Requirements Based Test Cases for Sports League Administration Management System

Version 1.0 approved

Prepared by Anushka Angamuthu, Grace Bowser, Maryam Masood, & Stephen Tobisch

Group 1

Municipal Parks & Recreation Department

CSCI 5801, Professor Kevin Wendt

November 6th, 2021

1. Test Requirements

1.1 Objective

The purpose of the Test Requirements section is to list ALL hardware and software test requirements, whether explicitly determined from any relevant documents or implicitly determined from experience and product knowledge. For most projects, the documents referred to may be the Product Definition Document, Software/Hardware Requirements Specification and perhaps the Software/Hardware Design Specification. A Test Case Matrix is provided that simply lists all the test cases by title or description, and includes a method of tracking when the test case was run and whether it passed or not.

1.2 Definitions and Acronyms

List any technical terms or acronyms used in the document, along with their meanings. Examples for this document:

SRS Software Requirements Specification

TM Traceability Matrix

1.3 Traceability Matrix

The purpose of a Traceability Matrix is to show which test cases verify which requirements. A possible format for a Traceability Matrix is as follows

3.1

Requirement / Test Case	Test Case ID 1	Test Case ID 2	Test Case ID 3	Test Case ID 4	Test Case ID 5	Test Case ID 6	Test Case ID 7
Requirement 3.1.1	x						
Requirement 3.1.2		x					
Requirement 3.1.3			x				
Requirement 3.1.4				x			
Requirement 3.1.5					x		
Requirement 3.1.6						x	
Requirement 3.1.7							x

3.2 - 3.4

Requirement / Test Case	Test Case ID 8	Test Case ID 9	Test Case ID 10	Test Case ID 11	Test Case ID 12	Test Case ID 13
Requirement 3.2.1	x					
Requirement 3.2.2		x				
Requirement 3.3.1			x			
Requirement 3.3.2				x		
Requirement 3.4.1					x	
Requirement 3.4.2						x

3.5

Requirement / Test Cases	Test Case ID 14	Test Case ID 15	Test Case ID 16	Test Case ID 17
Requirement 3.5.1	x			
Requirement 3.5.2		x		

Requirement 3.5.3			x	x

3.6

Requirement / Test Cases	Test Case ID 18	Test Case ID 19	Test Case ID 20	Test Case ID 21
Requirement 3.6.1	x			
Requirement 3.6.2		x		
Requirement 3.6.3			x	x

3.7

Requirement / Test Cases	Test Case ID 22	Test Case ID 23
Requirement 3.7.1	x	
Requirement 3.7.2		x

3.8

Requirement / Test Cases	Test Case ID 24
Requirement 3.8.1	x

3.9

Requirement / Test Cases	Test Case ID 25
Requirement 3.9.1	x

3.10

Requirement / Test Cases	Test Case ID 26
Requirement 3.10.1	x

3.11

Requirement / Test Cases	Test Case ID 27	Test Case ID 28
Requirement 3.11.1	x	

Requirement 3.11.2		x

3.12

Requirement / Test Cases	Test Case ID 29
Requirement 3.12.1	x

3.13

Requirement / Test Cases	Test Case ID 30
Requirement 3.13.1	x

2. Test Cases

2.1 Description

Owners: Anushka Angamuthu, Grace Bowser, Maryam Masood, Stephen Tobisch

1 Create a profile

Description: Ensures that a user has the ability to create a profile under their desired role.

Test Inputs: Profile parameters as follows:

Name: Grace Bowser

Role: One test will be done for each of the following roles: parent, player (adult, youth, and free agent).

Address: 420 SE 23rd Ave, Minneapolis, MN 55455

Communication (Phone and Email) 612-624-8080 bowse025@umn.edu

Authentication Information (User's Password): CSCi5801@!

Expected Results: The user's profile will be successfully created with the following information:

Name: Grace Bowser

Role: One test will be done for each of the following roles: parent, player (adult, youth, and free agent).

Address: 420 SE 23rd Ave, Minneapolis, MN 55455

Communication (Phone and Email) 612-624-8080 bowse025@umn.edu

Authentication Information (User's Password): CSCi5801@!

Dependencies: None

Initialization: Functionality of profiles supported by software.

Test Steps:

1. Request that profile is to be created.
2. Enter user information (name, contact information, role, address, password)
3. Verify that the password entered meets the security specifications.

