

PERSONAL EXPENSE TRACKER APPLICATION

A PROJECT REPORT

Submitted by

NAVEEN KUMAR S [711719104055]

NEWTTON HABAKUK S [711719104057]

NIVETHITHA S [711719104061]

POOJA B [711719104063]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KGiSL INSTITUTE OF TECHNOLOGY, SARAVANAMPATTI

ANNA UNIVERSITY :: CHENNAI 600 025

ABSTRACT

The web application “Personal Expense Tracker Application” is developed to manage the daily expenses in a more efficient and manageable way. By using this application, we can reduce the manual calculations of the daily expenses and keep track of the expenditure. In this application, user can provide his income to calculate his total expenses per day and these results will be stored for each user. Each user will be required to register on the system at registration time, the user will be provided id, which will be used to maintain the record of each unique user. Expense Tracker application which will keep a track of Income-Expense of a user on a day-to-day basis. By using application or managing expense tracking will help to control unnecessary expenses. It will distribute your expenses in different categories suitable for the user. An expense history will also be provided in application. Our goal is to create an “Personal Expense Tracker Application” where user can be tracking all financial activities and view previous income and expense report. The application allows users to track their expenses daily, weekly, monthly, and yearly in terms of summary, bar graphs, and pie-charts. User can set their limit based on their monthly income, if entered limit exceed application will sends the alert email notification to the user with expense details. User entered information are stored in the IBM db2 cloud database and the application will fetch the data using the flask framework. Build application is dockerized, it provides the private port number to connect to the application. The expense tracker will help any organization to deal with all their expenses more efficiently. By using software for managing expense tracking will help to control unnecessary expenses. There are several benefits and advantages of using online expense trackers, expense tracking makes any organization run faster and smoother.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF TABLES	ii
1.	INTRODUCTION	1
	1.1 PROJECT OVERVIEW	1
	1.2 PURPOSE	2
2.	LITERATURE SURVEY	3
	2.1 EXISTING PROBLEM	3
	2.2 REFERENCES	3
	2.3 PROBLEM STATEMENT DEFINITION	5
3.	IDEATION AND PROPOSED SOLUTION	6
	3.1 EMPATHY MAP CANVAS	6
	3.2 IDEATION AND BRAINSTORMING	7
	3.3 PROPOSED SOLUTION	9
	3.4 PROBLEM SOLUTION FIT	11
4.	REQUIREMENT ANALYSIS	12
	4.1 FUNCTIONAL REQUIREMENTS	12
	4.2 NON-FUNCTIONAL REQUIREMENTS	13
5.	PROJECT DESIGN	14
	5.1 DATA FLOW DIAGRAM	14
	5.2 SOLUTION & TECHNICAL ARCHITECTURE	15
	5.3 USER STORIES	17
6.	PROJECT PLANNING AND SCHEDULING	18
	6.1 SPRINT PLANNING AND ESTIMATION	18
	6.2 SPRINT DELIVERY SCHEDULE	19
7.	CODING AND SOLUTIONING	20
	7.1 FEATURE 1	20
	7.2 FEATURE 2	24

8.	TESTING	29
	8.1 TEST CASES	29
	8.2 USER ACCEPTANCE TESTING	29
9.	RESULTS	30
	9.1 PERFORMANCE METRICS	30
10.	ADVANTAGES AND DISADVANTAGES	31
11.	CONCLUSION	32
12.	FUTURE SCOPE	33
13.	APPENDIX	34
	13.1 SOURCE CODE	34
	13.2 GITHUB AND PROJECT DEMO	114
14.	REFERENCES	115

KEYWORDS

1. Add expense
2. Limit calculation
3. Alert Email
4. IBM DB2 cloud database
5. Daily, Monthly, Yearly Expense report
6. Waterfall Model

CHAPTER 1

INTRODUCTION

Personal Expense Tracker Application is designed to keep a track of Income-Expense of an organization on a day-to-day basis. This application divides the Income based on daily expenses. If exceed monthly expense, system will calculate income and sent the notification to users registered mail with expense details. Daily expense tracking System will generate report at the end of day, month, year to show Income-Expense graph. User can change their expenses details by using edit and delete features in the application. Expense management application analyzes overall expenses, identifies cost-saving opportunities, and controls excessive spending. “Personal Expense Tracker Application” is an application where one can enter their daily expenses and end of the day, they know their expenses in charts. An expense history will also be provided in application. If limit exceed application will automatically send the notification to the user with expenses details.

1.1 PROJECT OVERVIEW

“Personal Expense Tracker Application” is developed to manage the daily expenses in a more efficient and manageable way. By using this application, we can reduce the manual calculations of the daily expenses and keep track of the expenditure. In this application, user can provide his income to calculate his total expenses per day, month, year and these results of each user information will be stored in the IBM db2 cloud database.

Every purchase made are added in the personal expense tracker. If the limit set is exceeded a notification will be sent to the users Mail with the daily expense’s details and incomes that are help to tracking system of the application. Expense Tracking takes a crucial role in managing the expenses of our daily life and business organizations. Expense tracking will bring in several advantages for an organization. That are helpful for the stake-holders in processes of expense. The expense tracker will help any organization to deal with all their expenses more efficiently.

1.2 PURPOSE

By using application or managing expense tracking will help to control unnecessary expenses. It will distribute your expenses in different categories suitable for the user. An expense history will also be provided in application. Our goal is to create an “Personal Expense Tracker Application” where user can be tracking all financial activities and view previous income and expense report.

- Add Expense and Income.
- User can update or delete records.
- User can set their own limits based on their monthly income and budget.
- User can easily review the reports daily, weekly, monthly and yearly.
- User can get notification if monthly limit exceed.

CHAPTER 2

LITERATURE SURVEY

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 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. There can be many disadvantages of using a manual accounting system.
- Accounting, for any business, can be a complex undertaking. A manual accounting system requires you to understand the accounting process in a way that may be unnecessary with a computerized accounting system. This can be an advantage or a disadvantage, depending on the person doing the bookkeeping; often, a specially trained professional is needed to ensure that accounting is done properly. Unraveling the complexity of your financial records by hand may be time consuming. Since it takes time to generate reports.

2.2 REFERENCES

I) Title: Expenditure Management System

Authors: Dr. V. Geetha, G. Nikhitha, H. Sri Lasya Dr. C.K.Gomathy

Description: Expense Tracker is an everyday expense control application designed to track effortlessly and efficiently each day costs. This helps us to get rid of the need of paper responsibilities that systematically maintains information. This device can be utilized by any individual to govern their income expenditure from each day to annual basis and to hold an eye on their spending, Including the person to whom the payments were made and the purpose for the payment. On a weekly, monthly, and yearly basis, details of expenses will be displayed in the form of a pie chart. It aids us in remembering and adding information about what money we receive from others and what costs or payments we must make on a given date or month. We have

categories in the expense tracker such as add expense, monthly expenses, add new expense, and so on. It gives the daily remainder about the savings we need to do. Year: 3 March, 2022

II) Title: Expense Tracker : A Smart Approach to Track Everyday Expense

Authors: Hrithik Gupta, Anant Prakash Singh, Navneet Kumar and J. Angelin Blessy

Description: Expense Tracker is a day-to-day expense management system designed to easily and efficiently track the daily expenses of unpaid and unpaid staff through a computerized system that eliminates the need for manual paper tasks that systematically maintains records and easily accesses data stored by the user. End users with window running devices can use this software. The language databases we use to develop this system are Java (Apache Netbeans 11.3) and MySQL Workbench 8.0 CE. This application is a GUI (Graphics User Interface) based application. If you are a window user, you can download the application. Year: December 25, 2020

III) Title: Daily Expense Tracker

Authors: Abishek Hagawane, Roshan Gopalghare, Prathmesh Isawe, Mrunal Aware

Description: Daily Expense tracker is based on income and expense tracking system. This project offers some opportunities that will help user to sustain all financial activities like automated dairy. The main aim of the project is to create a faster, easier and smooth system between the expense and the income. So, for the better expense tracking we built Expense Tracker is very simple and attractive, Which makes it easy for the user to understand and use it. A daily expense tracker is best way to record your financial data. Most of the people cannot track their money. They find it difficult to save their money. On one way of thinking they face money crisis, in such case Daily Expense Tracker can help people to track their income and expense. This project will save time and lead a healthy financial lifestyle. Year: 07 June 2022

2.3 PROBLEM STATEMENT DEFINITION

Personal finance entails all the financial decisions and activities that a Finance app makes your life easier by helping you to manage your finances efficiently. A personal finance app will not only help you with budgeting and accounting but also give you helpful insights about money management. Personal finance applications will ask users to add their expenses and based on their expense's wallet balance will be updated which will be visible to the user. Also, users can get an analysis of their expenditure in graphical forms. They have an option to set a limit for the amount to be used for that particular month if the limit is exceeded the user will be notified with an email alert through SendGrid software.

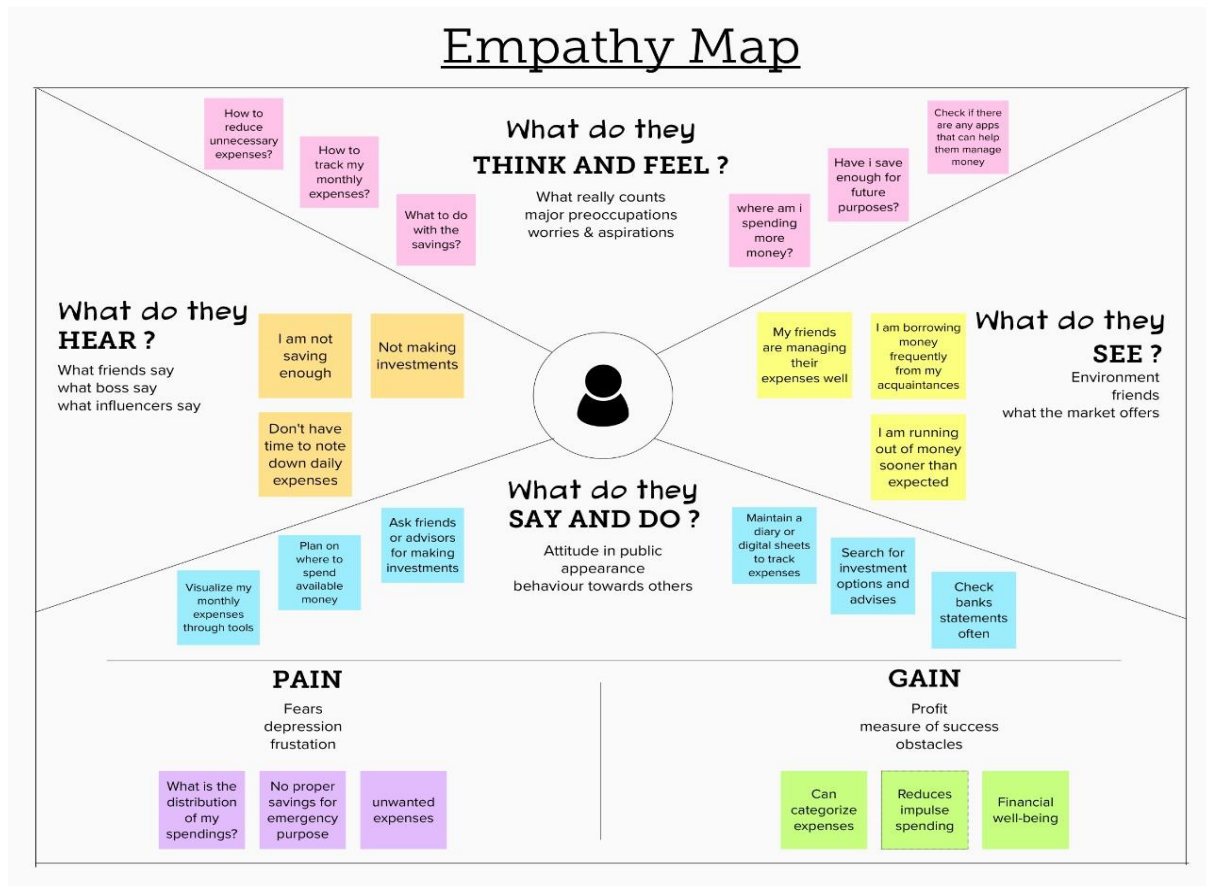
Who does the problem affect?	Working individuals, students and budget conscious consumers.
What are the boundaries of the problem?	Limited features to provide for expense tracking.
When does this issue occur?	When people are not able to track their expenses properly.
Where is the issue occurring?	In daily life of employees as well as students.
Why is it important that we fix the problem?	By solving this issue those people getting regular wages can track their expenses and avoid unwanted expenses.

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is useful to solve the user problems in the application. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges. Effective expense tracking and reporting to avoid conflict. As a project manager or business owner, you can set clear policies for the expense types and reimbursement limits to avoid misunderstandings are about costs. Tracking the project expenses by asking team members to provide receipts is helpful to avoid conflict and maintain compliance also. An excellent reporting mechanism is extremely helpful to support the amount to be reimbursed to your team and also invoicing to your customer




3.2 IDEATION & BRAINSTORMING

Brainstorm & Idea Prioritization:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Template




Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

[Share template feedback](#)



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1


Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes


PROBLEM


Developing a application help the user to get tracking of monthly expenses and send alerts about spending expenses





Key rules of brainstorming


To run a smooth and productive session


 Stay in topic.

 Encourage wild ideas.

 Defer judgment.

 Listen to others.

 Go for volume.

 If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

⌚ 10 minutes

Newton Habakuk S

Based on data to enter the spending	Add income and expenses	Personalized account that showing money
Add a reminder and get notified	List income and expenses	Warning for budget
	Remember to save	

Nivenitha S

Manager to use the dashboard	Set budget
Personalized reports	Personalized report and get notified
Get notified when report is generated	The dashboard

Naveen Kumar S

Create a monthly view of the money	Get monthly report as pdf or more view	Generate monthly report
Filter the budgeted personally	Personalized view of the money as a report to generate monthly	Personalized view of the money as a report to generate monthly
	Give a table view of the data	

Pooja B

Personalized reports	Generate the reports	To remind the user to enter the data regularly
Personalized reports	Remind for data entry	Alerts and notifications to user
	Rich dashboard report	

3 Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and split it up into smaller sub-groups.

⌚ 20 minutes

```

graph TD
    A((Track your income and expenses)) --- B[Get a monthly report]
    A --- C[Add income and expenses]
    A --- D[Shows cash spend]
    A --- E[Set budget]
    A --- F[Send user to enter the spendings]
    F --- G[Send a message to the user if the bill of the month is received]
    F --- H[Remind the user to enter the spendings]
    F --- I[Remind the user to enter the spendings]
  
```

Step-3: Idea Prioritization

4 Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⌚ 20 minutes

Importance

If most of these tasks would get done without any difficulty or cost, which would have the most positive impact?

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 PROPOSED SOLUTION

To reduce manual calculations, we propose an application. This application allows users to maintain a digital automated diary. Each user will be required to register on the system at registration time, the user will be provided id, which will be used to maintain the record of each unique user. Expense Tracker application which will keep a track of Income-Expense of a user on a day-to-day basis. By using application or managing expense tracking will help to control unnecessary expenses. It will distribute your expenses in different categories suitable for the user. An expense history will also be provided in application. Our goal is to create an “Personal Expense Tracker Application” where user can be tracking all financial activities and view previous income and expense report.

- Add Expense and Income.
- User can update or delete records.
- User can set their own limits based on their monthly income and budget.
- User can easily review the reports daily, weekly, monthly and yearly.
- User can get notification if monthly limit exceeds.
- User information's are stored in the cloud database IBM db2.

S.NO.	Parameter	Description
1.	Problem Statement	To manage the expenses of an individual in an efficient and manageable manner, as compared to the traditional way of expense tracking.
2.	Scalability of the Solution	This application can handle large number of users and data with high performance and security. This application can adapt for both large-scale and small-scale purposes. Easily available in all kinds of devices.

3.	Idea / Solution description	Daily expense management system which is specially designed for non-salaried and salaried personnel for keeping track of their daily expenditure with easy and effective way through computerized system which tends to eliminate manual paper works. Personal finance applications will ask users to add their expenses and based on their expense's wallet balance will be updated which will be visible to the user. They have an option to set a limit for the amount to be used for that particular month if the limit is exceeded the user will be notified with an email alert.
4.	Novelty / Uniqueness	The user gets notified when their expense exceeds the limit and also it reminds the user when they forgot to make entry. Tracking expenses through SMS. Data analytics on expenses.
5.	Social Impact / Customer Satisfaction	The application should be able to generate reports of their spending and notify users if they have exceeded their budget. It is designed to be dynamic to produce the prediction. It also provides users' personal information, their income as well as their expenses. This application can create awareness among common people about finance and stuffs. This application also helps user to be financially responsible. It Reduces time rather than entering details manually.
6.	Business Model (Revenue Model)	This Application is provided for free of cost. But It will have some advertisement. In premium version there is no advertisement and contains some additional features.

3.4 PROBLEM SOLUTION FIT

“Personal Expense Tracker Application” is developed to manage the daily expenses in a more efficient and manageable way. User problems are solved based on the daily, monthly, yearly expenses, then email will be sent to the user if limit is exceeding. Problem solution fit explain based on the daily, monthly and yearly expenses.

