INDEX

1. CERTIFICATE
2. ACKNOWLEDGMENT
3. INTRODUCTION
4. MODULES USED
5. SOURCE CODE
6. PROJECT OUTPUT
7. USER MANUAL
8. BIBILOGRAPHY

CERTIFICATE

This is to certify that Aniket Das of Class- XII Section M of Amity International School, Mayur Vihar has successfully completed his project in computer science for the AISSCE as prescribed by CBSE in the academic year 2021-22.

Sign and Name of Teacher In-charge

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my teacher Ms.Deepshika Sethi who gave me the opportunity to do this project and also helped me in doing a lot of Research. I came to know about a lot of new things related to python. Secondly, I would also like to thank my parents who helped me a lot in finalizing this project with in the limited timeframe.

Name of Student

Aniket Das

Name of Teacher

Ms.DeepshikhaSethi

INTRODUCTION

Our application is designed to function in rural India and provide appropriate financial advice to farmers and other village dwellers on the basis of their assets.

This application uses Kmean algorithm to sort users into various groups and then provides advice according to these groups. These groups are made on the basis of user’s income and balance which is provided by the user.

Apart from this the application is also capable of calculating a Custom Credit Score of the user by taking into account their land holdings, livestock, balance etc. The credit score is calculated out of 100.

He employee can also edit the user database using the application. The application allows the employee to create new user record, update existing user data and remove user from the database.

Language used to code application- Python

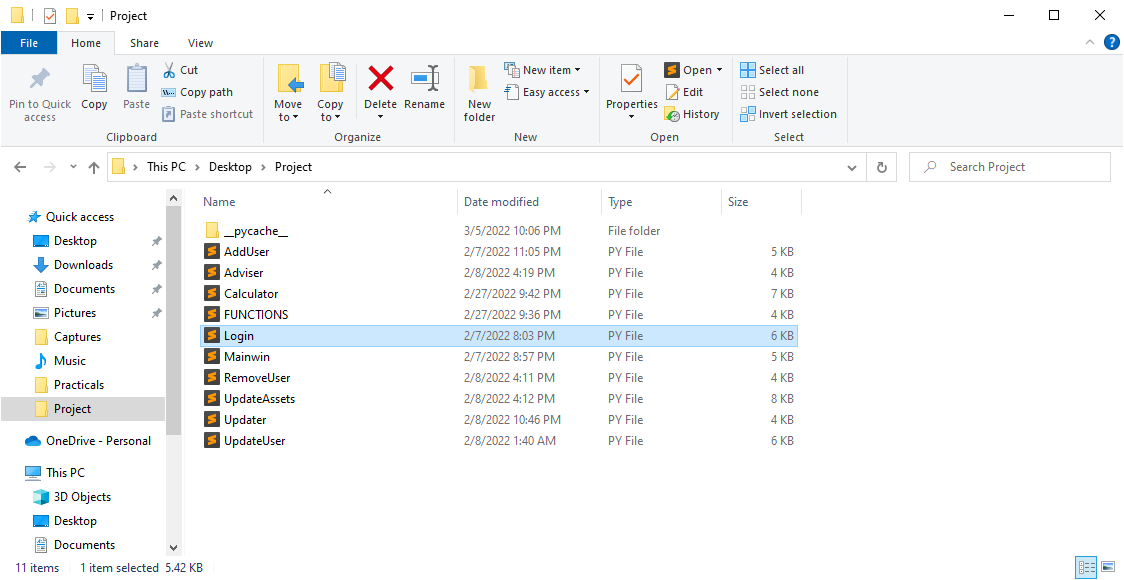
Interface Language: English

MODULES USED

* PyQt5
* pandas
* numpy
* matplotlib
* math
* random
* sqlalchemy
* sklearn
* sys

SOURCE CODE

# PROJECT DIRECTORY



# ORDER OF FILES-

1. AddUser.py
2. Adviser.py
3. Calculator.py
4. FUNCTIONS.py
5. Login.py
6. Mainwin.py
7. RemoveUser.py
8. UpdateAssets.py
9. Updater.py
10. UpdateUser.py

# AddUser.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

import random

class Ui\_NEWUSER(object):

def setupUi(self, NEWUSER):

NEWUSER.setObjectName("NEWUSER")

NEWUSER.resize(395, 347)

NEWUSER.setStyleSheet("background-color: rgb(170, 170, 255);")

self.centralwidget = QtWidgets.QWidget(NEWUSER)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(130, 30, 121, 41))

self.Heading.setStyleSheet("font: 75 italic 16pt \"Verdana\";\n"

"color: rgb(255, 255, 127);")

self.Heading.setObjectName("Heading")

self.field = QtWidgets.QLabel(self.centralwidget)

self.field.setGeometry(QtCore.QRect(80, 100, 91, 31))

self.field.setStyleSheet("color: rgb(255, 255, 127);\n"

"font: 10pt \"MS Shell Dlg 2\";")

self.field.setObjectName("field")

self.fiels = QtWidgets.QLabel(self.centralwidget)

self.fiels.setGeometry(QtCore.QRect(80, 150, 91, 31))

self.fiels.setStyleSheet("color: rgb(255, 255, 127);\n"

"font: 10pt \"MS Shell Dlg 2\";")

self.fiels.setObjectName("fiels")

self.field\_2 = QtWidgets.QLabel(self.centralwidget)

self.field\_2.setGeometry(QtCore.QRect(80, 200, 91, 31))

self.field\_2.setStyleSheet("font: 10pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 250, 99);")

self.field\_2.setObjectName("field\_2")

self.AddUSER = QtWidgets.QPushButton(self.centralwidget, clicked= lambda: self.Add())

self.AddUSER.setGeometry(QtCore.QRect(130, 260, 121, 51))

self.AddUSER.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"color: rgb(0, 0, 127);\n"

"font: 75 12pt \"Times New Roman\";")

self.AddUSER.setObjectName("AddUSER")

self.UserN = QtWidgets.QLineEdit(self.centralwidget)

self.UserN.setGeometry(QtCore.QRect(210, 99, 151, 31))

self.UserN.setStyleSheet("background-color: rgb(255, 255, 255);")

self.UserN.setObjectName("UserN")

self.Balance = QtWidgets.QLineEdit(self.centralwidget)

self.Balance.setGeometry(QtCore.QRect(210, 150, 151, 31))

self.Balance.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Balance.setObjectName("Balance")

self.Income = QtWidgets.QLineEdit(self.centralwidget)

self.Income.setGeometry(QtCore.QRect(210, 200, 151, 31))

self.Income.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Income.setObjectName("Income")

NEWUSER.setCentralWidget(self.centralwidget)

self.retranslateUi(NEWUSER)

QtCore.QMetaObject.connectSlotsByName(NEWUSER)

def Add(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Recs=pd.read\_sql\_table("records",engine)

Assets=pd.read\_sql\_table("assets",engine)

UserID=Recs["User\_ID"].to\_numpy()

