1. Write a program in Python to scrape about 100 web pages from a single health-related website. For each page calculate at least 5 NLP-related statistics and store all the results in a file.

2. Create one more file named aggregated\_results (1 record) which is the average of the 100 statistics.

3. Create a simple local web application in Java using a web framework like Spring/Play containing one or more web pages. It should take input of a text file or text in the request and then generate nlp statistics of the text similar to the earlier task.

4. Return the comparison of the statistics between the input text and statistics from the earlier task(aggregated\_results) in a txt file or on another web page ( in a table format)

Some examples of statistics are: 1) the number of sentences, 2) the number of words, 3) the number of nouns (anything with a part-of-speech that starts with NN)

Write your code assuming that it will be used as part of a larger project and that someone else may also be calling your methods/classes.

To submit:

- create a public github repo.

- Make sure to include any external jars (or a MAVEN file) in your repo

- Include a README in markup that outlines what you did.

- Follow the best practices of code development and reproducibility. Code should be reproducible on any system

- If possible record a demo and add that in the GitHub repo as well.

Above is the requirements of the task. Write me a professional read me file using the details that are present below:

For Tasks 1 and 2, IDE – Visual Studio and Python version used is 3.11. Packages used include

1. Requests – \*\*fill its use here\*\*
2. Bs4 – \*\*fill its use here\*\*
3. Spacy – \*\*fill its use here\*\*
4. Textblob – \*\*fill its use here\*\*
5. Nltk – \*\*fill its use here\*\*
6. Pandas – \*\*fill its use here\*\*

The program accesses the website <https://www.healthline.com/directory/topics> and extracts href links from <a> tags with the specified class 'css-1hacg05' which is stored in a list. It then visits each of the page in that list and web scrapes data from <p> tags. The NLP related stats that the program then performs on the <p> tag data are that it calculates the number of words, sentences, verbs, common noun (both singular and plural), proper noun(both singular and plural) and type token ratio. Type-token ratio (TTR) is a measure used in linguistics and text analysis to determine the diversity of words in a piece of text. A higher TTR indicates a more diverse vocabulary, while a lower TTR suggests that a smaller set of words is being repeated frequently in the text. These stats are temporarily stored in a list of Maps. After all the 100 web pages are scraped. It is converted to a dataframe and then stored as a csv file named 'nlp\_statistics.csv' into the machine. The mean of the 100 stats in the dataframe is taken and stored in a csv file named 'aggregated\_results.csv' which is also stored in the machine.

For Tasks 3 and 4, IDE used - Spring Tool Suite 4 and Project Configurations are as follows

1. Sprint Boot v3.2.2 web Application, Thymeleaf html template and spring Web
2. Java version: JDK 17
3. Type: Maven
4. Packaging: Jar
5. stanford core nlp java package 4.2.0

The Spring Boot Applications runs on Tomcat started on port 8080 (http). It renders 2 web pages namely index.html and comparison.html. Index.html uses a radio button to take input either as a file or from the text box. It generates NLP related statistics similar to the Python code. The program then reads the aggregated\_results.csv from the machine and stores it temporarily. (The File Path is given explicitly. You can change the file location in nlp-stats-app-2 => src/main/java => com.example.demo.service => NLPService.java In NLPService.java, change the file path present in the function readAggregatedResults()). A comparison between the spring boot app generated NLP and NLP stats from the aggregated\_results.csv is made and shown in a table format in the comparison.html webpage.