MEWS: Eric Sohel, Wilson Mach, Shinji Kusakabe, Maya Mori
SoftDev
P01: ArRESTed Development
Time Spent: 1.5 Hour
2022-12-05

Target Ship Date: 2022-12-23
Idea: Character higher lower game. Use superheroAPI and others to create a higher lower game with superheroes as the subjects. Also a pokemon quiz game, with questions on pokemon names and types.

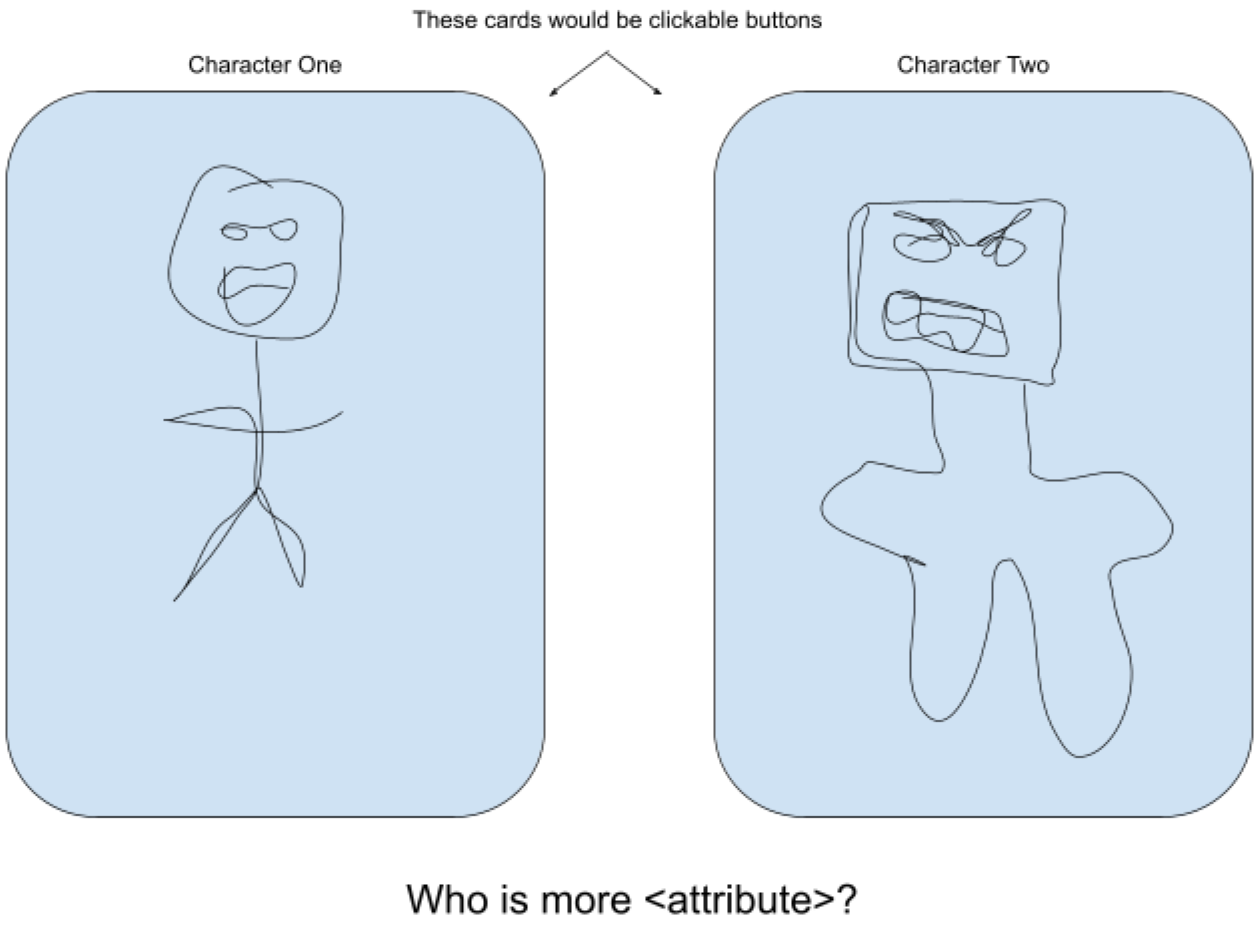
List of APIs (update with links):

- Superhero API (get superhero stats for comparisons) - https://www.superheroapi.com/index.html
- Yes/No API - https://yesno.wtf/api
    - Use "force" query to get a GIF for yes, no and maybe
- Pokemon API - https://pokeapi.co/

