Personal Expense Tracker

Team ID: PNT2022TMID34858

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYNMENT AND ENTERPRENEURSHIP

A PROJECT REPORT

submitted by

NITHINTON PRAKASH R (962819104063)
UDHAYAKUMAR K (962819104083)
AZHAGAPPAN T (962819104302)
THANEESH S (962819104305)

BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

University College Of Engineering
Nagercoil - 629004

INDEX

INTRODUCTION

1.1Project Overview

Expense tracking is an important part of creating a budget for our small business. Keeping a daily record of our expenses by tracking receipts, invoices and other outgoing expenses improves the financial health of our budget. It can manage both personal as well as business finances. The application has the provision to predict the income and expense for the manager using data mining. Tracking expenses throughout a project provides you the ability to view various expense categories and time periods. This can help you understand how much money you can spend for the rest of the project while staying within your budget.

1.2 Purpose

The purpose of an expense tracker is to identify where you are spending your money, and from there, you can identify the ways to save or invest more of your money and reduce your cost of living. At the end of each month, check the expenses you tracked to compare what you spent, what you planned to spend according to your budget. If you overspent, look for way to cut spending in a certain category, tracking your expenses might reveal that you budgeted too little for food or neglected to budget for one-time expenses such as holiday gifts, in which case you can incorporate these infrequent expenses and build a more realistic, comprehensive budget for the next month.

2. LITERATURE SURVEY

2.1 Existing problem

The web application “Expense Tracker” 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. The application has the provision to predict the income and expense for the manager using data mining. In this application, there are 3 logins such as admin, manager and staff. Admin has the privilege to add, edit, delete manager, add, edit, delete staff, and to get all custom reports. For Manager, the privileges are to add type of expense, verify expense, add type of income, verify income and generate reports. For staff, the privileges are to add and edit expense, income and calculations, and send for verifications.

2.2 References

[1]**.** Palestinian Ministry of Education and Higher Education. Palestinian Higher Education Statistics .

[2]**.** Accreditation and Quality Assurance Committee (AQAC) in Palestine. General Report of Information Technology and Engineering Higher Education in Palestine. Accreditation and Quality Assurance Commission (AQAC). Ramallah, Palestine: Palestinian Ministry of Education and Higher Education; 2007 Apr.

[3]**.** Engineering Association of Palestine. Current Engineering Statistics Book. Ramallah; 2005.

[4]**.** Prados J, Peterson G, Lattuca L. Quality Assurance of Engineering Education Through Accreditation: The Impact of Engineering Criteria 2000 and Its Global Influence. Journal of Engineering Education. 2005 Jan; 94(1):165–84.

[5]**.** Chen JW, Yen M. Engineering Accreditation: A Foundation for Continuing Quality Improvement. 2005 Mar 1–5; Tainan. Exploring Innovation in Education and Research,

2.3 Problem Statement Definition

1. Students who calculate their budgets in a notebook take more time to calculate so they get lazy and frustrated easily.

2. Students writes and calculates their daily expense and balance in a notebook so it makes them feel lazy.

3. Students write down all expenses and calculations in a notebook and it makes the page messy so it makes them confused.

4. Students take their expenses in their notebooks but if they lost that it makes them more frustrated.

5. Student scary an expense notebook everywhere he goes it makes them feel burdened.

6. Students write their expenses in a notebook and find it difficult in finding the major expenses category so it tends them to expend more.

7. Students may miss out the writing their expenses in the notebook so which makes them frustrated.

8. Students may miss out on paying the recurring bills so it makes them feel more stressed and less responsible.

9. Student scary their expense book with them so anyone can see that and makes them feel insecure.

10. Student comparing their budget with their imaginary budget is difficult so they send more.

11. Family guardians take monthly grocery lists and bills in a notebook so they feel very hard to maintain.

12. Family guardian stake the monthly grocery list on a piece of paper so they lost that they get frustrated.
13. Computer user track their expenses in an excel sheet by using it and get bored with the rows and columns.

14. The user manages all his expenses in software and faces difficulties with internet issues.

15. Mobile Users manage their expenses in a mobile app they spend more if it doesn’t have daily limit remainders so they get disappointment easily.

16. Small-scale shopkeepers maintain their profit and losses in their notebooks so it is difficult to maintain that and get confusion.

17. People with loans often forgot to pay the amount on time due to they get frustrated.

18. Buyers Losses the bill so they can’t return the product and feel sad.

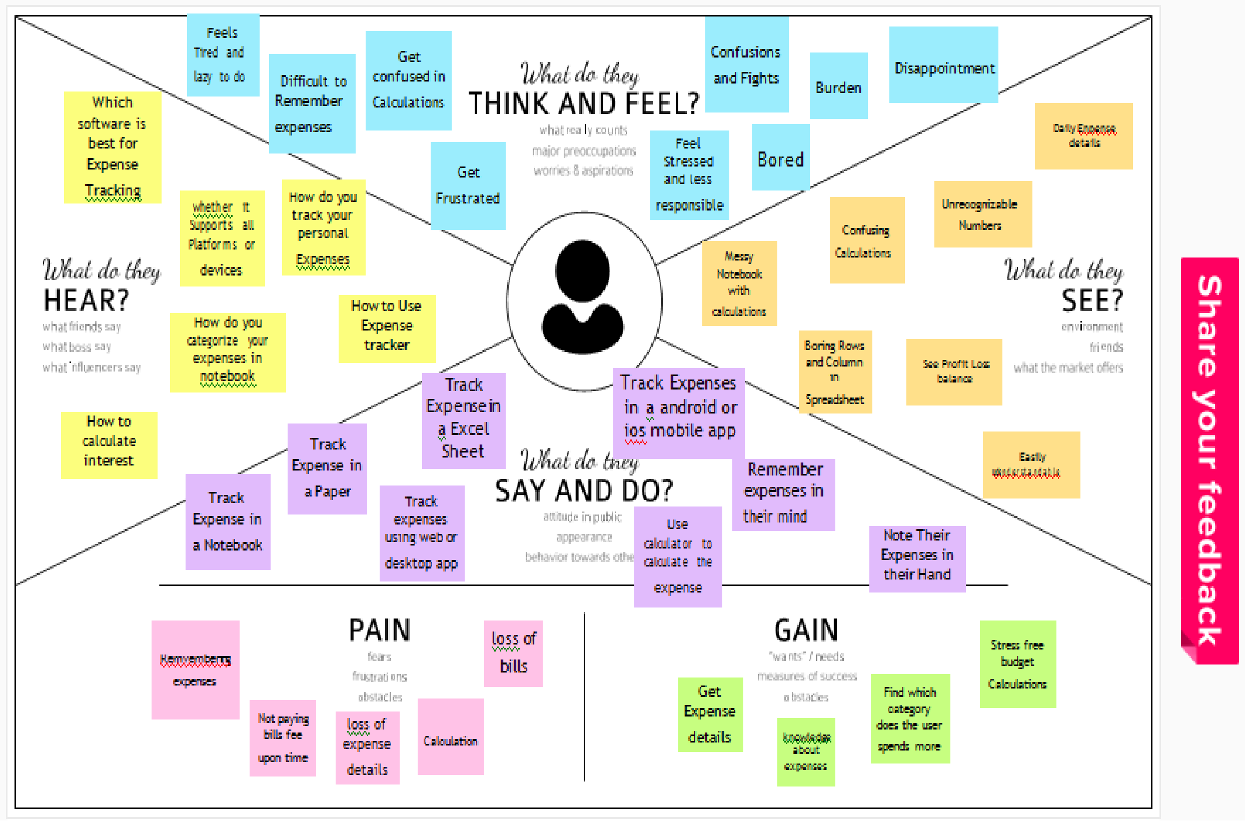
19. Students find it difficult in tracking semester fees and collecting fee receipts at the correct time so they face many problems.

20. Money Leander’s able to track the money is difficult to remember and track that so it can lead to confusion and fights.

3. IDEATION & PROPOSED SOLUTION

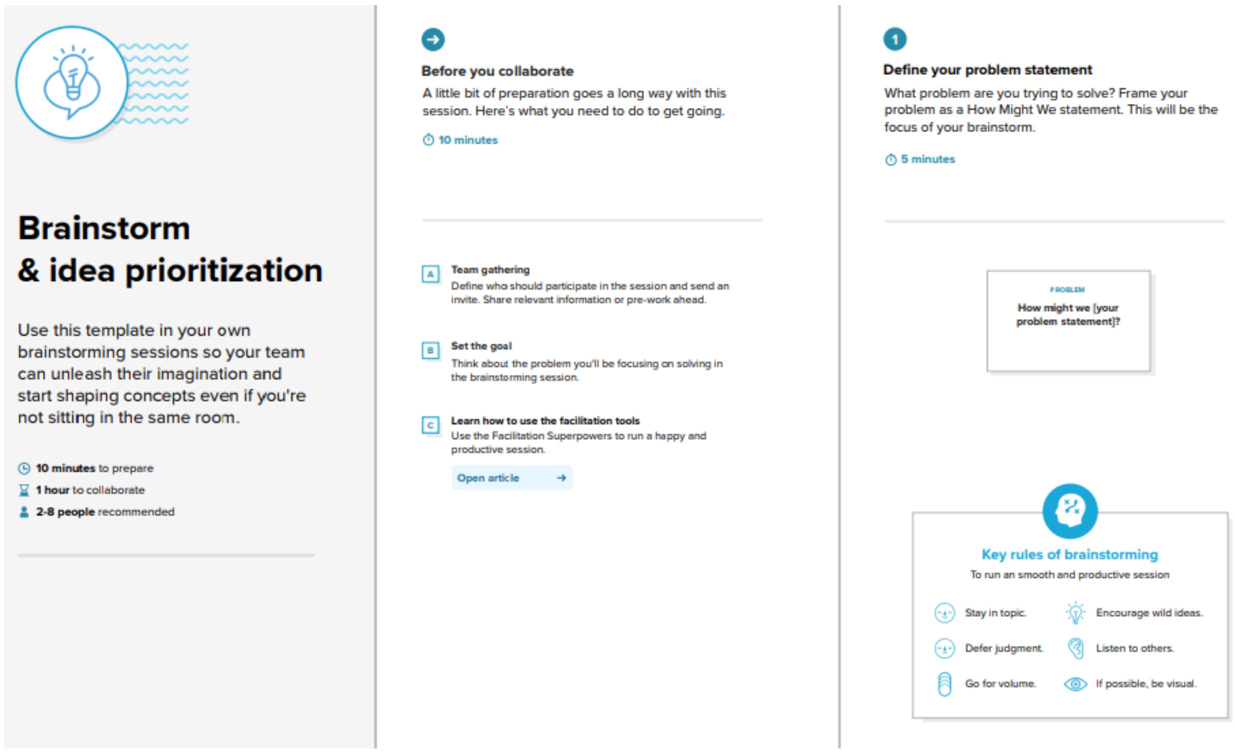
3.1 Empathy Map Canvas

Build empathy and keep your focus on the user by putting yourself in their shoes.


