PERSONAL EXPENSE TRACKER

Using python flask

Developed by: Sanjana G P

VMware Tanzu Build-A-Thon

1. INTRODUCTION

In today's busy and expensive life, we are in a great rush to make money. But at the end of the month, we broke off. As we are unknowingly spending money on little and unwanted things. So, we have come over with the idea to track our earnings. Personal Expense Tracker (PET) aims to help everyone who are planning to know their expenses and save from it. PET is an website which users can execute in their mobile phones and update their daily expenses so that they are well known to their expenses. Here user can define their own categories for expense type like food, clothing, rent and bills where they have to enter the money that has been spent and also can add some information in additional information to specify the expense. Although this website is focused on new job holders, interns and teenagers, everyone who wants to track their expense can use this app. Personal Expense Tracker System is intended to monitor Income-Expense of a person on an everyday premise. This System separates the Income-based daily expenses.

1.1 Overview

Personal Expense Tracker System is designed to keep a track of Income-Expense of a person on a day-to-day basis. This System divides the Income based on daily expenses. If you exceed day's expense, system will cut it from your income and will provide new daily expense allowed amount. It will let you add the savings amount, which you had saved for some particular days. This System takes Income from person and divides in daily expense allowed. If u exceed that day's expense it will cut if from your income and gives an alert mail for the registered person. Once we start off by tracking our expenses each day, we will be able to get a better idea where you are spending your money, so you stay in control and achieve your goal.

1.2 Purpose

Personal Expense Tracker (PET) aims to help everyone who are planning to know their expenses and save from it. PET is a website in which users can execute in their laptop and update their daily expenses so that they are well known to their expenses. Here user can define their own categories for expense type like food, clothing, rent and bills where they have to enter the money that has been spent and also can add some information in additional information to specify the expense. Although this app is focused on new job holders, interns and teenagers, everyone who wants to track their expense can use this app.

2. LITERATURE SURVEY

Tracking daily expense is not so innovative. Many traditional and technological approach is found to track our expenses and budget with their own functionality. From decades ago, and today we have been writing our expenditure in a register to calculate the profit or saving. Not only this many desktop and mobile applications has been developed for this purpose. Quicken and Microsoft money where the first desktop applications was developed decades ago but was not so familiar with the users. Personal capital and dollar bird application were used to visualize the expenses in chart or graphs with the calendar system. QuickBooks were the application for the small business holder to wrap up their whole business. YNAB and Penny were the latest application which were embedded with AI and applicable for importing expenses automatically. However, Mint was the one which was widely used and trusted. Explaining about the latest application built in this category, YNAB is an expense tracker that gives the automatic tracking of our expense through our bank account or credit cards.

2.1 Existing problem

In existing, we need to maintain the Excel sheets, CSV etc. files for the user daily and monthly expenses. In existing, there is no as such complete solution to keep a track of its daily expenditure easily. To do so a person as to keep a log in a diary or in a computer, also all the calculations need to be done by the user which may sometimes results in errors leading to losses.

2.2 Proposed solution

To reduce manual calculations, we propose an application which is developed by Python flask. This application allows users to maintain a digital automated diary. Each user will be required to register on the system at any time, the user will be provided id, which will be used to maintain the record of each unique user. Personal Expense Tracker application which will keep a track of Income-Expense of a user on a day-to-day basis. This application takes Income from user and divides in daily expense allowed. If u exceed that day's expense it will cut if from your income and give alert mail to the registered user.
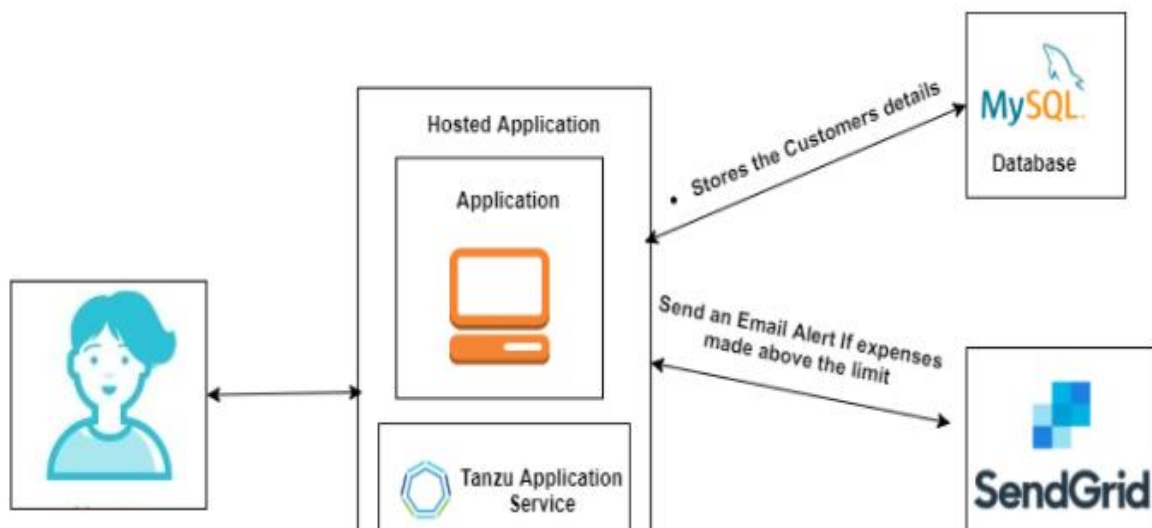
3. THEORITICAL ANALYSIS

