Title: Build an Analytical Platform for eCommerce using AWS Services

B. Ajith Kumar Reddy - 2200080242

CLOUD SERVERLESS COMPUTING–22CEC3305A

Section 32

Under the guidance of

Dr. S.Kavitha

Abstract

The project, titled "Build an Analytical Platform for eCommerce using AWS Services," aims to develop a modern, cloud-based data analytics solution that enables eCommerce businesses to make faster, smarter, and more informed decisions. In today's highly competitive digital retail landscape, companies must continually analyze vast amounts of data to understand customer behavior, monitor sales performance, and optimize their operations. This platform utilizes a combination of **AWS services** to build a robust and scalable analytics workflow.

Data generated from various e-commerce activities—such as order transactions, customer clicks, product searches, inventory updates, and user reviews—is collected and stored in **Amazon S3**, serving as a cost-effective and durable data lake. Using **AWS Glue**, the raw data is automatically crawled, transformed, and cleaned through ETL (Extract, Transform, Load) jobs. The transformed data is then loaded into **Amazon Redshift**, a powerful data warehousing service that supports complex queries and large-scale data analytics.

To make the insights easily understandable and accessible, **Amazon QuickSight** is used for building dynamic dashboards and visual reports. These visualizations provide key metrics such as top-selling products, customer purchase patterns, sales trends, and regional performance, allowing stakeholders across marketing, finance, and operations teams to track and act on business performance in real time.

Additionally, **AWS Lambda** is used to trigger automation tasks within the pipeline—such as data updates or scheduled jobs—eliminating the need for dedicated servers and reducing operational overhead. The solution ensures high scalability, fault tolerance, and performance while keeping infrastructure management minimal.

Modules

1. Data Ingestion Module

This module is responsible for capturing and pushing e-commerce data into the analytical pipeline. Data is simulated using AWS CloudShell with Python scripts and Boto3 SDK to mimic customer activity such as browsing, cart additions, purchases, etc.

Tools Used: AWS CloudShell, Boto3, Amazon S3, Amazon Kinesis Data Streams

Functionalities:

- Simulates real-time transactional and behavioral data.
- Uploads static files (CSV) into S3 for batch analytics.
- Streams live data to Kinesis for real-time analytics

2. Real-Time Analytics and Stream Processing Module

This module handles real-time analytics using streaming services to process customer activity on-the-fly. It allows near-instant detection of user trends, fraudulent behavior, inventory depletion, and promotional responses.

Tools Used: Amazon Kinesis Data Analytics (Flink), AWS Lambda

Functionalities:

- Applies real-time filters, aggregations, and pattern detection using Apache Flink.
- Processes data instantly and invokes AWS Lambda functions.
- Helps trigger alerts, store analytics in DynamoDB, and update logs.

3. ETL and Batch Processing Module

In this module, larger datasets collected over time are cleaned, transformed, and stored in a more analytically friendly format for in-depth analysis. This enables long-term trend analysis, customer segmentation, and behavior insights.

Tools Used: AWS Glue, Amazon S3

Functionalities:

- AWS Glue performs automated schema detection and data transformation.
- Stores final datasets in formats like Parquet in S3 for efficient querying.

4. Storage and Query Module

Data, both structured and unstructured, is stored securely and is readily accessible for analysis. Different storage types support a range of queries from real-time lookups to batch reports.

Tools Used: Amazon S3 (raw + transformed), Amazon DynamoDB

Functionalities:

- Stores simulation logs, transaction data, and analytic output.
- Uses DynamoDB for fast key-value storage and quick retrievals.
- Acts as the backbone for reporting and visualization modules.

5. Monitoring, Logging & Alerting Module

This module ensures system health and performance are consistently monitored. It provides visibility into failures, usage spikes, or abnormal activities. It also integrates alerting mechanisms for real-time response.

Tools Used: Amazon CloudWatch, AWS Lambda, Amazon SNS

Functionalities:

- Lambda sends logs and metrics to CloudWatch for tracking.
- SNS sends notifications (SMS/email) to stakeholders upon defined triggers (e.g., fraud detection, system errors).
- Helps in audit, debugging, and performance tuning.

6. Reporting & Visualization Module

This module is the final stage of the pipeline where insights are visually represented. Business intelligence dashboards and charts are created to help decision-makers understand trends and patterns.

Tools Used: Amazon QuickSight

Functionalities:

- Visualizes transformed data for dashboards on revenue, customer behavior, sales performance, etc.
- Allows scheduled report generation and sharing with stakeholders.
- Enables self-service BI without the need for deep technical skills.

Event-Driven Architecture Overview

The platform architecture provides a complete end-to-end pipeline starting from data simulation to business analytics, all within a serverless, scalable AWS environment. It supports both real-time and batch analytics.

1. **Data Generation and Ingestion**

- The simulation starts in **AWS CloudShell**, where Python scripts using Boto3 simulate eCommerce activities like orders, product views, and payments.
- Data is directed into:
    - **Amazon S3** for CSV-based batch data.
    - **Amazon Kinesis Data Streams** for continuous real-time ingestion.

2. **Stream Processing & Analytics**

- **Amazon Kinesis Data Analytics (Flink)** processes real-time data from Kinesis Streams.
- Flink jobs apply transformations (like windowed aggregations and pattern detection).
- Output is sent to:
    - **AWS Lambda** for further logic execution.
    - Lambda functions write to:
        - **Amazon DynamoDB** for structured storage.
        - **Amazon CloudWatch** for logs and monitoring.
        - **Amazon SNS** to notify about specific events (e.g., purchase spike, suspicious activity).

3. Batch Pipeline and ETL

- Raw streaming data is also buffered by **Kinesis Firehose**, which stores it in Amazon S3.
- **AWS Glue** crawls the S3 bucket, performs ETL, and writes transformed data in Parquet format into a separate S3 bucket for analytics.

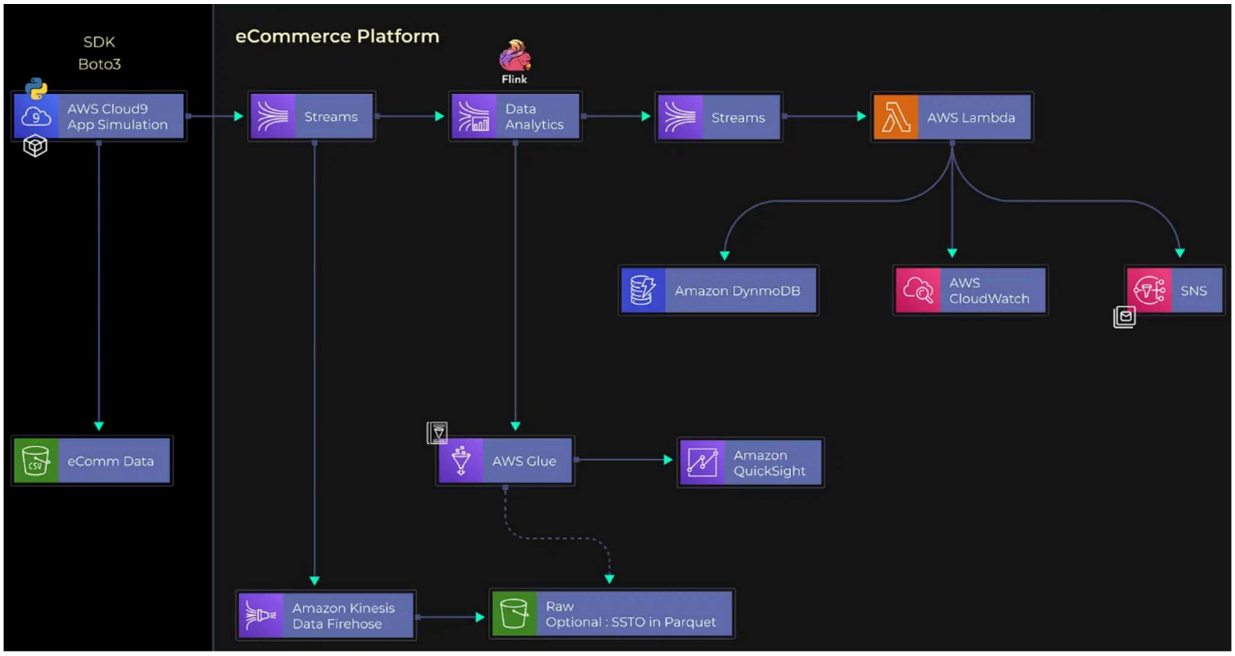
4. Business Intelligence & Visualization

- The final transformed data in S3 is connected to **Amazon QuickSight**.
- QuickSight dashboards present interactive charts and reports on key business KPIs:
    - Sales trends
    - Customer engagement
    - Inventory performance
    - Conversion rates

5. Key Architectural Features

- **Fully Serverless**: No manual provisioning; scalable architecture.
- **Real-Time + Batch Integration**: Handles both high-velocity streams and bulk data.
- **Secure and Reliable**: Uses AWS IAM, encryption (SSE), and CloudWatch for logging.
- **Cost-Effective**: Pay-as-you-go model; uses managed services like Glue and QuickSight

Architecture diagram



Services Used in the Project:

1. **Amazon Kinesis Data Streams**

   Used to ingest high-speed streaming data from eCommerce platforms, such as user clicks, product views, and transactions. It enables real-time data collection and further processing.

   Example: Capturing user browsing behavior as soon as it happens on the platform.

2. **Amazon Kinesis Data Analytics**

   This service is used to analyze and transform streaming data on the fly using SQL. It processes incoming data from Kinesis Streams and extracts meaningful insights, such as top-selling products or spikes in user traffic.

   Example: Detecting sudden increases in demand for a product in real time.

3. **AWS CloudShell**

   A browser-based terminal used to write and execute scripts for simulating real-time data generation using AWS CLI and Python. This eliminated the need for EC2 or S3-triggered uploads during development.

   Used for convenience to simulate data instead of deploying separate compute resources.

4. **AWS Lambda**

Lambda is triggered by Kinesis Analytics to act on processed data. It helps route the analyzed data to storage or notification systems, allowing for event-driven automation without provisioning servers.

Example: Automatically inserting processed data into DynamoDB or logging alerts to CloudWatch.

5. **Amazon DynamoDB**

A fast and flexible NoSQL database service used to store real-time metrics such as user activity stats, order counts, or inventory summaries. Ideal for quick reads/writes in milliseconds.

Enables rapid querying of up-to-date analytics for the admin panel or monitoring.

6. **Amazon CloudWatch**

Used for monitoring system behavior, collecting logs, and setting up alarms. It plays a crucial role in observing data flow and troubleshooting errors within Lambda, Kinesis, and Glue.

Example: Alerting if the incoming data rate exceeds the threshold or if Lambda fails.

7. **Amazon S3 (Simple Storage Service)**

Used for storing batch data files in formats like CSV or JSON. These files represent historical sales records, customer databases, or transaction logs.

Acts as the central storage for offline analytics and large datasets.

