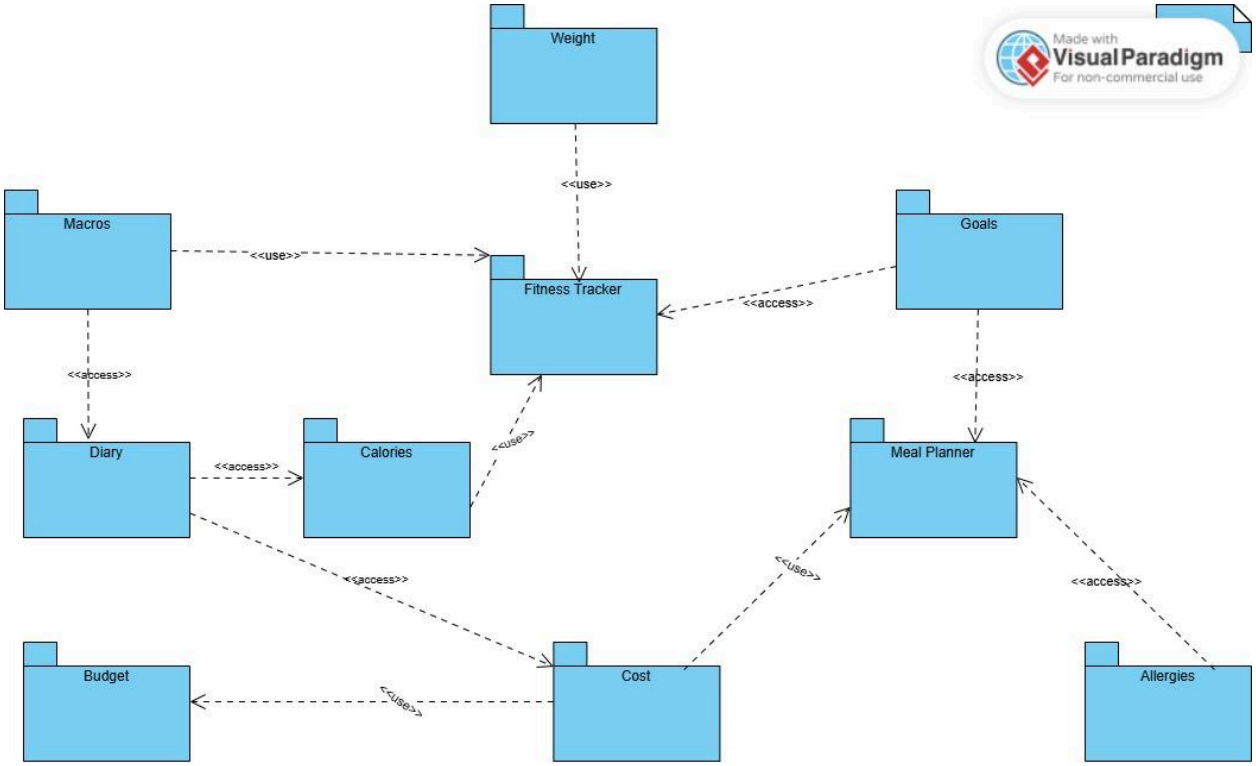
1. Description

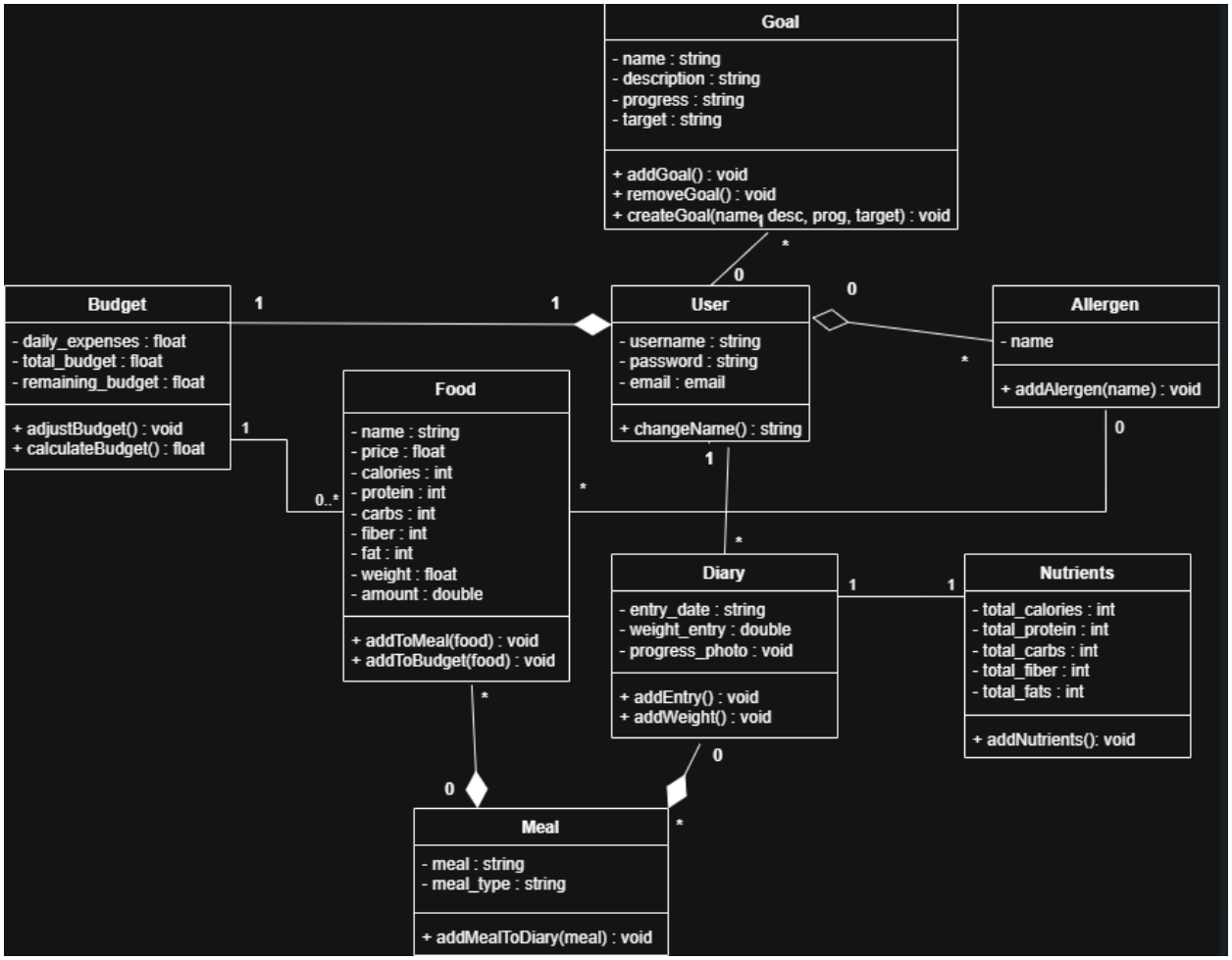
A common problem that many college students have had in the past and will have for the foreseeable future is the issue of juggling personal nutrition, budgeting, and a stressful academic life. Living a life that is full of these stresses can impact the mental and physical health of students in our modern age. With ‘CalPal’, we want these stressors to get thrown out, alongside the many empty containers of cup noodles. As such, our website will offer the ability for health-conscious college students to improve their overall wellbeing by both managing their time and their health, whether financial or physical. Our website will also provide the ability for those looking to maintain or even improve their physical health to set user-specific goals that cater to whatever they desire.

Our system achieves these goals by implementing a system to track fitness goals, calories, macros, allergies and budget to make maintaining health easier for students. For the fitness goals and allergies the system keeps track of these by providing several toggleable options that affect the website's suggestions. For the calories and macros trackers the website keeps track of the meals a user enters into the site and provides statistics about their eating habits as well as their progress towards specific fitness goals. The budget tracker allows for a user to set a budget and log their spending in order to better manage their meal spending.

