Software Requirements Specification

for

Personal Budget Tracker

Version < 3.0 >

Group No.: 1

CHIN ZHEN HO(Leader)	1221102540
ERIC TEOH WEI XIANG	1221102007
BERNARD RYAN SIM KANG XUAN	1221101777
GAN SHAO YANG	1221103201

Date: <12.2.2025>

CONTENTS

REVISION

Version	Primary Author(s)	Description of Version	Date Completed
V 1.0	CHIN ZHEN HO ERIC TEOH WEI XIANG BERNARD RYAN SIM KANG XUAN GAN SHAO YANG	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	08/12/24
V 2.0	CHIN ZHEN HO ERIC TEOH WEI XIANG BERNARD RYAN SIM KANG XUAN GAN SHAO YANG	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	14/01/25
V 3.0	CHIN ZHEN HO ERIC TEOH WEI XIANG BERNARD RYAN SIM KANG XUAN GAN SHAO YANG	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	12/02/25

1 Project Management

1.1 Team Members

<TO DO: List down the team members and their assigned actor/processes>

Name	Actor/Processes
CHIN ZHEN HO	ADMIN
ERIC TEOH WEI XIANG	USER
BERNARD RYAN SIM KANG XUAN	GUEST
GAN SHAO YANG	FINANCIAL ADVISOR

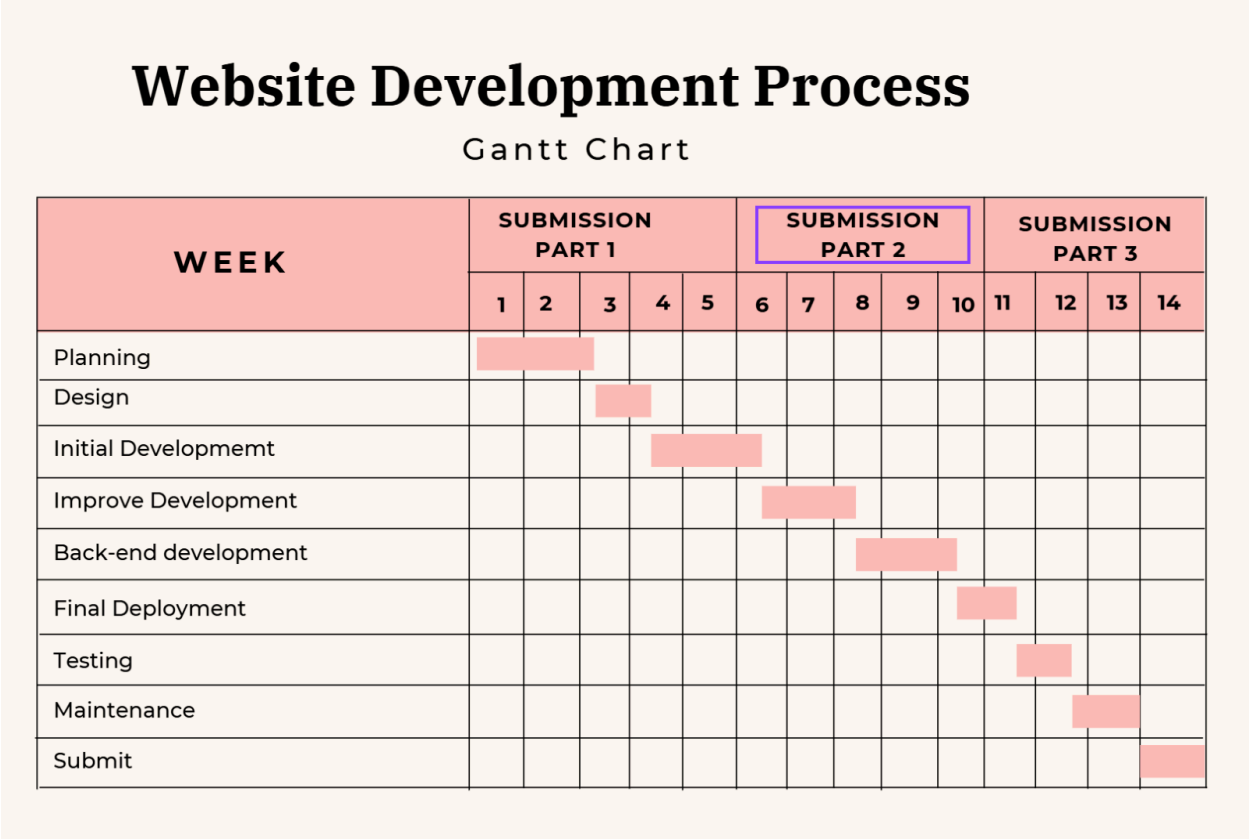
1.2 Problem statement

<TO DO: State the problem as a clear explanation of an issue or challenge that sums up what you want to change.>

Many individuals need help managing their finances effectively, often due to a lack of real-time tracking and understanding of their spending habits. Despite the availability of budgeting tools, current websites and apps often fail to provide an intuitive, comprehensive platform that tracks income, expenses, and savings in a way that is personalized to users' financial goals. This gap in functionality and user-centric design limits users' ability to make informed financial decisions and achieve long-term financial stability.

1.3 Project Plan

<TO DO: Place the project Gantt Chart>



2 System Overview

2.1 Description

<*Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high-level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top-level data flow diagram or object class diagram, will be effective.*
TO DO: Describe the major processes to be performed by the system and the actors involved in each process.>

This web application is designed to help users worldwide track and manage their personal finances, aiming to improve their financial situations. The primary functions of the system are structured around various processes that involve different actors: users, admins, financial advisors, and guests.

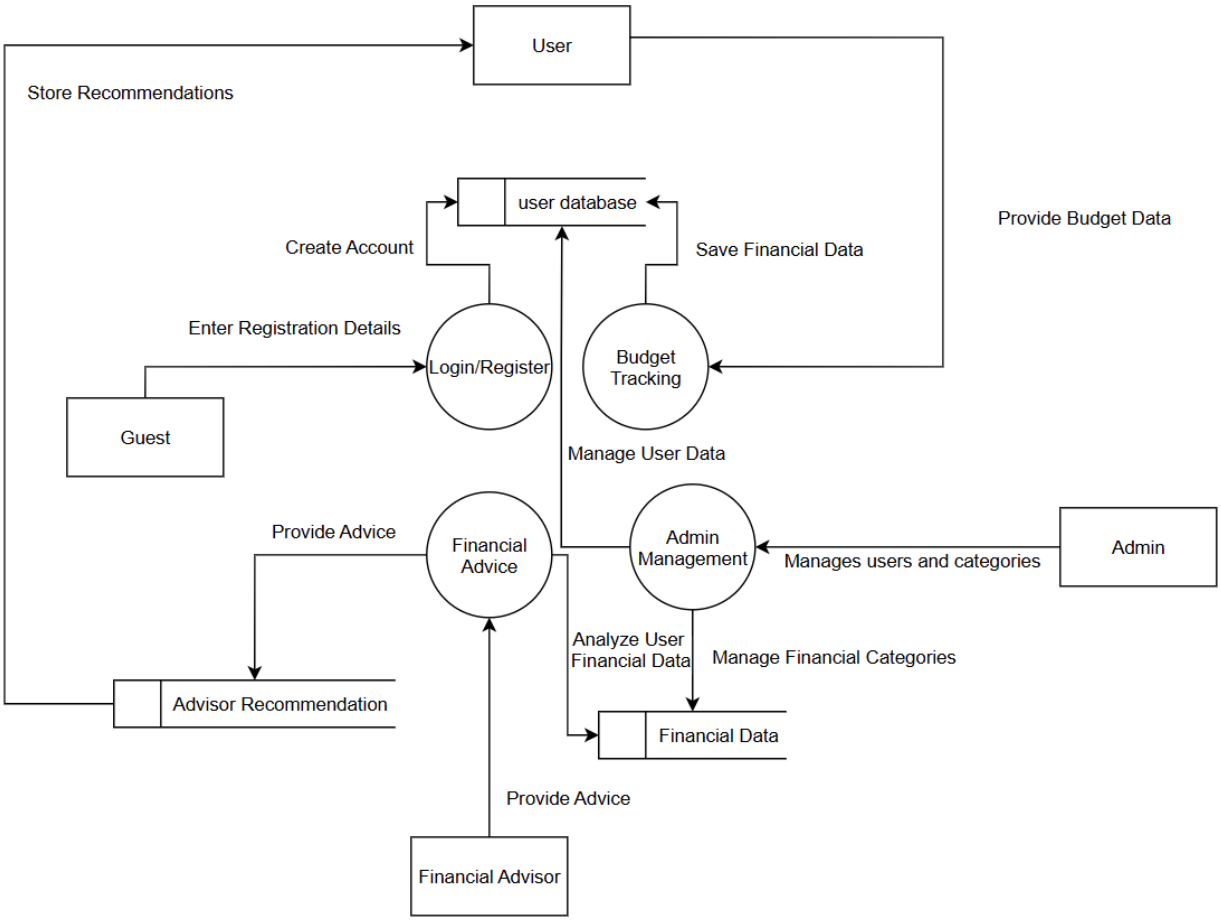
Users of the system can create an account, log in, and manage their financial data. They can input and categorize their income, expenses, and savings, track their spending, and set financial goals. The system provides tools for users to create budgets and monitor their progress towards these goals. Additionally, users can access a financial history to observe spending trends and adjust their budgeting strategies accordingly. For more personalized support, users can also seek guidance from financial advisors who can review their data and offer tailored advice.

Admins have full control over the system and all user accounts. They can create, modify, or delete user profiles and oversee the overall system's functionality. Admins are responsible for managing the integrity of financial data and ensuring the proper functioning of the system, including generating reports on user activity and system performance. In addition to their oversight roles, admins have the capability to monitor budget progress and provide universal support to users.

Financial advisors have access to user data to provide personalized financial advice. They can assist users in setting up budgets, suggest better financial practices, and guide them toward achieving their financial goals. Advisors can view user spending patterns, offer recommendations, and help users optimize their financial decisions.

Guests, who do not have full accounts, are granted limited access to the system. They can explore basic features such as inputting financial data and tracking budgets but cannot save their entries or access advanced features. This allows potential users to get a feel for the application before deciding to sign up for full access.

The interactions between these actors and processes can be effectively represented through a data flow diagram or object class diagram, illustrating how each role contributes to the overall functionality of the web application.



2.2 Actors

<Identify the various actors that will interact with this product.
TO DO: List the actors and the use cases/functions that involve each actor>

Actor	**Use Cases**
Admin	Login as admin
	Manage Categories
	Financial Category Statistics
	Analysis of "Others" Category Descriptions
	View Users Details
User	Log in

	View transaction history
	Add income
	Add expense
	Edit Profile
	View Chart For Total Income and Expense
	Set Budget
	View Message
	View Public Dashboard
Guest	Sign Up
	Log In as Guest
	View transaction history (GUEST)
	Add expense (GUEST)
	Add income (GUEST)
	View Bar Chart Total Income and Expense
Finance Advisor	Log In As Advisor
	View User Details
	Tools
	Personalized Recommendations
	Dashboard Access

2.3 Assumptions and Dependencies Assumptions

Assumptions:

User Technical Proficiency:

It assumes that users have basic technical literacy, meaning they can navigate web applications, input financial data, and interpret visualizations like charts and graphs without extensive technical support or training.

Data Volume and Storage:

It assumes that the platform is designed to scale efficiently and can support thousands of users, ensuring that performance does not degrade as the user base grows and data accumulates.

Development Resources:

It assumes that the development team has access to the necessary tools, frameworks, and libraries required to implement the planned features, including those related to security, user interfaces, and data management.

Dependencies:

Third-Party APIs:

The application will rely on third-party APIs for optional features, such as currency conversion and financial data syncing. This means the functionality of these features is dependent on the availability, stability, and reliability of the third-party services.

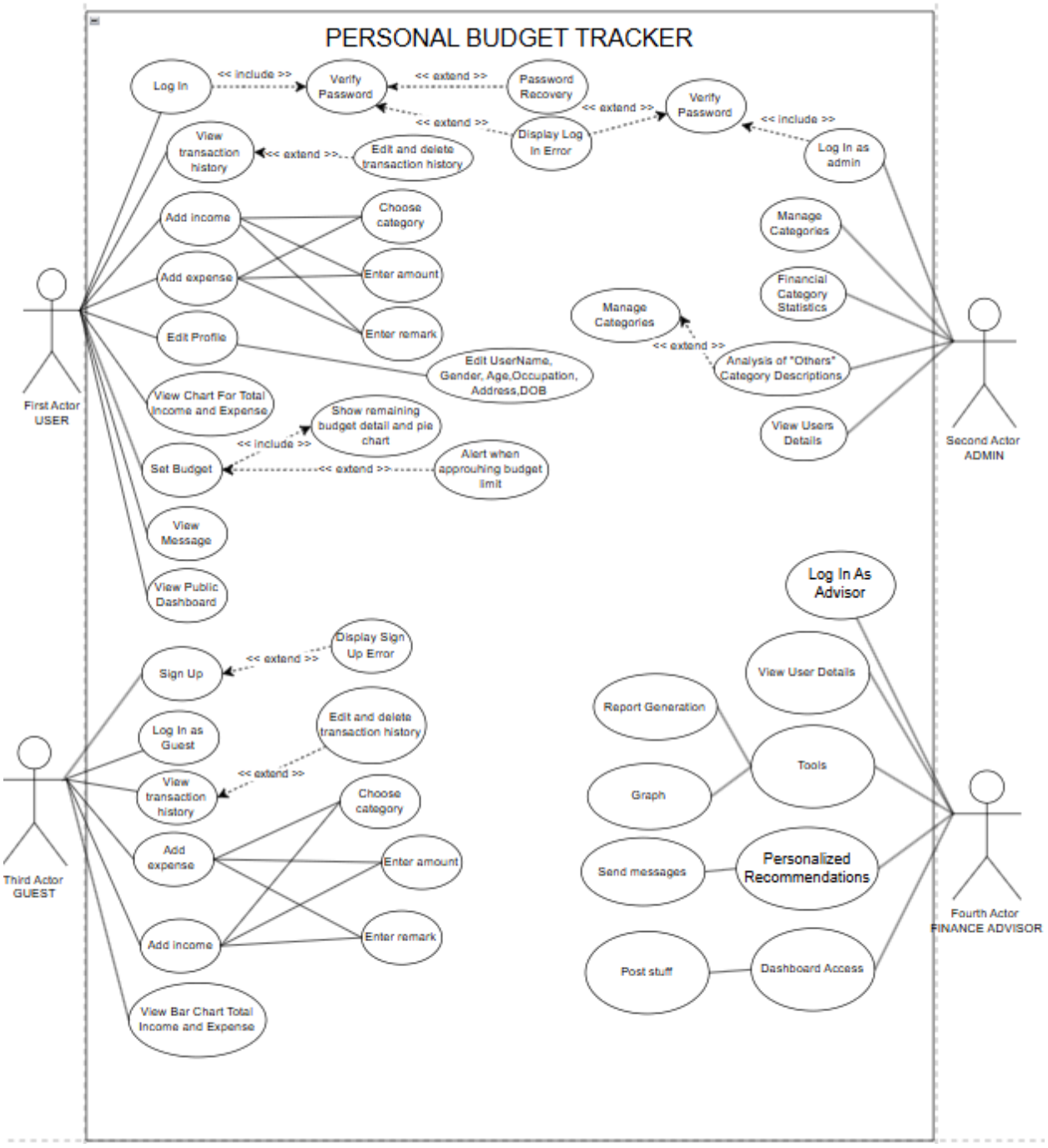
Database and Hosting:

The application depends on a scalable and reliable database solution to store and retrieve user data. The choice of database and hosting infrastructure is critical to ensuring smooth performance, especially as the volume of user data grows.

Browser Compatibility:

The system depends on modern browser standards and ongoing support for critical web APIs. This assumption ensures the application can function across various browsers, but it also relies on continuous updates and support for these standards.

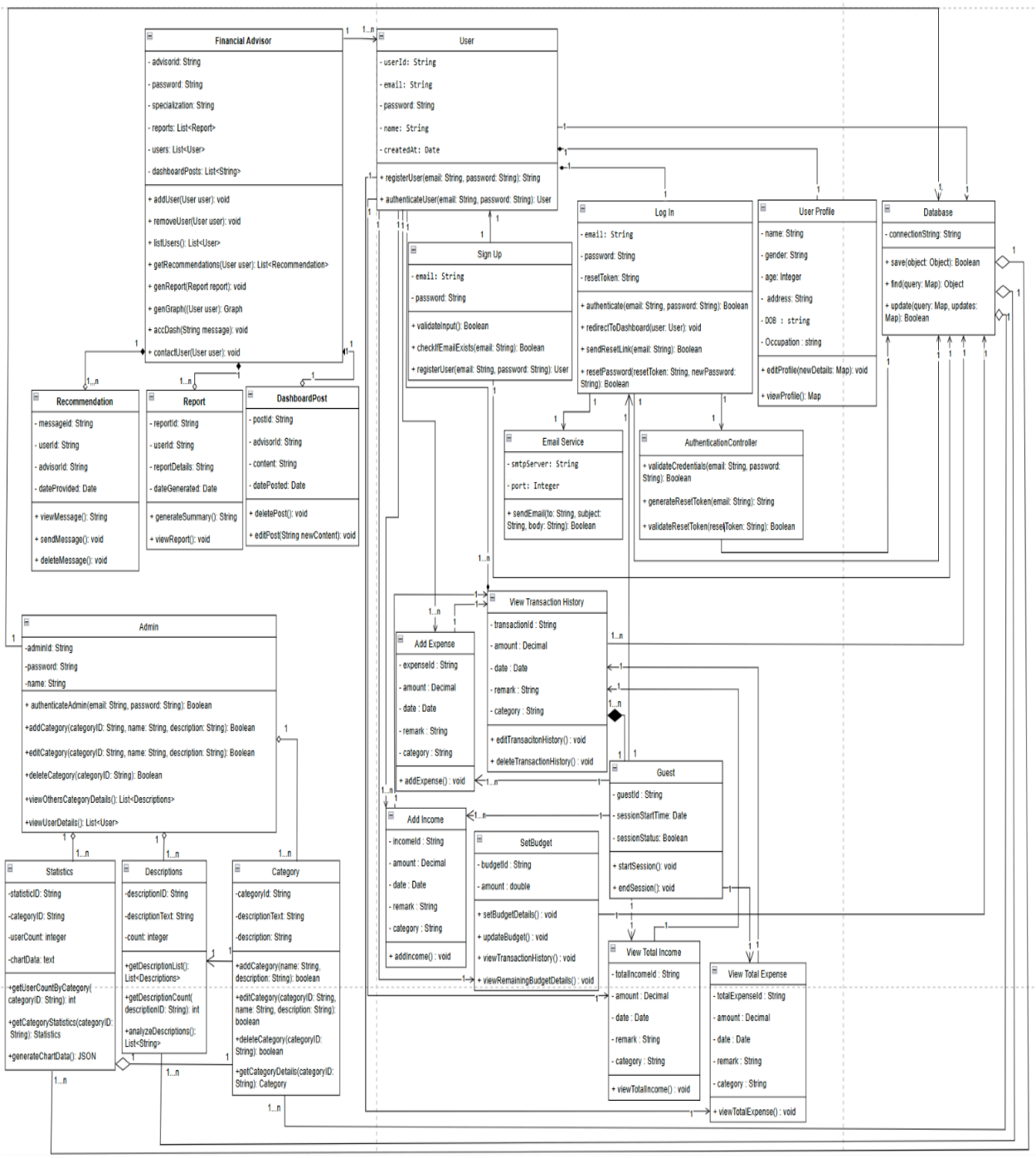
2.4 Use Case Diagram



3 Requirements

3.1 Class Diagrams / ERD

<TO DO: Place the class diagram / ER diagram here.>

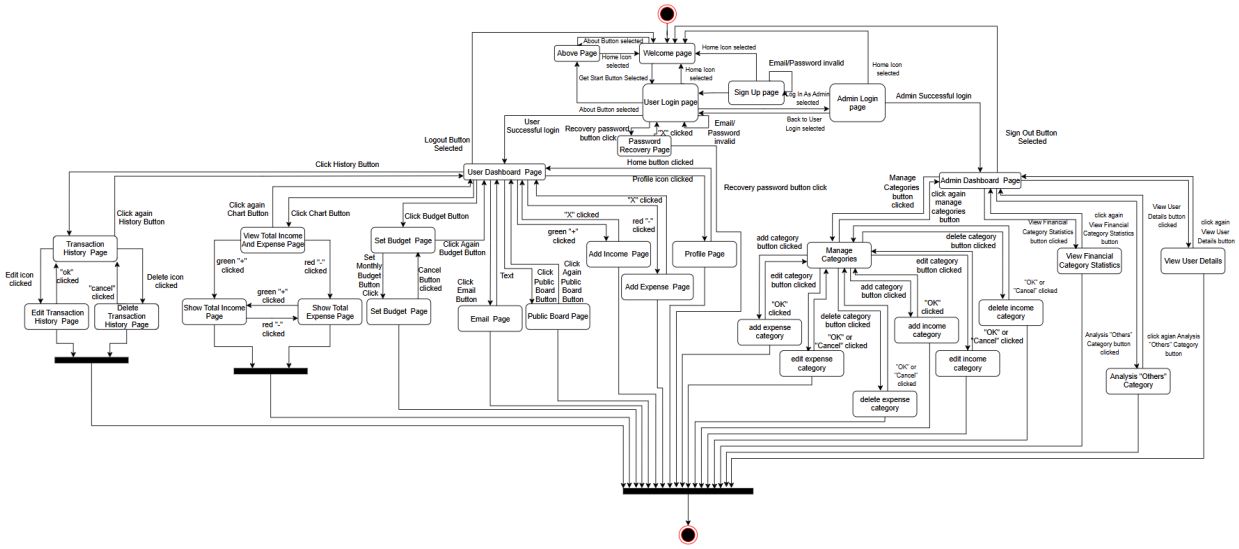


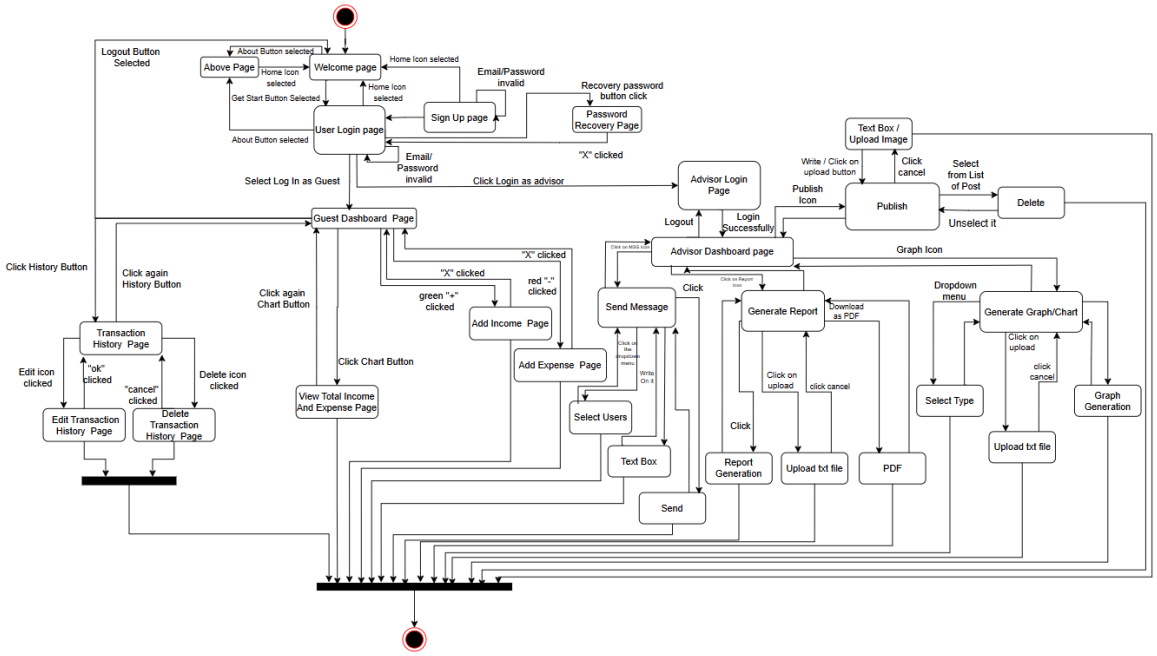
3.2 State Diagrams

<TO DO: Place the state diagram here.>

3.2.1 Overall State Diagram

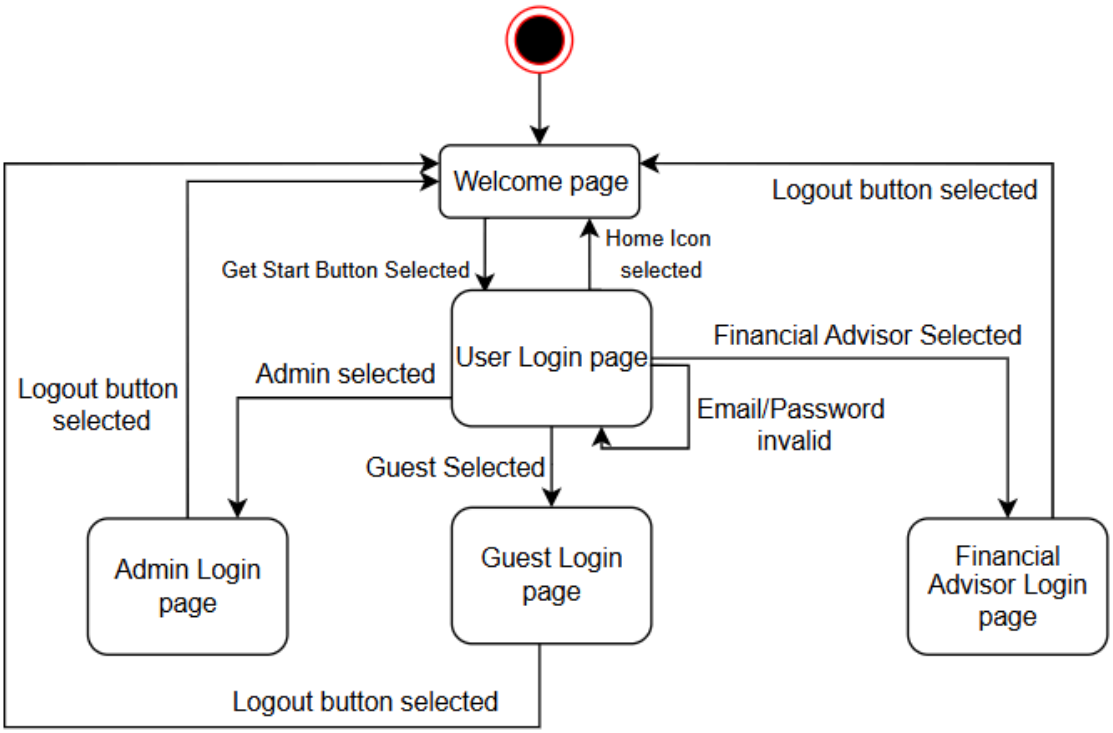
The overall state diagram describes the operation process of the system and the choices made by the user, admin, guest and financial advisor.





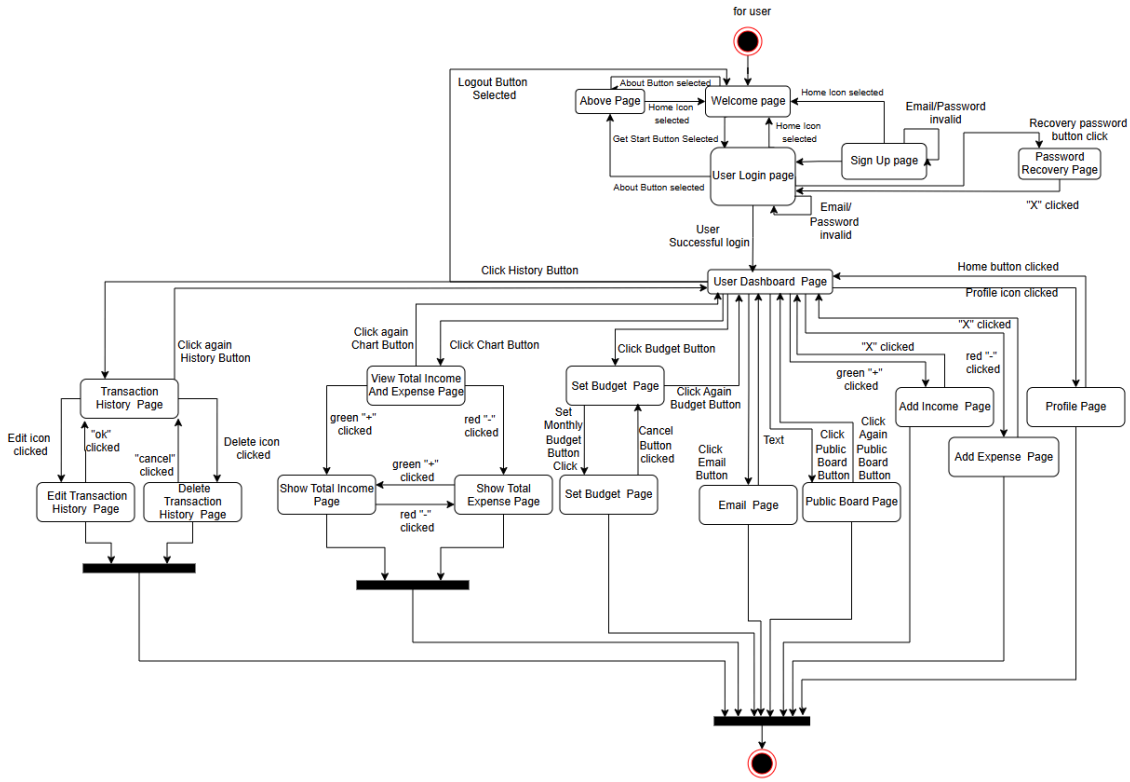
3.2.2 Login State Diagram

Login state diagrams describe the user login according to their responsibilities which sign in at the login page and jump into their respective main page.



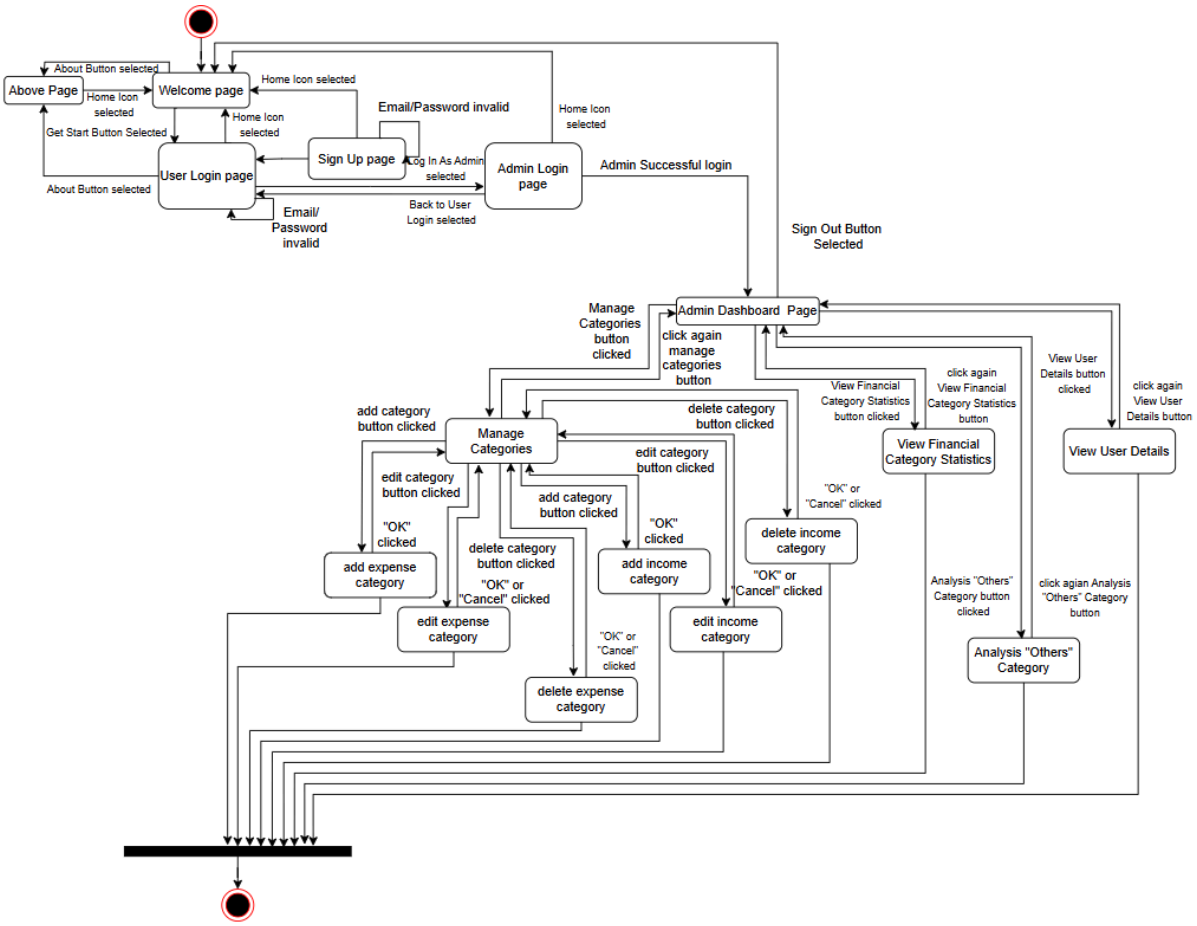
3.2.3 User State Diagram

User state diagram shows that the user after login successfully will jump into main page to run the task that user done which are Transaction History Page, View Total Income And Expense Page, Set Budget Page, Email Page, Public Board Page, Add Income Page, Add Expense Page and Profile Page.



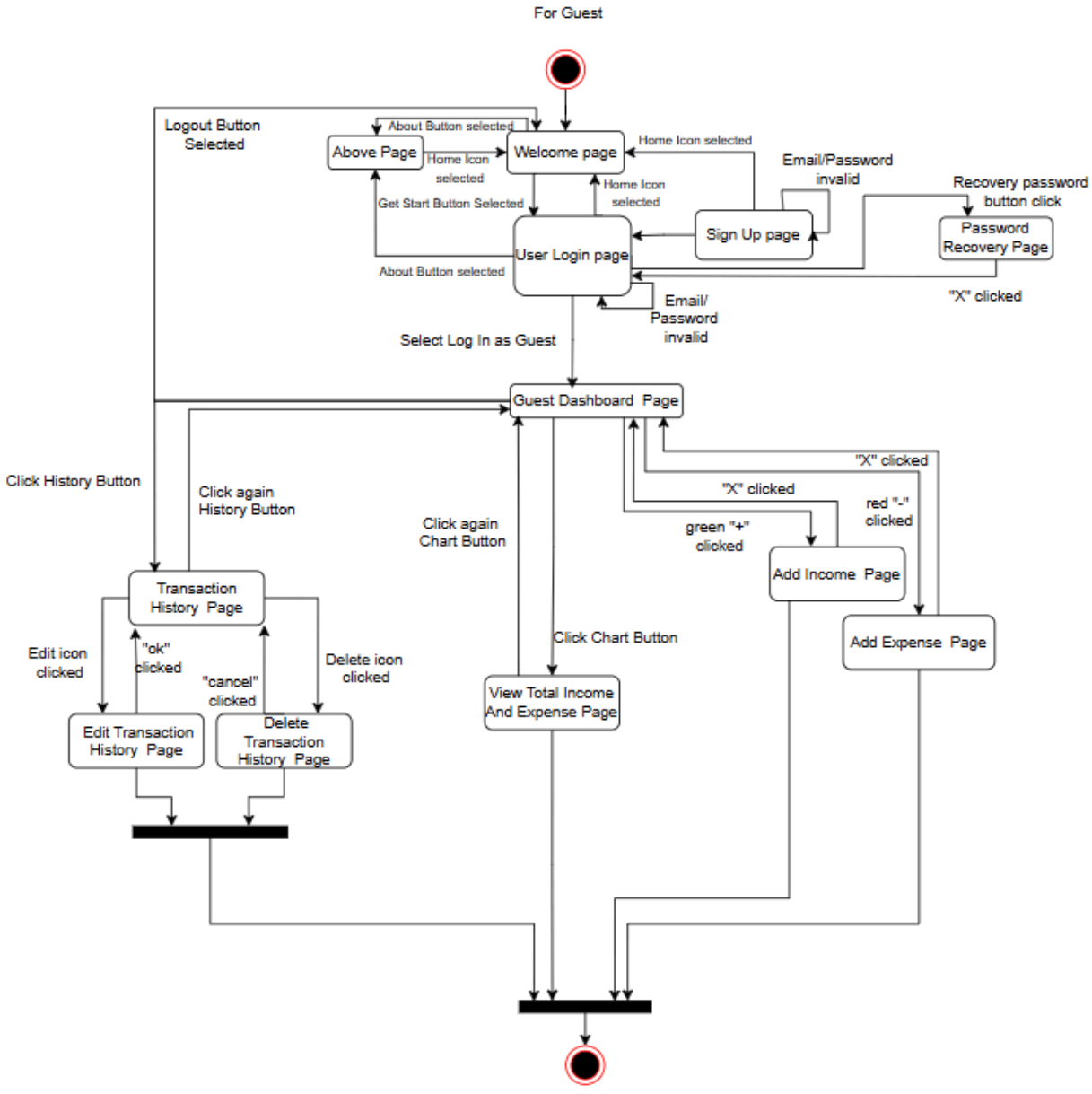
3.2.4 Admin State Diagram

Admin State Diagram illustrates the flow of actions and states an administrator can navigate through in the application. Starting from the Welcome Page, the admin logs in via the Admin Login Page and, upon successful authentication, accesses the Admin Dashboard Page. From here, the admin can manage categories by adding, editing, or deleting income and expense categories, view financial category statistics , analyze "Others" Category Description for reviewing uncategorized data, and access user details. The admin can log out or return to the dashboard at any time, providing a streamlined and efficient process for managing application functionalities.



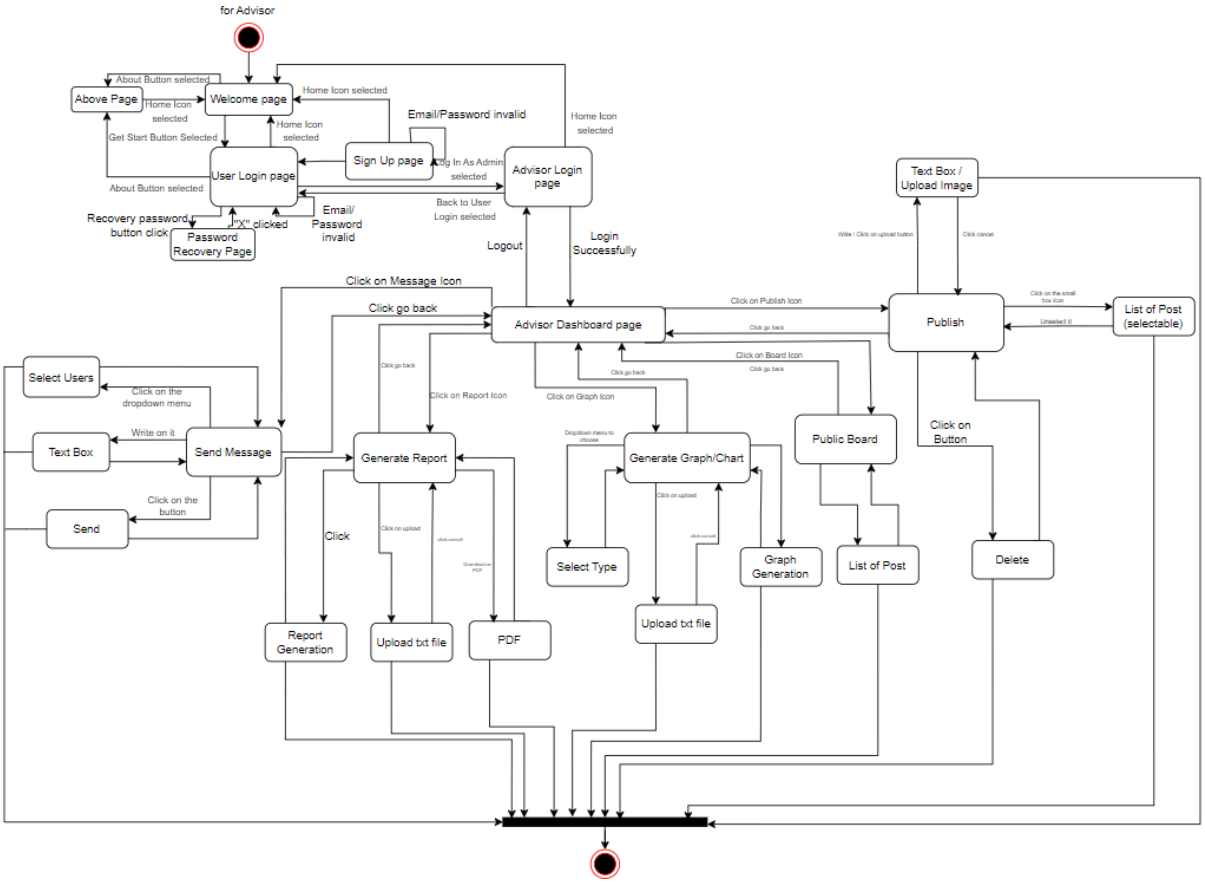
3.2.5 Guest State Diagram

Guest state diagram shows that the Guest after select Log In as Guest will jump into Guest Dashboard Page to run the task that Guest done which are Transaction History Page, View Total Income And Expense Page, Add Income Page, Add Expense Page.



3.2.6 Financial Advisor State Diagram

This diagram displays the transition between each different components, from user login page, advisors choose to click on advisor login page to login as advisor after inputting the correct email and password, and then go into advisor dashboard where they have access to different functions such as send message, generate report, generate graph, public board and publish.



4 Design

4.1 Data Dictionary

<TO DO: Describe the data dictionary and place the table with the details here>

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
email	string	30	-	Yes	Not Null	User's email address
password	string	30	-	Yes	Not Null	User's password

4.1.1 Login

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
userId	string	30	PK	Yes	Not Null	User's unique ID
email	string	30	-	Yes	Not Null	User's email address
password	string	30	-	Yes	Not Null	User's password
name	string	30	-	Yes	Not Null	User's name

4.1.2 User

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
userId	string	30	PK	Yes	Not Null	User's unique ID
name	string	30	-	Yes	Not Null	User's name
gender	string	10	-	Yes	Not Null	User's gender
age	int	5	-	Yes	Not Null	User's age
address	string	50	-	Yes	Not Null	User's address
DOB	Date	10	-	Yes	Not Null	User's Date of Birth
occupation	string	20	-	Yes	Not Null	User's occupation

4.1.3 User Profile

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
connectionString	string	30	-	Yes	Not Null	Connecting string from

						system to database

4.1.4 Database

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
smtpServer	string	30	-	Yes	Not Null	SMTP server is used to send emails.
port	int	10	-	Yes	Not Null	port specifies communication channel for sending emails

4.1.5 Email Server

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
IncomeId	string	30	PK	Yes	Not Null	Income's unique ID
userId	string	30	FK	Yes	Not Null	User's unique ID
amount	int	10	-	Yes	Not Null	Amount that user enter for income

date	Date	10	-	Yes	Not Null	Date that user select to save the income
remark	string	30	-	Yes	Not Null	Remark that user enter for income
category	string	30	-	Yes	Not Null	Category that user choose for income

4.1.6 Add Income

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
ExpenseId	string	30	PK	Yes	Not Null	Expense's unique ID
userId	string	30	FK	Yes	Not Null	User's unique ID
amount	int	10	-	Yes	Not Null	Amount that user enter for Expense
date	Date	10	-	Yes	Not Null	Date that user select to save the Expense

remark	string	30	-	Yes	Not Null	Remark that user enter for expense
category	string	30	-	Yes	Not Null	Category that user choose for expense

4.1.7 Add Expense

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
transactionId	string	30	PK	Yes	Not Null	Transaction history unique ID
userId	string	30	FK	Yes	Not Null	User's unique ID
ExpenseId	string	30	FK	Yes	Not Null	Expense's unique ID
IncomeId	string	30	FK	Yes	Not Null	Income's unique ID
amount	int	10	-	Yes	Not Null	Amount that user enter for Expense
date	Date	10	-	Yes	Not Null	Date that user select to

						save the Expense
remark	string	30	-	Yes	Not Null	Remark that user enter for expense
category	string	30	-	Yes	Not Null	Category that user choose for expense

4.1.8 View Transaction History

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
budgetId	string	10	PK	Yes	Not Null	Monthly estimate budget unique ID
userId	string	30	FK	Yes	Not Null	User's unique ID
transactionId	string	30	FK	Yes	Not Null	Transaction history unique ID
ExpenseId	string	30	FK	Yes	Not Null	Expense's unique ID
IncomeId	string	30	FK	Yes	Not Null	Income's unique ID

amount	int	10	-	Yes	Not Null	Amount that user enter to set monthly estimate budget

4.1.9 Set Budget

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
fileContent	string	Variable	None	No	Null	Holds the content of the uploaded file after it is successfully read.
event	Object	N/A	None	Yes	Not Null	Captures the event information during file upload or button click.
file	File Object	Variable	None	No	Null	Represents the selected file during the upload process
reader	Object	N/A	None	No	Null	FileReader object used to read the content of the uploaded `.txt` file.

lines	Array of Strings	Variable	None	No	Null	Contains formatted file content split into lines to fit within the PDF page
jsPDF	Object	N/A	None	Yes	Not Null	Represents the external library used for generating and managing PDF files.

