PROJECT REPORT - PERSONAL EXPENSE TRACKER

1. INTRODUCTION:

Project Overview:

The main objective of this project is to design an application for tracking personal expenses and provide an analysis which can be understood easily by the user.

The project has been idealized and completed through eventual progress and brainstorming sessions of the team and its members:

Maniyarasan M

Lokesh J

Lokesh V

Muniasamy M

Purpose:

- To manage finances efficiently.
- To help with budgeting and accounting.
- To obtain insights about financial management.
- To reduce unnecessary expenses.
- To provide alerts when limits are exceeded.

2. LITERATURE SURVEY:

Personal Expense Tracker Application: An exhaustive literature survey on related topic suggest that earlier tracking was performed manually. These things were done in an old school way you can say more likely in a notebook or copybook these things were written as per the willingness of a person in simple words you can call is as “The quite manually stuff”. Then after that they have calculate the entire expenditure at the end of the month or week and a report is generated against the expenditure in comparison to the previous month or information related to that. So, they face a certain problem that time are Data is not accurate, reports in not up to the mark ,a single mistake in a manual calculation and actually cost you much ,its time consuming boring and most of the time insufficient,going through all the data back then rewriting them off actually makes way fussier,it can be easily stolen or loss as well your information is not safe there. Babad and Balachandran in 1993 states that traditional cost accounting systems maintain all overheads in one pool and give equal weight to all activities and costs in it We always have known that “pen is mightier than sword” but that thing doesn’t fit with every specific tasks it varies from need-to-need or tasks-to-tasks these days when the amount data is quite enormous. It becomes way more difficult to handle them off. Soon excel also become a way on maintain a record of expenses and analysis.

A Research at university on Tennessee on expense tracker of by (Dan Underwood, 2011)

In which using excel accounting term designed a Cost Allocation tool 1 in which a spreadsheet is used to allocate the product category both by site and the cooperation and a Cost allocation tool 2 which is a developed to further integrate and allocate cost to identify which manufacturer is profitable or which is not. This research used excel and designed this CAT tool in which both the spreadsheets are required to use to identify where we could reduce expenses or better managed it. (Girish Bekaroo, 2007) did a research on intelligent online budget that manages the expenses and used to give the graphical analysis of data.(Stephan snow and Dhayal Vyas, 2015) mentioned in his paper. "Managing finances is a practice carried out daily in homes across the world. Despite this, the practice is not yet a strong focus for HCI work in the home".

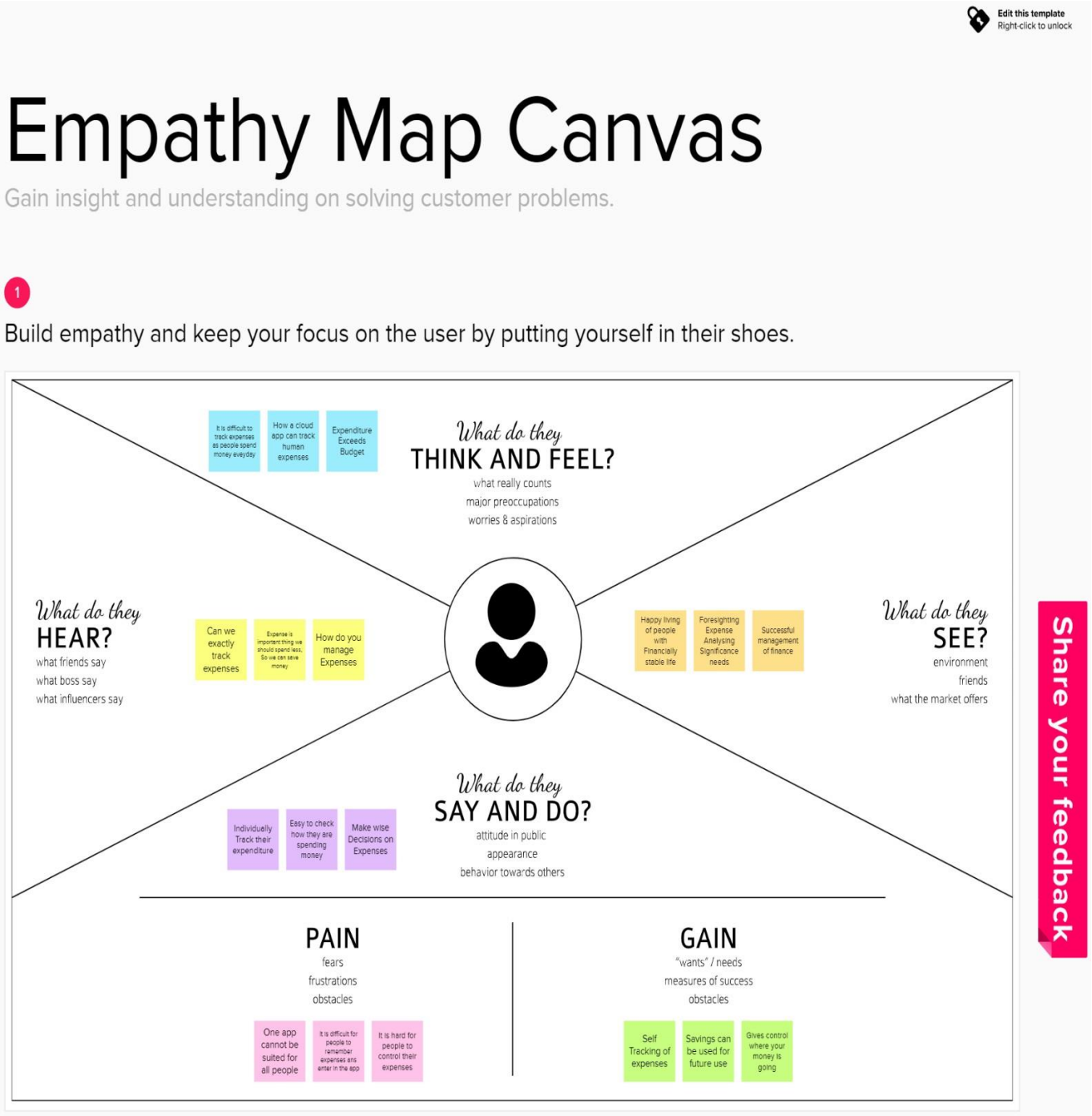
Researchers of Nandha and Anna university (2016) created an android version of expense manager in with they used post and remark techniques for underlining the expenses and some of the data mining features for analyzing the market value well. (R N Rajprabha, 2017) created an android version of family budget manager with later evolved in PDA and tablet features. (Ravi Sharma, 2017) stated users sometimes feels uncomfortable in sharing their personal information with an app and he suggested security and usability are two major concerns. Even the advanced UI needs to maintain retention .Researchers of Mother Terresa university, Andhra Pradesh (2019) also stated an online income and budget tracker in a website mode but that project used csv mode to store data but that project had a drawback in its existing model as it can't handle the data efficiently in addition to that it wasn't user-friendly and an unpopulated data project. All these researches above suggest some of the modern way of dealing with expense tracking. Many of the researches like these actually represents the evolution in ideas with time "evolution is not a necessity it's more like change in thinking and time" in which we analyze estimate and evaluate the things according to new requirements. But still the kind of technology used in it is kind of projects were used in previous days there are certain android apps as well still they too also have different consequences as well as drawbacks in itself. And I also feel like these should be way easier to handle to a desktop device. As sometimes android apps will provide in accurate results if the information is incorrect and many of the times, we almost got forget to enter details too and most them don't even provide notification for that as well.

