**Project Title**

**BUILDING A VISUALIZATION AND MONITORING DASHBOARD FOR IOT DATA**

***Submitted to***

**University of Central Missouri**

***For partial fulfillment of the degree in***

**Master’s Degree**

**(Computer Science)**

***by***

**Nithusha Reddy Baddam (700699779)**

**Nikesh Sai Sriram Bachina (700699778)**

**Madhavi Ummalanani (700692203)**

**Dinesh Botta (700697014)**

***Under the Guidance of***

**Prof. Dr. Jun Wang**



**Department of Computer Science**

**University of Central Missouri**

**Spring 2020**

**PROJECT DESCRIPTION:**

We are creating a Visualization and Monitoring Dashboard for IoT Data using Amazon Kinesis, Amazon S3 and Amazon Quicksight. The IoT devices are a series of internet-connected kegerators, these devices send sensor outputs like temperature, humidity, flow and sound in a JSON payload to AWS IoT gateway in the form of MQTT messages. We are using a python script to generate the device data into AWS IoT from EC2 instance. Once the data is in IoT, we set AWS IoT rules engine to select data from MQTT messages received and then process it. Which then is sent to Kinesis Firehose delivery stream to combine the real time data/continuous data into batches. As the data is sent to Firehose Delivery stream it is then fed into a Kinesis Analytics application. In the Analytics applications, we process the real time data using SQL queries to aggregate minimum and maximum temperature values from the sensors per minute. The processed data from these queries is then fed into two Firehose delivery streams, they batch the data into JSON files every minute and store it in S3 buckets. As the data is available in S3 we can load it in Amazon QuickSight to create a monitoring dashboard. The dashboards help to check if the devices are over-heating or cooling down during use, monitor every device’s parameters such as temperature, sound levels, humidity, and the time stamp. We created business dashboards with different type of visualizations that can be shared with users in a secure manner.

# AWS SERVICES USED

# AWS IoT:

AWS IoT is a cloud platform that lets the devices that are connected to IoT to interact easily and securely with any cloud applications and other devices. As there are many devices used in every organization there is a need to connect them, store and analyze device data, AWS IoT provides a solution for this. IoT can process huge number of messages and then fed them to other AWS services and to other devices in a secure manner. AWS IoT is the only cloud vendor to bring together data management and rich analytics in easy to use services designed for noisy IoT data.

# AWS EC2

Amazon EC2 provides virtual computing environment for the users allowing them to use web service interfaces. The users can choose from a variety of operating systems, install any application, manage network’s access permissions, and can choose any number of instances based on the requirement.

# AWS Kinesis Firehose

Firehose is the easiest way to load streaming data continuously into AWS from different data sources and then store and process that data into AWS data storage services like S3, Redshift and Splunk. It doesn’t require continuous monitoring as it is a fully managed service which can automatically scale to match the throughput of the incoming data.

# AWS Kinesis Analytics

Amazon Kinesis Data Analytics is the easiest way to analyze streaming data, build insights which helps the business needs to be taken care of in real time. Analytics provides interactive SQL editor to process streaming data coming from IoT devices in real time and extract specific components to perform real-time ETL on it.

# AWS S3

Amazon Simple Storage Service is an object storage service that is highly scalable and secure. It provides management features that are easy to use so that one can organize data and configure finely tuned access controls to meet specific requirements.

# AWS Quicksight

Amazon QuickSight is a fast, cloud-based BI (Business Intelligence) service that makes it easy to build dashboards, ad-hoc analysis and create insights which can then be accessed from any device, and embedded into applications, portals, and websites. It is a fully managed service that allows us to easily create and publish interactive dashboards that include ML Insights.

**ARCHITECTURE**

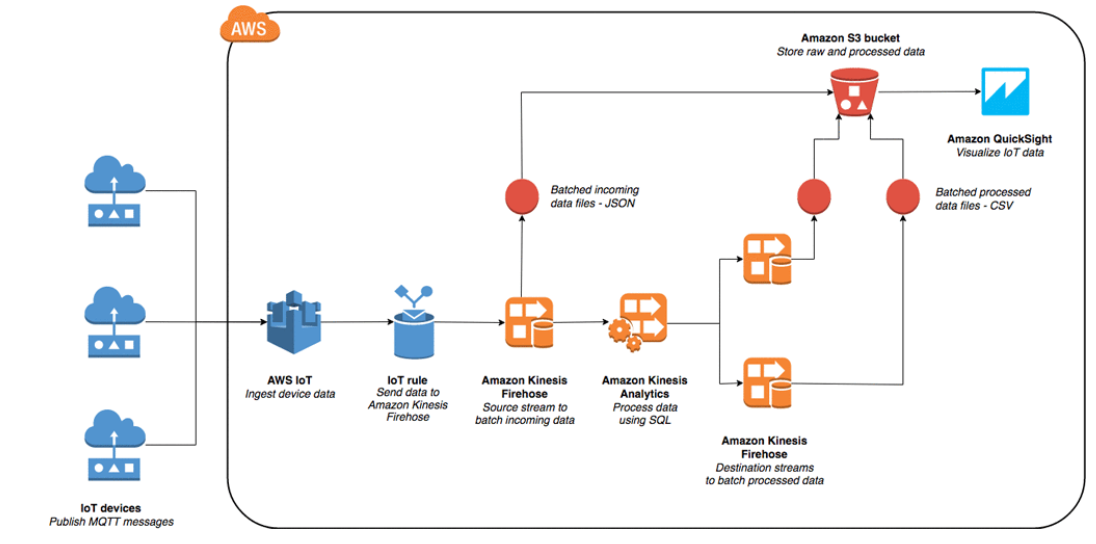


Figure 1: Architecture

**IMPLEMENTATION**

**Step 1. Setting up AWS IoT data source**

First generate sample device data using a python script. There are two ways to do it, one is to run the script on local machine and publish it to AWS IoT and the other is to launch an AWS EC2 instance, set the role as root user and run the script. Choose appropriate IAM role while launching the instance so that it allows to access AWS IoT. Running the Python script generates AWS IoT messages from multiple SBS devices. We need access to AWS CLI credentials and install pip and boto3 on the instance running the script.

