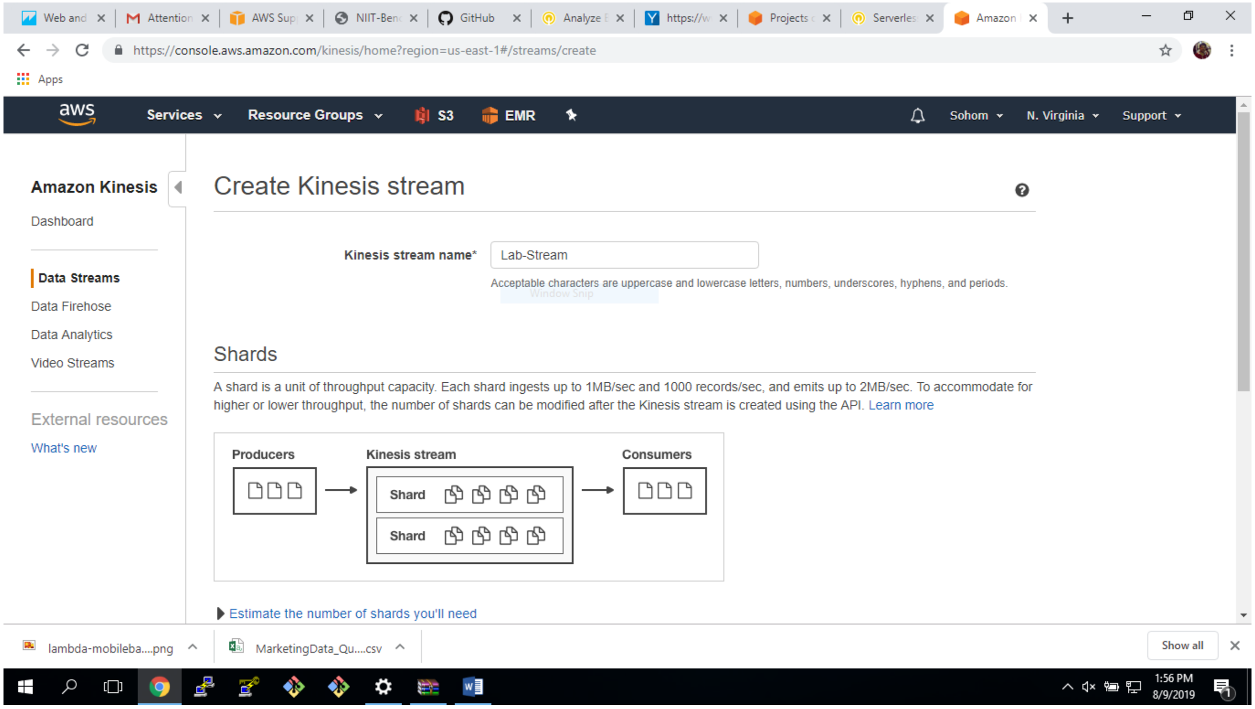
Date: 09.08.2019

Serverless Architectures with Amazon DynamoDB and Amazon Kinesis Streams with AWS Lambda

