**Project 1**

**Serverless Image Processing**

Create a serverless image processing application that automatically resizes and optimizes images

uploaded to an Amazon S3 bucket.

📄 Table of Contents

* What is AWS Lambda?
* Basic Lambda Structure
* Common Use Cases
* Key Benefits
* Pictorial Representation
* Implementation Steps

**🧠 What is AWS Lambda?**

**AWS Lambda** is a **serverless compute service** that lets you **run code without provisioning or managing servers**. You simply upload your function code, and Lambda takes care of everything required to run and scale it.

**📦 Basic Lambda Structure**

A Lambda function typically consists of:

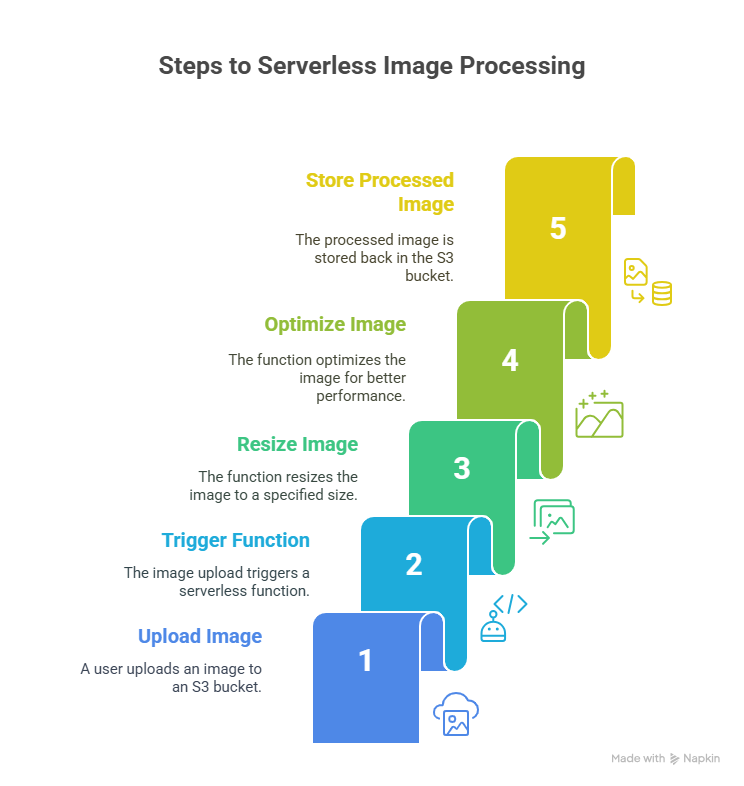
* **Handler**: the entry point of your code
* **Event**: data passed to your function when triggered
* **Context**: runtime information provided by Lambda

**🧰 Common Use Cases**

| **Use Case** | **Example** |
| --- | --- |
| **Data Processing** | Resize images on S3 upload |
| **Real-time File Processing** | Analyze logs stored in S3 |
| **Web Backend** | Handle HTTP requests via API Gateway |
| **Automation** | Clean up resources on a schedule |
| **Event Streaming** | Process messages from SQS, SNS, or Kinesis |

**💡 Key Benefits**

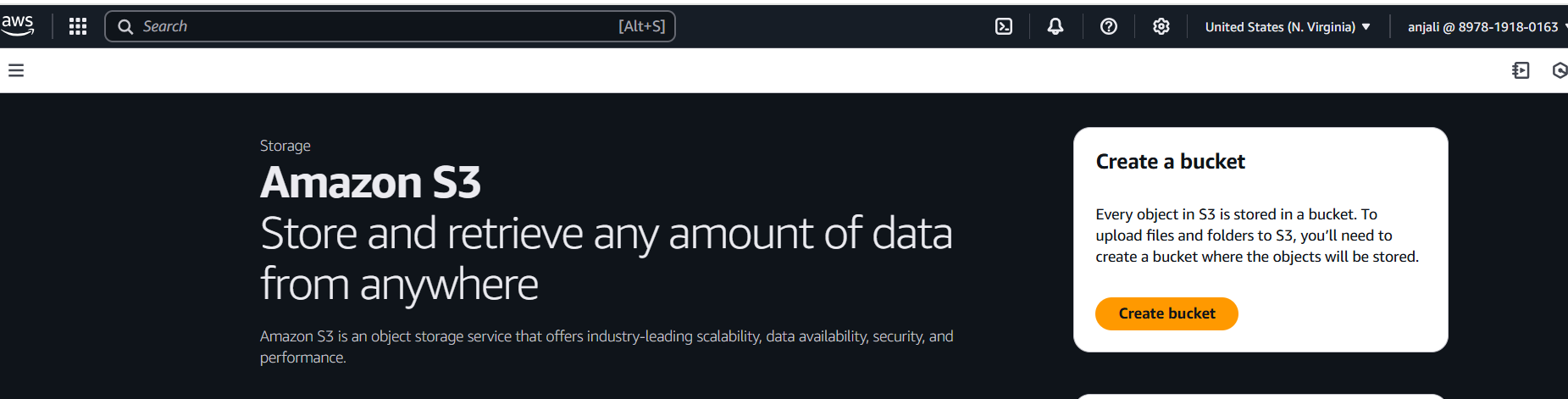
* **No server management**
* **Automatic scaling**
* **Pay per use** (only for compute time used)
* **Quick deployment**
* **Tight integration with AWS ecosystem**
* **Pictorial representation of serverless image resizing**