8. **AWS Glue**

A serverless data integration service used to crawl, catalog, and prepare batch data stored in S3. It helps define a unified schema, making the data ready for querying via Athena or visualization via QuickSight.

It bridges S3 storage and Athena querying by creating a data catalog.

9. **Amazon Athena**

Used to run SQL queries directly on S3 data without the need to set up a traditional database. It provides fast, serverless querying capability for batch analytics.

Example: Running queries to identify monthly top-performing product categories.

10. **Amazon QuickSight**

A powerful business intelligence tool used to create dashboards and visualizations for stakeholders. It pulls data from Athena or DynamoDB to present interactive insights.

Final visual output to the end-user — showing trends, patterns, and predictions.

Lambda Function Code:

```
import base64
import json
import boto3
from decimal import Decimal  # Add this import

dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('EcommOrders')

sns = boto3.client('sns')
SNS_TOPIC_ARN = 'arn:aws:sns:your-region:your-account-
id:ecomm-alerts'  # Replace if using alerts

def lambda_handler(event, context):
    for record in event['Records']:
        payload =
base64.b64decode(record['kinesis']['data']).decode('utf-8')
        data = json.loads(payload)

        if not data.get('InvoiceNo'):
            continue

        try:
            item = {
                'InvoiceNo': str(data.get('InvoiceNo')),
                'StockCode': str(data.get('StockCode', '')),
                'Description': str(data.get('Description', '')),
                'Quantity': int(float(data.get('Quantity', 0))),
                'InvoiceDate': str(data.get('InvoiceDate', '')),
                'UnitPrice': Decimal(str(data.get('UnitPrice', 0.0))),  #
Use Decimal
                'CustomerID': str(data.get('CustomerID', '')),
                'Country': str(data.get('Country', ''))
            }

            table.put_item(Item=item)

            if item['UnitPrice'] > Decimal('1000'):
                sns.publish(
```

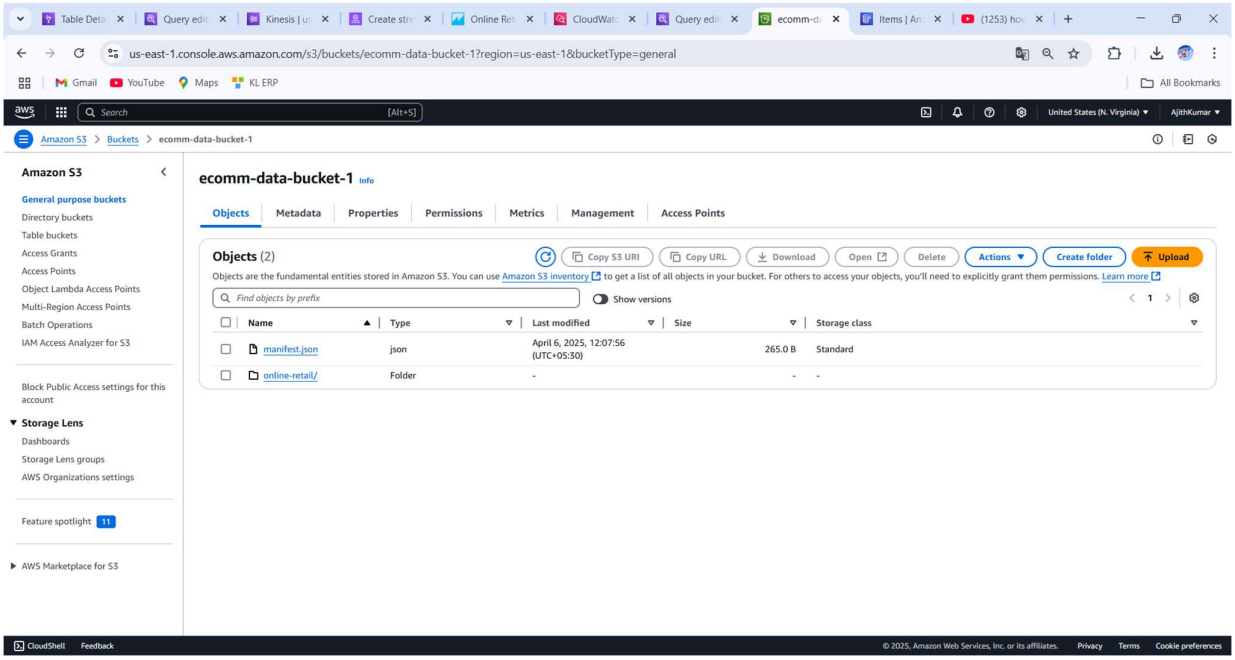
```
                TopicArn=SNS_TOPIC_ARN,
                Message=f"High value order detected:
{item['InvoiceNo']}",
                Subject="Alert: High Order"
            )

        except Exception as e:
            print(f"Error inserting: {e}")
```

Process and Screenshots

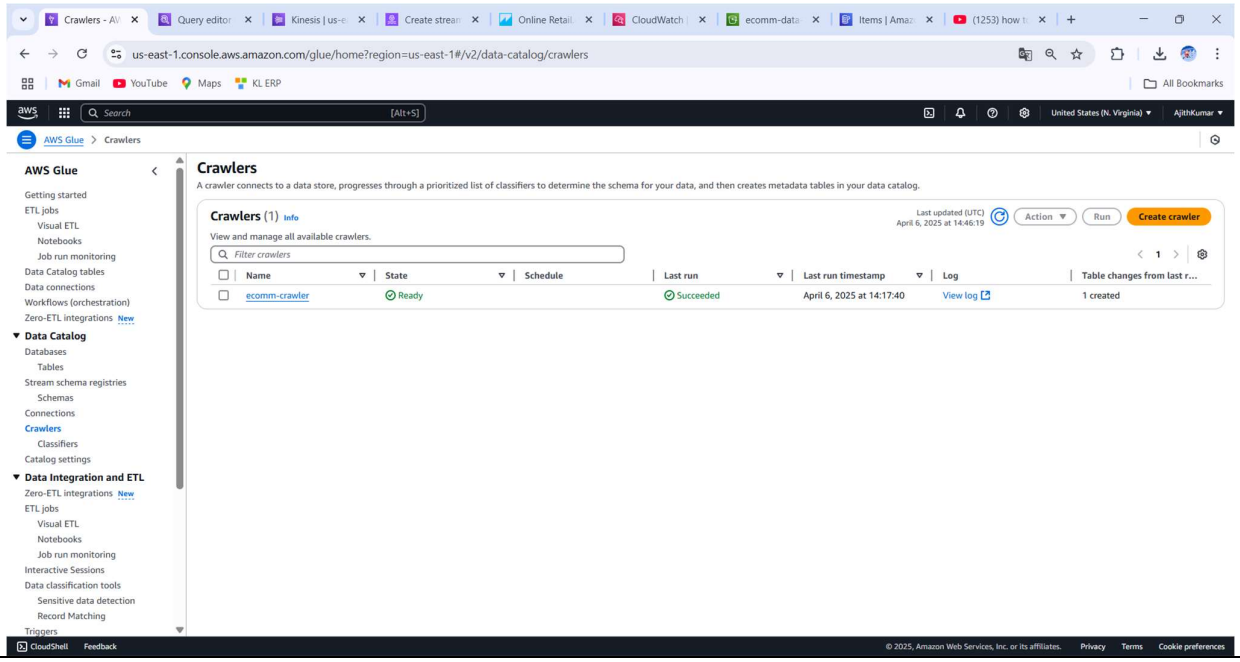
➢ Upload Dataset to S3

- Uploaded the Online Retail.csv file to an S3 bucket (ecomm-data-bucket-1) for storage and access.

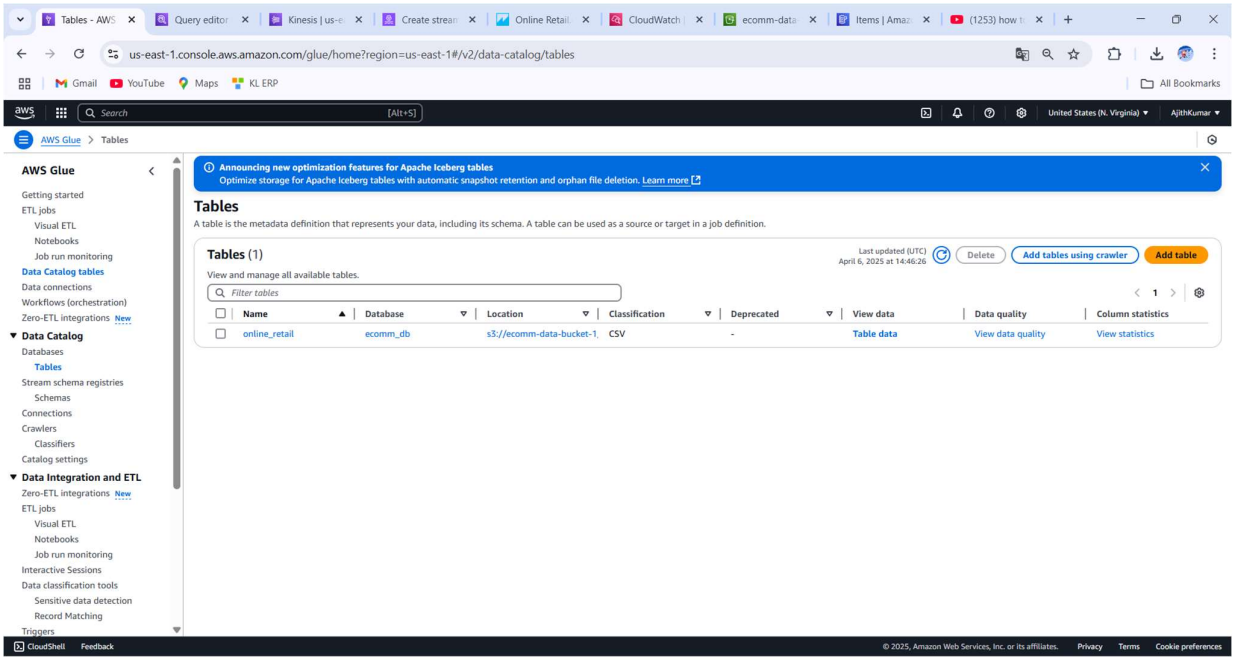


➢ Catalog Data using AWS Glue

- Created a Glue Crawler to scan the S3 data and catalog it into the AWS Glue Data Catalog for querying.



