Serverless Real-time Analytics Dashboard with AWS Kinesis Analytics

A Course Project Report Submitted in partial fulfillment of the course requirements for the award of grades in the subject of

CLOUD BASED AIML SPECIALITY
(22SDCS07A)

by

Nalluri Tulasi

(2210030388)

Under the esteemed guidance of

Ms. P. Sree Lakshmi
Assistant Professor,
Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K L Deemed to be UNIVERSITY

Aziznagar, Moinabad, Hyderabad,
Telangana, Pincode: 500075

April 2025

K L Deemed to be UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

This is Certified that the project entitled **"Serverless Real Time Analytics Dashboard with AWS Kinesis Analytics"** which is a experimental &/ Simulation work carried out by Nalluri Tulasi (2210030388), in partial fulfillment of the course requirements for the award of grades in the subject of **CLOUD BASED AIML SPECIALITY**, during the year **2024-2025**. The project has been approved as it satisfies the academic requirements.

Ms.P.Sree Lakshmi	**Dr. Arpita Gupta**
Course Coordinator	**Head of the Department**

Ms. P. Sree Lakshmi

Course Instructor

CONTENTS

1. INTRODUCTION

In today's fast-paced digital world, businesses need real-time insights to stay competitive. Traditional analytics systems often struggle with scalability, latency, and operational costs, making serverless solutions a more efficient alternative. The Serverless Real-time Analytics Dashboard with AWS Kinesis Analytics leverages AWS Kinesis to process streaming data efficiently, offering a scalable, cost-effective, and high-performance solution [9].

AWS Kinesis Analytics enables businesses to analyze real-time data instantly without managing complex infrastructure. This is particularly valuable in industries like e-commerce, healthcare, gaming, and fraud detection, where immediate insights improve decision-making, security, and customer engagement [4][5][7]. In e-commerce, real-time data helps create AI-driven recommendation systems, improving user experience and conversion rates [1][2]. Fraud detection systems use AWS Kinesis to monitor transactions in real-time, identifying anomalies and preventing financial losses [7][9]. The gaming industry benefits from real-time analytics by detecting unusual player behavior and optimizing game mechanics dynamically [6].

By integrating services like Kinesis Firehose, Amazon S3, AWS Lambda, and QuickSight, this project aims to develop a serverless analytics dashboard that provides real-time data visualization and actionable insights. With high availability, fault tolerance, and scalability, this architecture ensures seamless integration into modern business environments [9].

This project demonstrates how AWS-native services can power real-time analytics, helping businesses process and interpret large volumes of streaming data efficiently. By leveraging serverless technologies, organizations can reduce infrastructure overhead while gaining deep insights into their operations in real time.

2. AWS SERVICES USED AS PART OF THE PROJECT

Amazon Kinesis – Real-time Data Streaming & Analytics

Amazon Kinesis is the backbone of the analytics pipeline, handling real-time data ingestion and transformation.

- Kinesis Data Streams (KDS) → Captures real-time data from various sources (e.g., logs, transactions, IoT devices). KDS provides high-throughput and low-latency streaming.
- Kinesis Data Firehose → A fully managed service that automatically loads streaming data into Amazon S3 for storage and further analysis.
- Kinesis Data Analytics (KDA) → Performs real-time transformations and analysis on streaming data using SQL-based processing before forwarding it to storage or further processing [9].

AWS Lambda – Serverless Compute for Data Processing

AWS Lambda processes streaming data in real time without managing servers. It can:

- Filter, aggregate, and transform data from Kinesis streams before storage.
- Trigger automated actions based on real-time analytics results.
- Seamlessly integrate with Kinesis, Firehose, and S3 for event-driven processing [9].

Amazon S3 – Scalable Storage for Streaming Data

Amazon S3 acts as the primary storage layer for raw and processed data. It provides:

- Highly durable and scalable storage for real-time event logs and processed analytics data.
- Seamless integration with other AWS services for further processing.
- Low-cost archival options for long-term data retention [9].

AWS IAM (Identity & Access Management) – Security & Access Control

AWS IAM ensures secure access and permissions across all AWS services in the project:

- Role-based access control (RBAC) restricts data access to authorized users and services.
- Encryption and authentication policies protect streaming and stored data.
- Integration with AWS CloudTrail allows tracking and auditing of access logs [9].

Amazon CloudWatch – Real-time Monitoring & Logging

Amazon CloudWatch collects logs, metrics, and alerts to monitor system performance and detect anomalies:

- Tracks Kinesis, Lambda, and Firehose performance to ensure seamless data flow.
- Detects failures, anomalies, and latency issues in real-time analytics.
- Triggers alerts for operational failures or unusual data patterns [9].

Amazon EC2 – Compute Environment for Code Execution

Amazon EC2 provides a flexible compute environment to write, deploy, and execute code for real-time analytics processing.

- Code Execution: EC2 instances serve as the primary development and execution environment for analytics scripts interacting with streaming data.
- Data Integration: EC2 integrates with Kinesis streams, Lambda, and S3 to process, analyze, and visualize real-time data efficiently [9].
- Scalability: EC2 allows running additional workloads, such as machine learning models for anomaly detection or predictive analytics [7].

3. STEPS INVOLVED IN SOLVING PROJECT PROBLEM STATEMENT

1. **Understanding the Problem Statement**
   - Identify the data source – a temperature sensor continuously generating real-time temperature readings.
   - Define real-time analytics requirements, such as monitoring temperature variations, triggering alerts for anomalies, and storing data for further analysis.
   - Determine the scalability and latency constraints to ensure efficient real-time processing.

2. **Designing the AWS Architecture**
   - Use Amazon Kinesis Data Streams for ingesting continuous real-time temperature data from IoT sensors [9].
   - Implement Amazon Kinesis Data Firehose to load the streaming data into Amazon S3 for storage and further analysis [10].
   - Deploy AWS Lambda to process incoming temperature data, check for anomalies (e.g., extremely high or low temperatures), and trigger alerts [9].
   - Utilize Amazon EC2 instances to execute and test custom analytics scripts that process the temperature data in real time [7].
   - Use Amazon CloudWatch to monitor system performance and log sensor data for insights [9].

3. **Data Ingestion & Processing**
   - Simulate real-time temperature data from a temperature sensor.
   - Stream the data into Kinesis Data Streams, ensuring low-latency ingestion.
   - Use AWS Lambda to process and filter temperature readings, identifying extreme values.
   - Store the processed temperature data in Amazon S3 for further analysis.

4. **Security & Access Control**
   - Set up AWS IAM roles and policies to restrict access to AWS services (Kinesis, Lambda, EC2, and S3) [9].
   - Implement encryption and secure authentication mechanisms to protect sensor data.

5. **Monitoring & Optimization**
   - Use Amazon CloudWatch to monitor logs, track anomalies, and generate system alerts [9].
   - Optimize AWS resource utilization (e.g., configuring EC2 auto-scaling, adjusting Kinesis shard capacity) to enhance performance.

6. **Deployment & Testing**
   - Deploy the temperature monitoring system and verify real-time data flow.
   - Test anomaly detection and alert triggers by simulating temperature variations.
   - Optimize performance based on CloudWatch metrics.

7. **Real-time Visualization & Alerts**
   - Monitor real-time temperature data via the AWS Dashboard.
   - Trigger alerts when temperature readings exceed predefined thresholds.

4. STEPWISE SCREENSHOTS WITH BRIEF DESCRIPTION

Step 1: Create a IAM user to securely manage access to AWS services and resources.

- click on create user → enter the name → Attach policies required
  - AmazonS3FullAccess
  - AmazonAthenaFullAccess
  - AmazonKinesisFullAccess
- click on create user
- go to → security crendentials → create Access Key