What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Feels Tired and lazy to do
Difficult to Remember expenses
Get confused in Calculations
Get Frustrated
Confusions and Fights
Burden
Disappointment
Feel Stressed and less responsible
Bored
What do they HEAR?
what friends say
what boss say
what influencers say
Which software is best for Expense Tracking
whether it Supports all Platforms or devices
How do you track your personal Expenses
How do you categorize your expenses in notebook
How to Use Expense tracker
How to calculate interest
What do they SEE?
environment
friends
what the market offers
Daily Expense details
Unrecognizable Numbers
Confusing Calculations
Messy Notebook with calculations
Boring Rows and Column in Spreadsheet
See Profit Loss balance
Easily understandable
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Track Expense in a Excel Sheet
Track Expense in a Paper
Track Expense in a Notebook
Track expenses using web or desktop app
Track Expenses in a android or ios mobile app
Remember expenses in their mind
Use calculator to calculate the expense
Note Their Expenses in their Hand
PAIN
fears
frustrations
obstacles
Remembering expenses
Not paying bills fee upon time
loss of expense details
Calculation
loss of bills
GAIN
"wants" / needs
measures of success
obstacles
Get Expense details
knowledge about expenses
Find which category does the user spends more
Stress free budget Calculations
Share your feedback

3.2 Ideation & Brainstorming

STEP 1



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

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

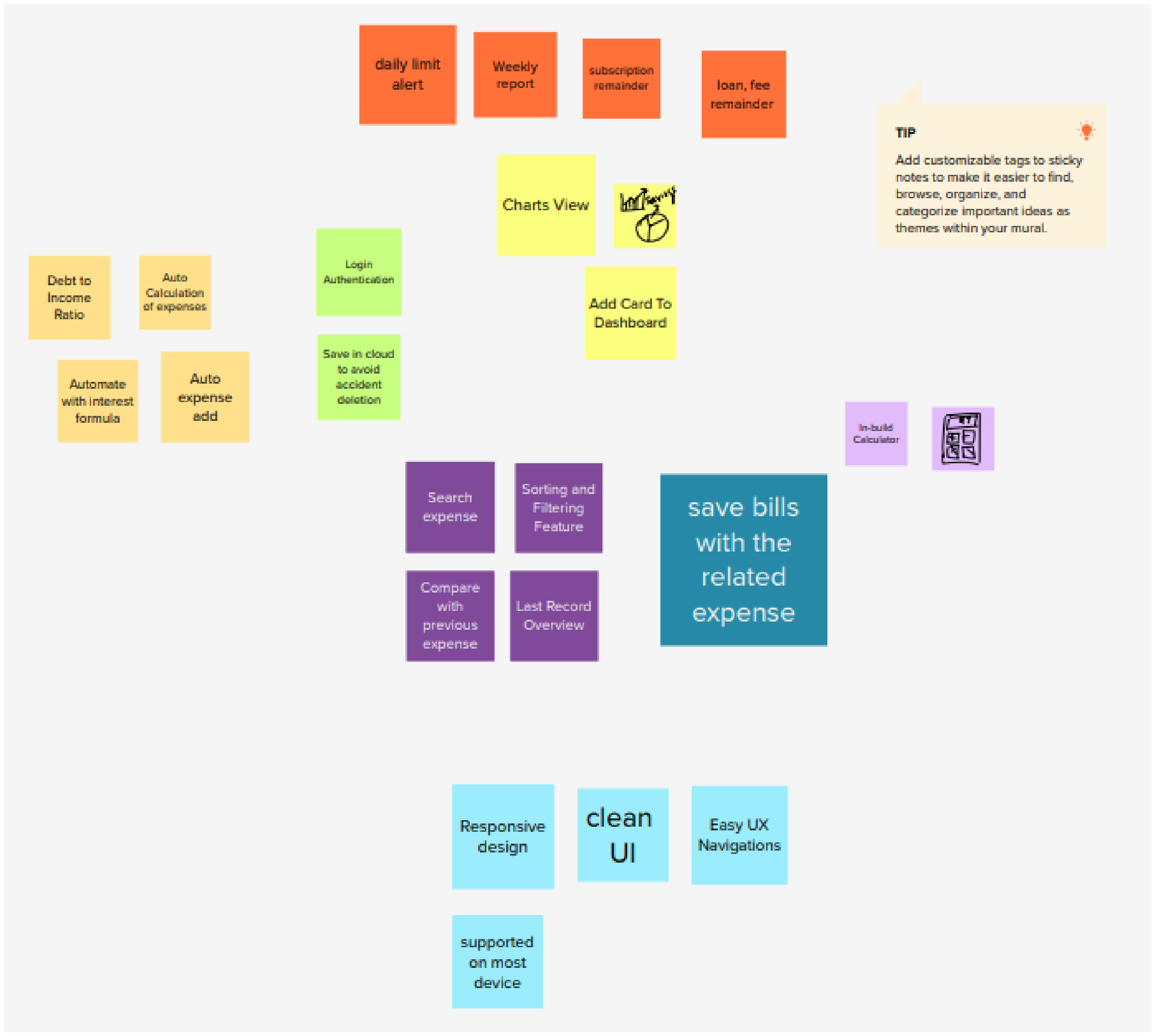
PROBLEM
How might we [your problem statement]?

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.

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


daily limit alert
Weekly report
subscription remainder
loan, fee remainder
TIP
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.
Charts View
Login Authentication
Debt to Income Ratio
Auto Calculation of expenses
Add Card To Dashboard
Save in cloud to avoid accident deletion
Automate with interest formula
Auto expense add
In-build Calculator
Search expense
Sorting and Filtering Feature
save bills with the related expense
Compare with previous expense
Last Record Overview
Responsive design
clean UI
Easy UX Navigations
supported on most device

STEP 2

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!

Person 1

Auto Calculation of expenses	In-build Calculator	Auto expense add
clean UI	Save in cloud to avoid accident deletion	Sorting and Filtering Feature
categorize Expenses	Easy UX Navigations	Hide Amounts

Person 2

Manual and auto adding expense	Login Authentication	Charts View
Create an imaginary budget	Compare with imaginary budget	Automate with interest formula
loan, fee remainder	Adjust Balance	Keep Your Warranties in One Place

Person 3

Search expense	save bills with the related expense	supported on most device
subscription remainder	budget limit alert	
Budgets & Goals	Create Template	Add Card To Dashboard

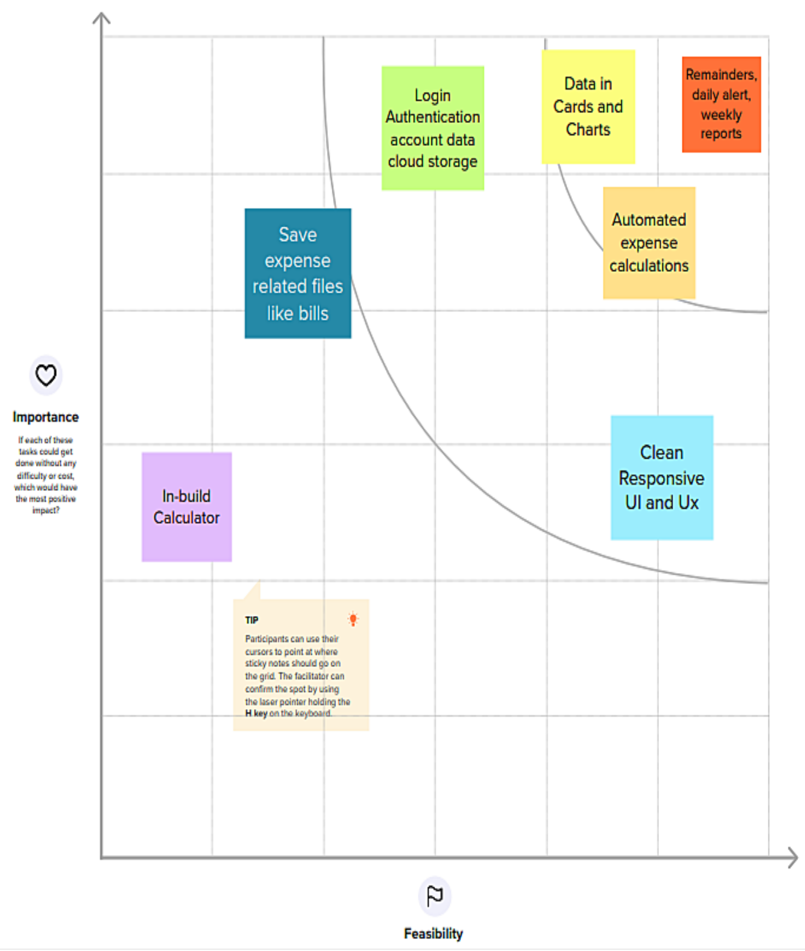
Person 4

Compare with previous expense	Responsive design	daily limit alert
Weekly report	Remember money leaning details	
Last Record Overview	Debt to Income Ratio	List of Accounts

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 each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Login Authentication account data cloud storage
Data in Cards and Charts
Remainders, daily alert, weekly reports
Save expense related files like bills
Automated expense calculations
Clean Responsive UI and Ux
In-build Calculator
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.
Feasibility

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

3.3Proposed Solution

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S No	Parameter	Description
1.	Problem Statement (Problem to besolved)	Lack of planning of income leads to money crisis in order to avoid this they log their daily expenditure in a notebook and do all boring and repetiative calculations daily.
2.	Idea / Solution description	Using web Cloud Application the expenditure islogged into the cloud with automatic calculation it gives an clear understanding where the money is going, prioritize your expenditure, plan your expenses
3.	Novelty / Uniqueness	Adding daily expense limit and remainder helps to spend less.
4.	Social Impact/ Customer Satisfaction	With automatic logging of expenditure and calculations helps user to be stress free in money management
5.	Business Model(Revenue Model)	With the help of personalized ads and premium plans or features revenue is made
6.	Scalability of the Solution	As this web application uses cloud storage with micro services the scalability is made easily withaffecting other feature

