JSS Mahavidyapeetha

JSS Academy of Technical Education

Kengeri - Uttarahalli Main Road, Bangalore-560060

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ASSIGNMENT PROJECT SYNOPSIS

COURSE NAME: PYTHON APPLICATION PROGRAMMING

COURSE CODE: 17CS664

TOPIC: SUSPESION DETECTION AND REPORTING BOT

UNDER THE GUIDENCE OF:

Dr NAVEEN NC

Prepared by: - **HARSHITH R SHEKAR[1JS17CS039]**

DHRUVA V [1JS18CS403]

BINDUSHREE R [1JS17CS025]

Marks Scored: **Semester / Branch: 6th 'A' sec**

Signature of the Staff:

TABLE OF CONTENTS

NAME	Page Number
Overview	2
Background Work	3
Architecture	4
Implementation	6
Area of Focus / Optimizations	30
Results / Conclusion	31
References	35

OVERVIEW

The ATM System is the project which is used to access their bank accounts to make cash withdrawals and deposits. Whenever the user needs to make cash withdraws, they can enter their PIN number (personal identification number) and it will display the amount to be withdrawn and deposit. Once their withdrawn was successful, the amount will be debited in their account and if deposit is complete, amount will be credited to the account.

The ATM System is developed in Python and a CSV file (Comma-separated values) using file read write. Python is relatively simple, so it is easy to learn since it requires a unique syntax that focuses on readability. Developers can read and translate Python code much easier than other languages. In turn, this reduces the cost of program maintenance and development because it allows teams to work collaboratively without significant language and experience barriers. Hence, we use this software in our project.

The ATM will service one customer at a time. A customer will be required to enter ATM Card number, personal identification number (PIN) – both of which will be sent to the database for validation as part of each transaction. The customer will then be able to perform one or more transactions. Also, customer must be able to make a balance inquiry of any account linked to the card.

If a transaction fails for any reason other than an invalid PIN, the ATM will display an explanation of the problem, and will then ask the customer whether he/she wants to do another transaction.

Millions of times per day around the globe people are instantly withdrawing money at automatic teller machines (ATMs). Given the fast pace of the world today, it is not surprising that the demand for access to quick cash is so immense. The power of ATMs would not be possible without secure connections. The final act of ATM dispending cash is the result of an amazingly fast burst of the customer never sees, but a trust is being done in a confidential manner.

BACKGROUND WORK

PYTHON LIBRARIES

Since some print statements can be parsed as function calls or statements, python 2 to python 3 cannot always read files containing the print function. When 2 to 3 detects the presence of the from __future__ import print_function compiler directive, it modifies its internal grammar to interpret print() as a function. This change can also be enabled manually with the -p flag. Use -p to run fixers on code that already has had its print statements converted.

os module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

sys module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. It is always available.

The smtplib module defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. For details of SMTP and ESMTP operation, consult RFC 821 (Simple Mail Transfer Protocol) and RFC 1869 (SMTP Service Extensions).

Time module provides various time-related functions

The datetime module supplies classes for manipulating dates and times.

The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. CSV format was used for many years prior to attempts to describe the format in a standardized way in RFC 4180. The lack of a well-defined standard means that subtle differences often exist in the data produced and consumed by different applications. These differences can make it annoying to process CSV files from multiple sources. Still, while the delimiters and quoting characters vary, the overall format is similar enough that it is possible to write a single module which can efficiently manipulate such data, hiding the details of reading and writing the data from the programmer.

re module provides regular expression matching operations similar to those found in Perl.

random module implements pseudo-random number generators for various distributions.

The getpass module provides two functions:

- getpass.getpass(prompt='Password: ', stream=None)
- getpass.getuser()

ARCHITECTURE

ATM Functionalities(Using Basic Concepts of Python):

1. Account Handling
2. Activation\De-Activation of Accounts
3. Admin Control
4. File Handling (using csv file)
5. Account Number Auto-Generation
6. Pin Auto-Generation
7. Simple Encryption\Decryption
8. Date and Time Implication
9. Exceptional Handling
10. Email Notifier Using SMTP Library
11. Amount Transfer
12. Made Applicable for Any Version of Python

Then the functionalities are divided by different python files,

- **Python File** : acc_no_gen.py

  Functionalities implemented Account Number Auto-Generation , Exceptional Handling and Made Applicable for Any Version of Python.

- **Python File** : ATM.py

  Functionalities implemented Account Handling, Amount Transfer, Exceptional Handling and Made Applicable for Any Version of Python.

- **Python File** : Data.py

Functionalities implemented File Handling (using csv file), Exceptional Handling and Made Applicable for Any Version of Python.

- **Python File** : encrypt.py

  Functionalities implemented Simple Encryption\Decryption.

- **Python File** : login.py

  Functionalities implemented Admin Control, Pin Auto-Generation, Date and Time Implication, Exceptional Handling and Made Applicable for Any Version of Python.

- **Python File** : acc_no_gen.py

  Functionalities implemented Email Notifier Using SMTP Library, Exceptional Handling and Made Applicable for Any Version of Python.

Implementation

- **Python File** : acc_no_gen.py

```
from random import randint, randrange
from Data import data

def account_no_gen(user_name):
    d = data()
    alphabets = 'abcdefghijklmnopqrstuvwxyz'
    acc_no = ''
    acc_no += str(len(d.keys()) + 10)
    for alpha in user_name:
        if (len(acc_no) < 12):
            if alpha in alphabets:
                index = alphabets.rfind(alpha)
                acc_no += str(index + 1)

            else:
                acc_no += '0'

        else:
            break

    if len(acc_no) > 12:
        final_acc_no = ''
        for index in acc_no:
            if len(final_acc_no) < 12:
                final_acc_no += index

        acc_no = final_acc_no
        return acc_no

    if len(acc_no) < 12:
        remain_index = 12 - len(acc_no)
        for index in range(remain_index):
            acc_no += str(randint(0,9))

        return acc_no

    else:
        return acc_no

def code():
    a = 'qwertyuiopasdfghjklzxcvbnm'
    conf_code = ''
    for i in range(3):
```

```
        conf_code += str(a[randrange(9)])+str(randrange(9))
    return conf_code
```

