## PROJECT REPORT

Personal Expense Tracker Application

Domain : Cloud App Development

ID : PNT2022TMID04806

College Name : Kongu Engineering College

Department : Information Technology

V.GOKHULAVAASAN K.GOPINATH M.GUNANITHI V.HARI

(737819ITR021) (737819ITR022) (737819ITR023) (737819ITR024)

**TABLE OF CONTENTS**

1. INTRODUCTION……………………………………………………………………………...... 3

1.1. Project Overview...................................................................................................................... 3

1.2. Purpose..................................................................................................................................... 3

2. LITERATURE SURVEY……………………………………………………………………...... 4

2.1. Existing problem...................................................................................................................... 4

2.2. References................................................................................................................................ 5

2.3. Problem Statement Definition.................................................................................................. 5

3. IDEATION & PROPOSED SOLUTION……………………………………………………...... 6

3.1. Empathy Map Canvas.............................................................................................................. 6

3.2. Ideation & Brainstorming........................................................................................................ 6

3.3. Proposed Solution.................................................................................................................... 7

3.4. Problem Solution fit................................................................................................................. 8

4. REQUIREMENT ANALYSIS………………………………………………………………...... 9

4.1. Functional requirement............................................................................................................ 9

4.2. Non-Functional requirements.................................................................................................. 10

5. PROJECT DESIGN……………………………………………………………………............... 11

5.1. Data Flow Diagrams................................................................................................................ 11

5.2. Solution & Technical Architecture.......................................................................................... 12

5.3. User Stories.............................................................................................................................. 13

6. PROJECT PLANNING & SCHEDULING……………………………………………….......... 15

6.1. Sprint Planning & Estimation.................................................................................................. 16

6.2. Sprint Delivery Schedule......................................................................................................... 19

7. CODING & SOLUTIONING………………………………………………………………........ 20

7.1. Feature 1................................................................................................................................... 20

8. TESTING……………………………………………………………………............................... 21

8.1. Test Cases................................................................................................................................ 21

9. RESULTS…………………………………………………………………….............................. 24

9.1. Performance Metrics................................................................................................................ 24

10. ADVANTAGES & DISADVANTAGES…………………………………………………….... 25

11. CONCLUSION…………………………………………………………………….................... 26

12. FUTURE SCOPE……………………………………………………………………................ 27

13. APPENDIX……………………………………………………………………......................... 28

13.1. Source Code.......................................................................................................................... 28

13.2. GitHub & Project Demo Link............................................................................................... 33

## **CHAPTER 1**

## **1.INTRODUCTION**

A Personal Expense Tracker Application is a particular form of digital diary that aids in keeping track of all of our cash transitions and moreover offers daily, weekly, monthly, and yearly reports on all financial activities .User receives alerts to keep track of income and expenses that can system for tracking the application. All data is kept in offline mode for easy access at any time and from any location. The Daily Expense Tracker's user interface is incredibly straightforward and appealing, making it simple to grasp and the finest approach to record our financial data.

### **1.1. PROJECT OVERVIEW**

Simply put, personal finance includes all of the financial decisions and actions that a finance software facilitates by assisting you in effectively managing your finances. A personal finance software will not only assist you with accounting and budgeting, but it will also provide you with valuable advice on money management .Users of personal finance applications will be prompted to enter their costs, after which their wallet balance will be updated and displayed to them. Users can also receive a graphical analysis of their expenses. They can choose to establish a cap on how much can be used in that month, and if the cap is surpassed, the user will receive an email alert.

### **1.2. PURPOSE**

When you keep track of your spending, you can make sure your money is being utilized wisely and you will know where it goes. You can learn why you're in debt and how you got there by keeping track of your spending. You can then use this information to create a debt relief plan that works for you .You may plan for both short-term and long-term expenses by using a budget to make sure you're not spending more than you're earning. It's a simple, practical solution for folks with all types of income and expenses to maintain order in their finances.

## 

## **CHAPTER 2**

## **2. LITERATURE SURVEY**

A literature review is a piece of academic writing that places the academic literature on a particular topic in perspective to show knowledge of and understanding of it. This chapter shows the different techniques that have been implemented.

### **2.1. EXISTING PROBLEM**

The expense tracker existing system does not offer the user portable device management level, is only used on desktop software, and is therefore impossible to update anywhere expenses are done and is unable to update the location of the expense details disrupting that the proposed system provides. The user's daily, weekly, and monthly spending must be maintained in Excel sheets and CSV files at the moment. The ability to conveniently keep track of one's everyday costs does not now have a fully comprehensive answer.To do this, one must maintain a journal in a diary or computer system, and all calculations must be made by the user, which might occasionally result in errors that cause losses. Due to imperfect data maintenance, the current system is not user friendly. The sole negative where the rest are absent from this endeavor is that there will be no reminder to stay a human on a specified date. This project won't have any information because it doesn't remind people to do anything each month, which has some drawbacks. However, it can be used to calculate income and expenses, so we suggest a new project to solve this issue.

