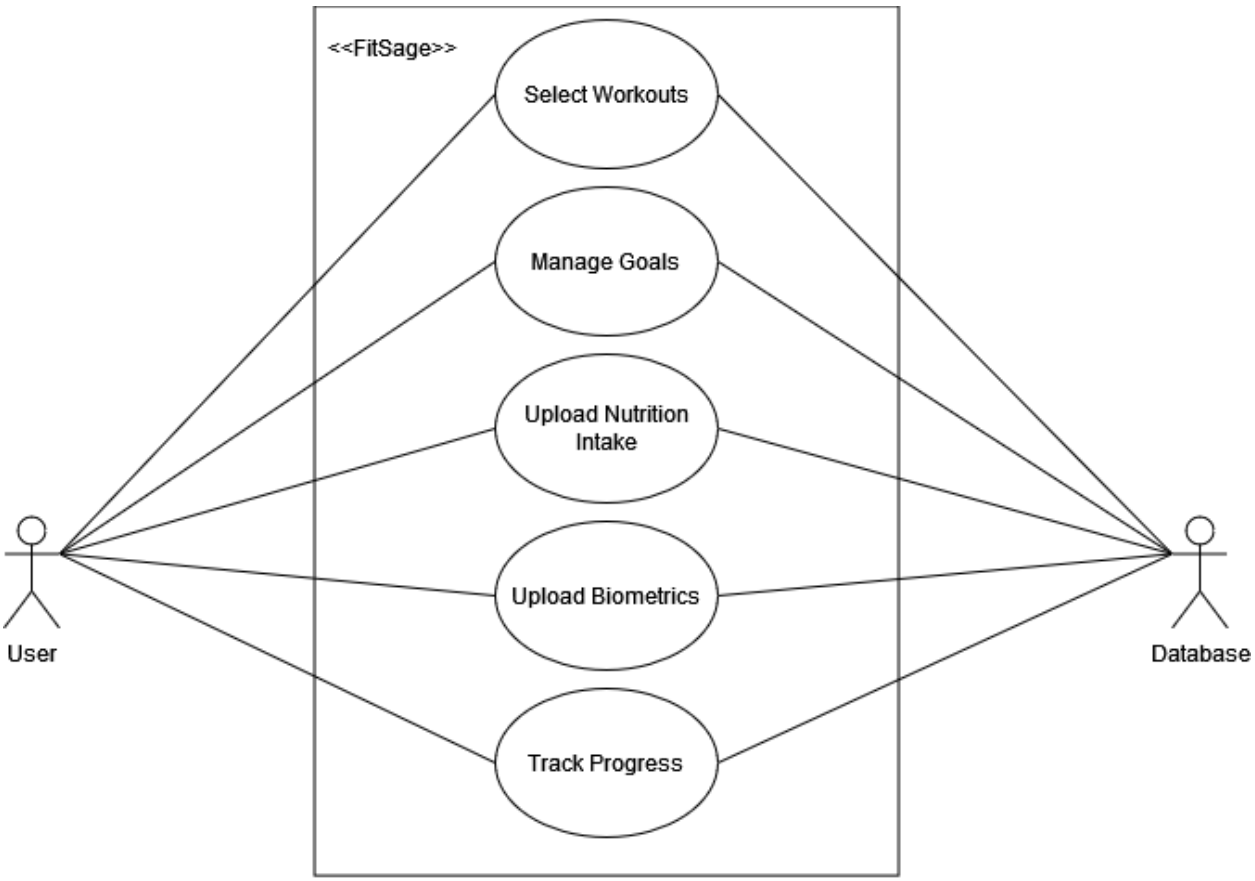
Project Outline Document

Project Name: FitSage

Team Name: Team 11

Members: Ryan Ferguson, Daniel Machado, Jim Pamplona, Jack Lee

Use Case Diagram:



Use Case Descriptions:

UC1: Select Workouts

Description: A user wishes to work out using the FitSage app. The user will have already uploaded their biometric information and selected a goal. The user will select muscle groups they wish to exercise, and the app will provide a list of recommended workouts involving those muscle groups. The recommended list will include information regarding that workout, and the user will be able to interact with workouts in the list.

Actors: User, Database

UC2: Manage Goal

Description: A user wants to select a goal to follow or change their current goal. The user may or may not have already uploaded their biometric information. The app will update the database with the user's current goal.