- **Python File** : ATM.py

```
#IMPORTS
## USING PYTHON BUILT-IN LIBRARIES
from __future__ import print_function
from getpass import getpass as gp
import os, csv

##USING SOURCE FILES
from encrypt import rot13
from Data import join,data
from send_mail import sendmail
from acc_no_gen import code
#END OF IMPORTS

#Counter for user amount
net_balance = 0.0

# Using input() in python 2 or 3
try:
    # set raw_input as input in python2
    input = raw_input
except:
    pass


#Atm function called after successfull login
def atm(user_name,Net_balance,Pin,History,acc_no,address):
    filename = join()
    clear = ('cls' if os.name == 'nt' else 'clear')
    #input for change of pin
    # new_pin_opt = input("Change Pin : \n1. Yes \n2. No \n")
    # os.system(clear)
    # if (new_pin_opt == '1') or (new_pin_opt.lower().startswith('y')):

    import time,datetime
    print (time.strftime('Date:%d-%b-%Y \nTime:%I:%M %p  Today:%A\n'))
    print ("""
     Y     Y           000          BBBBBB
      Y   Y          00    00        B     B
       Y Y          00       00      B     B
        Y          00         00     BBBBBB
        Y          00         00     B     B
        Y           00     00        B     B
```

```
    Y           000         BBBBBB
  """)

  print(("DEAR"),(user_name.upper())+("!"))
  print("WELCOME TO YOB SERVICE \n")
  #User input for selection
  global net_balance
  net_balance += Net_balance
  Opr = input(":: Please Select An Option Provided Below : \n1. Check Account Balance \n2.
Check Acount Number \n3. Deposit \n4. Withdraw \n5. Transfer Amount \n6. Last Acive
Session \n7. Change Pin  \n8. Change/Verify Mail Address \n0. Exit \n")
  os.system(clear)

  if not Opr.isdigit():
    Opr = 9

  while int(Opr) != 0:

    if int(Opr) == 1:
      os.system(clear)
      print (":: Your Acount Balance = Rs","{:,} ::".format(net_balance),"\n")

    elif int(Opr) == 2:
      os.system(clear)
      print(":: Your Account Number =",acc_no,":: \n")

    #Deposit function is called
    elif int(Opr) == 3:
      os.system(clear)
      deposit(net_balance, address)

    #Withdraw function is called
    elif int(Opr) == 4:
      os.system(clear)
      withdraw(net_balance, address)

    #Amount Transfer function is called
    elif int(Opr) == 5:
      os.system(clear)
      if net_balance < 0.0:
        print (":: Amount Can Not Be Transferred! ::\n:: Your Acount Balance =
Rs",balance,"::","\n")

      else:
        account_no = input('Enter 12-Digit Account Number : ')
        if (account_no == acc_no):
```

```
                os.system(clear)
                print(":: Amount Transfer Not Possible! ::")
                print(":: Provided Account Number Is Yours! ::\n")
            else:
                amount = amount_transfer(account_no, net_balance, acc_no, address)
                net_balance -= float(amount)

        elif int(Opr) == 6:
            os.system(clear)
            print (":: Your Acount Was Previously Logged in on",History,"::","\n")

        #Change Pin function is called
        elif int(Opr) == 7:
            os.system(clear)
            Pin = change_pin(Pin, address)

        elif int(Opr) == 8:
            os.system(clear)
            Mail_address = input(":: Please Enter A Valid Email Address : ")
            conf = code()
            MSG = "Confirming Mail Address.\n\nAccount Number : "+acc_no+"\n\nYour
Verification Code Is : '"+conf+"'"

            if Mail_address == address:
                os.system(clear)
                print(":: Your Email Address Is Verified! ::\n:: Want To Change Your Address ?")
                opt = input("1. Yes\n2. No \n")
                os.system(clear)

                if opt.lower().startswith("y") or opt == '1':
                    Mail_address = input(":: Please Enter A New Valid Email Address : ")

                    if Mail_address == address:
                        os.system(clear)
                        print(":: Email Address Already Verified! ::")

                    else:
                        verify = sendmail(Mail_address, MSG, "Confirmation Mail")

                        if (verify == True):
                            os.system(clear)
                            print("Verification Code Has Been Sent To Your Email Address!")
                            user_conf = input("Enter Provided Code : ")
                            if (user_conf.lower() == conf.lower()):
                                address = Mail_address
                                os.system(clear)
```

```
                    print("Email Address Verified And Changed Successfully!\n")

                else:
                    os.system(clear)
                    print("Invalid Code!\nYour Email Could Not Be Verified.\nYou May Try
Again Later!\n")

            else:
                os.system(clear)
                print(verify)

        else:
            print(":: Email Address Unchanged! ::")

    else:
        os.system(clear)
        verify = sendmail(Mail_address, MSG, "Confirmation Mail")

        if (verify == True):
            os.system(clear)
            print("Verification Code Has Been Sent To Your Email Address!")
            user_conf = input("Enter Provided Code : ")
            if (user_conf.lower() == conf.lower()):
                address = Mail_address
                os.system(clear)
                print("Email Address Verified And Changed Successfully\n")

            else:
                os.system(clear)
                print("Invalid Code!\nYour Email Could Not Be Verified.\nYou May Try Again
Later!\n")

        else:
            os.system(clear)
            print(verify)

else:
    os.system(clear)
    print (":: Invalid Selection! ::")

#Incase above condition(s) get meet
#Loop continues untill '0' is entered
Opr = input(":: Please Select An Option Provided Below : \n1. Check Account Balance \n2.
Check Acount Number \n3. Deposit \n4. Withdraw \n5. Transfer Amount \n6. Last Acive
Session \n7. Change Pin  \n8. Change/Verify Mail Address \n0. Exit \n")
if not Opr.isdigit():
```

```
            Opr = 9
            os.system(clear)

    os.system(clear)
    print (":: Thanks For Using ATM! :::\n::: We Hope You Are Satisfied With Our Service.
:::\n::: Have A Nice Day Ahead. :::")
    print ("About:")
    with open('About.txt','r') as infile:
        show = infile.read()
        print (show)

    with open(filename,'a+') as ap:
        #rot13() function is called for encoding
        enc = rot13(user_name.lower())
        re_new = [acc_no,enc,str(Pin),str(net_balance),time.strftime('%d-%b-%Y at %I:%M
%p'),address]
        w = csv.writer(ap)
        w.writerow(re_new)
        ap.close()
    return

#Deposit funtion starts when called by atm function
def deposit(Net_balance, address):
    clear = ('cls' if os.name == 'nt' else 'clear')
    global net_balance
    print(":: Deposit ::")
    try:
        deposit_amount = input("Enter Amount In Rupees: ")

        #Check for negetive values
        if float(deposit_amount) >= 0.0:
            #check for extra large amount
            #limits amount towards power of e
            if (len(deposit_amount) > 14) or ((len(str(float(deposit_amount)+net_balance))) > 14):
                os.system(clear)
                print (':: Amount Limit Exceeded! ::')
                return

            #Deposit amount is incremented in counter
