# **Bank Application**

## Student Name:-

## Student Id:-

# **ABSTRACT:-**

This report includes the bank account management application. In this project, we are going to create a bank application through python using object-oriented programming. In this project, we fully use all the aim of objected-oriented programming like encapsulation, Inheritance and polymorphism.

# Introduction:-

In this project, we use the Object-Oriented programming method for developing this application. We also create the class and function for developing our applications so that it is easy to cover all the topics presented in brief like Encapsulation, Inheritance, and Polymorphism. There are various ways of creating this application but the best way to do is through objected-oriented programming. The Objected-oriented programming provides us a large range of solution of any problem and this application fulfill all the requirement of Object-oriented programming so it’s also easy and best way to do it through using OOPs in python.

# **Bank Application:-**

Bank Management System is a decent support based application that is worked in python. Fundamentally, this framework contains python content (main.py) and a database record in the task document. This framework is a straightforward comfort based framework so it is extremely simple to grasp and use. Discussing the framework, it incorporates all the key highlights required in a bank. There is no login framework as this is a scaled-down venture. Furthermore, this implies the client can utilize each one of those accessible highlights effectively with no limitations. It's too easy to even think about using, the client can examine the records of complete financial balance without any problem.

Essentially, this framework likewise gives the least complex administration of financial balance and exchange. In short, the fundamental focal point of these activities depends on CRUD. In this smaller than normal venture, for sparing a client information consistently, it utilizes an inner database association document.

# **Main Features of this Application:**

* Create a new account and view account holders record
* Edit account details.
* Users can create an account by providing the name of the account holder name.
* User can select the amount type (Deposited or Current account).
* They can also provide an initial amount which is must be greater than equal to 1euro.
* He/ She can modify their account detail if they want to.

# **Technologies Used In Project:-**

* Object-Oriented Programming.
* Encapsulation
* Inheritance
* Polymorphism

# **Planning Of Project:-**

This Project needs to well plan before its development is start. We first need to understand the requirement of the project. Then we can start the planning of the project according to the requirement of the project.

## Requirement of project:

We need to develop a bank application through python which has following facilities:-

* Create a new account and view account holders record
* Edit account details.
* Users can create an account by providing the name of the account holder name.
* User can select the amount type (Deposit or Current account).
* They can also provide an initial amount which is must be greater than equal to 1euro.
* She/he can modify their account detail if they want to.

As we have now the requirement of the application, we can start planning this project. First, we need to provide the main three features to the users like create account, existing users, and for banking staff. In the create account option we provide the user option of creating the new account in which the need provide their first name, last name, account type, and minimum balance. After taking this information we will create the account of the users with a unique account number, unique password, and pin. Now the new user data will be stored in the database file which is in .txt format. Now if the user wants to change their password and pin they need to log in as existing users. Also for banking staff, we provide various options like his/her will be able to view the account holder name and available balance in the existing user's details.

# **Pseudocode of bank application:-**

**Need to install in python before executing code**

def install\_and\_import(package):

import importlib

try:

importlib.import\_module(package)

except ImportError:

import pip

pip.main(['install', package])

finally:

globals()[package] = importlib.import\_module(package)

**Necessary modules need to import and install:**

install\_and\_import('string')

install\_and\_import('random')

**Code where class is creates:**

**class NewAccount(object):**

**#defining all the variables of class NewAccount in init function**

def \_\_init\_\_(self):

self.fname = ""

self.lname = ""

self.acc = ""

self.tobeshown = []

self.\_\_tobehidden = []…………..

**For creating Function:**

def PassPinGenerator(self,a):

self.\_\_pin = random.randint(1000,9999)

low = ['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z']

upp = ['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','R','S','T','U','V','W','X','Y','Z']

num=[0,1,2,3,4,5,6,7,8,9]

spe=['£','$','%','@','?','#']

def CreateAccount(self):

……………………………………..

**Creating Conditional Statement:**

while(flag==1):

try:

self.\_\_minbal = int(input("Minimum balance you would like to deposite(must be more than 1 euro and must be in integer) : "))

if self.\_\_minbal>=1:

flag=0

except:

print("the balance you enter must be in integer format and should not be 0")

line=""

with open('myfile.txt') as file:

for line in (file.readlines() [-1:]):

#print(line)……………………………

**Providing Optins to User’s:**

while(flag=='1'):

print("Enter 1 if you are new user")

print("Enter 2 if you are Existing user")

print("Enter 3 if you are banker staff")

option = int(input("Enter your choice : "))

The above are the pseudo code for developing applications.

# **Object-Oriented programming:-**

Item arranged Programming, or OOP for short, is a programming worldview that gives methods for organizing programs with the goal that properties and practices are packaged into singular articles.

For example, an item could speak to an individual with a name property, age, address, and so forth. With practices like strolling, talking, breathing, and running. Or then again an email with properties like beneficiary rundown, subject, body, and so forth. And practices like including connections and sending.

