Project Documentation: Fitness Portal

Contents

Introduction 3

Empowering Your Fitness Journey 3

A Digital Wellness Partner 3

The Need for Nutritional Awareness 3

Analyzing Habits for a Healthier Future 3

User Documentation 4

1. User Authentication 4

1.1 Creating an Account 4

1.2 Logging In 5

1.3 Logging Out 6

2. Food Journal 7

2.1 Logging Food Intake 7

2.2 Viewing Food Logs 8

3. Graphical Analytics 9

3.1 Visualizing Today's Calorie Consumption 9

3.2 Visualizing Calorie Consumption over Weeks or Months 11

4. Miscellaneous 13

4.1 Changing to Dark Mode / Light Mode 13

Developer Documentation 14

1. Project Structure 14

1.1 Root Directory 14

1.2 Source Directory 14

1.3 Configuration Files 14

1.4 Documentation 14

1.5 Build and Deployment 15

2. Architecture Overview 15

2.1 Motivation 15

2.2 Implementation details 16

3. Authentication 17

4. Future improvements 17

Introduction

Fitness Portal is a Blazor-based fitness web application designed to empower users in tracking their daily nutrition. Utilizing MudBlazor components for a rich user interface and Entity Framework for efficient data management, the Fitness Portal aims to provide a comprehensive platform for health enthusiasts.

Empowering Your Fitness Journey

In the hustle and bustle of modern life, it's easy to overlook the critical role nutrition plays in our overall health. Recognizing this, Fitness Portal is not just an application; it's a companion in your journey towards a healthier lifestyle. With a user-friendly interface, the app makes it effortless for users to take charge of their nutritional habits and transform their well-being.

A Digital Wellness Partner

The primary objective of Fitness Portal is to be your digital wellness partner. In a world inundated with fast-paced living, it becomes imperative to have tools that seamlessly integrate into our lives, fostering healthy habits. This application, with its secure user authentication, detailed food journal, graphical analytics, and personalized BMI calculator, is designed to empower users with the insights and motivation needed to make informed choices about their health.

The Need for Nutritional Awareness

In an era where convenience often trumps consciousness, it's crucial to pay attention to what we eat. Modern lifestyles, characterized by sedentary routines and fast-food culture, have made it challenging for individuals to maintain a balanced diet. Fitness Portal addresses this need, recognizing that a deeper understanding of nutritional habits is the first step towards positive change.

Analyzing Habits for a Healthier Future

Fitness Portal excels in providing users with the freedom to analyze their nutritional habits. Through insightful visualizations of daily calorie intake categorized by food type and weekly or monthly breakdowns by day, users gain a profound understanding of their dietary patterns. This data-driven approach empowers individuals to make informed choices, fostering a healthier and more balanced lifestyle.

As we embark on this fitness journey together, let Fitness Portal be your trusted companion, guiding you towards a healthier, more vibrant life. Your well-being is our priority, and we are excited to be part of your pursuit of a healthier and happier you.

Welcome to Fitness Portal—**where health meets innovation**.

User Documentation

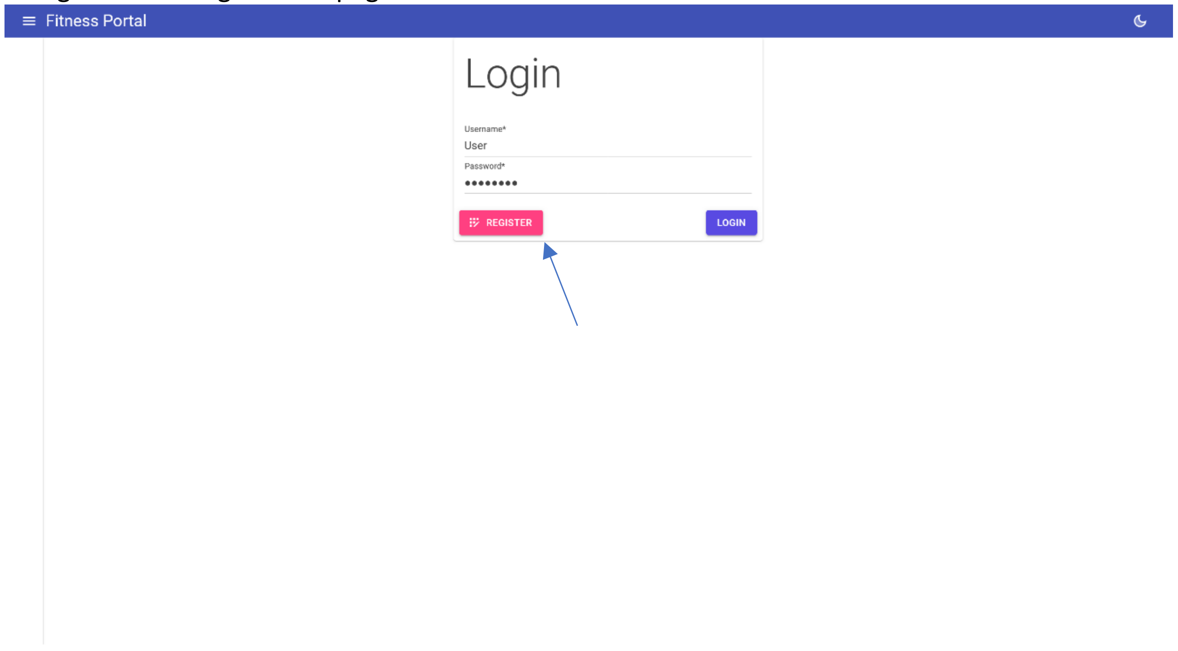
Welcome to the Fitness Portal! This section provides guidance on how to use the application effectively to manage your fitness journey. Whether you are a seasoned health enthusiast or just getting started, this user guide will help you make the most out of the features offered by the Fitness Portal.

