TerpFit

Project Team 17

- Aryan Shrestha
- Johning To
- William Phoone
- Naaman Trumbull
- Ashley Didriksen

Overview

Terpfit is a one-stop app for individuals at the University of Maryland, College Park who are trying to live healthily and be fit. Terpfit allows you to set goals, track calories, log workouts, and navigate the user to the fitness centers of UMD. Our App offers a variety of features that allow users to manage their health goals effectively.
Our app consists of separate tabs for Summary(Home), Logging Food(Food), Logging Workouts(Workout), Navigation to the nearby fitness-related facilities(Map), and Personal Information(Profile).

Goals

The primary aim of TerpFit was to provide a comprehensive platform where users can set, monitor, and achieve their fitness and dietary goals. Users would be able to customize their daily calorie intake and burn goals, and track their progress in real-time. This app also has navigation to the nearby fitness-related facilities which can help new students as well give them exposure to other fitness facilities such as Ritchie Coliseum and Severn Building Fitness Center.

Home Dashboard

The Home Tab is the first tab that a user sees when they open the app. This tab gives a summary of the User's goals and a visual representation of the user's Calorie Intake and Calories burned. The purpose of this tab is to be a central hub for users to quickly glance at their daily health statistics. There are three cards and a calendar displayed in this tab. The calendar allows the user to pick a date and see their goals and progress made for that specific date. The first card is the goal card, which users can modify with the pencil icon next to the title. The second card is the Calories Intake card which has a linear progress bar showing the progress of calorie intake as well as a text that shows the amount of calories intake remaining. The third card is the Calories Burned card which consists of a circle progress bar that shows the amount of calories that the user has burned after logging in their workouts as well as a text that shows the amount of remaining calories to be burned to achieve the user's goal.

- Features:
    - Display of daily calorie intake and burn through intuitive graphics.
    - Progress bars showing the percentage of goals achieved.
    - Quick navigation on the bottom to other app tabs like Food, Workout, Map, and Profile.

Logging Food

The Food tab allows the user to log the food they consume throughout the day. On the top of the tab, The app displays the current data, as well as the total calories consumed for the day which the app calculates the user logs in their food. Users can log in to their food by clicking on the floating action button with the icon of add which leads the user to a form with input fields Food Name, Calories, and Meal type. The tab relies on the food model state. As the user logs in their food, the state of the food model gets updated. The food is displayed via a listview with 4 meal types. The meal types are Breakfast, Lunch, Dinner, and Snacks. The types can be expanded to show all the food that falls under the type and can be dismissed as well. To remove a food, the user can swipe right on the item.

- Features:
    - Users can add food items for breakfast, lunch, dinner, and snacks.
    - The app calculates and displays the total caloric intake against the daily goal.
    - Allows review and modification of past entries (organized by calendar date).

Working Out and Logging Exercises

The Workout Tab facilitates users in logging various exercises and tracking calories burned. The tab has a floating action button which leads the user to a form. If the user chooses the Cardio as their type of workout the form expands with input fields Exercise name, Duration(minutes), and Calories Burned and if the user chooses Weight Training as their type of workout the form expands with input fields Exercise name, Weight(lbs), Reps and Sets. Once the user logs in to their workout, the item is displayed in a list view with the name as a title along with an icon to delete the workout on the right of the item. The tab relies on the ExerciseData state. The exercises are loaded from the state and when the user logs in an exercise, the state is updated. The exercises and calories burned are converted into JSON and stored in the shared_preferences with the key being the date. When the user opens the app, the state checks if there are exercises stored for the current date, and if yes displays it on the exercise tab.

- Features:
    - Logging of different physical activities and workouts.
    - Calculation of calories burned per exercise for cardio. We were not able to implement calories burned for weight lifting as tracking this metric requires the calculation of many factors not limited to the user’s weight/height, RPE (effort level) during lift, and other structural differences across different body types.
    - Visualization of daily and weekly exercise totals.