PROBLEM STATEMENT:

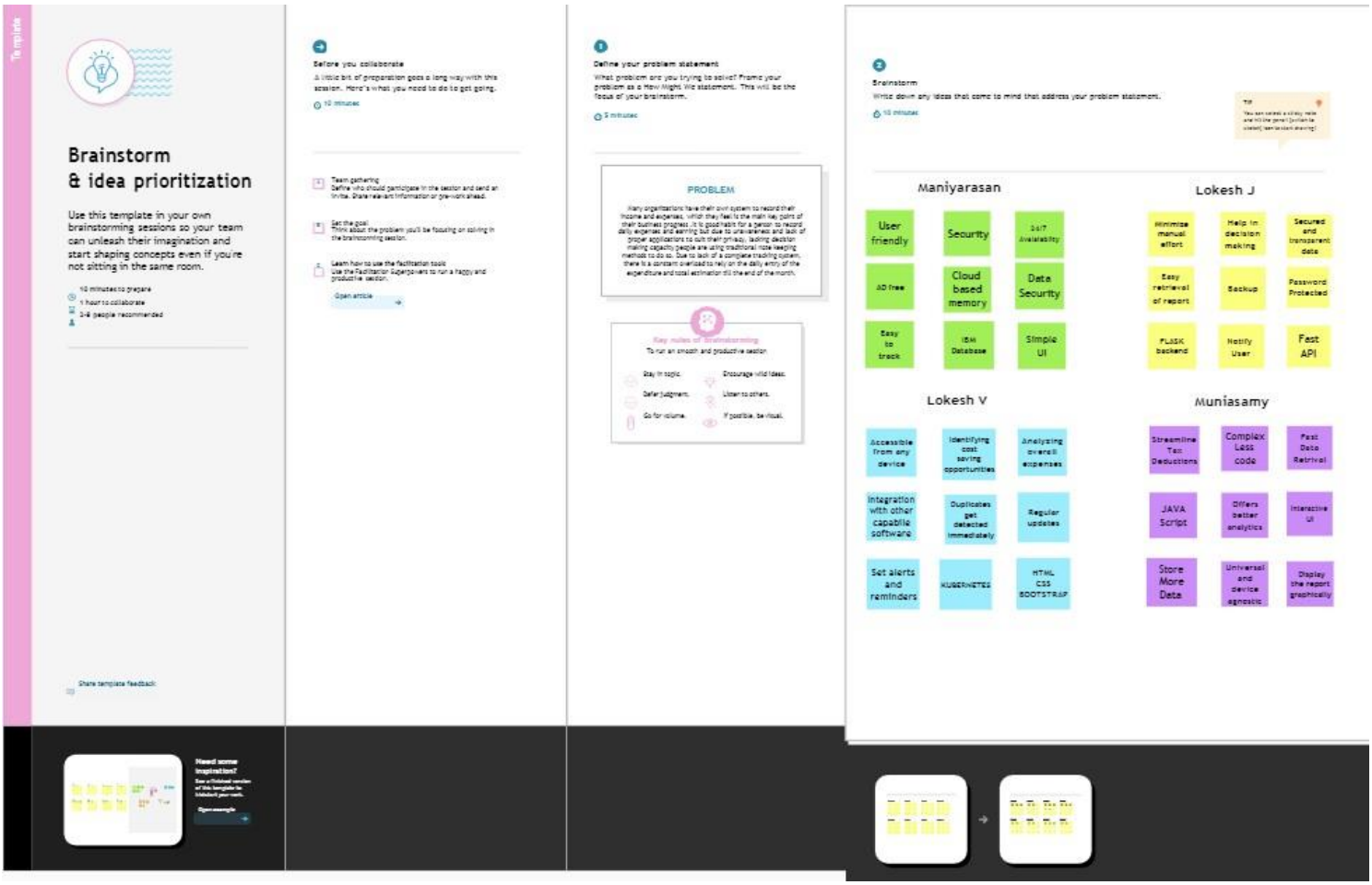
Expense Tracker At the instant, there is no as such complete solution present easily or we should say free of cost which enables a person to keep a track of its daily expenditure easily. To do so a person has to keep a log in a diary or in a computer, also all the calculations needs to be done by the user which may sometimes results in errors leading to losses. Due to lack of a complete tracking system, there is a constant overload to rely on the daily entry of the expenditure and total estimation till the end of the month. As the name itself suggests, this project is an attempt to manage our daily expenses in a more efficient and manageable way. The system attempts to free the user with as much as possible the burden of manual calculation and to keep the track of the expenditure. Instead of keeping a dairy or a log of the expenses

3. IDEATION AND PROPOSED SOLUTION:\

Empathy Map Canvas



Ideation & Brainstorming:

