AnimalsOnFire -- Joshua Kloepfer, Yaying Liang Li, Qina Liu, Thomas Yu

SoftDev

P01 -- Design Doc

2021-12-09

Components

- Collectible: image of animals you can receive when you answer trivia questions right
- Trivia questions: questions testing user on facts
- Burning: image on fire
- Login accounts: users must need account to keep collectibles
- Collection: all the collectibles, items, special collectibles, achievements that the account has; essentially like user's profile
- Random Picker: mechanism by which a random collectible is given to user; basically like a gumball machine; duplicates not allowed
- Achievements: when user gets a certain item or a certain amount of collectibles (or an other assortment of tasks), user gets items or special collectible
    - Achievement page: shows all possible achievements you can get, ones that user have gotten are highlighted
- Items: items users can use to aid in collectible collecting [Hint and Extinguisher items]
    - Some make it easier to answer questions [Hints?]
    - Fire Extinguisher: put collectible out of fire and receive collectible despite answering question wrong
- Special collectible: collectible you can only get through achievements; basically like trophies
- *Optional (for when additional time)* -- add risky questions, where if you get it wrong, you lose a collectible you already own

Python:
- Will set up the database by creating the tables and adding information to each table
    - Users Table: username, password
    - Individual Tables
        - Name of the table will be the same as the username
        - Will keep track of all collectibles, items, achievements, etc
- Will interact with the APIs by making http requests
- Will send information using flask to create the webpage

Flask:
- Will create the webpages the user will interact with
    - Sends questions and the collectibles obtained by a user to be displayed
- Will create a session when a user is logged in

Sqlite3:
- Will have tables that will store information related to a user's account that can be pulled via python

Jinja2:

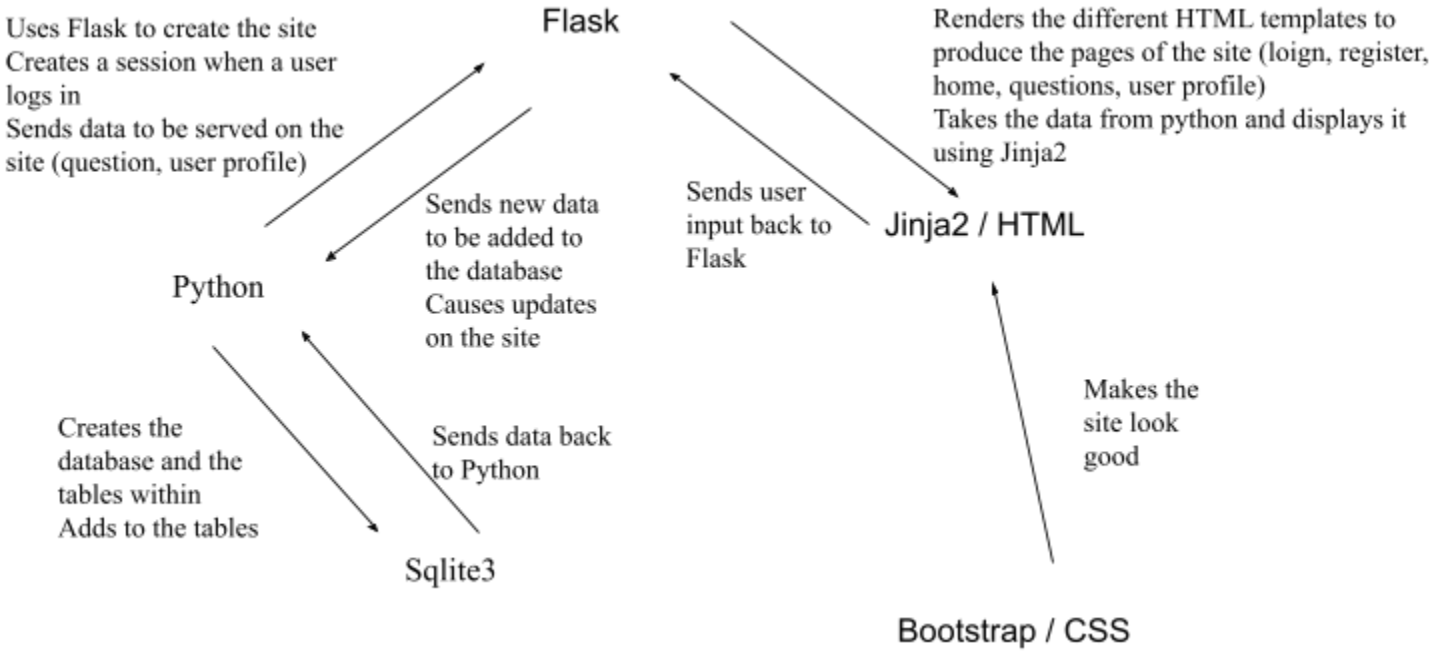
- Will be used to create the HTML templates that can display different trivia questions and the user's profile
- Also displays the collectibles' images and their burning on fire

