Develop and Deploy Php Application in Linux Virtual Machine (LAB-M01-01)

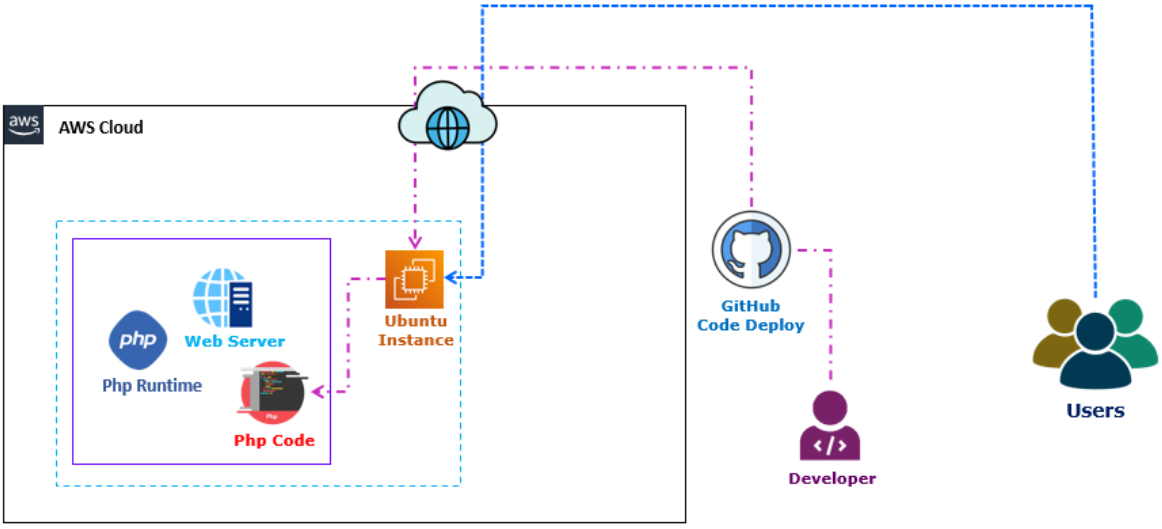
Lab scenario

You're preparing to deploy web application in AWS. As a development group, your team has decided to use Php application to deploy in the Linux environment in AWS.

Objectives

After you complete this lab, you will be able to:

- Create new virtual machine.
- Build the Run-time environment.
- Deploy the Php code.
- Access your web application.



Task 1: Develop Php Application

In this task, you will develop the Php code for deployment.

Step 1: Develop the Code to Display the Server IP Address

1. **Unzip** the **LAB-01-01.zip** (**Php code**).

Note: **Lab-01-01.zip** code file is available with the Lab Manual.

2. **Open** the **index.php** in the **notepad**.
3. **Update** the **code** to **display** the *server Private address*.

   a. Look for the **TO DO** section.

```
</head>
<body class="landing">

        <!-- Banner -->
                <section id="banner">
                        <h2>AWS Developer LAB</h2>
                        <! TO DO >

                        <ul class="actions">
                                <li><a href="#" class="button special big">Get Started</a></li>
                        </ul>
                </section>
```

   b. **Add** the **Code** after **TO DO** to display the server Private IP address.

Info:

1. You can also use the below code to display the Server Private IP Address.

<font color="white">Server IP Address <?= $_SERVER['SERVER_ADDR'] ?>

2. Add the above code below to **<! TODO >** in the index.php.

```
anner -->
 <section id="banner">
        <h2>AWS Developer LAB</h2>
        <! TO DO >
        <p class="ui header center"><font color="white">Server IP Address <?= $_SERVER['SERVER_ADDR'] ?></p>

        <ul class="actions">
                <li><a href="#" class="button special big">Get Started</a></li>
        </ul>
 </section>
```

   c. Select the **File**.

d. Select **Save**.

Step 2: Create GitHub Account

Info

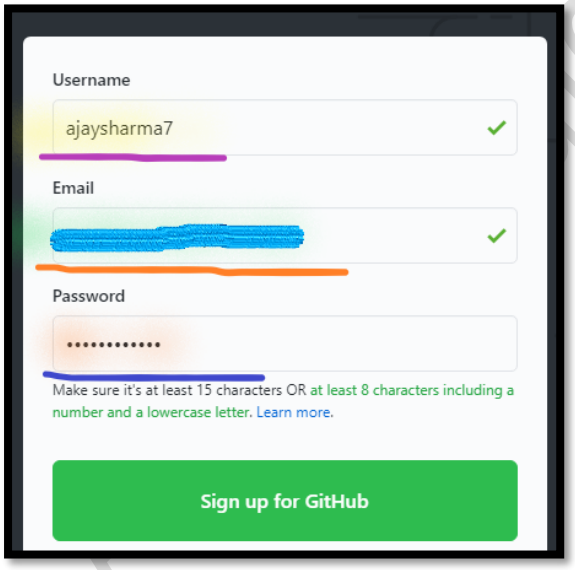
- If you already have Github account, use your account and follow the **Step 3**.
- If you don't have Github account, follow the below steps.

4. **Open** the below URL in browser

   https://github.com/

5. In the **Signup for GitHub** section, provide:

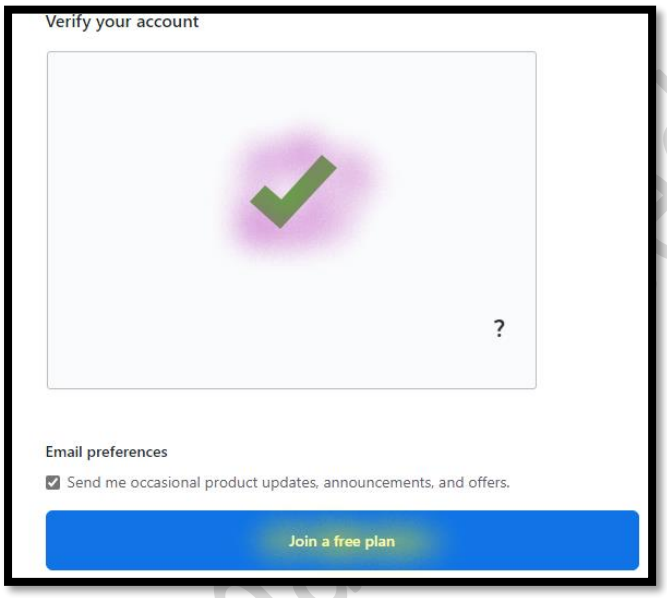
   i. **Username**: Provide unique **user name**.
   ii. **Email ID**: Provide your **email Id**.
   iii. **Password**: Provide **password**.
   iv. Select **Signup for GitHub**.



6. In the ***Verify your account section***, Select the **Verify** and complete the process for verification.

Note: If succesfully completed the process, you get Right Checkmark.