3.1 Block Diagram

Fig: Block Diagram

3.2 Hardware / Software Designing

3.2.1 Hardware Requirements

- Processor: Intel® Core™ i3-7020 CPU @ 2.30GHz
- RAM: 4GB
- Hard Disk: 1TB

3.2.2 Software Requirements

- Operating System: Windows 10
- Language: Python
- Database: Remote MySQL
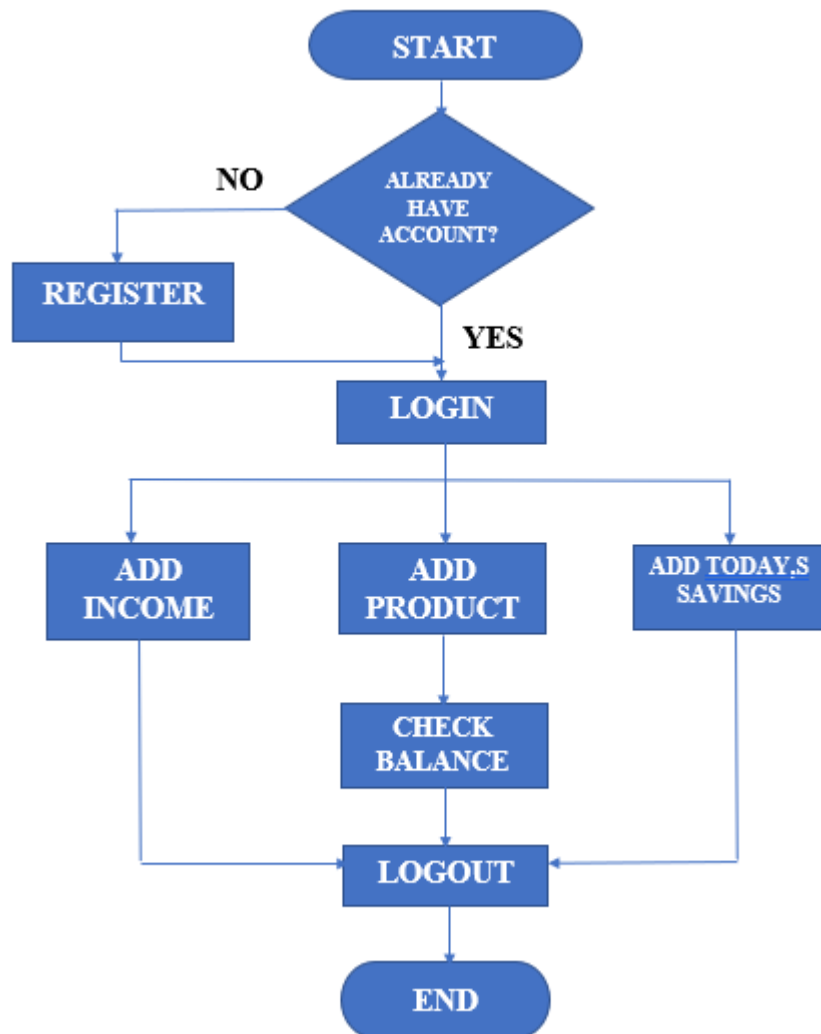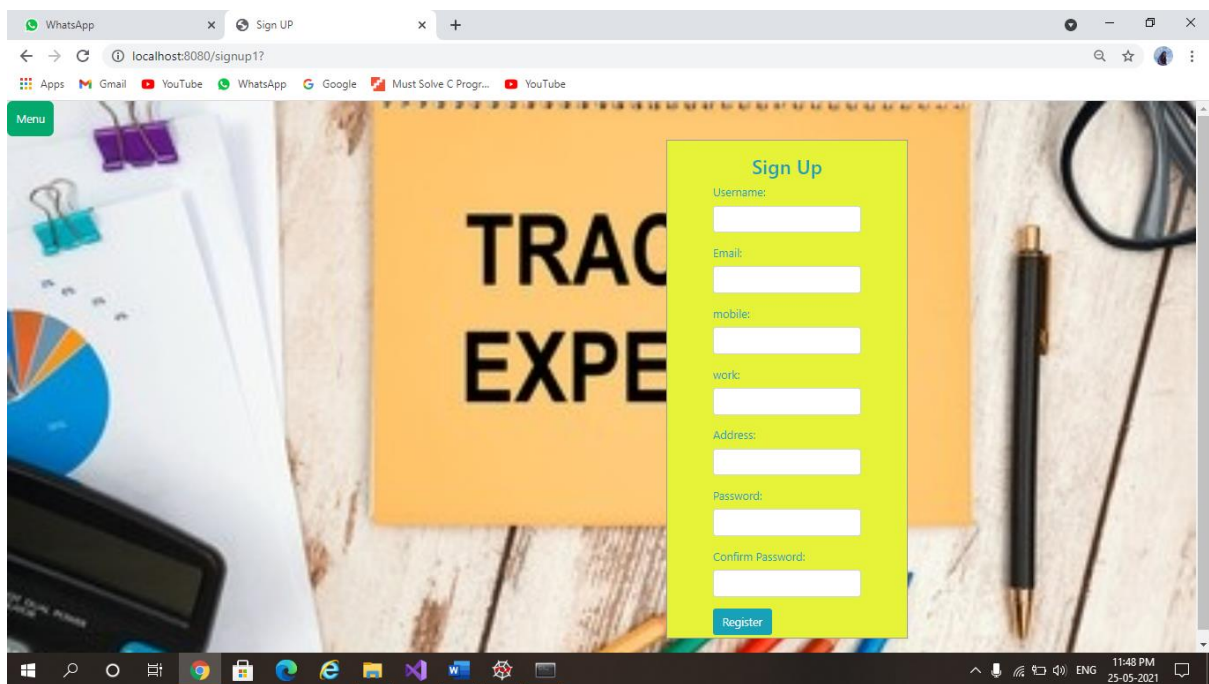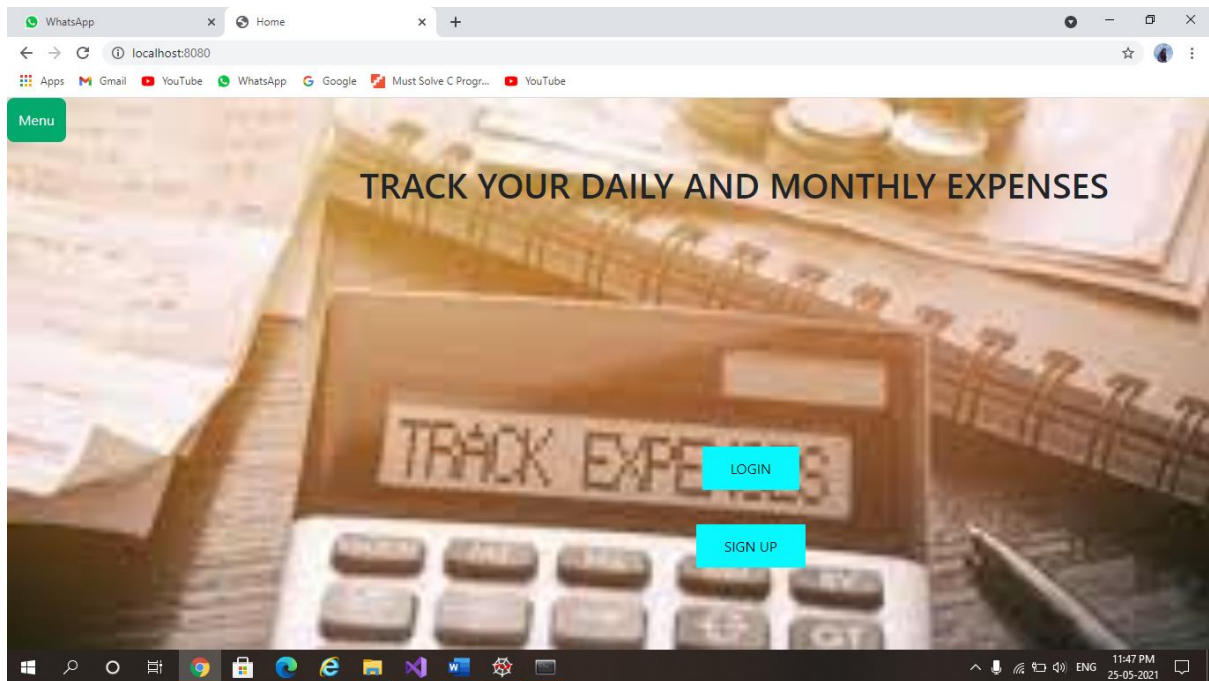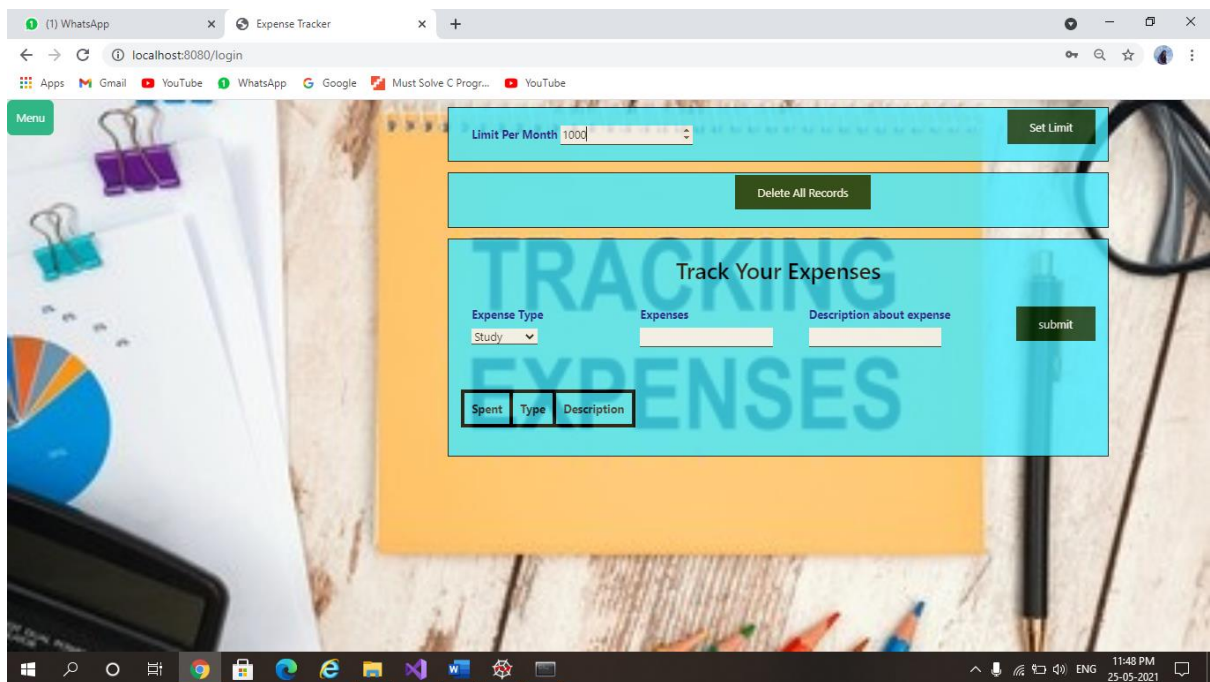- IDE: Spyder
- Browser: Google Chrome

