

SPRINT-3

PROJECT DEVELOPMENT PHASE

Team ID	PNT2022TMID01338
Project Name	PERSONAL EXPENSE TRACKER APPLICATION

app.py

```
from flask import Flask, render_template, request, redirect, session
import re

from base import DB2
from flask_db2 import DB2
import ibm_db
import ibm_db_dbi
from sendgrid import sendgridmail, sendmail

import os

app = Flask(__name__)

app.secret_key = 'a'

"""
dsn_hostname = "9938aec0-8105-433e-8bf9-
0fbb7e483086.clogj3sd0tgtu0lqde00.databases.appdomain.cloud"
dsn_uid = "vpq40294"
dsn_pwd = "2LZdL55bbMzhzKmr"
dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "bludb"
dsn_port = "32459"
dsn_protocol = "tcpip"

dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
).format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol,
dsn_uid, dsn_pwd)
"""
```

```

app.config['database'] = 'bludb'
app.config['hostname'] = '9938aec0-8105-433e-8bf9-
0fbb7e483086.clogj3sd0tgtu0lqde00.databases.appdomain.cloud'
app.config['port'] = '32459'
app.config['protocol'] = 'tcpip'
app.config['uid'] = 'vpq40294'
app.config['pwd'] = '2LZdL55bbMzhzKmr'
app.config['security'] = 'SSL'
try:
    mysql = DB2(app)

    conn_str='database=bludb;hostname=9938aec0-8105-433e-8bf9-
0fbb7e483086.clogj3sd0tgtu0lqde00.databases.appdomain.cloud;\
uid=vpq40294;pwd=2LZdL55bbMzhzKmr;security=SSL'
    ibm_db_conn = ibm_db.connect(conn_str, '', '')

    print("Database connected without any error !!")
except:
    print("IBM DB Connection error      :      " + DB2.conn_errormsg())

#HOME--PAGE
@app.route("/base")
def home():
    return render_template("base1.html")

@app.route("/")
def add():
    return render_template("home1.html")

#SIGN--UP--OR--REGISTER

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

@app.route('/register', methods =['GET', 'POST'])
def register():
    msg = ''
    print("Break point1")
    if request.method == 'POST' :

```

```

username = request.form['username']
email = request.form['email']
password = request.form['password']

print("Break point2" + "name: " + username + "-----" + email + "-----" + password)

try:
    print("Break point3")
    connectionID = ibm_db_dbi.connect(conn_str, '', '')
    cursor = connectionID.cursor()
    print("Break point4")
except:
    print("No connection Established")

print("Break point5")
sql = "SELECT * FROM register WHERE username = ?"
stmt = ibm_db.prepare(ibm_db_conn, sql)
ibm_db.bind_param(stmt, 1, username)
ibm_db.execute(stmt)
result = ibm_db.execute(stmt)
print(result)
account = ibm_db.fetch_row(stmt)
print(account)

param = "SELECT * FROM register WHERE username = " + "\"" + username + "\"

res = ibm_db.exec_immediate(ibm_db_conn, param)
print("---- ")
dictionary = ibm_db.fetch_assoc(res)
while dictionary != False:
    print("The ID is : ", dictionary["USERNAME"])
    dictionary = ibm_db.fetch_assoc(res)

print("break point 6")
if account:
    msg = 'Username already exists !'
elif not re.match(r'^@+@[^@]+\.[^@]+', email):
    msg = 'Invalid email address !'
elif not re.match(r'[A-Za-z0-9]+', username):
    msg = 'name must contain only characters and numbers !'
else:
    sql2 = "INSERT INTO register (username, email,password) VALUES (?, ?, ?)"
    stmt2 = ibm_db.prepare(ibm_db_conn, sql2)

