**Lab Steps**

Task 1: Sign in to AWS Management Console

1. Click on the button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

* Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
* Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button

     3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

Task 2: Setup Visual Studio Code

1. Open the visual studio code.
2. If you have already installed and using Visual studio code, open a new window.
3. A new window will open a new file and release notes page (only if you have installed or updated Visual Studio Code recently). Close the Release notes tab.
4. Open Terminal by selecting View from the Menu bar and choose Terminal.
5. It may take up to 2 minutes to open the terminal window.
6. Once the terminal is ready, let us navigate to the Desktop.

cd Desktop

1. Create a new folder by running the below command.

mkdir task\_10107

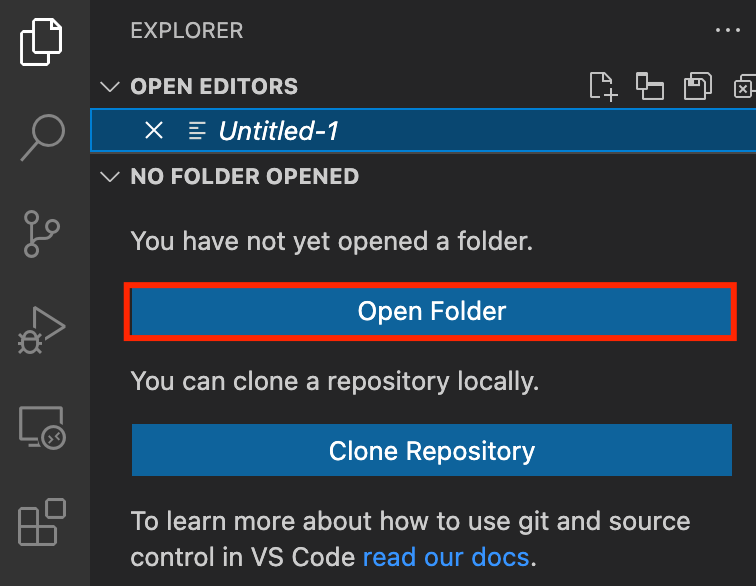
1. Change your present working directory to use the newly created folder by running the below command:

cd task\_10107

1. Get the location of the present working directory by running the below command:

pwd

1. Note down the location, as you will open the same in the next steps.
2. Now click on the first icon Explorer present on the left sidebar.
3. Click on the button called Open folder and navigate to the location of folder **task\_10107**.



1. (Optional) Click on Authorize button for allowing Visual Studio Code to use the task\_10107 folder. This will only be asked when you have been using Visual Studio code for a while as you are allowing a new folder to be accessed by VSC.
2. Visual Studio Code is now ready to use.

Task 3: Create a variable file

In this task, you will create variable files where you will declare all the global variables with a short description and a default value.

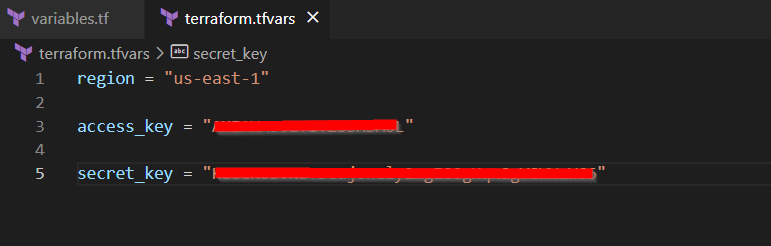
1. To create a variable file, expand the folder **task\_10107** and click on the **New** **File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note:** Don't change the location of the new file, keep it default, i.e. inside the **task\_10107** folder**.**
4. Paste the below contents in **variables.tf** file.

|  |
| --- |
| variable "access\_key" {      description = "Access key to AWS console"  }  variable "secret\_key" {      description = "Secret key to AWS console"  }  variable "region" {      description = "AWS region"  } |

1. In the above content, you are declaring a variable called, access\_key, secret\_key, and region with a short description of all 3.
2. After pasting the above contents, save the file by pressing **ctrl + S**.
3. Now expand the folder **task\_10107** and click on the **New File** icon to add the file.
4. Name the file as **terraform.tfvars** and press **Enter** to save it.
5. Paste the below content into the **terraform.tfvars** file.

|  |
| --- |
| region = "us-east-1"  access\_key = "<YOUR AWS CONSOLE ACCESS ID>"  secret\_key = "<YOUR AWS CONSOLE SECRET KEY>" |

1. In the above code, you are defining the dynamic values of variables declared earlier.
2. Replace the values of access\_key and secret\_key by copying from the lab page.
3. After replacing the values of access\_key and secret\_key, save the file by pressing Ctrl + S.