Created by-
Sohom Bosu Choudhury



https://console.aws.amazon.com/kinesis/home?region=us-east-1#/streams/create

Create Kinesis stream

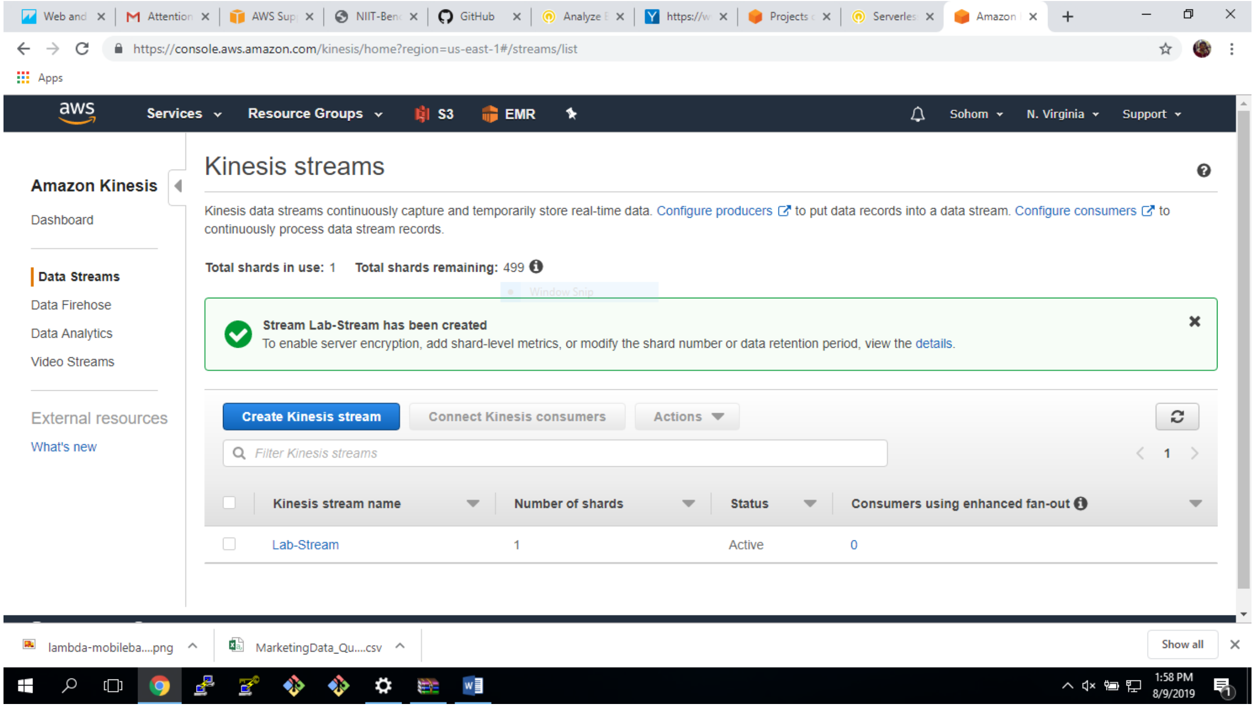
Kinesis stream name* Lab-Stream

Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.

Shards

A shard is a unit of throughput capacity. Each shard ingests up to 1MB/sec and 1000 records/sec, and emits up to 2MB/sec. To accommodate for higher or lower throughput, the number of shards can be modified after the Kinesis stream is created using the API. Learn more

Create Kinesis stream



Kinesis streams

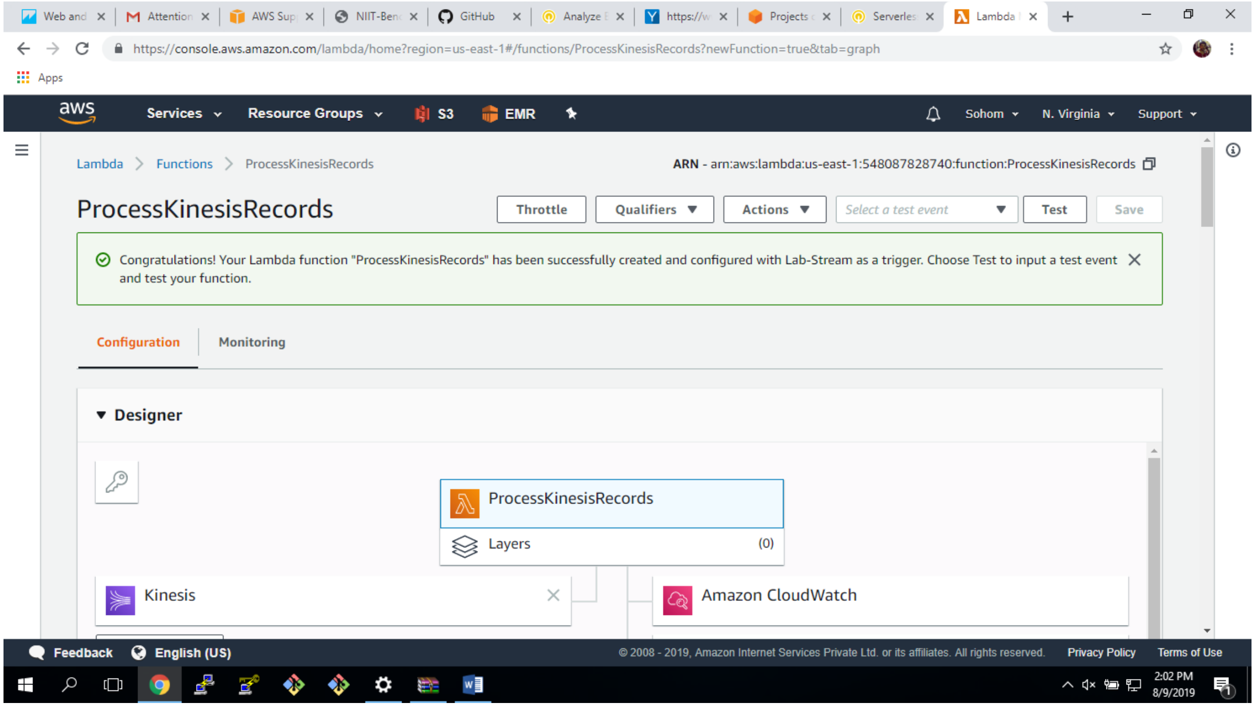
Kinesis data streams continuously capture and temporarily store real-time data. Configure producers to put data records into a data stream. Configure consumers to continuously process data stream records.

