Lab 3 - Team-42

Ford St. John, Eduardo Rocha, Robert Arenas, Kyle Dean

Features (user stories) to Implement in Next Sprint:

1. NFL Combine from 1987 to present: (combine.csv)
    a. Feature 1: as a user I want to see the combine performance of player *p*
    b. Feature 2: as a user I want to see the combine performance for year *y*
    c. Feature 3: as a user I want to see the top *n* players for combine measure *c* for year *y*
    d. Feature 4: as a user I want to see the combine performance for players from college *x*

2. NFL League Rushing Statistics: (rusher.csv , players.csv, draft.csv)
    a. Feature 5: as a user I want to see the total rushing performance of player *p*
    b. Feature 6: as a user I want to see the total rushing yards for each team *x*
    c. Feature 7: as a user I want to see the top *n* players for rushing for team *x*
    d. Feature 8: as a user I want to see the top rusher of all time *t*

3. NFL League Receiver Statistics: (receiver.csv, players.csv, draft.csv)
    a. Feature 9: as a user I want to see Player Total receiving yards
    b. Feature 10: as a user I want to see total receiving yards for each team
    c. Feature 11: as a user I want to see Player with most receiving yards
    d. Feature 12: as a user I want to see teams with most receiving yards

4. NFL League Passer Statistics: (passer.csv, players.csv, draft.csv, plays.csv, games.csv)
    a. Feature 13: as a user, I want to see the passing statistics of player *p*
    b. Feature 14: as a user, I want to see the passing statistics for year *y*
    c. Feature 15: as a user, I want to see the top *n* players for passing length *L* for year *y*
    d. Feature 16: as a user, I want to see the top *n* players for passing outcome (complete/incomplete/interception) for year *y*

Test Cases

1. Feature 1 Test Cases:
    a. Test Case 1: as a user in the combine performance page, I search for player *p* by inputting a player name into the search box
    Correct Output: the site displays a table with all combine records for player *p*
        i. Pick 2 players to test, players TBD (haven't looked at the data yet
    b. Test Case 2: test that combine performance displayed matches expected display
    Correct Output: page displayed on user selection matches specified page display
        i. Django has automated test case functionality that enables "pretend" client interaction with the site and a test harness that can verify if the content of an "expected" webpage matches the context you specify
            1. Ex. expected content on page for player *p* matches explicit content passed to test case

2. Feature 2 Test Cases:
    a. Test Case 1: as a user in the combine performance page, I search for combine performance for year *y* but inputting a year in the combine year search box
    Correct Output: the site displays a table with all combine records for year *y*
        i. Test on 2019 and 2018 (latest 2 years in the dataset)
    b. Test Case 2: test that the combine performance display page matches expected display exactly
    Correct Output: page displayed on user selection matches specified page display

3. Feature 3 Test Cases:
    a. Test Case 1: as a user in the combine performance page, I search for the top *10* records for combine measure *c* for combine year *y*
    Correct Output: the site displays a table with the correct records the user selected (e.g. the data values match what's in the data structure)
    b. Test Case 2: test that the combine performance display page matches expected display exactly
    Correct Output: page displayed on user selection matches specified page display

4. Feature 4 Test Cases:
    a. Test Case 1: as a user in the combine performance page, I search college *x* and records for the players who attended that college and performed at the combine are displayed
    Correct Output: the site displays a table with the correct records for the college the user selected (e.g. the data values displayed match what's in the data structure)
    b. Test Case 2: test that the combine performance display page matches expected display exactly
    Correct Output: page displayed on user selection matches specified page display

5. Feature 5 Test Cases:
    a. Test Case 1: as a user I want to see the total rushing performance of player *p*
    Correct Output: site display the total number of rushing yards for entered player p.

6. Feature 6 Test Cases:
    a. Test Case 1: as a user I want to see the total rushing yards for each team *x*
    Correct Output: site display the total rushing yards for team x

7. Feature 7 Test Cases:
    a. Test Case 1: as a user I want to see the top *n* players for rushing for team *x*
    Correct Output: site displays the top n rushers of that team for team x

8. Feature 8 Test Cases:
    a. Test Case 1: as a user I want to see the top rusher of all time *t*
    Correct Output: site displays the top rusher for all years in dataset

9. Feature 9 Test Cases:
    a. Test Case 1: as a user in the receiving page i want to enter a player name and view their total receiving yards.
    Correct Output: site displays total number of receiving yards for entered players.

10. Feature 10 Test Cases:
    a. Test Case 1: as a user in the receiving page i want to enter a team name and view total receiving yards for each team
    Correct Output: site displays total number of receiving yards for each team

11. Feature 11 Test Cases:
    a. Test Case 1: as a user in the receiving page I want a list of the top 10 players with the most receiving yards
    Correct Output: sites displays a list of top ten players with the most receiving yards

12. Feature 12 Test Cases:
    a. Test Case 1: as a user in the receiving page I want to see top 5 teams with the most receiving yards
    Correct Output: site displays a list of the top 5 teams with receiving yards

13. Feature 13 Test Cases:
    a. Test Case 1: as a user in the passing statistics page, I search for player *p* by inputting a player name into the search box
    Correct Output: the site displays a table with the passing statistics for player *p*

14. Feature 14 Test Cases:

a. Test Case 1: as a user in the passing statistics page, I search for passing statistics for year *y* by inputting a year in the passing year search box
   Correct Output: the site displays a table with the passing statistics for year *y*

15. Feature 15 Test Cases:
    a. Test Case 1: as a user in the passing statistics page, I can select to search top players, type in the number of players to display, and enter a passing length in a search box
       Correct Output: the site displays a table with the passing statistics for the selected number of top players with the selected passing length and year

16. Feature 16 Test Cases:
    a. Test Case 1: as a user in the passing statistics page, I can select to search top players, type in the number of players to display, and enter a passing outcome (complete/incomplete/interception) and a year in a search box
       Correct Output: the site displays a table with the passing statistics for the top selected number of players with the selected passing outcome and year

TODO List

Done list of last sprint (week of 10/5 - 10/11)

1. Set up Django web application architecture
    a. Top-level executable: nfl_site (name of our project)
    b. First application (Django terminology for module that renders to a web page): response
        i. Ford St. John designed application welcome page and client-server response functionality
        ii. Robert Arenas designed application welcome page and client-server response functionality
        iii. Eduardo Rocha designed application welcome page and client-server response functionality
        iv. Kyle Dean designed application welcome page and client-server response functionality
    c. Each of us practiced step b. to gain familiarity with the Django framework. We each produced a test branch in git, and randomly selected one of the branches to merge into our main project
        i. Happened to select Ford St. John's welcome page and client-server interaction example

ToDo task list for next sprint:

1. Backend program to convert combine csv to combine class data structure
2. Backend program to convert rushers csv to rusher class data structure
3. Backend program to convert passers csv to passers data structure
4. Backend program to convert receivers csv to receivers data structure
5. Create combine application to render combine web app views
    a. Link backend data structure to front-end views
    b. Display userforms to select values
    c. Render data table of data based on selections
6. Create rushers application to render rushers web app views
    a. Link backend data structure to front-end views
    b. Display userforms to select values
    c. Render data table of data based on selections
7. Create passers application to render passers web app views
    a. Link backend data structure to front-end views
    b. Display userforms to select values
    c. Render data table of data based on selections
8. Create receivers application to render receivers web app views
    a. Link backend data structure to front-end views
    b. Display userforms to select values
    c. Render data table of data based on selections