

**Savitribai Phule Pune University**

Ganeshkhind Rd, Ganeshkhind, Pune, Maharashtra 411007

Project on

**CAL-GET**

**Calorie and Budget Tracker App & Web**

S.Y. M.C.A, 2023-24

**Submitted by :**

1. Yashraj Gosavi (22112026)

2. Kamesh Rathore (22112007)

3. Prins Pardhi (22112035)

**Guided by :**

Prof: Jyoti Yadav

**CONTENT :**

1. **Introduction**
2. **Problem Description**
3. **Proposed Solution**
4. **Project Objective**
5. **Project Scope**
6. **Methodology**
7. **Project Deliverable**
8. **Functional Requirements**
9. **Non Functional Requirements**
10. **Diagrams**
11. **Demonstration**
12. **Conclusion**
13. **Introduction:**

Cal-Get is a mobile application that aims to solve the problem of tracking and managing both expenses and calorie intake for individuals who want to take control of their daily routines. With busy lifestyles and increasing health concerns, it can be challenging for people to keep track of their expenses and calorie intake on a daily basis. This application provides a user-friendly interface that allows individuals to easily input their daily meals and expenses, view their progress, and receive personalized recommendations on how to achieve their goals.

1. **Problem Description:**

The problem of managing expenses and calorie intake is becoming increasingly important as people seek to take control of their daily routines and achieve their health and financial goals. With the increasing availability of fast food and convenience foods, it can be challenging to maintain a healthy diet and stay within a budget. Additionally, people often struggle to keep track of their expenses and may overspend without realizing it.

1. **Proposed Solution:**

Cal-Get is a mobile application that offers a simple and effective solution for tracking and managing expenses and calorie intake. The application includes features such as a food database, budget tracking tool, and personalized recommendations that help users stay on track and achieve their goals. With Cal-Get, users can easily track their calorie intake, monitor their spending habits, and receive recommendations on how to adjust their diet or budget to achieve their goals.

1. **Project Objectives:**

The main objectives of Cal-Get are to develop an easy-to-use and efficient budget and calorie tracking system, provide real-time tracking of expenses and calories, and offer personalized recommendations to help users achieve their goals. Additionally, Cal-Get aims to provide a convenient and accessible solution for individuals who want to manage their health and finances simultaneously.

1. **Project Scope:**

The scope of Cal-Get includes the development of a mobile application that allows users to track and manage their expenses and calorie intake. The application includes features such as a food database, budget tracking tool, and personalized recommendations. The application will be available on both Android and iOS platforms and will be accessible through a web browser as well.

1. **Methodology:**

Cal-Get is developed using the JavaScript programming language and will utilize a web framework for developing the application. The application will be developed using an agile development process, and the development team will work in sprints to ensure timely delivery of the project. The application will be tested thoroughly to ensure its functionality, security, and usability.

1. **Project Deliverables:**

The deliverables for Cal-Get include a fully functional mobile application that allows users to track and manage their expenses and calorie intake. The application will include a food database, budget tracking tool, and personalized recommendations. Additionally, the project will include documentation, support services, and ongoing maintenance and updates.

1. **Functional Requirements:**

**User Registration and Authentication:**

Users should be able to create an account and securely authenticate themselves to access the application.

User registration should include capturing necessary details such as name, email, and password.

Expense Tracking:

Users should be able to input their daily expenses and categorize them.

The application should provide a form or interface to enter expenses, including date, description, amount, and category.

Users should be able to view and edit their expense history.

**Calorie Intake Tracking:**

Users should be able to input their daily meals and track their calorie intake.

The application should include a food database or allow users to manually enter food items and their corresponding calorie values.

Users should be able to view and edit their meal history and calorie intake.

**Real-time Tracking:**

The application should provide real-time updates and calculations for expenses and calorie intake.

Users should be able to view their current expense total and remaining budget.

Users should be able to view their calorie intake for the day and track progress towards their goals.

**Goal Setting and Progress Monitoring:**

Users should be able to set goals related to their expenses and calorie intake.

The application should allow users to specify a target budget and desired calorie intake.

Users should be able to track their progress towards their goals and receive notifications or reminders.

**STATISTICAL & ML MODEL:**

Food Recommendation System Using Cosine Similarity:

The food recommendation system employs cosine similarity to provide personalized food recommendations based on user preferences. By calculating the similarity between the user's nutritional preferences and a database of food items, the system suggests relevant choices. This model is useful for tailoring food recommendations to individual preferences, contributing to a more personalized and satisfying user experience in dietary decision-making.

User Spending Forecast with ARIMA:

The time-series analysis using the ARIMA model allows for forecasting future user expenses based on historical spending patterns. By fitting the ARIMA model to the user's spending data, the script provides insights into potential future spending trends. This model aids in financial planning and decision-making by offering users a glimpse into their expected expenditure patterns.

Nutritional Score Calculation with FSAm-NPS:

The FSAm-NPS score calculation script evaluates the nutritional quality of food items in a user's expenses. By considering various nutritional attributes and applying predefined weights, the model computes an overall score and assigns a grade (A to E) indicating the nutritional quality. This model serves as a valuable tool for users seeking to make healthier food choices and gain insights into the nutritional profiles of their dietary selections.

User Clustering based on Spending and Calorie Patterns:

The K-Means clustering model segments users into distinct clusters based on their spending behavior and average calorie consumption patterns. The script extracts relevant data from user expenses, standardizes it, and applies K-Means clustering to identify groups of users with similar spending and dietary habits. This model enables businesses or service providers to understand and cater to different user segments, tailoring strategies to meet the unique needs of each cluster.

1. **Non-Functional Requirements:**

**User-Friendly Interface:**

The application should have an intuitive and visually appealing interface that is easy to navigate.

Users should be able to quickly understand how to input expenses, track calories, and view their progress.

**Performance:**

The application should respond quickly to user actions and provide real-time updates without noticeable delays.

The system should be able to handle multiple users simultaneously without significant performance degradation.

**Security:**

User data, including personal information, expenses, and calorie intake, should be securely stored and protected.

The application should implement proper authentication and authorization mechanisms to ensure user privacy and data integrity.

**Compatibility:**

The application should be compatible with major web browsers, such as Chrome, Firefox, and Safari.

The mobile application should be compatible with the latest versions of Android and iOS operating systems.

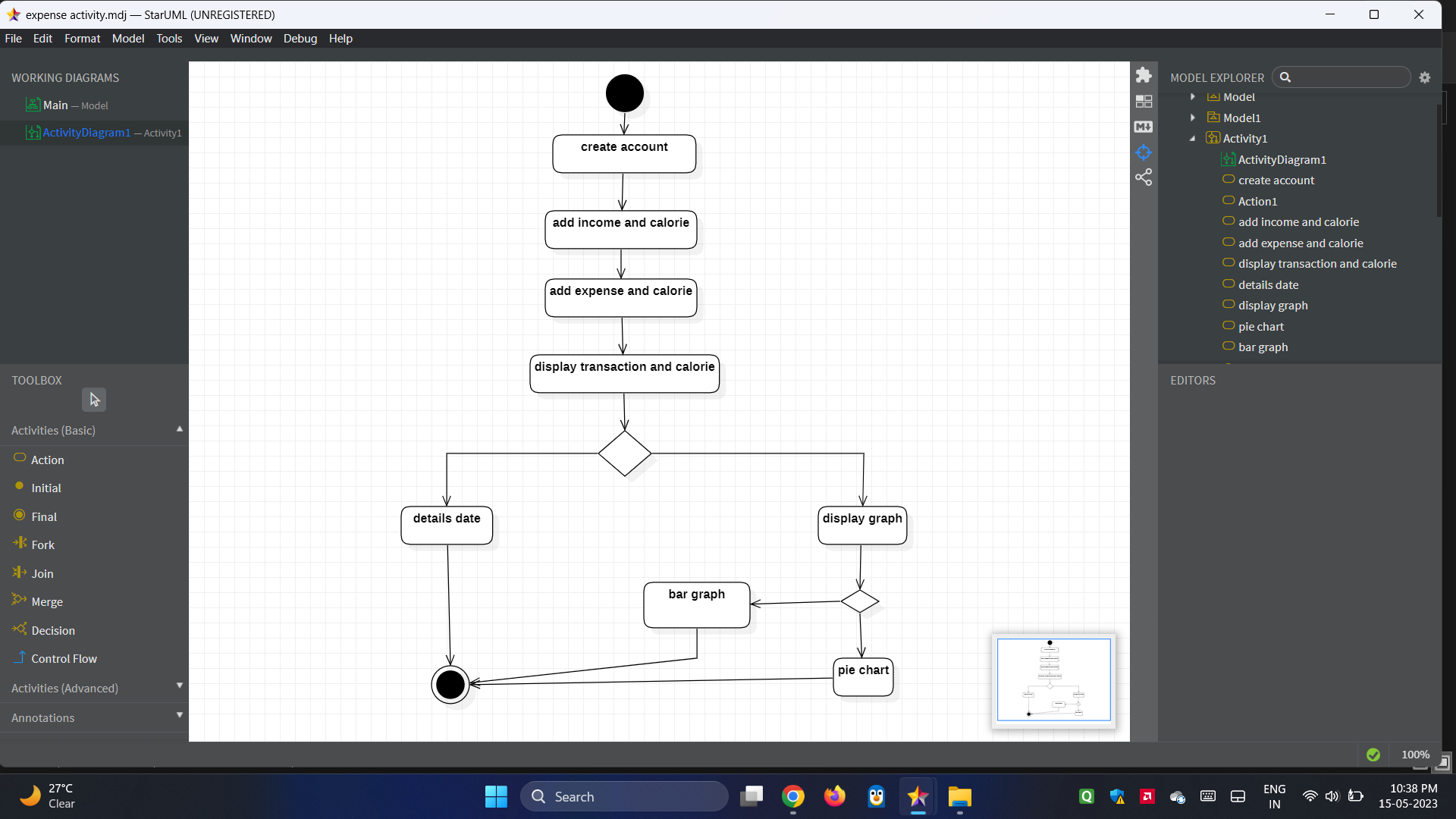
**Scalability:**

The application should be designed to handle an increasing number of users and growing data volumes.