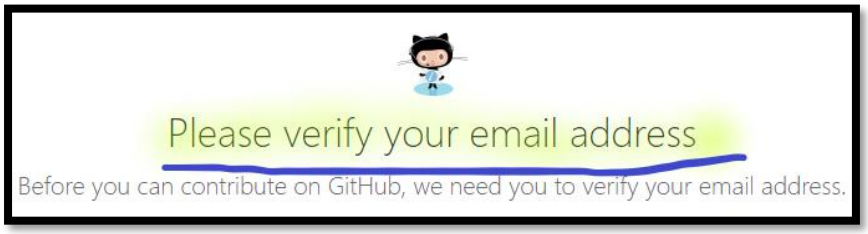
i. Select **Join a Free plan**



ii. Select **Complete the Setup**

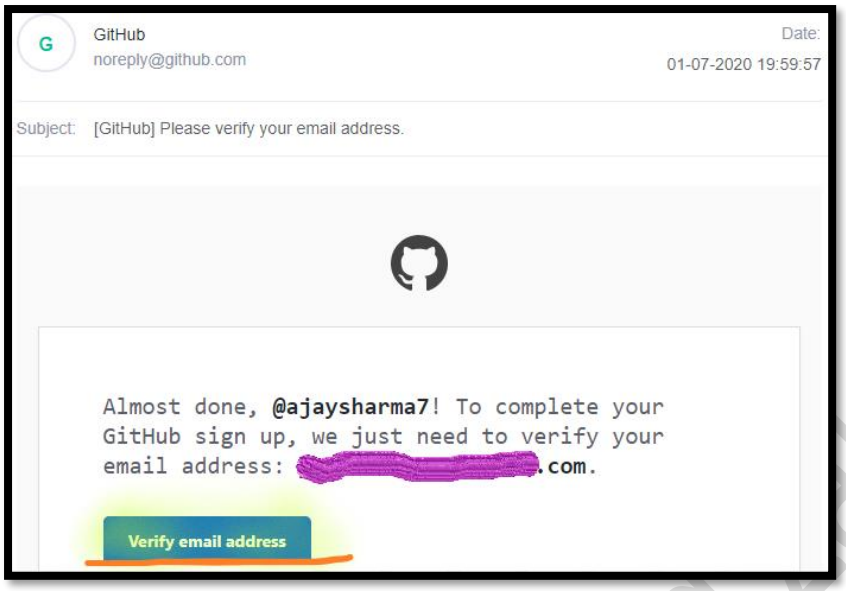
Note: You can provide other details also.

iii. You get the message to **Please verify your email address**.

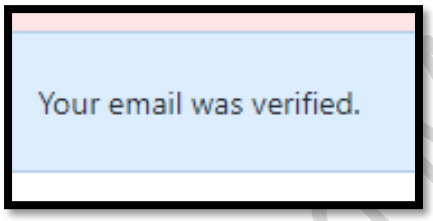


7. ***Returned to your email box*** to complete the verification. Check your email Id, you get email from GitHub to verify your email Id.

   i. Select **verify email address**.



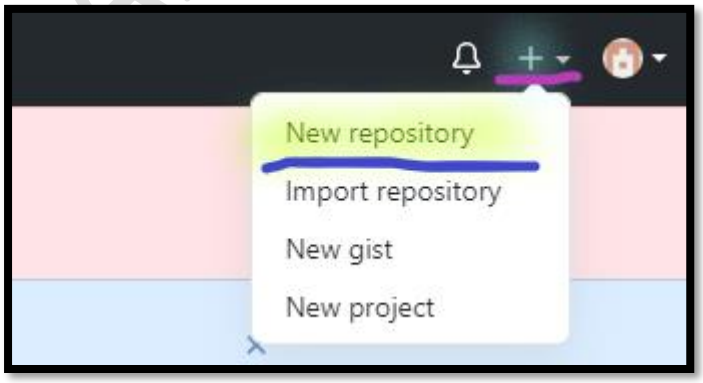
   ii. Once verified you get **message** email verification message on GitHub.



Step 3: Create GitHub Repository

8. **Login** into your GitHub account.

9. **To create repository**, go to right top side and Select **+** sign.



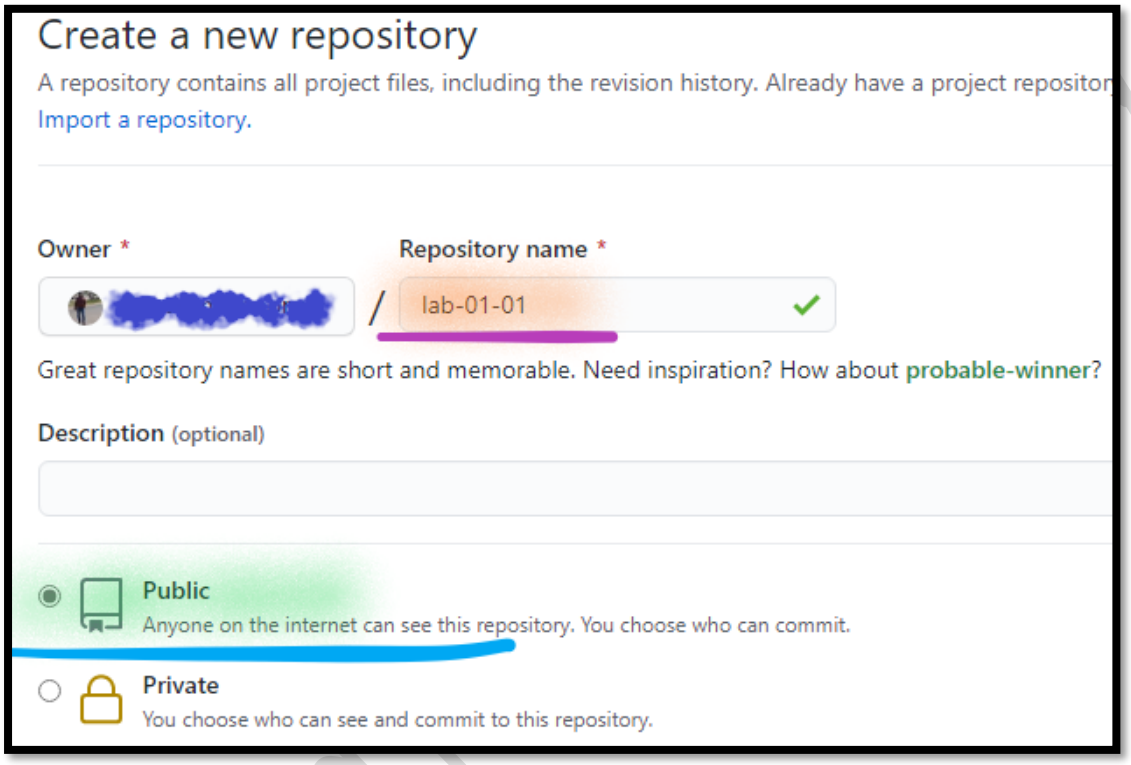
i. Select **New repository** and **configure**:

a. **Repository name**: Write **lab-01-01**

b. Select **Public**

c. Select **initialise this repository with a README**

d. Select **Create repository**



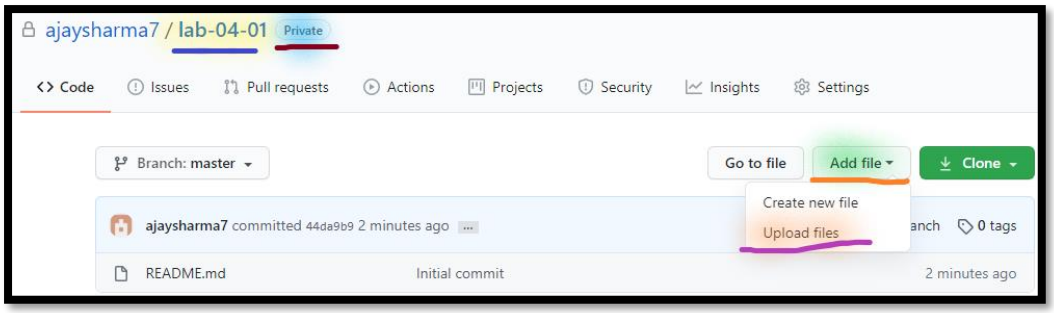
Note: Once repository created, **lab-01-01 repsoitory** page gets opened.

