Hero Wars by Teem_sqrt4469
P01: ArRESTed Development
Yuhang Pan (PM), Matthew Ciu, Michelle Chen, Thomas Mackey
2025-12-22
TARGET SHIP DATE: 2025-12-22

Project Description: Our website is a local two-player Pokemon showdown game, where we use anime characters and superheroes instead of Pokemons using the Jikan API and the Superheroes API. For the in-game stats, we will get data from those API's and convert those info into in-game stats. Those stats will be assigned to random Pokemon moves and be given to the superheroes and anime characters.

Game Balancing:

Damage = $atk \times (1 - (\frac{1}{10} log_{10} def + 0.002 def))$

Move Damage = $10 \times damage^{\frac{log_{10}(10-pp)}{2.5}}$

Superhero:

- hp = durability
- atk = power
- speed = speed
- def = strength

Anime Character:

- hp = $\frac{1}{2} \times \sqrt{(\#\ of\ favorites)}$
- atk = $rand[5, 10] \times (\#\ of\ animes\ they're\ in)$
- speed = $rand[5, 10] \times (\#\ of\ mangas\ they're\ in)$
- def = $rand[5, 10] \times (\#\ of\ voice\ actors\ have\ played\ them)$

Game Currency: Rizu Coins (rizz in japanese)

Program Components:

A. Flask App (Python)
    - **__init__.py** creates flask app, make routes, and run app
        - **/register** adds entered username & password into users table, checks that username has to be unique, stores username in session then redirects to /home
        - **/login** checks if the entered username & password is stored in users table, if so, stores username in sessions and redirects to /home
        - **/home** extracts the user's info (name, win/loss ratio, amount of rizu coins)
        - **/loadingpage** redirects to menu
        - **/menu**

        - Calls functions that make API calls to fetch characters (superheroes and anime characters)
        - Randomly selects a list of characters to battle (maybe an option to reselect)
    - **/game**
        - Stores match data in flask session
        - Allow players to interact with battle (attacks, etc.)
        - Can use evilinsults API to taunt the other player
        - Handles turn based logic
    - **/gameover**
        - Displays winner
        - Stores win/loss and rizu coins gained in user table
- **build_db.py** creates the user table
- **apis.py** handles all API calls & fetch and convert data, convert certain stats to in-game stats
- **game.py & battle.py** includes functions like forming a random team, attack function to calculate damage, and checks for alive characters to switch to when one dies

B. Templates (HTML templates for each page w/ Jinja2)
- **login.html** & **register.html** (form input boxes, handles authentication)
- **home.html** (homepage showing profile info: name, profile picture, #wins/losses, amount of rizu coins you have)
- **loadingpage.html** displays a cat gif while the menu page loads, redirects to **error.html** if fails
- **menu.html** (pregame phase that randomly generates a team for both players, one superhero and one anime characters; cards display characters on the team with a button to reroll for a different character; a start button)
- **game.html** (the battle arena: both players with their respective active character will be displayed separated by a battle log; there will be buttons beside each character for their moves, to switch character, and to trash talk)
- **gameover.html** (page that shows that the game is over and displays the winner of the showdown)

C. Database (SQLite3)
- User table (stores user data when they register and updates it when they change profile picture or finish a game)

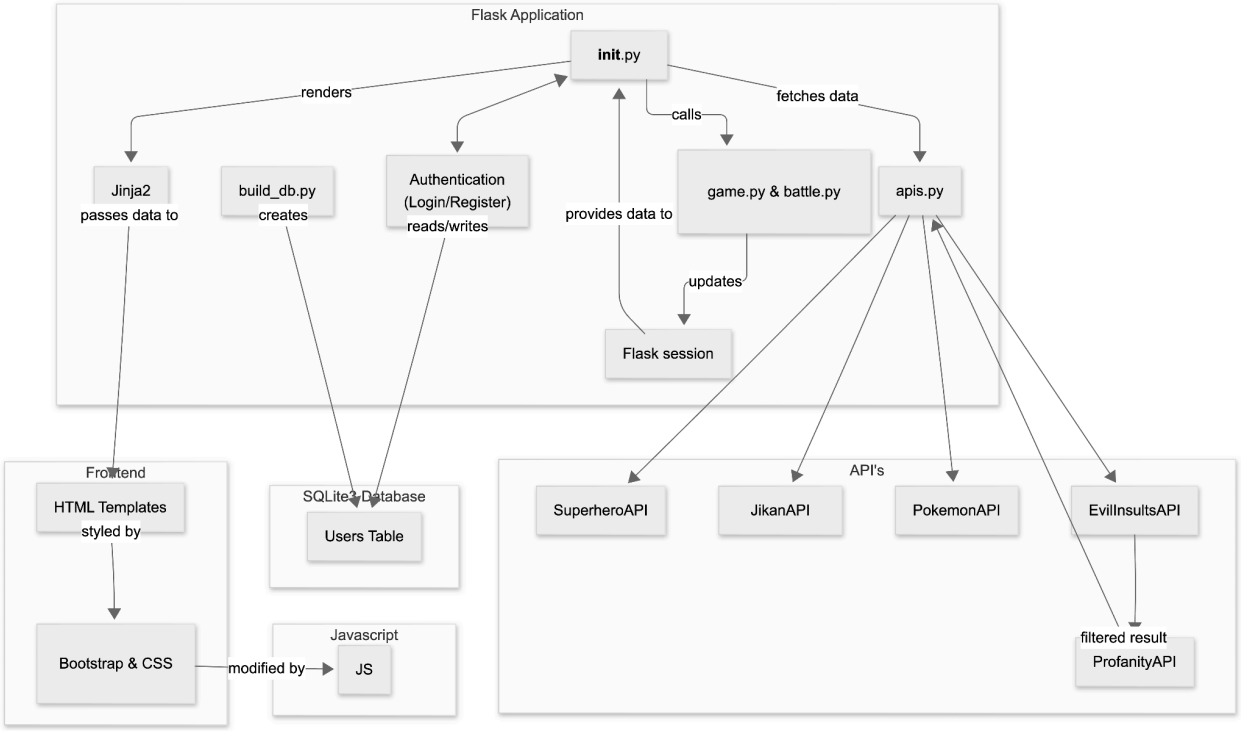
D. API's
- SuperheroAPI - https://akabab.github.io/superhero-api/api/ (no quotas)
- JikanAPI - https://jikan.moe/?ref=public-apis (Daily Unlimited, Per Minute 60 requests, Per Second 3 requests)
- PokeAPI - https://emiliebarnard.github.io/pokemon-in-python/#moves (no quotas)