Bootstrap:

- Will be used to setup the CSS for the site

How The Components Relate

- User answers trivia questions
  - If get it right → receive random collectible (no duplicates)
  - If get it wrong → sees collectible they would have gotten in fire
  - Collectible they get is chosen by random picker
- Account is needed for user to build collection (and thus all the things that require a collection to have, like items, achievements, special collectibles)
  - If logged in, get question right → user obtains collectible
  - If not logged in, user will see collectible (not on fire, will say that user has gotten said collectible) but will not show up in their collection
  - If not signed in, collection page will tell user to please login in order to build a collection
  - Not needed to sign in to answer questions, only needed to keep track collectibles gotten [I like that lol]
- Achievements gotten from collecting (based on collection)
  - Can result in:
    - Items that make collecting easier
    - Special collectibles
- Collection keeps track of and shows user's collectible, items, special collectibles, achievements



Database

- **Users** Table: Table to track usernames and passwords of registered accounts
    - Usernames must be unique
    - Password and username must exist

Username (TEXT UNIQUE)	Password (TEXT)

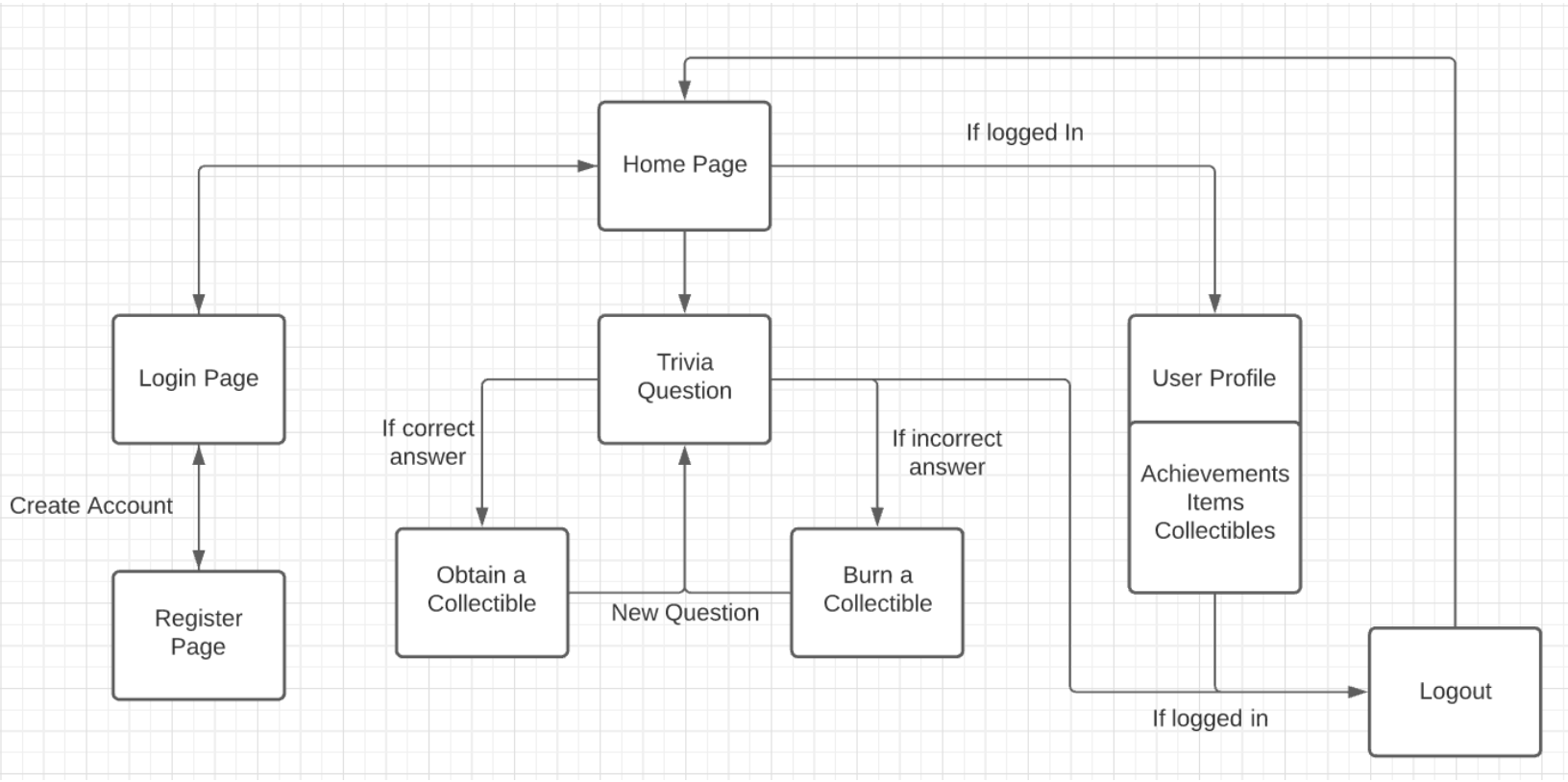
- **User Data** Tables: Individual tables for each user
    - Name for each table would be the username associated with it
    - Type of Object include collectibles, items, and achievements, question id
    - Can only have the ids of the questions the user answered correctly if the API provides an id with each question (Jservice or Trivia API)
    - Number will be used primarily for items (as you can have multiple)

