JEOPARDY! by Team Saturated Sidewalks

Roles:

Peihua Huang : Project Manager
Alice Ni : Frontend (Bootstrap)/APIs
Jacob Olin : Databases
Hilary Zen : Flask

Project Description:

We will recreate the famous Jeopardy game. After users login, they will be able to create their own Jeopardy boards with their own categories and questions. These boards will be saved and be open for all other users to play. They can also choose a randomly generated board with predetermined categories and random questions from the APIs. Users can also search for an existing board based on the name of the board. When a user decides on a board to play, they will first have to specify how many teams are playing and the team names. There will be a limit of 5 teams. Once the game begins, the teams and their scores will be displayed on the bottom of the page. After choosing a question, the answer will be revealed with a click of a button after the team answers. If the answer matches, the user will add the points to that team. If the answer is wrong, the user will deduct the points. The game ends when all of the categories have been chosen and the winner of the game will be recorded in the user's game history.

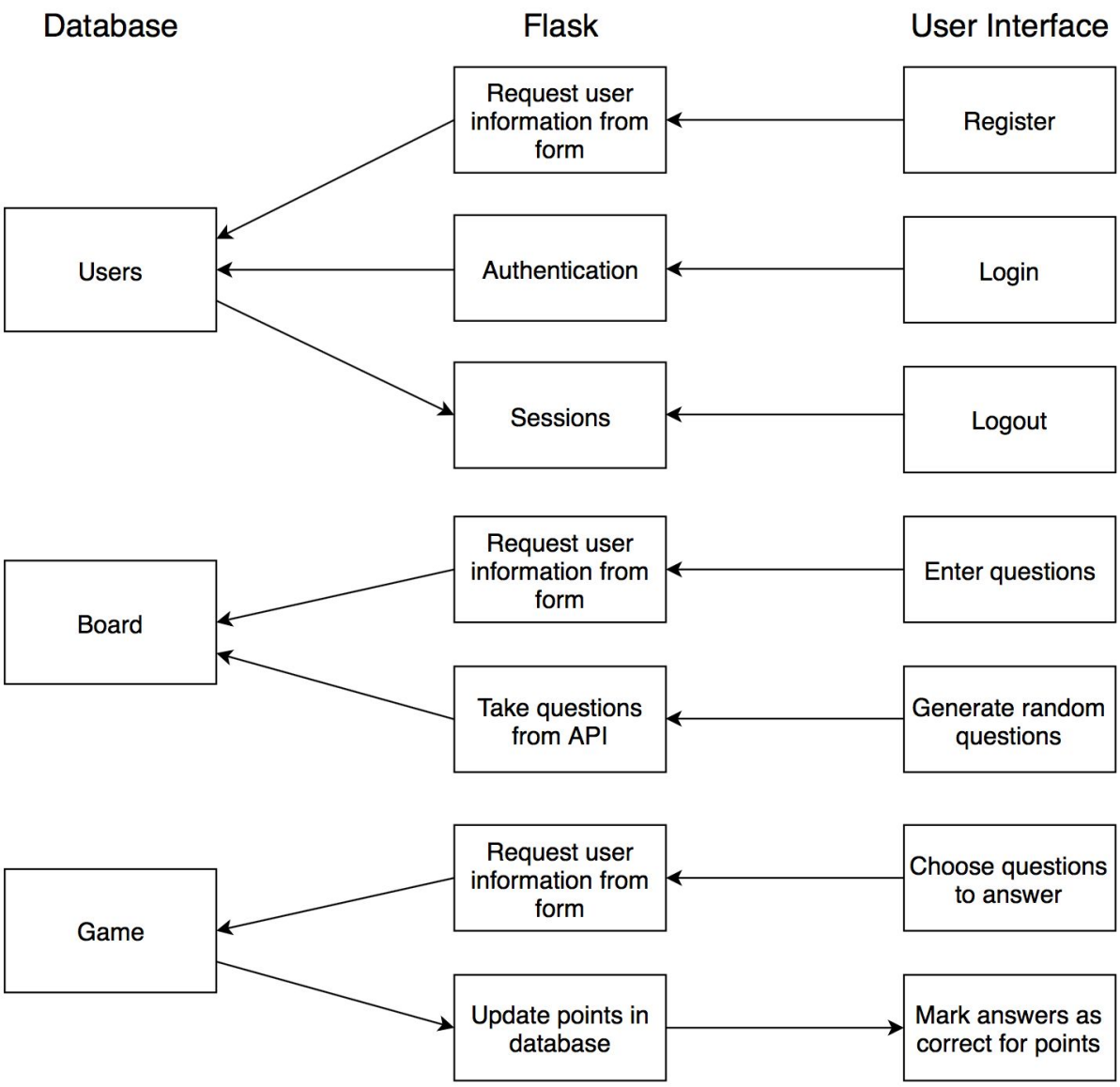
Core Components:

- Login system
- Users can start games, which will be recorded in their history
- Randomly generating questions
- Form to create their own board
- Keep track of number of rounds, which team is answering, and each team's points
- Determine a winner at the end of the game

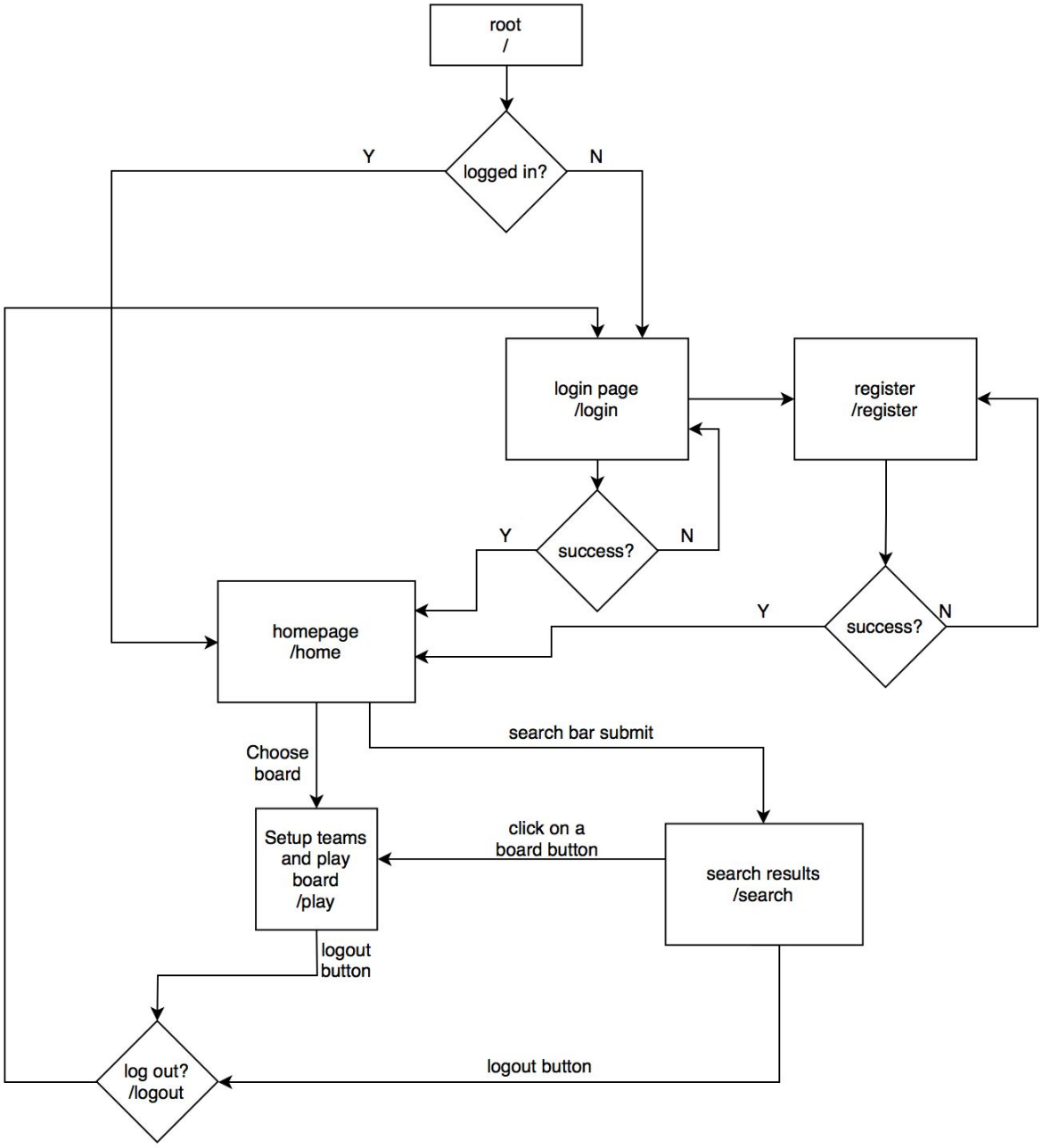
APIs Used:

Open Trivia - used to provide questions for the randomly generated boards
Rest Countries - used to get country flags and names for randomly generated questions
Rick and Morty - used to get character images and names for randomly generated questions
PokéAPI - used to get Pokemon images and names for randomly generated questions

Component Map:



Site Map:


root
/
logged in?
Y
N
login page
/login
register
/register
success?
Y
N
homepage
/home
success?
Y
N
Choose
board
search bar submit
Setup teams
and play
board
/play
click on a
board button
search results
/search
logout
button
log out?
/logout
logout button

Database:

Users

user_id	username	password
INTEGER	TEXT	TEXT
3892	bob	123

Board

board_id	user_id	board_name	row	category	q1	q2	q3	q4	q5
INT	INT	INT	INT	TEXT	INT	INT	INT	INT	INT
4337	3892	4337	0	pokemon	0	1	2	3	4
(5 rows per board id)	...	...	...	...	...	...	...	...	...
4337	3892	4337	4	countries	20	21	22	23	24

Teams

game_id	team_name	turn	score
INTEGER	TEXT	INTEGER	INTEGER
5238	Saturated Sidewalks	0	500

Board status

user_id	board_id	game_id	row	category	q1	q2	q3	q4	q5
INT	INT	INT	INT	TEXT	INT 0 or 1	INT 0 or 1	INT 0 or 1	INT 0 or 1	INT 0 or 1
3892	4337	5238	0	pokemon	0	0	0	1	0

Questions

Question id	Category	Question	Answer
INT	TEXT	TEXT	TEXT

0	pokemon	What is Pikachu?	<image link>

Front End:

- **base.html**
  - Base template for all the pages
- **board.html**
  - Gives user option for creating board
  - Redirects to “/create” to create a randomly generated one
  - Redirects to “/customize” to create a board with their own Q & As
- **customize.html**
  - Form for user to create their own Q & As
  - Redirects to “/home”
- **game.html**
  - Displays the jeopardy board and scoreboard for the game
- **login.html**
  - Form for login
  - Redirects to “/login” if unsuccessful, else redirects to “/home”
- **register.html**
  - Form for registration
  - Redirects to “/login”
- **home.html**
  - Search bar for available boards
    - Redirects to “/search”
  - Link to create game
    - Redirects to “/create”
  - Button to play games user already created
    - Redirects to “/play”
  - Button to start games using boards the user created
    - Redirects to “/play”
- **create.html**
  - Form for creating a randomly generated jeopardy board
  - Redirects to “/home”
- **play.html**
  - Form for user to input the number of teams and team names
- **search.html**
  - Buttons to all the boards whose names somewhat match the keyboard
    - Redirects to “/play”

Back End:

- **create_db.py**
    - SQL code to create the database tables needed
- **app.py**
    - **index():** Root route
    - **register():** Users can create a new account
    - **login():** Authenticates users who are logging in
    - **home():** Displays game history and boards created
    - **board():** Gives user options for creating board
    - **create():** Uses APIs to generate a board with questions
    - **customize():** Takes the 25 questions and answers that user enters, and creates a new board with them
    - **get_questions():** Gets the 25 questions from an already created board
    - **search():** Filters through all existing boards and returns matching results
    - **play():** Moves user into the game screen
    - **logout():** Logs user out