# Script to generate IoT device data

import json

import random

import datetime

import boto3

import time

deviceNames = ['SBS01', 'SBS02', 'SBS03', 'SBS04', 'SBS05']

iot = boto3.client('iot-data');

# generate Flow values

def getFlowValues():

data = {}

data['deviceValue'] = random.randint(60, 100)

data['deviceParameter'] = 'Flow'

data['deviceId'] = random.choice(deviceNames)

data['dateTime'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

return data

# generate Temperature values

def getTemperatureValues():

data = {}

data['deviceValue'] = random.randint(15, 35)

data['deviceParameter'] = 'Temperature'

data['deviceId'] = random.choice(deviceNames)

data['dateTime'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

return data

# generate Humidity values

def getHumidityValues():

data = {}

data['deviceValue'] = random.randint(50, 90)

data['deviceParameter'] = 'Humidity'

data['deviceId'] = random.choice(deviceNames)

data['dateTime'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

return data

# generate Sound values

def getSoundValues():

data = {}

data['deviceValue'] = random.randint(100, 140)

data['deviceParameter'] = 'Sound'

data['deviceId'] = random.choice(deviceNames)

data['dateTime'] = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

return data

# Generate each parameter's data input in varying proportions

while True:

time.sleep(1)

rnd = random.random()

if (0 <= rnd < 0.20):

data = json.dumps(getFlowValues())

print(data)

response = iot.publish(

topic='/sbs/devicedata/flow',

payload=data

)

elif (0.20<= rnd < 0.55):

data = json.dumps(getTemperatureValues())

print(data)

response = iot.publish(

topic='/sbs/devicedata/temperature',

payload=data

)

elif (0.55<= rnd < 0.70):

data = json.dumps(getHumidityValues())

print(data)

response = iot.publish(

topic='/sbs/devicedata/humidity',

payload=data

)

else:

data = json.dumps(getSoundValues())

print(data)

response = iot.publish(

topic='/sbs/devicedata/sound',

payload=data

)

A screenshot of a cell phone

Description automatically generated

Figure 2:EC2 instance created

Commands to install pip, boto3 and run the script:

sudo yum install python-pip

python -m pip install --user boto3

python IOT\_Data.py

**Step 2. Create three Firehose delivery streams**

We require three Firehose delivery streams for our project: one to batch raw data from AWS IoT, and two to batch output device data and aggregated data from Analytics.

* Choose Kinesis from the Services menu and select Firehose.
* Create three Firehose delivery streams
* Keep all default settings except change the Buffer interval to 60 seconds.

