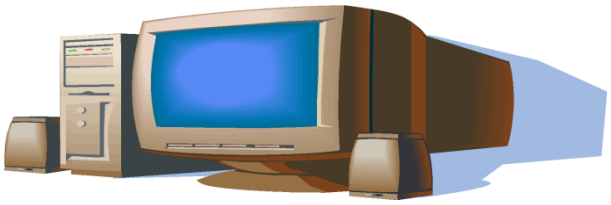
ST.XAVIER'S SCHOOL
HAZARIBAG

COMPUTER SCIENCE
PROJECT REPORT FILE
ON
BANK MANAGEMENT SYSTEM



For

AISSCE 2021-22 Examination

[As a part of the Computer Science Course(083)]

PROJECT PREPARED BY:

Ashish Kumar

XII A

Reg No.-

Roll No:-

Under the Guidance of

MR. SAMIR CHATTARAJ

PGT(COMPUTER SCIENCE)

TABLE OF CONTENTS

Certificate

This is to certify that Ashish Kumar of class twelve, St.Xavier's School, Hazaribagh has successfully completed his project under my guidance in computer practicals for the AISSCE as prescribed by CBSE in the year 2021-2022.

Date :

Registration No. :

Signature of Internal Examiner **Signature of External Examiner**

Acknowledgement

I thank my Computer Science teacher Mr. Samir Chattaraj for guidance and support. I also thank my Principal Father Rosner Xalxo. I would also like to thank my parents and my sister for encouraging me during the course of this project. Finally I would like to thank CBSE for giving me this opportunity to undertake this project.

Ashish Kumar
XII A

Introduction

This project is designed for keeping records of customer account and transaction as well as searching of transaction details and displaying. The title of the project is Bank Management system. Through this software bank can keep records of customer transactions and their information details. Daily transactions record can be maintained through this software..

The purpose of the project is to develop a program for a bank to provide user friendly interface for its user to keep information and records of the cutomers of his/her bank.

This software, being simple in design and working, does not require much of training to users and can be used

as a powerful tool for keeping records and doing transactions.During coding Spyder ide is used and the programming concept of python and database mysql is used in this project.

Objective and scope of the Project

The objective of this software is to develop a computerized Information system to automate the functions of Bank.This software is also aimed to enhance the current record keeping sysem, which will managers to retrieve the up-to-date information at right time at right place.
The Proposed software system is expected to do the following functionality-

- To Provide a user friendly Interface to its user.
- The proposed software will maintain all the records and transactions and will generate the required reports and information when required.

In its current scope , the software enables user to retrieve and update the information from database using mysql and python.

Despite of the best effort of the developer, the following limitations and functional boundaries are visible which limits the scope of this application software

1. This software can store records and produce reports in pre-designed format in soft copy. There is no facility yet to produce customized reports. Only specified reports are covered. (**Note- You can write more limitations according to your project**.)

Theoretical Background

What is Python?

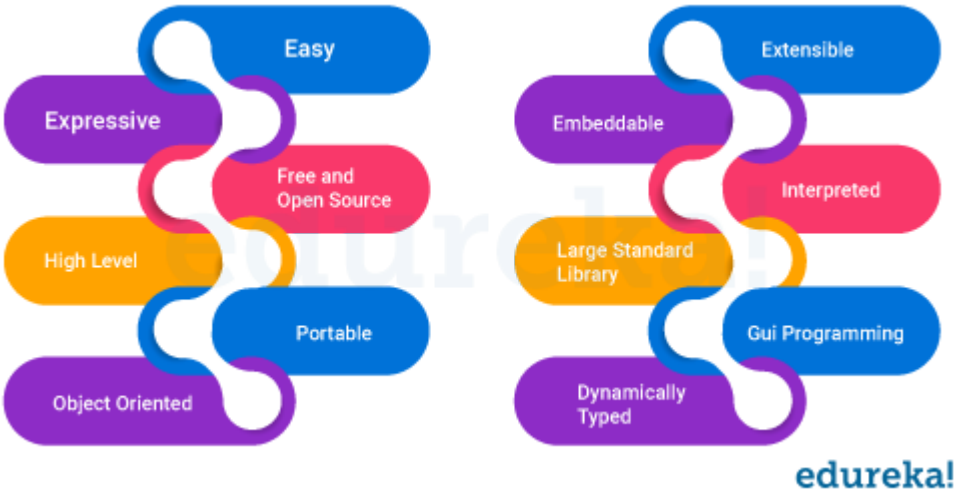
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest

way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Features of Python

As a programming language, the features of Python brought to the table are many. Some of the most significant features of Python are:



Easy to Code

Python is a very developer-friendly language which means that anyone and everyone can learn to code it in a couple of hours or days. As compared to other object-oriented programming languages like Java, C, C++, and C#, Python is one of the easiest to learn.

Open Source and Free

Python is an open-source programming language which means that anyone can create and contribute to its development. Python has an online forum where thousands of coders gather daily to improve this language further. Along with this Python is free to download and use in any operating system, be it Windows, Mac or Linux.

Support for GUI

GUI or Graphical User Interface is one of the key aspects of any programming language because it has the ability to add flair to code and make the results more visual. Python has support for a wide array of GUIs which can easily be imported to the interpreter, thus making this one of the most favorite languages for developers.

Object-Oriented Approach

One of the key aspects of Python is its object-oriented approach. This basically means that Python recognizes the concept of class and object encapsulation thus allowing programs to be efficient in the long run.

High-Level Language

Python has been designed to be a high-level programming language, which means that when you code in Python you don't need to be aware of the coding structure, architecture as well as memory management.

Integrated by Nature

Python is an integrated language by nature. This means that the python interpreter executes codes one line at a time. Unlike other object-oriented programming languages, we don't need to compile Python code thus making the debugging process much easier and efficient. Another advantage of this is, that upon execution the Python code is immediately converted into an intermediate form also known as byte-code which makes it easier to execute and also saves runtime in the long run.

Highly Portable

Suppose you are running Python on Windows and you need to shift the same to either a Mac or a Linux system, then you can easily achieve the same in Python without having to worry about changing the code. This is not possible in

other programming languages, thus making Python one of the most portable languages available in the industry.

Highly Dynamic

As mentioned in an earlier paragraph, Python is one of the most dynamic languages available in the industry today. What this basically means is that the type of a variable is decided at the run time and not in advance. Due to the presence of this feature, we do not need to specify the type of the variable during coding, thus saving time and increasing efficiency.

Extensive Array of Library

Out of the box, Python comes inbuilt with a large number of libraries that can be imported at any instance and be used in a specific program. The presence of libraries also makes sure that you don't need to write all the code yourself and can import the same from those that already exist in the libraries.

Support for Other Languages

Being coded in C, Python by default supports the execution of code written in other programming languages such as Java, C, and C#, thus making it one of the versatile in the industry.

.

File Handling in Python

Python too supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but alike other concepts of Python, this concept here is also easy and short. Python treats file differently as text or binary and this is important. Each line of code includes a sequence of characters and they form text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. Let's start with Reading and Writing files.

Working of open() function

We use **open ()** function in Python to open a file in read or write mode. As explained above, open ( ) will return a file object. To return a file object we use **open()** function along with two arguments, that accepts file name and the

mode, whether to read or write. So, the syntax being: **open(filename, mode)**. There are three kinds of mode, that Python provides and how files can be opened:

- " **r** ", for reading.
- " **w** ", for writing.
- " **a** ", for appending.
- " **r+** ", for both reading and writing

One must keep in mind that the mode argument is not mandatory. If not passed, then Python will assume it to be " **r** " by default. Let's look at this program and try to analyze how the read mode works:

filter_none
edit
play_arrow
brightness_4

```
# a file named "geek", will be opened with the reading mode.
file = open('geek.txt', 'r')
# This will print every line one by one in the file
for each in file:
    print (each)
```

The open command will open the file in the read mode and the for loop will print each line present in the file.

