A PROJECT REPORT ON

PERSONAL EXPENSE TRACKER APPLICATION

NALAIYATHIRAN PROJECT BASED LEARNING ON PROFESSIONAL READINESS FOR INNOVATION, EMPLOYING AND ENTERPRENEURSHIP

DOMAIN: CLOUD APP DEVELOPMENT

TEAM ID: PNT2022TMID01341

SUBMITTED BY

MOHAN RAJ S(211419205106)

Department of Information Technology

Panimalar Engineering College

JAGADESH E (211419205072)

Department of Information Technology

Panimalar Engineering College

GANESH N(211419205050)

Department of Information Technology

Panimalar Engineering College

AHILESHWARAN S(211419205010)

Department of Information Technology

Panimalar Engineering College

PROJECT REPORT FORMAT

1. INTRODUCTION:

1.1 PROJECT OVERVIEW:

Web application are pinnacle in person comfort and feature lost sight of the internet packages in phrases of reputation and usability. There are various web packages that offer answers to control private and institution rate however not lots of them offer a complete view of each cases. In this, we develop a web application that continues document of person private fees, his/her contribution in institution expenditures, pinnacle funding options , view of the contemporary inventory marketplace, study authenticated economic information and take hold of the excellent ongoing gives with inside the marketplace in famous categories. The proposed website would dispose of messy sticky notes, spreadsheets confusion and statistics dealing with inconsistency issues even as supplying the excellent evaluation of your fees. The web application is developed to manage the daily expenses in a more efficient and manageable way.

The functions of the app are designed in a manner that will help you for higher finance management making plans so you can hold song of , examine and optimize your finances or spending's. In this utility we also are going to accumulate user's facts with authenticated permissions and examine and observe their sample charges in sure class or through awesome varieties of spending that may be used for reading markel generate file on the stop of month to reveal Income-Expense Curve. It will allow you to upload the financial savings amount that you had stored for a few unique Festivals or day like Birthday or Anniversary.

1.2 PURPOSE:

People without financial knowledge don't know where they are spending their money unnecessarily , so they end up spending a lot of money without even thinking about it. So to help the people keep track of their income and expenses , with the help of this app the user can track all their spending habbits and this app displays the information in a graphical manner so that the user can easily understand. This app will also send user an email alert if their expenses exceeds the budget amount. With this the user don't have to worry about spending unnecessarily . The app will take care of tracking of all the expenses .So with this the user can be financially disciplined.

2. LITERATURE SURVEY:

2.1 EXISTING PROBLEM:

Currently, we must maintain Excel sheets, CSV files, and other documents for each user's daily and monthly spending. There is currently no such comprehensive way to conveniently keep track of daily expenses. To achieve this, a person must maintain a log in. All computations must be performed by the user in either a diary or a computer, which occasionally leads to errors resulting in losses.

Using a manual accounting system may have a number of drawbacks. Accounting may be a challenge for any organisation complicated task. You must have a thorough understanding of the accounting process to use a manual accounting system possibly not required with a computerised.

A mobile application called The Expense Manager is designed to run on Android-based smartphones. By removing imparting costs and settling friendship pledges, Expense Manager is intended to efficiently meet user wants. The software encourages corresponding users to assist in determining who owes whom what and for what. The idea is to develop improved methods to make it easier for consumers and their friends to split costs. This new application will enable group users and their friends to view individual expenditures in detail inside the application. By clicking on the expense name in any cost list, the app's users

Some of the traditional approaches used to address this issue in everyday situations include using sticky notes by regular users, spreadsheets by proficient persons to track spending, and ledgers by specialists to manage enormous volumes of data. This demonstrates that various people employ a variety of ways. This results in inconsistent use of the data. There are still issues in the areas of inconsistent data, the possibility of missing important inputs, and the possibility of human mistake.

2.2 REFERENCES

[1] Sabab, S. A., Islam, S. S., Rana, M. J., & Hossain,M. (2018, September). eExpense: A smart approach to track everyday expense. In 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT) (pp. 136-141). IEEE.

[2] Rajaprabha, M. N. (2017). Family Expense Manager Application in Android. MS&E, 263(4), 042050.

[3] Kan, C., Lynch, J., & Fernbach, P. (2015). How budgeting helps consumers achieve financial goals. ACR North American Advances.

[4] Sharma, R., 2020. Case Study Of Expense Tracking App: Get Daily Alerts Of Your Expense. [online] Medium.

[5] Thanapal, M. P., Patel, Y., Lokesh, R. T. P., & Satheesh, K. J. (2015). Income and expense tracker. Indian Journal of Science and Technology, 8(S2), 118-122.

[6]Manchanda, A. (2012). Expense Tracker Mobile Application (Doctoral dissertation, San Diego State University).

2.3 PROBLEM STATEMENT DEFINITION

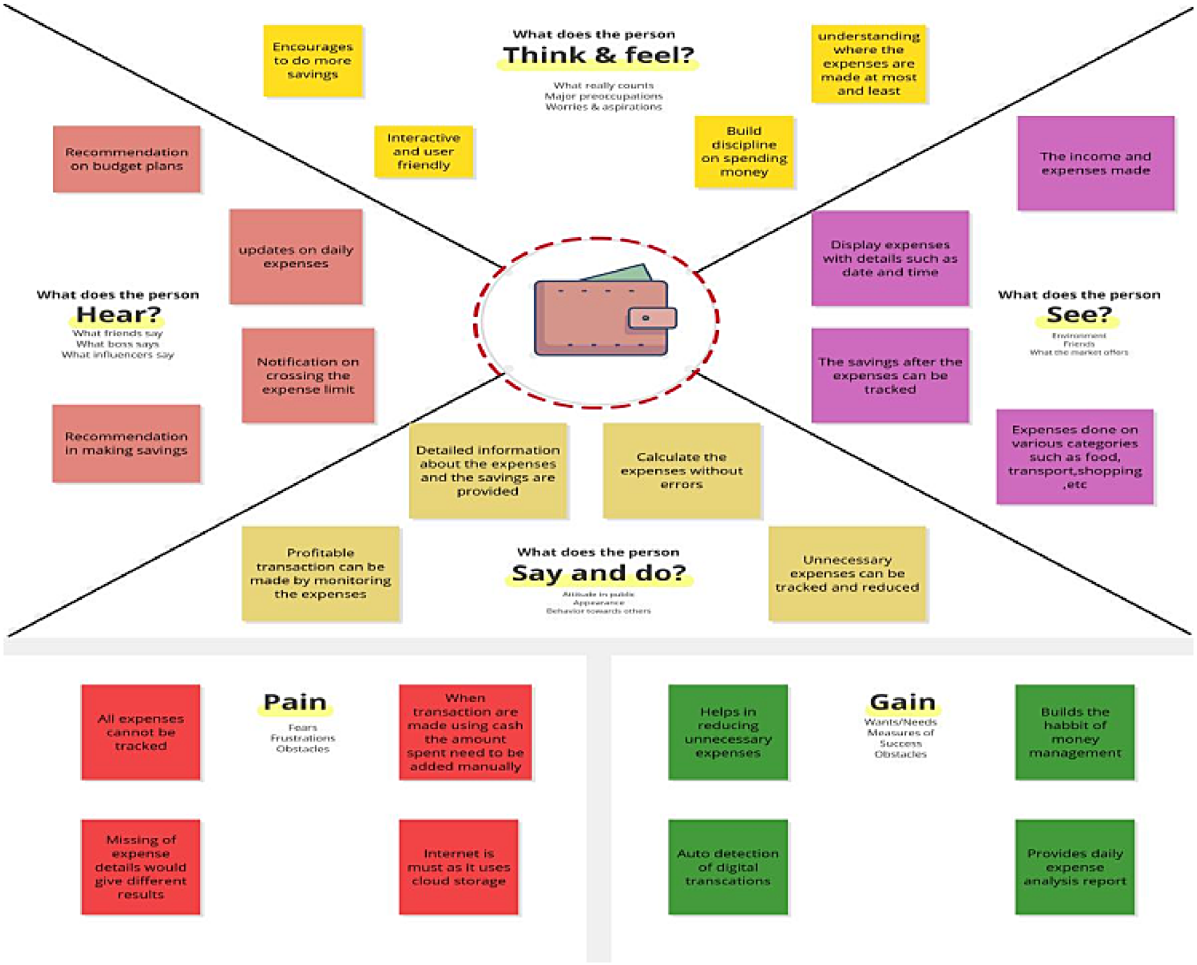
The price tracker is an web app which runs on all web platforms. It allows person to control all their expenses in an effective manner and it helps to budget and save money.This might avoid price range managing problems and offers us green effects on our savings.In everyone's life, cash performs an essential role. A person who can't control his costs can't efficiently lead a household and satisfy his goals. In the present day global where mobile telephones and laptops have come to be part of living,such an app might be available to address all our costs. A individual commonly can't preserve music of all his expenses through the conventional pen and paper technique and might miss some of his small expenses and might even miss some bills. Such a scenario will in no way rise up while we use an app. We could make smooth comparisons through seeing the graphs,that's not possible with inside the rigorous methods.

All Android platforms can operate the expense tracker android app. It enables users to efficiently manage all of their expenses and keep tabs on their spending. This would prevent. budgeting challenges and provides us with effective savings solutions.Money plays a big part in everyone's life. A person who is unable to control his spending will be unable to successfully run a family and achieve his objectives. Such an app would be useful to manage all of our spending in the modern world when smartphones and laptops have become commonplace.

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

Teams can utilize an empathy map as a collaborative tool to learn more about their clients. Below is the empathy map for our project.

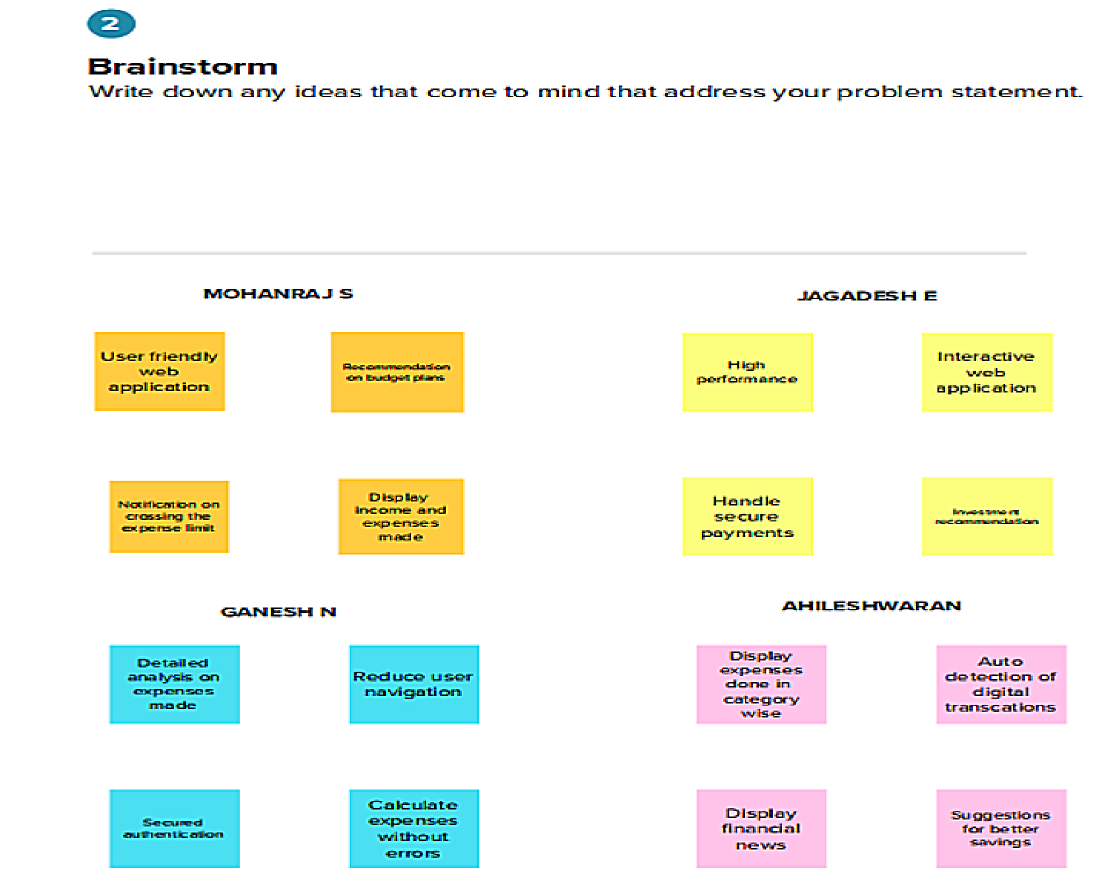

What does the person
Think & feel?
What really counts
Major preoccupations
Worries & aspirations
Encourages to do more savings
Interactive and user friendly
Build discipline on spending money
understanding where the expenses are made at most and least
What does the person
Hear?
What friends say
What boss says
What influencers say
Recommendation on budget plans
updates on daily expenses
Notification on crossing the expense limit
Recommendation in making savings
What does the person
See?
Environment
Friends
What the market offers
The income and expenses made
Display expenses with details such as date and time
The savings after the expenses can be tracked
Expenses done on various categories such as food, transport,shopping ,etc
What does the person
Say and do?
Attitude in public
Appearance
Behavior towards others
Detailed information about the expenses and the savings are provided
Calculate the expenses without errors
Profitable transaction can be made by monitoring the expenses
Unnecessary expenses can be tracked and reduced
Pain
Fears
Frustrations
Obstacles
All expenses cannot be tracked
When transaction are made using cash the amount spent need to be added manually
Missing of expense details would give different results
Internet is must as it uses cloud storage
Gain
Wants/Needs
Measures of Success
Obstacles
Helps in reducing unnecessary expenses
Builds the habbit of money management
Auto detection of digital transcations
Provides daily expense analysis report

3.2 IDEATION & BRAINSTORMING

Brainstorm, Idea Listing and Grouping:

Brainstorm is nothing but to suggest idea for the project before starting the project. The process of brainstorming can assist the group focus its ideas and find solutions.You can start drafting proposals for upcoming research funding applications using project ideas. In this phase, you're gathering crucial project-related information and looking for collaborators, potential funders, budget information, and metadata linked to the project. You may also create tasks and distribute them to participants in the project.

A project group is an organizational grouping of projects. Administrative actions can affect multiple projects and users at once thanks to project groups.



2

Brainstorm

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

MOHANRAJ S

User friendly web application

Recommendation on budget plans

Notification on crossing the expense limit

Display income and expenses made

JAGADESH E

High performance

Interactive web application

Handle secure payments

Investment recommendation

GANESH N

Detailed analysis on expenses made

Reduce user navigation

Secured authentication

Calculate expenses without errors

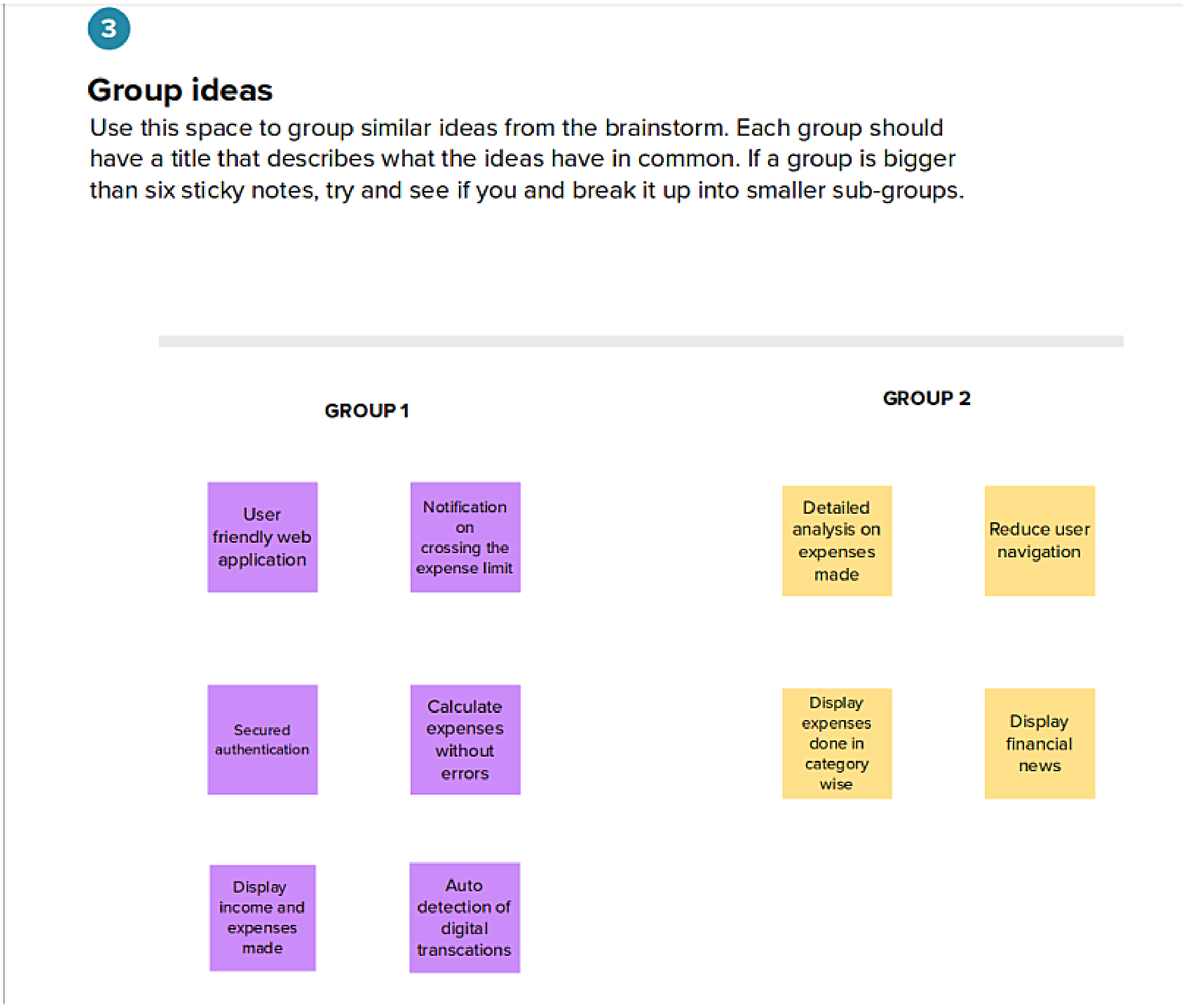
AHILESHWARAN

Display expenses done in category wise

Auto detection of digital transcations

Display financial news

Suggestions for better savings

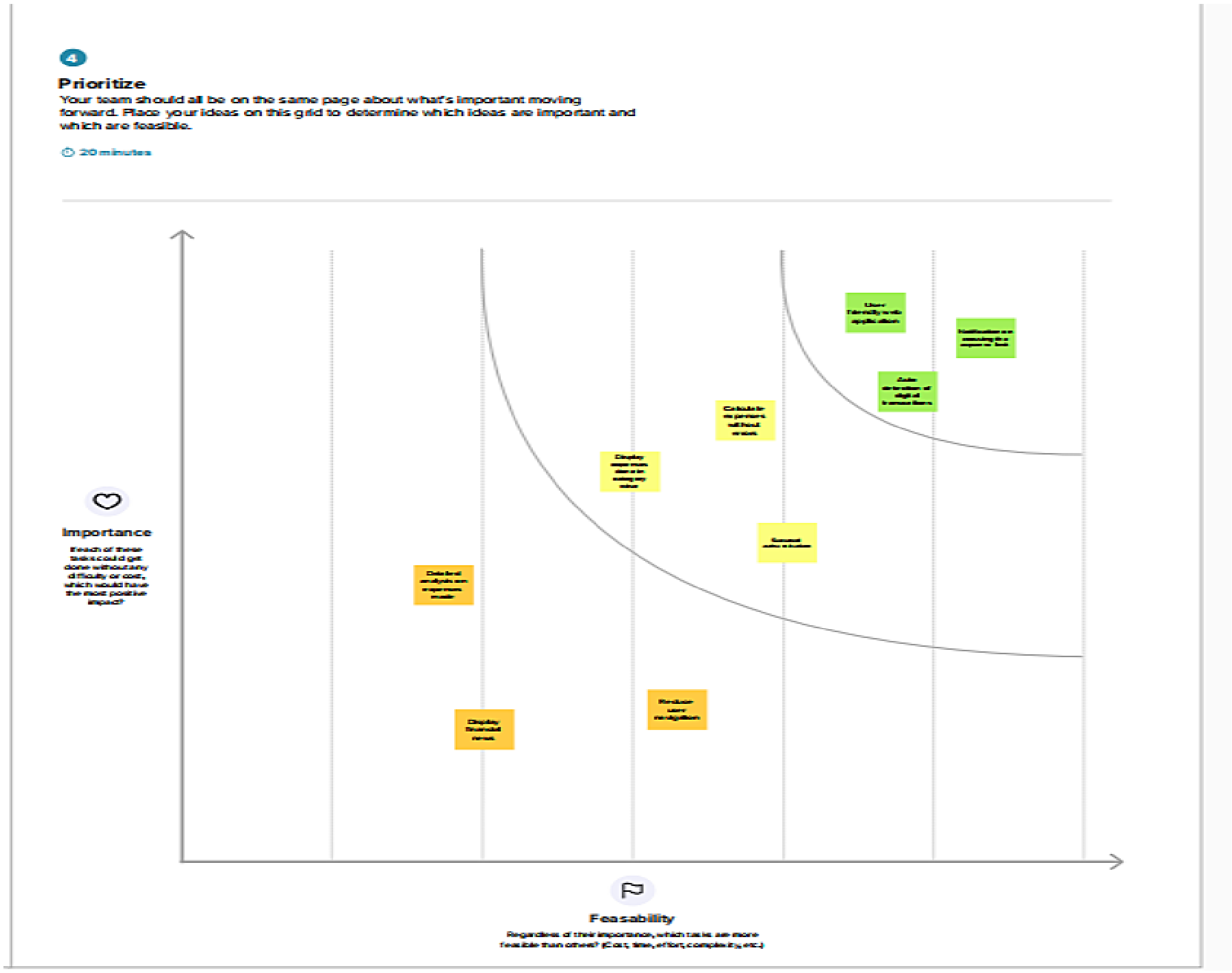


3

Group ideas

Use this space to group similar ideas from the brainstorm. Each group should have a title that describes what the ideas have in common. If a group is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

