

CALIFORNIA STATE UNIVERSITY,
FULLERTON



STREAMIFY-
ONLINE STREAMING APPLICATION

Course: CPSC 531-01 13480 Advance Database Management
Professor: Tseng-Ching James Shen

Prepared by:
Jahanvi Paliwal
Bhavyom Singh Kushwaha

TABLE OF CONTENTS

INTRODUCTION:	3
PROBLEM STATEMENT:	3
AIM OF THE PROJECT:	3
APPROACH:	4
TOOLS AND TECHNOLOGIES USED:	4
ARCHITECTURE:	6
DATASET INFORMATION:	6
STEPS TO RUN THE APPLICATION:	7
FLOW OF OUR APPLICATION:	8
FUTURE SCOPE:	12
CONCLUSION:	12

INTRODUCTION

Online streaming website

In today's digital age, online streaming has become an integral part of our lives, with millions of people around the world turning to streaming websites to watch their favorite content in real-time. Online streaming websites have revolutionized the way we consume content, offering users access to a vast array of content from news and sports to entertainment and education. In this project, we will explore big data technologies to power an online streaming website to help improve the user experience, increase user engagement and satisfaction, and drive business growth.

PROBLEM STATEMENT

The problem statement for our presentation on an online streaming website is to address the challenges and limitations that online streaming websites face when collecting and processing large volumes of user data, and how big data technologies can help overcome these challenges. We will also discuss the challenges of personalizing content for each user, making recommendations based on their viewing history and preferences, and providing a more engaging and personalized user experience.

AIM OF THE PROJECT

The aim of the project on the topic of "Online Streaming Website Using Big Data Technologies" will be to design and develop a robust and scalable online streaming platform that leverages big data technologies to provide a personalized and engaging user experience. The project will involve exploring and implementing various big data technologies to address the challenges and limitations faced by online streaming websites in collecting and processing large volumes of user data. The ultimate goal of the project will be to demonstrate the importance of big data technologies in the context of online streaming and how they can be used to enhance the overall user experience and drive business success in today's digital age.

APPROACH TO DESIGN THE PROJECT

- Kafka is an open-source distributed stream-processing framework that collects real-time data from the e-commerce website and stores it in Kafka-broker (Cluster).
- This stream of data is directed to the processing engine (Spark) through Kafka-consumer.
- The data streams undergo various processes such as Data injection, Data Storage, Data Processing.
- The raw data is stored in Cassandra (for further enhancement in the application).

