Personal Expense Tracker

Project Report

1 INTRODUCTION

1.1 PROJECT OVERVIEW

When it comes to tracking expenses, you can make your system as simple as collecting receipts and organizing them once a month.

You might get a little more information from other expense tracking systems (listing them in a spreadsheet, using money management software or even choosing an online application), but all methods have one thing in common: you have to get in the habit of thinking about your expenses.

It's very easy to misplace a receipt or forget about any cash you spent. You may even think that a cup of coffee or a trip to the vending machine isn't worth tracking — although those little expenses can add up amazingly fast.

There are all sorts of opportunities to throw a kick into your plan to track expenses. You have to get in the habit of doing so, to reduce those lapses, and make sure that the data you're basing financial decisions on is solid.

This project will request the clients to add their expenses and in view of their costs ,wallet status will be refreshed which will be noticeable to the client.

1.2 PURPOSE

- Help the people to track their expenses.
- Alert users when they exceed the limit of their budget.
- A personal finance app will not only help you with budgeting and accounting but also give you helpful insights about financial management

2 LITERATURE SURVEY

Literature survey is "A survey of related literature refers to a study done before or after selecting a research problem to know about the previous research work, ideas, theories, procedures, techniques, problems occurring during the research, etc. is done for.”

2.1 EXISTING PROBLEM

The already existing solutions possess the following problems:

It may seem like a lot of work to itemize your expenses when you first begin, but understanding why it's important to track expenses and how to do so with minimal effort can help you successfully commit to the activity and become more aware of your spending. Monitoring your expenses throughout the month holds you accountable for your finances in a few key ways.

After you set up a budget, which is a monthly plan for spending that takes into account your income and expenses, tracking expenses daily is essential to keeping you on that budget. If you don't track your money, you won't know when to stop spending in a given category (food or clothing, for example)

2.2 REFERENCES

The following research papers and projects were referred to during the process of literature survey. These papers were extracted from www.researchgate.net

https://www.researchgate.net/publication/273500084_Income_and_Expense_Tracker

https://www.researchgate.net/publication/351233145_Expense_Tracker

https://www.researchgate.net/publication/362517794_TRIPULATOR-The_Trip_Expense_Tracker

https://www.researchgate.net/publication/360620084_EXPENDITURE_MANAGEMENT_SYSTEM

2.3 PROBLEM STATEMENT

The problem statement of this project is defined user perspective to analyze what the users need to satisfy their needs.

The user needs a way to track expenses easily so that they can avoid unwanted expenses.

The user needs a way to get notified immediately once the budget limit is exceeded so that they stay away from debt.

The user needs a way to have clear ideas on transactions with people so that they can stay aware of their money .

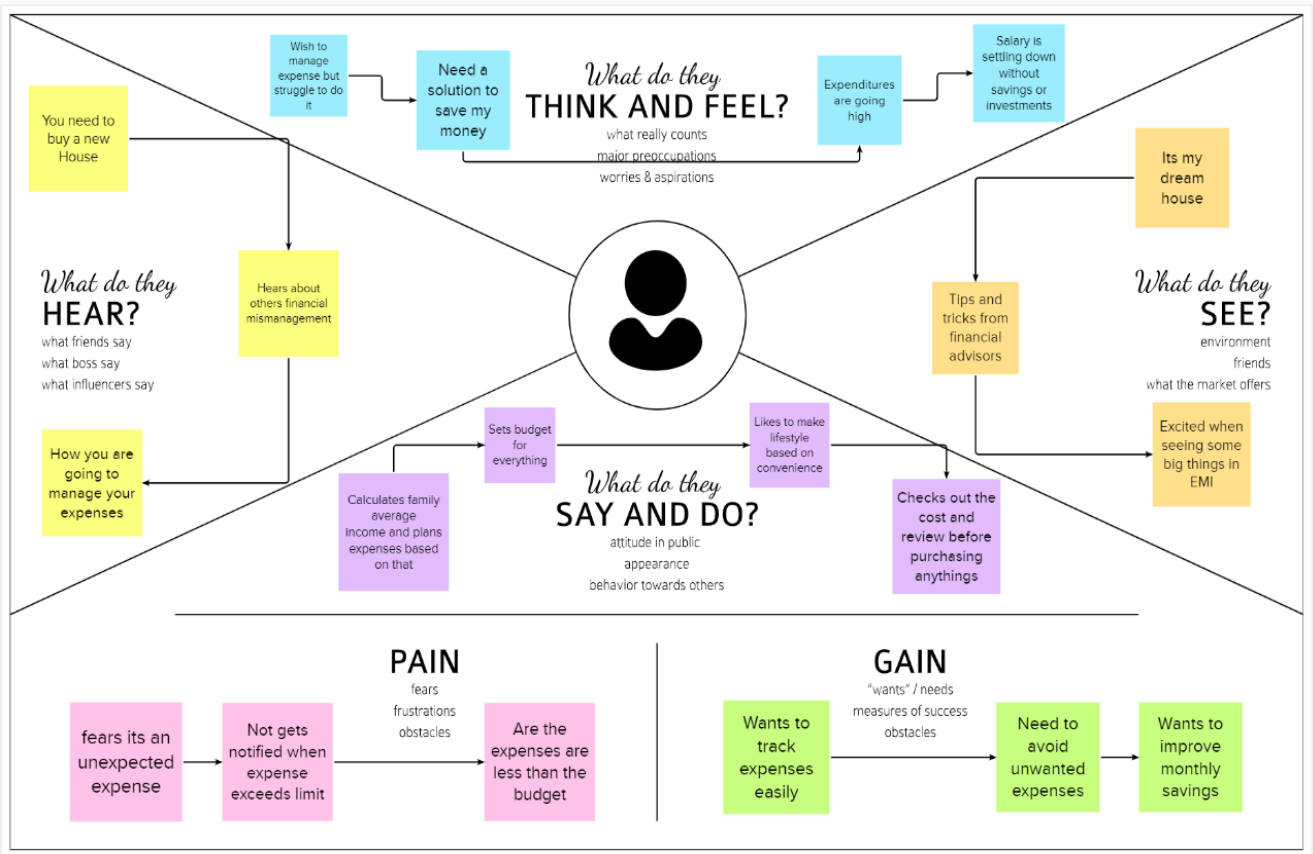
The user needs a way to plan their budget so that they can take care of their expenditure.

The user needs a way to track their expenses so that they can save their money.

3 IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

The empathy map is used to know about user perceptive. It is a visualization tool used to articulate what a product team knows about a user. This tool helps product teams build a broader understanding of user's needs.



3.2 Ideation & Brainstorming

Dhanush

Navigate to the dashboard	Edit User Profile	Visualize the expenses
Add income and expenses	Add remainder and get notify	Set budget

Dhileepan raja

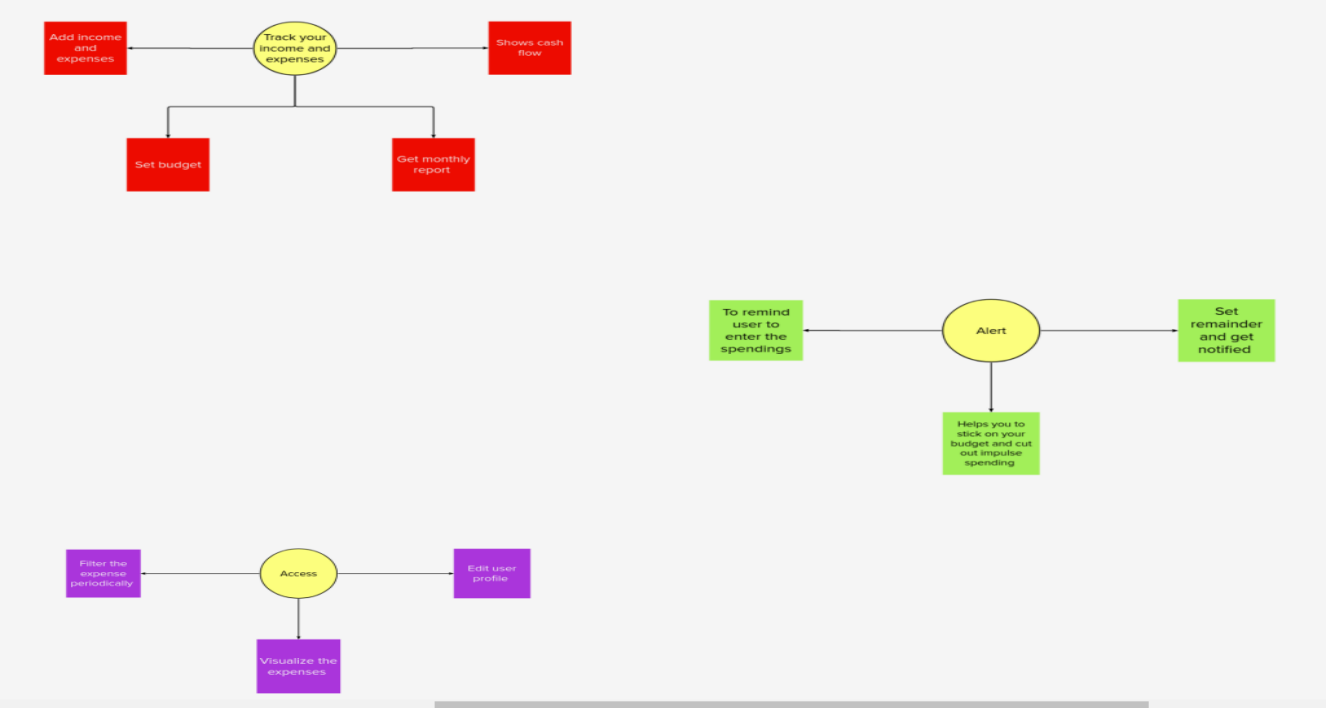
Filter the expenses graphically	Edit income and expenses	Keep accurate records
Create a additional steam of income	Shows cash flow	Generate Monthly report

