**Lambda Function in AWS**

A Course End Project Submitted in Fulfillment of the Requirements

for the Course of

**A7514- CLOUD COMPUTING AND VIRTUALIZATION**

In

# Department of Computer Science and Engineering

**Submitted By**

|  |  |
| --- | --- |
| **Name of the Student** | **Roll Number** |
| KrishnaVarshita Reddy | 21881A05F8 |

|  |
| --- |
| **COURSE END PROJECT** |

**VARDHAMAN COLLEGE OF ENGINEERING**

**(AUTONOMOUS)**

Affiliated to **JNTUH**, Approved by **AICTE**, Accredited by **NAAC** with **A++** Grade, **ISO 9001:2015** Certified

Kacharam, Shamshabad, Hyderabad – 501218, Telangana, India

**January 2024**

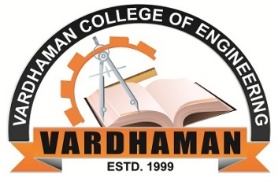
**VARDHAMAN COLLEGE OF ENGINEERING**

**(AUTONOMOUS)**

Affiliated to **JNTUH**, Approved by **AICTE**, Accredited by **NAAC** with **A++** Grade, **ISO 9001:2015** Certified

Kacharam, Shamshabad, Hyderabad – 501218, Telangana, India

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

Certified that this is a bonafide record of the course end project work entitled, **“Lambda function in AWS”**, done by, **K.KrishnaVarshita Reddy (21881A05F8)** submitted to the faculty of **Computer Science and Engineering**, in partial fulfillment of the requirements for the course of **Cloud Computing and Virtualization** during the year 2023-2024 (VI Semester).

Semester End Examination held on ……………………………………………**­­­­­­­­­­**

**Course Instructor: Head of Department:**

**Dr. U. Seshadri Dr. Ramesh Karnati**

**(Professor)**  **(HOD, CSE)**

**Abstract**

AWS Lambda is a serverless compute service offered by Amazon Web Services (AWS) that enables users to run code without provisioning or managing servers. AWS Lambda allows developers to upload their code in the form of functions and execute them in response to events such as changes in data, HTTP requests, or system state changes. The service automatically scales the infrastructure to accommodate the workload, providing high availability and reliability.

Key features of AWS Lambda include support for multiple programming languages, seamless integration with other AWS services, pay-per-use pricing model, and fine-grained access control through AWS Identity and Access Management (IAM).Common use cases for AWS Lambda include real-time file processing, data processing, IoT applications, event-driven microservices, and backend tasks for web and mobile applications.

**List of Contents**

**Abstract**

**CHAPTER 1 Description**

**CHAPTER 2** **Steps**

**CHAPTER 3 Conclusions**

**CHAPTER 4** **REFERENCES**

**CHAPTER 1**

**DESCRIPTION**

* 1. **Project Description:**

In this project, we aim to build a serverless image processing pipeline using AWS Lambda. The pipeline will allow users to upload images to an S3 bucket, trigger Lambda functions to process the images, and store the processed results back into another S3 bucket.

**Goal and Advantages:**

Scalability: AWS Lambda automatically scales based on the incoming workload, ensuring consistent performance even during peak times.

Cost-Effectiveness: With the pay-per-use pricing model, users only pay for the compute time consumed by Lambda functions, resulting in cost savings.

Serverless Architecture: Eliminates the need for managing servers, allowing developers to focus on application logic rather than infrastructure management.

Flexibility: Lambda supports multiple programming languages, enabling developers to use their preferred language for writing functions.

**Features**:

**AWS Lambda Functions:**

Image Processing Function: This Lambda function will be triggered whenever a new image is uploaded to the input S3 bucket. It will use image processing libraries like Pillow or OpenCV to perform operations such as resizing, cropping, or applying filters to the image.

Thumbnail Generation Function: Another Lambda function triggered by the upload event will generate thumbnails of the processed images to be displayed as previews.

**Amazon S3 Buckets:**

Input Bucket: This bucket will be used by users to upload their images.

Processed Image Bucket: The processed images will be stored in this bucket.

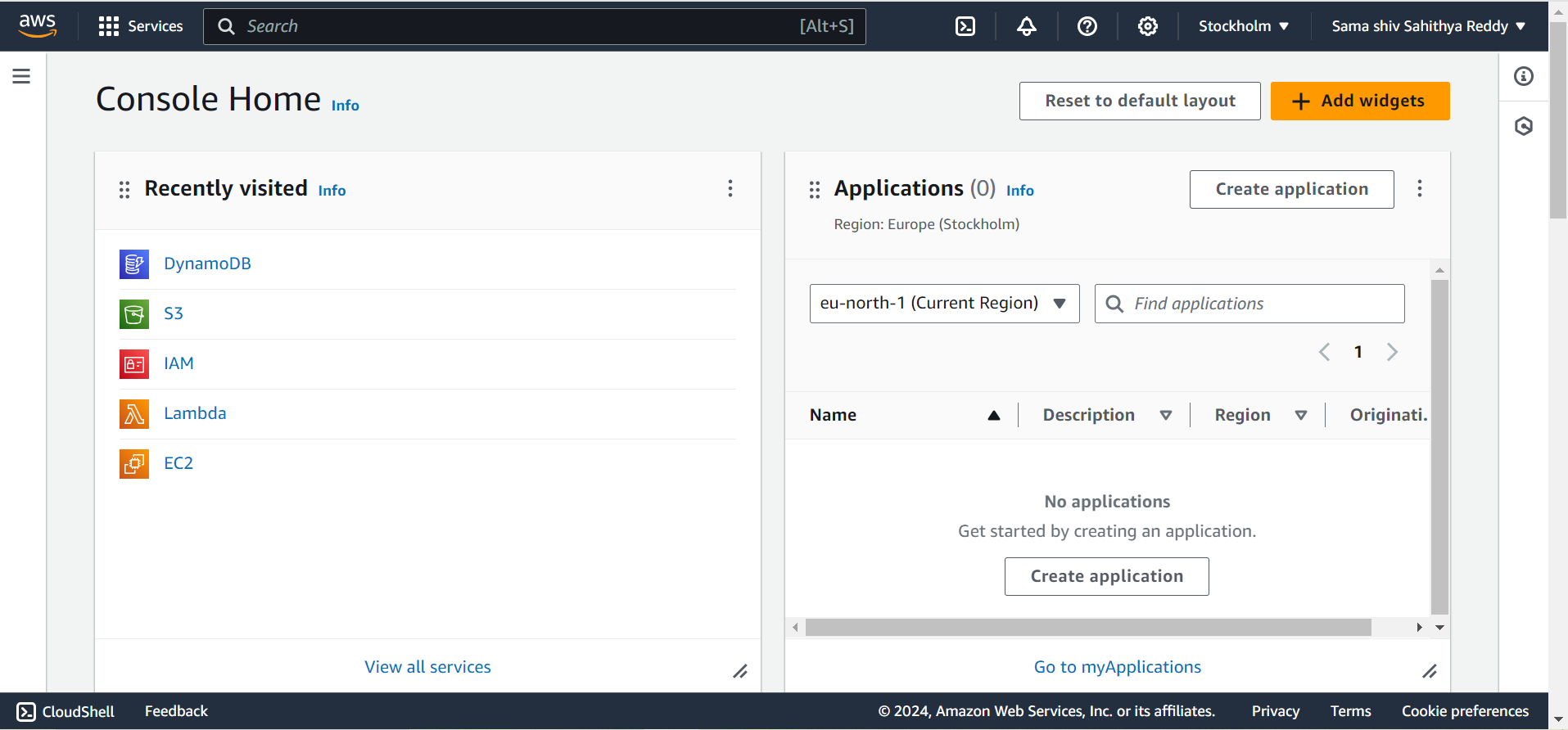
**AWS IAM (Identity and Access Management):**

IAM roles and policies will be defined to grant necessary permissions to Lambda functions to access S3 buckets, CloudWatch logs, and other AWS services.

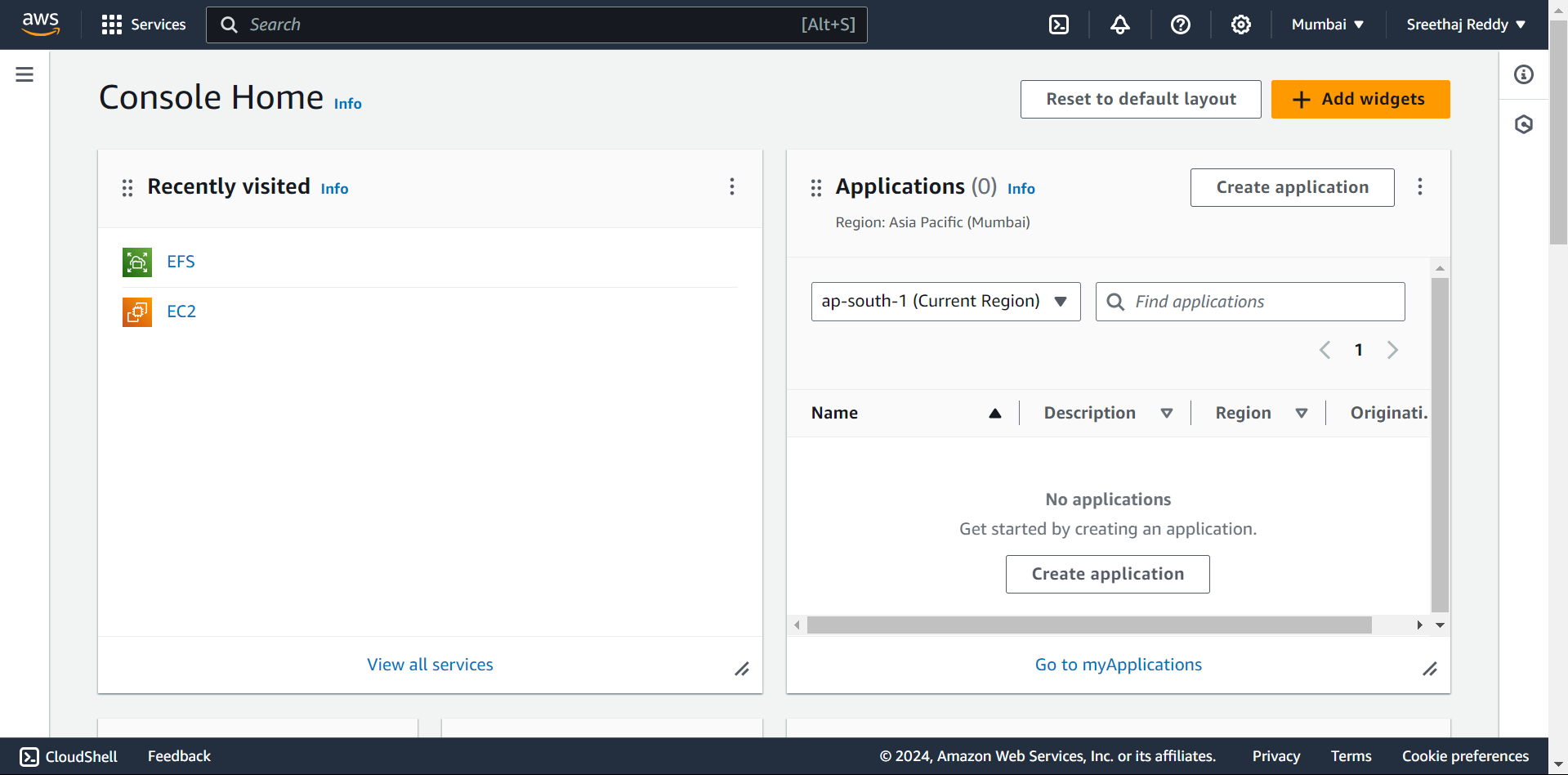
**CHAPTER 2**

**STEPS:**

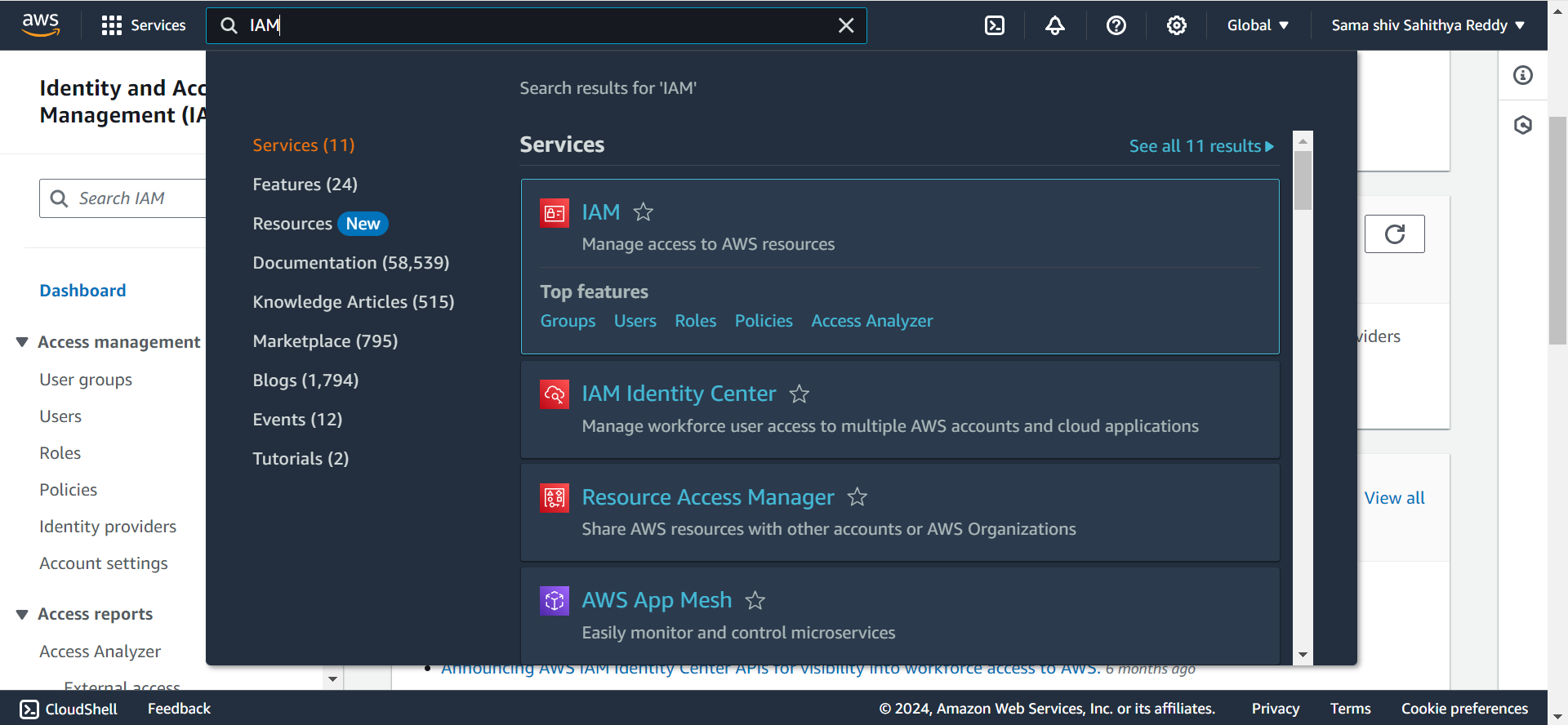
1.Open amazon web console form any webserver and then login into the aws management console .



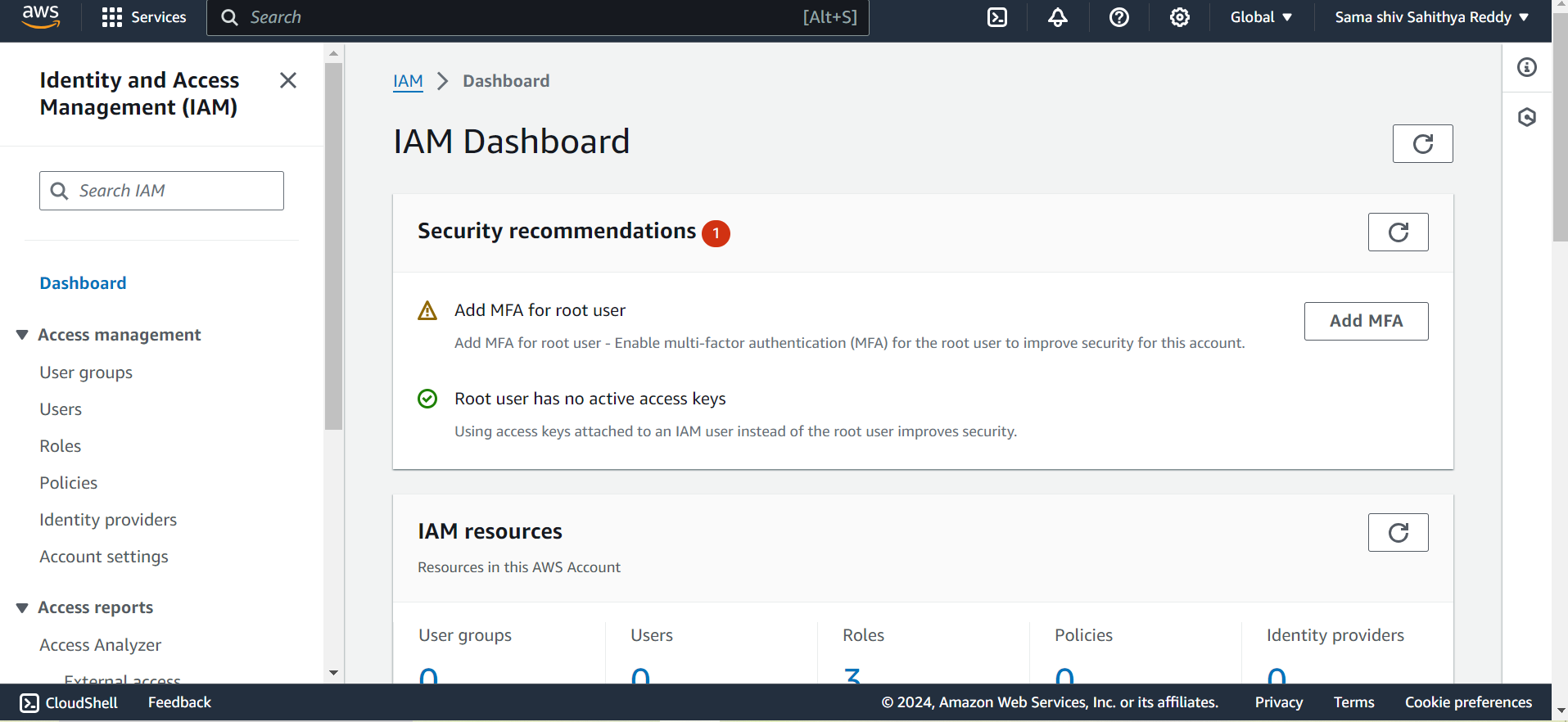
2.After login then set the server to Mumbai region



