PROJECT REPORT
PERSONAL EXPENSE TRACKER

TEAM ID: PNT2022TMID38243

TEAM MEMBERS:

NAME	ROLL NUMBER
Abilash.M	412319205001
Abimanyu.A	412319205002
Arul Gnana Prakash.D	412319205007
Nithish Kumar.V	412319205018

INDEX

INTRODUCTION

1. Project Overview

Personal Expense Tracker is a web application that a

llows you to track the daily expense of the user and help them to keep track of their expenses daily, monthly, weekly and yearly basis. It will alsocreate a digitalrecords for the user's income and various expenses spent by the user is calculated. It also gets the input from the user as how much income earned and the date of the income earned and further creates a transaction entry and sums up all the total income. Further we can give voice commands and it is responsive. It will be very helpful for the users to manage their needs and they can spend in a better way by keeping track of the application daily basis.

2. Purpose

The main purpose of personal expense tracker application is used to keep track of expenses based on the user income and how much they spent and they can keep track of their expenses daily, monthly, weekly and yearly basis.

1. **LITERATURE SURVEY**
    1. Existing problem

The problem of current generation population is that they can't remember where all of the money they earned have gone and ultimately have to live while sustaining the little money they have left for their essential needs. In this time there is no such perfect solution which helps a person to track their daily expenditure easily and efficiently and notify them about the money shortage they have. For doing so have to maintain long ledgers or computer logs to maintain such data and the calculation is done manually by the user, which may generate error leading to losses. Not having a complete tracking.

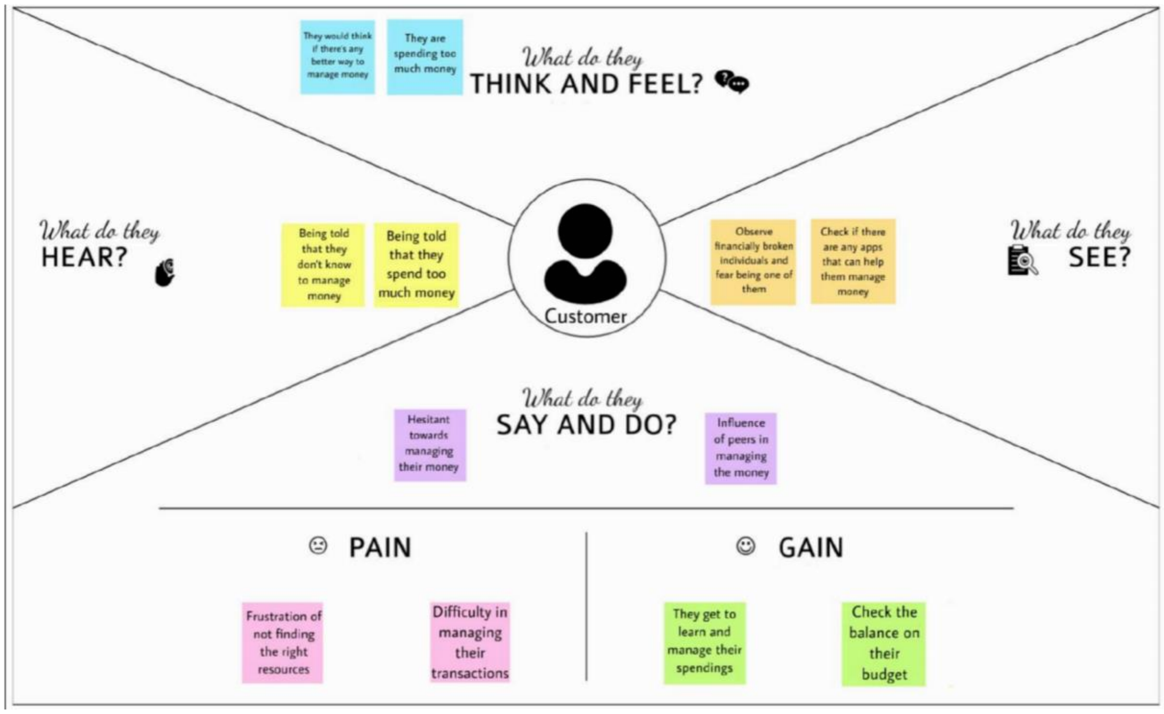
    2. References

2. Problem Statement Definition

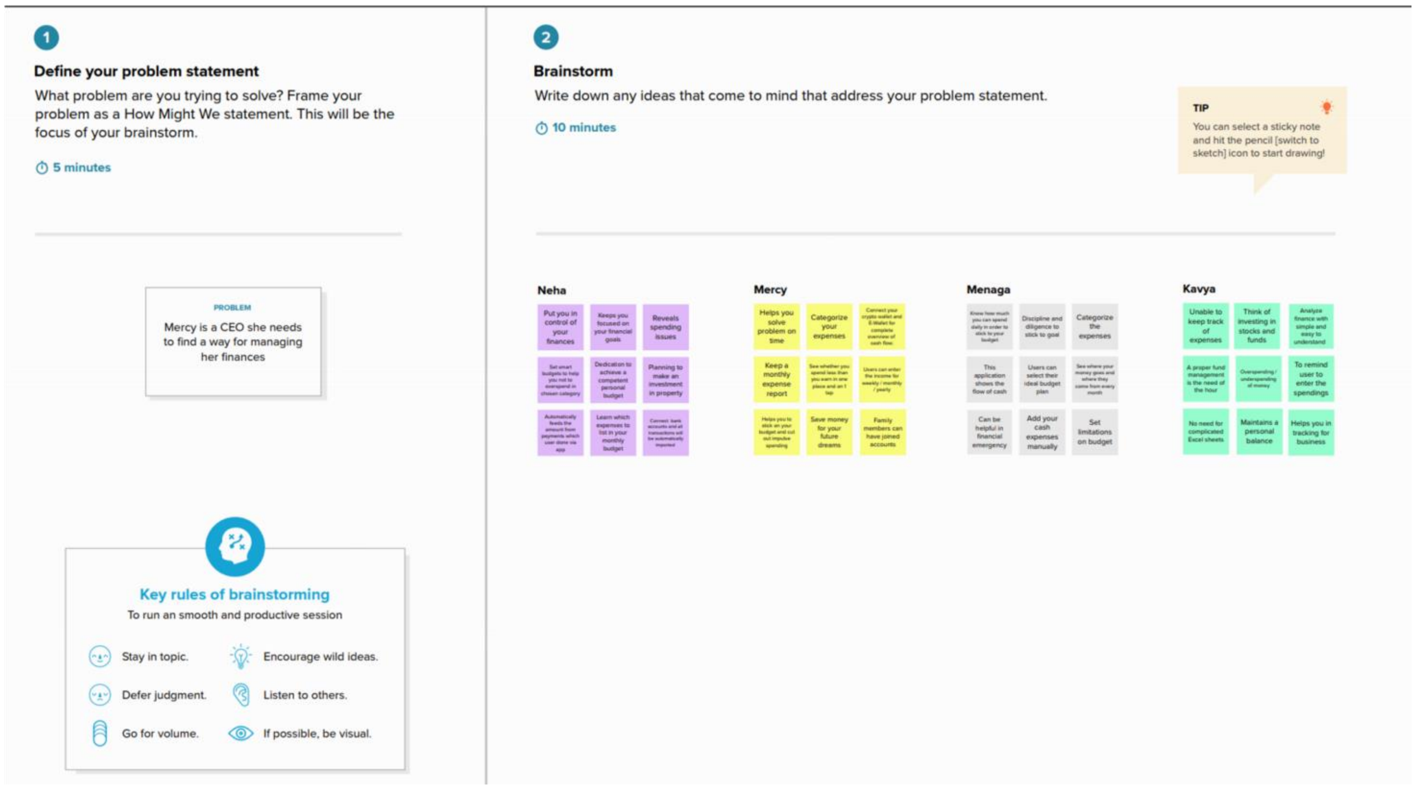
This Expense Tracker is a web application that facilitates the users to keep track and manage their personal as well as business expenses. This application helps the users to keep a digital diary. It will keep track of a user's income and expenses on a daily basis. The user will be able to add his/her expenditures instantly and can review them anywhere and anytime with the help of the internet. He/she can easily import transactions from his/her mobile wallets without risking his/her information and efficiently protecting his/her privacy. This expense tracker provides a complete digital solution to this problem. Excel sheets do very little to help in tracking Furthermore, they don't have the advanced functionality of preparing graphical visuals automatically. Not only it will save the time of the people but also it will assure error free calculations. The user just has to enter the income and expenditures and everything else will be performed by the system. Keywords: Expense Tracker, budget, planning, savings, graphical visualization of expenditure.