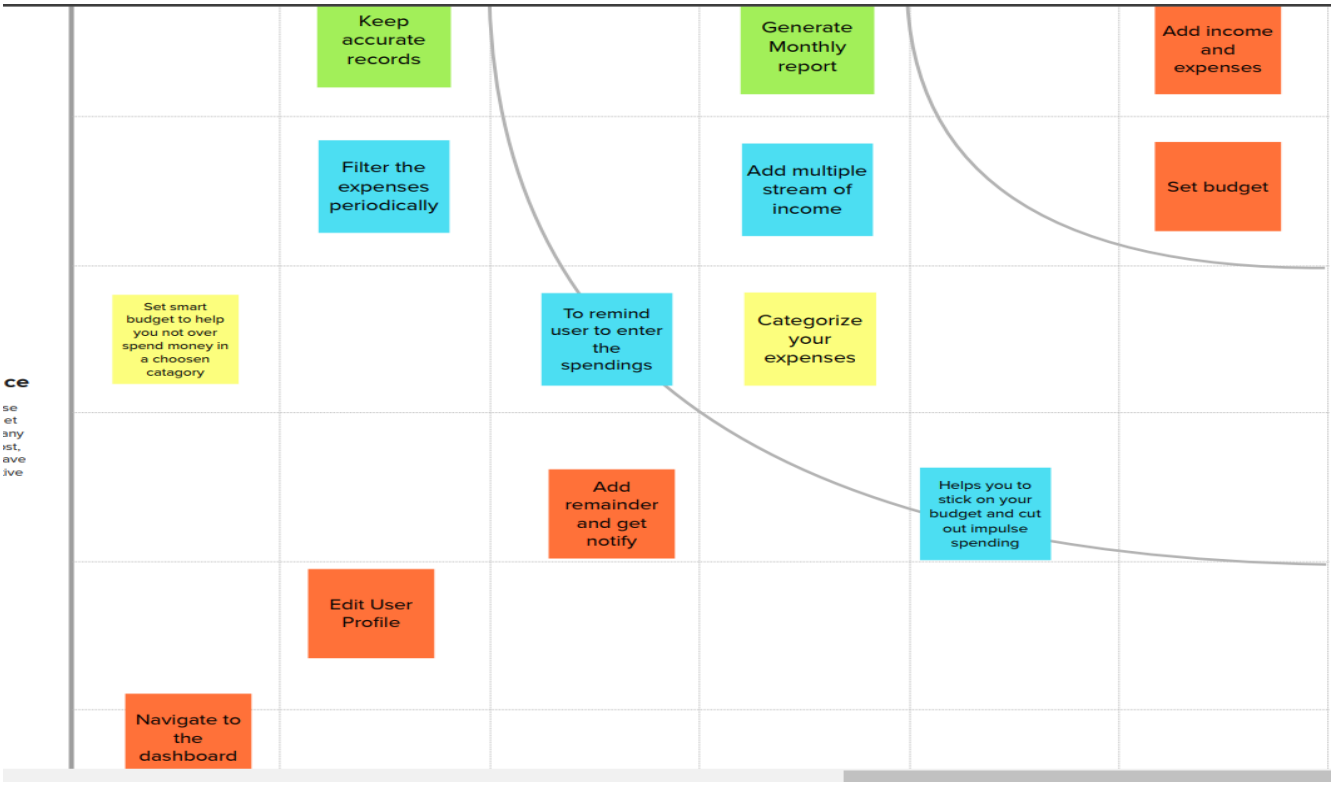
Dhinathayalan

Set smart budget to help you not over spend money in a choosen catagory	No need for complicated Excel sheets	Categorize your expenses
Feedback System	Get monthly report as pdf or excel sheet	Overspending / underspending of money

Dinesh

To remind user to enter the spendings	Categorize the expenses	Limitations for budget
Filter the expenses periodically	Add multiple stream of income	Helps you to stick on your budget and cut out impulse spending


Add income and expenses
Track your income and expenses
Shows cash flow
Set budget
Get monthly report
To remind user to enter the spendings
Alert
Set remainder and get notified
Helps you to stick on your budget and cut out impulse spending
Filter the expense periodically
Access
Edit user profile
Visualize the expenses


Keep accurate records
Generate Monthly report
Add income and expenses
Filter the expenses periodically
Add multiple stream of income
Set budget
Set smart budget to help you not over spend money in a choosen catagory
To remind user to enter the spendings
Categorize your expenses
Add remainder and get notify
Helps you to stick on your budget and cut out impulse spending
Edit User Profile
Navigate to the dashboard

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	In a Traditional Paper based expense tracking system, it is difficult to track our monthly expenses manually. Some of the records may get lost in case of fire, floods, etc. We are trying to solve this problem in a more efficient way.
2.	Idea / Solution description	This expense tracker is a computerised application which keeps track of all your finances and helps in accounting and budgeting.
3.	Novelty / Uniqueness	The User gets notified once their expense touches 50% 75% 90% & 100% of their limits. Display the costs on a monthly and weekly basis in a pie chart.
4.	Social Impact / Customer Satisfaction	This Application is able to generate reports of their spendings. It can create awareness among common people about finance.It makes users financially responsible and satisfy them without letting them to debt.
5.	Business Model (Revenue Model)	As this project is intended purely for educational purposes, we keep this application free of cost.
6.	Scalability of the Solution	This Application can handle large numbers of users and data with high performance and security. This application can be used for both large scale and small scale purposes.

3.4 Problem Solution fit

Project Title: Personal Expense Tracker Application **Project Design Phase-I - Solution Fit** **Team ID:** PNT2022TMID06716

1. CUSTOMER SEGMENT(S) CS

Be it a common man or a bigfish.... Our app comes in handy to all of those who wish to boost their expense potential. People who are unaware of financial things on how to spend their money can make use of this app.

6. CUSTOMER CONSTRAINTS CC

User have to entry every record manually, sometimes while maintaining a large amount of data may look messy. User who is maintaining the system must have some technical knowledge.

5. AVAILABLE SOLUTIONS AS

User can add their income and expenses. They have an option to set a limit on how much they can spend on their salary or savings. IF that particular limit is exceeded they are notified by email.

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

In a Traditional Paper based expense tracking system, it is difficult to track our monthly expenses manually. Some of the records may get lost in case of fire, floods, etc

9. PROBLEM ROOT CAUSE RC

Peoples are unaware of their spending and exceeds their limit.They spend a lot without calculating their in hand salary and savings in the first place.

7. BEHAVIOUR BE

People may kee notes on their mobile. They try to remember the expenses they do and calculate the whole expenses at once at the end of the month which may take a long time to calculate.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

This application can create awareness among common people about their income and expenses.It reduces time rather than entering details manually.

4. EMOTIONS: BEFORE / AFTER EM

Frustration, confusion, rage / Feeling smart , leading happy life, being an example for others

10. YOUR SOLUTION SL

This expense tracker is a computerized application which keeps track of all your finances and helps in accounting and budgeting. It can handle large numbers of users and data with high performance and security. This application can be used for both large scale and small scale purposes.

8.CHANNELS of BEHAVIOUR CH

ONLINE

People may use online tools to calculate their expenses

OFFLINE

People may use a ledger to calculate their expenses.

