Beirut Arab University
Faculty of Engineering
Department of Electrical and Computer Engineering

Course Programming for Engineers – COMP215
Instructor Eng. Hadi Al-Mubasher

Bank Management System

Final Lab Report

Prepared by

Jana Moslemani	202304315
Dima Abdallah	202101973
Omar Omar	202102211
Zaher Ismail	202205114

Table of Contents

Objective

1. Allow the user to register and login
2. Allow the user to deposit money/update file
3. Allows the user to withdraw money/ update file
4. Allow the user to send transactions to other people
5. Allow the user to view/ edit personal details.

Program Description

The python-based bank management system project is a console that executes the key operations of banking software. The user can create a new account, makes deposits and withdrawals, and lastly perform transactions to other users.

Existing System vs. Proposed System

This Proposed Application Affords:

- Information is more secure.
- Facilitates dealing with banks.
- Availability 24/7.
- Editing is easier, faster and simpler.
- Avoids calculation errors.
- More user-friendly.
- Reliable and Efficient.

Some drawbacks of existing system include:

- No high security of bank and customer information
- Require physical efforts
- Manual entry and data processing
- Papers and documents are not safe
  There doesn't exist a backup of information

Thus, digitalizing in the banking system has a lot of benefits beyond the project goals and objective.

Glimpse of the Output

1. Main Screen

The program's output starts with a main screen- a title, the bank's motto, and a picture representing the bank, and has two buttons:

- the register button, which allows new users to create a file that will be saved in the directory

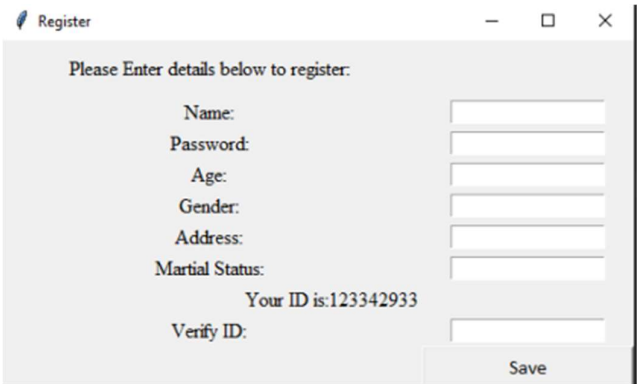
- the login button, which opens the account screen for old users to modify and view some information.

2. Register Screen

The register screen asks the user to enter some detailed information to be stored in a new file under his/her name.

The details include :

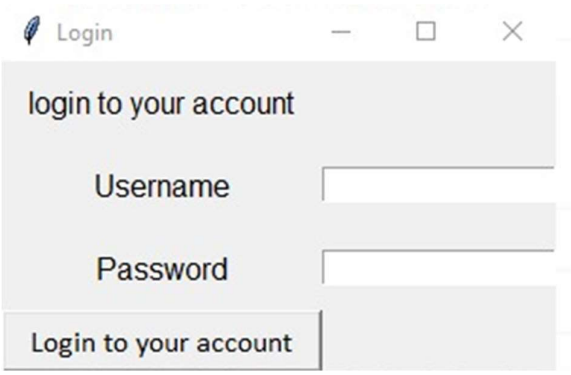
- user's name,
- user's password,
- age,
- gender,
- martial status
- ID verification



3. Login Screen

The login Screen asks the user to enter his name and password and checks all the names in the directory.

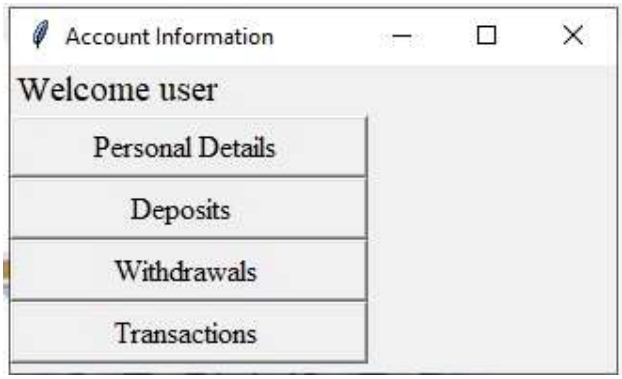
If the entered name matches a name in the directory, the file is opened and the program checks if the password entered is the same as the one in the file.



If both values are found, the account details screen pops up where the user can edit and/or view his details.