4. FLOWCHART

Fig: Flowchart

5. RESULT

Menu

Limit Per Month [_____] Set Limit

Delete All Records

Track Your Expenses

Expense Type	Expenses	Description about expense
Shopping ▾	200	dress

submit

Total Amount : 500.0 Rs

Spent	Type	Description
500Rs	study	study material

Gmail

Compose

- Inbox 10
- Starred
- Snoozed
- Sent
- Drafts
- More

Meet
- New meeting
- Join a meeting

Hangouts
- dummy

No recent chats
Start a new one

1 of 12

Expenses Reminder Inbox ×

dummy3843@gmail.com   11:49 PM (1 minute ago)
to me

This Email is To Intemiate You that your expenses raises its limits. That is your prefeered limits are 1000. But the Total amount spend was 1300.0 Thank you
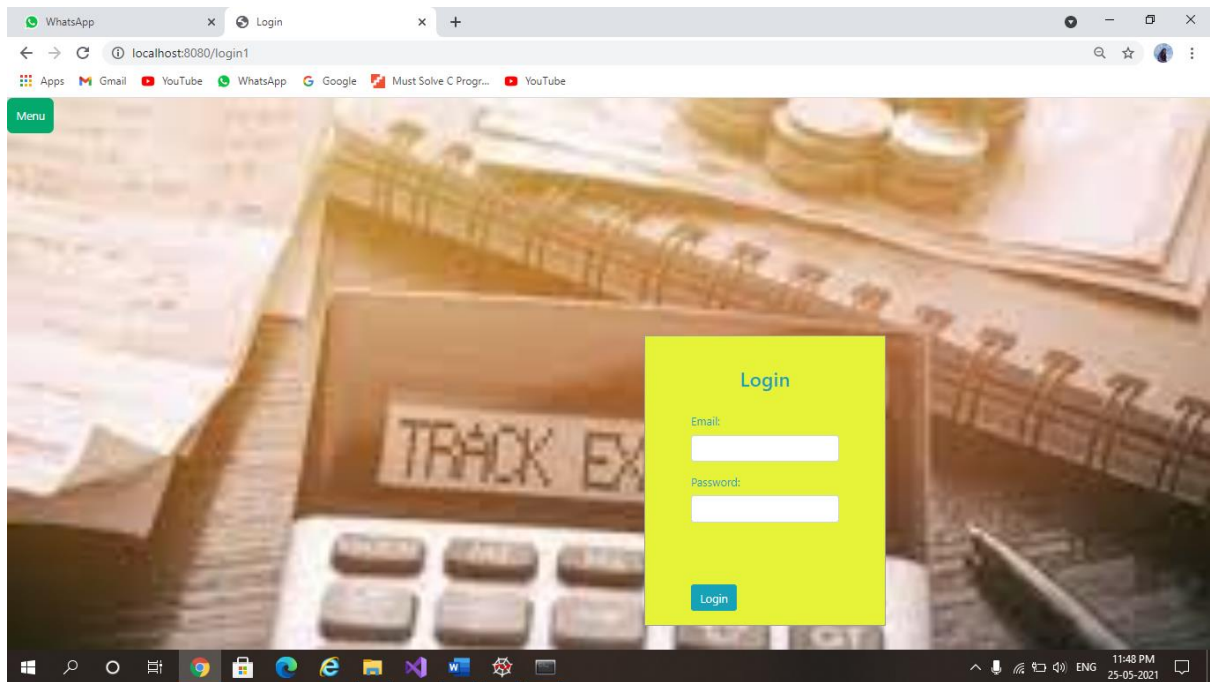
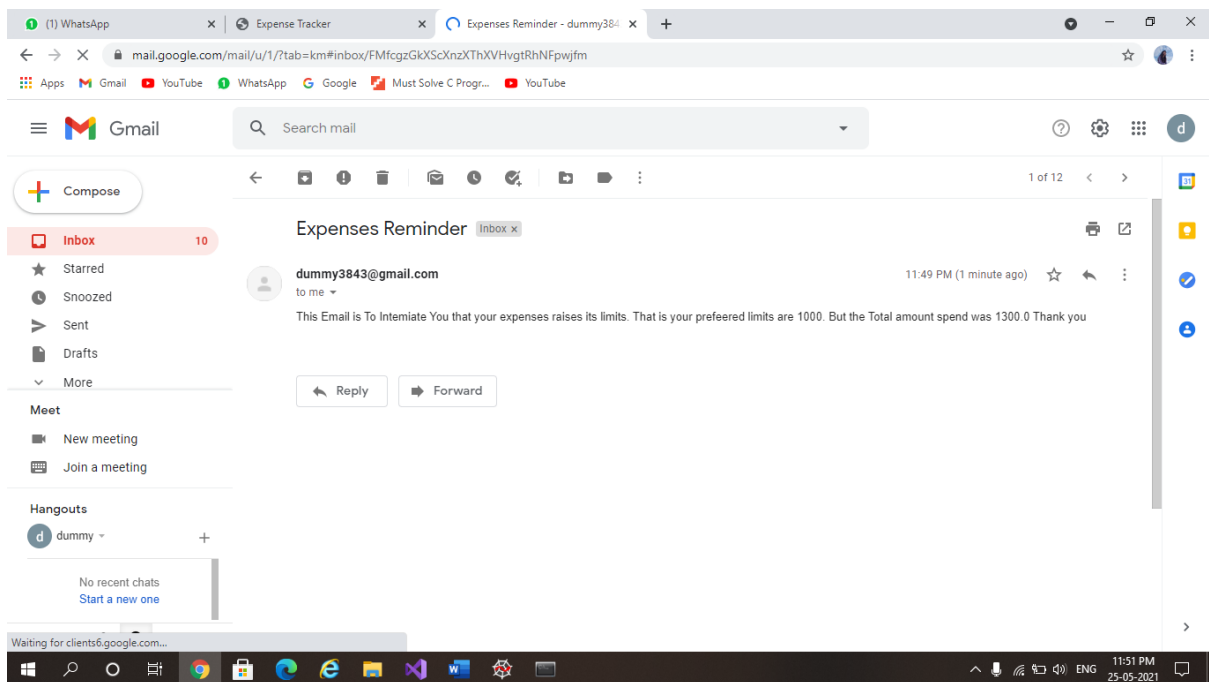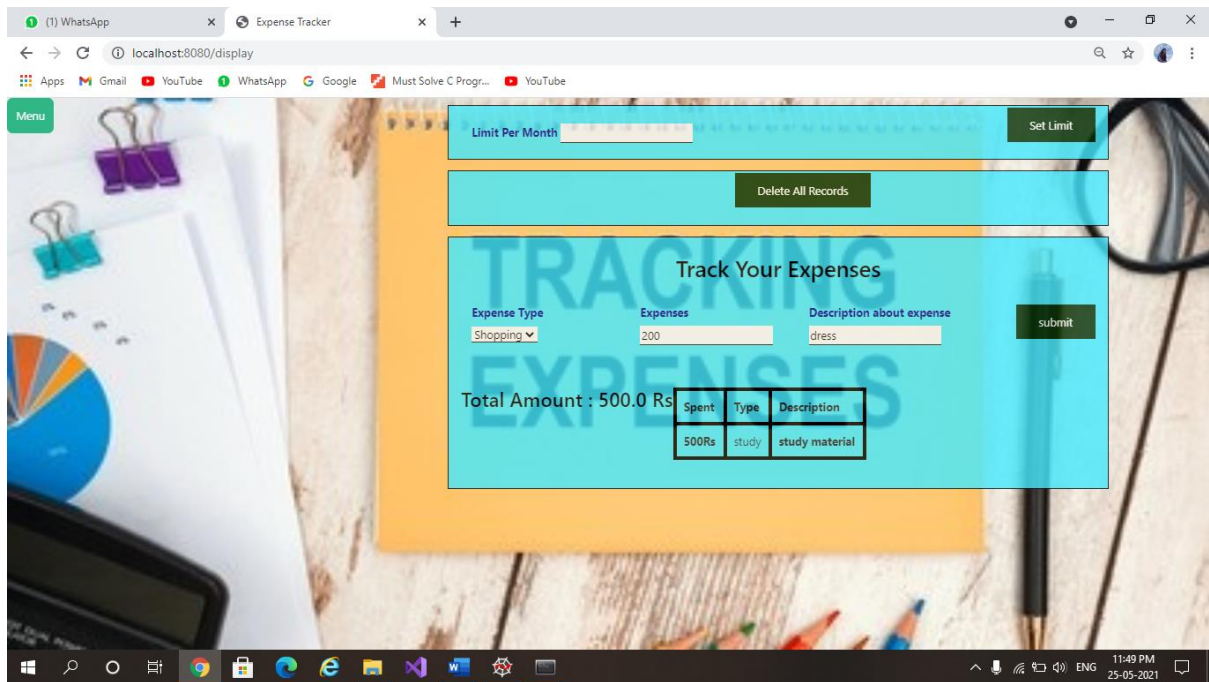↩ Reply   ➡ Forward