Actors: User, Database

UC3: Upload Nutrition Intake

Description: A user wishes to upload nutrition information from food they ate. The user will have already uploaded their biometric information by creating their account. The app shall display the calories from the Nutrition Facts label so that the user can verify that the information was correctly interpreted. The calories will be added to the daily total in the database.

Actors: User, Database

UC4: Upload Biometrics

Description: A user wishes to upload or update their biometric information (age, height, weight), and the database stores the information. The user may or may not have already uploaded their biometric information. The app shall calculate the user's Body Mass Index and Body Fat Percentage metrics. The app shall display them on the screen and upload them along with the biometric information to the database.

Actors: User, Database

UC5: Track Progress

Description: A user wishes to see their progress towards their chosen goal. The user will have already uploaded their biometric information and selected a goal. The app will display the user's daily calorie benchmark as well as their past history. The app will also display the user's current streak and past streaks, if applicable. The information to be displayed will be accessed from the database.

Actors: User, Database

Context Diagram:

Context Diagram Description:

Source: Phone Screen

Description: The Phone Screen is the interface the user uses to access the application. The application displays information to the user, and the user can touch buttons and navigation areas within the application to change the contents displayed on the screen. The user can also input biometric information, such as their age, height, and weight, onto the screen. The screen will relay that information to the app, which will add it to the Database.