1. User Authentication

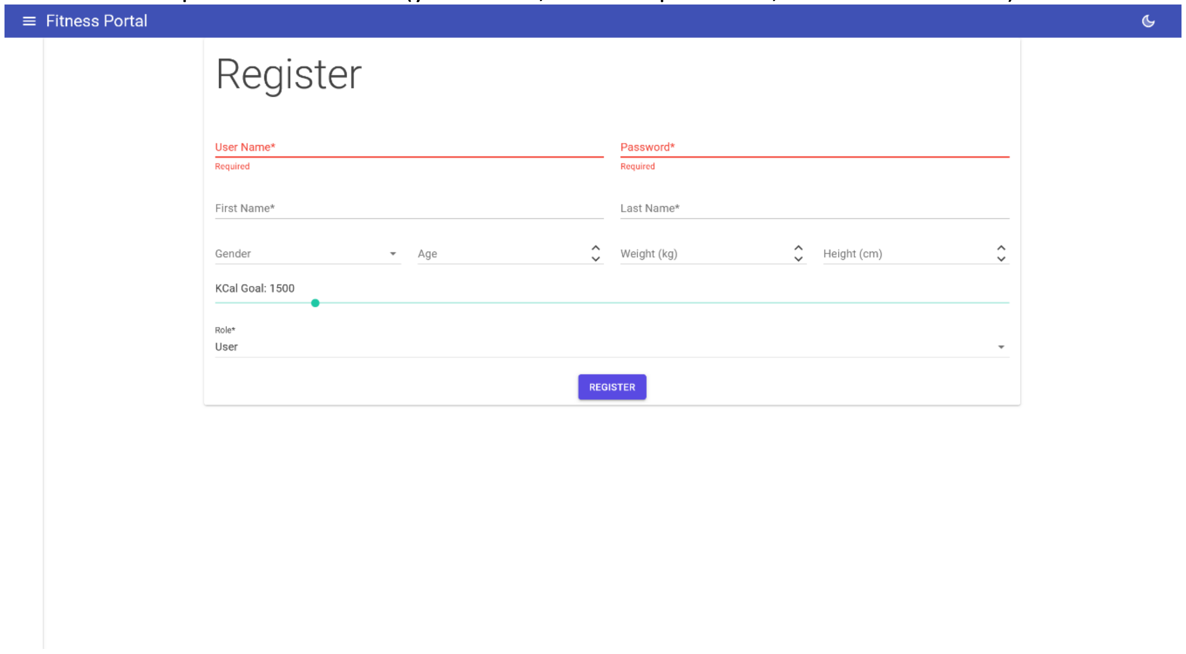
1.1 Creating an Account

To access the features of the Fitness Portal, you need to create an account. Follow these steps:

1. Navigate to the registration page.



2. Provide the required information (your email, a secure password, first and last name).

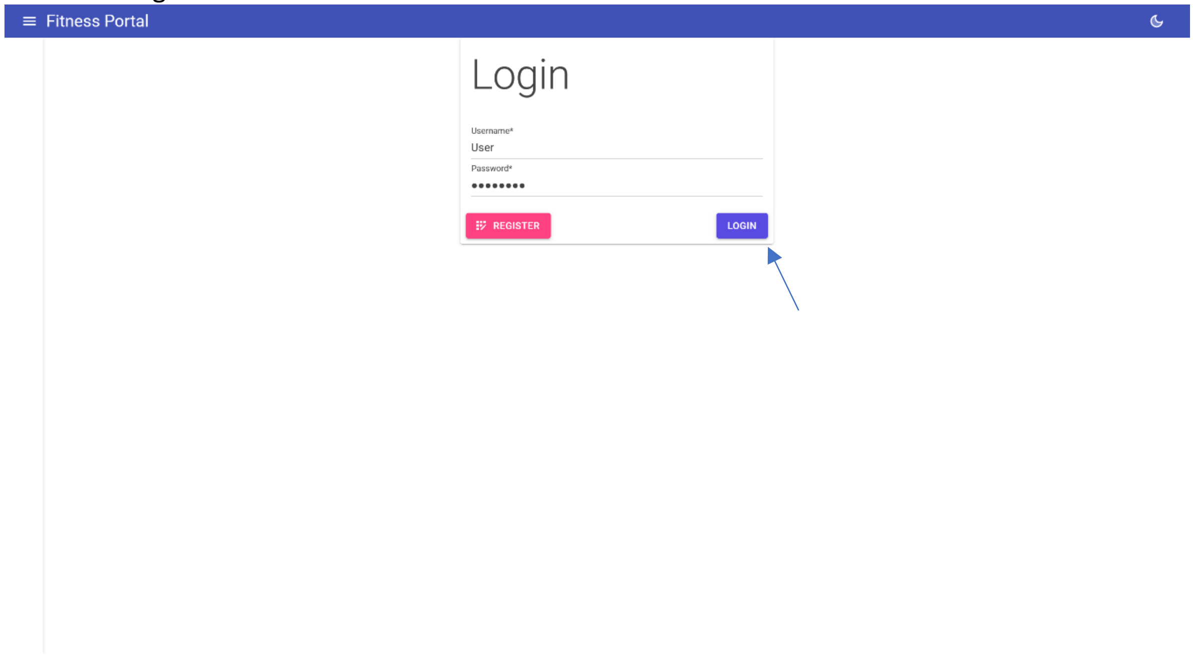


3. Click the "Register" button to create your account. This will redirect you back to the Login page.

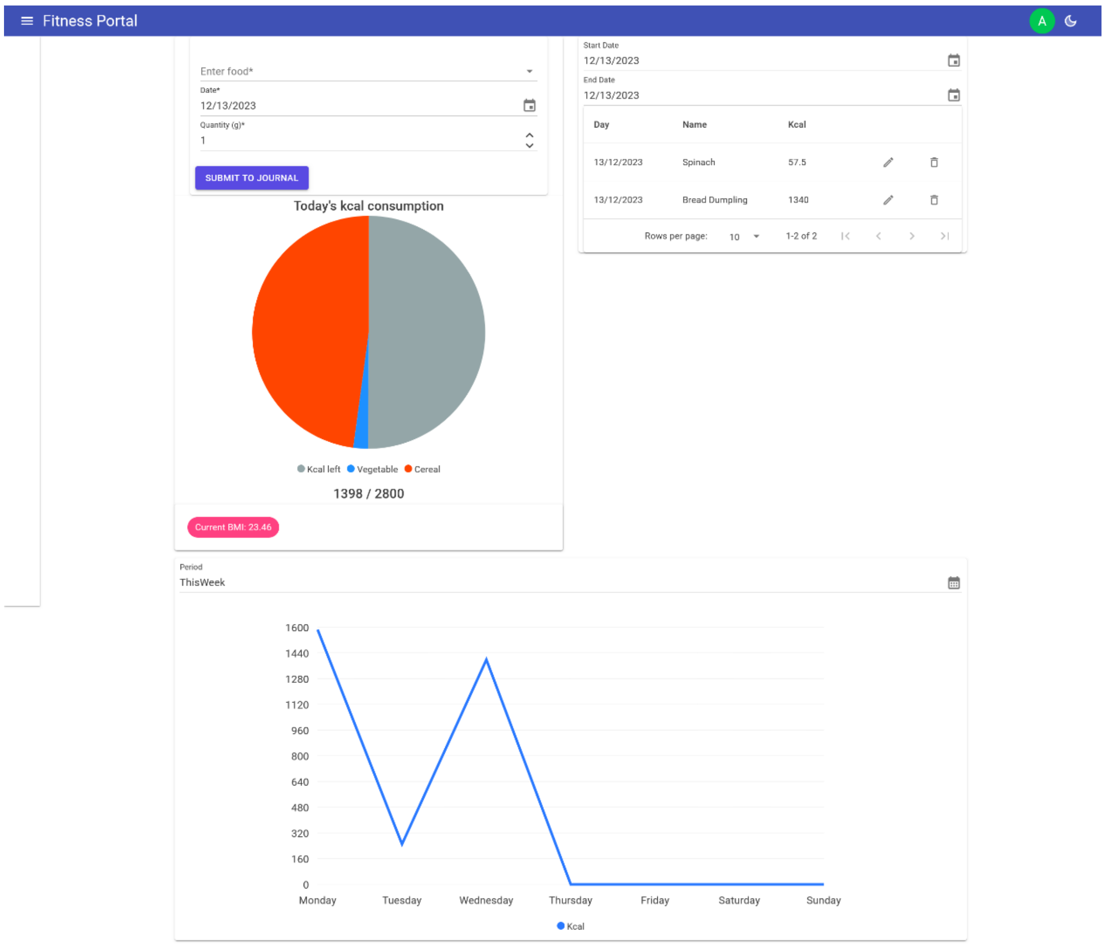
1.2 Logging In

Once you have an account, log in to access your personalized fitness data:

1. Enter the app (redirects to login page should the user be unauthenticated).
2. Enter your registered username and password.
3. Click the "Login" button.



4. You will be redirected now to the home page and main hub of the app.



Fitness Portal

Enter food*

Date*
12/13/2023

Quantity (g)*
1

SUBMIT TO JOURNAL

Start Date
12/13/2023

End Date
12/13/2023

Day	Name	Kcal
13/12/2023	Spinach	57.5
13/12/2023	Bread Dumpling	1340

Rows per page: 10 1-2 of 2

Today's kcal consumption

● Kcal left ● Vegetable ● Cereal

1398 / 2800

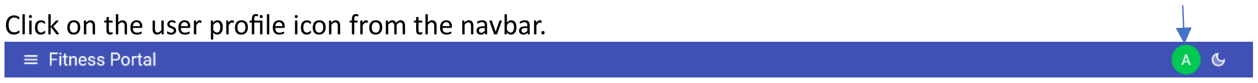
Current BMI: 23.46

Period
ThisWeek

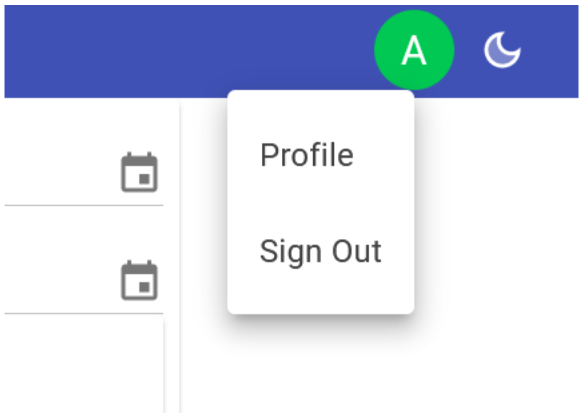
1.3 Logging Out

To log out of your account:

1. Click on the user profile icon from the navbar.



2. Select "Sign Out" from the dropdown menu.



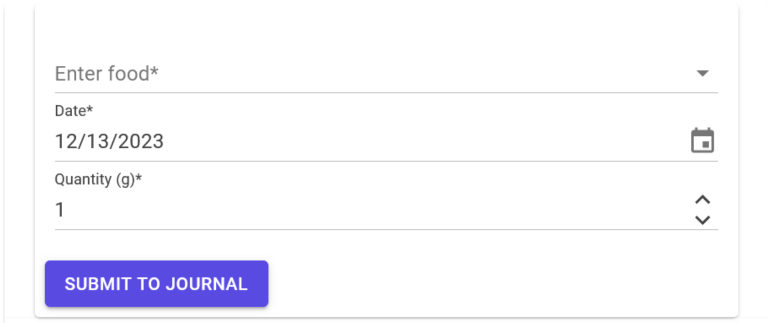
3. The app will return you to the login page.

2. Food Journal

2.1 Logging Food Intake

Keep track of your nutritional intake by logging your meals:

1. Navigate to the Food Addition section from the home page.

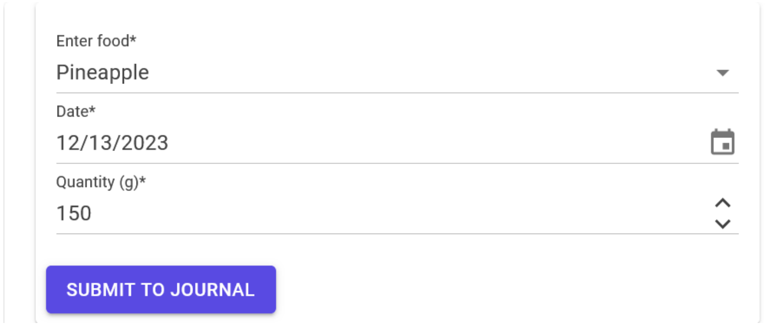


Enter food*

Date*
12/13/2023

Quantity (g)*
1

2. Enter the name of the food you ate, the date in which you did and the quantity in grams.



Enter food*
Pineapple

Date*
12/13/2023

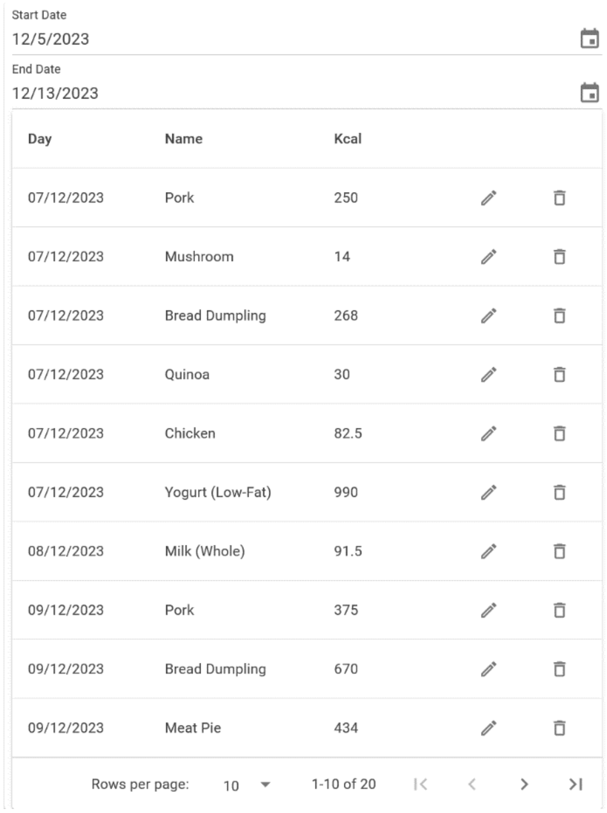
Quantity (g)*
150

3. Click on the "SUBMIT TO JOURNAL" button and the entry will be added to the Food Journal.

2.2 Viewing Food Logs

To review your food logs:

1. Navigate to the Food Journal section from the home page.



Start Date
12/5/2023

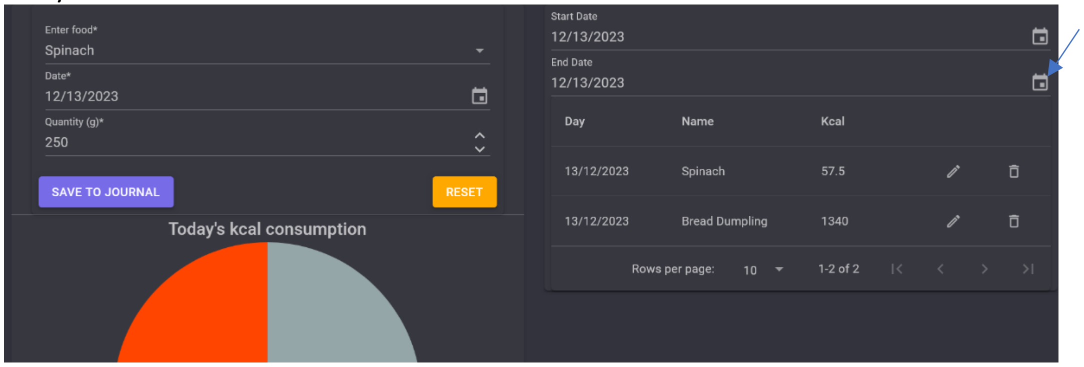
End Date
12/13/2023

Day	Name	Kcal
07/12/2023	Pork	250
07/12/2023	Mushroom	14
07/12/2023	Bread Dumpling	268
07/12/2023	Quinoa	30
07/12/2023	Chicken	82.5
07/12/2023	Yogurt (Low-Fat)	990
08/12/2023	Milk (Whole)	91.5
09/12/2023	Pork	375
09/12/2023	Bread Dumpling	670
09/12/2023	Meat Pie	434

Rows per page: 10 1-10 of 20

2. View a summary of your calorie intake by date, during the time interval chosen with the Date Pickers located above the table.
3. Click on the Edit button of individual entry to see edit it in the Food Add section, which will be used for editing until reset. Example:

   A. Having these 2 entries in the journal, if I decide to change the quantity of Spinach, I can press the Edit button and the Food Addition section will contain the information of the entry.



Enter food*
Spinach

Date*
12/13/2023

Quantity (g)*
250

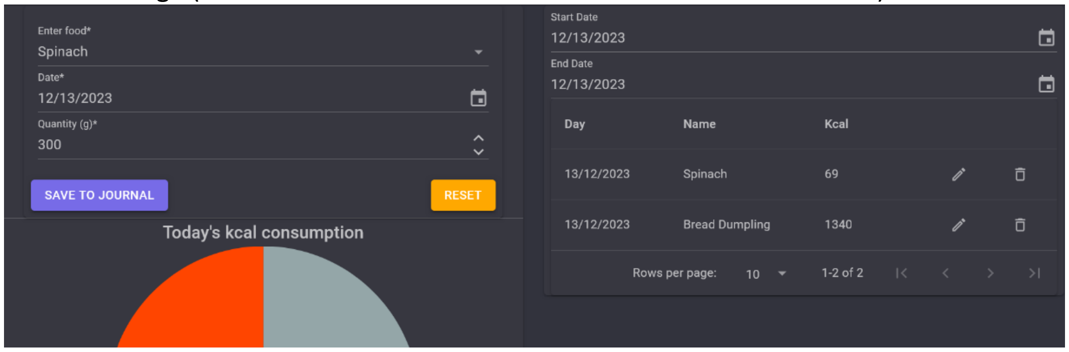
Start Date
12/13/2023

End Date
12/13/2023

Day	Name	Kcal
13/12/2023	Spinach	57.5
13/12/2023	Bread Dumpling	1340

Rows per page: 10 1-2 of 2

B. Changing the quantity to 300g and pressing the "SAVE TO JOURNAL" button enables the desired change (the kcal value from the table is increased from 57.5 to 69).



