CSP 554 Big Data Technologies

Project Report

Online Retail Simulation and Analytics

Team Members:

Ravi Teja Batchala	(A20512513)
Rajesh Mavi	(A20481442)
Sadanand Satish Kolhe	(A20517969)
Rik Ganguli Biswas	(A20513718)

Under The Supervision
of
Prof. Joseph Rosen

College Of Computing
Illinois Institute of Technology
Chicago, Illinois. December 2023

Contents

I: PROJECT DETAILS

1.1 Project Topic

Apply a range of big data tools to explore some interesting data sets and derive insights from them. Ingest data, apply transformations, profile the data, summarize it, visualize it.

1.2 Overview

This project aims at crafting a lifelike simulation for an ecommerce platform's online shopping experience. The goal is to mimic user interactions, efficiently store data in both structured and unstructured formats, and analyze behaviors for valuable insights. Real-time processing, user segmentation, product analysis, and optional machine learning predictions are on the agenda.

1.3 Introduction

In the dynamic landscape of e-commerce, understanding user behavior and optimizing the online shopping experience is paramount for business success. This project endeavors to create a sophisticated and lifelike simulation of an e-commerce platform, aiming to replicate user interactions, streamline data storage, and extract valuable insights through robust real-time processing and analysis.

The primary objectives of this project encompass a comprehensive approach to user simulation, data management, and behavioral analysis. By generating synthetic data for various user actions, employing efficient storage mechanisms such as MySQL, CSV, and Parquet, and implementing a robust data pipeline using Apache Kafka, PySpark Streaming, and Apache Hive, the project aspires to deliver a holistic solution for e-commerce analytics. The project aims to improve decision-making by creating brief hourly reports and utilizing data visualization tools like Matplotlib and Plotly to display insightful charts and dashboards. This adaptability to changing user preferences and market trends is crucial to staying competitive in the dynamic field of online retail.

1.4 Literature Review

Citation	Author	Year	Key Findings
A study of preferences in a simulated online shopping experiment	Asle Fagerstrøm, Erik Arntzen, Gordon Foxall	2011	Study reveals consumer brand loyalty development influenced by environmental contingencies, showing preferences and reduced switching over time.
Effects of Pros and Cons of Applying Big Data Analytics to Consumers' Responses in an E-Commerce Context	Le, T. M, Liaw, S.-Y	2017	Positive effects of Big Data analytics on customer responses, emphasizing recommendation systems and dynamic pricing.
Analysis of the Impact of Big Data on E-Commerce in Cloud Computing Environment	Rongrui Yu, Chunqiong Wu, Bingwen Yan, Baoqin Yu, Xiukao Zhou, Yanliang Yu, Na Chen	2021	The study integrates MySQL, HBase, and cloud computing for e-commerce efficiency, leveraging big data for precision marketing and enhanced competitiveness.
Statistical Modeling and Simulation of Online Shopping Customer Loyalty Based on Machine Learning and Big Data Analysis	Jui-Chan Huang, Po-Chang Ko, Cher-Min Fong, Sn-Man Lai, Hsin-Hung Chen, Ching-Tang Hsieh	2021	Deleting null values and estimating replacements in commodity datasets improve data quality for effective credit risk assessment.
Mechanism of Big Data Analytics in Consumer Behavior on Online Shopping	Ruby Evangelin, Vasantha Shanmugam	2022	Big data variables (Capacity, Promptness, Data Range) impact online browsing behavior and influence purchasing decisions.

1.5 Technologies

1. Apache Spark: Apache Spark is an open-source, distributed computing system that provides a fast and general-purpose cluster-computing framework for big data processing, enabling scalable and efficient data analytics and machine learning applications.

2.Apache PySpark Streaming: Spark Streaming is a micro-batch processing framework for real-time data. It enables the processing of data streams in near real-time, making it suitable for applications requiring low-latency analytics.

3. Apache Kafka: Apache Kafka is a distributed event streaming platform designed for building real-time data pipelines and streaming applications. It is capable of handling high-throughput, fault-tolerant, and scalable data streaming. Kafka provides a publish-subscribe model for exchanging records between producers and consumers.

4. Apache Hive: Apache Hive is a data warehousing and SQL-like query language system built on top of Hadoop. It provides a high-level interface for analyzing and querying large datasets stored in Hadoop Distributed File System (HDFS) or other compatible file systems.

5. MySql: MySQL is an open-source relational database management system widely used for storing and retrieving structured data. Renowned for its performance, scalability, and ease of use,

6. Matplotlib/Plotly: Matplotlib, a versatile Python plotting library, empowers our data analysis by offering a comprehensive suite of visualization tools. Its diverse range of customizable plots, from line graphs to histograms, enables us to effectively communicate and explore complex datasets.

7. Hadoop: Hadoop is an open-source framework for distributed storage and processing of large datasets across clusters of computers, using a programming model called MapReduce

8. AWS S3: Amazon S3, part of Amazon Web Services (AWS), is a scalable object storage service facilitating the storage and retrieval of data from any web location, providing seamless scalability.

9. Zookeeper: Apache ZooKeeper is an open-source coordination service for distributed applications. It provides a centralized infrastructure for maintaining configuration information, naming, providing distributed synchronization, and group services.

10. Jupyter Notebook

II: DATA OPERATIONS AND ANALYSIS

2.1 Architecture and Data Flow

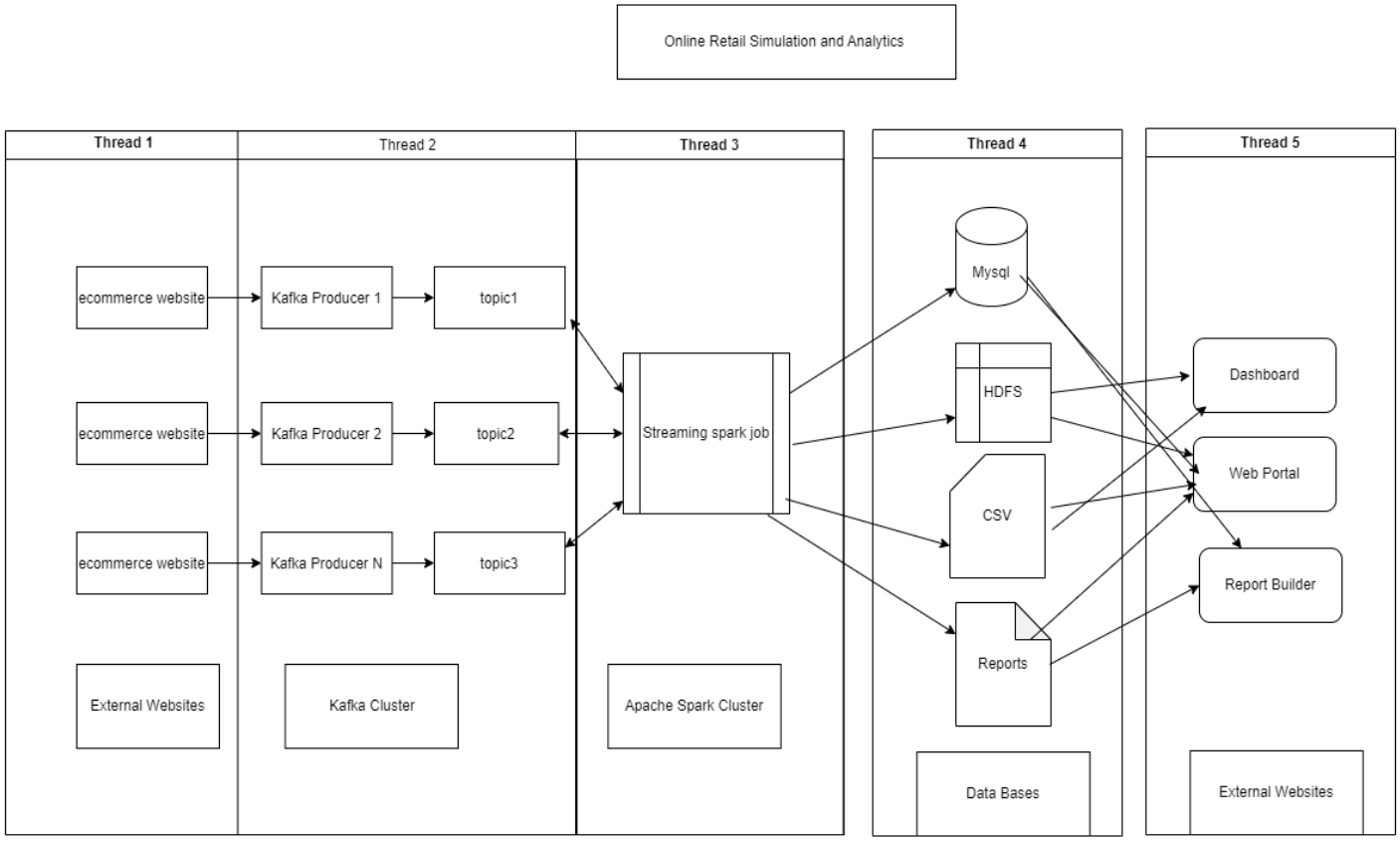


Fig 1: Architectural Diagram of Online Retail Simulation and Analytics System

1. Overview:

- System designed for real-time e-commerce system with Kafka producers, brokers, and PySpark streaming job. Workflow: ingest data, publish to Kafka, process with PySpark, generate reports, and store in a database.

2. Data Flow:

- E-commerce data sent to Kafka producers and Kafka producers publish data to Kafka topics.
- PySpark streaming job consumes data from Kafka topics in real-time.

3. Processing:

- PySpark streaming job executes intricate transformations on the incoming data.
- Processed data forms the basis for generating detailed reports.

4. Storage:

- Reports are systematically stored in a structured database.
- Database acts as a repository for future analytical exploration.

2.2 Data Source and Feature analysis

UserProfilesGenerator

Utilized UserProfilesGenerator to synthesize user profiles by considering factors such as demographic information, preferences, and historical behavior. The generated **UserProfiles_data.csv** includes attributes such as User_ID, Name, Age, Gender, Income, Contact, and Location, offering a comprehensive collection of user characteristics within the dataset.

ProductDataGenerator

Executed ProductDataGenerator to create synthetic product data by considering features such as Product_ID, Product_Name, Category, and Price. The resulting **Product_Data.csv** dataset encapsulates a varied assortment of products, mimicking a diverse product ecosystem.

UserInteractionsGenerator

Simulated user interactions with products using the tool, incorporating parameters such as user preferences, behavior patterns, and interaction types. The resulting dataset,**UserInteractions_data.csv**, includes attributes such as Interaction_ID, Timestamp, User_ID, Interaction_Type, Product_ID, Product_Name, Quantity, and Payment_Method, providing a diverse set of user engagement scenarios.

