PERSONAL EXPENSE TRACKER APPLICATION

TEAM ID: PNT2022TMID13651

TEAM LEADER:

GOPIKA R (621319205009)

TEAM MEMBERS:

POOJA S S (621319205031)

RAGAVI S (621319205037)

VIJAYA BHARATHI L (621319205056)

TEAM MENTOR:

MR.R.PALANIKUMAR M.E.,

TABLE OF CONTENTS

S.NO	TITLE	PAGE NO
1	**INTRODUCTION**	
	1.1 Project Overview	4

CHAPTER 1
INTRODUCTION

The administration of one's own finances plays a significant role in daily life. But not everyone has the time or the knowledge to manage their funds properly. And even if someone has the time and knowledge to manage their costs, they choose not to because they find it tiresome and time-consuming. You can now actively manage your finances by using an expense tracker, so you don't have to worry about keeping track of your spending.

1.1 PROJECT OVERVIEW

Due to the weak economy and the plethora of spending temptations, every economic piece and player needs to develop their own particular money management abilities. Nevertheless, managing money can occasionally be challenging because financial information can take many different forms (bills, statements, invoices, pay stubs, receipts, to name a few).As a result, it is frequently impossible for consumers to even locate any of these financial records while looking for them, let alone maintain and monitor their own cash flow.

The movement of money is only noteworthy when it is coming in and is frequently forgotten when it is going out because people frequently underestimate how much money they spend on unplanned expenses and purchases from a mental budget. The Daily Expense Tracker System is made to keep track of a user's daily income and expenses. This method distributes the funds according to daily costs. If you exceed your daily spending allowance, the system will deduct it from your earnings and provide you a new allowance. If the cost for that day is less, the system will save it. The income-expenditure curve will be shown in the monthly report generated by the daily spending tracking system.

1.2 PURPOSE

Tracking your expense and expenditure is very important. Spending tracking can be done manually by writing down all purchases. Or use various apps available in this digital age. The very important reason you need to track your spending is to identify and eliminate wasteful spending habits in your financial life. Plus, tracking your spending consistently helps you manage your finances and promote better financial habits likeSaving and Investing. Ideally, it's a significant activity that should be done times each day throughout the month. Tracking your spending makes you more aware of what you're spending and where you're spending it. Tracking your spending is the only way to know how much you spend on unnecessary things and spendings. It is very useful to know the spendings you make in your daily life.

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING SYSTEM

In existing system, need to keep the user's daily and monthly spending in Excel sheets, CSV, etc. There isn't yet a way to quickly keep track of everyday expenses using Accumulate. This requires keeping note of, either in a diary or online.Additionally, since all computations must be done manually, there is a chance for errors and consequent losses. There may be a variety of disadvantages to using a manual accounting system. The accounting procedure in any business can be difficult. You might not need to understand the accounting process as thoroughly as you would with a manual accounting system if you use a computerized accounting system. Depending on who is doing the bookkeeping, this may be advantageous or disadvantageous. It is frequently necessary to hire a competent professional to ensure that accounting is performed appropriately. It may take some time to manually filter through your financial details due to their complexity. Consequently, creating reports requires more time.

2.2 REFERENCES

[1]Dr.V.Geetha, G.Nikhitha, H.SrLasya,Dr.C.K.Gomathy,**EXPENDITURE MANAGEMENT SYSTEM, 2022.**

[2] Dr. V. Geetha, G. Nikhitha, H. Sri Lasya Dr. C.K.Gomathy,**EXPENDITURE MANAGEMENT SYSTEM,2022.**

[3] R. Radhika, Anagha Praveen, G. Gokul Krishna, Adithya Anand,and T. Anjali**,FINANCE TRACKER,2022.**

[4]Saumya Dubey, Pragya Dubey, Rigved Rishabh Kumar, AaishaKhatoon**,STUDENT EXPENSE TRACKING APPLICATION,2022.**

[5]Nidhi Jitendra Jadhav, Rutuja Vijay Chakor, Trupti Mahesh Gunjal, Damayanti. D. Pawar**,EXPENSE TRACKER,2022.**

[6]Aman Garg, Mukul Goel, Sagar Mittal, Mr. Shekhar Singh**,EXPENSE TRACKER,2021.**

[7] Muskaan Sharma, Ayush Bansal,Dr.RajuRanjan,Shivam Sethi,**A NOVEL EXPENSE TRACKER USING STATISTICAL ANALYSIS,2021.**

[8]Velmurugan A, Albert Mayan, Niranjana P and Richard Francis ,**PERSONAL EXPENSE TRACKER,2021.**

[9]Muskaan Sharma, Ayush Bansal, Dr. Raju Ranjan, Shivam Sethi **,A NOVEL EXPENSE TRACKER USING STATISTICAL ANALYSIS,2021.**

[10] Prof Miriam Thomas, Lekshmi and Dr. Mahalekshmi**,EXPENSE TRACKER,2020.**

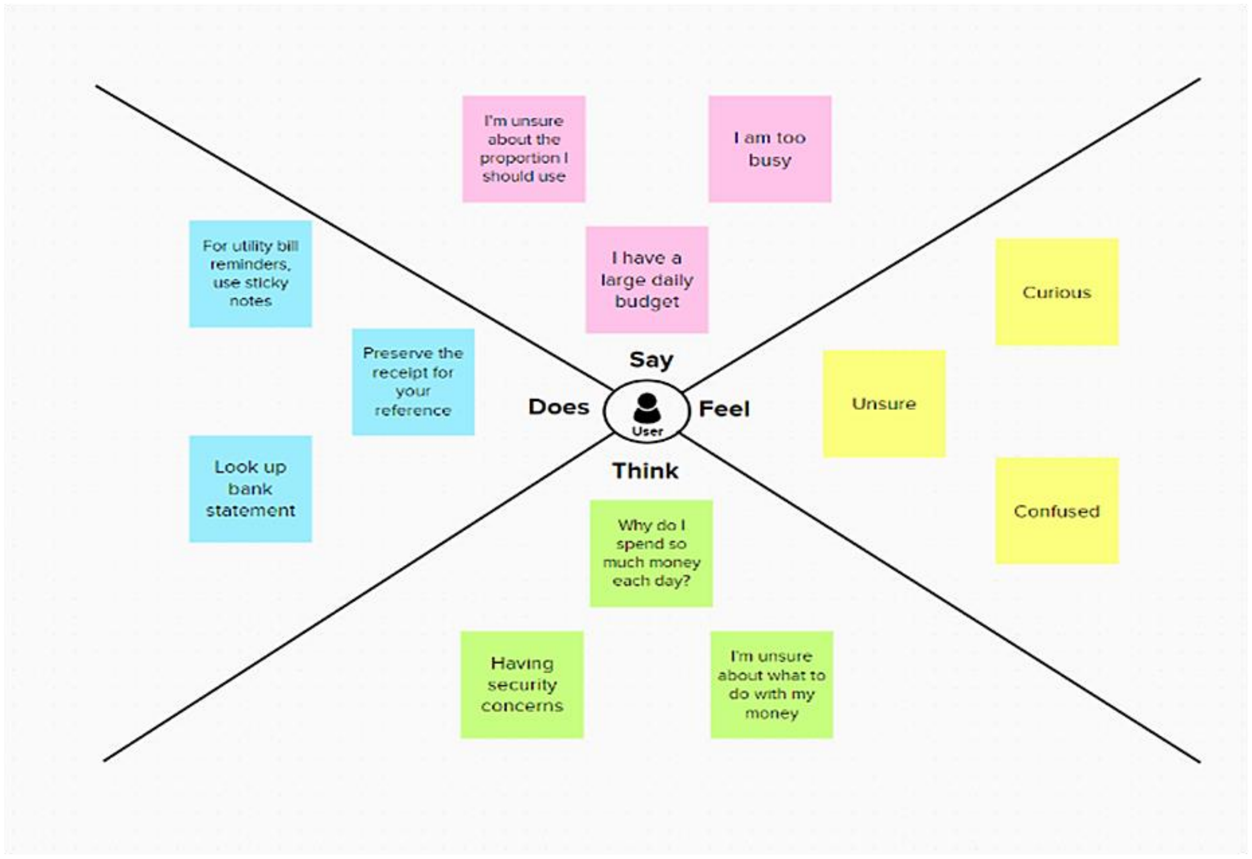
2.3 PROBLEM STATEMENT DEFINITION

In existing system, need to maintain the Excel sheets, CSV etc. files for the user daily and monthly expenses There isn't an Accumulate to quickly keep track of daily costs yet. One needs to keep a track for this, either in a diary or on a computer. Additionally, since all computations must be done manually, there is a chance for errors and consequent losses. There may be a variety of disadvantages to using a manual accounting system. The accounting procedure in any business can be difficult. You might not need to understand the accounting process as thoroughly as you would with a manual accounting system if you use a computerised accounting system.User has to update their daily Expense details and get a graphical pie-chart per month with the details . Users can retain and monitor daily spending with the usage of this programme as opposed to keeping track of them on a paper sheet.Calculating the monthly cost specifics requires more time in a manual procedure.