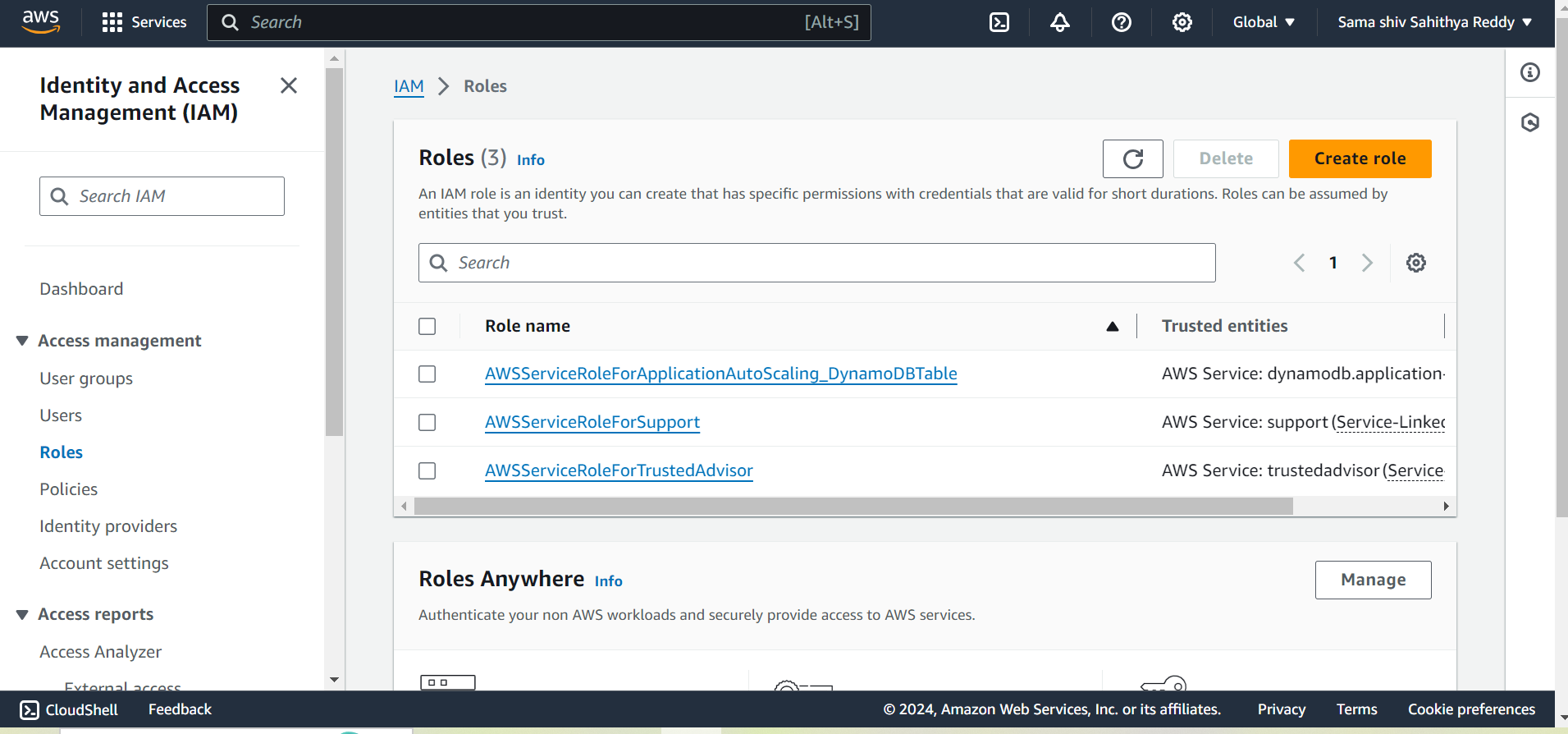
3.To open IAM service from services then we will be directed to services page select compute the we find IAM service.



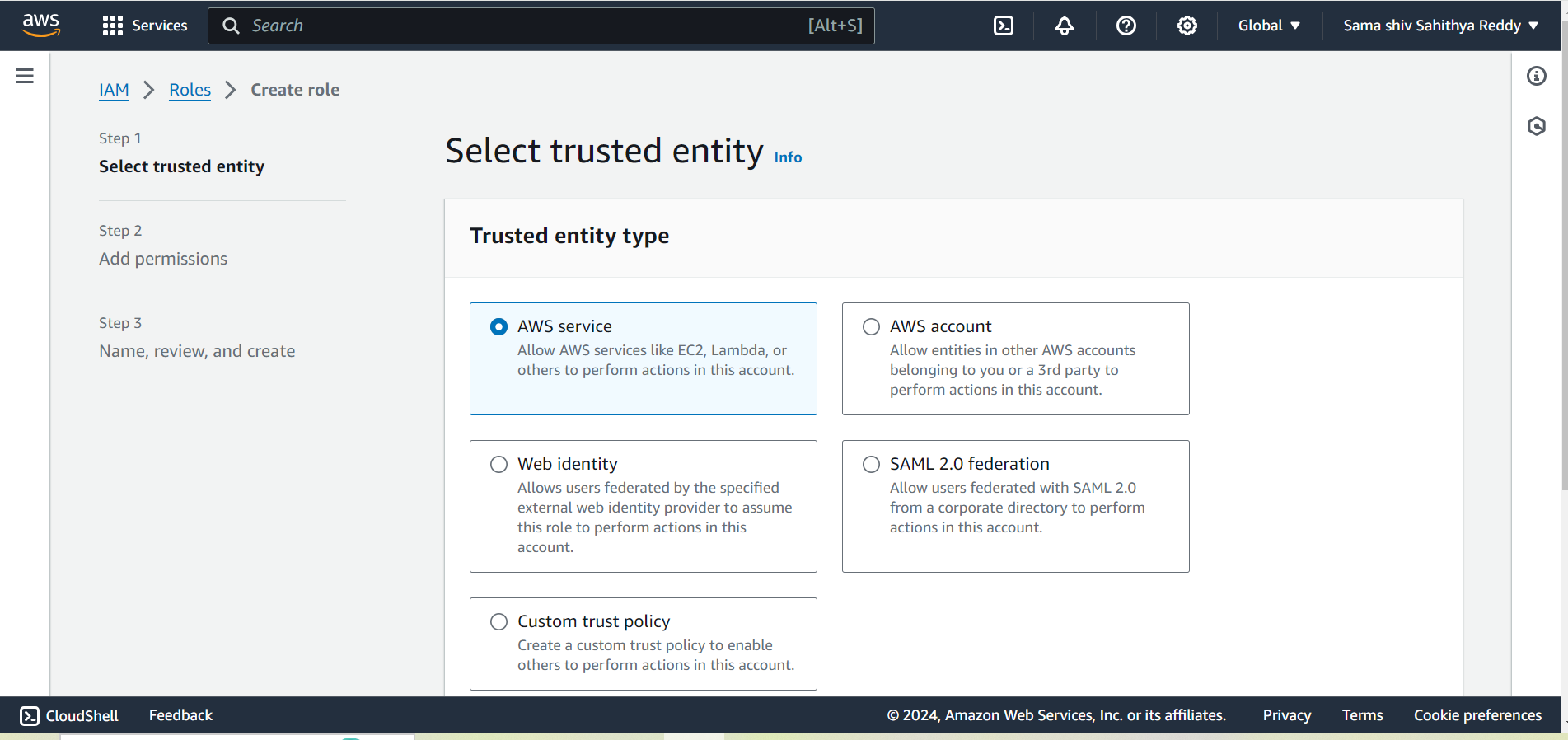
4.Select IAM service and then we will be redirected to IAM dashboard where we can see all our resources.



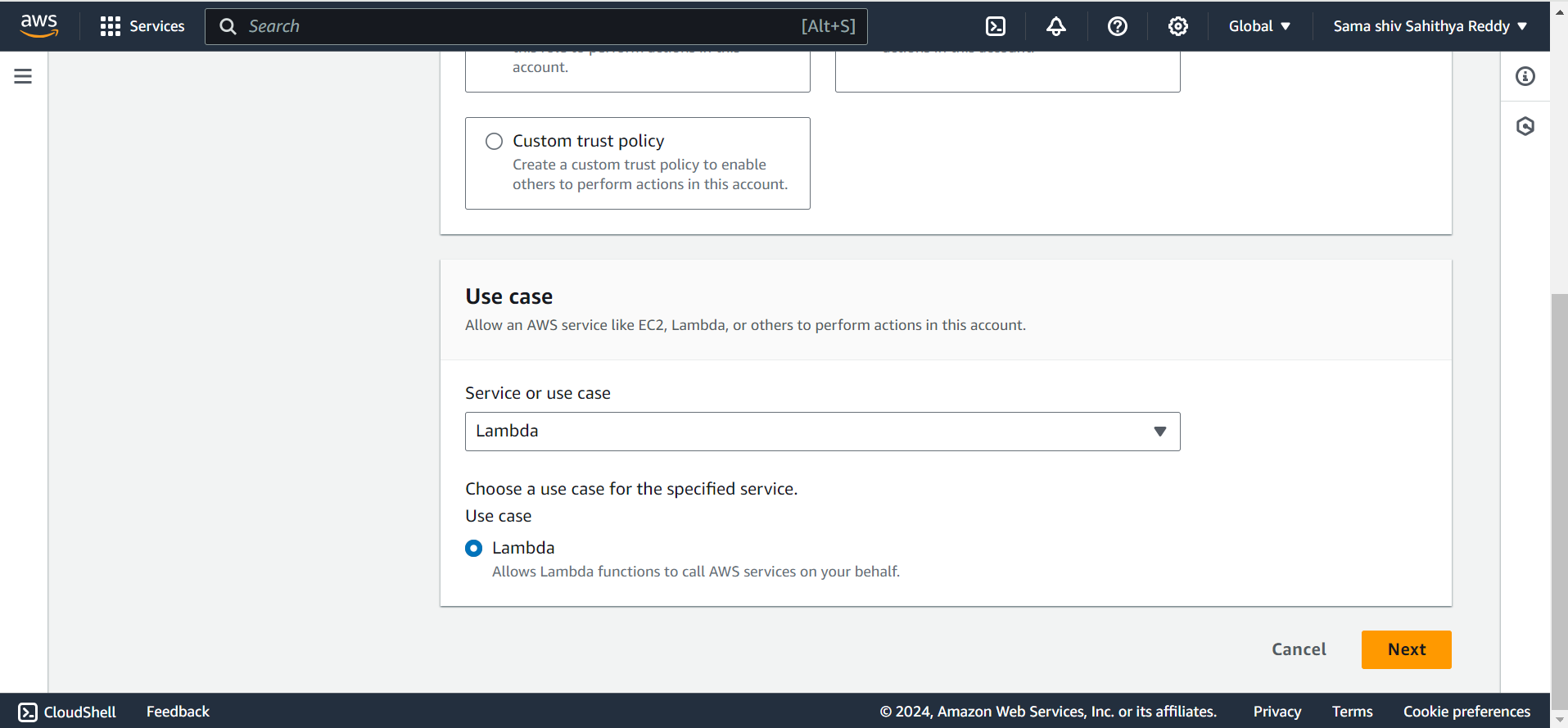
5.We need to select the roles on the left side to create the role .



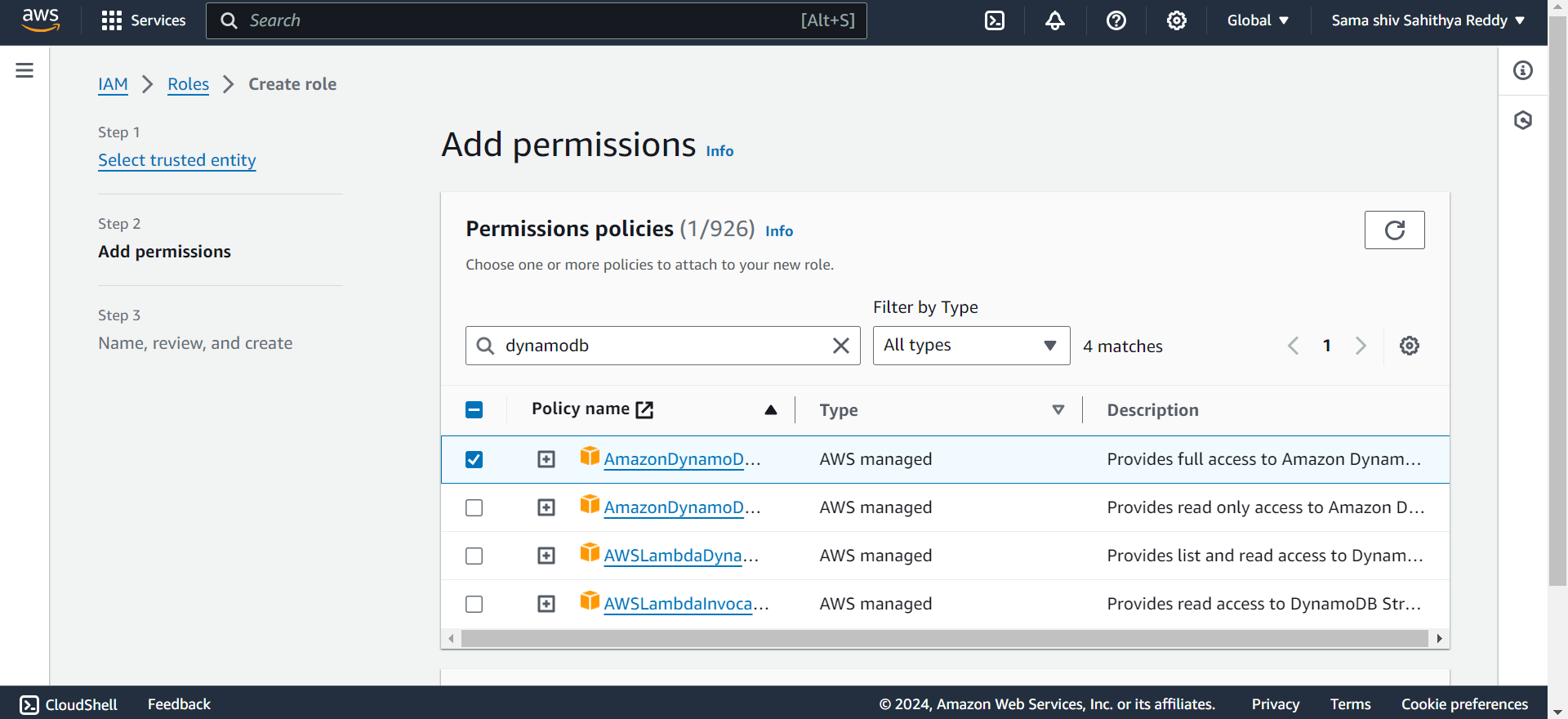
6. After clicking on roles , go to create role and select the options, and select the entity.



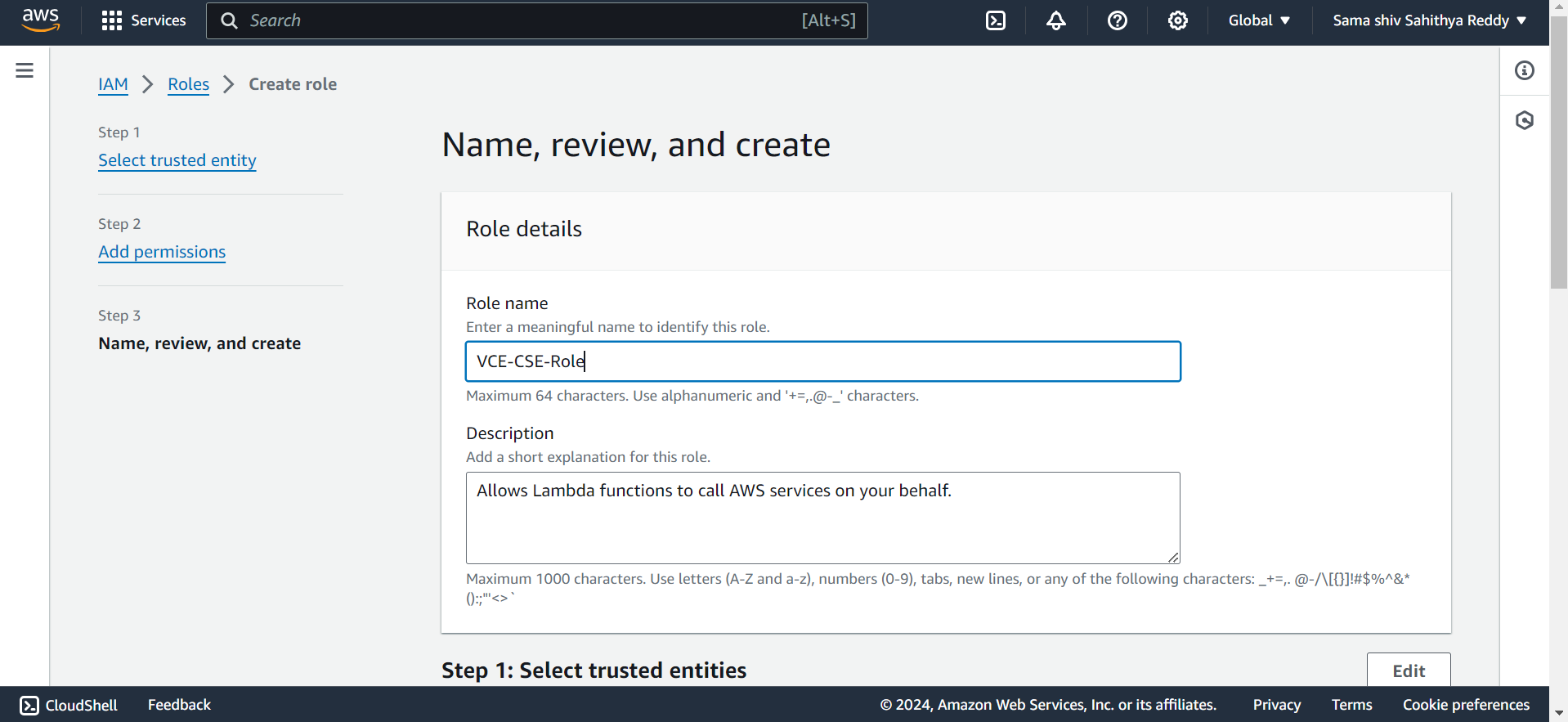
7. We need to select Lambda as the use case.