Identify strong TR & EM

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through our Website
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Add Expenses	Enter day to day expenditure as input Categorise the expenditure
FR-4	Remainder Mail	Reminds the user once their budget limit crosses 50% 75% 90% 100% of their limit
FR-5	Graph	Creates graph based on the day to day and weekly expenditure
FR-6	Add Salary	User needs to add salary at the start of the month
FR-7	Export CSV	User can export the raw data of their expenses for their own reference in the form of pdf or csv

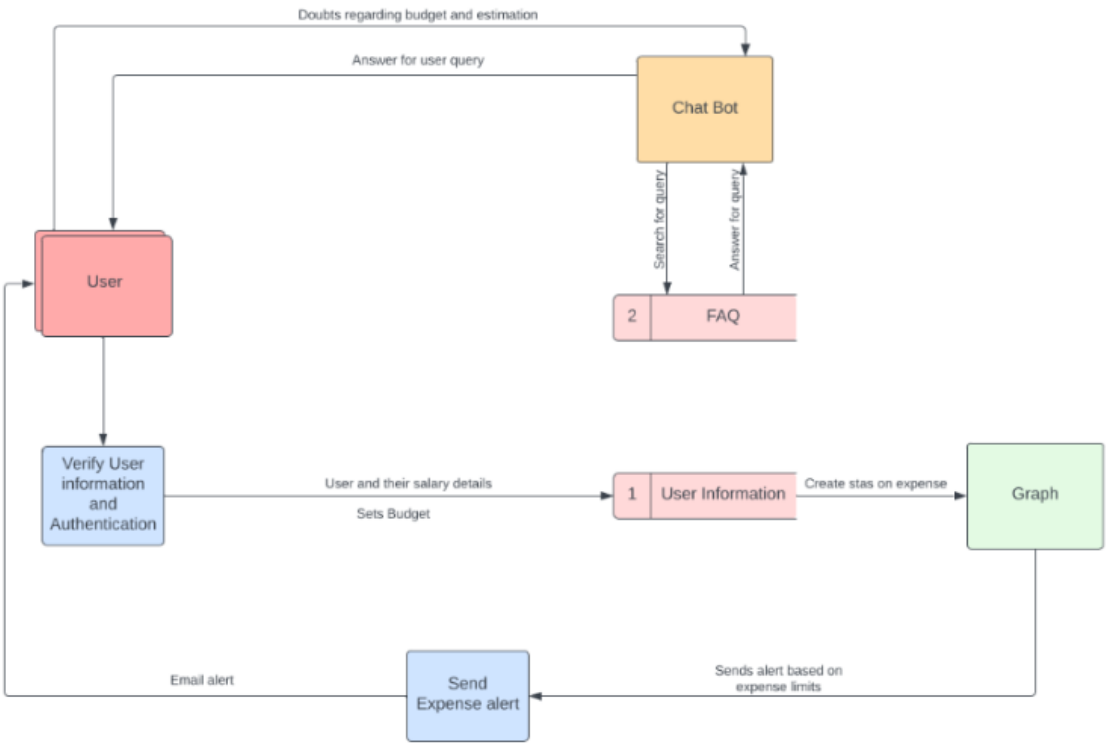
4.2 Non-Functional requirement

Following are the non-functional requirements of the proposed solution.

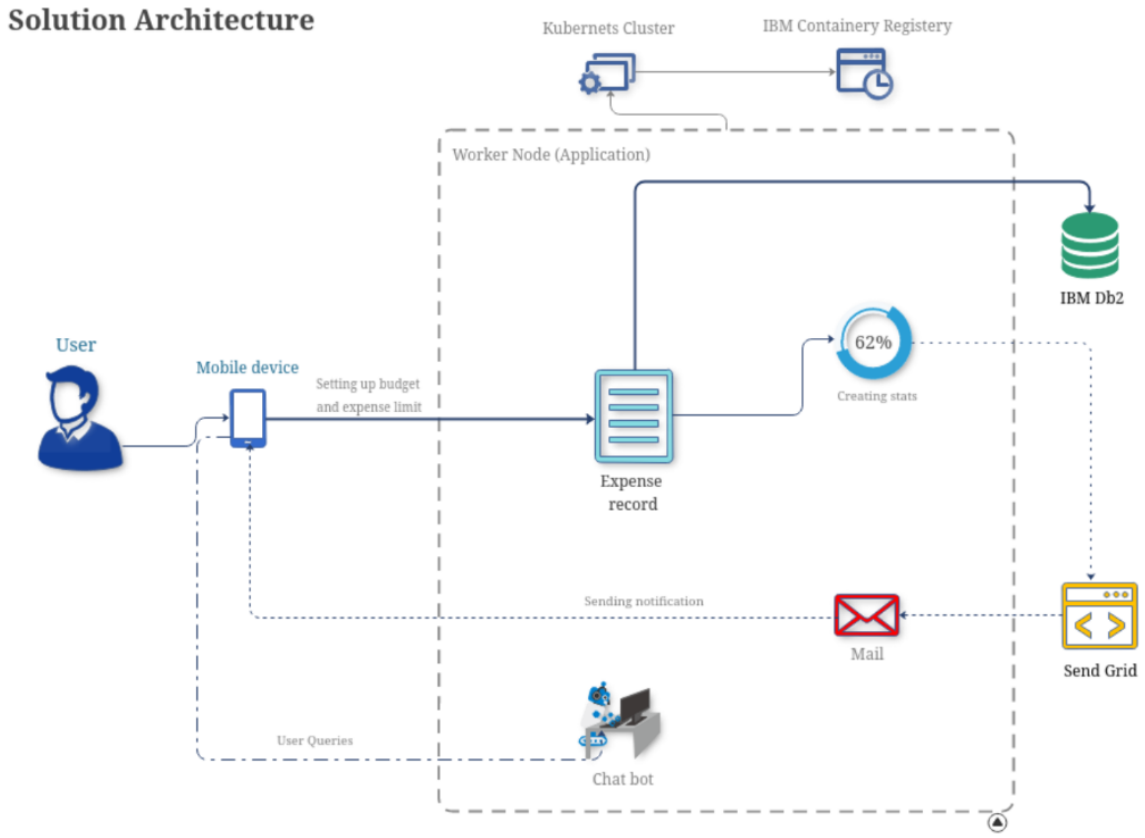
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The interface is user friendly and it is easy to use by all type of users.
NFR-2	**Security**	Every data is secured and encrypted using many encryption algorithms.
NFR-3	**Reliability**	The transaction must rollback if there is any system failure or network issue.The data is saved when updation of data is failed between the process.Even if there is a failure,the data can be restored within some time.
NFR-4	**Performance**	The application should not take more than 30 seconds to load.The response is quick even if there is heavy traffic.
NFR-5	**Availability**	This application is globally available all the time regardless of the traffic.
NFR-6	**Scalability**	This application is scalable for multiple users as we use docker and kubernetes.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Mobile Number	I can register & access the dashboard with Mobile Login	Low	Sprint-2
	Login	USN-4	As a user, I can log into the application by entering email & password	I can access the application	High	Sprint-2
	Dashboard	USN-5	As a user, to able to see the basic features of the application can be viewed	I can see the particular feature by click on it	High	Sprint-2
Customer (Web user)		USN-6	As a customer is able to set up their salary and save his expenses	I can manage and control my expenses made	Medium	Sprint-4
		USN-7	Customer can able to track their expense by checking the expenditure graph	I can change data in database to calculate expenses made	Medium	Sprint-3
		USN-8	Customer can export their expense graph	I can manage changes easily	High	Sprint-3
		USN-9	Customer can edit their expense limits on the mid way of the month	I can set out a fixed value I know must be incurred (i.e. standing expenses).	High	Sprint-4

