**Final Project CS523 (Big Data Technology)**

**Netflix Movie Data Analysis**

**Project GitHub Link for BDT-FINAL-PROJECT (click to browse)**

## Project Overview

- Objective: The project aims to identify top-rated Netflix movies based on user votes and ratings, providing real-time insights into movie popularity.

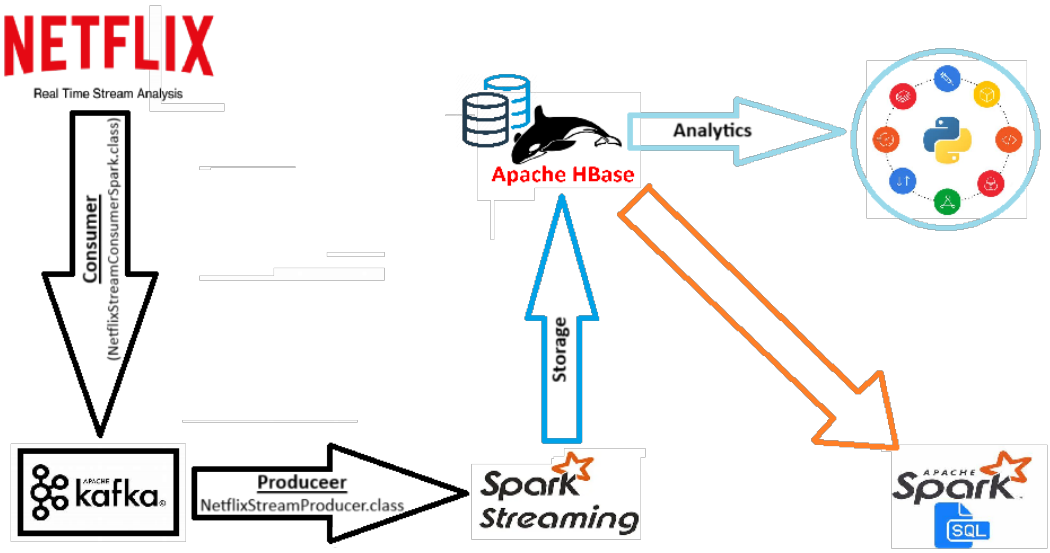
- Data Ingestion: Apache Kafka is used for streaming movie data (in CSV format) into Kafka topics. This enables real-time data ingestion for continuous updates.

- Data Storage and Processing: Data is processed using Apache Spark, which applies transformations to clean, filter, and aggregate the data. Processed data is stored in HBase for scalable, high-performance access. The pipeline ensures efficient handling of large datasets.

- Analysis & Visualization: Python handles the analysis and visualization. Libraries like Pandas and Matplotlib are used to identify trends, such as the most frequently top-rated movies. These insights are visualized through charts and graphs to demonstrate movie popularity.

- System Development: The core system is developed in Java, ensuring seamless integration with Kafka and HBase. Maven is used for managing dependencies and automating builds, supporting smooth development workflows.

- Architecture and process flow:
    1. Kafka Producer: Streams movie data.
    2. Spark Streaming: Processes real-time data from Kafka.
    3. HBase: Stores processed data for scalable access.
    4. Python Analytics: Runs queries and generates visualizations.