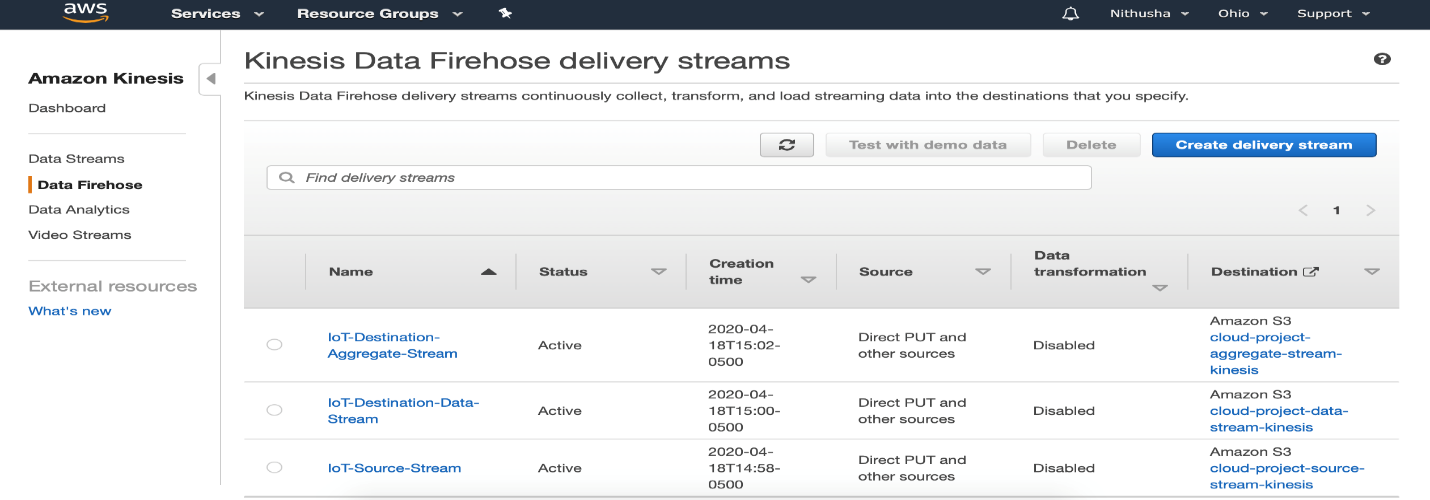


Figure 3:Kinesis Delivery Streams

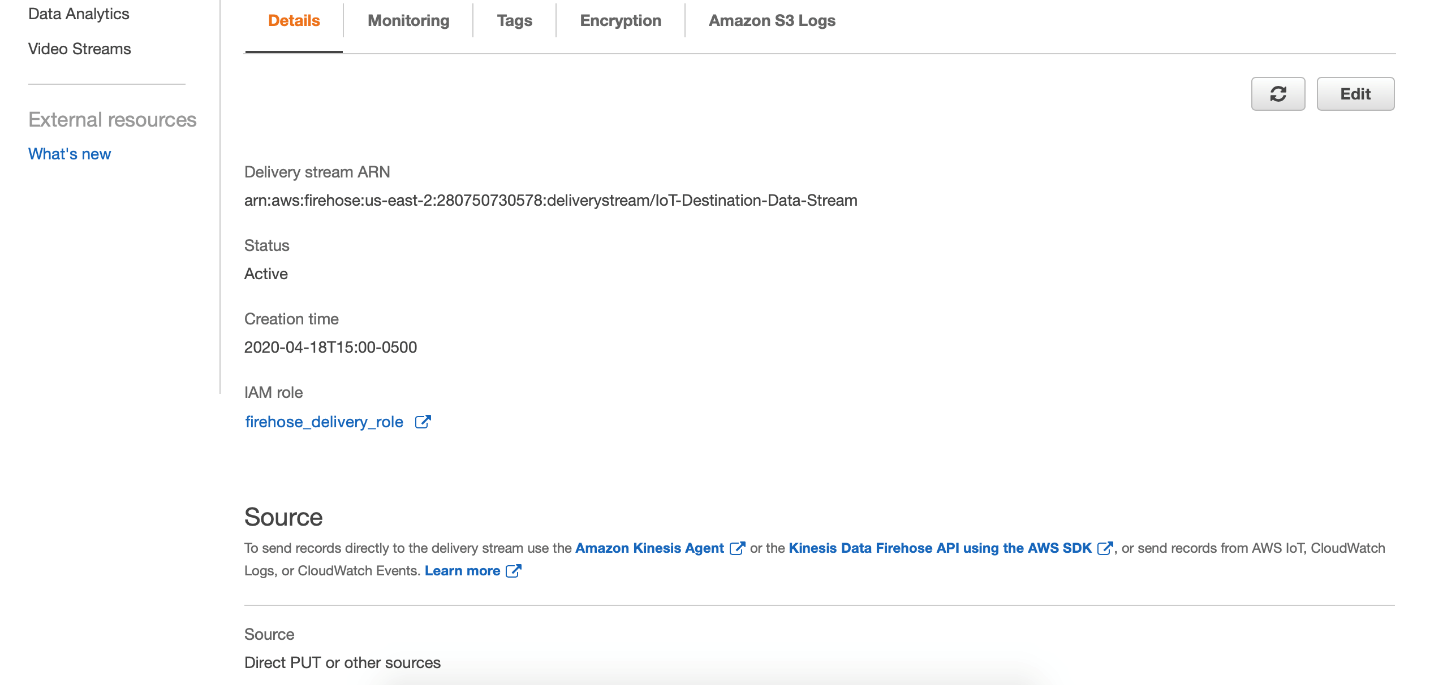


Figure 4:Kinesis Analytics Project

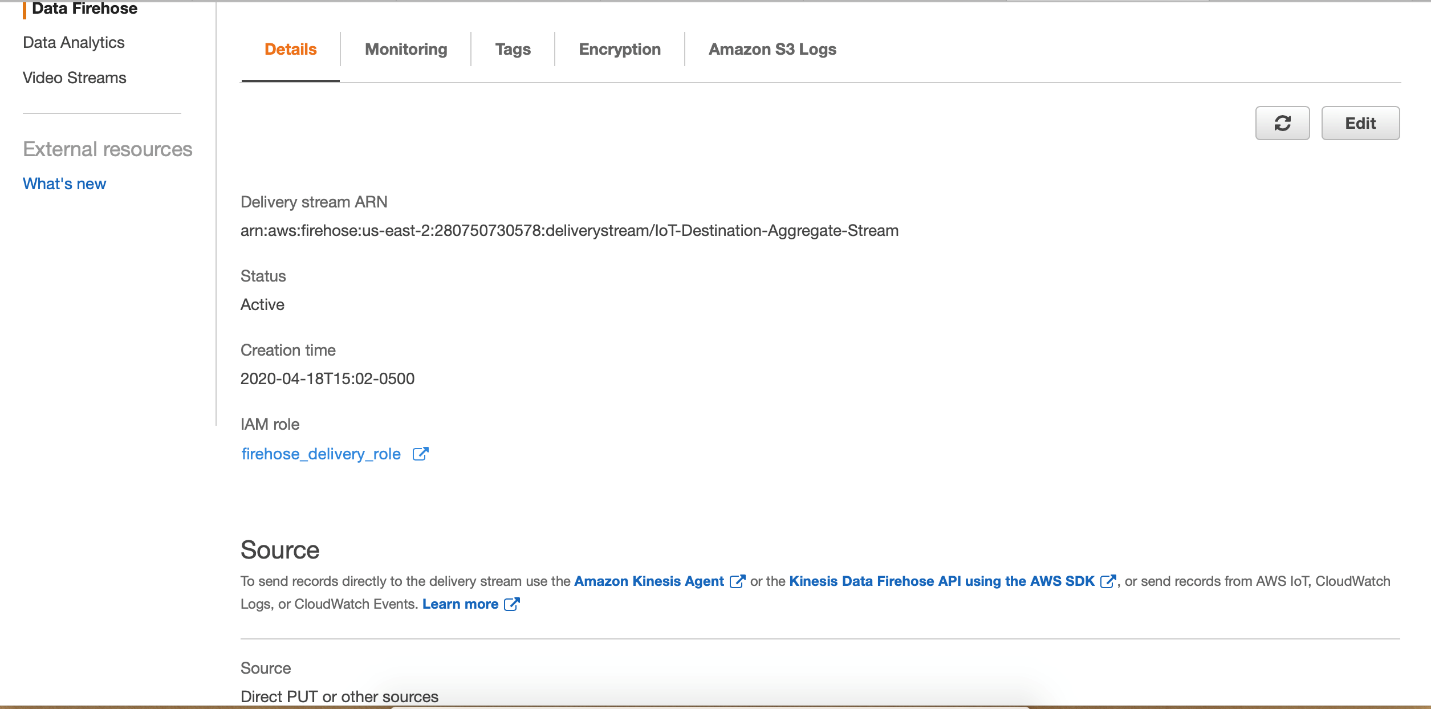
f

Figure 5: Kinesis Analysis Project

**Step 3. Set up AWS IoT to receive and forward incoming data**

* Choose AWS IoT from the services menu.
* Select Act available in the menu on left side of the page and Create a rule.
* Field Values are shown in below screenshot

