Queenie Guan
Professor Li
MIS3640 Problem Solving & Software Design
April 12, 2020

WebApp-MBTA Project Writeup and Reflection

Project Overview

When users land at the home page, there's a welcome message and they can choose from a list of service (currently only "Find nearest MBTA" available). When users click on "Find nearest MBTA," the website will redirect them to the form page. In the form page, users need to input their location, and have the choice to check any extra information they would like to include in the result, including wheelchair accessibility and next arrival time. If there is a MBTA station within 1 mile of the input location, the website will redirect to the result page, which contains the station name and any other extra information required by the user. If there is no MBTA station within 1 mile of the input location, the website will stay at the form page, but show an error message to inform the use that there's no MBTA station nearby.

Project Reflection

I finished the project in two sessions. At the first night, I finished getting the latitude and longitude from MapQuest, and got stuck in figuring out the MBTA URL. In the second day, after I carefully studied the MBTA documentation, I figured it out and finished the rest. Even though I am very unfamiliar in html and Flask, finishing the basic requirements wasn't too hard since I have the WebApp demo as a reference.

After I finished tall the basic requirements, I started to add more feature. After trying out a few, I finally decided to add the next arrival time as an extra piece of data. To improve the UX on the WebApp, I decided to provide a feature for users to include extra information in their result. I spent a lot of time learning html and flask during this part. First, I used select tag and option, but I couldn't figure out how to request in Flask. Then I realized there's a checkbox type input, so I used checkbox instead since that's actually what I wanted from the very beginning. The most challenging part was writing the Flask code, because unlike Python, there is way less resources on Google, and it took me so much time to debug. I wish I knew more about Flask before!

One thing I realized during unit testing is the importance to include try/except blocks in our code. I didn't feel it that much in the past because we always dealt with perfect data/input. But here, we are getting data from the API, and we don't know what's in there exactly. At start, I got a lot of error because there's no data being returned but the program is processing the data as if it's there. After I realize that sometimes the data is just not there, I added try/except blocks so that when error arises, the program will run without error and return a message or set variables to empty.