

IBM-10530-1662556173
PERSONAL EXPENSE TRACKER REPORT

CONTENTS

1. INTRODUCTION
 - 1.1. Project Overview
 - 1.2. Purpose
2. LITERATURE SURVEY
 - 2.1. Existing problem
 - 2.2. References
 - 2.3. Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
 - 3.1. Empathy Map Canvas
 - 3.2. Ideation & Brainstorming
 - 3.3. Proposed Solution
 - 3.4. Problem Solution fit
4. REQUIREMENT ANALYSIS
 - 4.1. Functional requirement
 - 4.2. Non-Functional requirements
5. PROJECT DESIGN
 - 5.1. Data Flow Diagrams
 - 5.2. Solution & Technical Architecture
 3. User Stories
6. PROJECT PLANNING & SCHEDULING
 - 6.1. Sprint Planning & Estimation
 - 6.2. Sprint Delivery Schedule
7. CODING & SOLUTIONING
 - 7.1. Feature 1
 - 7.2. Feature 2
 - 7.3. Database Schema (if Applicable)
8. TESTING
 - 8.1. Test Cases
 - 8.2. User Acceptance Testing
9. RESULTS
 - 9.1. Performance Metrics
10. ADVANTAGES & DISADVANTAGES
11. CONCLUSION

12. FUTURE SCOPE

13. Source Code GitHub & Project Demo Link

1. INTRODUCTION

1.1. Project Overview

We are building an application named as "Personal Expense Tracker". As the name suggests, this project is an application which is used to track the daily expenses of the user. It is like digital record keeping which keeps the records of expenses done by a user. The application keeps the track of the Income and Expenses both of user on a day-to-day basis. This application takes the income of a user and manage its daily expenses so that the user can save money. If you exceed daily expense allowed amount it will give you a warning by email, so that you don't spend much and that specific day. The application generates report of the expenses for each day, month and year. The expense made by the user can be visualised in various models like bar and pie charts. Also This segregates expenses into different categories like EMI, Rent, Food and so on.

1.2. Purpose

The personal 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 any money saved. Only at the end of the year do they find that they do not have any money saved up. This application can help visualis expense timelaine and the categories on which they were spent on.

2. LITERATURE SURVEY

2.1. Existing problem

The problem which exists as a very serious matter is Expense Management. People lose a lot of money and spend irrationally without any boundaries as they are not able to categorize what they spend on. Also, they are not able to set any expense limit per day, month or year and overindulge accidentally which leaves them at financial risk. This problem can be addressed with the help of an expense tracker which can keep track of expenses, limit the expenses and categorize them.

2.2. References

[1] Y. Anitha, R. Ranjini, S. Gomathi, "Easy App for Expanses Manager Using Android", International Journals of Computer Techniques, Volume: 3 Issue: 2, ISSN: 2394-2231

(March-April 2016).

[2] N. ZahiraJahan MCA., M. Phil, K. I. Vinodhini, "Personalized Expense Managing Assistant Using Android", International Journals of Computer Techniques (IJCT), Volume: 3 Issue: 2, ISSN: 2394-2231 (March-April 2016).

[3] S. Chandini, T. Poojitha, D. Ranjith, V. J. Mohammed Akram, M. S. Vani, V. Rajyalakshmi, "Online Income and Expense Tracker", International Research Journal of Engineering and Technology (IRJET), Volume: 06 Issue: 3, e-ISSN: 2395-0056, p-ISSN: 2395- 0072 (March 2019).

[4] P. Thanapal, Mohammed Yaseen Patel, T. P. Lokesh Raj and J. Satheesh Kumar, "Income and Expense Tracker", Indian Journal of Science and Technology, Vol 8(S2), ISSN: 0974-5645 (January 2014)

[5] Girish Bekaroo and Sameer Sunhaloo, "Intelligent Online Budget Tracker", Computer Science and IT Education Conference (2014)