MYSQL

MySQL is a relational database management system (RDBMS) based on the SQL (Structured Query Language) queries. It is one of the most popular languages for accessing and managing the records in the table. MySQL is open-source and free software under the GNU license. Oracle Company supports it.
The following are the most important features of MySQL:

Relational Database Management System (RDBMS)
MySQL is a relational database management system. This database language is based on the SQL queries to access and manage the records of the table.

Easy to use
MySQL is easy to use. We have to get only the basic knowledge of SQL. We can build and interact with MySQL by using only a few simple SQL statements.

It is secure
MySQL consists of a solid data security layer that protects sensitive data from intruders. Also, passwords are encrypted in MySQL.

Client/ Server Architecture
MySQL follows the working of a client/server architecture. There is a database server (MySQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they can query data, save changes, etc.

Free to download
MySQL is free to use so that we can download it from MySQL official website without any cost.

It is scalable
MySQL supports multi-threading that makes it easily scalable. It can handle almost any amount of data, up to as much as 50 million rows or more. The default file size limit is about 4 GB. However, we can increase this number to a theoretical limit of 8 TB of data.

Speed
MySQL is considered one of the very fast database languages, backed by a large number of the benchmark test.
High Flexibility
MySQL supports a large number of embedded applications, which makes MySQL very flexible.
Compatible on many operating systems
MySQL is compatible to run on many operating systems, like Novell NetWare, Windows* Linux*, many varieties of UNIX* (such as Sun* Solaris*, AIX, and DEC* UNIX), OS/2, FreeBSD*, and others. MySQL also provides a facility that the clients can run on the same computer as the server or on another computer (communication via a local network or the Internet).
Allows roll-back
MySQL allows transactions to be rolled back, commit, and crash recovery.
Memory efficiency
Its efficiency is high because it has a very low memory leakage problem.
High Performance
MySQL is faster, more reliable, and cheaper because of its unique storage engine architecture. It provides very high-performance results in comparison to other databases without losing an essential functionality of the software. It has fast loading utilities because of the different cache memory.
High Productivity
MySQL uses Triggers, Stored procedures, and views that allow the developer to give higher productivity.
Platform Independent
It can download, install, and execute on most of the available operating systems.
Partitioning
This feature improves the performance and provides fast management of the large database.
GUI Support
MySQL provides a unified visual database graphical user interface tool named "**MySQL Workbench**" to work with database architects, developers, and Database Administrators. MySQL Workbench provides SQL development, data modeling, data migration, and comprehensive administration tools for server configuration, user administration, backup, and many more. MySQL has a fully GUI supports from MySQL Server version 5.6 and higher.
Dual Password Support
MySQL version 8.0 provides support for dual passwords: one is the current password, and another is a secondary password, which allows us to transition to the new password.

Disadvantages/Drawback of MySQL

Following are the few disadvantages of MySQL:

- MySQL version less than 5.0 doesn't support ROLE, COMMIT, and stored procedure.
- MySQL does not support a very large database size as efficiently.
- MySQL doesn't handle transactions very efficiently, and it is prone to data corruption.

- MySQL is accused that it doesn't have a good developing and debugging tool compared to paid databases.
- MySQL doesn't support SQL check constraints.

MODULES USED AND THEIR PURPOSE

Module

math	*Mathematical functions (sin() etc.).*
random	*Generate pseudo-random numbers with various common distributions.*
operator	*Functions corresponding to the standard operators.*
Os	*Miscellaneous operating system interfaces.*