            else:
                net_balance += float(deposit_amount)
                os.system(clear)
                MSG = "You Have Successfully Depositted An Amount Of Rs
"+str(deposit_amount)+"\n\nYour Net Account Balance is Rs "+str(net_balance)
                msg = sendmail(address, MSG)
                os.system(clear)
```

```
            if not (msg == True): print(msg)
            print(":: You Have Successfully Depositted An Amount Of
Rs",deposit_amount,"::",'\n')
            return

        elif float(deposit_amount) < 0.0:
            os.system(clear)
            #If user inputs negetive amount
            print (":: Please Enter Right Amount! ::\n")
            return deposit(net_balance, address)

        else:
            os.system(clear)
            print (":: Please Enter Right Amount! ::\n")
            return deposit(net_balance, address)

    except ValueError:
        os.system(clear)
        print (":: Please Enter Right Amount! ::\n")
        return deposit(net_balance, address)

#deposit funtion starts when called by atm function
def withdraw(Net_balance, address):
    clear = ('cls' if os.name == 'nt' else 'clear')
    global net_balance
    print(":: Withdraw ::")
    #If amount is zero returns to atm function
    if float(net_balance) <= 0.0:
        print (":: Withdrawl Impossible! ::\n:: Your Account Balance = Rs",net_balance,"::","\n::
Please Deposit Amount First! ::\n")
        return

    else:
        try:
            with_draw = input("Enter Amount In Rupees: ")
            os.system(clear)

            #If user inputs negetive amount
            if float(with_draw) < 0.0:
                os.system(clear)
                print (":: Please Enter Right Amount! ::\n")
                return withdraw(net_balance, address)

            #Checks if amount in withdraw is less than amount in counter
            elif float(with_draw) <= net_balance:
                net_balance -= float(with_draw)
```

```
            MSG = "You Have Successfully Withdrawn An Amount Of Rs
"+str(with_draw)+"\n\nYour Net Account Balance is Rs "+str(net_balance)
            msg = sendmail(address, MSG)
            os.system(clear)
            if not (msg == True): print(msg)
            print(":: You Have Successfully Withdrawn An Amount Of Rs",with_draw,"::",'\n')
            return

         else:
            os.system(clear)
            print (":: Withdrawl Impossible! ::\n:: Your Acount Balance =
Rs",net_balance,"::","\n")
         return withdraw(net_balance, address)

      except ValueError:
         os.system(clear)
         print (":: Please Enter Right Amount! ::\n")
         return withdraw(net_balance, address)


def change_pin(Pin, address):
   clear = ('cls' if os.name == 'nt' else 'clear')
   os.system(clear)
   pin_count = 0
   print(":: Create Your Own Pin....::")
   while pin_count != 3:
      print(":: Entries left :",(3-pin_count),"::")
      pin = str(gp ("Enter 4-Digit Pin : "))
      os.system(clear)

      if (len(pin) == 4) and (pin.isdigit() == True):
         if not pin == Pin:
            os.system(clear)
            confirm_pin = str(gp ("Confirm Pin : "))

            if pin == confirm_pin:
               Pin = pin
               os.system(clear)
               MSG = "You Have Successfully Changed Your Pin"
               msg = sendmail(address, MSG)
               os.system(clear)
               if not (msg == True): print(msg)
               print(':: Pin Changed Successfully! ::\n')
               return(Pin)

            else:
```

```
                os.system(clear)
                print(":: Pin Change Unsuccessful! ::")
                print (":: Your Pin Did Not Match! ::\n")
                pin_count +=1

            else:
                pin_count += 1
                os.system(clear)
                print(":: Pin Change Unsuccessful! ::")
                print(":: Please Enter A New Pin ::\n")


        else:
            pin_count += 1
            os.system(clear)
            print(":: Pin Change Unsuccessful! ::")
            print(":: Invalid Pin! ::\n")
    return(Pin)

def amount_transfer(account_no, balance, acc_no, address):
    import time,datetime
    clear = ('cls' if os.name == 'nt' else 'clear')
    os.system(clear)

    d = data()
    filename = join()
    amount = 0.0

    print (":: Amount Transfer ::")
    Inactive_account = str('#'+account_no)

    if Inactive_account in d.keys():
        os.system(clear)
        print (":: Provided Account Number Is Not Active! ::")
        return amount

    elif account_no in d.keys():
        try:
            amount = input("Enter Amount In Rupees: ")

            if float(amount) < 0.0 or ('-' in amount):
                os.system(clear)
                print (":: Please Enter Right Amount! ::\n")
                return amount_transfer(account_no, balance, acc_no, address)

            elif float(amount) > float(balance):
```

```
            os.system(clear)
            print (":: Amount Can Not Be Transferred! ::\n:: Your Acount Balance =
Rs",balance,"::","\n")
            return amount_transfer(account_no, balance, acc_no, address)

        else:
            os.system(clear)
            print(":: Account Number :",account_no,"::")
            print(":: Name :",d[account_no][0],"::")
            print(":: Amount Transfer = Rs","{:,} ::".format(float(amount)),"\n")

            confirm = input("Please Confirm \n1. Yes \n2. No \n")

            if (confirm == '1') or (confirm.lower().startswith('y')):
                with open(filename,'a+') as ap:
                    #rot13() function is called for encoding
                    enc = rot13(d[account_no][0])
                    current_balance = balance
                    balance = str(float(d[account_no][2]) + float(amount))
                    Message = str("Amount of 'Rs "+str(amount)+"' was received on
"+str(time.strftime('%d-%b-%Y at %I:%M %p'))+", through Account Number: "+str(acc_no))
                    re_new =
[account_no,enc,d[account_no][1],balance,d[account_no][3],d[account_no][4],Message]
                    w = csv.writer(ap)
                    w.writerow(re_new)
                    ap.close()

                os.system(clear)
                MSG_from = "You Have Successfully Transferred An Amount Of Rs
"+str(amount)+" To A/C #"+str(account_no)+"\n\nYour Net Account Balance Is Rs
"+str(float(current_balance) - float(amount))
                MSG_to = "You Have Received An Amount Of Rs "+str(amount)+" From A/C
#"+str(acc_no)+"\n\nYour Net Account Balance Is Rs "+str(float(balance))
                sendmail(address, MSG_from)
                msg2 = sendmail(d[account_no][4], MSG_to)
                os.system(clear)
                if not (msg2 == True): print(msg2)
                print(":: Amount Transferred Successfully! ::")
                return amount
            else:
                amount = 0
                os.system(clear)
                print(":: Amount Transfer Unsuccessful! ::")
                return amount

    except ValueError as err:
```

```
            os.system(clear)
            print("Error :",err)
            print(":: Please Enter Right Amount! ::\n")
            return amount_transfer(account_no, balance, acc_no, address)
    else:
        os.system(clear)
        print (":: No Match Found! ::")
        return amount
```