Total shards in use: 1 **Total shards remaining:** 499

Stream Lab-Stream has been created
To enable server encryption, add shard-level metrics, or modify the shard number or data retention period, view the details.

Kinesis stream name	Number of shards	Status	Consumers using enhanced fan-out
Lab-Stream	1	Active	0

Kinesis stream is created

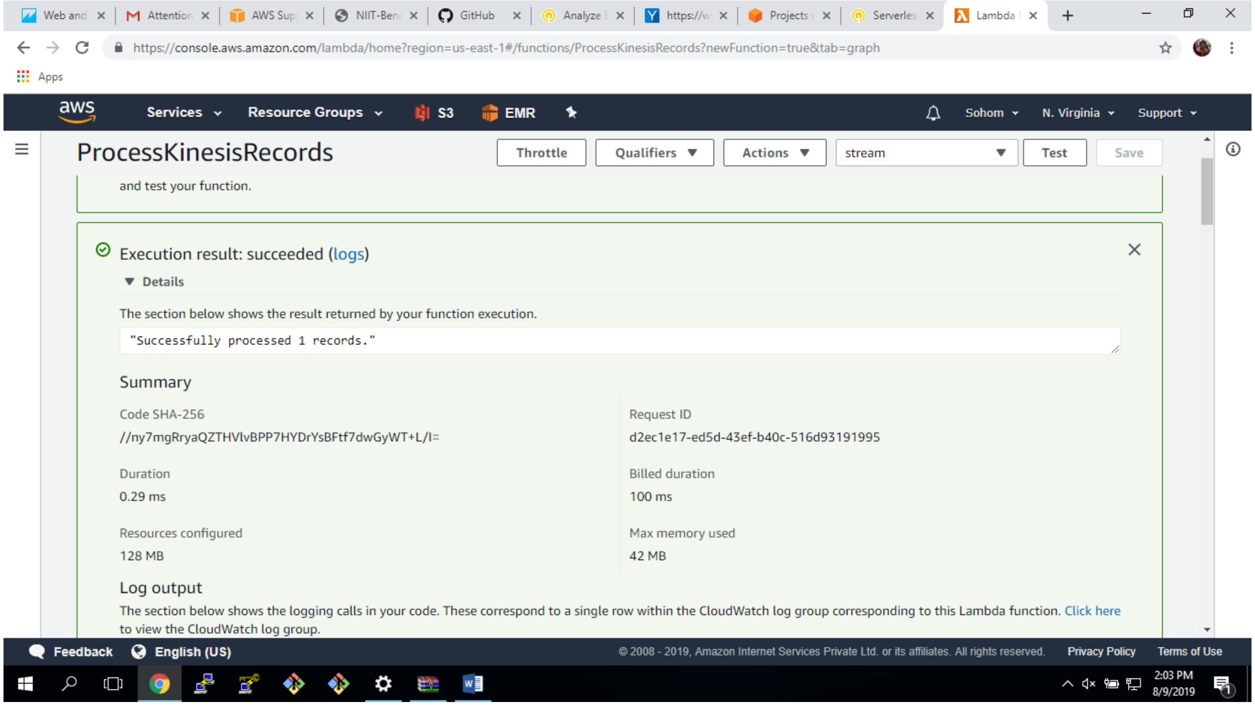


ARN - arn:aws:lambda:us-east-1:548087828740:function:ProcessKinesisRecords

ProcessKinesisRecords

Congratulations! Your Lambda function "ProcessKinesisRecords" has been successfully created and configured with Lab-Stream as a trigger. Choose Test to input a test event and test your function.

