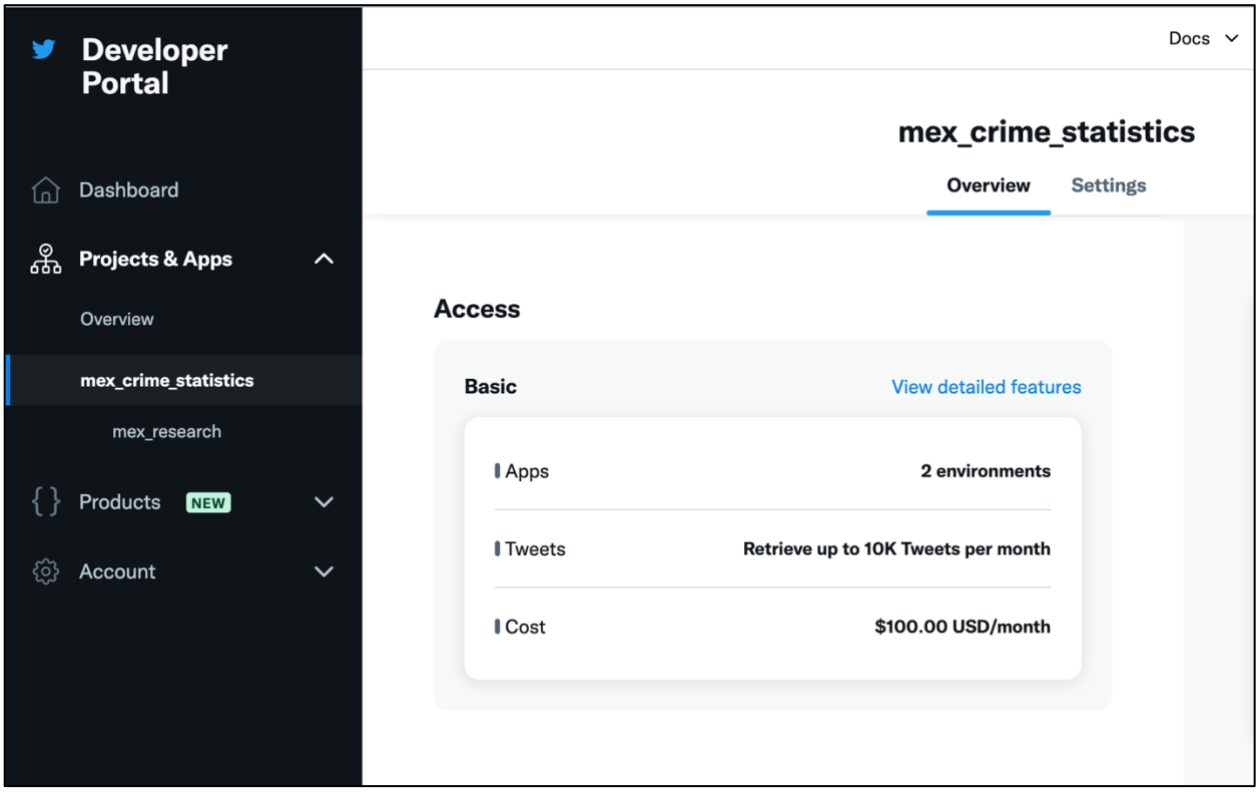
Technical Analysis Section

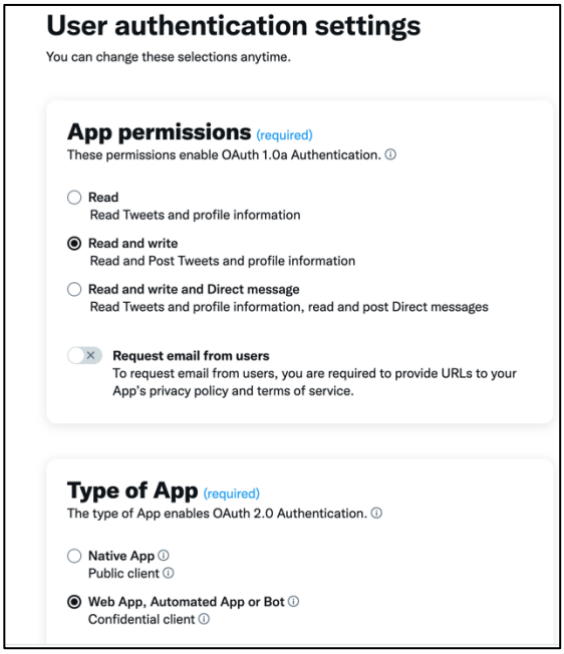
This section serves as a walkthrough analysis on how to do the two approaches I presented in my capstone research paper. In order to use Twitter (X) data, a user first must gain access to Twitter API. Currently, there are 4 levels of API with each level increasing the number of services and products available to the user. This paper uses the Basic API version, which is $100 per month making it the most affordable yet restrictive as it limits the types of twitter data available. For instance, this API does not allow the user to filter (do a mass search) tweets based on geo location, language, hashtags, or provide access to the Twitter archives, which are all critical elements necessary for this analysis to be as accurate and comprehensive as possible. However, these features are available on the higher API levels like the Pro version for $5,000 per month, or the Enterprise version for $42,000 per month. That being said, given my limitations on the data, I have produced a two walkthrough analyses, which can be combined to serve as a model, idea, or guide to help users with higher API access conduct a more comprehensive and accurate analysis on crime in Mexico.