GROUP 1
User friendly web application
Notification on crossing the expense limit
Secured authentication
Calculate expenses without errors
Display income and expenses made
Auto detection of digital transcations
GROUP 2
Detailed analysis on expenses made
Reduce user navigation
Display expenses done in category wise
Display financial news



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

Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasability
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)
User friendly web application
Notifications on exceeding the expense limit
Auto detection of digital transactions
Calculate expenses without errors
Display expenses data in category wise
Secured online solution
Detailed analysis on expenses made
Display financial news
Reduce user navigation

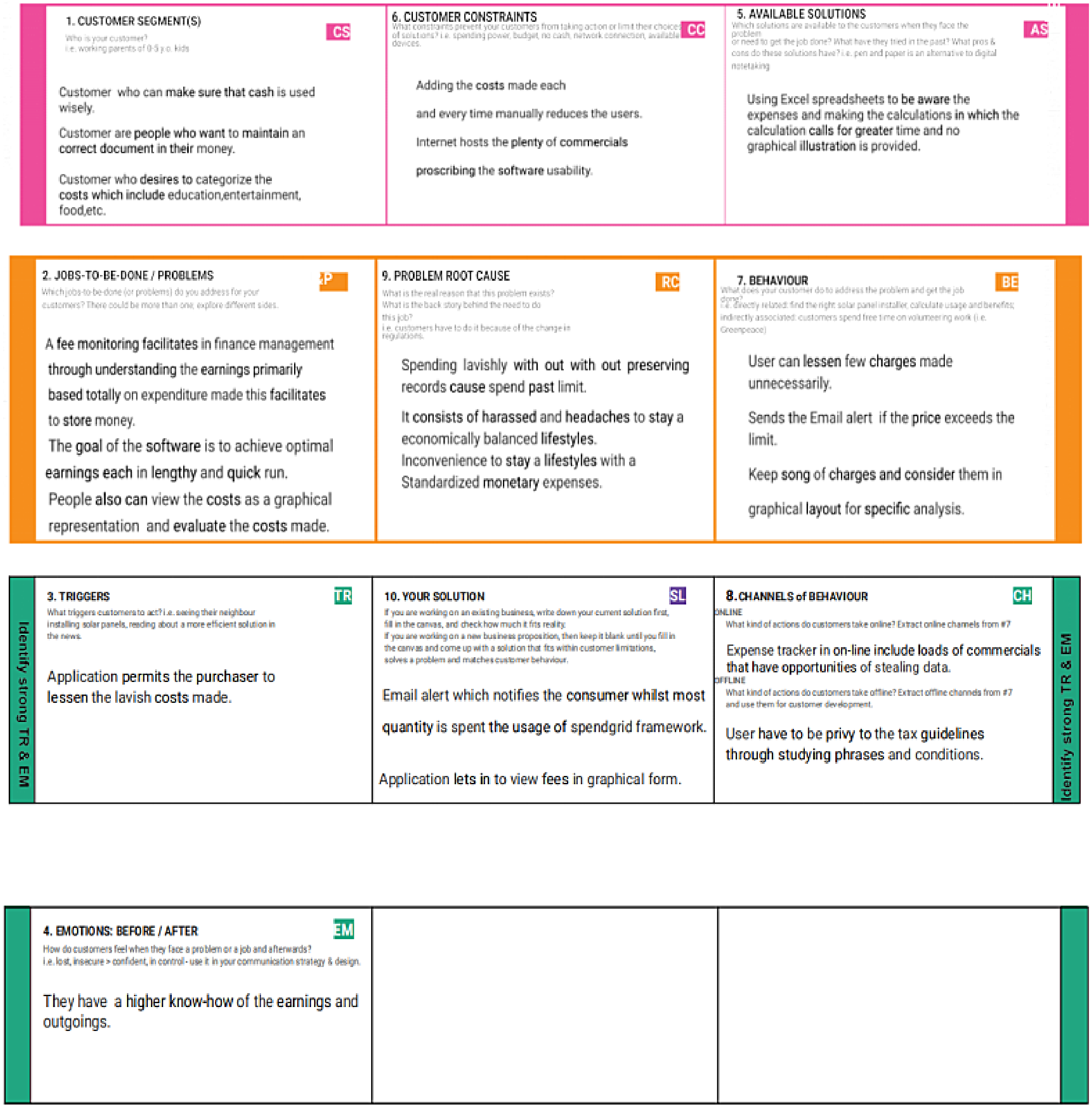
3.3 PROPOSED SOLUTION

The technical solution that the implementation agency will present in response to the requirements and goals of the project is referred to as the proposed solution. The following is the project's suggested solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	This might avoid price range managing problems and offers us green effects on our savings. In everyone’s life, cash performs an essential role.
2.	Idea / Solution description	The predominant concept is to permit customers to set limits for their Expenses and Alert them via Mail When the Limit Exceeds.
3.	Novelty / Uniqueness	Detailed analyses of what and the way that the consumer spends, and all of the spending behaviour can be tracked and consequently adjusted. The weekly, monthly, and year-smart contrast of expenses may be and could allow the consumer know thru Alerts When the restriction exceeds. When you tune your expenses, you take control of your finances. It helps you to alter spending impulses and cast off nugatory spending, thereby averting debt. At each point, you will be aware of how an awful lot cash you're left with.
4.	Social Impact / Customer Satisfaction	Using this software eases the tiring and complicated procedure of retaining tune of one’s costs and allows them refer their transactions everywhere each time from their phones.
5.	Business Model (Revenue Model)	Through our utility the sales for the organization can be withinside the shape of subscription plans. Makes the consumer realize approximately what are all the good stuff and trending approaches to invest cash appropriately and securely.
6.	Scalability of the Solution	The method can cope with a massive range of users given that it's miles primarily based totally at the cloud device. New functions can additionally be delivered every time as in keeping with requirement.

3.4 PROBLEM SOLUTION FIT

Proposed solution fit is nothing but identify an existing problem and to solve it in with a solution that customers find useful and satisfying.



1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

Customer who can make sure that cash is used wisely.

Customer are people who want to maintain an correct document in their money.

Customer who desires to categorize the costs which include education,entertainment, food,etc.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

Adding the costs made each

and every time manually reduces the users.

Internet hosts the plenty of commercials

proscribing the software usability.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem
or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

Using Excel spreadsheets to be aware the expenses and making the calculations in which the calculation calls for greater time and no graphical illustration is provided.

2. JOBS-TO-BE-DONE / PROBLEMS P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

A fee monitoring facilitates in finance management through understanding the earnings primarily based totally on expenditure made this facilitates to store money.

The goal of the software is to achieve optimal earnings each in lengthy and quick run.

People also can view the costs as a graphical representation and evaluate the costs made.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

Spending lavishly with out with out preserving records cause spend past limit.

It consists of harassed and headaches to stay a economically balanced lifestyles.
Inconvenience to stay a lifestyles with a Standardized monetary expenses.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

User can lessen few charges made unnecessarily.

Sends the Email alert if the price exceeds the limit.

Keep song of charges and consider them in

graphical layout for specific analysis.

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

Application permits the purchaser to lessen the lavish costs made.

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

Email alert which notifies the consumer whilst most

quantity is spent the usage of spendgrid framework.

Application lets in to view fees in graphical form.

8. CHANNELS of BEHAVIOUR CH

NLINE
What kind of actions do customers take online? Extract online channels from #7

Expense tracker in on-line include loads of commercials that have opportunities of stealing data.

FFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

User have to be privy to the tax guidelines through studying phrases and conditions.

Identify strong TR & EM

4. EMOTIONS: BEFORE / AFTER EM

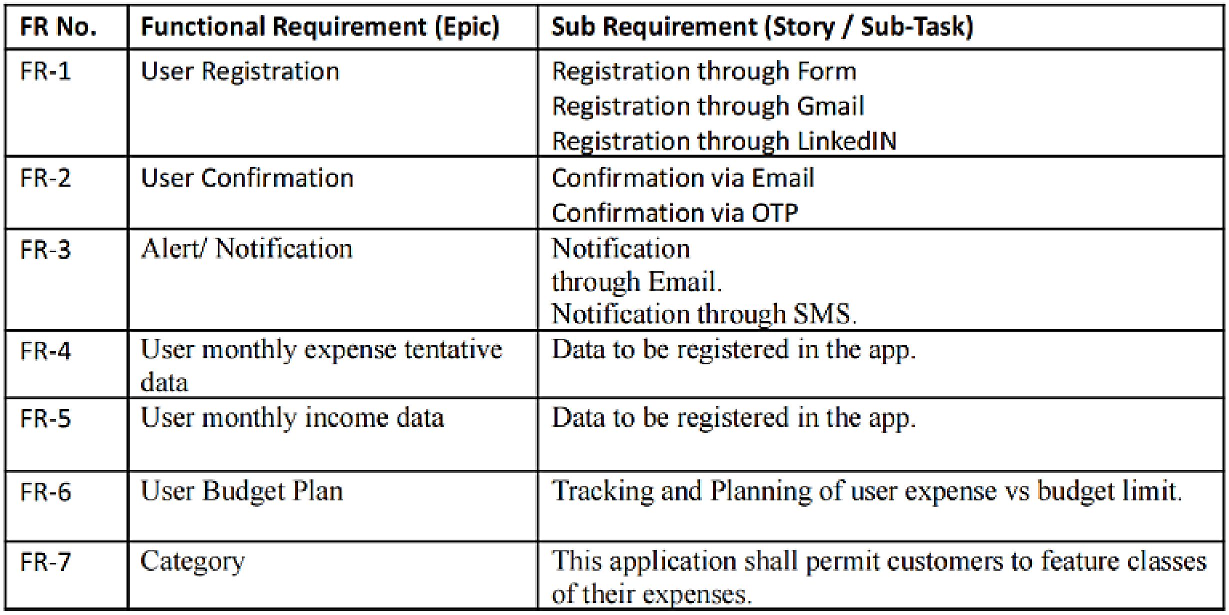
How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

They have a higher know-how of the earnings and outgoings.

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

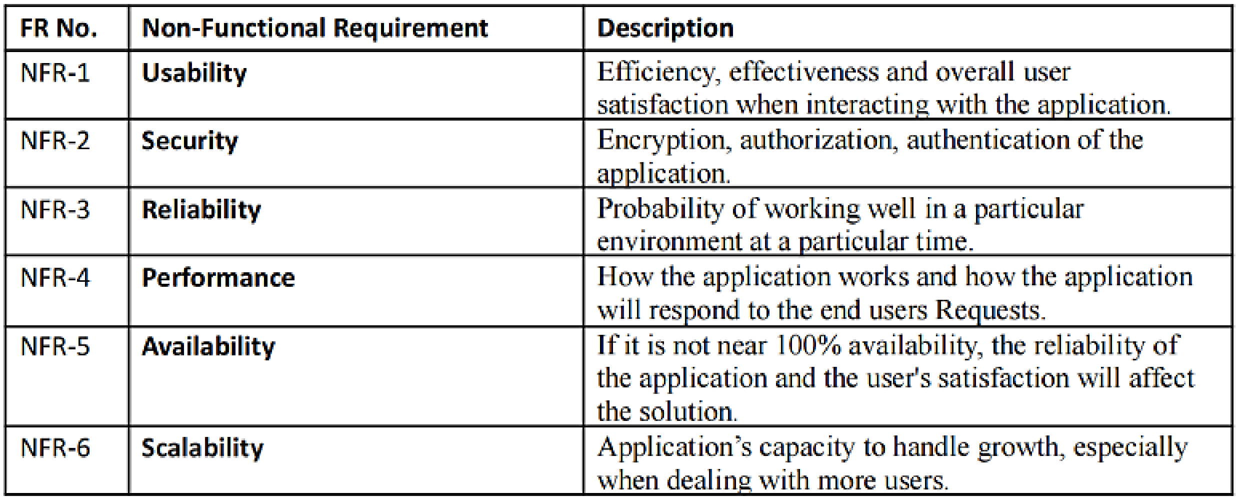
In software development and systems engineering, the desired operations of a program or system are referred to as functional requirements. Users cannot do their tasks without developers implementing certain features or functions into a product. Making them clear is essential for both the development team and the stakeholders. Functional requirements frequently describe the behaviour of a system in specific situations.



FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Alert/ Notification	Notification through Email. Notification through SMS.
FR-4	User monthly expense tentative data	Data to be registered in the app.
FR-5	User monthly income data	Data to be registered in the app.
FR-6	User Budget Plan	Tracking and Planning of user expense vs budget limit.
FR-7	Category	This application shall permit customers to feature classes of their expenses.

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements list the core attributes of a system. Sometimes, people refer to them as characteristics. The system's usability, scalability, maintainability, and performance are among the characteristics that are defined. They serve as restrictions or limitations on how the system is built for the different backlogs.

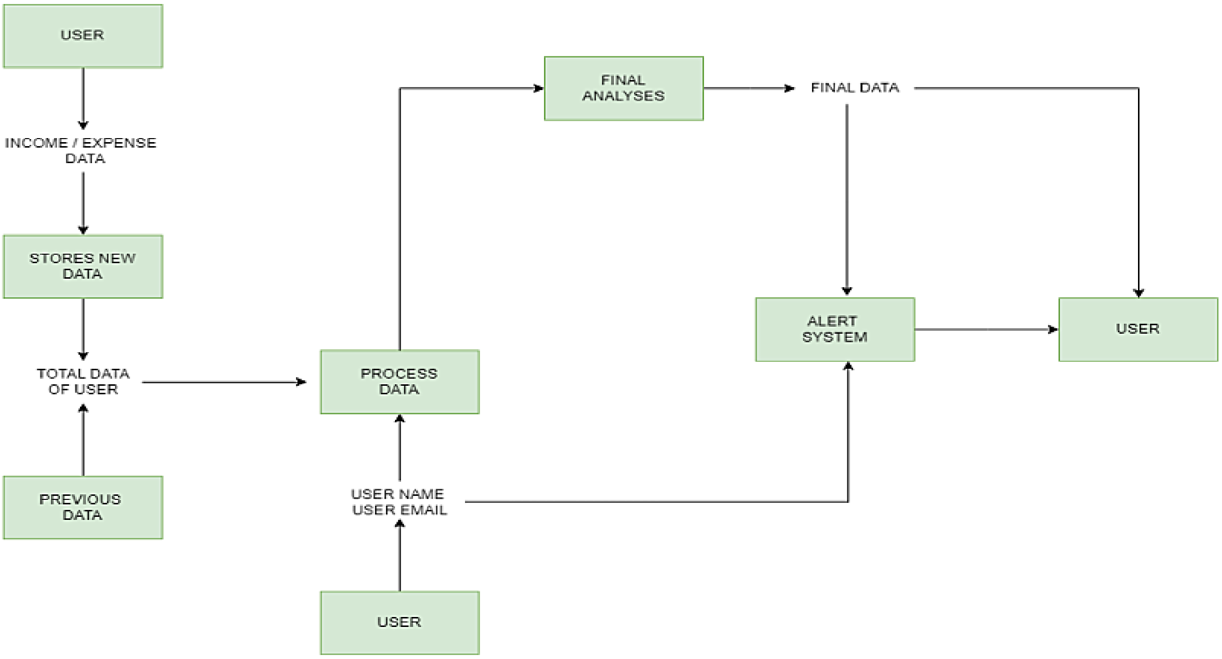


FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Efficiency, effectiveness and overall user satisfaction when interacting with the application.
NFR-2	**Security**	Encryption, authorization, authentication of the application.
NFR-3	**Reliability**	Probability of working well in a particular environment at a particular time.
NFR-4	**Performance**	How the application works and how the application will respond to the end users Requests.
NFR-5	**Availability**	If it is not near 100% availability, the reliability of the application and the user's satisfaction will affect the solution.
NFR-6	**Scalability**	Application's capacity to handle growth, especially when dealing with more users.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

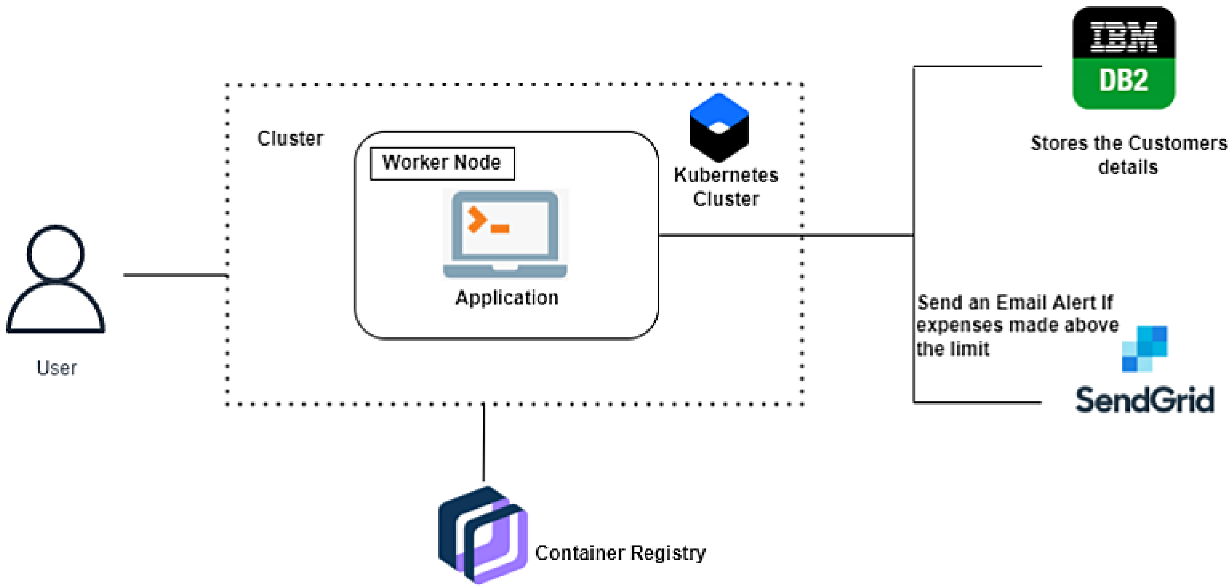
Data Flow Diagrams It demonstrates the many types of data that will be input into and exported from the system, as well as where the data will be stored. A DFD is frequently an expansion of a context diagram to reveal more of the system's finer details that were initially depicted by the context diagram.



5.2 SOLUTION & TECHNICAL ARCHITECTURE

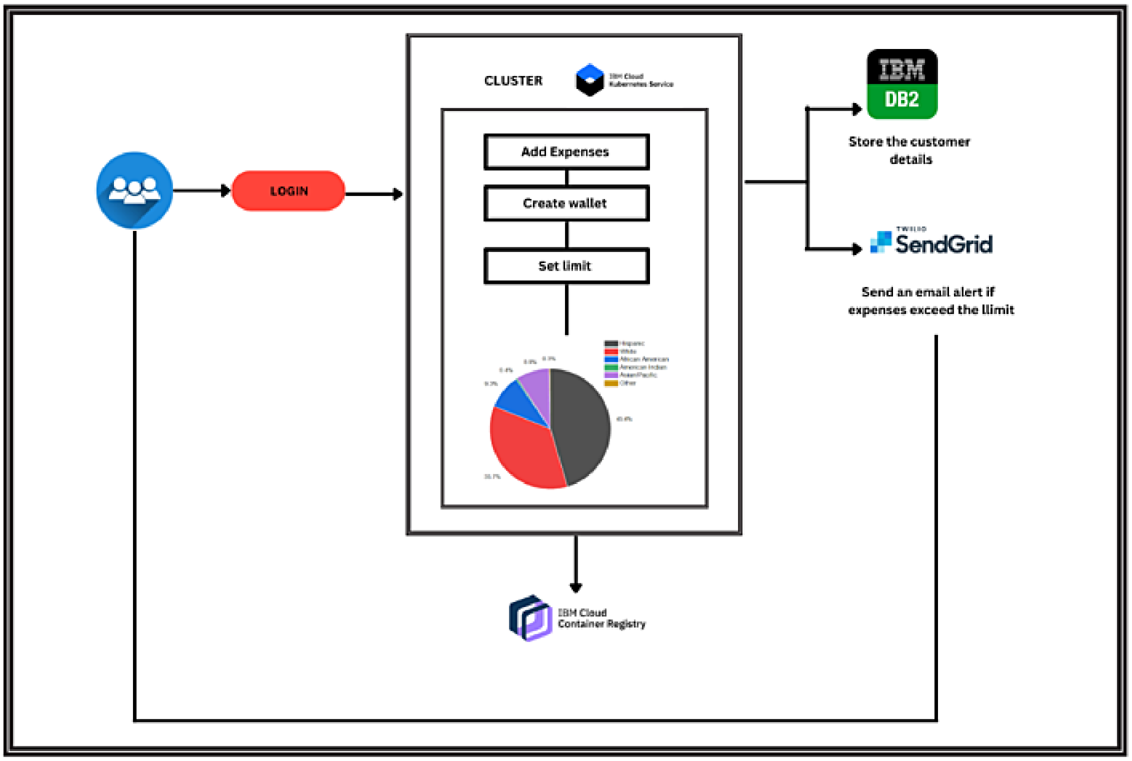
TECHNICAL ARCHITECTURE

The main system components, their connections, and the agreements that specify how the components interact are all included in the technical architecture. The objective of technical architects is to fulfil all business requirements with an application that is both performance- and security-optimized. creating the framework for technological systems. controlling the execution of programs. collaborating with the software development group to make sure the system functions properly.



