John Falcone

SI 206 - Fall 2017

Final Report

**Goals**

My original plan upon starting the project was to use Spotify and Google’s respective APIs to create a playlist of songs about a certain city or location (“Boston” or “New York City”, for example). After experimenting with Spotify’s API, I came to the realization that Spotify didn’t include artist location data in any of their API calls. Unable to retrieve the data I needed, I decided to pursue an alternative plan. I wanted to use data about my Facebook activity, and see how it corresponded with the weather and news of the day the activity took place. With my new plan, this was the goal - creating a report of what the world around me was like while I was interacting with Facebook.

**Achieved**

I achieved that which I’d set out to do in my plan. My project fetches data from Facebook, and uses it to extract date and time data which is then turned around and used in the other two API calls. This data is then used to create a report on the national weather and top news headlines on that date. There were many setbacks - primarily regarding the data that I was retrieving from Facebook.

**Setbacks**

The biggest setback occurred early on in the development process. Facebook’s Graph API is tricky, and it doesn’t always provide what you’d be looking for. Where I was looking for all interaction activity, Facebook would only allow me to fetch activity on pages that I’ve liked. As I was attempting to create a library of my recent Facebook interactions, it came to my attention how little I’d actually used Facebook. I wouldn’t consider myself an avid Facebook user, but according to Facebook I’ve only “liked” eight pages since mid-2015.

Here’s where the first pivot occured. In attempting to gather 100+ interactions, it wouldn’t be possible using the Facebook Graph API. I also didn’t want to go and like 90 more pages on Facebook, because then the data I’d be gathering would be inauthentic. So instead of relying on Facebook as my source for 100 interactions, I decided to use the data that Facebook provides to create the basis on which the rest of the project is built. In practice, I used one key data point from Facebook - the date and time at which I last liked a page - as the key piece of data to be used in the rest of the project. This date is then used to call the DarkSky API. The DarkSky API returns historical weather data from that date, and in my case the API is called 50 times - one for each state capitol - and provides the high and low temperature in that capitol on that day. This call conveniently covers 100 interactions all on its own.

While we’re on the subject, accessing three different APIs was also a struggle. Each API has it’s own requirements and calling protocol, and for each the data structures are different. Figuring out how to extract the exact data I wanted from each API call took some time.

Another general setback was the caching management. I use three different file types for caching; JSON, SQL, and .txt (for the news data). Managing these different file types and having them all function as intended, in coordination, posed a challenge. Along the way, I gained a better understanding of these file types and how to use them, so I can’t complain.

**Report/Output**

Here’s an example of the report that the project generates. It outlines the actions it performed and the data it collected, and then presents it to the user.

Data in the cache

Fetching most recent date of Facebook Like.

Like Date matches Storage Date. Cached data is up-to-date.

Weather Data in SQL Database is up to date. No need to make request to DarkSky API!

Headline Data in cache file is up to date. No need to make request to News API!

\*\*\* DATA SUMMARY \*\*\*

Welcome to the data summary! Here, you can see the data that the program has collected. Let's go one API at a time. First, using a call to the Facebook API, we received the date of your most recent post like.

Your most recent post like occured on: 2017-12-05

Using that date, the DarkSky API can be called. The DarkSky API can take a timestamp as input and return historical weather data. Take a look at what the high and low temperatures were on that date in all 50 US state captials.

On 2017-12-05 the high and low temperatures in US state capitols were:

Albany High: 39 Low: 35

Annapolis High: 50 Low: 44

Atlanta High: 65 Low: 55

Augusta High: 37 Low: 27

Austin High: 82 Low: 54

Baton Rouge High: 76 Low: 64

Bismarck High: 27 Low: 8

Boise High: 39 Low: 20

Boston High: 44 Low: 34

Carson City High: 35 Low: 14

Charleston High: 64 Low: 53

Cheyenne High: 30 Low: 14

Columbia High: 65 Low: 47

Columbus High: 56 Low: 50

Concord High: 43 Low: 29

Denver High: 37 Low: 16

Des Moines High: 67 Low: 24

Dover High: 51 Low: 42

Frankfort High: 67 Low: 55

Harrisburg High: 49 Low: 42

Hartford High: 48 Low: 35

Helana High: 29 Low: 23

Honolulu High: 77 Low: 70

Indianapolis High: 63 Low: 39

Jackson High: 72 Low: 64

Jefferson City High: 69 Low: 31

Juneau High: 40 Low: 37

Lansing High: 56 Low: 40

Lincoln High: 60 Low: 24

Little Rock High: 66 Low: 46

Madison High: 60 Low: 25

Montgomery High: 61 Low: 55

Montpelier High: 41 Low: 33

Nashville High: 68 Low: 56

Oklahoma City High: 74 Low: 34

Olympia High: 41 Low: 32

Phoenix High: 71 Low: 48

Pierre High: 35 Low: 17

Providence High: 47 Low: 35

Raleigh High: 60 Low: 42

Richmond High: 56 Low: 39

Sacramento High: 59 Low: 43

Saint Paul High: 55 Low: 16

Salem High: 44 Low: 33

Salt Lake City High: 32 Low: 19

Santa Fe High: 46 Low: 23

Springfield High: 65 Low: 31

Tallahassee High: 72 Low: 56

Topeka High: 68 Low: 26

Trenton High: 51 Low: 43

Pretty cool, right? Now, let's use the News API to dig up the top news stories from that day. Summaries of the five most popular articles from CNN can be found below.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Danny Masterson off 'The Ranch' amid rape allegations

Danny Masterson is out of the Netflix comedy "The Ranch" amid rape allegations made against him.