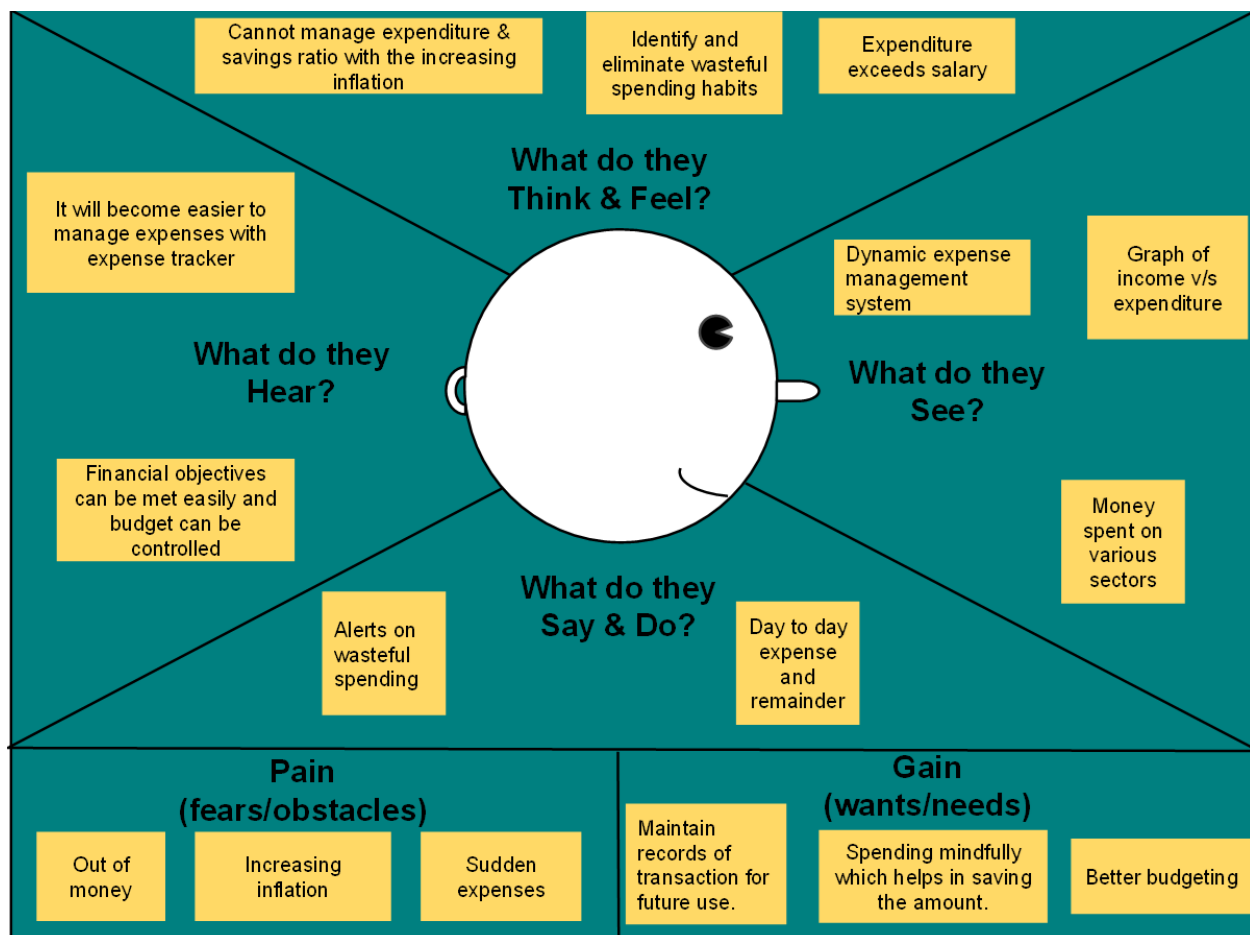
2.3. Problem Statement Definition

This Expense Tracker is a web application that facilitates the users to keep track and manage their personal as well as business expenses. This application helps the users to keep a digital diary. It will keep track of a user's income and expenses on a daily basis. The user will be able to add his/her expenditures instantly and can review them anywhere and anytime with the help of the internet. He/she can easily import transactions from his/her mobile wallets without risking his/her information and efficiently protecting his/her privacy. This expense tracker provides a complete digital solution to this problem. Excel sheets do very little to help in tracking. Furthermore, they don't have the advanced functionality of preparing graphical visuals automatically. Not only it will save the time of the people but also it will assure error free calculations. The user just has to enter the income and expenditures and everything else will be performed by the system. Keywords: Expense Tracker, budget, planning, savings,

graphical visualization of expenditure.

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas



3.2. Ideation & Brainstorming

2

Brainstorm

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

10 minutes

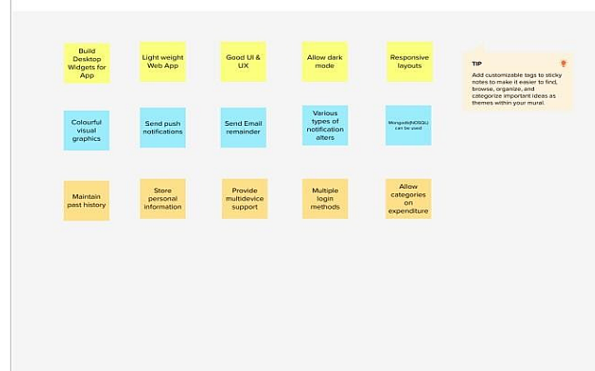


3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

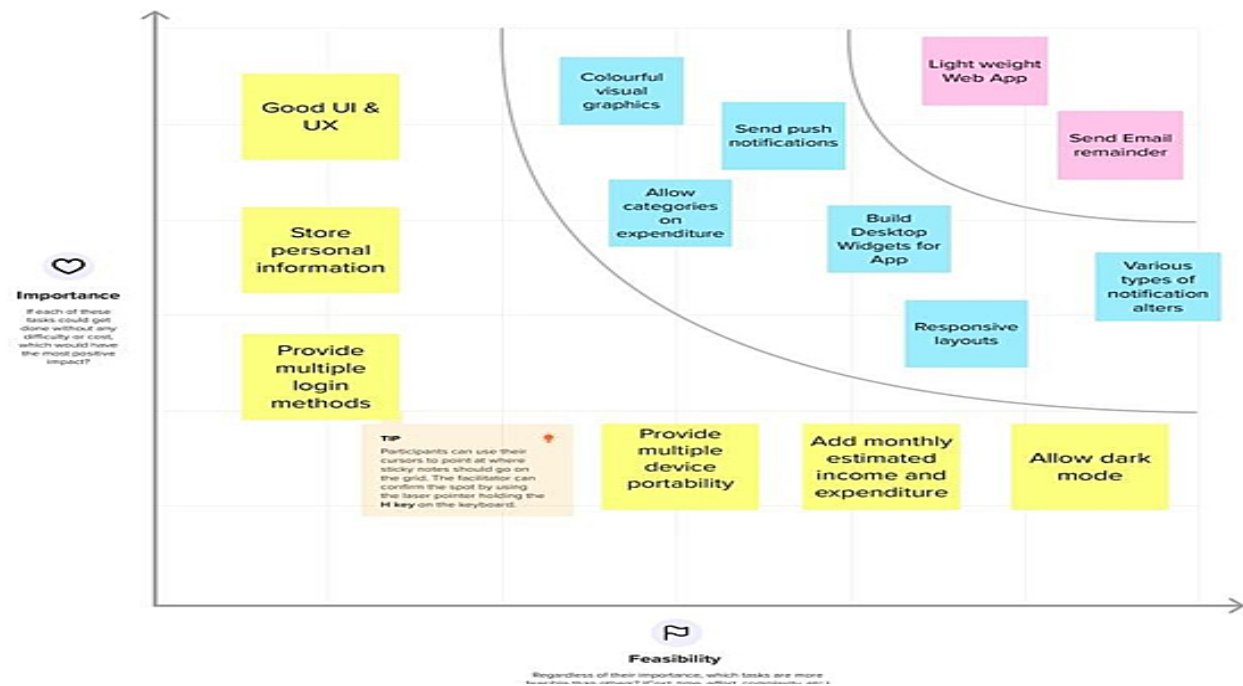


4

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



3.3. Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ol style="list-style-type: none"> 1. Lack of proper planning of our income 2. User has to keep a expense details in a diary or ina computer 3. All the calculations needs to be manually done by the user 4. Hard to keep trackof daily expenses

2.	Idea / Solution description	<ol style="list-style-type: none"> 1. To know where the money is going and spend only on priorities 2. To save money for pre-defined expenses 3. To track the performance of investments 4. To track day-to-day expense
3.	Novelty / Uniqueness	<ol style="list-style-type: none"> 1. To notify you of upcoming bill due dates 2. To generate profit and loss report 3. Categorize expenses 4. Smart budget planning
4.	Social Impact / Customer Satisfaction	<ol style="list-style-type: none"> 1. This application helps user to be financially responsible 2. Free of cost 3. Designed to be dynamic to produce prediction 4. Can reduce burden of manual calculation 5. Create awareness among people
5.	Business Model (Revenue Model)	<ol style="list-style-type: none"> 1. Application is provided free of cost, but it will have some advertisement 2. In premium version, there is no advertisement and have some additional features
6.	Scalability of the Solution	<ol style="list-style-type: none"> 1. This application can handle large number of user data 2. Provides high performance and security 3. Easily available all kinds of devices

