Akasha: Qianjun Zhou, Aidan Wong, Ivan Gontchar, Jason Chao
SoftDev
P02: Makers Makin' It, Act I
2025-01-07
Time Spent: 5
TARGET SHIP DATE: 2025-01-18

DESIGN DOCUMENT (VERSION 1)

I. Description

Up or Nah is a game where users can choose between two options for various topics. There will be three modes: “Classic”, “Tournament”, and “Wild West.” Classic mode is a spin on “Higher or Lower” from higherorlowergame.com, where users choose the option with the higher search count on Google. After making a choice, the numbers are revealed with a fun animation. The game will end when one incorrect answer is made, and the goal is to get the largest streak possible. There will also be fun mods like a time limit for choosing an option, reverse mode (where the goal is to pick the lower-searched topic), and even grid mode, where there are more than two options! The tournament mode is a space where user-generated datasets are put together into a bracket to see the best content of that category (ex. Music, anime, etc). The Wild West mode is where chaos ensues, ranging from guessing the exact search count of a topic to the rules changing mid-game. Although this is playable without an account, one must track personal scores, compete with other users on the leaderboards, and contribute data sets. This simple yet addictive game is the perfect way to pass the time while keeping you updated with trending and popular topics.

A. Program Components

a. User Accounts
   i. Creation of accounts and login/logout functionality
   ii. Score Tracking: Storing the users’ highest scores on a leaderboard
   iii. Sessions
b. Routes to different pages of the website using Flask and Python
c. APIs
   i. SerpApi - Google Trends API for checking trend value for each randomly generated topic
   ii. GiphyAPI - Generating related gifs for each Google search
d. SQLite3 Database
   i. Stores data of the user, user-generated datasets, and leaderboard information

e. Jinja Templates
    i. Main Game Page: Containers for comparing the two topics

B. Program APIs

a. SerpApi - Google Trends API for checking trend value for each randomly generated topic
b. GiphyAPI - Generating related GIFs for each Google search

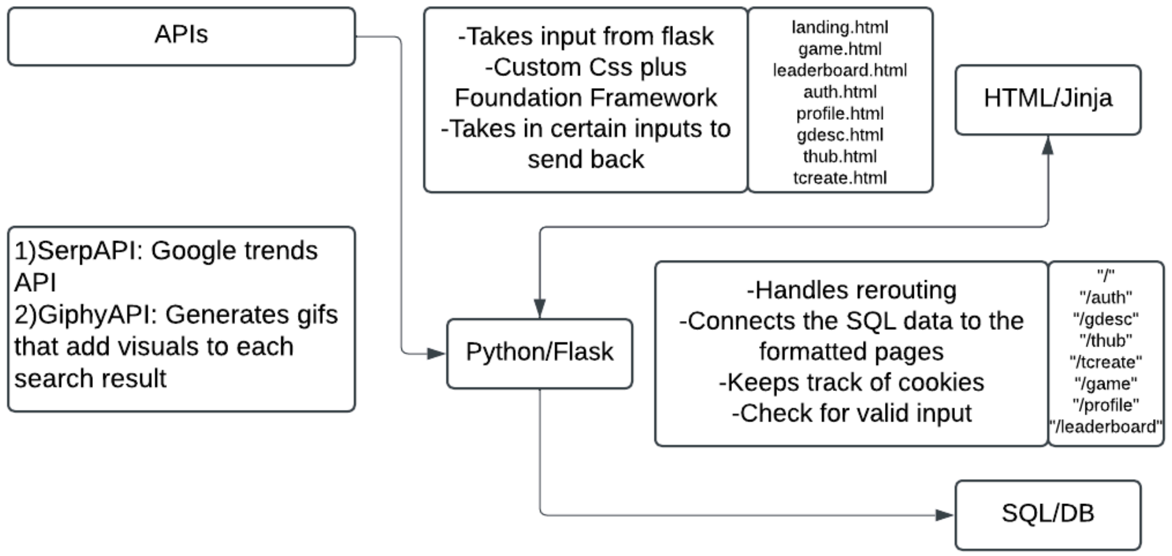
C. Frontend Framework: Foundation

a. Why Foundation?
    i. Easy-to-use and easy-to-follow tutorials
    ii. Vast amount of CSS components and JS ones that will make things look and feel good
b. How Foundation?
    i. Grid System: Structure layout of pages like main game page (Website will be able to adapt to size changes)
    ii. Pre-designed Components: Buttons, Forms, etc
    iii. JS Plugins: Modal Windows for confirmation messages

