CLOUD APPLICATION DEVELOPMENT – BANKING & FINANCE

PERSONAL EXPENSE TRACKER APPLICATION

PROJECT REPORT

Submitted by:

Jeffrey D Daniel Victor (950019104017)

Jestus A (950019104018)

Safeer Ali S (950019104039)

Aadithya Prasad P (950019104702)

ANNA UNIVERSITY REGIONAL CAMPUS – TIRUNELVELI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AUGUST 2022 – NOVEMBER 2022

CHAPTER-1

INTRODUCTION

1.1 Project Overview

Personal finance entails all the financial decisions and activities that a Finance app makes life easier by helping the user to manage their finances efficiently. A personal finance app will not only help them with budgeting and accounting but also give helpful insights about money management. In this world, people spend lots of money than earning it. Most of the time they spend more money on unwanted things. By this activity they are facing so much struggles to run their family at the end of the month or year. By solving this issue, an application which is used to add the expenses of a user and spend money according to that plan. If a user spends additional money, this application notifies them through their mail. Also, by developing this application financial issue in a family will not be arise anymore.

1.2 Purpose

The purpose of this expense tracker application is to make people to spend money on useful things and to avoid many financial problems in a family. Financial issue occurs when people spend lots of money on buying unwanted stuffs and things. They do not have any plan to spend it. So, to avoid these issues this application is created and it is very useful for all kind of people.

CHAPTER-2
LITERATURE REVIEW:

***Application Name**: Spendee.*

Spendee is a free money tracker app for budget planning and money management. It is useful to someone who just wants to track daily expenses, instead of being confused by the complicated expenses bookkeeping.

Pros

Free to use: Spendee has a free plan that provides limited functionality for users. The most useful tools, however, are reserved for the paid subscription plans.

Easy-to-use design: The Spendee app sports a simple design that optimizes the user experience. The beautiful interface allows for a smooth signup process, easy navigation and generally attractive displays and charts.

International availability: Spendee is available in Canada and countries in North America, South America, Asia, Europe and Africa. Whichever country you are in, you can set up a Spendee account, and gain access to more than 2,500 banks globally. You can create your account with which currency you desire. You are also free to switch currencies depending on your immediate need.

Bank-level security: Spendee deploys tight security measures to ensure that customers' data is securely protected. All transactions and information exchange are encrypted such that only parties authorized by you have access to it. Spendee's servers are currently hosted on Google Cloud, a trusted and tested security-oriented platform.

One-glance overview of your money: The Spendee app provides you with an opportunity to link all of your financial institutions with your Spendee account. You can synchronize different banks, online financial platforms like PayPal, as well as cryptocurrency trading platforms such as Binance and Coinbase. This enables you to see all your important financial details in one place.

Monitor and regulate expenditure: Seeing all your money in one place gives you a feel of the bigger picture, and you can make more informed and well-rounded financial decisions.

With your financial information neatly displayed with insightful analytics, you can take steps to optimize your spendings and savings to reach your desired financial goal.

Gain valuable financial knowledge: Spendee maintains an online blog that contains relevant tips and information to increase your awareness. Useful financial insights are also regularly disseminated on the platform to help you make more educated decisions.

Cons

Best services are limited to paid plans: Spendee operates on a three-tiered basis, each with its own cost. However, the most advanced tools are restricted to the Spendee Premium, which is the highest of all three tiers.

Problems with app updates: Android and iOS users of the Spendee app complain of bugs that come with new updates. On many occasions, currencies fail to display, automatic synchronization breaks down and error messages interrupt transactions.

Does not support some banks: Despite being available in many countries of the world, Spendee does not support some Canadian banks such as HSBC, the Bank of Montreal, the Equitable Bank, among others.

2.2 References

https://www.spendee.com/

2.3 Problem Statement Definition

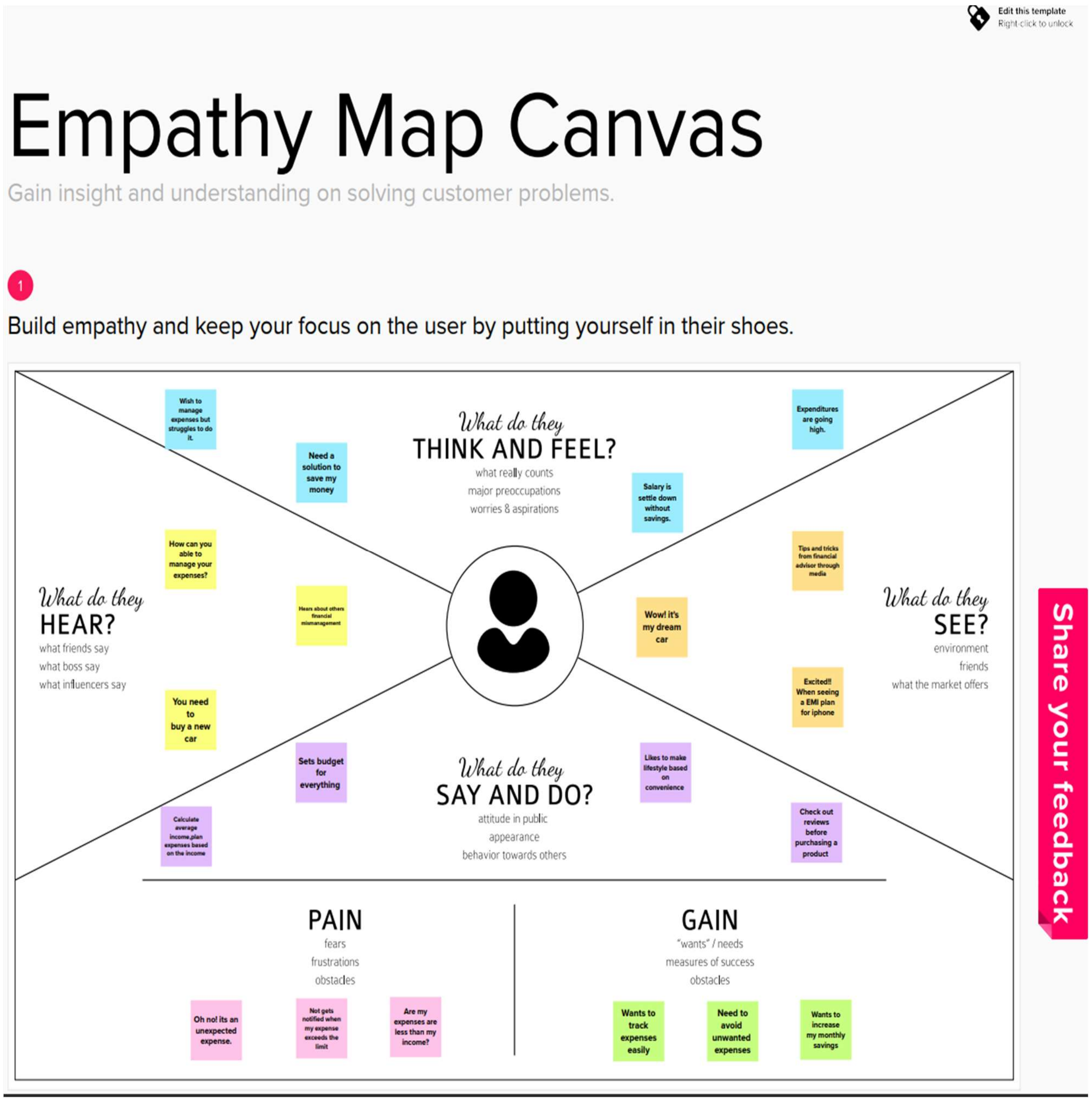
Many organizations have their own system to record their income and expenses, which they feel is the main key point of their business progress. It is good habit for a person to record daily expenses and earning but due to unawareness and lack of proper applications to suit their privacy, lacking decision making capacity people are using traditional note keeping methods to do so. Due to lack of a complete tracking system, there is a 2 constant overload to rely on the daily entry of the expenditure and total estimation till the end of the month.

Who does the problem affect?	People getting regular wages.
What is the issue?	The paper based expense tracker system does not provide the user portability , existing system only used on paper based records so unable to update anywhere expenses done and unable to update the location of the expense details disruptive that the proposed system.
When does the issue occurs?	When the digits could not be recognized correctly. When the transactions are not successful. When the elder people unable to understand the smaller handwritten digits. When the paper based expense tracker records are subjected to fire accident, flood, etc.
Where is the issue occurring?	The issue occurs when the person is unable to track his income and expenditure.
Why is it important that we fix the problem?	By solving this issue those people getting regular wages can track their expenses and avoid unwanted expenses.

CHAPTER-3

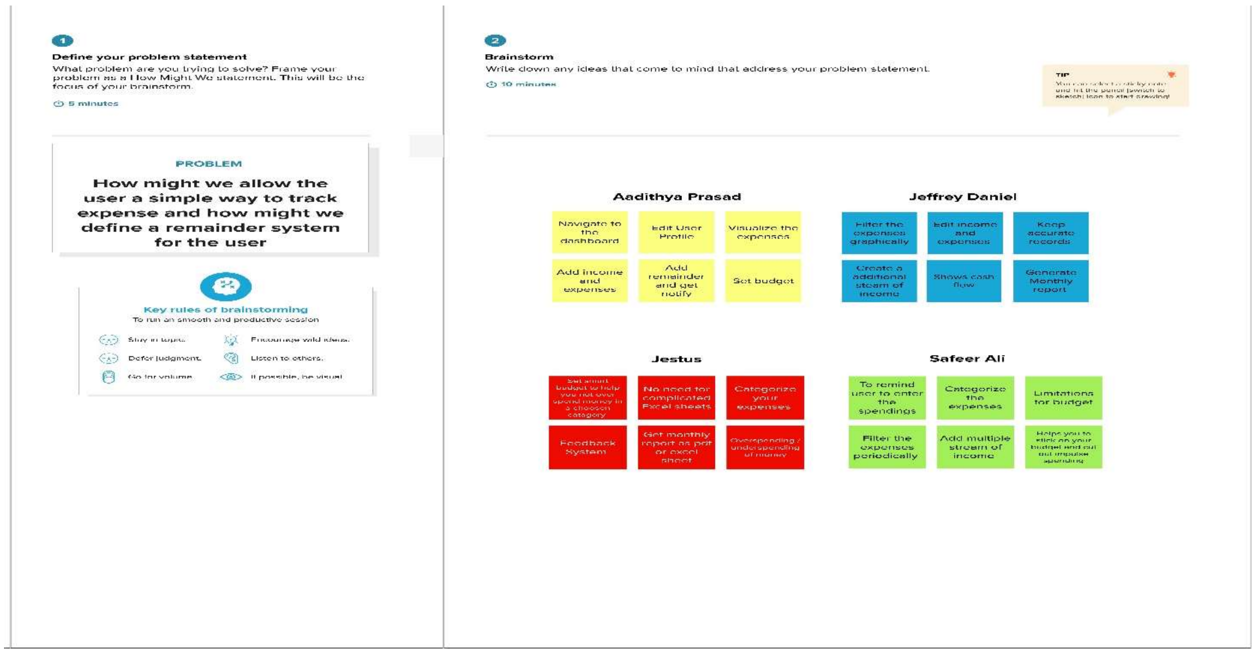
IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



Edit this template
Right-click to unlock
Empathy Map Canvas
Gain insight and understanding on solving customer problems.
1
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
Wish to manage expenses but struggles to do it.
Need a solution to save my money
Salary is settle down without savings.
Expenditures are going high.
What do they HEAR?
what friends say
what boss say
what influencers say
How can you able to manage your expenses?
Hears about others financial mismanagement
You need to buy a new car
What do they SEE?
environment
friends
what the market offers
Tips and tricks from financial advisor through media
Wow! it's my dream car
Excited!! When seeing a EMI plan for iphone
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Sets budget for everything
Calculate average income,plan expenses based on the income
Likes to make lifestyle based on convenience
Check out reviews before purchasing a product
PAIN
fears
frustrations
obstacles
Oh no! its an unexpected expense.
Not gets notified when my expense exceeds the limit
Are my expenses are less than my income?
GAIN
"wants" / needs
measures of success
obstacles
Wants to track expenses easily
Need to avoid unwanted expenses
Wants to increase my monthly savings
Share your feedback

3.2 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

How might we allow the user a simple way to track expense and how might we define a remainder system for the user

