# Dilawaiz Didar

**You said:**  
I have the ner\_counts.tsv file ready. I want to get coordinates for each place name. What’s the best way to do that using the GeoNames API like we did in class?

**ChatGPT said:**  
You're on the right track.  
Use the requests library to query the GeoNames API and get JSON responses.  
Your class code uses a function like this:

python

CopyEdit

def get\_coordinates(place, username="your\_username"):

url = "http://api.geonames.org/searchJSON?"

params = {"q": place, "username": username, "maxRows": 1}

response = requests.get(url, params=params)

...

Just make sure to replace "your\_username" with your GeoNames account name.

**You said:**  
How do I avoid making too many requests and getting blocked?

**ChatGPT said:**  
Good thinking! To avoid getting rate-limited, include a time.sleep() delay between requests — usually 1 second is enough:

python

CopyEdit

time.sleep(1)

Add it right before or after your requests.get() call.

**You said:**  
Can I test the geocoding function on just one or two place names first?

**ChatGPT said:**  
Absolutely! Before applying it to all place names, test your function like this:

python

CopyEdit

print(get\_coordinates("Gaza"))

print(get\_coordinates("Jerusalem"))

This helps confirm that your API key is working and the output format is correct.

**You said:**  
What if the API doesn’t return anything for a place? Like “Gaza’s” or “Al-Quds”?

**ChatGPT said:**  
That happens often with messy data.  
Wrap your result extraction in a try-except block, and return "NA" if no result is found:

python

CopyEdit

try:

result = results["geonames"][0]

return {"latitude": result["lat"], "longitude": result["lng"]}

except (IndexError, KeyError):

return {"latitude": "NA", "longitude": "NA"}

**You said:**  
Should I clean the place names like we did before NER — e.g., remove 's, the, etc. — before geocoding?

**ChatGPT said:**  
Yes — cleaning is essential.  
Use the same logic from task 2B:

* Remove possessives like 's or ’s
* Strip punctuation  
  That way, "Gaza’s" becomes "Gaza" and "The West Bank" becomes "West Bank" — improving geocoding accuracy.

**You said:**  
I read the place names from ner\_counts.tsv. How do I write the output to a new .tsv file with lat/lon?

**ChatGPT said:**  
Open an output file in write mode, write a header, and for each place name, write a row like this:

python

CopyEdit

with open(output\_file, "w", encoding="utf-8") as out\_file:

out\_file.write("Name\tLatitude\tLongitude\n")

...

Then write each result as a tab-separated string:

python

CopyEdit

out\_file.write(f"{place}\t{lat}\t{lon}\n")

**You said:**  
Can I call the output file NER\_gazetteer.tsv and save it in a folder?

**ChatGPT said:**  
Yes, that’s exactly what your instruction asks.  
Make sure you:

* Name the file: NER\_gazetteer.tsv
* Place it in the gazetteers/ folder (you may have to create this folder in your repo if it doesn’t exist)

**You said:**  
After running, I still have some “NA” results. What do I do with them?

**ChatGPT said:**  
Check those place names manually using Google Maps or Wikipedia.  
Once you find their correct coordinates:

* Open NER\_gazetteer.tsv
* Replace the "NA" values with the correct latitude and longitude  
  Also, in your **README**, include a section like:

markdown

CopyEdit

### Manual Geocoding Notes

These places were geocoded manually:

- Al-Quds → 31.7683, 35.2137

- Gaza’s → 31.5017, 34.4668

**You said:**  
Should I document this gazetteer code in the repo?

**ChatGPT said:**  
Yes.  
If you’re using Colab, save it as build\_gazetteer.ipynb.  
If using Python locally, save it as build\_gazetteer.py.  
Either way, commit the file to your GitHub repo under the correct folder and reference it in your README.

**You said:**  
how to save map as HTML or PNG. Is it okay to use fig.write\_html() and fig.write\_image()?

**ChatGPT said:**  
Yes — these two methods are part of Plotly Express, just not shown in class slides.  
They allow you to **export interactive maps** to .html (clickable) and .png (static image).  
This makes it easy to include the map in reports or GitHub documentation.

**You said:**  
One more thing I used "size='Count'" to show place frequency. Is this the correct column to map size by?

**ChatGPT said:**  
Yes, absolutely.  
By mapping size="Count", you visually scale each point based on how often the place was mentioned — this is exactly what you want for a frequency-based map.

**You said:**  
I used "color='Place'" too,Is that okay?

**ChatGPT said:**  
Yes — it’s a visual choice to **differentiate place names by color**, which is helpful for distinguishing between many points. While not required, it's totally fine if it helps interpret your map clearly.