AIM : Implement Part-of-Speech (POS) tagging and Named Entity Recognition (NER) using spaCy.

Description :

This program demonstrates the implementation of **Natural Language Processing (NLP)** techniques—**Part-of-Speech (POS) tagging** and **Named Entity Recognition (NER)**—using the spaCy library, one of the most popular and efficient NLP tools in Python.

**Part-of-Speech (POS) Tagging:**

POS tagging is the process of labeling each word in a sentence with its appropriate grammatical role, such as noun, verb, adjective, etc. It helps in understanding the structure and meaning of sentences.

#### ****Named Entity Recognition (NER)****:

NER identifies and classifies named entities in text into predefined categories such as **persons, organizations, locations, dates, and more**. This is useful for extracting meaningful information from large bodies of text.

#### ****How it works****:

1. The user provides an input English sentence.
2. The spaCy NLP pipeline processes the sentence.
3. Each token is labeled with its POS tag.
4. Entities are recognized and classified into relevant categories.

This implementation leverages the **en\_core\_web\_sm** model provided by spaCy, which includes all necessary components for POS tagging and NER.

**COMMAND PROMPT INSTALL**

pip install spacy

pip install spacy pymupdf

pip install --upgrade spacy

pip show spacy

python -m venv venv

pip install nltk

python -m spacy download en\_core\_web\_sm

python -m spacy info

pip install spacy pymupdf pandas openpyxl

**PROGRAM**

import spacy

import fitz # PyMuPDF

import pandas as pd

# Load spaCy English model

nlp = spacy.load("en\_core\_web\_sm")

# Function to extract text from PDF

def extract\_text\_from\_pdf(pdf\_path):

text = ""

with fitz.open(pdf\_path) as doc:

for page in doc:

text += page.get\_text()

return text

# Function to perform POS tagging

def perform\_pos\_tagging(doc):

pos\_data = []

for token in doc:

pos\_data.append({

"Text": token.text,

"POS": token.pos\_,

"Tag": token.tag\_,

"Description": spacy.explain(token.tag\_)

})

return pos\_data

# Function to perform NER

def perform\_ner(doc):

ner\_data = []

for ent in doc.ents:

ner\_data.append({

"Entity": ent.text,

"Label": ent.label\_,

"Description": spacy.explain(ent.label\_)

})

return ner\_data

# === Main execution ===

if \_\_name\_\_ == "\_\_main\_\_":

# Path to your input PDF

**pdf\_path = " D:\surekha\Final\Sample.pdf" # Replace with your file path**

# Extract text

text = extract\_text\_from\_pdf(pdf\_path)

# Analyze text with spaCy

doc = nlp(text)

# Get POS tagging and NER results

pos\_results = perform\_pos\_tagging(doc)

ner\_results = perform\_ner(doc)

# Convert to DataFrames

df\_pos = pd.DataFrame(pos\_results)

df\_ner = pd.DataFrame(ner\_results)

# Export to Excel

output\_excel = "nlp\_output.xlsx"

with pd.ExcelWriter(output\_excel) as writer:

df\_pos.to\_excel(writer, sheet\_name="POS\_Tagging", index=False)

df\_ner.to\_excel(writer, sheet\_name="NER", index=False)

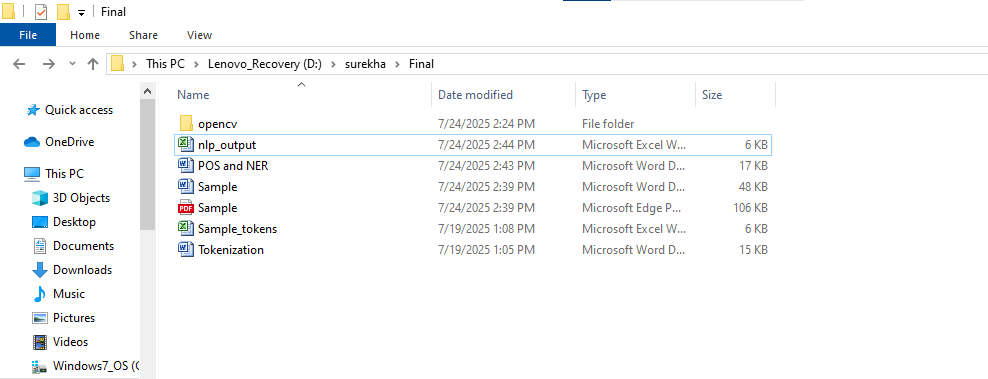
print(f"Analysis complete. Results saved to '{output\_excel}'")

