Sweat Rivals

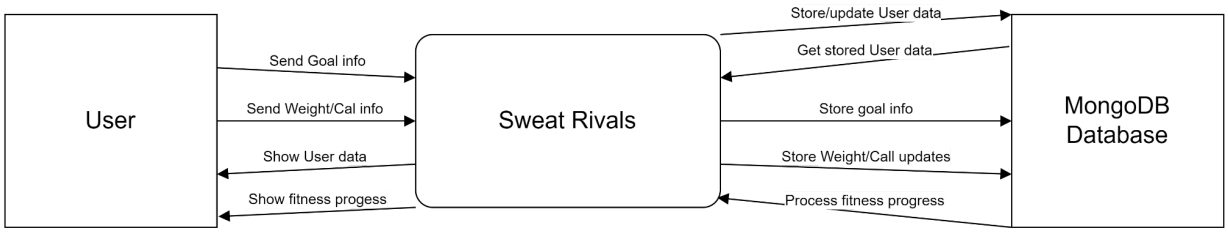
Group 11

Caden Lalley, George Viveros-Zavaleta, Harrison Whetzel, Michael Daramola, Alexander Griffin

Project Overview

Most people when working out will use some kind of fitness app that will track calories consumed as well calories burned. Fitness apps are a great way to keep track of your fitness progress as well as compete with friends through those numbers. While most fitness apps have friends and such, none are focused on that competitive side. Our application will fill that void by focusing on competition between friends using point based challenges. These challenges can range from weight and calorie loss to actual exercises. By completing these challenges and earning points you can compete with your friends as well as groups that can be made. You can also create goals that you want to complete, though they would not give you points at this time. We want to design an application that will help the average person get more excited about exercising as well as helping better their health.

Context Diagram:



User Stories:

- As a user who wants to track fitness data, I need to be able to create an account and login, so that I can access and keep track of my personal fitness progress and data.
- As a user who switches devices constantly, I need to be able to logout, So that i can manage multiple accounts, or be able to simply login temporarily onto a new device
- As a user who would use this app frequently, I need to be able to access the app without having to login every single time , So that i can just boot the app and get to work
- As a user, I need to be able to delete my account, So that i can restart my account, or so i can stop sharing my information with the app

- As a user who works out with friends, I need to be able to add and remove friends from my network, so that I can compete with and keep accountable the specific group of friends I choose to work out with.
- As a user, I need to be able to create and delete goals, So that I can motivate myself to burn calories
- As a user who works out with friends, I need to be able to add and remove friends from my network, so that I can compete with and keep accountable the specific group of friends I choose to work out with.
- As a user, I need to be able to create and join groups with my friends, so that we can participate in weekly challenges together, keep each other accountable, and compete to reach our shared fitness goals.
- As a user, I need to be able to track my calorie intake and calories lost, so that I can ensure I am meeting my calorie goals
- As a user, I need to be able to track my weight, So that I can ensure I am gaining/losing weight at a consistent rate
- As a user, I need to be able to get points from challenges, So that my friends and I can remain accountable and compete, making fitness more fun.
- As a user, I need to be able to view how many points my friends and I have, So that I can be competitive with my friends to motivate myself to burn calories

Product Backlog:

- As a user, i often forget what password I use for my accounts, so i would like to be able to remember it by confirming the password i used for my sweat rivals account

Architectural Overview

We are developing a web application that follows the MVC architecture. We found that this architecture suited the needs of our application and the natural design pattern of Express.js. Within the MVC design pattern, we chose to add a middleware module that allows for quick

development and implementation of middlewares as we see fit. For our view we used javascript and ejs.

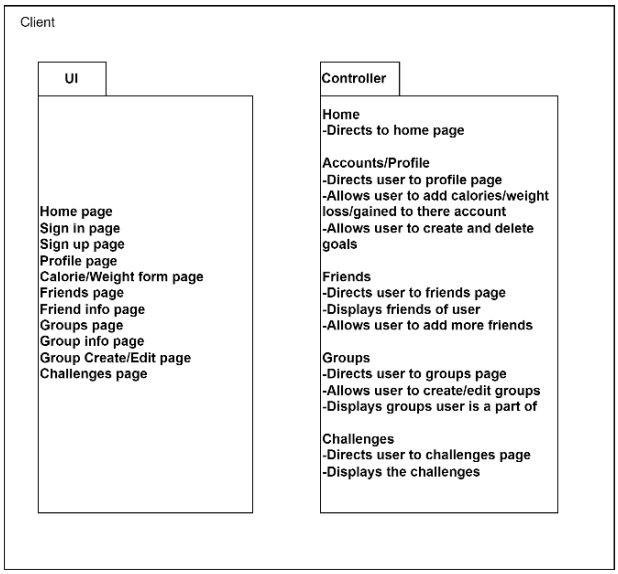
Another process we were going to use was Model-View-Presenter (MVP). It's a derivation of MVC and used mostly for building user interfaces. We decided not to use MVP because we all knew and have used MVC before, as well as it being more balanced and easier for us to fully understand.