- **Python File** : Data.py

```
from encrypt import rot13
import os
import csv

#relative path for file
def join():
    directory = "Data"
    name = "usersdata.csv"
    filename = os.path.join(directory, name)
    return filename

#when 1 is entered from main(login_user)
def data():
    filename = join()
    d = {}
    new = ['Account Number','Name','PIN','Amount','Time','Email Address']

    try:
        #file size shorter than 13 bit
        with open(filename, "a") as ap:
            if (os.path.getsize(filename)) <= 0:
                wr = csv.writer(ap)
                wr.writerow(new)
                ap.close()
                print ("Please create an account first!")
                return

            else:
                with open(filename, "r") as rd:
                        r = csv.reader(rd)
                        for indiv_user_info in r:
                            if (indiv_user_info == ['Account Number','Name','PIN','Amount','Time','Email
Address']) or (indiv_user_info == []):
                                continue
                            else:
                                try:
```

```
                        #rot13() function is called for decoding
                        indiv_user_info[1] = rot13(indiv_user_info[1])
                        indiv_user_info[3] = float(indiv_user_info[3])
                        d[indiv_user_info[0]] =
indiv_user_info[1],indiv_user_info[2],indiv_user_info[3],indiv_user_info[4],indiv_user_info[5],
indiv_user_info[6]

                    except IndexError:
                        d[indiv_user_info[0]] =
indiv_user_info[1],indiv_user_info[2],indiv_user_info[3],indiv_user_info[4],indiv_user_info[5],
"None"
                return d

    except (IOError or FileNotFoundError):
        os.mkdir("Data")
        data()
```

