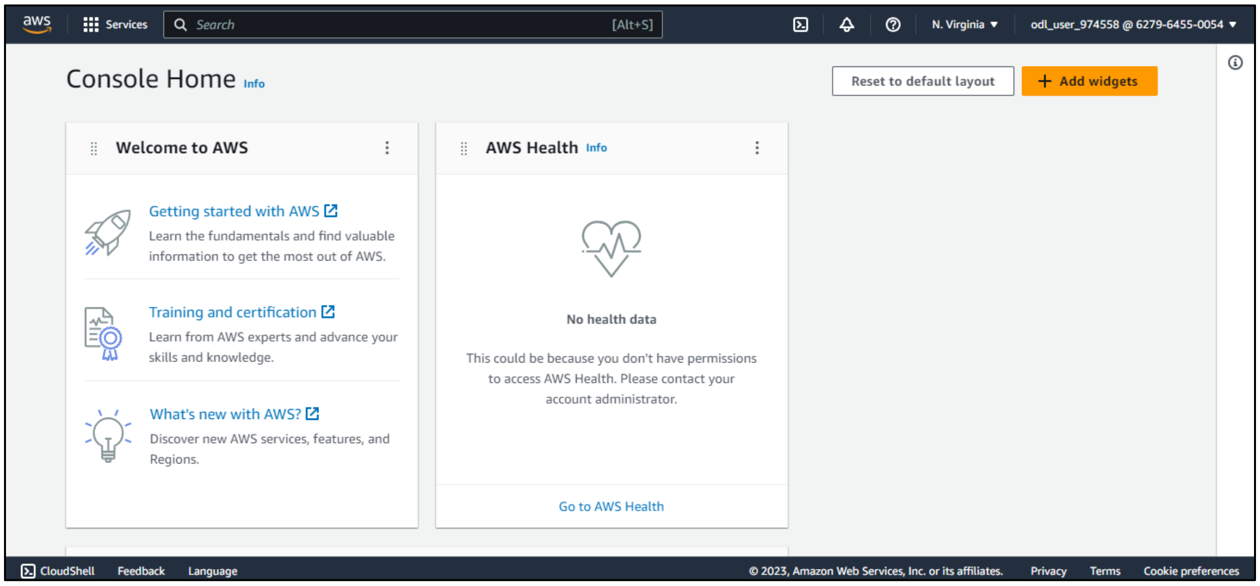
Terraform with AWS

Before we see Terraform in action, first let's check if we have EC2 instance using below steps,

Step 1: Login to AWS account



Learn the fundamentals and find valuable information to get the most out of AWS.

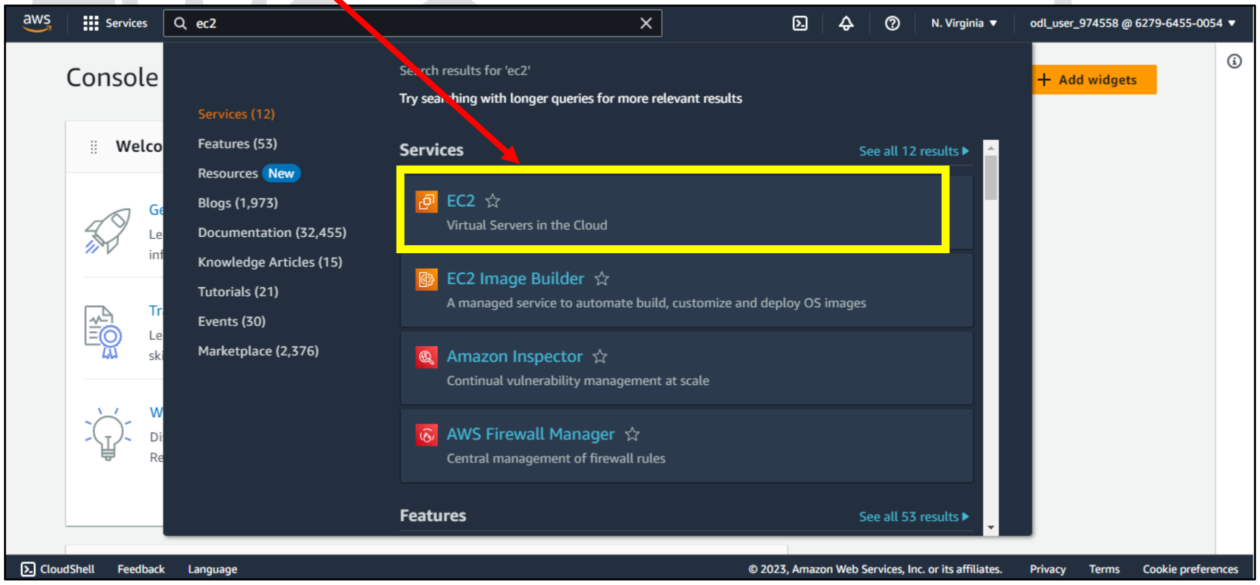
Learn from AWS experts and advance your skills and knowledge.