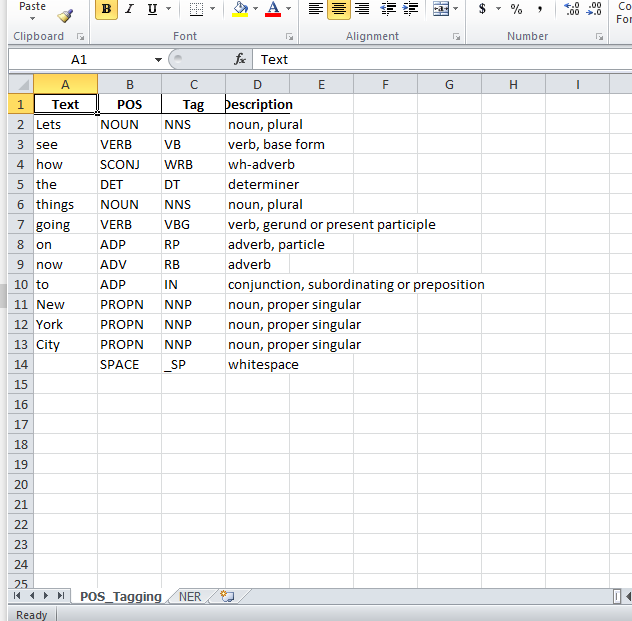
**OUTPUT**

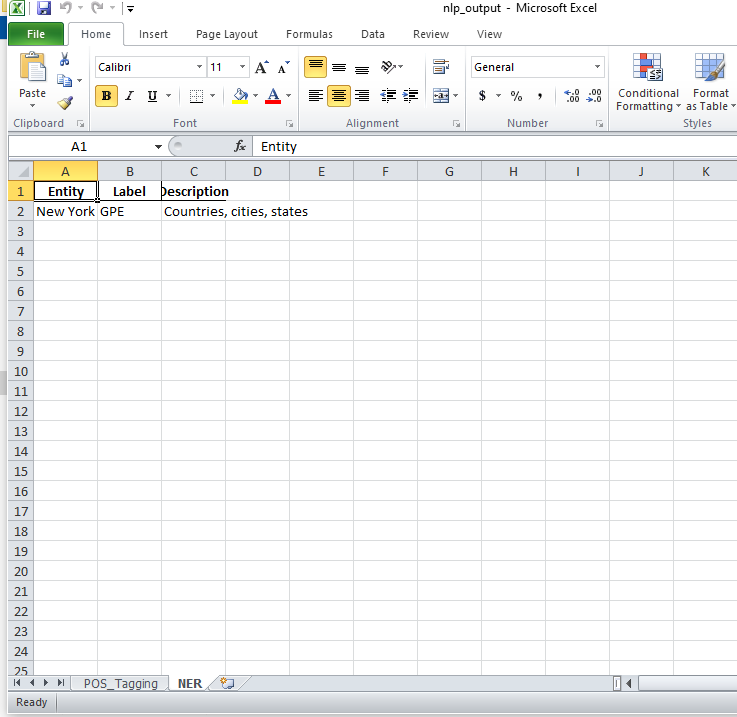
C:\Users\student\PycharmProjects\PythonProject\.venv\Scripts\python.exe C:\Users\student\PycharmProjects\PythonProject\.venv\opencv.py

Analysis complete. Results saved to 'D:\surekha\Final\nlp\_output.xlsx'

Process finished with exit code 0







**RESULT**

Thus, we implemented Part-of-Speech (POS) tagging and Named Entity Recognition (NER) using spaCy. successfully.