3. IDEATION & PROPOSED SOLUTION

1. Empathy Map Canvas


They would think if there's any better way to manage money
They are spending too much money
What do they THINK AND FEEL?
What do they HEAR?
Being told that they don't know to manage money
Being told that they spend too much money
Customer
Observe financially broken individuals and fear being one of them
Check if there are any apps that can help them manage money
What do they SEE?
What do they SAY AND DO?
Hesitant towards managing their money
Influence of peers in managing the money
PAIN
GAIN
Frustration of not finding the right resources
Difficulty in managing their transactions
They get to learn and manage their spendings
Check the balance on their budget

2. Ideation &Brainstorming



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

Mercy is a CEO she needs to find a way for managing her finances

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

2

Brainstorm

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

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Neha

- Put you in control of your finances
- Keeps you focused on your financial goals
- Reveals spending issues
- Set smart budgets to help you not to overspend in chosen category
- Dedication to achieve a competent personal budget
- Planning to make an investment in property
- Automatically feeds the amount from payments which user done via app
- Learn which expenses to list in your monthly budget
- Connect bank accounts and all transactions will be automatically imported

Mercy

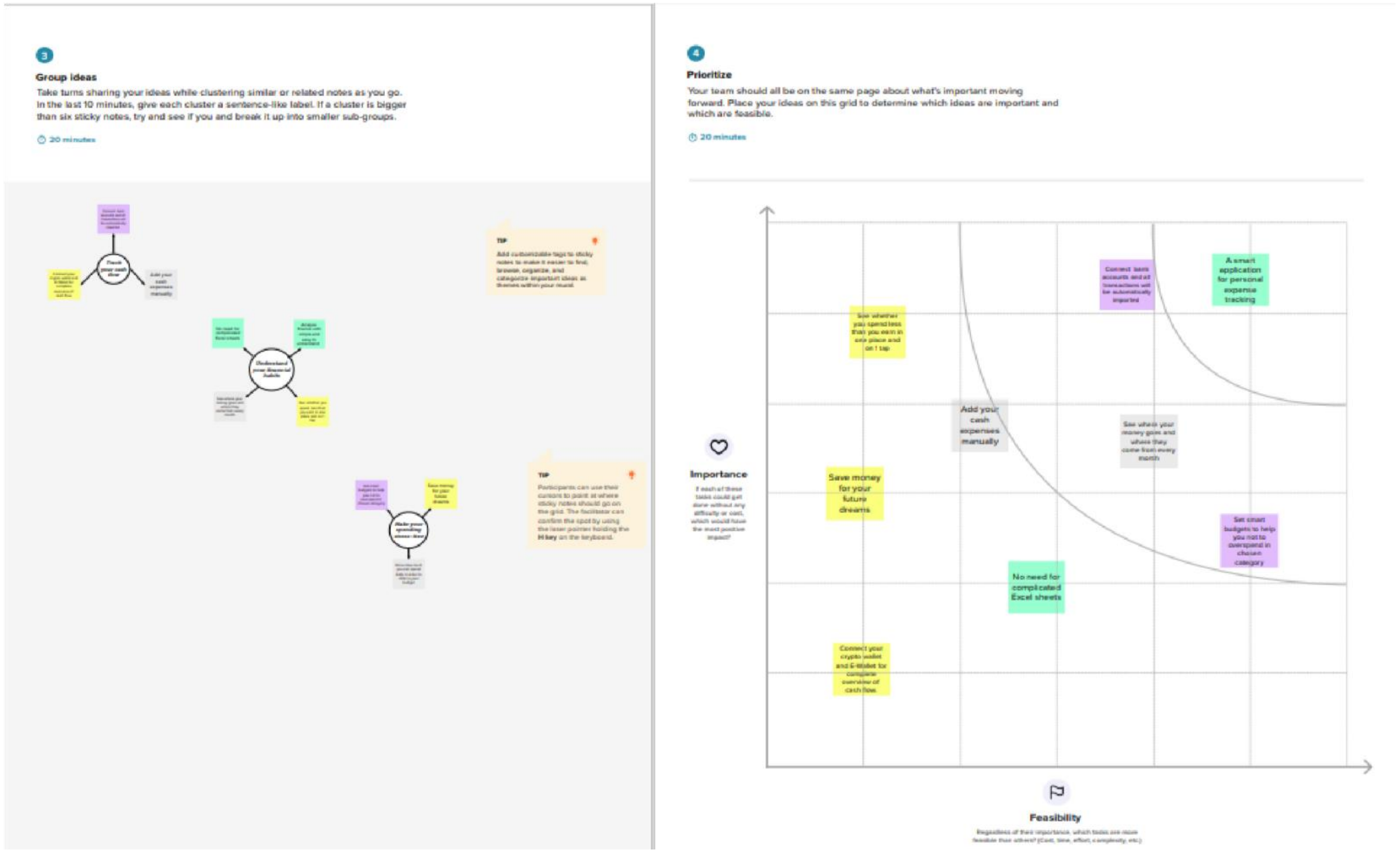
- Helps you solve problem on time
- Categorize your expenses
- Connect your crypto wallet and E-Wallet for complete overview of cash flow
- Keep a monthly expense report
- See whether you spend less than you earn in one place and on 1 tap
- Users can enter the income for weekly / monthly / yearly
- Helps you to stick on your budget and cut out impulse spending
- Save money for your future dreams
- Family members can have joined accounts

Menaga

- Know how much you can spend daily in order to stick to your budget
- Discipline and diligence to stick to goal
- Categorize the expenses
- This application shows the flow of cash
- Users can select their ideal budget plan
- See where your money goes and where they come from every month
- Can be helpful in financial emergency
- Add your cash expenses manually
- Set limitations on budget

Kavya

- Unable to keep track of expenses
- Think of investing in stocks and funds
- Analyze finance with simple and easy to understand
- A proper fund management is the need of the hour
- Overspending / underspending of money
- To remind user to enter the spendings
- No need for complicated Excel sheets
- Maintains a personal balance
- Helps you in tracking for business



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Track your cash flow
Add your cash expenses manually
Understand your financial habits
Make your spending stress-free
Save money for your future dreams

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

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.