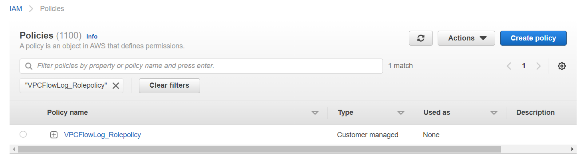
Task 4: Create IAM Role for EC2 in main.tf file

 In this task, you will create a **main.tf** file where you will add details of the provider and resources.

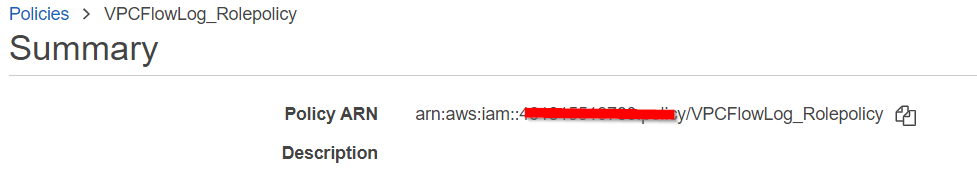
1. To create a **main.tf** file, expand the folder **task\_10107** and click on the **New** **File** icon to add the file.
2. Name the file as **main.tf** and press **Enter** to save it.
3. Paste the below content into the **main.tf** file.

|  |
| --- |
| provider "aws" {      region     = "${var.region}"      access\_key = "${var.access\_key}"      secret\_key = "${var.secret\_key}"  } |

1. In the above code, you are defining the provider as aws.
2. Next, we want to tell Terraform to create an IAM Role for EC2 Instance
3. In Console navigate to **IAM** by clicking on **Services** on the top, then click on **IAM**.
4. Click on the **policies** on the left navigation panel and search **VPCFlowLog\_Rolepolicy**



1. **Click on VPCFlowLog\_Rolepolicy and copy the Policy ARN**



1. To create an IAM Role Paste the below content into the **main.tf** file after the provider

|  |
| --- |
| #################### Creating Instance Profile ######################  resource "aws\_iam\_instance\_profile" "VPCFlowLog\_profile" {      name = "VPCFlowLog\_Instance\_profile"      role = "${aws\_iam\_role.VPCFlowLog\_Role.name}"  }  ####################### Creating IAM Role for EC2 #######################  resource "aws\_iam\_role" "VPCFlowLog\_Role" {    name = "VPCFlowLog\_Role"    assume\_role\_policy = <<EOF  {      "Version": "2012-10-17",      "Statement": [          {              "Sid": "",              "Effect": "Allow",              "Action": "sts:AssumeRole",              "Principal": {                  "Service": "vpc-flow-logs.amazonaws.com"              }          }      ]  }  EOF  }  # Attaching IAM Policy to IAM Role  resource "aws\_iam\_role\_policy\_attachment" "test-attach" {    role       = aws\_iam\_role.VPCFlowLog\_Role.name    policy\_arn = "<Paste the Policy ARN here"  } |

1. This Terraform code is defining an AWS IAM instance profile and role for the purpose of enabling VPC Flow Logs.
2. Replace the policy\_arn with **VPCFlowLog\_Rolepolicy** ARN

Task 5: Create CloudWatch log group in main.tf file

In this task, you will create a CloudWatch log group in main.tf file

1. To create a CloudWatch log group add another block of code just below the IAM code into the **main.tf** file

|  |
| --- |
| # Create CloudWatch Logs group  resource "aws\_cloudwatch\_log\_group" "log" {    name = "whizvpclogs"  } |