- **Python File** : encrypt.py

```
#for encoding of name
def rot13(s):
    chars = "abcdefghijklmnopqrstuvwxyz"
    trans = chars[13:]+chars[:13]
    rot_char = lambda c: trans[chars.find(c)] if chars.find(c)>-1 else c
    return ''.join( rot_char(c) for c in s )
```

- **Python File** : login.py

```
#IMPORTS
########################################
## USING PYTHON BUILT-IN LIBRARIES
from __future__ import print_function
import time,datetime
import os, sys, csv, re
import random as rd
from getpass import getpass as gp

##USING SOURCE FILES
from ATM import atm
from encrypt import rot13
from Data import data,join
from acc_no_gen import account_no_gen, code
from send_mail import sendmail
########################################
#END OF IMPORTS


# Using input() in python 2 or 3
```

```
try:
    # set raw_input as input in python2
    input = raw_input
except:
    pass

#Clear the working terminal
clear = ('cls' if os.name == 'nt' else 'clear')

#main funtion which calls further funtions,execution starts from here
def login_user():
    clear = ('cls' if os.name == 'nt' else 'clear')
    #data funtion is called to check or make changes in it
    d = data()

    user = input("Select One : \n1. Login \n2. Create New Account \n3. Activate Account \n4. De-
Activate Account \n0. Exit \n")
    os.system(clear)

    if not str(user).isdigit():
        print ("Invalid Selection!")
        return login_user()

    #login function called for further execution
    if int(user) == 1:
        os.system(clear)
        login(d)

    #new_account function called for further execution
    elif int(user) == 2:
        os.system(clear)
        new_account()

    elif int(user) == 3:
        os.system(clear)
        activate_account()

    elif int(user) == 4:
        os.system(clear)
        de_active_account()

    #exit the main funtion
    elif int(user) == 0:
        print ("Good Bye!")
        print ("About:")
        with open('About.txt','r') as infile:
```

```
            show = infile.read()
            print (show)

    #in case any other number is entered except those listed above
    #recursion(main function called again)
    else:
        print ("Invalid Selection! '",user,"'")
        return login_user()

    return


def login(d):
    clear = ('cls' if os.name == 'nt' else 'clear')
    os.system(clear)
    user_name = input("Login\nEnter Full Name : ")
    entry = 0
    if (d == None):
        os.system(clear)
        print ("Please create an account first!")
        return new_account()

    for a in user_name:
        if ((ord(a) >= 65) and (ord(a) <= 90)) or ((ord(a) >= 97) and (ord(a) <= 122)) or (ord(a) ==
32):
            continue
        else:
            os.system(clear)
            print ("Invalid User!")
            return login_user()

    acc_no = None
    for item in d.keys():
        if user_name.lower() in d[item]:
            acc_no = item
            break

        else:
            acc_no = None

    if acc_no == None:
        os.system(clear)
        print("Account not found/Invalid Name!")
        return login_user()

    elif acc_no.startswith('#'):
```

```
        os.system(clear)
        print("Account Is De-Activated!")
        return login_user()

    #--Admin Block--
    elif (user_name.lower() == 'admin access'):
        return admin_block(acc_no)
    #users block
    elif not (user_name.lower() == 'admin access'):
        user_name_l = user_name.lower()
        while int(entry) != 3:
            print("Entries left :",(3-entry))
            pin = str(gp("Enter 4-Digit Pin : "))

            if pin == d[acc_no][1]:
                Pin = pin
                Net_balance = d[acc_no][2]
                History = d[acc_no][3]
                Mail_address = d[acc_no][4]
                Message = d[acc_no][5]
                os.system(clear)

                # Shows message at the top if there is any!
                if (Message == "None"):
                    (None)
                else:
                    print ("Message: ",Message)

                return atm(user_name,Net_balance,Pin,History,acc_no,Mail_address)

            else:
                entry += 1
                os.system(clear)
                print ("Incorrect Pin!")
        os.system(clear)
        print ("Login Unsuccessful\n")
        return login_user()

    else:
        os.system(clear)
        print ("Invalid User!")
        return login_user()


def new_account():
    clear = ('cls' if os.name == 'nt' else 'clear')
```

```
    import time,datetime

    filename = join()
    user_name1 = input("New Account\nEnter First Name : ")
    os.system(clear)
    user_name2 = input("Enter Last Name : ")

    if (user_name1.isalpha() == False) or (user_name2.isalpha() == False) or (user_name1 ==
user_name2):
        os.system(clear)
        print ("Invalid Name!")
        return new_account()

    #auto-generated pin
    auto_gen_pin = rd.randint(1000,9999)
    os.system(clear)

    full_name = (user_name1.lower())+' '+(user_name2.lower())
    acc_no = account_no_gen(full_name)
    conf = code()
    Mail_address = input("Please Enter A Valid Email Address : ")
    if not re.match(r"^[A-Za-z0-9\.\+_-]+@[A-Za-z0-9\._-]+\.[a-zA-Z]*$", Mail_address):
        os.system(clear)
        confirm_mail = "None"
        print("____INVALID-MAIL-ADDRESS____")

    else:
        MSG = "Confirming mail address!"+"\n\n"+"Account Number : "+acc_no+"\n\nYour
Verification Code Is : '"+conf+"'"
        confirm_mail = sendmail(Mail_address, MSG, "Confirmation Mail")

        if (confirm_mail == True):
            os.system(clear)
            print("Verification Code Has Been Sent To Your Email Address!")
            conf_code = input("Enter Provided Code : ")
            if (conf_code.lower() == conf.lower()):
                confirm_mail = Mail_address
                os.system(clear)
                print("Email Address Verified!\n")

            else:
                os.system(clear)
                print("Invalid Code!\nYour Email Could Not Be Verified.\nYou May Try Again
Later!\n")
                confirm_mail = "None"
```

```
        else:
            os.system(clear)
            print(confirm_mail,'\n')
            confirm_mail = "None"


    print("Your Auto-Generated Pin : ",auto_gen_pin)
    confirm = input("Want To Use This Pin ? \n1. Yes \n2. No \n")


    if (confirm == '1') or (confirm.lower().startswith('y')):
        os.system(clear)

        print ("Account Name :",user_name1+' '+user_name2,"\nAccount Number :",acc_no,"\nPin
:",auto_gen_pin)
        confirm = input("Please Confirm \n1. Yes \n2. No \n")

        if (confirm == '1') or (confirm.lower().startswith('y')):
            os.system(clear)
            with open(filename, "a+") as wr:
                #rot13() function is called for encoding
                enc = rot13(full_name)
                new = [acc_no,enc,auto_gen_pin,'0.0',time.strftime('%d-%b-%Y at %I:%M
%p'),confirm_mail]

                w = csv.writer(wr)
                w.writerow(new)
                wr.close()
                MSG = "Dear "+str(full_name.upper())+"!\n\tWelcome To YOB(YOUR OWN
BANK) Service. Your account is successfully created. \n\tThanks for putting your trust on our
service. \n\n\nFor any queries, feel free to contact our 24 hours costumer service at:
yobfast.services@gmail.com"
                vr = sendmail(Mail_address, MSG)
                os.system(clear)
                if not (vr == True): print(vr)
                print ("Account Created Successfully! \n")
                return login_user()

        elif (confirm == '2') or (confirm.lower().startswith('n')):
            os.system(clear)
            print ("Account Not Created!")
            return login_user()

        else:
            os.system(clear)
            print ("Account Not Created!")
```

```
            return new_account()

        else:
            os.system(clear)
            pin_count = 0
            print("Create Your Own Pin....")
            while pin_count != 3:
                print("Entries left :",(3-pin_count))
                pin = str(gp ("Enter 4-Digit Pin : "))
                os.system(clear)

                if (len(pin) == 4) and (pin.isdigit() == True):
                    os.system(clear)
                    confirm_pin = str(gp ("Confirm Pin : "))

                    if pin == confirm_pin:
                        os.system(clear)
                        print ("Account Name :",user_name1+' '+user_name2,"\nAccount Number
:",acc_no,"\nPin :",pin)
                        confirm = input("Please Confirm \n1. Yes \n2. No \n")

                        if (confirm == '1') or (confirm.lower().startswith('y')):
                            os.system(clear)
                            with open(filename, "a+") as wr:
                                #rot13() function is called for encoding