4. Account Screen

The account screen allows the user to access his personal details and deposit or withdraw some money, and send some money transactions to other people.



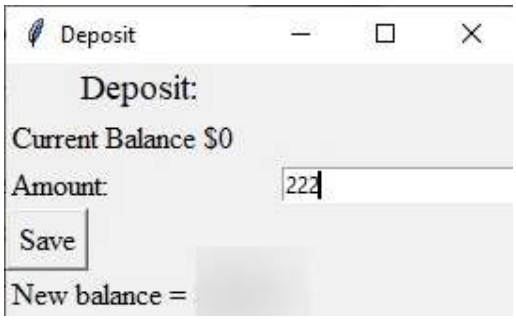
5. Personal Details

This screen displays the data about user in the form of “name”, “password”, “age”, “gender”, “birth”, “status”, “account id”, “current balance”.

6. Deposit Screen

This screen helps users to deposit a certain amount of money in their accounts.

It will ask the amount and add it to the available balance.



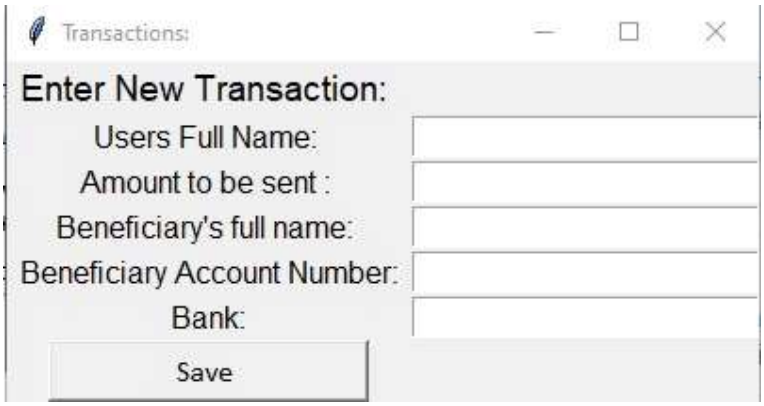
7. Withdrawals Screen

This screen helps users to withdraw a certain amount of money from their accounts. Even If the user does not have sufficient amount, the bank shows the amount taken and displays them as negative amount.

Total Balance = Current Balance – Withdrawal Amount

8. Transaction Screen

This bank system allows the user to send transactions to other people, and with the help of other banks too.



Future Work - what can be further added to this program

- Improve System's security
- Editing two user's accounts that are sending transactions under the same bank
- Allow the user to take out loans
- Calculate percentage interest when putting a certain amount of money
- Add administrator tasks
- Advanced GUI techniques