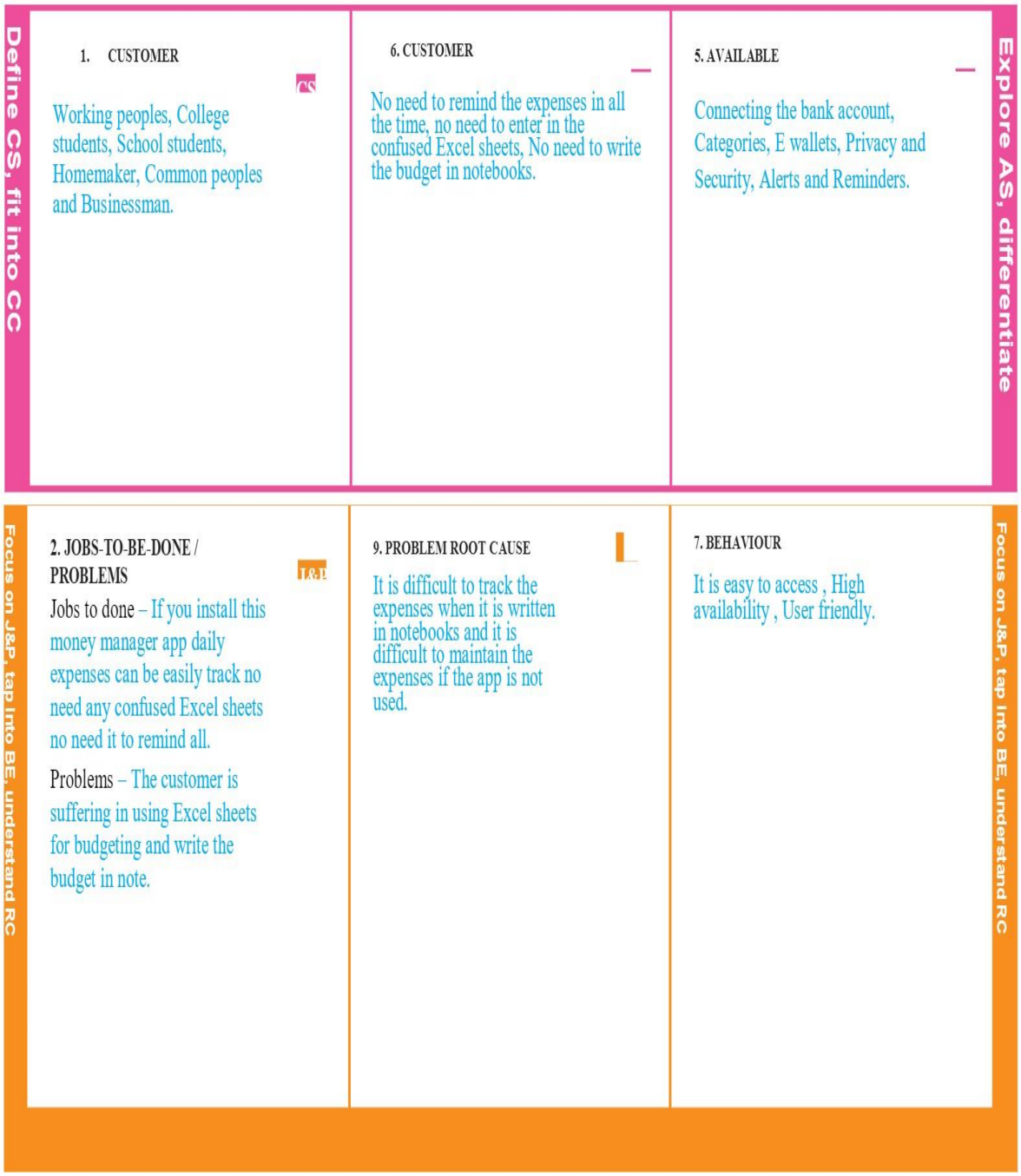


Proposed Solution

S.No.	Parameter	Description
1.	**Problem Statement (Problem to be solved)**	Personal expense tracker is used to track daily and monthly expenses and savings of family and individual person.
2.	**Idea/Solution description**	We are developing a cloud application for expense tracking which will track expenses.
3.	**Novelty/ Uniqueness**	In previous days expenses are tracked manually using humans, But now application is being developed which will reduce human work.
4.	**Social Impact/ Customer Satisfaction**	The app will track daily expenses and show them So customers can plan according to their budget daily and spend money. Their money is saved.
5.	**Business Model (Revenue Model)**	App is developed and it can be given to people on Subscription basis for some amount. So app owner Can also be benefitted .
6.	**Scalability of the Solution**	It can be implemented using any web framework and can be made available to everyone in need.

Problem Solution fit

Project Title: PERSONAL EXPENSE TRACKER APPLICATION **Project Design Phase-I - Solution Fit**
Template Team ID: PNT2022TMID00810



4. REQUIREMENT ANALYSIS

Functional requirement

Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Date	16 October 2022
Team ID	PNT2022TMID00810
Project Name	Project – Personal Expense Tracker Application
Maximum Marks	4 Marks

Functional Requirements:

- A functional requirement defines a function of a system or its component, where a function is described as a specification of behaviour between inputs and outputs.
- It specifies "what should the software system do?"'
- It is mandatory.
- Defined at a component level.
- Usually easy to define'
- Helps you verify the functionality of the software'

Following are the functional requirements of the proposed solution:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN.
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP.
FR-3	User Access/login	Login using e-mail and password.
FR-4	User Data	Adding user expenses through input field and categories.
FR-5	User Alert	Sending alert to e-mail.

Non-Functional requirements

Non-functional Requirements:

- A non-functional requirement defines the quality attribute of a software system.
- It places constraint on "How should the software system fulfil the functional requirements?"'
- It is not mandatory'

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Customers can use the application in all types of smartphones .
NFR-2	**Security**	Customers are asked to set a password while registering , then everytime they login again it asks for password.
NFR-3	**Reliability**	There is chat option in app ,if there is any issue , Customers can send them through chat.
NFR-4	**Performance**	Customers will have a smooth experience while using the application, as it is simple and is well optimised.
NFR-5	**Availability**	Mobile App.
NFR-6	**Scalability**	In future, may be cross-platform mobile applications can be developed as the user base grows.