### **2.2 REFERENCE**

[1] Expense Tracker ATIYA KAZI, PRAPHULLA S. KHERADE, RAJ S. VILANKAR, PARAG M. SAWANT May 2021

[2] Intelligent Online Budget Tracker Girish Bekaroo and Sameer Sunhaloo Proceedings of the 2007 Computer Science and 2007

[3] Online Income and Expense Tracker S. Chandini, T. Poojitha, D. Ranjith, V.J. Mohammed Akram, M.S. Vani, V. Rajyalakshmi Mar 2019

[4] Family Expense Manager Application Rajaprabha M N 2017

[5] A Novel Expense Tracker using Statistical Analysis Muskaan Sharma, Ayush Bansal, Dr. Raju Ranjan, Shivam Sethi June 2021

[6] Expense Tracker Hrithik Gupta, Anant Prakash Singh, Navneet Kumar and J. Angelin Blessy December 2020

[7] Expense Manager: An Expense Tracking Application using Image Processing Nupur Sawarkar, Pranay Yenagandula, Devang Shetye, Prof. Shruti Agrawal April 2022

[8] D2D Expense Tracker Application Anjali Kumar, Utkarsh Ra, Aman Kumar 2021

[9] Daily Expense Tracker Mobile Application Nuura Najati Binti Mustafa 2021

[10] Daily Expense Tracker Shivam Mehra, Prabhat Parashar 2021

### **2.3 PROBLEM STATEMENT DEFINITION**

In our daily life money is the most important portion and without it we cannot last one day on earth but if we keep on track all financial data then we can overcome this problem. Most of the people cannot track their expenses and income one way they face the money crisis and depression. This situation motivates us to make an android app to track all financial activities. Using the Personal Expense Tracker Application user can be tracking expenses day to day and making life tension free.