6.PROJECT PLANNING & SCHEDULING

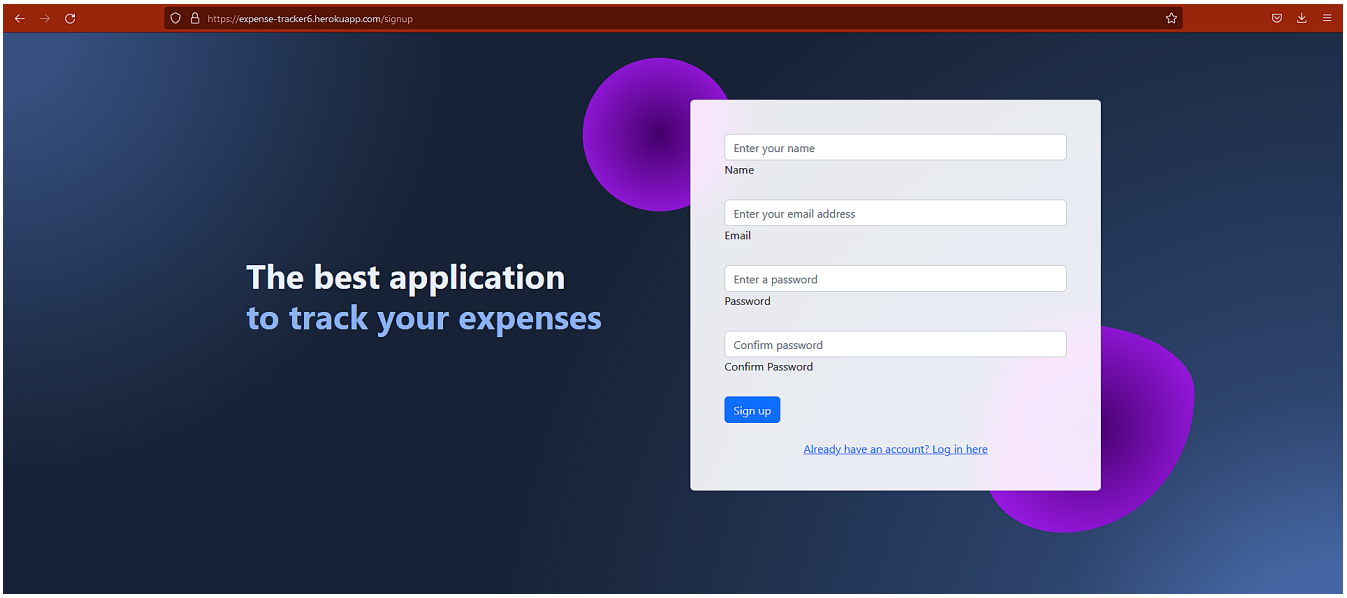
6.1 Sprint Planning & Estimation

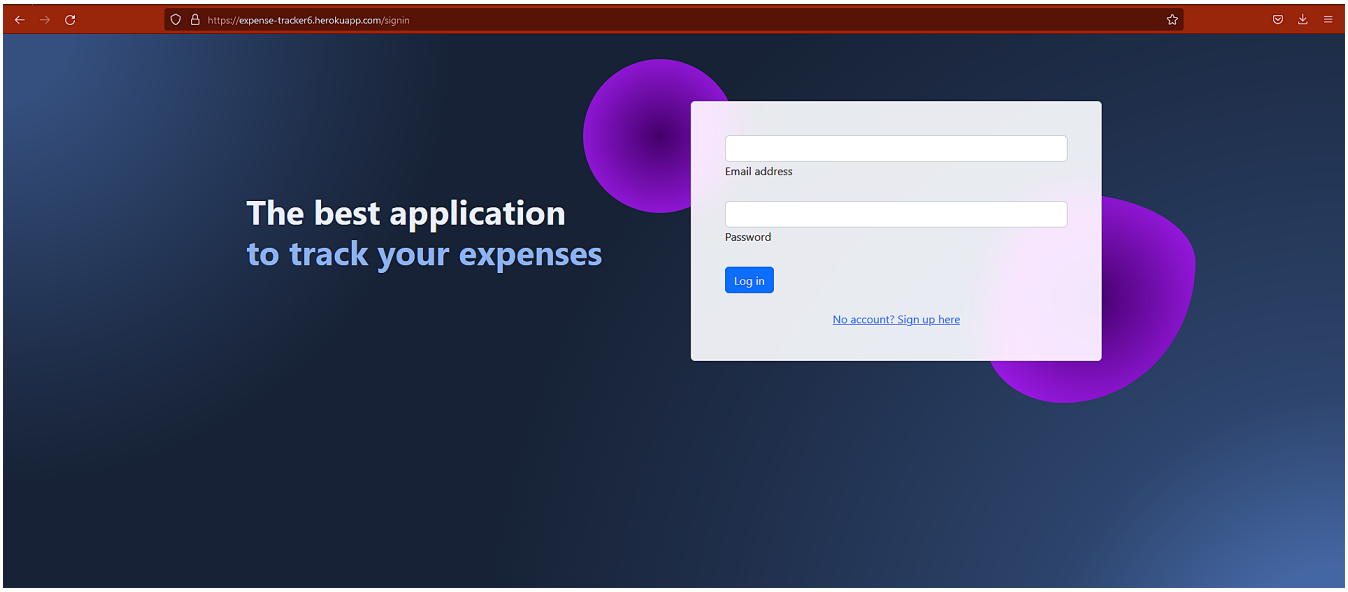
6.2 Sprint Delivery Schedule

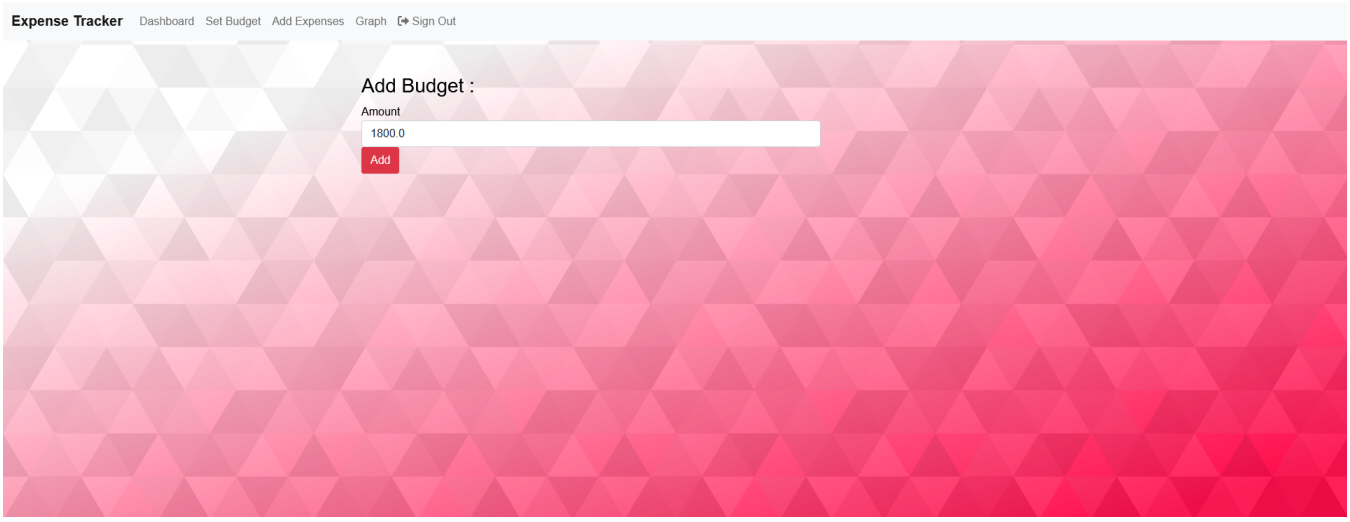
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)
Sprint-1	5	6 Days	24 Oct 2022	29 Oct 2022
Sprint-2	7	6 Days	31 Oct 2022	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022