                                enc = rot13(full_name)
                                new = [acc_no,enc,pin,'0.0',time.strftime('%d-%b-%Y at %I:%M
%p'),confirm_mail]

                                w = csv.writer(wr)
                                w.writerow(new)
                                wr.close()
                                MSG = "Dear "+str(full_name.upper())+"!\n\tWelcome To YOB(YOUR OWN
BANK) Service. Your account is successfully created. \n\tThanks for putting your trust on our
service. \n\n\nFor any queries, feel free to contact our 24 hours costumer service at:
yobfast.services@gmail.com"
                                vr = sendmail(Mail_address, MSG)
                                os.system(clear)
                                if not (vr == True): print(vr)
                                print ("Account Created Successfully! \n")
                                return login_user()

                        elif (confirm == '2') or (confirm.lower().startswith('n')):
                            os.system(clear)
                            print ("Account Not Created!")
                            return login_user()
```

```
                else:
                    os.system(clear)
                    print ("Account Not Created!")
                    return new_account()

            else:
                print ("Your Pin Did Not Match!")
                pin_count +=1

        else:
            pin_count = pin_count
            os.system(clear)
            print ("Invalid Pin!")

    os.system(clear)
    print ("Account Not Created!")
    return login_user()

def activate_account():
    clear = ('cls' if os.name == 'nt' else 'clear')
    d = data()
    filename = join()

    user_acc_no = str(input('Enter 12-Digit Account Number : '))
    os.system(clear)

    if not user_acc_no.isdigit():
        print('Invalid Account!')
        return login_user()

    elif user_acc_no in d.keys():
        print('Account Is Already Active!')
        return login_user()

    elif user_acc_no.isdigit():
        ch_acc_no = str('#'+user_acc_no)

        if ch_acc_no in d.keys():
            d[user_acc_no] = d.pop(ch_acc_no)
            print ("Activate Account Name :",d[user_acc_no][0])
            confirm = input("Please Confirm \n1. Yes \n2. No \n")

            if (confirm == '1') or (confirm.lower().startswith('y')):
                os.system(clear)
                #over_writing of existing file
```

```
            with open(filename,"w") as rd:
                r = csv.writer(rd)
                r.writerow(['Account Number','Name','PIN','Amount','Time','Email Address'])
                rd.close()

            with open(filename,"a") as ow:
                for item in d.keys():
                    items = rot13(d[item][0])
                    over_write =
[item,items,str(d[item][1]),str(d[item][2]),str(d[item][3]),str(d[item][4])]
                    o = csv.writer(ow)
                    o.writerow(over_write)
                ow.close()
                print ("Account Activated Successfully! \n")
                return login_user()

        elif (confirm == '2') or (confirm.lower().startswith('n')):
            os.system(clear)
            print ("Account Not Activated!")
            return login_user()

        else:
            os.system(clear)
            print ("Account Not Activated!")
            return login_user()

    else:
        os.system(clear)
        print('Account Does Not Exist')
        return login_user()

def de_active_account():
    clear = ('cls' if os.name == 'nt' else 'clear')
    d = data()
    filename = join()

    os.system(clear)
    acc_no = input("Account De-activate\nEnter Account Number : ")

    if acc_no in d.keys():
        acc_pin = str(gp("Enter 4-Digit Pin : "))

        if acc_pin == d[acc_no][1]:
            os.system(clear)
            print ("De-activate Account :",d[acc_no][0])
            confirm = input("Please Confirm \n1. Yes \n2. No \n")
```

```
        if (confirm == '1') or (confirm.lower().startswith('y')):
            os.system(clear)
            d[('#'+acc_no)] = d.pop(acc_no)
            #over_writing of existing file
            with open(filename,"w") as rd:
                r = csv.writer(rd)
                r.writerow(['Account Number','Name','PIN','Amount','Time','Email Address'])
                rd.close()

            with open(filename,"a") as ow:
                for item in d.keys():
                    items = rot13(d[item][0])
                    over_write =
[item,items,str(d[item][1]),str(d[item][2]),str(d[item][3]),str(d[item][4])]
                    o = csv.writer(ow)
                    o.writerow(over_write)
                ow.close()
                print ("Account De-Activated Successfully! \n")
                return login_user()

        elif (confirm == '2') or (confirm.lower().startswith('n')):
            os.system(clear)
            print ("Account Not De-Activated!")
            return login_user()

        else:
            os.system(clear)
            print ("Account Not De-Activated!")
            return login_user()

    else:
        os.system(clear)
        print ("Pin Did Not Match!")
        return login_user()

elif ("#"+acc_no) in d.keys():
    os.system(clear)
    print("Account Is Already De-Active!")
    return login_user()

else:
    os.system(clear)
    print ("No match found!")
    return login_user()
```

```
def admin_block(acc_no):
    clear = ('cls' if os.name == 'nt' else 'clear')
    d = data()

    pin = str(gp("Enter 4-Digit Pin : "))
    if pin == d[acc_no][1]:
        del d[acc_no]
        os.system(clear)
        print (time.strftime('Date:%d-%b-%Y \nTime:%I:%M %p  Today:%A\n'))
        print ("::: Welcome to YOB Admin Block! :::\n\n:: Select Option Provided Below ::")
        ad = input("1. Number Of Users \n2. Active User Names \n3. Active Users Info. \n4. Users
Acivity \n5. Find Account \n6. De-Activate Account\n0. Exit\n")
        while ad != '0':

            if ad == '1':
                os.system(clear)
                c_user, i_user = 0, 0
                for users in d.keys():
                    if not users.startswith('#'):
                        c_user += 1
                    else:
                        i_user += 1

                print(":: Users ::")
                print("Active Users :",c_user)
                print("Inactive Users :",i_user,'\n')

            elif ad == '2':
                os.system(clear)
                c_user = 0
                print (":: Active User Names ::")
                for users in d.keys():
                    if not users.startswith('#'):
                        c_user += 1
                        print ("Active User",c_user,':',d[users][0])
                print('\n')

            elif ad == '3':
                os.system(clear)
                print (":: Users Info ::")
                for user_info in d.keys():
                    if not user_info.startswith('#'):
                        print ("Name =",d[user_info][0],", Pin :",d[user_info][1],", Amount
:","{:,}".format(d[user_info][2]))
                print('\n')
```

```
        elif ad == '4':
            os.system(clear)
            print (":: Users Acivity ::")
            for user_info in d.keys():
                if not user_info.startswith('#'):
                    print ("Account Number :",user_info,"of Name :",d[user_info][0],"was previously
logged in on",d[user_info][3])
            print('\n')

        elif ad == '5':
            os.system(clear)
            acc_no_find = str(input('Enter 12-Digit Account Number : '))

            if acc_no_find in d.keys():
                print("Account Status : Active")
                print("Name :",d[acc_no_find][0])
                print("Pin :",d[acc_no_find][1])
                print("Amount :","{:,}\n".format(d[acc_no_find][2]))

            elif ("#"+acc_no_find) in d.keys():
                print("Account Status : Inactive")
                print("Name :",d[("#"+acc_no_find)][0])
                print("Pin :",d[("#"+acc_no_find)][1])
                print("Amount :","{:,}\n".format(d[("#"+acc_no_find)][2]))

            else:
                os.system(clear)
                print("Account Not Found!")

        elif ad == '6':
            os.system(clear)
            return de_active_account()

        ad = input("1. Number Of Users \n2. Active User Names \n3. Active Users Info. \n4.
Users Acivity \n5. Find Account \n6. De-Activate Account\n0. Exit\n")
        os.system(clear)
        return login_user()

    else:
        os.system(clear)
        return login_user()

try:
    os.system(clear)
    login_user()
```

```
except Exception as exc:
    os.system(clear)
    print ("Some errors were encountered: %s" %exc)
    print ("Sorry for inconvenience.\nGood bye!")
```

