CALIFORNIA STATE UNIVERSITY
FULLERTON

Department of Computer Science

Project Report

on

Architecture to capture Realtime event data using AWS ecosystem

CPSC 531-02 (Spring, 2023)

We express our deep sense of gratitude to **Prof. Tseng-Ching James Shen, Ph.D.**, for his encouragement to develop something innovative during this project and the course.

Team Members:

Sai Kumar Reddy Bandi (CWID: 885175166)

Sai Venkata Naresh Kumar Digutla (CWID: 885190603)

Contents

List of Figures

1. Introduction

Capturing real-time event data has become a critical aspect of modern business operations, enabling organizations to make data-driven decisions and respond quickly to changing circumstances. The rise of cloud computing has made it easier than ever to capture, store, and analyze large volumes of real-time data. Amazon Web Services (AWS) provides a comprehensive ecosystem of tools and services designed to capture and process real-time data, making it an attractive choice for businesses looking to leverage real-time data to gain a competitive edge.

AWS offers a range of tools and services for capturing real-time event data. We used Amazon API Gateway, AWS Glue, AWS Athena, Amazon Kinesis, AWS Lambda, Amazon EventBridge, AWS CloudWatch, Tableau, and Amazon S3 to process, transform and visualize real-time event data in our project.

Amazon API Gateway is a fully managed service provided by AWS that enables developers to create, deploy, and manage APIs (Application Programming Interfaces) for their applications. With Amazon API Gateway, developers can easily build scalable APIs that can be used by their applications or third-party developers. The service supports RESTful APIs, WebSocket APIs, and HTTP APIs, allowing developers to choose the API type that best suits their needs.

Amazon EventBridge is a serverless event bus that allows users to route events between different AWS services and applications. With EventBridge, users can set up rules to trigger actions in real-time based on events from multiple sources. This makes it easy to build complex event-driven architectures that can handle large volumes of real-time data.

Amazon Kinesis is a fully managed service that allows users to collect, process, and analyze streaming data in real-time. With Kinesis, users can ingest data from a variety of sources, including websites, mobile devices, IoT sensors, and other data streams. Kinesis can process data at scale and in real-time, making it ideal for use cases such as log processing, clickstream analysis, and IoT data processing.

AWS Lambda is a serverless computing service that allows users to run code without provisioning or managing servers. With Lambda, users can process data in real-time, triggered by events from other AWS services such as Kinesis, S3, or EventBridge. This makes Lambda a flexible and powerful tool for real-time event processing and data analysis.

Amazon S3 (Simple Storage Service) is a highly scalable, secure, and durable object storage service offered by AWS. It allows users to store and retrieve any amount of data from anywhere on the web. S3 provides virtually unlimited storage capacity, and users only pay for what they use. S3 can store and retrieve any type of data, including images, videos, documents, and application backups. S3 also provides features such as versioning, encryption, lifecycle policies, and access control, making it an ideal choice for storing data in a variety of use cases.

AWS Glue is a fully managed ETL (Extract, Transform, and Load) service that makes it easy to move data between data sources and destinations. It provides a managed infrastructure that automates the tasks of discovering data, transforming it into a format suitable for analysis, and moving it to the target destination. With Glue, users can extract data from various sources such as databases, flat files, and streaming sources, transform it using a visual editor or custom code, and load it into various data stores, including S3, Redshift, and other databases.

AWS CloudWatch is a monitoring and observability service provided by Amazon Web Services (AWS). It is designed to collect and track metrics, collect and monitor log files, and set alarms to gain insights into the performance and operational health of AWS resources and applications.

Amazon Athena is an interactive query service that allows users to analyze data stored in Amazon S3 using SQL. Athena provides a serverless, pay-per-query model that allows users to analyze data quickly and easily without the need to provision or manage infrastructure. With Athena, users can query data stored in S3 using standard SQL syntax and visualize the results using various BI tools.

Tableau is a widely used data visualization and business intelligence (BI) software that helps organizations analyze and understand their data. It provides a powerful and intuitive platform for creating interactive visualizations, dashboards, and reports, enabling users to explore and communicate insights from their data effectively.

2. Functionalities

- Third-party vendors will send data through an API via a POST request, and we will collect it in near real-time.
- The data from the vendors will be then captured by the API gateway by exposing the API endpoint to Amazon API gateway.
- The data will be fed from the Amazon API Gateway to Amazon EventBridge, an event bus that connects different applications and enables many-to-many routing.
- From the Amazon EventBridge, the data will go through some set of defined rules and will help us divert the data to different destinations.
- In the destination, we have Amazon firehose to capture the data from EventBridge. AWS lambda is used to do transformations on the Firehose.
- The transformed data is then stored to S3 which would act as a Data Lake.
- Later we can query the data from the S3 locations by Athena and AWS glue is used to create databases and tables.
- Tableau, a third-party data visualizing tool will be connected to AWS Athena and Athena database will be analyzed.

3. Architecture and Design

Architecture diagram:

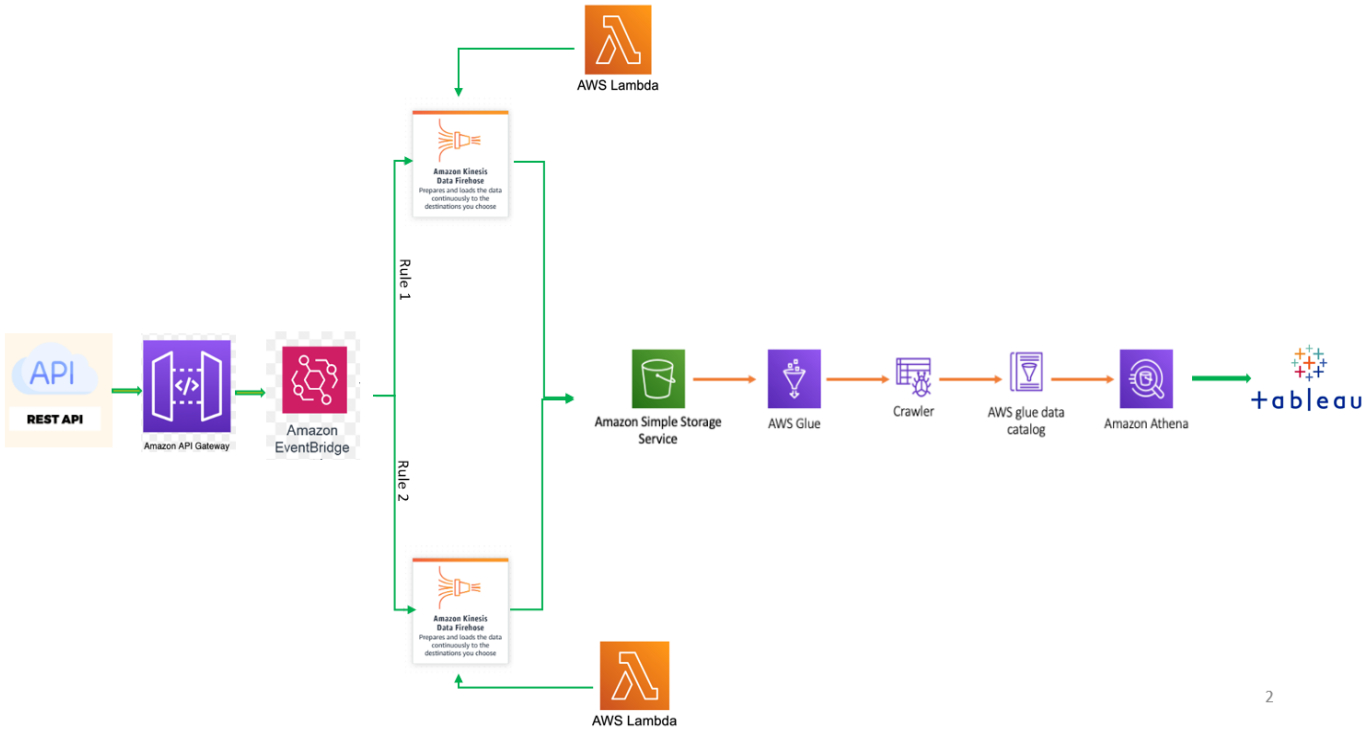

Fig 3.1: Architecture diagram

- In the project, we will take a csv file which contains data about sport events, and we will feed this data to Amazon API Gateway using python code.
- Then the data will be sent to Amazon EventBridge which will have a data pipeline called databus, where we will write rules to make decisions to direct it to different destinations.
- The data will be sent to Kinesis firehose which is a ETL service where we make transformations on the data using AWS Lambda.
- The transformed data will be stored in AWS S3, we will run AWS Glue crawler to create meta data and we store it in AWS Glue catalogue.
- We use Athena, to query the data that is stored in S3 with the help of metadata stored in Glue catalogue.
- At last, we will connect Tableau to Athena to visualize the data.

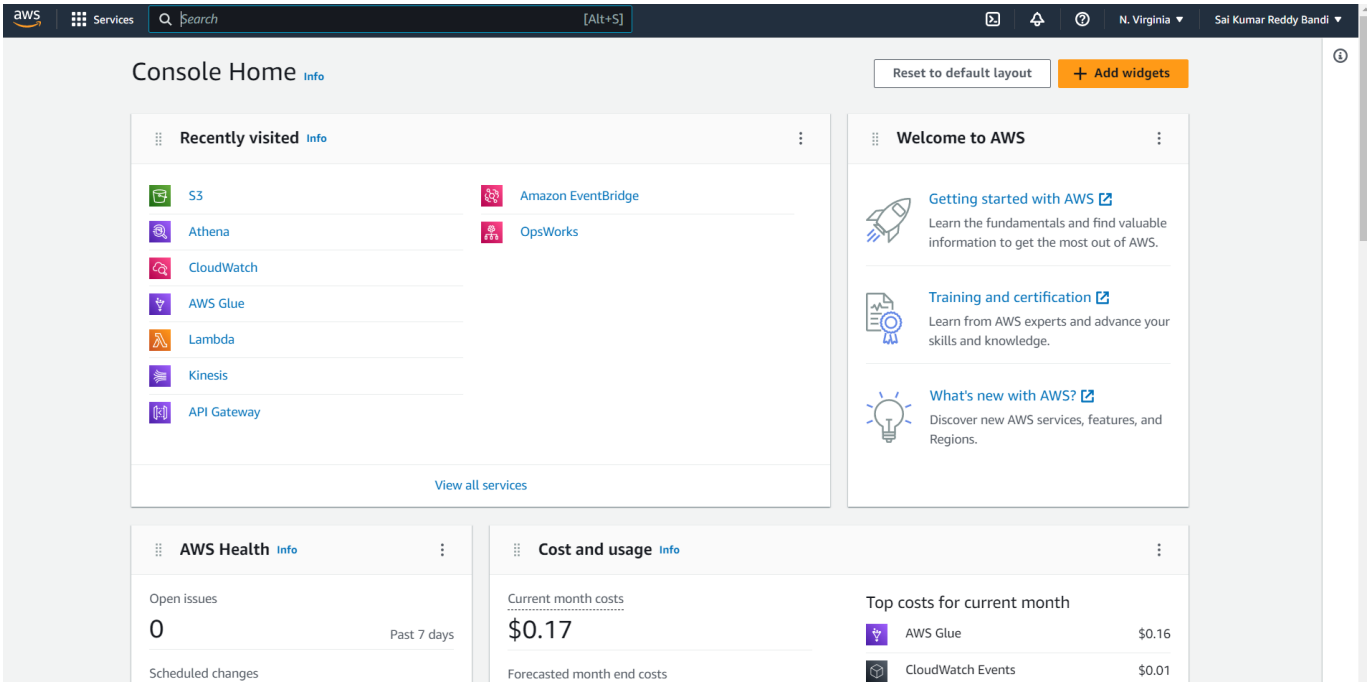

Fig 3.2: AWS Console Dashboard

4. GitHub location of the code

Click on the below image to redirect to the GitHub location of the project.

Note: We have uploaded project report and the python codes used to run the project.

5. Deployment Instructions

Create Amazon API Gateway:

Follow below steps to create and configure API Gateway:

- Create an AWS free tier account and sign into the AWS Management Console and open the API Gateway service.
- Click on "Create API" to start creating a new API.
- Choose the "REST" protocol and select the "New API" option.
- Provide a name for your API and choose the desired endpoint type (Edge-optimized, Regional, or Private).
- Click on "Create API" to create the API.
- Once the API is created, you'll be redirected to the API Gateway dashboard. From there, click on "Actions" and select "Create Method" for the desired resource.
- In the "Create Method" dropdown, choose "POST" as the HTTP method.
- Configure the integration type based on where you want to send the POST request data. In our case, we are integrating it with EventBridge.
- After configuring the integration, click on "Save" to save the changes.
- Once you have finished configuring the method settings, click on "Deploy API" to deploy your API to a stage. Choose an existing stage or create a new one.
- After deployment, you will receive an endpoint URL for your API. This URL can be used to send POST requests to the API Gateway.