Map

The Map tab helps users find nearby fitness-related facilities and plan routes. This tab gives exposure to other fitness centers in UMD to students whom they do not have knowledge of. In order to use the map feature, the user needs to give permission to location services. The location is achieved through Geolocator.getPositionStream() and updates the user's location in real time. This tab consists of a map that is centered to the user's location. There is an arrow on top of the map that gives user directions to the closest fitness-related facilities. The flutter_map package is used to display the map. There are four markers(Eppley, Rictchie, SPH, and Severn) on the map that display the fitness-related facilities. On the bottom of the screen, there is a text that calculates the distance to the closest facility. Our stretch goal consisted of allowing the user to choose the facility they wanted to go to and giving them real-time updates on how busy the facility was. But due to Time constraints we were unable to implement this feature.

- Features:
    - Interactive map showing the user's current location and nearby gyms, parks, and tracks.
    - Functionality to show closed paths or areas and calculate distances to fitness centers.
    - Real-time navigation and an arrow that points and guides the user to the nearest fitness center
    - Dynamic location updates

Profile

The Profile is the final tab of the navigation bar. This tab allows users to manage their personal information and settings. The tab consists of a form that has the input field Age, Gender, Weight(lbs) and Height (inches) and a button to save changes. There are default values set to these fields. When the user modifies the data and saves the changes, it is saved to the shared_preferences so that the data is persistent. There is a BMI calculator under the form that calculates the user's BMI based on the data provided on the form. Additionally, the app shows if the user is underweight or overweight based on the user's BMI. We wanted to display a form when the user first opened the app which removed the use of default values, but due to time constraints we were unable to implement it.

- Features:
    - Input fields for age, gender, weight, and height.
    - Calculation of BMI (Body Mass Index) and other health metrics based on the profile.
    - These options provide a sense of personalization and easy access to health status according to BMI.

Foot Traffic Data

Our goal was to display to the user how busy Eppley and Ritchie were at a specific time using a foot traffic API. It would also suggest which times are the least crowded. We were unable to complete this minimal goal.

In place of this minimal goal, we implemented the map tab instead, which helps users find the closest UMD fitness center and even includes an arrow that updates with the correct direction to guide the user to the closest center.

Personalization of Fitness Results

As a stretch goal, we aimed to personalize fitness results (such as calories burned) based on the user's weight, height, sex, and other factors. Yet, we found that there were additional factors that needed to be included. For example, the intensity of the workout and conditions that the user worked out in if logging a weight lifting exercise. For cardio workouts, the different cardio workouts that the user performed and their intensity. To address the issue with the cardio workouts, we opted to make the user enter the calories burned instead as most gym equipment nowadays that is used for cardio displays the calories burned during use.

In place of this stretch goal and to provide a sense of personalization, the Profile tab allows the user to keep track of their weight and height. The tab also automatically calculates the BMI for a user and will update accordingly to any updates in the values provided.

Using AI to provide recommendations

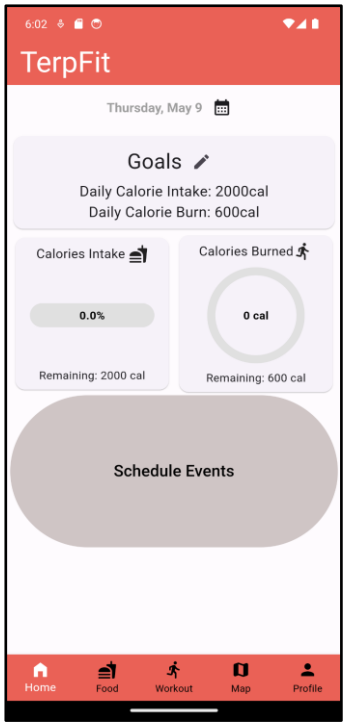
We wanted to use AI to provide recommendations on the user's eating habits and activities based on the data that were logged in. This was another stretch goal that had more logistical difficulties than we anticipated. The need for API keys and paid services seemed unreasonable. Time constraints were also a factor.