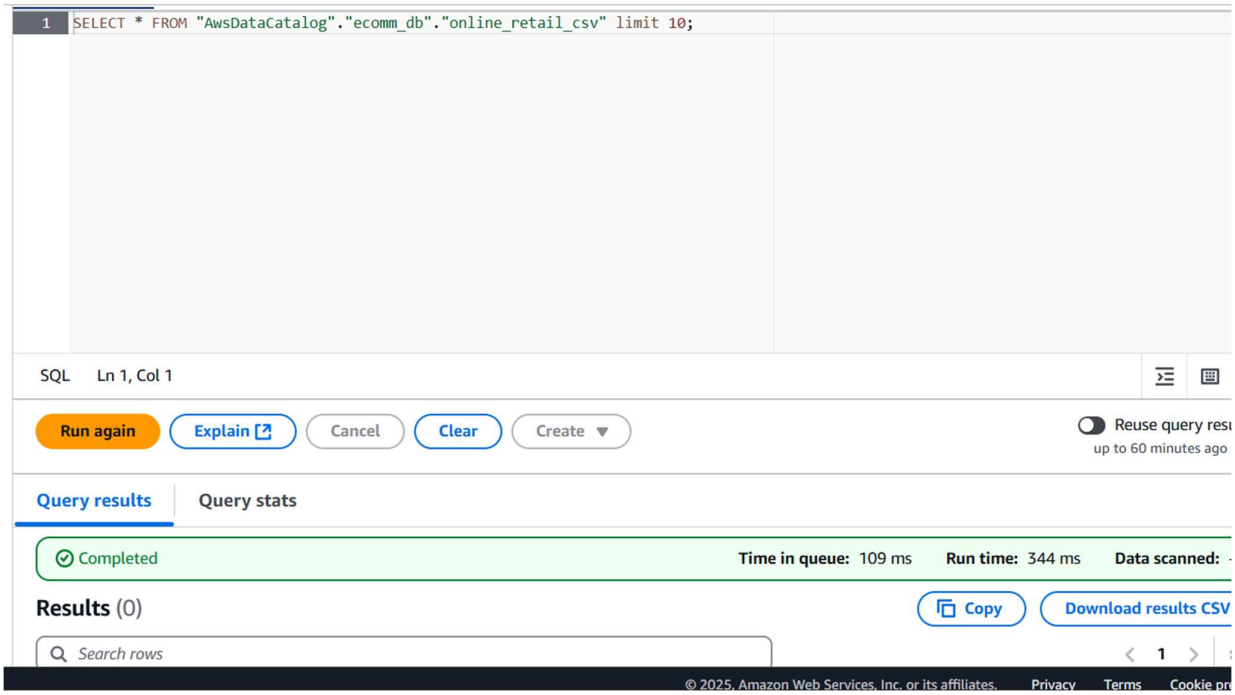
- create database and table while creating the crawler and select the s3 bucket correctly



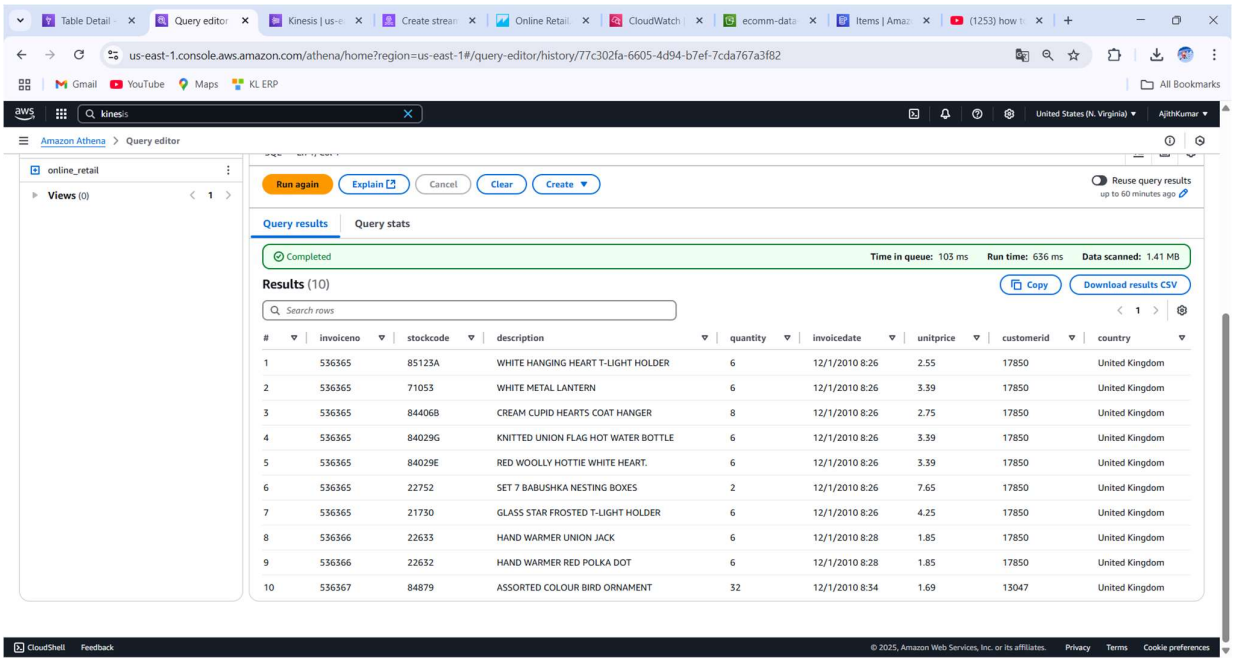
#	Column name	Data type	Partition key	Comment
1	invoiceno	string	-	-
2	stockcode	string	-	-
3	description	string	-	-
4	quantity	bigint	-	-
5	invoicedate	string	-	-
6	unitprice	double	-	-
7	customerid	bigint	-	-
8	country	string	-	-

➢ Query Data with Amazon Athena

- Used Athena to run SQL queries on the data directly from S3 for batch data analysis.
- Open the table and click on the action-> view data



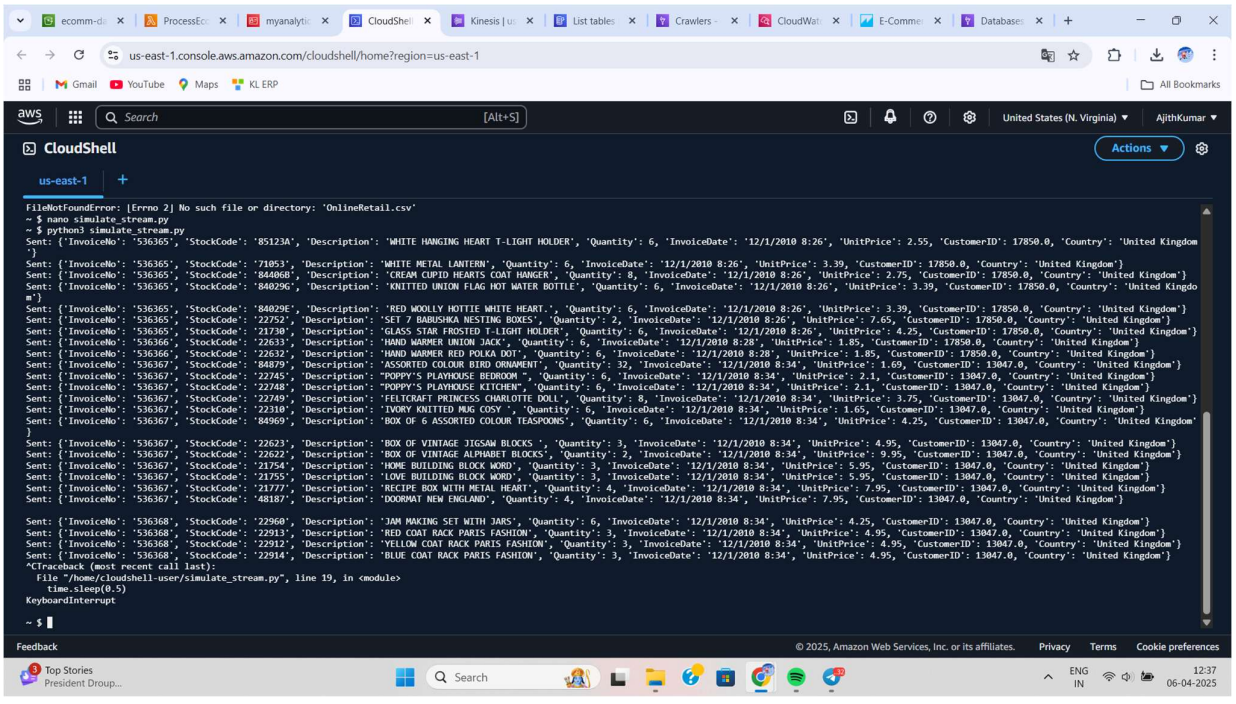
- After clicking on the run you can see the results



➢ Set up the cloudshell

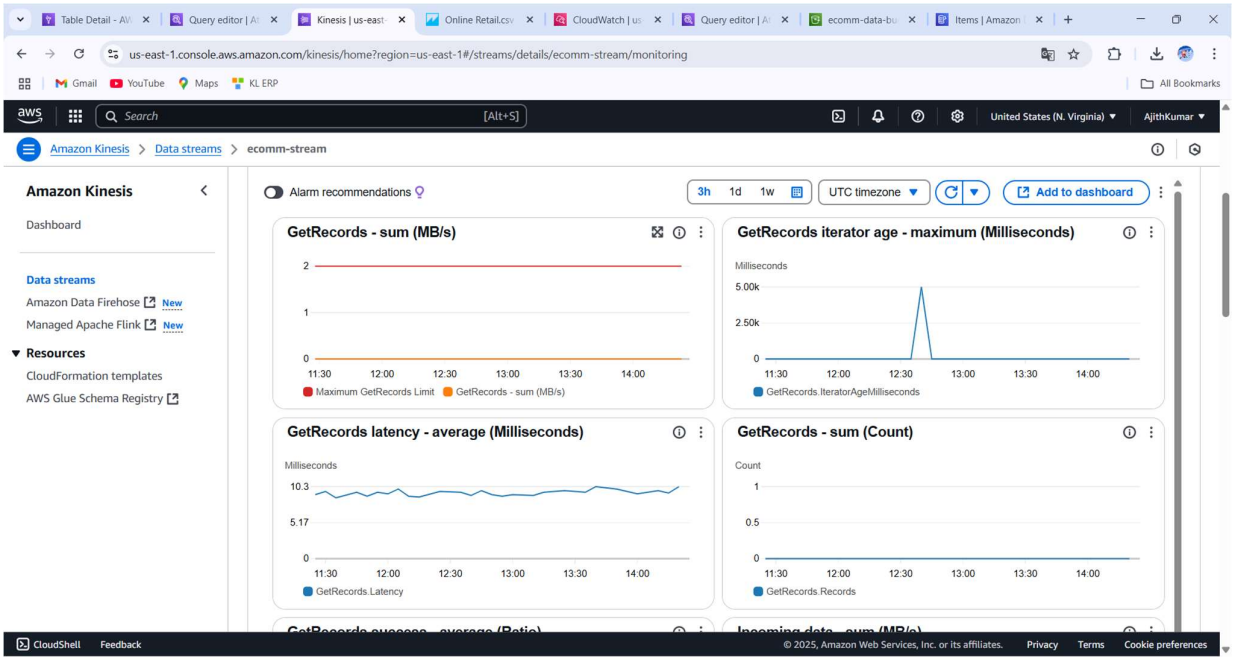
- uploaded the dataset to AWS CloudShell and used a Python script with boto3 to send data to the Kinesis Data Stream.
  The script read the CSV, converted rows to JSON or string, and streamed them into Kinesis in real-time.