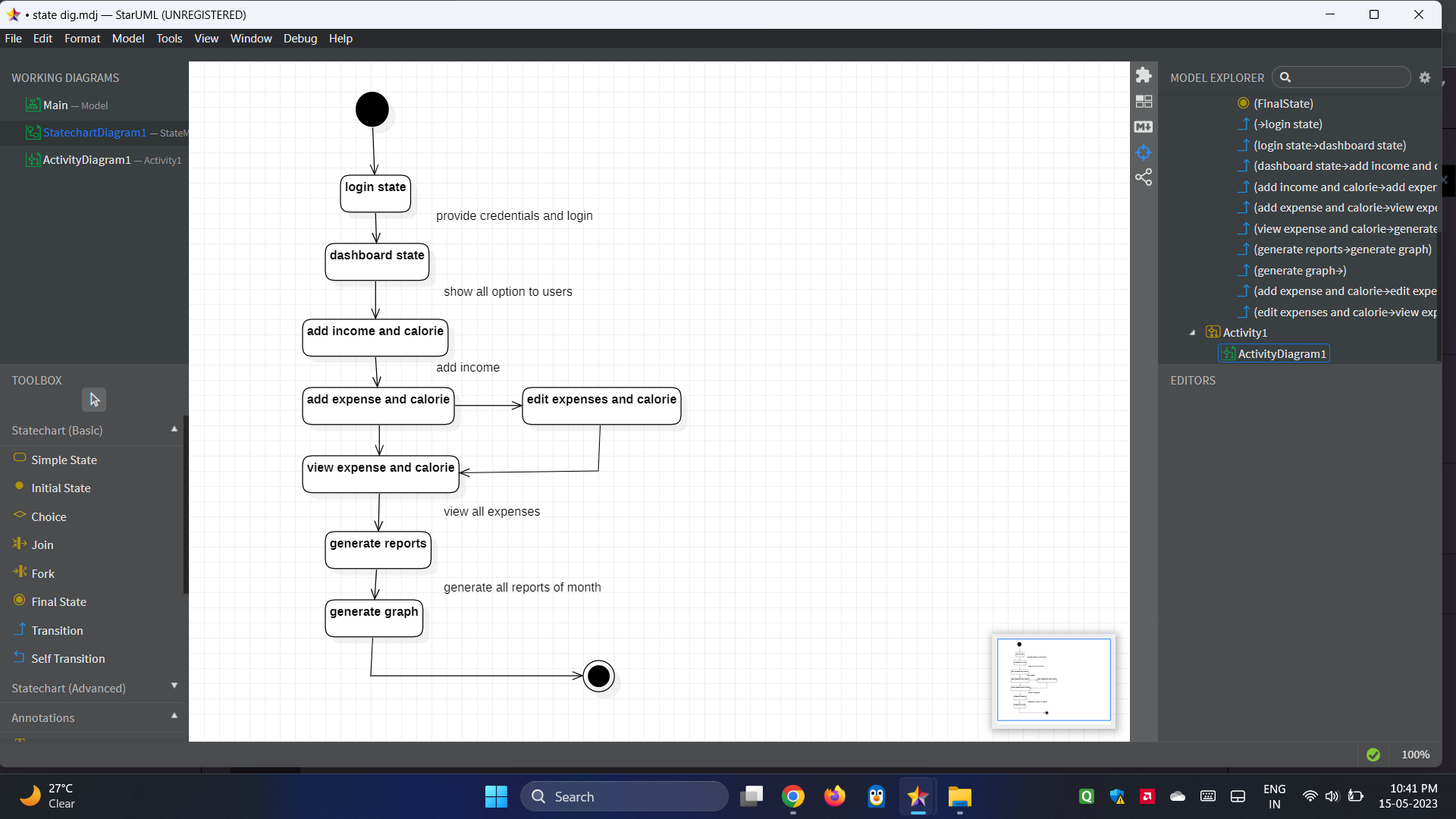
The system should be able to scale up or down to accommodate changing demands without compromising performance.