Tracking the walking distance of the user

This was a stretch goal we didn't implement primarily due to time constraints. We wanted to create a background service that would track the distance the user has walked for the day and include it as a part of their calories burned but to do so, it would've required much more complex implementations. We also figured that the user could just log their walk as an exercise anyway.

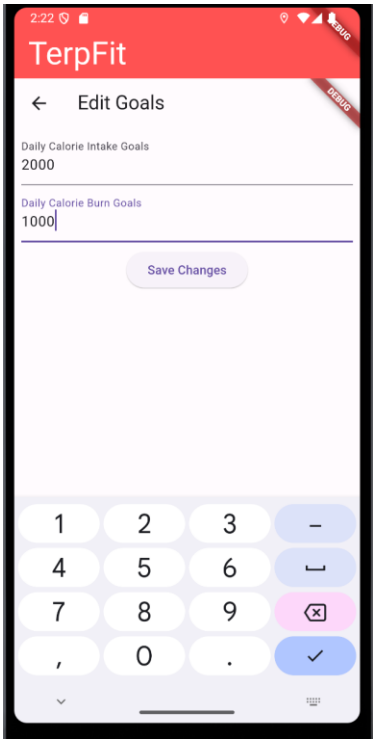
User Interactions

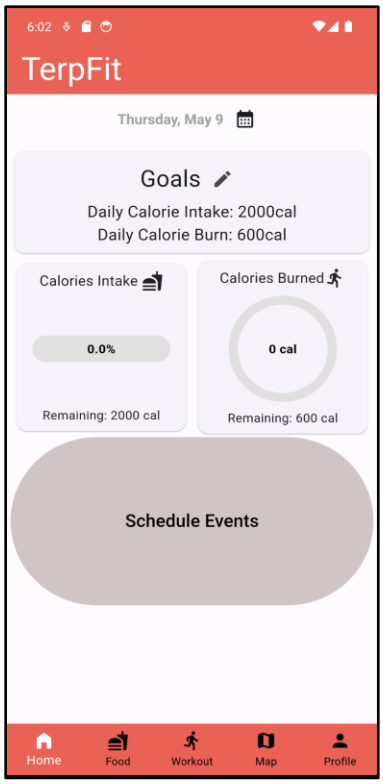
Dashboard



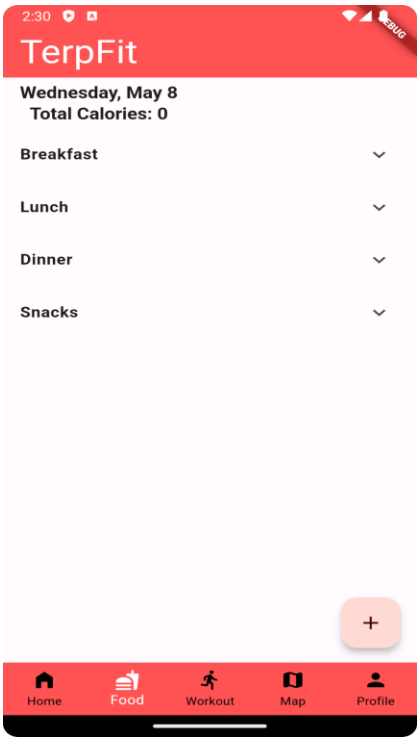
←This is what the application looks like on startup.

Clicking the pencil/edit icon from the startup page next to Goals will allow the user to customize their calorie intake and daily calorie burn goals. After saving their changes, the cards are updated. →



