

# A PROJECT REPORT

## CONTENTS

### **1. INTRODUCTION**

1. Project Overview
2. Purpose

### **2. LITERATURE SURVEY**

1. Existing problem
2. References
3. Problem Statement Definition

### **3. IDEATION & PROPOSED SOLUTION**

1. Empathy Map Canvas
2. Ideation & Brainstorming
3. Proposed Solution
4. Problem Solution fit

### **4. REQUIREMENT ANALYSIS**

1. Functional requirement
2. Non-Functional requirements

### **5. PROJECT DESIGN**

1. Data Flow Diagrams
2. Solution & Technical Architecture
3. User Stories

### **6. PROJECT PLANNING & SCHEDULING**

1. Sprint Planning & Estimation
2. Sprint Delivery Schedule

### **7. CODING & SOLUTIONING (Explain the features added in the project along with code)**

1. Feature 1
2. Feature 2

### **8. TESTING**

1. Test Cases
2. User Acceptance Testing

### **9. RESULTS**

1. Performance Metrics

### **10.ADVANTAGES & DISADVANTAGES**

### **11.CONCLUSION**

### **12.FUTURE SCOPE**

### **13.APPENDIX** Source Code

GitHub & Project Demo Link

# PERSONAL EXPENSE TRACKER

---

Team ID	PNT2022TMID22332
Project Name	Project – Personal Expense Tracker Application
Team Members	Muthu Subramanian M    113119UG03061 Pradesh B                      113119UG03071 Vignesh D                      113119UG03113 Sree Prakash M                113119UG03102

## 1.INTRODUCTION

### 1.1 Project overview

Mobile applications are top in user convenience and have over passed the web applications in terms of popularity and usability. There are various mobile applications that provide solutions to manage personal and group expense but not many of them provide a comprehensive view of both cases. In this paper, we develop a mobile application developed for the android platform that keeps record of user personal expenses, his/her contribution in group expenditures, top investment options, view of the current stock market, read authenticated financial news and grab the best ongoing offers in the market in popular categories. The proposed application would eliminate messy sticky notes, spreadsheets confusion and data handling inconsistency problems while offering the best overview of your expenses. With our application can manage their expenses and decide on their budget more effectively

### 1.2 Purpose

It also known as expense manager and money manager, 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.LITERATURE SURVEY

### 2.1 Existing problem

The problem of current generation population is that they can't remember where all of the money they earned have gone and ultimately have to live while sustaining the little money they have left for their essential needs. In this time there is no such perfect solution

which helps a person to track their daily expenditure easily and efficiently and notify them about the money shortage they have. For doing so they have to maintain long ledger's or computer logs to maintain such data and the calculation is done manually by the user, which may generate error leading to losses. Not having a complete tracking

## 2.2 References

- <https://nevonprojects.com/daily-expense-tracker-system/>
- <https://data-flair.training/blogs/expense-tracker-python/>
- <https://phpgurukul.com/daily-expense-tracker-using-php-and-mysql/>
- <https://ijarsct.co.in/Paper391.pdf>
- [https://kandi.openweaver.com/?landingpage=python\\_all\\_projects&utm\\_source=google&utm\\_medium=cpc&utm\\_campaign=promo\\_kandi\\_ie&utm\\_content=kandi\\_ie\\_search&utm\\_term=python\\_devs&gclid=Cj0KCQiAgribBhDkARIsAASA5bukrZgbI9UZxzpoyf0PofB1mZNxzc-okUP-3TchpYMclHTYFYiqP8aAmmwEALw\\_wcB](https://kandi.openweaver.com/?landingpage=python_all_projects&utm_source=google&utm_medium=cpc&utm_campaign=promo_kandi_ie&utm_content=kandi_ie_search&utm_term=python_devs&gclid=Cj0KCQiAgribBhDkARIsAASA5bukrZgbI9UZxzpoyf0PofB1mZNxzc-okUP-3TchpYMclHTYFYiqP8aAmmwEALw_wcB)

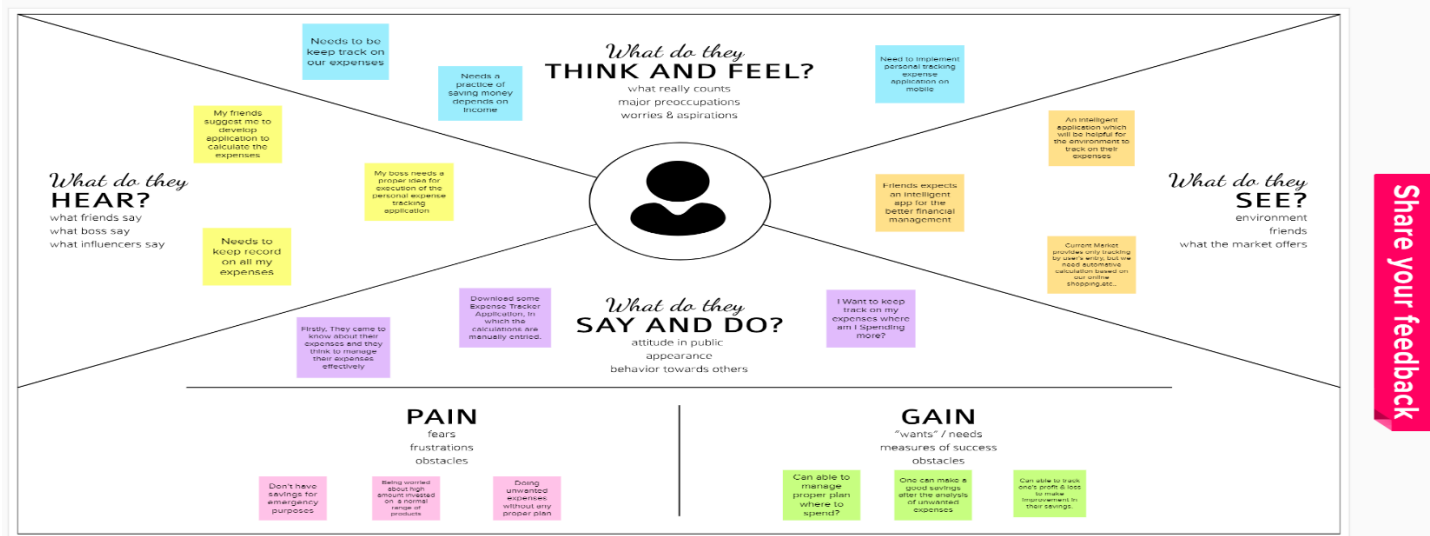
## 2.3 Problem Statement Definition

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. It is common to delete files accidentally or misplace files. 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

1 Build empathy and keep your focus on the user by putting yourself in their shoes.



### 3.2 Ideation & Brainstorming

[illegible]

### 3.3 Proposed solution

All people in the earning sector needs a way to manage their financial resources and track their expenditure, so that they can improve and monitor their spending habits. This makes them understand the importance of financial management and makes them better decisions in the future .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. The solution to this problem is, the people who gets regular payments can able to track their payments and void unwanted expenses. If the limit is exceeded the user will be notified with an email alert

### 3.4 Proposed Solution Fit

The solution to this problem is, the people who gets regular payments can able to track their payments and avoid unwanted expenses. If the limit is exceeded the user will be notified with an email alert.

