

# FESPA

**BY**  
Varsha Balaji

## Index

S. No	Content	Page Number
1	Acknowledgement	3
2	Synopsis	4
3	Salient Features	6
4	Project Analysis	7
5	Project Listing	12
6	Output	43

## **Synopsis**

**FESPA** consists of 5 main modules:

- 1. Monthly Income**
- 2. Yearly Income**
- 3. Current Net Worth**
- 4. Financial Freedom**
- 5. Retirement Planning**

Financial income /expense /savings data has been tabulated for 2 people who started their respective careers in 2005 at the age of 23. Both have had similar growth in their career but differed in the way that they spent their money and saved it. The 2 people are Savitri and Shyam.

## **1 & 2) Monthly Income, Yearly Income**

Monthly Income, Yearly income represents the revenue generated by the household of Savitri and Shyam on a monthly and yearly basis respectively after taking into account their salaries, other sources of income, rent, cost of education for their offspring, groceries, vegetables, essential and non essential articles of clothing, entertainment, various types of insurance, fixed and equity investments, tax and transportation.

## **3) Current Net Worth**

This computes the current monetary value that savitri/shyam holds with the help of 2 main parameters:

- Expected Average Equity Return
- Expected Fixed Average Income Rate

## **4) Financial Freedom**

Financial Freedom is achieved if income generated from one's net worth (Equity/Fixed income savings) can substitute salary income. If a person's worth is 25 times their expense, they are said to have achieved financial independence. We take the previous year's annual expense and apply inflation to it to calculate the next years expense and we repeat it until their net worth is more than 25 times her annual expense.

## **5) Retirement Planning**

Using the previously calculated current net worth, we apply Equity and Fixed income returns to predict what the person's net worth would be at 60. We also predict the person's annual expenses at 60 based on their current year's expenses and use Inflation to know what the expense that they incur at 60 would be.

## **Salient Features**

- Usage of user's financial data to predict financial independence and suggest effective retirement planning
- Usage of graph plotting libraries to add a visual aspect to the user's finances
- Usage of html to create a website for easy access to our program
- Allows creation of a user profile on our website
- Usage of csv files to store user's data
- Usage of mysql workbench as a database
- Checks for validity of user's data to prevent errors and provide a satisfactory experience
- Uses mouse interface

## Project Analysis

Our project has been developed using python as our primary programming language and html as our secondary programming language and MySQL workbench as our database. Some of the concepts used in our program are:

- Arrays
- Loops
- CSV files
- Graphs
- Functions
- SQL-Python Connector
- HTML-Python Connector

**FESPA** uses many functions, each with a distinct purpose.

The various functions used are:

### **1. Add\_header()**

*This function changes settings of the header which holds all the TCP protocols. This function makes sure that no cache is being stored and each time a fresh resource is called from the server, displaying the static resources such as an image.*

### **2. Index()**

*This function is used to display the home page of our website pre logging in.*

### **3. Home()**

*This function displays the Home page of our website post logging in.*

### **4. AboutPreLogin()**

*This function displays the about page of our website before logging in.*

### **5. About()**

*This function displays the about page of our website post logging in.*

## **6. Login()**

*This function displays the login page of our website.*

## **7. PostLogin()**

*This Function displays the login page where the user enters their Email ID and password.*

## **8. Signup()**

*This Function displays the signup page of our website.*

## **9. Log()**

*This Function displays the details of the registered customer and reports the success or failure of the user's attempt to login to our website.*

## **10. BrowseFiles()**

*This Function displays the browse page of our website which redirects you to the report page to view desired output.*

## **11. Import()**

*This function displays the import page of our website.*

## **12. Report()**

*This function displays the report page of our website.*

## **13. MonthlyFinance()**

*This function displays the Monthly Finance page of our website.*

## **14. YearlyFinance()**

*This function displays the Yearly Finance page of our website.*

## **15. Final()**

*This function displays the monthly or yearly profile (in both numerical and graphical form) of the user depending on the user's choice.*

## **16. CurrentNetWorth()**

*This function displays the current net worth page of our website.*

## **17. CurrentFinal()**

*This function takes in average interest and returns of the user and displays the user's net worth in both numerical and graphical form*

## **18. FinanceFreedom()**

*This function displays the financial independence page of our website.*

## **19. FinanceFinal()**

*This function takes in average returns of the user, average interest rates and rates of inflation to display financial independence data and the user's future net worth in both numerical and graphical form.*

## **20. RetirementPlanning()**

*This function displays the retirement planning page of our website.*

## **21. RetirementFinal()**

*This function takes in average returns of the user, average interest rates and inflation rates along with the user's ideal retirement age and the user's life expectancy to suggest an effective retirement plan and display future net worth in both numerical and graphical form.*

## **22. RegisterCustomer()**

*This function registers the user with our website after accepting their personal details.*

## **23. LoginCustomer()**

*This function checks whether the users login details are true and correct and logs the user into our website.*

## **24. ImportFinancialData()**

*This function imports the financial data of the user from their respective CSV file into our MySQL Workbench database from which we use the data to perform various operations listed above.*



## **25. MonthlyProfileCalculation()**

*This function calculates the monthly profile of the given user which consists of their income, expenses and savings.*

## **26. YearlyProfileCalculation()**

*This function calculates the yearly profile of the given user which consists of their income, expenses and savings.*

## **27. Graph()**

*This function creates a pie chart which represents the user's income, expenses and savings.*

## **28. Graph2()**

*This function creates a pie chart which represents the user's current equity net worth and current fixed income net worth.*

## **29. Graph3(), Graph4()**

*These functions create a pie chart which represent the user's future equity net worth and future income net worth.*

*Graph3() is saved as graph4.png and is used in the financial freedom page of our website, whereas Graph4() is saved as graph5.png and is used in the retirement planning page of our website.*

## **30. Networth\_Calculation()**

*This function mathematically calculates the current net worth of the user and returns their current equity net worth, current fixed income net worth and their total net worth.*

## **31. GetFutureNetworth()**

*This function mathematically calculates the future net worth of the user and returns their future equity net worth, future fixed income net worth and their total future net worth.*

## **32. FinancialFreedomCal()**

*This function predicts when the user can achieve financial independence after taking into account inflation rates, equity returns and interest received from fixed income.*

### **33. RetirementPlan():**

*This function tells the user when they can retire comfortably and how many years worth of expenses they will have, after taking into account inflation rates, equity returns and interest received from fixed income.*

### **34. mysql\_Connector():**

*This function tells the user whether a connection to our MySQL database was successful or not.*

## Project Listing

### APP.PY

```
from flask import Flask , render_template , request
from MySqlConnection import *
import mysql.connector
import csv
import sys, os, traceback
import pickle
import tkinter
from tkinter import *
from tkinter import filedialog
```

```
import numpy as np
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import datetime
```

```
from flask_cachebuster import CacheBuster
```

```
app = Flask(__name__)
LoginDetails=[]
l=[]
ld=""
h=[]
em=""
g=""
keys=[]
d={}
t=()
i=[]
e=[]
f=()
```

```
@app.after_request
def add_header(r):
```

```

r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
r.headers["Pragma"] = "no-cache"
r.headers["Expires"] = "0"
r.headers['Cache-Control'] = 'public, max-age=0'
return r

```

```

@app.route('/')
def index():
    title = "Fespa"
    return render_template("home.html", title=title )

```

```

@app.route('/Home')
def Home():
    title = "Home"
    return render_template("homePostLogin.html", title=title )

```

```

@app.route("/AboutPreLogin")
def AboutPreLogin():
    title = "Fespa"
    return render_template("aboutPost.html", title=title)

```

```

@app.route("/About")
def About():
    title = "About"
    return render_template("about.html", title=title)

```

```

@app.route('/Login')
def Login():
    title = "Login"
    return render_template("login.html", title=title)

```

```

@app.route('/PostLogin' , methods=["POST"])
def PostLogin():
    title = "PostLogin"
    email = request.form.get("emailId")
    password = request.form.get("password")
    h,t,f=LoginCustomer(email,password)
    return render_template(f, title=h ,h=h ,t=t)

```

```

@app.route('/SignUp', methods=["POST"])
def SignUp():
    title="SignUp"
    return render_template("sign_up.html", title=title)

@app.route('/Calculate', methods=["POST"])
def Calculate():
    title="Calculate"
    return render_template("calculate.html", title=title)

@app.route('/Log', methods=["POST"])
def Log():
    first_Name = request.form.get("First_Name")
    Second_Name = request.form.get("Second_Name")
    email = request.form.get("emailId")
    Mobile_Number = request.form.get("Mobile_Number")
    DOB= request.form.get("DOB")
    password = request.form.get("password")
    title="Log"
    t,h= RegisterCustomer(first_Name,Second_Name,email,Mobile_Number,DOB,password)
    return render_template( "log.html",title=title , rc=t ,rm=h)

@app.route('/browseFiles', methods=["POST"])
def browseFiles():
    global label_file_explorer
    global em
    title = "Browse"
    root = tkinter.Tk()
    label_file_explorer= tkinter.Label(root)
    filename = filedialog.askopenfilename(initialdir = "/", title = "Select a File", filetypes = (("Csv files",
".csv"), ("all files",".")))
    ImportFinanceData(filename,em )
    return render_template("browseFiles.html",title=title , h=filename)

@app.route('/Import')
def Import():
    title= "Import"
    return render_template("import.html", title=title)