1. Save the file by pressing **Ctrl + S**.

Task 6: Create VPC, Internet Gateway and Subnet in main.tf file

In this task, you will create a VPC, Internet Gateway and Subnet in main.tf file

1. To create a VPC, Internet Gateway and Subnet add another block of code just below the CloudWatch log group code into the **main.tf** file

|  |
| --- |
| # Create a VPC  resource "aws\_vpc" "vpc" {    cidr\_block = "10.1.0.0/16"    tags = {      Name = "MyVPC"    }  }  # Create an Internet Gateway  resource "aws\_internet\_gateway" "igw" {    vpc\_id = "${aws\_vpc.vpc.id}"    tags = {      Name = "MyInternetGateway"    }  }  # Adding route  resource "aws\_route" "route" {      route\_table\_id         = aws\_vpc.vpc.main\_route\_table\_id      destination\_cidr\_block = "0.0.0.0/0"      gateway\_id             = aws\_internet\_gateway.igw.id  }  # Create a Subnet  resource "aws\_subnet" "subnet" {    vpc\_id     = aws\_vpc.vpc.id    cidr\_block = "10.1.1.0/24"    availability\_zone = "us-east-1a"     tags = {      Name = "whizsub"    }  } |



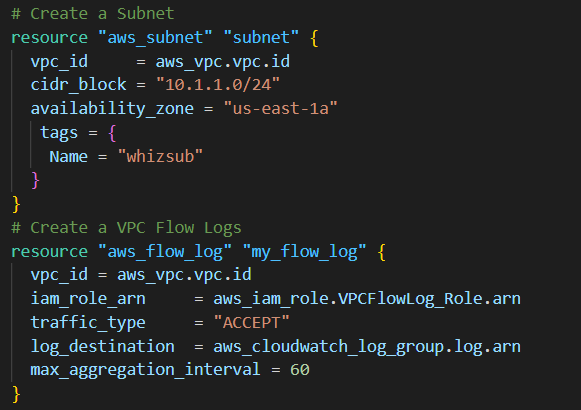
1. This Terraform code creates a Virtual Private Cloud (VPC) in AWS, an Internet Gateway to allow the VPC to communicate with the internet, and a Subnet within the VPC.
2. Save the file by pressing **Ctrl + S**.

Task 7: Create VPC Flow Logs in main.tf file

In this task, you will create a VPC Flow Logs in main.tf file

1. To create a VPC Flow Logs add another block of code just below the subnet code into the **main.tf** file

|  |
| --- |
| # Create a VPC Flow Logs  resource "aws\_flow\_log" "my\_flow\_log" {    vpc\_id = aws\_vpc.vpc.id    iam\_role\_arn     = aws\_iam\_role.VPCFlowLog\_Role.arn    traffic\_type     = "ACCEPT"    log\_destination  = aws\_cloudwatch\_log\_group.log.arn    max\_aggregation\_interval = 60  } |



1. The code creates a VPC flow log. A VPC flow log is a network traffic monitoring tool that captures information about the IP traffic going to and from network interfaces in a Virtual Private Cloud (VPC).
2. Save the file by pressing **Ctrl + S**.

Task 8: Create a Security group and key pair for EC2 in main.tf file

In this task, you will create a Security group and key pair for EC2 instance in main.tf file

1. To create a security group and key pair Paste the below content into the **main.tf** file after the provider

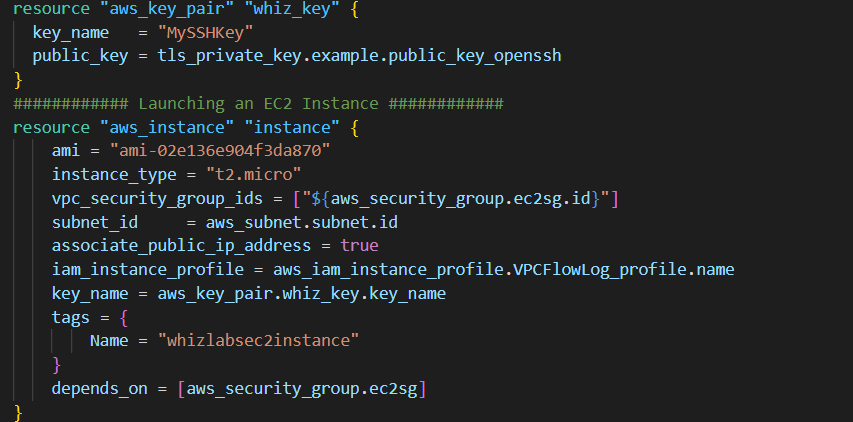
|  |
| --- |
| # Create Security group for EC2  resource "aws\_security\_group" "ec2sg" {    name        = "whiz\_sg"    description = "whizlabssecuritygroup"    vpc\_id      = aws\_vpc.vpc.id    ingress {      from\_port        = 22      to\_port          = 22      protocol         = "tcp"      cidr\_blocks      = ["0.0.0.0/0"]    }    ingress {      from\_port        = 80      to\_port          = 80      protocol         = "tcp"      cidr\_blocks      = ["0.0.0.0/0"]    }    egress {      from\_port        = 0      to\_port          = 0      protocol         = "-1"      cidr\_blocks      = ["0.0.0.0/0"]    }    tags = {      Name = "whiz\_sg"    }  }  ############ Creating Key pair for EC2 ############  resource "tls\_private\_key" "example" {    algorithm = "RSA"    rsa\_bits  = 4096  }  resource "aws\_key\_pair" "whiz\_key" {    key\_name   = "MySSHKey"    public\_key = tls\_private\_key.example.public\_key\_openssh  } |