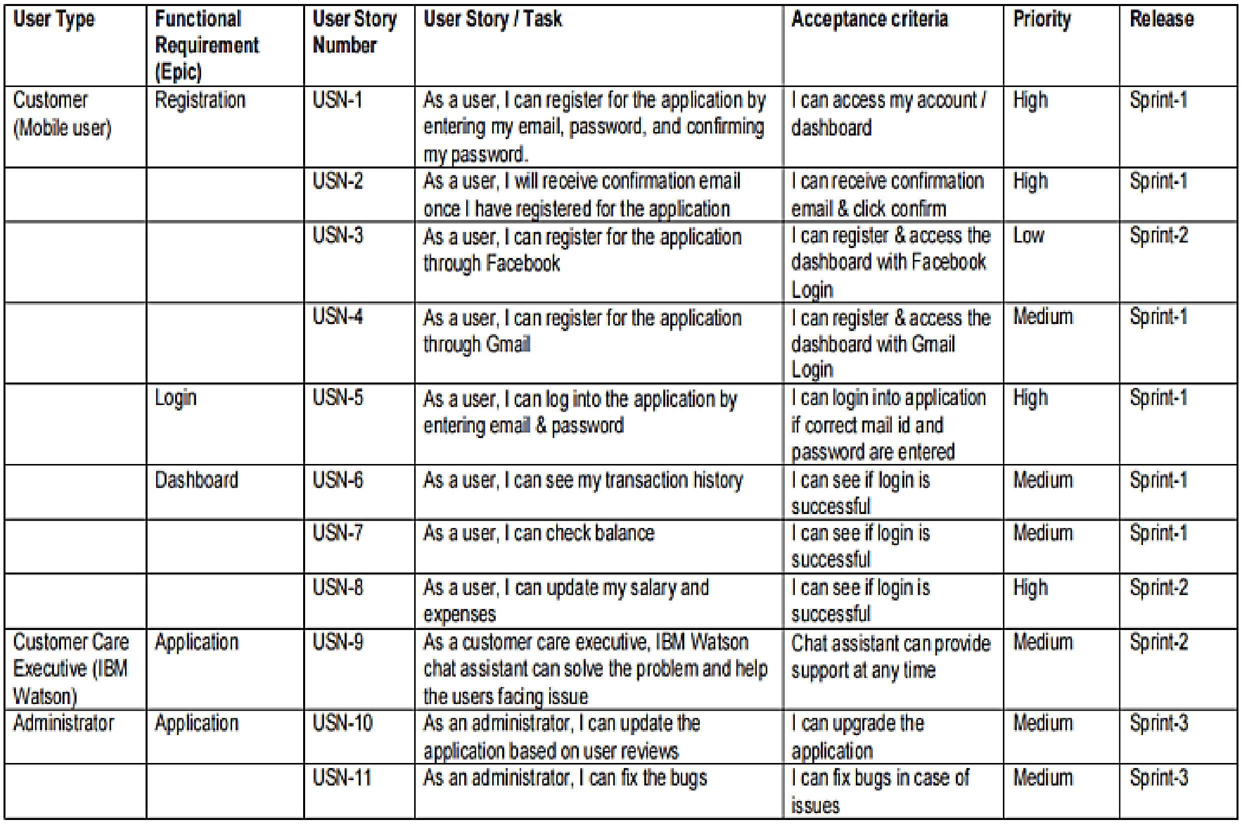
SOLUTION ARCHITECTURE

Solution architecture lays the groundwork for software development activities by specifying the functional requirements and implementation phases of IT systems and tailoring them to specific business objectives. It is broken down into a variety of subprocesses, each of which is influenced by a distinct viewpoint on corporate architecture. It is possible to ensure that a new system will function properly in the existing business environment thanks to the solution architecture. In order to perform this function, a solution architect must be aware of how operating systems, application structures, and processes interact. The following describes the solution architecture for our project:


LOGIN
CLUSTER
IBM Cloud Kubernetes Service
Add Expenses
Create wallet
Set limit
Hispanic
White
African American
American Indian
Asian/Pacific
Other
IBM
DB2
Store the customer details
TWILIO
SendGrid
Send an email alert if expenses exceed the llimit
IBM Cloud Container Registry

5.3 USER STORIES

An informal, comprehensive description of a software feature written from the client's or end user's perspective is known as a "user story." The purpose of a user story is to explain how a piece of work will give the client a particular value. The main benefit of employing user stories in agile product development may be that they are not designed to stand alone, unlike requirements or use cases. Instead, each user narrative serves as a pending topic for discussion with the development team.

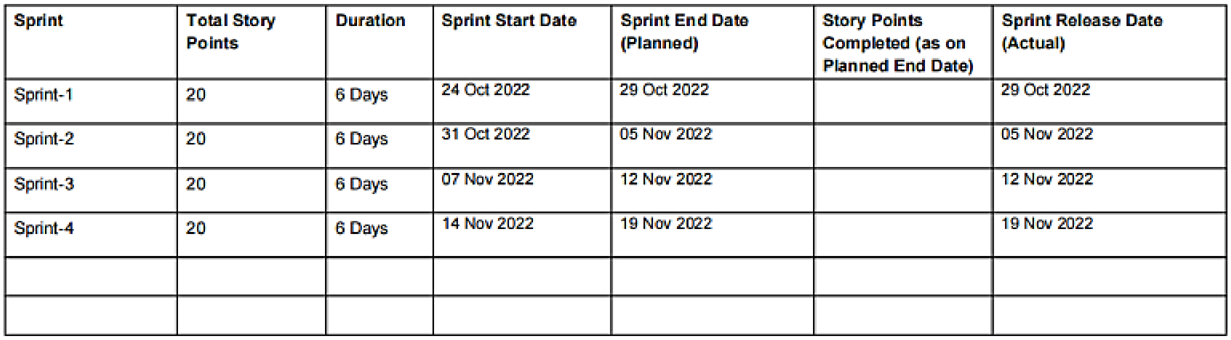


User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login into application if correct mail id and password are entered	High	Sprint-1
	Dashboard	USN-6	As a user, I can see my transaction history	I can see if login is successful	Medium	Sprint-1
		USN-7	As a user, I can check balance	I can see if login is successful	Medium	Sprint-1
		USN-8	As a user, I can update my salary and expenses	I can see if login is successful	High	Sprint-2
Customer Care Executive (IBM Watson)	Application	USN-9	As a customer care executive, IBM Watson chat assistant can solve the problem and help the users facing issue	Chat assistant can provide support at any time	Medium	Sprint-2
Administrator	Application	USN-10	As an administrator, I can update the application based on user reviews	I can upgrade the application	Medium	Sprint-3
		USN-11	As an administrator, I can fix the bugs	I can fix bugs in case of issues	Medium	Sprint-3

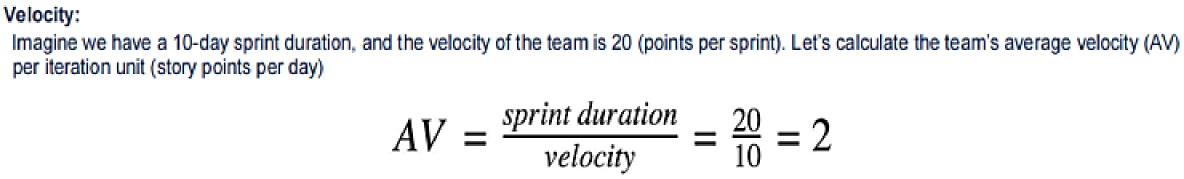
6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

In the scrum process, sprint planning marks the beginning of the sprint. Sprint planning's goal is to specify what can be completed in a sprint and how it will be done. The entire scrum team collaborates on sprint planning.



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



Velocity:

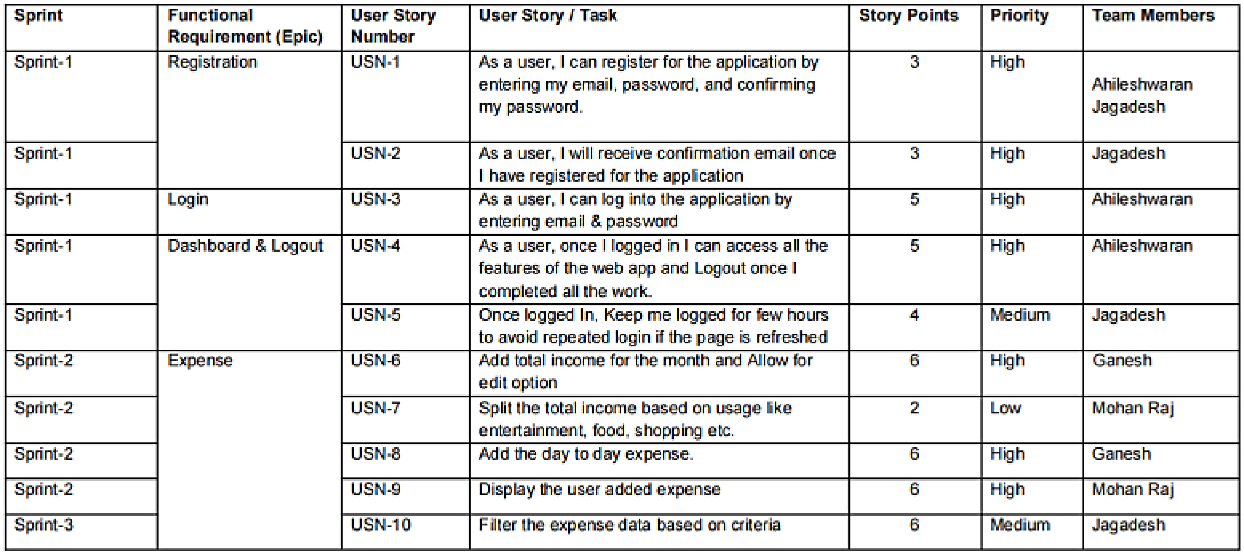
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

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

6.2 SPRINT DELIVERY SCHEDULE

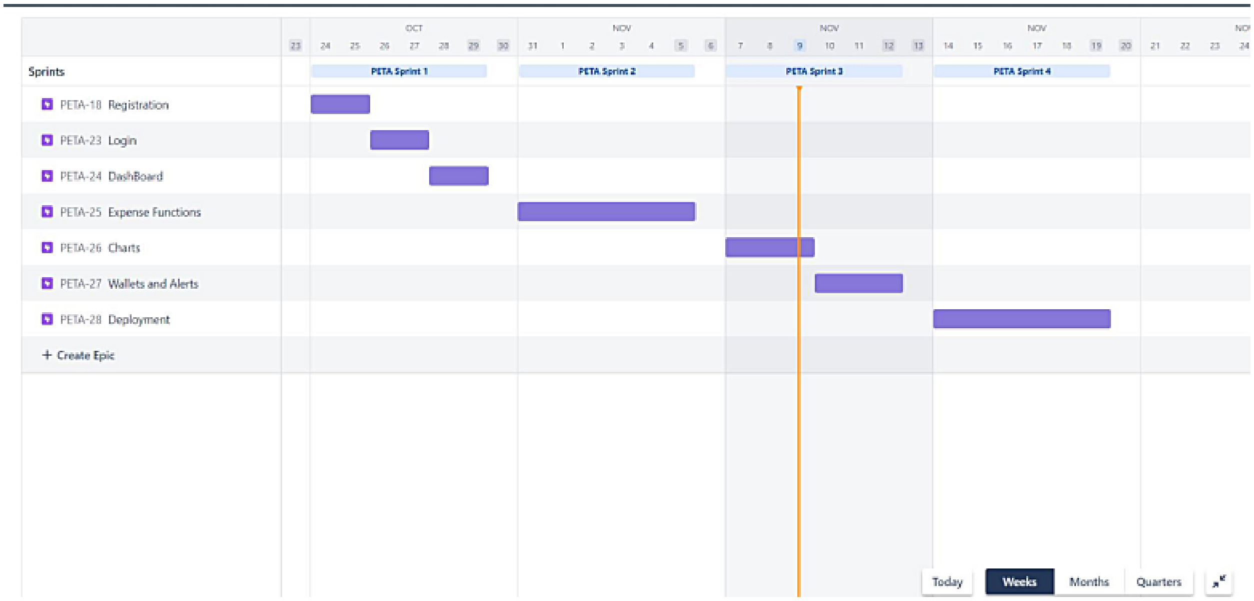
A sprint schedule is a written description of the entire sprint planning process. It's one of the initial steps in the agile sprint planning process, and it calls for sufficient investigation, preparation, and coordination. It centres on a product backlog, which is a list of open requests fordevelopment and iteration.

A burndown chart, which displays how rapidly a team is progressing through a customer's user stories, is a project management chart. This agile tool records the description of a feature from the viewpoint of the end user and compares the overall effort to the quantity of work for each agile sprint.



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.	3	High	Ahileshwaran Jagadesh
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	3	High	Jagadesh
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	5	High	Ahileshwaran
Sprint-1	Dashboard & Logout	USN-4	As a user, once I logged in I can access all the features of the web app and Logout once I completed all the work.	5	High	Ahileshwaran
Sprint-1		USN-5	Once logged In, Keep me logged for few hours to avoid repeated login if the page is refreshed	4	Medium	Jagadesh
Sprint-2	Expense	USN-6	Add total income for the month and Allow for edit option	6	High	Ganesh
Sprint-2		USN-7	Split the total income based on usage like entertainment, food, shopping etc.	2	Low	Mohan Raj
Sprint-2		USN-8	Add the day to day expense.	6	High	Ganesh
Sprint-2		USN-9	Display the user added expense	6	High	Mohan Raj
Sprint-3		USN-10	Filter the expense data based on criteria	6	Medium	Jagadesh

6.3 REPORTS FROM JIRA



Sprints

PETA Sprint 1

PETA Sprint 2

PETA Sprint 3

PETA Sprint 4

PETA-18 Registration

PETA-23 Login

PETA-24 DashBoard

PETA-25 Expense Functions

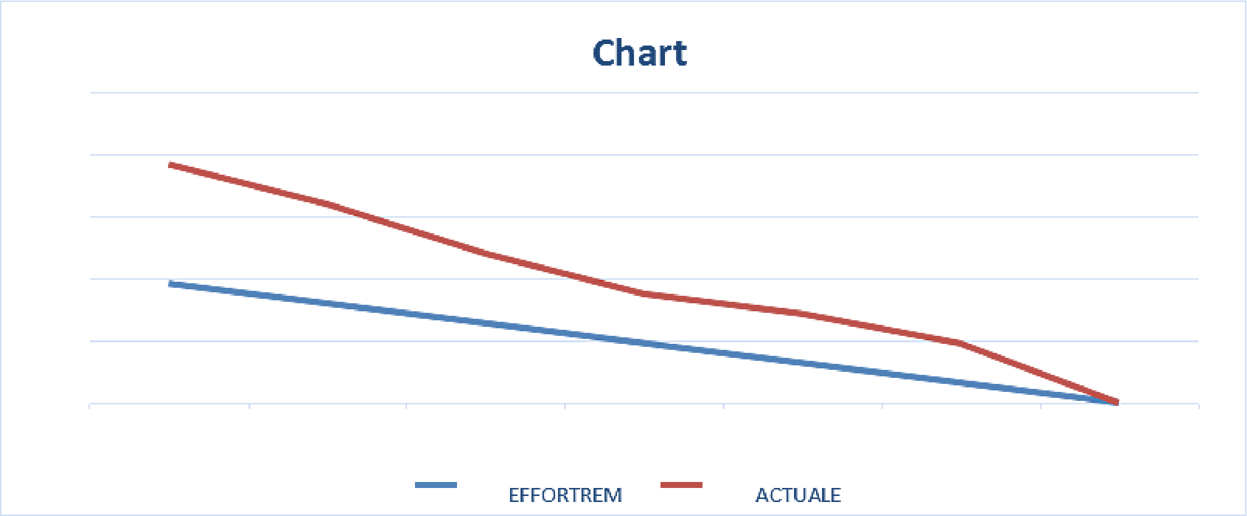
PETA-26 Charts

PETA-27 Wallets and Alerts

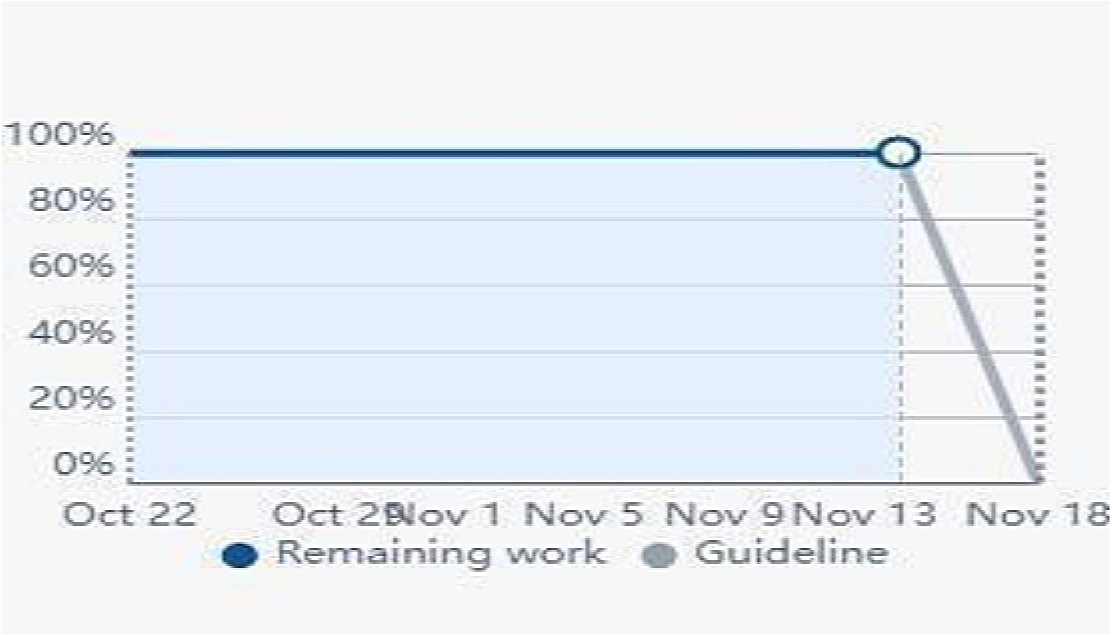
PETA-28 Deployment

+ Create Epic

Burn down Chart



Sprint Burn down chart



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

7.1 FEATURE 1

HOME

Home page of our Budget app the user can sign up or login to redirect to next page by clicking the relevant button.