```

```

@app.route("/Report")
def Report():
    title = "Report"
    return render_template("report.html", title=title)

@app.route("/Monthly_Finance" , methods=["POST"])
def Monthly_Finance():
    global g
    title = "Monthly Finance"
    g='monthly'
    return render_template("monthly_Finance.html", title=title)

@app.route("/Yearly_Finance" , methods=["POST"])
def Yearly_Finance():
    global g
    title = "Yearly Finance"
    g ="yearly"
    return render_template("yearly_Finance.html", title=title)

@app.route("/Final" , methods=["POST"])
def Final():
    Month =request.form.get("Month")
    Year=request.form.get("Year")
    global g
    global Id
    global t
    global i
    global e
    global f
    title = "Final"
    if g == 'monthly':
        t,i,e = Monthly_ProfileCalculation(Id,Month,Year)
        f=(t,i,e)
    elif g == 'yearly':
        t,i,e = Yearly_ProfileCalculation(Id,Year)
        f=(t,i,e)
    Graph(f)
    return render_template("final.html", title=title , t=t , i=i, e=e , g=g , Id=Id,f=f )

```

```

@app.route('/CurrentNetWorth', methods=["POST"])
def CurrentNetWorth():
    title="CurrentNetWorth"
    return render_template("currentNetworth.html", title=title)

@app.route('/CurrentFinal', methods=["POST"])
def CurrentFinal():
    global Id
    AvgReturn=request.form.get('AvgReturn')
    AvgIntrest=request.form.get('AvgIntrest')
    title="CurrentFinal"
    y= Networkworth_Calculation(Id,int(AvgReturn),int(AvgIntrest))
    Graph2(y)
    return render_template("currentFinal.html", title=title , k=y)

@app.route('/FinanceFreedom', methods=["POST"])
def FinanceFreedom():
    title='FinanceFreedom'
    return render_template('financeFreedom.html', title=title)

@app.route('/FinanceFinal', methods=["POST"])
def FinanceFinal():
    global Id
    AvgReturn=request.form.get('AvgReturnFF')
    AvgIntrest=request.form.get('AvgIntrestFF')
    Inflation=request.form.get('InflationFF')
    title='FinanceFinal'
    y=FinancialFreedomCal(Id, int(AvgReturn), int(AvgIntrest), int(Inflation))
    k=GetFutureNetworth(Id, int(AvgReturn), int(AvgIntrest), y[0])
    Graph3(k)
    if y[0]>=9999:
        msg="You cannot achieve financial independence based on your expense/investment patterns"
    else:
        msg="Congratulations! You will achieve financial independence by Year "+str (y[0]) +"."+str(y[1])+" You
will be "+ str(y[1]) +" years old at this time."
    return render_template('financeFinal.html', title=title , msg=msg ,k=k)

@app.route('/RetirementPlanning', methods=["POST"])

```

```

def RetirementPlanning():
    title="Retirement Planning"
    return render_template("retirementPlanning.html",title=title)

@app.route('/RetirementFinal', methods=["POST"])
def RetirementFinal():
    global Id
    AvgReturn=request.form.get("AvgReturnRP")
    AvgIntrest=request.form.get("AvgIntrestRP")
    Inflation=request.form.get("InflationRP")
    retirementAge=request.form.get("RetirementAge")
    LifeExpectancy=request.form.get("LifeExpectancy")
    title="RetirementFinal"

    y=RetirementPlan(Id,int(AvgReturn),int(AvgIntrest),int(Inflation),int(retirementAge),int(LifeExpectancy)
    )
    k=GetFutureNetworth(Id,int(AvgReturn),int(AvgIntrest),y[1])
    Graph4(k)
    return render_template("retirementFinal.html" , title=title ,y=y ,k=k)

def RegisterCustomer(FirstName, LastName, EmailAddress, Mobile, DOB, Password):
    global Id
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    if MySQLReturnCode == 'Failure' :
        return(MySQLReturnCode,MySQLReturnMessage)
    try:
        mycursor =con.cursor()
        sql=("select count(*) from customer where emailid=%s")
        mycursor.execute(sql,(EmailAddress,))
        count = mycursor.fetchone()[0]
        if count>0:
            returnCode = "Failure"
            returnMesage = "Person already Registered"
            return(returnCode, returnMesage)
        sql="select max(CustomerID) from customer"
        mycursor.execute(sql)
        row=mycursor.fetchone()

```



```

if row[0] is None:
    CustomerID=1
else:
    MaxCustomerID = row[0]
    CustomerID= MaxCustomerID+1

Id=CustomerID
sql="insert into customer values (%s,%s,%s,%s,%s,%s,%s)"
values=(CustomerID,FirstName,LastName,EmailAddress,Mobile,DOB>Password)
mycursor.execute(sql,values)
con.commit()
except mysql.connector.Error as error:
    returnCode = "Failure"
    returnMessage = "Database Call Failed. Contact Technical Support " + format(error)
    return(returnCode, returnMessage)
except Exception as excp:
    exc_type, exc_obj, exc_traceback = sys.exc_info()
    traceback.print_tb(exc_traceback)
    tb = traceback.extract_tb(exc_traceback)[-1]
    returnCode = "Failure"
    returnMessage = "Failure in User Registration. Contact Technical Support " + str(exc_obj) + " Error
in Linenumber " + str(tb[1]) + " in python program " + str(tb[0])
    return(returnCode, returnMessage)
returnCode = "Success"
returnMessage = FirstName + " " + LastName + " Successfully Registered "
return(returnCode, returnMessage)

def LoginCustomer( EmailAddress, Password):
    global Id
    global em
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    if MySQLReturnCode == 'Failure' :
        return(MySQLReturnCode,MySQLReturnMessage)
    try:
        mycursor =con.cursor()
        sql=("select Password from customer where emailid=%s")
        mycursor.execute(sql,(EmailAddress,))
        row =mycursor.fetchone()

```

```

if row is None:
    returnCode = "Failure"
    returnMesage = "User Not Registered "
    returnTemplate = "error.html"
    return(returnCode, returnMesage , returnTemplate )

dbPassword = row[0]
mycursor.execute("select CustomerId from customer where emailid= %s" , (EmailAddress,))
fld=mycursor.fetchone()[0]
if dbPassword==Password:
    returnCode = "Success"
    returnMesage = "User Successfully logged in "
    returnTemplate = "form.html"
    Id=fld
    em=EmailAddress
    return(returnCode, returnMesage , returnTemplate )

else:
    returnCode = "Failure"
    returnMesage = "Invalid Password Entered"
    returnTemplate = "error.html"
    return(returnCode, returnMesage , returnTemplate )
except mysql.connector.Error as error:
    returnCode = "Failure"
    returnMesage = "Database Call Failed. Contact Technical Support " + format(error)
    returnTemplate = "error.html"
    return(returnCode, returnMesage, returnTemplate )
except Exception as excp:
    exc_type, exc_obj, exc_traceback = sys.exc_info()
    traceback.print_tb(exc_traceback)
    tb = traceback.extract_tb(exc_traceback)[-1]
    returnCode = "Failure"
    returnMesage = "Failure in User Login. Contact Technical Support " + str(exc_obj) + " Error in
Linenumber " + str(tb[1]) + " in python program " + str(tb[0])
    returnTemplate = "error.html"
    return(returnCode, returnMesage , returnTemplate )

```

```

def ImportFinanceData(path,Email):
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    mycursor =con.cursor()
    sql = ("SELECT CustomerID FROM customer WHERE Emailid = %s")
    mycursor.execute(sql,(Email,))
    CustomerID=mycursor.fetchone()[0]
    sql = ("DELETE FROM financedata where CustomerID = %s")
    mycursor.execute(sql,(CustomerID,))
    con.commit()
    f=open(path,"r",newline="")
    csvreader=csv.reader(f)
    L=[]
    eg=[]
    for row in csvreader:
        L.append(row)

    for i in range(0,len(L)):
        ddmmyyyy = L[i][0]

        AccEntryDate = ddmmyyyy[6:10]+'-'+ddmmyyyy[3:5]+'-'+ddmmyyyy[0:2]
        AccEntryItemDesc=L[i][1]
        Amount=L[i][2]
        AccEntryType=L[i][3]
        AccEntrySubType=L[i][4]
        sql="insert into financedata values (%s,%s,%s,%s,%s,%s)"
        values=(CustomerID,AccEntryDate,AccEntryItemDesc,Amount,AccEntryType,
AccEntrySubType)
        eg=eg+[values]

    mycursor.executemany(sql,eg)
    con.commit()

    f.close()

def Monthly_ProfileCalculation(CustomerID,Month,Year):
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    mycursor =con.cursor()
    sql="""SELECT Month(AccEntryDate) as Month, \