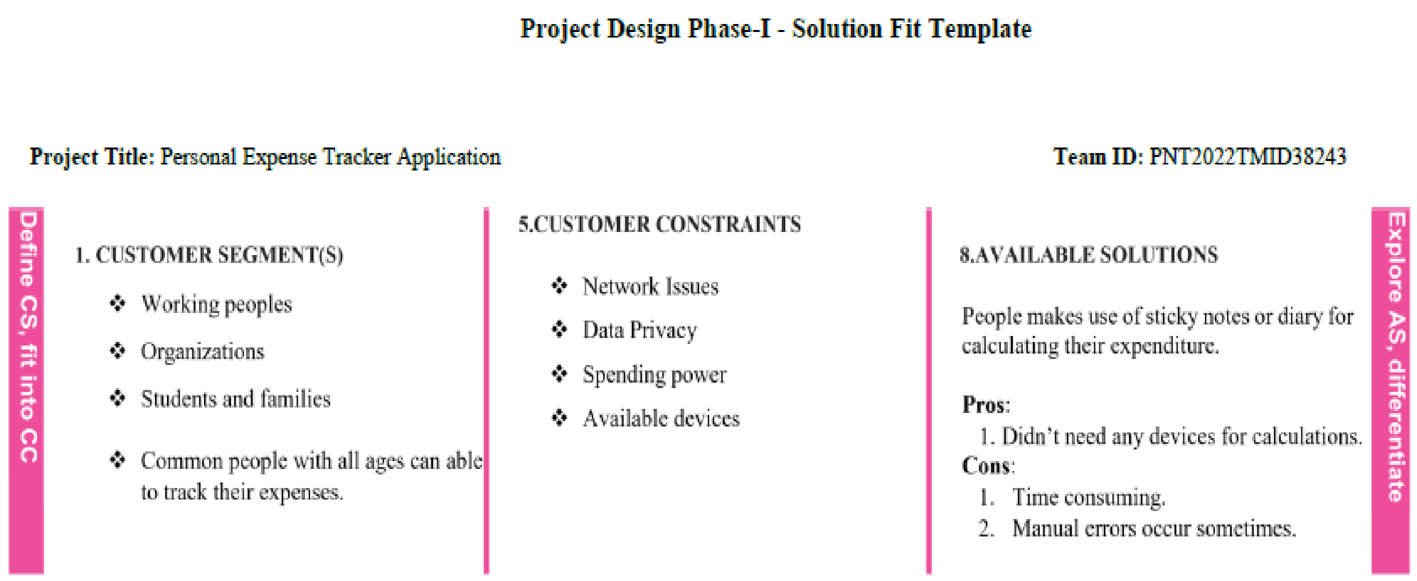
20 minutes

Connect bank accounts and all transactions will be automatically imported
A smart application for personal expense tracking
See whether you spend less than you earn in one place and on 1 tap
Add your cash expenses manually
See where your money goes and where they come from every month
Save money for your future dreams
Set smart budgets to help you not to overspend in chosen category
No need for complicated Excel sheets
Connect your crypto wallet and E-Wallet for complete overview of cash flow
Importance
If each of these tasks could 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.)

2. Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	At the moment, there is no such simple or free solution available that allows a person to effortlessly keep track of his or her daily expenses. In order to accomplish this, a person must maintain a journal in a diary or on a computer. Additionally, all computations must be performed by the user, which might occasionally result in mistakes that result in losses. Because there is no comprehensive tracking system, it is constantly burdensome to rely on the daily entry of expenditures and total estimates through the end of the month.
2.	Idea / Solution description	We're going to create a web application to make it simple to track spending and to provide useful insights into money management in order to manage expenses effortlessly.
3.	Novelty / Uniqueness	When the user's spending limit is exceeded, they will receive emails and text messages, and if they neglect to enter their expenses, a remainder will be set. Additionally, we will add automated ideas that will aid in budget planning, and we will group spending according to categories like entertainment, shopping, etc.
4.	Social Impact / Customer Satisfaction	It will aid consumers in keeping track of their spending and warn them when they go over their budget's allotment.
5.	Business Model (Revenue Model)	We can provide the application in a monthly subscription plan.
6.	Scalability of the Solution	This application can handle large number of users.

3. Problem Solutionfit


Define CS, fit into CC
Explore AS, differentiate

Project Design Phase-I - Solution Fit Template

Project Title: Personal Expense Tracker Application

Team ID: PNT2022TMID38243

1. CUSTOMER SEGMENT(S)

- ❖ Working peoples
- ❖ Organizations
- ❖ Students and families
- ❖ Common people with all ages can able to track their expenses.

5.CUSTOMER CONSTRAINTS

- ❖ Network Issues
- ❖ Data Privacy
- ❖ Spending power
- ❖ Available devices

8.AVAILABLE SOLUTIONS

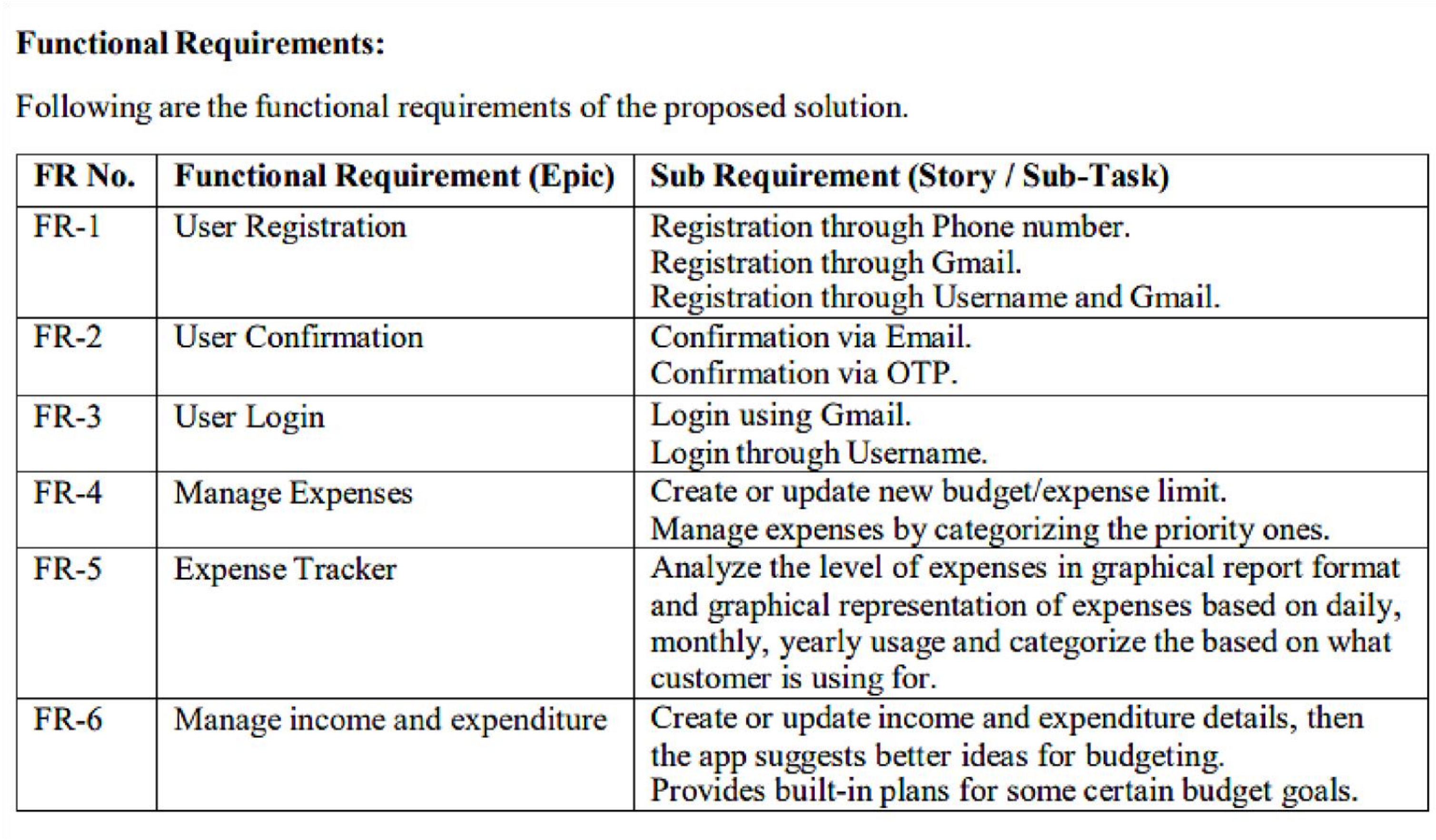
People makes use of sticky notes or diary for calculating their expenditure.

Pros:
1. Didn't need any devices for calculations.

Cons:
1. Time consuming.
2. Manual errors occur sometimes.

4. **REQUIREMENT ANALYSIS**

1. Functional requirement



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 Phone number. Registration through Gmail. Registration through Username and Gmail.
FR-2	User Confirmation	Confirmation via Email. Confirmation via OTP.
FR-3	User Login	Login using Gmail. Login through Username.
FR-4	Manage Expenses	Create or update new budget/expense limit. Manage expenses by categorizing the priority ones.
FR-5	Expense Tracker	Analyze the level of expenses in graphical report format and graphical representation of expenses based on daily, monthly, yearly usage and categorize the based on what customer is using for.
FR-6	Manage income and expenditure	Create or update income and expenditure details, then the app suggests better ideas for budgeting. Provides built-in plans for some certain budget goals.

