**Lab Steps**

Task 1: Sign in to AWS Management Console

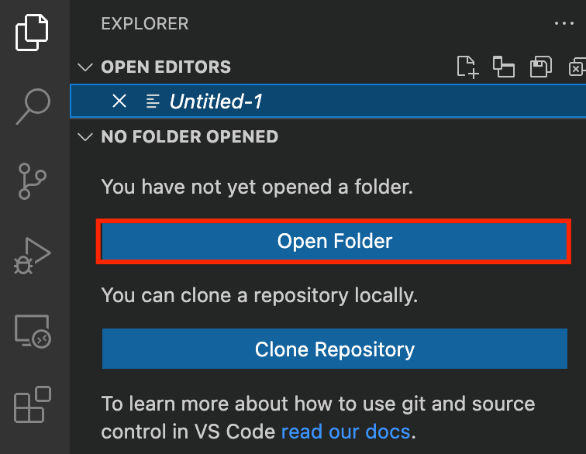
1. Click on the button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

* 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

1. 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.



1. 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\_10120\_cf

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

cd task\_10120\_cf

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\_10120\_cf**
4. (Optional) Click on Authorize button for allowing Visual Studio Code to use the **task\_10120\_cf** 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.
5. 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.

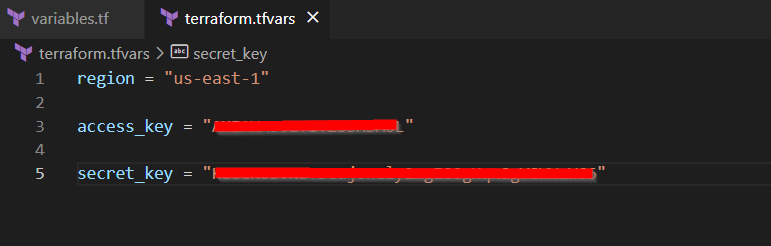
1. To create a variable file, expand the folder **task\_10120\_cf** and click on the **New File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note**: Don't change the location of the new file, keep it default, i.e. inside the **task\_10120\_cf** folder.
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\_10120\_cf** 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\_ACCESS\_KEY>"  secret\_key = "<YOUR\_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 a key pair for EC2 Instance in the main.tf file

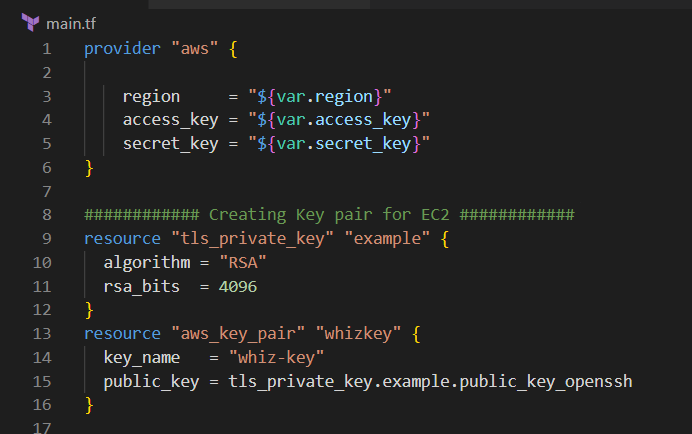
 In this task, you will create a **main.tf** file where you will add details of the provider and resources.

1. To create a **main.tf** file, expand the folder **task\_10120\_cf** 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 a key pair for the EC2 Instance.
3. Paste the following code into the **main.tf** file.

|  |
| --- |
| ############ Creating Key pair for EC2 ############  resource "tls\_private\_key" "example" {    algorithm = "RSA"    rsa\_bits  = 4096  }  resource "aws\_key\_pair" "whizkey" {    key\_name   = "whiz-key"    public\_key = tls\_private\_key.example.public\_key\_openssh  } |



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

Task 5: Create a CloudFormation Stack in main.tf file

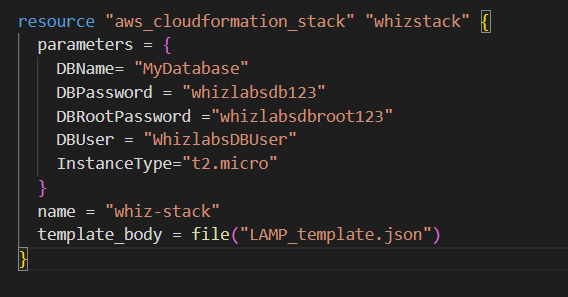
In this task we are going to create a cloudformation stack.