```

```

        AccEntryType as Type, \
        AccEntrySubType as SubType, \
        Sum(Amount) as Amount \
FROM financeapp.financedata \
WHERE
    CustomerId = %s AND \
    Month(AccEntryDate) = %s AND \
    Year(AccEntryDate) = %s AND \
    AccEntryType != 'One-Time-Expense' \
    GROUP BY Month, Type, SubType""""
mycursor.execute(sql,(CustomerID,Month,Year))
result = mycursor.fetchall()

income=[]
expense=[]
title = ('Income','Expenses & Savings')
for record in result:
    if record[1]=="Income":
        income=income + [ tuple ( [ record[1],record[2],float(record[3]) ] ) ]
    else:
        expense=expense + [ tuple( [ record[1],record[2],float(record[3]) ] ) ]
return (title, income,expense)

```

```

def Yearly_ProfileCalculation(CustomerID,Year):
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    mycursor =con.cursor()
    sql="""SELECT Year(AccEntryDate) as Year, \
            AccEntryType as Type, \
            AccEntrySubType as SubType, \
            Sum(Amount) as Amount \
FROM financeapp.financedata \
WHERE
    CustomerId = %s AND
    Year(AccEntryDate) = %s AND \
    AccEntryType != 'One-Time-Expense' \
    GROUP BY Year, Type, SubType""""
    mycursor.execute(sql,(CustomerID,Year))
    result = mycursor.fetchall()

```

```

title = ('Income','Expenses & Savings')
income=[]
expense=[]

for record in result:
    if record[1]=="Income":
        income=income + [ tuple ( [ record[1],record[2],float(record[3]) ] ) ]
    else:
        expense=expense + [ tuple( [ record[1],record[2],float(record[3]) ] ) ]
return (title,income,expense)

```

```

def Graph(tup):

```

```

def piechartcreation():
    global keys
    global d

    list1=tup
    income=list1[1][0][2]
    expense_nd=list1[2][0][2]
    expense_d=list1[2][1][2]
    investment=list1[2][2][2]
    values=[income,expense_nd,expense_d,investment]
    keys=["Income","Expense(Non discretionary)","Expense(Discretionary)","Investment"]

    d={}
    for i in range(len(values)):
        d[keys[i]]=values[i]

```

```

piechartcreation()

```

```

headers=[[keys[0]],[keys[1],keys[2],keys[3]]]
parameters= [[d[keys[0]]],[d[keys[1]],d[keys[2]],d[keys[3]]]]
colors=("#EC6B56","#FFC154","#47B39C")
wp={'linewidth': 1,'edgecolor':"black"}

```

```

def func(pct,allvalues):

```

```
absolute = int(pct / 100.*np.sum(allvalues))
return"{:.1f}%\n({:d} g)".format(pct, absolute)
```

```
for i in range(2):
```

```
    fig, ax = plt.subplots(figsize =(10, 7))
    wedges, texts, autotexts = ax.pie(parameters[i],
                                       autopct = lambda pct: func(pct, parameters[i]),
                                       labels = headers[i],
                                       colors = colors,
                                       startangle = 90,
                                       wedgeprops = wp,
                                       textprops = dict(color ="black"))
```

```
    ax.legend(wedges,headers[i], title ="Key", loc ="center left", bbox_to_anchor =(1,0,0.5,1))
    fig.set_facecolor("black")
    plt.savefig("static/graph"+str(i)+".png")
```

```
piechartcreation()
```

```
def Graph2(tup):
```

```
def piechartcreation():
    global keys
    global d
    list1=tup
    equity=list1[0]
    fixedIncome=list1[1]
    values=[equity,fixedIncome]
    keys=["CurrentEquityNetworth","CurrentFixedIncomeNetworth"]
```

```
    d={}
    for i in range(len(values)):
        d[keys[i]]=values[i]
```

```
piechartcreation()
```

```
headers=[[keys[0],keys[1]]]
```

```

parameters= [[d[keys[0]],d[keys[1]]]]
colors=("#EC6B56","#FFC154","#47B39C")
wp={'linewidth': 1,'edgecolor':"black"}

def func(pct,allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return"{:.1f}%\n({:d} g)".format(pct, absolute)

for i in range(1):
    fig, ax = plt.subplots(figsize =(10, 7))
    wedges, texts, autotexts = ax.pie(parameters[i],
                                      autopct = lambda pct: func(pct, parameters[i]),
                                      labels = headers[i],
                                      shadow = True,
                                      colors = colors,
                                      startangle = 90,
                                      wedgeprops = wp,
                                      textprops = dict(color ="black"))

    ax.legend(wedges,headers[i], title ="Key", loc ="center left", bbox_to_anchor =(1,0,0.5,1))
    fig.set_facecolor("black")
    plt.savefig("static/graph3.png")

piechartcreation()

def Graph3(tup):

    def piechartcreation():
        global keys
        global d
        list1=tup
        equity=list1[0]
        fixedIncome=list1[1]
        values=[equity,fixedIncome]
        keys=["FutureEquityNetworth","FutureIncomeNetworth"]

```

```

d={}
for i in range(len(values)):
    d[keys[i]]=values[i]

```

```
piechartcreation()
```

```

headers=[[keys[0],keys[1]]]
parameters= [[d[keys[0]],d[keys[1]]]]
colors=("#EC6B56","#FFC154","#47B39C")
wp={'linewidth': 1,'edgecolor':"black"}

```

```

def func(pct,allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return"{:.1f}%\n({:d} g)".format(pct, absolute)

```

```

for i in range(1):
    fig, ax = plt.subplots(figsize =(10, 7))
    wedges, texts, autotexts = ax.pie(parameters[i],
                                        autopct = lambda pct: func(pct, parameters[i]),
                                        labels = headers[i],
                                        shadow = True,
                                        colors = colors,
                                        startangle = 90,
                                        wedgeprops = wp,
                                        textprops = dict(color ="black"))

    ax.legend(wedges,headers[i], title ="Key", loc ="center left", bbox_to_anchor =(1,0,0.5,1))
    fig.set_facecolor("black")
    plt.savefig("static/graph4.png")

```

```
piechartcreation()
```

```
def Graph4(tup):
```

```

def piechartcreation():
    global keys
    global d
    list1=tup

```



```
equity=list1[0]
fixedIncome=list1[1]
values=[equity,fixedIncome]
keys=["FutureEquityNetworth","FutureIncomeNetworth"]
```

```
d={}
for i in range(len(values)):
    d[keys[i]]=values[i]
```

```
piechartcreation()
```

```
headers=[[keys[0],keys[1]]]
parameters= [[d[keys[0]],d[keys[1]]]]
colors=("#EC6B56","#FFC154","#47B39C")
wp={'linewidth': 1,'edgecolor':"black"}
```

```
def func(pct,allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return"{:.1f}%\n({:d} g)".format(pct, absolute)
```

```
for i in range(1):
    fig, ax = plt.subplots(figsize =(10, 7))
    wedges, texts, autotexts = ax.pie(parameters[i],
                                       autopct = lambda pct: func(pct, parameters[i]),
                                       labels = headers[i],
                                       shadow = True,
                                       colors = colors,
                                       startangle = 90,
                                       wedgeprops = wp,
                                       textprops = dict(color ="black"))

    ax.legend(wedges,headers[i], title ="Key", loc ="center left", bbox_to_anchor =(1,0,0.5,1))
    fig.set_facecolor("black")
    plt.savefig("static/graph5.png")
```

```
piechartcreation()
```

```
def Networth_Calculation(CustomerID,EquityReturnPct,FixedIncomeInterestPct,Year=0):
```

```

con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
mycursor =con.cursor()
if (Year == 0):
    sql="""SELECT AccEntryDate,AccEntryItemDesc, Amount,AccEntryType \
        FROM financeapp.financedata \
        WHERE CustomerID = %s AND \
        AccEntryType IN ( 'Investment' , 'One-Time-Expense') ORDER BY AccEntryDate ASC,
AccEntryType DESC"""mycursor.execute(sql,(CustomerID,))
else:
    sql="""SELECT AccEntryDate,AccEntryItemDesc, Amount,AccEntryType \
        FROM financeapp.financedata \
        WHERE CustomerID = %s AND \
        Year(AccEntryDate)<=%s AND
        AccEntryType IN ( 'Investment' , 'One-Time-Expense') ORDER BY AccEntryDate ASC,
AccEntryType DESC"""
    mycursor.execute(sql,(CustomerID,Year))
result = mycursor.fetchall()
EquityReturn=EquityReturnPct/100
FixedIncomeInterest=FixedIncomeInterestPct/100
CurrentEquityNetworth = 0.0
CurrentFixedIncomeNetworth = 0.0
for record in result:

    if record[1]== 'Equity Investment':
        CurrentEquityNetworth = CurrentEquityNetworth+ float(record[2])
        CurrentEquityNetworth = CurrentEquityNetworth*(1 + EquityReturn/12.0)

    elif record[1] == 'Fixed Income Investment (PF/PPF/FDs/Debt MF)':
        CurrentFixedIncomeNetworth=CurrentFixedIncomeNetworth+float(record[2])
        CurrentFixedIncomeNetworth = CurrentFixedIncomeNetworth*(1 +
FixedIncomeInterest/12.0)

    elif record[1] == 'One-Time-Expense':
        CurrentEquityNetworth = CurrentEquityNetworth - float(record[2])/2.0
        CurrentFixedIncomeNetworth = CurrentFixedIncomeNetworth - float(record[2])/2.0
CurrentEquityNetworth = round(CurrentEquityNetworth,2)
CurrentFixedIncomeNetworth = round(CurrentFixedIncomeNetworth,2)