- Novelty / Uniqueness Notification can be receive through email.
- Social Impact / Customer Satisfaction Using this application one can track their personal expenses and frame a monthly/annual budget. If your expense exceeded than specified limit, the application will show you an alert message. This will make an impact on Mobile Banking for Customers' Satisfaction.
- Business Model (Revenue Model) Business people can use subscription/premium feature of this application to gain revenue.
- Scalability of the Solution The scalability of the application depends on security, the working of the application even during when the network gets down etc...

## 4.REQUIREMENT ANALYSIS

### 4.1 Functional requirement

Following are the functional requirements of the proposed solution.

- FR-1 User Registration-Registration form for collecting details.
- FR-2 Login-Enter username and password.
- FR-3 Calendar-Personal Expense Tracker application must follow user to add the data to the expenses.
- FR-4 Expense Tracker-This application should graphically represent the expense in the form of report.
- FR-5 Report Generation-Graphical representation of report must be generated.
- FR-6 Category-This application shall allow user to add different categories of their expenses.

## 4.2 Non Functional Requirement

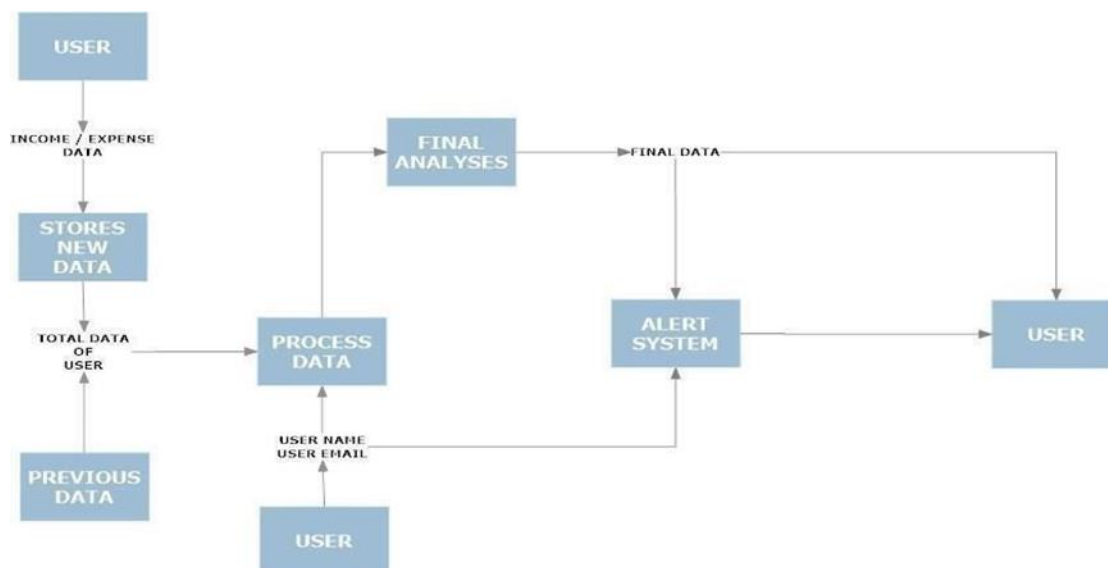
Following are the non-functional requirements of the proposed solution.

- NFR-1 Usability You will able to allocate money to different priorities and also help you to cut down on unnecessary spending
- NFR-2 Security More security of the customer data and bank account details.
- NFR-3 Reliability Used to manage his/her expense so that the user is the path of financial stability. It is categorized by week, month, and year and also helps to see more expenses made. Helps to define their own categories.
- NFR-4 Performance The types of expense are categories along with an option throughput of the system is increased due to light weight database support.
- NFR-5 Availability Able to track business expense and monitor important for maintaining healthy cash flow.
- NFR-6 Scalability The ability to appropriately handle increasing demands.