4. Finish creation and verify the profile was successfully created by receiving the confirmation message.

2 Log in

Description: Ensures that a user has to log in under their profile.

Test Inputs: Profile parameters as follows:

Email: bowse025@umn.edu

Authentication Information (Password): CSCi5801@!

Expected Results: Login for bowse025@umn.edu's profile will be successful. If the password entered on test does not match CSCi5801@!, login will not be successful. If the email bowse025@umn.edu is entered incorrectly, login to that profile will be unsuccessful.

Dependencies: 1 (user must have an existing profile)

Initialization: Login interface exists

Test Steps:

1. Go to the login section of the system.
2. Enter in username and password into respective boxes for each.
3. Verify that login was successful based on no outputs (only receive outputs if login was unsuccessful. This output will be an error message).

3 Register (Admin)

Description: Ensures that admin, coaches, managers, or officials can register in order to be granted higher permissions than players/parents. Ensures that higher permissions are granted upon registration.

Test Inputs: Profile parameters as follows:

Email : angam003@umn.edu

Password : aDmIn12!

Name : Anushka Angamuthu

Privilege Role : enter role (either official, admin, coach, manager). One test case should be done for each of these roles.

Expected Results: A profile will be created for Anushka with the correct password (aDmIn12!) and email (angam003@umn.edu). The profile will receive different permissions based on the role entered. These cases are:

1. Official - will be able to enter scores from games
2. Admin - will be able to register other admin/coaches/managers/officials, and will also be able to appoint and assign officials, create leagues/tournament brackets, create schedules for leagues/teams, report any statistics, and affect league makeups.
3. Coach - should be able to create teams
4. Manager - should be able to create teams

Dependencies: None

Initialization: The user's profile is created in the system and the user is granted correct permissions.

Test Steps:

1. Request that profile is to be created.
2. Enter user information (name, contact information, role, address, password)
3. Verify that the password entered meets the security specifications.
4. Finish creation and verify the profile was successfully created by receiving the confirmation message.
5. Further test permissions granted to the account by trying to complete tasks that require the set of permissions that should be granted by status.
6. Further test permissions by trying to complete actions that are out of the scope of the profile created.

4 Modify Account Information

Description: Ensures that a user has the ability to modify their account information.

Test Inputs: The number of parameters for this can vary. The number of parameters is dependent on the number of items the user wishes to modify in their profile. Since the user is modifying their profile, the user will be taken to their profile page to make edits, and the user can erase the current input and modify it. Say that the initial profile looked like the following:

First name : Stephn
Last name : Tobisc
Gender identification : Female
Birthday : 05/30/2000
Phone number : 612-624-8081
Address : 420 23rd Ave Minneaplis, MN 55414
Email : bowse025@umn.edu
Password : goGophers1!

Then, valid inputs for modification would include the following:

First name : Stephen
Last name : Tobisch
Gender identification : Male
Birthday : 01/18/2000
Phone number : 612-624-8080
Address : 420 23rd Ave Minneapolis, MN 55414
Email : tobis004@umn.edu
Password : goGophs1!

In the testing, all of these attributes should be tested for modification at least once.

Expected Results: In the case the user wants to change all of the information in the profile, the results would look like:

First name : Stephen
Last name : Tobisch
Gender identification : Male
Birthday : 01/18/2000
Phone number : 612-624-8080
Address : 420 23rd Ave Minneapolis, MN 55414
Email : tobis004@umn.edu
Password : goGophs1!

Dependencies: If player or parent : 1
If coach, manager, official or admin : 3
Initialization: The user's profile is created in the system.
Test Steps:

1. Create profile for testing.
2. Login to existing profile.
3. Request that profile is to be modified.
4. System takes the user to the profile page that allows modification.
5. Change attributes of profile.
6. Save changes and check if modifications are correctly saved and visible on profile.

5 Delete account

Description: Ensures that user possesses the ability to delete their account
Test Inputs: Parameters as follows:
Email : masoo013@umn.edu
Password : it'SNovember7!
Expected Results: The account under masoo013@umn.edu will be deleted.
Dependencies: If parent or player : 1 (profile should already exist)
If official, coach, manager, or admin : 3 (profile should already exist)
In either of these dependencies, the account being deleted should be "fake", or have been created under the pretenses that it would be used to test the deletion feature. The profile should not be a real profile that someone requires use for.
Initialization: The user's profile has already been created.
Test Steps:

1. Create profile for testing
2. Request that profile is to be deleted.
3. Enter user information (email and password) one more time to verify deletion.
4. Receive confirmation message that the profile was deleted.
5. Attempt to login in again and receive a message that profile does not exist.