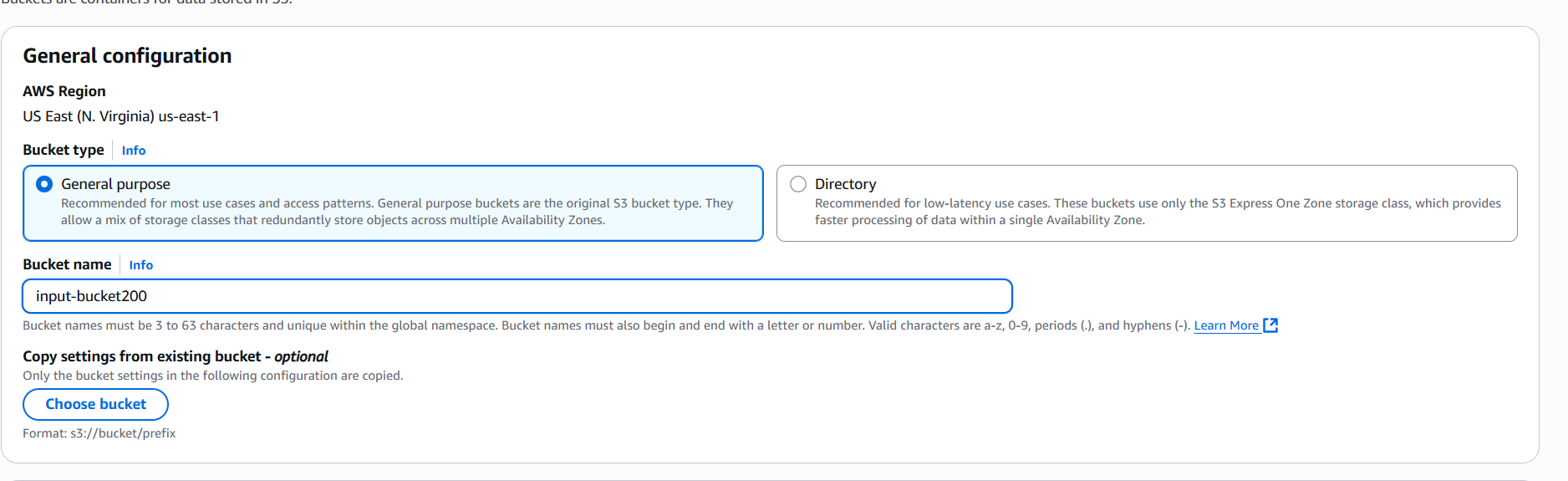


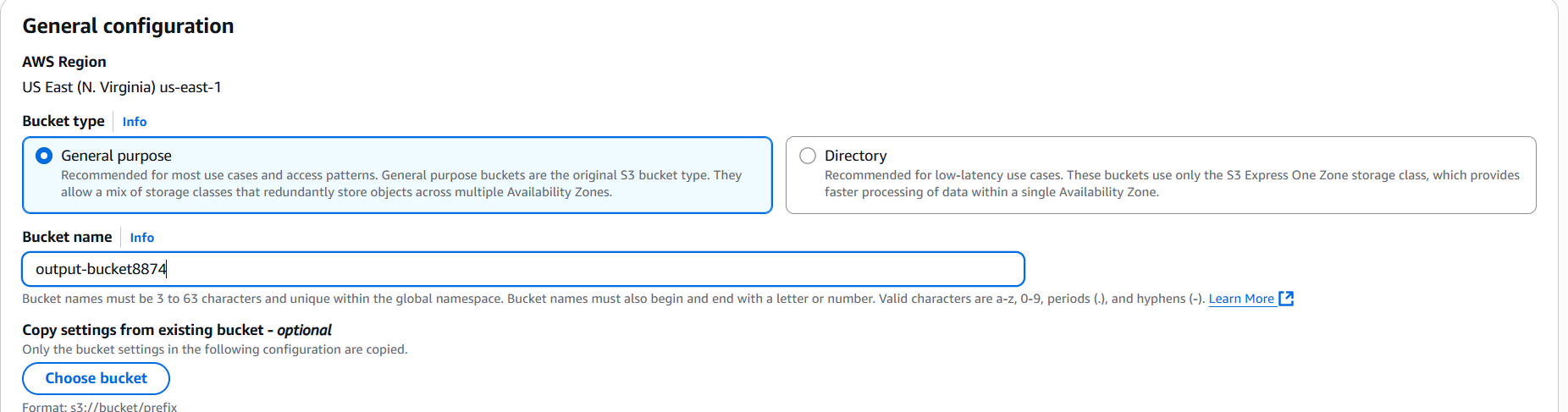
🔧 Implementation Steps

**Step - 1. Setting Up S3 Buckets**

1. Create Two S3 Buckets: One for uploading the original images (input-bucket) and another for storing the resized images (output-bucket).