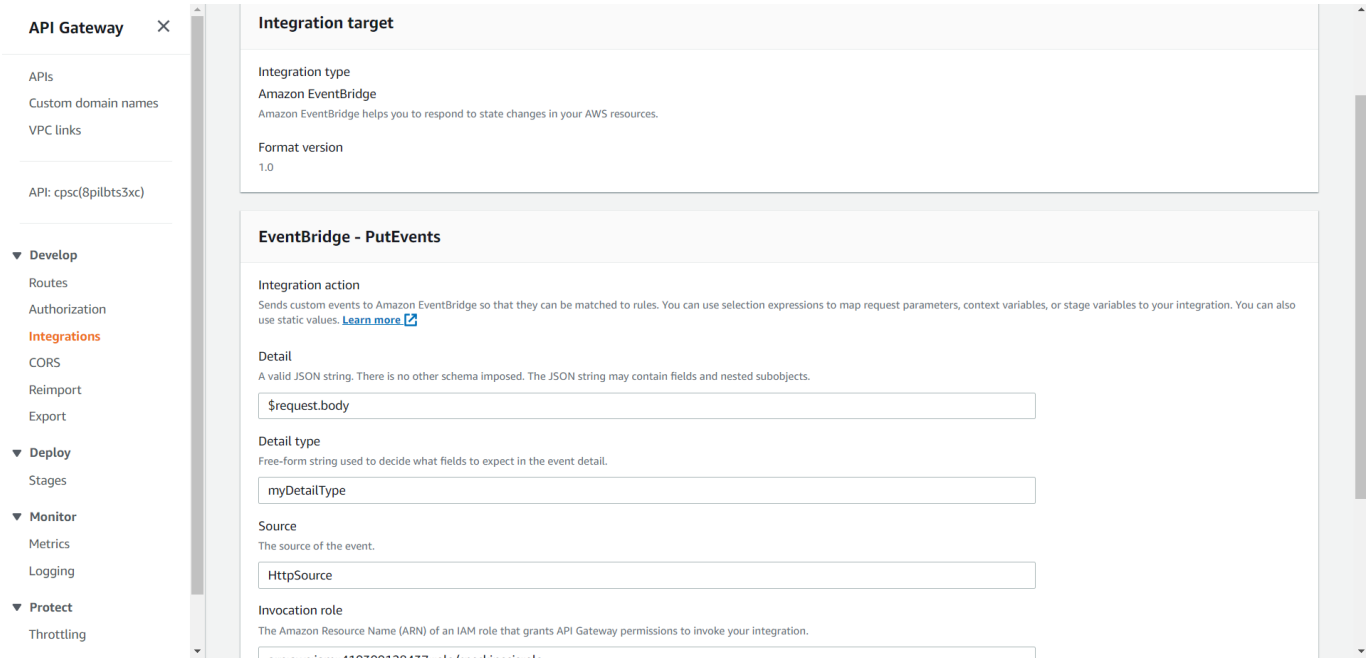

Fig 5.1: API Gateway

Create and Configure Amazon EventBridge

Follow below steps to create and configure EventBridge:

- In the AWS Management Console, open the EventBridge service.
- Click on "Create rule" to start creating a new rule.
- Provide a name and description for your rule.
- Under "Targets," click on "Add target" and select "Kinesis Firehose."
- Choose the specific Kinesis Firehose delivery stream to which you want to send the events.
- Configure the rule's target options, such as input transformer (optional) and batch size.
- Review the rule settings and click on "Create rule" to create the rule.
- Now, any events that match the defined pattern will be sent to the specified Kinesis Firehose delivery stream.
- In our case we created two rules to detect event type and we chose two kinesis destinations if the rules are matched.

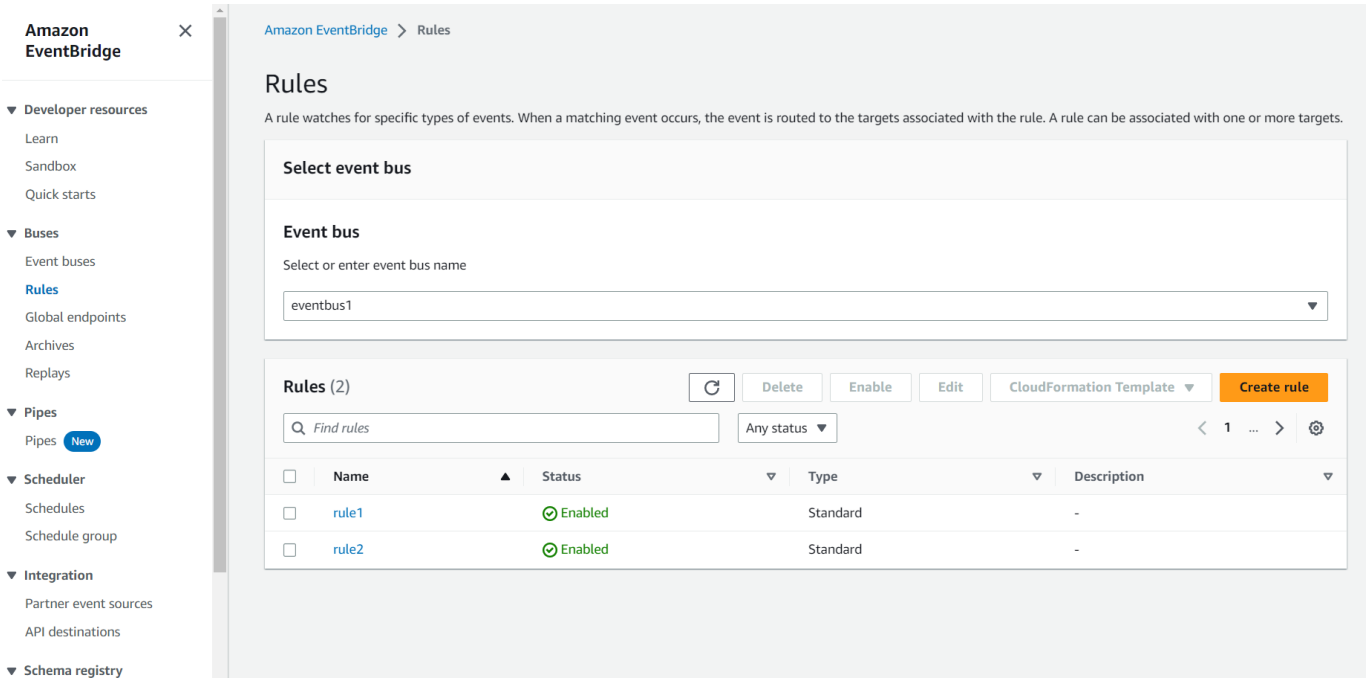

Fig 5.2: Amazon EventBridge

Create and Configure Amazon Kinesis Firehose

Follow below steps to Create and Configure Amazon Kinesis Firehose:

- In the AWS Management Console and open the Kinesis Firehose service.
- Click on "Create delivery stream" to start creating a new delivery stream.
- Provide a name for your delivery stream and choose the desired source for data ingestion.
- Configure the transformation settings by selecting "Transform source records with AWS Lambda" as the transformation source.
- Create a lambda function to perform the data transformation. Make sure the Lambda function receives the incoming data, applies the required transformation logic, and returns the transformed data. (Download the lambda code that is in GitHub and use it for transformations in lambda)
- Configure additional settings for your delivery stream, such as buffering, data compression, and error logging.
- Specify the destination for the transformed data by selecting “Amazon S3” as the destination.
- Create a new S3 bucket to store the transformed data.
- Configure the output settings for S3, including the S3 prefix, format conversion options and encryption settings.

- Review the delivery stream settings and click on “Create delivery stream” to create the stream.
- Once the delivery stream is created, it will be ready to receive and process data. We can monitor the status and health of the delivery stream in the Kinesis Firehose console.
- In our case we created two delivery steams and attached lambda functions to each delivery stream. We are storing this transformed data in cat1, cat2 folders in AWS S3.

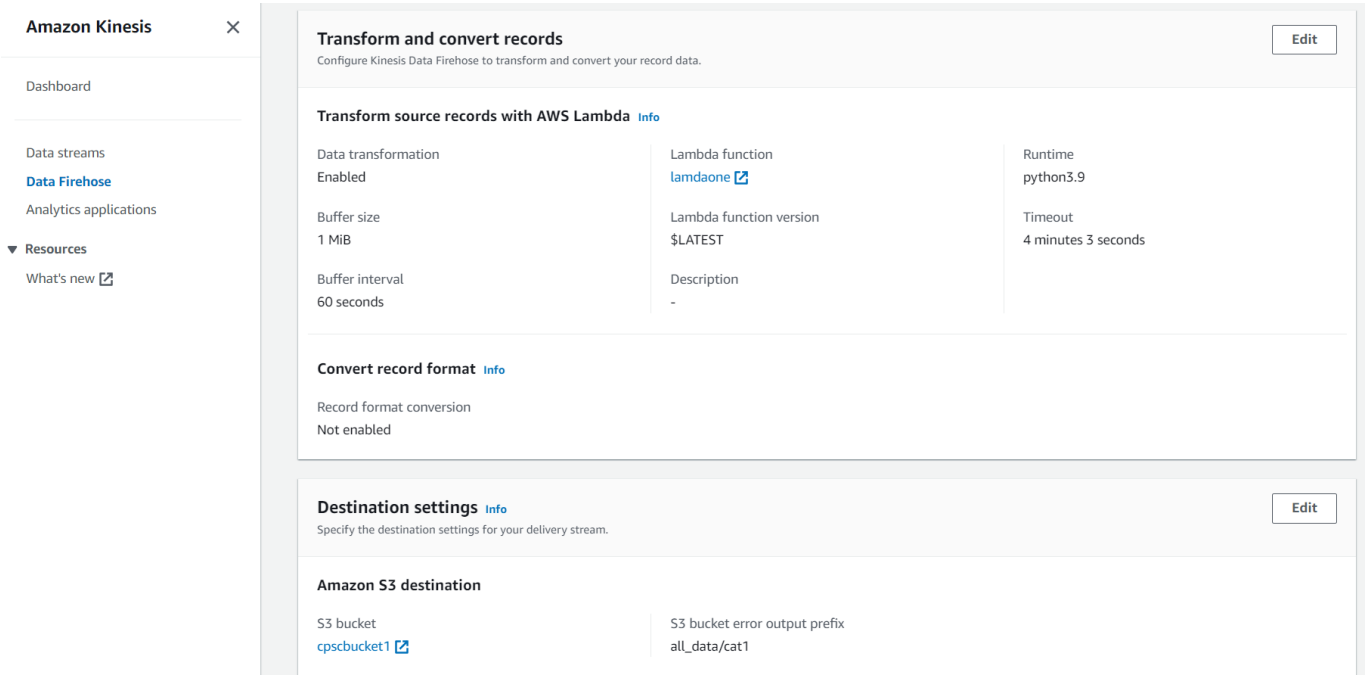

Fig 5.3: Amazon Kinesis Firehose

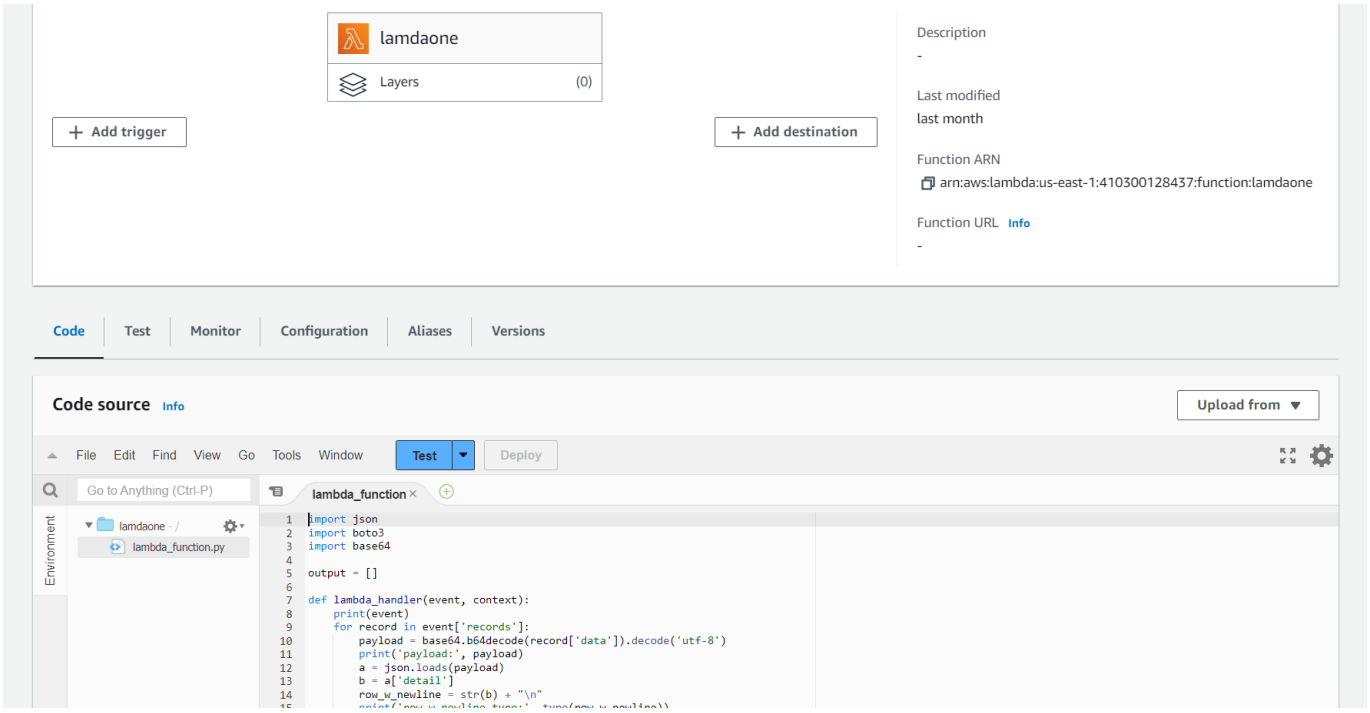

Fig 5.4: Lambda Function

Create and configure AWS CloudWatch log groups

To create and configure AWS CloudWatch log groups for Amazon API Gateway and AWS Lambda, you can follow these steps:

Creating a CloudWatch Log Group for Amazon API Gateway:

- Open the AWS Management Console and navigate to the CloudWatch service.
- In the CloudWatch console, click on “Logs” in the sidebar.
- Click on the “Create log group” button.
- Provide a unique name for your log group, such as “APIGatewayLogs”.
