DEVOPS BACKEND ENGINEER TEST

Question 1: Advanced Web Scraper for Media Data

Objective:
Develop a Python-based web scraper using Scrapy to extract detailed information from IMDb's Top 50 movies list. The scraper should be robust, handling concurrent requests efficiently, and packaged in a Docker container for deployment.

Problem Description:

- The scraper must target the IMDb "Top 50" movies list at this URL: IMDb Top 50 Movies.
- Extract the movie name, year of release, director, and main stars from each movie's detail page linked from the Top 50 list.
- The scraper must manage concurrent requests and comply with IMDb's robots.txt to respect their scraping policies.
- Package the Scrapy scraper within a Docker container to ensure portability and ease of deployment.

Sample Input:

- No initial input required; the scraper starts upon invocation and accesses the specified URL.

Expected Output:

- A JSON file containing an array of movies, with each object including the name, year, director, and main stars of a movie.

Question 2: Deployment Using Docker and Kubernetes

Objective:
Deploy the Dockerized Scrapy scraper developed in Question 1 on a Kubernetes cluster using AWS EKS, ensuring scalability and robustness.

Problem Description:

- Write a Dockerfile to containerize the Scrapy project from Question 1.
- Create Kubernetes manifest files for deploying the Docker container, including a Deployment, Service, and Horizontal Pod Autoscaler to manage load based on CPU usage.
- Utilize AWS EKS for the Kubernetes environment, ensuring the scraper service is scalable and resilient under varying loads.
- Configure Kubernetes Secrets to securely handle sensitive configurations like API keys or database credentials.

Sample Input:

- Dockerfile, Kubernetes YAML configuration files, and necessary AWS CLI commands for resource management.

Expected Output:

- Steps to deploy and scale the web scraper on AWS EKS, including monitoring logs to ensure operational integrity.
- Demonstration of effective load handling and auto-scaling capabilities of the deployed service.

Submission Guidelines:

Submit your code as a ZIP file. Ensure the zip file includes a README with instructions on how to run your application, how you have used each library in the task, and their purpose.

Note: Don’t worry about copyright. If you are not selected, this task is yours to showcase on your GitHub forever.