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

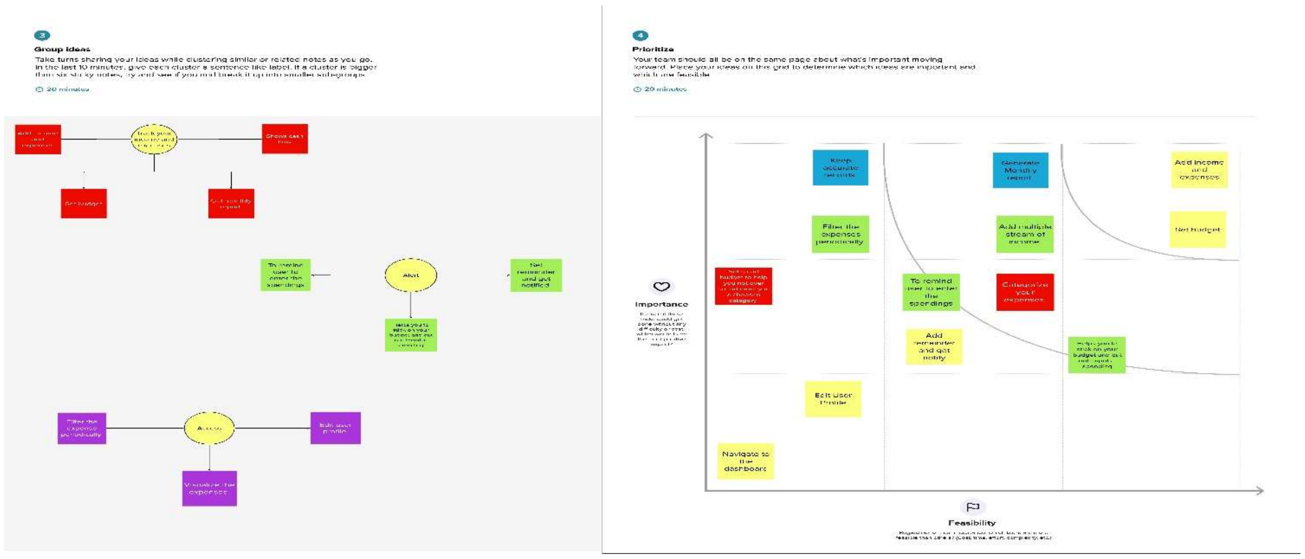
Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

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

Aadithya Prasad
Navigate to the dashboard
Edit User Profile
Visualize the expenses
Add income and expenses
Add remainder and get notify
Set budget
Jeffrey Daniel
Filter the expenses graphically
Edit income and expenses
Keep accurate records
Create a additional steam of income
Shows cash flow
Generate Monthly report
Jestus
Set smart budget to help you not over spend money in a choosen catagory
No need for complicated Excel sheets
Categorize your expenses
Feedback System
Get monthly report as pdf or excel sheet
Overspending / underspending of money
Safeer Ali
To remind user to enter the spendings
Categorize the expenses
Limitations for budget
Filter the expenses periodically
Add multiple stream of income
Helps you to stick on your budget and cut out impulse spending


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

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

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.

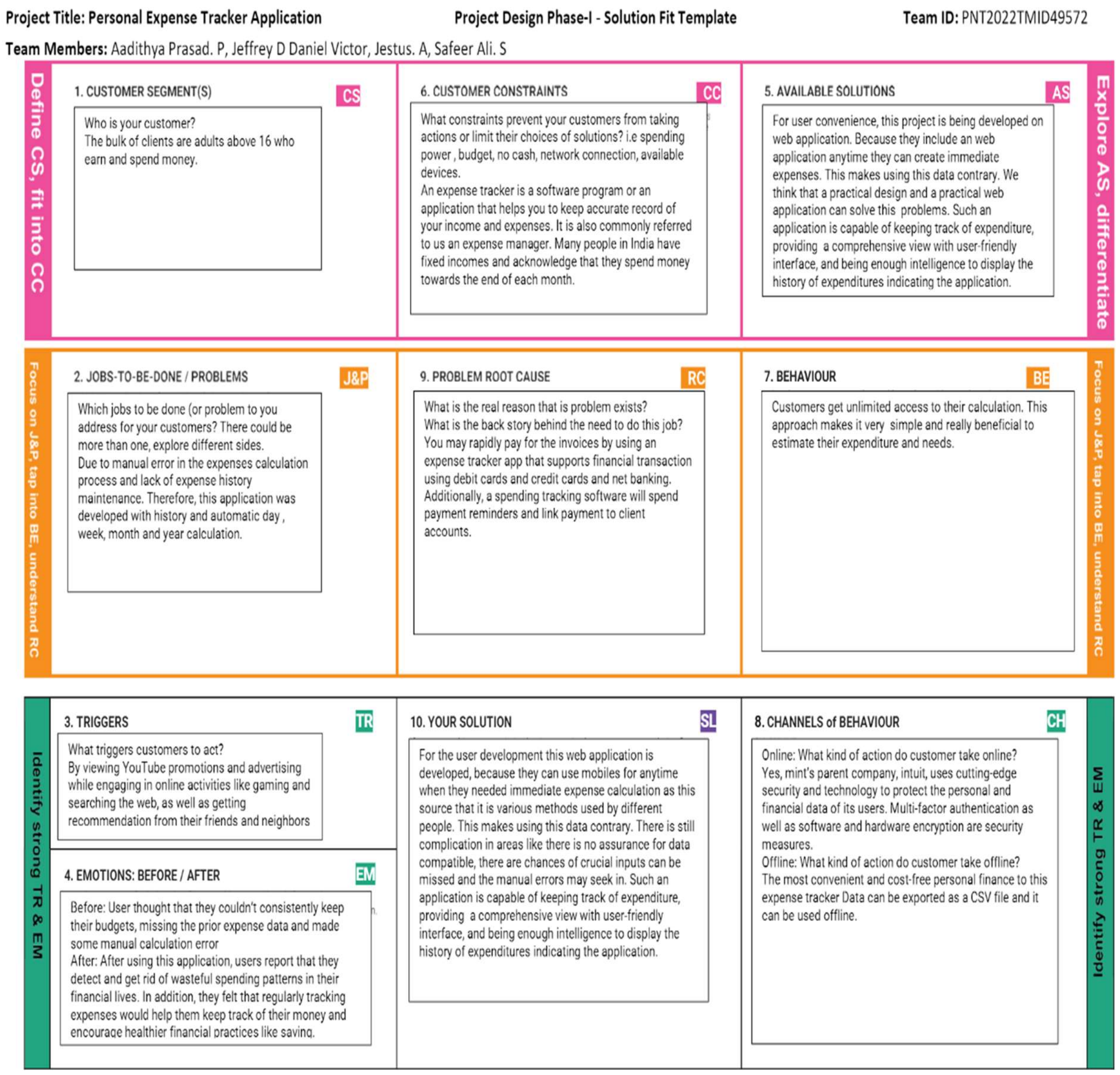
20 minutes

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

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	In paper-based expense tracker system it is difficult to track our monthly expenses manually. The paper-based expense records may get lost in case of fire accidents, flood etc.
2.	Idea / Solution description	This app makes your life easier by helping you to manage your finances efficiently. This personal expense app will not only help you with budgeting and accounting but also give you helpful insights about financial management.
3.	Novelty / Uniqueness	The user gets notified when their expense exceeds the limit and also it reminds the user when they forgot to make entry.
4.	Social Impact / Customer Satisfaction	It will help the people to track their expenses and also alerts when they exceed the limit of their budget.
5.	Business Model (Revenue Model)	We can provide the application in a subscription based.
6.	Scalability of the Solution	This application can handle large number of users simultaneously.

3.4 Problem Solution Fit



Project Title: Personal Expense Tracker Application **Project Design Phase-I - Solution Fit Template** **Team ID:** PNT2022TMID49572

Team Members: Aadithya Prasad. P, Jeffrey D Daniel Victor, Jestus. A, Safeer Ali. S

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
The bulk of clients are adults above 16 who earn and spend money.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking actions or limit their choices of solutions? i.e spending power , budget, no cash, network connection, available devices.
An expense tracker is a software program or an application that helps you to keep accurate record of your income and expenses. It is also commonly referred to us an expense manager. Many people in India have fixed incomes and acknowledge that they spend money towards the end of each month.

5. AVAILABLE SOLUTIONS AS

For user convenience, this project is being developed on web application. Because they include an web application anytime they can create immediate expenses. This makes using this data contrary. We think that a practical design and a practical web application can solve this problems. Such an application is capable of keeping track of expenditure, providing a comprehensive view with user-friendly interface, and being enough intelligence to display the history of expenditures indicating the application.

Explore AS, differentiate

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

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

Which jobs to be done (or problem to you address for your customers? There could be more than one, explore different sides.
Due to manual error in the expenses calculation process and lack of expense history maintenance. Therefore, this application was developed with history and automatic day , week, month and year calculation.

9. PROBLEM ROOT CAUSE RC

What is the real reason that is problem exists?
What is the back story behind the need to do this job?
You may rapidly pay for the invoices by using an expense tracker app that supports financial transaction using debit cards and credit cards and net banking. Additionally, a spending tracking software will spend payment reminders and link payment to client accounts.

7. BEHAVIOUR BE

Customers get unlimited access to their calculation. This approach makes it very simple and really beneficial to estimate their expenditure and needs.

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

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act?
By viewing YouTube promotions and advertising while engaging in online activities like gaming and searching the web, as well as getting recommendation from their friends and neighbors

4. EMOTIONS: BEFORE / AFTER EM

Before: User thought that they couldn't consistently keep their budgets, missing the prior expense data and made some manual calculation error
After: After using this application, users report that they detect and get rid of wasteful spending patterns in their financial lives. In addition, they felt that regularly tracking expenses would help them keep track of their money and encourage healthier financial practices like saving.

10. YOUR SOLUTION SL

For the user development this web application is developed, because they can use mobiles for anytime when they needed immediate expense calculation as this source that it is various methods used by different people. This makes using this data contrary. There is still complication in areas like there is no assurance for data compatible, there are chances of crucial inputs can be missed and the manual errors may seek in. Such an application is capable of keeping track of expenditure, providing a comprehensive view with user-friendly interface, and being enough intelligence to display the history of expenditures indicating the application.

8. CHANNELS of BEHAVIOUR CH

Online: What kind of action do customer take online?
Yes, mint's parent company, intuit, uses cutting-edge security and technology to protect the personal and financial data of its users. Multi-factor authentication as well as software and hardware encryption are security measures.
Offline: What kind of action do customer take offline?
The most convenient and cost-free personal finance to this expense tracker Data can be exported as a CSV file and it can be used offline.

Identify strong TR & EM

CHAPTER-4

REQUIREMENT ANALYSIS

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	User's registration details such as e-mail id and mobile number are collected through forms and also user name and password should be set by them.
FR-2	Login	Users need to enter the username and password which was already registered to log into the application.
FR-3	Wallet	The user should add their income and expense in the wallet.
FR-4	Expense Tracker	This application will graphically represent the expense of the user in the form of report.
FR-5	Notification	The user gets notified when their expense exceeds the limit which is already set by them.
FR-6	Category	The user will be allowed to select the category of their expense.

Non-functional Requirements:

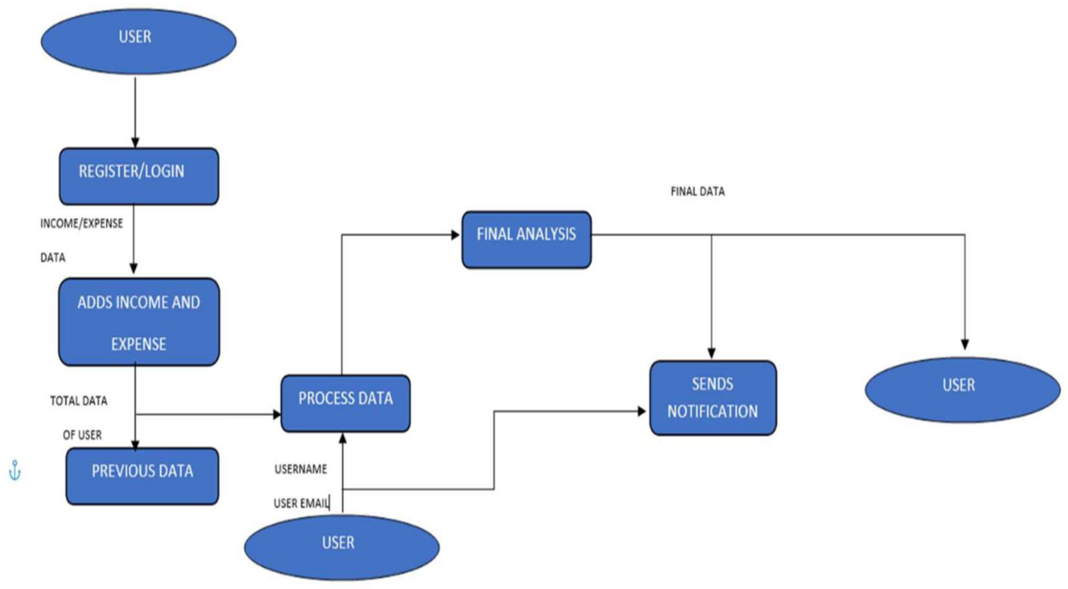
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Helps the user to maintain the accurate record of their income and expense.
NFR-2	Security	This application is very safe from cyber-attacks.
NFR-3	Reliability	Each data is stored in well built efficient data base schema. There is no risk of data loss.
NFR-4	Performance	The throughput of the application is increased due to the light weight data base support.
NFR-5	Availability	This application will have a 100% up-time.
NFR-6	Scalability	This application has ability to appropriately handle increasingly demands.