Account\_No=Recs["Account\_Number"].to\_numpy()

while True:

UID=random.randint(1000,1999)

Acc\_No=random.randint(13469000,13469999)

if UID not in UserID and Acc\_No not in Account\_No:

break

Name=str(self.UserN.text())

Bal=int(self.Balance.text())

Inc=int(self.Income.text())

record=[{"User\_ID": UID,"Account\_Number":Acc\_No,"Name": Name,"Income": Inc,"Balance": Bal}]

Recs=Recs.append(record,ignore\_index=True)

asset=[{"UserID": UID,"Land\_Area":0,"Cows": 0,"Goats": 0,"Chicken":0}]

Assets=Assets.append(asset,ignore\_index=True)

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

Recs.to\_sql(name="records",con=engine,if\_exists="replace",index=False)

def retranslateUi(self, NEWUSER):

\_translate = QtCore.QCoreApplication.translate

NEWUSER.setWindowTitle(\_translate("NEWUSER", "Add New User"))

self.Heading.setText(\_translate("NEWUSER", "NEW USER"))

self.field.setText(\_translate("NEWUSER", "USER NAME"))

self.fiels.setText(\_translate("NEWUSER", "BALANCE"))

self.field\_2.setText(\_translate("NEWUSER", "INCOME"))

self.AddUSER.setText(\_translate("NEWUSER", "ADD USER"))

self.UserN.setPlaceholderText(\_translate("NEWUSER", "Enter User Name"))

self.Balance.setPlaceholderText(\_translate("NEWUSER", "Enter Balance of User"))

self.Income.setPlaceholderText(\_translate("NEWUSER", "Enter Income of the User"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

NEWUSER = QtWidgets.QMainWindow()

ui = Ui\_NEWUSER()

ui.setupUi(NEWUSER)

NEWUSER.show()

sys.exit(app.exec\_())

# ADVISER.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

from FUNCTIONS import \*

class Ui\_Advice(object):

def setupUi(self, Advice):

Advice.setObjectName("Advice")

Advice.resize(479, 284)

Advice.setStyleSheet("background-color: rgb(218, 192, 44);")

self.centralwidget = QtWidgets.QWidget(Advice)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(130, 30, 211, 31))

self.Heading.setStyleSheet("font: 63 italic 16pt \"Lucida Bright\";\n"

"color: rgb(0, 85, 255);")

self.Heading.setObjectName("Heading")

self.FIELD = QtWidgets.QLabel(self.centralwidget)

self.FIELD.setGeometry(QtCore.QRect(120, 90, 81, 31))

self.FIELD.setStyleSheet("color: rgb(26, 41, 255);\n"

"font: 75 11pt \"MS Shell Dlg 2\";")

self.FIELD.setObjectName("FIELD")

self.Userid = QtWidgets.QLineEdit(self.centralwidget)

self.Userid.setGeometry(QtCore.QRect(220, 89, 171, 31))

self.Userid.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Userid.setObjectName("Userid")

self.pushButton = QtWidgets.QPushButton(self.centralwidget, clicked= lambda: self.advanced\_adv())

self.pushButton.setGeometry(QtCore.QRect(160, 150, 141, 41))

self.pushButton.setStyleSheet("background-color: rgb(0, 85, 255);\n"

"color: rgb(240, 244, 23);\n"

"font: 75 10pt \"Verdana\";")

self.pushButton.setObjectName("pushButton")

self.Advise = QtWidgets.QTextBrowser(self.centralwidget)

self.Advise.setGeometry(QtCore.QRect(25, 211, 431, 61))

self.Advise.setObjectName("Advise")

Advice.setCentralWidget(self.centralwidget)

self.retranslateUi(Advice)

QtCore.QMetaObject.connectSlotsByName(Advice)

def advanced\_adv(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Recs= pd.read\_sql\_table("records",engine)

UserID=Recs["User\_ID"].to\_numpy()

self.Advise.setText("")

UID=int(self.Userid.text())

if UID in UserID:

Adv=FINAL(UID)

self.Advise.setText(Adv)

else:

self.Advise.setText("Invalid User ID")

def retranslateUi(self, Advice):

\_translate = QtCore.QCoreApplication.translate

Advice.setWindowTitle(\_translate("Advice", "Advicer"))

self.Heading.setText(\_translate("Advice", "FINANCIAL ADVICE"))

self.FIELD.setText(\_translate("Advice", "USER ID"))

self.pushButton.setText(\_translate("Advice", "SUGGEST ADVICE"))

self.Advise.setHtml(\_translate("Advice", "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"

"<html><head><meta name=\"qrichtext\" content=\"1\" /><style type=\"text/css\">\n"

"p, li { white-space: pre-wrap; }\n"

"</style></head><body style=\" font-family:\'MS Shell Dlg 2\'; font-size:8.25pt; font-weight:400; font-style:normal;\">\n"

"<p style=\"-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><br /></p></body></html>"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

Advice = QtWidgets.QMainWindow()

ui = Ui\_Advice()

ui.setupUi(Advice)

Advice.show()

sys.exit(app.exec\_())

# CALCULATOR.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import sys

import pandas as pd

import numpy as np

import sqlalchemy

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(485, 361)

MainWindow.setStyleSheet("background-color: rgb(194, 255, 193);")

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setEnabled(True)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(80, 10, 321, 61))

self.Heading.setStyleSheet("font: 75 16pt \"Palatino Linotype\";\n"

"color: rgb(132, 157, 171);")

self.Heading.setObjectName("Heading")

self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit.setGeometry(QtCore.QRect(250, 110, 181, 31))

font = QtGui.QFont()

font.setPointSize(10)

self.lineEdit.setFont(font)

self.lineEdit.setAutoFillBackground(False)

self.lineEdit.setStyleSheet("background-color: rgb(255, 255, 255);")

self.lineEdit.setInputMethodHints(QtCore.Qt.ImhDigitsOnly|QtCore.Qt.ImhPreferNumbers)

self.lineEdit.setText("")

self.lineEdit.setObjectName("lineEdit")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(60, 110, 91, 31))

font = QtGui.QFont()

font.setFamily("MS Shell Dlg 2")

font.setPointSize(12)

font.setBold(False)

font.setItalic(False)

font.setWeight(9)

self.label.setFont(font)

self.label.setStyleSheet("color: rgb(255, 123, 123);\n"

"font: 75 12pt \"MS Shell Dlg 2\";\n"

"")

self.label.setObjectName("label")

self.Invalid = QtWidgets.QLabel(self.centralwidget)

self.Invalid.setGeometry(QtCore.QRect(250, 150, 181, 31))

font = QtGui.QFont()

font.setFamily("MS Shell Dlg 2")

font.setPointSize(10)

font.setBold(False)

font.setItalic(False)

font.setWeight(9)

self.Invalid.setFont(font)

self.Invalid.setStyleSheet("color: rgb(255, 123, 123);\n"

"font: 75 10pt \"MS Shell Dlg 2\";\n"