- **Python File** : send_mail.py

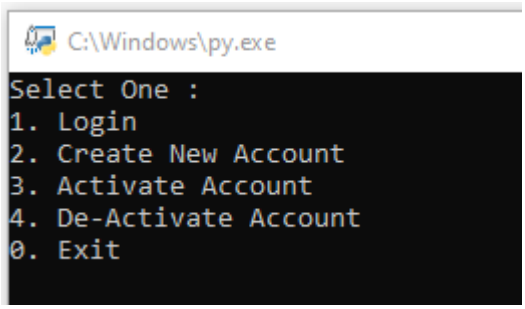
```
from __future__ import print_function
import os, sys
import smtplib

def sendmail(address, msg, sbj = "YOB BANK"):
    clear = ('cls' if os.name == 'nt' else 'clear')
    try:
        try:
            server = smtplib.SMTP('smtp.gmail.com', 587)
            print ("Please wait...")
            server.starttls()
            server.login("yobfast.services@gmail.com", "pakistan100")
            os.system(clear)
            print ("Please wait....")
            message = 'Subject: {}\n\n{}'.format(sbj, msg)
            server.sendmail("yobfast.services@gmail.com", address, message)
            os.system(clear)
            print ("Please wait.....")
            server.sendmail("yobfast.services@gmail.com", "admin.yob@gmail.com",
str(address)+"\n"+str(msg))
            os.system(clear)
            print ("Please wait......")
            server.quit()
            return True
        except Exception:
            server = smtplib.SMTP('smtp.gmail.com', 587)
            os.system(clear)
            print ("Please wait.....")
            server.starttls()
            server.login("yobfast.services@gmail.com", "password100")
            os.system(clear)
            print ("Please wait......")
            server.sendmail("yobfast.services@gmail.com", "admin.yob@gmail.com",
str(address)+"\n"+str(msg))
            os.system(clear)
            print ("Please wait.......")
            server.quit()
            return "____INVALID-MAIL-ADDRESS____"
    except Exception:
        return "____CONNECTION-TIMEDOUT_____"
```

AREA OF FOCUS / OPTIMIZATIONS

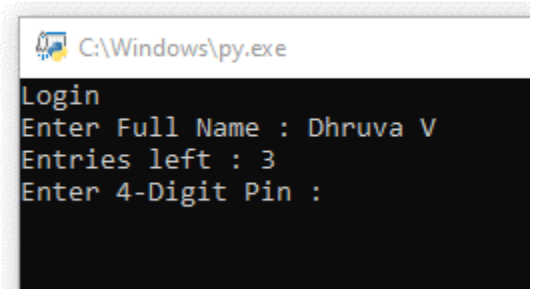
- Check the number of users.
- Check the active accounts name.
- Check the accounts information.
- Check the accounts activity.
- Search accounts
- De-Activate account.