Define CS, fit

Explore AS,

<div>1. CUSTOMER SEGMENT(S)</div> <ul style="list-style-type: none"> Working Individuals Entrepreneur Budget conscious consumer 	<div>6. CUSTOMER CONSTRAINTS</div> <ul style="list-style-type: none"> Internet Access Device to access the application Data privacy Cost of existing application Trust 	<div>5. AVAILABLE SOLUTIONS</div> <ul style="list-style-type: none"> Expense daily or Excel sheet <p>PROS: Have to make a note daily which helps to be constantly aware</p> <p>CONS: Inconvenient ,takes a lot of time</p>
--	---	---

<div>2. JOBS-TO-BE-DONE / PROBLEMS</div> <ul style="list-style-type: none"> To keep track of money lent To keep track of daily transaction Alert when a threshold limit is reached 	<div>9. PROBLEM ROOT CAUSE</div> <div>RC</div> <ul style="list-style-type: none"> Reckless spending Indecisive about the finances Procrastination Difficult to maintain a note of daily spending 	<div>7. BEHAVIOUR</div> <div>BE</div> <ul style="list-style-type: none"> Make a note of the expenses on a regular basis Completely reduce spending all of the savings Make use of online tool to interpret monthly expense patterns
---	--	--

<div>3. TRIGGERS</div> <ul style="list-style-type: none"> Excessive spending No money in case of emergency 	<div>10. YOUR SOLUTION</div> <p>Creating an application to manage the expenses of an individual in an efficient and manageable manner, as compared to traditional methods</p>	<div>8. CHANNELS of BEHAVIOUR</div> <div>8.1ONLINE</div> <p>Maintain excel sheet and use visualizing tools</p>							
<div>4. EMOTIONS: BEFORE / AFTER</div> <table> <tr> <th>Before</th> <th>After</th> </tr> <tr> <td>Anxious</td> <td>Confident</td> </tr> <tr> <td>Confused</td> <td>Composed</td> </tr> <tr> <td>Fear</td> <td>Calm</td> </tr> </table>		Before	After	Anxious	Confident	Confused	Composed	Fear	Calm
Before	After								
Anxious	Confident								
Confused	Composed								
Fear	Calm								

Focus on J&P, tab into BE

Focus on J&P, tab into BE

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements. 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 Gmail Registration through Email Account
FR-2	Category	This application shall allow users to add categories of their expenses.
FR-3	Calendar	Personal expense tracker application must allow user to add the data to their expenses.
FR-4	Graphical Representation	This application should graphically represent the expense in the form of report.
FR-5	Report Generation	Graphical representation of report must be generated.
FR-6	Expense Limit	User can set their own limit based on their monthly budget
FR-7	Alert Email	If limit exceed then application will send alert email to the user with expense

4.2 NON-FUNCTIONAL REQUIREMENTS

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Helps to keep an accurate record and track of their income and expenses easily.
NFR-2	Security	We save the password in the encrypted form so it will add more secure to the application user.
NFR-3	Reliability	Each data record is stored on a well-built efficient database schema. There is no risk of data loss.
NFR-4	Performance	Expense kinds include categories and an option. The system's throughput is boosted because to the lightweight database support.
NFR-5	Availability	User can able to access the application with the help of the internet throw the web browser.
NFR-6	Scalability	The ability to appropriately handle increasing demands.

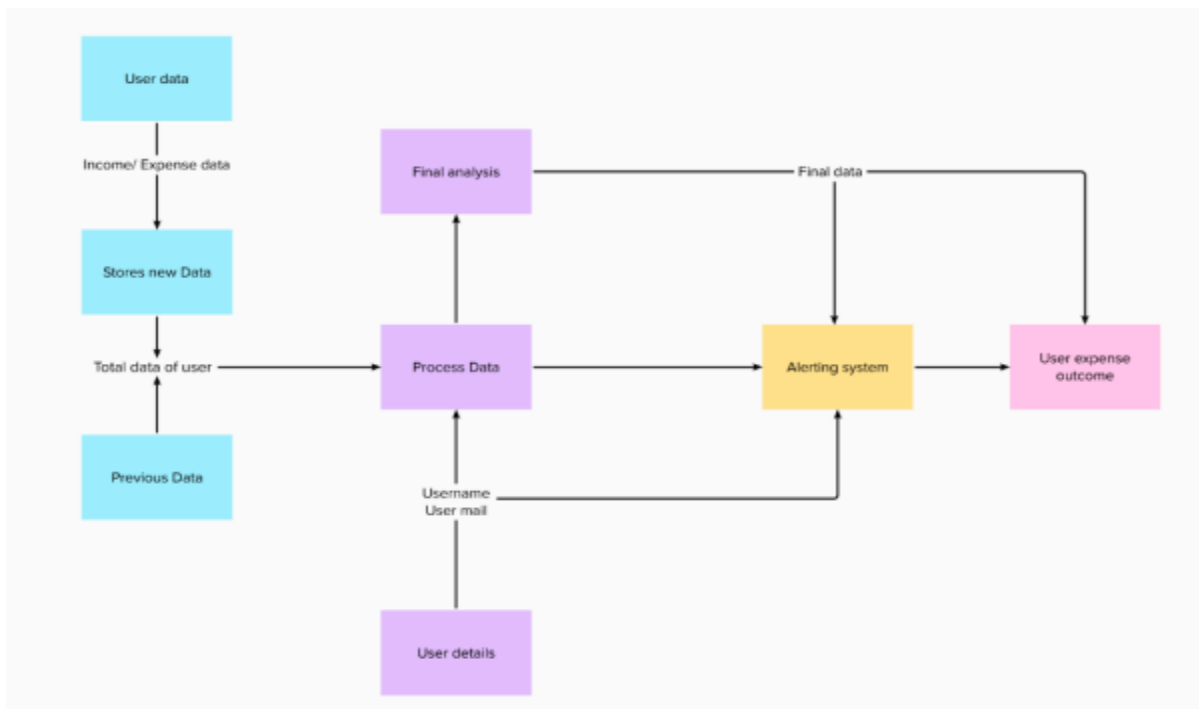
CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

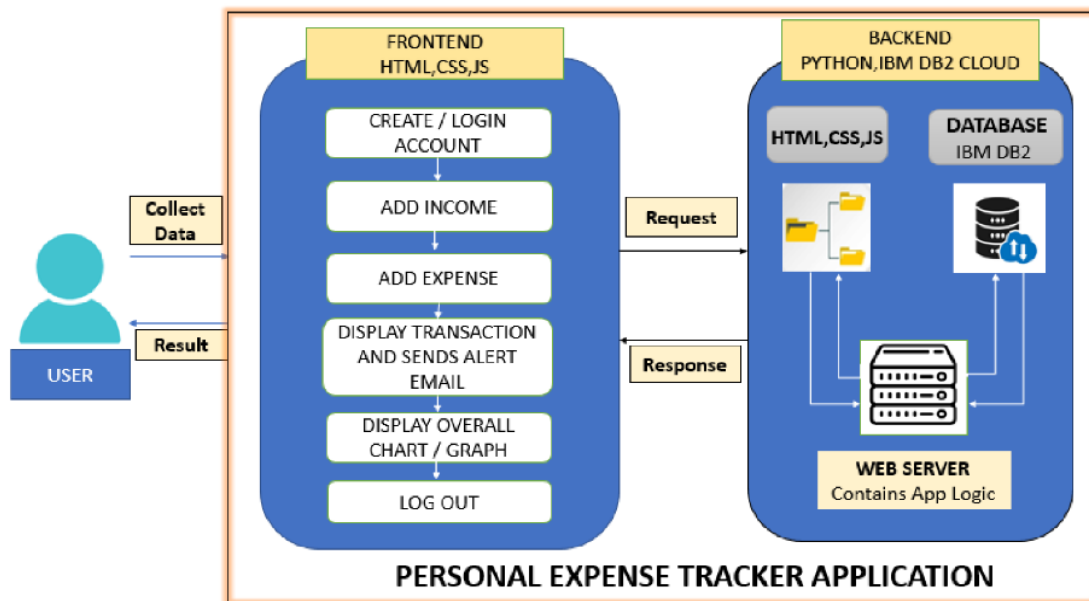
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

User information's are considered as a data, it will be stored in IBM db2 cloud database and flask framework is used to communicate with the cloud database to fetch the information for the user.



5.2 SOLUTION & TECHNICAL ARCHITECTURE

“Personal Expense Tracker Application” is developed to manage the daily expenses in a more efficient and manageable way. By using this application, we can reduce the manual calculations of the daily expenses and keep track of the expenditure. In this application, user can provide his income to calculate his total expenses per day, month, year and these results of each user information will be stored in the IBM db2 cloud database.

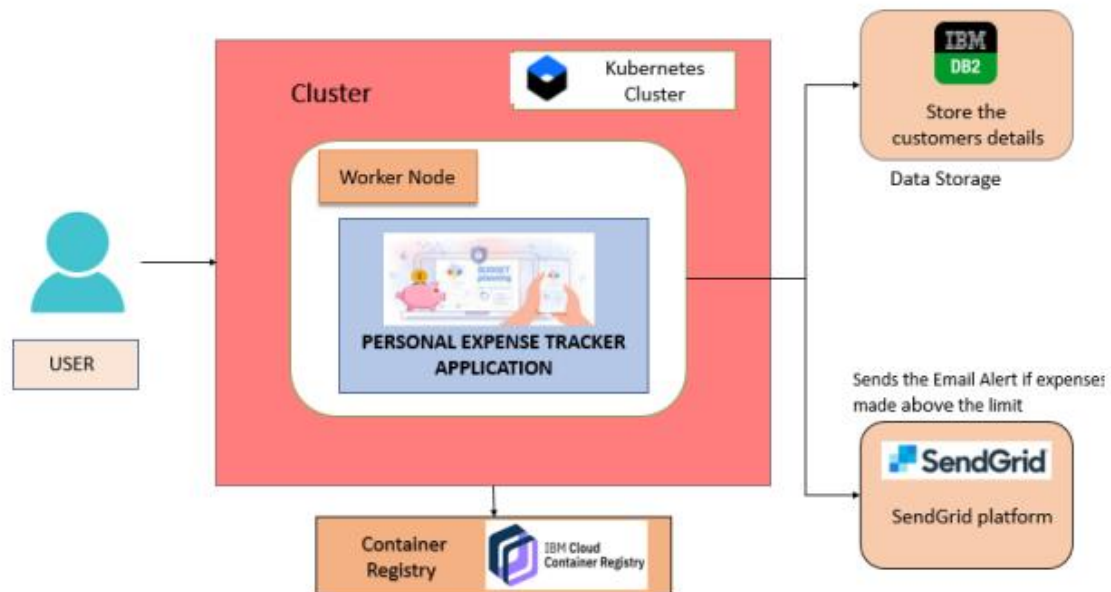


SOLUTION ARCHITECTURE

User will enter the information in the application that will be stored in IBM db2 cloud database.” Personal Expense Tracker Application” is dockerized to get the image file and port number for the application, then Kubernetes cluster is used to deploy the application in the IBM cloud. If limit exceed then application will send alert email to the user with expense details through SendGrid platform.

Technologies Used:

- IBM Cloud,
- HTML
- CSS
- JavaScript
- IBM Cloud Object Storage
- Python-Flask
- Kubernetes
- Docker
- IBM DB2
- IBM Container Registry
- SendGrid



TECHNICAL ARCHITECTURE

5.3 USER STORIES

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
User	Homepage	USN-1	As a user, I can visit the page to know about the personal expense trackers services and contact details	I can know about the services and functionality available in the application	High	Sprint-1
	Registration	USN-2	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
	Login	USN-3	As a user, I can register for the application through Gmail	I can access my account/dashboard	High	Sprint-1
	Forgot password	USN-4	As a customer, I can reset my password by this option in case I forgot my old password.	I get access to my account again	High	Sprint -1
	Dashboard	USN-5	As a user, I can log into the application by entering email & password	I get all the info needed in my dashboard.	Low	Sprint-2
	Add expenses	USN-6	User can add their expenses	I get expense details	Medium	Sprint-2
	Expense History	USN-7	User can see the expenses details and overview graph	I get overview history	High	Sprint-3
	Report	USN-8	As a user, I can see the daily, monthly , yearly expense report with the overview graph in piechart manner	I get better understanding	Medium	Sprint-3
	Limit and Email Alert	USN-9	User can set their own limit if it exceed then alert email will be sent to the user with expense details	I can set their own limit	High	Sprint-4

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

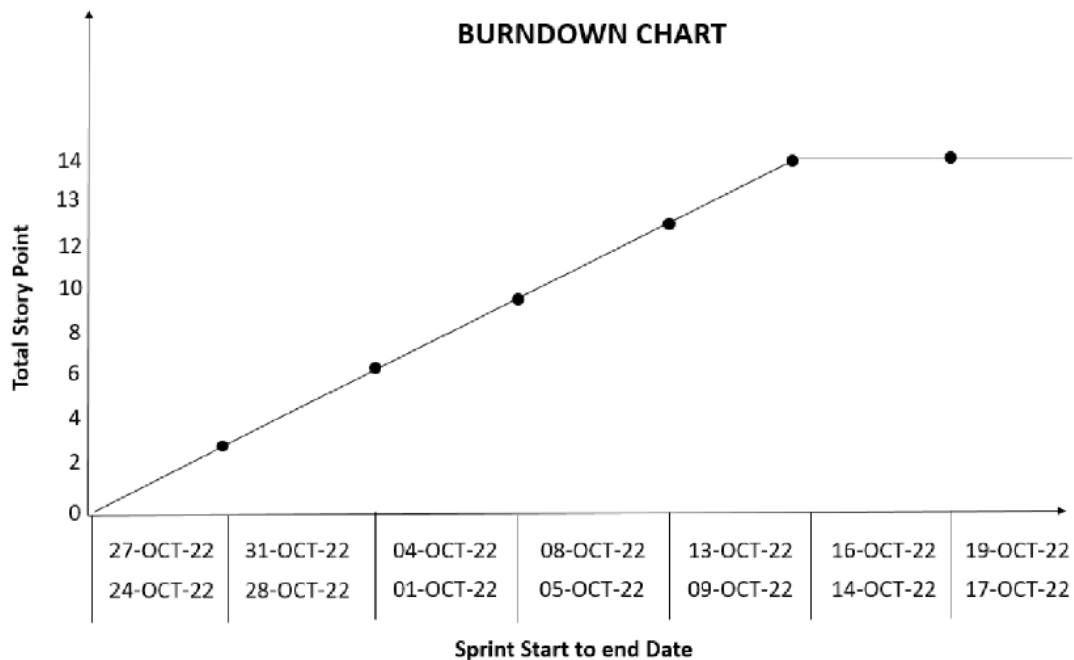
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Homepage	USN-1	AS a user I can view the index page to see the about of the Expense tracker	10	High	Newton Habakuk
Sprint-1	Registration	USN-2	As a User, I need to register user id and password for every worker over there in municipality	10	High	Nivethitha
Sprint-1	Login	USN-3	As a user, I need to login with user id and password to get in to the website	10	High	Naveen Kumar
Sprint-2	Dashboard	USN-4	As a User, I will follow Co-Admin's instruction to reach the filling bin in short roots and save time	20	Low	Nivethitha, Newton Habakuk
Sprint-3	Add Expenses	USN-5	As a User I will add my expense throughout the month I spend on	20	Medium	Pooja
Sprint-3	Total Expense Graph	USN-6	As a User I can view my expense in a graph of overview of the expense I spend.	20	Medium	Naveen ,Pooja
Sprint-4	Limit, email alert,Deployment in cloud	USN-7	As a User I can access the cloud to store my data of expense	20	High	Newton Habakuk ,Nivethitha ,Pooja Naveen kumar

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	23 Oct 2022	28 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	30 Oct 2022	04 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	06 Nov 2022	11 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	13 Nov 2022	18 Nov 2022	20	19 Nov 2022

6.3 REPORTS FROM JIRA

This graph denotes the whole spring planning and project development phase completion report in the detailed manner by mentioning the date and levels in every phase of development.



CHAPTER 7

CODING & SOLUTIONING

7.1 FEATURE 1

7.1.1 ADD EXPENSE

In “Personal Expenses Tracker Application”, we can add daily, monthly and yearly expenses to maintain a report to avoid spending too much of money in unwanted things and also, we can save more money for our future use.

In this code, user will enter the details of their expense like date, expense name, category, amount and pay mode. These details are stored as an information in the IBM db2 cloud database and help of the cloud database we can store large amount of data of user.

CODE:

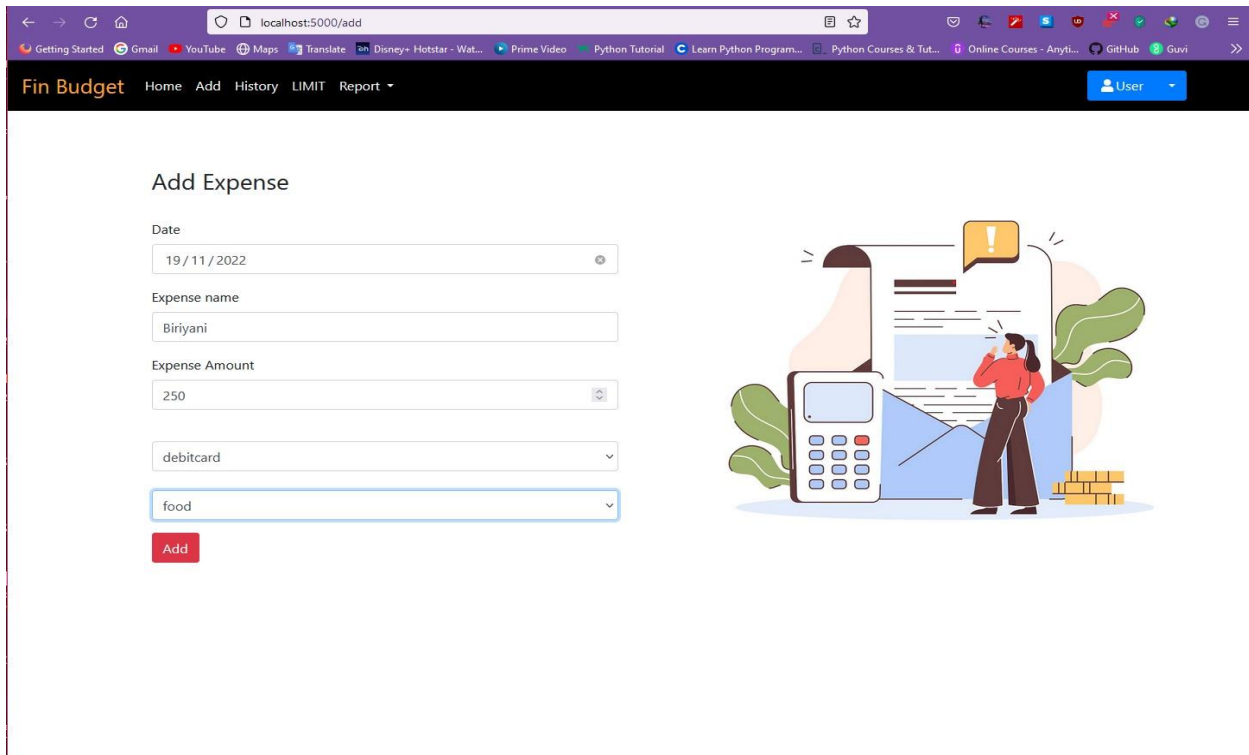
```
@app.route("/add")
def adding():
    return render_template('add.html')
@app.route('/addexpense',methods=['GET', 'POST'])
def addexpense():
    global user_email
    que = "SELECT * FROM expenses where id = ? ORDER BY 'dates' DESC"
    stm = ibm_db.prepare(connection, que)
    ibm_db.bind_param(stm, 1, session['email'])
    ibm_db.execute(stm)
    dictionary=ibm_db.fetch_assoc(stm)
    expense=[]
    while dictionary != False:
        exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],di
ctinary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"])
        expense.append(exp)
        dictionary = ibm_db.fetch_assoc(stm)
    i=len(expense)+1,   idx=str(i)
    dates = request.form['date']
    expensename = request.form['expensename']
    amount = request.form['amount']
    paymode = request.form['paymode']
```

```

category = request.form['category']
query = "INSERT INTO expenses VALUES (?, ?, ?, ?, ?, ?, ?);"
stmt = ibm_db.prepare(connection, query)
ibm_db.bind_param(stmt, 1, session['email'])
ibm_db.bind_param(stmt, 2, dates)
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.bind_param(stmt, 7, idx)
ibm_db.execute(stmt)
print(dates + " " + expensename + " " + amount + " " + paymode + " " + category)
return redirect("/display")

```

OUTPUT



The screenshot shows a web browser at localhost:5000/add. The page title is 'Fin Budget'. The navigation bar includes 'Home', 'Add', 'History', 'LIMIT', and 'Report'. A user profile 'User' is logged in. The main content area is titled 'Add Expense' and contains the following form fields:

- Date: 19 / 11 / 2022
- Expense name: Biryani
- Expense Amount: 250
- Paymode: debitcard
- Category: food
- An 'Add' button in a red box.