2. Non-Functional requirements

Non-functional Requirements:

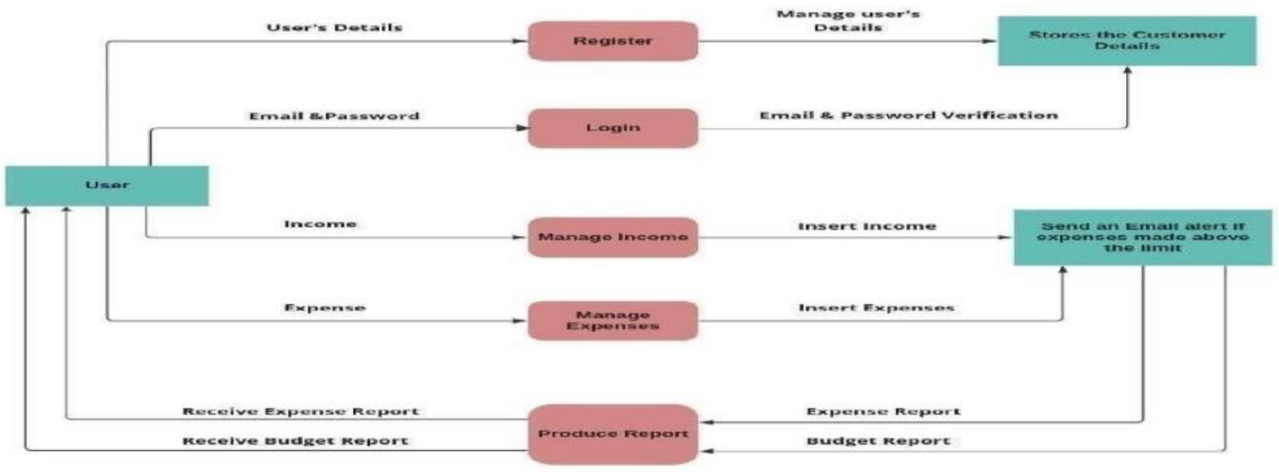
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This system will be used by anyone who needs to manage their expenses and to make better budgeting ideas.
NFR-2	**Security**	This system prevents customer's data securely and protects from malware attacks or unauthorized access.
NFR-3	**Reliability**	This system is highly reliable and it reduces the manual work load.
NFR-4	**Performance**	It tracks the expenses and generates reports quickly. It engages users efficiently with better budgeting ideas.
NFR-5	**Availability**	User can make his/her reports offline and this report is operational at any time.
NFR-6	**Scalability**	This system has better storage capacity and it manages large no of user's data.

5. PROJECT DESIGN

1. Data Flow Diagrams

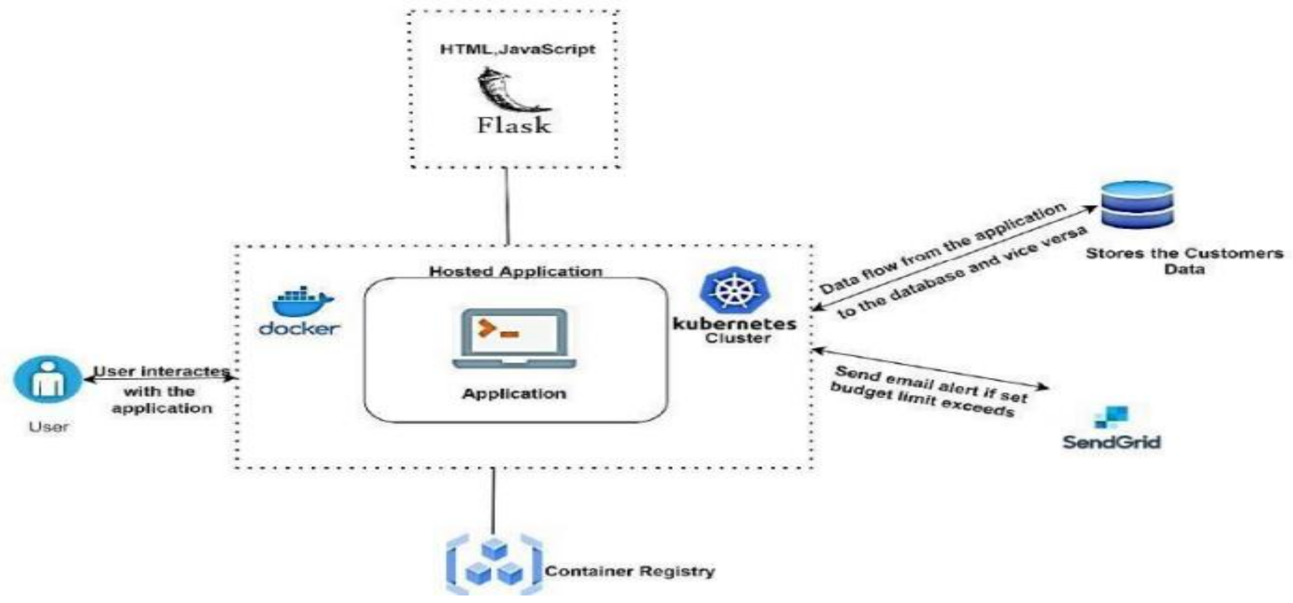
Data Flow Diagrams:



2. Technical Architecture

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table-1 & table-2

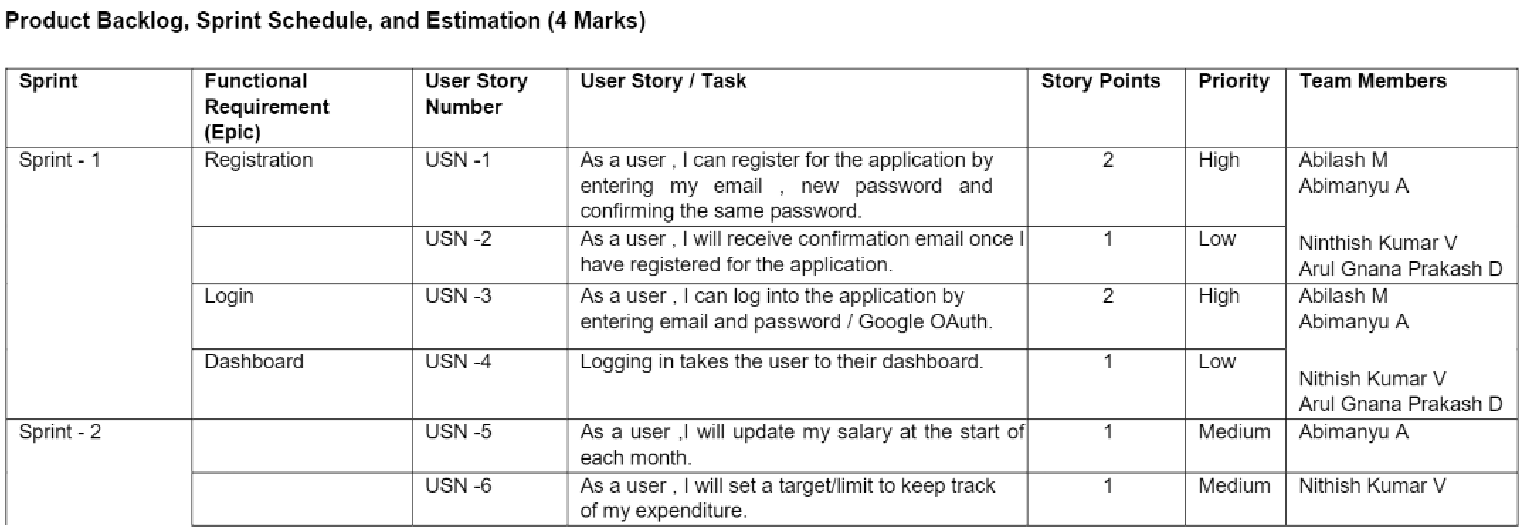
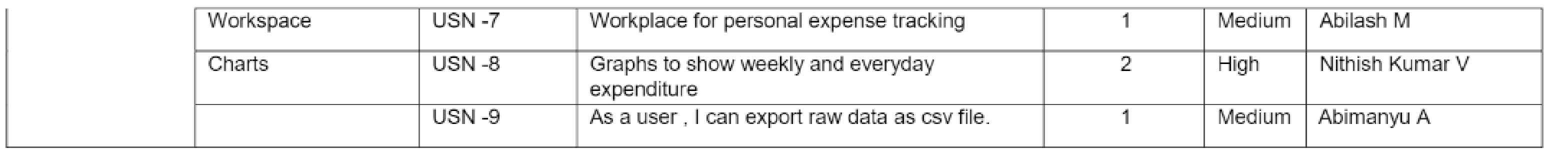