Enter food*: Spinach
Date*: 12/13/2023
Quantity (g)*: 300

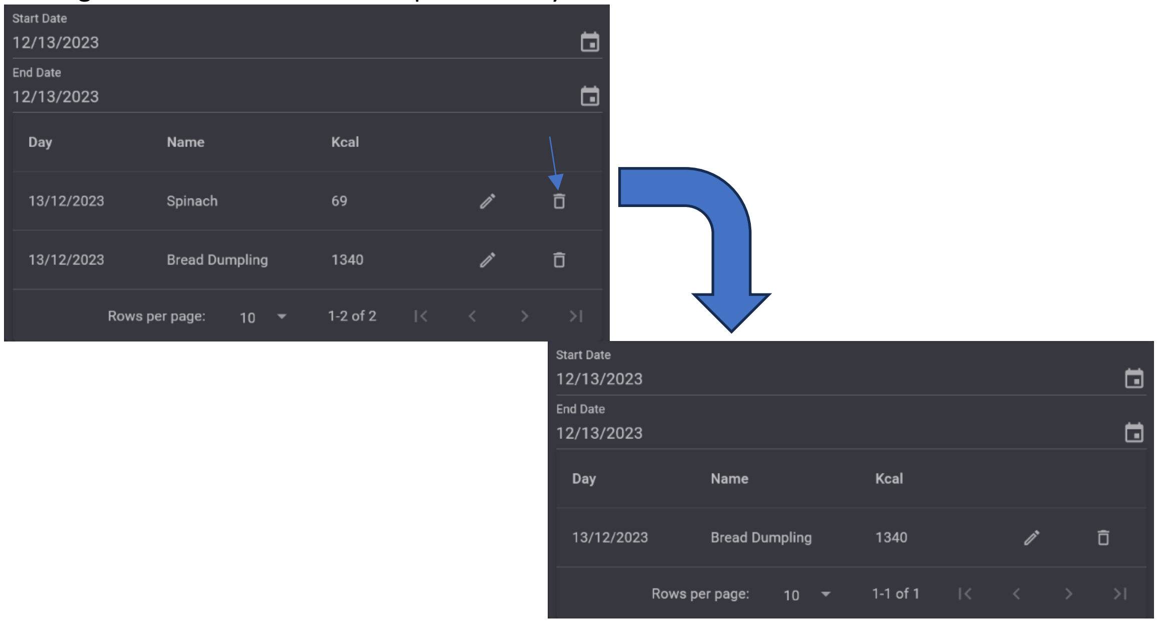
Start Date: 12/13/2023
End Date: 12/13/2023

Day	Name	Kcal
13/12/2023	Spinach	69
13/12/2023	Bread Dumpling	1340

Rows per page: 10 1-2 of 2

C. Pressing the "RESET" button will reset the Food Addition section to it's default state.

4. Click on the Delete button of individual entry to delete it from your Food Journal. For example, clicking the Delete button on the Spinach entry will remove it.



Start Date: 12/13/2023
End Date: 12/13/2023

Day	Name	Kcal
13/12/2023	Spinach	69
13/12/2023	Bread Dumpling	1340

Rows per page: 10 1-2 of 2

Start Date: 12/13/2023
End Date: 12/13/2023

Day	Name	Kcal
13/12/2023	Bread Dumpling	1340

Rows per page: 10 1-1 of 1

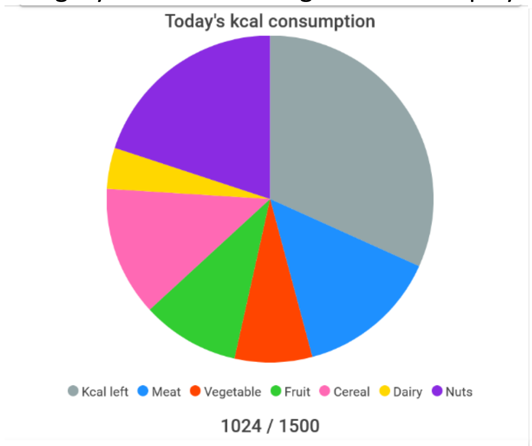
3. Graphical Analytics

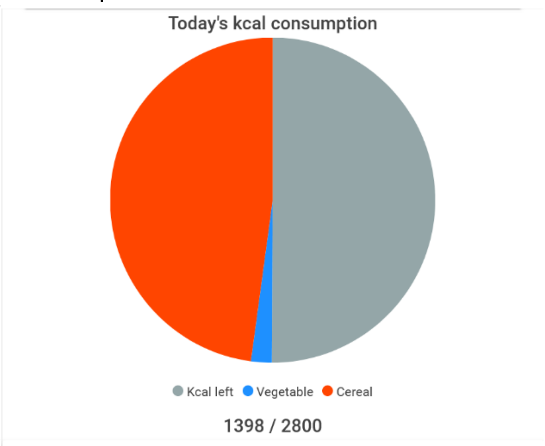
3.1 Visualizing Today's Calorie Consumption

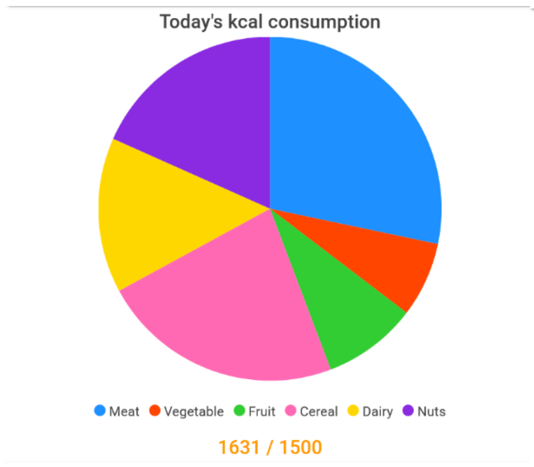
Explore the graphical representation of your calorie consumption during the current day:

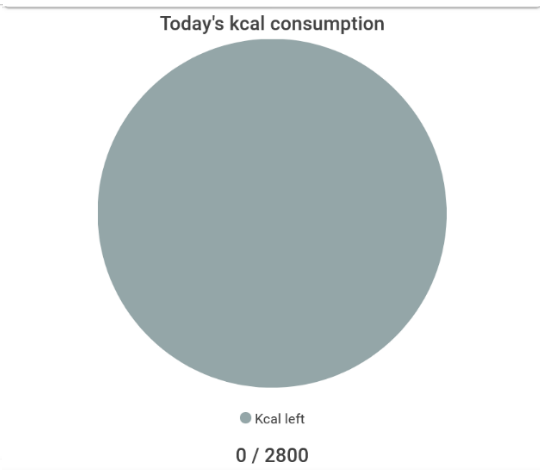
1. Navigate to the Pie Chart section from the home page.