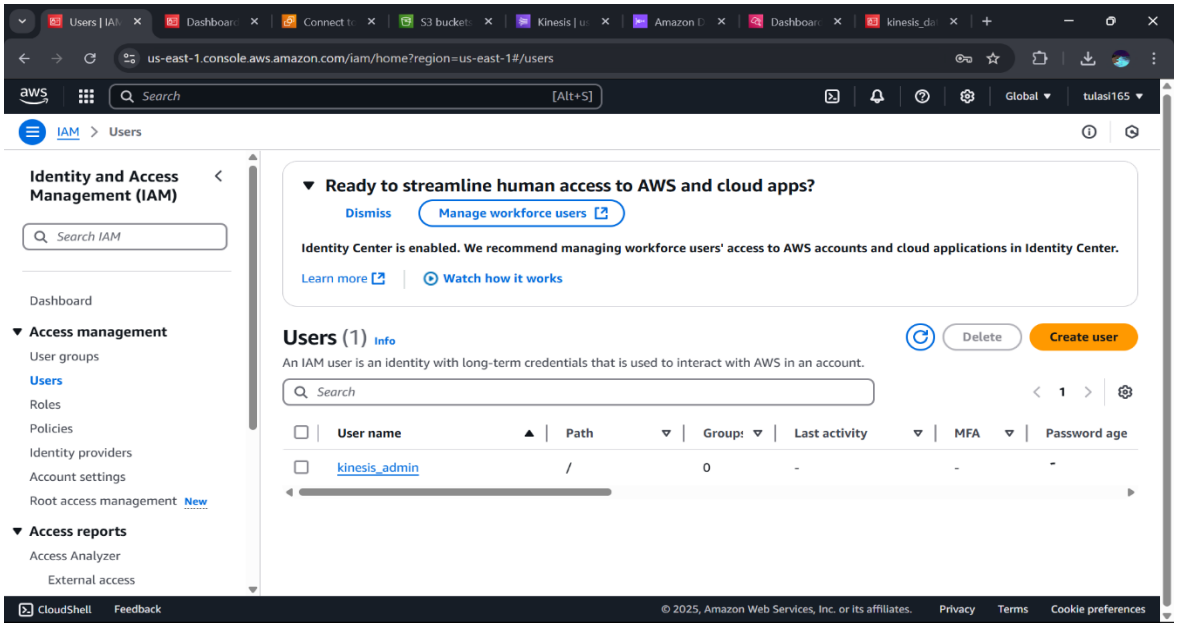


Fig 4.1.1 Creating the IAM user and also create the access key

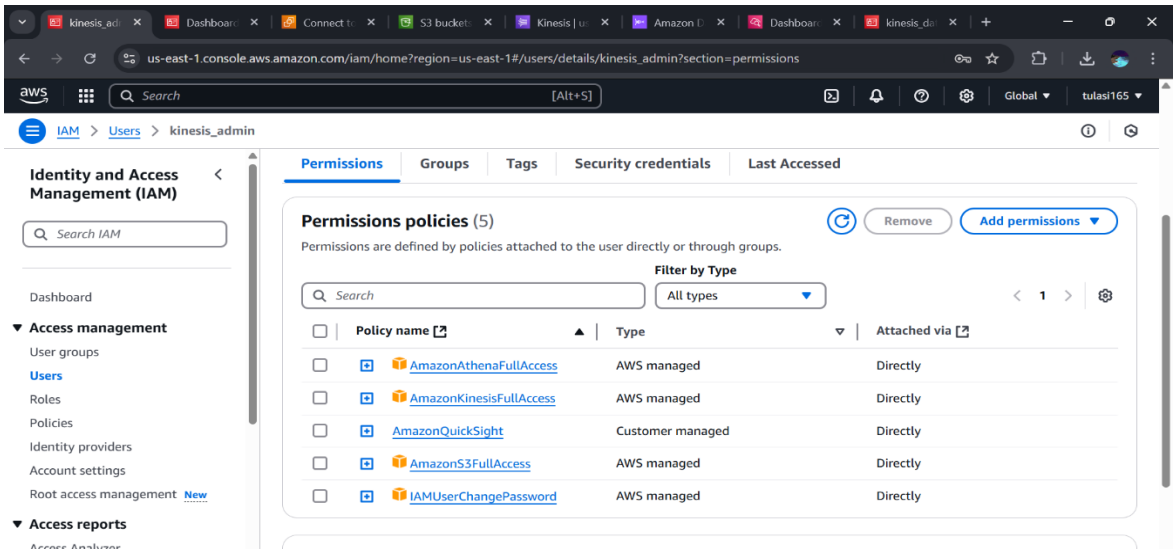


Fig 4.1.2 Adding the permissions/policies required for the project.

Step 2 : Create a EC2 Instance to Simulate Real-Time Streaming Data

- Click on launch instance → name the instance → select Amazon Linux → Instance type is t2.micro → create a key pair →click launch
- Instance is successfully created.
- After Creating the EC2 Instance click on connect and copy the SHH client

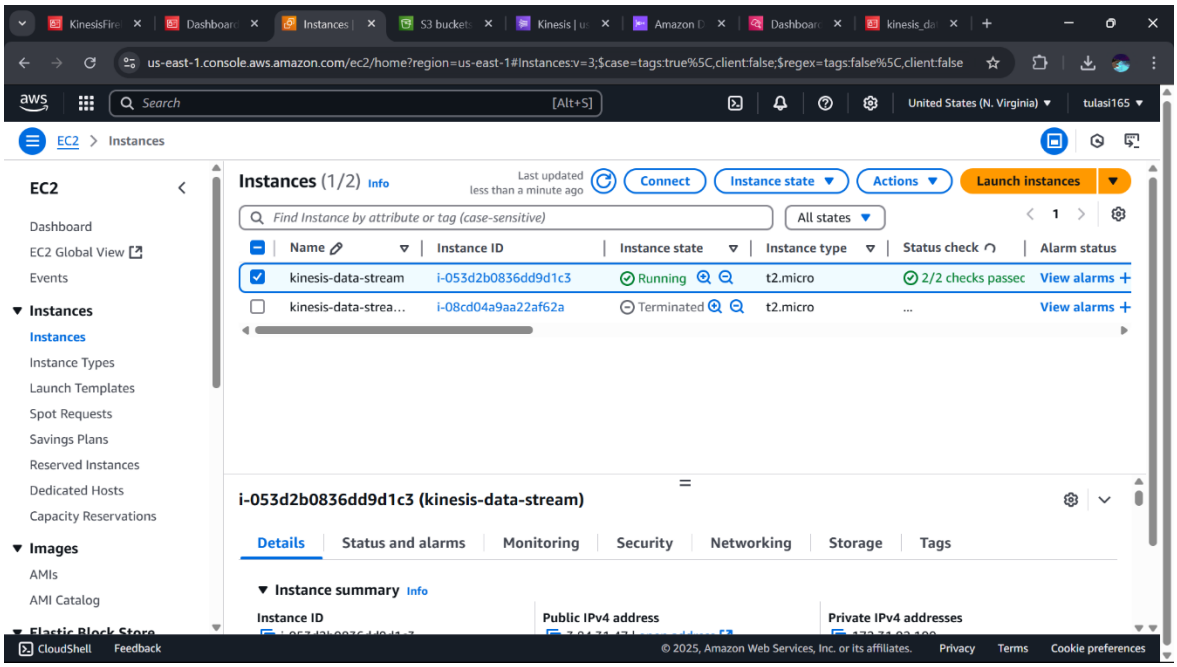


Fig 4.2.1 Successfully created EC2 Instance

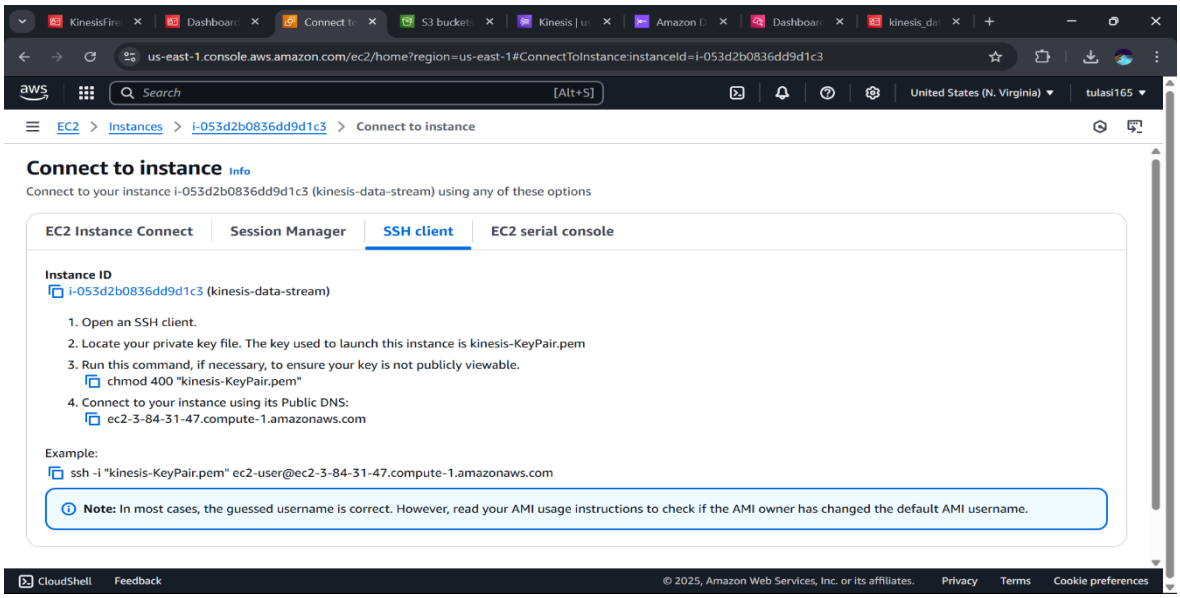


Fig 4.2.2 click on connect and copy the SHH client

Step 3 : Connecting to AWS CLI

- Open CMD → check for AWS CLI version → navigate to location where the keypair in downloaded → paste the SSH client → configure → enter access key , security key, region, output format.
- Install all the required packages for your project or code.

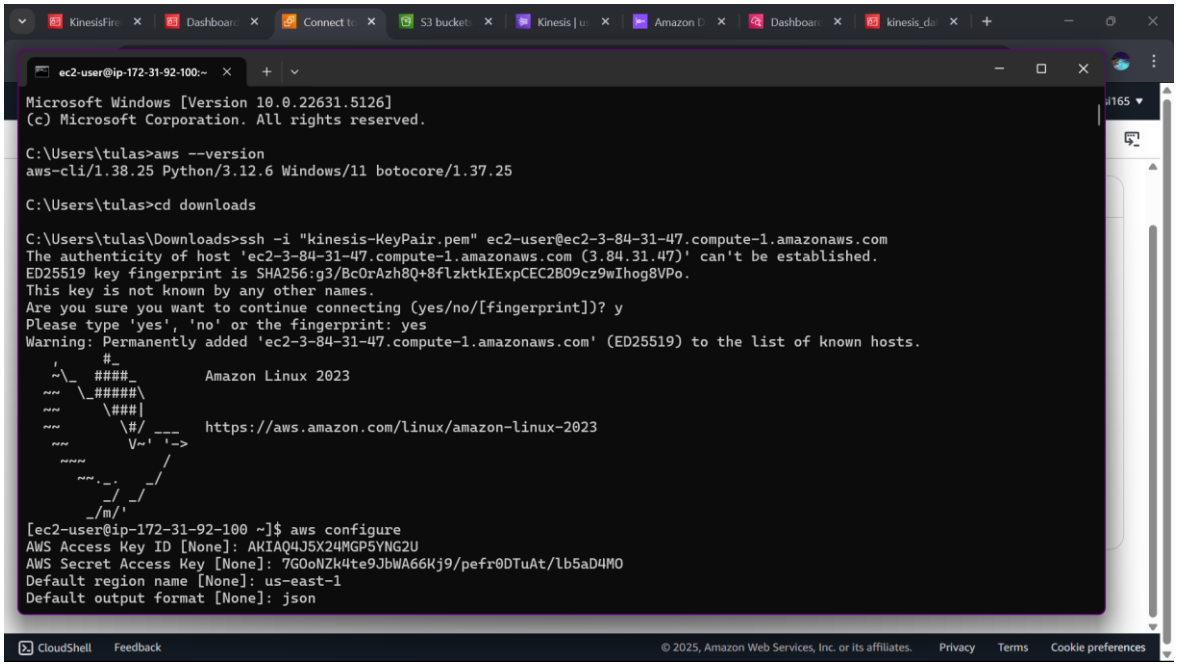


Fig 4.3.1 Configure the Instance with the terminal with the access key, secret key.