"")

self.Invalid.setText("")

self.Invalid.setObjectName("Invalid")

self.pushButton = QtWidgets.QPushButton(self.centralwidget)

self.pushButton.setGeometry(QtCore.QRect(90, 190, 141, 51))

self.pushButton.setStyleSheet("background-color: rgb(255, 255, 127);\n"

"color: rgb(76, 130, 181);")

self.pushButton.setObjectName("pushButton")

self.pushButton.clicked.connect(lambda: self.calc())

self.Output = QtWidgets.QLabel(self.centralwidget)

self.Output.setGeometry(QtCore.QRect(60, 260, 331, 41))

self.Output.setStyleSheet("color: rgb(200, 133, 0);\n"

"font: 75 14pt \"Times New Roman\";")

self.Output.setText("")

self.Output.setObjectName("Output")

self.EXIT = QtWidgets.QPushButton(self.centralwidget)

self.EXIT.setGeometry(QtCore.QRect(260, 190, 141, 51))

self.EXIT.setStyleSheet("background-color: rgb(255, 255, 127);\n"

"color: rgb(76, 130, 181);")

self.EXIT.setObjectName("EXIT")

self.EXIT.clicked.connect(MainWindow.close)

MainWindow.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setEnabled(False)

self.menubar.setGeometry(QtCore.QRect(0, 0, 485, 21))

self.menubar.setObjectName("menubar")

MainWindow.setMenuBar(self.menubar)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setEnabled(False)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "Credit Score Calculator"))

self.Heading.setText(\_translate("MainWindow", "CREDIT SCORE CALCULATOR"))

self.lineEdit.setPlaceholderText(\_translate("MainWindow", "Enter User ID"))

self.label.setText(\_translate("MainWindow", "User ID"))

self.pushButton.setText(\_translate("MainWindow", "Calculate"))

self.EXIT.setText(\_translate("MainWindow", "RETURN"))

def calc(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Recs=pd.read\_sql\_table("records",engine)

Assets=pd.read\_sql\_table("assets",engine)

assets=Assets[["UserID","Land\_Area","Cows","Goats","Chicken"]].to\_numpy()

Records=Recs[["User\_ID","Balance","Income"]].to\_numpy()

self.Invalid.setText("")

self.Output.setText("")

UID=int(self.lineEdit.text())

if UID not in Records:

self.Invalid.setText("Invalid User ID")

return

for i in Records:

if UID==i[0]:

Bal,Inc=i[1],i[2]

break

Area,Cows,Goats,Chickens=0,0,0,0

for j in assets:

if UID==j[0]:

Area,Cows,Goats,Chickens=j[1],j[2],j[3],j[4]

break

Score=0

if Inc<=100000:

Score+=10

elif Inc>=100000:

Score+=20

if Bal<=250000:

Score+=10

elif Bal>250000 and Bal<=370000:

Score+=20

elif Bal>370000:

Score+=30

if Area==0:

Score+=0

elif Area<108000:

Score+=5

elif Area<200000 and Area>=108000:

Score+=10

elif Area>=200000 and Area<250000:

Score+=15

elif Area>=250000:

Score+=20

if Cows==0:

Score+=0

elif Cows<5:

Score+=3

elif Cows>=5 and Cows<10:

Score+=5

elif Cows>=10 and Cows<20:

Score+=8

elif Cows>=20:

Score+=10

if Goats==0:

Score+=0

elif Goats<10:

Score+=3

elif Goats>=10 and Goats<50:

Score+=5

elif Goats>=50:

Score+=10

if Chickens==0:

Score+=0

elif Chickens<20:

Score+=3

elif Chickens>=20 and Chickens<60:

Score+=5

elif Chickens>=60:

Score+=10

self.Output.setText(f"Calculated Credit Score is {Score}/100")

if \_\_name\_\_ == "\_\_main\_\_":

app = QtWidgets.QApplication(sys.argv)

MainWindow = QtWidgets.QMainWindow()

ui = Ui\_MainWindow()

ui.setupUi(MainWindow)

MainWindow.show()

sys.exit(app.exec\_())

# LOGIN.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

from Mainwin import \*

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(469, 290)

MainWindow.setStyleSheet("background-color: rgb(202, 202, 202)")

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.login = QtWidgets.QPushButton(self.centralwidget)

self.login.setGeometry(QtCore.QRect(110, 200, 191, 51))

font = QtGui.QFont()

font.setFamily("MS Sans Serif")

font.setBold(True)

font.setItalic(False)

font.setWeight(75)

self.login.setFont(font)

self.login.setStyleSheet("background-color: rgb(0, 170, 0);\n"

"color: rgb(255, 255, 255);\n"

"border-color: rgb(255, 255, 255);")

self.login.setObjectName("login")

D=self.login.clicked.connect(lambda: self.check())

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(40, 70, 151, 31))

font = QtGui.QFont()

font.setFamily("MS Sans Serif")

font.setPointSize(12)

font.setBold(True)

font.setWeight(75)

self.label.setFont(font)

self.label.setStyleSheet("color:rgb(17, 41, 255)")

self.label.setObjectName("label")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(40, 130, 141, 31))

font = QtGui.QFont()

font.setFamily("MS Sans Serif")

font.setPointSize(12)

font.setBold(True)

font.setWeight(75)

self.label\_2.setFont(font)

self.label\_2.setStyleSheet("color:rgb(17, 41, 255)")

self.label\_2.setObjectName("label\_2")

self.label\_3 = QtWidgets.QLabel(self.centralwidget)

self.label\_3.setGeometry(QtCore.QRect(100, 10, 261, 31))

font = QtGui.QFont()

font.setFamily("Rockwell")

font.setPointSize(12)

font.setBold(True)

font.setItalic(True)

font.setUnderline(False)

font.setWeight(75)

self.label\_3.setFont(font)

self.label\_3.setStyleSheet("color: rgb(0, 170, 0)")

self.label\_3.setObjectName("label\_3")

self.emp\_ID = QtWidgets.QLineEdit(self.centralwidget)

self.emp\_ID.setGeometry(QtCore.QRect(210, 70, 231, 31))

self.emp\_ID.setStyleSheet("background-color: rgb(255, 255, 255);")

self.emp\_ID.setObjectName("emp\_ID")

self.Pass = QtWidgets.QLineEdit(self.centralwidget)

self.Pass.setGeometry(QtCore.QRect(210, 130, 231, 31))

self.Pass.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Pass.setEchoMode(QtWidgets.QLineEdit.Password)

self.Pass.setObjectName("Pass")

self.label\_4 = QtWidgets.QLabel(self.centralwidget)

self.label\_4.setGeometry(QtCore.QRect(210, 170, 231, 21))

font = QtGui.QFont()

font.setFamily("MS Sans Serif")

font.setPointSize(8)

font.setBold(True)

font.setWeight(75)

self.label\_4.setFont(font)

self.label\_4.setStyleSheet("color:rgb(17, 41, 255)")

self.label\_4.setText("")

self.label\_4.setObjectName("label\_4")