CHAPTER 3
IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

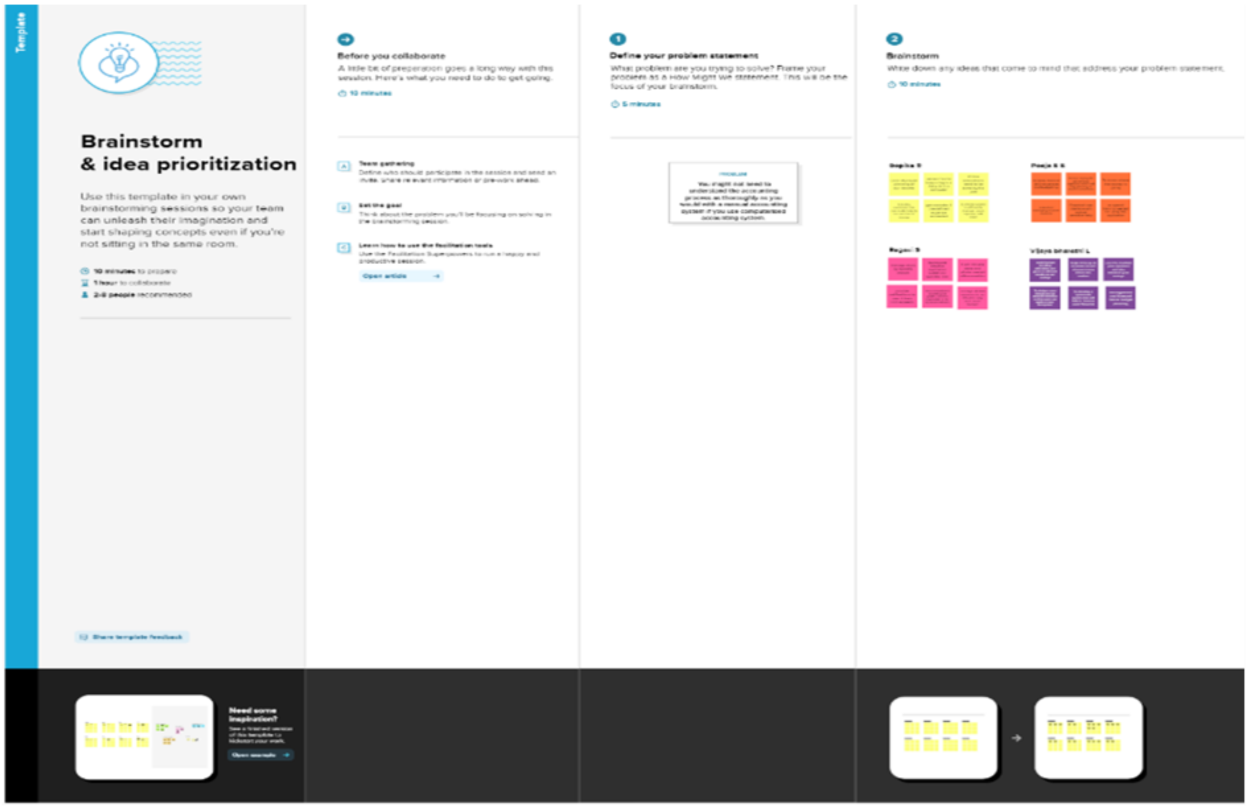
Teams can utilise an empathy map as a collaborative tool to learn more about their clients. An empathy map can depict a group of users, such as a consumer segment, in a manner similar to user personas. The agile community has embraced the empathy map, which was first developed by Dave Gray.


I'm unsure about the proportion I should use
I am too busy
I have a large daily budget
Say
For utility bill reminders, use sticky notes
Preserve the receipt for your reference
Look up bank statement
Does
User
Feel
Curious
Unsure
Confused
Think
Why do I spend so much money each day?
Having security concerns
I'm unsure about what to do with my money

1

3.2 IDEATION & BRAINSTORMING

Ideation and the practise of brainstorming, a particular method for coming up with fresh ideas, are frequently closely related. The main distinction between ideation and brainstorming is that whereas brainstorming is nearly often done .



Template

Brainstorm & idea prioritization

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

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

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

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

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

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

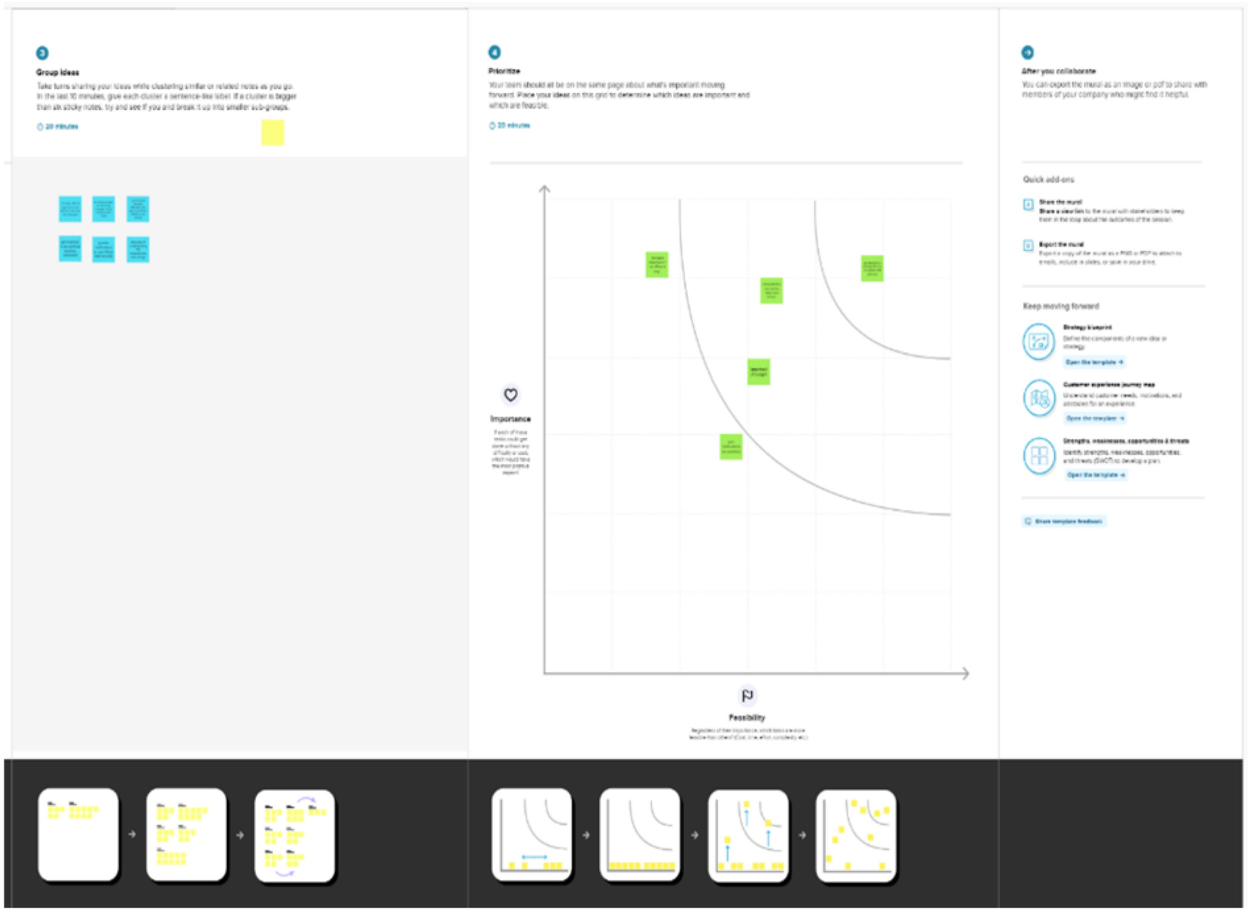
PROBLEM
You might not need to understand the accounting process as thoroughly as you would with a manual accounting system if you use computerised accounting system.