Notes:

- For the character higher lower game: players would be shown 2(+) characters and a category, such as intelligence, strength, speed, durability, power, combat and the player would try to pick which character has the higher metrics for that category. The values in these categories would be given in a number that we would compare between the different characters.
- For the pokemon game: players are shown a picture of a pokemon and guess either its name or its type(s).
- We will track how many questions the user got right out of 10 rounds.
- Alternatively, if we have the time, keep track of how many questions they got right in a row (or use a life system where you lose a life if you get it wrong) as the score and keep going until the player loses.
- We can use the Yes/No API to tell the users if they got the question right with a gif to make it more interesting
- Can use the database to record the player's high score and if we have time, maybe let them customize the settings that they want to play the game with.
- Leaderboard to show top 10 players

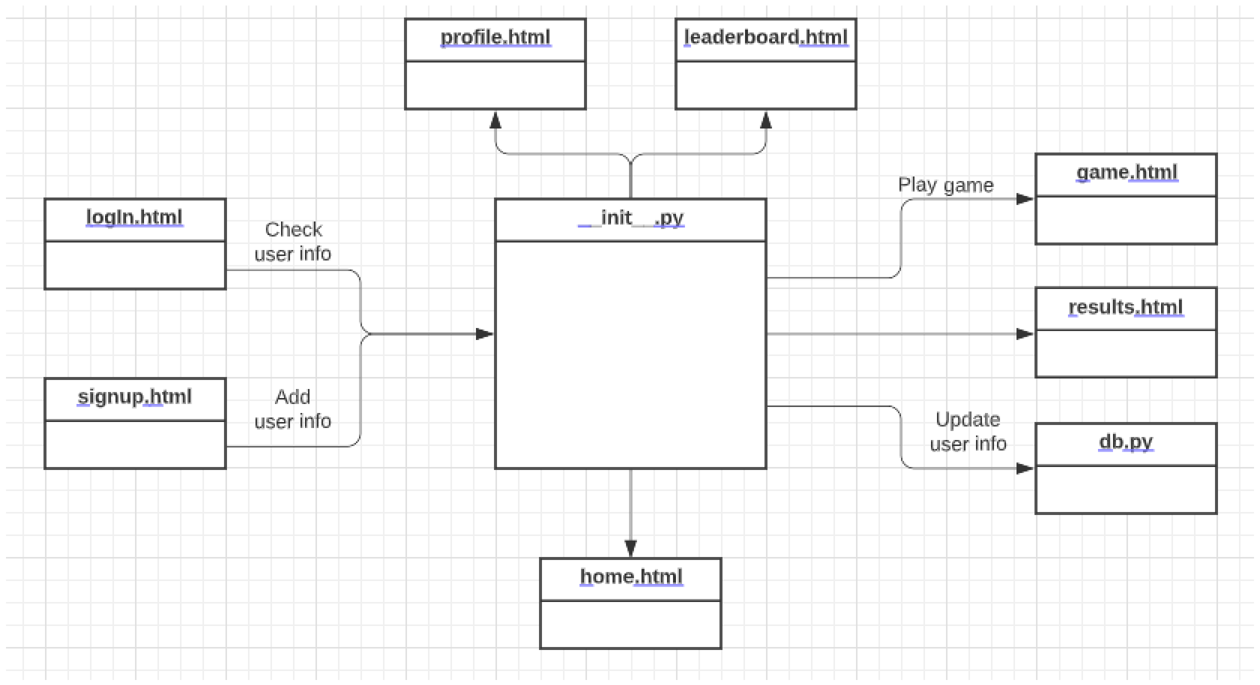
Example of a round:



Implementation:
-For each round, the program would select 2(+) random IDs from the character IDs in the superhero API (or random ones from a list of those that we choose to use) and randomly pick a metric to compare the characters.
-We would then use the information we have on those metrics given by the superhero API for those character IDs and mark the correct one (might need either a tie as an option or do some coding to make it reroll the characters/metrics if there is a tie).
-For the pokemon game, we will use information from the pokeapi and ask them questions about either a single pokemon's type(s), or their name.
-We would render choices on our website and the player would select whatever they thought was the answer and will be told if it's correct or not. If they are correct, increment the score by 1 and

generate another question. Once the player loses, take them to the result screen where they can see how they did and choose to play again or go to the homepage/their profile.
-Scores would be recorded and stored in the database and there could be a leaderboard page to see the best scores/players.
-Profile page could show some of the player's stats. These include: account name, profile picture, high score, accuracy, number of games played and average score per game.
Frontend Framework: Bootstrap
Reason: While Foundation is focused on the visual and customizability aspects of a project, Bootstrap is primarily focused on functionality and not so much in the way the final project will look. For the purposes of this project, we are more concerned with the functionality that is offered by bootstrap rather than the aesthetics offered by foundation.