MainWindow.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setGeometry(QtCore.QRect(0, 0, 469, 21))

self.menubar.setObjectName("menubar")

MainWindow.setMenuBar(self.menubar)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def next(self):

self.Window = QtWidgets.QMainWindow()

self.ui = Ui\_Adviser()

self.ui.setupUi(self.Window)

self.Window.show()

MainWindow.close()

def check(self):

self.empID=int(self.emp\_ID.text())

self.Pas=int(self.Pass.text())

self.employee=pd.read\_sql\_table("employee",engine)

self.emp=self.employee[["EmpID","PIN"]].to\_numpy()

for k in self.emp:

if k[0]==self.empID and k[1]==self.Pas:

self.next()

return True

else:

self.label\_4.setText("Invalid Password or Emplyee ID")

break

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))

self.login.setText(\_translate("MainWindow", "LOGIN"))

self.label.setText(\_translate("MainWindow", "Employee ID "))

self.label\_2.setText(\_translate("MainWindow", "Password"))

self.label\_3.setText(\_translate("MainWindow", "FINANCIAL ADVICER SOFTWARE"))

self.emp\_ID.setPlaceholderText(\_translate("MainWindow", "Enter User ID"))

self.Pass.setPlaceholderText(\_translate("MainWindow", "Enter Password"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

MainWindow = QtWidgets.QMainWindow()

ui = Ui\_MainWindow()

ui.setupUi(MainWindow)

MainWindow.show()

sys.exit(app.exec\_()

# FUNCTIONS.PY

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import math as m

import random

import names

import sqlalchemy

from datetime import date as dt

from sklearn.cluster import KMeans

#Functions for Kmeans Data Analysis

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Recs= pd.read\_sql\_table("records",engine)

Data=Recs[["Balance","Income","User\_ID"]].to\_numpy()

points=Recs[["Balance","Income"]].to\_numpy()

User\_Data=Recs[["User\_ID","Account\_Number","Name","Balance","Income"]].to\_numpy()

UserID=Recs["User\_ID"].to\_numpy()

Account\_No=Recs["Account\_Number"].to\_numpy()

kmeans=KMeans(n\_clusters=6,init="k-means++")

kmeans.fit(points) #Fitting data in kmeans object

centers = kmeans.cluster\_centers\_ #Get cluster centers

clusters=kmeans.fit\_predict(points) #Assigning Numbers to points according to their clusters [index in both lists is same]

def kmean\_graph(Data,clusters,user=None): #To show clustered graph

plt.scatter(Data[clusters==0,0],Data[clusters==0,1],c="red")

plt.scatter(Data[clusters==1,0],Data[clusters==1,1],c="green")

plt.scatter(Data[clusters==2,0],Data[clusters==2,1],c="blue")

plt.scatter(Data[clusters==3,0],Data[clusters==3,1],c="yellow")

plt.scatter(Data[clusters==4,0],Data[clusters==4,1],c="purple")

plt.scatter(Data[clusters==5,0],Data[clusters==5,1],c="brown")

if user is not None:

plt.scatter(user[0],user[1],c="black",marker="D")

plt.xlabel("Balance")

plt.ylabel("Income")

plt.show()

def graph():

UID=int(input("Enter UserID [1000-1500] "))

if 1000<UID<1500:

rec=get\_rec(UID)

kmean\_graph(Data,clusters,rec)

elif UID==0:

kmean\_graph(Data,clusters)

else:

print("Invalid UserID")

def get\_rec(UID): #Read name obvio

rec=[]

for i in Data:

if UID==i[2]:

rec.append(i[0])

rec.append(i[1])

return rec

def Find\_centre(rec,centers): #find nearest centre

mini=100000000000

x,y=int(rec[0]),int(rec[1])

req\_center=[]

for c in centers:

x1,y1=int(c[0]),int(c[1])

distance=m.sqrt((x1-x)\*\*2+(y1-y)\*\*2)

if distance<mini:

mini=distance

req\_center=[x1,y1]

return req\_center

def Output\_process(Center): #advice on the basis of nearest centre

l1=["FD of rupees 50000 with 6% interest per annum.","Create a PPF at 8.7% interest rate for 15 yrs "]

l2=["FD of rupees 75000 with 6.4% interest per annum.","Create a PPF at 9% interest rate for 10yrs"]

l3=["FD of rupees 80000 with 6.3% interest per annum","Take Gold Loans at low interest rates to invest in Gold easily"]

l4=["FD of rupees 90000 with 6.3% interest per annum.","Take Gold Loans at low interest rates to invest in Gold easily"]

l5=["FD of rupees 95000 with 6.4% interest per annum.","Issue loans at 13% interest rate to establish a local industry"]

l6=["Open demat account and start investing in stocks through our local branch broker","Take loans at 15% interest rate to buy plots and start a real estate business"]

if Center[1]<100000:

if Center[0]<225000:

return(random.choice(l1))

elif 225000<Center[0]<375000:

return(random.choice(l2))

elif 375000<Center[0]:

return(random.choice(l3))

if Center[1]>100000:

if Center[0]<225000:

return(random.choice(l4))

elif 225000<Center[0]<375000:

return(random.choice(l5))

elif 375000<Center[0]:

return(random.choice(l6))

def FINAL(UID): #All in one function

rec=get\_rec(UID)

Center=Find\_centre(rec,centers)

return Output\_process(Center)

def advanced\_adv(UID):

if 1000<UID<1500:

return FINAL(UID)

else:

print("Invalid UserID")

# MAINWIN.PY

from PyQt5 import QtCore, QtGui, QtWidgets

from Calculator import \*

from Adviser import \*

from Updater import \*

class Ui\_Adviser(object):

def setupUi(self, Adviser):

Adviser.setObjectName("Adviser")

Adviser.resize(593, 332)

Adviser.setStyleSheet("background-color: rgb(83, 210, 238);")

self.centralwidget = QtWidgets.QWidget(Adviser)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(190, 20, 221, 51))

self.Heading.setStyleSheet("font: 75 16pt \"Verdana\";\n"

"color: rgb(255, 255, 0);")

self.Heading.setObjectName("Heading")

self.Description = QtWidgets.QTextBrowser(self.centralwidget)

self.Description.setGeometry(QtCore.QRect(50, 80, 491, 91))

self.Description.setStyleSheet("color: rgb(255, 255, 255);\n"

"border-color: rgb(85, 255, 255);\n"

"font: 75 12pt \"Rockwell Condensed\";")

self.Description.setObjectName("Description")

self.updatebutton = QtWidgets.QPushButton(self.centralwidget,clicked= lambda: self.Updaterr())

self.updatebutton.setGeometry(QtCore.QRect(30, 210, 141, 61))

self.updatebutton.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"color: rgb(5, 105, 255);\n"

