Period 9's Team KungFuTea

Kyle Lin, **F**abiha Ahmed, **T**ina Chen

KaFooT

Information:

General Description:

KaFooT is a educational game website that derives its inspiration from Kahoot. The user can select the trivia genre from a list of categories that are pre-defined or user-defined. The user gains points based on whether or not the correct answer is selected within a set time period; there is no penalty for answering wrong. The user has the option of reading more about the current question after the answer has been selected if desired.

Game:

When presented with a question, the user has 3 seconds where the answer choices are disabled to read the questions and the answer choices. The answer choices are presented in a random order. After 3 seconds have passed, the answer choices are available to click on. The user gains points based on how quickly they answer the question correctly. There is no penalty for choosing a wrong answer. Once any of the four answer choices are selected, the buttons are then disabled until the selection-phase time period of 7 seconds has passed.
Throughout this process, there are 15-30 bot "accounts" that play along with you. These select the correct answer 90% of the time. The time that they take to select the answer is randomly generated between 0-7 seconds after the selection-phase is opened. Based on the user-selected bot difficulty, the time the bots take to answer is weighted towards 0-2.99 seconds, 3-5.49 seconds, or 5.50-7 seconds for hard, normal, and easy respectively.
After 10 questions have been presented and answered, the game will end and a leaderboard is shown. The user's placing, points, and question category are recorded in the leaderboard.

Account:
The user can create an account with a unique username and a password of choice. The password will be hashed to add some security (Attempt).
When a user is not signed up, an error is displayed.
When a user has the wrong password, an error is displayed.
When a user registers, their username and hashed password are added to the user database and a userrecord-<USERNAME> table is created (which is why the usernames must be unique).
At the user account page, the user can change their display name and reset their stats.

- Done using a database

User Record:
A user record can be navigated to where the user can check up on their records and categories of those records.
A table is displayed along with the category of the entry, their rank for that entry, and the score they got for that entry.
A user can delete any single entry of their userrecord if desired. (Will have a confirmation box).

- Done using a database

Own Trivia Category:
The user can create their own trivia category if wanted. They will be required to input a category name, 10 questions, and 4 possible "answers" for each question.
For the "answer" inputs, the first one should always be the correct answer (although it the order will be randomly displayed to the user during the actual game phase).

- Done using javascript? (json formatted data).

Databases:

Table	**Values**
users	username TEXT PRIMARY KEY, password TEXT NOT NULL
userstats-<USER>	category TEXT PRIMARY KEY, rank INTEGER, score INTEGER

- username: a user's unique username
- password: a user's password. (encrypted)
- category: category of that "run"
- rank: rank achieved in that "run"
- score: score achieved in that "run"

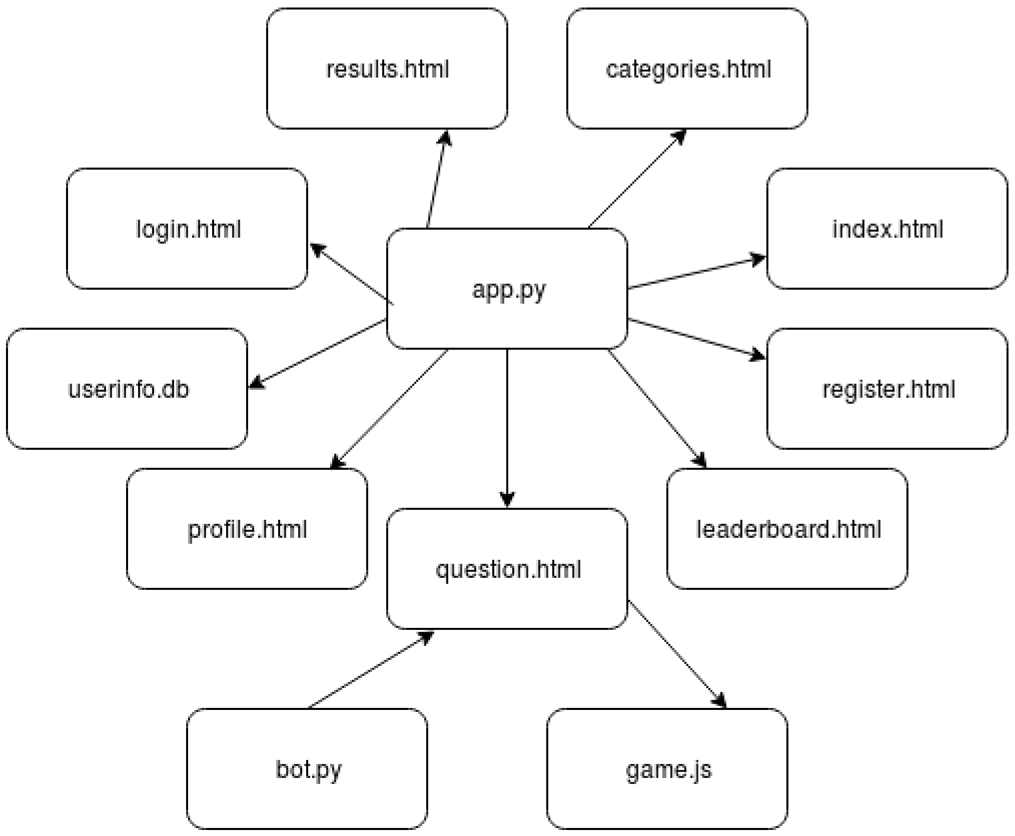
Maps and Innards:

Components:

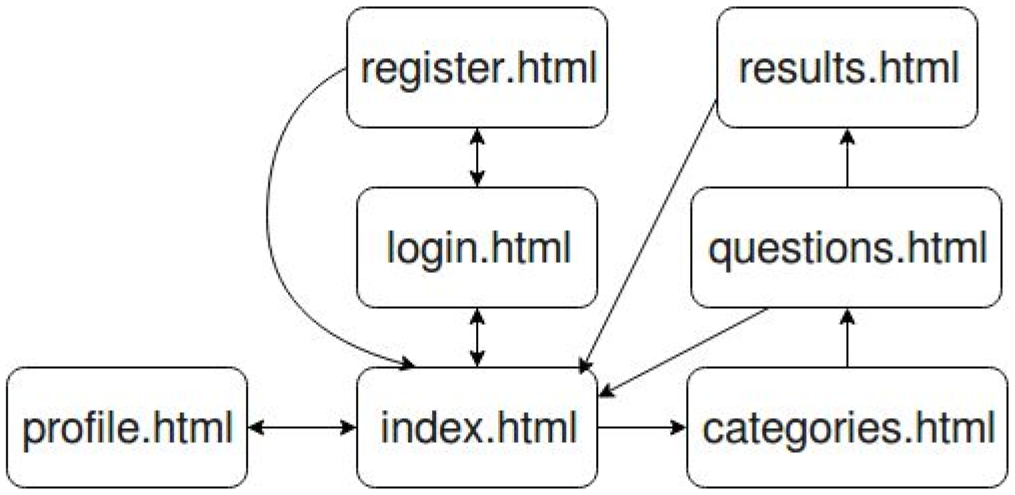
- Open Trivia DB (API): Pre-defined set of questions / answers
- Wikimedia (API): Learn more about the question
- Sqlite Database: Used to store user information
- Templates:
    - categories.html
        - List of categories of questions you can choose from
    - frame.html
        - Frame for all pages. Contains navbar, footer, and generic css / js.
    - index.html
        - Homepage. Description of website.
    - login.html
        - Login page.
    - profile.html
        - View the user's profile. Contains option to change password.
        - User statistics. List of all "runs" of the game. Contains option to delete an entry in the list.
    - question.html
        - The trivia page. Display questions + answer choices
    - register.html
        - Registration page.
    - results.html
        - Results of the game just played. Contains all users + their score.
- Functional Code
    - app.py
        - index(): Renders homepage.
        - login(): Renders login page.
        - logout(): Logs the user out.
        - register(): Registers the user (adds to the users database).
        - categories(): Renders the list of categories page.
        - question(): Renders the game page.
        - results(): Renders the game results page.
        - profile(): Renders the user profile page.
    - auth.py

        - encrypt(password): Hashes the entered string password using the sha224 algorithm.
        - new_user(): Adds a user to the users database. Password is encrypted before being stored.
        - verify(): Verifies if the password entered at login is the same as the one in the database.
        - user_exists(): True or false depending on whether or not there exists a password entry for a given username (Returns false if the username exists).
        - logged_in(): Is the user logged in?
        - login(): Log the user in (using session).
        - logout(): Log the user out.
    - database.py: Creates a user database if it doesn't already exist
        - adduser(username, password): Adds the user to the database. Returns True if user is successfully added to the database. Returns False otherwise.
        - empty_db(): Checks if the database is empty. True if empty, False otherwise.
        - get_password(username): Returns the associated password given a username. Returns None if there doesn't exist a password.
    - trivia.py
        - call_api(category)

Component Map:



Site Map:



Work Allocation:

Person	**Tasks (Highest -> Lowest priority)**
Kyle Lin (PM)	Those things nobody else wants to do, bug fixes, databases, Wikimedia API, bot logic
Tina Chen	Bootstrap, javascript, html
Fabiha Ahmed	Trivia API, app.py