```

```

        ibm_db.bind_param(stmt2, 1, username)
        ibm_db.bind_param(stmt2, 2, email)
        ibm_db.bind_param(stmt2, 3, password)
        ibm_db.execute(stmt2)
        msg = 'You have successfully registered !'
        return render_template('signup.html', msg = msg)

#LOGIN--PAGE

@app.route("/signin")
def signin():
    return render_template("login1.html")

@app.route('/login',methods =['GET', 'POST'])
def login():
    global userid
    msg = ''

    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']

        sql = "SELECT * FROM register WHERE username = ? and password = ?"
        stmt = ibm_db.prepare(ibm_db_conn, sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.bind_param(stmt, 2, password)
        result = ibm_db.execute(stmt)
        print(result)
        account = ibm_db.fetch_row(stmt)
        print(account)

        param = "SELECT * FROM register WHERE username = " + "\"" + username +
        "\"" + " and password = " + "\"" + password + "\""
        res = ibm_db.exec_immediate(ibm_db_conn, param)
        dictionary = ibm_db.fetch_assoc(res)

        # sendmail("hello","lokhitarajan@gmail.com")

    if account:
        session['loggedin'] = True
        session['id'] = dictionary["ID"]
        userid = dictionary["ID"]
        session['username'] = dictionary["USERNAME"]

```

```

        session['email'] = dictionary["EMAIL"]

        return redirect('/home1')
    else:
        msg = 'Incorrect username / password !'

    return render_template('login1.html', msg = msg)

#ADDING----DATA

@app.route("/add1")
def adding():
    return render_template('add1.html')

@app.route('/addexpense',methods=['GET', 'POST'])
def addexpense():

    date = request.form['date']
    expensename = request.form['expensename']
    amount = request.form['amount']
    paymode = request.form['paymode']
    category = request.form['category']

    print(date)
    p1 = date[0:10]
    p2 = date[11:13]
    p3 = date[14:]
    p4 = p1 + "-" + p2 + "." + p3 + ".00"
    print(p4)

    sql = "INSERT INTO expenses (userid, date, expensename, amount, paymode,
category) VALUES (?, ?, ?, ?, ?, ?)"
    stmt = ibm_db.prepare(ibm_db_conn, sql)
    ibm_db.bind_param(stmt, 1, session['id'])
    ibm_db.bind_param(stmt, 2, p4)
    ibm_db.bind_param(stmt, 3, expensename)
    ibm_db.bind_param(stmt, 4, amount)
    ibm_db.bind_param(stmt, 5, paymode)

```

```

ibm_db.bind_param(stmt, 6, category)
ibm_db.execute(stmt)

print("Expenses added")

# email part

param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) + "
AND MONTH(date) = MONTH(current timestamp) AND YEAR(date) = YEAR(current
timestamp) ORDER BY date DESC"
res = ibm_db.exec_immediate(ibm_db_conn, param)
dictionary = ibm_db.fetch_assoc(res)
expense = []
while dictionary != False:
    temp = []
    temp.append(dictionary["ID"])
    temp.append(dictionary["USERID"])
    temp.append(dictionary["DATE"])
    temp.append(dictionary["EXPENSENAME"])
    temp.append(dictionary["AMOUNT"])
    temp.append(dictionary["PAYMODE"])
    temp.append(dictionary["CATEGORY"])
    expense.append(temp)
    print(temp)
    dictionary = ibm_db.fetch_assoc(res)

total=0
for x in expense:
    total += x[4]

param = "SELECT id, limitss FROM limits WHERE userid = " +
str(session['id']) + " ORDER BY id DESC LIMIT 1"
res = ibm_db.exec_immediate(ibm_db_conn, param)
dictionary = ibm_db.fetch_assoc(res)
row = []
s = 0
while dictionary != False:
    temp = []
    temp.append(dictionary["LIMITSS"])
    row.append(temp)
    dictionary = ibm_db.fetch_assoc(res)
    s = temp[0]

if total > int(s):
    msg = "Hello " + session['username'] + " , " + "you have crossed the
monthly limit of Rs. " + s + "/- !!!" + "\n" + "Thank you, " + "\n" + "Team
Personal Expense Tracker."
    sendmail(msg,session['email'])