1. Save the file by pressing **Ctrl + S**.

Task 9: Launch an EC2 Instance in main.tf file

1. To Launch an EC2 Instance add another block of code just below the key pair code into the **main.tf** file

|  |
| --- |
| ############ Launching an EC2 Instance ############  resource "aws\_instance" "instance" {      ami = "ami-02e136e904f3da870"      instance\_type = "t2.micro"      vpc\_security\_group\_ids = ["${aws\_security\_group.ec2sg.id}"]      subnet\_id     = aws\_subnet.subnet.id      associate\_public\_ip\_address = true      iam\_instance\_profile = aws\_iam\_instance\_profile.VPCFlowLog\_profile.name      key\_name = aws\_key\_pair.whiz\_key.key\_name      tags = {          Name = "whizlabsec2instance"      }      depends\_on = [aws\_security\_group.ec2sg]  } |



1. Save the file by pressing **Ctrl + S**.

Task 10: Create an Output file

In this task, you will create an **output.tf** file where you will add details of the provider and resources.

1. To create an **output.tf** file, expand the folder **task\_10107** and click on the **New** **File** icon to add the file.
2. Name the file as **output.tf** and press **Enter** to save it.
3. Paste the below content into the **output.tf** file.

|  |
| --- |
| output "vpc\_id" {    value       = aws\_vpc.vpc.id  }  output "igw\_id" {    value       = aws\_internet\_gateway.igw.id  }  output "subnet\_id" {    value       = aws\_subnet.subnet.id  }  output "vpc\_flow\_log\_id" {    value       = aws\_flow\_log.my\_flow\_log.id  }  output "instance\_id" {    description = "ID of the EC2 instance"    value       = aws\_instance.instance.id  } |

1. In the above code, we will extract details of resources created to confirm that they are created.

Task 11: Confirm the installation of Terraform by checking the version

1. In the Visual Studio Code, open Terminal by selecting **View** from the Menu bar and choose **Terminal**.
2. If you are not in the newly created folder change your present working directory by running the below command.

cd task\_10107

1. To confirm the installation of Terraform, run the below command to check the version:

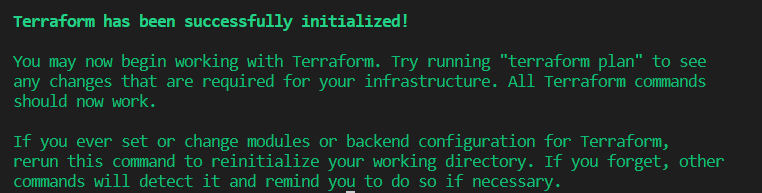
terraform version

1. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

Task 12: Apply terraform configurations

1. Initialize Terraform by running the below command,

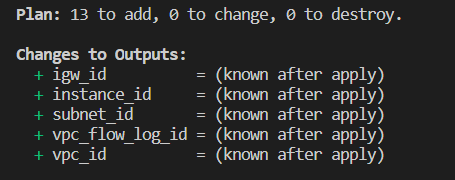
terraform init

****

**Note:** terraform init will check for all the plugin dependencies and download them if required, this will be used for creating a deployment plan