Step 2: Upload the code to GitHub Repository

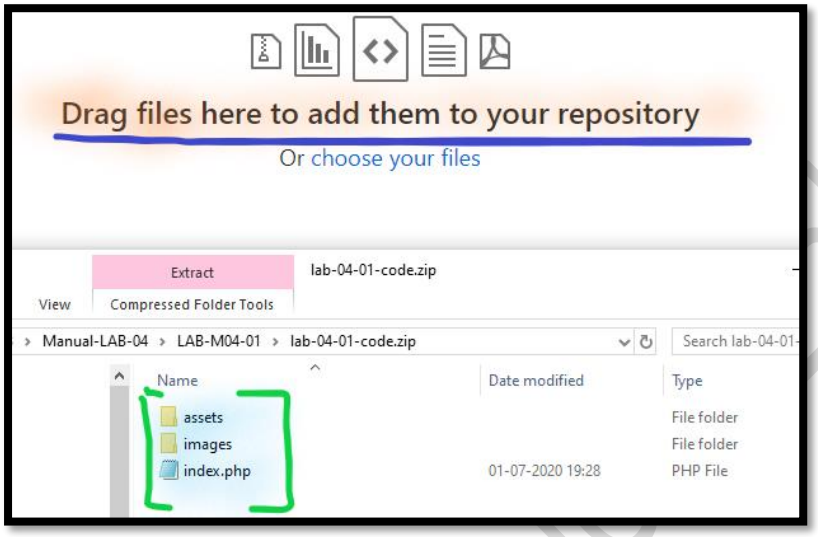
10. From the **lab-01-01 repository:**

i. Select **Add file**.

ii. Select **Upload files**.



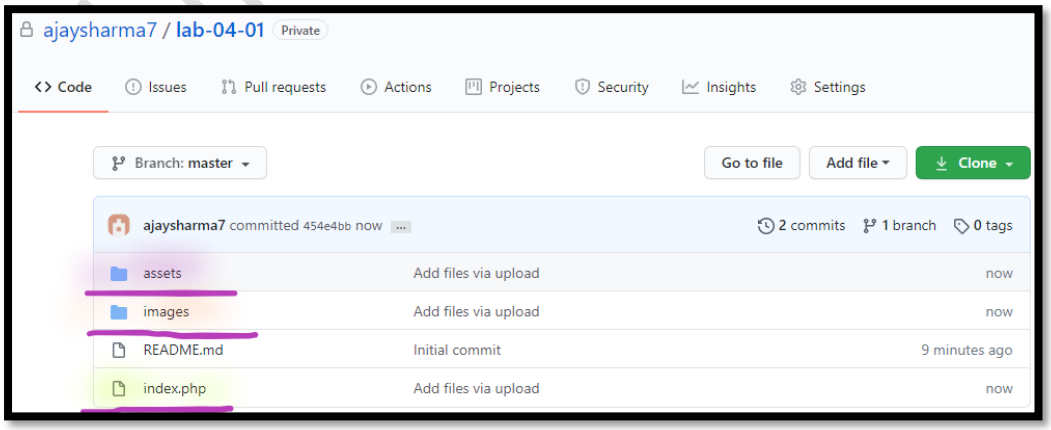
iii. **Drag and Drop** the **lab-01-01-code**.



Note: **Copy** the **Code structure** (file and folders) **not Zip file**.

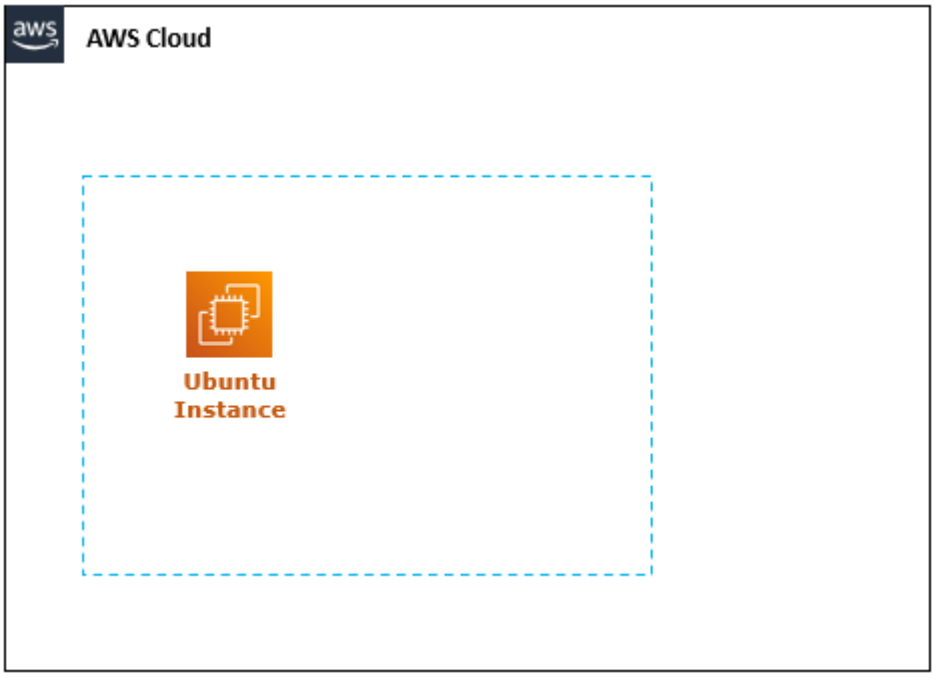
iv. Select **Commit Changes**.

Note: Once code **uploaded successfully**, you can see them in repository.



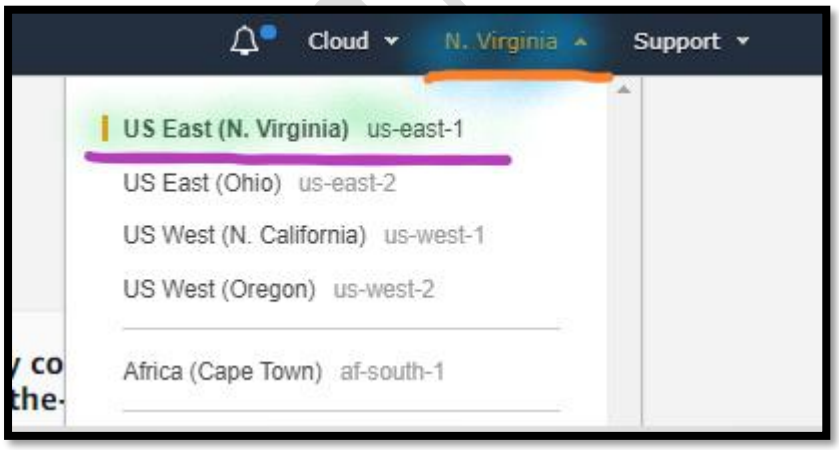
Task 2: Create Server to Deploy Php Application

In this task, you will launch an Amazon EC2 instance using the management console. The instance will be used to deploy the PHP Application.

