PROCESAMIENTO MASIVO DE DATOS

Luis Mario Ballar Zamora - C20937
Luis Daniel Solano Solano - C17640
Jean Carlo Calderón Rojas - C21509

Agenda

1. Tecnolgías Seleccionadas
2. Puntos Claves para Entender la Tecnología
3. Problema Propuesto
4. Solución Propuesta
5. Resultados Obtenidos
6. Conclusiones

1

Tecnologías Seleccionadas

Python, Kafka, Zookeeper y FastApi

Python

Lenguaje de programación versátil y de alto nivel, ampliamente utilizado en desarrollo backend y procesamiento de datos.

- Compatible con Múltiples Frameworks
- Gran comunidad y Documentación
- Buen Ecosistema de Librerías

Kafka

Plataforma de streaming distribuido y mensajería asincrónica, diseñada para manejar grandes volúmenes de datos en tiempo real.

- Comunicación entre microservicios
- Procesamiento de eventos en tiempo real
- Alta escalabilidad
- Alta tolerancia a fallos

Zookeeper

Servicio centralizado para coordinar y mantener información de configuración distribuida entre sistemas.

- Gestión de configuración centralizada
- Elección de líder (Leader Election)
- Sincronización de servicios
- Registro de servicio

FastAPI

Framework moderno y rápido para construir APIs con Python. Framework de backend para desarrollo de APIs RESTful.

- Alto rendimiento (basado en ASGI y Starlette)
- Validación automática con Pydantic
- Documentación interactiva (Swagger y ReDoc)
- Código limpio y tipado (soportepara type hints)

2

Puntos Claves para Entender las Tecnologías

¿Por Qué y Qué?

¿Por Qué Se Eligió Sobre Otras?

- Escalabilidad y Rendimiento: **Kafka**
- Coordinación Distribuida Confiable: **Zookeeper**
- Desarrollo ágil y Moderno de APIs: **FastAPI**
- Compatibilidad y Sinergia

¿Qué Problemas Resuelven?

- **Kafka –** Dificultad de comunicar múltiples servicios entre sí de forma eficiente, escalable y desacoplada
- **ZooKeeper –** Coordinación y consistencia en sistemas distribuidos
- **FastAPI –** Necesidad de desarrollar APIs rápido y de forma robusta

2

Problema Propuesto

Descripción del Problema

Descripción

La empresa Daily Planet recibe diariamente una gran cantidad de correos electrónicos de sus suscriptores.

El sistema creado debe:

- Clasificar automáticamente los correos entrantes.
- Enviar confirmación en menos de 5 minutos.
- Escalar ante alta demanda y generar métricas del proceso.

2

Solución Propuesta

Descripción de la Solución

Descripción

1. **Collector:** Recibe los correos entrantes y los publica en un tópico de Kafka.
2. **Processor:** Procesa y clasifica los correos en tópicos según su contenido y envía un mensaje de confirmación de procesamiento al tópico (acknowledge).
3. **Responder:** Cierra la comunicación enviando un acuse de recibo a las direcciones de correo de los clientes correspondientes.

2

Resultados Obtenidos

Resultados

Resultados

- Sistema que recibe, clasifica y responde correos automáticamente.
- Procesamiento en tiempo real con baja latencia gracias a Kafka.
- Arquitectura escalable y modular.
- Cumplimiento exitoso de los objetivos planteados.

2

Conclusiones

Conclusiones...

Conclusiones

- El sistema automatiza con éxito la gestión de correos, mejorando la eficiencia.
- La integración de Kafka y FastAPI permitió una arquitectura escalable y desacoplada.
- Futuras mejoras incluyen optimizar la clasificación y agregar monitoreo en tiempo real.

Referencias

DanielSolano. (s. f.). *GitHub - DanielSolano8/IF6100-C17640: Analisis*. GitHub. https://github.com/DanielSolano8/IF6100-C17640.git

Apache Kafka. (s. f.). Apache Kafka. https://kafka.apache.org/documentation/

FastAPI. (s. f.). https://fastapi.tiangolo.com/

Apache ZooKeeper. (s. f.-b). https://zookeeper.apache.org/

GRACIAS

"Cualquier tecnología suficientemente avanzada es indistinguible de la magia." – Arthur C. Clarke