Feature Analysis

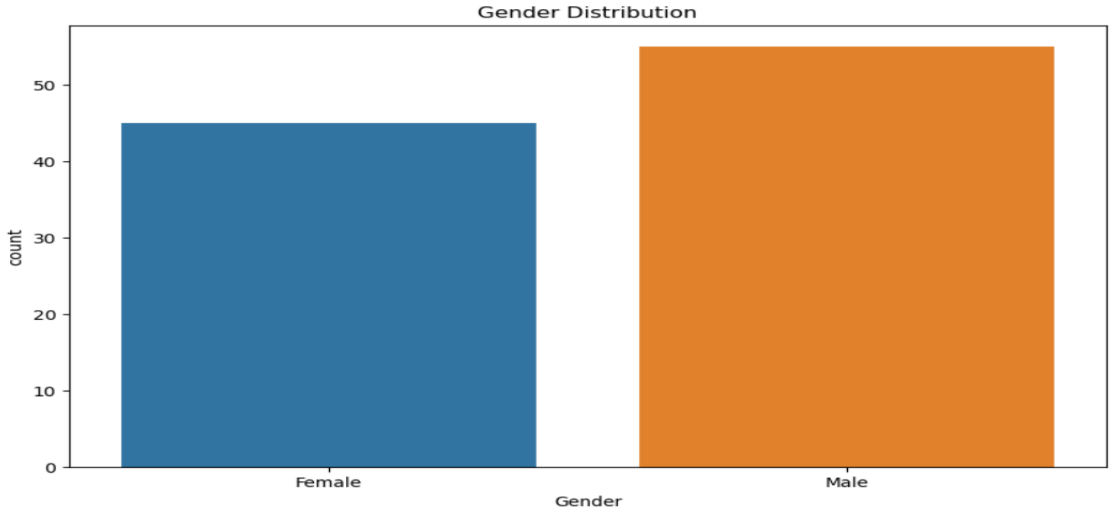
1) **UserProfiles_data**

Attribute	**Description**	**Data Type**
User_ID	User identifier	Numeric
Name	User's name	String
Age	User's age	Numeric
Gender	User's gender	String
Income	User's income	Numeric
Contact	User's contact info	String
Location	User's location	String

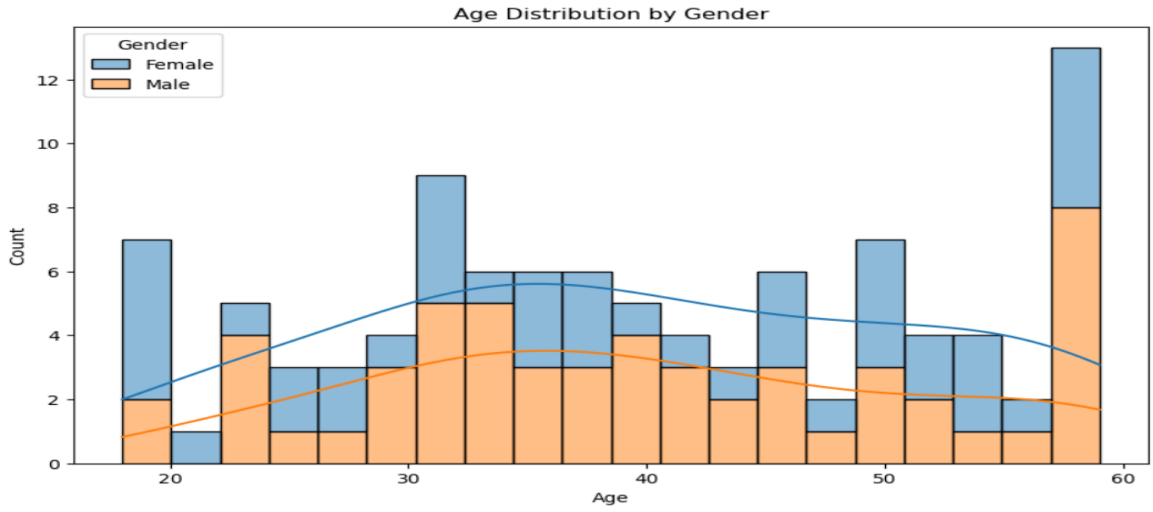
User_ID	Name	Age	Gender	Income	Contact	Location
1	Heather Burke	35	Female	68059	347-414-3911	61.389715, -7.198571
2	Margaret Brown	40	Male	73613	+1-632-865-1129	-61.234150, -9.520890
3	Pamela Crane	53	Male	75572	816.752.4431x56859	-68.287243, -135.834580
4	Erika Williamson	36	Male	63795	+1-652-669-8271	MDR37A, Basni, Nagaur Tehsil, Nagaur District, Rajasthan, 341021, India
5	Lori Mcintosh	38	Male	31765	777-357-5255x67102	80.204909, -172.165824

Fig 2: Sample Data of UserProfile

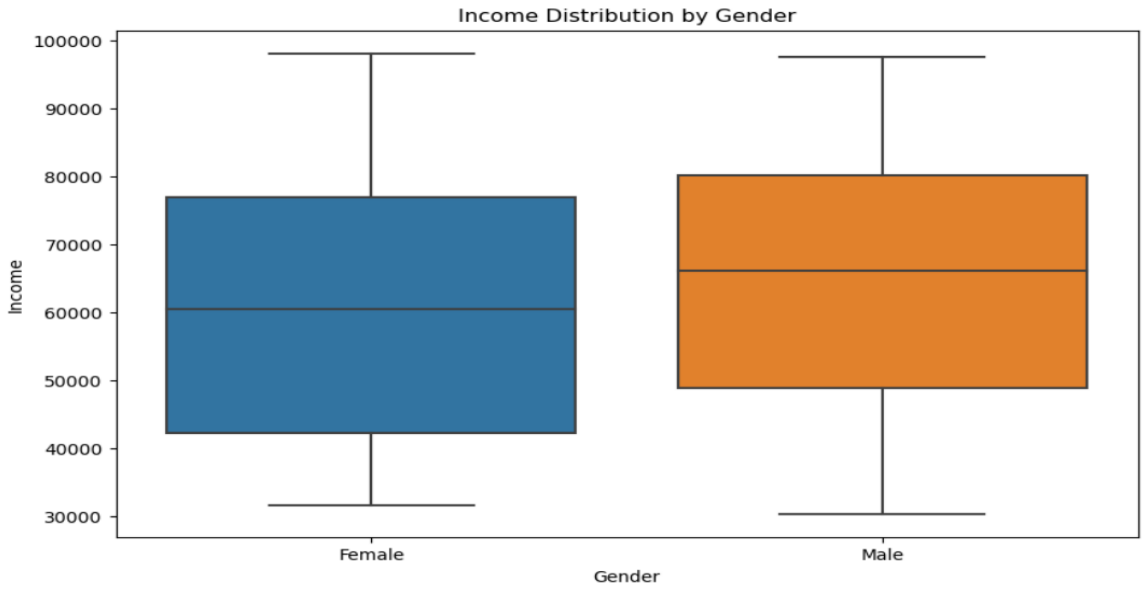
Gender Distribution: This provides a clear overview of the gender composition within a given dataset



Age distribution by gender: the x-axis represents different age groups, and the y-axis shows the count or percentage of individuals in each age group, segregated by male and female.



Income Distribution by gender: This visualization provides insights into income disparities or patterns across different gender demographics within a given population or dataset.



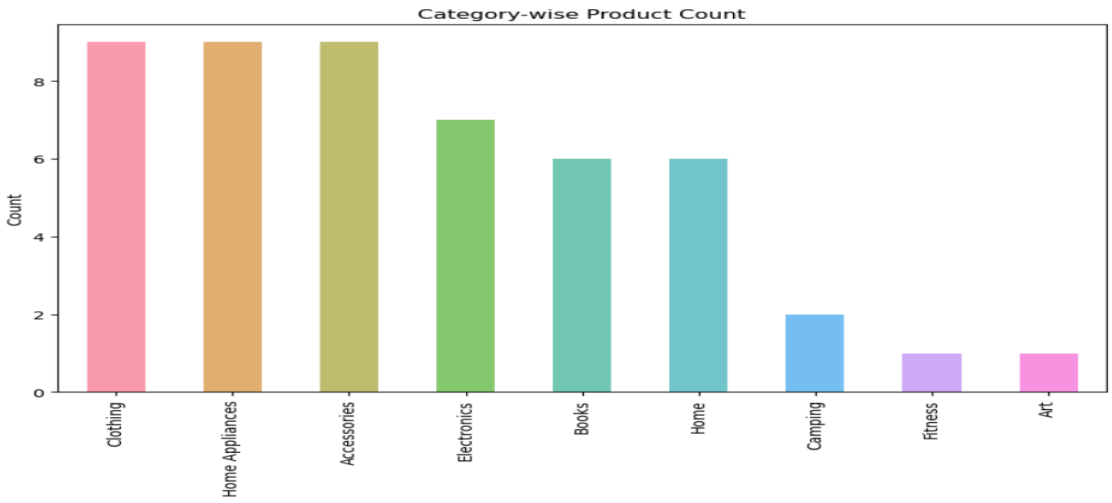
2) Products_data

Attribute	Description	Data Type
Product_ID	Product identifier	Numeric
Product_Name	Name of the product	String
Category	Product category	String
Price	Price of the product	Numeric

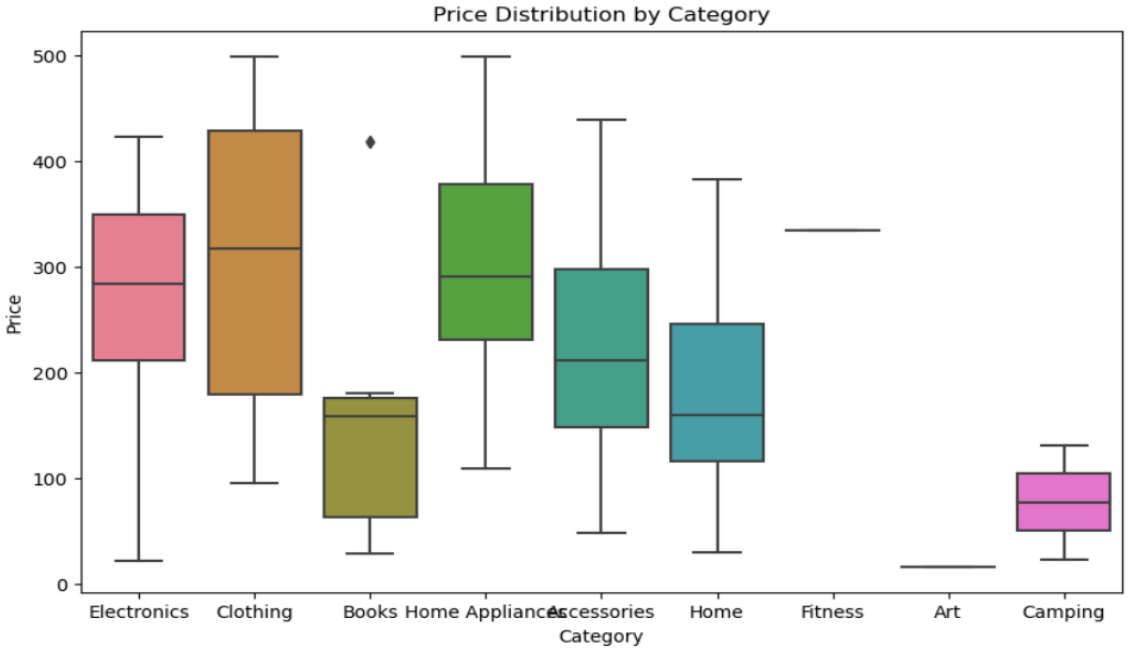
Product_ID	Product_Name	Category	Price
1	Laptop	Electronics	423
2	Smartphone	Electronics	174
3	Headphones	Electronics	406
4	Camera	Electronics	248
5	Television	Electronics	22

Fig 3: Sample Data of UserProfile

Category wise product count:-provides a clear overview of the distribution of products across various categories



Price Distribution by category:-The graph displays the distribution of prices for different product categories using a box plot. Provides a comparative analysis of price variations across different product categories



3) UserInteractions_data

Attribute	Description	Data Type
Interaction_ID	Interaction identifier	Numeric
Timestamp	Time of the interaction	Datetime
User_ID	User identifier	Numeric
Interaction_Type	Type of interaction	String
Product_ID	Product identifier	Numeric
Product_Name	Name of the product	String
Quantity	Quantity of the product	Numeric
Payment_Method	Payment method used	String

Interaction_ID	Timestamp	User_ID	Interaction_Type	Product_ID	Product_Name	Quantity	Payment_Method
1	2023-11-01 00:00:00	1	product_search	-	laptop	0.0	Not Applicable
2	2023-11-01 00:00:00	2	product_view	49	Camping Tent	0.0	Not Applicable
3	2023-11-01 00:00:00	3	product_search	-	shoes	0.0	Not Applicable
4	2023-11-01 00:00:00	4	product_search	-	electronics	0.0	Not Applicable
5	2023-11-01 00:00:00	5	product_search	-	phone	0.0	Not Applicable

Fig 4: Sample Data of UserInteractions_data

2.3 Data Cleaning

Removed outliers:- Removed outliers to enhance data quality and improve model performance by eliminating extreme values that could distort statistical analyses, ensuring a more accurate representation of the underlying data distribution.

```
def find_outliers(self, data):
    # Explain outliers for numerical attribute 'Price'
    outliers = data.filter(F.col('Price') > 1000)
    print("Outliers in Price:")
    outliers.show()
```

```
def find_outliers(self, data):
    # Explain outliers for numerical attribute 'Age'
    outliers = data.filter(F.col('Age') > 100)
    print("Outliers in Age:")
    outliers.show()
```

