System Requirements

Section I. Functional Requirements

<table>
<tr><td colspan="3">Feature Name:
User Login</td><td colspan="3">Priority:
High</td></tr>
<tr><td colspan="6">Description:
The system will allow the user to login or sign up for an account, by clicking the top right account icon. Once clicked it will prompt a screen to allow users to enter their email/password or sign up with email/password. If a user does not create a login, they will still be able to navigate our site and access all features but not be able to store data until an account is created. Users will also be able to reset their password by clicking the “reset or forgot password” button.</td></tr>
<tr><td colspan="2">Inputs:
Username and password</td><td colspan="2">Outputs:
Added record in login database.
Users have access to the rest of the website.</td><td colspan="2">Processes:
When a user inputs data fields, his/her info will be logged into our database. Salt will be added to the user’s passwords for security reasons.</td></tr>
<tr><td colspan="6">Dependencies:
User authentication via Firebase, and database access.</td></tr>
</table>

<table>
<tr><td>Feature Name:
User Profile</td><td>Priority
High</td></tr>
<tr><td colspan="2">Description:
The system shall allow users to create and manage their profiles, including storing personal data such as name, age, weight, workout preferences, and fitness goals. This will be done by clicking the account icon on the top left (this is a different page from the user login). A menu will display asking the user to enter info about themselves and a drop-down menu will be available to allow the user to choose a color/background. A color wheel will be available to choose colors.</td></tr>
</table>

<table>
<tr><td>Inputs:

User_name, User_age, User_weight, Workout preferred, fitness goals.</td><td>Outputs:

A customized user profile based on the inputs that the user chose.</td><td>Processes:

Users enter their name, age, weight, workout preference, and fitness goals.
A new entry is created to the database according to the user query.
The database updates its log and records the new entry.</td></tr>
<tr><td colspan="3">Dependencies:

Users must create a login to access the profile.
User profile is created and stored in the database.</td></tr>
</table>

<table>
<tr><td colspan="3">Feature Name:

Progress Tracking</td><td colspan="3">Priority:

High</td></tr>
<tr><td colspan="6">Description:

Users will manually enter their workout information such as sets and reps into the current day's workout. They will be able to access the current day via an interactive calendar on the right side of the screen or by clicking on the workout tab on the left tab menu. This will be a separate page from the other features</td></tr>
<tr><td colspan="2">Inputs:

Sets, Reps, Duration, Type of Exercise</td><td colspan="2">Outputs:

Progress report/graph displaying progress (GITHUB GRAPH) Calendar showing progress/consistency</td><td colspan="2">Processes:

The user selects the workout day.
The user enters sets, reps, duration, and type of exercise in corresponding cells.
The system creates a new log to record the progress.
The new log is updated in the database.</td></tr>
<tr><td colspan="6">Dependencies:

User profile and user login. Workout history is stored in a database, a data visualization tool for graphing progress.</td></tr>
</table>

<table>
<tr><td colspan="2">Feature Name:
App Navigation
(Workout Plans by Muscle Group / Videos)</td><td>Priority:
High</td></tr>
<tr><td colspan="3">Description:
The system shall provide predefined workout plans categorized by muscle groups such as Arms, Legs, Push Day, Pull Day, Chest, Back, Quads, and Glutes. Users will manually enter their preferred category in the search bar and upon clicking search the plans will be displayed in table format, with the workout itself and videos along with it.</td></tr>
<tr><td>Inputs:
Desired muscle group string entry</td><td>Outputs:
A table list of exercise names and videos showing how to perform such exercises.</td><td>Processes:
A user will click on the search bar to manually type in a muscle group. Clicking submit will guide them to a new page showing the table list.</td></tr>
<tr><td colspan="3">Dependencies:
Database of workouts</td></tr>
</table>

