

Home / AWS / Guided Lab / Launch an EC2 Instance as a web server using Terraform

Launch an EC2 Instance as a web server using Terraform

Level: Fundamental

Amazon EC2 Amazon Web Services Terraform



0h 44m 34s left



End Lab

Open Console

Validation

Lab Credentials

User Name ⓘ

Whiz_User_80425.88365888



Password ⓘ

5b4d329c-2989-43b4-abbe-722c3bfc7b0



Access Key ⓘ

AKIA4GZ7DAACMZPMUY6Q



Secret Key ⓘ

5m96mY3yYokDFPO6hfRIFckd6p5oC1kN4mPVNFEp






Lab Resources

No Lab Resources Found

Support Documents

1. [FAQs and Troubleshooting](#)

Need help?

-  How to use Hands on Lab
-  Troubleshooting Lab
-  FAQs

[Submit Feedback](#)[Share](#)[Lab Overview](#)[Lab Steps](#)[Lab Validation](#) Cloud Architect Compute, Infrastructure

Lab Steps

Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** 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**.

Note: If you face any issues, please go through [FAQs and Troubleshooting for Labs](#).

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
```



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

```
mkdir task_10001_ec2
```



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

```
cd task_10001_ec2
```

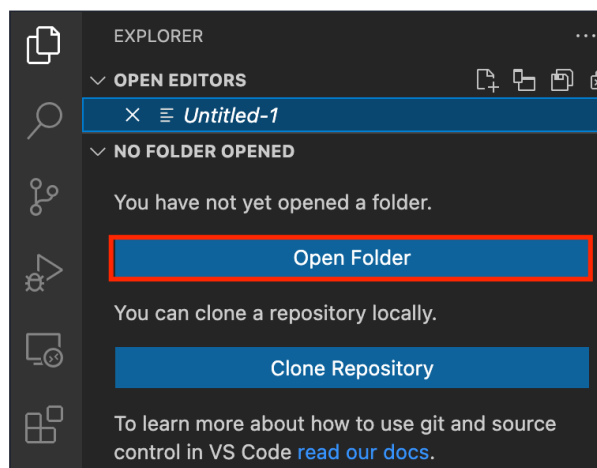


9. Get the location of the present working directory by running the below command:

```
pwd
```



10. Note down the location, as you will open the same in the next steps.
11. Now click on the first icon Explorer present on the left sidebar.
12. Click on the button called Open folder and navigate to the location of folder **task_10001_ec2**.



13. (Optional) Click on Authorize button for allowing Visual Studio Code to use the task_10001_ec2 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.
14. Visual Studio Code is now ready to use.

Task 3: Create a variables 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_10001_ec2** 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_10001_ec2** 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 = "Region of AWS VPC"  
}
```



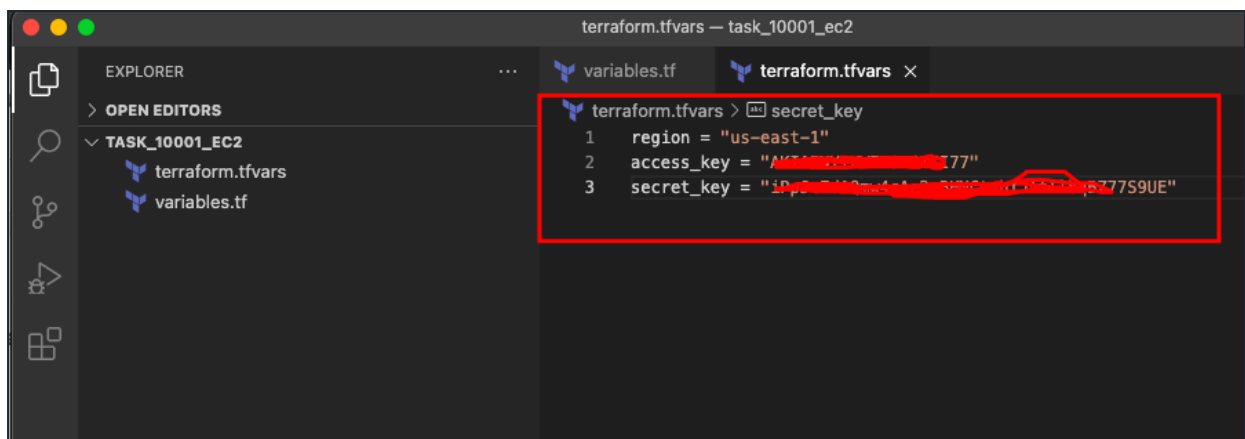
5. In the above content, you are declaring a variable called, access_key, secret_key, and region with a short description of all 3.
6. After pasting the above contents, save the file by pressing **ctrl + S**.
7. Now expand the folder **task_10001_ec2** and click on the **New File** icon to add the file.
8. Name the file as **terraform.tfvars** and press **Enter** to save it.
9. 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>"
```



10. In the above code, you are defining the dynamic values of variables declared earlier.
11. Replace the values of access_key and secret_key by copying from the lab page.
12. After replacing the values of access_key and secret_key, save the file by pressing **Ctrl + S**.





Task 4: Create EC2 and its components 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_10001_ec2** 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}"
}
```



