Prueba Técnica: API de Inteligencia de Noticias

Objetivo del Proyecto:

Diseñar, desarrollar y desplegar un servicio de backend robusto y escalable que ingiera artículos de noticias y ofrezca capacidades de búsqueda y análisis semántico a través de una API RESTful, desplegado en un entorno de Kubernetes.

Parte 1: Desarrollo de la API de Búsqueda

Descripción del Problema:

Se requiere una API que permita a los usuarios buscar artículos de noticias basándose en el significado o la intención de su consulta, en lugar de una simple coincidencia de palabras clave.

Requisitos Funcionales:

- **Ingesta de Datos:** El sistema debe ser capaz de obtener y procesar artículos de noticias de al menos una fuente de datos externa. Deberás gestionar el proceso de obtención y preparación de estos datos para su posterior indexación.
- **Generación de Vectores:** El contenido textual de los artículos (como el título o el cuerpo) debe ser transformado en representaciones vectoriales numéricas (embeddings) que capturen su significado semántico.
- **Almacenamiento y Búsqueda:** Estos vectores deben ser almacenados y gestionados en una base de datos optimizada para búsquedas de similitud a gran escala (una base de datos vectorial). La API debe utilizar esta base de datos para encontrar los artículos más relevantes para una consulta de búsqueda determinada.

Requisitos No Funcionales:

- **Lenguaje:** La implementación debe realizarse en Go o Python.
- **Calidad del Código:** El código debe ser de alta calidad, legible, mantenible y seguir principios de diseño de software sólidos que promuevan la separación de responsabilidades.
- **Pruebas:** El proyecto debe incluir un conjunto de pruebas automáticas que validen la corrección de la lógica de negocio y las integraciones clave del sistema.

Endpoints Mínimos Requeridos:

- **POST /index:** Un endpoint para recibir y procesar nuevos artículos de noticias para su indexación.
- **GET /search:** Un endpoint que acepta una consulta de texto y devuelve una lista de los artículos más relevantes semánticamente, ordenados por su grado de similitud.

Parte 2: Despliegue y Automatización con Kubernetes

Descripción del Problema:

La API debe ser desplegada en un entorno de Kubernetes de manera automatizada, reproducible y segura, siguiendo las mejores prácticas de DevOps.

Requisitos:

- **Plataforma de Despliegue:** El servicio debe ser desplegado en un clúster de **Kubernetes** (por ejemplo, Google Kubernetes Engine - GKE, Amazon EKS, o Azure AKS).
- **Contenerización:** La aplicación debe estar empaquetada en una imagen de contenedor Docker.
- **Infraestructura como Código (IaC):** Toda la infraestructura de la nube necesaria para ejecutar el servicio (incluyendo la configuración del clúster de Kubernetes, nodos, y cualquier otro recurso dependiente) debe ser definida y gestionada mediante Terraform.
- **Manifiestos de Kubernetes:** Deberás crear los manifiestos de Kubernetes necesarios para el despliegue de la aplicación (e.g., Deployment, Service, Ingress, ConfigMaps, Secrets).
- **CI/CD:** Se debe implementar un pipeline de integración y despliegue continuos. Este pipeline debe automatizar el proceso de prueba, construcción de la imagen del contenedor, publicación en un registro de contenedores (como Google Artifact Registry o Docker Hub) y el despliegue en el clúster de Kubernetes cada vez que se realizan cambios en el repositorio de código fuente principal.

Bonus: Endpoints

- **GET /storyline:** Dada una consulta sobre un tema, este endpoint no debe devolver una simple lista de artículos, sino identificar y agrupar los artículos que pertenecen a los mismos sub-eventos o "hilos de noticias". La respuesta debe presentar estos hilos de manera estructurada, idealmente en un orden que refleje su evolución temporal.
- **GET /analysis/perspective:** Este endpoint debe aceptar una consulta y una lista de fuentes de noticias. Su objetivo es realizar un análisis comparativo de cómo esas fuentes cubren el tema. La respuesta debe destacar cuantitativamente las diferencias clave en la cobertura, como podrían ser el tono general o las figuras o conceptos centrales en los que cada fuente se enfoca.

- **GET /graph/entities:** Para una consulta dada, este endpoint debe identificar las principales entidades (como personas, organizaciones o lugares) mencionadas en los artículos relevantes. Luego, debe mapear y devolver las relaciones inferidas entre estas entidades, presentando el resultado en una estructura de grafo (nodos y aristas).