- Optionally, you can add a description for your log group.
- Click on the “Create log group” button to create the log group.

Creating a CloudWatch Log Group for AWS Lambda:

- Open the AWS Management Console and navigate to the CloudWatch service.
- In the CloudWatch console, click on “Logs” in the sidebar.
- Click on the “Create log group” button.

- Provide a unique name for your log group, such as “LambdaLogs”.
- Optionally, you can add a description for your log group.
- Click on the “Create log group” button to create the log group.

Configuring Amazon API Gateway to Use the Log Group:

- Open the API Gateway console and select your API.
- Navigate to the “Stages” section of your API.
- Select the desired stage and click on the “Logs/Tracing” tab.
- Click on the “Edit” button next to the “CloudWatch Settings” option.
- Enable the “Enable CloudWatch Logs” option.
- Choose the appropriate log format. You can choose either “CloudWatch Logs” or “Amazon S3”.
- In the “Log level” field, specify the desired log level for your API Gateway.
- In the “Log group name” field, enter the ARN (Amazon Resource Name) or the name of the CloudWatch log group you created for API Gateway.
- Click on the “Save Changes” button to save the configuration.

Configuring AWS Lambda to Use the Log Group:

- Open the AWS Lambda console and select your Lambda function.
- In the “Function overview” section, click on the “Monitoring” tab.
- Enable the “Enable CloudWatch Logs” option.
- Choose the appropriate log format. You can choose either “New log groups” or “Existing log groups”.
- In the “Create new log group” section, enter the name of the CloudWatch log group you created for Lambda, or choose an existing log group from the dropdown.
- Click on the “Save” button to save the configuration.

By following these steps, you can create and configure AWS CloudWatch log groups for Amazon API Gateway and AWS Lambda, allowing you to collect and analyze logs generated by these services.

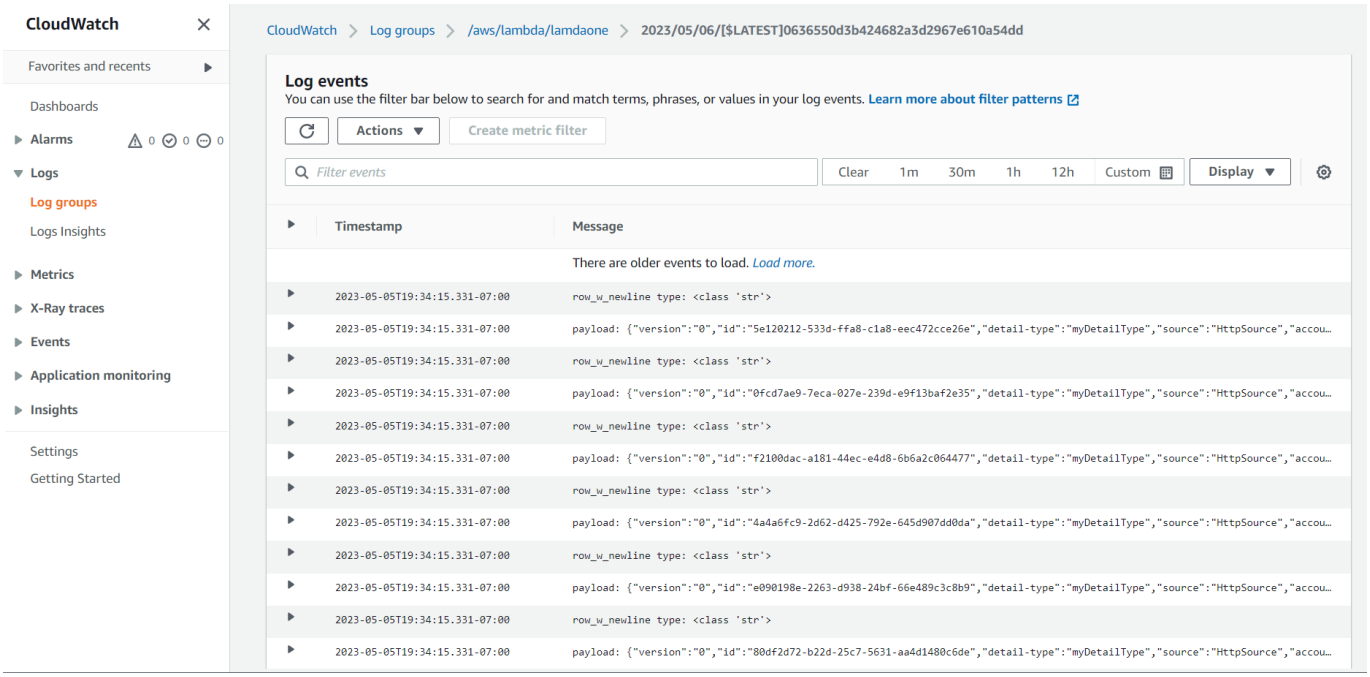

Fig 5.5: AWS CloudWatch

Create and Configure AWS S3

Follow below steps to create and configure AWS S3:

- In the AWS Management Console and open the Amazon S3 service.
- Click on “Create bucket” to start creating a new S3 bucket.
- Provide a globally unique name for your bucket. Note that bucket names must adhere to specific naming conventions.
- Choose the region where you want to create the bucket. Consider selecting a region that is geographically closest to the data source or data consumers for reduced latency.
- Configure the bucket settings. You can choose the default settings or customize them as per your requirements. These settings include versioning, logging, and encryption options.
- Review the settings and click on “Create bucket” to create the S3 bucket.
- In our case, we are creating S3 bucket and storing transformed data from Kinesis delivery streams in cat1, cat2 folders.

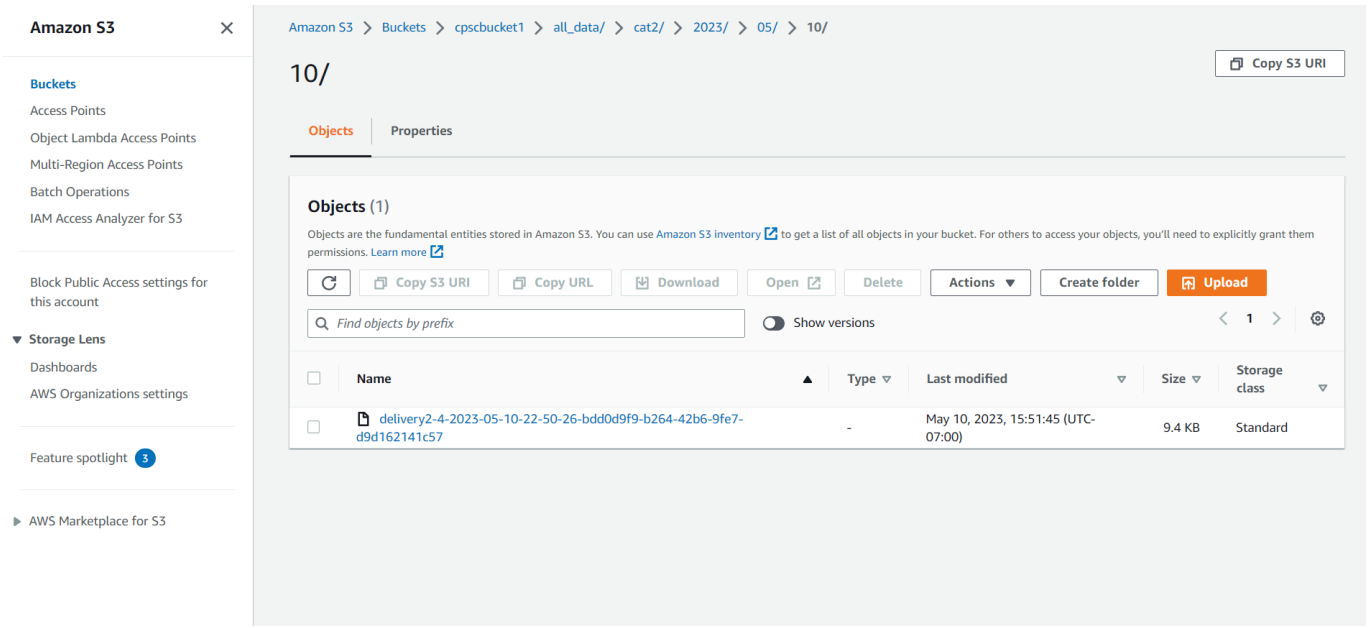

Fig 5.6: AWS S3 Bucket

Create and configure AWS Glue

Follow below instructions to setup AWS Glue to create database and tables for the data in S3:

- In the AWS Management Console and open the AWS Glue service.
- Click on “Databases” in the left navigation pane.
- Click on “Add database” to create a new database.
- Provide a name for the database and click on “Create” to create the database.
- Once the database is created, click on the database name to open it.
- In the database view, click on “Tables” in the left navigation pane.
- Click on “Add tables” to create a new table.
- Click on “Crawlers” in the left navigation pane.
- Click on “Add crawler” to create a new crawler.
- Provide a name for the crawler and choose the data store where the S3 data is located.
- Specify the S3 path where the data is stored, including the bucket and prefix.
- Configure the crawler settings, such as the IAM role, frequency, and output location for the crawler results.

- Review and edit any additional settings, such as metadata options or inclusion/exclusion patterns.
- Run the crawler to scan the S3 data and create the table schema based on the discovered data.
- Once the crawler has completed, go back to the database view and refresh the page. You should see the newly created tables listed under the database.
- Click on a table to view its details and schema.