3.4. Problem Solution fit

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS <ul style="list-style-type: none"> People above the age of 16 who earn or spend money Customers who find it difficult to keep track of their expenses. Customers who want to wisely handle their savings and money 	6. CUSTOMER CONSTRAINTS CC <ul style="list-style-type: none"> All data should be entered manually by the user. Privacy and security Network issues Not compatible in all devices Not enough balance due to lavish spending 	5. AVAILABLE SOLUTIONS AS <ul style="list-style-type: none"> Excel sheet Expense diary Expense tracker app with minimal features. It has less security features and no customer-support. 	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS J&P <ul style="list-style-type: none"> To keep track of daily expenses. Alert when a threshold limit is reached Categorizing expenses to have a good visualization of it. Difficult to track monthly expenses manually. Remembering of expenses is difficult 	9. PROBLEM ROOT CAUSE RC <ul style="list-style-type: none"> Due to many platforms which uses online payment, expenses are more and untracked Reckless spending Forgetting payments Linking of financial account to the application 	7. BEHAVIOUR BE <ul style="list-style-type: none"> Have a proper record of all the expenses. Set up monthly limit in the expenses. Would prefer a graphical representation of their daily, monthly and yearly expenses. Start saving money and reduce unwanted expenses. 	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	3. TRIGGERS <ul style="list-style-type: none"> Reduces time and manual effort Insufficient money during emergency Excessive spending Saving money effectively 	10. YOUR SOLUTION <ul style="list-style-type: none"> A personal expense tracker application to handle and keep track of the monthly income and daily/monthly/yearly expenses. Alerting the user when expenses exceed a particular limit Providing a graphical view with proper categorization of the spent amount. An application with good security and real-time tracking of expenses. 	8. CHANNELS of BEHAVIOUR 8.1 ONLINE <ul style="list-style-type: none"> The application comes with a lot of advertisements which can be irritating to the customers. Stealing of private data can be easy in online Data can be stored in cloud which can be secure Accurate graphical representation Untracked expenses if not manually updated 8.2 OFFLINE <ul style="list-style-type: none"> No real time tracking Backup cannot be guaranteed Difficult in visualization of the amount spent 	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER Before <ul style="list-style-type: none"> Confused Frustrated Stressed After <ul style="list-style-type: none"> Customers get clarity on the expenses Confident 			

4. REQUIREMENT ANALYSIS

4.1. Functional requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User Registration	Registration through Application Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User monthly expense tentativedata	Data to be registered in the app
FR-4	User monthly income data	Data to be registered in the app
FR-5	Alert/ Notification	Alert through E-mail Alert through SMS

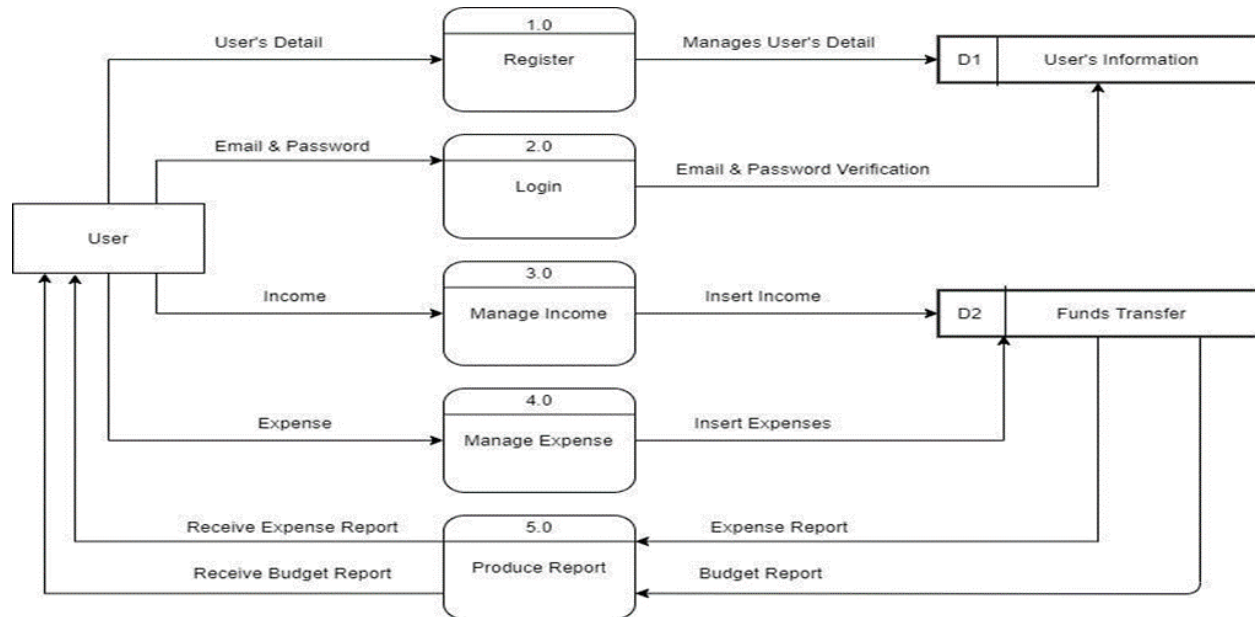
FR-6	User BudgetPlan	Planning and Tracking of user expense vs budget limit
------	-----------------	---

