UNIVERSIDAD NACIONAL DE SAN AGUSTÍN

Facultad de Ingeniería de Producción y Servicios

Ciencia de la Computación

Proyecto Final de Big Data

ALUMNOS:
Vilca Limachi Santiago Javier

DOCENTES:
Mamani Aliaga, Alvaro

CURSO:
Big Data

16 de julio de 2025

Índice

1. Introducción

En el presente proyecto se propone el diseño y desarrollo de una plataforma distribuida para la publicación, visualización y análisis de cómics y artworks. Esta plataforma se inspira en redes como Webtoon o Instagram, permitiendo a los usuarios subir su contenido, visualizar obras de otros autores, dar likes, y realizar búsquedas de cómics o creadores.

Con el objetivo de aprovechar tecnologías modernas de Big Data, se ha diseñado un pipeline de datos que permite gestionar grandes volúmenes de información, procesar eventos en tiempo real, y generar análisis batch para la toma de decisiones o presentación de contenido personalizado.

2. Objetivos

Objetivo General

Implementar un pipeline de datos distribuido para una plataforma de gestión y visualización de cómics, utilizando herramientas de procesamiento en tiempo real y almacenamiento escalable que permitan ofrecer análisis de comportamiento de los usuarios y recomendación de contenido.

Objetivos Específicos

- Desarrollar una API backend en .NET Core para recibir y gestionar las acciones del usuario.
- Construir un sistema de ingesta de eventos basado en Apache Kafka para registrar actividades como visualizaciones, likes y subidas.
- Aplicar Apache Spark Streaming para procesar estos eventos en tiempo real.
- Almacenar los datos procesados y los archivos en un sistema de archivos distribuido mediante HDFS.
- Implementar un motor de búsqueda eficiente usando Elasticsearch para indexar los cómics y permitir búsquedas por título, autor o etiquetas.

3. Pipeline

El pipeline de datos es el corazón del sistema propuesto. Permite procesar en tiempo real las interacciones generadas por los usuarios, almacenar el contenido distribuido de manera eficiente, y presentar resultados tanto en forma de dashboards como búsquedas.

Componentes del Pipeline

- **Frontend (React):** Interfaz principal desde donde el usuario interactúa con la plataforma. Permite visualizar cómics, dar likes, hacer búsquedas y subir contenido.
- **Backend (.NET Core):** API encargada de recibir las peticiones del usuario, validar entradas, subir archivos y producir eventos a Kafka según las acciones realizadas.
- **Apache Kafka:** Sistema de mensajería que permite desacoplar la entrada de datos del procesamiento. Se han definido cuatro topics:
  - `Topic1`: Subida de nuevo cómic.
  - `Topic2`: Visualización de un cómic.

  - `Topic3`: Registro de likes.
  - `Topic4`: Registro de eventos de búsqueda o navegación.

- **Apache Spark Streaming:** Motor de procesamiento en tiempo real. Se encarga de consumir los eventos de Kafka, realizar agregaciones, conteos y aplicar posibles algoritmos de clustering o ranking. Los resultados son almacenados en HDFS o exportados como archivos CSV/JSON para ser consumidos por dashboards.
- **HDFS (Hadoop Distributed File System):** Almacén principal del sistema. Contiene las imágenes de cómics, capítulos, así como los datos históricos y resultados procesados por Spark.
- **PostgreSQL / Archivos CSV/JSON:** Repositorio para los resultados estructurados generados por Spark en procesos batch. Facilita la creación de dashboards e informes.
- **Elasticsearch (opcional):** Motor de búsqueda que indexa los metadatos de los cómics y permite realizar búsquedas rápidas por parte del usuario en la interfaz.

Flujo General del Pipeline

1. El usuario realiza una acción en la plataforma: subir un cómic, ver una página, dar like, etc.
2. La API recibe esta acción y produce un mensaje al topic correspondiente de Kafka.
3. Spark Structured Streaming consume los mensajes en tiempo real, procesa los datos y los almacena en HDFS, o genera resultados de resumen que se guardan en PostgreSQL o archivos CSV/JSON.
4. Cuando se sube un nuevo cómic, su metadata es enviada a Elasticsearch para que esté disponible en la barra de búsqueda.
5. El frontend consume la información desde la API y presenta resultados actualizados de forma dinámica.

Ventajas del Enfoque Distribuido

- Alta escalabilidad y tolerancia a fallos gracias a la naturaleza distribuida de Kafka, Spark y HDFS.
- Separación clara entre la capa de ingesta, procesamiento, almacenamiento y presentación.
- Posibilidad de realizar análisis en tiempo real y procesamiento batch en paralelo.
- Mejora significativa en los tiempos de búsqueda mediante Elasticsearch.
- Estructura mantenible y modular, facilitando el desarrollo colaborativo y la extensión futura del sistema.