2 Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes

[illegible]

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example

1



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

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

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

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

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

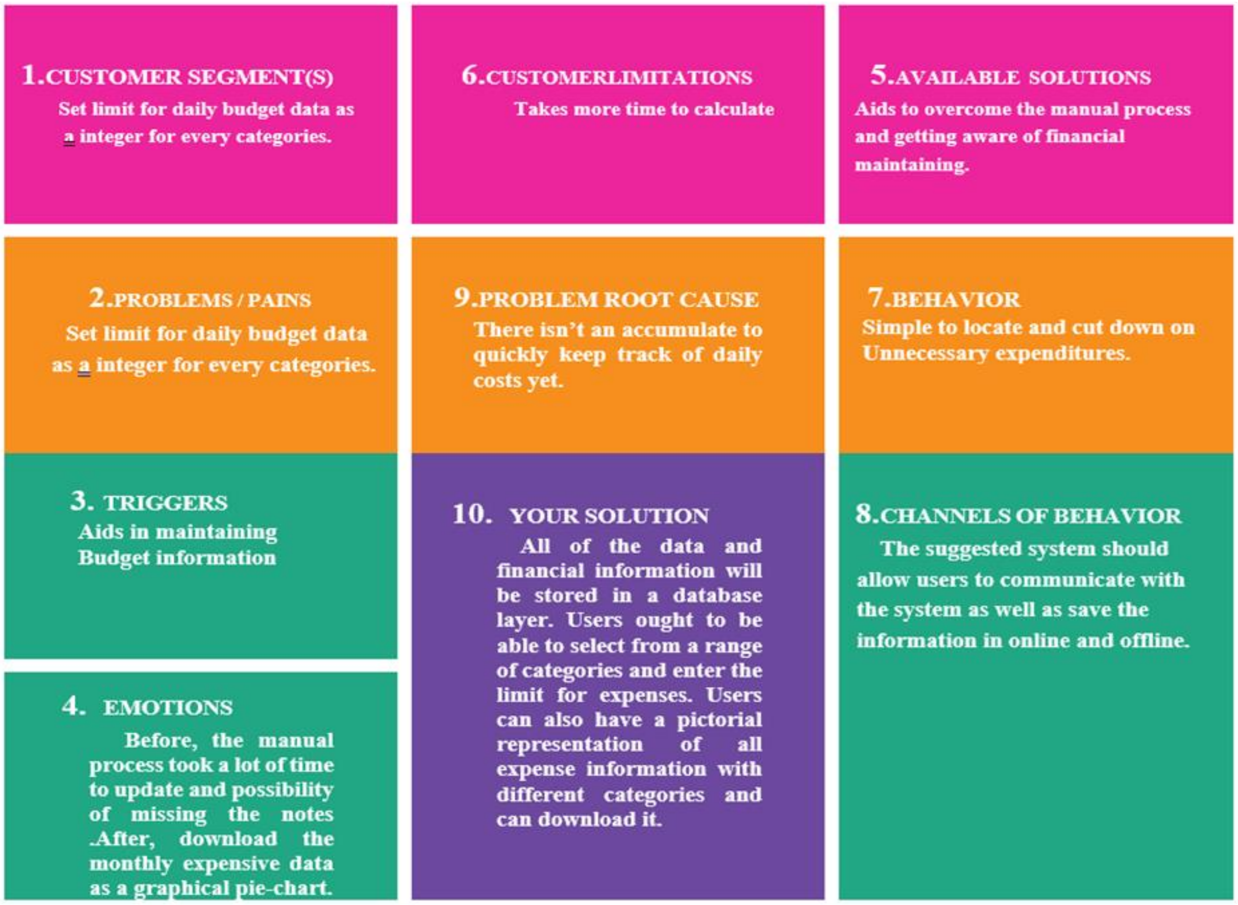
3.3 PROPOSED SOLUTION

Proposed Solution refers to the technical response that the Implementation agency will offer in response to the Project's requirements and objectives. Proposed Solution refers to the Proposed System modified to satisfy the Agency's requirements as outlined in this RFP.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	One must perform all calculations themselves, which can occasionally result in mistakes that result in losses
2.	Idea / Solution description	Goal of assisting this application will track all of the user’s daily and monthly expenses.
3.	Novelty / Uniqueness	Provide graphical pie-chart of the users’ Expenses
4.	Social Impact / Customer Satisfaction	It provides a graphical pie-chart that makes it simple to understand the pricey details. This application makes the financial situation more apparent.
5.	Business Model (Revenue Model)	Whether you're an entrepreneur or a small business owner, aexpense tracker can help to simplify your finances. These apps can help you keep track of receipts, organize all of your costs into categories and even integrate with popular accounting software.
6.	Scalability of the Solution	Ability to track the Budget Details

3.4 PROBLEM SOLUTION FIT

The problem-solution approach Fit simply means that the problem you've identified with your consumer is solved by the solution you've created.



1.CUSTOMER SEGMENT(S)
Set limit for daily budget data as a integer for every categories.

6.CUSTOMERLIMITATIONS
Takes more time to calculate

5.AVAILABLE SOLUTIONS
Aids to overcome the manual process and getting aware of financial maintaining.

2.PROBLEMS / PAINS
Set limit for daily budget data as a integer for every categories.

9.PROBLEM ROOT CAUSE
There isn't an accumulate to quickly keep track of daily costs yet.

7.BEHAVIOR
Simple to locate and cut down on Unnecessary expenditures.

3. TRIGGERS
Aids in maintaining Budget information

10. YOUR SOLUTION
All of the data and financial information will be stored in a database layer. Users ought to be able to select from a range of categories and enter the limit for expenses. Users can also have a pictorial representation of all expense information with different categories and can download it.

8.CHANNELS OF BEHAVIOR
The suggested system should allow users to communicate with the system as well as save the information in online and offline.

4. EMOTIONS
Before, the manual process took a lot of time to update and possibility of missing the notes .After, download the monthly expensive data as a graphical pie-chart.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Functional requirements specify what a system should be able to do through computations, technical details, data manipulation and processing, and other specialized functions. The use cases that are used by the system to implement the functional requirements are reflected in the behavioral requirements. The suggested solution's functional requirements are listed below.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	Register	Registration is the process of the user to complete the application's form. Certain details must be submitted such as e-mail address, password, and password confirmation. The user is identified using these details.
FR-2	Login	The login screen is used to confirm the user's identification. The account can be accessed using the user's registered email address and password.
FR-3	Categories	On the main page, we can see overall revenue and spending, as well as the balance remaining after expenditure, as well as the user's entire categories namely Entertainment, Cloth, Food and Drinks, Health and Fitness and so on.
FR-4	Update Daily Expensive	The user can upload the daily expensive details what they are spending on each day. The details such as cloth, entertainment, food, health etc.,
FR-5	View Expensive Chart	This module used to see a pictorial depiction of all details in a form of a pie chart, where each slice of the pie chart represents that the viewer to gain an approximate notion of which category has the highest expenses.

FR-6	Set Alert	When a user attempts to spend more than the pre-defined amount limit, the app will automatically send an alert if the threshold amount they selected for an alert is exceeded.
FR-7	Notification	A feature of the application called push notifications lets the administrator or developer send a personalized tip of money management to all of the app's users.

4.2 NON-FUNCTIONAL REQUIREMENT

A non-functional requirement (NFR) is a requirement that, rather of defining specific behaviours, specifies criteria that can be used to assess how well a system performs. Functional requirements, on the other hand, define particular behaviours or functions. The system design includes a thorough plan for putting functional requirements into practise. Because they are typically architecturally significant requirements, the system architecture includes a thorough plan for implementing non-functional requirements. The non-functional requirements for the suggested solution are listed below.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The system must be accessible to users through a web application on a computer.The system uses a web application as an interface. The system is user friendly which makes the system easy.
NFR-2	**Security**	A security requirement is a declaration of essential security functionality that guarantees the fulfilment of one of numerous security attributes of software.
NFR-3	**Reliability**	The system has to be 100% reliable due to the importance of data and the damages that can caused by incorrect or incomplete data. The system will run 7 days a week. 24 hours a day.
NFR-4	**Performance**	Depending on whether or not the application has undergone any upgrades, the data is updated. Within two seconds of the request submission, the system must react to the member. When processing massive amounts of data, the system should be given more leeway. Responses to information requests must display on screen within 5 seconds.
NFR-5	**Availability**	The system is used around-the-clock, 365 days a year, and is completely accessible to the user. The system must function seven days a week, twenty-four hours a day.
NFR-6	**Scalability**	Scalability is a term used to describe a system's capacity to adjust its performance and cost in response to shifting application and system processing requirements.