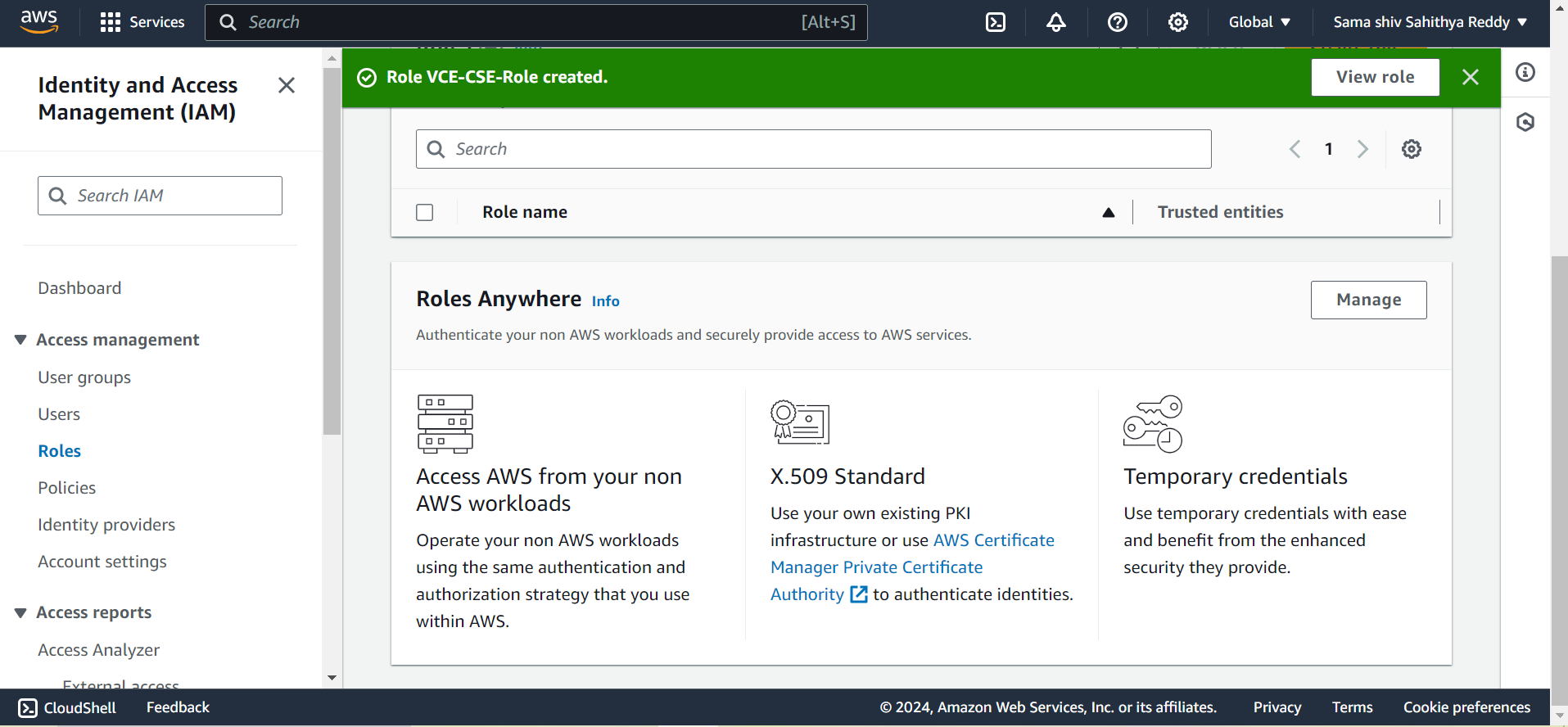
8. We need to choose next , and move to the add permissions and click on Dynamodb which has all the access .



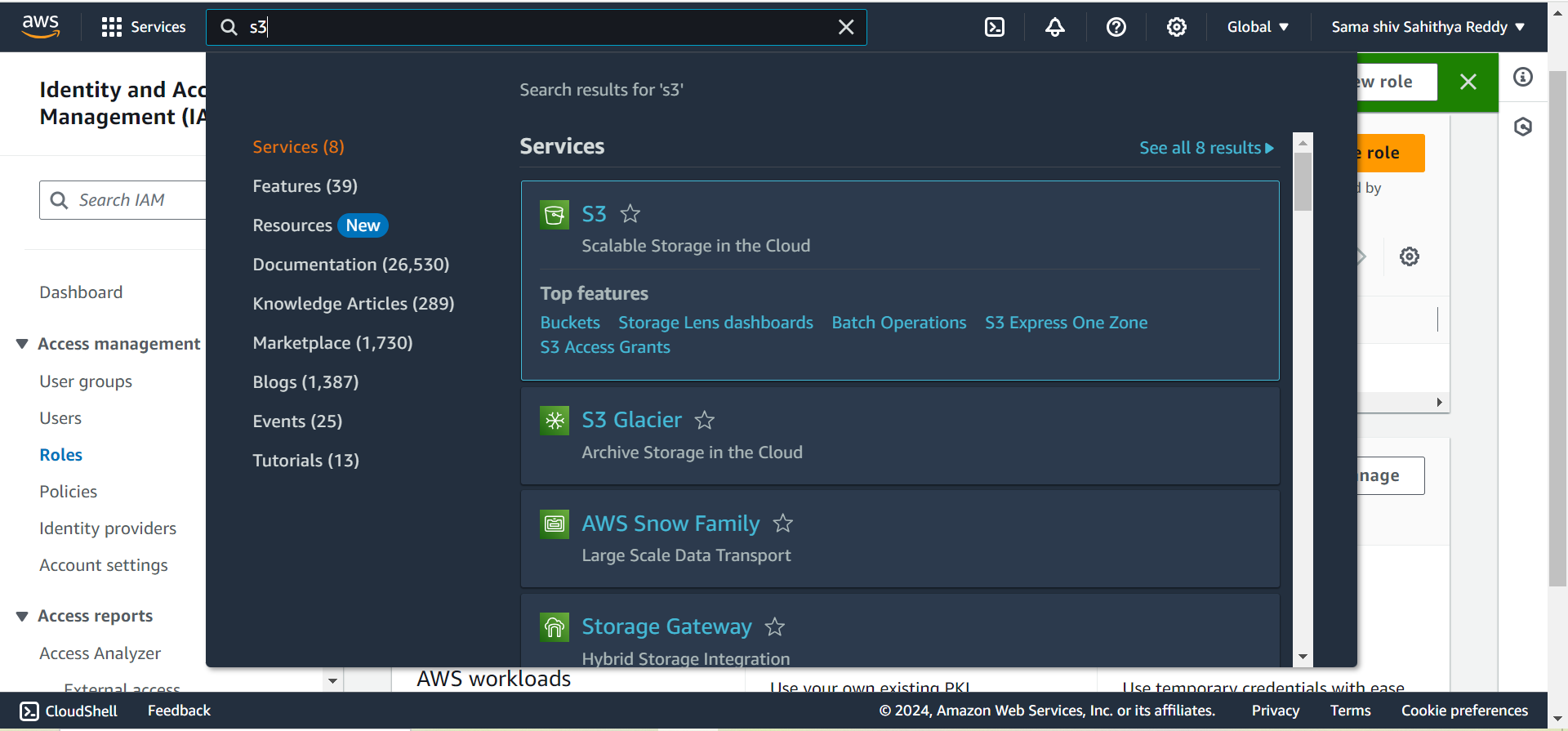
9. The next Step is to select next and give the name of the role .



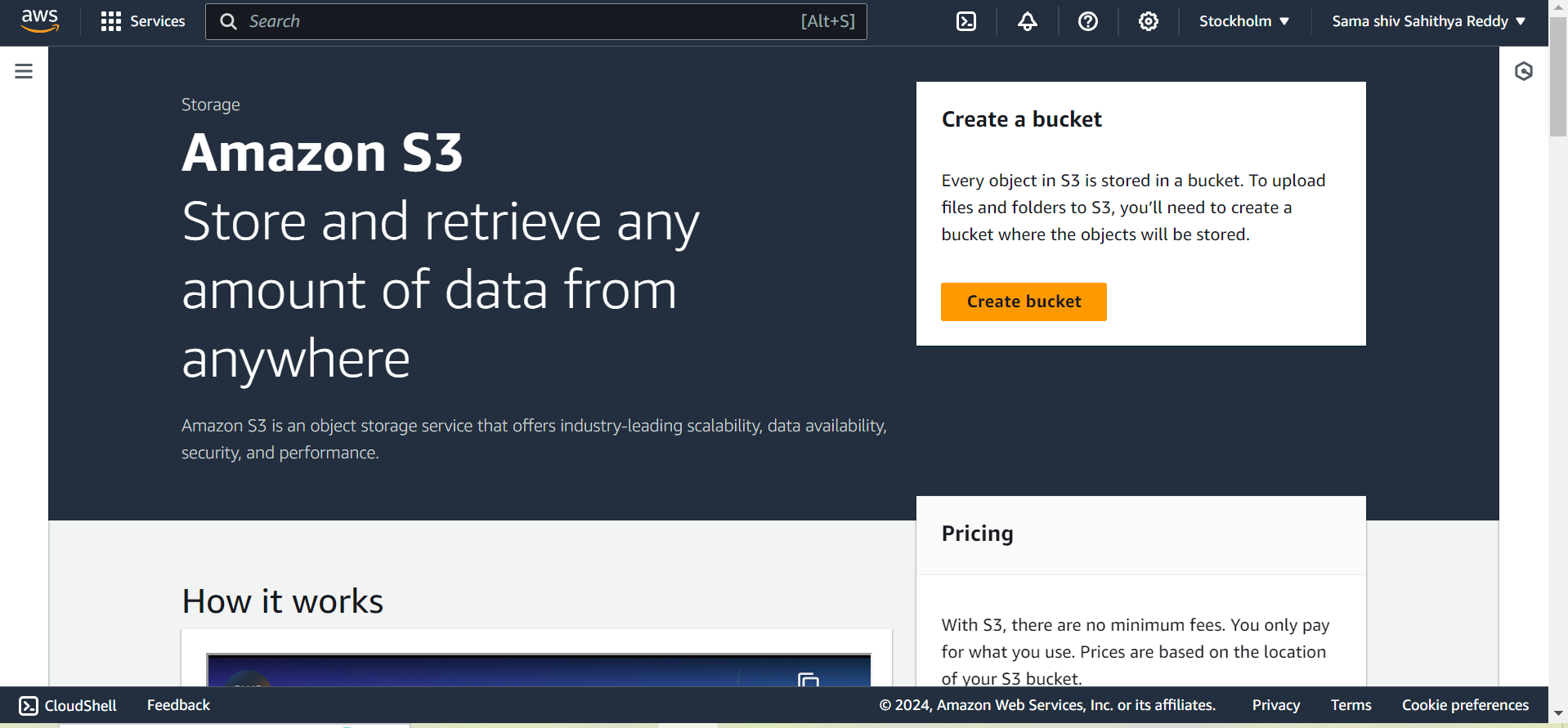
10. After giving the name , click next and now we successfully created the role.



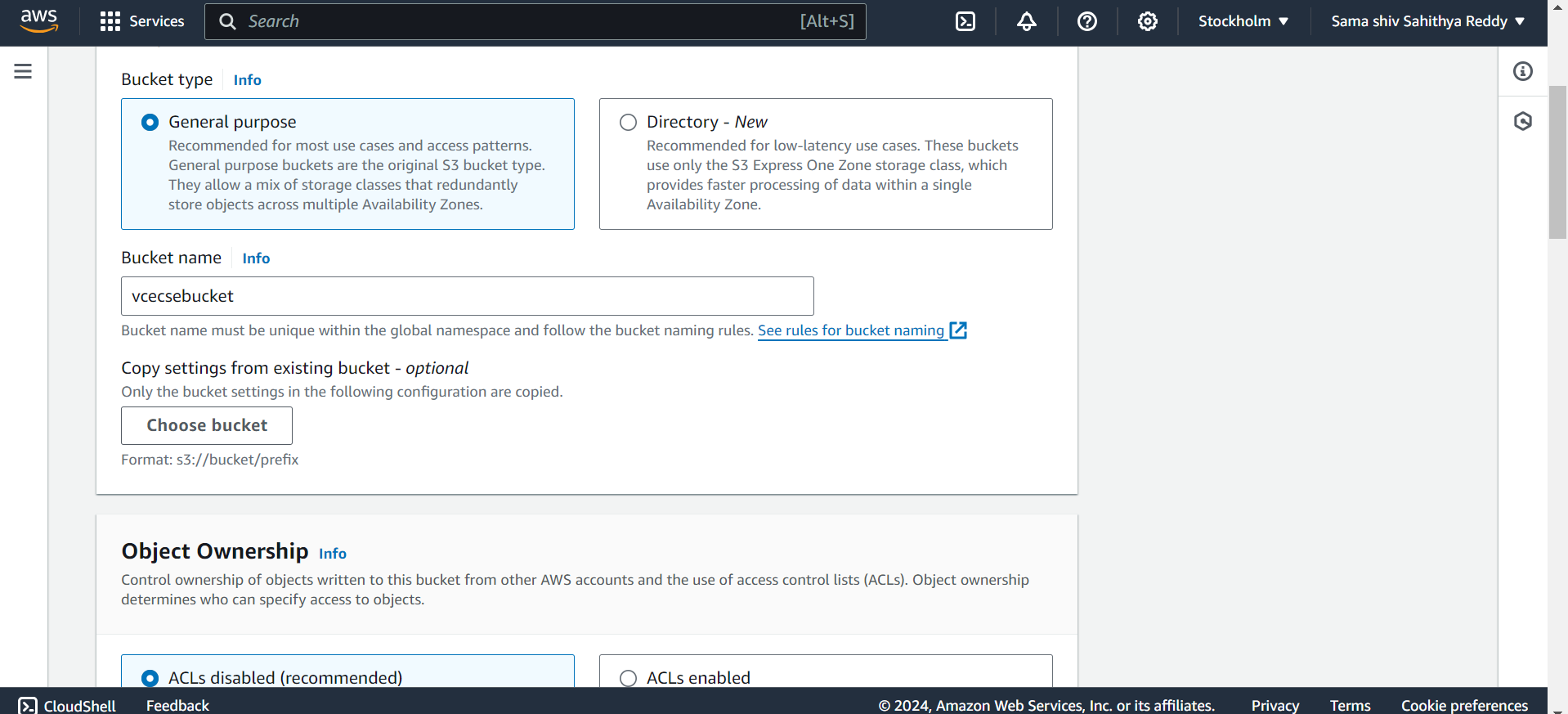
11.Now select the s3 services .



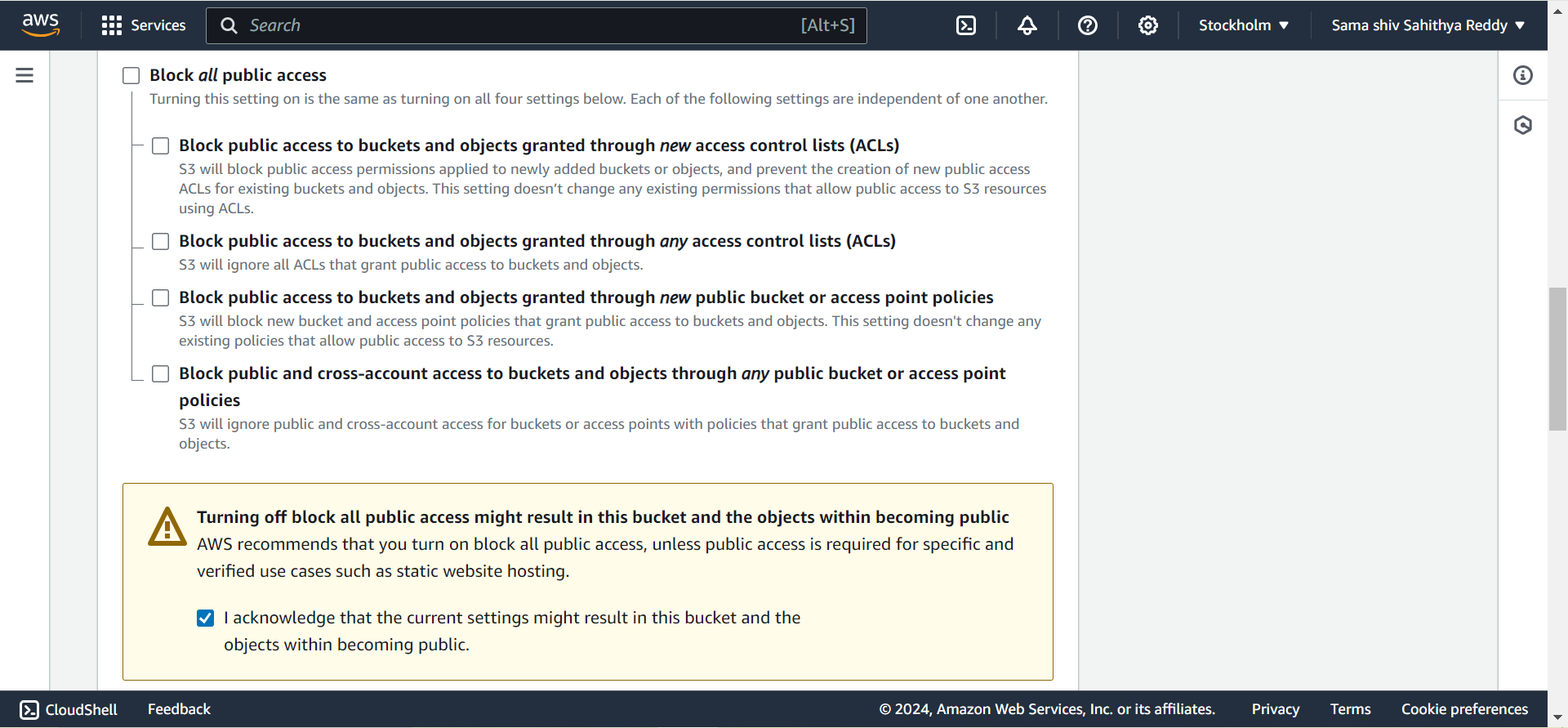
12. We need to go to the S3 dashboard and select the create bucket.