4.2. Non-Functional requirements:

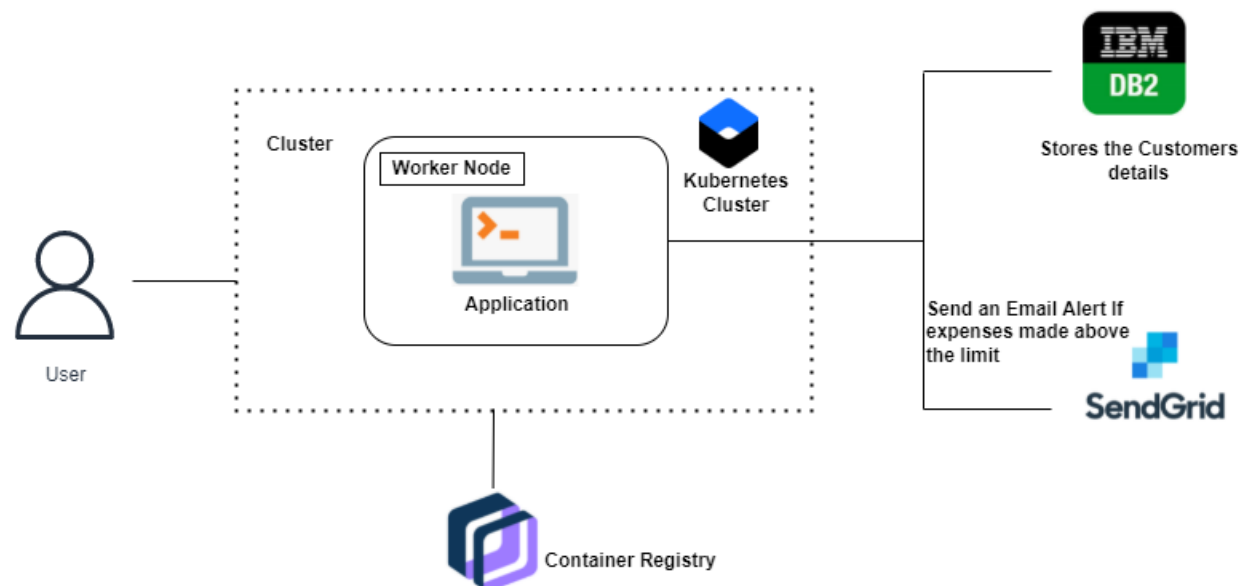
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Effectiveness, efficiency and overall satisfaction of the user while interacting with our application.
NFR-2	Security	Authentication, authorization, encryption of the application.
NFR-3	Reliability	Probability of failure-free operations in a specified environment for a specified time.
NFR-4	Performance	How the application is functioning and how responsive the application is to the end-users.
NFR-5	Availability	Without near 100% availability, application reliability and the user satisfaction will affect the solution.
NFR-6	Scalability	Capacity of the application to handle growth, especially in handling more users.

5. PROJECT DESIGN

5.1. Data Flow Diagrams



5.2. Solution & Technical Architecture



5.3. User Stories

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)	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 can track my expenses and manage budgets.	I can track my expenses and manage my budgets.	High	Sprint-2
		USN-3	As a user, I can be alerted if expenses exceed the budget.	I can mail or SMS, if expense exceeds the budget.	High	Sprint-3
		USN-4	As a user, I can see previous expenses.	I can view to the previous expenditures.	High	Sprint-2
	Login	USN-5	As a user, I can log into the application by entering email & password	I can see my daily expenses.	High	Sprint-2
	Dashboard	USN-6	As a user, I can enter the expenditure limit	I can see the alert as it exceeds to the budget.	Medium	Sprint-3
Customer (Web user)		USN-7	As a customer, I can visualize the overall expenses.	I can reduce the daily expenses.	High	Sprint-3
Customer Care Executive		USN-8	As a customer care executive, I can solve the log-in and other application issues.	I can provide services as anywhere at any time.	Low	Sprint-4
Administrator	Application	USN-9	As an administrator, I can maintain the databases and upgrade the applications.	I can degrade the errors while using this application.	Low	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-1		USN-3	As a user, I can register for the application through Facebook	2	Low	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Gokulan Kamalesh Vishnu Prabu Maharajan

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-2	Dashboard	USN-6	Logging in takes the user to their dashboard.	2	High	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-2		USN-7	As a user, I will update my balance amount or income at the start of each month.	2	Medium	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-2		USN-8	As a user , I will set a limit of the amount to spend.	2	Medium	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-3	Database	USN-9	As a user , I can view the track of my expenses	4	High	Gokulan Kamalesh Vishnu Prabu Maharajan
Sprint-4	Deployment	USN-10	Container of applications using docker kubernetes and deployment the application.	4	High	Gokulan Kamalesh Vishnu Prabu Maharajan

6.2. Sprint Delivery Schedule

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	20	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	6	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	4	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	4	19 Nov 2022

7. CODING & SOLUTIONING

7.1. Feature 1

Add Expense Feature:

This feature is the function that gets expense amounts from users and functions with the help of IBM DB2 database service offered by the IBM Cloud. This has form fields like date, expense name, amount, and pay mode which the user has to specify. The user's expenses that are added can be visualized on the display page.

```
@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']
```

```
p1 = date[0:10]
p2 = date[11:13]
p3 = date[14:]
p4 = p1 + "-" + p2 + "." + p3 + ".00"
```

```
sql = "INSERT INTO expenses (userid, date, expensename, amount, paymode,
category) VALUES (?, ?, ?, ?, ?, ?)"
```

```
stmt = ibm_db.prepare(ibm_db_conn, sql)
ibm_db.bind_param(stmt, 1, session['id'])
ibm_db.bind_param(stmt, 2, p4)
ibm_db.bind_param(stmt, 3, expensename)
ibm_db.bind_param(stmt, 4, amount)
ibm_db.bind_param(stmt, 5, paymode)
ibm_db.bind_param(stmt, 6, category)
ibm_db.execute(stmt)
```

```
print("Expenses added")
```

```
sql = "SELECT "+str(category)+" , total FROM expenses_on_category WHERE userid = "
+ str(session['id'])
```

```
res = ibm_db.exec_immediate(ibm_db_conn, sql)
dictionary = ibm_db.fetch_assoc(res)
total = 0
category_amount = 0
if dictionary != False:
    category_amount = dictionary[str(category).upper()]
    total = dictionary["TOTAL"]
```

```
total += int(amount)
category_amount += int(amount)
```

```
sql = "UPDATE expenses_on_category SET "+str(category)+" = ? , total = ? WHERE
userid = ?"
```

```
stmt = ibm_db.prepare(ibm_db_conn, sql)
ibm_db.bind_param(stmt, 1, category_amount )
ibm_db.bind_param(stmt, 2, total )
ibm_db.bind_param(stmt, 3, str(session['id']))
```

```
ibm_db.execute(stmt)
```

```
print("Expenses On Category added")
```

```
param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) + " AND  
MONTH(date) = MONTH(current timestamp) AND YEAR(date) = YEAR(current  
timestamp) ORDER BY date DESC"
```

```
res = ibm_db.exec_immediate(ibm_db_conn, param)
```

```
dictionary = ibm_db.fetch_assoc(res)
```

```
expense = []
```

```
while dictionary != False:
```

```
    temp = []
```

```
    temp.append(dictionary["ID"])
```

```
    temp.append(dictionary["USERID"])
```

```
    temp.append(dictionary["DATE"])
```

```
    temp.append(dictionary["EXPENSENAME"])
```

```
    temp.append(dictionary["AMOUNT"])
```

```
    temp.append(dictionary["PAYMODE"])
```

```
    temp.append(dictionary["CATEGORY"])
```

```
    expense.append(temp)
```

```
    dictionary = ibm_db.fetch_assoc(res)
```

```
total=0
```

```
for x in expense:
```

```
    total += x[4]
```

```
param = "SELECT id, limitss FROM limits WHERE userid = " + str(session['id']) + "  
ORDER BY id DESC LIMIT 1"
```

```
res = ibm_db.exec_immediate(ibm_db_conn, param)
```

```
dictionary = ibm_db.fetch_assoc(res)
```

```
row = []
```

```
s = 0
```

```
while dictionary != False:
```

```
    temp = []
```

```
    temp.append(dictionary["LIMITSS"])
```

```
    row.append(temp)
```

```
    dictionary = ibm_db.fetch_assoc(res)
```

```

s = temp[0]

if total > int(s):
    msg = "Hello " + session['username'] + " , " + "you have crossed the monthly limit of
Rs. " + str(s) + "/- !!!" + "\n" + "Thank you, " + "\n" + "Team Personal Expense Tracker."
    sendmail(msg,session['email'])

return redirect("/display")