To the right of the form is an illustration of a person in a red shirt and black pants standing next to a large document with an exclamation mark, a calculator, and a stack of gold coins.

By using ibm_db library, we are fetching the information from the cloud database, then writing the query to store the entered information in the IBM db2 database. Next, the ibm_db2 starts execution to insert the data into the database, after insertion application will automatically redirect you to the display page. There you can view the overall chart, entered expenses details, also update and delete will be done in display page.

7.1.2 EXPENSE OVERALL REPORT

In this application, the overall report will be generated with the help of user entered expense information. Application will calculate the daily, monthly and yearly expenses by the user information and show it in the graph manner.

CODE:

```
@app.route("/year")
def year():
    query = "SELECT MONTH(dates) as DATES, SUM(amount) as AMOUNT
FROM expenses WHERE id=? AND YEAR(dates)= YEAR(now()) GROUP BY
MONTH(dates);"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1,session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    texpanse=[]
    while dictionary != False:
        exp=(dictionary["DATES"],dictionary["AMOUNT"])
        texpanse.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    print(texpanse)
    query = "SELECT * FROM expenses WHERE id = ? AND YEAR(dates)=
YEAR(now());"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1,session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    expense=[]
    while dictionary != False:
        exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],di
ctionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary[
"IDX"])
        expense.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    total=0
    t_food=0
    t_entertainment=0
    t_business=0
    t_rent=0
    t_EMI=0
```

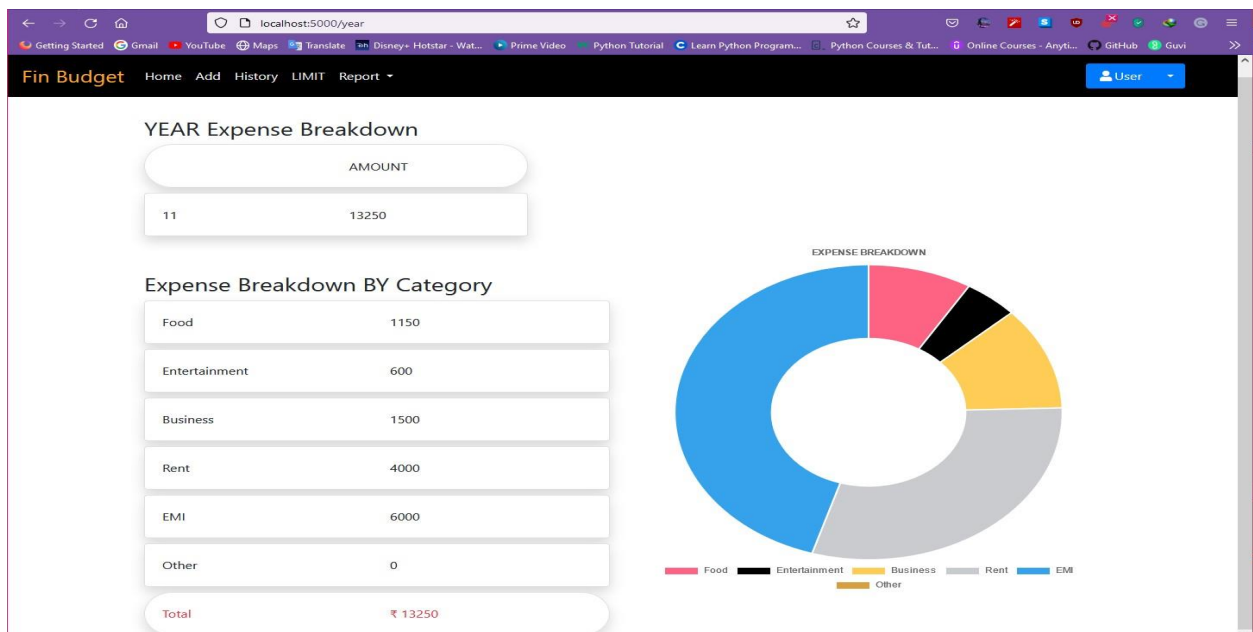
```

t_other=0
for x in expense:
    total += x[3]
    if x[5] == "food":
        t_food += x[3]
    elif x[5] == "entertainment":
        t_entertainment += x[3]
    elif x[5] == "business":
        t_business += x[3]
    elif x[5] == "rent":
        t_rent += x[3]
    elif x[5] == "EMI":
        t_EMI += x[3]
    elif x[5] == "other":
        t_other += x[3]

return render_template("today.html", texpense = texpense, 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 )

```

OUTPUT



In “personal expenses tracker application” they will calculate the expenses and show it in a graph manner structured view to the user. Every category of amount is calculated one by one in the flask and it will fetch data from the cloud database to generate the overall report for the user.

7.2 FEATURE 2

7.2.1 SETTING LIMIT BASED ON MONTHLY BUDGET

In “Personal Expenses Tracker Application”, User can enter their own limit based on their monthly budget to get the alert mail from the application if the entered limit is exceeded.

In this code, first it will render to limit.html page and then user can enter their limit for month based on the monthly budget. The entered amount is stored in the limit table to maintain the information's in cloud database. It will select all the rows in the limits to fetch the last entered amount in the limit page.

CODE :

```
@app.route("/limit" )

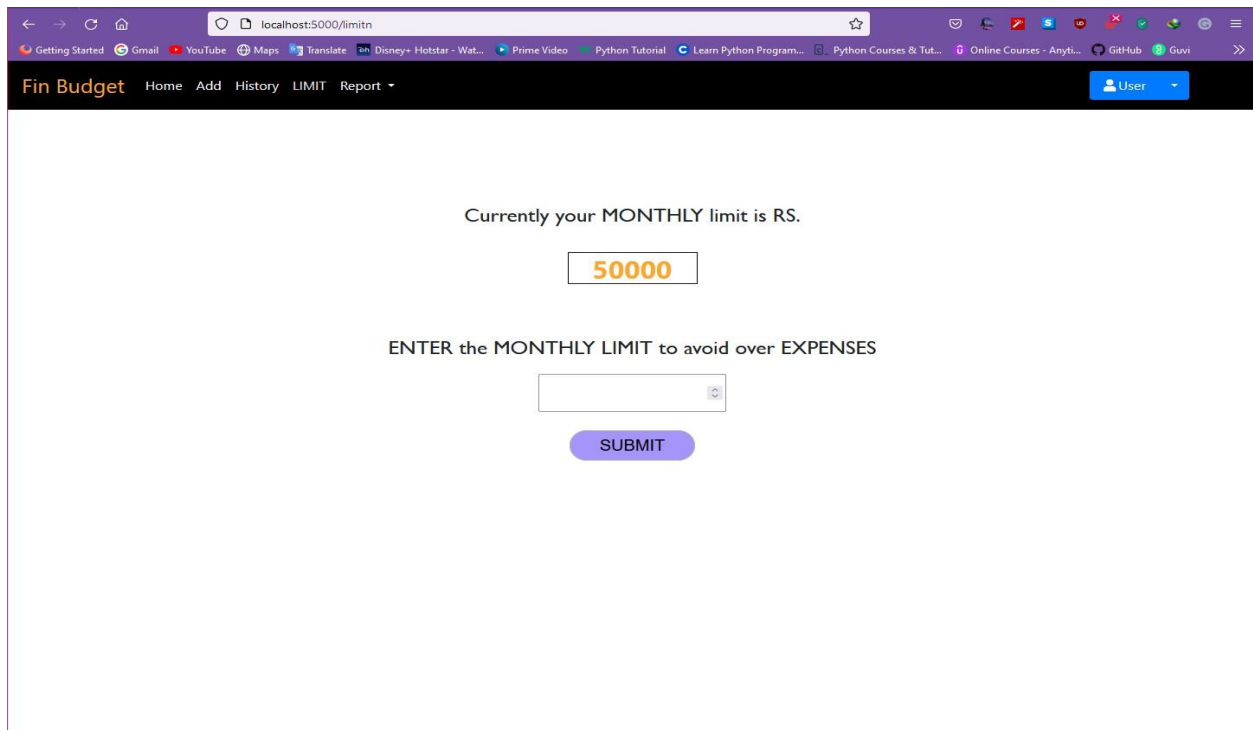
def limit():
    return render_template('limit.html')
@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
    que = "SELECT * FROM limits where id = ? ;"
    stm = ibm_db.prepare(connection, que)
    ibm_db.bind_param(stm, 1, session['email'])
    ibm_db.execute(stm)
    if request.method == "POST":
        dictionary=ibm_db.fetch_assoc(stm)
        expense=[]
        while dictionary != False:
            exp=(dictionary['ID'],dictionary['NUMBER'],dictionary['IDX'])
            expense.append(exp)
            dictionary = ibm_db.fetch_assoc(stm)
        i=len(expense)+1,idx=str(i)
        number= request.form['number']
        query = "INSERT INTO limits VALUES(?,?,?)"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, session['email'])
        ibm_db.bind_param(stmt, 2, number)
        ibm_db.bind_param(stmt, 3, idx)
        ibm_db.execute(stmt),
        return redirect('/limitn')
```

```

@app.route("/limitn")
def limitn():
    query = "SELECT max(IDX) as IDX FROM limits where id=?;"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.execute(stmt)
    dictionary = ibm_db.fetch_assoc(stmt)
    expense=[]
    while dictionary != False:
        exp=(dictionary["IDX"])
        expense.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    k=expense[0]
    que = "SELECT NUMBER FROM limits where id=? and idx=?"
    stmt = ibm_db.prepare(connection, que)
    ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.bind_param(stmt, 2, k)
    ibm_db.execute(stmt)
    dictionary = ibm_db.fetch_assoc(stmt)
    s=expense[0]
    return render_template("limit.html" , y= s)

```

OUTPUT



Currently your MONTHLY limit is RS.

50000

ENTER the MONTHLY LIMIT to avoid over EXPENSES

SUBMIT

If the user entered limit is calculated by entering the add expense details. If it exceeds, then email will send to the user with the expense's notification message. The entered limit amount will be shown in the limit.html page for the user.

7.2.2 SENDING ALERT EMAIL TO THE USER

Python, being a powerful language don't need any external library to import and offers a native library to send emails- "SMTP lib". "Smtplib" creates a Simple Mail Transfer Protocol client session object which is used to send emails to any valid email id on the internet. Flask-Mail is configured through the standard Flask config API.

CODE :

```
import os

from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail

message = Mail(

    from_email=os.environ.get('SENDER_EMAIL'),

    to_emails='nivethithashanmuganathan5@gmail.com',

    subject='Sending with Twilio SendGrid is Fun',

    html_content='<strong>and easy to do anywhere, even with Python</strong>')

try:

    sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))

    response = sg.send(message)

    print(response.status_code)

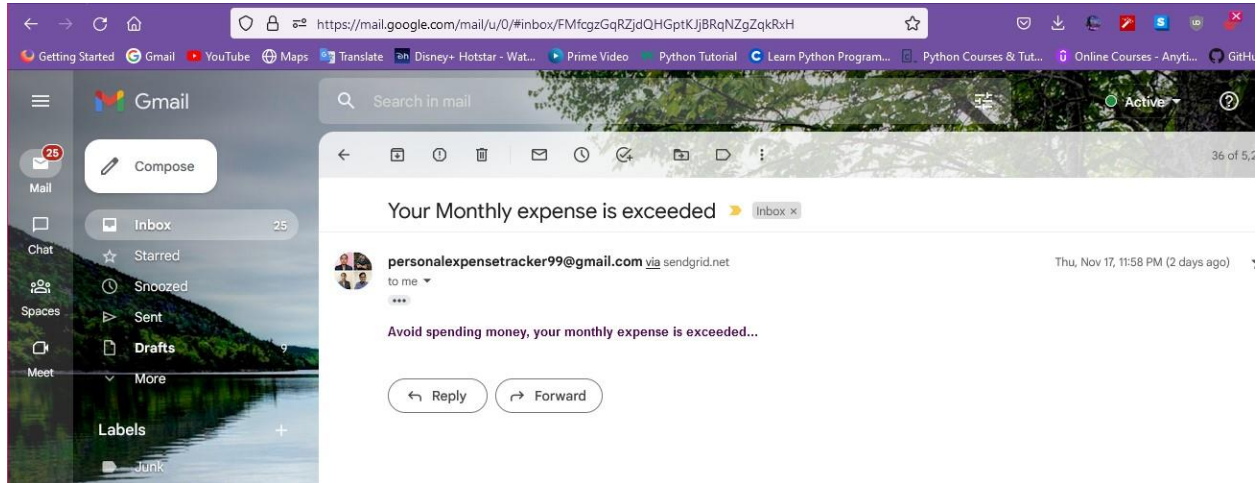
    print(response.body)

    print(response.headers)

except Exception as e:

    print(e)
```

OUTPUT



In “Personal Expenses Tracker Application”, if entered limit is exceed then SendGrid will send the alert email to the user with expenses detail message and it automatically response to the user by the help of SendGrid software.

7.3 DATABASE SCHEMA

7.3.1 REGISTER TABLE DATABASE SCHEMA

Name	Data type	Nullable	Length	Scale
USERNAME	VARCHAR	N	210	0
EMAIL	VARCHAR	N	100	0
PASSWORD	VARCHAR	N	300	0

7.3.2 EXPENSE TABLE DATABASE SCHEMA

IBM Db2 on Cloud

Load DataLoad HistoryTablesViewsIndexesAliasesMQTsSequencesApplication objects

Find schemas or tables

Refresh

SQL

Schemas

Tables

New table +

Filter

Sort

Close

EXPENSES

VJX02808

...

LIMITS

VJX02808

...

REGISTER

VJX02808

...

USERDETAILS

VJX02808

...

Total: 4, selected: 0

Table definition

EXPENSES

No statistics available.

Name	Data type	Nullable	Length	Scale	
ID	VARCHAR	N	320	0	👁
DATES	DATE	N	4	0	👁
EXPENSENAME	VARCHAR	N	320	0	👁
AMOUNT	INTEGER	N		0	👁
PAYMODE	VARCHAR	N	32	0	👁
CATEGORY	VARCHAR	N	32	0	👁
IDX	INTEGER	N		0	👁

View data

7.3.3 LIMITS TABLE DATABASE SCHEMA

IBM Db2 on Cloud

Load DataLoad HistoryTablesViewsIndexesAliasesMQTsSequencesApplication objects

Find schemas or tables

Refresh

SQL

Schemas

Tables

New table +

Filter

Sort

Close

EXPENSES

VJX02808

...

☒ LIMITS

VJX02808

...

REGISTER

VJX02808

...

USERDETAILS

VJX02808

...

Total: 4, selected: 1

Table definition

LIMITS

No statistics available.

Name	Data type	Nullable	Length	Scale	
ID	VARCHAR	N	320	0	👁
NUMBER	INTEGER	N		0	👁
IDX	INTEGER	N		0	👁

View data

CHAPTER 8

TESTING

8.1 TEST CASES

Software Testing Type is a classification of different testing activities into categories, each having, a defined test objective, test strategy, and test deliverables. The goal of having a testing type is to validate the Application under Test for the defined Test Objective.

For instance, the goal of Accessibility testing is to validate the AUT to be accessible by disabled people. So, if your Software solution must be disabled friendly, you check it against Accessibility Test Cases.

A test case is a set of actions performed on a system to determine if it satisfies software requirements and functions correctly. The purpose of a test case is to determine if different features within a system are performing as expected and to confirm that the system satisfies all related standards, guidelines and customer requirements. The process of writing a test case can also help reveal errors or defects within the system.

8.2 USER ACCEPTANCE TESTING

- User Acceptance Testing is a level of the software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.
- Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.
- User Acceptance Testing is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.
- It is probably the most important test phase of all as it is where we confirm that the system is fit for purpose to the business stakeholders.
- In this Web application, the customer's acceptance is been monitored and it is been put into usage.

CHAPTER 9

RESULTS

Building a Personal expense tracker to track a user's expense and keep them in an economic track in spending the money. User's monthly expense is tracked to spend the money correctly on the things that are bought. Daily, Monthly and Yearly Analysis are generated in the form of pie chart.

Every purchase made are added in the personal expense tracker. If the limit set is exceeded a notification will be sent to the users Mail. In "Personal Expenses Tracker Application", if entered limit is exceed then SendGrid will send the alert email to the user with expenses detail message and it automatically response to the user by the help of SendGrid software.

9.1 PERFORMANCE METRICS

- In "Personal Expense Tracker Application" will help an organizations can also use employee performance metrics to assess their own competitiveness. These metrics are generally used to assess the efficiency of an entire workforce, as opposed to individual employees.
- This function calculates the revenue per FTE (full-time equivalent). This metric gives a ball-park estimate of how much an individual employee brings in. Low revenue and many employees give a lower rating than the combination of high revenue and fewer employees. This metric can also be used to benchmark companies.
- Every purchase made are added in the personal expense tracker. If the limit set is exceeded a notification will be sent to the users Mail with the daily expense's details and incomes that are help to tracking system of the application. Expense Tracking takes a crucial role in managing the expenses of our daily life and business organizations.
- Expense tracking will bring in several advantages for an organization. That are helpful for the stake-holders in processes of expense. The expense tracker will help any organization to deal with all their expenses more efficiently.