CHAPTER-5

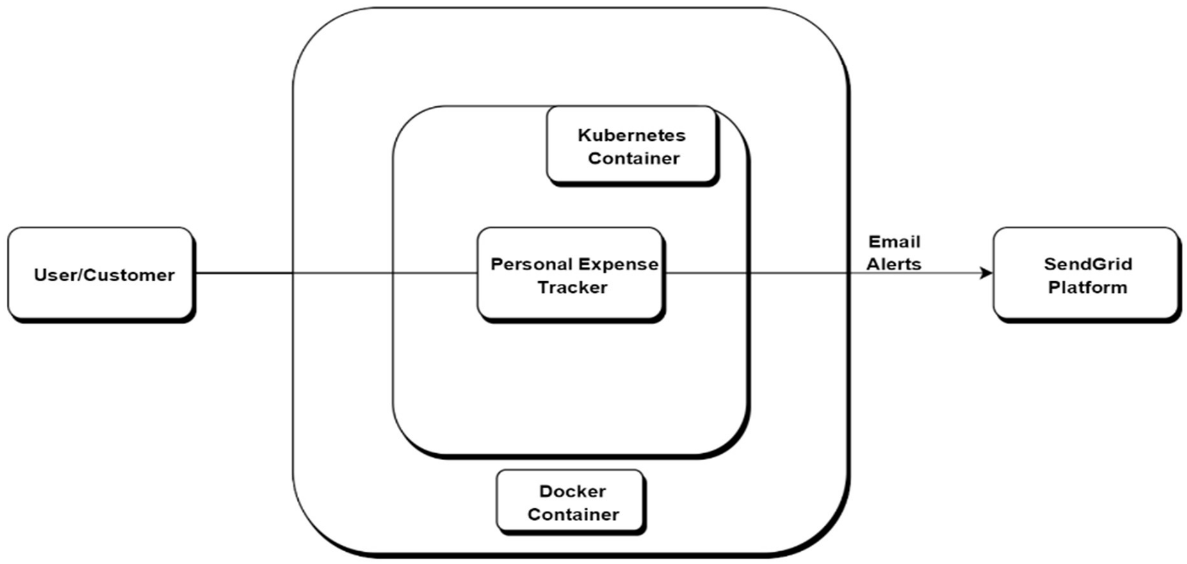
PROJECT DESIGN

5.1 Data Flow Diagram

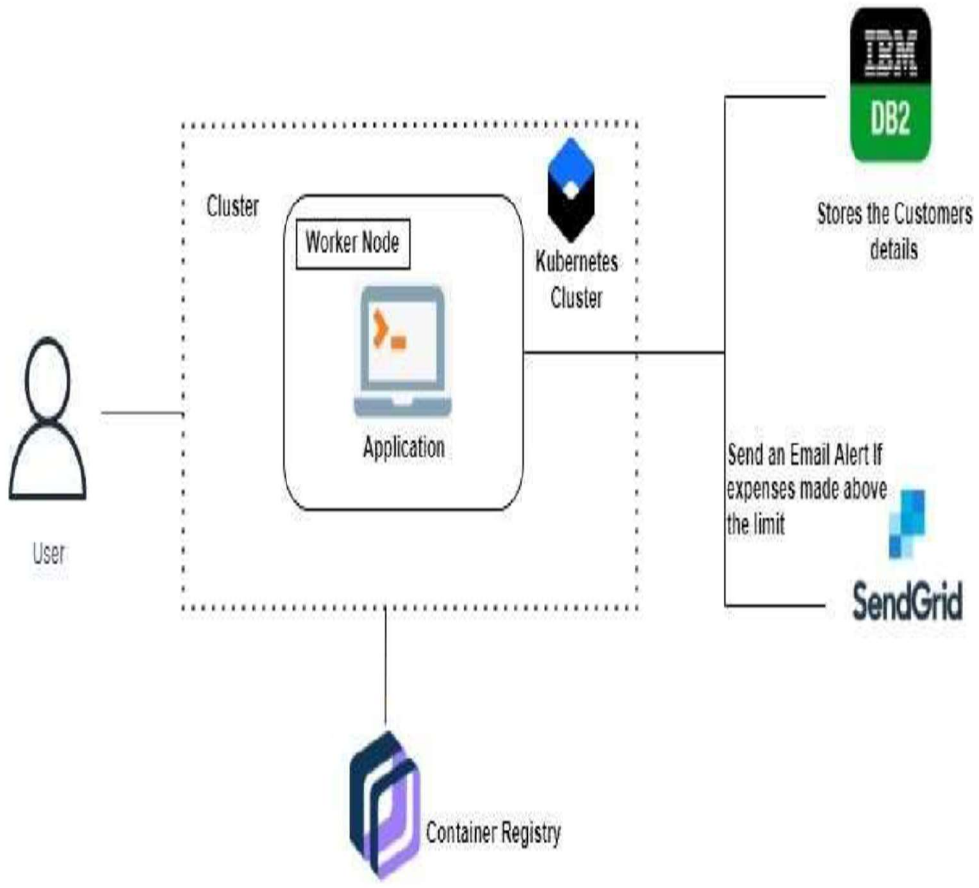


5.2 Solution & Technical Architecture

Solution Architecture



Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Login	USN-2	As a user, I can register for the application through Gmail	I can receive confirmation email & click confirm	High	Sprint-1
	Dashboard	USN-3	As a user, I can log into the application by entering email & password	I can login by giving email and password	High	Sprint-1
	Workspace	USN-1	Workspace for personal expense tracking	I can see the amount in the wallet which was added	High	Sprint-2
	Charts	USN-2	Creating various graphs and statistics of customer’s data	I can able to see the updated wallet and graph format of my monthly expenses	Medium	Sprint-2
	Connecting to IBM DB2	USN-3	Linking database with dashboard	I can get the email alert	High	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
		USN-4	Making dashboard interactive with JS	I can set the limit in the wallet	High	Sprint-2
Customer		USN-1	Wrapping up the server side works of frontend	I can access my account / dashboard	Medium	Sprint-3
	SendGrid	USN-2	Using SendGrid to send mail to the user about their expenses	I can login by giving email and password	Low	Sprint-3
		USN-3	Integrating both frontend and backend	I can see the amount in the wallet which was added	High	Sprint-3
	Docker	USN-1	Creating image of website using docker	I can able to see the updated wallet and graph format of my monthly expenses	High	Sprint-4
	Cloud Registry	USN-2	Uploading docker image to IBM Cloud registry	I can get the email alert	High	Sprint-4
	kubernetes	USN-3	Create container using the docker image and hosting the site	I can set the limit in the wallet	High	Sprint-4

CHAPTER-6
PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning and Estimation

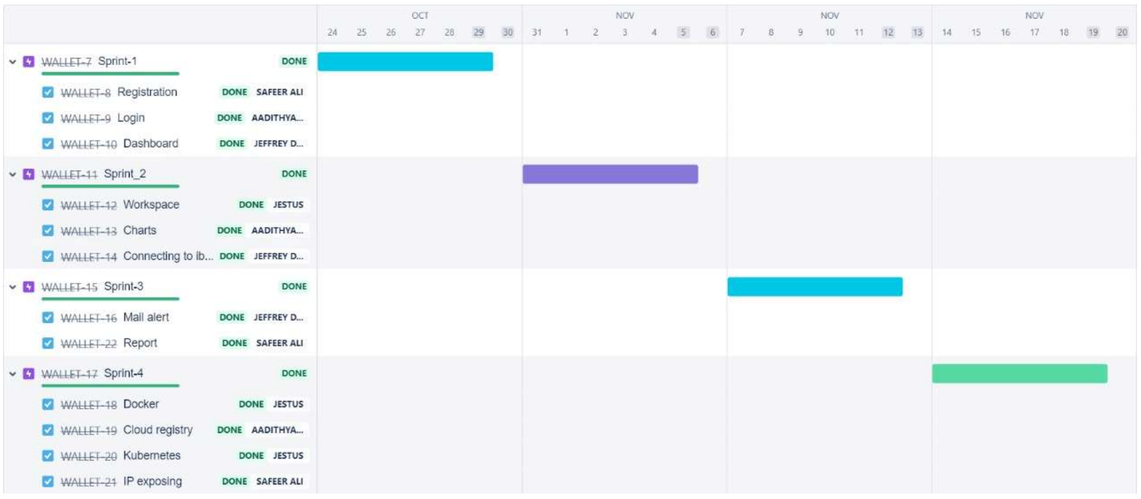
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Safeer Ali S
Sprint-1	Login	USN-2	As a user, I can register for the application through Gmail	1	High	Aadithya Prasad P
Sprint-1	Dashboard	USN-3	As a user, I can log into the application by entering email & password	2	High	Jeffrey Daniel
Sprint-2	Workspace	USN-1	Workspace for personal expense tracking	2	High	Jestus A
Sprint-2	Charts	USN-2	Creating various graphs and statistics of customer's data	1	Medium	Aadithya Prasad P
Sprint-2	Connecting to IBM DB2	USN-3	Linking database with dashboard	2	High	Jeffrey Daniel
Sprint-2		USN-4	Making dashboard interactive with JS	2	High	Jestus A
Sprint-3		USN-1	Wrapping up the server side works of frontend	1	Medium	Safeer Ali S
Sprint-3	SendGrid	USN-2	Using SendGrid to send mail to the user about their expenses	1	Low	Jeffrey Daniel
Sprint-3		USN-3	Integrating both frontend and backend	2	High	Safeer Ali S
Sprint-4	Docker	USN-1	Creating image of website using docker	2	High	Jestus A

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-4	Cloud Registry	USN-2	Uploading docker image to IBM Cloud registry	2	High	Aadithya Prasad P
Sprint-4	kubernetes	USN-3	Create container using the docker image and hosting the site	2	High	JeffreyDaniel/Jestus
Sprint-4	Exposing	USN-4	Exposing IP/Ports for the site	2	High	Aadithya Prasad P/Safeer Ali S

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

6.3 Reports From JIRA



~~WALLET-7~~	Sprint-1	DONE	
~~WALLET-8~~	Registration	DONE	SAFEER ALI
~~WALLET-9~~	Login	DONE	AADITHYA...
~~WALLET-10~~	Dashboard	DONE	JEFFREY D...
~~WALLET-11~~	Sprint_2	DONE	
~~WALLET-12~~	Workspace	DONE	JESTUS
~~WALLET-13~~	Charts	DONE	AADITHYA...
~~WALLET-14~~	Connecting to ib...	DONE	JEFFREY D...
~~WALLET-15~~	Sprint-3	DONE	
~~WALLET-16~~	Mail alert	DONE	JEFFREY D...
~~WALLET-22~~	Report	DONE	SAFEER ALI
~~WALLET-17~~	Sprint-4	DONE	
~~WALLET-18~~	Docker	DONE	JESTUS
~~WALLET-19~~	Cloud registry	DONE	AADITHYA...
~~WALLET-20~~	Kubernetes	DONE	JESTUS
~~WALLET-21~~	IP exposing	DONE	SAFEER ALI