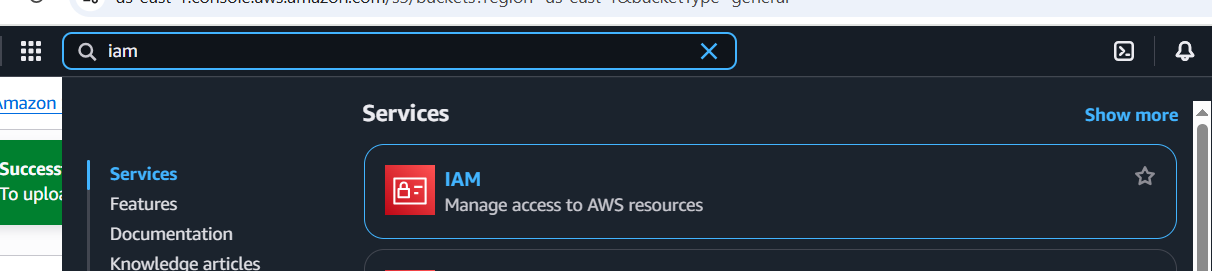




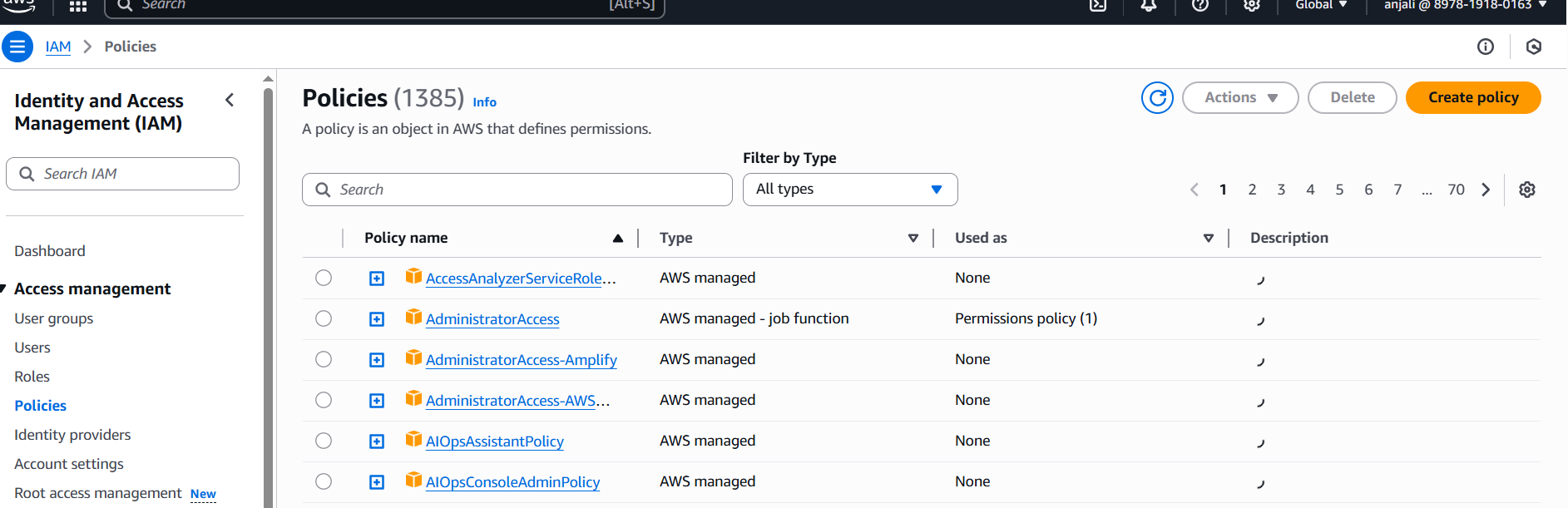


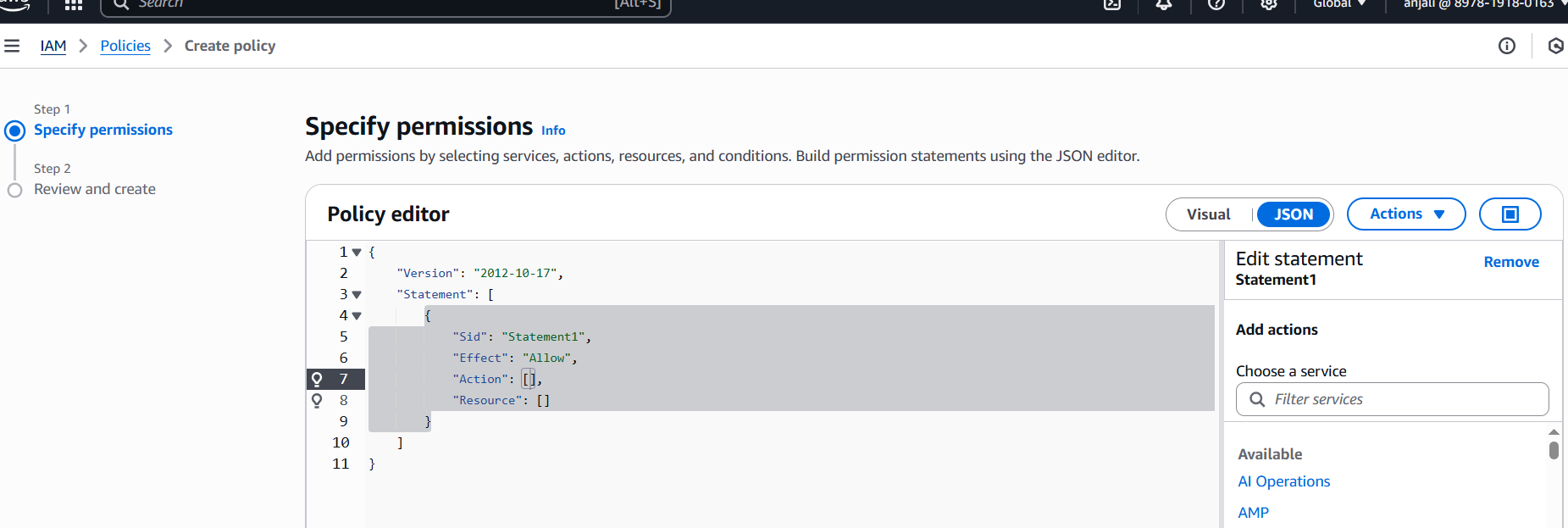
**Step - 2. Setting Up IAM Role**

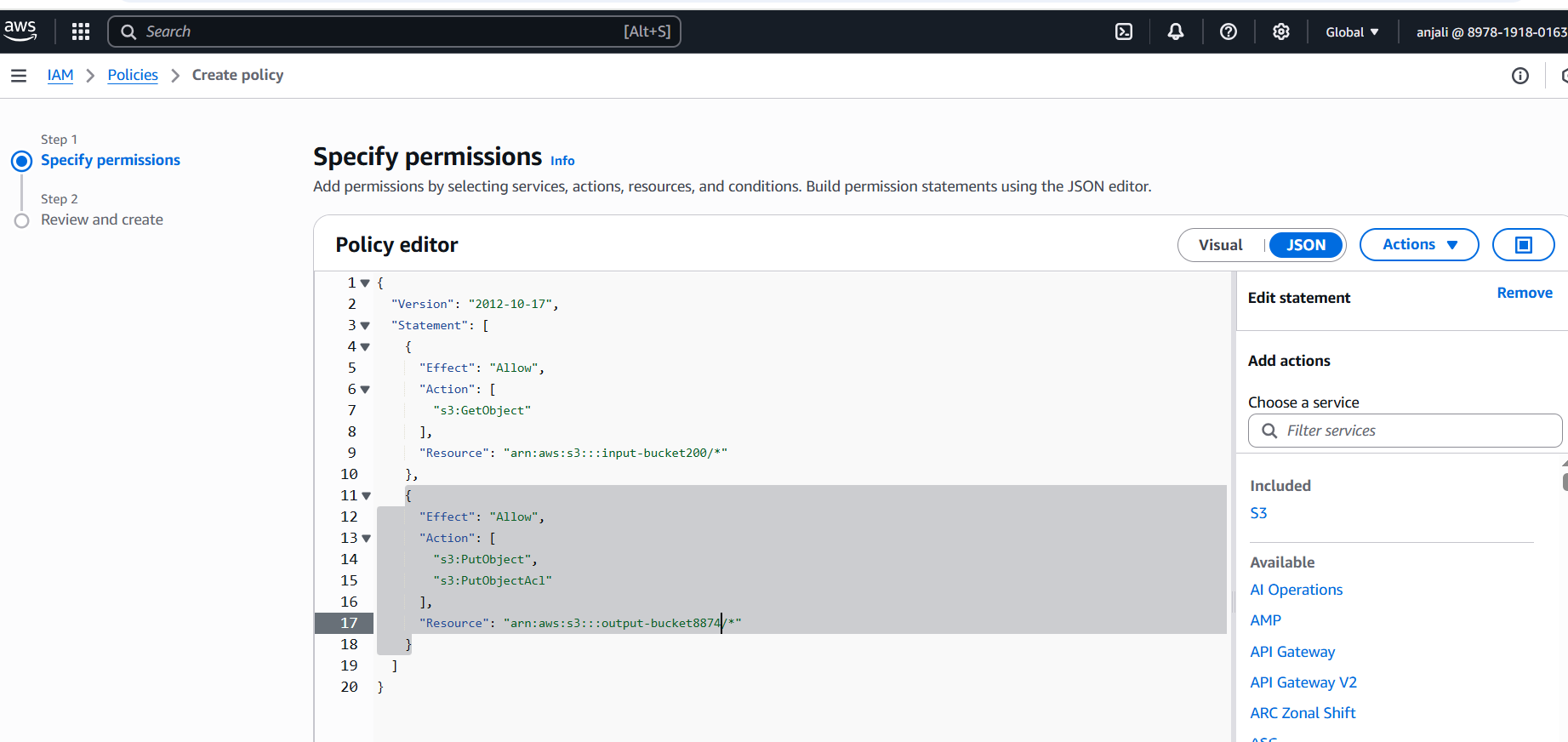
1. Create a Custom IAM Policy:

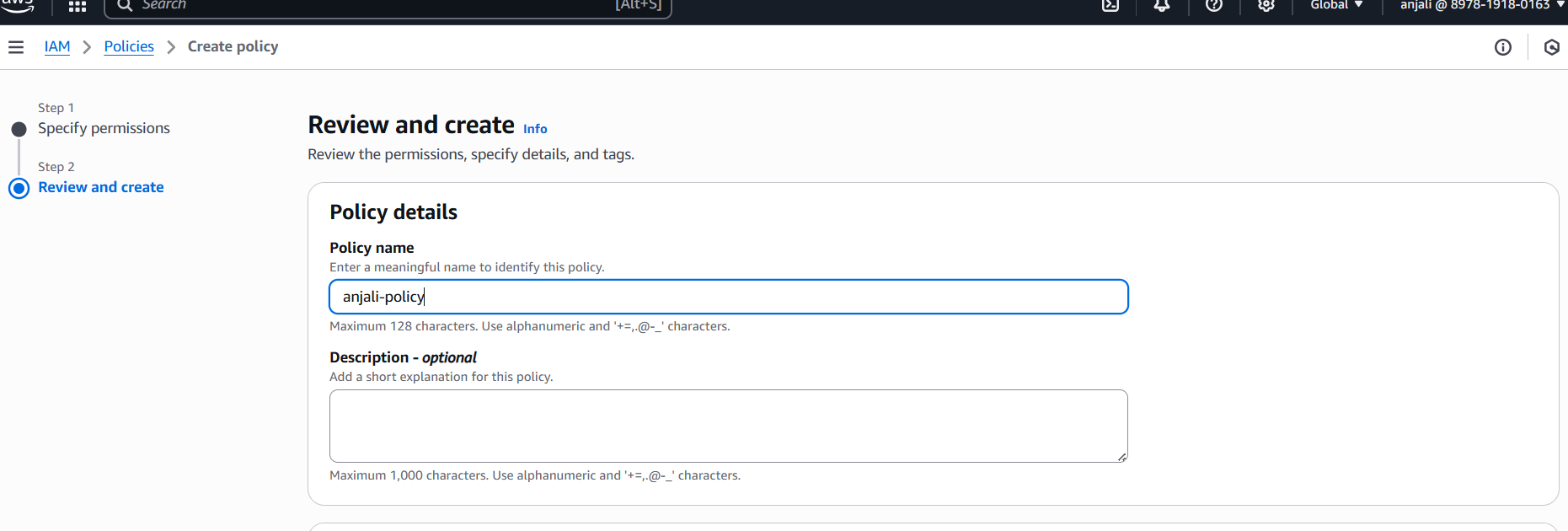


* Go to the IAM dashboard and create a new policy.