Kafka Internal Functioning

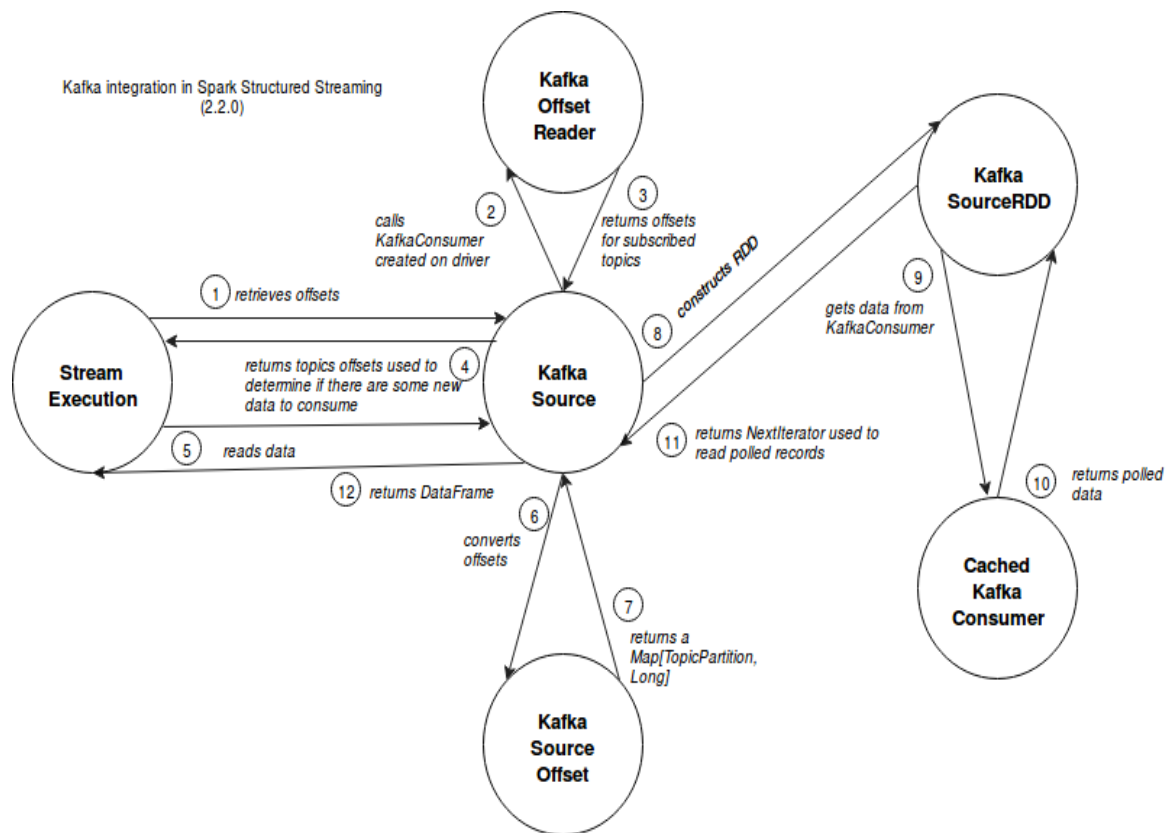
- The topic Streamify is created in the Kafka-Cluster.
- Kafka mainly has two important API's.
- Kafka-Producer collects all the stream of data from the user and allows the application to write that into the topic in the cluster.
- From the other side, the kafka-Consumer allows the application to read the stream of data from the topic in the cluster and gives it to spark for further processing.

Fig 1: Internal working of Kafka

TOOLS AND TECHNOLOGIES USED

- Hadoop - open-source software utility used for storage and computation of data.
- Kafka - used for capturing and storing streams of real-time data into categories called topics and later gives it to a processing engine for analysis.
- Spark - a data processing engine used for collecting data-stream from Kafka and further storing, processing, and analyzing the data.

- Cassandra – NoSQL database management system used to store raw data.
- Python – programming language.
- Visual Studio Code – Integrated development environment (IDE).



ARCHITECTURE

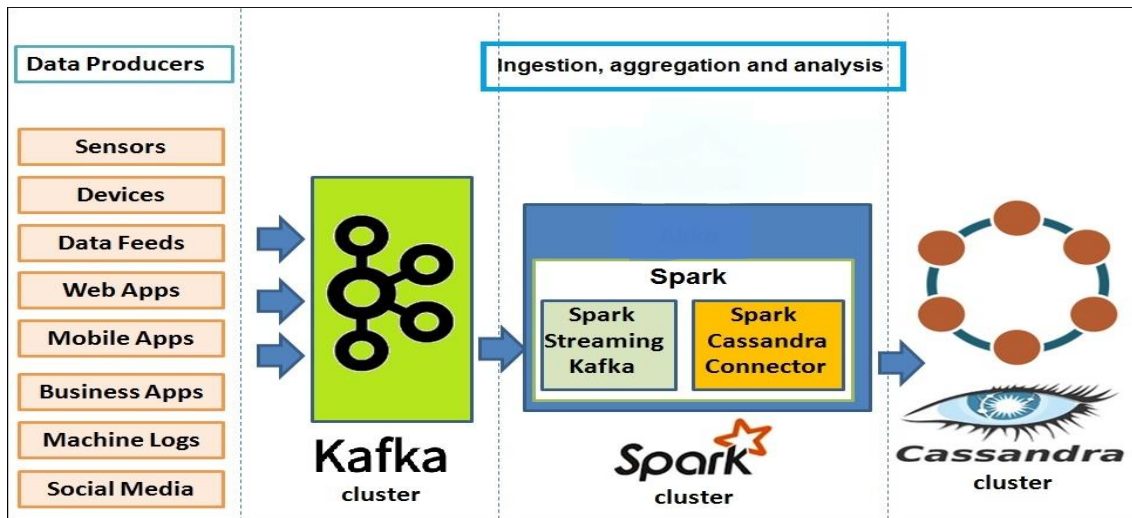


Fig: Real-Time Data Analysis Application Architecture

DATASET INFORMATION

This datasets was used:

- Customers dataset

Data Attributes: customers.csv

- Name - customer's name
- Email - customer's email address
- Gender - customer's gender
- Age - customer's age
- Interest - customer's interest

	A	B	C	D	E	F
1	Name	Email id	Gender	Age	Interest	
2	Akarsh Gupta	akarshgupta494@gmail.com	Male	27	Sports	
3	Simran kaur	simrank@gmail.com	Female	28	Book	
4	Parvati Gupta	parvatigup456@gmail.com	Female	26	Food	
5	Manikarna Verma	manikarna@gmail.com	Male	18	Beauty	
6	Yash Jain	yashjain123@gmail.com	Male	19	Food	
7	Rashi Srivastava	rashisri@gmail.com	Female	39	Clothing	
8	Pankhudee gupta	pankhudee30@gmail.com	Female	33	Book	
9	Meenal Sarwaiya	Meenalsarwaiya@gmail.com	Female	26	Home	
10	Prachi	prachi234@gmail.com	Female	27	Home	
11	Neha	nehatayal2292@gmail.com	Female	40	Sports	
12	Abhishek Tayal	abhishektayal25@gmail.com	Male	25	Book	
13	Sonali Tiwari	sonusarveshtiwari@gmail.com	Female	34	Clothing	
14	Pushpal	Pushpal@pushpal.com	Female	28	Pet Supply	
15	Jyoti modi	modijyoti4@gmail.com	Female	37	Food	
16	Ruchi	Ruchi20.rj@gmail. Com	Female	36	Food	
17	Pankaj	pankajjain7119@gmail.com	Male	36	Beauty	
18	Ayush Jain	ayushjain71@gmail.com	Male	29	Beauty	
19	Vijaya jain	Vijayajain19@gmail.com	Female	37	Beauty	
20	Tanishq Wadhwani	wadhwanit@gmail.com	Male	38	Food	
21	Sarvesh Tiwari	sarveshtiwariibjp@gmail.com	Male	26	Food	
22	Mridul Tayal	Sarla1641@gmail.com	Male	35	Book	
23	Sanjeev Garg	sanjeevca2039@gmail.com	Male	27	Beauty	

