Distributed Anime Streaming Analytics Platform

J'Moi testamark
CS 4265 – Big Data Analytics
Kennesaw State University
Email: jtestama@students.kennesaw.edu

***Abstract*—This project presents the design of a distributed big data analytics system for analyzing large-scale anime streaming and viewing data. The system processes public anime APIs like MyAnimeList API and AniList API. The platform enables scalable trend analysis, genre distribution analysis, and user behavior modeling.**

I. Introduction

The rapid growth of online streaming platforms has resulted in massive volumes of user interaction data. Anime streaming services generate millions of viewing records daily, creating challenges in storage, processing, and querying. Traditional single-machine systems cannot efficiently handle such large-scale workloads. This project proposes a distributed analytics platform that enables scalable processing of anime streaming data using big data technologies.

II. Project Overview

A. Project Title

Distributed Anime Streaming Analytics Platform

B. Domain

Streaming Media Analytics and Entertainment Data Processing

C. Problem Statement

Anime streaming platforms must analyze large volumes of user viewing logs, reviews, and metadata. The system must answer questions such as:

- Which anime titles are trending?
- What genres are most popular?
- When do users binge-watch episodes?
- How does popularity vary by region?

These analyses require distributed storage and parallel processing.

D. Scope

This project focuses on:

- Batch ingestion of anime metadata and viewing logs
- Distributed processing using Apache Spark
- Storage in HDFS and MongoDB
- Querying using Spark SQL

The project does not include real-time dashboards, machine learning model training, or access to proprietary data. Only publicly available APIs and data sets.

III. System Description

A. Data Sources

The system uses the following data sources:

- AniList public API
- MyAnimeList public API
- Public Kaggle anime datasets

B. Data Characteristics

1) Volume: The dataset is expected to range from 10GB to 50GB.

2) Variety: Data is stored in JSON, CSV, and Parquet formats.

3) Velocity: Data is processed in batch mode with daily updates.

C. Big Data Stack Layers

TABLE I
Technology Stack

Layer	Technology
Storage	HDFS
Data Format	JSON, Parquet
Processing	Apache Spark
Data Store	MongoDB
Query	Spark SQL

The system utilizes HDFS block storage and replication, Spark DAG-based execution, and partitioned data layouts.

D. Assumptions

The following assumptions are made:

- All user identifiers are anonymized
- Network connectivity is stable
- Input data is reasonably clean

IV. Implementation Approach

A. Technology Choices

Apache Spark is selected for its in-memory processing capabilities and scalability. HDFS provides fault-tolerant distributed storage. MongoDB supports flexible document storage.

B. Processing Model

The system follows a batch ETL model. Raw data is ingested, cleaned, transformed, and loaded into analytical storage.

C. Scalability Plan

Scalability is achieved through:

- Horizontal scaling of Spark workers
- HDFS block replication
- Data partitioning by date and region

D. Evaluation Metrics

System performance is evaluated using:

- Job execution time
- Throughput (records per second)
- Query response latency

V. Literature Review

A. Big Data Textbook

The textbook explains the foundations of big data systems, including distributed storage, data pipelines, batch, and stream processing. The source provides me with the foundation for my information to tackle this project. This project adapts the pipeline model but uses Python, Apache Spark, and MongoDB.

B. Apache Spark

Apache Spark is a distributed data processing framework that supports in-memory computation. It also provides APIs for dataframes, SQL queries, and machine learning. Spark is the main processing engine for analyzing large anime viewing datasets. The project uses PySpark on my device.

C. MongoDB

MongoDB is a NoSQL document-oriented database that stores JSON files. It will be used to store anime data, viewing statistics, and the recommendations from Spark. The project will use a single-node MongoDB instead of a distributed cluster.

D. bigtable

Bigtable introduces a distributed, column-oriented storage system designed for large data sets. For this project, it will implement similar design principles using MongoDB.

E. MapReduce

MapReduce is a programming model for processing large datasets using parallel map and reduce functions. Following using Spark. MapReduce will be used for operations such as view counting or grouping by genres. This project uses Spark's high-level APIs, which implement MapReduce-like processing.

VI. Conclusion

This paper presented the design of a distributed analytics platform for large-scale anime streaming data. The proposed system integrates HDFS, Spark, and MongoDB to enable scalable processing and querying. Future work includes real-time ingestion and advanced analytics.

Repository

GitHub Repository: https://github.com/jtestamark999/anime-bigdata-analytics

References

[1] Fourny, G. (2024). The Big Data Textbook: From Clay Tablets to Data Lakehouses (2nd ed.). ISBN 979-8337883137. https://ghislainfourny.github.io/big-data-textbook/

[2] Apache Software Foundation. (2025). Apache Spark Documentation. https://spark.apache.org/docs/latest/'

[3] MongoDB Inc. (2025). MongoDB Manual. https://www.mongodb.com/docs/manual/

[4] Chang, F., et al. (2006). Bigtable: A Distributed Storage System for Structured Data. OSDI '06.

[5] Dean, J., Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. OSDI '04.