# Lab 4 - Team-42

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

---

**Features (user stories) to Implement in Next Sprint:**

1. Combine site:
    a. Feature 1: As a user I want to select the top N performers for a selected combine event (40-yard dash, bench press, etc.) for a selected position group (WR, RB, etc.)
    b. Feature 2: As a user I want to be able to export the data on my screen (e.g. the filtered query results that are returned

2. Passing site:
    a. Feature 3: as a user I want to select the top *n* players with the passing length *L,* passing outcome (complete/incomplete/interception), and year *y*
    b. Feature 4: as a user, I want to export searched passing data

3. Receiving site:
    a. Feature 5: As a user I want to export the receiving data I have searched
    b. Feature 6: As a user I want to see the top n receiving yard players

4. Rushers site:
    a. Feature 7: As a user I want to search through who were the top 5-10 rushers in a certain year, certain team, all time. I want to see it displayed in a nice column data format with maybe a picture of some sort.
    b. Feature 8: As a user I want to export each of these features in rushers into a .csv so that I can use later.

5. “Player management site” - manage the NFL player “database”
    a. Feature 9: as a user I want to be able to add a player to the NFL database
    b. Feature 10: as a user I want to be able to edit a player in the NFL database
    c. Feature 11: as a user I want to be able to delete a player from the NFL database