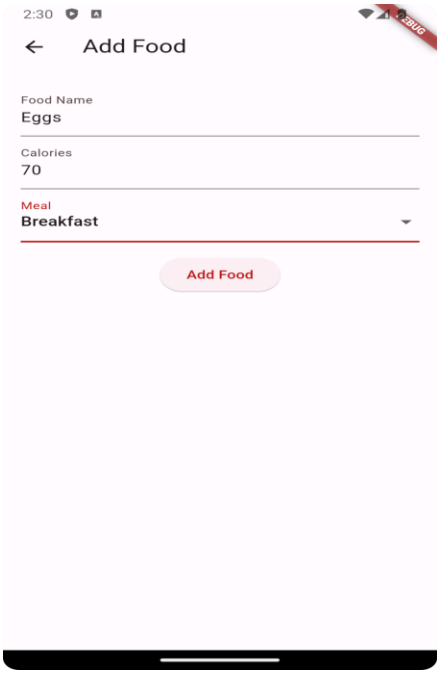


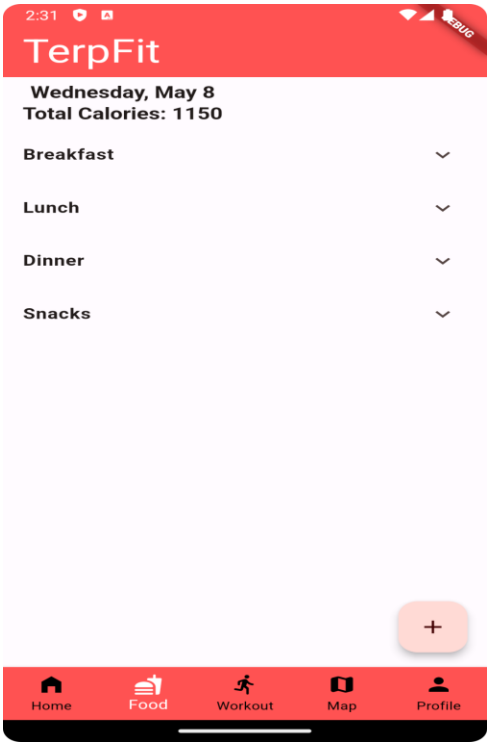
Logging Food



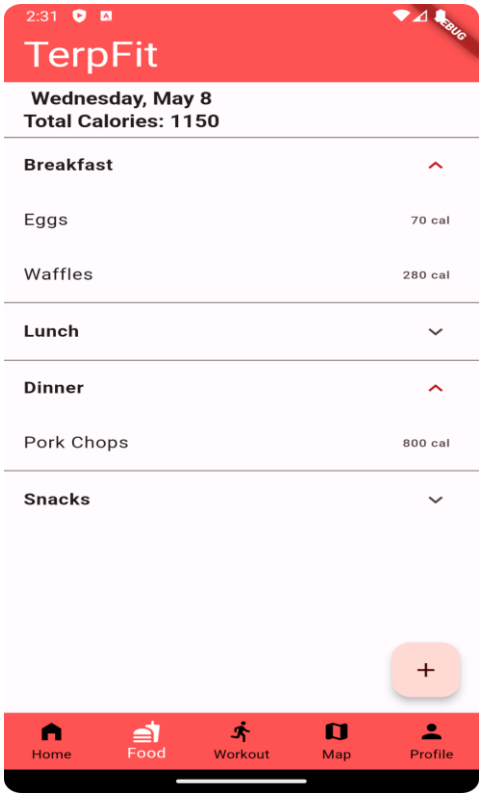
← This is what the user will see the they press on the food icon.

This is the form that appears when the user clicks on the floating action button.

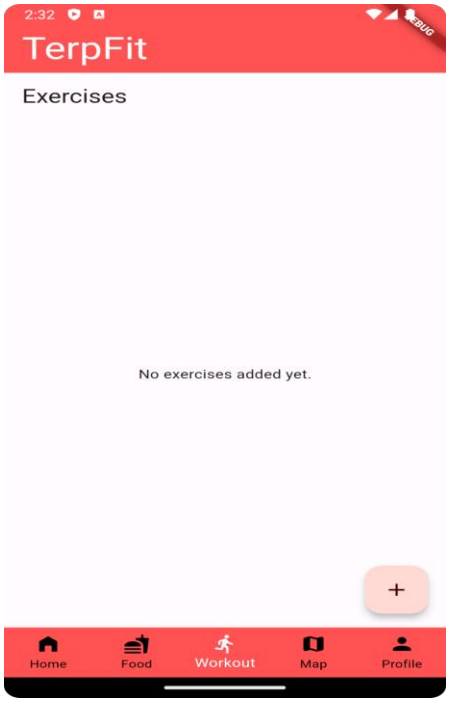




← This is the view when all of the lists are not expanded after logging in food.

