



Launch an EC2 Instance with Lambda

Advance AWS Cloud Computing With DevOps Fundamentals

Trainer : Mrs. Vinolin Jermiah

Advance AWS Cloud Computing With DevOps Fundamentals

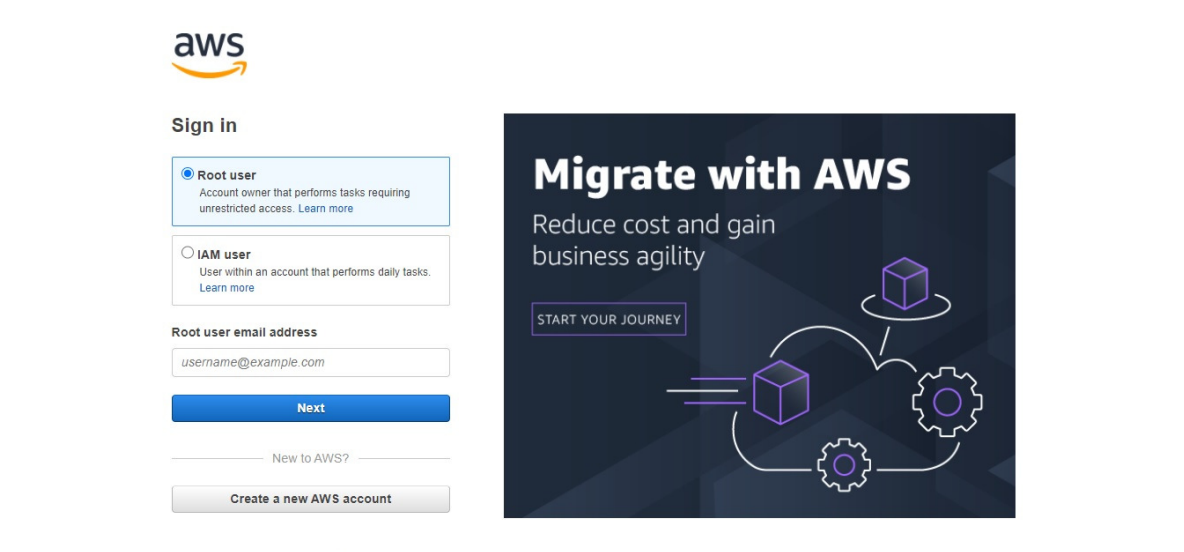
Project Question: Provision EC2 instance with Lambda

Tasks

1. Log into the AWS Management Console.
2. Create an IAM Policy and an IAM Role.
3. Create a lambda function.
4. Configure a test event.
5. Trigger the lambda function manually using the test event.
Test the new EC2 instance.

Advance AWS Cloud Computing With DevOps Fundamentals

Login to EC2 Console :



The image shows the AWS sign-in page. On the left, there is a 'Sign in' section with two options: 'Root user' (selected) and 'IAM user'. Below these is a text field for 'Root user email address' containing 'username@example.com'. A blue 'Next' button is below the field. At the bottom of the sign-in section are links for 'New to AWS?' and 'Create a new AWS account'. On the right, there is a large banner titled 'Migrate with AWS' with the text 'Reduce cost and gain business agility' and a 'START YOUR JOURNEY' button. The banner features a diagram of a cube, a gear, and a cloud.

aws

Sign in

☒ **Root user**
Account owner that performs tasks requiring unrestricted access. [Learn more](#)

☐ **IAM user**
User within an account that performs daily tasks. [Learn more](#)