Component Map:



Data Base(There would be a high_score, total_score and games_played column for each game type that we have, currently only pokemon and character quiz):

username	password	<game>_high_score	<game>_total_score	<game>_games_played
Each user will be able to create their own account!	Their progress/ scores will save!	Other users will be able to compete for the highest score!	We can use this along with games_played to compute average scores and accuracy.	

Python Files:

- db.py - deals with the database
- __init__.py - run Flask app
- game.py - accesses data from the Superhero API, calls function in db.py to update high_score and deals with the logic of the game (keeps track of current score, rounds, answer)
- api.py - gets data from APIs

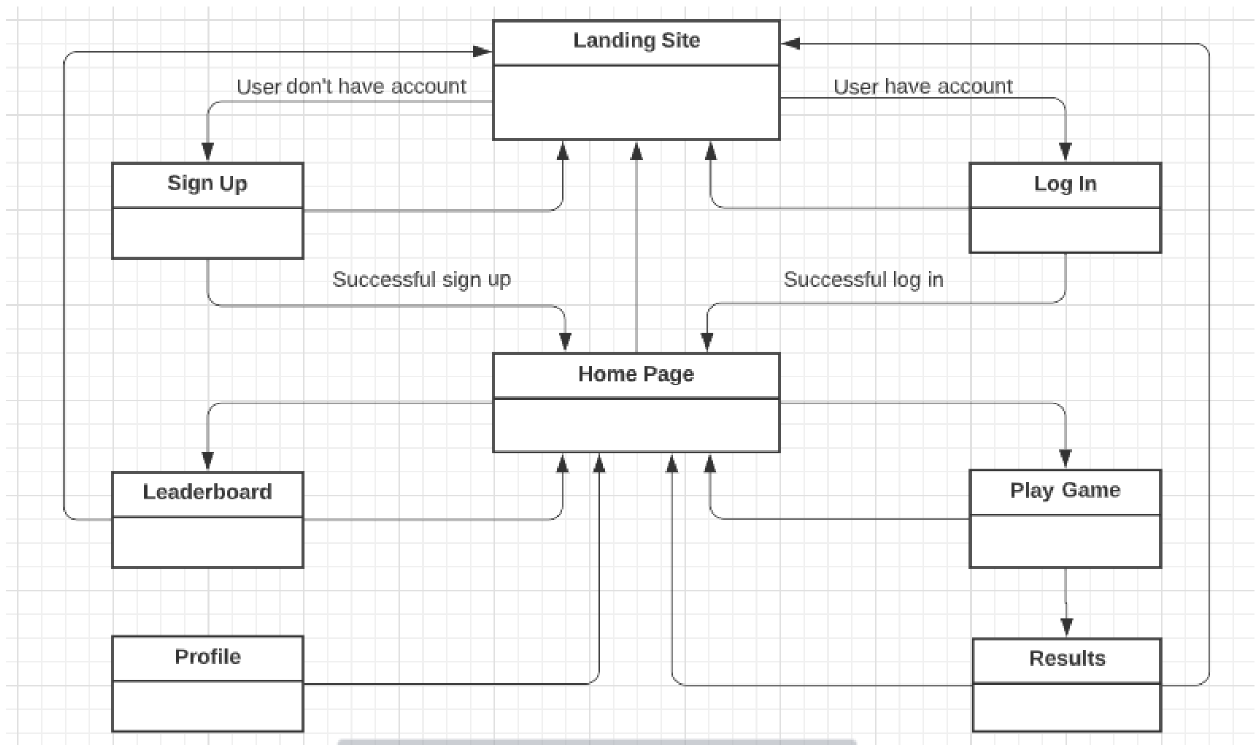
HTML Files (templates):

- home.html - some description of the site
- signup.html - make username and password (confirm password too)
- login.html - input username and password
- profile.html - displays the user's username, profile picture and high score
- leaderboard.html - displays the top 10 players in terms of high score
- game.html - displays 2 characters per round for the user to guess from
- results.html - shows score and high score

CSS File:

- style.css

Site Map:



Roles:
Databases - Maya
Game - Eric/Shinji
Front End- Eric/Shinji
Flask- Wilson/Maya