Null/empty/hyphen values:- Handled null values by replacing them with the mean, mitigating bias in analyses, maintaining data integrity, and ensuring consistency in data processing. This approach enhances the robustness of statistical computations and facilitates a more accurate representation of the dataset.

```
def preprocess_data(self, data):
    # Handle null values for numerical attribute 'Price'
    data = data.na.fill({'Price': data.select(F.mean('Price')).collect()[0][0]})
    return data
```

```
def preprocess_data(self, data):
    # Handle null values for numerical attribute 'Age'
    data = data.na.fill({'Age': data.select(F.mean('Age')).collect()[0][0]})
    return data
```

```
def preprocess_data(self, data: DataFrame) -> DataFrame:
    # Additional handling for null, empty, hyphen, or negative Product_ID
    return data.withColumn('Product_ID', F.when((F.col('Product_ID').isNull()) | (
    F.col('Product_ID') == '') | (F.col('Product_ID') == '-') |
    (F.col('Product_ID') < 0), 0).otherwise(F.col('Product_ID')))
```

```
df['Product_ID'].replace('-', np.nan, inplace=True)
```

2.4 Data Processing

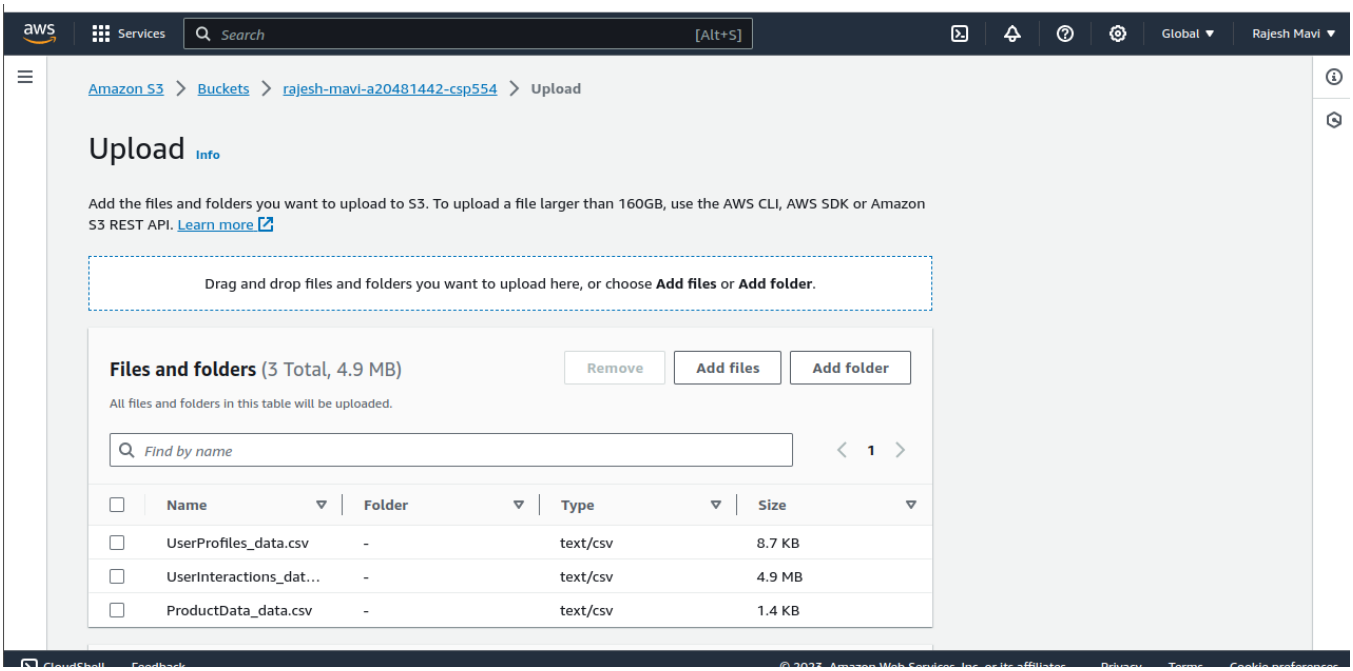


Fig 5: Creating a Bucket and added the data files.