Root user email address
username@example.com

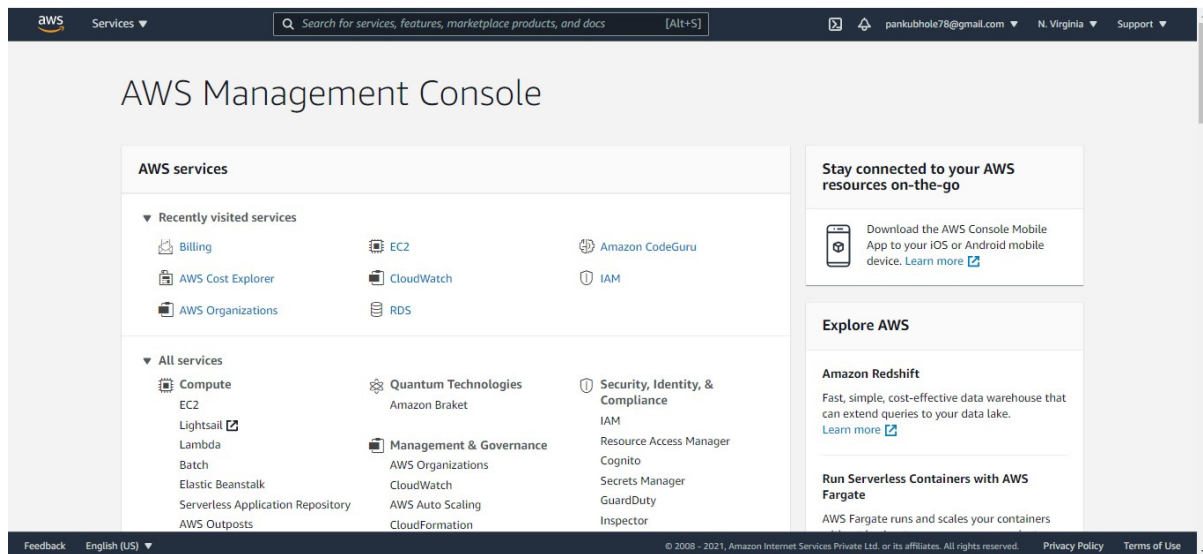
Next

[New to AWS?](#)

[Create a new AWS account](#)

Migrate with AWS
Reduce cost and gain business agility

[START YOUR JOURNEY](#)



The image shows the AWS Management Console dashboard. The top navigation bar includes the AWS logo, 'Services', a search bar, and user information. The main content area is titled 'AWS Management Console' and features a grid of service tiles. On the right, there are sidebars with links to mobile apps and featured services like Amazon Redshift and AWS Fargate.

aws Services [Alt+S] pankubhole78@gmail.com N. Virginia Support

AWS Management Console

AWS services

Recently visited services

- Billing
- AWS Cost Explorer
- AWS Organizations
- EC2
- CloudWatch
- RDS
- Amazon CodeGuru
- IAM

All services

- Compute**
 - EC2
 - Lightsail
 - Lambda
 - Batch
 - Elastic Beanstalk
 - Serverless Application Repository
 - AWS Outposts
- Quantum Technologies**
 - Amazon Braket
- Management & Governance**
 - AWS Organizations
 - CloudWatch
 - AWS Auto Scaling
 - CloudFormation
- Security, Identity, & Compliance**
 - IAM
 - Resource Access Manager
 - Cognito
 - Secrets Manager
 - GuardDuty
 - Inspector

Stay connected to your AWS resources on-the-go

Download the AWS Console Mobile App to your iOS or Android mobile device. [Learn more](#)

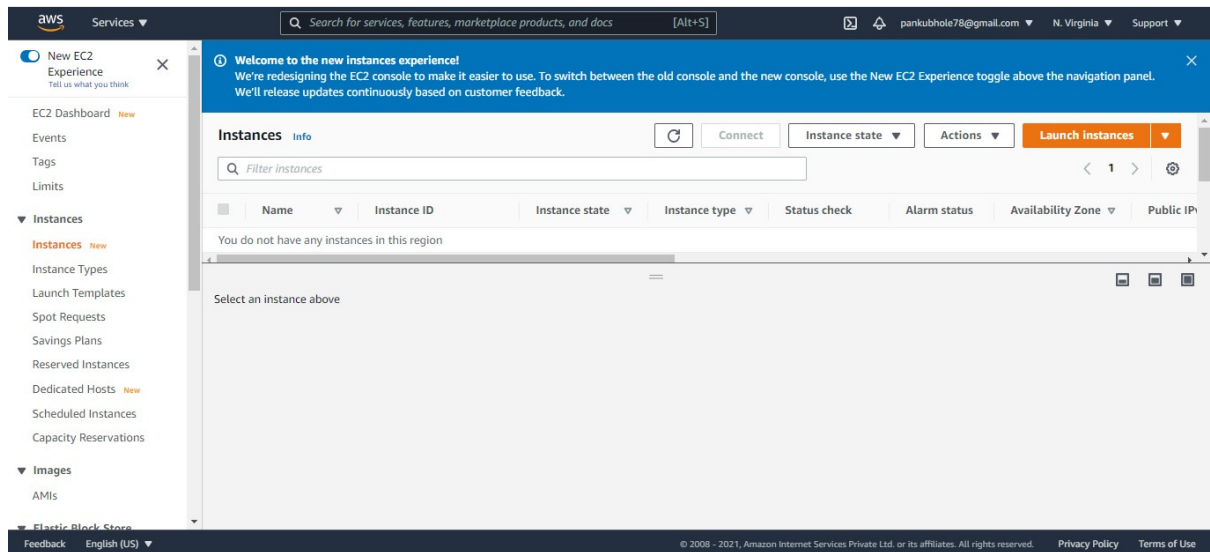
Explore AWS

Amazon Redshift
Fast, simple, cost-effective data warehouse that can extend queries to your data lake. [Learn more](#)

Run Serverless Containers with AWS Fargate
AWS Fargate runs and scales your containers