CHAPTER 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES

- The best organizations have a way of tracking and handling these reimbursements. This ideal practice guarantees that the expenses tracked are accurately and in a timely manner. From a company perspective, timely settlements of these expenses when tracked well will certainly boost employees' morale
- Financially Aware and Improve Money Management tracking your expenditures ensures you achieve your project financial targets.
- Effective expense tracking and reporting to avoid conflict. As a project manager or business owner, you can set clear policies for the expense types and reimbursement limits to avoid misunderstandings are about costs. Tracking the project expenses by asking team members to provide receipts is helpful to avoid conflict and maintain compliance also. An excellent reporting mechanism is extremely helpful to support the amount to be reimbursed to your team and also invoicing to your customer.
- Limit of the users monthly budget can be set after logging which keeps the user in a economic track
- User can set their own limit based on their monthly budget, if limit exceed then application will send alert email to the user with expense details.

DISADVANTAGES

- Manually tracking all cash that is spent can be irritating as well as time consuming
- Eventually you may look at your data and realize that you're blown your budget over the last two months, but by then it is too late. So, if you do choose to use this program, ensure that you are also being diligent in checking in on your finances.
- Set up a weekly or biweekly check for yourself to go through your finances and hit on the important points.

CHAPTER 11

CONCLUSION

The new system has overcome most of the limitations of the existing system and works according to the design specification given. The project what we have developed is work more efficient than the other income and expense tracker. The project successfully avoids the manual calculation for avoiding calculating the income and expense per month. The modules are developed with efficient and also in an attractive manner. The developed systems dispense the problem and meet the needs of by providing reliable and comprehensive information. All the requirements projected by the user have been met by the system. The newly developed system consumes less processing time and all the details are updated and processed immediately. Since the screen provides online help messages and is very user friendly, any user will get familiarized with its usage. Modules are designed to be highly flexible so that any failure requirements can be easily added to the modules without facing many problems. The best organizations have a way of tracking and handling these reimbursements. This ideal practice guarantees that the expenses tracked are accurately and in a timely manner. From a company perspective, timely settlements of these expenses when tracked well will certainly boost your employees' morale

- Quickly Determine How Profitable Business Is
- Effective expense tracking and reporting to avoid conflict

CHAPTER 12

FUTURE SCOPE

- The Future Enhancements of the application can be allowed to support in all the upcoming android versions.
- Statistics could be prepared based on the Income, Expense details of the user. Sharing files via Bluetooth, WhatsApp can be allowed. Printing the details of the particular income or expense details can be made. Some of the extra components are like enabling users to register to the application using existing email or social network account, it will synchronize the users profile data to the application

CHAPTER 13

APPENDIX

13.1 SOURCE CODE

app.py

```
from flask import Flask, render_template, request, redirect, session ,url_for
import ibm_db
import re
import sendemail

app = Flask(__name__)

hostname = '1bbf73c5-d84a-4bb0-85b9-
ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;'
uid = 'vjx02808'
pwd = 'ZheXo0qLEishDDb0'
driver = "{IBM DB2 ODBC DRIVER}"
db_name = 'Bludb'
port = '32286'
protocol = 'TCPIP'
cert = "certi.crt"
dsn = (
    "DATABASE={0};"
    "HOSTNAME={1};"
    "PORT={2};"
    "UID={3};"
    "SECURITY=SSL;"
    "PROTOCOL={4};"
    "PWD={6};"
).format(db_name, hostname, port, uid, protocol, cert, pwd)
connection = ibm_db.connect(dsn, "", "")
app.secret_key = 'a'

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

@app.route("/")
```

```

def add():
    return render_template("home.html")
#SIGN--UP--OR--REGISTER
@app.route("/signup")
def signup():
    return render_template("signup.html")

@app.route('/register', methods=['GET', 'POST'])
def register():
    global user_email
    msg = "
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        query = "SELECT * FROM register WHERE email=?;"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account 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:
            query = "INSERT INTO register values(?,?,?);"
            stmt = ibm_db.prepare(connection, query)
            ibm_db.bind_param(stmt, 1, username)
            ibm_db.bind_param(stmt, 2, email)
            ibm_db.bind_param(stmt, 3, password)
            ibm_db.execute(stmt)
            session['loggedin'] = True
            session['id'] = email
            user_email = email
            session['email'] = email
            session['username'] = username

```



```

        msg = 'You have successfully registered ! Proceed Login Process'
        return render_template('login.html', msg = msg)
    else:
        msg = 'PLEASE FILL OUT OF THE FORM'
        return render_template('register.html', msg=msg)

#LOGIN--PAGE

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

@app.route('/login',methods =['GET', 'POST'])
def login():
    global user_email
    msg = "
    if request.method == 'POST' :
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM register WHERE email =? AND password=?;"
        stmt = ibm_db.prepare(connection, sql)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)

    if account:
        session['loggedin'] = True
        session['id'] = account['EMAIL']
        user_email= account['EMAIL']
        session['email']=account['EMAIL']
        session['username'] = account['USERNAME']

        return redirect('/home')
    else:
        msg = 'Incorrect username / password !'
        return render_template('login.html', msg = msg)

#CHANGE FORGOT PASSWORD

```

```

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

@app.route("/forgotpw", methods=['GET', 'POST'])
def forgotpw():
    msg = ""
    if request.method == 'POST' :
        email = request.form['email']
        password = request.form['password']
        query = "SELECT * FROM register WHERE email=?;"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            query = "UPDATE register SET password = ? WHERE email = ?;"
            stmt = ibm_db.prepare(connection, query)
            ibm_db.bind_param(stmt, 1, password)
            ibm_db.bind_param(stmt, 2, email)
            ibm_db.execute(stmt)
            msg = 'Successfully changed your password ! Proceed Login Process'
            return render_template('login.html', msg = msg)
        else:
            msg = 'PLEASE FILL OUT THE CORRECT DETAILS'
            return render_template('forgot.html', msg=msg)

#ADDING----DATA
@app.route("/add")
def adding():
    return render_template('add.html')

@app.route('/addexpense',methods=['GET', 'POST'])
def addexpense():
    global user_email
    que = "SELECT * FROM expenses where id = ? ORDER BY 'dates' DESC"
    stm = ibm_db.prepare(connection, que)
    ibm_db.bind_param(stm, 1, session['email'])
    ibm_db.execute(stm)

```

```

dictionary=ibm_db.fetch_assoc(stm)
expense=[]
while dictionary != False:
    exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],di
ctinary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"])
    expense.append(exp)
    dictionary = ibm_db.fetch_assoc(stm)
i=len(expense)+1
idx=str(i)
dates = request.form['date']
expensename = request.form['expensename']
amount = request.form['amount']
paymode = request.form['paymode']
category = request.form['category']
query = "INSERT INTO expenses VALUES (?, ?, ?, ?, ?, ?);"
stmt = ibm_db.prepare(connection, query)
ibm_db.bind_param(stmt, 1, session['email'])
ibm_db.bind_param(stmt, 2, dates)
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.bind_param(stmt, 7, idx)
ibm_db.execute(stmt)
print(dates + " " + expensename + " " + amount + " " + paymode + " " + category)

return redirect("/display")

```

```

#DISPLAY---graph
@app.route("/display")
def display():
    query = "SELECT * FROM expenses where id = ? ;"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    rexpense=[]
    while dictionary !=False:

```

```

        exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],dictionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary["IDX"])

        rexpense.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
        que = "SELECT MONTH(dates) as DATES, SUM(amount) as AMOUNT FROM expenses WHERE id=? AND YEAR(dates)= YEAR(now()) GROUP BY MONTH(dates);"
        stm = ibm_db.prepare(connection, que)
        ibm_db.bind_param(stm, 1,session['email'])
        ibm_db.execute(stm)
        dictionary=ibm_db.fetch_assoc(stm)
        texpense=[]
        while dictionary != False:
            exp=(dictionary["DATES"],dictionary["AMOUNT"])
            texpense.append(exp)
            dictionary = ibm_db.fetch_assoc(stm)
        print(texpense)

        quer = "SELECT * FROM expenses WHERE id = ? AND YEAR(dates)= YEAR(now());"
        st = ibm_db.prepare(connection, quer)
        ibm_db.bind_param(st, 1,session['email'])
        ibm_db.execute(st)
        dictionary=ibm_db.fetch_assoc(st)
        expense=[]
        while dictionary != False:
            exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],dictionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary["IDX"])
            expense.append(exp)
            dictionary = ibm_db.fetch_assoc(st)

        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[3]
    if x[5] == "food":
        t_food += x[3]

    elif x[5] == "entertainment":
        t_entertainment += x[3]

    elif x[5] == "business":
        t_business += x[3]
    elif x[5] == "rent":
        t_rent += x[3]

    elif x[5] == "EMI":
        t_EMI += x[3]

    elif x[5] == "other":
        t_other += x[3]

print(total)

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

qur = "SELECT * FROM expenses WHERE id = ? AND MONTH(dates)=
MONTH(now());"
stt = ibm_db.prepare(connection, qur)
ibm_db.bind_param(stt, 1, session['email'])
ibm_db.execute(stt)
dictionary=ibm_db.fetch_assoc(stt)
lexpense=[]
while dictionary != False:
    exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],di
ctionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary[
"IDX"])

```

```

    lexpense.append(exp)
    dictionary = ibm_db.fetch_assoc(stt)

    tttotal=0
    to_food=0
    to_entertainment=0
    to_business=0
    to_rent=0
    to_EMI=0
    to_other=0

    for x in lexpense:
        tttotal += x[3]
        if x[5] == "food":
            to_food += x[3]

        elif x[5] == "entertainment":
            to_entertainment += x[3]

        elif x[5] == "business":
            to_business += x[3]
        elif x[5] == "rent":
            to_rent += x[3]

        elif x[5] == "EMI":
            to_EMI += x[3]

        elif x[5] == "other":
            to_other += x[3]

    print(tttotal)

    qy = "SELECT max(IDX) as IDX FROM limits where id=?;"
    smt = ibm_db.prepare(connection, qy)
    ibm_db.bind_param(smt, 1, session['email'])
    ibm_db.execute(smt)
    dictionary = ibm_db.fetch_assoc(smt)
    uexpense=[]
    while dictionary != False:

```

```

        exp=(dictionary["IDX"])
        uexpense.append(exp)
        dictionary = ibm_db.fetch_assoc(smt)
    k=uexpense[0]
    qu = "SELECT NUMBER FROM limits where id=? and idx=?"
    sm = ibm_db.prepare(connection, qu)
    ibm_db.bind_param(sm, 1, session['email'])
    ibm_db.bind_param(sm, 2, k)
    ibm_db.execute(sm)
    dictionary = ibm_db.fetch_assoc(sm)
    fexpense=[]
    while dictionary != False:
        exp=(dictionary["NUMBER"])
        fexpense.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)

    if len(fexpense) <= 0:
        print("Enter the limit First")
    else:
        if tttotal > fexpense[0]:
            m=sendemail.sendgridmail(session["email"])
            print(m)
        else: print("Error")
    return render_template("display.html",rexpense=rexpense, texpense = texpense,
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 )

#delete---the--data
@app.route('/delete/<idx>', methods = ['POST', 'GET' ])
def delete(idx):
    query = "DELETE FROM expenses WHERE id=? and idx=?;"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, session["email"])
    ibm_db.bind_param(stmt, 2, idx)
    ibm_db.execute(stmt)
    print('deleted successfully')
    return render_template("display.html")

```

```

#UPDATE---DATA
@app.route('/edit/<id>', methods = ['POST', 'GET' ])
def edit(id):
    query = "SELECT * FROM expenses WHERE id=? and idx=?;"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.bind_param(stmt, 2, id)
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    expense=[]
    while dictionary != False:
        exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],dictionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary["IDX"])
        expense.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    print(expense)
    return render_template('edit.html', expenses = expense[0])

@app.route('/update/<id>', methods = ['POST'])
def update(id):
    if request.method == 'POST' :
        dates = request.form['date']
        expensename = request.form['expensename']
        amount = request.form['amount']
        paymode = request.form['paymode']
        category = request.form['category']
        query = "UPDATE expenses SET dates = ? , expensename = ? , amount = ?, paymode = ?, category = ? WHERE id = ? and idx=?;"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, dates)
        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, session['email'])

```



```

    ibm_db.bind_param(stmt, 7, id)
    ibm_db.execute(stmt)

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

#limit
@app.route("/limit" )
def limit():
    return render_template('limit.html')

@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
    que = "SELECT * FROM limits where id = ? ;"
    stm = ibm_db.prepare(connection, que)
    ibm_db.bind_param(stm, 1, session['email'])
    ibm_db.execute(stm)
    if request.method == "POST":
        dictionary=ibm_db.fetch_assoc(stm)
        expense=[]
        while dictionary != False:
            exp=(dictionary['ID'],dictionary['NUMBER'],dictionary['IDX'])
            expense.append(exp)
            dictionary = ibm_db.fetch_assoc(stm)
        i=len(expense)+1
        idx=str(i)
        number= request.form['number']
        query = "INSERT INTO limits VALUES(?,?,?)"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, session['email'])
        ibm_db.bind_param(stmt, 2, number)
        ibm_db.bind_param(stmt, 3, idx)
        ibm_db.execute(stmt)
        return redirect('/limitn')

@app.route("/limitn")
def limitn():
    query = "SELECT max(IDX) as IDX FROM limits where id=?;"

```

```

stmt = ibm_db.prepare(connection, query)
ibm_db.bind_param(stmt, 1, session['email'])
ibm_db.execute(stmt)
dictionary = ibm_db.fetch_assoc(stmt)
expense=[]
while dictionary != False:
    exp=(dictionary["IDX"])
    expense.append(exp)
    dictionary = ibm_db.fetch_assoc(stmt)
k=expense[0]
que = "SELECT NUMBER FROM limits where id=? and idx=?"
stmt = ibm_db.prepare(connection, que)
ibm_db.bind_param(stmt, 1, session['email'])
ibm_db.bind_param(stmt, 2, k)
ibm_db.execute(stmt)
dictionary = ibm_db.fetch_assoc(stmt)
texpanse=[]
while dictionary != False:
    exp=(dictionary["NUMBER"])
    texpanse.append(exp)
    dictionary = ibm_db.fetch_assoc(stmt)
s=texpense[0]
return render_template("limit.html" , y= s)

```

```

#REPORT
@app.route("/today")
def today():
    query = "SELECT dates, amount FROM expenses WHERE id = ? AND
DATE(dates) = DATE(NOW()); "
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, str(session['email']))
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    texpanse=[]
    while dictionary != False:
        exp=(dictionary["DATES"],dictionary["AMOUNT"])
        texpanse.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    print(texpanse)

```

```

        query = "SELECT * FROM expenses WHERE id = ? AND DATE(dates) =
DATE(NOW())"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, session['email'])
        ibm_db.execute(stmt)
        dictionary=ibm_db.fetch_assoc(stmt)
        expense=[]
        while dictionary != False:
            exp=(dictionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEG
ORY"])
            expense.append(exp)
            dictionary = ibm_db.fetch_assoc(stmt)

        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[0]
            if x[2] == "food":
                t_food += x[0]

            elif x[2] == "entertainment":
                t_entertainment += x[0]

            elif x[2] == "business":
                t_business += x[0]
            elif x[2] == "rent":
                t_rent += x[0]

            elif x[2] == "EMI":
                t_EMI += x[0]

            elif x[2] == "other":

```

```

        t_other += x[0]

    print(total)

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

    return render_template("today.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("/month")
def month():
    query = "SELECT dates, SUM(amount) as AMOUNT FROM expenses WHERE
id= ? AND MONTH(dates)= MONTH(now()) GROUP BY dates ORDER BY dates;"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, str(session['email']))
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    texpanse=[]
    while dictionary != False:
        exp=(dictionary["DATES"],dictionary["AMOUNT"])
        texpanse.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    print(texpanse)

    query = "SELECT * FROM expenses WHERE id = ? AND MONTH(dates)=
MONTH(now());"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)

```

```

expense=[]
while dictionary != False:
    exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],dictionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary["IDX"])
    expense.append(exp)
    dictionary = ibm_db.fetch_assoc(stmt)

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[3]
    if x[5] == "food":
        t_food += x[3]

    elif x[5] == "entertainment":
        t_entertainment += x[3]

    elif x[5] == "business":
        t_business += x[3]
    elif x[5] == "rent":
        t_rent += x[3]

    elif x[5] == "EMI":
        t_EMI += x[3]

    elif x[5] == "other":
        t_other += x[3]

print(total)

print(t_food)
print(t_entertainment)

```

```

print(t_business)
print(t_rent)
print(t_EMI)
print(t_other)

return render_template("today.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("/year")
def year():
    query = "SELECT MONTH(dates) as DATES, SUM(amount) as AMOUNT
FROM expenses WHERE id=? AND YEAR(dates)= YEAR(now()) GROUP BY
MONTH(dates);"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1,session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    texpanse=[]
    while dictionary != False:
        exp=(dictionary["DATES"],dictionary["AMOUNT"])
        texpanse.append(exp)
        dictionary = ibm_db.fetch_assoc(stmt)
    print(texpanse)

    query = "SELECT * FROM expenses WHERE id = ? AND YEAR(dates)=
YEAR(now());"
    stmt = ibm_db.prepare(connection, query)
    ibm_db.bind_param(stmt, 1,session['email'])
    ibm_db.execute(stmt)
    dictionary=ibm_db.fetch_assoc(stmt)
    expense=[]
    while dictionary != False:
        exp=(dictionary["ID"],dictionary["DATES"],dictionary["EXPENSENAME"],di
ctionary["AMOUNT"],dictionary["PAYMODE"],dictionary["CATEGORY"],dictionary[
"IDX"])
        expense.append(exp)

```

```
dictionary = ibm_db.fetch_assoc(stmt)
```

```
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[3]
```

```
    if x[5] == "food":
```

```
        t_food += x[3]
```

```
    elif x[5] == "entertainment":
```

```
        t_entertainment += x[3]
```

```
    elif x[5] == "business":
```

```
        t_business += x[3]
```

```
    elif x[5] == "rent":
```

```
        t_rent += x[3]
```

```
    elif x[5] == "EMI":
```

```
        t_EMI += x[3]
```

```
    elif x[5] == "other":
```

```
        t_other += x[3]
```

```
print(total)
```

```
print(t_food)
```

```
print(t_entertainment)
```

```
print(t_business)
```

```
print(t_rent)
```

```
print(t_EMI)
```

```
print(t_other)
```

```

        return render_template("today.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)
    return render_template('home.html')

if __name__ == "__main__":
    app.run(debug=True)