"font: 75 9pt \"MS Shell Dlg 2\";")

self.updatebutton.setObjectName("updatebutton")

self.advicebutton = QtWidgets.QPushButton(self.centralwidget,clicked= lambda: self.Advice())

self.advicebutton.setGeometry(QtCore.QRect(210, 210, 151, 61))

self.advicebutton.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"color: rgb(5, 105, 255);\n"

"font: 75 9pt \"MS Shell Dlg 2\";")

self.advicebutton.setObjectName("advicebutton")

self.calculatebutton = QtWidgets.QPushButton(self.centralwidget,clicked= lambda: self.Calculator())

self.calculatebutton.setGeometry(QtCore.QRect(390, 210, 171, 61))

self.calculatebutton.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"color: rgb(5, 105, 255);\n"

"font: 75 9pt \"MS Shell Dlg 2\";")

self.calculatebutton.setObjectName("calculatebutton")

self.Thnx = QtWidgets.QLabel(self.centralwidget)

self.Thnx.setGeometry(QtCore.QRect(180, 280, 221, 41))

self.Thnx.setStyleSheet("font: 75 10pt \"MS Shell Dlg 2\";\n"

"color: rgb(223, 91, 68);")

self.Thnx.setObjectName("Thnx")

Adviser.setCentralWidget(self.centralwidget)

self.retranslateUi(Adviser)

QtCore.QMetaObject.connectSlotsByName(Adviser)

def Calculator(self):

self.calculator = QtWidgets.QMainWindow()

self.ui = Ui\_MainWindow()

self.ui.setupUi(self.calculator)

self.calculator.show()

def Advice(self):

self.adv= QtWidgets.QMainWindow()

self.ui = Ui\_Advice()

self.ui.setupUi(self.adv)

self.adv.show()

def Updaterr(self):

self.upd= QtWidgets.QMainWindow()

self.ui = Ui\_Updation()

self.ui.setupUi(self.upd)

self.upd.show()

def retranslateUi(self, Adviser):

\_translate = QtCore.QCoreApplication.translate

Adviser.setWindowTitle(\_translate("Adviser", "Financial Adviser"))

self.Heading.setText(\_translate("Adviser", "FINANCIAL ADVISER "))

self.Description.setHtml(\_translate("Adviser", "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"

"<html><head><meta name=\"qrichtext\" content=\"1\" /><style type=\"text/css\">\n"

"p, li { white-space: pre-wrap; }\n"

"</style></head><body style=\" font-family:\'Rockwell Condensed\'; font-size:12pt; font-weight:72; font-style:normal;\">\n"

"<p style=\" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'MS Shell Dlg 2\'; font-size:9pt; font-weight:400; color:#ffffff;\"> This application is designed to function in rural India and provide appropriate financial advice to farmers and other village dwellers on the basis of their assets.It application is also capable of calculating a Custom Credit Score of the user by taking into account their land holdings, livestock, balance etc. The credit score is calculated out of 100..</span></p>\n"

"<p style=\" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\"><span style=\" font-family:\'MS Shell Dlg 2\'; font-size:9pt; font-weight:400; color:#ffffff;\">Developed by Aniket Das</span></p></body></html>"))

self.updatebutton.setText(\_translate("Adviser", "UPDATE USER DATA"))

self.advicebutton.setText(\_translate("Adviser", "GET FINANCIAL ADVICE"))

self.calculatebutton.setText(\_translate("Adviser", "CALCULATE CREDIT SCORE"))

self.Thnx.setText(\_translate("Adviser", "Thank You for using our Application!!"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

Adviser = QtWidgets.QMainWindow()

ui = Ui\_Adviser()

ui.setupUi(Adviser)

Adviser.show()

sys.exit(app.exec\_())

# REMOVEUSER.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

class Ui\_RemoveUser(object):

def setupUi(self, RemoveUser):

RemoveUser.setObjectName("RemoveUser")

RemoveUser.resize(426, 245)

RemoveUser.setStyleSheet("background-color: rgb(247, 178, 146);")

self.centralwidget = QtWidgets.QWidget(RemoveUser)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(80, 10, 251, 41))

self.Heading.setStyleSheet("font: 75 16pt \"Verdana\";\n"

"color: rgb(0, 0, 127);")

self.Heading.setObjectName("Heading")

self.Field = QtWidgets.QLabel(self.centralwidget)

self.Field.setGeometry(QtCore.QRect(90, 80, 81, 31))

self.Field.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(0, 0, 127);")

self.Field.setObjectName("Field")

self.UserN = QtWidgets.QLineEdit(self.centralwidget)

self.UserN.setGeometry(QtCore.QRect(210, 80, 161, 31))

self.UserN.setStyleSheet("background-color: rgb(255, 255, 255);")

self.UserN.setObjectName("UserN")

self.Removebutton = QtWidgets.QPushButton(self.centralwidget,clicked= lambda: self.Remove\_User())

self.Removebutton.setGeometry(QtCore.QRect(140, 150, 131, 41))

self.Removebutton.setStyleSheet("background-color: rgb(18, 123, 221);\n"

"color: rgb(255, 255, 0);\n"

"font: 75 10pt \"Verdana\";")

self.Removebutton.setObjectName("Removebutton")

self.Invalid = QtWidgets.QLabel(self.centralwidget)

self.Invalid.setGeometry(QtCore.QRect(210, 120, 161, 20))

self.Invalid.setText("")

self.Invalid.setObjectName("Invalid")

self.Output = QtWidgets.QLabel(self.centralwidget)

self.Output.setGeometry(QtCore.QRect(80, 202, 271, 31))

self.Output.setText("")

self.Output.setObjectName("Output")

RemoveUser.setCentralWidget(self.centralwidget)

self.retranslateUi(RemoveUser)

QtCore.QMetaObject.connectSlotsByName(RemoveUser)

def Remove\_User(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Assets=pd.read\_sql\_table("assets",engine)

Recs= pd.read\_sql\_table("records",engine)

UserID=Recs["User\_ID"].to\_numpy()

UID=int(self.UserN.text())

if UID in UserID:

Recs.drop(Recs[Recs["User\_ID"]==UID].index,inplace=True)

Assets.drop(Assets[Assets["UserID"]==UID].index,inplace=True)

Recs.to\_sql(name="records",con=engine,if\_exists="replace",index=False)

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

self.Output.setText("Successfully Removed Record")

return

else:

self.Invalid.setText("Invalid User ID")

return

def retranslateUi(self, RemoveUser):

\_translate = QtCore.QCoreApplication.translate

RemoveUser.setWindowTitle(\_translate("RemoveUser", "Delete User Record"))

self.Heading.setText(\_translate("RemoveUser", "REMOVE USER RECORD"))

self.Field.setText(\_translate("RemoveUser", "USER ID"))

self.Removebutton.setText(\_translate("RemoveUser", "REMOVE RECORD"))

self.UserN.setPlaceholderText(\_translate("RemoveUser","Enter User ID"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

RemoveUser = QtWidgets.QMainWindow()

ui = Ui\_RemoveUser()

ui.setupUi(RemoveUser)

RemoveUser.show()

sys.exit(app.exec\_())