1. **Diagrams**

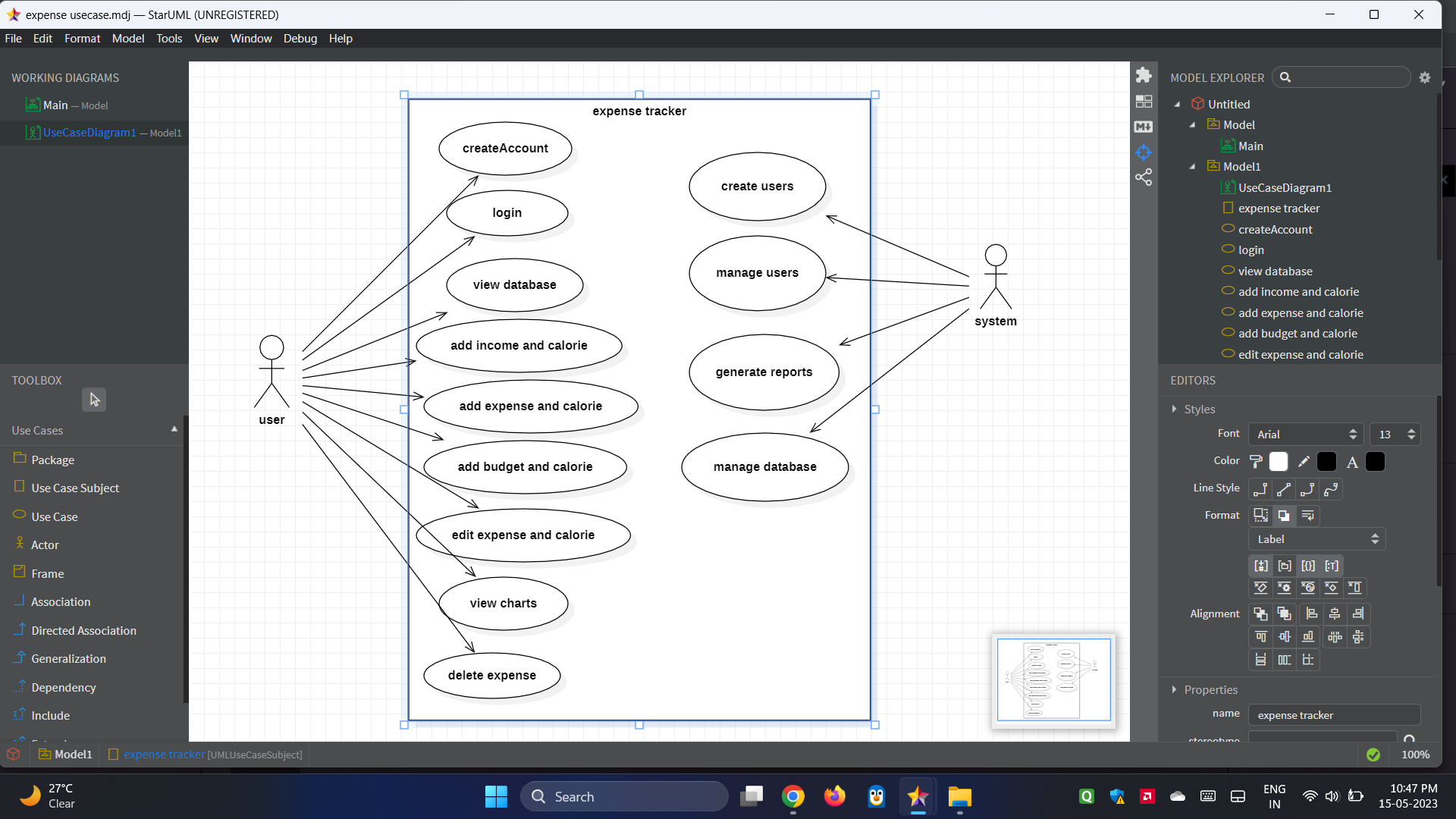
**ACTIVITY DIAGRAM**



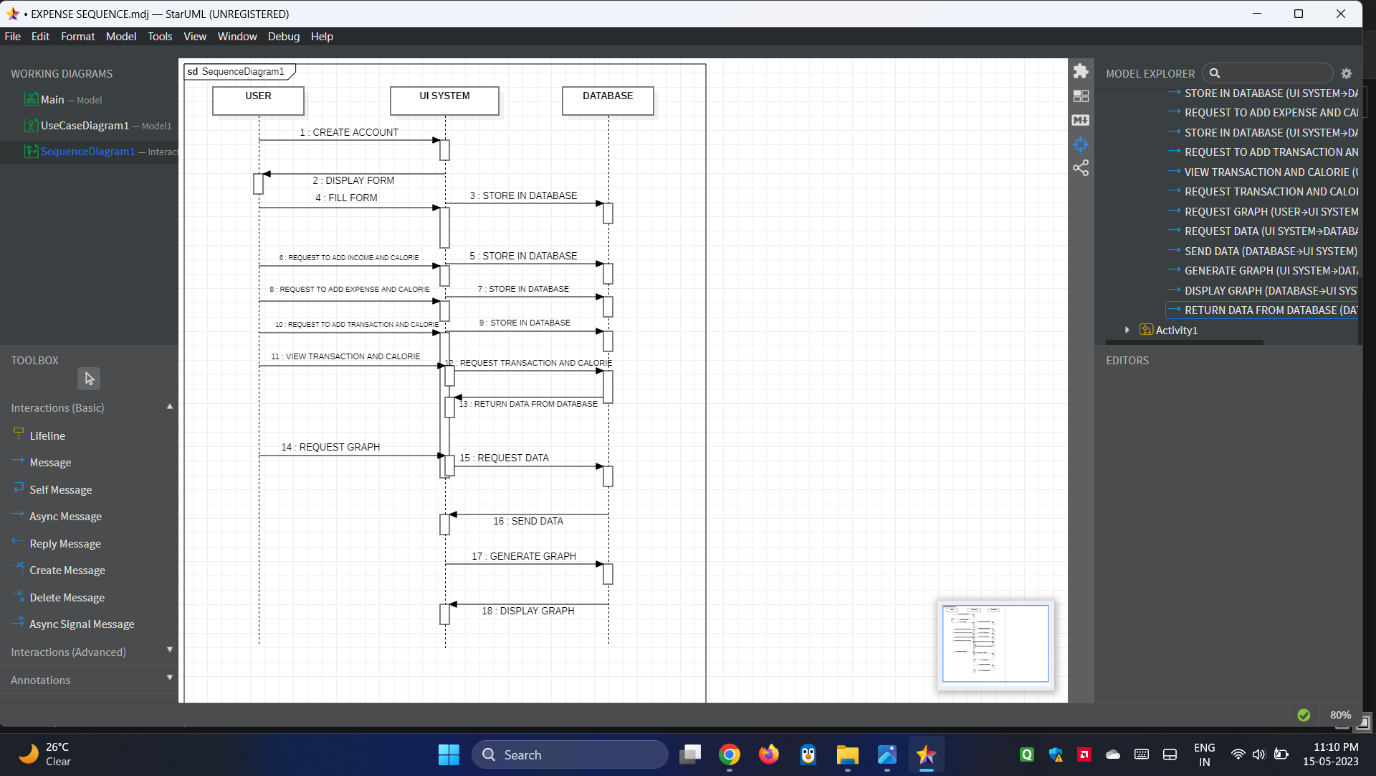
**STATE DIAGRAM**



**USERCASE DIAGRAM**



**SEQUENCE DIAGRAM**

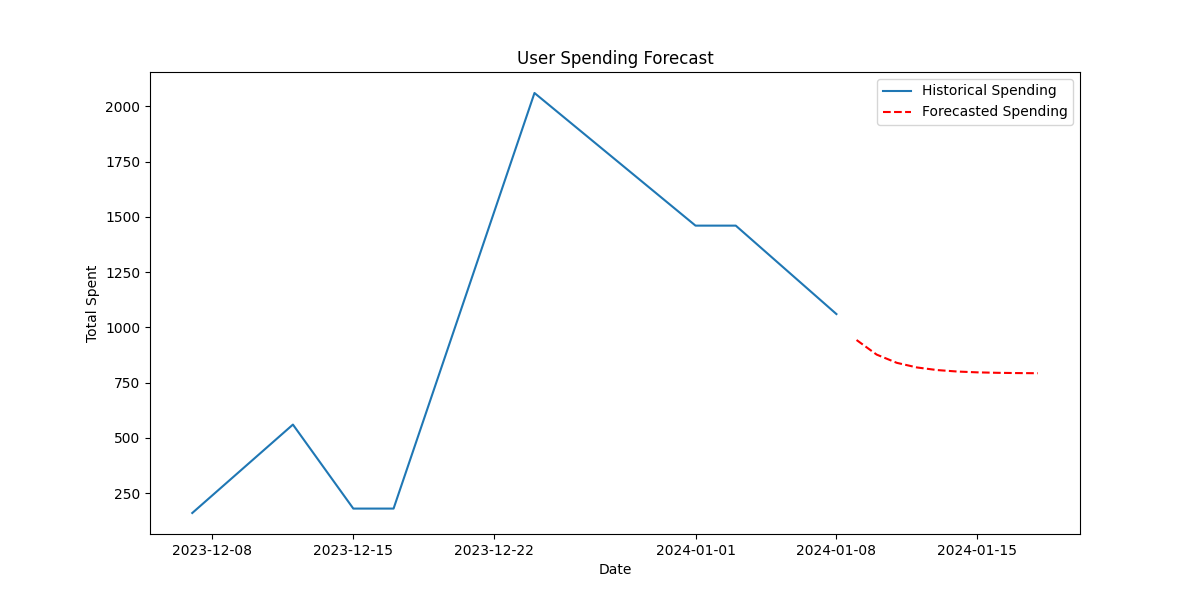
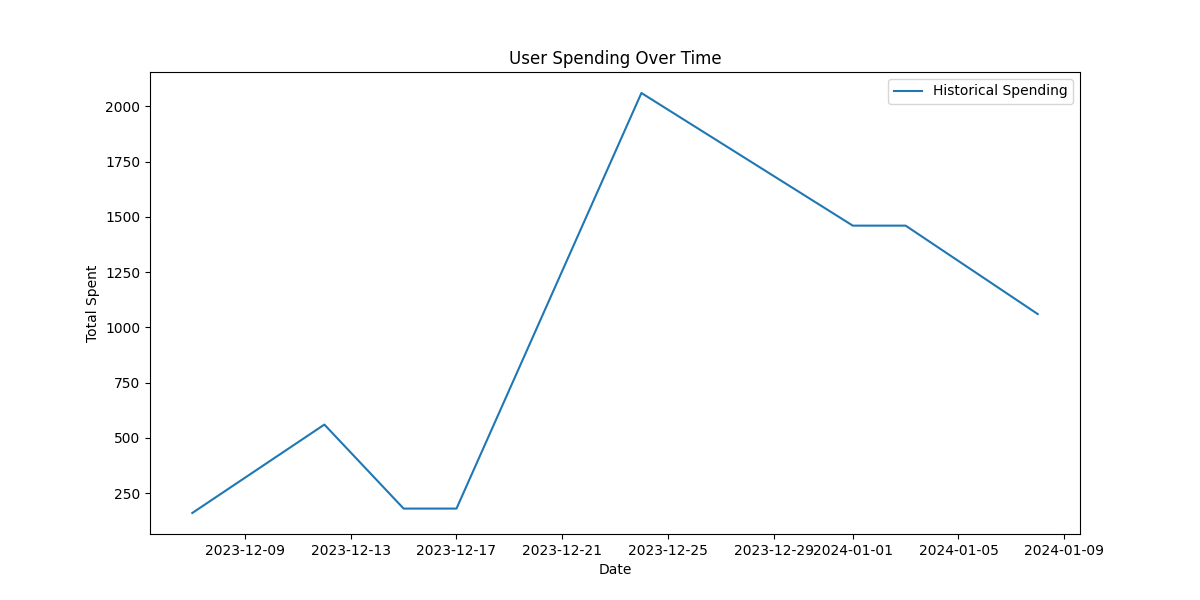


**STATISTICAL & ML MODEL OUTPUT:**

1. **ARIMA (Autoregressive Integrated Moving Average) model**

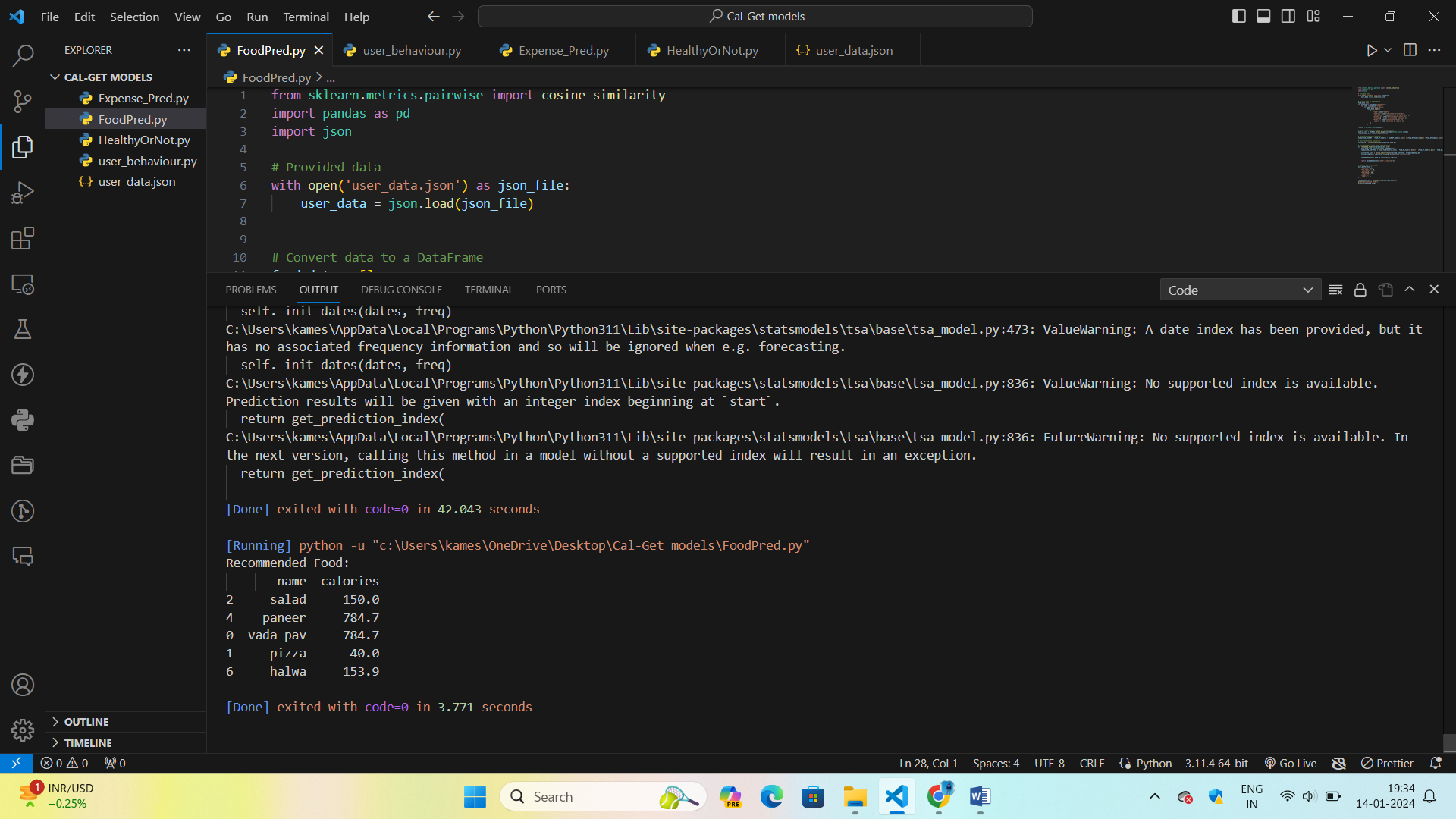
This Python code conducts time-series analysis on a user's spending data using the ARIMA (AutoRegressive Integrated Moving Average) model. It begins by loading user data from a JSON file, focusing on the 'expenses' field. The relevant time-series data, comprising dates and total spending amounts, is extracted and organized into a Pandas DataFrame. The historical spending pattern is then visualized through a Matplotlib plot. Subsequently, an ARIMA model with an order of (1, 1, 1) is fitted to the historical spending data. The model is used to forecast future expenses for a specified number of steps (in this case, 10 steps). The forecasted values are plotted alongside the historical spending data in a second Matplotlib plot, providing a visualization of both past and predicted future spending trends over time. The ARIMA model allows users to gain insights into potential future spending patterns based on historical data.