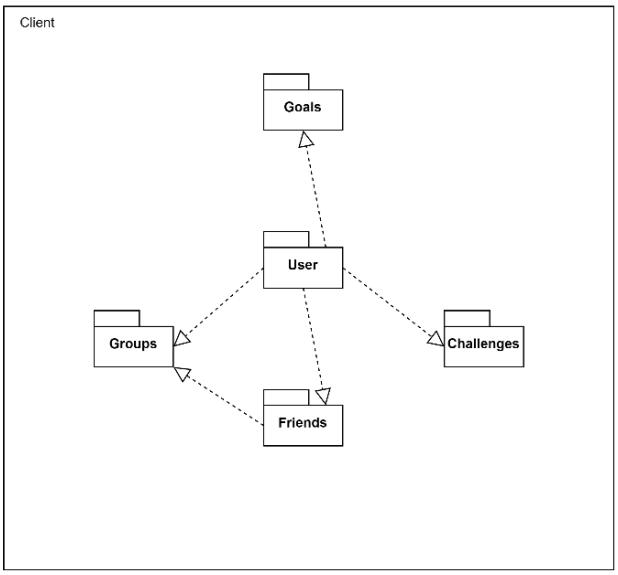
Subsystem Architecture

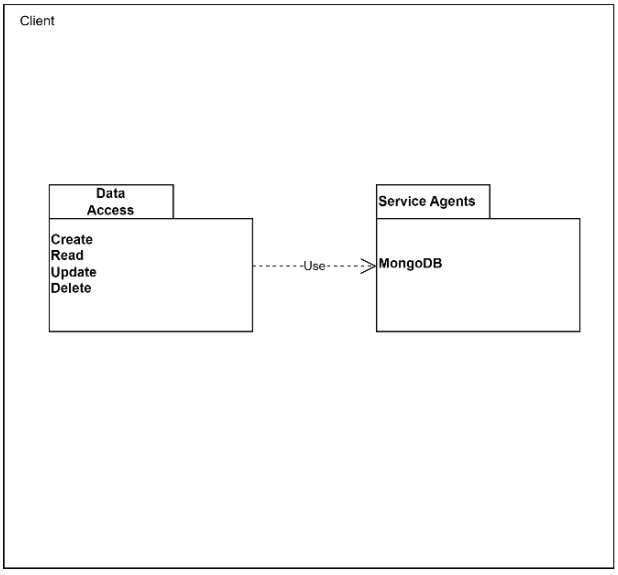
It has two layers, view and server

- Client
    - Our client is mainly composed of multiple web pages, each having a different purpose
- Server
    - The server consists of four modules:
        - Model
        - Controller
        - Routes
        - Middlewares