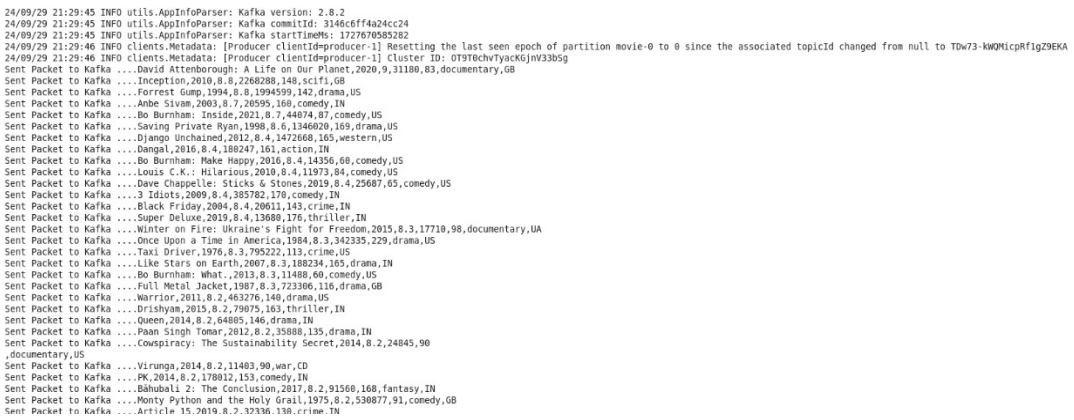



- Tools and Technologies:
  - ✓ Hadoop: Version 2.6.0-cdh5.13.0
  - ✓ HBase: Version 1.2.0-cdh5.13.0
  - ✓ Spark: Version 1.6.0
  - ✓ Kafka: Version 2.13-2.8.2
  - ✓ Java JDK & JRE: Version 1.8.0-181
  - ✓ Python
  - ✓ Platform: CentOS Version 6.7
  - ✓ IDE: Eclipse LUNA 4.4.2, VS Code

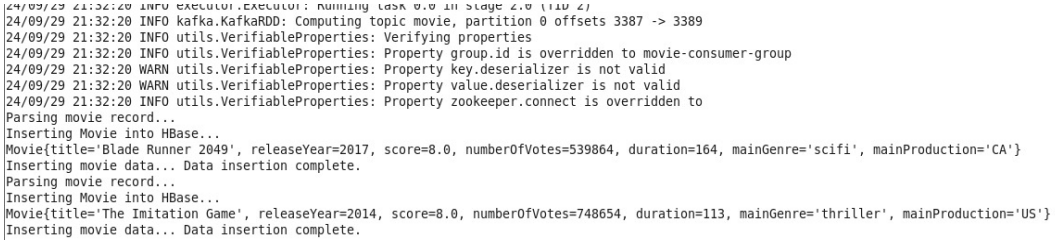
**Project Implementation Details:**

1. A custom Java producer application was developed to read and stream Netflix movie data from a CSV file. This application streams the data to a Kafka topic

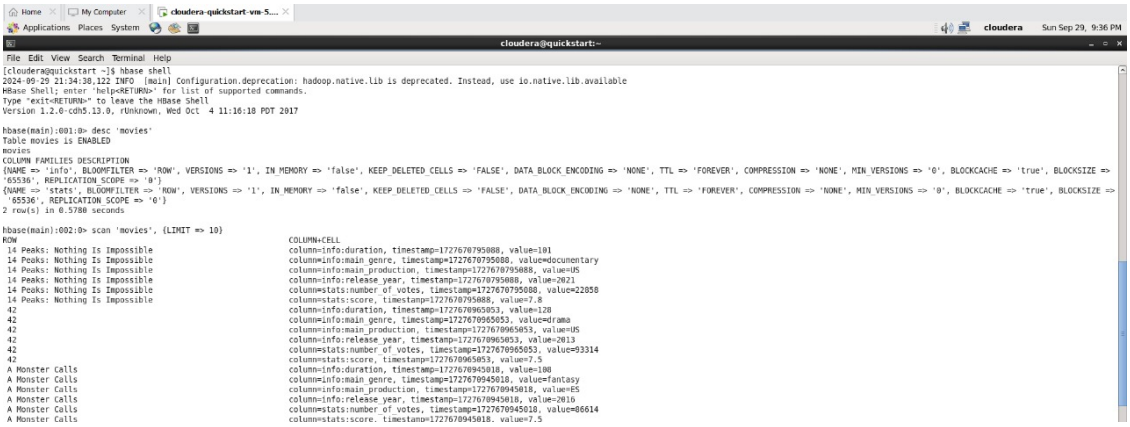
for further processing in real-time.