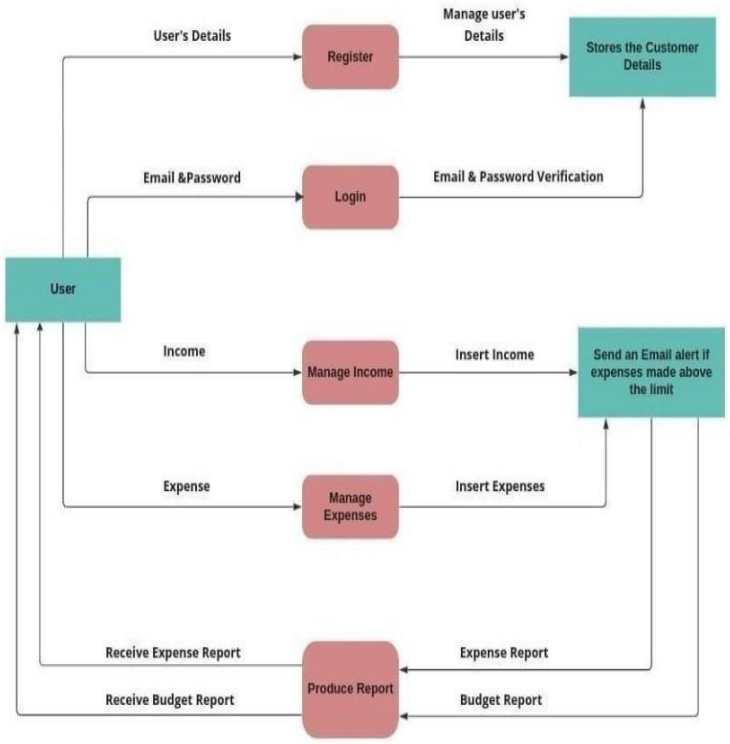
5. PROJECT DESIGN

Data Flow Diagrams

Project Design Phase-II
Data Flow Diagram & User Stories

Date	16 October 2022
Team ID	PNT2022TMID00810
Project Name	PersonalExpense Tracker Application
Maximum Marks	4 Marks

Data Flow Diagram:



Solution & Technical Architecture

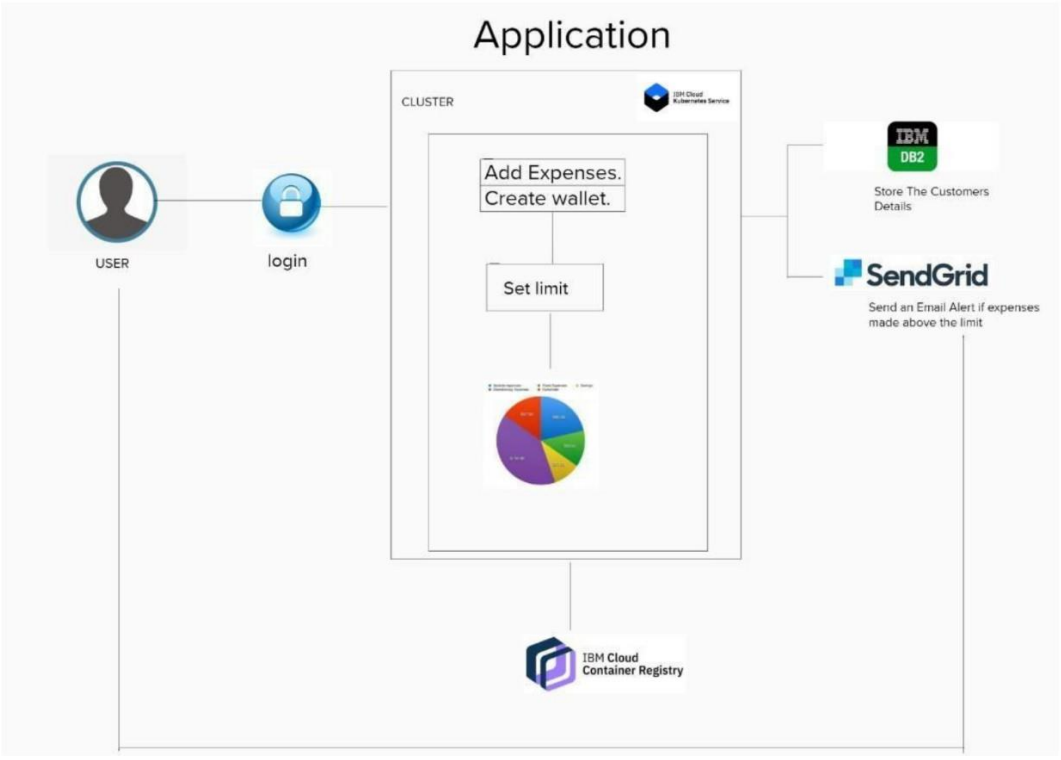
Project Design Phase-I
Solution Architecture

Solution Architecture:

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

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

Example - Solution Architecture Diagram:



Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	14 October 2022
Team ID	PNT2022TMID00810
Project Name	Personal Expense Tracker Application
Maximum Marks	4 Marks

Technical Architecture:

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

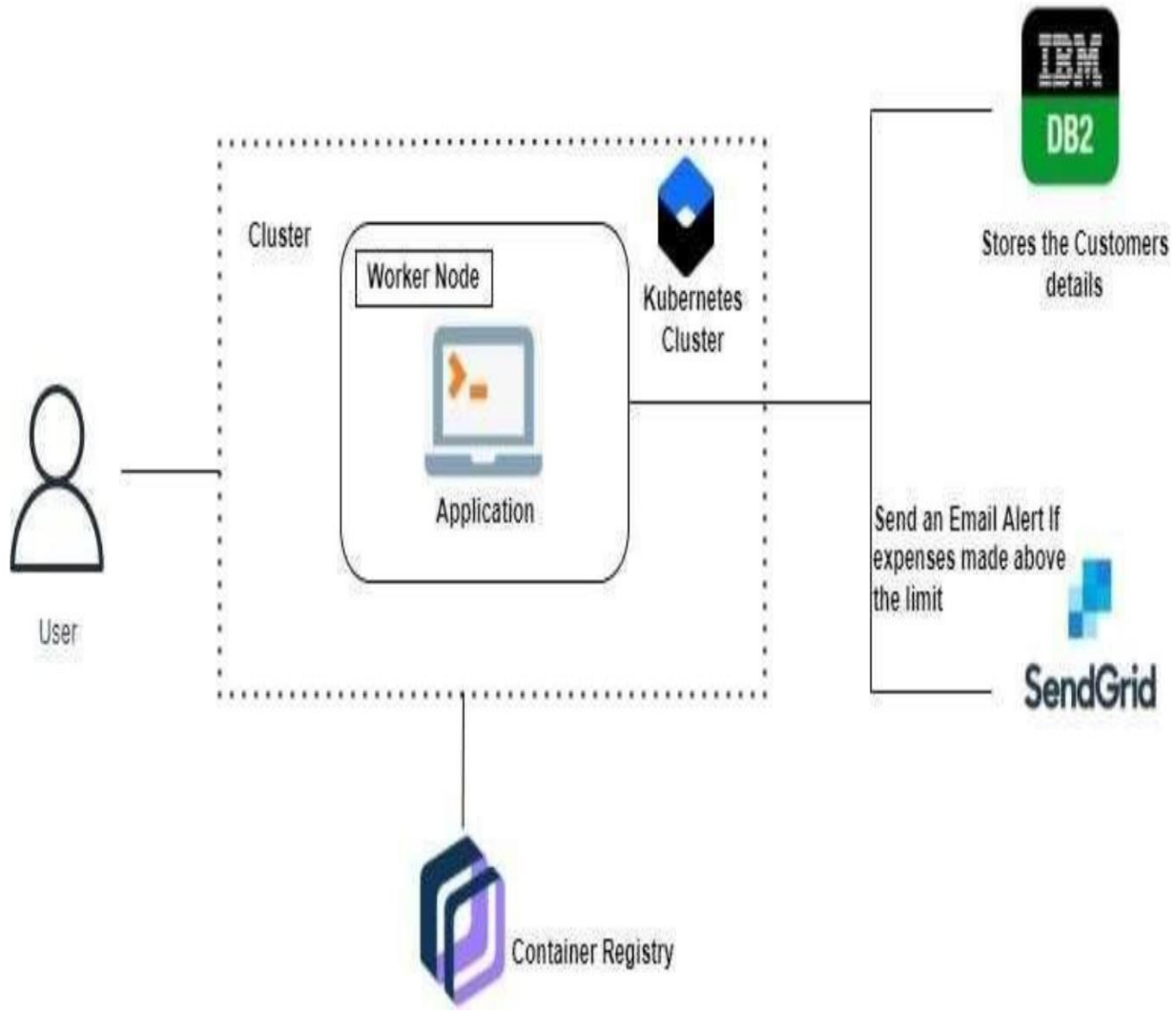


Table-1: Components & Technologies:

S.No.	Component	Description	Technology
1.	User Interface	The user can Interact with the application with use of Chatbot	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	The application contains the sign in/sign up where the user will login into the main dashboard	Java / Python
3.	Application Logic-2	Dashboard contains the fields like Add income, Add Expenses, Save Money	IBM Watson STT service
4.	Application Logic-3	The user will get the expense report in the graph form and also get alerts if the expense limit exceeds	IBM Watson Assistant,SendGrid
5.	Database	The Income and Expense data are stored in the MySQL database	MySQL, NoSQL, etc.
6.	Cloud Database	With use of Database Service on Cloud, the User data are stored in a well secured Manner	IBM DB2, IBM Cloudant etc.
7.	File Storage	IBM Block Storage used to store the Financial data of the user	IBM Block Storage or Other Storage Service or Local Filesystem

Table-2: Application Characteristics:

S.No.	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask Framework in Python is used to implement this Application	Python-Flask
2.	Security Implementations	This Application Provides high security to the user Financial data. It can be done by using the Container Registry in IBM cloud	Container Registry, Kubernetes Cluster
3.	Scalable Architecture	Expense Tracker is a life time access supplication. It's demand will increase when the user's income are high	Container Registry, Kubernetes Cluster
4.	Availability	This application will be available to the user at any part of time	Container Registry, Kubernetes Cluster
5.	Performance	The performance will be high because there will be no network traffics in the application	Kubernetes Cluster

User Stories

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 to the application by providing my email, password, and confirmingmy password.	I can access my account / dashboard	High	
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	
	Login	USN-4	As a user, I can log into the application by entering email & password	I can access the application	High	
	Dashboard	USN-5	As a user I can enter my income and expenditure details.	I can view my daily expenses	High	
Customer Care Executive		USN – 6	As a customer care executive, I can solve the log in issues and other issues of the application.	I can provide support or solution at any time 24*7	Medium	
Administrator	Application	USN – 7	As an administrator I can upgrade or update the application.	I can fix the bug which arises for the customers and users of the application	Medium	

6. PROJECT PLANNING & SCHEDULING

Sprint Planning , Estimation and Delivery Schedule:

Project Planning Phase
Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	05 November 2022
Team ID	PNT2022TMID00810
Project Name	Project – Personal Expense Tracker
Maximum Marks	8 Marks

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

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Lokesh &Maniyarasan
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application.	2	High	Muniasamy &Lokesh

Sprint-1	Dashboard	USN-3	As a user, I can login and access my dashboard and expenses page.	3	High	Maniyarsan, Lokesh &Muniasamy.
Sprint-2	User Action	USN-4	As a user, I can add an expense.	2	High	Lokesh
Sprint-2	User Action	USN-5	As a user, I can add money to my account anytime necessary.	2	Medium	Lokesh
Sprint-2	Dashboard	USN-6	As a user, I can view my dashboard to see balance remaining, the last transactions made, and where I have spent them.	3	High	Maniyarasan ,Lokesh&Muniasa my.
Sprint-3	Warning	USN-7	As a user, if I exceed my limit, I should be warned with an email.	3	Low	Lokesh

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Customization	USN-8	As a user, I should be able to set rewards and goals for myself to feel inclined to continue to spend wisely.	5	High	Maniyarasan ,Lokesh&M uniasamy.
Sprint-3	User Action	USN-9	As a user, I can set a monthly limit for my expenses.	2	Medium	Muniasamy
Sprint-4	Customization	USN-10	As a user, I can create custom categories that are given to me as a choice when I upload/update an	3	Medium	Maniyarsan
Sprint-4	Analysis	USN-11	At the end of every month, as a user, I should be able to view my monthly expenses, projections in the form of dashboards and graphs.	3	High	Muniasamy &Maniyars an
Sprint-4	Warning	USN-12	As a user, I should be able to set reminders to alert me of periodic transactions or delayed expenses	3	Medium	Lokesh

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

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	06	6 Days	24 Oct 2022	29 Oct 2022	07	- (Meet Planned Date)
Sprint-2	06	6 Days	31 Oct 2022	05 Nov 2022	07	- (Meet Planned Date)
Sprint-3	06	6 Days	07 Nov 2022	12 Nov 2022	10	- (Meet Planned Date)
Sprint-4	06	6 Days	14 Nov 2022	19 Nov 2022	09	- (Meet Planned Date)

Velocity:

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

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

Average Velocity:

Average points per sprint = (7 + 7 + 10 + 9) / 4 = 8.25

Story points per day/ Average Velocity = 8.25/6 = 1.375

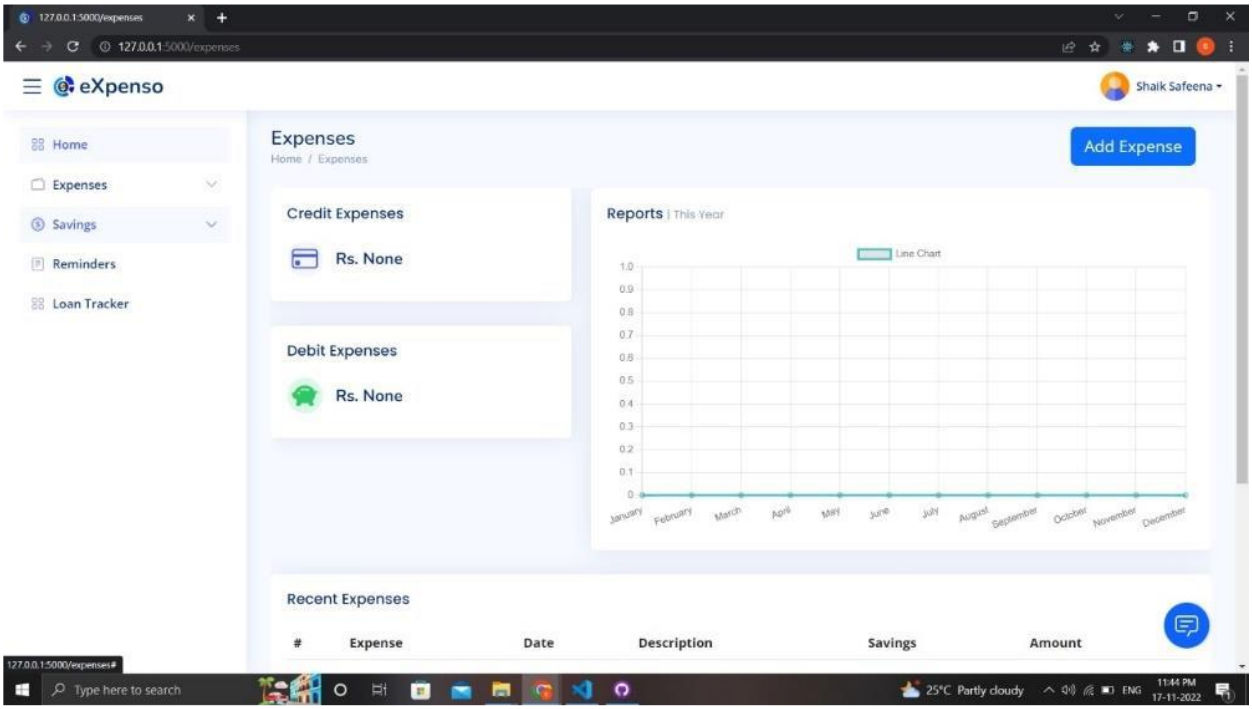
Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

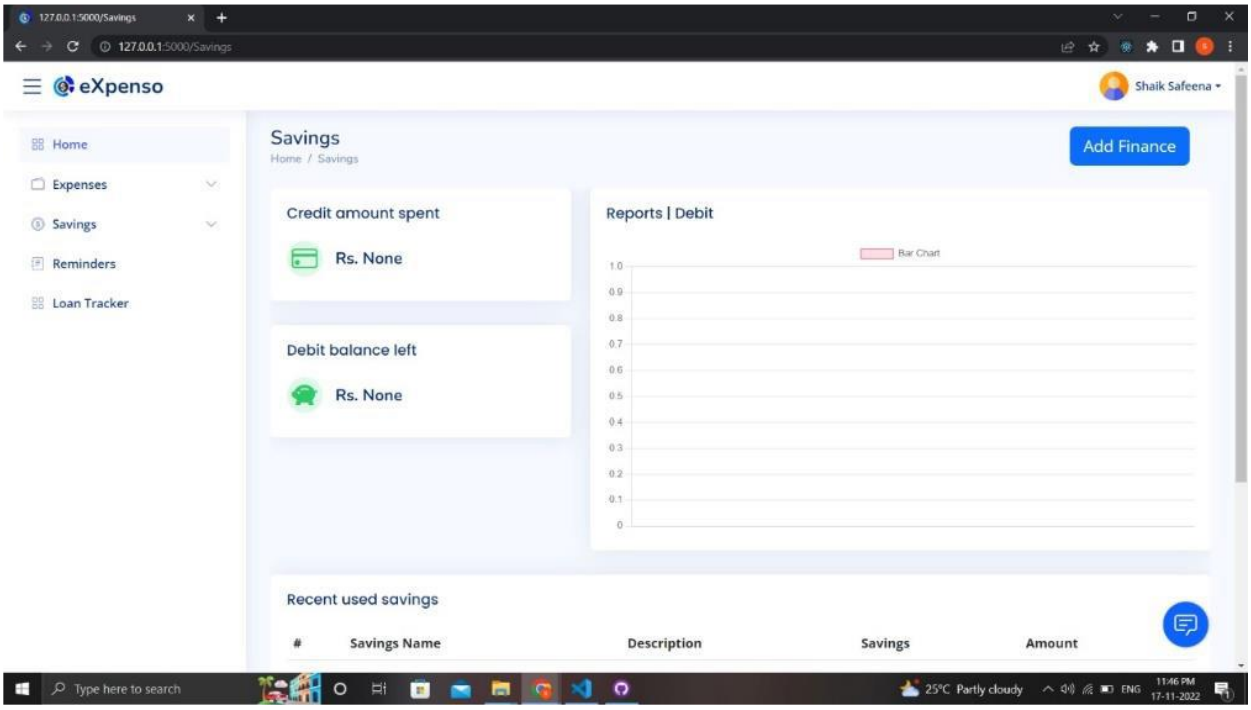
The burndown chart can only be generated once a sprint or two is completed. It currently doesn't generate a burndown chart. We will upload the same to the Jira Files directory as and when our burndown chart gets updated.