4. In the above code, you are defining the provider as aws.
5. Next, we want to tell Terraform to create a Security Group within AWS EC2, and populate it with rules to allow traffic on specific ports. In our case, we are allowing the tcp port 80 (HTTP).
6. We also want to make sure the instance can connect outbound on any port, so we're including an egress section below as well.
7. Paste the below content into the **main.tf** file after the provider.

```
resource "aws_security_group" "web-server" {
  name = "web-server"
  description = "Allow incoming HTTP Connections"
  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"]
}
}

```

8. Finally, to complete the main.tf file, let's add another set of code after security group creation where you will create an EC2 instance.

```

resource "aws_instance" "web-server" {
  ami = "ami-02e136e904f3da870"
  instance_type = "t2.micro"
  key_name = "whizlabs-key"
  security_groups = ["${aws_security_group.web-server.name}"]
  user_data = <<-EOF
#!/bin/bash
sudo su
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo "<html><h1> Welcome to Whizlabs. Happy Learning... </h1></html>" >>
/var/www/html/index.html
EOF
tags = {
  Name = "web_instance"
}
}

```



9. In the above code, we have defined the Amazon Linux 2 AMI. The AMI ID mentioned above is for the us-east-1 region.

10. We have mentioned the resource which SSH key to use (which is already present in your AWS EC2 console). The security group ID is automatically taken by using the variable which will be set during the creation process.

11. We have added the user data to install the apache server.

12. We have provided tags for the EC2 instance.

13. Save the file by pressing Ctrl + S.

Task 5: 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_10001_ec2** 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 "web_instance_ip" {  
  value = aws_instance.web-server.public_ip  
}
```



4. In the above code, we will extract the Public IP of the created EC2 instance and display it once the instance is created.

Task 6: 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_10001_ec2
```



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

```
terraform version
```



4. 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 7: Apply terraform configurations

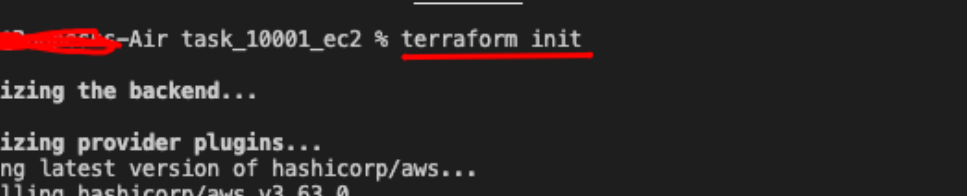
1. Initialize Terraform by running the below command,

```
terraform init
```



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





The screenshot shows a terminal window with a dark background. At the top, there are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active. The terminal prompt is 'root@ip-10-0-10-10:~#'. The user has entered the command 'terraform init'. The output shows the initialization process, including finding the latest version of hashicorp/aws, installing it, and creating a lock file. The final message is 'Terraform has been successfully initialized!' in green text.

```
root@ip-10-0-10-10:~# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.63.0...
- Installed hashicorp/aws v3.63.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
```

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

```
terraform plan
```



4. Review the whole generated plan.

```

s-MacBook-Air task_10001_ec2 % terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  + create

Terraform will perform the following actions:

# aws_instance.web-server will be created
+ resource "aws_instance" "web-server" {
  + ami              = "ami-02e136e904f3da870"
  + arn              = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)

```

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

```
terraform apply
```



6. You will be able to see the resources which will be created, approve the creation of all the resources by entering **yes**.

