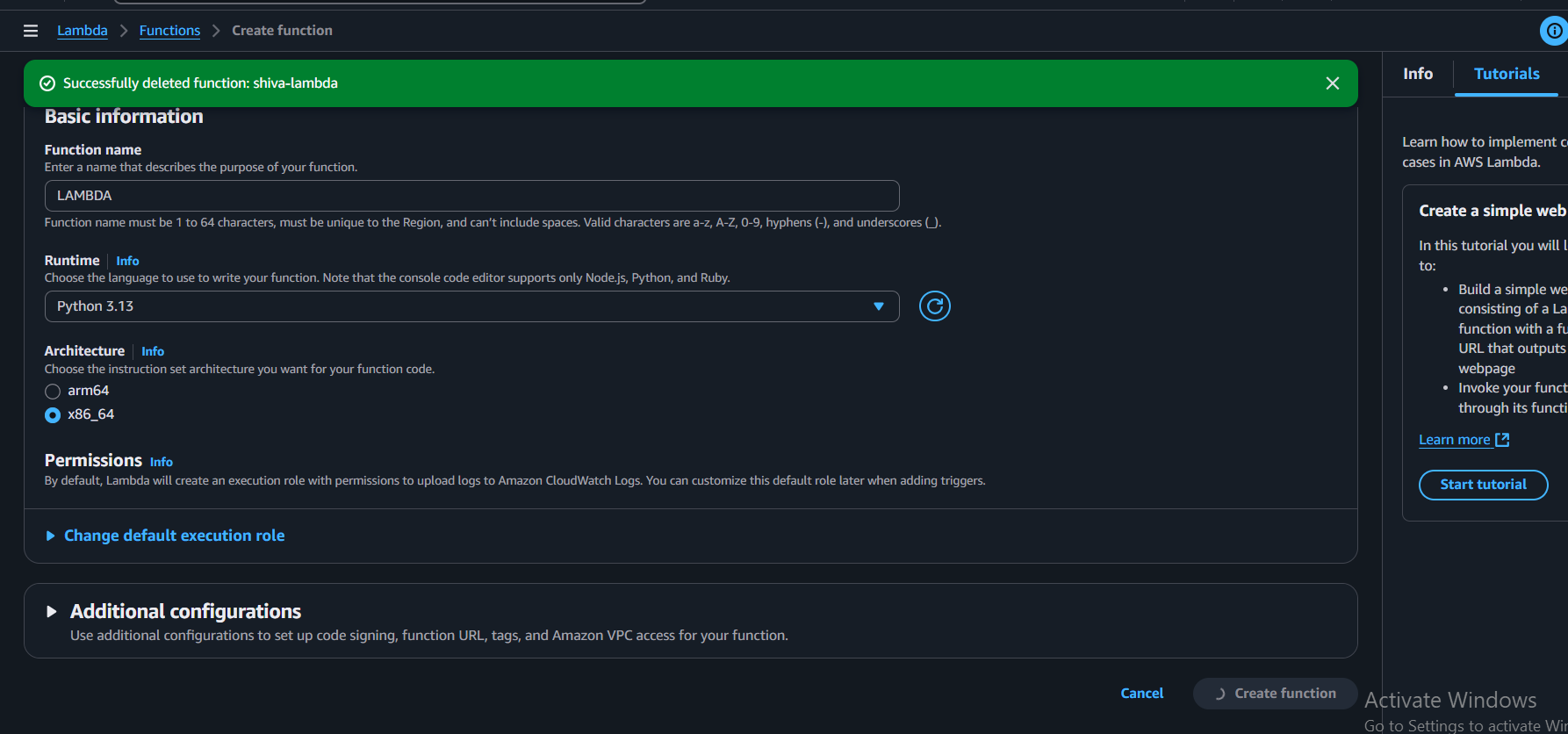
AWS LAMBDA:

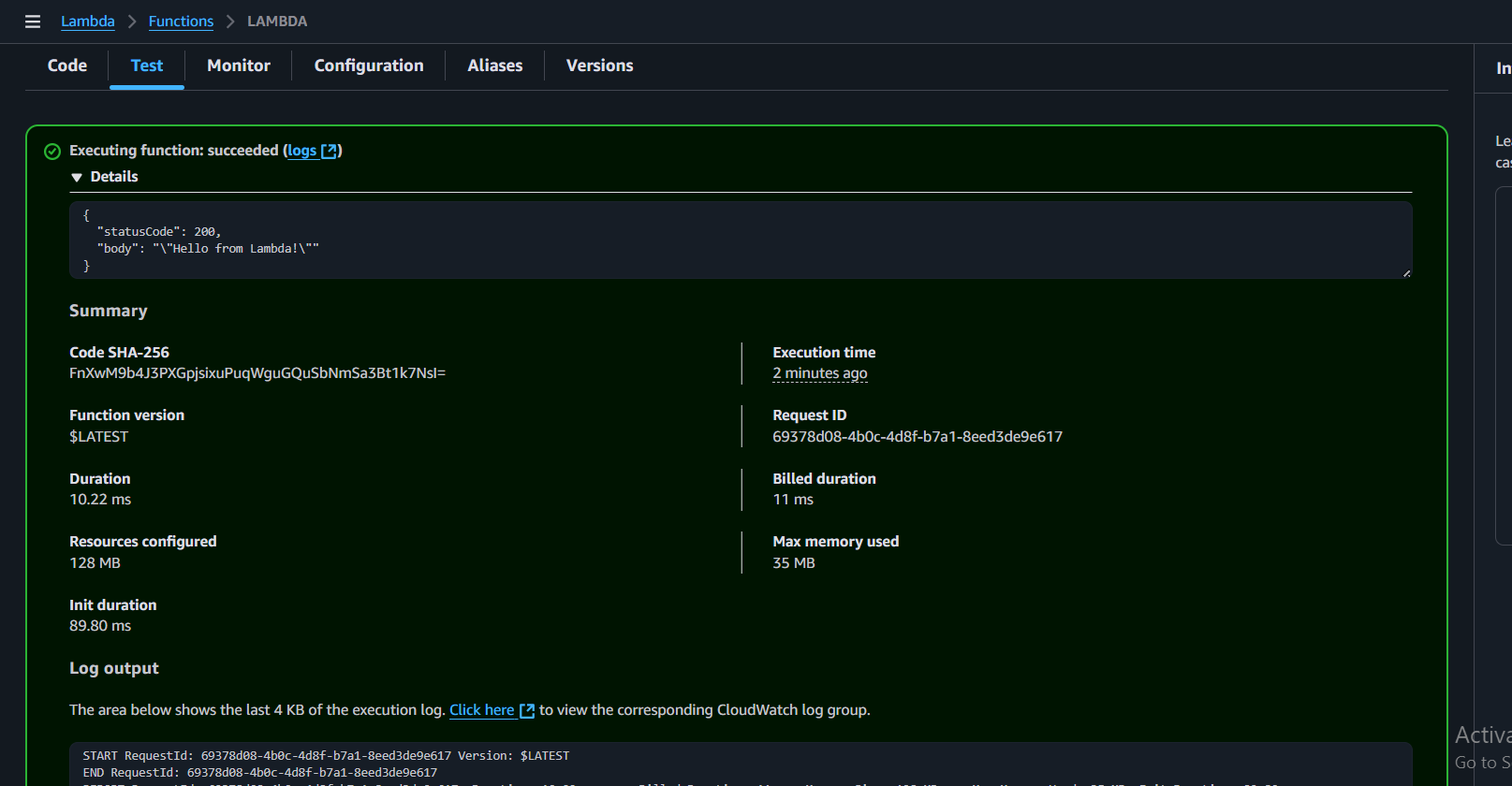
Created a Lambda function named LAMBDA.

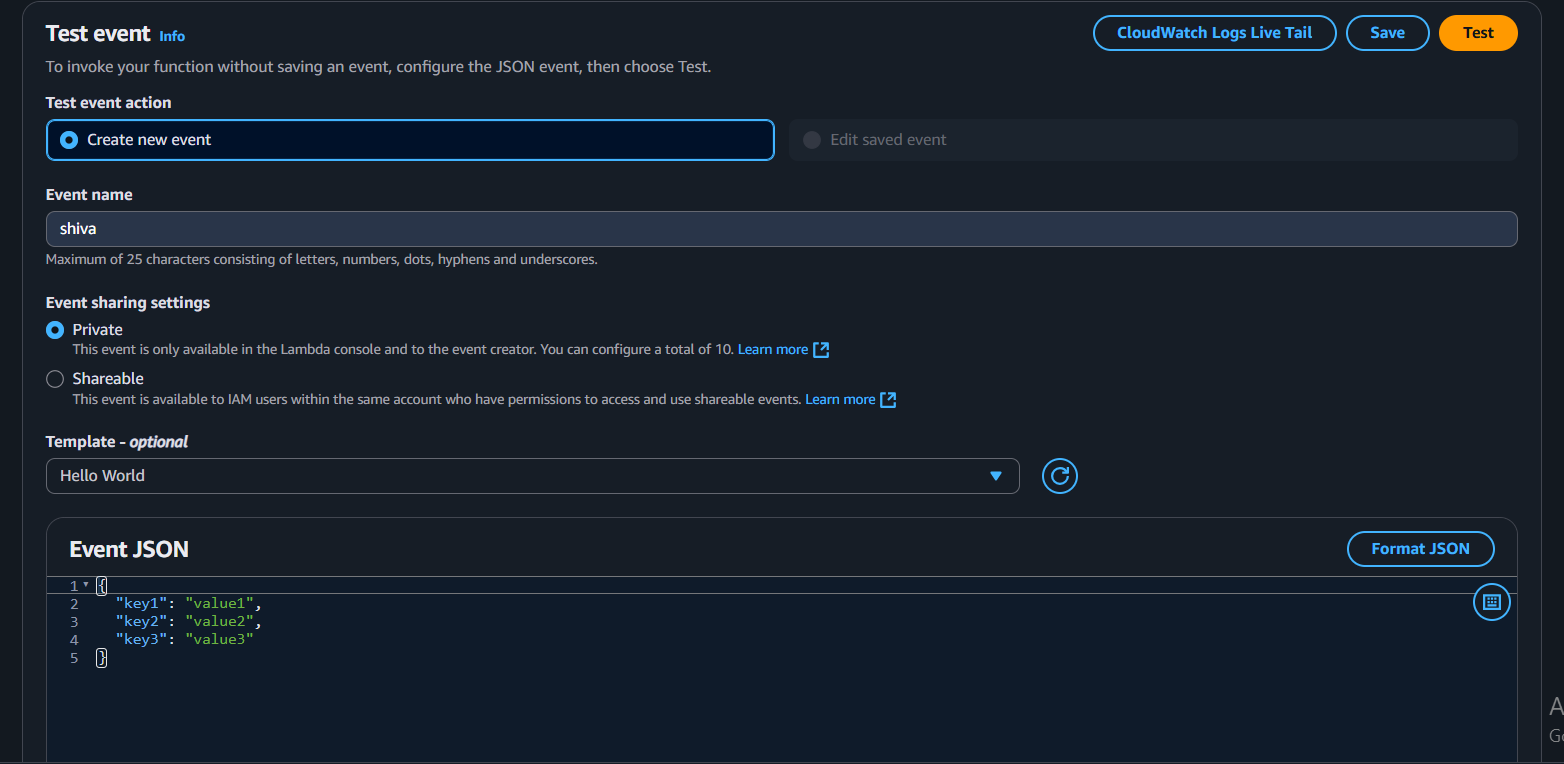
Wrote basic Lambda function code (lambda\_handler).

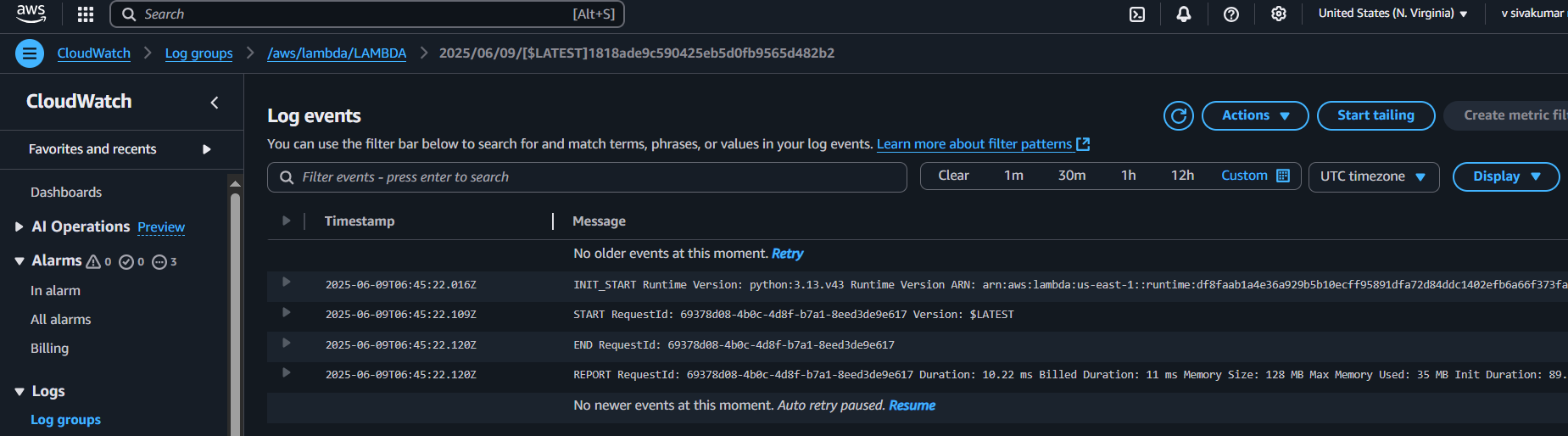
Deployed the Lambda function in AWS Console.

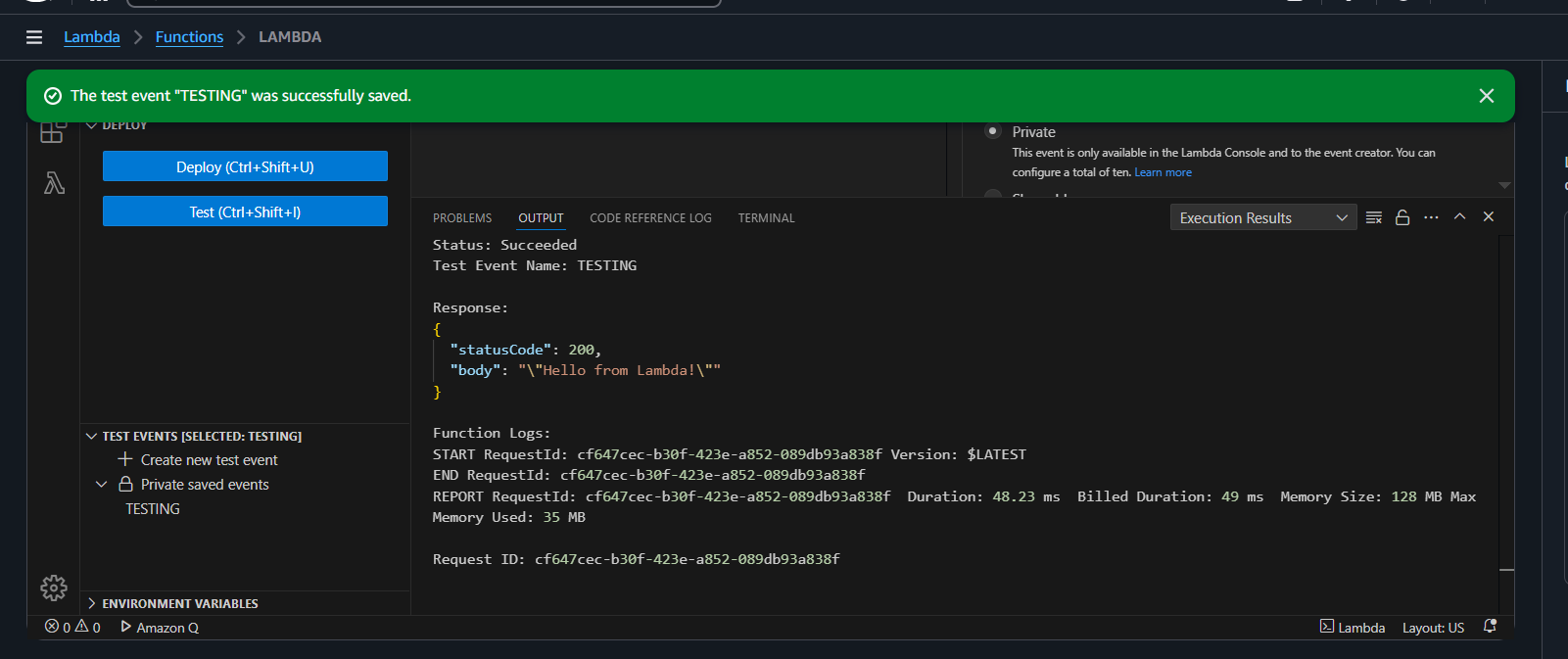
Tested the Lambda function by running a test event from the console.











