Final Objective

Build a system where a student or researcher can type "Who is working on sustainable energy and carbon capture?" and find relevant faculty members, even if those specific phrases aren't in their official department title.

Project 1 Objective

Build a data pipeline to crawl, extract, and clean faculty data (names, bios, research interests) from a college website to provide a clean dataset for a semantic search engine.

Current Role: Data Engineer

Recommended Tools(not mandatory. Feel free to explore others)

Lifecycle Step	Tool Recommended	Potential Sources
Generation	HTML / Web Content	College Faculty Directory pages
Ingestion	`requests`, `BeautifulSoup`, `scrappy`	Scraping the "Source Code" of the site
Storage	`sqlite3`	Relational DB (Tables: Faculty, Research_Tags)
Transformation	`pandas` / `cleantext`	Stripping HTML tags, fixing encoding issues
Serving	`FastAPI`	An API endpoint: `/faculty/{id}` or `/all`

Project Specs & Steps

1. Ingestion (The Scraper)

- **The Task:** Navigate the college directory. Fetch the HTML of individual faculty profiles.
- **Challenge:** Handling pagination (e.g., Page 1, 2, 3) or clicking "Load More."

2. Transformation (The Cleaner)

- **The Task:** Extract specific entities from the messy HTML (e.g., separate the "Bio" section from the "Education" section).
- **Challenge:** Some faculty might have missing bios or weird characters in their names. You must handle these "null" values.

3. Storage (The Structured Home)

- **The Task:** Design a schema.
- **Tool:** Use `sqlite3` to ensure the data persists after the script finishes.

4. Serving (The Hand-off)

- **The Task:** Build a FastAPI route that allows a Data Scientist to "GET" all faculty data as a JSON object so they can begin embedding the text.

Outcomes

1. **Clean Dataset:** A SQLite database or a master CSV/JSON file ready for NLP tasks.
2. **Github Repository:** Clean separation of codebase.
3. **README.md:** Documentation of the "Schema" and instructions on how to run the pipeline.

Scoring Policy

Category	Weightage	Description
Documentation	30%	Clear setup instructions and data dictionary.
Code Quality	35%	Error handling (what if the website is down?) and modularity.
Data Management	35%	Efficient SQL storage and successful cleaning of HTML "noise."

LLM Policy

1. Feel free to use an LLM provided each and every prompt is logged along with the response and the tool used in a markdown file (e.g., `logs/llm_usage.md`).

Future Steps

1. **Switch Role to Data Scientist:** Implement a vector search where you turn the `bio_text` into embeddings and allow "Natural Language" queries against the faculty list.