**Output:**



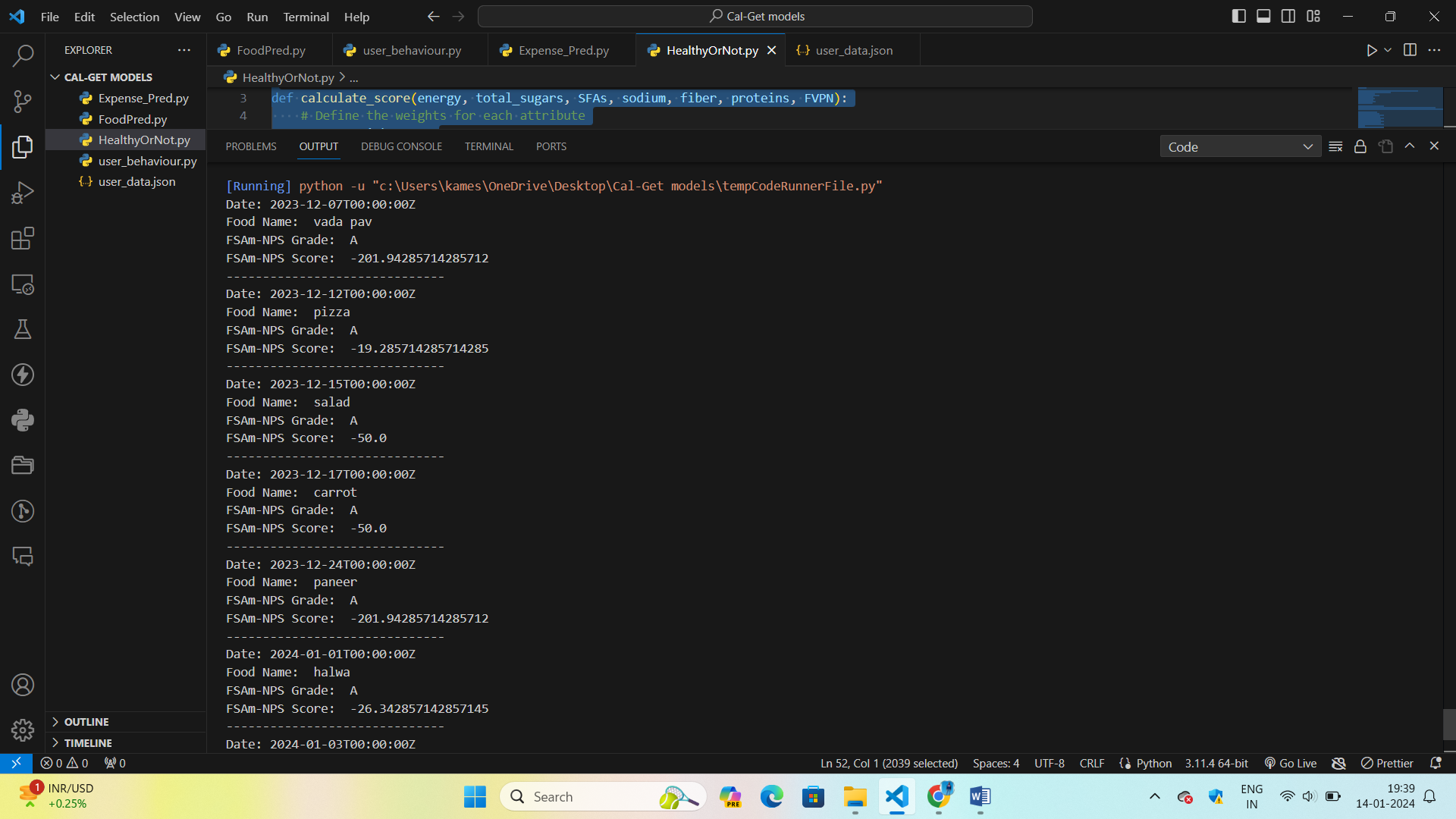
**2.** **Food Recommendation System Using Cosine Similarity**

This Python code performs food recommendation based on user preferences using cosine similarity. It starts by loading user data from a JSON file, specifically focusing on food-related expenses. The relevant nutritional information for each food item is then extracted and organized into a Pandas DataFrame. Numeric columns are selected for normalization, and the data is normalized to ensure consistent scaling. Cosine similarity is calculated among the normalized food items, forming a similarity matrix. The code defines a function, **recommend\_food**, which takes user preferences (calories, fat, protein, sodium, fiber, and sugar) and recommends food items based on their similarity to the user's preferences. The function computes the cosine similarity between the user preferences and the normalized food items, identifies the most similar items, and returns a specified number of recommendations. An example user preference is provided, and the recommended food items, along with their calorie content, are printed to the console. Overall, the code enables personalized food recommendations by leveraging cosine similarity on nutritional features.



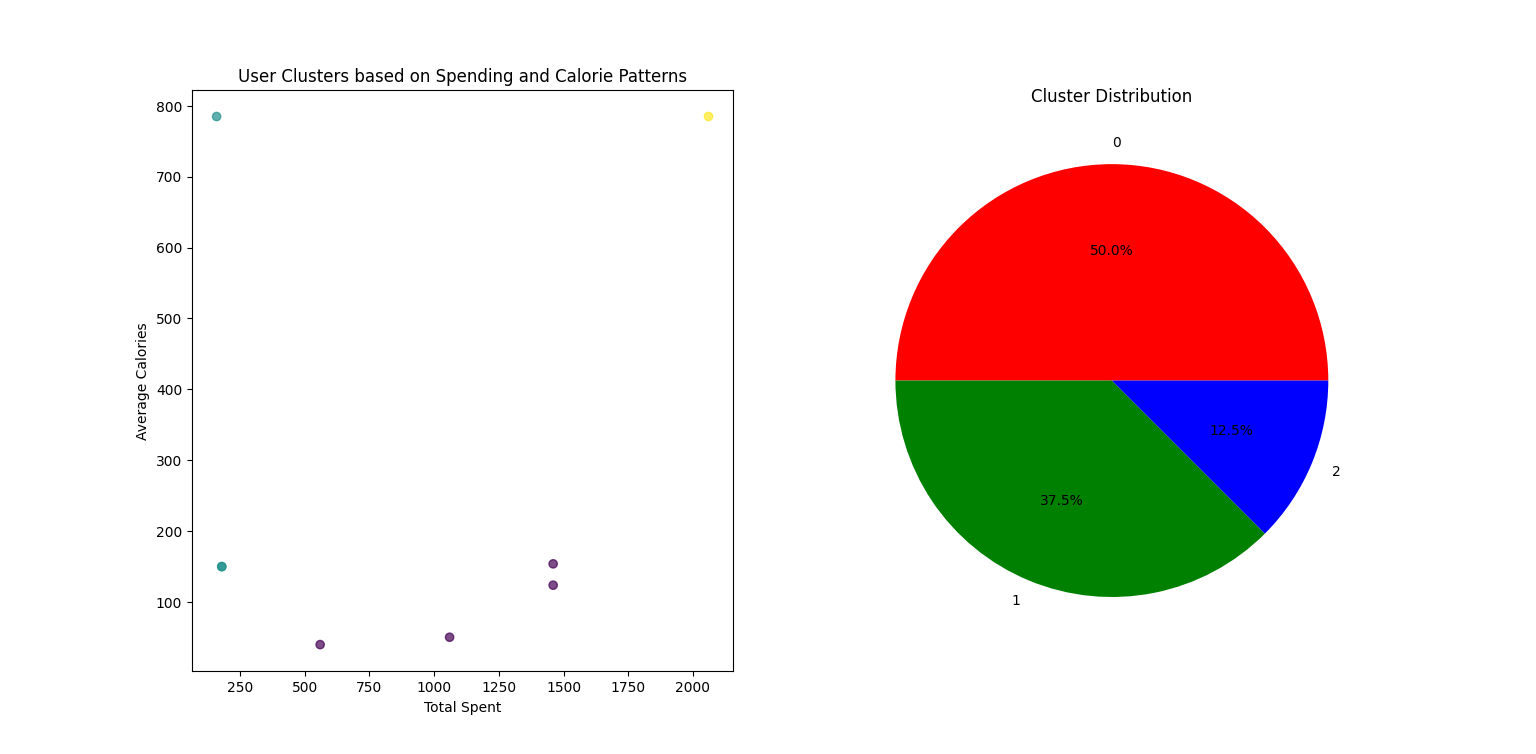
**3.** **Food Nutritional Score Calculation**

This Python script calculates the FSAm-NPS (Food Standards Australia and New Zealand Nutrient Profiling Scoring Criterion) score for each food item in a user's expenses based on its nutritional attributes. The FSAm-NPS score is computed using a weighted sum of energy, total sugars, saturated fatty acids (SFAs), sodium, fiber, proteins, and the optional FVPN (Fruit, Vegetable, and Nut content). The weights for each attribute are defined, and the final score is derived by dividing the sum by the total number of attributes. The script then assigns a nutritional grade (A to E) based on predetermined score thresholds. The user's expenses, including food items, are loaded from a JSON file, and the FSAm-NPS grade and score for each food item are printed along with the date. This script facilitates the evaluation of the nutritional quality of food choices and can be a valuable tool for promoting healthier eating habits.