```

```

        return redirect("/display1")

#DISPLAY---graph

@app.route("/display1")
def display():
    print(session["username"],session['id'])

    param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) + "
ORDER BY date DESC"
    res = ibm_db.exec_immediate(ibm_db_conn, param)
    dictionary = ibm_db.fetch_assoc(res)
    expense = []
    while dictionary != False:
        temp = []
        temp.append(dictionary["ID"])
        temp.append(dictionary["USERID"])
        temp.append(dictionary["DATE"])
        temp.append(dictionary["EXPENSENAME"])
        temp.append(dictionary["AMOUNT"])
        temp.append(dictionary["PAYMODE"])
        temp.append(dictionary["CATEGORY"])
        expense.append(temp)
        print(temp)
        dictionary = ibm_db.fetch_assoc(res)

    return render_template('display1.html' ,expense = expense)

#delete---the--data

@app.route('/delete/<string:id>', methods = ['POST', 'GET' ])
def delete(id):

    param = "DELETE FROM expenses WHERE id = " + id
    res = ibm_db.exec_immediate(ibm_db_conn, param)

    print('deleted successfully')
    return redirect("/display1")

```

```
#UPDATE---DATA
```

```
@app.route('/edit/<id>', methods = ['POST', 'GET' ])  
def edit(id):
```

```
    param = "SELECT * FROM expenses WHERE id = " + id  
    res = ibm_db.exec_immediate(ibm_db_conn, param)  
    dictionary = ibm_db.fetch_assoc(res)  
    row = []  
    while dictionary != False:  
        temp = []  
        temp.append(dictionary["ID"])  
        temp.append(dictionary["USERID"])  
        temp.append(dictionary["DATE"])  
        temp.append(dictionary["EXPENSENAME"])  
        temp.append(dictionary["AMOUNT"])  
        temp.append(dictionary["PAYMODE"])  
        temp.append(dictionary["CATEGORY"])  
        row.append(temp)  
        print(temp)  
        dictionary = ibm_db.fetch_assoc(res)  
  
    print(row[0])  
    return render_template('edit.html', expenses = row[0])
```

```
@app.route('/update/<id>', methods = ['POST'])  
def update(id):
```

```
    if request.method == 'POST' :
```

```
        date = request.form['date']  
        expensename = request.form['expensename']  
        amount = request.form['amount']  
        paymode = request.form['paymode']  
        category = request.form['category']
```

```
        p1 = date[0:10]  
        p2 = date[11:13]  
        p3 = date[14:]  
        p4 = p1 + "-" + p2 + "." + p3 + ".00"
```

```
        sql = "UPDATE expenses SET date = ? , expensename = ? , amount = ?,  
paymode = ?, category = ? WHERE id = ?"  
        stmt = ibm_db.prepare(ibm_db_conn, sql)
```



```

        ibm_db.bind_param(stmt, 1, p4)
        ibm_db.bind_param(stmt, 2, expensename)
        ibm_db.bind_param(stmt, 3, amount)
        ibm_db.bind_param(stmt, 4, paymode)
        ibm_db.bind_param(stmt, 5, category)
        ibm_db.bind_param(stmt, 6, id)
        ibm_db.execute(stmt)

    print('successfully updated')
    return redirect("/display1")

#limit
@app.route("/limit1" )
def limit():
    return redirect('/limitn')

@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
    if request.method == "POST":
        number= request.form['number']
        # cursor = mysql.connection.cursor()
        # cursor.execute('INSERT INTO limits VALUES (NULL, % s, % s)
        ',(session['id'], number))
        # mysql.connection.commit()

        sql = "INSERT INTO limits (userid, limitss) VALUES (?, ?)"
        stmt = ibm_db.prepare(ibm_db_conn, sql)
        ibm_db.bind_param(stmt, 1, session['id'])
        ibm_db.bind_param(stmt, 2, number)
        ibm_db.execute(stmt)

        return redirect('/limitn')

@app.route("/limitn")
def limitn():
    # cursor = mysql.connection.cursor()
    # cursor.execute('SELECT limitss FROM `limits` ORDER BY `limits`.`id` DESC
    LIMIT 1')
    # x= cursor.fetchone()
    # s = x[0]