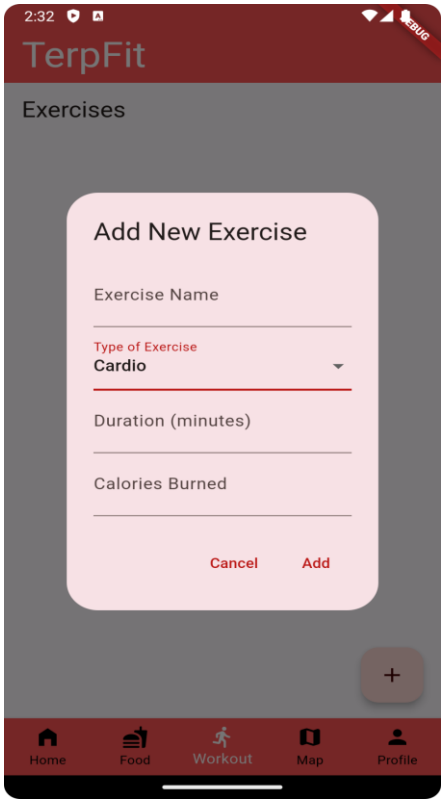


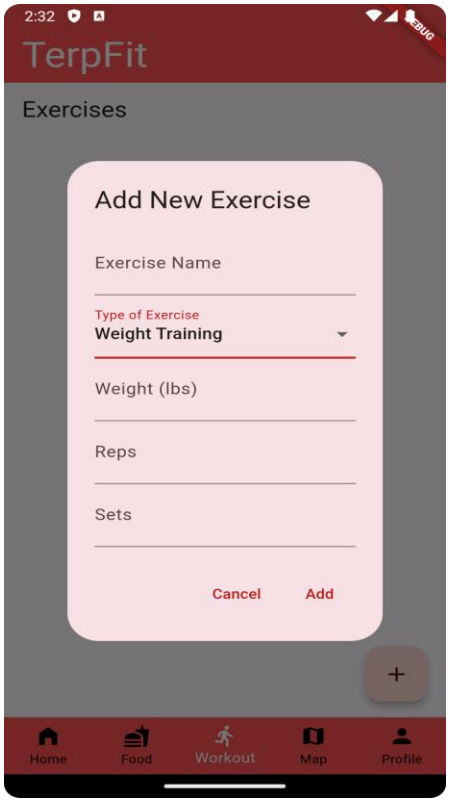
← This is the view when all of the lists are expanded after logging in food.



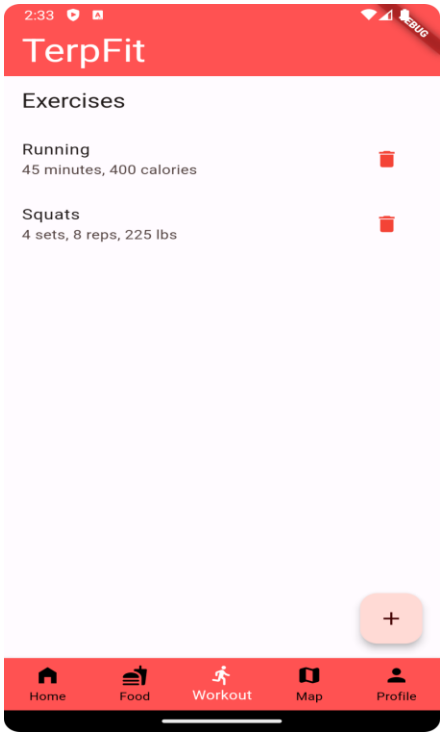
Working Out and Logging Exercises

← This is the view of the exercise tab when the user presses on the workout tab.