7. It may take up to 2 minutes for the terraform apply command to create the resources.

8. Id's of all the resources created by terraform will be visible there.

9. The output i.e Public IP of the EC2 instance is extracted and displayed. Copy the Public IP.


```
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_security_group.web-server: Creating...
aws_security_group.web-server: Still creating... [10s elapsed]
aws_security_group.web-server: Creation complete after 13s [id=sg-0f3b1fd0693b392ae]
aws_instance.web-server: Creating...
aws_instance.web-server: Still creating... [10s elapsed]
aws_instance.web-server: Still creating... [21s elapsed]
aws_instance.web-server: Still creating... [31s elapsed]
aws_instance.web-server: Still creating... [41s elapsed]
aws_instance.web-server: Creation complete after 43s [id=i-079861878683e7128]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

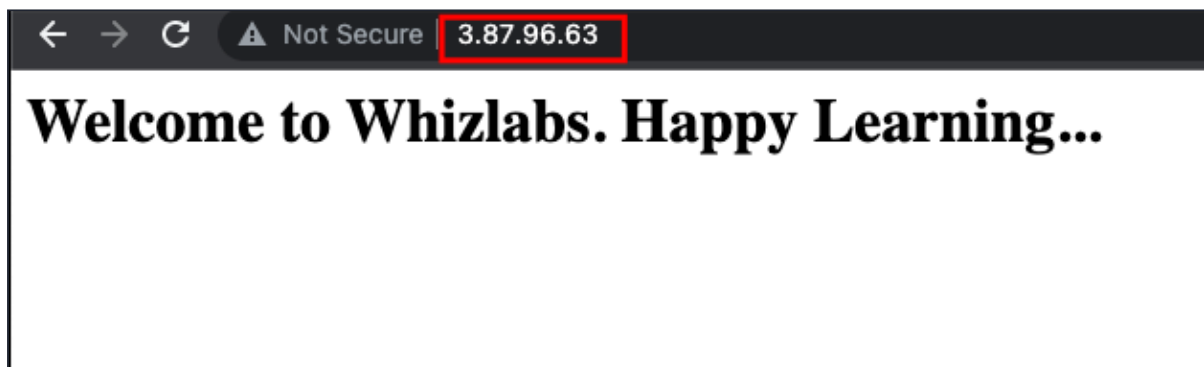
Outputs:

web_instance_ip = "3.87.96.63"
```

10. Optionally, you can note down the IDs of all the resources.

Task 8: Check the HTML page

1. In the terraform file, we have used user data to create an apache server and publish a HTML page.
2. Open a new tab in the browser and paste the Public IP of the created EC2 instance.
3. The HTML content created in the user data is displayed in the page.

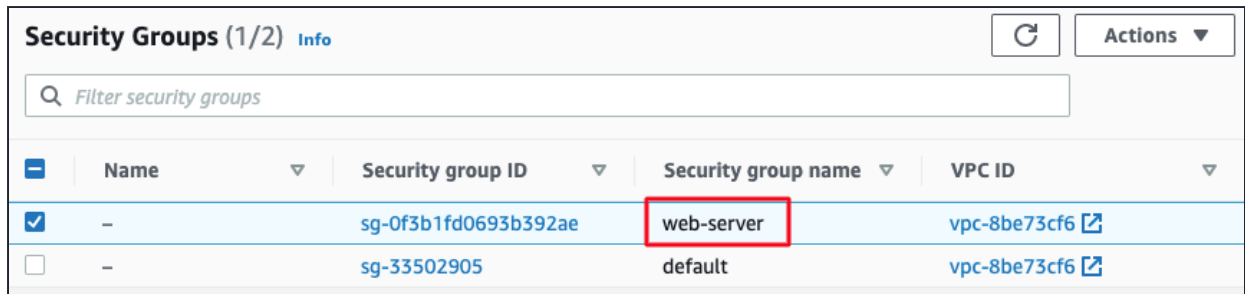


4. We can now say that the EC2 instance has been created with the apache server and the HTML content is published properly.
5. We can also confirm that the security group is allowing HTTP incoming requests.

