PROJECT REPORT-PERSONAL EXPENSE TRACKER
TEAM ID:PNT2022TMID07305

INTRODUCTION:

Personal Income Expense Tracker is to easily manage your finance by recording your monthly incomes and expenses. Sometimes at the end of every month, we usually find a shortage of money due to our unaccounted expenses or our bad spending habits. It is necessary to keep track of our incomes and expenses.

1.1 Project Overview

Personal Expense Tracker (PET) is a daily expense management system which is specially designed for non- salaried and salaried personnel for keeping track of their daily expenditure with easy and effective way . Personal expense or finance entails all the financial decisions and activities that a Finance app makes your life easier by helping you to manage your finances efficiently.

A personal finance app will not only help you with budgeting and accounting but also give you helpful insights about money management. Personal expense or finance applications will ask users to add their expenses and based on their expenses wallet balance will be updated which will be visible to the user.

Also, users can get an analysis of their expenditure in graphical forms. They have an option to set a limit for the amount to be used for that particular month if the limit is exceeded the user will be notified with an email alert.

1.2 Purpose

The purpose of the project is to help you control your expenses in order to manage the proper spending of money. About the System The Expense Tracker App was created in a HTML web browser that use JavaScript to give user a great interactive experience when using an app.

When you track your spending, you know where your money goes and you can ensure that your money is used wisely. Tracking your expenditures also allows you to understand why you're in debt and how you got there. This will then help you design a befitting strategy of getting out of debt. Budgeting ensures you're not spending more than you're making, allowing you to plan for short- and long-term expenses. It's an easy, helpful way for people with all types of income and expenses to keep their finances in order.

LITERATURE SURVEY

Literature review was carried out to gain knowledge and improve the skills needed to complete this project. This chapter shows the different techniques that have been implemented.

2.1 Existing Problem

An expense tracker is a software or application that helps to keep an accurate record of your money inflow and outflow. Many people in India live on a fixed income, and they find that towards the end of the month they don't have sufficient money to meet their needs.

2.2 References

[1] http://expense-manager.com/how-expense software/

[2] https://www.splitwise.com/terms

[3] http://code.google.com/p/socialauthandroid/wiki/Facebook

[4] http://code.google.com/p/socialauth-android

[5] Developer.android.com

[6] http://www.appbrain.com/app/expensemanager/com.expensemanager

[7] https://www.xpenditure.com/en?

[8] http://expense-manager.com/how-expense software/

[9] Donn Felker, "Android Application Development for Dummies", published by For Dummies, 2010.

[10] Ed Burnette, "Hello, Android: Introducing Google's Mobile Development Platform", published by Pragmatic Bookshelf, 2009.

Lee, "Beginning Android Application Development", Published by WroxPress, 2011.

2.3 Problem Statement Definition

It is tough to keep track of all the financial decisions and activities that a person makes. Traditional expense tracking methods are inconvenient and unreliable. In order to get a quick overview about your total incomes and expenses and control spending , its convenient to digitize the process by having a personal expense tracker.

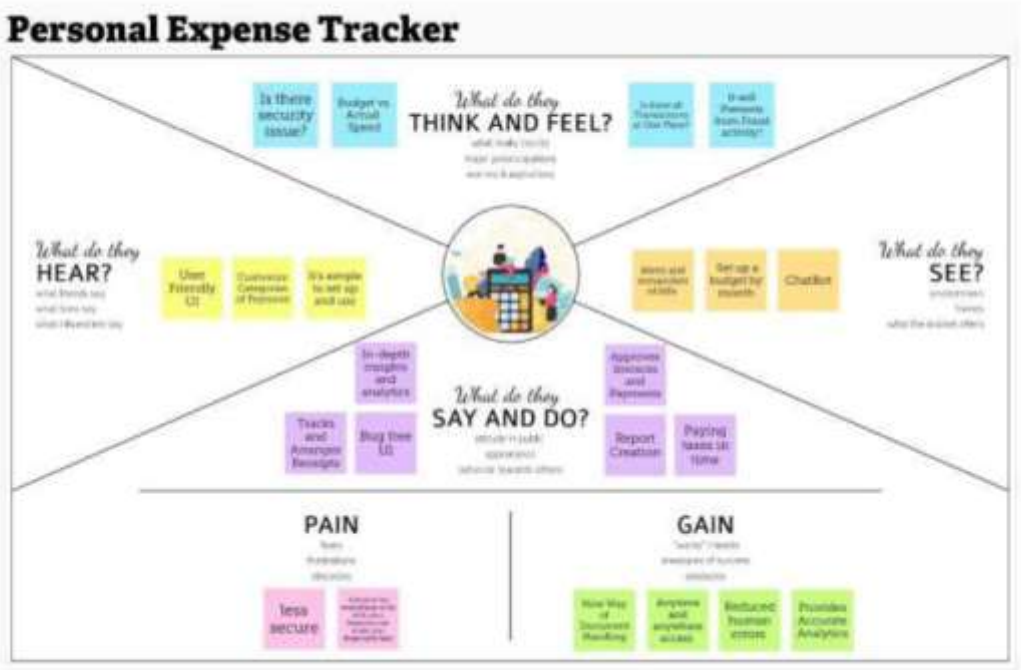
Who does the problem affect?	Working individuals, students and budget conscious consumers.
What are the boundaries of the problem?	Limited features to provide for expense tracking.
What is the issue?	To be vigilant about the expense spent increases financial stress. Being indecisive about the finances may result in less financial security and exceed the budget.
When does this issue occur?	When people are not able to track their expenses properly.
Where is the issue occurring?	In daily life of employees as well as students.
Why is it important that we fix the problem?	Fixing this issue will help users to better plan their budget and lead to financial well-being.

- Sophie, who is a homemaker, finds it hard to control her desire to shop. To stop herself from overindulging in impulsive purchases, she needs to track her expenses and holdherself accountable.
- Sam is a high school student who usually gets a limited allowance from his parents. Tracking his expenses and good budgeting technique allows him to spend on his regular expenses as well as on himself.
- Percy, who is a novice budgeter, finds it tedious to track and manage the expenses amongst his busy schedule. Prioritizing his expenses will help him to curtail his unnecessary expenditures.