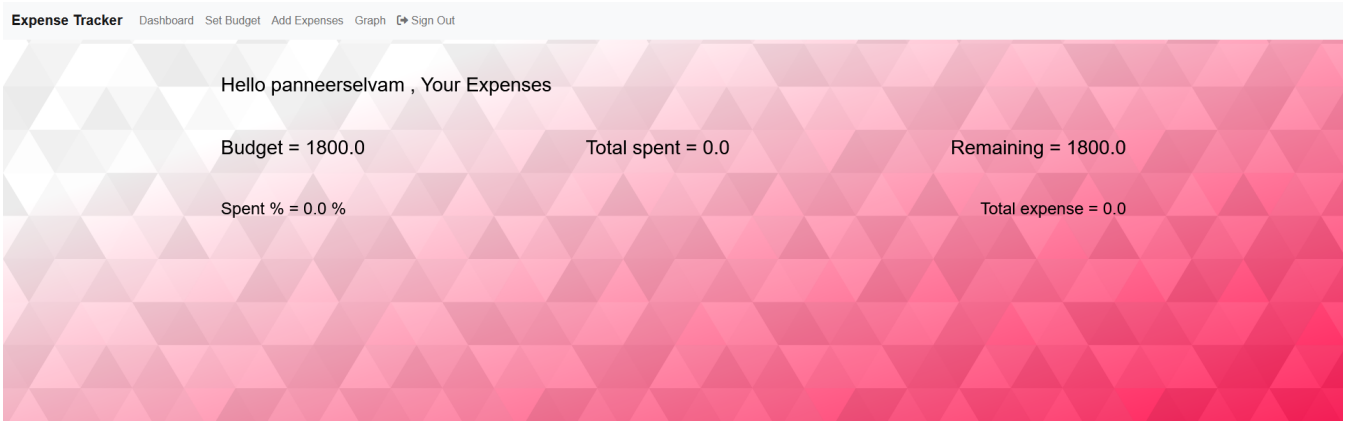
7.CODING & SOLUTIONING

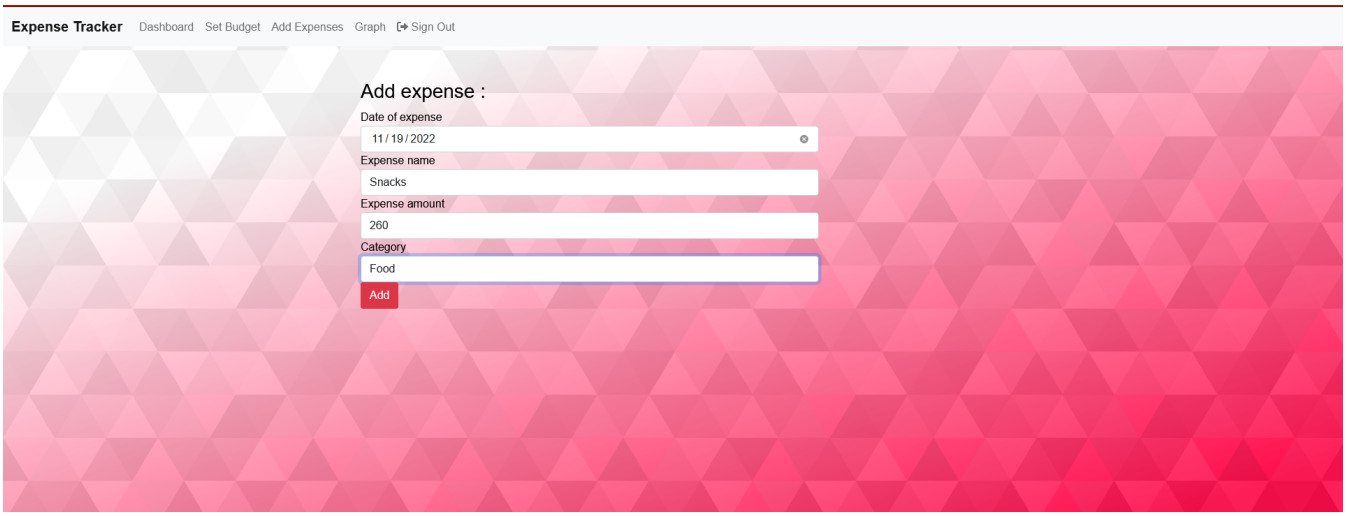
7.1 Dashboard

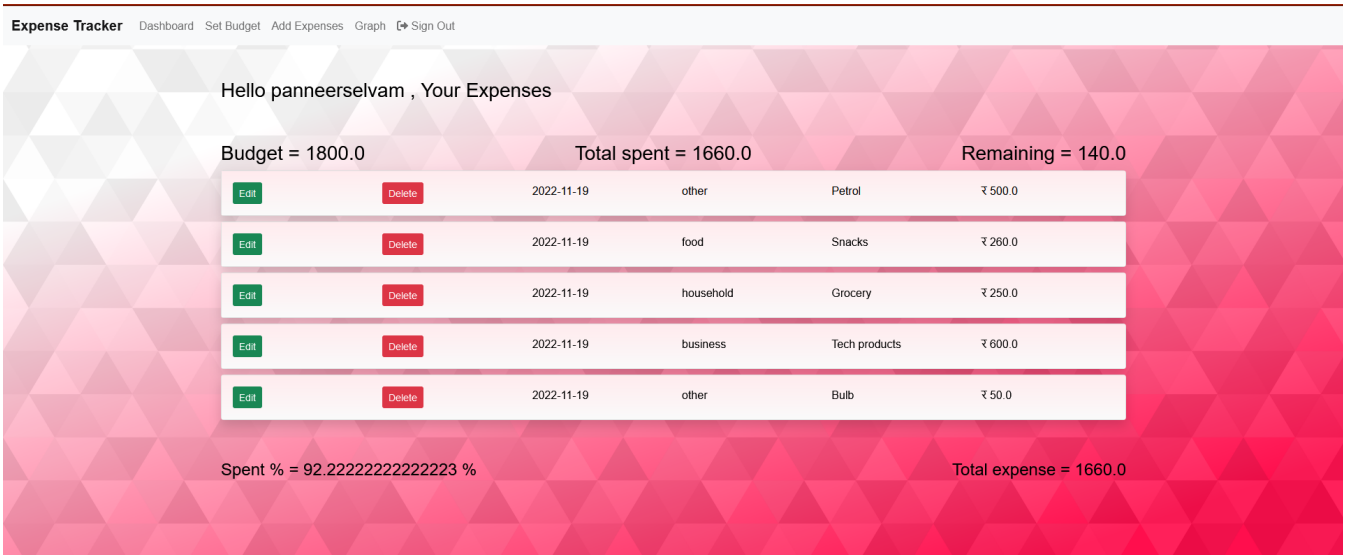


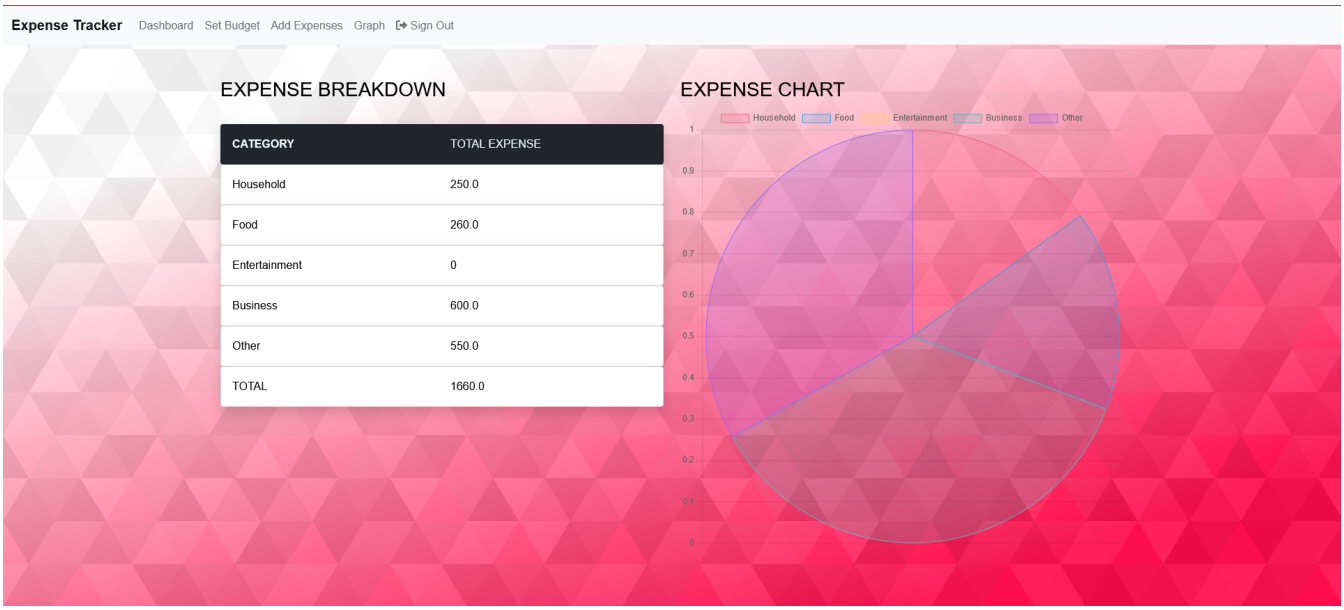



Expense Tracker
Dashboard
Set Budget
Add Expenses
Graph
Sign Out
Add Budget :
Amount
1800.0
Add


Expense Tracker
Dashboard
Set Budget
Add Expenses
Graph
Sign Out
Hello panneerselvam , Your Expenses
Budget = 1800.0
Total spent = 0.0
Remaining = 1800.0
Spent % = 0.0 %
Total expense = 0.0


Expense Tracker
Dashboard
Set Budget
Add Expenses
Graph
Sign Out
Add expense :
Date of expense
11 / 19 / 2022
Expense name
Snacks
Expense amount
260
Category
Food
Add


Expense Tracker
Dashboard
Set Budget
Add Expenses
Graph
Sign Out
Hello panneerselvam , Your Expenses
Budget = 1800.0
Total spent = 1660.0
Remaining = 140.0
Edit Delete 2022-11-19 other Petrol ₹ 500.0
Edit Delete 2022-11-19 food Snacks ₹ 260.0
Edit Delete 2022-11-19 household Grocery ₹ 250.0
Edit Delete 2022-11-19 business Tech products ₹ 600.0
Edit Delete 2022-11-19 other Bulb ₹ 50.0
Spent % = 92.22222222222223 %
Total expense = 1660.0



CATEGORY	TOTAL EXPENSE
Household	250.0
Food	260.0
Entertainment	0
Business	600.0
Other	550.0
TOTAL	1660.0

9 ADVANTAGES & DISADVANTAGES

Advantages:

- With this application users can keep track of their expenses without fear of runing out of budget.
- The users data are stored in a secured manner.
- Authentication is needed to access the application hence it is more secured.
- The app manages the expenses made and could even calculate the remaining amount.
- The application could make pie chart of the expenses made so the user can be aware of the expenses made.
- With this application the user can reduce unnecessary expenses.
- The application is free of cost.
- The user's will be notified if their expense exceeds the limit.

Disadvantages:

- Eventhough expense is tracked continuously it does not guarentee that user's expense will be reduced.
- The user need to periodically update the expenses.
- The data collected from the user can be misused.

