PocketSports

A Digital Coaching App

Test Plan Document
Version 1.0

Team Members:

Garrett Gmeiner - ggmeiner2021@my.fit.edu

Tyler Ton - tton2021@my.fit.edu

Parker Cummings - pcummings2021@my.fit.edu

Taylor Carlson - tcarlson2021@my.fit.edu

Faculty Advisor:

Fitzroy Nembhard - fnembhard@fit.edu

Client:

Brad MacArthur - bmacarthur@fit.edu

Florida Institute of Technology

9/30/2024

Table of Contents

1. Introduction

1.1. Document identifier

This document outlines the test plan for the PocketSports Digital Coaching App, version 1.0.

1.2. Scope

The scope of this test plan covers the functionality, performance, and integration testing for the PocketSports Digital Coaching App. The app supports multiple user roles including owners, coaches, players, and parents. The main functionalities include creating and managing drills, practice plans, team management, stat tracking, and goal setting. The test plan outlines the testing process for ensuring these features are implemented effectively and function as intended across multiple sports (Lacrosse, Basketball, Volleyball).

1.3. References

Mohsen Taleb. (2024, January 5). JavaScript unit testing frameworks in 2024: A comparison. Raygun Blog. https://raygun.com/blog/javascript-unit-testing-frameworks/

1.4. System overview and key features

PocketSports is a web application that caters to coaches, players, and parents. Its main features include:

- **Team Management**: Ability to create and manage teams, assign roles (coach, player), and add or remove members.
- **Drill/Play Creation**: Coaches can design and store drills/plays using a customizable field/court interface for different sports.
- **Practice Planning**: Coaches can create, edit, save, and execute practice plans that include drills from the drill bank.
- **Stat Tracking**: Real-time stat tracking for individual players and teams during practices or games.
- **Goal Setting and Tracking**: Coaches and players can set performance-based goals and monitor progress via graphical reports.
- **User Roles and Permissions**: Different roles (owner, coach, player, parent) have unique permissions for managing and viewing team data.

1.5. Test overview

1.5.1 Organization

The testing process will be conducted by the PocketSports team. The testing will be organized into phases:

1) Create Account test case
2) Login Testing
3) Forgot Password Testing
4) Create Team
5) Create account
6) Invites Test
7) Relinquish Test
8) Removal Test
9) Switching Test
10) Calendar Test
11) Create, Save, Edit drills Test
12) Tag Test
13) Practice Plan Test
14) Add Media Test
15) Play Drill Animated Test
16) Play Drill Field Views Test
17) Add Stats Test
18) View Drills Test
19) Execute a Plan Test
20) Create team goals Test
21) Create Goals Player Test
22) Destroy Goals Player Test
23) Create/Edit/Delete Team Goal Test
24) Create/Edit/Delete Individual Goal Test

1.5.2 Master test schedule

Testing will be conducted according to the following schedule:

1. Create Account test case Week 1
2. Login Testing Week 1
3. Forgot Password Testing Week 1
4. Create Team Week 2
5. Create account Week 2
6. Invites Test Week 2
7. Relinquish Test Week 3
8. Removal Test Week 3
9. Switching Test Week 3
10. Calendar Test Week 4
11. Create, Save, Edit drills Test Week 4
12. Tag Test Week 4
13. Practice Plan Test Week 5
14. Add Media Test Week 5
15. Play Drill Animated Test Week 5
16. Play Drill Field Views Test Week 6
17. Add Stats Test Week 6
18. View Drills Test Week 6
19. Execute a Plan Test Week 7
20. Create team goals Test Week 7
21. Create Goals Player Test Week 7
22. Destroy Goals Player Test Week 8
23. Create/Edit/Delete Team Goal Test Week 8
24. Create/Edit/Delete Individual Goal Test Week 8

1.5.3 Integrity level schema

PocketSports uses a three-tier integrity level schema to categorize features and functionalities:

Critical: Features essential for app functionality (login, account creation, stat tracking).
High: Features important for overall user experience but not critical to basic operation (team management, drill creation).
Medium: Optional features or enhancements (AI/ML recommendations, year planner).

1.5.4 Resources summary

Testing Environments: Multiple devices (iOS, Android), browsers (Chrome, Safari, Firefox), and operating systems (Windows, macOS).

1.5.5 Responsibilities

Garett Gmeiner: Oversees the testing process, coordinates tasks, and ensures adherence to the schedule.
Parker Cummings: Perform manual and automated tests, document results, and report bugs.
Taylor Carlson & Tyler Ton: Fix bugs and provide support during integration testing.