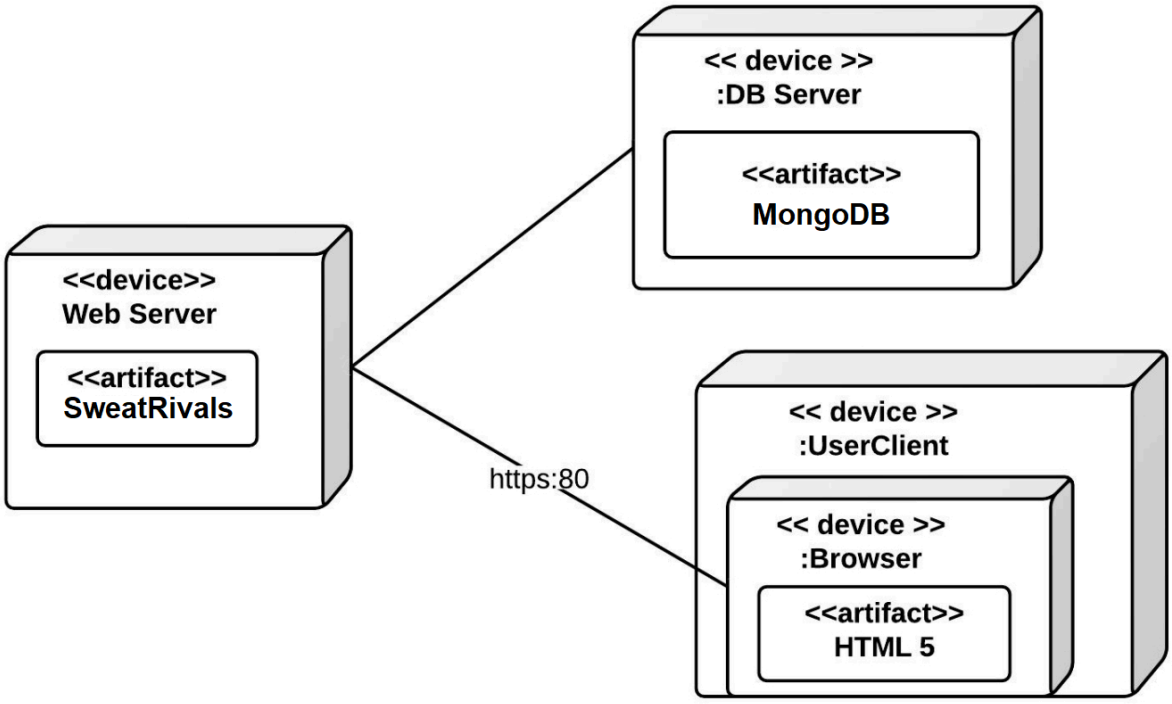
These modules are used to dynamically send views to the client with data that is stored in our persistent DB. The model, controller and middleware modules all interact with the requests made by the client to dynamically display different pieces of data.


Client
UI
Home page
Sign in page
Sign up page
Profile page
Calorie/Weight form page
Friends page
Friend info page
Groups page
Group info page
Group Create/Edit page
Challenges page
Controller
Home
-Directs to home page
Accounts/Profile
-Directs user to profile page
-Allows user to add calories/weight loss/gained to there account
-Allows user to create and delete goals
Friends
-Directs user to friends page
-Displays friends of user
-Allows user to add more friends
Groups
-Directs user to groups page
-Allows user to create/edit groups
-Displays groups user is a part of
Challenges
-Directs user to challenges page
-Displays the challenges


Client
Goals
User
Groups
Challenges
Friends


Client
Data Access
Create
Read
Update
Delete
Use
Service Agents
MongoDB

Deployment Architecture



The client uses the HTTPS on their browser to connect to the SweatRivals server over the internet. Express is used to host the SweatRivals server as well as the models, views, and controllers. All of this is then connected to the database server using MongoDB.

Global Control Flow

SweatRivals is event-driven due to the involvement of user actions, such as clicking or inputting data. The execution of SweatRivals functions are due to these interactions. There are a few time dependent factors in SweatRivals. These features include: weekly challenges being time gated (a week) and data being displayed over the course of the past week or month. SweatRivals is concurrent, which means that users are able to access it at the same time as each other. Multiple users are also able to access data at the same time as each other.

