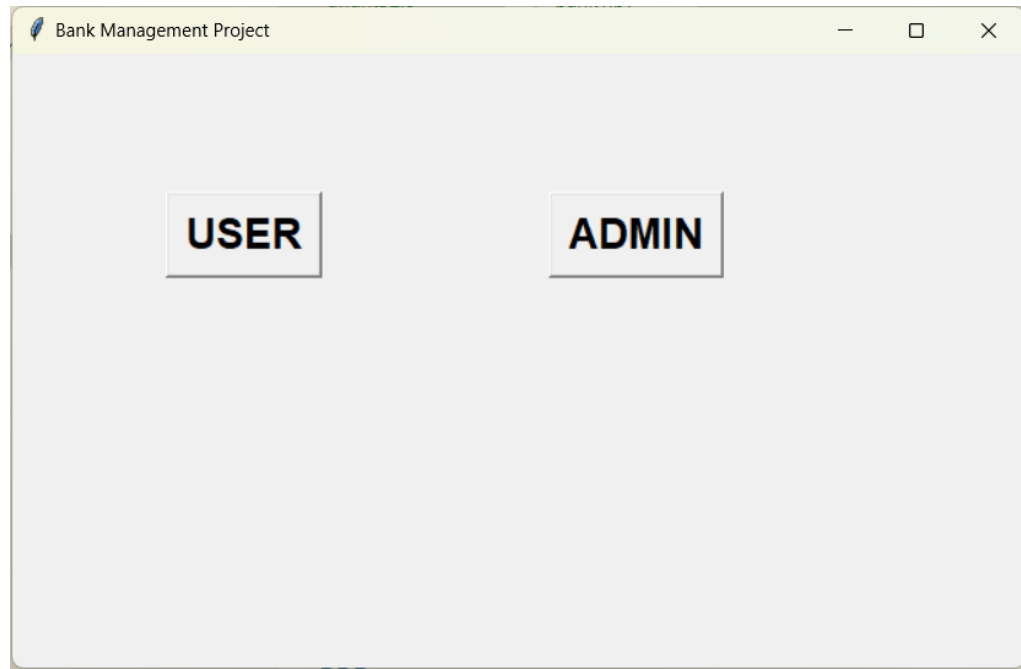


Figure 0.1.

## Declaration

We hereby declare that the project work entitled "BANKING MANAGEMENT SYSTEM" using Python programmer submitted to BANGLADESH UNIVERSITY OF BUSINESS TECHNOLOGY, is a record of an original work done by us under the guidance of our lecturer MD Shahiduzzaman. We think that this report has not been submitted previously to any other university for any examination.

Hamim Sheikh  
ID: 20215103049

---

Signature

Rabeya Basri Sumona  
ID: 20215103015

---

Signature

John Pritom Biswas  
ID: 20215103032

---

Signature

Azizur Rahman Ankon  
ID: 20215103104

---

Signature

Zarin Akter  
ID: 20215103016

---

Signature

## **Approval**

Firstly, We would like to thank our Course teacher MD Shahiduzzaman ,Lecturer,Deparment of Computer Science and Engineering ,Bangladesh University of Business and Technology(BUBT) and Our Project Team Member's for their enormous support and encouragement.

## Acknowledgement

We would like to express our heartly gratitude to the almighty Allah who offered upon our family and us kind care throughout this journey until the fulfillment of this project.

Firstly, We would like to thank our Course Teacher MD Shahiduzzaman ,Laecturer ,Deparment of Computer Science and Engineering ,Bangladesh University of Business and Technology(BUBT) and our Team Member's for their enormous support and encouragement.

we should also like to thank my Faculty In charge Ms. Sabiha Firdaus who helped clear any doubt what so ever we had to encounter while making this project.

Lastly, We should like to thank our Family for the tremendous amount of support they offered us while making this Project. we also were the invaluable support of our friends and colleagues which helped us in completing this Project.

## **Abstract**

The sole intention behind the consideration of this Project is to generate and manage a simple database for question. This project is developed considering "Banking Management".

## List of Figures

0.1	.....	ii
4.1	Flowchart .....	11
4.2	welcome page .....	38
4.3	admin page .....	39
4.4	User Login Page .....	39
4.5	Database .....	40
4.6	Database .....	41
4.7	Database .....	42
4.8	Account Creation Page .....	42
4.9	User Page .....	43

# Contents

<b>Declaration</b>	<b>iii</b>
<b>Approval</b>	<b>iv</b>
<b>Acknowledgement</b>	<b>v</b>
<b>Abstract</b>	<b>vi</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Abbreviations</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Problem Background . . . . .	3
1.4 Objectives . . . . .	3
1.5 Motivations . . . . .	4
<b>2 Background</b>	<b>5</b>
2.1 Introduction . . . . .	5
2.2 Existing System Analysis . . . . .	6
2.3 Feasibility Study . . . . .	6
2.3.1 Economic Feasibility . . . . .	6
2.3.2 Technical Feasibility . . . . .	6
2.3.3 Schedule Feasibility . . . . .	7
<b>3 Requirement Analysis</b>	<b>8</b>
3.1 Introduction . . . . .	8
3.2 Software Requirement . . . . .	8



3.3	Hardware Requirement . . . . .	8
<b>4</b>	<b>Implementation and Testing</b>	<b>10</b>
4.1	Introduction . . . . .	10
4.2	Flowchart . . . . .	11
4.3	System Setup . . . . .	12
4.4	Functionalities . . . . .	13
4.4.1	Function 1 . . . . .	13
4.4.2	Function 2 . . . . .	13
4.4.3	Function 3 . . . . .	13
4.4.4	Function 4 . . . . .	13
4.4.5	Function 5 . . . . .	14
4.4.6	Function 6 . . . . .	14
4.5	Implementation . . . . .	14
4.6	Testing . . . . .	38
<b>5</b>	<b>Impacts, Ethics, and Challenges</b>	<b>44</b>
5.1	Impacts (on Society) . . . . .	44
5.2	Ethics . . . . .	45
<b>6</b>	<b>Conclusion</b>	<b>46</b>
	<b>Future Work</b>	<b>48</b>

# **Introduction**

## **1.1 Introduction**

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. The primary aim of this “Bank Account Management System” is to provide an improved design methodology, which envisages the future expansion, and modification, which is necessary for a core sector like banking. This necessitates the design to be expandable and modifiable and so a modular approach is used in developing the application

software. Anybody who is an Account holder in this bank can become a member of Bank Account Management System. He has to fill a form with his personal details and Account Number. 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. Now a day's, managing a bank is tedious job up to certain limit. So software that reduces the work is essential. Also today's world is a genuine computer world and is getting faster and faster day-by-day. Thus, considering above necessities, the software for bank management has become necessary which would be useful in managing the bank more efficiently. All transactions are carried out online by transferring from accounts in the same Bank or international bank. The software is meant to overcome the drawbacks of the manual system. The software has been developed using the most powerful and secure backend MYSQL database and the most widely accepted web oriented as well as application oriented.