# UPDATEASSETS.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

class Ui\_Assets(object):

def setupUi(self, Assets):

Assets.setObjectName("Assets")

Assets.resize(494, 481)

Assets.setStyleSheet("background-color: rgb(145, 169, 208);")

self.centralwidget = QtWidgets.QWidget(Assets)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(140, 20, 231, 51))

self.Heading.setStyleSheet("font: 75 14pt \"Verdana\";\n"

"color: rgb(243, 231, 68);")

self.Heading.setObjectName("Heading")

self.Field\_3 = QtWidgets.QLabel(self.centralwidget)

self.Field\_3.setGeometry(QtCore.QRect(50, 90, 111, 41))

self.Field\_3.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_3.setObjectName("Field\_3")

self.Userid = QtWidgets.QLineEdit(self.centralwidget)

self.Userid.setGeometry(QtCore.QRect(250, 100, 171, 31))

self.Userid.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Userid.setObjectName("Userid")

self.Field\_4 = QtWidgets.QLabel(self.centralwidget)

self.Field\_4.setGeometry(QtCore.QRect(50, 140, 181, 41))

self.Field\_4.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_4.setObjectName("Field\_4")

self.Land = QtWidgets.QLineEdit(self.centralwidget)

self.Land.setGeometry(QtCore.QRect(250, 150, 171, 31))

self.Land.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Land.setObjectName("Land")

self.Cow = QtWidgets.QLineEdit(self.centralwidget)

self.Cow.setGeometry(QtCore.QRect(250, 220, 171, 31))

self.Cow.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Cow.setObjectName("Cow")

self.Chicken = QtWidgets.QLineEdit(self.centralwidget)

self.Chicken.setGeometry(QtCore.QRect(250, 320, 171, 31))

self.Chicken.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Chicken.setObjectName("Chicken")

self.Goat = QtWidgets.QLineEdit(self.centralwidget)

self.Goat.setGeometry(QtCore.QRect(250, 270, 171, 31))

self.Goat.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Goat.setObjectName("Goat")

self.Field\_5 = QtWidgets.QLabel(self.centralwidget)

self.Field\_5.setGeometry(QtCore.QRect(20, 180, 181, 41))

self.Field\_5.setStyleSheet("font: 75 11pt \"Verdana\";\n"

"color: rgb(255, 255, 0);")

self.Field\_5.setObjectName("Field\_5")

self.Field\_6 = QtWidgets.QLabel(self.centralwidget)

self.Field\_6.setGeometry(QtCore.QRect(50, 220, 111, 31))

self.Field\_6.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_6.setObjectName("Field\_6")

self.Field\_7 = QtWidgets.QLabel(self.centralwidget)

self.Field\_7.setGeometry(QtCore.QRect(50, 270, 141, 31))

self.Field\_7.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_7.setObjectName("Field\_7")

self.Field\_8 = QtWidgets.QLabel(self.centralwidget)

self.Field\_8.setGeometry(QtCore.QRect(50, 320, 141, 31))

self.Field\_8.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_8.setObjectName("Field\_8")

self.Action = QtWidgets.QPushButton(self.centralwidget,clicked= lambda:self.Update\_Assets())

self.Action.setGeometry(QtCore.QRect(170, 370, 131, 51))

self.Action.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"font: 75 10pt \"Verdana\";\n"

"color: rgb(3, 111, 252);")

self.Action.setObjectName("Action")

self.Output = QtWidgets.QLabel(self.centralwidget)

self.Output.setGeometry(QtCore.QRect(40, 430, 411, 31))

self.Output.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Output.setText("")

self.Output.setObjectName("Output")

Assets.setCentralWidget(self.centralwidget)

self.retranslateUi(Assets)

QtCore.QMetaObject.connectSlotsByName(Assets)

def Update\_Assets(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Assets=pd.read\_sql\_table("assets",engine)

UserID=Assets[["UserID"]].to\_numpy()

UID=int(self.Userid.text())

if UID in UserID:

if self.Land.text():

Area=int(self.Land.text())

Assets.loc[Assets["UserID"]==UID, "Land\_Area"]= Area

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

if self.Cow.text():

C=int(self.Cow.text())

Assets.loc[Assets["UserID"]==UID, "Cows"]= C

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

if self.Goat.text():

G=int(self.Goat.text())

Assets.loc[Assets["UserID"]==UID, "Goats"]= G

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

if self.Chicken.text():

Ch=int(self.Chicken.text())

Assets.loc[Assets["UserID"]==UID, "Chicken"]= Ch

Assets.to\_sql(name="assets",con=engine,if\_exists="replace",index=False)

self.Output.setText("Assets Successfully Updated")

return

else:

self.Output.setText("Invalid User ID")

return

def retranslateUi(self, Assets):

\_translate = QtCore.QCoreApplication.translate

Assets.setWindowTitle(\_translate("Assets", "Update User Assets"))

self.Heading.setText(\_translate("Assets", "UPDATE USER ASSETS"))

self.Field\_3.setText(\_translate("Assets", "USER ID"))

self.Userid.setPlaceholderText(\_translate("Assets", "Enter User ID"))

self.Field\_4.setText(\_translate("Assets", "Total Land (in sq. metres)"))

self.Land.setPlaceholderText(\_translate("Assets", "Leave empty if no change"))

self.Cow.setPlaceholderText(\_translate("Assets", "Leave empty if no change"))

self.Chicken.setPlaceholderText(\_translate("Assets", "Leave empty if no change"))

self.Goat.setPlaceholderText(\_translate("Assets", "Leave empty if no change"))

self.Field\_5.setText(\_translate("Assets", "Cattle Data"))

self.Field\_6.setText(\_translate("Assets", "Number of Cows"))

self.Field\_7.setText(\_translate("Assets", "Number of Goats"))

self.Field\_8.setText(\_translate("Assets", "Number of Chickens"))

self.Action.setText(\_translate("Assets", "UPDATE"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

Assets = QtWidgets.QMainWindow()

ui = Ui\_Assets()

ui.setupUi(Assets)

Assets.show()

sys.exit(app.exec\_())

# UPDATER.PY

from PyQt5 import QtCore, QtGui, QtWidgets

from AddUser import \*

from RemoveUser import \*

from UpdateUser import \*

from UpdateAssets import \*

class Ui\_Updation(object):

def setupUi(self, Updation):

Updation.setObjectName("Updation")

Updation.resize(481, 237)

Updation.setStyleSheet("background-color: rgb(121, 139, 145);")

self.centralwidget = QtWidgets.QWidget(Updation)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(110, 20, 281, 41))

self.Heading.setStyleSheet("color: rgb(15, 255, 247);\n"

"font: 75 16pt \"Magneto\";")

self.Heading.setObjectName("Heading")

self.NEW\_USER = QtWidgets.QPushButton(self.centralwidget, clicked = lambda: self.Add())

self.NEW\_USER.setGeometry(QtCore.QRect(20, 100, 131, 41))

self.NEW\_USER.setStyleSheet("color: rgb(255, 0, 0);\n"