7. CODING AND SOLUTIONING

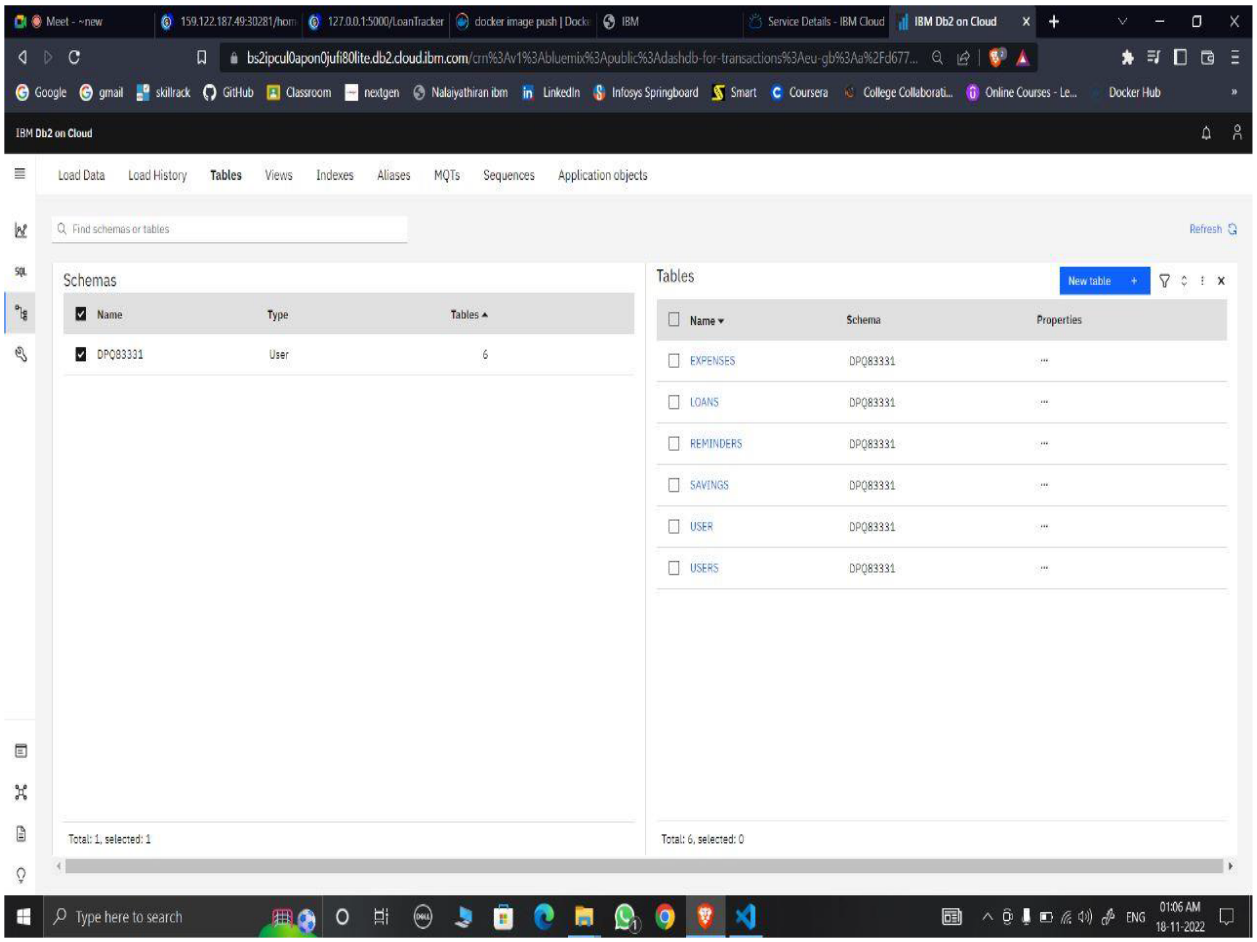
7.1 Feature 1



7.2 FEATURE 2


127.0.0.1:5000/Savings
eXpenso
Shaik Safeena
Home
Expenses
Savings
Reminders
Loan Tracker
Savings
Home / Savings
Add Finance
Credit amount spent
Rs. None
Debit balance left
Rs. None
Reports | Debit
Bar Chart
Recent used savings
Savings Name Description Savings Amount
Type here to search
25°C Partly cloudy
ENG
11:46 PM
17-11-2022

7.3 DATABASE SCHEMA



8. TESTING

8.1 TEST CASES

S.No	Test Cases	Passed/ Failed
1.	Login and Logout	Passed
2.	Profile Creation and Editing	Passed
3.	Expenses – Add, Edit and Delete	Passed
4.	Expense Analysis Module	Passed
5.	Savings – Add, Edit and Delete	Passed
6.	Savings Analysis Module	Passed
7.	Reminders – Add, Edit and Delete	Passed
8.	Loan Tracker – Add, Edit and Delete	Failed to Deploy

8.2 USER ACCEPTANCE TESTING

S.No	Test Cases	Yes/ No
1.	Keyword driven	Yes
2.	Responds in manually drafted rules	Yes
3.	Manages multiple users	Yes

4.	Conversational Paradigm	Yes
3.	Learns from real interactions	Yes
4.	Training via historical data	No
5.	Has decision-making skills	No

9. RESULTS

9.1 Performance Metrics

Cloud performance monitoring and testing tools help organizations gain visibility into their cloud environments, using specific metrics and techniques to assess performance. Efficient cloud performance is critical for maintaining business continuity and ensuring all relevant parties gain access to cloud services. This is true for basic cloud usage of public clouds and complex hybrid clouds and multi-cloud architectures. Cloud performance metrics enable you to effectively monitor your cloud resources, to ensure all components communicate seamlessly. Typically, cloud performance metrics measure input/output operations per second (IOPS), filesystem performance, caching, and autoscaling.

S. No	**Test Cases**	**Time**
1.	Error rates	1 second
2.	Response times	3 seconds
3.	Request rates	1.14 seconds
4.	Customer experience	Good

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

- Available 24/7 across the globe
- Easy tracking of savings and expenses
- User friendly dashboards
- Understandable charts and graphs
- Updated to the latest details
- Easy to setup and communicate

10.2 Disadvantages

- Direct connection with bank account could not be achieved
- Loan Tracker page could not be deployed
- Multiple profiles not available

11. CONCLUSION

This project focuses on tracking the personal finances of a user. It is implemented in a user-friendly and accessible manner to achieve all the financial asset maintenance goals of an individual. 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 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. Since the screen provides online help messages and is very user-friendly, any user will get familiarized with its usage. Module s are designed to be highly flexible so that any failure requirements can be easily added to the modules without facing many problems. The best organizations have a way of tracking and handling these reimbursements. This project also focuses on giving set reminders for users who wish to be up to date in their payments. This project gives aesthetic charts and analysis for the user's financial records.

12. FUTURE SCOPE

The future of project lies entirely on how the customers get benefitted from the interaction and the interface. We would have to make improvements in the application to make it more user-friendly. The following areas could have a serious impact on our scope:

i. Support for multiple accounts
ii. Direct Connection with bank
iii. Tips and Recommendations on where to save money.