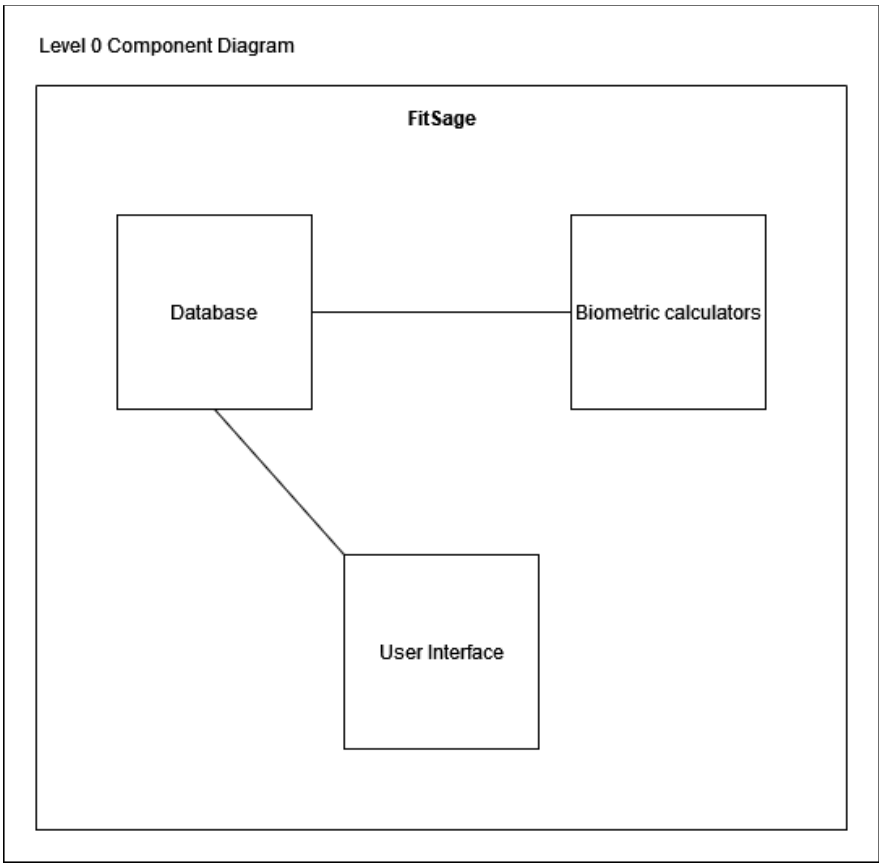
System: FitSage App

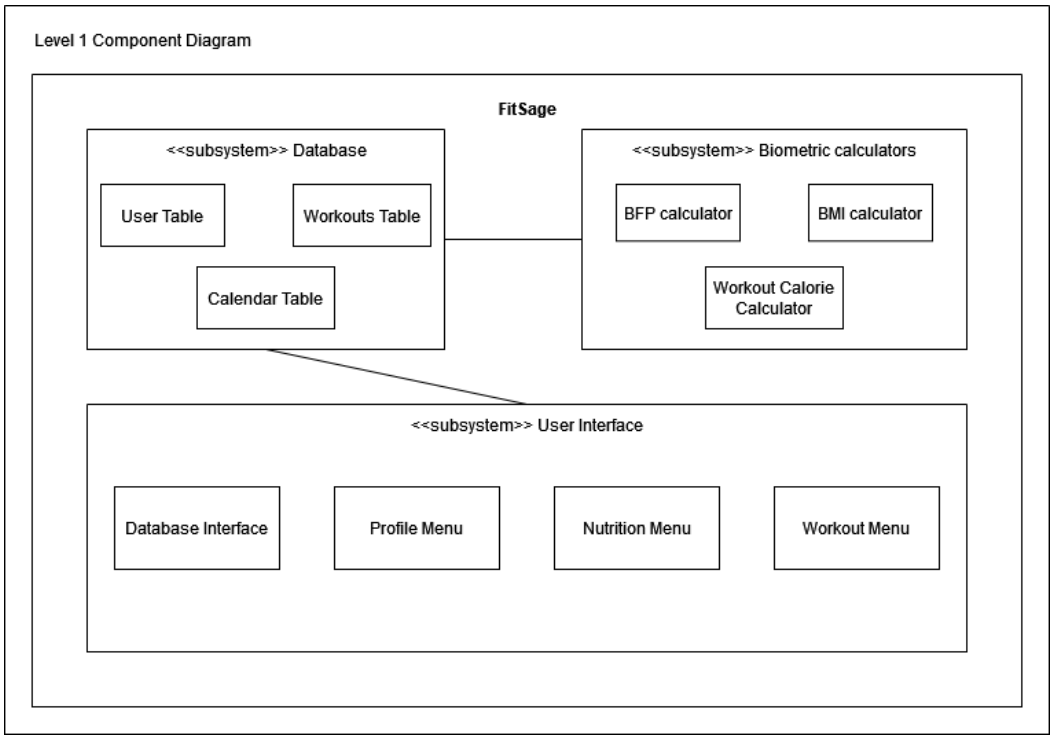
Description: The FitSage App is the main system we are developing for this project. It interprets the user's input through the Phone Screen and sends the user's biometric data and workout information to the Database. The FitSage App also sends information, such as User Interface menus and workout information, to the Phone Screen.

Sink: Database

Description: The Database stores the user's biometric information (name, age, height, weight, sex) as well as information about various workouts and data collected as the user engages with the app. The Database can only receive information from the FitSage App. In addition to storing information, the Database also gives out information to the Fitsage App. The App can query data from the Database to display or perform calculations with.

Component Diagram:





Component Diagram Descriptions:

Component: BMI calculator

Description: Accesses the User Table in the Database for the user's Age, Height, and Weight. Calculates the Body Mass Index (BMI) of the user.

Component: BFP Calculator

Description: Accesses the User Table in the Database for the user's Age, Height, and Weight. Calculates the Body Fat Percentage (BFP) of the user.

Component: Workout Calorie Calculator

Description: Queries the Calendar, User, and Workouts Tables of the database to get information about each workout the user performed, such as the repetitions, amount of time the workout was performed for, workout name, MET value, and user's weight. It then calculates the total calories that were burned from working out and subtracts the calculated value from the daily total calories before updating the Calendar table with the new daily total.

Component: Workouts Table

Description: Stores information about various workouts including the muscle group activated, workout name, and MET value of the workout. This component is accessed by the Workout Menu to display information about relevant workouts to the user.

Component: User Table

Description: Stores information about the user including the user's name, height, weight, age, goal, and goal calorie range. This component communicates with the Profile Menu to display and update the user's information.

Component: Calendar Table

Description: Stores information about each day the user uses the app such as the date, total daily calories, workouts performed, number of times each workout was performed, how long each workout was performed for, and streak. It communicates with the Nutrition and Workout menus to update and access the daily total calories. It also communicates with the Workout Calorie Calculator to determine how many calories were burned from working out. It is also accessed by the Profile Menu to display the user's current streak and their past streak history. Finally, it updates the age field of the User Table each year as the user ages.