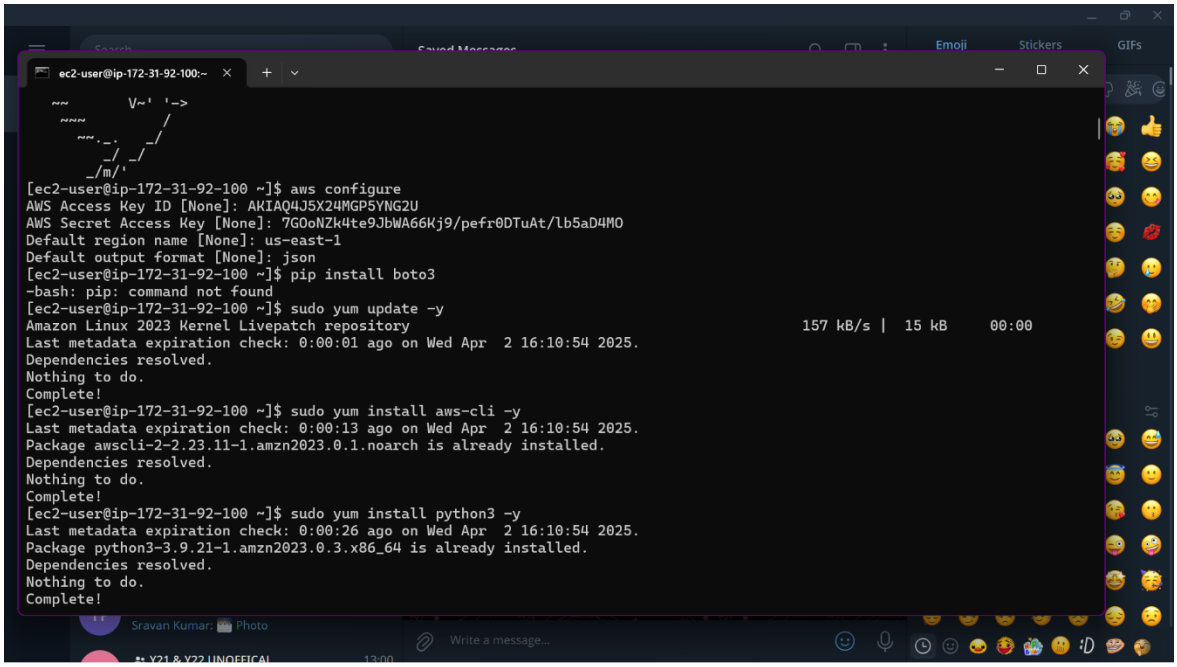


Fig 4.3.2 Installing all the required libraries and packages

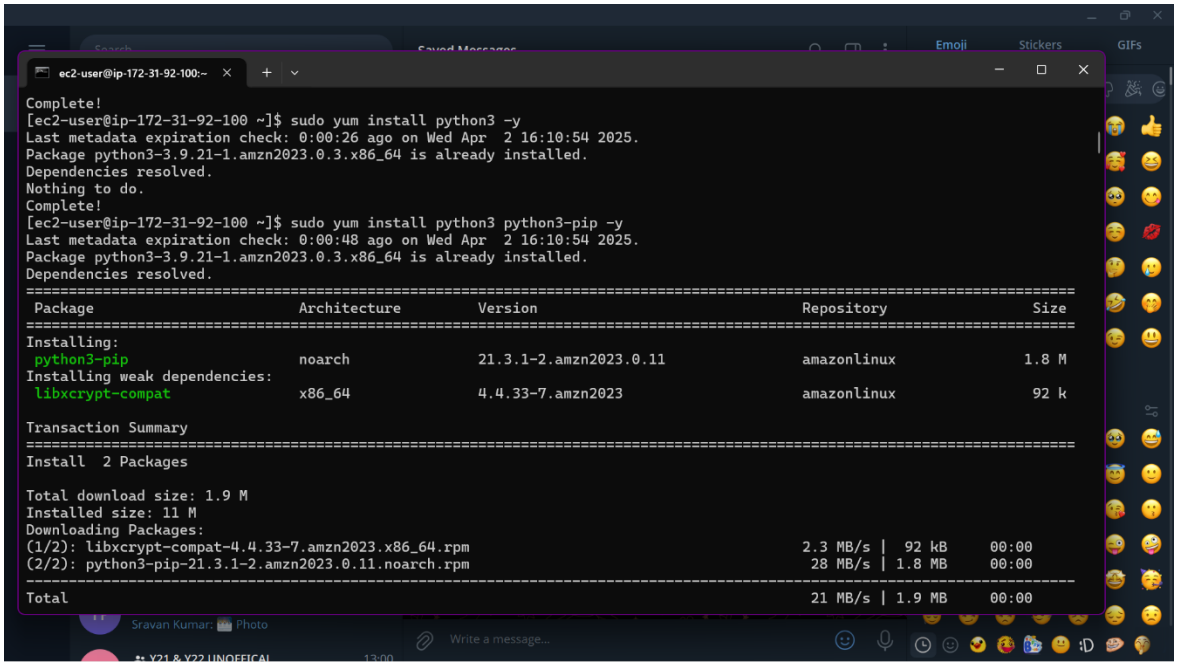


Fig 4.3.3 Checking the version of pip

Step 4 : Creating a Kinesis Data Stream

- Click on create data stream → name the data stream → data stream capacity(provisioned) → click create data stream
- Successfully created data stream.

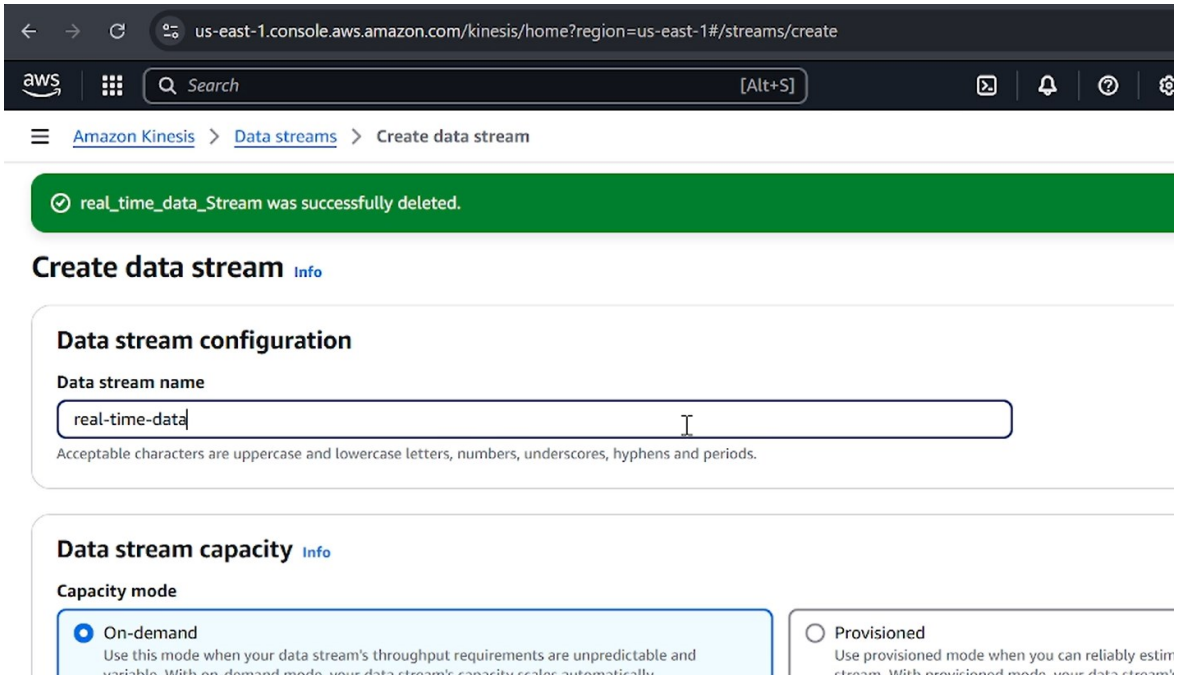


Fig 4.4.1 Create a Amazon Kinesis data stream to analyize the real time data.