HTML,JavaScript
Flask
Hosted Application
docker
Application
kubernetes Cluster
User interactes with the application
User
Data flow from the application to the database and vice versa
Stores the Customers Data
Send email alert if set budget limit exceeds
SendGrid
Container Registry

3. User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High	
	Login	USN-2	As a user, I can log into the application by entering email & password	I can access theapplication	High	
	Dashboard	USN-3	As a user I can enter my income andexpenditure details.	I can view my dailyexpenses	High	
Customer Care Executive		USN-4	As a customer care executive, I can solvethe log in issues and other issues of the application.	I can provide support or solution at any time 24*7	Medium	
Administrator	Application	USN-5	As an administrator I can upgrade or update the application.	I can fix the bug which arises for the customersand users of the application	Medium	

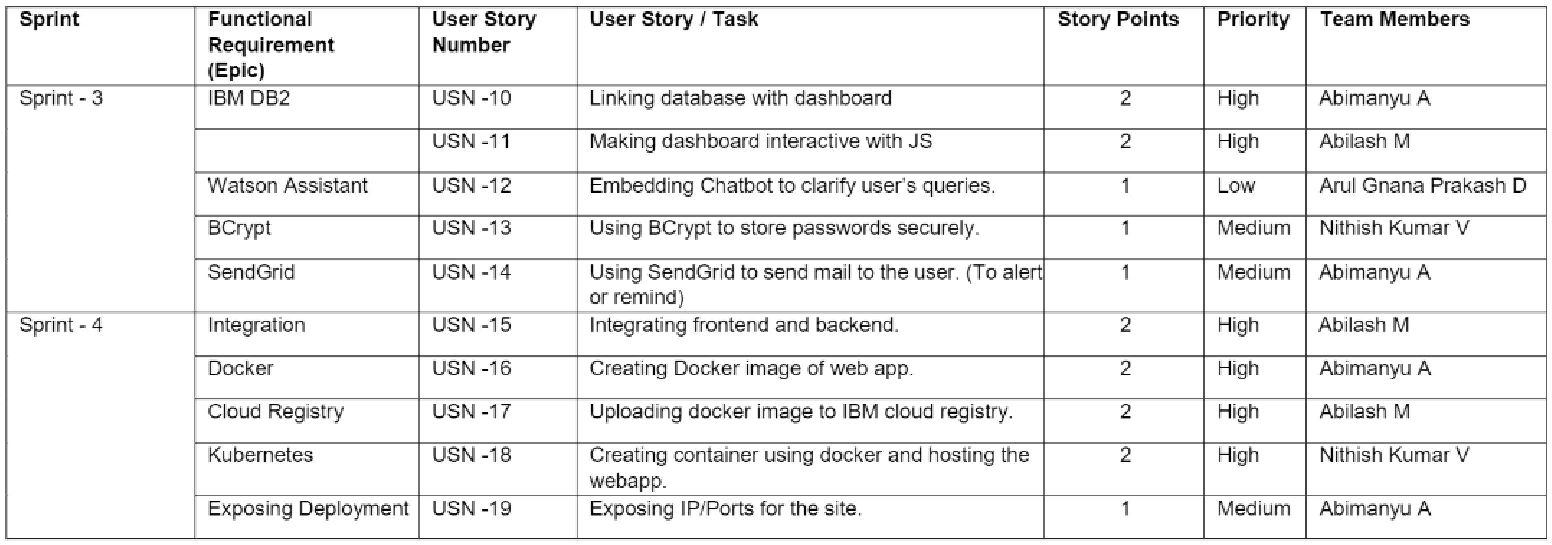
6. PROJECT PLANNING & SCHEDULING

1. Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

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 , new password and confirming the same password.	2	High	Abilash M Abimanyu A
		USN -2	As a user , I will receive confirmation email once I have registered for the application.	1	Low	Ninthish Kumar V Arul Gnana Prakash D
	Login	USN -3	As a user , I can log into the application by entering email and password / Google OAuth.	2	High	Abilash M Abimanyu A
	Dashboard	USN -4	Logging in takes the user to their dashboard.	1	Low	Nithish Kumar V Arul Gnana Prakash D
Sprint - 2		USN -5	As a user ,I will update my salary at the start of each month.	1	Medium	Abimanyu A
		USN -6	As a user , I will set a target/limit to keep track of my expenditure.	1	Medium	Nithish Kumar V
	Workspace	USN -7	Workplace for personal expense tracking	1	Medium	Abilash M
	Charts	USN -8	Graphs to show weekly and everyday expenditure	2	High	Nithish Kumar V
		USN -9	As a user , I can export raw data as csv file.	1	Medium	Abimanyu A



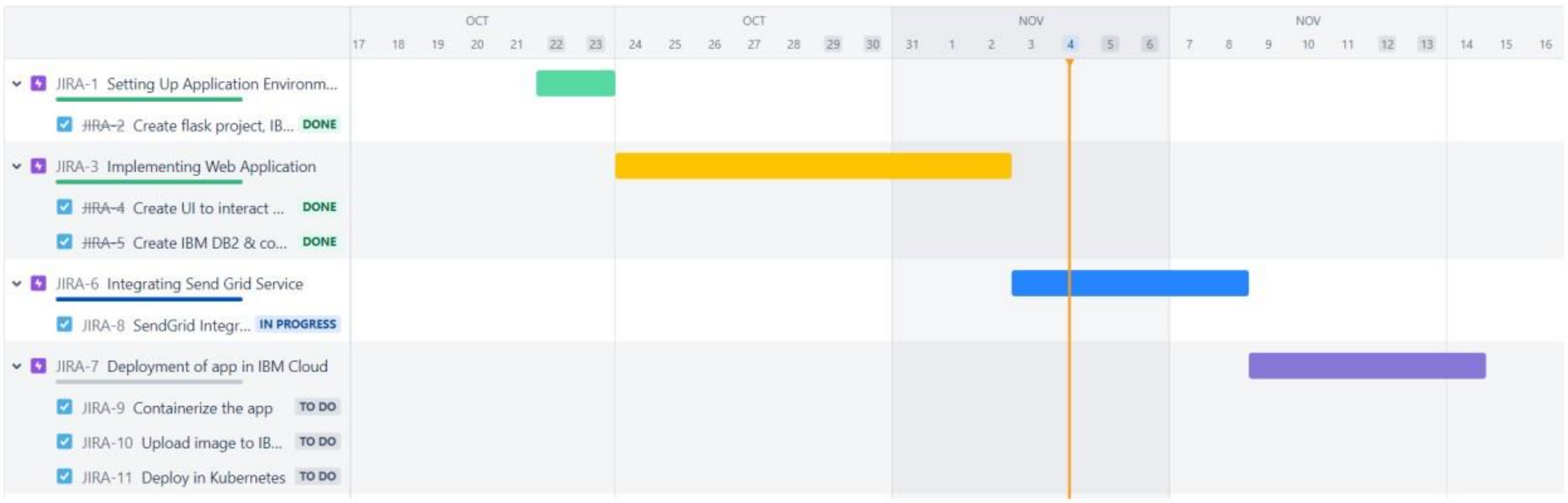
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 3	IBM DB2	USN -10	Linking database with dashboard	2	High	Abimanyu A
		USN -11	Making dashboard interactive with JS	2	High	Abilash M
	Watson Assistant	USN -12	Embedding Chatbot to clarify user's queries.	1	Low	Arul Gnana Prakash D
	BCrypt	USN -13	Using BCrypt to store passwords securely.	1	Medium	Nithish Kumar V
	SendGrid	USN -14	Using SendGrid to send mail to the user. (To alert or remind)	1	Medium	Abimanyu A
Sprint - 4	Integration	USN -15	Integrating frontend and backend.	2	High	Abilash M
	Docker	USN -16	Creating Docker image of web app.	2	High	Abimanyu A
	Cloud Registry	USN -17	Uploading docker image to IBM cloud registry.	2	High	Abilash M
	Kubernetes	USN -18	Creating container using docker and hosting the webapp.	2	High	Nithish Kumar V
	Exposing Deployment	USN -19	Exposing IP/Ports for the site.	1	Medium	Abimanyu A

e2. Sprint Delivery Schedul

Project Tracker, Velocity & Burndown Chart:

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

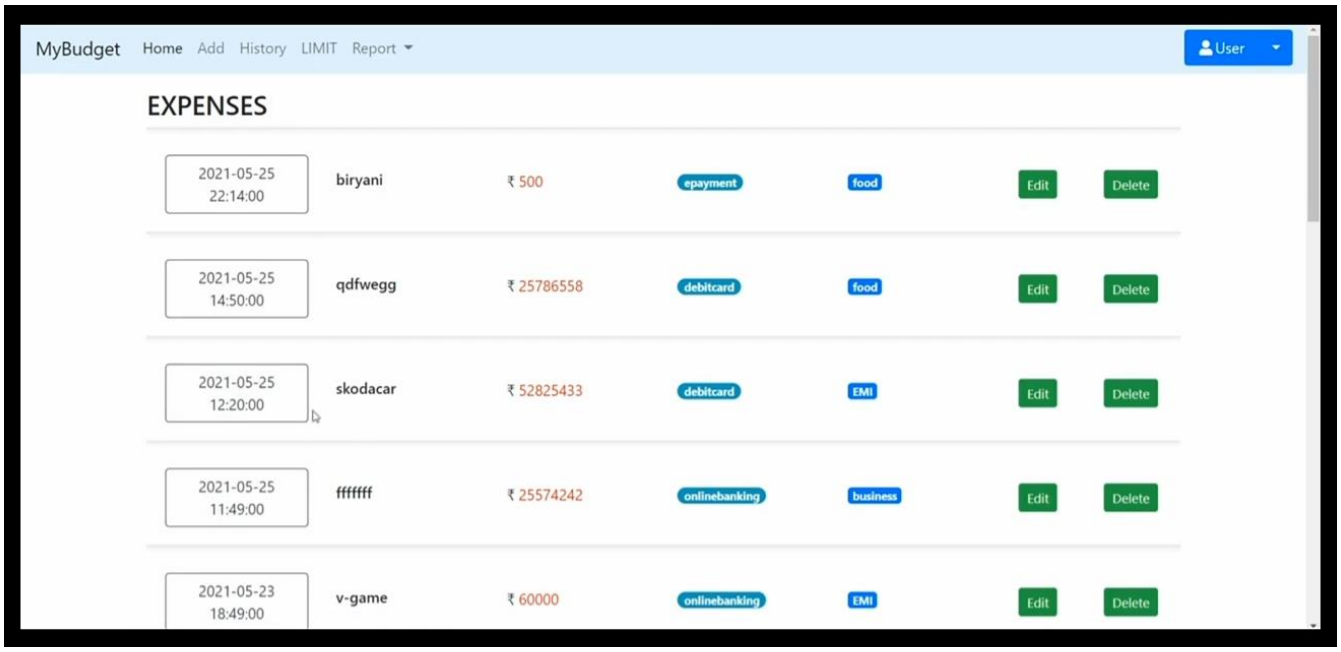
3. Reports from JIRA



JIRA-1 Setting Up Application Environm...
~~JIRA-2~~ Create flask project, IB... DONE
JIRA-3 Implementing Web Application
~~JIRA-4~~ Create UI to interact ... DONE
~~JIRA-5~~ Create IBM DB2 & co... DONE
JIRA-6 Integrating Send Grid Service
JIRA-8 SendGrid Integr... IN PROGRESS
JIRA-7 Deployment of app in IBM Cloud
JIRA-9 Containerize the app TO DO
JIRA-10 Upload image to IB... TO DO
JIRA-11 Deploy in Kubernetes TO DO

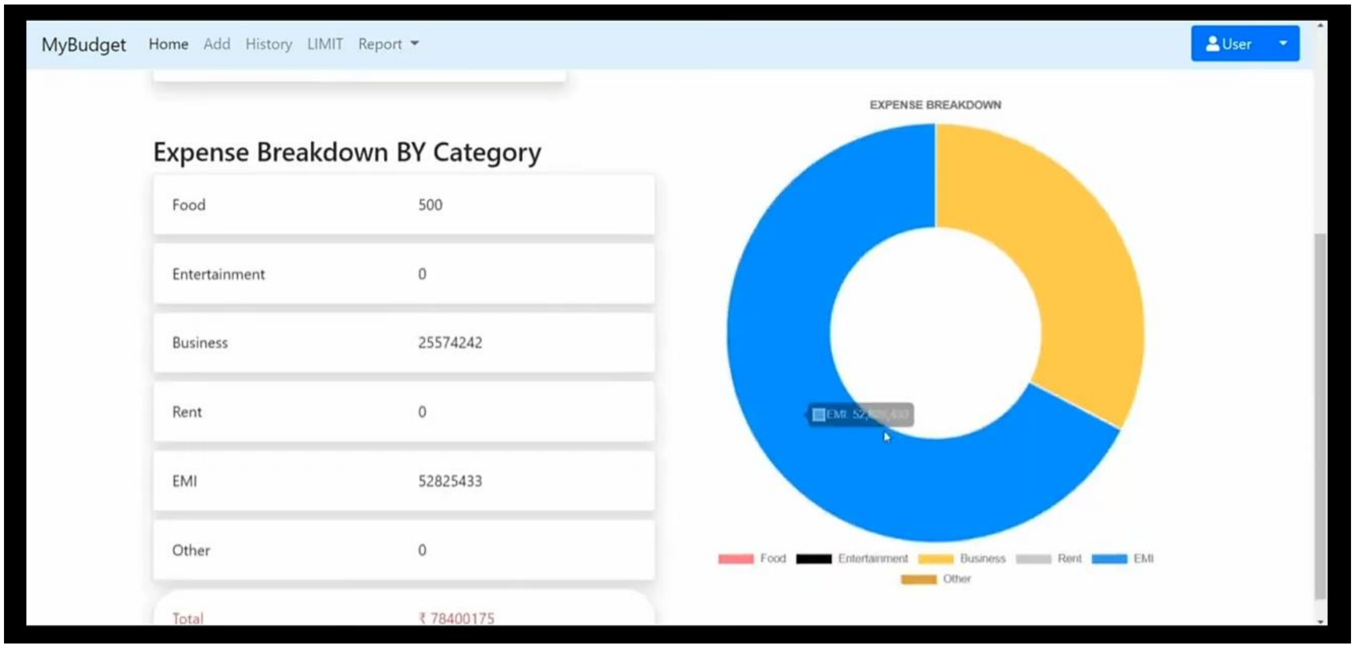
7. **CODING & SOLUTIONING**

1. Update Expense
2. Add Income



EXPENSES

2021-05-25 22:14:00	biryani	₹ 500	epayment	food	Edit	Delete
2021-05-25 14:50:00	qdfwegg	₹ 25786558	debitcard	food	Edit	Delete
2021-05-25 12:20:00	skodacar	₹ 52825433	debitcard	EMI	Edit	Delete
2021-05-25 11:49:00	fffffff	₹ 25574242	onlinebanking	business	Edit	Delete
2021-05-23 18:49:00	v-game	₹ 60000	onlinebanking	EMI	Edit	Delete