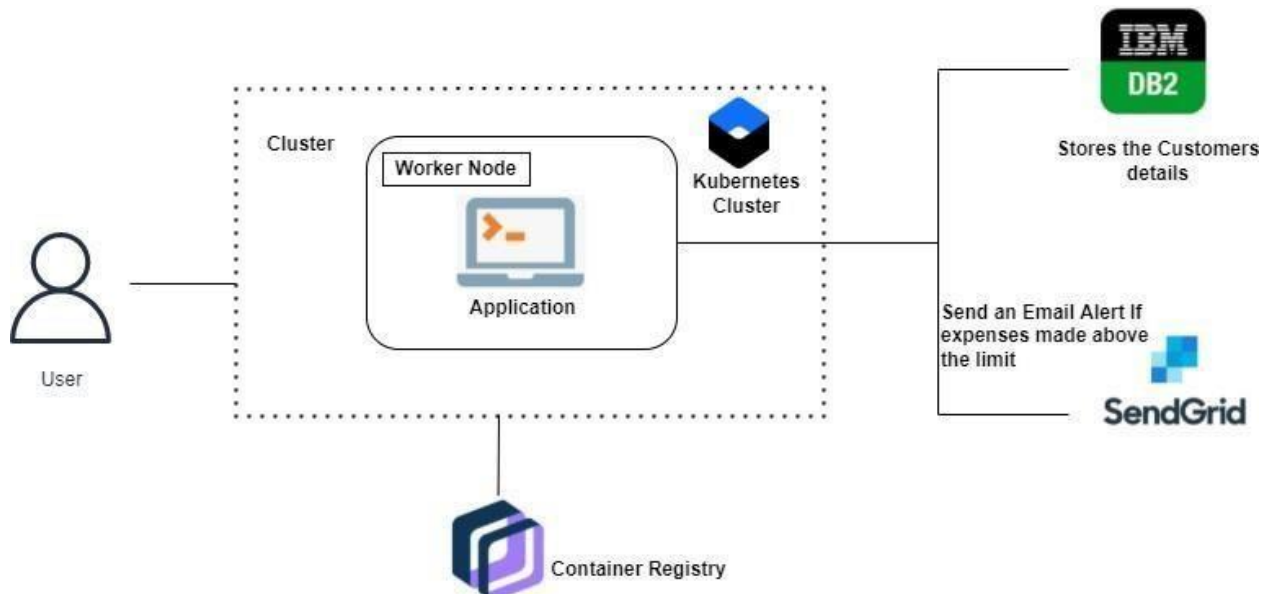
## 5. PROJECT DESIGN

### 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



## 5.2 Solution & Technical 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)	Registration	USN-1	As a user, I can register for the application by entering my email, and confirming my password.	I can access my account / dashboard	High	Sprint 1
		USN-2	As a user, I will receive confirmation that I need to signup before login	I can register my account if I not signed up before	High	Sprint-1
		USN-3	As a user, I can register for the application through my Gmail	I can register & access the dashboard with Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register by entering the details	Medium	Sprint 1

	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my dashboard	High	Sprint 1
	Dashboard	USN-6	As a user ,I can log into the dashboard and manage income	I can access my account / dashboard	High	Sprint 1
Customer (Web user)		USN-7	As a user, I can register for the application by Bank account.	I can access my account / dashboard	High	Sprint 1
Customer Care Executive		USN-8	As a user, I can get a report is	I can manage my money by	Medium	Sprint 1

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

	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, new password and confirming the same password.	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
		USN -2	As a user, I will receive confirmation email once I have registered for the application.	1	Low	
	Login	USN -3	As a user, I can log into the application by entering email and password / Google OAuth.	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	Dashboard	USN -4	Logging in takes the user to their dashboard.	1	Low	



Sprint - 2		USN -5	As a user, I will update my salary at the start of each month.	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
		USN -6	As a user, I will set a target/limit to keep track of my expenditure.	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	Workspace	USN -7	Workplace for personal expense tracking	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	Charts	USN -8	Graphs to show weekly and everyday expenditure	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
		USN -9	As a user, I can export raw data as csv file.	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 3	IBM DB2	USN -10	Linking database with dashboard	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
		USN -11	Making dashboard interactive with JS	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash

	Watson Assistant	USN -12	Embedding Chatbot to clarify user's queries.	1	Low	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	SendGrid	USN -13	Using SendGrid to send mail to the user. (To alert or remind)	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash

Sprint - 4	Integration	USN -14	Integrating frontend and backend.	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	Docker	USN -15	Creating Docker image of web app.	2	High	Muthu Subramanian, Pradesh,
	Kubernetes	USN -17	Creating container using docker and hosting the webapp.	2	High	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash
	Exposing Deployment	USN -18	Exposing IP/Ports for the site.	1	Medium	Muthu Subramanian, Pradesh, Vignesh, Sreeprakash

## 6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint1	20	6 Days	24-10-2022	29-10-2022	20	30-10-2022
Sprint2	20	6 Days	31-10-2022	05-11-2022	20	06-11-2022
Sprint3	20	6 Days	07-11-2022	12-11-2022	20	13-11-2022
Sprint4	20	6 Days	14-11-2022	19-11-2022	20	19-11-2022

## 7. COADING AND SOLUTIONING

### 7.1 Features

Feature 1: Add Expense

Feature 2: Update Expense

Feature 3: View Expenses

Feature 4: Set Limit

Feature 5: Send Alert Emails To Users

## 7.2 Others Features

Track your expenses anywhere, anytime. Seamlessly manage your money and budget without any financial paperwork. Just click and submit your invoices and expenditures. Access, submit, and approve invoices irrespective of time and location. Avoid data loss by scanning your tickets and bills and saving in the app. Approval of bills and expenditures in real-time and get notified instantly. Quick settlement of claims and reduced human errors with an automated and streamlined billing process

### Codes:

#### BASE.HTML

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Personal Expense Tracker</title>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"
  integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOMLAsJc"
  crossorigin="anonymous">
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v6.1.1/css/all.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.7.2/css/bulma.min.css" />
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
  <link rel="shortcut icon" type="image/jpg" href="/static/images/favicon.ico"/>
</body>

<div class="hero-head">
  <nav class="navbar bg-dark">
    <div class="container">

      <div id="navbarMenuHeroA" class="navbar-menu">

        <h2 class="text-white font-weight-bold">Personal Expense Tracker</h2>

      </div>
    </div>
  </nav>
</div>

<div class="navbar-end">
  <a href="{{ url_for('main.index') }}" class="navbar-item">
    <p class="text-white">Home</p>
  </a>
  {% if current_user.is_authenticated %}
  <a href="{{ url_for('main.dashboard') }}" class="navbar-item">
    <p class="text-white">Dashboard</p>
  </a>
  {% endif %}
</div>
```

```

        {% if not current_user.is_authenticated %}
        <a href="{{ url_for('auth.login') }}" class="navbar-item">
            <p class="text-white">Login</p>
        </a>
        <a href="{{ url_for('auth.signup') }}" class="navbar-item">
            <p class="text-white">Sign Up</p>
        </a>
        {% endif %}
        {% if current_user.is_authenticated %}
        <a href="{{ url_for('auth.logout') }}" class="navbar-item">
            <p class="text-white">Logout</p>
        </a>
        {% endif %}
    </div>
</div>
</div>
</nav>
</div>

<section class="hero is-fullheight" style="background-color:#FFFFFF;">

    <div class="hero-body">
        <div class="container has-text-centered">
            {% block content %}{% endblock %}
        </div>
    </div>
</section>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</body>
<footer class="bg-dark text-center text-white">
    <!-- Grid container -->
    <div class="container p-4 pb-0">
        <!-- Section: Social media -->
        <section class="mb-4">
            <!-- LinkedIn -->
            <a class="btn btn-outline-light btn-floating m-1" href="https://linkedin.com/in/muthu-subramanian-m-0a14a01ba" role="button">
                <i class="fab fa-linkedin-in"></i>
            </a>
            <!-- Github -->
            <a class="btn btn-outline-light btn-floating m-1" href="https://github.com/Muthusubramanian1008"
role="button">
                <i class="fab fa-github"></i>
            </a>
        </section>
        <!-- Section: Social media -->
    </div>
    <!-- Grid container -->