## **1.2 Problem Statement**

Bank Account Management System is a software developed to conduct an Bank based on time constraints. Bank System is accessed by entering

the username and password file which is added to the database. Before start of the system, the rules and regulations are displayed that includes descriptions.

## **1.3 Problem Background**

## **1.4 Objectives**

- Our motto is to develop a software program for managing the entire bank process related to Administration accounts customer accounts and to keep each every track about their property and their various transaction processes efficiently.

- Hereby, our main objective is the customer's satisfaction considering today's faster in the world.

- Client can do his operations comfortably without any risk or losing of his privacy.

- Our software will perform and fulfill all the tasks that any customer would desire.

- Client doesn't need to go to the bank to do small operation.

- It helps the customer to be satisfied and comfortable in his choices, this protection contains customer's account, money and his privacy.

- Help client transferring money to/or another bank or country.

## 1.5 Motivations

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 updation, 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 updation 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.

## **Background**

### **2.1 Introduction**

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.

## **2.2 Existing System Analysis**

The existing bank system is slow as every task is being performed by the human being and comparing the computer task speed with a computer is not fair. The complexity of this system is increased when an increase in the number of customers and with that there will be a number of transactions will be performed now everything needs to log in to a file for reference in the future which is simply not the kind of scenario we need at this time.

## **2.3 Feasibility Study**

The online banking system is economically feasible as it builds on an already existing system and the cost of hardware resources needed is relatively low, software applications needed are readily available making the project budget to be manageable.

### **2.3.1 Economic Feasibility**

No fund

### **2.3.2 Technical Feasibility**

Technical Feasibility includes existing and new Hardware and Software requirements that are required to operate the project on the platform VS Code. The basic Software requirement is VS Code and Python in which the front end of the Banking Management System project has been done.

### **2.3.3 Schedule Feasibility**

Adequate time availability and a lot of time in the advisor has given us all the opportunity to finish the PROJECT.



## **Requirement Analysis**

### **3.1 Introduction**

Python is a dynamic, interpreted (bytecode-compiled) language. There are no type declarations of variables, parameters, functions, or methods in source code. This makes the code short and flexible, and you lose the compile-time type checking of the source code.

### **3.2 Software Requirement**

Software Requirements Specification :

Front End/Language : Python/Database : MYSQL

Additional Tools : XAPM Server

Operating System : Windows 7, 8, 9, 10, XP.

### **3.3 Hardware Requirement**

Hardware Requirements Specification :

Processor : Intel Pentium III or later

Main Memory(RAM) : 256 MB

Cache Memory : 512 KB

Monitor : 14 inch Color Monitor

Keyboard : 108 Keys

Mouse : Optical Mouse

Hard Disk : 160 GB

## **Implementation and Testing**

### **4.1 Introduction**

Banking refers to a financial activity to manage and safeguard your hard-earned money. Banks cater to all sorts of individuals, small businesses, and large corporations. Banks offer financial management products, including various types of accounts and loans.

## 4.2 Flowchart

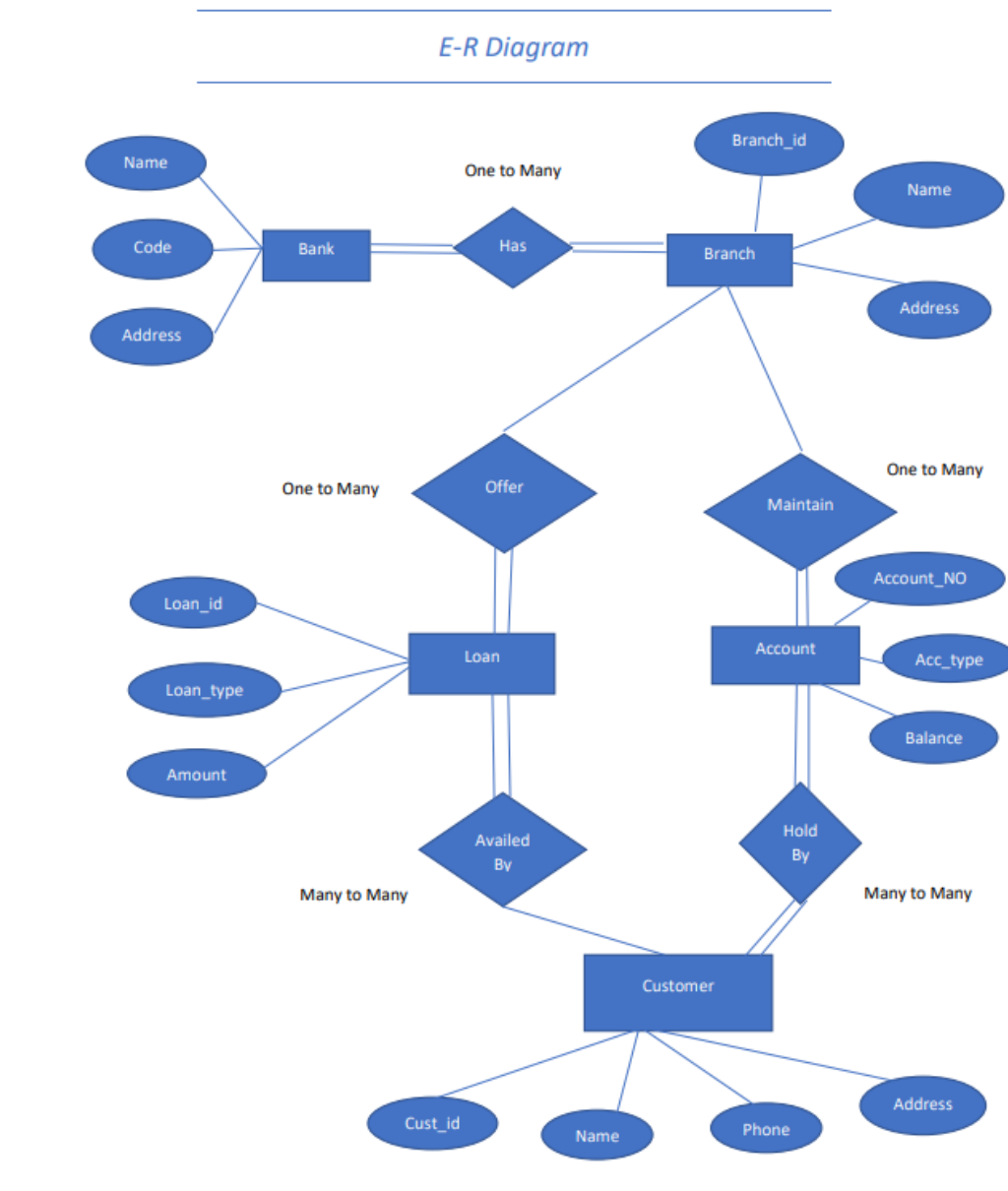


Figure 4.1. Flowchart

## 4.3 System Setup

An Intel Pentium 4 2.00 GHz CPU is required at a minimum to run It's System Time. However, the developers recommend a CPU greater or equal to an Intel Core i7-920 to run the System.