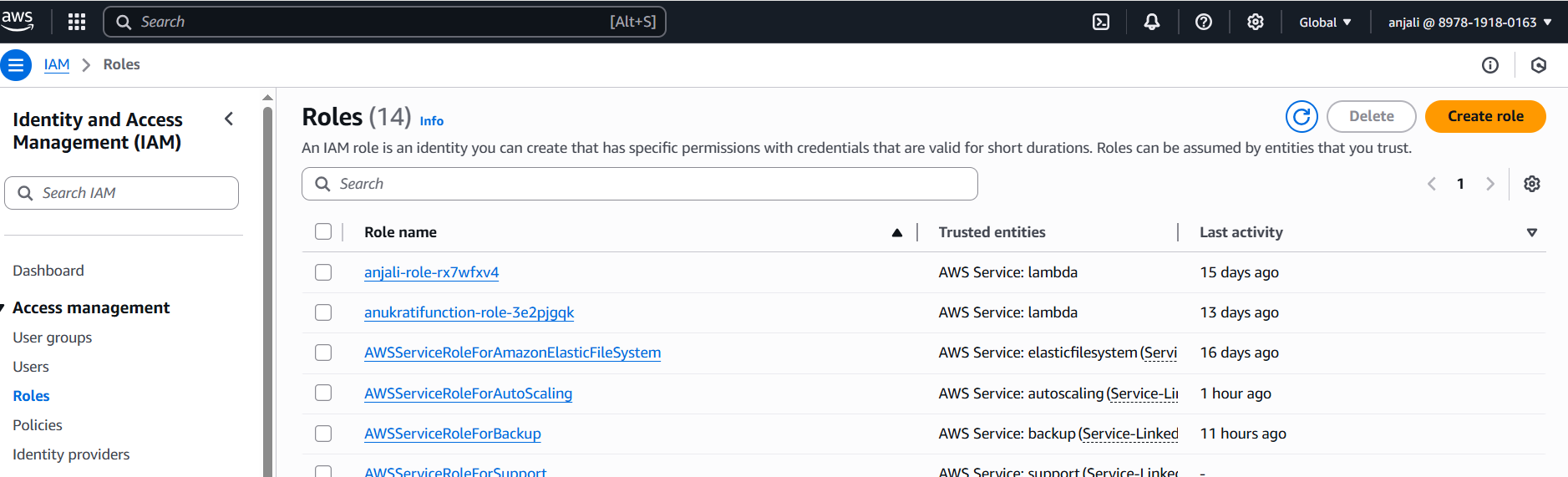


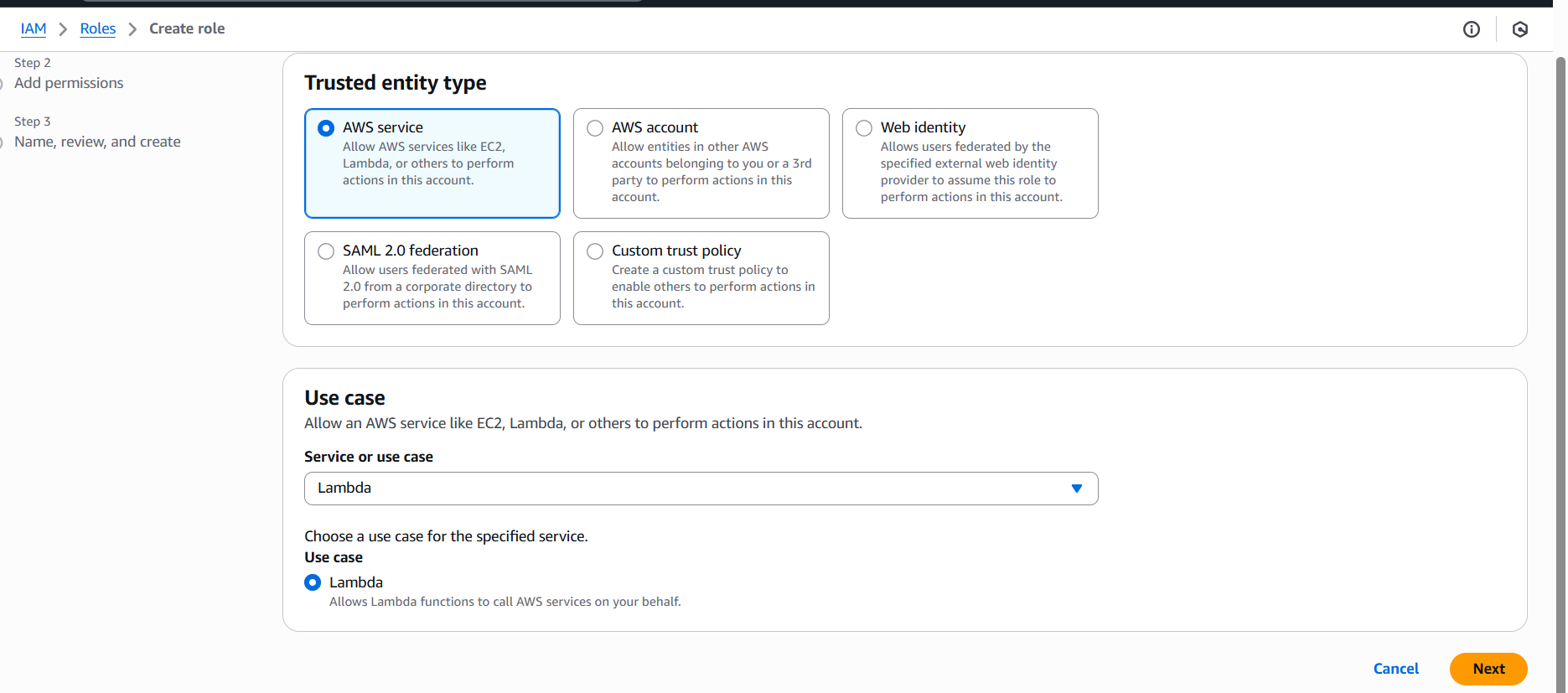




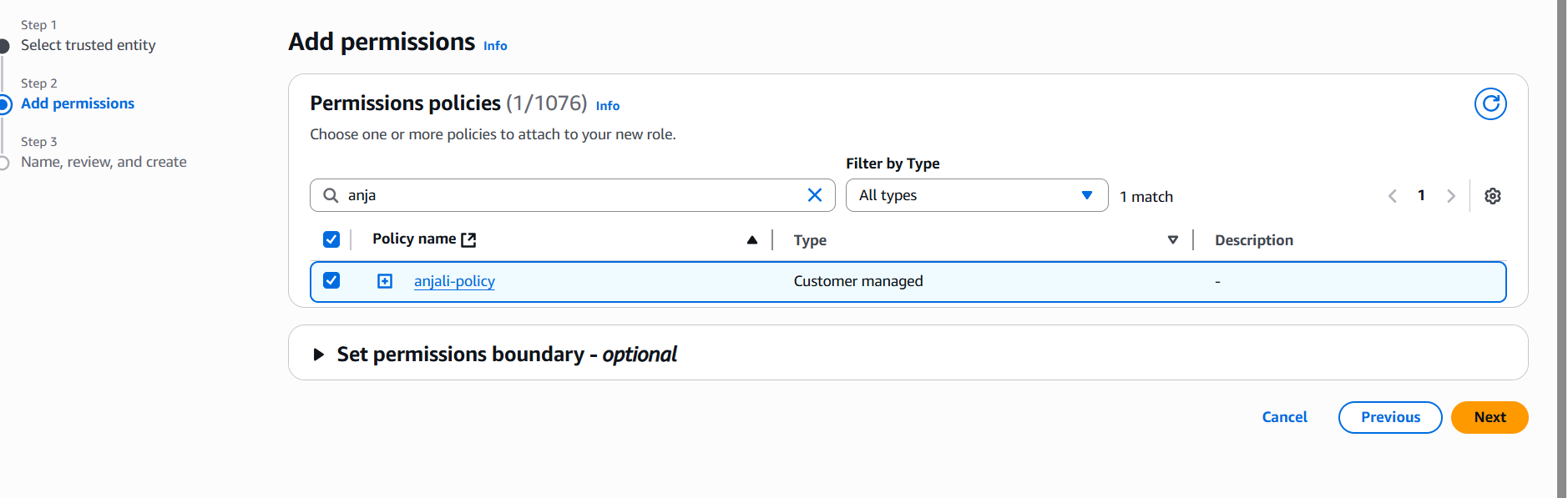


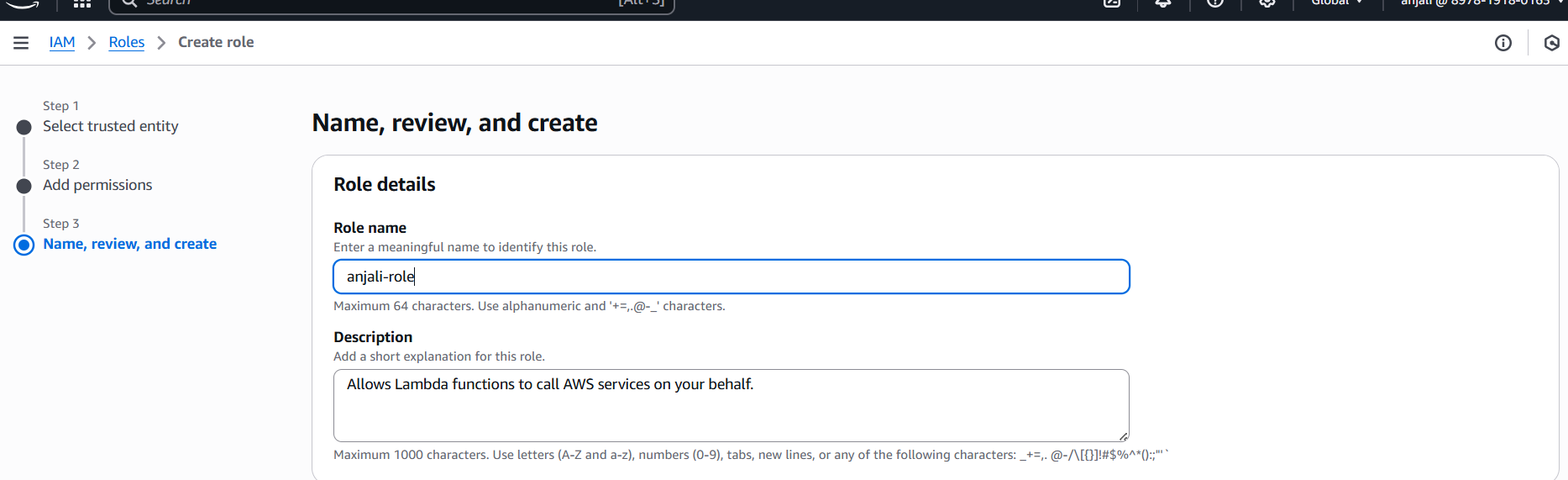
* Use the following template, replacing input-bucket-199 and output-bucket-199 with your actual bucket names:
* Now go to the IAM dashboard and create a new role.





* Select AWS Lambda as the use case.
* Find the policy you just created using the policy name, select it, and attach it to the role.

