Akasha: Qianjun Zhou, Aidan Wong, Ivan Gontchar, Jason Chao
SoftDev
P02: Makers Makin' It, Act I
2025-01-07
Time Spent: 5 days
TARGET SHIP DATE: 2025-01-17

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”, “Timed Trial”, and “Mania.” 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 to see if the player managed to succeed. The game will end when one incorrect answer is made, and the goal is to get the largest streak possible. In the timed mode, players will have to use quick judgment and reactions to choose the higher-searched item, while being careful to not run down the clock. In “Mania” the player has to fend for themselves, as the game launches unique game settings at them, in a flurry of many options and stress. 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
    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
    iii. APINinja - Getting relevant information for other higher-lower possibilities not based on Google searches
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. APINinja - Getting relevant information for other higher-lower possibilities not based on Google searches

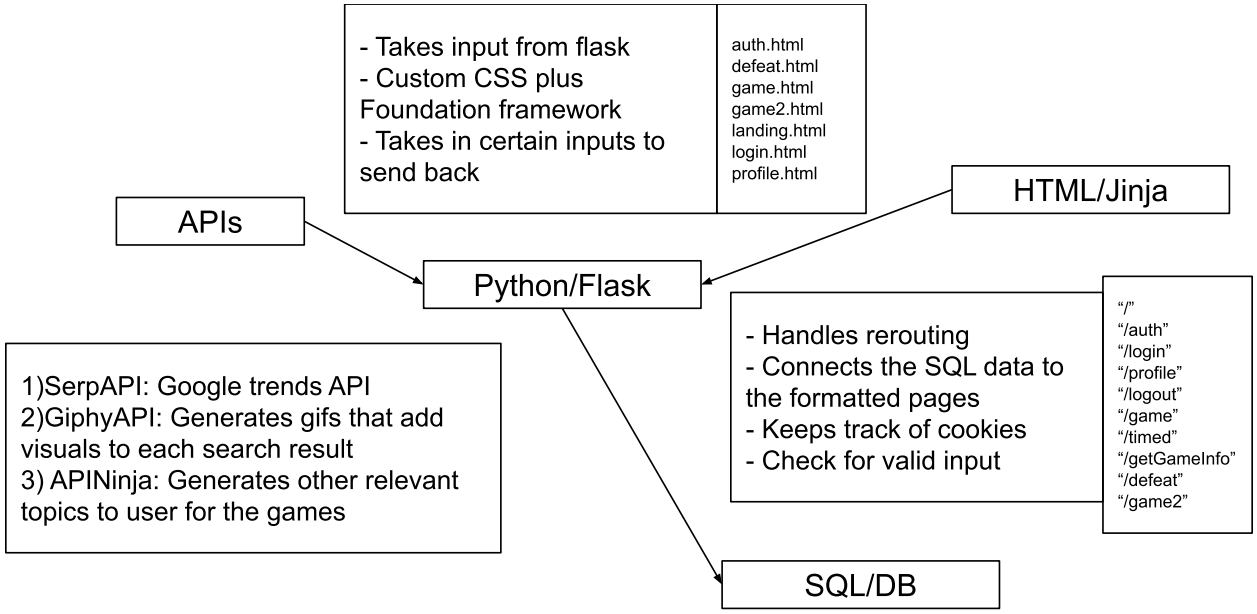
C. Frontend Framework: Foundation

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
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

Table to store User Data

userName (string)	userPassword (string)

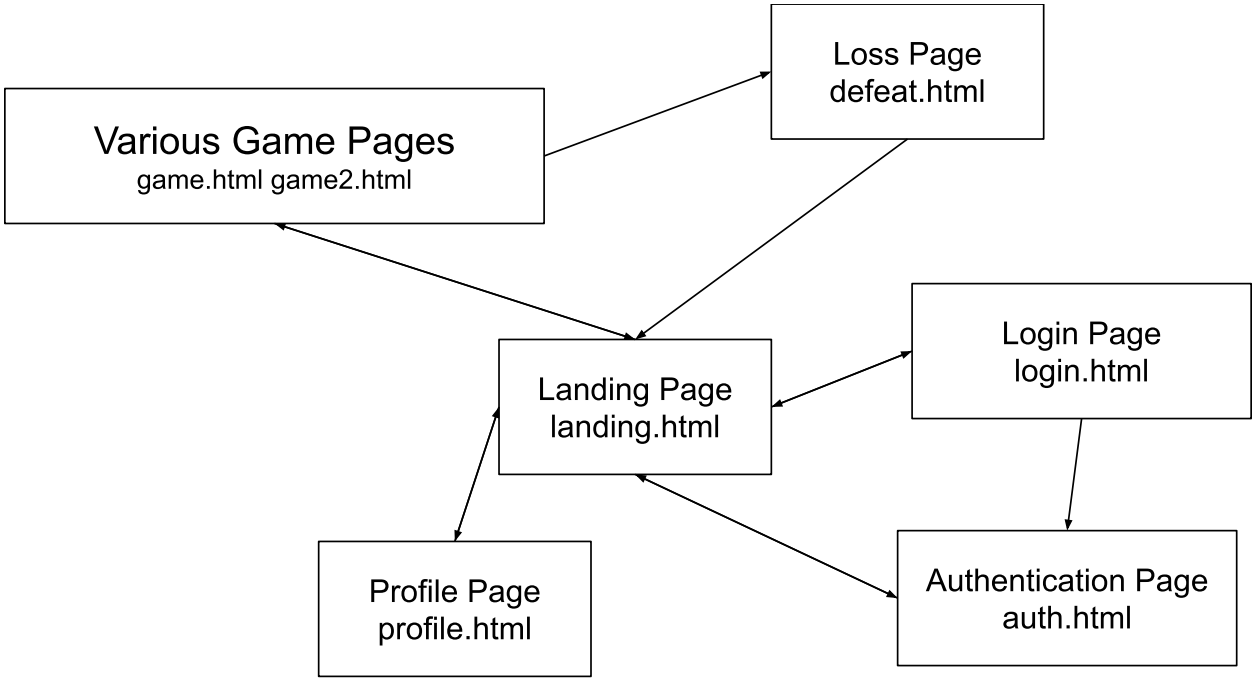
Table to store Game specific Data

userName (string)	bestScore (int)	foreignKey (refernces usernames in user data)

a. User Table
   i. Username (PK)
   ii. Password
b. Game Data
   i. Username (FK)
   ii. High Score

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

G. Site Map + Descriptions



a. Landing Page (/): Homepage to the game; Contains buttons to log in, sign up, and play the various games
b. Main Game Page (/game): Changes depending on the game chosen: classic, time trial, or mania
c. Login/Sign Up Page (/auth): Allows the user to register/login to their account
d. 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