    <!-- Copyright -->
    <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.2);">
        © 2022 Copyright: About me-
        <a class="text-white" href="https://muthusubramanian1008.github.io/Muthu-Personal-site/?fbclid=PAAabpO4LgzoaOCFp52AXrGco_CGEJ8MrbQ1BMr5mvv0-GYe7yjUIvObASTQ4 ">Muthu
Subramanian</a>

```

```
</div>
<!-- Copyright -->
</footer>
</html>
```

## DASHBOARD.HTML

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Personal Expense Tracker</title>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link rel="shortcut icon" type="image/jpg" href="/static/images/favicon.ico"/>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOMLAsjC"
crossorigin="anonymous">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.7.2/css/bulma.min.css" />
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>

<body>

<div class="hero-head">
  <nav class="navbar bg-dark">
    <div class="container">

      <div id="navbarMenuHeroA" class="navbar-menu">

        <p class="text-white font-weight-bold">Personal Expense Tracker</p>

        <div class="navbar-end center">
          <div class="alert alert-success" role="alert">
            <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
            <strong>Success!</strong> You have been signed in successfully!
          </div>

          <a href="{ { url_for('main.index') } }" class="navbar-item">
            <p class="text-white">Home</p>
          </a>
          { % if current_user.is_authenticated % }
          <a href="{ { url_for('main.addexpenses') } }" class="navbar-item">
            <p class="text-white">Add Expenses</p>
          </a>
          <script>
```

```

window.setTimeout(function() {
    $(".alert-success").fadeOut(500, 0).slideUp(500, function(){
        $(this).remove();
    });
}, 2000);

```

```

</script>
<a href="{ { url_for('main.dashboard') } }" class="navbar-item">
    <p class="text-white">Dashboard</p>
</a>
<a href="{ { url_for('main.history') } }" class="navbar-item">
    <p class="text-white">History</p>
</a>
<a href="{ { url_for('main.limit') } }" class="navbar-item">
    <p class="text-white">Limit</p>
</a>
<a href="{ { url_for('main.report') } }" class="navbar-item">
    <p class="text-white">Report</p>
</a>
{% endif %}
{% if not current_user.is_authenticated %}

```

```

<a href="{ { url_for('auth.login') } }" class="navbar-item">
    <p class="text-white">Login</p>
</a>
<a href="{ { url_for('auth.signup') } }" class="navbar-item">
    <p class="text-white">Sign Up</p>
</a>
{% endif %}
{% if current_user.is_authenticated %}

```

```

<a href="{ { url_for('auth.logout') } }" class="navbar-item">
    <p class="text-white"> Logout</p>
</a>
{% endif %}

```

```

</div>

```

```

</div>

```

```

</div>

```

```

</nav>

```

```

</div>

```

```

<h1 class="title"><center>Monthly Expense Tracker, Spending Planner - Budgeting</center></h1>
<br>
<h2 class="subtitle text-info">
    <center><b>Welcome, { { name } }!</b></center>
</h2>

```

```

<p class="text-justify">

```

The idea of tracking your expenses can feel overwhelming, especially if you've been avoiding it for a while or have never done it before. But once you get started really looking at your budget and finances,

you'll feel a sense of relief and control. Finally getting on top of your money and debts comes with huge pay offs: peace of mind and no more debts!

When it comes to tracking your spending, there can be different reasons for doing it. Maybe you're curious about where your money is going, are working towards a specific goal, or want to deal with your debt once and for all. Whatever your reasons, we've got the tools and resources to help you get started.

</p>  
<div>  
<br>  
<p class="title">How to Track Your Spending and Expenses</p>  
<p class="text-justify">

Tracking where your money is going is the process of writing down what you spend. You may use a little notebook to do this, a spreadsheet, an app on your phone or tablet, or other software programs. You can track as you spend money or you collect receipts and track at the end of the day or week. However you do it, your goal is to see what you're spending money on so that you can figure out how to spend it more wisely.

</p>  
<script> //Watson Assistant  
window.watsonAssistantChatOptions = {  
 integrationID: "6b6760dd-00e6-4c3b-98e5-6f4431370cf4", // The ID of this integration.  
 region: "jp-tok", // The region your integration is hosted in.  
 serviceInstanceID: "bfdc1ed1-04d9-46dd-b90b-e6bdd9bad74a", // The ID of your service instance.  
 onLoad: function(instance) { instance.render(); }  
};  
setTimeout(function(){  
 const t=document.createElement('script');  
 t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +  
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";  
 document.head.appendChild(t);  
});  
</script>

</body>  
</html>

## INDEX.HTML

{% extends "base.html" %}

{% block content %}

<h1 class="title ">  
 Track All Your Expenses at One Go!  
</h1>

<h2 class="subtitle">  
 We Value Your Expenses!!  
</h2>

<h2 class="teal-text">Click Here to <a href="{ {url\_for('auth.signup')}} ">Signup</a></h2>

```
{% endblock % }
```

## LOGIN.HTML

```
{% extends "base.html" % }
```

```
{% block content % }
```

```
<div class="column is-4 is-offset-4">
```

```
  <h3 class="title">Login</h3>
```

```
  <div class="box">
```

```
    {% with messages = get_flashed_messages() % }
```

```
    {% if messages % }
```

```
      <div class="notification is-danger">
```

```
        { { messages[0] } }
```

```
      </div>
```

```
    {% endif % }
```

```
  {% endwith % }
```

```
  <form method="POST" action="/login">
```

```
    <div class="field">
```

```
      <div class="control">
```

```
        <input class="input is-large" type="email" name="email" placeholder="Your Email"
autofocus="">
```

```
      </div>
```

```
    </div>
```

```
    <div class="field">
```

```
      <div class="control">
```

```
        <input class="input is-large" type="password" name="password" placeholder="Your
Password">
```

```
      </div>
```

```
    </div>
```

```
    <div class="field">
```

```
      <label class="checkbox">
```

```
        <input type="checkbox">
```

```
        Remember me
```

```
      </label>
```

```
    </div>
```

```
    <button class="button is-block is-info is-large is-fullwidth">Login</button>
```

```
  </form>
```

```
</div>
```

```
</div>
```

```
{% endblock % }
```