Task 9: Check the resources in AWS Console

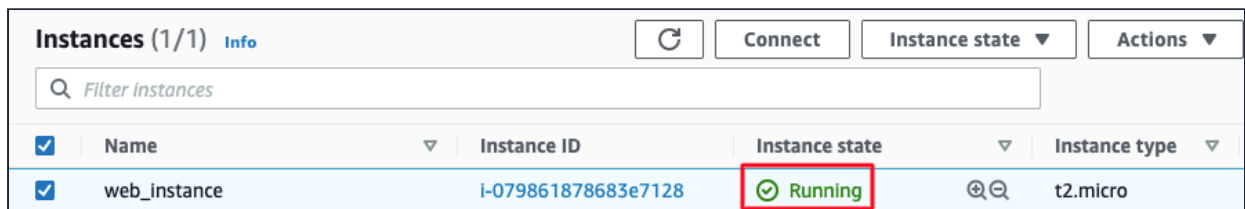
1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **EC2** by clicking on **Services** on the top, then click on **EC2** in the **Compute** section.
3. Navigate to **Security Groups** under **Network & Security** on the left panel.

- You will be able to see the security group with the name **web-server** which we have created in the terraform.



	Name	Security group ID	Security group name	VPC ID
<input checked="" type="checkbox"/>	-	sg-0f3b1fd0693b392ae	web-server	vpc-8be73cf6
<input type="checkbox"/>	-	sg-33502905	default	vpc-8be73cf6

- Navigate to **Instances** under **Instances** on the left panel.
- You can see the instance created. You can check the configurations that we applied in the terraform file like key pair, security group, instance type, etc.



	Name	Instance ID	Instance state	Instance type
<input checked="" type="checkbox"/>	web_instance	i-079861878683e7128	Running	t2.micro

Do You Know ?

Terraform is an open-source infrastructure as code (IaC) tool that allows you to define and provision infrastructure resources in a declarative manner. It provides a simple and consistent way to manage infrastructure across different cloud providers, including AWS.

Task 10: Validation of the lab



- Once the lab steps are completed, please click on the **Validation** button on the right side panel.
- This will validate the resources in the AWS account and displays whether you have completed this lab successfully or not.
- Sample output :


Lab Overview

Lab Steps

Lab Validation

Check your Validation

If any checks fail , you can use the remaining time it the Lab to work on making the checks pass . Click Validate My Lab again to rerun the checks at any time.

Validate My Lab 

Launch an Amazon EC2 Instance

Check whether a t2.micro Amazon Linux 2 EC2 Instance is created or not.

Install Apache Web Server

Check whether Apache Web Server is installed and index.html file is added in EC2 Instance or not.

Task 11: Delete AWS Resources

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

```
terraform destroy
```



3. Enter **yes** to confirm the deletion.





```
Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.web-server: Destroying... [id=i-079861878683e7128]
aws_instance.web-server: Still destroying... [id=i-079861878683e7128, 10s elapsed]
aws_instance.web-server: Still destroying... [id=i-079861878683e7128, 20s elapsed]
aws_instance.web-server: Still destroying... [id=i-079861878683e7128, 30s elapsed]
aws_instance.web-server: Destruction complete after 39s
aws_security_group.web-server: Destroying... [id=sg-0f3b1fd0693b392ae]
aws_security_group.web-server: Destruction complete after 3s

Destroy complete! Resources: 2 destroyed.
```

4. You can verify the deletion of resources in the AWS Console.

Instances (1) Info							Connect	Instance state 
 Filter instances								
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status			
<input type="checkbox"/>	web_instance	I-0b822a38ba3e99440	 Terminated	t2.micro	-			

Completion and Conclusion

- You have set up the Visual Studio Code editor.
- You have created variables.tf and terraform.tfvars files.
- You have created a main.tf file.
- You have executed the terraform configuration commands to create the resources.
- You have checked all the resources created by opening the Console.
- You have deleted all the resources.

End Lab

1. Sign out of AWS Account.
2. You have successfully completed the lab.
3. Once you have completed the steps, click on **End Lab** from your whizlabs dashboard.

[About Us](#) [Subscription](#) [Instructions and Guidelines](#) [FAQ's](#) [Contact Us](#)



© 2023, Whizlabs Software Pvt. Ltd.