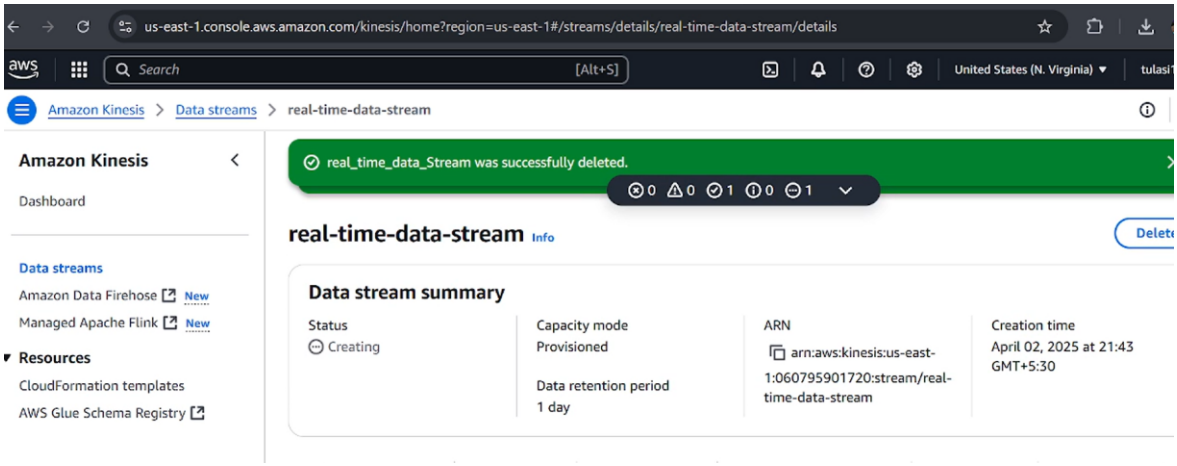
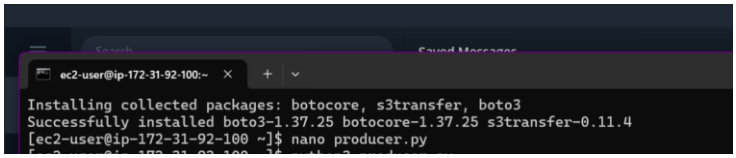


Fig 4.4.2 Wait until you get the active status.

Step 5 : Create a python file

- In cmd write a command (nano producer.py) → write youe code which creates th e temperature senson real time data → save it.
- run the code using : Python3 producer.py (command)
- you can see the streaming data.
- You can see the data in the Kinesis data stream.



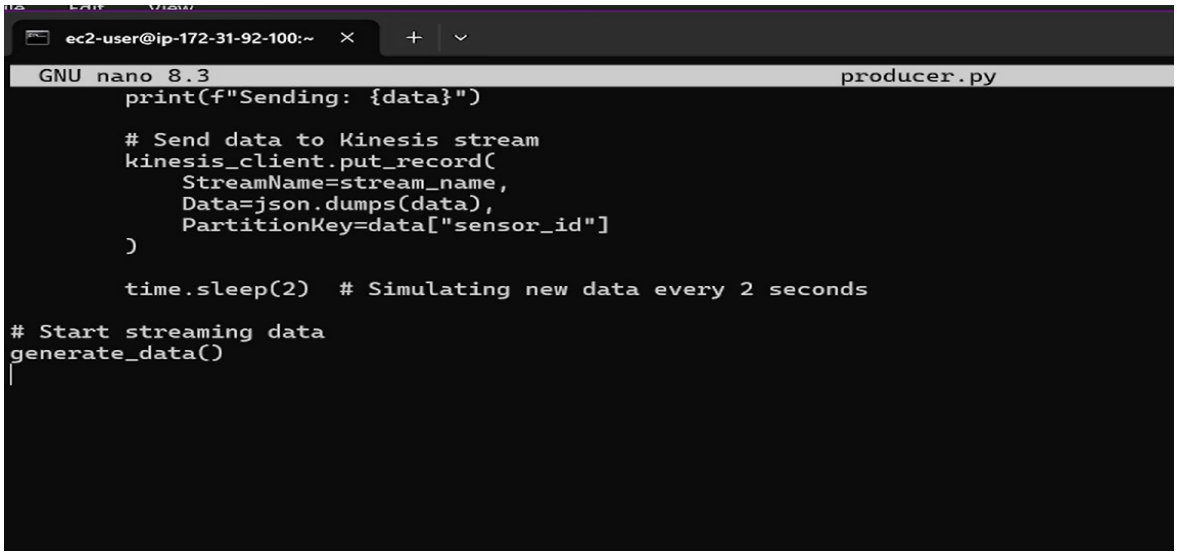


Fig 4.5.1 Create a python file(nano)

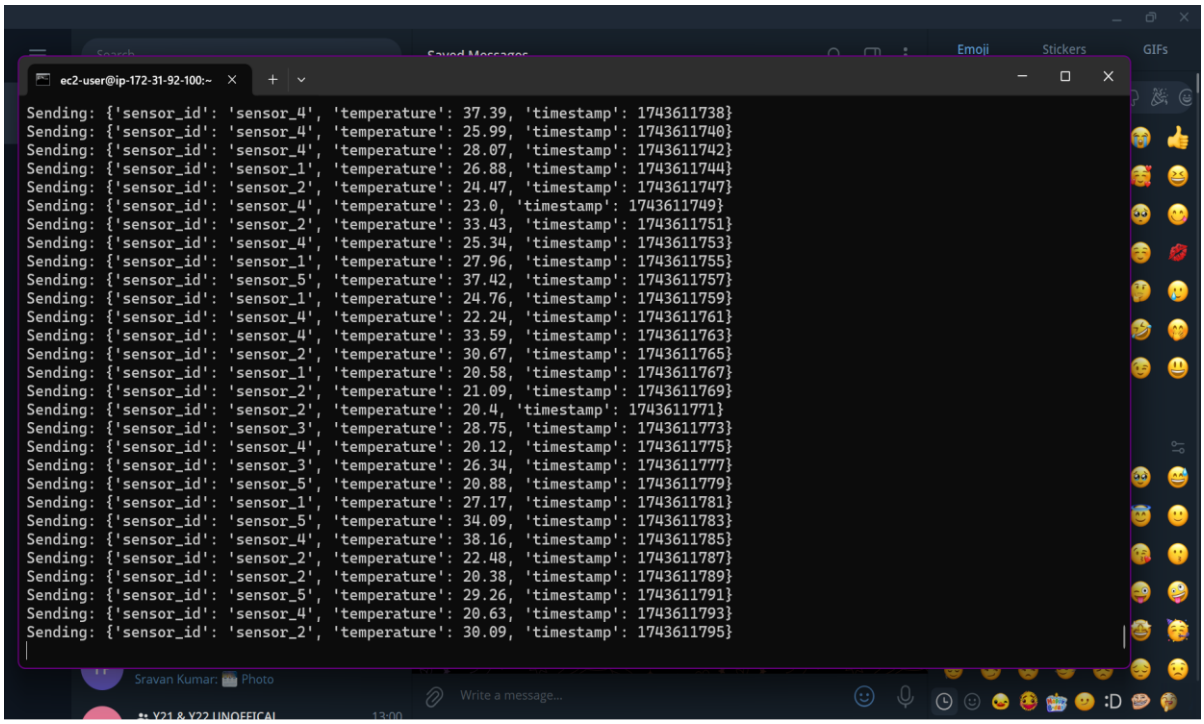


Fig 4.5.2 Here is the real time data

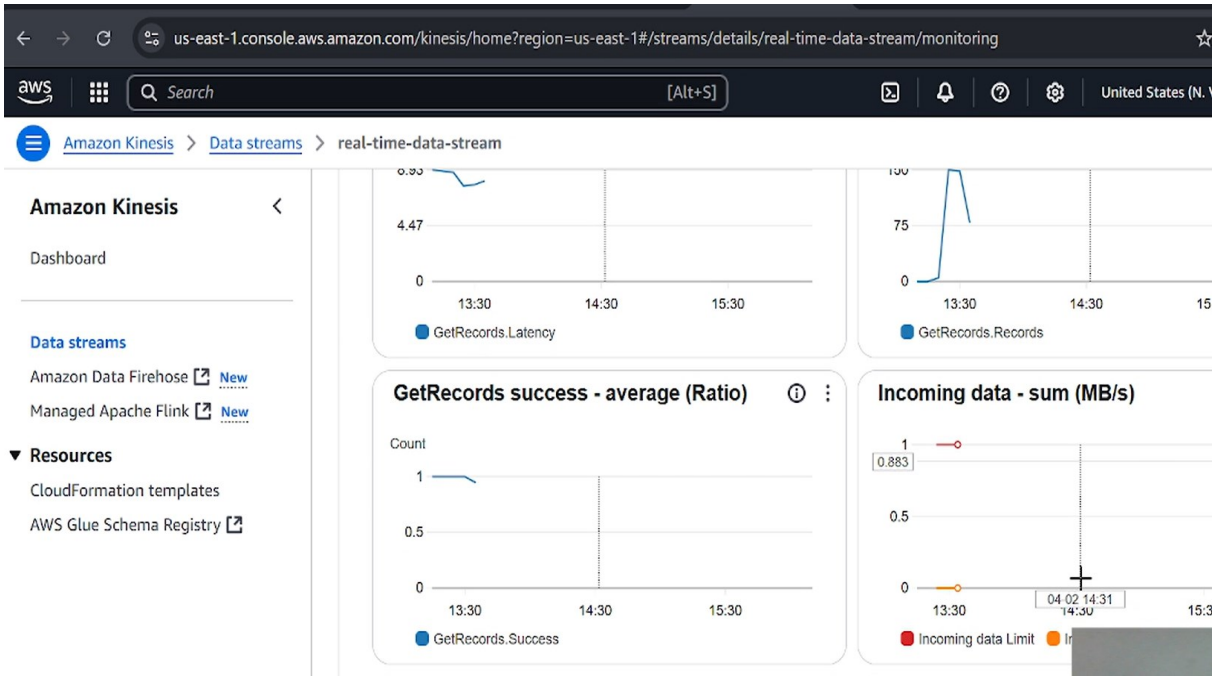


Fig 4.5.3 Here is the incoming data to the data stream .