Feedback English (US) © 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Advance AWS Cloud Computing With DevOps Fundamentals



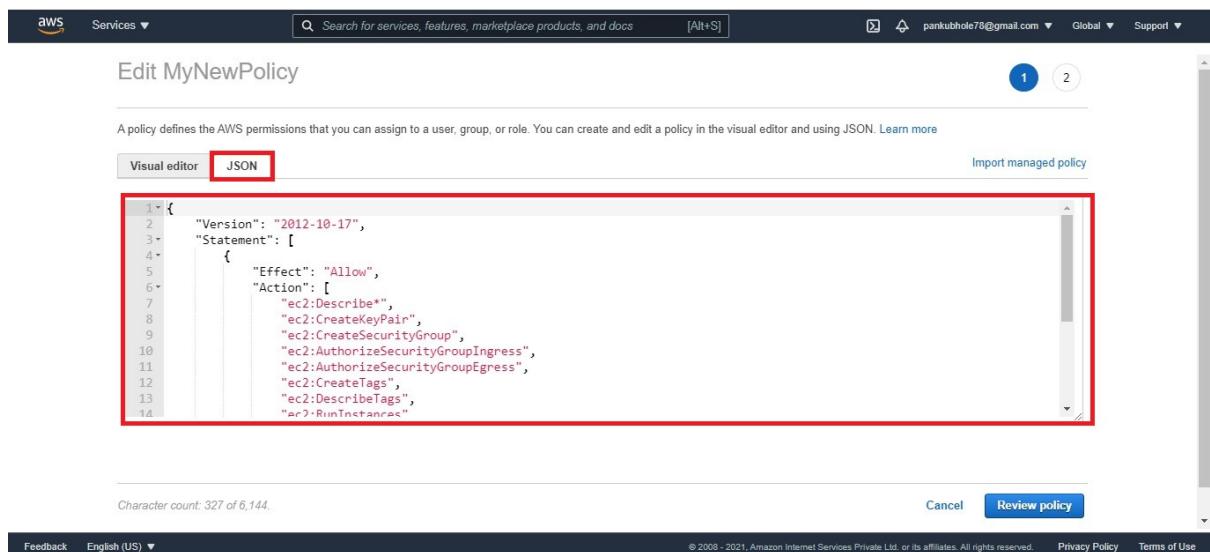
Create an IAM Policy

As a prerequisite for creating Lambda function, we need to create a user role with a custom policy.

Go to services and select IAM.

Click on policies and click create policy.

Click on the JSON tab and copy / paste the below policy statement into the editor:



Advance AWS Cloud Computing With DevOps Fundamentals

Policy JSON:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:Describe*",
        "ec2:CreateKeyPair",
        "ec2:CreateSecurityGroup",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:CreateTags",
        "ec2:DescribeTags",
        "ec2:RunInstances"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ec2:Region": "us-east-1"
        }
      }
    }
  ]
}
```

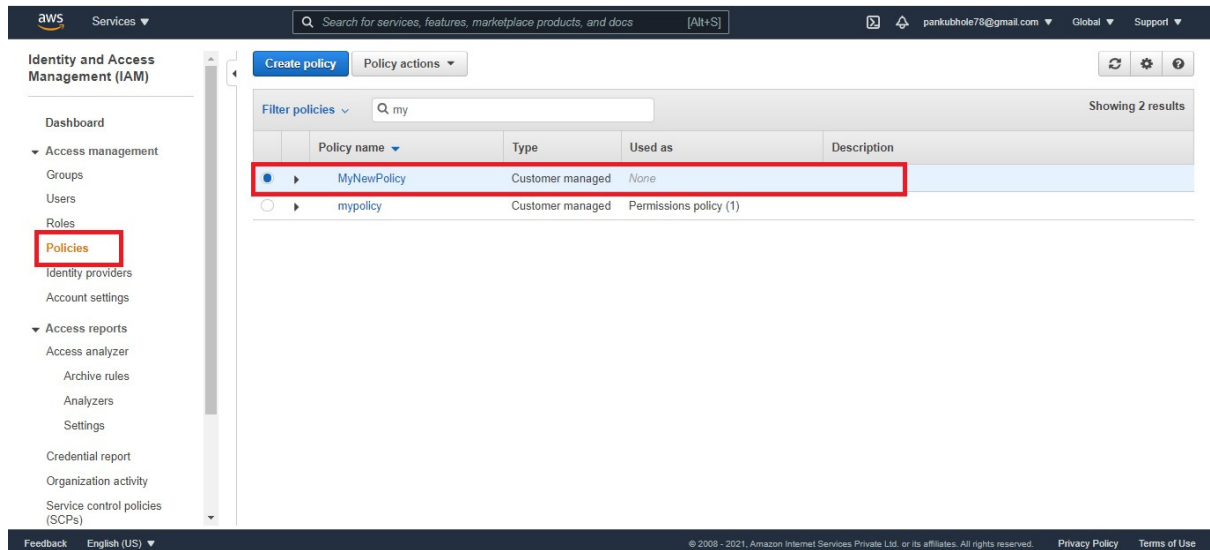
Click on Review policy.

On Create Policy page:

Policy Name: **MyNewpolicy.**

Click on the **create policy button.**

Advance AWS Cloud Computing With DevOps Fundamentals



An IAM Policy named **MyNewpolicy** is now created.

Create an IAM Role

In the left menu, click on Roles. Click on the create role button.

Select Lambda from the AWS Services list.

Click on Next: permissions.

Filter Policies: Search for your policy by name (**MyNewpolicy**).

Select your policy and click on the Next: tags.

Add Tags: Provide a key-value pair for the role:

Key: name

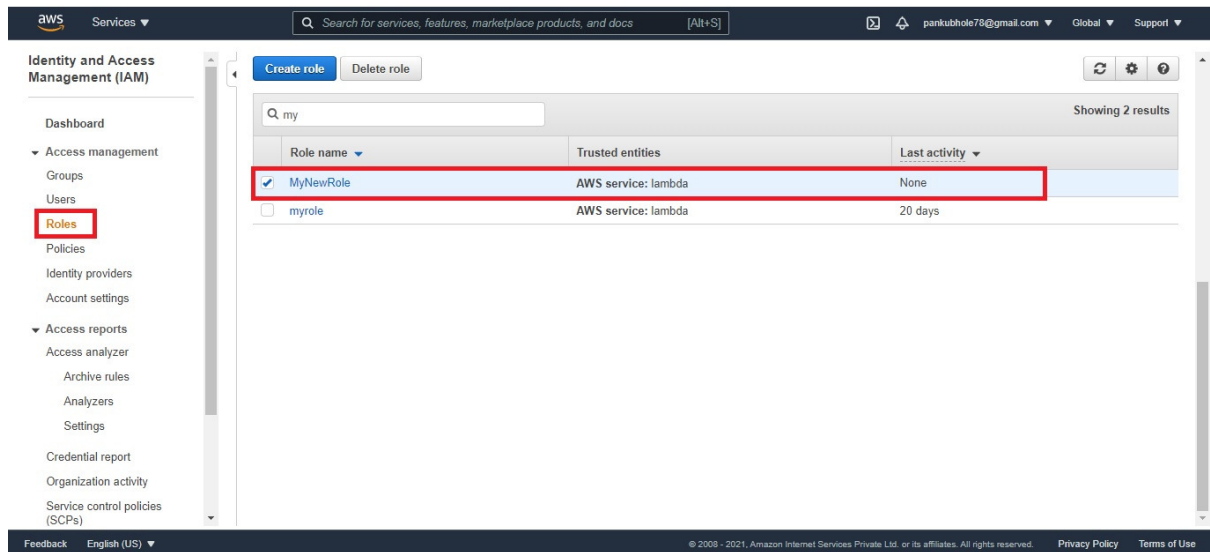
Value: **MyNewRole**

Click on the Next: review button.

Role name: **MyNewRole**

Click on the create role button.

Advance AWS Cloud Computing With DevOps Fundamentals