SIGN UP

The user can create an account by clicking sign up button using sign up page and enter all details for sign up

LOGIN

Once the user can created the account then the user will be redirected to login page and enter all the details in login page and then click login button

Login.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">



    <!-- CSS only -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>
```

```
<link rel="stylesheet" href="/static/css/styles.css">
<style>
  input,h6{
    margin:20px;
  }
  input{
    height:40px;
    width: 300px;
  }
</style>
    <title>Document</title>
</head>
<body>
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
        <div class="container-fluid">
          <a class="navbar-brand" href="#">Expense Tracker</a>
                      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
```

```
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">

          <li class="nav-item">
            <a class="nav-link" href="/register">Register</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/">Login</a>
          </li>
           </ul>
      </div>
    </div>
  </nav>
  <br>
  <br>
  <br>
  <div class="content">
  <!-- content  -->
<h1>Login</h1>
```

```
    {% if error %}

    <div class="alert alert-danger" role="alert">

      {{error}}

    </div>

    {% endif %}

<div class="form" >

      <form style="width: 500px;height:auto; padding: 20px;" action="/" method="post">

        <h6>UserName</h6>

        <input type="text" name="Uname" id="">

        <h6>Password</h6>

        <input type="password" name="pass" id="">

          <a href="/forgotpassword"><p>Forgot Password</p></a>

            <button type="submit" class="btn btn-dark">Login</button>

      </form>

    </div

</div

</body>

</html>
```

Signup.html

```html
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">



    <!-- CSS only -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>



<link rel="stylesheet" href="/static/css/styles.css">

<!-- <link rel="JavaScript" href="/static/js/script.js"> -->

<style>

   .btn{
    display: block;

    margin: 2px auto;

  }

</style>

    <title>Document</title>

</head>

<body >

    <!-- navbar  -->

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">

        <div class="container-fluid">

          <a class="navbar-brand" href="#">Expense Tracker</a>

          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

          </button>

          <div class="collapse navbar-collapse" id="navbarNav">

            <ul class="navbar-nav">



              <li class="nav-item">

                <a class="nav-link" href="/register">Register</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/">Login</a>

            </ul>
          </div>
        </div> </nav>
<br>  <br> <br>
      <div class="content register">
      <!-- content  -->
    <h1>Register</h1>
{% if error %}
<div class="alert alert-danger" role="alert">
  {{error}}
</div>
{% endif %}
<div class="form">
      <div class="register-form container">
                      <form    name="myForm"    action="/register"    onsubmit="return    validateForm()"
method="post">
```

```
        <h6>UserName</h6>
        <input type="text"  id="usrname" name="Uname" id="">
        <h6>Email</h6>
        <input type="email"  id="email" name="email" id="">

        <h6>Password</h6>
        <!-- <input type="password" name="pass" id=""> -->
                <input  type="password"  id="psw"  name="pass"  pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}" title="Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more characters" required>
            <button type="submit" class="btn btn-dark">Register</button>
      </form>
    </div>
  </div>
    <div id="message">
      <h3>Password must contain the following:</h3>
      <p id="letter" class="invalid">A <b>lowercase</b> letter</p>
      <p id="capital" class="invalid">A <b>capital (uppercase)</b> letter</p>
      <p id="number" class="invalid">A <b>number</b></p>
      <p id="length" class="invalid">Minimum <b>8 characters</b></p>
    </div>
```

```
</div>
  <script>
   var uname = document.getElementById("usrname");
   var myInput = document.getElementById("psw");
   var letter = document.getElementById("letter");
   var capital = document.getElementById("capital");
   var number = document.getElementById("number");
   var length = document.getElementById("length");
   // When the user clicks on the password field, show the message box
   myInput.onfocus = function() {
    document.getElementById("message").style.display = "block";
   }
   // When the user clicks outside of the password field, hide the message box
   myInput.onblur = function() {
    document.getElementById("message").style.display = "none";
   }
   // When the user starts to type something inside the password field
   myInput.onkeyup = function() {
```

```
// Validate lowercase letters
var lowerCaseLetters = /[a-z]/g;
if(myInput.value.match(lowerCaseLetters)) {
  letter.classList.remove("invalid");
  letter.classList.add("valid");
} else {
  letter.classList.remove("valid");
  letter.classList.add("invalid");
}
// Validate capital letters
var upperCaseLetters = /[A-Z]/g;
if(myInput.value.match(upperCaseLetters)) {
  capital.classList.remove("invalid");
  capital.classList.add("valid");
} else {
  capital.classList.remove("valid");
  capital.classList.add("invalid");

// Validate numbers
```

```
      var numbers = /[0-9]/g;
      if(myInput.value.match(numbers)) {
        number.classList.remove("invalid");
        number.classList.add("valid");
      } else {
        number.classList.remove("valid");
        number.classList.add("invalid");


      // Validate length
      if(myInput.value.length >= 8) {
        length.classList.remove("invalid");
        length.classList.add("valid");
      } else {
        length.classList.remove("valid");
        length.classList.add("invalid");
      }
    }
    function validateForm() {
let x = document.forms["myForm"]["Uname"].value;
if (x == "") {
    alert("Name must be filled out");

    return false;

  }

  else{

    return true;

  }

}

      </script>

</body>

</html>
```

Home.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <!-- CSS only -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>



<link rel="stylesheet" href="/static/css/styles.css">



<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>




<style>

  table {

   font-family: arial, sans-serif;

   border-collapse: collapse;

   width: 60%;
```

```
<body>
 <!-- navbar  -->
 <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Expense Tracker</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/expenses">Expenses</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/budgets">Budgets</a>
        </li>
    <li class="nav-item">
     <a class="nav-link" href="/reports">Reports</a>
    </li>
    <li class="nav-item">
     <a class="nav-link" href="/account">Account</a>
    </li>

   </ul>
  </div>
 </div>
</nav>
<br><br>

  <!-- content  -->
  <div class="content">
  <h1>Dashboard</h1>
  <div class="d-grid gap-2 col-6 mx-auto">
           <a href="/expenses"><button class="btn btn-outline-dark" type="button">Add Expense</button></a>
  </div>
<!-- expenses  -->

<div class="dash-expenses">
  <h1>Your Expenses</h1>

  <div class="card text-bg-dark mb-3" style="max-width: 18rem;">
    <div class="card-header">Income</div>
    <div class="card-body">
     <h5 class="card-title">{{amount}}</h5>

    </div>
   </div>
   <div class="card text-bg-dark mb-3" style="max-width: 18rem;">
    <div class="card-header">Remaining Income</div>
    <div class="card-body">
     <h5 class="card-title " id="rem" >{{remaining}}</h5>

    </div>
   </div>
```

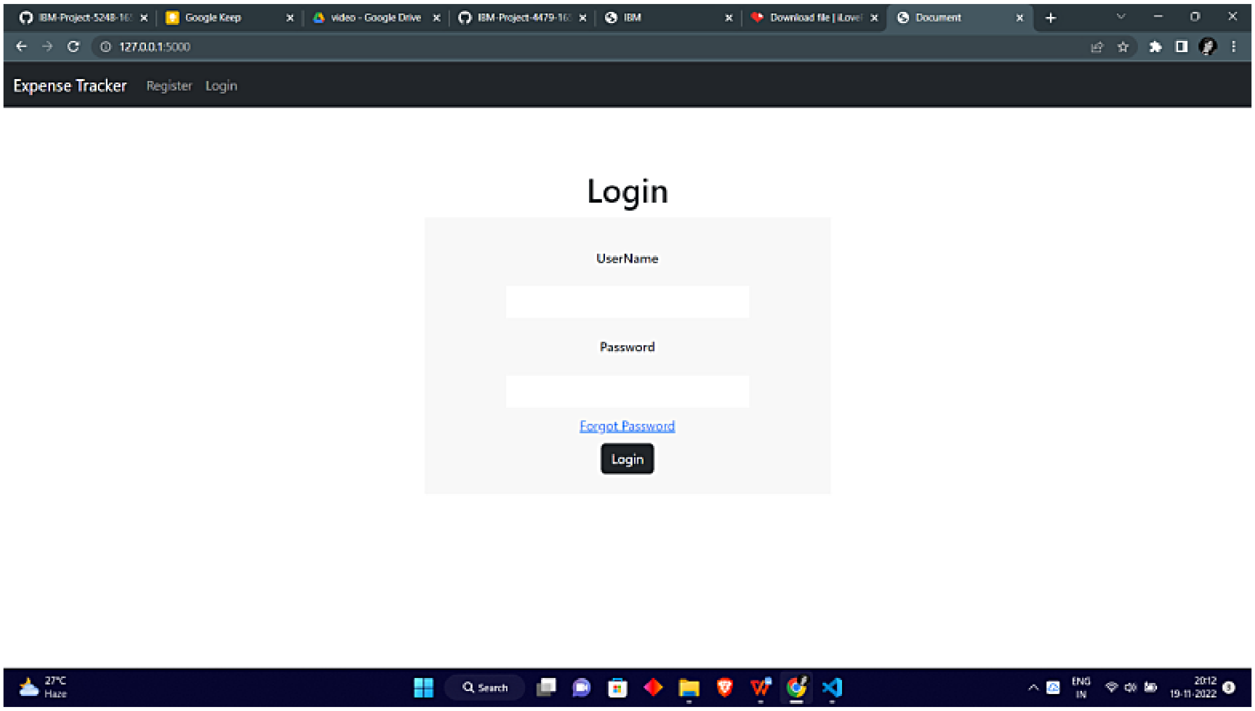
```
   <div class="card text-bg-dark mb-3" style="max-width: 18rem;">
    <div class="card-header">Monthly Expenses</div>
    <div class="card-body">
     <h5 class="card-title " id="exp">{{expense}}</h5>

    </div>
   </div>
  </div>
  <h1>Last 5 Expenses</h1>

{% if expenselist %}
<table>
 <tr>
  <th>Description</th>
  <th>Category</th>
  <th>Date</th>
  <th>Amount</th>
 </tr>
 {% for i in expenselist %}
```

```
  <tr>

    <td>{{i[2]}}</td>

    <td>{{i[3]}} </td>

    <td>{{i[4]}}</td>

    <td>{{i[5]}}</td>

  </tr>

  {% endfor %}

</table>


{% endif %}


    <h1>Your Budgets</h1>

    <p></p>


    <div style="height:500px ; width:500px ;margin: 10px auto;">

      <canvas id="myChart"></canvas>

    </div>

  </div>

    <script>
```

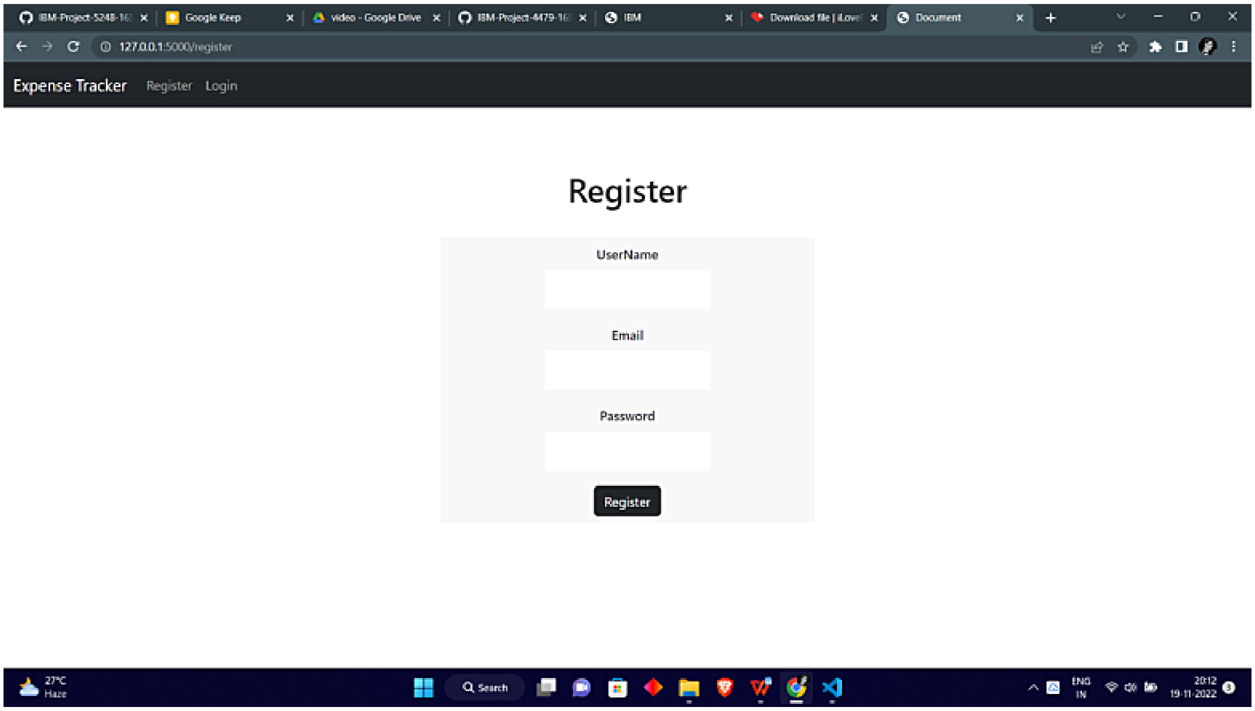
```
const labels = [
  'Remaining',
  'Spent',

];

var remaining = document.getElementById("rem").innerText;
var expense = document.getElementById("exp").innerText;

const data = {
  labels: labels,
  datasets: [{
    label: 'My First dataset',
    backgroundColor:['rgb(255, 99, 132)',
          'rgb(54, 162, 235)'],
    data: [remaining, expense],
  }]
};
const config = {
  type: 'pie',
    data: data,
    options: {}
   };
   const myChart = new Chart(
  document.getElementById('myChart'),
  config
 );
  </script>

</body>
</html>
```

Output:

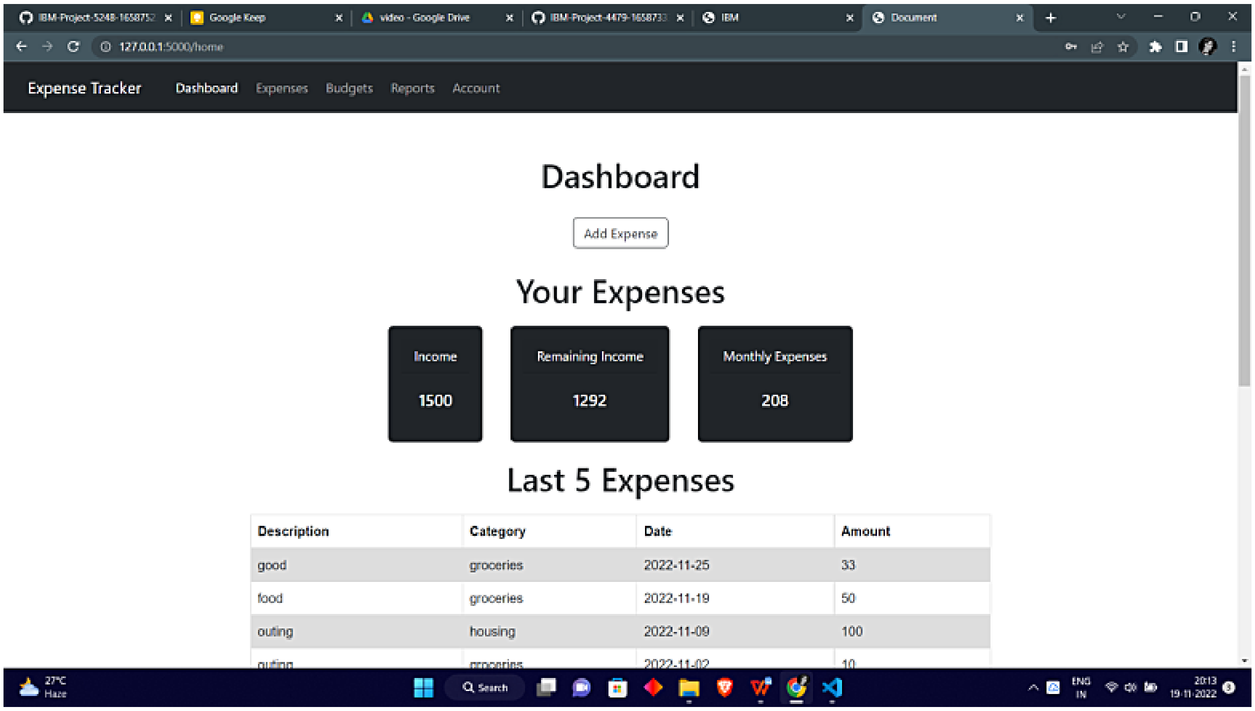
Login:


127.0.0.1:5000
Expense Tracker
Register
Login
Login
UserName
Password
Forgot Password
Login

Register:


127.0.0.1:5000/register
Expense Tracker
Register
Login
Register
UserName
Email
Password
Register

Home:



Dashboard

Your Expenses

Income	Remaining Income	Monthly Expenses
1500	1292	208

Last 5 Expenses

Description	Category	Date	Amount
good	groceries	2022-11-25	33
food	groceries	2022-11-19	50
outing	housing	2022-11-09	100
outing	groceries	2022-11-02	10

7.2 FEATURE 2

EXPENSE

For adding expense in our app first fill the details like date, expense name, expense amount then choose payment method etc. and then click the add button it will be redirect to a page where we can add our expenses.

REPORT

The user's added expenses are stored in the history and it will be display all the expenses

BUDGET

he user can enter the budget limit to avoid over expenses

expense.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="/static/css/styles.css">

<style>
  table {
   font-family: arial, sans-serif;
   border-collapse: collapse;
   width: 60%;
   margin: 20px auto;
 }

 td, th {
   border: 1px solid #dddddd;
   text-align: left;
   padding: 8px;
 }
 tr:nth-child(even) {
   background-color: #dddddd;
 }

 .table h1,h6,a{
   padding: 15px;
   margin: 3px;
 }
 form{
   width: 400px;
   height: auto;
   padding: 10px;
 }
 .btn{
   display: block;
   margin: 2px auto;
 }
</style>

    <title>Document</title>

</head>
<body>
    <!-- navbar  -->
```

```
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
   <div class="container-fluid">
    <a class="navbar-brand" href="#">Expense Tracker</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
     <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
     <ul class="navbar-nav">
      <li class="nav-item">
       <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
      </li>
      <li class="nav-item">
       <a class="nav-link" href="/expenses">Expenses</a>
      </li>
      <li class="nav-item">
       <a class="nav-link" href="/budgets">Budgets</a>
      </li>

      <li class="nav-item">
       <a class="nav-link" href="/reports">Reports</a>
      </li>
      <li class="nav-item">
       <a class="nav-link" href="/account">Account</a>
```

```
            </li>


          </ul>

        </div>

      </div>

    </nav>

    <br><br>


    <div class="content">

<!-- expense  -->


<h1>Add Expenses</h1>


<form action="/expenses" method="post">

<p>Description:</p>

</select>

<p>Date:</p>

<input type="date" name="dateofexpense" id="">

<p>Amount:</p>

<input type="number" name="amount" id="">

<button type="submit" class="btn btn-dark">Save</button>


</form>

<h1>Expense History</h1>
```

```
{% if error %}
<div class="alert alert-danger" role="alert">
  {{error}}
</div>
{% endif %}


{% if expenselist %}
<table>
  <tr>
    <th>Description</th>
    <th>Category</th>
    <th>Date</th>
    <th>Amount</th>
    <th>Delete expense</th>
  </tr>
  {% for i in expenselist %}

  <tr>
    <td>{{i[2]}}</td>
    <td>{{i[3]}} </td>
    <td>{{i[4]}}</td>
    <td>{{i[5]}}</td>
    <form action="/deleteexpense" method="post">
      <input type="number" style="display:none;" name="id" value={{i[0]}} id="">
```

```
   <input type="text" style="display:none;" name="uname" value={{i[1]}} id="">
  <td><button type="submit" class="btn btn-dark">Delete</button></td>
 </form>
 </tr>
 {% endfor %}
</table>

{% endif %}
</div>

</body>
</html>
```

Report.html

```
<body>
  <!-- navbar  -->
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
   <div class="container-fluid">
    <a class="navbar-brand" href="#">Expense Tracker</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
     <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
```

```
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/expenses">Expenses</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/budgets">Budgets</a>
            </li>

            <li class="nav-item">
              <a class="nav-link" href="/reports">Reports</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/account">Account</a>
            </li>

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

<br><br>
    <div class="content">
```

```
<!-- contents  -->


<h1>Reports</h1>
<div class="reportcard">
<div class="card text-center text-bg-dark mb-3" style="width: 18rem;">
    <div class="card-body">
      <h5 class="card-title">Budget Overview</h5>
      <a href="/budgetoverview" class="btn btn-outline-light">View Report</a>
    </div>
  </div>
  <div class="card text-center text-bg-dark mb-3" style="width: 18rem;">
    <div class="card-body">
      <h5 class="card-title">Monthly Spending</h5>

      <a href="/monthlyspending" class="btn btn-outline-light"> View Report</a>
    </div>
  </div>
</div>
<div class="reportcard">
  <div class="card text-center text-bg-dark mb-3" style="width: 18rem;">
    <div class="card-body">
      <h5 class="card-title">Spending Categories</h5>

      <a href="/spendingcategory" class="btn btn-outline-light">View Report </a>
    </div>
  </div>

  <div class="card text-center text-bg-dark mb-3" style="width: 18rem;">

    <div class="card-body">

      <h5 class="card-title">Expense Report </h5>


      <a href="/expensereport" class="btn btn-outline-light"> View Report</a>

    </div>

  </div>

</div>

</div>


</body>

</html>
```

Budget.html

```
<body>

    <!-- navbar  -->

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">

      <div class="container-fluid">

        <a class="navbar-brand" href="#">Expense Tracker</a>

        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

          <span class="navbar-toggler-icon"></span>
```

```
  </button>
  <div class="collapse navbar-collapse" id="navbarNav">
   <ul class="navbar-nav">
    <li class="nav-item">
     <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
    </li>
    <li class="nav-item">
     <a class="nav-link" href="/expenses">Expenses</a>
    </li>
    <li class="nav-item">
     <a class="nav-link" href="/budgets">Budgets</a>
    </li>

    <li class="nav-item">
     <a class="nav-link" href="/reports">Reports</a>
    </li>
    <li class="nav-item">
     <a class="nav-link" href="/account">Account</a>
    </li>