Discover new AWS services, features, and Regions.

This could be because you don't have permissions to access AWS Health. Please contact your account administrator.

© 2023, Amazon Web Services, Inc. or its affiliates.

Step 2: Search and goto **EC2** service.



Search results for 'ec2'

Try searching with longer queries for more relevant results

Virtual Servers in the Cloud

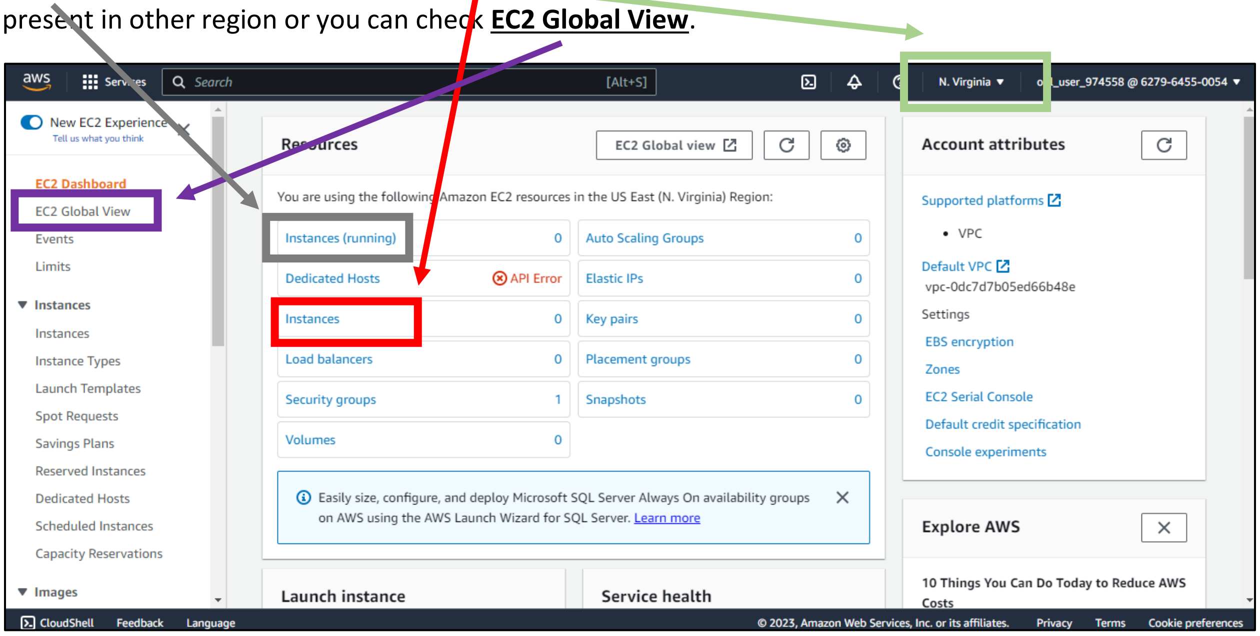
A managed service to automate build, customize and deploy OS images

Continual vulnerability management at scale

Central management of firewall rules

© 2023, Amazon Web Services, Inc. or its affiliates.