3.4 Problem Solution fit

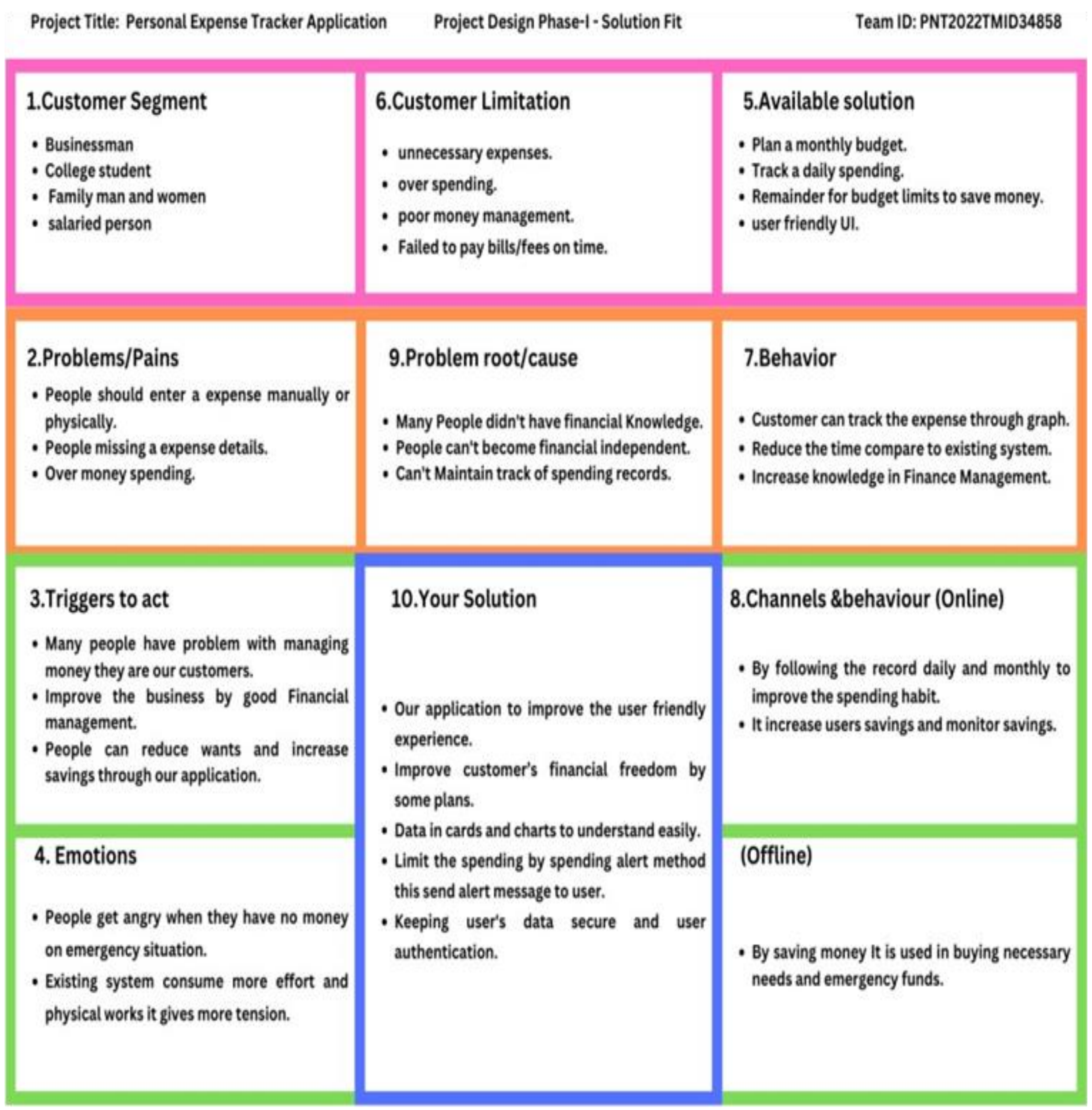
Problem – Solution Fit Template:

The Problem-Solution Fit simply means that you have found a problem with your customer and thatthe solution you have realized for it actually solves the customer’s problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why.

Purpose:

a. Solve complex problems in a way that fits the state of your customers.
b. Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
c. Sharpen your communication and marketing strategy with the right triggers and messaging.
d. Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
e. Understand the existing situation in order to improve it for your target group.

Template:



Project Title: Personal Expense Tracker Application Project Design Phase-I - Solution Fit Team ID: PNT2022TMID34858

1.Customer Segment

- Businessman
- College student
- Family man and women
- salaried person

6.Customer Limitation

- unnecessary expenses.
- over spending.
- poor money management.
- Failed to pay bills/fees on time.

5.Available solution

- Plan a monthly budget.
- Track a daily spending.
- Remainder for budget limits to save money.
- user friendly UI.

2.Problems/Pains

- People should enter a expense manually or physically.
- People missing a expense details.
- Over money spending.

9.Problem root/cause

- Many People didn't have financial Knowledge.
- People can't become financial independent.
- Can't Maintain track of spending records.

7.Behavior

- Customer can track the expense through graph.
- Reduce the time compare to existing system.
- Increase knowledge in Finance Management.

3.Triggers to act

- Many people have problem with managing money they are our customers.
- Improve the business by good Financial management.
- People can reduce wants and increase savings through our application.

4. Emotions

- People get angry when they have no money on emergency situation.
- Existing system consume more effort and physical works it gives more tension.

10.Your Solution

- Our application to improve the user friendly experience.
- Improve customer's financial freedom by some plans.
- Data in cards and charts to understand easily.
- Limit the spending by spending alert method this send alert message to user.
- Keeping user's data secure and user authentication.

8.Channels &behaviour (Online)

- By following the record daily and monthly to improve the spending habit.
- It increase users savings and monitor savings.

(Offline)

- By saving money It is used in buying necessary needs and emergency funds.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail, Registration through Expense tracker app
FR-2	User Confirmation	Confirmation via Gmail, Confirmation via OTP
FR-3	Login	Using mail id and password
FR-4	Financial Transaction	Using debit card, credit card and net banking
FR-5	Add transaction	This application allows adding transaction
FR-6	Delete transaction	This application allows to deleting transaction
FR-7	Total amount	It allows seeing total amount, amount spent in different categories and balance amount
FR-8	Pass code	This option to set a pass code for security
FR-9	Dashboard	Check their weekly, monthly and yearly expense details

4.2 Non-Functional requirements:

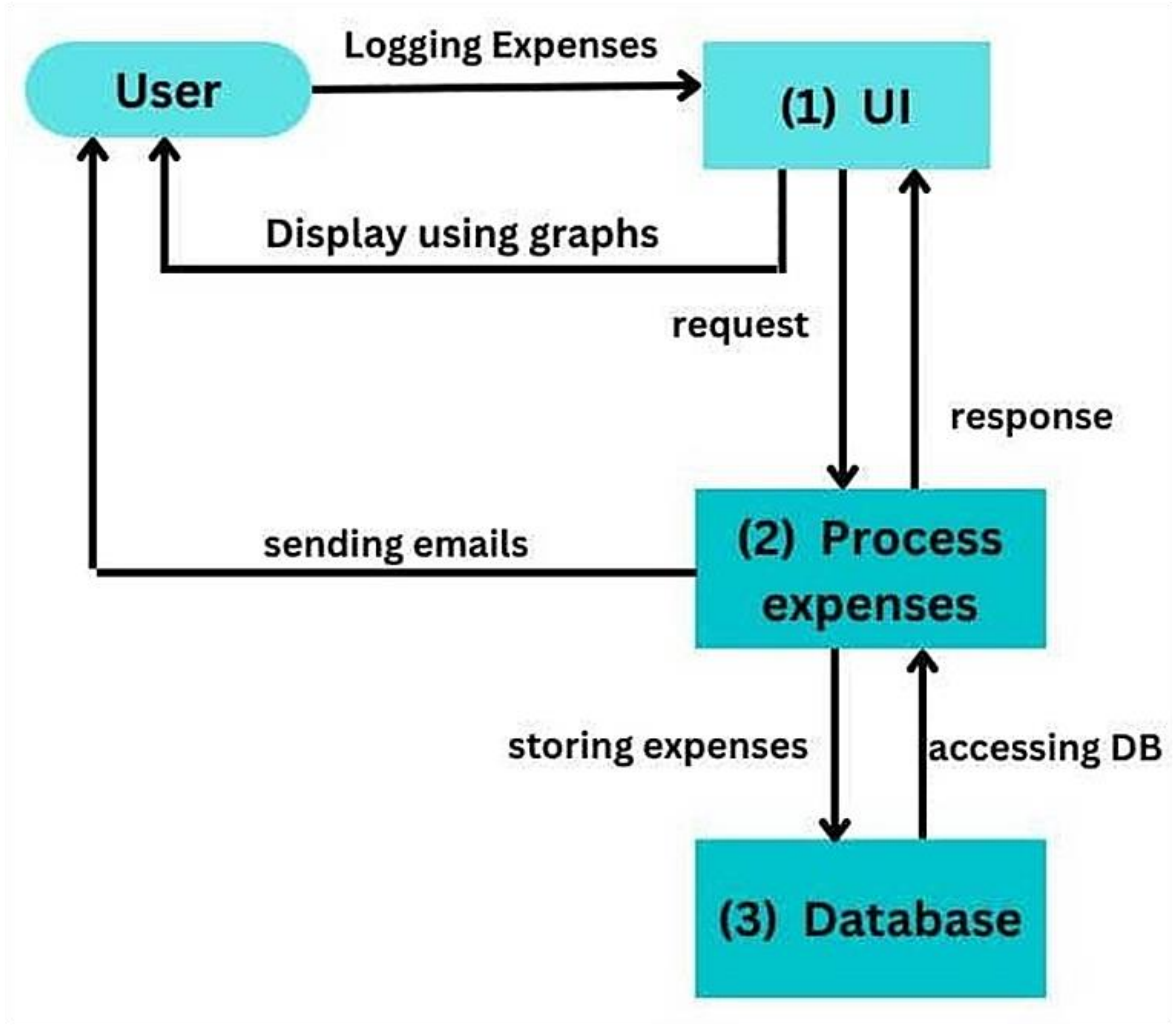
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	It helps to reduce your budget.
NFR-2	Reliability	It is fully secure, no need to share your account details. Improving financial security. Overcome wastage of money.
NFR-3	Performance	Make a better budget.
NFR-4	Availability	Budgeting tools, credit monitoring, receipt keeping.
NFR-5	Scalability	This application limits our purchase.

5. PROJECT DESIGN

5.1 Data Flow Diagrams:

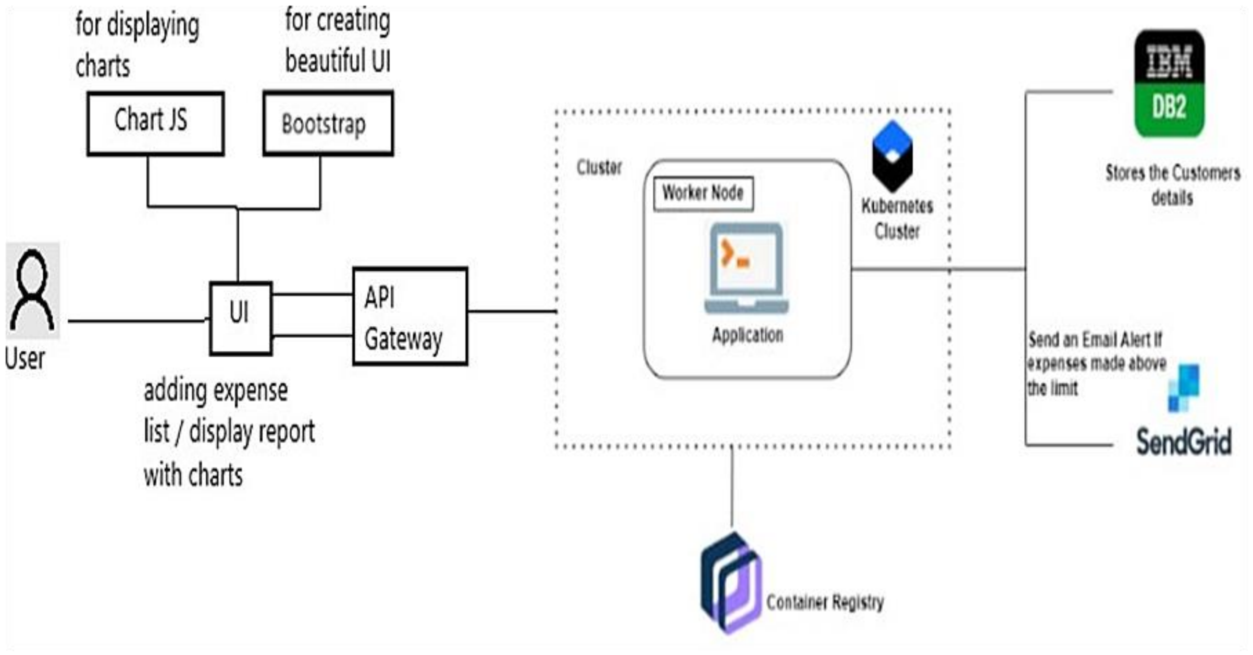
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical Architecture:

Solution architecture is a complex process – with many sub-processes – that bridgesthe gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of thesoftware to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, anddelivered.


for displaying charts
for creating beautiful UI
Chart JS
Bootstrap
User
UI
API Gateway
adding expense list / display report with charts
Cluster
Worker Node
Application
Kubernetes Cluster
IBM DB2
Stores the Customers details
Send an Email Alert If expenses made above the limit
SendGrid
Container Registry

5.3 User Stories

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

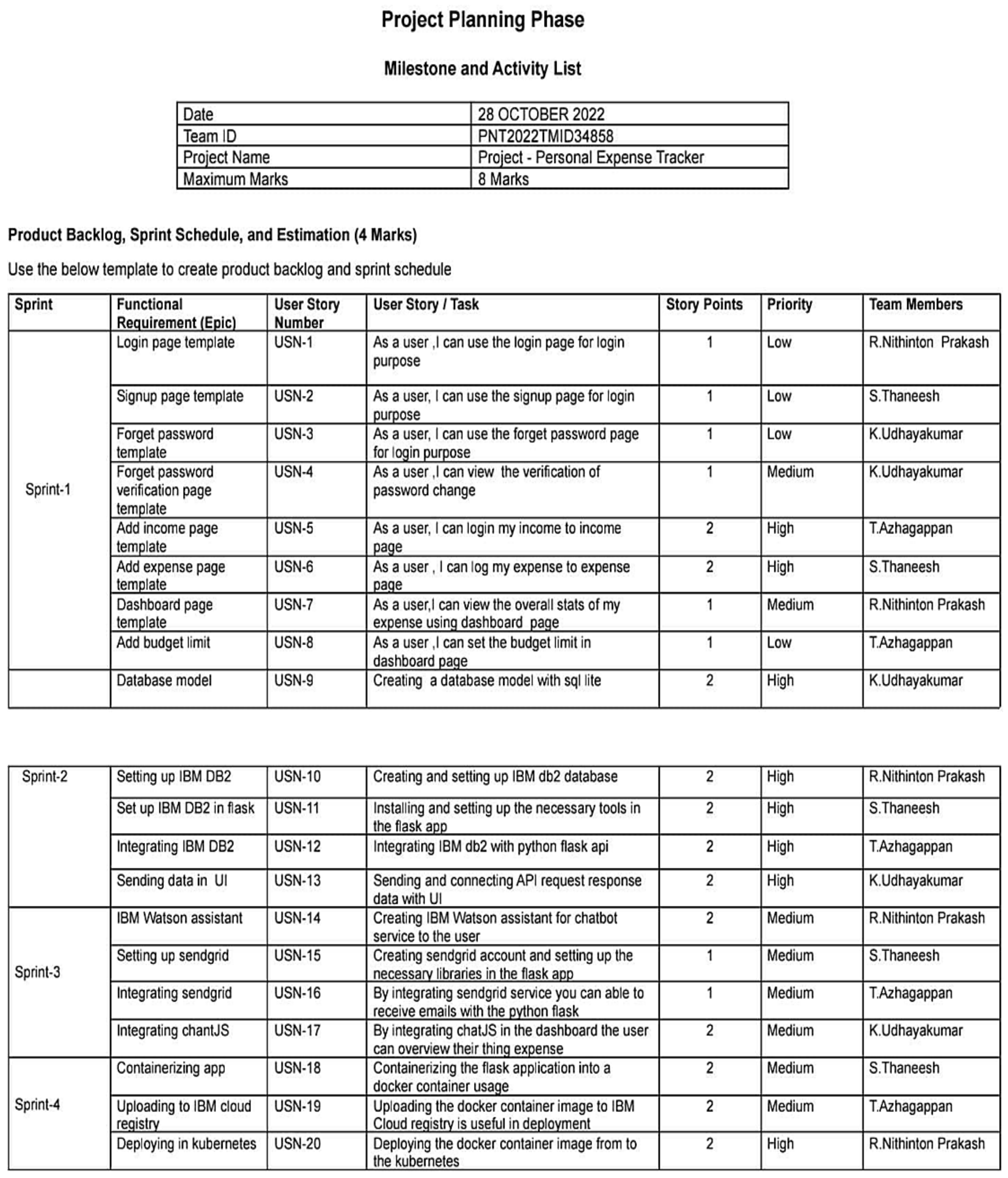
User Type	Functional Requirement(Epic)	User story Number	User story/Task	Acceptance criteria	Priority	Release
Customer (Family man or women, Business man & others)	Registration	USN-1	As a user, I can register for the application byentering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	Once a user login the application don't want to enter details every time from same account.	I can access my account without entering details	Medium	Sprint-1
		USN-4	After login user can view the dashboard	I can access dashboard	low	Sprint-1
	Dashboard	USN-5	As a user, I can view my Expense Graph	I can see my expense plans and expense	High	Sprint-1

			,Balance Trends	behavior		
		USN-6	As a user, I can access chat bot for get help.	I get help from IBM watson assistance to get help.	High	Sprint-1
	Alerting system	USN-7	As a user, I can set a limits for my expense, when it gets the limit triggers a alert message	I get alert message in email	High	Sprint-2
	Planning payments	USN-8	As a user, I can plan my payments for my monthly expenses	I can plan my monthly expenses	Medium	Sprint-1
		USN-9	As a user, I can Set a spending plan to manage my monthly salary by separating my expense in wants, needs, must category .	I can plan my monthly spendings in wants, needs, must category	High	Sprint-1
		USN-10	As a user, I can create goals new home, holiday trip, emergency fund etc	I can set goals and save money for that goals	low	Sprint-1
	Graph	USN-11	As a user, I	I can view	High	Sprint-1

			can view my expense structure on my dashboard	my expense structure to understand my expense		
		USN-12	As a user, I can view my wants, needs, must graph to see my financial behavior	I can view my monthly wants, needs, must category and previous records on graph.	Medium	Sprint-1
	Balance Trends	USN-13	As a user, I can enter my savings Details ,borrowings details and it is viewed in graph	I can see my actual balance and outlook ash-flow to manage expense ease to business and families.	High	Sprint-1
	Reports	USN-14	As a user, I can view my last month cash-flow table	I get a clear view about my financial trend by viewing the report	High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation



Project Planning Phase

Milestone and Activity List

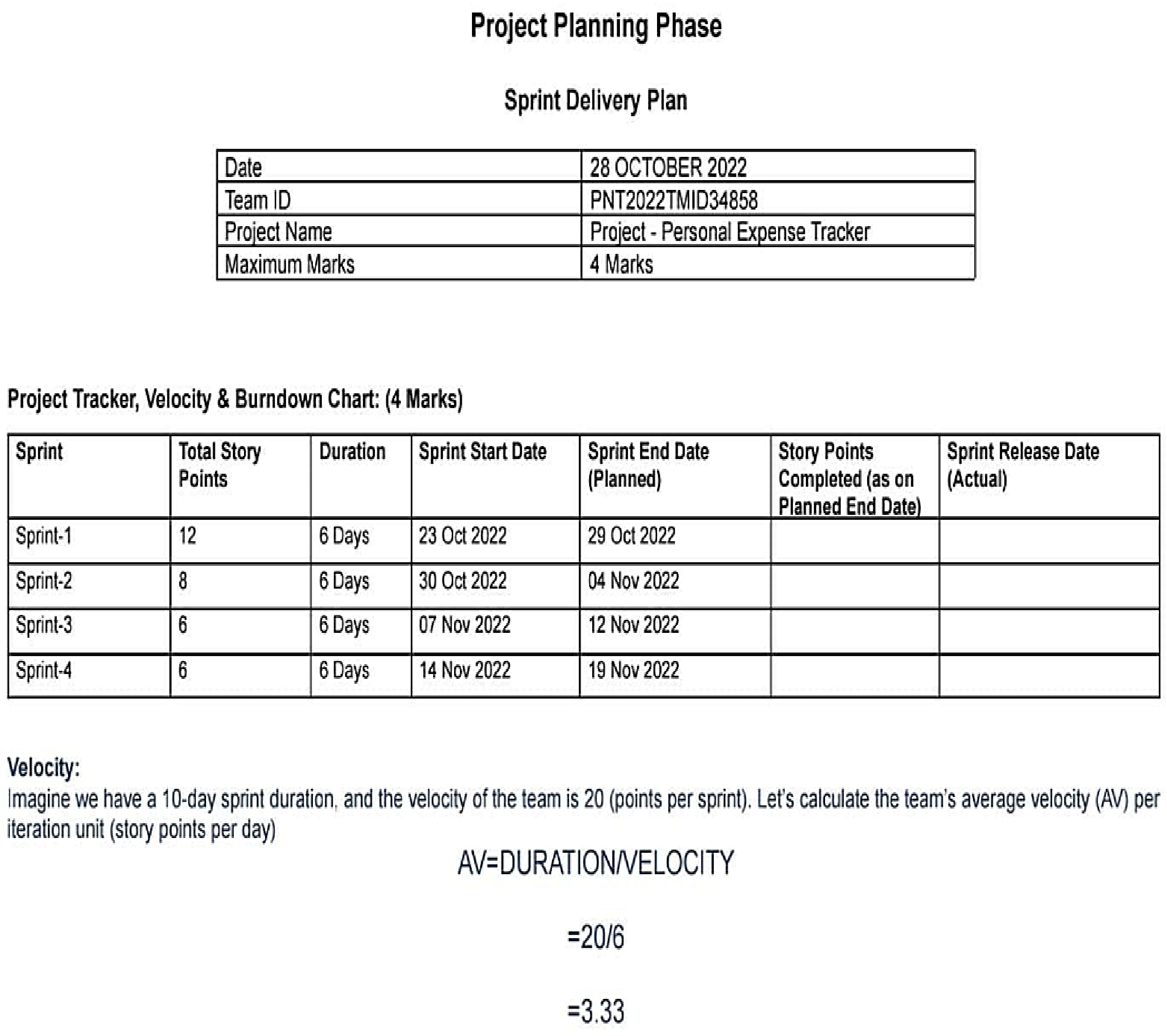
Date	28 OCTOBER 2022
Team ID	PNT2022TMID34858
Project Name	Project - Personal Expense Tracker
Maximum Marks	8 Marks