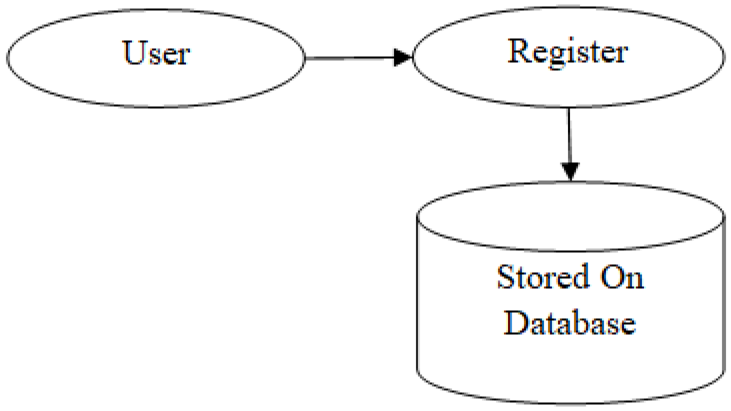
CHAPTER 5

PROJECT DESIGN

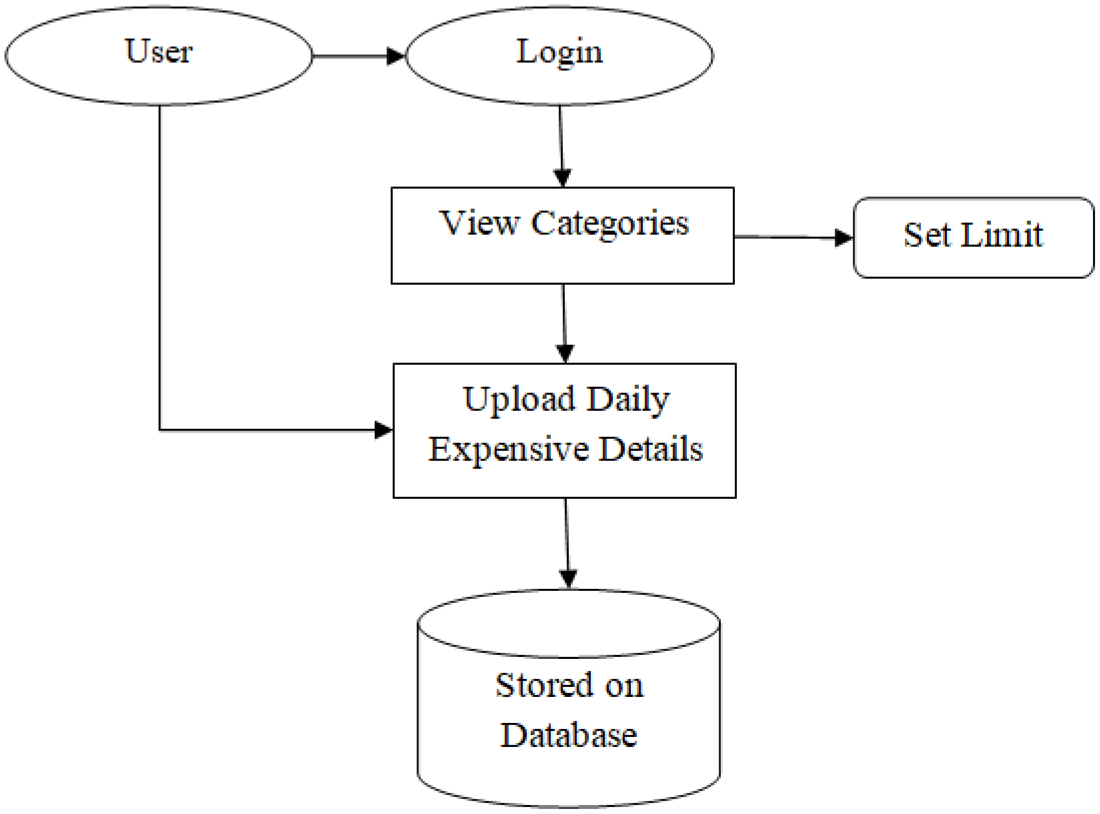
5.1 DATA FLOW DIAGRAM

A data flow diagram demonstrates the movement of data within a procedure or system. This covers data input/output, data storage, and the numerous data-moving supporting operations. Standardized symbols and notations are used to generate DFDs in order to characterize diverse entities and their relationships.