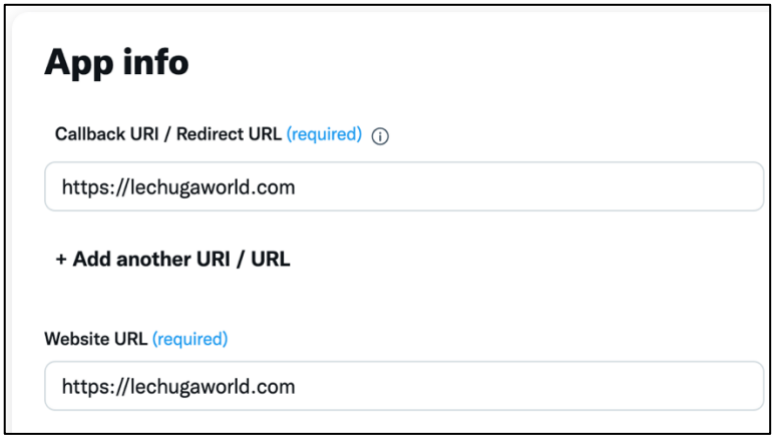
A) Gaining Twitter API Access

Before being able to extract any Twitter data, the first step is to gain API access by creating a developer account on https://developer.twitter.com/en/docs/twitter-api. Once an account is created, the user needs to pay for the Basic API level.

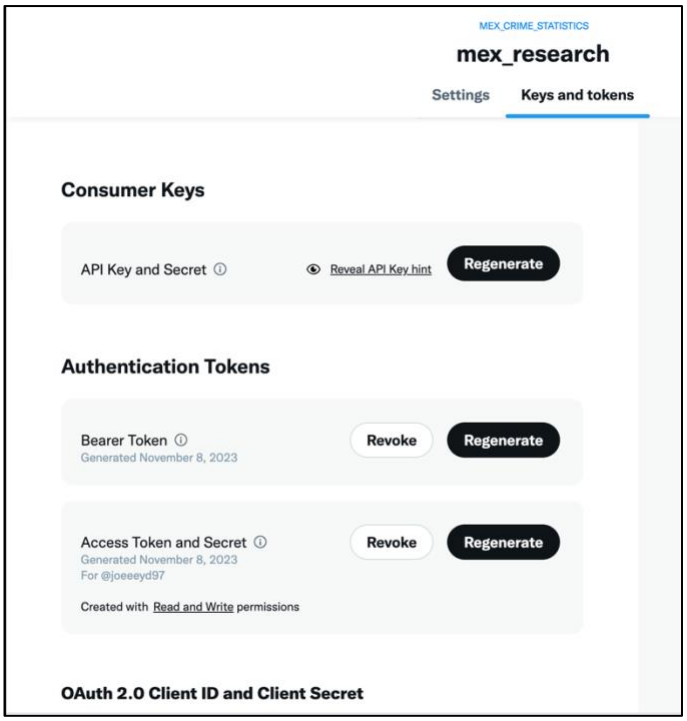


On the bottom of this page is the “Apps” section. Please create and name your app. This will prompt your “Keys & Tokens”, which will all need to be regenerated after the proper API environments are set up. When the app is created, please click on “App Settings” and navigate to “User authentication settings”. Adjust the settings to look like the below pictures. As for the “App info”, please feel free to put any made-up link.