1.5.6 Tools, techniques, methods, and metrics

Tools:

JIRA (Issue Tracking).

Techniques:

Manual functional testing and security vulnerability scanning.

Methods:

Black-box testing for end-user functionality, white-box testing for code coverage.

Metrics:

Defect Density: Number of bugs per function/module.
Test Coverage: Percentage of code or features tested.
Mean Time to Detect (MTTD): Average time to discover a bug.
Mean Time to Repair (MTTR): Average time to fix a bug.

2. Details of the Master Test Plan

2.1 Create Account

2.1.1 Test case identifier

UT_01

2.1.2 Objective

Test that an account can be created using the “create account” system and ensure it is in the user database. This is critical for the system to function properly and it is the first step of using the software.

2.1.3 Inputs

Inputs needed to create an account include first name, last name, email address, and password. The password must be entered twice and contain the following at a minimum: One uppercase letter, one number, one special character, and length of 8 characters.

2.1.4 Outcome(s)

If an account is made using an email address that is already in the database, it should not process the creation of the user profile and should flag the error. Additionally, if the password does not match or meet the requirements, it should not process the creation of the user profile and should flag the error. Otherwise, the profile should be created and inserted into the database.

2.1.5 Intercase dependencies

There are none. This is the first step to unlock the functionalities of the software.

2.2 Login

2.2.1 Test case identifier

UT_02

2.2.2 Objective

Test logging into an already created account to ensure the login’s query functions of the database are correct.

2.2.3 Inputs

Inputs needed to log in are the user's email address and his or her password.

2.2.4 Outcome(s)

If the account enters invalid credentials, the user should not move on to the home page signed into an account, rather it should flag the user that his or her username or password are incorrect.

2.2.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to log in.

2.3 Forgot Password

2.3.1 Test case identifier

UT_03

2.3.2 Objective

Test a user resetting his or her password due to forgetting it or it being compromised.

2.3.3 Inputs

User's email address

2.3.4 Outcome(s)

If the user enters the 6-digit code sent to his or her email into a box inside the forgot password page, then the user will be prompted to enter his or her new password twice. If passwords match, the user will see a success screen and should go back and log in with the new password.

2.3.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to reset a password and log in.

2.4 Create a Team

2.4.1 Test case identifier

UT_04

2.4.2 Objective

Test the functionality of creating a team

2.4.3 Inputs

User email address, password, and organization

2.4.4 Outcome(s)

A team should be created and the user should be listed as the owner of the team.

2.4.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team.

2.5 Join a Team

2.5.1 Test case identifier

UT_05

2.5.2 Objective

Test the functionality of the join team functionality.

2.5.3 Inputs

In order to join a team, a user must have the team_ID and the organization_ID

2.5.4 Outcome(s)

If a team has the entered team_ID and organization_ID that are paired to each other, then the user will have a pending join request.

2.5.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team. This is also dependent on UT_04 since a team must be created and the owner of the team must accept the join request.

2.6 Invite a coach, player, and parent to a team

2.6.1 Test case identifier

UT_06

2.6.2 Objective

Test the functionality of an owner sending out invitations to join a team.

2.6.3 Inputs

The inputs for this function will be __ lists. A list for the team_ID, organization_ID, user role, and user email. The lists must all be the same length. The team_ID and organization_ID lists must contain a uniform value. The user role should be specified with a string naming the role as ‘coach’, ‘player’, or ‘parent’. The user emails should be valid email addresses.

2.6.4 Outcome(s)

Each user receiving an invite should get a link sent to his or her email address to join the team. If clicked, it will pose to the user a question of ‘Do you want to join Club Soccer at Florida Institute of Technology. The user can accept or reject the invitation. If the user rejects, it should thank the user for using PocketSports and redirect them to the landing page. If the user accepts and already has an account, he or she will be redirected to the login screen. On the other hand, if the user accepts and does not have an account, he or she will be redirected to the ‘create an account’ screen so he or she can make an account to join the team.

2.6.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team. This is also dependent on UT_04 since a team must be created for the owner of the team to invite users to join the team.

2.7 Relinquish owner to a coach

2.7.1 Test case identifier

UT_07

2.7.2 Objective

Test the functionality of relinquishing team ownership to a new user.

2.7.3 Inputs

The inputs needed to switch the owner are the owner’s user_ID, the coaches user_ID, the team_ID, and the organization_ID.

2.7.4 Outcome(s)

If the user is not a coach, it should not process and give error messages saying that the user must be a coach to be given the permissions. If the user is a coach

and the team and organization ID's match, the roles should be switched. This should be verified on the team roster.