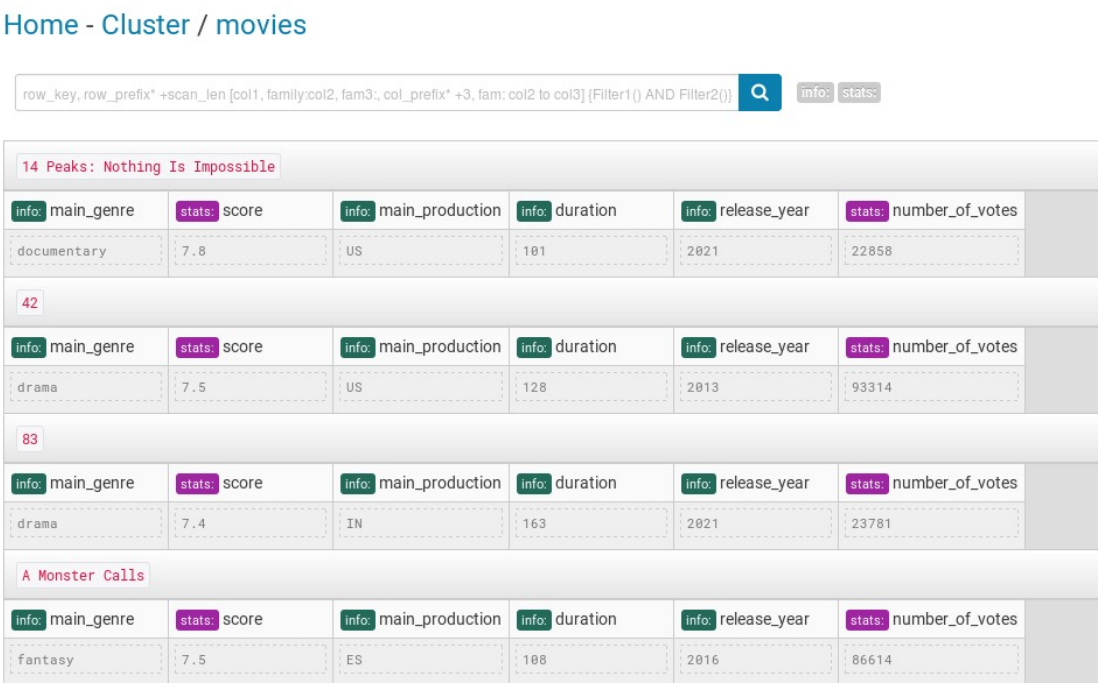



2. A custom Java application was developed to leverage Apache Spark's streaming functionality, which pulls data from the Kafka topic. The processed data is then stored in an HBase table for efficient access and scalability.




3. An HBase table was created to facilitate efficient data storage, access, and management using HBase's powerful tools and capabilities.




## Visualization:

We developed a custom Python application as our visualization tool, hosted on the local machine. To interact with the HBase tables, we utilized the HappyBase library. This application allows users to select different parameters, generating analytical insights and graphical visualizations based on the chosen dataset.

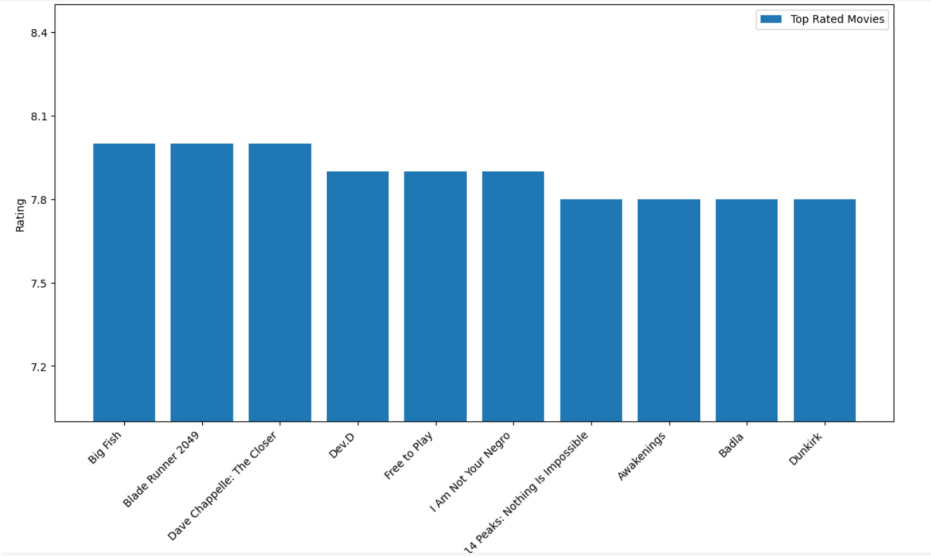
Creating Dataframe based on Data consumed From Hbase table "MOVIES"

```
import happybase
import pandas as pd

# HBase server details
hbase_host = '192.168.117.152'
hbase_table = 'movies'

# Connect to HBase
connection = happybase.Connection(hbase_host)

# Open the table
table = connection.table(hbase_table)

# Scan HBase table and fetch data
data = []
for key, value in table.scan(limit=100):  # Limit to 100 rows
    # Decode row key (could be an ID or date)
    row_key = key.decode('utf-8')

    # Decode column names and values
    row_data = {col.decode('utf-8'): value[col].decode('utf-8') for col in value}

    # Include the row key (e.g., ID) in the row data
    row_data['row_key'] = row_key

    data.append(row_data)

# Create DataFrame from HBase data
df = pd.DataFrame(data)
df['stats:score'] = pd.to_numeric(df['stats:score'], errors='coerce')
df['stats:number_of_votes'] = pd.to_numeric(df['stats:number_of_votes'], errors='coerce')
# Close the HBase connection
connection.close()
```