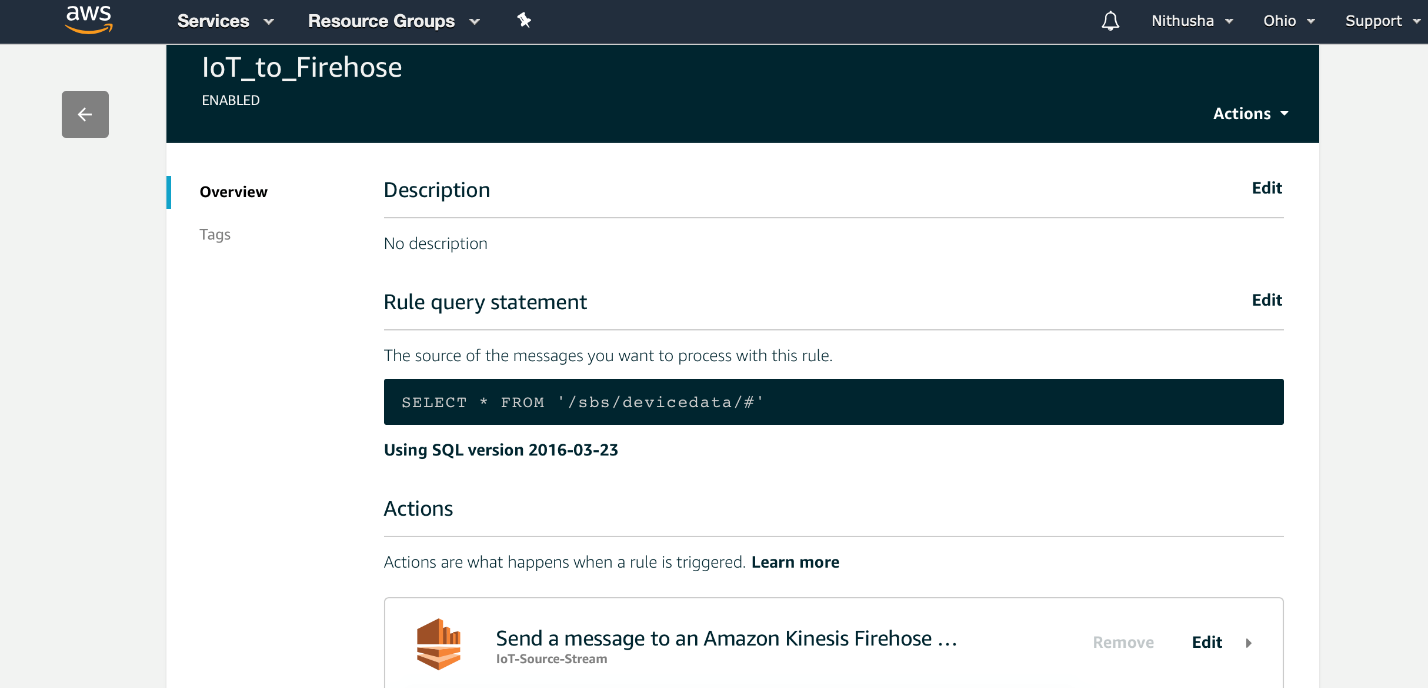


Figure 6: IOT\_Role\_Setup

**Step 4: Create an Analytics application to process data**

* Created a new application in Kinesis Analytics.
* Selected IoT-Source-Stream as source.
* Selected ‘let the console create/update an IAM role to use with Kinesis Analytic’ for Access to chosen resources section.
* By clicking Discover Schema button, a table with device parameters, device values, device ID and timestamp will be populated. Selected Save and continue.
* Go to SQL editor and ran the sql queries to generate two output streams.

DESTINATION\_SQL\_DATA\_STREAM - contains the device ID, parameter, value, and the time stamp from the incoming stream.

DESTINATION\_SQL\_AGGREGATE\_STREAM - aggregates the maximum and minimum values of temperatures from the sensors over a one-minute period from the incoming data.

CREATE OR REPLACE STREAM "DESTINATION\_SQL\_BASIC\_STREAM" (dateTime TIMESTAMP, deviceId VARCHAR(8), deviceParameter VARCHAR(16), deviceValue INTEGER);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_1" AS INSERT INTO "DESTINATION\_SQL\_BASIC\_STREAM"

SELECT STREAM "dateTime", "deviceId", "deviceParameter", "deviceValue" FROM "SOURCE\_SQL\_STREAM\_001";

CREATE OR REPLACE STREAM "DESTINATION\_SQL\_AGGREGATE\_STREAM" (dateTime TIMESTAMP, highestTemp SMALLINT, lowestTemp SMALLINT);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_2" AS INSERT INTO "DESTINATION\_SQL\_AGGREGATE\_STREAM"

SELECT STREAM FLOOR("SOURCE\_SQL\_STREAM\_001".ROWTIME TO MINUTE) AS "dateTime", MAX("deviceValue") AS "highestTemp", MIN("deviceValue") AS "lowestTemp" FROM "SOURCE\_SQL\_STREAM\_001" WHERE "deviceParameter"='Temperature' GROUP BY FLOOR("SOURCE\_SQL\_STREAM\_001".ROWTIME TO MINUTE);