Step 6 : Creating the S3 Bucket to store the data.

- Click create bucket → General purpose → name the bucket.
- Successfully created bucket.

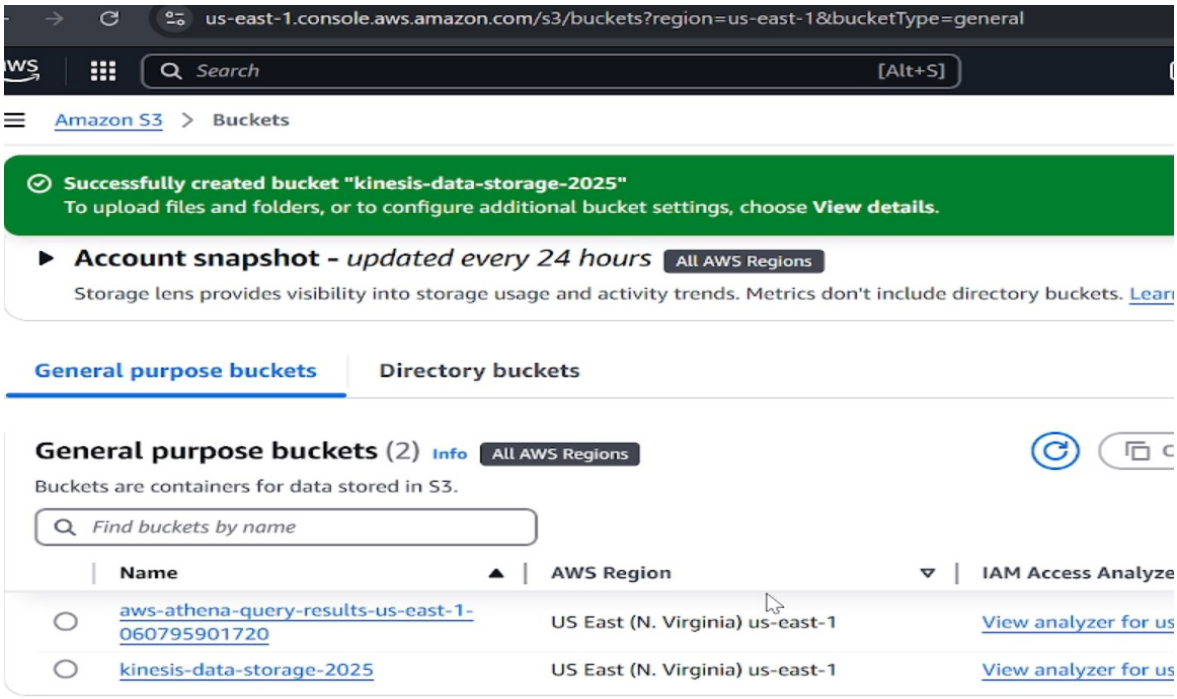


Fig 4.6 Create a bucket so that you can store the real time data in the AWS S3.

Step 7 : Creating Amazon Data Firehose to deliver the data from data stream to S3 bucket

- Click on create firehose → select the source and destination → name the firehose → add the IAM rule (If not exist create one)

  Source : Amazon Kinesis Data Streams

  Destination : Amazon S3

- Policies added :

  Amazon Kinesis Firehose Full Access

  Amazon Kinesis Full Access

  Amazon S3 Full Access.

- Click create firehose.

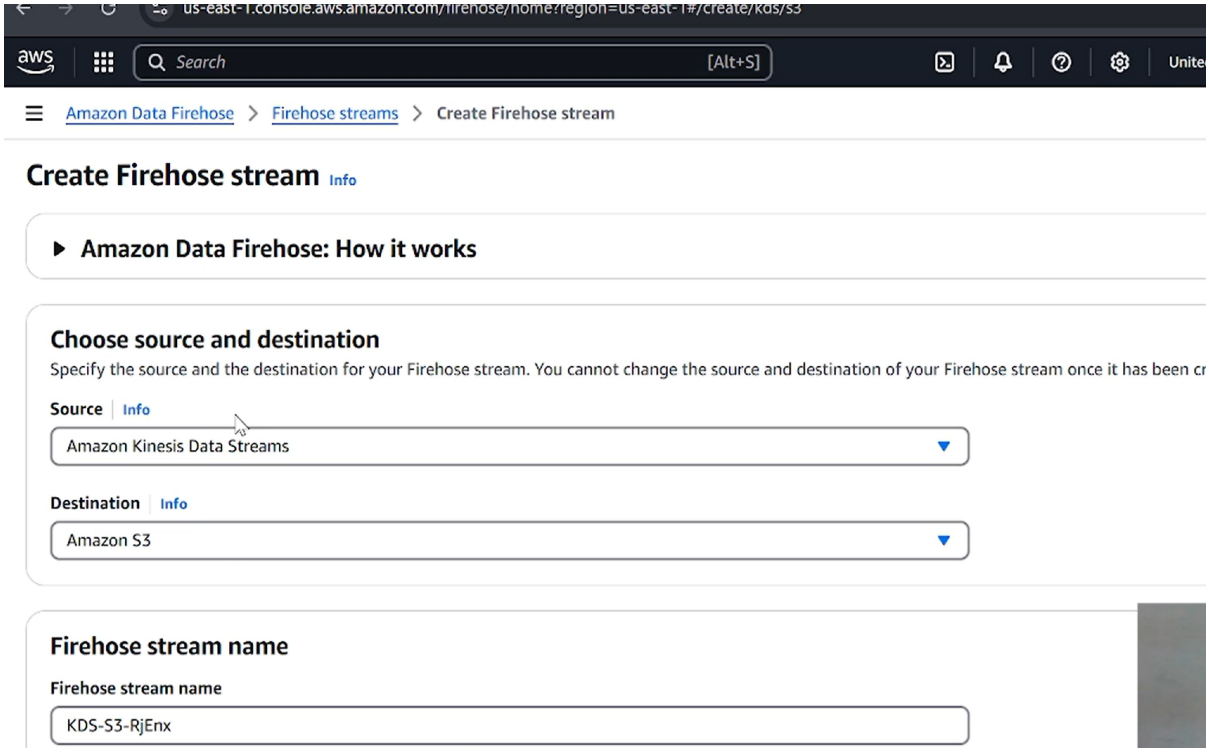


Fig 4.7.1 Creating Kinesis Firehose to deliver the data from data stream to S3 .

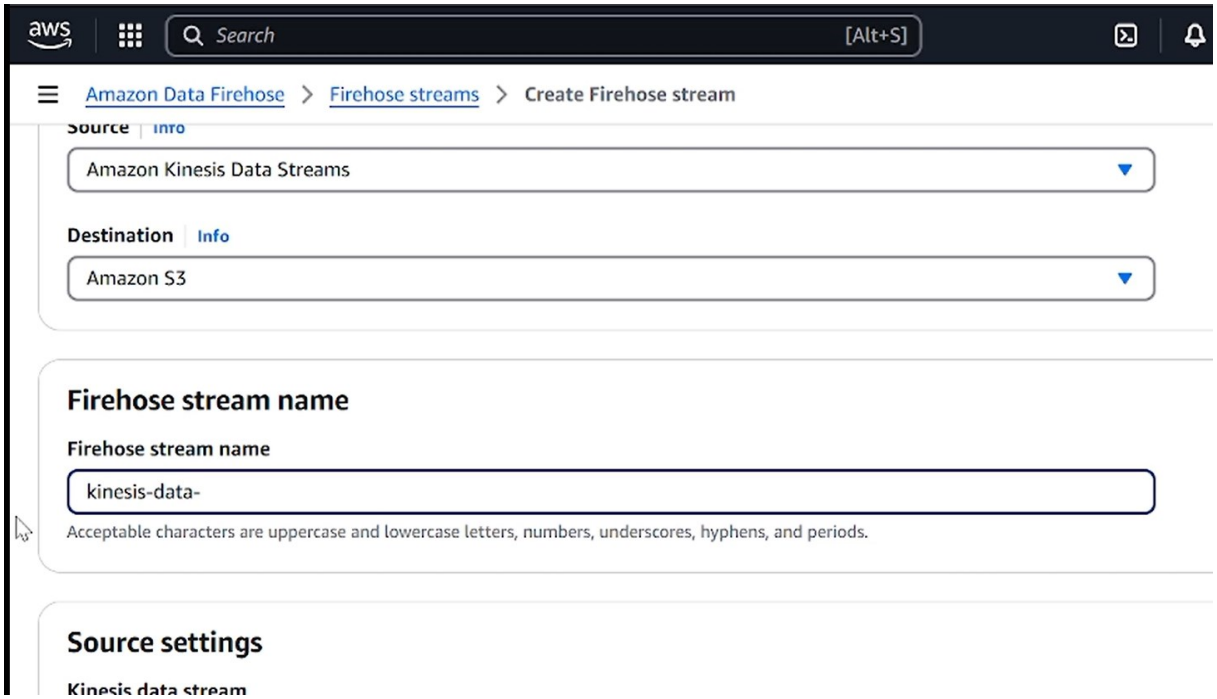


Fig 4.7.2 Name the firehose.

