Bank management System

نظام إدارة بنك

2022

By names :

Aya Taher Abdelmonem
Aliaa Essam
Aliaa Elsaied Mohamed
Mohamed Abdou Abdelrahman
Hatem Ibrahem
Abdelsatar Ehab Ebraheam
Mahmoud Reda Elghareeb
Ziad Mustafa Megahed
Ahmed Hossam
Ahmed Mahmoud Hassan
Mohamed Gamal Mohamed

Instructor / Eng Mohamed Abdo

Abstract:

The project aims to build a simple banking system

the project has eleven functions ,every function do its job.

The first function is login and its job :

the user enters the username and password. If the username and password are in the database of the users, the system displays the main menu page, else the system prints an error message.

Coding to this function is:

```
def login():
    usBrnamB=input("EntBr your usErnamE")

    password=input("EntEr your password")
    D=usEr_data["usErnamE"].valuES
    P=usEr_data["password"].valuES
    if usernamB in D and password in P  :
        return "MM"
    if usernamB not in D and password not in P  :
        printC'error")
        return "L"
```

The second function is main menu and its job:

This function is used to show two choices to the user to choose from

what he wants to do.
#We print out 2 choices created with the if and else if functions. #When the customer enters this function, he finds two choices, which are:

Add customer
Customer services

#If the customer chooses any of these options, he will be transferred to the function he chose, but if he chooses a number that is invalid, a message “error” will appear and he will be returned to the options again until he chooses a number that is in the list.

Coding to this function is:

```
def mainMenuO:
os.systBmCcIs')
tBmpJist<"CS","AC"]
printC'Choose one action from the following list")
print("5ic5ic ************** ******)
printC'CS: Customer ServicBs")
printC'AC: Add NEW Customer")
pri<—,
while True:
    statE=input("Enter a valid choice:")
    if state not in tempjist:
        print("invalid input, try again") EISE:
        brEak
return state
```

The third function is customer menu and its job:

this function is used to show a list for the client to choose from what he wants to do. We print out 7 choices .created with the While, if and else if functions
When the customer enters this list, he finds in front of :him seven choices, which are
Account Details
Deposit Cash
Withdraw Cash
Transfer Money
Transaction History
Add Account
Main Menu
If the customer chooses any of these options, he will be transferred to the function he chose, but if he chooses a number that is not in the list, a message "Wrong Choice Please Try Again" will appear and he will be returned to the options again until he chooses a number that is in the list
Coding to this function is : def customErMenuO:

```
printC'Choose one action from the following list") print(H** ********** *****")
print("AD :  Account Details")
printC'DC: Deposit Cash")
printC'WC: Withdraw Cash")
printC'TM: Transfer Money")
print("TH :  Transaction History") printC'AA: Add Account")
print("MM :  Return to the main menu")
print(H** ********** *****")
while True:
```

```
        state=input("Enter a valid choice:") if state not in tempjist:
            print("invalid input, try again")
        EISE:
            brEak
    return state
```

The fourth function is add account and its job :

this function is used when a customer comes to the bank to add new account. And also he has the option to add multiple account, this function adds flexibility between the bank and customer. Add account by applying few .steps

First ,the customer has insert an id to complete the account registration and that id must be at least 6 digits and doesn't contain any letters or emotion or underscore. Second , the system of bank enter a random number each customer has a unique number which enhances the security of the bank system and .also the privacy of their accounts

Third ,at the end function the customer choose the initial balance to start his an account and pay the initial balance be deposited on his bank account

Coding to this function is :

```
def error():
    print("incorrect id\ntry again press:0\nquit press:1")
    s=int(input())
    if(s==0):
        addAccount()


import random

def addAccount():
    global account_data
    os.system("cls")
    account_number= random.randint(100000,999999)
    print("Account number :",account_number)
    customer_id=(input("customer_id :"))
    if (len(customer_id)==6):
        if (customer_id.isnumeric()==True):
            print("correct id")
        else:
            error()
    else:
        error()

    balance=(input("the initial balance :"))
```

```
    account_data=account_data.append({"account_number":account_number,
      "customer_id":customer_id,
      "balance":balance,
      "account_type":"Saving Account"
    }, ignore_index=True)
    account_data.to_excel("account_data.xlsx",index=False)
    print("the Account has been successfully added")
    input("Press Enter to continue: ")
    return "CM"
```