CREATE OR REPLACE STREAM "DESTINATION\_SQL\_BASIC\_STREAM" (dateTime TIMESTAMP, deviceId VARCHAR(8), deviceParameter VARCHAR(16), deviceValue INTEGER);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_1" AS INSERT INTO "DESTINATION\_SQL\_BASIC\_STREAM"

SELECT STREAM "dateTime", "deviceId", "deviceParameter", "deviceValue" FROM "SOURCE\_SQL\_STREAM\_001";

CREATE OR REPLACE STREAM "DESTINATION\_SQL\_AGGREGATE\_STREAM" (dateTime TIMESTAMP, highestTemp SMALLINT, lowestTemp SMALLINT);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_2" AS INSERT INTO "DESTINATION\_SQL\_AGGREGATE\_STREAM"

SELECT STREAM FLOOR("SOURCE\_SQL\_STREAM\_001".ROWTIME TO MINUTE) AS "dateTime", MAX("deviceValue") AS "highestTemp", MIN("deviceValue") AS "lowestTemp" FROM "SOURCE\_SQL\_STREAM\_001" WHERE "deviceParameter"='Temperature' GROUP BY FLOOR("SOURCE\_SQL\_STREAM\_001".ROWTIME TO MINUTE);

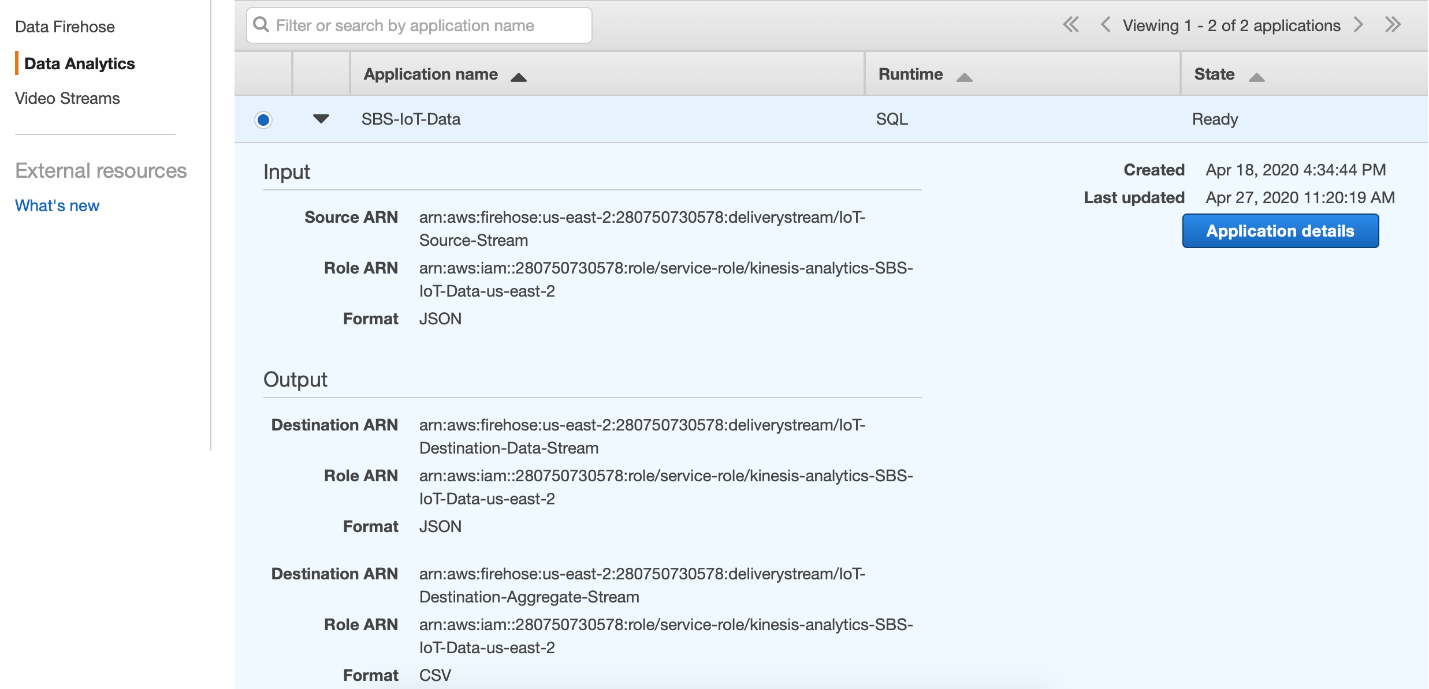


Figure 7: Data\_Aggregate \_Stream

**Step 5: Connect the Analytics application to output Firehose delivery streams**

We created two destinations for two delivery streams that were created in the previous step. We need AWS CLI to set up the Destinations. So, we are using EC2 instance that we created in step 1 to run the following commands to create destinations.

1. For current-application-version-ID, run the following command:

aws kinesisanalytics describe-application --application-name SBS-IoT-Data | grep ApplicationVersionId

1. For ResourceARN, run the following command:

aws firehose describe-delivery-stream --delivery-stream-name IoT-Destination-Data-Stream | grep DeliveryStreamARN

1. For RoleARN, run the following command:

aws kinesisanalytics describe-application --application-name SBS-IoT-Data | grep RoleARN

1. The AWS CLI command to create a new output Firehose delivery stream:

aws kinesisanalytics add-application-output --application-name SBS\_IoT\_Data --current-application-version-id 4 --application-output 'Name=DESTINATION\_SQL\_BASIC\_STREAM,KinesisFirehoseOutput={ResourceARN=arn:aws:firehose:us-east-2:280750730578:deliverystream/IoT-Destination-Data-Stream,RoleARN=arn:aws:iam::280750730578:role/service-role/kinesis-analytics-SBS-IoT-Data-us-east-2},DestinationSchema={RecordFormatType=JSON}'

To Create the second destination delivery stream, we followed similar steps as above. Except for changing name to DESTINATION\_SQL\_AGGREGATE\_STREAM and change the delivery stream name to IoT-Destination-Aggregate-Stream in ResourceARN command.