Fig : Dataset: customers.csv

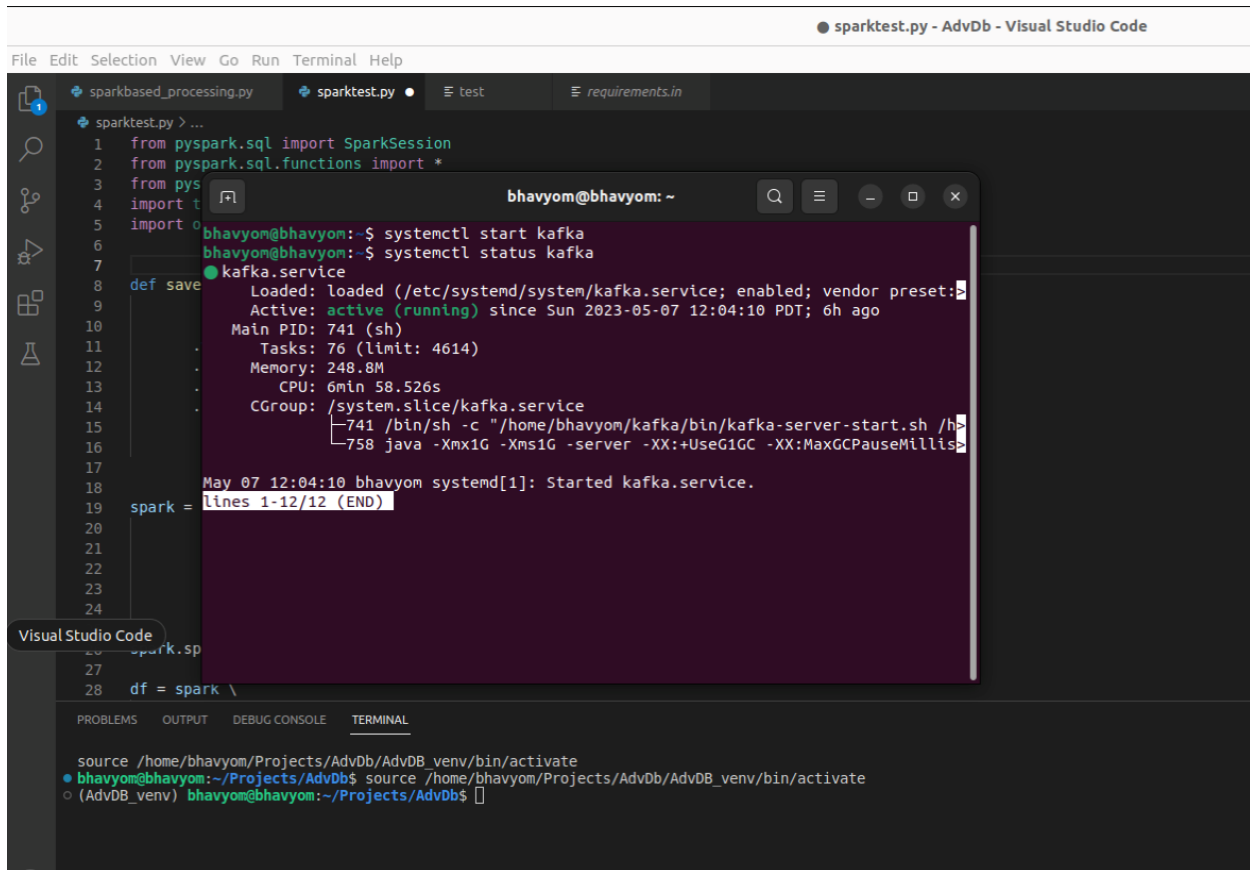
STEPS TO RUN THE APPLICATION

1. Open 2 terminals in Ubuntu.
2. We must start zookeeper and kafka. I have create a single file where I have written commands for running both of these. On one terminal, write a command, “systemctl start kafka”. There will be a prompt asking for password, enter the password. The zookeeper and kafka server will start. To check if the server are running or not, use “systemctl status kafka”.
3. Write the command “kafka-topics.sh --bootstrap-server localhost:9092 --topic *<topic name>* --replication-factor 2”. Then press enter. This command as the same suggest, create a topic and bind it to localhost at port 9092.
4. Now, we need to start the producer. Write the command, “kafka-console-producer.sh --bootstrap-server localhost:9092 --topic *<topic name>*”. Then press enter. This command will start a console producer at localhost:9092 with some properties such as topic, replication factor and partition.
5. This terminal will act as a producer of information. It will write anything that you provide on the topic. Things written on the topic will be available for the consumer to consume.
6. On the other terminal, write the following command, “spark-submit --packages “com.datastax.spark:spark-cassandra-

connector_2.12:3.3.0","org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.2"<path of the file where you are reading messages from kafka topic>". Then press enter. This will deploy the spark cluster.

7. In the above command, we are specifying a package for spark-cassandra connector library, that is used for establishing connection between spark and Cassandra. The other package is for kafka-spark connection.
8. Now, messages produced by producer will be available for consumer (spark application in our case). Once the consumer consumes the data, it will be processed by spark and stored in a specific format in Apache Cassandra.