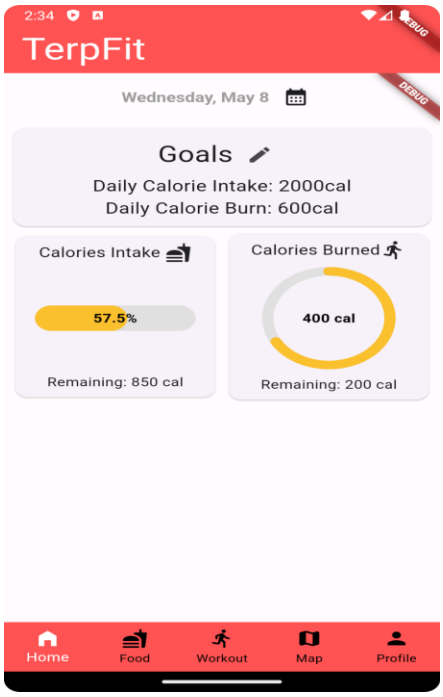




This is the form that appears when the user clicks on the floating action button. The form varies depending on the type of exercise. If the user selects cardio, then the input fields are Duration and Calories Burned. If the user selects Weight Training, then the input fields are Weight, Reps, and Sets.



← This is the view of the workout tab after the user logs in the their exercises. There is a delete icon on the right of the exercises, if the user wants to delete the exercise.

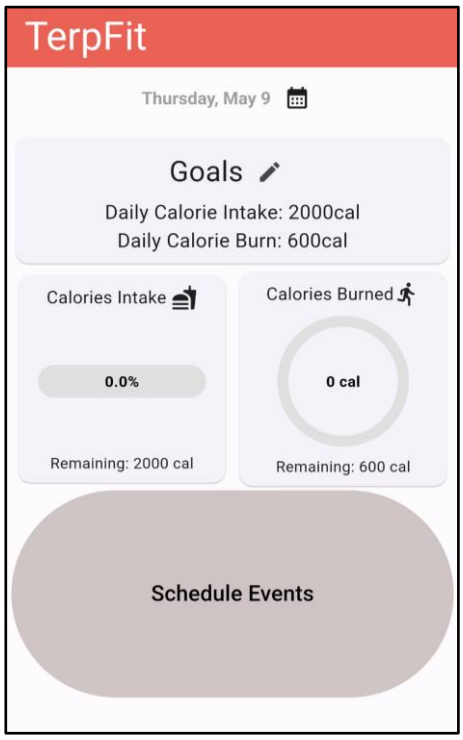


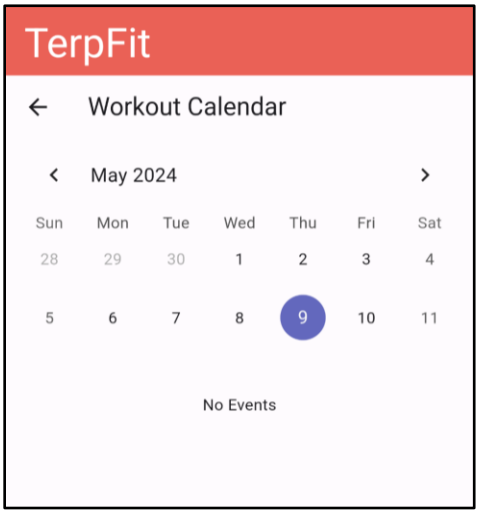
← This is the Dashboard after the user logs in their food intake and workout/exercises.

Map

← This is the view when the user clicks on the map tab. The map shows the user's current location and also weight icons that represent the fitness facilities. On the top of the icon is an arrow that navigates the user to the facility. And on the bottom is the distance (meters) away from the user.