4.1.10 Generate Report

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
graphType	DOM Element	N/A	None	Yes	Not Null	Dropdown element allowing users to select the graph type (e.g., expenses, budget, savings).
fileInput	DOM Element	N/A	None	Yes	Not Null	File input element for uploading a .txt file containing data.
viewGraphButton	DOM Element	N/A	None	Yes	Not Null	Button element that triggers the generation of the selected graph.

chartCanvas	Canvas Context	N/A	None	Yes	Not Null	The 2D rendering context for the canvas where the graph is displayed.
myChart	Chart Instance	N/A	None	No	Null	Stores the active Chart.js instance for the graph.
graph	String	Variable	None	Yes	Not Null	Holds the value of the selected graph type from the dropdown menu.
file	File Object	Variable	None	Yes	Null	Represents the uploaded .txt file containing graph data.
fileData	Array of Strings	Variable	None	No	Null	Holds the processed content of the uploaded file, split by lines

4.1.11 Generate Graph

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
userId	String	Variable	None	Yes	Not Null	The selected user's unique Firestore document ID.
message	String	Variable	None	Yes	Not Null	The message text typed by the user to send.
timestamp	Date	N/A	None	Yes	Not Null	The date and time when the message is sent, stored in Firestore.
firebaseConfig	Object	N/A	None	Yes	Not Null	Stores Firebase configuration settings such as API keys and project details.
messageInput	DOM Element	N/A	None	Yes	Not Null	Text input field for typing the message to send.
sendMessageButton	DOM Element	N/A	None	Yes	Not Null	Button to trigger sending the message.

statusMessage	DOM Element	N/A	None	Yes	Not Null	Element displaying status messages such as errors or success notifications

4.1.12 Messages

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
publicBoardList	DOM Element	N/A	None	Yes	Not Null	Represents the element where posts are dynamically appended as list items.
emptyMessage	DOM Element	N/A	None	Yes	Not Null	Represents the paragraph element showing a "No posts available" message when no posts exist.
publicBoardPosts	Array of Objects	Variable	None	Yes	Not Null	An array retrieved from localStorage containing all posts to be displayed on the public board

post.text	String	Variable	None	Yes	Not Null	The text content of a single post retrieved from the publicBoardPosts array.
post.image	String	Variable	None	Yes	Null	The URL of an image associated with a single post, if available.
postContent	DOM Element	N/A	None	Yes	Not Null	A element that holds the content (text and image) for a single post.
img	DOM Element	N/A	None	No	Null	Represents an <img> element containing the image of a post, if one exists

4.1.13 Board

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
email	string	50	-	Yes	Not Null	Admin's email address
password	string	50	-	Yes	Not Null	Admin's password

4.1.14 Admin

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
categoryId	string	30	PK	Yes	Not Null	Unique ID for each category
name	string	50	-	Yes	Not Null	Name of the category
type	string	10	-	Yes	Not Null	Specifies if the category is income or expense

4.1.15 Category

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
description	string	255	-	Yes	Not Null	User-defined description for the "Others" category
expensesCount	int	-	-	Yes	Not Null	Total number of times this description was used in expenses
incomesCount	int	-	-	Yes	Not Null	Total number of times this description was used in incomes

4.1.16 Others Category Analysis

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
expenseStatistics	Map<String, Int>	-	-	Yes	Not Null	Stores expense categories and their usage counts
incomeStatistics	Map<String, Int>	-	-	Yes	Not Null	Stores income categories and their usage counts

4.1.17 Financial Statistics

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
categoryId	string	30	PK	Yes	Not Null	Unique identifier for the category
userId	string	30	FK	Yes	Not Null	Foreign key referencing the User table
amount	int	-	-	Yes	Not Null	Amount associated with the category
date	date	-	-	Yes	Not Null	Date the category is used

remark	string	255	-	Yes	Null	Remark provided for the category

4.1.18 Income and Expense Categories

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
email	string	30	-	Yes	Not Null	User's email address
password	string	30	-	Yes	Not Null	User's password

4.1.19 Sign Up

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
incomeData	string	30	PK	Yes	Not Null	Income's unique ID
amount	int	10	-	Yes	Not Null	Amount that guest enter for income
date	Date	10	-	Yes	Not Null	Date that guest select

						to save the income
remark	string	30	-	Yes	Not Null	Remark that guest enter for income
category	string	30	-	Yes	Not Null	Category that guest choose for income

4.1.20 Add Income (GUEST)

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
expenseData	string	30	PK	Yes	Not Null	Expense's unique ID
amount	int	10	-	Yes	Not Null	Amount that guest enter for Expense
date	Date	10	-	Yes	Not Null	Date that guest select to save the Expense
remark	string	30	-	Yes	Not Null	Date that guest select

						to save the Expense
category	string	30	-	Yes	Not Null	Category that guest choose for expense

4.1.21 Add Expense (GUEST)

Field Name	Data Type	Length	PK/FK	Required	Null / Not Null	Description
transactionId	string	30	PK	Yes	Not Null	Transaction history unique ID
expenseData	string	30	FK	Yes	Not Null	Expense's unique ID
incomeData	string	30	FK	Yes	Not Null	Income's unique ID
amount	int	10	-	Yes	Not Null	Amount that guest enter for Expense
date	Date	10	-	Yes	Not Null	Date that guest select to save the Expense

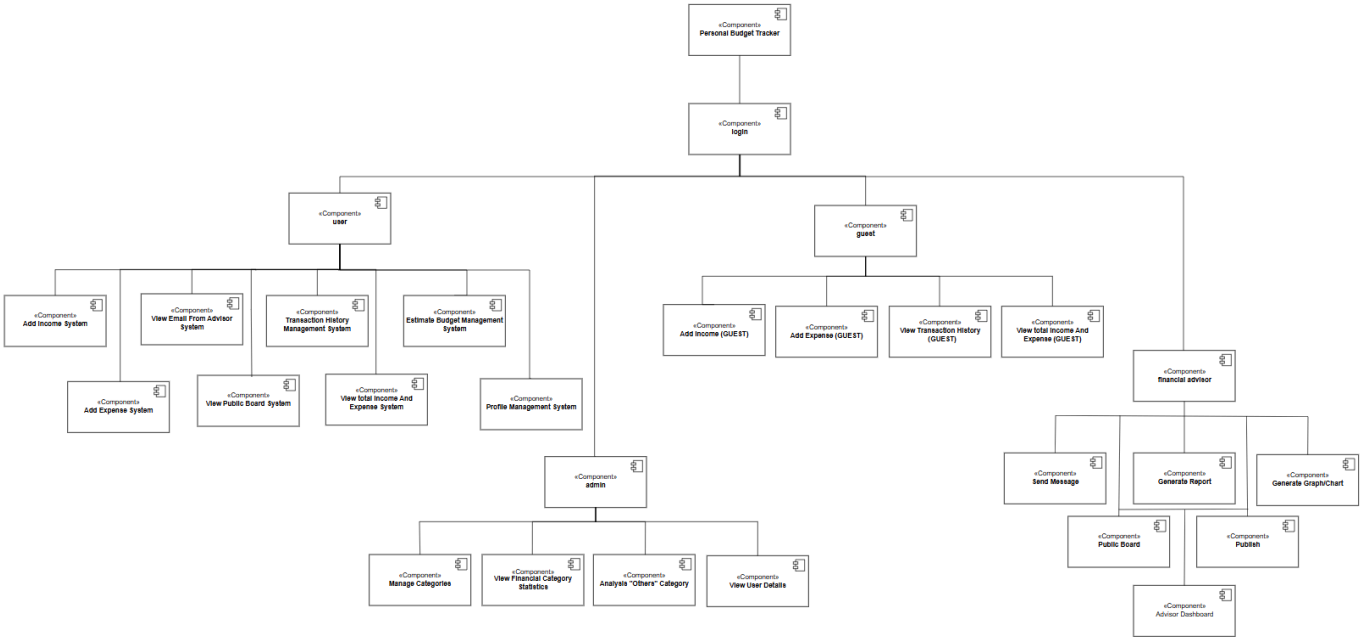
remark	string	30	-	Yes	Not Null	Remark that guest enter for expense
category	string	30	-	Yes	Not Null	Category that guest choose for expense

4.1.22 View Transaction History (GUEST)

4.2 Software Architecture

<TO DO: Describe the software architecture and place the architecture diagram here.>

The Personal Budget Tracker System consists of five subsystems, which are login, user, admin, guest and financial advisor. The login subsystem acts as the entry point, directing users to the appropriate subsystem based on their roles.



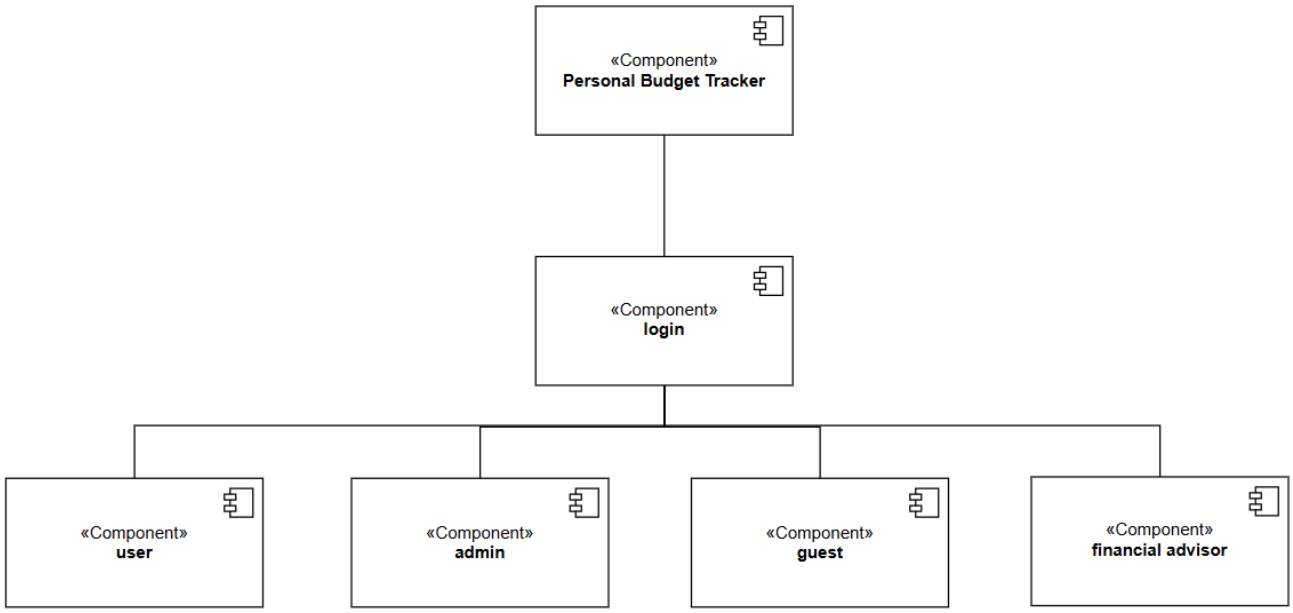
<TO DO: Describe the separation of the system into subsystems and how the subsystems are assigned to team members.>

Subsystem	Team members
Login Subsystem	Each member done for it own actor login function

User Subsystem	ERIC TEOH WEI XIANG
Admin Subsystem	CHIN ZHEN HO
Guest Subsystem	BERNARD RYAN SIIM KANG XUAN
Financial Advisor Subsystem	GAN SHAO YANG

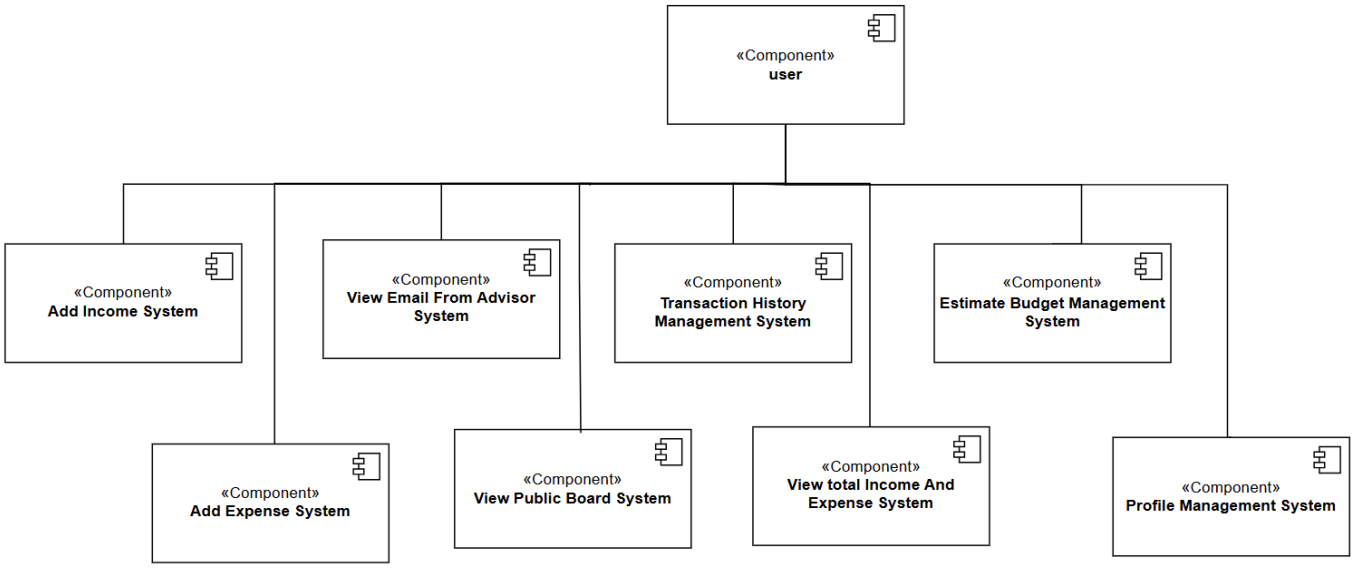
4.2.1 Login Subsystem

The login subsystems are responsible for user authentication and authorization. Upon successful login, the user is directed to one of the four specific subsystems based on their role : User, Admin, Guest and Financial Advisor.



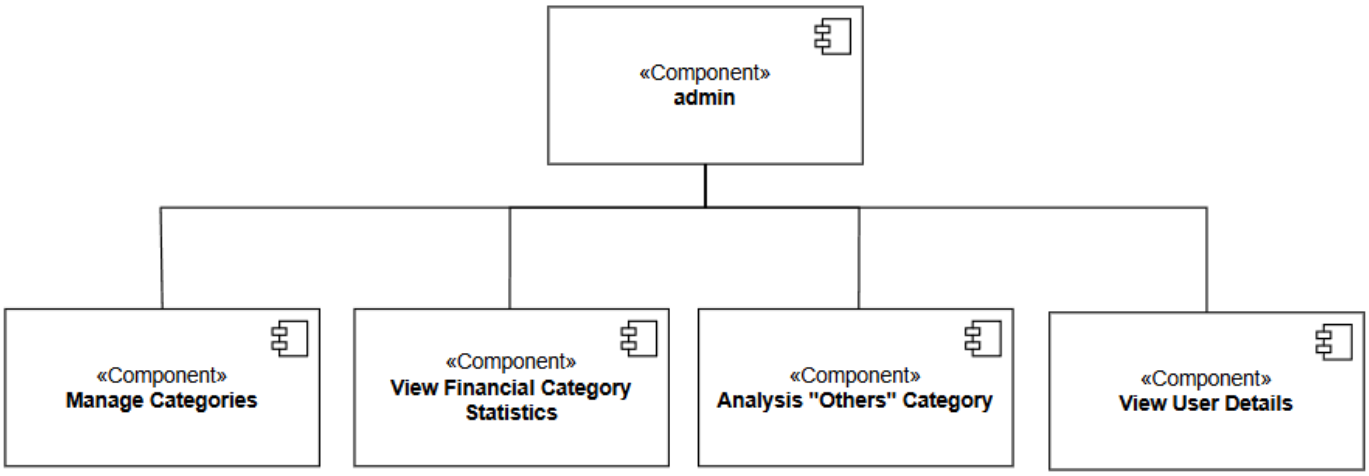
4.2.2 User Subsystem

The user subsystem consists of eight functions which are add income system, add expense system, view email from advisor system, view public dashboard system, transaction history management system, view total income and expense system, estimate budget management system and profile management system.



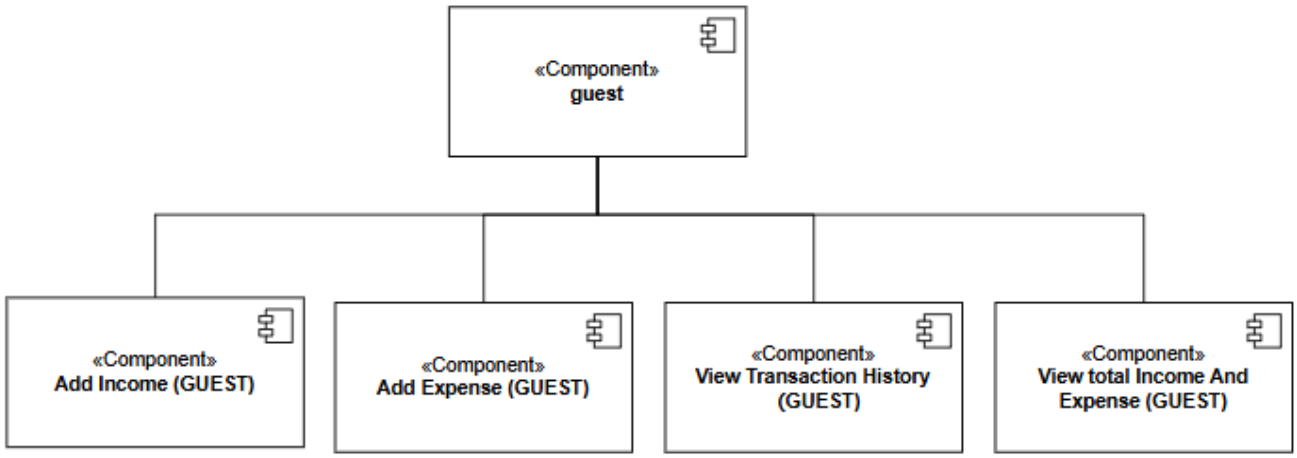
4.2.3 Admin Subsystem

The Admin Subsystem consists of four main functions, which are Manage Categories, View Financial Category Statistics, Analysis of "Others" Category, and View User Details. The Manage Categories function allows administrators to create, edit, or delete financial categories to ensure proper organization. The View Financial Category Statistics function provides insights into categorized data, helping to monitor trends and financial performance. The Analysis of "Others" Category function focuses on reviewing uncategorized data. Lastly, the View User Details function enables administrators to access and manage user profiles, monitor activities, and assist users as needed.



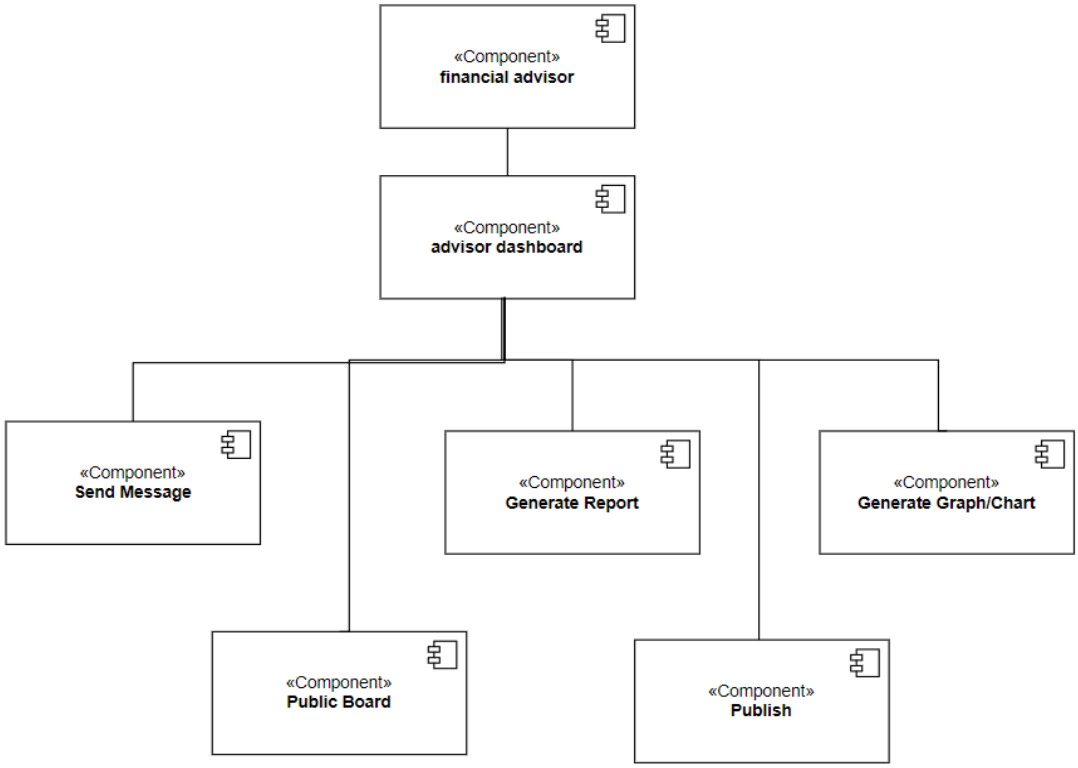
4.2.4 Guest Subsystem

The Guest subsystem consists of four functions which are add income (GUEST), add expense (GUEST), view transaction history (GUEST), view total income and expense (GUEST).



4.2.5 Financial Advisor Subsystem

The advisor subsystem consists of 5 functions which are send Message, generate Report, generate Graph/Chart, public board and publish.

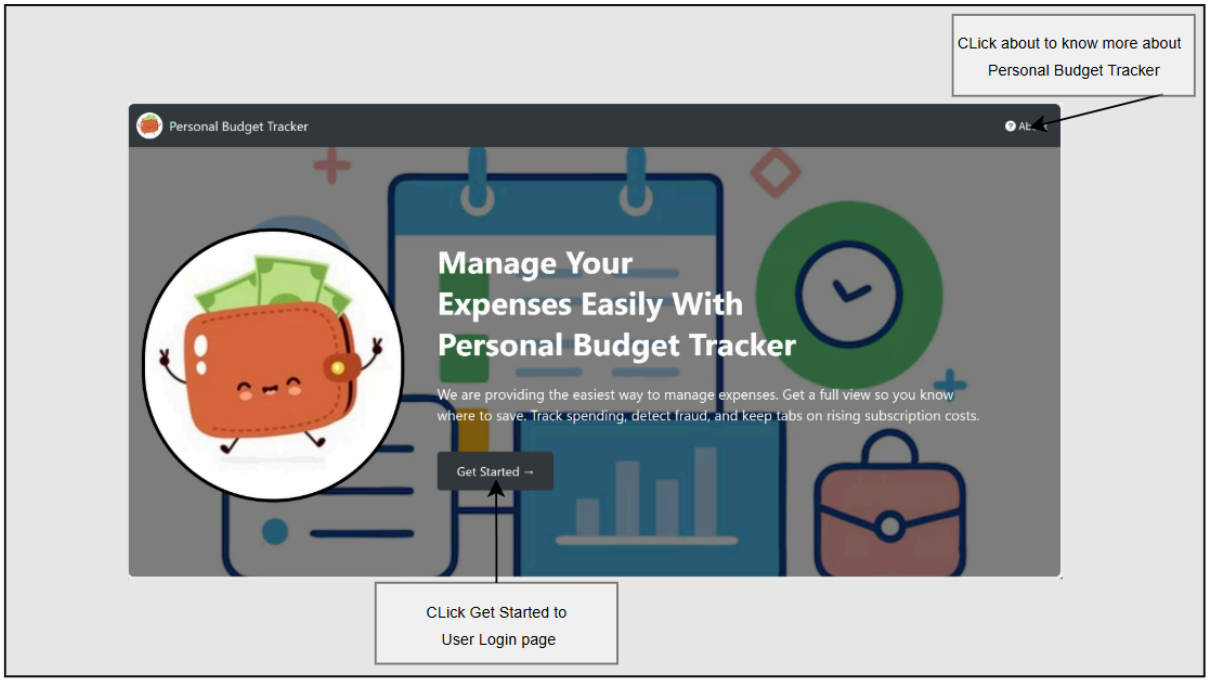


4.3 Main Screens

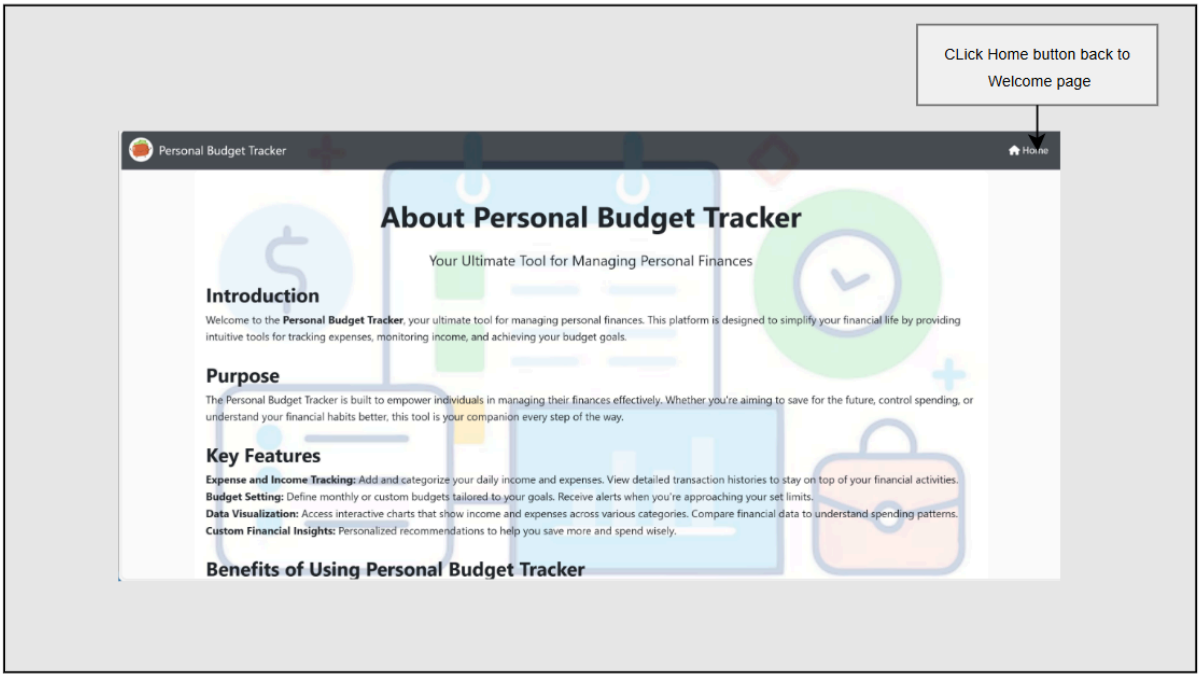
<TO DO: Describe the main screens of the system and place the screen designs here.>

4.3.1 Welcome Page

The **Personal Budget Tracker** screen includes a friendly wallet logo, a bold headline, key features summary, a clear "Get Started" button, and an "About" button for app details, all within a modern, approachable design.

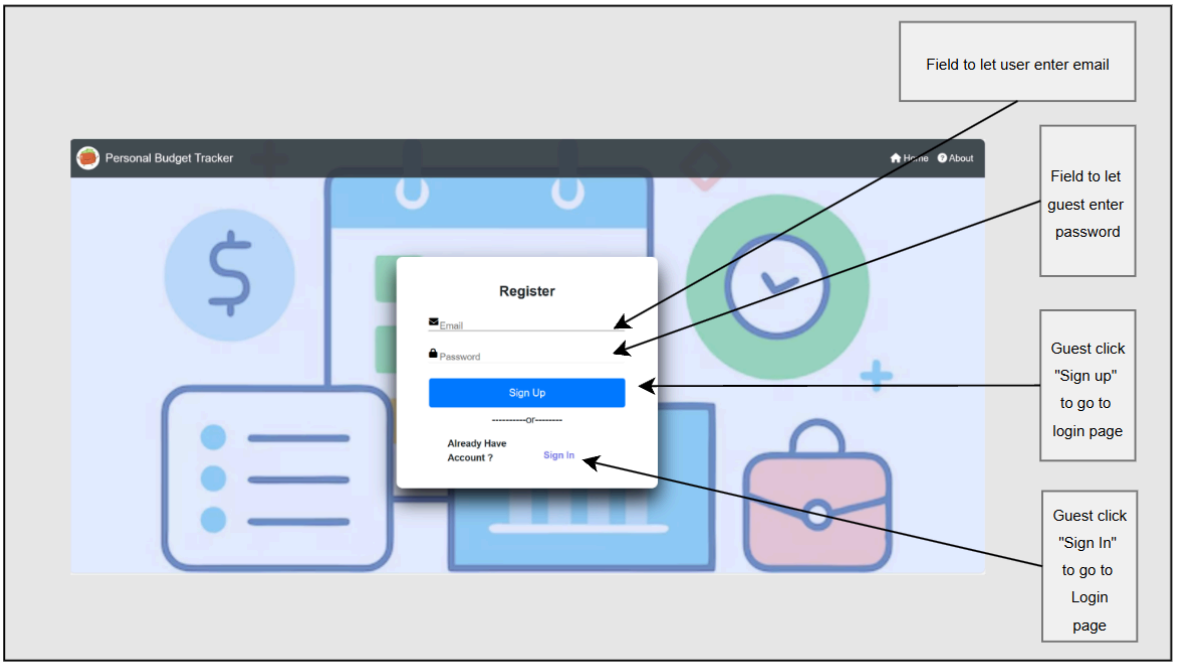


After the user clicks the about button will go to the about page to know more about personal budget tracker.



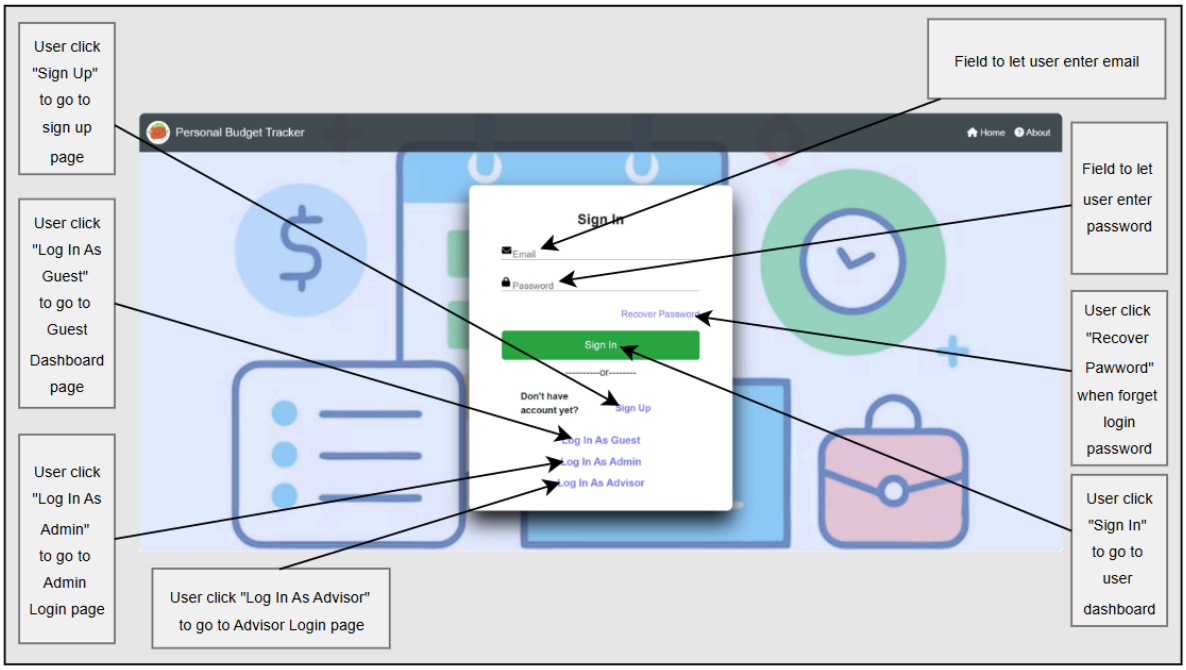
4.3.2 Sign Up Page

The **Sign-Up Page**, where guests can register by entering their email and setting a password. The page features a prominent "Sign Up" button to create an account, and for those who already have accounts, a "Sign In" link is included to redirect them to the login page. This design ensures a seamless user registration experience.



4.3.3 Login Page

The **Login Page** of the Personal Budget Tracker, featuring fields for users to input their email and password for authentication. It includes a "Recover Password" link to help users reset forgotten credentials, a "Sign In" button to log in and access the user dashboard, and additional options for "Log In as Guest," "LogIn as Admin," and "Log In as Advisor" for role-specific access. A "Sign Up" link is also provided to redirect new users to the registration page.

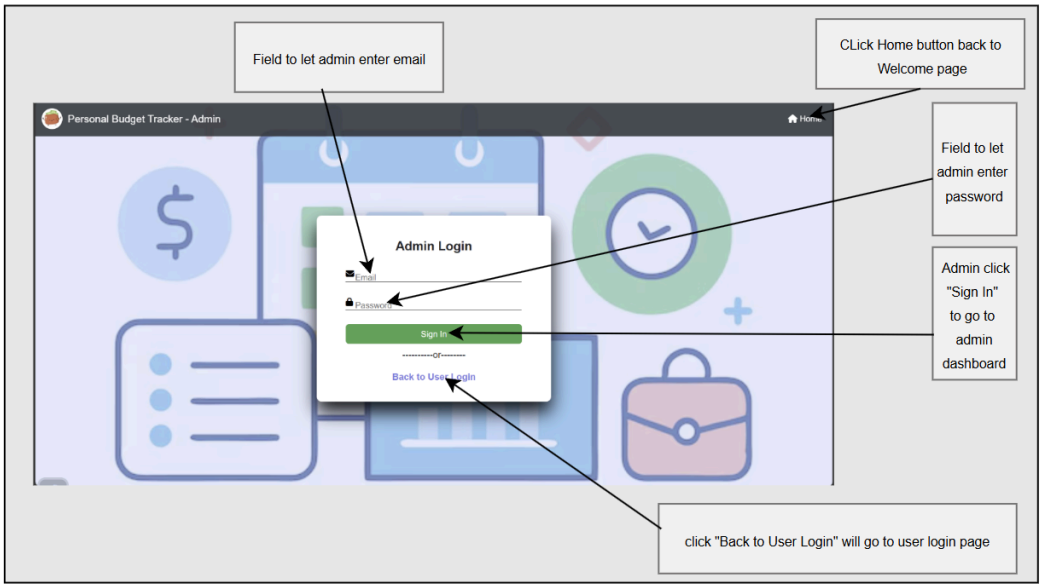


4.3.4 Password Recovery Page

The **Password Recovery Page**, designed to help users recover their accounts. It includes a field for entering the registered email and a "Send Reset Link" button to confirm the entry and initiate the password recovery process. This page provides a simple and efficient solution for users to reset their login credentials and regain access to their accounts.

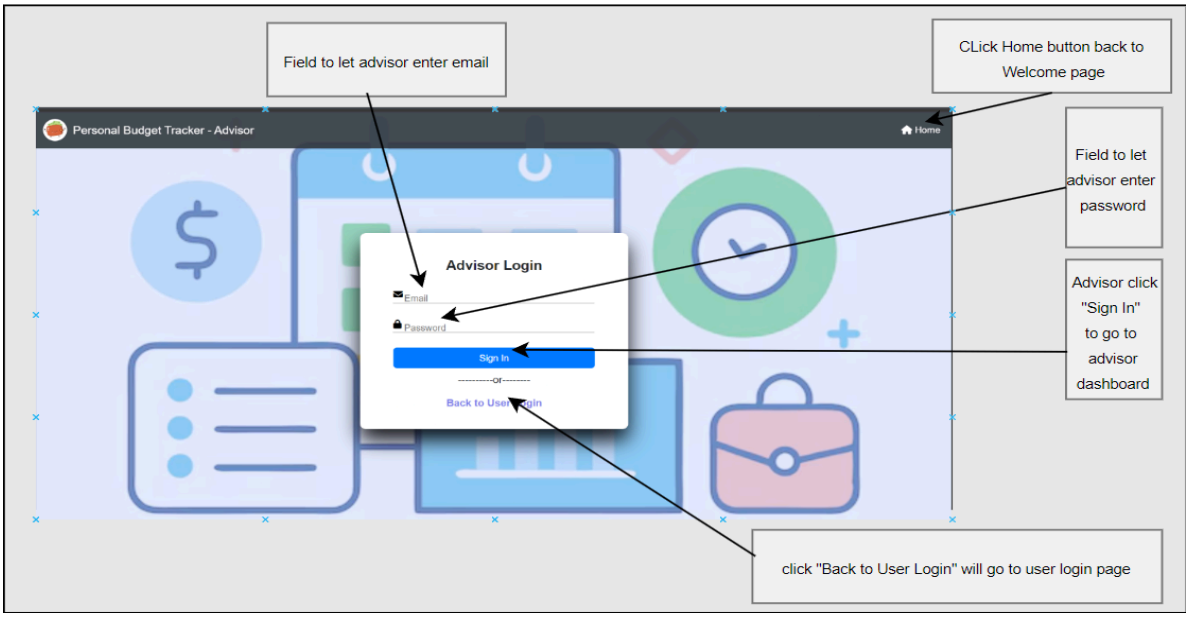
4.3.5 Admin Login Page

The **Admin Login Page** of the Personal Budget Tracker is a secure interface designed for administrators to access the system. It features a clear and simple design with an input field for the admin's email, marked with an envelope icon, and a password field, marked with a lock icon, to ensure security. A green **Sign In** button allows administrators to authenticate their credentials and proceed to the Admin Dashboard. For additional navigation, a **Back to User Login** link is provided, enabling users to return to the user login page. Additionally, a **Home** button in the top-right corner allows users to return to the main welcome page.



4.3.6 Financial Advisor Login Page

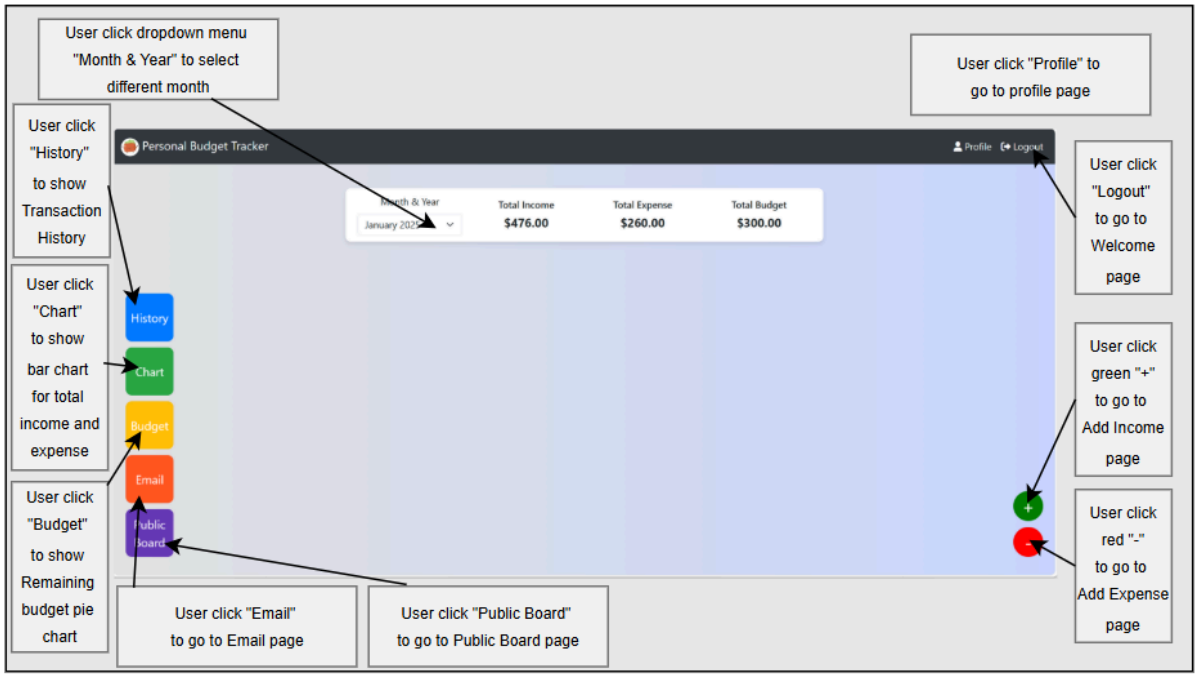
This is a important page as it serves as a security which means it only allows people who are advisors and are given these email and passwords to login, this specific details are stored inside the database so the web page will fetch it out and if the details are correct then it will lead advisors to their personal dashboard.



4.4 Subsystem User Screen

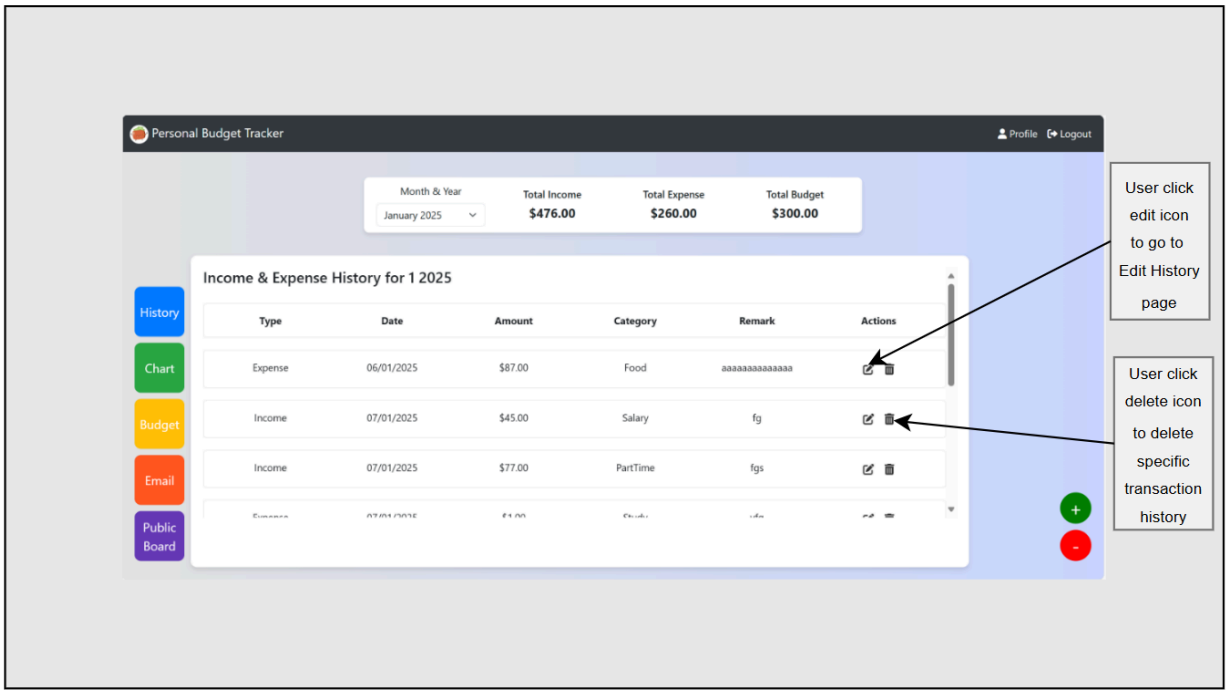
4.4.1 Main Page (User Dashboard)

The **User Dashboard** of the Personal Budget Tracker provides an organized and user-friendly interface for managing finances. At the top, a dropdown menu allows users to select the "Month & Year" to view specific financial data, with key metrics such as **Total Income**, **Total Expense**, and **Total Budget** displayed prominently. On the left, a vertical menu offers quick navigation to various features, including **History** for viewing transaction records, **Chart** for visualizing income or expense bar charts, **Budget** for tracking remaining budget via a pie chart, **Email** for accessing email features, and **Public Board** for viewing shared boards. In the top-right corner, users can access their **Profile** or log out via the **Logout** button. Additionally, floating buttons in the bottom-right corner allow users to add income with the **green "+" button** or add expenses with the **red "-" button**, ensuring easy access to essential actions.



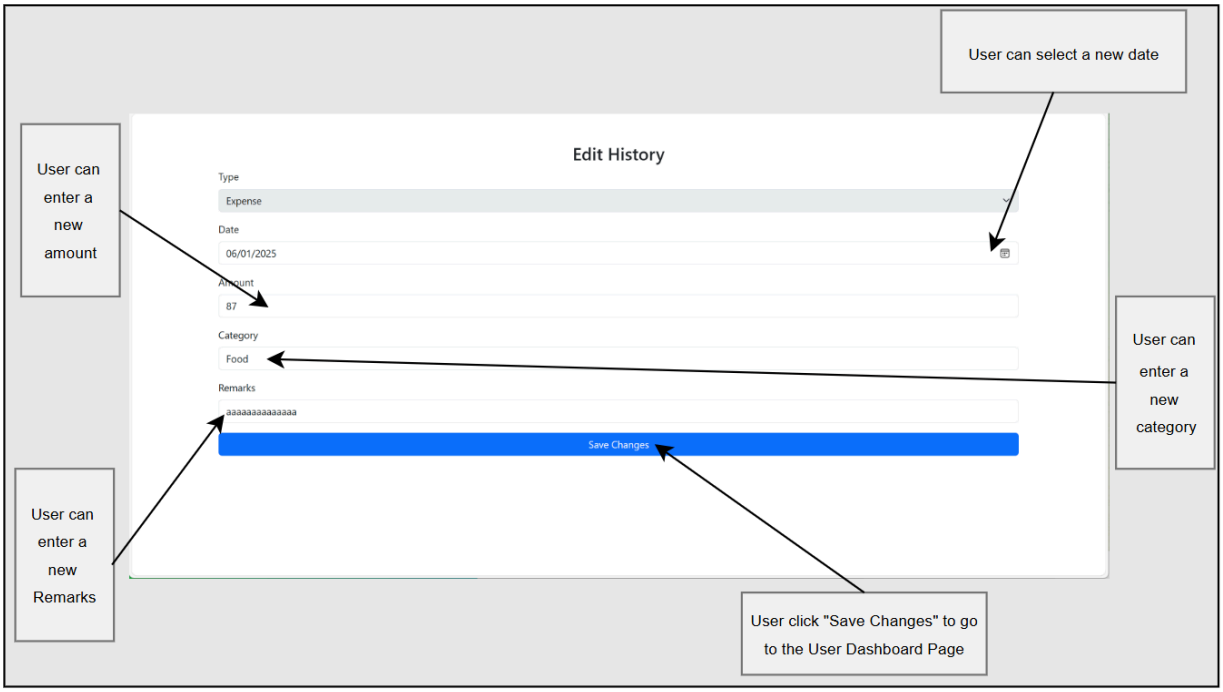
4.4.2 Transaction History Page

The **Transaction History Page** displays a detailed list of the user's income and expense records for the selected month and year. Each record includes fields for type (income or expense), date, amount, category, and remarks. Users can interact with each record through the action icons: clicking the **edit icon** navigates to the **Edit History Page**, where the record can be modified, while clicking the **delete icon** removes the respective transaction from the history. This layout allows for clear and efficient management of transaction data.



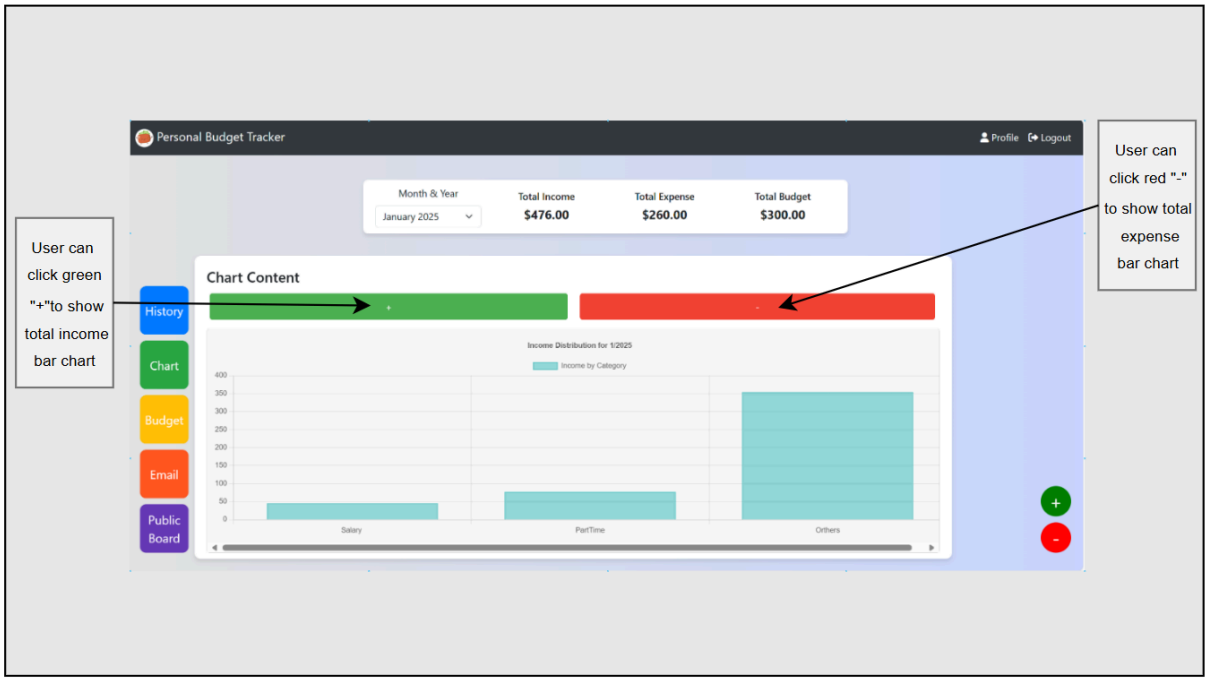
4.4.3 Edit History Page

The **Edit History Page** provides a form for users to modify specific transaction details. Users can select a new date, change the amount, edit the category, and add or revise remarks. Once the necessary changes are made, clicking the **Save Changes** button updates the transaction and redirects the user back to the dashboard. This page offers a straightforward and user-friendly interface for editing transaction records, ensuring data accuracy and flexibility.



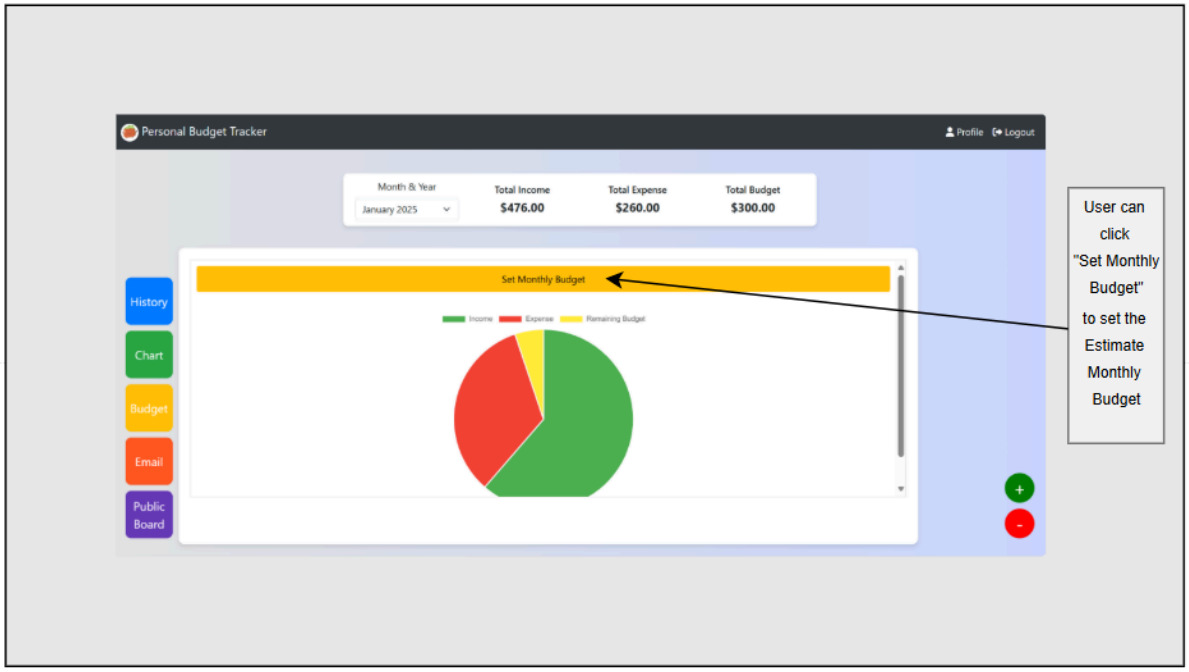
4.4.4 View Total Income And Expense Page

The **View Total Income and Expense Page** provides a graphical representation of the user's financial data. At the top, users can switch between viewing income and expense data by clicking the **green "+" button** to display a bar chart of total income or the **red "-" button** to show a bar chart of total expenses. The chart content dynamically updates based on the selected category, offering a clear visualization of income and expense distribution by category. This page ensures an easy-to-understand summary of financial data, helping users analyze their spending and earning patterns effectively.