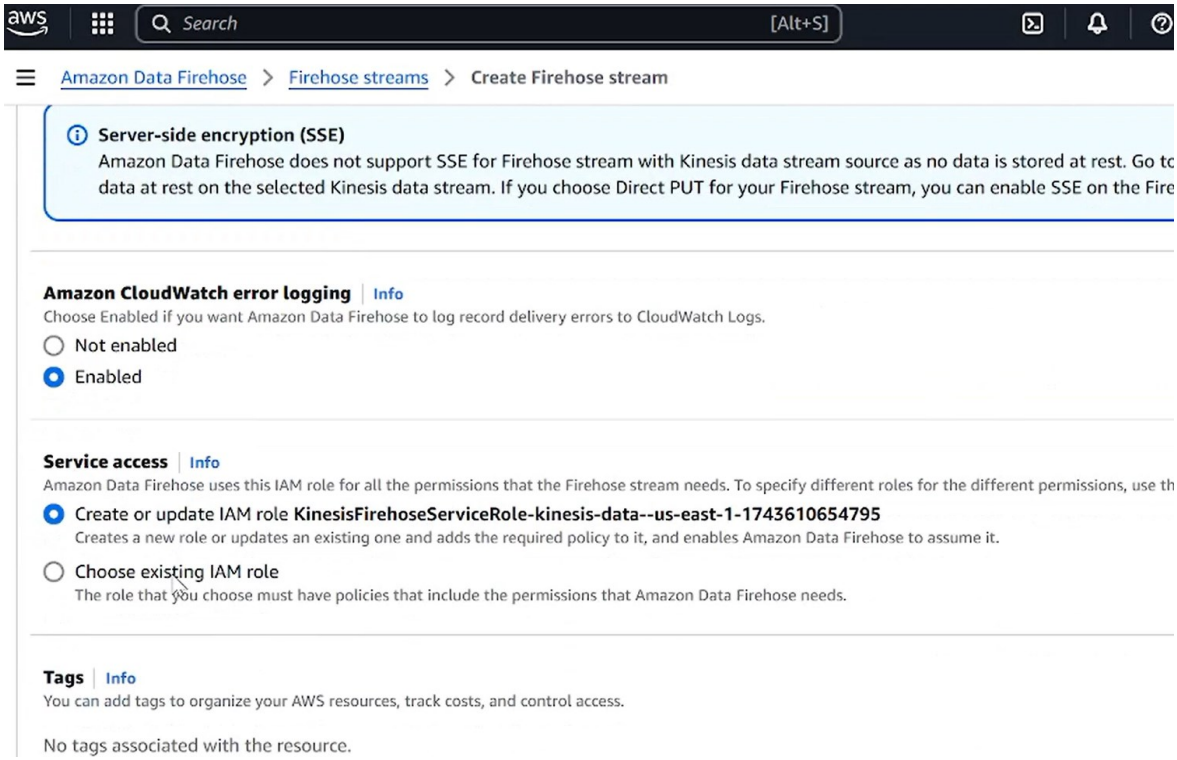


Fig 4.7.3 Create a new IAM role for firehose.

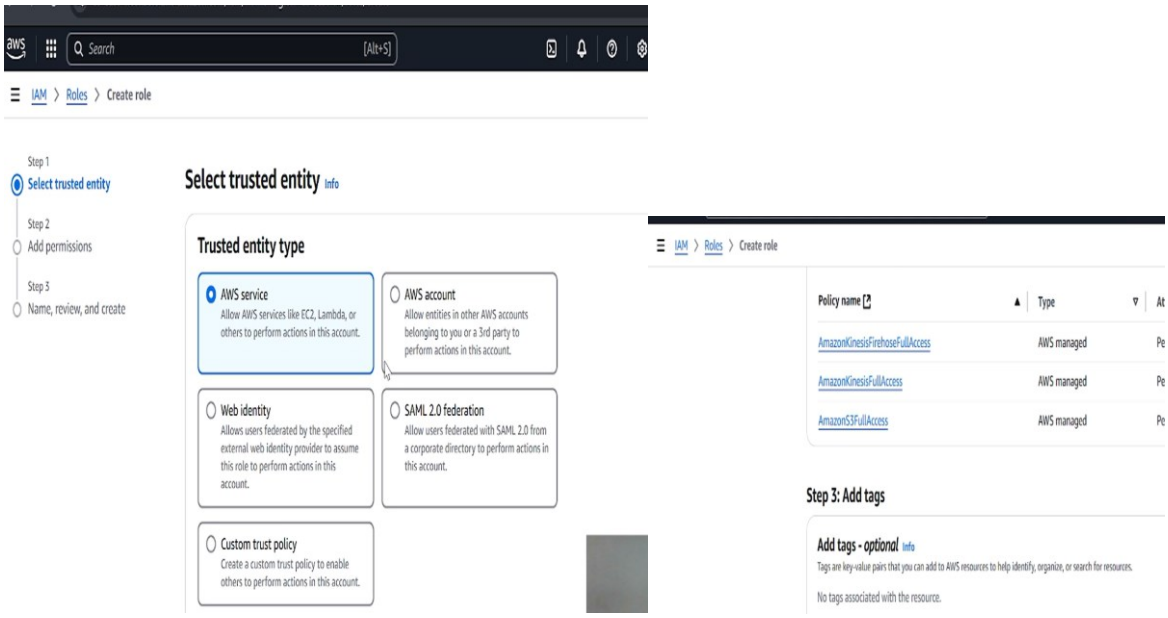


Fig 4.7.4 IAM rule

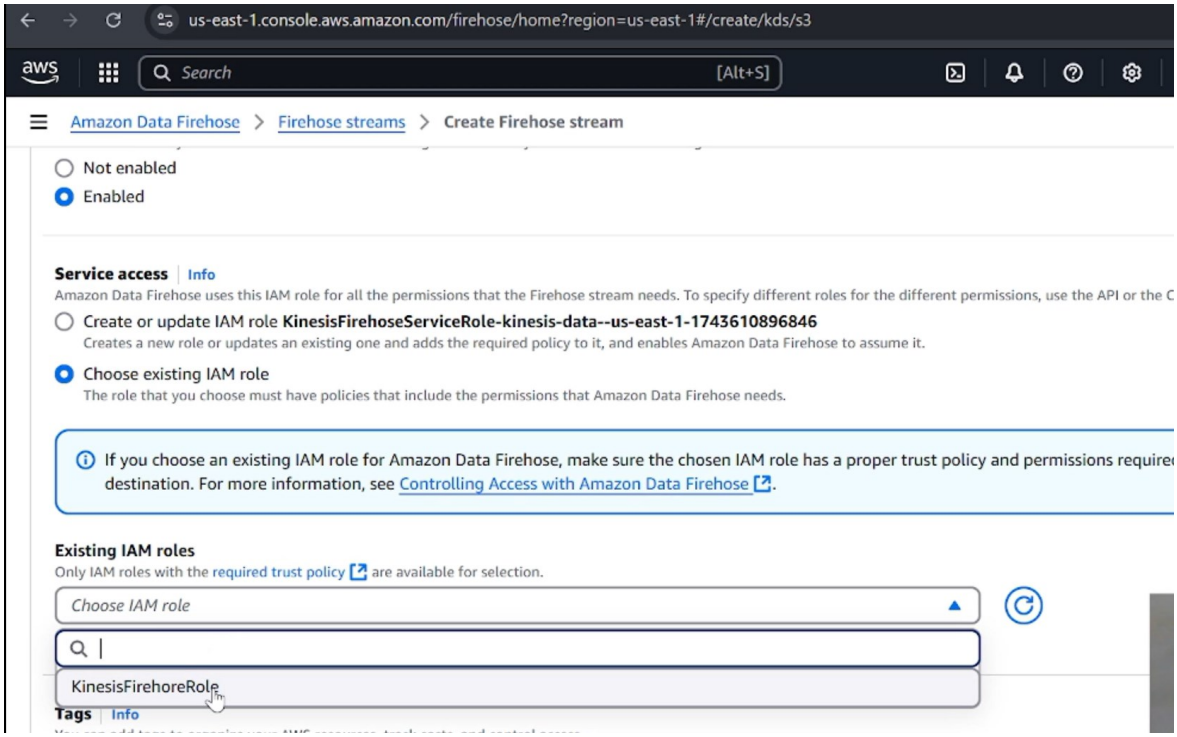


Fig 4.7.5 Select the created role for Kinesis Firehose.