## 4.4 Functionalities

### 4.4.1 Function 1

in python we use `from tkinter import *` for (GUI) graphical user interface `Tk()`, `mainloop()`.

### 4.4.2 Function 2

we use `Frame(root)` for new page in GUI and `def show_frame(frame) :`  
`frame.tkraise()` for showing the pages.

The geometry of GUI is `geometry("900x400")` function to create a proper interface.

### 4.4.3 Function 3

for loop for using the pages i create

```
for frame in (page0, admin_page1, admin_page2, user_page1, user_page2, admin_page) :  
    frame.grid(row = 0, column = 0, sticky = 'nsew')
```

### 4.4.4 Function 4

By using `def insert()`: function we can insert user data in to Mysql database

`def delete()`: use for delete all the data we want to .

`def update()`: if we want to change any data from Mysql database.

`def get()`: find out any user in the sql database .

```
def pg2show() : for showing user information.
```

```
def apg2show() : for showing admin information.
```

#### 4.4.5 Function 5

```
Button(page0, text='USER', font=( 'Arial', 20, 'bold'), command=lambda:  
show_frame(user_page1))
```

executing all the button that we use to run our code properly.

```
Label(admin_page1, text = ' Username', font = ('Arial', 15, 'bold'))
```

show in admin page 1 the Username text.

```
Entry(admin_page1) for input data by any user.
```

place(x=50, y=100) for place text in the correct place we want for.

#### 4.4.6 Function 6

```
import tkinter.messagebox as MessageBox
```

```
import mysql.connector as mysql
```

messagebox for showing message for any ambiguity

and connector for connect with the Mysql.

### 4.5 Implementation

```
\documentclass{article}
```

```
from tkinter import *
```

```

import tkinter.messagebox as MessageBox
import mysql.connector as mysql

G_userid = ""
G_amount = ""
G_address = ""
G_phone = ""
G_name = ""

def insert():
    id = pg2_e_id.get()
    name = pg2_e_name.get()
    phone = pg2_e_phone.get()
    password = pg2_e_pass.get()
    amount = pg2_e_amount.get()
    address = pg2_e_address.get()
    if (id == "" or name == "" or phone == ""
or password == "" or amount == "" or address == ""):
        MessageBox.showinfo("Insert Status", "All fields are required")
    else:
        con = mysql.connect(host="localhost", user="root",
password="ankon20158",

database="test")
        cursor = con.cursor()

```



```

cursor.execute("insert into bankdata values('"+id+"', '"+
name+"', '"+phone+"', '"+password+"', '"+amount+"', '"+address+"')")
cursor.execute("commit")
pg2_e_id.delete(0, 'end')
pg2_e_name.delete(0, 'end')
pg2_e_phone.delete(0, 'end')
pg2_e_pass.delete(0, 'end')
pg2_e_amount.delete(0, 'end')
pg2_e_address.delete(0, 'end')
pg2.show()
MessageBox.showinfo("Insert Status", "Inserted Successfully")
con.close()

def admin_insert():
name = apg1_entry.get()
password = apg1_entry2.get()

if (name == "" or password == ""):
MessageBox.showinfo("Insert Status", "All fields are required")
else:
con = mysql.connect(host="localhost", user="root",
password="ankon20158", database="test")
cursor = con.cursor()
cursor.execute("insert into admin values('"+
name+"', '"+password+"')")

```

```
cursor.execute("commit")
```

```
apg1_entry.delete(0, 'end')
```

```
apg1_entry2.delete(0, 'end')
```

```
apg2_show()
```

```
MessageBox.showinfo("Insert Status", "Inserted Successfully")
```

```
con.close()
```

```
def delete():
```

```
id = pg2_e_id.get()
```

```
if (id == ""):
```

```
    MessageBox.showinfo("Delete Status","ID is compolsary for delete")
```

```
else:
```

```
    con = mysql.connect(host="localhost", user="root",
```

```
    password="ankon20158", database="test")
```

```
    cursor = con.cursor()
```

```
    cursor.execute("delete from bankdata where id='"+id+"'")
```

```
    cursor.execute("commit")
```

```
pg2_e_id.delete(0, 'end')
```

```
pg2_e_name.delete(0, 'end')
```

```
pg2_e_phone.delete(0, 'end')
```

```
pg2_e_pass.delete(0, 'end')
```

```
pg2_e_amount.delete(0, 'end')
```

```

pg2_e_address.delete(0, 'end')
pg2_show()
MessageBox.showinfo("Delete Status", "Deleted Successfully")
con.close()

def update():
    id = pg2_e_id.get()
    name = pg2_e_name.get()
    phone = pg2_e_phone.get()
    amount = pg2_e_amount.get()
    address = pg2_e_address.get()

    if (id == "" or name == "" or phone == "" or amount == ""):
        MessageBox.showinfo(
            "Update Status", "All fields are required except Password")
    else:
        con = mysql.connect(host="localhost", user="root",
            password="ankon20158", database="test")
        cursor = con.cursor()
        cursor.execute("update bankdata set name='"+name+"',phone='"+
            phone + "',amount='"+amount + "',
            address='"+address + "' where id='"+id+"'")
        cursor.execute("commit")

pg2_e_id.delete(0, 'end')

```

```

pg2_e_name.delete(0, 'end')
pg2_e_phone.delete(0, 'end')
pg2_e_pass.delete(0, 'end')
pg2_e_amount.delete(0, 'end')
pg2_e_address.delete(0, 'end')
pg2_show()
MessageBox.showinfo("Update Status", "Updated Successfully")
con.close()

```

```

def user_update():
    phone = upg2_e_phone.get()
    password = upg2_e_pass.get()
    address = upg2_e_address.get()

    if (phone == "" or password == "" or address == ""):
        MessageBox.showinfo("Update Status", "All fields are required")
    else:
        con = mysql.connect(host="localhost", user="root",
            password="ankon20158", database="test")
        cursor = con.cursor()
        cursor.execute("update bankdata set password='"+password +
            "' ,phone='"+phone + "' ,
            address='"+address + "' where id='"+G_userid+"'")
        cursor.execute("commit")

```

```

upg2_e_pass.delete(0, 'end')
upg2_e_phone.delete(0, 'end')
upg2_e_address.delete(0, 'end')
upg2_show()
MessageBox.showinfo("Update Status", "Updated Successfully")
con.close()

```

```

def user_withdraw():
    wd = upg2_e_trans.get()
    dpo = "0"

    if (wd == ""):
        MessageBox.showinfo("Update Status",
        "No amount written to withdraw")
    else:
        con = mysql.connect(host="localhost", user="root",
        password="ankon20158", database="test")
        cursor = con.cursor()
        cursor.execute("insert into transactions values('" +
        G_userid+"', '"+wd+"', '"+dpo+"')")
        cursor.execute("commit")

        upg2_e_trans.delete(0, 'end')
        global G_amount
        tmp = int(G_amount) - int(wd)