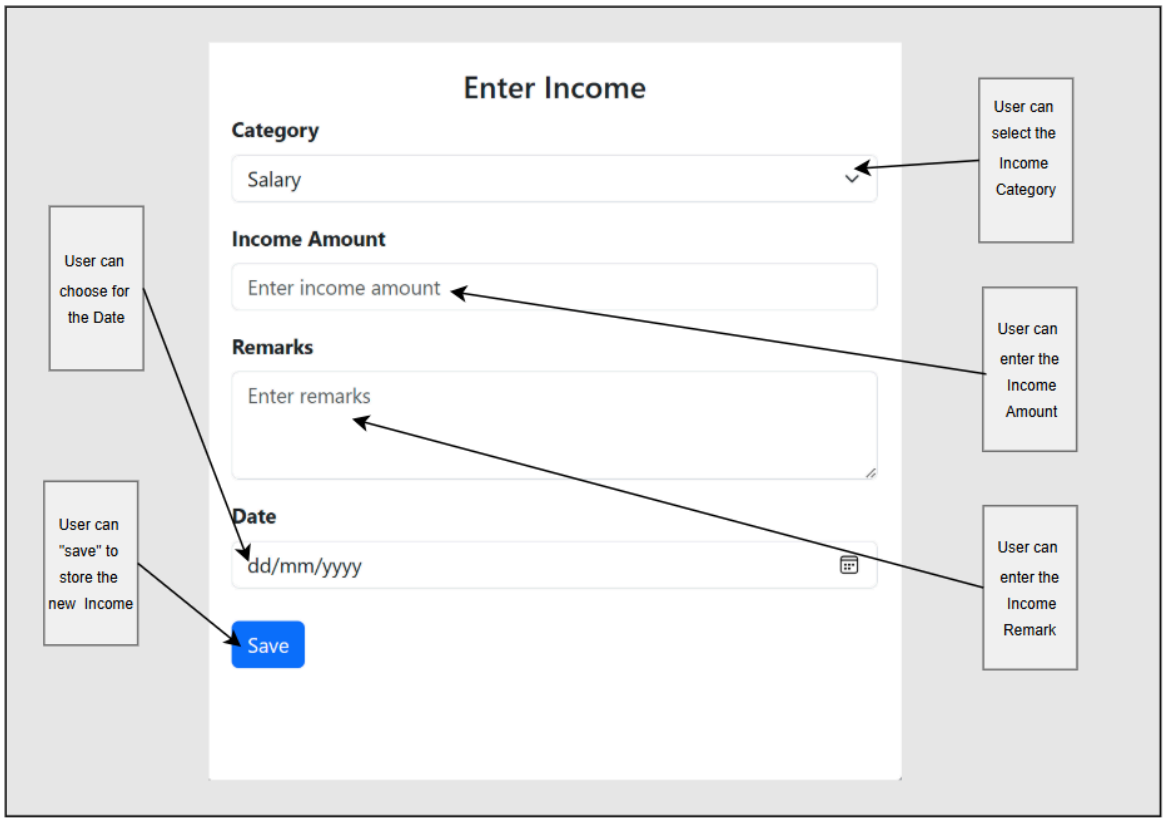
4.4.5 Set Budget Page

The **Set Budget Page** allows users to manage and monitor their monthly budgets effectively. At the top, users can click the **"Set Monthly Budget"** button to input or update their estimated monthly budget. A pie chart visually represents the breakdown of **income**, **expenses**, and the **remaining budget**, providing a clear snapshot of the user's financial status. Additionally, the system is programmed to alert users when their remaining budget reaches **10%**, ensuring they stay informed and can take necessary actions to avoid overspending. This page offers an intuitive and proactive approach to budget management.



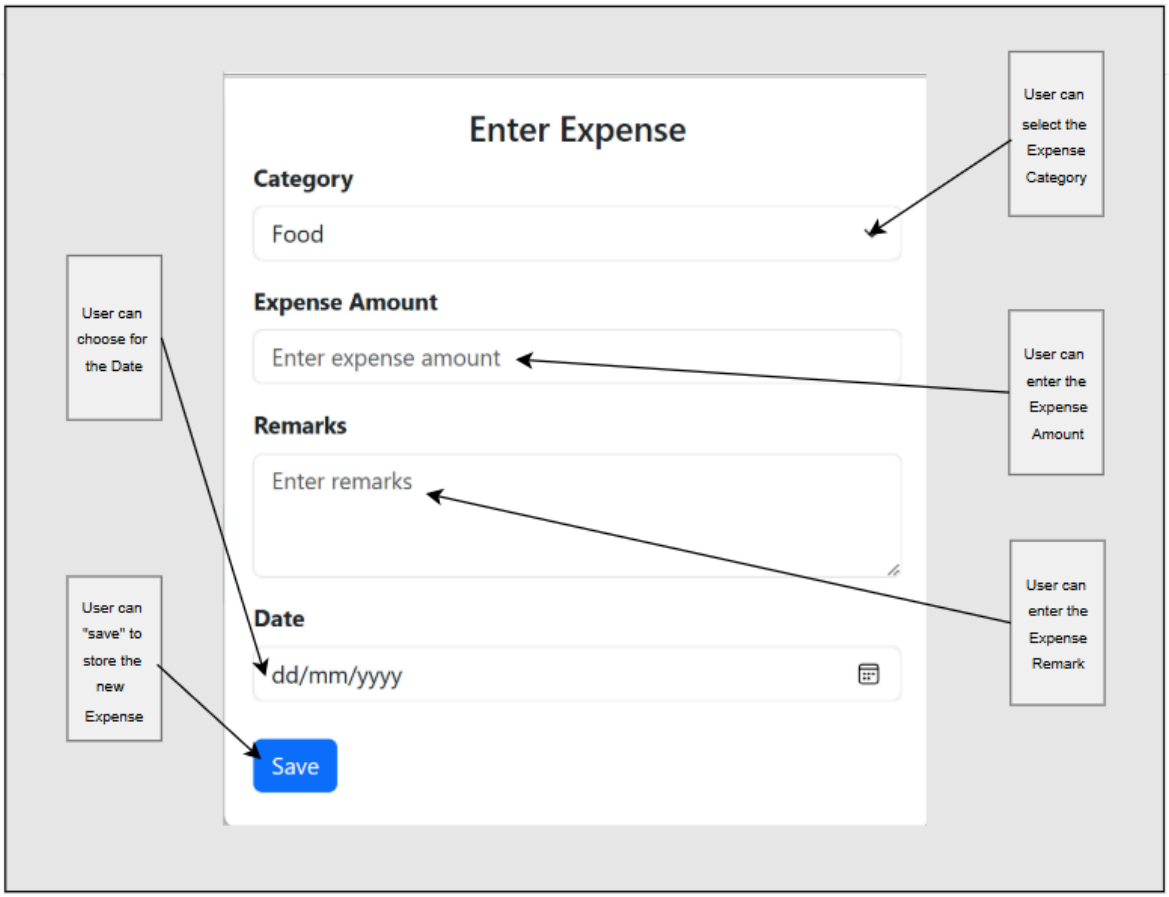
4.4.6 Add Income Page

The **Add Income Page** allows users to record new income transactions by providing several fields for input. Users can select an income **Category** from a dropdown menu, enter the **Income Amount**, and include any additional details in the **Remarks** section. They can also choose a specific **Date** for the income entry using a date picker. Once all the necessary information is filled out, users can click the **Save** button to store the income record in the system.



4.4.7 Add Expense Page

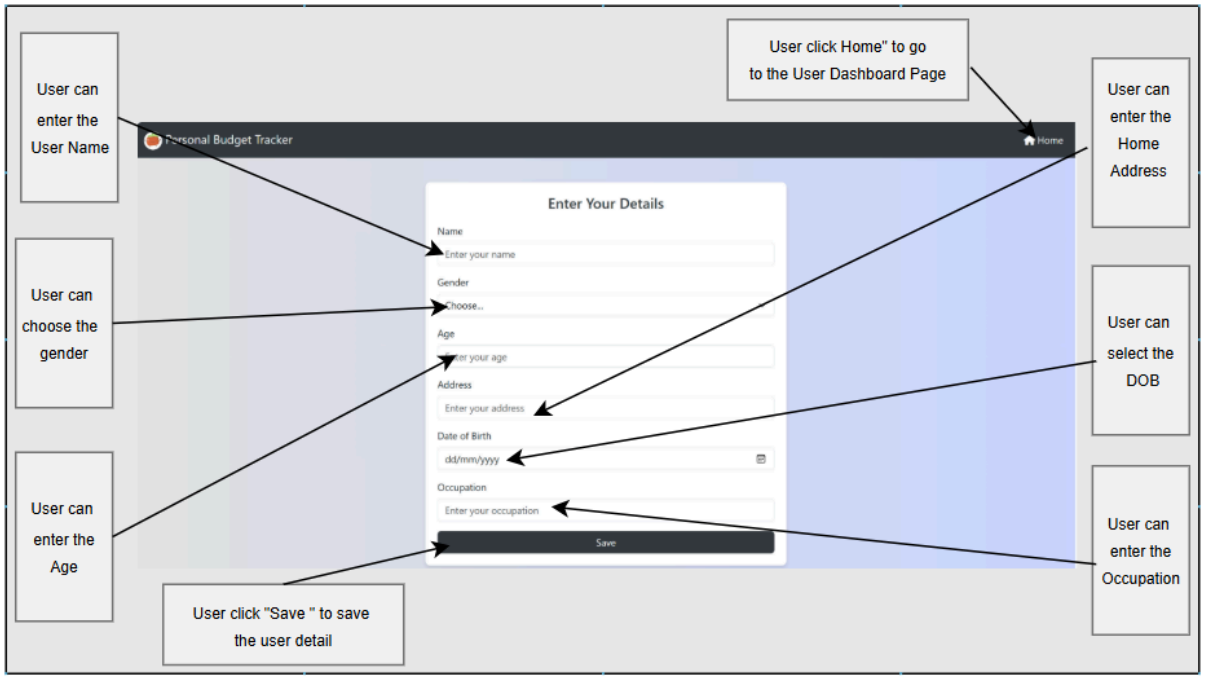
The **Add Expense Page** functions similarly, enabling users to log new expense transactions. Users can select an expense **Category**, input the **Expense Amount**, and add relevant details in the **Remarks** field. They can also specify a **Date** for the expense entry using the date picker. After entering the details, users can click the **Save** button to save the expense record. This page ensures accurate tracking of expenditures, complementing the income tracking functionality.



4.4.8 Profile Page

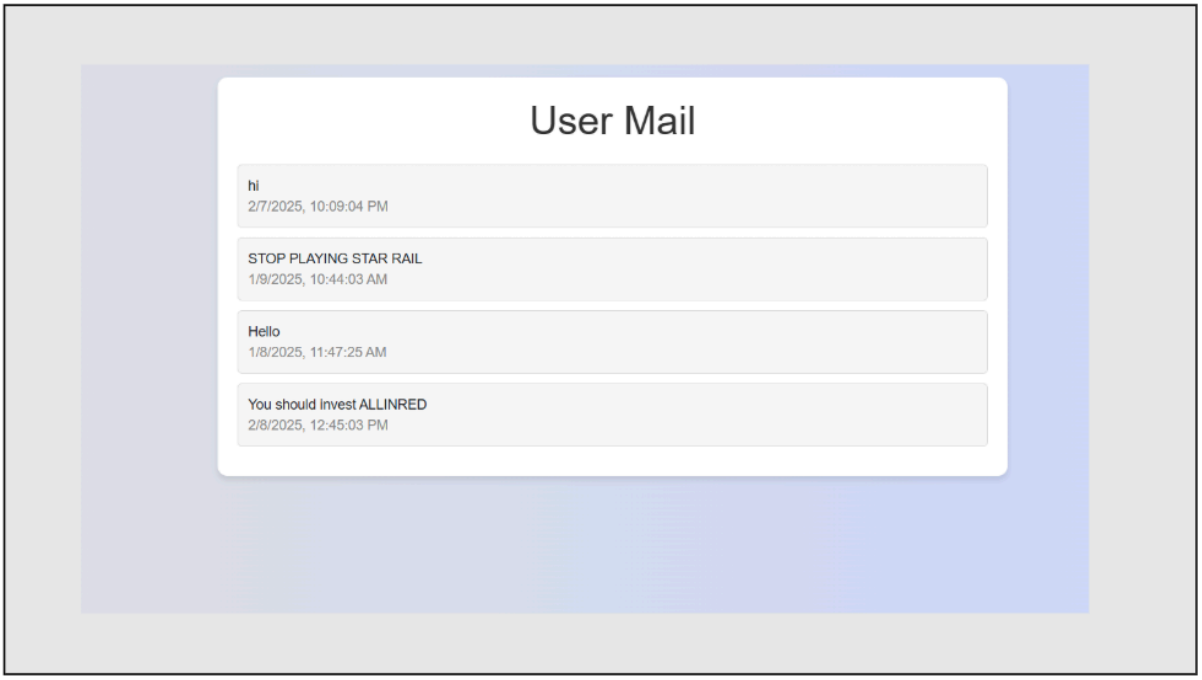
The **Profile Page** allows users to input and manage their personal details. Users can enter their **Name**, select their **Gender** from a dropdown menu, provide their **Age**, and input their **Home Address**. Additional fields include the ability to select the user's **Date of Birth (DOB)** using a date picker and input their **Occupation**.

Once all the required information is filled out, users can click the **Save** button to store their profile details. A **Home** button is available in the top-right corner, allowing users to return to the **User Dashboard Page** seamlessly. This page provides a comprehensive and user-friendly interface for managing personal information within the system.



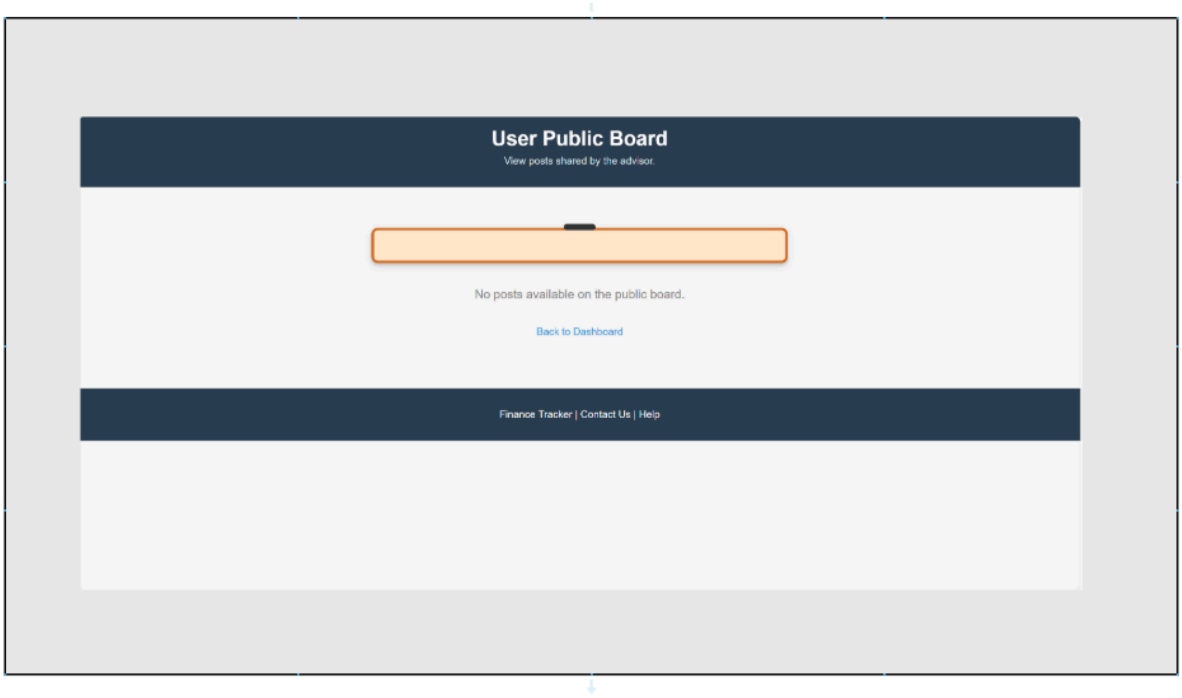
4.4.9 Email Page

The **Email Page** is designed to allow users to view messages sent by their financial advisor. This page will provide a secure and organized platform for users to access and manage their communication with their financial advisor, ensuring streamlined and effective communication.



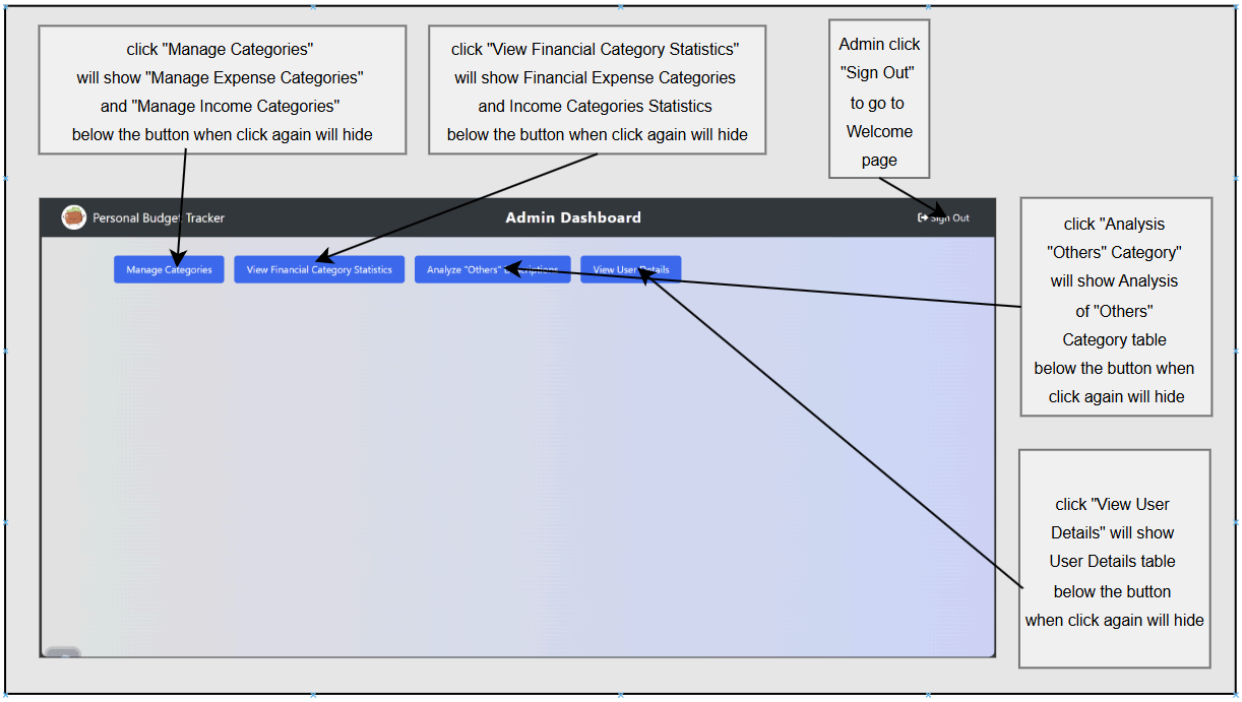
4.4.10 Public Board Page

The **Public Board Page** allows users to view posts shared by their financial advisor. When posts are added by the advisor, this page will display them in a structured and organized manner for the user to access. A **"Back to Dashboard"** link is provided for easy navigation to the main dashboard. Additionally, footer links such as **Finance Tracker**, **Contact Us**, and **Help** offer users quick access to related resources and support.



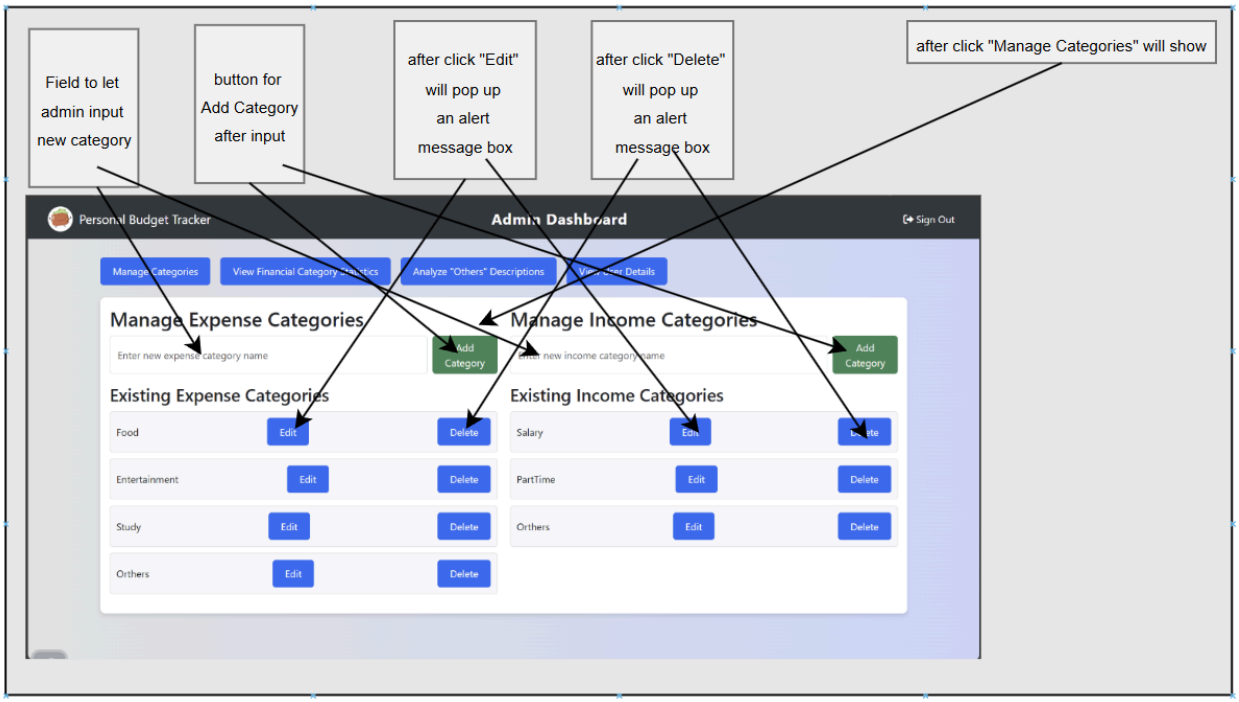
4.5 Subsystem Admin Screen

4.5.1 Main Page

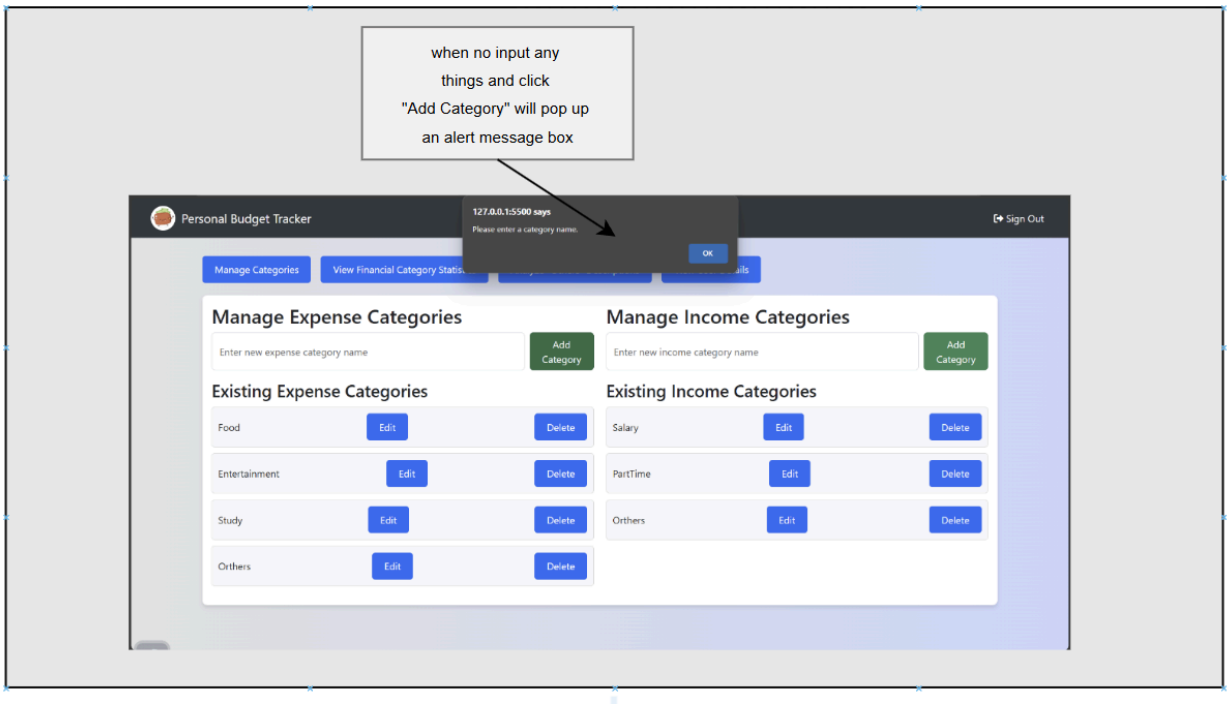


The **Admin Dashboard** of the Personal Budget Tracker provides essential tools for managing and analyzing financial data. It includes buttons like **Manage Categories** to handle expense and income categories, **View Financial Category Statistics** to display income and expense statistics, and **Analyze "Others" Category Description** for reviewing uncategorized data. Additionally, the **View User Details** button shows user information in a table format. All these buttons feature toggle functionality to show or hide their respective details. A **Sign Out** button in the top-right corner allows secure logout. The dashboard's clean layout ensures easy navigation and efficient management.

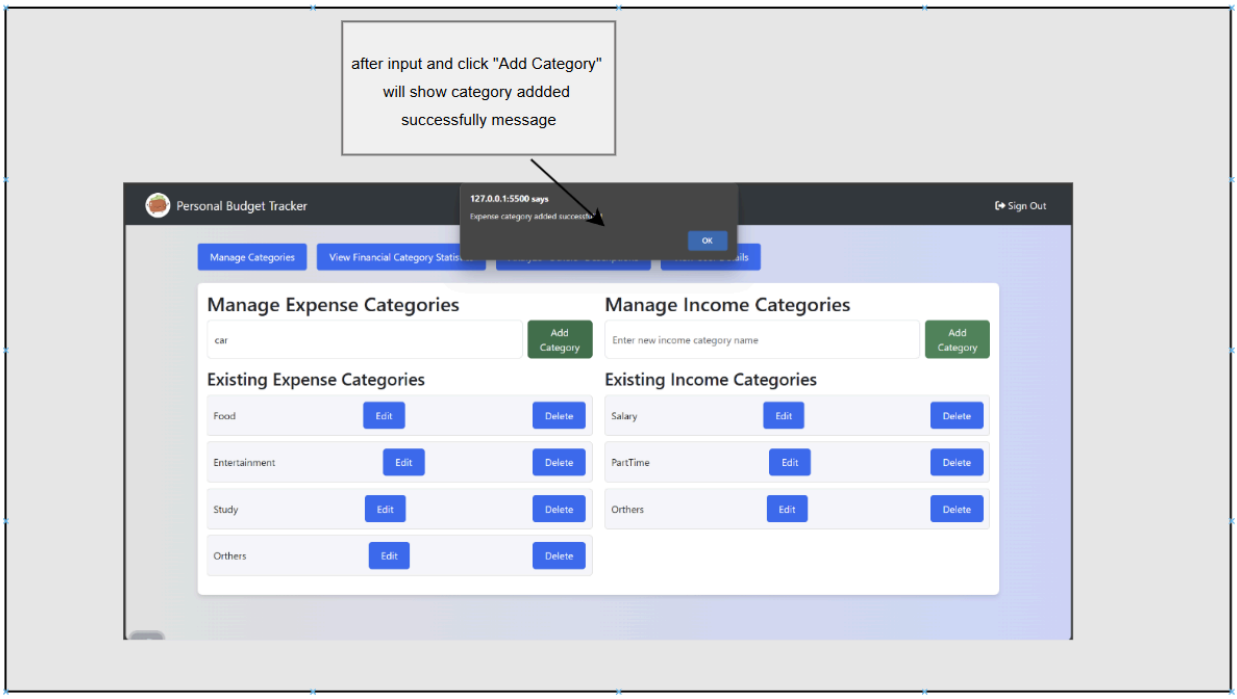
4.5.2 Manage Categories Page



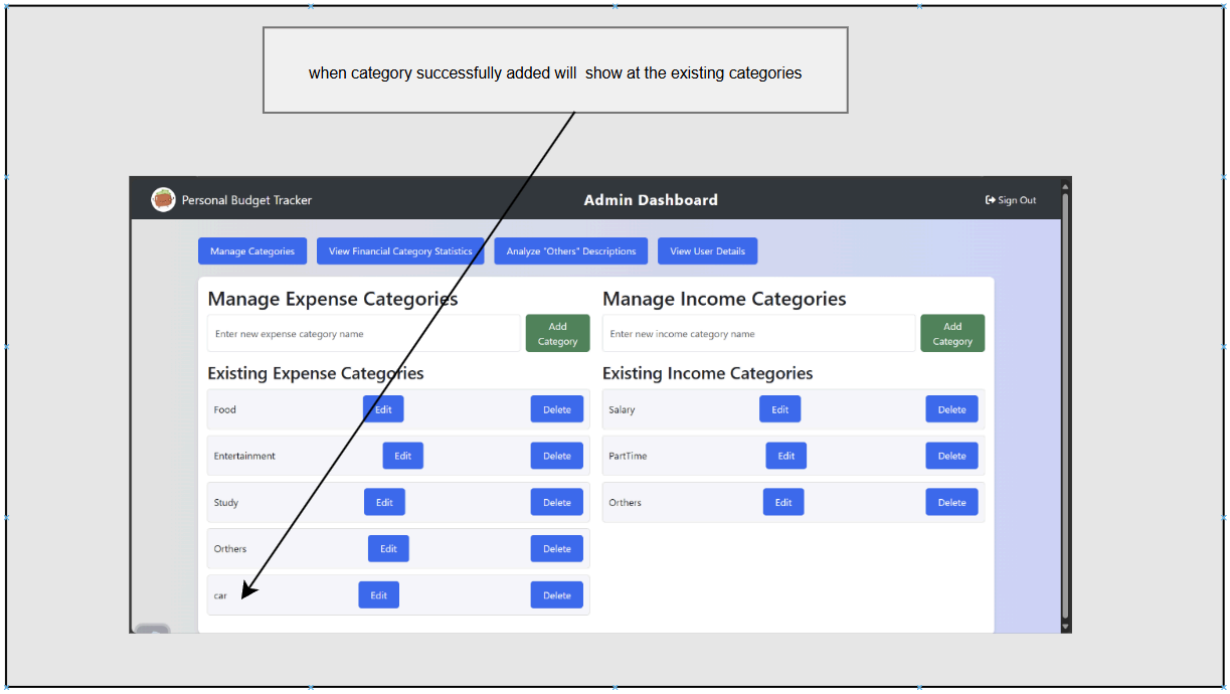
The **Manage Categories** section of the Admin Dashboard allows administrators to manage both expense and income categories. It includes input fields for adding new categories, accompanied by a green **Add Category** button to save the entries. Existing categories are listed with **Edit** and **Delete** buttons for modification or removal. Clicking the **Edit** or **Delete** buttons triggers an alert message for confirmation. This section streamlines the process of organizing financial data, ensuring that categories for income and expenses can be easily customized and maintained. The interface is simple and user-friendly, enhancing efficiency for administrators.



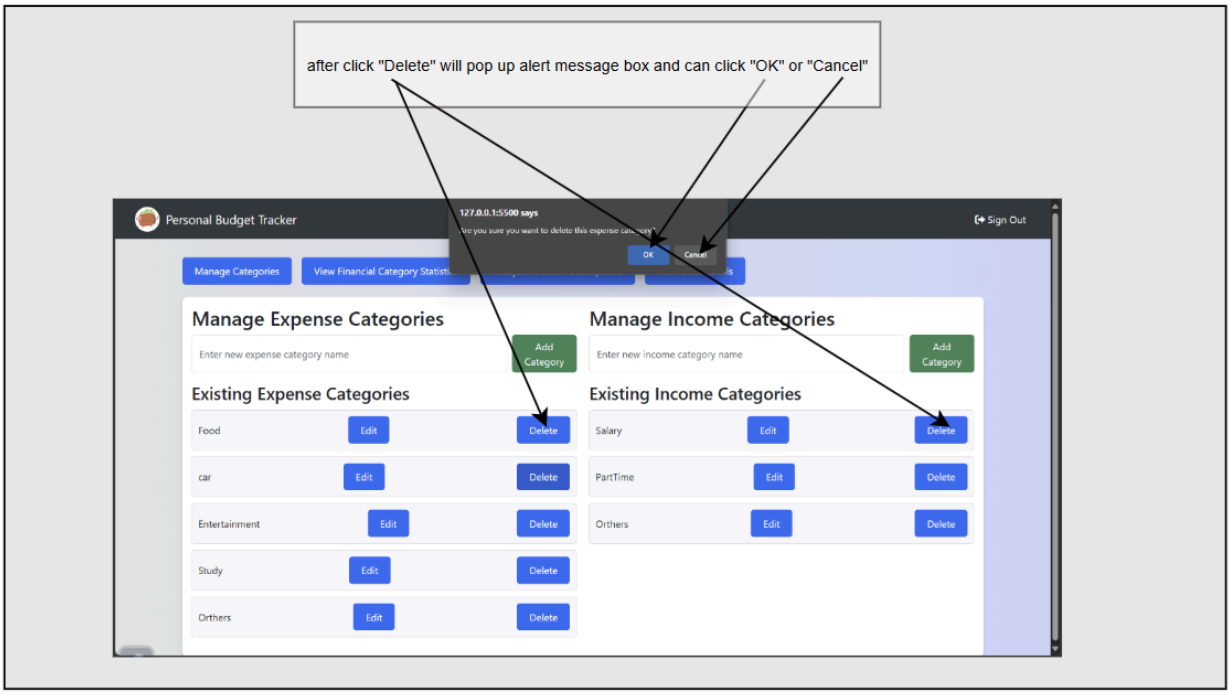
The **Manage Categories** section includes a validation feature for adding new categories. If the **Add Category** button is clicked without entering any input in the category name field, an alert message box appears with a prompt stating, "Please enter a category name." This ensures that users cannot add empty or invalid entries, maintaining the integrity of the category data.



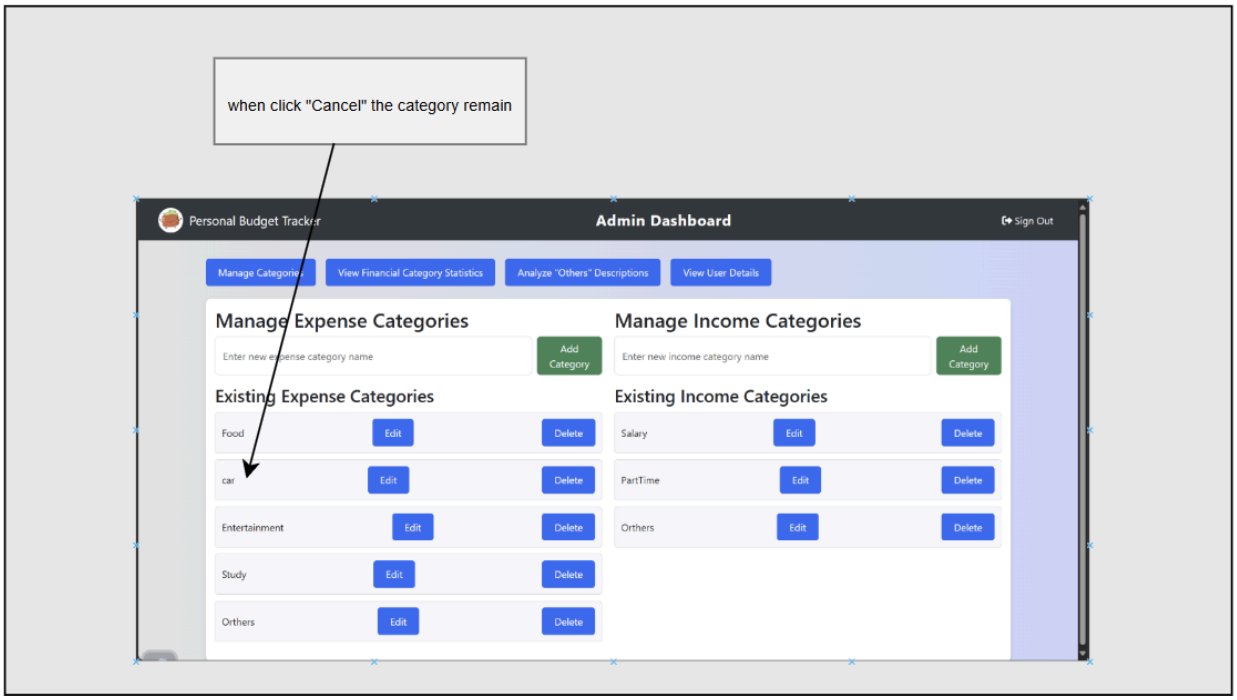
When a new category is successfully added in the **Manage Categories** section, an alert message box confirms the action with a message like "Expense category added successfully."



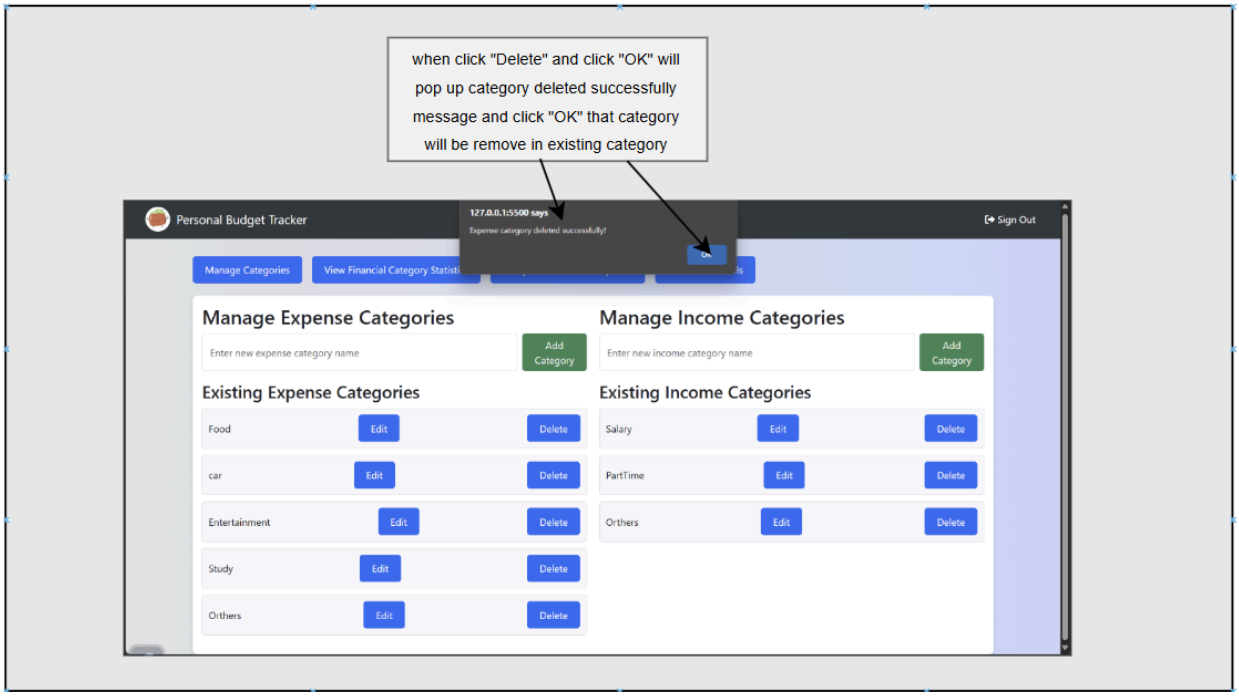
The newly added category is then displayed under the **Existing Categories** list, along with options to **Edit** or **Delete** it. This ensures that the admin can immediately see and manage the added category, maintaining a smooth workflow and clear organization of expense and income categories.



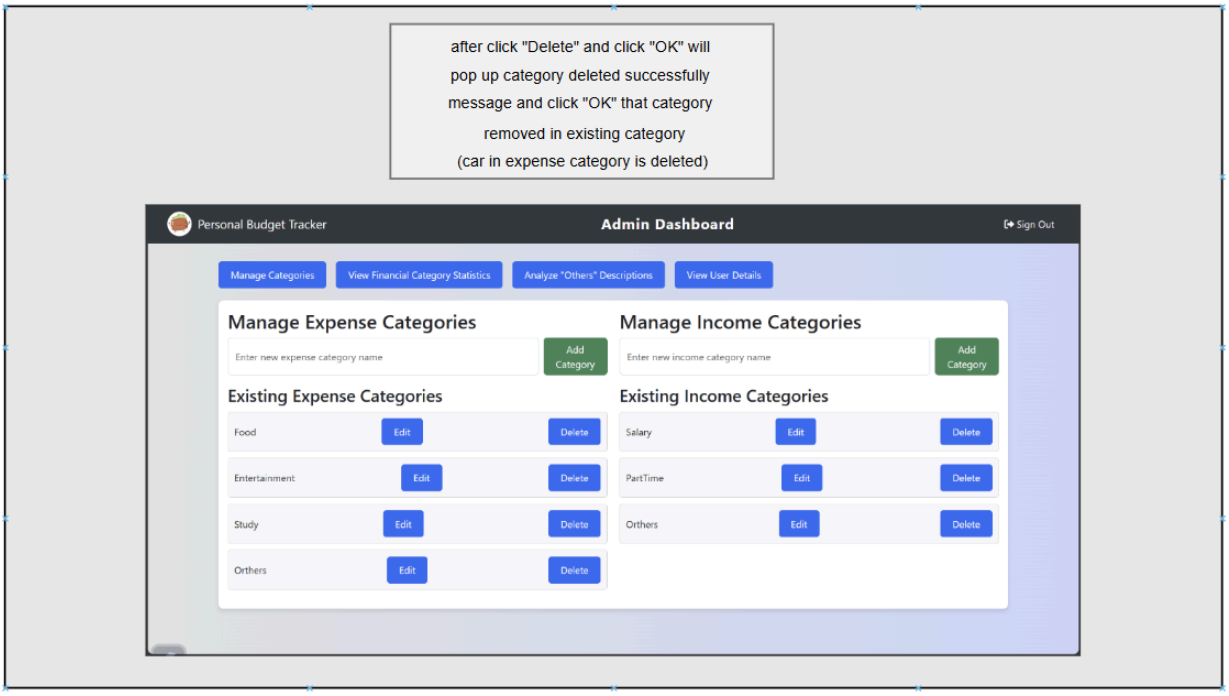
When the **Delete** button is clicked in the **Manage Categories** section, a confirmation alert message box appears, asking the admin to confirm the deletion by selecting either "OK" or "Cancel."



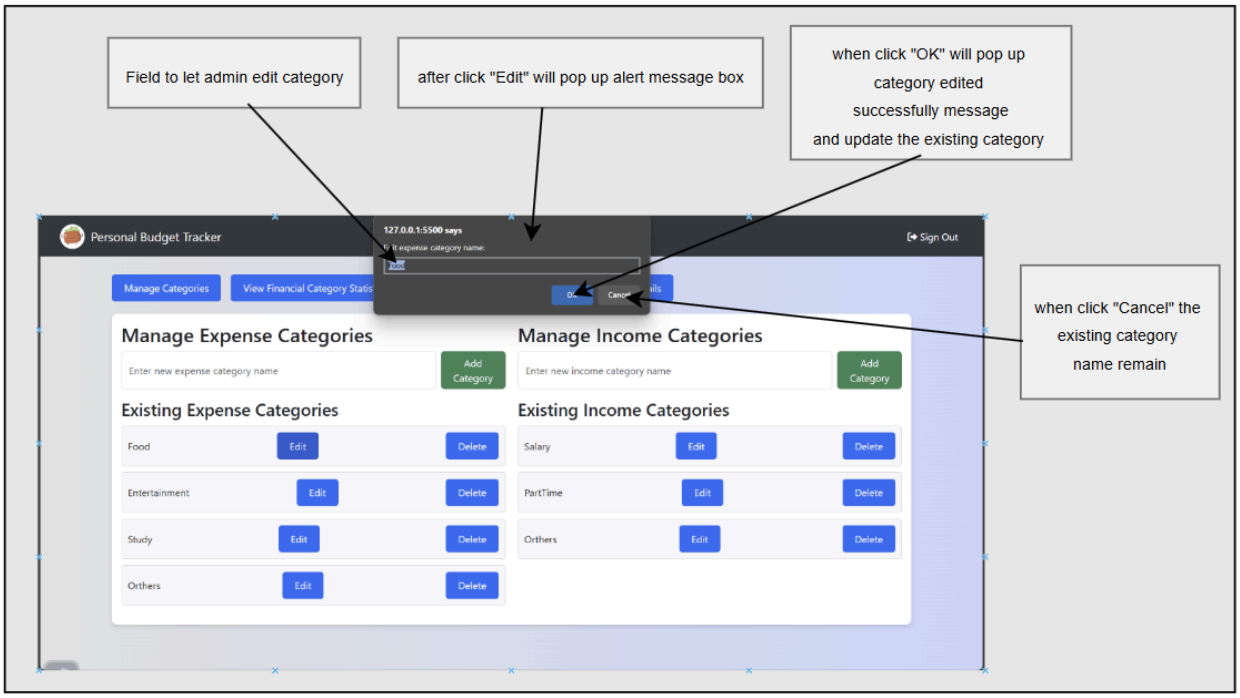
If "Cancel" is selected, the category remains unchanged.



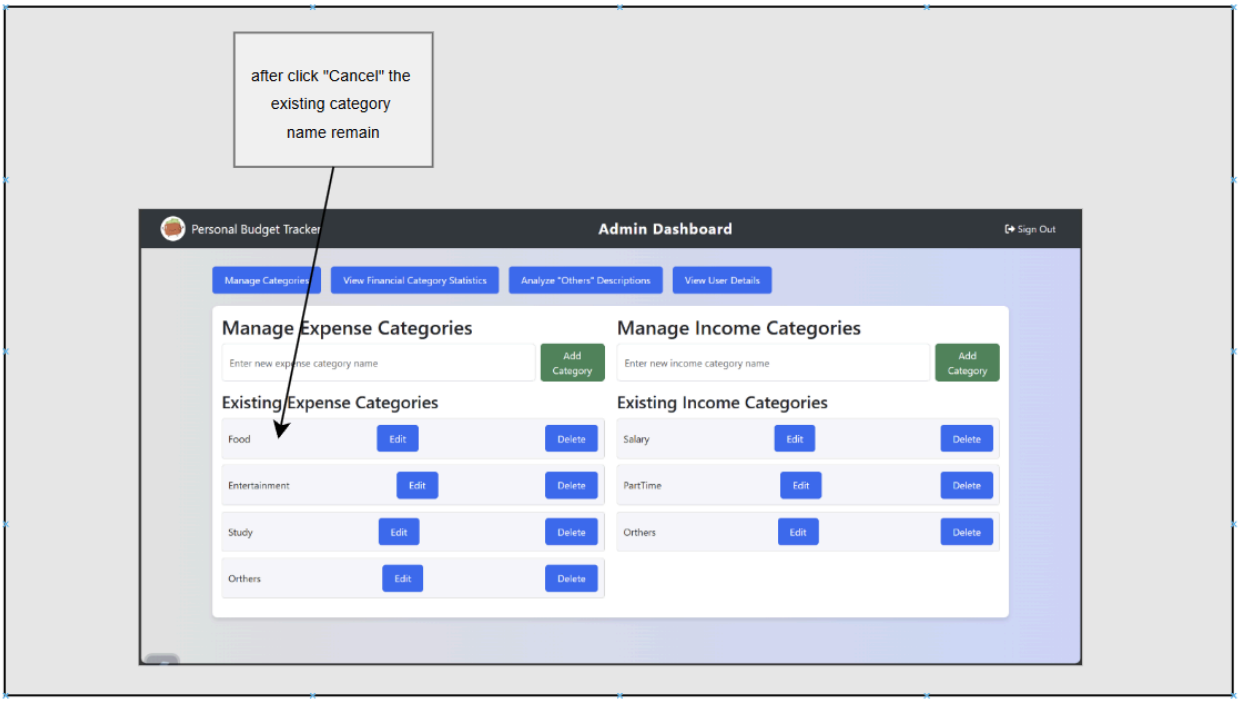
If "OK" is clicked, a success message confirms that the category has been deleted.



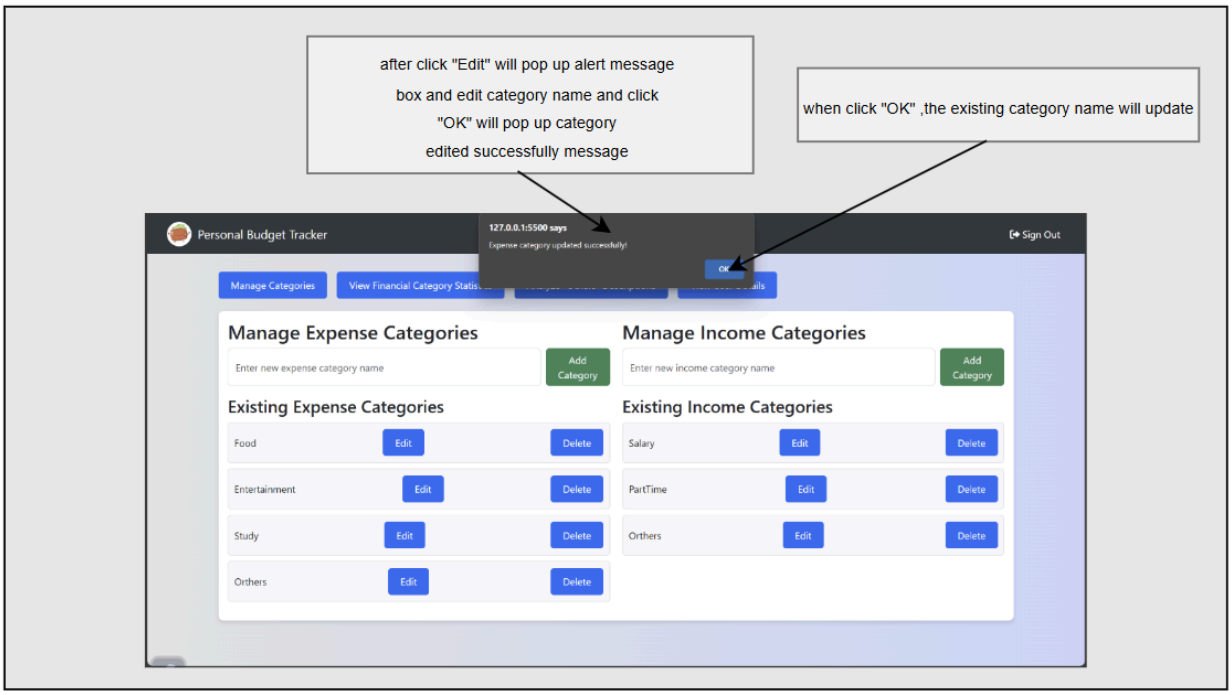
The selected category is removed from the **Existing Categories** list. This functionality ensures that administrators have control over category deletion while minimizing accidental changes through confirmation prompts.



