I used chatpgt for help with creating a map in 4A. I had already done all the steps but I ended up with just a blank grid. I tried multiple things, made multiple changes, but it did not work. I asked chatgpt what might be the issue and it gave me a fix. I got to know that my merged data it was missing coordinates. I cleaned the placenames columns, and got a working map.

Here is the whole conversation with Chatgpt

I am trying to create a map using plotly express as px. I have done all the things that i know. It is generating a map, but the map only has grids with nothing in it. it is giving me the shared image.

***ChatGPT said:***

Thanks for sharing the image — it confirms that your map is rendering as an **empty plot with just gridlines**, which typically means:

🔍 **Likely Issues:**

1. ***Missing or NaN coordinates (latitude / longitude)***

If your merged DataFrame (merged\_df) has null or zero values for coordinates, no markers will show.

✅ ***Fix: Check coordinates with this:***

python CopyEdit

print(merged\_df[['placename', 'latitude', 'longitude']].head(10)) print(merged\_df[['latitude', 'longitude']].isna().sum())

print((merged\_df['latitude'] == 0).sum(), (merged\_df['longitude'] == 0).sum())

1. ***Latitude and longitude columns are strings***

They **must be numeric**. If read as strings, Plotly can't interpret them.

✅ ***Fix: Convert them to floats:***

python CopyEdit

merged\_df['latitude'] = pd.to\_numeric(merged\_df['latitude'], errors='coerce')

merged\_df['longitude'] = pd.to\_numeric(merged\_df['longitude'], errors='coerce')

Then re-run the map code.

1. ***Month format issue in animation\_frame="month"***

Even if the column name is month, values may be invalid (e.g., strings like "January" instead of sortable values like "2024-01").

✅ ***Fix: Check unique values:***

python CopyEdit

print(merged\_df['month'].unique())

You may need to sort or standardize the month column to work well with animation\_frame.

✅ **Next Steps**

Please run the above checks and share the outputs, or simply paste the first 5 rows of your merged\_df using:

python CopyEdit

print(merged\_df.head())

This will help confirm if coordinates and animation frame values are valid. Would you like help cleaning or converting the values if needed?

4o

**You said:**

#### 1\. \*\*Missing or NaN coordinates (`latitude` / `longitude`)\*\* If your merged DataFrame (`merged\_df`) has null or zero values for coordinates, no markers will show.

#### Fix: Check coordinates with this: python CopyEdit `print(merged\_df[['placename', 'latitude', 'longitude']].head(10))

print(merged\_df[['latitude', 'longitude']].isna().sum()) print((merged\_df['latitude'] == 0).sum(), (merged\_df['longitude'] == 0).sum())`

Python 3.13.2 (tags/v3.13.2:4f8bb39, Feb 4 2025, 15:23:48) [MSC v.1942 64 bit (AMD64)] on win32 Type "help", "copyright", "credits" or "license()" for more information. = RESTART:

C:/Users/Haroon Traders/Downloads/FASDH25-portfolio2/Scripts/regex\_map.py

Index(['placename', 'month', 'frequency', 'latitude', 'longitude', 'geonameid', 'name', 'alternatenames'], dtype='object') Empty DataFrame Columns: [placename, latitude, longitude] Index: [] latitude 0 longitude 0 dtype: int64 0 0

***ChatGPT said:***

Thanks for running the check — here's the problem:

**Your merged DataFrame is completely empty.**

This means the merge between freq\_df and coordinates\_df failed — **no matching placename values** exist between the two files after renaming.

🔍 **What’s likely wrong?**

There are **mismatches between placename values** in your two files. These could be caused by:

* Differences in **capitalization**
* Extra **spaces**
* Different **spellings**
* One file using Asciiname, the other using asciiname, but you renamed **asciiname in both**, which may be a mistake.

✅ **Fix in 3 steps:**

1. ***Revert one renaming line — likely this one is wrong:***

python CopyEdit

freq\_df.rename(columns={"asciiname": "placename"}, inplace=True) # likely incorrect

It should probably be:

python CopyEdit

freq\_df.rename(columns={"Asciiname": "placename"}, inplace=True)

1. ***Clean the placename columns in both DataFrames:***

python CopyEdit

freq\_df['placename'] = freq\_df['placename'].str.strip().str.lower() coordinates\_df['placename'] = coordinates\_df['placename'].str.strip().str.lower()

1. ***Try merging again and print the first few rows:***

python CopyEdit

merged\_df = pd.merge(freq\_df, coordinates\_df, on="placename", how="inner")

print(merged\_df.head())

If this doesn't fix it, I can help you inspect the actual values. Would you like to print a

sample of freq\_df['placename'] and coordinates\_df['placename'] to compare them?

**Geocoding:**

For the first Geocoding code, I took help from Chatgpt and from the scripts of Aamna Mam for guidance, I asked Aamna mam regarding taking guidance from her collab space and she said yes for it.