Step 3: Here you will see if you have any **instances** present in your account and which of them are **running** and other info. And look for the **region** which is selected, you can change the region if it is present in other region or you can check **EC2 Global View**.



You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

Instances (running)	0	Auto Scaling Groups	0
Dedicated Hosts	API Error	Elastic IPs	0
Instances	0	Key pairs	0
Load balancers	0	Placement groups	0
Security groups	1	Snapshots	0
Volumes	0		

vpc-0dc7d7b05ed66b48e

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

10 Things You Can Do Today to Reduce AWS Costs

And as of now, we don't have instances in our account.

So now, we'll create a EC2 instance using terraform and then we'll see if gets created or not using above steps.

Terraform In Action

Step 1: Install terraform on your machine and complete the setup.

Step 2: In order to connect to AWS using Terraform, terraform has to successfully authenticate. It is done with the help of Programmatic API Keys (Access Key and Secret). So we'll need to create or use the existing ones for authentication. To do that please check steps given in below doc.

aws_setup_progra
mmatic_access.txt

Step 3: Now we need to set these variables on our CLI, so we can use those for authentication.

If you are using **Windows**, then use below commands to set these variables -

```
set AWS_ACCESS_KEY_ID=JKAb************

set AWS_SECRET_ACCESS_KEY=Zasdak***************
```

If you are using **Linux**, then use below commands to set these variables -

```
export AWS_ACCESS_KEY_ID=JKAb************

export AWS_SECRET_ACCESS_KEY=Zasdak***************
```

Please check below logs for reference.

export_variables_cm
d_output.txt

Step 4: Now create a new file with tf extension, for e.g. main.tf. You can create this file in your current directory or desired directory.

aws_ec2.tf

Step 5: Now run terraform init command. It initializes a new or existing Terraform configuration. It downloads the necessary provider plugins and sets up the backend configuration.

Please check below logs for reference.

terraform_init_cmd_output.txt

Step 6: Now run terraform plan command. It will create an execution plan that shows the changes Terraform will make to the infrastructure. It compares the desired state in the Terraform configuration with the current state and displays a list of actions that will be taken.

Please check below logs for reference.

terraform_plan_cmd_output.txt

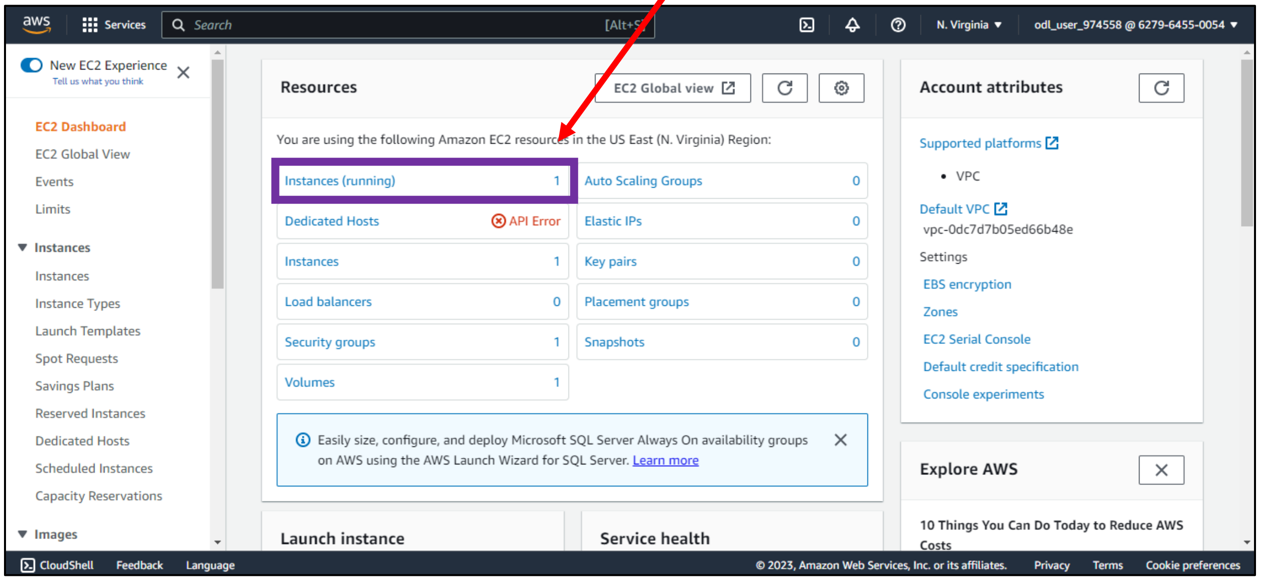
Step 7: Now run terraform apply command. It will apply the changes described in the execution plan. Creates, modifies, or destroys resources to match the desired state defined in the configuration.