API GATEWAYS:

Created a new REST API named MyLambdaAPI.

Created a resource path /hello under the API.

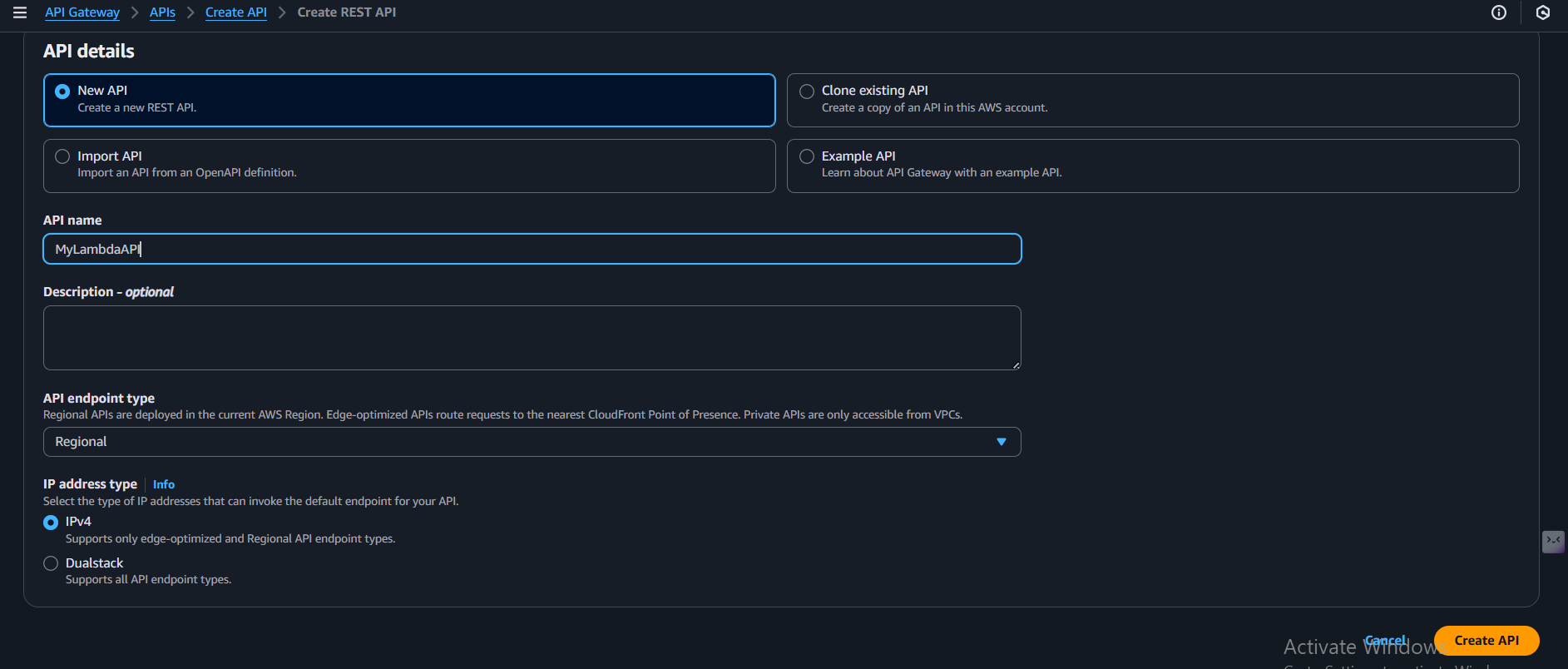
Created a method (ANY) on the /hello resource.

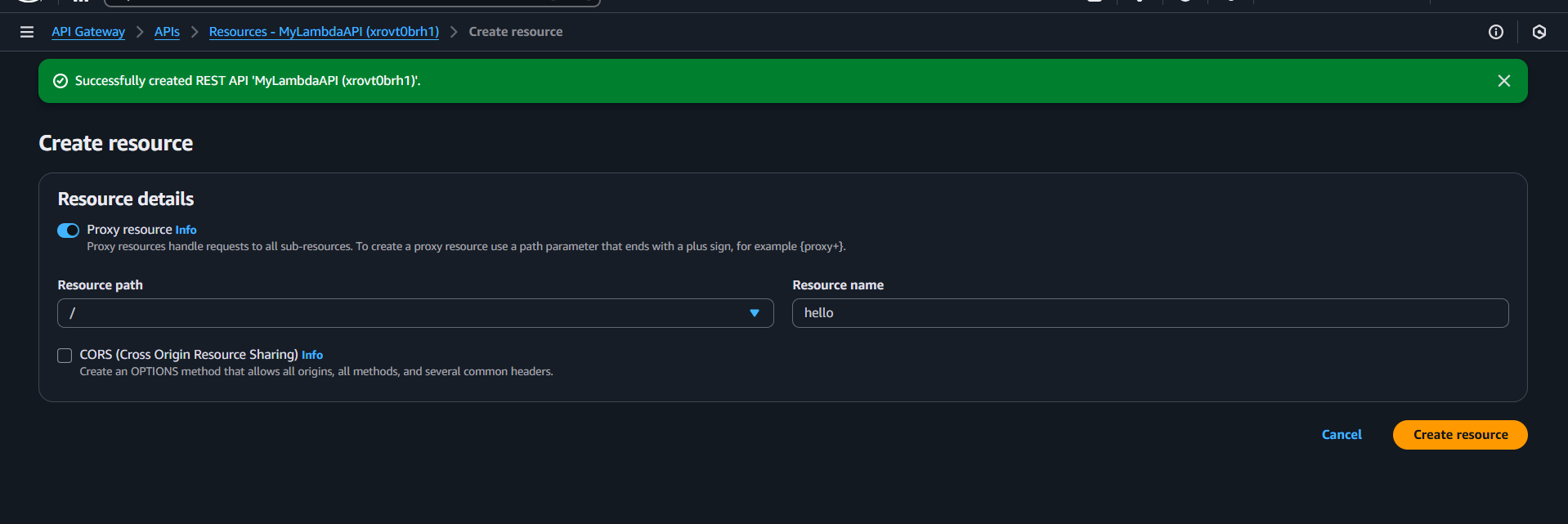
Integrated the method with your Lambda function (LAMBDA) using Lambda proxy integration.

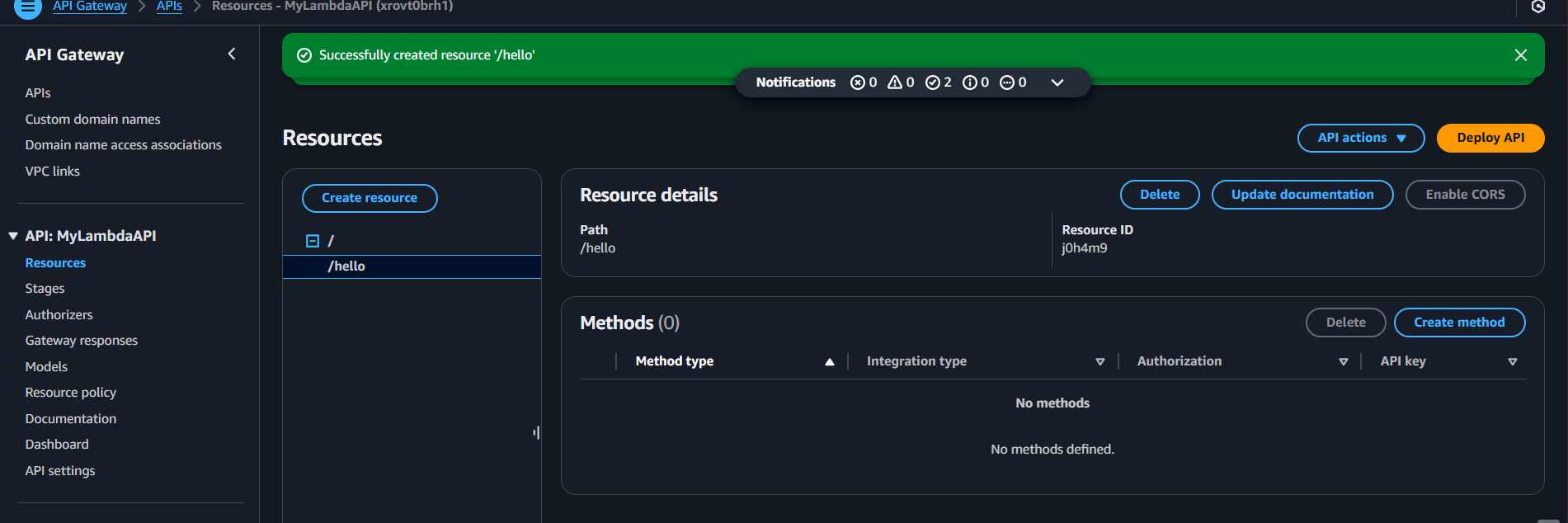
Granted API Gateway permission to invoke your Lambda function.

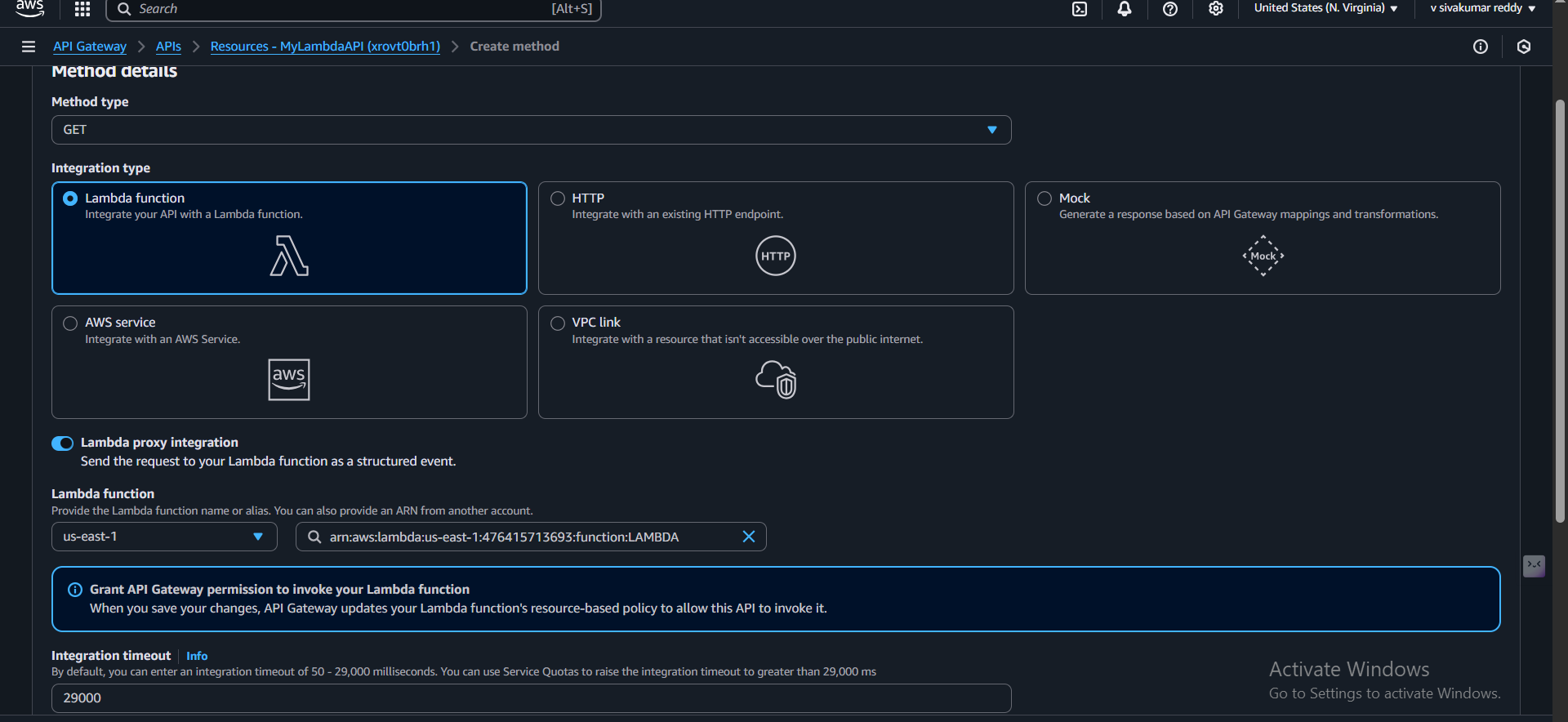
Deployed the API to a stage named prod.

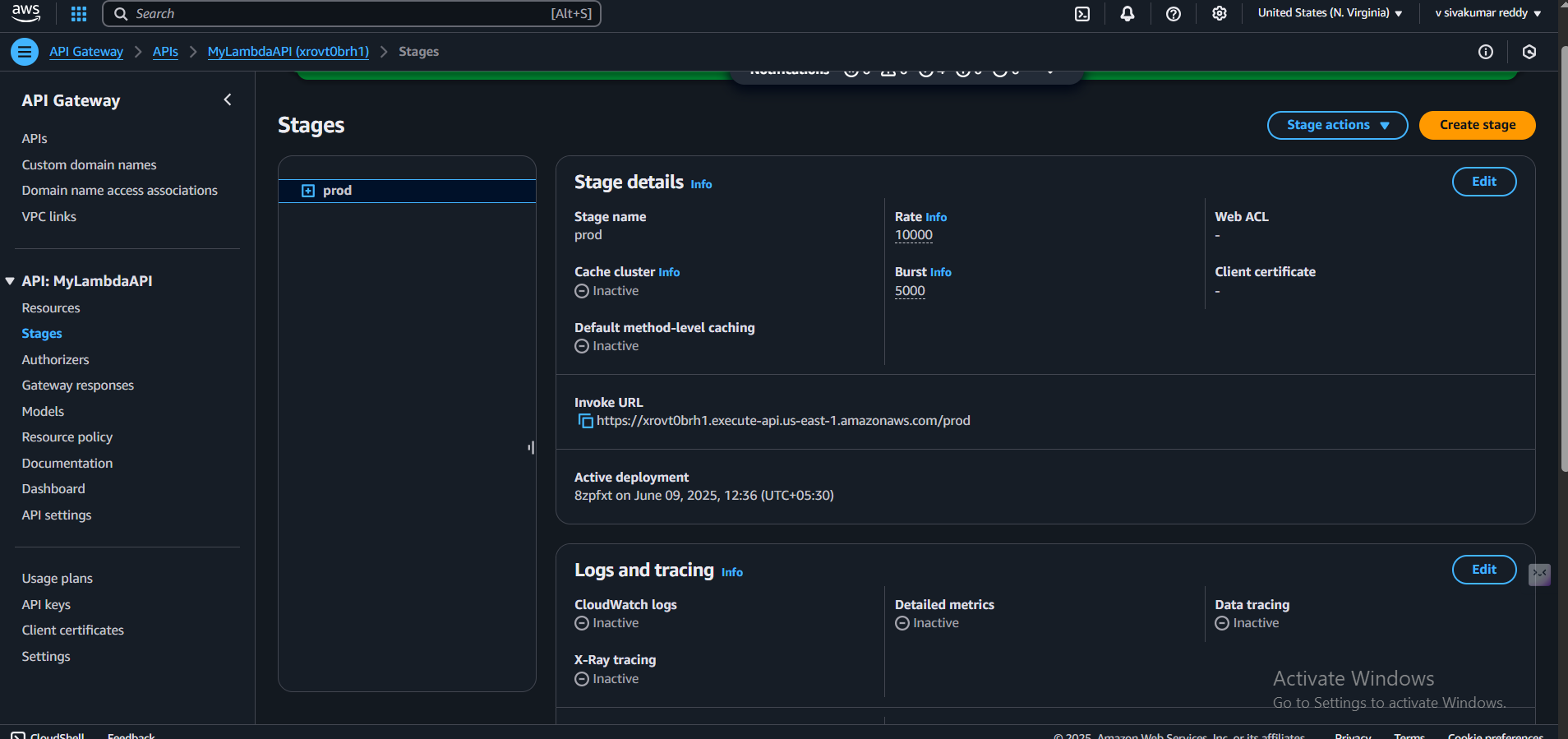
Tested the API endpoint using the Invoke URL (e.g., https://xrovt0brh1.execute-api.us-east-1.amazonaws.com/prod/hello).