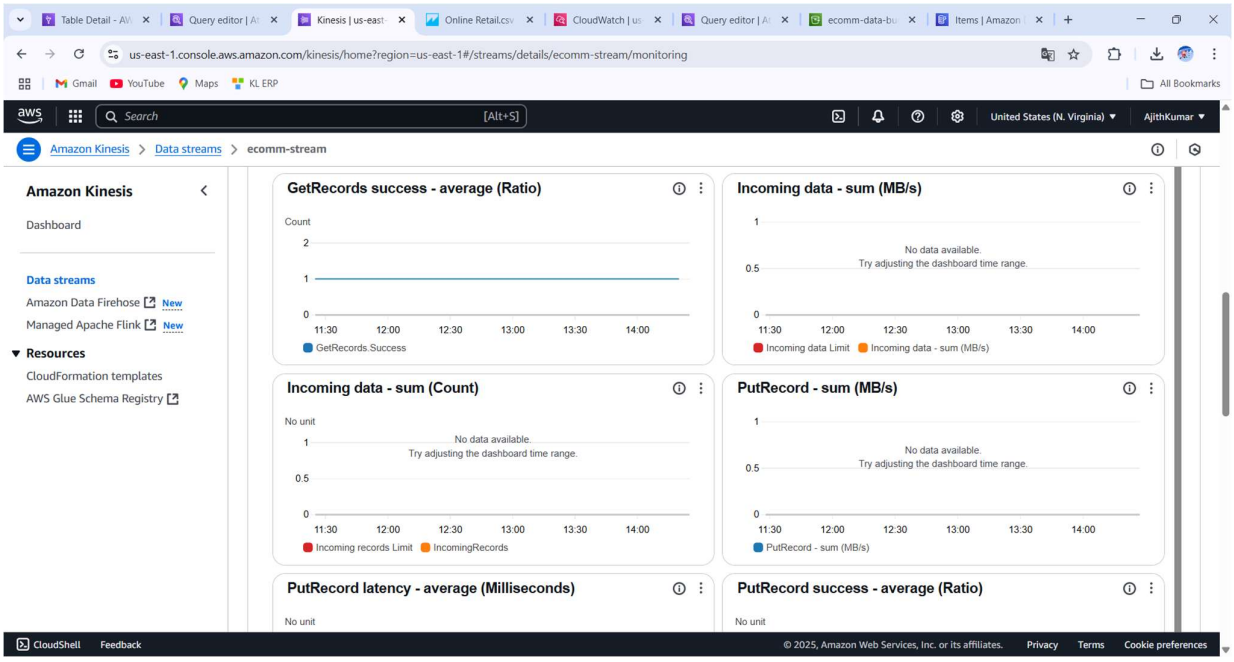
- You can see the uploaded data in the kinesis stream using the Cloud Shell



➢ Set Up Real-Time Ingestion with Kinesis Data Streams

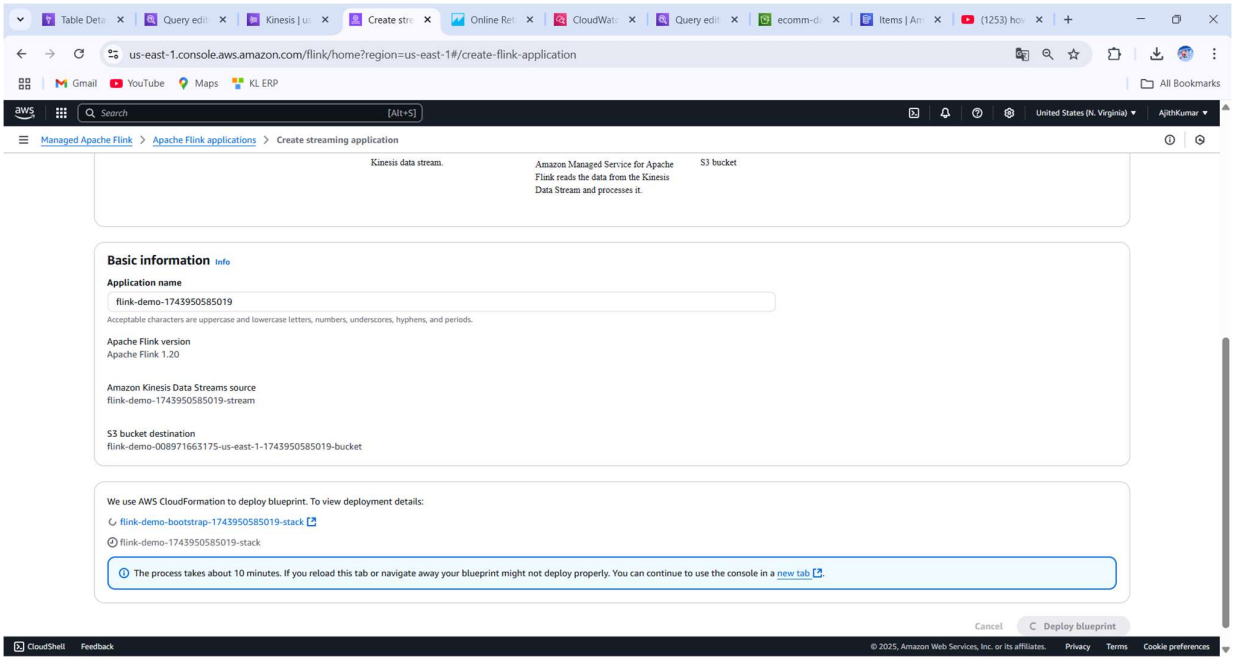
- Streamed eCommerce events (e.g., product views, purchases) into Kinesis for real-time processing.