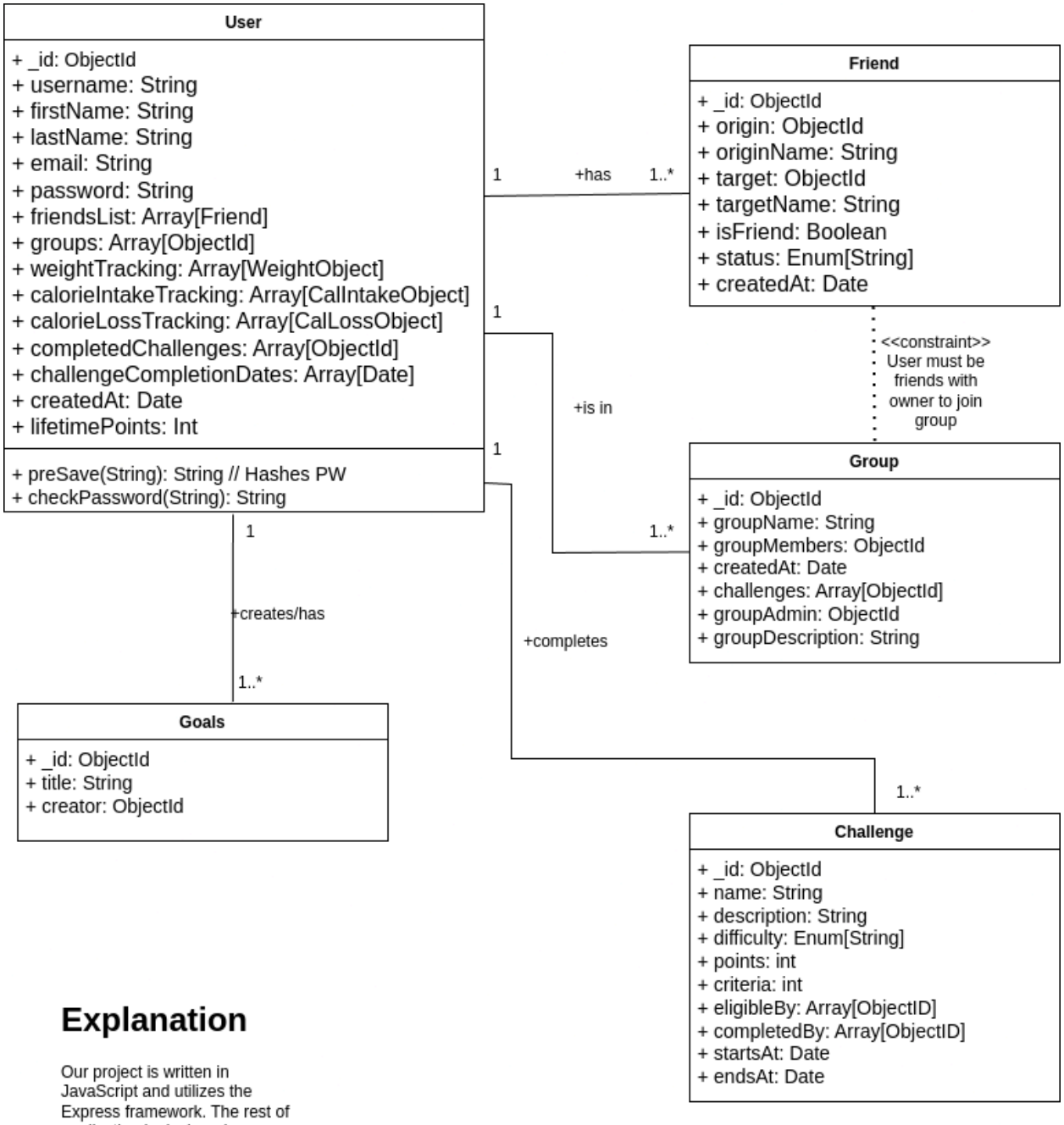
Persistent Data Storage

We will be storing all our information, user accounts, challenges, goals, friends, and groups in a mongoDB database. All of that information will usually be added to the database through the use of forms except for challenges which we will manually make in the database. The table for the user accounts has username, first name, last name, email, password, private, friendsList which is an array of friend schema, groups which is an array of object id, weightTracking which is an array of weightInforSchema, calorieIntakeTracking which is an array of calorieIntakeInfoSchema, calorieLossTracking which is an array of calorieLossTracking, completedChallenges which is an array of object id, createdAt, and lifetimePoints. The table that's for challenges has the name of the challenge, its description, the difficulty, amount of points it's worth, criteria which is what you need to complete it, eligibleBy which is an array of object id, completedBy which is an array of object id, startsAt, and expiresAt. The table for goals has the goal title as well as a goal creator which holds an objectId. The table that's for friends has origin that holds an object id, originName, target that holds an object id, targetName, isFriend, status, and createdAt. The table that is for groups has groupName, groupMembers that's an array of object id, created_at, challenges that's an array of objectId, groupAdmin that holds an objectId, and groupDescription.

Detailed System Design

For our model we used the Node.js and express framework, as well as ejs, javascript, and CSS. We used mongoDB as our database and used mongoose to populate it. Our web page views were created with ejs, and were formatted with a mix of bootstrap and css. Any kind of information needed from the database is displayed by getting that information in the controller, sending it through the route and using javascript in the ejs.