IDEATION & PROPOSED SOLUTION

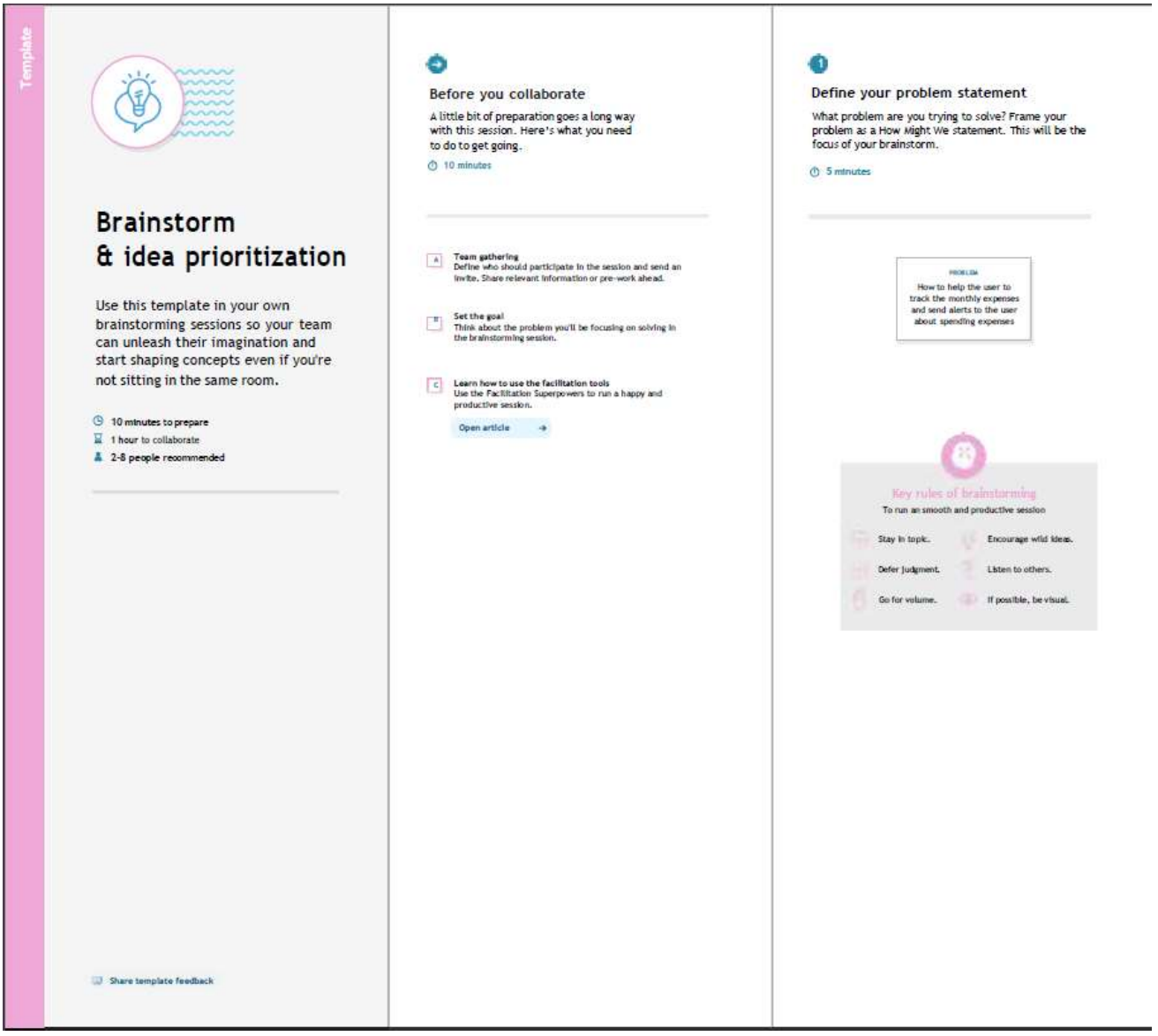
3.1Empathy Map Canvas

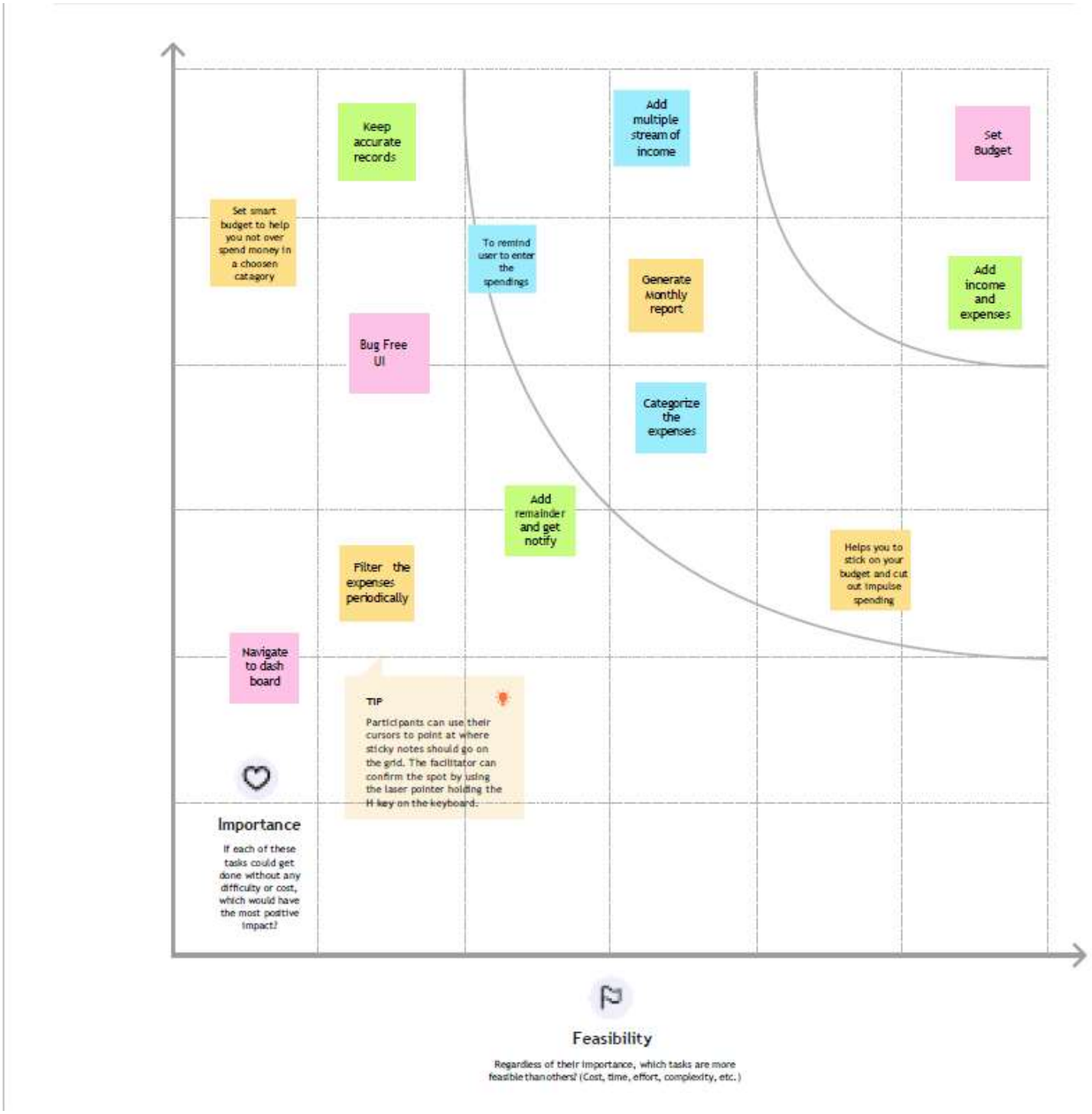
Personal Expenses Tracker Application



3.1 Ideation & Brainstorming

3.1.a Brainstorm





PROBLEM STATEMENT

Their own system are used by many business to keep track of their income and expenses because they believe this to be the most important factor in how well their operations are going. As people are unaware of the proper applications to protect their privacy and lack the capacity to make decisions it is a good habit to keep track of daily expenses and income. Even they are still using the old-fashioned note-taking techniques to do so. There is a constant overload to rely on the daily entry of the expenditure and total estimation up until the end of the month because there isn't a complete tracking system.