```

Feature 2

Visualize Expenses Feature:

This is a feature that shows the user the Total Expense Breakdown with time and amount spent as a parameter. This also shows expense breakdown by categories such as Food, Entertainment, Rent, and Total Expense. It uses Canvas Chart features for better data visualization and offers a variety of them like bar, pie, and line charts

Flask functionality

```
@app.route("/display")
```

```
def display():
```

```

    param = "SELECT * FROM expenses WHERE userid = " + str(session['id']) + " ORDER
BY date DESC"

```

```
    res = ibm_db.exec_immediate(ibm_db_conn, param)
```

```
    dictionary = ibm_db.fetch_assoc(res)
```

```
    expense = []
```

```
    while dictionary != False:
```

```
        temp = []
```

```
        temp.append(dictionary["ID"])
```

```
        temp.append(dictionary["USERID"])
```

```
        temp.append(dictionary["DATE"])
```

```
        temp.append(dictionary["EXPENSENAME"])
```

```
        temp.append(dictionary["AMOUNT"])
```

```
        temp.append(dictionary["PAYMODE"])
```

```
        temp.append(dictionary["CATEGORY"])
```

```
        expense.append(temp)
```

```
        dictionary = ibm_db.fetch_assoc(res)
```

```

param = "SELECT * FROM expenses_on_category WHERE userid = " + str(session['id'])
res = ibm_db.exec_immediate(ibm_db_conn, param)
dictionary = ibm_db.fetch_assoc(res)
expenses_on_category = [0,0,0,0,0,0,0]
while dictionary != False:
    temp = []
    temp.append(dictionary["FOOD"])
    temp.append(dictionary["ENTERTAINMENT"])
    temp.append(dictionary["BUSINESS"])
    temp.append(dictionary["RENT"])
    temp.append(dictionary["EMI"])
    temp.append(dictionary["OTHER"])
    temp.append(dictionary["TOTAL"])
    expenses_on_category = temp
    dictionary = ibm_db.fetch_assoc(res)

return render_template('display.html', expense = expense, expenses_on_category =
expenses_on_category)

```

HTML TEMPLATE

```

{% extends 'base.html' %}
{% block body %}

<style>

#chartArea {
    display: flex;
    justify-content: space-between;
}

#chartType {
    width: 100%;
    height: 50px;
    border: 0;
    background-color: rgb(100, 220, 100);
    text-align: center;

```



```
}
```

```
option{  
  text-align: center;  
  background-color: rgb(172, 225, 172);  
}
```

```
</style>
```

```
<div class="container ">  
  <h3 class="mt-3">EXPENSES</h3>
```

```
{% if expense is defined %}  
{% for row in expense %}
```

```
<div class="row">
```

```
  <div class="col-md-12">
```

```
    <div class="card shadow-sm mb-2 bg-white rounded"></div>
```

```
    <div class="card-body">
```

```
      <div class="row">
```

```
        <div class="col-md-2">
```

```
          <span class="btn btn-outline-dark">{{row[2]}}</span> </div>
```

```
        <div class="col-md-2 mt-3"><H6>{{row[3]}}</H6></div>
```

```
        <div class="col-md-2 mt-3" > ₹<span style=" color: rgb(255, 0, 0) "> {{row[4]}}
```

```
</div>
```

```
      <div class="col-md-2 mt-3">
```

```
        <span class="badge badge-pill badge-info">{{row[5]}}</span>
```

```
      </div>
```

```
    <div class="col-md-2 mt-3">
```

```
      <span class="badge badge-primary">{{row[6]}}</span>
```

```
    </div>
```

```
<div class="col-md-1 mt-3">
  <a href="/edit/{{row[0]}}" class="btn btn-sm btn-success">Edit</a>
</div>
```

```
<div class="col-md-1 mt-3">
  <a href="/delete/{{row[0]}}" class="btn btn-sm btnDelete btn-
success">Delete</a>
</div>
```

```
</div>
</div>
```

```
</div>
</div>
<!--when no DATA-Found-->
{% else %}
<div class="card shadow-sm mb-2 bg-white rounded"></div>
<div class="card-body">
<div style="text-align: center ; font-family: monospace; color:red ; "><h5><a
href="/add"> ADD-DATA </a> to Display</h3></div>

</div>
```

```
{% endfor %}
{% endif %}
```

```
<div class="row">
  <div class="col-md-6">
    <h3 class="mt-5">Expense Breakdown</h3>

    <div class="card shadow mb-2 bg-white rounded-bottom">
      <div class="card-body ">
        <div class="row">
          <div class="col-md-6">Food</div>
```

```
        <div id="tfood" class="col-md-6"> {{ expenses_on_category[0] }} </div>
    </div>
</div>
</div>
```

```
<div class="card shadow mb-2 bg-white rounded">
    <div class="card-body">
        <div class="row">
            <div class="col-md-6">Entertainment</div>
            <div id="tentertainment" class="col-md-6"> {{ expenses_on_category[1] }}
</div>
        </div>
    </div>
</div>
```

```
<div class="card shadow mb-2 bg-white rounded">
    <div class="card-body">
        <div class="row">
            <div class="col-md-6">Business</div>
            <div id="tbusiness" class="col-md-6"> {{ expenses_on_category[2] }} </div>
        </div>
    </div>
</div>
```

```
<div class="card shadow mb-2 bg-white rounded">
    <div class="card-body">
        <div class="row">
            <div class="col-md-6">Rent</div>
            <div id="trent" class="col-md-6"> {{ expenses_on_category[3] }} </div>
        </div>
    </div>
</div>
```

```
<div class="card shadow mb-2 bg-white rounded">
```

```
<div class="card-body">
<div class="row">
  <div class="col-md-6">EMI</div>
  <div id="temi" class="col-md-6">{{ expenses_on_category[4] }} </div>
</div>
</div>
</div>
```

```
<div class="card shadow mb-2 bg-white rounded">
  <div class="card-body">
    <div class="row">
      <div class="col-md-6">Other</div>
      <div id="tother" class="col-md-6"> {{ expenses_on_category[5] }}</div>
    </div>
  </div>
</div>
</div>
```

```
<div class="card shadow mb-2 btn-outline-danger rounded-pill">
  <div class="card-body">
    <div class="row">
      <div class="col-md-6">Total</div>
      <div class="col-md-6">₹ {{ expenses_on_category[6] }} </div>
    </div>
  </div>
</div>
</div>
```

```
</div>
<div class="col-md-6">
<div id="chartArea">
  <select id="chartType" onchange="loadChart()">
    <option value="bar">Bar</option>
    <option value="doughnut">Doughnut</option>
    <option value="line">Line</option>
    <option value="pie">Pie</option>
  </select>
</div>
```

```
<canvas id="myChart" width="400" height="400"></canvas>
```

```
<script>
```

```
let food = document.getElementById('tfood').innerHTML;  
let entertainment = document.getElementById('tentertainment').innerHTML;  
let business = document.getElementById('tbusiness').innerHTML;  
let rent = document.getElementById('trent').innerHTML;  
let emi = document.getElementById('temi').innerHTML;  
let other = document.getElementById('tother').innerHTML;
```

```
function loadChart()  
{
```

```
    var chartType = document.getElementById('chartType').value;
```

```
    let chartStatus = Chart.getChart("myChart");  
    if (chartStatus !== undefined) {  
        chartStatus.destroy();  
    }
```

```
    var ctx = document.getElementById('myChart').getContext('2d');
```

```
    var myChart = new Chart(ctx, {  
        type: chartType, //'line',  
        data: {  
            labels: ['Food', 'Entertainment', 'Business', 'Rent', 'EMI', 'Other'],  
            datasets: [{  
                label: 'Expenses Chart',  
                data: [food, entertainment, business, rent, emi, other],  
                backgroundColor: [  
                    'rgb(255, 99, 132)',  
                    'rgb(0, 0, 0)',  
                    'rgb(255, 205, 86)',  
                    'rgb(201, 203, 207)',  
                    'rgb(54, 162, 235)',  
                    'rgb(215, 159, 64)'  
                ],  
            }  
        }  
    });
```

```

        }]
    },
    options: {
        responsive: true,
        plugins: {
            legend: {
                position: 'bottom',
            },
            title: {
                display: true,
                text: 'EXPENSE BREAKDOWN'
            }
        }
    }
});

}

loadChart();
</script>

</div>
</div>
</div>
{% endblock %}