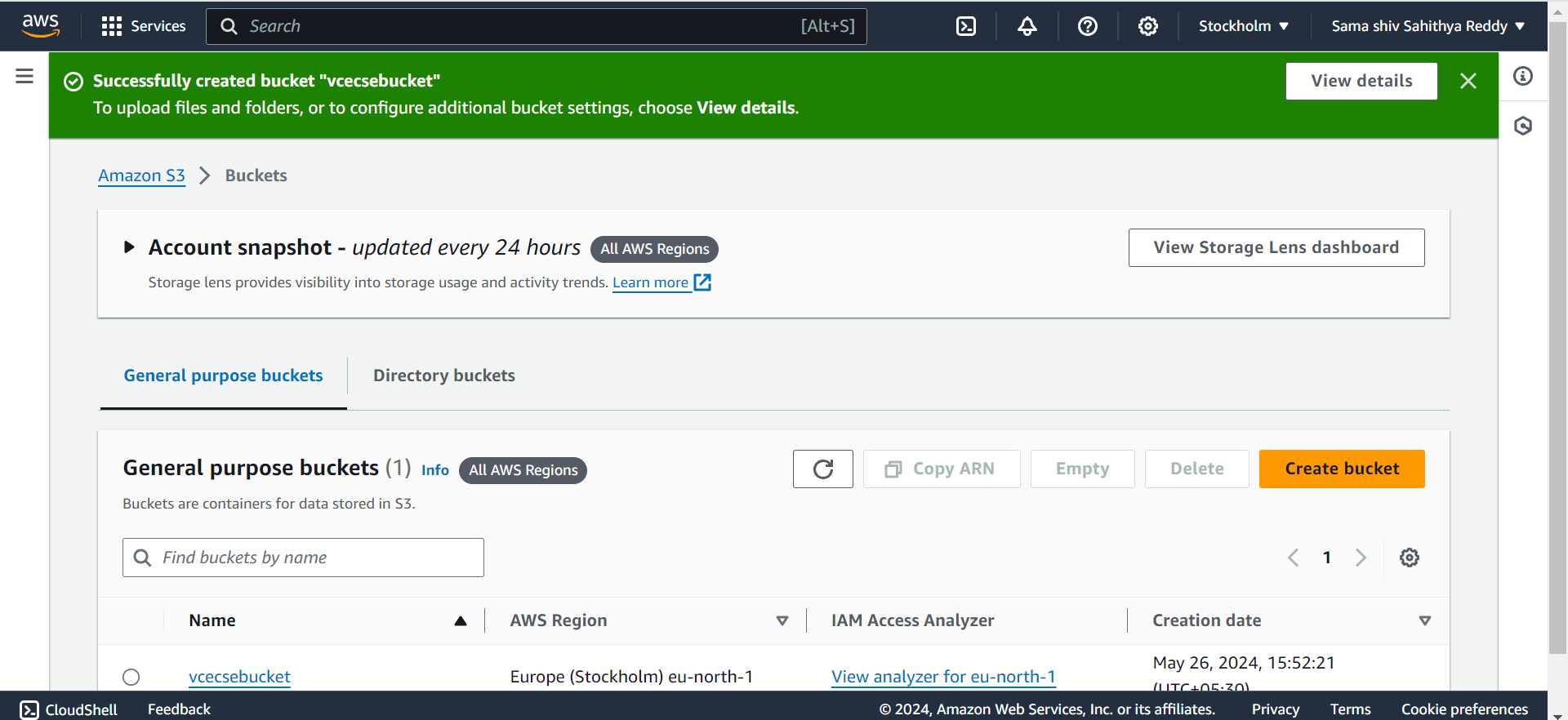
13. After that ,give the name for the bucket .



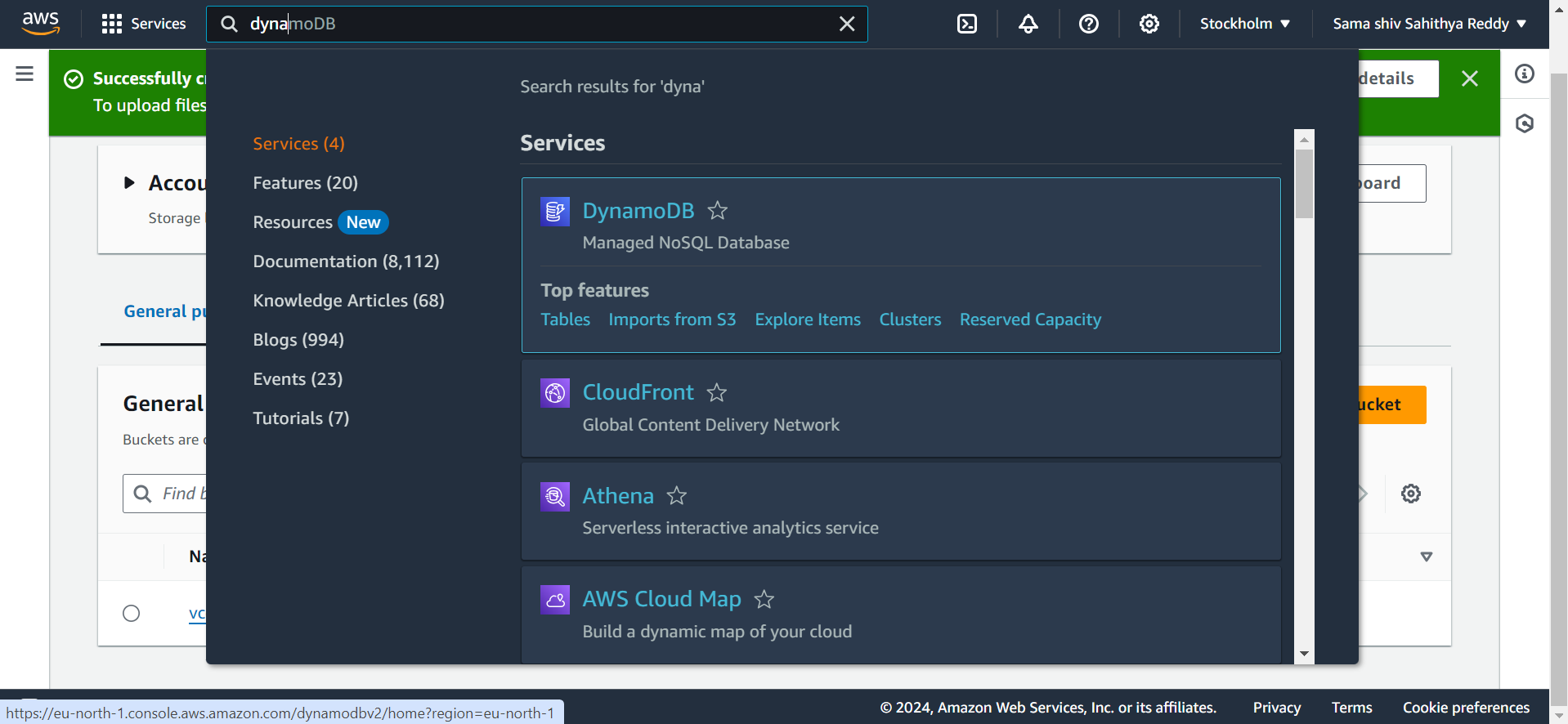
14. Then enable the permissons and click on create bucket.



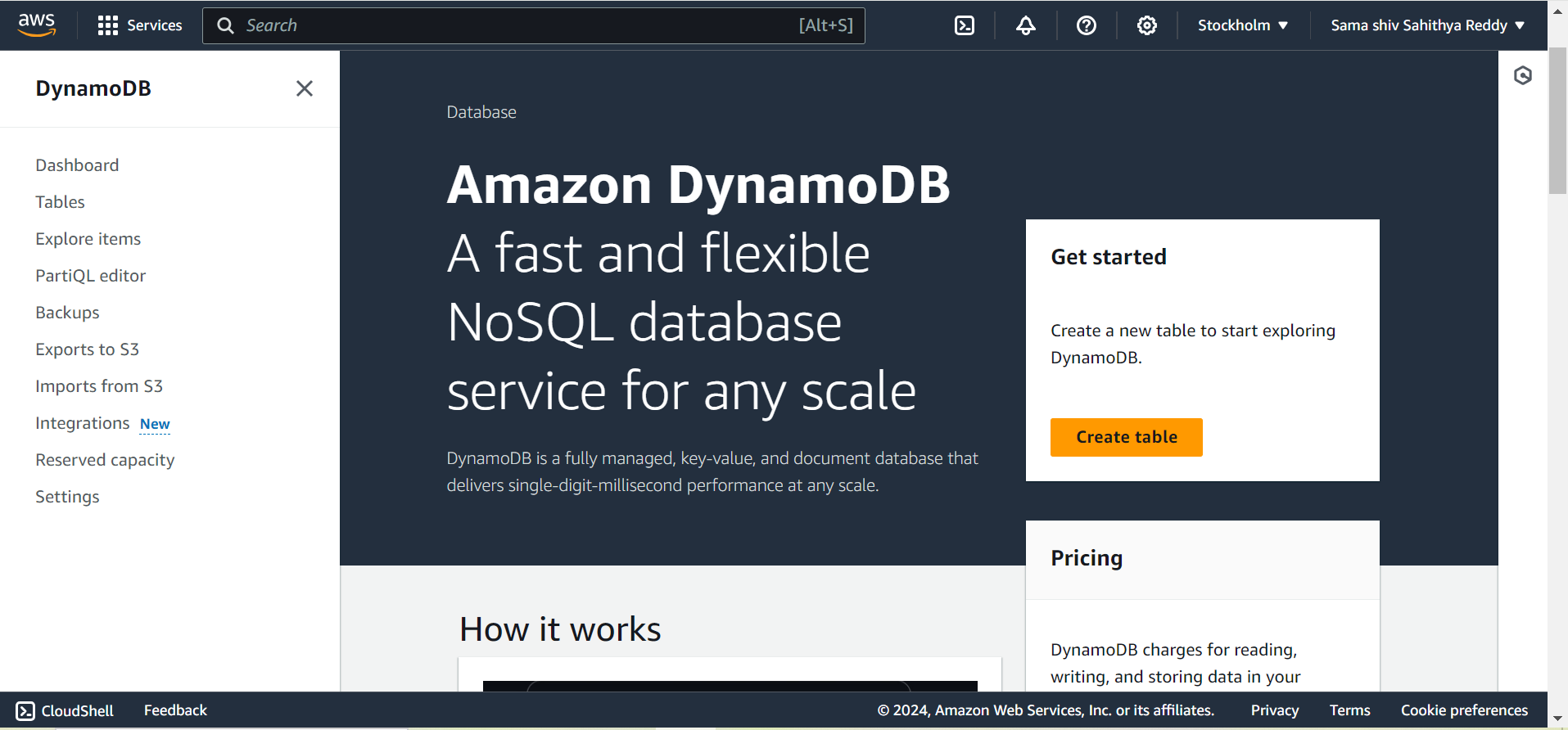
15**.** The bucket is created successfully in S3 service.



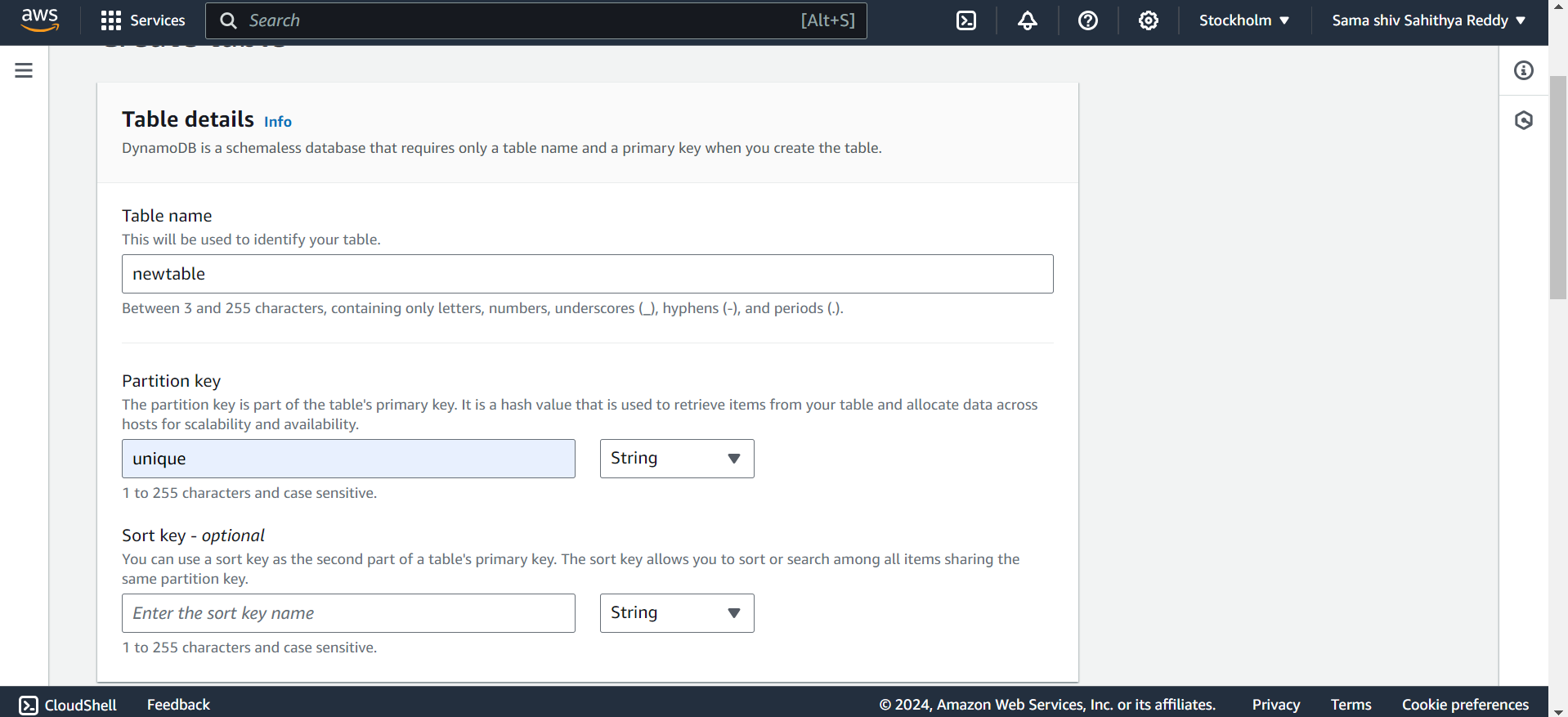
16.Now go to the sevices and search for DynamoDb.

****

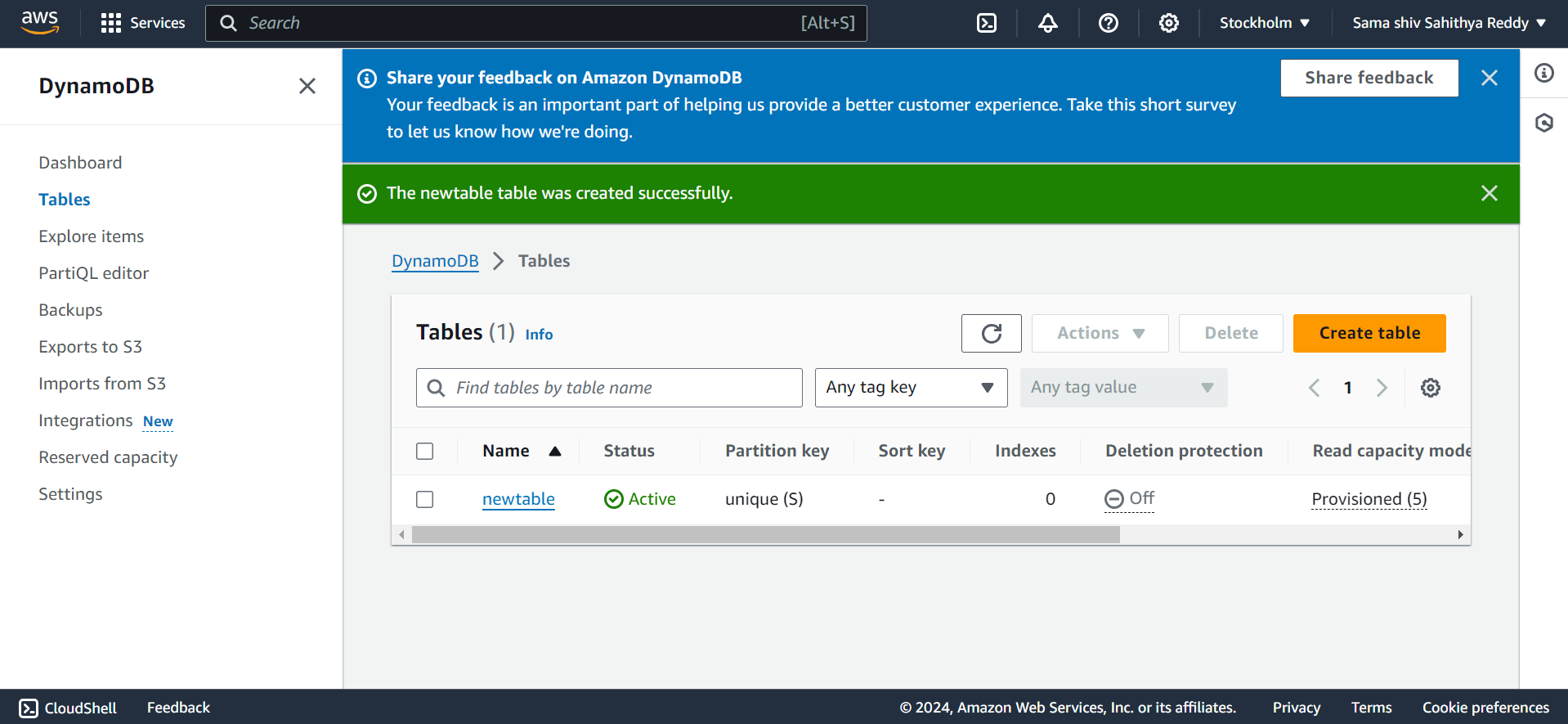
17. After that , go to the dashboard of Dyanamodb and click on create table.