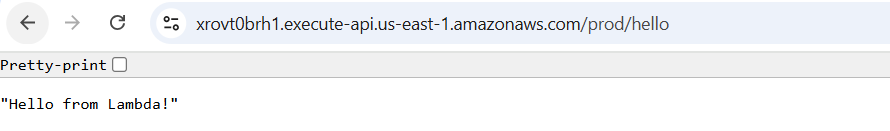












CUSTOMER MANAGED KEYS:

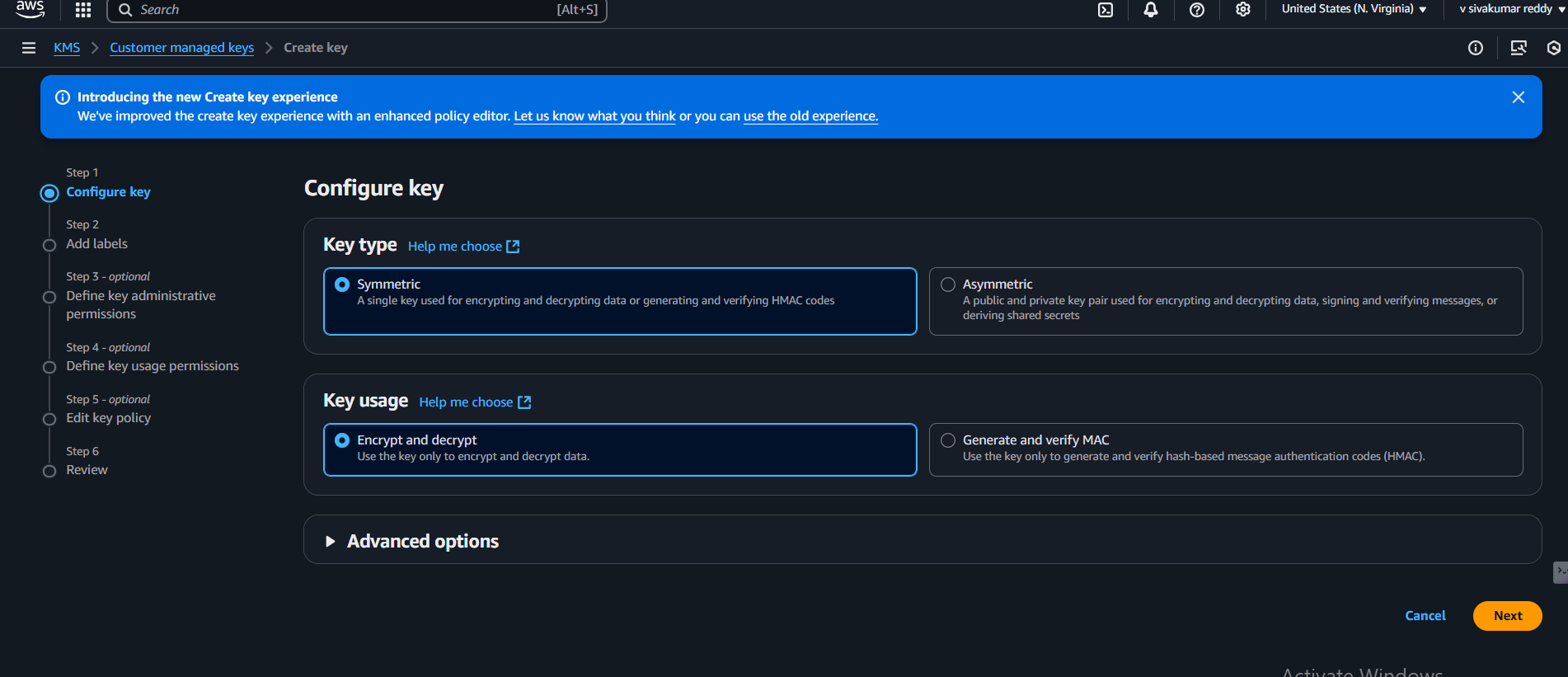
Created a new symmetric Customer Managed Key (CMK) in AWS KMS.

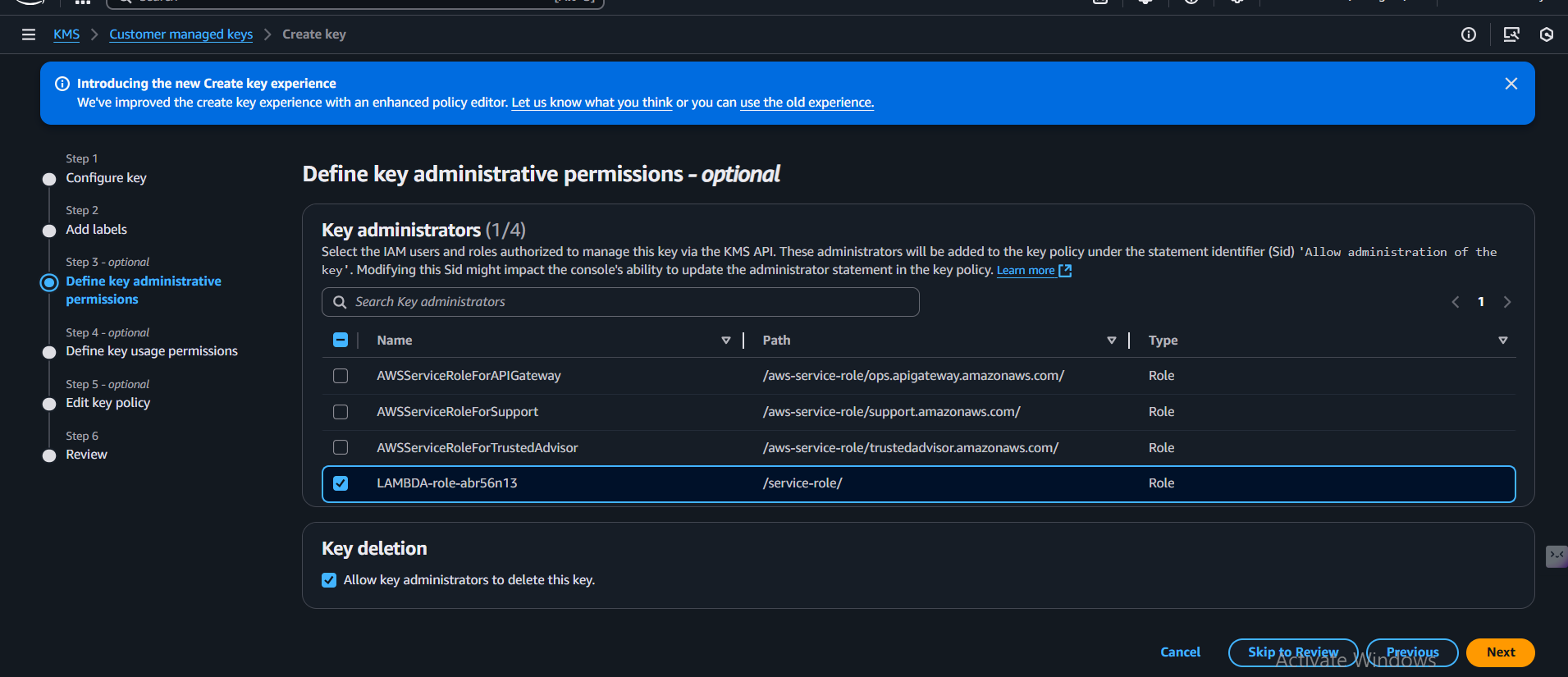
Configured the key settings like alias (LAMBDA) and key usage (Encrypt and Decrypt).

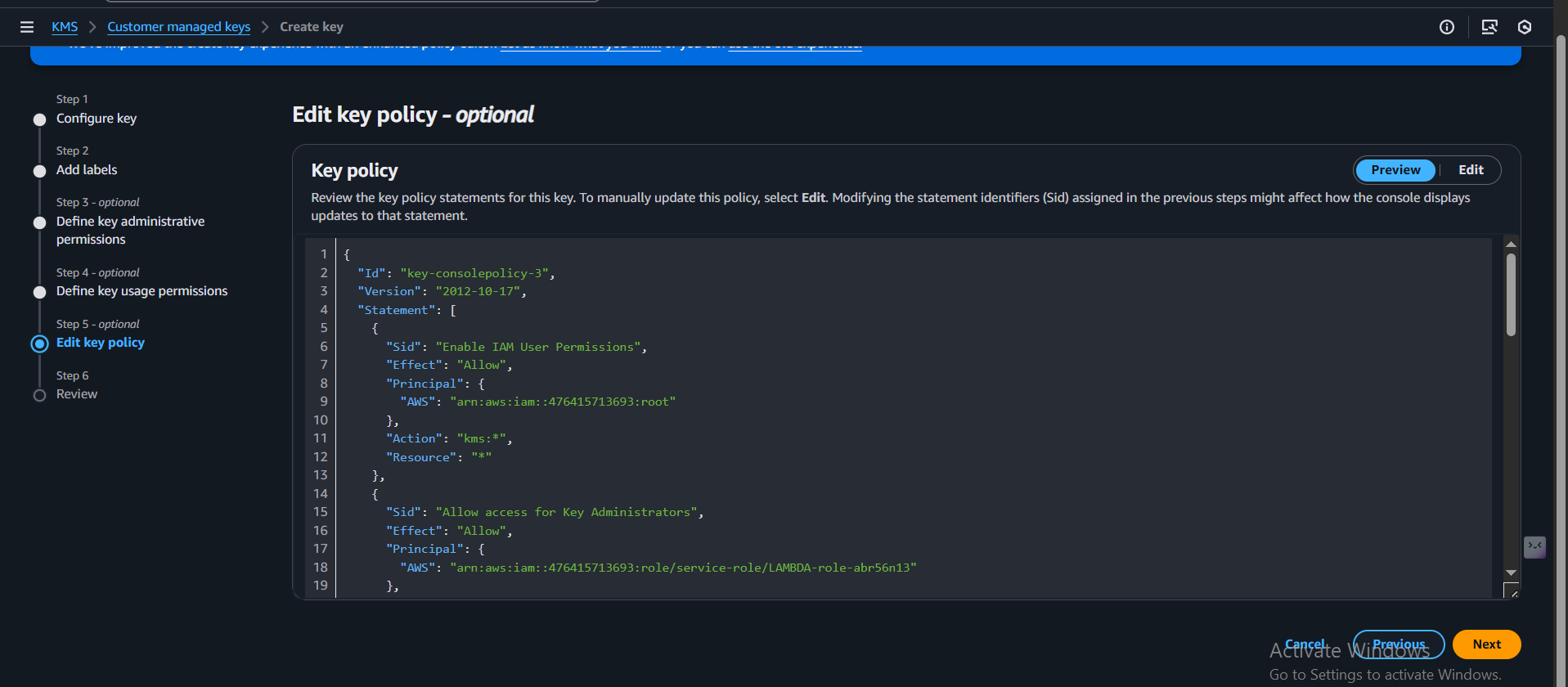
Added key administrators (IAM roles/users who can manage the key).

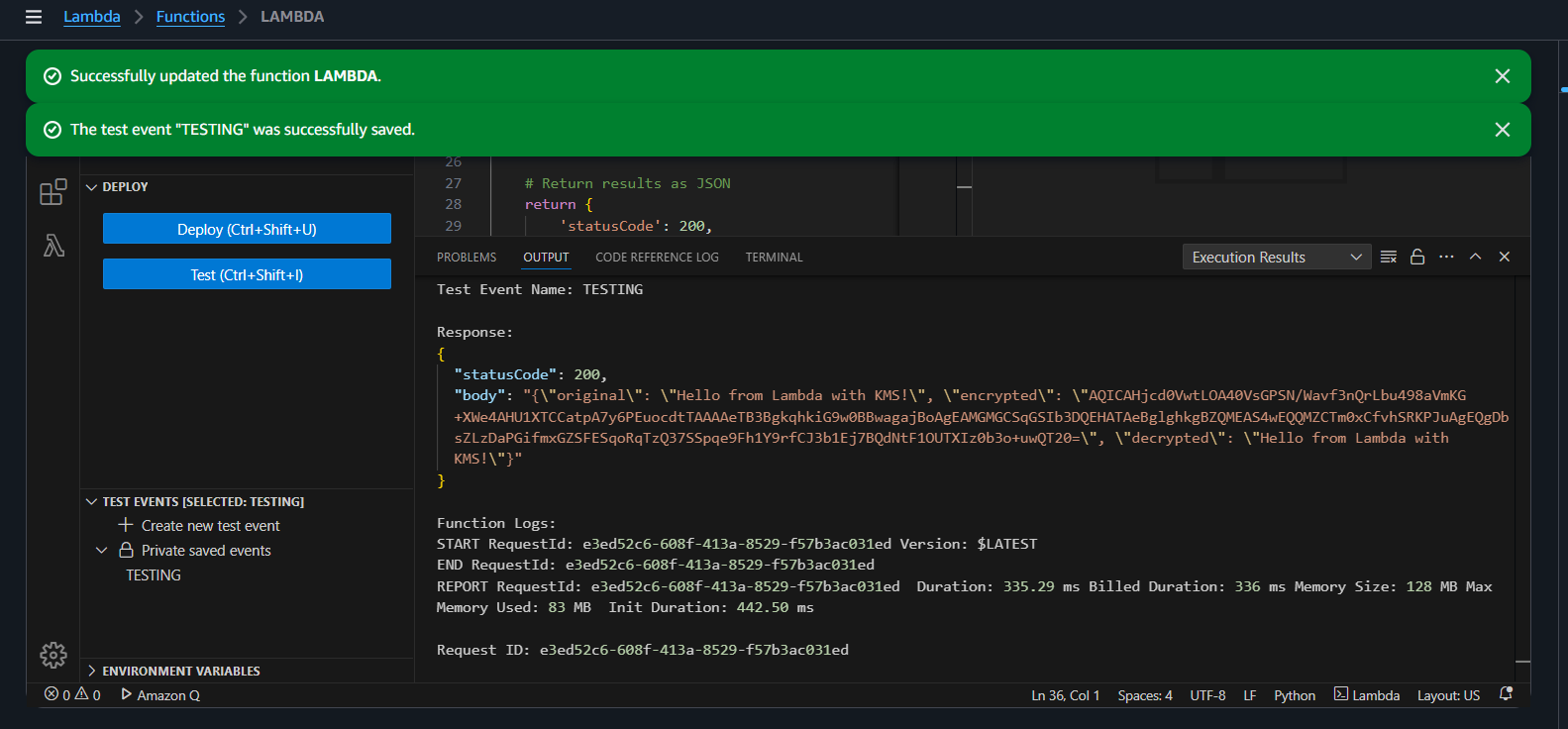
Added key users (IAM roles/users allowed to use the key for cryptographic operations) — included your Lambda execution role.