Source Code

```
from tkinter import *
import os
from PIL import ImageTk, Image
import random




main_screen = Tk()
main_screen.title('Bank Management System')



image = Image.open('image.png')
image = image.resize((400, 300))
image = ImageTk.PhotoImage(image)



def generate_account_number():
    return str(random.randrange(1111111111, 9999999999))

def get_account_number():
    current_number = open('incrementer.txt','r')
    current_number = current_number.readline()
    current_number = int(current_number)
    current_number += 1
    current_number = str(current_number)
    current = open('incrementer.txt','w')
    current.write(current_number)
    current.close()
    return current_number


def finish_reg():
    global register_screen
    entered_name = name_temp.get()
    entered_password = password_temp.get()
    entered_age = age_temp.get()
    entered_gender = gender_temp.get()
    entered_address = address_temp.get()
    entered_status = status_temp.get()
    entered_id = id_temp.get()
    all_files = os.listdir()

    if entered_name == '' or entered_password == '' or entered_gender =='' 
or entered_address=='' or entered_status=='' or entered_age=='':
        Label(register_screen, text='Please enter all fields! ', 
font=('Times',12)).grid(row=16, column=0)
        return

        new_account = open(entered_name, 'w')
        new_account.write(entered_name + '\n')
        new_account.write(entered_password + '\n')
        new_account.write(entered_age + '\n')
        new_account.write(entered_gender + '\n')
```

```
        new_account.write(entered_address + '\n')
        new_account.write(entered_status + '\n')
        new_account.write(entered_id + '\n')
        new_account.write('0')
        new_account.close()


def register():
    global register_screen
    register_screen = Toplevel(main_screen)
    register_screen.title('Register')



    global name_temp
    global password_temp
    global age_temp
    global gender_temp
    global address_temp
    global status_temp
    global id_temp
    name_temp = StringVar()
    age_temp = StringVar()
    gender_temp = StringVar()
    password_temp = StringVar()
    address_temp = StringVar()
    status_temp = StringVar()
    id_temp = StringVar()



    Label(register_screen, text='Please Enter details below to register: ',
font = ('Times', 12)).grid(row = 0, sticky = N, pady = 10)
    Label(register_screen, text='Name: ', font=('Times', 12)).grid(row=1,
sticky=N, column = 0)
    Label(register_screen, text='Password: ', font = ('Times',
12)).grid(row = 2, sticky = N, column = 0)
    Label(register_screen, text='Age: ', font=('Times', 12)).grid(row=3,
sticky=N, column = 0)
    Label(register_screen, text='Gender: ', font=('Times', 12)).grid(row=4,
sticky=N, column = 0)
    Label(register_screen, text='Address: ', font=('Times',
12)).grid(row=5, sticky=N, column = 0)
    Label(register_screen, text='Martial Status: ', font=('Times',
12)).grid(row=6, sticky=N, column = 0)
    Label(register_screen, text='\t\t\tYour ID is:' +get_account_number() ,
font=('Times', 12)).grid(row=7, sticky=N, column = 0)
    Label(register_screen, text='Verify ID: ', font=('Times',
12)).grid(row=8, sticky=N, column=0)


    Entry(register_screen, textvariable=name_temp).grid(row =1, column = 1)
    Entry(register_screen, textvariable=password_temp, show =
'*').grid(row=2, column = 1)
    Entry(register_screen, textvariable=age_temp).grid(row=3, column = 1)
    Entry(register_screen, textvariable=gender_temp).grid(row=4, column =
1)
    Entry(register_screen, textvariable=address_temp).grid(row=5, column =
1)
    Entry(register_screen, textvariable=status_temp).grid(row=6, column =
1)
```

```
    Entry(register_screen, textvariable=id_temp).grid(row=8, column=1)

    Save_Button = Button(register_screen, text = 'Save', font = ('Calibri',
12), width = 20, command = finish_reg)
    Save_Button.grid(row=15, column = 1)

def check_login():
    global entered_name
    global entered_password
    global login_screen
    all_files = os.listdir()
    entered_name = temp_login_name.get()
    entered_password = temp_login_password.get()
    for name in all_files:
        if name == entered_name:
            file = open(name, 'r')
            file1 = file.read()
            file1 = file1.split()
            if entered_password == file1[1]:
                Login_screen.destroy()
                account_screen = Toplevel(main_screen)
                account_screen.title('Account Information ')
                Label(account_screen, text='Welcome ' + entered_name ,
font=('Times', 14)).grid(row = 0, sticky = W)
                Button(account_screen, text='Personal Details ',
font=('Times', 12), width = 20, command=personal_details).grid(row = 1,