Put another way, object-situated writing computer programs is a methodology for demonstrating solid, certifiable things like vehicles just as relations between things like organizations and workers, understudies and educators, and so forth. OOP models true elements as programming objects, which have a few information related to them and can play out specific capacities.

Another regular programming worldview is procedural programming which structures a program like a formula in that it gives a lot of steps, as capacities and code squares, which stream consecutively so as to finish an undertaking.

The key takeaway is that articles are at the focal point of the item arranged programming worldview, not just speaking to the information, as in procedural programming, however in the general structure of the program too.

## Classes in Python

Focusing first on the data, everything or thing is an event of some class. The rough data structures open in Python, like numbers, strings, and records are expected to address fundamental things like the cost of something, the name of a piece, and your favored tints, exclusively. Think about how conceivable it is that you expected to address something altogether progressively puzzled. For example, assume you expected to follow different animals. If you used a summary, the fundamental part could be the animal's name while the ensuing segment could address its age. By what means may you know which part ought to be which? Think about how conceivable it is that you had 100 remarkable animals. Is it precise to state that you are certain each animal has both a name and an age, and so forth? Envision a situation where you expected to add various properties to these animals. This needs affiliation, and it's the particular necessity for classes. Classes are used to make new customers described data structures that contain abstract information about something. By virtue of an animal, we could make an Animal () class to follow properties about the Animal-like the name and age. It's basic to observe that a class just gives structure—it's a layout for how something should be described, yet it doesn't generally give any veritable substance itself. The Animal () class may discover that the name and age are imperative for portraying an animal, yet it won't generally state what a specific animal's name or age is. It may help with considering a class an idea for how something should be portrayed.

## Python Objects (Instances)

While the class is the framework, an event is a copy of the class with veritable characteristics, really an article having a spot with a specific class. It is definitely not an idea anymore; it's a genuine animal, like a canine named Roger who's eight years old. Put another way, a class takes after a structure or survey. It describes the necessary information. After you balance the structure, your specific copy is an event of the class; it contains genuine information material to you. You can balance different copies to make a wide scope of events, yet without the structure as a guide, you would be lost, not fathoming what information is required. Along these lines, before you can make solitary instances of an article, we ought to at first figure out what is required by describing a class.

### Instance Attributes

All classes make things, and all articles contain characteristics called qualities (insinuated as properties in the underlying entry). Use the \_\_init\_\_ () methodology to instate (e.g., decide) an article's fundamental characteristics by giving them their default worth (or state). This procedure must-have in any occasion one conflict similarly as the self variable, which suggests the article itself (e.g., Dog). By virtue of our Dog () class, each pooch has a specific name and age, which is unmistakably basic to know for when you start truly making different mutts. Remember: the class is just for portraying the Dog, not so much creation instances of individual pooches with express names and ages; we'll get to that in a split second.

Furthermore, the self variable is in like manner an instance of the class. Since instances of a class have varying qualities we could state Dog.name = name rather than self.name = name. Regardless, since not all mutts share a comparative name, we ought to have the alternative to consign different characteristics to different events. In this way the necessity for the remarkable self variable, which will help with checking particular instances of each class?

## Python Object Inheritance

Heritage is the technique by which one class expects the characteristics and methodologies for another. As of late molded classes are called kid classes, and the classes that youth classes are gotten from are called parent classes.

Note that adolescent classes override or expand the value (e.g., characteristics and practices) of parent classes. By the day's end, youth classes obtain the aggregate of the parent's attributes and practices yet can in like manner decide unmistakable direct to follow. The most basic kind of class is a thing, which all-around every single diverse class procure as their parent. Right when you portray another class, Python 3 it obviously uses the object as the parent class. So the goings with two definitions are practically identical.