6 Request Access for role of higher scope

Description: Ensures that a role can be added to a profile when the role is of higher scope than the current role on the profile.

Test Inputs: Parameters as follows:

Existing Profile : Grace Bowser's profile

Request Feature : There should be a request feature supported by the interface that the test will need to access. This is constant throughout the whole system.

Expected Results: The role of higher scope is successfully added to Grace Bowser's profile, and permissions granted with that role are also given to Grace Bowser.

Dependencies: If parent or player, 1

If coach, manager, official, or admin : 3

Initialization: The user's profile is created in the system and the request feature can be found by the user.

Test Steps:

1. Create profile for testing.
2. Log in to existing profile
3. Test request feature by requesting the new role (should try this step for each of the roles that are considered higher scope (coach, official, manager, admin)).
4. An admin should be aware that the testing is occurring, so an admin is aware to accept the request of the test profile.
5. Upon successful acception of the request, the permissions are granted to the profile.
6. The permissions should then be tested by trying to access each of the new features (Ex: if role of official is granted, testing should be done to see whether or not profile can enter scores).
7. Test should then also look to see if profile can be denied, and ensure that someone who is not eligible for the role requested is denied permission.
    a. For this step, will need to use another existing profile.

7 Add role of lower scope

Description: Ensures that a role of lower or equal scope can be added to profile.

Test Inputs: Parameters include:

Existing Profile : Anushka Angamuthu's profile

Role : List of roles the profile is qualified to add should be tested. One test should be done for each role to ensure that the correct settings are on each of the roles (ex: ensure that roles of higher scope are not appearing in eligible roles to add to profile).

Expected Results: Role of lower scope successfully added to the profile (will have result matching for each of the roles added).
Dependencies: 1 or 3 (depending on current role). Profile should exist in the system.
Initialization: The user's profile is created in the system and the user is granted the set of current permissions based on role(s) already on their profile.
Test Steps:

1. Go to the list of roles eligible to be added on profile.
2. Select role to be added.
3. Receive confirmation.
4. View change updated to profile.

8 Register Adult Players

Description: Ensures that the registration functionality adds the user's information to the database.
Test Inputs: Profile parameters as follows, in the order as specified by the database's insertion function:

Name: Grace Bowser
Role: Adult player
Address: 420 SE 23rd Ave, Minneapolis, MN 55455
Phone number: 612-624-8080
Email: bowse025@umn.edu
Prefered Communication: phone number/email
Authentication Information (User's Password): CSCi5801@!
Authentication Information Confirmation: password entered twice to make sure that it meets security requirements and matches the first password

Expected Results: The user's profile will be successfully added to the database, and a success message is sent to the front-end functionality.
Dependencies:

1. User must have a profile to register

Initialization: The user's profile is created in the system.
Test Steps:

1. Request that profile is to be created.
2. Enter user information (name, contact information, role, address, password)
3. Verify that the password entered meets the security specifications.
4. Verify that the user has a profile in the system
5. Verify that the user is not already registered in the system.
6. Finish creation and verify the profile was successfully created by receiving the confirmation message.

9 Register Youth Players

Description: Ensures that the registration functionality adds the user's information to the database.

Test Inputs: Profile parameters as follows, in the order as specified by the database's insertion function:

Name: Grace Bowser
Role: Adult player
Address: 420 SE 23rd Ave, Minneapolis, MN 55455
Phone number: 612-624-8080
Email: bowse025@umn.edu
Prefered Communication: phone number/email
Parent/Guardian Name: Sally Bowser
Authentication Information (User's Password): CSCi5801@!
Authentication Information Confirmation: password entered twice to make sure that it meets security requirements and matches the first password

Expected Results: The user's profile will be successfully added to the database, and a success message is sent to the front-end functionality.

Dependencies:

1. User must have a profile to register
2. User's parent/guardian must have a profile in the system for the user to register

Initialization: The user's profile is created in the system.

Test Steps:

1. Request that profile is to be created.
2. Enter user information (name, contact information, role, address, password)
3. Verify that the password entered meets the security specifications.
4. Verify that the user has a profile in the system
5. Verify that the user's parent/guardian has a profile in the system.
6. Verify that the user is not already registered in the system.
7. Finish creation and verify the profile was successfully created by receiving the confirmation message.