Static View

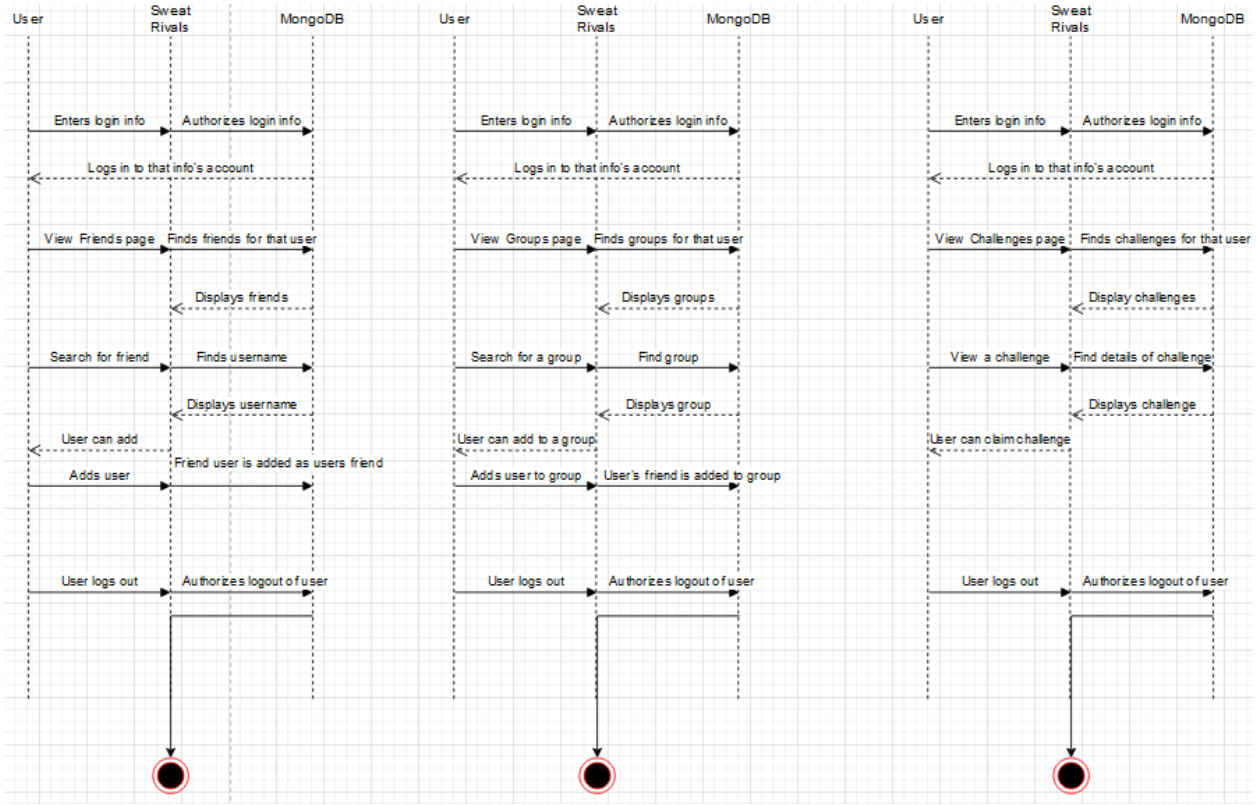


Explanation

Our project is written in JavaScript and utilizes the Express framework. The rest of application is designed functionally, so there are no other classes or objects. The rest of our functions simply parse the data within the objects displayed here and render that to the frontend.

Our User table holds the users information: username, first name, last name, email, password, if they want to be private or not, an array for friendsList, and array for groups, and array for weightTracking, two arrays for calorieIntakeTracking and loss tracking, an array for completedChallenges, createdAt, and lifetimePoints which is the total amount of points the user has earned. The user table is connected to all other tables. The table that's for challenges has the name of the challenge, its description, the difficulty, amount of points it's worth, criteria which is what you need to complete it, eligibleBy which is an array of object id, completedBy which is an array of object id, startsAt, and expiresAt. Challenges are connected to users as the user can complete them and the completed challenges are saved in the completedChallenges array. The table for goals has the goal title as well as a goal creator which holds an objectId. Goals are connected to the user table with the user's id saved in creator. The table that's for friends has origin that holds an object id, originName, target that holds an object id, targetName, isFriend, status, and createdAt. Friends is connected to user as the user's id is stored in origin and the friends id is stored in target. The table that is for groups has groupName, groupMembers that's an array of object id, created_at, challenges that's an array of objectId, groupAdmin that holds an objectId, and groupDescription. Group is connected to the user as the creator of the group is stored in groupAdmin. Friends and groups are connected to each other since the creator of the groups can only add their friends to it.

Dynamic View



Signing up gives you the ability to login. Logging in allows you to view your account page. Additionally, you are able to view friends, challenges, groups, or calories weight/info. Navigating to the friends tab allows you to view the profiles of yourself and your friends, add friends or remove friends. The groups tab allows you to create groups and view groups. In the view group screen, you're able to edit groups that you have the ability to. Under challenges, the only manual option available to the user is to view challenges and their progress on said challenges.