```

Feature 3

Send Expense Limit Alert:

This Feature alerts the user by email when the total expense exceeds the limit set by the user.

We use SendGrid API to deliver emails to the user's email ID and alert them that they have exceeded the expense limit.

```
if total > int(s):
```

```
    msg = "Hello " + session['username'] + " , " + "you have crossed the monthly limit of
Rs. " + str(s) + "/- !!!" + "\n" + "Thank you, " + "\n" + "Team Personal Expense Tracker."
```

```

        sendmail(msg,session['email'])

    return redirect("/display")

import smtplib
import sendgrid as sg
import os
from sendgrid.helpers.mail import Mail, Email, To, Content
SUBJECT = "expense tracker"
s = smtplib.SMTP('smtp.gmail.com', 587)

def sendmail(TEXT,email):
    print("sorry we cant process your candidature")
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login("tproduct8080@gmail.com", "lxixbmpnexbkiemh")
    message = 'Subject: {}\n\n{}'.format(SUBJECT, TEXT)
    s.sendmail("il.tproduct8080@gmail.com", email, message)
    s.quit()
def sendgridmail(user,TEXT):

    from_email = Email("kamaleshelango@gmail.com")
    to_email = To(user)
    subject = "Sending with SendGrid is Fun"
    content = Content("text/plain",TEXT)
    mail = Mail(from_email, to_email, subject, content)

    # Get a JSON-ready representation of the Mail object
    mail_json = mail.get()
    # Send an HTTP POST request to /mail/send
    response = sg.client.mail.send.post(request_body=mail_json)
    print(response.status_code)
    print(response.headers)

if __name__ == "__main__":
    sendgridmail("winvishnu97@gmail.com","Expense Limit Exceeded !")

```

7.3. Database Schema (if Applicable)

The screenshot displays the IBM Db2 on Cloud web interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is active, showing a list of tables under the 'WKZ37381' schema. The 'Table definition' panel on the right shows the structure of the 'EXPENSES' table.

Name	Type	Tables
WKZ37381	User	8

Name	Schema	Properties
EXPENSES	WKZ37381	...
EXPENSES_ON...	WKZ37381	...
LIMITS	WKZ37381	...
PETA_CATEGO...	WKZ37381	...
PETA_EXPENSE	WKZ37381	...
PETA_GROUPS	WKZ37381	...
PETA_USER	WKZ37381	...
REGISTER	WKZ37381	...

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
USERID	INTEGER	Y		0
DATE	TIMESTAMP	Y	10	6
EXPENS ENAME	VARCHAR	Y	255	0
AMOUN T	INTEGER	Y		0
PAYMOD E	VARCHAR	Y	255	0
CATEGO RY	VARCHAR	Y	255	0

8. TESTING

8.1. Test Cases

Component	Functionality	Test Result
Signup Page	Register new user with username and password	Pass
Login Page	The user is able to log in with valid credentials	Pass
Add Expenses	User is able to add expenses and it is shown in the expense history	Pass
Dashboard	User is able to visualize expenses in different formats	Pass
Login Page	The user is able to log in with invalid credentials	Fail
Expense Limit Alert	The user receives an email whenever the expense limit is exceeded	Pass

8.2. User Acceptance Testing

Defect Analysis

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

they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Severity 5
By Design	1	0	0	0	1
Duplicate	1	0	0	0	1
External	3	1	0	0	4
Fixed	4	1	0	0	5
Not Reproduced	0	0	0	0	1
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	9	2	0	0	11