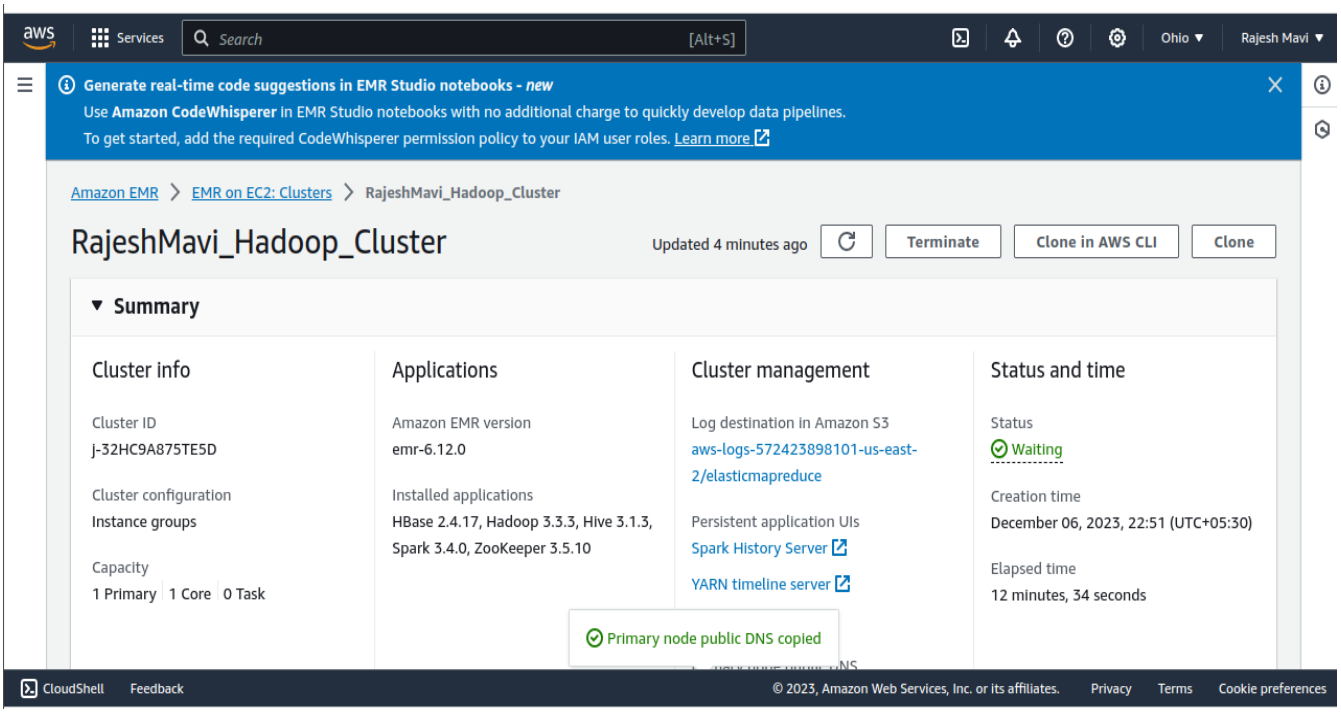


Fig 6: Started an EMR cluster to run jobs

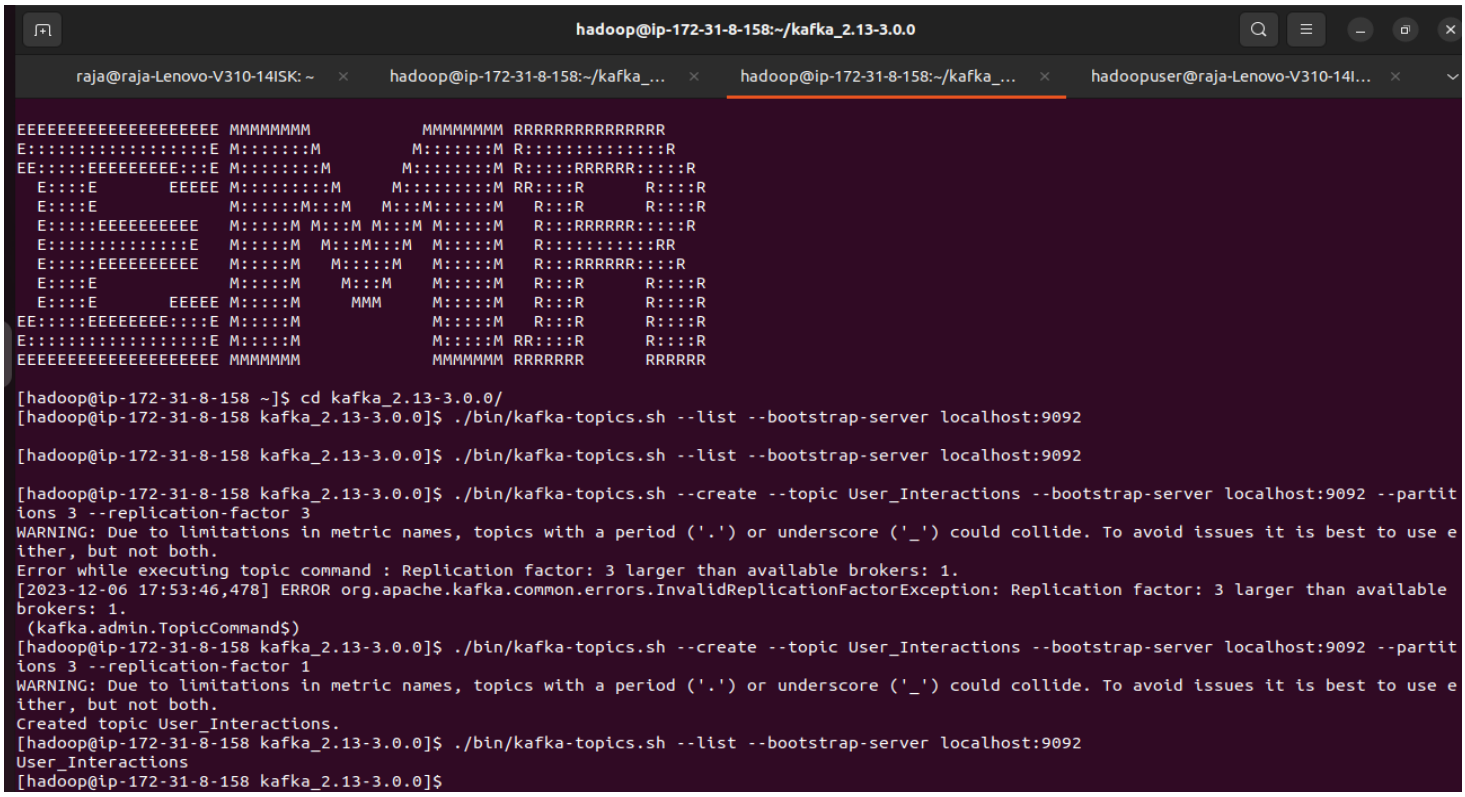


Fig 7: Creating kafka topic user_Interaction for stream processing

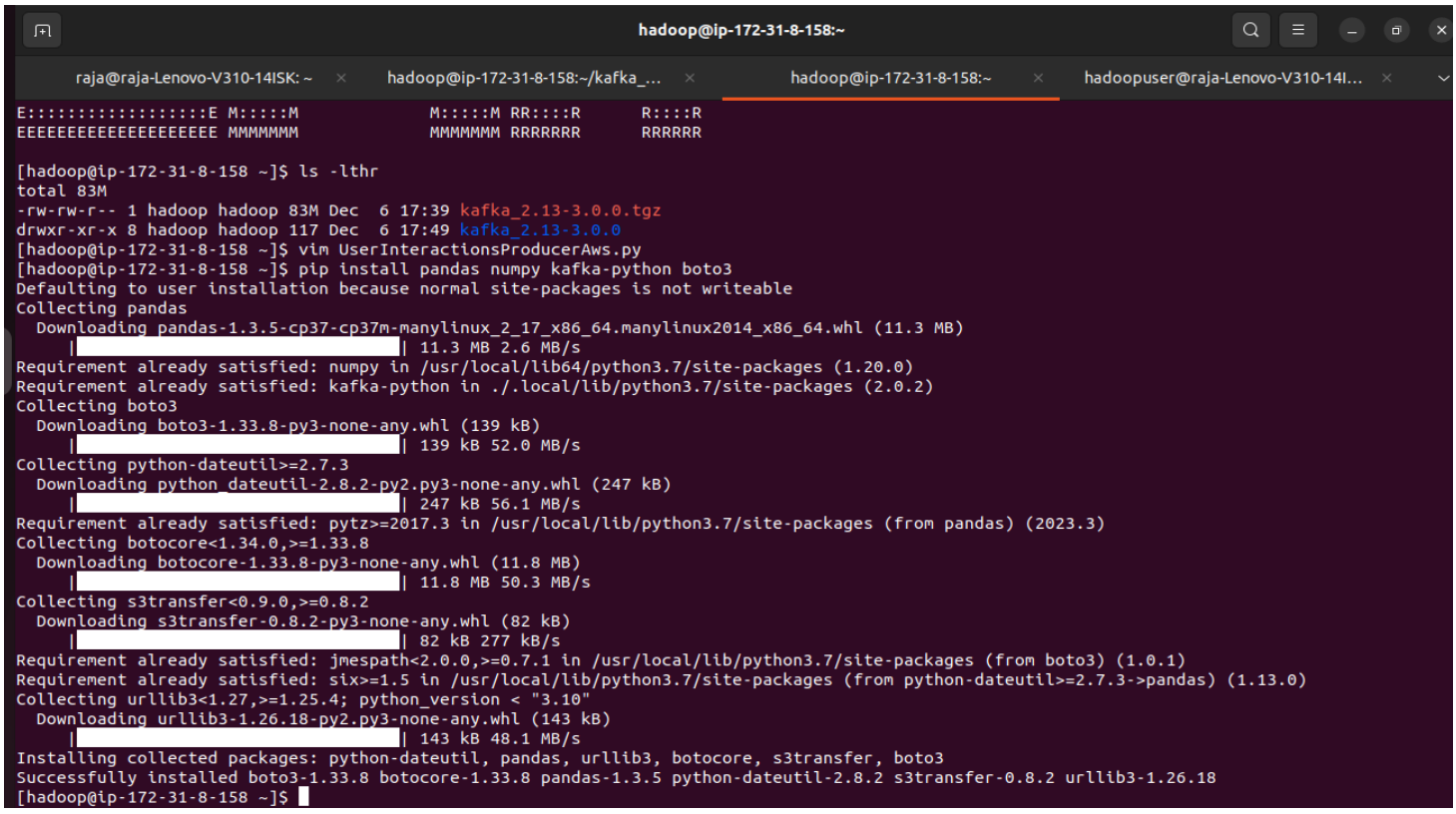


Fig 8: Installing pandas, numpy, kafka-python, boto3

```
[hadoop@ip-172-31-8-158 ~]$ ls -ltrh
total 88M
-rw-rw-r-- 1 hadoop hadoop  83M Dec  6 17:39 kafka_2.13-3.0.0.tgz
drwxr-xr-x 8 hadoop hadoop  117 Dec  6 17:49 kafka_2.13-3.0.0
-rw-rw-r-- 1 hadoop hadoop 3.4K Dec  6 18:14 UserInteractionsProducerAws.py
-rw-rw-r-- 1 hadoop hadoop 8.8K Dec  6 18:22 UserProfiles_data.csv
-rw-rw-r-- 1 hadoop hadoop 1.4K Dec  6 18:23 ProductData_data.csv
-rw-rw-r-- 1 hadoop hadoop 5.0M Dec  6 18:23 UserInteractions_data.csv
[hadoop@ip-172-31-8-158 ~]$ pwd
/home/hadoop
[hadoop@ip-172-31-8-158 ~]$ rm -rf UserInteractionsProducerAws.py
[hadoop@ip-172-31-8-158 ~]$ vim UserInteractionsProducerAws.py
[hadoop@ip-172-31-8-158 ~]$ pw
-bash: pw: command not found
[hadoop@ip-172-31-8-158 ~]$ pwd
/home/hadoop
[hadoop@ip-172-31-8-158 ~]$ rm -rf UserInteractionsProducerAws.py
[hadoop@ip-172-31-8-158 ~]$ vim UserInteractionsProducerAws.py
[hadoop@ip-172-31-8-158 ~]$ python UserInteractionsProducerAws.py
```

