



# Python Project Synopsis



## Web Scraper with Data Cleaning and Analysis

### Objective:

The primary goal of this project is to develop a versatile Python-based web scraper that extracts tabular data from websites, performs fundamental data cleaning and analysis operations, and offers an interactive user interface. The project aims to serve as a foundation for future enhancements, including the addition of more advanced cleaning and analytical operations and a transition from the command-line interface (CLI) to a graphical user interface (GUI).

### Key Features:

#### 1) Data Retrieval:

- ❖ Users can choose to scrape data from a specified URL, read from a local CSV file, or read from a local Excel file.

#### 2) Web Scraping:

- ❖ Utilizes the `requests` library to send HTTP requests to the specified URL.
- ❖ Utilizes `BeautifulSoup` for HTML parsing and table extraction.

#### 3) Data Cleaning and Analysis:

- ❖ Users can choose from various operations:
  - 🔗 **Remove Duplicates:** Removes duplicate rows based on the specified column.
  - 🔗 **Handle Missing Values:** Removes rows with missing values in the specified column.
  - 🔗 **Filter Data:** Filters rows based on a specified threshold value in a given column.
  - 🔗 **Aggregate Data:** Calculates aggregate statistics for a specified numeric column.
  - 🔗 **Visualize Data:** Generates histograms for a specified numeric column.

#### 4) Export Data:

- ❖ Users can export the cleaned and analysed DataFrame to a local CSV file.

#### 5) User Interface:

- ❖ Provides a user-friendly command-line interface with interactive prompts.
- ❖ Displays the current state of the DataFrame after each operation.
- ❖ Allows the user to perform multiple operations in a single session.

### Usage:

- 1) Users input the data source (web, CSV, or Excel).
- 2) For web scraping, users provide the URL, and for local files, they provide the file path.
- 3) Users can choose from a menu of operations and provide necessary inputs (column names, threshold values, etc.).
- 4) The DataFrame is updated after each operation, and users have the option to export the final DataFrame to a CSV file.
- 5) The program provides a clean and intuitive interface for users to interact with the data.

## Future Enhancements:





### 1. Advanced Operations:

- ❖ Additional data cleaning and analytical operations will be incorporated to provide users with a broader set of tools for data manipulation and exploration.

### 2. Graphical User Interface (GUI):

- ❖ The command-line interface will be replaced with a graphical user interface for a more intuitive and visually appealing user experience.
- ❖ The GUI will enhance user interaction, making it accessible to users with varying levels of technical expertise.

## Python Libraries Used:

-  ``requests`` for making HTTP requests.
-  ``BeautifulSoup`` for HTML parsing.
-  ``pandas`` for data manipulation and analysis.
-  ``matplotlib`` for data visualization.

## Ethical Considerations:

- ✓ Users should respect the terms of service of the website being scraped
- ✓ Ensure legal and ethical use of data.
- ✓ Responsible web scraping practices are encouraged.

## Conclusion:

This project provides a practical tool for users to scrape and analyse tabular data from the web or local files, facilitating data-driven decision-making and insights. The user interface enhances accessibility and user experience, making it suitable for both beginners and intermediate users.

**Submitted by** : Rushabh Sanghvi

**Class** : SY-BSC-DS

**Roll No.** : 44

**Submitted on** : 15-03-2025