6. ADVANTAGES AND DISADVANTAGES

6.1 Advantages

- This system helps a person to reduce their expenses.
- Easy to use.
- You'll have better insight into your spending habits.
- It provides a better overview and comprehensive analysis.
- Saving and Investment.

6.2 Disadvantages

- May get inaccurate results if data is not inserted in correct manner.
- User have to entry every record manually.
- Person who is handling system must have some technical knowledge.

7. CONCLUSION

After making this application we assure that this application will help its users to manage the cost of their daily expenditure. It will guide them and aware them about their daily expenses. It will prove to be helpful for the people who are frustrated with their daily budget management, irritated because of amount of expenses and wishes to manage money and to preserve the record of their daily cost which may be useful to change their way of spending money. In short, this application will help its users to overcome the wastage of money.

8. FUTURE SCOPE

In further days, there will be calendar based reports and pay mode embedded with the app. Also, backup details will be recorded on cloud. Not only can you track your expenses, you can categorise them too. So, you know exactly how much of money you are spending under each of these categories.

9. BIBLIOGRAPHY

- https://www.geeksforgeeks.org/login-and-registration-project-using-flask-and-mysql/
- https://www.w3schools.com/
- https://html.com
- https://www.w3schools.com/icons/fontawesome5_icons_finance.asp
- https://www.filemakr.com/btech-final-year-project-report-daily-expense-tracker