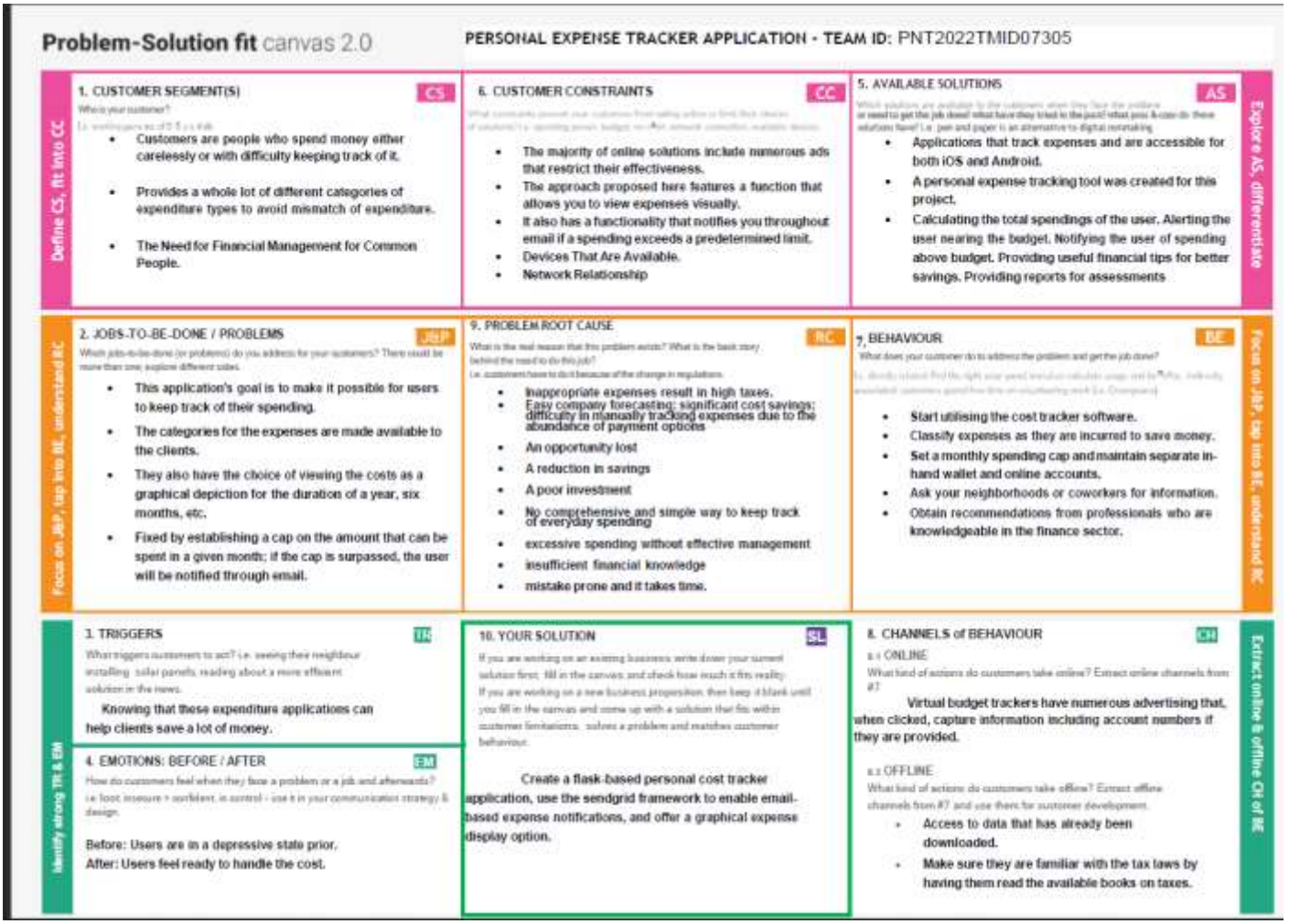
Who does the problem affect?	Investors, savers, big spenders, debtors, consumers on a tight budget, and shoppers.
What are the boundaries of the problem?	Expense tracking software for employees, students, and regular people.
What is the issue	Being watchful of expenses incurred increases financial strain. Making rash financial decisions could decrease financial security and cause you to go over your budget.
When does this issue occur?	When employing improper budgeting methods. When you don't keep track of your expenses, you can't determine how much was actually spent.
Where is the issue occurring	Working people who struggle to keep track of their expenses.
Why is it important that we fix the problem?	By designating the income for spending, saving, and giving, resolving this problem promotes accountability and encourages financial planning with purpose. This promotes monetary stability.