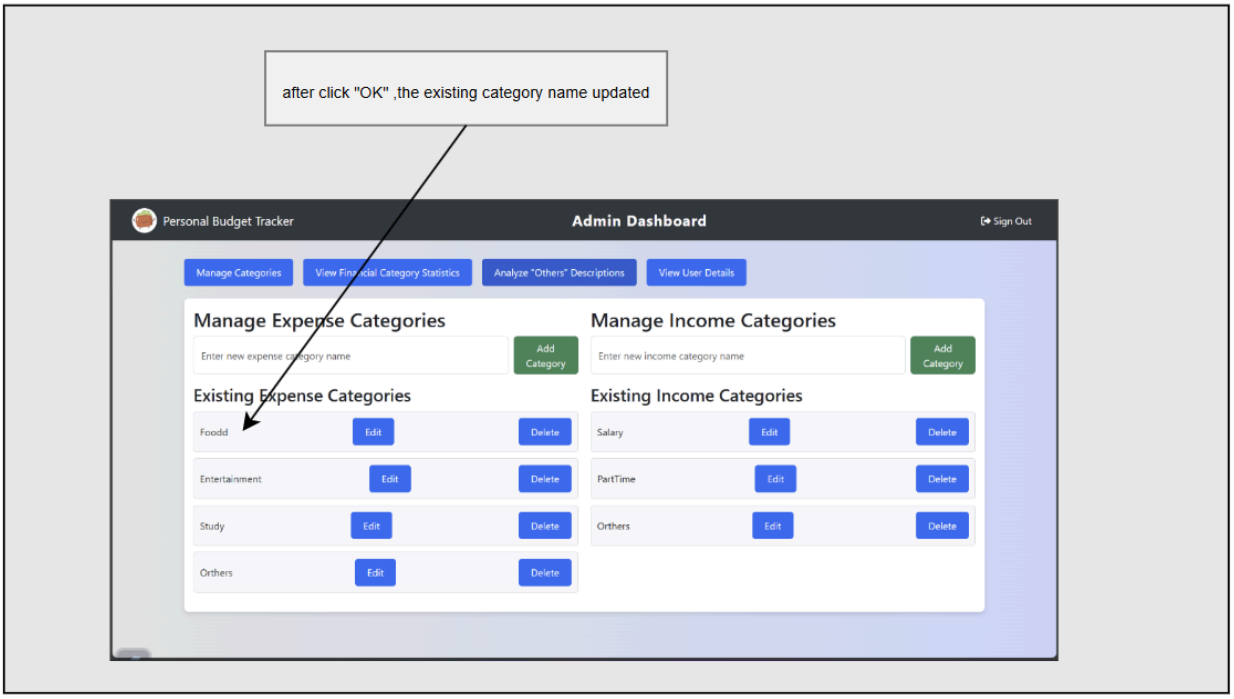
In the **Edit Category** process, clicking the **Edit** button opens an alert message box that allows the admin to input a new name for the selected category.



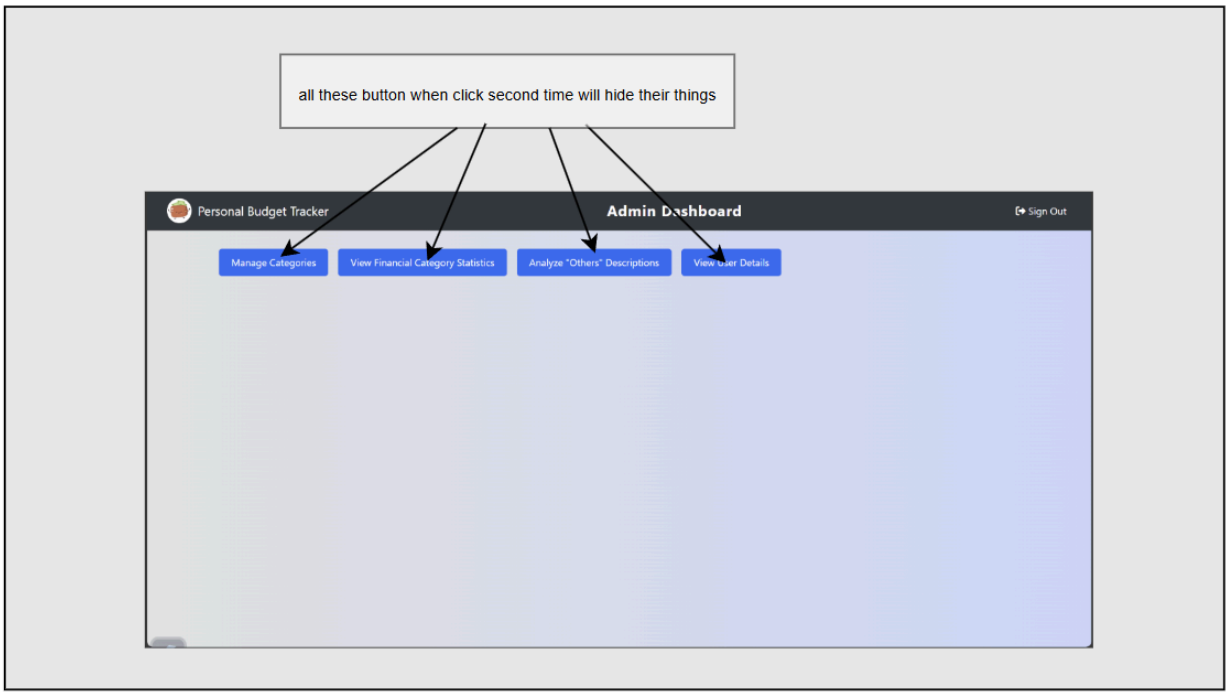
If the admin clicks **Cancel**, the category remains unchanged.



If the admin clicks **OK**, a success message confirms the update.



The category name is immediately updated in the **Existing Categories** list.This feature ensures flexibility and control in managing category names while providing clear confirmation for every action.

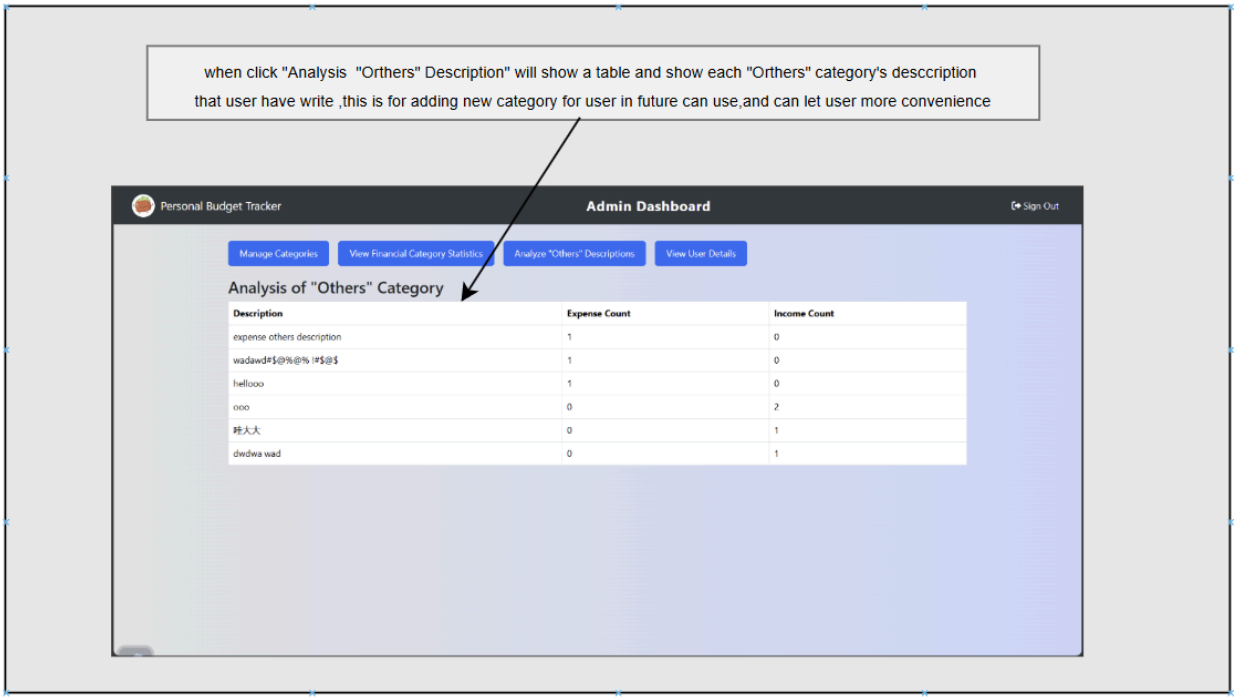


The buttons on the **Admin Dashboard** (such as **Manage Categories**, **View Financial Category Statistics**, **Analyze "Others" Descriptions**, and **View User Details**) are toggle buttons. When clicked the first time, they reveal their respective content or functionality. Clicking them a second time hides the displayed content, allowing for a clean and uncluttered interface. This toggle functionality ensures better usability and efficient navigation for administrators.

4.5.3 View Financial Category Statistics Page

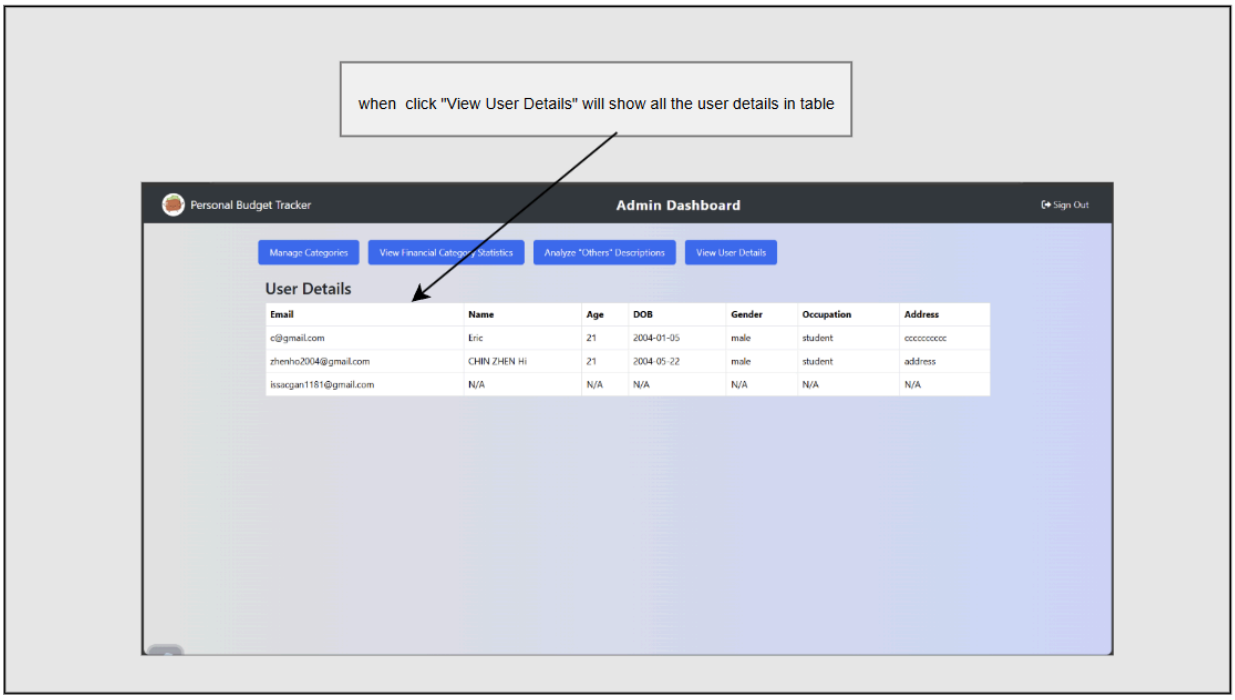
When the **View Financial Category Statistics** button is clicked, two pie charts are displayed, representing **Expense Categories** and **Income Categories**. Each chart visually breaks down the categories, such as food, study, salary, and others. Hovering over a category in the chart reveals a tooltip showing how many users have selected that category. This feature provides an intuitive and interactive way for administrators to analyze category usage statistics.

4.5.4 Analyze "Others" Descriptions Page



When the **Analyze "Others" Descriptions** button is clicked, a table appears displaying the descriptions under the "Others" category along with their corresponding **Expense Count** and **Income Count**. This feature allows administrators to review user-provided descriptions, helping them identify patterns or common entries that could be used to create new categories. It enhances user convenience by refining the category system based on actual usage and feedback.

4.5.5 View User Details Page

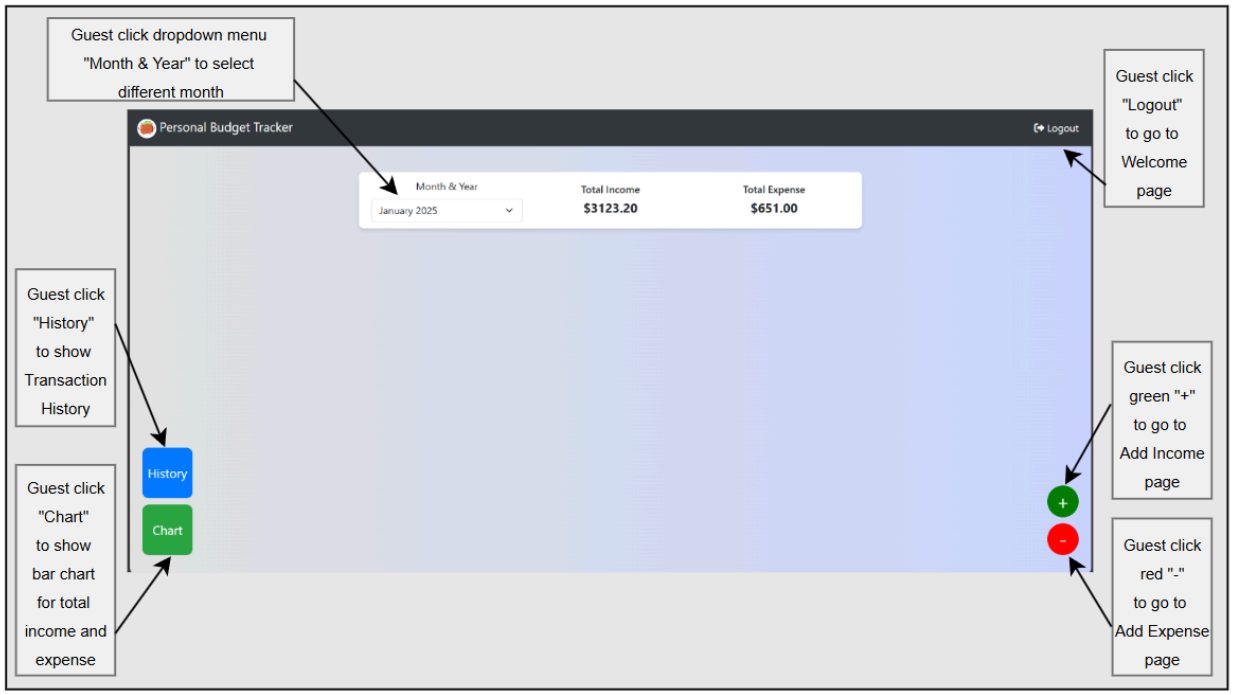


When the **View User Details** button is clicked, a table is displayed showing all user information, including fields such as **Email**, **Name**, **Age**, **Date of Birth (DOB)**, **Gender**, **Occupation**, and **Address**. This feature allows administrators to view and manage user data effectively, providing a clear overview of user details within the system.

4.6 Subsystem Guest Screen

4.6.1 Main Page (Guest Dashboard)

The Guest Dashboard offers a simplified interface for guest users to view and manage their financial data temporarily. Guests can select a specific month and year using the dropdown menu and view summarized totals for income and expenses. Navigation options include the "History" button to access the Transaction History and the "Chart" button to view income and expense bar charts. Guests can add new entries using the green "+" and red "-" buttons to record income and expenses, respectively, and a "Logout" button allows easy exit to the Welcome Page.

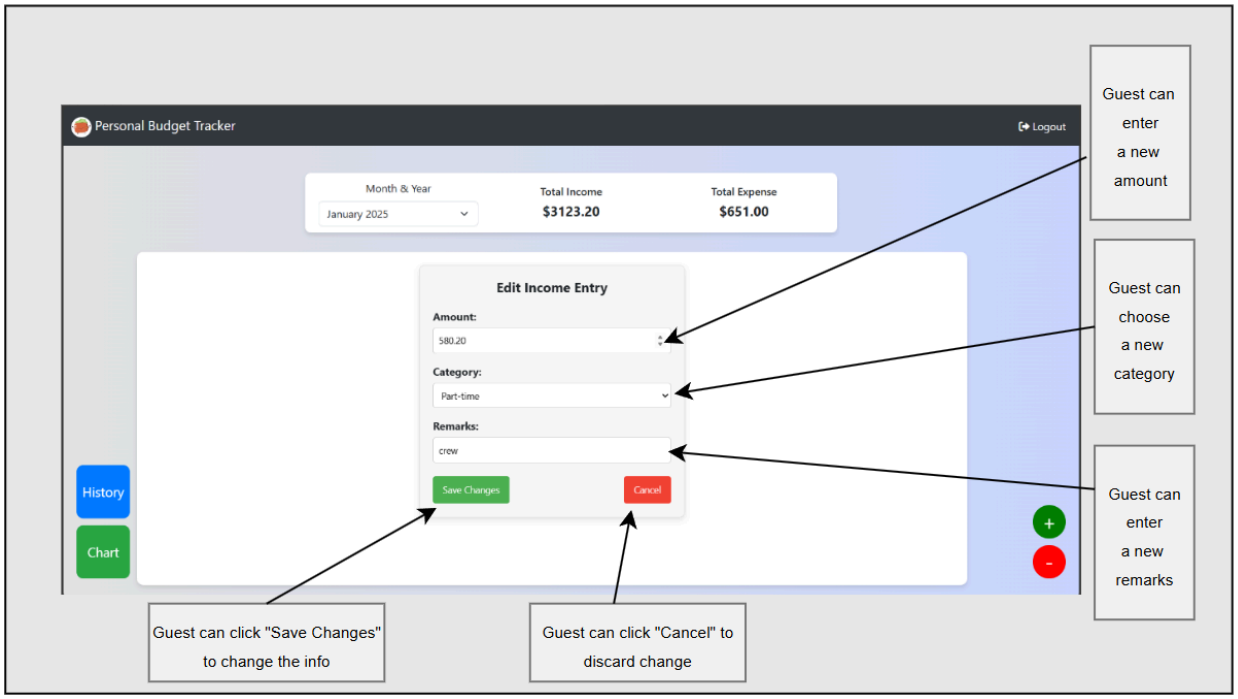


4.6.2 Transaction History Page (GUEST)

The Transaction History page (GUEST) displays a detailed list of temporary income and expense entries for the selected month. Each entry includes fields for type, date, amount, category, and remarks. Guests can edit entries using the edit icon, which redirects them to the Edit History Page, or delete specific entries using the delete icon, ensuring easy and efficient record management.

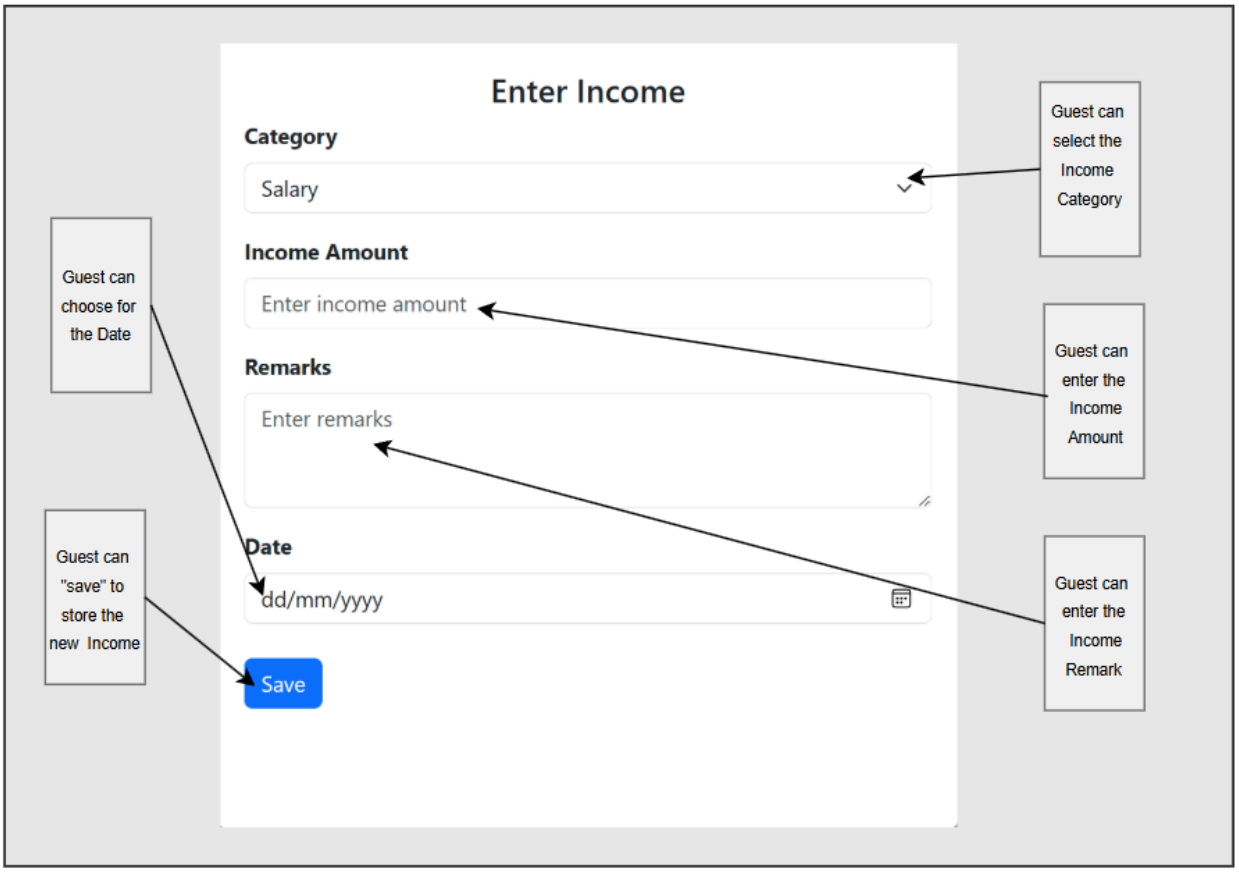
4.6.3 Edit History Page (GUEST)

The Edit History Page (GUEST) allows guests to modify existing income/expense records. Guests can update fields such as amount, category, and remarks, and then save the changes using the “Save Changes” button. Alternatively, they can discard the updates by clicking the “Cancel” button. This feature provides flexibility for guests to adjust temporary data during their session.



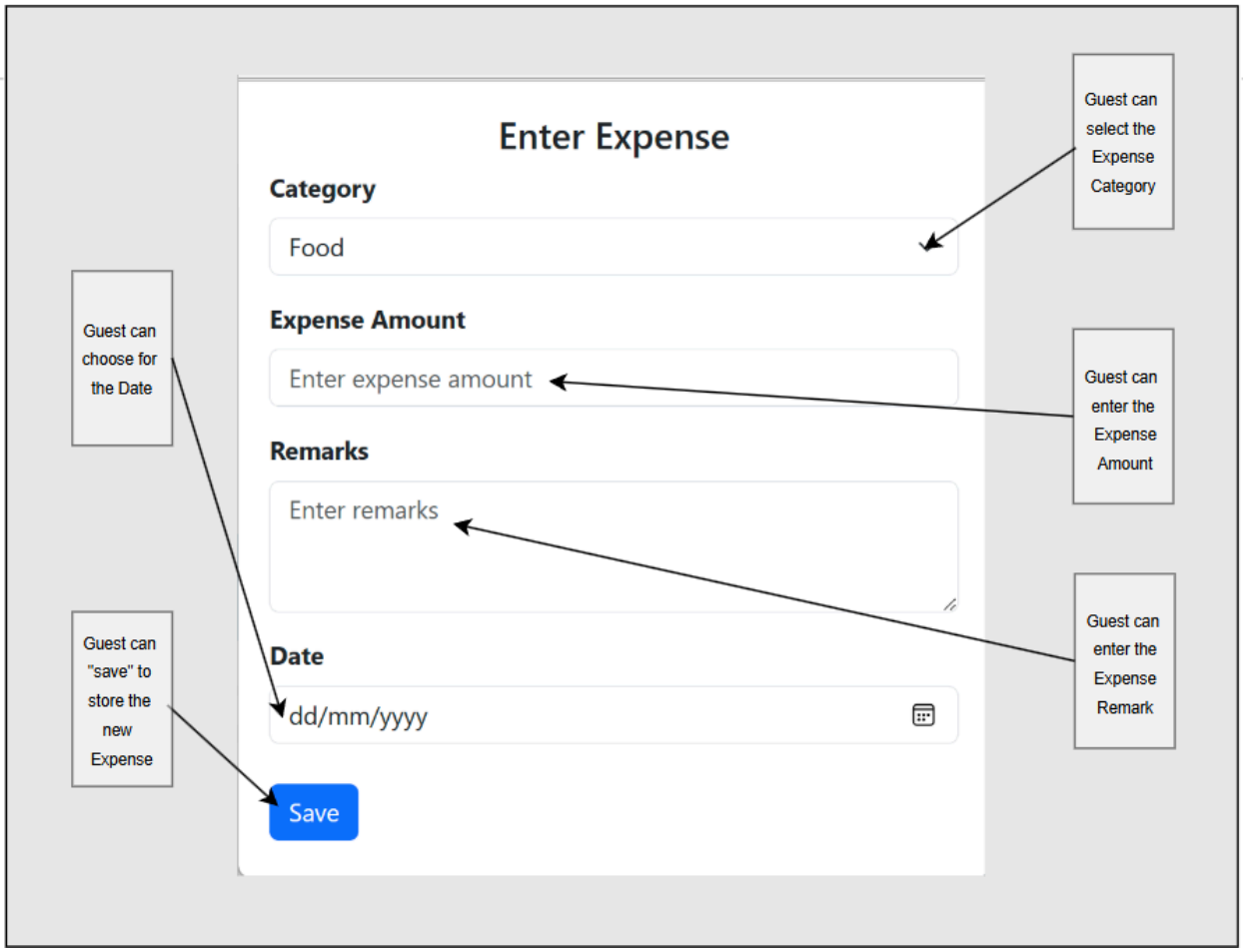
4.6.4 Add Income Page (GUEST)

The Add Income Page (GUEST) allows guests to temporarily record new income entries by selecting a category, entering the income amount, providing additional remarks, and specifying the date. Once all details are entered, guests can save the entry using the Save button, making it accessible for the duration of their session.



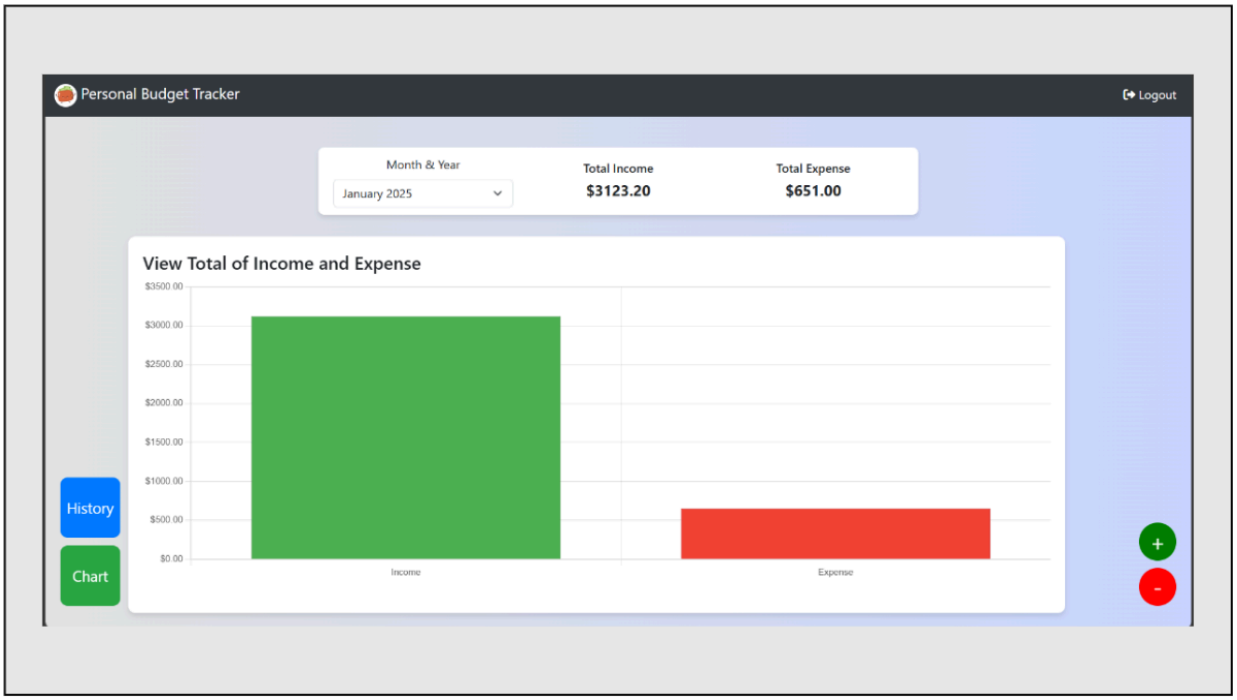
4.6.5 Add Expense Page (GUEST)

The Add Expense Page (GUEST) enables guests to record temporary expense entries. Users can input details such as category, expense amount, remarks, and date, and then save the entry using the Save button. This ensures that guests can manage their expense records conveniently during their session.



4.6.6 View Bar Chart Total Income And Expense Page (GUEST)

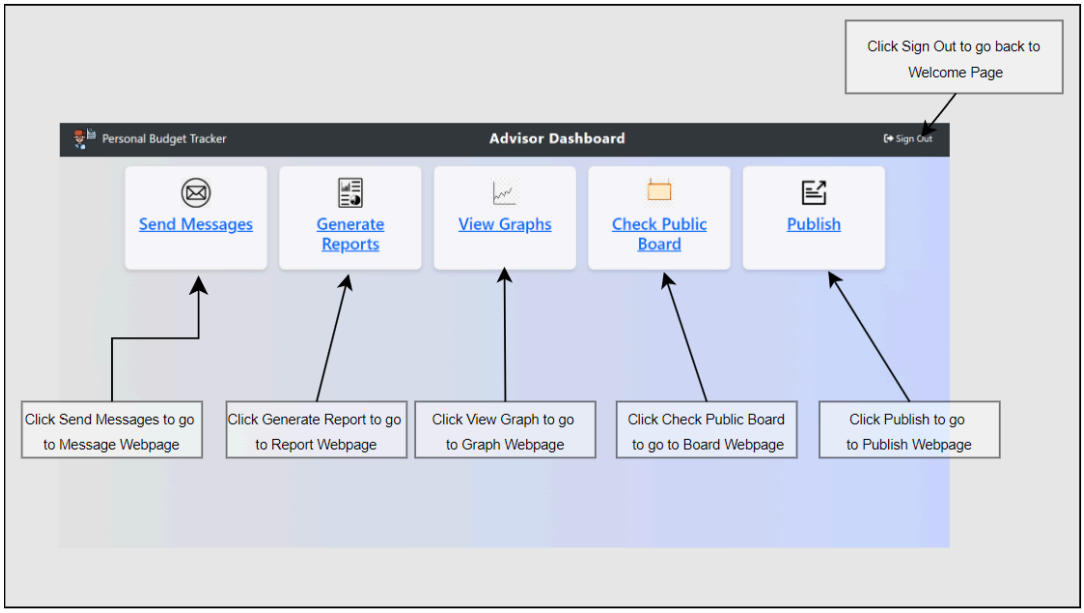
The View Bar Chart Total Income and Expense Page (GUEST) provides a visual summary of the guest's financial data. A bar chart compares total income and expenses, offering clear insights into the guest's financial balance for the selected month. This feature helps guests analyze their temporary financial records effectively.



4.7 Subsystem Financial Adviser

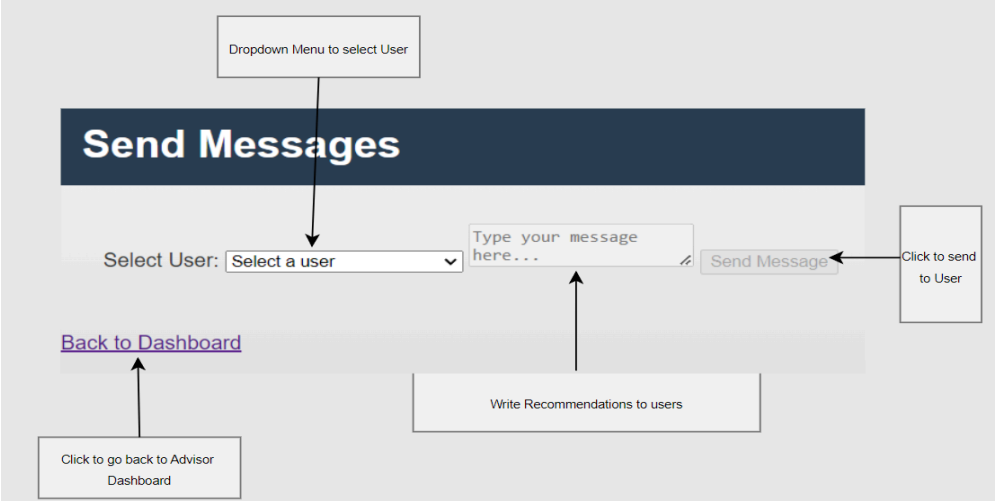
4.7.1 Advisor Dashboard

This page serves as the main homepage for advisors and is where they will be redirected to different web pages based on what they want to do, be it sending messages to users to try to analyze data using reports or graphs and lastly to share something to every user for them to see.



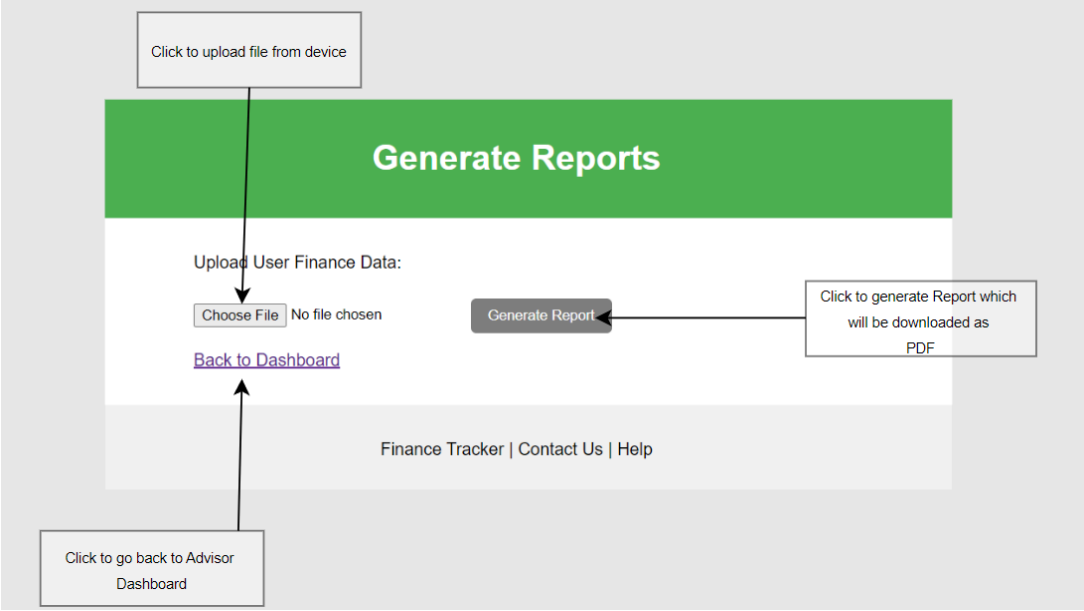
4.7.2 Message webpage

This is the page where advisors if they want to send personal messages to users of their choice then they will use this, advisors will be able to see the list from the drop-down menu which is connect to the database based on the number of accounts created, and then write their recommendation or message and then send to the user.



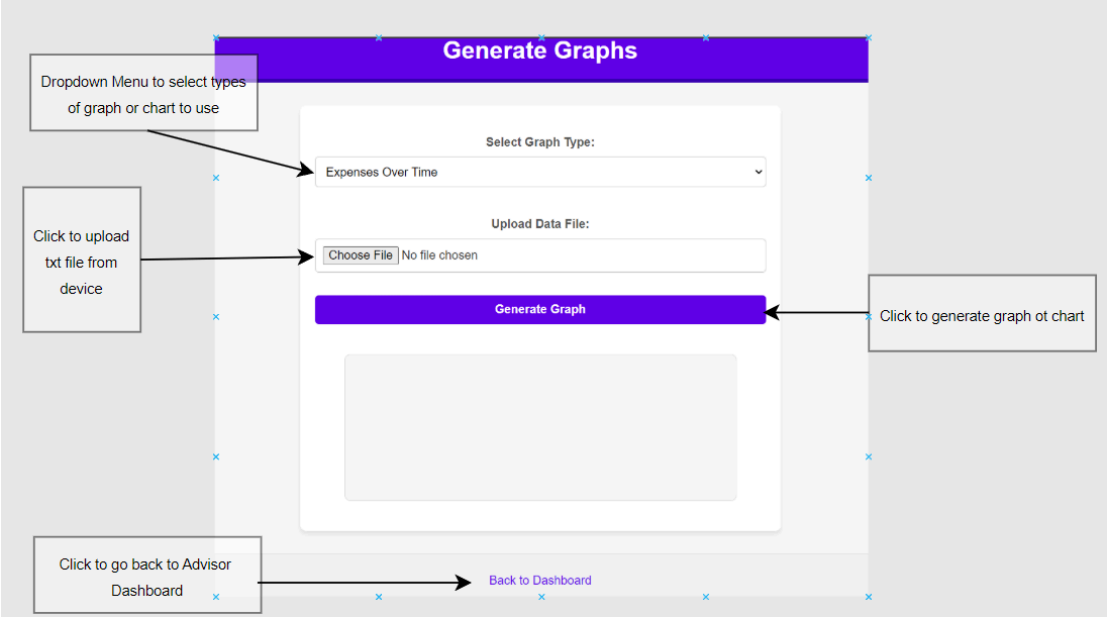
4.7.3 Generate Report

This website is designed for generating financial reports by allowing users to upload their finance data, generate a downloadable PDF report, and navigate back to the advisor dashboard via the provided link.



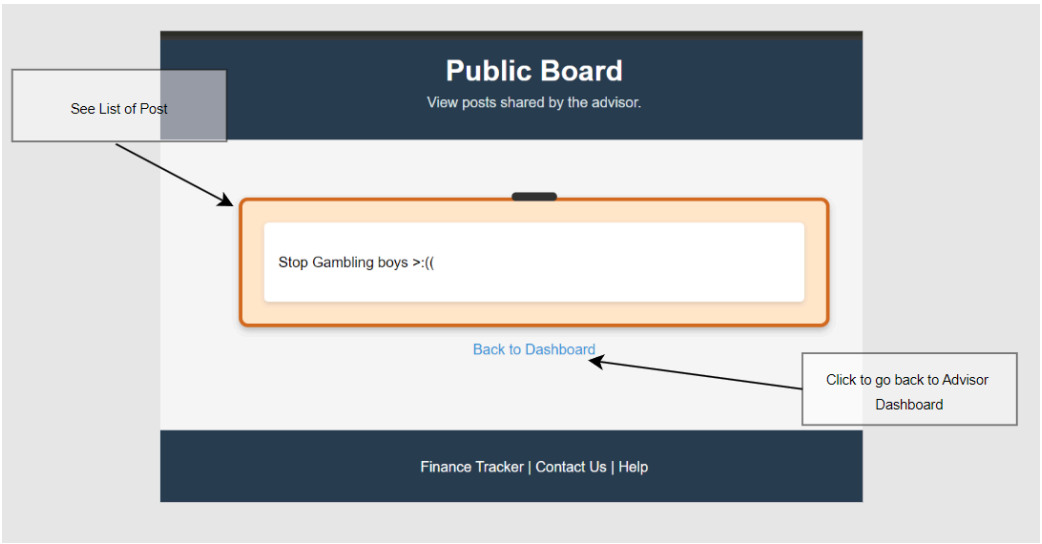
4.7.4 Generate Graph

This website allows users to generate graphs by selecting a graph type from a dropdown menu, uploading a data file, and clicking the button to create the desired graph, with an option to navigate back to the advisor dashboard.



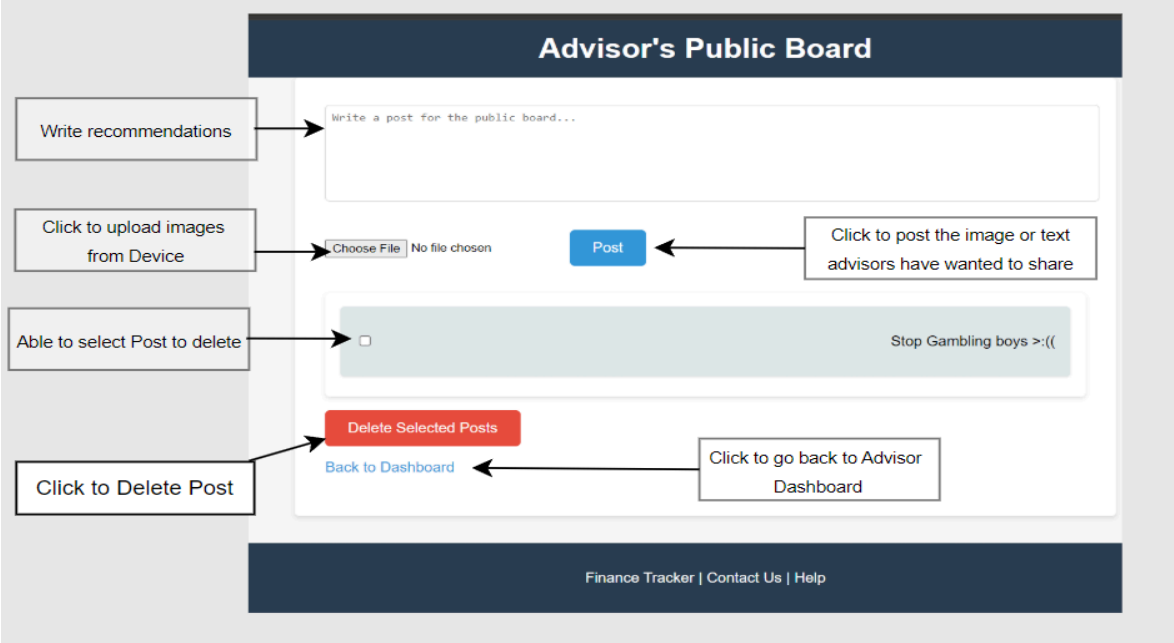
4.7.5 Public Board

This website serves as a public board where users can view posts shared by the advisor, with an option to navigate back to the advisor dashboard via the provided link.



4.7.6 Publish

This website allows advisors to share what they think is good to all users by either writing the message or uploading pictures and then if advisors feel like the post they made is not good, then they have the option to select from the list of posts and delete them.



4.8 Main Components

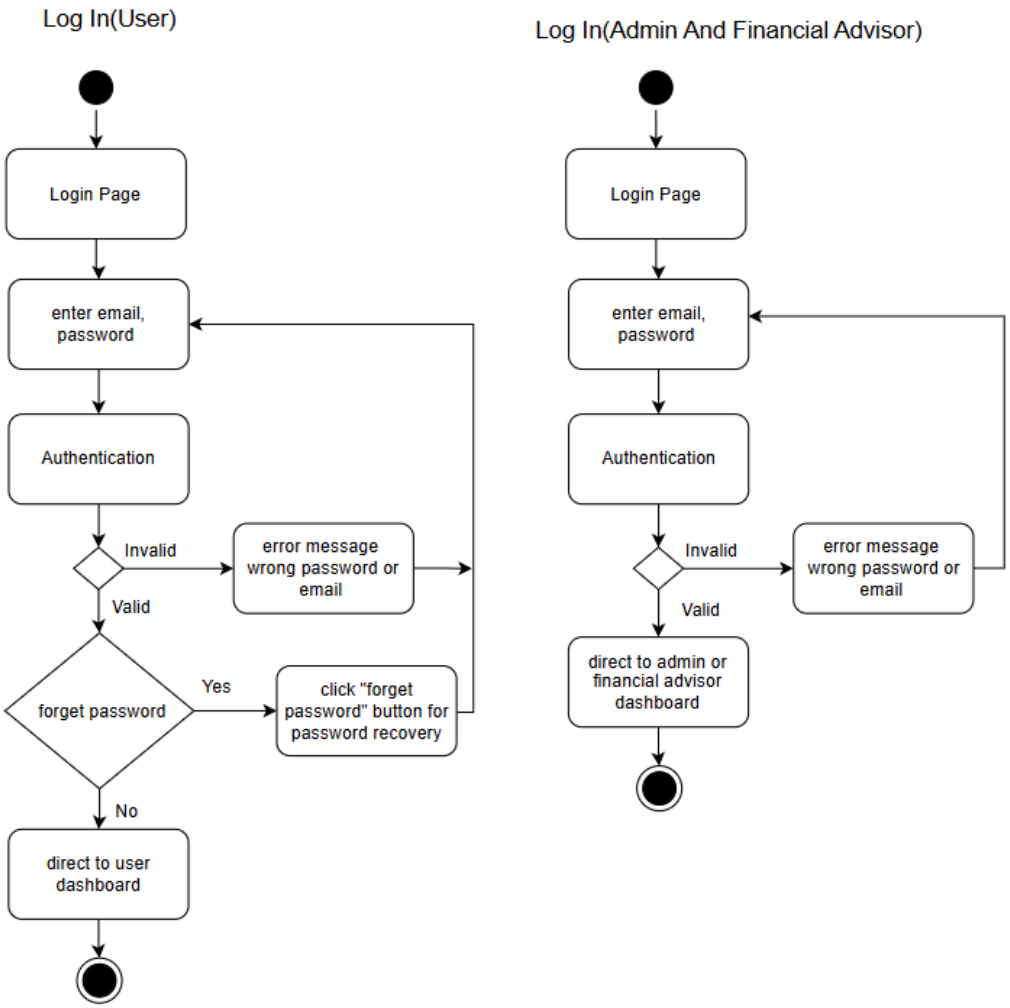
<TO DO: Describe the main components (modules, classes, packages, etc.) and the table with the components and related subsystems here.>

Actor	Component
All Actor	Login
User	View Transaction History
	View Total Income And Expense
	Set Budget
	Add Income
	Add Expense
	Edit Profile
	View Email From Advisor
	View Public Board
Admin	Manage Categories
	View Financial Category Statistics
	Analysis "Others" Descriptions
	View User Details
Guest	Sign Up
	Log In as Guest
	View Transaction History (GUEST)
	Add Income (GUEST)
	Add Expense (GUEST)
	Total Income and Expenses (GUEST)
Financial Advisor	Send Message
	Generate Report
	Generate Graph/Chart

	Public Board
	Publish

4.8.1 Login

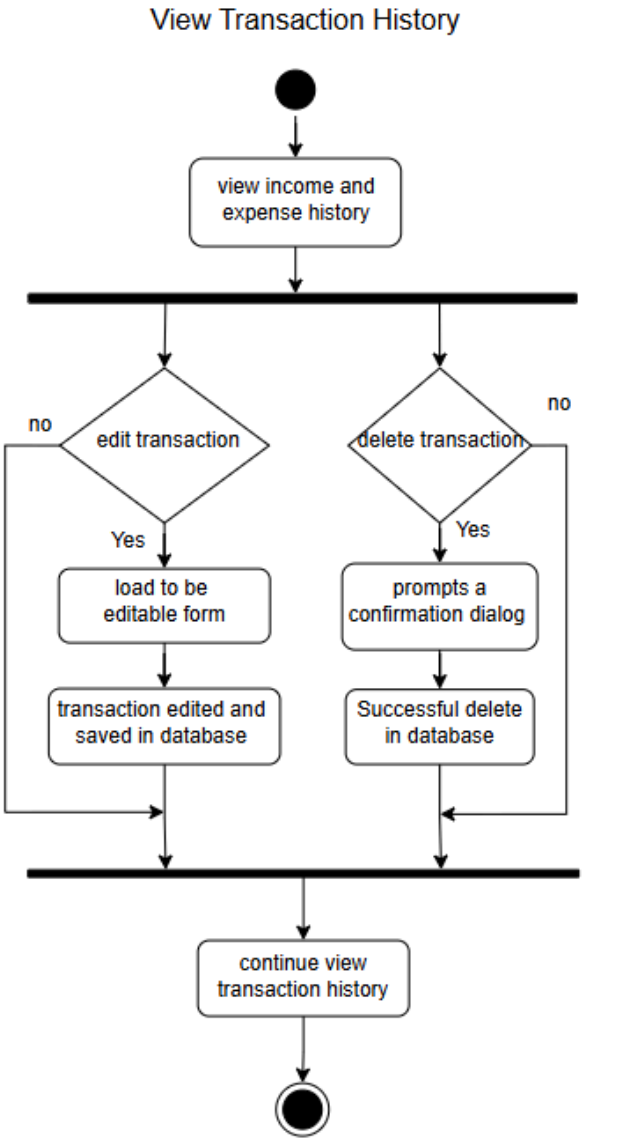
When a user or admin or financial advisor wants to login, they will redirect to the login page. They need to enter their email and password. Then, the system will authenticate if the email and password are valid. If the email and password are valid, then it is considered as successful login.If not, then vice versa.If user forgot password user can click the Recovery Password to reset the password.