## 

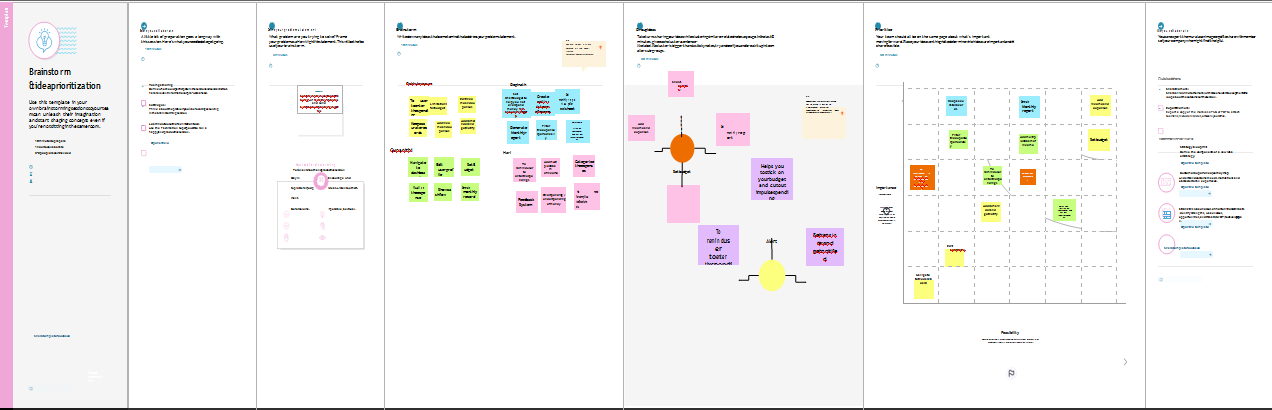
## **CHAPTER 3**

## **3. IDEATION & PROPOSED SOLUTION**

### **3.1 EMPATHY MAP CANVAS**



### **3.2. IDEATION & BRAINSTORMING**

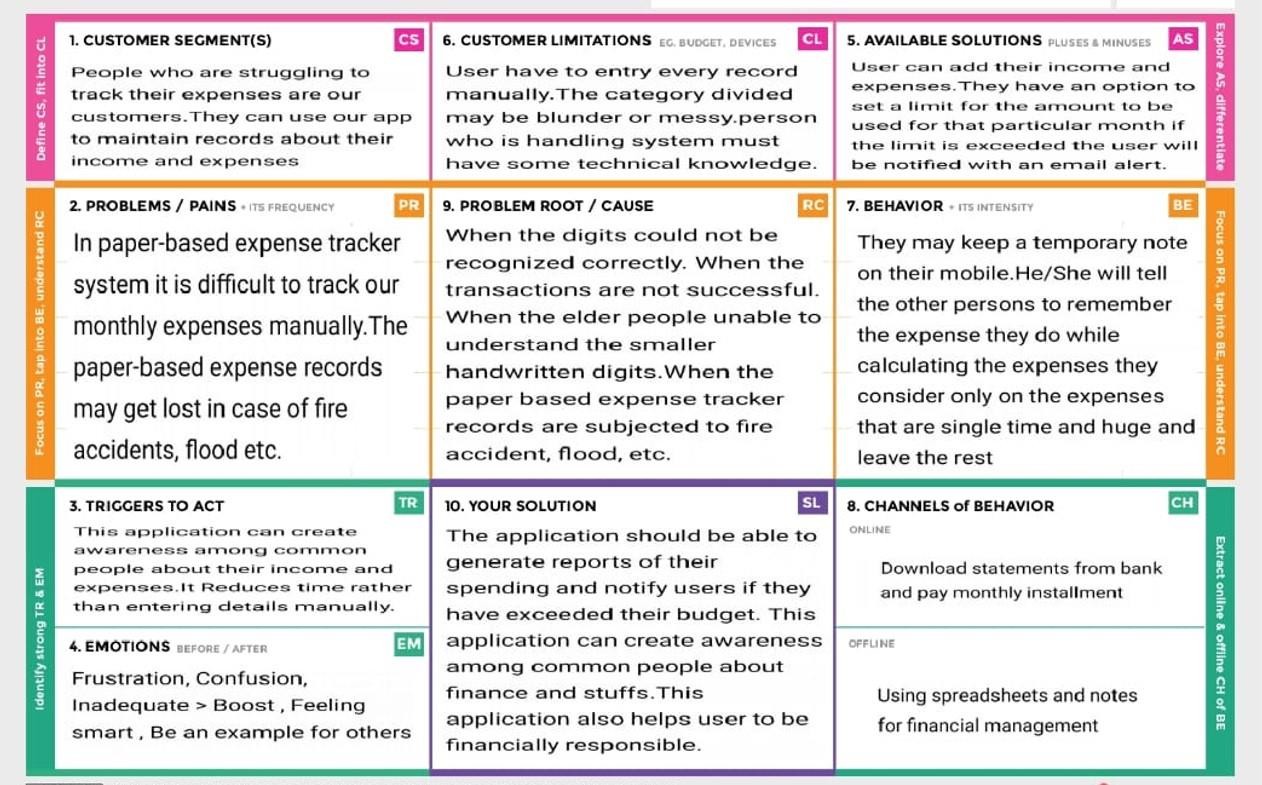


**3.3. PROPOSED SOLUTION**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **Parameter** | **Description** |
| 1. | Problem Statement | It is difficult to manually track monthly expenses in a paper cost tracking system. It is difficult to manually track monthly expenses in a paper cost tracking system. Paper expense reports can be lost. fire, flood, etc. |
| 2. | Scalability of the Solution | This programme has the performance and security to manage a huge number of users and data. This programme can be configured for both large- and small-scale uses. Simple to access on a variety of devices. |
| 3. | Idea / Solution description | A daily expense management system was created specifically for salaried and unpaid employees to keep track of their daily expenses in a simple and efficient manner through a computerised system that tends to reduce manual paper work. Users of personal finance applications will be prompted to enter their costs, after which their wallet balance will be updated and displayed to them. They can choose to establish a cap on how much can be used in that month, and if the cap is surpassed, the user will receive an email alert. |
| 4. | Novelty / Uniqueness | When the user's expenses go over the limit, they are alerted, and they are also reminded when they forget to make an entry. SMS-based spending tracking. analytics of financial data. Cost projections for the future |

|  |  |  |
| --- | --- | --- |
| 5. | Social Impact / Customer Satisfaction | The programme should be able to produce reports on users' spending and alert them when they go over their budget. In order to create the prediction, it is intended to be dynamic. Additionally, it offers users' personal data, their earnings, and their expenses. This programme can educate regular people about money and other topics. Additionally, this app supports user financial responsibility. As opposed to manually entering information, it saves time. |
| 6. | Business Model (Revenue Model) | This application is offered without charge. However, there will be some advertising. The premium edition has extra features and is free of advertisements. |

### **3.4. PROBLEM SOLUTION FIT**



## **CHAPTER 4**

## **4. REQUIREMENT ANALYSIS**

### **4.1 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 | Form for collecting details |
| FR-2 | Login | Enter username and password |
| FR-3 | Calendar | Personal expense tracker application must allow user to add the data to their expenses. |
| FR-4 | Expense Tracker | 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 | Category | This application shall allow users to add categories of their expenses. |

### 

### **4.2 Non Functional Requirements**

Following are the non functional requirements of the Proposed solution

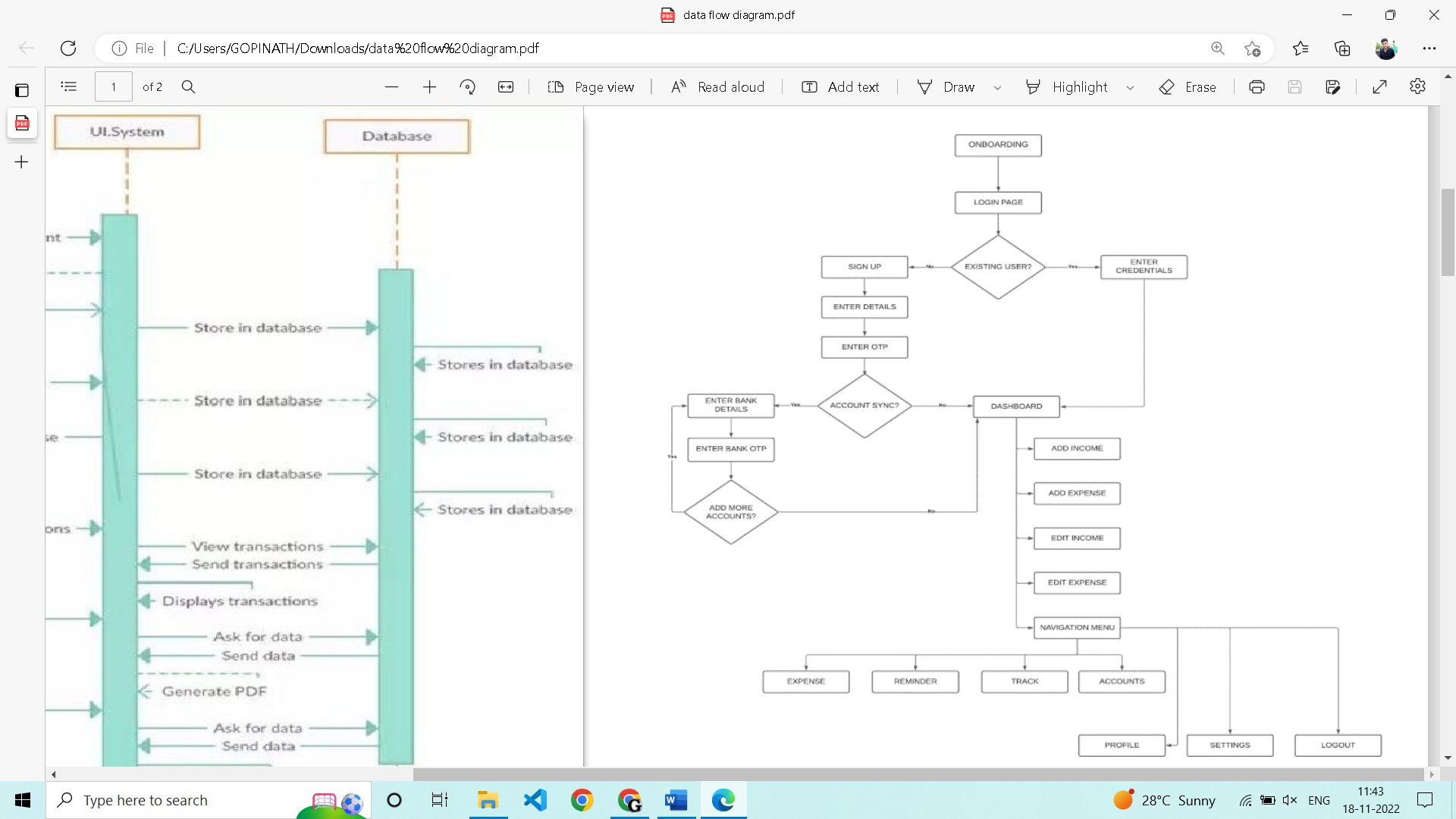
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirements** | **Description** |
| NFR-1 | **Usability** | Helps to keep an accurate record of your income and expenses. |
| NFR-2 | **Security** | Budget tracking apps are considered very safe from those  who commit cyber crimes. |
| NFR-3 | **Reliability** | Each data record is stored on a well built efficient database schema. There is no risk of data loss. |
| NFR-4 | **Performance** | The types of expense are categories along with an option. Throughput of the system is increased due to  light weight database support. |
| NFR-5 | **Availability** | The application must have a 100% up-time. |
| NFR-6 | **Scalability** | The ability to appropriately handle increasing demands. |

