

Team Name: Music Under the Weather (Donia Tung, Giorgio Vidali, Masha Zorin)

### **Outline of Project:**

Combine use of the Weather Underground API and YouTube API in a website in which users will input their city and, depending on the weather of their location, the site will provide them with a playlist of songs (and possible links/youtube windows).

### **A list of program components(different files to do different things, file, data structure) with role of each specified:**

#### **APIs:**

Youtube:

<https://developers.google.com/youtube/v3/getting-started>

With this API, we're going to get information about the songs (like title, artist, and # of views) and display it to the user so they can know about the songs.

[https://developers.google.com/youtube/iframe\\_api\\_reference](https://developers.google.com/youtube/iframe_api_reference)

We're going to access YouTube videos and insert the videos into the website. It's this thing called iFrame which lets you put the embed the video into the website, not just a link.

Weather Underground

<https://www.wunderground.com/weather/api/>

From this API, we're going to gather weather information based on the city that the user inputs. There's a url link

(<http://api.wunderground.com/api/31c0e27929b4d46c/conditions/q/<STATE>/<CITY>.json>)

#### **Python:**

- Flask routes- To handle navigation, errors, and displaying

- Data parsing functions- To read databases and keep web pages updated

- Table modification functions- To update database with new users and user history/likes

- Search function- to lookup cities/countries

#### **Bootstrap + HTML stuff:**

- Search Result pages - show the playlist/videos that match the user's city's weather

- Login page - To login/sign up

- Home page - Searching for a City

- Signup page - for creating a new account

- Layout Page- Contains html layout for other pages

- Error Page- Displayed when something that shouldn't be accessed is accessed

**Database:**

Credentials Table: ID|Username|Password- For checking user credentials

History Table: ID|Video ID (of liked/viewed videos)

**Explanation of how each component relates to the others:****APIs:**

Weather Underground- users will input their city and we'll use the API to find the weather in their neighborhood

YouTube- will be used to get the popup YouTube window

**Database:**

Credentials Table- Checked by data parsing functions, updated by table modification functions

User History- Contains info to be parsed by python functions and shown on website

**HTML files:**

Search Result pages - Routed by Flask in python, searched by search function

Login page - Routed by Flask in python, form data authorized by data parsing functions

Home page - Routed by Flask in python

Sign Up page - routed by Flask in python for credentials and history tables to be modified

Layout Page- For other files to inherit

Error Page- The last case for most flask routes

**Python script:**

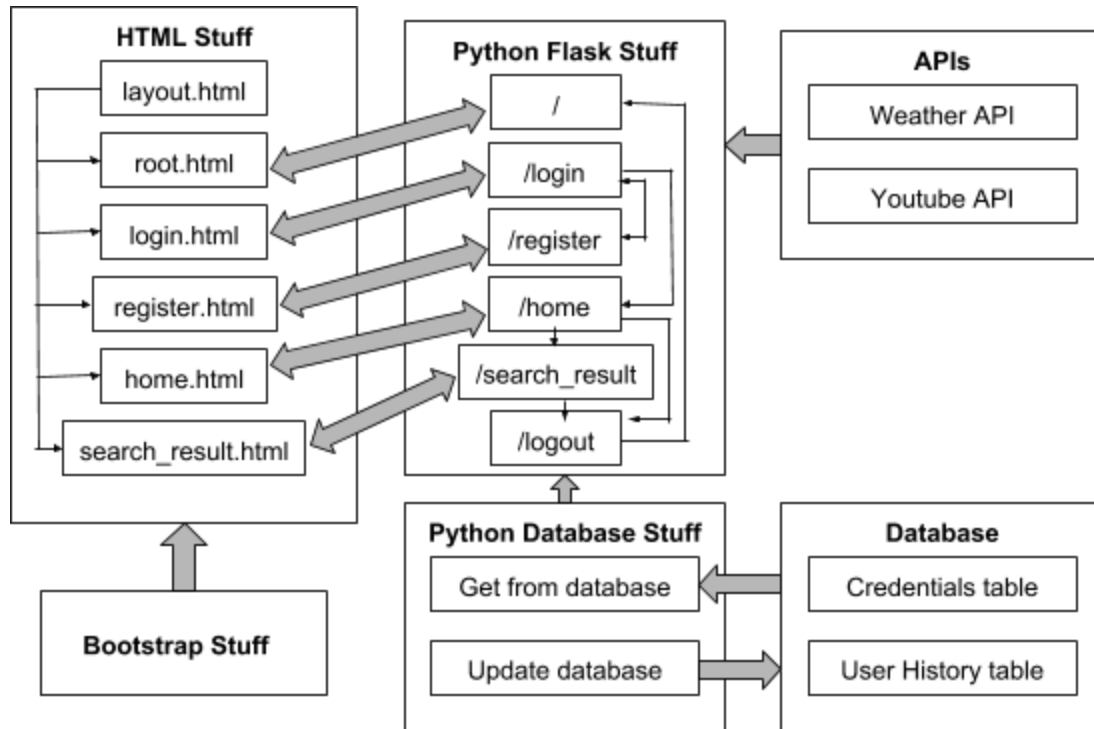
Flask routes- Handles navigation of HTML files by URL

Data parsing functions- Reads data from tables and updates web pages

Table modification functions- Updates data tables

Search function- Finds weather related to a city, finds the videos for the weather

Component map visualizing relationships between components



## Site map for front end:

Login

Welcome to  
Music Under the Weather!