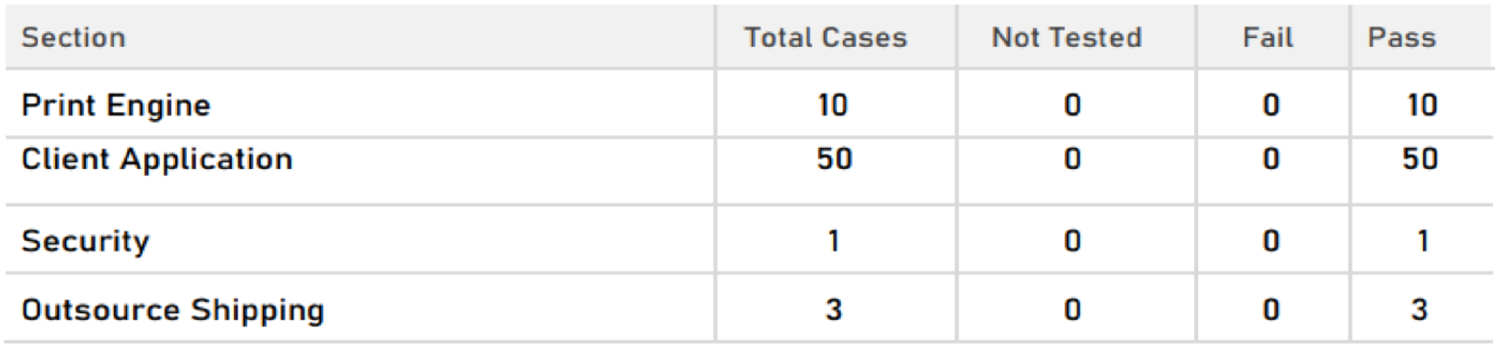


Expense Breakdown BY Category

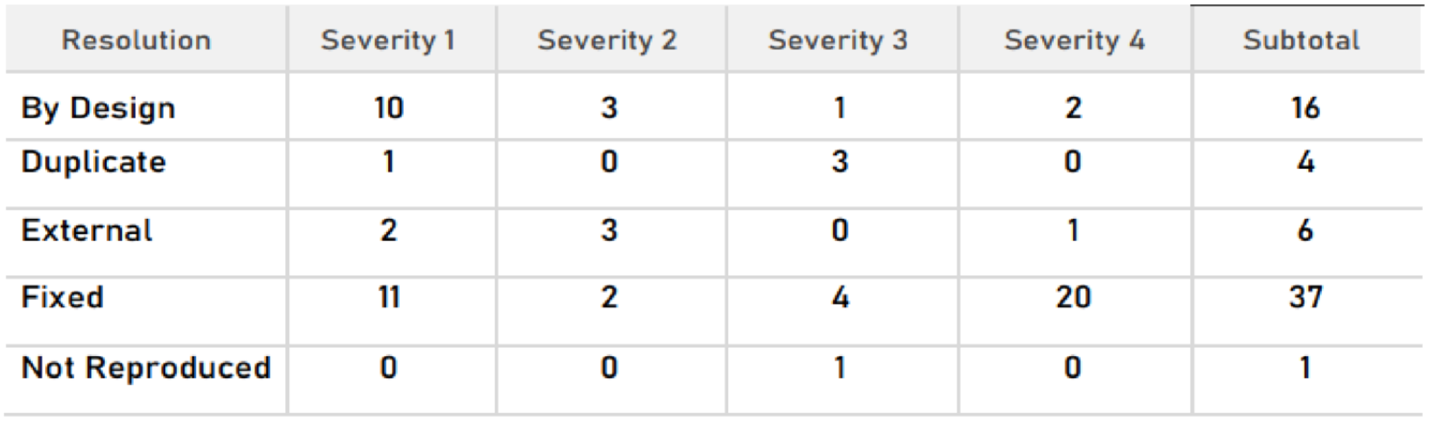
Food	500
Entertainment	0
Business	25574242
Rent	0
EMI	52825433
Other	0
Total	₹ 78400175

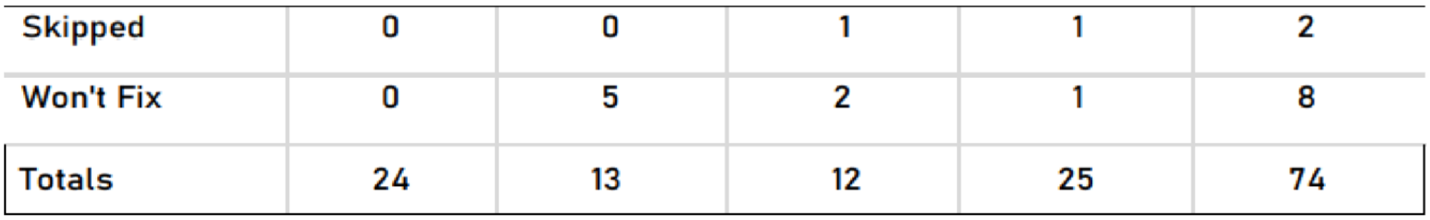
8. **TESTING**

1. Test Cases



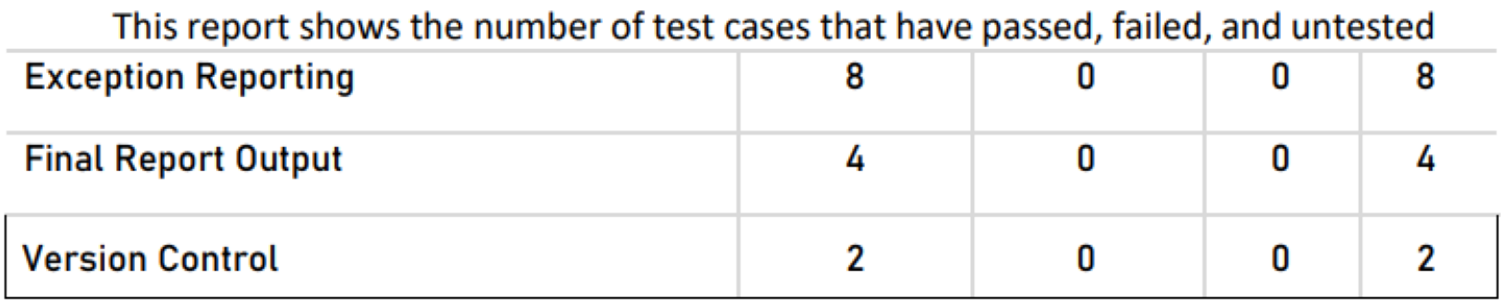
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Security	1	0	0	1
Outsource Shipping	3	0	0	3





Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	3	1	2	16
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	13	12	25	74

2. User Acceptance Testing



This report shows the number of test cases that have passed, failed, and untested

Exception Reporting	8	0	0	8
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. **RESULTS**

1. Performance Metrics

- Tracking income and expenses: Monitoring the income and tracking all expenditures (through bank accounts, mobile wallets, and credit & debit cards).
- Transaction Receipts: Capture and organize your payment receipts to keep track of your expenditure.
- Organizing Taxes: Import your documents to the expense tracking app, and it will streamline your income and expenses under the appropriate tax categories.
- Payments & Invoices: Accept and pay from credit cards, debit cards, net banking, mobile wallets, and bank transfers, and track the status of your invoices and bills in the mobile app itself. Also, the tracking app sendsremindersfor payments and automatically matches the payments with invoices.
- Reports: The expense tracking app generates and sends reports to give a detailed insight about profits, losses, budgets, income, balance sheets, etc.,

- E-commerce integration: Integrate your expense tracking app with your eCommerce store and track your sales through payments received via multiple payment methods. ● Vendors and Contractors: Manage and track all the payments to the vendors and contractors added to the mobile app.

- Access control: Increase your team productivity by providing access control to particular users through custom permissions.

- Track Projects: Determine project profitability by tracking labor costs, payroll, expenses, etc., of your ongoing project.

- Inventory tracking: An expense tracking app can do it all. Right from tracking products orthe cost of goods, sending alert notifications when the product is running out of stock or the product is not selling, to purchase orders.
- In-depth insights and analytics: Provides in-built tools to generate reports with easyto- understand visuals and graphics to gain insights about the performance of your business.

- Recurrent Expenses: Rely on your budgeting app to track, streamline, and automate all the recurrent expenses and remind you on a timely basis.

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

One of the major pros of tracking spending is always being aware of the state of one's personal finances. Tracking what you spend can help you stickto your budget, not just in a general way, but in each category such as housing, food, transportation and gifts. While a con is that manually trackingall cash that is spent can be irritating as well as time consuming, a pro is that doing this automatically can be quick and simple. Another pro is that many automatic spending tracking software programs are available for free. Having the program on a hand-held device can be a main pro since it can be checked before spending occurs in order to be sure of the available budget.