Please check below logs for reference.

terraform_apply_cmd_output.txt

Step 8: Again check the first steps mentioned in top section of document to verify if resource got created or not.

Goto EC2 service window. And now you can see the **EC2 instance** got created, so click on it.



You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

Instances (running)	1	Auto Scaling Groups	0
Dedicated Hosts	API Error	Elastic IPs	0
Instances	1	Key pairs	0
Load balancers	0	Placement groups	0
Security groups	1	Snapshots	0
Volumes	1		

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

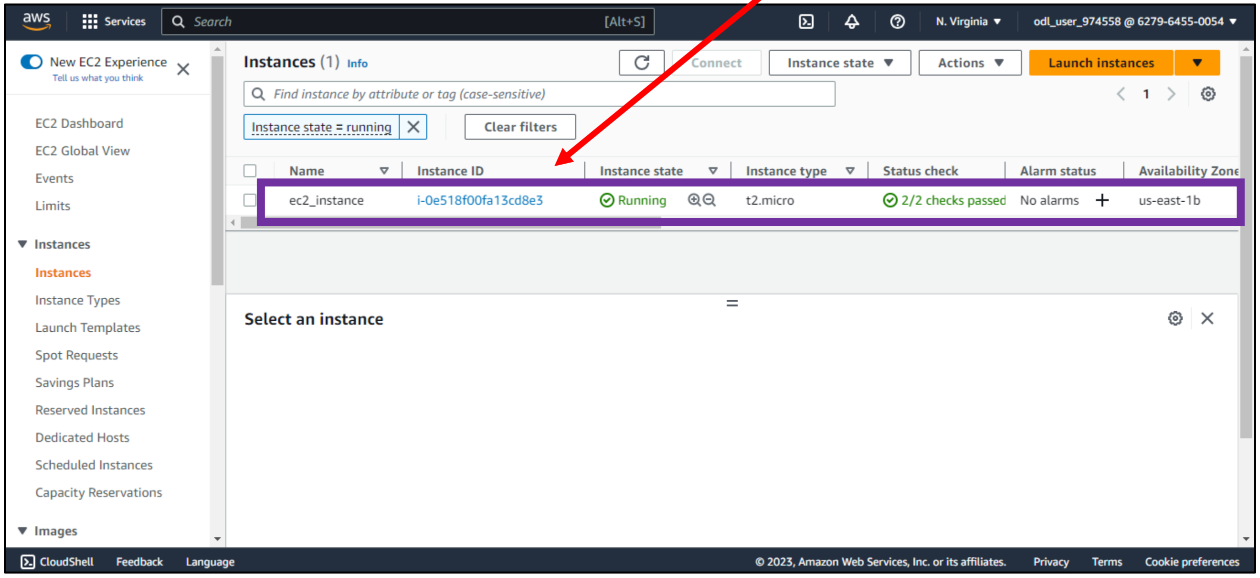
• VPC

vpc-0dc7d7b05ed66b48e

10 Things You Can Do Today to Reduce AWS Costs

© 2023, Amazon Web Services, Inc. or its affiliates.