10 Appoint Coaches

Description: Tests if the "appoint coaches" functionality assigns a coach to a league.

Test Inputs: Appointment parameters as follows:

Coach Name: Stephen Tobisch
League Name: The Big Red

Expected Results: The coach's identifying information will be present in the intended league's coach field in the database, and a success message is sent to the front-end functionality.

Dependencies:

1. The coach must have a profile in the system (must be identifiable by the system)

2. The league must be present in the system (must be identifiable by the system)

Initialization: The coach is appointed to a league.

Test Steps:

1. Request to appoint a coach to a league is received from the front-end functionality.
2. Enter coach's identifying information into the system's search functionality.
3. Search up the intended league.
4. Send this coach's identifying information as an input to the coach appointment functionality.
5. Verify that the coach's identifying information will be present in the intended league's coach field in the database.

11 Assign Coaches

Description: Tests if the "assign coaches" functionality assigns a coach to a team.

Test Inputs: Assignment parameters as follows:

Coach's Name: Stephen Tobisch

Team Name: Stephen's Team

Expected Results: The coach's identifying information will be present in the intended team's coach field in the database, and a success message is sent to the front-end functionality.

Dependencies:

1. The coach must have a profile in the system (must be identifiable by the system)
2. The team must be present in the system (must be identifiable by the system)

Initialization: The coach is assigned to a team.

Test Steps:

1. Request to assign a coach to a team is received from the front-end functionality.
2. Enter coach's identifying information into the system's search functionality.
3. Search up the intended team.
4. Send this coach's identifying information as an input to the coach assignment functionality.
5. Verify that the coach's identifying information will be present in the intended team's coach field in the database.

12 Appoint Officials

Description: Tests if the "appoint official" functionality assigns an official to a league.

Test Inputs: Appointment parameters as follows:

Official's Name: Anushka Angamuthu

League Name: Anushka's League

Expected Results: The official's identifying information will be present in the intended league's official field in the database, and a success message is sent to the front-end functionality.

Dependencies:

1. The official must have a profile in the system (must be identifiable by the system)
2. The league must be present in the system (must be identifiable by the system)

Initialization: The official is appointed to a league.

Test Steps:

1. Request to appoint an official to a league is received from the front-end functionality.
2. Enter official's identifying information into the system's search functionality.
3. Search up the intended league.
4. Send this official's identifying information as an input to the coach appointment functionality.
5. Verify that the official's identifying information will be present in the intended league's coach field in the database.

13 Assign Officials to Games

Description: Tests if the "assign official" functionality assigns an official to a game.

Test Inputs: Assignment parameters as follows:

Official's Name: Anushka Angamuthu

League Name: Anushka's Game

Expected Results: The official's identifying information will be present in the intended game's official field in the database, and a success message is sent to the front-end functionality.

Dependencies:

1. The official must have a profile in the system (must be identifiable by the system)
2. The game must be present in the system (must be identifiable by the system)

Initialization: The official is assigned to a game.

Test Steps:

1. Request to assign an official to a game is received from the front-end functionality.
2. Enter official's identifying information into the system's search functionality.
3. Search up the intended game.
4. Send this official's identifying information as an input to the official assignment functionality.
5. Verify that the official's identifying information will be present in the intended game's official field in the database.

14 Create League

Description: Tests if the league creation functionality works and creates a league in the system.

Test Inputs: Creation parameters as follows:

League Name: Big Red Baseball League

Teams (listed by names): Anushka's Team, Grace's Team, Stephan's Team

Expected Results: The league is created in the system, and is populated with the provided teams.

Dependencies:

1. The teams must be in the system.

Initialization: The league is created in the system.

Test Steps:

1. Request to create a league
2. Enter league name
3. Enter names of the teams in the league
4. Verify that the league is now in the system and is populated with the correct teams

15 Create league schedule

Description: Ensures that a league schedule is generated.

Test Inputs: League scheduling parameters as follows:

Dates: Tuesday 11/16/21, Thursday 11/18/21

Times: 6:00pm

Locations: Central Park

Number of games per week: 2

League: B soccer

Duration of games: 2 hours

Expected Results: A schedule is generated using the input values. It schedules the following two games on

- Tuesday 11/16/21 from 6:00pm-8:00pm.
- Thursday 11/18/21 from 6:00pm-8:00pm.