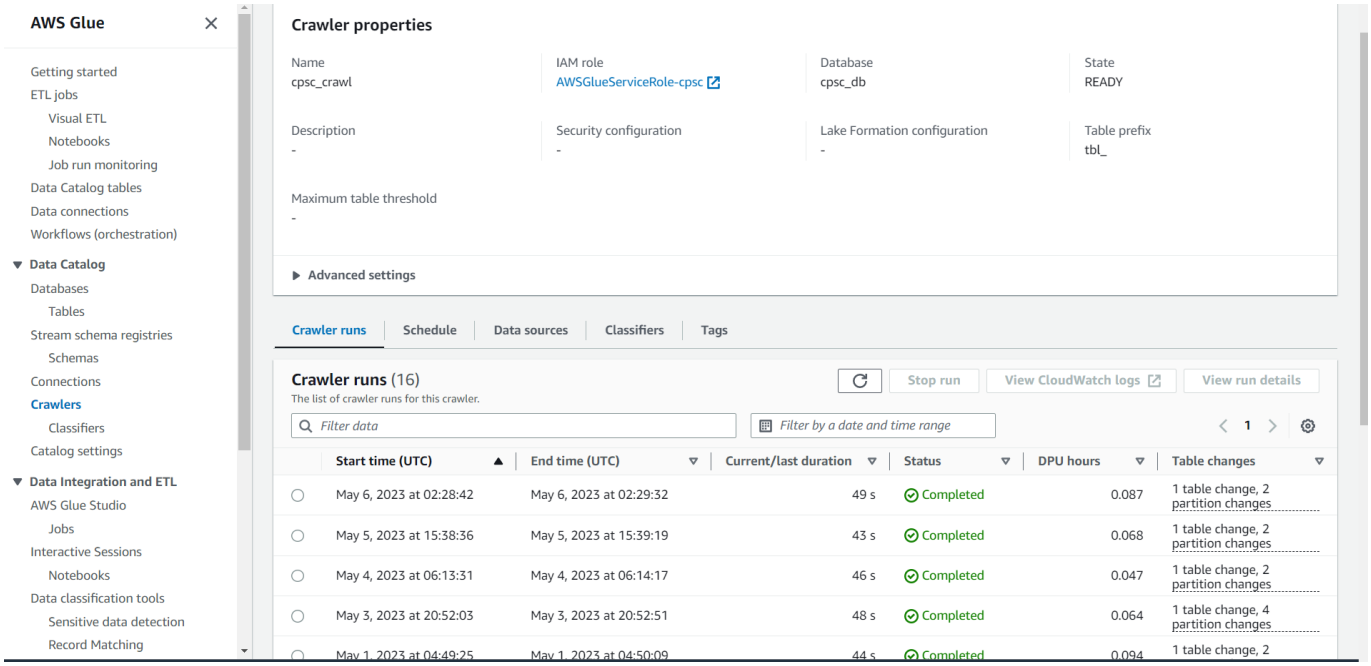

Fig 5.7: AWS Glue

Configure Amazon Athena

Follow below steps to query data using Athena:

- In the Athena console, ensure that you are in the correct AWS region by checking the region selector in the top right corner.
- Click on “Query Editor” in the left navigation pane to open the query editor interface.
- In the query editor, select the appropriate database from the “Database” dropdown. Choose the database that contains the data you want to query.
- Run “SELECT * FROM tble_all_data; to see all the records in the table. (tble_all_data is our table name)

Connect AWS Athena to Tableau and Visualize data

Follow below steps to connect Athena to Tableau:

- Sign into the Tableau Desktop application.
- In Tableau Desktop, click on “Connect to Data” to open the data connection options.
- Search for and select “Amazon Athena” from the list of available connectors.
- In the Amazon Athena connection dialog, enter the necessary connection details:
    1. Server: Enter the AWS region-specific Athena endpoint URL. For example, **https://athena.<region>.amazonaws.com**.
    2. Port: Leave this field empty.
    3. Authentication type: Select “Username and password”.
    4. Username: Enter your AWS access key ID or IAM username.
    5. Password: Enter your AWS secret access key or the corresponding IAM user’s password.
- Click on “Sign In” to establish the connection with AWS Athena.
- Once connected, you will see a list of databases available in Athena. Select the desired database that contains the records you want to view.

6. Steps to run the application

- Download the code from the GitHub which runs through the csv file and sends the data to API Gateway.
- Download Jupyter notebook and install all the necessary packages.
- Go to anaconda command prompt and execute – "python data_generator.py" command to start the code.
- You will see the status code of each record that is being sent to API Gateway.
- Log in to AWS Management Console and check CloudWatch logs to verify if the data is being processed.
- The data will process through API Gateway, EventBridge, Kinesis firehose and it will be saved in AWS S3.
- Go to AWS Glue service and run the crawler to create metadata and records in the created database.
- Now you can go to Athena to query all the data in the database.
- Use tableau to visualize the data by following the steps provided.