CHAPTER-7

Features

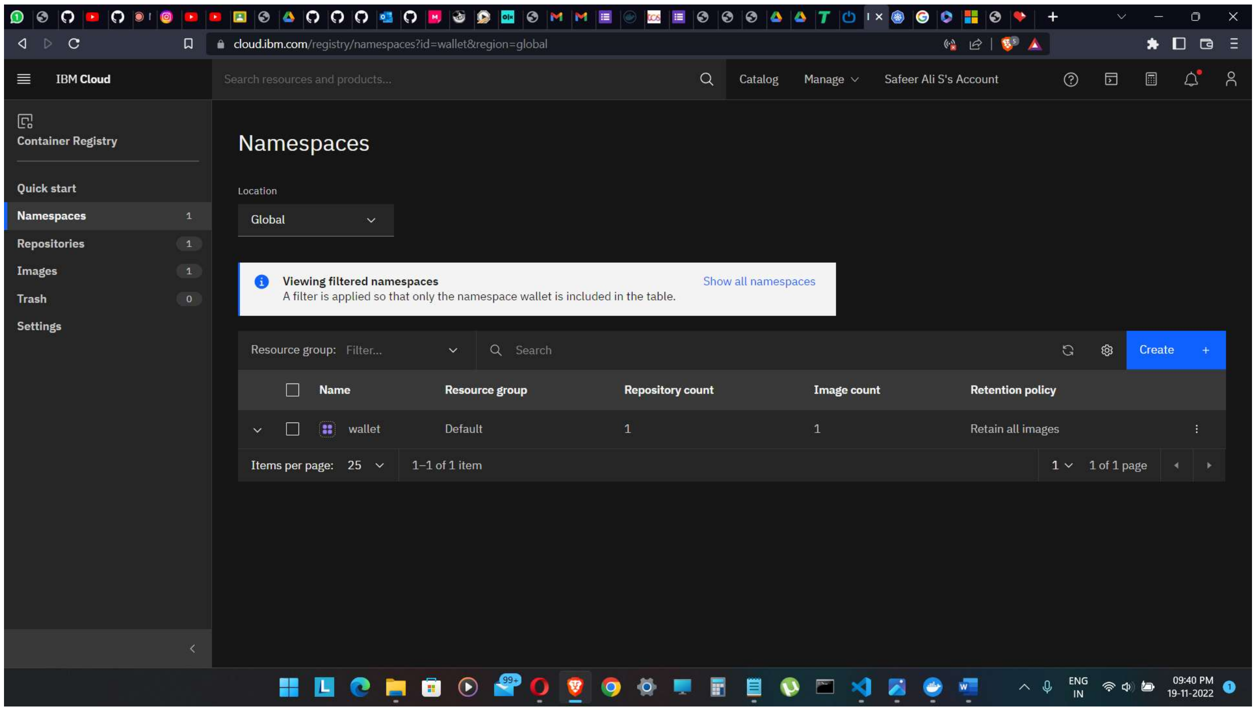
7.1.1 Python Flask

We have used python flask to develop our project. Python flask is a web framework anda python module that lets you develop web applications easily.

```
from unicodedata import name
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mail import Mail, Message
import ibm_db
import re
import json

app = Flask(__name__)
mail = Mail(app)

# configuration of mail
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = 'team.personelexpensetracker@gmail.com'
app.config['MAIL_PASSWORD'] = 'ggdiidnpqiqacncu'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)

app.secret_key = 'a'
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-4a4f-a32e-
9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;Security=S
SL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=hlv21927;PWD=42CWpSQmvebEyNj
l;",'','')

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

@app.route('/register', methods =['GET', 'POST'])
def register():
    msg = ''
    if request.method == 'POST' and 'Name' in request.form and 'Password' in
request.form and 'Email' in request.form :
        username = request.form['Name']
        password = request.form['Password']
        email = request.form['Email']
        mobileno=request.form['Mobileno']
        job=request.form['Job']
        amount=request.form['Amount']
        limit=request.form['Limit']
        stmt = ibm_db.prepare(conn,'SELECT * FROM users WHERE name = ?')
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'Username must contain only characters and numbers !'
        elif not username or not password or not email:
            msg = 'Please fill out the form !'
        else:
            prep_stmt = ibm_db.prepare(conn,"INSERT INTO users(name, email,
password, mobileno, job, amount, limit) VALUES(?, ?, ?, ?, ?, ?, ?)")
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.bind_param(prep_stmt, 4, mobileno)
            ibm_db.bind_param(prep_stmt, 5, job)
            ibm_db.bind_param(prep_stmt, 6, amount)
            ibm_db.bind_param(prep_stmt, 7, limit)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'



        return render_template('register.html', msg = msg)
    return render_template('register.html')

@app.route('/login', methods =['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST' and 'Name' in request.form and 'Password' in
request.form:
        username = request.form['Name']
        password = request.form['Password']
        stmt = ibm_db.prepare(conn,'SELECT * FROM users WHERE name = ? AND
password = ?')
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        if account:
            session['loggedin'] = True
            session['name'] = account['NAME']
            return redirect(url_for('dashboard'))
        else:
            msg = 'Incorrect username / password !'
            return render_template('login.html', a = msg)
    return render_template('login.html')


@app.route('/dashboard')
def dashboard():
    uid = session['name']
    return render_template('dashboard.html',name=uid)



@app.route('/graph')
def graph():
    if 'name' in session:
        uid = session['name']
        stmt1 = ibm_db.prepare(conn, '    SELECT EXTRACT(MONTH FROM expensedate)
as month ,sum(amount) as amount FROM expense Where uid= ? group by
month(expensedate)')
        ibm_db.bind_param(stmt1, 1, uid)
        ibm_db.execute(stmt1)
        acc1  = ibm_db.fetch_assoc(stmt1)

        data = []
        while acc1 != False:
              data.append(acc1)
              acc1 = ibm_db.fetch_assoc(stmt1)
        print(data)

        stmt2 = ibm_db.prepare(conn, ' SELECT category as cat,sum(amount) as amt
FROM expense Where uid= ? group by category order by category ')
        ibm_db.bind_param(stmt2, 1, uid)
        ibm_db.execute(stmt2)
        acc2  = ibm_db.fetch_assoc(stmt2)

        datat = []
        while acc2 != False:
              datat.append(acc2)
              acc2 = ibm_db.fetch_assoc(stmt2)
        print(datat)
        return render_template('graph.html',tdata = data,name=uid , data1 =
datat)


    render_template('graph.html')



@app.route('/report' ,methods=["POST","GET"])
def report():

       if 'name' in session:
           uid = session['name']


           stmt = ibm_db.prepare(conn, 'SELECT * FROM users WHERE name = ? ')
           ibm_db.bind_param(stmt, 1, uid)
           ibm_db.execute(stmt)
           acc = ibm_db.fetch_tuple(stmt)
           stmt1 = ibm_db.prepare(conn, 'SELECT expensedate, expensename,
category, amount FROM expense WHERE uid = ? order by expensedate desc ')
           ibm_db.bind_param(stmt1, 1, uid)
           ibm_db.execute(stmt1)
           acc1  = ibm_db.fetch_assoc(stmt1)

           data = []
           while acc1 != False:
              data.append(acc1)
              acc1 = ibm_db.fetch_assoc(stmt1)
           print(data)
           return
render_template('report.html',username=acc[1],email=acc[2],mobileno=acc[4],w=acc[
6],l=acc[7],tdata = data)
       return render_template('report.html')




@app.route('/profile',methods=["POST","GET"])
def profile():
    if 'name' in session:
        uid = session['name']
        stmt = ibm_db.prepare(conn, 'SELECT * FROM users WHERE name = ?')
        ibm_db.bind_param(stmt, 1, uid)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_tuple(stmt)
        return
render_template('profile.html',username=acc[1],email=acc[2],mobileno=acc[4],job=a
cc[5],income=acc[6],family=acc[7])
    return render_template('profile.html')

@app.route('/edit', methods =['GET', 'POST'])
def edit():

    if request.method == 'POST' :

        msg = ''
        if 'name' in session:
            uid = session['name']
            username = request.form['Name']
            email = request.form['Email']
            mobileno=request.form['Mobileno']
            job=request.form['Job']
            limit=request.form['Limit']
            prep_stmt = ibm_db.prepare(conn,"UPDATE users SET name = ?, email =
?, mobileno = ?, job = ?, limit = ? WHERE name = ?")
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, mobileno)
            ibm_db.bind_param(prep_stmt, 4, job)
            ibm_db.bind_param(prep_stmt, 5, limit)
            ibm_db.bind_param(prep_stmt, 6, uid)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully updated !'
            return render_template('edit.html', msg = msg)
    return render_template('edit.html')


@app.route('/add',methods=["POST","GET"])
def add():
    if request.method == "POST":
        if 'name' in session:
            msg1 = ''
            uid=session['name']
            paymode = request.form['paymode']
            expensename = request.form['expensename']
            expensedate = request.form['expensedate']
            category = request.form['category']
            amount = request.form['amount']
            sqlselect = "SELECT * FROM users WHERE name = ?"
            stmt = ibm_db.prepare(conn,sqlselect)
            ibm_db.bind_param(stmt, 1, uid)
            ibm_db.execute(stmt)
            sqlinsert = 'INSERT INTO expense(uid, paymode, expensename ,
expensedate, category, amount ) VALUES (?, ?, ?, ?, ?, ?)'
            prep_stmt = ibm_db.prepare(conn,sqlinsert)
            ibm_db.bind_param(prep_stmt, 1, uid)
            ibm_db.bind_param(prep_stmt, 2, paymode)
            ibm_db.bind_param(prep_stmt, 3, expensename)
            ibm_db.bind_param(prep_stmt, 4, expensedate)
            ibm_db.bind_param(prep_stmt, 5, category)
            ibm_db.bind_param(prep_stmt, 6, amount)
            ibm_db.execute(prep_stmt)
            prep_stmt1 = ibm_db.prepare(conn, "UPDATE users SET amount = amount -
? WHERE name = ?")
            ibm_db.bind_param(prep_stmt1, 1, amount)
            ibm_db.bind_param(prep_stmt1, 2, uid)
            ibm_db.execute(prep_stmt1)
            msg1 = 'You have successfully added your expense'

            sqlselect = "SELECT * FROM users WHERE name = ?"
            stmt2 = ibm_db.prepare(conn,sqlselect)
            ibm_db.bind_param(stmt2, 1, uid)
            ibm_db.execute(stmt2)
            acc = ibm_db.fetch_tuple(stmt2)

            wallet=acc[6]
            limit=acc[7]
            email=acc[2]

            wal = str(wallet)
            lim = str(limit)

            if wallet <= limit:
                 msg = Message('Hello', sender = 
'team.personelexpensetracker@gmail.com', recipients = [email])
                 msg.body = " Hello" + uid +", You exceed your wallet limit fo 
Rs. "+ lim + ".Now your Wallet Balance is Rs."+ wal +" ,Please add balance or do 
necessary action :)"
                 mail.send(msg)

        return render_template('add.html',a = msg1)
    return render_template('add.html')

@app.route('/wallet',methods=["POST","GET"])
def wallet():
    if request.method == "POST":
        if 'name' in session:
            msg = ''
            name=session['name']
            amount=request.form['Amount']
            prep_stmt = ibm_db.prepare(conn, "UPDATE users SET amount = amount + 
?  WHERE name = ?")
            ibm_db.bind_param(prep_stmt, 1, amount)
            ibm_db.bind_param(prep_stmt, 2, name)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully added your amount'

            sqlselect = "SELECT * FROM users WHERE name = ?"
            stmt2 = ibm_db.prepare(conn,sqlselect)
            ibm_db.bind_param(stmt2, 1, name)
            ibm_db.execute(stmt2)
            acc = ibm_db.fetch_tuple(stmt2)
            wallet = acc[6]
            limit = acc[7]


        return render_template('wallet.html',a = msg,s= wallet,l=limit,name =
name)
    return render_template('wallet.html')




@app.route('/help')
def help():
    uid=session['name']
    return  render_template('help.html',name=uid)





@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('Name', None)
    return redirect(url_for('login'))



if __name__ == '__main__':
    app.debug = True
    app.run(host='0.0.0.0',port=8080)
```

7.1.2 IBM Cloud Container Registry

We have deployed our app as Docker image at IBM Cloud Registry.


cloud.ibm.com/registry/namespaces?id=wallet&region=global

IBM Cloud | Search resources and products... | Catalog | Manage | Safeer Ali S's Account

Container Registry

- Quick start
- Namespaces 1
- Repositories 1
- Images 1
- Trash 0
- Settings

Namespaces

Location: Global

Viewing filtered namespaces Show all namespaces
A filter is applied so that only the namespace wallet is included in the table.