3.2 Proposed Solution

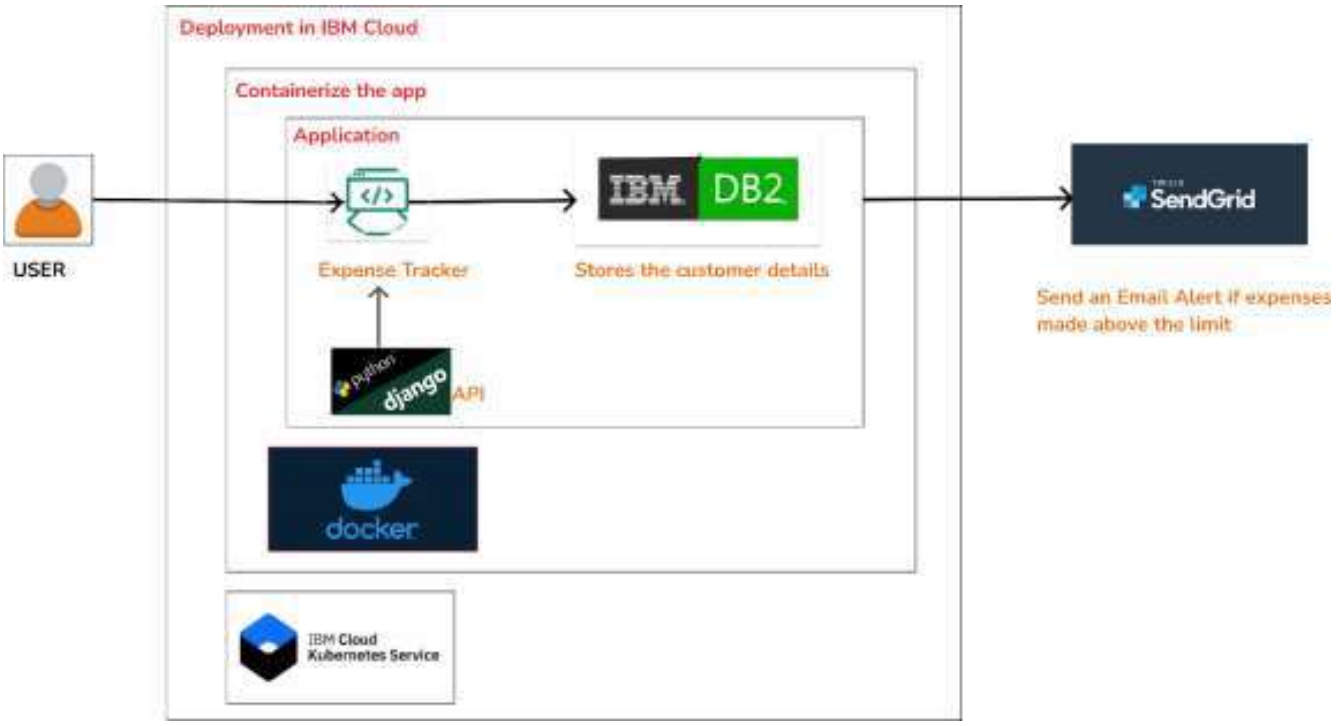
Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem is that people find it difficult to maintain track of their monthly expenses and to avoid obsessive spending.
2.	Idea / Solution description	The user can enter income and expenses into the personal cost tracker programme. As a result, the expense wallet is updated. It can obtain the graphical breakdown of the expense. The user is also notifiedif the budget's monthly maximum is surpassed.
3.	Novelty / Uniqueness	The personal expense tracker application helps the user not only in budgeting and accounting; it also provides the insights about money management through the analysis. The user also gets notified if the monthly limit is exceeded.
4.	Social Impact / Customer Satisfaction	The personal spending tracker programme not only assists the user in budgeting and accounting, but it also provides insights into money management through analysis. If the monthly limit is surpassed, the user is also notified.
5.	Business Model (Revenue Model)	The application may have a free and premium version, with the user having the option to upgrade to the premium version to gain access to additional features. Furthermore, the premium version may be ad-free.
6.	Scalability of the Solution	This application is not only for personal use, but it can also be used in the field of business organization.

3.3 Problem Solution fit



SOLUTION ARCHITECTURE



CHAPTER 4 FUNCTIONAL REQUIREMENT

4.1 Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration Form for collecting details.
FR-2	User Login	Enter username and password.
FR-3	Forget Password	Resetting the password by sending an OTP to user's mail.
FR-4	Calendar	Personal expense tracker application must allow user to add the data to their expenses.
FR-6	Dashboard	User can add the expense and can evaluate them using the provided options.
FR-5	Expense Tracker	This application must graphically represent the expense like report.
FR-6	Report generation	Report must be generated in a graphical form.
FR-7	Category	This application shall allow users to add categories of their expenses.
FR-8	Result Page	Show the user result.

5.1 Non Functional Requirements

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

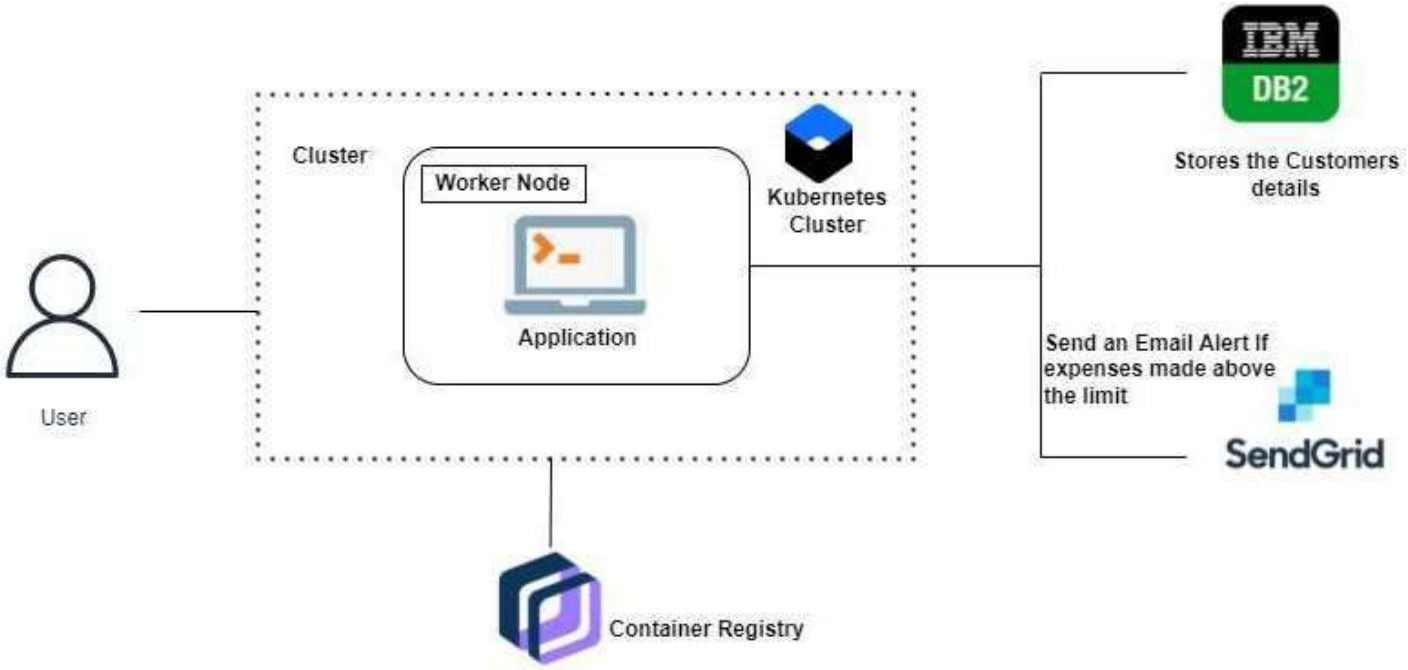
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Most web browsers allow users to access the application. The application's attractive and detailed user interface makes it easier to use. It makes it easier for you to monitor your earnings and expenses.
NFR-2	**Security**	Customers are required to set up an account for themselves using their email, which is secured by a longer password of six characters. This application might prevent you from engaging in online crimes.
NFR-3	**Reliability**	Each data record is kept in an effective database schema that is well built. No chance of data loss exists.
NFR-4	**Performance**	Expense types include categories and an option. The system's throughput is increased thanks to the lightweight database support.