2. There you will automatically have the chart display today's calorie consumption by each main category. The set calorie goal is also displayed. Examples:









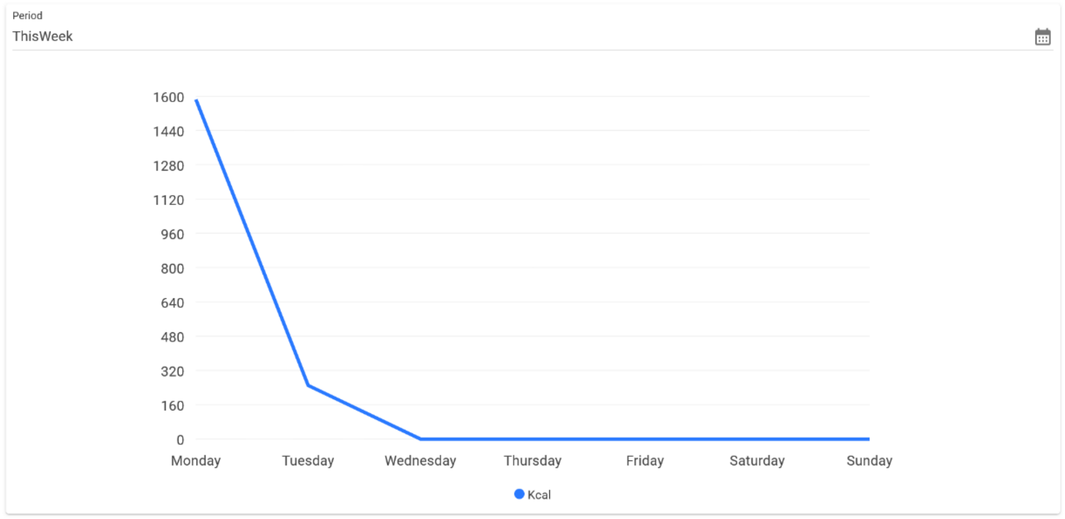
3.2 Visualizing Calorie Consumption over Weeks or Months

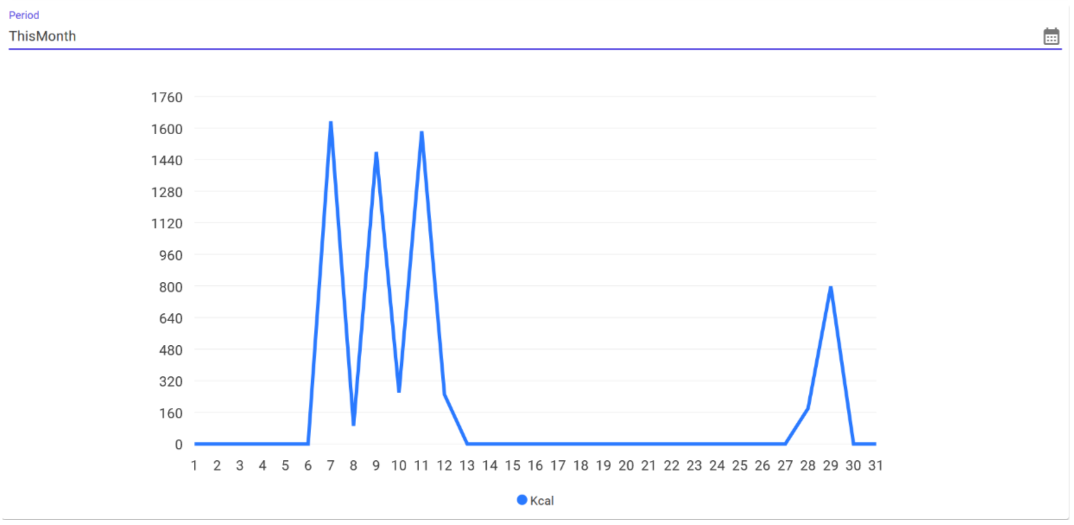
View the graphical representation of your calorie consumption during:

- This week
- Last week
- This month
- Last month

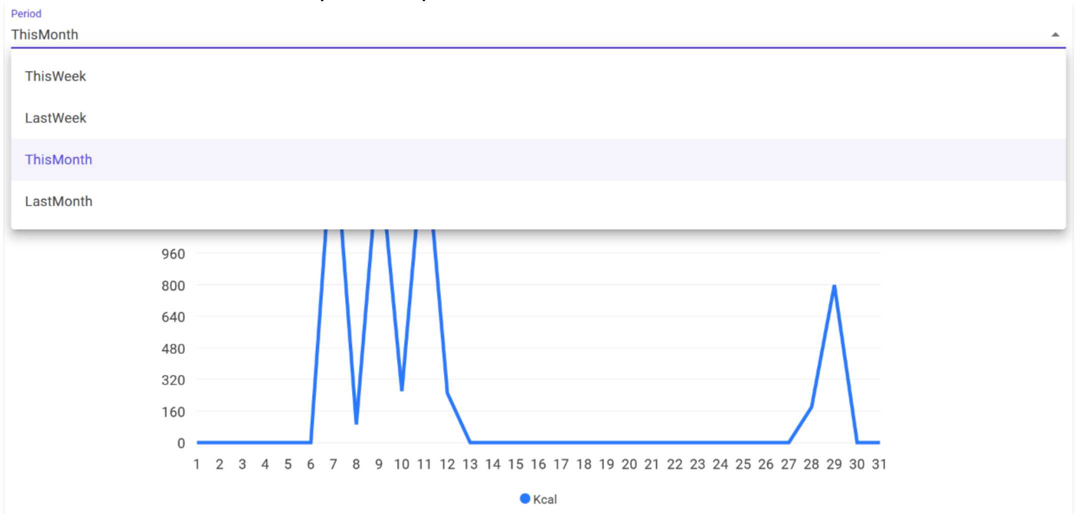
To visualize the bar chart:

1. Navigate to the Bar Chart section from the home page.





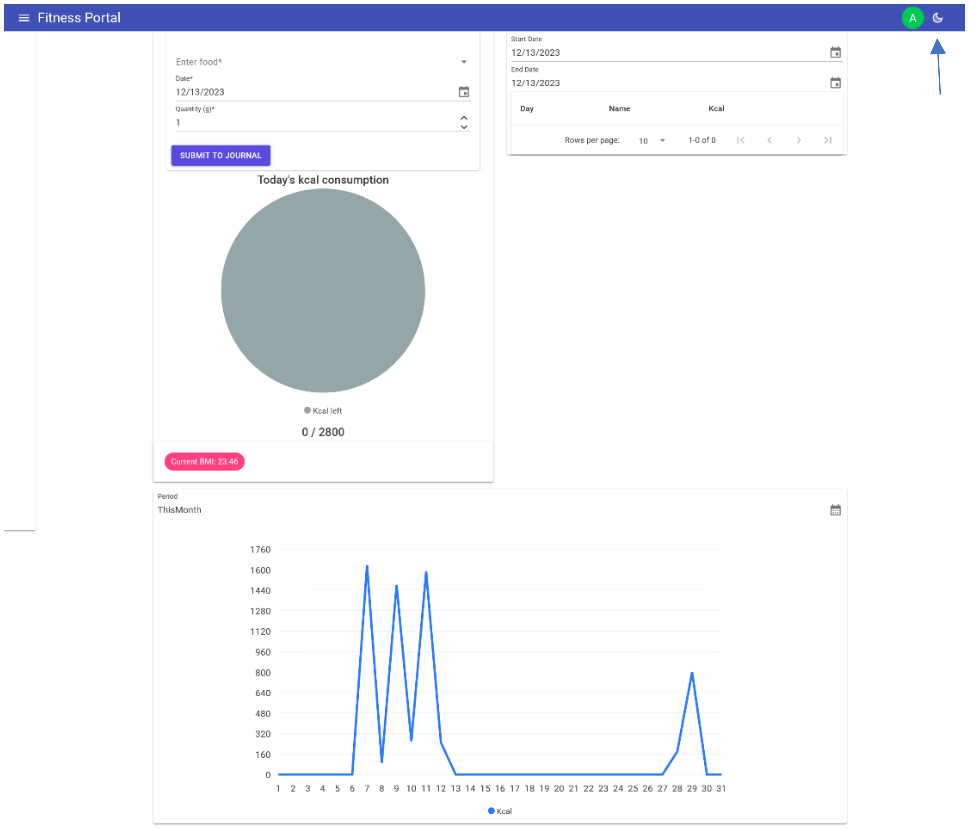
2. You can select one of the specified periods.



4. Miscellaneous

4.1 Changing to Dark Mode / Light Mode

By pressing Moon / Sun icon from the navbar.



Fitness Portal

Enter food*

Date*
12/13/2023

Quantity (g)*
1

SUBMIT TO JOURNAL

Start Date
12/13/2023

End Date
12/13/2023

Day	Name	Kcal

Rows per page: 10 1-0 of 0

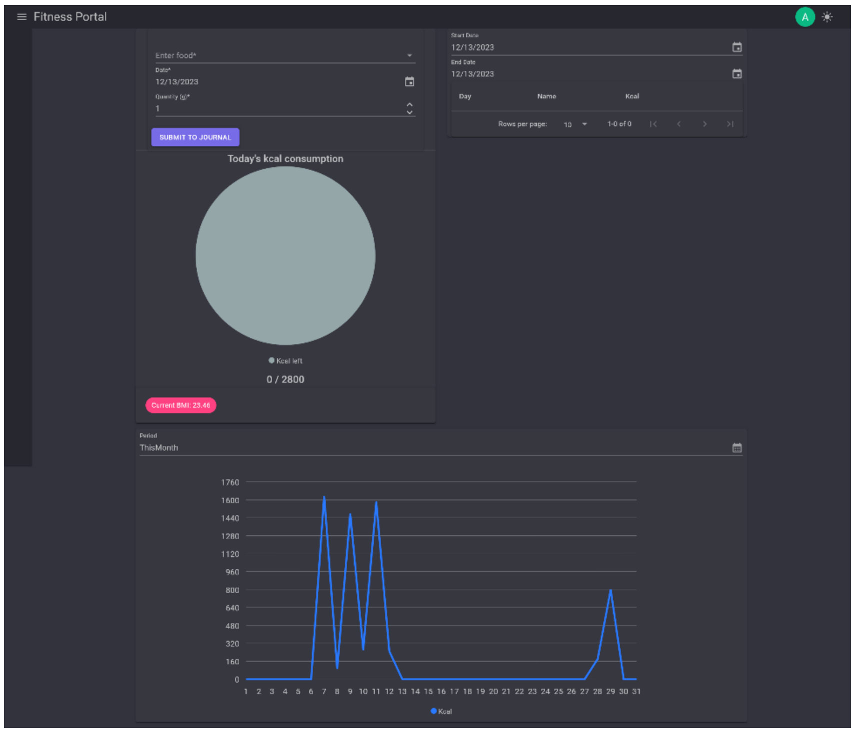
Today's kcal consumption

Kcal left

0 / 2800

Current BMI: 23.46

Period
ThisMonth



Fitness Portal

Enter food*

Date*
12/13/2023

Quantity (g)*
1

SUBMIT TO JOURNAL

Start Date
12/13/2023

End Date
12/13/2023

Day	Name	Kcal

Rows per page: 10 1-0 of 0

Today's kcal consumption

Kcal left

0 / 2800

Current BMI: 23.46

Period
ThisMonth

Developer Documentation

Welcome, developers! This section provides comprehensive information to help you understand the Fitness Portal project. Whether you are a seasoned developer or just getting started, this documentation will guide you through the various aspects of the project.

1. Project Structure

Understanding the organization of the Fitness Portal project is important for developers to navigate the codebase efficiently. The project structure has been designed with modularity and maintainability in mind. Here's an overview of the key directories and their purposes:

1.1 Root Directory

- **`FitnessPortal.sln`**: The solution file for the entire project.
- The source code is directly situated in the root directory.

1.2 Source Directory

- **`/ FitnessPortal`**: This directory contains the source code of the Fitness Portal application.
    - **`/Authentication:`** includes the custom authentication state provider implementation
    - **`/Data`**: Data access layer, including Entity Framework context, configurations, entities and DTOs.
        - **`/Entities:`** Data models used in the application
        - **`/DTOs:`** DTO models used in the application
    - **`/Migrations`**: Database migration files for Entity Framework.
    - **`/Components:`**The razor components of the application, including the layout, pages, routing and imports
    - **`/Services:`** controllers handling *server*-side logic
    - **`/wwwroot`**: Static files, such as images and stylesheets.
    - **`/Utils`**: contains Enums and other classes that are useful

1.3 Configuration Files

- **`appsettings.json`**: Application-level configuration settings.
- **`Program.cs`**: Configuration of services and middleware during application startup.

1.4 Documentation

- **`/Documentation`**: Project documentation, including user guides and developer documentation.

1.5 Build and Deployment

- **`Properties`**: launchSettings.json file is a configuration file used to configure various aspects of how our application should be launched and debugged during development
- **`.github/workflows`**: GitHub Actions workflows for CI/CD.