Component: Database Interface

Description: Queries and updates the Database with data as the user uses the app. This component acts as the bridge between the Database and the rest of the User Interface.

Component: Profile Menu

Description: Allows the user to view and change information about their profile, including everything in the User Table of the Database and their current daily calorie total with streak and goal achievement information. Goal achievement is earned when the user's daily calorie total is within the calorie range generated by the app for their selected goal.

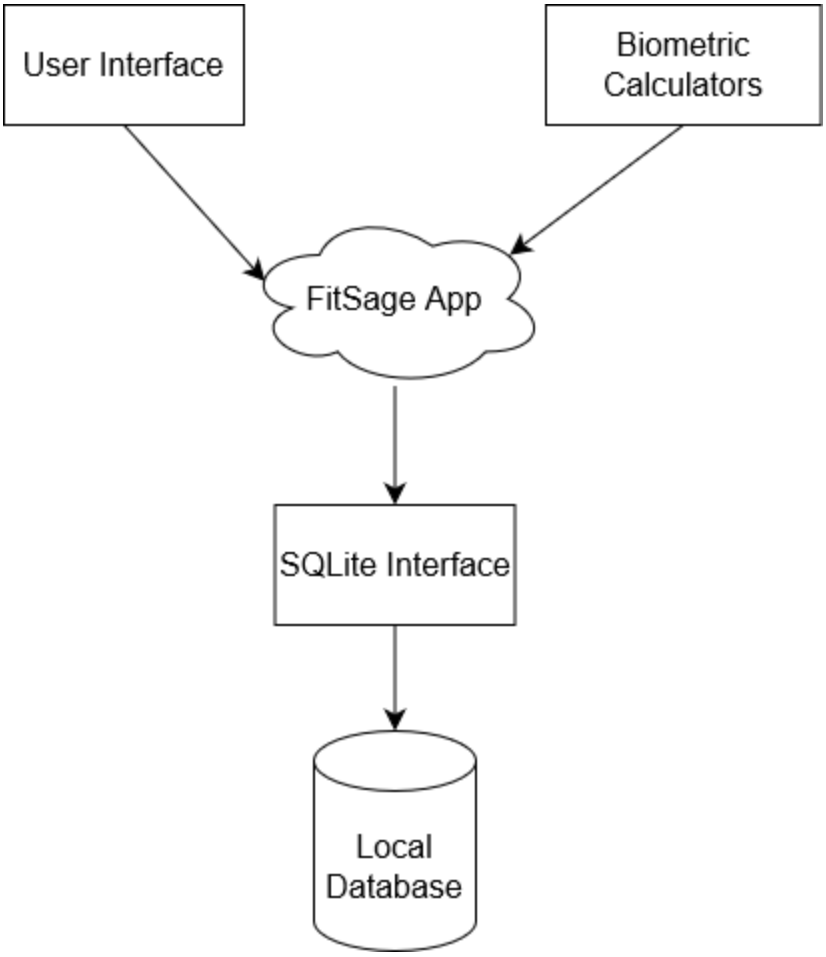
Component: Nutrition Menu

Description: Scans Nutrition Facts labels from food the user consumes and updates the Calendar Table of the Database with the calories from the food.

Component: Workout Menu

Description: Allows the user to select the muscle groups they want to exercise and queries the Workouts Table of the Database for workouts matching the chosen muscle group. The user can sort the workouts by difficulty or intensity (MET value). Once a workout is chosen, a set of instructions will appear and the user will start exercising for a specified amount of repetitions or time. The workouts performed, repetitions, and how long each workout was performed is stored in the Calendar database. Then, the Workout Calorie Calculator is used to calculate the calories burned from working out and to update the daily total.