### Parent vs. Child Classes



Figure 1 Parent Class



Figure 2 Function Of Parent Class

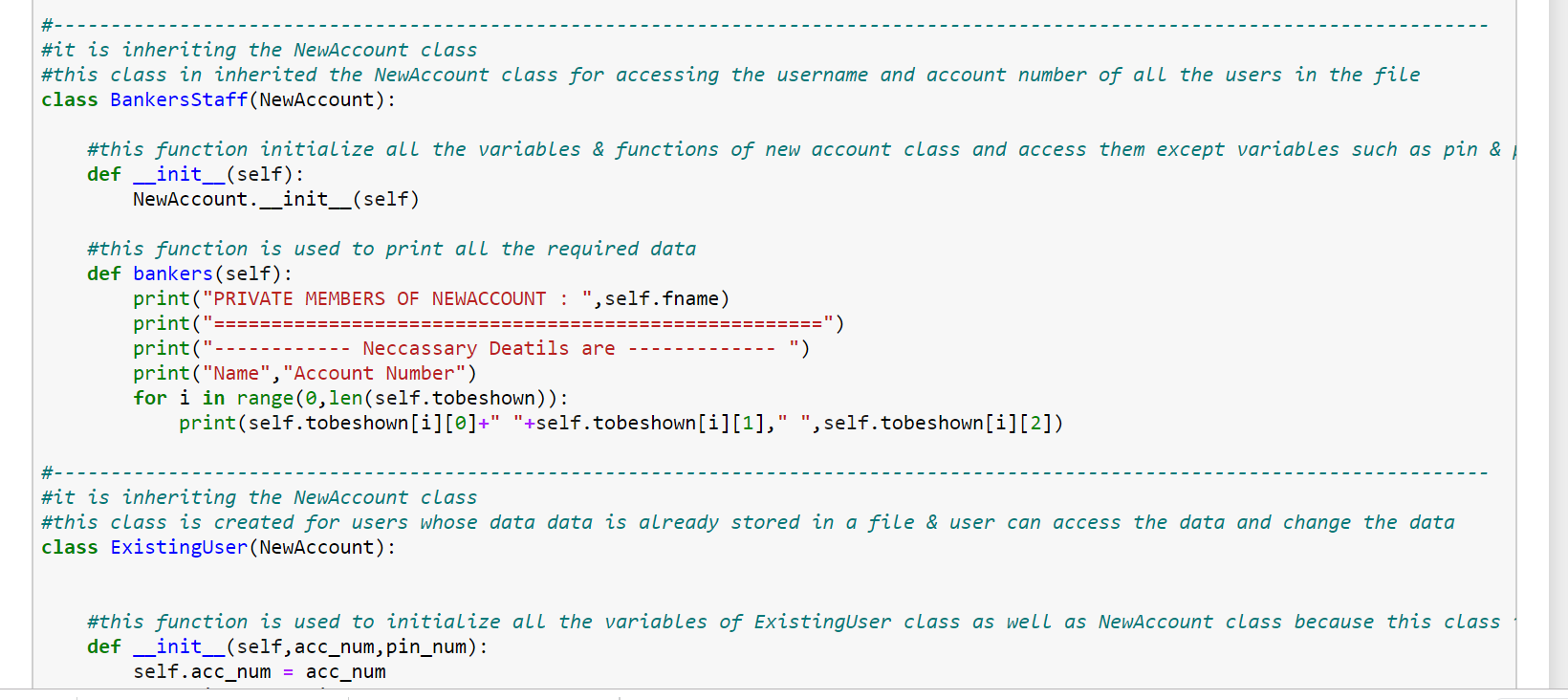


Figure 3 Child Class

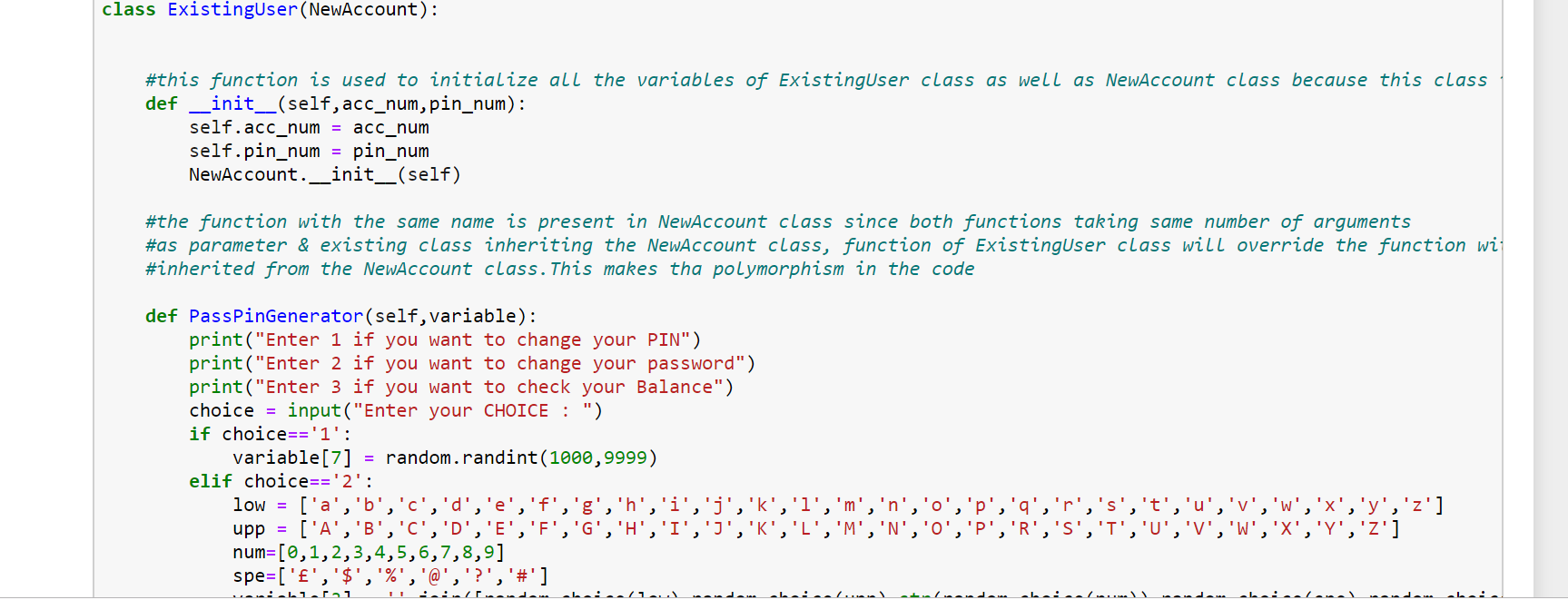


Figure 4 Function of Child Class

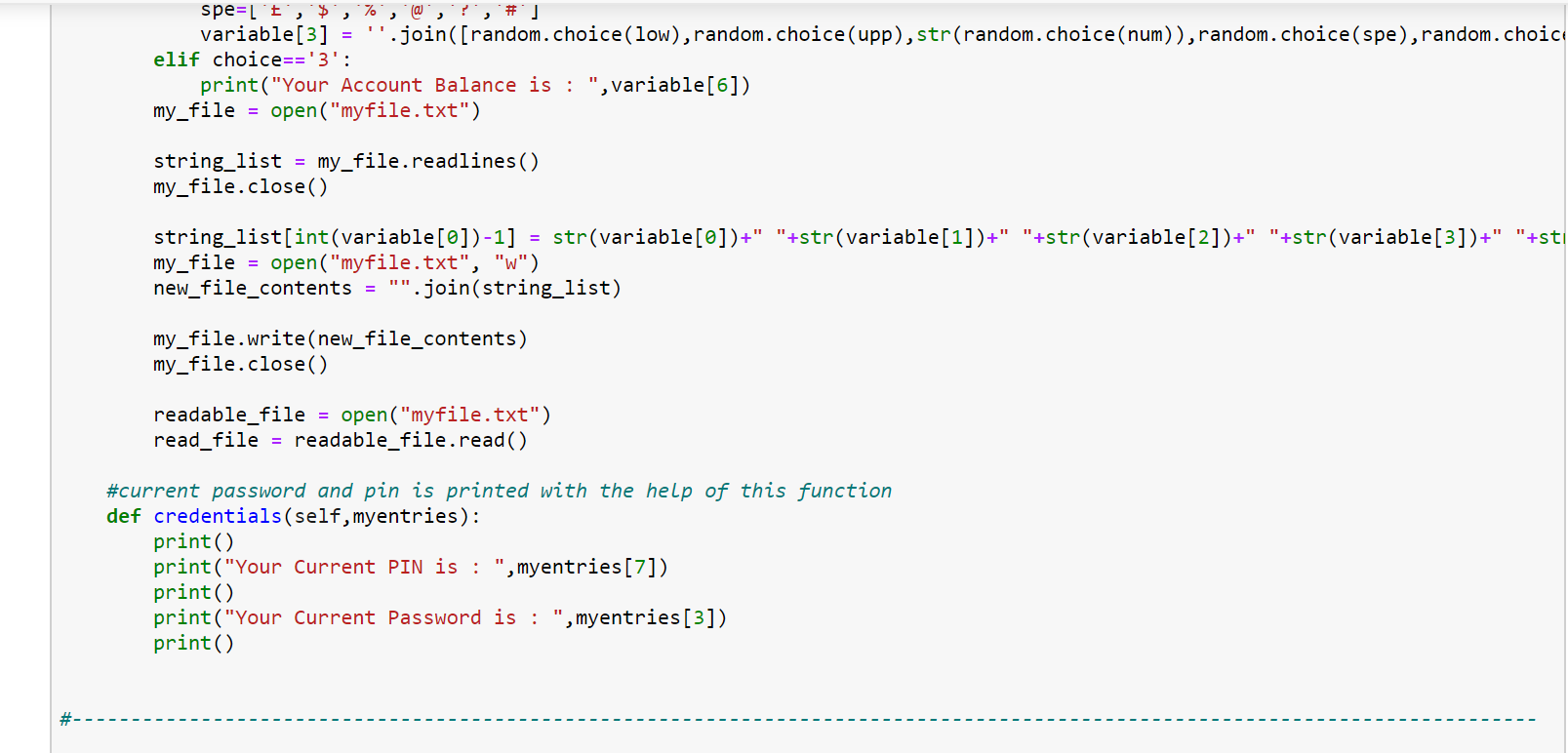


Figure 5 Conditional Statement in Child Class

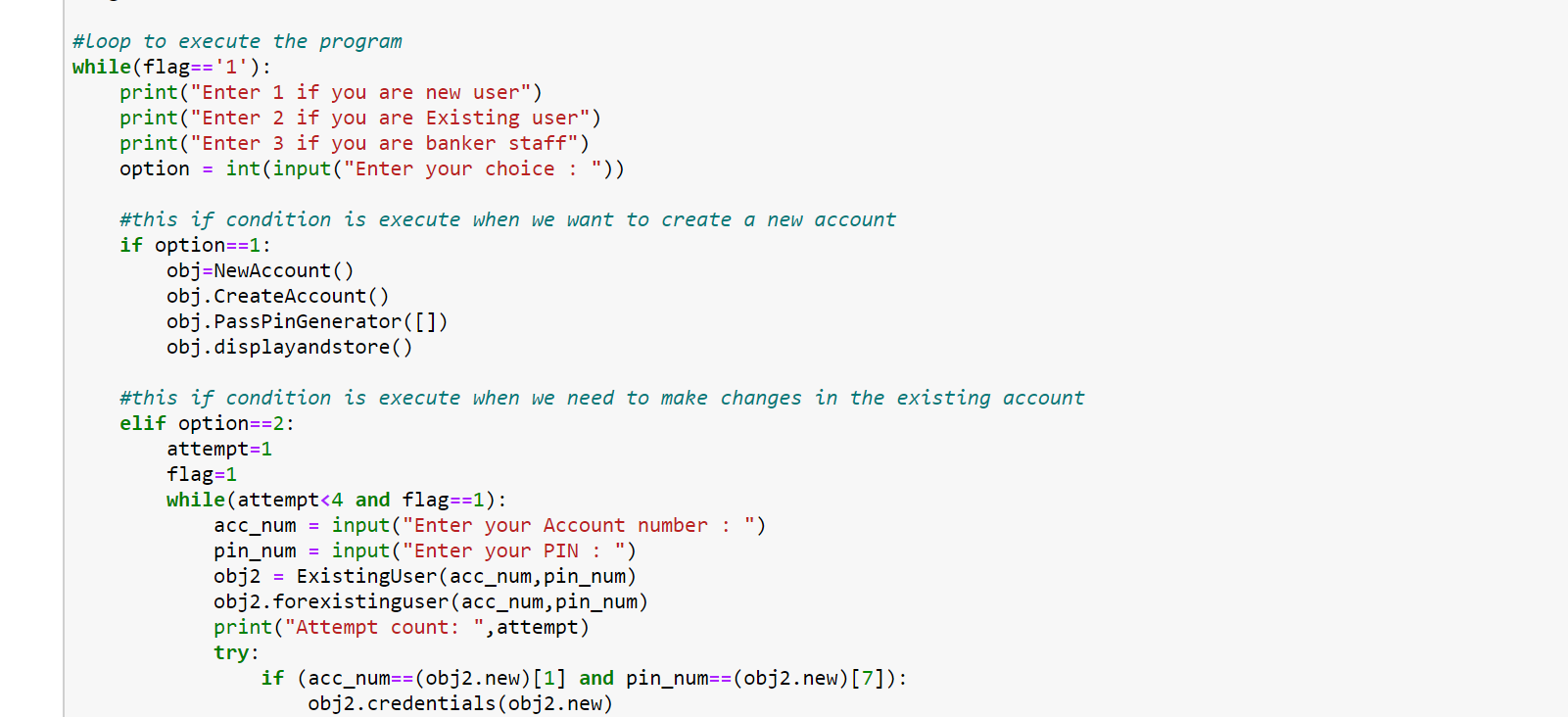


Figure 6 loop For Executing Code

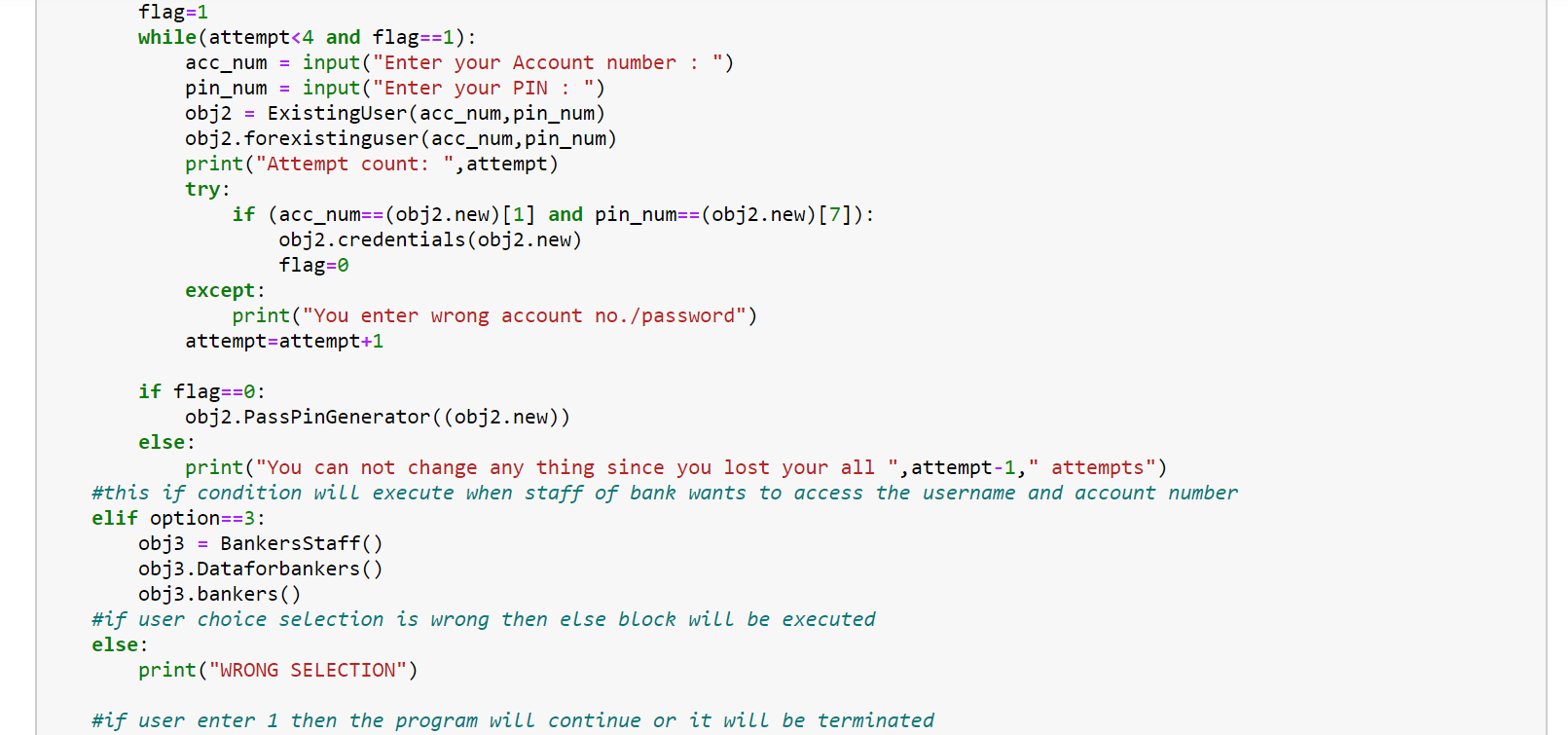


Figure 7 Validation Checking of User's

**Testing of Code:-**

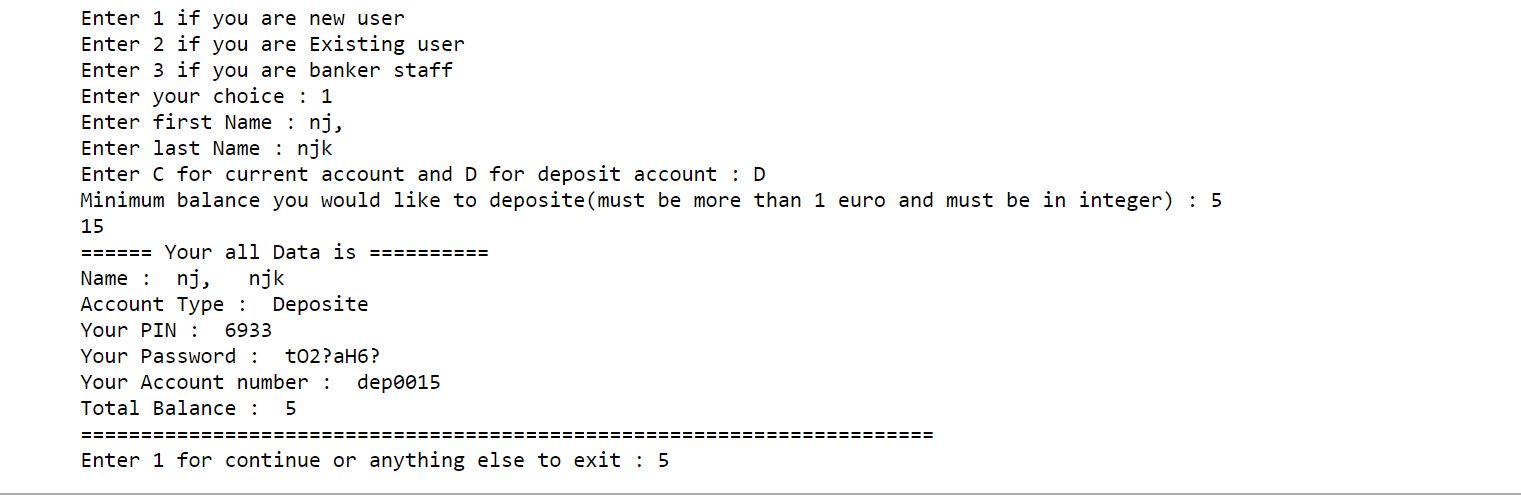
****

Figure 8 Creating New User's