13. APPENDIX

1. SOURCE CODE

a) app.py

```
import os
from flask import *
from database import *
from models import *
from random import *
from flask_mail import Mail, Message
from datetime import *
from time import *
database = Database()
app = Flask(_name_)
app.secret_key = "FlaskNotFoundError"
app.config['MAIL_SERVER'] = 'smtp.sendgrid.net'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USERNAME'] = 'apikey'
app.config['MAIL_PASSWORD'] = credentials.SENDGRID_API_KEY
app.config['MAIL_DEFAULT_SENDER'] = credentials.MAIL_DEFAULT_SENDER
mail = Mail(app)
#Index
@app.route('/')
def index():
return render_template('index.html')
#Signup
@app.route('/signup', methods = ['GET','POST'])
def signup():
msg=None 32
```

```
if request.method == 'POST':
name = request.form['name']
email = request.form['email']
password = request.form['password']
account = database.fetchUser(email)
if account:
flash("You are already a member, please login using your details")
return redirect('signin')
else:
if database.insertSignUpUserData(email,password,name):
msg = Message('eXpenso Registration', recipients=[email])
msg.body = 'Thank you for registering with eXpenso! Happy Managing!!'
msg.html = """<h1>Sucessfully Registered with eXpenso</h1>
<h3>Thank you for registering with eXpenso! Happy Managing!!</h3>
"""
mail.send(msg)
flash("Registration successfull...")
return redirect('signin')
else:
msg="Unable to Register!! Try again"
return render_template('signup.html',msg=msg)
#Signin
@app.route('/signin',methods = ["GET","POST"])
def signin():
invalidLogin = None
if request.method == 'POST':
email = request.form['email']
password = request.form['password']
if database.fetchUser(email):
fetchedPassword = database.fetchPassword(email)
if fetchedPassword==password:
session['email']=email 33
```

```
return redirect('home')
else:
invalidLogin="Your Password is wrong!!"
else:
invalidLogin="You have not Registered yet!!"
return render_template('signin.html',invalidLogin=invalidLogin)
@app.route('/signout')
def signOut():
if 'email' in session:
session.pop('email',None)
return redirect('/')
return redirect('/')
#Home
@app.route('/home')
def presentHome():
if 'email' not in session:
return redirect('/')
email=session['email']
user = database.fetchUser(email)
totalExpenses = database.getTotalExpenseAmount(email)
totalSavings = database.getDebitSavingsAmount(email)
expenseFilter = "year"
expenses = database.fetchExpensesPreview(email,5)
monthExpenses=database.getExpensesThisYear(email)
monthLabels=['January', 'February', 'March', 'April', 'May', 'June', 'July','August','September','October','November','December']
monthExpenseList = [0]*12
for expense in monthExpenses:
monthExpenseList[int(expense["MONTH"])-1]=expense["AMOUNT"]
totalLoanPaid = database.getTotalLoanPaid(email)
totalLoanLeft = database.getTotalLoanLeft(email) 34
```

```
reminders = database.readRemindersWithLimit(email)
reminderList = []
for reminder in reminders:
days = (date(int(reminder["YEAR"]),int(reminder["MONTH"]),int(reminder["DATE"])) - date.today()).days
label = "left"
if days<0:
continue
elif days == 0:
label = "today"
name = reminder["REMINDERNAME"]
reminderdate = reminder["DATE"]+"/"+reminder["MONTH"]
description = reminder["DESCRIPTION"]
reminderList.append([days,name,description,label,reminderdate])
return render_template('home.html',user = user,expenseFilter = expenseFilter,totalExpenses = totalExpenses, totalSavings =
totalSavings, expenses = expenses, monthLabels = monthLabels, monthExpenseList = monthExpenseList, totalLoanPaid =
totalLoanPaid , totalLoanLeft = totalLoanLeft,reminders = reminderList)
```

b) database.py

```
import credentials
import ibm_db
import ibm_boto3
from ibm_botocore.client import Config, ClientError
from datetime import *
from models import *
conn =
ibm_db.connect("DATABASE="+credentials.DB2_DATABASE_NAME+";HOSTNAME="+credentials.DB2_HOST_NAME+";P
ORT="+credentials.DB2_PORT+";SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID="+credentials.DB2_UI
D+";PWD="+credentials.DB2_PWD+"","")
cos = ibm_boto3.resource("s3",
ibm_api_key_id=credentials.COS_API_KEY_ID,
ibm_service_instance_id=credentials.COS_INSTANCE_CRN,
config=Config(signature_version="oauth"), 35
```

```
endpoint_url=credentials.COS_ENDPOINT
)
class Database:
def _init_(self) -> None:
pass
def fetchUser(self,email):
sql = "SELECT email,name,phone,country FROM user WHERE email=?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
if account:
user = User(account["EMAIL"],account["NAME"],account["PHONE"],account["COUNTRY"])
return user
return None
def fetchPassword(self,email):
try:
sql = "SELECT password FROM user WHERE email = ?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
if account:
return account["PASSWORD"]
else:
return False
except:
return False
def insertSignUpUserData(self,email,password,name):
try:
insert_sql = "INSERT INTO user(email,password,name) VALUES (?,?,?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prep_stmt, 1, email)
ibm_db.bind_param(prep_stmt, 2, password)
ibm_db.bind_param(prep_stmt, 3, name)
ibm_db.execute(prep_stmt)
except:
print("error")
return False
return True
def updateUserData(self,email,name,country,phone):
try:
sql = "update user set name = ? , country = ?, phone = ? where email = ?;"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,name)
ibm_db.bind_param(stmt,2,country)
ibm_db.bind_param(stmt,3,phone)
ibm_db.bind_param(stmt,4,email)
ibm_db.execute(stmt)
except:
return False
return True
def updatePassword(self,email,password):
try:
sql = "update user set password = ? where email = ?;"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,password)
ibm_db.bind_param(stmt,2,email)
```

```
ibm_db.execute(stmt)
except:
return False
return True
def fetchExpensesPreview(self,email,limit=10):
sql ="SELECT expensename,date,month,year,expenses.description,savingsname,savingstype,expenses.amount 37
```

```
FROM expenses join savings on expenses.savingsid=savings.savingsid WHERE expenses.email=? order by expenseid desc limit ?;"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.bind_param(stmt,2,limit)
ibm_db.execute(stmt)
expense = ibm_db.fetch_both(stmt)
expenseList = []
while expense != False:
expenseList.append(expense)
expense = ibm_db.fetch_both(stmt)
return expenseList
```

GIT HUB LINK :

https://github.com/IBM-EPBL/IBM-Project- 7678-1658895730