```

```

G_amount = str(tmp)
cursor.execute("update bankdata set amount=" +
G_amount + "' where id=" + G_userid + "'")
cursor.execute("commit")

upg2_show()
MessageBox.showinfo("Update Status",
"Withdrawn Successfully")
con.close()

def user_deposit():
dpo = upg2_e_trans.get()
wd = "0"

if (dpo == ""):
MessageBox.showinfo("Update Status",
"No amount written to deposit")
else:
con = mysql.connect(host="localhost", user="root",
password="ankon20158", database="test")
cursor = con.cursor()
cursor.execute("insert into transactions values(' +
G_userid + "', '" + wd + "', '" + dpo + "')")
cursor.execute("commit")

```

```

upg2_e_trans.delete(0, 'end')
global G_amount
tmp = int(G_amount) + int(dpo)
G_amount = str(tmp)
cursor.execute("update bankdata set amount=" +
G_amount + " ' where id=" + G_userid + " ")
cursor.execute("commit")

upg2_show()
MessageBox.showinfo("Update Status",
"deposited Successfully")
con.close()

def get():
id = pg2_e_id.get()
if (id == ""):
MessageBox.showinfo("Fetch Status",
"ID is compolsary to get data")
else:
con = mysql.connect(host="localhost", user="root",
password="ankon20158", database="test")
cursor = con.cursor()
cursor.execute("select * from bankdata where id=" + id + " ")
rows = cursor.fetchall()

```

```

for row in rows:
    pg2_e_name.insert(0, row[1])
    pg2_e_phone.insert(0, row[2])
    pg2_e_amount.insert(0, row[4])
    pg2_e_address.insert(0, row[5])

con.close()

def pg2_show():
    con = mysql.connect(host="localhost", user="root",
        password="ankon20158", database="test")
    cursor = con.cursor()
    cursor.execute("select * from bankdata")
    rows = cursor.fetchall()
    pg2_list.delete(0, pg2_list.size())

    for row in rows:
        insertData = 'ID:' + str(row[0]) + '   Name:' + row[1] + \
            '   Phone:' + row[2] + '   Amount:' + \
            str(row[4]) + '   Address:' + str(row[5])
        pg2_list.insert(pg2_list.size()+1, insertData)

    con.close()

```



```

def apg2_show():
    con = mysql.connect(host="localhost", user="root",
        password="ankon20158", database="test")
    cursor = con.cursor()
    cursor.execute("select * from admin")
    rows = cursor.fetchall()
    apg2_list.delete(0, apg2_list.size())
    cnt = 1

    for row in rows:
        insertData = str(cnt) + ': ' + row[0]
        apg2_list.insert(apg2_list.size()+1, insertData)
        cnt = cnt+1

    con.close()

```

```

def admin_check():
    pw = pg1_entry2.get()
    usr = pg1_entry.get()
    pg1_entry.delete(0, 'end')
    pg1_entry2.delete(0, 'end')
    con = mysql.connect(host="localhost", user="root",
        password="ankon20158", database="test")
    cursor = con.cursor()

```

```

cursor.execute("select * from admin")
rows = cursor.fetchall()
idfound = False
password_admin = ""
local_amount = ""

for row in rows:
    if (usr == str(row[0])):
        idfound = True
        password_user = row[1]

    if (idfound != True):
        MessageBox.showinfo("Login Status", "Invalid ID!")
    else:
        if (pw != password_user):
            MessageBox.showinfo("Login Status", "Invalid Password!")
        else:
            show_frame(admin_page_2)

con.close()

def user_check():
    pw = upg1_entry2.get()
    userid = upg1_entry.get()
    con = mysql.connect(host="localhost", user="root",

```

```

password="ankon20158", database="test")
cursor = con.cursor()
cursor.execute("select * from bankdata")
rows = cursor.fetchall()
idfound = False
password_user = ""
local_amount = ""
upg1_entry.delete(0, 'end')
upg1_entry2.delete(0, 'end')

for row in rows:
    if (userid == str(row[0])):
        idfound = True
        password_user = row[3]
        global G_userid
        G_userid = str(row[0])
        global G_amount
        G_amount = str(row[4])
        global G_name
        G_name = row[1]

insertName = "Welcome, " + G_name + "      "
upg2_welcome = Label(
    user_page_2, text=insertName, font=('bold', 15))
upg2_welcome.place(x=20, y=20)
upg2_amount1 = Label(

```

```

user_page_2 ,
text="Your Current Amount:" , font=('bold ' , 15))
upg2_amount1.place(x=600, y=90)
upg2_amount2 = Label(
user_page_2 , text=G.amount + "      " , font=('bold ' , 20))
upg2_amount2.place(x=650, y=120)
if (idfound != True):
MessageBox.showinfo("Login Status", "Invalid ID!")
else:
if (pw != password_user):
MessageBox.showinfo("Login Status", "Invalid Password!")
else:
show_frame(user_page_2)

con.close()

```

```

def upg2_show():
insertName = "Welcome, " + G_name + "      "
upg2_welcome = Label(
user_page_2 , text=insertName , font=('bold ' , 15))
upg2_welcome.place(x=20, y=20)
upg2_amount1 = Label(
user_page_2 , text="Your Current Amount:" ,
font=('bold ' , 15))
upg2_amount1.place(x=600, y=90)

```

```

upg2_amount2 = Label(
user_page_2 , text=G.amount + "      ",
font=('bold' , 20))
upg2_amount2.place(x=650, y=120)

```

```

root = Tk()
root.geometry("900x400")
root.title("Bank Management Project")
root.rowconfigure(0, weight=1)
root.columnconfigure(0, weight=1)

```

```

page0 = Frame(root)
admin_page_1 = Frame(root)
admin_page_2 = Frame(root)
user_page_1 = Frame(root)
user_page_2 = Frame(root)
c_admin_page = Frame(root)

```

```

for frame in (page0 , admin_page_1 ,
admin_page_2 , user_page_1 , user_page_2 , c_admin_page):
frame.grid(row=0, column=0, sticky='nsew')

```

```

def show_frame(frame):
    frame.tkraise()

show_frame(page0)

pg0_button = Button(page0, text='USER', font=(
    'Arial', 20, 'bold'),
    command=lambda: show_frame(user_page_1))
pg0_button.place(x=100, y=90)

pg0_button = Button(page0, text='ADMIN',
    font=(
    'Arial', 20, 'bold'),
    command=lambda: show_frame(admin_page_1))
pg0_button.place(x=350, y=90)

# ===== admin login page =====
pg1_label = Label(admin_page_1, text='Username',
    font=('Arial', 15, 'bold'))
pg1_label.place(x=50, y=100)

pg1_entry = Entry(admin_page_1)
pg1_entry.place(x=170, y=106)