2.7.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team. This is also dependent on UT_04 since a team must be created. Furthermore, this is also dependent on either UT_05 or UT_06 since a second coach must be able to join the team for this to be possible.

2.8 Remove a coach, player, and/or parent

2.8.1 Test case identifier

UT_08

2.8.2 Objective

Test the functionality of removing the different user types that can be on a team.

2.8.3 Inputs

The inputs needed to remove a coach, player, or parent are the user's user_ID, the team_ID, and the organization_ID.

2.8.4 Outcome(s)

If the user_ID can be found on the team with the team_ID and organization_ID, the user should be removed from the team. Additionally, this can only be done by the owner of the team.

2.8.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team. This is also dependent on UT_04 since a team must be created. Furthermore, this is also dependent on either UT_05 or UT_06 since a second user must be able to join the team for this to be possible.

2.9 Flip between multiple teams

2.9.1 Test case identifier

UT_09

2.9.2 Objective

Test the functionality of seamless transitioning between multiple teams.

2.9.3 Inputs

Both team_ID's, organization_ID's, and the same user_ID that is part of the team.

2.9.4 Outcome(s)

If the team_ID is part of both teams with the team_ID and organization_ID's, the user should be able to flip back and forth between the teams.

2.9.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team. This is also dependent on UT_04 since multiple teams must be created.

2.10 Calendar clearly shows chosen date

2.10.1 Test case identifier

UT_10

2.10.2 Objective

Test that the calendar clearly shows the chosen date.

2.10.3 Inputs

Inputs include a date.

2.10.4 Outcome(s)

If a user clicks on a date, the correct date is highlighted on the calendar and can be switched when another date is clicked.

2.10.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to see the calendar. This is also dependent on UT_04 since multiple teams must be created.

2.11 Create, save, edit, and destroy a drill

2.11.1 Test case identifier

UT_11

2.11.2 Objective

Test being able for the coach to create a drill and save it in the system. Then, test opening up the drill and editing it further and saving changes to the database, or deleting the drill from the database.

2.11.3 Inputs

Inputs include drawings, name of drill, and any description or categories/tags that correlate with the drill.

2.11.4 Outcome(s)

If a coach creates a new drill, fills out the information, and saves it, then the drill should be saved to the database. If the coach wants to edit the drill further, they will be able to click on the drill and continue editing the information and save it to the database for future use. If the coach deletes the drill, then the drill will be deleted from the database.

2.11.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit any drills in an account and UT_04 since a team must also be created to create a drill.

2.12 Create and add a tag on a new drill and add it on a saved drill

2.12.1 Test case identifier

UT_12

2.12.2 Objective

Test being able to create and add a tag on a new or saved drill.

2.12.3 Inputs

Inputs include the name of the tag.

2.12.4 Outcome(s)

If the coach wants to add a category/tag to a new drill, then the tag will be available to add when filling out new drill information. If a coach wants to add a tag to a saved drill, then the coach can edit the drill and add the tag.

2.12.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit any drills in an account, UT_11 since a drill must be able to be created and saved, and UT_04 since a team must also be created to create a drill.

2.13 Create, save, assign, edit, and delete a practice plan

2.13.1 Test case identifier

UT_13

2.13.2 Objective

Test being able to create, save, assign, edit, and delete a practice plan.

2.13.3 Inputs

Inputs include name of practice plan, any drills, and any categories/tags that correlate with the practice plan.

2.13.4 Outcome(s)

If a coach creates a practice plan, they will be able to add drills or any information to the practice plan and assign the practice plan to a player or to the whole team. The practice plan will be added to the database when saved. When the coach wants to add more or edit the practice plan, they can click on the practice plan and edit the information being pulled from the database and save new information to the database. If a coach deleted the practice plan, the information will be deleted from the database.

2.13.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit any practice plans in an account , UT_11 since a drill must be able to be created and saved to add to the practice plan, and UT_04 since a team must also be created to create a drill.

2.14 Add media, photo/video, to a drill description

2.14.1 Test case identifier

UT_14

2.14.2 Objective

Test adding a media, such as a photo or video, to a drill description.

2.14.3 Inputs

Inputs include a photo or video.

2.14.4 Outcome(s)

If a coach wants to add a photo or video to a drill, the coach can upload a link or photo to the description part of the drill. The photo and link will then be saved to the database for users to see in their accounts.

2.14.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit and drills in an account , UT_11 since a drill must be able to be created and saved to add a photo and video to, and UT_04 since a team must also be created to create a drill.