Then Updated the IAM role for Analytics to allow writing to both output streams. The steps followed are

* IAM from the service menu and select Roles.
* Select the appropriate role that we created in step 4 while creating new analytics application.
* Choose Policy, JSON, and Edit.
* Find “Sid”: “WriteOutputFirehose” in the JSON document, go to the “Resource” section and added the ARNs we created in previous step.

The incoming IoT data is now processed by Analytics and delivered, using two output delivery streams, to two separate folders in your S3 bucket.

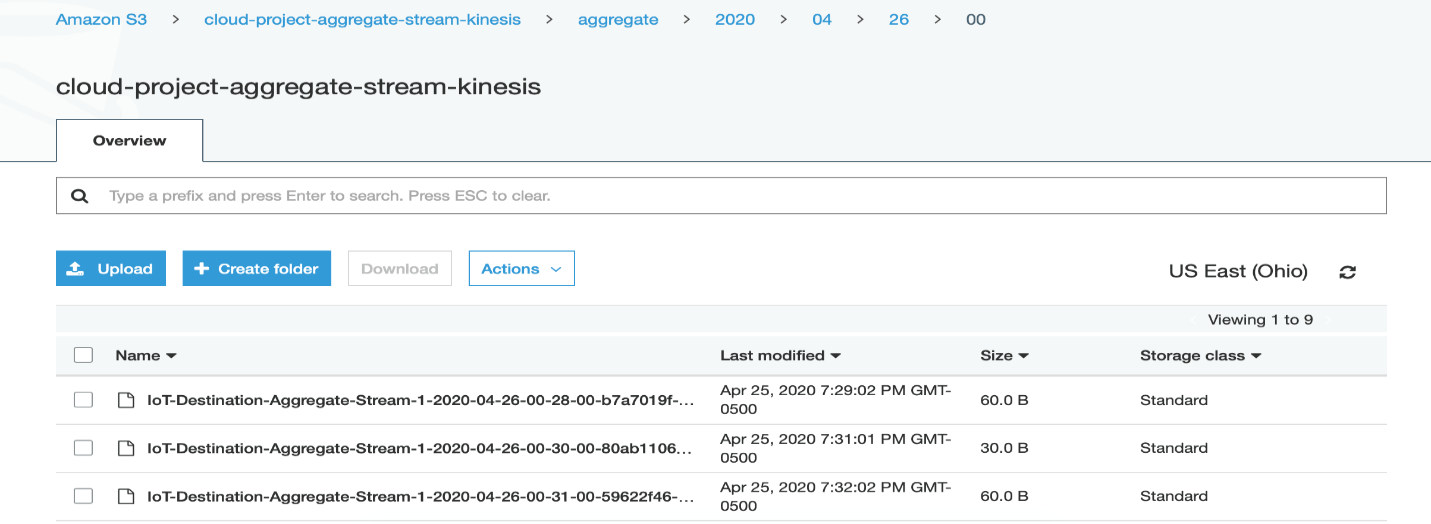


Figure 8 :Data In S3 Buckets

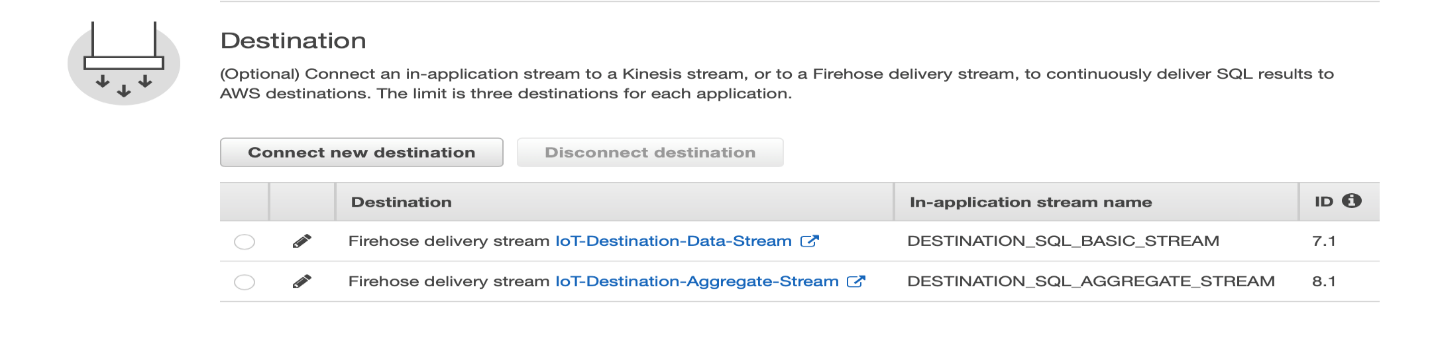


Figure 9:Destination For Delivering SQL Queries to S3

**Step 6: Set up Amazon QuickSight to analyze the data**

To build the dashboards, loaded the processed JSON files from the S3 bucket into Amazon QuickSight.

* In Quicksight added two New data set for the analysis, one general device data and other is aggregated data.
* To import data from S3, choose New Data set - > S3 - > Name - > provided the manifest file which has the s3 bucket locations.