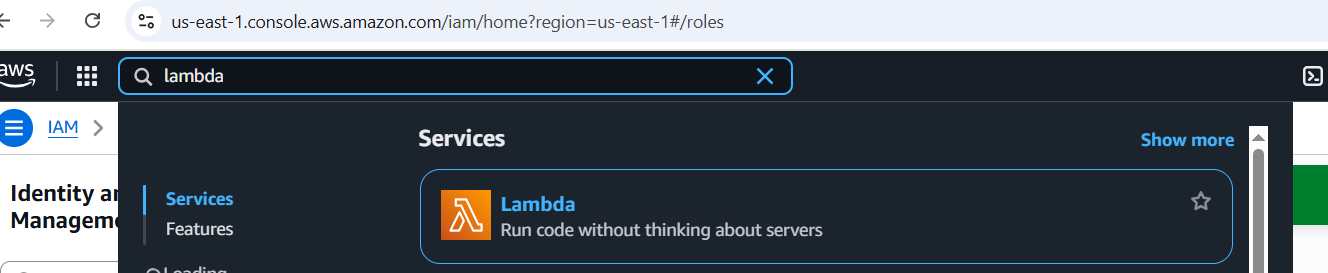


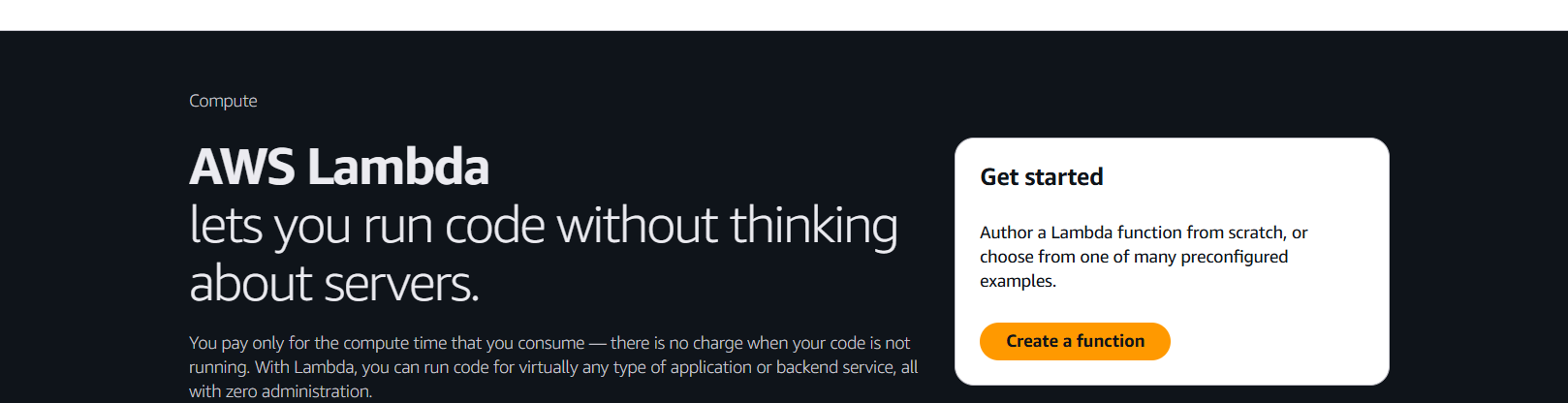


**Step - 3.Creating the Lambda Function**

1. Write the Lambda Function:

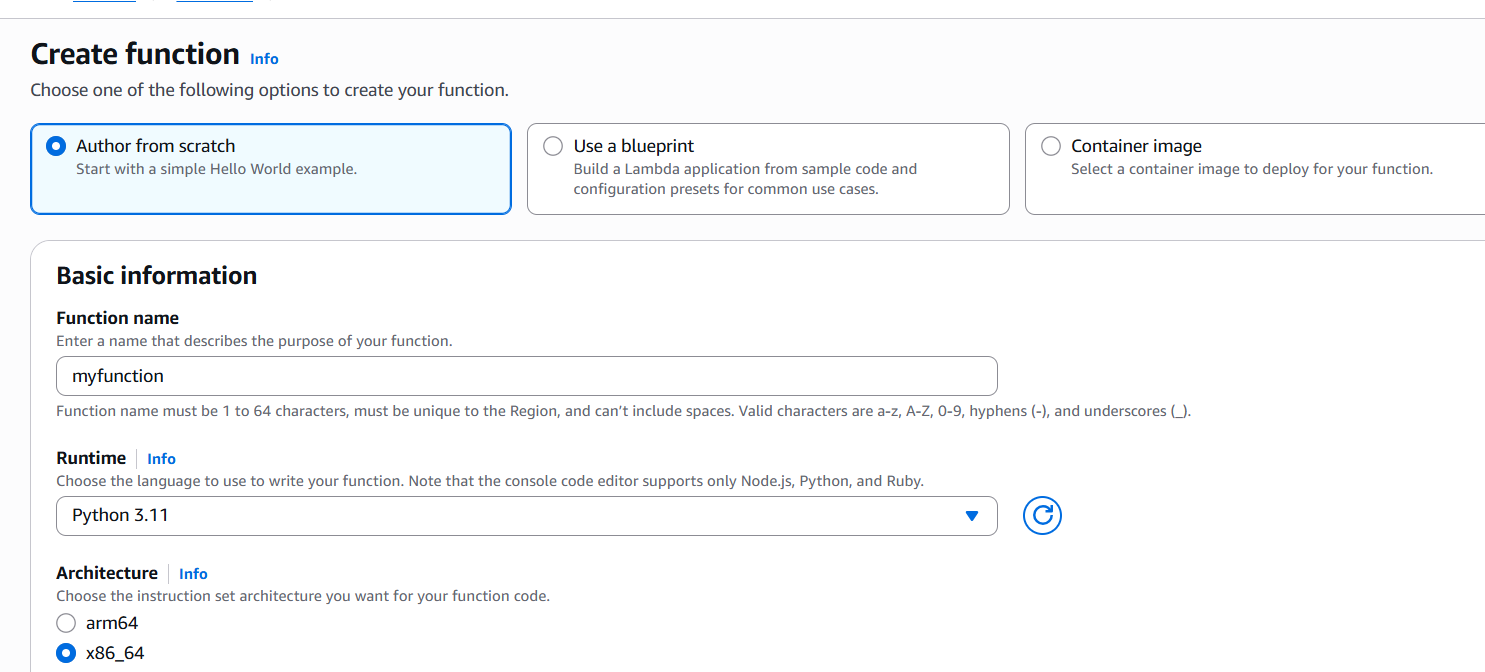
Head over to the AWS Lambda dashboard and click on ‘Create function’.