10 CONCLUSION

The application tends to be useful for people who tends to over exploit money. It will be useful to manage the expenses of the user. You will need a defined goal and a clear vision for grasping the business and personal finances. That’s when an expense tracking app comes into the picture. An expense tracking app is an exclusive suite of services for people who seek to handle their earnings and plan their expenses and savings efficiently. It helps you track all transactions like bills, refunds, payrolls, receipts, taxes, etc., on a daily, weekly, and monthly basis.

11 Future Scope

- Optimize the working of the application.
- Payment options could be included .
- Increase efficiency.
- Security can be increased.
- User's feedback can be got for further improvement.

12 APPENDIX

app.py

```
from flask import Flask, render_template, redirect, url_for, request, session, flash
import ibm_db
import sendgrid
import os
from dotenv import load_dotenv
from sendgrid.helpers.mail import Mail, Email, To, Content

app = Flask(__name__)
# secret key required to maintain unique user sessions
app.secret_key =
'f39c244d6c896864abe3310b839091799fed56007a438d637baf526007609fe0'

# establish connection with IBM Db2 Database
connection = ibm_db.connect("DATABASE=bludb;HOSTNAME=8e359033-a1c9-4643-
82ef-
8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30120;SECU
RITY=SSL;UID=gyq42313;PWD=TRn8xYTPTGloNwQC;", "", "")

load_dotenv()  # load keys from .env
sg = sendgrid.SendGridAPIClient(api_key=os.environ.get(
    'SENDGRID_API_KEY'))  # set SendGrid API Key
# the address that sends emails to the users
from_email = Email("dhanushcodepro@gmail.com")

# Handle expense model according to ibm db
```

```
@app.route('/')
@app.route('/dashboard')
def dashboard():
    if 'username' not in session:
        # ask user to sign in if not done already
        return redirect(url_for('signin'))
    # Fetch the list of expenses from db
    sql = 'select * from expenses where cid = '+str(session['id'])
    stmt = ibm_db.exec_immediate(connection, sql)
    flag=0
    expense_list = {}
    i=0
    while (res:=ibm_db.fetch_assoc(stmt)) != False:
        expense_list[i] = res
        flag=1
        i+=1
    # go to homepage if signed in
    sql='select sum(eamount) from expenses where cid= '+str(session['id'])
    stmt = ibm_db.exec_immediate(connection, sql)
    sum_dict= ibm_db.fetch_assoc(stmt)
    sum=0.0
    sum = sum_dict['1']
    if flag !=1:
        sum=0.0
    sql='select budget from users where id= '+str(session['id'])
```

```
    stmt = ibm_db.exec_immediate(connection, sql)
    budget_list= ibm_db.fetch_assoc(stmt)
    # budget = budget_list.values()
    for key,value in budget_list.items():
        pass
    rem = value - sum
    per=0
    if value !=0:
        per = (sum/value)*100
    return render_template('dashboard.html',
ex_list=expense_list,esum=sum,budget=value,rem=rem,per=per,name=session['usernam
e'])




@app.route('/addexpense')
def add():
    return render_template('addexpense.html')

@app.route('/addbudget')
def addb():
    sql='select budget from users where id= '+str(session['id'])
    stmt = ibm_db.exec_immediate(connection, sql)
    budget_list= ibm_db.fetch_assoc(stmt)
    # budget = budget_list.values()
    for key,value in budget_list.items():
        pass
    return render_template('addbudget.html',budget=value,id=session['id'])
```

```
@app.route('/addbudget/<int:id>',methods=['POST'])
def addbudget(id):
    budget = request.form['budget']
    sql = 'update users set budget = ? where id = '+str(id)
    # Add to the database here
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.bind_param(pstmt, 1, budget)
    ibm_db.execute(pstmt)
    return redirect('/dashboard')

@app.route('/addexpense', methods=['POST'])
def addexpense():
    date = request.form['date']
    expensename = request.form['expensename']
    amount = request.form['amount']
    category = request.form['category']
    sql = 'INSERT INTO expenses(edate,ename,eamount,ecategory,cid) VALUES(?,?,?,?,?)'
    # Add to the database here
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.bind_param(pstmt, 1, date)
    ibm_db.bind_param(pstmt, 2, expensename)
    ibm_db.bind_param(pstmt, 3, amount)
    ibm_db.bind_param(pstmt, 4, category)
    ibm_db.bind_param(pstmt, 5, session['id'])
    ibm_db.execute(pstmt)
    flash('Expense added Successfully')
```

```
    return redirect('/dashboard')


@app.route('/expense/update/<int:id>')
def update(id):
    # Get from the database
    sql = 'select * from expenses where id = '+str(id)
    # Add to the database here
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.execute(pstmt)
    acc = ibm_db.fetch_assoc(pstmt)
    return render_template('updateexpense.html',acc=acc)


@app.route('/edit', methods=['POST'])
def edit():
    id = request.form["id"]
    date = request.form['date']
    expensename = request.form['expensename']
    amount = request.form['amount']
    category = request.form['category']

    sql = 'update expenses set edate = ?,ename = ? ,eamount = ? ,ecategory = ? where id =
'+str(id)
    # Add to the database here
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.bind_param(pstmt, 1, date)
    ibm_db.bind_param(pstmt, 2, expensename)
    ibm_db.bind_param(pstmt, 3, amount)
```

```
    ibm_db.bind_param(pstmt, 4, category)
    ibm_db.execute(pstmt)

    return redirect('/dashboard')



@app.route('/expense/delete/<int:id>', methods=['GET'])
def delete(id):
    # Database operation
    # flash(str(id))
    sql = 'delete from expenses where id = '+str(id)   #check if user is already registered
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.execute(pstmt)

    return redirect('/dashboard')



@app.route('/graph')
def graph():
    sql = 'select * from expenses where cid = '+str(session['id'])
    stmt = ibm_db.exec_immediate(connection, sql)
    expense_list = {}
    i=0
    while (res:=ibm_db.fetch_assoc(stmt)) != False:
        expense_list[i] = res
        i+=1
    total = 0
    household = 0
    food = 0
```

```
    entertainment = 0
    business = 0
    other = 0
    for key,value in expense_list.items():
        total += value['EAMOUNT']
        if value['ECATEGORY'] == 'household':
            household += value['EAMOUNT']
        elif value['ECATEGORY'] == 'food':
            food += value['EAMOUNT']
        elif value['ECATEGORY'] == 'entertainment':
            entertainment += value['EAMOUNT']
        elif value['ECATEGORY'] == 'business':
            business += value['EAMOUNT']
        elif value['ECATEGORY'] == 'other':
            other += value['EAMOUNT']

    return render_template('graph.html',  total=total, household=household, food=food,
entertainment=entertainment, business=business, other=other)



@app.route('/signout')
def signout():
    session.pop('username', None)  # remove user session upon signing out
    return redirect('/')



@app.route('/signup')
def register():
    if 'username' in session:   #inform user if they're already signed in the same session
```

```
        flash('You are already signed in! Sign out to login with a different account')
        return redirect(url_for('dashboard'))

    else:
        return render_template('signup.html') #take user to the registration page


@app.route('/signup', methods=['POST'])
def regform():
    uname = request.form['uname']   #get user id and password from the form
    email = request.form['email']
    pwd = request.form['pass']
    print(uname,email,pwd)
    sql = 'SELECT * from users WHERE email=?'   #check if user is already registered
    pstmt = ibm_db.prepare(connection, sql)
    ibm_db.bind_param(pstmt, 1, email)
    ibm_db.execute(pstmt)

    acc = ibm_db.fetch_assoc(pstmt)

    if acc:     #inform user to sign in if they have an existing account
        flash('You are already a member. Please sign in using your registered credentials')

    else:
        sql = 'INSERT INTO users(username,password,email) VALUES(?,?,?)' #insert
credentials of new user to the database
        pstmt = ibm_db.prepare(connection, sql)
        ibm_db.bind_param(pstmt, 1, uname)
        ibm_db.bind_param(pstmt, 2, pwd)
```

```
        ibm_db.bind_param(pstmt, 3, email)
        ibm_db.execute(pstmt)




        flash('Registration Successful! Sign in using the registered credentials to continue')


    return redirect(url_for('signin'))  #ask users to sign in after registration




@app.route('/signin')
def signin():
    if 'username' in session:  # inform user if they're already signed in the same session
        flash('You are already signed in! Sign out to login with a different account')
        return redirect(url_for('dashboard'))
    return render_template('login.html')  # take user to the sign in page


@app.route('/signinform', methods=['POST'])
def signinform():
    uid = request.form['email']  # get user id and password from the form
    pwd = request.form['pass']

    # check user credentials in the database
    sql = 'SELECT * from users WHERE email=? AND password=?'
    pstmt = ibm_db.prepare(connection, sql)
```

```
    ibm_db.bind_param(pstmt, 1, uid)
    ibm_db.bind_param(pstmt, 2, pwd)
    ibm_db.execute(pstmt)

    acc = ibm_db.fetch_assoc(pstmt)

    if acc: #if the user is already registered to the application
        session['username'] = acc['USERNAME']
        session['id'] = acc['ID']
        flash(session['username'] + str(session['id'])+'Signed in successfully!')
        return redirect(url_for('dashboard'))
    else:   #warn upon entering incorrect credentials
        flash('Incorrect credentials. Please try again!')
        return render_template('login.html')
if __name__ == '__main__':
    app.run(debug=True)
```