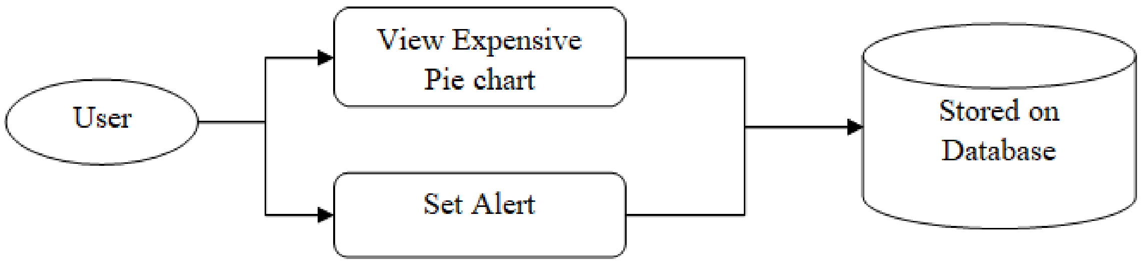
LEVEL 0



LEVEL 1



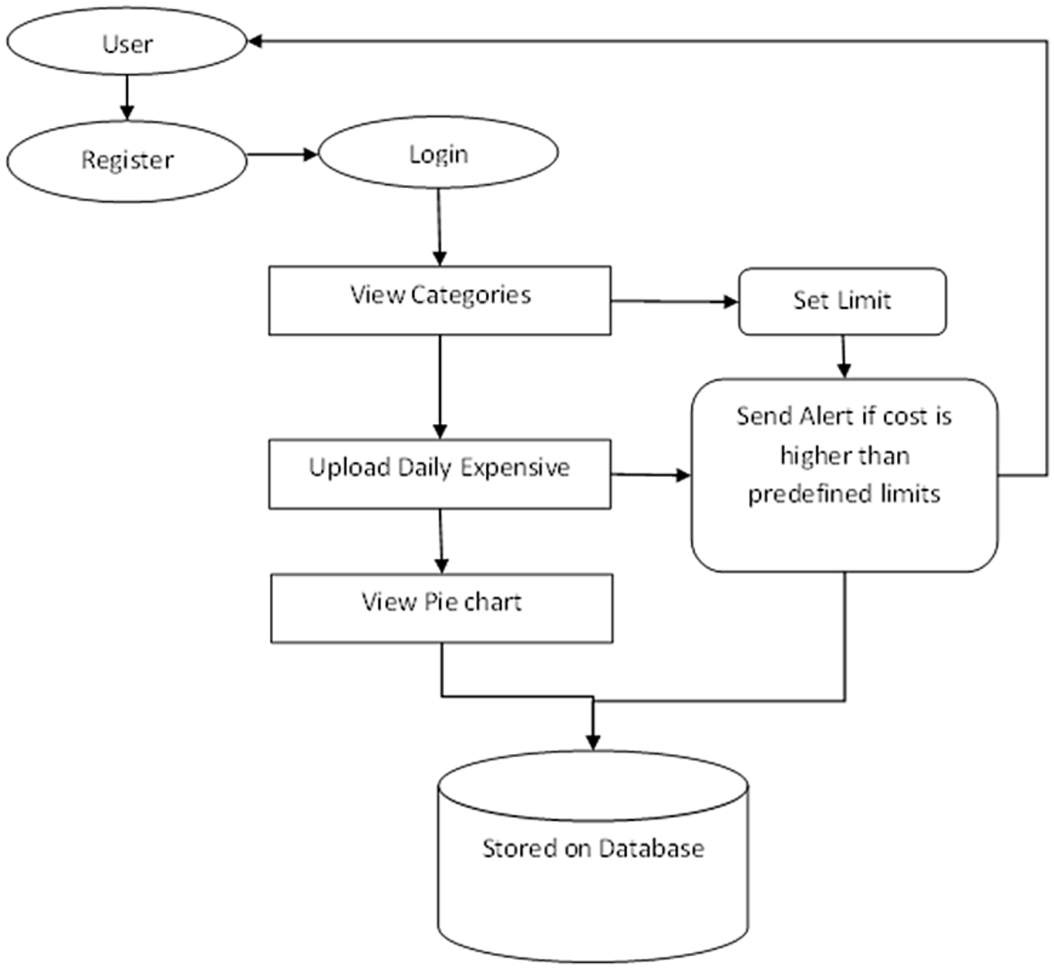
LEVEL 2



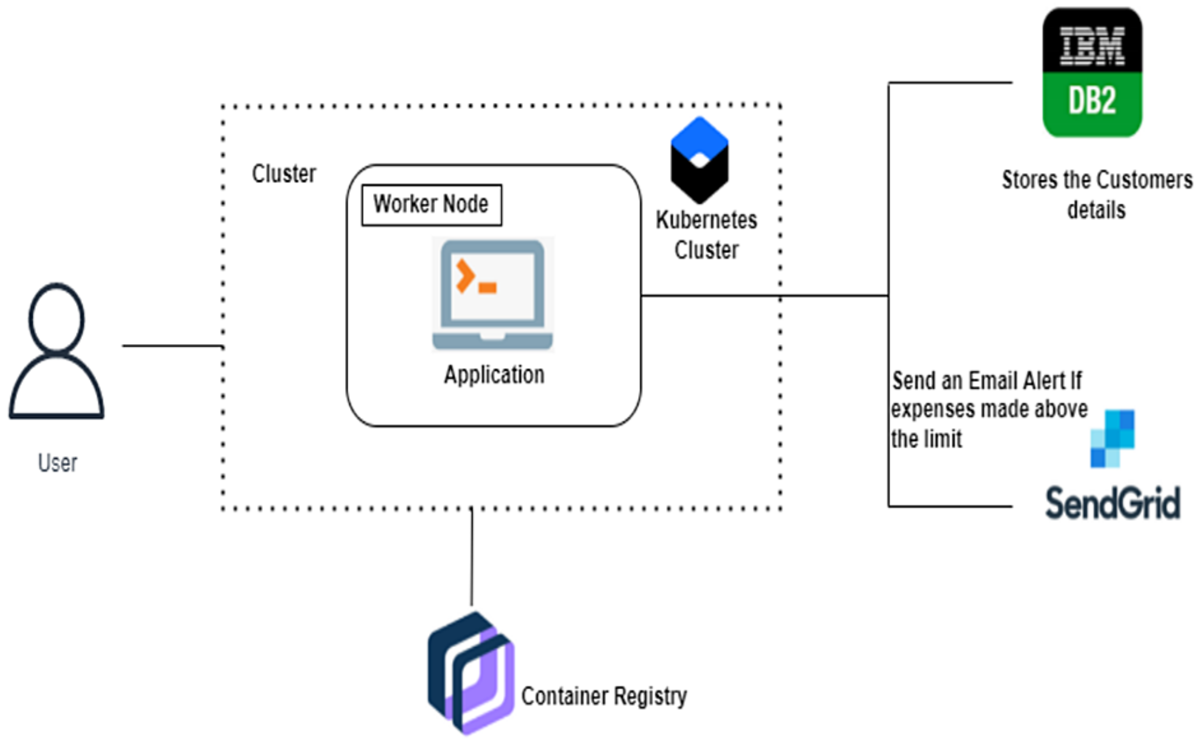
5.2 SOLUTION &TECHNICAL ARCHITECTURE

A solutions architect creates the broad technical vision for a specific strategy to address a business problem. A solutions architect creates the broad technical vision for a specific strategy to address a business problem. They design, describe, and supervise the solution.

SOLUTION ARCHITECTURE :

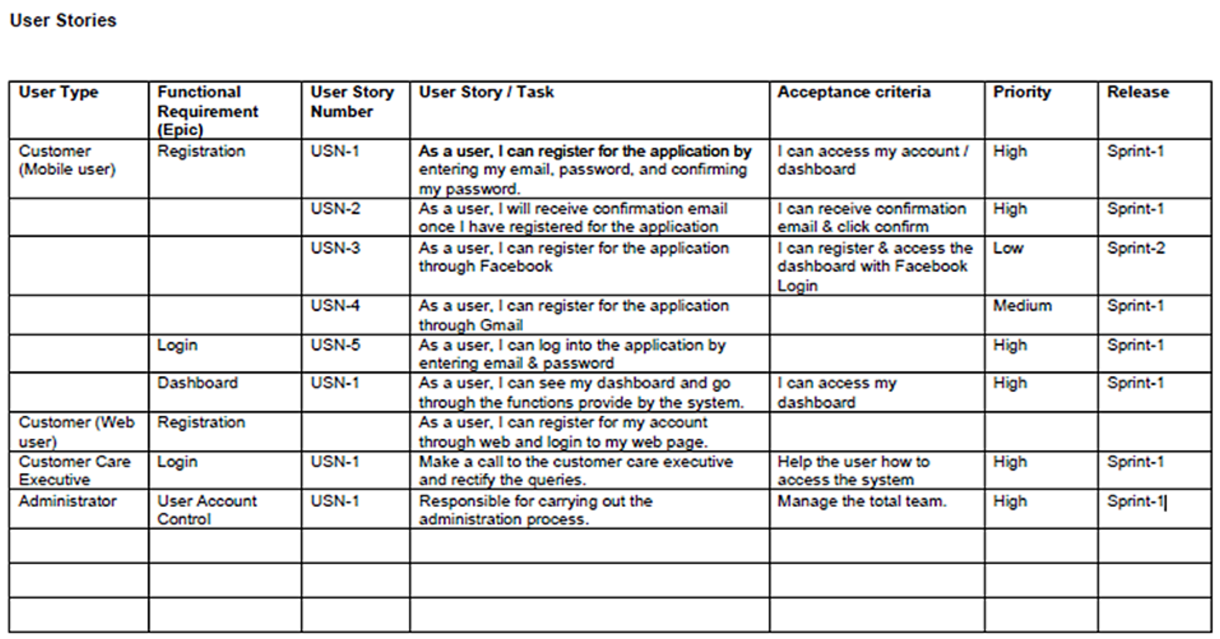


TECHNICAL ARCHITECTURE :



5.3 USER STORIES

An informal , generalized explanation of a software feature written from the client's or end user's point of view is known as a "user narrative." In a user story, the value that a piece of work will provide the client is explained.



User Stories

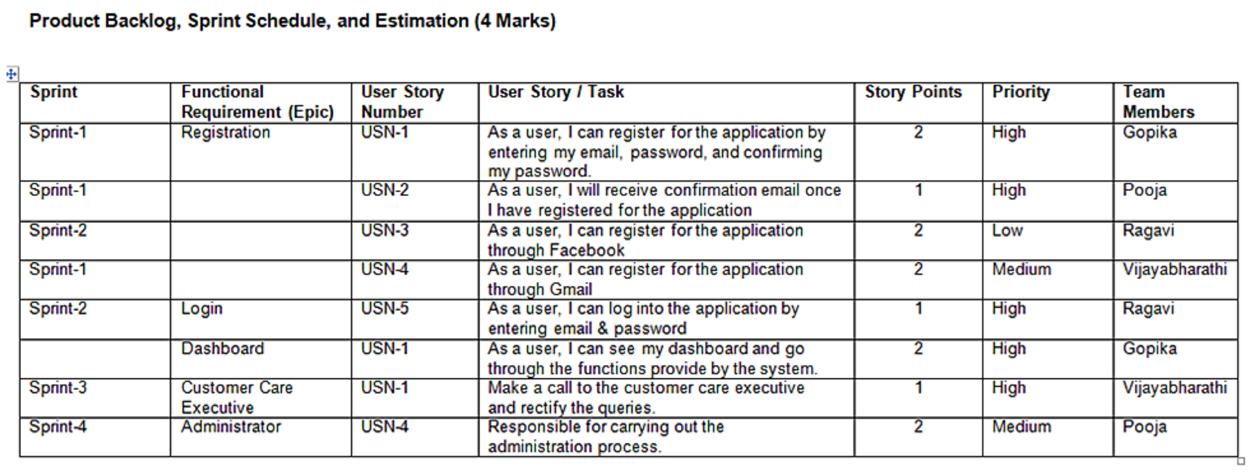
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	**As a user, I can register for the application by** entering my email, password, and confirming my password.	I can access my account / dashboard	High	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
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-1	As a user, I can see my dashboard and go through the functions provide by the system.	I can access my dashboard	High	Sprint-1
Customer (Web user)	Registration		As a user, I can register for my account through web and login to my web page.			
Customer Care Executive	Login	USN-1	Make a call to the customer care executive and rectify the queries.	Help the user how to access the system	High	Sprint-1
Administrator	User Account Control	USN-1	Responsible for carrying out the administration process.	Manage the total team.	High	Sprint-1

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

The team as a whole estimates during the sprint planning meeting. The goal of the estimation would be to prioritise the User Stories for the Sprint and assess the team's capacity to complete them inside the Sprint's Time Box.