Test Case Analysis

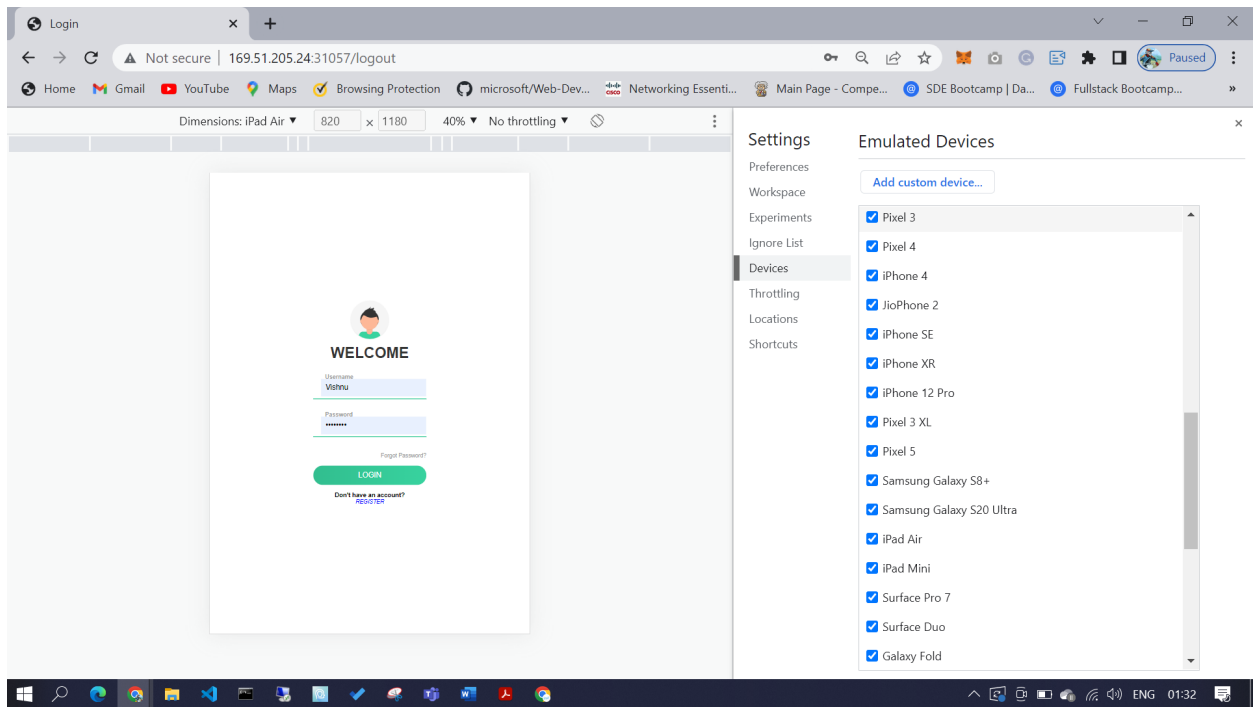
This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	0	0	0	0
Client Application	5	0	0	5
Security	0	0	0	0
Outsource Shipping	0	0	0	0
Exception Reporting	5	0	0	5
Final Report Output	0	0	0	0
Version Control	0	0	0	0

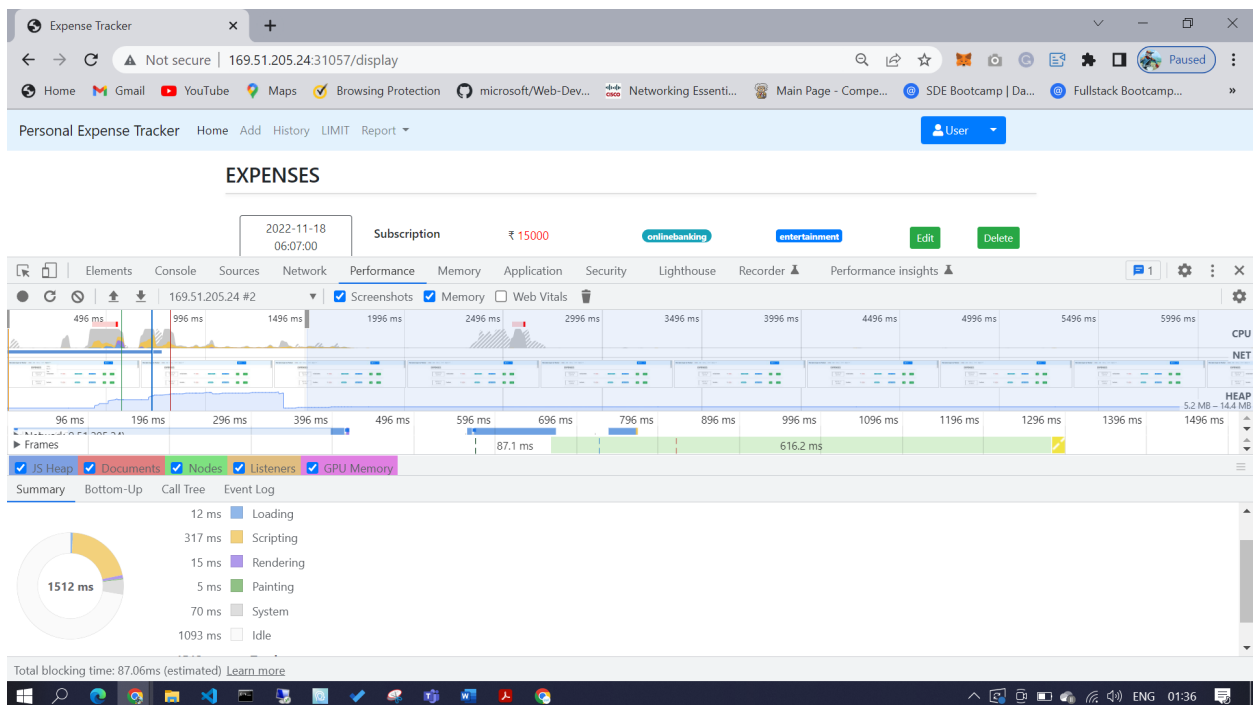
9. RESULTS

9.1. Performance Metrics

Device Ready



Performance



Memory

Expense Tracker

Not secure | 169.51.205.24:31057/home

Personal Expense Tracker Home Add History LIMIT Report

User

Memory

Summary Class filter All objects

Profiles	Constructor	Distance	Shallow Size	Retained Size
HEAP SNAPSHOTS	<ul style="list-style-type: none"> (closure) x12842 (compiled code) x30112 InternalNode x1481 Object x2520 system / Context x1531 (system) x24651 CSSStyleRule x10659 Window x26 	2	381 404 7 %	2 097 380 37 %
Snapshot 1 3/4 MB		3	1 375 260 25 %	1 834 432 33 %
		-	0 0 %	1 705 224 30 %
		2	64 108 1 %	1 315 144 23 %
		3	47 064 1 %	1 170 836 21 %
		2	662 160 12 %	1 078 116 19 %
		7	511 632 9 %	937 992 17 %
		2	3 600 0 %	915 672 16 %