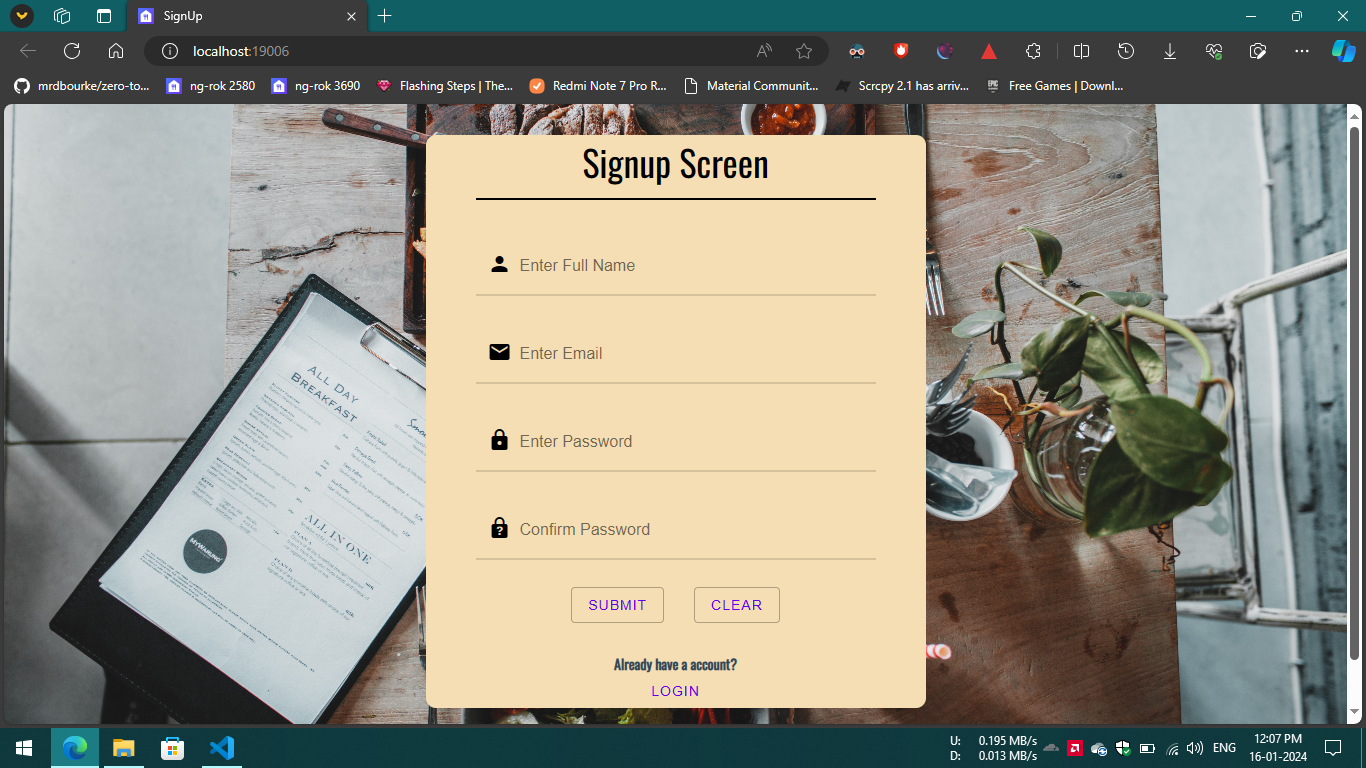


**4.** **User Clustering based on Spending and Calorie Patterns:**

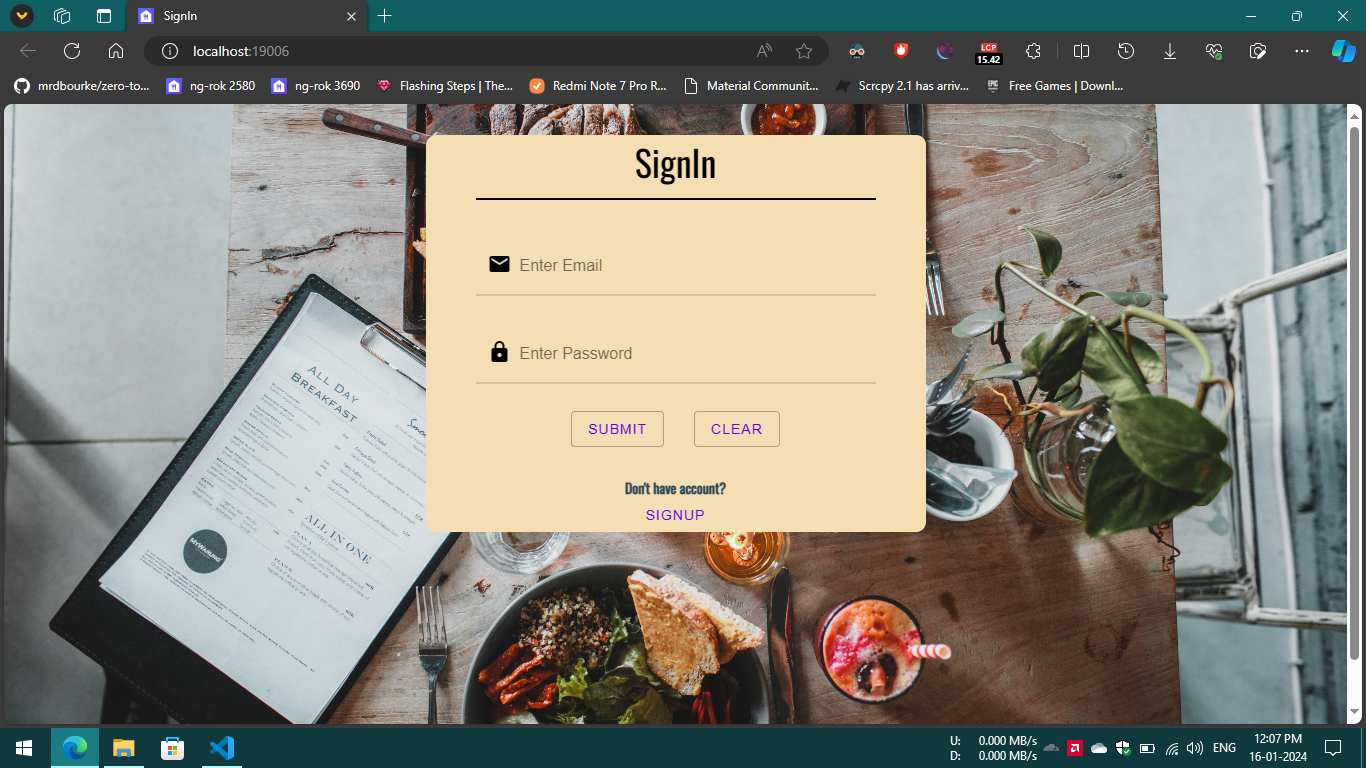
This Python script utilizes K-Means clustering to categorize users based on their spending behavior and average calorie consumption patterns. It loads user expense data from a JSON file, extracts relevant information such as total spending and average calories per food item, and creates a DataFrame for analysis. The data is standardized to ensure equal contribution of features in clustering. K-Means clustering with three clusters is applied, and the resulting clusters are visualized using a scatter plot and a pie chart. The scatter plot illustrates user distribution in the spending and calorie space, while the pie chart shows the proportion of users in each cluster. The script concludes by displaying the cluster assignments for each user, providing insights into distinct user segments with similar spending and calorie consumption tendencies.