Lambda Function is created successfully



ProcessKinesisRecords

and test your function.

Execution result: succeeded (logs)

Details

The section below shows the result returned by your function execution.

```
"Successfully processed 1 records."
```

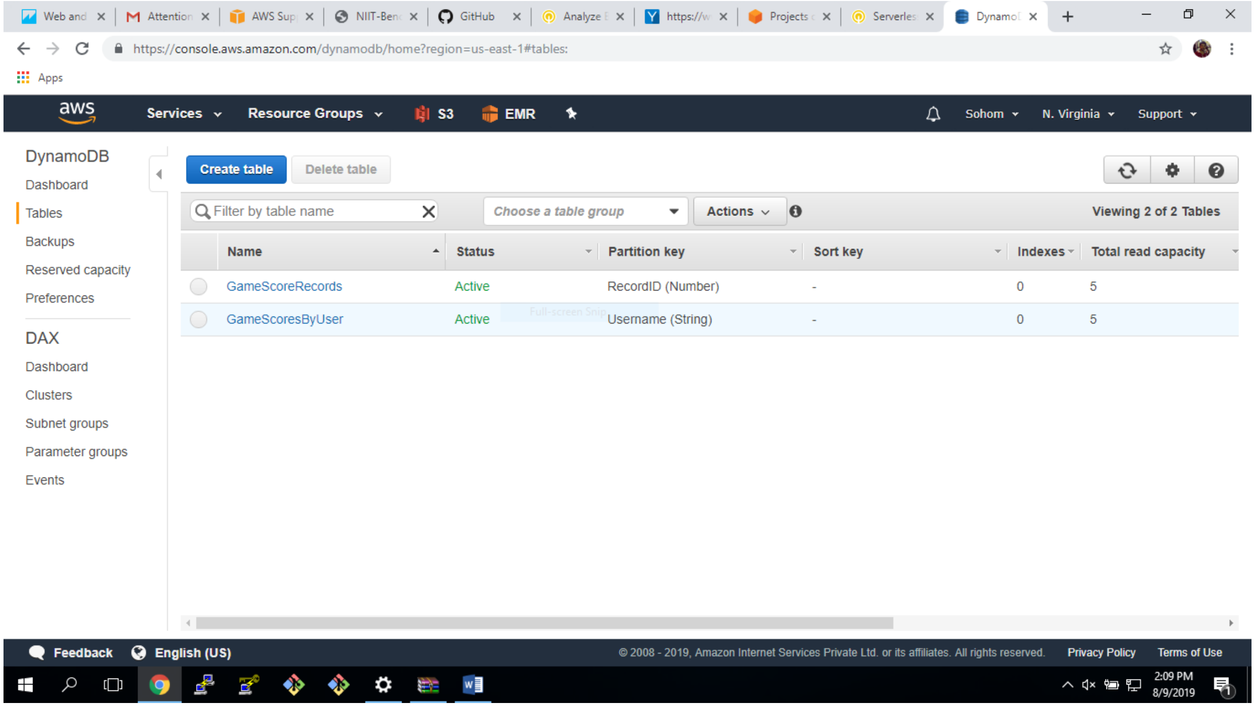
Summary

Code SHA-256 //ny7mgRryaQZTHVlvBPP7HYDrYsBFtf7dwGyWT+L/I=	Request ID d2ec1e17-ed5d-43ef-b40c-516d93191995
Duration 0.29 ms	Billed duration 100 ms
Resources configured 128 MB	Max memory used 42 MB