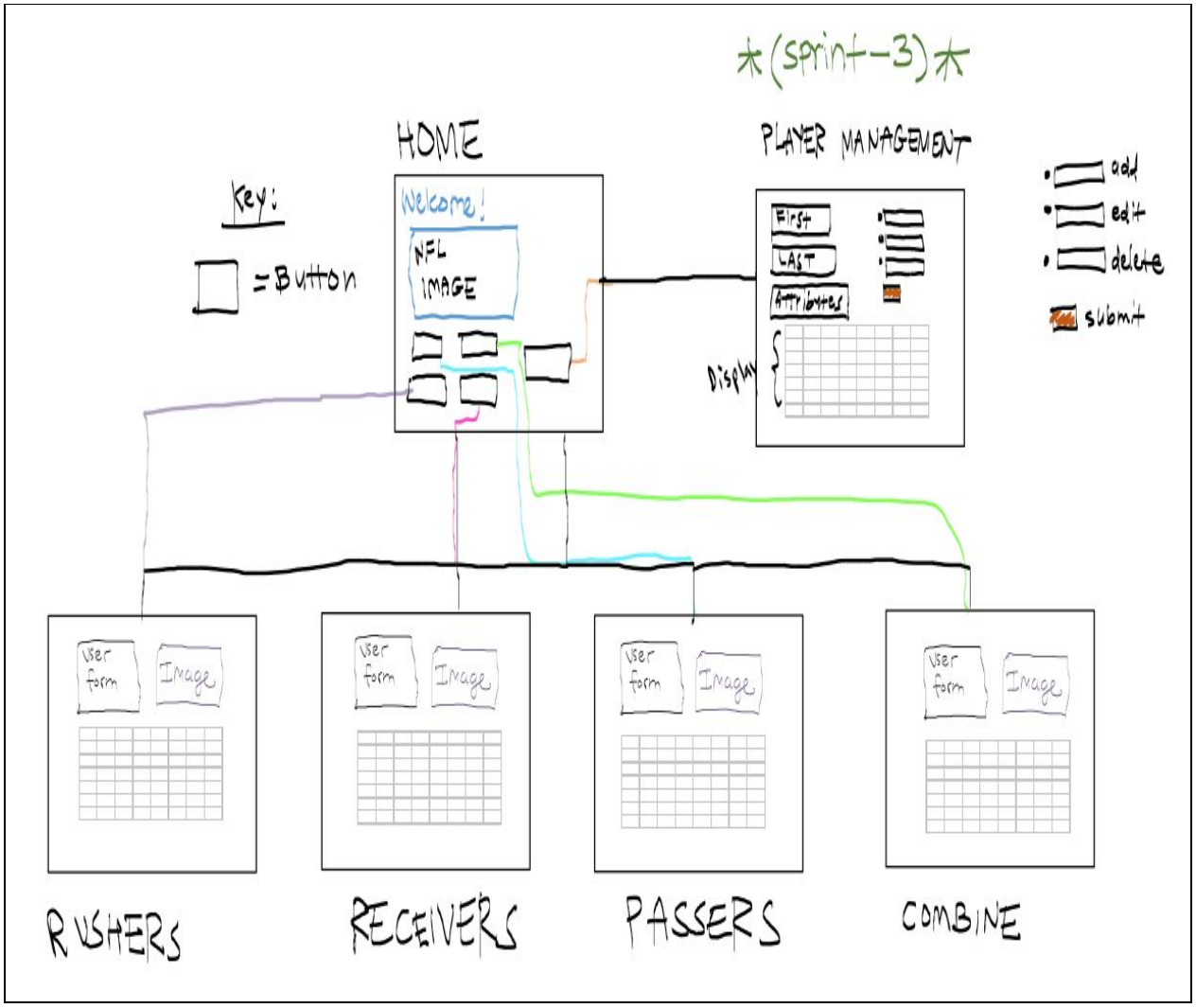


*Image also included in *Sprint-3* directory*

**Test Cases**

1. Feature 1 test cases
    a. Test Case 1: as a user I search for the top 5 40-yard dash times for WR in 2019 combine
    Correct output: Top 5 WR 40-yard dash times are displayed from 2019 combine
    b. Test Case 2: as a user I search for the top 5 bench press counts for RB in 2019 combine
    Correct output: Top 5 RB bench press amounts are displayed from the 2019 combine
2. Feature 2 Test Cases:
    a. Test Case 1:as a user I want to be able to export the current view to a csv
    Correct Output: a CSV is made available to the user and can be downloaded with

the correct data based on the current view

3. Feature 3 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, enter a passing length, passing outcome, and passing year in a form.
    Correct Output: the site displays a table with the passing statistics for the selected number of top players with the selected passing length, passing outcome, and year

4. Feature 4 test cases
    a. Test Case 1: as a user in the passing statistics page, I can click a button to export the passing statistics data that I searched for.
    Correct Output: export button is clicked a download of the csv file initiates

5. Feature 5 test cases:
    a. Test Case 1: as a user I look up the top 10 players by receiving yards
        i. Correct output: list of the top 10 players is displayed on the page.

6. Feature 6 test cases:
    a. Test Case1: as a user i would like to export the data i have searched
        i. Correct output: user clicks export button and is able to download csv file of data.

7. Feature 7 test cases
    a. Test Case 1: As a user I search the top 5 rushers for the Los Angeles Chargers of all Time.
    Correct Output: The site displays in a clean column format, the top 5 Los Angeles Chargers rushers.

8. Feature 8 test cases
    a. Test Case 1: As a user I want to save the results of the query in a .csv file outside of the website.
    Correct Output: The site is able to export the data for the rushers site to a .csv file.

9. Feature 9 Test Cases:
    a. Test Case 1: As a User I press the Add A Player Option and input a player's attributes.
    Correct Output: The New Player is created with their attributes and sent to the players.csv file.
10. Feature 10 Test Cases:

a. Test Case 1: As a User I press the Edit Option and edit a player's attributes.
   Correct Output: The Player's attributes are updated and sent to a .csv

11. Feature 11 Test Cases:
    a. Test Case 1: as a user I press the delete button for a selected player, and that player is deleted from the "database"
       Correct Output: the selected player is no longer present in the players.csv dataset (e.g. has been effectively deleted)

**TODO List**

Done list of last sprint (week of 10/12 - 10/18)