```

```
return [CurrentEquityNetworth,CurrentFixedIncomeNetworth,(CurrentEquityNetworth +
CurrentFixedIncomeNetworth)]
```

```
def GetFutureNetworth(CustomerID,EquityReturnPct,FixedIncomeInterestPct,FutureYear):
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    mycursor =con.cursor()
    now = datetime.datetime.now()
    CurrentYear = now.year
    PreviousYear = CurrentYear - 1

    BaseYear = PreviousYear - 1
    BaseYearSavings = Yearly_ProfileCalculation(CustomerID,BaseYear)[2][2][2]

    PreviousYearSavings = Yearly_ProfileCalculation(CustomerID,PreviousYear)[2][2][2]
    InvestmentGrowthRate = round((PreviousYearSavings-BaseYearSavings)/BaseYearSavings,2)
```

```
sql="""SELECT
        AccEntryItemDesc as Type, \
        Sum(Amount) as Amount \
    FROM financeapp.financedata \
    WHERE
        CustomerId = %s AND
        Year(AccEntryDate) = %s AND \
        AccEntryType = 'Investment' \
    GROUP BY AccEntryItemDesc"""
```

```
mycursor.execute(sql,(CustomerID,PreviousYear))
result = mycursor.fetchall()
YearlyEquityInvestment=0
YearlyFixedIncomeInvestment=0
for record in result:
    if record[0]=='Equity Investment':
        YearlyEquityInvestment=float(record[1])
    if record[0]=='Fixed Income Investment (PF/PPF/FDs/Debt MF)':
        YearlyFixedIncomeInvestment=float(record[1])
```

```
EquityReturn = EquityReturnPct/100
```

```
FixedIncomeInterest = FixedIncomeInterestPct/100
```

```
Networth = Networth_Calculation(CustomerID,EquityReturnPct,FixedIncomeInterestPct,  
PreviousYear)
```

```
EquityNetworth = Networth[0]
```

```
FixedIncomeNetworth = Networth[1]
```

```
NumOfFutureYears = FutureYear - PreviousYear
```

```
for i in range(NumOfFutureYears):
```

```
    YearlyEquityInvestment = YearlyEquityInvestment * (1+InvestmentGrowthRate)
```

```
    EquityNetworth = EquityNetworth * (1+EquityReturn) + YearlyEquityInvestment
```

```
    YearlyFixedIncomeInvestment = YearlyFixedIncomeInvestment*(1+InvestmentGrowthRate)
```

```
    FixedIncomeNetworth = FixedIncomeNetworth * (1+FixedIncomeInterest) +
```

```
    YearlyFixedIncomeInvestment
```

```
    return [round(EquityNetworth,2) , round(FixedIncomeNetworth,2) , round((EquityNetworth +  
FixedIncomeNetworth),2)]
```

```
def FinancialFreedomCal(CustomerID,EquityReturnPct,FixedIncomeInterestPct,InflationPct):
```

```
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
```

```
    mycursor =con.cursor()
```

```
    sql="""SELECT Year(DateofBirth) \  
        FROM  financeapp.customer \  
        WHERE CustomerId = %s"""
```

```
    mycursor.execute(sql,(CustomerID,))
```

```
    result = mycursor.fetchall()
```

```
    YearOfBirth = result[0][0]
```

```
    Year100 = YearOfBirth + 100
```

```
    Inflation = InflationPct/100
```

```
    now = datetime.datetime.now()
```

```
    CurrentYear = now.year
```

```
    PreviousYear = CurrentYear - 1
```

```
    BaseYear = PreviousYear - 1
```

```

BaseYearExpense =
Yearly_ProfileCalculation(CustomerID,BaseYear)[2][0][2]+Yearly_ProfileCalculation(CustomerID,BaseYear)[2][1][2]
CurrentYearExpense = BaseYearExpense * ( 1 + Inflation)

EquityReturn = EquityReturnPct/100
FixedIncomeInterest = FixedIncomeInterestPct/100

CurrentNetworth = Networth_Calculation(CustomerID,EquityReturnPct,
FixedIncomeInterestPct)[2]

if ( (CurrentYearExpense * 25) <= CurrentNetworth):
    return (CurrentYear, CurrentYear - YearOfBirth)

NextYear = CurrentYear
NextYearExpense = CurrentYearExpense
NextYearNetworth = CurrentNetworth

while ( ( (NextYearExpense * 25) > NextYearNetworth ) and ( NextYear <= Year100 ) ):
    NextYear = NextYear + 1
    NextYearExpense = NextYearExpense * (1 + Inflation)
    NextYearNetworth = GetFutureNetworth(CustomerID,EquityReturnPct,
FixedIncomeInterestPct,NextYear)[2]

if NextYear > Year100 :
    return (9999,99)

AgeWhenFinancialFreedomAchieved = NextYear - YearOfBirth
return (NextYear,AgeWhenFinancialFreedomAchieved)

def RetirementPlan(CustomerID,EquityReturnPct,FixedIncomeInterestPct,InflationPct,
RetirementAge,LifeExpectancy):
    con,MySQLReturnCode,MySQLReturnMessage=mysql_connection()
    mycursor =con.cursor()

    sql="""SELECT Year(DateOfBirth) \
        FROM  financeapp.customer \
        WHERE  CustomerId = %s"""

```

```

mycursor.execute(sql,(CustomerID,))
result = mycursor.fetchall()
YearOfBirth = result[0][0]
RetirementYear=YearOfBirth + RetirementAge
FutureYear=YearOfBirth + LifeExpectancy
NumberOfYearsInRetirement=LifeExpectancy-RetirementAge

Inflation = InflationPct/100

now = datetime.datetime.now()
CurrentYear = now.year
PreviousYear = CurrentYear - 1
PreviousYearExpense = Yearly_ProfileCalculation(CustomerID,PreviousYear)[2][0][2]
+Yearly_ProfileCalculation(CustomerID,PreviousYear)[2][1][2]
CurrentYearExpense = PreviousYearExpense * (1 + Inflation)
RetirementYearExpense = CurrentYearExpense * (1 + Inflation)**(RetirementYear-CurrentYear)
RetirementYearNetworth = GetFutureNetworth(CustomerID,EquityReturnPct,
FixedIncomeInterestPct,RetirementYear)[2]

EquityReturn = EquityReturnPct/100
FixedIncomeInterest = FixedIncomeInterestPct/100
YearsOfStability=RetirementYearNetworth//RetirementYearExpense
if ( (RetirementYearExpense * NumberOfYearsInRetirement) <= RetirementYearNetworth):

    returntext="You can comfortably Retire at " + str(RetirementAge) + ". You will have " +
str(YearsOfStability) + " Years of expenses covered."
else:
    returntext="You can't Retire at " + str(RetirementAge) + ". You only have " +
str(YearsOfStability) + " Years of expenses covered, upto age of " +
str(int(RetirementAge+YearsOfStability)) + "."

return [returntext,RetirementYear]

```

MySQLConnector.py:

```
def mysql_connection():
```

```

import mysql.connector

try:
    con= mysql.connector.connect(user="root",password="santhome@58",
auth_plugin="mysql_native_password",database='financeapp')

except mysql.connector.Error as error:
    returnMesage = "Something went wrong: " + format(error)
    return(None,"Failure",returnMesage)
    return(con,"Success","Success")

```

### HTML TEMPLATES

Base.html , BasePreLogin.html :

```

<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0JlfiDPvq6uqKI2xXr2"
crossorigin="anonymous">

    <title>{{ title }}</title>
</head>
<body style='background-image: url("static/bg.png"); '>
    <nav class="navbar navbar-expand-lg " style='background-color:#1F2833'>
    <a class="navbar-brand text-white" style="font-family:bahnschrift Semibold" href="{{ url_for('Home')
}}">HOME</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
            <li class="nav-item">

