

UnitedPurpleNPCs: Henry Bach, Jeffery Tang, David Deng, Shafiul Haque

SoftDev Period 8

P1 – Creepy Website

2022-12-04

Time Spent: 2 hours

Target Ship Date: 2023-01-05

### **Current Plan**

- User Feature:
  - Log in/log out features, and creating user profiles
  - Login page
    - Spaces to input username and password
    - Submit button to lead to home page
    - Register button to lead to sign up page
  - Sign up page
    - Spaces to input username and password
    - Create button to create account
    - Back button to return to login page
- Getting the information:
  - Use **IPStack API** to get the location of the user and get IP (user consent does not matter, we will get it either way)
  - Use **WorldTimeAPI** to get time of user using their location
  - Get air quality of location to somewhat calm them down since their IP just got taken
  - Use either weatherbit or weatherAPI to get the weather in that location
- Submit Button:
  - Use **LoveCalculatorAPI** or other program that takes some inputs and outputs some other type of information (gives us a percentage)
- Output:
  - Similar results feature (User can see other users with results that line up)
  - Use the percentages to display the possibilities of many things (chance you'll find love today, chance you're existence upon this planet that I write this from will end today, etc.)
  - Keeps those results and specifics in db to share as necessary (store most recent)

### **Possible Stretch Goal**

- DM them their results over Discord through a bot created using DiscordAPI
- Link discord with your account login created through our websites

### **List of Program Components:**

- Templates

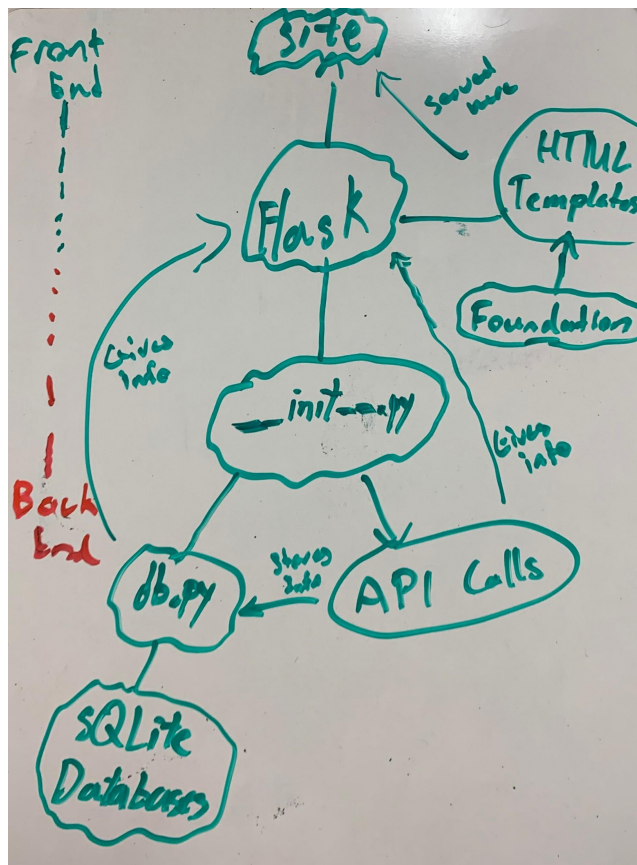
- Use **Foundation** as a Front End Framework to format pages
- Serves as a foundation for front-end to display the website (\_\_init\_\_.py)
- Database data displays in certain sections of the html
- index.html
  - The landing/main page
  - The path that the flask would go through is /
  - We plan to have a design of our team logo on the top of the page
  - If the user is logged in, there would be a logout button at the top to return to home page, a profile button to see your results if you have them, and a calculate results button if you haven't done so already
- main.html
  - The landing/main page
  - The path that the flask would go through is /
  - We plan to have a design of our team logo on the top of the page
  - If the user isn't logged in, there will be a login and sign up button
- login.html
  - The login page
  - The path that flask would go through is /login
  - If a user is already logged in, then this page would automatically redirect to the main page
  - There are signup and go back home buttons for navigations
- signup.html
  - The signup page
  - The path that flask would go through is /signup
  - A user who isn't registered already would signup here
  - There are login and go back home buttons for navigations
- results.html
  - The results page
  - The path that flask would go through is /results
  - A user who has clicked the calculate results button would be able to see their results
  - You can navigate back to the main homepage to try to get new results, or see comparison amongst other users
- similar.html
  - The similar page
  - The path that flask would go through is /similar
  - A user can see which statistics from their
  - You can navigate back to the main homepage to try to get new results, or go back to see your own results
- Flask App