   </ul>
  </div>
 </div>
</nav>
<br><br>
```

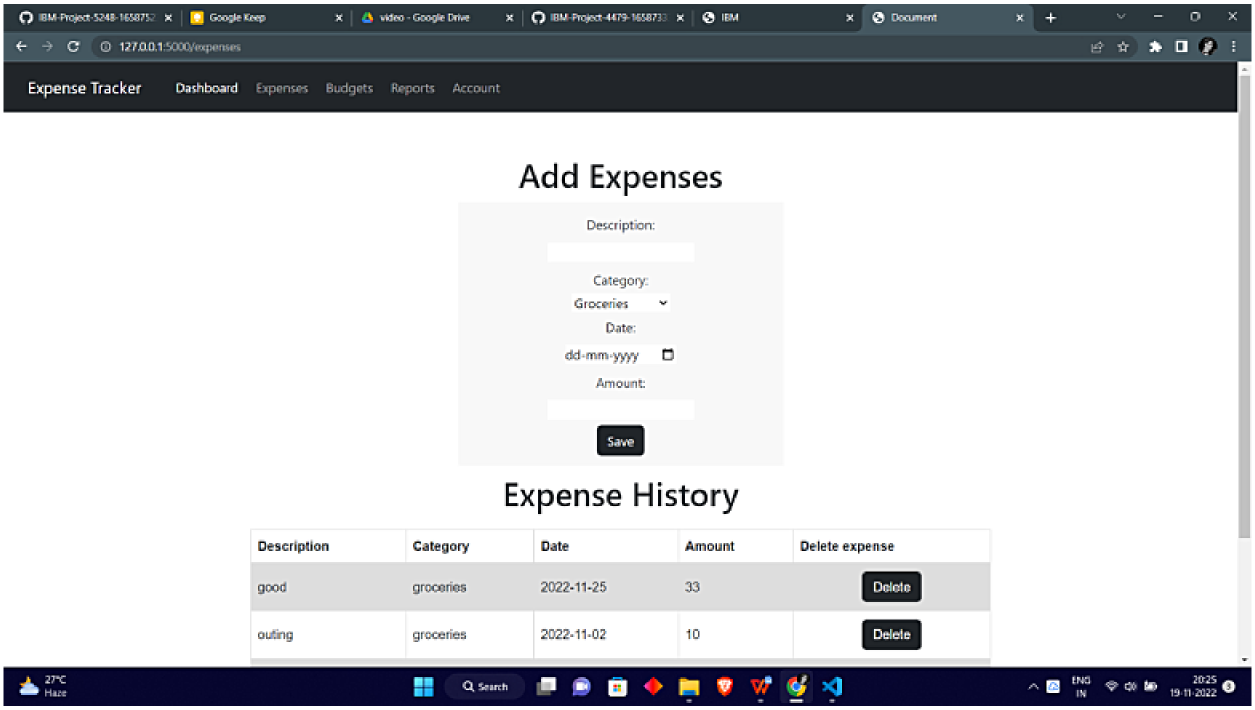
```
  <div class="content">
 <!-- content     -->

 {% if not budgetlist %}

 <h1>Create a Budget</h1>

  <h6>Amount (Monthly)</h6>
  <input type="number" name="amount" value="0" id="amount">
  <h6>Spending Categories</h6>
  <div class="formdata">

  <p>Groceries</p>
  <input type="number" name="Groceries" value="0" id="1">
  <p>Housing</p>
  <input type="number" name="Housing" value="0" id="2">
  <p>Utilities</p>
  <input type="number" name="Utilities" value="0" id="3">
</div>
<div class="formdata">
  <p>DiningOut</p>
  <input type="number" name="DiningOut" value="0" id="4">
  <p>Shopping</p>
  <input type="number" name="Shopping" value="0" id="5">
  <p>Travel</p>
  <input type="number" name="Travel" value="0" id="6">
</div>
  <div class="formdata">
  <p>Entertainment</p>
  <input type="number" name="Entertainment" value="0" id="7">
  <p>Others</p>
  <input type="number" name="Others" value="0" id="8">
  <p>Savings</p>
  <input type="number" name="Savings" value="0" id="9">
</div>
  <button type="submit" class="btn btn-dark">Save</button>

</form>
{% endif %}

{% if budgetlist %}
<div class="table">
  <h1>Budget Details</h1>
  <h6>Name:{{budgetlist[1]}} </h6>

  <h6>Month:{{budgetlist[2]}}</h6>
  <h6>Amount:{{budgetlist[3]}} </h6>
  <a href="/deletebudget"><button  class="btn btn-dark">Delete The Budget</button></a>

  <div class="table">
```

```
<table>
  <tr>
    <th>Category</th>
    <th>Amount</th>

  </tr>
  <tr>
    <td>Groceries </td>
    <td>{{budgetlist[4]}} </td>

  </tr>
  <tr>
    <td>Housing</td>
    <td>{{budgetlist[5]}} </td>
  </tr>
  <tr>
    <td>Utilities</td>
    <td>{{budgetlist[6]}} </td>
  </tr>
  <tr>
    <td>DiningOut</td>
    <td>{{budgetlist[7]}} </td>
  </tr>
  <tr>
    <td>Shopping</td>
    <td>{{budgetlist[8]}} </td>
  </tr>
  <tr>
    <td>Travel</td>
    <td>{{budgetlist[9]}} </td>
  </tr>
  <tr>
    <td>Entertainment</td>
    <td>{{budgetlist[10]}} </td>
  </tr>
  <tr>
    <td>Others</td>
    <td>{{budgetlist[11]}} </td>
  </tr>
  <tr>
    <td>Savings</td>
    <td>{{budgetlist[12]}} </td>
  </tr>
  <tr>
    <td>Total</td>
    <td>{{budgetlist[3]}} </td>
  </tr>