Fig 9: Running Kafka producer UserInteractionsProducerAWS

Fig 10: User interaction kafka topic data

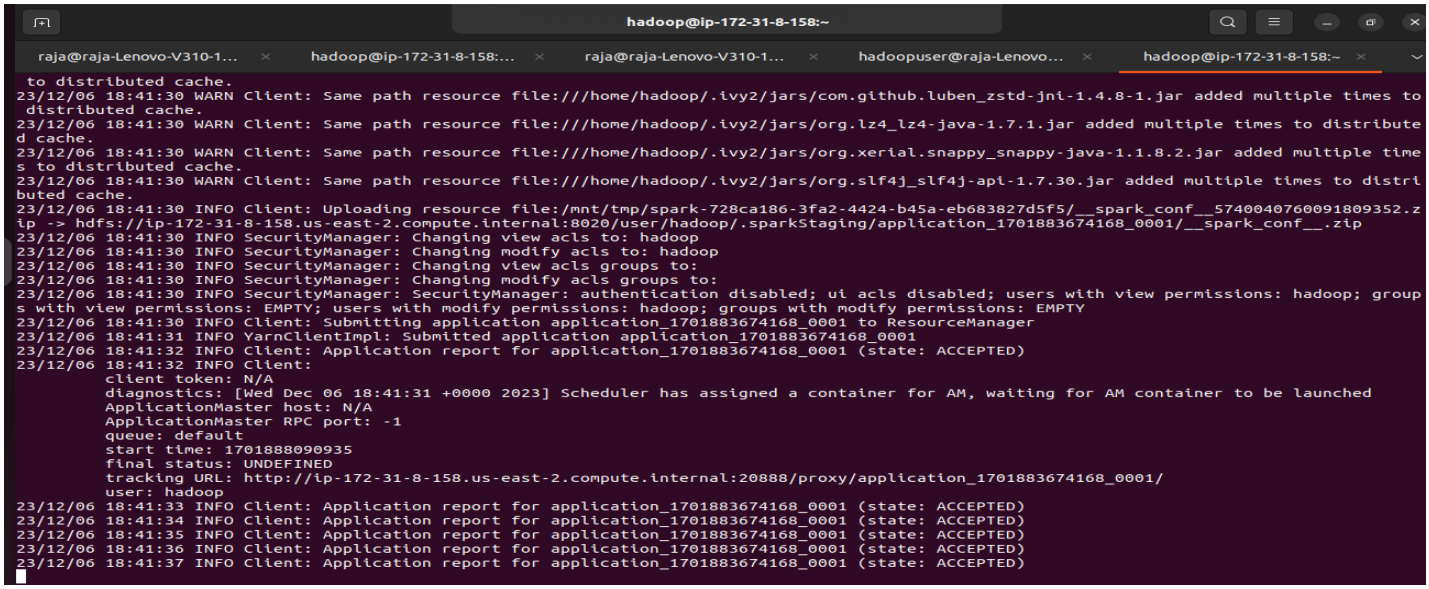


Fig 11: Streaming kafkasparkstreamingCSVWrite job

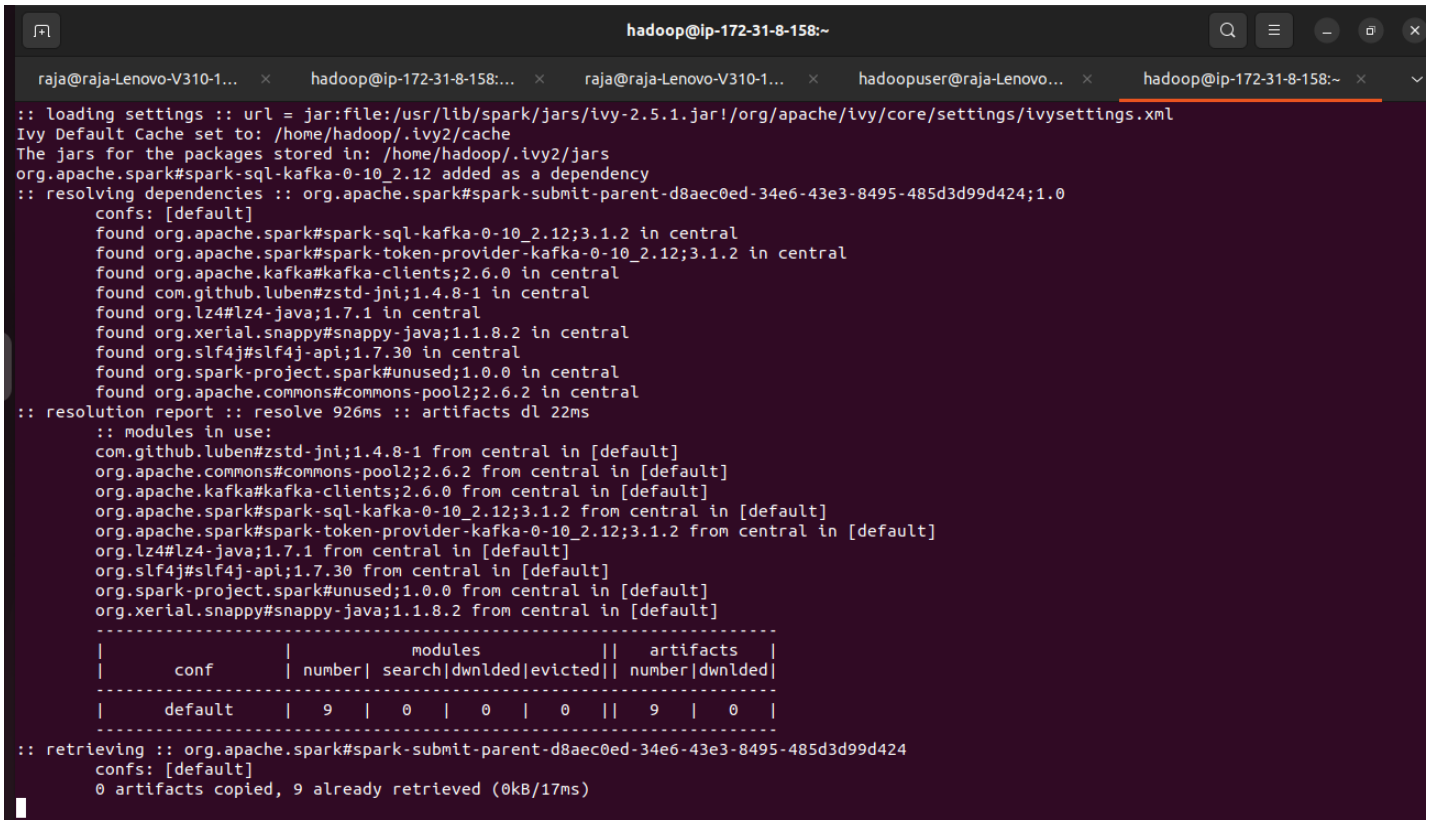


Fig 12: Job starting

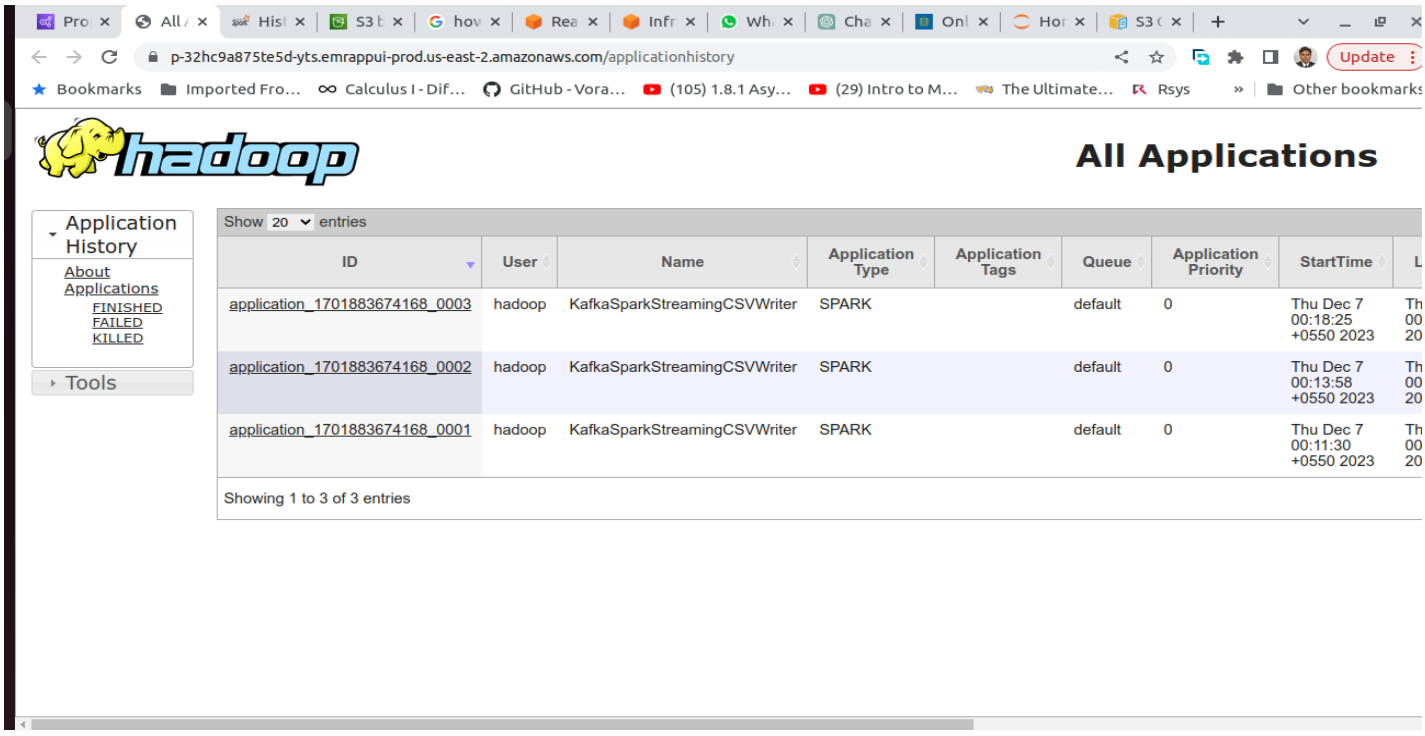


Fig 13: Hadoop Yarn UI to track the spark streaming job

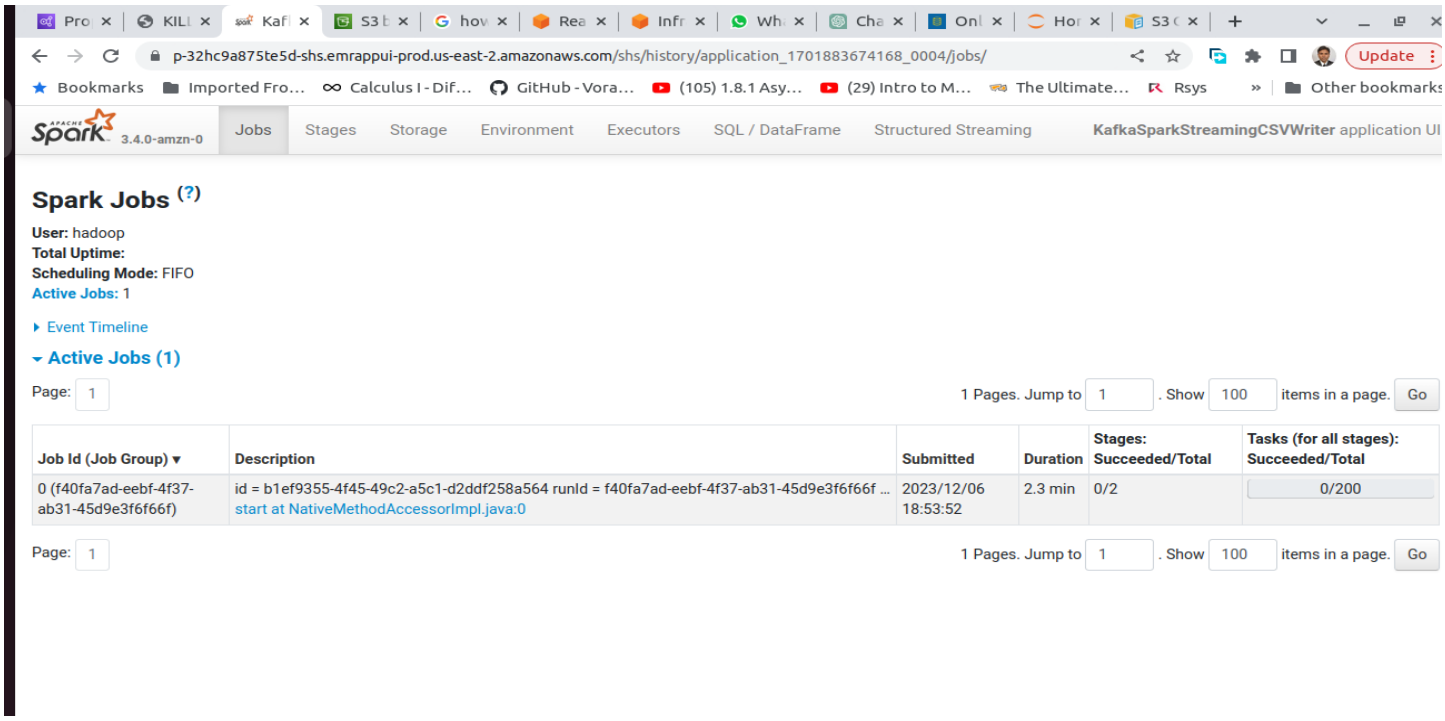


Fig 14: List of tasks spark job is assigned to finish

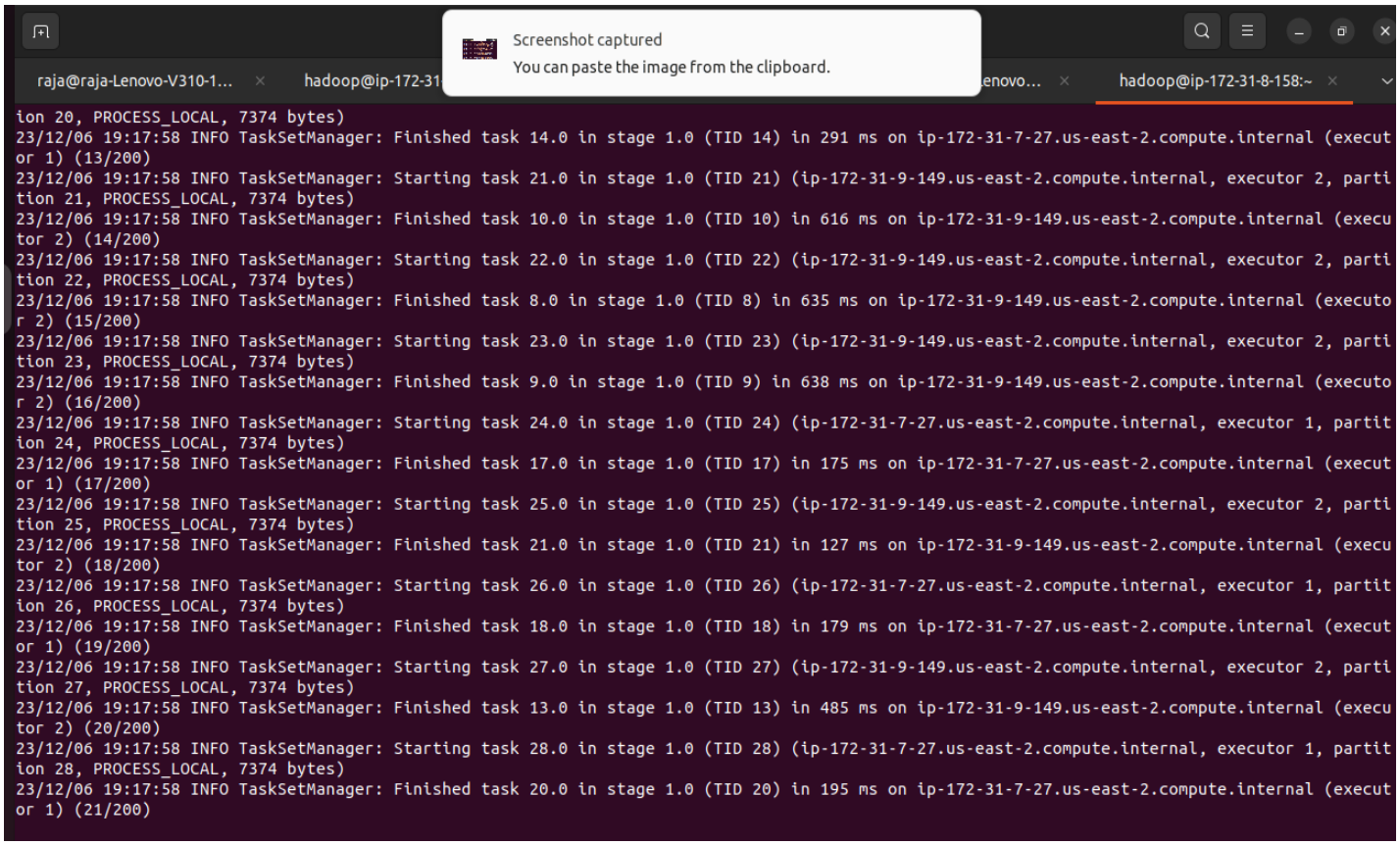