4.8.2 User Component

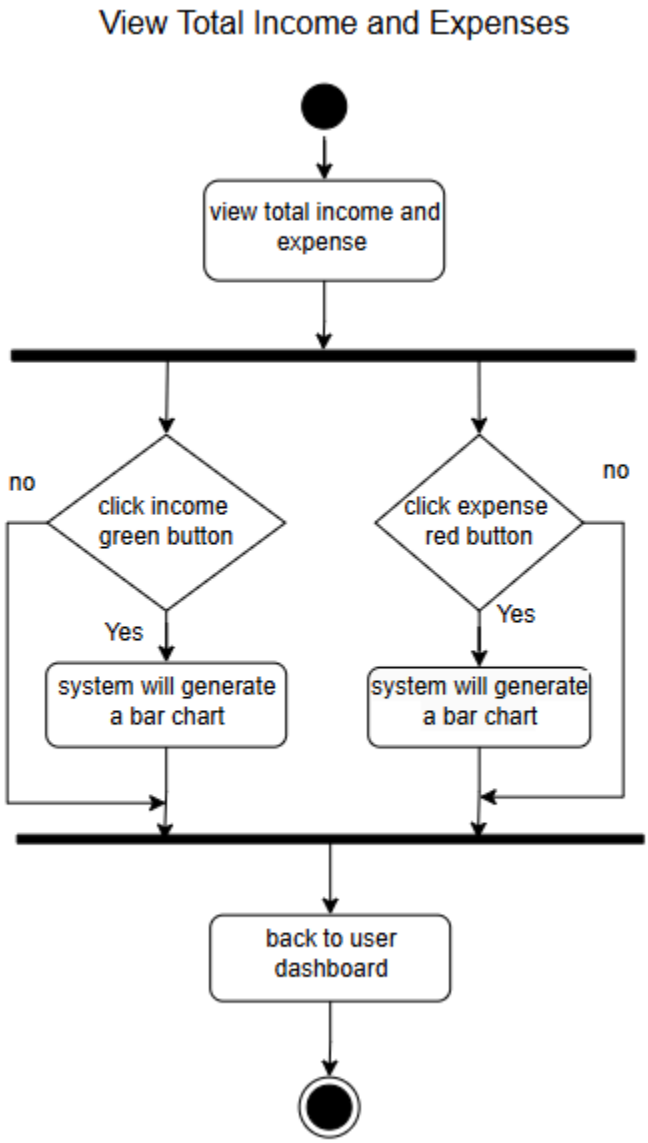
4.8.2.1 Viewing Transaction History

The user begins by viewing their income and expense history. They can choose to **edit a transaction** or **delete a transaction**. If the user chooses to edit, the selected transaction is loaded into an editable form, where changes are made and saved back to the database. If the user chooses to delete, a confirmation dialog appears; upon confirmation, the transaction is successfully deleted from the database. In both cases, the user returns to continue viewing the transaction history. If no action is taken, the user remains on the transaction history page.



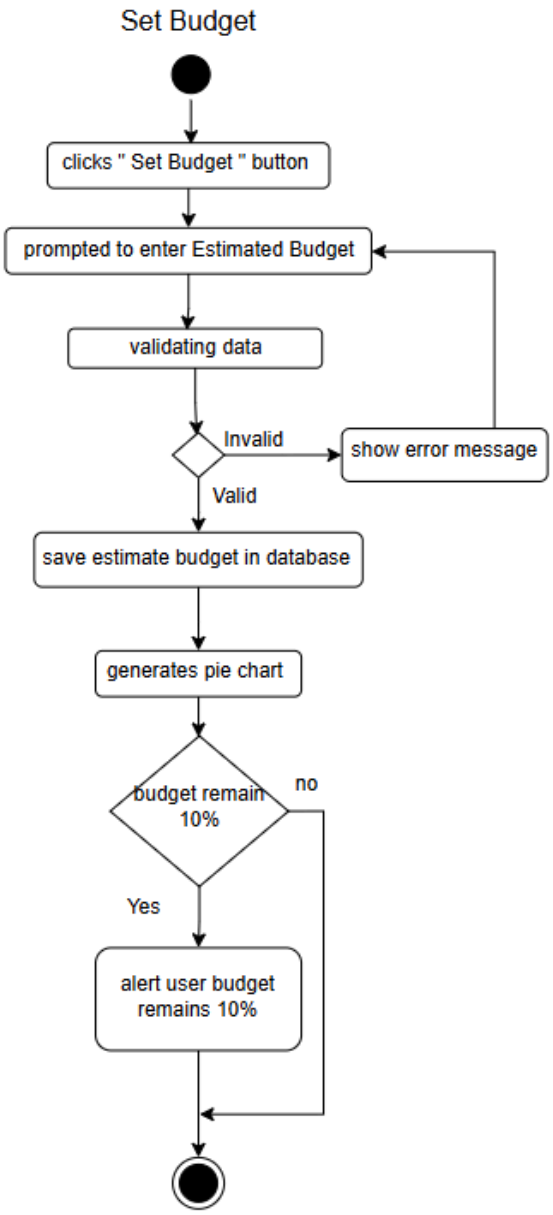
4.8.2.2 View Total Income And Expense

The user starts by navigating to the **View Total Income and Expense** page. They can choose to click the **income (green " + ") button** to generate a bar chart of total income or click the **expense (red " - ") button** to generate a bar chart of total expenses. If no action is taken, the user remains on the page. After generating the desired bar chart, the user can return to the **User Dashboard** to continue their activities. This process provides a simple way to visualize financial data.



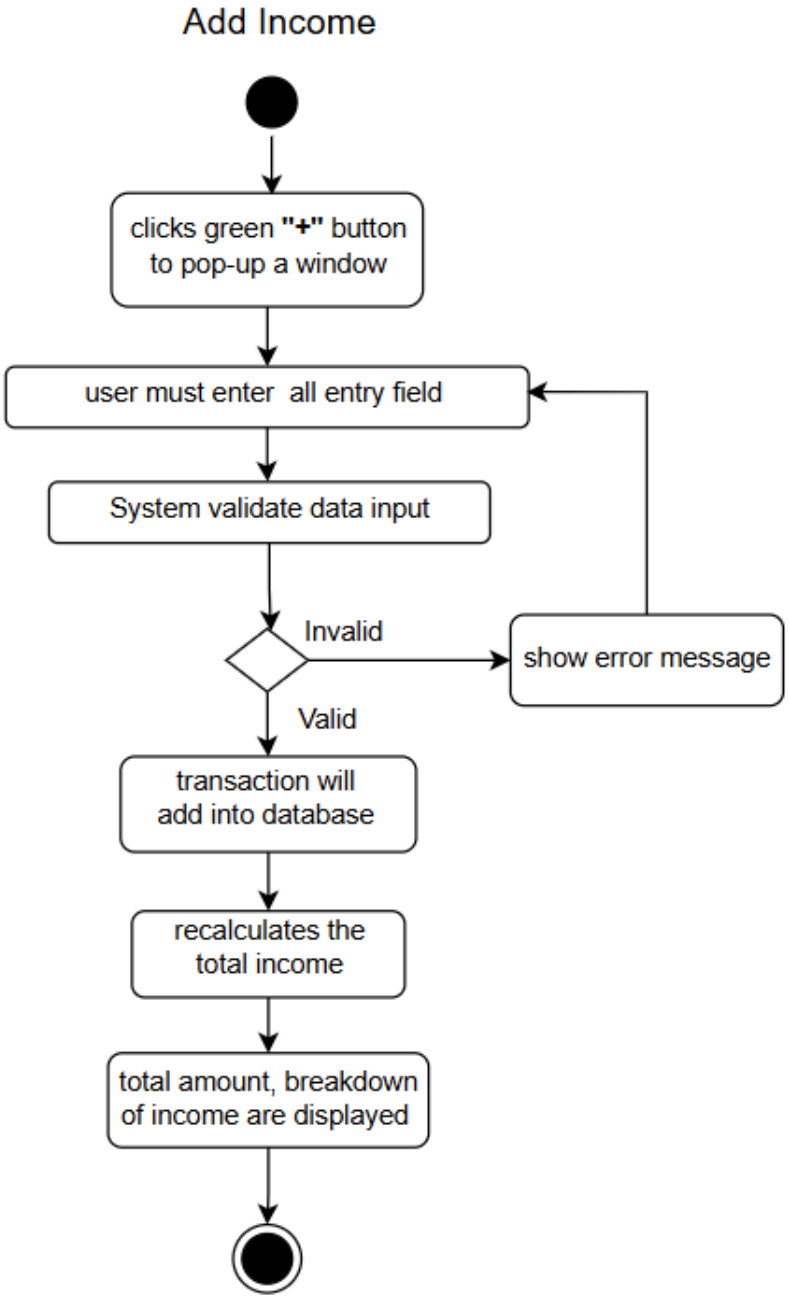
4.8.2.3 Set Budget

The user initiates by clicking the **"Set Budget" button** and is prompted to enter an estimated budget. The system validates the input data. If the data is invalid, an **error message** is displayed, and the user is prompted to re-enter the information. If valid, the system saves the estimated budget in the database and generates a pie chart showing the financial breakdown. The system then monitors the remaining budget. If the budget drops to **10% or less**, the user is alerted; otherwise, the process continues without an alert. This ensures users stay informed about their budget limits.



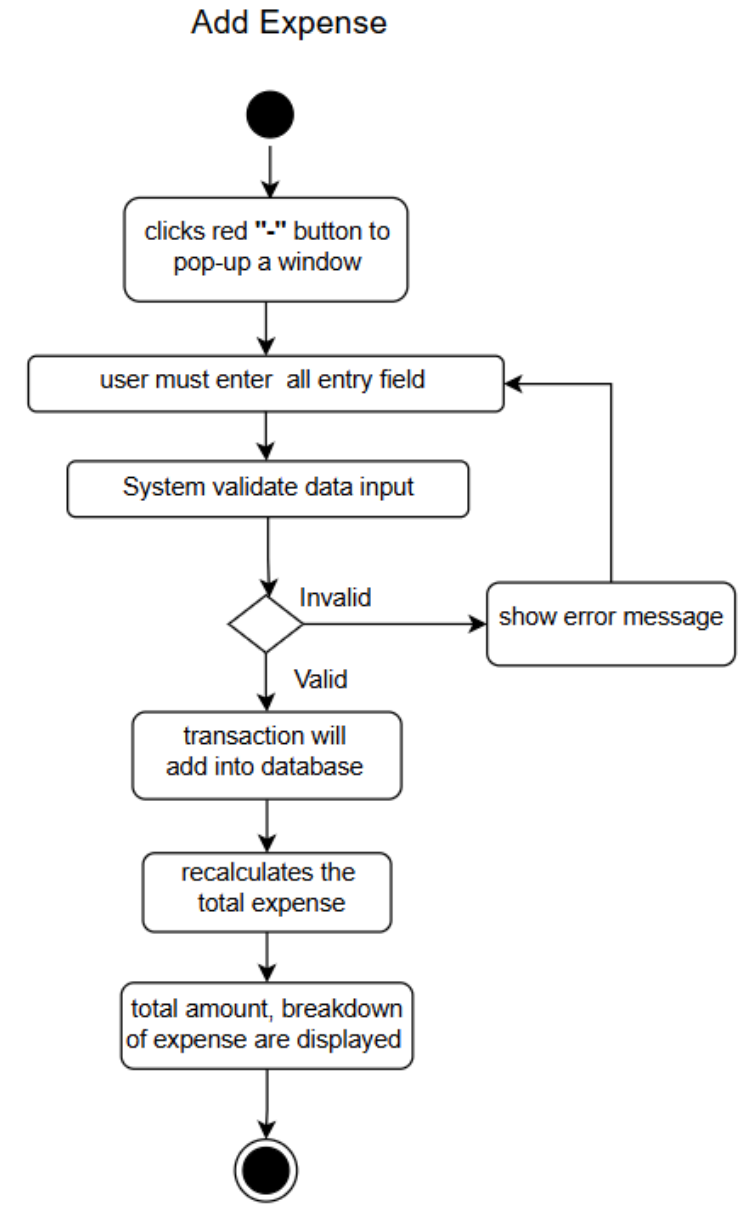
4.8.2.4 Add Income

When the user clicks the **green "+" button**, which opens a pop-up window for entering income details. The user must fill in all required fields, and the system validates the input. If the input is invalid, an **error message** is displayed, and the user is prompted to correct the information. If valid, the transaction is added to the database, the total income is recalculated, and a detailed breakdown of the income is displayed for the user.



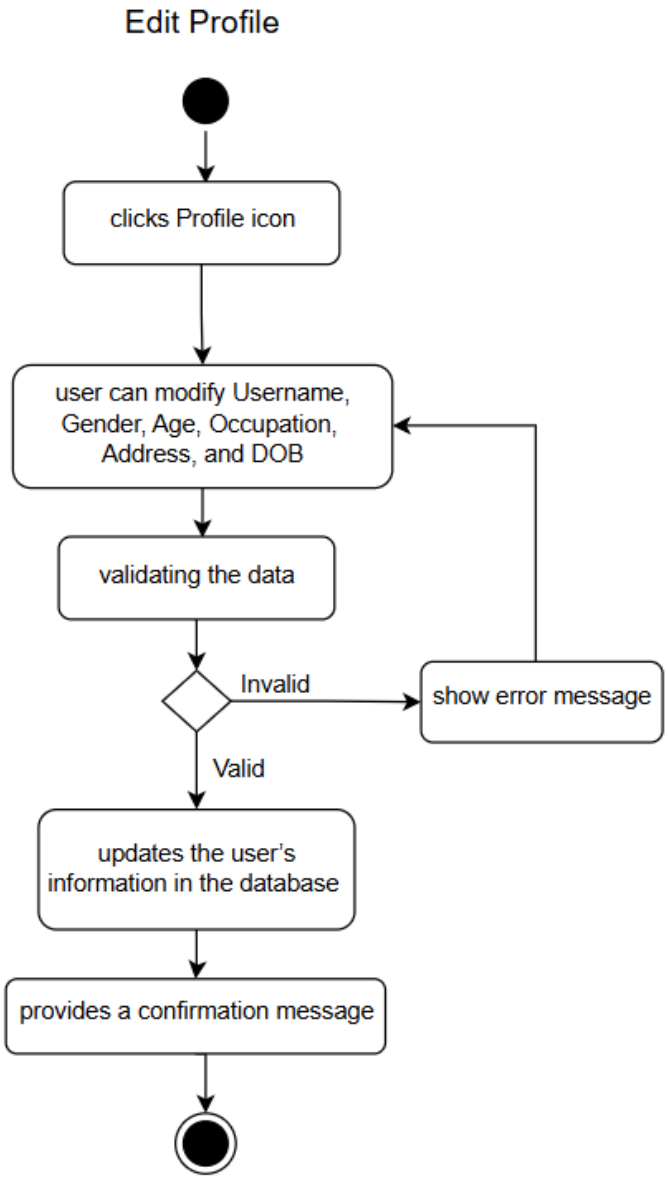
4.8.2.5 Add Expense

When the user clicks the **red "-" button**, triggering a pop-up window for entering expense details. The user must complete all required fields, and the system validates the data. If the input is invalid, an **error message** prompts the user to correct the entry. When valid, the expense is added to the database, the total expense is recalculated, and the user is presented with a detailed breakdown of their expenses.



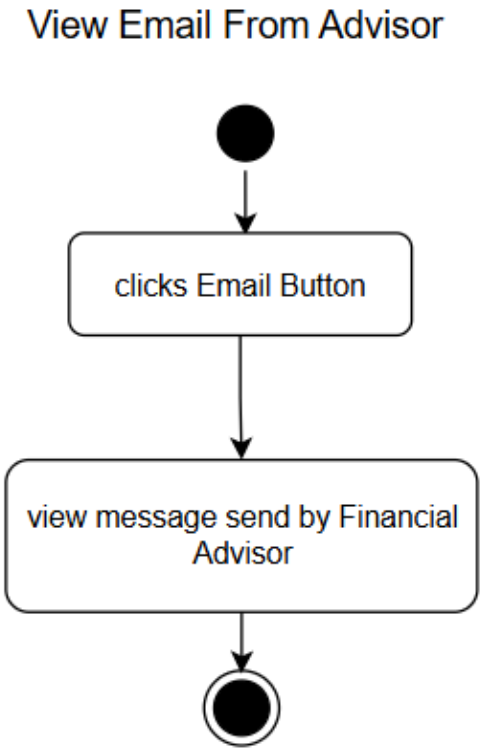
4.8.2.6 Edit Profile

The **Edit Profile** process begins when the user clicks the **Profile icon**. This action allows the user to modify personal details, including **Username**, **Gender**, **Age**, **Occupation**, **Address**, and **Date of Birth (DOB)**. Once the user makes the changes, the system validates the data. If the input is invalid, an **error message** is displayed, and the user is prompted to correct the errors. If the data is valid, the updated information is saved to the database, and the system provides a **confirmation message** to inform the user of the successful update.



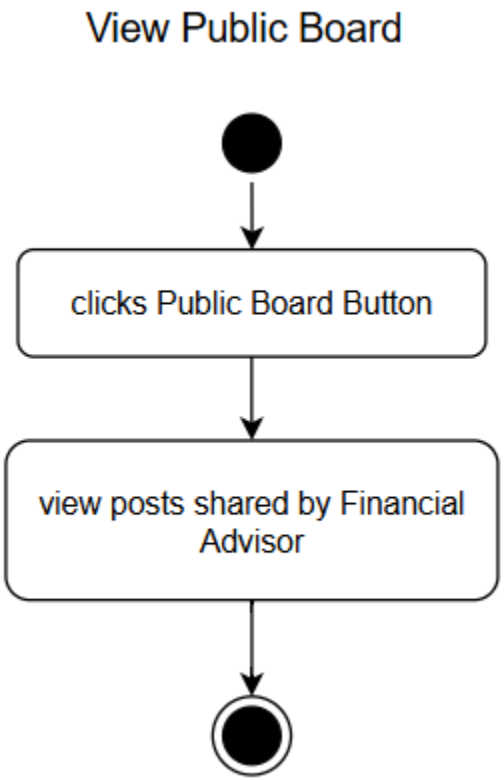
4.8.2.7 View Email From Advisor

The **View Email From Advisor** process begins when the user clicks the **Email Button**. This action allows the user to view messages sent by their financial advisor. The process is straightforward and provides the user with easy access to important communications.



4.8.2.8 View Public Board

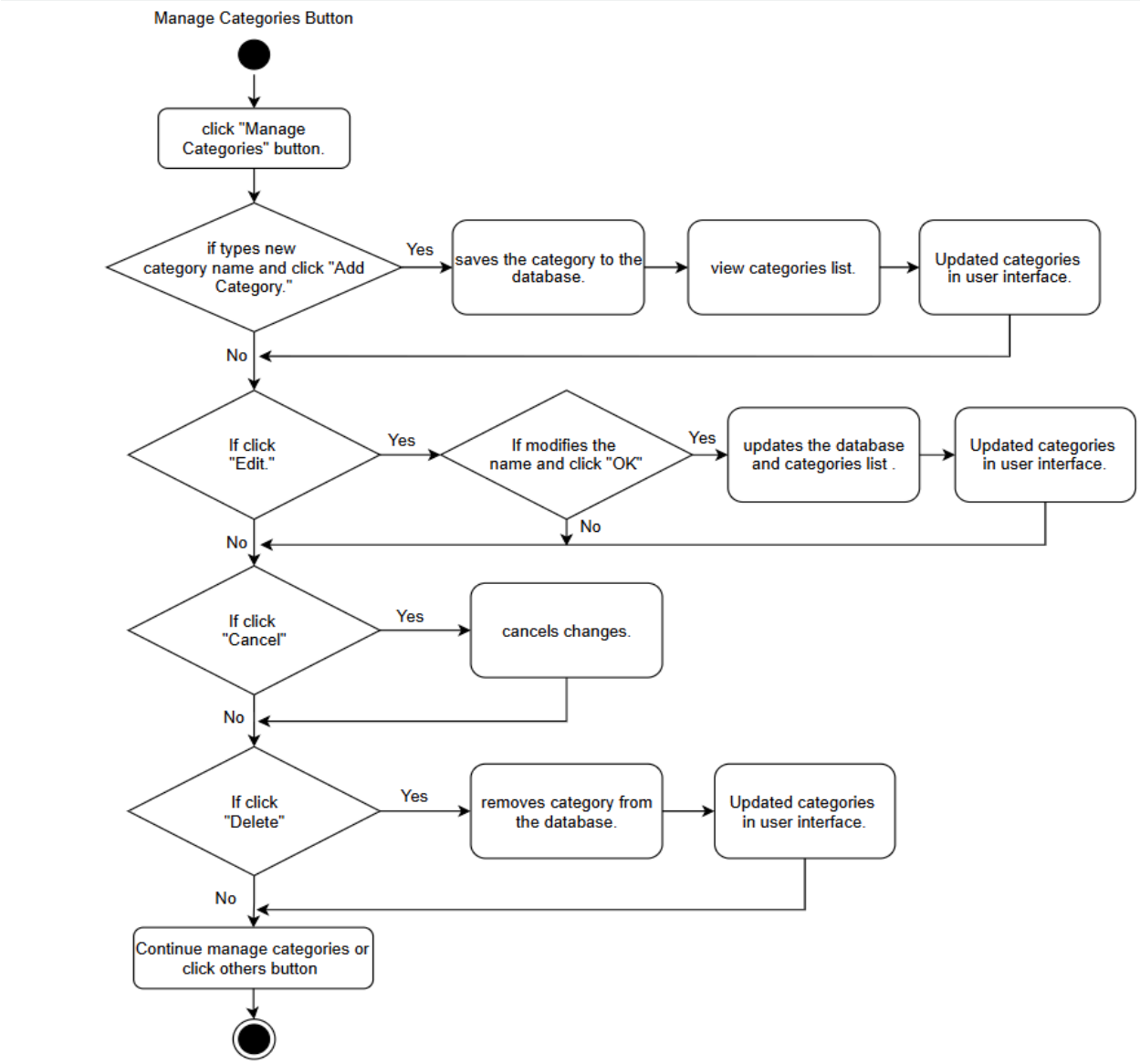
The **View Public Board** process starts when the user clicks the **Public Board Button**. This action enables the user to view posts shared by their financial advisor on the public board, offering insights or updates intended for a broader audience.



4.8.3 Admin Component

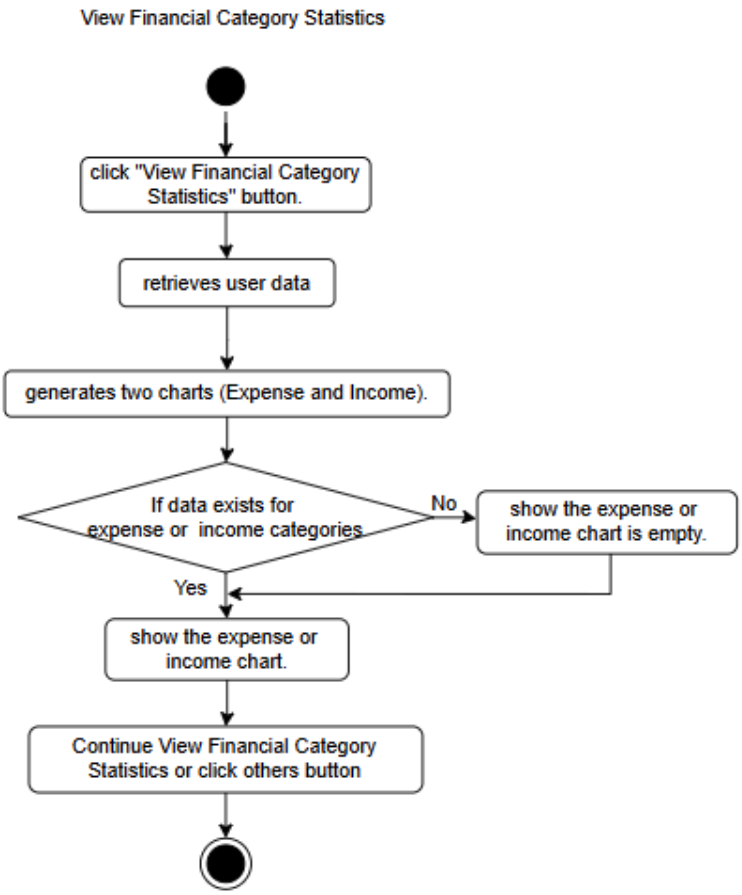
4.8.3.1 Manage Categories

The **Manage Categories** feature allows users to add, edit, delete, or cancel modifications to categories. Users can type a new category name and click **"Add Category"**, which validates and saves the data to the database, updating the category list in the interface. For editing, admin click **"Edit"**, modify the name, and confirm with **"OK"**, after which the system validates and updates the database. If users cancel during editing, changes are discarded. For deletion, users click **"Delete"**, confirm the action, and the category is removed from the database, with updates reflected in the interface. Throughout, error handling ensures invalid inputs are flagged, and confirmations are required for critical actions like deletion, ensuring a smooth and user-friendly process.



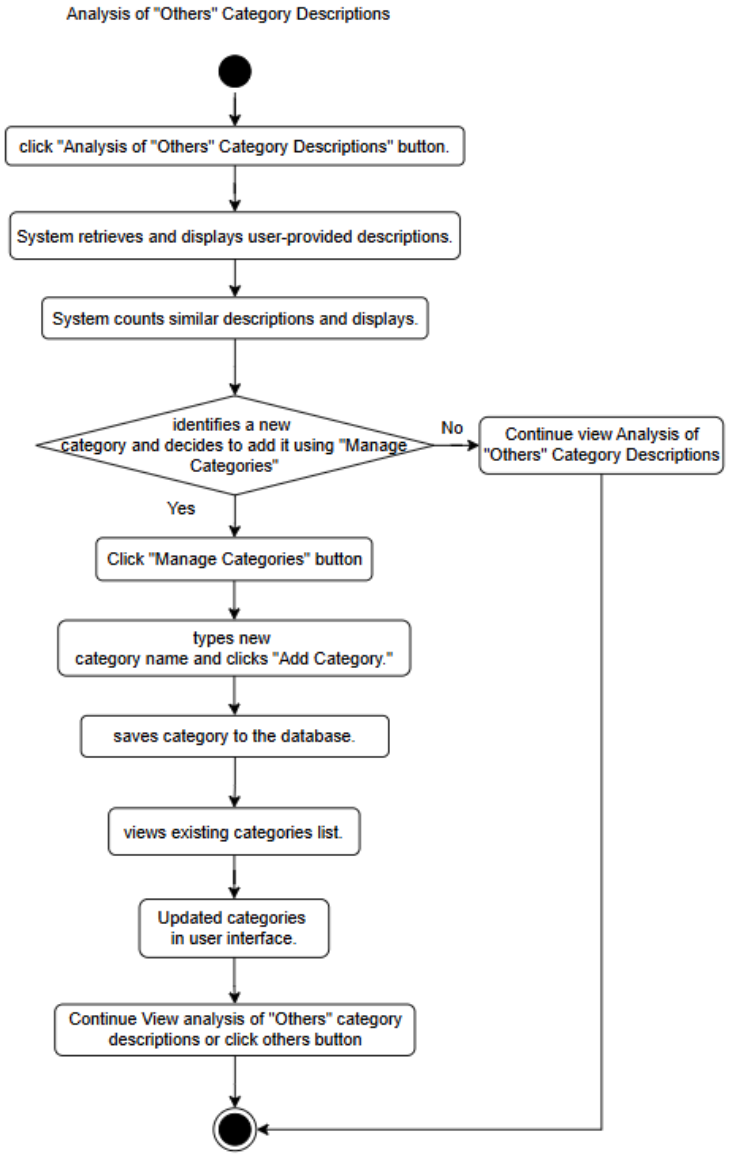
4.8.3.2 View Financial Category Statistics

The **View Financial Category Statistics** feature allows users to visualize their financial data through charts. When the user clicks the **"View Financial Category Statistics"** button, the system retrieves the user's financial data. It generates two charts: one for expenses and one for income. If the data exists for either category, the respective chart is displayed to the admin. If no data is available, the system shows an empty chart with a message indicating the lack of data. After viewing the statistics, the user can continue exploring this feature or navigate to other sections of the application. This design ensures that users can easily understand and track their financial activities through clear visual representations.



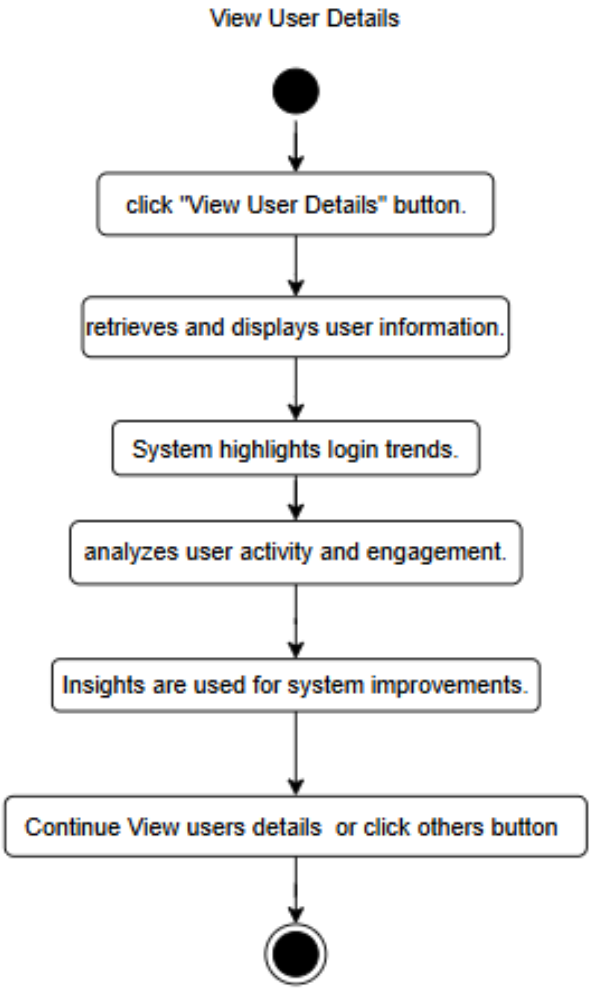
4.8.3.3 Analysis "Others" Descriptions

The **Analysis of "Others" Category Descriptions** feature allows admin to review and organize descriptions under the "Others" category. Upon clicking the **"Analysis of 'Others' Category Descriptions"** button, the system retrieves and displays user-provided descriptions, counts similar entries, and presents them for analysis. If the admin identifies a new category, admin can add it using the **"Manage Categories"** feature by entering the category name, which is then saved to the database and reflected in the updated category list.Admin can continue analyzing descriptions or navigate to other sections, ensuring better data organization and clarity.



4.8.3.4 View User Details

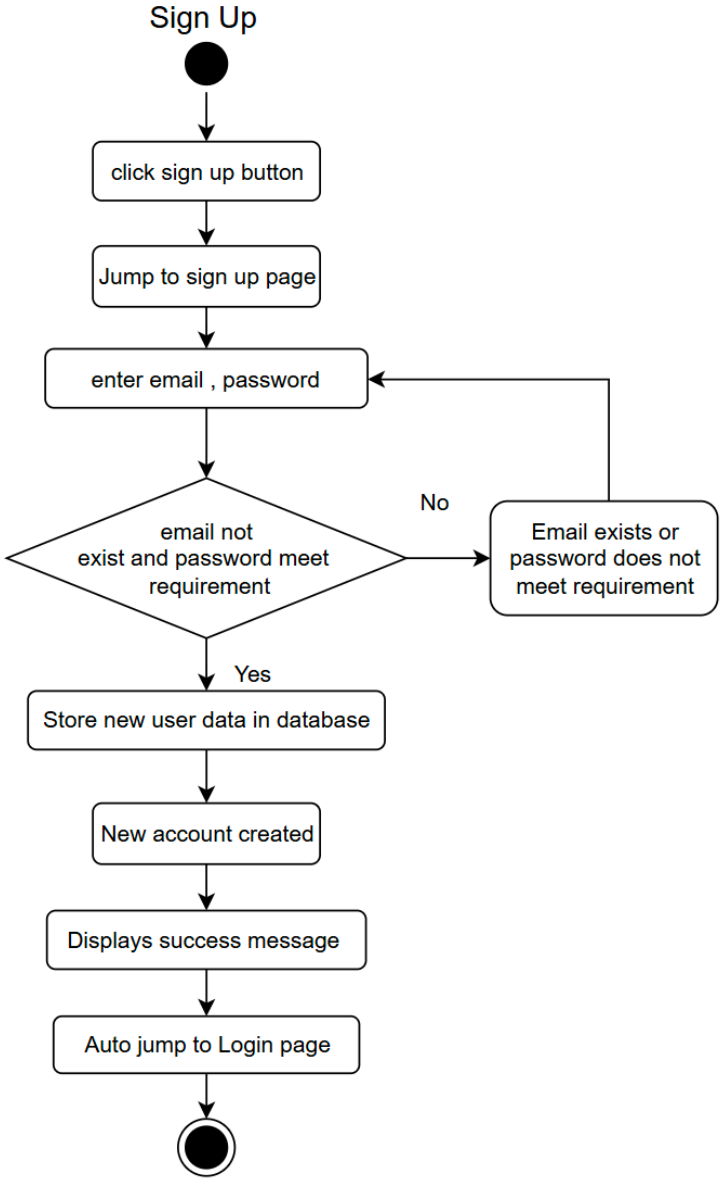
The **View User Details** feature allows admin to access detailed insights about user activity. When the **"View User Details"** button is clicked, the system retrieves and displays user information, including profile details and login trends. It further analyzes user activity and engagement patterns to provide meaningful insights. These insights are then utilized to identify areas for system improvements, ensuring a better user experience. After viewing the details, the admin can either continue reviewing the information or navigate to other sections of the application.



4.8.4 Guest Component

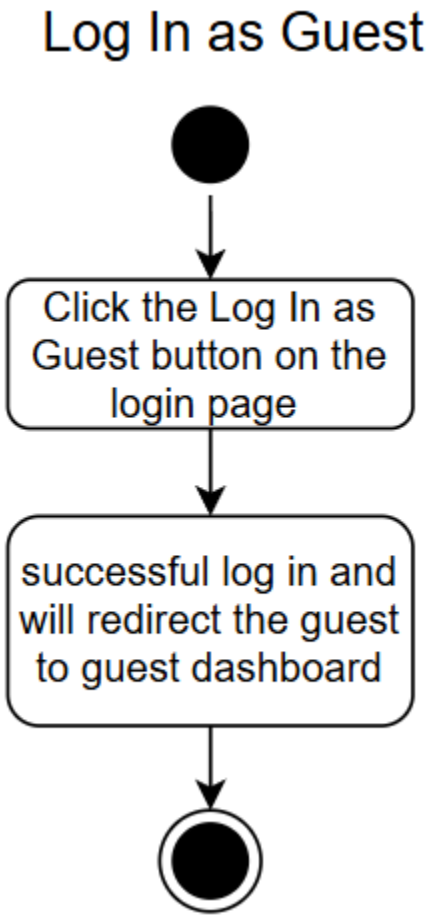
4.8.4.1 Sign Up

When the guest clicks the **Sign-Up Button**, which redirects them to the **Sign-Up Page**. The guest enters their email and password, which are then validated. If the email does not already exist and the password meets the specified requirements, the system stores the new user's data in the database, creating an account. A **success message** is displayed, and the user is automatically redirected to the **Login Page**. If validation fails, the system informs the user of the issue, such as an existing email or invalid password, prompting them to re-enter the details.



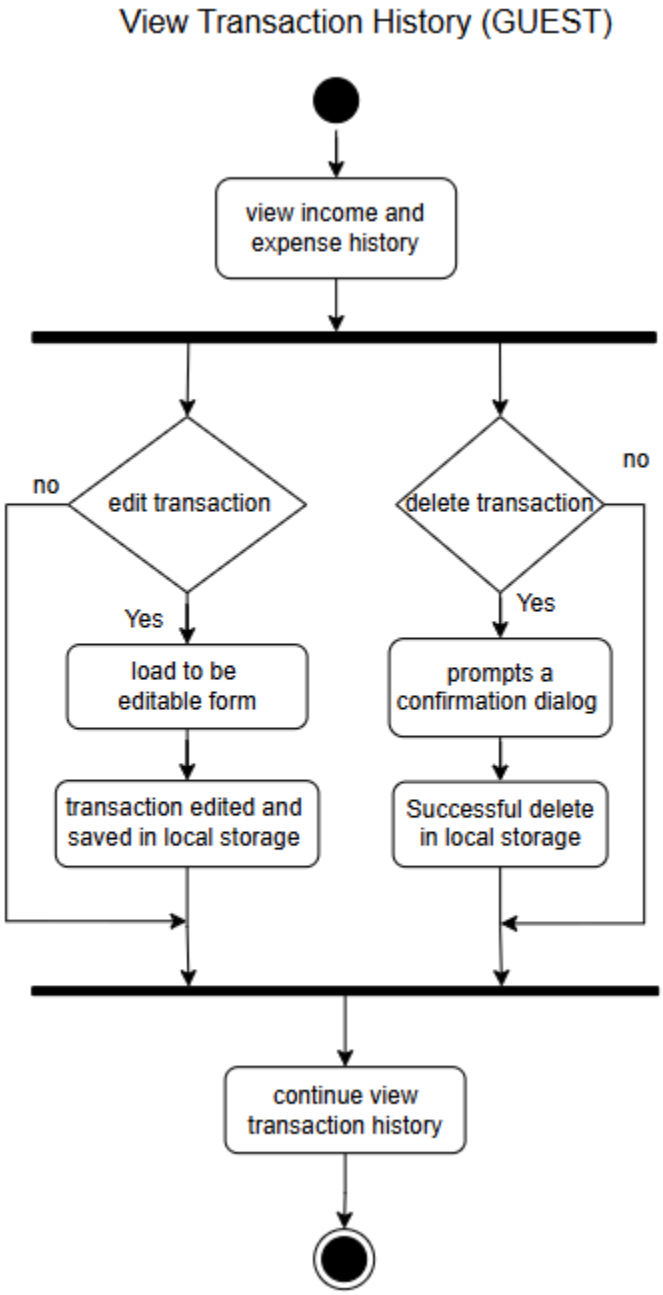
4.8.4.2 Log In as Guest

The Log In as Guest flow offers a straightforward way for guests to access the system. By clicking the "Log In as Guest" button on the login page, the guest is authenticated and redirected to the guest dashboard. This enables temporary access to system features like viewing income/expense history and generating visual data, offering flexibility without storing sensitive user information.



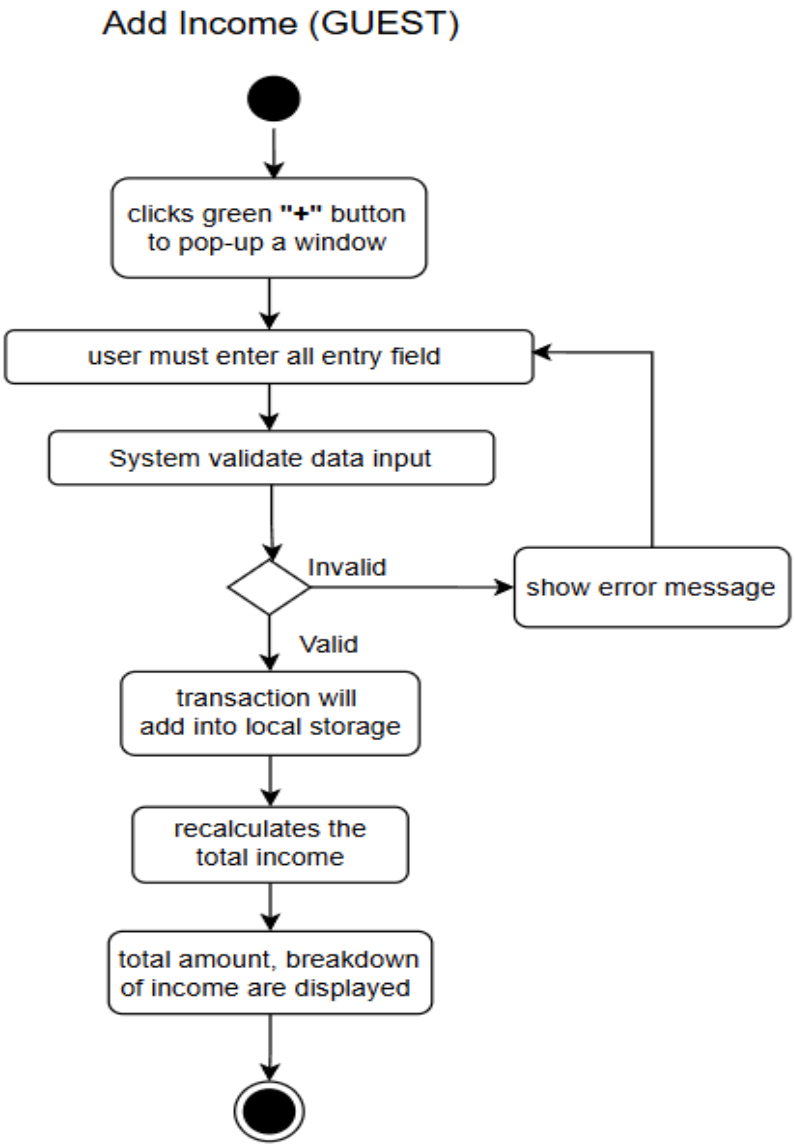
4.8.4.3 View Transaction History (GUEST)

The guest begins by viewing their income and expense history. They can choose to **edit a transaction** or **delete a transaction**. If the guest chooses to edit, the selected transaction is loaded into an editable form, where changes are made and saved back to the local storage. If the guest chooses to delete, a confirmation dialog appears; upon confirmation, the transaction is successfully deleted from the local storage. In both cases, the guest returns to continue viewing the transaction history. If no action is taken, the guest remains on the transaction history page.



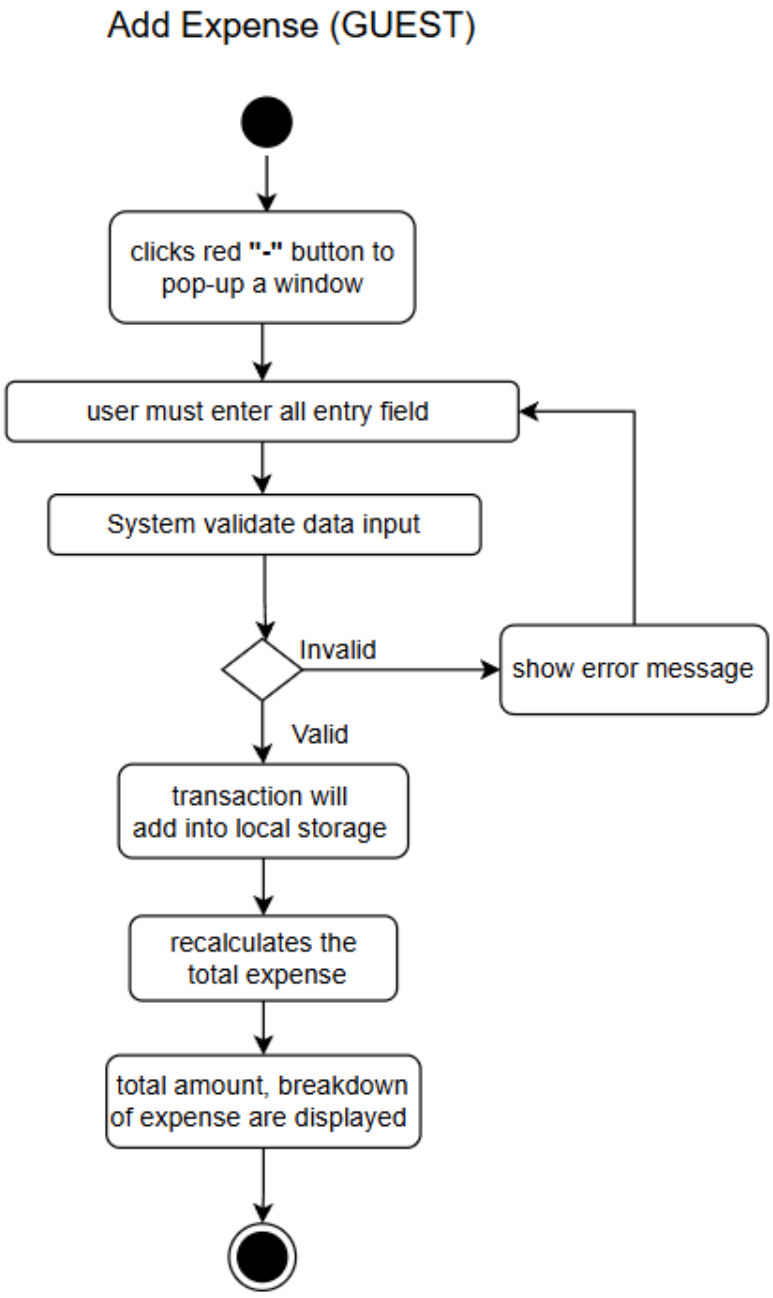
4.8.4.4 Add Income (GUEST)

When the guest clicks the **green "+" button**, which opens a pop-up window for entering income details. The guest must fill in all required fields, and the system validates the input. If the input is invalid, an **error message** is displayed, and the guest is prompted to correct the information. If valid, the transaction is added to the local storage, the total income is recalculated, and a detailed breakdown of the income is displayed for the guest.



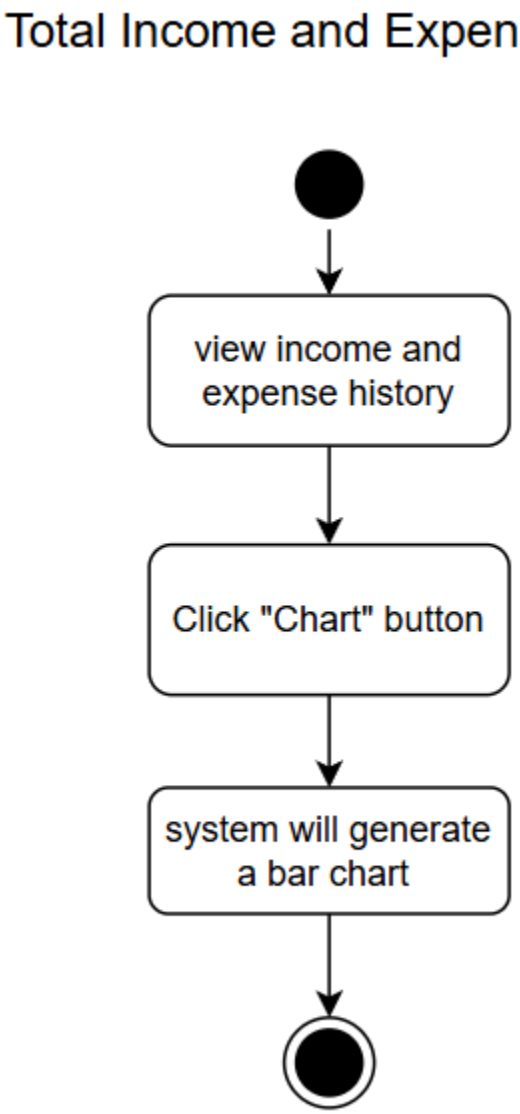
4.8.4.5 Add Expense (GUEST)

When the guest clicks the **red "-" button**, triggering a pop-up window for entering expense details. The guest must complete all required fields, and the system validates the data. If the input is invalid, an **error message** prompts the guest to correct the entry. When valid, the expense is added to the local storage, the total expense is recalculated, and the guest is presented with a detailed breakdown of their expenses.



