UncommonApp - Dennis Chen, T Fabiha, Addison Huang, Michelle Tang
SoftDev6
Project 2 – The End
2018-01-07

Project Name: WeLoveHue
Framework: BootStrap
APIs to be Used: Color Scheme, IP Identifier, Weather API

Functionality:
Using RESTful APIs we will mimic the game "i love hue." People will be given a color scheme that broken is into pieces then randomized. Certain pieces will be locked, which will serve as a point of reference for people to complete the puzzle. They will then have to reorganize the pieces until the original color scheme is achieved. Creating an account will also allow users to create their own puzzles with customizations such as the size of the puzzle and the color scheme. Based off of a player's location and weather, players will be able to play limited edition puzzles. There will be puzzles that we (the devs) created that are open to everyone. For all puzzles, there will be a tracker to count how many moves are made per puzzle, which is then compared to the 'world' average.

APIs:
COLOURlovers API: http://www.colourlovers.com/api#palettes
IP Identifier: http://ip-api.com/
Weather API: https://openweathermap.org/api

Components:
Front End:

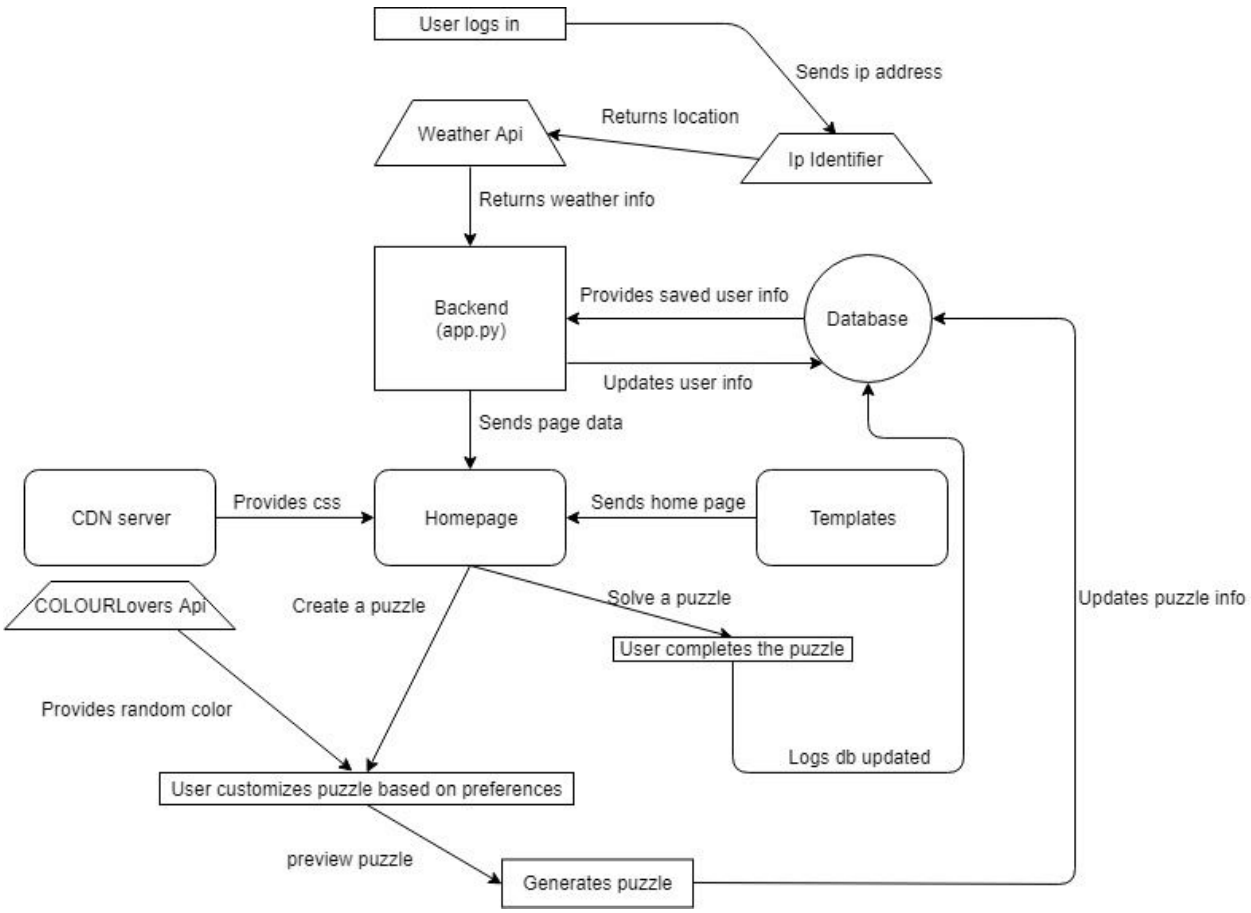
- Create an account, log in
- Create a puzzle
    - Selects a random color from COLOURLovers API, user picks the variability of rgb values, user picks the size of the puzzle
        - Jinja, html, bootstrap, javascript
        - User created puzzles are open to all other users!
    - Through the use of javascript, the user will be able to preview their puzzle as they are selecting their preferences
- Solve a puzzle (one that a player created or a preset one)
    - Javascript
    - Allow users to click two pieces and initiate a switch
    - Tracks the number of moves it takes to complete

    - Prevents users from making moves when the puzzle is solved
- HTML/CSS Templates
    - Jinja, html, javascript
    - Bootstrap will be used because it is reliable, consistent, and provides all the necessary tools.