PRODUCT DESIGN

5.1 Data Flow Diagrams

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

5.2Technical Architecture



5.3 User Stories

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

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user & web user)	Registration	USN-1	As a user, I can register for the application by entering my email, and password, and confirming my password.	I can access my account/dashboard	High	Sprint-1
		USN-2	As a user, I will receive a confirmation email once I have registered for the application	I can receive a confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook3	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through a Google account.	I can register & access the dashboard with a Google Account login.	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering my email & password	I can access the application.	High	Sprint-1
	Dashboard	USN-6	As a user, I can see the daily expenses and expenditure details.	I can view the daily expenses and add the expense details.	High	Sprint-1
Customer Care Executive		USN-7	As a customer care executive ,it is easy to solve the problem that faced by the customers.	I can provide support to customers at any time 24*7.	Medium	Sprint-1
Administrator	Application	USN-8	As an administrator, I can update the application and provide necessary upgrades.	I can fix any bugs raised by customers and upgrade the application.	Medium	Sprint-1

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

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	Ansu K Sam
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Aurno Shyrin
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Dinesh Kumar E
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Chandru S
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Ansu K Sam
Sprint-3	Dashboard	USN-6	As a user I can see the expenditure details on the application	3	High	Dinesh Kumar E
Sprint-3	Limits	USN-6	As a user I can set my monthly expense limit so that I receive a mail on exceeding that	4	High	Aurno Shyrin
Sprint-4	Reports	USN-6	As a user I can view the graphical form of my expenses category wise	5	Medium	Ansu K Sam

6.2 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	6	6 Days	24 Oct 2022	29 Oct 2022	6	29 Oct 2022
Sprint-2	2	6 Days	31 Oct 2022	05 Nov 2022	2	05 Nov 2022
Sprint-3	7	6 Days	07 Nov 2022	12 Nov 2022	7	12 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022	5	19 Nov 2022

7.1 Feature 1

1. Expense and revenue tracking.
2. Managing transaction receipts and records.
3. Paying taxes in time.
4. Processing payment and invoices.
5. Create in-depth reports.

TESTING

8.1 Test Cases

DASHBOARD

apiVersion: v1
kind:
ServiceAccount
metadata:
name: admin-
user
namespace:
kubernetes-
dashboard

---

apiVersion: v1
kind: Secret
metadata:
name: admin-
user-token
namespace:
kubernetes-
dashboard
annotations:

kubernetes.io/serv
ice-account.name:
admin-user

type:
kubernetes.io/serv
ice-account-token

---

apiVersion:
rbac.authorization

.k8s.io/v
1kind:

ClusterRoleBinding

metadata:
name: admin-
user

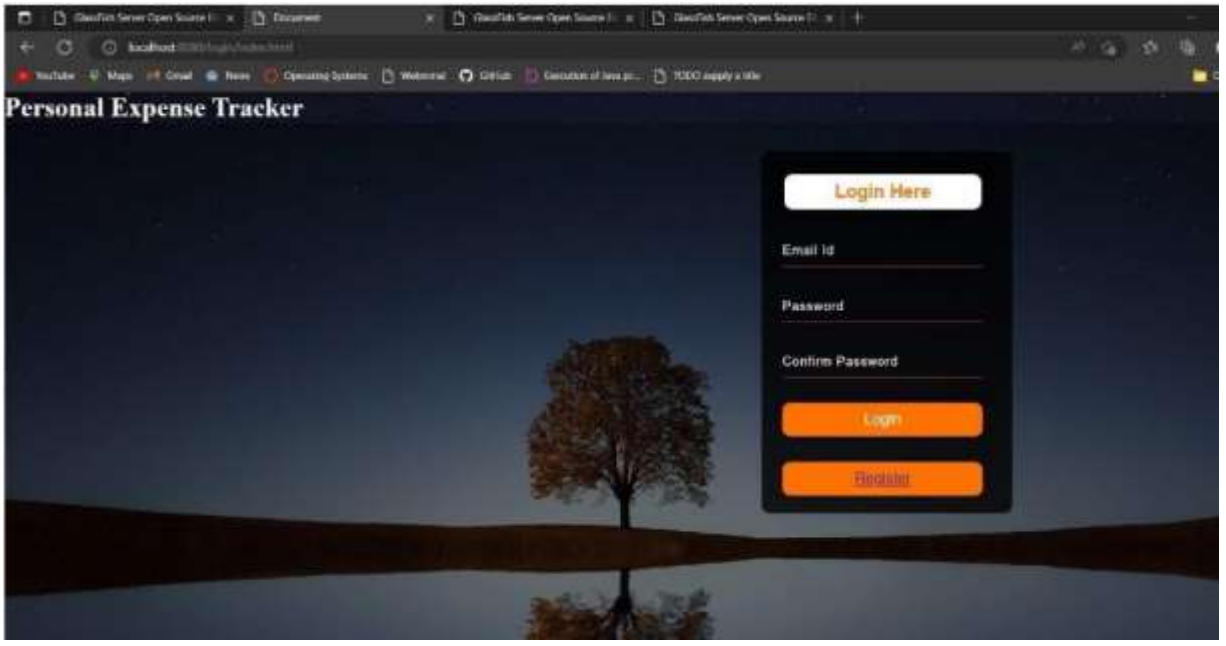
roleRef: apiGroup:
rbac.authorization

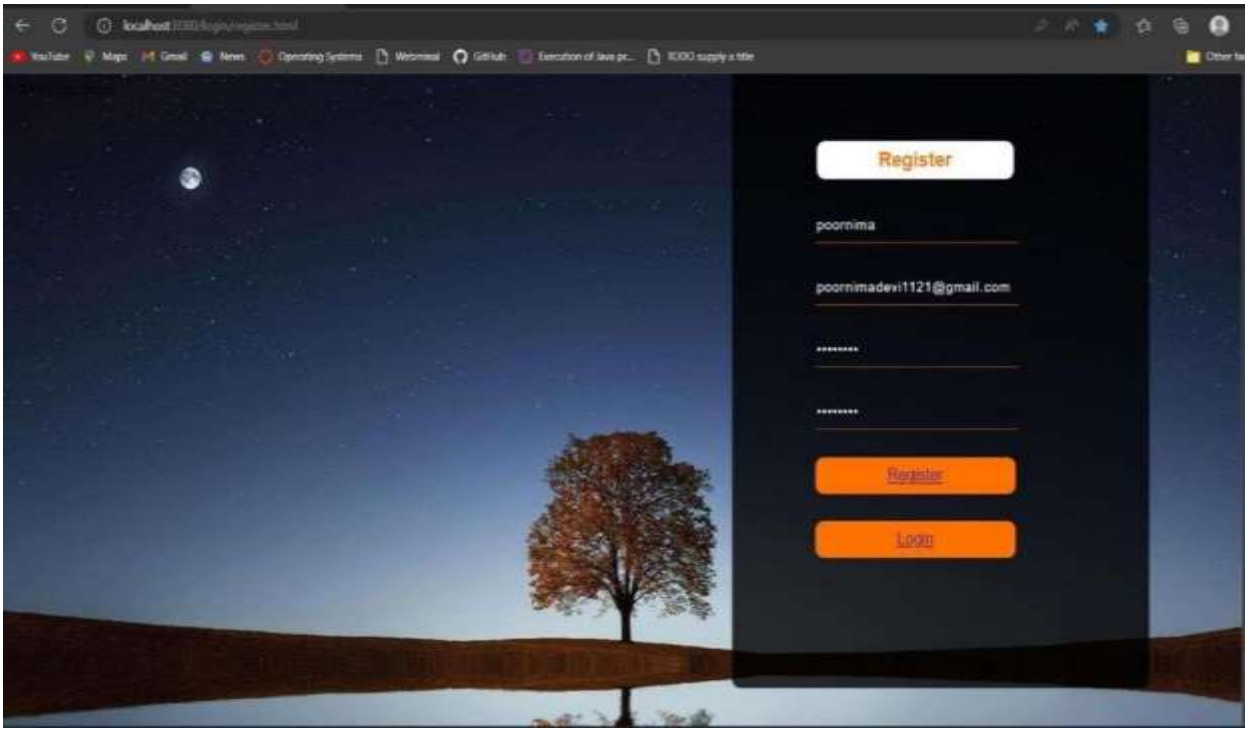
.k8s.io

```
  kind: ClusterRole
name: cluster-
admin

subjects:

- kind:
ServiceAccount
name: admin-
user
namespace:
kubernetes-
dashboard
```