**The Prompt:** Create a gazetteer for the NER places for all the places in the ner\_counts.tsv file, use geocoding to find their coordinates; use “NA” for places for which you did not find coordinates. Write your results to a tsv file in the gazetteer folder called “NER\_gazetteer.tsv”, which will contain 3 columns: placename, latitude and longitude. Save the script that generated the gazetteer as build\_gazetteer.py (or build\_gazetteer.ipynb if you did this on Google Colab). If the ner\_counts.tsv file contains “NA” values, look up the coordinates manually (using Google) and add them to the tsv file. Make a note of all the place names for which you had to look up the coordinates manually in your README file. I am a student of Digital Humanities and i am doing an assignment and i have assigned to do this task in the group.so please guide me out in adapting to do it. dont give me exact solution instead help me out in how to write the code rather than generating the code.

**The Response:**

**Great! Since you're a Digital Humanities student and this is a group assignment, it's excellent that you want to understand how to approach this task rather than just getting the answer. I'll walk you through the process step by step like a mentor, so you and your team can build it together and learn from it.**

**🎯 Goal:**

**Create a gazetteer (a geographic lookup table) from a file of named entities (ner\_counts.tsv) by geocoding the place names into latitude and longitude, then save the results in a new TSV file.**

**✅ Step-by-step Plan:**

**1. Understand Your Input File:**

**You are given a file called ner\_counts.tsv. This file likely contains named entities (places), possibly along with some metadata like frequency counts.**

**Action:  
Open the file and inspect it:**

**tsv**

**CopyEdit**

**PlaceName Count**

**London 12**

**Paris 8**

**...**

**You’ll want to extract the place names only (from the first column).**

**2. Choose a Geocoding Tool:**

**To find latitude and longitude, you'll use a geocoding API. A few options:**

* **Nominatim API (free and doesn't need API key)**
* **Google Maps API (requires an API key)**
* **Geopy library (works with multiple geocoders, including Nominatim)**

**For students: I recommend using the geopy library with Nominatim.**

**3. Read the File and Clean the Data:**

**In your Python script or Jupyter notebook:**

* **Use pandas to load the ner\_counts.tsv**
* **Clean out duplicates**
* **Extract the place names**

**4. Geocode the Places:**

**Use geopy's Nominatim geocoder in a loop to fetch coordinates. Be sure to:**

* **Handle exceptions (e.g. when the place is not found)**
* **Throttle your requests (Nominatim may block you if you’re too fast — use time.sleep() between calls)**

**Store results in a list or DataFrame.**

**5. Save the Output:**

**Write the result to gazetteer/NER\_gazetteer.tsv with 3 columns:**

* **placename**
* **latitude**
* **longitude**

**Use "NA" for places where coordinates couldn't be found.**

**6. Manual Lookups:**

**For any "NA" values:**

* **Open Google Maps manually and find coordinates.**
* **Replace the "NA" entries in your file.**
* **Document the names you fixed manually in a README.md file.**

**🧠 Tips:**

* **Create a small test file with 5-10 place names before running on the full file.**
* **Use progress printouts/logs to see which names failed.**
* **Wrap your geocoding in a function — this makes your code cleaner and reusable.**
* **Use try...except to avoid your script crashing on errors.**

**🛠 Libraries to Import:**

**python**

**CopyEdit**

**import pandas as pd**

**from geopy.geocoders import Nominatim**

**from geopy.exc import GeocoderTimedOut**

**import time**

**Would you like help writing the basic skeleton of this script or notebook (no actual geocoding yet, just structure)?**

**4o**

**You said:**

**wil i need make a dictionary?**

**ChatGPT said:**

**Great question — yes, creating a dictionary can help organize your data, especially when you're building up your gazetteer with place names and their coordinates.**

**Let me explain how and why you might use a dictionary in this task.**

**✅ Why Use a Dictionary?**

**A dictionary is useful for mapping each place name to its coordinates (latitude and longitude), like this:**

**python**

**CopyEdit**

**gazetteer = {**

**"London": (51.5074, -0.1278),**

**"Paris": (48.8566, 2.3522),**

**"Karachi": ("NA", "NA"), # If not found**

**}**

**This makes it easy to:**

* **Look up coordinates later**
* **Update values (like when you manually fix "NA")**
* **Convert to a pandas DataFrame for saving as a TSV**

**✅ When to Use It in Your Script**

**Here's where in your script a dictionary fits well:**

1. **After loading the file, you loop over each place name.**
2. **For each name:**
   * **Try to get coordinates**
   * **Store the result in the dictionary like: dict[placename] = (lat, lon)**
3. **After the loop, convert the dictionary to a DataFrame and save to TSV.**

**💡 Alternative**

**If you’re more comfortable with pandas, you can also build a list of rows (each row is a [place, lat, lon]) and then turn that into a DataFrame. But using a dictionary is great for clarity and mapping.**

**🔁 Bonus: You Can Reuse It**

**Let’s say you later geocode only the "NA" entries again — having a dictionary lets you update only the missing ones without touching the rest.**

**Would you like a template showing how to use a dictionary in this context?**

**Prompt**: No, Thank you I got it. I will ask you if i get stuck somewhere

**Response:** Perfect — sounds like you're on the right track! 👍  
I'm here whenever you need help — whether it's debugging, testing, or finalizing your README.

Good luck with your assignment and enjoy building your gazetteer!