Verified key status is Enabled and the key policy allows proper access.









AWS SQS:

1. Created a Standard Queue

You created a new Standard Queue named MyTestQueue in the us-east-1 region.

Queue URL:

https://sqs.us-east-1.amazonaws.com/476415713693/MyTestQueue

2. Checked Queue Details

Verified details like:

Queue ARN

Encryption type (SSE-SQS using Amazon SQS-managed key)

No dead-letter queue (DLQ) configured yet

3. Sent a Message to the Queue

Sent a message with body:

"HELLO FROM SHIVA"

Used the SQS Console to send the message manually.

4. Polled and Received the Message

Used the “Poll for messages” option.

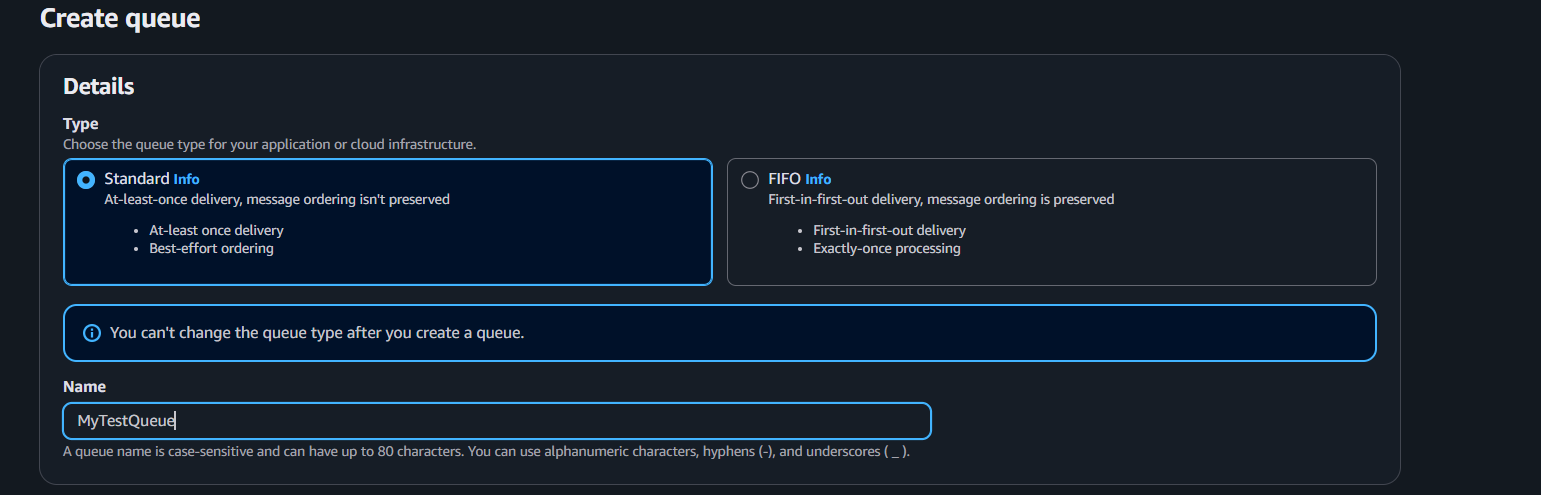
Successfully received the message:

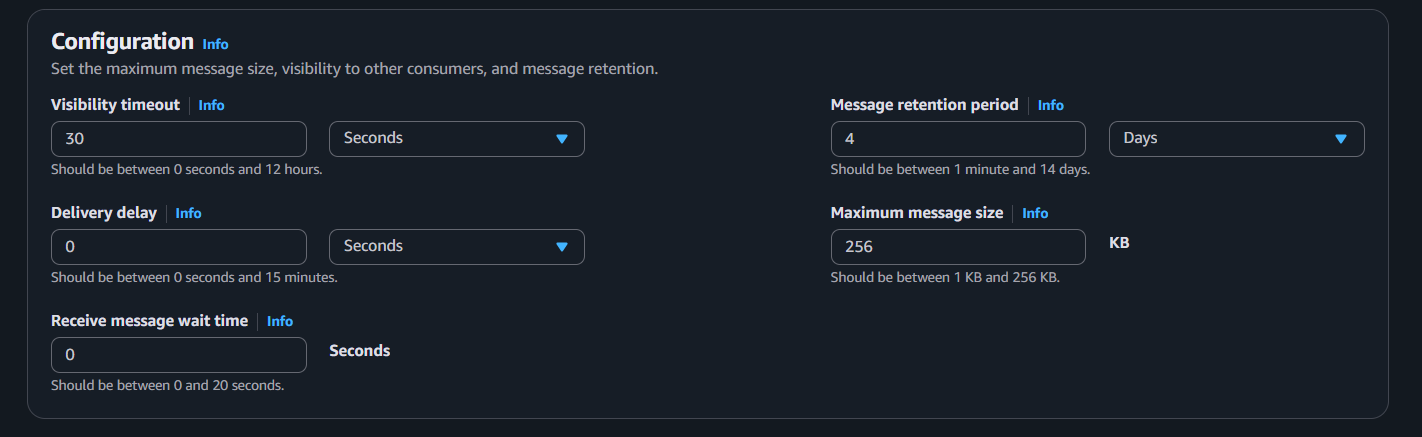
Message ID: bef49a80-fc6f-4501-8667-3016c2daf21c

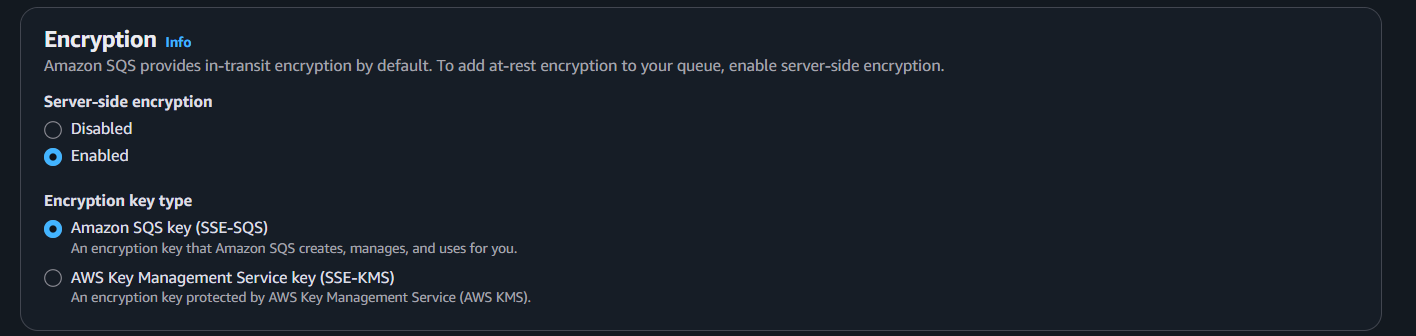
Message body: "HELLO FROM SHIVA"

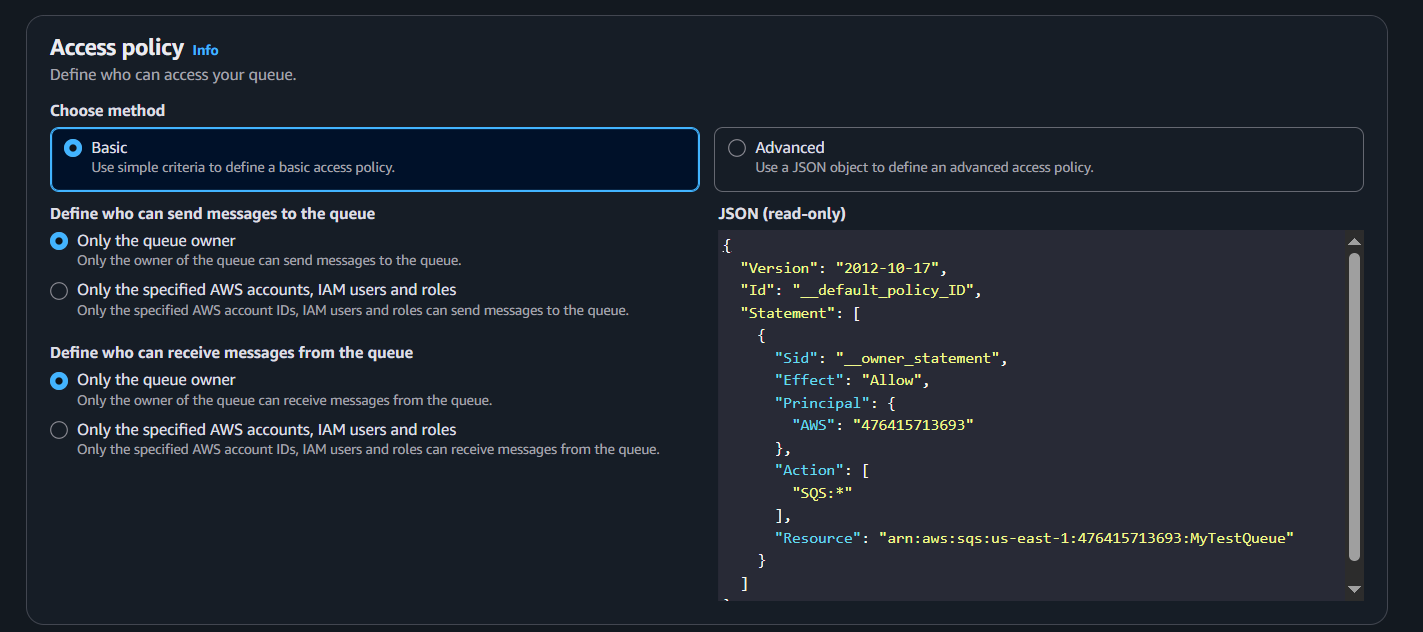
Size: 16 bytes

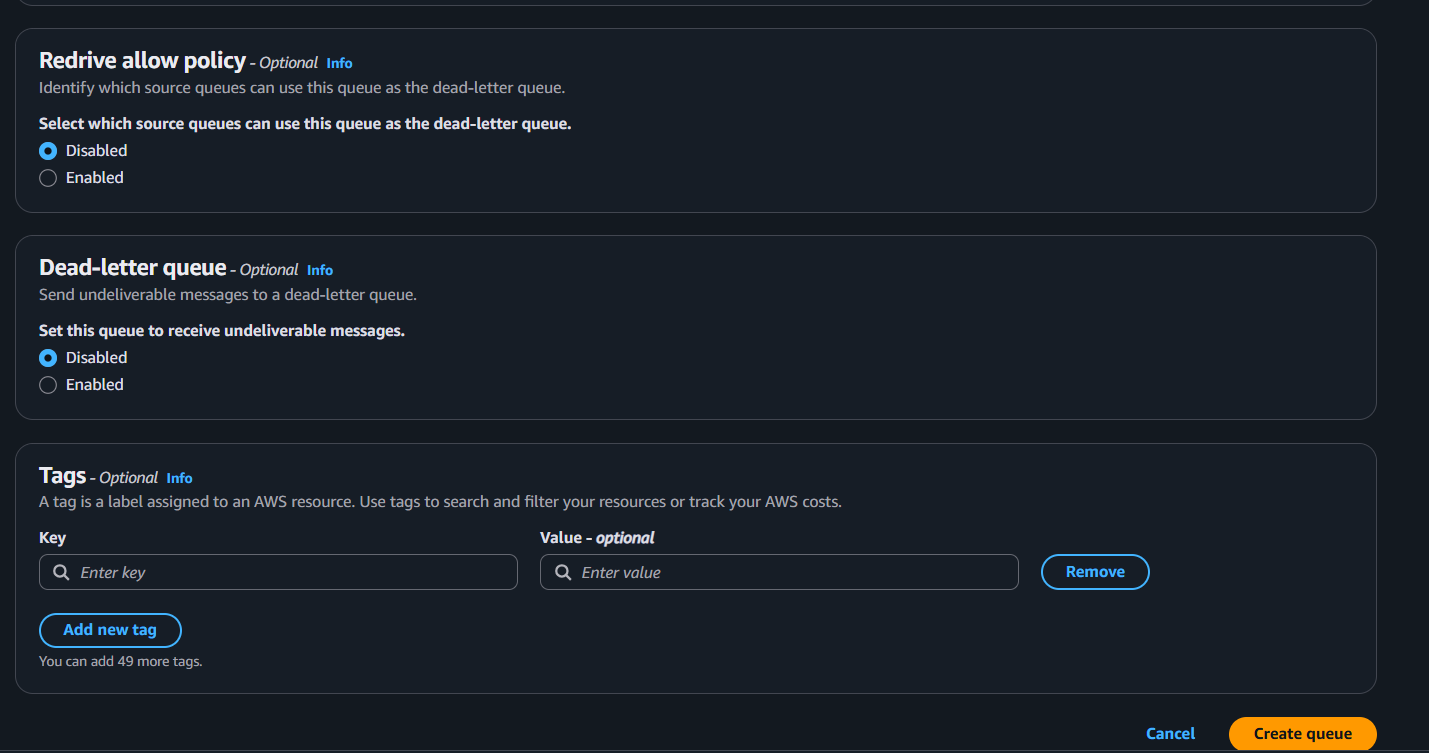
Receive count: 2

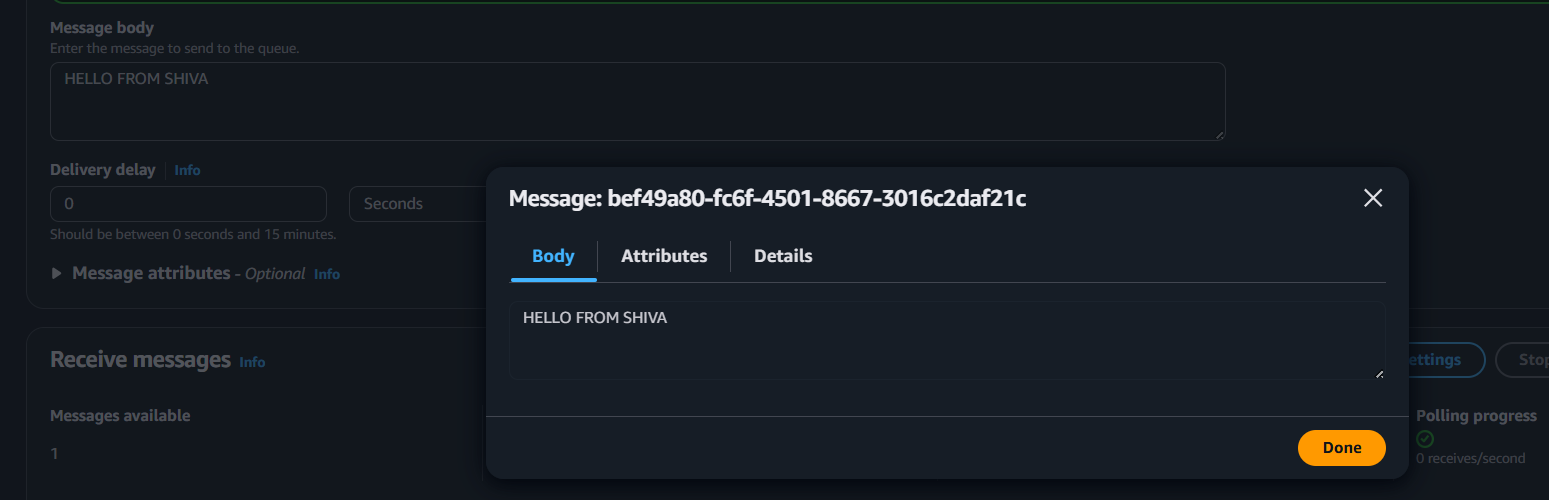












AMAZON MQ:

Created Broker:

Name: MyTestBroker

Engine: Apache ActiveMQ (v5.18.7)

Instance Type: mq.t3.micro (Free Tier eligible)

Deployment Mode: Active/standby

Storage Type: Amazon EFS

Configured Access:

Enabled public accessibility ✅

Setup admin credentials for accessing the ActiveMQ Web Console

Username: admin

Password: Shivakumar@11

Broker Status:

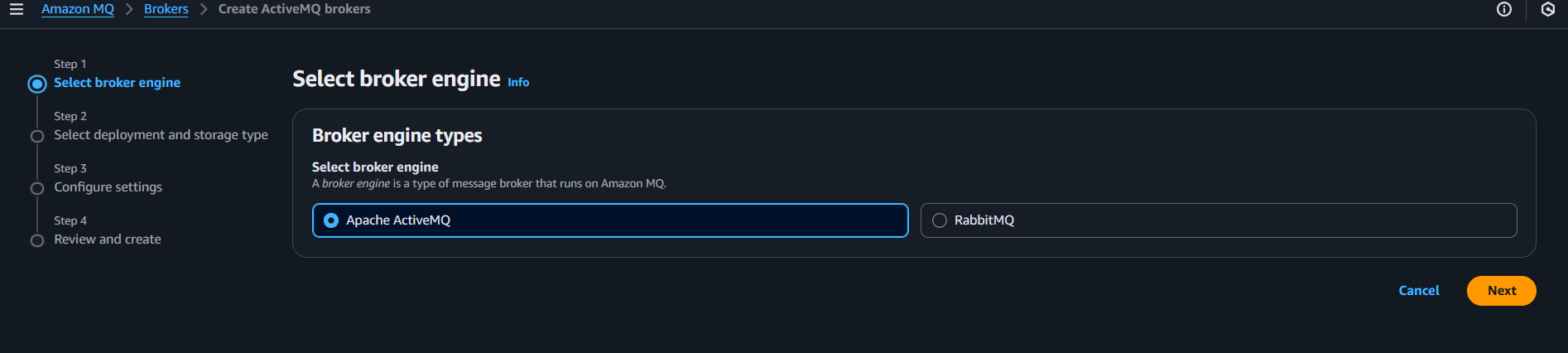
Successfully deployed and showing status: Running

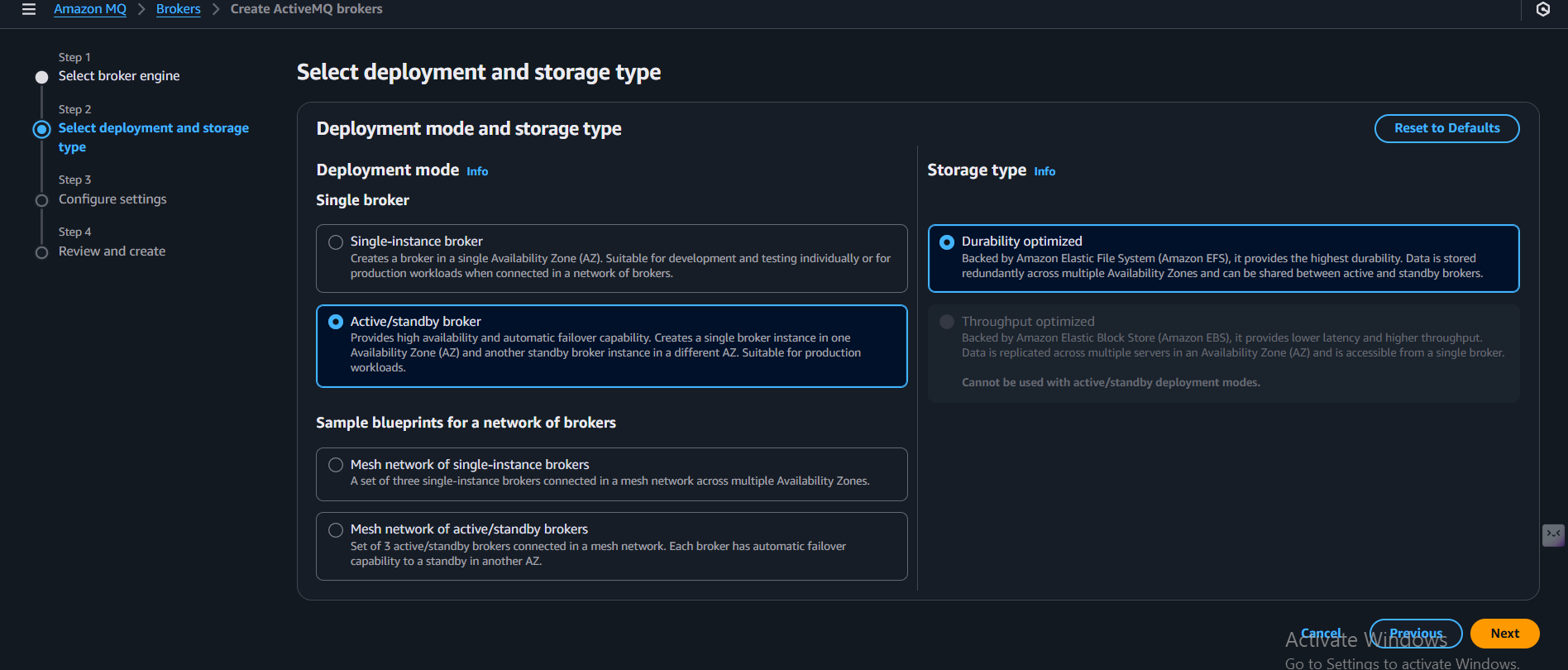
Console Access URLs:

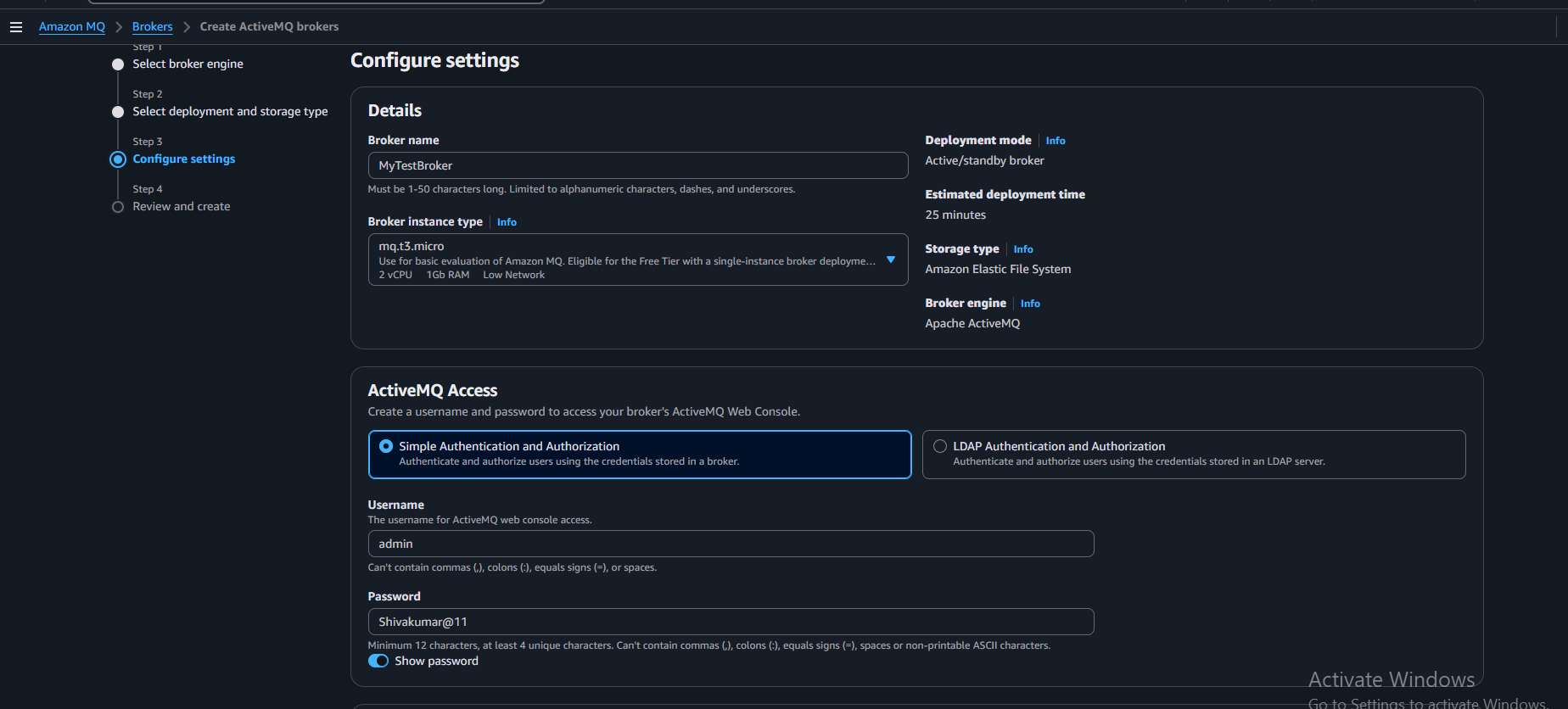
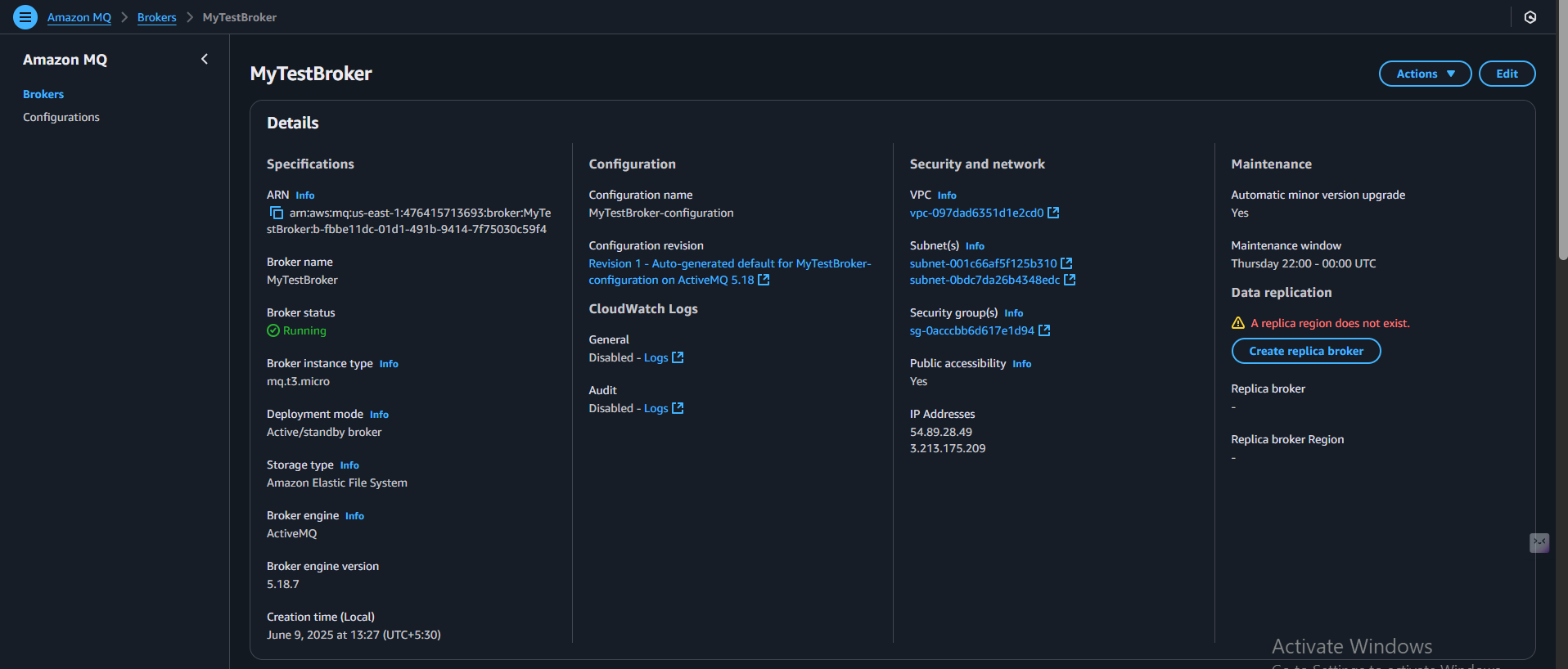
You can access the broker’s ActiveMQ Web Console using:

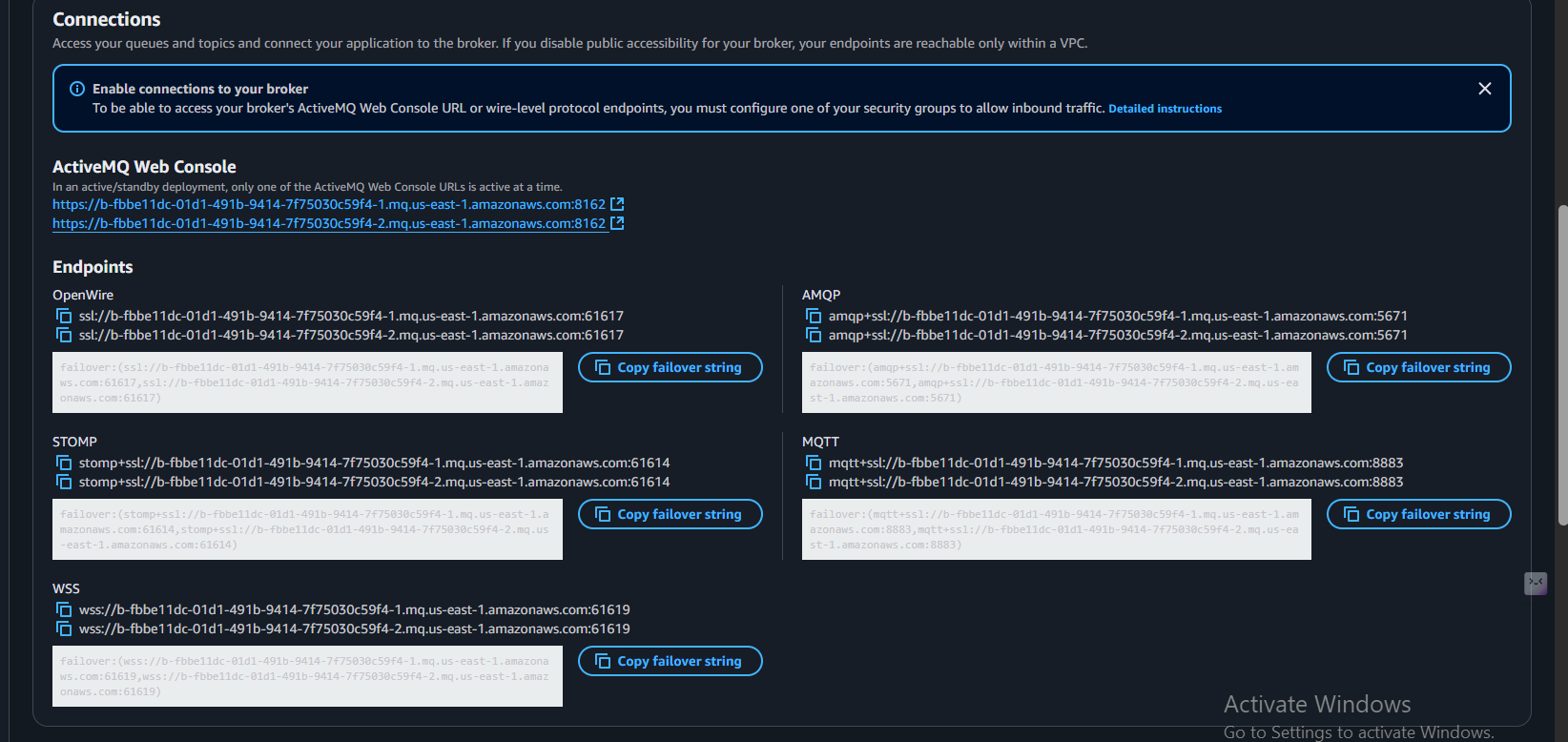
Console URL 1

Console URL 2









AMAZON SNS:

Created an SNS FIFO Topic

Name: MyTestTopic.fifo

Topic ARN: arn:aws:sns:us-east-1:476415713693:MyTestTopic.fifo

Created an Amazon SQS Queue

Name: MyTestQueue

This queue will receive messages from the SNS topic.