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

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Gopika
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Pooja
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Ragavi
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Vijayabharathi
Sprint-2	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Ragavi
	Dashboard	USN-1	As a user, I can see my dashboard and go through the functions provide by the system.	2	High	Gopika
Sprint-3	Customer Care Executive	USN-1	Make a call to the customer care executive and rectify the queries.	1	High	Vijayabharathi
Sprint-4	Administrator	USN-4	Responsible for carrying out the administration process.	2	Medium	Pooja

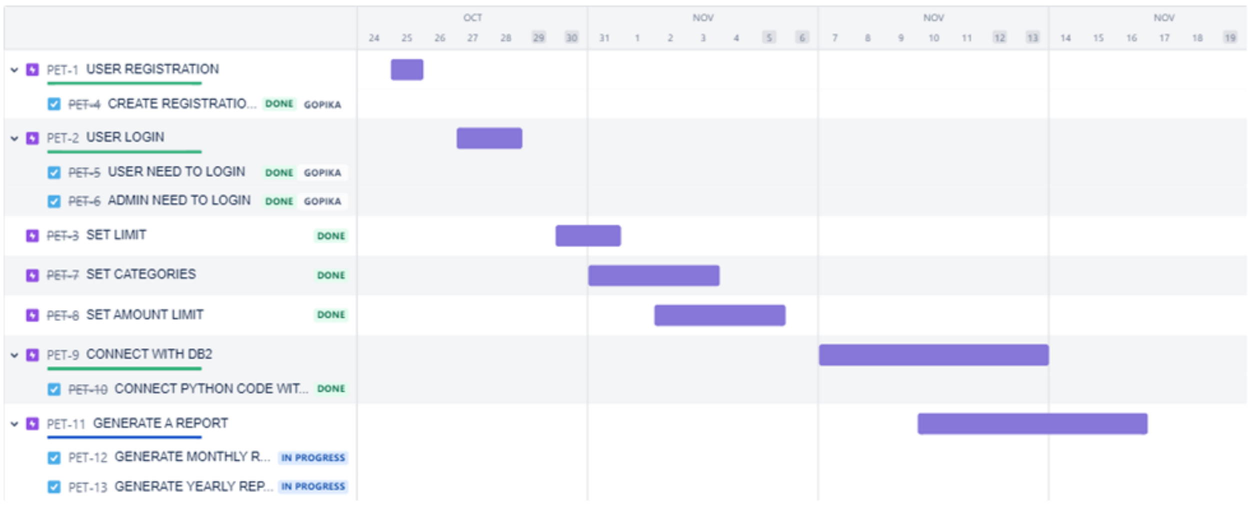
6.2 SPRINT DELIVERY SCHEDULE

Planning a sprint schedule is one of the first steps in the agile sprint planning process. It's something that requires adequate research, planning, and communication.

Sprint	Total StoryPoints	Duration	SprintStartDate	SprintEndDate(Planned)	StoryPoints Completed (as onPlannedEndDate)	Sprint Release Date(Actual)
Sprint-1	20	6Days	24Oct2022	29Oct2022	20	29Oct2022

Sprint -2	20	6Days	31Oct2022	05Nov 2022	20	05Nov2022
Sprint -3	20	6Days	07Nov2022	12Nov2022	20	12Nov2022
Sprint -4	20	6Days	14Nov2022	19Nov2022	20	19Nov2022

6.3 REPORTS FROM JIRA



PET-1 USER REGISTRATION
~~PET-4~~ CREATE REGISTRATIO... DONE GOPIKA
PET-2 USER LOGIN
~~PET-5~~ USER NEED TO LOGIN DONE GOPIKA
~~PET-6~~ ADMIN NEED TO LOGIN DONE GOPIKA
~~PET-3~~ SET LIMIT DONE
~~PET-7~~ SET CATEGORIES DONE
~~PET-8~~ SET AMOUNT LIMIT DONE
PET-9 CONNECT WITH DB2
~~PET-10~~ CONNECT PYTHON CODE WIT... DONE
PET-11 GENERATE A REPORT
PET-12 GENERATE MONTHLY R... IN PROGRESS
PET-13 GENERATE YEARLY REP... IN PROGRESS

CHAPTER 7

CODING & SOLUTIONING

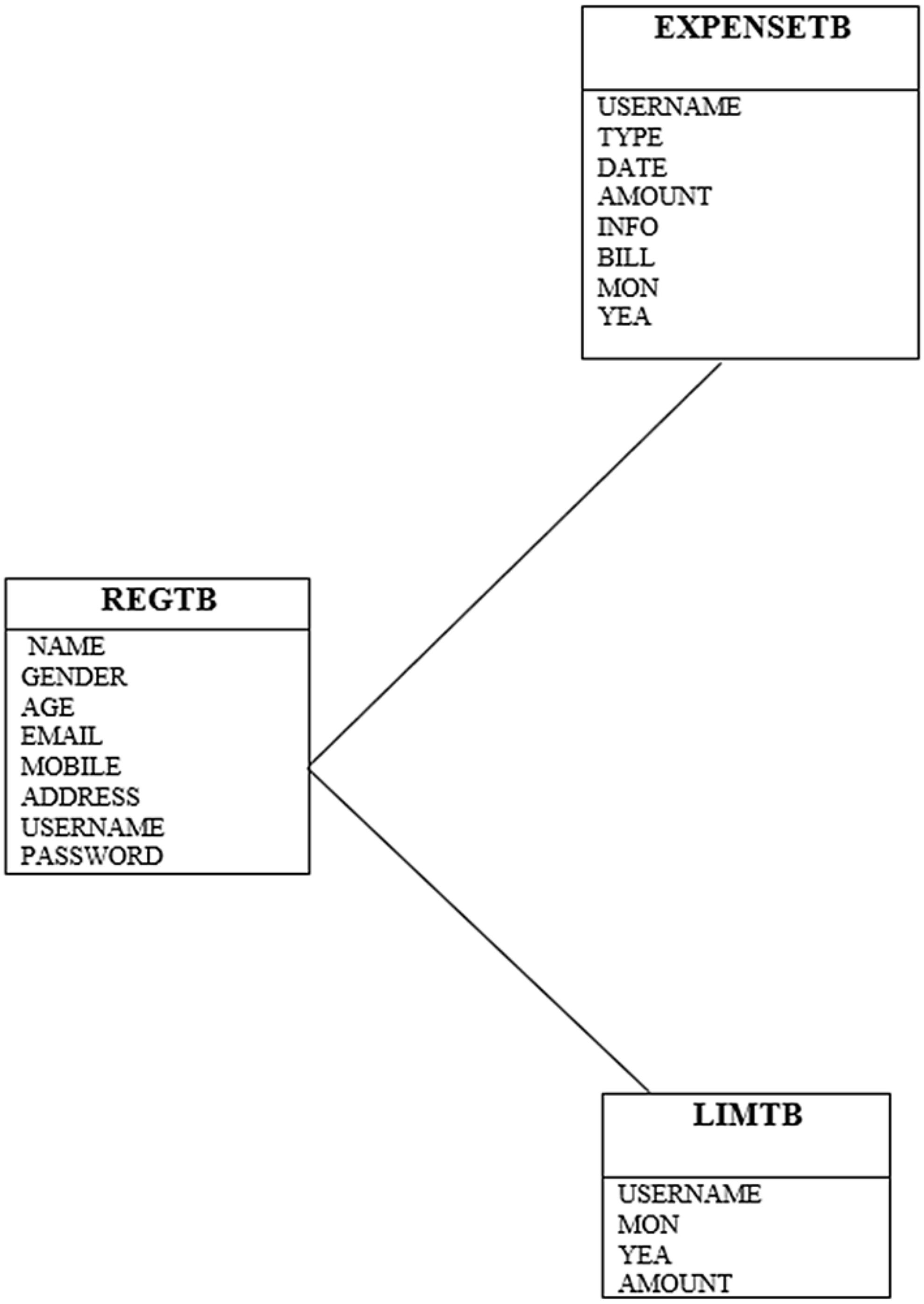
7.1 FEATURE

An email alert to send if an user goes beyond the limit.Following are the code to send the email alert.

```
 def sendmsg(Mailid,message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders
    fromaddr = "sampletest685@gmail.com"
    toaddr = Mailid
    msg = MIMEMultipart()
    msg['From'] = fromaddr
    msg['To'] = toaddr
    msg['Subject'] = "Alert"
    body = message
    msg.attach(MIMEText(body, 'plain'))
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login(fromaddr, "hneucvnontsuwgpj")
    text = msg.as_string()
    s.sendmail(fromaddr, toaddr, text)
if __name__ == '__main__':
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,host='0.0.0.0')
```

7.2 DATABASE SCHEMA

The word "database schema" is used to describe a structure or layout thatidentifies a set of data .In other words, it outlines the structure and connectivityof the data. A database schema may therefore contain schema objects such astables, views, fields, relationships, packages, indexes, types, and many othercomponents.