1. Firstly you will download[**LAMP\_template.json**](https://labresources.whizlabs.com/353db87bf9bfe4fc5ee7c92c5fa30a87/LAMP_template.json) and upload it locally to your folder **task\_10120\_cf**. This template contains the creation of LAMP server.
2. Name the file as **LAMP\_template.json**.



1. To create a Cloudformation stack, paste the following contents in the **main.tf** file:

|  |
| --- |
| resource "aws\_cloudformation\_stack" "whizstack" {    parameters = {      DBName= "MyDatabase"      DBPassword = "whizlabsdb123"      DBRootPassword ="whizlabsdbroot123"      DBUser = "WhizlabsDBUser"      InstanceType="t2.micro"    }    name = "whiz-stack"    template\_body = file("LAMP\_template.json")  } |



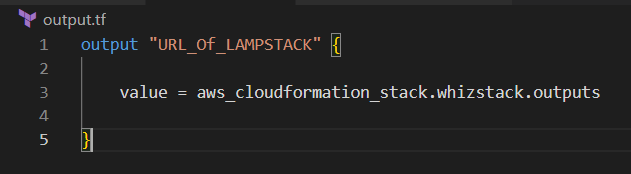
1. In the above code, the parameters for the webserver and database are defined that will be required for the creation of LAMP server. Also the name and template body is defined as well.
2. Save the file by pressing **Ctrl + S**.

Task 6: 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\_10120\_cf** 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. In the above code, we will extract the LAMP server URL.

|  |
| --- |
| output "URL\_Of\_LAMPSTACK" {      value = aws\_cloudformation\_stack.whizstack.outputs  } |



Task 7: 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\_10120\_cf

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

terraform version

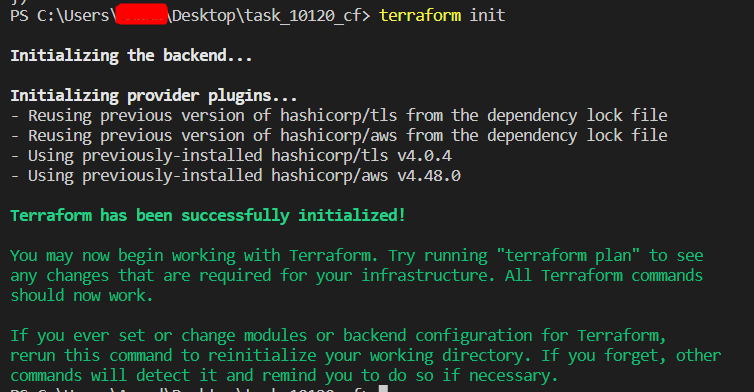
1. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