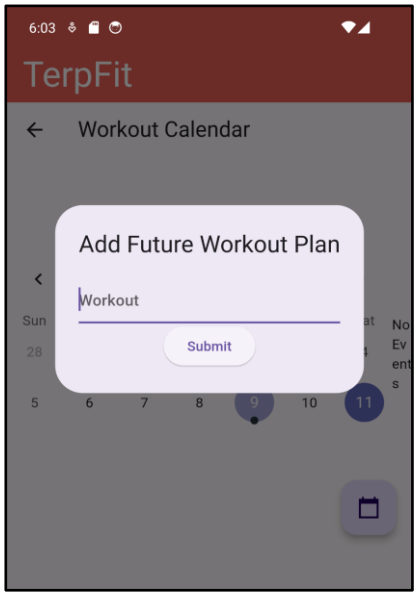
Schedule Events





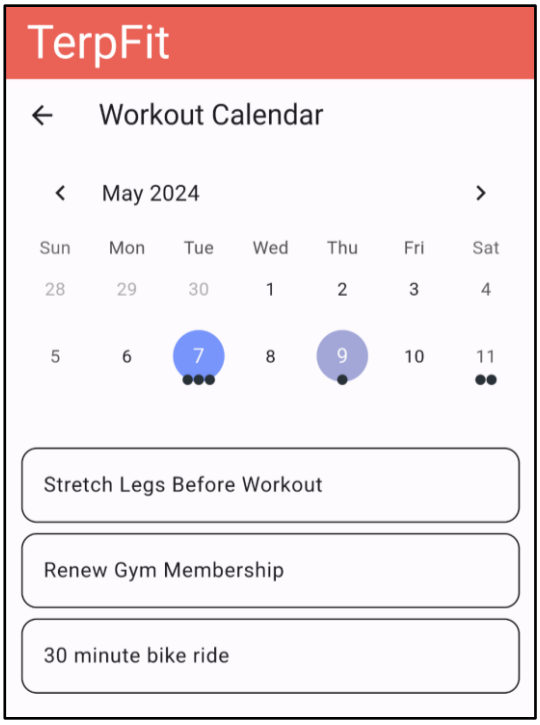
← This is the default calendar page that appears when "Schedule Events" is selected.

← Selecting "Schedule Events" on the Home Tab will open a calendar for users to write memos and make future fitness plans.

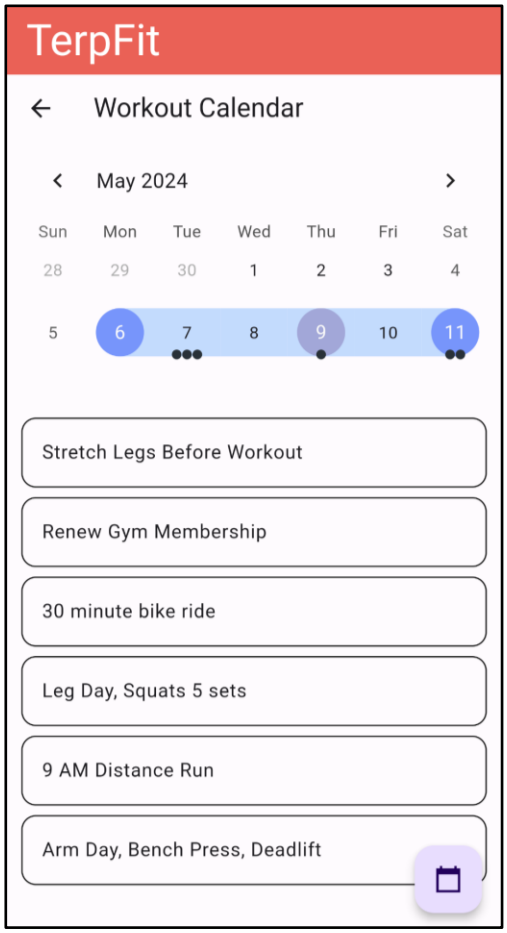


← To add a memo/future workout plan to the currently selected day, select the floating action button with the calendar in the bottom right corner.

← An alert dialog appears with a form. Type the memo in the text field and tap "Submit" to save it in the calendar for the currently selected day.

