\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_URL: <https://www.energystar.gov/productfinder/advanced> \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Process Document: Automated Data Scraping, Mapping, and Comparison

1. Objective:

The objective of this process is to automate the extraction of data from the ENERGY STAR Certified Product Data Sets and APIs webpage, map the raw data according to customer requirements, create formatted UP files, convert them into JSON files, and compare them with the previous month's UP files to identify any new columns.

2. Steps:

Step 1: Data Scraping:

Utilize Python and relevant libraries (e.g., BeautifulSoup, requests) to scrape data from the ENERGY STAR webpage.

Identify and scrape data under the "Heating & Cooling" header and its 12 subcategories.

Handle cases where direct scraping is not possible by using HTML code.

Store the scraped data in Excel files into the RAW folder which is created by code.

Step 2: Data Mapping

Based on customer requirements, map the raw Excel files to create formatted UP files.

Also map the raw excel file by sing brands mapping and also map-api to create formatted files

Theses formatted files are stored in the FORMATTED folder.

Perform necessary data manipulation, cleaning, and formatting during the mapping process.

Step 3: JSON File Creation

Convert the formatted UP files into JSON files using Python.

Ensure the JSON files adhere to the specified structure and contain all necessary data fields.

Step 4: Comparison with Previous Month

Retrieve the UP files from the previous month.

Automatically compare the current formatted UP files with the previous month's UP files.

Identify any new columns added in the current files compared to the previous month.

Step 5: Automated Execution

Develop a Python script to automate the entire process.

Use spyder IDE to execute the script, providing all required file paths in the config.txt file as input

Implement error handling and logging mechanisms to ensure smooth execution and troubleshooting.

5. Quality Assurance:

Implement quality control checks at each step of the process to ensure accuracy, completeness, and consistency of the data.

Conduct regular reviews and audits to identify and address any issues or discrepancies.

into the HVAC.rar file:

Into HVAC.rar file having files and folders like as mentioned below:  
1) Scripts

2) HVAC last month

3) brands-mapping.xlsx

4) map.xlsx

5) Map-Api.xlsx

1) Scripts:

In this folder having all the scripts of HVAC process

It having a) **Automate\_\_API.py** python file it shows:  
- The Python script is designed to scrape data related to heating and cooling products from the Energy Star website. It performs web scraping, data processing, and output formatting tasks.

- The script imports the necessary libraries and modules, including:

* lxml for parsing HTML content.
* requests for sending HTTP requests.
* pandas for data manipulation and analysis.
* sodapy for interacting with the Socrata API.
* tqdm for displaying progress bars.
* logging for logging error messages.
* BeautifulSoup from bs4 for HTML parsing.
* re for regular expressions.
* warnings for handling warnings.

-The script loads configuration details from a config.txt file, which includes file paths

- scrape\_all\_pages(url): Scrapes data from all pages of a specified website.

scrape\_central\_air(url): Scrapes data for central air products from a given URL.

Main() : The main function describes the execution flow of the script. It creates a directory for storing raw data, scrapes data from web pages, processes the scraped data, and saves it as raw Excel files. Additionally, it executes other tasks such as formatting, JSON creation, and Excel file comparison.

b) **Automate\_\_formatted.py**

- The script begins by importing necessary libraries such as pandas, re, os, openpyxl, and datetime to facilitate data manipulation and file operations.

- the script creates a directory named "FORMATTED" to store the formatted data files.

- The script iterates through each file in the "RAW" directory, preparing them for formatting.

- For Heat Pumps and Central Air Conditioners:

NA values are replaced with None.

Special characters and extra spaces are removed.

Column names are standardized using mapping.

Brand names are standardized using brand mapping.

Additional columns are added and filled with default values.

Timestamp and SKU are generated based on certain conditions.

Duplicate entries are removed and SKU is modified if necessary.

For other product categories: Similar data cleaning and standardization procedures are applied based on product category.

**Excel Formatting:**

- Excel files undergo formatting enhancements to improve readability and usability.

- Freeze panes are applied to keep headers visible while scrolling through large datasets.

- Auto-filtering is activated to enable quick and efficient data filtering based on specific criteria.

- Text alignment is adjusted to left-align the content in the first row, enhancing visual clarity.

- save all the formatted files into FORMATTED folder

c) **Automate\_JSON.py**

- It first creates a "JSON" directory within the specified output directory (file\_path\_to\_output) to store the JSON files.

- it then iterates through each file in the "FORMATTED" directory and reads it into a pandas DataFrame.

- Depending on the file name, if it contains "Heat Pumps (Ducted)" or "Central Air Conditioners (Ducted)," it processes the data differently from other files.

- For files related to heat pumps or central air conditioners, it extracts specific columns, creates a list of dictionaries, and then writes the data to both JSON and JSONL files.

- For other files, it follows a similar process but includes additional conditional logic to handle various columns and data types.

d) **Comparing\_excel\_up.py**

- This script provides a mechanism to compare Excel files between a "last month" directory and a "formatted" directory.

- It helps identify any new columns added in the formatted files compared to the last month's files.

-This can be useful for tracking changes or updates in data structures over time, particularly when dealing with large datasets or ongoing data collection processes.