2.15 Play an animated drill

2.15.1 Test case identifier

UT_15

2.15.2 Objective

Test playing an animated drill.

2.15.3 Inputs

Inputs would include an animated drill.

2.15.4 Outcome(s)

If a user wants to watch an animated drill, the user would click on play and the drill would appear and move how it was made to show the users.

2.15.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit any drills in an account, UT_11 since a drill must be able to be created and saved to add media, and UT_04 since a team must also be created to create a drill.

2.16 Create a drill with multiple different field views and put them on the same practice plan

2.16.1 Test case identifier

UT_16

2.16.2 Objective

Test creating a drill with multiple different field views and putting them on the same practice plan.

2.16.3 Inputs

Inputs include different field view drills.

2.16.4 Outcome(s)

If a coach wants to add different drills onto a practice plan, the coach would be able to select the different drills and add them to the specified practice plan for players to see. The practice plan would be saved to the database for future use.

2.16.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit any practice plans in an account , UT_11 since a drill must be able to be created and saved to add to the practice plan, and UT_04 since a team must also be created to create a drill.

2.17 Add stats to keep track of on a drill, put it in a practice plan, execute the plan, add a stat in execution, and see the stat in the results

2.17.1 Test case identifier

UT_17

2.17.2 Objective

Test a coach's ability to add statistics to track for a drill and put that drill in a practice plan. Test the ability to see how the statistic is tracked as the practice progresses.

2.17.3 Inputs

A given statistic that will be worked on in a given team practice as well as a drill that implements skills related to that statistic

2.17.4 Outcome(s)

After a specific statistic is added to a drill and that drill is added to a practice plan, after the practice is completed, the coach can update that specific skill with the progress made during the team practice.

2.17.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit and drills in an account , UT_11 since a drill must be able to be created and saved to add a photo and video to, and UT_04 since a team must also be created to create a drill.

2.18 View all drills saved in the drill database from each user type

2.18.1 Test case identifier

UT_18

2.18.2 Objective

Be able to view all saved drills as a player, coach, parent, and owner

2.18.3 Inputs

It is assumed at least one drill has already been added to the database

2.18.4 Outcome(s)

As a player, coach, parent, or owner, the drills created will be present and viewable in the drill section of the user interface.

2.18.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit and drills in an account , UT_11 since a drill must be able to be created and saved to add a photo and video to, and UT_04 since a team must also be created to create a drill.

2.19 Execute a practice plan for 3 minutes, switch between drills

2.19.1 Test case identifier

UT_19

2.19.2 Objective

Be able to “start” a practice plan with an initial drill and after three minutes of that drill, switch to a second drill.

2.19.3 Inputs

A given initial drill to start the practice on, and a second drill to switch to after three minutes have passed.

2.19.4 Outcome(s)

After three minutes of executing a single drill, the coach can switch to the next drill in the practice plan seamlessly and the UI will respond with the new drill ready to begin.

2.19.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create and edit and drills in an account , UT_11 since a drill must be able to be created and saved to add a photo and video to, and UT_04 since a team must also be created to create a drill. It is also dependent on UT_13 as a coach must be able to make a practice plan.

2.20 Create a team goal as a coach, execute a practice plan with the same tags as the goal

2.20.1 Test case identifier

UT_20

2.20.2 Objective

Create a tagged team goal as a coach and execute a practice plan suggested by the application with the same tags as the goal.

2.20.3 Inputs

A team goal with tags (such as skill worked on, position focused on, etc.)

2.20.4 Outcome(s)

Receive a suggested practice plan from the application with the same tags as the goal and execute that plan.

2.20.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a goal and execute a practice plan. It is also dependent on UT_13 as a coach must be able to make a practice plan, and UT_23 as a team goal must be able to be created.

2.21 Create and update individual goals as a player

2.21.1 Test case identifier

UT_21

2.21.2 Objective

Enter the application as a player and create and update an individual goal.

2.21.3 Inputs

A goal for a certain player to achieve based on a skillset or statistic.

2.21.4 Outcome(s)

After creation, the goal will be present in the UI and after edits are made, the changes are present in the UI. This is also dependent on UT_04 since a team must be created.

2.21.5 Intercase dependencies

This test case is dependent on UT_01 since a player account must be created in order to create a goal. This is also dependent on UT_04 since a team must be created.

2.22 Destroy individual goals of a player as a coach

2.22.1 Test case identifier

UT_22

2.22.2 Objective

As a coach, go into the goals of a player and remove a goal.