</table>
```

```
</div>
{% endif %}
</div>
</body>
</html>
```

OUTPUT:

Expense.html



Add Expenses

Description:

Category:
Groceries

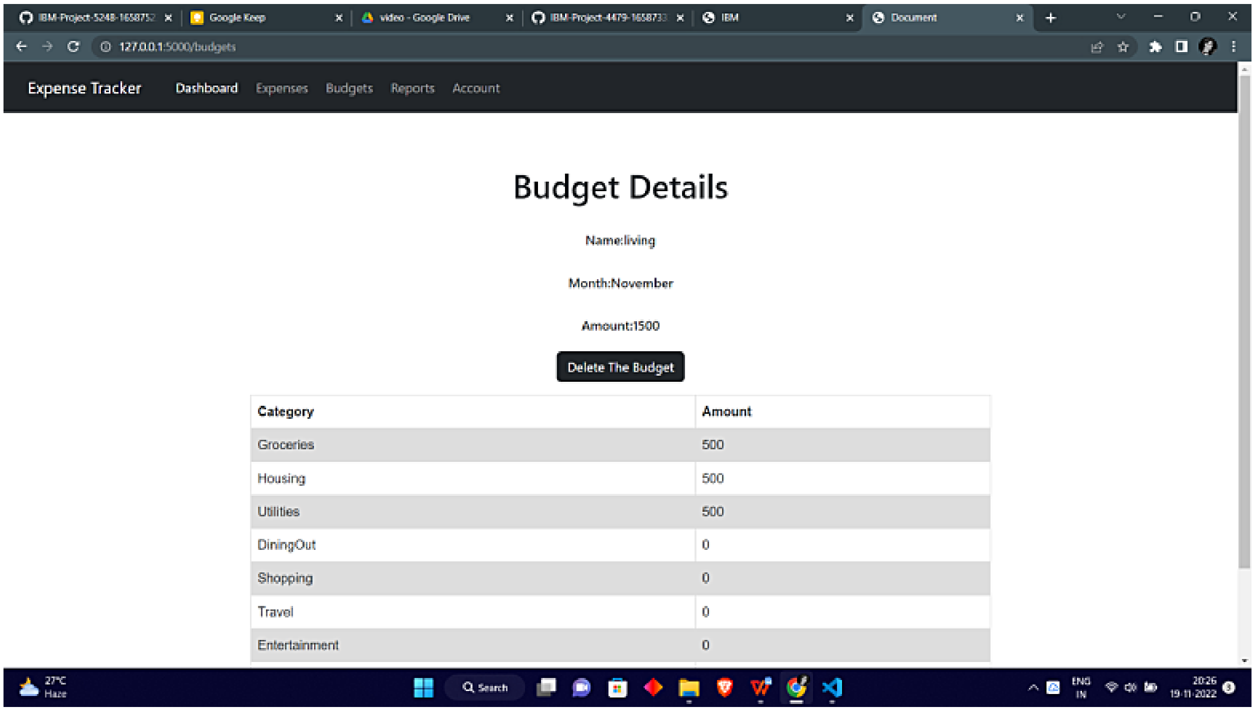
Date:
dd-mm-yyyy

Amount:

Expense History

Description	Category	Date	Amount	Delete expense
good	groceries	2022-11-25	33	Delete
outing	groceries	2022-11-02	10	Delete

Budget.html



Budget Details

Name:living

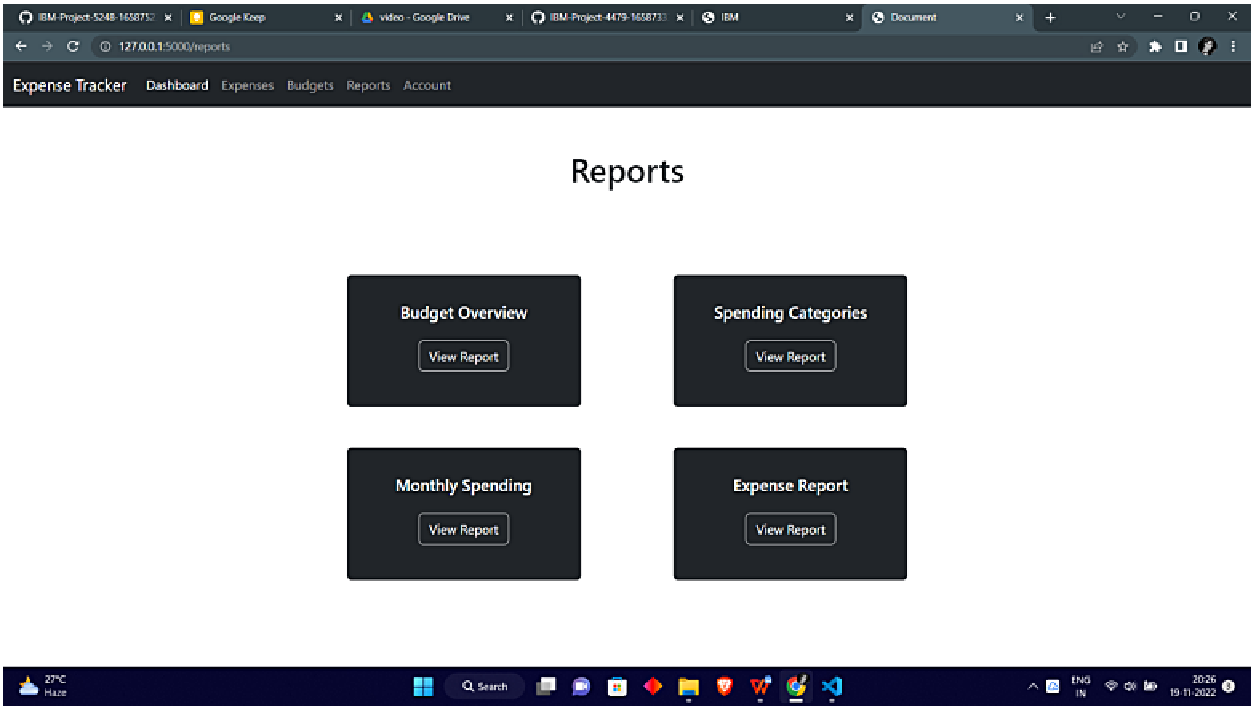
Month:November

Amount:1500

Delete The Budget

Category	Amount
Groceries	500
Housing	500
Utilities	500
DiningOut	0
Shopping	0
Travel	0
Entertainment	0

Reports.html



Reports

7.3 DATABASE SCHEMA (if Applicable)

User table:

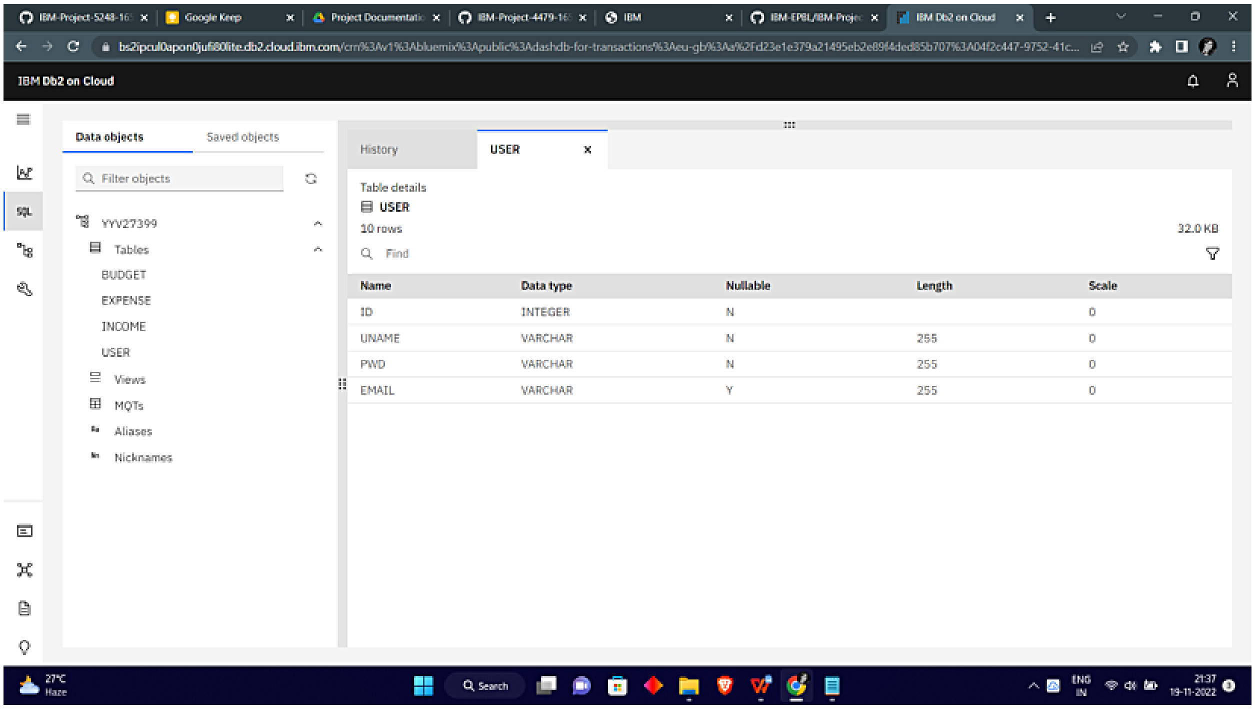


Table details

USER

10 rows

32.0 KB

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
UNAME	VARCHAR	N	255	0
PWD	VARCHAR	N	255	0
EMAIL	VARCHAR	Y	255	0

Income table

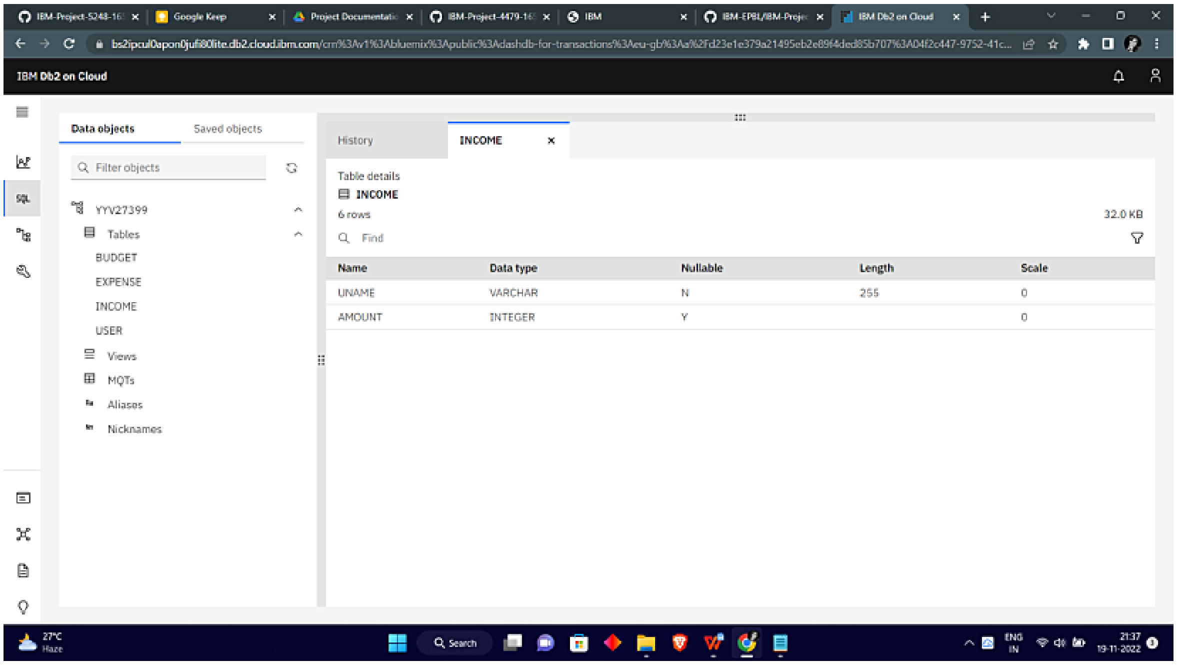


Table details

INCOME

6 rows

32.0 KB

Name	Data type	Nullable	Length	Scale
UNAME	VARCHAR	N	255	0
AMOUNT	INTEGER	Y		0

Expense table

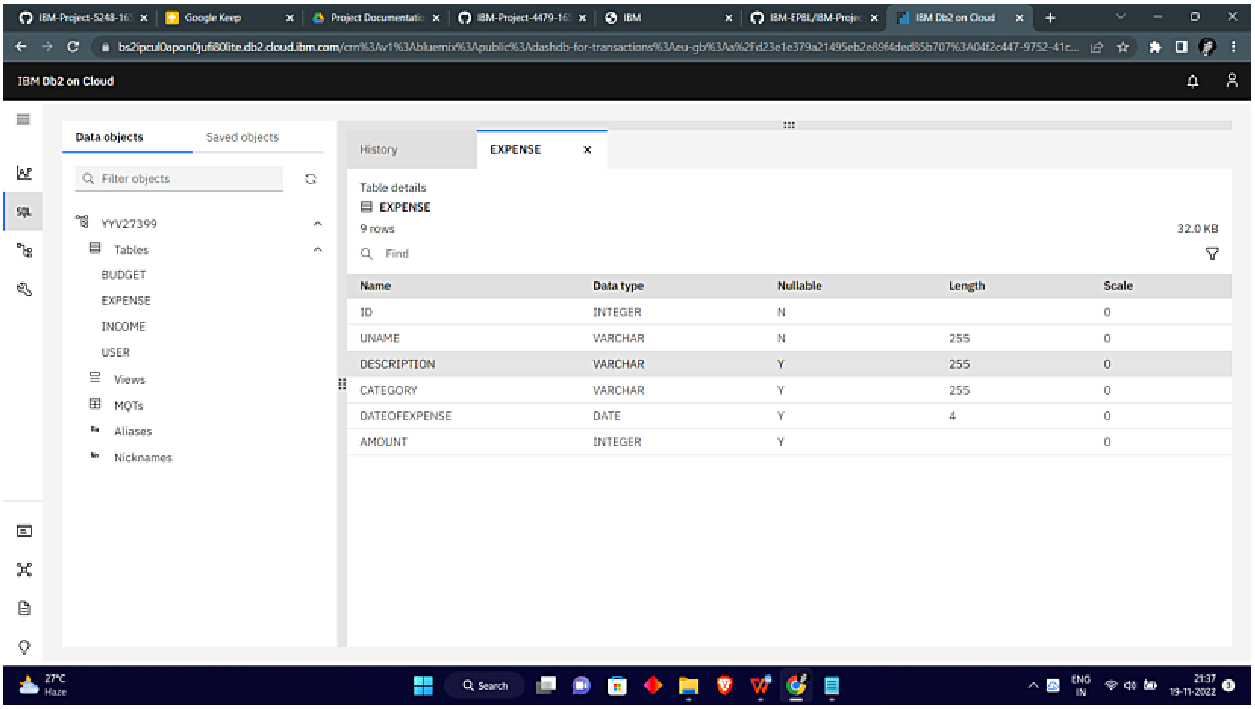


Table details
EXPENSE
9 rows
32.0 KB

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
UNAME	VARCHAR	N	255	0
DESCRIPTION	VARCHAR	Y	255	0
CATEGORY	VARCHAR	Y	255	0
DATEOFEXPENSE	DATE	Y	4	0
AMOUNT	INTEGER	Y		0

Budget table

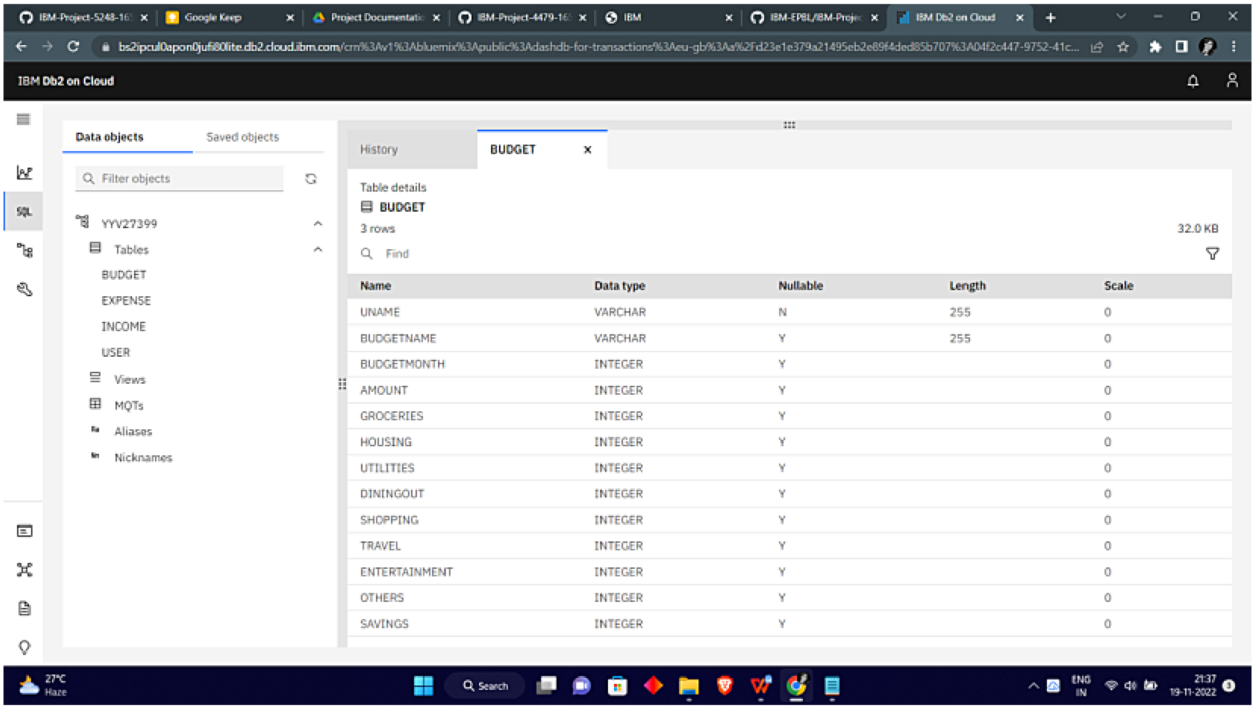
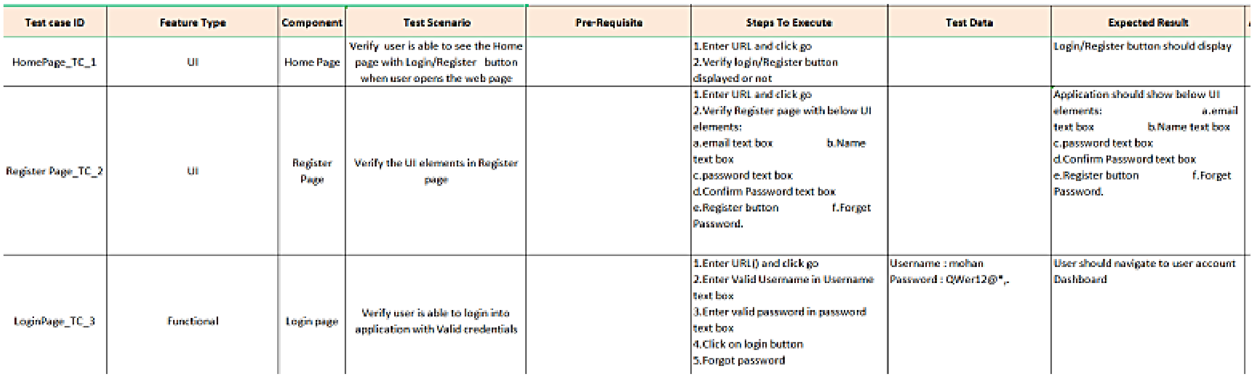


Table details
BUDGET
3 rows
32.0 KB

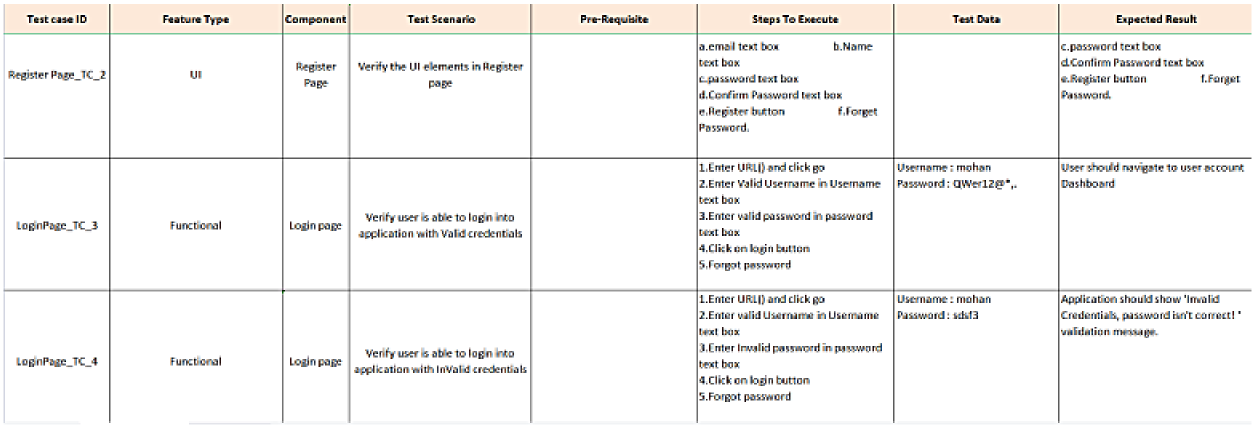
Name	Data type	Nullable	Length	Scale
UNAME	VARCHAR	N	255	0
BUDGETNAME	VARCHAR	Y	255	0
BUDGETMONTH	INTEGER	Y		0
AMOUNT	INTEGER	Y		0
GROCERIES	INTEGER	Y		0
HOUSING	INTEGER	Y		0
UTILITIES	INTEGER	Y		0
DININGOUT	INTEGER	Y		0
SHOPPING	INTEGER	Y		0
TRAVEL	INTEGER	Y		0
ENTERTAINMENT	INTEGER	Y		0
OTHERS	INTEGER	Y		0
SAVINGS	INTEGER	Y		0

8. TESTING

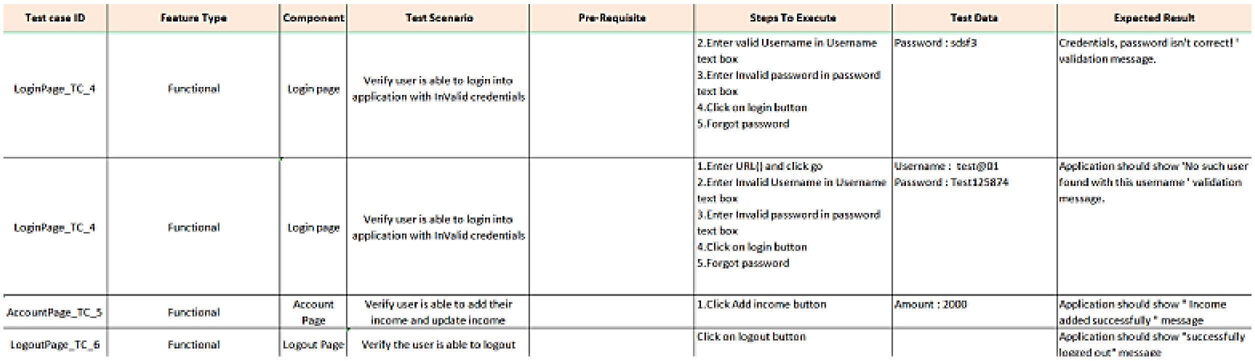
8.1 TEST CASES



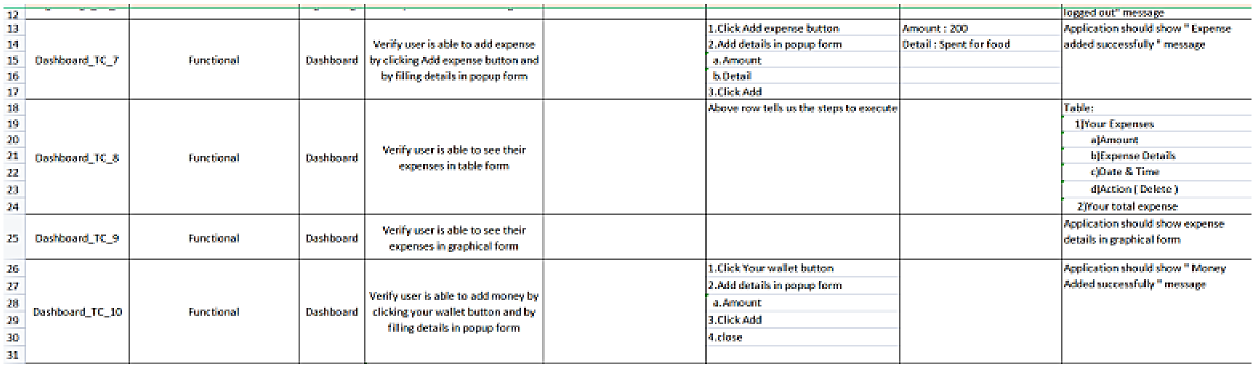
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result
HomePage_TC_1	UI	Home Page	Verify user is able to see the Home page with Login/Register button when user opens the web page		1.Enter URL and click go 2.Verify login/Register button displayed or not		Login/Register button should display
Register Page_TC_2	UI	Register Page	Verify the UI elements in Register page		1.Enter URL and click go 2.Verify Register page with below UI elements: a.email text box b.Name text box c.password text box d.Confirm Password text box e.Register button f.Forget Password.		Application should show below UI elements: a.email text box b.Name text box c.password text box d.Confirm Password text box e.Register button f.Forget Password.
LoginPage_TC_3	Functional	Login page	Verify user is able to login into application with Valid credentials		1.Enter URL() and click go 2.Enter Valid Username in Username text box 3.Enter valid password in password text box 4.Click on login button 5.Forgot password	Username : mohan Password : QWer12@*,.	User should navigate to user account Dashboard



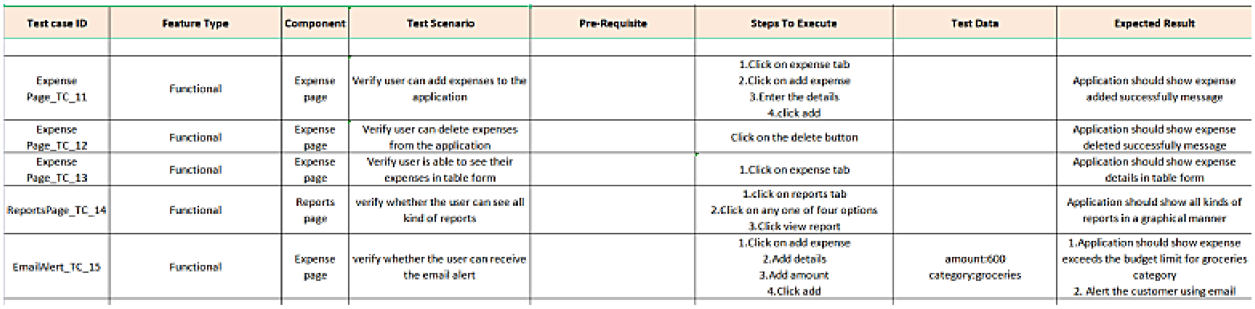
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result
Register Page_TC_2	UI	Register Page	Verify the UI elements in Register page		a.email text box b.Name text box c.password text box d.Confirm Password text box e.Register button f.Forget Password.		c.password text box d.Confirm Password text box e.Register button f.Forget Password.
LoginPage_TC_3	Functional	Login page	Verify user is able to login into application with Valid credentials		1.Enter URL() and click go 2.Enter Valid Username in Username text box 3.Enter valid password in password text box 4.Click on login button 5.Forgot password	Username : mohan Password : QWer12@*,.	User should navigate to user account Dashboard
LoginPage_TC_4	Functional	Login page	Verify user is able to login into application with InValid credentials		1.Enter URL() and click go 2.Enter valid Username in Username text box 3.Enter Invalid password in password text box 4.Click on login button 5.Forgot password	Username : mohan Password : sdsf3	Application should show 'Invalid Credentials, password isn't correct! ' validation message.



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result
LoginPage_TC_4	Functional	Login page	Verify user is able to login into application with InValid credentials		2.Enter valid Username in Username text box 3.Enter Invalid password in password text box 4.Click on login button 5.Forgot password	Password : sdsf3	Credentials, password isn't correct! ' validation message.
LoginPage_TC_4	Functional	Login page	Verify user is able to login into application with InValid credentials		1.Enter URL() and click go 2.Enter Invalid Username in Username text box 3.Enter Invalid password in password text box 4.Click on login button 5.Forgot password	Username : test@01 Password : Test125874	Application should show 'No such user found with this username ' validation message.
AccountPage_TC_5	Functional	Account Page	Verify user is able to add their income and update income		1.Click Add income button	Amount : 2000	Application should show " Income added successfully " message
LogoutPage_TC_6	Functional	Logout Page	Verify the user is able to logout		Click on logout button		Application should show "successfully logged out" message



12								logged out" message
13 14 15 16 17	Dashboard_TC_7	Functional	Dashboard	Verify user is able to add expense by clicking Add expense button and by filling details in popup form		1.Click Add expense button 2.Add details in popup form a.Amount b.Detail 3.Click Add	Amount : 200 Detail : Spent for food	Application should show " Expense added successfully " message
18 19 20 21 22 23 24	Dashboard_TC_8	Functional	Dashboard	Verify user is able to see their expenses in table form		Above row tells us the steps to execute		Table: 1)Your Expenses a)Amount b)Expense Details c)Date & Time d)Action (Delete) 2)Your total expense
25	Dashboard_TC_9	Functional	Dashboard	Verify user is able to see their expenses in graphical form				Application should show expense details in graphical form
26 27 28 29 30 31	Dashboard_TC_10	Functional	Dashboard	Verify user is able to add money by clicking your wallet button and by filling details in popup form		1.Click Your wallet button 2.Add details in popup form a.Amount 3.Click Add 4.close		Application should show " Money Added successfully " message



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result
Expense Page_TC_11	Functional	Expense page	Verify user can add expenses to the application		1.Click on expense tab 2.Click on add expense 3.Enter the details 4.click add		Application should show expense added successfully message
Expense Page_TC_12	Functional	Expense page	Verify user can delete expenses from the application		Click on the delete button		Application should show expense deleted successfully message
Expense Page_TC_13	Functional	Expense page	Verify user is able to see their expenses in table form		1.Click on expense tab		Application should show expense details in table form
ReportsPage_TC_14	Functional	Reports page	verify whether the user can see all kind of reports		1.click on reports tab 2.Click on any one of four options 3.Click view report		Application should show all kinds of reports in a graphical manner
EmailAlert_TC_15	Functional	Expense page	verify whether the user can receive the email alert		1.Click on add expense 2.Add details 3.Add amount 4.Click add	amount:600 category:groceries	1.Application should show expense exceeds the budget limit for groceries category 2. Alert the customer using email

8.2 USER ACCEPTANCE TESTING

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Personal Expense Tracker Application project at the time of the release to User Acceptance Testing (UAT).

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	Subtotal
By Design	10	4	2	3	20
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	2	1	8
Totals	22	14	11	22	51

Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Interface	7	0	0	7
Login	43	0	0	43
Logout	2	0	0	2
Limit	3	0	0	3
Edit	8	0	0	8
Adding expenses	9	0	0	9

History	4	0	0	4
Final report	2	0	0	2

9.RESULTS

9.1 PERFORMANCE METRICS

a. **Tracking income and expenses:** Monitoring the income and tracking all expenditures (through bank accounts, mobile wallets, and credit & debit cards).
b. **Transaction Receipts:** Capture and organize your payment receipts to keep track of your expenditure.
c. **Organizing Taxes:** Import your documents to the expense tracking app, and it will streamline your income and expenses under the appropriate tax categories.
d. **Payments & Invoices:** Accept and pay from credit cards, debit cards, net banking, mobile wallets, and bank transfers, and track the status of your invoices and bills in the mobile app itself. Also, the tracking app sends reminders for payments and automatically matches the payments with invoices.
e. **Reports:** The expense tracking app generates and sends reports to give a detailed insight about profits, losses, budgets, income, balance sheets, etc.,
f. **E-commerce integration:** Integrate your expense tracking app with your eCommerce store and track your sales through payments received via multiple payment methods.
g. **Vendors and Contractors:** Manage and track all the payments to the vendors and contractors added to the mobile app.
h. **Access control:** Increase your team productivity by providing access control to particular users through custom permissions.
i. **Track Projects:** Determine project profitability by tracking labor costs, payroll, expenses, etc., of your ongoing project.
j. **Inventory tracking:** An expense tracking app can do it all. Right from tracking products or the cost of goods, sending alert notifications when the product is running out of stock or the product is not selling, to purchase orders.

k. I**n-depth insights and analytics:** Provides in-built tools to generate reports with easy-tounderstand visuals and graphics to gain insights about the performance of your business.

l. **Recurrent Expenses:** Rely on your budgeting app to track, streamline, and automate all the recurrent expenses and remind you on a timely basis

m. **Budget Vs. Actual Spent:** This is one of the most common features in an expense tracking mobile app. The user gets a detailed insight into the real-time income and expenditure. Thus, you can plan your budget strategically to reduce unnecessary expenses.

10.ADVANTAGES & DISADVANTAGES

Advantages:

n. It Helps You Stick to Your Budget.
o. Tracking Your Expenses Can Reveal Spending Issues.
p. It Helps You Meet Your Financial Objectives.
q. Record Expenses With Pen and Paper.
r. Make It Easier With an App or Software.
s. Work Together as a Couple.

Disadvantages:

t. Determining the right process.
u. Feeling constrained.
v. Spending more than necessary.
w. Finding the time for it.
x. Making the right decisions.
y. Impacting how employees feel.
z. Overlooking important factors.
aa. Having top-level employees do all the planning.

11.CONCLUSION

The new system has solved the majority of the restrictions of the current system and operates in accordance with the design specifications provided. The project we created is more efficient than the other income and expense tracker.We have created a web application that keeps track of all of your daily transactions, records any money you have borrowed or lent, suggests the best options for investments, provides discounts on popular items, allows you to view exchange, and allows you to read the most recent authenticated financial news. Sticky notes, spreadsheets, and handling vast amounts of data were all eliminated in this paper, and it was a success. The new experience was also highly convenient and hassle-free. The user can now manage his costs more efficiently thanks to our application.

12.FUTURE SCOPE

We pledge to incorporate the following highlights into our cost-control application, showcasing patterns before including patterns. This is frequently a graphical representation of the total client costs for a given time frame, which could be monthly, quarterly, half-yearly, or even annually. assessing costs The other intriguing feature of our cost supervisor programme is the ability to estimate expenditures that clients might incur today or one month from now. In addition to acquiring client information, this can help the client manage his costs effectively and avoid running out of money. Due to the fact that every client action is recorded directly in a database server maintained, gathering knowledge of client expenses and pay from databases is a simple task.

13.APPENDIX

Source Code

BACKEND:

Sendmail.py

```
import os

from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail




def sendmail(receiver,msg):

    message = Mail(

        from_email=os.environ.get('EMAIL_SENDER'),

        to_emails=receiver,

        subject='Expense tracker',
        body=msg)
    try:
        sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
        sg.send(message)


    except Exception as e:
        print(e.message)
```

Deployment.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: personal-expense-tracker
spec:
  replicas: 1
  selector:
    matchLabels:
      app: flasknode
  template:
    metadata:
      labels:
        app: flasknode
    spec:
      containers:
      - name: flasknode
```

```
        image: mohanraj27/pet

        imagePullPolicy: Always

        ports:

        - containerPort: 5000

---

apiVersion: v1

kind: Service

metadata:

  name: pet

spec:

  selector:

    app: flask-app

  ports:

  - name: http

    protocol: TCP

    port: 3000

    targetPort: 5000

  type: LoadBalancer
```

Dockerfile

```
FROM python:3


WORKDIR /app

COPY . /app

RUN pip install -r requirements.txt
```

```
EXPOSE 5000


ENTRYPOINT [ "python3" ]

CMD [ "app.py" ]
```

App.py

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

import ibm_db

from dbconfig import get_db_credential

import random

import calendar

from datetime import datetime

from sendmail import sendmail

from dotenv import load_dotenv

load_dotenv()


app = Flask(__name__)


# finding current year

def curyear():

    today = datetime.now()


    year = today.strftime("%Y")

    return yea
```

```
# finding current month
def curmon():
    today = datetime.now()
    month3 = today.strftime("%m")
    return month3

def fullmonth():
    today = datetime.now()

    month1 = today.strftime("%B")
    return month1

# generating ids

randomlist=[]
def randomno():

    random_id=random.randrange(100,999)
    if random_id not in randomlist:
        randomlist.append(random_id)
        return random_id
    else:
        randomno()
# default values
nameOfUser=" "
```

```
# conn = ibm_db.connect(get_db_credential()," "," ")


# sql = "SELECT * FROM user"

# print(sql)

# stmt = ibm_db.exec_immediate(conn, sql)

# student = ibm_db.fetch_row(stmt)

# print ("The Name is : ",  student)


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

def login():

    print(get_db_credential)


    if request.method == "POST":

        try:

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

            userName = request.form['Uname']

            Password = request.form['pass']

            sql = "SELECT * FROM user WHERE uname =? and pwd=?"

            stmt = ibm_db.prepare(conn, sql)

            ibm_db.bind_param(stmt,1,userName)

            ibm_db.bind_param(stmt,2,Password)

            ibm_db.execute(stmt)

            details = ibm_db.fetch_assoc(stmt)

            global nameOfUser
            nameOfUser=userName
            if details:
                return redirect("/home")

            else:
                error="Invalid username or password"
                return render_template("login.html",error=error)
        except:
            print("error occured while login")
        finally:
            ibm_db.close(conn)

    else:
        return render_template("login.html")