enum *Implementation of an enumeration class.*

CODING

#**mainbank.py**

```
import  mysql.connector as sql
conn=sql.connect(host='localhost',user='root',passwd='',database=
'bank',charset='utf8')
cur = conn.cursor()
#cur.execute('create table user_table(username varchar(25)
primary key,passwrd varchar(25) not null )')
```

```
print('==========================WELCOME TO
STARK
BANK=========================================
==================')
import datetime as dt
print(dt.datetime.now())
print('1.REGISTER')
print()
print('2.LOGIN')
print()



n=int(input('enter your choice='))
print()

if n== 1:
    name=input('Enter a Username=')
    print()
    passwd=int(input('Enter a 4 DIGIT Password='))
    print()
    V_SQLInsert="INSERT  INTO user_table
(passwrd,username) values (" +  str (passwd) + ",' " + name + " ')
"
    cur.execute(V_SQLInsert)
    conn.commit()
    print()
    print('USER created succesfully')
    import menu

if  n==2 :
    name=input('Enter your Username=')
    print()
    passwd=int(input('Enter your 4 DIGIT Password='))
```

```
    V_Sql_Sel="select * from user_table where passwrd='"+str
(passwd)+"' and username=  ' " +name+ " ' "
    cur.execute(V_Sql_Sel)
    if cur.fetchone() is  None:
        print()
        print('Invalid username or password')
    else:
        print()
        import menu
```

#menu.py

```
import datetime as dt
import  mysql.connector as sql
conn=sql.connect(host='localhost',user='root',passwd='',database=
'bank',charset='utf8')
cur = conn.cursor()



conn.autocommit = True
c = 'y'
while c == 'y':

                    print()
                    print('1.CREATE BANK ACCOUNT')
                    print()
                    print('2.TRANSACTION')
                    print()
                    print('3.CUSTOMER DETAILS')
                    print()
                    print('4.TRANSACTION DETAILS')
                    print()
                    print('5.DELETE ACCOUNT')
                    print()
```

```
                print('6.QUIT')

                print()
                n=int(input('Enter your CHOICE='))
                print()

                if n == 1:

                        acc_no=int(input('Enter your ACCOUNT
NUMBER='))
                        print()
                        acc_name=input('Enter your ACCOUNT
NAME=')
                        print()
                        ph_no=int(input('Enter your PHONE
NUMBER='))
                        print()
                        add=(input('Enter your place='))
                        print()
                        cr_amt=int(input('Enter your credit
amount='))
                        V_SQLInsert="INSERT  INTO
customer_details values (" +  str (acc_no) + ",' " + acc_name + "
','"+str(ph_no) + ",' " +add + " ','"+ str (cr_amt) + " ) "
                        cur.execute(V_SQLInsert)
                        print()
                        print('Account Created Succesfully!!!!!')
                        conn.commit()


                if n == 2:
                    acct_no=int(input('Enter Your Account
Number='))
                    cur.execute('select * from customer_details
where acct_no='+str (acct_no) )
```

```
                    data=cur.fetchall()
                    count=cur.rowcount
                    conn.commit()
                    if count == 0:
                        print()
                        print('Account Number Invalid Sorry Try
Again Later')
                        print()
                    else:
                        print()
                        print('1.WITHDRAW AMOUNT')
                        print()
                        print('2.ADD AMOUNT')
                        print()

                        print()
                        x=int(input('Enter your CHOICE='))
                        print()
                        if x == 1:
                            amt=int(input('Enter withdrawl
amount='))
                            cr_amt=0
                            cur.execute('update customer_details set
cr_amt=cr_amt-'+str(amt) +  ' where acct_no=' +str(acct_no) )
                            V_SQLInsert="INSERT  INTO
transactions values ({} , '{}' , {} , {})
".format(acct_no,dt.datetime.today(),amt,cr_amt)
                            cur.execute(  V_SQLInsert)
                            conn.commit()
                            print()
                            print('Account Updated
Succesfully!!!!!')
```

```
                        if x== 2:
                            amt=int(input('Enter amount to be
added='))
                            cr_amt=0
                            cur.execute('update customer_details set
cr_amt=cr_amt+'+str(amt) +  ' where acct_no=' +str(acct_no) )
                            V_SQLInsert="INSERT  INTO
transactions values ({} , '{}' , {} , {})
".format(acct_no,dt.datetime.today(),cr_amt,amt)
                            cur.execute(  V_SQLInsert)
                            conn.commit()
                            print()
                            print('Account Updated
Succesfully!!!!!')

                    if n == 3:
                        acct_no=int(input('Enter your account
number='))
                        print()
                        cur.execute('select * from customer_details
where acct_no='+str(acct_no) )
                        if cur.fetchone() is  None:
                            print()
                            print('Invalid Account number')
                        else:
                            cur.execute('select * from customer_details
where acct_no='+str(acct_no) )
                            data=cur.fetchall()
                            for row in data:
                                print('ACCOUNT NO=',acct_no)
                                print()
                                print('ACCOUNT NAME=',row[1])
                                print()
                                print(' PHONE NUMBER=',row[2])
                                print()
```

```
                    print('ADDRESS=',row[3])
                    print()
                    print('cr_amt=',row[4])
            if n == 4:
                acct_no=int(input('Enter your account
number='))
                print()
                cur.execute('select * from customer_details
where acct_no='+str(acct_no) )
                if cur.fetchone() is  None:
                   print()
                   print('Invalid Account number')
                else:
                   cur.execute('select * from transactions
where acct_no='+str(acct_no) )
                   data=cur.fetchall()
                   for row in data:
                       print('ACCOUNT NO=',acct_no)
                       print()
                       print('DATE=',row[1])
                       print()
                       print(' WITHDRAWAL
AMOUNT=',row[2])
                       print()
                       print('AMOUNT ADDED=',row[3])
                       print()


            if n == 5:
                print('DELETE YOUR ACCOUNT')
                acct_no=int(input('Enter your account
number='))

                cur.execute('delete from customer_details
where acct_no='+str(acct_no) )
```

```
                    print('ACCOUNT DELETED
SUCCESFULLY')

                if n == 6:
                    print('DO YO WANT TO EXIT(y/n)')
                    c=input ('enter your choice=')




else:
    print('THANK YOU PLEASE VISIT AGAIN')
    quit()
```

