

Lab 4 (Oct 29) #Team1

Sprint-4

Retrospective meeting: any updates since last meeting? any issues?

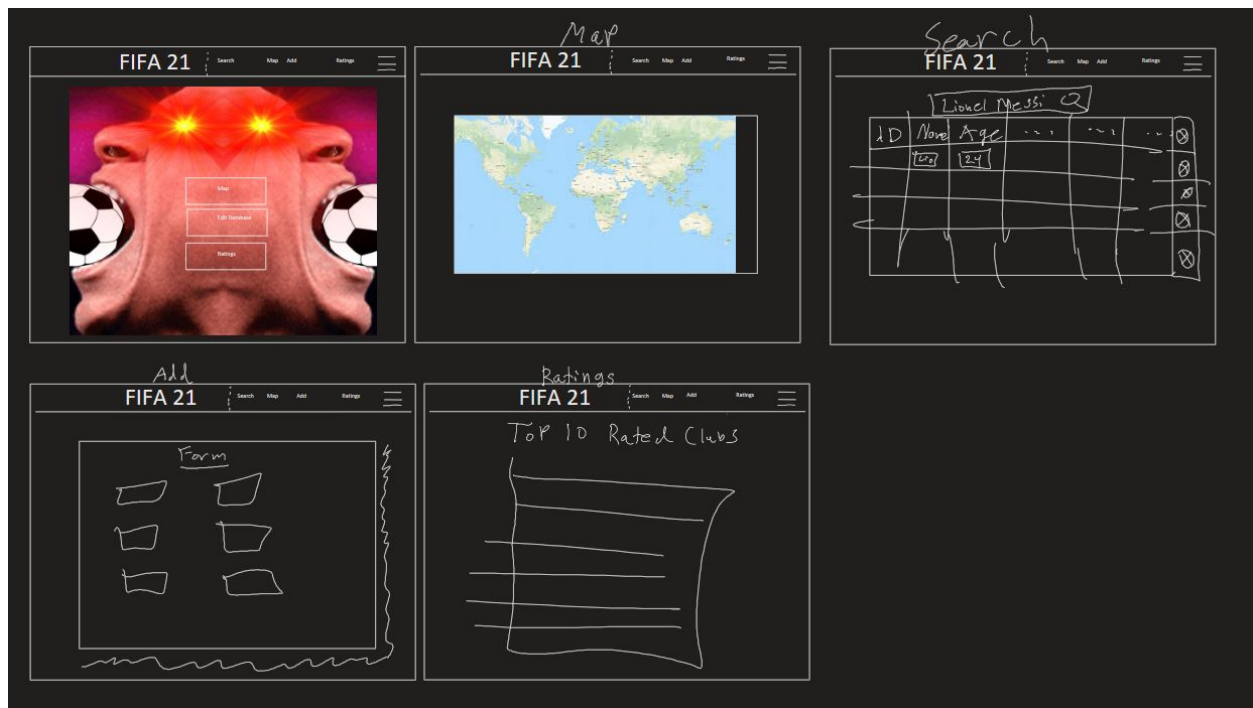
Planning meeting:

- what analytics feature(s) to implement next week?
- breakdown the goals into the actionable tasks
- define task completeness criteria (test cases if needed)

Feature:

- Analytics:
 - Top 100 rated players
 - Most common age
 - Top 100 players with best hits

GUI:






Taskboard:

Done list of last sprint 3:






- Created search function for each attribute in database i.e name, player id, nationality, team, etc (Mustafa Many)
- Function takes in attribute to search for and the search entry ex: search('name', 'Lionel Messi')
- Function returns dataframe (which can be converted to html)
- Acceptance Criteria: function returns row of data in the form of a dataframe
- Created Test cases for search functions (Mustafa Many)
- Full name of existing player
- First name of player
- Last name of existing player
- Number entry (breaks)
- Single character
- Empty string (breaks)
- Created python database class to manage and manipulate database (Abraham/Evan)
- Class includes python pandas import
- Acceptance Criteria: pandas is imported without compiler errors
- Class includes reference to newly created file to edit (so we can avoid changing fifa csv file)
- Acceptance Criteria: file variable exists and can be called upon
- Class includes search, add, modify, and delete functions
- Acceptance Criteria: functions are callable from other classes/python files
- Class stores all current data in mysite/fifa/fifaCS180.txt
- Acceptance Criteria: text file is appropriately named and accessible from other classes/python files
- Created database function to modify existing entry (Evan/Mustafa/Abraham)
- Function takes in player id argument along with all other elements
- All soccer player elements changed or unchanged should be sent to database
- Acceptance Criteria: function changes existing row according to arguments on same line
- Created database function to delete existing entry (Evan/Abraham)
- Function will take in player id as argument and remove row accordingly
- Acceptance Criteria: fifaCS180.txt no longer contains row with passed in player id
- Created database function to add new db entry (Evan)
- Acceptance Criteria: fifaCS180.txt adds new row after function call
- Create frontend form for search (Calvin/Hoda)
- Acceptance criteria: looks pretty, clicking on the button receives data from database, confirms update in backend, displays search result.
- Create frontend form for add (Calvin/Hoda)
- Acceptance criteria: looks pretty, clicking on the button receives data from database, confirms update in backend.
- Create render functions to communicate with frontend(Mustafa)
- Acceptance Criteria: Intended data is sent and received to and from forms
- Create dropdown (Calvin/Hoda)
- Acceptance criteria: looks pretty, ll attributes available and selecting each attribute updates form label
- Form css (Calvin/Hoda)
- Acceptance criteria: looks pretty, perfectly centered, shadowing to create elevation
- Fix button spacing(Calvin,Hoda)

- Acceptance criteria: button text is perfectly centered
- Create frontend form for modify(Calvin,Hoda)
 - Acceptance criteria: looks pretty, clicking on the button receives data from database, confirms update in backend.

Todo list for next sprint:

-  Mobile view/desktop view (Hoda)
 - Acceptance criteria: Website looks pretty when the screen size is shrunk to less than 800px and no functionality of the website is hidden
-  Create shared nav bar view (Hoda)
 - Acceptance criteria: nav bar works on all pages of the website, button links take you to each page
-  Create frontend form for ratings (Hoda)
 - Acceptance criteria: looks pretty, clicking on the button receives data from database, confirms update in backend.

Analytics

- 1)  Create MostCommonAge analytic function (Sprint 4) (Calvin, Mustafa)
 - Acceptance criteria: Return list of players with the most common age.
- 2)  Create TopandLowestRated analytic function (sprint 4) (Abraham, Evan)
 - Acceptance criteria: Return list of players who are highest and lowest rate (10 each)
- 3)  Create average rating calculation analytics function (sprint 4) (Abrahm, Evan, Calvin, Mustafa)
 - Acceptance Criteria: Returns list of teams with rating values for each team
- 4) Create map and populate with player names and nationality (Todo)
 - Acceptance Criteria: Map entity displays with all players
- 5) Create MostPopularPlayer analytic function (Todo)
 - Acceptance Criteria: Return list of top 10 most popular players.
- 6) Create MostPopularPosition analytic function (Todo)
 - Acceptance Criteria: Return top 3 most popular position.
-
-  Create database function to add new db entry
 - Acceptance Criteria: fifaCS180.txt adds new row after function call
-  Create database function to delete existing entry
 - Function will take in player id as argument and remove row accordingly
 - Acceptance Criteria: fifaCS180.txt no longer contains row with passed in player id

Test Cases:

- **Feature 1 (Insert):** As a user I want to be able to add a new player to the Database with the information needed (name, nationality, team, player id).
 - **Test case 1:** User enters all the necessary information of a player into the website
 - **Correct output:** Website successfully takes in new information and updates the database with the player
 - **Test case 2:** User enters incomplete information about a player (eg. missing age, rating, etc.)
 - **Correct output:** Website throws an error requiring the user to enter all fields with information
 - **Test case 3:** User inputs a non alphanumeric character in any of the fields
 - **Correct output:** An error is thrown to notify the user that an illegal character has been inputted for information