4.8.4.6 View Bar Chart Total Income and Expenses (GUEST)

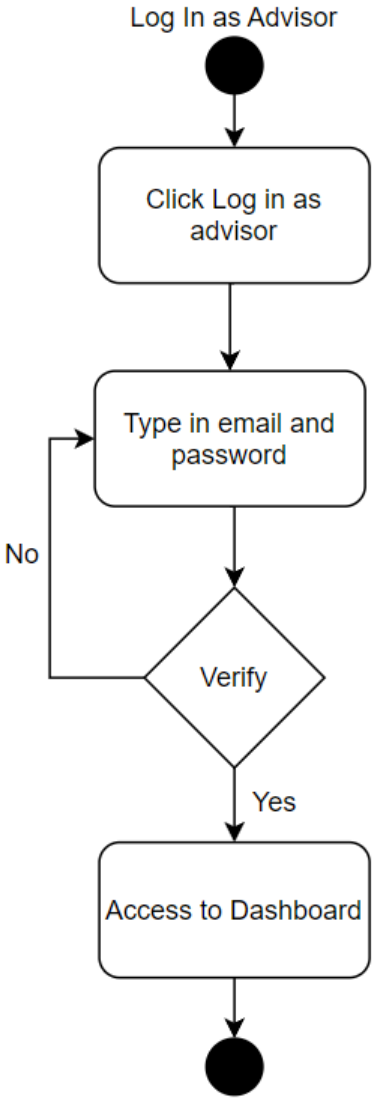
The View Bar Chart Total Income and Expenses (GUEST) flow allows a guest user to view their financial data visually. The process begins with the guest accessing the income and expense history. Upon clicking the "Chart" button, the system generates a bar chart representing the distribution of income and expenses. This simple, intuitive process provides a clear graphical representation for better financial analysis without requiring user credentials.



4.8.5 Financial Advisor Component

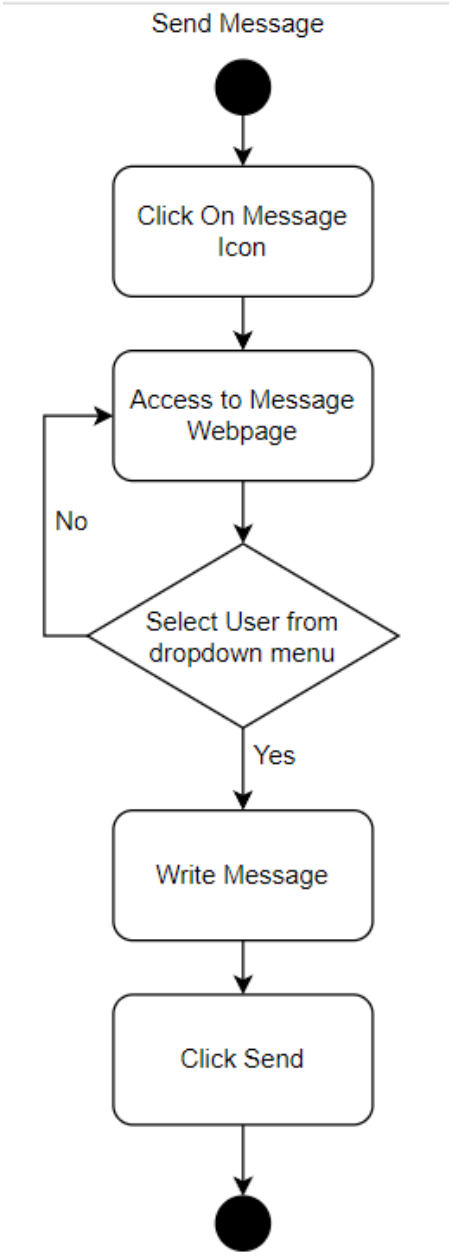
4.8.5.1 Log In as Advisor

Serves as a login page in order special for advisors so they can have access to their personal dashboard. Advisors will click on log in as advisor, then redirect to a login page for them and then inputting the correct email and password will only grant them access to the dashboard.



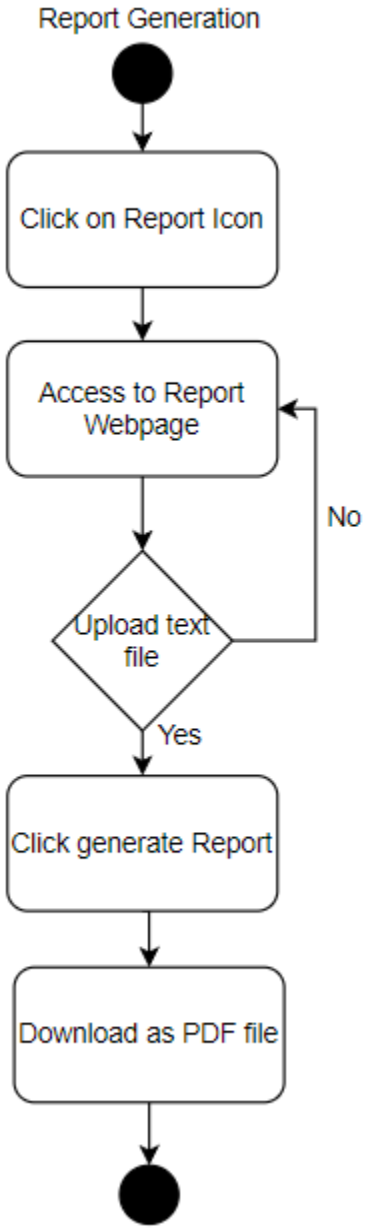
4.8.5.2 Send Messages

This webpage serves as a way of communication from advisor to user about their financial situation. The advisors first selects users from a drop-down menu which fetches data from the database and then once selected, they can write whatever they want and then click send and the message will be saved under user data.



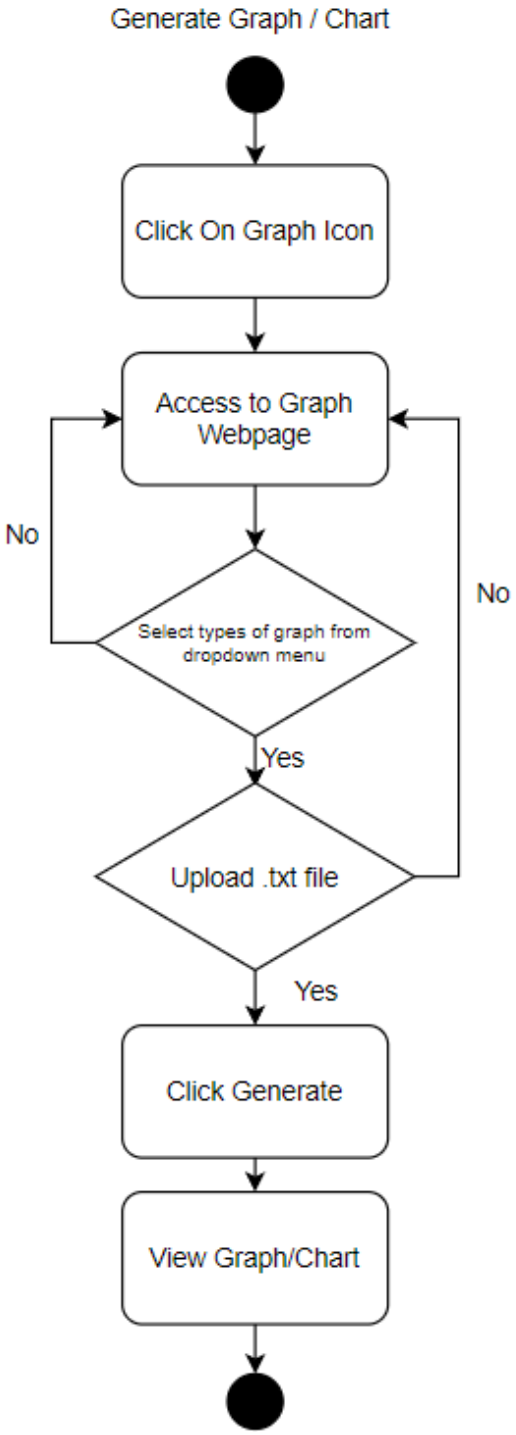
4.8.5.3 Generate Report

This webpage serves as a way for advisors to analyse user's data by creating a report to do so. They first upload the data, which must be in a text file, and then once uploaded successfully, can click generate and then a report will be generated and downloaded as a PDF file.



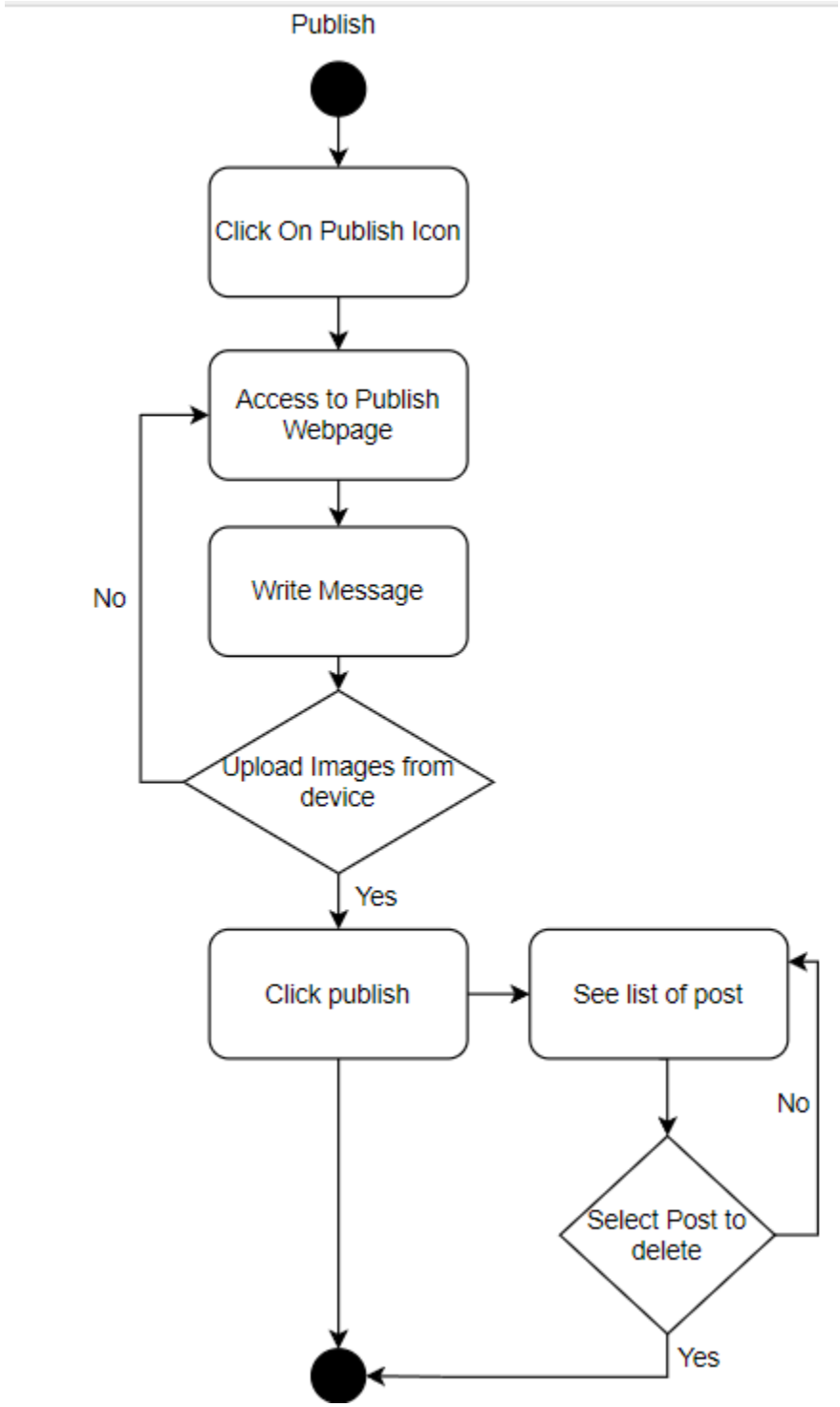
4.8.5.4 Generate Graph/Chart

This is also another web page that serves as a way for advisors to analyze data through graphs and charts! All they have to do is just select the type of graph or chart they want to see and then upload the data in a text file, once successful, they can click on generate and a graph or chart will be shown visually.



4.8.5.5 Publish

This webpage is used if the advisors want to share something interesting whether it's by text or images. They first write something or upload an image from the device and once successful, they can click publish and then the post will be seen on the public board. If they want to delete, they just need to select the post from the list and click delete.



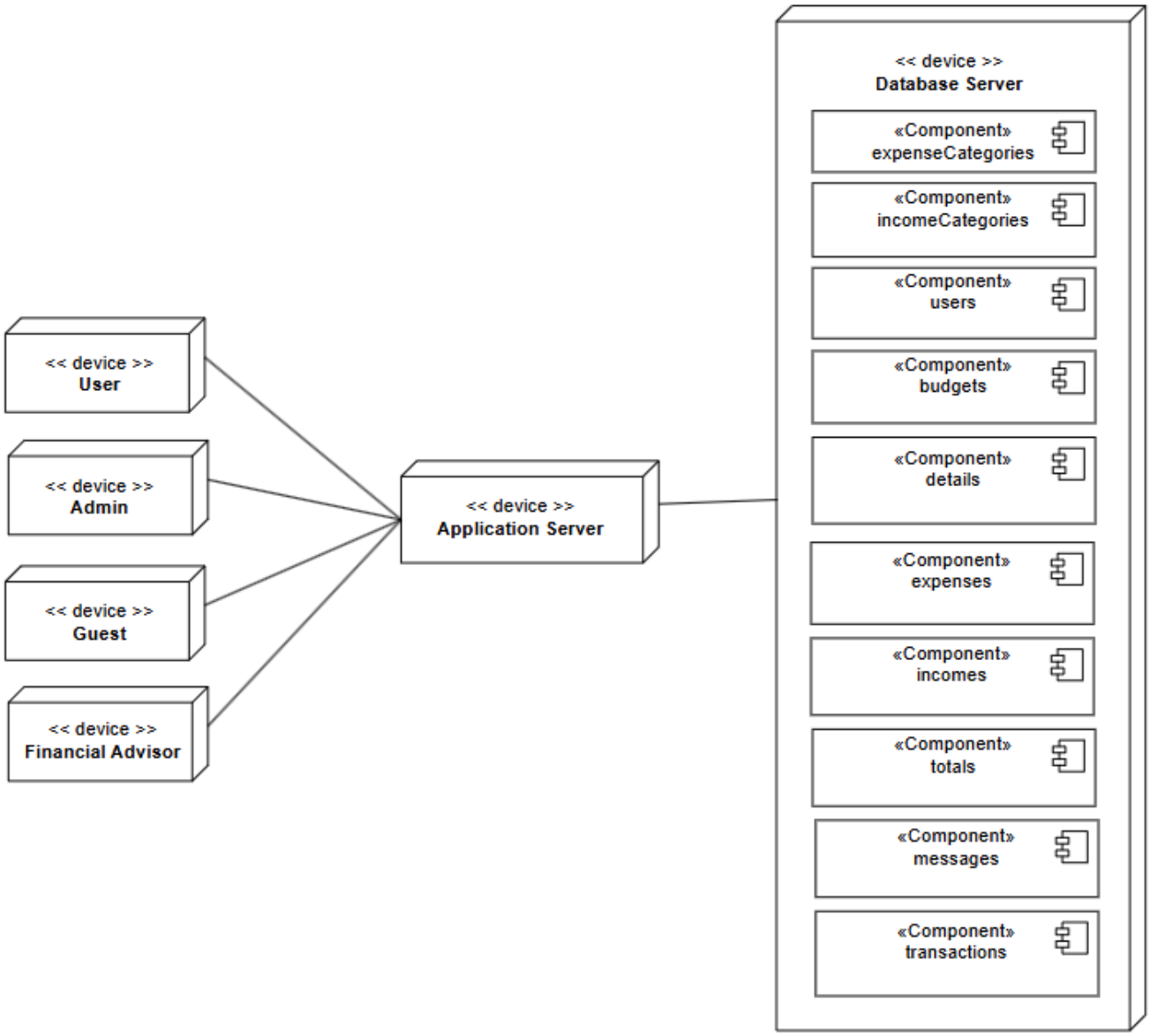
4.9 Deployment Diagram

<TO DO: Describe the deployment diagram and place the diagram here.>

A deployment diagram in UML (Unified Modeling Language) represents the physical deployment of artifacts (such as software components) on nodes (hardware or software environments). It provides a clear view of how a system's software and hardware interact. In the provided deployment diagram, there are three main nodes: user devices, the application server, and the database server. User devices are categorized into three roles: User, Guest, and Financial Advisor, which represent the different client types interacting with the system. These devices communicate with the application server, which acts as the central hub responsible for processing requests and executing business logic. The application server connects to the database server, where all data is stored.

The database server contains several components that represent logical divisions of data, such as expenseCategories, incomeCategories, users, budgets, and transactions. These components likely correspond to database tables or modules designed to handle specific aspects of financial data. The system appears to be a financial management platform where users can track expenses, incomes, budgets, and transactions, while messages and summaries provide additional interaction and insights.

The communication paths between the user devices, application server, and database server indicate how requests are processed. For example, users submit their actions (like managing budgets or tracking expenses) through the application server, which retrieves or updates the relevant data in the database server. This structure ensures modularity, scalability, and clear separation of concerns, making it easier to manage and maintain the system. Overall, the deployment diagram provides a high-level overview of the system's physical architecture and its interaction between hardware and software components.


<< device >>
User
<< device >>
Admin
<< device >>
Guest
<< device >>
Financial Advisor
<< device >>
Application Server
<< device >>
Database Server
«Component»
expenseCategories
«Component»
incomeCategories
«Component»
users
«Component»
budgets
«Component»
details
«Component»
expenses
«Component»
incomes
«Component»
totals
«Component»
messages
«Component»
transactions

5 Implementation

5.1 Development Environment

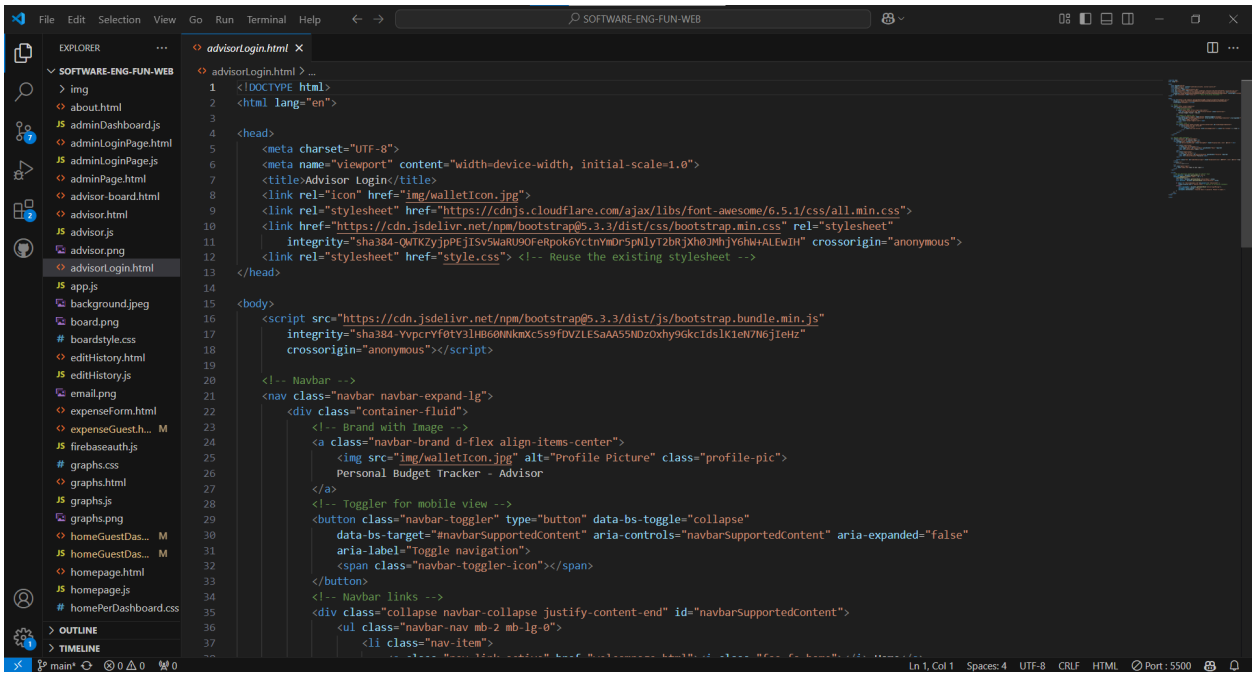
<TO DO: Describe the development environment here with relevant images.>

In this assignment, we have used Visual Studio Code as our main IDE and used Firebase as our database.

VS Code:

For this project, Visual Studio Code (VS Code) was used as the primary development environment. VS Code is a lightweight yet powerful editor developed by Microsoft, known for its user-friendly interface, flexibility, and extensive support for multiple programming languages.

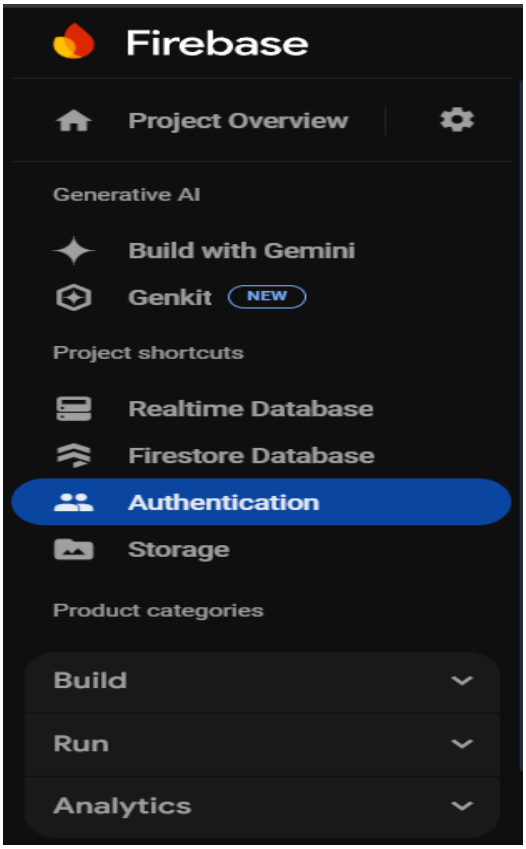
VS Code includes an integrated terminal, built-in debugging tools, and Git integration, allowing seamless coding, testing, and version control. Its extension support enables developers to enhance functionality with plugins for Python, C++, Java, and other tools. Additionally, customization options like themes and shortcuts improve workflow efficiency.



Firebase:

Firebase is a cloud-based platform by Google that provides backend services like real-time databases, authentication, cloud storage, and hosting. It simplifies app development by handling data synchronization, user authentication, and secure storage.

Key features include the Realtime Database and Firestore for live data updates, Authentication Services for secure logins, and Cloud Storage for managing user-generated content. Additionally, Cloud Functions enable serverless backend logic, while Firebase Hosting ensures fast and secure web deployment



Programming elements:

HTML:

HTML (HyperText Markup Language) is the standard language used for creating and structuring web pages. It defines the basic layout and content of a webpage using elements such as headings, paragraphs, images, links, and forms.

CSS:

CSS(**Cascading Style Sheets**) is a stylesheet language that is used in order to change the appearance and layout of HTML elements. It helps to enhance the visual presentation of these web pages by making styles such as maybe adding colors, spacing or even positioning. Every property contains one or more values that define how it looks or behave in a certain way.

Java Script:

JavaScript (**JS**) is a special programming language that helps these web pages by adding interactions and functions. It allows developers to create responsive user interactions, handle events, manipulate CSS and HTML and interact with APIs.

JavaScript runs in browsers and is used for things such as form validation, animations or event handling. It works alongside HTML for structure and CSS for styling, making this language an essential part of web development.

5.2 Software Integration

<TO DO: Describe a strategy to integrate all the subsystems here with relevant images.>

File	Description
1. Visual Studio Code (IDE)	1. Code editor for writing HTML, CSS, and JavaScript. 2. Debugging tools for error handling. 3. Git integration for version control.
2. HTML, CSS, and JavaScript	1. HTML templates are linked to CSS for styling. 2. JavaScript files are used to handle UI behavior (e.g., form validation, API requests). 3. All frontend files are managed and edited in Visual Studio Code.
3. Firebase (Firestore)	1. The frontend (JavaScript) communicates with Firebase using the Firebase SDK. 2. Firestore queries allow retrieving, updating, and managing data dynamically. 3. Firebase Authentication manages user logins and session tracking.

Relevant images:

HTML:

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Admin Login</title>
    <link rel="icon" href="img/walletIcon.jpg">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">
    <link rel="stylesheet" href="style.css"> <!-- Reuse the existing stylesheet -->
</head>
```

CSS:

```css
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    flex-direction: column;
    align-items: center;
}

header {
    background-color: #4CAF50;
    color: white;
    padding: 1rem;
    text-align: center;
    width: 100%;
}

main {
    padding: 1rem;
    max-width: 600px;
    width: 100%;
}
```

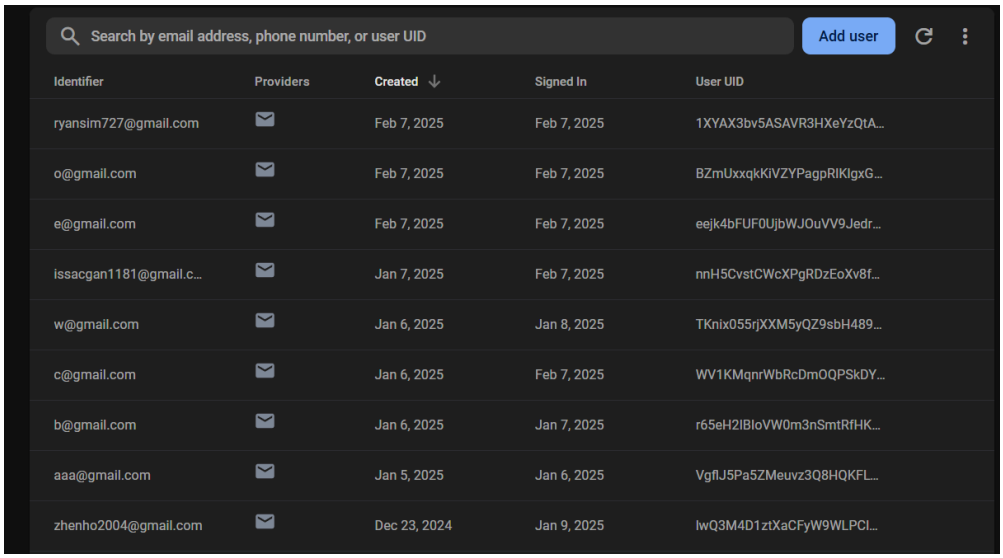
JS:

```js
viewGraphButton.addEventListener('click', () => {
    const graph = graphType.value;
    const file = fileInput.files[0];