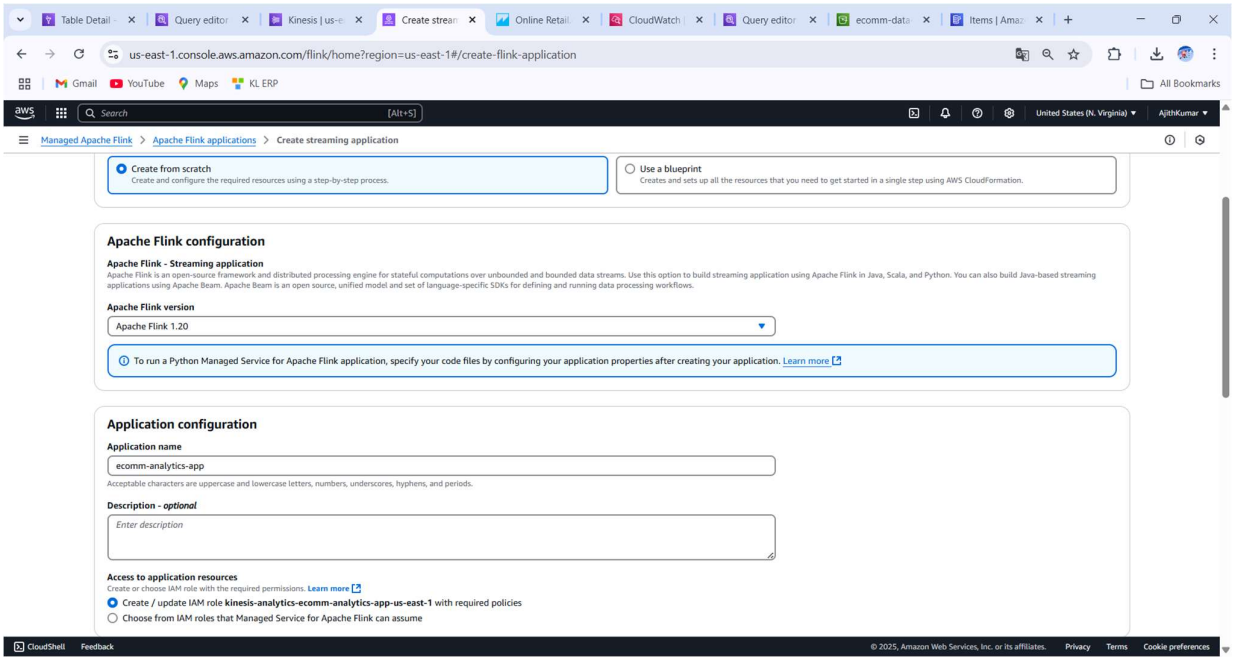


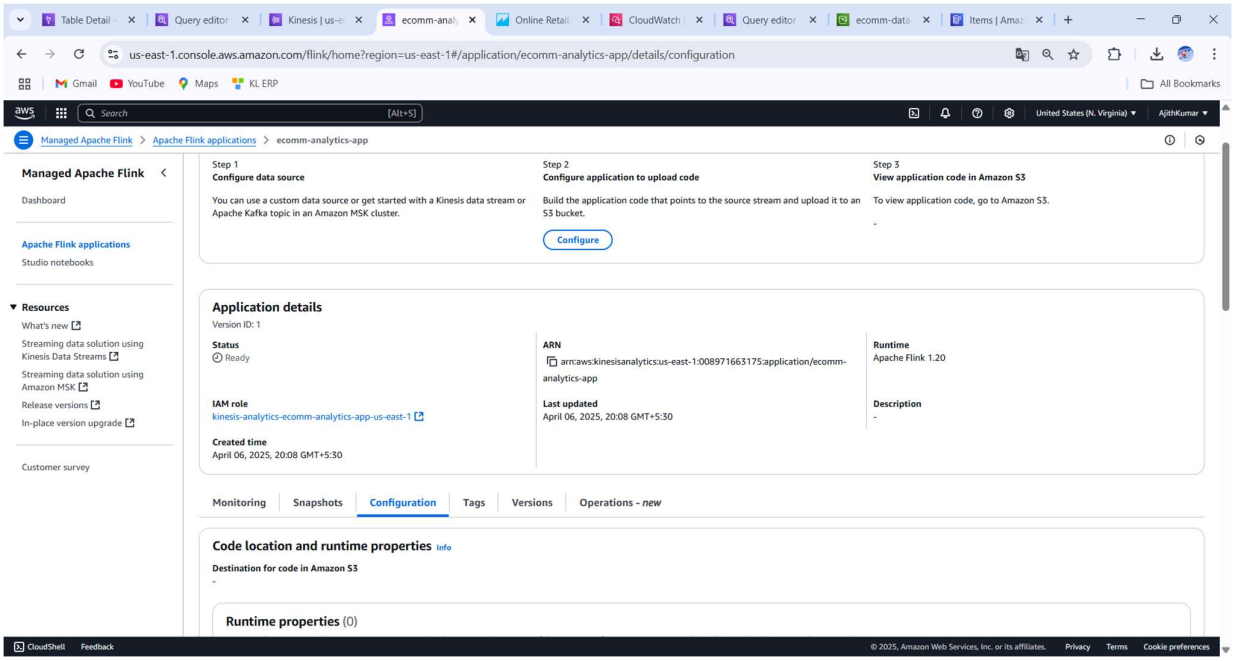


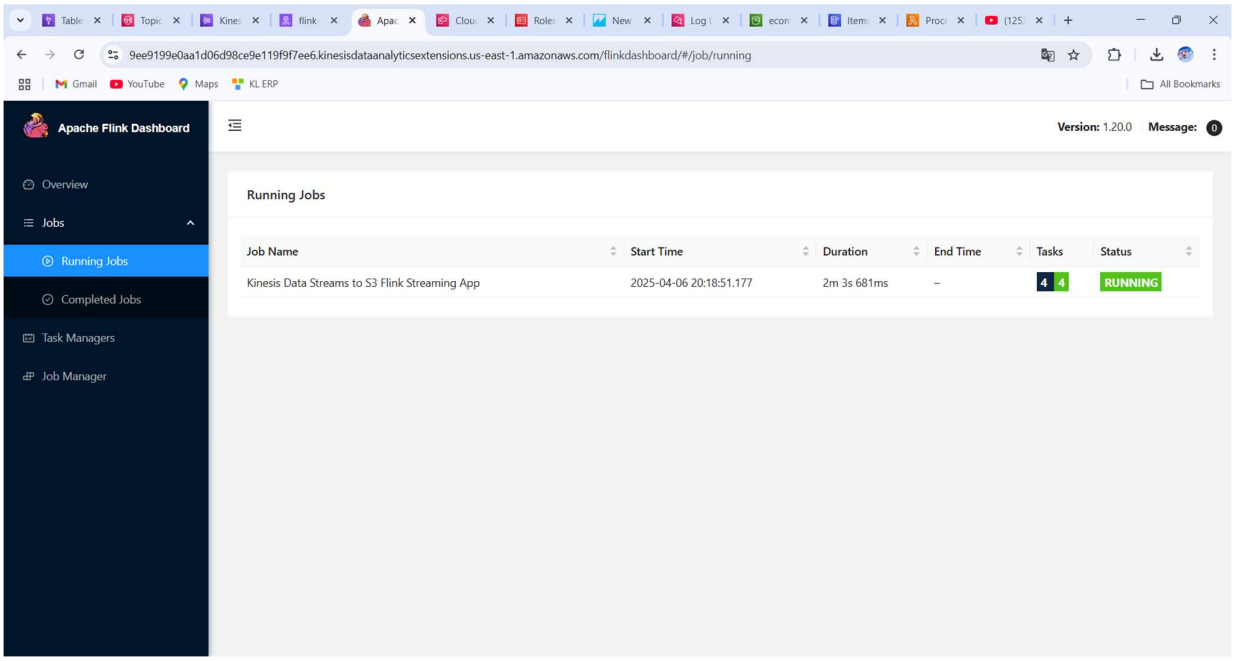
➢ Process Live Data with Kinesis Data Analytics (Flink)

- Built a Flink-based app to process and analyze streaming data for real-time dashboards and insights.



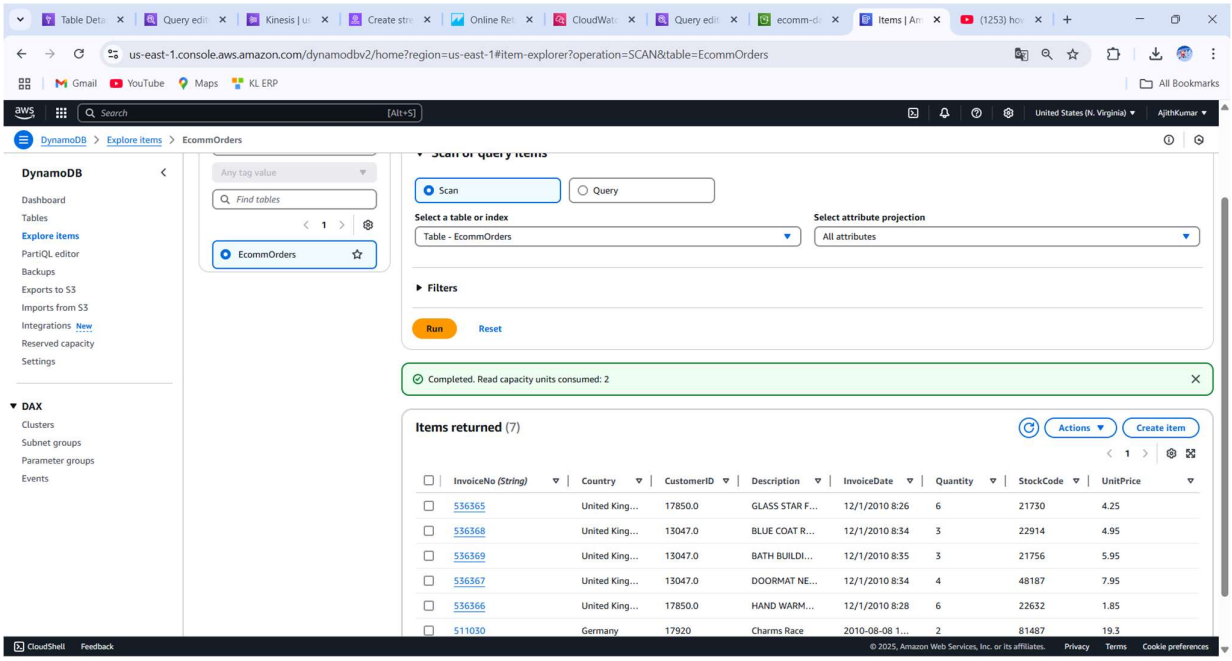



us-east-1.console.aws.amazon.com/flink/home?region=us-east-1#/application/ecomm-analytics-app/details/configuration
Managed Apache Flink > Apache Flink applications > ecomm-analytics-app
Managed Apache Flink
Dashboard
Apache Flink applications
Studio notebooks
Resources
What's new
Streaming data solution using Kinesis Data Streams
Streaming data solution using Amazon MSK
Release versions
In-place version upgrade
Customer survey
Step 1
Configure data source
You can use a custom data source or get started with a Kinesis data stream or Apache Kafka topic in an Amazon MSK cluster.
Step 2
Configure application to upload code
Build the application code that points to the source stream and upload it to an S3 bucket.
Configure
Step 3
View application code in Amazon S3
To view application code, go to Amazon S3.
Application details
Version ID: 1
Status
Ready
ARN
arn:aws:kinesisanalytics:us-east-1:008971663175:application/ecomm-analytics-app
Runtime
Apache Flink 1.20
IAM role
kinesis-analytics-ecomm-analytics-app-us-east-1
Last updated
April 06, 2025, 20:08 GMT+5:30
Description
-
Created time
April 06, 2025, 20:08 GMT+5:30
Monitoring
Snapshots
Configuration
Tags
Versions
Operations - new
Code location and runtime properties Info
Destination for code in Amazon S3
-
Runtime properties (0)
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


9ee9199e0aa1d06d98ce9e119f9f7ee6.kinesisdataanalyticsextensions.us-east-1.amazonaws.com/flinkdashboard/#/job/running
Apache Flink Dashboard
Version: 1.20.0
Message: 0
Overview
Jobs
Running Jobs
Completed Jobs
Task Managers
Job Manager
Running Jobs
Job Name
Start Time
Duration
End Time
Tasks
Status
Kinesis Data Streams to S3 Flink Streaming App
2025-04-06 20:18:51.177
2m 3s 681ms
–
4 4
RUNNING