```

Homepage.html

```

{% extends 'base.html' %}

{% block body %}

<style>

body{

    background-repeat: no-repeat;

    background-attachment: fixed;

    overflow: hidden;

}

```



```

img{
    border-radius: 14px;
}

H1 {
    position: relative;
    right: -650PX;
    top: -350PX;
    color:#000000;
    font-size: 60px;
}

p{
position: relative;
right: -680px;
top: -300px;
font-family:monospace;
font-size: 18px;
color: #000000;
}

span{
    position: relative;
    right: -650px;
    top: -310px;
    color: rgb(221, 67, 157);
}

.ccc {

```

```

    position: relative;

    top:80px;

    left:-100px;

    bottom: 100px;

}

</style>

<div id=aa class="container">

<div class="ccc">

    <!--  -->

    <video height="450px" width="450px" style="position: relative; top: 50px; left: 90PX;"
autoplay loop>

        <source src="/static/images/hbg1.mp4" type="video/mp4">

        <source src="/static/images/hbg1.ogg" type="video/ogg">

        Your browser does not support the video tag.

    </video>

    <h1>LET'S START JOURNEY</h1>

    <P>Fin Budget web application helps<br> you to maintain budget<br>

        and analyse the expense...</P>

</div>

    <span class="btn btn-outline-dark"><a style="color: #FFA62B;" href="/add">Let's
Begin</a></span>

</div>

{% endblock %}

```

Signup.html

```
<html>

<head>

<meta charset="utf-8">

<title>Sign-up</title>

<link href="..\static\css\signup.css" rel="stylesheet">

<script src="https://kit.fontawesome.com/a81368914c.js"></script>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
n5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">

</head>

<body style="background-color:#A594F9;">

<!--container----->

<div class="container">

<!--sign-up-box-container--->

<div class="sign-up">

    <div id="png"><a href="/" title="HOME"></a></div>

    <!--heading-->

    <form action="/register" method="post">

        <div class="msg"><h4>{{ msg }}</h4></div>

        <h1 class="heading">Hello,Friend</h1>

        <!--name-box-->

        <div class="text">

            
```

```

<input placeholder="Name" type="text" name="username"/>
</div>

<!--Email-box-->

<div class="text">



<input placeholder=" Example@gmail.com" type="email" name="email" />

</div>

<!--Password-box-->

<div class="text">



<input placeholder=" Password" type="password" name="password"/>

</div>

<!--terms-->

<div class="terms">

    <p class="conditions" style="color:black"><br><input class="check" type="checkbox" required>I
read and agree to <a href="#">Terms & Conditions</a></p>

</div>

<!--button-->

<div class="toop">

<button type="submit" class="btn btn-primary" >CREATE ACCOUNT</button> </div>

<br>

</form>

<!--sign-in-->

<div class="colu">

    <p class="conditions" id="p3">Already have an account? <a href="/signin">Sign in</a></p>
</div></div>

```

```

</div>

<!--text-container-->

<div class="text-container">

<h1 style="color: #2d2c2c;font-family:cursive;">Glad to see you</h1>

<div class="diag"></div>

<div class="para"> <b>Welcome</b>,.Please Fill in the blanks for sign up</div>

</div>

</div>

</body>

</html>

```

Login.html

```

<!DOCTYPE html>

<html>

<head>

<title>Animated Login Form</title>

<link rel="stylesheet" type="text/css" href="..\static\css\login.css">

<link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

<script src="https://kit.fontawesome.com/a81368914c.js"></script>

<meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<body >



<div class="container">

```

```

<div class="img">

    <div id="png"><a href="/" title="HOME"></a></div>

</div>


<div class="login-content">

    <form action="/login" method="POST">

        <div class="msg">{ { msg } }</div>

        <h2 class="title">Welcome</h2>

        <div class="input-div one">

            <div class="i">

                <i class="fas fa-user"></i>

            </div>

            <div class="div">

                <input type="email" name="email" Placeholder="Email" class="input" required>

            </div>

        </div>

        <div class="input-div pass">

            <div class="i">

                <i class="fas fa-lock"></i>

            </div>

            <div class="div">

```

```

        <input type="password" name="password" placeholder="Password" class="input"
required>

        </div>

    </div>

    <a href="/forgot">Forgot Password?</a>

    <input type="submit" class="btn" value="Login">

    <br>

    <br>

    <div class="app" >

        <p>Don't have an account? <a id="app1" href="/signup">Signup</a></p></div>

    </form>

</div>

</div>

<script type="text/javascript" src="login.js"></script>

</body>

</html>

```

Forgot.html

```

<html lang="en">

<head>

<style>

    *, *:before, *:after {

    -webkit-box-sizing: border-box;

        box-sizing: border-box;

    }

```

```

body {

    background-image: url('static/images/fpbgl.jpeg');

    background-repeat: no-repeat;

    background-attachment: fixed;

    background-size: cover;

    font-family: 'Lato', sans-serif;

}

/***** For Smartphones *****/

.container-center {

    position: absolute;

    left: 25%;

    width: 95%;

    height: 90%;

    -webkit-transition: all 0.1s;

    transition: all 0.1s;

    bottom: 40%;

    -webkit-transform: translateX(-50%) translateY(50%);

    transform: translateX(-50%) translateY(50%);

}

h2, img {

    text-align: center;

    color: white;

    font-weight: 10;

    text-shadow: 0px 1px rgba(0, 0, 0, 0.3);

}

```



```
h4{  
    text-align: center;  
    color:black;  
    font-size: 1.1em;  
    font-family: times;  
    font-style:normal;  
    line-height: 130%;  
    opacity: .6;  
}
```

```
form {  
    width: 100%;  
    overflow: hidden;  
    background-color: #FEFEFE;  
    padding: 70px 13px;  
    border-radius: 21px;  
    -webkit-box-shadow: 0px 5px 34px rgba(0, 0, 0, 0.1);  
    box-shadow: 0px 5px 34px rgba(0, 0, 0, 0.1);  
}
```

```
formgroup {  
    position: relative;  
    width: 100%;  
    display: block;  
    margin: 1em 0;  
    font-size: 1em;
```

```
}  
  
formgroup input {  
    width: 100%;  
    border: none;  
    border-bottom: 1px solid #888888;  
    padding: 8px 0;  
    font-size: inherit !important;  
    margin-bottom: 13px;  
    outline: none;  
    opacity: 0.7;  
    font-weight: 600;  
}  
  
formgroup input:focus {  
    opacity: 1;  
    border-bottom: 2px solid #F71442;  
    color: #F71442;  
}  
  
formgroup label {  
    position: absolute;  
    font-size: 0.8em;  
    top: -1em;  
    left: 0;  
    -webkit-transition: all 0.3s;  
    transition: all 0.3s;  
    opacity: 0.7;  
    color: #888888;
```

```

    text-transform: uppercase;
}

formgroup span {
    position: absolute;
    top: -1em;
    left: -500px;
    opacity: 0;
    color: #333333;
    font-weight: bold;
    text-transform: uppercase;
    font-size: 0.8em;
    -webkit-transition: all 0.3s;
    transition: all 0.3s;
}

formgroup input:focus + label {
    left: 500px;
    opacity: 0;
}

formgroup input:focus ~ span {
    left: 0;
    opacity: 1;
}

.forgot {
    display: block;
    width: 100%;

```

```
text-align: center;

font-size: 1em;

font-weight: bold;

margin-top: 21px;

opacity: 0.8;

}
```

```
#login-btn {

border: none;

color: #FEFEFE;

padding: 0.8em 0;

font-size: 1em;

font-weight: 300;

width: 100%;

border-radius: 55px;

-webkit-box-shadow: 0px 3px 21px #A594F9;

    box-shadow: 0px 3px 21px #A594F9;

background: -webkit-gradient(linear, left top, right top, from(#F98340), to(#F71442));

background: #A594F9;

background-size: 100%;

text-shadow: 0px 1px 2px rgba(0, 0, 0, 0.2);

}
```

```
.social {

margin-top: 1.8em;

display: -webkit-box;
```

```

display: -ms-flexbox;

display: flex;

}

.social button {

width: 50%;

margin: 0 8px;

padding: 10px 0;

font-size: 0.9em;

border: none;

border-radius: 34px;

-webkit-box-shadow: 0px 1px 13px rgba(0, 0, 0, 0.2);

    box-shadow: 0px 1px 13px rgba(0, 0, 0, 0.2);

color: white;

font-weight: bold;

cursor: pointer;

}

.social #facebook {

background: #1F4788;

background: -webkit-gradient(linear, left top, right top, from(#4B77BE), to(#1F4788));

background: linear-gradient(to right, #4B77BE, #1F4788);

background-size: 100%;

}

.social #google {

background: #FEFEFE;

background: -webkit-gradient(linear, left top, right top, from(#FEFEFE), to(#f1f1f1));

background: linear-gradient(to right, #FEFEFE, #f1f1f1);

```

```
background-size: 100%;  
color: #F71442;  
}
```

```
p {  
  color: white;  
  text-align: center;  
}
```

```
p a {  
  color: inherit;  
  text-decoration: none;  
  font-weight: bold;  
}
```

```
/****** For Tablets *****/
```

```
@media screen and (min-width: 480px) {  
  .container-center {  
    width: 70%;  
  }  
}
```

```
#login-btn {  
  padding: 0.8em 0;  
  font-size: 1.2em;  
}  
}
```

```
/****** For Desktop Monitors *****/
```

```

@media screen and (min-width: 768px) {

.container-center {

width: 500px;

}

}

</style>

</head>

<body>

<div class="container-center">

<center><h1>Change Your Password</h1></center>

<form action="/forgotpw" method="POST">

<h4>Enter your email and new password

</h4>

<br>

<formgroup>

<input type="text" name="email" id="email" placeholder="Email"/>

<span>enter your email</span>

</formgroup>

<formgroup>

<input type="text" name="password" id="password" placeholder="New Password"/>

<span>enter your new password</span>

</formgroup>

<br>

<input type="submit" id="login-btn" value="Reset">

</form>

```

```
<p>Did you remember? <a href="/login">Login</a></p>
</div>
</body>
</html>
```

Home.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="..\static\css\home.css">

    <title>Fin Budget</title>

  </head>

  <body>

    <!-- Header -->

    <section id="header">

      <div class="header container">

        <div class="nav-bar">

          <div class="brand">

            <a href="#hero">

              <h1><span>F</span>in <span>B</span>udget</h1>

            </a>

          </div>

          <div class="nav-list">
```



```

<div class="hamburger">

  <div class="bar"></div>

</div>

<ul>

  <li><a href="#hero" data-after="Home">Home</a></li>

  <li><a href="#services" data-after="Service">Services</a></li>


  <li><a href="#about" data-after="About">About</a></li>

  <li><a href="#contact" data-after="Contact">Contact</a></li>

  <LI><a href="/signin" data-after="Login">-Login-</a></LI>

</ul>

</div>

</div>

</div>

</section>

<!-- End Header -->


<!-- Hero Section -->

<section id="hero">

  <div class="hero container">

    <div>

      <h1>Hello, <span></span></h1>

      <h1>Welcome To <span></span></h1>

      <h1>Personal Expense Tracker Application <span></span></h1>

      <a href="/signup" type="button" class="cta">Sign-up</a>

    </div>

```

```

</div>

</section>

<!-- End Hero Section -->


<!-- Service Section -->

<section id="services">

  <div class="services container">

    <div class="service-top">

      <h1 class="section-title">Services</h1>

      <p>Finbudget provides a many services to the customer and industries. Financial solutions to meet your needs whatever your money goals,there is a Finbudget solution to help you reach them </p>

    </div>

    <div class="service-bottom">

      <div class="service-item">

        <div class="icon"></div>

        <h2>Personal Expenses</h2>

        <p>Budgeting is more than paying bills and setting aside savings.it's about creating a money plan for the life you want</p>

      </div>

      <div class="service-item">

        <div class="icon"></div>

        <h2>Investments</h2>

        <p>Follow your investments and bring your portfolio into focus with support for stocks,bonds,CDs,mutual funds and more</p>

      </div>

      <div class="service-item">

```

```
<div class="icon"></div>
```

```
<h2>Online Banking</h2>
```

```
<p>Finbudget application can automatically download transactions and send payments online
from many financial institutions</p>
```

```
</div>
```

```
<div class="service-item">
```

```
<div class="icon"></div>
```

```
<h2>Financial Life</h2>
```

```
<p>Get your Complete financial picture at a glance. With Finbudget application you can view
your all the financial activities
```

```
</p>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</section>
```

```
<!-- End Service Section -->
```

```
<!-- About Section -->
```

```
<section id="about">
```

```
<div class="about container">
```

```
<div class="col-left">
```

```
<div class="about-img">
```

```

```

```
<br><br><br><br><br>
```

```
<div><h2>Personal Expense Tracker ceo, Nivethitha S, Newton Habakuk S, Pooja B, Naveen
Kumar S </h2></div>
```

```
</div>
```

</div>

<div class="col-right">

<h1 class="section-title">About Us</h1>

<h2>Financial Solution</h2>

<p>Fin budget financial solution is one among Leading financial company. Finbudget provides a many services to the customer and industries. Financial solutions to meet your needs whatever your money goals,there is a Finbudget solution to help you reach them. you can Contact our service center for further information and also follow our social media for update on new services </p>

Follow Us

</div>

</div>

</section>

<!-- End About Section -->

<!-- Contact Section -->

<section id="contact">

<div class="contact container">

<div>

<h1 class="section-title">Contact info</h1>

</div>

<div class="contact-items">

<div class="contact-item">

<div class="icon"></div>

<div class="contact-info">

<h1>Phone</h1>

```

        <h2>+91 9944236079</h2>

        <h2>+91 9514095105</h2>

        <h2>+91 9660301238</h2>

        <h2>+91 7010636259</h2>

    </div>

</div>

<div class="contact-item">

    <div class="icon"></div>

    <div class="contact-info">

        <h1>Email</h1>

        <h2>nivethithashanmuganathan5@gmail.com</h2>

        <h2>newttonhabakuk.s@gmail.com</h2>

        <h2>bpooja37202@gmail.com</h2>

        <h2>naveenkmr.0402@gmail.com</h2>

    </div>

</div>

<div class="contact-item">

    <div class="icon"></div>

    <div class="contact-info">

        <h1>Address</h1>

        <h2>Coimbatore,Tamilnadu,India</h2>

    </div>

</div>

</div>

</div>

```

```

</section>

<!-- End Contact Section -->


<!-- Footer -->

<section id="footer">

  <div class="footer container">

    <div class="brand">

      <h1><span>F</span>in <span>B</span>udget</h1>

    </div>

    <h2>Your Complete Financial Solution</h2>

    <div class="social-icon">

      <div class="social-item">

        <a href="https://twitter.com/Newtz_newtton"></a>

      </div>

      <div class="social-item">

        <a href="https://www.linkedin.com/in/nivethitha-shanmuganathan/"></a>

      </div>

      <div class="social-item">

        <a href="https://www.instagram.com/___naveen.___01/"></a>

      </div>

      <div class="social-item">

        <a href="http://t.me/poojabalak"></a>

      </div>

    </div>

  </div>

```

```
<p>Copyright © 2021 . All rights reserved</p>
</div>
</section>
<!-- End Footer -->
<script src="..\static\js\home.js"></script>
</body>
</html>
```

Add.html

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

{% block body %}

<style>

    body{

        overflow: hidden;

    }

</style>

<div class="container">

    <div class="row">

        <div class="col-md-6">

            <br>

            <br>

            <br>

            <h3>Add Expense</h3>

            <form action="/addexpense" method="POST">

                <div class="form-group">
```

```

<br>

<label for="">Date</label>

<input class="form-control" type="date" name="date" id="date"></div>

<div class="form-group"> <label for="">Expense name</label>

    <input class="form-control" type="text" name="expensename" id="expensename">

</div>

<div class="form-group">

    <label for="">Expense Amount</label>

    <input class="form-control" type="number" min="0" name="amount" id="amount">

</div>

<div class="form-group">

    <label for=""></label>

    <select class="form-control" name="paymode" id="paymode">

        <option selected hidden>Pay-Mode</option>

        <option name="cash" value="cash">cash</option>

        <option name="debitcard" value="debitcard">debitcard</option>

        <option name="creditcard" value="creditcard">creditcard</option>

        <option name="epayment" value="epayment">epayment</option>

        <option name="onlinebanking" value="onlinebanking">onlinebanking</option>

    </select>

<div class="form-group">

    <label for=""></label>

    <select class="form-control" name="category" id="category">

```



```

        <option selected hidden>Category</option>

        <option name = "food" value="food">food</option>

        <option name = "entertainment" value="entertainment">Entertainment</option>

        <option name = "business" value="business">Business</option>

        <option name = "rent" value="rent">Rent</option>

        <option name = "EMI" value="EMI">EMI</option>

        <option name = "other" value="other">other</option>

    </select>

</div>

<input class="btn btn-danger" type="submit" value="Add" id="">

</form>

<div style="position: relative; left: 590px; top: -460px;" class="image">

</div>

</div>

</div>

</div>

    {% endblock %}

```

Display.html

```

{% extends 'base.html' %}

{% block body %}

<div class="container ">

    <h3 class="mt-3">EXPENSES</h3>

    {% if expense is defined %}