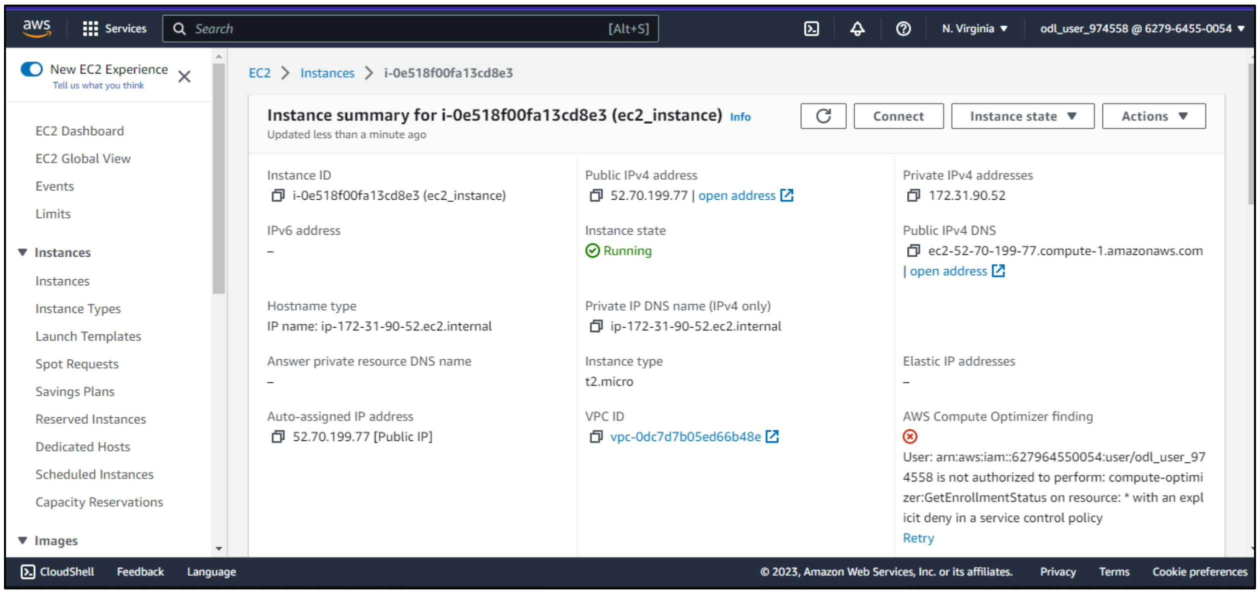
And here you can see EC2 instance got created with name **ec_instance** which we gave in main.tf file



Instances (1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
ec2_instance	i-0e518f00fa13cd8e3	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b

Don't try to connect to this EC2 instance, as the SSH rule is not added in NSG.



Instance summary for i-0e518f00fa13cd8e3 (ec2_instance) Info
Updated less than a minute ago

Instance ID: i-0e518f00fa13cd8e3 (ec2_instance)
Public IPv4 address: 52.70.199.77 | open address
Private IPv4 addresses: 172.31.90.52
IPv6 address: –
Instance state: Running
Public IPv4 DNS: ec2-52-70-199-77.compute-1.amazonaws.com | open address
Hostname type: IP name: ip-172-31-90-52.ec2.internal
Private IP DNS name (IPv4 only): ip-172-31-90-52.ec2.internal
Answer private resource DNS name: –
Instance type: t2.micro
Elastic IP addresses: –
Auto-assigned IP address: 52.70.199.77 [Public IP]
VPC ID: vpc-0dc7d7b05ed66b48e
AWS Compute Optimizer finding: User: arn:aws:iam::627964550054:user/odl_user_974558 is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * with an explicit deny in a service control policy
Retry