Manifest file for general device data:

**{**

**"fileLocations": [{**

**"URIPrefixes": [**

**"https://s3.amazonaws.com/cloud-project-data-stream-kinesis/data/2020/04/23/01/",**

**"https://s3.amazonaws.com/cloud-project-data-stream-kinesis/data/2020/04/23/03/",**

**"https://s3.amazonaws.com/cloud-project-data-stream-kinesis/data/2020/04/25/01/",**

**"https://s3.amazonaws.com/cloud-project-data-stream-kinesis/data/2020/04/26/00/"**

**]**

**}**

**],**

**"globalUploadSettings": {**

**"format": "TSV",**

**"delimiter": "\t",**

**"containsHeader": "true"**

**}**

**}**

Manifest file for Aggregate data:

**{**

**"fileLocations": [{**

**"URIPrefixes": [**

**"https://s3.amazonaws.com/cloud-project-aggregate-stream-kinesis/aggregate/2020/04/23/01/",**

**"https://s3.amazonaws.com/cloud-project-aggregate-stream-kinesis/aggregate/2020/04/23/03/",**

**"https://s3.amazonaws.com/cloud-project-aggregate-stream-kinesis/aggregate/2020/04/25/00/",**

**"https://s3.amazonaws.com/cloud-project-aggregate-stream-kinesis/aggregate/2020/04/25/01/",**