#customer details

```
import  mysql.connector as sql
conn=sql.connect(host='localhost',user='root',passwd='',database=
'bank',charset='utf8')
#if conn.is_connected():
    #print('connected succesfully')
cur = conn.cursor()
cur.execute('create table customer_details(acct_no int primary
key,acct_name varchar(25) ,phone_no bigint(25)
check(phone_no>11),address varchar(25),cr_amt float )')
```

#table.py

```
import  mysql.connector as sql
conn=sql.connect(host='localhost',user='root',passwd='',database=
'bank',charset='utf8')
cur = conn.cursor()
cur.execute('create table user_table(username varchar(25)
primary key,passwrd varchar(25) not null )')
```

#transacton_table.py

```
import  mysql.connector as sql
conn=sql.connect(host='localhost',user='root',passwd='',database=
'bank',charset='utf8')
cur = conn.curs;
;’ or()
cur.execute('create table transactions(acct_no int(11),date date
,withdrawal_amt bigint(20),amount_added bigint(20) )')
```

OUTPUT

(Note- I am giving just few output you have to give all necessary output of your coding.)

Output-1

==========

===WELCOME TO STARK BANK==========

==========

==========

2020-09-12 11:07:29.031122

1.REGISTER

2.LOGIN

enter your choice=

Output-2

enter your choice=1

Enter a Username=samir
Enter a 4 DIGIT
Password=1234

Output-3

enter your choice=2

Enter your Username=samir

Enter your 4 DIGIT Password=1234

1.CREATE BANK ACCOUNT

2.TRANSACTION

3.CUSTOMER DETAILS

4.TRANSACTION DETAILS

5.DELETE ACCOUNT

6.QUIT

Enter your CHOICE=

REQUIREMENTS

❖ HARDWARE REQUIRED

- Printer, to print the required documents of the project
- Compact Drive
- Processor : Pentium III
- Ram : 128 MB
- Harddisk : 20 Gb.

❖ SOFTWARE REQUIRED

- Operating system : Windows XP,Window-10
- Spyder,pycharm for execution of program and
- Ms word for documentation.

BIBLIOGRAPHY

In order to work on this project titled-Bank Management system ,the following books and literature are referred by me during the various phases of development of the project.

1. **COMPUTER SCIENCE With Python BY :– SUMITA ARORA**

2. **Object Oriented Programming In python By- ********

3.Various websites(You can mention name of websites)

Other than the above mentioned books, the suggestions and supervision of my teacher and my class experience also helped me to develop this software project.