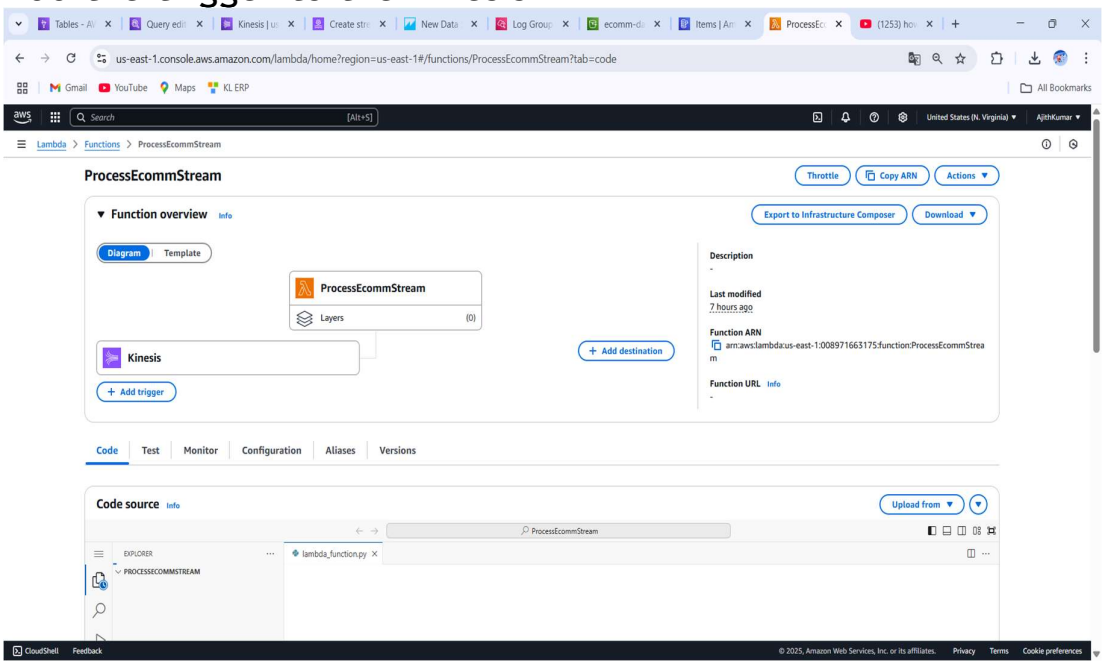
➢ Store Processed Events in DynamoDB

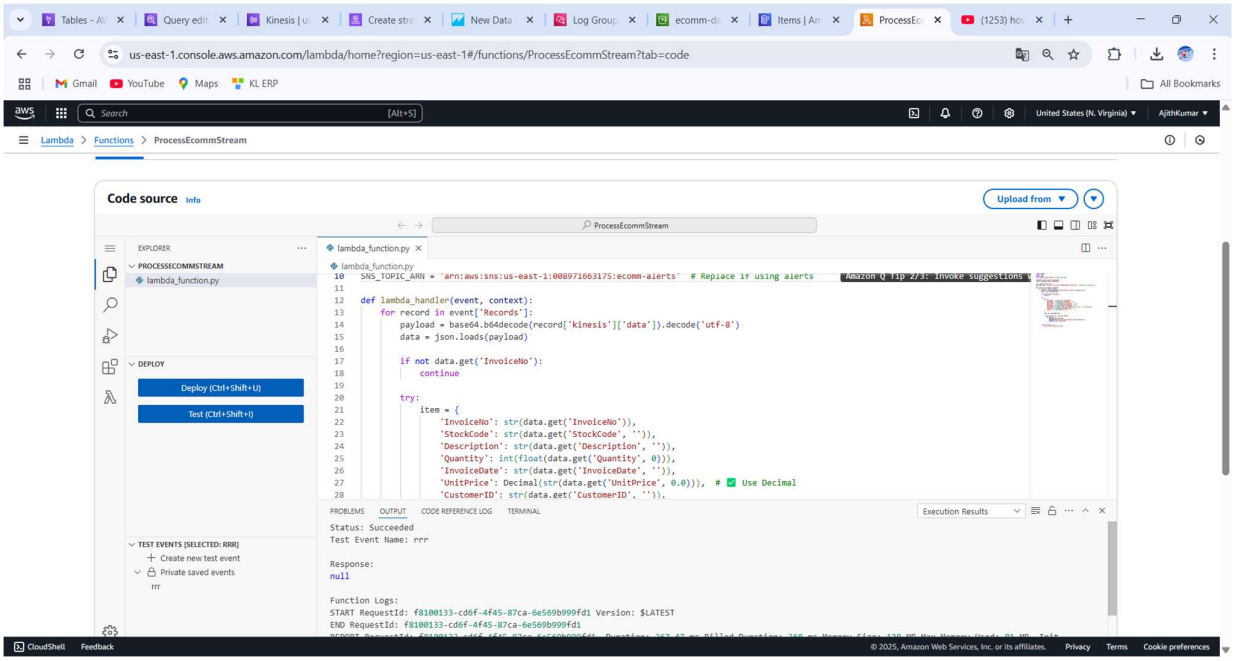
- Streamed important transactional data to DynamoDB for persistent, low-latency access.



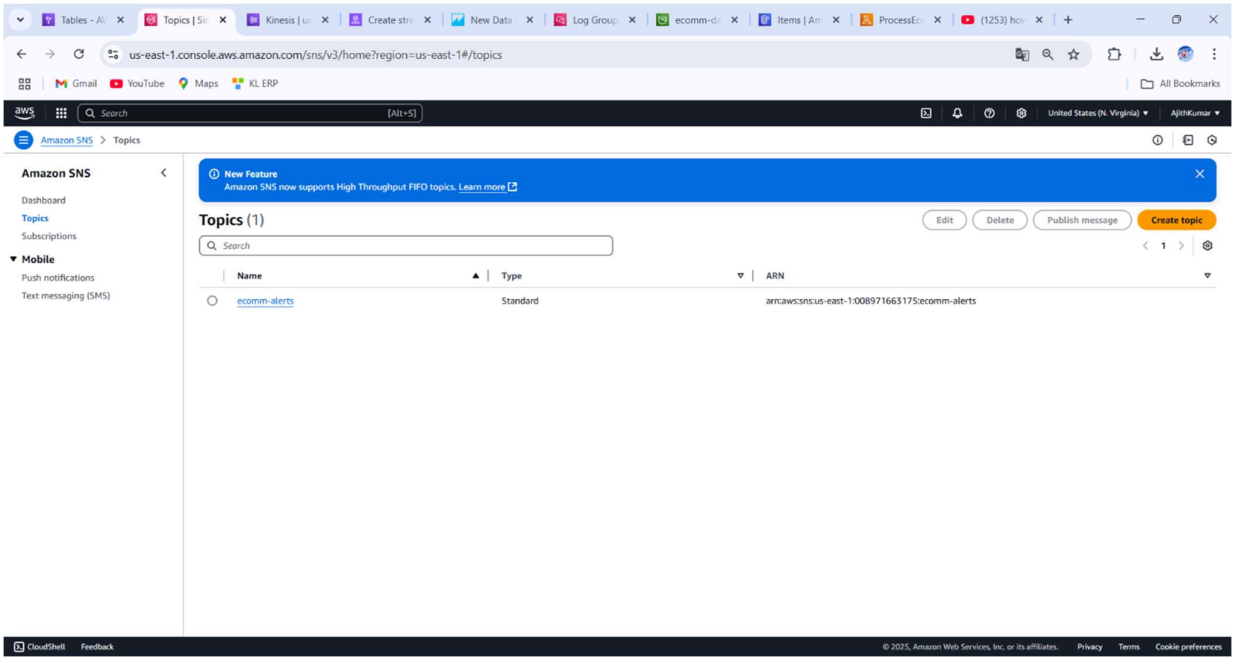
➢ Trigger Notifications using AWS Lambda + SNS

- Configured Lambda functions to process DynamoDB streams and send alerts via Amazon SNS.
- Add the trigger to the kinesis