## SIGNUP.HTML

```
{% extends "base.html" % }
```

```
{% block content % }
```

```
<div class="column is-4 is-offset-4">
```

```
  <h3 class="title">Sign Up</h3>
```

```
  <div class="box">
```



```

{% with messages = get_flashed_messages() %}
{% if messages %}
    <div class="notification is-danger">
        {{ messages[0] }}<br> Go to <a href="{{ url_for('auth.login') }}">login page</a>.
    </div>
{% endif %}
{% endwith %}

<form method="POST" action="/signup">
    <div class="field">
        <div class="control">
            <input class="input is-large" type="email" name="email" placeholder="Email"
autofocus="">
        </div>
    </div>

    <div class="field">
        <div class="control">
            <input class="input is-large" type="text" name="name" placeholder="Name" autofocus="">
        </div>
    </div>

    <div class="field">
        <div class="control">
            <input class="input is-large" type="password" name="password" placeholder="Password">
        </div>
    </div>

    <button class="button is-block is-info is-large is-fullwidth">Sign Up</button>
</form>
</div>
{% endblock %}

```

### **\_\_init\_\_.py:**

```

from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager

# init SQLAlchemy so we can use it later in our models
db = SQLAlchemy()
app = Flask(__name__) # creates the Flask instance, __name__ is the name of the current Python module
# app.config['SECRET_KEY'] = 'secret-key-goes-here'
# db.init_app(app)
def create_app():
    app = Flask(__name__) # creates the Flask instance, __name__ is the name of the current Python module
    app.config['SECRET_KEY'] = 'secret-key-goes-here' # it is used by Flask and extensions to keep data safe

```

```

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///db.sqlite' #it is the path where the SQLite
database file will be saved
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False # deactivate Flask-SQLAlchemy
track modifications
db.init_app(app) # Initialiaze sqlite database
# The login manager contains the code that lets your application and Flask-Login work together
login_manager = LoginManager() # Create a Login Manager instance
login_manager.login_view = 'auth.login' # define the redirection path when login required and we
attempt to access without being logged in
login_manager.init_app(app) # configure it for login
from models import User
@login_manager.user_loader
def load_user(user_id): #reload user object from the user ID stored in the session
    # since the user_id is just the primary key of our user table, use it in the query for the user
    return User.query.get(int(user_id))

# blueprint for auth routes in our app
# blueprint allow you to orgnize your flask app
from auth import auth as auth_blueprint
app.register_blueprint(auth_blueprint)

# blueprint for non-auth parts of app
from main import main as main_blueprint
app.register_blueprint(main_blueprint)
return app

# with app.app_context():
#     db.create_all()

```

### **auth.py:**

```

from flask import Blueprint, render_template, redirect, url_for, request, flash
from werkzeug.security import generate_password_hash, check_password_hash
from models import User
from flask_login import login_user, logout_user, login_required, current_user
from __init__ import db

auth = Blueprint('auth', __name__) # create a Blueprint object that we name 'auth'

@auth.route('/login', methods=['GET', 'POST']) # define login page path
def login(): # define login page fuction
    global userid
    msg = ""
    if request.method=='GET': # if the request is a GET we return the login page
        return render_template('login.html',msg=msg)
    else: # if the request is POST the we check if the user exist and with te right password
        email = request.form.get('email')
        password = request.form.get('password')
        remember = True if request.form.get('remember') else False
        user = User.query.filter_by(email=email).first()

```

```

# check if the user actually exists
# take the user-supplied password, hash it, and compare it to the hashed password in the database
if not user:
    flash('Please sign up before!')
    return redirect(url_for('auth.signup'))
elif not check_password_hash(user.password, password):
    flash('Please check your login details and try again.')
    return redirect(url_for('auth.login')) # if the user doesn't exist or password is wrong, reload the
page
# if the above check passes, then we know the user has the right credentials
login_user(user, remember=remember)
return redirect(url_for('main.dashboard'))

# if request.method == 'POST' :

# username = request.form['username']
# password = request.form['password']
# sql = "SELECT * FROM REGISTER WHERE USERNAME=? AND PASSWORD=?"
# stmt = ibm_db.prepare(conn, sql)
# ibm_db.bind_param(stmt,1,username)
# ibm_db.bind_param(stmt,2,password)
# ibm_db.execute(stmt)
# account = ibm_db.fetch_assoc(stmt)
# print(account)

# if account:
#     session['loggedin'] = True
#     session['id'] = account["ID"]
#     userid= account["ID"]
#     session['username'] = account["USERNAME"]
#     session['email']=account["EMAIL"]

#     return redirect('/')
# else:
#     msg = 'Incorrect username / password !'

@auth.route('/signup', methods=['GET', 'POST'])# we define the sign up path
def signup(): # define the sign up function
    if request.method=='GET': # If the request is GET we return the sign up page and forms
        return render_template('signup.html')
    else: # if the request is POST, then we check if the email doesn't already exist and then we save data
        email = request.form.get('email')
        name = request.form.get('name')
        password = request.form.get('password')
        user = User.query.filter_by(email=email).first() # if this returns a user, then the email already exists
in database
        if user: # if a user is found, we want to redirect back to signup page so user can try again
            flash('Email address already exists')
            return redirect(url_for('auth.signup'))
        # create a new user with the form data. Hash the password so the plaintext version isn't saved.

```

```

        new_user = User(email=email, name=name, password=generate_password_hash(password,
method='sha256')) #
        # add the new user to the database
        db.session.add(new_user)
        db.session.commit()
        return redirect(url_for('auth.login'))

@auth.route('/logout') # define logout path
@login_required
def logout(): #define the logout function
    logout_user()
    return redirect(url_for('main.index'))

```

### **main.py:**

```

from flask import Blueprint, render_template, flash, redirect, request, session
from flask_login import login_required, current_user
from __init__ import create_app, db
import sqlite3
import ibm_db
import re
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from models import User , Expenses

main = Blueprint('main', __name__)

main.secret_key="secured"

conn = sqlite3.connect('db.sqlite')

@main.route('/') # home page that return 'index'
def index():
    return render_template('index.html')

@main.route('/dashboard') # profile page that return 'profile'
@login_required
def dashboard():
    return render_template('dashboard.html', name=current_user.name)

@main.route("/limit")
@login_required
def limit():
    return redirect('/limitn')

@main.route("/limitnum" , methods = ['POST' ])
def limitnum():
    if request.method == "POST":