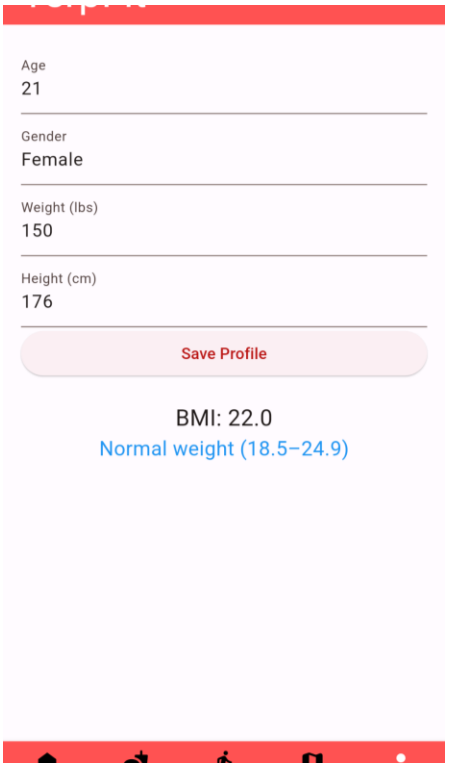


← Users can see a summary of all the memos they have added by looking at the calendar. Dots will appear on specific calendar dates that have had entries added. When a specific date is selected, all the registered memos for that day are listed in a scrollable field next to the calendar.

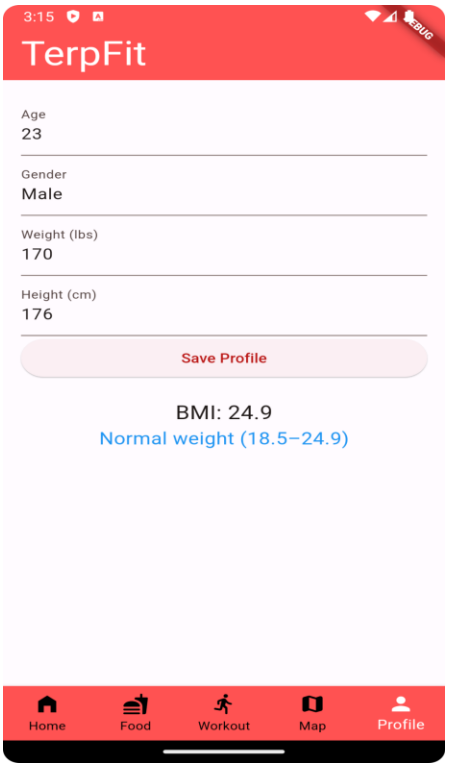


← Holding down on a date will turn on range selection. When range selection is on, tapping on a second date will highlight all days in the range (inclusive). Every memo that appears in the selected range will be listed.

Profile



← This is the view when the user clicks on the profile tab. There are default values set to this form. The form consists of Age, Gender, Weight(lbs) and Height(cm). Additionally, this page also consists of a BMI that gets calculated based on the information provided on the form and if the user is Underweight, Normal, Overweight or Obese.



← This is the Profile tab after the user updates their data. The BMI is recalculated.

Development Process

Our development process began with planning out our initial design document and outlining the goals for our application as well as the features we wanted to implement. From there we met as a team and had our members pick which parts of the project they wanted to work on so that we could split the workload amongst the different tabs of the application. By using git we were able to split the development by each tab (Home, Food, Workout, Map, Profile) and had a different person working on each tab before merging them together into the new base application branch. This allowed us to divide the workload of the project without forcing each person to rely on the progress of the other group members, meaning everyone was able to work at their own pace on their assignment.

Future Direction

Potential future directions for this project could involve going back and implementing stretch goals that we either didn't have the time or resources to add to the application. Among these potential future directions are features like AI integration to help the user get recommendations and workout plans customized to their specific needs. This was an initial idea we had when planning out the application, but the logistics of actually implementing it as a feature would have been more involved than potentially the rest of the project combined. In the end, we decided against attempting to implement it in a rushed or poorly designed manner, and to simply note that it was an idea we didn't end up moving forward with.