"background-color: rgb(0, 255, 255);\n"

"font: 75 10pt \"MS Shell Dlg 2\";")

self.NEW\_USER.setObjectName("NEW\_USER")

self.Update\_USER = QtWidgets.QPushButton(self.centralwidget, clicked = lambda: self.Updatedata())

self.Update\_USER.setGeometry(QtCore.QRect(170, 100, 141, 41))

self.Update\_USER.setStyleSheet("color: rgb(255, 0, 0);\n"

"background-color: rgb(0, 255, 255);\n"

"font: 75 10pt \"MS Shell Dlg 2\";")

self.Update\_USER.setObjectName("Update\_USER")

self.Remove\_User = QtWidgets.QPushButton(self.centralwidget, clicked = lambda: self.Remove())

self.Remove\_User.setGeometry(QtCore.QRect(330, 100, 131, 41))

self.Remove\_User.setStyleSheet("color: rgb(255, 0, 0);\n"

"background-color: rgb(0, 255, 255);\n"

"font: 75 10pt \"MS Shell Dlg 2\";")

self.Remove\_User.setObjectName("Remove\_User")

self.Assets = QtWidgets.QPushButton(self.centralwidget, clicked = lambda: self.Update\_Asset())

self.Assets.setGeometry(QtCore.QRect(160, 170, 161, 41))

self.Assets.setStyleSheet("color: rgb(255, 0, 0);\n"

"background-color: rgb(0, 255, 255);\n"

"font: 75 10pt \"MS Shell Dlg 2\";")

self.Assets.setObjectName("Assets")

Updation.setCentralWidget(self.centralwidget)

self.retranslateUi(Updation)

QtCore.QMetaObject.connectSlotsByName(Updation)

def Add(self):

self.Adduser= QtWidgets.QMainWindow()

self.ui = Ui\_NEWUSER()

self.ui.setupUi(self.Adduser)

self.Adduser.show()

def Remove(self):

self.Removeuser=QtWidgets.QMainWindow()

self.ui = Ui\_RemoveUser()

self.ui.setupUi(self.Removeuser)

self.Removeuser.show()

def Updatedata(self):

self.updater=QtWidgets.QMainWindow()

self.ui = Ui\_Update\_Data()

self.ui.setupUi(self.updater)

self.updater.show()

def Update\_Asset(self):

self.updateA=QtWidgets.QMainWindow()

self.ui= Ui\_Assets()

self.ui.setupUi(self.updateA)

self.updateA.show()

def retranslateUi(self, Updation):

\_translate = QtCore.QCoreApplication.translate

Updation.setWindowTitle(\_translate("Updation", "Financial Advisor"))

self.Heading.setText(\_translate("Updation", "UPDATE DATABASE"))

self.NEW\_USER.setText(\_translate("Updation", "ADD NEW USER"))

self.Update\_USER.setText(\_translate("Updation", "UPDATE USER DATA"))

self.Remove\_User.setText(\_translate("Updation", "REMOVE USER"))

self.Assets.setText(\_translate("Updation", "UPDATE USER ASSETS"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

Updation = QtWidgets.QMainWindow()

ui = Ui\_Updation()

ui.setupUi(Updation)

Updation.show()

sys.exit(app.exec\_())

# UPDATEUSER.PY

from PyQt5 import QtCore, QtGui, QtWidgets

import pandas as pd

import numpy as np

import sqlalchemy

class Ui\_Update\_Data(object):

def setupUi(self, Update\_Data):

Update\_Data.setObjectName("Update\_Data")

Update\_Data.resize(496, 404)

Update\_Data.setStyleSheet("background-color: rgb(157, 206, 177);")

self.centralwidget = QtWidgets.QWidget(Update\_Data)

self.centralwidget.setObjectName("centralwidget")

self.Heading = QtWidgets.QLabel(self.centralwidget)

self.Heading.setGeometry(QtCore.QRect(150, 20, 201, 41))

self.Heading.setStyleSheet("font: 75 14pt \"Verdana\";\n"

"color: rgb(243, 231, 68);")

self.Heading.setObjectName("Heading")

self.Field = QtWidgets.QLabel(self.centralwidget)

self.Field.setGeometry(QtCore.QRect(90, 140, 111, 41))

self.Field.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field.setObjectName("Field")

self.Field\_2 = QtWidgets.QLabel(self.centralwidget)

self.Field\_2.setGeometry(QtCore.QRect(90, 200, 111, 41))

self.Field\_2.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_2.setObjectName("Field\_2")

self.Balance = QtWidgets.QLineEdit(self.centralwidget)

self.Balance.setGeometry(QtCore.QRect(280, 150, 171, 31))

self.Balance.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Balance.setObjectName("Balance")

self.Income = QtWidgets.QLineEdit(self.centralwidget)

self.Income.setGeometry(QtCore.QRect(280, 200, 171, 31))

self.Income.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Income.setObjectName("Income")

self.Action = QtWidgets.QPushButton(self.centralwidget, clicked= lambda: self.Update\_User())

self.Action.setGeometry(QtCore.QRect(180, 280, 121, 51))

self.Action.setStyleSheet("background-color: rgb(255, 255, 0);\n"

"font: 75 10pt \"Verdana\";\n"

"color: rgb(3, 111, 252);")

self.Action.setObjectName("Action")

self.Field\_3 = QtWidgets.QLabel(self.centralwidget)

self.Field\_3.setGeometry(QtCore.QRect(90, 90, 111, 41))

self.Field\_3.setStyleSheet("font: 75 11pt \"MS Shell Dlg 2\";\n"

"color: rgb(255, 255, 0);")

self.Field\_3.setObjectName("Field\_3")

self.Userid = QtWidgets.QLineEdit(self.centralwidget)

self.Userid.setGeometry(QtCore.QRect(280, 100, 171, 31))

self.Userid.setStyleSheet("background-color: rgb(255, 255, 255);")

self.Userid.setObjectName("Userid")

self.Output = QtWidgets.QLabel(self.centralwidget)

self.Output.setGeometry(QtCore.QRect(60, 350, 391, 41))

self.Output.setStyleSheet("color: rgb(0, 0, 127);\n"

"font: 75 11pt \"MS Shell Dlg 2\";")

self.Output.setText("")

self.Output.setObjectName("Output")

Update\_Data.setCentralWidget(self.centralwidget)

self.retranslateUi(Update\_Data)

QtCore.QMetaObject.connectSlotsByName(Update\_Data)

def Update\_User(self):

engine=sqlalchemy.create\_engine("mysql+pymysql://BANK:Benk@localhost:3306/BANK")

Recs= pd.read\_sql\_table("records",engine)

UserID=Recs["User\_ID"].to\_numpy()

UID=int(self.Userid.text())

if UID in UserID:

if self.Income.text():

New\_Inc=int(self.Income.text())