2.22.3 Inputs

A given goal that the coach wants to remove.

2.22.4 Outcome(s)

After removal, the goal should be absent from both the coach and player UI.

2.22.5 Intercase dependencies

This test case is dependent on UT_01 since a player account must be created in order to create a goal, as well as UT_21 as the player must be able to create a goal so that it can be removed by the coach.

2.23 Create, save, assign, edit, and delete a team goal as a coach

2.23.1 Test case identifier

UT_23

2.23.2 Objective

Create a team goal as a coach and save it. Assign the goal to a player or players. Edit the goal. Then remove the goal.

2.23.3 Inputs

A skill or statistic to base the goal off of.

2.23.4 Outcome(s)

After creation, the goal should be present in the UI. After assignment, the goal should be seen in the UI by the player its assigned to. After edits, the changes should be present in both the player and coach UI. After deletion, the goal should be removed from the database and the player/coach UI.

2.23.5 Intercase dependencies

This test case is dependent on UT_01 since an account must be created in order to create a team goal. This is also dependent on UT_04 since a team must be created.

2.24 Create, save, edit, and delete an individual goal

2.24.1 Test case identifier

UT_24

2.24.2 Objective

Perform CRUD operations as a player on an individual goal

2.24.3 Inputs

A statistic or skillset to base the goal off of

2.24.4 Outcome(s)

After creation and saving, the goal should be present in the player UI. After editing, the changes should be present in the UI. After deletion, the goal should no longer be present in the player UI

2.24.5 Intercase dependencies

This test case is dependent on UT_01 since a player account must be created in order to create a goal, as well as UT_21 as the player must be able to create a goal so that CRUD operations can be performed.

2.25 Environmental Needs

2.25.1 Hardware

In order to execute the test cases, a developer should have a computer that can connect to the internet.

2.25.2 Software

Jest will be used for the JavaScript unit tests. A developer should consider using VSCode or Intellij to view, manage the files, and run unit tests.

2.26 Special procedural requirements

In order to execute the test cases, a developer should run a script in the command line that runs through and checks all the test cases. If the test case is passed, it will be shown with a '.' character in the command line. On the other hand, if it fails, it will be shown with a 'x' character in the command line. Additionally, the passes and failures should show up in an XML document that is generated to view the results of pass or fail.

3. General

3.1. Glossary -Tyler

A

- API (Application Programming Interface): A set of routines, protocols, and tools that allow different software applications to communicate with each other.
- AI/ML (Artificial Intelligence/Machine Learning): Refers to the application of machine learning algorithms in PocketSports to provide recommendations or analyze performance data.

C

- Coach: A user role in PocketSports that allows the creation and management of teams, players, drills, practice plans, and tracking of statistics.

- CRUD Operations: Refers to the basic Create, Read, Update, and Delete operations for data management in the app.

D

- Drill: A planned practice activity designed to improve player skills, which can be created, edited, and assigned to practice sessions.

E

- Entity: In data modeling, an entity is an object or concept that is represented in the system, such as a Player or a Team.

F

- Functional Requirements: Specific system behaviors or functionalities that PocketSports must perform to meet user needs (player stat tracking, practice plan creation).

P

- Parent: A user role that allows viewing of their child's progress and team information in the app, but does not grant editing permissions.
- Player: A user role that allows athletes to view and track their practice sessions, goals, and performance statistics.
- Practice Plan: A set of drills and instructions created by coaches, designed for a specific practice session.

S

- Stat Tracking: A feature in PocketSports that allows the tracking of player statistics such as goals, assists, and other performance metrics during practice and games.

T

- Team: A group of players and coaches managed within the PocketSports application, where practice plans, stats, and drills are tracked.
- Test Case: A set of conditions or variables used to determine if the system behaves as expected during testing.

U

- User Acceptance Testing (UAT): A phase of testing where real-world users (such as coaches or players) test the system to ensure it meets their needs.

3.2. Document change procedures and history -Tyler

Name	Date	Reason for Changes	Version
Garrett Gmeiner	9/29/24	Initial Creation	1.0

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4578383 PAGE 53
Test Plan ("How to verify that the system does what it should do?")
at least teamSize * 1 page (single-spaced, 12-pt font, 1-in margins) [for example, at least 3 pages for a team with 3 members]
for each required feature/behavior in the Requirement Document, discuss test cases to verify the feature/behavior.
Generally, test cases are designed to find bugs. Hence, besides the "usual" input values, "ununsal" input values should be considered as well.
[e.g. page 53 of IEEE Standard for Software Test Documentation]