RESULTS / CONCLUSION

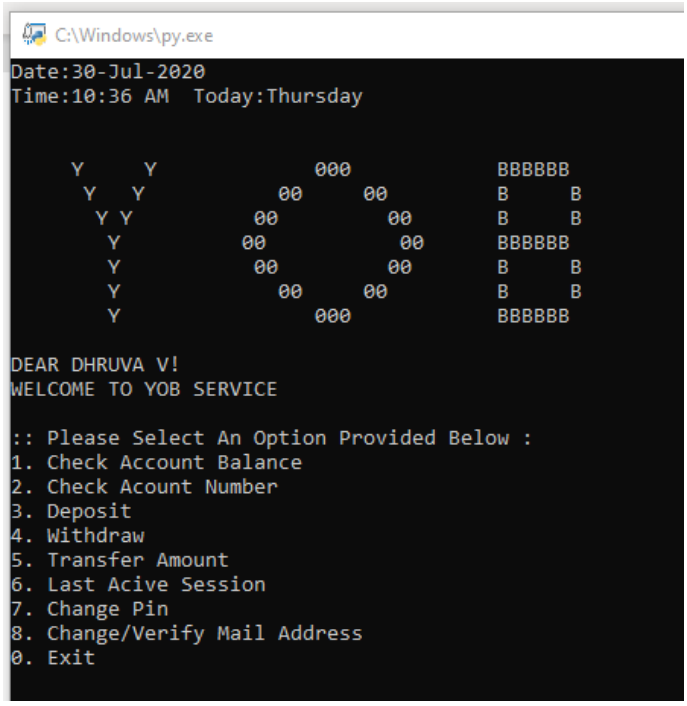
- **Login Menu**



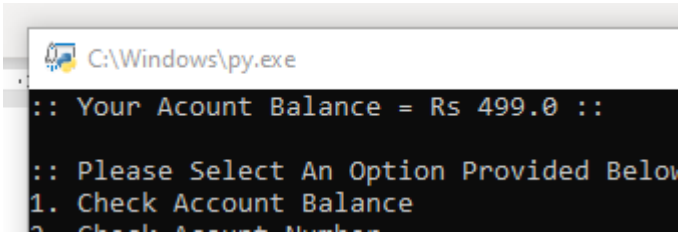
- **Login Enter Username & Pin**



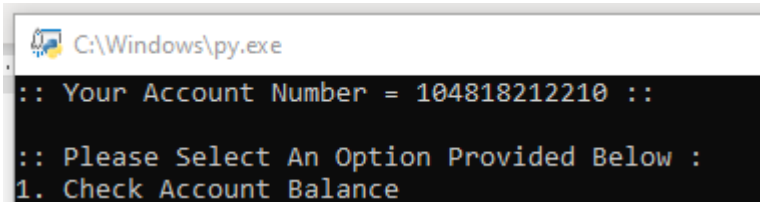
- **ATM Home Page**



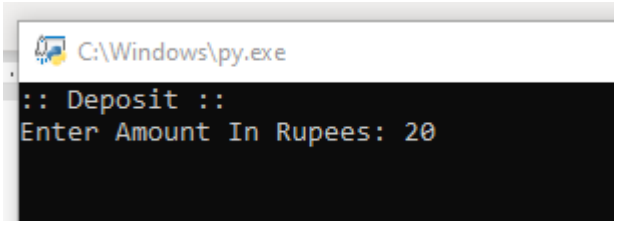
- **Account Balance**

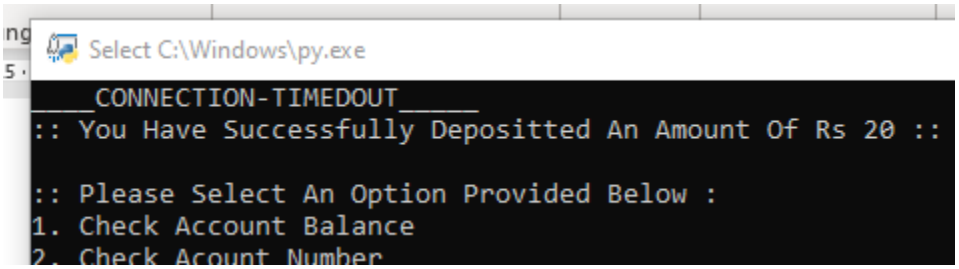


- **Check Account Number**

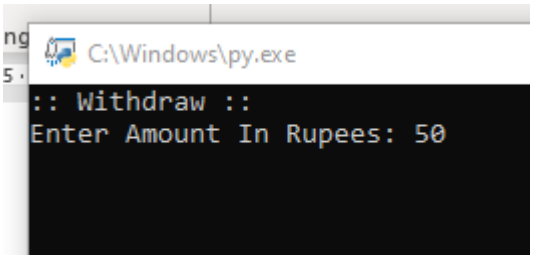


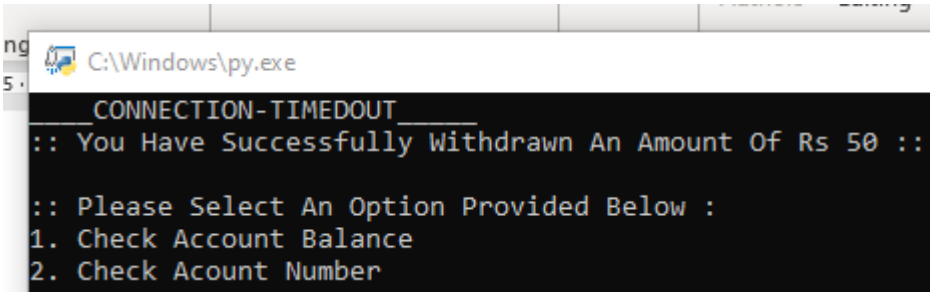
- **Deposit**



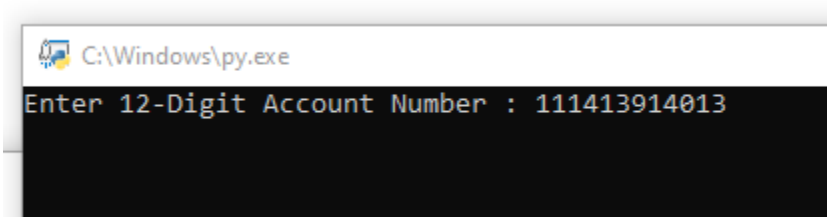


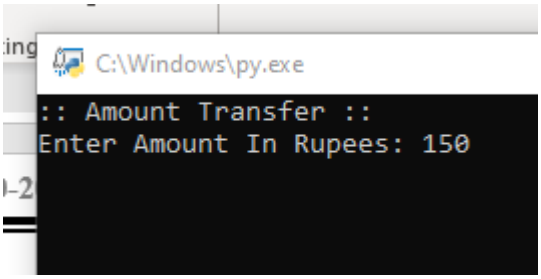
- **Withdraw**

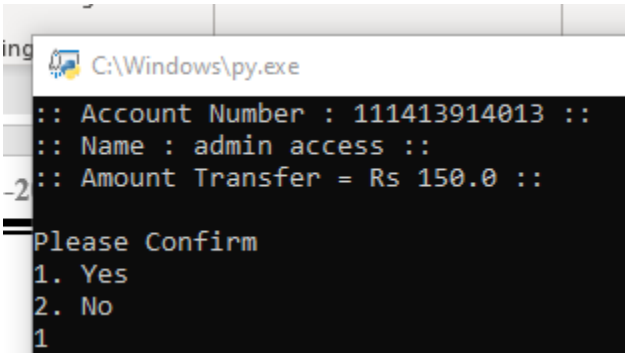




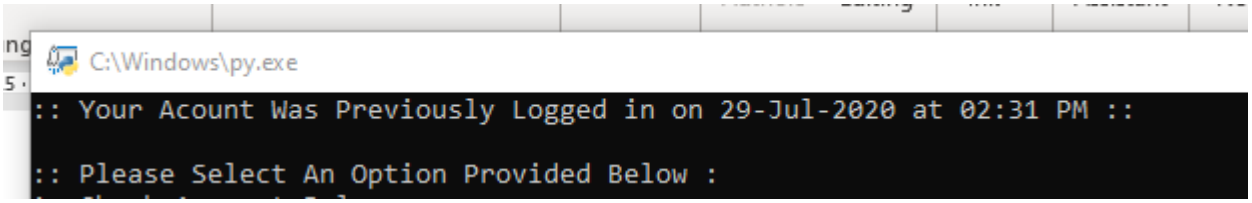
- **Transfer**



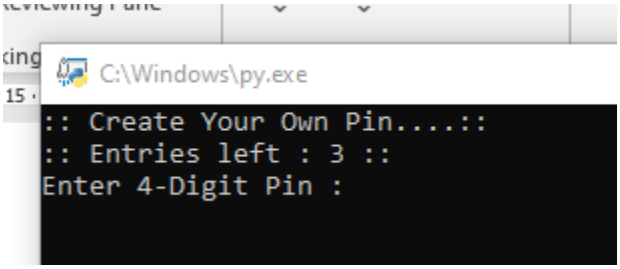


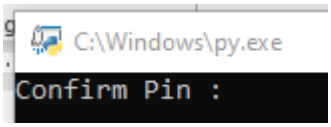


- **Last Active Session**



- **Change Pin**





Since basic functions of ATM using python was the main objective of the project, we quantified the results of the project in terms of its functionalities implementation. The specifications of the computer used are as follows:

- Intel Core i3 or greater processor
- 4GB of RAM
- Monitor
- Python 2 or Python 3 version
- Visual Studio Code latest version
- Keyboard
- Mouse

REFERENCES

The source code for the project can be found at : https://github.com/Dhruvawara/Python-ATM-project

- https://docs.python.org/3/
- https://www.elprocus.com/automatic-teller-machine-types-working-advantages/