Type of Object (TEXT)	Object (TEXT)	Number (INT)

User1

Type	Object
Item	Fire Extinguisher
Collectible	URL to Collectible0
Achievement	FIRST QUESTION
Collectible	URL to Collectible1

Sitemap



https://lucid.app/lucidchart/6623fcfd-1850-4195-bd06-d5f162a1d51f/edit?invitationId=inv_815d74d3-8ea0-44ad-aadb-41a884b35a9f&page=0_0#

Breakdown of Tasks

- Joshua Kloepfer: (Backend) Selecting random questions as well as checking if they are correct or not, Users Table, Trivia Table
- Yaying Liang Li: (Backend) Storing collectibles as well as questions already asked, Collectibles Table, selecting which collectible to give
- Qina Liu (PM): Frontend (How Site Looks), Collection page, burning, login + accounts
- Thomas Yu: Frontend (How Site Looks), achievements, items, special collectibles

APIs

Trivia: Could be just one or multiple

- Open Trivia (https://opentdb.com/)
    - Multiple choice or T/F, but no question id, has text encoding
- Jservice (https://jservice.io/)

    - Open ended questions, has question id
- Trivia API (https://trivia.willfry.co.uk/)
    - Multiple Choice questions, has id
- All APIs can:
    - Serve questions based on category
    - Serve multiple questions at once
- Will be used to request the trivia questions to serve to the user

Collectibles:

- Axolotls (https://theaxolotlapi.netlify.app/)
- Cats (https://alexwohlbruck.github.io/cat-facts/docs/)
- Dogs (https://dog.ceo/dog-api/documentation/sub-breed)
- Ducks (https://random-d.uk/api)
- Pets (https://www.petfinder.com/developers/v2/docs/)
- Each API will be used to obtain images that will be used as collectibles for the user to obtain

Framework:

Bootstrap

- Found it to be easier to work with compared to Foundation
- Documentation is easier to navigate for Bootstrap (https://getbootstrap.com/docs/4.0/getting-started/introduction/)

Target Ship Date: Jan 6, 2022

- We would like the due date to be after break rather than right before break, and the day following the first day returning seems conducive for non-stressed-out good code development
- Break good time for dry run too!