*Logo Matching and Grouping Based on Visual Similarity*

The goal of this project was to match and group websites based on the visual similarity of their logos, without using machine learning clustering algorithms. From the start, I focused on simplicity, robustness, and transparency of the solution.

My first step was to extract logos for each domain in the provided ***logos.snappy.parquet*** file. In order to achieve this task, I opted to use the [Clearbit Logo API](https://logo.clearbit.com/) , which provides clean, high-quality logos just by supplying a domain. This allowed for very fast and uniform image retrieval across thousands of sites.

Once I had all the images locally, I needed a reliable way to compare them visually. Since machine learning clustering was not allowed, I used **perceptual hashing (pHash)** via the ***imagehash*** Python library. This method transforms images into hashes that reflect their visual structure. The lower the hash distance between two images, the more visually similar they are.

For the grouping part, I implemented a custom logic that loops through all pairs of images and groups together those whose hash distance is below a certain threshold (I used 10). This ensured that logos that are not exactly the same but look very similar (like dark/light variants) are correctly clustered.

The final output of the script is a JSON file containing clusters of similar logos. Each group is a list of websites whose logos were determined to be visually similar.

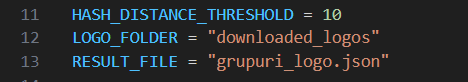
Each logo was saved locally in the downloaded\_logos/ folder for visual inspection, and the output groups were written to grupuri\_logo.json.

The detailed steps in the Python Script:

1. Imports

* pandas – used to read the .parquet file containing domains.
* requests – for sending HTTP requests and downloading logos.
* Pillow (Image) – image processing library
* imagehash – calculates perceptual hashes for image comparison
* BytesIO – lets you open images directly from memory
* urlparse – extracts the base domain from a full URL
* os – handles folders and file operations
* json – for saving the final grouped results
* tqdm – displays progress bars in the terminal

1. Configuration

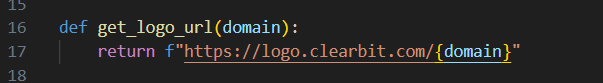


* Sets the threshold under which two logos are considered similar.
* Defines the folder to save the downloaded logos.
* Sets the output filename that will contain the grouped websites.

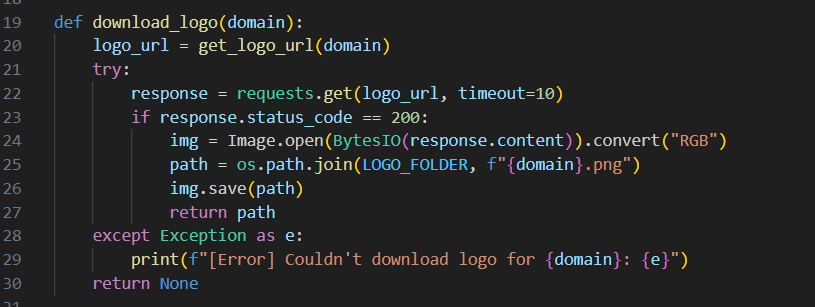
1. Create the image folder



1. **Generate the logo URL using Clearbit**



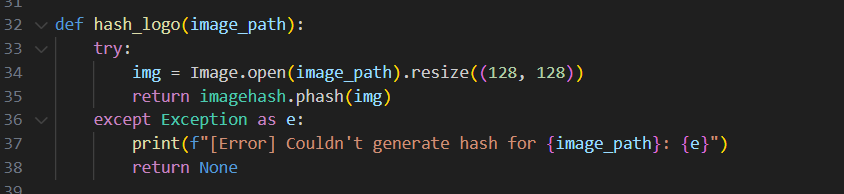
1. Download and save logo locally



* Sends a request to Clearbit.
* If the response is successful (HTTP 200), it:
  1. Opens the image.
  2. Converts it to RGB format.
  3. Saves it to the local folder using the website’s domain as the filename.
* Returns the path to the saved image (or None if failed).

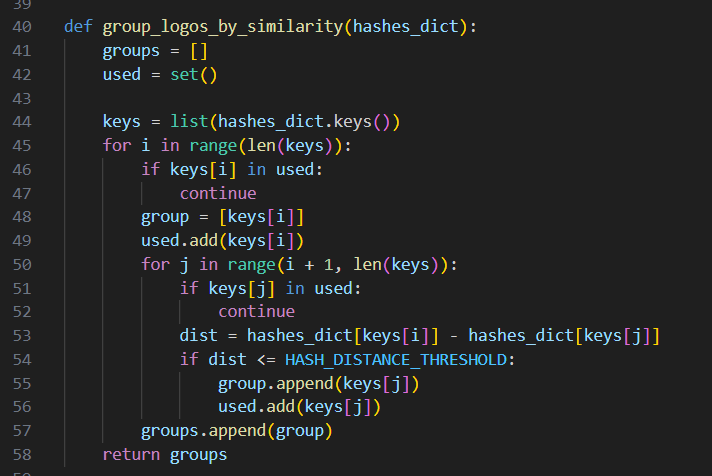
1. Generate perceptual hash for an image

* Opens the image from disk and resizes it to a standard size.
* Computes a **pHash** which is a numeric summary of the image’s visual content.
* Returns the hash (or None on error).



1. Group logos by visual similarity

* Compares all logos with each other using hash distance.
* If the distance between two hashes is small (e.g. ≤ 10), they are considered similar and added to the same group.
* Keeps track of already-used domains so they’re not added to multiple groups.
* And then the Main function



Conclusion:

This project showed that logos can be grouped by visual similarity without using machine learning, by applying perceptual hashing and a custom grouping logic. The method successfully extracted and compared logos, delivering accurate and explainable results using simple yet effective tools.