- EvilInsultsAPI - https://evilinsult.com/generate_insult.php?lang=en&type=json (no quotas)
- PurgoMalum API - https://www.purgomalum.com/ to filter EvilInsultsAPI (no quotas)

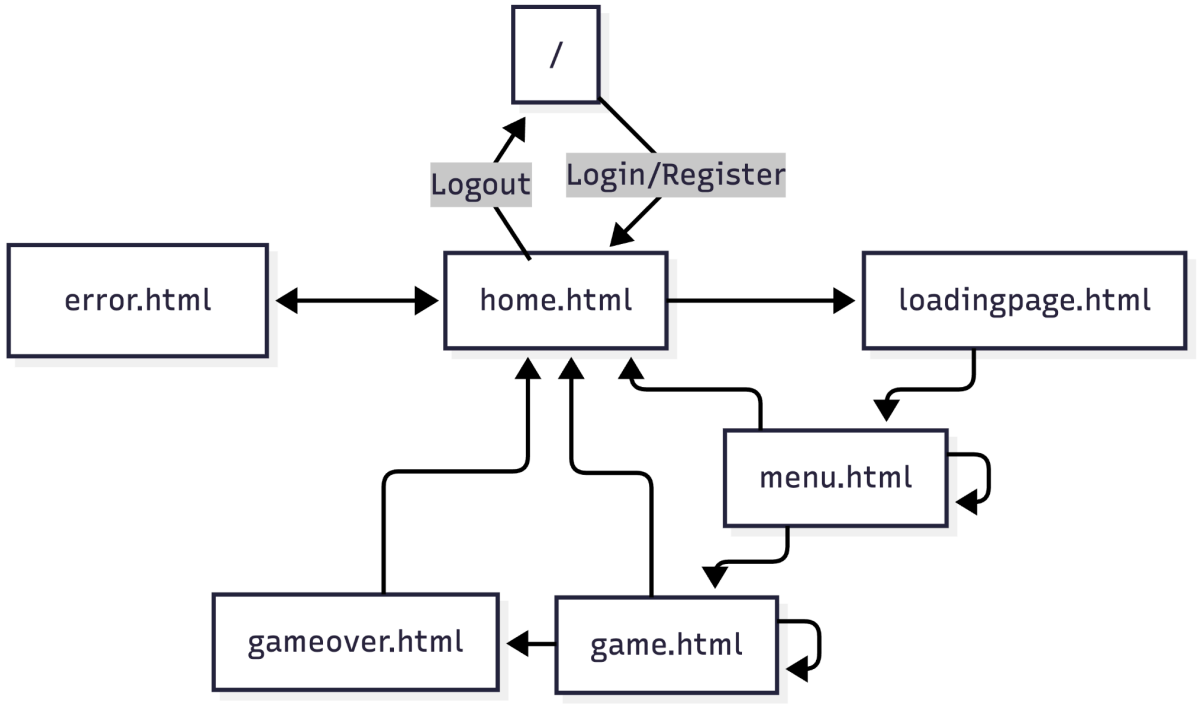
Front-end Framework (Bootstrap):

- Good documentation since the documentation for Foundation isn't great; Tailwind is confusing with all the utility classes in the HTML
- Bootstrap grid system is easy to work with which we can use to display the abilities of our characters; can also be used to divide the battle page to left and right for the two players
- Card container to display randomly selected characters

Component Map:



Site Map:



Database Organization:

USERS

TEXT	name	PK	Username is unique
TEXT	password		Used for authentication
INTEGER	wins		# of wins
INTEGER	losses		# of losses
INTEGER	rizu_coin		Amount of rizu coins
TEXT	profile_pic		Random Pokemon from PokemonAPI

Tasks:

- Yuhang Pan
  - apis.py (call API and fetch data + convert stats into in-game stats)
  - build_db.py (sets up database)
- Matthew Ciu
  - game.py (game logic with turn-based gameplay, store game state in session)
- Michelle Chen
  - Bootstrap styling on HTML Templates + templates themselves
- Thomas Mackey
  - game.py (game logic with turn-based gameplay, store game state in session)