****

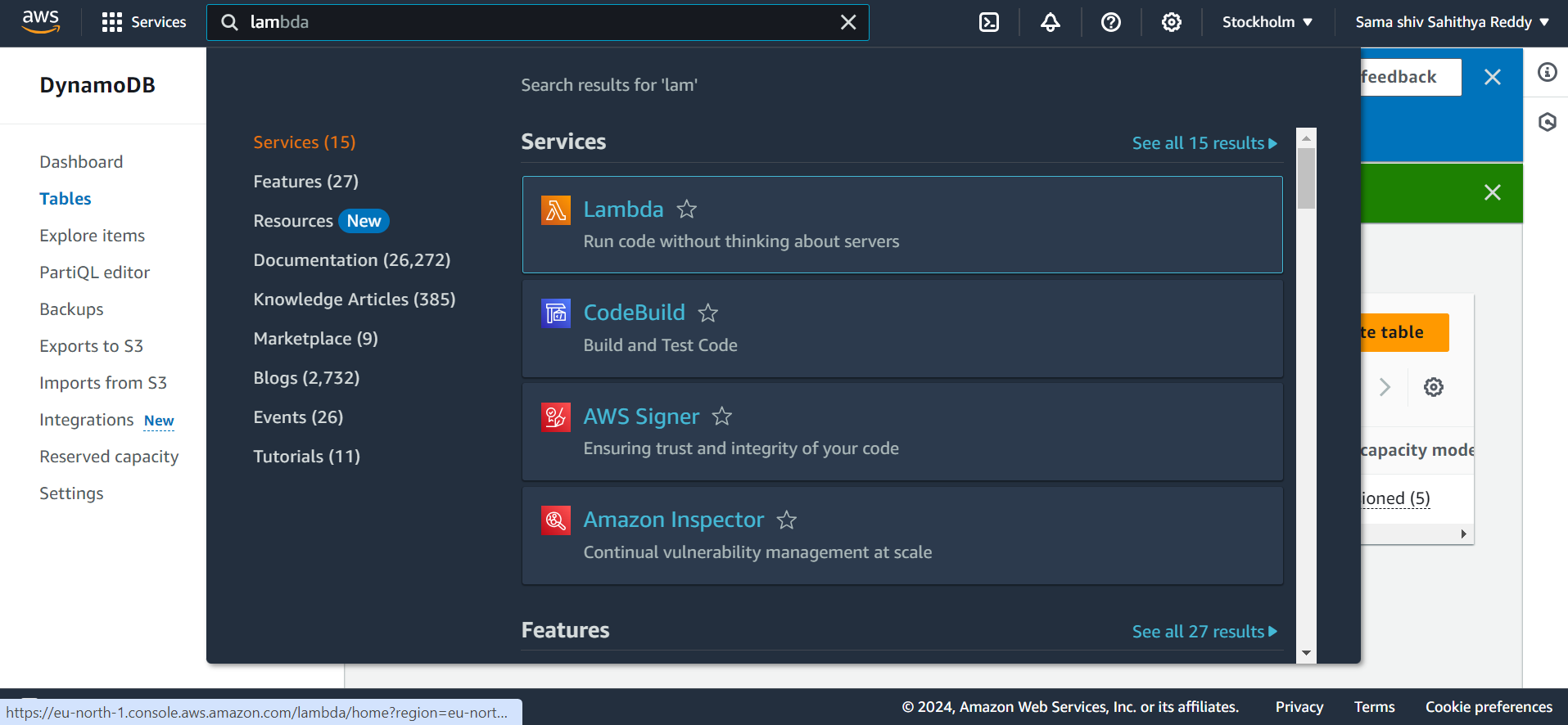
18. After that,give the name of the table as ” newtable” and partition key as “unique”.



19. Then the newtable has been created in the dynamodb.

****

20. The next step is go the services and search for lambda service.

****

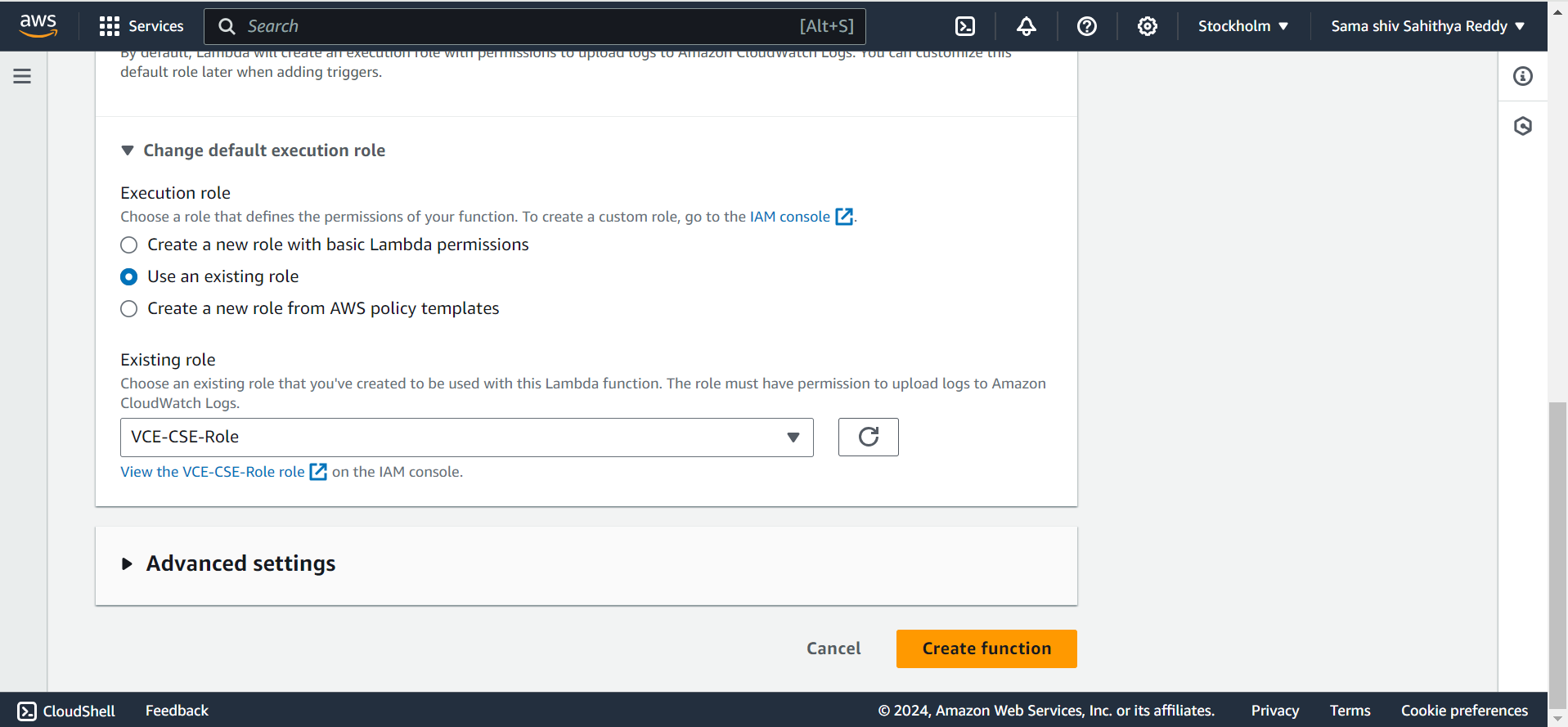
21. After that ,Go to the lambda dashboard and select create function.



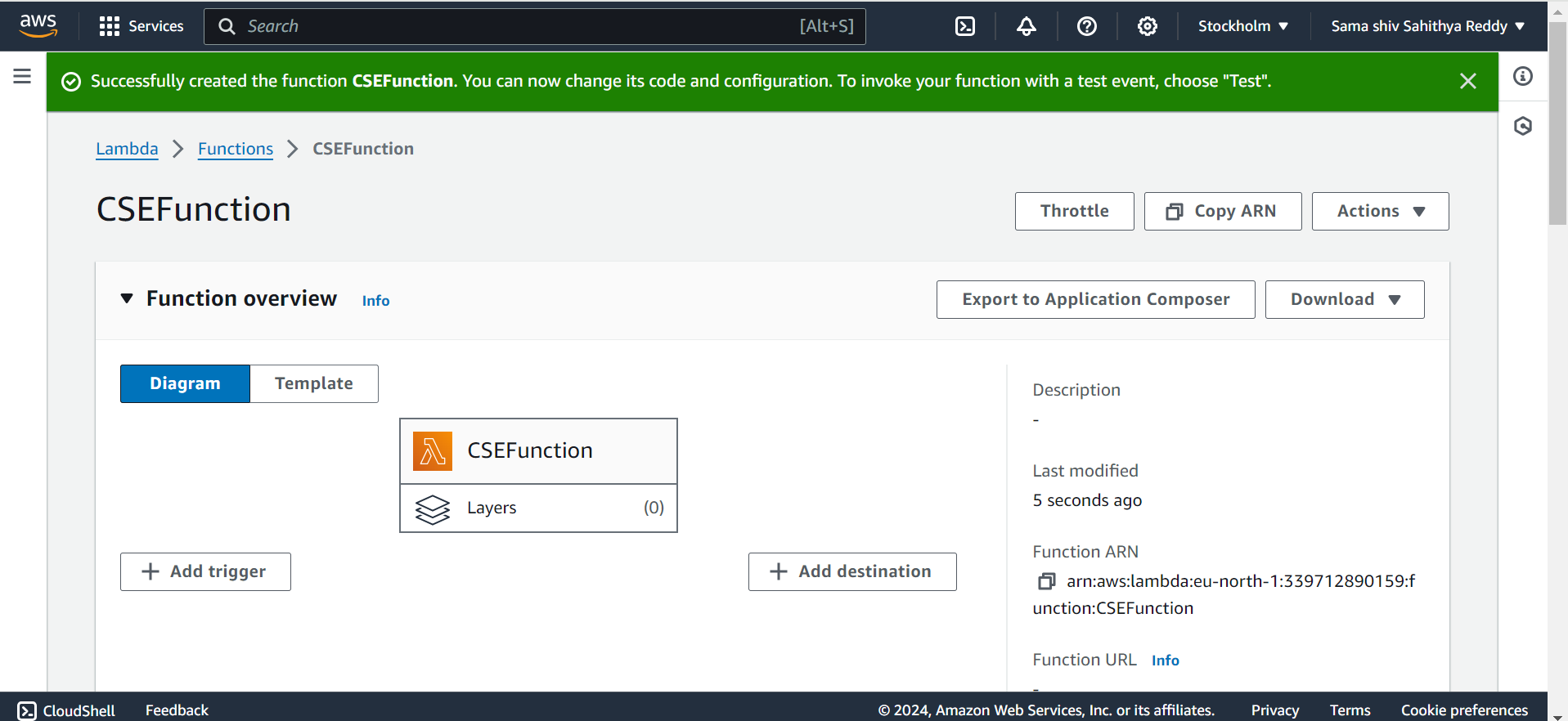
22. After that ,give the name of the function as “CSEFunction “ and change the runtime to python 3.7 and change the role to existing role as the role we created .



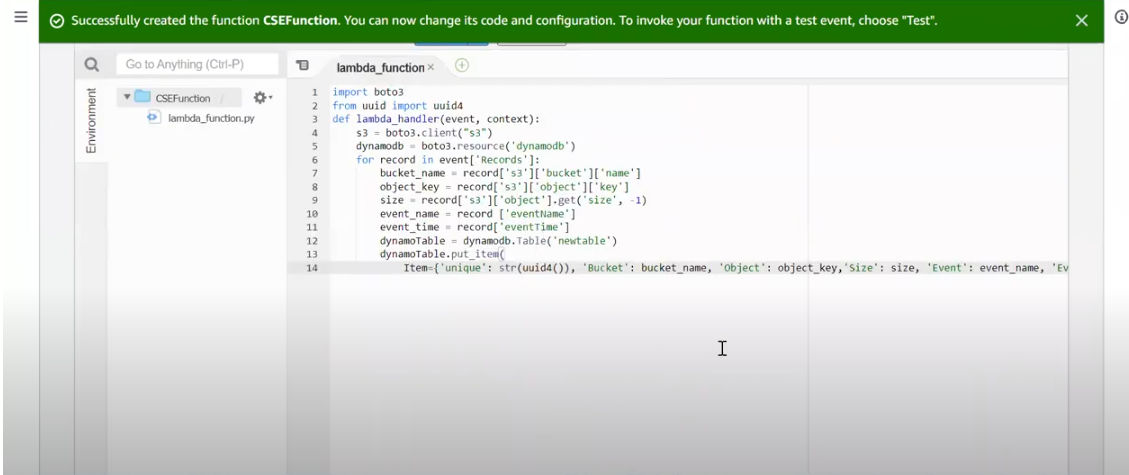
23. After that click on create function .