```

```
{% for row in expense %}
```

```
<div class="row">

  <div class="col-md-12">

    <div class="card shadow-sm mb-2 bg-white rounded"></div>

    <div class="card-body">

      <div class="row">

        <div class="col-md-2">

          <span class="btn btn-outline-dark">{{ row[1] }}</span> </div>

          <div class="col-md-2 mt-3"><H6>{{ row[2] }}</H6></div>

          <div class="col-md-2 mt-3"> ₹<span style=" color: rgb(255, 0, 0) "> {{ row[3] }} </div>

          <div class="col-md-2 mt-3">

            <span class="badge badge-pill badge-info">{{ row[4] }}</span>

          </div>

          <div class="col-md-2 mt-3">

            <span class="badge badge-primary">{{ row[5] }}</span>

          </div>

          <div class="col-md-1 mt-3">

            <a href="/edit/{{ row[6] }}" class="btn btn-sm btn-success">Edit</a>

          </div>

          <div class="col-md-1 mt-3">

            <a href="/delete/{{ row[6] }}" class="btn btn-sm btnDelete btn-
success">Delete</a>

          </div>

        </div>

      </div>

    </div>

  </div>
```

```

    </div>

</div>

<!--when no DATA-Found-->

{% else %}

<div class="card shadow-sm mb-2 bg-white rounded"></div>

<div class="card-body">

    <div style="text-align: center ; font-family: monospace; color:red ; "><h5><a href="/add"> ADD-
DATA </a> to Display</h3></div>

</div>

{% endfor %}

{% endif %}

<div class="row">

    <div class="col-md-6">

        <h3 class="mt-5">Expense Breakdown</h3>

        <div class="card shadow mb-2 bg-white rounded-bottom">

            <div class="card-body ">

                <div class="row">

                    <div class="col-md-6">Food</div>

                    <div id="tfood" class="col-md-6"> {{ t_food}} </div>

                </div>

            </div>

        </div>

    </div>

    <div class="card shadow mb-2 bg-white rounded">

        <div class="card-body">

            <div class="row">

```

```

    <div class="col-md-6">Entertainment</div>

    <div id="tentertainment" class="col-md-6"> {{ t_entertainment}} </div>

</div>

</div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Business</div>

            <div id="tbusiness" class="col-md-6"> {{t_business}} </div>

        </div>

    </div>

</div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Rent</div>

            <div id="trent" class="col-md-6"> {{ t_rent }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">EMI</div>

```

```

        <div id="temi" class="col-md-6">{{ t_EMI }} </div>

    </div>

</div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Other</div>

            <div id="tother" class="col-md-6"> {{ t_other }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 btn-outline-danger rounded-pill">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Total</div>

            <div class="col-md-6">₹ {{total}} </div>

        </div>

    </div>

</div>

</div>

</div>

<div class="col-md-6">

<canvas id="myChart" width="400" height="400"></canvas>

<script>

    let food = document.getElementById('tfood').innerHTML

    let entertainment = document.getElementById('tentertainment').innerHTML

```

```

let business = document.getElementById('tbusiness').innerHTML

let rent = document.getElementById('trent').innerHTML

let emi = document.getElementById('temi').innerHTML

let other = document.getElementById('tother').innerHTML

var ctx = document.getElementById('myChart').getContext('2d');
var myChart = new Chart(ctx, {
  type: 'doughnut',
  data: {
    labels: ['Food', 'Entertainment', 'Business', 'Rent', 'EMI', 'Other'],
    datasets: [{
      label: 'Expenses Chart',
      data: [food, entertainment, business, rent, emi, other],
      backgroundColor: [
        'rgb(255, 99, 132)',
        'rgb(0, 0, 0)',
        'rgb(255, 205, 86)',
        'rgb(201, 203, 207)',
        'rgb(54, 162, 235)',
        'rgb(215, 159, 64)'
      ],
    }],
  },
  options: {
    responsive: true,
    plugins: {
      legend: {

```

```

        position: 'bottom',
    },
    title: {
        display: true,
        text: 'EXPENSE BREAKDOWN'
    }
}

});

</script>

</div>

</div>

</div>

{% endblock %}

```

Edit.html:

```

{% extends 'base.html' %}

{% block body %}

<div class="container">

    <div class="row">

        <div class="col-md-6">

            <h3>Edit Expense</h3>

            <form action="/update/{{ expenses[6] }}" method="POST">

                <input type="hidden" class="form-control" name="" value = "{{ expenses[6] }}"
id="">

                <div class="form-group">

                    <label for="">Date</label>

```

```
<input class="form-control" type="date" name="date"
value="{{ expenses[1] }}" id="date"></div>
```

```
<script type="text/javascript">
```

```
var d = new Date(value="{{ expenses[1] }}" );
```

```
var elem = document.getElementById("date");
```

```
elem.value = d.toISOString().slice(0,16);
```

```
</script>
```

```
<div class="form-group"> <label for="">Expense name</label>
```

```
<input class="form-control" type="text" name="expensename" value="{{ expenses[2] }}"
id="expensename">
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="">Expense Amount</label>
```

```
<input class="form-control" type="number" min="0" name="amount"
value="{{ expenses[3] }}" id="amount">
```

```
</div>
```

```
<div class="form-group">
```

```
<label for=""></label>
```

```
<select class="form-control" name="paymode"
value="{{ expenses[4] }}" id="paymode">
```

```
<option selected hidden>{{ expenses[4] }}</option>
```

```
<option value="cash">cash</option>
```

```
<option value="debitcard">debitcard</option>
```

```
<option value="creditcard">creditcard</option>
```

```
<option value="epayment">epayment</option>
```

```
<option value="onlinebanking">onlinebanking</option>
```

```
</select>
```



```

    <div class="form-group">

        <label for=""></label>

        <select class="form-control" name="category"
value="{{ expenses[5] }}" id="category">

            <option selected hidden>{{ expenses[5] }}</option>

            <option value="food">food</option>

            <option value="entertainment">Entertainment</option>

            <option value="business ">Business</option>

            <option value="rent">Rent</option>

            <option value="EMI">EMI</option>

            <option value="other">other</option>

        </select>

    </div>

    <input class="btn btn-danger" type="submit" value="Update" id="">

</form>

</div>

</div>

</div>

{% endblock %}

```

Base.html

```

<!DOCTYPE html>

<html lang="en">

<head>

    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkglIXeMed4M0jlfdPvg6uqKI2xXr2"
crossorigin="anonymous">

```

```

<script src="https://cdn.jsdelivr.net/npm/chart.js@3.2.1/dist/chart.min.js"></script>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
integrity="sha384-
9/reFTGAW83EW2RDu2S0VKAzlap3H66lZ81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
integrity="sha384-B4gt1jrGC7Jh4AgTPSdUUtOBvfO8shuf57BaghqFfPIYxofvL8/KUEfYiJOMMV+rV"
crossorigin="anonymous"></script>

<link rel="stylesheet" href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css"
integrity="sha384-
AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p"
crossorigin="anonymous"/>

<title>Fin Budget</title>

</head>

<body>

<nav class="navbar sticky-top navbar-expand-lg navbar-light" style="background-color:
#000000;">

<a style="color: #FFA62B; font-size: 25px;" class="navbar-brand" href="#">Fin Budget</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav mr-auto" >

<li class="nav-item active">

<a style="color: #ffffff;" class="nav-link" href="/home">Home <span class="sr-
only">(current)</span></a>

</li>

```

```

<li class="nav-item">

  <a style="color: rgb(255, 255, 255);" class="nav-link" href="/add">Add</a>

</li>

<li class="nav-item">

  <a style="color: rgb(255, 255, 255);" class="nav-link" href="/display">History</a>

</li>

<li class="nav-item">

  <a style="color: rgb(236, 236, 236);" class="nav-link" href="/limit">LIMIT</a>

</li>

<li class="nav-item dropdown">

  <a style="color: rgb(255, 255, 255);" class="nav-link dropdown-toggle" href="#"
id="navbarDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">

    Report

  </a>

  <div class="dropdown-menu" aria-labelledby="navbarDropdown">

    <a style="color: rgb(0, 0, 0);" class="dropdown-item" href="/today">TODAY</a>

    <a style="color: rgb(0, 0, 0);" class="dropdown-item" href="/month">Month</a>

    <a style="color: rgb(0, 0, 0);" class="dropdown-item" href="/year">Year</a>

  </div>

</li>

</ul>

<div class="form-inline my-5 my-lg-1" style="position: absolute; left: 1250px;">

<ul class="navbar-nav mr-auto " > <li class="nav-item dropdown btn-group open">

  <a style="color: rgb(255, 255, 255);" class="btn btn-primary" href="#"><i class="fa fa-user fa-
fw"></i>User </a>

```

```

    <a style="color: rgb(255, 255, 255);" class="btn btn-primary dropdown-toggle "href="#"
id="navbarDropdown" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">

</a>

    <div class="dropdown-menu" aria-labelledby="navbarDropdown">

        <div class="dropdown-divider"></div>

        <a class="dropdown-item" style="color: darkred;" href="/logout">Log-Out <i class="fa fa-
sign-out" aria-hidden="true"></i></a>

    </div>

</li>

<div >

</ul>

</div>

</div>

</nav>

{ % block body % }

{ % endblock % }

</body>

</html>

```

Today.html

```

{ % extends 'base.html' % }

{ % block body % }

<div class="container ">

<div class="row">

```

```

<div class="col-md-5">

  <h3 class="mt-5">Today Expense Breakdown</h3>

  <div class="card shadow mb-2 bg-white rounded-pill">

    <div class="card-body ">

      <div class="row">

        <div class="col-md-6">DATE</div>

        <div class="col-md-6"> AMOUNT  </div>

      </div>

    </div>

  </div>

  <div>

    { % for row in texpense % }

  </div>

  <div class="card shadow mb-2 bg-white rounded-bottom">

    <div class="card-body ">

      <div class="row">

        <div id="ttime" class="col-md-6">{{ row [0] }}</div>

        <div id="tamount" class="col-md-6"> {{ row[1] }}  </div>

      </div>

    </div>

  </div>

  { % endfor % }

</div>

</div>

<section>

  <div class="row">

    <div class="col-md-6">

```

```

<h3 class="mt-5">Expense Breakdown BY Category</h3>

<div class="card shadow mb-2 bg-white rounded-bottom">

  <div class="card-body ">

    <div class="row">

      <div class="col-md-6">Food</div>

      <div id="tfood" class="col-md-6"> {{ t_food}} </div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Entertainment</div>

      <div id="tentertainment" class="col-md-6"> {{ t_entertainment}} </div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Business</div>

      <div id="tbusiness" class="col-md-6"> {{t_business}} </div>

    </div>

  </div>

</div>

```

```

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Rent</div>

      <div id="trent" class="col-md-6">{{ t_rent }} </div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">EMI</div>

      <div id="temi" class="col-md-6">{{ t_EMI }} </div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Other</div>

      <div id="tother" class="col-md-6">{{ t_other }}</div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 btn-outline-danger rounded-pill">

```

```

<div class="card-body">

<div class="row">

  <div class="col-md-6">Total</div>

  <div class="col-md-6">₹ {{total}} </div>

</div>

</div>

</div>

<div class="col-md-6">

<canvas id="myChart" width="400" height="400"></canvas>

<script>

  let food = document.getElementById('tfood').innerHTML

  let entertainment = document.getElementById('tentertainment').innerHTML

  let business = document.getElementById('tbusiness').innerHTML

  let rent = document.getElementById('trent').innerHTML

  let emi = document.getElementById('temi').innerHTML

  let other = document.getElementById('tother').innerHTML

var ctx = document.getElementById('myChart').getContext('2d');

var myChart = new Chart(ctx, {

  type: 'doughnut',

  data: {

    labels: ['Food', 'Entertainment', 'Business', 'Rent', 'EMI', 'Other'],

    datasets: [{

      label: 'Expenses Chart',

      data: [food, entertainment, business, rent, emi, other],

      backgroundColor: [

```



```

        'rgb(255, 99, 132)',
        'rgb(0, 0, 0)',
        'rgb(255, 205, 86)',
        'rgb(201, 203, 207)',
        'rgb(54, 162, 235)',
        'rgb(215, 159, 64)'
    ],

    ]]

    },

    options: {

        responsive: true,

        plugins: {

legend: {

        position: 'bottom',

    },

    title: {

        display: true,

        text: 'EXPENSE BREAKDOWN'

    }

}

    }

    });

</script>

</div>

</div>

```

```

</div>

</section>

</div>

{% endblock %}

```

Month.html

```

{% extends 'base.html' %}

{% block body %}

<div class="container ">

<div class="row">

  <div class="col-md-5">

    <h3 class="mt-5">MONTH Expense Breakdown</h3>

    <div class="card shadow mb-2 bg-white rounded-pill">

      <div class="card-body ">

        <div class="row">

          <div class="col-md-6"> DATE </div>

          <div class="col-md-6"> AMOUNT </div>

        </div>

      </div>

    </div>

  </div>

  <div class="card shadow mb-2 bg-white rounded-bottom">

    <div class="card-body ">

      <div class="row">

        <div id="ttime" class="col-md-6">{{ row [0] }}</div>

```

```

        <div id="tamount" class="col-md-6"> {{row[1] }} </div>

    </div>

</div>

</div>

    {% endfor %}

</div>

</div>

<section>

    <div class="row">

        <div class="col-md-6">

            <h3 class="mt-5">Expense Breakdown BY Category</h3>

            <div class="card shadow mb-2 bg-white rounded-bottom">

                <div class="card-body ">

                    <div class="row">

                        <div class="col-md-6">Food</div>

                        <div id="tfood" class="col-md-6"> {{ t_food }} </div>

                    </div>

                </div>

            </div>

        </div>

        <div class="card shadow mb-2 bg-white rounded">

            <div class="card-body">

                <div class="row">

                    <div class="col-md-6">Entertainment</div>

                    <div id="tentertainment" class="col-md-6"> {{ t_entertainment }} </div>

```

```

</div>

</div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Business</div>

      <div id="tbusiness" class="col-md-6"> {{ t_business }} </div>

    </div>

  </div>

</div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">Rent</div>

      <div id="trent" class="col-md-6"> {{ t_rent }} </div>

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

  <div class="card-body">

    <div class="row">

      <div class="col-md-6">EMI</div>

      <div id="temi" class="col-md-6"> {{ t_EMI }} </div>

```

```

    </div>

  </div>

</div>

<div class="card shadow mb-2 bg-white rounded">
  <div class="card-body">
    <div class="row">
      <div class="col-md-6">Other</div>
      <div id="tother" class="col-md-6"> {{ t_other}} </div>
    </div>
  </div>
</div>

<div class="card shadow mb-2 btn-outline-danger rounded-pill">
  <div class="card-body">
    <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">
  <canvas id="myChart" width="400" height="400"></canvas>
  <script>

```

```

let food = document.getElementById('tfood').innerHTML

let entertainment = document.getElementById('tentertainment').innerHTML

let business = document.getElementById('tbusiness').innerHTML

let rent = document.getElementById('trent').innerHTML

let emi = document.getElementById('temi').innerHTML

let other = document.getElementById('tother').innerHTML

var ctx = document.getElementById('myChart').getContext('2d');

var myChart = new Chart(ctx, {

  type: 'doughnut',

  data: {

    labels: ['Food', 'Entertainment', 'Business', 'Rent', 'EMI', 'Other'],

    datasets: [{

      label: 'Expenses Chart',

      data: [food, entertainment, business, rent, emi, other],

      backgroundColor: [

        'rgb(255, 99, 132)',

        'rgb(0, 0, 0)',

        'rgb(255, 205, 86)',

        'rgb(201, 203, 207)',

        'rgb(54, 162, 235)',

        'rgb(215, 159, 64)'

      ],

    }],

  },

  options: {

    responsive: true,

```

```

        plugins: {
          legend: {
            position: 'bottom',
          },
          title: {
            display: true,
            text: 'EXPENSE BREAKDOWN'
          }
        }
      });
    </script>
  </div>
</div>
</div>
</section>
</div>
{% endblock %}

```

Year.html:

```

{% extends 'base.html' %}
{% block body %}
<div class="container ">
<div class="row">
  <div class="col-md-5">
    <h3 class="mt-5">YEAR Expense Breakdown</h3>
    <div class="card shadow mb-2 bg-white rounded-pill">

```

```

<div class="card-body ">

<div class="row">

    <div value = "month" class="col-md-6"> </div>

    <div class="col-md-6"> AMOUNT </div>

</div>

</div>

</div>

{ % for row in texpanse % }


<div class="card shadow mb-2 bg-white rounded-bottom">

<div class="card-body ">

<div class="row">

    <div id ="ttime" class="col-md-6">{{ row [0] }}</div>

    <div id="tamount" class="col-md-6"> {{ row[1] }} </div>

</div>

</div>

</div>

{ % endfor % }

</div>

</div>

<section>

<div class="row">

<div class="col-md-6">

<h3 class="mt-5">Expense Breakdown BY Category</h3>


<div class="card shadow mb-2 bg-white rounded-bottom">

<div class="card-body ">

<div class="row">

```



```

        <div class="col-md-6">Food</div>

        <div id="tfood" class="col-md-6"> {{ t_food }} </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Entertainment</div>

            <div id="tentertainment" class="col-md-6"> {{ t_entertainment }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Business</div>

            <div id="tbusiness" class="col-md-6"> {{ t_business }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Rent</div>

            <div id="trent" class="col-md-6"> {{ t_rent }} </div>

```

```

    </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">EMI</div>

            <div id="temi" class="col-md-6">{{ t_EMI }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 bg-white rounded">

    <div class="card-body">

        <div class="row">

            <div class="col-md-6">Other</div>

            <div id="tother" class="col-md-6"> {{ t_other }} </div>

        </div>

    </div>

</div>

<div class="card shadow mb-2 btn-outline-danger rounded-pill">

    <div class="card-body">

        <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">

  <canvas id="myChart" width="400" height="400"></canvas>

  <script>

    let food = document.getElementById('tfood').innerHTML

    let entertainment = document.getElementById('tentertainment').innerHTML

    let business = document.getElementById('tbusiness').innerHTML

    let rent = document.getElementById('trent').innerHTML

    let emi = document.getElementById('temi').innerHTML

    let other = document.getElementById('tother').innerHTML

    var ctx = document.getElementById('myChart').getContext('2d');

    var myChart = new Chart(ctx, {

      type: 'doughnut',

      data: {

        labels: ['Food', 'Entertainment', 'Business', 'Rent', 'EMI', 'Other'],

        datasets: [{

          label: 'Expenses Chart',

          data: [food, entertainment, business, rent, emi, other],

          backgroundColor: [

            'rgb(255, 99, 132)',

            'rgb(0, 0, 0)',

            'rgb(255, 205, 86)',

            'rgb(201, 203, 207)',

            'rgb(54, 162, 235)',

            'rgb(215, 159, 64)'

          ],

        }],

      }

    })

  </script>

</div>

```

```
    },
    options: {
      responsive: true,
      plugins: {
        legend: {
          position: 'bottom',
        },
        title: {
          display: true,
          text: 'EXPENSE BREAKDOWN'
        }
      }
    }
  });
</script>
</div>
</div>
</div>
</section>
</div>
{% endblock %}
```