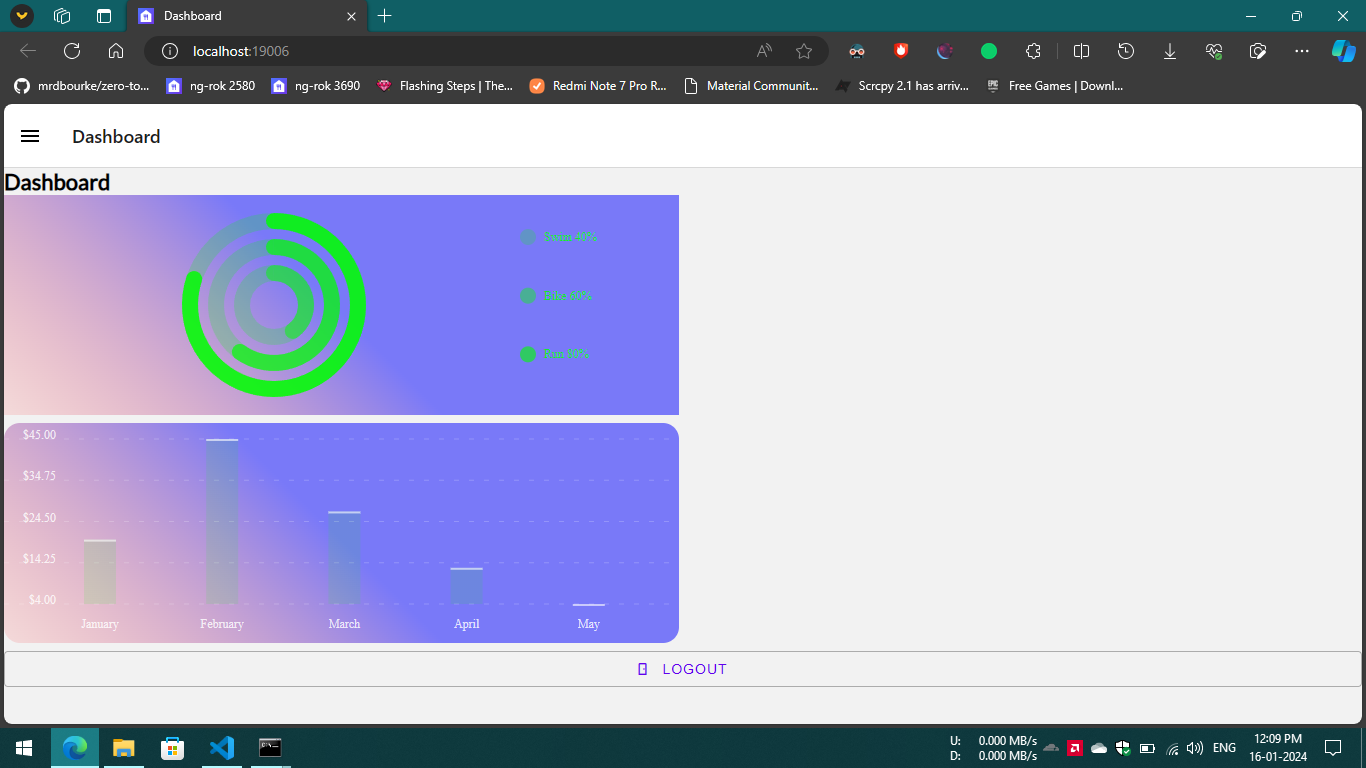
1. **Demonstration:**

****

**Fig. 1: User need to Sign-up.**

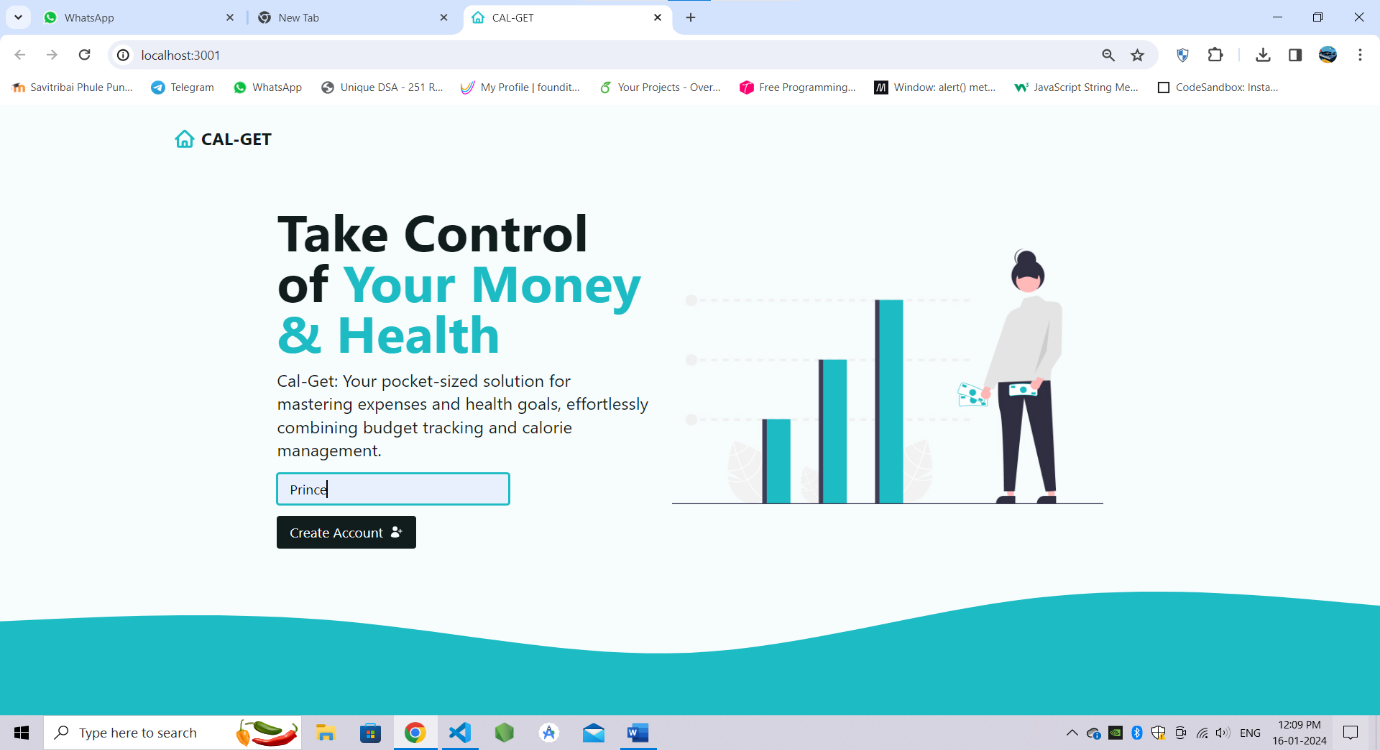
****

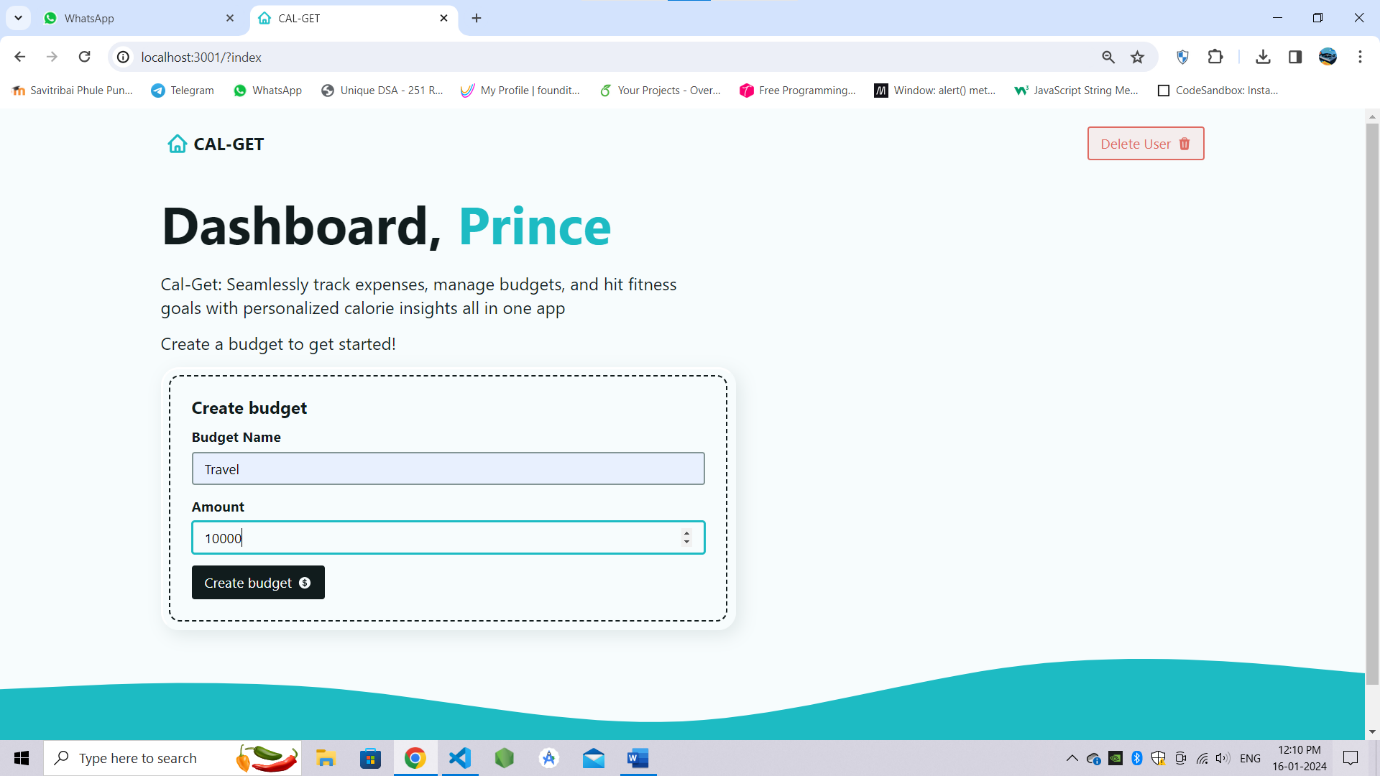
**Fig. 2: Sign In Screen**

****

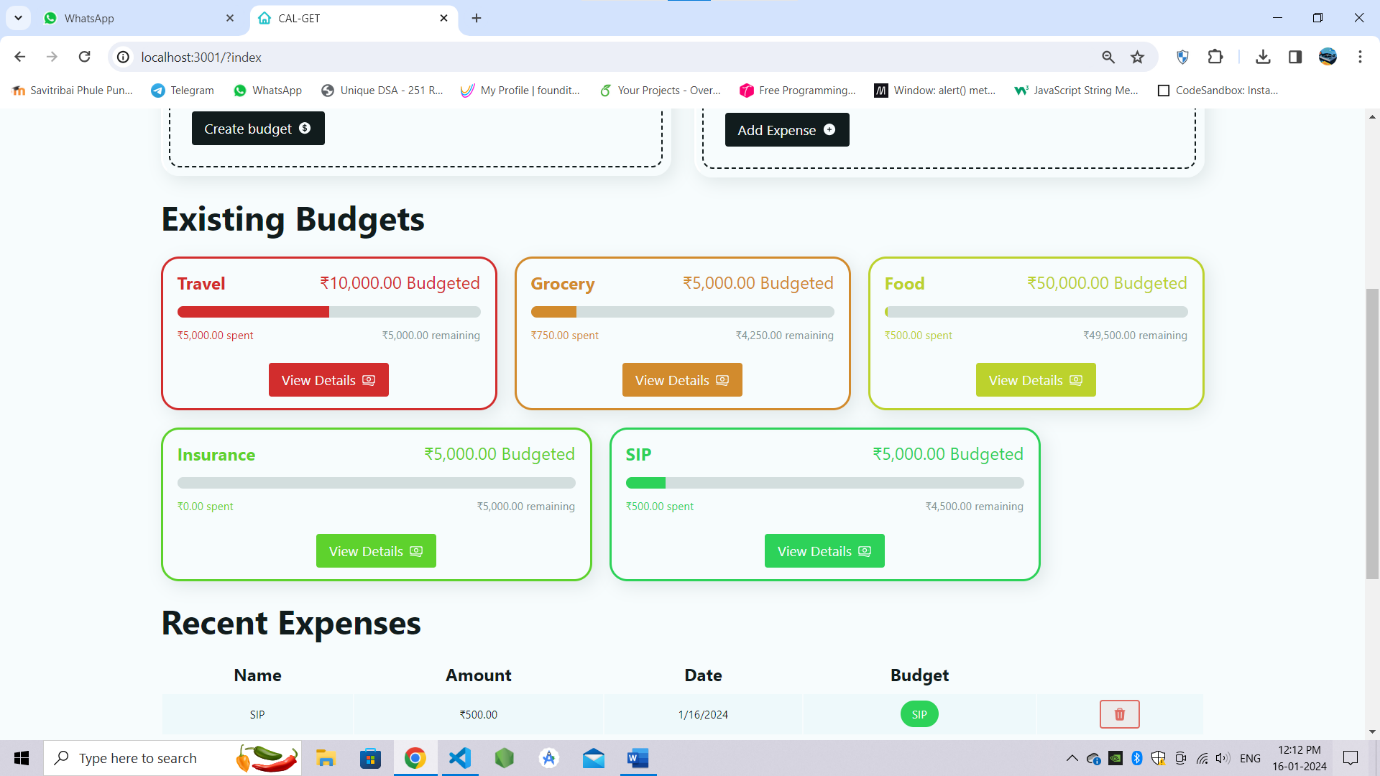
**Fig. 3: Dashboard of mobile app.**

**Web application demonstration:**

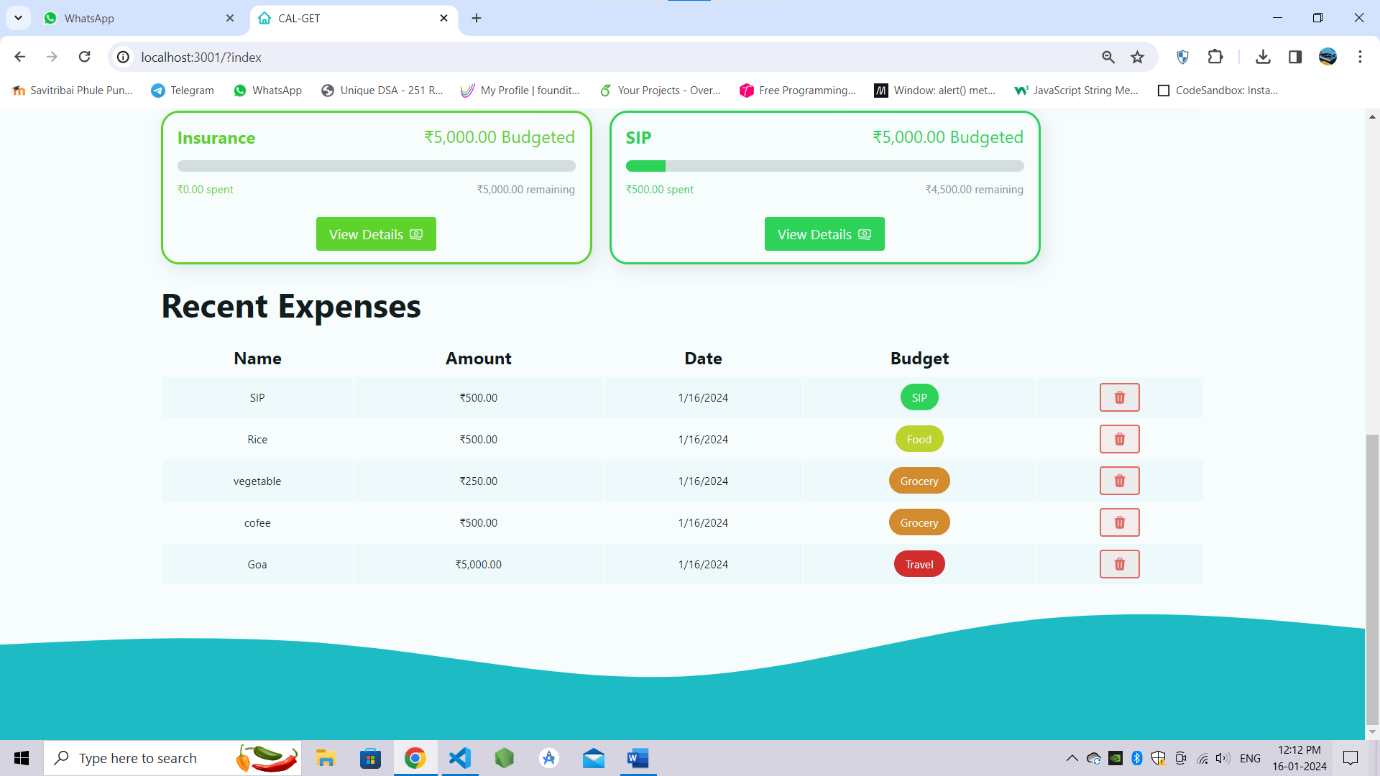
** Fig. 1: User need to enter their profile name.**