APPENDIX

A. Source Code

```python
from flask import Flask , render_template ,request,session,redirect,url_for
from flask_mysqldb import MySQL
from flask_mail import Mail,Message


app = Flask(__name__)


app.config['MYSQL_HOST'] = "remotemysql.com"
app.config['MYSQL_USER'] = 's3jotogZcF'
app.config['MYSQL_PASSWORD'] = '#password'
app.config['MYSQL_DB'] = 's3jotogZcF'


app.config["MAIL_SERVER"] = 'smtp.gmail.com'
app.config['MAIL_PORT']=465
app.config['MAIL_USERNAME']="#email"
app.config['MAIL_PASSWORD']=  "#password"
app.config["MAIL_USE_TLS"]= False
app.config["MAIL_USE_SSL"]=True
mail1 = Mail(app)

app.secret_key = 'a'
mysql = MySQL(app)
```

```python
@app.route('/')
def home():
    return render_template("home.html")


@app.route('/signup1')
def signup1():
    return render_template("sign_up.html")


@app.route('/login1',methods=['POST','GET'])
def login1():
    return render_template("login.html")


@app.route('/signup',methods=['POST'])
def signup():
    msg = ''
    if(request.method=="POST"):
        name  =request.form["username"]
        email  =request.form["email"]
        work = request.form["work"]
        mobile = request.form["mobile"]
        address = request.form["address"]
        pass1 = request.form["password1"]
        pass2 = request.form["password2"]
        a1 = [name,email,mobile,work,address]
        cursor = mysql.connection.cursor()
        cursor.execute('SELECT * FROM users WHERE email= %s', (email,))
        email_info = cursor.fetchone()

        if(email_info!=None):
            msg="Email Already Exists"
        elif(pass1!=pass2):
            msg="Password doesn't Match"
        else:
```

```python
        cursor.execute('INSERT INTO users VALUES (NULL, % s, % s, %s, % s, %
s,%s)', (str(name), str(email),str(mobile),str(work),str(address),str(pass1)))
        msg = 'You have successfully registered !'
    mysql.connection.commit()
  return render_template('sign_up.html', msg = msg)




@app.route('/login',methods=['POST'])
def login():
  msg = ''
  if(request.method=="POST"):
    email  =request.form["email"]
    session["email"]=email
    password = request.form["password"]
    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM users WHERE email= %s', (str(email),))
    email_info = cursor.fetchone()
    cursor1 = mysql.connection.cursor()
    cursor1.execute('SELECT * FROM users WHERE email= %s AND pass1= %s',
[str(email),str(password)])
    email_pass_info = cursor1.fetchone()
    mysql.connection.commit()
    if(email_info==None):
      msg="Email dosen't Exists"
      return render_template('login.html',msg1=msg)
    elif(email_pass_info==None):
      msg="Password is incorrect"
      return render_template('login.html',msg1=msg)
    else:
      cursor2 = mysql.connection.cursor()
      cursor4 = mysql.connection.cursor()
      cursor5 = mysql.connection.cursor()
      print(session['email'])
```

```python
        cursor2.execute('SELECT * FROM exp WHERE email =
%s',[session['email']])
        cursor4.execute('SELECT sum(exp) FROM exp  WHERE email =
%s',[session['email']])
        cursor5.execute('SELECT  * FROM limits  WHERE email =
%s',[session['email']])
        amount = cursor4.fetchall()
        account = cursor2.fetchall()
        limits1 =cursor5.fetchall()
        flag=1
        if(None in amount[0]):
            print("yes--------------------------------------------------yes")
            flag=0
        print("Limits: Amount :",limits1,len(amount),amount,account)


        if(None in amount[0]):
            return render_template('main.html')
        print("Limits: Amount :",limits1,len(amount),amount,account)


        if(len(limits1)==0):
            print("------------------------------------------------------------------------------
------------------------")
            print("account : ",account)
            if(account!=None or flag):
                return render_template('main.html',pred = account,amount="Total
Amount : "+str(*amount[0])+" Rs")
            else:
                return render_template('main.html')
        elif(float(limits1[-1][-1])<float(amount[0][0])):
            message = Message("Expenses
Reminder",sender="#Email",recipients=[session['email']])
            message.body = "This Email is To Intemiate You that your expenses raises
its limits. That is your prefeered limits are "+str(limits1[-1][-1])+". But the Total
amount spend was "+str(amount[0][0])+" Thank you"
```

```python
            mail1.send(message)

        mysql.connection.commit()
        # print("accountdislay",account)
        # session.__exit__
        if(account==None or amount[0][0]==None):