```

```

number= request.form['number']

return redirect('/limitn')

@main.route("/limitn")
def limitn():
    row = 100000

    return render_template("limit.html" , y= row)

@main.route('/addexpenses',methods=['GET', 'POST'])
@login_required
def addexpenses():
    return render_template('addexpenses.html')

@main.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']
    time=request.form['time']
    object =
Expenses(user_id=current_user.id,date=date,Expensename=expensename,amount=amount,paymentmode
=paymode,category=category,time=time)
    db.session.add(object)
    db.session.commit()
    limit=5
    if(limit>10):
        mail_from = 'muthusubramanian1008@gmail.com'
        mail_to = session['email']

        msg = MIMEMultipart()
        msg['From'] = mail_from
        msg['To'] = mail_to
        msg['Subject'] = 'Expense Alert Limit'
        mail_body = """
Dear User, You have exceeded the specified monthly expense Limit!!!!

"""
        msg.attach(MIMEText(mail_body))

    try:
        server = smtplib.SMTP_SSL('smtp.sendgrid.net', 465)
        server.ehlo()
        server.login('apikey',
'SG.abtZTW0XTv6MWJXdVW2sg.r_1bDQUJUwsDAtcxaVKQClBW9akQCV0cOy02XtN1Uwo')
        server.sendmail(mail_from, mail_to, msg.as_string())
        server.close()

```

```
        print("mail sent")
    except:
        print("issue")
```

```
return redirect("/history")
```

```
@main.route('/history')
```

```
@login_required
```

```
def history():
```

```
    object = Expenses.query.filter_by(user_id=current_user.id).all()
```

```
    food = 0
```

```
    Entertainment = 0
```

```
    Business = 0
```

```
    Rent = 0
```

```
    EMI= 0
```

```
    other = 0
```

```
    total = 0
```

```
    for i in object:
```

```
        if i.category == 'food':
```

```
            food = food + i.amount
```

```
        if i.category == 'entertainment':
```

```
            Entertainment = Entertainment + i.amount
```

```
        if i.category == 'EMI':
```

```
            EMI = EMI + i.amount
```

```
        if i.category == 'rent':
```

```
            Rent = Rent + i.amount
```

```
        if i.category == 'other':
```

```
            other = other + i.amount
```

```
        if i.category == 'business':
```

```
            Business = Business + i.amount
```

```
    total = food+Entertainment+Business+Rent+EMI+other
```

```
    return
```

```
render_template('history.html',total=total,food=food,Entertainment=Entertainment,Business=Business,Rent=Rent,EMI=EMI,other=other)
```

```
#delete---the--data
```

```
@main.route('/delete/<string:id>', methods = ['POST', 'GET' ])
```

```
def delete(id):
```

```
    print(id)
```

```
    return redirect("/display")
```

```
@main.route('/report')
```

```
@login_required
```

```
def report():
```

```

object = Expenses.query.filter_by(user_id=current_user.id).all()
food = 0
Entertainment = 0
Business = 0
Rent = 0
EMI= 0
other = 0
total = 0

for i in object:
    if i.category == 'food':
        food = food + i.amount
    if i.category == 'entertainment':
        Entertainment = Entertainment + i.amount
    if i.category == 'EMI':
        EMI = EMI + i.amount
    if i.category == 'rent':
        Rent = Rent + i.amount
    if i.category == 'other':
        other = other + i.amount
    if i.category == 'business':
        Business = Business + i.amount

total = food+Entertainment+Business+Rent+EMI+other

return
render_template('report.html',total=total,food=food,Entertainment=Entertainment,Business=Business,Rent=Rent,EMI=EMI,other=other)

app = create_app() # we initialize our flask app using the __init__.py function

if __name__ == '__main__':
    db.create_all(app=create_app())
    app.run(debug=True)

from flask_login import UserMixin
from __init__ import db

class User(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key=True) # primary keys are required by SQLAlchemy
    email = db.Column(db.String(100), unique=True)
    password = db.Column(db.String(100))
    name = db.Column(db.String(1000))

class Expenses(db.Model):
    #USERID,DATE,EXPENSENAME,AMOUNT,PAYMENTMODE,CATEGORY,TIME
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer,db.ForeignKey('user.id'))
    date = db.Column(db.String(100))
    Expensename = db.Column(db.String(100))

```

```
amount=db.Column(db.Integer())
paymentmode=db.Column(db.String(100))
category =db.Column(db.String(100))
time=db.Column(db.String(100))
```

## 8. TESTING:

### 8.1 Testing:

- Login Page (Functional)
- Login Page (UI)
- Add Expense Page (Functional)

### 8.2 User Acceptance Testing:

#### 1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

#### 2. Defect Analysis

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	10	4	2	8	15
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	9	2	4	11	20
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	0	1	8
Totals	22	14	11	22	51

#### 3.Test Case Analysis

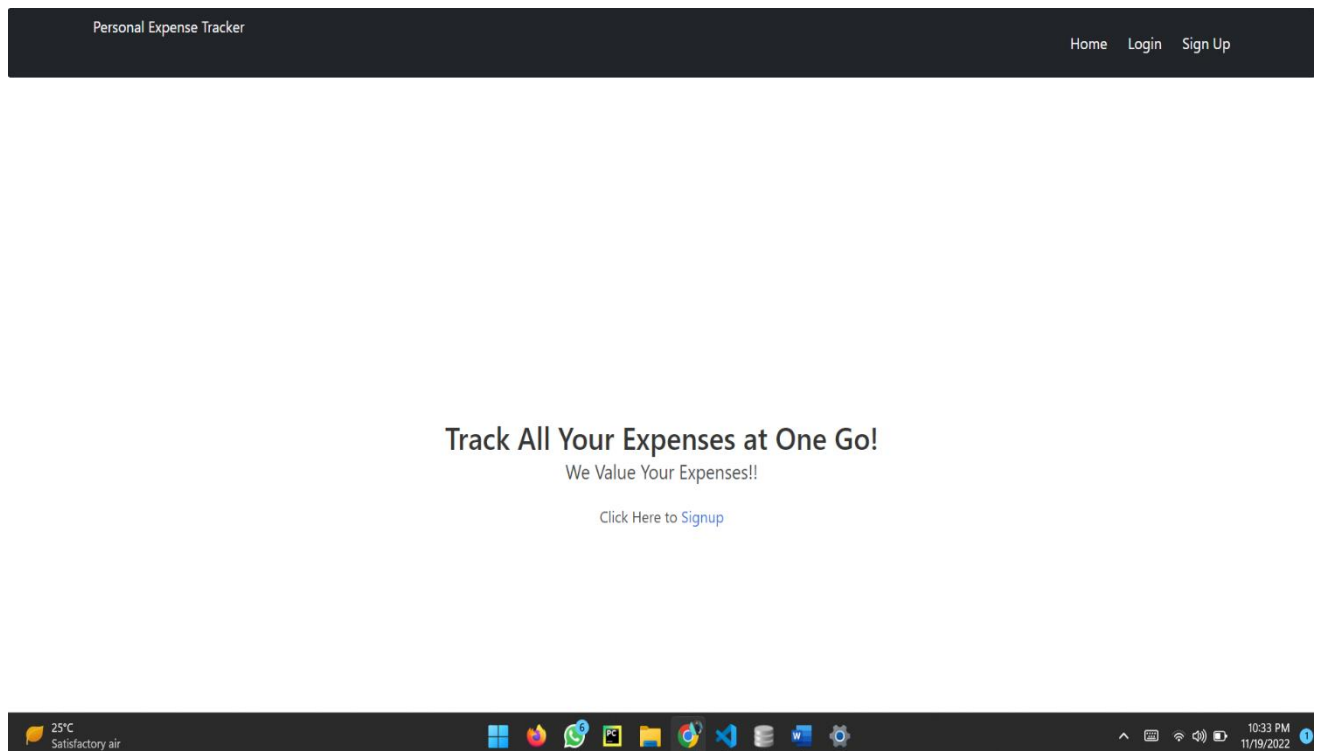


This report shows the number of test cases that have passed, failed, and untested