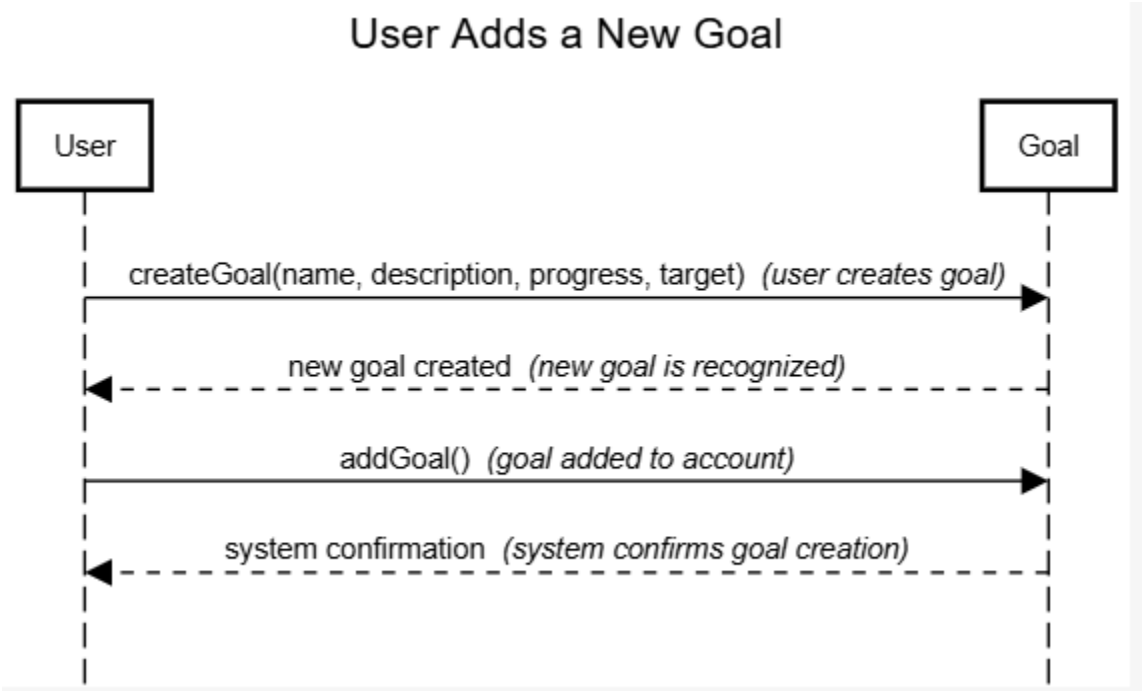
2. Architecture



3. Class diagram



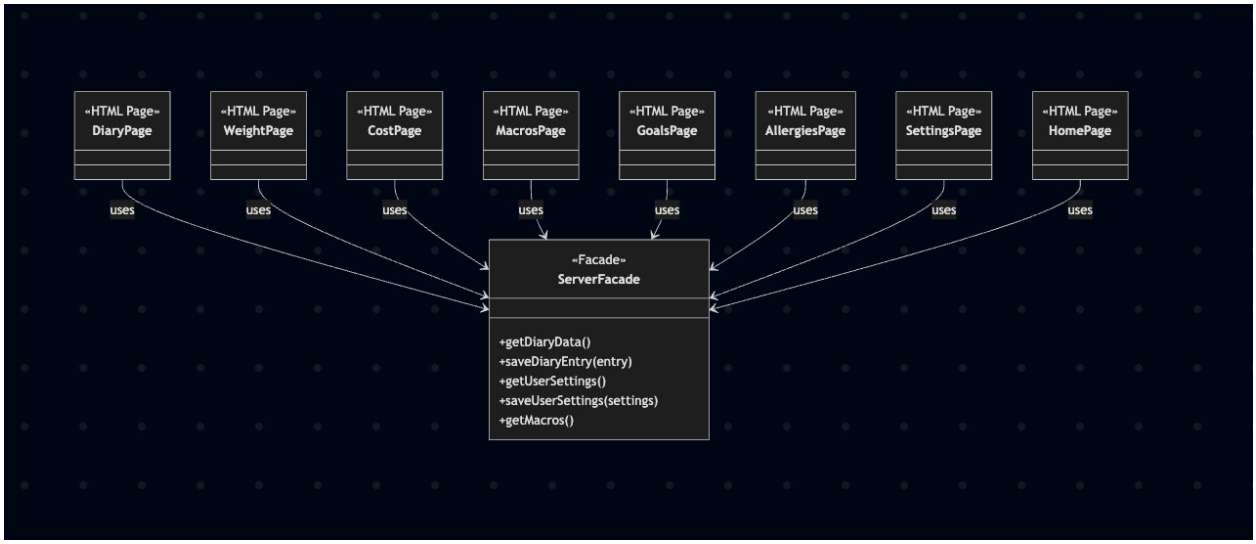
4. Sequence diagram



5. Design Patterns

Pattern 1

Pattern name: Facade (multiple subsystems on a single interface)
Intent: The Facade pattern allows an interface that is objectively simple. Each page only uses a set of functions and nothing more.
Where: This is seen in the index.html, diary.html, weight.html, cost.html, macros.html, settings.html, and app.js.
UML:
The reason each page of our webpage follows a facade pattern is because they are working with just one file, that app.js file. This file allows every page to use buttons, forms, loads data, accesses the api, etc. Each file doesn't have to have its own unique and complex code.

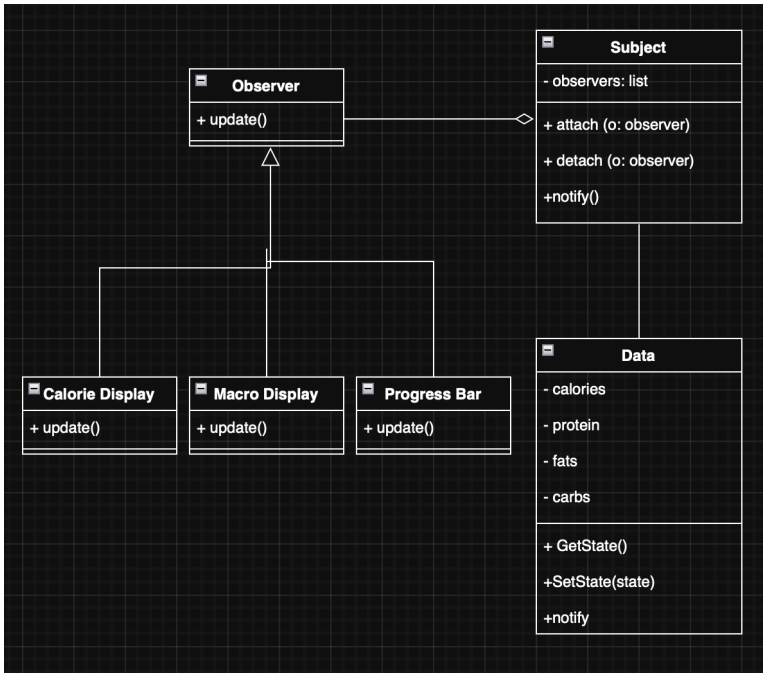


Class diagram:
Allergies page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/allergies.html
Cost page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/cost.html
Diary page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/diary.html
Goals page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/goals.html
Macros page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/macros.html
Settings page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/settings.html
Weight page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/weight.html
Home page:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/server.js
Page formatting:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/index.html

Pattern 2

Pattern name: Observer (changing one thing affects all of the other things)
Intent: The Observer pattern allows the user to see what is affected by dietary or exercise choices they make and record. Each change is reflected in the other categories.

Class diagram:



Progress:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/goals.html
Macros:
https://github.com/loganb7869/CS386-Project-Repo/blob/main/CS_386/macros.html
Data :https://github.com/loganb7869/CS386-Project-Repo/blob/main/server.js

6. Design Principles

1.) Single responsibility principle: Each class or module has a single functionality.

2.) Don't repeat yourself principle: Calculations for our meal plan app are in a single section and aren't repeated.

3.) Simplicity - KISS (Keep It Simple, Stupid) Our meal plan app has a very simple homepage design with each option being outlined and direct.