1. To generate the action plans run the below command,

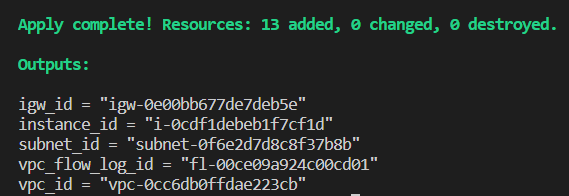
terraform plan



1. To create all the resources declared in main.tf configuration file, run the below command:

terraform apply

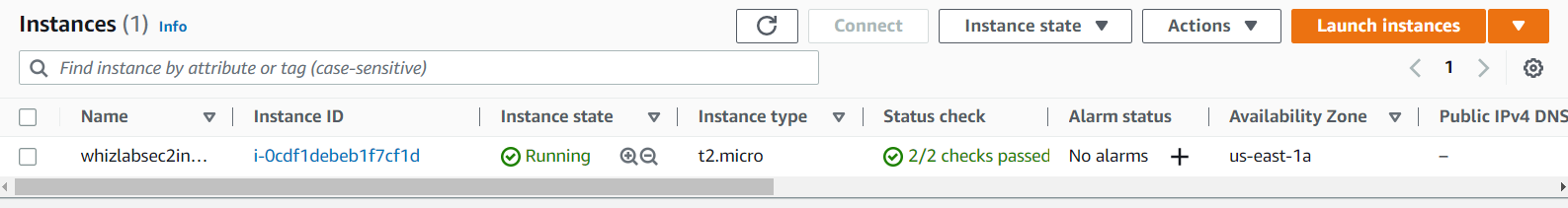
1. Approve the creation of all the resources by entering **yes**.



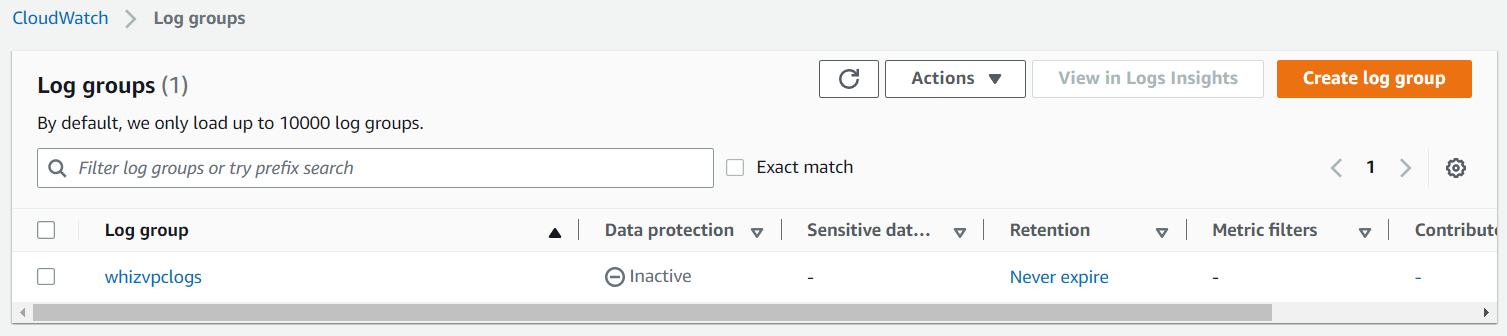
1. It may take up to 2-5 minutes for the terraform apply command to create the resources.
2. Id’s of all the resources created by terraform will be visible there.

Task 13: Check the resources in AWS Console

1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **EC2** by clicking on **Services** on the top, then click on **EC2** in the **Compute** section.
3. Click on the **Instances** on the left navigation panel. You can see the instance created successfully.