EXPENSETB
USERNAME
TYPE
DATE
AMOUNT
INFO
BILL
MON
YEA
REGTB
NAME
GENDER
AGE
EMAIL
MOBILE
ADDRESS
USERNAME
PASSWORD
LIMTB
USERNAME
MON
YEA
AMOUNT

CHAPTER 8

TESTING

8.1 TEST CASES

An activity sequence carried out on a system to verify software compliance and proper operation is known as a test case. A test case's objective is to assess whether different system features operate as anticipated and to verify that the system complies with all applicable standards, directives, and user requirements. A good technique to identify issues or flaws with the system is to create a test case. Test cases, which serve as step-by-step instructions for each system test, are typically created by members of the testing team or the quality assurance (QA) team. Once a system feature or group of features has been completed by the development team, testing can start. A test suite is a grouping or collection of test cases.

8.2 USER ACCEPTANCE TESTING

Before deploying the software programme to a production environment,the end user or client does a sort of testing known as user acceptance testing, orUAT. After functional, integration, and system testing are complete, UAT iscarried out as the last stage of testing.

8.2.1 PURPOSE OF USER ACCEPTANCE TESTING

UAT's primary goal is to verify the whole business process. It doesn'tconcentrate on minor typos, misspellings, or system testing.A User AcceptanceTesting is performed using production-like data prepared in a separate testingenvironment. It will resemble black box testing and involve two or more endusers.

8.2.2 NEEDS OF USER ACCEPTANCE TESTING

After software has undergone Unit, Integration, and System testing, UserAcceptance Testing becomes necessary because developers may have builtsoftware based on requirements documents according to their own understandingand further required changes during development may not have been effectivelycommunicated to them. This makes it necessary to test whether the final productis accepted by the client or end-user.

1. Product is created by developers based on requirements documents, which may not accurately reflect the demands of the client for the software.

1. During the project, requirements changes might not be successfully conveyed to the developers.

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

Performance metrics are described as numbers and information that areindicative of the activities, capacities, and general calibre of a company.Performance measurements may take many various forms, such as sales, profit,ROI, customer satisfaction, customer reviews, personal reviews, general quality,and reputation in the market. When evaluated through various industries,performance indicators might differ greatly.Metrics of performance are essentialto the success of a company. Because these metrics assist direct and assess anorganization's success, it is crucial that businesses choose their primaryperformance measures and concentrate on these areas. Important successelements are only helpful if they are recognised and monitored. In order forbusiness metrics to provide accurate responses and for the proper questions to beposed, attentive management is also required.

9.1.1 REQUIREMENTS FOR MEASURING PERFORMANCE

The unit or variable being measured together with a justification for why itis important to the measurement's goal.A practical explanation A thorough yet simple explanation of the measuring method

Plan for analysis: A monthly report including comparisons to the monthbefore, year over year, and year to date is a common illustration. The various timeperiods provide the data a richer context and enable graphical presentation.

A straightforward analysis plan template is a control chart. In addition tosome analysis (control limits) that enables the viewer to distinguish betweencommon causes, special causes, and random variation, it also provides a graphicalcontext that illustrates the continuity of changes over time.

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

10.1 ADVANTAGES

1. It Aids in Budget Compliance.

2. We can become conscious of bad spending behaviour.
3. Tracking Your Expenses Can Reveal Spending Issues.
4. Save Time by Reducing Number Crunching.

10.2 DISADVANTAGES

Every system needs some limitation to ensure it performs as it supposed. There are also a few problems and limitations that occur throughout the development of these project which are: The system cannot retrieve username and password if a user happens to forget them. The report only shows separated reports for budget, income and expense, thus the chart do not show or evaluate income versus expenses made.

CHAPTER 11

CONCLUSION

The majority of the shortcomings of the old system have been mostly resolved by the newone, which operates in accordance with the provided design specification. Design the projectthat is more effective than existing income and expense trackers. The manual calculation fordetermining the income and expense every month is successfully avoided by the project. Themodules are created in a way that is both effective and appealing. The created methodseliminate the issue and satisfy the needs by giving accurate and thorough information. Thesystem has complied with every criterion that the user envisaged. The newly created systemtakes less time to process, and all the information is updated and processed right away.

CHAPTER 12

FUTURE SCOPE

The system is well operated and function as planned, however, there are a few suggestion tomake it even more better and usable in the future. Here are some enchantment for the systemto work more efficient in future. For current report, it is only limited to a few options.Hence, maybe it will be better if the report can generate a bar graph or line graph that showsall reports together instead being separated. It would a lot easier for user to evaluate theirspending too. l It would be great if the system can perform any online payment instead ofonly able to generate reports and forecast budget.