Log output

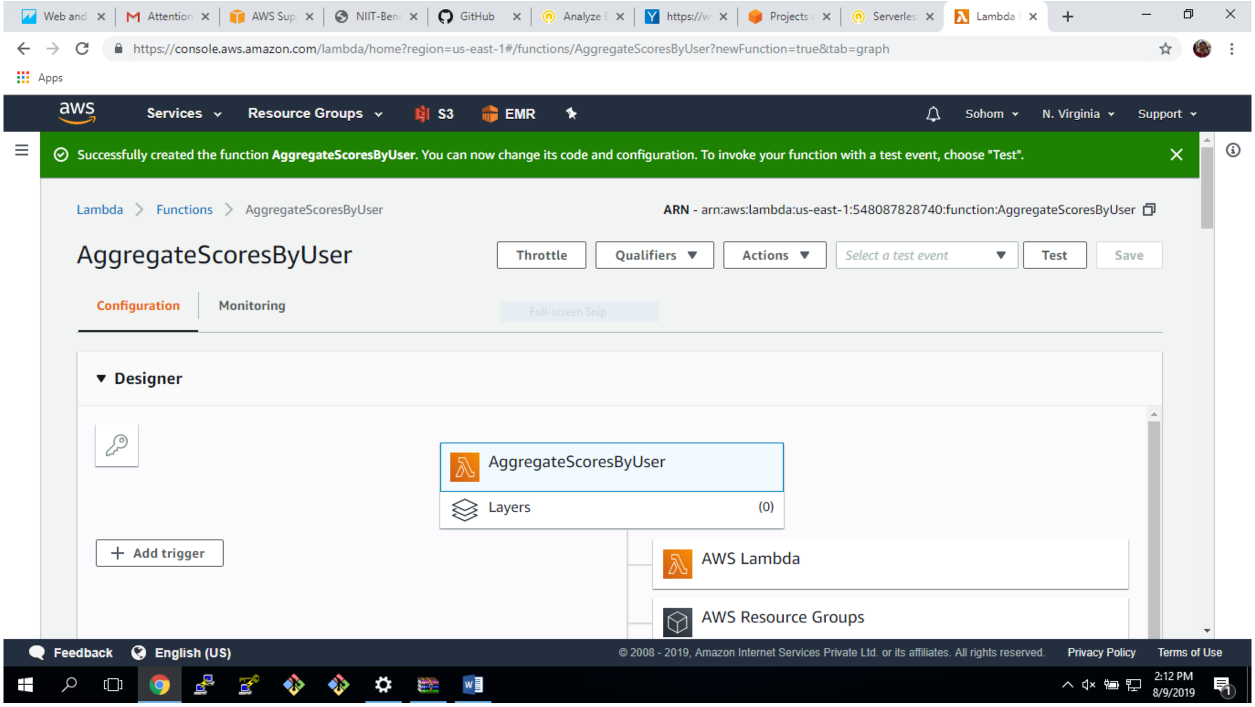
The section below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to this Lambda function. Click here to view the CloudWatch log group.

Tested successfully and Execution result is success



	Name	Status	Partition key	Sort key	Indexes	Total read capacity
○	GameScoreRecords	Active	RecordID (Number)	-	0	5
○	GameScoresByUser	Active	Username (String)	-	0	5

Created 2 database table in DynamoDB named GameScoreRecords and GameScoresByUser



Successfully created the function **AggregateScoresByUser**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > AggregateScoresByUser

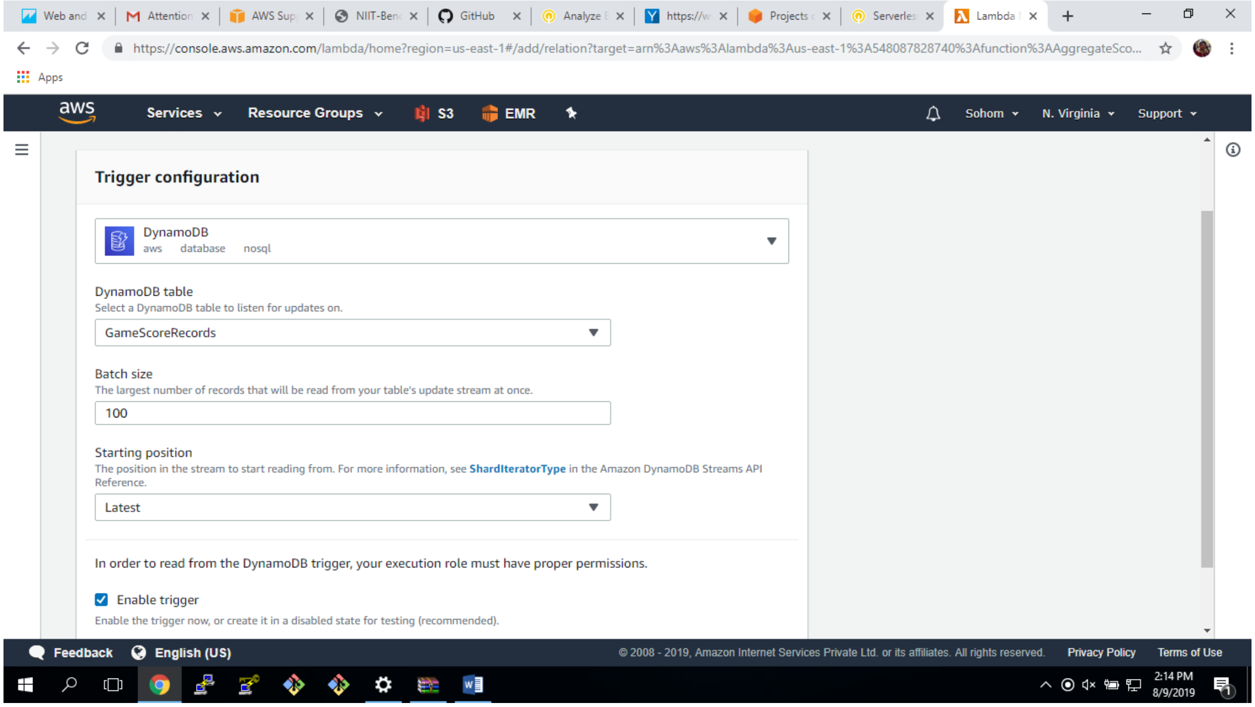
ARN - arn:aws:lambda:us-east-1:548087828740:function:AggregateScoresByUser