Both games are located in Central Park. No exception is thrown.

Dependencies: None

Initialization: The league B soccer has already been created and configured in the system.

Test Steps:

1. The request for the schedule is received.
2. Verify that all the parameters are met.
3. Verify that the schedule is following all constraints.
4. Verify that there are no scheduling conflicts.

16 Modify league schedule

Description: Ensures that a league schedule is able to be modified.

Test Inputs: League scheduling parameters that need modification as follows:

League schedule modified: B Soccer
Dates of games modified: Tuesday 11/16/21, Thursday 11/18/21
New dates: Monday 11/15/21, Wednesday 11/17/21, Friday 11/19/21
Times: 5:00pm
Locations: Riverside Park
Number of games per week: 3
Duration of games: 3 hours

Expected Results: The previous schedule created in test case 15 is modified based on the input modifications. The previous schedule is changed such that

- The schedule now has three games: Monday 11/15/21, Wednesday 11/17/21, and Friday 11/19/21. The previous two games are no longer on the schedule.
- Each game starts at 5:00pm and ends at 8:00pm.
- The locations of the games are now at Riverside Park.

No exception is thrown.

Dependencies: Test case 15

Initialization: The league B soccer has already been created and configured in the system and test case 15 is run before this test case.

Test Steps:

1. The request for the modified schedule is received.
2. Verify that all parameters/modifications are met.
3. Verify that the new schedule is following all constraints.
4. Verify that there are no scheduling conflicts.

17 Delete league schedule

Description: Ensures that a league schedule is deleted.

Test Inputs: Parameters as follows:

League: B Soccer

Expected Results: The schedule for the league B soccer is deleted from the system. No exception is thrown.

Dependencies: Test case 15

Initialization: The league B soccer has already been created and configured in the system. Test case 15 has already been run so a schedule has already been created for the league.

Test Steps:

1. The request for deleting the schedule is received.
2. Receive confirmation that the schedule would like to be deleted.
3. Verify that the schedule is deleted from the system.

18 Create tournament bracket

Description: Ensures that a tournament bracket is generated.

Test Inputs: Tournament bracket parameters as follows:

Tournament name: B Soccer Round-robin
Associated league: B soccer
Tournament constraints: None
Tournament type: Round-robin
Tournament minimum qualifications: .500 record
Number of teams: 3
Team names: Gophers, Fighting Fleck's, Kevin's Favorites

Expected Results: A tournament bracket is generated such that a round-robin tournament is created. That is, each team plays every other team. There should be a total of three rounds (in any order):

1. Gophers vs Fighting Fleck's. Kevin's Favorites has a bye.
2. Gophers vs Kevin's Favorites. Fighting Fleck's has a bye.
3. Fighting Fleck's vs Kevin's Favorites. Gophers has a bye.

No exception is thrown.

Dependencies: None

Initialization: The league B soccer has already been created and configured in the system. All three of the teams should have a minimum of a .500 record.

Test Steps:

1. The request for the tournament bracket is received.
2. Verify that all the parameters are met.
3. Verify that there is no preexisting tournament with an identical name and league.

19 Create tournament schedule

Description: Ensures that a tournament schedule is generated.

Test Inputs: League scheduling parameters as follows:

Dates: Saturday 11/13/21
Times: 12:00pm
Locations: Central Park
Number of games per week: 1
Tournament: B Soccer Round-robin
Duration of games: 2 hours

Expected Results: A schedule is generated using the input values and the bracket generated for the B Soccer Round-robin tournament. Each game should have the following times:

- Gophers vs Fighting Fleck's from 12:00pm-2:00pm.
- Gophers vs Kevin's Favorites from 2:00pm-4:00pm.
- Fighting Fleck's vs Kevin's Favorites from 4:00pm-6:00pm.

In addition, each game takes place on Saturday 11/13/21 in Central Park. No exception is thrown.

Dependencies: Test case 18

Initialization: The league B soccer has already been created and configured in the system. The test case 18 is run before this test case.

Test Steps:

1. The request for the tournament schedule is received.
2. Verify that all the parameters are met.
3. Verify that the schedule is following all constraints.
4. Verify that there are no scheduling conflicts.

20 Modify tournament

Description: Ensures that a tournament is able to be modified.