## **CHAPTER 5**

## **5. PRODUCT DESIGN**

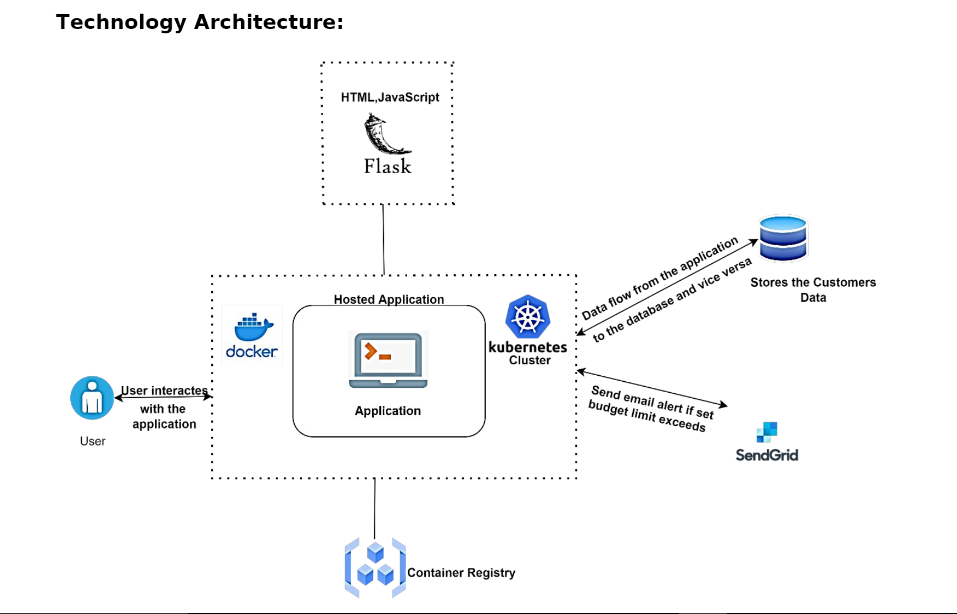
### **5.1 Data Flow Diagrams****:**

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.



### **5.2 Technical Architecture**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



### 

### **5.3 User Stories**

Use the below template to list all the user stories of the product

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Customer (Mobile user & web user ) | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | I can access my account / dashboard | High |  |
|  |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | I can receive confirmation email & click confirm | High |  |
|  |  | USN-3 | As a user, I can register for the application through Facebook | I can register & access the dashboard with Facebook Login | Low |  |
|  | Login | USN-4 | As a user, I can log into the application by entering email & password | I can access the application | High |  |
|  | Dashboard | USN-5 | As a user I can enter my income and expenditure details. | I can view my daily expenses | High |  |
| Customer Care Executive |  | USN-6 | As a customer care executive I can solve the log in issues and other issues of the application. | I can provide support or solution at any time 24\*7 | Medium |  |
| Administrator | Application | USN-7 | As an administrator I can upgrade or update the application. | I can fix the bug which arises for the customers and users of the application | Medium |  |

## 

## **CHAPTER 6**

## **6. PROJECT PLANNING & SCHEDULING**

### **6.1 Sprint Planning and Estimation**

## 

| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story**  **Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirmingmy password. | 3 | High | Aiswarya. S  Danujaa. R |
| Sprint-1 | USN-2 | As a user, I will receive confirmation email once  I have registered for the application | 3 | High | Aiswarya. S  Danujaa. R |
| Sprint-1 | Login | USN-3 | As a user, I can log into the application by  entering email & password | 5 | High | Aiswarya. S  Danujaa. R |
| Sprint-1 | Dashboard & Logout | USN-4 | As a user, once I logged in I can access all the features of the web app and Logout once I  completed all the work. | 5 | High | Aiswarya. S  Danujaa. R |
| Sprint-1 | USN-5 | Once logged In, Keep me logged for few hours  to avoid repeated login if the page is refreshed | 4 | Medium | Aiswarya. S  Danujaa. R |
| Sprint-2 | Expense | USN-6 | Add total income for the month and Allow for  edit option | 6 | High | Sharmitha. S  Sneha ganesh |
| Sprint-2 | USN-7 | Split the total income based on usage like  entertainment, food, shopping etc. | 2 | Low | Sharmitha. S  Sneha ganesh |
| Sprint-2 | USN-8 | Add the day to day expense. | 6 | High | Sharmitha. S  Sneha ganesh |
| Sprint-2 | USN-9 | Display the user added expense | 6 | High | Sharmitha. S  Sneha ganesh |
| Sprint-3 | USN-10 | Filter the expense data based on criteria | 6 | Medium | Aiswarya. S  Danujaa. R |

| Sprint-3 | Charts | USN-11 | As a user I can display it in graphs | 4 | Low | Aiswarya.S  Danujaa. R |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-3 | Alerts | USN-12 | As a user I create custom alert for the balance | 10 | High | Aiswarya. S  Danujaa. R |
| Sprint-4 | Deployment | USN-13 | As a user I should able to access it anywhere in the net | 20 | High | Sharmitha. S  Sneha ganesh |