\*\*\*

Moore's opponent blasts 'men who hurt girls'

Democratic Alabama Senate candidate Doug Jones references sexual abuse allegations against GOP Senate candidate Roy Moore by saying that men who hurt little girls belong in jail.

\*\*\*

Bodycam shows frantic effort to help choking newborn

Bodycam footage shows a Georgia police officer resuscitating a month-old infant, saving her life.

\*\*\*

Ball: I think my son has had enough

In an interview with CNN's Chris Cuomo, LaVar Ball explains why he pulled his son, LiAngelo, from UCLA, saying the punishment has been enough.

\*\*\*

Phony sign language interpreter signs gibberish

An imposter sign language interpreter crashed a Tampa police press conference. Police were announcing a major arrest in the recent Seminole Heights murders.

\*\*\*

There you go. Those are some great examples of the power of APIs. Thanks for using my program! - John

**Run/Setup Instructions:**

Four files are needed in the same directory to properly run the project:

1. The project file, titled **final\_project\_main.py**
2. A file in which you’ve stored three API codes, titled **api\_codes.py**

Within this file, you need the following:

facebook\_token = "YOUR FACEBOOK TOKEN HERE"

news\_token = "YOUR NEWS API TOKEN HERE"

weather\_token = "YOUR DARKSKY TOKEN HERE"

1. A SQL database file set up with a WEATHER table.  
   Refer to the code below for the exact setup of my file  
     
   import sqlite3

conn = sqlite3.connect("state\_weather\_db.sqlite")

cur = conn.cursor()

cur.execute("DROP TABLE IF EXISTS WEATHER")

cur.execute('CREATE TABLE WEATHER (city TEXT NOT NULL PRIMARY KEY UNIQUE, temp\_hi INTEGER, temp\_low INTEGER, data DATETIME)')

1. A json file, **state\_data.json**, which contains the U.S. capitol names, latitudes, and longitudes used in the call to the DarkSky API. This file is included in the project zip file.

An important note about Facebook’s Graph API - from my experience, specific fields need to be requested when setting up the Graph API call in the Facebook Developer’s panel. Make sure that the code you’re using is requesting the fields *id, name, likes* as shown in the project code. This may be the default call. Either way the program should run on its own, as the cache file should have enough stored data to work with. In summary - if there’s an issue with the Facebook API call, it’s likely due to the setup of the call in the Graph API.

If this is done properly, the code should run successfully as-is. In my experience, a completely fresh run of the program runs ~20 seconds. This, I believe, is due largely to the DarkSky call, in which 50 calls are made to the DarkSky API and the data is extracted and stored for each one. After this initial run, the program runs much quicker working off the cached data.

**Documentation**

The main project file **final\_project\_main.py** is completely commented. Everything within the file should be commented and self-explanatory, and accurately describe how the project works piece-by-piece. Provided the setup instructions have been correctly followed, the code should run without issue. Please see the main project file for specifics on how the code functions.

**Resources**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Issue Description** | **Location of Resource** | **Result**  **(did it solve the issue?** |
| 12/7 | State Capitol Locations needed | https://inkplant.com/code/state-latitudes-longitudes | Yes |
| 12/7 | News API documentation needed | https://newsapi.org/ | Yes |
| 12/7 | Facebook Graph API documentation needed | https://developers.facebook.com/docs/graph-api/ | Yes |
| 12/7 | DarkSky Weather API documentation needed | https://darksky.net/dev/docs | Yes |

**Takeaways**

This project taught me a lot about APIs and caching. The Facebook, News, and DarkSky APIs were difficult to manage at first but eventually I was able to accomplish what I’d set out to do with them. In the future, I’d like to see this project be applied to a wider range of dates. Because of Facebook’s API limitations and my limited Facebook use, I didn’t receive many timestamps to work with from the Facebook API call. I’d like to find an API that would allow me to perform the same type of analysis regarding my social media use and the environment around me, but on a much larger scale. In doing this, I would hope to be able to perform deeper analysis and pick up on trends regarding my social media use and the world around me. As a daily user of Twitter, perhaps through their API I’d be able to accomplish this. Regardless, I’m glad that I got to do this project and demonstrate what can be done with APIs and Caching. Thanks for reading!