When the proper environments are set up, head over to “Keys and Tokens” and regenerate everything. It is very important to save your keys and tokens as they are needed to connect to the API.



B) Postman

After successfully creating the twitter developer account, I now have the necessary keys to call on Twitter for data extraction. With the use of the keys, you can either use your local desktop to call on twitter, or you can use a web application. Recall, in the earlier steps I selected the "Web App, Automated App, or Bot" option. This is because to call on twitter through your local desktop requires constructing complex code, whereas using a web application is as simple as inserting your keys and filters for your tweets. That being said, I selected to use Postman as the main web application for its simplicity and user-friendly environment. This website will allow the user to make requests to Twitter using your keys and tokens. Click on this link https://developer.twitter.com/en/docs/tutorials/postman-getting-started for Twitter's walkthrough on how to link your Postman account. The steps are a bit outdated, so on the final step you will get an error result, but to avoid this please create your own individual Postman Account and redirect back to the link to connect your account with Twitter's API v2 collection by "Creating a fork".

Getting started with Twitter's Postman collections

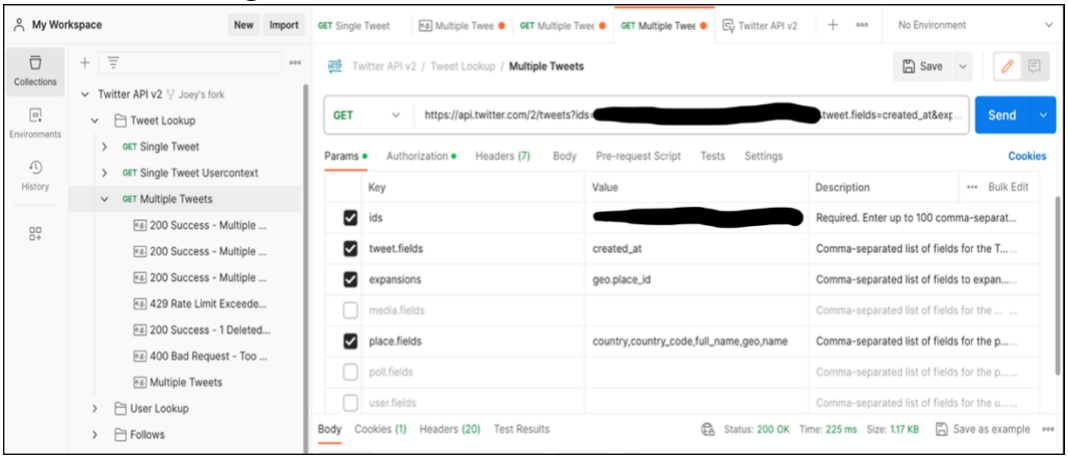
Step one: Add one of the Twitter Postman collections to your account

While you could build out the specific endpoints that you'd like to use within Postman, we did all of the heavy lifting for you and built out a ready to use collection of relevant APIs. Just click one of the links in the earlier "Postman collections" section and a collection with all of the endpoints associated with the selected API will be added to your Postman app. These collections are also available in the Postman API network. Each endpoint will automatically include available parameters, example responses, and authentication type plugged in, so you just need to add your credentials and parameter values to start exploring the functionality.

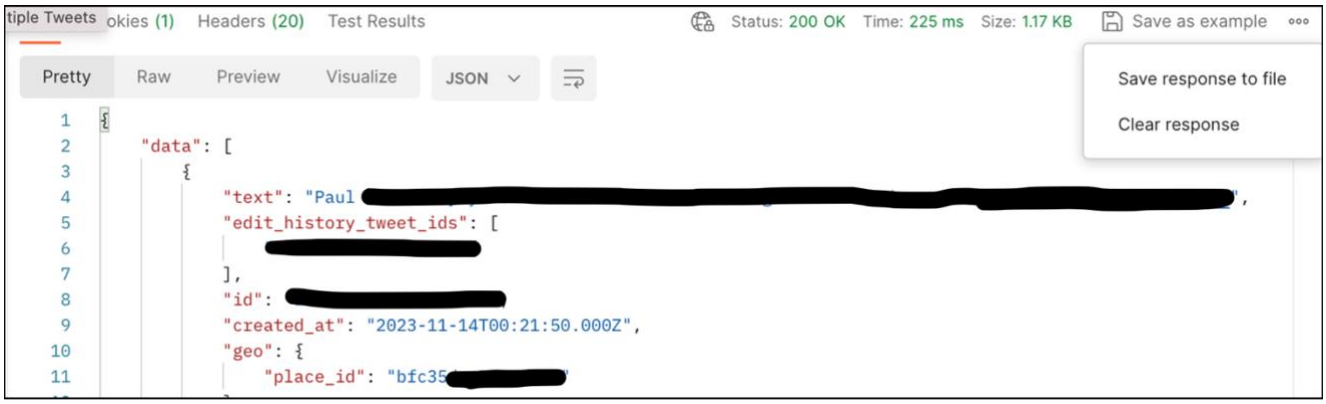
In this example, we are going to work with the Twitter API v2 collection.

