Test Plan

Forrest Brown, Evan DiFilippo, and JP Burger

Part I. Description of Overall Test Plan

First and foremost, functional testing is required to validate that the application is functioning as intended. For our project, this includes validating that users can log workouts seamlessly, workout progress can easily be visualized, the Artificial Intelligence system recommends adjusting the weight and repetition counts based on user progress, and the camera-based repetition detection system accurately counts the user's repetitions.

Secondly, performance testing is essential to assess the application's responsiveness. For our project, this involves stress testing the application's repetition detection system and user interface.

Finally, usability testing is vital to ensure a positive user experience. For our project, this includes evaluating the intuitiveness of the user interface, the effectiveness of AI recommendations for enhancing workouts, and the user-friendliness of the repetition detection system.

Part II. Test Case Descriptions

1. **Rep has hit full depth state (this can be split into different tests for different movements)**
    1.1. rep_full_depth
    1.2. To verify that our mocap functions properly given that an exercise hits full depth.
    1.3. Given a movement like squat or bench press, only trigger when the bar is low enough to be considered a rep.
    1.4. INPUT: Mocap recording.
    1.5. OUTPUT: Either depth=true or depth=false

1.6. Can be any of the following: Normal: a regular movement; Abnormal: the wrong movement; Boundary: extreme depth or no movement.

1.7. Blackbox test.

1.8. Functional test.

1.9. Unit test.

2. Rep is complete state

2.1. rep_complete

2.2. To verify that mocap functions recognize when a rep is complete.

2.3. Given a movement like squat or bench press, only trigger when the bar is high enough to be considered a rep.

2.4. INPUT: Mocap recording.

2.5. OUTPUT: Either complete=true or depth=false

2.6. Can be any of the following: Normal: a regular movement; Abnormal: the wrong movement; Boundary: completely locked completion or no movement.

2.7. Blackbox.

2.8. Functional.

2.9. Unit.

3. New target rep count (+1)

3.1. plus_one_rep

3.2. To verify that the AI properly recommends an increase in reps.

3.3. Given historical dummy data, should output an increase in one rep for the next set.

3.4. INPUT: A csv or json or pandas dataframe.

3.5. OUTPUT: Previous rep count for a movement +1

3.6. Normal case.

3.7. Blackbox.

3.8. Functional.

3.9. Unit.

4. New target rep count (+2)

4.1. plus_two_rep

4.2. To verify that the AI properly recommends an increase in reps.

4.3. Given historical dummy data, should output an increase in two reps for the next set.

4.4. INPUT: A csv or json or pandas dataframe.

4.5. OUTPUT: Previous rep count for a movement +2

4.6. Normal case.

4.7. Blackbox.

4.8. Functional.

4.9. Unit.

5. New target weight (+5lbs)

5.1. plus_five_weight

5.2. To verify that the AI properly recommends an increase in weight.

5.3. Given historical dummy data, should output an increase of 5 pounds for the next set.

5.4. INPUT: A csv or json or pandas dataframe.

5.5. OUTPUT: Previous weight for a movement +5

5.6. Normal case.

5.7. Blackbox.

5.8. Functional.

5.9. Unit.

6. New target weight (+10lbs)

6.1. plus_ten_weight

6.2. To verify that the AI properly recommends an increase in weight.

6.3. Given historical dummy data, should output an increase in 10 pounds for the next set.

6.4. INPUT: A csv or json or pandas dataframe.

6.5. OUTPUT: Previous weight for a movement +10

6.6. Normal case.

6.7. Blackbox.

6.8. Functional.

6.9. Unit.

7. Graph (total volume moved over time)

7.1. volume_graph

7.2. To test graphing functionality

7.3. Given dummy data, should output the proper line graph.

7.4. INPUT: A csv or json or pandas dataframe.

7.5. OUTPUT: Line graph

7.6. Normal case.

7.7. Blackbox.

7.8. Functional.

7.9. Unit.

8. Graph (days overloaded / underloaded)

8.1. overload_graph

8.2. To test graphing functionality

8.3. Given dummy data, should output the graph.

8.4. INPUT: A csv or json or pandas dataframe.

8.5. OUTPUT: Graph

8.6. Normal case.

8.7. Blackbox.

8.8. Functional.

8.9. Unit.

9. Graph (specific exercise weight over time)

9.1. weight_graph

9.2. To test graphing functionality

9.3. Given dummy data, should output the proper line graph.

9.4. INPUT: A csv or json or pandas dataframe.

9.5. OUTPUT: Line graph

9.6. Normal case.

9.7. Blackbox.

9.8. Functional.

9.9. Unit.

10. Graph (specific exercise reps over time)

10.1. rep_graph

10.2. To test graphing functionality (this could be in tandem with weight)

10.3. Given dummy data, should output the proper line graph.

10.4. INPUT: A csv or json or pandas dataframe.

10.5. OUTPUT: Line graph

10.6. Normal case.

10.7. Blackbox.

10.8. Functional.

10.9. Unit.

Part III. Test Case Matrix

Test Case ID	Normal/Ab.	Black/Whitebox	Funct/Perform	Unit/Integration
rep_full_depth	Both	Blackbox	Functional	Unit
rep_complete	Both	Blackbox	Functional	Unit
plus_one_rep	Normal	Blackbox	Functional	Unit
plus_two_rep	Normal	Blackbox	Functional	Unit
plus_five_weight	Normal	Blackbox	Functional	Unit
plus_ten_weight	Normal	Blackbox	Functional	Unit
volume_graph	Normal	Blackbox	Functional	Unit
overload_graph	Normal	Blackbox	Functional	Unit
weight_graph	Normal	Blackbox	Functional	Unit
rep_graph	Normal	Blackbox	Functional	Unit