- Helps display the html files through paths for certain linkages
- SQLite database functions references in order to add/delete data in the database
- `__init__.py`
  - routes to connect to HTML files
- `db.py`
  - Log in and out
  - Grab data from API and retrieve and set database information
- SQLite Database
  - Displays information in the html templates
  - Functions are used in the flask app to add/delete data in the database
  - `discobandit.db`
    - Log in and out
    - Store data from APIs (so we can find users with similar results)

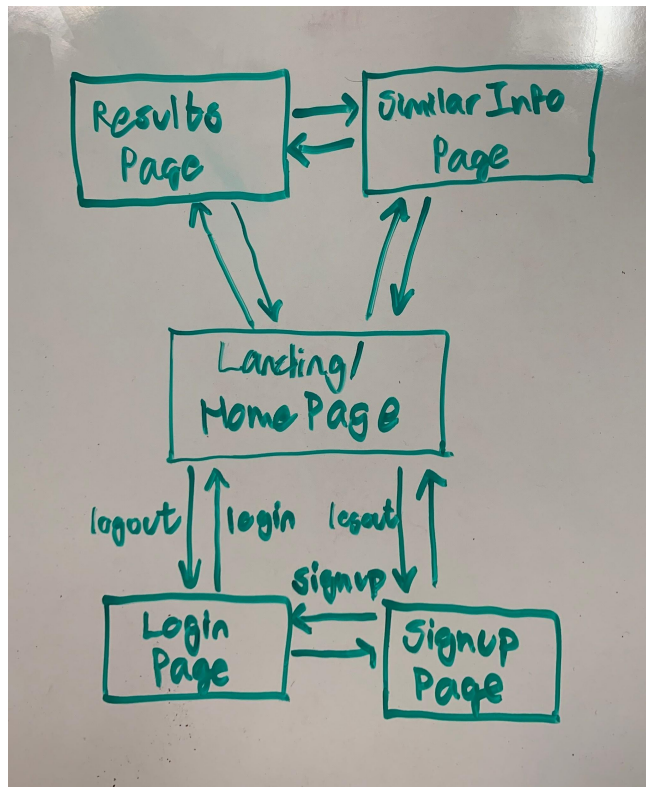
### Database Structure:

- Table with usernames (must be unique!) and passwords
- Data from IpStack API (city, zip code), WorldTime API (time), WeatherAPI (weather), and LoveCalculator API (percentage)

### Component Map:



## Site Map:



## Tasks:

- create the database and functions required for the app
- create html templates to display information and linking redirects from one page to another
- create the python app file in order to have the app running
- import the API data successfully, and use it in order to display data
- Maybe read from database and display previous results ~~leak other people's information~~ on the website as a possible frontend feature since right now, we only have a single form that will take their Discord username and their IP (I think we can get it even if they don't enter it so...) as website content.

## General Focuses (per member):

- Jeffery (sqlite)
- Shafiul (html)
- David (api/flask)
- Henry (api/flask)

**APIs:** IPStack API, WorldTimeAPI, LoveCalculatorAPI, DiscordAPI

**Front-End Framework:** Foundation

- the do it yourself style offers more unique and custom designs for websites
- more freedom on components such as grids and cells, which we will make of use

Make sure apis work