Subscribed the SQS Queue to the SNS Topic

Protocol: Amazon SQS

Subscription Status: Confirmed

Published a Message to the SNS Topic

Subject: Test message from SNS FIFO topic

Message Body: hello from shiva

Message Group ID: group1 (mandatory for FIFO topics)

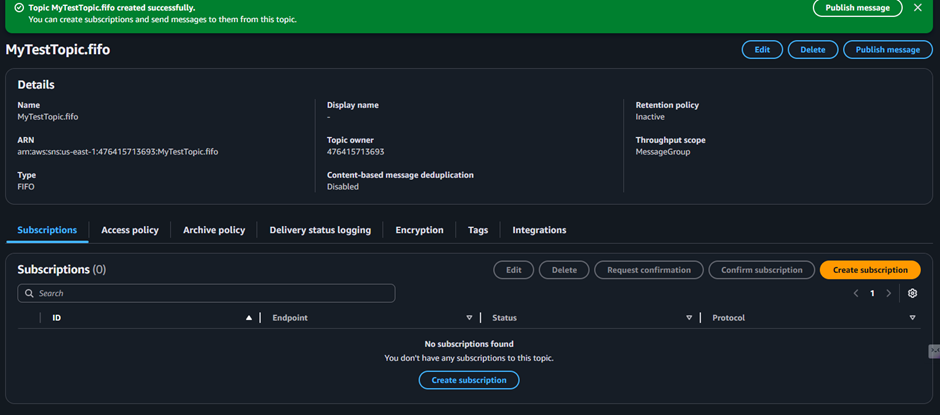
Message Deduplication ID: a unique string (e.g., msg1)

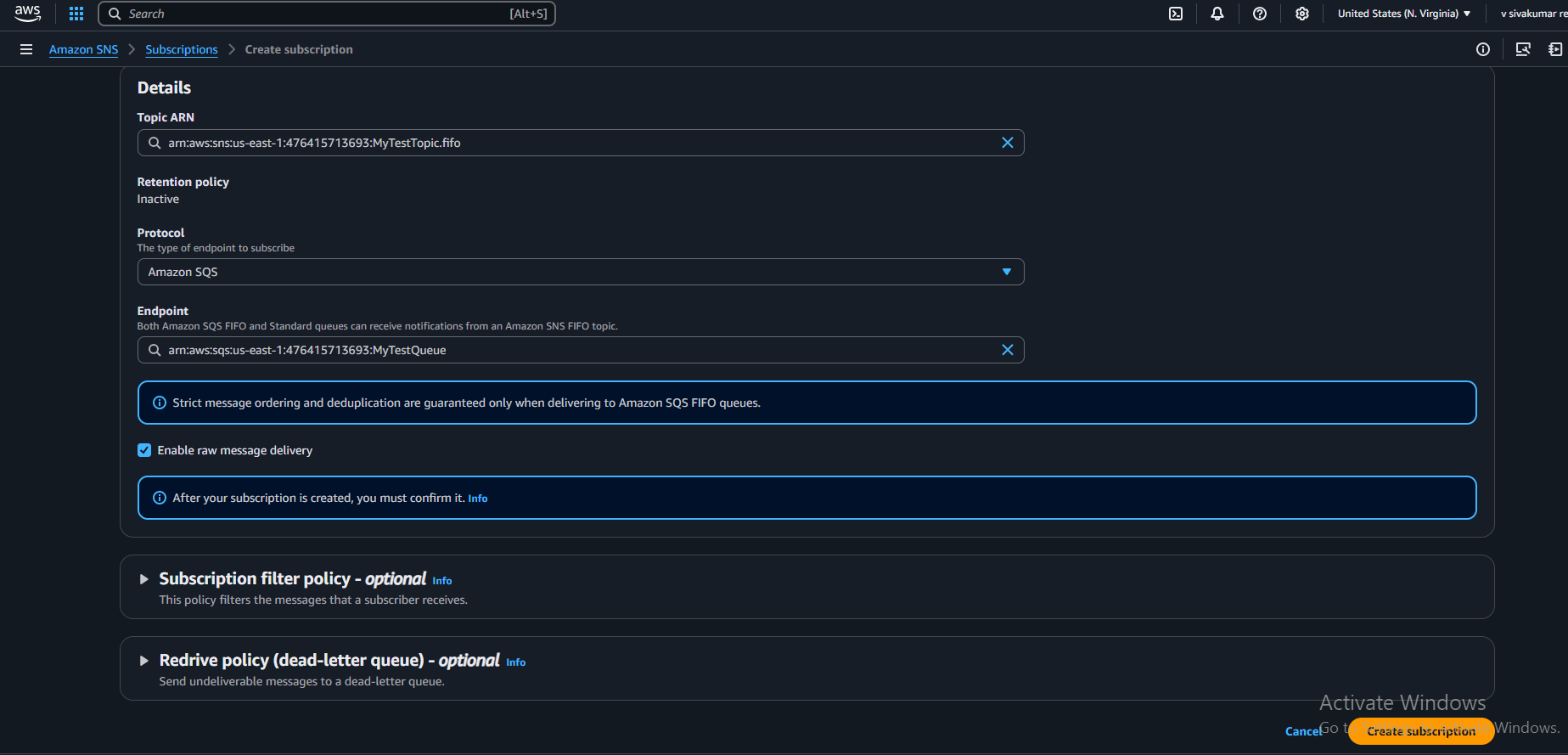
Verified the Message Delivery to SQS

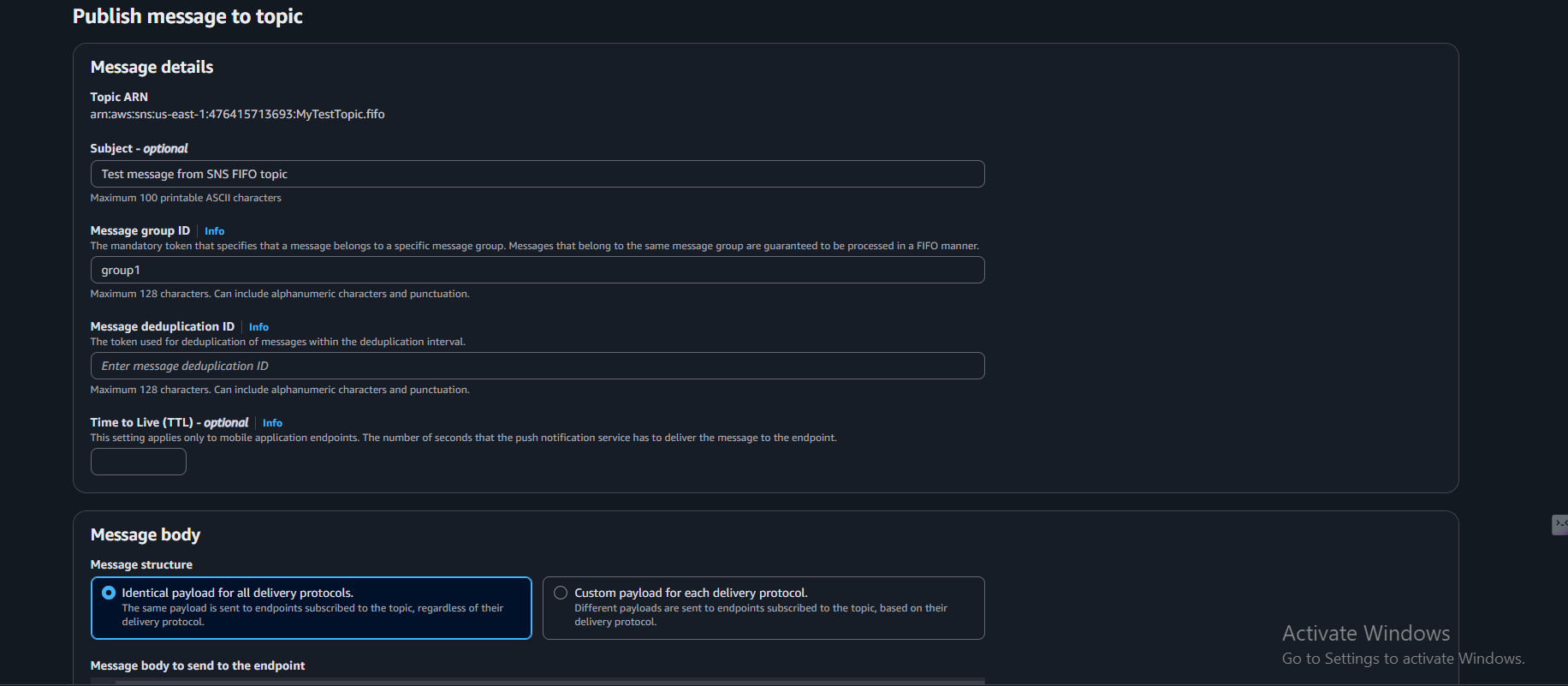
Went to the SQS console

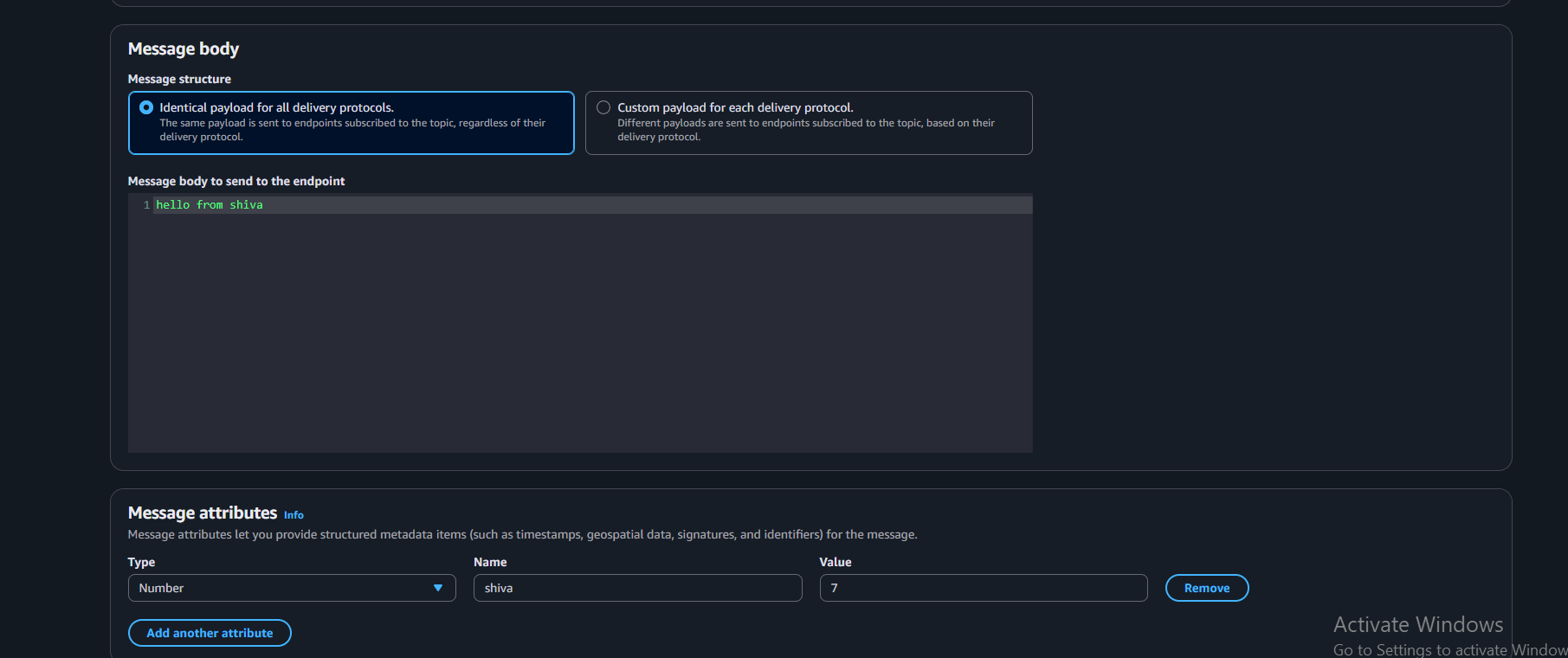
Polled the queue

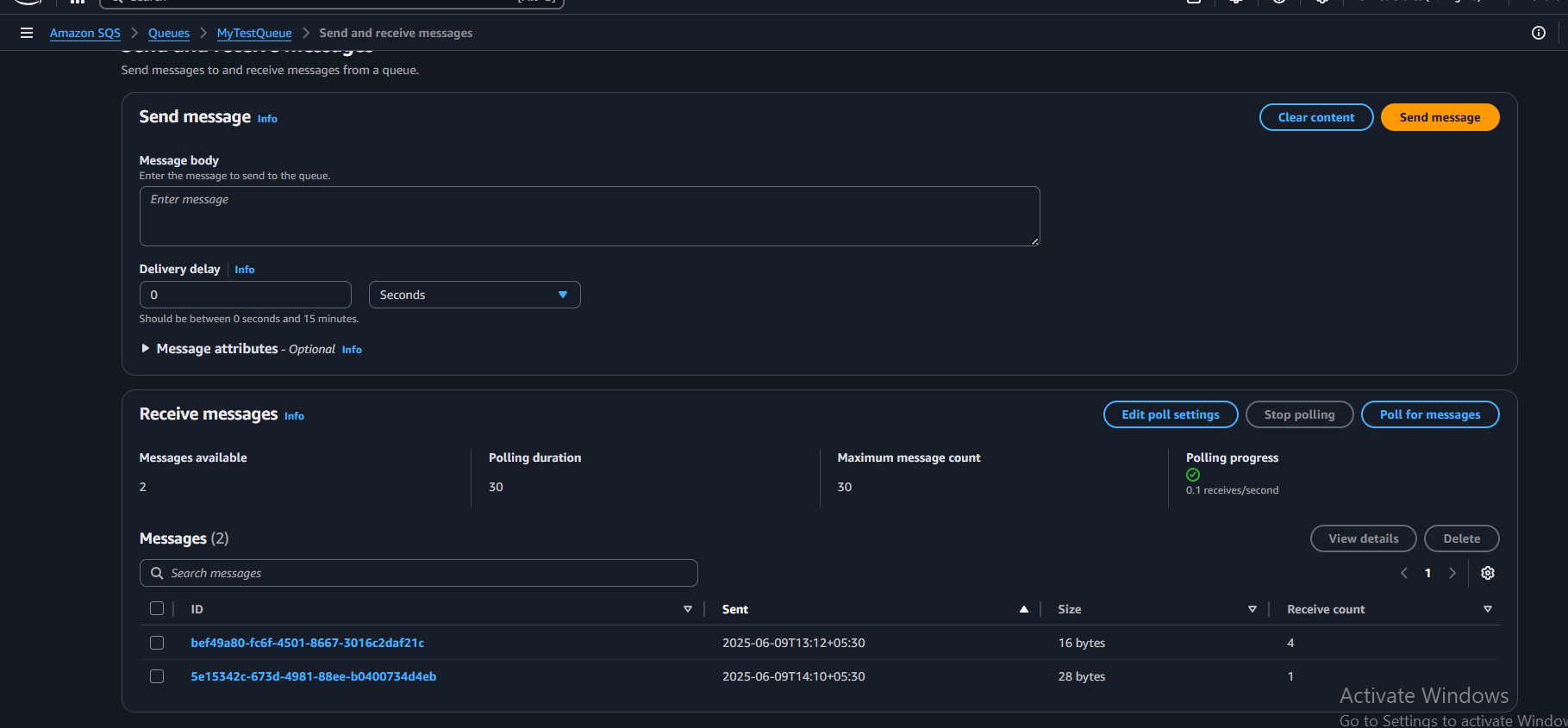
Successfully received the message published from SNS