Resource group: Filter... | Search | Create +

Name	Resource group	Repository count	Image count	Retention policy
wallet	Default	1	1	Retain all images

Items per page: 25 | 1–1 of 1 item | 1 | 1 of 1 page

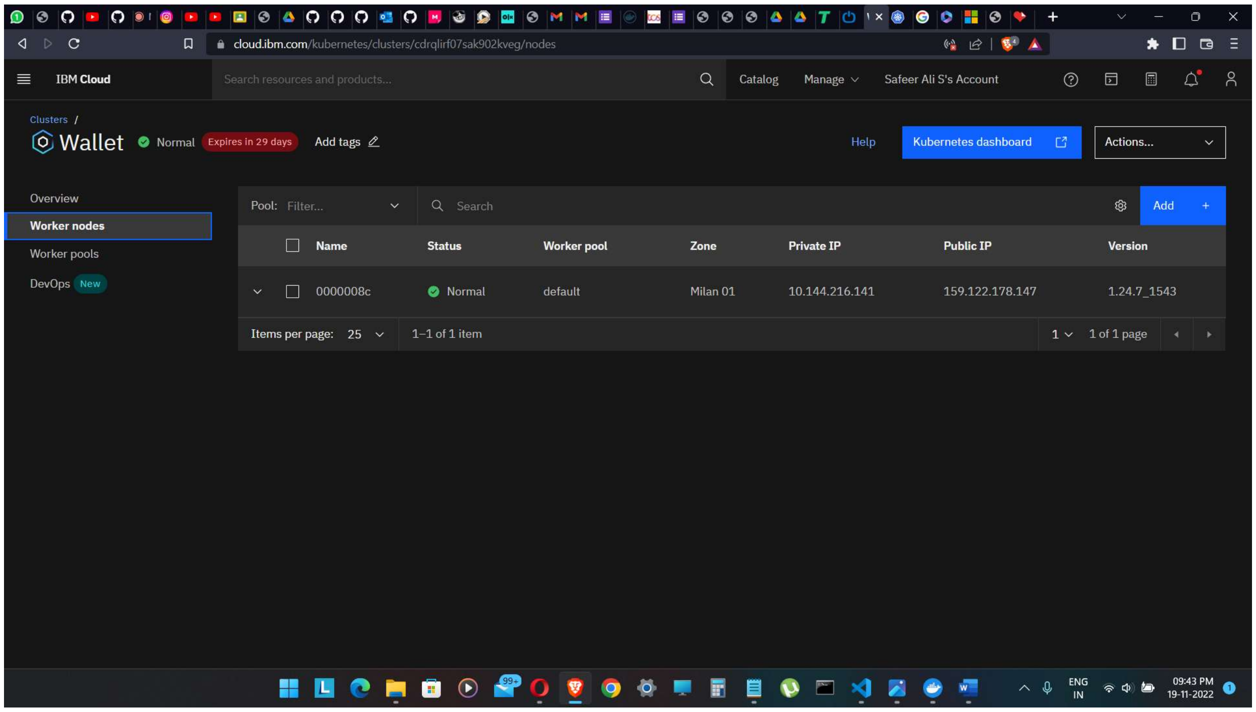
ENG IN 09:40 PM 19-11-2022

7.1.3 Kubernetes

These containers are managed by Kubernetes which automates the operational tasks ofthe container.

```yaml
apiVersion: apps/v1
 kind: Deployment
 metadata:
   name: wallet
 spec:
   replicas: 1
   selector:
     matchLabels:
       app: flasknode
   template:
     metadata:
       labels:
         app: flasknode
     spec:
       containers:
       - name: flasknode
         image: icr.io/wallet/wallet
```

```
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
```



cloud.ibm.com/kubernetes/clusters/cdrqlirf07sak902kveg/nodes

IBM Cloud | Search resources and products... | Catalog | Manage | Safeer Ali S's Account

Clusters /
Wallet Normal Expires in 29 days Add tags

Help | Kubernetes dashboard | Actions...

Overview
Worker nodes
Worker pools
DevOps New

Pool: Filter... | Search | Add

Name	Status	Worker pool	Zone	Private IP	Public IP	Version
0000008c	Normal	default	Milan 01	10.144.216.141	159.122.178.147	1.24.7_1543

Items per page: 25 | 1–1 of 1 item | 1 | 1 of 1 page

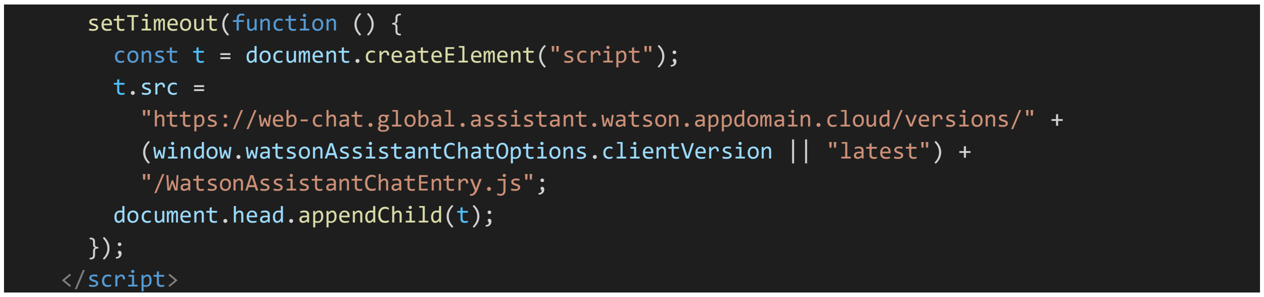
ENG IN 09:43 PM 19-11-2022

CODING & SOLUTIONING

IBM Watson Assistant Chatbot

We have integrated IBM Watson Assistant Chatbot. Here the users can know about the personal expense tracker application using the information given by the bot.

```
  <script>
      window.watsonAssistantChatOptions = {
        integrationID: "9dc4a90e-9254-4048-872a-aca2a91edf97", // The ID of this integration.
        region: "au-syd", // The region your integration is hosted in.
        serviceInstanceID: "1067f2b5-f4cd-4ced-b7ae-ab425c8ad9f0", // The ID of your service 
instance.
        onLoad: function (instance) {
          instance.render();
        },
      };
```

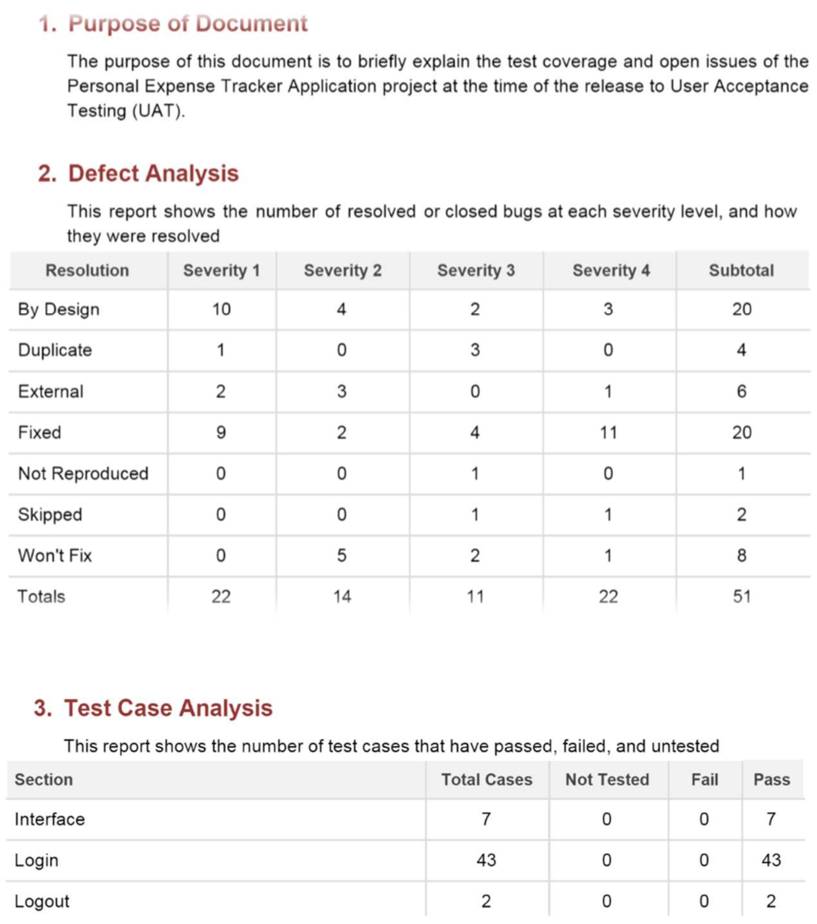


```
      setTimeout(function () {
        const t = document.createElement("script");
        t.src =
          "https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
          (window.watsonAssistantChatOptions.clientVersion || "latest") +
          "/WatsonAssistantChatEntry.js";
        document.head.appendChild(t);
      });
    </script>
```

Chapter-8

Testing:

8.1 User Acceptance Testing:



1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Personal Expense Tracker Application project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	9	2	4	11	20
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	22	14	11	22	51

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Interface	7	0	0	7
Login	43	0	0	43
Logout	2	0	0	2

Limit	3	0	0	3
Adding expenses	9	0	0	9
History	4	0	0	4
Final report	2	0	0	2

9. RESULTS:

The new system has overcome most of the limitations of the existing system and works according to the design specification given. The project what we have developed is work more efficient than the other income and expense tracker. The project successfully avoids the manual calculation for avoiding calculating the income and expense per month. The modules are developed with efficient and also in an attractive manner. The developed systems dispense the problem and meet the needs of by providing reliable and comprehensive information. All the requirements projected by the user have been met by the system. The newly developed system consumes less processing time and all the details are updated and processed immediately.

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

● User can have a control over their money and expenses.

● Users are alerted with an email when they exceed their limit.

● Reports are generated based on the users expenses.

10.2 Disadvantages

● Less Secured

● Limited Accessibility

11. CONCLUSION

Personal Expense Tracker Application is an web based application. We created this application so that a user can accurately calculate his daily cost. Using this application, the user will see the amount of his income and how much a user is spending, and a notification will be sent to the user's if he exceeds the limit and also report is generated.

12. FUTURE SCOPE

Now in our application we covered almost all features but in future we will add some more futures. The features are below

● Multiple account support.

● Include currency converter

13. APPENDIX

13.1 GitHub Link

https://github.com/IBM-EPBL/IBM-Project-30460-1660146931

13.2 Project Demo Link:

https://www.youtube.com/watch?v=mJKncb9t0ao

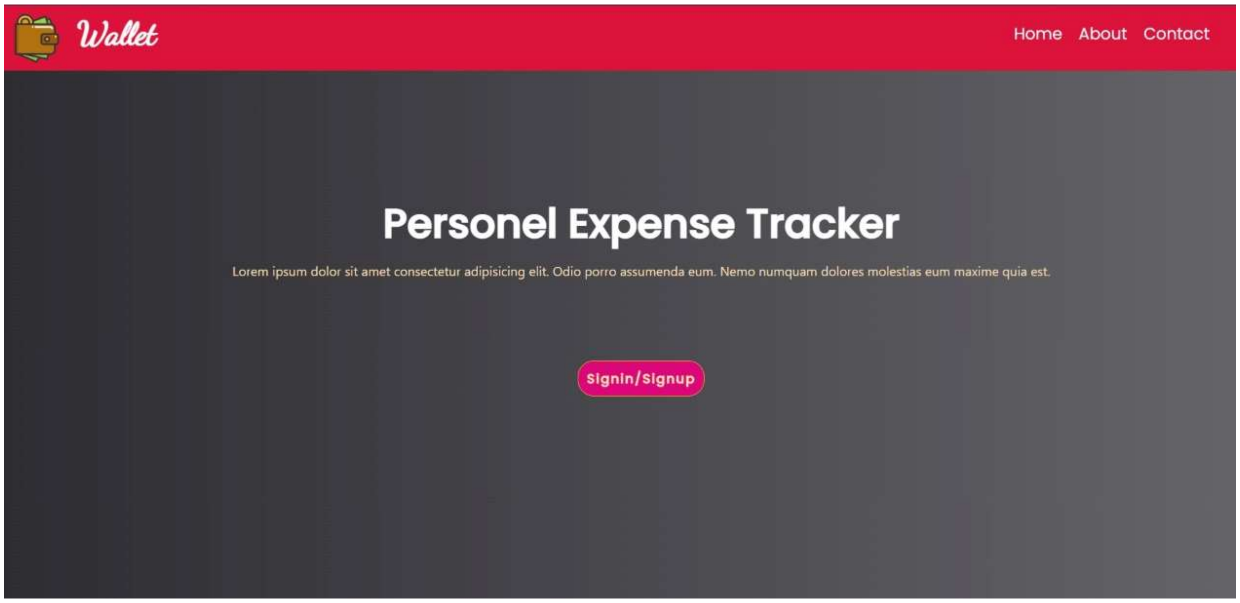
13.3 Source code:

index.html

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

```html
<head>
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <script src="/static/index.js"></script>
  <script
    src="https://kit.fontawesome.com/17041f96e9.js"
    crossorigin="anonymous"
  ></script>
  <link
    rel="shortcut icon"
    href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
    type="image/x-icon"
  />
  <link rel="stylesheet" href="/static/index.css" />
  <title>Wallet</title>
</head>
<body>
  <section id="home">
    <div class="nav" id="nav">
      <div class="logo">
        <img
          src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
          alt="Wallet png"
        />
        <h2>Wallet</h2>
      </div>
      <div class="tags">
        <a href="#home"></i>Home</a>
        <a href="#about">About</a>
        <a href="#help">Contact</a>
      </div>
    </div>
  </section>
  <section class="content" align="center" >

    <div class="c">


      <h1>Personel expense tracker</h1>
      <p>
        Lorem ipsum dolor sit amet consectetur adipisicing elit. Odio porro
        assumenda eum. Nemo numquam dolores molestias eum maxime quia est.
      </p>
      <a href="/login">Signin/Signup</a>
    </div>
  </section>
  <section class="con1" id="about">
    <h2>Features</h2>
    <div class="box">
      <div class="b1">
        <img
          src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
          alt="nothing"
        />
```

```
          <div class="m">
            <h3>Add Expense</h3>
            <br />
            <p>
              Lorem ipsum dolor sit amet consectetur adipisicing elit. Nemo vero vitae
incidunt eos voluptatum debitis et id sint harum reiciendis.


            </p>
          </div>
        </div>
        <div class="b2">
          <img
            src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
            alt="nothing"
          />
          <div class="m">
            <h3>Graph</h3>
            <br />
            <p>
             Lorem ipsum, dolor sit amet consectetur adipisicing elit. Rerum impedit,
non voluptas magni hic quo eos. Sunt accusantium alias quasi.
            </p>
          </div>
        </div>
        <div class="b1">
          <img
            src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
            alt="nothing"
          />
          <div class="m">
            <h3>Report</h3>
            <br />
            <p>
              Lorem ipsum dolor, sit amet consectetur adipisicing elit. Aut pariatur illo
neque est eius maiores itaque commodi reprehenderit, dicta suscipit.
            </p>
          </div>
        </div>
        <div class="b2">
          <img
            src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
            alt="nothing"
          />
          <div class="m">
            <h3>E-Mail</h3>
            <br />
            <p>
              Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab explicabo
ipsum enim, repellendus mollitia eaque! Beatae quibusdam molestiae perspiciatis
sed!
            </p>
          </div>
        </div>
      </div>
    </section>
    <footer>
      <center>
      <div class="foot" id="help">
        <div class="logo" >
          <img src="https://cdn-icons-png.flaticon.com/512/218/218390.png" alt="">
          <h1>Wallet</h1>
        </div>
        <div class="contact" align="center">
          <h3>Contact us</h3>


        </div>
          <div class="link">
           <a href="https://github.com/IBM-EPBL/IBM-Project-30460-1660146931">
<i class="fab fa-github"></i></a>
           <a href = "mailto:team.personelexpensetracker@gmail.com?subject =
Feedback & body = Message"><i class="fas fa-envelope"></i></a>
           <a href="https://www.linkedin.com/in/jeffrey-daniel-120801252"><i
class="fab fa-linkedin-in"></i></a>
          </div>


        <h3 style="width:100%;bottom: 0;">--  Copyright © NILL --</h3>


      </div>
    </center>
    </footer>
  </body>
</html>
```