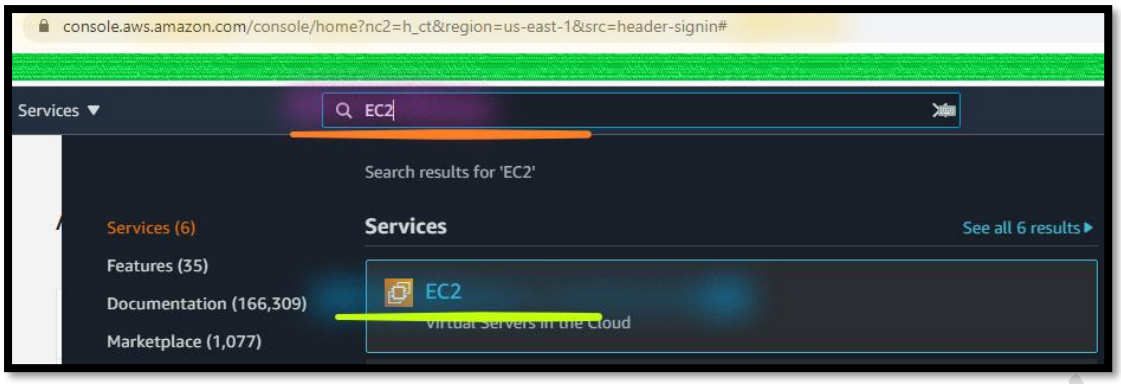


Step 1: Create EC2 Instance

11. Choose the **US East (N. Virginia)** region list to the right of your account information on the navigation bar.



12. In the **AWS Management Console**, on the **Services** menu **Search** and **Select EC2**.



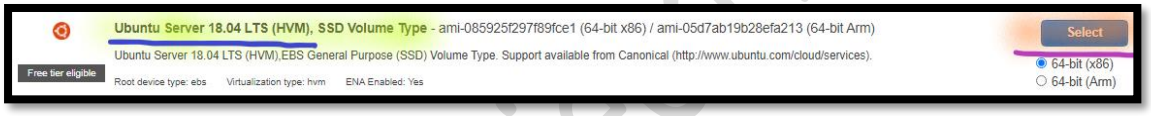
13. Click **Launch Instance**.

    i. Click **Launch Instance**.

14. **Choose Amazon Machine Image (AMI)** section:

    i. **Go below** and search for **Ubuntu Server 18.04 LTS**.

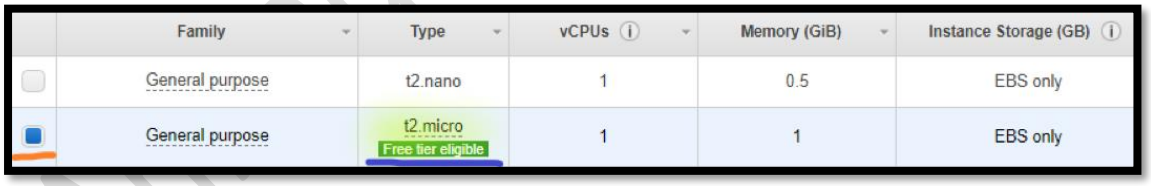
    ii. Click on **Select**.



15. **Choose Instance Type** section:

    i. Choose an **t2.micro**.

    ii. Click **Next: Configure Instance Details**.



Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)
General purpose	t2.nano	1	0.5	EBS only
General purpose	t2.micro Free tier eligible	1	1	EBS only

16. **Configure Instance Details** section:

    i. Select **Next: Add Storage**.

Note: Leave the detail as default.

17. **Add Storage** section:

    i. Select **Next: Add Tags**.

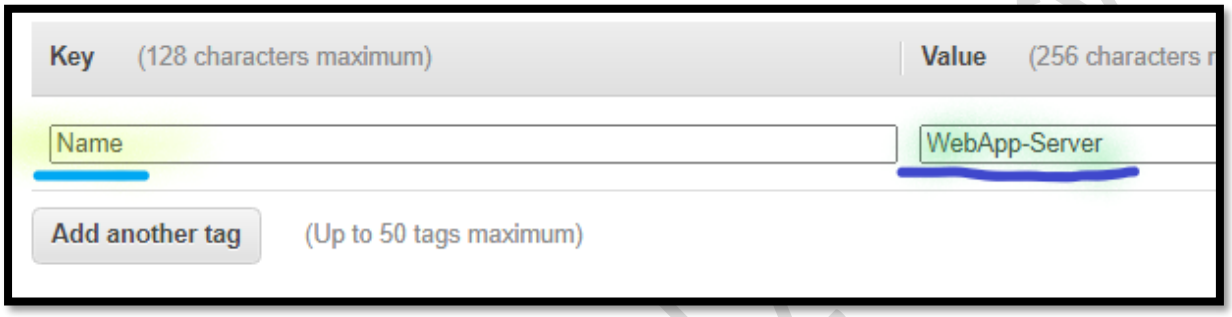
Note: Leave the detail as default.

18. **Add Tags** section:

    i. Select **Add Tag**.

        a. **Key Name**: Write **Name**.

        b. **Value**: Write **PHP App Server**.



    ii. Choose **Next: Configure security group**.

19. **Configure Security Group** section:

Note: The wizard automatically defines the launch-wizard-*x* security group and creates an inbound rule to allow you to connect to your instance over SSH (**port 22**).

    i. Assign a security group: Select **Create a new security group**.

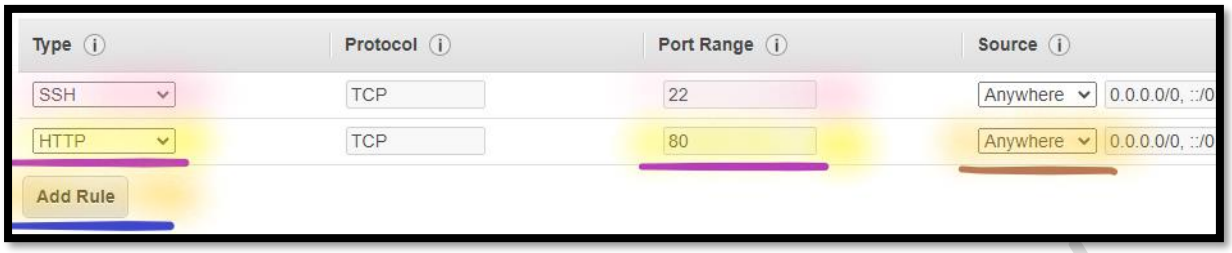
        a. **Security group name**: Write **PHP App Server**.

        b. **Description**: Write **Ubuntu App Server Security Group**.

        c. Select **Add Rule**.

Note: **Don't remove** or update the SSH rule. **Add the new rule**.

- **Type**: Dropdown and Select **HTTP**.
- **Source**: Dropdown and Select **Anywhere**.



ii. Click **Review and Launch**.

20. Click **Launch**

Note: New window will pop-up for Key pair.

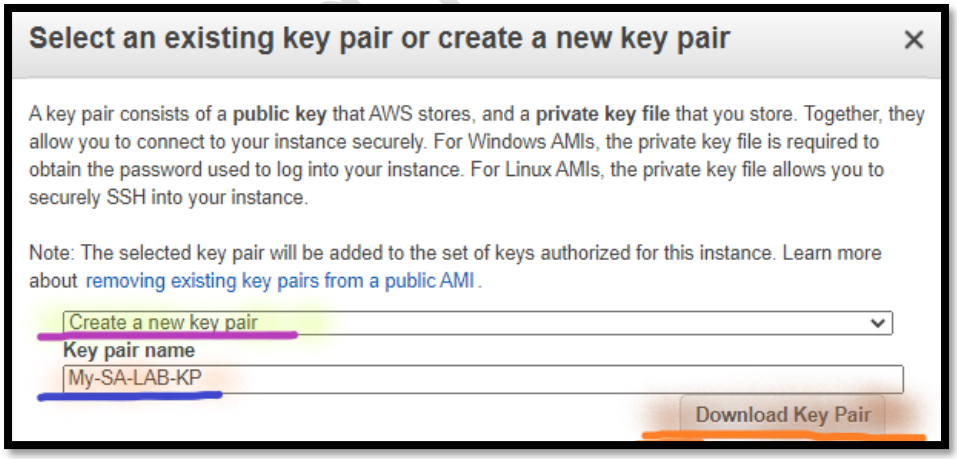
21. **Select an existing key pair or create a new key pair** section:

i. In the **popover**, dropdown and select **Create a new key pair**.

a. **Key pair name**: Write **My-Dev-LAB-KP**.

ii. Click **Download Key Pair**.