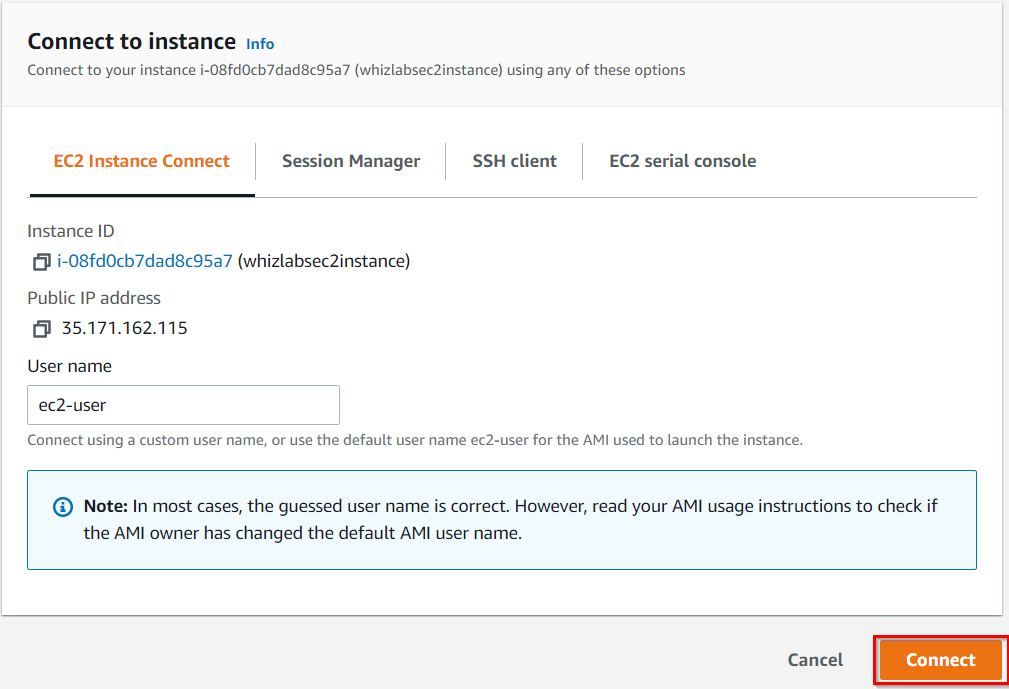


1. To Check Log groups creation choose **CloudWatch** under **Management and Governance**.
2. Click on the **log groups** on the left navigation panel. You can see that the log group is created successfully.

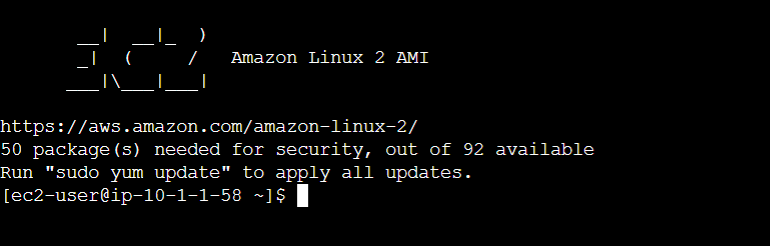


Task 14: Generating Traffic

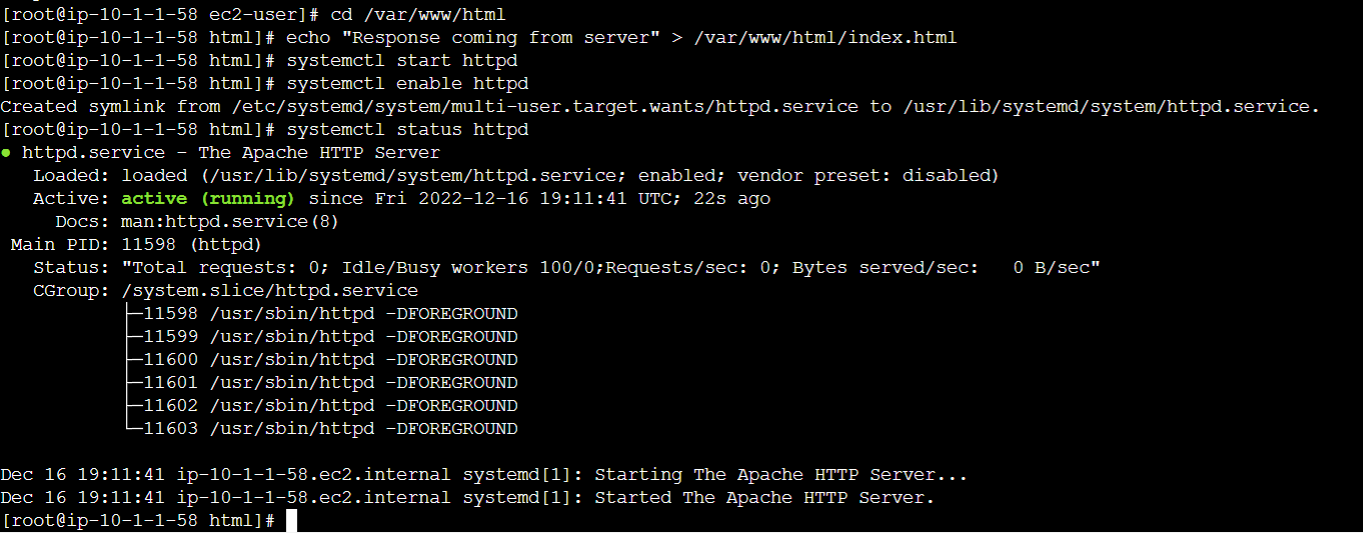
1. Navigate to the EC2 Dashboard and select the **whizlabsec2instance**
2. Click on button to SSH into the EC2 instance
3. In Connect to Instance page select EC2 Instance Connect tab and click on **Connect** button