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

## 9. Results

### 9.1 Home page



## 9.2 Sign Up Page

Personal Expense Tracker

HomeLoginSign Up

Sign Up

Email

Name

Password

Sign Up

25°C Cloudy

10:34 PM

11/19/2022

## 9.3 Login page

Personal Expense Tracker

HomeLoginSign Up

Login

Your Email

Your Password

☐ Remember me

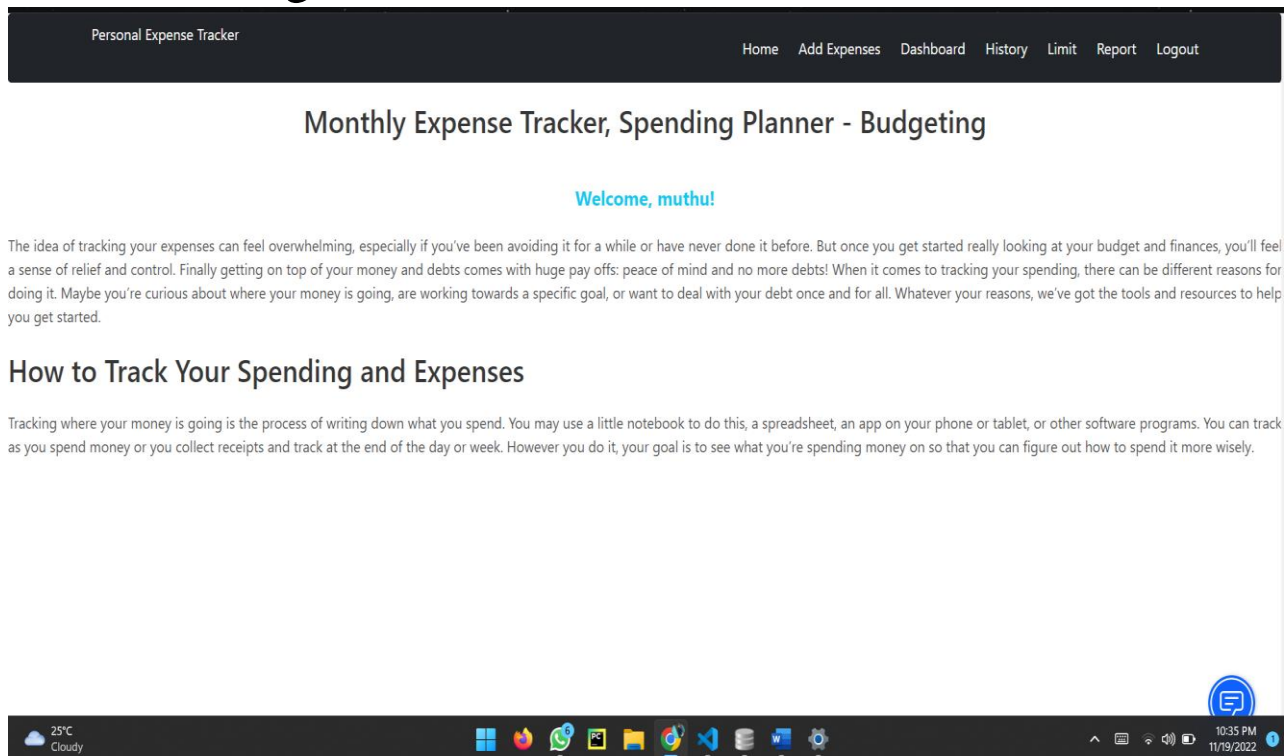
Login

25°C AQI 59

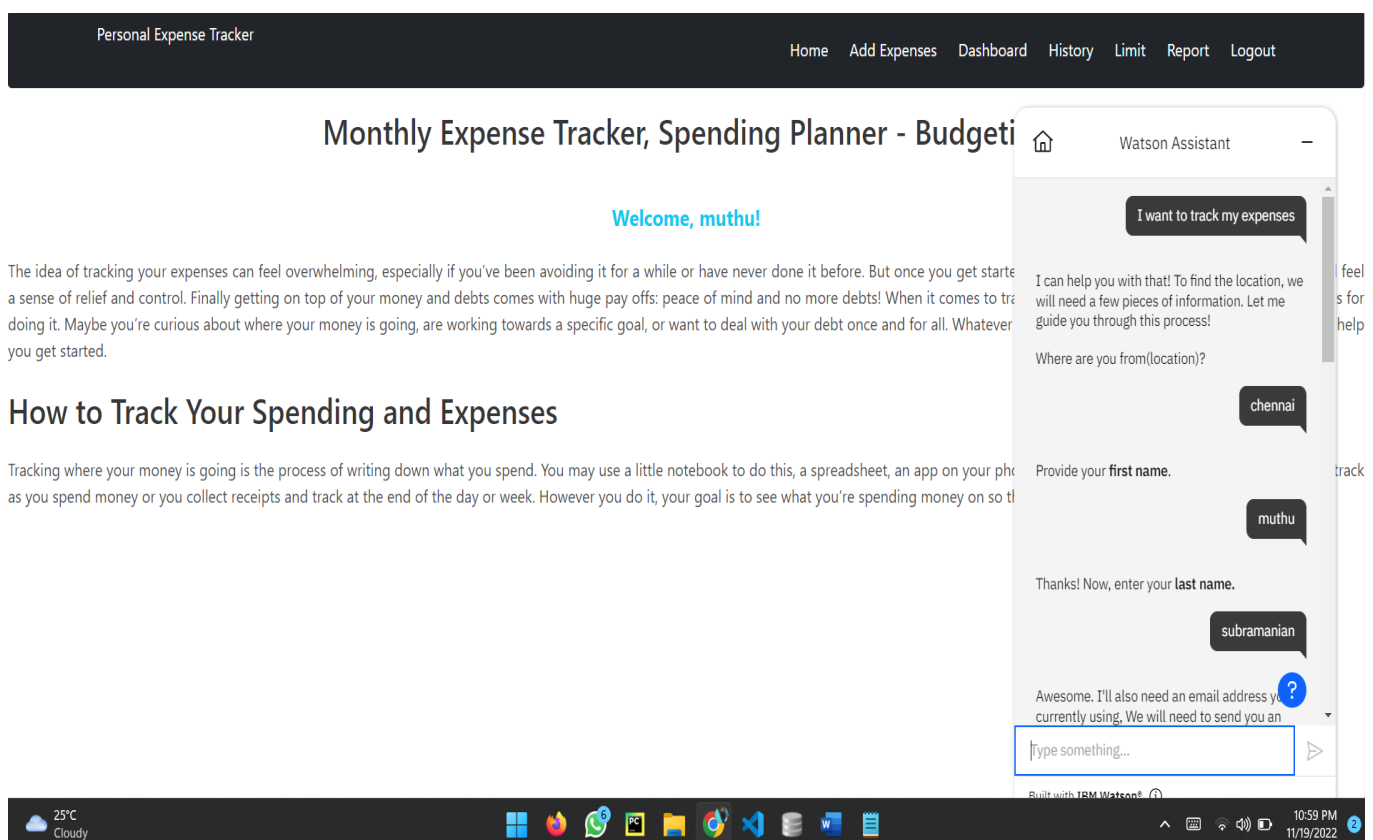
10:34 PM

11/19/2022

## 9.4 Home Page



## 9.5 IBM Watson Assistant



## 9.6 Add Expense Page

Personal Expense Tracker

HomeAdd ExpensesDashboardHistoryLimitReportLogout

Add Expense

Date

11/19/2022

Time

10:36 AM

Expense name

Bike


Expense Amount

5000


debitcard

EMI

Add



25°C Cloudy



10:37 PM11/19/2022

## 9.7 (History)Breakdown Of Expense Page

Personal Expense Tracker


HomeAdd ExpensesDashboardHistoryLimitReportLogout

EXPENSES

Expense Breakdown

Food	0
Entertainment	0
Business	0
Rent	0
EMI	2000
Other	0
Total	₹ 2000

25°C Cloudy



10:37 PM11/19/2022

## 9.8 Limit Page


Personal Expense Tracker

[Home](#) [Add Expenses](#) [Dashboard](#) [History](#) [Limit](#) [Report](#) [Logout](#)

Currently your MONTHLY limit is ₹ 100000  
ENTER the MONTHLY LIMIT to avoid over EXPENSES

ENTER

25°C  
Cloudy



10:38 PM  
11/19/2022

## 9.9 Monthly Report

Personal Expense Tracker


[Home](#) [Add Expenses](#) [Dashboard](#) [History](#) [Limit](#) [Report](#) [Logout](#)

Monthly Expense Breakdown

Expense Breakdown By Category: CURRENT MONTH

Food	0
Entertainment	0
Business	0
Rent	0
EMI	2000
Other	0
Total	₹ 2000

25°C  
Cloudy



10:39 PM  
11/19/2022

# 10. ADVANTAGES AND DISADVANTAGES

## 10.1 ADVANTAGES

One of the major pros of tracking spending is always being aware of the state of one's personal finances. Tracking what you spend can help you stick to your budget, not just in a general way, but in each category such as housing, food, transportation and gifts. While a con is that manually tracking all cash that is spent can be irritating as well as time consuming, a pro is that doing this automatically can be quick and simple

Another pro is that many automatic spending tracking software programs are available for free. Having the program on a hand-held device can be a main pro since it can be checked before spending occurs in order to be sure of the available budget. Another pro is that for those who just wish to keep tracking spending by hand with a paper and pen or by entering data onto a computer spreadsheet, these options are also available. Some people like to keep a file folder or box to store receipts and record the cash spent each day. A pro of this simple daily tracking system is that it can make one more aware of where the money is going way before the end of a pay period or month