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

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login page template	USN-1	As a user ,I can use the login page for login purpose	1	Low	R.Nithinton Prakash
	Signup page template	USN-2	As a user, I can use the signup page for login purpose	1	Low	S.Thaneesh
	Forget password template	USN-3	As a user, I can use the forget password page for login purpose	1	Low	K.Udhayakumar
	Forget password verification page template	USN-4	As a user ,I can view the verification of password change	1	Medium	K.Udhayakumar
	Add income page template	USN-5	As a user, I can login my income to income page	2	High	T.Azhagappan
	Add expense page template	USN-6	As a user , I can log my expense to expense page	2	High	S.Thaneesh
	Dashboard page template	USN-7	As a user,I can view the overall stats of my expense using dashboard page	1	Medium	R.Nithinton Prakash
	Add budget limit	USN-8	As a user ,I can set the budget limit in dashboard page	1	Low	T.Azhagappan
	Database model	USN-9	Creating a database model with sql lite	2	High	K.Udhayakumar
Sprint-2	Setting up IBM DB2	USN-10	Creating and setting up IBM db2 database	2	High	R.Nithinton Prakash
	Set up IBM DB2 in flask	USN-11	Installing and setting up the necessary tools in the flask app	2	High	S.Thaneesh
	Integrating IBM DB2	USN-12	Integrating IBM db2 with python flask api	2	High	T.Azhagappan
	Sending data in UI	USN-13	Sending and connecting API request response data with UI	2	High	K.Udhayakumar
Sprint-3	IBM Watson assistant	USN-14	Creating IBM Watson assistant for chatbot service to the user	2	Medium	R.Nithinton Prakash
	Setting up sendgrid	USN-15	Creating sendgrid account and setting up the necessary libraries in the flask app	1	Medium	S.Thaneesh
	Integrating sendgrid	USN-16	By integrating sendgrid service you can able to receive emails with the python flask	1	Medium	T.Azhagappan
	Integrating chantJS	USN-17	By integrating chatJS in the dashboard the user can overview their thing expense	2	Medium	K.Udhayakumar
Sprint-4	Containerizing app	USN-18	Containerizing the flask application into a docker container usage	2	Medium	S.Thaneesh
	Uploading to IBM cloud registry	USN-19	Uploading the docker container image to IBM Cloud registry is useful in deployment	2	Medium	T.Azhagappan
	Deploying in kubernetes	USN-20	Deploying the docker container image from to the kubernetes	2	High	R.Nithinton Prakash

6.2 Sprint Delivery Schedule



Project Planning Phase

Sprint Delivery Plan

Date	28 OCTOBER 2022
Team ID	PNT2022TMID34858
Project Name	Project - Personal Expense Tracker
Maximum Marks	4 Marks

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	12	6 Days	23 Oct 2022	29 Oct 2022		
Sprint-2	8	6 Days	30 Oct 2022	04 Nov 2022		
Sprint-3	6	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	6	6 Days	14 Nov 2022	19 Nov 2022		

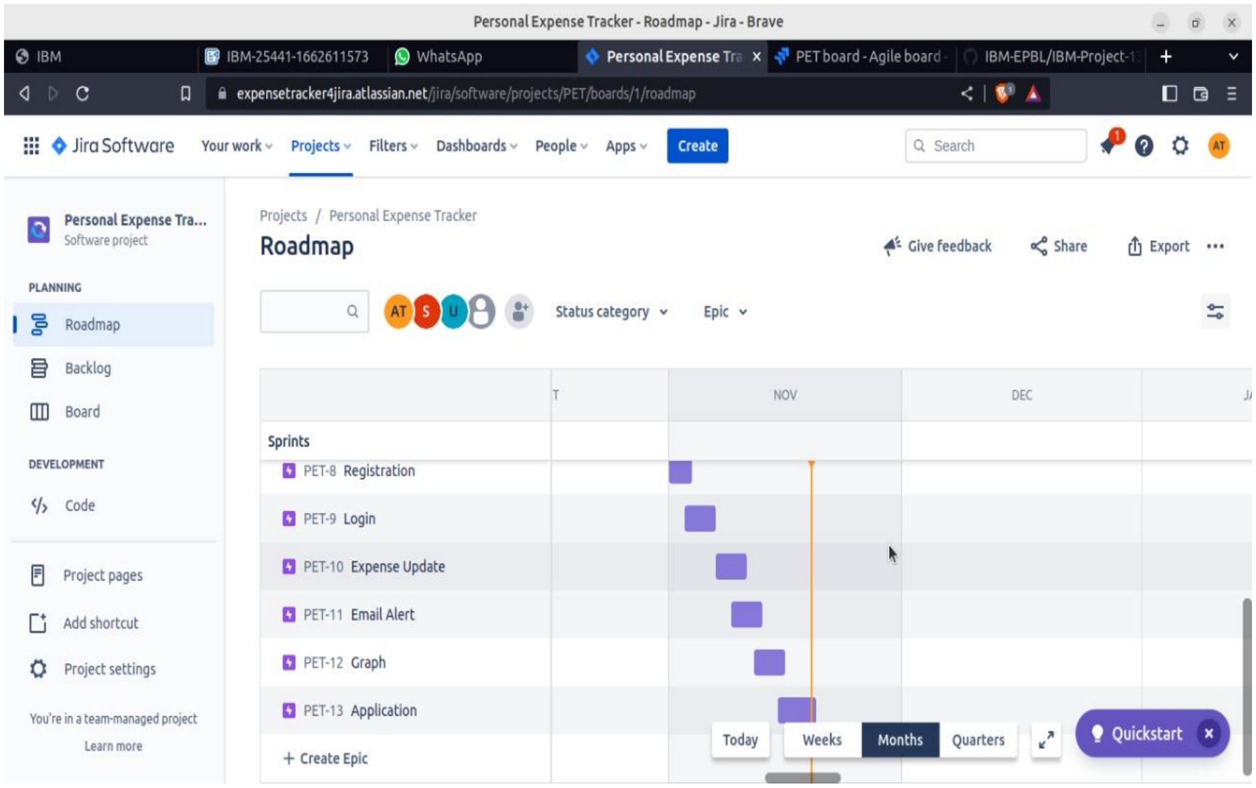
Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

AV=DURATION/VELOCITY

=20/6

=3.33

6.3 Reports from JIRA



Projects / Personal Expense Tracker

Roadmap

Sprints

PET-8 Registration

PET-9 Login

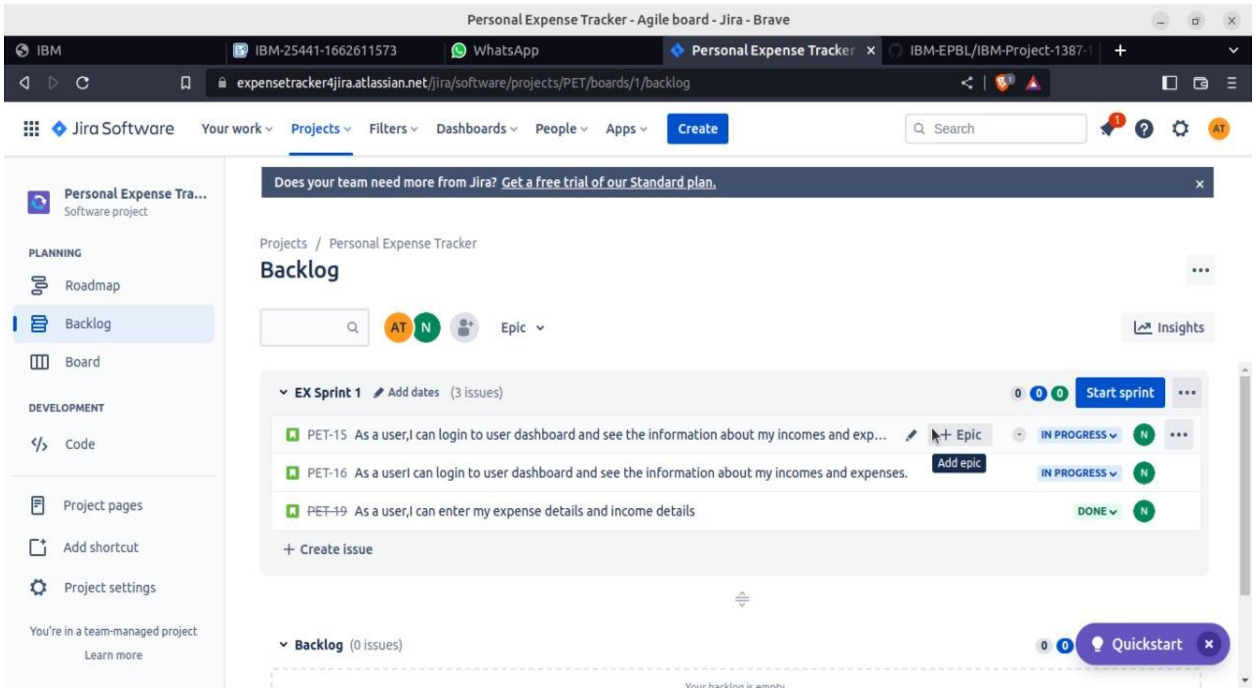
PET-10 Expense Update

PET-11 Email Alert

PET-12 Graph

PET-13 Application

+ Create Epic



Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Personal Expense Tracker

Backlog

EX Sprint 1 Add dates (3 issues)

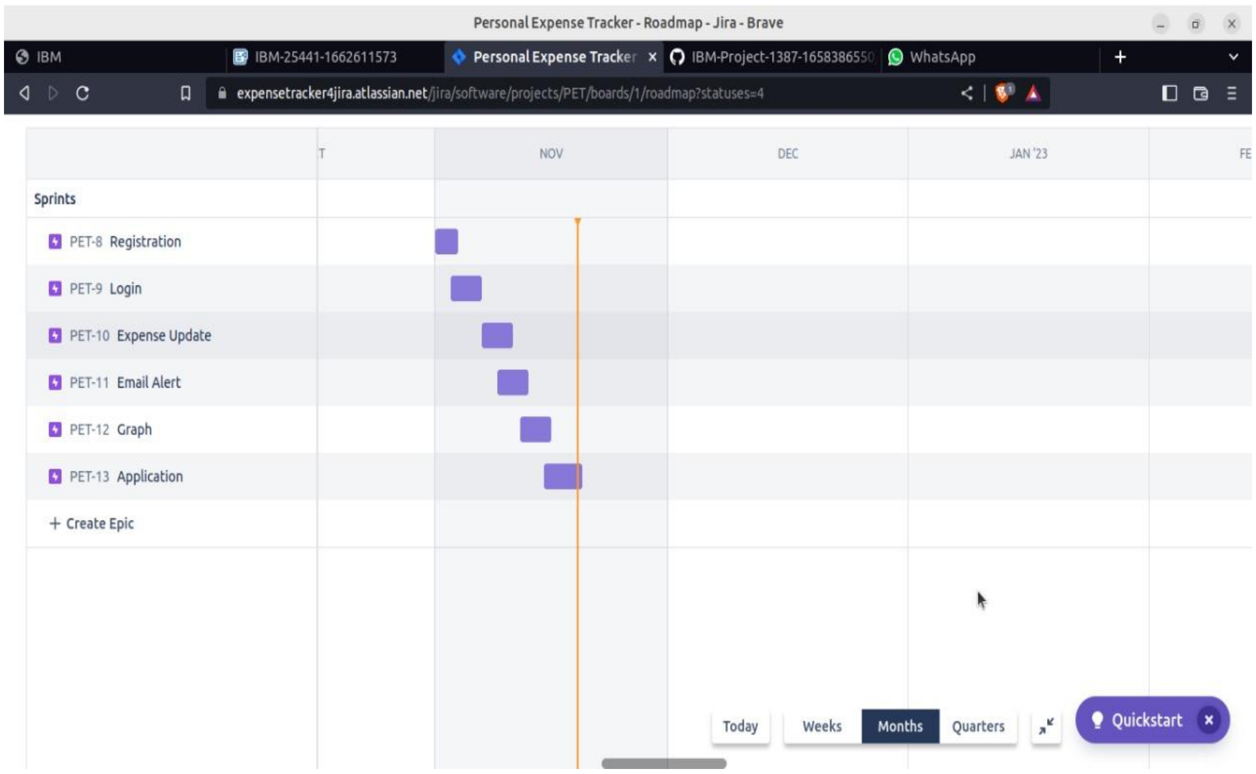
PET-15 As a user,I can login to user dashboard and see the information about my incomes and exp... IN PROGRESS

PET-16 As a userI can login to user dashboard and see the information about my incomes and expenses. IN PROGRESS

~~PET-19~~ As a user,I can enter my expense details and income details DONE

+ Create issue

Backlog (0 issues)


Personal Expense Tracker - Roadmap - Jira - Brave
IBM
IBM-25441-1662611573
Personal Expense Tracker
IBM-Project-1387-1658386550
WhatsApp
expensetracker4jira.atlassian.net/jira/software/projects/PET/boards/1/roadmap?statuses=4
T
NOV
DEC
JAN '23
FE
Sprints
PET-8 Registration
PET-9 Login
PET-10 Expense Update
PET-11 Email Alert
PET-12 Graph
PET-13 Application
+ Create Epic
Today
Weeks
Months
Quarters
Quickstart

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

Tracking income and expenses: Monitoring the income and tracking all expenditures.