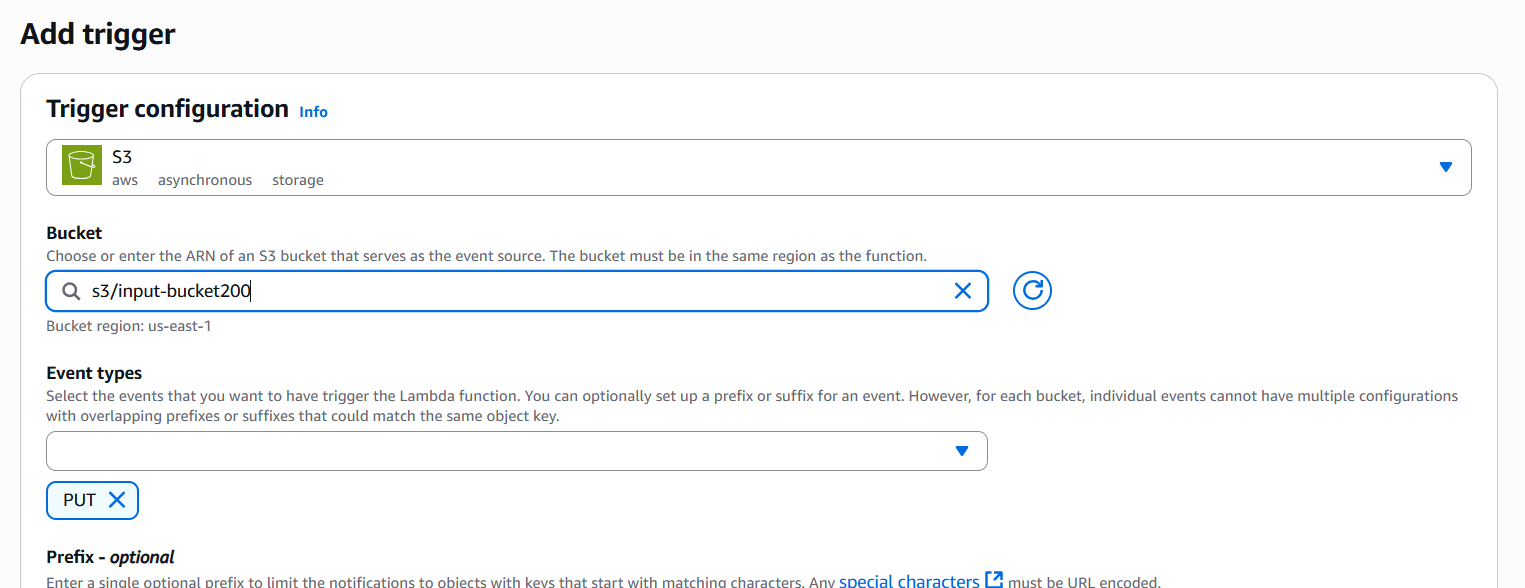


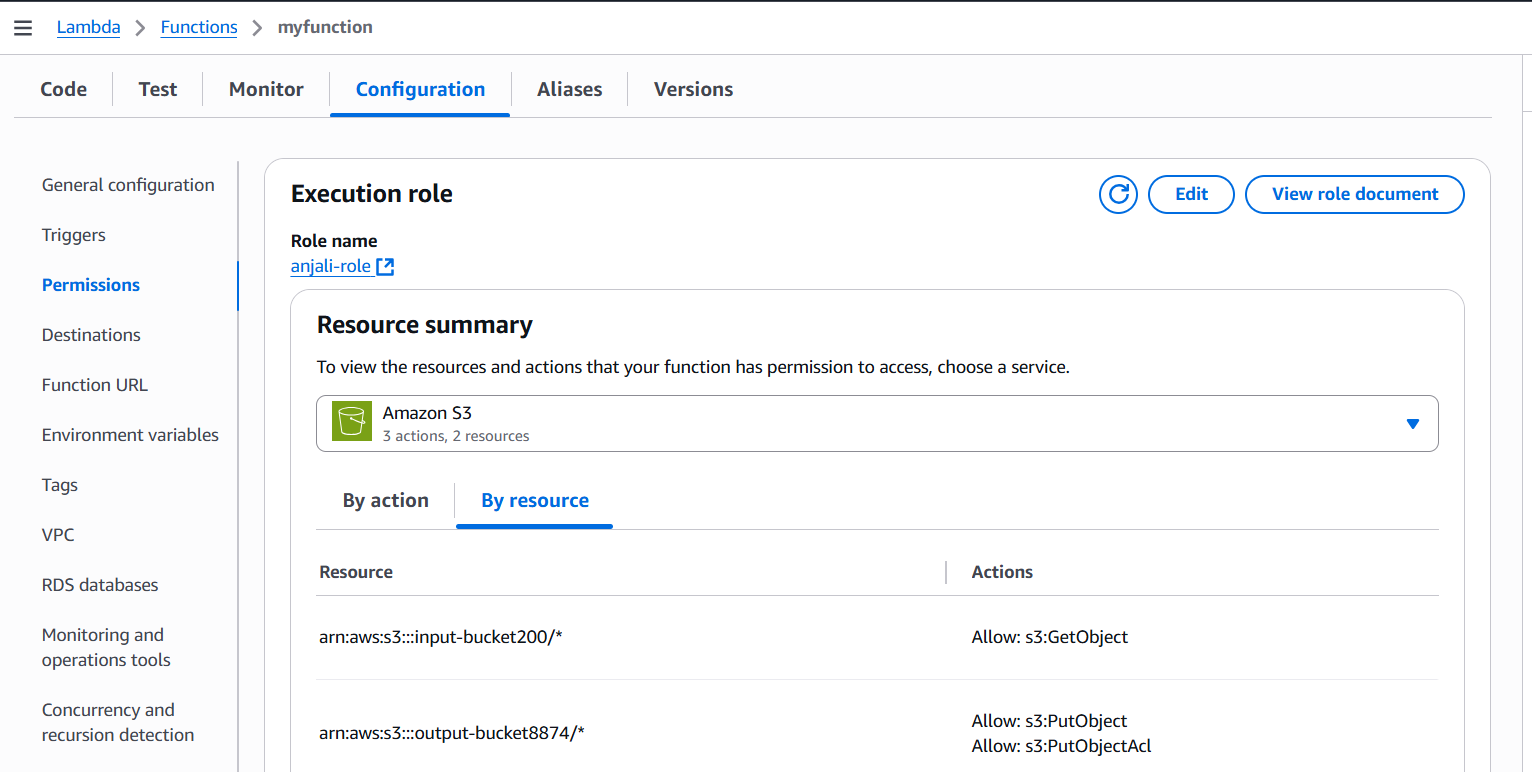
Choose ‘Author from scratch’ and set a name for your function.

* Choose Python as the runtime.



* Use the Python Imaging Library (Pillow) for image processing.
* The function should be triggered whenever a new image is uploaded to the input-bucket.





* After processing (resizing), the function should save the image to the output-bucket.

Example Python code for the Lambda function:

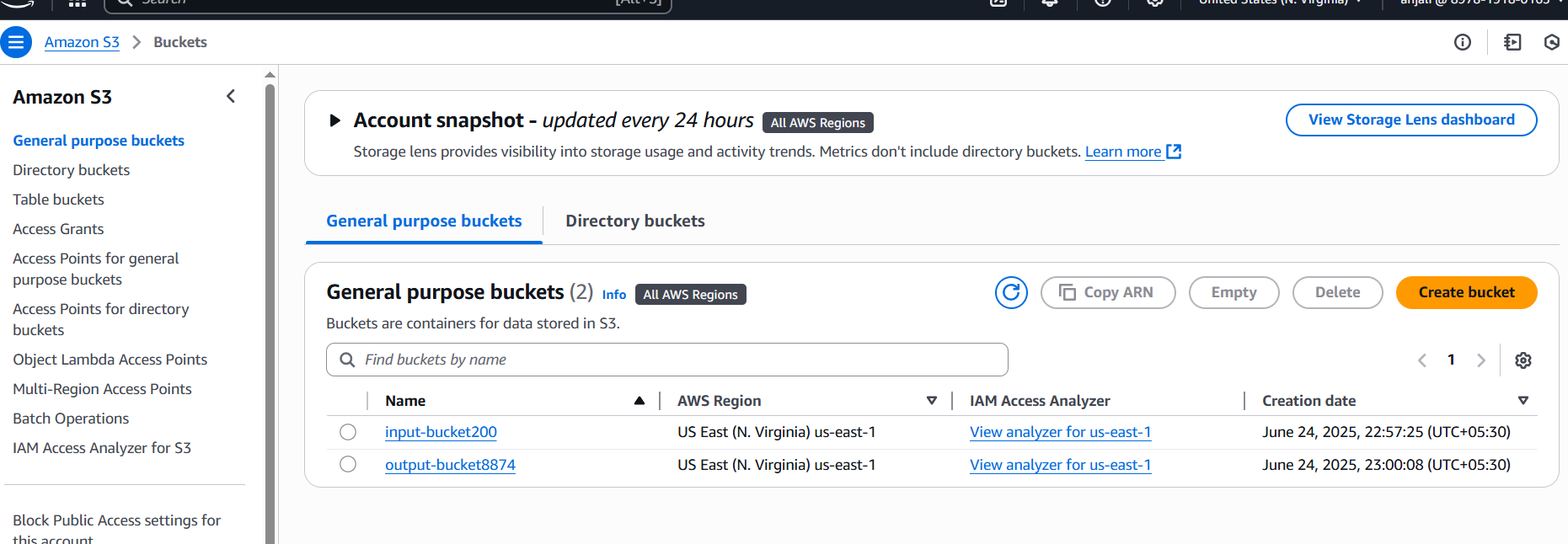
1. Example Python code for the Lambda function:
2. . Head over to the function and add a trigger to the function: choose S3, select the input bucket, and specify the event type (e.g., PUT).