```

```

pg1_label2 = Label(admin_page_1 , text='Password' ,
font=('Arial' , 15 , 'bold' ))
pg1_label2.place(x=50, y=150)

pg1_entry2 = Entry(admin_page_1)
pg1_entry2.place(x=170, y=155)

pg1_button = Button(admin_page_1 , text='LOGIN' , font=(
'Arial' , 13 , 'bold' ) , command=admin_check)
pg1_button.place(x=170, y=200)

apg2_button = Button(admin_page_1 , text='Back' , font=(
'Arial' , 13 , 'bold' ) , command=lambda: show_frame(page0))
apg2_button.place(x=750, y=350)

# ----- create admin page -----
apg1_label = Label(c_admin_page , text='Username' ,
font=('Arial' , 15 , 'bold' ))
apg1_label.place(x=50, y=100)

apg1_entry = Entry(c_admin_page)
apg1_entry.place(x=170, y=106)

apg1_label2 = Label(c_admin_page , text='Password' ,

```

```

font=('Arial ', 15, 'bold'))
apg1_label2.place(x=50, y=150)

apg1_entry2 = Entry(c_admin_page)
apg1_entry2.place(x=170, y=155)

apg2_list = Listbox(c_admin_page, width=70)
apg2_list.place(x=350, y=60)

apg1_button = Button(c_admin_page, text='Create',
font=(
'Arial ', 13, 'bold'), command=admin_insert)
apg1_button.place(x=170, y=200)

apg2_button = Button(c_admin_page, text='Back',
font=(
'Arial ', 13, 'bold'),
command=lambda: show_frame(admin_page_2))
apg2_button.place(x=750, y=350)

apg2.show()

#===== user login page =====
upg1_label = Label(user_page_1,
text='ID', font=('Arial ', 15, 'bold'))

```



```
upg1_label.place(x=50, y=100)
```

```
upg1_entry = Entry(user_page_1)
```

```
upg1_entry.place(x=170, y=106)
```

```
upg1_label2 = Label(user_page_1 ,  
text='Password ', font=('Arial ', 15, 'bold '))  
upg1_label2.place(x=50, y=150)
```

```
upg1_entry2 = Entry(user_page_1)  
upg1_entry2.place(x=170, y=155)
```

```
upg1_button = Button(user_page_1 , text='LOGIN', font=(  
'Arial ', 13, 'bold '), command=user_check)  
upg1_button.place(x=170, y=200)
```

```
apg2_button = Button(user_page_1 , text='Back', font=(  
'Arial ', 13, 'bold '),  
command=lambda: show_frame(page0))  
apg2_button.place(x=750, y=350)
```

```
# ===== admin Page 2 =====  
pg2_id = Label(admin_page_2 ,  
text='Enter ID', font=('bold ', 10))  
pg2_id.place(x=20, y=30)
```

```
pg2_name = Label(admin_page_2 ,  
text='Enter Name' , font=('bold ' , 10))  
pg2_name.place(x=20, y=60)
```

```
pg2_phone = Label(admin_page_2 ,  
text='Enter Phone' , font=('bold ' , 10))  
pg2_phone.place(x=20, y=90)
```

```
pg2_address = Label(admin_page_2 ,  
text='Enter Address ' , font=('bold ' , 10))  
pg2_address.place(x=20, y=120)
```

```
pg2_pass = Label(admin_page_2 ,  
text='Enter Password ' , font=('bold ' , 10))  
pg2_pass.place(x=20, y=150)
```

```
pg2_amount = Label(admin_page_2 ,  
text='Enter Amount' , font=('bold ' , 10))  
pg2_amount.place(x=20, y=180)
```

```
pg2_e_id = Entry(admin_page_2)  
pg2_e_id.place(x=150, y=30)
```

```
pg2_e_name = Entry(admin_page_2)  
pg2_e_name.place(x=150, y=60)
```

```
pg2_e_phone = Entry(admin_page_2)
```

```
pg2_e_phone.place(x=150, y=90)
```

```
pg2_e_address = Entry(admin_page_2)
```

```
pg2_e_address.place(x=150, y=120)
```

```
pg2_e_pass = Entry(admin_page_2)
```

```
pg2_e_pass.place(x=150, y=150)
```

```
pg2_e_amount = Entry(admin_page_2)
```

```
pg2_e_amount.place(x=150, y=180)
```

```
pg2_insert = Button(admin_page_2,
```

```
text='Insert ', font=(
```

```
"italic", 10), bg="white", command=insert)
```

```
pg2_insert.place(x=20, y=300)
```

```
pg2_delete = Button(admin_page_2,
```

```
text='Delete ', font=(
```

```
"italic", 10), bg="white", command=delete)
```

```
pg2_delete.place(x=70, y=300)
```

```
pg2_update = Button(admin_page_2,
```

```
text='Update ', font=(
```

```
"italic", 10), bg="white", command=update)
```

```
pg2_update.place(x=130, y=300)
```

```
pg2_get = Button(admin_page_2, text='Get', font=(  
"italic", 10), bg="white", command=get)  
pg2_get.place(x=190, y=300)
```

```
pg2_get = Button(admin_page_2,  
text='Create a new admin', font=(  
"italic", 10), bg="white",  
command=lambda: show_frame(c_admin_page))  
pg2_get.place(x=600, y=300)
```

```
pg2_logout = Button(admin_page_2,  
text='Logout', font=(  
"italic", 10), bg="white",  
command=lambda: show_frame(page0))  
pg2_logout.place(x=800, y=10)
```

```
pg2_list = Listbox(admin_page_2, width=90)  
pg2_list.place(x=290, y=60)  
pg2_show()
```

```
# ===== user Page 2 =====
```

```
upg2_phone = Label(user_page_2,  
text='Enter Phone', font=('bold', 10))
```

```
upg2_phone.place(x=20, y=140)
```

```
upg2_pass = Label(user_page_2 ,  
text='Enter Password ', font=('bold ', 10))  
upg2_pass.place(x=20, y=170)
```

```
upg2_address = Label(user_page_2 ,  
text='Enter Address ', font=('bold ', 10))  
upg2_address.place(x=20, y=200)
```

```
upg2_witdepo = Label(user_page_2 ,  
text='Withdraw/deposit ', font=('bold ', 10))  
upg2_witdepo.place(x=470, y=200)
```

```
upg2_toup = Label(user_page_2 ,  
text='To update your password ,  
address and phone number:', font=('bold ', 10))  
upg2_toup.place(x=20, y=110)
```

```
upg2_e_phone = Entry(user_page_2)  
upg2_e_phone.place(x=150, y=140)
```

```
upg2_e_pass = Entry(user_page_2)  
upg2_e_pass.place(x=150, y=170)
```

```
upg2_e_address = Entry(user_page_2)
```

```
upg2_e_address.place(x=150, y=200)
```

```
upg2_e_trans = Entry(user_page_2)
```

```
upg2_e_trans.place(x=600, y=200)
```

```
upg2_update = Button(user_page_2 ,  
text='Update' , font=(  
"italic" , 10), bg="white" , command=user_update)  
upg2_update.place(x=130, y=300)
```

```
upg2_withdraw = Button(user_page_2 ,  
text='Withdraw' , font=(  
"italic" , 10), bg="white" , command=user_withdraw)  
upg2_withdraw.place(x=600, y=230)
```

```
upg2_deposit = Button(user_page_2 ,  
text='Deposit' , font=(  
"italic" , 10), bg="white" , command=user_deposit)  
upg2_deposit.place(x=670, y=230)
```

```
upg2_logout = Button(user_page_2 ,  
text='Logout' , font=(  
"italic" , 10), bg="white" ,  
command=lambda: show_frame(page0))  
upg2_logout.place(x=750, y=10)
```

```
upg2_show()
```

```
root.mainloop()
```