Task 8: Apply terraform configurations

1. Initialize Terraform by running the below command:

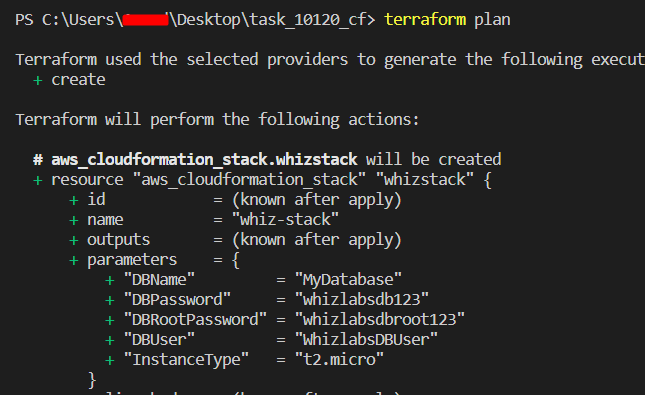
terraform init

**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,

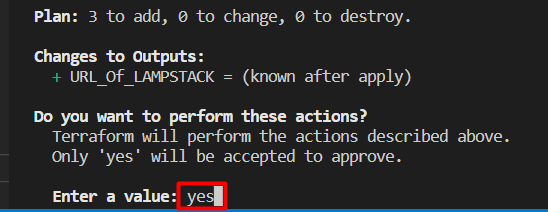
terraform plan



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

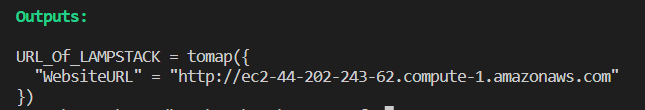
terraform apply

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

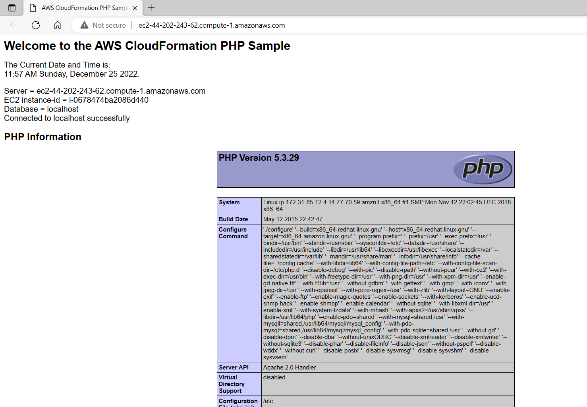


* **Note: This process will take around 5-10 minutes.**

1. **URL** of the Lamp stack  created by terraform will be visible there.

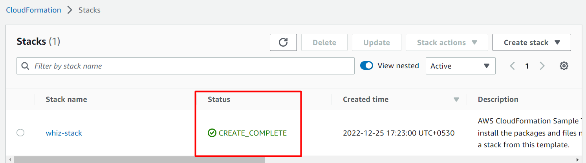


1. **Copy** the URL and **paste** it in the browser to see if the lamp server is deployed successfully.If you see the PHP info and your database connection, it means you have completed a LAMP server setup with AWS CloudFormation. Sample screenshot provided below:



Task 9: Check the resources in AWS Console

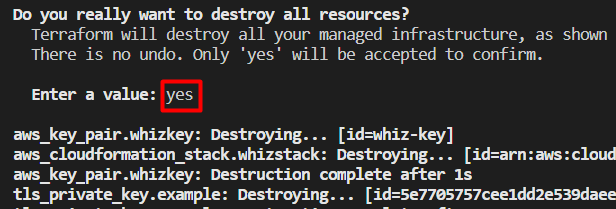
1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **CloudFormation** by clicking on **Services** on the top, then click on **CloudFormation** in the **Management and Governance** section.
3. Click on **Stacks** in left navigation menu.  You can see that the stack is created successfully.



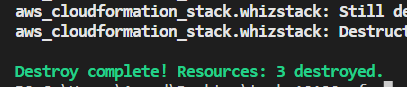
Task 10: Delete AWS Resources

1. To delete the resources, open Terminal again.
2. Run the below command to delete all the resources. Enter **yes** to confirm the deletion.

 terraform destroy



1. You can see the **Destroy complete!** message.



**Completion and Conclusion**

* You have succesfully set up the Visual Studio Code editor.
* You have succesfully created variables.tf and terraform.tfvars files.
* You have successfully created a key pair for the EC2 instance in the main.tf file
* You have successfully created a CloudFormation stack in the main.tf file
* You have successfully created an output file.
* You have successfully checked all the resources created by opening the Console.
* You have deleted all the resources.

**End Lab**