You have successfully created an IAM role by the name **MyNewRole**.

Create a Lambda Function

Go to the services menu and click on lambda.

Make sure you are in the **US East (N. Virginia) us-east-1 region**.

Click on the create function button.

Choose an author from scratch.

Function name: myEC2LambdaFunction

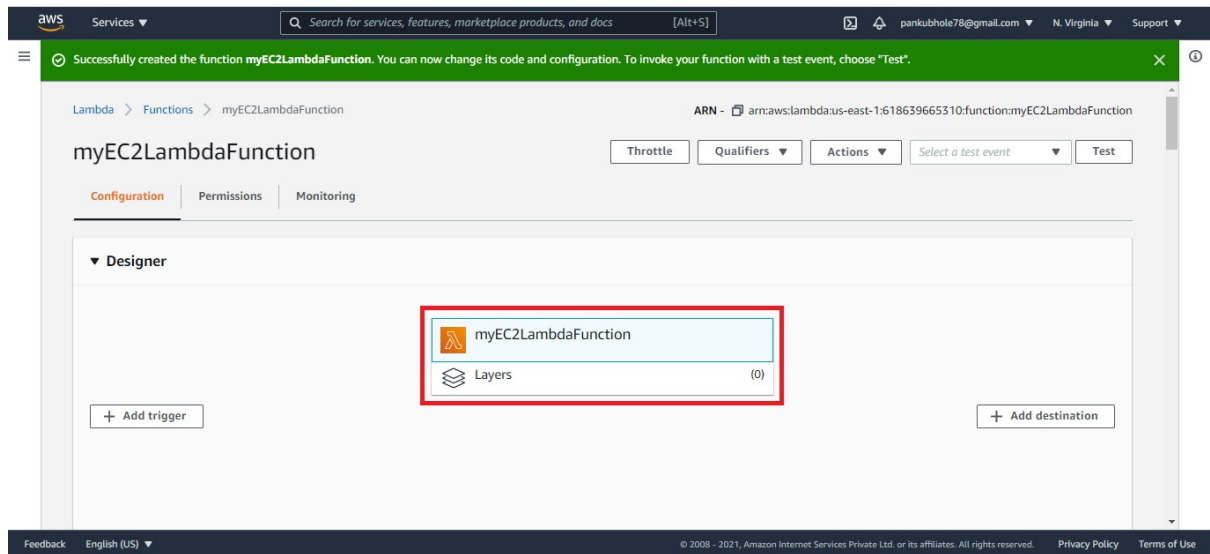
Runtime: Select Python 3.8

Role: In the permissions section, click on to use an existing role.

Existing role: Select MyNewRole

Click on **create function**.

Advance AWS Cloud Computing With DevOps Fundamentals

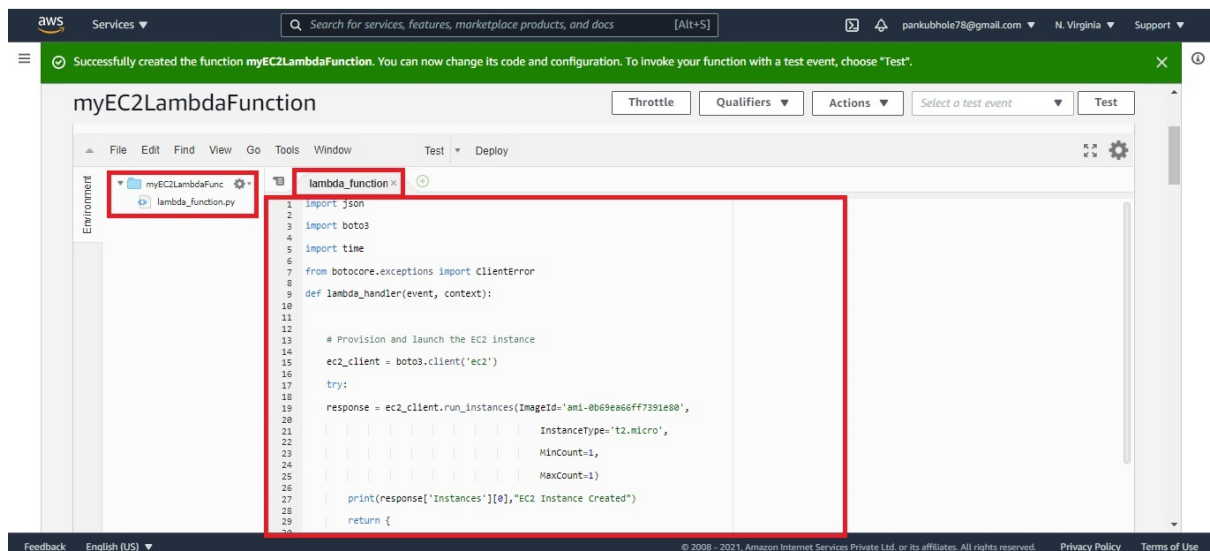


Configuration Page: Here we need to configure our lambda function. If you scroll down, you can see the

Function code section. Here we need to write some Python code to provision an EC2 instance.

You will be using boto3 SDK for AWS to write the python code.

Remove the existing code in the **lambda_function.py** file. **Copy the below code and paste it into your lambda_function.py file.**



Note: We will not get into the details of the python code in this lab, but at a high level, it will provision an EC2 instance when it is triggered.

Advance AWS Cloud Computing With DevOps Fundamentals

