**✅ High-Level Goal**

Group websites by logo similarity using a custom pipeline that:

* Extracts logos from websites
* Compares logos using multiple features
* Groups logos based on similarity (without classic ML clustering)
* Provides a clear, interpretable presentation of results

**🔧 Tools & Tech Stack**

* **Language**: Python
* **Libraries**:
  + Web scraping: requests, beautifulsoup4, selenium
  + Image processing: Pillow, OpenCV, imagehash
  + Similarity: SSIM, ORB feature matching, perceptual hash
  + Visualization: matplotlib, networkx, Plotly
  + Optionally: Flask or Streamlit for demo UI
* **Optional Enhancements**: Use Docker, Celery, Redis for scaling and concurrency

**✅ Day 1 – Dataset Prep + Logo Extraction**

**1. Parse the logos\_list**

* Accept list of company websites from a file (e.g. CSV or TXT)
* Validate accessibility of the URLs

**2. Extract Logos**

* Crawl homepage and extract <img> tags
* Heuristics to detect the logo:
  + alt contains "logo"
  + filename contains "logo"
  + id or class includes "logo"
  + Largest image near top-left of page
* Fallback: Use headless browser (selenium) to capture a screenshot and crop top-left

**3. Save Logos**

* Save to logos/<domain>.png
* Normalize size (e.g., 200x200), remove transparency
* Create JSON/CSV map of website → logo file

**✅ Day 2 – Logo Similarity Computation**

**4. Feature Extraction**

* Compute:
  + Perceptual Hash (imagehash)
  + SSIM (Structural Similarity Index)
  + ORB features (OpenCV)
  + Color histogram comparison
  + Edge histogram (using Canny edges)

**5. Pairwise Comparison**

* For each pair of logos:
  + Compute similarity scores across features
  + Use rule-based thresholds (no clustering algo!)
    - E.g. SSIM > 0.8 + pHash < 10 + ORB match count > 20 = match
* Store in similarity matrix or edge list

**✅ Day 3 – Grouping + Presentation**

**6. Graph-Based Grouping (Non-ML Clustering)**

* Build a graph where:
  + Nodes = websites
  + Edges = high-similarity pairs
* Use connected components to find groups

**7. Output Results**

* JSON: [{ group\_id: 1, websites: [ ... ] }, ...]
* CSV summary
* Save grouped logos in folders: groups/group\_1/, etc.

**8. Visualization**

* Optional UI (Flask/Streamlit) with:
  + Logo grid per group
  + Hover to see domain names
* Graph visualization (networkx, Plotly) showing logo similarity clusters

**📦 Bonus Ideas (If Time Permits)**

* Add OCR (e.g. Tesseract) to detect text in logos for additional comparison
* Evaluate logo alignment and shape using contour matching
* Add retry/caching for crawling
* Generate HTML report

**📝 Deliverables**

* Codebase (clean, modular, documented)
* README.md with:
  + Problem explanation
  + Architecture
  + Instructions to run
  + Sample results
* Output:
  + JSON/CSV of groups
  + Screenshots / demo UI
  + GitHub link for submission

**📌 Why Build a Custom Pipeline?**

* Full control over every step
* Transparent and explainable results
* Fits the problem tightly (no “black box” clustering)
* Easier to debug, adjust, or scale
* It shows deeper understanding of the problem