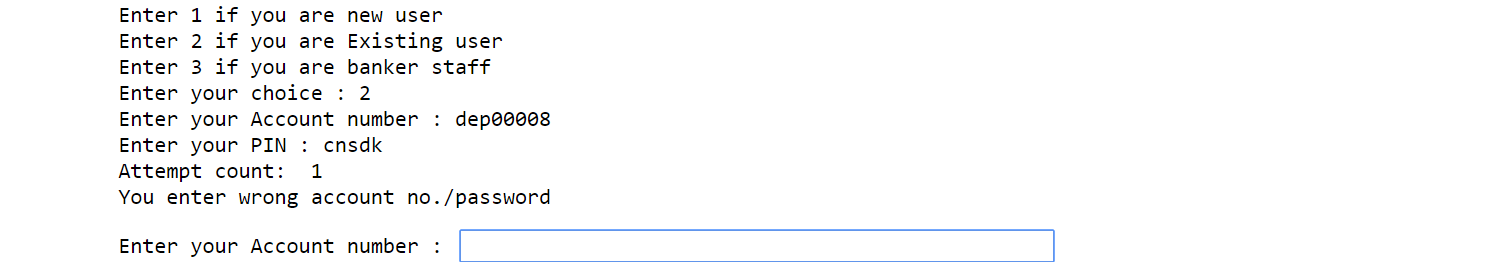


Figure 9 Checking of user's entry

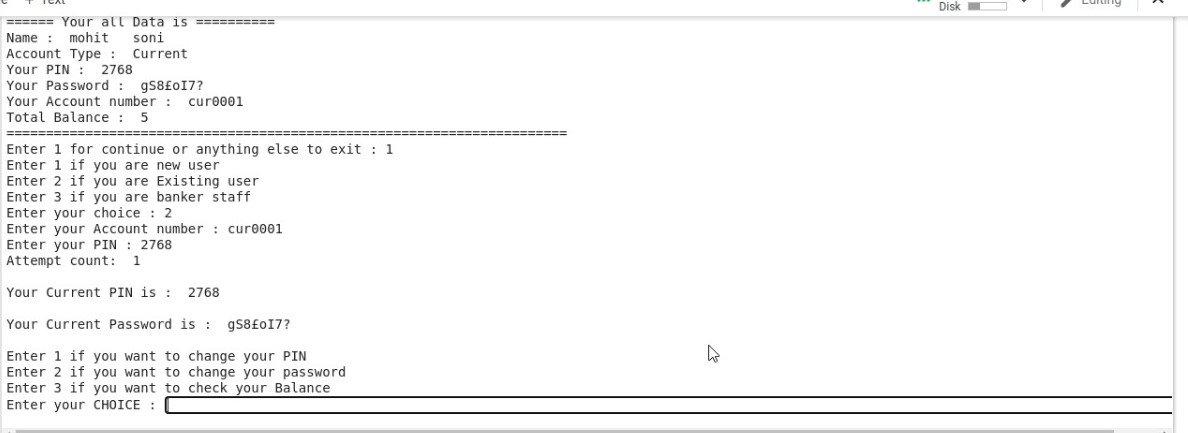


Figure 10 If Entry is right Showing details of user's

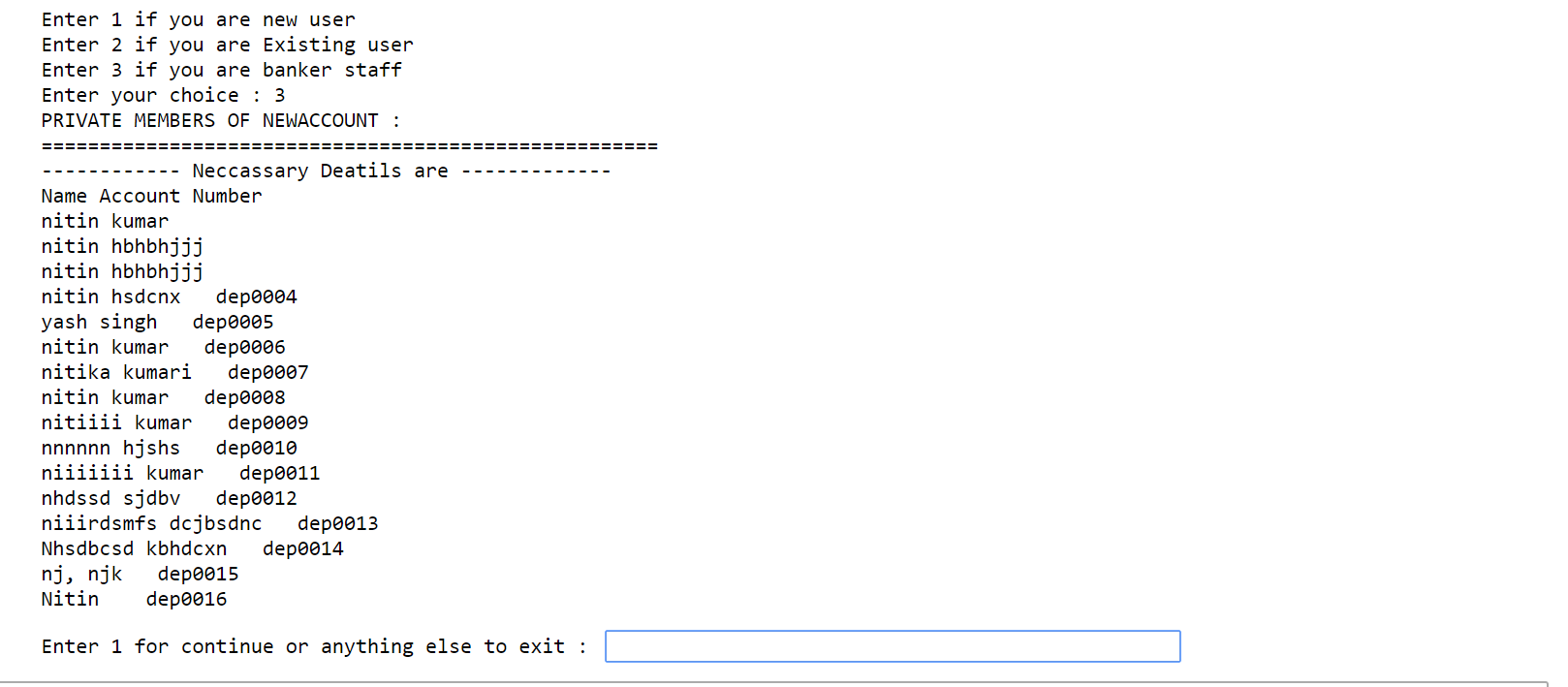


Figure 11 Details For Banking Staff

# **Conclusion:-**

Banking frameworks have been with us for whatever timeframe that individuals have been utilizing cash. Banks and other cash related affiliations offer security to people, affiliations, and governments, the comparable. We should recap what has been recognized with this instructional exercise: When everything is said in done, what banks do is absolutely