7. Test results

Below is the screenshot of query results in AWS Athena

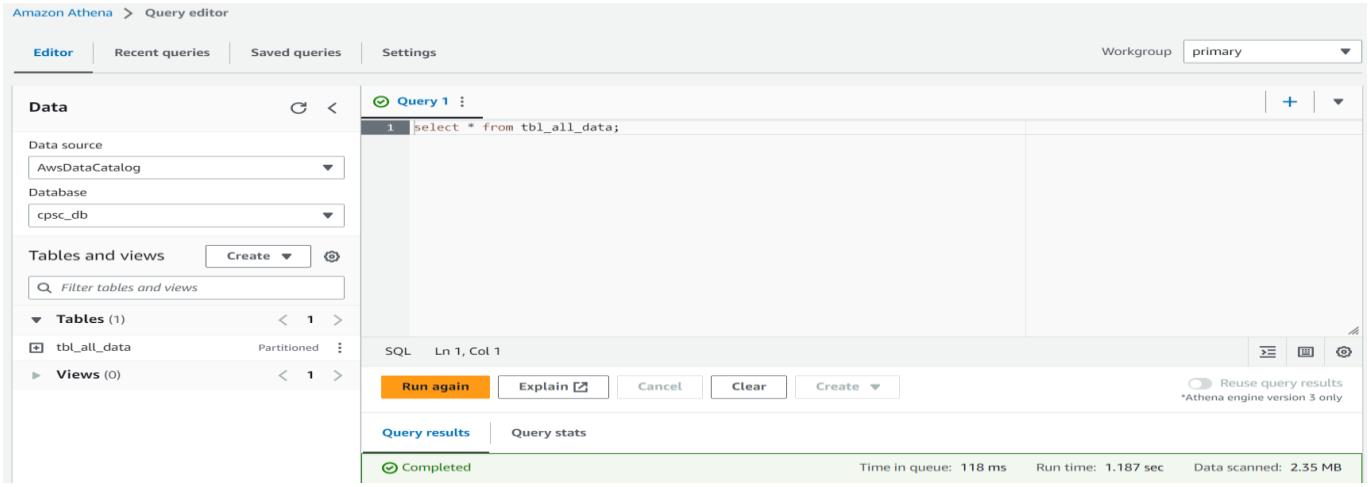

Fig 7.1: Amazon Athena

Below is the screenshot of data visualization in Tableau

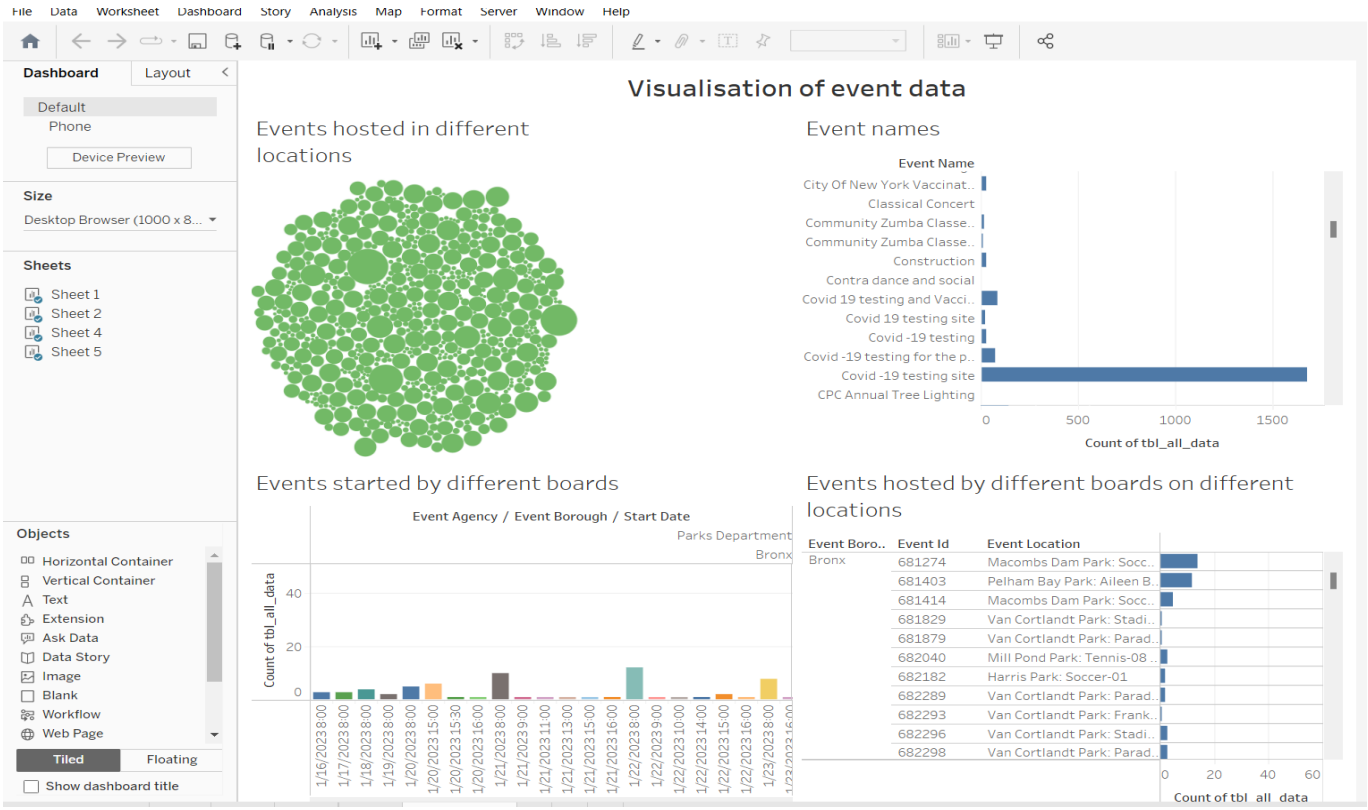

Fig 7.2: Tableau Dashboard

Conclusion

In conclusion, the AWS project that incorporates Amazon API Gateway, EventBridge, Kinesis Firehose, CloudWatch Logs, AWS S3, Glue, Athena, and Tableau supplies a robust and comprehensive solution for data ingestion, processing, analysis, and visualization.

The project begins with Amazon API Gateway, which serves as the entry point for receiving and managing API requests. It enables easy integration with various AWS services and allows for the secure and controlled exposure of APIs to external consumers.

EventBridge plays a pivotal role in the project by supplying a highly scalable and reliable event-driven architecture. It eases the routing and transformation of events from API Gateway to other AWS services, allowing for real-time data processing and analytics.

Kinesis Firehose serves as a powerful data ingestion tool that efficiently collects and prepares streaming data from EventBridge for further processing. It allows seamless delivery of data to various destinations, including AWS S3.

CloudWatch Logs enables centralized logging and monitoring of the entire system. It captures and stores logs generated by different AWS services, supplying visibility into system behavior, troubleshooting capabilities, and real-time alerts through CloudWatch Alarms.

AWS S3 acts as a durable and scalable storage solution for the project. It securely stores data ingested from Kinesis Firehose, allowing for long-term retention and cost-effective analysis.

Glue, the AWS data catalog and ETL service, plays a crucial role in preparing the data for analysis. It automatically discovers and catalogs data in S3, performs data transformations, and creates optimized data sets for analysis in Athena.

Athena enables interactive and serverless querying of data stored in S3. It allows users to run SQL queries on the data cataloged by Glue, providing quick and ad-hoc analysis capabilities without the need for managing infrastructure.

Lastly, Tableau serves as the data visualization and business intelligence tool, seamlessly connecting to Athena for retrieving and visualizing data insights. Tableau empowers users to create interactive dashboards, reports, and visualizations, enabling data-driven decision-making and enhancing data exploration capabilities.

In summary, this AWS project uses a powerful combination of services to set up an end-to-end data pipeline, from data ingestion to analysis and visualization. By using Amazon API Gateway, EventBridge, Kinesis Firehose, CloudWatch Logs, AWS S3, Glue, Athena, and Tableau, organizations can effectively process, analyze, and gain valuable insights from their data, leading to improved decision-making, enhanced operational efficiency, and better business outcomes.

References

[1] Gramm, J. (2019). *AWS: the complete guide from beginners to advanced for Amazon Web Services*.

[2] Wadia, Y. (2019). *Implementing AWS: design, build, and manage your infrastructure : leverage AWS features to build highly secure, fault-tolerant and scalable cloud environments*. Packt.

[3] Subramanian, S., & Natu, S. (2021). *AWS Certified Machine Learning Study Guide: Specialty (MLS-C01) Exam*. John Wiley & Sons, Incorporated.

[4] Abdoulaye, P. (2021). *Transforming Your Business with AWS: Getting the Most Out of Using AWS to Modernize and Innovate Your Digital Services*. Wiley.

[5] Valaxy Technologies. *AWS Cloud Formation Basics*. Packt Publishing, 2022.