CHAPTER 13

APPENDIX

13.1 SOURCE CODE

```
from flask import Flask, render_template, flash, request, session,send_file
from flask import render_template, redirect, url_for, request
import datetime
import sys
import os
import ibm_db
import pandas
import ibm_db_dbi
from sqlalchemy import create_engine
engine = create_engine('sqlite://',
                    echo = False)
dsn_hostname = "9938aec0-8105-433e-8bf9-
0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud"
dsn_uid = "fpb08636"
dsn_pwd = "yN8FP21wXjrDrbPK"
dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"
dsn_port = "32459"
dsn_protocol = "TCPIP"
dsn_security = "SSL"
dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol, dsn_uid,
dsn_pwd,dsn_security)
try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host: ", dsn_hostname)
except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )
app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'
@app.route("/")
def homepage():
    return render_template('index.html')
```

```
@app.route("/AdminLogin")
def AdminLogin():
   return render_template('AdminLogin.html')
@app.route("/UserLogin")
def UserLogin():
   return render_template('UserLogin.html')
@app.route("/NewUser")
def NewUser():
   return render_template('NewUser.html')
@app.route("/Search")
def Search():
   return render_template('Search.html')
@app.route("/MonthReport")
def MonthReport():
   return render_template('MonthReport.html')
@app.route("/AdminHome")
def AdminHome():
   conn = ibm_db.connect(dsn, "", "")
   pd_conn = ibm_db_dbi.Connection(conn)
   selectQuery = "SELECT * from regtb "
   dataframe = pandas.read_sql(selectQuery, pd_conn)
   dataframe.to_sql('Employee_Data',
                    con=engine,
                    if_exists='append')
   data = engine.execute("SELECT * FROM Employee_Data").fetchall()
   return render_template('AdminHome.html',data=data)
@app.route("/SetLimit")
def SetLimit():
   user = session['uname']
   conn = ibm_db.connect(dsn, "", "")
```

```
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM limtb where  username ='" + user + "' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data',con=engine,if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('Limit.html', data=data)
@app.route("/Report")
def Report():
    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM expensetb  "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('Report.html',data=data)
@app.route("/UserHome")
def UserHome():
    user = session['uname']
    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM regtb where username='" + user + "'"
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('UserHome.html',data=data)
@app.route("/adminlogin", methods=['GET', 'POST'])
def adminlogin():
    error = None
    if request.method == 'POST':
       if request.form['uname'] == 'admin' or request.form['password'] == 'admin':
            conn = ibm_db.connect(dsn, "", "")
            pd_conn = ibm_db_dbi.Connection(conn)
            selectQuery = "SELECT * FROM regtb "
            dataframe = pandas.read_sql(selectQuery, pd_conn)
            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
            data = engine.execute("SELECT * FROM Employee_Data").fetchall()
            return render_template('AdminHome.html' , data=data)
        else:
         return render_template('index.html', error=error)
@app.route("/userlogin", methods=['GET', 'POST'])
def userlogin():
    if request.method == 'POST':
        username = request.form['uname']
        password = request.form['password']
        session['uname'] = request.form['uname']
        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery = "SELECT * from regtb where UserName='" + username + "' and password='" +
password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)
        if dataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
            selectQuery = "SELECT * from regtb where UserName='" + username + "' and password='" +
password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)
            dataframe.to_sql('Employee_Data',
                             con=engine,
                             if_exists='append')
            # run a sql query

            data1= engine.execute("SELECT * FROM Employee_Data").fetchall()

            for item in data1:

                session["mail"] = item[4]

            print(session["mail"])

        return render_template('UserHome.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())

@app.route("/UReport")

def UReport():

    name1 = session['uname']

    conn = ibm_db.connect(dsn, "", "")

    pd_conn = ibm_db_dbi.Connection(conn)

    selectQuery = "SELECT * FROM expensetb where username='"+ name1 +"' "

    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')

    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('UReport.html',data=data)

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

def dsearch():

    if request.method == 'POST':

        import datetime

        file = request.files['fileupload']

        file.save('static/upload/'+file.filename)

        name1 = session['uname']

        type = request.form['c1']

        dat = request.form['t1']

        amt = request.form['t2']

        info = request.form['t3']

        date_object = datetime.datetime.strptime(dat, '%Y-%m-%d').date()

        mon = date_object.strftime("%m")
        yea = date_object.strftime("%Y")

        global lim1

        global lim2

        lim1 = 0

        lim2 = 0

        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery = "SELECT * from limtb where mon='" + mon + "' and yea='" + yea + "' and 
Username='"+ name1 +"'"

        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:

            alert = 'Please Set Expense Limit'

            return render_template('goback.html', data=alert)

        else:

            dataframe.to_sql('limtb',con=engine,if_exists='append')

            data1 = engine.execute("SELECT * FROM limtb").fetchall()

            for item in data1:

                lim1 = item[4]

                print(lim1)

        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery =  "SELECT sum(Amount) as amt  from expensetb where mon='" + mon + "' and yea='" + 
yea + "' and Username='" + name1 + "'"

        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:

            lim2 = float(0.00)

        else:

            dataframe.to_sql('expensetb', con=engine, if_exists='append')

            data1 = engine.execute("SELECT * FROM expensetb").fetchall()

            for item2 in data1:

                lim2 = item2[1]
            print(lim1)
        if lim2 is None:  # Checking if the variable is None

          lim2 = 0.00
        else:
            print("Not None")
        if (float(lim2) <= float(lim1)):
            conn = ibm_db.connect(dsn, "", "")
            insertQuery =  "INSERT INTO expensetb VALUES ('" + name1 + "','" + type + "','" + dat + "','" + amt
+ "','" + info + "','" + file.filename + "','" + date_object.strftime("%m") + "','" + date_object.strftime("%Y")
+ "')"
            insert_table = ibm_db.exec_immediate(conn, insertQuery)
            print(insert_table)
            alert = 'New Expense Info Saved'
            return render_template('goback.html', data=alert)
        else:
            alert = 'Limit Above  Expense'

            sendmsg(session["mail"],"Limit Above  Expense")
            return render_template('goback.html', data=alert)
@app.route("/setlimit", methods=['GET', 'POST'])
def setlimit():
    if request.method == 'POST':
        name1 = session['uname']
        mon = request.form['mon']
        yea = request.form['yea']
        amt = request.form['t2']
        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery = "SELECT * from limtb where username='" + name1 + "' and mon='" + mon + "' and
yea='"+ yea +"' "
```

```
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:

            insertQuery = "INSERT INTO limtb VALUES ('" + name1 + "','" + mon + "','" + yea + "','" + amt + "')"

            insert_table = ibm_db.exec_immediate(conn, insertQuery)

            print(insert_table)

            conn = ibm_db.connect(dsn, "", "")

            pd_conn = ibm_db_dbi.Connection(conn)

            selectQuery = "SELECT * FROM limtb where  username ='" + name1 + "' "

            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')

            data = engine.execute("SELECT * FROM Employee_Data").fetchall()

            return render_template('Limit.html', data=data)

        else:


            alert = 'Already Set  Expense limit Remove And Set New!'

            return render_template('goback.html', data=alert)

@app.route("/remove")

def remove():

    uname =  request.args.get('uname')

    mon = request.args.get('mon')

    year = request.args.get('year')

    conn = ibm_db.connect(dsn, "", "")

    pd_conn = ibm_db_dbi.Connection(conn)

    insertQuery = "delete from limtb  where UserName='"+ uname +"' and mon='"+ mon +"' and Yea='"+
year +"' "

    insert_table = ibm_db.exec_immediate(conn, insertQuery)

    selectQuery = "SELECT * from limtb "

    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data',

                    con=engine,
                 if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('Limit.html', data=data )
@app.route("/newuser", methods=['GET', 'POST'])
def newuser():
    if request.method == 'POST':
        name1 = request.form['name']
        gender1 = request.form['gender']
        Age = request.form['age']
        email = request.form['email']
        pnumber = request.form['phone']
        address = request.form['address']
        uname = request.form['uname']
        password = request.form['psw']
        conn = ibm_db.connect(dsn, "", "")
        insertQuery = "INSERT INTO regtb VALUES ('" + name1 + "','" + gender1 + "','" + Age + "','" + email +
"','" + pnumber + "','" + address + "','" + uname + "','" + password + "')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)
        print(insert_table)
        # return 'file register successfully'
    return render_template('UserLogin.html')
@app.route("/msearch", methods=['GET', 'POST'])
def msearch():
    if request.method == 'POST':
        if request.form["submit"] == "Search":
            mon = request.form['mon']
            yea = request.form['yea']
            uname = session['uname']
            import matplotlib.pyplot as plt
            import matplotlib
            matplotlib.use('Agg')

            conn = ibm_db.connect(dsn, "", "")

            pd_conn = ibm_db_dbi.Connection(conn)

            selectQuery = "select Type, sum(Amount) as MSales from expensetb where mon='" + mon + "'
and yea='"+ yea +"' and Username='"+ uname +"' group by Type "

            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('expensetb',

                    con=engine,

                    if_exists='append')

            data = engine.execute("SELECT * FROM expensetb").fetchall()

            Month = []

            MSales = []

            Month.clear()

            MSales.clear()

            for i in data:

                Month.append(i[1])

                MSales.append(i[2])

            print("Month = ", Month)

            print("Total Sales = ", MSales)

            plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])

            plt.xlabel("Type")

            plt.ylabel("Total Expenses")

            plt.title("Monthly Expenses")

            import random

            n = random.randint(1111, 9999)

            plt.savefig('static/plott/' + str(n) + '.jpg')

            iimg = 'static/plott/' + str(n) + '.jpg'

            selectQuery = "SELECT * FROM expensetb where mon='" + mon + "' and yea='"+ yea +"' and
Username='"+ uname +"' "

            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
            data = engine.execute("SELECT * FROM Employee_Data").fetchall()

            return render_template('MonthReport.html', data=data, dataimg=iimg)

        elif request.form["submit"] == "DSearch":

            d1 = request.form['d1']

            d2 = request.form['d2']

            uname = session['uname']

            import matplotlib.pyplot as plt

            import matplotlib

            matplotlib.use('Agg')

            conn = ibm_db.connect(dsn, "", "")

            pd_conn = ibm_db_dbi.Connection(conn)

            selectQuery ="select Type, sum(Amount) as MSales,date from expensetb where date between '"
+ d1 + "' and '" + d2 + "' and Username='" + uname + "' group by Type,date "

            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('expensetb',

                             con=engine,

                             if_exists='append')

            data = engine.execute("SELECT * FROM expensetb").fetchall()

            Month = []

            MSales = []

            Month.clear()

            MSales.clear()

            for i in data:

                Month.append(i[1])

                MSales.append(i[2])

            print("Month = ", Month)

            print("Total Sales = ", MSales)

            plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])

            plt.xlabel("Type")

            plt.ylabel("Total Expenses")
            plt.title("Date To Date  Expenses")
            import random
            n = random.randint(1111, 9999)
            plt.savefig('static/plott/' + str(n) + '.jpg')
            iimg = 'static/plott/' + str(n) + '.jpg'
            selectQuery =  "SELECT * FROM expensetb where date between '" + d1 + "' and '" + d2 + "' and 
Username='" + uname + "' "
            dataframe = pandas.read_sql(selectQuery, pd_conn)
            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
            data = engine.execute("SELECT * FROM Employee_Data").fetchall()
            return render_template('MonthReport.html', data=data, dataimg=iimg)
def sendmsg(Mailid,message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders
    fromaddr = "sampletest685@gmail.com"
    toaddr = Mailid
    msg = MIMEMultipart()
    msg['From'] = fromaddr
    msg['To'] = toaddr
    msg['Subject'] = "Alert"
    body = message
    msg.attach(MIMEText(body, 'plain'))
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login(fromaddr, "hneucvnontsuwgpj")
    text = msg.as_string()
    s.sendmail(fromaddr, toaddr, text)
```

```
if __name__ == '__main__':
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,host='0.0.0.0')
```

13.2 GITHUB & PROJECT DEMO LINK

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-1172-1658377123

PROJECT DEMO LINK:

https://drive.google.com/file/d/1Ma4fc_0xUDlkKG4zOqO82PPuvGULC92R/view?usp=sharing