Personel Expense Tracker

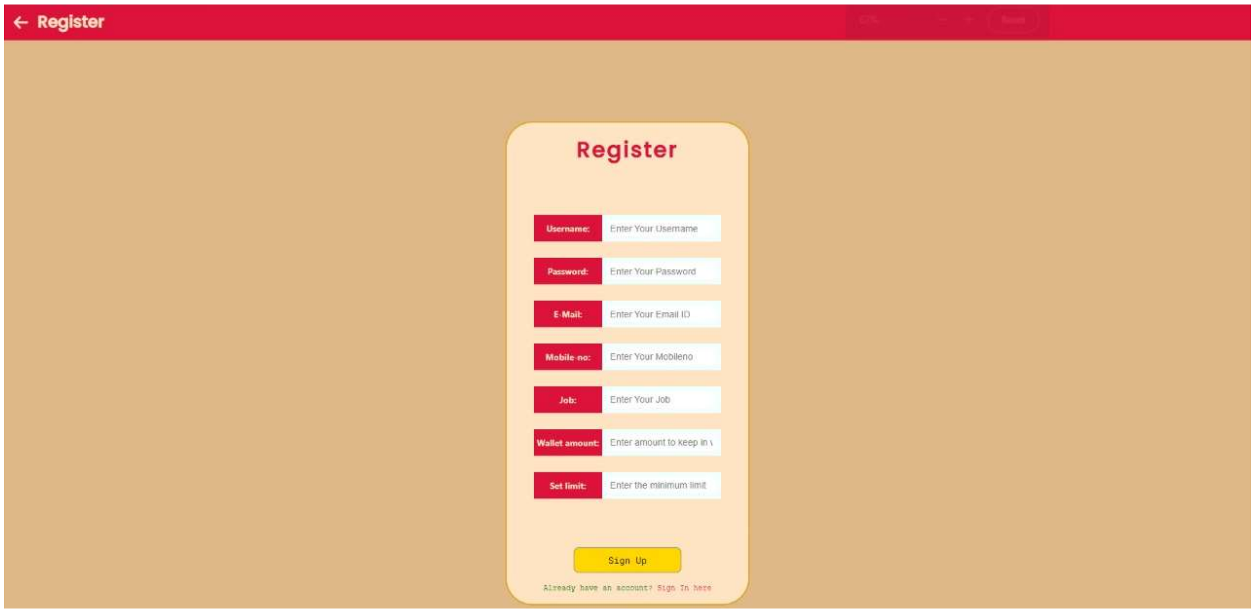
Lorem ipsum dolor sit amet consectetur adipisicing elit. Odio porro assumenda eum. Nemo numquam dolores molestias eum maxime quia est.

register.html

```
<html>
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <script
      src="https://kit.fontawesome.com/17041f96e9.js"
      crossorigin="anonymous"
    ></script>
    <link
      rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
      type="image/x-icon"
```

```
 />
 <link rel="stylesheet" href="/static/style.css" />
 <title>Register</title>
</head>
<body>
 <div class="nav">
  <div class="logo">
   <a href="/login"> <i class="fas fa-arrow-left"></i></a>
   <h3>Register</h3>
  </div>
  <!-- <div class="name">
   <h3>Name</h3>
  </div> -->
 </div>
 <center>
  <div align="center" class="border">
   <div class="header">
    <h1>Register</h1>
   </div>

   <form action="/register" method="post">
    <h3 class="msg">{{msg}}</h3>
    <table>
     <tr>
      <th>Username:</th>
      <td>
       <input
      id="Name"
      name="Name"
      type="text"
      placeholder="Enter Your Username"
      class="textbox"
    />
  </td>
</tr>
<tr>
  <th>Password:</th>
  <td>
    <input
      id="Password"
      name="Password"
      type="password"
      placeholder="Enter Your Password"
      class="textbox"
    />
  </td>
</tr>
<tr>
  <th>E-Mail:</th>
  <td>
    <input
      id="Email"
      name="Email"
      type="text"
      placeholder="Enter Your Email ID"
      class="textbox"
    />
  </td>
</tr>
<tr>
  <th>Mobile-no:</th>
  <td>
    <input
      id="Mobileno"
      name="Mobileno"
      type="text"
      placeholder="Enter Your Mobileno"
      class="textbox"
    />
  </td>
</tr>
<tr>
  <th>Job:</th>
  <td>
    <input
      id="Job"
      name="Job"
      type="text"
      placeholder="Enter Your Job"
      class="textbox"
    />
    </td>
  </tr>
  <tr>
    <th>Wallet amount:</th>
    <td>
      <input
        id="Amount"
        name="Amount"
        type="text"
        placeholder="Enter amount to keep in wallet"
        class="textbox"
      />
    </td>
  </tr>
  <tr>
    <th>Set limit:</th>
    <td>
      <input
        id="Limit"
        name="Limit"
        type="text"
        placeholder="Enter the minimum limit"
        class="textbox"
      />
    </td>
  </tr>
</table>
```

```
          <input type="submit" class="btn" value="Sign Up" />
        </form>

        <p class="bottom">
          Already have an account?
          <a class="bottom" href="{{url_for('login')}}"> Sign In here</a>
        </p>
      </div>
    </center>
  </body>
</html>
```



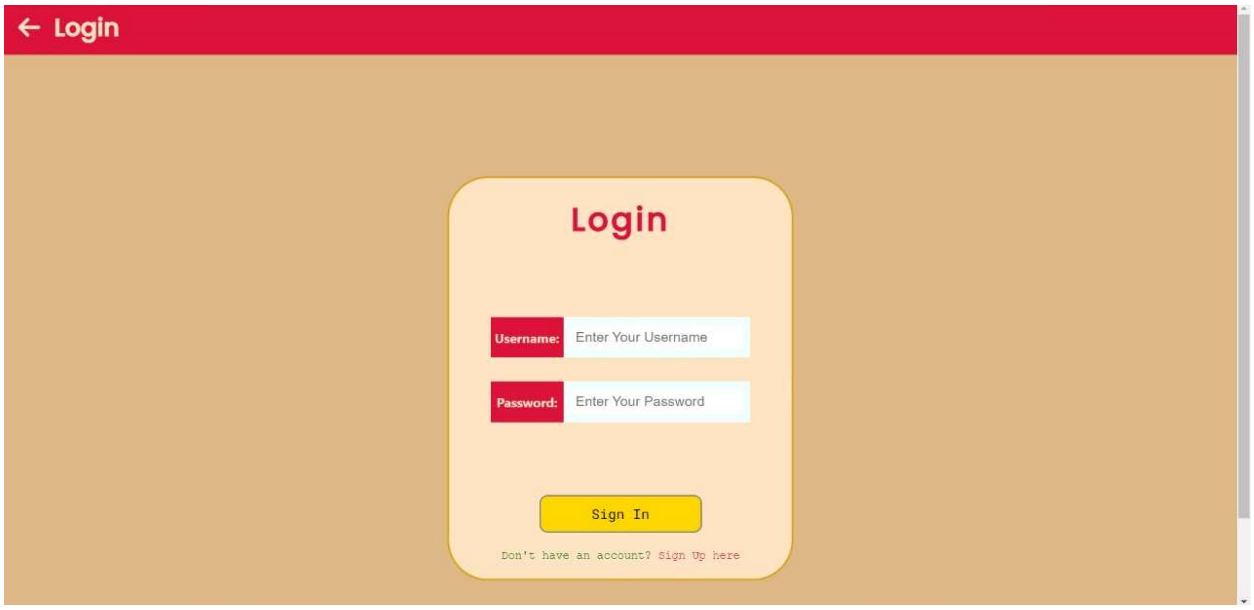
login.html

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

```
<head>
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <script
    src="https://kit.fontawesome.com/17041f96e9.js"
    crossorigin="anonymous"
  ></script>
  <link
    rel="shortcut icon"
    href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
    type="image/x-icon"
  />
  <link rel="stylesheet" href="/static/style.css" />
  <title>Login</title>
</head>
<body>
  <div class="nav">
    <div class="logo">
      <a href="/index"> <i class="fas fa-arrow-left"></i></a>
      <h3>Login</h3>
    </div>
    <!-- <div class="name">
      <h3>Name</h3>
    </div> -->
  </div>
```

```
<center>
 <div align="center" class="border">
   <div class="header">
     <h1 class="word">Login</h1>
   </div>
   <form action="/login" method="post">
     <h3 class="msg">{{a}}</h3>
     <table>
       <tr>
         <th>Username:</th>
         <td>
           <input
             id="Name"
             name="Name"
             type="text"
             placeholder="Enter Your Username"
             class="textbox"
           />
         </td>
       </tr>

       <tr>
         <th>Password:</th>
         <td>
           <input
             id="Password"
             name="Password"
```

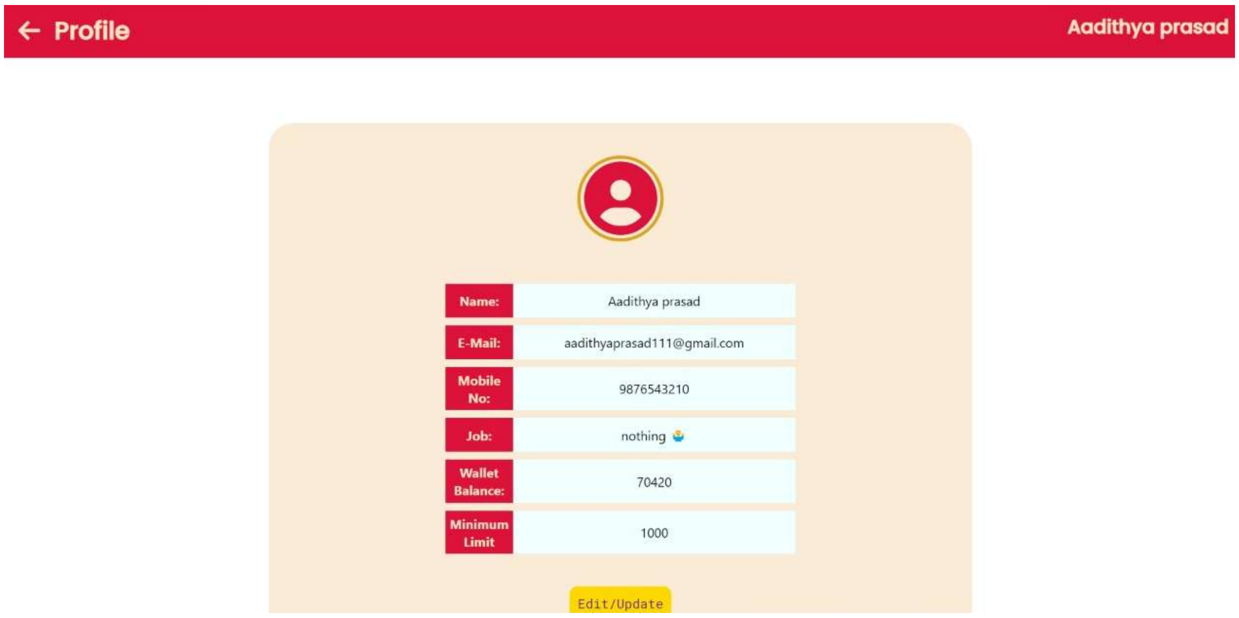
```
            type="password"
            placeholder="Enter Your Password"
            class="textbox"
           />
          </td>
         </tr>
        </table>

        <input type="submit" class="btn" value="Sign In" />
       </form>
       <p class="bottom">
        Don't have an account?
        <a class="bottom" href="/register"> Sign Up here</a>
       </p>
      </div>
     </center>
    </body>
   </html>
```



userprofile.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="/static/profile.css" />
    <script
      src="https://kit.fontawesome.com/17041f96e9.js"
      crossorigin="anonymous"
    ></script>
    <link
      rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
      type="image/x-icon"
    />
    <title>Profile</title>
  </head>
  <body>
    <div class="nav">
      <div class="logo">
        <a href="/dashboard"> <i class="fas fa-arrow-left"></i></a>
        <h3>Profile</h3>
      </div>
      <div class="name">
        <h3>{{username}}</h3>
      </div>
    </div>
    <center>
      <div class="cont" align="center">
        <div class="logo">
          <i class="fas fa-user-circle"></i>
        </div>
        <div class="profile">
          <table>
            <tr>
              <th>Name:</th>
              <td>{{username}}</td>
            </tr>
            <tr>
              <th>E-Mail:</th>
```