### 

### **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 | 24 Oct 2022 | 29 Oct 2022 | 20 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 05 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 12 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 19 Nov 2022 |

## **CHAPTER 7**

### **7.1 Feature 1**

1. Track revenues & expenses.

2. Managing transaction receipts and records.

3. Record & arrange receipts

4. Paying taxes in time.

5. Processing payment and invoices.

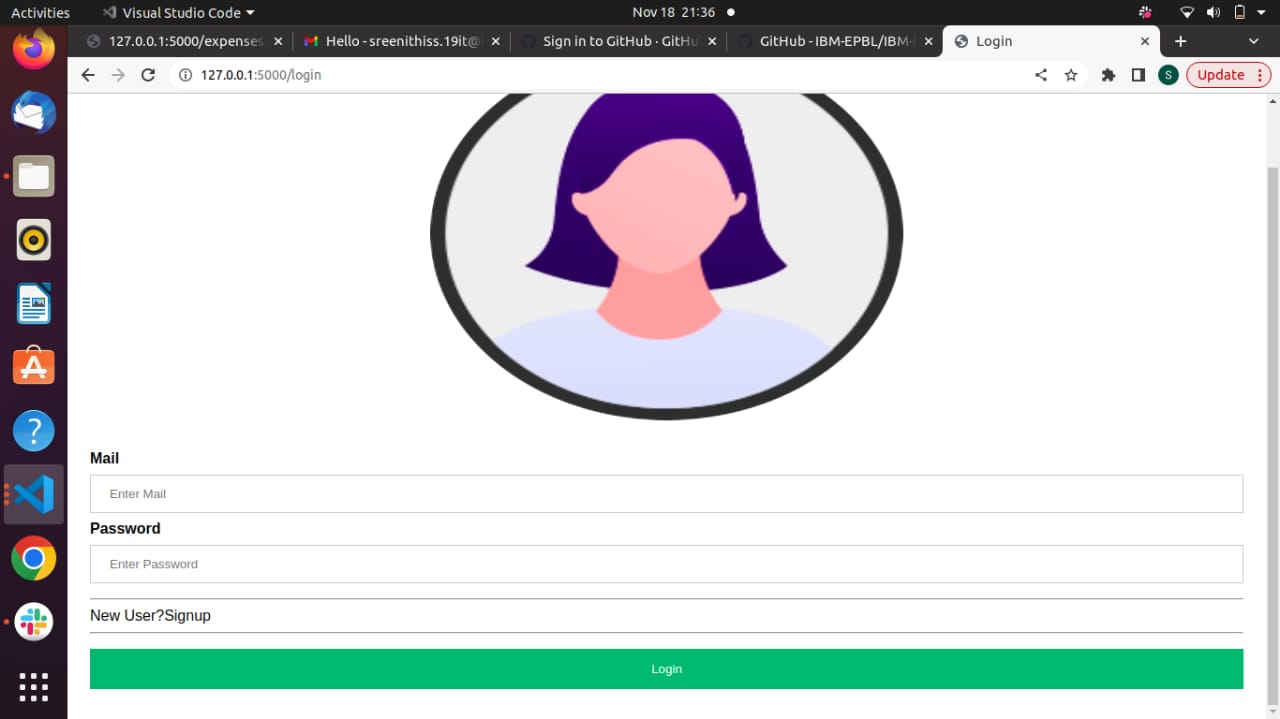
6. Create in-depth reports.