simple to fathom. For the conventional individual banks perceive stores, impact advances, to give a protected spot to cash and assets, and go about as bit heads among merchants and banks. Banks are crucial to the economy and are connected with such budgetary exercises as giving cash, settling divides, credit intermediation, headway change, and cash creation as fragmentary hold banking. To get cash, banks use stores and entire plan stores, share worth and charges, and fervor from duty, advances, and purchaser credit, for example, MasterCard's and bank costs. In spite of charges and credits, banks are besides attracted to different sorts of progressing and activities including, purchase/hold protections, non-premium compensation, confirmation and renting, and segment treasury associations.

# **References:-**

Wu, T., and Liang, X., 2017, August. Exploration and practice of inter-bank application based on blockchain. In *2017 12th International Conference on Computer Science and Education (ICCSE)* (pp. 219-224). IEEE.

Pande, S., Mate, S., Mawal, P., Jambulkar, A. and More, N.S., 2018. E-blood bank application using cloud computing. *International Research Journal of Engineering and Technology (IRJET)*, *5*(2).

Made, S.P., Patil, B.D. and Holambe, R.S., 2020. Design of a frequency spectrum-based versatile two-dimensional arbitrary shape filter bank: application to contact lens detection. *Pattern Analysis and Applications*, *23*(1), pp.45-58.

Nebot, N., and Jere, N., 2018, September. Developing a time bank application for Havana informal settlement. In *Southern Africa Telecommunication Networks and Applications Conference (SATNAC), Western Cape, South Africa*.

Nishad, A., Pachori, R.B., and Acharya, U.R., 2018. Application of TQWT based filter-bank for sleep apnea screening using ECG signals. *Journal of Ambient Intelligence and Humanized Computing*, pp.1-12.

Shaddady, A., and Moore, T., 2019. Investigation of the effects of financial regulation and supervision on bank stability: The application of CAMELS-DEA to quantile regressions. *Journal of International Financial Markets, Institutions and Money*, *58*, pp.96-116.

Mohammedi, A., Rekioua, D., Rekioua, T. and Mebarki, N.E., 2018. Comparative assessment for the feasibility of storage bank in small scale power photovoltaic pumping system for building application. *Energy Conversion and Management*, *172*, pp.579-587.