- **Feature 2 (modify):** User want to update the players' information and keep the data up to date.(team, rating, age, position)
 - **Test case 1:** Users want to change the age for one of the players in the data. Users will find the players who need to be changed. Then,click the modify button and select the age. The textbox will show up and users are required to enter the correct age in the textbox and click the submit button.
 - **Correct output:** Server will be updated and show the age that users changed.
 - **Test case 2:** Users want to change the position for one of the players in the data. Users will find the players who need to be changed. Then,click the modify button and select the position. The dropdown menu will show up and users are required to select the position from the menu and click the submit button.
 - **Correct output:** Server will be updated and show the position that users changed.
 - **Test case 3:** Users want to change the rating for one of the players in the data. Users will find the players who need to be changed. Then,click the modify button and select the age. The textbox will show up and users are required to enter the correct rating in the textbox and click the submit button.
 - **Correct output:** Server will be updated and show the rating that users changed.
 - **Correct output:** Server will bring to user and show all the player list from the Club that users choose and the bottom of the list will show the average rating.

- **Feature 3 (Search):** As a user I want to be able to search a player(s) from the Database with the information needed (name, nationality, team, age).
 - **Test case 1:** User enters all the necessary information of a player into the website
 - **Correct output:** Website successfully takes the input and searches the qualified players from the database and displays them.
 - **Test case 2:** User enters wrong information about a player (eg. invalid age, rating, etc.)
 - **Correct output:** Website throws an error requiring the user to enter field with valid information
 - **Test case 3:** User leaves the field empty.
 - **Correct output:** An error is thrown to notify the user are required to fill the field

- **Analytic 1 (World map):** We want to display a world map that indicates where players are from each country
 - **Test case 1:** Display all players and country of origin
 - **Correct output:** World map is populated with 'points or dots' that shows each country and how many players are from there. Example Portugal has 14 players, so there will be 14 points on portugal.
 - **Test case 2:** Country with no players eg (Monaco)
 - **Correct output:** Monaco map will be displayed with no points as fifa 21 has not included any players in their database.

- **Analytic 2 (Best Hits):** User want to checkout which players have the best hits
 - **Test case:** User wants to see which player is most popular
 - **Correct output:** Returns list of players with most hits.

- **Analytic 3:** Users want to know the average rating for Club A. Users will type the Club name in the text box. Then,click the average button.**Feature 3 (World map):** We want to display a world map that indicates where players are from each country
 - **Test case 1:** Display all players and country of origin
 - **Correct output:** World map is populated with 'points or dots' that shows each country and how many players are from there. Example Portugal has 14 players, so there will be 14 points on portugal.
 - **Test case 2:** Country with no players eg (Monaco)
 - **Correct output:** Monaco map will be displayed with no points as fifa 21 has not included any players in their database.

Feature 4 (Average): User want to checkout the average rating for each club

Test case 1: Users want to know the average rating for Club A. Users will type the Club name in the text box. Then,click the average button.

- **Feature (Delete):** User want to delete a player from Database
 - **Test Case1:** Player enters 'Lionel Messi' to delete him from database
 - **Correct Output:** Player 'Lionel Messi' and all his attributes are deleted from Database
 - **Test Case 2:** User enters incorrect information filled with non alphanumeric characters
 - **Correct output:** The page throws an error that tells the user of incorrect input
 - **Test Case 3:** User enters a player name that does not exist (eg 'John Cena')
 - **Correct output:** 'Player does not exist in database'
- **Analytic 4 (TopAndLowestRated):** User want to checkout the top and lowest rated player
 - **Test case 1:** User will type 'top10' and it will return top 10 players
 - **Test case 2:** User will type 'low10' and it will return lowest rated 10 players