The fifth function is account detail and its job:

This function is used to print the account details for a customer,
It takes the .customer id as input parameter

print (account_data[account_data[''customer_id)"]==" "([(customerjd

This state print the details of the account data for the entered ID only if the ID isn't found in the database it print "data frame"

Coding to this function is :

```
def accountDBtail(customBr_id):
    os.systEm("cls")

print(account_data[account_data["customErJd"]==customEr_id][["account_numbEr","
balancE","account_typE"]])
    inputC'Press enter to continue:")
    return "CM"
```

The 6th function is transaction History:

this function is used to print the transaction history for a specific account for the customer

the function takes the customer id as an parameter and ask the customer to choose one of the accounts If the customer choice not in account_details print ("this account is invalid") and ask the userf'Enter 'yes' to try again, 'no' to return:")
If he choose "no"the function will return to the customer menu.Otherwise the function will return to transaction history

If the customer choice in account_details print a list of a transactionHistory from the chosen account and return to the customer menu

Coding to this function is :

```
def transactionHistory(customBr_id):
    print("List of your accounts : \n\n\n\n")

account_dBtails=account_data[account_data["customBr⌋  d"]==customEr_id][["account
_numbEr","balance","account_typE"]]
    print(account_dEtails)
    while True:
        account_numbEr=int(input("EntEr Account Number:"))
        if account_numbBr not in account_dBtails["account_numbBr"].valuBS :  printC'This
            account is not found")
            statE=input("EntEr 'T' to try again, 'C to cancel and return:") if
            statE=='C':
                return "CM"
```

```
        else:
            break

trans_hist=trans_history[trans_history["account_numbEr"]==account_numbEr][["timE"
/'type","amount"]]
    os.systBmC'cls")
    print(trans_hist)
    inputC'Press enter to continue:")
    return "CM"
```

The 7th function is withdraw Cash and its job:

withdrawCash(123456)

When the customer want to withdraw cash from an account the function takes the customer as input as an input parameter then the customer should choose his account from a list of the customer's account to choose the account he wants to withdraw cash from and the amount of money he wants to withdraw should be less than his balance to complete his request and this function updates continously his data and transiction history

Coding to this function is:

```
def withdrawCash(customBr_id):
    global account_data
    global trans_history
    os.systBmC'cls")
    print("List of your accounts : \n\n\n\n")
```

```
account_dBtails=account_data[account_data["customBr ⌋  d"]==customEr_id][["account
_numbEr","balancE","account_typE"]]
    print(account_dEtails)
    while True:
        account_numbEr=int(input("EntEr Account Number:"))
        if account_numbBr not in account_dBtails["account_numbBr"].valuBS :  print("fhis
            account is not found")
            statB=input("EntBr 'T' to try again, 'C to cancel and return:")
            if statE=='C':
                return "CM"
        EISE:
            break

currnBt_balancB=account_data[account_data["account_numbBr"]==account_numbBr]["
balancB"].valuBs[D]
    while True:
        cash_amount=int(input("Enter the amount of cash:"))
        if cash_amount > currnBt_balancB :

            printCThe cash Exceeds the total balance in the account")
            statB=input("EntBr 'T' to try again, 'C to cancel and return:") if statE=='C':
                rEturn "CM"
         BISE:
            brEak


account_data.loc[account_data[account_data["account_numbBr"]==account_numbBr].in
dEX,"balancE"]=currnEt_balancE-cash_amount
       datE_timE=datEtimE.now()
       currEnt_datE=datE_timE.strftimE("%d_%m_%Y")
       currEnt_timE=datE_timE.strftimE("%H : %M : %S")
       trans_history=trans_history.appBnd({"account_numbBr" : account_numbBr,
         "timE":curTEnt_datE+" - "+currBnt_timB,
         "tyPB" : "Withdraw",                    "
         "amount" : cash_amount
         }, ignore⌋  ndEX=TruE)
```

```
    account_data.to_EXCEl("account_data.xlsx",indEX=FalsE)
    trans_history.to_BXCBl("trans_history.xlsx",indBX=FalsB)
    printCThe process has been successfully done")
    inputCPress enter to continue:")
    state="CM"
    return state
```