## 4.6 Testing

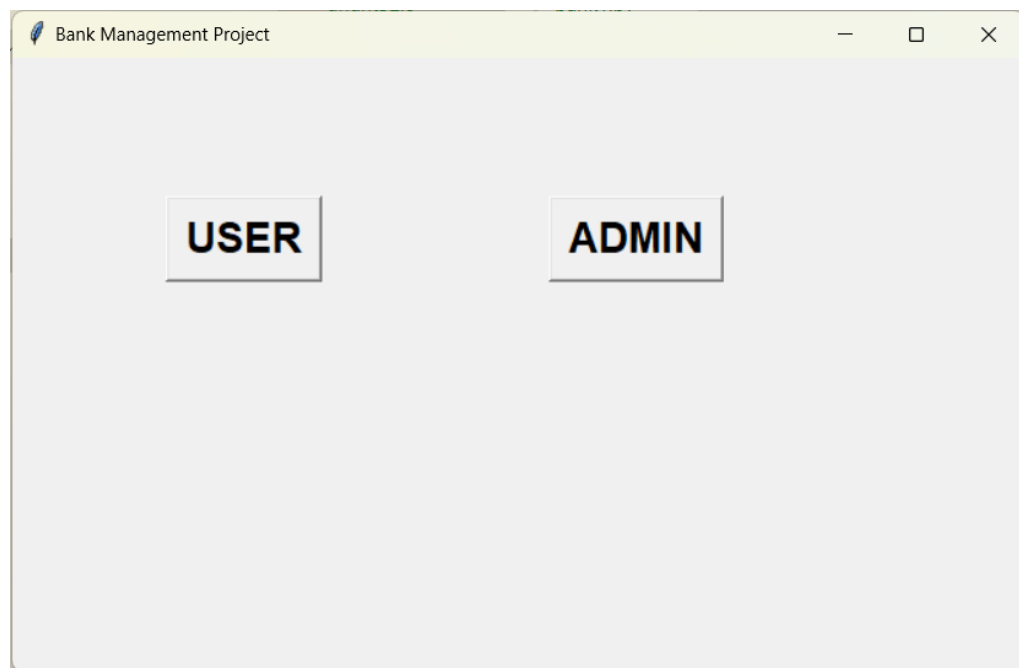
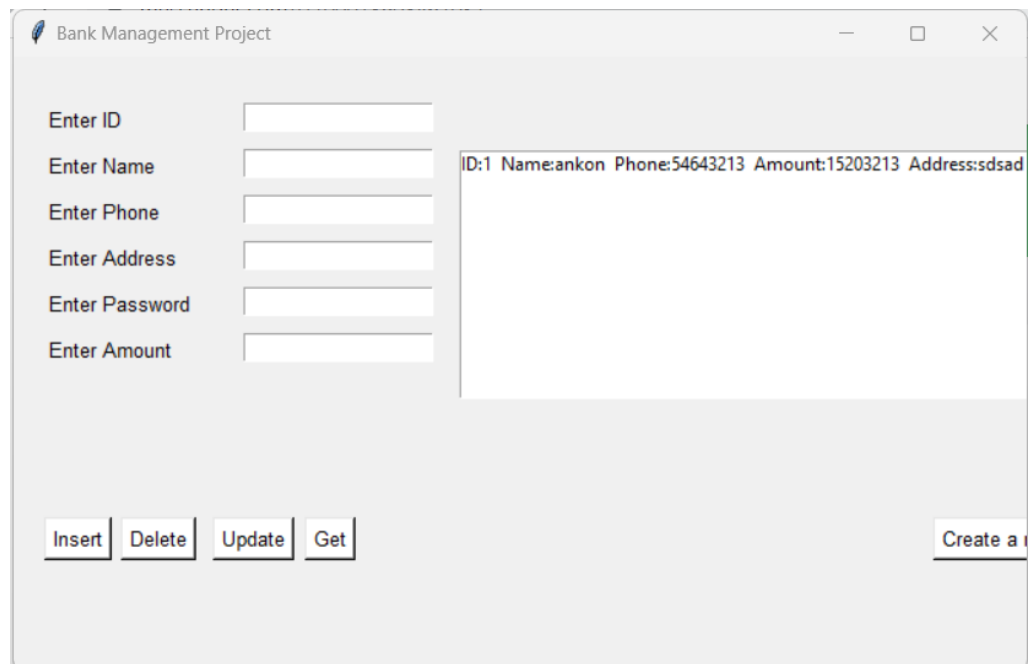