Recs.loc[Recs["User\_ID"]==UID, "Income"]=New\_Inc

Recs.to\_sql(name="records",con=engine,if\_exists="replace",index=False)

if self.Balance.text():

Amt=int(self.Balance.text())

Recs.loc[Recs["User\_ID"]==UID, "Balance"]=Amt

Recs.to\_sql(name="records",con=engine,if\_exists="replace",index=False)

self.Output.setText("Database Successfully Updated")

return

else:

self.Output.setText("User ID not found")

return

def retranslateUi(self, Update\_Data):

\_translate = QtCore.QCoreApplication.translate

Update\_Data.setWindowTitle(\_translate("Update\_Data", "Update User Data"))

self.Heading.setText(\_translate("Update\_Data", "UPDATE USER DATA"))

self.Field.setText(\_translate("Update\_Data", "New Balance"))

self.Field\_2.setText(\_translate("Update\_Data", "New Income"))

self.Balance.setPlaceholderText(\_translate("Update\_Data", "Leave empty if no change"))

self.Income.setPlaceholderText(\_translate("Update\_Data", "Leave empty if no change"))

self.Action.setText(\_translate("Update\_Data", "UPDATE"))

self.Field\_3.setText(\_translate("Update\_Data", "USER ID"))

self.Userid.setPlaceholderText(\_translate("Update\_Data", "Enter User ID"))

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

Update\_Data = QtWidgets.QMainWindow()

ui = Ui\_Update\_Data()

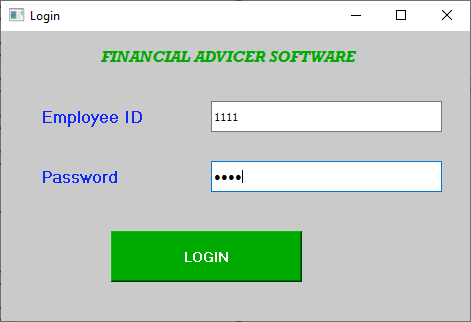
ui.setupUi(Update\_Data)

Update\_Data.show()

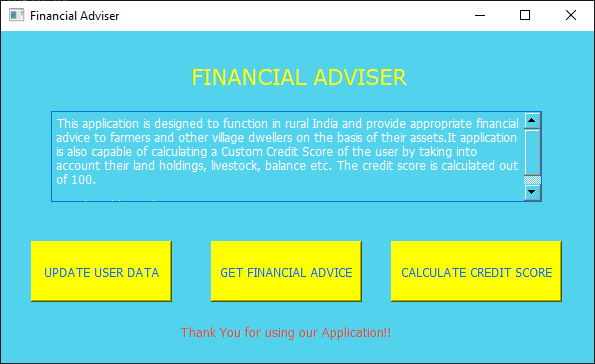
sys.exit(app.exec\_())

PROJECT OUTPUT

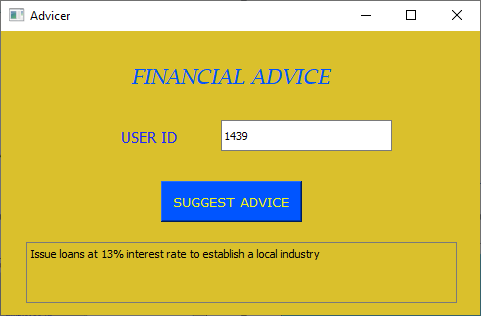
# LOGIN PAGE



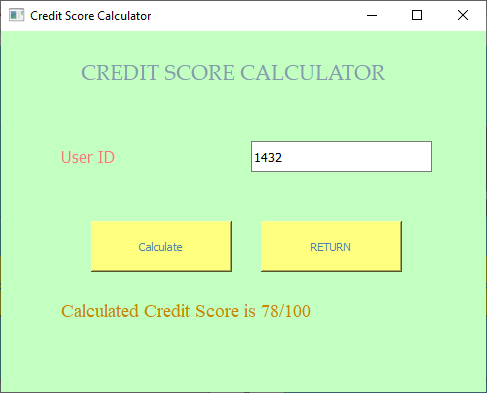
# MAIN WINDOW



# ADVISER



# CALCULATOR



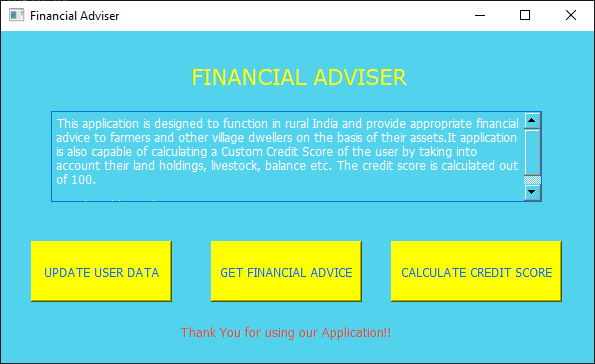
USER MANUAL

The Login page accepts Employee ID and Password and grants access to the software. The employee id should already be registered and saved on our database then only you will get to access to use our application.

Graphical user interface, application

Description automatically generated

Once you login in our application you will get three choices to update user data, get financial advice or calculate credit score. You can choose according to your preference and need.



Once you come to this screen if you choose to update user data, click on “UPDATE USER DATA” button. You will be able to see the following screen with four option to add user, update user data, remove user data and to update user’s assests.  
Graphical user interface, website

Description automatically generated

If you choose to add new user click on “Add New User” and you will be able to see the following screen. In this screen you will be able to see to add user’s name, their income and their balance. Once you enter these details our system will provide them with unique user id and you will be able to use that unique user id to access their data anytime you need. Click on add user to finish the process.

Graphical user interface, application

Description automatically generated  
If you choose to update user data click on “UPDATE USER DATA” button. In this option you will be able to update the balance and income of a person using their unique user id.

Graphical user interface

Description automatically generated  
If you choose to remove user click on “REMOVE USER” button to delete the record of a person permanently from the server.

Graphical user interface, application

Description automatically generated

If you wish to update user’s assets click on “UPDATE USER ASSEST” button. Here you will be able to change the area of land the user owns and update their cattle count by using their unique user id.

Graphical user interface, application

Description automatically generated  
 If you wish to get Financial Advice for a registered person click on “GET FINANCIAL ADVICE” button on the login page and you will be able to see the following on your screen. Here you will get few financial advice for the respective person you would like to know for by entering their unique user id.

Graphical user interface

Description automatically generated

If you wish to know Credit Score of a registered person click on “CALCULATE CREDIT SCORE” button on the login screen and you will be able to see the following screen. Here you will be able to see the credit score of the person by entering their user id.  
  
Graphical user interface, website

Description automatically generated

The credit score is calculated on the basis of various user assets-

1. Balance
2. Income
3. Land
4. Livestock

BIBILIOGRAPHY

* <https://docs.python.org/3/>
* <https://matplotlib.org/stable/users/index.html>
* <https://doc.qt.io/qtforpython/>
* <https://scikit-learn.org/0.21/documentation.html>
* NCERT Computer Science textbook