2. Architecture Overview

Understanding the architecture of the Fitness Portal project is crucial for developers to grasp the underlying design principles and make informed decisions when adding or editing the code. This section provides an overview of the architectural components, including the role of Blazor components, MudBlazor integration, and Entity Framework for data management. The motivation behind using this architecture is based on a few principles:

2.1 Motivation

2.1.1 Real-Time Interaction with Server-Side Blazor

- **Responsive User Experience:** Blazor Server-Side provides a responsive and interactive user experience. With server-side rendering, the UI logic is executed on the server, and the updates are efficiently sent to the client. This reduces the need for heavy client-side processing and ensures a smoother user experience.
- **Rich User Interfaces:** Server-Side Blazor allows for the creation of rich, dynamic interfaces with minimal client-side scripting. This is beneficial for developers who are more comfortable working with C# and server-side logic, making it easier to maintain and develop features.

2.1.2 Entity Framework for Data Management

- **Abstraction Over Database Operations:** Entity Framework provides a high-level abstraction over database operations, allowing developers to interact with the database using C# objects. This simplifies data access code and promotes a more intuitive and maintainable codebase.
- **Code-First Approach:** Entity Framework Code-First approach used in Fitness Portal allows developers to define the data model using C# classes. This approach aligns well with the object-oriented principles, making it easier to represent business entities in code and translate them into a relational database.
- **Migrations for Database Evolution:** Entity Framework migrations enable developers to evolve the database schema over time. This is essential for a project like Fitness Portal, where data models may change or new features are introduced, ensuring that the database schema stays synchronized with the application's requirements.

2.1.3 Seamless Integration with MudBlazor Components

- **Reusable UI Components:** The use of MudBlazor components enhances the user interface with reusable, customizable, and responsive UI elements. This component library integrates well with Blazor and complements the server-side architecture, contributing to the overall consistency and aesthetics of the application.

2.2 Implementation details

2.2.1 Blazor Components

Formally referred to as Razor components, the Blazor Components are reusable, encapsulated elements that can include both UI and logic (Razor is a syntax for combining HTML markup with C# code)

- **Pages**: Represent different sections of the application, such as the `Login.razor`, `Registration.razor` or `Home.razor`. Each page is associated with a corresponding URL route.
- **Layout**: `MainLayout.razor` file which serves as a component present on all pages, one that organizes the layout of the web application
- **Route Definitions:** `Routes.razor` contains route definitions and is also used in assuring secure authentication-based access to the app.
- **Imports:** `Imports.razor` is used for importing namespaces and making them available throughout the Blazor application. It is a convenient place to centralize all the using directives (namespaces) that are commonly used across multiple components, thereby avoiding redundancy.

2.2.2 Entity Framework Integration

- **Database Context**: The ApplicationDbContext class manages database connections and transactions. It includes DbSet properties representing different entities in the application: Users, UsersDetails, FoodsNutrition, FoodsJournal.
- **Data Models**: Entity Framework entities represent the data structures used in the application. These models define the database schema and relationships between different entities. The app contains the following models now:
    - **User** – ID, UserName, Password, Role and associations to UserDetails and FoodJournals
    - **UserDetails** – ID, FirstName, LastName, Gender, Age, Weight, Height, BMI, KcalGoal, UserID and association to User
    - **FoodNutrition** – ID, Name, Category, Kcal and association to FoodJournals
    - **FoodJournal** – ID, Quantity, KcalTotal, Date, UserID, FoodNutritionID and associations to User and FoodNutrition

2.2.3 Interaction Between Layers

The architecture of Fitness Portal encourages a modular and decoupled design, promoting maintainability and extensibility. The interaction between different layers involves:

- **Services and Entity Framework**: Business logic and data access operations are encapsulated in services that interact with the Blazor components and Entity Framework data layer. All queries to access the database are made by using LINQ. The app uses services for the following purposes:
    - *User account service* (`IUserAccountService.cs`): responsible for retrieving User information for the Blazor components and adding or updating Users and UsersDetails to the database.
    - *Food service* (`IFoodService.cs`): retrieves FoodNutrition entries.
    - *Food journal service* (`IFoodJournalService.cs`): gets, adds, or updates FoodJournal entries.
- **Dependency Injection**: Use of dependency injection to inject services into Blazor components, promoting loose coupling and testability.

3. Authentication

In this Blazor application, a robust and tailored authentication system is implemented through a custom authentication state provider, `CustomAuthenticationStateProvider.cs`. This class is responsible for managing the authentication state of users, seamlessly integrating it into the overall application flow. Additionally, a `UserSession.cs` class encapsulates the essential user session data.

- The heart of the authentication mechanism lies within the `GetAuthenticationStateAsync` method. This method is tasked with asynchronously retrieving the current authentication state based on the user session. It utilizes a `ProtectedSessionStorage` instance injected during the class's instantiation to securely store and retrieve user session data.
    - Upon invoking this method, it attempts to fetch the `UserSession` from the protected session storage. If the retrieval is successful, it constructs a `ClaimsPrincipal` containing crucial claims such as the user's name, role, and identifier. In the case of an unsuccessful retrieval, it gracefully returns an anonymous `ClaimsPrincipal`. The resulting `AuthenticationState` is then provided to the application.
- To further enrich the user experience, the application facilitates the `AuthenticateUser` method within the `CustomAuthenticationStateProvider`. This method plays a pivotal role in authenticating the user. It accepts a `UserSession` object, representing the user's session data. When a non-null `UserSession` is provided, it stores this session data securely using the protected session storage. It then crafts a `ClaimsPrincipal` based on the user session data, incorporating essential claims.
    - Conversely, when a null `UserSession` is passed, the method deletes the existing session data from storage and employs an anonymous `ClaimsPrincipal`. Notably, this method ensures that the application is notified of any authentication state changes, ensuring a seamless and dynamic user experience.
- The `UserSession.cs` class serves as a clear representation of the essential user session data. With properties such as `ID`, `Name`, and `Role`, it encapsulates the information necessary for the authentication process.
- The passwords are stored using SHA256 hashes.

4. Future improvements

Among the features that were not implemented in the current version, the following are worth noting:

- **Sleep Journals**: users should be able to add entries in their journals, where they can insert the length and quality of each sleep session; users should be able to analyze their sleeping habits through charts.
- **FoodNutrition adding as Admin**: users who have the admin role should be able to insert food items into the FoodNutrition table using a separate section of the app.
- **BMI-based Calorie Goal Recommendation**: users should be able to benefit from a Kcal Goal Recommendation based on the calculated BMI if they inserted their weight and height.