Note: **My-Dev-LAB-KP.pem** will be downloaded to your computer. Make sure to save this key pair in a safe location on your computer.



22. Click **Launch Instances**.

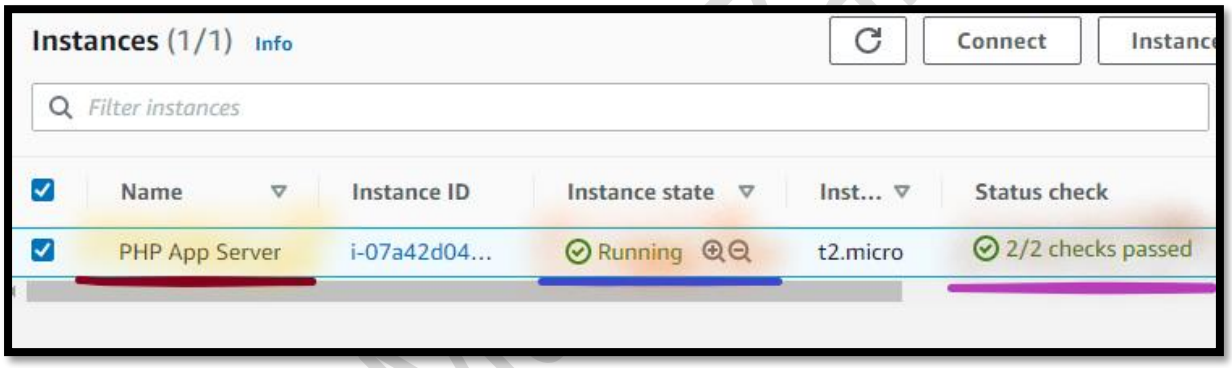
Note: Wait for few minutes to launch your instance.

23. Click on **View Instance**.

> **Note**: **Keep** the **My-Dev-LAB-KP** key pair **secure**. We will use the same key pair in all upcoming labs.

Step 2: Check the Php App Server Status

24. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
25. Click **Instance**.
26. Select **PHP App Server**.
    i. **Wait** for the **Instance State** to change to **Running** state.
    ii. **Wait** for the **Status check** to change to **2/2 checks passed**.



Task 3: Connect to Php App Server

In this task, you will log into the PHP Application (Linux) Server that you just created.

Step 1: Copy Php Application Server Public IP address

27. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
28. Click **Instances**.
29. Select **PHP App Server**.
    i. Go below and click on **Details**.
    ii. Expand **Instance summary**.
    iii. **Copy** the **Public IP address**.

Note

- The following instructions now vary slightly depending on whether you are using Windows or Mac/Linux.

Step 2: Install PuTTy

Note: If you are using **Mac** Operating System, go below to follow the **Step 5**.

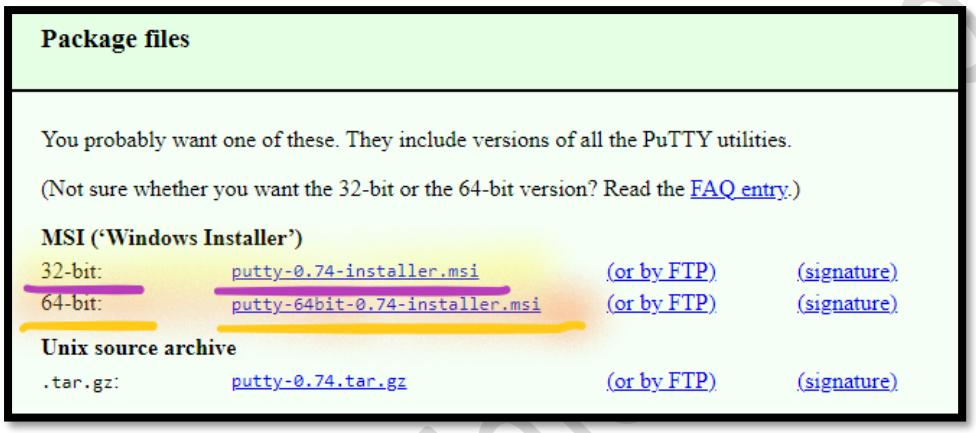
PuTTY does not natively support the private key format (.pem) generated by Amazon EC2. PuTTY has a tool named PuTTYgen, which can convert keys to the required PuTTY format (.ppk). You must convert your private key into this format (.ppk) before attempting to connect to your instance using PuTTY.

30. **Install** **PuTTY** on **Windows** Operating System.

    i. Browse **www.putty.org**.

    ii. Click on **You can download PuTTY here** under **Download PuTTY**.

iii. **Download** the **MSI Installer** (32-bit or 64-bit as per your local desktop/ laptop OS version) under **Package files**.



31. **Install** the downloaded **MSI Installer**.

Note: You can also copy the **PuTTY**.exe and **PuTTygen**.exe, which is available with the Lab manual.

Step 3: Convert PEM file into PPK file

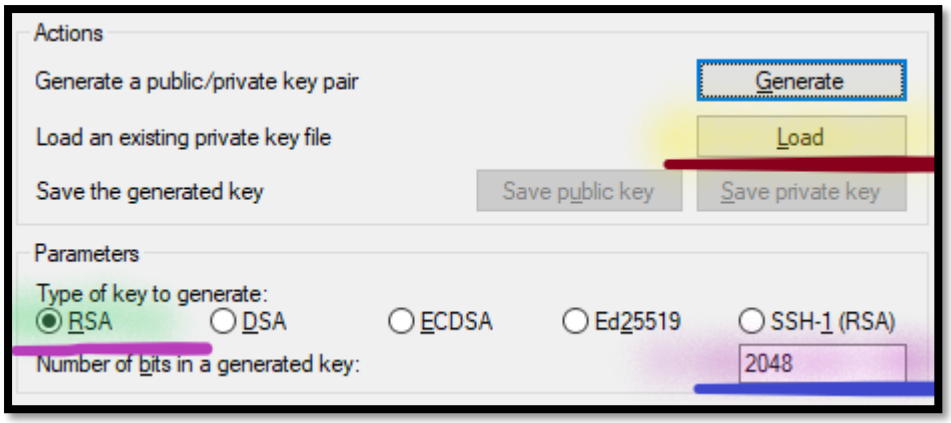
32. From your ***local desktop/ laptop*** (Windows), right click on **Start** & **Run**.

33. In the **Open**, type **puttygen.exe**.

    i. **Parameters**: Select **RSA**.

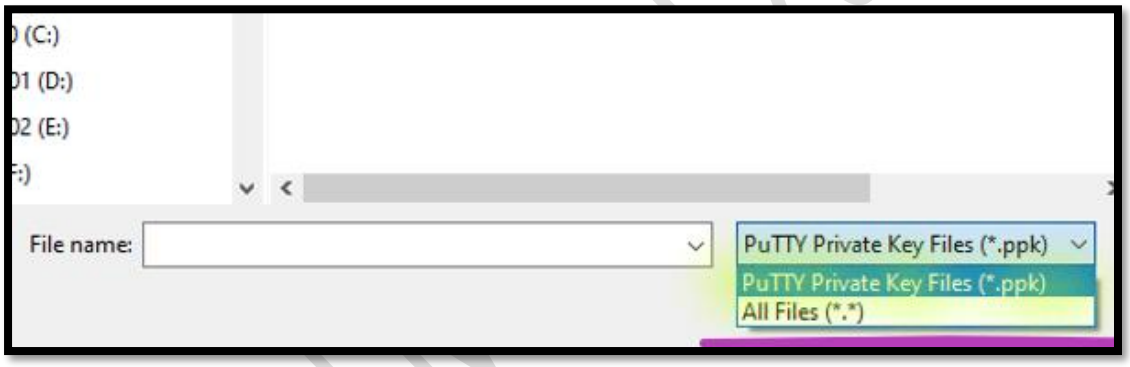
        a. **No. of bits in generated key** should be **2048**.

    ii. Choose **Load**.