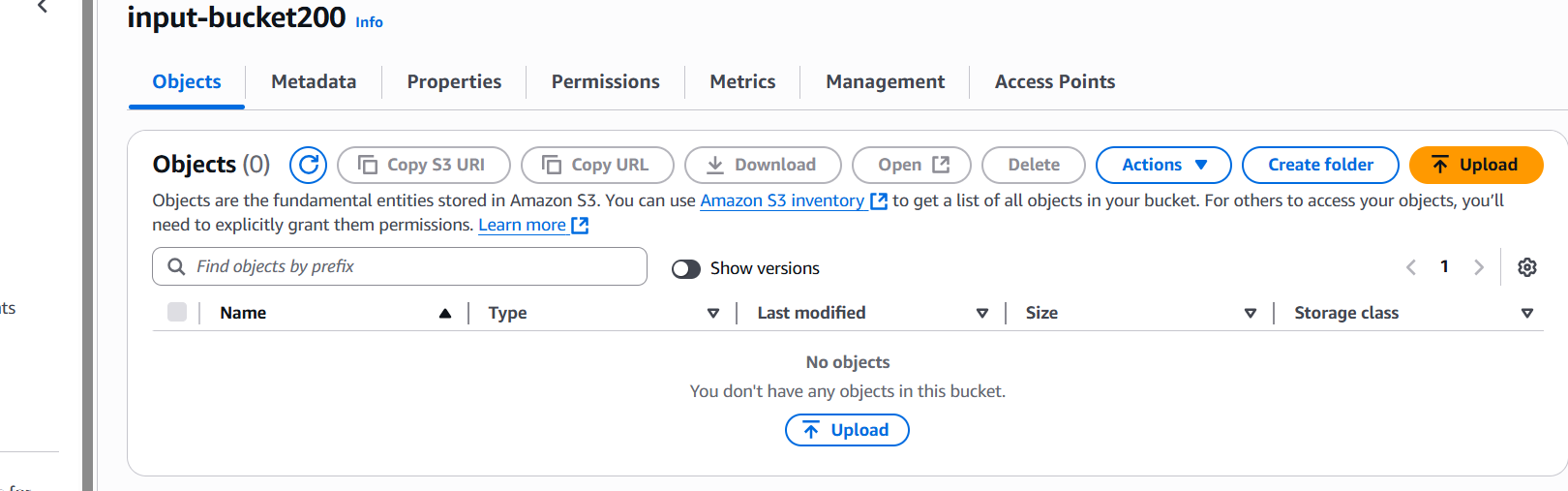
 Now select the role that we have created earlier.

**Testing the Application**

1. Upload an Image:

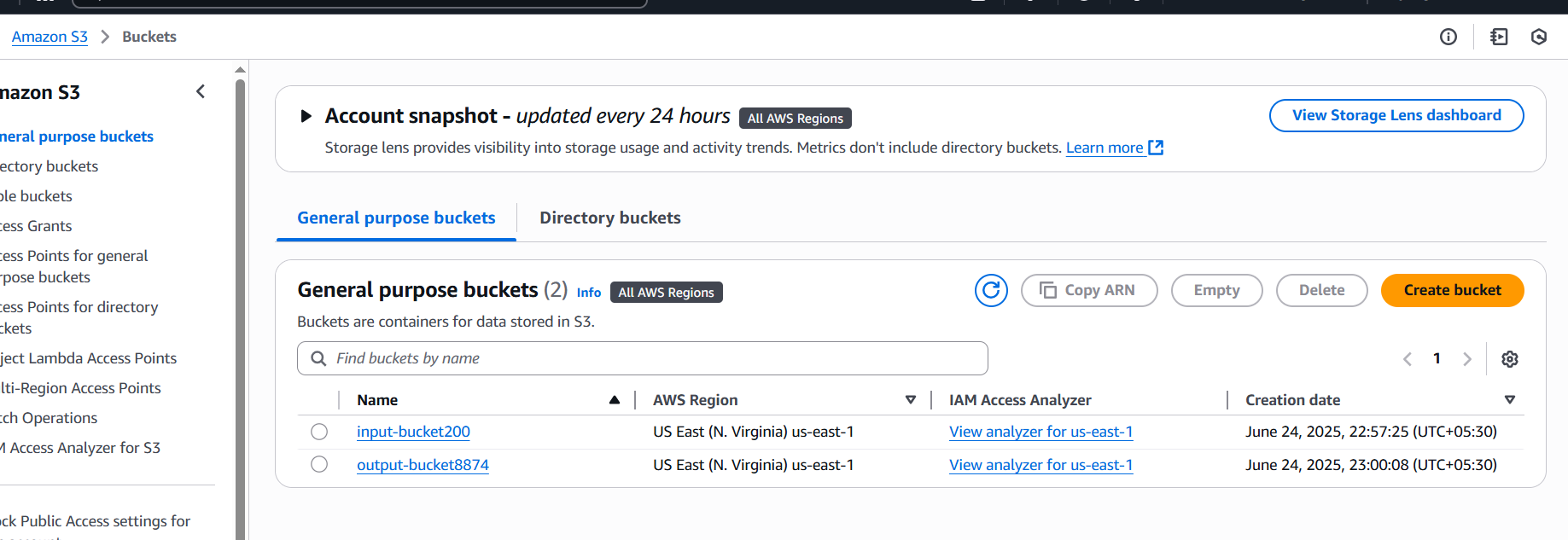
* Upload an image to the input-bucket.







* The Lambda function should automatically trigger, resize the image, and upload it to the output-bucket.





**3. Monitoring and Logging**

Monitor the Lambda Function:

* Use AWS CloudWatch to monitor the function’s execution and logs.
* Check for any errors or issues during processing.

**Thankyou….**