Figure 4.2. welcome page



Bank Management Project

Enter ID

Enter Name

Enter Phone

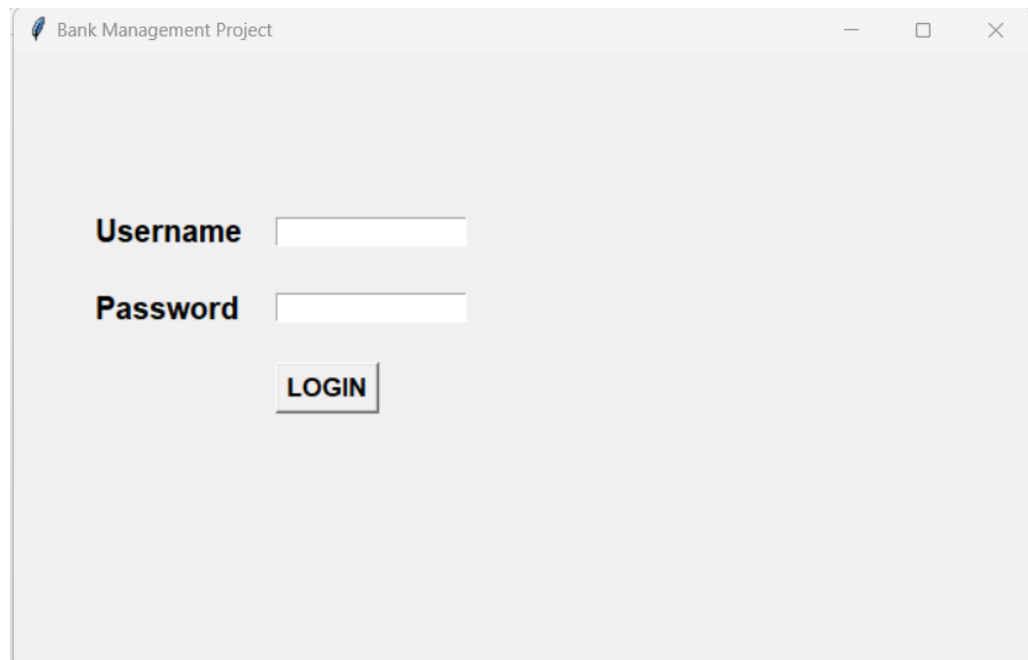
Enter Address

Enter Password

Enter Amount

ID:1 Name:ankon Phone:54643213 Amount:15203213 Address:sdsad

Figure 4.3. admin page



Bank Management Project

**Username**

**Password**

Figure 4.4. User Login Page



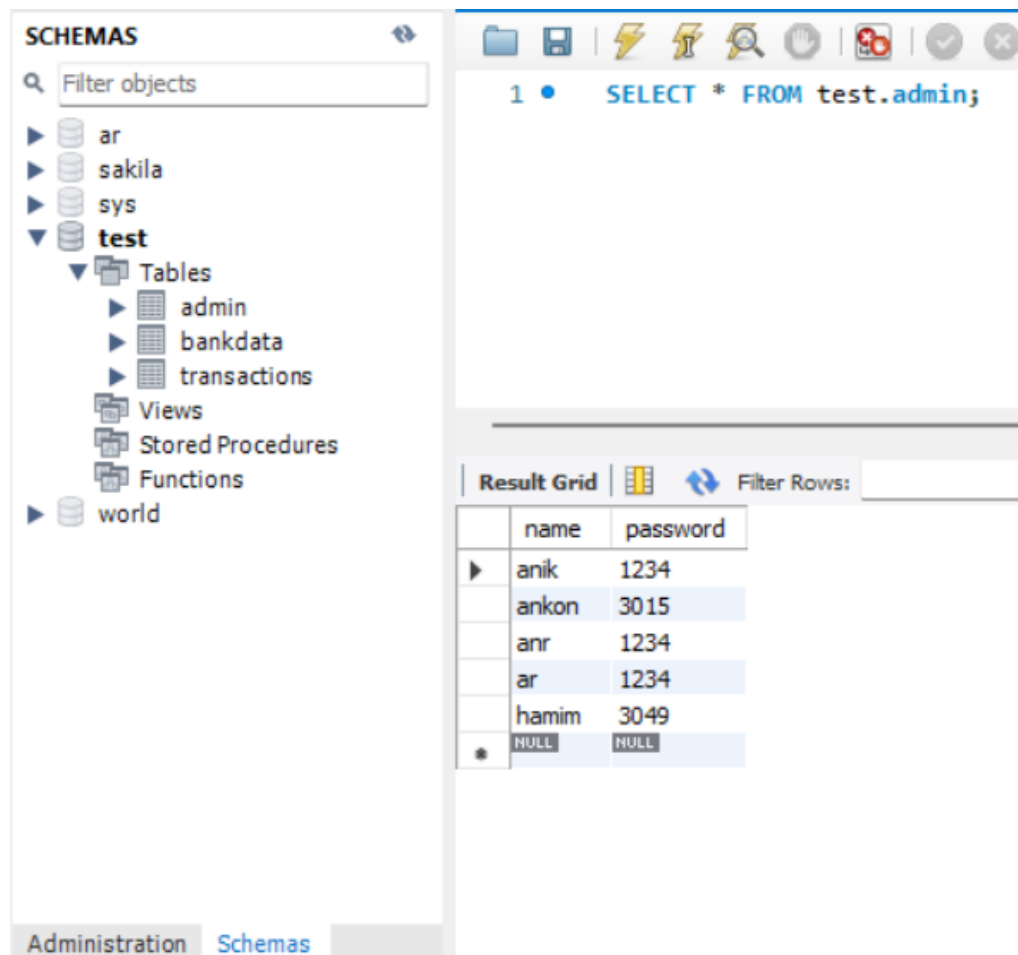


Figure 4.5. Database

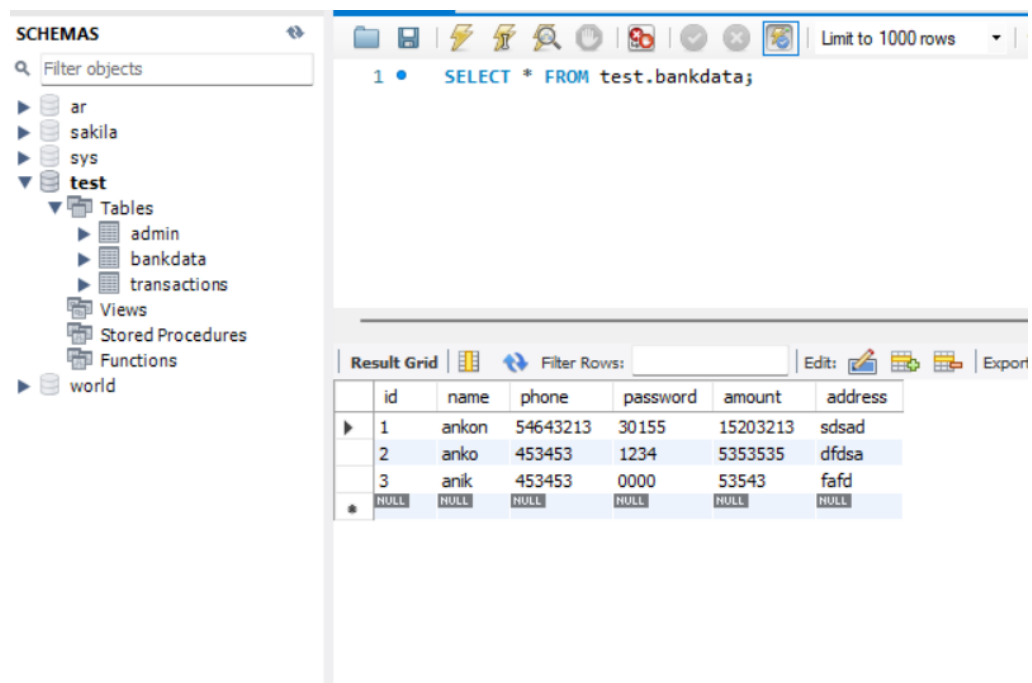


Figure 4.6. Database

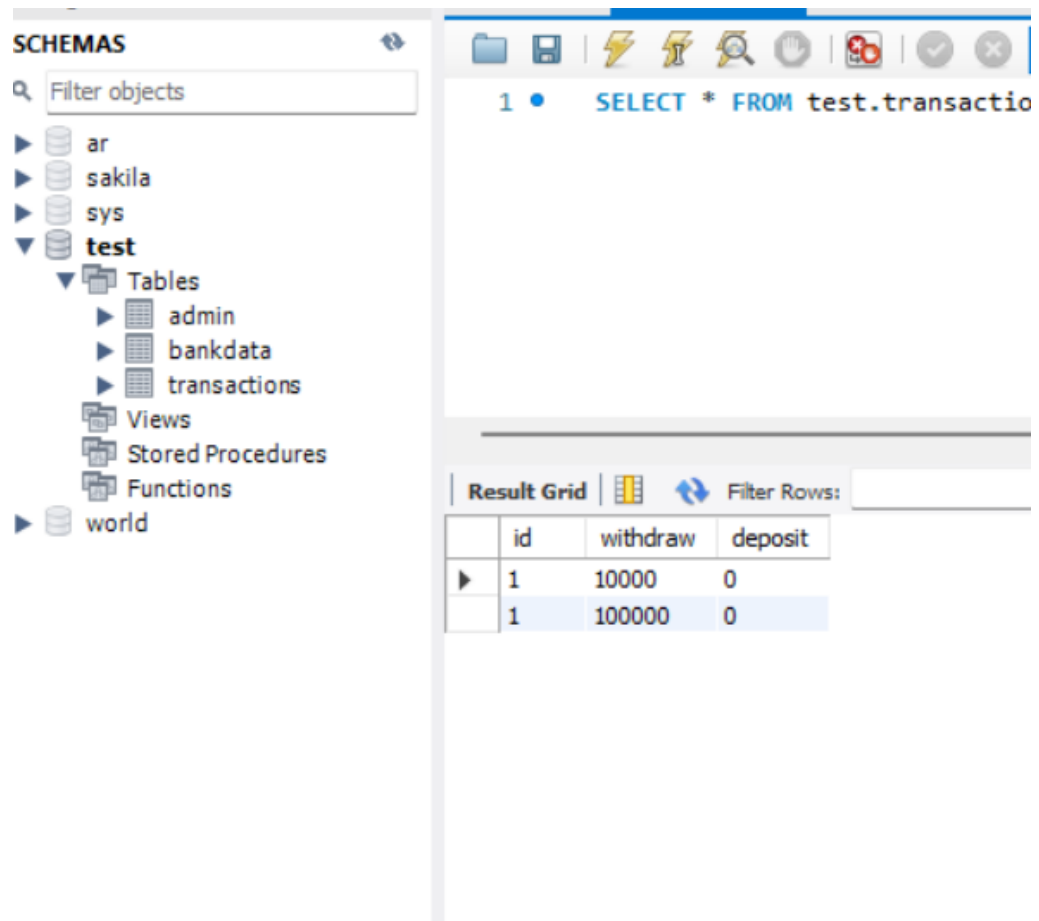


Figure 4.7. Database

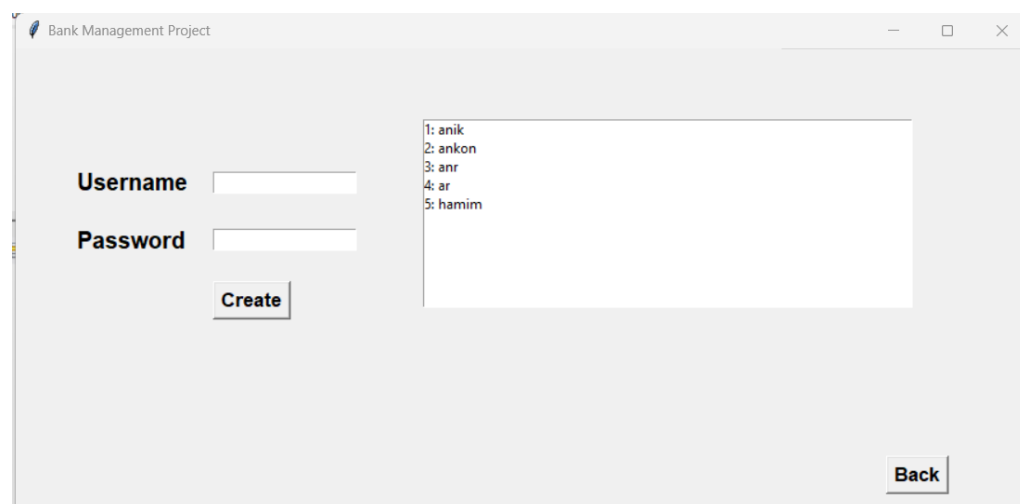


Figure 4.8. Account Creation Page

Bank Management Project

Logout

Welcome, anik

To update your password, address and phone number:

Enter Phone

Enter Password

Enter Address

Update

Your Current Amount:

53543

Withdraw/deposit

Withdraw

Deposit

Figure 4.9. User Page

## Impacts, Ethics, and Challenges

### 5.1 Impacts (on Society)

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 work space 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 manual 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. The project analyzes the system requirements and then comes up with the requirements specifications. It studies other related systems and then come up with system specifications. The system is then designed in accordance with specifications to satisfy the requirements. The system design is then implemented with MYSQL, PHP and HTML. The system is designed as an interactive and content management system. The content management system deals with data entry, validation confirm and updating whiles the interactive system deals with system interaction with the administration and users. Thus, above features of this project will save transaction time and therefore increase the efficiency of the system.