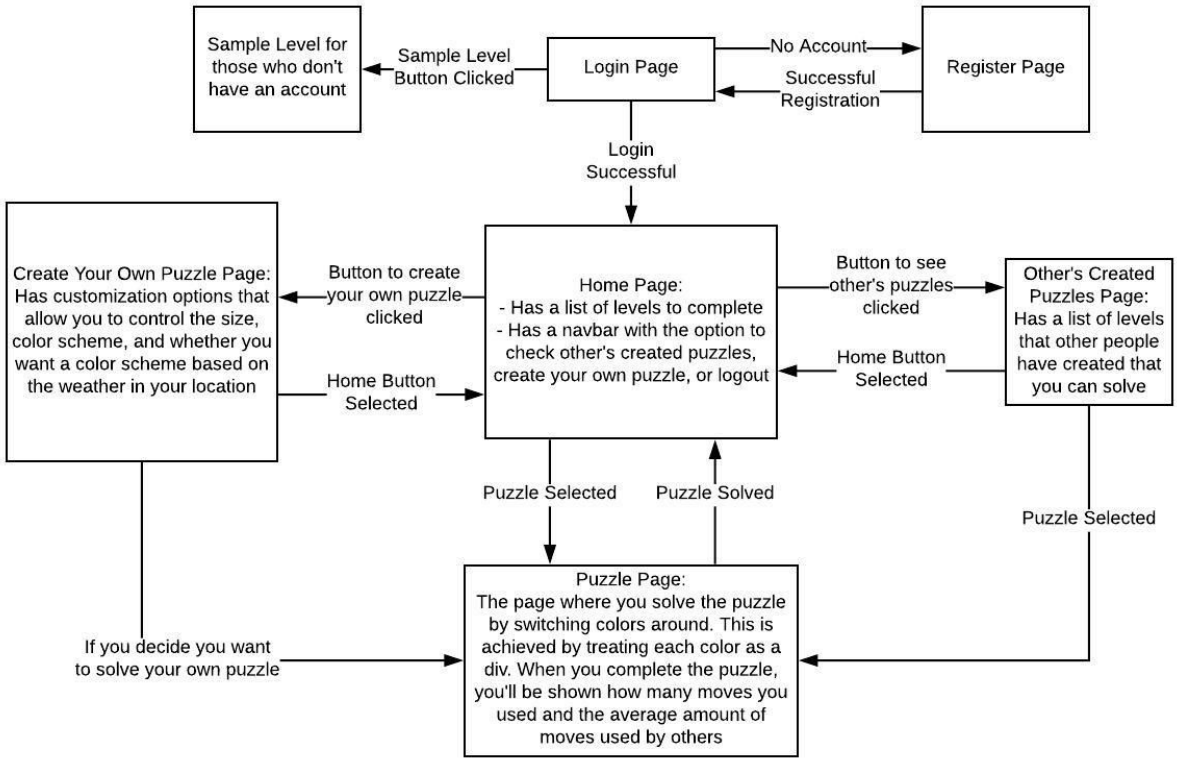
Back End:

- Flask route functions to handle the APIs, and rendering each page as appropriate.
- Work with the database to generate saved puzzles so that users can replay levels (will be displayed in a personal feed)
- Work with the database to generate public puzzles so that users can play alternate levels
- Create sessions
- Track moves of users to complete a puzzle to establish a 'world' average

Component Map:



Site map:



Database Schema:

- Facilitate logging in/out of users, as well as storing passwords(sessions) using dbs
- Maintain the storage of puzzles by assigning a puzzle an id, its content (colors), and how many moves, on average, it took to complete it
- To create a unique puzzle, we will follow a set format that will then be processed by our python file
    - For example, "{width in divs}, {length in divs}, #{upper-left color}, {right-horizontal change}, {down-vertical change}"
- Logs will track the history of a puzzle's completion including the username of those who completed the puzzle and the number of moves it took for them to complete it,

USERINFO

username	TEXT
pass	TEXT
moves	INT

liked_puzzle	TEXT

PUZZLES

Puzzle_id	INT
puzzle_content	TEXT
average_moves	INT

LOGS

user	TEXT
moves	INT
puzzle_id	INT

Role Distribution:

- Dennis:
    - Backend: Javascript, Python and flask functions
- Addison:
    - Frontend: Javascript and HTML/CSS
- Fabiha:
    - Project Manager || Prime Minister || Pretty Marvelous || Perfectly Mindful
- Michelle:
    - Backend: Python functions, and database creation and code for modification, creation or retrieval of entries. (For both tables within the database)