## **CHAPTER 8**

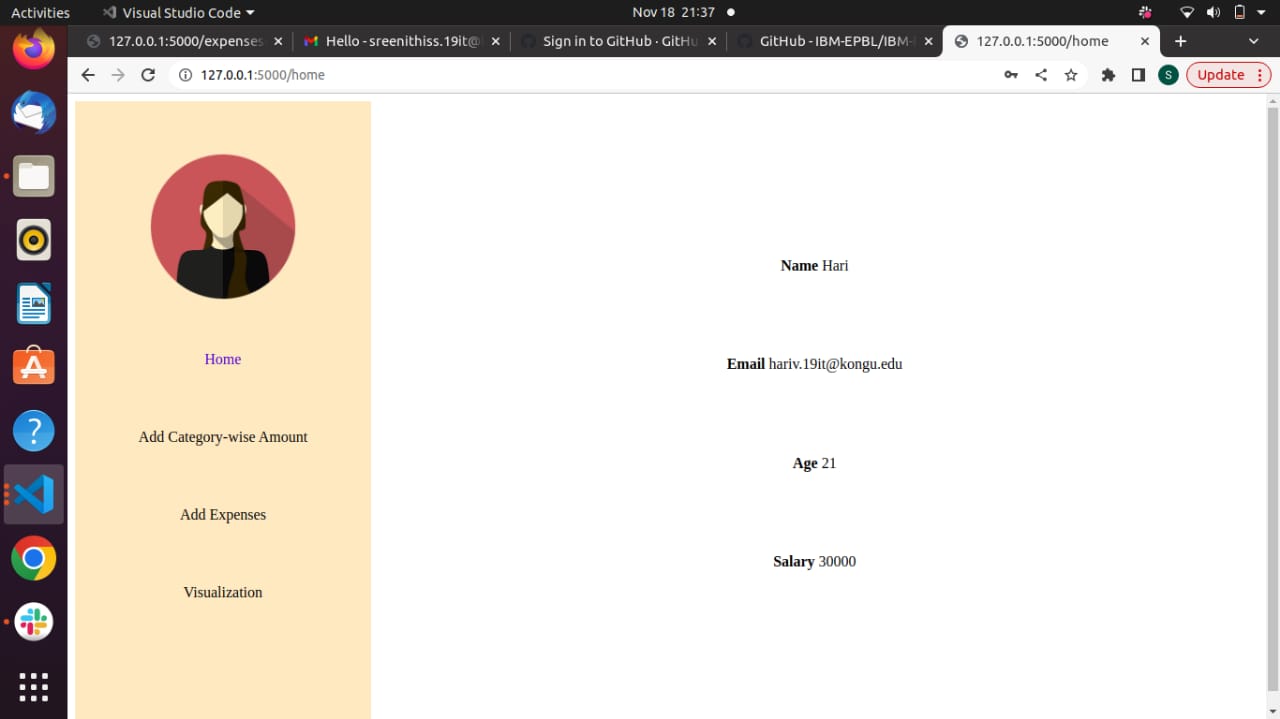
## **TESTING**

### **8.1 Test Cases**

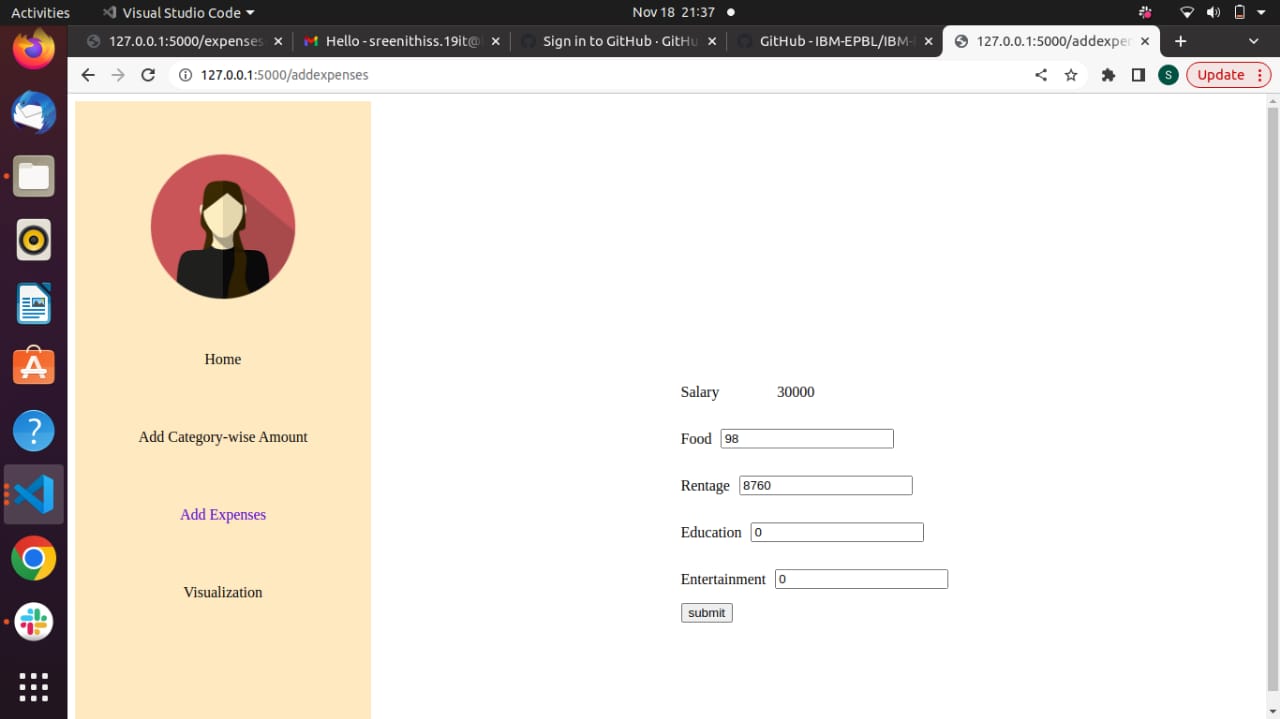
**Login Page**



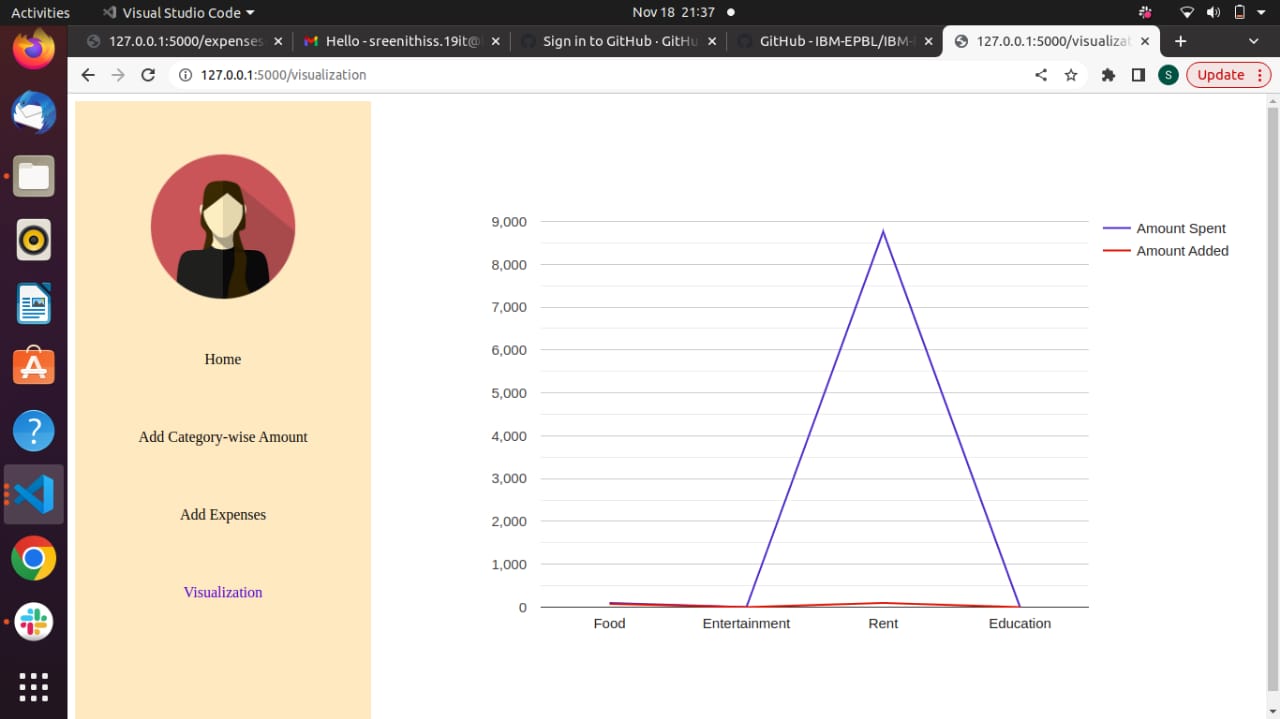
**HOME PAGE**

****

**ADD EXPENSES**



**VISUALIZATION**

****

## **CHAPTER 9**

## **RESULTS**

### **9.1Performance Metrics**

An application can be a very powerful tool for businesses if once the app becomes a success. However, the success of an app is measured through numbers, metrics, and analytics. Developing an app takes quite a lot, so once you’ve dedicated much time, money, and effort to the process, it’s mandatory to measure mobile app performance.

## **CHAPTER 10**

## **ADVANTAGES & DISADVANTAGES**

### **10.1 Advantages**

• Improved customer service

• Cloud-based solution

• Order Fulfillment

• Harness Customer Loyalty and Retention

### **10.2 Disadvantages**

• System Clash

• Reduced Physical Audits

• No solution to improve or eliminate bottlenecks in the service cycle

## **CHAPTER 11**

## **11. CONCLUSION**

Taking proper care of our record is crucial in every business, no matter how big or little, we must understand. We must educate ourselves about the idea of effective inventory management and its applications because we can see that managers do not fully grasp it. A company's inventory management system is one of the reasons for its failure. Many customs to combat failure are present, and we can start from this point. Modern technologies can support us in managing and keeping an eye on our inventory. We may learn, put new ideas into practice, and assess our company.

## 

## 

## **CHAPTER 12**

## **12. FUTURE SCOPE**

1) It will have a variety of record-keeping choices (such as food, travel expenses, salary, etc.).

2) It will continue to give updates about our daily spending automatically.

3) Despite being in a haste to make money in today's hectic and expensive world, we eventually gave up. As we naively waste money on unnecessary items and titles. We came over with the intention of following our profit.

4) The user can specify their own expense categories here, such as those for food, clothing, rent, and bills, where they must input the money that has been spent.

## **CHAPTER 13**

## **13. APPENDIX**

### **13.1. SOURCE CODE**

from flask import Flask, render\_template, request, redirect, session,url\_for

