CLOUDCULATE

Task Description

Organizations spend millions on cloud infrastructure but struggle to find the best cloud architecture solutions for their use case.

You are required to build a cloud-architecture parsing and scraping application using the following technologies:

- **Backend**: Python for parsing, and scraping, and api
- **Database**: MongoDB for storing parsed architecture data
- **Frontend (Bonus)**: Web application (any web technology, React optional)
- **DevOps (Bonus)**: Docker for containerization

Detailed Requirements

1. Architecture Parsing and Scraping with Python

Write a Python module that:

- Scrapes cloud architectures
- Parses the scraped content, extracting relevant resources and data (be creative you might want to use AI)
- Stores the parsed architecture data directly into a MongoDB database

2. Backend Server with Python API (FastAPI)

Create a Python backend API server that:

- Provides endpoints to:
  - Retrieve a list of previously scraped and parsed architectures (with timestamps and metadata)
  - Trigger scraping and parsing
- Runs as a Docker container (bonus)

3. Frontend (Bonus): Web Application

Develop a web application that communicates with the Python backend API.

The frontend should have a user-friendly interface to:

- Trigger scraping and parsing
- Display a list of parsed architectures along with timestamps
- Runs as a Docker container (bonus)

4. Docker and DevOps (Bonus)

- Create Dockerfiles for each component (Python script/API and web frontend)
- Provide a `docker-compose.yml` file that orchestrates MongoDB, the Python backend, and the web frontend
- Document clearly in your README how to set up, build, and run your application locally using Docker Compose

Submission

GitHub repository or ZIP file containing:

- Complete source code
- Dockerfiles and Docker Compose configuration

A comprehensive README including:

- Clear step-by-step instructions for setting up and running the application locally using Docker Compose
- Explanation of the implemented approach and architecture
- Any considerations or trade-offs made during implementation