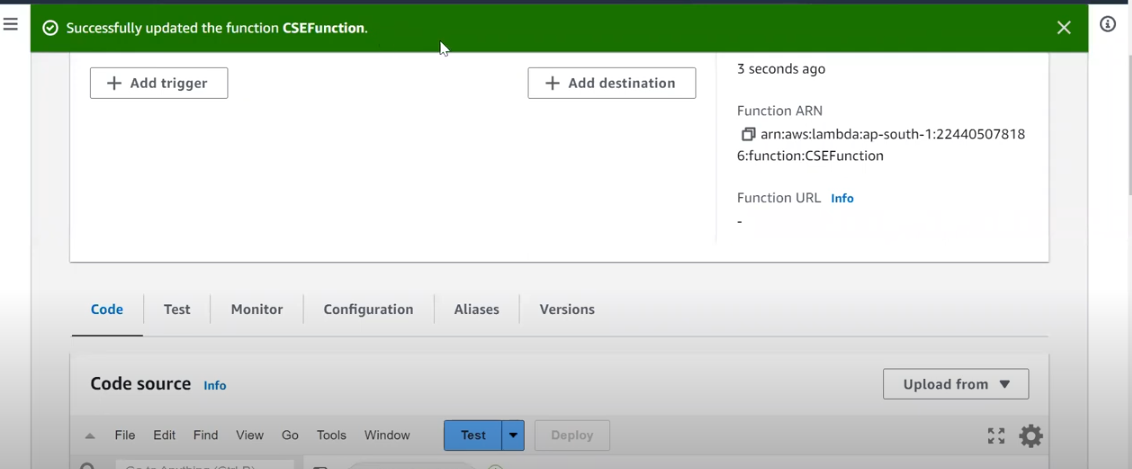
****

24. After that, the lambda function is created .

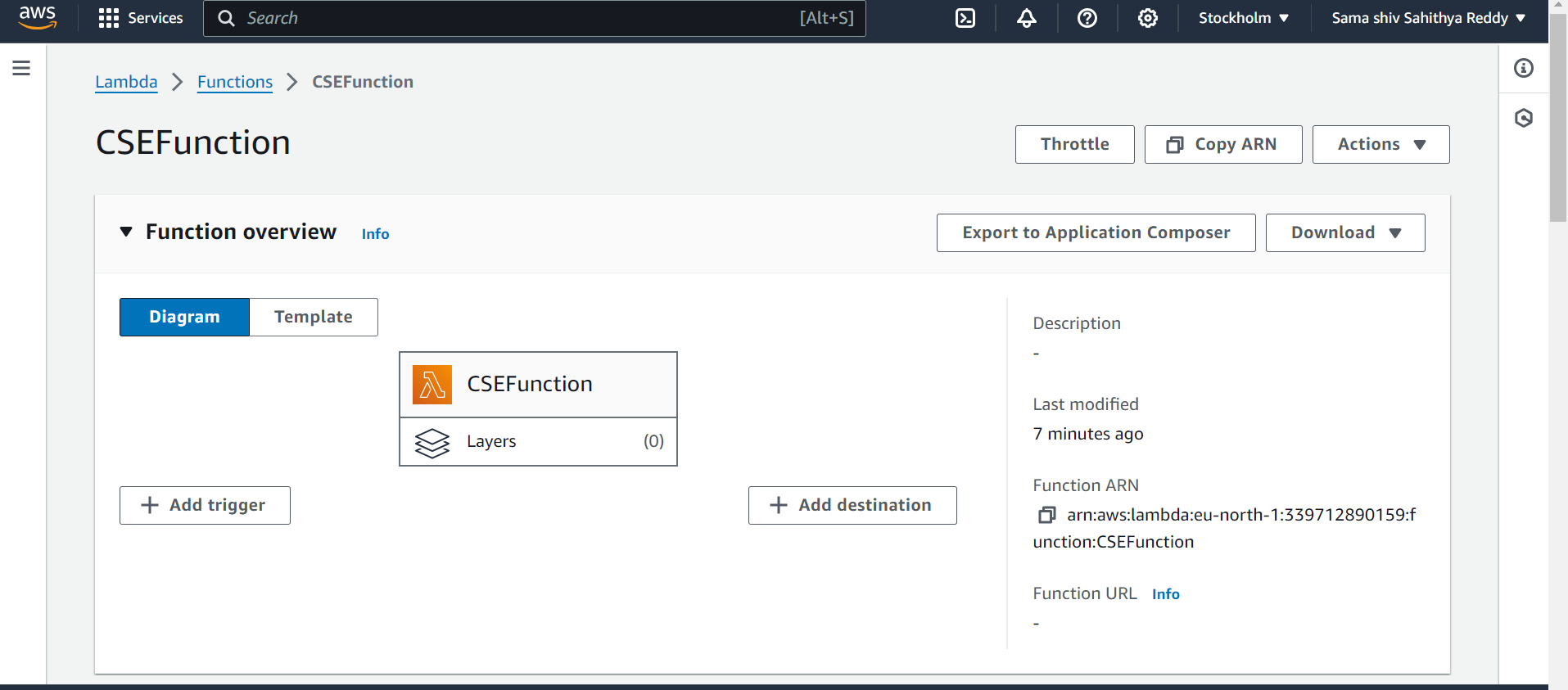
****

25. The next step is to scroll down and write the program and then deploy the program.

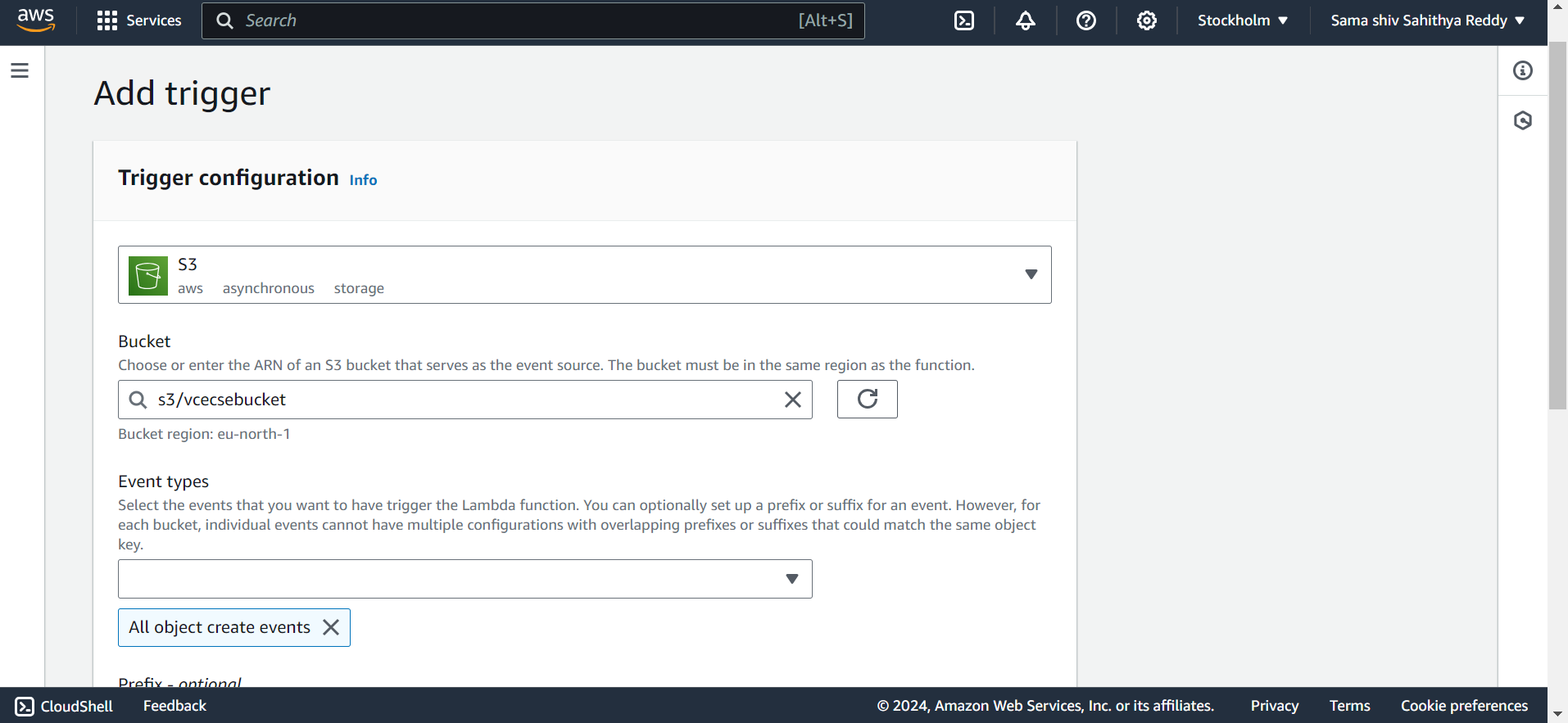




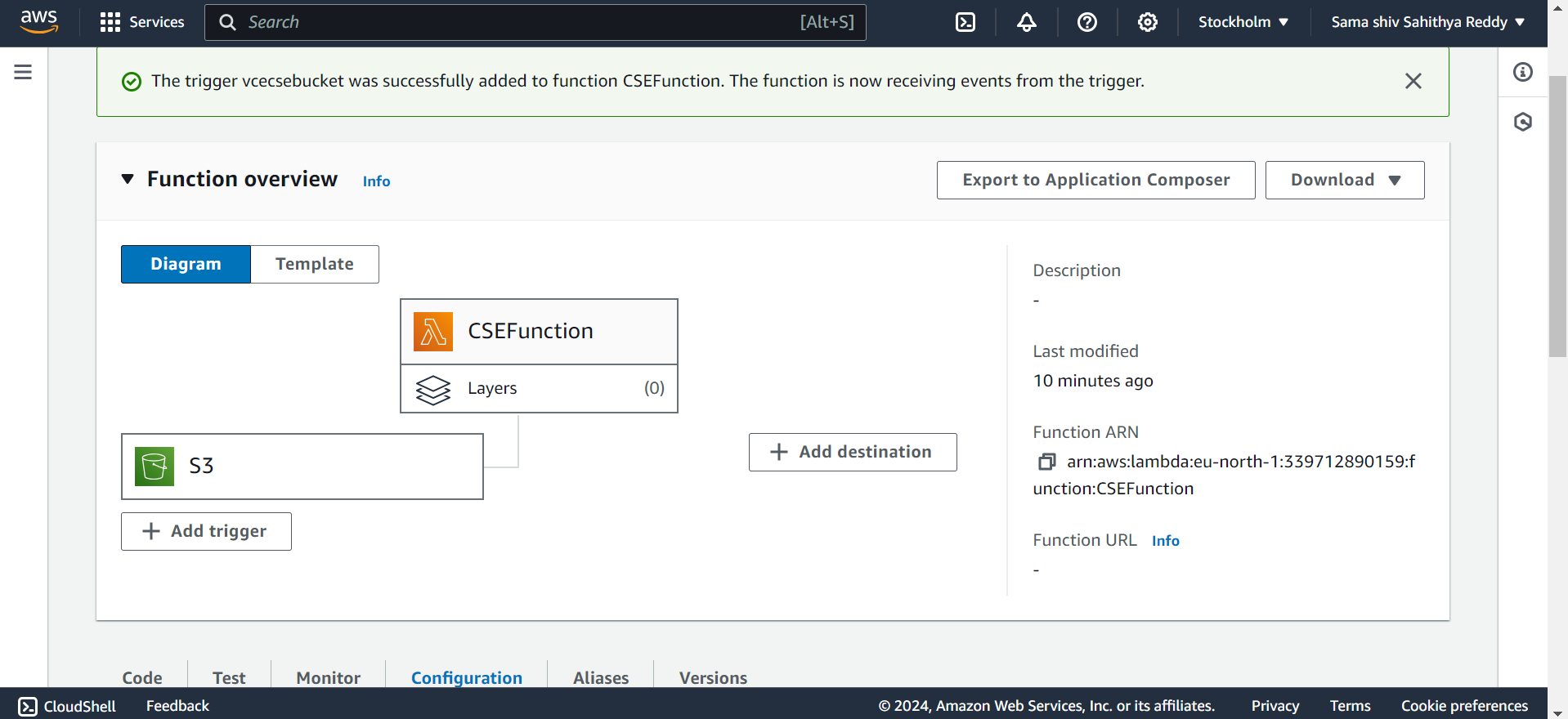
26. Now we should create a trigger and enable it.



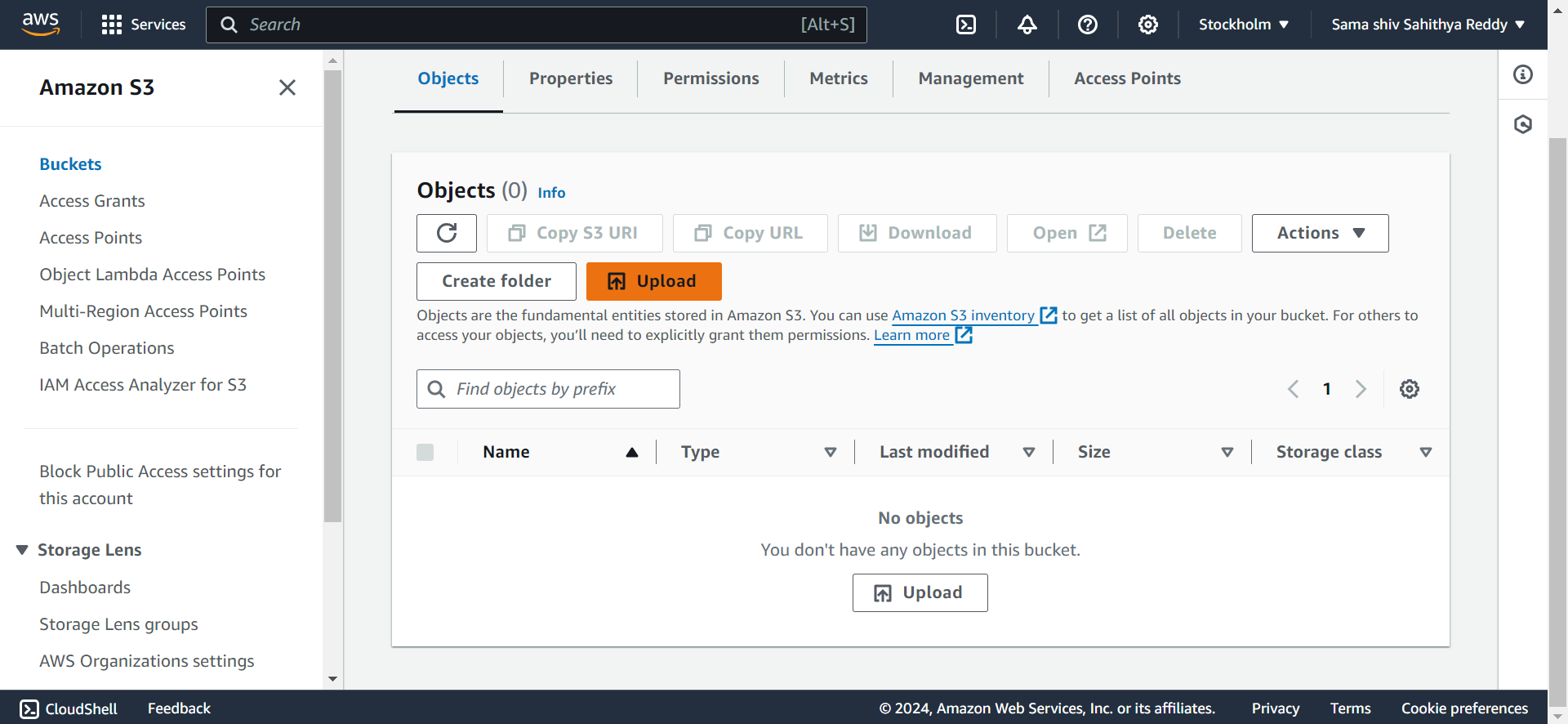
27. After that, add the trigger configuration and select the bucket , that we made and the event types .



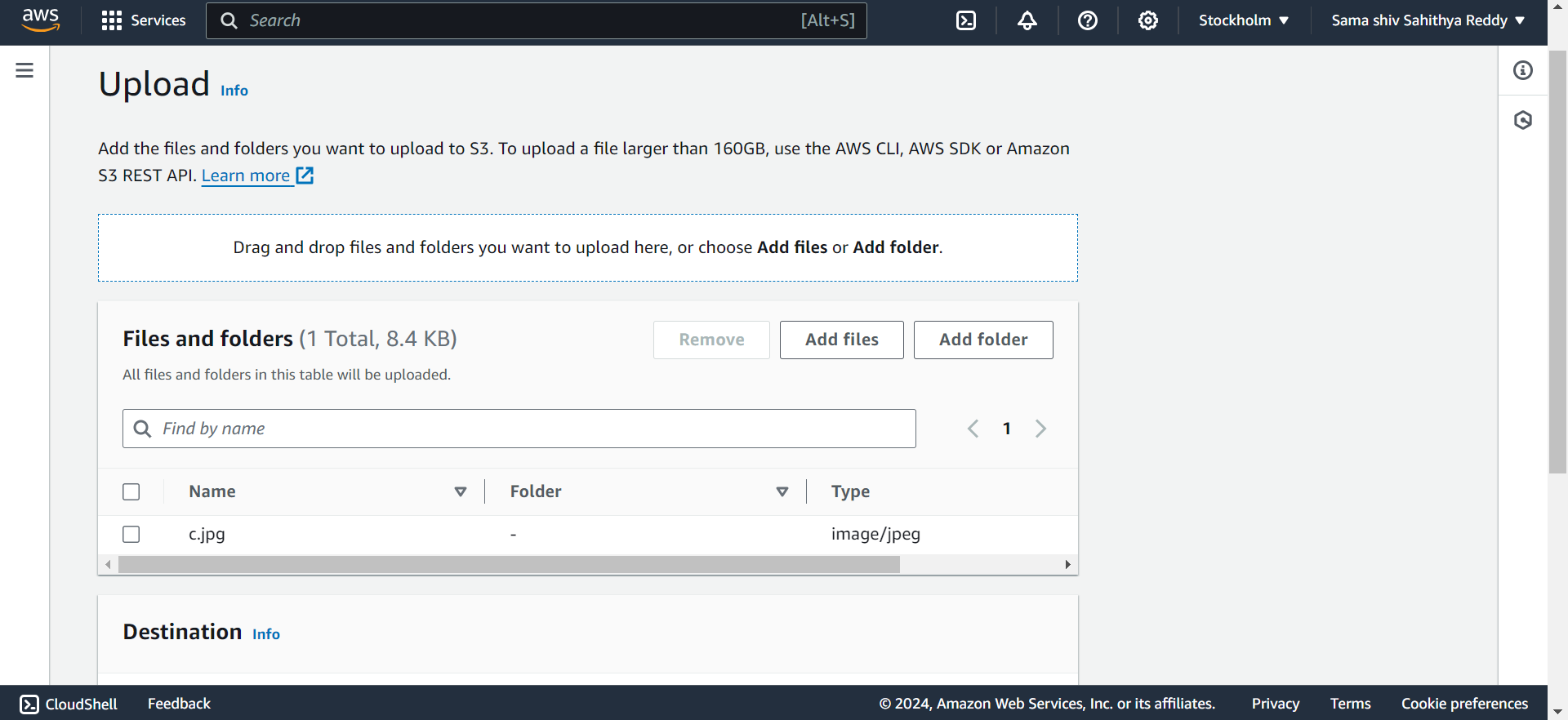
28. Now the trigger is added and link os established for the lambda service .

****

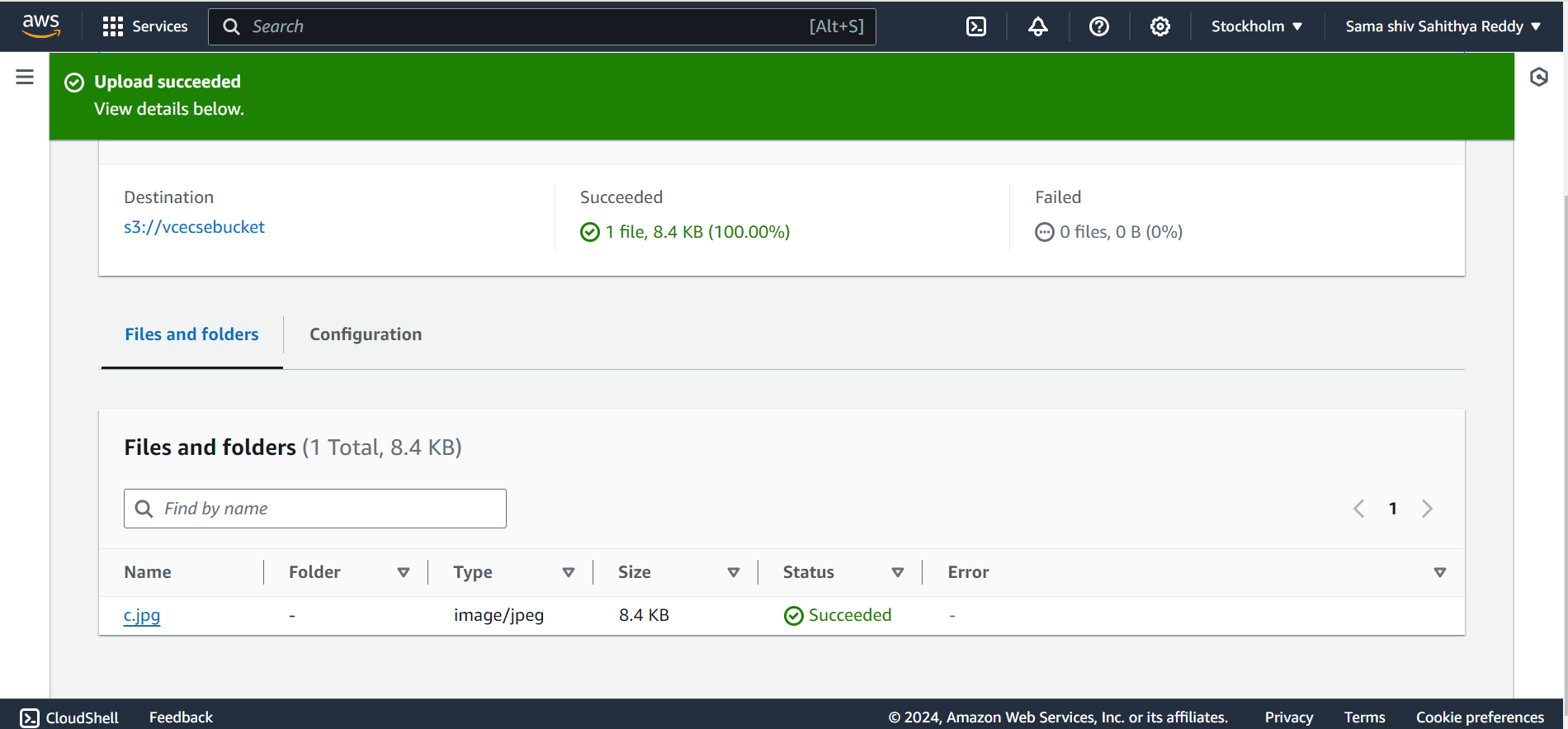
29. Now go to the S3 Bucket and upload any image from the files.