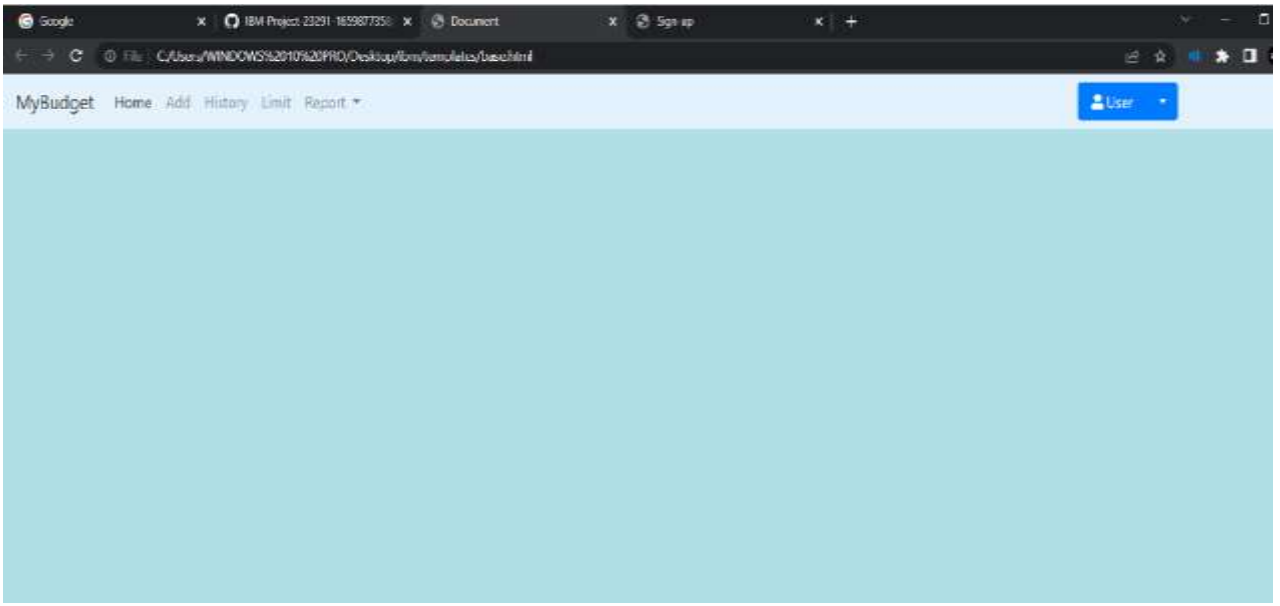

Personal Expense Tracker
Login Here
Email Id
Password
Confirm Password
Login
Register


Register
poornima
poornimadevi1121@gmail.com
Register
Login

BUDGET TRACKING

LET'S START THE JOURNEY

Let's Begin



9.1 Performance Metrics

RESULTS

An application can be a very powerful tool for businesses if once the app becomes a success. However, the success of an app is measured through numbers, metrics, and analytics. Developing an app takes quite a lot, so once you've dedicated much time, money, and effort to the process, it's mandatory to measure mobile app performance.

10.1 Advantages

1. Maintaining Financial Control

When if comes to personal finance, being out of control is not something anybody would strive for. There's nothing financially worse than feeling like you don't have any idea what's going on with your money.

The good news is, when you make an effort to record every financial transaction you make, you are essentially, taking the reins on anything and everything involving your money. At any one time, you will know exactly how much money is sitting in your bank account, and how much you can spend.

In other words, when you track your expenses, you take complete control over your finances.

Holding Yourself Accountable

If you have any plans on saving, investing, getting out of debt, or building wealth, what is holding you accountable. I mean, we can all set financial goals, and have financial dreams, but if you aren't tracking your expenses, there is nothing to hold you accountable when you make a bad financial decision. 1. Susceptible to costly human errors

Did you know that up to 9 out of 10 spreadsheets consist of human errors?

Unfortunately, even the smallest of mistakes in a spreadsheet can cause catastrophic consequences. Fidelity Magellan Fund once suffered a $2.6 billion overstatement when an accountant accidentally omitted the minus sign on a net capital loss of $1.3 billion.

There is always a greater chance of human error with manual processes, especially when it comes to complex data sets, such as those involved with expense management. Failure to accurately track you company's expenditure and pay invoices on time can wreak havoc on your business's bottom line.

2. Lack of collaboration and access

Because Excel spreadsheets are a single file, only one user at a time may access and modify the data. It can also be challenging to collaborate with other departments because

you have to manually share or email a copy of the relevant spreadsheet with your colleagues.

When it comes to expense management data, however, these Excel spreadsheets are frequently shared and proofed across numerous teams and departments. To guarantee that everyone is viewing the current version, users must be rigorous about version control and sharing when updates are made.

3. Time-consuming manual processes

The quantity of expense management data you need to review, analyse, and track will grow as your business evolves. The only way to validate your data when using Excel spreadsheets, however, is to manually double check and re-enter any inaccurate information. This is a time-consuming and labour-intensive task.

As a result, Excel spreadsheets slow workers down and reduce accuracy by requiring them to perform repetitive processes that could be simplified or automated using expense management and invoicing software.

4. Inaccuracy leads to slower decision making

There's no denying that manual processes which increase the chances of inaccuracy lead to slower decision making within companies. Extracting expense data and invoices from different departments, as well as consolidating them and summarising the information, is incredibly time consuming.

Because spreadsheets are prone to inaccuracies, everyone involved in processing the information must double-check the data as much as possible, which can further slow the process.