```

```

        param = "SELECT id, limitss FROM limits WHERE userid = " +
str(session['id']) + " ORDER BY id DESC LIMIT 1"
        res = ibm_db.exec_immediate(ibm_db_conn, param)
        dictionary = ibm_db.fetch_assoc(res)
        row = []
        s = " /-"
        while dictionary != False:
            temp = []
            temp.append(dictionary["LIMITSS"])
            row.append(temp)
            dictionary = ibm_db.fetch_assoc(res)
            s = temp[0]

        return render_template("limit1.html" , y= s)

#REPORT

@app.route("/today1")
def today():

    param1 = "SELECT TIME(date) as tn, amount FROM expenses WHERE userid = "
+ str(session['id']) + " AND DATE(date) = DATE(current timestamp) ORDER BY
date DESC"
    res1 = ibm_db.exec_immediate(ibm_db_conn, param1)
    dictionary1 = ibm_db.fetch_assoc(res1)
    texpanse = []

    while dictionary1 != False:
        temp = []
        temp.append(dictionary1["TN"])
        temp.append(dictionary1["AMOUNT"])
        texpanse.append(temp)
        print(temp)
        dictionary1 = ibm_db.fetch_assoc(res1)


    param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) +
" AND DATE(date) = DATE(current timestamp) ORDER BY date DESC"
    res = ibm_db.exec_immediate(ibm_db_conn, param)
    dictionary = ibm_db.fetch_assoc(res)
    expense = []
    while dictionary != False:
        temp = []
        temp.append(dictionary["ID"])
        temp.append(dictionary["USERID"])
        temp.append(dictionary["DATE"])

```

```
temp.append(dictionary["EXPENSENAME"])
temp.append(dictionary["AMOUNT"])
temp.append(dictionary["PAYMODE"])
temp.append(dictionary["CATEGORY"])
expense.append(temp)
print(temp)
dictionary = ibm_db.fetch_assoc(res)

total=0
t_food=0
t_entertainment=0
t_business=0
t_rent=0
t_EMI=0
t_other=0

for x in expense:
    total += x[4]
    if x[6] == "food":
        t_food += x[4]

    elif x[6] == "entertainment":
        t_entertainment += x[4]

    elif x[6] == "business":
        t_business += x[4]
    elif x[6] == "rent":
        t_rent += x[4]

    elif x[6] == "EMI":
        t_EMI += x[4]

    elif x[6] == "other":
        t_other += x[4]

print(total)

print(t_food)
print(t_entertainment)
print(t_business)
print(t_rent)
print(t_EMI)
print(t_other)
```

```

        return render_template("today1.html", texpanse = texpanse, expense =
expense, total = total ,
                                t_food = t_food,t_entertainment = t_entertainment,
                                t_business = t_business, t_rent = t_rent,
                                t_EMI = t_EMI, t_other = t_other )

@app.route("/month1")
def month():

    param1 = "SELECT DATE(date) as dt, SUM(amount) as tot FROM expenses
WHERE userid = " + str(session['id']) + " AND MONTH(date) = MONTH(current
timestamp) AND YEAR(date) = YEAR(current timestamp) GROUP BY DATE(date) ORDER
BY DATE(date)"
    res1 = ibm_db.exec_immediate(ibm_db_conn, param1)
    dictionary1 = ibm_db.fetch_assoc(res1)
    texpanse = []

    while dictionary1 != False:
        temp = []
        temp.append(dictionary1["DT"])
        temp.append(dictionary1["TOT"])
        texpanse.append(temp)
        print(temp)
        dictionary1 = ibm_db.fetch_assoc(res1)

    param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) +
" AND MONTH(date) = MONTH(current timestamp) AND YEAR(date) = YEAR(current
timestamp) ORDER BY date DESC"
    res = ibm_db.exec_immediate(ibm_db_conn, param)
    dictionary = ibm_db.fetch_assoc(res)
    expense = []
    while dictionary != False:
        temp = []
        temp.append(dictionary["ID"])
        temp.append(dictionary["USERID"])
        temp.append(dictionary["DATE"])
        temp.append(dictionary["EXPENSENAME"])
        temp.append(dictionary["AMOUNT"])
        temp.append(dictionary["PAYMODE"])
        temp.append(dictionary["CATEGORY"])
        expense.append(temp)
        print(temp)
        dictionary = ibm_db.fetch_assoc(res)