Fig 15: Logs of spark jobs showing completion of the job

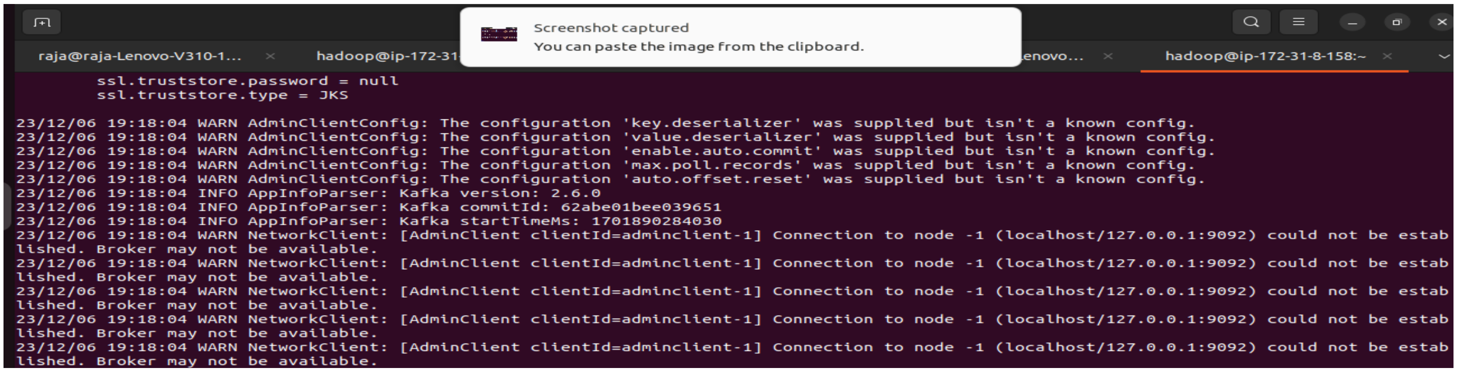


Fig 16: Showing the connected nodes

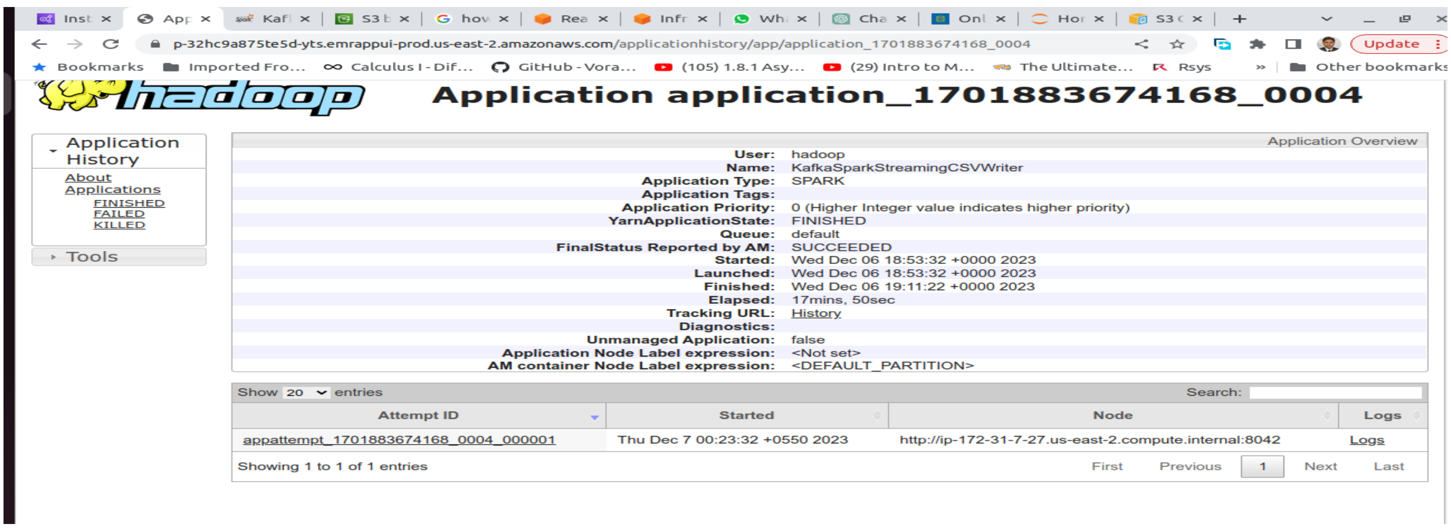


Fig 17: Complete information about the spark job

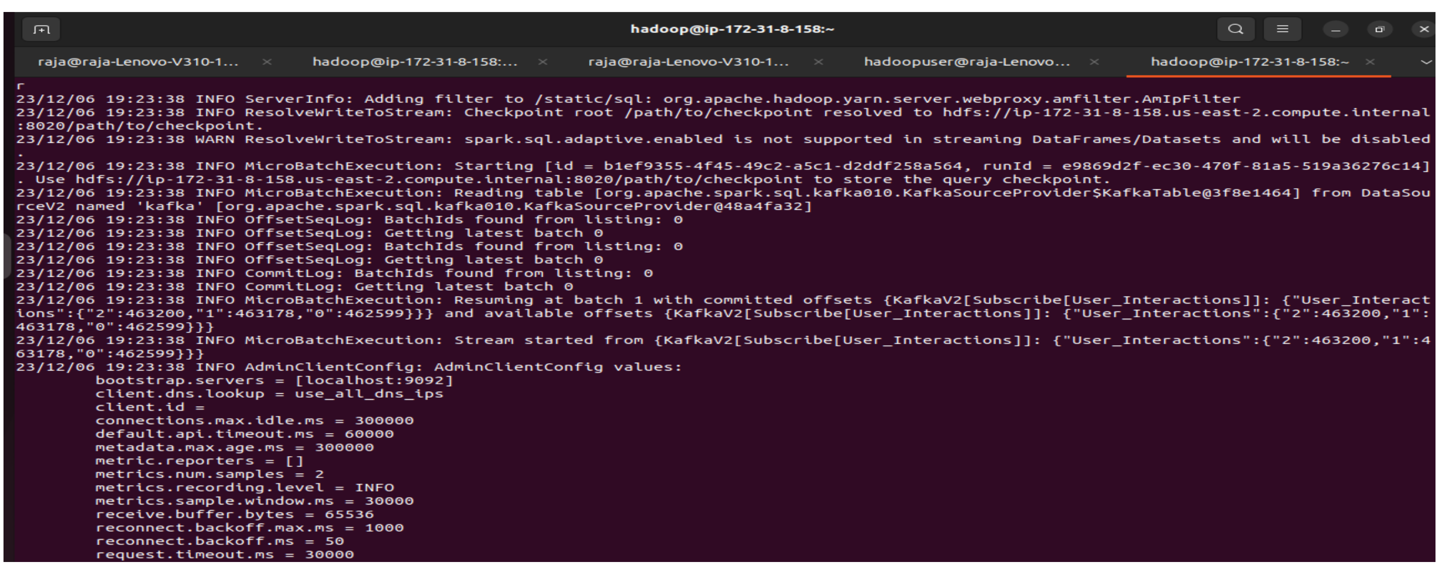


Fig 18: Reading the data from the Kafka topic by the streaming spark job

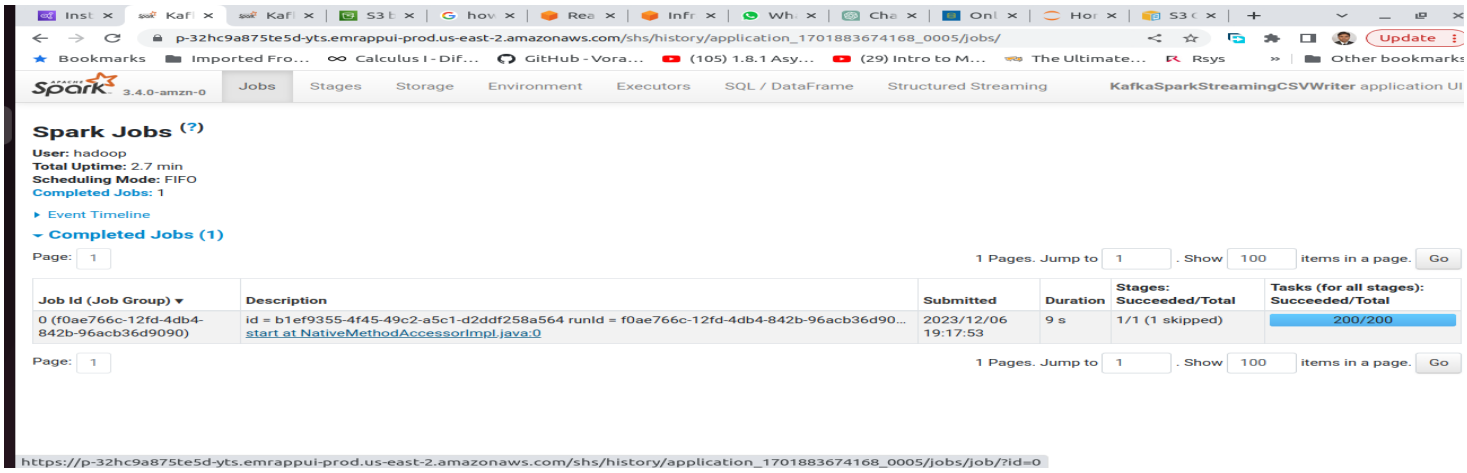


Fig 19: Tracking the list of tasks that that has executed successfully.