****

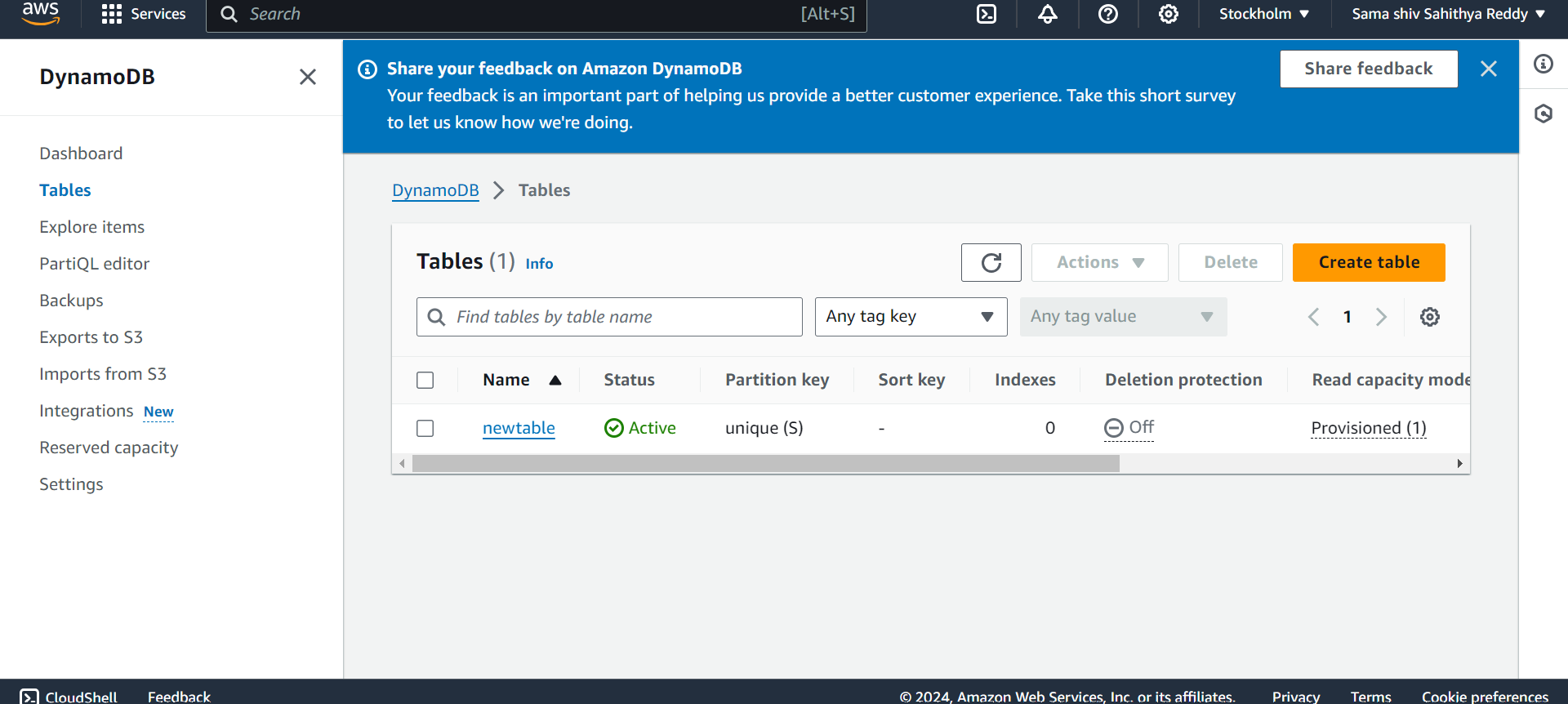
30.Now, Upload the file from add files .

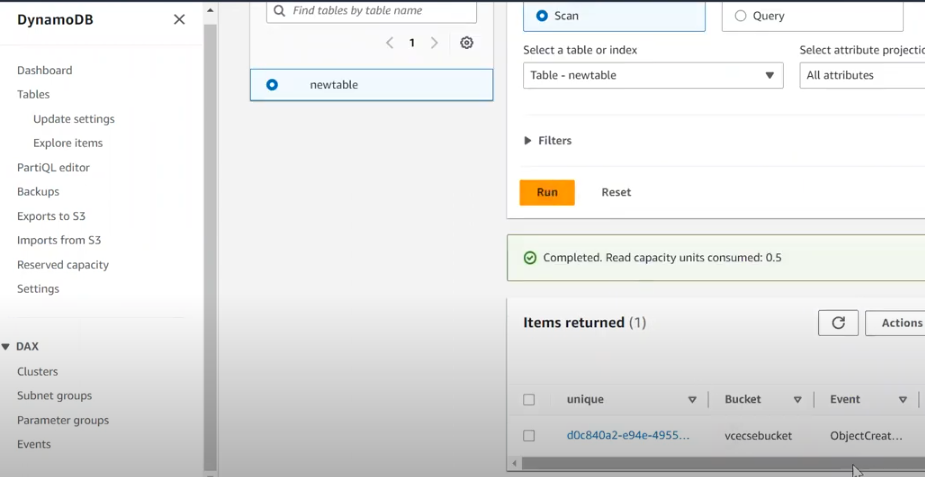
****

31.Now ,the file is been uploaded in the bucket successfully.

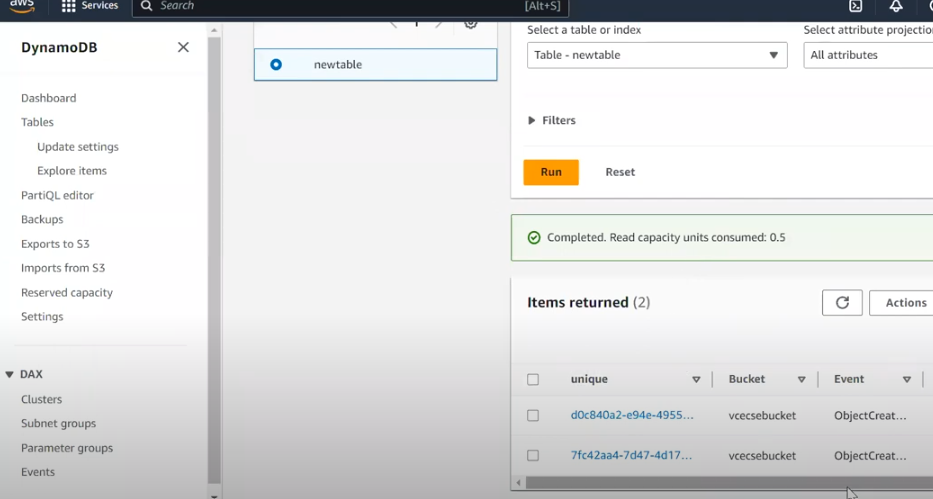
****

32**.** Now, again go to dynamoDb and select the table and go to explore table items .

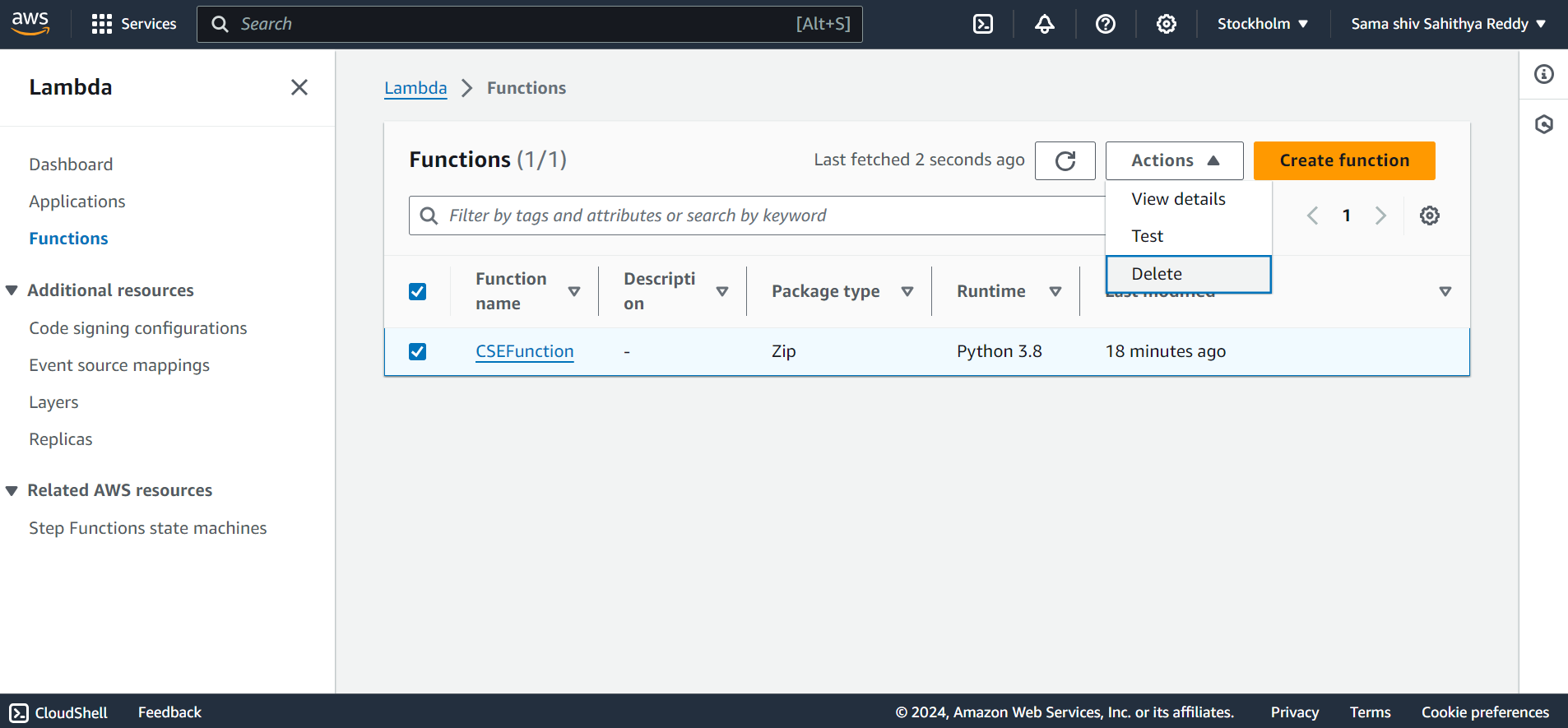
****

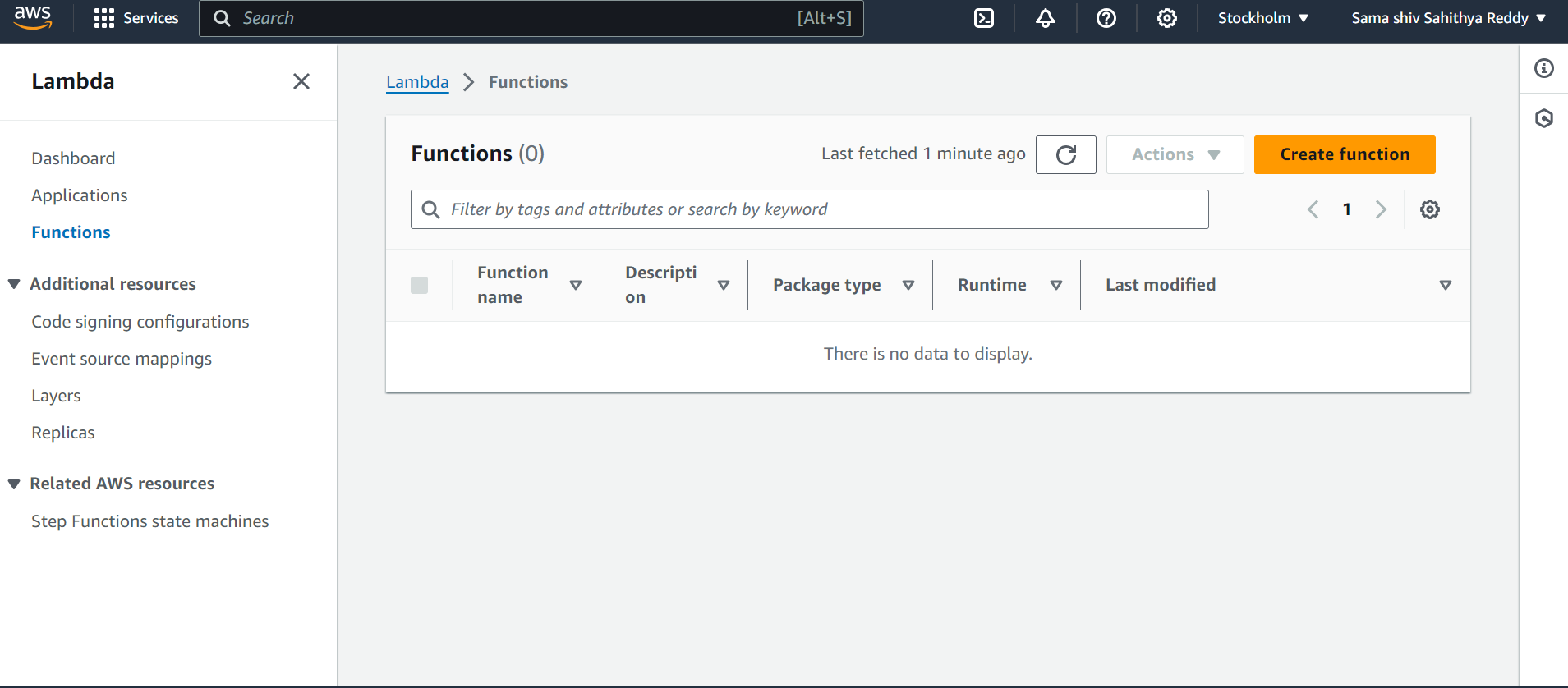
****

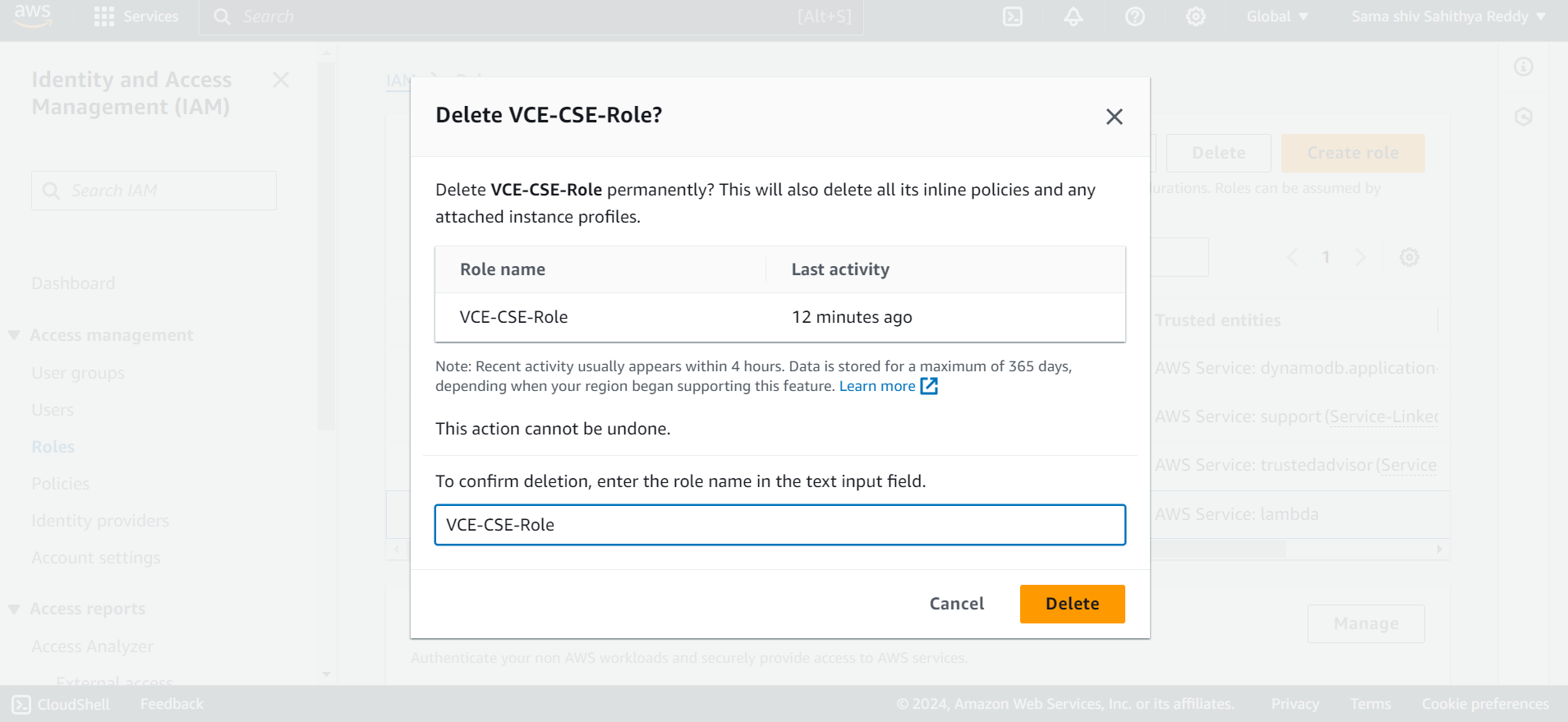
33**.**Now, repeat the process of upload another image from the s3 bucket and check in the dynamodb table if the image is uploaded .