AggregateScoresByUser

▼ **Designer**

AggregateScoresByUser

Successfully created the function AggregateScoresByUser


Web and
Attention
AWS Sup
NIIT-Ben
GitHub
Analyze
https://w
Projects
Serverles
Lambda
https://console.aws.amazon.com/lambda/home?region=us-east-1#/add/relation?target=arn%3Aaws%3Alambda%3Aus-east-1%3A548087828740%3Afunction%3AAggregateSco...
Apps
aws
Services
Resource Groups
S3
EMR
Sohom
N. Virginia
Support
Feedback
English (US)
© 2008 - 2019, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
2:14 PM
8/9/2019

Trigger configuration

DynamoDB
aws database nosql

DynamoDB table
Select a DynamoDB table to listen for updates on.
GameScoreRecords

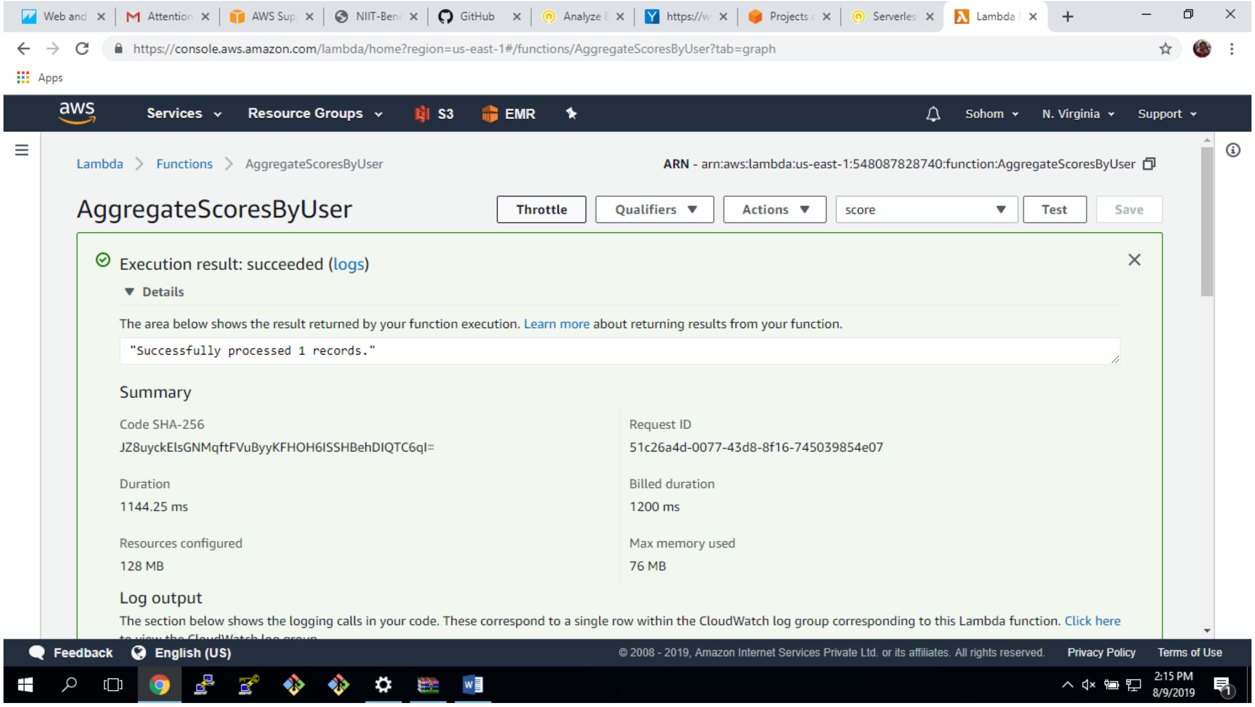
Batch size
The largest number of records that will be read from your table's update stream at once.
100