DISADVANTAGES:

A con with any system used to track spending is that one may start doing it then taper off until it's forgotten about all together. Yet, this is a risk for any new goal such as trying to lose weight or quit smoking. If a person first makes a budget plan, then places money in savings before spending any each new pay period or month, the tracking goal can help. In this way, tracking spending and making sure all receipts are accounted for only needs to be done once or twice a month. Even with constant tracking of one's spending habits, there is no guarantee that financial goals will be met. Although this can be considered to be a con of tracking spending, it could be changed into a pro if one makes up his or her mind to keep trying to properly manage all finances.

11. CONCLUSION

From this project, we are able to manage and keep tracking the daily expenses as well as income. While making this project, we gained a lot of experience of working as a team. We discovered various predicted and unpredicted problems and we enjoyed a lot solving them as a team. We adopted things like video tutorials, text tutorials, internet and learning materials to make our project complete.

12. FUTURE SCOPE

The project assists well to record the income and expenses in general.However, this project has some limitations:

● The application is unable to maintain the backup of data once it isuninstalled.

● This application does not provide higher decision capability.

To further enhance the capability of this application, we recommend the followingfeatures to be incorporated into the system:

● Multiple language interface.

● Provide backup and recovery of data.

● Provide better user interface for user.

● Mobile apps advantage.

13. APPENDIX

Source Code

ibm_web_app.py

```
import email
from flask import Flask ,render_template,request,redirect,url_for,session,flash

import re


import ibm_db
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=0c77d6f2-5da9-48a9-81f8-
86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31198;SECURITY=SSL;SSLServerCertificate=DigiCe
rtGlobalRootCA.crt;UID=jsl71809;PWD=bl03j1eFRYzVCv4R",'','')

app = Flask(__name__)
app.secret_key = 'qwdqwjdjecnwj'




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

@app.route("/home")
def homepage():
    return render_template("homepage.html")

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

@app.route("/register", methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        Password = request.form['password']
        sql = "SELECT * FROM user WHERE name=?"
        prep_stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(prep_stmt, 1, name)
        ibm_db.execute(prep_stmt)
        account = ibm_db.fetch_assoc(prep_stmt)
        print(account)
        if account:
            error = "Account already exists! Log in to continue !"
        else:
            insert_sql = "INSERT INTO user values(?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, name)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, Password)
            ibm_db.execute(prep_stmt)
    return render_template('login.html')




@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        print(email, password)
        sql = "SELECT * FROM user WHERE email=? AND password=?"
        stmt = ibm_db.prepare(conn, 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['email'] = account['EMAIL']
          return render_template('homepage.html')

        else:
          error = "Incorrect username / password"
    return render_template('login.html')

#ADDING----DATA


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


@app.route('/addexpense',methods=['GET', 'POST'])
def addexpense():
    date = request.form['date']
    expense_name = request.form['expense_name']
    amount = request.form['amount']
    paymode = request.form['paymode']
    category = request.form['category']
    insert_sql = 'INSERT INTO expenses (date,expense_name,amount,paymode,category) VALUES (?,?,?,?,?)'
    pstmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(pstmt, 1, date)
    ibm_db.bind_param(pstmt, 2, expense_name)
    ibm_db.bind_param(pstmt, 3, amount)
    ibm_db.bind_param(pstmt, 4, paymode)
    ibm_db.bind_param(pstmt, 5, category)
    ibm_db.execute(pstmt)



    return redirect("/display.html")

#DISPLAY---graph


@app.route("/display")
def display():
    print(session["username"],session['id'])
    sql = 'SELECT * FROM expenses WHERE userid = % s AND date ORDER BY `expenses`.`date` DESC',(str(session['id']))
    prep_stmt = ibm_db.prepare(conn, sql)
    expense = ibm_db.fetchall()


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

#delete---the--data

@app.route('/delete/<string:id>', methods = ['POST', 'GET' ])
def delete(id):
     sql = 'DELETE FROM expenses WHERE  id = {0}'.format(id)
     prep_stmt = ibm_db.prepare(conn, sql)
     ibm_db.execute()
     print('deleted successfully')
     return redirect("/display")

#UPDATE---DATA

@app.route('/edit/<id>', methods = ['POST', 'GET' ])
def edit(id):
    sql = 'SELECT * FROM expenses WHERE  id = %s', (id,)
    prep_stmt = ibm_db.prepare(conn, sql)
    row = ibm_db.fetchall()

    print(row[0])
    return render_template('edit.html', expenses = row[0])


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

      date = request.form['date']
      expense_name = request.form['expensename']
      amount = request.form['amount']
      paymode = request.form['paymode']
      category = request.form['category']
      sql = "UPDATE `expenses` SET `date` = % s , `expensename` = % s , `amount` = % s, `paymode` = % s, `category` = % s
WHERE `expenses`.`id` = % s ",(date, expense_name, amount, str(paymode), str(category),id)
      prep_stmt = ibm_db.prepare(conn, sql)
      ibm_db.execute()
      print('successfully updated')
      return redirect("/display")


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

@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
     if request.method == "POST":
        number= request.form['number']
        sql = 'INSERT INTO limits VALUES (NULL, % s, % s) ',(session['id'], number)
        prep_stmt = ibm_db.prepare(conn, sql)
        ibm_db.execute()

        return redirect('/limitn')

@app.route("/limitn")
def limitn():
    sql = 'SELECT limitss FROM `limits` ORDER BY `limits`.`id` DESC LIMIT 1'
    prep_stmt = ibm_db.prepare(conn, sql)
    x= ibm_db.fetchone()
    s = x[0]


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


#REPORT

@app.route("/today")
def today():
      sql = 'SELECT TIME(date)   , amount FROM expenses  WHERE userid = %s AND DATE(date) = DATE(NOW())
',(str(session['id']))
      prep_stmt = ibm_db.prepare(conn, sql)
      texpense = ibm_db.cursor.fetchall()
      print(texpense)

      sql = 'SELECT * FROM expenses WHERE userid = % s AND DATE(date) = DATE(NOW()) AND date ORDER BY
`expenses`.`date` DESC',(str(session['id']))
      prep_stmt = ibm_db.prepare(conn, sql)
      expense = ibm_db.fetchall()

      total=0
      t_food=0
      t_entertainment=0
      t_business=0
      t_rent=0
      t_EMI=0
      t_other=0
    for x in expense:
      total += x[4]
      if x[6] == "food":
        t_food += x[4]

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

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

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

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

    print(total)

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



    return render_template("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 )


@app.route("/month")
def month():
    sql = 'SELECT DATE(date), SUM(amount) FROM expenses WHERE userid= %s AND MONTH(DATE(date))= MONTH(now())
GROUP BY DATE(date) ORDER BY DATE(date) ',(str(session['id']))
    prep_stmt = ibm_db.prepare(conn, sql)
    texpense = ibm_db.fetchall()
    print(texpense)

    sql = 'SELECT * FROM expenses WHERE userid = % s AND MONTH(DATE(date))= MONTH(now()) AND date ORDER BY
`expenses`.`date` DESC',(str(session['id']))
    prep_stmt = ibm_db.prepare(conn, sql)
    expense = ibm_db.fetchall()

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


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

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

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

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

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

    print(total)

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



    return render_template("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 )

@app.route("/year")
def year():
    sql = 'SELECT MONTH(date), SUM(amount) FROM expenses WHERE userid= %s AND YEAR(DATE(date))= YEAR(now())
GROUP BY MONTH(date) ORDER BY MONTH(date) ',(str(session['id']))
    texpense = ibm_db.fetchall()
    print(texpense)


    sql = 'SELECT * FROM expenses WHERE userid = % s AND YEAR(DATE(date))= YEAR(now()) AND date ORDER BY
`expenses`.`date` DESC',(str(session['id']))
    expense = ibm_db.fetchall()
    total=0
    t_food=0
    t_entertainment=0
    t_business=0
    t_rent=0
    t_EMI=0
    t_other=0


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

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

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

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

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

    print(total)

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



    return render_template("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 )



#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)
```