    if (!file) {
        alert('Please upload a data file.');
        return;
    }

    const reader = new FileReader();
    reader.onload = () => {
        const fileData = reader.result.trim().split('\n');
        if (graph === 'expenses') {
            createExpensesGraph(fileData);
        } else if (graph === 'budget') {
            createBudgetGraph(fileData);
        } else if (graph === 'savings') {
            createSavingsGraph(fileData);
        }
    };
    reader.readAsText(file);
});
```

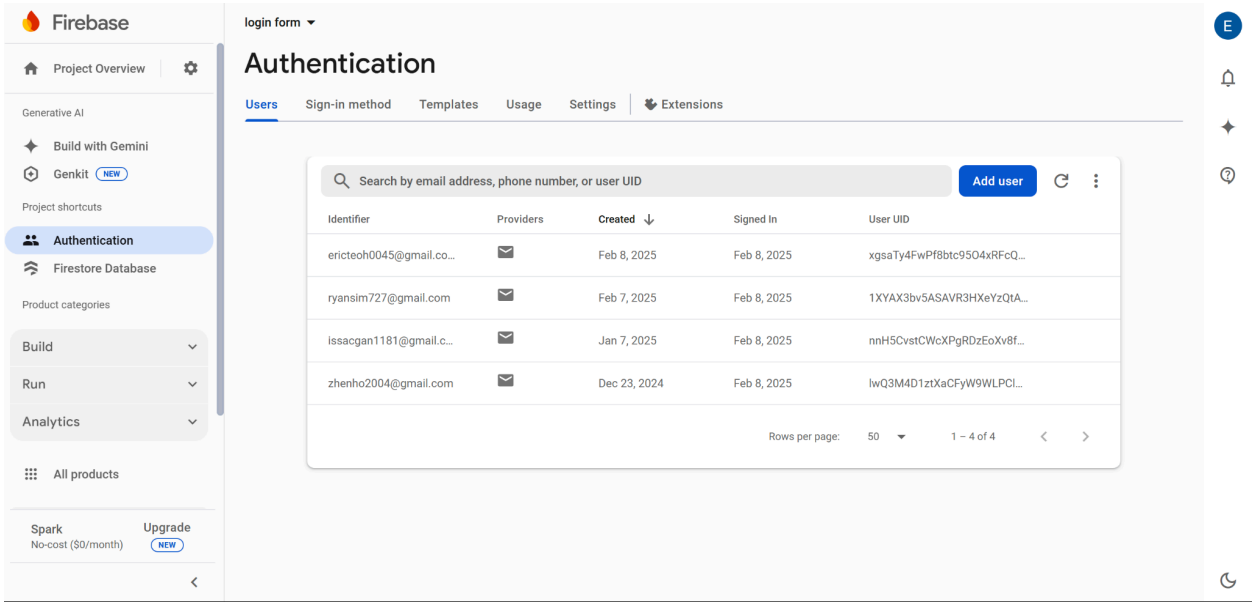
FireBase:



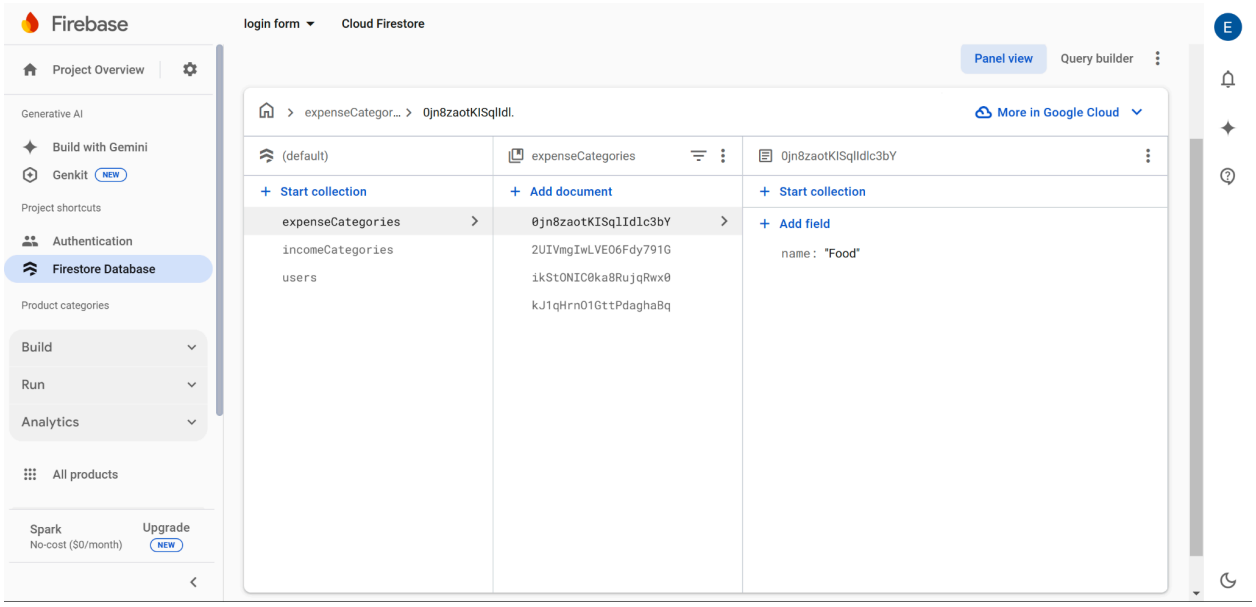
5.3 Database

<TO DO: Describe the database implementation here with relevant images.>

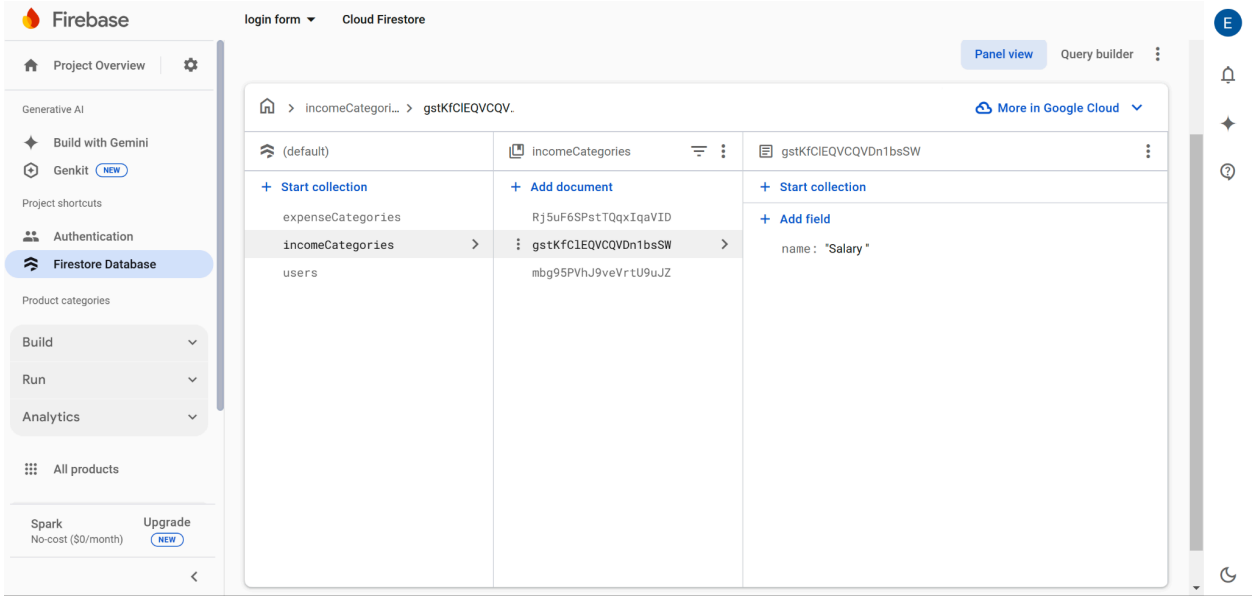
This authentication management page allows administrators to view user accounts, including their email addresses, account creation dates, last login dates, and unique user IDs (UIDs)



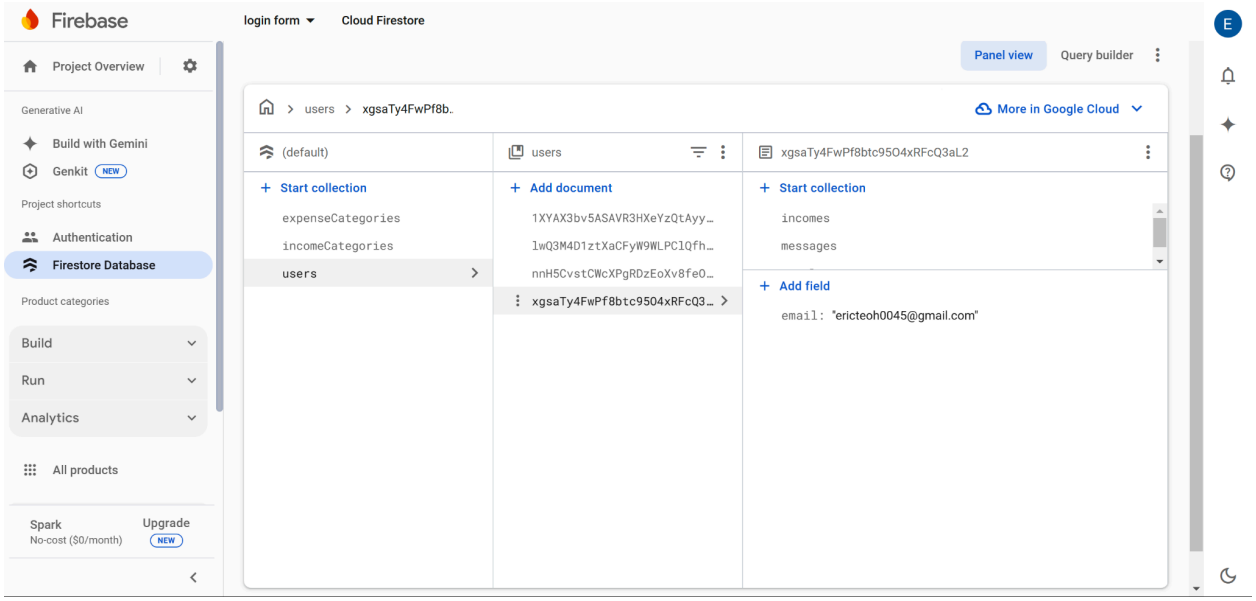
This Cloud Firestore interface displays the database structure, allowing administrators to view and manage expense categories. Within the expenseCategories collection, individual documents represent different expense types, including one with the field name: "Food".



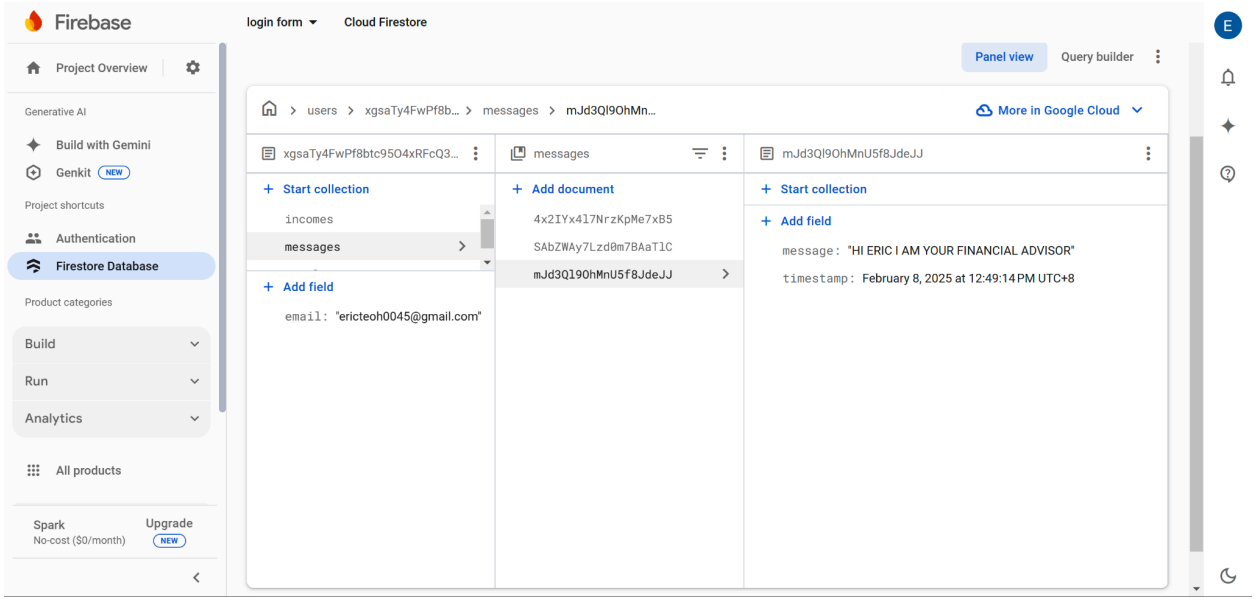
This Cloud Firestore interface displays the database structure, allowing administrators to view and manage income categories. Within the incomeCategories collection, individual documents represent different income types, including one with the field name: "Salary".



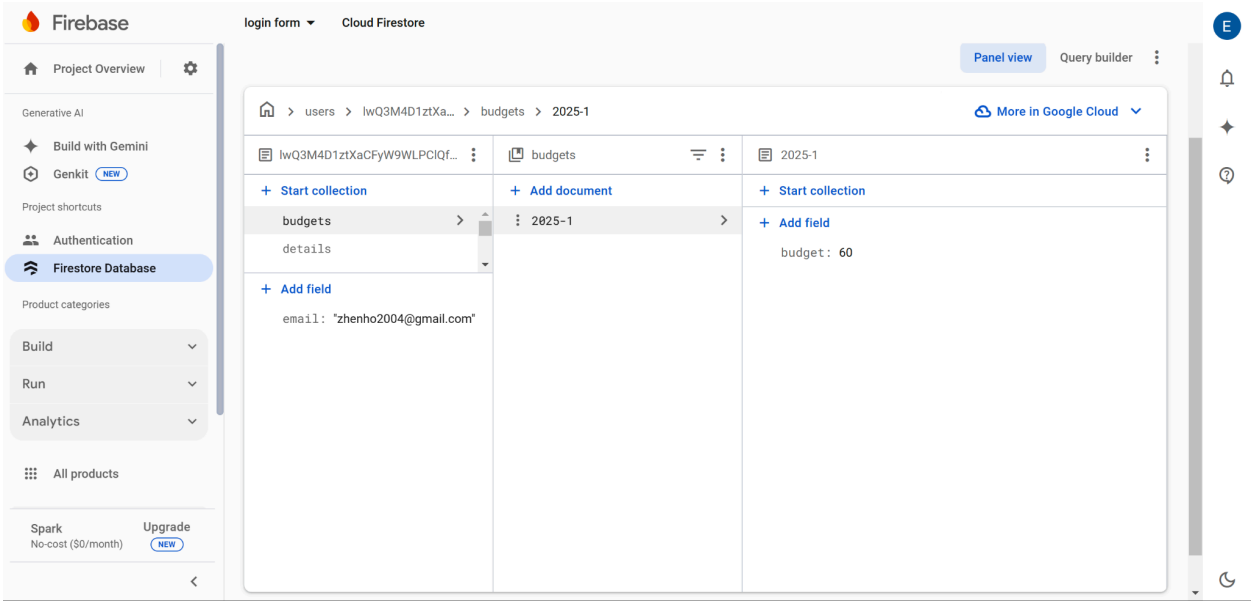
This Cloud Firestore interface displays the users collection, allowing administrators to view and manage user accounts. The currently selected user document contains an email field with the email address. Additionally, there is a sub-collection named messages.



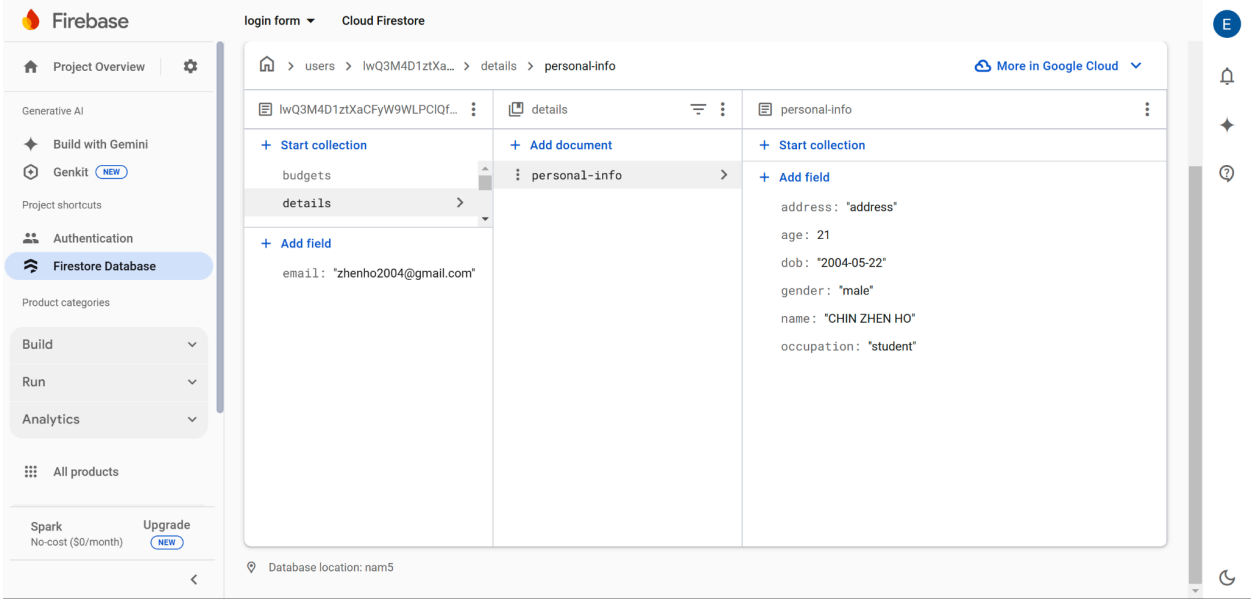
The Cloud Firestore interface displays a user's messages stored in a sub-collection under their unique user ID. Each message is represented as a document with a unique identifier. The document contains a message with the content, along with a timestamp indicating when the message was sent. This structure allows efficient tracking and retrieval of messages along with their corresponding timestamps, ensuring a well-organized messaging system within the application.



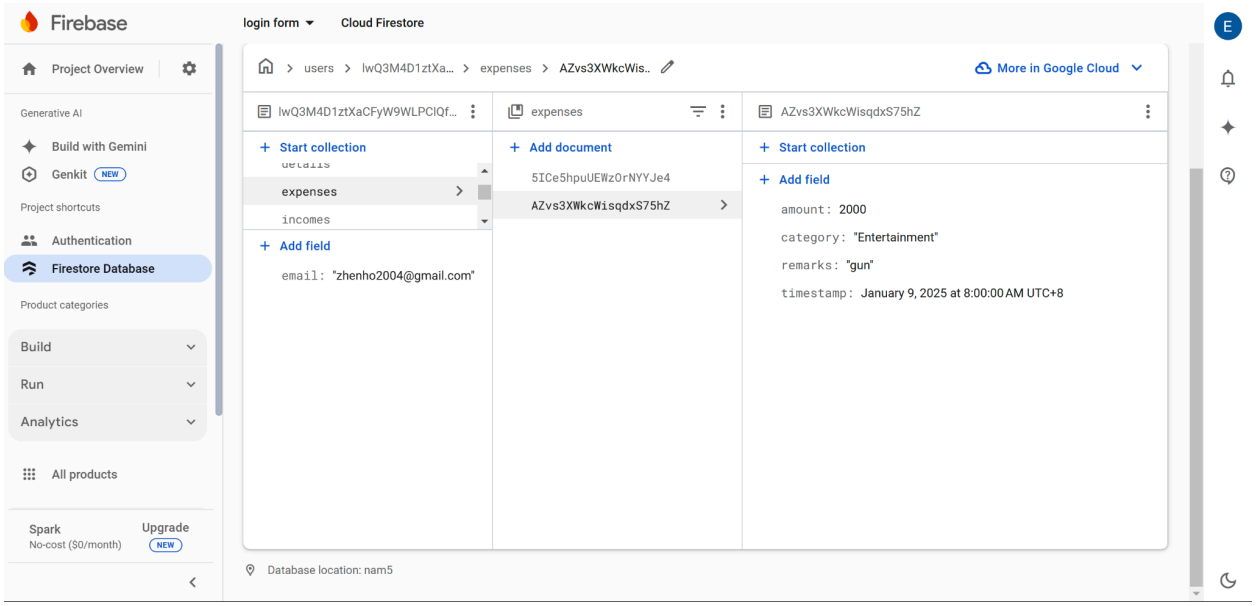
The Cloud Firestore interface displays a user's budget records stored in a sub-collection under their unique user ID. Each budget entry is represented as a document corresponding to a specific month and year. In this case, the document labeled **"2025-1"** represents the budget for January 2025. The budget amount for this month is recorded as **60**. This structured format allows users to efficiently track and manage their monthly budgets, ensuring organized financial data storage and retrieval.



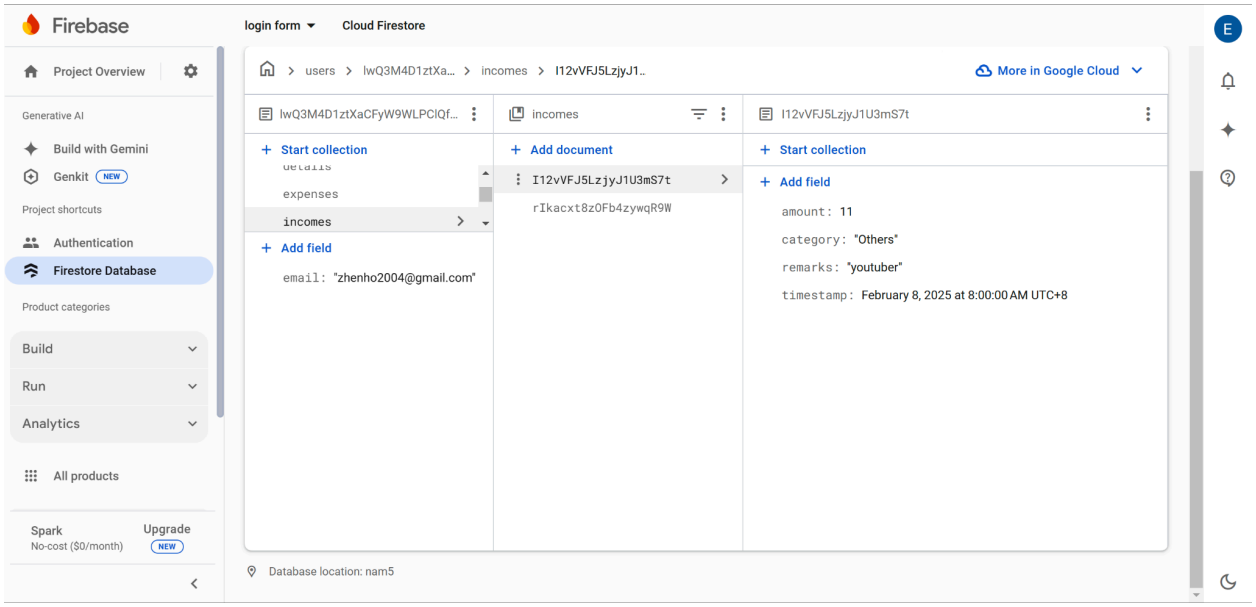
The Cloud Firestore interface displays a user's personal information stored in a structured format. The data includes fields such as address, age, date of birth, gender, name, and occupation. This information is organized within a collection under the user's unique ID, making it easy to manage and retrieve personal details for authentication and personalization purposes.



The Cloud Firestore interface displays a user's expense record, organized within a collection under their unique ID. The record includes information such as the amount spent, expense category, remarks, and a timestamp. This structured data storage allows efficient tracking and retrieval of financial transactions, enabling users to manage and analyze their expenses effectively.



The Cloud Firestore interface displays a user's income record stored under their unique ID within the "incomes" collection. The record includes details such as the income amount, category, remarks, and timestamp. This structured data storage allows users to efficiently track and manage their income sources, facilitating better financial organization and analysis.



6 Testing

6.1 Testing Strategy

<TO DO: Describe the integration testing strategy here.>

Objective:Ensure the seamless integration of all system components in the Personal Budget Tracker web application, verifying that user interactions, financial data management, and role-based functionalities work cohesively without errors. The strategy aims to validate data consistency, system security, and usability across different user roles.

Scope:
Integration testing will cover the following key interactions between system components:

1.User Authentication & Access Control:

- Verify that users, admins, financial advisors, and guests can access only their respective functionalities.
- Ensure session management is secure, preventing unauthorized access.

2.Financial Data Management:

- Validate that users can input, edit, and categorize income, expenses, and savings.
- Ensure data is correctly processed, stored, and retrieved when generating reports.

3.Budget Tracking & Goal Setting:

- Check the correct implementation of budget creation and monitoring.
- Verify notifications and alerts related to budget deviations.

4.Financial Advisor Interaction:

- Ensure financial advisors can securely access and review user financial data.
- Validate the system's ability to store and display advisor recommendations.

5.Admin Functions:

- Confirm that admins can manage user accounts, generate system reports, and monitor financial data integrity.
- Ensure data modifications by admins are logged and do not disrupt system performance.

6.Guest User Experience:

- Verify that guests can interact with limited features but cannot save data or access premium functionalities.
- Ensure a smooth transition from guest mode to registered user mode.

7.Data Flow & System Performance:

- Test database interactions to prevent data loss or duplication.
- Verify system load performance when handling multiple concurrent users.

Testing Type: Integration Testing
1.Top-Down Approach: Test high-level functionalities first (e.g., user authentication, data processing) before moving to individual feature validation.
2.Bottom-Up Approach: Verify individual modules such as budget calculations, financial reports, and advisory tools before integrating them into the main system.
3.Regression Testing: Ensure new updates do not break existing functionalities.
4.API Testing: Validate interactions between frontend, backend, and third-party financial tools (if integrated).

6.2 Test Data

<TO DO: Provide a detailed description of the test data used to verify the system here.>

6.2.1 Admin Test Data

6.2.1.1 Login as admin

Actions	Input	Expected Output	Pass/Fail
Admin login with valid credentials	Correct Email and Password	Go to admin dashboard	Pass
Admin login with invalid credentials	Incorrect Email and Password	“Invalid email or password. Please try again.”	Pass

6.2.1.2 Manage expense categories

Actions	Input	Expected Output	Pass/Fail
Admin input new expense category name and click add category	New expense category name	“Expense category added successfully!”	Pass
Admin did not input new expense category name and click add category	Did not input new expense category name	“Please enter a category name.”	Pass
Admin edit expense category name	New expense category name	“Expense category updated successfully!”	Pass
Admin delete expense category	click “OK”	“Expense category deleted successfully!”	Pass

6.2.1.3 Manage income categories

Actions	Input	Expected Output	Pass/Fail
Admin input new income category name and click add category	New income category name	“Expense category added successfully!”	Pass
Admin did not input new income category name and click add category	Did not input new income category name	“Please enter a category name.”	Pass
Admin edit income category name	New expense category name	“Income category updated successfully!”	Pass
Admin delete Income category	click “OK”	“Income category deleted successfully!”	Pass

6.2.2 User Test Data

6.2.2.1 Login

Actions	Input	Expected Output	Pass/Fail
User login with valid credentials	Correct Email and Password	“Login Successful” and will go to user dashboard	Pass
User login with invalid credentials	Incorrect Email and Password	“An error occurred during login.”	Pass

6.2.2.2 Recovery Password

Actions	Input	Expected Output	Pass/Fail
User enter email	Correct email	Password reset email sent. Check your inbox!	Pass
User enter email	Incorrect email	Please include an '@'in the email address 'xxx'is missing an '@'"	Pass

6.2.2.3 Edit Income or Expense

Actions	Input	Expected Output	Pass/Fail
User edit income or expense	Enter correct Income or expense detail	Display "History updated successfully!" and History for Income and Expense will be updated.	Pass
User edit income or expense	Missing enter some income or expense detail	Please fill out this field	Pass
User edit income or expense	Enter Incorrect income or expense detail	Please enter a valid value	Pass

6.2.2.4 Delete Income or Expense

Actions	Input	Expected Output	Pass/Fail
User choose income or expense to delete	Click ‘OK’ after choose for deleted income or expense	History for the selected Income or Expense will be deleted.	Pass

6.2.2.5 Add Income

Actions	Input	Expected Output	Pass/Fail
User enter income	Enter correct Income	Display “Income has been successfully added!” and History for Income will be displayed.	Pass
User enter income	Missing enter some income detail	Please fill out this field	Pass

6.2.2.6 Add Expense

Actions	Input	Expected Output	Pass/Fail
User enter expense	Enter correct Expense	Display “Expense has been successfully added!” and History for Expense will be displayed.	Pass
User enter expense	Missing enter some expense detail	Please fill out this field	Pass

6.2.2.7 Set Monthly Budget

Actions	Input	Expected Output	Pass/Fail
User enter Monthly Budget	Enter correct Monthly Budget	Display "Budget for M-YYYY successfully set to $xxxx" and Monthly Budget will be displayed.	Pass
User enter Monthly Budget	Missing enter Monthly Budget	Please enter a valid budget amount.	Pass
User enter Monthly Budget	Enter wrong Monthly Budget	Please enter a valid budget amount.	Pass

6.2.2.8 Edit Profile

Actions	Input	Expected Output	Pass/Fail
User enter User Detail in Profile	Enter correct User Details	Display "Data saved successfully!" and User Details will be displayed.	Pass
User enter User Detail in Profile	Missing enter User Details	Please fill out all the fields.	Pass

6.2.3 Guest Test Data

6.2.3.1 Sign Up

Actions	Input	Expected Output	Pass/Fail
Guest enter email and password	Enter correct email and 6 digit above of password	Account Created Successfully	Pass
Guest enter registered email	Enter repeated email	Email Address Already Exists !!!	Pass
Guest enter incorrect email and incorrect password	Please include an '@'in the email address 'xxx'is missing an '@', enter less than 6 digit password	"Unable to create User."	Pass

6.2.3.2 Add Income (GUEST)

Actions	Input	Expected Output	Pass/Fail
Guest enter income	Enter correct Income	Display "Income has been successfully added!" and History for Income will be displayed.	Pass
Guest enter income	Missing enter some income detail	Please fill out this field	Pass

6.2.3.3 Add Expense (GUEST)

Actions	Input	Expected Output	Pass/Fail
Guest enter expense	Enter correct Expense	Display “Expense has been successfully added!” and History for Expense will be displayed.	Pass
Guest enter expense	Missing enter some expense detail	Please fill out this field	Pass

6.2.3.4 Edit Income or Expense (GUEST)

Actions	Input	Expected Output	Pass/Fail
Guest edit income or expense	Enter correct Income or expense detail	History for Income and Expense will be updated.	Pass

6.2.3.5 Delete Income or Expense (GUEST)

Actions	Input	Expected Output	Pass/Fail