1. Once you SSH into the instance, install an Apache Server. To install it, follow the below steps. Run these commands one-by-one.



* sudo su
* yum -y update
* yum install httpd -y
* cd /var/www/html
* echo "Response coming from server" > /var/www/html/index.html
* systemctl start httpd
* systemctl enable httpd
* systemctl status httpd

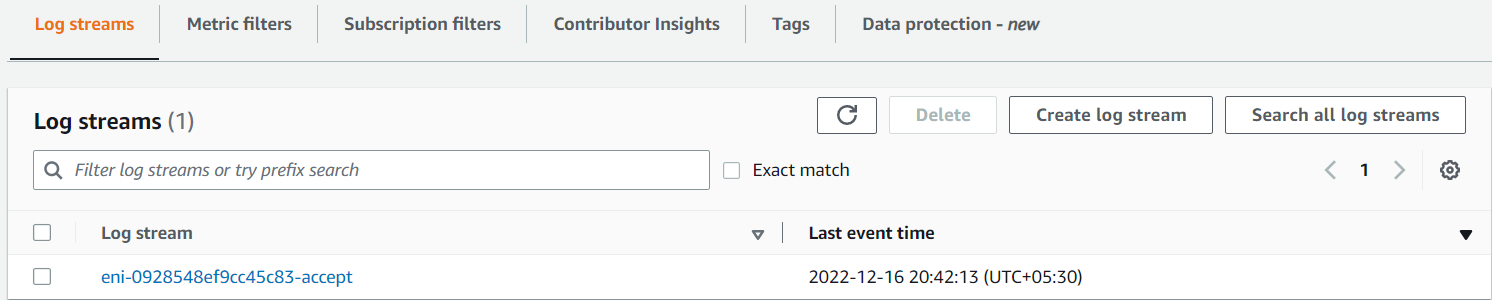
****

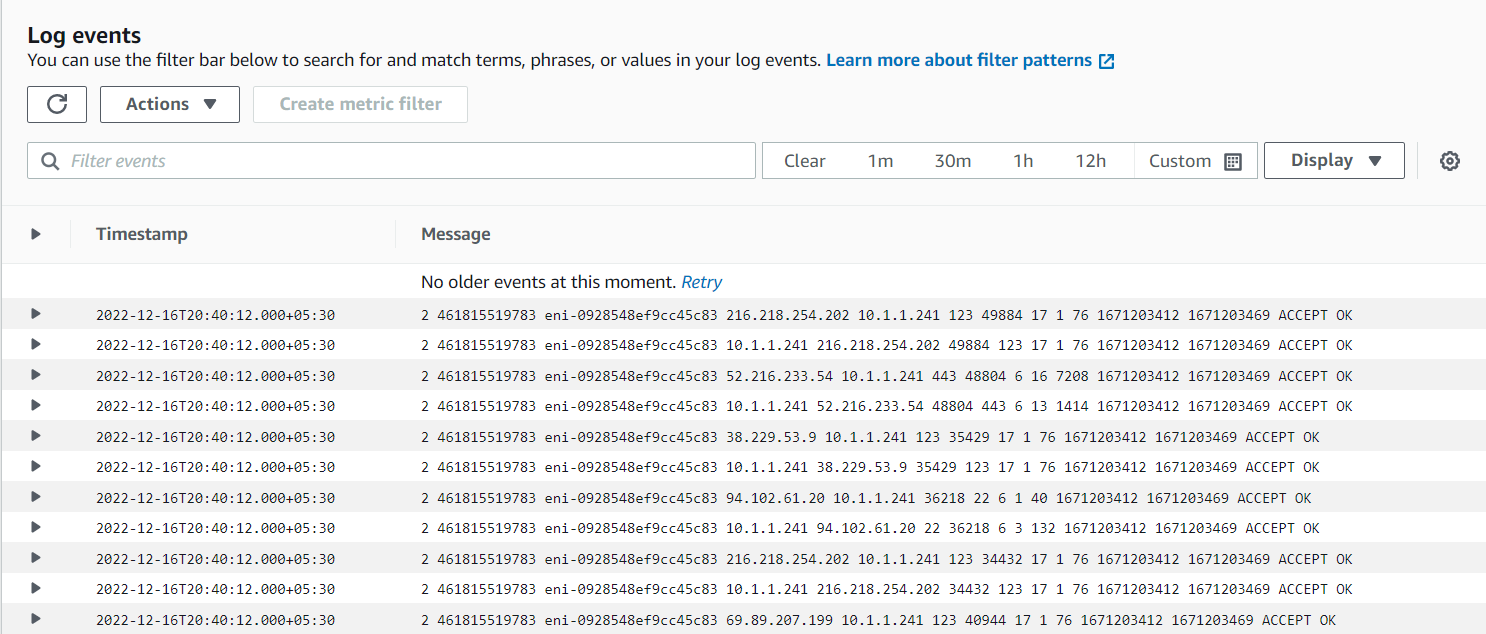
1. Copy your instance's **Public IP**, paste it into your browser and hit enter.



Task 15: Viewing log events in Cloudwatch Log groups.

1. Navigate to the **Services** menu at the top and choose **CloudWatch** under **Management and Governance**.
2. Click on **Log Groups** in the left side panel and click on **whizvpclogs**
3. Go to **Log streams** section and click on the log stream that has been created [In my case - **eni-0928548ef9cc45c83-accept**]



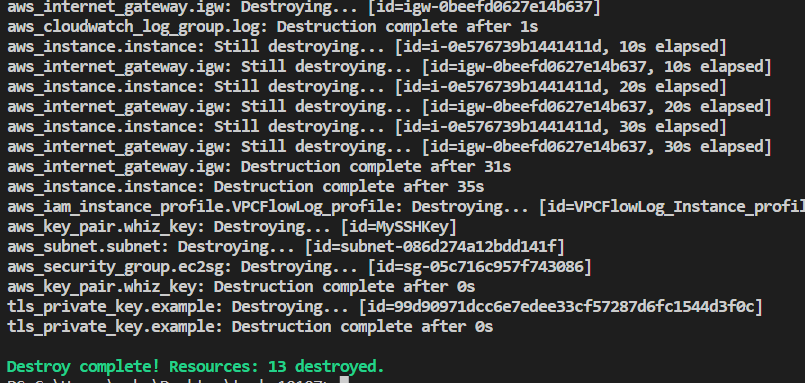
1. You will be able to view all the log events i.e the traffic that is being directed to our EC2 Instance.  
     

Task 16: Delete AWS Resources

1. To delete the resources, open Terminal again.
2. Run the below command to delete all the resources.

terraform destroy

1. Approve the creation of all the resources by entering **yes**. You can see the **Destroy complete!** message.



**Completion and Conclusion**

* You have successfully set up the Visual Studio Code editor.
* You have successfully created variables.tf and terraform.tfvars files.
* You have successfully created IAM Role using terraform
* You have successfully created Cloudwatch Logs using terraform
* You have successfully created VPC, Internet Gateway and Subnet using terraform
* You have successfully created VPC Flow Logs using terraform
* You have successfully Launched an EC2 Instance using the terraform.
* You have successfully created output.tf
* You have successfully executed the terraform configuration commands to create the resources.
* You have successfully checked all the resources created by opening the Console.
* You have successfully deleted all the resources.

**End Lab**