Reports: The expense tracking app generates and sends reports to give a detailed insight about profits, losses, budgets, income, balance sheets, etc.,

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.

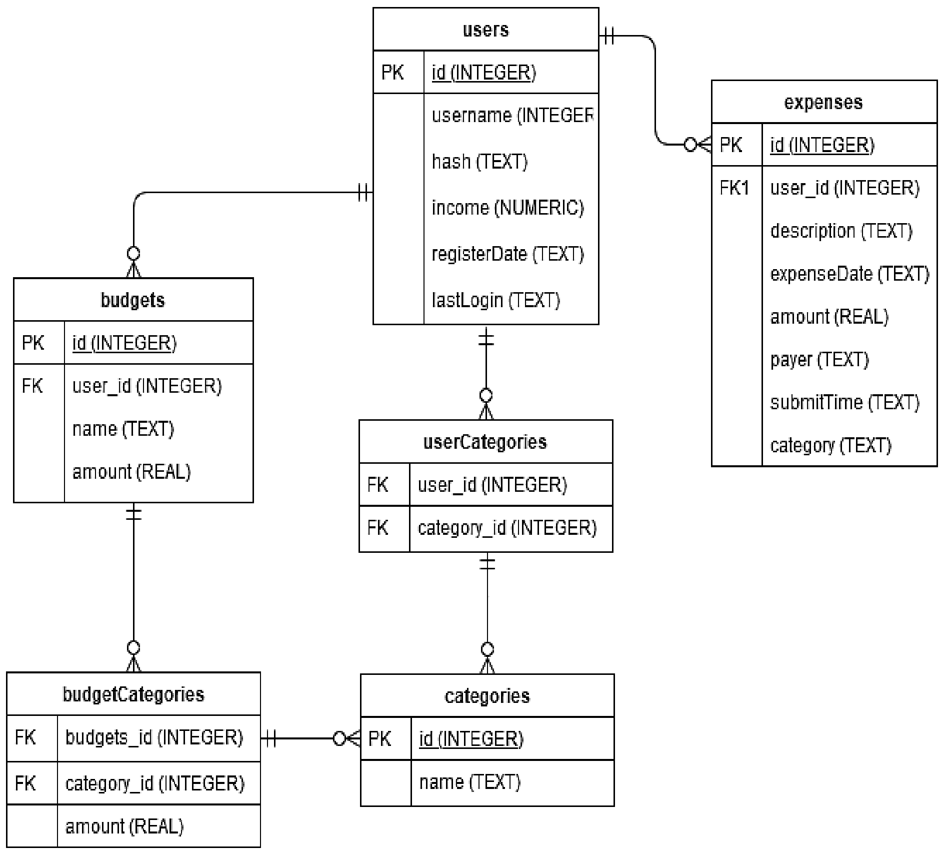
Budget Vs. Actual Spent: This is one of the most common features in an expense tracking mobile app. The user gets a detailed insight into the real-time income and expenditure. Thus, you can plan your budget strategically to reduce unnecessary expenses.

7.2 Feature 2

All the features that are mostly present in any regular expense tracking app. However, if there is any particular need for your business, you can also have a unique app get designed that caters to your business demands synchronized into your app.

At last, everything comes down to how much you can spend. It would help if you kept in mind your profits because that is why you are running a business. However, if you have to invest in something that will be a long-term investment, you cannot overlook it as well.

7.3 Database Schema (if Applicable)



8. TESTING

8.1 Test Cases

1.User Authentication
2.Add Income
3.Add Expense
4.Report View
5.Security

8.2 User Acceptance Testing

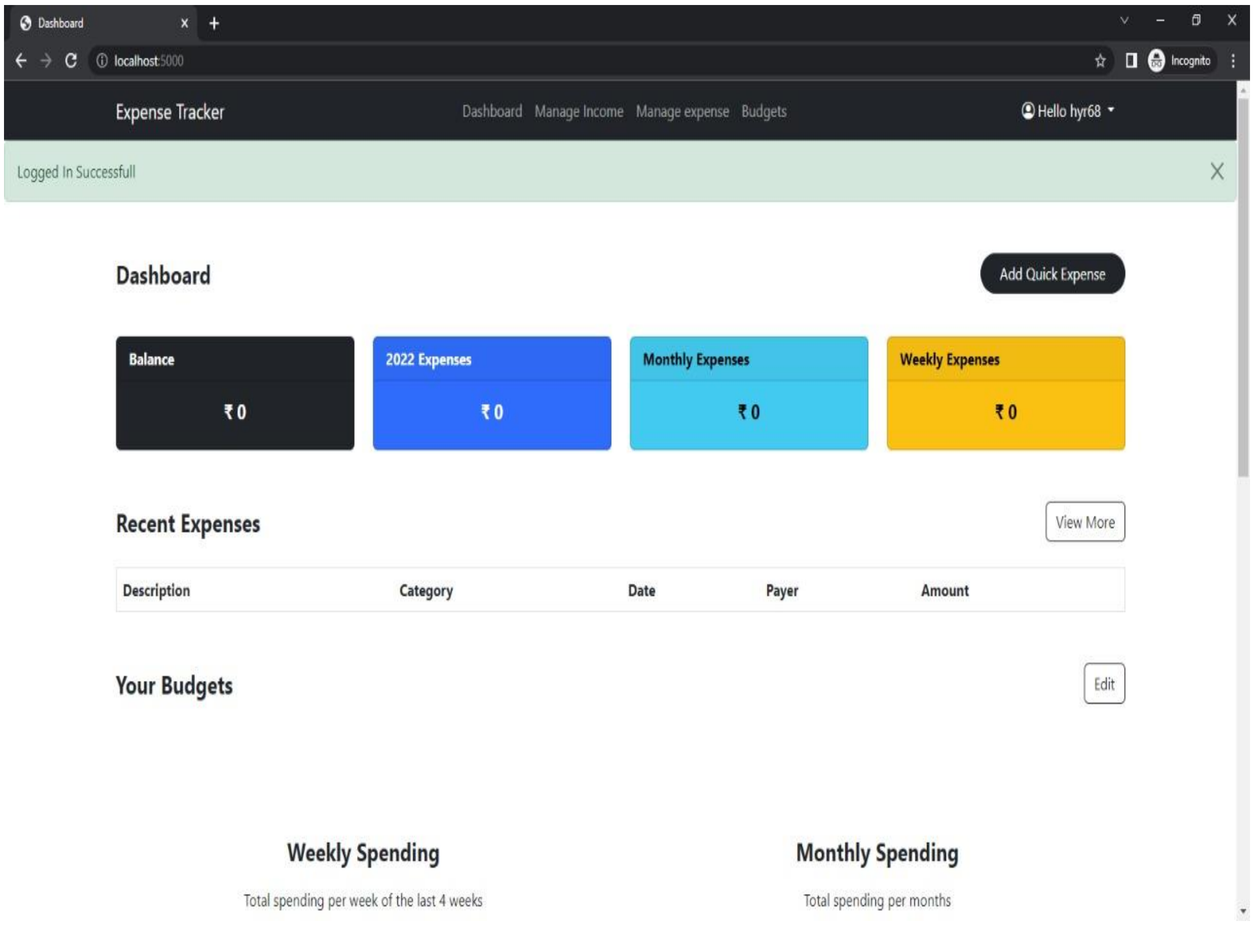
These types of test cases validate the product from the end user's perspective. An end user or client conducts user acceptance tests in a testing environment to validate the end-to-end flow of the product.

User acceptance tests can come in handy when expenses requirements.

User acceptance test case example: Validate that a user can register for a new account and that they receive an email confirmation.