```

```

    <a class="nav-link text-white" href="{{ url_for('About') }}">About</a>
  </li>
  <li class="nav-item">
    <a class="nav-link text-white" href="{{ url_for('Import') }}">Import</a>
  </li>
  <li class="nav-item">
    <a class="nav-link text-white" href="{{ url_for('Report') }}">Report</a>
  </li>
  <li class="nav-item">
    <a class="nav-link text-white" href="{{ url_for('index') }}">LogOut</a>
  </li>
</ul>
</div>
</nav>
<br>

```

```

{% block content %}
{% endblock %}

```

```

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"
crossorigin="anonymous"></script>
</body>
</html>

```

### **home.html:**

```

{% extends 'basePreLogin.html' %}
{% block content %}
  <div class="container" style="color:#66FCF1">
    <br>
    <br>
    <br>
    <br>
  </div>

```



<u style="color:#66FCF1"><h1 class= "text-center bi-type-underline " style="font-family:bahnschrift light" >PERSONALIZED CLIENT EXPERIENCE</h1></u>

<h5 class=" text-center" style=" font-family:bahnschrift light">Taking Care Of What's Important So You Can Focus On What Matters To You.</h5>

<br>

<br>

<br>

<br>

<br>

<h5 style="color:#66FCF1"><h1 class= "text-center bi-type-underline " style="font-family:bahnschrift light" >FESPA</h5>

<h5 class=" text-center" style=" font-family:bahnschrift light">Financial Expense-Saving Planning App</h5>

<br>

<br>

<br>

<br>

<br>

<br>

</div>

<div>

<h5 class="text-white text-center" style=" font-family:bahnschrift light">Contact Us At</h5>

<p class="text-white text-center" style=" font-family:bahnschrift light" >+91 98409 31317 | Fespa@gmail.com</p>

</div>

{% endblock %}

### **about.html:**

{% extends 'base.html' %}

{% block content %}

<div class="container text-white" style="font-family:bahnschrift light">

<h1>About page</h1>

<br>

<h4>

Financial Planning Web Software that catalogues a person's income, expenses and investments, and provides projections for future investments and retirement simulations.

</h4>

<br>

<h4>

Financial income/expense/savings data has been tabulated for 2 people who started their respective careers in 2005 at the age of 23. Both had similar growth in their career but differed in the way they spent their money and saved. The 2 people are Savitri and Shyam.

</h4>

<br>

<br>

<h3>Developed By:</h3>

<h5> 1.Vignesh </h5>

<h5> 2.Varsha </h5>

<h5> 3.Ramnath </h5>

<h5> 4.Niveditha </h5>

</div>

{% endblock %}

#### **login.html:**

{% extends 'basePreLogin.html' %}

{% block content %}

<div class="container">

<h1 class="text-white ">Login</h1>

<br/><br/>

<form action="PostLogin" method="POST">

<div class="form-group">

<label class="text-white " for="exampleInputEmail1">Email address</label>

<input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" name="emailId">

</div>

<div class="form-group">

<label class="text-white " for="exampleInputPassword1">Password</label>

<input type="password" class="form-control" id="exampleInputPassword1" name="password">

</div>

<button type="submit" class="btn btn-primary">Submit</button>

<br>

<br>

```

</form>
    <form action="SignUp" method="POST">
        <div class="form-group">
            <button type="submit" class="btn btn-secondary">SignUp</button>
        </div>
    </form>
</div>

```

### **sign\_up.html:**

```

{% extends 'basePreLogin.html' %}
{% block content %}
    <div class="container">
        <h1 class="text-white">Sign Up Page</h1>

        <form action="Log" method="POST">
            <p class="text-white">Name</p>
            <div class="row">
                <div class="col">
                    <input type="text" class="form-control" placeholder="First name" name="First_Name">
                </div>
                <div class="col">
                    <input type="text" class="form-control" placeholder="Last name" name="Second_Name">
                </div>
            </div>
            <div class="form-group">
                <label class="text-white" for="Email">Email address</label>
                <input type="email" class="form-control" placeholder="Email Address" id="Email" aria-
describedby="emailHelp" name="emailId">
            </div>
            <div class="form-group">
                <label class="text-white" for="Number">Mobile Number</label>
                <input type="number" class="form-control" placeholder="Mobile Number" id="Number"
name="Mobile_Number">
            </div>
            <div class="form-group">
                <label class="text-white" for="DOB">Date Of Birth</label>
                <input type="date" class="form-control" placeholder="Date of Birth" id="DOB" name="DOB">
            </div>
        </form>
    </div>

```

```

</div>
<div class="form-group">
  <label class="text-white" for="Password">Password</label>
  <input type="password" class="form-control" placeholder="Password" id="Password"
name="password">
</div>

  <button type="submit" class="btn btn-success">Submit</button>
</div>
{% endblock %}

```

### **import.html:**

```

{% extends 'base.html' %}
{% block content %}
  <div class="container">
    <h1 class="text-white">Import page</h1>
    <form action="browseFiles" : method="POST">
      <div class="form-group">
        <button type="submit" class="btn btn-success">Import</button>
      </div>
    </form>
  </div>
{% endblock %}

```

### **report.html:**

```

{% extends 'base.html' %}
{% block content %}
  <div class="container">
    <h1 class="text-white">Report</h1>
    <br>
    <br>
    <form action="Monthly_Finance" : method="POST">
      <div class="form-group">
        <button type="submit" class="btn btn-success">Monthly
Income/Expense/Saving</button>
      </div>
    </form>
  </div>

```

```

        </form>
        <br>
        <form action="Yearly_Finance" : method="POST">
        <div class="form-group">
            <button type="submit" class="btn btn-success">Yearly
Income/Expense/Saving</button>
        </div>
        </form>
        <br>
        <form action="CurrentNetWorth" : method="POST">
        <div class="form-group">
            <button type="submit" class="btn btn-success">Current Net Worth</button>
        </div>
        </form>
        <br>
        <form action="FinanceFreedom" : method="POST">
        <div class="form-group">
            <button type="submit" class="btn btn-success">Financial Freedom</button>
        </div>
        </form>
        <br>
        <form action="RetirementPlanning" : method="POST">
        <div class="form-group">
            <button type="submit" class="btn btn-success">Retirement Planning</button>
        </div>
        </form>
    </div>
{% endblock %}

```

### **monthly\_Finance.html:**

```

{% extends 'base.html' %}
{% block content %}
    <div class="container" style="color:#66FCF1">
    <h1 >Monthly Finance</h1>
    <form action="Final" method="POST">
        <div class="form-group">
        <br>

```

```

        <h4 class="text-white" style="font-family:bahnschrift semibold">Month</h4>
        <input type="number" class="form-control" placeholder="MM" name="Month">
    </div>
    <div class="form-group">
<br>
        <h4 class="text-white" style="font-family:bahnschrift semibold">Year</h4>
        <input type="number" class="form-control" placeholder="YYYY" name="Year">
    </div>
    <form action="Final" : method="POST">
        <div class="form-group">
<br>
            <button type="submit" class="btn btn-success" >Submit</button>
        </div>
    </form>
</div>
{% endblock %}

```

### **currentNetworth.html:**

```

{% extends 'base.html' %}
{% block content %}
    <div class="container text-white">
        <h1>Current Net Worth </h1>
        <form action="CurrentFinal" method="POST">
    <div class="form-group">
        <label class="text-white ">Expected Equity Average Return </label>
        <input type="number" class="form-control" name="AvgReturn">
    </div>
    <div class="form-group">
        <label class="text-white ">Expected Fixed Income Average Interest</label>
        <input type="number" class="form-control" name="AvgIntrest">
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
    </div>
{% endblock %}

```

### **financeFreedom.html:**

```
{% extends 'base.html' %}
{% block content %}
    <div class="container text-white">
        <h1>Financial Freedom </h1>
        <h2> Financial Freedom is a simple thumb rule. A Person is said to achieve Financial Freedom
when their networth can take care of atleast 25 years of their latest annual expense.
        </h2>
        <br>
        <br>
        <form action="FinanceFinal" method="POST">
<div class="form-group">
    <label class="text-white ">Expected Equity Average Return </label>
    <input type="number" class="form-control" name="AvgReturnFF">
</div>
<div class="form-group">
    <label class="text-white ">Expected Fixed Income Average Interest</label>
    <input type="number" class="form-control" name="AvgIntrestFF">
</div>
<div class="form-group">
    <label class="text-white ">Expected Inflation</label>
    <input type="number" class="form-control" name="InflationFF">
</div>
<button type="submit" class="btn btn-primary">Submit</button>
    </div>
{% endblock %}
```