FLOW OF OUR APPLICATION



The screenshot shows the Visual Studio Code interface. The editor displays a file named `sparktest.py` with the following content:

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql.functions import *
3 from pys
4 import t
5 import o
6
7
8 def save
9
10
11
12
13
14
15
16
17
18
19 spark =
20
21
22
23
24
27
28 df = spark \
```

Overlaid on the editor is a terminal window titled `bhavyom@bhavyom: ~`. It shows the following commands and output:

```
bhavyom@bhavyom:~$ systemctl start kafka
bhavyom@bhavyom:~$ systemctl status kafka
● kafka.service
   Loaded: loaded (/etc/systemd/system/kafka.service; enabled; vendor preset:
   Active: active (running) since Sun 2023-05-07 12:04:10 PDT; 6h ago
     Main PID: 741 (sh)
        Tasks: 76 (limit: 4614)
       Memory: 248.8M
          CPU: 6min 58.526s
        CGroup: /system.slice/kafka.service
               └─741 /bin/sh -c "/home/bhavyom/kafka/bin/kafka-server-start.sh /h
                 └─758 java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis
May 07 12:04:10 bhavyom systemd[1]: Started kafka.service.
```

Below the terminal window, the `TERMINAL` tab is active, showing the following commands:

```
source /home/bhavyom/Projects/AdvDb/AdvDB_venv/bin/activate
bhavyom@bhavyom:~/Projects/AdvDb$ source /home/bhavyom/Projects/AdvDb/AdvDB_venv/bin/activate
(AdvDB_venv) bhavyom@bhavyom:~/Projects/AdvDb$
```



```

bhavyom@bhavyom: ~
kafka-topics.sh --create --bootstrap-server localhost:9092 --
topic kafkaSpark --partitions 3 --replication-factor 1
Created topic kafkaSpark.
bhavyom@bhavyom:~$
```

```

bhavyom@bhavyom:~$ kafka-topics.sh --create --bootstrap-server localhost:9092 --
topic kafkaSpark --partitions 3 --replication-factor 1
Created topic kafkaSpark.
bhavyom@bhavyom:~$ kafka-console-producer.sh --bootstrap-server localhost:9092 -
-topic kafkaSpark
>
```

```

(AdvDB_venv) bhavyom@bhavyom:~/Projects/AdvDB$ spark-submit --packages "com.datastax.spark:spark-cassandra-connector:2.12:3.3.0","org.apache.spark:spark-sql-kafka-0-10:2.12:3.3.2" sparktest.py
23/05/07 19:08:24 WARN Utils: Your hostname, bhavyom resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s3)
23/05/07 19:08:24 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
:: loading settings :: url = jar:file:/home/bhavyom/Projects/AdvDB/AdvDB_venv/lib/python3.10/site-packages/pyspark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/bhavyom/.ivy2/cache
The jars for the packages stored in: /home/bhavyom/.ivy2/jars
com.datastax.spark:spark-cassandra-connector:2.12 added as a dependency
org.apache.spark:spark-sql-kafka-0-10:2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-9f059c96-df58-41a3-b8ad-b1290c82db5f:1.0
  confs: [default]
```

```

bhavyom@bhavyom:~$ kafka-topics.sh --create --bootstrap-server localhost:9092 --
topic kafkaSpark --partitions 3 --replication-factor 1
Created topic kafkaSpark.
bhavyom@bhavyom:~$ kafka-console-producer.sh --bootstrap-server localhost:9092 -
-topic kafkaSpark
>{'id':7, 'name':'Itachi','createdat':'04-05-2023', 'interestedin':'socialMedia'
,'age':26}
>
```