## **5.2 Ethics**

Do not use computers to harm other users. Do not use computers to steal others information. Do not access files without the permission of the owner. Do not copy copyrighted software without the author's permission. Participant privacy, confidentiality and anonymity. Participant privacy, confidentiality and anonymity were the most commonly reported ethical concerns. These concerns are applicable to internet research across all disciplines, not just those involving families and children

## **Conclusion**

Bank management system is a virtualization of transactions in banking system. The banking system are used manual working but when we used online banking system it is totally virtualization process which avoid manual process and converts it in automatic process . If user can make a transaction in bank management system it is available in any were also user can link with account, change branch location easily. Bank management system is saving the time with accuracy than bank manual system.

## References

- a. C 8.0 and .NET, Core 3.0 written by Mark J. Price
- b. Head First C is written by Andrew Stellman (Author),Jennifer Greene
- c. C in Depth is written by Jon Skeet



## **Future Work**

Our whole documentation shows the work we had done up till now. So, particularly our program is fully complete and we further want to focus on the future to making it a better and more efficient product.

In future, we are thinking of adding more specifications and features to make it look better and easy to use. We are thinking of making a formula that will generate cash flow statement, daily transactions, owner's equity, all will be generated after one full year to show the progress of the business annually. Future plan is still being in our thought and our concern, we will soon release a version far better than this one and hope that our work will be appreciated.