```

```

total=0
t_food=0
t_entertainment=0
t_business=0
t_rent=0
t_EMI=0
t_other=0

for x in expense:
    total += x[4]
    if x[6] == "food":
        t_food += x[4]

    elif x[6] == "entertainment":
        t_entertainment += x[4]

    elif x[6] == "business":
        t_business += x[4]
    elif x[6] == "rent":
        t_rent += x[4]

    elif x[6] == "EMI":
        t_EMI += x[4]

    elif x[6] == "other":
        t_other += x[4]

print(total)

print(t_food)
print(t_entertainment)
print(t_business)
print(t_rent)
print(t_EMI)
print(t_other)

return render_template("today1.html", texpanse = texpanse, expense =
expense, total = total ,
                        t_food = t_food,t_entertainment = t_entertainment,
                        t_business = t_business, t_rent = t_rent,
                        t_EMI = t_EMI, t_other = t_other )

@app.route("/year1")
def year():

```

```

        param1 = "SELECT MONTH(date) as mn, SUM(amount) as tot FROM expenses
WHERE userid = " + str(session['id']) + " AND YEAR(date) = YEAR(current
timestamp) GROUP BY MONTH(date) ORDER BY MONTH(date)"
        res1 = ibm_db.exec_immediate(ibm_db_conn, param1)
        dictionary1 = ibm_db.fetch_assoc(res1)
        texpanse = []

        while dictionary1 != False:
            temp = []
            temp.append(dictionary1["MN"])
            temp.append(dictionary1["TOT"])
            texpanse.append(temp)
            print(temp)
            dictionary1 = ibm_db.fetch_assoc(res1)

        param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) +
" AND YEAR(date) = YEAR(current timestamp) ORDER BY date DESC"
        res = ibm_db.exec_immediate(ibm_db_conn, param)
        dictionary = ibm_db.fetch_assoc(res)
        expense = []
        while dictionary != False:
            temp = []
            temp.append(dictionary["ID"])
            temp.append(dictionary["USERID"])
            temp.append(dictionary["DATE"])
            temp.append(dictionary["EXPENSENAME"])
            temp.append(dictionary["AMOUNT"])
            temp.append(dictionary["PAYMODE"])
            temp.append(dictionary["CATEGORY"])
            expense.append(temp)
            print(temp)
            dictionary = ibm_db.fetch_assoc(res)

        total=0
        t_food=0
        t_entertainment=0
        t_business=0
        t_rent=0
        t_EMI=0
        t_other=0

        for x in expense:

```

```

        total += x[4]
        if x[6] == "food":
            t_food += x[4]

        elif x[6] == "entertainment":
            t_entertainment += x[4]

        elif x[6] == "business":
            t_business += x[4]
        elif x[6] == "rent":
            t_rent += x[4]

        elif x[6] == "EMI":
            t_EMI += x[4]

        elif x[6] == "other":
            t_other += x[4]

    print(total)

    print(t_food)
    print(t_entertainment)
    print(t_business)
    print(t_rent)
    print(t_EMI)
    print(t_other)

    return render_template("today1.html", texpanse = texpanse, expense =
expense, total = total ,
                           t_food = t_food,t_entertainment = t_entertainment,
                           t_business = t_business, t_rent = t_rent,
                           t_EMI = t_EMI, t_other = t_other )

#log-out

@app.route('/logout')

def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    session.pop('email', None)
    return render_template('home1.html')

port = os.getenv('VCAP_APP_PORT', '8080')
if __name__ == "__main__":

```

```
app.secret_key = os.urandom(12)
app.run(debug=True, host='0.0.0.0', port=port)
```

OUTPUT:

add1.html

File | C:/Users/mahal/OneDrive/Desktop/sam/add1.html

MyBudget Home Add History LIMIT Report

Add Expense

Date: 16-11-2022 11:30

Expense name: miscellaneous

Expense Amount: 8500

Category: cash

other

Add

display1.html

File | C:/Users/mahal/OneDrive/Desktop/sam/display1.html

MyBudget Home (current) Add History LIMIT Report

EXPENSES

2022-11-16 11:30:17	biriyani	₹ 150	cash	food	Edit	Delete
2022-9-19 16:37:56	laptop	₹ 150000	debitcard	EMI	Edit	Delete
2022-10-15 11:30:17	Television	₹ 25000	epayment	entertainment	Edit	Delete
2022-12-31 09:35:15	project	₹ 10000	creditcard	business	Edit	Delete
2022-11-26 15:12:53	rent	₹ 5000	onlinebanking	rent	Edit	Delete
2022-11-16 11:30:17	miscellaneous	₹ 8500	cash	other	Edit	Delete