```
              <td>{{email}}</td>
            </tr>
            <tr>
              <th>Mobile No:</th>
              <td>{{mobileno}}</td>
            </tr>
            <tr>
              <th>Job:</th>
              <td>{{job}}</td>
            </tr>
            <tr>
              <th>Wallet Balance:</th>
              <td>{{income}}</td>
            </tr>
            <tr>
              <th>Minimum Limit</th>
              <td>{{family}}</td>
            </tr>
          </table>
          <a href="/edit">Edit/Update</a>
        </div>
      </div>
    </center>
  </body>
</html>
```



Name:	Aadithya prasad
E-Mail:	aadithyaprasad111@gmail.com
Mobile No:	9876543210
Job:	nothing
Wallet Balance:	70420
Minimum Limit	1000

graph.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script
      src="https://kit.fontawesome.com/17041f96e9.js"
      crossorigin="anonymous"
    ></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.js"></script>
    <link
      rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
      type="image/x-icon"
    />
    <link rel="stylesheet" href="/static/graph.css" />
    <title>Report</title>
  </head>
  <body>
    <div class="nav">
      <div class="logo">
        <a href="/dashboard"> <i class="fas fa-arrow-left"></i></a>
        <h3>Expense Graph</h3>
      </div>
      <div class="name">
        <h3>{{name}}</h3>
      </div>
    </div>
    <div class="con">
      <div class="chart">
        <h3 style="font-family: Poppins;">Monthly Report</h3>
        <div class="box" align="center">
            <canvas id="myChart" style="width:100%;max-width:700px"></canvas>
        </div>
      </div>
      <div class="chart">
        <h3 style="font-family: Poppins;">Category Report</h3>
        <div class="box" align="center">
            <canvas id="myChart-1" style="width:100%;max-
width:600px"></canvas>
```

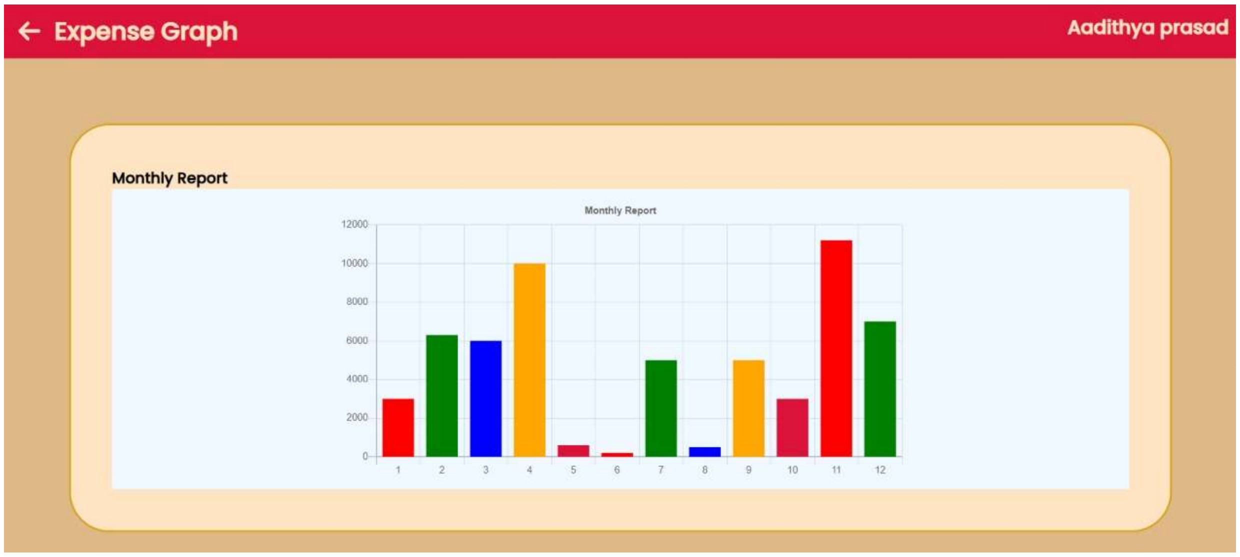
```
      </div>
    </div>
    <!-- <div class="chart">
        <h3 style="font-family: Poppins;">Daily Report</h3>
        <div class="box" align="center">
            <canvas id="myChart-2" style="width:100%;max-
width:600px"></canvas>
      </div>
    </div> -->




    <!-- script for bar chart -->


    <script>



      var data = JSON .parse('{{ tdata | tojson | safe}}');


       console.log(data);
       var mon = [];
       var vals = [];


      // val = data[1].MONTH;


      //   console.log(val);
```

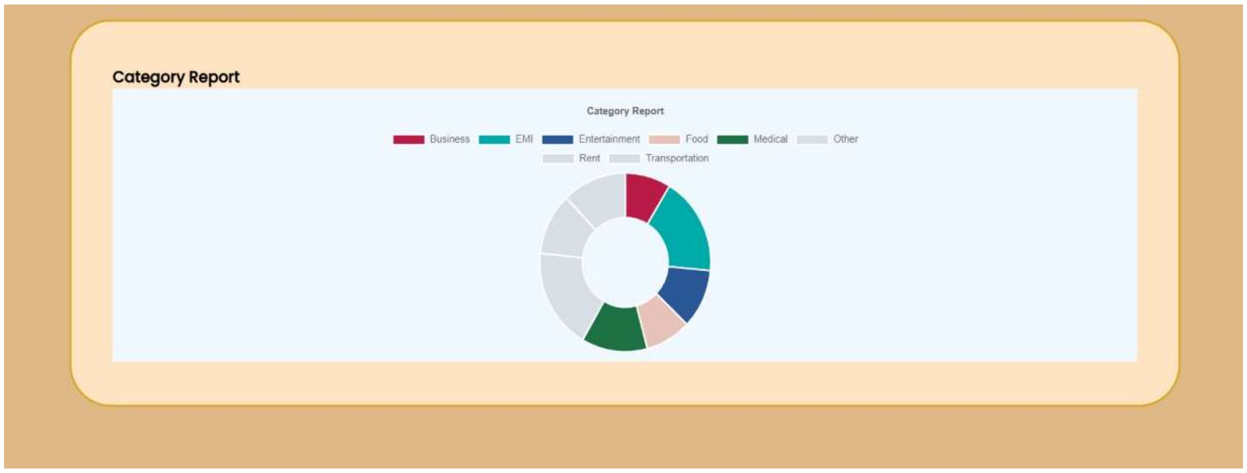
```
     for(var i in data){
       mon.push(data[i].MONTH)
     }


     for(var i in data){
               vals.push(data[i].AMOUNT);
     }


       console.log(vals);
       console.log(mon);




       var barColors = ["red", "green","blue","orange","crimson","red",
"green","blue","orange","crimson","red", "green","blue","orange","brown","red",
"green","blue","orange","crimson","red", "green","blue","orange","crimson","red",
"green","blue","orange","brown","red", "green","blue","orange","crimson","red",
"green","blue","orange","crimson","red", "green","blue","orange","brown","red",
"green","blue","orange","crimson","red", "green","blue","orange","crimson","red",
"green","blue","orange","brown"];


       new Chart("myChart", {
         type: "bar",
         data: {
           labels: mon,
           datasets: [{
             backgroundColor: barColors,
             data: vals
```

```
            }]
          },
          options: {
            legend: {display: false},
            title: {
              display: true,
              text: "Monthly Report"
            }
          }
        });
        </script>

<!-- script for doughnut chart  -->
<script>

    var datat = JSON .parse('{{ data1 | tojson | safe}}');



    var xValues = [];
    var yValues = [];

    for(var i in datat){
        xValues.push(datat[i].CAT)
      }

      for(var i in datat){
              yValues.push(datat[i].AMT);
```

```
  }

var barColors = [
  "#b91d47",
  "#00aba9",
  "#2b5797",
  "#e8c3b9",
  "#1e7145"
];

new Chart("myChart-1", {
  type: "doughnut",
  data: {
    labels: xValues,
    datasets: [{
      backgroundColor: barColors,
      data: yValues
    }]
  },
  options: {
    title: {
      display: true,
      text: "Category Report"
    }
  }
});
</script>
```

```
<!-- script for graph  -->

<script>
  var xValues = [50,60,70,80,90,100,110,120,130,140,150];
  var yValues = [7,8,8,9,9,9,10,11,14,14,15];

  new Chart("myChart-2", {
    type: "line",
    data: {
      labels: xValues,
      datasets: [{
        fill: false,
        lineTension: 0,
        backgroundColor: "rgba(0,0,255,1.0)",
        borderColor: "rgba(0,0,255,0.1)",
        data: yValues
      }]
    },
    options: {
      legend: {display: false},
      scales: {
        yAxes: [{ticks: {min: 6, max:16}}],
      }
    }
  });
  </script>
  </body>
</html>
```


Expense Graph
Aadithya prasad
Monthly Report
Monthly Report
12000
10000
8000
6000
4000
2000
0
1
2
3
4
5
6
7
8
9
10
11
12


Category Report
Category Report
Business
EMI
Entertainment
Food
Medical
Other
Rent
Transportation

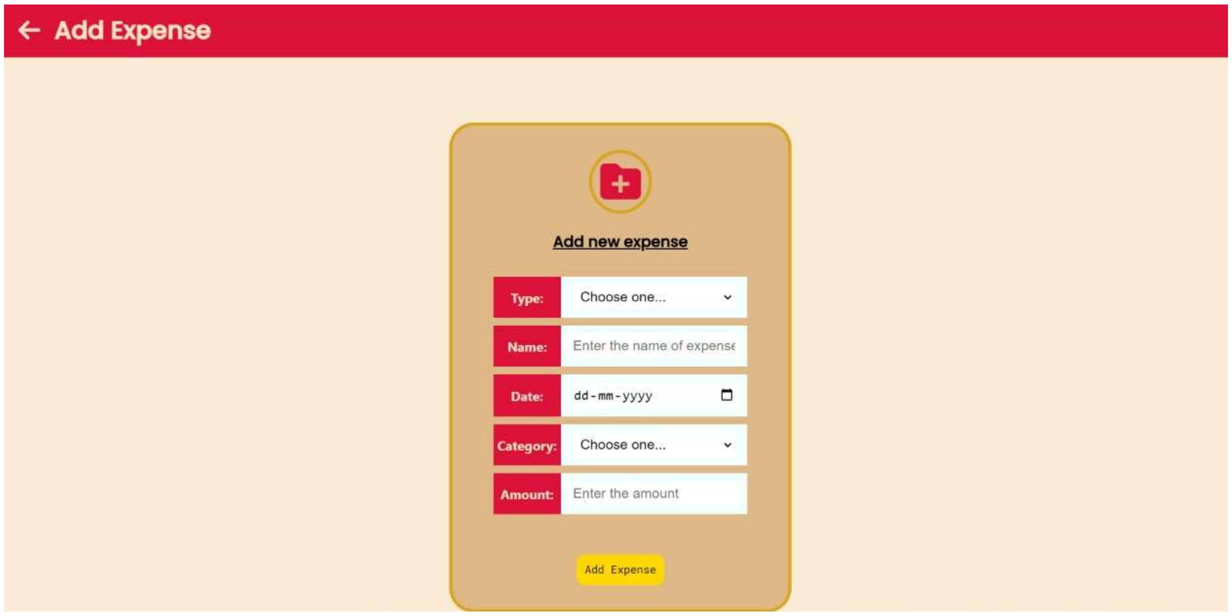
add.html

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="/static/addexpense.css">
  <script
  src="https://kit.fontawesome.com/17041f96e9.js"
  crossorigin="anonymous"
 ></script>
 <link
  rel="shortcut icon"
  href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
  type="image/x-icon"
 />
  <title>Add Expense</title>
</head>
<body>
  <div class="nav">
    <div class="logo">
      <a href="/dashboard">  <i class="fas fa-arrow-left"></i></a>
        <h3>Add Expense</h3>
    </div>
    <form action="/add" method="post">