Test Inputs: Tournament parameters as follows:

Tournament modified name: B Soccer Round-robin
Dates of games modified: Saturday 11/13/21
New dates: Saturday 11/20/21, Sunday 11/21/21
Times: 2:00pm
Locations: Riverside Park
Number of games per week: 2
Duration of games: 3 hours
Tournament name: B Soccer Playoff
Associated league: B soccer
Tournament constraints: Seeding based on record
Tournament type: Single elimination
Tournament minimum qualifications: .600 record
Number of teams: 3
Team names: Gophers, Fighting Fleck's, Kevin's Favorites

Expected Results: The tournament B Soccer Round-robin is modified such that it is now named B Soccer Playoff, and it is now a single elimination tournament with seeding. It should have two rounds that span two days and take place in Riverside Park. The schedule should be consistent with the following two rounds:

1. Gophers vs Fighting Fleck's from 2:00pm-5:00pm on Saturday 11/20/21. Kevin's Favorites have a bye.
2. Winner from round 1 vs Kevin's Favorites from 2:00pm-5:00pm on Sunday 11/21/21.

This is under the assumption that Gophers have the worst record and are seeded 3, Fighting Fleck's have the next worst record and are seeded 2, and Kevin's Favorites have the best record and are seeded 1. No exception is thrown.

Dependencies: Test cases 18 and 19

Initialization: The league B soccer has already been created and configured in the system. The test cases 18 and 19 have been run before this test.

Test Steps:

1. The request for the modified tournament is received.

2. Verify that all the parameters/modifications are met.
3. Verify that there is no preexisting tournament with an identical name and league.
4. Verify that the modified tournament schedule is following all constraints.
5. Verify that there are no scheduling conflicts.

21 Delete tournament

Description: Ensures that a tournament is deleted.

Test Inputs: Parameters as follows:

Tournament: B Soccer Round-robin

Expected Results: The tournament B soccer Round-robin is deleted from the system. No exception is thrown.

Dependencies: Test case 18

Initialization: The league B soccer has already been created and configured in the system. Test case 18 has already been run so the tournament B soccer Round-robin is created in the system.

Test Steps:

1. The request for deleting the tournament is received.
2. Receive confirmation that the tournament would like to be deleted.
3. Verify that the tournament is deleted from the system.

22 Get report with tournament scores

Description: Ensures that a report including tournament scores is generated.

Test Inputs: The games that will be included in the report will be using the parameters as follows:

Tournament name: B Soccer Playoff
Tournament game dates and times: Saturday 11/20/21 from 2:00pm-5:00pm, Sunday 11/21/21 from 2:00pm-5:00pm

Expected Results: A report should be generated including the scores of the games in the tournament that were selected in the input. The report should display the following scores:

- Fighting Fleck's 3, Gophers 0
- Kevin's Favorites 3, Fighting Fleck's 2

No exception is thrown.

Dependencies: Test cases 18, 19, and 20

Initialization: The league B soccer has already been created and configured in the system. The test cases 18, 19, and 20 have been run before this test. The scores for both games have already been entered into the system consistent with the scores in the expected results.

Test Steps:

1. The request for generating the report with scores is received.
2. Verify that all the scores from the games inputted are in the report.
3. Display the report.

23 Get report with tournament standings

Description: Ensures that a report including tournament standings is generated.

Test Inputs: League scheduling parameters as follows:

Tournament name: B Soccer Playoff

Expected Results: A report should be generated including the standings of the teams for the tournament selected in the input. The report should display the following standings:

- Gophers: 3rd place
- Fighting Fleck's: 2nd place
- Kevin's Favorites: 1st place

No exception is thrown.

Dependencies: Test cases 18, 19, and 20

Initialization: The league B soccer has already been created and configured in the system. The test cases 18, 19, and 20 have been run before this test. The tournament standings have already been entered in the system consistent with the standings in the expected results.

Test Steps:

1. The request for generating the report with standings is received.
2. Verify that the teams are in the report.
3. Verify that the team standings are correct based on tournament record.
4. Display the report.

24 Get suggestions for league makeup

Description: Ensures that staff are able to view suggestions for league makeup improvements.

Test Inputs: The league name which the staff would like to view league makeup changes for:

League name: B Soccer

Expected Results: A report is generated which staff are able to view. The report shall contain suggestions for the following:

1. If the number of players in the league shall be increased or decreased and to which amount
2. If the number of teams in the league shall be increased or decreased and to which amount
3. If the number of games in the league shall be increased or decreased and to which amount
4. If the frequency of games played in the league shall be increased or decreased and to which amount
5. If the game durations in the league shall be increased or decreased and to which amount paired with their respective games

Dependencies: None

Initialization: The league B soccer has been created and configured in the system.