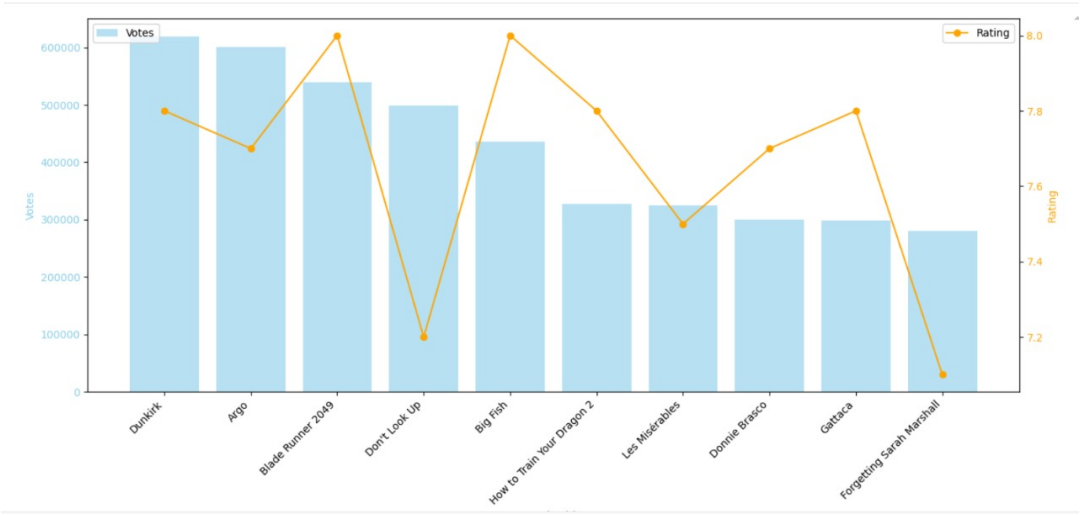
[190]

## User Analytics:

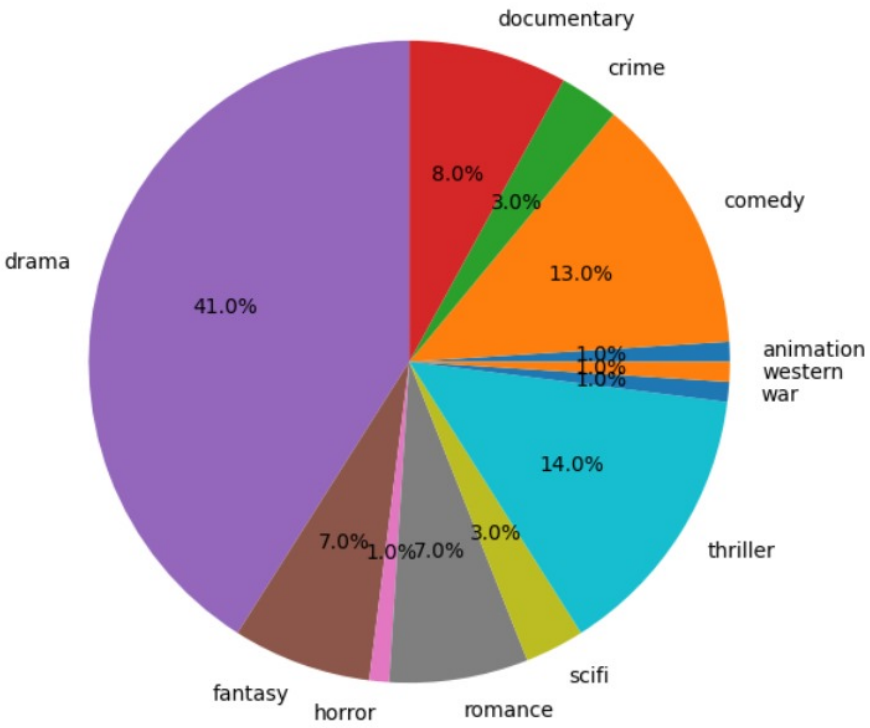
1. List of Top-Rated Movies




2. List of Movies rated by voters




3. Distribution of Movies by Genre




[Project GitHub Link for BDT-FINAL-PROJECT (click to browse)]