Starting position
The position in the stream to start reading from. For more information, see **ShardIteratorType** in the Amazon DynamoDB Streams API Reference.
Latest

In order to read from the DynamoDB trigger, your execution role must have proper permissions.

☑ Enable trigger
Enable the trigger now, or create it in a disabled state for testing (recommended).

Configure the trigger



ARN - arn:aws:lambda:us-east-1:548087828740:function:AggregateScoresByUser

AggregateScoresByUser

Execution result: succeeded (logs)

The area below shows the result returned by your function execution. Learn more about returning results from your function.

```
"Successfully processed 1 records."
```

Summary

Code SHA-256
JZ8uyckElsGNMqftFVuByyKFHOH6ISSHBehDIQTC6ql=

Request ID
51c26a4d-0077-43d8-8f16-745039854e07

Duration
1144.25 ms

Billed duration
1200 ms

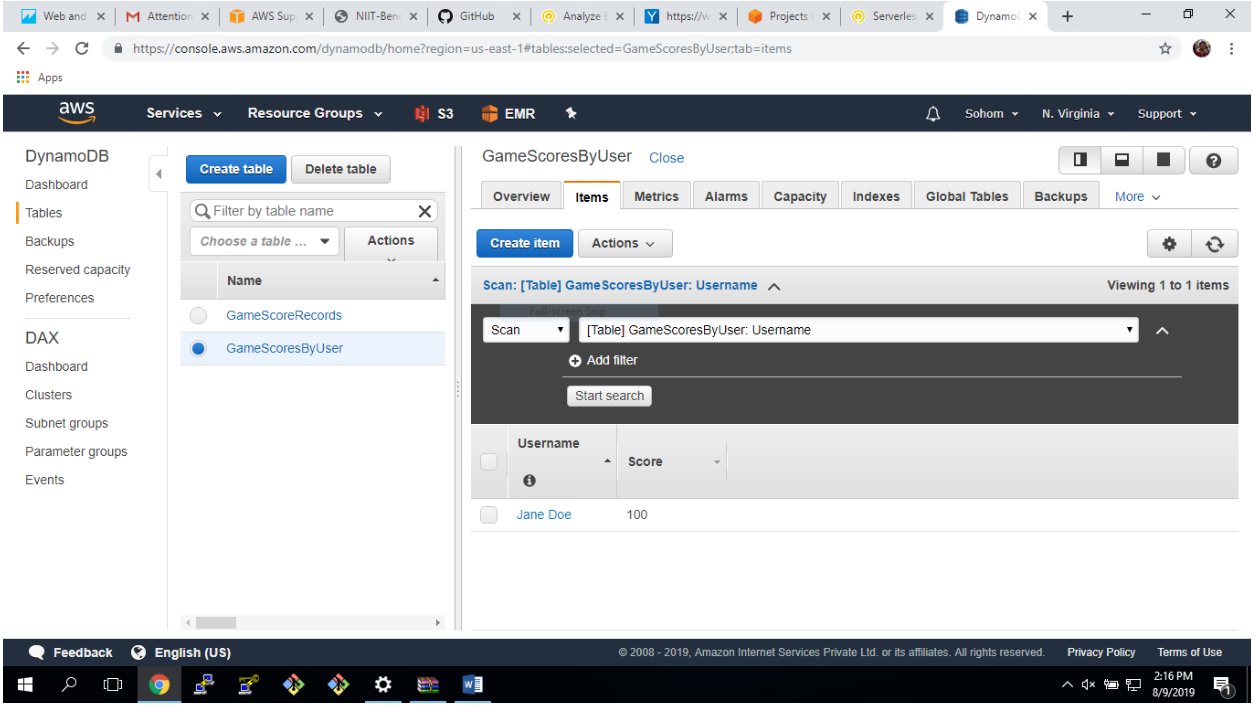
Resources configured
128 MB

Max memory used
76 MB

Log output

The section below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to this Lambda function. Click here to view the CloudWatch log group.