            return render_template('main.html')
        return render_template('main.html',pred = account,amount="Total Amount :
"+str(*amount[0])+" Rs")




@app.route('/display',methods=['POST','GET'])
def display():
    if(request.method=='POST'):
        type1 = request.form["type"]
        exp = request.form["exp"]
        description = request.form["description"]


        cursor2 = mysql.connection.cursor()
        cursor3 = mysql.connection.cursor()
        cursor4 = mysql.connection.cursor()
        cursor5 = mysql.connection.cursor()
        print(session['email'])
        cursor2.execute('INSERT INTO exp VALUES(NULL,
%s,%s,%s,%s)',[session['email'],str(type1),str(exp),str(description)])
        mysql.connection.commit()
        cursor3.execute('SELECT * FROM exp WHERE email = %s',[session['email']])
        cursor4.execute('SELECT sum(exp) FROM exp  WHERE email =
%s',[session['email']])
        cursor5.execute('SELECT * FROM limits  WHERE email =
%s',[session['email']])
        account = cursor3.fetchall()
        amount = cursor4.fetchall()
        limits1 = cursor5.fetchall()
```

```python
        # print(limits1[-1][-1],amount[0][0])
        flag=1
        if(None in amount[0]):
            print("yes")
            flag=0
        if(None in amount[0]):
            return render_template('main.html')
        if(len(limits1)==0):
          if(account!=None or flag):
            return render_template('main.html',pred = account,amount="Total Amount :
"+str(*amount[0])+" Rs")
          else:
            return render_template('main.html')
        elif(float(limits1[-1][-1]) < float(amount[0][0])):
            message = Message("Expenses
Reminder",sender="#Email",recipients=[session['email']])
            message.body = "This Email is To Intemiate You that your expenses raises its
limits. That is your prefeered limits are "+str(limits1[-1][-1])+". But the Total amount
spend was "+str(amount[0][0])+" Thank you"
            mail1.send(message)
        mysql.connection.commit()
        if(account==None or amount[0][0]==None):
          return render_template('main.html')
        return render_template('main.html',pred = account,amount="Total Amount :
"+str(*amount[0])+" Rs")


@app.route('/display1',methods=['post','Get'])
def display1():
  if(request.method=='POST'):
    limit = request.form["limit"]
    cursor2 = mysql.connection.cursor()
    print(limit)
    cursor2.execute('INSERT INTO limits VALUES(NULL,
%s,%s)',[session['email'],str(limit)])
```

```python
        mysql.connection.commit()
        cursor3 = mysql.connection.cursor()
        cursor4 = mysql.connection.cursor()
        cursor5 = mysql.connection.cursor()
        print(session['email'])
        cursor3.execute('SELECT * FROM exp WHERE email = %s',[session['email']])
        cursor4.execute('SELECT sum(exp) FROM exp  WHERE email =
%s',[session['email']])
        cursor5.execute('SELECT * FROM limits  WHERE email =
%s',[session['email']])
        account = cursor3.fetchall()
        amount = cursor4.fetchall()
        limits1 = cursor5.fetchall()
        mysql.connection.commit()
        if(None in amount[0]):
            print("yes")
            flag=0
        if(None in amount[0]):
            return render_template('main.html')
        if(len(limits1)==0):
          if(account!=None or flag):
            return render_template('main.html',pred = account,amount="Total Amount :
"+str(*amount[0])+" Rs")
          else:
            return render_template('main.html')
        elif(float(limits1[-1][-1])<float(amount[0][0])):
          message = Message("Expenses
Reminder",sender="#Email",recipients=[session['email']])
          message.body = "This Email is To Intemiate You that your expenses raises its
limits. That is your prefeered limits are "+str(limits1[-1][-1])+". But the Total amount
spend was "+str(amount[0][0])+" Thank you"
          mail1.send(message)
        mysql.connection.commit()
        if(account==None or amount[0][0]==None):
```

```python
        return render_template('main.html')
    return render_template('main.html',pred = account,amount="Total Amount :
"+str(*amount[0])+" Rs")



@app.route('/delete',methods=['post','Get'])
def delete():
    if(request.method=='POST'):
        cursor = mysql.connection.cursor()
        cursor.execute('DELETE  FROM exp WHERE email= %s',(session['email'],))
        mysql.connection.commit()
        return render_template('main.html')


@app.route('/logout',methods=['post','Get'])
def logout():
    session.__exit__
    return render_template("home.html")



if __name__=="__main__":
    app.run(host="0.0.0.0",debug=True,port=8080)
```