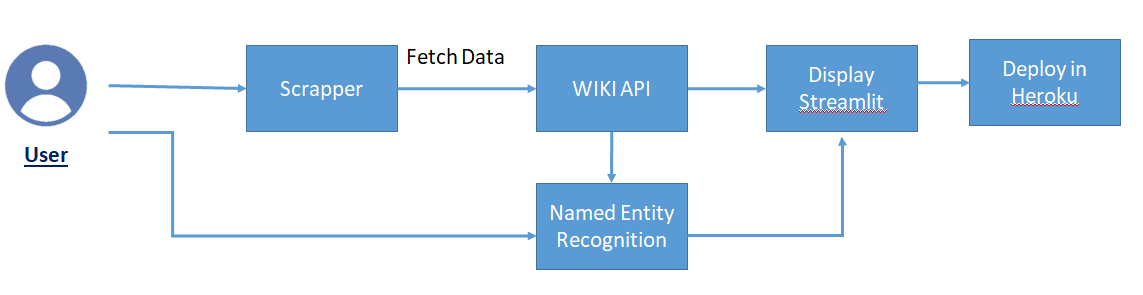
**Introduction:**

Named Entity Recognition is a process where an algorithm takes a sentence or paragraph of text as input and identifies relevant nouns (people, places, and organizations) that are present in that paragraph.

The main aim of this project is to build a web scrapper and Named Entity Recognizer. The goal of the scrapper is to create a web scraper in Python using Streamlit that will scrape Wikipedia pages and get the summary of the input text given and Named Entity Recogniser is to extract the entities(people, places, and organizations) that are present in that paragraph when paragraph(summary of the scrapper) is given as input to NER.

**Technical Architecture:**



**Pre requisites:**

**To complete this project, you must require following software’s,  concepts and packages**

* **Anaconda navigator:**
  + Refer to the link below to download anaconda navigator
  + [Installation video](https://www.youtube.com/watch?v=5mDYijMfSzs)
* **Pythonpackages:**
* **Streamlit –** It is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.

[Streamlit](https://docs.streamlit.io/en/stable/#:~:text=Streamlit%20is%20an%20open%2Dsource,apps%20%2D%20so%20let's%20get%20started!)

* **Displacy:** If you're running a Jupyter notebook,displaCy will detect this and return the markup in a format ready to be rendered and exported. The quickest way to visualize Doc is to use displacy

[Displacy](https://spacy.io/usage/visualizers)

* **Spacy:** spaCy is a free, open-source library for NLP in Python. It's written in Cython and is designed to build information extraction or natural language understanding systems.

[Spacy](https://spacy.io/)

* **Wikipedia-API -** It is easy to use the Python wrapper for [Wikipedias’](https://www.mediawiki.org/wiki/API:Main_page) API. It supports extracting texts, sections, links, categories, translations, etc from Wikipedia. Documentation provides code snippets for the most common use cases.

* **Flask:**  Flask is a popular Python web framework, meaning it is a third-party Python library used for developing web applications.

[Flask Basics](https://www.youtube.com/watch?v=lj4I_CvBnt0)

**Required installation:**

Steps:

1. create a anaconda environment using the command

**conda create -n yourenvname python= pythonversion anaconda**

**ex:** conda create -n ner python= 3.7.4 anaconda

https://lh4.googleusercontent.com/8-8_sR8XrpOMnxq0M3tnzksGxkQfuMN0qQEmaXrNDFmLR-9z4VotrhmruZ80X-bE_AQKduBJSFCQpsZQDR7FTnp8s317jFim4vkTJY-h-ecD-bt6S1bd5pIw2Zs9NriToT8wS4dB

1. Activate your environment

**conda activate ner**

**https://lh4.googleusercontent.com/4h-roopPMxGJhDu5-1Ne6g3pxWjcLAz8KrsKE15acafwckEmcz8AJO9LbZkwYGZ533gADlZdYguwIddaz6J6GEc-F3oOdX3HPdUCLeQB6hO904iMwVph9eWMgGU2H5n5JeJU1fXq**

1. Once the environment is created you need to install all the below packaged:
   1. **pip install wikipedia**
   2. **pip install streamlit**
   3. **conda install -c conda-forge spacy**
   4. **python -m spacy download en\_core\_web\_sm**

(or)

1. Follow the same steps till 2 after that run the below command:

**pip install -r requirements.txt**

**Project Flow:**

* User interacts with the UI (User Interface)  to give the input
* Depending on the input our Wikipedia scraper will scrape Wikipedia pages and get the summary of the input text given.
* Finally the scrapped output will be fed as input to our NER model and it will extract the entities such as people, organisation etc., present in the the scrapped input.