addbudget.html

```
{% extends 'base.html' %}
{% block body %}

<div class="container">
    <div class="row">
        <div class="col-md-8">
            <h3 class="mt-5" style="margin-left: 200px;">Add Budget :</h3>
            <form method='post' action='/addbudget/{{id}}' style="margin-left: 200px;">
                <div class="form-group">
                    <label>Amount</label>
```

```
                <input class="form-control" type="text" name="budget" id="budget"
value="{{budget}}">
            </div>
            <input class="btn btn-danger" type="submit" value="Add">
        </form>
    </div>
  </div>
</div>
{% endblock %}
```

addexpense.html

```
{% extends 'base.html' %}
{% block body %}

<div class="container">
  <div class="row">
    <div class="col-md-8">
      <h3 class="mt-5" style="margin-left: 200px;">Add expense :</h3>
      <form method='post' action='/addexpense' style="margin-left: 200px;">
        <div class="form-group">
          <label>Date of expense</label>
          <input class="form-control" type="date" name="date" id="date">
        </div>
        <div class="form-group">
          <label>Expense name</label>
          <input class="form-control" type="text" name="expensename"
id="expensename">
        </div>
```

```
            <div class="form-group">
                <label>Expense amount</label>
                <input class="form-control" type="text" name="amount" id="amount">
            </div>
            <div class="form-group">
                <label>Category</label>
                <select class="form-control" name="category" id="category">
                    <option value="household">Household</option>
                    <option value="food">Food</option>
                    <option value="entertainment">Entertainment</option>
                    <option value="business">Business</option>
                    <option value="other">Other</option>
                </select>
            </div>
            <input class="btn btn-danger" type="submit" value="Add">
        </form>
    </div>
  </div>
</div>
{% endblock %}
```

base.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  {% block head %} {% endblock %}
  <meta charset="UTF-8">
```

```
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.2.0/css/all.min.css">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
        crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js@3.2.0/dist/chart.min.js"></script>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@700&display=swap"
rel="stylesheet">
    <link rel="stylesheet" href="{{url_for('static', filename='custom.css')}}">
</head>

<body>
    <nav class="navbar navbar-expand-lg sticky-top navbar-light bg-light">
        <div class="container-fluid">
            <a class="navbar-brand fw-bold" href="/">
                Expense Tracker
            </a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
```

```
target="#navbarNav"
            aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav">
                <li class="nav-item">
                    <a class="nav-link" href="/" href="#">Dashboard</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="/addbudget">Set Budget <span class="sr-
only">(current)</span></a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="/addexpense">Add Expenses <span class="sr-
only">(current)</span></a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="/graph">Graph</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="/signout">
                        <i class="fa-solid fa-right-from-bracket"></i>
                        Sign Out
                    </a>
                </li>
            </ul>
        </div>
```

```
        </div>
    </nav>

    {% with messages = get_flashed_messages() %}
    {% if messages %}
    {% for message in messages %}
    <script>alert('{{ message }}')</script>
    {% endfor %}
    {% endif %}
    {% endwith %}

    {% block body %} {% endblock %}
</body>

</html>
```

dashboard.html

```
{% extends 'base.html' %}

{% block head %}
<title>Expense Tracker - Dashboard</title>
{% endblock %}



{% block body %}
<div class="container" >
    <h3 class="mt-5">Hello {{name}} , Your Expenses</h3>
    <div class="d-flex justify-content-between">
```

```
    <h3 class="mt-5">Budget = {{budget}} </h3>
    <h3 class="mt-5">Total spent = {{esum}}</h3>

    <h3 class="mt-5">Remaining = {{rem}}</h3>
  </div>
  {% for key,value in ex_list.items() %}
  <div class="row" >
    <div class="col-md-12">
      <div class="card shadow mb-2 bg-white rounded">
        <div class="card-body" style="background: linear-gradient(#FFEBEE,#FAFAFA)">
          <div class="row">
            <div class="col-md-2">
              <a href="expense/update/{{value['ID']}}" class="btn btn-sm btn-
success">Edit</a>
            </div>
            <div class="col-md-2">
              <a href="expense/delete/{{value['ID']}}" class="btn btn-sm btn-
danger">Delete</a>
            </div>
            <div class="col-md-2" style="color: #000000">
              {{value['EDATE']}}
            </div>
            <div class="col-md-2"style="color: #000000">
              {{value['ECATEGORY']}}
            </div>
            <div class="col-md-2"style="color: #000000">
              {{value['ENAME']}}
            </div>
            <div class="col-md-2"style="color: #000000">
```

```
                    ₹ {{value['EAMOUNT']}}
                </div>
            </div>
        </div>
    </div>
  </div>
  {% endfor %}
  <div class="d-flex justify-content-between">
    <h4 class="mt-5">Spent % = {{per}} %    </h3>
    <h4 class="mt-5">Total expense = {{esum}}</h3>
  </div>
</div>




</body>
{% endblock %}
</html>
```

graph.html

```
{% extends 'base.html' %}
{% block body %}

<div class="container">
  <div class="row">
    <div class="col-md-6">
      <h3 class="mt-5">EXPENSE BREAKDOWN</h3><br>
```

```
<div class="card shadow bb-2 bg-dark rounded">
  <div class="card-body">
    <div class="row">
      <div class="col-md-6" style="color: aliceblue;"><b>CATEGORY</b></div>
      <div class="col-md-6" style="color: aliceblue;">TOTAL EXPENSE</div>
    </div>
  </div>
</div>
<div class="card shadow bb-2 bg-white rounded">
  <div class="card-body" style="color: #000000">
    <div class="row">
      <div class="col-md-6">Household</div>
      <div id="thousehold" class="col-md-6">{{household}}</div>
    </div>
  </div>
</div>
<div class="card shadow bb-2 bg-white rounded">
  <div class="card-body" style="color: #000000">
    <div class="row">
      <div class="col-md-6">Food</div>
      <div id="tfood" class="col-md-6">{{food}}</div>
    </div>
  </div>
</div>
<div class="card shadow bb-2 bg-white rounded">
  <div class="card-body" style="color: #000000">
    <div class="row">
      <div class="col-md-6">Entertainment</div>
      <div id="tentertainment" class="col-md-6">{{entertainment}}</div>
```

```
            </div>
        </div>
    </div>
    <div class="card shadow bb-2 bg-white rounded">
        <div class="card-body" style="color: #000000">
            <div class="row">
                <div class="col-md-6">Business</div>
                <div id="tbusiness" class="col-md-6">{{business}}</div>
            </div>
        </div>
    </div>
    <div class="card shadow bb-2 bg-white rounded">
        <div class="card-body" style="color: #000000">
            <div class="row">
                <div class="col-md-6">Other</div>
                <div id="tother" class="col-md-6">{{other}}</div>
            </div>
        </div>
    </div>
    <div class="card shadow bb-2 bg-white rounded">
        <div class="card-body" style="color: #000000">
            <div class="row">
                <div class="col-md-6">TOTAL</div>
                <div class="col-md-6">{{total}}</div>
            </div>
        </div>
    </div>
</div>
<div class="col-md-6">
```

```
<h3 class="mt-5">EXPENSE CHART</h3>
<canvas id="myChart" width="400" height="400"></canvas>
<script>
    let household = document.getElementById('thousehold').innerHTML
    let food = document.getElementById('tfood').innerHTML
    let entertainment = document.getElementById('tentertainment').innerHTML
    let business = document.getElementById('tbusiness').innerHTML
    let other = document.getElementById('tother').innerHTML
    var ctx = document.getElementById('myChart').getContext('2d');
    var myChart = new Chart(ctx, {
        type: 'pie',
        data: {
            labels: ['Household','Food', 'Entertainment', 'Business', 'Other'],
            datasets: [{
                label: 'Expense amount',
                data: [household,food,entertainment,business,other],
                backgroundColor: [
                    'rgba(255, 99, 132, 0.2)',
                    'rgba(54, 162, 235, 0.2)',
                    'rgba(255, 206, 86, 0.2)',
                    'rgba(75, 192, 192, 0.2)',
                    'rgba(153, 102, 255, 0.2)',
                    'rgba(255, 159, 64, 0.2)'
                ],
                borderColor: [
                    'rgba(255, 99, 132, 1)',
                    'rgba(54, 162, 235, 1)',
                    'rgba(255, 206, 86, 1)',
                    'rgba(75, 192, 192, 1)',
```