Retainers

Object	Distance	Shallow Size	Retained Size

Security

Expense Tracker

Not secure | 169.51.205.24:31057/display

Personal Expense Tracker Home Add History LIMIT Report

User

EXPENSES

2022-11-18 06:07:00	Subscription	₹ 15000	onlinebanking	entertainment	Edit	Delete
2022-11-17 15:45:00	Earbuds	₹ 10000	epayment	entertainment	Edit	Delete

Security

Overview

Main origin (non-secure)

http://169.51.205.24:31057

Secure origins

- https://cdn.jsdelivr.net
- https://code.jquery.com
- https://stackpath.bootstrapcdn.com
- https://pro.fontawesome.com

Origin

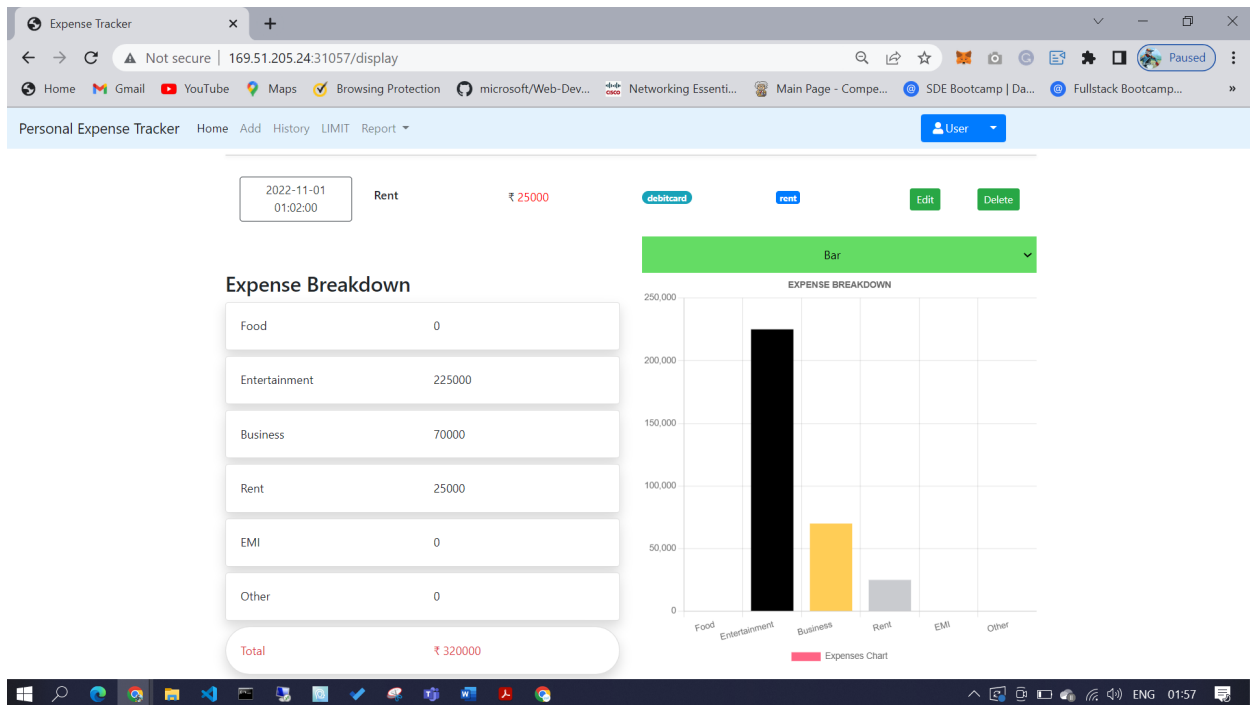
http://169.51.205.24:31057

View requests in Network Panel

Not secure

Your connection to this origin is not secure.

Visualization



10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Improved user experience
- A cloud-based solution where the load can be balanced and the application is scalable
- Alerting user via email when expense limit exceeds
- Handheld use of the application ensures ease of accessibility.
- Data is visualized graphically

DISADVANTAGES

1. Reduced Physical Audits
2. No solution to improve or eliminate bottlenecks in the service cycle

11. CONCLUSION

Taking proper care of our records is crucial in business and personal life, no matter how big or little, we must understand. We must educate ourselves about the idea of effective expense management and its applications because we will be able to spend our money wisely and save a lot of it. Modern technologies can support us in managing and keeping an eye on our expenses. We may learn,

put new ideas into practice, and assess our expense history which may improve our life.

12. FUTURE SCOPE

- A tailored mobile app that perfectly fits in your hand.
- Scale-up if needed.
- Deliver an outstanding customer experience through additional control over the app.
- Improve the security of your business and customer data.
- Increase efficiency and customer satisfaction with an app aligned to their needs.
- Chats: Equip your expense tracking app with a bot that can understand and answer all user queries and address their needs such as account balance, credit score, etc.
- Prediction: With the help of AI, your mobile app can predict your next purchase, according to your spending behaviour. Moreover, it can recommend products and provide unique insights on saving money. It brings out the factors causing fluctuations in your expenses.

13. Source Code GitHub & Project Demo Link

GitHub Link -- <https://github.com/IBM-EPBL/IBM-Project-10530-1659184998>

Demo Link -- <https://youtu.be/r1SMSqLttS8>

Project Snapshots:

Login

expense tracker - winishnu97@ x Login

Not secure | 169.51.205.24:31057/logout

WELCOME

Username

Password

[Forgot Password?](#)


LOGIN

Don't have an account? [REGISTER](#)

Register

expense tracker - winishnu97@ x Sign Up x +

← → ↻ Not secure | 169.51.205.24:31057/signup 🔍 📄 ☆ 🌙 🗑 Error



REGISTER NOW

Username

Email

Password

REGISTER

Already have an account [Sign in](#)

Windows taskbar: 03:09

Add Expenses

expense tracker - winishnu97@ x Expense Tracker x +

← → ↻ Not secure | 169.51.205.24:31057/add 🔍 📄 ☆ 🌙 🗑 Error

Personal Expense Tracker Home Add History LIMIT Report User

Add Expense

Date: 19-11-2022 03:11


Expense name: Movie Tickets

Expense Amount: 300

cash

food

Add



Windows taskbar: 03:11

Visualize Expenses