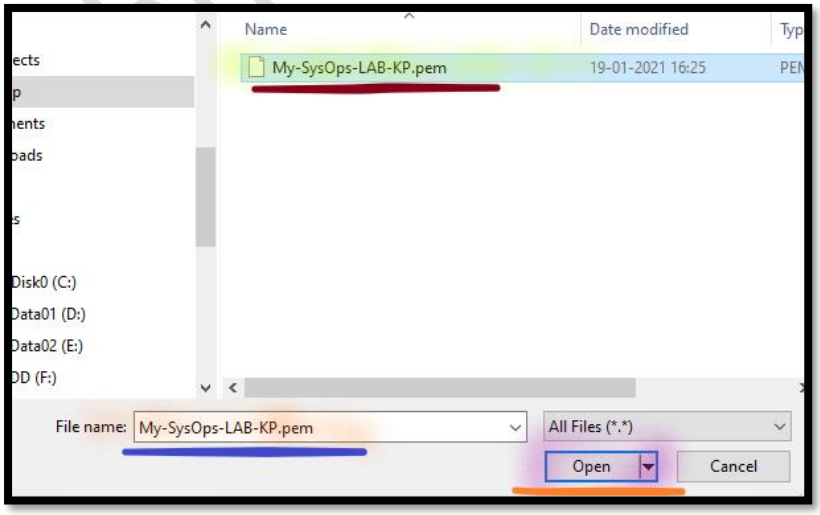
Note: By default, PuTTYgen displays only files with the **extension .ppk**. To locate your pem file.

iii. **File extension**: Dropdown and Select **All files**, to locate your ***pem extension*** files.

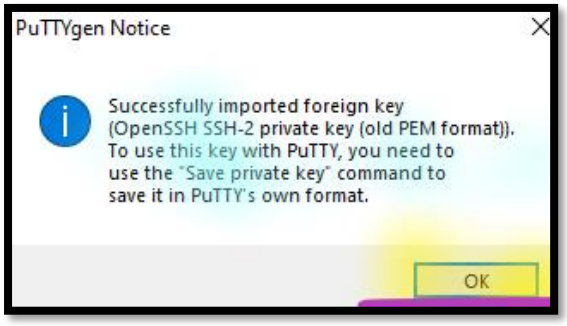


iv. **Navigate** to the folder where you downloaded your key pair and Select **My-Dev-LAB-KP**.pem file.

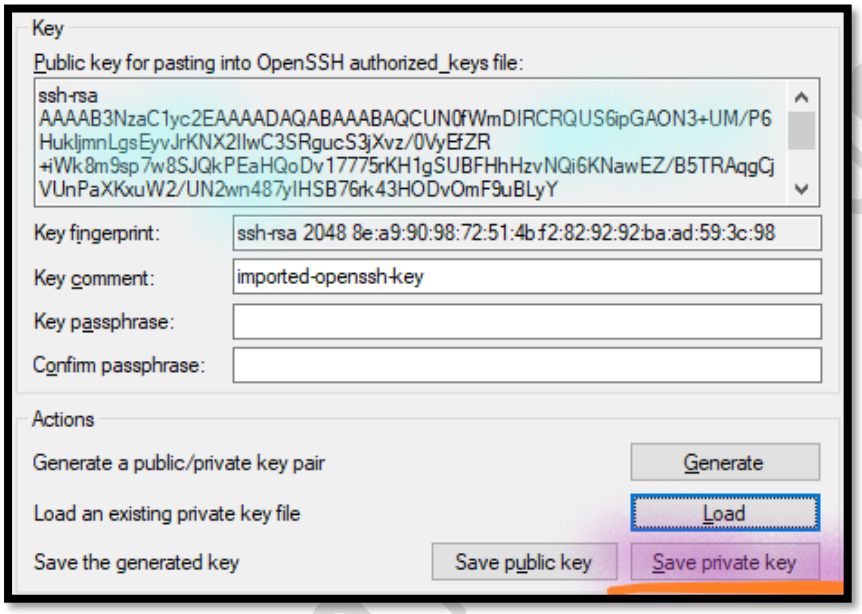
a. Select **Open**.



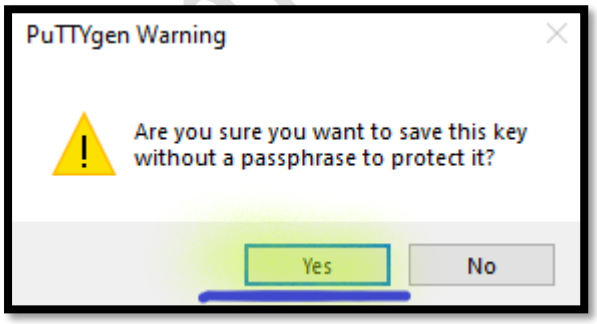
b. Choose **OK** on the confirmation dialog box.



v. Choose **Save private key** to save the key in the format that PuTTY can use.

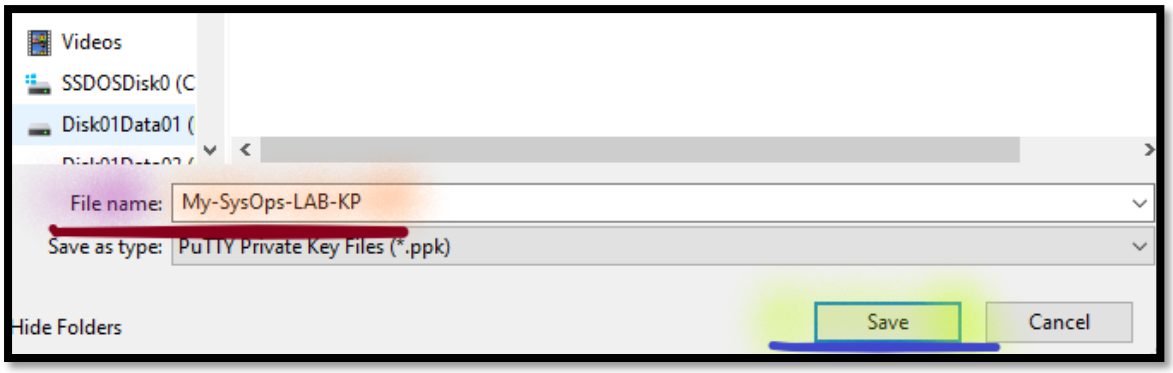


vi. Choose **Yes**, when PuTTYgen displays a warning about saving the key without a passphrase.



a. **File name**: Write **My-Dev-LAB-KP**.

b. Select **Save**, PuTTY automatically adds the **ppk** file extension.



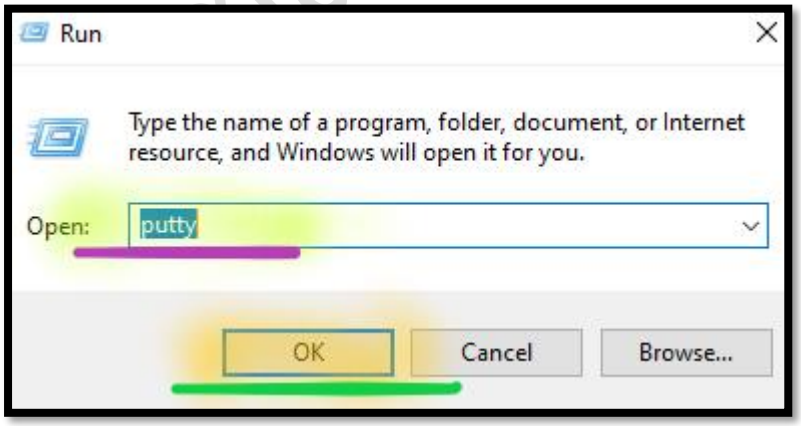
Note: Your private key is now in the correct format for use with PuTTY. You can now connect to your instance using PuTTY's SSH client.