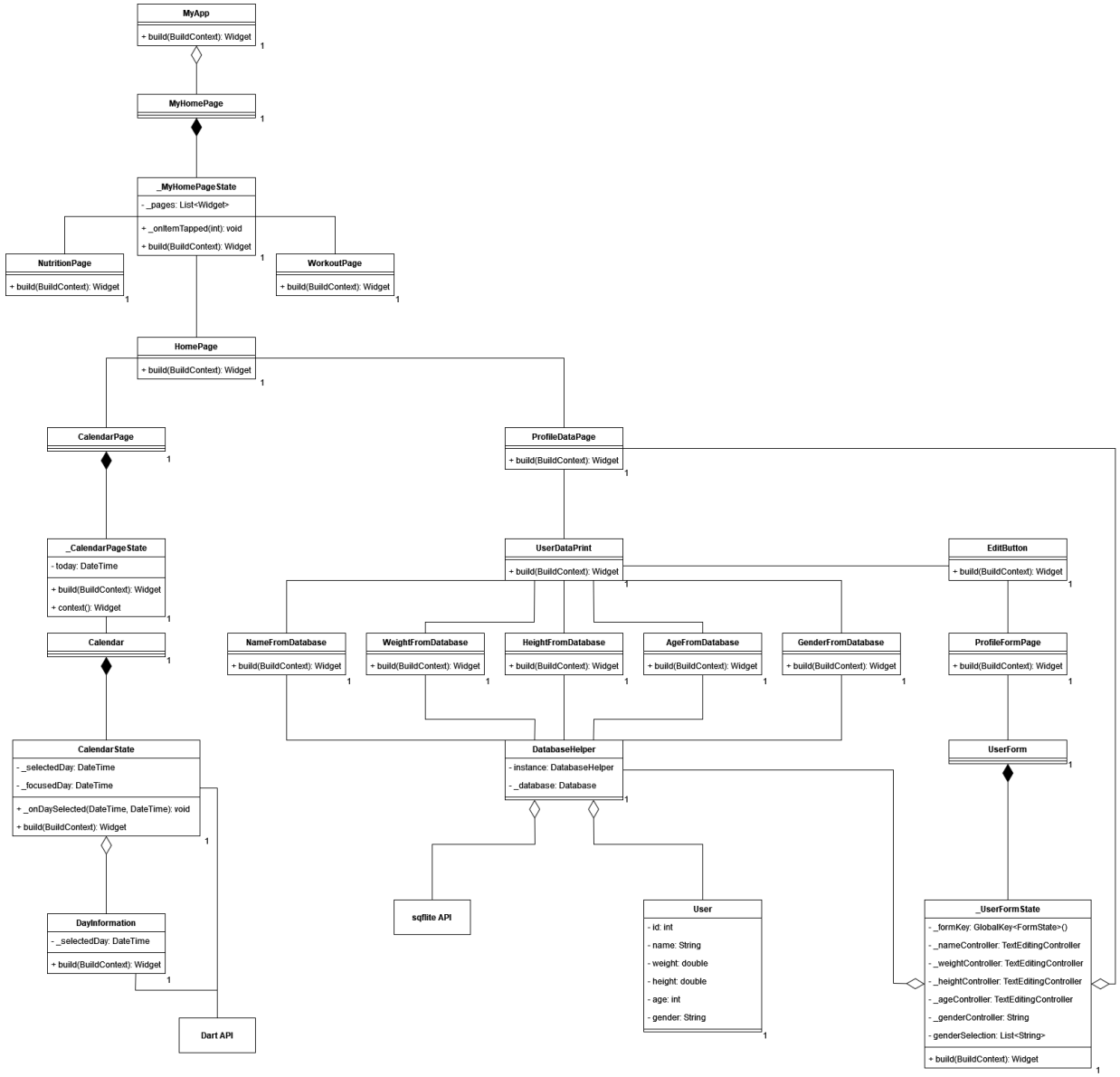
Architecture Diagram:



Architecture Diagram Description:

We chose a client-server architecture because it best fits the design of our project. The User Interface and Biometric Calculators act as clients to the Database, which stores information about the user. We use a SQLite interface to manage the database, which is stored locally on the user's Android device. While there is no external network, the relationships between the Database, Biometric Calculators, and User Interface act as a local network of data exchange. Because each component is essential to the operation of the Application, the local network is represented as the FitSage App.

Class Diagram:



Class Diagram Description:

This diagram shows all of the classes in the current iteration of FitSage and their relationships. In Flutter each visual element is classified as a widget, which is a type of class that can display things to the screen and interact with other widgets. Some widgets, like Calendar, UserForm, and MyHomePage, are stateful widgets and require a state widget to function. These relationships are represented as Aggregation because the state widgets could not function without a stateful widget calling them. Composition relationships are shown where one widget uses another widget as part of its functionality. For example, the DatabaseHelper needs the User class to store information about the user, but the User class can function without the DatabaseHelper. The methods and attributes of each class are also shown.