import ibm\_db

from flask\_mail import Mail, Message

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32716;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=tkt02689;PWD=iJblyvngVsuVA5ae;",'','')

app = Flask(\_\_name\_\_)

# mail = Mail(app) # instantiate the mail class

# configuration of mail

app.config['MAIL\_SERVER']='smtp.gmail.com'

app.config['MAIL\_PORT'] = 465

app.config['MAIL\_USERNAME'] = 'sreenithiss.19it@kongu.edu'

app.config['MAIL\_PASSWORD'] = '01-Jul-02'

app.config['MAIL\_USE\_TLS'] = False

app.config['MAIL\_USE\_SSL'] = True

mail = Mail(app)

app.secret\_key = 'fasdgfdgdfg'

global gemail

@app.route('/login',methods=['POST','GET'])

def login():

if request.method == 'POST':

try:

global gemail

mail = request.form['mail']

pwd = request.form['psw']

sql = "SELECT \* from employee where email = '{}'".format(mail)

stmt = ibm\_db.exec\_immediate(conn, sql)

dict = ibm\_db.fetch\_assoc(stmt)

if (mail == dict['EMAIL'].strip() and pwd == dict['PASSWORD'].strip()):

print("if clause")

gemail = dict['EMAIL']

return redirect(url\_for("home"))

else:

return render\_template("signin.html",message = "Not a valid user")

except:

print ("ex")

if request.method == 'GET':

return render\_template("signin.html")

@app.route('/signup',methods=['POST','GET'])

def signup():

if request.method == 'POST':

try:

dict = {}

name = request.form['user\_name']

email = request.form['email']

age = request.form['age']

pw = request.form['psw']

sal = request.form['sal']

sql1 = "SELECT email from employee where email = '{}'".format(email)

stmt = ibm\_db.exec\_immediate(conn, sql1)

dict = ibm\_db.fetch\_assoc(stmt)

if(dict == False):

sql = "INSERT into employee values ('{}', '{}','{}', '{}','{}')".format(name, email, age, pw,sal)

stmt = ibm\_db.exec\_immediate(conn, sql)

sql = "INSERT into target values ('{}', 0,0,0,0)".format(email)

stmt = ibm\_db.exec\_immediate(conn, sql)

sql = "INSERT into expenses values ('{}', 0,0,0,0)".format(email)

stmt = ibm\_db.exec\_immediate(conn, sql)

return render\_template('signin.html')

else:

return redirect(url\_for('exists'))

except:

print ("sys.exc\_info()[0]")

if request.method == 'GET':

return render\_template('signup.html')

@app.route('/exists')

def exists():

return render\_template('signup.html',exists = "User already exists")