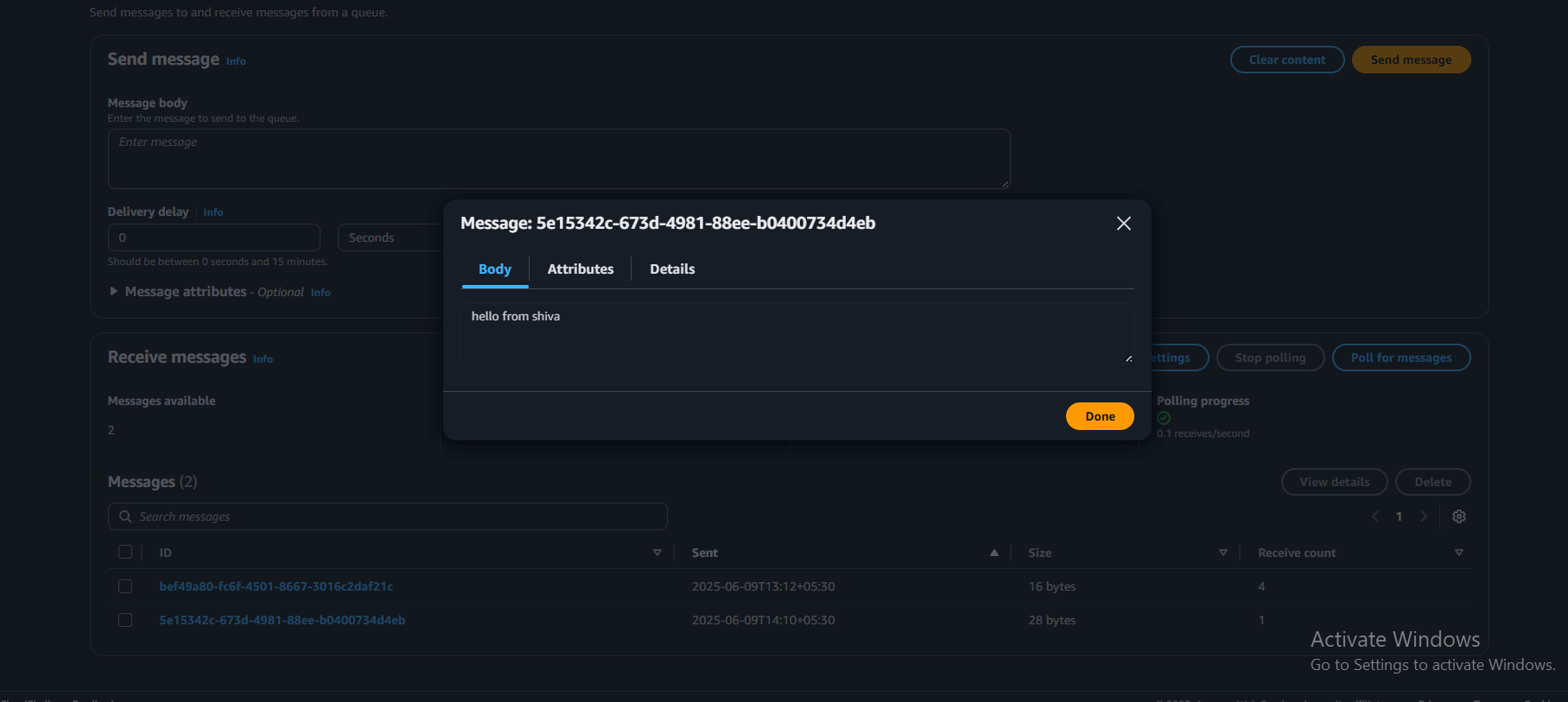












AWS SECRET MANAGER:

Navigated to AWS Secrets Manager

Clicked “Store a new secret”

Selected Other type of secret

Entered key-value:

admin : MySecurePass123

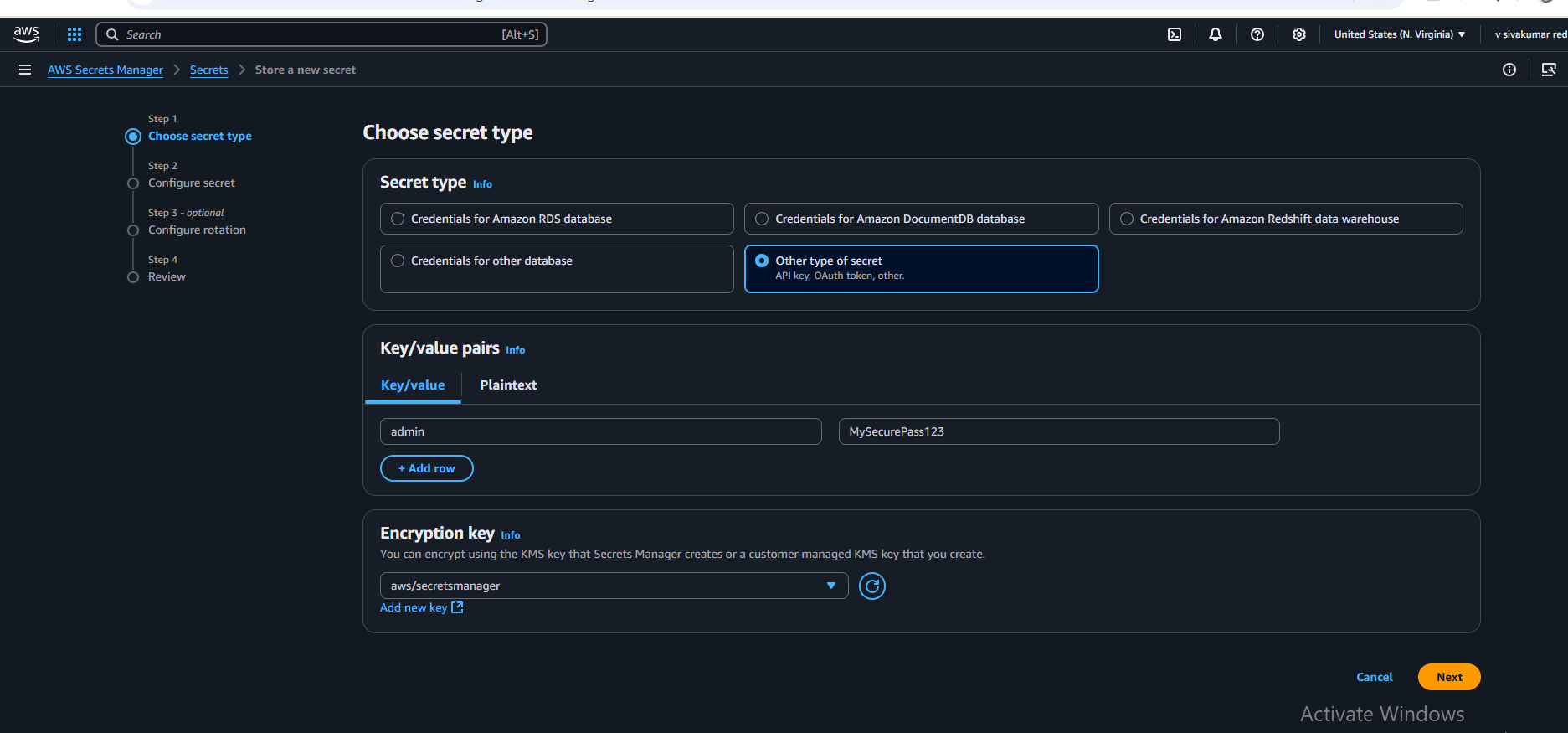
Named the secret: MyAppSecret

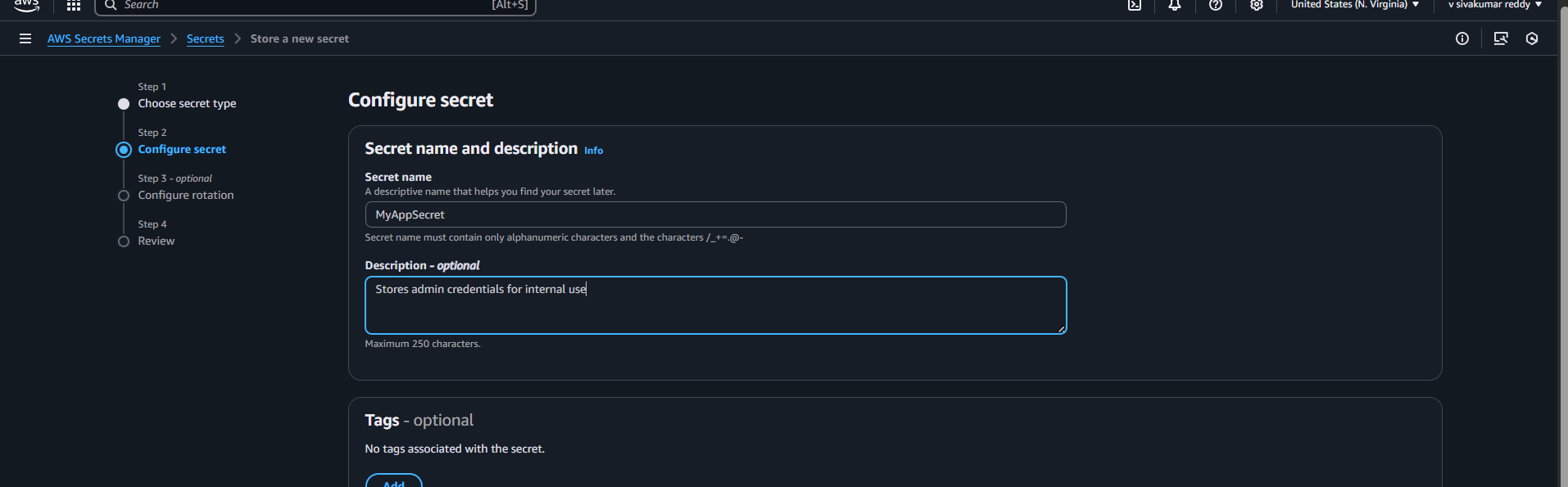
Chose default encryption key: aws/secretsmanager

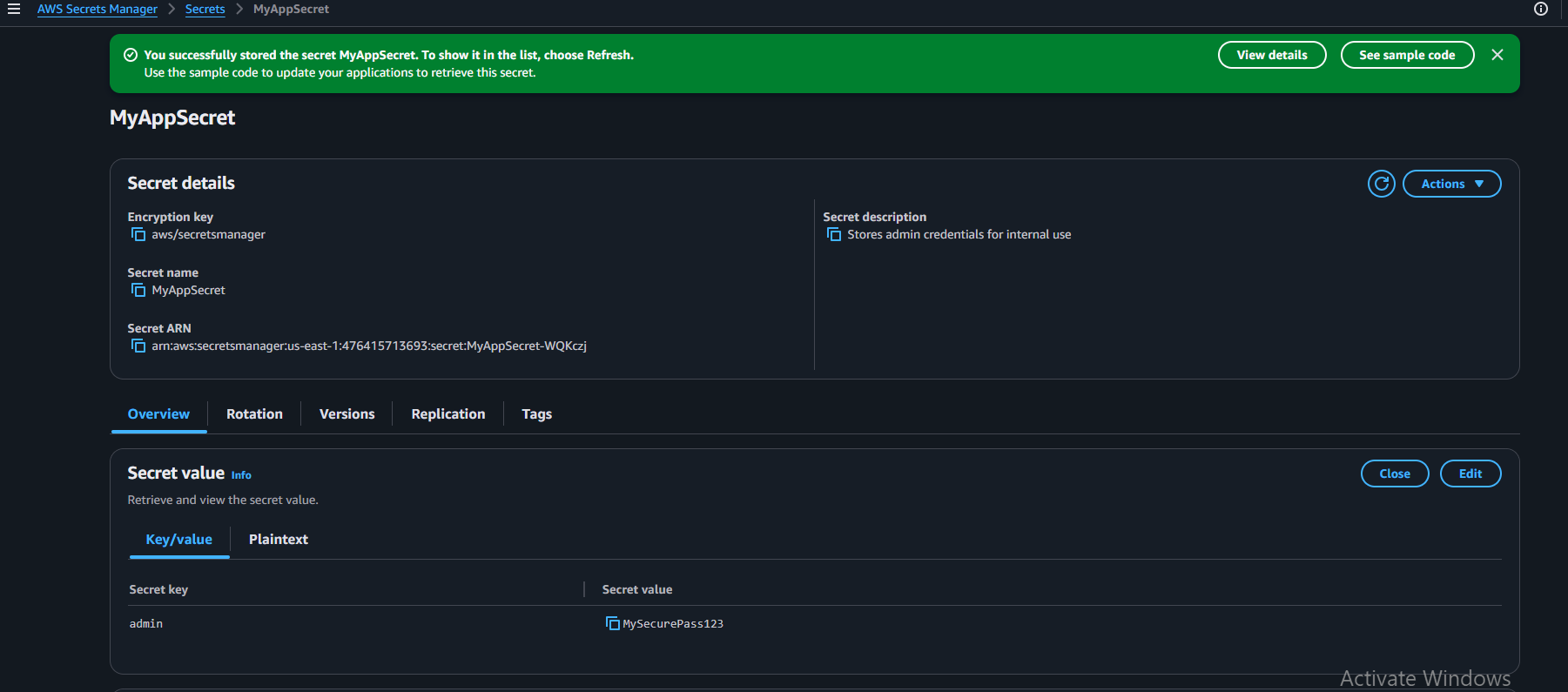
Skipped rotation

Clicked Store

Successfully viewed the stored secret value







AWS WAF:

Opened AWS WAF in the console

→ Searched for “WAF” and went to the AWS WAF Dashboard.

Created a Web ACL (Access Control List)

→ Gave it a name like MyTestWebACL.

Associated the Web ACL

→ (Optional step if linked to a resource like CloudFront, ALB, or API Gateway.)

Created an IP Set

→ Added your IP address (or range) to allow/block specific IPs.