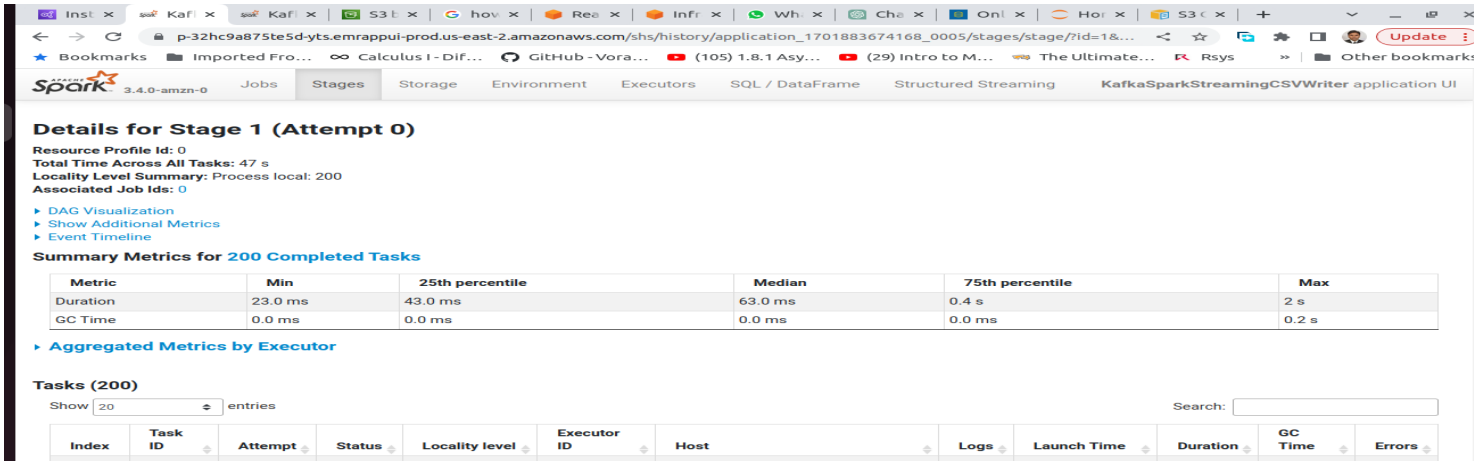


Fig 20: Details of spark job stages

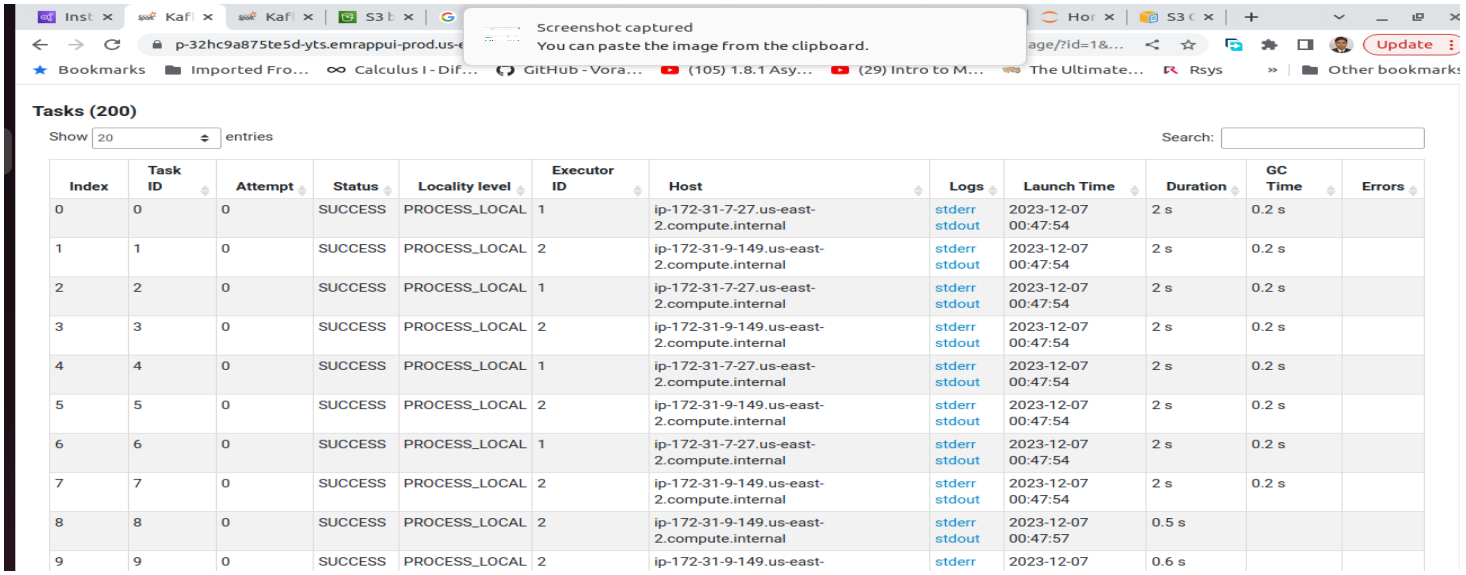


Fig 21: All the details of number of tasks executed on a node

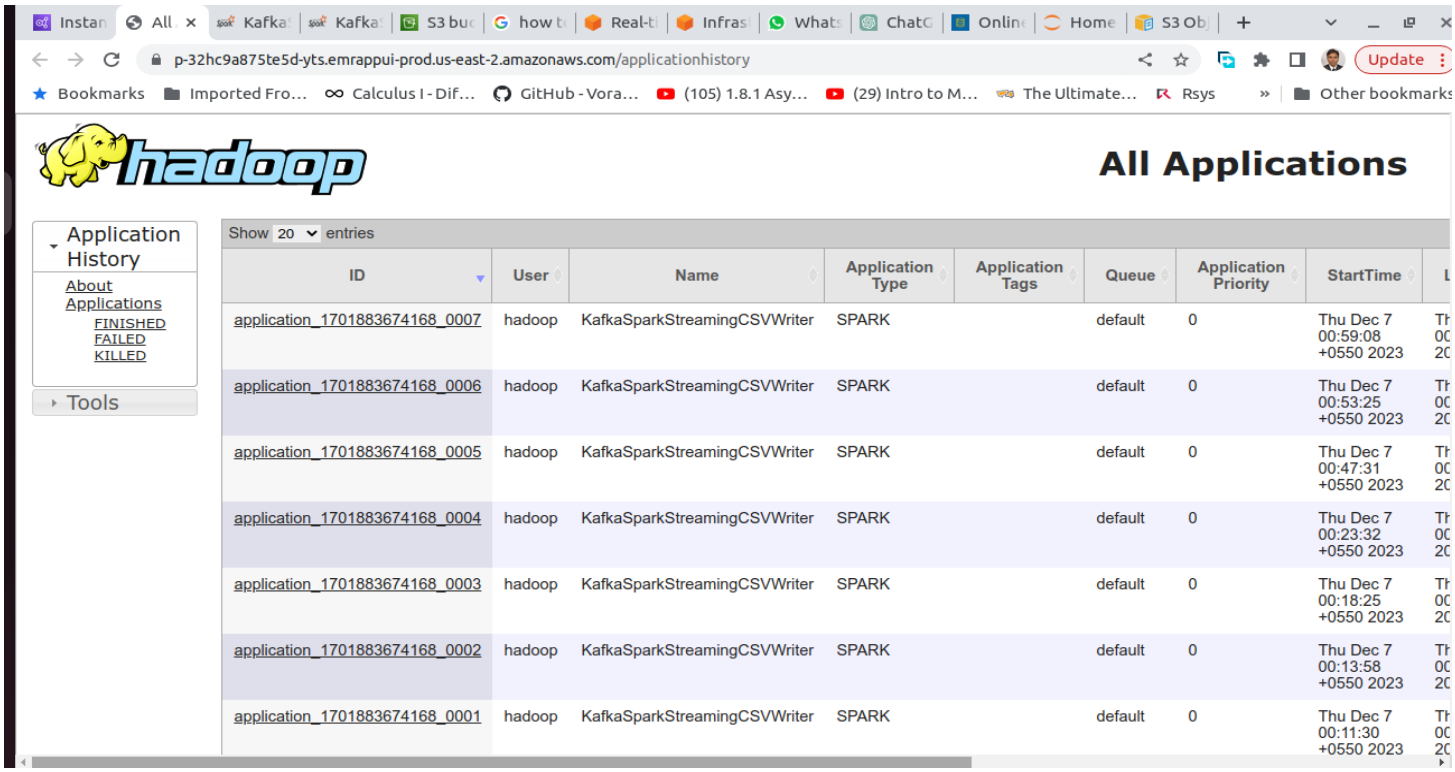


Fig 22: History of the all streaming spark jobs

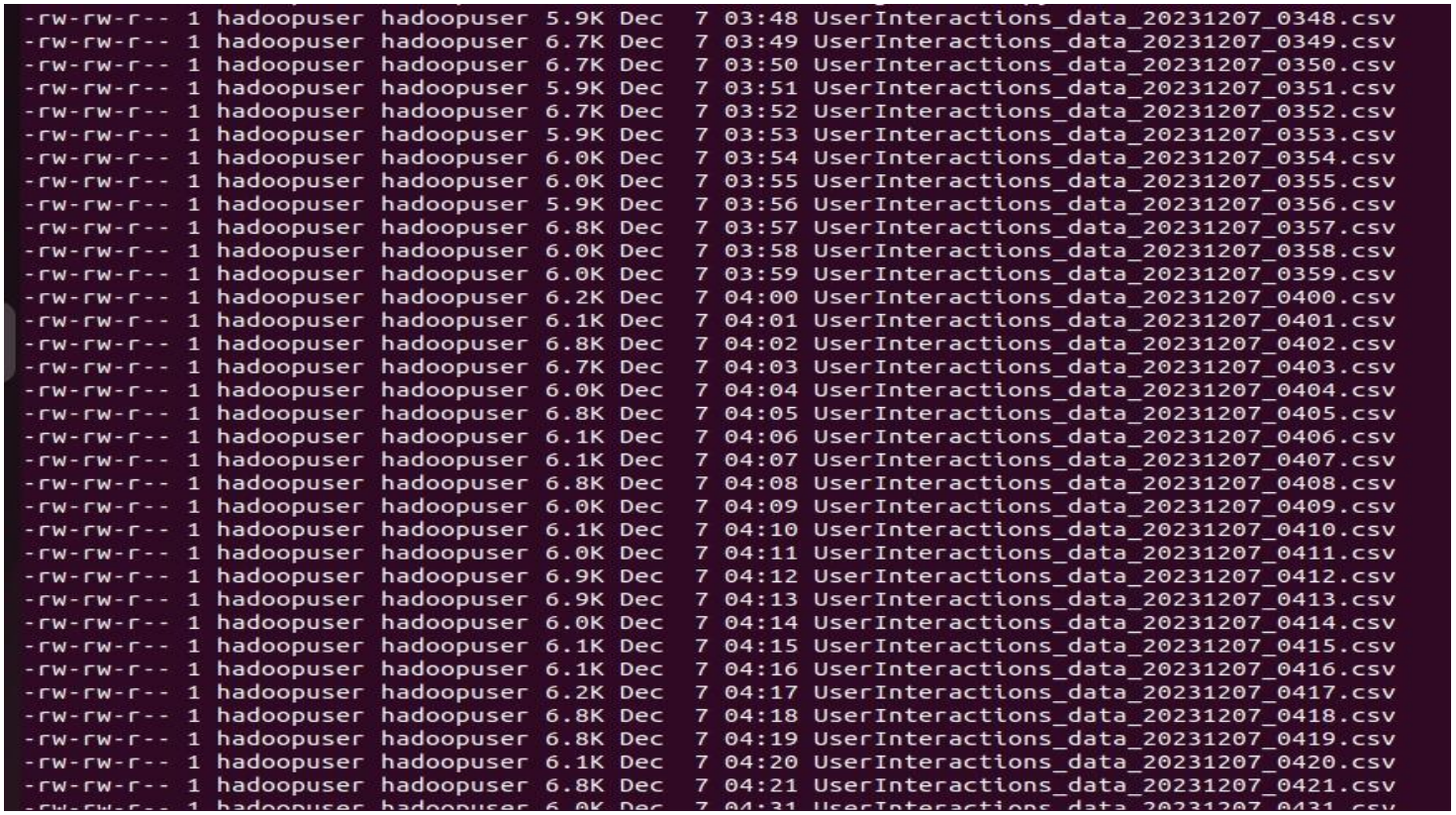
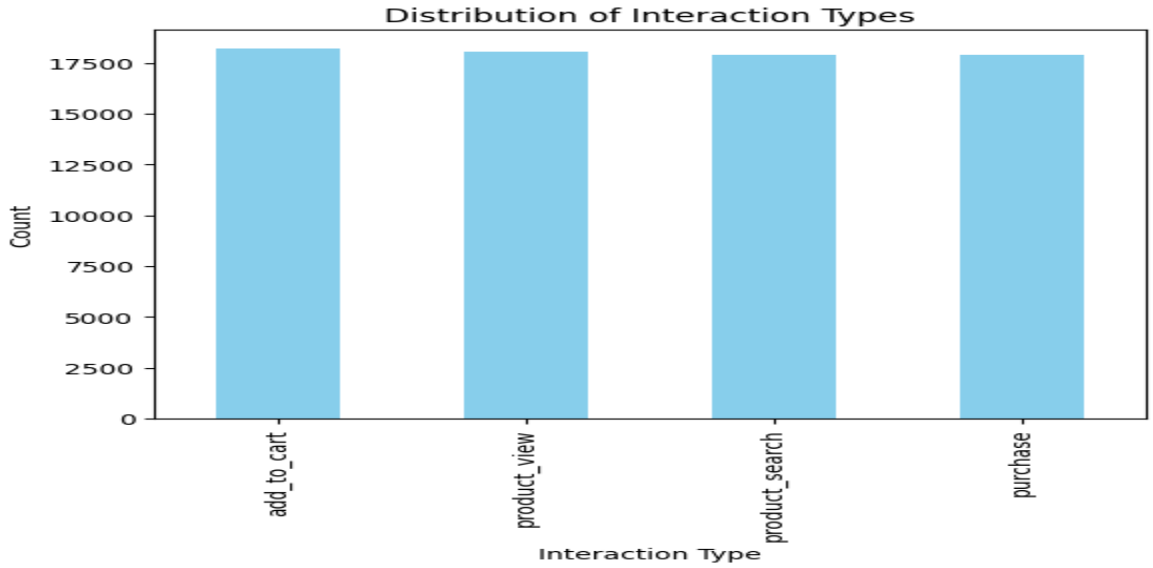


Fig 23: List of all files generated

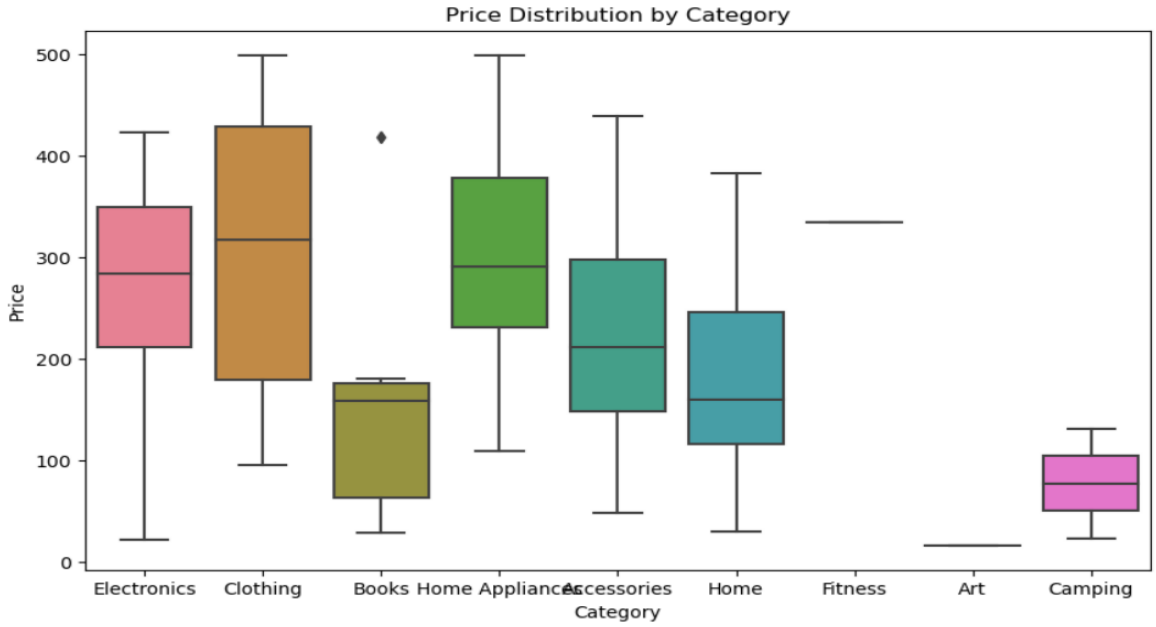
2.5 Data Visualization

Distribution of interaction types: The graph illustrates the distribution of user interactions, including "Add to Cart," "Product View," "Product Search," and "Purchase," with the count represented on the y-axis. This visual representation provides insights into the varying engagement levels across these key interaction types, offering a comprehensive overview of user behavior within the given context.