The 8th function is deposit cash and its job:

This function is used to deposit cash to an account The function takes the customer id as an input
.parameter
The function prints a list of the customer's accounts .and asks the user to choose one of them
If the user choice in account's list, the function asks .the user to enter the amount of money to deposit The function first updates the account-data to add the amount of money that was deposited to the balance in .the chosen account
The function updates the transaction history by adding a new row that provides all information about the .deposit process
The system displays a message stating that the process is completely done and asks the user to press enter to continue, so, the system returns to customer .menu
If the chosen account isn't in the list, the system displays
a message stating (it is invalid choice) and asks the user if he wants to try again if he choses yes, the system returns to depositcash function, otherwise the system returns to mainmenu
coding to this function is:

```
def dBpositCash(customBr_id):
    global account_data
    global trans_history
    os.systBm("cls")
    print("List of your accounts : \n\n\n\n")
```

```
account_dBtails=account_data[account_data["customBr ⌋  d"]==customEr_id][["account
_numbEr","balancE","account_typE"]]
      print(account_dEtails)
      while True:
         account_numbEr=int(input("EntEr Account Number:"))
       if accountjiumber not in account_dBtails["account_numbBr"].valuBs: printC'This account is
          not found")
          state^inputC'Enter 'T' to try again, 'C to cancel and return:")
          if statE=='C':
             return "CM"
      EISE:
          break
   cash_amount=int(input("EntBr the amount of cash:"))

currnBt_balancB=account_data[account_data["account_numbBr"]==account_numbBr]["
balancE"].valuEs[D]

account_data.loc[account_data[account_data["account_numbEr"]==account_numbEr].in
dEX,"balancE"]=currnEt_balancE+cash_amount
   datB_timB=datBtimB.now()
   currBnt_datB=datB_timB.strftimB("%d_%m_%Y")
   currBnt_timB=datB_timB.strftimB("%H : %M : %S")
   trans_history=trans_history.appBnd({"account_numbBr" : account_numbBr,
     "timB" : currBnt_datB+" - "+currBnt_timB,
     "typE":"DEposit",
     "amount" : cash_amount
     }, ignore⌋  ndEX=TruE)
   account_data.to_EXCEl("account_data.xlsx",indEX=FalsE)
   trans_history.to_BXCBl("trans_history.xlsx",indBX=FalsB)
   printCThe process has been successfully done")
   inputCPress enter to continue:")
   state="CM"
   return state
```

The 9th function is transfer money and its job:

This function is used to transfer money from one account to other The function takes the customer id as an input parameter The

function prints a list of the customer's accounts and asks the user to choose of them
The function asks the user to enter the amount of money to transfer
The function asks the user to enter the recipient's account number
The function updates the account data and transaction history