column = 0)
                Button(account_screen, text='Deposits  ', font=('Times',
12), width = 20, command = deposits).grid(row=2, column = 0)
                Button(account_screen, text='Withdrawals ', font=('Times',
12), width = 20, command = withdrawals).grid(row=3, column = 0)
                Button(account_screen, text='Transactions ', font=('Times',
12), width = 20, command=transactions).grid(row=4, column=0)
            else:
                Label(Login_screen, text = 'Incorrect Password ', font =
('Times', 12)).grid(row = 6, column = 0)
def personal_details():
    file = open(entered_name, 'r')
    file2 = file.read()
    information = file2.split('\n')
    name_split = information[0]
    password_split = information[1]
    age_split = information[2]
    gender_split = information[3]
    birth_split = information[4]
    status_split = information[5]
    id_split = information[6]
    balance_split = information[7]
    details_screen=Toplevel(main_screen)
    details_screen.title(''' User's Personal Details ''')
    Label(details_screen, text = 'Personal Details ', font= ('Times',
14)).grid(row=0, sticky=N)
    Label(details_screen, text='Name: '+name_split, font=('Times',
12)).grid(row=1, sticky=W)
    Label(details_screen, text='Password: '+password_split, font=('Times',
12)).grid(row=2, sticky=W)
    Label(details_screen, text='Age: ' + age_split, font=('Times',
12)).grid(row=3, sticky=W)
    Label(details_screen, text='Gender: ' + gender_split, font=('Times',
12)).grid(row=4, sticky=W)
    Label(details_screen, text='Birth: '+birth_split, font=('Times',
```

```
12)).grid(row=5, sticky=W)
    Label(details_screen, text='Status: '+status_split, font=('Times',
12)).grid(row=6, sticky=W)
    Label(details_screen, text='Account ID: '+id_split, font=('Times',
12)).grid(row=7, sticky=W)
    Label(details_screen, text='Current Balance: $'+balance_split,
font=('Times', 12)).grid(row=8, sticky=W)




def deposits():
    global amount
    global current_balance_label
    global deposit_notif
    amount = StringVar()
    file= open(entered_name, 'r')
    file2 = file.read()
    details = file2.split('\n')
    balance = details[7]
    global deposit_screen
    deposit_screen = Toplevel(main_screen)
    deposit_screen.title('Deposit')
    Label(deposit_screen, text = 'Deposit: ', font = ('Times',
14)).grid(row = 0, sticky = N)
    current_balance_label = Label(deposit_screen, text = 'Current Balance
$'+ balance, font=('Times',12))
    current_balance_label.grid(row = 1, sticky = W)
    Label(deposit_screen, text = 'Amount: ', font = ('Times', 12)).grid(row
= 2,column = 0, sticky = W)
    Entry(deposit_screen,textvariable=amount).grid(row =2, column=1)
    Button(deposit_screen,text='Save',font=('Times', 12), command =
finish_deposit).grid(row=3, sticky=W)




def finish_deposit():
    global deposit_screen
    global all_deposits
    global total_deposit
    if amount.get() == '':
        Label(deposit_screen, text = 'Amount is required ', font
=('Times',12)).grid(row = 4, sticky = N)
    elif float(amount.get()) < 0:
        Label(deposit_screen, text='Value should be positive:  ',
font=('Times', 12)).grid(row=5, sticky=N)
    else:
        file = open(entered_name, 'r+')
        file2 = file.read()
        details = file2.split('\n')
        balance = details[7]
        updated_balance = balance
        updated_balance = round(float(updated_balance) +
float(amount.get()), 2)
        file2 = file2.replace(balance , str(updated_balance))
        file.seek(0)
        file.truncate(0)
        file.write(file2)
```

```
        file.close()
        Label(deposit_screen, text='New balance = $' +
str(updated_balance), font=('Times', 12)).grid(row = 6, column = 1)


def withdrawals():
    global amount
    global current_balance_label
    global withdrawal_screen
    amount = StringVar()
    file = open(entered_name, 'r')
    file2 = file.read()
    details = file2.split('\n')
    balance = details[7]
    withdrawal_screen = Toplevel(main_screen)
    withdrawal_screen.title('Withdrawal')
    Label(withdrawal_screen, text='Withdrawal: ', font=('Times',
14)).grid(row=0, sticky=N)
    current_balance_label = Label(withdrawal_screen, text='Current Balance
$' + balance, font=('Times', 12))
    current_balance_label.grid(row=1, sticky=W)
    Label(withdrawal_screen, text='Amount: ', font=('Times',
12)).grid(row=2, column=0, sticky=W)
    Entry(withdrawal_screen, textvariable=amount).grid(row=2, column=1)
    Button(withdrawal_screen, text='Save', font=('Times', 12),
command=finish_withdrawal).grid(row=3, sticky=W)

def finish_withdrawal():
    global withdrawal_screen
    if amount.get() == '':
        Label(withdrawal_screen, text = 'Amount is required ', font
=('Times',12)).grid(row = 4, sticky = N)