1. NFL Combine Site
    a. Create functions to amend data in certain fields for cleaner presentation
       [finished by Ford St. John, verified by team]
    b. Read in correct .csv file from local directory (combine.csv) for display on site
       [finished by Ford St. John, verified by team]
    c. Create userform that enables client to enter player first name, player last name, combine year, combine event, and player position group on the GUI
       [finished by Ford St. John, verified by team]
    d. Script to retrieve userform data entered by client and use entries to filter combine data to display the correct results back to the client
       [finished by Ford St. John, verified by team]
    e. HTML edits to dynamically render data table of filtered combine data based on user selections
       [finished by Ford St. John, verified by team]
    f. Checked in combine application, updated necessary URL mappings, committed code changes to github repo
       [finished by Ford St. John, verified by team]

2. Passing Statistics Site
    a. Created the passing statistics app on the site allowing the user to search a player's passing statistics
       [finished by Kyle Dean, verified by team]
    b. Created the form python file to allow user to enter search fields on the passing statistics site
       [finished by Kyle Dean, verified by team]
    c. Read the correct .csv files from local directory (passer.csv and players.csv) to query data based on the user's search fields
       [finished by Kyle Dean, verified by team]
    d. Updated the HTML file that is loaded for the passing statistics site
       [finished by Kyle Dean, verified by team]

e. HTML automatically updates to display a table once the user enters search fields and clicks the 'Submit' button
[finished by Kyle Dean, verified by team]
f. Properly created separate branch for passing statistics site and merged to main once site was finished
[finished by Kyle Dean, verified by team]

3. Receiving site
    a. Create function to retrieve player id from players.csv
    [finished by Robert Arenas]
    b. Create function to calculate total receiving yards of a player using receiver.csv
    [finished by Robert Arenas]
    c. Create the receiving app of the site allowing the user to search and see a player's receiving data.
    [finished by Robert Arenas]

4. Rushers Site
    a. Created the rushing statistics app on the site allowing the user to search a player's rushing statistics
    [finished by Eduardo Rocha]
    b. Created the forms.py python file to allow user to enter data to search for on the passing statistics site
    [finished by Eduardo Rocha]
    c. Created the user interface at the home page to go to different pages(apps) and return back to home from those pages.
    [finished by Eduardo Rocha]
    d. Created functions in views.py to use for extracting data from the .csv's I'm reading in.
    [finished by Eduardo Rocha]

<u>ToDo task list for next sprint:</u>

1. Create "libraries.py" file in nfl_site directory to manage custom definitions and scripts to retrieve data
2. Create backend API get_data(data_source) that reads desired csv from Google Drive location
    a. Will also function as a backup/import tool, as datasets retrieved from this Google Drive location are "static" from the Kaggle website and will never change (e.g. will never reflect record inserts/changes/deletes that a user might make)
3. Backend function to add a player record
    a. Validate player doesn't already exist, need to generate new unique player ID, confirm new player ID does not already exist
4. Backend function to change (edit) a player record
    a. Validate that the player record already exists and therefore can be changed
5. Backend function to delete a plate record

    a. Validate that the player exists before being able to delete
6. Backend function to write user inserts/changes/deletions to csv
7. Backend function to save dataset amendments from the client to a separate folder on Google Drive
    a. The saved CSV should be <dataset name>_<user edits>_<date of changes>.csv
8. Create frontend “player management site”
9. Create userform to enter player information for add/change/delete
10. Create appropriate buttons that enable add/change/delete functionality from client
11. Write test cases testing client player add functionality
12. Write test cases testing client player change functionality
13. Write test cases testing client player delete functionality
14. Write test case confirming correct export of client changes to csv in Google Drive
15. Combine site - finish feature allowing client to select top N for specific combine event, combine year, and player position
16. Combine site - finish feature allowing client to export data in the view as a CSV
17. Receiving site - finish feature to allow  searching of receiving data for teams.
18. Receiving site - add css to page