## 10.2 DISADVANTAGES

A con with any system used to track spending is that one may start doing it then taper off until it's forgotten about all together. Yet, this is a risk for any new goal such as trying to lose weight or quit smoking. If a person first makes a budget plan, then places money in savings before spending any each new pay period or month, the tracking goal can help. In this way, tracking spending and making sure all receipts are accounted for only needs to be done once or twice a month. Even with constant tracking of one's spending habits, there is no guarantee that financial goals will be met. Although this can be considered to be a con of tracking spending, it could be changed into a pro if one makes up his or her mind to keep trying to properly manage all finances

## 11. CONCLUSION

A comprehensive money management strategy requires clarity and conviction for decision- making. You will need a defined goal and a clear vision for grasping the business and personal finances. That's when an expense tracking app comes into the picture. An expense tracking app is an exclusive suite of services for people who seek to handle their earnings and plan their expenses and savings efficiently. It helps you track all transactions like bills, refunds, payrolls, receipts, taxes, etc., on a daily, weekly, and monthly basis

## 12. FUTURE SCOPE

- Achieve your business goals with a tailored mobile app that perfectly fits your business.
- Scale-up at the pace your business is growing.
- Deliver an outstanding customer experience through additional control over the app.
- Control the security of your business and customer data.
- Open direct marketing channels with no extra costs with methods such as push notifications.
- Boost the productivity of all the processes within the organization.
- Increase efficiency and customer satisfaction with an app aligned to their needs.
- Seamlessly integrate with existing infrastructure
- Ability to provide valuable insights.
- Optimize sales processes to generate more revenue through enhanced data collection.
- Robo Advisors: Get expert investment advice and solutions with the Robo-advisors feature. This feature will analyze, monitor, optimize, and improve diversification in investments by turning data into actionable insights in real-time.
- 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 behavior. Moreover, it can recommend products and provide unique insights on saving money. It brings out the factors causing fluctuations in your expenses.
- Employee Travel Budgeting: Most businesses save money with a travel budgeting app as it helps prepare a budget for an employee's entire business trip. The feature will predict the expenses and allocate resources according to the prediction

## 13.APPENDIX

### SOURCE CODE

The source code has been uploaded in Github,

To refer the final source code click [SOURCE CODE](#)

### GITHUB & PROJECT DEMO VIDEO LINK

The Github Link : [Github](#)

The Project Video Link : [Video Link](#)