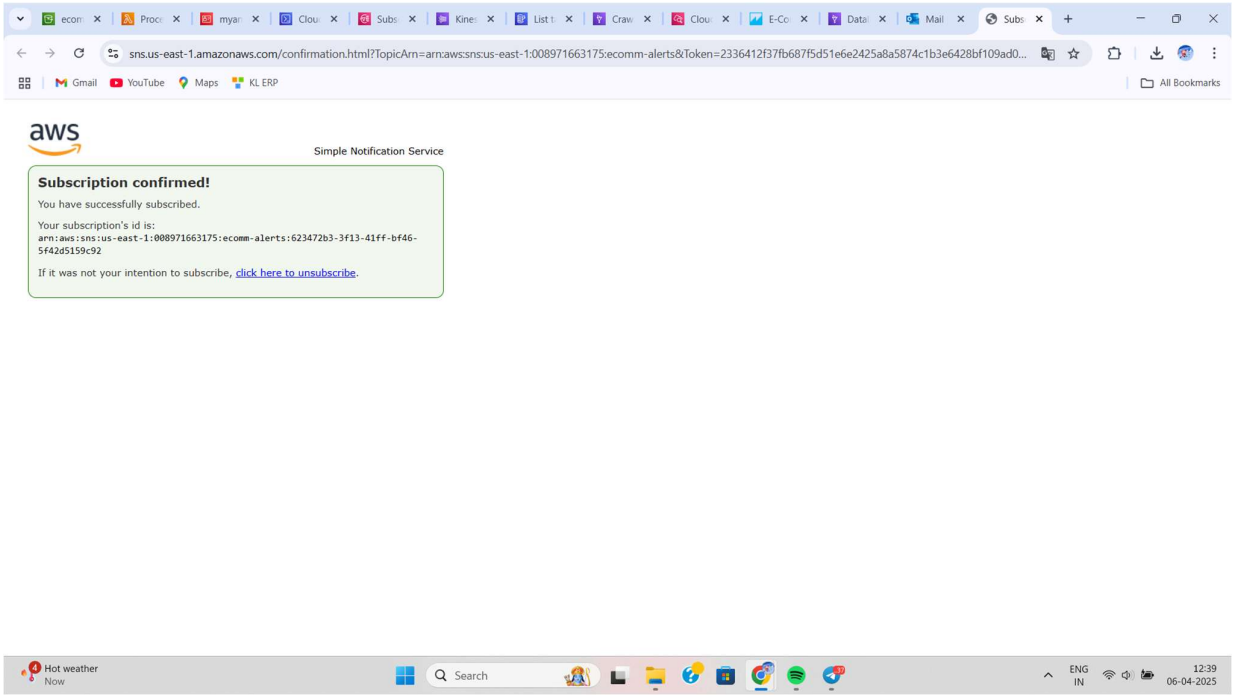




- Create a SNS topic and add the protocol type to the email

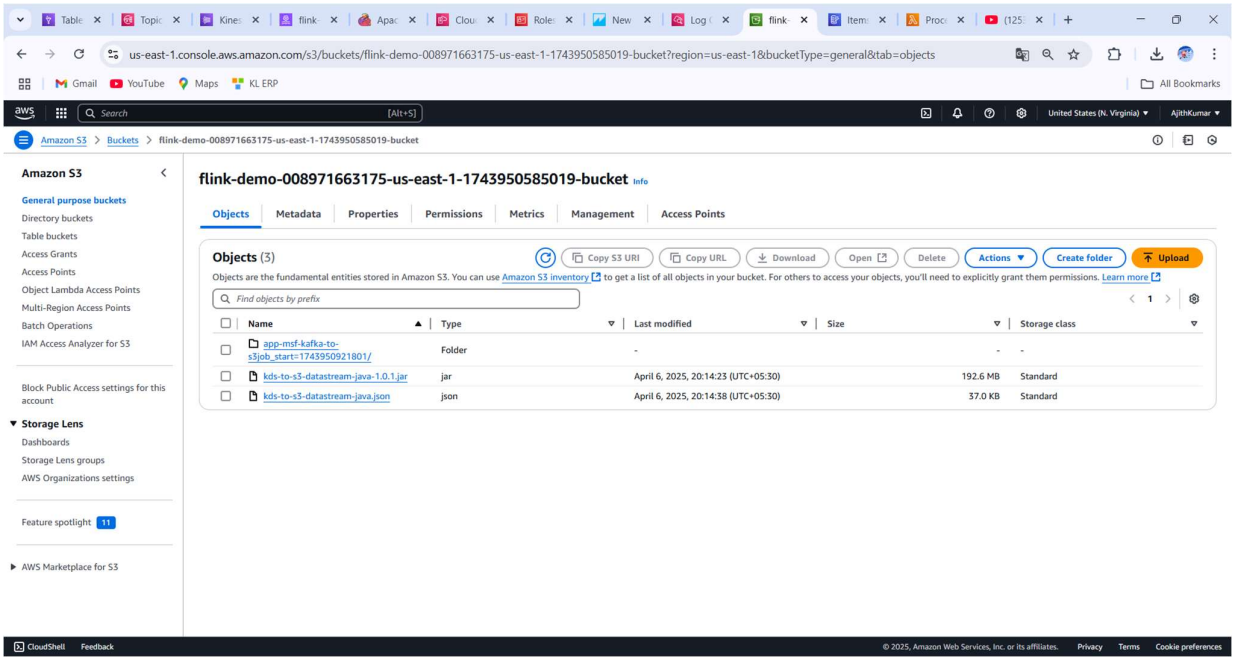


- Confirm the subscription

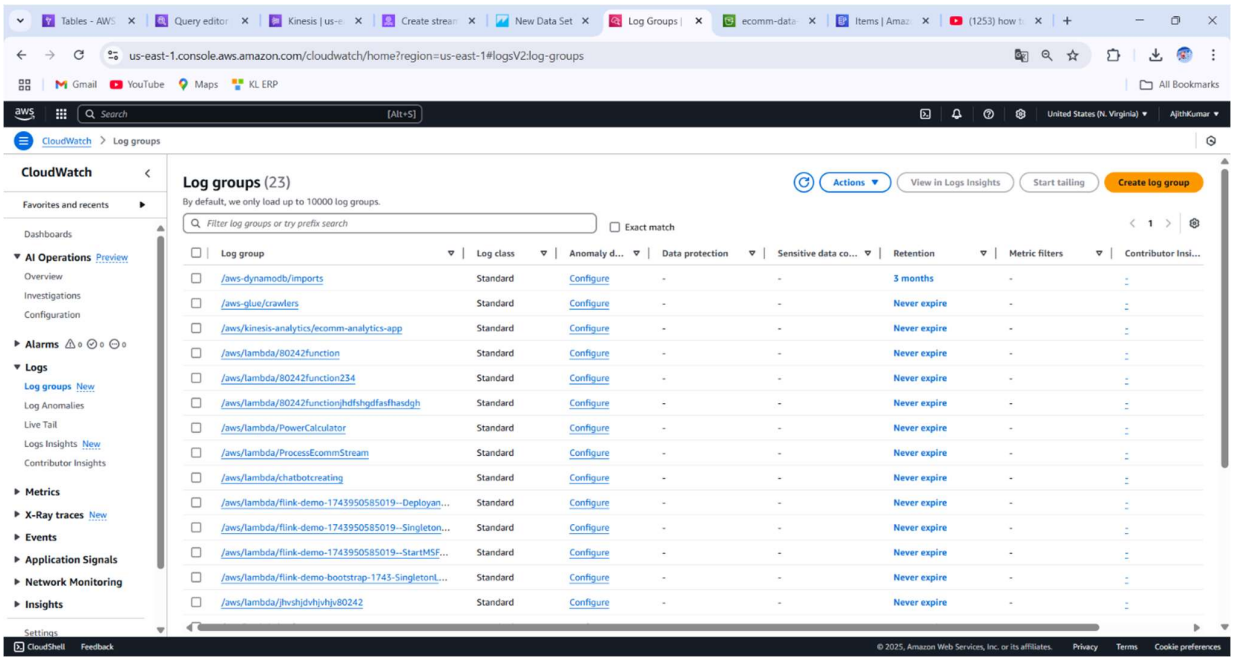


Visualize Results / Monitor Outputs

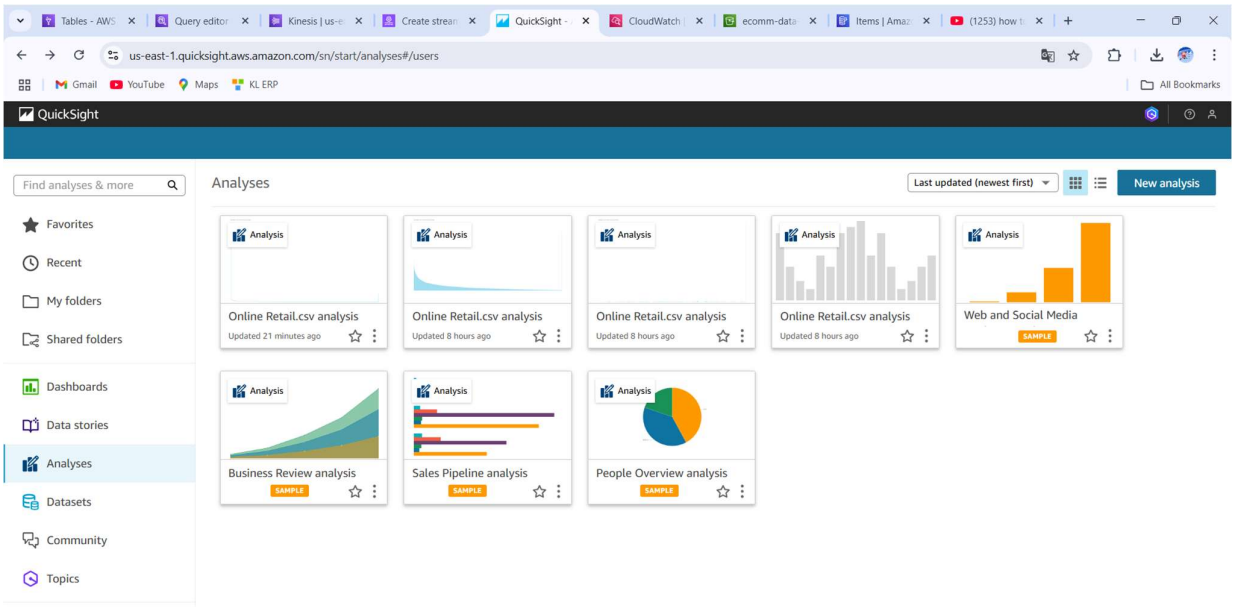
- Analyzed output files from the **Flink output S3 bucket**, checked streaming logs, visualized data using **Amazon QuickSight**
- Flink send the output to another bucket



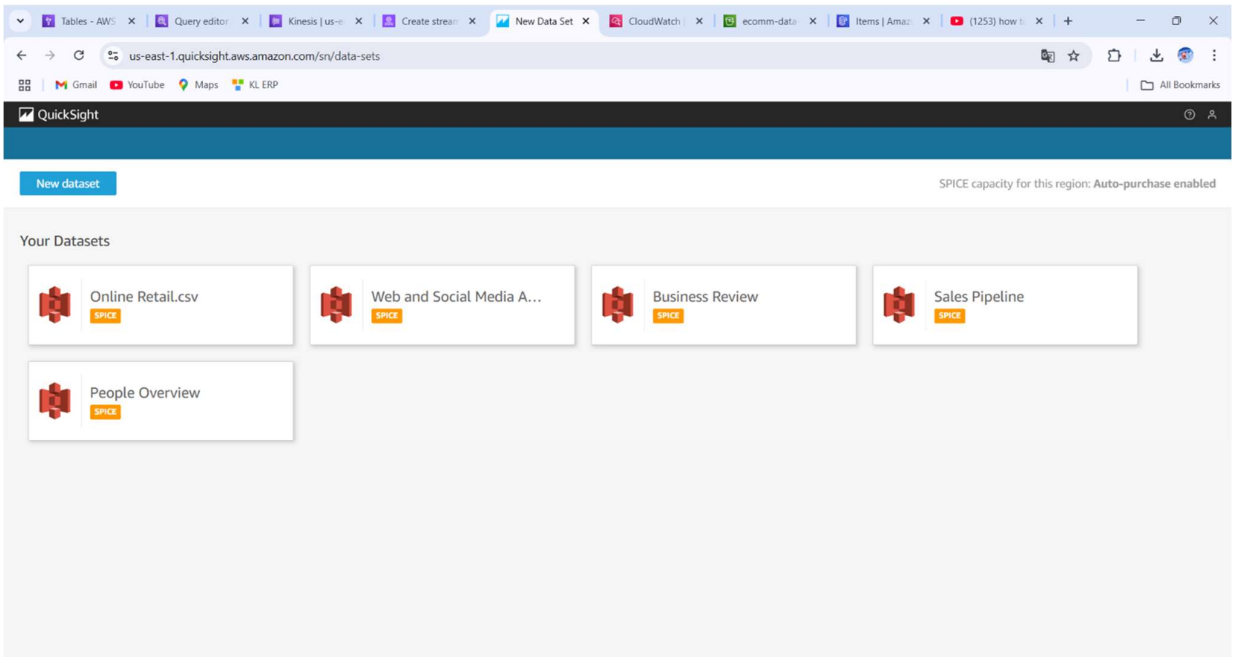
- **Checking CloudWatch logs**



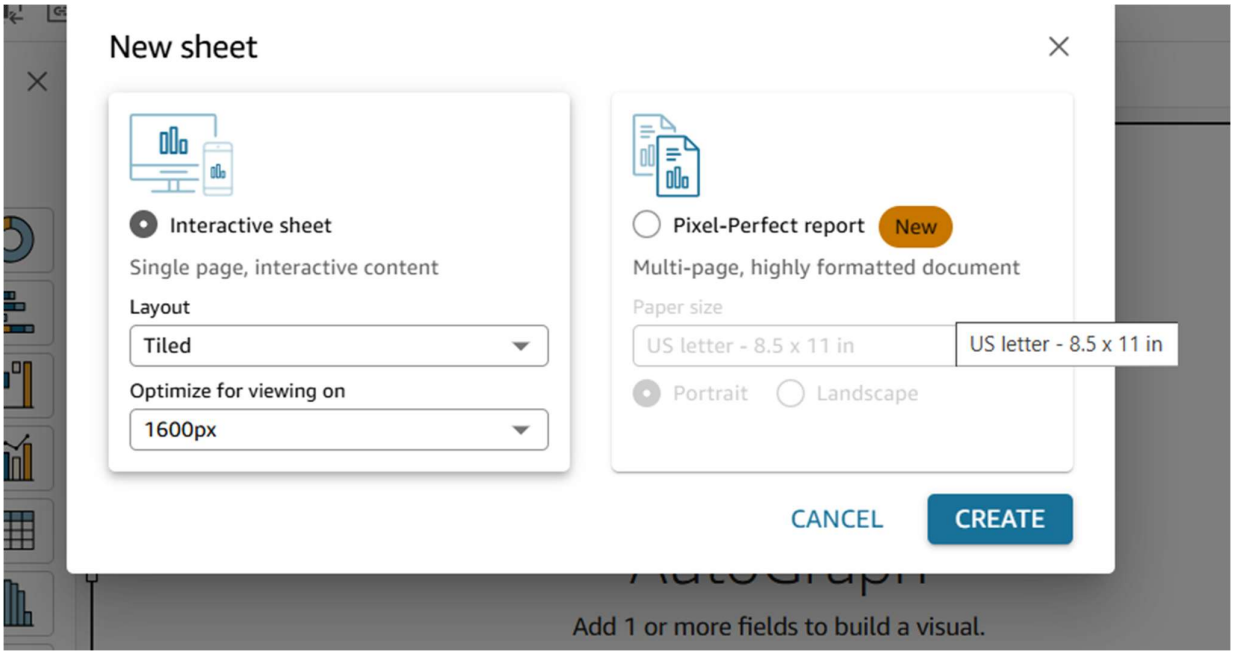
- Accessed Amazon QuickSight to visualize the streaming output data stored in the S3 bucket by connecting to the existing dataset, enabling insightful analysis and real-time dashboards



