**https://codeshare.io/5PJzwP**

**Team 7**

Nathan Carney - function for all categories

Vedika Jajodia - expense calculator

Justin Koehn - visualisation

Darius Driggers - csv + email

Sam Song - ui, server

ISTM 3119

Team Project Proposal

Introduction:

Successful budgeting is vital for achieving optimal financial wellness. Creating a budget allows one to prioritize expenses and make a plan for every dollar earned within a given month. With a budget, an individual can focus on the things most important to them (i.e. saving for a home or vacation, paying off debt, or starting a business). A budget also prepares an individual to handle life’s emergencies confidently and without worry.

Objective:

The objective of our project it to make an ‘Expense Calculator’ of sorts with a built-in ‘Savings Simulator’ that helps a user in tracking their expenses and making a neat and visually appealing manner. This platform aims to ease the lives of people on a monthly budget keep a track of their expenses and be mindful of their savings!

Summary of Project:

Our team intends to create a budgeting application that allows users to input their expenses as they arise throughout the day. Our application will then determine if the user is spending above, below or at their means. At the end of each week, the user will also receive a graphic representation of their spending by category (Entertainment, Office Supplies, Transportation, Food/Drink, etc.) This will allow consumers to better track spending and have a visual representation of where a majority of their expenses lay. The user can also have a Savings amount set in the application. As the user punches in his/her expenses as they arise, the application will show him/her a warning if they exceed the limit.

Scope:

The Expense Calculator can be used as a financial planning and budgeting tool. The graphical breakdown of the expenses will allow the user to visually identify areas in which he or she may be overspending. The calculator will include a feature that will receive the users income as an input to create a budget for the user. Our expense calculator can also prevent overdraft fees as the user can visualize their total weekly expenditure and receive a warning message when they are spending too much. In addition, the user can easily compare their expenses to the amount they wish to save, an asset which will be useful in long-term financial planning. We plan to have basic categories of spending (listed above) and will also add a miscellaneous category for spending that does not fall into a specific category.

Limitations:

* The expense calculator will not have all possible expense categories.
* The expense calculator will need the user’s constant input of expenses to run properly.

import datetime

dt = datetime.datetime.today()

spending\_ledger={"Automotive & Gas":0,"Utilities":0,"Education":0,"Entertainment":0,"Food & Drink":0,

"Groceries":0,"Miscellanous":0,"Shopping":0,"Travel":0}

def updateAutomotiveandGas():

spending\_ledger["Automotive&Gas"]=+int(input("How much did you spend on Automotive&Gas?"))

def updateUtilities():

spending\_ledger["Utilities"]=+int(input("How much did you spend on Utilities?"))

def updateEducation():

spending\_ledger["Education"]=+int(input("How much did you spend on Education?"))

def updateEntertainment():

spending\_ledger["Entertainment"]=+int(input("How much did you spend on Entertainment?"))

def updateFoodandDrink():

spending\_ledger["Food & Drink"]=+int(input("How much did you spend on Food & Drink?"))

def updateGroceries():

spending\_ledger["Groceries"]=+int(input("How much did you spend on Groceries?"))

def updateMiscellanous():

spending\_ledger["Miscellanous"]=+int(input("How much did you spend on Miscellanois?"))

def updateShopping():

spending\_ledger["Shopping"]=+int(input("How much did you spend on Shopping?"))

def updateTravel():

spending\_ledger["Travel"]=+int(input("How much did you spend on Travel?"))

grand\_total=sum(spending\_ledger.values())

day\_month=dt.day

daily\_spending=grand\_total/day\_month

**Graph and Visualization Code:**

import matplotlib.pyplot as plt

import numpy as np

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#Variable Initialization

labels = ('Food', 'Travel', 'Entertainment', 'Auto','Other')

#percentage= [15, 23, 45, 10,7]

explode = (0, 0, 0, 0,0) # only "explode" the 2nd slice (i.e. 'Hogs')

expense = [10,8,6,4,7]

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**#Ploting pie chart**

fig1, ax1 = plt.subplots()

ax1.pie(expense, explode=explode, labels=labels, autopct='%1.0f%%',

shadow=True, startangle=90)

ax1.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show(fig1)

#Ploting bar chart here

y\_pos = np.arange(len(labels))

plt.bar(y\_pos, expense, align='center', alpha=0.5)

plt.xticks(y\_pos, labels)

plt.ylabel('Dollars Spent')

plt.title('Expenses')

plt.show()

import smtplib, ssl

port=465

smtp\_server="smtp.gmail.com"

sender\_email="istm3119@gmail.com"

receiver\_email="ddriggers@gwu.edu"

password= input("Type your password and press enter: ")

message="""\

Subject: Spending Update

Your have exceeded your budgeted spending."""

context=ssl.create\_default\_context()

with smtplib.SMTP\_SSL(smtp\_server, port, context=context) as server:

server.login(sender\_email, password)

server.sendmail(sender\_email,receiver\_email, message)

**Ledger code -**

expenseAmount = 100; #set input expense amount here, per transaction

income = 500; #set input income value here

name = 'vedika'; #set input name value here

category = "Shopping"

bankrupt = False

class Ledger:

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

self.bankrupt = bankrupt;

self.expense = 0.0;

self.income = income

self.name = name;

self.surplus = self.income - self.expense;

self.spending\_categories ={"Automotive & Gas":0.0,"Utilities":0.0,"Education":0.0,

"Entertainment":0.0,"Food & Drink":0.0,

"Groceries":0.0,"Miscellanous":0.0,"Shopping":0.0,"Travel":0.0}

return;

def credit(self, amount): # credited amount

self.income += amount;

self.surplus = self.income - self.expense;

return self.expense, self.surplus;

def debit(self, amount, category): #amount of one debit transaction, category of transaction

if (amount > self.surplus):

print("You have exceeded your monthly income")

bankrupt = True;

else:

self.expense +=amount;

self.surplus = self.income - self.expense;

if(category in self.spending\_categories):

self.spending\_categories[category] += amount;

else:

self.spending\_categories[category] = amount

return self.expense, self.surplus;

def getExpense(self):

string = "" + self.name + "'s Expenses are : " + str(self.expense)

return string

def getIncome(self):

string = "" + self.name + "'s Monthly Income is : " + str(self.income)

return string;

def getSurplus(self):

string = "" + self.name + "'s Remaining Surplus is : " + str(self.surplus)

return string;

def getCategoryExpense(self, category):

string = "" + self.name + "'s spending for " + category+ " is " + str(self.spending\_categories[category])

return string

def getCategoryDict(self):

return self.spending\_categories

vedika = Ledger(name);

print(vedika.getIncome())

print(vedika.getExpense())

print(vedika.getSurplus())

vedika.debit(expenseAmount, category)

print(vedika.getCategoryExpense("Entertainment"))

#print(vedika.getCategoryExpense("Shopping"))

vedika.getCategoryDict()

print(vedika.income)

print(vedika.surplus)

print(vedika.expense)

Final Report Paper Outline

Subject and Purpose: Nate

Problem Analysis: Nate

General Feature Overview: Everyone writes on about their own feature and describes their code (2 pages per person)

Challenges: Darius

References: add sources