YouTube  
vid will go  
here

Logout

Enter Your City

SUBMIT

Logout

Music suggestions:

YouTube  
vidz

Music Under The Weather

Login

Username: \_\_\_\_\_  
Password: \_\_\_\_\_

Submit!

Create Account

Music Under The Weather

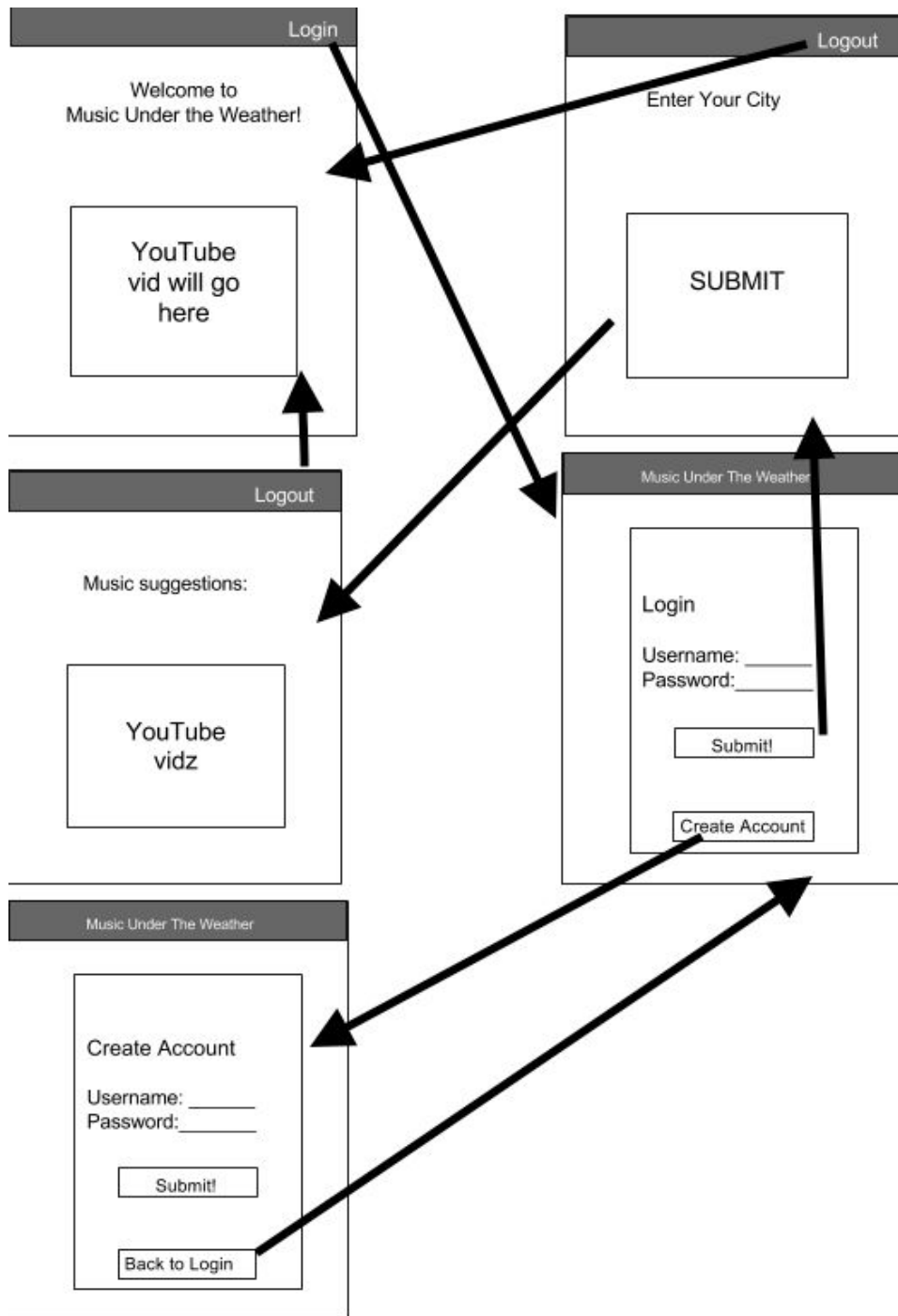
Create Account

Username: \_\_\_\_\_  
Password: \_\_\_\_\_

Submit!

Back to Login

### Site Map for Front End (With Arrows):



**Order of Creation/Importance:**

1. Search for city and get temperature from API
2. Return video (+have the video pop up) when a city is searched for
3. Login/Create Account
4. Accessing history of previously returned videos

**A breakdown of the different tasks required to complete this project:**

- Giorgio: html expert (bootstrap stuff), playlist creator, Chief coder, Total bro
- Masha: python expert
- Donia: Project Manager→ responsible for maintenance of devlog, monitoring equal distribution of work, and ensuring that a schedule is followed. Makes sure the group is consistently moving together, handles (minor) coding tasks as necessary, makes certain the design document is coherent and that the group is adhering to the agreed-upon design, be aware of all changes to the design.