Successfully tested the function AggregateScoresByUser



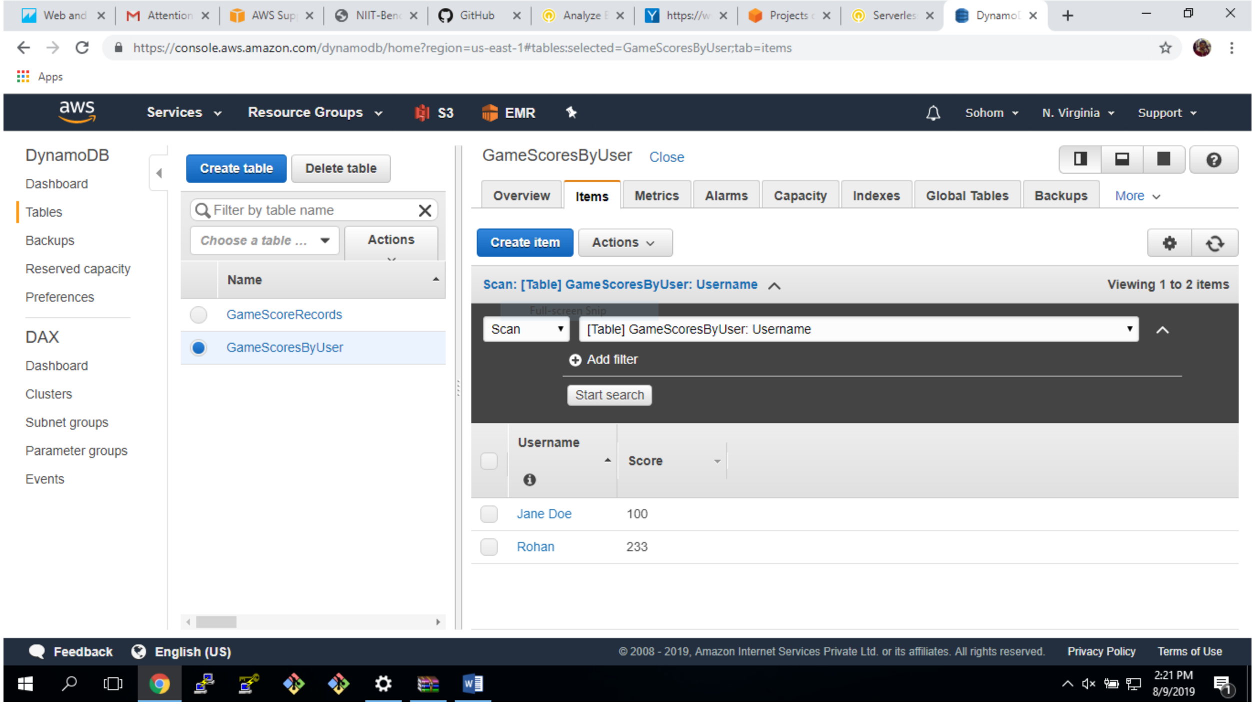
GameScoresByUser

Scan: [Table] GameScoresByUser: Username — Viewing 1 to 1 items

Scan — [Table] GameScoresByUser: Username

Username	Score
Jane Doe	100

You will now verify that the data was updated in DynamoDB with a record



GameScoresByUser

Scan: [Table] GameScoresByUser: Username — Viewing 1 to 2 items

Scan — [Table] GameScoresByUser: Username

Username	Score
Jane Doe	100
Rohan	233

Manually now create an item from Create item tab and observe the updation of data in table

Now we have achieved :

- Created a Lambda function from a blueprint.
- Created an Amazon Kinesis Stream and used it to trigger your Lambda function.
- Used CloudWatch to monitor your function.
- Create an Amazon DynamoDB table and inserted sample data.
- Enabled Amazon DynamoDB Streams.
- Tested and enabled the Lambda function on an Amazon DynamoDB table.