### **retirementPlanning.html:**

```
{% extends 'base.html' %}
{% block content %}
    <div class="container text-white">
        <h1>RETIREMENT PLANNING</h1>
        <br>
        <br>
        <form action="RetirementFinal" method="POST">
<div class="form-group">
```

```

    <label class="text-white ">Expected Equity Average Return </label>
    <input type="number" class="form-control" name="AvgReturnRP">
</div>
<div class="form-group">
    <label class="text-white ">Expected Fixed Income Average Interest</label>
    <input type="number" class="form-control" name="AvgIntrestRP">
</div>
<div class="form-group">
    <label class="text-white ">Expected Inflation</label>
    <input type="number" class="form-control" name="InflationRP">
</div>
<div class="form-group">
    <label class="text-white ">Retirement Age</label>
    <input type="number" class="form-control" name="RetirementAge">
</div>
<div class="form-group">
    <label class="text-white ">Life Expectancy</label>
    <input type="number" class="form-control" name="LifeExpectancy">
</div>
<button type="submit" class="btn btn-primary">Submit</button>
</div>
{% endblock %}

```

### **retirementFinal.html:**

```

{% extends 'base.html' %}
{% block content %}
<div style="background-color: black">
    <div class="container" style="color:#66FCF1" >
        <br>
        <h1 style=" font-family:bahnschrift "> {{y[0]}}</h1>
        <br>
        <table class="table table-bordered">
        <thead>
        <tr style='background-color:#45A29E'>
            <th class='text-white'>TYPE</th>
            <th class='text-white'>AMOUNT</th>
        </tr>

```



```

</thead>
<tbody style='background-color:#1F2833'>
  <tr>
    <td class='text-white'>FUTURE EQUITY NETWORTH</td>
    <td class='text-white'>₹ {{ k[0] }}</td>
  </tr>
  <tr>
    <td class='text-white'>FUTURE FIXED INCOME NETWORTH</td>
    <td class='text-white'>₹ {{ k[1] }}</td>
  </tr>
  <tr>
    <td class='text-white'>FUTURE NETWORTH</td>
    <td class='text-white'>₹ {{ k[2] }}</td>
  </tr>
</tbody>
</table>
<img src = "{{ url_for('static' , filename='graph5.png') }}" alt = "Graph5 Image" width = "660" height
= "462"/>
</div>
</div>
{% endblock %}

```

## PERSONALIZED CLIENT EXPERIENCE

Taking Care Of What's Important So You Can Focus On What Matters To You.

### FESPA

Financial Expense-Saving Planning App

Contact Us At

+91 98409 31317 | Fespa@gmail.com

## About page

Financial Planning Web Software that catalogues a person's income, expenses and investments, and provides projections for future investments and retirement simulations.

Financial income/expense/savings data has been tabulated for 2 people who started their respective careers in 2005 at the age of 23. Both had similar growth in their career but differed in the way they spent their money and saved. The 2 people are Savitri and Shyam.

Developed By:

Varsha Balaji

HOME ABOUT LOGIN

## Login

Email address

Password

[Submit](#)

[SignUp](#)

HOME ABOUT LOGIN


## Sign Up Page

Name

Email address

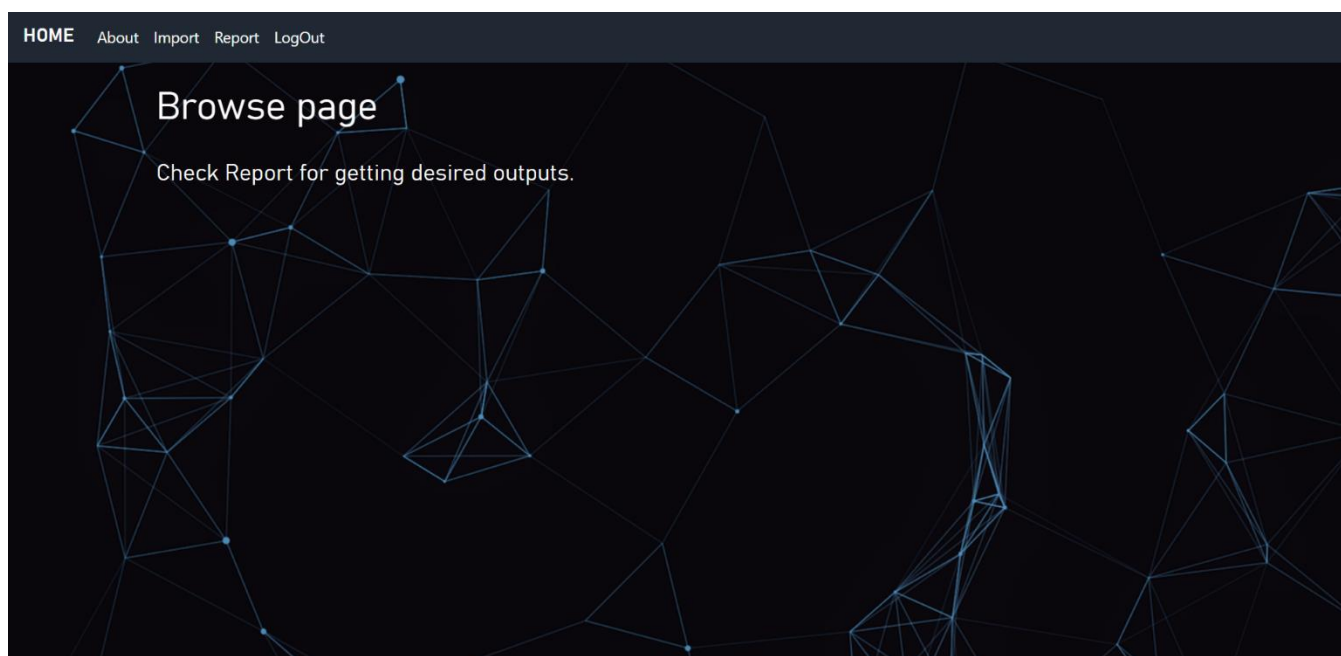
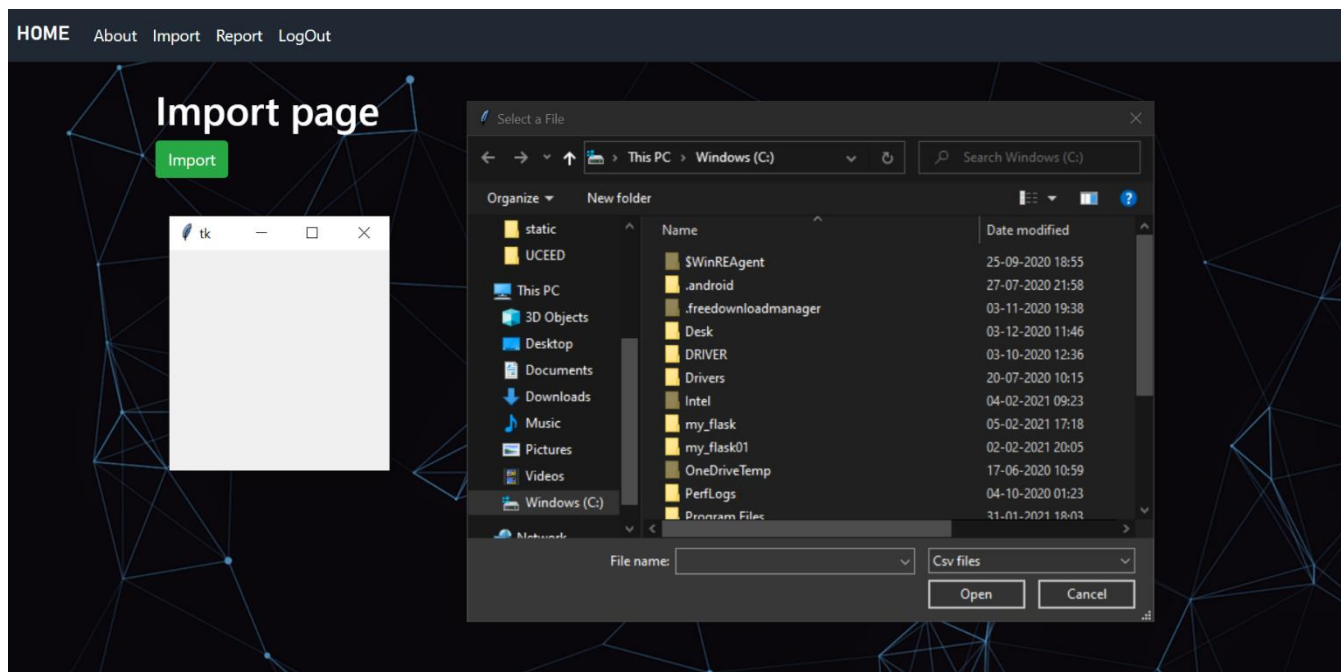
Mobile Number