    <div class="name">
```

```
          <h3>{{ username}}</h3>
        </div>
    </div>
    <center>
    <div class="cont" align="center">
      <div class="logo">
        <i class="fas fa-folder-plus"></i>
      </div>
      <div class="profile">
        <h3 style="font-family: Poppins;text-decoration: underline;">Add new
expense</h3>
        <h3 style="font-family: Poppins;">{{a}}</h3>
        <table>
          <tr>
            <th>Type:</th>
            <td>  <select id="paymode" name="paymode">
                <option value="chooseOne">Choose one...</option>
                <option value="Card">Card</option>
                <option value="Cash">Cash</option>
                <option value="Wallet">Wallet</option>
                <option value="Other">Other</option>
            </select></td>
          </tr>
          <tr>
            <th>Name:</th>
            <td><input type="text" name="expensename" id="expensename"
placeholder="Enter the name of expense"></td>
```

```
          </tr>
          <tr>
            <th>Date:</th>
            <td><input type="date" id="expensedate" name="expensedate"
placeholder="date"></td>
          </tr>
          <tr>
            <th>Category:</th>
            <td><select id="type" name="category" id="category"
onchange="yesnoCheck(this);">
                <option id="noCheck" value="ChooseOne">Choose one...</option>
                <option id="noCheck" value="Food">food</option>
                <option id="noCheck" value="Entertainment">entertainment</option>
                <option id="noCheck" value="Business">business</option>
                <option id="noCheck" value="Rent">rent</option>
                <option id="noCheck" value="EMI">EMI</option>
                <option id="noCheck" value="Transportation">Transportation</option>
                <option id="noCheck" value="Medical">Medical</option>
                <option id="noCheck" value="Other" id="yesCheck">Other</option>
            </select></td>
          </tr>
          <tr id="ifYes" style="display: none">
            <th>Category:</th>
            <td>
              <input
                type="text"
                placeholder="Enter Category.."
                id="category"
                name="category"
              />
            </td>
          </tr>
          <tr>
            <th>Amount:</th>
            <td><input type="number" name="amount" id="amount"
placeholder="Enter the amount"></td>
          </tr>

         </table>
         <!-- <span class="alert {{indicator}}">{{a}}</span> -->
         <button type="submit" ><a >Add Expense</a></button>
       </form>

     </div>
   </div>
</center>

     <!-- script for input  -->
     <script>
       function yesnoCheck(that) {
         if (that.value == "Other") {
          document.getElementById("ifYes").style.display = "table-row";
         } else {
          document.getElementById("ifYes").style.display = "none";
         }
       }
     </script>

</body>
</html>
```

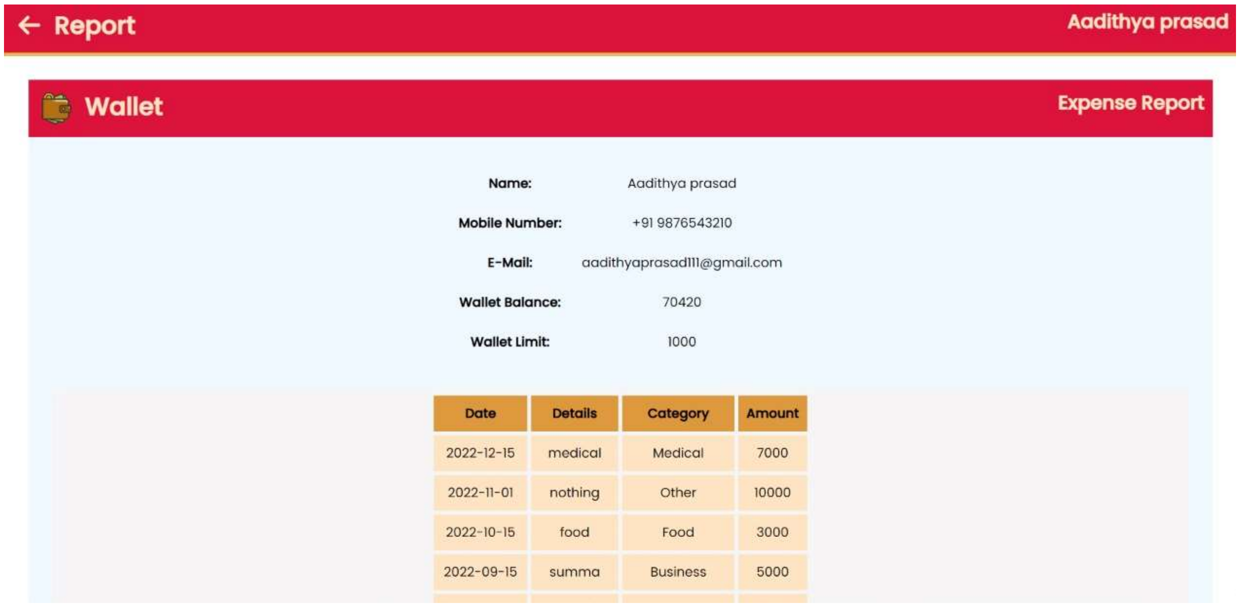

Add Expense
Add new expense
Type:
Choose one...
Name:
Enter the name of expense
Date:
dd-mm-yyyy
Category:
Choose one...
Amount:
Enter the amount
Add Expense

report.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script
      src="https://kit.fontawesome.com/17041f96e9.js"
      crossorigin="anonymous"
    ></script>
    <link
      rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
      type="image/x-icon"
    />
    <link rel="stylesheet" href="/static/report.css" />
    <title>Report</title>
  </head>
  <body>
    <header class="noprint">
      <div class="nav">
        <div class="logo">
          <a href="/dashboard"> <i class="fas fa-arrow-left"></i></a>
          <h3>Report</h3>
        </div>
        <div class="name">
          <h3>{{username}}</h3>
        </div>
      </div>
    </header>
    <!-- <div class="content , noprint" align="center">
      <h3>Get Report:</h3>
      <form method="post" action="/report">
        <table>
          <tr>
            <th>From :</th>
            <td><input type="date" name="from" id="from" /></td>
          </tr>
          <tr>
            <th>Untill :</th>
            <td><input type="date" name="to" id="to" value="" /></td>
          </tr>
          <tr>
            <td>
              <button onclick="OnCheck()" ondblclick="submit" id="btnCheck">
                Get details
              </button>
            </td>
            <td>
              <button onclick="window.print()" id="btnCheck">Get Pdf</button>
            </td>
          </tr>
        </table>
      </form> -->
      <div style="border: 2px solid goldenrod"></div>
    </div>

    <div class="details">
      <div class="nav">
        <div class="logo">
          <img
            src="https://cdn-icons-png.flaticon.com/512/218/218390.png"
            alt=""
          />
          <h3>Wallet</h3>
        </div>
        <div class="name">
          <h3>Expense Report</h3>
        </div>
      </div>
      <table>
        <tr>
          <th>Name:</th>
          <td>{{username}}</td>
        </tr>
        <tr>
          <th>Mobile Number:</th>
          <td>+91 {{mobileno}}</td>
        </tr>
        <tr>
          <th>E-Mail:</th>
          <td>{{email}}</td>
        </tr>
        <tr>
          <th>Wallet Balance:</th>
          <td>{{w}}</td>
        </tr>
        <tr>
          <th>Wallet Limit:</th>
          <td>{{l}}</td>
        </tr>
        <!-- <tr>
          <th>Report Untill:</th>
          <td>{{t}}</td>
        </tr> -->
      </table>

      <table class="report">
        <tr>
          <th>Date</th>
          <th>Details</th>
          <th>Category</th>
          <th>Amount</th>
        </tr>
        {% for val in tdata %}
        <tr>
          <td>{{val['EXPENSEDATE']}}</td>
          <td>{{val['EXPENSENAME']}}</td>
          <td>{{val['CATEGORY']}}</td>
          <td>{{val['AMOUNT']}}</td>
        </tr>
        {% endfor %}
      </table>
```

```
  </div>

  <!-- script for date checking  -->

  <script>
   function OnCheck() {
    var d_Date = document.getElementById("txtDate").value.split("-");

    var txtDate = new Date(d_Date[2], d_Date[1] - 1, d_Date[0]).setHours(
      0,
      0,
      0,
      0
    );

    var currentDate = new Date().setHours(0, 0, 0, 0);

    if (txtDate - currentDate == 0) {
      alert("Equal to Current Date");
    } else if (txtDate - currentDate > 0) {
      alert("greater than the current date");
    } else if (txtDate - currentDate < 0) {
      alert("Less then the current Date");
    }
   }
  </script>
</body>
```

```
</html>
```



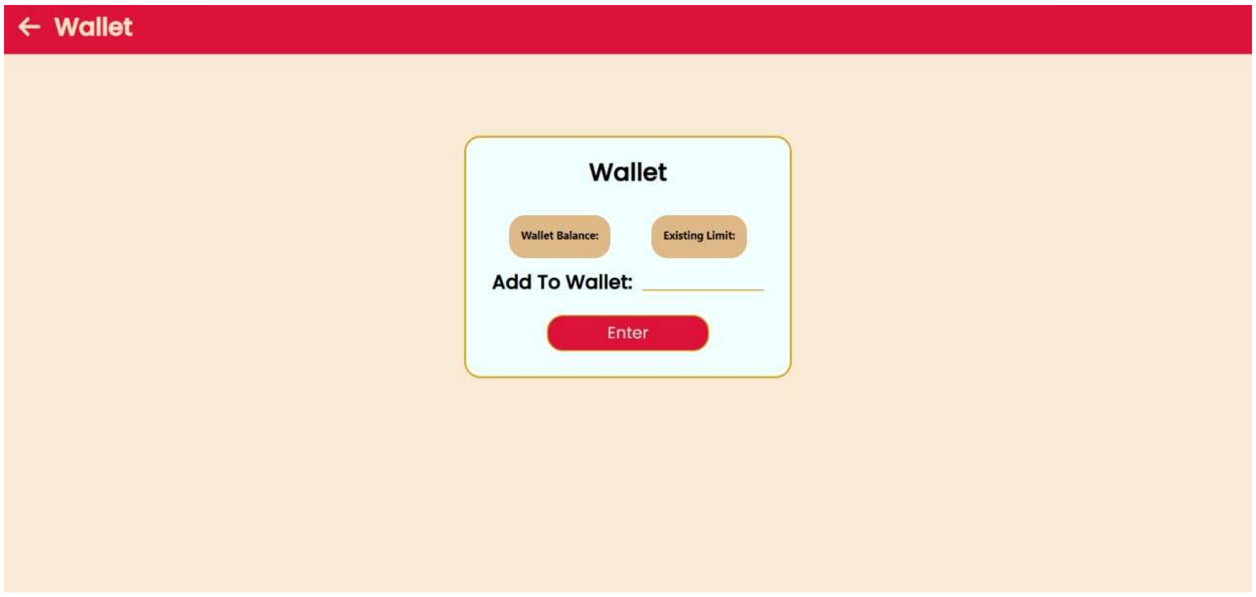
Name:	Aadithya prasad
Mobile Number:	+91 9876543210
E-Mail:	aadithyaprasad111@gmail.com
Wallet Balance:	70420
Wallet Limit:	1000

Date	Details	Category	Amount
2022-12-15	medical	Medical	7000
2022-11-01	nothing	Other	10000
2022-10-15	food	Food	3000
2022-09-15	summa	Business	5000

Wallet.html:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <script
      src="https://kit.fontawesome.com/17041f96e9.js"
      crossorigin="anonymous"
    ></script>
    <link
      rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/218/218390.png"
      type="image/x-icon"
    />
    <title>Wallet</title>
    <link rel="stylesheet" href="/static/wallet.css" />
  </head>
  <body>
    <div class="nav">
      <div class="logo">
        <a href="/dashboard"> <i class="fas fa-arrow-left"></i></a>
        <h3>Wallet</h3>
      </div>
      <div class="name">
        <h3>{{name}}</h3>
      </div>
    </div>
    <center>
      <div align="center" class="mail">
        <h3>Wallet</h3>
        <p>{{a}}</p>
        <div class="box">
          <div class="b1">
            <h5>Wallet Balance:</h5>
            <h6>{{s}}</h6>
          </div>
          <div class="b1">
```

```
          <h5>Existing Limit:</h5>
          <h6>{{l}}</h6>
        </div>
      </div>
      <form action="/wallet" method="post">
        <table>
          <tr>
            <td><h2>Add To Wallet:</h2></td>
            <td><input type="text" name="Amount" id="Amount" /></td>
          </tr>
        </table>
        <button type="submit"><a>Enter</a></button>
      </form>
    </div>
  </center>
 </body>
</html>
```


← Wallet
Wallet
Wallet Balance:
Existing Limit:
Add To Wallet:
Enter