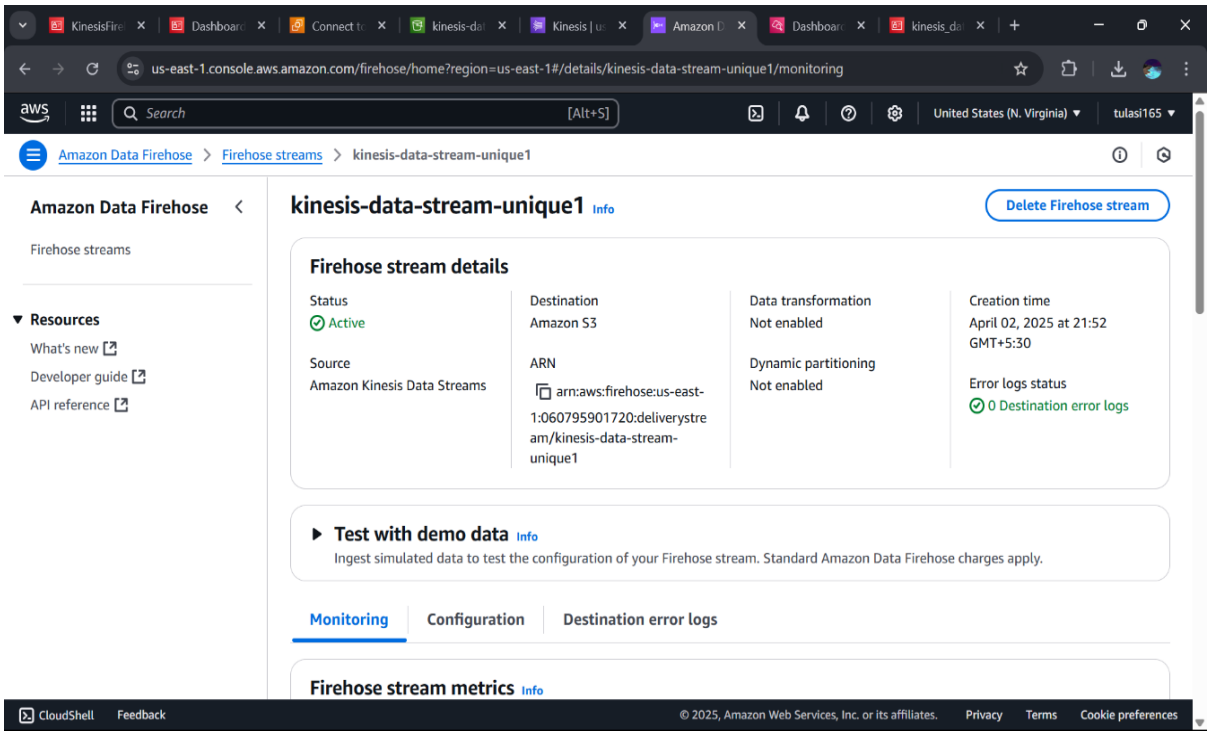


Fig 4.7.6 Kinesis firehose is successfully created.

Step 8: If still there are no objects in the bucket then edit the bucket policy

- Click on permissions → edit bucket policy.
- You can now see the objects in the Bucket.

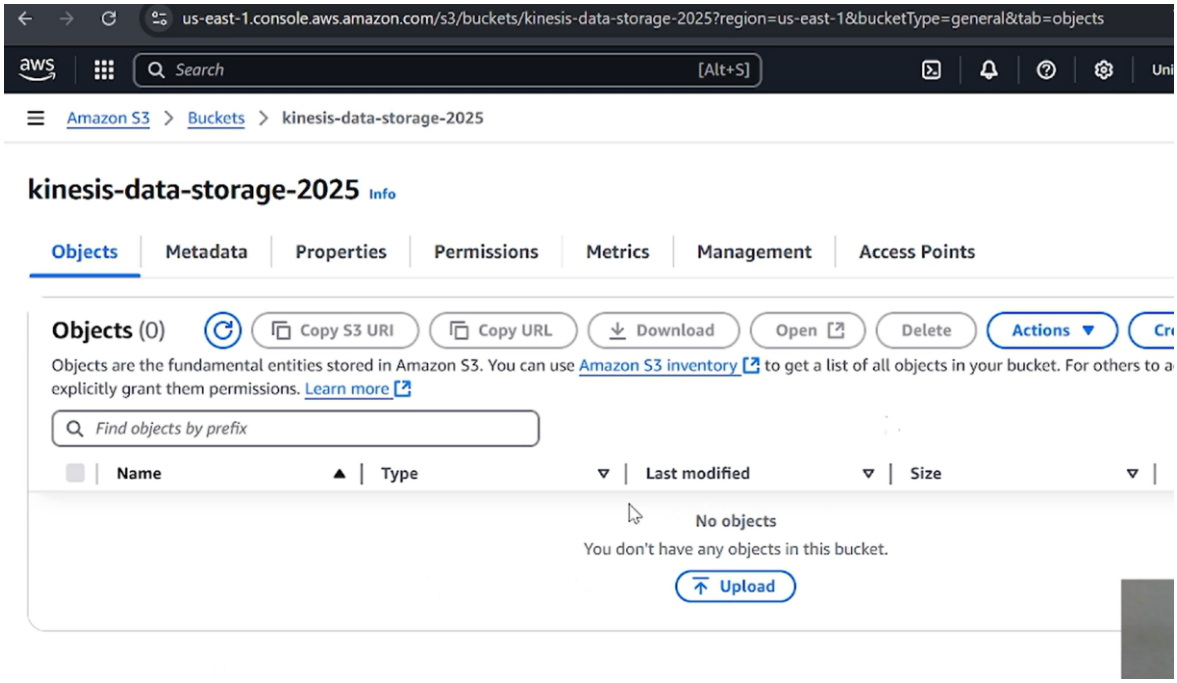


Fig 4 .8.1 Still there is no object in the bucket , you need to edit the bucket policy .

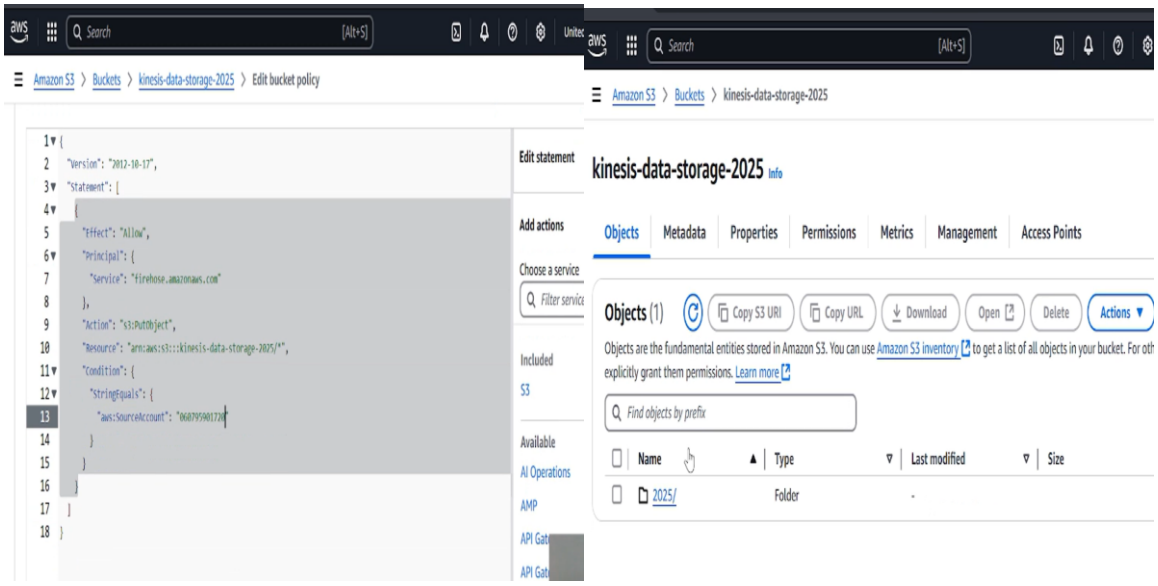


Fig 4 .8.2 Edit the bucket policy and save changes.

Step 9 : Create the dashboard for the Kinesis data stream.

- Check the Firehose monitoring for data incoming and outgoing.
- Click on add to dashboard → create a dashboard → name the dashboard
- You can see the final dashboard.

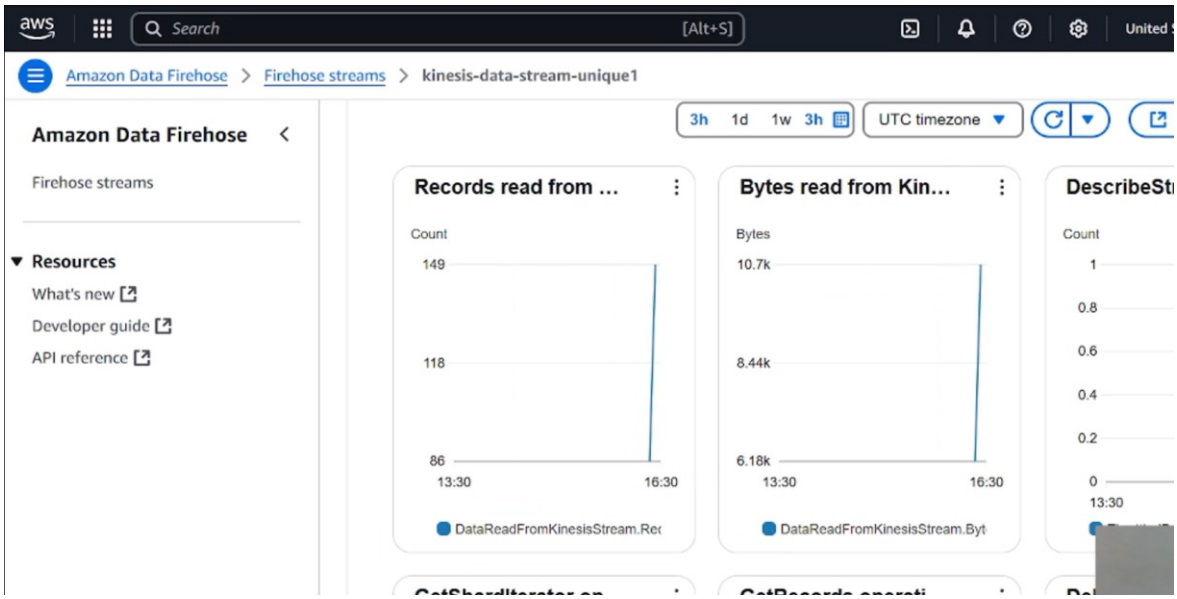


Fig 4 .9.1 Here is the graph with shows records read form kinesis and records delivered to S3.