****

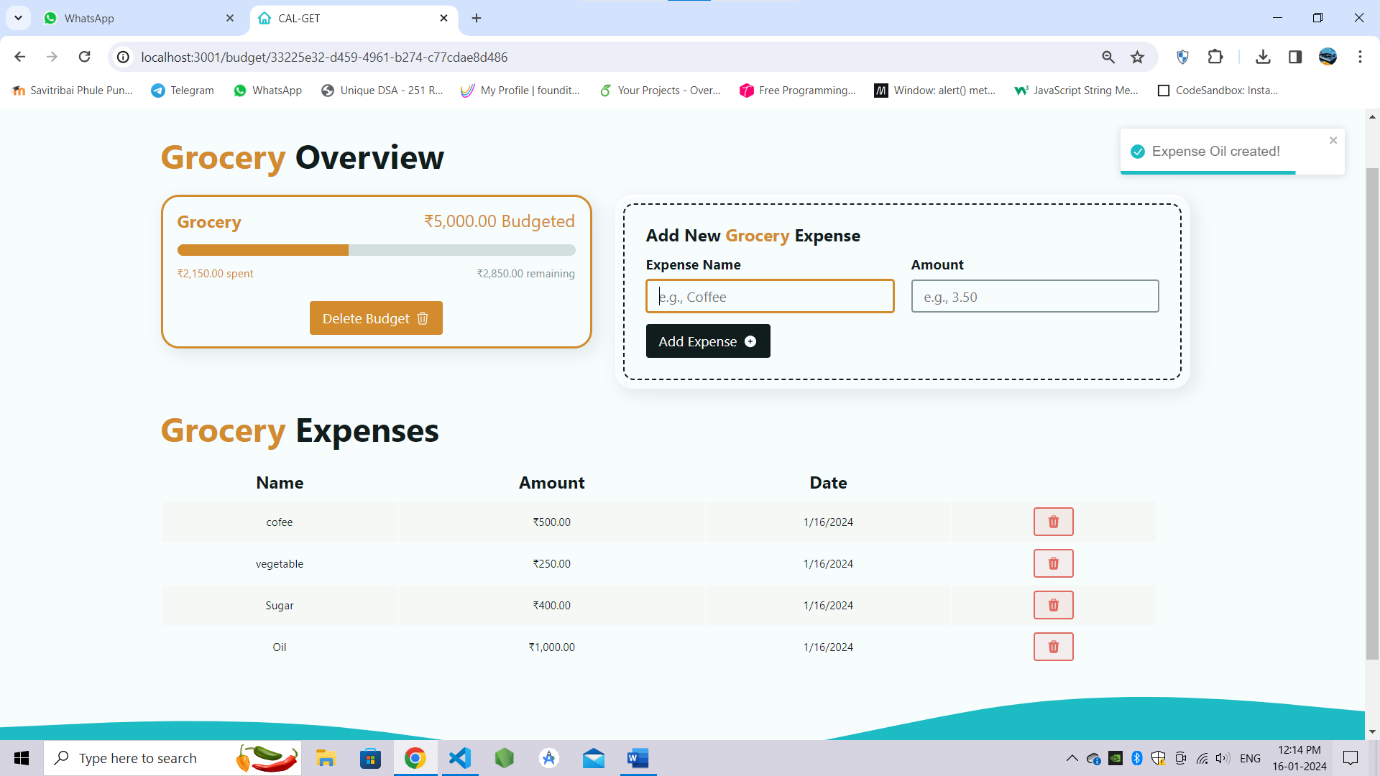
**Fig. 2: User can add budget name with amount.**

****

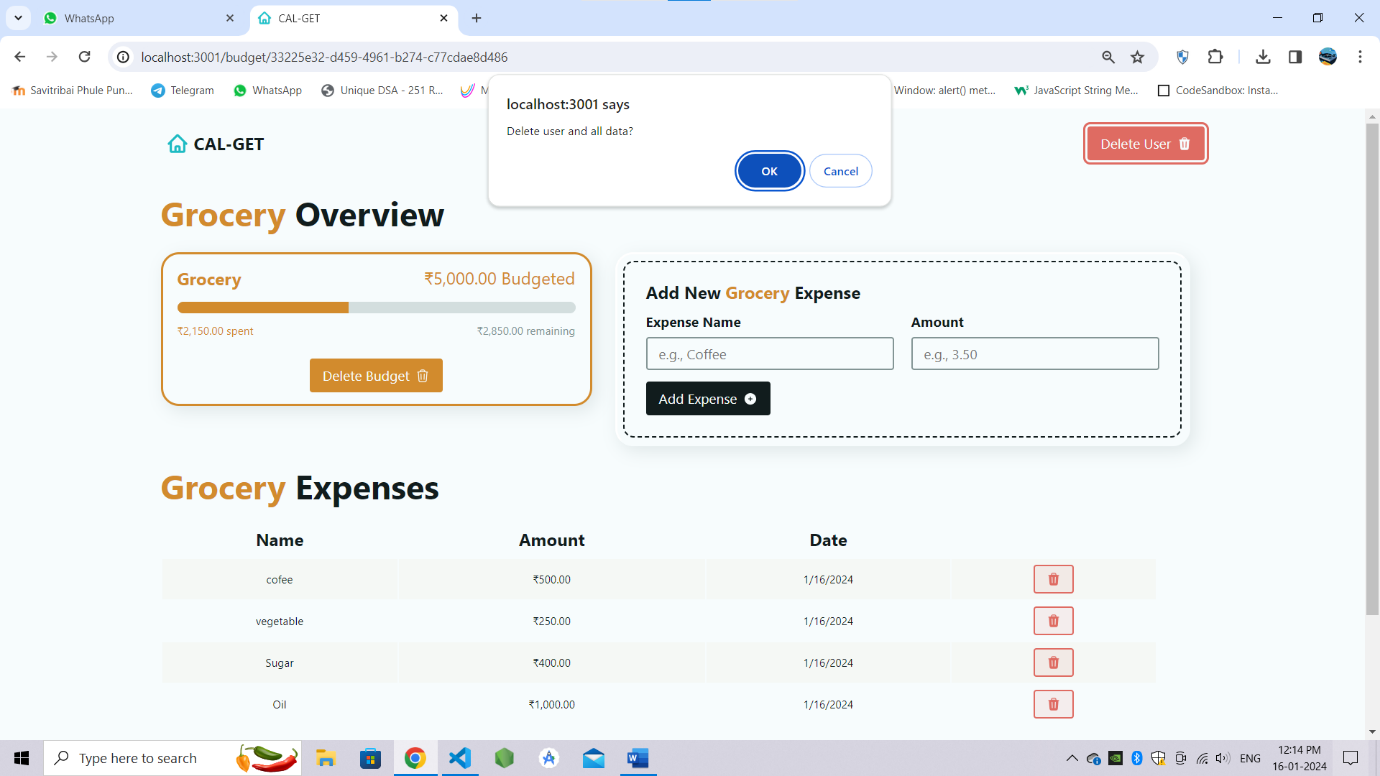
**Fig. 3: List of all added budget.**

****

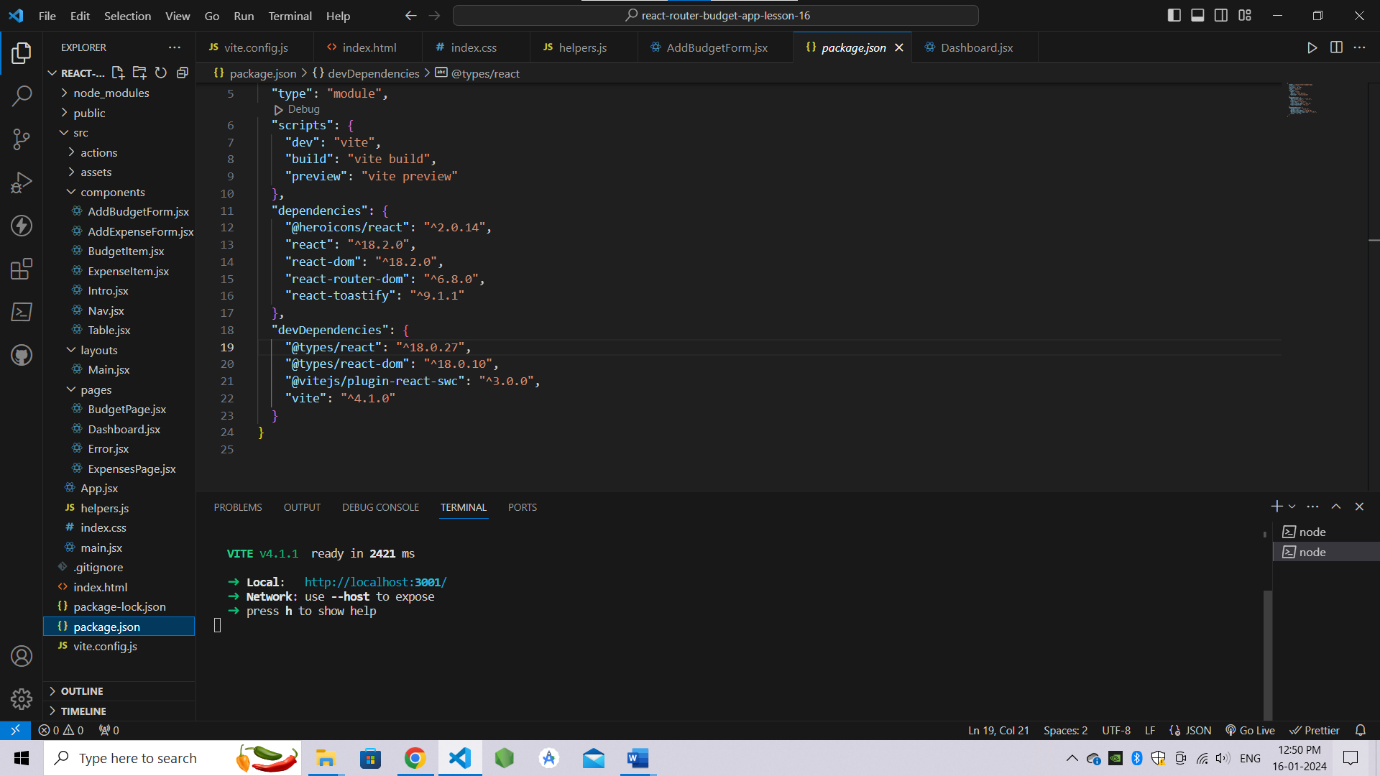
**Fig. 4: List of all expense of each category.**