5. Lack of version control

The sharing of Excel spreadsheets from team to team might lead to concerns with the data'sversion and validity. You should consider who had the most recent access to the data. Who did what to the spreadsheet and when? Can you confirm that the calculations are correct? If you don't trust the answers, you may need to start all over again

6. Data isn’t updated in real-time

Excel spreadsheets don’t update in real-time, so each update requires manual input. Because Excel spreadsheets can be difficult to modify, they are usually updated at the end of the day or every few days. Typically, this entails keeping daily paper records and then manually entering them to update the Excel spreadsheet at a later date. Not only is this a waste of time, but it also raises the likelihood of data being entered inaccurately or decisions being made based on out-of-date information.

7. Increased potential to lose important data

If a spreadsheet owner is unfamiliar with best practices for data storage and backup, they might keep just one version of their spreadsheet in a single location, such as on their desktop.

In the event of a technical issue, however, there’s no guarantee of complete data recovery, meaning a company could lose all of their vital data in a split-second.

- Improved customer service
- Cloud-based solution
- Order Fulfillment
- Harness Customer Loyalty and Retention

10.2 Disadvantages

1. Susceptible to costly human errors

Did you know that up to 9 out of 10 spreadsheets consist of human errors?

Unfortunately, even the smallest of mistakes in a spreadsheet can cause catastrophic consequences. Fidelity Magellan Fund once suffered a $2.6 billion overstatement when an accountant accidentally omitted the minus sign on a net capital loss of $1.3 billion.

There is always a greater chance of human error with manual processes, especially when it comes to complex data sets, such as those involved with expense management. Failure to accurately track you company's expenditure and pay invoices on time can wreak havoc on your business's bottom line.

2. Lack of collaboration and access

Because Excel spreadsheets are a single file, only one user at a time may access and modify the data. It can also be challenging to collaborate with other departments because you have to manually share or email a copy of the relevant spreadsheet with your colleagues.

When it comes to expense management data, however, these Excel spreadsheets are frequently shared and proofed across numerous teams and departments. To guarantee that everyone is viewing the current version, users must be rigorous about version control and sharing when updates are made.

3. Time-consuming manual processes

The quantity of expense management data you need to review, analyse, and track will grow as your business evolves. The only way to validate your data when using Excel spreadsheets, however, is to manually double check and re-enter any inaccurate information. This is a time-consuming and labour-intensive task.

As a result, Excel spreadsheets slow workers down and reduce accuracy by requiring them to perform repetitive processes that could be simplified or automated using expense management and invoicing software.

4. Inaccuracy leads to slower decision making

There's no denying that manual processes which increase the chances of inaccuracy lead to slower decision making within companies. Extracting expense data and invoices from different departments, as well as consolidating them and summarising the information, is incredibly time consuming.

Because spreadsheets are prone to inaccuracies, everyone involved in processing the information must double-check the data as much as possible, which can further slow the process.

5. Lack of version control

The sharing of Excel spreadsheets from team to team might lead to concerns with the data's version and validity. You should consider who had the most recent access to the data. Who did what to the spreadsheet and when? Can you confirm that the calculations are correct? If you don't trust the answers, you may need to start all over again.

6. Data isn't updated in real-time

Excel spreadsheets don't update in real-time, so each update requires manual input. Because Excel spreadsheets can be difficult to modify, they are usually updated at the end of the day or every few days. Typically, this entails keeping daily paper records and then manually entering them to update the Excel spreadsheet at a later date. Not only is this a waste of time, but it also raises the likelihood

of data being entered inaccurately or decisions being made based on out-of-date information.

7. Increased potential to lose important data

If a spreadsheet owner is unfamiliar with best practices for data storage and backup, they might keep just one version of their spreadsheet in a single location, such as on their desktop.

CONCLUSION

Taking proper care of our record is crucial in every business, no matter how big or little, we must understand. We must educate ourselves about the idea of effective inventory management and its applications because we can see that managers do not fully grasp it. A company's inventory management system is one of the reasons for its failure. Many customs to combat failure are present, and we can start from this point. Modern technologies can support us in managing and keeping an eye on our inventory. We may learn, put new ideas into practice, and assess our company**.**

FUTURE SCOPE

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

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

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

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

APPENDIX

SOURCE CODE:

REGISTRATION PAGE

```
<!DOCTYPE html>

<html>

<head>

<title>Login Form</title>

<link rel="stylesheet" type="text/css" href="..\static\css\login.css">

<link
href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

<script src="https://kit.fontawesome.com/a81368914c.js"></script>

<meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<body >

<img class="wave" src="..\static\images\wave.png">

<div class="container">

<div class="img">

<div id="png"></div>

</div>

<div class="login-content">

<form action='homepage.html' method="POST">

<div class="msg"></div>

<h2 class="title">Welcome</h2>

<div class="input-div one">
```

```
<div class="i">
<i class="fas fa-user"></i>
</div>
<div class="div">
<h5>Username</h5>
<input type="text" name="username" class="input" required>
</div>
</div>
<div class="input-div pass">
<div class="i">
<i class="fas fa-lock"></i>
</div>
<div class="div">
<h5>Password</h5>
<input type="password"  name="password" class="input" required>
</div></div>
<a href="#">Forgot Password?</a>
<input type="submit" class="btn" value="Login">
<span>OR</span>
<div><b>Login with</b></div>
<div>
<ul>
<li><a href="#"><i class="fab fa-facebook" aria-hidden="true"></i></a></li>
<li><a href="#"><i class="fab fa-twitter" aria-hidden="true"></i></a></li>
<li><a href="#"><i class="fab fa-google"  aria-hidden="true"></i></a></li>
```

```
<li><a href="#"><i class="fab fa-linkedin" aria-hidden="true"></i></a></li>

<li><a href="#"><i class="fab fa-instagram" aria-hidden="true"></i></a></li>

</ul>

</div>

<div class="app" ><b>Don't have an account?</b><a id="app1"
href="signup.html">REGISTER</a></div>

</form>

</div>

</div>

<script type="text/javascript" src="..\static\js\login.js"></script>

</body>

</html>
```