9. RESULTS

9.1 Performance Metrics



Logged In Successfull

Dashboard

Balance	2022 Expenses	Monthly Expenses	Weekly Expenses
₹0	₹0	₹0	₹0

Recent Expenses

Description	Category	Date	Payer	Amount

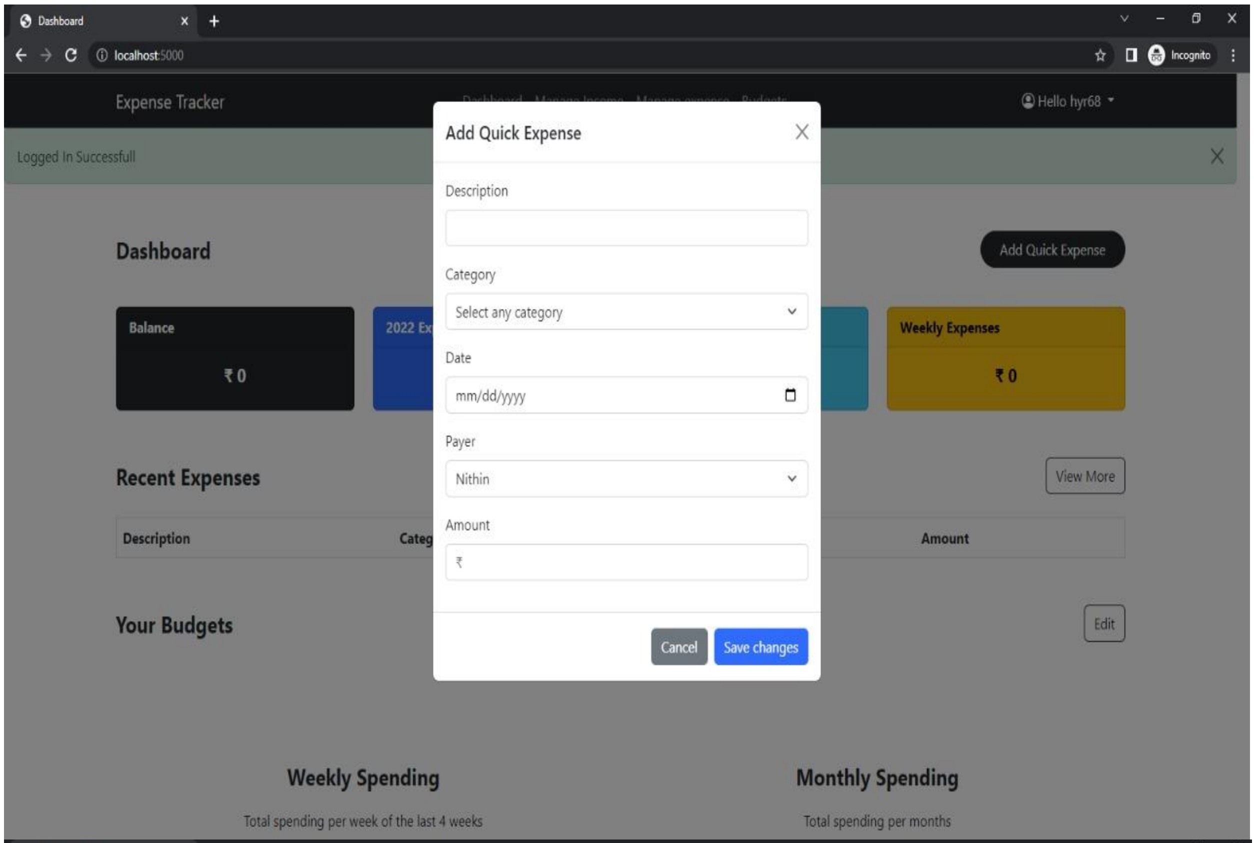
Your Budgets

Weekly Spending

Total spending per week of the last 4 weeks

Monthly Spending

Total spending per months


Dashboard

localhost:5000

Expense Tracker

Hello hyr68

Logged In Successfull

Dashboard

Add Quick Expense

Balance ₹ 0

2022 Ex

Weekly Expenses ₹ 0

Recent Expenses

View More

Description	Categ	Amount

Your Budgets

Edit

Weekly Spending

Total spending per week of the last 4 weeks

Monthly Spending

Total spending per months

Add Quick Expense

Description

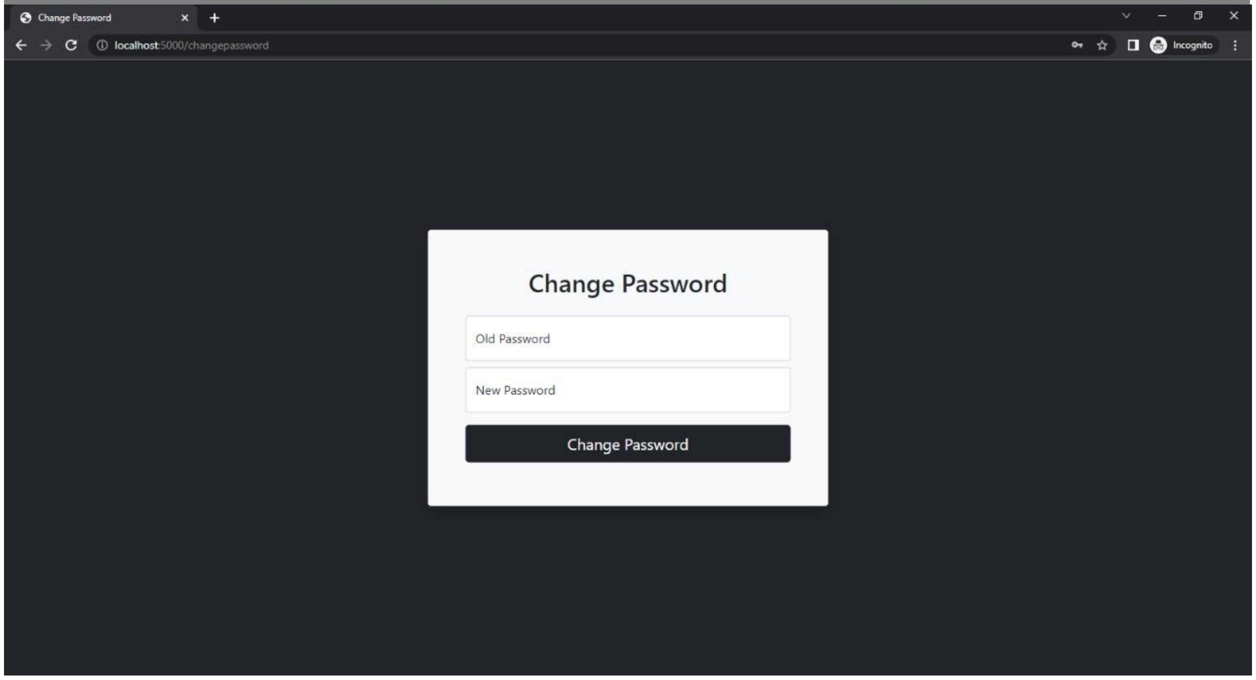
Category: Select any category

Date: mm/dd/yyyy

Payer: Nithin

Amount: ₹

Cancel Save changes


Change Password

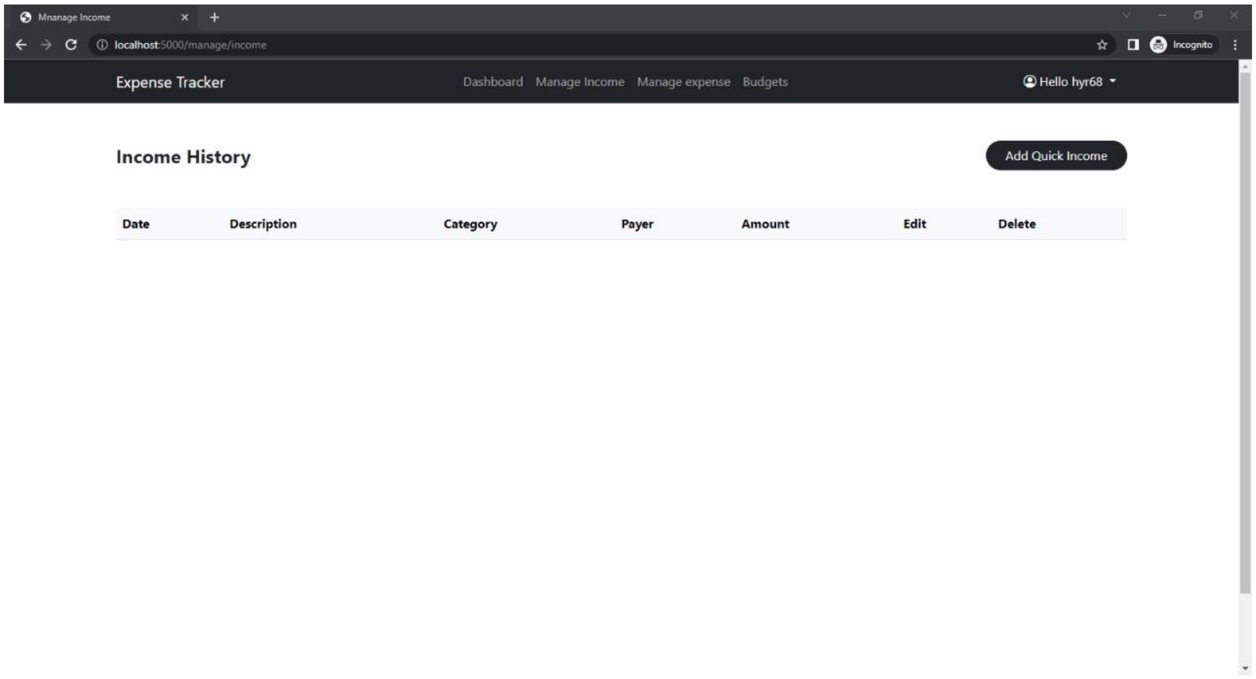
localhost:5000/changepassword

Change Password

Old Password

New Password

Change Password



Income History

Date	Description	Category	Payer	Amount	Edit	Delete

10. ADVANTAGES & DISADVANTAGES

Advantages:

With a daily expense manager, you will be able to allocate money to different priorities and this will also help you cut down on unnecessary spending. As a result, you will be able to save and be able to keep worry at bay. A daily money tracker helps you budget your money so that you use it wisely.

Disadvantages:

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.

11. CONCLUSION

Developing a personal expense tracker all expenses and spending is a crucial aspect of personal finances. Set aside a fixed amount in a savings account, they say you should always have three months work of your living expenses in a savings account in case of emergencies. The importance of actually seeing my spending on my budget sheet was enlightening. 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.

12. FUTURE SCOPE

1) It will have various options to keep record (for example Food, Travelling Fuel, Salary etc.).

2) Automatically it will keep on sending notifications for our daily expenditure.

3)In today’s busy and expensive life, we are in a great rush to make moneys, but at the end of the month we broke off. As we are unknowingly spending money on title and unwanted things. So, we have come over with the plan to follow our profit.

4) Here user can define their own categories for expense type like food, clothing, rent and bills where they have to enter the money that has been spend and likewise can add some data in extra data to indicate the expense.

5) Increase efficiency and customer satisfaction with an app aligned to their needs.