```
import json
import boto3
import time
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    # Provision and launch the EC2 instance
    ec2_client = boto3.client('ec2')

    try:
        response = ec2_client.run_instances(ImageId='ami-0b69ea66ff7391e80',
            InstanceType='t2.micro',
                                           MinCount=1,
                                           MaxCount=1)
        print(response['Instances'][0], "EC2 Instance Created")
        return {
            'statusCode': 200,
            'body': json.dumps("success")
        }
    except ClientError as e:
        print("Detailed error: ", e)
        return {
            'statusCode': 500,
            'body': json.dumps("error")
        }
    except Exception as e:
        print("Detailed error: ", e)
```

Advance AWS Cloud Computing With DevOps Fundamentals

```
return {  
    'statusCode': 500,  
    'body': json.dumps("error")  
}
```

7. Save the function by clicking on deploy in the top right corner.
8. Now scroll down the page and you will be able to see Basic settings click on the button.
9. In the **Timeout set seconds as 6 and then** click on the **save** button.

The screenshot shows the AWS Lambda console interface for configuring a function. The 'Basic settings' tab is active. The 'Description' field is optional. The 'Memory (MB)' is set to 128 MB. The 'Timeout' is set to 0 minutes and 6 seconds. Under 'Execution role', the option 'Use an existing role' is selected, and 'MyNewRole' is chosen from the dropdown menu. A red box highlights the 'Timeout' section, and another red box highlights the 'Existing role' section. The 'Save' button is visible at the bottom right.

Configure a Test Event

Click on the Test button at the top right corner of the page (under the ARN).

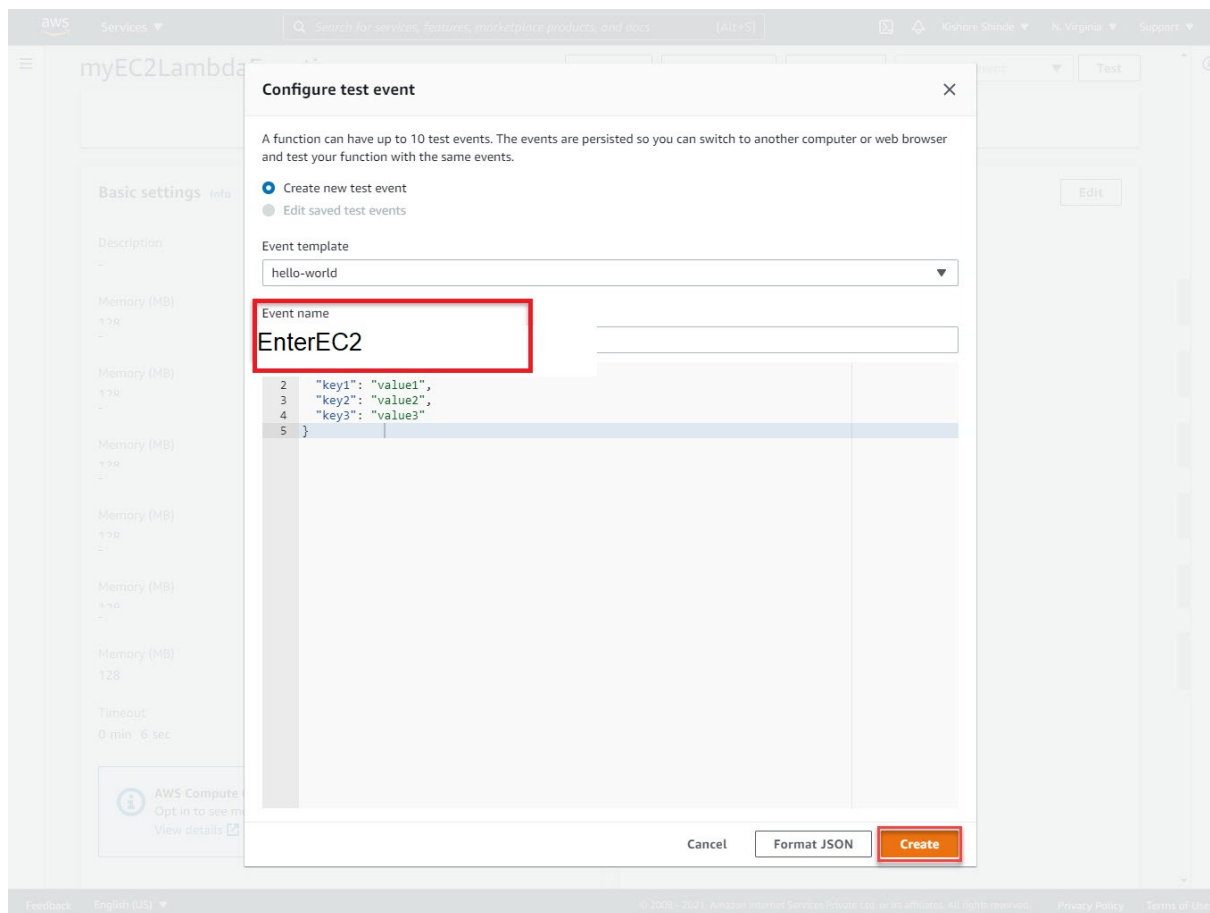
In Configure test event page,

Event Name: EnterEc2

Leave other fields as default.

Click on create.

Advance AWS Cloud Computing With DevOps Fundamentals

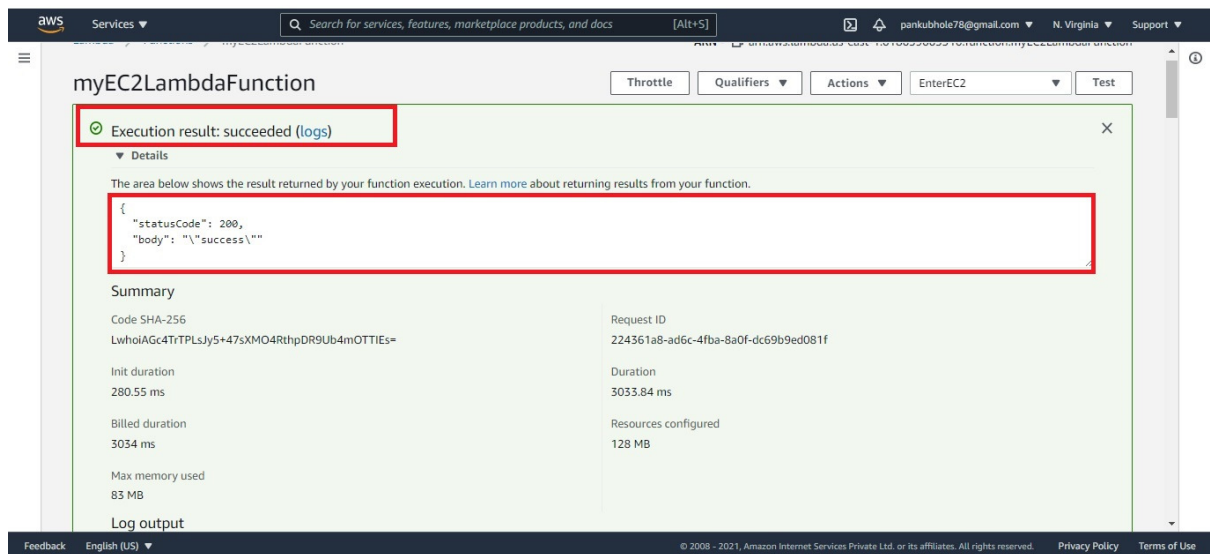


Provision an EC2 Instance using a Lambda Function

Once the ec2 is configured, we can trigger the lambda manually using this simple test.