Guest choose income or expense to delete	Click ‘OK’ after choose for deleted income or expense	History for the selected Income or Expense will be deleted.	Pass

6.2.4 Advisor Test Data

6.2.4.1 Advisor Login

Actions	Input	Expected Output	Pass/Fail
Advisor enter correct email	Enter correct email	“Login Successful” and will go to user dashboard	Pass
Advisor enter incorrect email	Enter incorrect email	“An error occurred during login.”	Pass

6.2.4.2 Send Message

Actions	Input	Expected Output	Pass/Fail
Database find user list	Choose from drop-down menu	Able to see list of user	Pass
Send Message	Write anything	Message sent to “user@gmail.com”	Pass

6.2.4.3 Generate Report

Actions	Input	Expected Output	Pass/Fail
Upload txt file	Text file containing data	Report.pdf generated	Pass

6.2.4.4 Generate Graph

Actions	Input	Expected Output	Pass/Fail
Upload txt file	Text file containing data	Graph generated	Pass

6.2.4.5 Publish

Actions	Input	Expected Output	Pass/Fail
Post Message	Write anything (word or numbers)	Post successfully	Pass
Post Image	Upload png file	Post Successfully	Pass

6.3 Acceptance Testing

<TO DO: Prepare the acceptance test for each team member>

6.3.1 Admin Acceptance Testing

Criteria	Fulfilled?	Remarks
Admin login with valid credentials	Yes	
Admin login with invalid credentials	Yes	
Add,edit,delete expense categories	Yes	
Add,edit,delete income categories	Yes	
View financial expense and income category statistics by chart	Yes	
View expense and income of "Others" category description	Yes	
View user details	Yes	

Date tested : 8/2/2025

% Complete : 100%

Tested by : Gan Shao Yang

Verified by :Chin Zhen Ho

6.3.2 User Acceptance Testing

Criteria	Fulfilled?	Remarks
User login with valid credentials	Yes	
User login with invalid credentials	Yes	
User enter correct email for password recovery	Yes	
User enter incorrect format email for password recovery	Yes	Can be done for more detail when checking email in the database.

User can view transaction History correctly	Yes	Can improve for auto refresh the Transaction History.
User can edit the Transaction History	Yes	Can improve the category load back as drop down menu like add income or expense
User can delete the Transaction History	Yes	
User can add Income	Yes	
User can add Expense	Yes	
User can edit Profile	Yes	
User can view the Bar chart for total income and total expense separately	Yes	
User can set Monthly Budget	Yes	
User can view Remaining Monthly	Yes	

Budget in Pie chart form		
User will receive alert message when budget remaining 10%	Yes	
User can view message send by the Financial Advisor	Yes	
User can view Public Dashboard	Yes	

Date tested : 8/2/2025

% Complete : 100%

Tested by : BERNARD RYAN SIM KANG XUAN

Verified by : ERIC TEOH WEI XIANG

6.3.3 Financial Advisor

Criteria	Fulfilled	Remarks
Login	Yes	Could be better if there was more login credential
Send Message	Yes	Would be better if it could upload an image or text file, but since it is locked behind a paywall there is no choice.
Generate Report	Yes	Need to be a bit more user-friendly
Generate Graph	Yes	
View Public Board	Yes	
Publish stuff, edit and delete	Yes	

Date tested :*8/2/2025*

% Complete : 100%

Tested by : Chin Zhen Ho

Verified by : Gan Shao Yang

6.3.4 Guest Acceptance Testing

Criteria	Fulfilled?	Remarks
Guest sign up with unregistered email and password	Yes	
Guest sign up with registered email and password	Yes	
Guest sign up with incorrect email and incorrect password	Yes	
Guest can view transaction History correctly	Yes	Can improve for auto refresh the Transaction History.
Guest can edit the Transaction History	Yes	Can improve the category load back as drop down menu like add income or expense
Guest can delete the Transaction History	Yes	
Guest can add Income	Yes	

Guest can add Expense	Yes	
Guest can view the Bar chart for total income and total expense	Yes	

Date tested :8/2/2025

% Complete : 100%

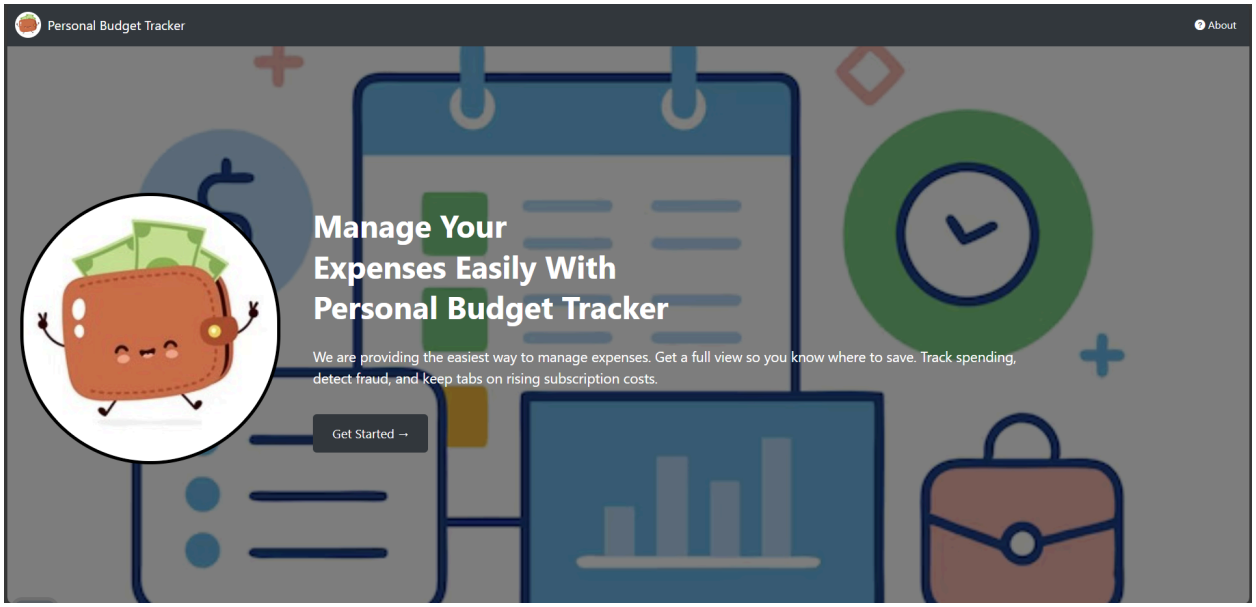
Tested by : Eric Teoh Wei Xiang

Verified by : Bernard Ryan Sim Kang Xuan

7 Sample Screens

7.1 Main Screen

<TO DO: Describe and place the main screens of the system here.>

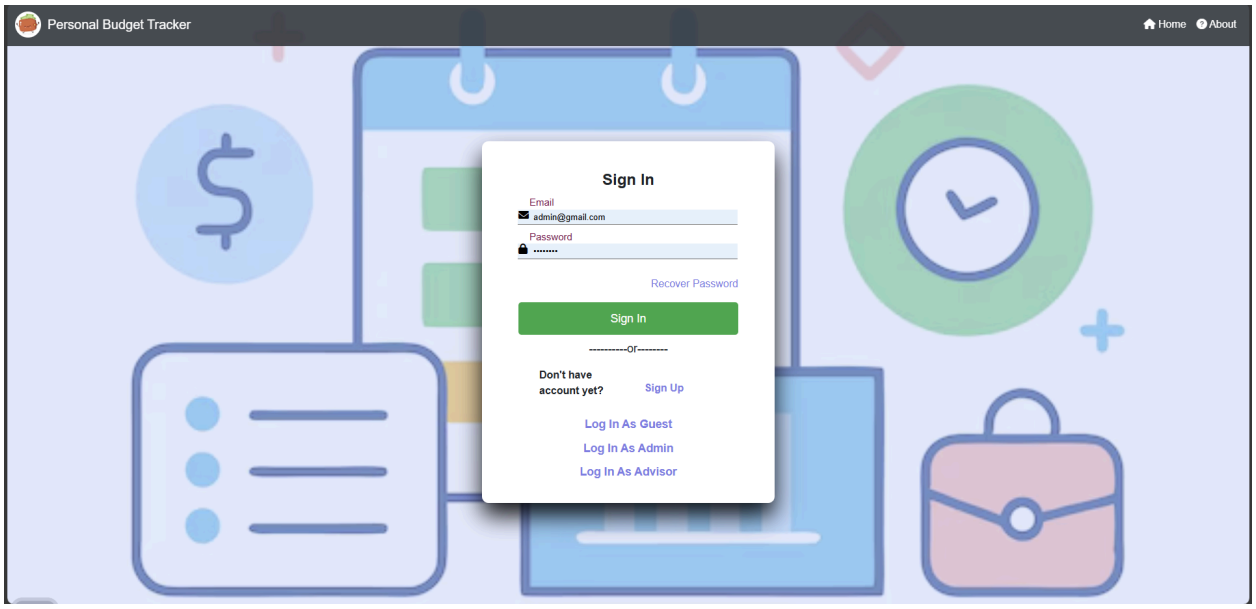


This is welcome page

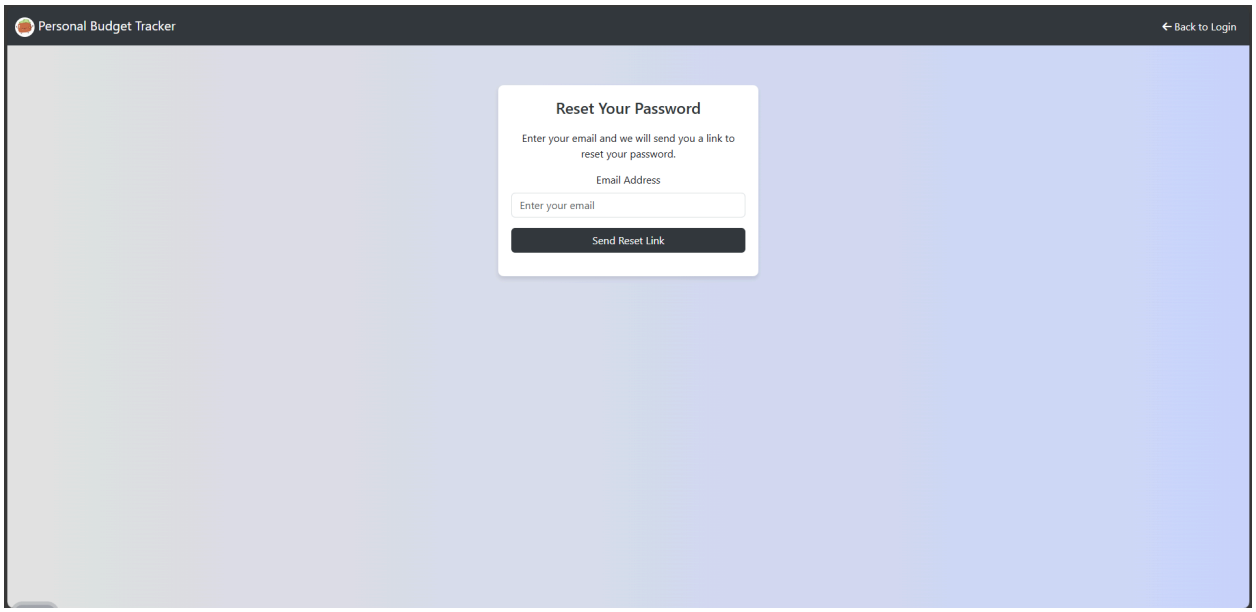
This is about page.

7.1.1 Subsystem User Subscreen

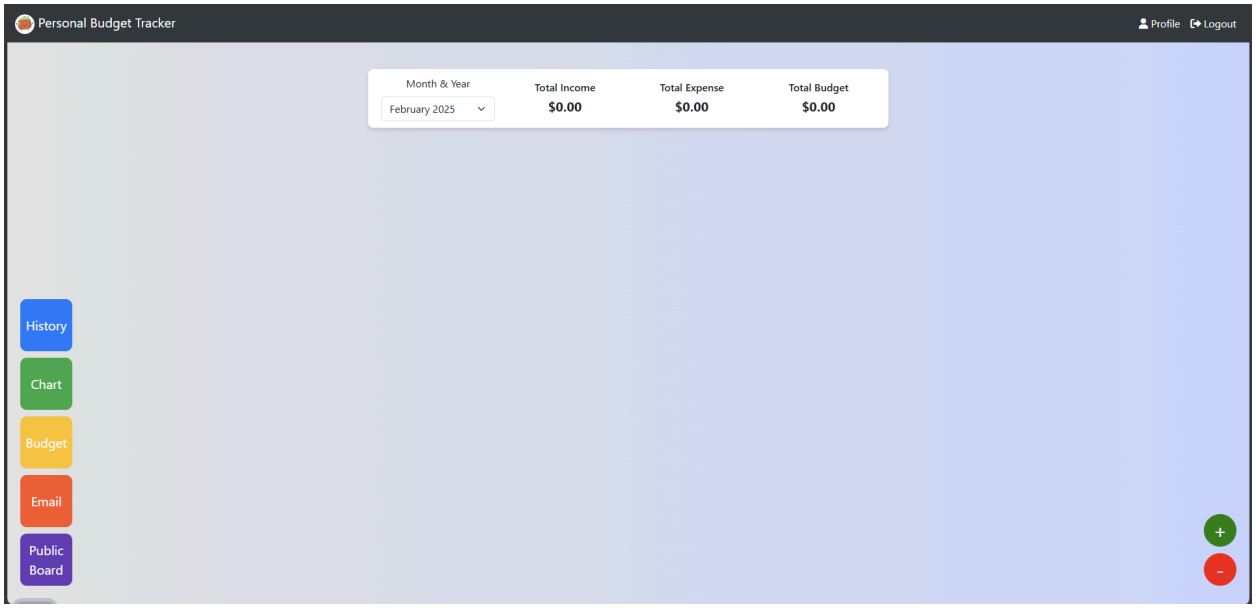
<TO DO: Describe and place the screens of subsystem 1 here.>



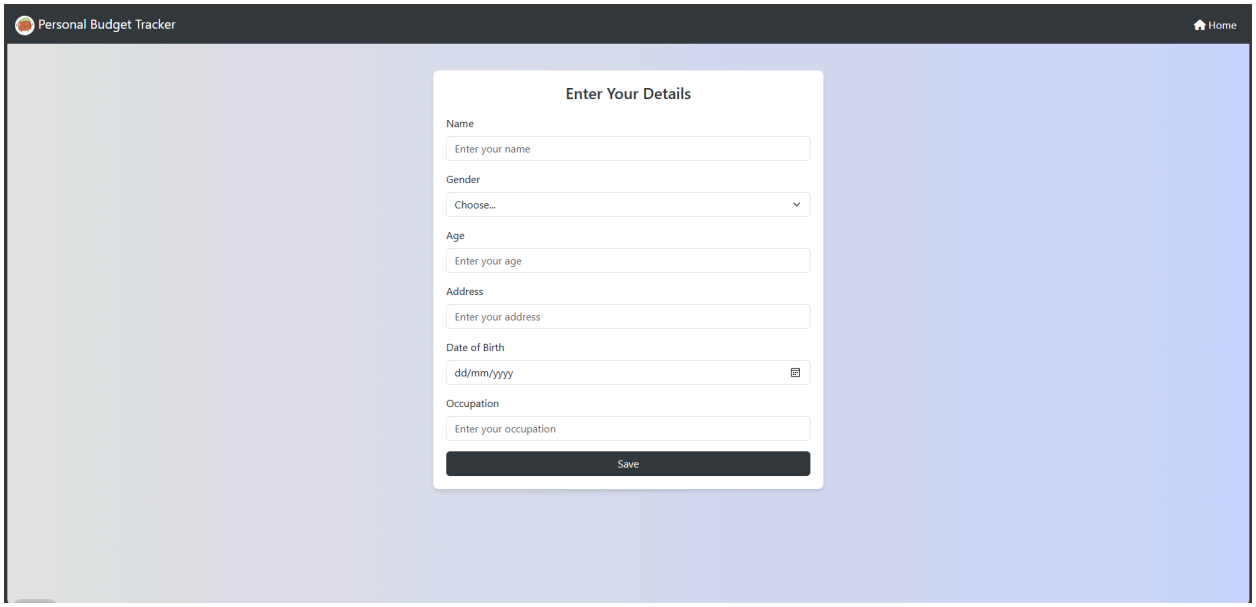
This is user login page.



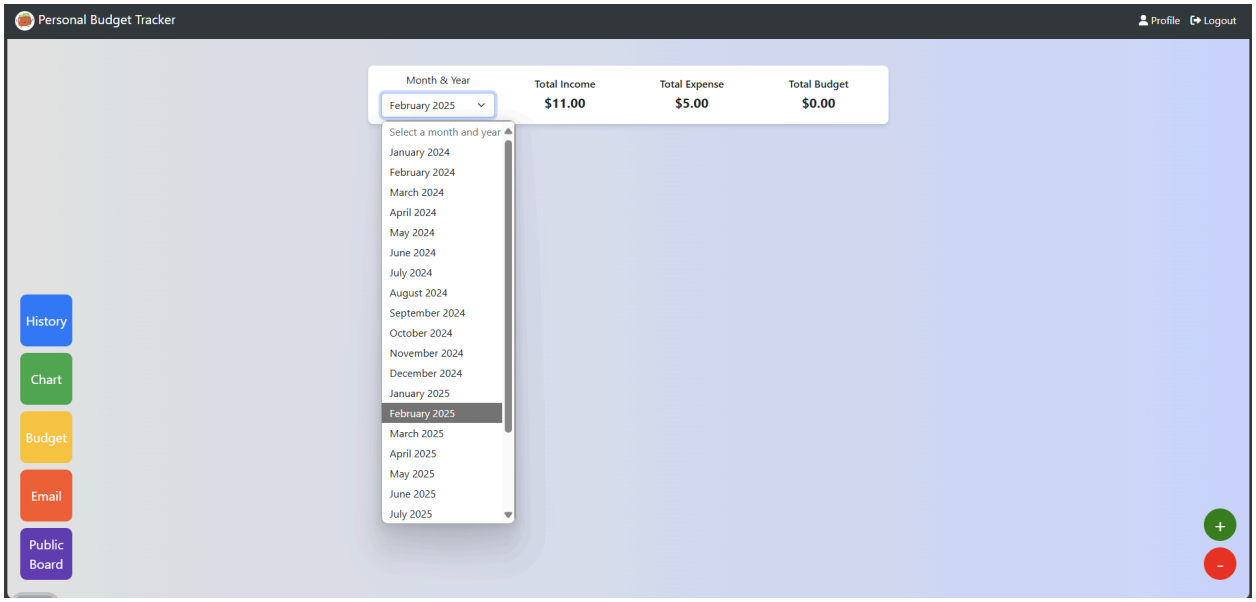
This is password recovery page.



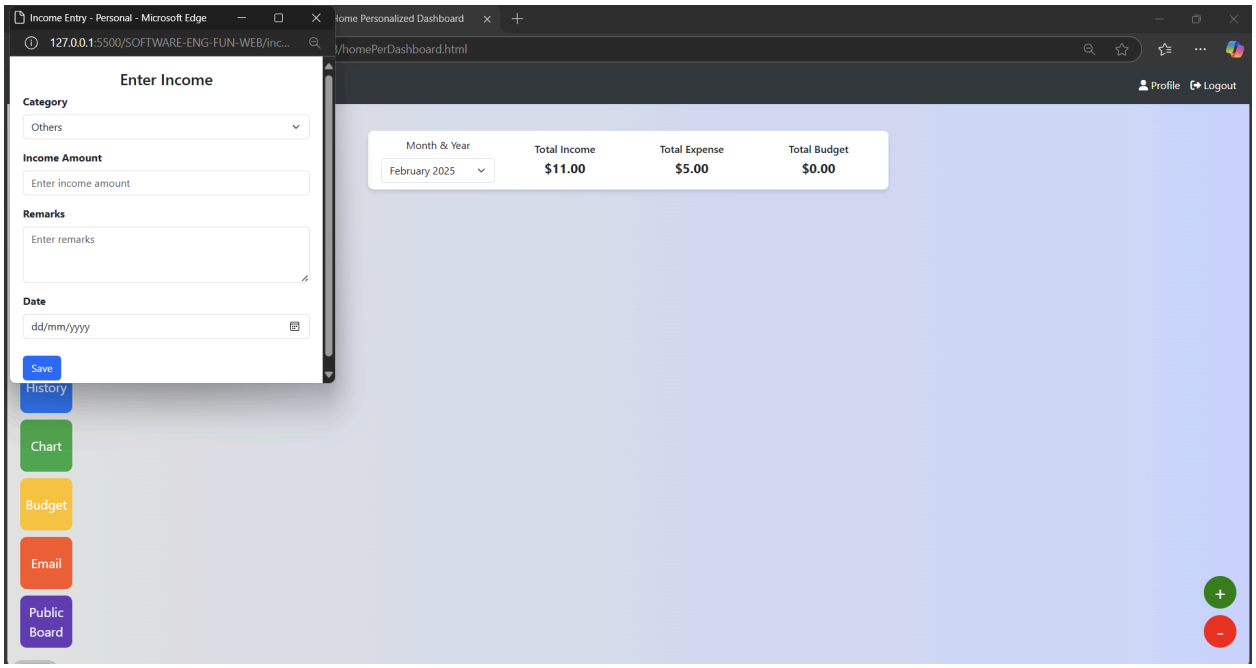
This is user dashboard.



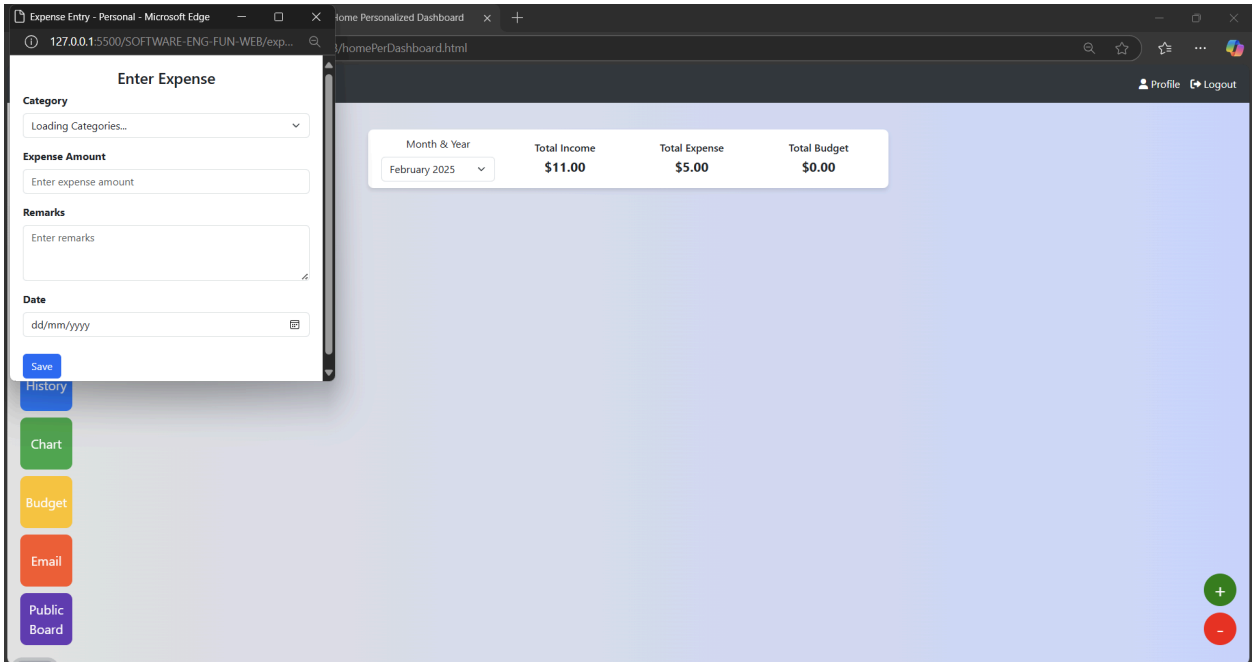
This is user profile page.



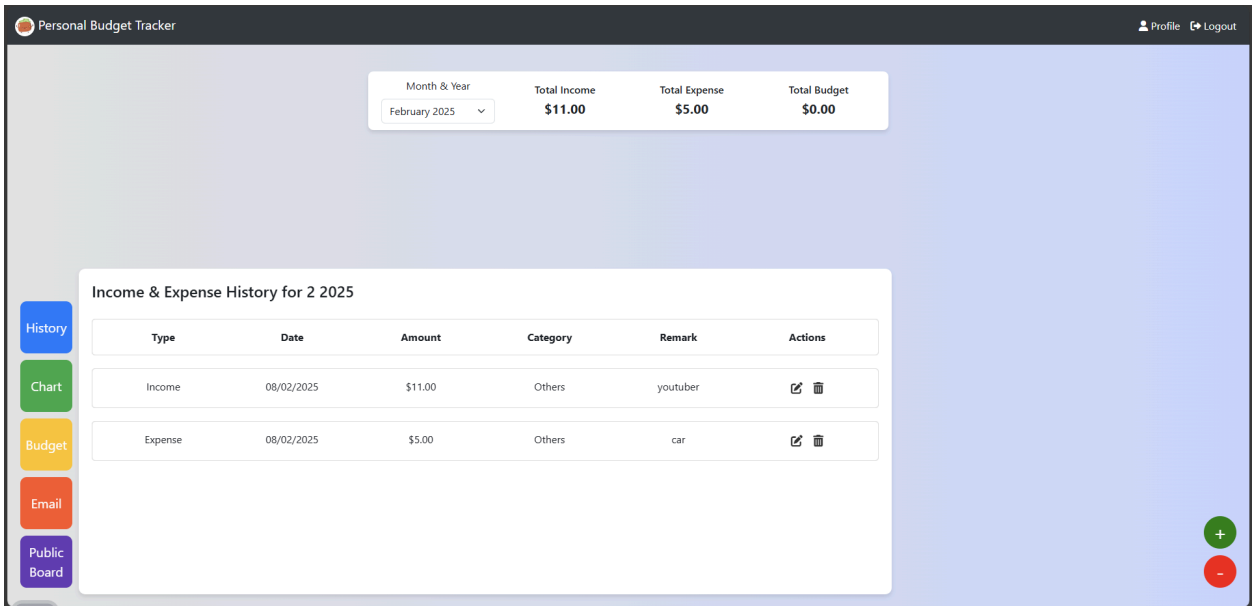
This can choose the specific history transaction month and year.



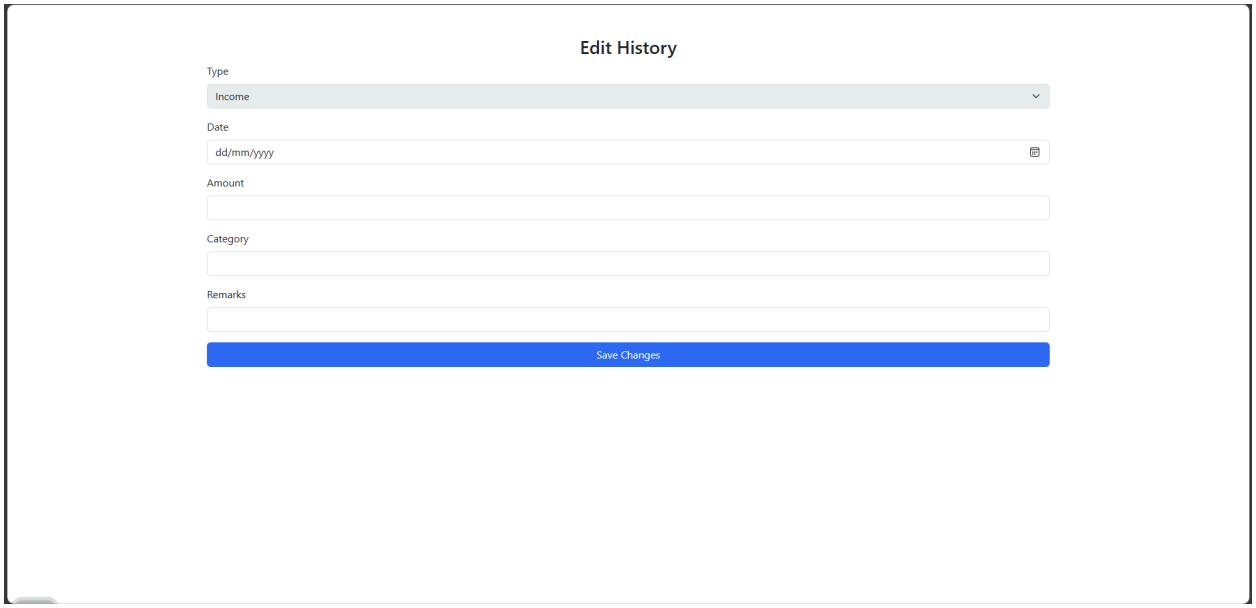
This “+” button can let users record income.



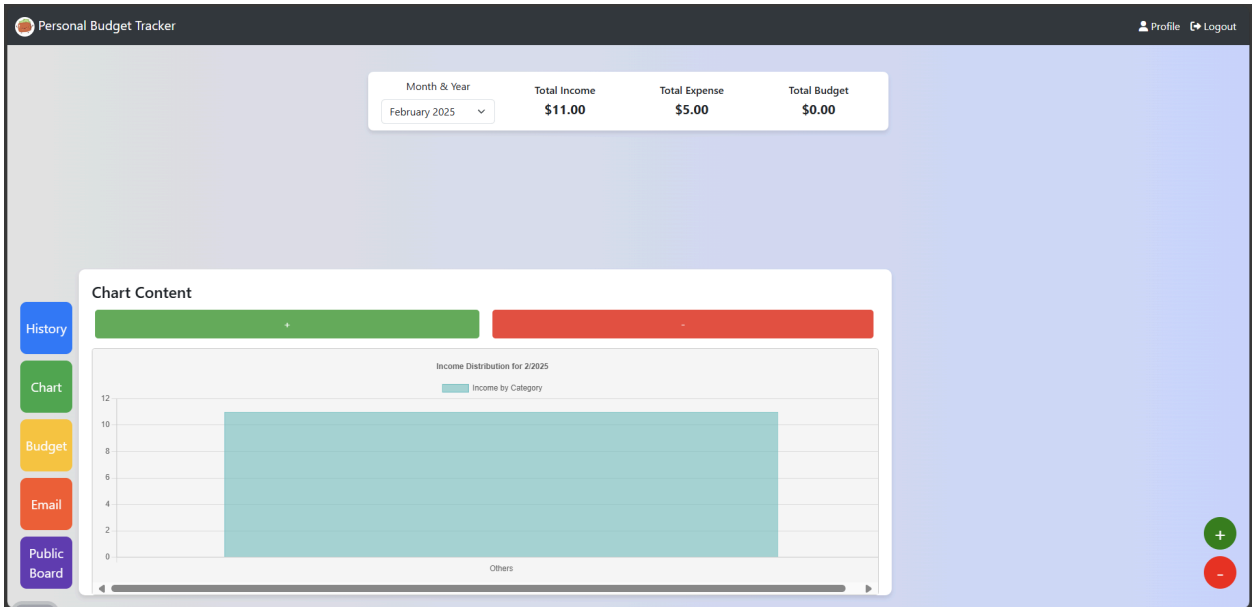
This “-” button can let users record expenses.



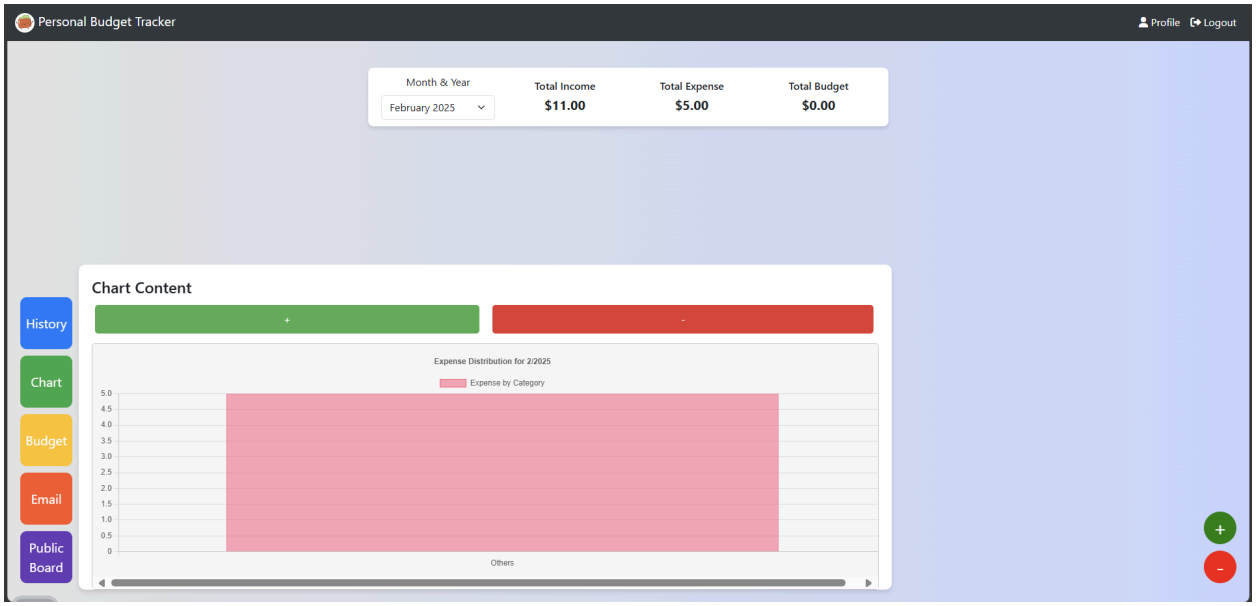
When you click “History” button,it will show the income and expenses history of that specific month and year you choose.



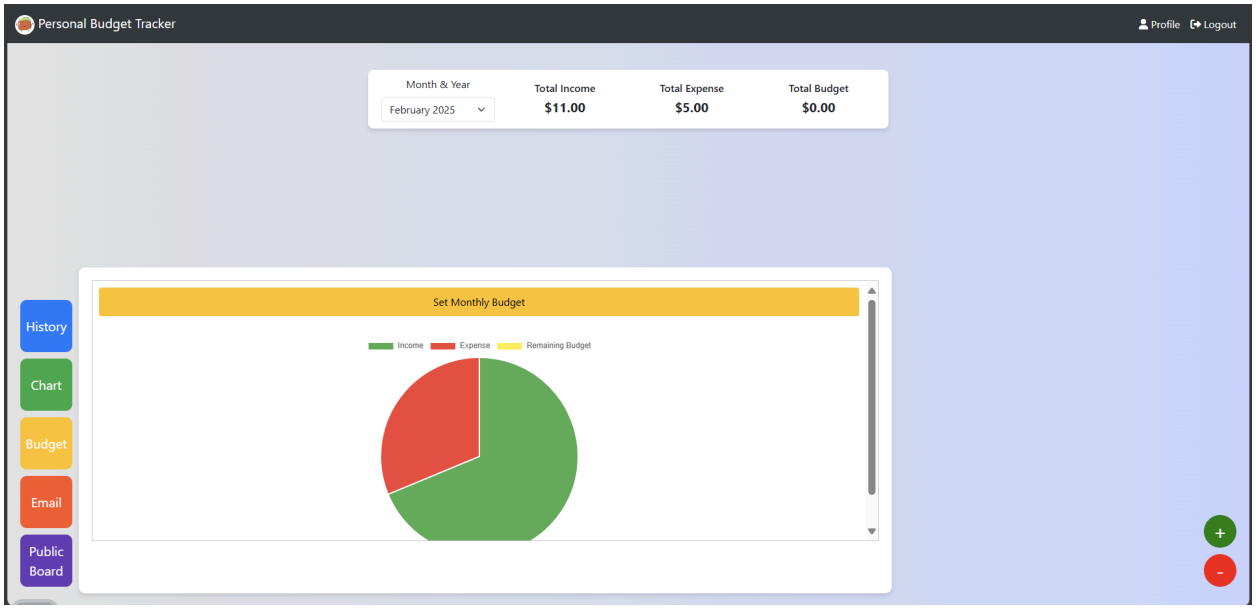
Edit history page.



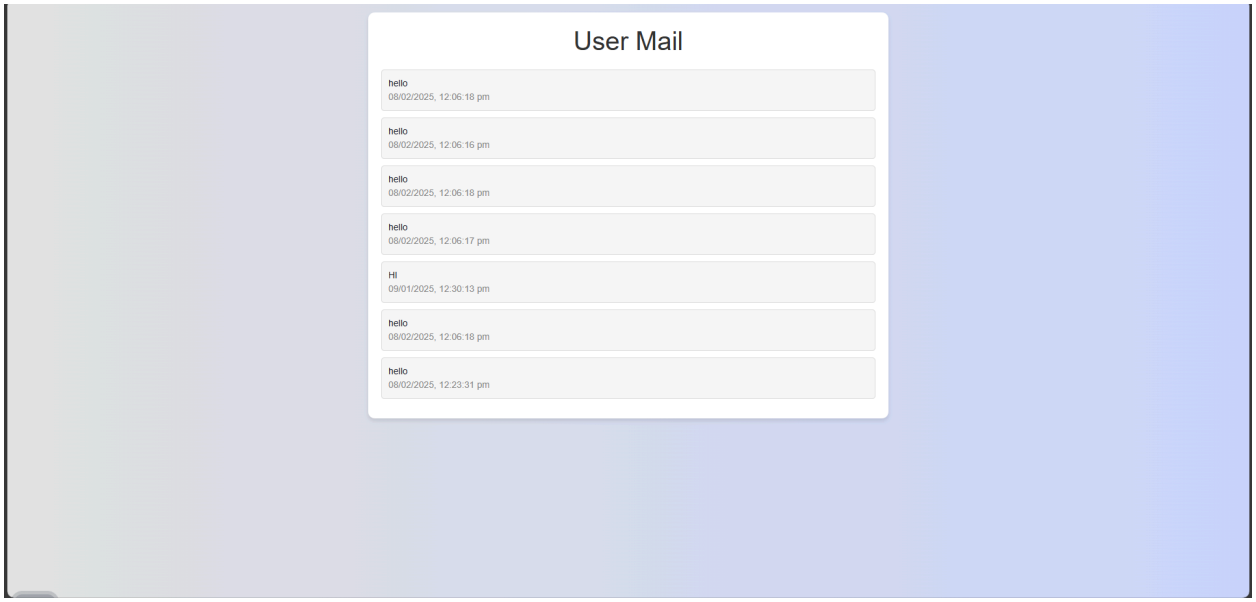
when click “Chart” button,it will show income category chart



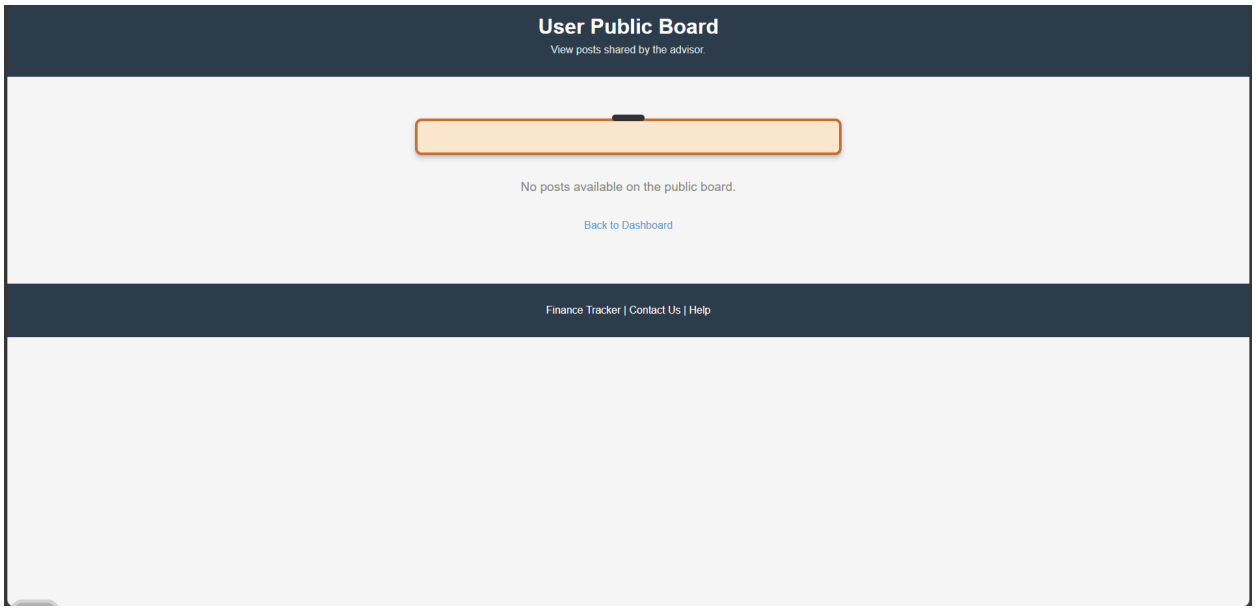
when click “Chart” button,it will show expense category chart



When click “Budget” button



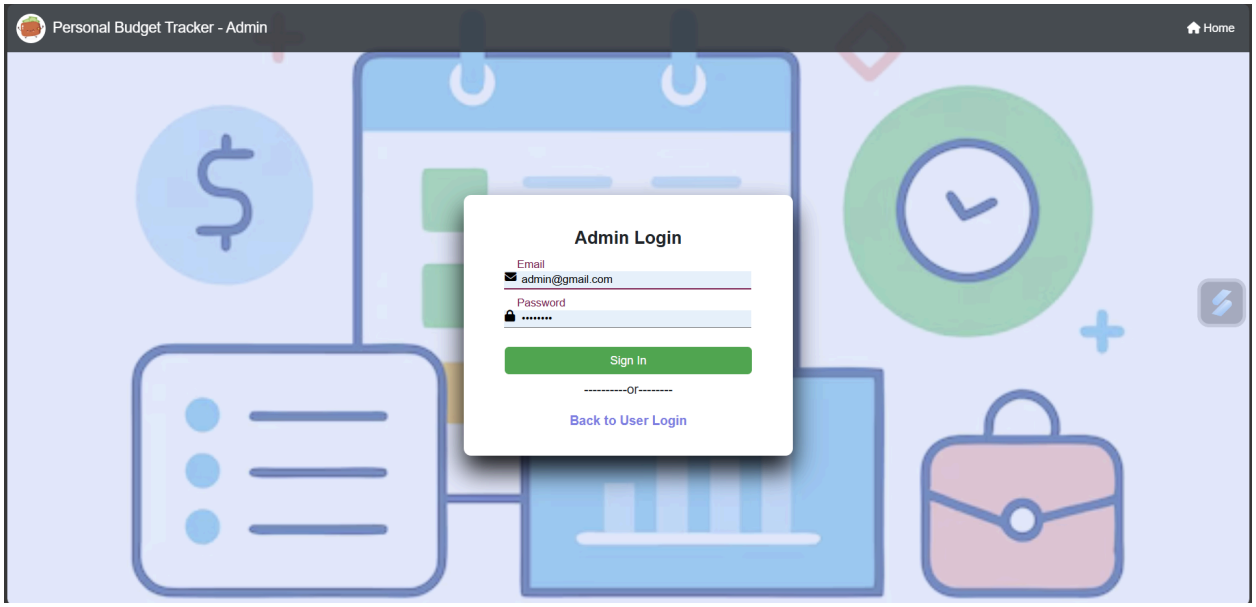
When click “Email” button will go to user mail page.



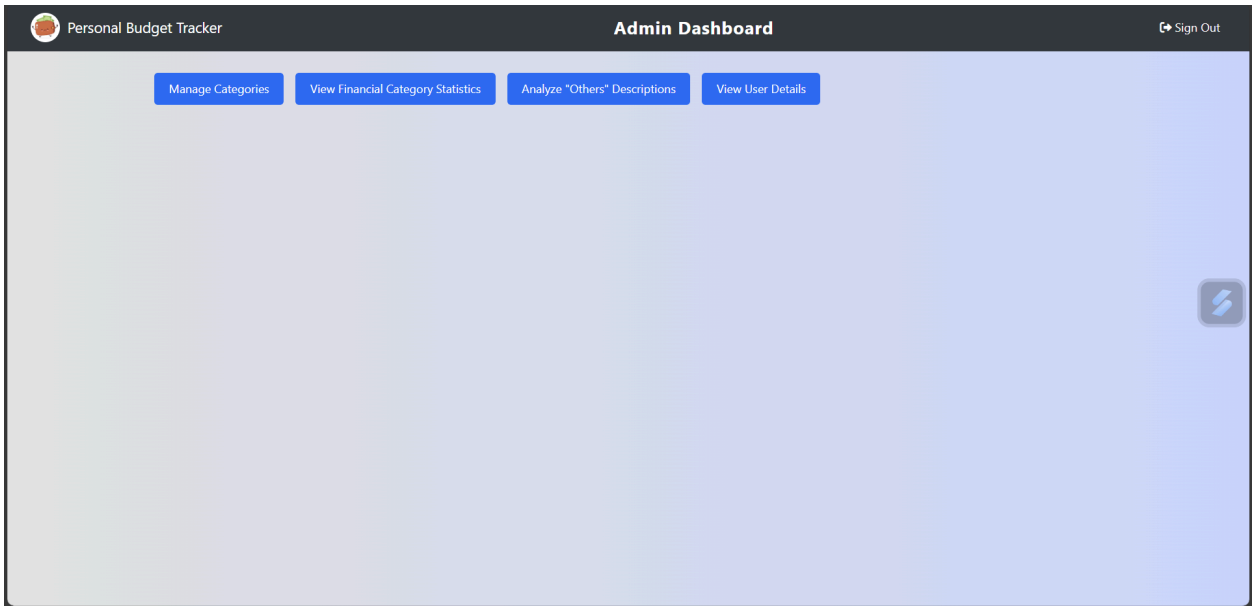
when click “Public Board” button will go to user public board page.

7.1.2 Subsystem Admin Subscreen

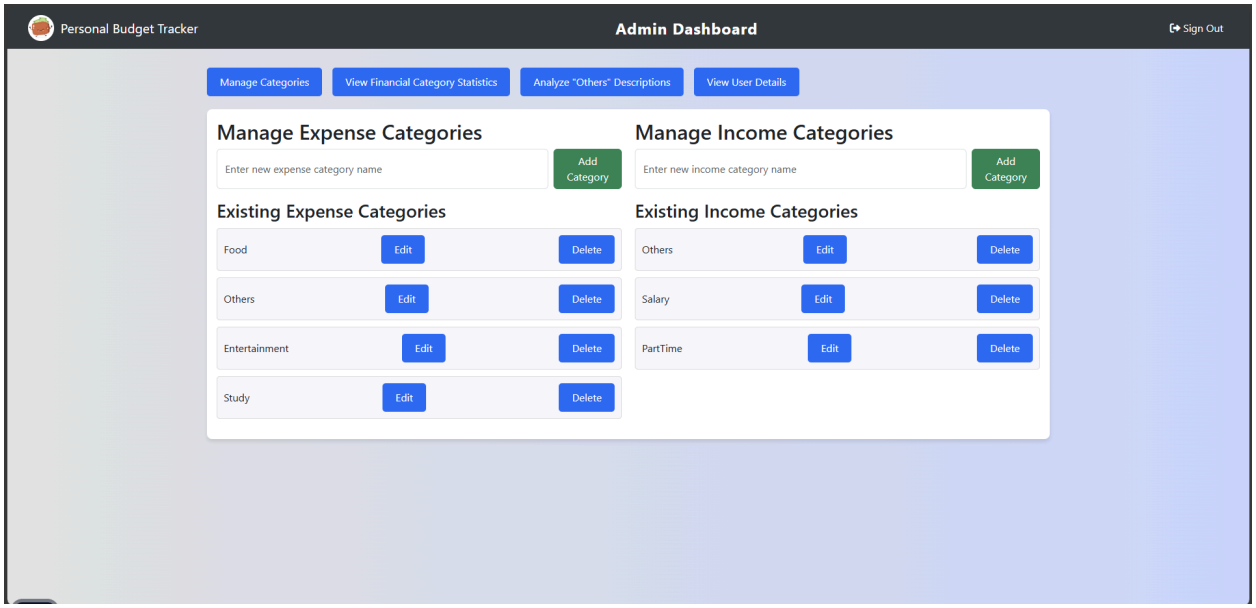
<TO DO: Describe and the screens of subsystem 2 here.>



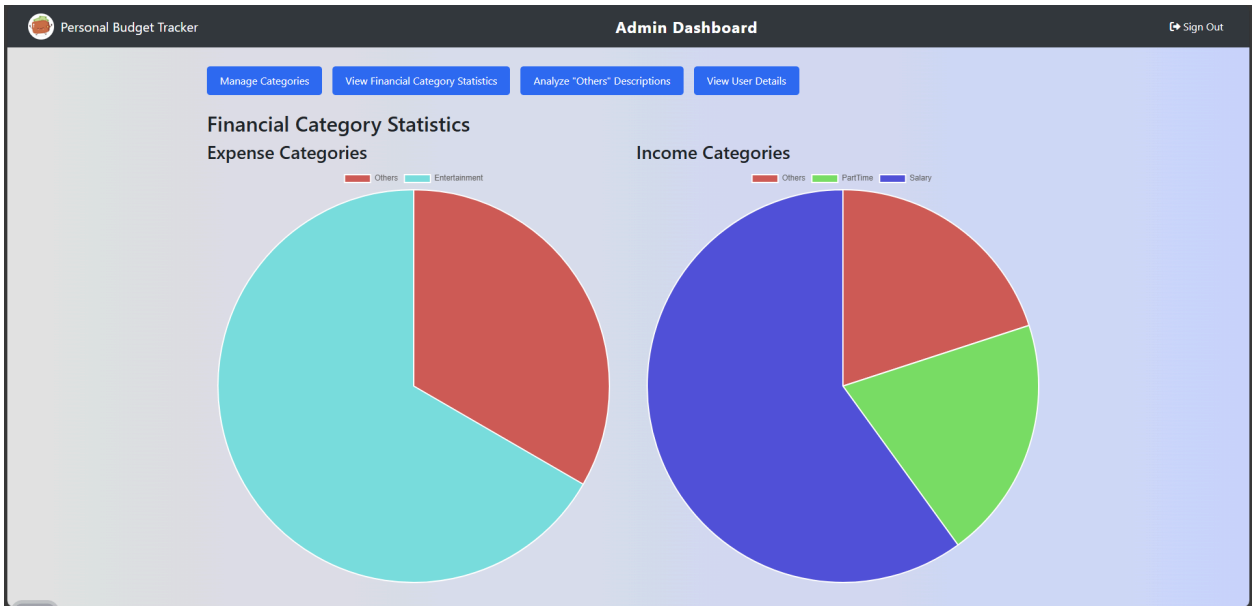
This is login as admin page



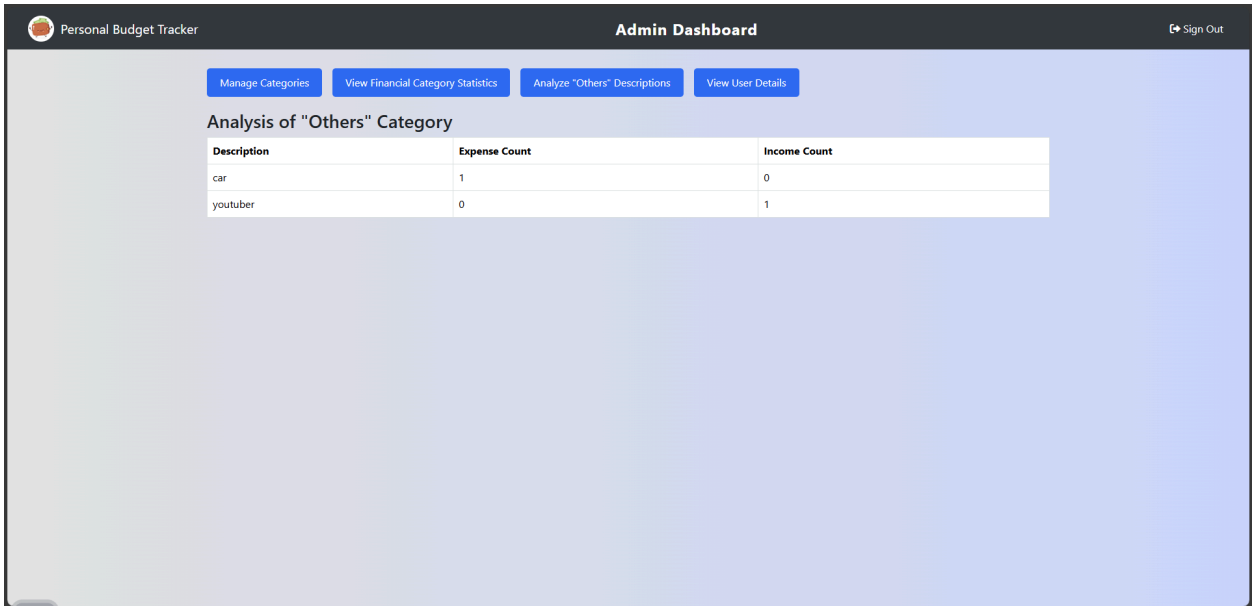
This is admin dashboard.



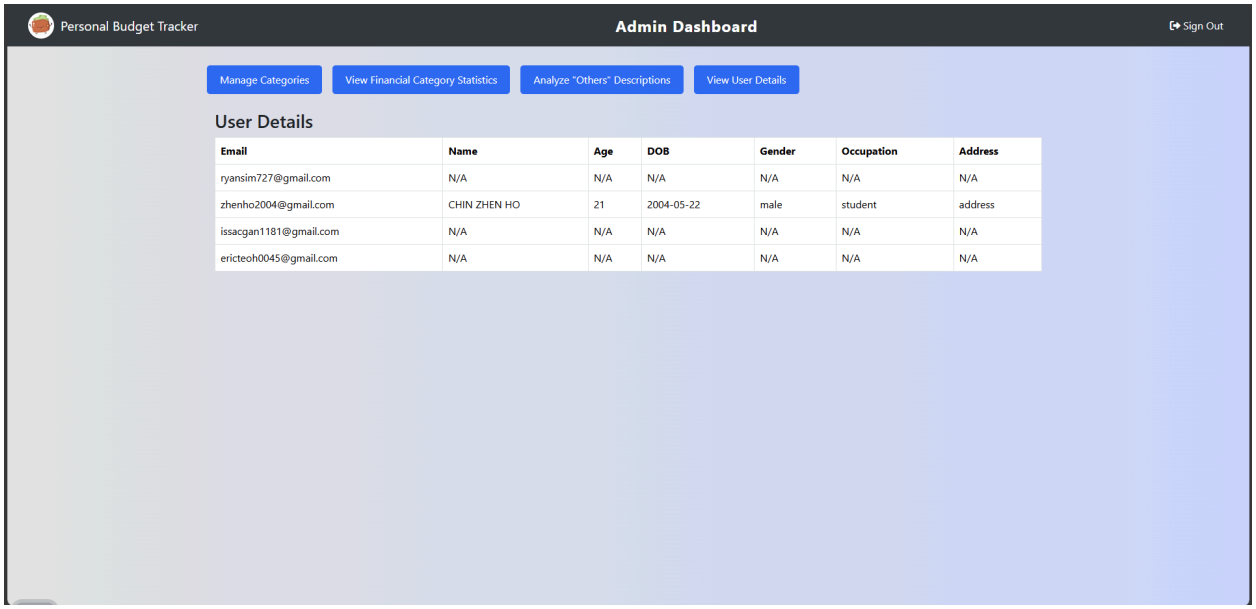
When clicking the “Manage Categories” button ,you can manage expense categories and manage income categories.



When clicking the “View Financial Category Statistics” button,you can view expense categories statistics and income categories statistics that show how many users have chosen that category.

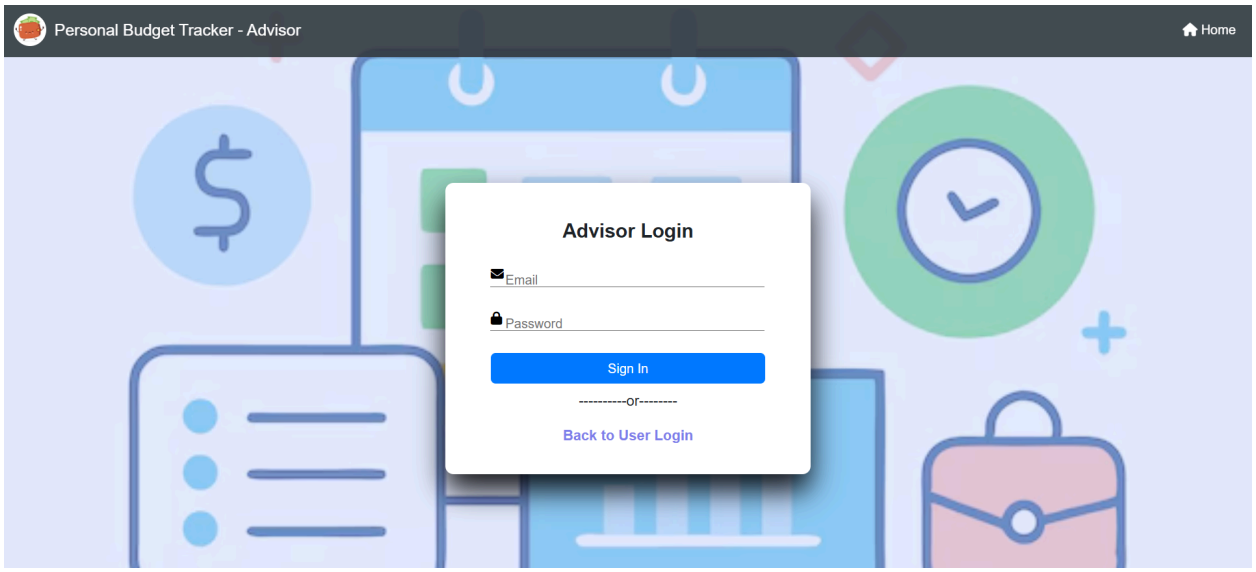


When clicking the “Analyze "Others" Descriptions” button,you can view all the "Others" category descriptions written by the user.

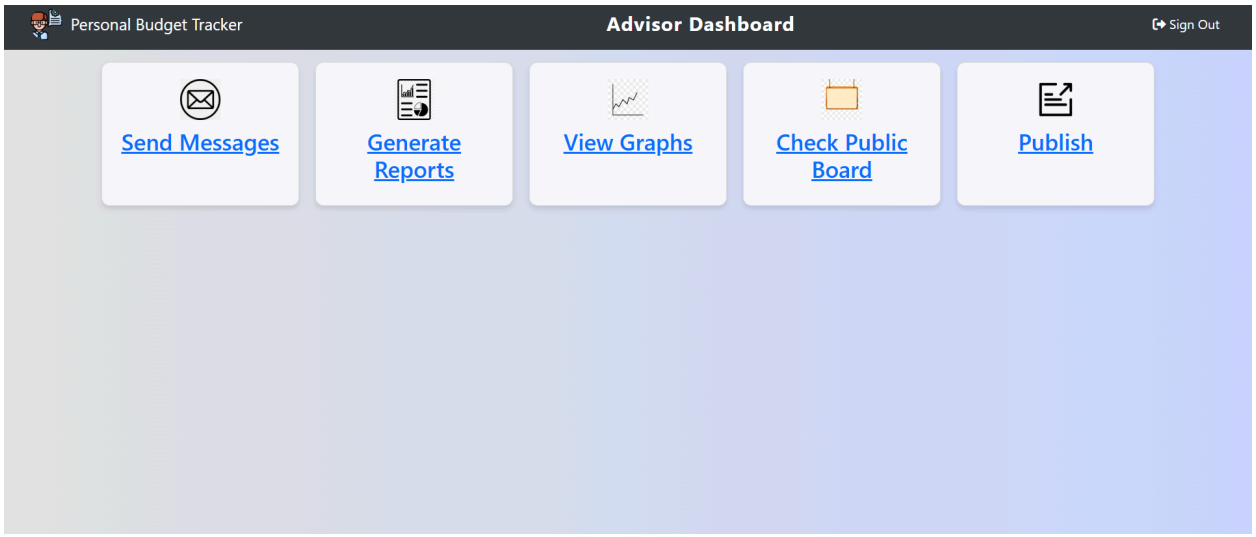


When click the “View User Details” button,you can view all the user details such as Email, Name,Age,Date of Birth,Gender,Occupation,and Address.

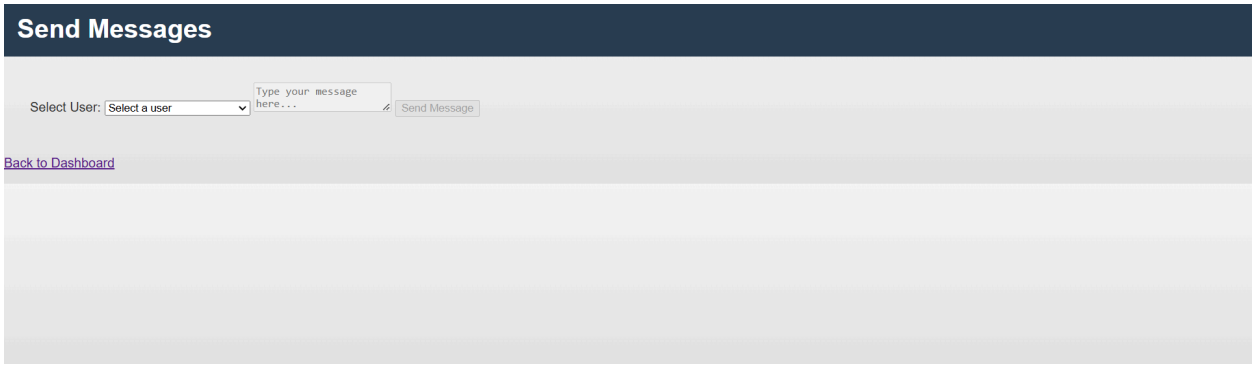
7.1.3 Subsystem Financial Advisor Subscreen



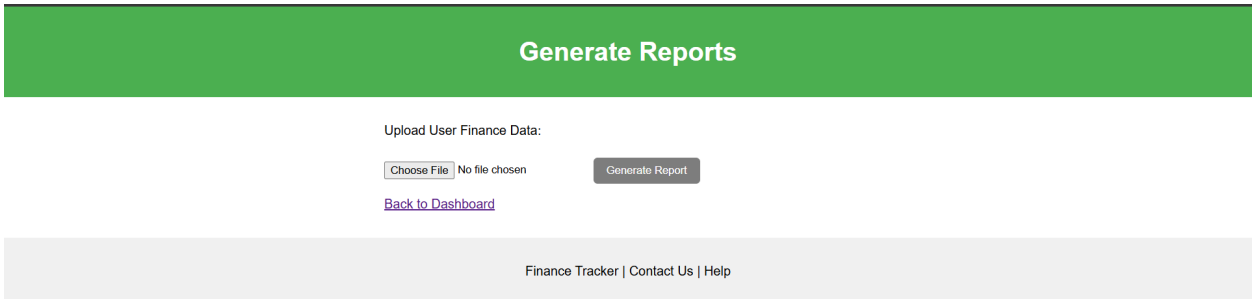
Adviser Login Screen



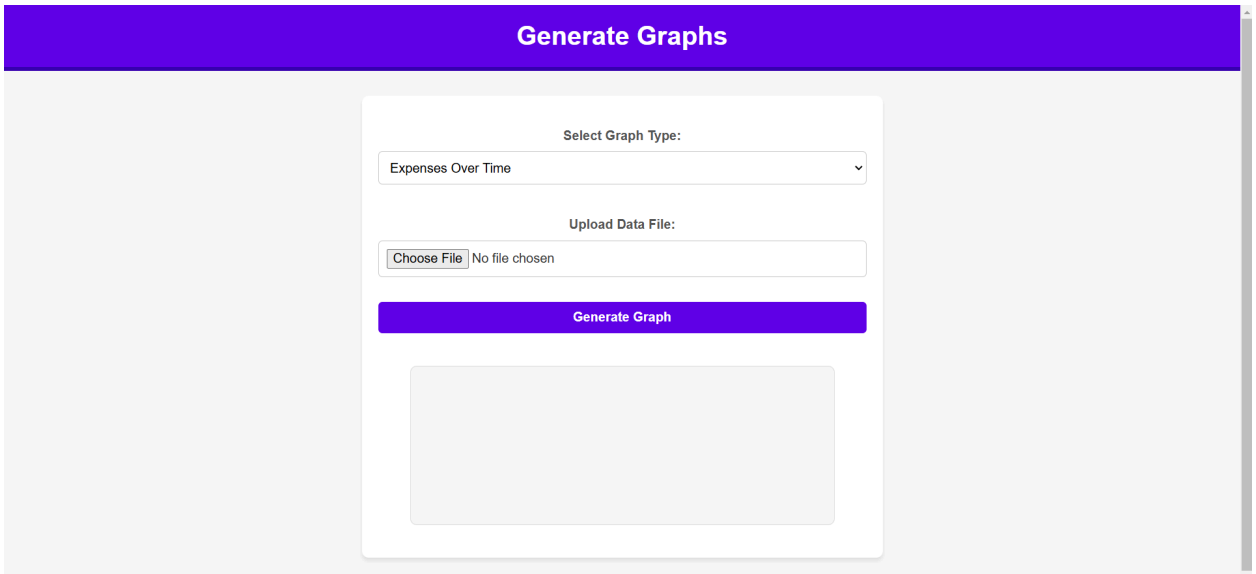
Advisor Dashboard



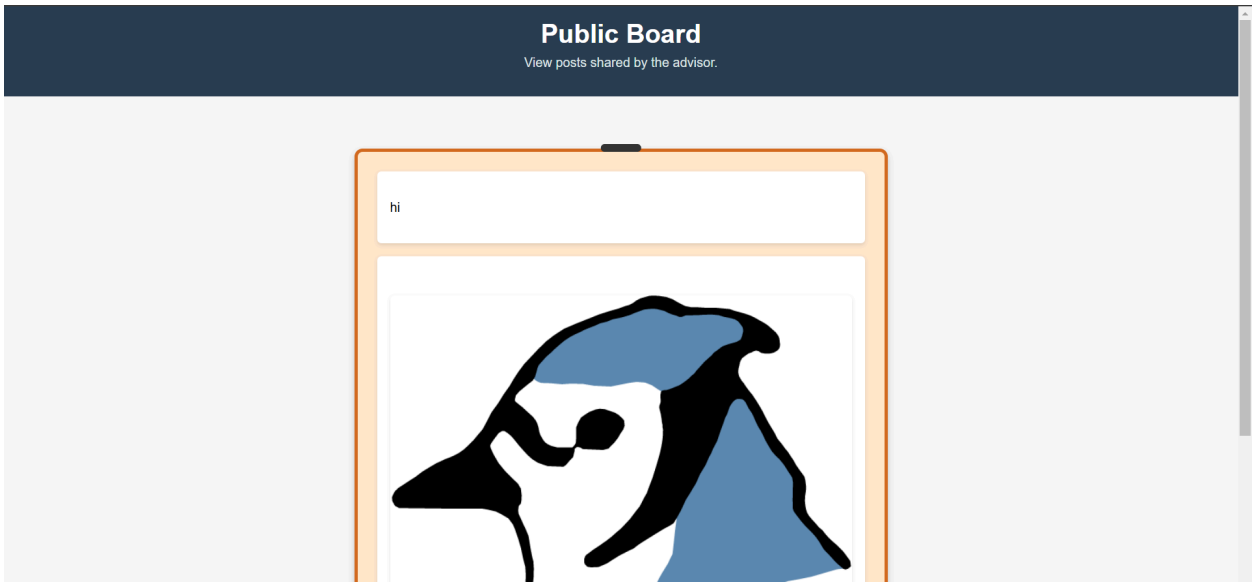
Send Messages to Users



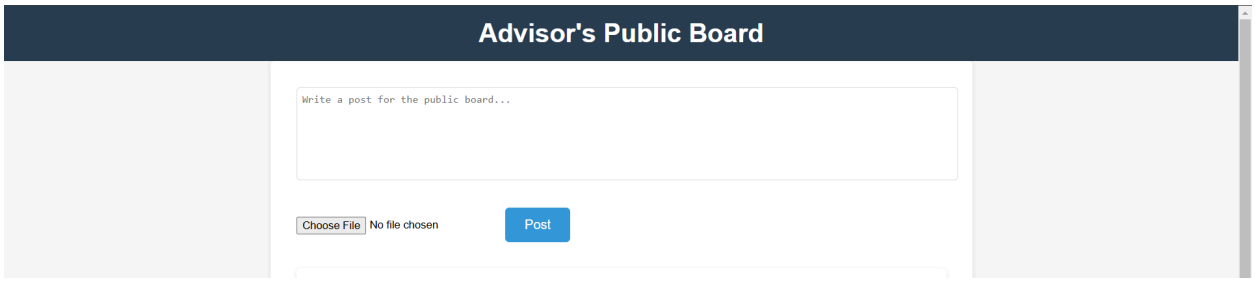
Generate Report based on data.



Generate Graph based on data.

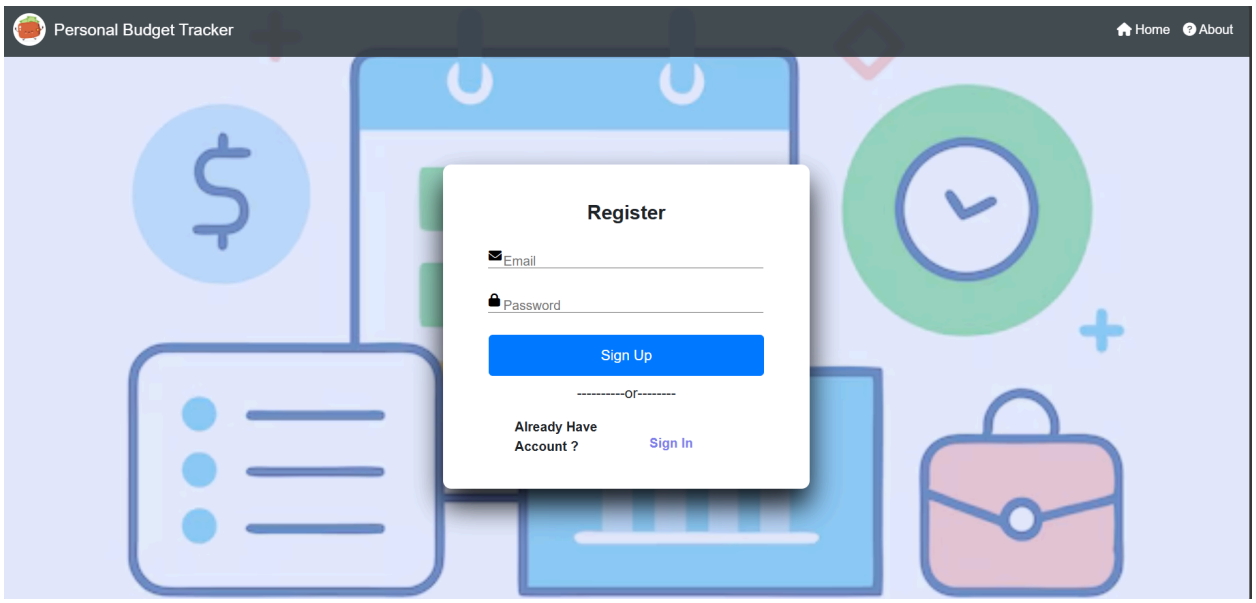


View Public Board

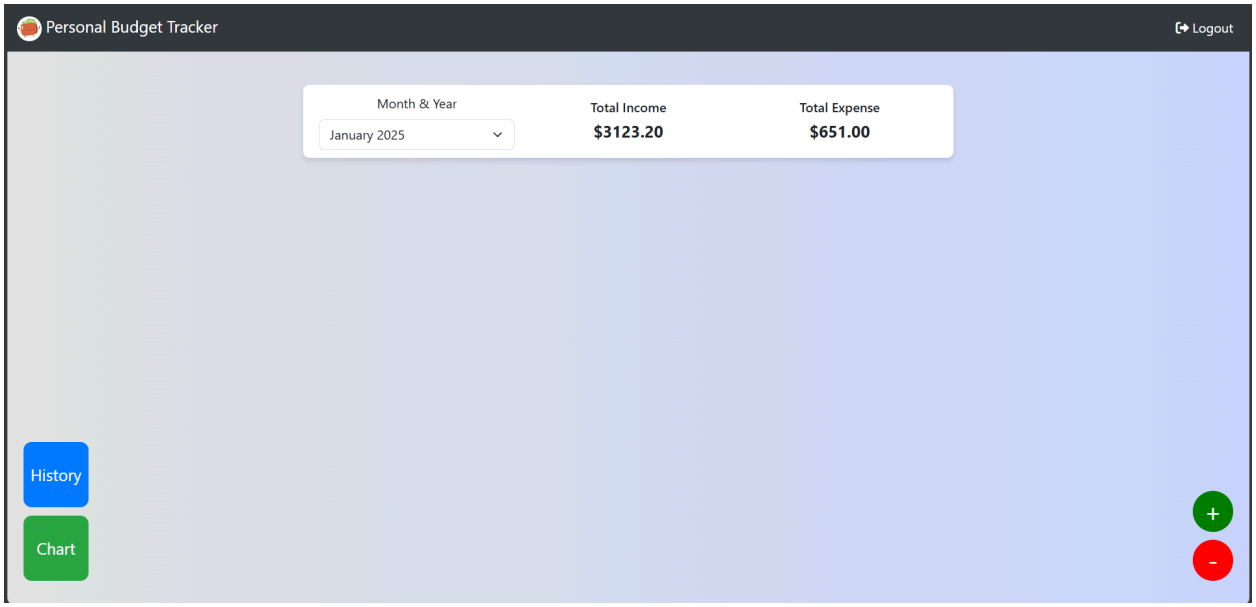


Post stuff on the Public Board.

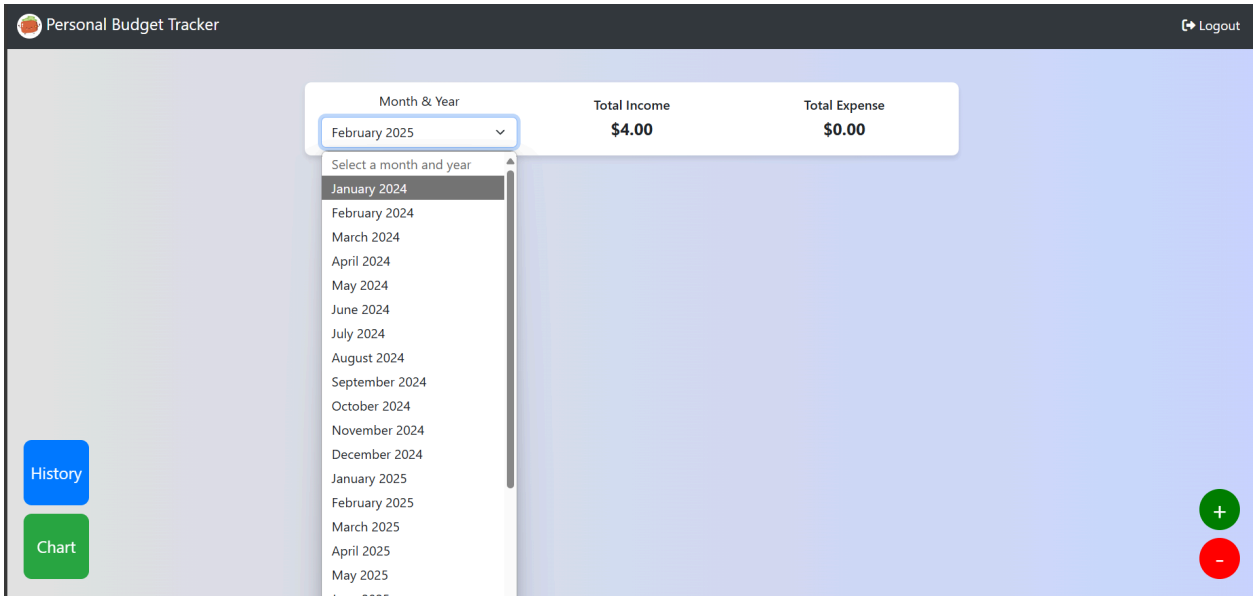
7.1.4 Subsystem Guest Subscreen



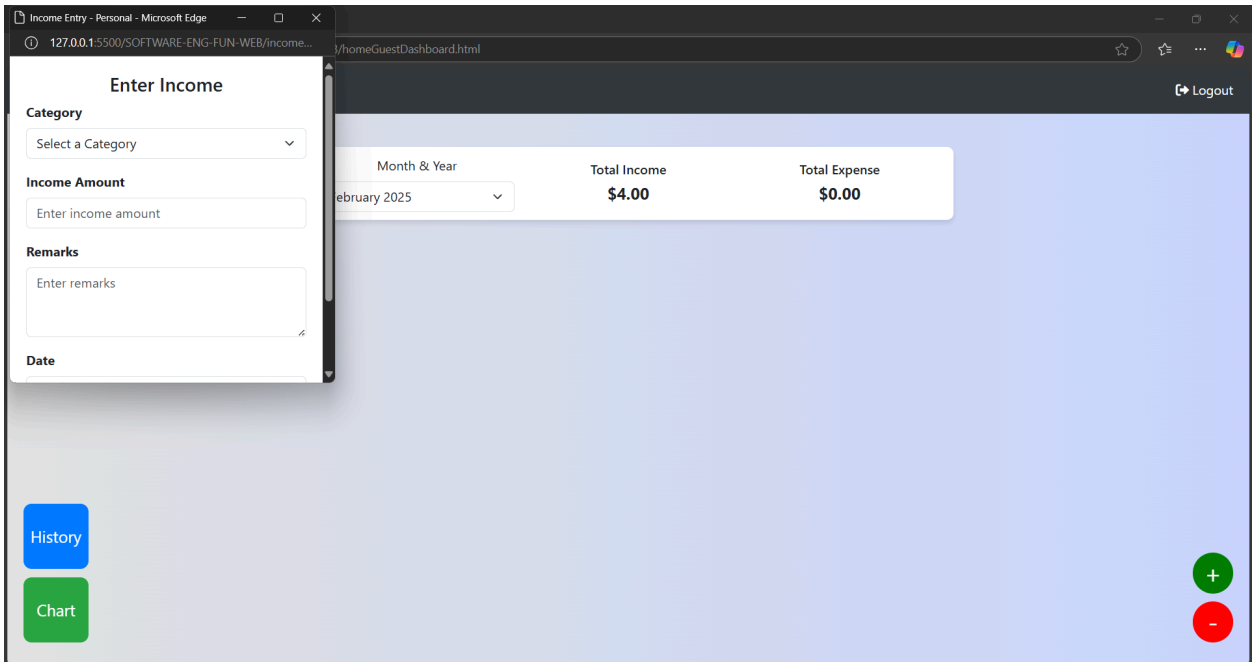
Sign Up Page



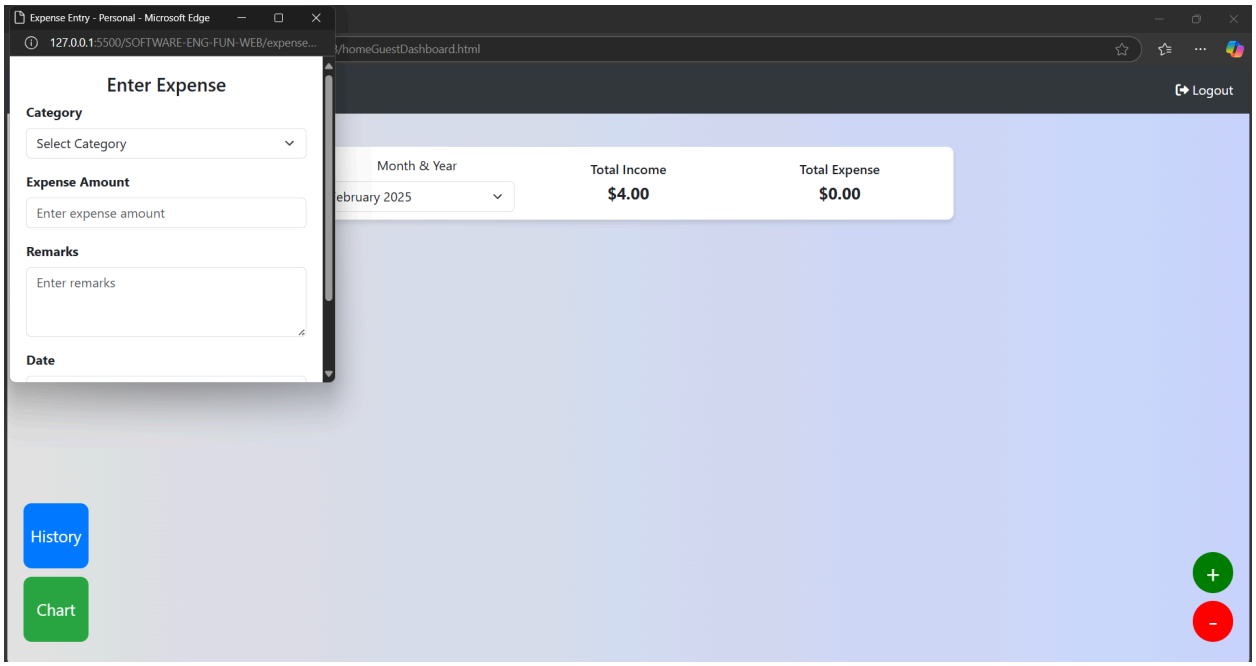
Guest dashboard



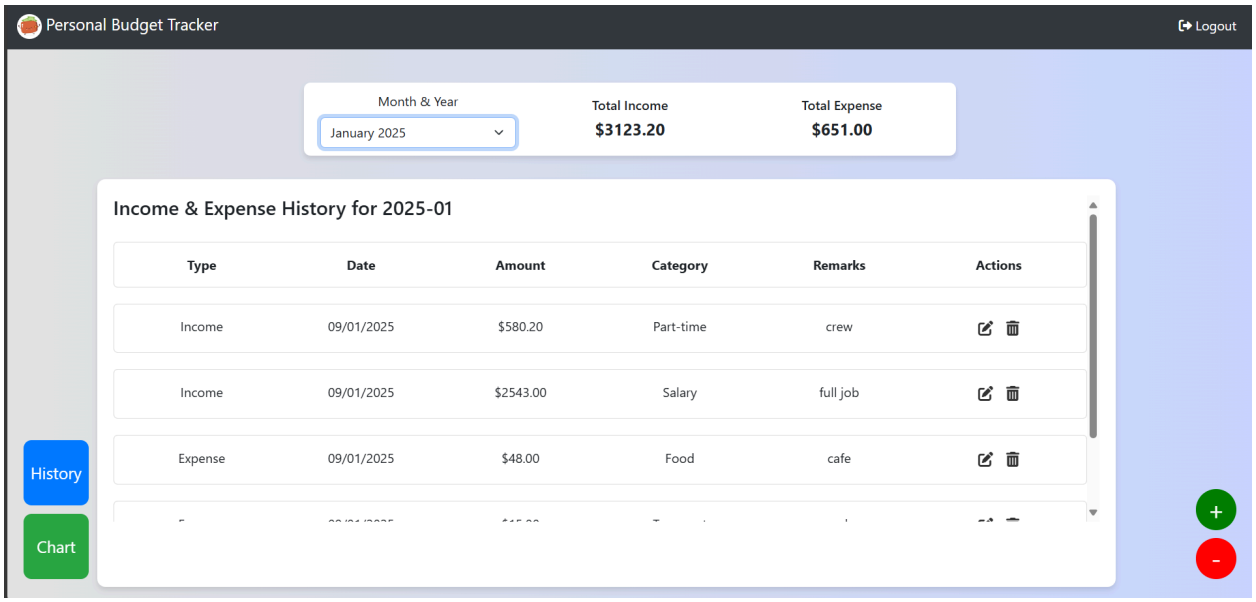
This can choose the specific history transaction month and year.



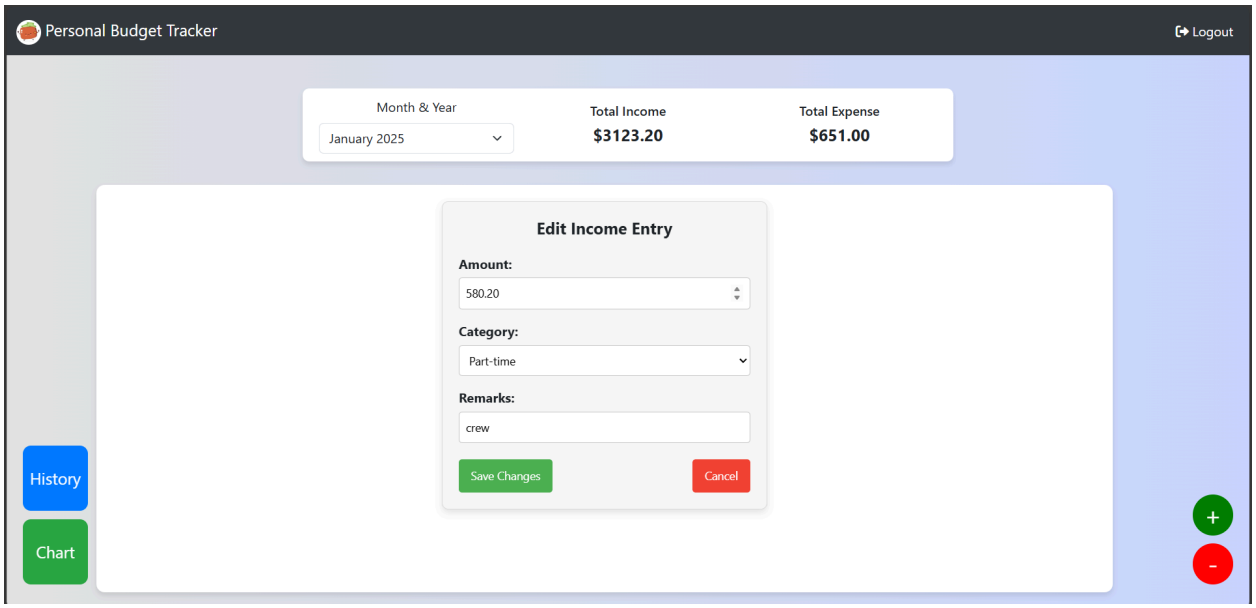
This “+” button can let users record income.



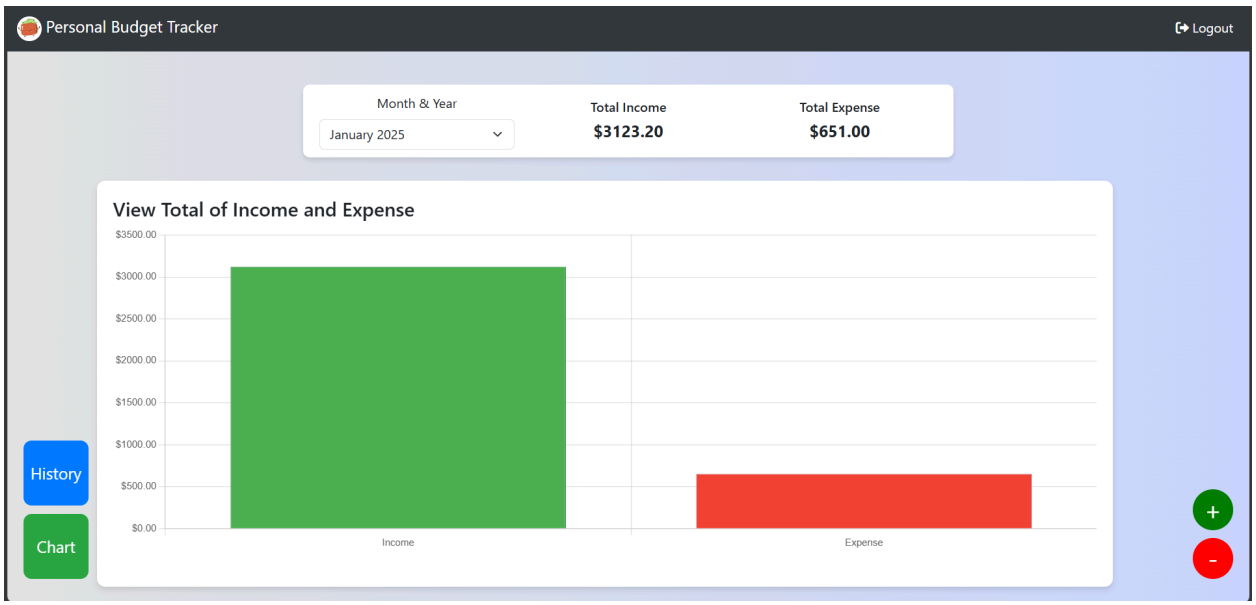
This “-” button can let users record expenses.



This “History” button can view transaction history.



The “Edit” button can edit Income / Expense Entry.



The “Chart” button can view the Bar Chart for total income and expense.

8 Conclusion

<This section should include details of completion of software, software quality assurance, group collaboration, problems encountered>

8.1 Project Achievements

This project has achieved most of its objectives and demonstrated the successful integration of key technologies. It has effectively implemented the functions and interactive systems we aimed for, utilizing HTML, CSS, and JavaScript as the core programming languages to ensure a well-structured, visually appealing, and responsive user interface. Additionally, Firebase served as the main database, providing real-time storage, authentication, and other essential services, enhancing both performance and user experience. The use of VS Code facilitated efficient debugging and version control, streamlining the development process. While some features, such as real-time messaging, were not fully implemented, the project successfully combined front-end development with backend services, resulting in a scalable and user-friendly web application.

8.2 Software quality assurance

Ensuring software quality was a crucial aspect of this project to guarantee reliability, performance, and usability. This was achieved through extensive testing and debugging at various stages of development. The project underwent functional testing to verify that all implemented features, such as user authentication and UI responsiveness, worked as intended. Additionally, usability testing was conducted to ensure a seamless and intuitive user experience. To maintain code quality, VS Code's debugging tools were used extensively, along with Firebase's built-in analytics and error-tracking capabilities. Version control with Git allowed for systematic updates, minimizing errors and ensuring a stable development process. Although some planned features were not fully implemented, the project maintained a high standard of quality, resulting in a functional, scalable, and user-friendly web application.

8.3 Group Collaboration

Our leader, Chin Zhen Ho, played a crucial role in this assignment by keeping the team organized, assigning tasks, and ensuring smooth coordination. He was also responsible for reviewing the final code and report, making sure everything was in order before submission.
Eric Teoh contributed significantly to the project by writing a large portion of the code and solving complex issues that others encountered. In addition, he kept track of the team's progress to ensure that all tasks were completed on schedule.
Our other team members, Gan Shao Yang and Bernard, also played important roles in coding, consistently completing assigned tasks to keep the project on track. They actively participated in discussions, offering valuable suggestions during meetings.
The documentation was a shared responsibility among all team members, including drawing diagrams, writing reports, and documenting progress. Through this collective effort, the project was successfully completed.

8.4 Problems Encountered

Of course, during development, we encountered several challenges that we were unable to fully overcome. One major issue was real-time messaging, which was intended to allow advisors or admins to communicate directly with users instead of relying on emails or messages. This feature was meant to facilitate easier access to user data for advisors. However, implementing this functionality required access to a paid service, which was beyond the project's limited budget, preventing us from integrating it.

Another set of challenges arose in the transaction history feature. The first issue occurred when users attempted to edit transaction details—the category selection, which was originally a drop-down menu, unexpectedly changed to a text-based input. Additionally, another problem surfaced when users added too many income or expense entries. If they did not click the history button again, it sometimes led to unintended duplication of records.

Despite these challenges, we managed to implement alternative solutions and ensured that the core functionalities of the project remained intact.