    elif float(amount.get()) < 0:
        Label(withdrawal_screen, text='Value should be positive:  ',
font=('Times', 12)).grid(row=5, sticky=N)
    else:
        file = open(entered_name, 'r+')
        file2 = file.read()
        details = file2.split('\n')
        balance = details[7]
        updated_balance = balance
        updated_balance = round(float(updated_balance) -
float(amount.get()), 2)
        file2 = file2.replace(balance , str(updated_balance))
        file.seek(0)
        file.truncate(0)
        file.write(file2)
        file.close()
        Label(withdrawal_screen, text='New balance = $' +
str(updated_balance), font=('Times', 12)).grid(row = 6, column =1)
def login():
    global temp_login_name
    global temp_login_password
    global Login_screen
    temp_login_name = StringVar()
    temp_login_password = StringVar()
    Login_screen = Toplevel(main_screen)
    Login_screen.title('Login')
    Label(Login_screen, text = 'login to your account', font = ('Times ',
12)).grid(row = 0, sticky=N, pady = 10)
    Label(Login_screen, text='Username', font=('Times ', 12)).grid(row=1,
```

```
sticky=N, pady=10)
    Label(Login_screen, text='Password', font=('Times ', 12)).grid(row=2,
sticky=N, pady=10)
    Entry(Login_screen, textvariable = temp_login_name).grid(row = 1,
column=1)
    Entry(Login_screen, textvariable=temp_login_password, show =
'*').grid(row=2, column=1)
    Button(Login_screen, text = 'Login to your account', command =
check_login, width = 20, font = ('Calibri', 12)).grid(row = 3, sticky = W)

def transactions():
    global transaction_screen
    transaction_screen= Toplevel(main_screen)
    transaction_screen.title('Transactions: ')
    global full_name
    global sent_amount
    global receiver
    global receiver_id
    global bank
    full_name = StringVar()
    sent_amount = StringVar()
    receiver = StringVar()
    receiver_id = StringVar()
    bank = StringVar()
    Label(transaction_screen, text='Enter New Transaction: ', font=('Times
', 14)).grid(row=0, sticky=N)
    Label(transaction_screen, text='Users Full Name: ', font=('Times ',
12)).grid(row=1, sticky=N)
    Label(transaction_screen, text='Amount to be sent : ', font=('Times ',
12)).grid(row=2, sticky=N)
    Label(transaction_screen, text='''Beneficiary's full name: ''',
font=('Times ', 12)).grid(row=3, sticky=N)
    Label(transaction_screen, text=' Beneficiary Account Number: ',
font=('Times ', 12)).grid(row=4, sticky=W)
    Label(transaction_screen, text=' Bank: ', font=('Times ',
12)).grid(row=5, sticky=N)
    Entry(transaction_screen, textvariable = full_name, font=('Times ',
12)).grid(row=1, column = 1)
    Entry(transaction_screen, textvariable=sent_amount, font=('Times ',
12)).grid(row=2, column=1)
    Entry(transaction_screen, textvariable=receiver, font=('Times ',
12)).grid(row=3, column=1)
    Entry(transaction_screen, textvariable=receiver_id, font=('Times ',
12)).grid(row=4, column=1)
    Entry(transaction_screen, textvariable=bank, font=('Times ',
12)).grid(row=5, column=1)
    Button(transaction_screen, text='Save ', command=save_trans, width=20,
font=('Calibri', 12)).grid(row=7,sticky=N)

def save_trans():
    global sent_amount
    full_name_temp = full_name.get()
    sent_amount_temp = sent_amount.get()
    receiver_temp = receiver.get()
    receiver_id_temp = receiver_id.get()
    bank_temp = bank.get()
    global transaction_screen
    file = os.listdir()
    file = open(full_name_temp, 'w')
    file.write(full_name_temp+'\n')
    file.write(sent_amount_temp+ '\n')
```

```
    file.write(receiver_temp+'\n')
    file.write(receiver_id_temp+'\n')
    file.write(bank_temp)
    file.close()
    file = open(entered_name, 'r+')
    file2 = file.read()
    details = file2.split('\n')
    balance = details[7]
    updated_balance = balance
    updated_balance = round(float(updated_balance) -
float(sent_amount.get()), 2)
    file2 = file2.replace(balance, str(updated_balance))
    file.seek(0)
    file.truncate(0)
    file.write(file2)
    file.close()
    Label(transaction_screen, text='New balance = $' +
str(updated_balance), font=('Times',12)).grid(row = 10, column = 1)


Label(main_screen, text = 'Bank Management System ', font = ('Times ',
14)).grid(row = 0, sticky=N, pady = 10)
Label(main_screen, text = 'We are with you every step of the way! ', font =
('Times ', 12)).grid(row = 2, column= 0)
Label(main_screen, image=image).grid(row = 1, sticky = N, pady = 15)


Button(main_screen, text = 'Register Here ', font = ('Times ', 12), width =
20, command = register).grid(row = 3, sticky = N)
Button(main_screen, text = 'Login Here ', font = ('Times ', 12), width =
20, command = login).grid(row = 4, sticky = N)


main_screen.mainloop()
```

Thank You!