**"https://s3.amazonaws.com/cloud-project-aggregate-stream-kinesis/aggregate/2020/04/26/00/"**

**]**

**}**

**],**

**"globalUploadSettings": {**

**"format": "TSV",**

**"delimiter": "\t",**

**"containsHeader": "true"**

**}**

**}**

**Data Preview**

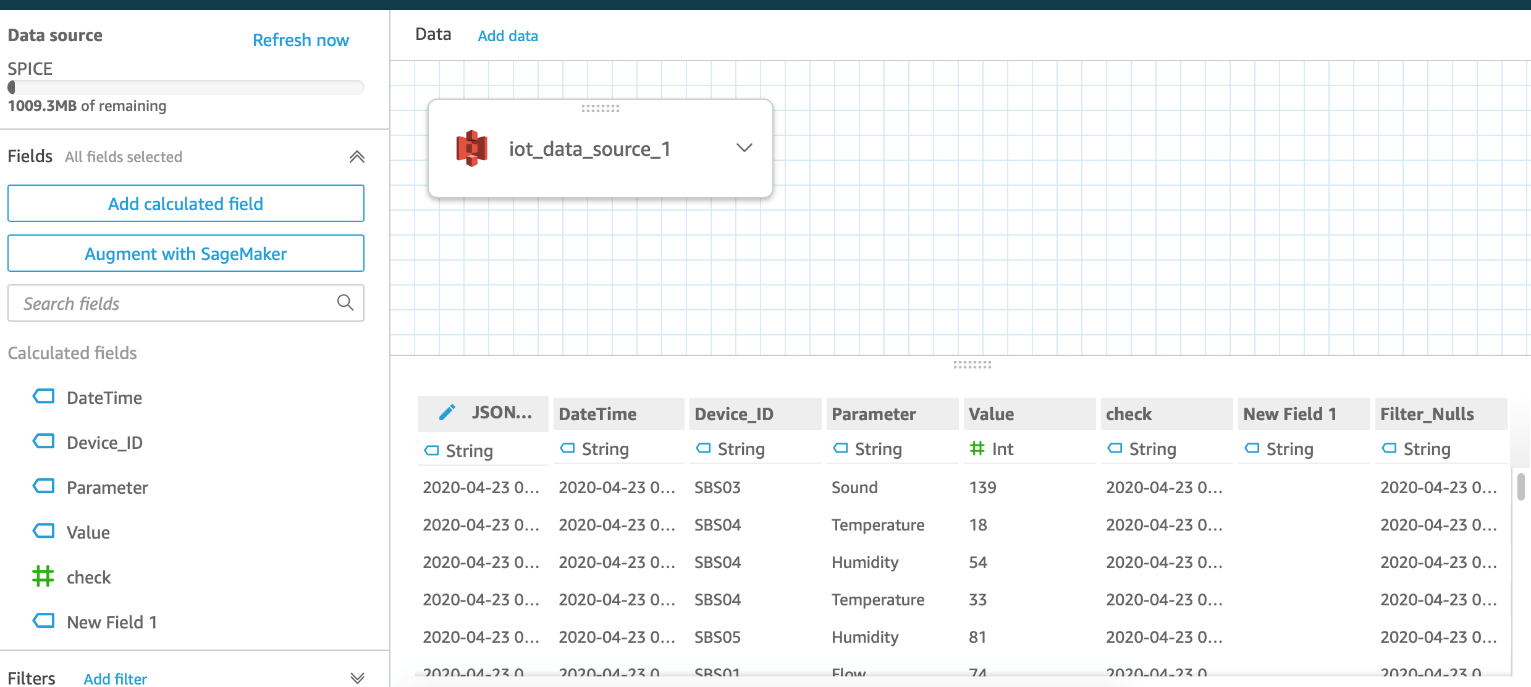


Figure 10:Data Preview In QuickSight

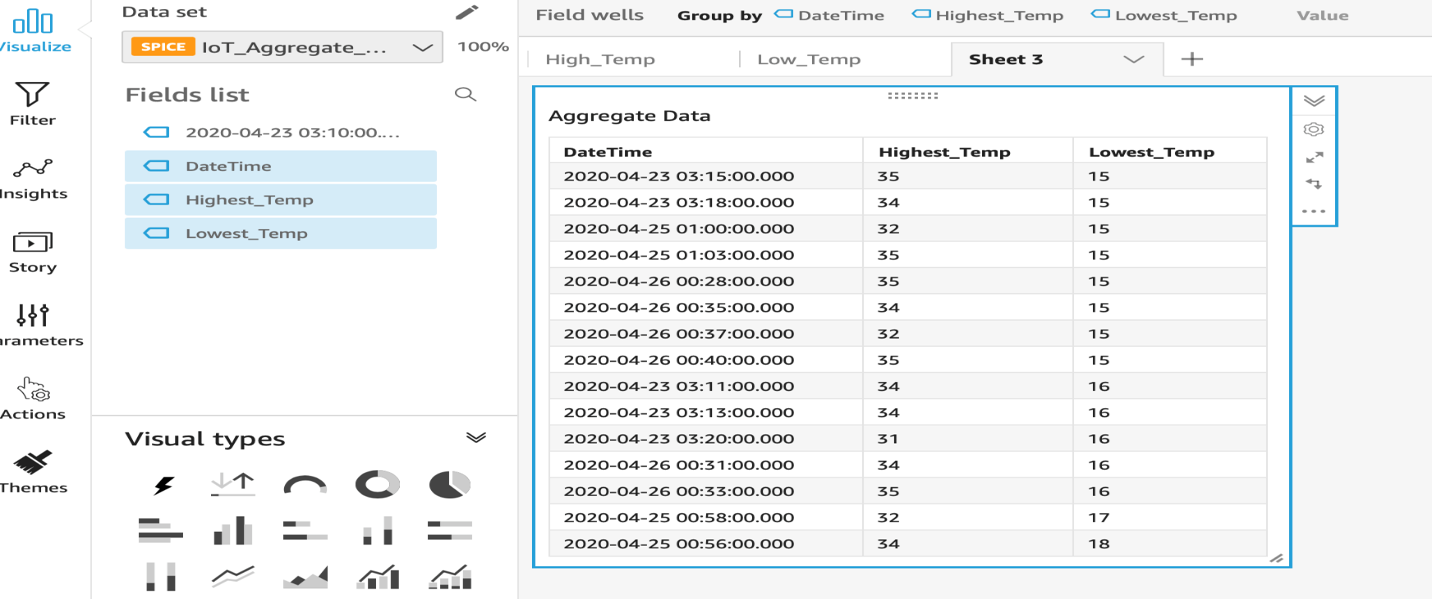


Figure 11: IOT\_Aggregate\_Data Dashboard

**DASHBOARDS**

**Summary Dashboard**

A screenshot of a cell phone

Description automatically generated

Figure 12**:** IOT\_Data\_Analysis\_Summary Dashboard

**Parameter Dashboard**

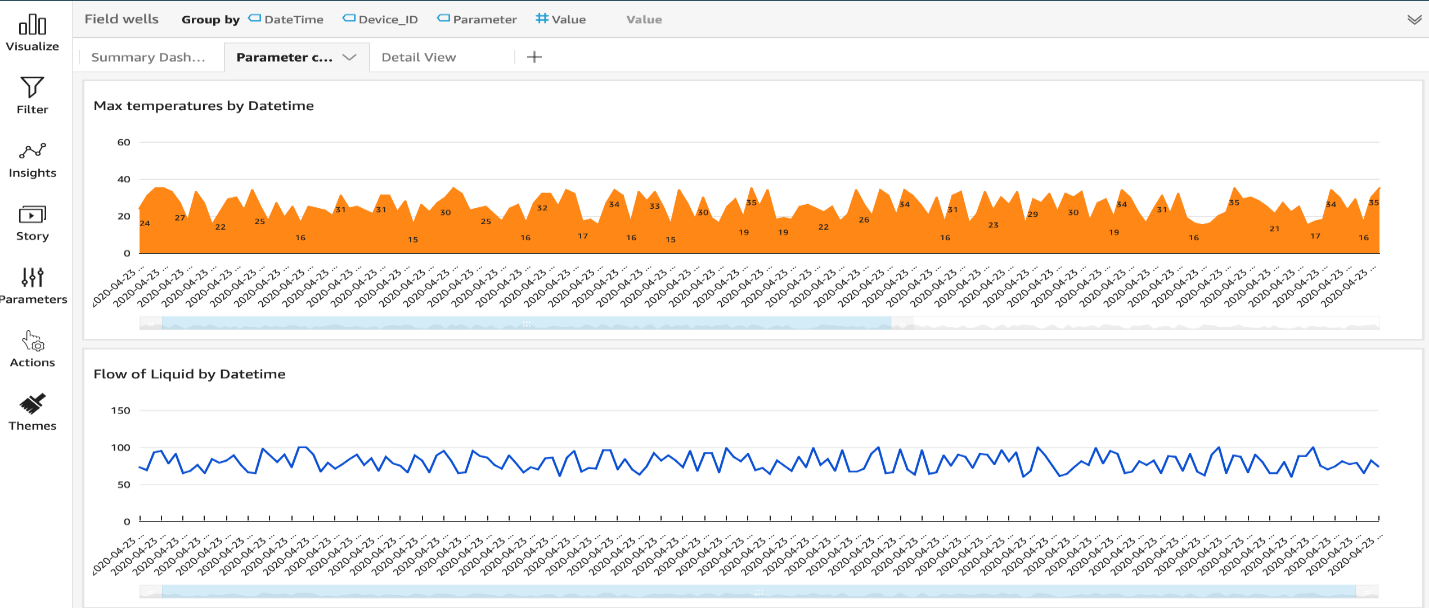


Figure 13: IOT\_Data\_Analysis\_Business Dashboard

**Detail Dashboard**

A screenshot of a social media post

Description automatically generated

Figure14: IOT\_Data\_Analysis\_Detailview

**CONCLUSION**

It is possible to quickly build a simple Analytics application to ingest, process, and visualize IoT data in near real time entirely using AWS managed services. This solution is scalable and reliable and costs a fraction of other business intelligence solutions.