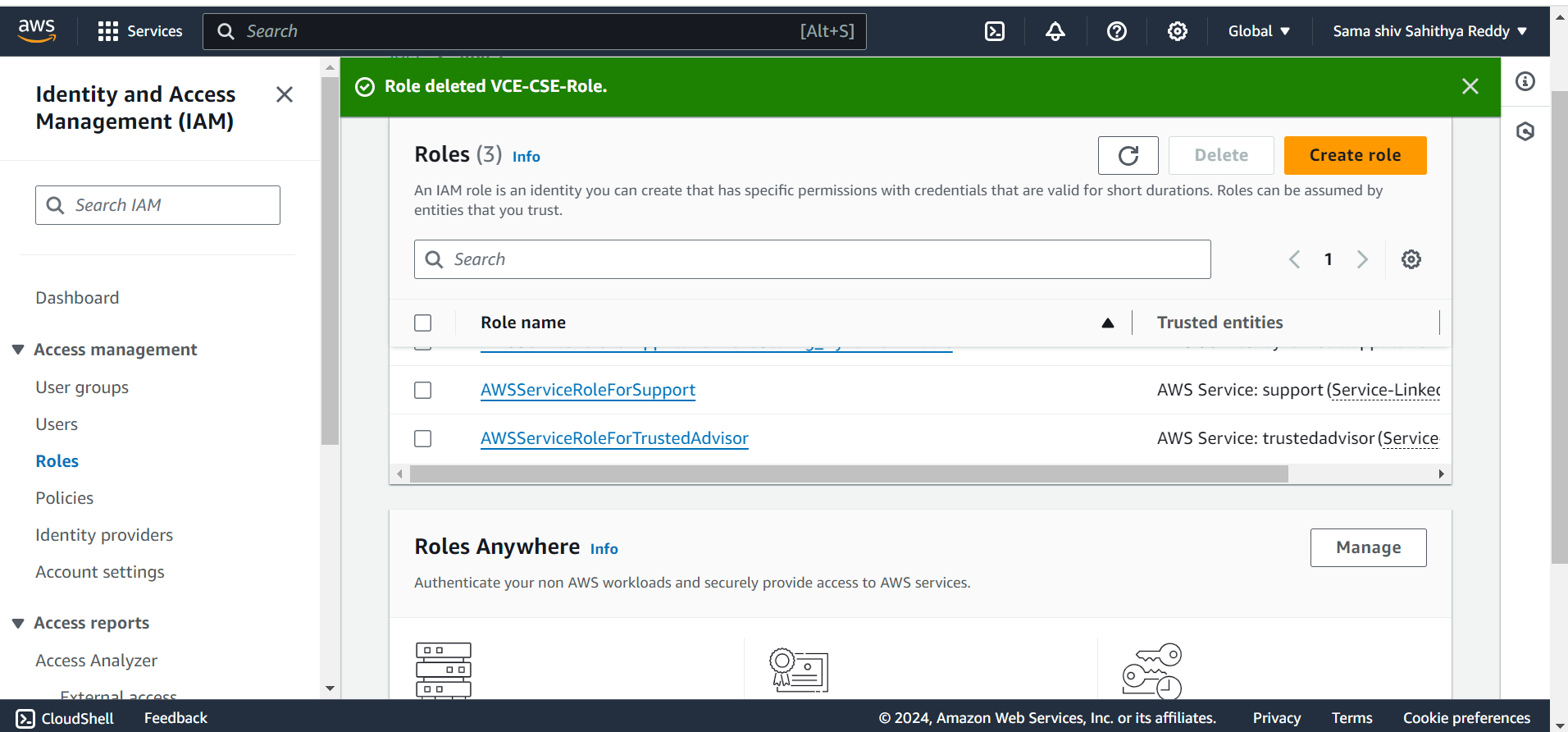
****

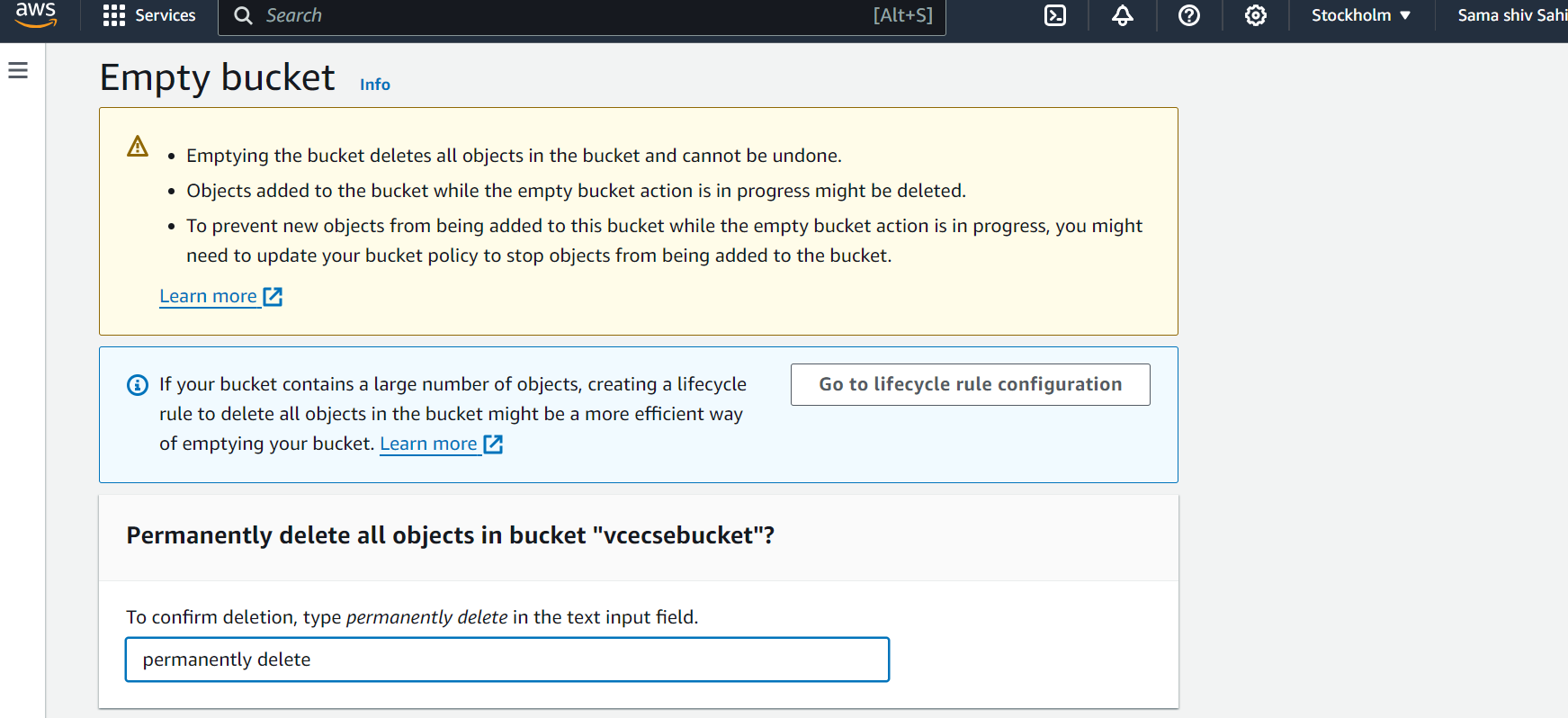
34**.** Now **,** we have to delete every function that we created and verify that evert function has been deleted.

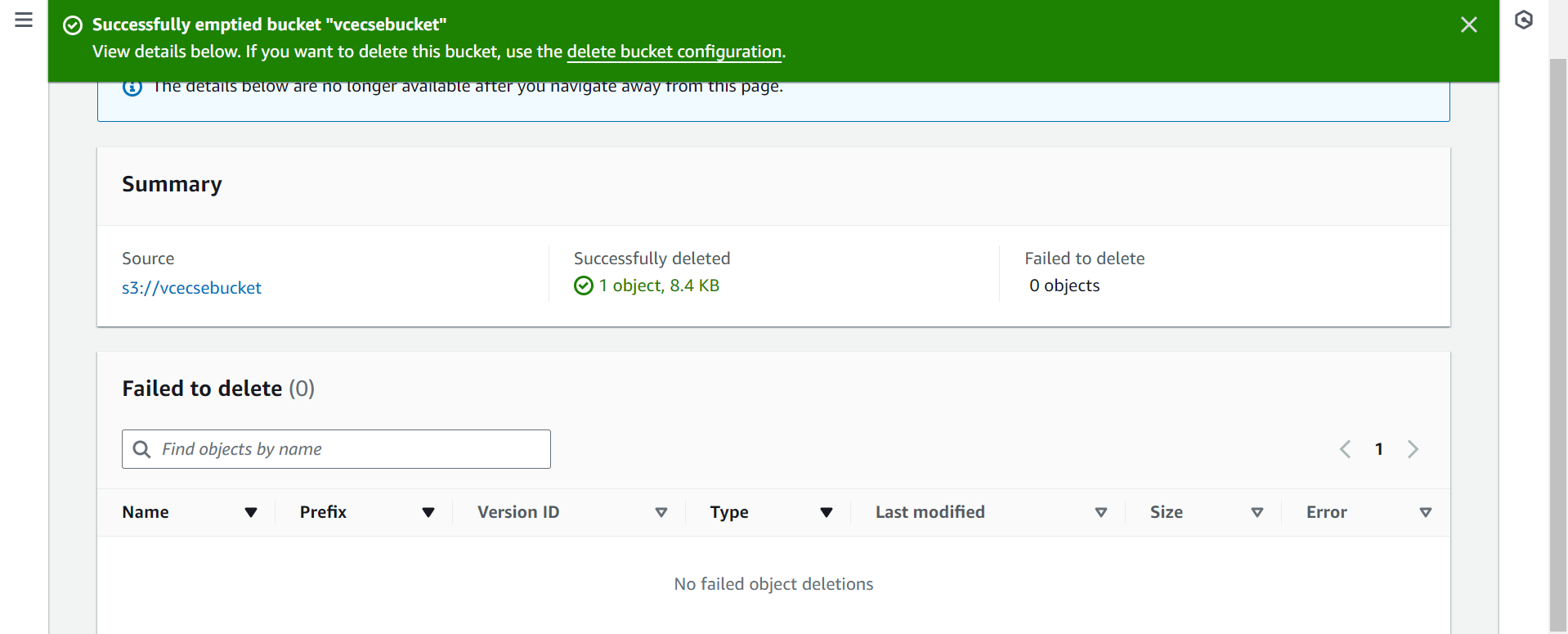
****

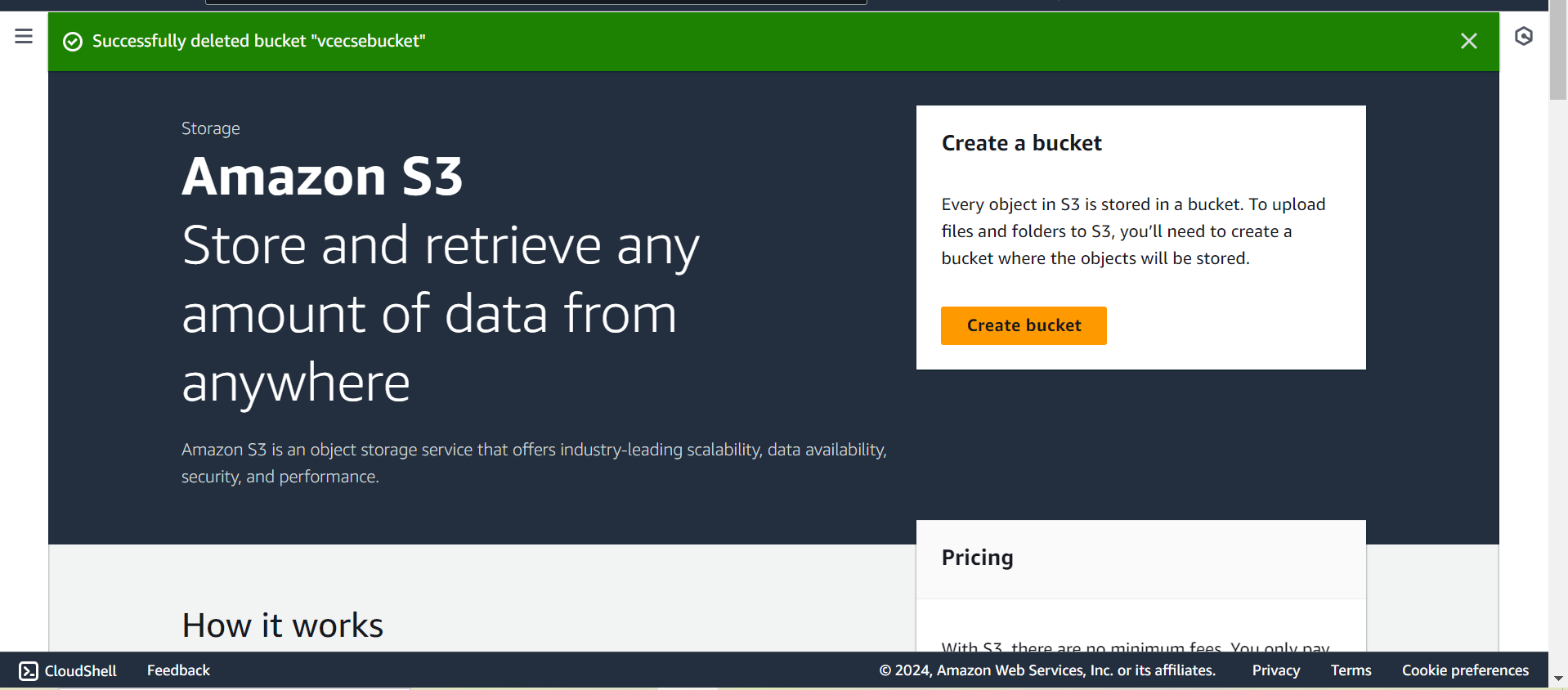
****

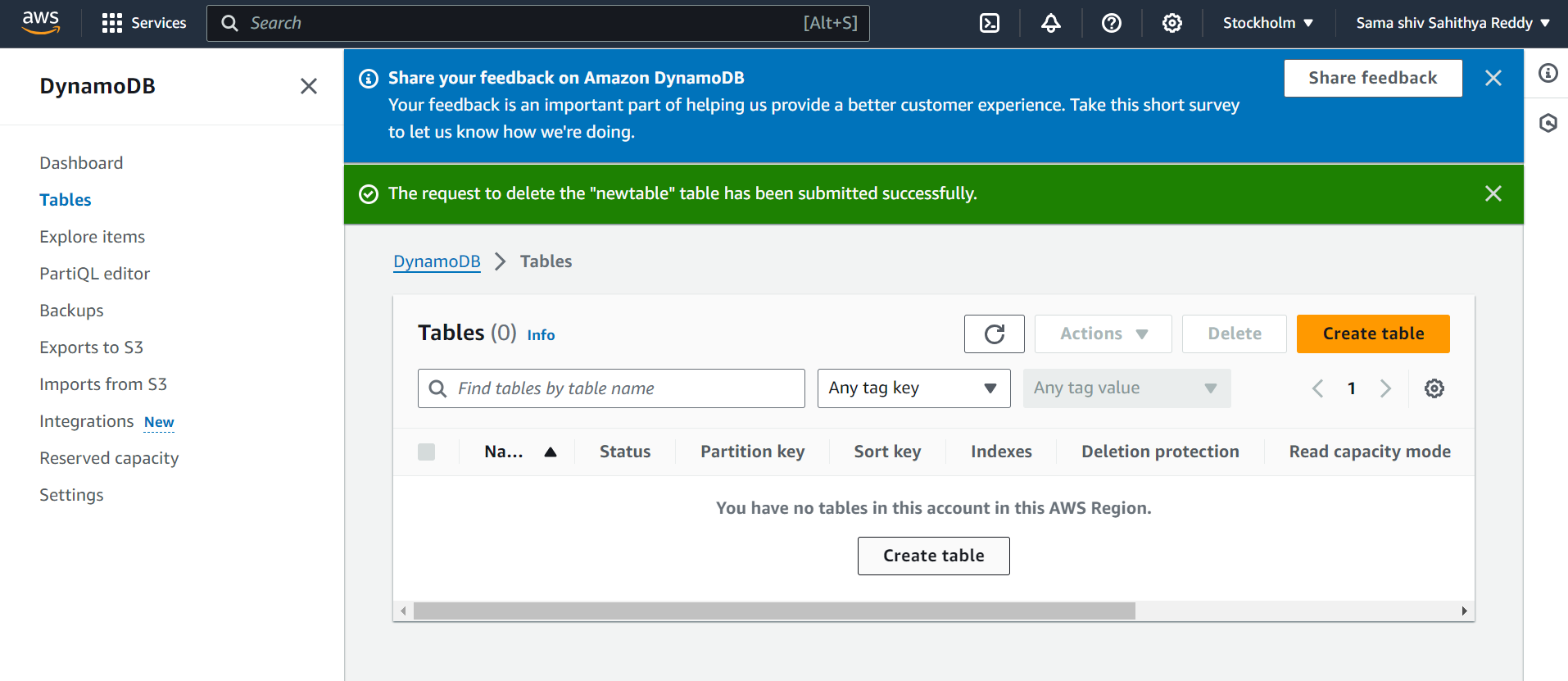
****

****

****

****

****

****

**CHAPTER-3**

**Conclusion:**

In conclusion, AWS Lambda stands as a pivotal service within the AWS ecosystem, revolutionizing the way developers build and deploy applications. Its serverless architecture eliminates the overhead of managing infrastructure, allowing for seamless scalability and cost-effectiveness. With support for multiple programming languages and deep integration with other AWS services, Lambda empowers developers to build highly responsive and efficient applications.

Moreover, AWS Lambda's pay-per-use pricing model and fine-grained access control make it an attractive choice for organizations seeking to optimize costs and enhance security. Its versatility enables a wide range of use cases, from real-time data processing to event-driven microservices, offering flexibility and agility to meet diverse application requirements.

**CHAPTER-4**

**References:**

**1. https://github.com**

**2.Youtube**