@app.route("/register",methods=["POST","GET"])
def register():
    if request.method == "POST":
        userName = request.form['Uname']
        Password = request.form['pass']
        email = request.form['email']

        conn = ibm_db.connect(get_db_credential()," "," ")
```

```
sql = "SELECT * FROM user WHERE uname =? "
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,userName)
ibm_db.execute(stmt)
details = ibm_db.fetch_assoc(stmt)
if details:
    error = "Username already taken"
    return render_template("register.html",error=error)


try:
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "INSERT INTO user VALUES (?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(prep_stmt, 1, randomno())

    ibm_db.bind_param(prep_stmt, 2, userName)
    ibm_db.bind_param(prep_stmt, 3, Password)
    ibm_db.bind_param(prep_stmt, 4, email)


    ibm_db.execute(prep_stmt)
    global nameOfUser
    nameOfUser=userName
```

```
        except:
            print("error occured while registering")
        finally:
            ibm_db.close(conn)
        return redirect("/home")
    else:
        return render_template("register.html")
@app.route("/forgotpassword",methods=["POST","GET"])
def forgotpassword():
    if request.method == "POST":
        userName = request.form['Uname']
        Password = request.form['pass']
        try:
            conn = ibm_db.connect(get_db_credential()," "," ")

            sql = "UPDATE user SET pwd = ? WHERE uname=?"
            prep_stmt = ibm_db.prepare(conn, sql)
            ibm_db.bind_param(prep_stmt, 1, Password)
            ibm_db.bind_param(prep_stmt, 2, userName)
            ibm_db.execute(prep_stmt)

        except:
            print("error occured while updating password")
        finally:
            ibm_db.close(conn)
        success="Password changed successfully !"

        return render_template("forgotpassword.html",success=success)

    else:
        return render_template("forgotpassword.html")

@app.route("/home",methods=["POST","GET"])
def home():
    if request.method == "GET":

        # getting amount
        expenselist=[]
        lis2=[]
        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT * FROM income WHERE uname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.execute(stmt)
        details = ibm_db.fetch_assoc(stmt)
        if details:
            amount=ibm_db.result(stmt,1)
        else:
            amount=0
        # grtting expenses
```

```
lis1=[]
currentmonth=curmon()
currentyear=curyear()
dateofexpense=f'{currentyear}-{currentmonth}-00'
enddate=f'{currentyear}-{currentmonth}-32'

sum=0

conn = ibm_db.connect(get_db_credential()," "," ")
sql = "SELECT amount FROM expense WHERE uname= ? and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,nameOfUser)
ibm_db.bind_param(stmt,2,dateofexpense)
ibm_db.bind_param(stmt,3,enddate)
ibm_db.execute(stmt)

    # ibm_db.exec_immediate(conn,sql)

while ibm_db.fetch_row(stmt)!= False:
    lis1.append(ibm_db.result(stmt,0))
for i in lis1:
    sum=sum+int(i)
lis1=[]


remaining=int(amount)-int(sum)


if not remaining:
    remaining=0


conn = ibm_db.connect(get_db_credential()," "," ")
sql = "SELECT * FROM expense WHERE uname= ? ORDER BY dateofexpense DESC LIMIT 5;"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,nameOfUser)
ibm_db.execute(stmt)


while ibm_db.fetch_row(stmt)!= False:
    for i in range(6):
        lis2.append(ibm_db.result(stmt,i))
    expenselist.append(lis2)
    lis2=[]



    # my_data = [remaining, sum]
    # my_labels = 'Remaining ', 'Spent '
    # my_colors = ['lightblue', 'lightsteelblue']
    # my_explode = (0, 0.1)
```

```
            # plt.pie(my_data, labels=my_labels, autopct='%1.1f%%', startangle=15, shadow=True, colors=my_colors, explode=my_explode)

            # plt.title('Your budget ')

            # plt.axis('equal')

            # plt.show()



        return render_template("home.html",amount=amount,remaining=remaining,expense=sum,expenselist=expenselist)



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

def account():

    if request.method == "POST":


        amount = request.form['amount']


        try:

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


            sql = "UPDATE income SET amount = ? WHERE uname=?"

            prep_stmt = ibm_db.prepare(conn, sql)

            ibm_db.bind_param(prep_stmt, 1, amount)

            ibm_db.bind_param(prep_stmt, 2, nameOfUser)

            ibm_db.execute(prep_stmt)
            global incomeofuser
            incomeofuser=amount


        except:
            print("error occured while updating amount")
        finally:
            ibm_db.close(conn)


        return redirect("/account")


    else:
        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT * FROM income WHERE uname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.execute(stmt)
        details = ibm_db.fetch_assoc(stmt)
        if details:
            amount=ibm_db.result(stmt,1)



        else:
            conn = ibm_db.connect(get_db_credential()," "," ")
            sql = "insert into income values(?,?)"
            stmt = ibm_db.prepare(conn, sql)
            ibm_db.bind_param(stmt,1,nameOfUser)

            ibm_db.bind_param(stmt,2,0)

            ibm_db.execute(stmt)

            amount=0


        return render_template("account.html",userName="Welcome!
"+nameOfUser,amount=amount)





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

def budgets():

    if request.method == "POST":


        budgetname = request.form['budgetname']

        month = request.form['month']

        amt = request.form['amount']

        Groceries = request.form['Groceries']

        Housing = request.form['Housing']

        Utilities = request.form['Utilities']

        DiningOut = request.form['DiningOut']

        Shopping = request.form['Shopping']

        Travel = request.form['Travel']

        Entertainment = request.form['Entertainment']

        Others = request.form['Others']
        Savings = request.form['Savings']

        sum =
int(Groceries)+int(Housing)+int(Utilities)+int(DiningOut)+int(Shopping)+int(Travel)+int(Entert
ainment)+int(Others)+int(Savings)


# checking amount is equal or not

# getting amount to calulate total is equal or not

        error=False

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

        sql = "SELECT * FROM income WHERE uname =?"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt,1,nameOfUser)

        ibm_db.execute(stmt)

        details = ibm_db.fetch_assoc(stmt)

        if details:

            amount=ibm_db.result(stmt,1)

        else:

            error="First add the income in account"

    # inserting budget


        if not error:

            if int(amt)==int(amount):

                if int(amt)==sum:

                            try:

                                conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "INSERT INTO budget VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(prep_stmt, 1,nameOfUser )


    ibm_db.bind_param(prep_stmt, 2, budgetname)
    ibm_db.bind_param(prep_stmt, 3, month)
    ibm_db.bind_param(prep_stmt, 4, amt)
    ibm_db.bind_param(prep_stmt, 5,Groceries )
    ibm_db.bind_param(prep_stmt, 6,Housing )
    ibm_db.bind_param(prep_stmt, 7,Utilities )
    ibm_db.bind_param(prep_stmt, 8, DiningOut)
    ibm_db.bind_param(prep_stmt, 9, Shopping)
    ibm_db.bind_param(prep_stmt, 10, Travel)
    ibm_db.bind_param(prep_stmt, 11,Entertainment )
    ibm_db.bind_param(prep_stmt, 12,Others )
    ibm_db.bind_param(prep_stmt, 13,Savings )




    ibm_db.execute(prep_stmt)


except:
    print("error occured while registering")
finally:
    ibm_db.close(conn)
```

```
                        return redirect("/budgets")
            else:
                error="Total amount is not equal to budget amount"
        else:
            error="Budget amount is not equal to income"

    if error:
        return render_template("budgets.html",error=error)



else:
    try:
        budgetlist=[]

        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT * FROM budget WHERE uname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.execute(stmt)
        if ibm_db.fetch_row(stmt)!= False:
            for i in range(13):
                budgetlist.append(ibm_db.result(stmt,i))
            print(budgetlist)
            x = budgetlist[2]
            budgetlist[2]= calendar.month_name[x]
```

```
                # print(calendar.Calendar.month_name[11])
                return render_template("budgets.html",budgetlist=budgetlist)

            else:
                return render_template("budgets.html")



        except:
            print("error while displaying budget")



@app.route("/deletebudget")
def deletebudget():
    # getting amount and user name to delete budget
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT * FROM income WHERE uname =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.execute(stmt)
    details = ibm_db.fetch_assoc(stmt)
    if details:
        amount=ibm_db.result(stmt,1)
    # delete the budget
    try:
        conn = ibm_db.connect(get_db_credential()," "," ")

        sql = "DELETE FROM budget WHERE uname=? and amount=?"
        prep_stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(prep_stmt, 1, nameOfUser)
        ibm_db.bind_param(prep_stmt, 2, amount)
        ibm_db.execute(prep_stmt)

    except:
        print("error occured while updating amount")
    finally:
        ibm_db.close(conn)

    return redirect("/budgets")



@app.route("/expenses",methods=["POST","GET"])
def expenses():



    if request.method == "POST":
        categorylist=[]
        expenseamount=0
```

```
description = request.form['description']
category = request.form['category']
dateofexpense = request.form['dateofexpense']
amount = request.form['amount']
try:
    expenseamountlist=[]
    sum=0
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT amount FROM expense WHERE uname=? and category=? "
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.bind_param(stmt,2,category)
    ibm_db.execute(stmt)

    # ibm_db.exec_immediate(conn,sql)

    while ibm_db.fetch_row(stmt)!= False:
        expenseamountlist.append(ibm_db.result(stmt,0))
    for i in expenseamountlist:
        sum=int(sum)+int(i)

except:
    print("error while checking expenses")
# checking whether the expense is within the budget
try:
```

```
conn = ibm_db.connect(get_db_credential()," "," ")


sql = "SELECT * FROM budget WHERE uname =?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,nameOfUser)
# # ibm_db.bind_param(stmt,2,category)
ibm_db.execute(stmt)



if ibm_db.fetch_row(stmt)!= False:
    val1=ibm_db.result(stmt,0)
    val2=ibm_db.result(stmt,1)
    val3=ibm_db.result(stmt,2)
    val4=ibm_db.result(stmt,3)
    val5=ibm_db.result(stmt,4)
    val6=ibm_db.result(stmt,5)
    val7=ibm_db.result(stmt,6)
    val8=ibm_db.result(stmt,7)
    val9=ibm_db.result(stmt,8)
    val10=ibm_db.result(stmt,9)
    val11=ibm_db.result(stmt,10)
    val12 =  ibm_db.result(stmt,11)
    val13=ibm_db.result(stmt,12)
```

```
        categorylist.append(["uname",val1])
        categorylist.append(["budgetname",val2])
        categorylist.append(["budgetmonth",val3])
        categorylist.append(["amount",val4])
        categorylist.append(["groceries",val5])
        categorylist.append(["housing",val6])
        categorylist.append(["utilities",val7])
        categorylist.append(["diningout",val8])
        categorylist.append(["shopping",val9])
        categorylist.append(["travel",val10])
        categorylist.append(["entertainment",val11])
        categorylist.append(["others",val12])
        categorylist.append(["savings",val13])
        print(categorylist)
        for i in categorylist:
            if i[0]==category:
                amt=i[1]

    else:
        error="budget is not created"
        return render_template("expenses.html",error=error)



except:
    print("error while displaying budget")
```

```
# checking expense is within the budget or not
expenseamount=int(amt)-sum
if int(amount)<=int(amt) and int(amount)<=expenseamount:

            #  insert into db
    try:
        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "INSERT INTO expense VALUES (?,?,?,?,?,?)"
        prep_stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(prep_stmt, 1, randomno())

        ibm_db.bind_param(prep_stmt, 2, nameOfUser)
        ibm_db.bind_param(prep_stmt, 3, description)
        ibm_db.bind_param(prep_stmt, 4, category)
        ibm_db.bind_param(prep_stmt, 5, dateofexpense)
        ibm_db.bind_param(prep_stmt, 6, amount)

        ibm_db.execute(prep_stmt)
    except:

        print("error occured while inserting expense")
    finally:
        ibm_db.close(conn)

    return redirect("/expenses")
    else:

        # getting customer mail id
        conn = ibm_db.connect(get_db_credential()," "," ")

        sql = "SELECT * FROM user WHERE uname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        # # ibm_db.bind_param(stmt,2,category)
        ibm_db.execute(stmt)



        if ibm_db.fetch_row(stmt)!= False:
            receiver=ibm_db.result(stmt,3)



        error=f"expense exceeds the budget amount for {category} category"
        sendmail(receiver,error)
        return render_template("expenses.html",error=error)
else:
    # display all the expenses

    # try:
    expenselist=[]
    lis1=[]
```

```
sum=0
error="Currently no expenses"
conn = ibm_db.connect(get_db_credential()," "," ")
sql = "SELECT * FROM expense WHERE uname=?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,nameOfUser)
ibm_db.execute(stmt)

    # ibm_db.exec_immediate(conn,sql)

while ibm_db.fetch_row(stmt)!= False:
    for i in range(6):
        lis1.append(ibm_db.result(stmt,i))
    expenselist.append(lis1)
    lis1=[]
    error=False

if error:
    return render_template("expenses.html",error=error)



# except:
#     print("error while checking expenses")

return render_template("expenses.html",expenselist=expenselist)
```

```
@app.route("/deleteexpense",methods=["POST","GET"])
def deleteexpense():
    if request.method == "POST":
        id = request.form['id']
        uname = request.form['uname']
        try:
            conn = ibm_db.connect(get_db_credential()," "," ")

            sql = "DELETE FROM expense WHERE id=? and uname=?"
            prep_stmt = ibm_db.prepare(conn, sql)
            ibm_db.bind_param(prep_stmt, 1, id)
            ibm_db.bind_param(prep_stmt, 2, uname)
            ibm_db.execute(prep_stmt)

        except:
            print("error occured while deleting expense")

        finally:
            ibm_db.close(conn)
        return redirect("/expenses")

@app.route("/reports")
def reports():
    return render_template("reports.html")


@app.route("/monthlyspending",methods=["POST","GET"])
def monthlyspending():
    if request.method == "GET":

        # getting amount
        expenselist=[]
        lis2=[]
        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT * FROM income WHERE uname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.execute(stmt)
        details = ibm_db.fetch_assoc(stmt)
        if details:
            amount=ibm_db.result(stmt,1)
        else:
            amount=0
        # getting expenses
        expenselist2=[]
        lis3=[]
        lis1=[]
        currentmonth=curmon()
        currentyear=curyear()
        dateofexpense=f'{currentyear}-{currentmonth}-00'

        enddate=f'{currentyear}-{currentmonth}-32'

        sum=0


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

        sql = "SELECT amount FROM expense WHERE uname= ? and ( VARCHAR_FORMAT
(dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-
DD')< ?);"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt,1,nameOfUser)

        ibm_db.bind_param(stmt,2,dateofexpense)

        ibm_db.bind_param(stmt,3,enddate)

        ibm_db.execute(stmt)


          # ibm_db.exec_immediate(conn,sql)


        while ibm_db.fetch_row(stmt)!= False:

          lis1.append(ibm_db.result(stmt,0))

        for i in lis1:

          sum=sum+int(i)

        lis1=[]


        remaining=int(amount)-int(sum)


        if not remaining:
```

```
      remaining=0


    #  display expenses
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT * FROM expense WHERE uname= ? and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.bind_param(stmt,2,dateofexpense)
    ibm_db.bind_param(stmt,3,enddate)
    ibm_db.execute(stmt)


      # ibm_db.exec_immediate(conn,sql)


    while ibm_db.fetch_row(stmt)!= False:
      for i in range(6):
          lis3.append(ibm_db.result(stmt,i))
      expenselist2.append(lis3)
      lis3=[]


    currentmonth=fullmonth()


    return render_template("monthly-spending.html",remaining=remaining,expense=sum,expenselist2=expenselist2,currentmonth=currentmonth)
```

```
@app.route("/spendingcategory",methods=["POST","GET"])

def spendingcategory():

    if request.method == "GET":



        # getting groceries


        expenselist2=[]

        lis1=[]

        currentmonth=curmon()

        currentyear=curyear()

        dateofexpense=f'{currentyear}-{currentmonth}-00'

        enddate=f'{currentyear}-{currentmonth}-32'

        sum=0


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

        sql = "SELECT amount FROM expense WHERE ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt,1,nameOfUser)

        ibm_db.bind_param(stmt,2,'groceries')
        ibm_db.bind_param(stmt,3,dateofexpense)
        ibm_db.bind_param(stmt,4,enddate)
        ibm_db.execute(stmt)

            # ibm_db.exec_immediate(conn,sql)

        while ibm_db.fetch_row(stmt)!= False:
            lis1.append(ibm_db.result(stmt,0))
        for i in lis1:
            sum=sum+int(i)
        groc=sum
        lis1=[]

        # getting housing

        expenselist2=[]
        lis1=[]
        currentmonth=curmon()
        currentyear=curyear()
        dateofexpense=f'{currentyear}-{currentmonth}-00'
        enddate=f'{currentyear}-{currentmonth}-32'
        sum=0

        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? )and (
    VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT
    (dateofexpense,'YYYY-MM-DD')< ?);"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt,1,nameOfUser)

        ibm_db.bind_param(stmt,2,'housing')

        ibm_db.bind_param(stmt,3,dateofexpense)

        ibm_db.bind_param(stmt,4,enddate)

        ibm_db.execute(stmt)


            # ibm_db.exec_immediate(conn,sql)


        while ibm_db.fetch_row(stmt)!= False:

            lis1.append(ibm_db.result(stmt,0))

        for i in lis1:

            sum=sum+int(i)

        hous=sum

        lis1=[]


         # getting utilities


        expenselist2=[]

        lis1=[]

        currentmonth=curmon()

        currentyear=curyear()

        dateofexpense=f'{currentyear}-{currentmonth}-00'
        enddate=f'{currentyear}-{currentmonth}-32'
        sum=0


        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.bind_param(stmt,2,'utilities')
        ibm_db.bind_param(stmt,3,dateofexpense)
        ibm_db.bind_param(stmt,4,enddate)
        ibm_db.execute(stmt)


            # ibm_db.exec_immediate(conn,sql)


        while ibm_db.fetch_row(stmt)!= False:
            lis1.append(ibm_db.result(stmt,0))
        for i in lis1:
            sum=sum+int(i)
        util=sum
        lis1=[]


         # getting diningout
        expenselist2=[]
        lis1=[]

        sum=0

        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.bind_param(stmt,2,'diningout')
        ibm_db.bind_param(stmt,3,dateofexpense)
        ibm_db.bind_param(stmt,4,enddate)
        ibm_db.execute(stmt)

            # ibm_db.exec_immediate(conn,sql)

        while ibm_db.fetch_row(stmt)!= False:
            lis1.append(ibm_db.result(stmt,0))
        for i in lis1:
            sum=sum+int(i)
        din=sum
        lis1=[]
```

```
  # getting shopping

expenselist2=[]

lis1=[]

sum=0

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

sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( 
VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT 
(dateofexpense,'YYYY-MM-DD')< ?);"

stmt = ibm_db.prepare(conn, sql)

ibm_db.bind_param(stmt,1,nameOfUser)

ibm_db.bind_param(stmt,2,'shopping')

ibm_db.bind_param(stmt,3,dateofexpense)

ibm_db.bind_param(stmt,4,enddate)

ibm_db.execute(stmt)

   # ibm_db.exec_immediate(conn,sql)

while ibm_db.fetch_row(stmt)!= False:

   lis1.append(ibm_db.result(stmt,0))

for i in lis1:

   sum=sum+int(i)

shop=sum
```

```
        lis1=[]


          # getting travel


        expenselist2=[]

        lis1=[]


        sum=0


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

        sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"

        stmt = ibm_db.prepare(conn, sql)

        ibm_db.bind_param(stmt,1,nameOfUser)

        ibm_db.bind_param(stmt,2,'travel')

        ibm_db.bind_param(stmt,3,dateofexpense)

        ibm_db.bind_param(stmt,4,enddate)

        ibm_db.execute(stmt)


            # ibm_db.exec_immediate(conn,sql)


        while ibm_db.fetch_row(stmt)!= False:

            lis1.append(ibm_db.result(stmt,0))

        for i in lis1:
            sum=sum+int(i)
        trav=sum
        lis1=[]

          # getting entertainment

        expenselist2=[]
        lis1=[]

        sum=0

        conn = ibm_db.connect(get_db_credential()," "," ")
        sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.bind_param(stmt,2,'entertainment')
        ibm_db.bind_param(stmt,3,dateofexpense)
        ibm_db.bind_param(stmt,4,enddate)
        ibm_db.execute(stmt)

            # ibm_db.exec_immediate(conn,sql)

        while ibm_db.fetch_row(stmt)!= False:
```

```
        lis1.append(ibm_db.result(stmt,0))
    for i in lis1:
        sum=sum+int(i)
    ent=sum
    lis1=[]


      # getting others


    expenselist2=[]
    lis1=[]


    sum=0


    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT amount FROM expense WHERE ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.bind_param(stmt,2,'others')
    ibm_db.bind_param(stmt,3,dateofexpense)
    ibm_db.bind_param(stmt,4,enddate)
    ibm_db.execute(stmt)


        # ibm_db.exec_immediate(conn,sql)
```

```
while ibm_db.fetch_row(stmt)!= False:
    lis1.append(ibm_db.result(stmt,0))
for i in lis1:
    sum=sum+int(i)
other=sum
lis1=[]

  # getting savings

expenselist2=[]
lis1=[]

sum=0

conn = ibm_db.connect(get_db_credential()," "," ")
sql = "SELECT amount FROM expense WHERE  ( uname= ? and category= ? ) and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt,1,nameOfUser)
ibm_db.bind_param(stmt,2,'savings')
ibm_db.bind_param(stmt,3,dateofexpense)
ibm_db.bind_param(stmt,4,enddate)
ibm_db.execute(stmt)
```