****

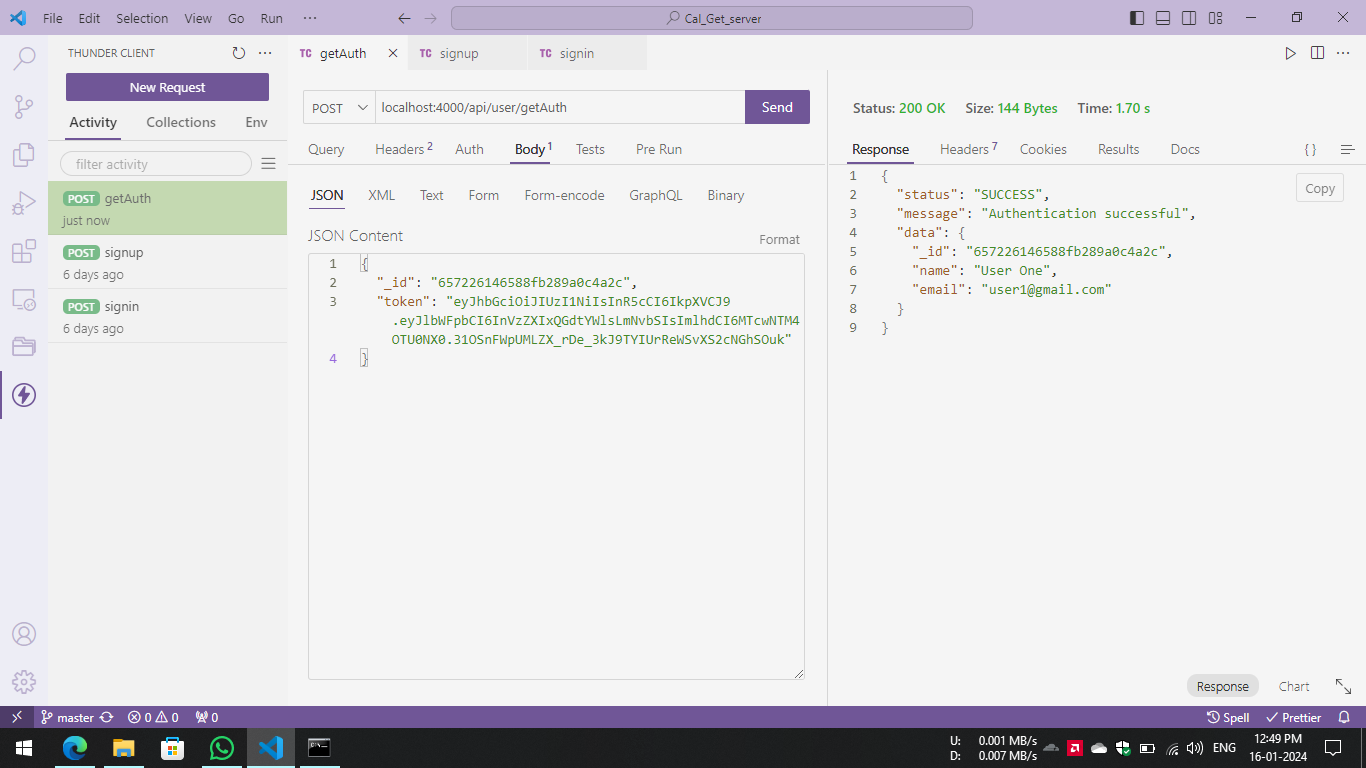
**Fig. 5: User can add more expense.**

****

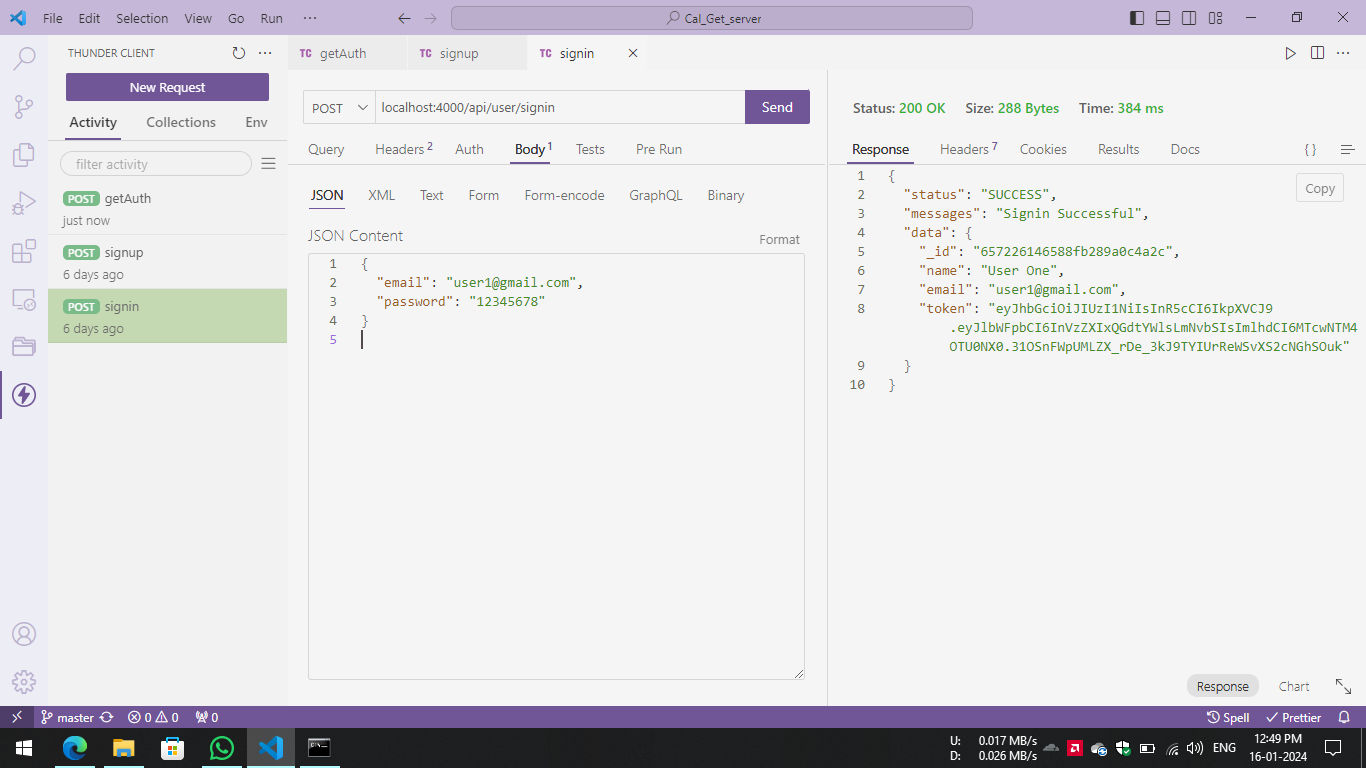
**Fig. 6: User can delete the expense and budget also.**

****

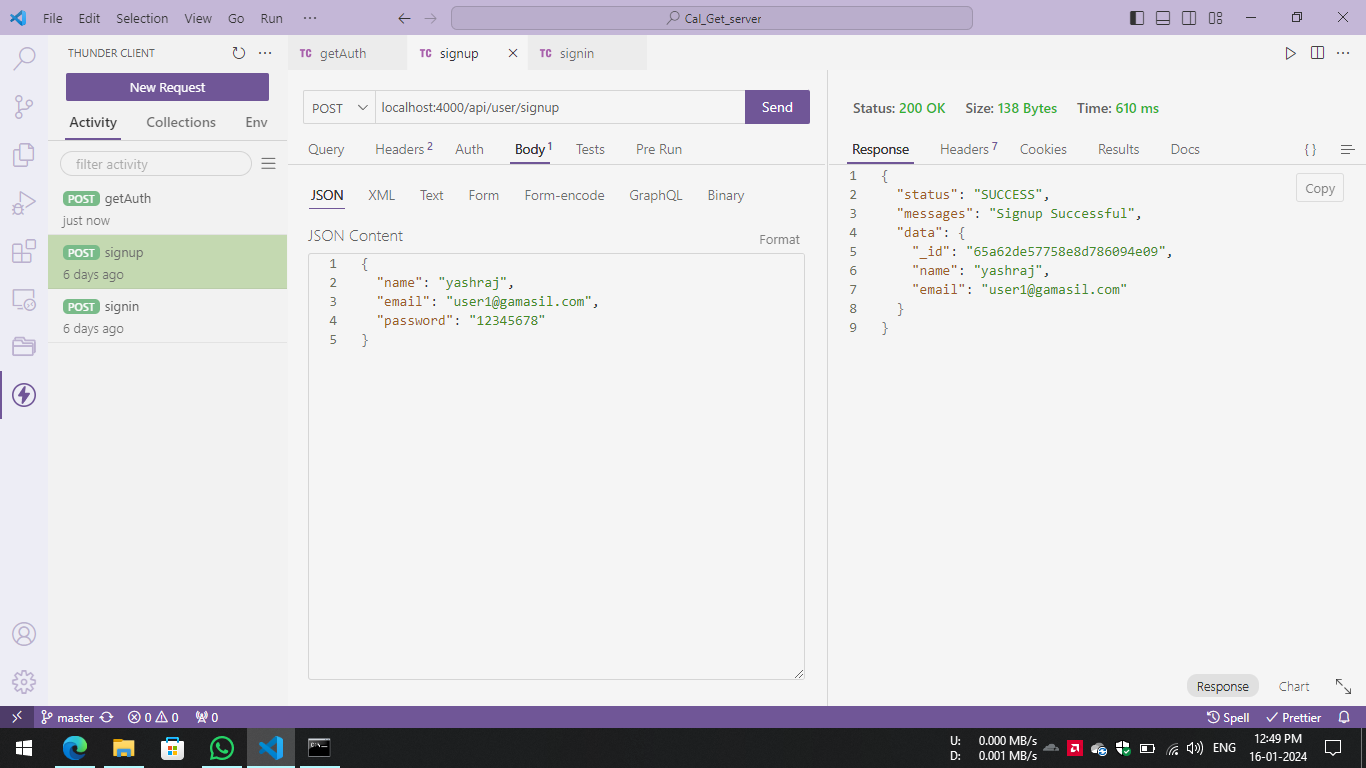
**Fig. 7: Screenshot of Code file.**

****

**Fig. 8: API authentication.**

****

**Fig. 9: Sign In data.**

****

**Fig. 10: Sign Up data.**

1. **Conclusion:**

In conclusion, Cal-Get is an important and timely solution for individuals who want to take control of their health and finances. With its user-friendly interface, personalized recommendations, and real-time tracking of expenses and calorie intake, Cal-Get offers a convenient and accessible solution for individuals who want to manage their daily routines effectively.