The python code to Connect with DB

```
from flask import Flask, render_template, request, redirect, session

import ibm_db

import re

app = Flask(__name__)

app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-4641-
8aba-
9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SE
CURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mbs46040;PWD
=MIEpZ1DoqwMRpGvs",'','')

#HOME--PAGE

@app.route("/home")

def home():

    return render_template("homepage.html")
```

```
@app.route("/")
def add():
   return render_template("home.html")
#SIGN--UP--OR--REGISTER
@app.route("/signup")
def signup():
   return render_template("signup.html")
@app.route('/register', methods =['GET', 'POST'])
def register():
   msg = ''
   if request.method == 'POST' :
      username = request.form['username']
      email = request.form['email']
      password = request.form['password']
      cursor = mysql.connection.cursor()
      cursor.execute('SELECT * FROM register WHERE username = % s',
(username, ))
      account = cursor.fetchone()
      print(account)
      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 = 'name must contain only characters and numbers !'
      else:
         cursor.execute('INSERT INTO register VALUES ( % s, % s, % s)', (username,
email,password))
         mysql.connection.commit()
```

```
            msg = 'You have successfully registered !'
            return render_template('signup.html', msg = msg)
 #LOGIN--PAGE


@app.route("/signin")
def signin():
    return render_template("login.html")
@app.route('/login',methods =['GET', 'POST'])
def login():
    global userid
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        cursor = mysql.connection.cursor()
        cursor.execute('SELECT * FROM register WHERE username = % s AND
password = % s', (username, password ),)
        account = cursor.fetchone()
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account[0]
            userid=  account[0]
            session['username'] = account[1]
            return redirect('/home')
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)
```

```
#ADDING--- DATA
@app.route("/add")
def adding():
    return render_template('add.html')
@app.route('/addexpense',methods=['GET', 'POST'])
def addexpense():
    date = request.form['date']
    expensename = request.form['expensename']
    amount = request.form['amount']
    paymode = request.form['paymode']
    category = request.form['category']
    cursor = mysql.connection.cursor()
    cursor.execute('INSERT INTO expenses VALUES ( % s, % s, % s, % s, % s, % s)',
(session['id'] ,date, expensename, amount, paymode, category))
    mysql.connection.commit()
    print(date + " " + expensename + " " + amount + " " + paymode + " " +
category)
    return redirect("/display")
#DISPLAY-- graph
@app.route("/display")
def display():
    print(session["username"],session['id'])
    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM expenses ORDER BY date
DESC'.format(str(session['id'])))
    expense = cursor.fetchall()
    return render_template('display.html' ,expense = expense)
```

```
#delete---the--data
@app.route('/delete/<string:id>', methods = ['POST', 'GET' ])
def delete(id):
    cursor = mysql.connection.cursor()
    cursor.execute('DELETE FROM expenses WHERE userid = %s', (id,))
    mysql.connection.commit()
    print('deleted successfully')
    return redirect("/display")

#UPDATE---DATA

@app.route('/edit/<id>', methods = ['POST', 'GET' ])
def edit(id):
   cursor = mysql.connection.cursor()
   cursor.execute('SELECT * FROM expenses WHERE userid = %s', (id,))
   row = cursor.fetchall()
   print(row[0])
   return render_template('edit.html', expenses = row[0])
@app.route('/update/<id>', methods = ['POST'])
def update(id):
  if request.method == 'POST' :
     date = request.form['date']
     expensename = request.form['expensename']
     amount = request.form['amount']
     paymode = request.form['paymode']
     category = request.form['category']
```

```
    cursor = mysql.connection.cursor()
    cursor.execute("UPDATE `expenses` SET `date` = % s , `expensename` = % s , `amount` = % s, `paymode` = % s, `category` = % s WHERE `expenses`.`userid` = % s ",(date, expensename, amount, str(paymode), str(category),id))
    mysql.connection.commit()
    print('successfully updated')
    return redirect("/display")


#limit
@app.route("/limit" )
def limit():
     return redirect('/limitn')
@app.route("/limitnum" , methods = ['POST' ])
def limitnum():
   if request.method == "POST":
       number= request.form['number']
       cursor = mysql.connection.cursor()
       cursor.execute('INSERT INTO limits VALUES (% s, % s) ',(session['id'], number))
       mysql.connection.commit()
       return redirect('/limitn')
@app.route("/limitn")
def limitn():
   cursor = mysql.connection.cursor()
   cursor.execute('SELECT limits FROM limits ORDER BY `limits`.`id` DESC LIMIT 1')
   x= cursor.fetchone()
   s = x#[0]



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

```
#REPORT


@app.route("/today")
def today():
    cursor = mysql.connection.cursor()
    print ("HI")
    #cursor.execute('SELECT TIME(date) , amount FROM expenses WHERE userid = {0} AND DATE(date) = DATE(NOW()) '.format(str(session['id'])))
    cursor.execute('SELECT TIME(date)  , amount FROM expenses  WHERE userid = %s AND DATE(date) = DATE(NOW()) ',(id,))


    texpense = cursor.fetchall()
    print(texpense)


    cursor = mysql.connection.cursor()
    print("HIII")
    #cursor.execute('SELECT * FROM expenses WHERE userid = {0} AND DATE(date) = DATE(NOW()) AND date ORDER BY `expenses`.`date` DESC'.format(str(session['id'])))
    cursor.execute('SELECT * FROM expenses WHERE userid = %s AND DATE(date) = DATE(NOW()) AND date ORDER BY `expenses`.`date` DESC',(id,))


    expense = cursor.fetchall()


    total=0
    t_food=0
    t_entertainment=0
    t_business=0
    t_rent=0
```

```
t_EMI=0
t_other=0


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

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

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

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

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

print(total)

print(t_food)
print(t_entertainment)
print(t_business)
print(t_rent)
```

```
    print(t_EMI)
    print(t_other)



    return render_template("today.html", texpense = texpense, expense = expense,  total = total ,
                    t_food = t_food,t_entertainment = t_entertainment,
                    t_business = t_business,  t_rent =  t_rent,
                    t_EMI =  t_EMI,  t_other =  t_other )



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


    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM expenses WHERE userid = % s AND MONTH(DATE(date))= MONTH(now()) AND date ORDER BY `expenses`.`date` DESC',(str(session['id'])))
    expense = cursor.fetchall()


    total=0
    t_food=0
    t_entertainment=0
```

```
t_business=0
t_rent=0
t_EMI=0
t_other=0


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

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

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

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

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

print(total)

print(t_food)
print(t_entertainment)
```

```
    print(t_business)
    print(t_rent)
    print(t_EMI)
    print(t_other)




    return render_template("today.html", texpense = texpense, expense = expense,  total = total ,
                    t_food = t_food,t_entertainment = t_entertainment,
                    t_business = t_business,  t_rent =  t_rent,
                    t_EMI =  t_EMI,  t_other =  t_other )


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


    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM expenses WHERE userid = % s AND YEAR(DATE(date))= YEAR(now()) AND date ORDER BY `expenses`.`date` DESC',(str(session['id'])))
    expense = cursor.fetchall()


    total=0
    t_food=0
```

```
t_entertainment=0
t_business=0
t_rent=0
t_EMI=0
t_other=0


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

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

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

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

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

print(total)

print(t_food)
```

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



    return render_template("today.html", texpense = texpense, expense = expense,  total = total ,
                    t_food = t_food,t_entertainment = t_entertainment,
                    t_business = t_business,  t_rent =  t_rent,
                    t_EMI =  t_EMI,  t_other =  t_other )

#log-out

@app.route('/logout')

def logout():
  session.pop('loggedin', None)
  session.pop('id', None)
  session.pop('username', None)
  return render_template('home.html')



if__name__== "__main__":
   app.run(debug=True)
```

CONCLUSION

Thus we conclude with our final deliverables of the project Personal expense tracker.