```
        # ibm_db.exec_immediate(conn,sql)


    while ibm_db.fetch_row(stmt)!= False:

        lis1.append(ibm_db.result(stmt,0))

    for i in lis1:

        sum=sum+int(i)

    sav=sum

    lis1=[]




    total= int(groc)+int(hous)+int(util)+int(din)+int(shop)+int(trav)+int(ent)+int()+int(other)+int(sav)

    currentmonth=fullmonth()


    # return render_template("monthly-spending.html",remaining=remaining,expense=sum,expenselist2=expenselist2,currentmonth=currentmonth)

    if not total:

        error="Currently no expenses"

        return render_template("spending-categories.html",error=error)

    return render_template("spending-categories.html",currentmonth=currentmonth,total=total,groc=groc,hous=hous,util=util,din=din,shop=shop,trav=trav,ent=ent,other=other,sav=sav)


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

def budgetoverview():
```

```
if request.method == "GET":

    # getting amount
    expenselist=[]
    lis2=[]
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT * FROM income WHERE uname =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.execute(stmt)
    details = ibm_db.fetch_assoc(stmt)
    if details:
        amount=ibm_db.result(stmt,1)
    else:
        amount=0
    # getting expenses
    expenselist2=[]
    lis3=[]
    lis1=[]
    currentmonth=curmon()
    currentyear=curyear()
    dateofexpense=f'{currentyear}-{currentmonth}-00'
    enddate=f'{currentyear}-{currentmonth}-32'
    sum=0
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT amount FROM expense WHERE uname= ? and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.bind_param(stmt,2,dateofexpense)
    ibm_db.bind_param(stmt,3,enddate)
    ibm_db.execute(stmt)

      # ibm_db.exec_immediate(conn,sql)

    while ibm_db.fetch_row(stmt)!= False:
      lis1.append(ibm_db.result(stmt,0))
    for i in lis1:
      sum=sum+int(i)
    lis1=[]

    remaining=int(amount)-int(sum)

    if not remaining:
      remaining=0

    #  display expenses
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT * FROM expense WHERE uname= ? and ( VARCHAR_FORMAT
(dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,nameOfUser)
        ibm_db.bind_param(stmt,2,dateofexpense)
        ibm_db.bind_param(stmt,3,enddate)
        ibm_db.execute(stmt)

            # ibm_db.exec_immediate(conn,sql)

        while ibm_db.fetch_row(stmt)!= False:
            for i in range(6):
                lis3.append(ibm_db.result(stmt,i))
            expenselist2.append(lis3)
            lis3=[]

        currentmonth=fullmonth()

        return render_template("budget-overview.html",amount=amount,remaining=remaining,expense=sum,expenselist2=expenselist2,currentmonth=currentmonth)



@app.route("/expensereport",methods=["POST","GET"])
def expensereport():
    if request.method == "POST":
```

```
month = request.form['month']

year = request.form['year']

error="Currently no expenses"

startdate=f'{year}-{month}-00'

edate=f'{year}-{month}-31'

expenselist=[]

lis1=[]

 #  display expenses

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

sql = "SELECT * FROM expense WHERE uname= ? and ( VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD') > ? and VARCHAR_FORMAT (dateofexpense,'YYYY-MM-DD')< ?);"

stmt = ibm_db.prepare(conn, sql)

ibm_db.bind_param(stmt,1,nameOfUser)

ibm_db.bind_param(stmt,2,startdate)

ibm_db.bind_param(stmt,3,edate)

ibm_db.execute(stmt)


    # ibm_db.exec_immediate(conn,sql)


while ibm_db.fetch_row(stmt)!= False:

    for i in range(6):

        lis1.append(ibm_db.result(stmt,i))

    expenselist.append(lis1)

    lis1=[]
```

```
        error=False
    if error:
        return render_template("expense-report.html",error=error)


    return render_template("expense-report.html",expenselist=expenselist)
if request.method == "GET":
    expenselist=[]
    lis1=[]
    error="Currently no expenses"
    conn = ibm_db.connect(get_db_credential()," "," ")
    sql = "SELECT * FROM expense WHERE uname=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,nameOfUser)
    ibm_db.execute(stmt)


        # ibm_db.exec_immediate(conn,sql)


    while ibm_db.fetch_row(stmt)!= False:
        for i in range(6):
            lis1.append(ibm_db.result(stmt,i))
        expenselist.append(lis1)
        lis1=[]
        error=False
    if error:
        return render_template("expense-report.html",error=error)
        return render_template("expense-report.html",expenselist=expenselist)
if __name__ == "__main__":

    app.run(host='0.0.0.0',port=5000,debug=True)
```

FRONTEND:

Account.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3" crossorigin="anonymous"></script>

<link rel="stylesheet" href="/static/css/styles.css">
```

```
<style>
   .btn{
   display: block;
   margin: 15px auto;
 }
 .account{
  margin-top: 100px;
 }
 p,h5,input{
  margin: 15px auto;
 }

</style>



    <title>Document</title>


</head>
<body onload="normal()">
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
        <div class="container-fluid">
          <a class="navbar-brand" href="#">Expense Tracker</a>
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
```

```
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/expenses">Expenses</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/budgets">Budgets</a>
            </li>

            <li class="nav-item">
              <a class="nav-link" href="/reports">Reports</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/account">Account</a>
            </li>

          </ul>
        </div>
      </div>
    </nav>
<div class="content account">

    {% if userName %}
    <h1>{{userName}}</h1>
    {% endif  %}


    <div class="card text-center text-bg-dark mb-3" style="width: 18rem;">
      <div class="card-body">
        <h5 class="card-title">Income</h5>
        {% if amount %}
        <p class="card-text">Your Income is {{amount}}</p>
        {% endif  %}


        <div id="update">
          <form action="/account" method="post" style="background-color: RGBA(33,37,41,var(--bs-bg-opacity,1));">
        <input type="number" name="amount" id="">
        <button type="submit" class="btn btn-outline-light">Update Income </button>
        </form>
        <!-- onclick="updatefun()" -->
        </div>
      </div>
    </div>
  </div>
  <br>

 <a style="text-decoration:none;" href="/"><button  style="width:286px ;" class="btn btn-dark">Log Out  </button></a>
  <!-- <script>
      function normal(){
          document.getElementById("update").style.display = "none";
      }
      function updatefun(){
          document.getElementById("update").style.display = "block";
      }
  </script> -->

</body>
</html>
```

Budget overview.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">


    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3" crossorigin="anonymous"></script>


<link rel="stylesheet" href="/static/css/styles.css">
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>


<style>
  table {
    font-family: arial, sans-serif;
    border-collapse: collapse;
    width: 60%;
    margin: 20px auto;
  }


  td, th {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 8px;
  }


  tr:nth-child(even) {
    background-color: #dddddd;
  }
  </style>


    <title>Document</title>


</head>
<body>
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">Expense Tracker</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/expenses">Expenses</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/budgets">Budgets</a>
            </li>

            <li class="nav-item">
              <a class="nav-link" href="/reports">Reports</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/account">Account</a>
            </li>

          </ul>
        </div>
      </div>
    </nav>
    <br>
    <br>
    <div class="content">
    <div class="dash-expenses">
      <h1>Budget Overview </h1>
```

```
<h6>Month: {{currentmonth}}</h6>


<br>
<h1>Your Expenses</h1>


<div class="card text-bg-dark mb-3" style="max-width: 18rem;">
  <div class="card-header">Income</div>
  <div class="card-body">
    <h5 class="card-title">{{amount}}</h5>


  </div>
 </div>
 <div class="card text-bg-dark mb-3" style="max-width: 18rem;">
  <div class="card-header">Remaining Income</div>
  <div class="card-body">
    <h5 class="card-title " id="rem" >{{remaining}}</h5>


  </div>
 </div>


 <div class="card text-bg-dark mb-3" style="max-width: 18rem;">
  <div class="card-header">Monthly Expenses</div>
  <div class="card-body">
    <h5 class="card-title " id="exp">{{expense}}</h5>
```

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



     <br>

     <h1>Your Budgets</h1>



     <div class="mychart" style="height:500px ; width:500px ; margin: 10px auto;">
       <canvas id="myChart"></canvas>
     </div>



{% if expenselist2 %}
<table style="margin: 10px auto;">
  <tr>
    <th>Description</th>
    <th>Category</th>
    <th>Date</th>
    <th>Amount</th>
  </tr>
  {% for i in expenselist2 %}
  <tr>
    <td>{{i[2]}}</td>
    <td>{{i[3]}} </td>
    <td>{{i[4]}}</td>
    <td>{{i[5]}}</td>
  </tr>
  {% endfor %}
</table>

{% endif %}

</div>

     <script>
       const labels = [
         'Remaining',
         'Spent',

       ];

       var remaining = document.getElementById("rem").innerText;
       var expense = document.getElementById("exp").innerText;

       const data = {
         labels: labels,
         datasets: [{
           label: 'Budget Overview',
           backgroundColor:['rgb(255, 99, 132)',
                 'rgb(54, 162, 235)'],

           data: [remaining, expense],
         }]
       };

       const config = {
         type: 'pie',
         data: data,
         options: {}
       };

       const myChart = new Chart(
     document.getElementById('myChart'),
     config
   );
     </script>

</body>
</html>
```

Spending-category.html

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3" crossorigin="anonymous"></script>

<link rel="stylesheet" href="/static/css/styles.css">
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<style>
  table {
    font-family: arial, sans-serif;
    border-collapse: collapse;
    width: 60%;
    margin: 20px auto;
  }
  td, th {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 8px;
  }

  tr:nth-child(even) {
    background-color: #dddddd;
  }
  </style>


    <title>Document</title>


</head>
<body>
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">Expense Tracker</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
```

```html
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/expenses">Expenses</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/budgets">Budgets</a>
        </li>

        <li class="nav-item">
          <a class="nav-link" href="/reports">Reports</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/account">Account</a>
        </li>

      </ul>
    </div>
  </div>
</nav>
```

```
<br><br>
    <div class="content">



  <h1>Spending Category Overview</h1>



  <h6>Month: {{currentmonth}}</h6>
  <br>

  {% if  error %}
  <div class="alert alert-danger" role="alert">
    {{error}}
  </div>
{% endif  %}

<br>
  <!--
total=total,groc=groc,hous=hous,util=util,din=din,shop=shop,trav=trav,ent=ent,other=other,sav=
sav)  -->
  {% if not error %}

  <div style="height:500px ; width:800px; margin: 20px auto;">
  <canvas id="myChart"></canvas>
</div>


<div class="table">


<table>
  <tr>
    <th>Category</th>
    <th>Amount</th>

  </tr>
  <tr>
    <td>Groceries </td>
    <td id="1">{{groc}} </td>

  </tr>
  <tr>
    <td>Housing</td>
    <td id="2">{{hous}} </td>
  </tr>
  <tr>
    <td>Utilities</td>
    <td id="3">{{util}} </td>
```

```
</tr>
<tr>
  <td>DiningOut</td>
  <td id="4">{{din}} </td>
</tr>
<tr>
  <td>Shopping</td>
  <td id="5">{{shop}} </td>
</tr>
<tr>
  <td>Travel</td>
  <td id="6">{{trav}} </td>
</tr>
<tr>
  <td>Entertainment</td>
  <td id="7">{{ent}} </td>
</tr>
<tr>
  <td>Others</td>
  <td id="8">{{other}} </td>
</tr>
<tr>
  <td>Savings</td>
  <td id="9">{{sav}} </td>
</tr>
```

```
    <tr>
      <td>Total</td>
      <td id="10">{{total}} </td>
    </tr>


  </table>
</div>
{% endif %}

    </div>

    <script>
      const labels = [
        'Groceries',
        'Housing',
        'Utilities',
        'Dinner',
        'Shopping',
        'Travel',
        'Entertainment',
        'Others',
        'Savings'
```

```
 ];


 var groc = document.getElementById("1").innerText;
 var hous = document.getElementById("2").innerText;
 var util = document.getElementById("3").innerText;
 var din = document.getElementById("4").innerText;
 var shop = document.getElementById("5").innerText;
 var trav = document.getElementById("6").innerText;
 var ent = document.getElementById("7").innerText;
 var other = document.getElementById("8").innerText;
 var sav = document.getElementById("9").innerText;
 var total = document.getElementById("10").innerText;



const data = {
labels: labels,
datasets: [{
 label: 'Spending Category Overview',
 barPercentage: 0.5,
   barThickness: 50,
   maxBarThickness: 50,
   minBarLength: 40,
 data: [groc,hous,util,din,shop,trav,ent,other,sav],
 backgroundColor: [
 'rgba(255, 99, 132, 0.2)',
      'rgba(255, 159, 64, 0.2)',
      'rgba(255, 205, 86, 0.2)',
      'rgba(75, 192, 192, 0.2)',
      'rgba(54, 162, 235, 0.2)',
      'rgba(153, 102, 255, 0.2)',
      'rgba(201, 203, 207, 0.2)',
      'rgba(153, 102, 255, 0.2)',
      'rgba(201, 203, 207, 0.2)'
  }]
};
      const config = {
  type: 'bar',
  data: data,
  options: {
    scales: {
      y: {
        beginAtZero: true
      }
    }
  },
};

      const myChart = new Chart(
    document.getElementById('myChart'),
    config
  );
    </script>

</body>
</html>
```

Forgot password.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3" crossorigin="anonymous"></script>

<link rel="stylesheet" href="/static/css/styles.css">
```

```
<style>
  .btn{
    display: block;
    margin: 2px auto;
  }
</style>

    <title>Document</title>

</head>
<body>
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
        <div class="container-fluid">
          <a class="navbar-brand" href="#">Expense Tracker</a>
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav">

              <li class="nav-item">
                <a class="nav-link" href="/register">Register</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/">Login</a>
              </li>




            </ul>
          </div>
        </div>
      </nav>
      <br><br>
      <div class="content">
      <h1>Forgot Password</h1>

      <div class="register-form container">
      <form name="myForm" action="/forgotpassword" style="width: 500px;height:auto;
padding: 20px;" onsubmit="return validateForm()" method="post">
        <h6>UserName</h6>
        <input type="text"  id="usrname" name="Uname" id="">
        <h6>Password</h6>
        <!-- <input type="password" name="pass" id=""> -->
        <input type="password" id="psw" name="pass" pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-
Z]).{8,}" title="Must contain at least one number and one uppercase and lowercase letter, and at
least 8 or more characters" required>
```

```
            <button type="submit" class="btn btn-dark">Confirm</button>


        </form>
      </div>


      <div id="message">
        <h3>Password must contain the following:</h3>
        <p id="letter" class="invalid">A <b>lowercase</b> letter</p>
        <p id="capital" class="invalid">A <b>capital (uppercase)</b> letter</p>
        <p id="number" class="invalid">A <b>number</b></p>
        <p id="length" class="invalid">Minimum <b>8 characters</b></p>
      </div>


      {% if success %}
      <div class="alert alert-success" role="alert">
        {{success}}
      </div>


      {% endif %}
    </div>


      <script>
        var uname = document.getElementById("usrname");
        var myInput = document.getElementById("psw");
```

```
var letter = document.getElementById("letter");
var capital = document.getElementById("capital");
var number = document.getElementById("number");
var length = document.getElementById("length");

// When the user clicks on the password field, show the message box
myInput.onfocus = function() {
  document.getElementById("message").style.display = "block";
}

// When the user clicks outside of the password field, hide the message box
myInput.onblur = function() {
  document.getElementById("message").style.display = "none";
}

// When the user starts to type something inside the password field
myInput.onkeyup = function() {


  // Validate lowercase letters
  var lowerCaseLetters = /[a-z]/g;
  if(myInput.value.match(lowerCaseLetters)) {
    letter.classList.remove("invalid");
    letter.classList.add("valid");
  } else {
```

```
  letter.classList.remove("valid");

  letter.classList.add("invalid");

}


// Validate capital letters

var upperCaseLetters = /[A-Z]/g;

if(myInput.value.match(upperCaseLetters)) {

  capital.classList.remove("invalid");

  capital.classList.add("valid");

} else {

  capital.classList.remove("valid");

  capital.classList.add("invalid");

}


// Validate numbers

var numbers = /[0-9]/g;

if(myInput.value.match(numbers)) {

  number.classList.remove("invalid");

  number.classList.add("valid");

} else {

  number.classList.remove("valid");

  number.classList.add("invalid");

}


// Validate length
```

```
        if(myInput.value.length >= 8) {

          length.classList.remove("invalid");

          length.classList.add("valid");

        } else {

          length.classList.remove("valid");

          length.classList.add("invalid");

        }


      }


      function validateForm() {

  let x = document.forms["myForm"]["Uname"].value;

  if (x == "") {

    alert("Name must be filled out");

    return false;

  }

  else{

    return true;

  }

}

      </script>


</body>

</html>
```

Monthly spending.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>

<link rel="stylesheet" href="/static/css/styles.css">
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>



<style>
  table {
   font-family: arial, sans-serif;
   border-collapse: collapse;
  width: 60%;
  margin: 30px auto;
}

td, th {
  border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
}

tr:nth-child(even) {
  background-color: #dddddd;
}

.table h1,h6,a{
  padding: 15px;
  margin: 3px;
}
form{
  width: 400px;
  height: auto;
  padding: 10px;
}
.btn{
  display: block;
```

```
   margin: 2px auto;
 }
</style>



    <title>Document</title>


</head>
<body>
    <!-- navbar  -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark navbar-custom">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">Expense Tracker</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="/home">Dashboard</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="/expenses">Expenses</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/budgets">Budgets</a>
        </li>

        <li class="nav-item">
          <a class="nav-link" href="/reports">Reports</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="/account">Account</a>
        </li>

      </ul>
    </div>
  </div>
</nav>
<br><br>
<div class="content">
<div class="card text-bg-dark mb-3" style="max-width: 18rem; display: none;">
  <div class="card-header">Remaining Income</div>
  <div class="card-body">
    <h5 class="card-title " id="rem" >{{remaining}}</h5>

  </div>
</div>
```

```
    <div class="card text-bg-dark mb-3" style="max-width: 18rem; display: none;">
      <div class="card-header">Monthly Expenses</div>
      <div class="card-body">
        <h5 class="card-title " id="exp">{{expense}}</h5>

      </div>
    </div>

    <h1>Monthly Spending Overview  </h1>

      <h6>Month: {{currentmonth}}</h6>



     <div style="height:500px ; width:500px ;margin: 15px auto;">
      <canvas id="myChart"></canvas>
    </div>



{% if expenselist2 %}
<table>
  <tr>
    <th>Description</th>
    <th>Category</th>
    <th>Date</th>
    <th>Amount</th>
  </tr>
  {% for i in expenselist2 %}

  <tr>
    <td>{{i[2]}}</td>
    <td>{{i[3]}} </td>
    <td>{{i[4]}}</td>
    <td>{{i[5]}}</td>
  </tr>
  {% endfor %}
</table>

{% endif %}

   </div>
    <script>
      const labels = [
        'Remaining',
        'Spent',

      ];

      var remaining = document.getElementById("rem").innerText;
      var expense = document.getElementById("exp").innerText;
      const data = {
        labels: labels,
        datasets: [{
         label: 'Monthly Spending Overview',
         backgroundColor:['rgb(255, 99, 132)',
            'rgb(54, 162, 235)'],

         data: [remaining, expense],
        }]
      };

      const config = {
        type: 'doughnut',
        data: data,
        options: {}
      };

      const myChart = new Chart(
    document.getElementById('myChart'),
    config
  );
    </script>

</body>
```

```
</html>
```

Styles.css

```
*{
  margin: 0px;
  padding: 0px;
  border: 0px;
  box-sizing: border-box;
}
.btn{
  margin: 5px;
}
.content{
  text-align: center;


}

form input{

  margin: 5px;
}
.register,.login .form{
  width: 500px;
  height: auto;
  display: block;
  margin-left: auto;
  margin-right: auto;
}
 form{

  background-color: #f8f8f8;
  padding: 5px;
  margin: 10px auto;
}
p,h1,h2,h3,h4,h5,h6{
  margin: 5px;
}

.account .card{
  margin: 10px auto;
}

.dash-expenses .card{
  display: inline-block;
  margin:15px;
  padding: 15px;
```

```
}



/* register form validation  */


/* Style all input fields */
.register-form input {


    padding: 12px;
    /* border: 1px solid #ccc; */
    /* border-radius: 4px; */
    box-sizing: border-box;
    margin-top: 6px;
    margin-bottom: 16px;
  }


  /* Style the submit button */
  .register-form input[type=submit] {
    background-color: #04AA6D;
    color: white;
  }


  /* Style the container for inputs */
  .container {
    /* background-color: #f1f1f1; */
    padding: 20px;
  }

  /* The message box is shown when the user clicks on the password field */
  #message {
    display:none;
    background: #f1f1f1;
    color: #000;
    position: relative;
    padding: 20px;
    margin-top: 10px;
  }

  #message p {
    padding: 10px 35px;
    font-size: 18px;
  }

  /* Add a green text color and a checkmark when the requirements are right */
  .valid {
    color: green;
  }

  .valid:before {
    position: relative;
```

```
    left: -35px;
    content: "✔";
  }
  /* Add a red text color and an "x" icon when the requirements are wrong */
  .invalid {
    color: red;
  }

  .invalid:before {
    position: relative;
    left: -35px
}
```

GitHub & Project Demo Link

Github link:

https://github.com/IBM-EPBL/IBM-Project-5248-1658752354

Project demo link:

https://drive.google.com/file/d/1RC21ey7ttXvkdCzJKLBlmfxppjq9fkd0/view?usp=share_link