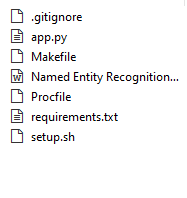
To accomplish this, we have to complete all the activities and tasks listed below

* Named Entity Recognition
* Import the necessary Libraries
* Loading the wrapper
* Creation of Scrapping and NER
* Running the model

* Deploying the application on Heroku Platform

**Project Structure:**

Create a Project folder which contains files as shown below



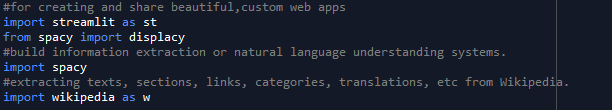
* **app.py :** is our main file where all the important codes related to our projects are present.
* **Procfile :** A Procfile is a mechanism for declaring what commands are run by your application's containers on the Deis platform. It follows the process model.
* **requirements.txt :** This file will contain all the packages  which are used for our project.
* **setup.sh :** this file will be used for implementing our streamlite app with the heroku platform.

**Named Entity Recognition**

Named entity recognition (NER) helps you easily identify the key elements in a text, like names of people, places, brands, monetary values, and more. Extracting the main entities in a text helps sort unstructured data and detect important information, which is crucial if you have to deal with large datasets.

**Activity 1 : Importing the Libraries**

For Implementing our project the first and foremost step is to Importing the necessary libraries

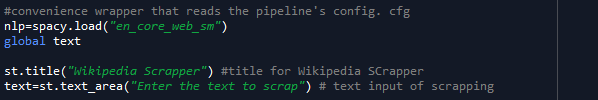


after implementing all the libraries now it's time to start with our project coding.

**Activity 2:Loading the wrapper**

As we will be giving the text as an input to our model so for understanding that we will use a wrapper that reads the pipeline's config. cfg , uses the language and pipeline information to construct a Language object, loads in the model data and weights, and returns it.

Along with the tokenization we are providing the input field for our scrapper.

****

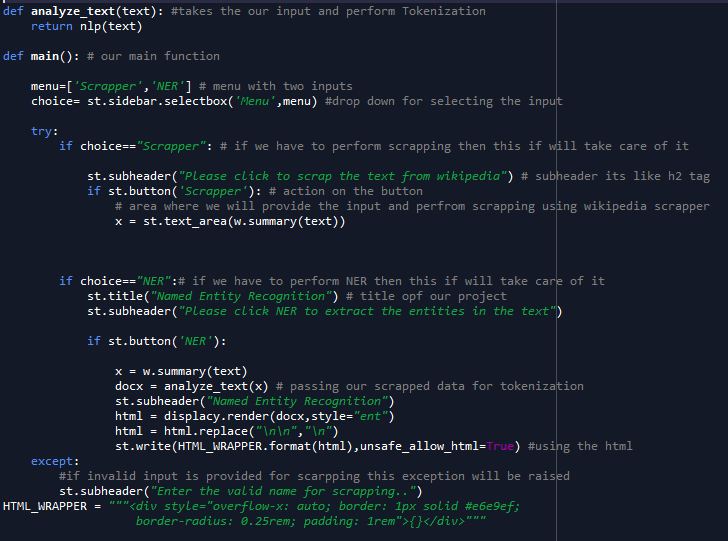
**Activity 3**: **Creation of Scrapping and NER**

After all the importing along with the text as input are done now it’s time to create our main function which will contain both Wikipedia and NER operations.

Firstly we will be creating a select box by using which we will able select the two operations and then we are creating two choices one for scrapper and NER functions.

When it goes inside the scrapper condition there we are performing the scraping the input summary using the wikipedia api. If the input which we provided for scrapping is not valid we will raise an exception.

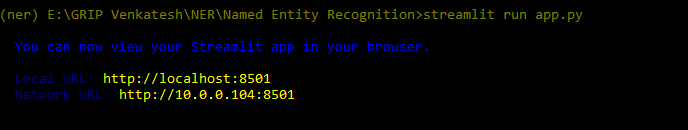
After we found the summary of our input then we will be performing NER by checking whether the select box is chosen  once the NER option is selected we will be passing our summary text for tokenization and displacy we will be showcasing our result in streamlit and also wrapping the html page for some beautification of identified entity.

