Major Features

- Daily Exercise Tracker
    - Users will be able to enter the amount of time they have exercised each day. They will be able to look back at previous days to spot trends. These trends are tracked by the app in order to advise the users' what their next steps should be to progress towards their goal
- Eating/Drinking Habit Tracker
    - Users will be able to enter what they have had to eat and drink in the day. They will be able to look back at previous days to spot trends.
- Point Rewards
    - When a user achieves a goal, they will receive a set amount of points. These points will allow the user to level up.
- Interaction Between Users
    - Users will be able to comment on activities, like activities, and message each other
- Preset Exercises
    - There will be different workout plans for different days. Each plan is a part of one guide geared to their goal that will help guide users to their specific goals. As the guide progresses, the intensity increases depending if the user can handle the increased intensity.
- Goal Setting
    - This will be an input by the user, so that the app can help lead the user into achieving the goal.

Requirements

(six features for functional and non total)

These must be: Specific, Measurable, Achievable, Relevant, Time Blocked

<u>Functional ("what")</u>

- Use Cases -
    - Add Use cases
        - Add User
        - Add Exercise
        - Add Eating/Drinking
    - Add - This can be tested by seeing if POST requests work through Insomnia/Postman API tests. This can be achievable, possibly within a sprint. This is relevant for the user to experience the functionality of the app.
    - Store Use cases
        - Keep track of reward points
        - Store goals
    - Store - This will be tested by checking to see if it is stored in the database with MongoDB Compass. This should be achievable in one-two sprints. This is relevant for users to access the data at a later time tied to their account because

the use of this would span multiple sign-ins. The time allotted should be about a week to maybe two weeks.

  - View Use cases
    - View exercise for the day
    - View current goal/points
      1. The app allows the user to view their personal goals on the homepage. This is measured as once the user logs in, the app goes straight to the homepage. This is achieved by testing to see if the log in goes straight to homepage afterwards. This is relevant because the app tracks fitness goals for the users. The time allowed to implement should be 1 to 2 weeks.
- What the system does for our organization
  - Store Use cases
    - Able to collect and store data about our users
  - Store - This will be tested by checking to see if it is stored in the database with MongoDB Compass. This should be achieved at the same time as the User store since they are the same thing. This is relevant for us to access the data at a later time tied to their account because the use of this would span multiple sign-ins. The time allotted should be about a week to maybe two weeks.

Non-Functional (“how”)

- User experience (SMART)
  - Specific: As more users utilize the app, if it doesn’t crash and can run smoothly with high foot traffic, use
  - Measurable: As high foot traffic occurs, we have testers to load the page and see if it can run smoothly, I.E no crashes or lags.
  - Achievable: This can be achieved because we can test for crashes
  - Relevant: The app is relevant because it accomplishes the goal for the project, in which we have or will have created an easy user-interface for people to track fitness goals.
  - Time Blocked: Finished within a sprint
- specifically how display information
  - (SMART)
    - We will have our program display fitness information in a manner such that it is always towards the top of the screen and in point of focus
    - Every time we load a new screen where fitness information is needed, we can check to make sure the information is in the correct position on the page.
    - Yes this a achievable since it is very easy to test.
    - Yes this is relevant because the goal for our project is to have an interface that makes tracking fitness goals and process easy.

        - Time budgeting for this implementation should not take any longer than an hour or two.
- we will have links that take you no more than 2 different pages, reduces clutter.
        - by reducing the amount of paths that links can take you too it will force us to have an interface that is simplistic and usable.
        - this is measurable as we can measure the length of sub links within different pages.
        - Yes this is achievable since we are able to measure the amount of links that each page can take you too.
        - Yes this is relevant for our goal because by keeping our page concise and straight forwards in this manner it will improve our users experience
        - Time budgeting for this might involve a working session or two to go through each section of the program.
- Follow one of the formats/templates provided in class.
- Your application will likely have many more features, and you should create requirements documents for all the features.
- But for this milestone, you need only turn in SIX requirements documents.

Project (ordered by feature creation sequence)

- Planning period (1 wk, 10/10 -10/16)
    - Decide who is working on what and then add that to this Project Plan
    - Setup Kanban board
    - Learn exact stack
    - Decide variable names and data being stored, etc.
    - Design front-end on paper, gather examples, etc.
    - Figure out version control

- User creation (2 wks, 10/16-10/30)
    - username, email, password, age, weight, height, initial points, goals
    - Create a front-end form to sign up
    - Take in information to the database
    - A way to access the information
    - Make sure front-end and back-end are working well together

- Daily Exercise Tracker, Eating/Drinking Habit Tracker, Point Rewards, Goal Setting (2 wks, 10/30-11/13)
    - Edit front-end (create navigation)
    - Determine point calculation
    - Preset goals, allow user to choose them
    - Make sure front-end and back-end are working well together

- Preset Exercises (1 wk, 11/13-11/20)
  - Edit front-end (create navigation to, pictures/videos of exercises, etc.)
  - Research exercises to add
  - Create a plan so that it mixes up weekly exercises in a proper way, not random
  - Make sure front-end and back-end are working well together

- Testing Stage (1 wk, 11/20-11/27)
  - Make sure that all functions are working properly
    - Ie buttons navigate to where they're supposed to, goals are updating, points rewarding, etc
  - Test the website with a lot of users
  - General functionality (have others use the website)

- Interaction Between Users, (Bonus, 1 wk, 11/27-12/2)
  - Socket.io (research this)
  - Front-end pop up message/chat box, or separate page

We are going to alternate who is working on front-end and back-end about every two weeks. This way, everyone will have a chance to learn both front and back end.