Added Rules to the Web ACL

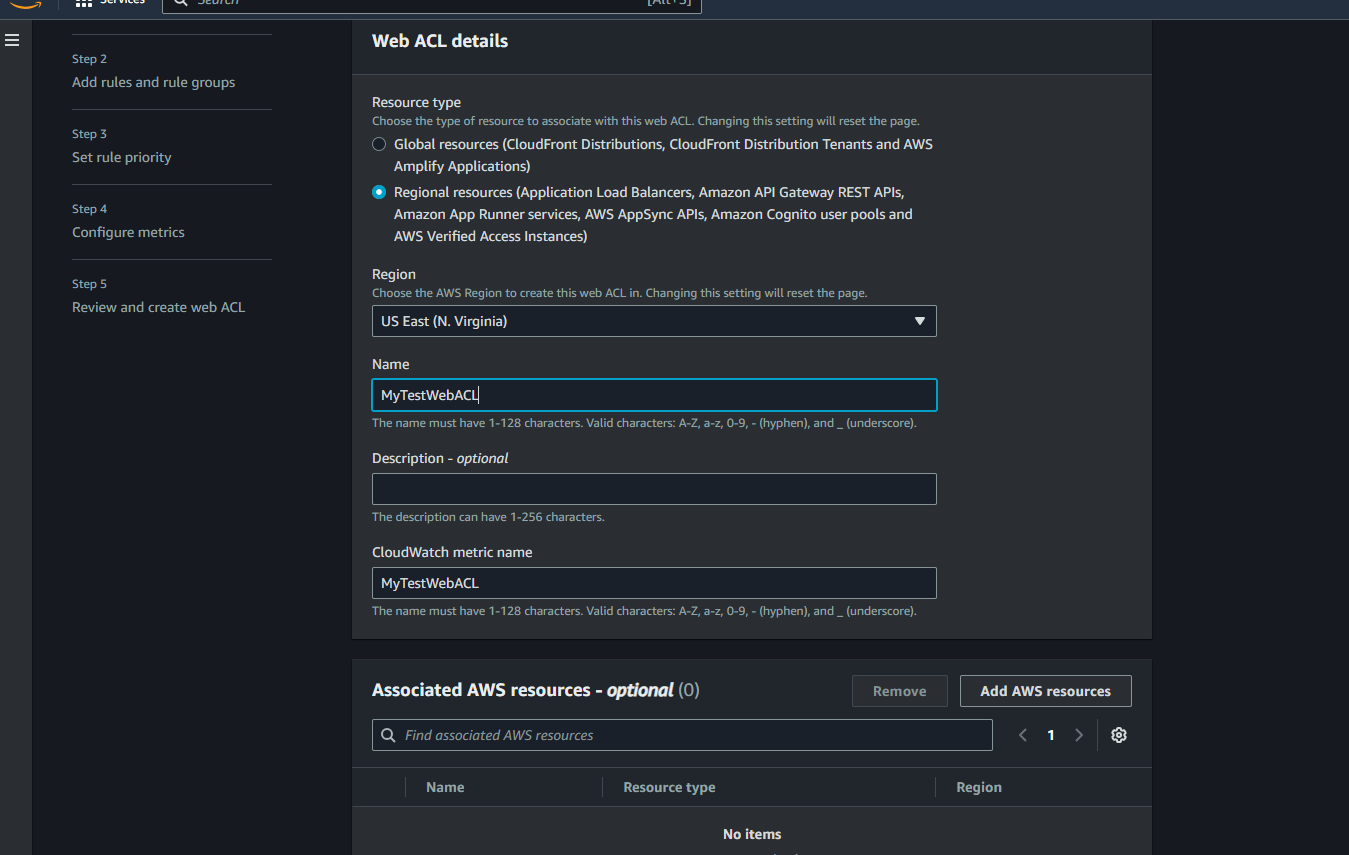
→ Added a rule using the IP Set to take action:

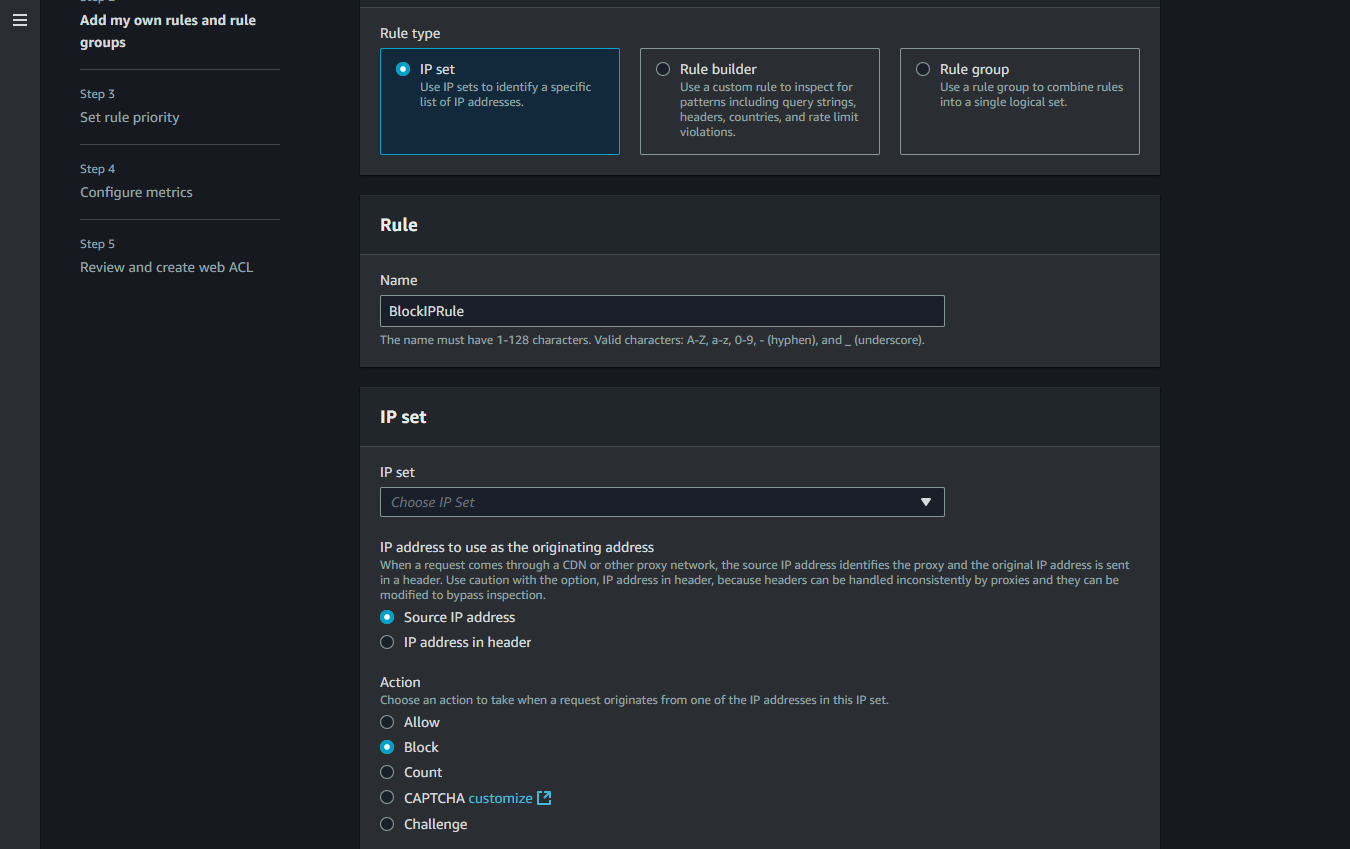
Allow, Block, or Count

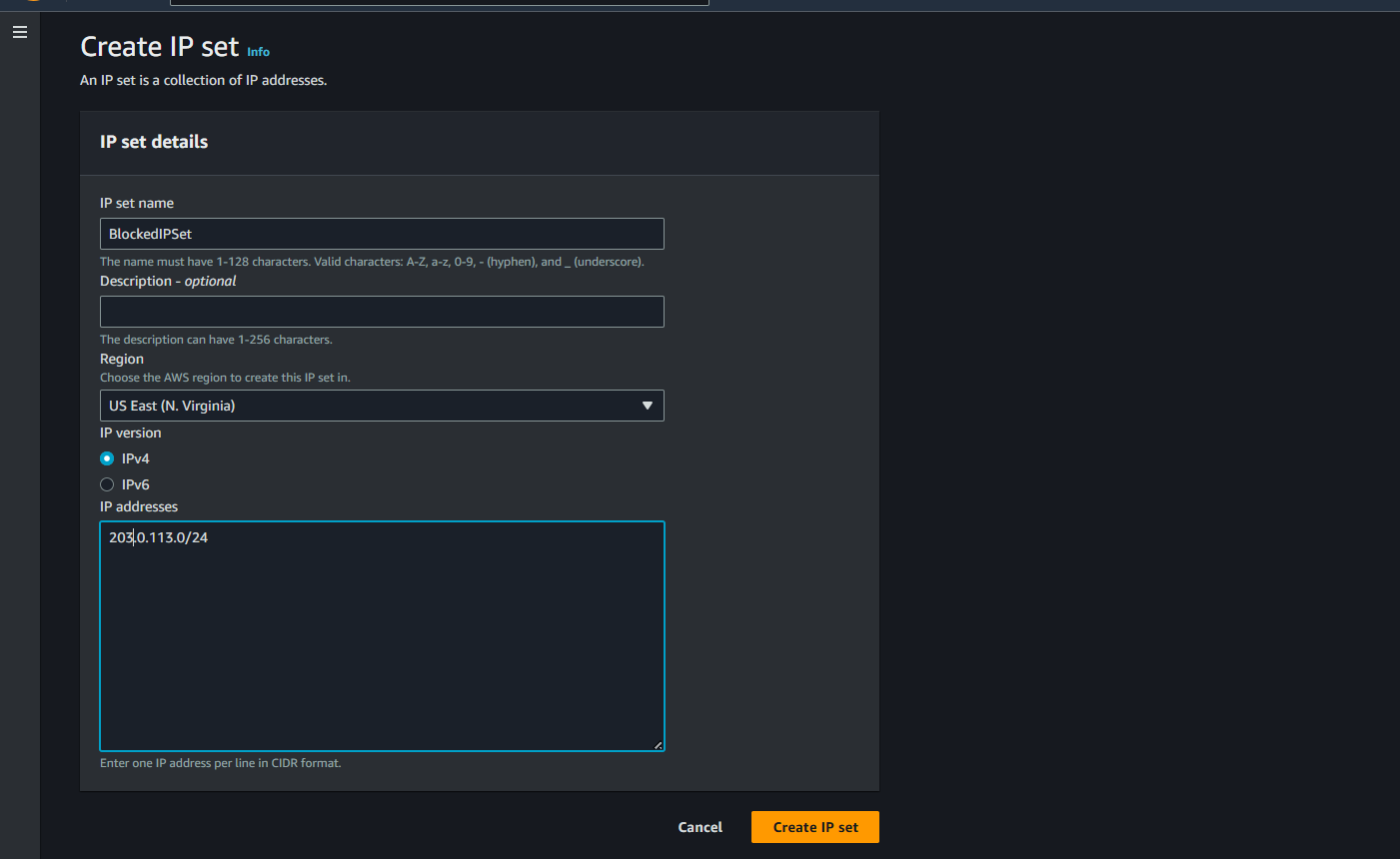
Configured Default Action

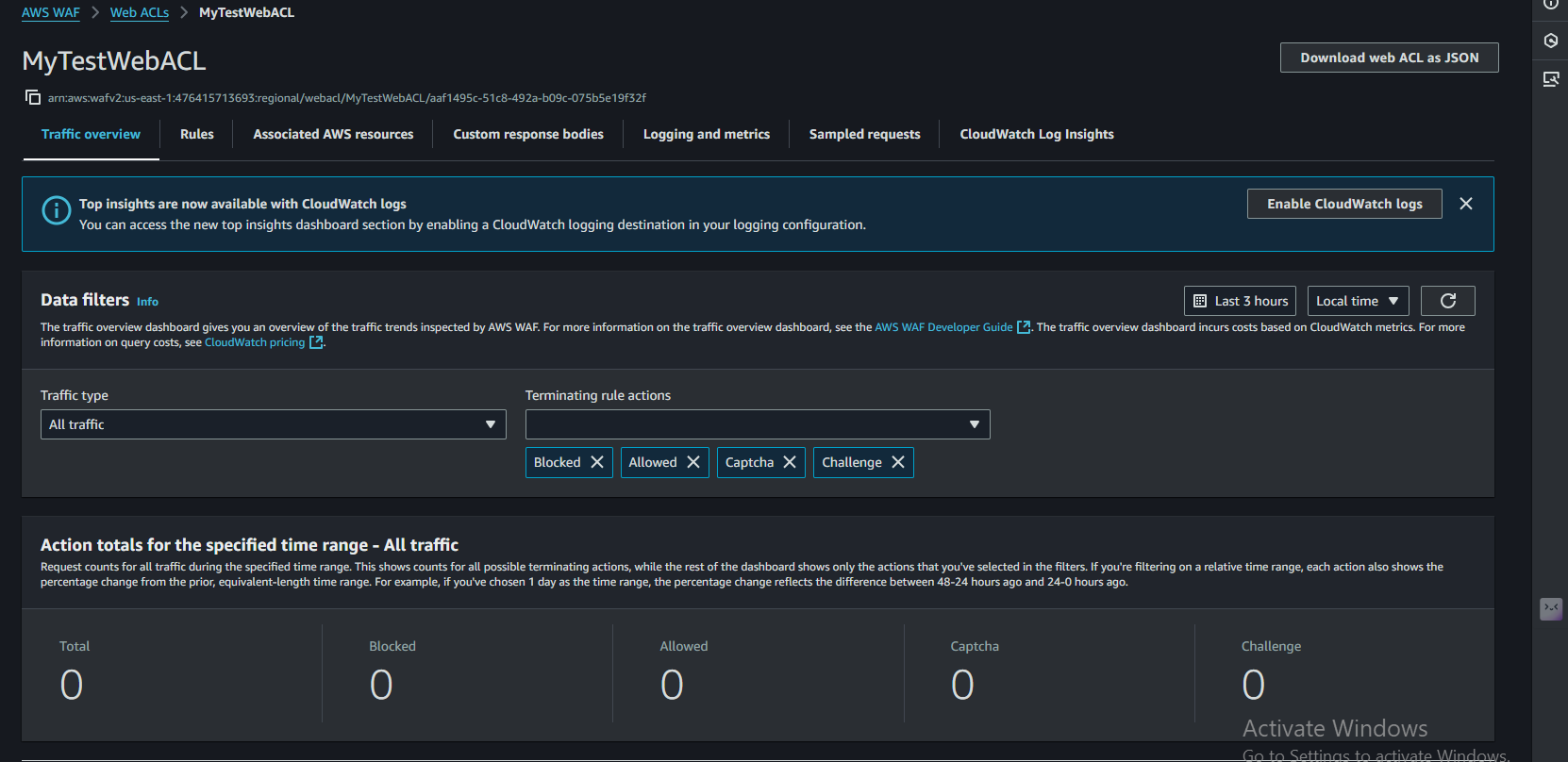
→ Defined what happens if a request doesn’t match any rule (usually Allow or Block).

Saved and Activated the Web ACL









AWS SYSTEM MANAGER PARAMETER STORE:

Accessed Parameter Store in AWS Console

Navigated to AWS Systems Manager > Parameter Store.

Created a New Parameter

Parameter Name: /MyApp/DBPassword

Parameter Type: SecureString (encrypted)

Parameter Value: MyS3cureP@ssw0rd

Description: Added description "Database password for MyApp"

Saved the Parameter

Successfully stored the parameter securely.

Verified the Parameter

Viewed the parameter details including:

ARN

Type (SecureString)

Encrypted value (decrypted view available)

Version and modification details

