

Python Development

Intern’s

# “WE SPEAK DATA”

**Task 1: Python Script for Web Scraping and Automation**

**Retrieving Camera Information from Flipkart Website**

In the first task of this project, I implemented a web scraping approach to gather details about cameras available on the Flipkart website. After entering the Flipkart URL, I selected the following key details for each camera from the loaded webpage:

1. Camera Name

2. Price

3. Features

4. Effective Pixels

5. Sensor Type

6. Warranty Information

Subsequently, this selected data was organized into a structured format by creating a data frame. This data frame served as the foundation for further analysis and processing in the subsequent tasks of the project.

In subsequent steps, each attribute of the selected data, including camera name, price, features, effective pixels, sensor type, and warranty, underwent a meticulous cleaning process. This ensured that the final dataset was refined and ready for downstream analysis and utilization.

This methodology allowed for the extraction of meaningful insights from the camera data, facilitating a more efficient and informed decision-making process for users interested in purchasing cameras from Flipkart.

url= "https://www.flipkart.com/search?q=camera&as=on&as-show=on&otracker=AS\_Query\_OrganicAutoSuggest\_4\_6\_na\_na\_na&otracker1=AS\_Query\_OrganicAutoSuggest\_4\_6\_na\_na\_na&as-pos=4&as-type=RECENT&suggestionId=camera&requestId=c47f1865-d7a5-45c2-94d3-944b5486d93f&as-searchtext=camera"

**1. Import Necessary Libraries:**

- Use `pip install` commands to install the required libraries: `bs4` and `requests`.

- Import the necessary libraries such as `BeautifulSoup`, `requests`, `pandas`, `BackgroundScheduler`, and `IntervalTrigger`.

**2. Define Scraping Function:**

- Define a function `scrape\_flipkart\_camera\_details` to scrape camera details from the Flipkart website.

- This function includes scraping information such as camera names, prices, features, effective pixels, sensor types, resolutions, and warranties.

**3. URL and Request:**

- Define the URL of the Flipkart camera search page.

- Use the `requests.get()` method to make an HTTP request to the specified URL.

**4. BeautifulSoup Parsing:**

- Use `BeautifulSoup` to parse the HTML content of the page.

- Extract camera details using CSS selectors for elements like name, price, features, etc.

**5. Create DataFrame:**

- Create a Pandas DataFrame (`camera\_dataset`) to store the extracted camera details.

- Use Series with NaN values and set a common index to ensure equal lengths for each column.

**6. Excel File Creation:**

- Use `pd.ExcelWriter` to create an Excel file (`Cameraflip.xlsx`) and write the DataFrame to a sheet named 'sheet1'.

- Save the Excel file.

**7. Data Cleaning:**

- Load the data from the created Excel file (`Cameraflip.xlsx`) using `pd.read\_excel`.

- Clean the data by removing unwanted characters and formatting issues in columns such as features, camera\_name, effective\_pixels, sensor\_type, resolution, and warranty.

**8. Cleaned Data Excel File:**

- Save the cleaned data to a new Excel file (`Cleaned\_Cameraflip.xlsx`).

**9. Automation Setup:**

- Install the `apscheduler` library using `pip install apscheduler`.

- Schedule the scraping job to run every hour using `BackgroundScheduler` and `IntervalTrigger`.

**10. Start Scheduler:**

- Start the scheduler using `scheduler.start()` to automate the scraping job.

**11. Load Cleaned Data:**

- Load the cleaned data from the Excel file.

**12. Document the Code:**

- Provide inline comments and explanations for each step in the code.

**13. Print Statements:**

- Include print statements for debugging and checking the execution flow.

**14. Output File Paths:**

- Output paths for the original and cleaned data files are displayed in the console.

**15. Documentation:**

- Add comments and explanations to make the code more understandable.

**16. Final Execution:**

- Execute the script, and the scraping job will run every hour automatically.

**Instructions for Using the Script:**

**1. Install Required Libraries:**

- Open your terminal or command prompt.

- Run the following commands to install the required libraries:

```bash

pip install bs4

pip install requests

pip install apscheduler

```

**2. Copy the Script:**

- Copy the entire Python script into a Python environment or a script file (e.g., `.py` file).

**3. Run the Script:**

- Execute the script by running it in a Python environment.

```bash

python your\_script.py

```

**4. Wait for Execution:**

- The script will scrape camera details from the Flipkart website and create an Excel file (`Cameraflip.xlsx`) containing the raw data.

**5. Data Cleaning:**

- The script will then clean the data and save it to a new Excel file (`Cleaned\_Cameraflip.xlsx`).

**6. Automation Setup:**

- The script is configured to run the scraping job every hour using the `apscheduler` library.

- The scheduler will automatically run the scraping job in the background.

**7. Check Console Output:**

- The console output will display the paths of the original and cleaned data files.

- You can find these files in the same directory as the script.

**8. Check Excel Files:**

- Open the Excel files (`Cameraflip.xlsx` and `Cleaned\_Cameraflip.xlsx`) to view the scraped and cleaned data, respectively.

**9. Customize Interval (Optional):**

- If you want to customize the scraping interval, modify the `scheduler.add\_job` line with the desired interval in hours.

**10. Stop the Script (Optional):**

- If you want to stop the script and scheduler manually, press `Ctrl + C` in the terminal where the script is running.

**The dependencies for the code are:**

**1. BeautifulSoup (`bs4`):**

- Used for web scraping. It helps parse HTML and extract data from it.

**2. Requests (`requests`):**

- Used for making HTTP requests. In this script, it's used to retrieve the HTML content of the Flipkart website.

**3. Pandas (`pandas`):**

- Used for data manipulation and analysis. The script creates a DataFrame to organize the scraped data and perform cleaning operations.

**4. APScheduler (`apscheduler`):**

- Used for scheduling recurring tasks. In this script, it's employed to automate the scraping job at regular intervals.



T

h

a

n

k

y

o

u

!