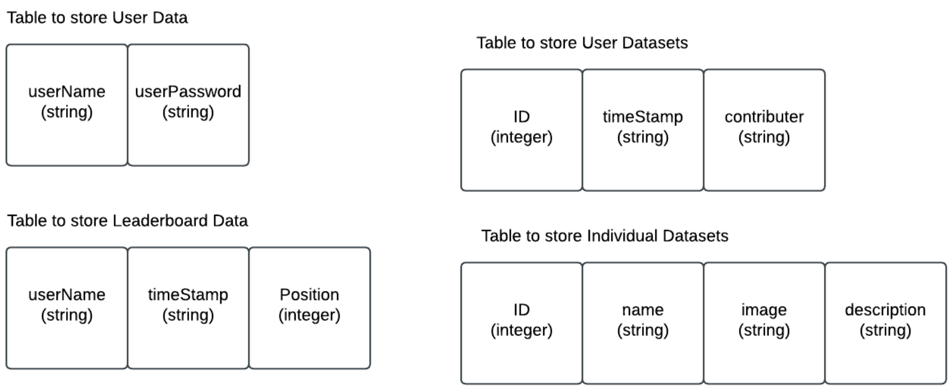
D. Program Component Connections

a. User accounts: Give access to individual statistics and player leaderboards
b. Routes + Python: Routes allow users to traverse the website. They connect the different pages (HTML documents) of the website. Python also interacts with APIs and the database and adds functionality to the various mods
c. Javascript: Animations and logical code for transitions
d. Database: Stores information related to the user (ID, Password, etc)
    i. One of the main factors for information exchange between components (has all the data)
e. APIs: Provides data for the randomly generated search topics
f. Templates: Allow for dynamic web pages

E. Component Map



F. Database Organization

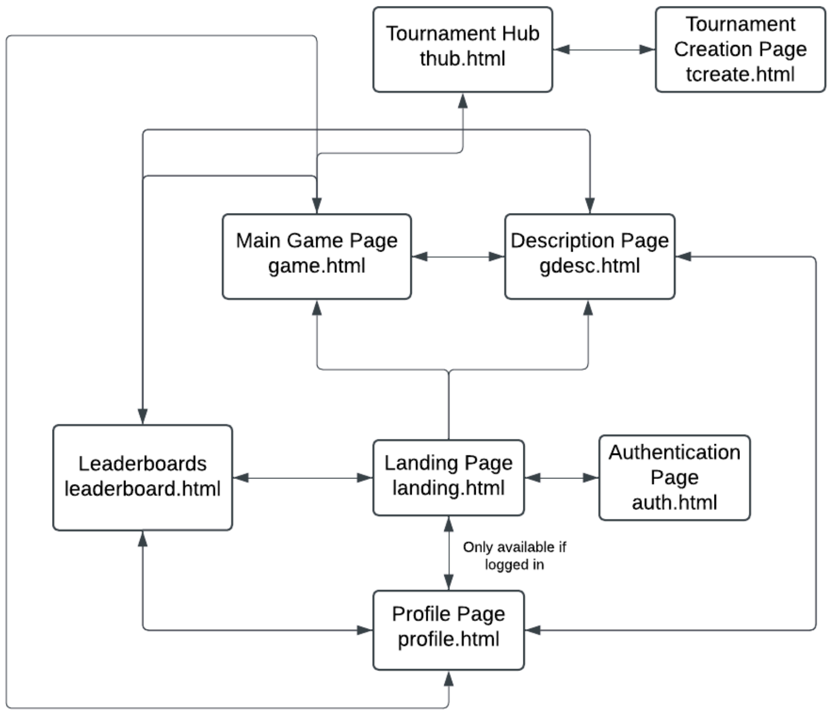


a. User Table
    i. Username (PK)
    ii. Password
b. Scores
    i. Username (FK)
    ii. Timestamp
    iii. Position
c. User Datasets
    i. ID (PK)
    ii. Contributor (FK - Username)
    iii. Timestamp

d. Individual Dataset
   i. ID (FK)
   ii. Name
   iii. Image
   iv. Description

Note: PK for primary key (each row value must be unique), FK for foreign key (to link tables)

G. Site Map + Descriptions


Tournament Hub thub.html
Tournament Creation Page tcreate.html
Main Game Page game.html
Description Page gdesc.html
Leaderboards leaderboard.html
Landing Page landing.html
Authentication Page auth.html
Only available if logged in
Profile Page profile.html

a. Landing Page (/): Homepage to the game; Contains buttons to log in, sign up, and play the various games
b. Game Description Page (/gdesc): Describes the rules for each game
c. Tournament Hub (/thub): Displays the various tournaments available from user-created datasets
d. Tournament Creation (/tcreate): Allows users to create datasets with custom names, descriptions, and images
e. Main Game Page (/game): Changes depending on the game chosen: classic, tournament or Wild West

f. Leaderboard (/leaderboard): Displays top player scores across all users for the various games
g. Login/Sign Up Page (/auth): Allows the user to register/login to their account
h. Profile Page (/profile): Displays the user's username and statistics

H. Task Breakdown

1. Qianjun Ryan Zhou: Frontend
    a. Create HTML pages with Jinja templating - Includes any forms required for logging in/signing up
    b. Design site style with CSS and front-end framework (Foundation)
2. Ivan Gontchar: Frontend (Javascript)
    a. Updating UI to reflect real-time changes in the game state/leaderboard state
    b. Animating site features (like buttons) and related game effects
3. Aidan Wong: Backend (Python and API functions)
    a. Routing and logic between pages + User session management
    b. Linking database and API functions with site features
4. Jason Chao: Backend (Database and API functions)
    a. Create SQLite3 database schema
    b. Work on database and API interaction modules for SerpAPI and GiphyAPI