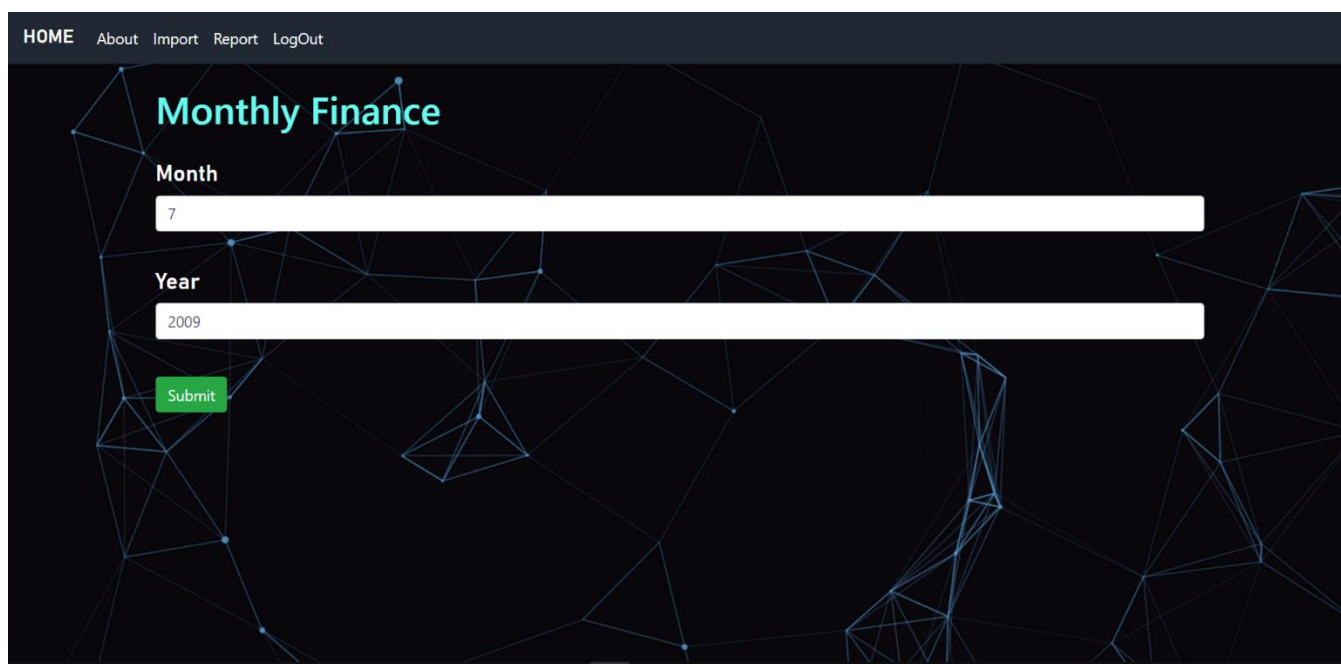
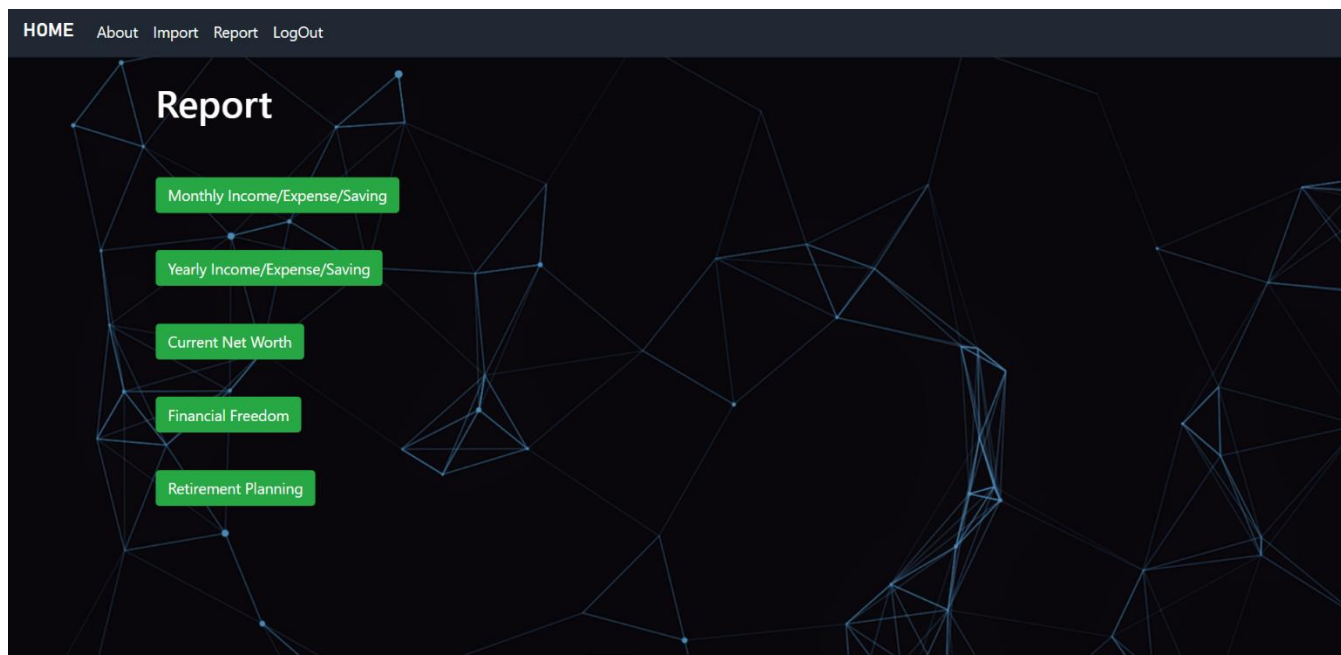
Date Of Birth

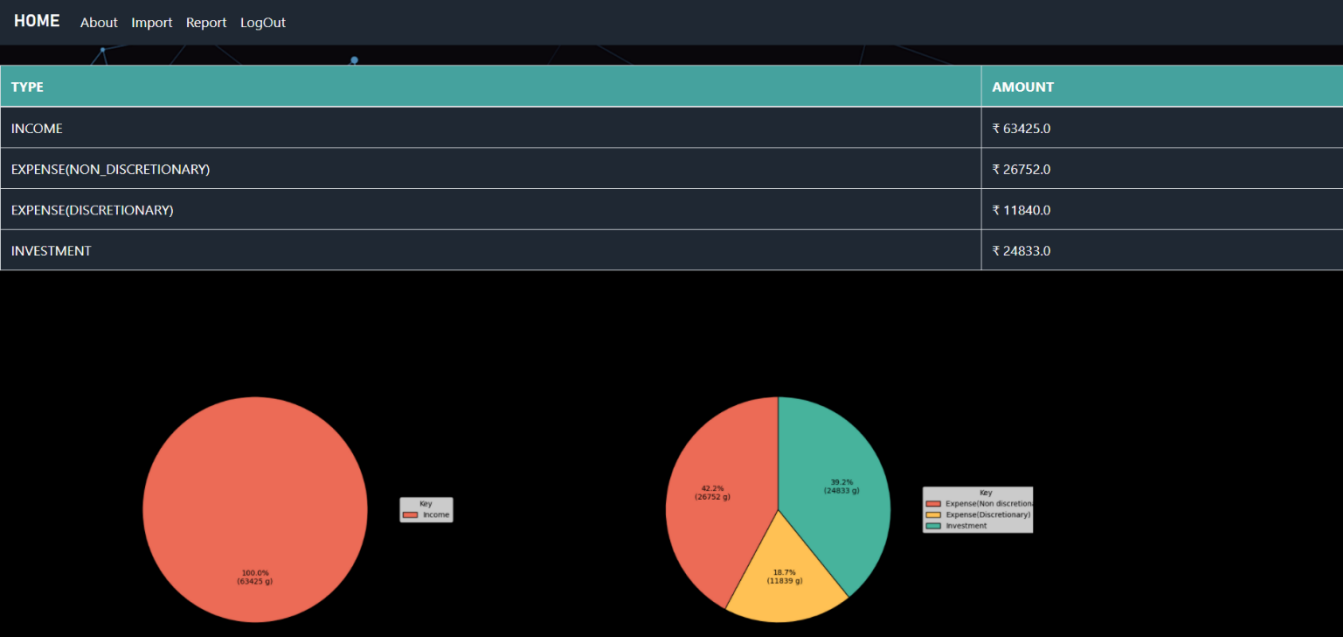


Password

[Submit](#)