```

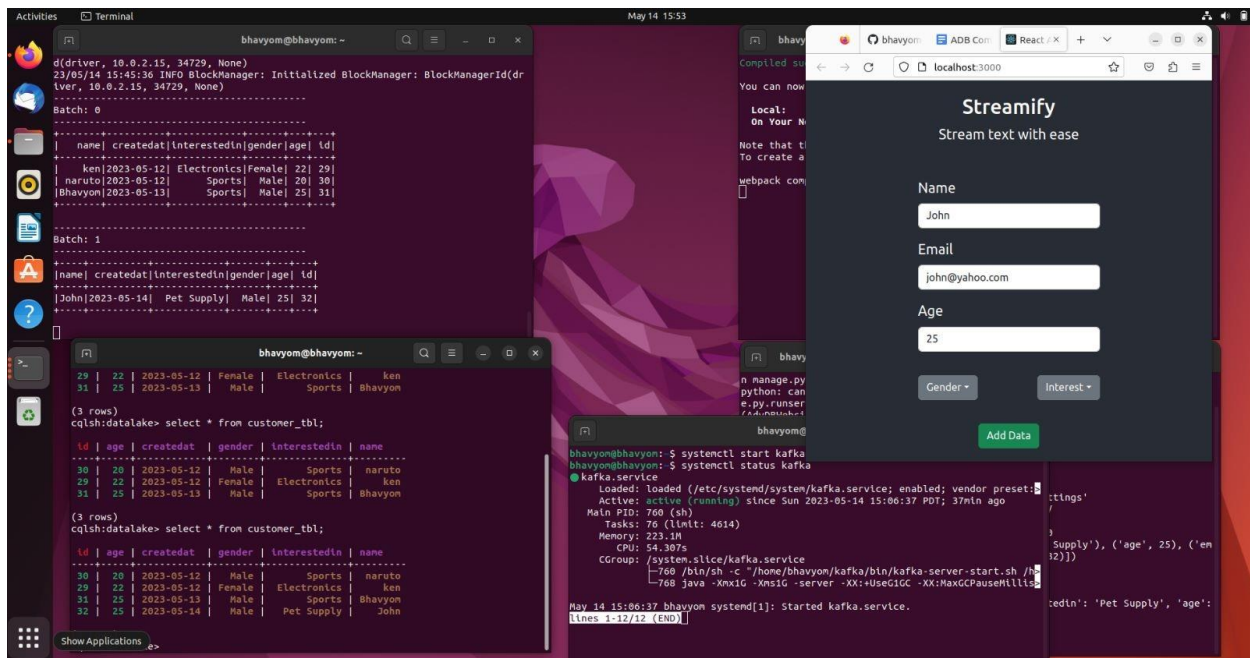
bhavyom@bhavyom:~$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.1 | CQL spec 3.4.6 | Nat
Use HELP for help.
cqlsh> use demotest
... ;
cqlsh:demotest> select * from customer_tbl;

id | age | createdat | interestedin | name
-----+-----+-----+-----+-----
5 | 25 | 04-05-2023 | books | shikamaru
1 | 28 | null | electronics | sasuke
2 | 23 | null | electronics | naruto
4 | 25 | 04-05-2023 | cloths | sakura
7 | 26 | 04-05-2023 | socialMedia | Itachi
6 | 21 | 04-05-2023 | socialMedia | jahanvi
3 | 28 | null | electronics | kakashi

(7 rows)
cqlsh:demotest>
```

```

Batch: 1
+-----+-----+-----+-----+-----+
| name| createdat|interestedin|age| id|
+-----+-----+-----+-----+-----+
|Itachi|04-05-2023| socialMedia| 26| 7|
+-----+-----+-----+-----+-----+
```

Github link for the project :

<https://github.com/bhavyom-singh/AdvanceDatabaseProject.git>

FUTURE SCOPE

In Future, We can provide Recommendations based on customer behavior using Machine Learning. Like, Which age group is interested in which kind of products based on their order history and the data that we are taking.

CONCLUSION

The conclusion will emphasize that by leveraging big data technologies, online streaming websites can improve user engagement, satisfaction, and retention. These technologies enable the collection and analysis of vast amounts of user data, leading to actionable insights for content curation, targeted advertising, and improving the overall user experience.

Additionally, the conclusion may touch upon the potential future developments in big data technologies and their impact on the online streaming industry. It may highlight emerging trends such as real-time analytics, AI-powered recommendation systems, and the integration of user-generated data.

Overall, the conclusion will reinforce the significance of big data technologies in the online streaming industry and emphasize their role in driving business success by providing a more personalized and engaging user experience.