Sendemail.py

```
import os

from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail

message = Mail(

    from_email=os.environ.get('SENDER_EMAIL'),

    to_emails='nivethithashanmuganathan5@gmail.com',

    subject='Sending with Twilio SendGrid is Fun',

    html_content='<strong>and easy to do anywhere, even with Python</strong>')

try:

    sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))

    response = sg.send(message)

    print(response.status_code)

    print(response.body)

    print(response.headers)

except Exception as e:

    print(e)
```

Dockerfile.py

```
FROM python:3.10.4

WORKDIR /pet

COPY . /pet

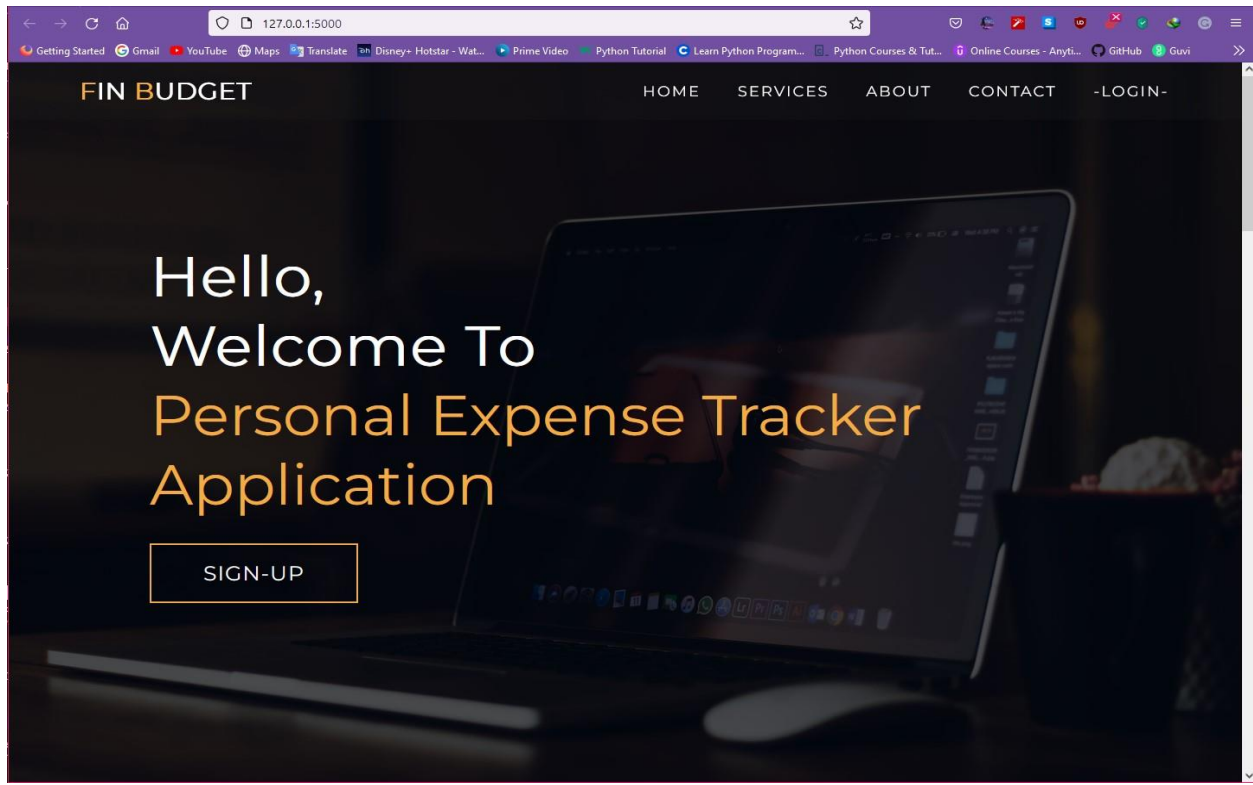
RUN pip install -r requirements.txt

EXPOSE 5000

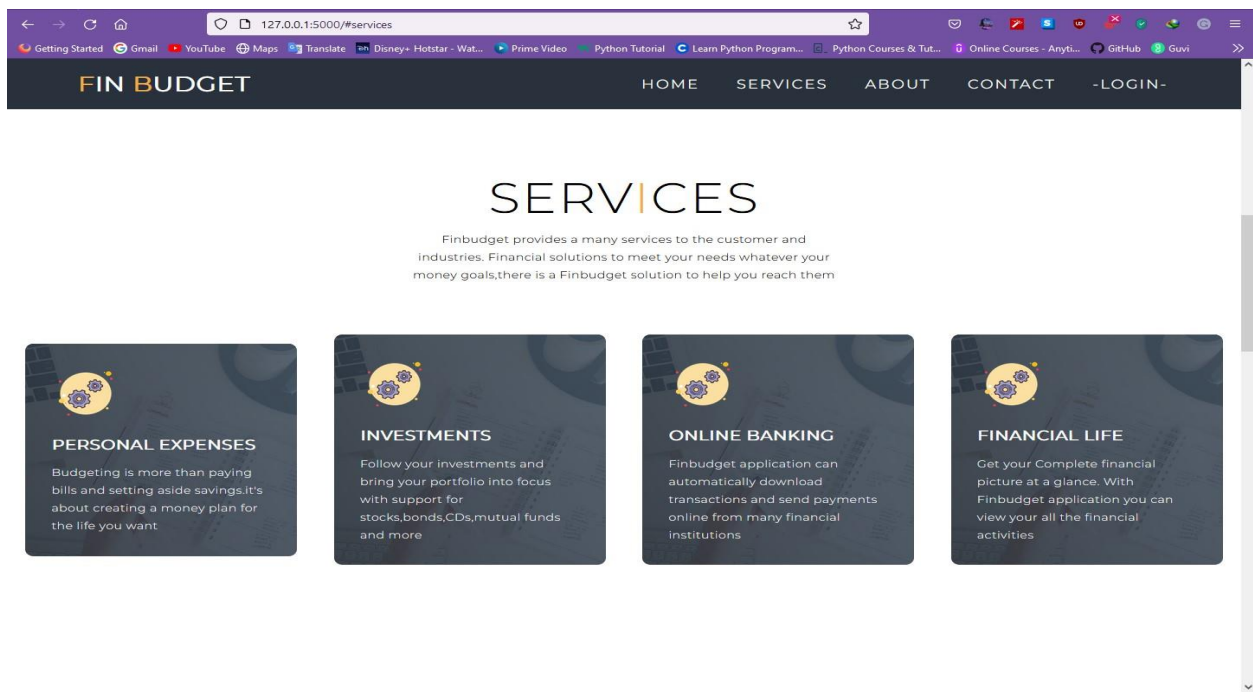
CMD [ "python", "app.py"]
```

OUTPUT:

HOME PAGE



SERVICES



ABOUT

FIN BUDGET

HOME SERVICES ABOUT CONTACT -LOGIN-

ABOUT US

Financial Solution

Fin budget financial solution is one among Leading financial company. Finbudget provides a many services to the customer and industries. Financial solutions to meet your needs whatever your money goals,there is a Finbudget solution to help you reach them. you can Contact our service center for further information and also follow our social media for update on new services

FOLLOW US

Personal Expense Tracker ceo,
Nivethitha S, Newton Habakuk S, Pooja
B, Naveen Kumar S

CONTACT

FIN BUDGET

HOME SERVICES ABOUT CONTACT -LOGIN-

CONTACT INFO

Phone
+91 9944236079
+91 9514095105
+91 9660301238
+91 7010636259

Email
nivethithashanmuganathan5@gmail.com
newttonhabakuks@gmail.com
bpooja37202@gmail.com
naveenkmr.0402@gmail.com

Address
Coimbatore,Tamilnadu,India


FIN BUDGET
Your Complete Financial Solution

Copyright © 2021 . All rights reserved

SIGNUP


localhost:5000/signup


Getting Started Gmail YouTube Maps Translate Disney+ Hotstar - Wat... Prime Video Python Tutorial Learn Python Program... Python Courses & Tut... Online Courses - Anyti... GitHub Guvi




Glad to see you

Hello, Friend

 Name


 Example@gmail.com

 Password

☐ I read and agree to [Terms & Conditions](#)

CREATE ACCOUNT

Already have an account? [Sign in](#)




Welcome, Please Fill in the blanks for sign up


LOGIN

signup'."/>


localhost:5000/signin


Getting Started Gmail YouTube Maps Translate Disney+ Hotstar - Wat... Prime Video Python Tutorial Learn Python Program... Python Courses & Tut... Online Courses - Anyti... GitHub Guvi





WELCOME

 Email

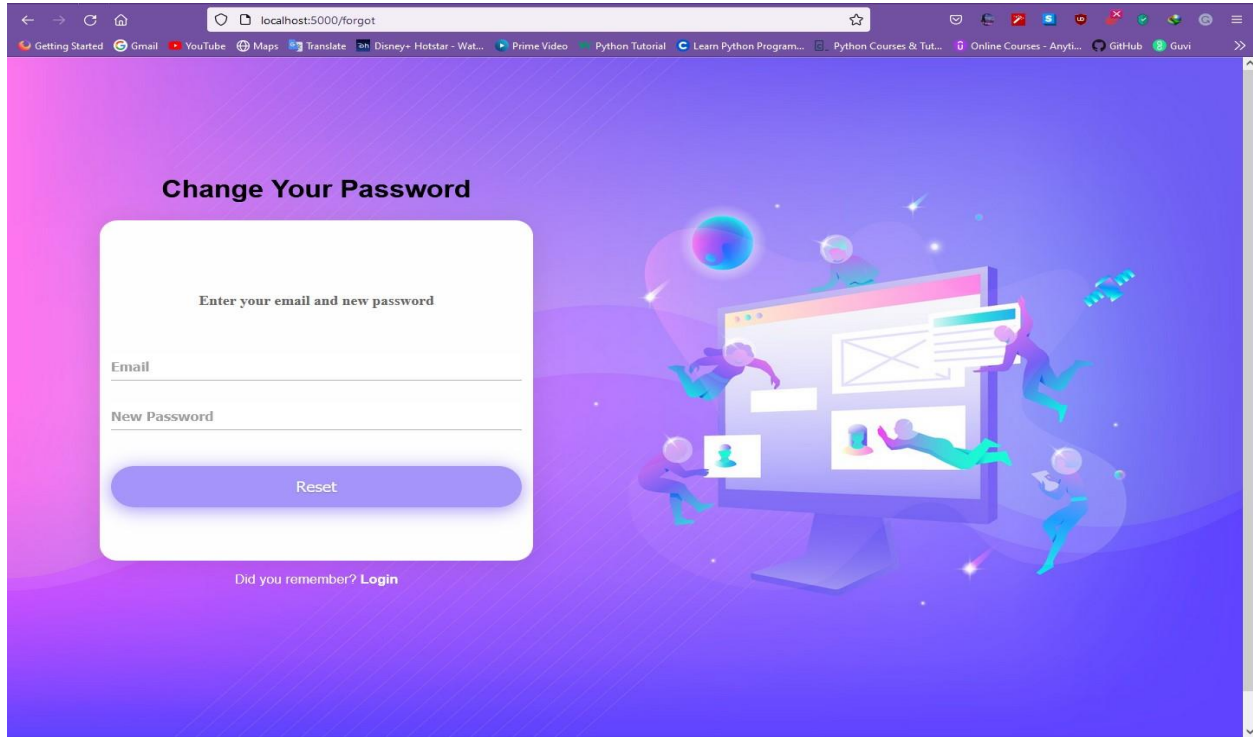
 Password

[Forgot Password?](#)

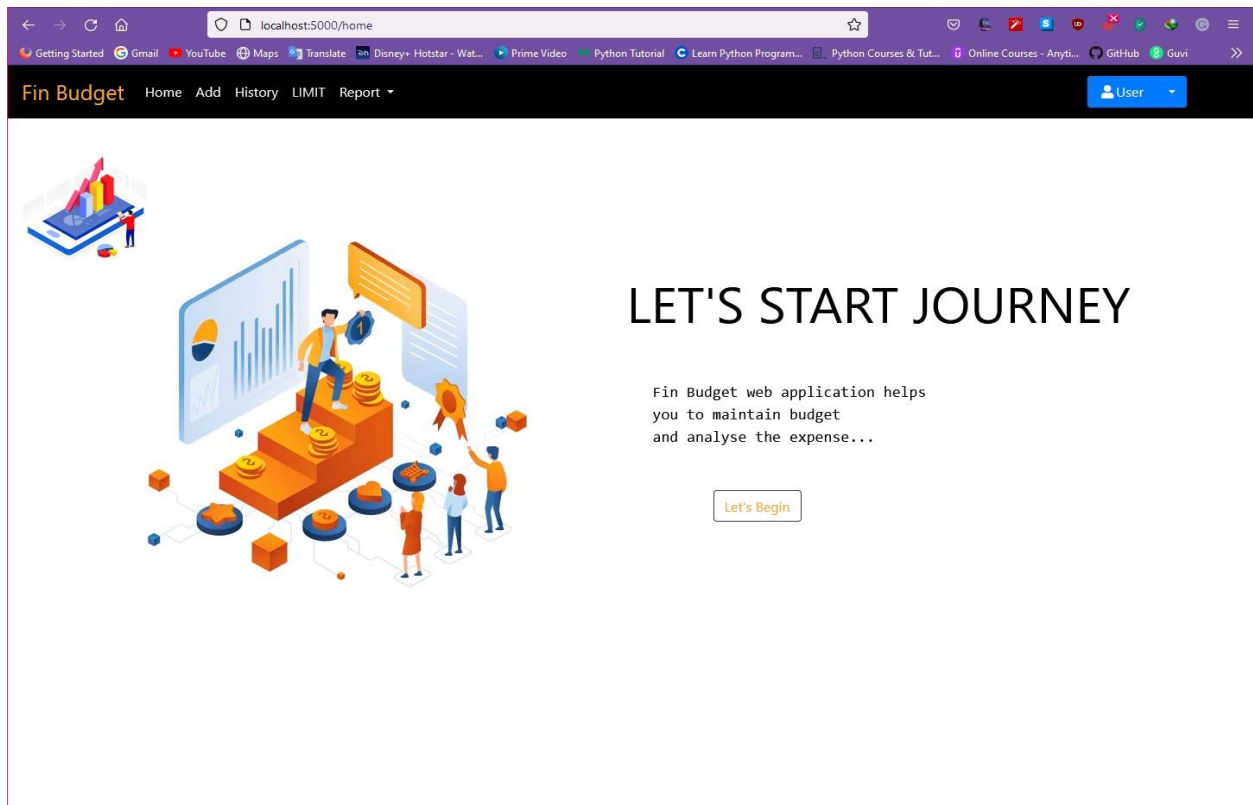
LOGIN

Don't have an account? [signup](#)

FORGOT



DASHBOARD



ADD EXPENSES

localhost:5000/add

Fin Budget Home Add History LIMIT Report User

Add Expense

Date: 19/11/2022


Expense name: Biryani

Expense Amount: 250

debitcard

food

Add



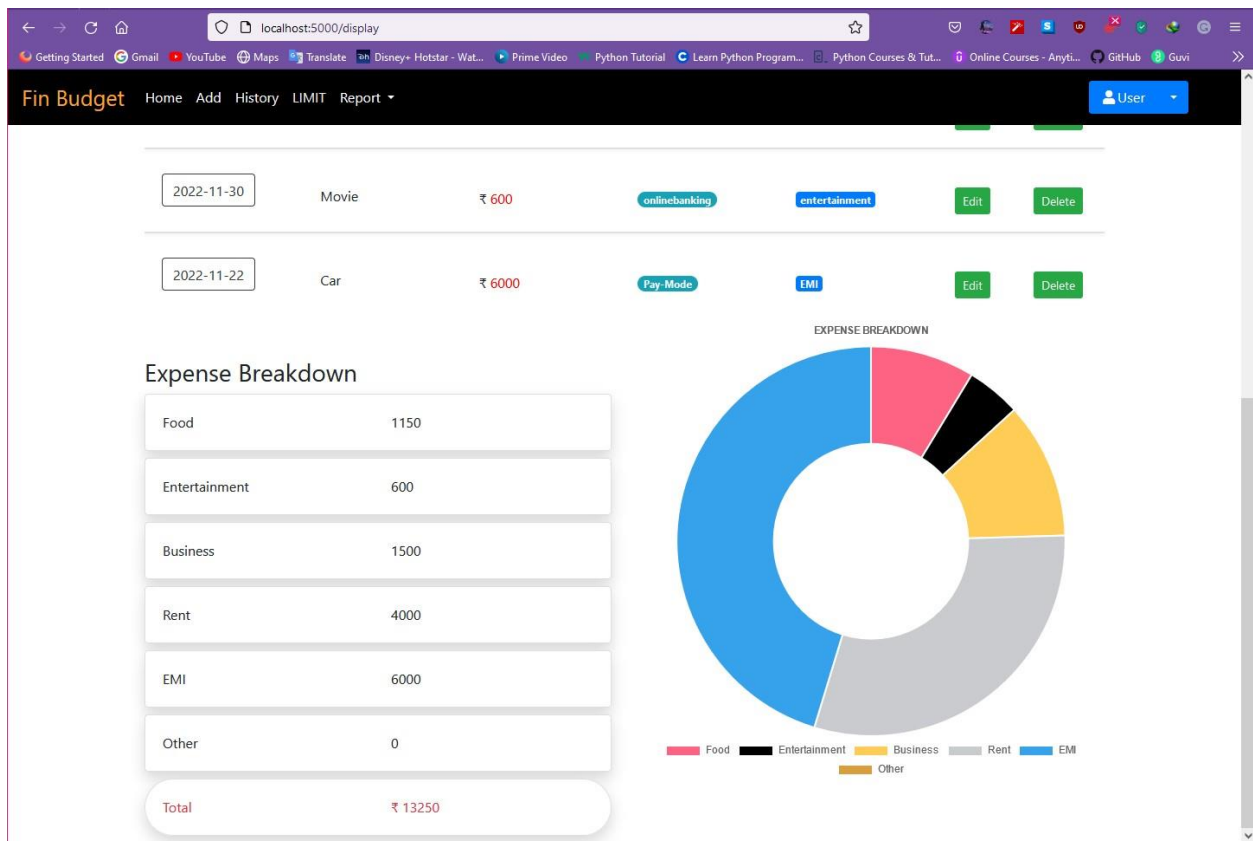
HISTORY – GRAPH OVERVIEW

localhost:5000/display

Fin Budget Home Add History LIMIT Report User

EXPENSES

2022-11-14	biryani	₹ 500	cash	food	Edit	Delete
2022-11-16	rent	₹ 1000	cash	rent	Edit	Delete
2022-11-13	party	₹ 1500	debitcard	business	Edit	Delete
2022-11-15	food	₹ 400	cash	food	Edit	Delete
2022-11-19	Biryani	₹ 250	debitcard	food	Edit	Delete
2022-11-29	Bills	₹ 3000	epayment	rent	Edit	Delete
2022-11-30	Movie	₹ 600	onlinebanking	entertainment	Edit	Delete
2022-11-22	Car	₹ 6000	Pay Mode	EMI	Edit	Delete