Test Steps:

1. The request to view league makeup suggestions has been received.
2. Verify that all parameters have been met.
3. Verify that a report has been generated including the components outlined in steps 1 - 5 of the expected results.

25 Get financial account balance

Description: Ensures that staff may view a user's account balance.

Test Inputs: The name of the user in the system whose account balance the staff would like to view:

Account holder name: Scott Gagnon

Expected Results: The current account balance for the user named 'Scott Gagnon' in the system which we expect to be $12.65. No exception is thrown.

Dependencies: None

Initialization: The user is Scott Gagnon already in the system.

Test Steps:

1. The request to view the user's account balance has been received.
2. Verify that all parameters have been met.
3. Verify that the account balance matches the expected result.

26 Create team

Description: Ensures that coaches may create a team.

Test Inputs: Team creation parameters as follows:

Team Name: The best soccer team
Players: Scott Gagnon, Adam Larson
League: B soccer
Sport: Soccer
Coach: Kevin Wendt
Division: D1

Expected Results: A team is generated using the input values. Its name is 'The best soccer team'. It has players Scott Gagnon, Adam Larson and coach Kevin Wendt. It has been created in league B soccer and division D1 under the sport 'Soccer'. No exception is thrown.

Dependencies: None

Initialization: The players and coach listed in the input parameters are already in the system.

Test Steps:

1. The request to create the team is received.
2. Verify that all the parameters are met.

3. Verify that there are no player conflicts and the team is not pre-existing in the system.

27 Get official scheduled availability

Description: Ensures that an official may view the games they are working on and modify their availability.

Test Inputs: Official availability parameters as follows (all are optional parameters):

Blocked date(s): 11/18/2021, 11/25/2021
Block times(s): 12:00 - 2:00 PM CST Sat., Sun.
Block location(s): Central Park

Expected Results: A generated schedule for the official. It shall include the games which they are scheduled to work along with their dates, times, teams, leagues, and locations. It shall also include any blocked times that are indicated. No exception is thrown.

Dependencies: None

Initialization: The official is already in the system.

Test Steps:

1. The request to enter blocked dates, times, and/or locations is received.
2. Verify that all the parameters are met.
3. Verify that there are no scheduling conflicts.

28 Get scheduling conflict alert

Description: Ensures that an official is alerted to any conflicts that occur during scheduling.

Test Inputs: None

Expected Results: A scheduling conflict message shall be sent to an official via their preferred communication method. It shall contain the type of schedule conflict and the date and time of the conflict.

Dependencies: None

Initialization: The official is already in the system.

Test Steps:

1. Verify that a scheduling conflict message has been sent.
2. Verify that it contains the type of schedule conflict and the date and time of the conflict.

29 Submit a game score

Description: Ensures that an official may submit a final game score.

Test Inputs: Final game scoring parameters as follows:

Team 1 score: 28
Team 2 score: 64

Forfeit/withdrawal (yes/no): no

Expected Results: The coach shall receive a confirmation that their scores for Team 1 of ‘28’ and Team 2 of ‘64’ have been submitted . No exception is thrown.

Dependencies: None

Initialization: The teams ‘Team 1’ and ‘Team 2’ are already in the system.

Test Steps:

1. The request to enter final game scores for each team is received.
2. Verify that all the parameters are met.
3. Verify that officials have been provided a message of their score submission confirmation.

30 Sponsor a team

Description: Ensures that a user may sponsor a team.

Test Inputs: Sponsor registration parameters as follows:

Team to Sponsor: The best soccer team

Sponsorship amount: $68,795,213

Payment method: Visa ending in 6203

Expected Results: The user shall receive a confirmation that their sponsorship for ‘The best soccer team’ for $68,795,213 has been submitted and they shall be charged on the Visa card ending in 6203. No exception is thrown.

Dependencies: None

Initialization: The user is already in the system and the payment method ‘Visa ending in 6203’ has been initialized in their financial profile.

Test Steps:

5. The request to sponsor the indicated team is received.
6. Verify that all the parameters are met.
7. Verify that sponsors have the indicated payment method linked to their financial account and that it is valid.
8. Verify that sponsors have been provided a message of their score submission confirmation.