Technical Specification for Workout Tracking Platform

User Stories/User Requirements:

1.1 User Account Creation: The user will be able to create an account by providing their email address and password. The user will also add personal information in order to assist the api in creating a recommended workout plan.

1.2 Fitness Goals: Users will be able to set fitness goals, such as losing weight, gaining muscle mass, or increasing endurance.

1.3 Logging Workouts: Users will be able to log their workouts, including the exercises performed, sets, reps, weights, and duration.

1.4 Recommendations: The platform will provide users with recommendations for workouts based on their fitness goals, previous activity and their user information.

Stretch goal:

1.5 Social Features: Users will be able to connect with friends and join groups to share their progress and keep each other motivated.

Endpoints:

2.1 User Account Creation:

POST /users

This endpoint takes the user data type and adds a new user to the database.

The user datatype is in the following structure:

{

User_id: generated user_id,
starting_Lbs: weight in pounds,
Name: name of fighter,
Height_inches: height of the user,
Avg_calorie_intake: average calorie intake of a user,
Age: age of the user,

Gender: gender of the user
}

GET /users
This endpoint gets the user based on its id. The output would be formatted as follows:

{
User_id: generated user_id,
starting_Lbs: weight in pounds,
Name: name of fighter,
Height_inches: height of the user,
Avg_calorie_intake: average calorie intake of a user,
Age: age of the user,
Gender: gender of the user
}

2.2 Fitness Goals:

GET /goals
This endpoint returns the goals related to a specific user id. The output would be formatted as follows:

{
User_id: user_id,
User_name: name of the user
Goals: list of the user's goals

{
Goal_id: id of the goal,
Type: goal type,
Target_weight: target weight associated to the goal

}

}

POST /goals
This endpoint would post the goal. The post would be in the following structure:

{
    user_id: the user id of the goal,
    Type_id: the type of workout (linked to a goal_type table)
    goal_id: the id of the goal,
    Target_weight: the target weight for the goal
    Workout_id: id of the workout that is produced from the goal
    User_id: user_id

}

2.3 Logging Workouts:

GET /log/user
This endpoint returns the logs associated with a user id. The output would be structured as follows:
{
    User_id: user_id,
    Name: name of the user,
    Logs: list of logs

        {
            Log_id: id of the log,
            Current_lbs: weight associated with the log,
            Time_posted: time the log was posted
        }

POST /log
This endpoint allows a user to posted a log
{
    User_id: the id of the user who's log this is being added to,
    Log_id: the log that the workout is being added to,

Current_lbs: the weight of the user for the log,
Time_posted: datetime for the log
}

GET /workouts/user
This endpoint returns the workouts associated with a user id. The output would be structured as follows:
{
User_id: user_id
Workouts: a list of workouts
{
Workout_id: workout_ id
workout_name: name of the workout,
Weight: the weight for the workout, null if not applicable
Distance_ft: the distance for the workout
Repetitions: number of repetiontions for the workout
Seconds: duration of the workout in seconds, null if not applicable
Sets: number of sets of the workout
Times_per_week: the number of times per week the workout is performed
}
}

2.5 Social Features: Social features will be expanded upon in v2.

Detailed Descriptions of Edge Cases and Transaction Flows:
3.1 User Account Creation: The endpoint should check if the provided email address is already in use. If it is, the endpoint should return an error message and prevent account creation.

3.2 Fitness Goals: When creating or updating a fitness goal, the endpoint should check that the user is authorized to perform the action. If the user is not authorized, the endpoint should return an error message and prevent the action. The endpoint should also check that the provided goal data is valid, including the goal type, target value, and target date.

3.3 Logging Workouts: When creating or updating a workout, the endpoint should check that the user is authorized to perform the action. If the user is not authorized, the endpoint should return an error message and prevent the action. The endpoint should also check that the provided workout data is valid, including the exercise type, sets, reps, weights, and duration.

3.4 Recommendations: The recommendations endpoint should use a recommendation algorithm based on the user's fitness goals and previous activity. If the user has not logged enough data, the endpoint should return a message indicating that more data is needed.

Stretch Goal:

3.5 Social Features: When creating or updating a friend or group, the endpoint should check that the user is authorized to perform the action. If the user is not authorized, the endpoint should return an error message and prevent the action. The endpoint should also check that the provided friend or group data is valid, including the friend or group name and user ids.

Additional Features:

4.1 Data Validation: All endpoints should perform data validation to ensure that the provided data is in the correct format and meets any constraints.

4.2 Error Handling: All endpoints should return appropriate error messages if an error occurs, including the HTTP status code and a description of the error.

4.3 Authentication: The API should use authentication to ensure that only authorized users can access or modify data.

4.4 Authorization: The API should use authorization to ensure that users can only access or modify their own data.

4.5 Security: The API should use HTTPS to ensure that all data is transmitted securely.

4.6 Scalability: The database should be designed for scalability to handle a large number of users and data over time.