Click on the Test button.

Advance AWS Cloud Computing With DevOps Fundamentals

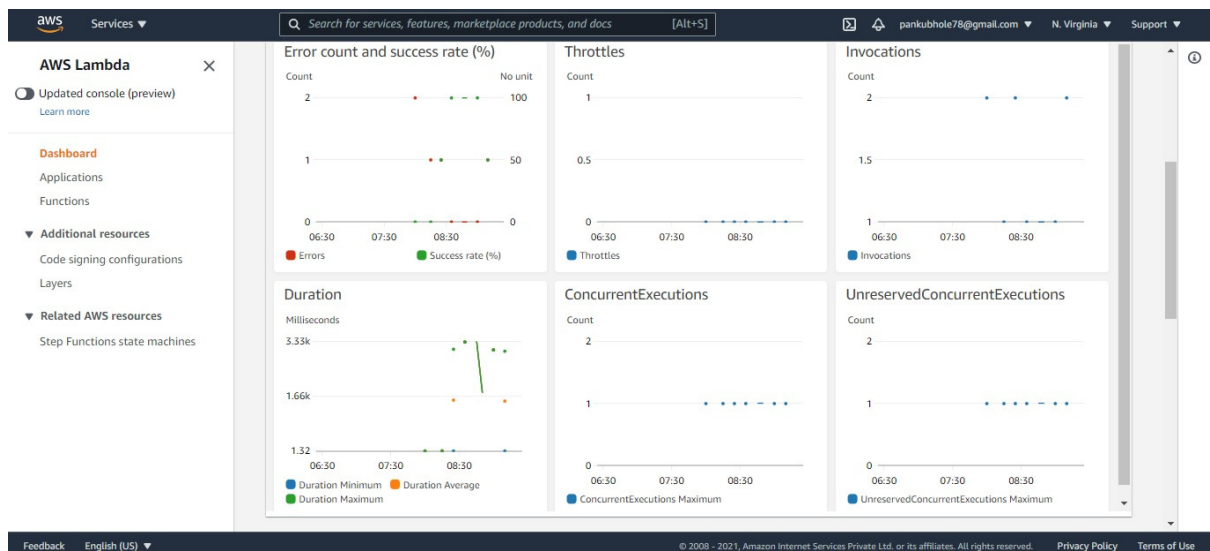


The lambda function now gets executed and an EC2 instance will be provisioned.

Once it's completed, you will be seeing a success message similar to the example shown below. It will

Display details such as:

Duration: Lambda execution time.



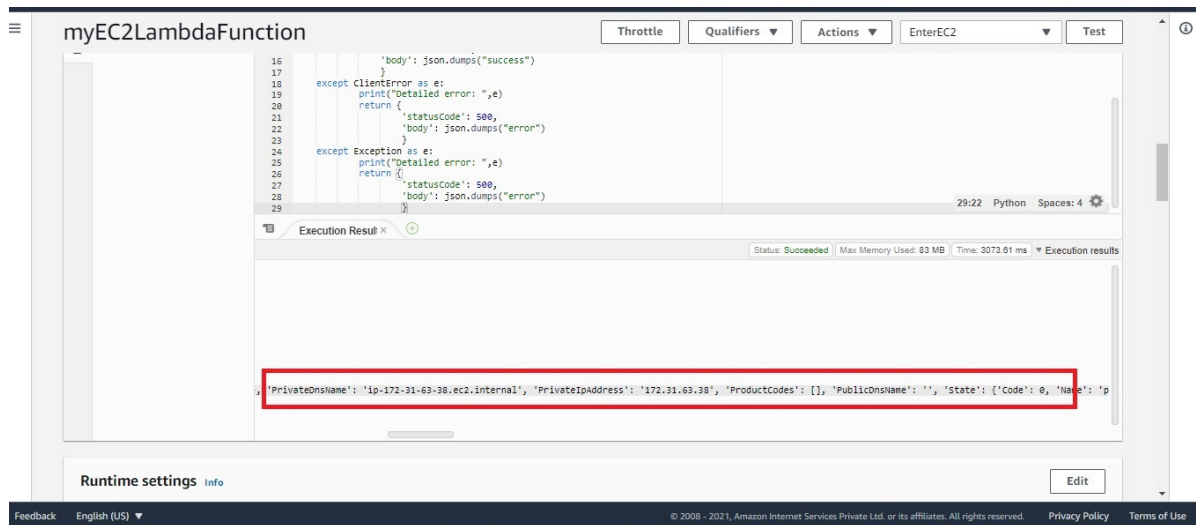
Log Output: Details of the EC2 instance provisioned.

etc...

Check that the EC2 instance launched successfully.

Navigate to the EC2 page from the services menu.

Advance AWS Cloud Computing With DevOps Fundamentals



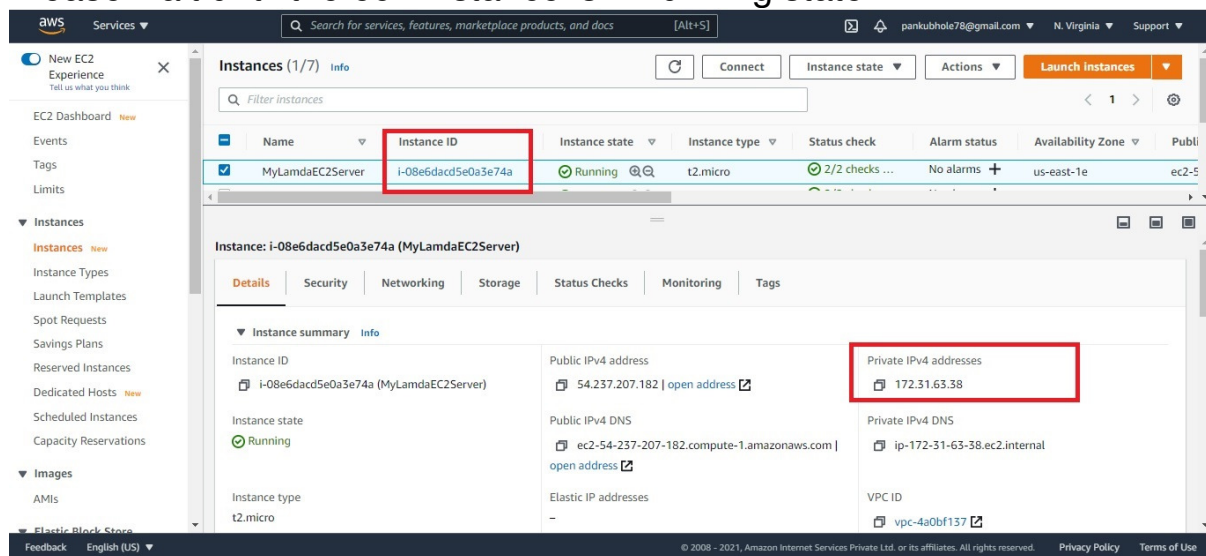
(IP Address: 172.31.63.38)

Switch off the New EC2 experience. Edit the feedback message and select yes for the experience. Click On.

This will allow us to use the old console.

Go to Instances in the left menu.

Please wait until the ec2 instance is in running state



IP Address: 172.31.63.38

You can see the EC2 instance that has been provisioned by the Lambda function