@app.route('/salarysubmit',methods=['POST'])

def salarysubmit():

if request.method == 'POST':

food = request.form['food']

rent = request.form['rent']

edu = request.form['edu']

ent = request.form['ent']

sql1 = "update target set food='{}',rent = '{}' , education = '{}' , entertainment = '{}' where email = '{}'".format(food,rent,edu,ent,gemail)

stmt = ibm\_db.exec\_immediate(conn, sql1)

sql = "SELECT \* from target where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn, sql)

dict = ibm\_db.fetch\_assoc(stmt)

sql1 = "SELECT \* from employee where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn, sql1)

dict1 = ibm\_db.fetch\_assoc(stmt)

return render\_template('salary.html',sal = dict1['SALARY'],food = dict['FOOD'],edu = dict['EDUCATION'],ent = dict['ENTERTAINMENT'],rent = dict['RENT'])

@app.route('/expensesubmit',methods=['POST'])

def expensesubmit():

if request.method == 'POST':

msg = ""

food = request.form['food']

rent = request.form['rent']

edu = request.form['edu']

ent = request.form['ent']

pre = "Select \* from target where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn, pre)

dict = ibm\_db.fetch\_assoc(stmt)

if(int(food) > dict['FOOD']):

print("if")

msg = Message(

'Hello',

sender ='sreenithiss.19it@kongu.edu',

recipients = [gemail]

)

msg.body = 'Exceeds the limit in food category'

mail.send(msg)

if(int(rent) > dict['RENT']):

print("if")

msg = Message(

'Hello',

sender ='sreenithiss.19it@kongu.edu',

recipients = [gemail]

)

msg.body = 'Exceeds the limit in rent category'

mail.send(msg)

if(int(edu) > dict['EDUCATION']):

print("if")

msg = Message(

'Hello',

sender ='sreenithiss.19it@kongu.edu',

recipients = [gemail]

)

msg.body = 'Exceeds the limit in education category'

mail.send(msg)

if(int(ent) > dict['ENTERTAINMENT']):

print("if")

msg = Message(

'Hello',

sender ='sreenithiss.19it@kongu.edu',

recipients = [gemail]

)

msg.body = 'Exceeds the limit in entertainment category'

mail.send(msg)

sql1 = "update expenses set food='{}',rent = '{}' , education = '{}' , entertainment = '{}' where email = '{}'".format(food,rent,edu,ent,gemail)

stmt = ibm\_db.exec\_immediate(conn, sql1)

sql = "SELECT \* from expenses where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn, sql)

dict = ibm\_db.fetch\_assoc(stmt)

sql1 = "SELECT \* from employee where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn, sql1)

dict1 = ibm\_db.fetch\_assoc(stmt)

return render\_template('expenses.html',sal = dict1['SALARY'],food = dict['FOOD'],edu = dict['EDUCATION'],ent = dict['ENTERTAINMENT'],rent = dict['RENT'])

@app.route('/home')

def home():

sql = "select \* from employee where email = '{}'".format(gemail)

stmt = ibm\_db.exec\_immediate(conn,sql)

dict = ibm\_db.fetch\_assoc(stmt)

return render\_template('home.html',name = dict['NAME'],email = dict['EMAIL'],age = dict['AGE'],sal = dict['SALARY'])

@app.route('/addsalary')

def salary():

global gemail

sql1 = "SELECT \* from employee where email = '{}'".format(gemail)

sql2 = "SELECT \* from target where email = '{}'".format(gemail)

print(gemail,"gmail")

stmt = ibm\_db.exec\_immediate(conn, sql1)

stmt1 = ibm\_db.exec\_immediate(conn, sql2)

dict1 = ibm\_db.fetch\_assoc(stmt)

dict = ibm\_db.fetch\_assoc(stmt1)

return render\_template('salary.html',sal = dict1['SALARY'],food = dict['FOOD'],edu = dict['EDUCATION'],ent = dict['ENTERTAINMENT'],rent = dict['RENT'])

@app.route('/addexpenses')

def expenses():

global gemail

sql1 = "SELECT \* from employee where email = '{}'".format(gemail)

sql2 = "SELECT \* from expenses where email = '{}'".format(gemail)

print(gemail,"gmail")

stmt = ibm\_db.exec\_immediate(conn, sql1)

stmt1 = ibm\_db.exec\_immediate(conn, sql2)

dict1 = ibm\_db.fetch\_assoc(stmt)

dict = ibm\_db.fetch\_assoc(stmt1)

print("dict1",dict1)

print("dict",dict)

return render\_template('expenses.html',sal = dict1['SALARY'],food = dict['FOOD'],edu = dict['EDUCATION'],ent = dict['ENTERTAINMENT'],rent = dict['RENT'])

@app.route('/visualization')

def vis():

sql2 = "SELECT \* from expenses where email = '{}'".format(gemail)

stmt1 = ibm\_db.exec\_immediate(conn, sql2)

dict = ibm\_db.fetch\_assoc(stmt1)

sql3 = "SELECT \* from target where email = '{}'".format(gemail)

stmt2 = ibm\_db.exec\_immediate(conn,sql3)

dict2 = ibm\_db.fetch\_assoc(stmt2)

return render\_template('vis.html',food = int(dict['FOOD']),edu = int(dict['EDUCATION']),ent = int(dict['ENTERTAINMENT']),rent = int(dict['RENT']),food1 = dict2['FOOD'],edu1 = dict2['EDUCATION'],ent1 = dict2['ENTERTAINMENT'],rent1 = dict2['RENT'])

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0', port=5000, debug=True)

### **13.2. GITHUB AND PROJECT DEMO LINK**

GitHub Link :  [https://github.com/IBM-EPBL/IBM-Project-29155-1660121684](%20https://github.com/IBM-EPBL/IBM-Project-29155-1660121684)