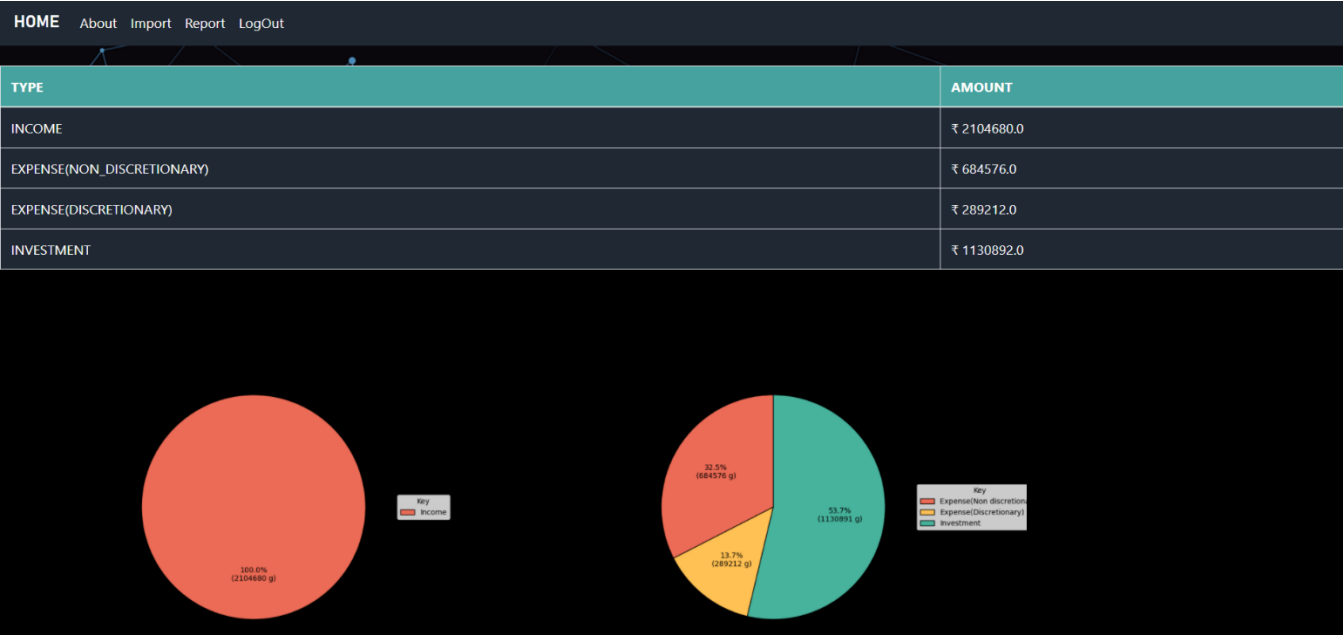
HOME About Import Report LogOut

Yearly Finance

Year

2020

Submit



HOME

About

Import

Report

LogOut

Current Net Worth

Expected Equity Average Return

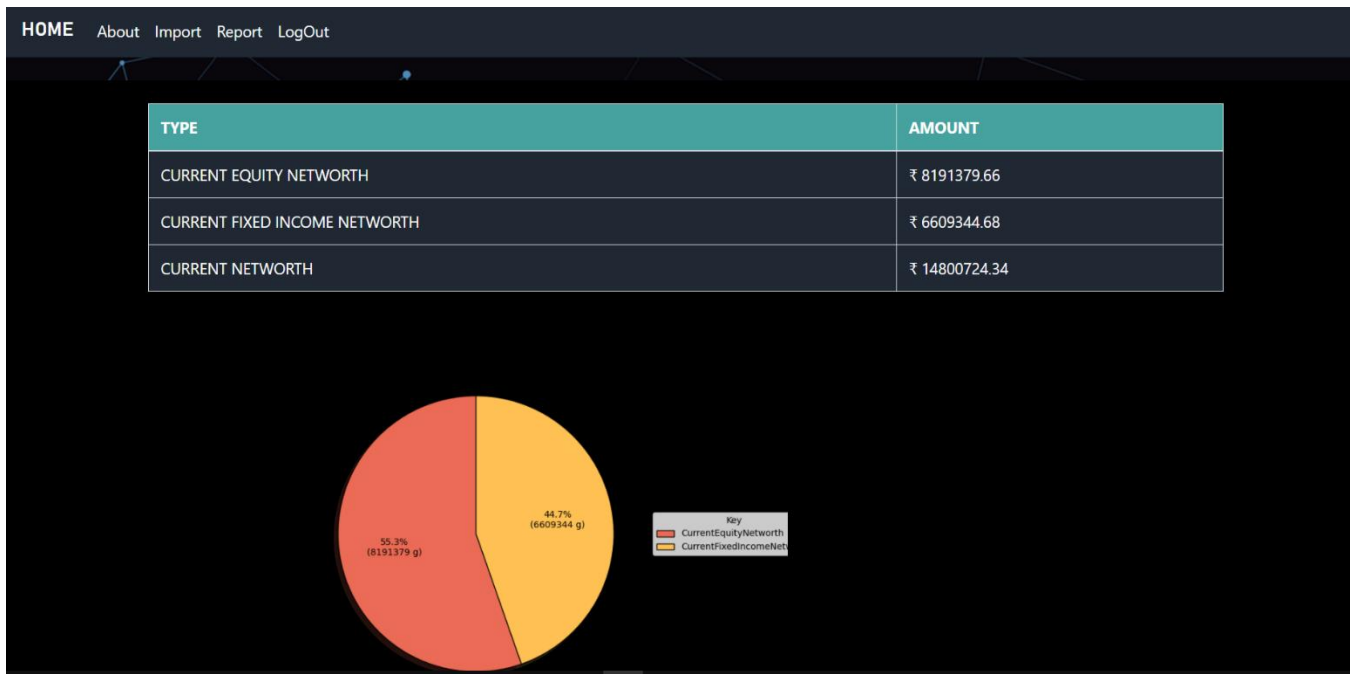
10

Expected Fixed Income Average Interest

7

Submit





HOME About Import Report LogOut

## Financial Freedom

Financial Freedom is a simple thumb rule. A Person is said to achieve Financial Freedom when their networkh can take care of atleast 25 years of their latest annual expense.

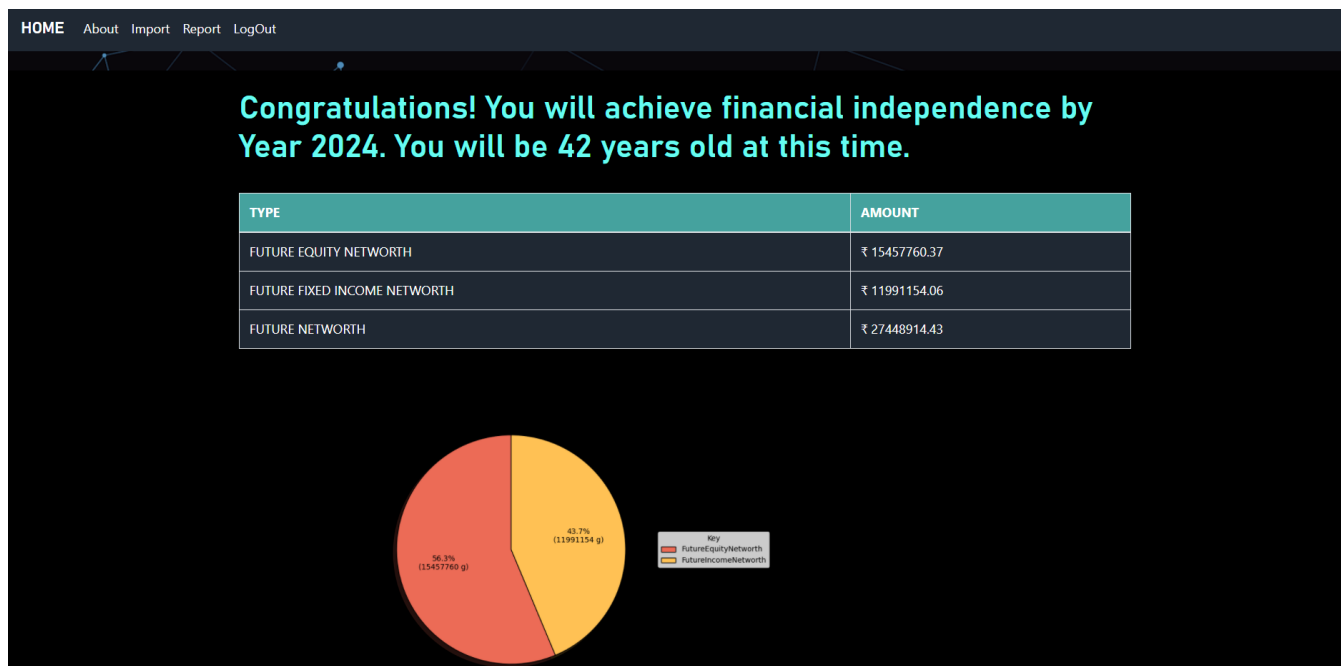
Expected Equity Average Return

Expected Fixed Income Average Interest

Expected Inflation

Submit





HOME About Import Report LogOut

## RETIREMENT PLANNING

Expected Equity Average Return

10

Expected Fixed Income Average Interest

7

Expected Inflation

2

Retirement Age

60

Life Expectancy

100

Submit

You can comfortably Retire at 60. You will have 211.0 Years of expenses covered.

TYPE	AMOUNT
FUTURE EQUITY NETWORK	₹ 192074499.51
FUTURE FIXED INCOME NETWORK	₹ 126429037.27
FUTURE NETWORK	₹ 318503536.78