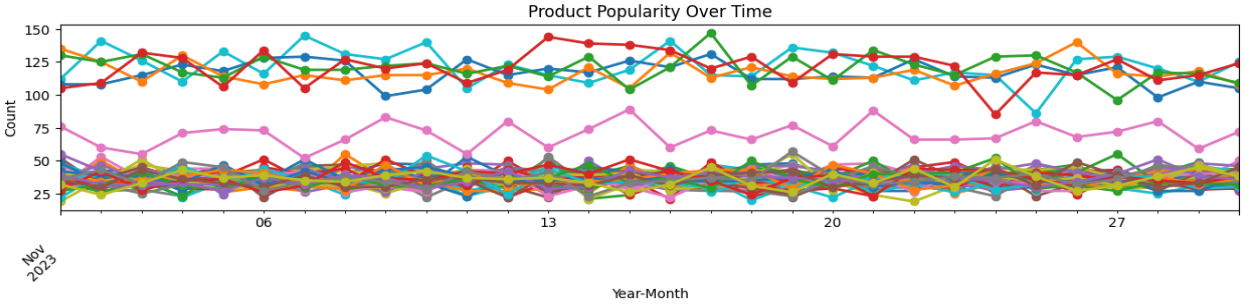


Distribution of quantity: The graph depicts a frequency distribution based on quantities, with the x-axis representing different quantity values and the y-axis indicating the frequency of occurrences. This visual representation offers insights into the distribution pattern and prevalence of specific quantity values within the analyzed dataset.

Quantity by interaction type: The box plot graph illustrates the distribution of quantities for different interaction types, emphasizing a focus on "Purchase" and "Add to Cart." The plot highlights the central tendency, spread, and potential outliers in the quantity data for these specific interactions, with a notable emphasis on the absence of quantities for "Product Search" and "Product View," marked as zero. This visual analysis provides a understanding of the variation in quantities associated with distinct user interactions, particularly emphasizing the transactional nature of "Purchase" and "Add to Cart."



Products Popularity over time: The graph showcases the popularity of products over a one-month period, utilizing the x-axis for time intervals and the y-axis to represent order counts. This visual representation allows for a temporal analysis of how various products have performed in terms of sales or orders, offering insights into their relative popularity dynamics over the specified timeframe.



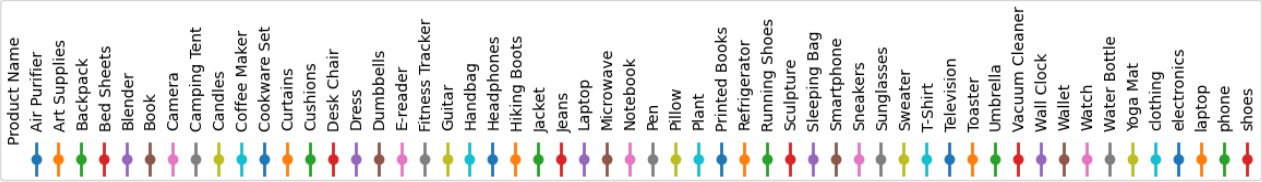
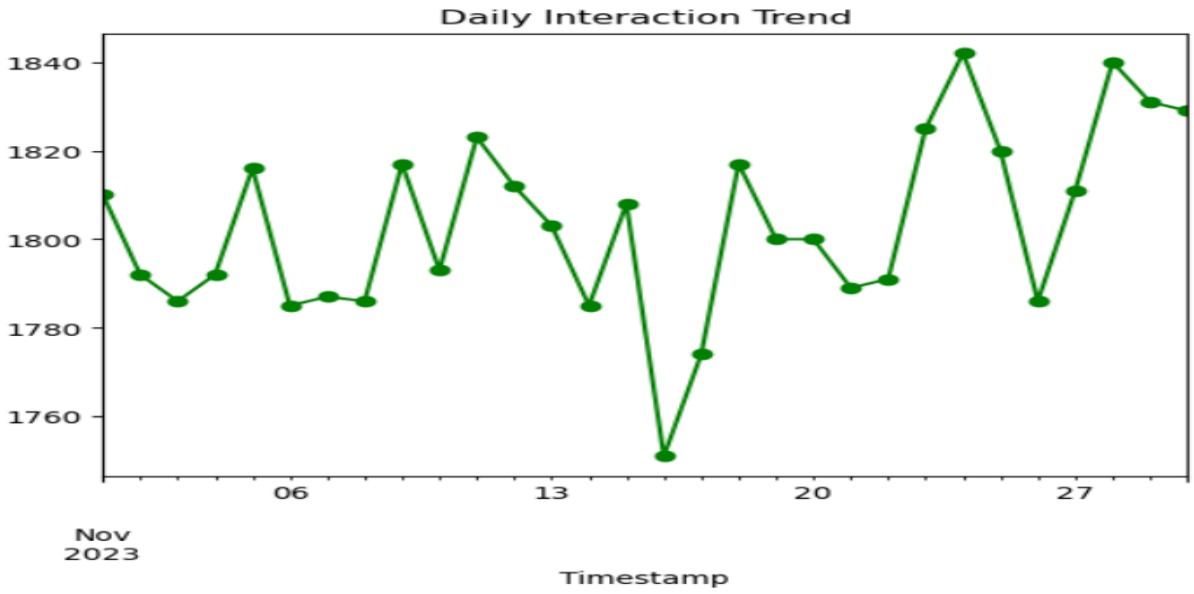


Fig 24: Legend for Products Popularity over time

Daily Interaction trend: This graph aims to depict how user interactions have fluctuated on a daily basis over the past month, offering a visual representation of the overall trend. Visualizing daily interaction trends is useful for businesses to track user engagement patterns, optimize marketing strategies, and make informed decisions about product performance and resource allocation.



Product Popularity Forecasting (MACHINE LEARNING); Used the ARIMA (AutoRegressive Integrated Moving Average) time series forecasting model from the statsmodel library to forecast the popularity of three products (laptop, phone, shoes) based on historical data, plots the historical data along with the forecasted values, and visualizes the product popularity trends over time. This visualization provides insights into the predicted trends and variations in the popularity of the specified products, aiding in decision-making and planning based on anticipated future product demand.

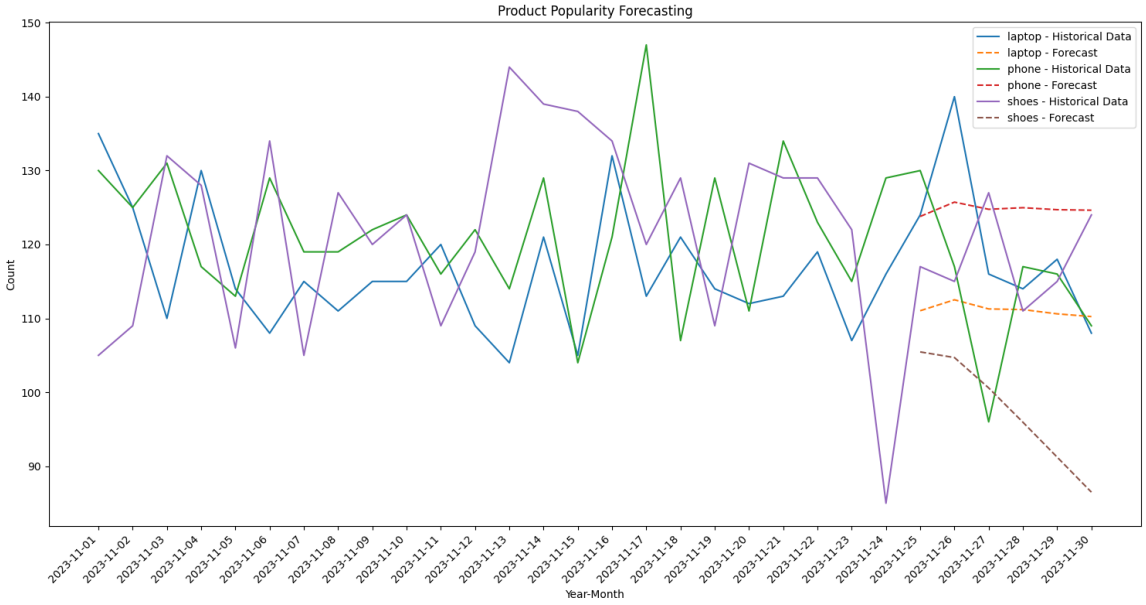


Fig 25: Product Popularity Forecasting

III: DIFFICULTIES FACED

1. Managing real-time data ingestion from S3, navigating challenges related to data consistency, latency, and optimizing transfer protocols.
2. Architecting Spark streaming jobs for AWS, addressing complexities in scalability, fault tolerance, and resource allocation across heterogeneous nodes.
3. Customizing an EMR cluster involves intricate configuration, dependency management, and the optimization of resource utilization.
4. Encountered challenges while configuring and publishing JSON-formatted messages to a Kafka topic. Additionally, faced difficulties in reading these messages within a PySpark Streaming job for the purpose of creating minute-wise aggregations.
5. Conducting a literature review poses challenges in identifying pertinent research papers on streaming data processing, custom cluster design, and live data from S3 for a comprehensive scholarly evaluation.
6. Considered various tools for distributed and real-time processing, including Hadoop MapReduce, Apache Flink, and Apache Storm. After meticulous evaluation, the chosen tool is PySpark Streaming with Kafka, leveraging PySpark's simplicity for real-time processing and Kafka's efficiency for event streaming.

IV: CONCLUSION

In conclusion, our project has successfully delivered a fully functional simulation that not only accommodates user interactions but also ensures efficient storage of both structured and unstructured data.

The implementation of real-time data processing has empowered continuous user behavior analysis, offering valuable insights into patterns, preferences, and product performance. With user segmentation, we gain a detailed understanding of different user types, while visualizations and reports provide actionable information for decision-making. Scalable system design positions us well to handle increasing data volumes in the future. Overall, this project establishes a robust foundation for comprehensive data-driven insights and user-focused strategies.

V: FUTURE SCOPE

1. Advanced Analytics: Explore and implement advanced analytics techniques, such as machine learning algorithms, for deeper insights into user behavior, predictive analytics, or anomaly detection.

2. Integration with External Data Systems: Extend the system's capabilities by integrating with external data sources or third-party APIs to enrich the dataset and provide a more comprehensive analysis.

3. Real-time Decision Support: Strengthen the decision-making process by incorporating real-time decision support mechanisms, allowing stakeholders to make informed decisions on the fly.

4. Behavioral Economics Analysis: Incorporate principles of behavioral economics into the analysis to understand how psychological factors influence user decisions, leading to more effective design and engagement strategies.

VI: REFERENCES

[1] Python

[2] Apache Hive

[3] Matplotlib

[4] Apache PySpark

[5] Apache Kafka

[6] Hadoop

[7] Zookeeper

[8] AWS

[9] Le, T. M., & Liaw, S.-Y. (2017). *Effects of Pros and Cons of Applying Big Data Analytics to Consumers' Responses in an E-Commerce Context.* Sustainability, 9(5), 798. Mdpi

[10] Evangelin, R., & Shanmugam, V. (2022). *Mechanism of Big Data Analytics in Consumer Behavior on Online Shopping.* May 2022. ResearchGate

[11] Rongrui Yu, Chunqiong Wu, Bingwen Yan, Baoqin Yu, Xiukao Zhou, Yanliang Yu, Na Chen, (2021). "*Analysis of the Impact of Big Data on E-Commerce in Cloud Computing Environment*", Article ID 5613599. Hindawi

[12] Jui-Chan Huang, Po-Chang Ko, Cher-Min Fong, Sn-Man Lai, Hsin-Hung Chen, Ching-Tang Hsieh,(2021) "*Statistical Modeling and Simulation of Online Shopping Customer Loyalty Based on Machine Learning and Big Data Analysis*". Hindawi

[13] Asle Fagerstrøm, Erik Arntzen, Gordon Foxall, (2011). "*A study of preferences in a simulated online shopping experiment*". ResearchGate