<table>
<tr><td colspan="2">Feature Name:
Calorie Tracker</td><td>Priority:
Medium</td></tr>
<tr><td colspan="3">Description
The system shall allow users to manually enter their daily calorie intake and macro information.
The logs of the calorie tracker will only be viewed from the page of this feature
This will be a separate page from the other features
The system will display visually a graph to show calorie intake over the calendar week/month
The system shall calculate calorie requirements based on user input such as weight, and fitness goals, and suggest daily target intake. (Low Priority)</td></tr>
<tr><td>Inputs:
User’s calorie intake (manual entry)</td><td>Outputs:
Daily calorie report, suggested calorie intake for a target goal</td><td>Processes:
User profile data, and calorie calculation algorithm</td></tr>
<tr><td colspan="3">Dependencies:
User Profile record
Calorie log record</td></tr>
</table>

<table>
<tr><td colspan="3">Feature Name:
Recommended Workouts</td><td colspan="3">Priority:
Low</td></tr>
<tr><td colspan="6">Description:
The system via an AI-based model will recommend the user a set of recommended workouts based on what the user has specified in their query.
This will be a separate page from the other features
The system will have a set list of recommended workouts based on their fitness goals that were inputted on the user profile page (Lower priority)</td></tr>
<tr><td colspan="2">Inputs:
User Query(Question/sentence)
User profile fitness goals</td><td colspan="2">Outputs:
List of suggested workouts</td><td colspan="2">Processes:
User enters their workout goals manually into a text box. Query gets submitted to AI-api and the result gets returned</td></tr>
<tr><td colspan="6">Dependencies:
AI-API(Llama/Openai)
Potentially User login / User profile fitness goals</td></tr>
</table>

Section II. Non Functional Requirements

User login:

Users will have the ability to sign up/login by clicking the top right User Icon. Interacting with sites(such as trying to save data) before signing up/in will redirect them to the same Login/Sign up page. Users will have to provide a valid email address along with an 8+ character password which must include at least: 1 capital letter and 2 special characters. Users will be notified with error messages indicating that they have an incorrect password or their email address was not found. If users forget/want to reset their password, they will be able to do so through a self-service option via their recovery email. Three simple steps should be receiving the resetting link via the recovery email, resetting the new password, and submitting it. Said passwords should be uploaded to our database with “salt” included by Firebase to enhance users' privacy.
Upon providing the login/sign-up info and clicking submit, Users should wait at most .5 seconds before the landing page displays. The site should be able to handle at least 100 concurrent users at a time. A timeout mechanism of login will be applied if inactivities last for 5 minutes.

User Profile:

Upon changing any data and clicking the save button, the site should update these attributes in real time, making it stored and retrieved without any data loss. Users should be able to see these changes without having to reload their current page. Users will have to manually input their data for each section and manually save them. Once saved is clicked the information uploaded should be stored in the database. Users will be limited to making at most 10 total appearance changes per day to their user profile (background color, user picture, etc) and at most 5 individual personal data changes per day(weight, height, bmi,etc). The color wheel being used will include primary, secondary, and tertiary colors. All of this information should not be accessible by anyone other than the user themselves.
If desired by the user, the system should allow users to delete their profile and erase all data the user had stored

Progress Tracking:

The system should use the same database update system as the calorie tracker. The system should be able to easily export the progress tracking graph to the user profile feature. The system should sanitize and validate user input to prevent NoSql injection attacks. Any inputs should be autosaved every 30 seconds to prevent data loss.The system should provide real time feedback notifying the user that the data is saved, by a small green icon saying “saved” next to the table being inserted in. The system should be able to take in decimal numbers as well as whole numbers. There should also be an option to switch between pounds and kilograms.

App Navigation:

The desired workout plans should be able to be returned within 100ms. The system should sanitize and validate user input to prevent NoSql injection attacks. The system should respond to user navigation requests within 1 second. Users should be able to navigate to any location of the site at every page. For example being able to access user login from the tracking page, etc). The users should be only able to view their own session, not other users' data.

Calorie Tracker:

The system shall not exceed more than 10 changes to each days calorie tracking per day. The system should use the same system as the progress tracker to update the database to reuse assets. The system should be able to easily export the progress tracking graph to the user profile feature. The system should sanitize and validate user input to prevent NoSql injection attacks

Recommended Workouts:

The system should have at least a 99% uptime. The system should be separated from other features into its own microservice. The system should not reveal the API token when making an API call to ensure the security of the key. The system should return a response from the API within 1000ms to ensure. The system for this feature shall not exceed 50 API calls per hour to limit excessive usage of API tokens.