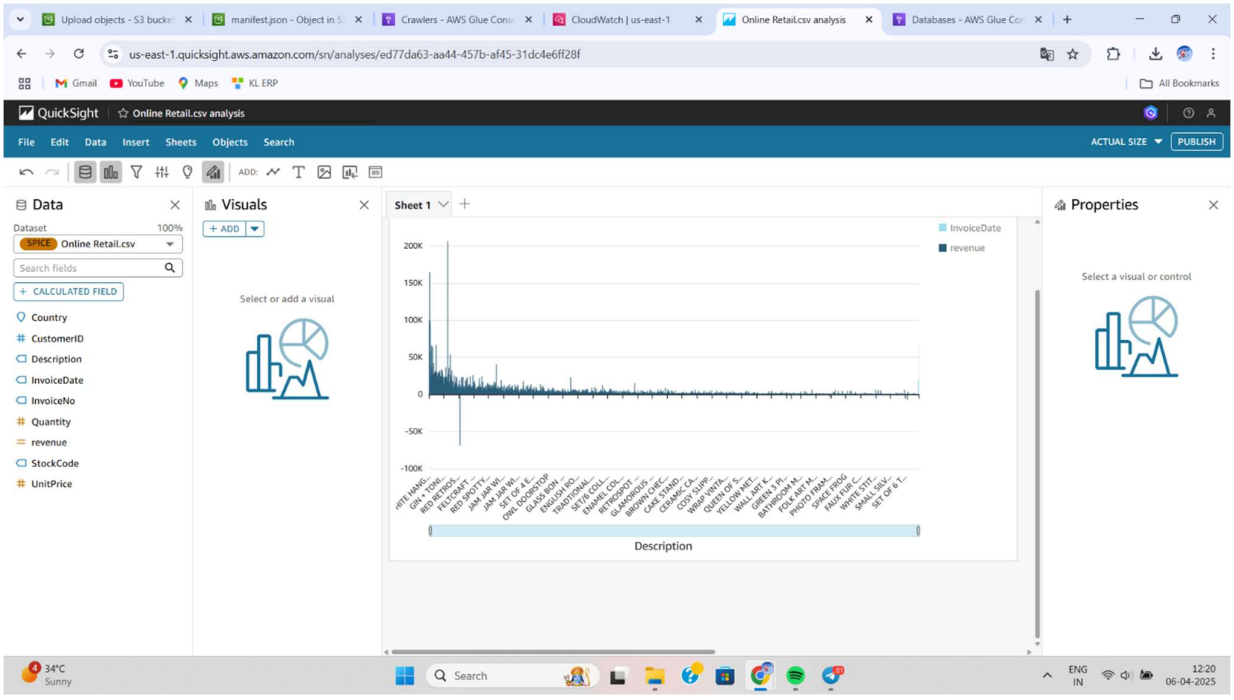
- Click on the new analysis and click on the new dataset, select the S3 URL and upload it

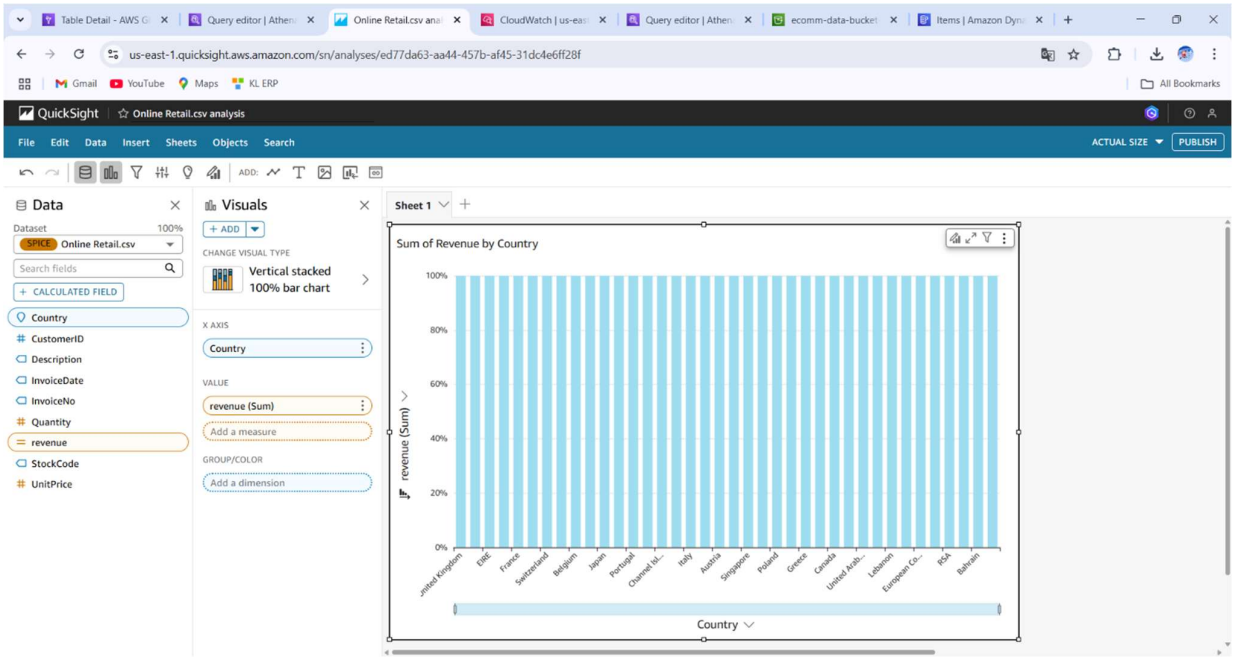


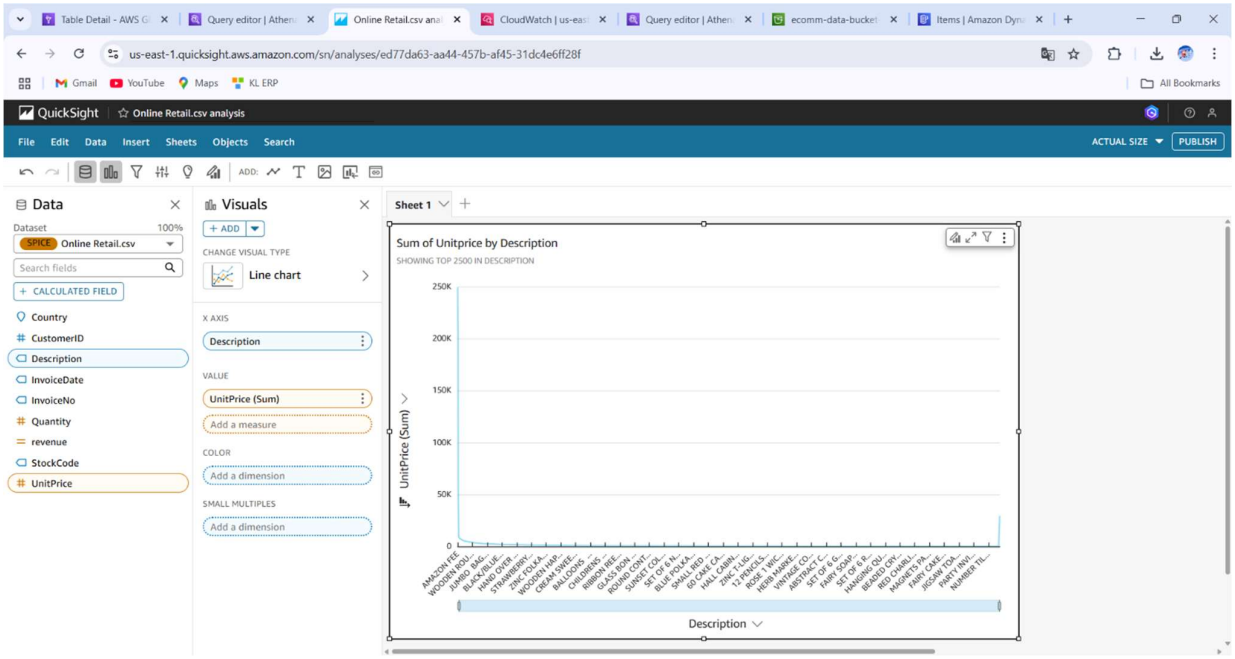
- Create an interactive sheet



- Now u can visualize and change the types like bar chart, vertical bar, line graph etc After that click on the publish, and it will create a dashboard




QuickSight
Online Retail.csv analysis
File Edit Data Insert Sheets Objects Search
ACTUAL SIZE
PUBLISH
Data
Dataset
SPICE Online Retail.csv
Search fields
+ CALCULATED FIELD
Country
CustomerID
Description
InvoiceDate
InvoiceNo
Quantity
revenue
StockCode
UnitPrice
Visuals
+ ADD
CHANGE VISUAL TYPE
Vertical stacked 100% bar chart
X AXIS
Country
VALUE
revenue (Sum)
Add a measure
GROUP/COLOR
Add a dimension
Sheet 1
Sum of Revenue by Country
100%
80%
60%
40%
20%
0%
revenue (Sum)
United Kingdom
EIRE
France
Switzerland
Belgium
Japan
Portugal
Channel Isl...
Italy
Austria
Singapore
Poland
Greece
Canada
United Arab...
Lebanon
European Co...
RSA
Bahrain
Country


QuickSight
Online Retail.csv analysis
File Edit Data Insert Sheets Objects Search
ACTUAL SIZE
PUBLISH
Data
Dataset
SPICE Online Retail.csv
Search fields
+ CALCULATED FIELD
Country
CustomerID
Description
InvoiceDate
InvoiceNo
Quantity
revenue
StockCode
UnitPrice
Visuals
+ ADD
CHANGE VISUAL TYPE
Line chart
X AXIS
Description
VALUE
UnitPrice (Sum)
Add a measure
COLOR
Add a dimension
SMALL MULTIPLES
Add a dimension
Sheet 1
Sum of Unitprice by Description
SHOWING TOP 2500 IN DESCRIPTION
250K
200K
150K
100K
50K
0
UnitPrice (Sum)
Description

Conclusion

In this project, we successfully designed and implemented a scalable, secure, and highly available analytical platform tailored for an e-commerce environment using AWS services. The primary goal was to simulate and analyze user activity data such as product views, cart events, and purchase histories, in both batch and real-time scenarios, enabling powerful insights and timely decision-making.

We started by ingesting historical eCommerce data into Amazon S3 and used AWS Glue to catalog the dataset and make it queryable via Amazon Athena. Glue Crawlers automated schema detection, while Athena provided serverless SQL querying capabilities for batch analytics.

For real-time data processing, Amazon Kinesis Data Streams was integrated to simulate live clickstream data. The stream was further analyzed in real-time using Kinesis Data Analytics, powered by Apache Flink, enabling near-instant insights and alert generation.

Additionally, AWS Lambda and DynamoDB Streams were leveraged to build a reactive notification system. As new orders were inserted into DynamoDB, Lambda functions triggered SNS (Simple Notification Service) to alert stakeholders or services about customer actions. This event-driven architecture demonstrated how microservices can react to business events without polling or delays.

To visualize and gain business insights from both batch and streaming outputs, **Amazon QuickSight** was used to build interactive dashboards by connecting directly to Athena queries and the output data stored in Amazon S3. This provided rich, real-time analytics and user-friendly visual reports for stakeholders.

Throughout the pipeline, security and best practices were maintained using IAM roles, S3 bucket policies, and fine-

grained access controls, ensuring data privacy and safe access to resources.

Ultimately, this project demonstrates how cloud-native services on AWS can be combined to build a powerful big data pipeline for eCommerce analytics, covering everything from data ingestion and transformation to real-time processing, visualization, querying, and notification. This end-to-end platform not only enables deep insights into customer behavior but also lays the foundation for predictive analytics, personalized marketing, inventory optimization, and better customer service — all critical for the success of modern digital businesses.