13. APPENDIX

Source Code

```
from flask import Flask,render_template,redirect,url_for,request,session,flash
from flask_sqlalchemy import SQLAlchemy
from datetime import timedelta, date

todays_date = date.today()


app = Flask(__name__)
app.secret_key = "made by humans"
app.permanent_session_lifetime = timedelta(days=6)
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///expense.db"
app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = False
db = SQLAlchemy(app)

#creating db models
class User(db.Model):
    id = db.Column(db.Integer,primary_key=True)
    email = db.Column(db.String(50),unique=True,nullable=False)
    fname = db.Column(db.String(20),nullable=False)
    lname = db.Column(db.String(20))
    pswd = db.Column(db.String(50),nullable=False,default=None)
    expense = db.relationship('Expense', backref='user')
    income = db.relationship('Income', backref='user')
    payer = db.relationship('Payer', backref='user')

    def __repr__(self):
        return f'<User "{self.fname}">'

class  Expense(db.Model):
    id = db.Column(db.Integer,primary_key=True)
    description = db.Column(db.String(50),nullable=True,default=None)
    category = db.Column(db.String(50),nullable=True,default=None)
    amount = db.Column(db.Integer,nullable=False)
    date = db.Column(db.String(10),nullable=True,default=None)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))

    def __repr__(self):
        return f'<Expense "{self.amount}">'

class  Income(db.Model):
    id = db.Column(db.Integer,primary_key=True)
    description = db.Column(db.String(50),nullable=True,default=None)
    amount = db.Column(db.Integer,nullable=False)
    date = db.Column(db.String(10),nullable=True,default=None)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))

    def __repr__(self):
        return f'<Income "{self.amount}">'

class  Payer(db.Model):
    id = db.Column(db.Integer,primary_key=True)
    name = db.Column(db.String(50),nullable=False)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))

    def __repr__(self):
        return f'<Payer "{self.name}">'

#Routes
@app.route('/',methods = ["POST","GET"])
def dashboard():
    if ("email" in session) and ("pswd" in session) and (session["pswd"] ==
User.query.filter_by(email=session["email"]).first().pswd) :
        if request.method == "POST":
            if request.form['submit'] == 'quick_expense' and request.form['amount']:
                description = request.form['description']
                category = request.form['category']
                date = request.form['date']
                #payer = request.form['payer']
                amount = request.form['amount']
            else:
                flash("Enter amount")
                return redirect(url_for('dashboard'))

            if session["email"]:
                usr_id = User.query.filter_by(email=session['email']).first().id
                expense =
Expense(description=description,category=category,date=date,amount=amount,user_id=usr_id)
                db.session.add(expense)
                db.session.commit()
                flash("Expense amount "+str(expense.amount)+"₹ added successfully")

            return redirect(url_for('dashboard'))
        else:
            flash("Logged In Successfull")
            #Get view
            totalExpense = 0
            monthlyExTotal = 0
            monthlyExpenses = [0,0,0,0,0,0,0,0,0,0,0,0]
            weeklyExpenses = [0,0,0,0]
            weeklyExTotal = 0
            for expense in User.query.filter_by(email=session['email']).first().expense:
                if expense.date[:4] == str(todays_date.year):
                    totalExpense += expense.amount #total year expense
                    for month in range(12):
                        if  int(expense.date[5:7])  == month+1:
                            monthlyExpenses[month] += expense.amount #monthly expense list
                if expense.date[:4] == str(todays_date.year) and expense.date[5:7] ==
str(todays_date.month):
                    monthlyExTotal += expense.amount #total monthly expense
                    if int(expense.date[-2:])<8:
                        if todays_date.day < 8 : weeklyExTotal += expense.amount
                        weeklyExpenses[0] += expense.amount
                    elif int(expense.date[-2:])<16:
                        if todays_date.day < 16 : weeklyExTotal += expense.amount
                        weeklyExpenses[1] += expense.amount
                    elif int(expense.date[-2:])<24:
                        if todays_date.day < 24 : weeklyExTotal += expense.amount
                        weeklyExpenses[2] += expense.amount
                    elif int(expense.date[-2:])<32:
                        if todays_date.day < 32 : weeklyExTotal += expense.amount
                        weeklyExpenses[3] += expense.amount

            totalIncome = 0
            monthlyInTotal = 0
            monthlyIncomes = [0,0,0,0,0,0,0,0,0,0,0,0]
            weeklyIncomes = [0,0,0,0]
            weeklyInTotal = 0
            for income in User.query.filter_by(email=session['email']).first().income:
                if expense.date[:4] == str(todays_date.year):
                    totalIncome += income.amount #total year income
                    for month in range(12):
                        if  int(income.date[5:7])  == month+1:
                            monthlyIncomes[month] += income.amount #monthly expense list
                if income.date[:4] == str(todays_date.year) and income.date[5:7] ==
str(todays_date.month):
                    monthlyInTotal += income.amount #total monthly expense
                    if int(income.date[-2:])<8:
                        if todays_date.day < 8 : weeklyInTotal += income.amount
                        weeklyIncomes[0] += income.amount
                    elif int(expense.date[-2:])<16:
                        if todays_date.day < 16 : weeklyInTotal += income.amount
                        weeklyIncomes[1] += income.amount
                    elif int(expense.date[-2:])<24:
                        if todays_date.day < 24 : weeklyInTotal += income.amount
                        weeklyIncomes[2] += income.amount
                    elif int(expense.date[-2:])<32:
                        if todays_date.day < 32 : weeklyInTotal += income.amount
                        weeklyIncomes[3] += income.amount

            data ={
                "balance": totalIncome-totalExpense,
                "yearlyStats": totalExpense,
                "monthlyStats": monthlyExTotal,
                "weeklyStats": weeklyExTotal,
                "monthlyEx": monthlyExpenses,
                "weeklyEx": weeklyExpenses,
                "monthlyIn": monthlyIncomes,
                "weeklyIn": weeklyIncomes,
                "recent": User.query.filter_by(email=session["email"]).first().expense[:6]
            }
            return render_template('dashboard.html',data = data)
        else:
            return redirect(url_for('login'))

@app.route('/login',methods = ["POST","GET"])
def login():
        if request.method =="POST":
            email = request.form['email']
            pswd = request.form['pswd']

            if ((User.query.filter_by(email=email).first().email == email) and
(User.query.filter_by(email=email).first().pswd == pswd)):
            session["email"] = User.query.filter_by(email=email).first().email
            session["pswd"] = User.query.filter_by(email=email).first().pswd
            session["fname"] = User.query.filter_by(email=email).first().fname

            if request.form.get('check') == "remember":
                session.permanent = True
            else:
                session.permanent = False
            else:
            flash("Incorrect Credentials")

            return redirect(url_for('dashboard'))
        else:
        if ("email" in session) and ("pswd" in session):
            flash("Already Logged In")
            return redirect(url_for('dashboard'))
        return render_template('login.html')

@app.route('/signup',methods = ["POST","GET"])
def signup():
    if request.method =="POST":
        if (request.form['fname'] and request.form['email'] and request.form['pswd']):
            fname = request.form['fname']
            lname = request.form['lname']
            email = request.form['email']
            pswd = request.form['pswd']
        else:
            flash("Enter All Details")
            return redirect(url_for('signup'))

        if not User.query.filter_by(email=email).first():
            #print(fname+" "+lname+"\n"+email)
            usr = User(email=email,fname=fname,lname=lname,pswd=pswd)
            db.session.add(usr)
            db.session.commit()
            flash(User.query.filter_by(email=email).first().email+" added successfully")
            return redirect(url_for('login'))
        else:
            flash("User already Exist")
            return redirect(url_for('signup'))
    else:
        return render_template('signup.html')

@app.route('/resetpassword',methods = ["POST","GET"])
def reset():
    if request.method =="POST":
        if (request.form['email'] and request.form['npswd']):
            email = request.form['email']
            pswd = request.form['npswd']
        else:
            flash("Enter All Details")
            return redirect(url_for('reset'))
        if User.query.filter_by(email=email).first():
            User.query.filter_by(email=email).first().pswd = pswd
            db.session.commit()
            return redirect(url_for('login'))
        else:
            flash("User Not Exist")
            return redirect(url_for('reset'))
    else:
        return render_template('resetpassword.html')

@app.route('/changepassword',methods = ["POST","GET"])
def changePwd():
    if request.method =="POST":
        if (request.form['opswd'] and request.form['npswd']):
            email = request.form['email']
            opswd = request.form['opswd']
            npswd = request.form['npswd']
        else:
            flash("Enter All Details")
            return redirect(url_for('changePwd'))
        if User.query.filter_by(email=email).first():
            if User.query.filter_by(email=email).first().pswd == opswd :
                User.query.filter_by(email=email).first().pswd = npswd
                session["pswd"] = User.query.filter_by(email=email).first().pswd
                db.session.commit()
                flash("Password Changed successfully")
            else:
                flash("Incorrect password")
                return redirect(url_for('changePwd'))
            return redirect(url_for('dashboard'))
        else:
            flash("Enter All Details")
            return redirect(url_for('changePswd'))

    else:
        return render_template('changePassword.html')

@app.route('/logout')
def logout():
    session.pop("email",None)
    session.pop("pswd",None)
    flash("logged Out Successfull")
    return redirect(url_for('login'))

@app.route('/manage/income',methods = ["POST","GET"])
def income():
    if request.method =="POST":
            #Quick Income
            if request.form["submit"] == "quick_income":
                if request.form['amount']:
                    description = request.form['description']
                    date = request.form['date']
                    #payer = request.form['payer']
                    amount = request.form['amount']

                    if session["email"]:
                        usr_id = User.query.filter_by(email=session['email']).first().id
                        income =
Income(description=description,date=date,amount=amount,user_id=usr_id)
                        db.session.add(income)
                        db.session.commit()
                        flash("Income amount "+str(income.amount)+"₹ added successfully")
                    else:
                        flash("Income not added")
                else:
                    flash("Enter amount")
                    return redirect(url_for('income'))

            #edit Income
            elif request.form['submit'] == "edit_income":
                if request.form['Eamount']:
                    description = request.form['Edescription']
                    category = request.form['Ecategory']
                    date = request.form['Edate']
                    #payer = request.form['payer']
                    amount = request.form['Eamount']
                    income_id = request.form['Eid']
                    print(amount)

                    if session["email"]:
                        usr_id = User.query.filter_by(email=session['email']).first().id
                        Income.query.filter_by(id= income_id).first().descrption = description
                        Income.query.filter_by(id= income_id).first().category = category
                        Income.query.filter_by(id= income_id).first().date = date
                        Income.query.filter_by(id= income_id).first().amount = amount
                        db.session.commit()
                        flash("Income amount "+str(Income.query.filter_by(id=
income_id).first().amount)+"₹ added successfully")
                    else:
                        flash("Income not added")
                else:
                    flash("Enter amount")
                    return redirect(url_for('income'))

            #delete Income
            elif request.form['submit'] == "delete_income":
                if request.form['incomeId']:
                    incomeId = request.form['incomeId']
                    income = Income.query.filter_by(id= incomeId).first()
                    db.session.delete(income)
                    db.session.commit()

            return redirect(url_for('income'))
    else:
        incomes = User.query.filter_by(email=session['email']).first().income
        return render_template('income.html', data = incomes)

@app.route('/manage/expense',methods = ["POST","GET"])
def expense():
    if request.method =="POST":
        #Quick Expense
        if request.form["submit"] == "quick_expense":
            if request.form['amount']:
                description = request.form['description']
                category = request.form['category']
                date = request.form['date']
                #payer = request.form['payer']
                amount = request.form['amount']

                if session["email"]:
                    usr_id = User.query.filter_by(email=session['email']).first().id
                    expense =
Expense(description=description,category=category,date=date,amount=amount,user_id=usr_id)
                    db.session.add(expense)
                    db.session.commit()
                    flash("Expense amount "+str(expense.amount)+"₹ added successfully")
                else:
                    flash("Expense not added")
            else:
                flash("Enter amount")
                return redirect(url_for('expense'))

        #edit Expense
        elif request.form['submit'] == "edit_expense":
            if request.form['Eamount']:
                description = request.form['Edescription']
                category = request.form['Ecategory']
                date = request.form['Edate']
                #payer = request.form['payer']
                amount = request.form['Eamount']
                expense_id = request.form['Eid']
                print(amount)

                if session["email"]:
                    usr_id = User.query.filter_by(email=session['email']).first().id
                    Expense.query.filter_by(id=expense_id).first().descrption = description
                    Expense.query.filter_by(id=expense_id).first().category = category
                    Expense.query.filter_by(id=expense_id).first().date = date
                    Expense.query.filter_by(id=expense_id).first().amount = amount
                    db.session.commit()
                    flash("Expense amount
"+str(Expense.query.filter_by(id=expense_id).first().amount)+"₹ added successfully")
                else:
                    flash("Expense not added")
            else:
                flash("Enter amount")
                return redirect(url_for('expense'))

        #delete Expense
        elif request.form['submit'] == "delete_expense":
            print("Hello")
            if request.form['expenseId']:
                expenseId = request.form['expenseId']
                expense = Expense.query.filter_by(id=expenseId).first()
                print(expense)
                db.session.delete(expense)
                db.session.commit()

        return redirect(url_for('expense'))
    else:
        expenses = User.query.filter_by(email=session['email']).first().expense
        return render_template('expenses.html', data = expenses)

if __name__ == '__main__':
    db.create_all()
    app.run(debug=True)
```

GitHub & Project Demo Link

Github: IBM-Project-25441-1659963561/Personal Expense Tracker/Final Deliverables at main · IBM-EPBL/IBM-Project-25441-1659963561 (github.com)

Demo link: Recordit: Record screencasts fast & free! with GIF Support!