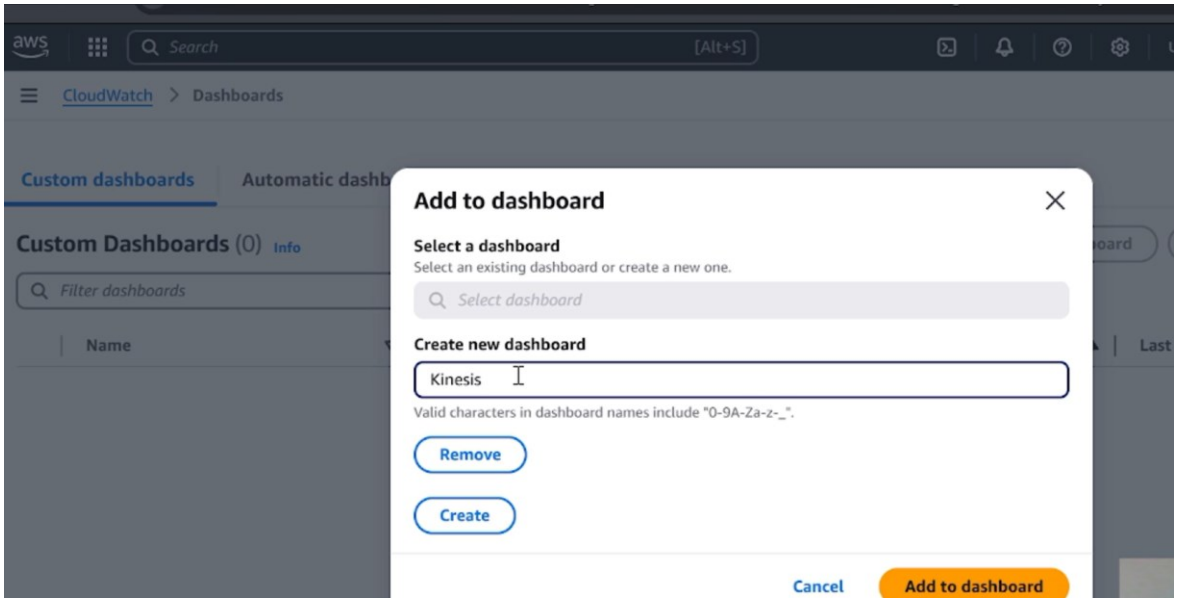


Fig 4.9.2 Dasdboard creation

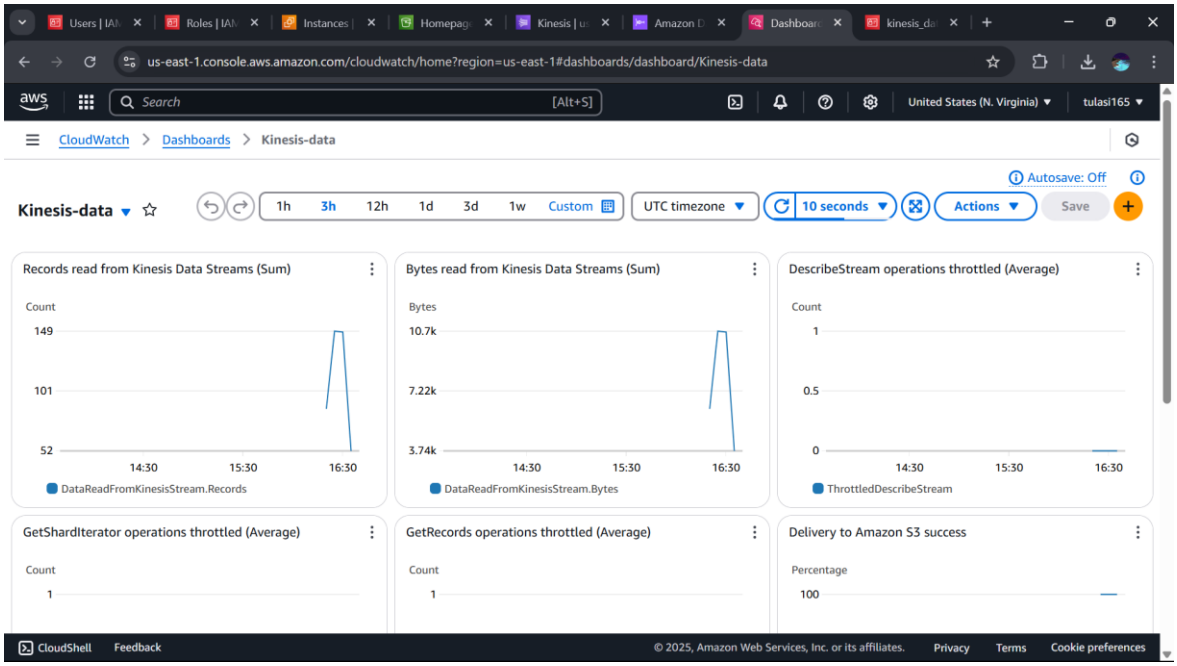


Fig 4.9.3 Here is the dashboard for real time data(temperature sensor data) using Amazon Kinesis.

5. LEARNING OUTCOMES

Through the development of the Serverless Real-time Analytics Dashboard using AWS Kinesis, several key technical and conceptual skills were acquired:

Understanding AWS Kinesis for Real-Time Data Streaming

This project provided hands-on experience in ingesting and processing high-velocity data streams using Amazon Kinesis Data Streams, enabling efficient handling of continuous data flows [1].

EC2 for Running Serverless Analytics Dashboard

The use of Amazon EC2 to execute and manage real-time analytics showcased the ability to integrate server-based resources with serverless services, ensuring efficient computational capabilities for data processing [2].

Data Storage and Pipeline Optimization with Amazon S3 and Firehose

Leveraging Amazon S3 as a data lake helped in storing raw and processed data efficiently. Additionally, Amazon Kinesis Firehose was used for automatic data delivery and transformation into storage systems, enhancing the robustness of real-time pipelines [3].

Enhanced Monitoring for Analytics

The project reinforced skills in AWS CloudWatch for real-time system monitoring and logging, helping in performance optimization and failure detection [4].

Scalability and Fault Tolerance in Cloud Architectures

Designing a system capable of handling high-throughput data streams emphasized the principles of fault tolerance, high availability, and scalability in cloud-based architectures [6].

This project has enhanced the understanding of cloud-native architectures and the practical applications of AWS services in real-time analytics.

6. CONCLUSION

The Serverless Real-time Analytics Dashboard using AWS Kinesis demonstrated how AWS services like Kinesis Data Streams, Firehose, S3, EC2, and CloudWatch can efficiently process, store, and analyze streaming data in real time. The project involved creating a temperature sensor data pipeline, which captured and visualized real-time data on an AWS dashboard, enhancing skills in IoT data processing and cloud storage. It also highlighted the design of fault-tolerant and scalable architectures using AWS-managed services, reducing operational complexity while ensuring high performance. This hands-on experience strengthened proficiency in cloud computing, data engineering, and system monitoring, laying a solid foundation for future advancements in cloud-based analytics and IoT applications.

7. FUTURE ENHANCEMENTS

To further improve the functionality and scalability of the serverless real-time analytics dashboard, several enhancements can be considered. Integrating Amazon SNS or EventBridge can enable real-time alerting to notify users of critical thresholds or anomalies. Incorporating Amazon SageMaker or Lookout for Metrics can bring predictive analytics and anomaly detection capabilities, allowing the system to forecast trends and automatically identify unusual patterns. Additionally, a custom front-end dashboard using React, API Gateway, and Lambda can replace QuickSight for greater interactivity and user control. Implementing AWS Cognito and fine-grained IAM policies would enhance security through role-based access management. For long-term storage optimization, S3 lifecycle policies with Glacier archiving can be utilized. Moreover, expanding data ingestion from multiple sources, adding real-time data transformation with Lambda in Firehose, and deploying across multiple regions for high availability can significantly elevate the system's performance, reliability, and user experience.

8. REFERENCES

[1] Amazon Personalize Customer Outreach on Your E-commerce Platform: https://aws.amazon.com/blogs/architecture/amazon-personalize-customer-outreach-on-yourecommerce-platform/

[2] E-commerce Personalization with AWS: https://www.brilworks.com/blog/ecommerce-personalization-with-aws/

[3] Research Paper on E-commerce Personalization (arXiv): https://arxiv.org/abs/2105.06508

[4] Real-time Analytics Use Cases and Examples: https://www.striim.com/blog/real-time-analytics-use-cases-and-examples/#use

[5] AWS in Healthcare Industry: https://aws.amazon.com/blogs/industries/category/industries/healthcare/

[6] Detect Game Anomalies with Amazon Lookout for Metrics: https://aws.amazon.com/blogs/gametech/detect-game-anomalies-amazon-lookout-for-metricsgame-analytics-pipeline/

[7] Real-time Fraud Detection Using AWS Serverless and Machine Learning Services: https://aws.amazon.com/blogs/machine-learning/real-time-fraud-detection-using-aws-serverlessand-machine-learning-services/

[8] Sentiment Analysis of Social Media – Big Data Analytics Options: https://docs.aws.amazon.com/whitepapers/latest/big-data-analytics-options/example-3-sentimentanalysis-of-social-media.html

[9] Log Analytics with AWS: https://aws.amazon.com/what-is/log-analytics/#:~:text=cost%20of%20operations.-,How%20does%20log%20analytics%20with%20AWS%20work?,for%20real%2Dtime%20operational%20intelligence.

[10] Amazon Kinesis Video Streams: https://aws.amazon.com/kinesis/video-streams/?amazon-kinesis-video-streams-resourcesblog.sort-by=item.additionalFields.createdDate&amazon-kinesis-video-streams-resourcesblog.sort-order=desc