EDIT EXPENSE

localhost:5000/edit/11

Fin Budget Home Add History LIMIT Report User

Edit Expense

Date: 24 / 11 / 2022

Expense name: Car

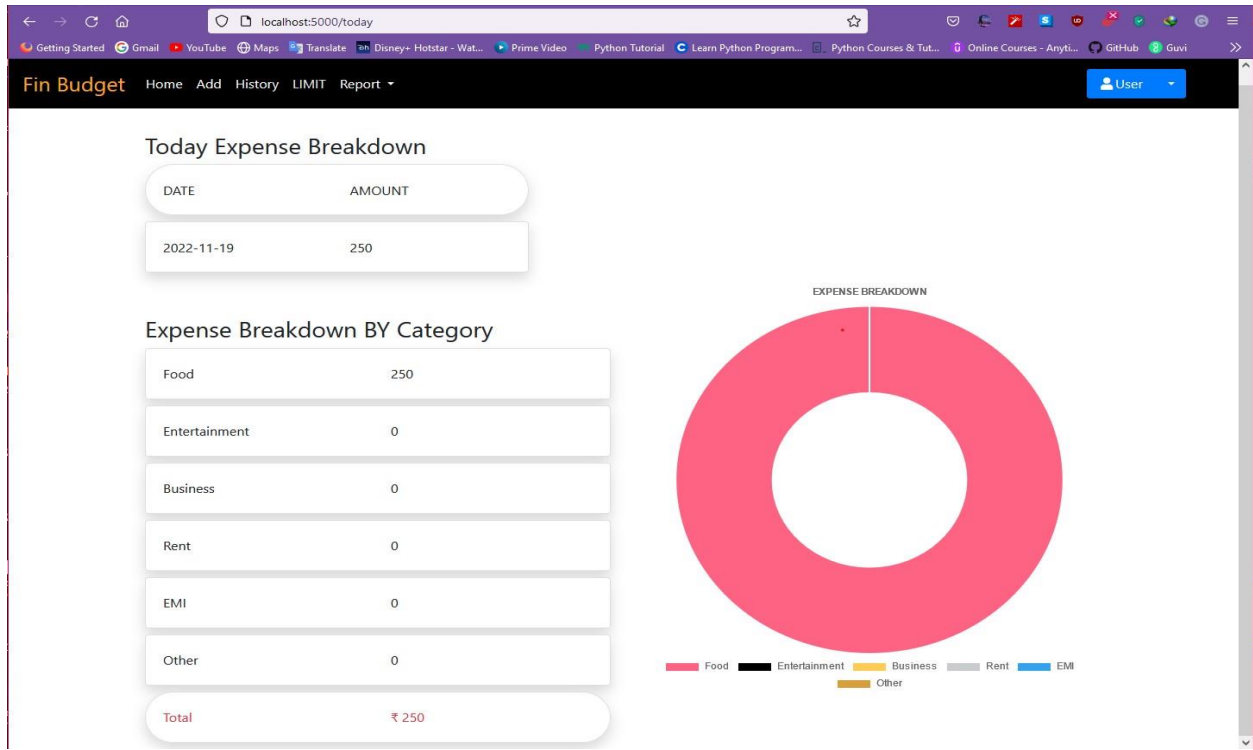
Expense Amount: 6000

Pay-Mode: Pay-Mode

EMI: EMI

Update

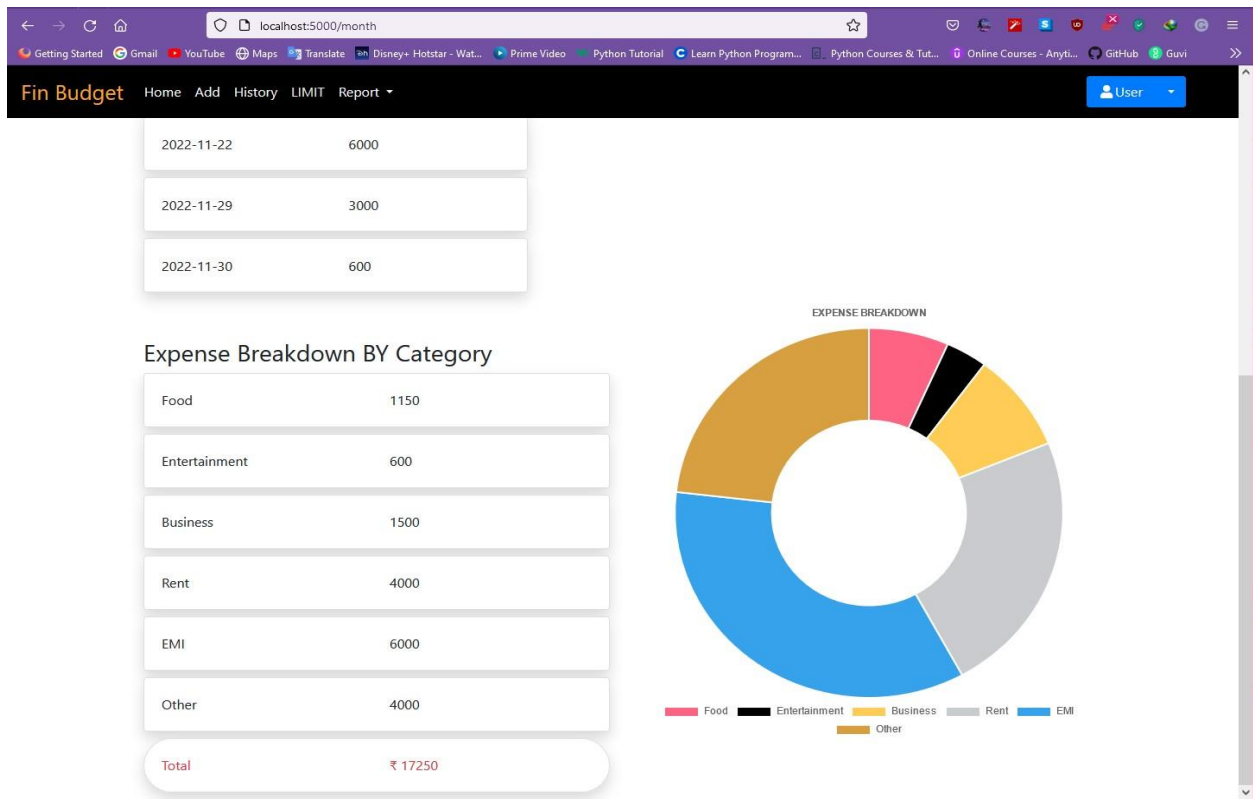
TODAY'S REPORT



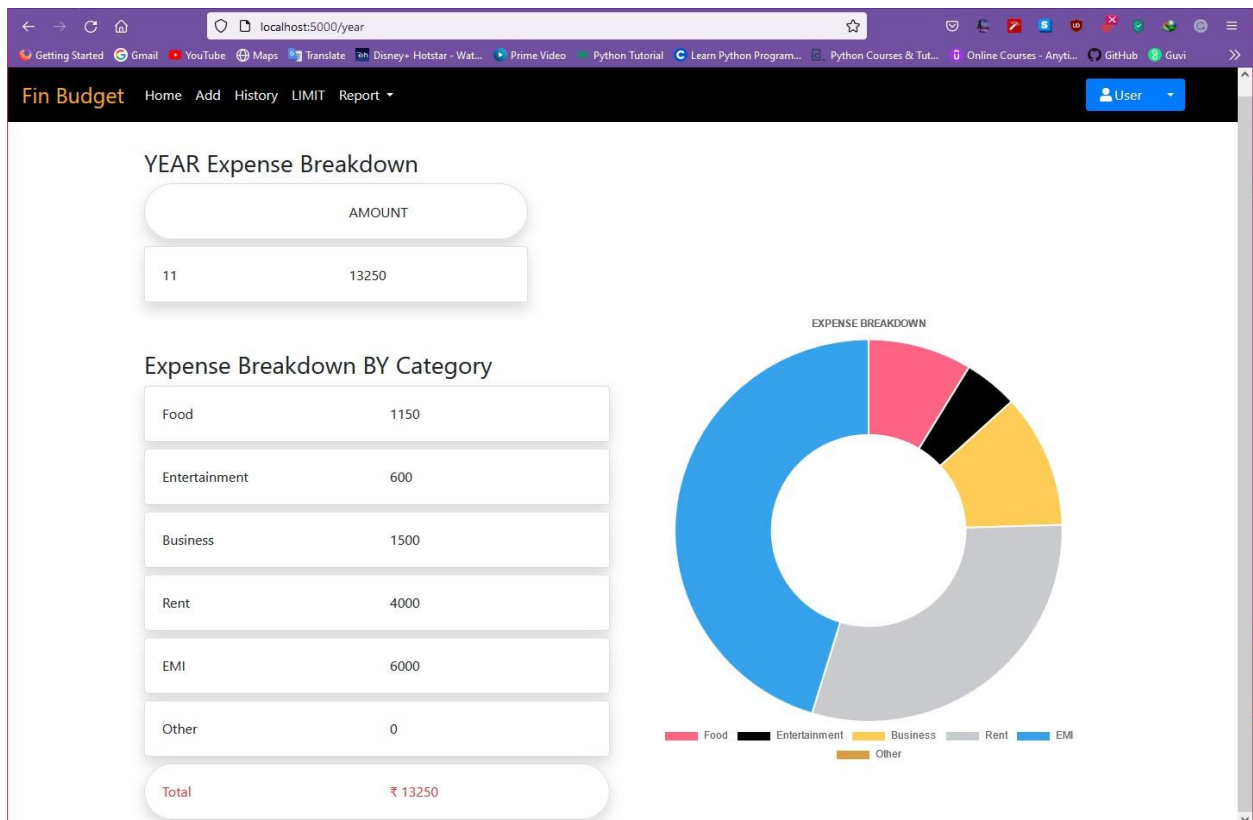
MONTHLY REPORT

MONTH Expense Breakdown

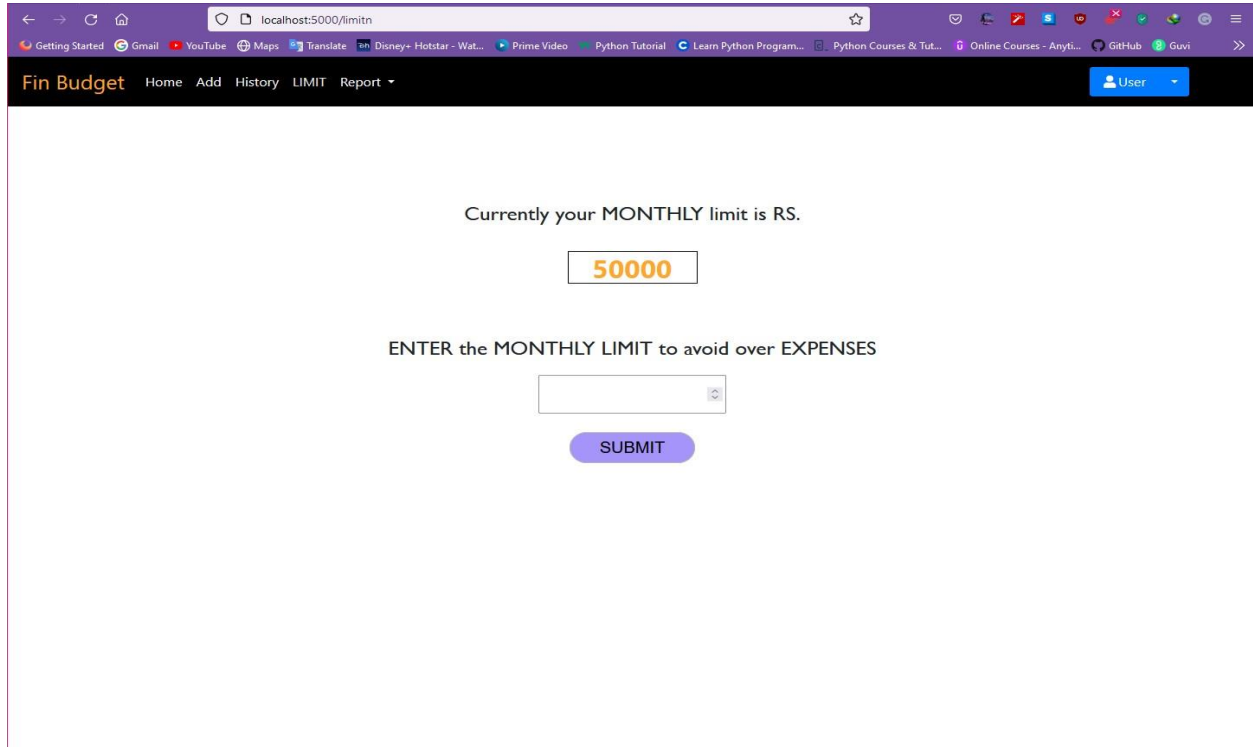
DATE	AMOUNT
2022-11-13	1500
2022-11-14	500
2022-11-15	400
2022-11-16	1000
2022-11-19	250
2022-11-20	4000
2022-11-22	6000
2022-11-29	3000
2022-11-30	600



YEARLY REPORT



EXPENSE LIMIT



localhost:5000/limitn

Getting Started Gmail YouTube Maps Translate Disney+ Hotstar - Wat... Prime Video Python Tutorial Learn Python Program... Python Courses & Tut... Online Courses - Anyti... GitHub Guvi

Fin Budget Home Add History LIMIT Report User

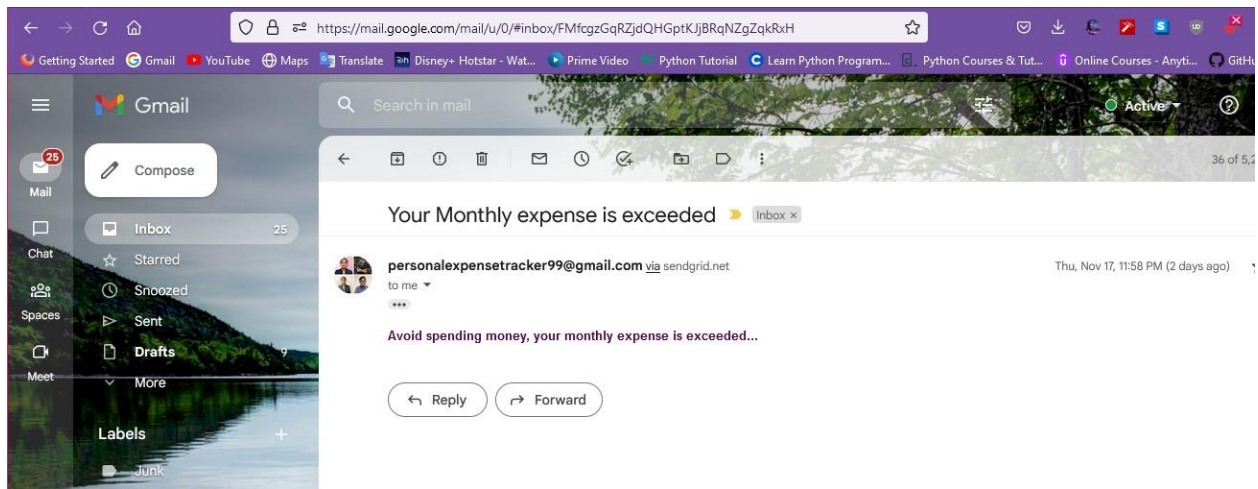
Currently your MONTHLY limit is RS.

50000

ENTER the MONTHLY LIMIT to avoid over EXPENSES

SUBMIT

SEND ALERT EMAIL



13.2 GITHUB & PROJECT DEMO LINK

GitHub - <https://github.com/IBM-EPBL/IBM-Project-46735-1660755122>

Project Demo link - <https://bit.ly/3Axa7oe>

CHAPTER 14

REFERENCES

- [1] Dr.C K Gomathy,Article: A Study On The Recent Advancements In Online Surveying , International Journal Of Emerging Technologies And Innovative Research (JETIR) Volume 5 | Issue 11 | ISSN : 2349-5162, P.No:327-331, Nov-2018.
- [2] Atiya Kazi , Praphulla S. Kherade, Raj S. Vilankar, Parag M. Sawant Expense Tracker © May 2021 | Ire Journals | Volume 4 Issue 11 | Issn: 2456-8880.
- [3] Muskaan Sharma, Ayush Bansal, Dr. Raju Ranjan, Shivam Sethi School Of Computer Science And Engineering, Galgotias University A Novel Expense Tracker Using Statistical Analysis © June 2021| IJIRT | Volume 8 Issue 1 | ISSN: 2349-6002 IJIRT.
- [4] Mayan, J.A., Priya, K.L., " Novel approach to reuse unused test cases in a GUI based application", IEEE International Conference on Circuit, Power and Computing Technologies, ICCPCT 2020.
- [5] Sankari.A,Albert Mayan.J,"Retrieving Call Logs And SMS By Messaging Services",International Journal Of Pharmacy & Technology,Vol. 8 , Issue No.4 , Pp.22951-22958,Dec 2016.
- [6] Bane, "Expense Tracker (Class Diagram (UML)," 2014.
- [7] L. T. Hong, "Android Mobile Application – Expenses With Geo-Location Tracking,University Tulku Abdul Rahman, Utar, 2015.
- [8] L. T. Hong, "Android Mobile Application – Expenses With Geo-Location Tracking,University Tulku Abdul Rahman, Utar, 2015.
- [9] Mary Posonia A, S. Vigneshwari, Albert Mayan J, D. Jamunarani,"Service Direct:Platform That Incorporates Service Providers And Consumers Directly" ,International Journal Of Engineering And Advanced Technology (IJEAT), Vol.8 ,No.6,Pp.3301-3304,2019.
- [10] Velmurugan A, Albert Mayan J, Niranjana P And Richard Francis "Expense Manager Application " Journal Of Physics: Conference Series,2020.