C) First Approach (geo_location_extraction)

Now that the keys and Postman account are fully created, I started with my first approach that focused on cleaning twitter data to extract the geocoordinates of tweets for map layering. As mentioned before, I have restricted use of Twitter data, but I was able to closely recreate steps that will work for higher API levels. For instance, instead of mass searching twitter posts based on the critical features, I had to manually look for tweets. To do this, I set up the parameters on Postman as the following:



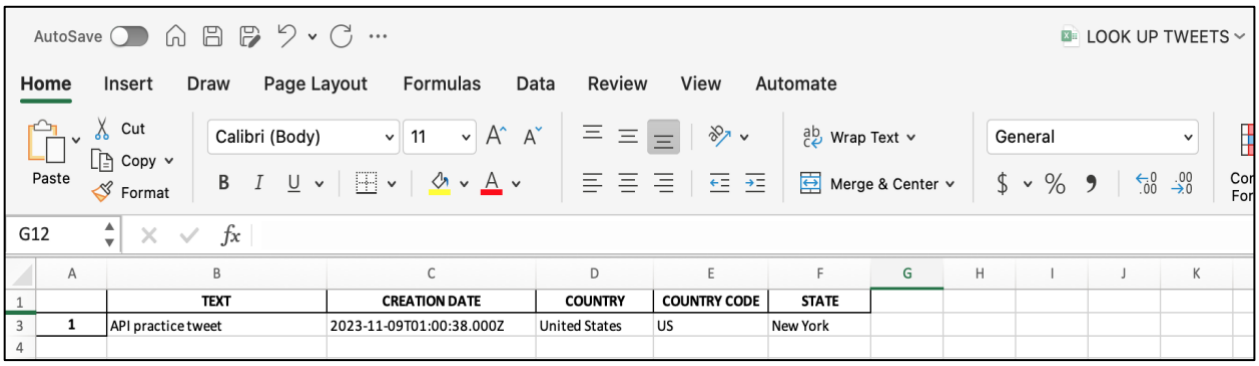
The “ids” are simply the numbers at the end of a Twitter URL post, so just I copied and pasted the IDs to the id box. Before pressing send, click on the “Authorization” tab and in the left section labeled “Type” click on “Bearer Token”. Please add your bearer token. You are now ready to make your first request. The request should pull twitter data in JSON format. If not, change it to JSON. Save it as a file.



NOTE: To get started on extracting the twitter data, I used Python and Jupyter Notebook. For the extraction to work, it is imperative that the JSON file is saved in the same location as the notebook. In this pull request, I extracted twitter posts that contained the following fields: created_at, geo.place_id, country, country_code, full_name, geo, and name. Ideally, a user with higher API access could simply add those exact parameters with no input of twitter ids.

Outcome of Approach 1

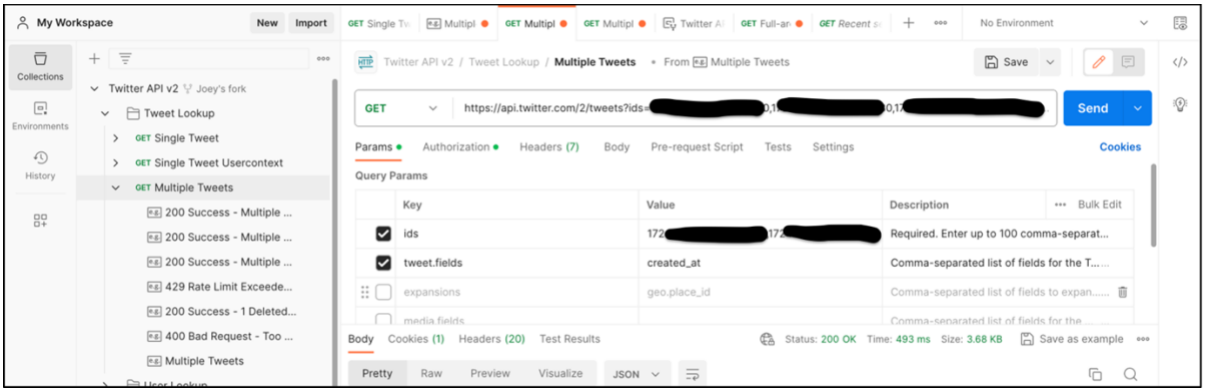
After following the steps in Postman and the code in the geo_location_extraction notebook, the final output is the following:



NOTE: Given this excel file, I could use a map making software, like QGIS, to layer it over a map of Mexico that contained geocoordinates. Unfortunately, I was unable to complete this step due to difficulties in downloading the mapping software. However, a simple join between common fields like “State” for both the excel file and map coordinates would have plotted the tweet as a point. Overall, even though I manually added tweets to be extracted, the exact tweet extraction code would be the same or extremely similar if a user with higher API access added the exact parameters into Postman.

D) Second Approach (nlp_tweet_practice)

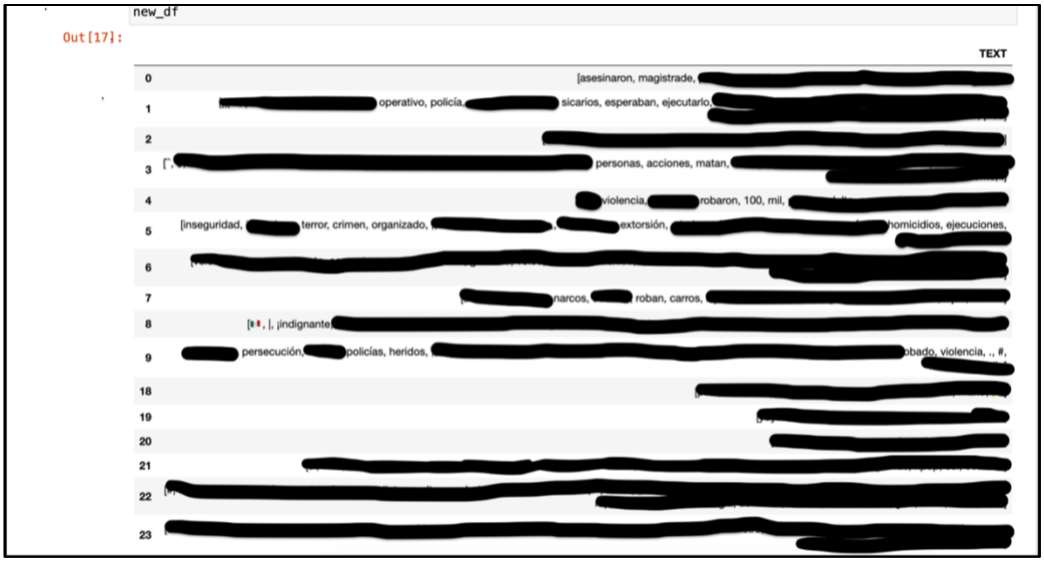
In this pull request, I attempted to recreate the mass tweet search functionality that is granted for higher API levels, but with more unpredictable factors. To clarify, with higher API levels a user can easily conduct a historical mass tweet search with geo location, language, and hashtag conditions by adding such filters on postman. However, for this example, I manually searched for tweets both in the English and Spanish language in order to simulate a scenario where a user might also scrap unwanted data. For the English tweets, they are completely random both in topics and dates. However, for the Spanish tweets, the majority are based on targeted/trigger words associated with crime and violence in Mexico, while the other portion of Spanish tweets are random both in topics and dates.



I added 24 tweets in the ids section. I also added a “created_at” parameter for tweet.fields. I did not use any other parameters because this example is strictly for discarding unwanted language data, and how to use the Natural Language Toolkit (NLTK) to analyze text data.

Outcome of Approach 2

After following the steps in Postman and the code in the nlp_tweet_practice notebook, the final output is the following:



NOTE: As mentioned in the nlp_tweet_practice notebook, I do not show the final result as I'm not too sure where the lines of Twitter's Policy are crossed in terms of me showcasing raw text, geocoordinates, and user ID. As a result, I tried my best to show the final result with redacted information, only showing very small samples of tweets in tokenized form and with no filler words.