```
                    'rgba(153, 102, 255, 1)',
                    'rgba(255, 159, 64, 1)'
                ],
                borderWidth: 1
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: true
                }
            }
        }
    });
</script>
</div>
</div>
</div>


{% endblock %}
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Log in</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8"
crossorigin="anonymous"></script>
</head>
<body>
  <!-- Section: Design Block -->
<section class="background-radial-gradient overflow-hidden vh-100">
  <style>
    .background-radial-gradient {
      background-color: hsl(218, 41%, 15%);
      background-image: radial-gradient(650px circle at 0% 0%,
          hsl(218, 41%, 35%) 15%,
          hsl(218, 41%, 30%) 35%,
          hsl(218, 41%, 20%) 75%,
          hsl(218, 41%, 19%) 80%,
          transparent 100%),
        radial-gradient(1250px circle at 100% 100%,
          hsl(218, 41%, 45%) 15%,
          hsl(218, 41%, 30%) 35%,
          hsl(218, 41%, 20%) 75%,
          hsl(218, 41%, 19%) 80%,
```

```
      transparent 100%);
  }
  #radius-shape-1 {
    height: 220px;
    width: 220px;
    top: -60px;
    left: -130px;
    background: radial-gradient(#44006b, #ad1fff);
    overflow: hidden;
  }
  #radius-shape-2 {
    border-radius: 38% 62% 63% 37% / 70% 33% 67% 30%;
    bottom: -60px;
    right: -110px;
    width: 300px;
    height: 300px;
    background: radial-gradient(#44006b, #ad1fff);
    overflow: hidden;
  }
  .bg-glass {
    background-color: hsla(0, 0%, 100%, 0.9) !important;
    backdrop-filter: saturate(200%) blur(25px);
  }
</style>
<div class="container px-4 py-5 px-md-5 text-center text-lg-start my-5">
  <div class="row gx-lg-5 align-items-center mb-5">
    <div class="col-lg-6 mb-5 mb-lg-0" style="z-index: 10">
      <h1 class="my-5 display-5 fw-bold ls-tight" style="color: hsl(218, 81%, 95%)">
        The best application <br />
```

```
      <span style="color: hsl(218, 81%, 75%)">to track your expenses</span>
    </h1>
  </div>
  <div class="col-lg-6 mb-5 mb-lg-0 position-relative">
    <div id="radius-shape-1" class="position-absolute rounded-circle shadow-5-
strong"></div>
    <div id="radius-shape-2" class="position-absolute shadow-5-strong"></div>
    <div class="card bg-glass">
      <div class="card-body px-4 py-5 px-md-5">
        <form action="/signinform" method="POST">
          <!-- Email input -->
          <div class="form-outline mb-4">
            <input type="email" id="form3Example3" class="form-control" name="email"
required/>
            <label class="form-label" for="form3Example3">Email address</label>
          </div>
          <!-- Password input -->
          <div class="form-outline mb-4">
            <input type="password" id="form3Example4" class="form-control"
name="pass" required/>
            <label class="form-label" for="form3Example4">Password</label>
          </div>

          <!-- Submit button -->
          <button type="submit" class="btn btn-primary btn-block mb-4">
            Log in
          </button>
          <!-- Register buttons -->
          <div class="text-center">
```

```
                <a href="/signup">No account? Sign up here</a>
              </div>
            </form>
          </div>
        </div>
      </div>
    </div>
  </div>
 </section>
 <!-- Section: Design Block -->
</body>
</html>
```

signup.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>SignUp</title>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
   <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
```

u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8"
crossorigin="anonymous"></script>
</head>
<body>
<!-- Section: Design Block -->
<section class="background-radial-gradient overflow-hidden vh-100">
<style>
.background-radial-gradient {
background-color: hsl(218, 41%, 15%);
background-image: radial-gradient(650px circle at 0% 0%,
hsl(218, 41%, 35%) 15%,
hsl(218, 41%, 30%) 35%,
hsl(218, 41%, 20%) 75%,
hsl(218, 41%, 19%) 80%,
transparent 100%),
radial-gradient(1250px circle at 100% 100%,
hsl(218, 41%, 45%) 15%,
hsl(218, 41%, 30%) 35%,
hsl(218, 41%, 20%) 75%,
hsl(218, 41%, 19%) 80%,
transparent 100%);
}
#radius-shape-1 {
height: 220px;
width: 220px;
top: -60px;
left: -130px;
background: radial-gradient(#44006b, #ad1fff);
overflow: hidden;

```
    }
    #radius-shape-2 {
      border-radius: 38% 62% 63% 37% / 70% 33% 67% 30%;
      bottom: -60px;
      right: -110px;
      width: 300px;
      height: 300px;
      background: radial-gradient(#44006b, #ad1fff);
      overflow: hidden;
    }
    .bg-glass {
      background-color: hsla(0, 0%, 100%, 0.9) !important;
      backdrop-filter: saturate(200%) blur(25px);
    }
  </style>
 <script>
  function validate(){

if(document.getElementById("pass").value.trim()==document.getElementById("cpass").v
alue.trim())
         return(true);
      alert("Password Mismatch");
      return false;
  }
 </script>
  <div class="container px-4 py-5 px-md-5 text-center text-lg-start my-5">
    <div class="row gx-lg-5 align-items-center mb-5">
      <div class="col-lg-6 mb-5 mb-lg-0" style="z-index: 10">
        <h1 class="my-5 display-5 fw-bold ls-tight" style="color: hsl(218, 81%, 95%)">
```

```
          The best application <br />
          <span style="color: hsl(218, 81%, 75%)">to track your expenses</span>
        </h1>
      </div>
      <div class="col-lg-6 mb-5 mb-lg-0 position-relative">
        <div id="radius-shape-1" class="position-absolute rounded-circle shadow-5-
strong"></div>
        <div id="radius-shape-2" class="position-absolute shadow-5-strong"></div>
        <div class="card bg-glass">
          <div class="card-body px-4 py-5 px-md-5">
            <form onsubmit="return validate()" action="/signup" method="POST">
              <!-- 2 column grid layout with text inputs for the first and last names -->
              <div class="form-outline mb-4">
                <input type="text" id="form3Example1" placeholder="Enter your name"
class="form-control" name="uname" required/>
                <label class="form-label" for="form3Example1">Name</label>
              </div>
              <!-- Email input -->
              <div class="form-outline mb-4">
                <input type="email" id="form3Example3" placeholder="Enter your email
address" class="form-control" name="email" required/>
                <label class="form-label" for="form3Example3">Email</label>
              </div>
              <!-- Password input -->
              <div class="form-outline mb-4">
                <input type="password" id="pass" placeholder="Enter a password"
class="form-control" name="pass" required/>
                <label class="form-label" for="pass">Password</label>
              </div>
```

```
            <div class="form-outline mb-4">
              <input type="password" id="cpass" placeholder="Confirm password"
class="form-control" required/>
              <label class="form-label" for="cpass">Confirm Password</label>
            </div>
            <!-- Submit button -->
            <button type="submit" class="btn btn-primary btn-block mb-4">
              Sign up
            </button>
            <!-- Register buttons -->
            <div class="text-center">
              <a href="/signin">Already have an account? Log in here</a>
            </div>
          </form>
        </div>
      </div>
    </div>
  </div>
 </section>
 <!-- Section: Design Block -->
</body>
</html>
```

updateexpense.html

```
{% extends 'base.html' %}
{% block body %}
```

```
<div class="container">
  <div class="row">
    <div class="col-md-6">
      <h3 class="mt-5"><i>Edit expense:</i></h3>
      <form method='POST' action = '/edit'>
        <input type="hidden" name="id" value="{{acc['ID']}}" >
        <div class="form-group">
          <label>Date of expense</label>
          <input class="form-control" type="date" name="date" id="date"
value="{{acc['EDATE']}}">
        </div>
        <div class="form-group">
          <label>Expense name</label>
          <input class="form-control" type="text" name="expensename"
id="expensename" value="{{acc['ENAME']}}">
        </div>
        <div class="form-group">
          <label>Expense amount</label>
          <input class="form-control" type="text" name="amount" id="amount"
value="{{acc['EAMOUNT']}}">
        </div>
        <div class="form-group">
          <label>Category</label>
          <select class="form-control" name="category" id="category">
            <option value="household">Household</option>
            <option value="food">Food</option>
            <option value="entertainment">Entertainment</option>
            <option value="business">Business</option>
```

```
                <option value="other" selected>Other</option>
              </select>
            </div>
            <input class="btn btn-danger" type="submit" value="Update">
          </form>
        </div>
    </div>
</div>
{% endblock %}
```

Github link : https://github.com/IBM-EPBL/IBM-Project-23109-1659867336.git

Demo video link : expense-tracker-demo-video.mp4