****

**Activity 3: Running the application**

 Finally it's time to run the application in your local system which we build using streamlit. For running your application you need to open your anaconda prompt and set the path where your files are present then we need to activate our environment which we discussed earlier and finally run the command

**streamlit run app.py**

****

**Deploying the application on Heroku**

As we run our streamlit app in our local system now its time to deploy our application in heroku which is a platform as a service (PaaS) allowing you to run applications written in multiple different programming languages – including Python and Ruby – on the cloud.

Below are some required files which we have to create before the deploying our model in heroku.

**Requirements.txt**

Firstle we’ll be generating a requirement.txt file that will auto-generate a requirements file based on your codebase. To make this super simple we’ll use pipreqs. To start, let’s install it

**pip install pipreqs**

then,

**pipreqss**

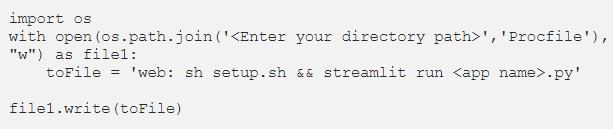
**setup.sh and Procfile**

setup.sh file



Next , we’ll set up the Procfile, run the code below and this will create the file we need without the extension (replace the directory path and your .py file name). If the Procfile has a .txt extension, Heroku will throw some errors your way.

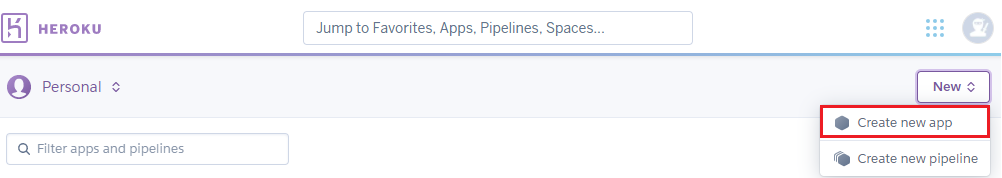
For more information on what the Procfile is and the process types/commands, you can check out their docs [here](https://devcenter.heroku.com/articles/procfile).



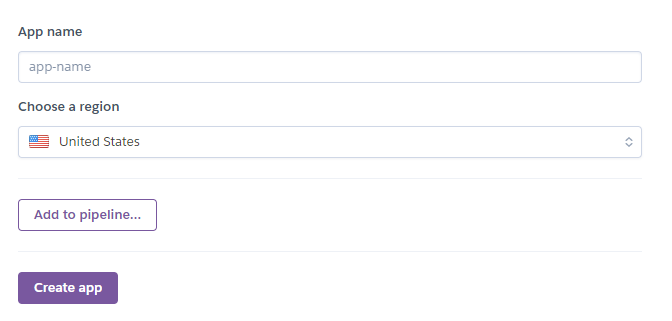
After all the files are ready now you need to follow the following steps:

Login to the [heroku](https://dashboard.heroku.com/) platform

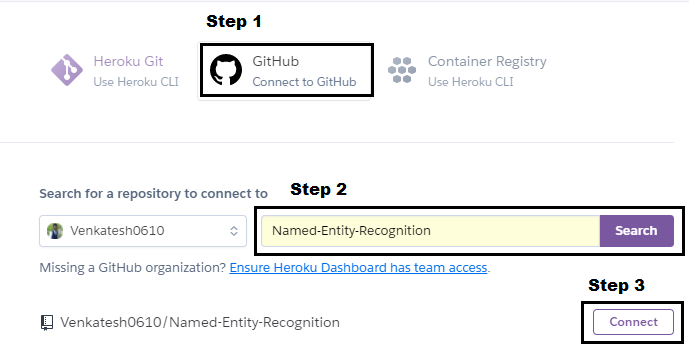
Click on **New** then **Create new app**

****

Provide the name of the app and click on **Create app**

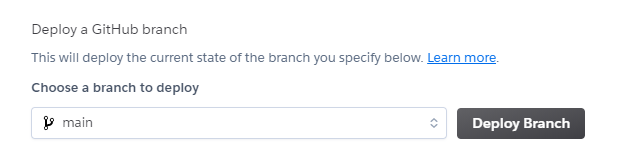
****

Click on Github icon and then add you repository name correctly and click on connect



       And finally go down and click on the Deploy it will take some time and after the

    deployment is done and you will get your deployed link.



**Final Output:**

**URL:** [**https://namedentityrecognition.herokuapp.com/**](https://namedentityrecognition.herokuapp.com/)

[**Output Video**](https://drive.google.com/file/d/1JVjT73CoJAcvjXmq3UCncdhqV81DBMoY/view?usp=sharing)