Coding to this function is:

```
def transfBrMonBy(customBr_id):
    global account_data
    global trans_history
    os.systBm("cls")
    print("List of your accounts : \n\n\n\n")


account_dBtails=account_data[account_data["customBr⌟  d"]==customEr_id][["account
_numbEr","balancE","account_typE"]]
    print(account_dEtails)
    while True:
        account_numbEr=int(input("EntEr Account Number:"))
        if account_numbBr not in account_dBtails["account_numbBr"].valuBS :  printC'This
            account is not found")
            statB=input("EntBr 'T' to try again, 'C to cancel and return:") if statE=='C':
                return "CM"
        else:

            break
currnBt_balancB=account_data[account_data["account_numbBr"]==account_numbBr]["
balancE"].valuEs[D]
    whilETruE:
        cash_amount=int(input("EntEr the amount of cash:"))
        if cash_amount > currnBt_balancB :
            printCThe cash Exceeds the total balance in the account")
            statB=input("EntBr 'T' to try again, 'C to cancel and return:") if statE=='C':
                return "CM"
        EISE:
            break
```

```
    while True:
        rBCBivB_account_numbBr=int(input("EntBrthB recipiEnt's account number:")) if
rBCBivB_account_numbBr not in account_data["account_numbBr"].valuBS or
rECEivE_account_numbEr== account_number :
            print("This account is not found")
            statE=input("Enter 'T' to try again, 'C to cancel and return:")
            if statE=='C':
                rEturn "CM"
        EISE:
            brEak


account_data.loc[account_data[account_data["account_numbBr"]==account_numbBr].in
dEX,"balancE"]=currnEt_balancE-cash_amount
    datB_timB=datBtimB.now()
    currBnt_datB=datB_timB.strftimB("%d_%m_%Y")
    currBnt_timB=datB_timB.strftimB("%H : %M : %S")
    tEmp_str="SEnt to account, rBCBiver ID:" + str(rECEivE_acoount_numbEr)
    trans_history=trans_history.appBnd({"account_numbBr" : account_numbBr, "timB" :
    currBnt_datB+" - "+currBnt_timB,
      "typB" : tBmp_str, "amount" : cash_amount }, ignoreJ—E)


currnEt_balancE=account_data[account_data["account_numbEr"]==rECEivE_account_nu
mbBr]["balancB"].valuBs[D]

account_data.loc[account_data[account_data["account_numbBr"]==rBCBivB_account_nu
mbBr].indBX,"balancB"]=currnBt_balancB+cash_amount
    tBmp_str="RBCBivBd from account, sender ID:" + str(account_numbBr)
    trans_history=trans_history.appBnd({"account_numbBr" : rBCBivB_account_numbBr,
      "timB" : currBnt_datB+" - "+currBnt_timB,
      "typB" : tBmp_str,
      "amount" : cash_amount
      }, ignore」 ndEX=TruE)
    account_data.to_EXCEl("account_data.xlsx",indEX=FalsE)
    trans_history.to_BXCBl("trans_history.xlsx",indBX=FalsB)
```

```
printCThe process has been successfully done")
print(account_data)
print(trans_history)
inputCPress enter to continue:")
state^'CM"
return state def addCustomer():
global customer_data
os.systBmC'cls")
customEr_namE=input("EntEr customer Name:") while True:
  na」 d=int(input("Enter the national ID:"))
  if najd in customBr_data["national_id"].valuBs:
      print("This ID airEady exists, try again")
  EISE:
      brEak
phonB_num=input("EntBr phonE numbEr:")
addrEss=input("EntErthE address:")
customBr_data=customBr_data.appBnd({"customBr_namB" : customBr_namB,
  "nationaljd" : na_id,
  "phonB_numbBr" : phonB_num,
  "addrBss" : addrBss
  }, ignore」 ndEX=TruE)
customErJdata.to_EXCEl("customEr_data.xlsx",indEX=FalsE)
print("thB CustomBr has been successfully added")
print("Please add an account to completB the registration")
inputC'Press Enter to continue:")
return "AA",na_id
```

The 10th function is customerservices and its job:

This function used to enter the customer menu The function asks

the user to enter the customer id If the customer is not found in the database, the function prints an error message and asks the user to try again The function returns the next state and the customer id

Coding to this function is:

```
def customBrServicBsO:
    os.systBmCcIs')
    while True:
        na」  d=int(inputC'Enter the national ID:"))
        if najd in customBr_data["national_id"].valuBs:
            break
        printCthe customer not found")
    statB^'CM"
    return state,najd
```

The 11th function is addCustomer and its job:

This function is used to add new customer Thefunction asks the user to enter: customer name, national ID which must be unique, phone number, and address The new customer must add an account to complete the registration
The function returns the next state and customer ID

Coding to function is:

```
def addCustomerO:
    global customBr_data
    os.systBm("cls")
    customBr_namB=input("EntBr customer Name:")
    while True:
        na」  d=int(input("EntEr the national ID:"))
        if najd in customBr_data["national_id"].valuBs:
            print("This ID already exists, try again")
        EISE:
            brEak
    phonB_num=input("EntBr phonE numbEr:")
    addrEss=inputj"EntErthE address:")
    customBr_data=customBr_data.appBnd({"customBr_namB" : customBr_namB,
```

```
    "nationaljd" : na_id,
    "phonB_numbBr" : phonB_num,
   "addrBss" : addrBss
   }, ignorE_indEX=TruE)
customErJdata.to_EXCEl("customEr_data.xlsx",indEX=FalsE) printC'the Customer has been
successfully added") print("Please add an account to completB the registration") inputCPress
Enter to continue:")
return "AA",na id
```