Step 4: Connect to Linux Server from Windows Operating System

34. From the **local Desktop/ Laptop** (Windows), right click on **Start** & **Run**.

i. In the **Open**, write **putty**.

ii. Select **Ok**, it will open the PuTTY window.



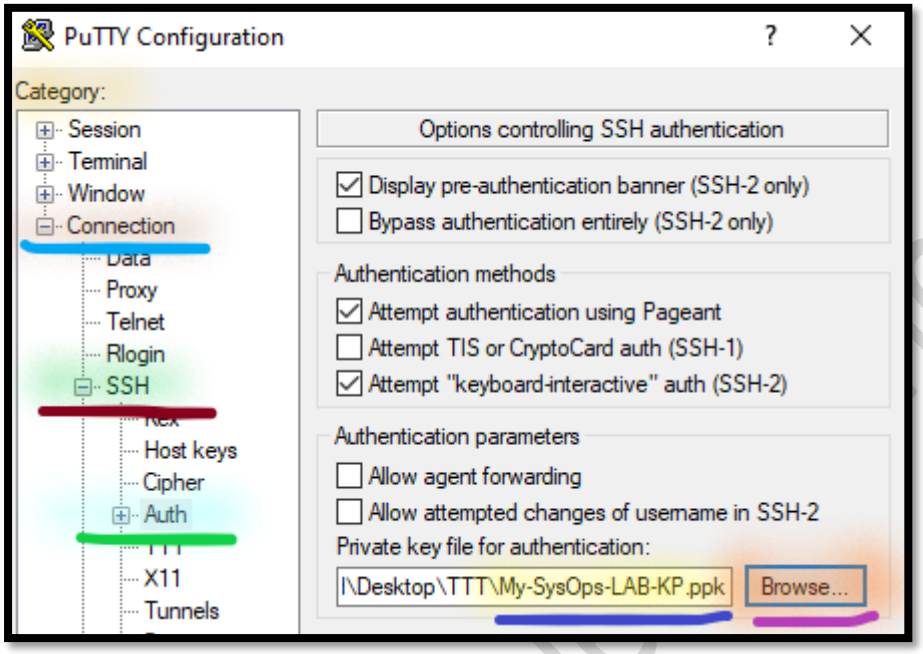
iii. Expand **Connection**, in the **Category** pane:

a. Expand **Connection**.

b. Expand **SSH** and Select **Auth**.

c. Select **Browse**.

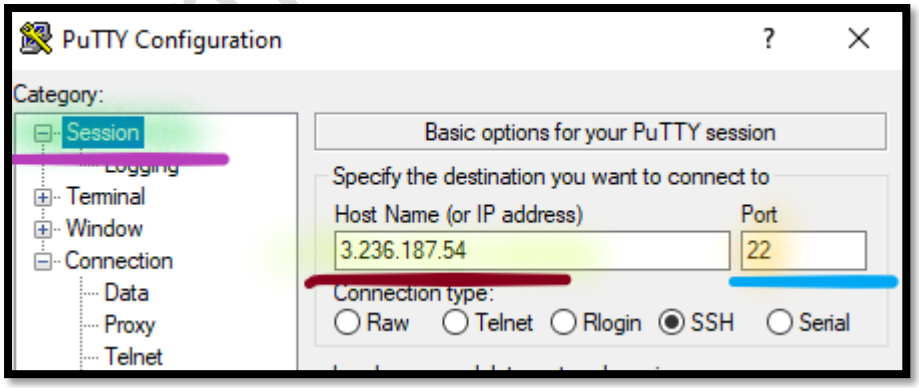
d. **Navigate** and Select the **My-Dev-LAB-KP**.ppk file that you generated in previous step.



iv. Select **Session**, in the **Category** pane.

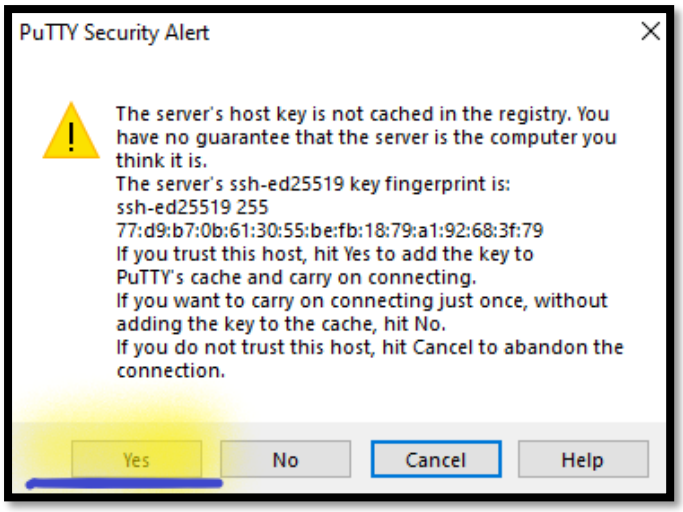
a. **Host name**: Write **Public IP Address** of PHP Application Server (Linux virtual machine), which you have copied from the previous step.

b. **Port**: Write **22**.



c. Select **Open** to start the PuTTY session.

d. Select **Yes**, once PuTTY displays a security alert dialog box.



Note: You can see the **Linux terminal** of your instance.

- **Login as**: Write username **ubuntu**.

Note: You are using a key pair for authentication, you will not be prompted for a password.

Note: **Don't close the Linux terminal**.

Note: **Skip** the Step 5 & Go to the **Task 4**.

Step 5: Connect to Linux Server from Mac/ Linux Operating System

Note: If you are using **Windows** Operating System, go above and follow from the **Step 2**.

You will need to know the location of your key pair you created when you launched your instance. Usually this will be in your "Downloads" folder, but you may want to move it elsewhere.

35. **Launch** the Mac/ Linux **terminal**.

i. **Change directory** to ***My-Dev-LAB-KP****.pem* keypair location. **Copy** the below command to a Mac/ Linux terminal.

```
cd ~/Downloads
```

Note: You need to use absolute path name where key pair stored.

ii. **Change** the **access permissions *of My-Dev-LAB-KP***.pem keypair. **Copy** the below command to a Mac/ Linux terminal:

```
chmod 400 My-Dev-LAB-KP.pem
```

iii. **Copy** this command to a **text editor**:

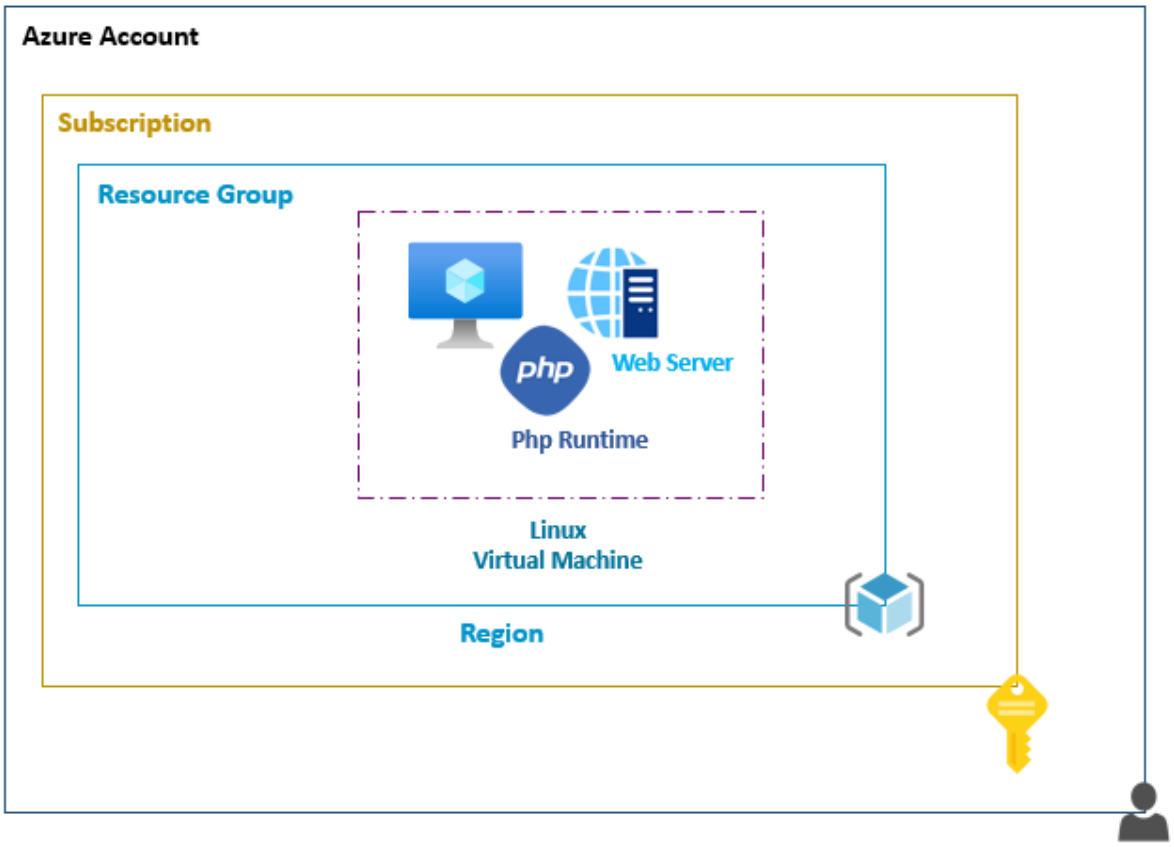
```
ssh -i My-Dev-LAB-KP.pem ubuntu@PUBLIC-IP-ADDRESS
```

Note: Replace the **Public IP Address** with the **Public IP address** of the **Php Application Server** (Linux virtual machine), which you have copied in the previous step.

iv. Type **Yes** when prompted to allow a connection to the remote SSH server.

Task 4: Deploy the Php App Code

In this task, you will install the Web Server and Php Run-time environment to deploy the php code.



Step 1: Install Php Runtime Environment to Deploy the PHP Code

36. From **Php App Server** (Linux virtual machine) instance ***terminal***:

   i. **Update** the **Packages**:
   sudo apt-get -y update

   ii. **Install** the **Apache**:
   sudo apt-get install -y apache2

   iii. **Install** the **Php**:
   sudo apt-get install -y php

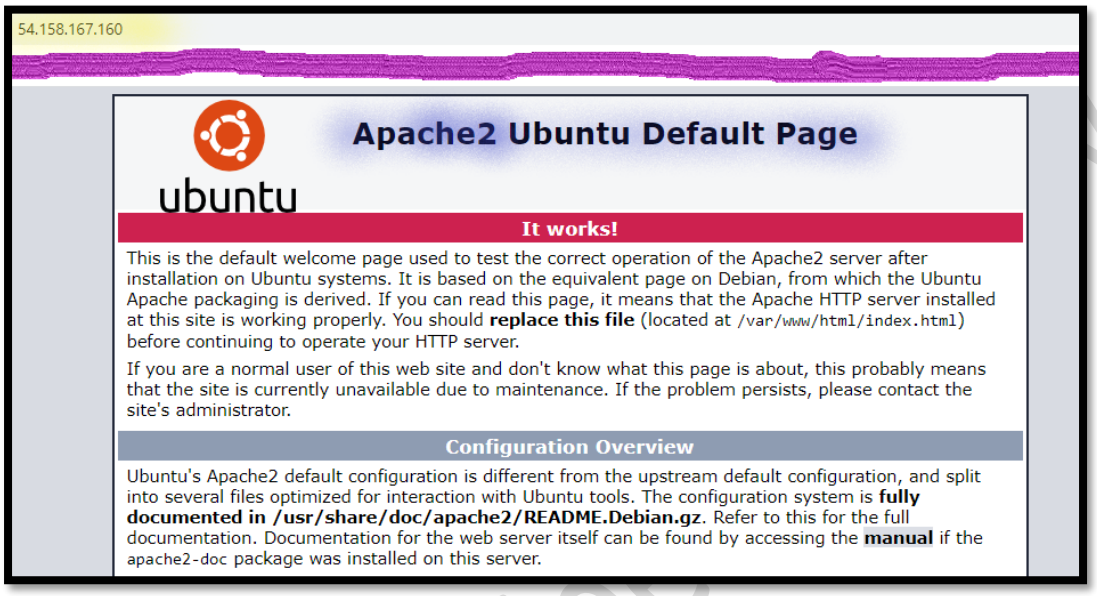
   iv. **Install** the **Git**:
   sudo apt-get install -y git

Note: Go to the next task **(but don't close the Linux terminal)**.

Step 2: Access the Php App Server

37. From the **Web browser**, type **Public IP Address** of **Php Application Server** (Linux virtual machine) and access your **Php website.**

Note: You will see the Default Apache Ubuntu page.



Note: Go to the next step, **But Don't close the Apache web page**.

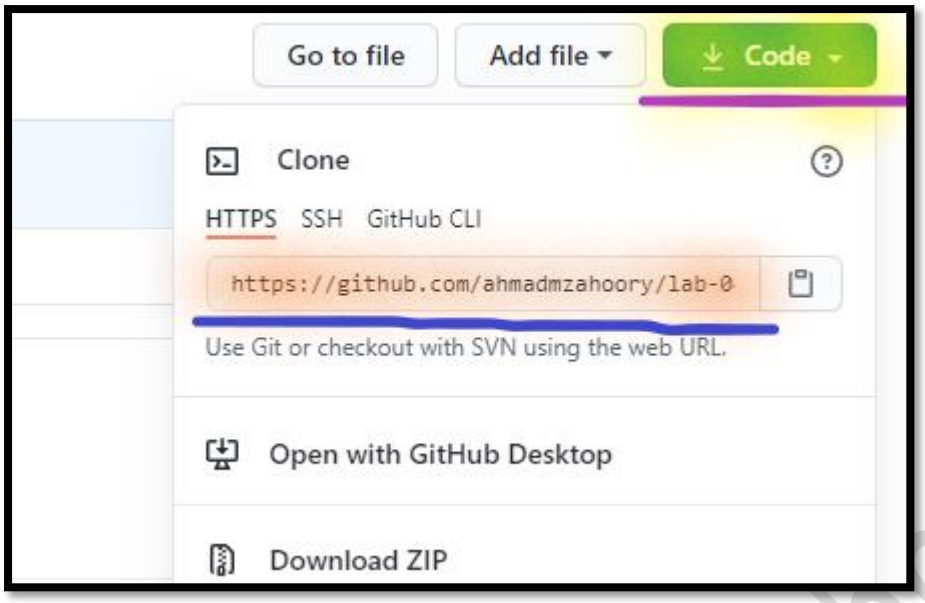
Step 3: Clone the GitHub Repository

38. **Login** into your GitHub account.

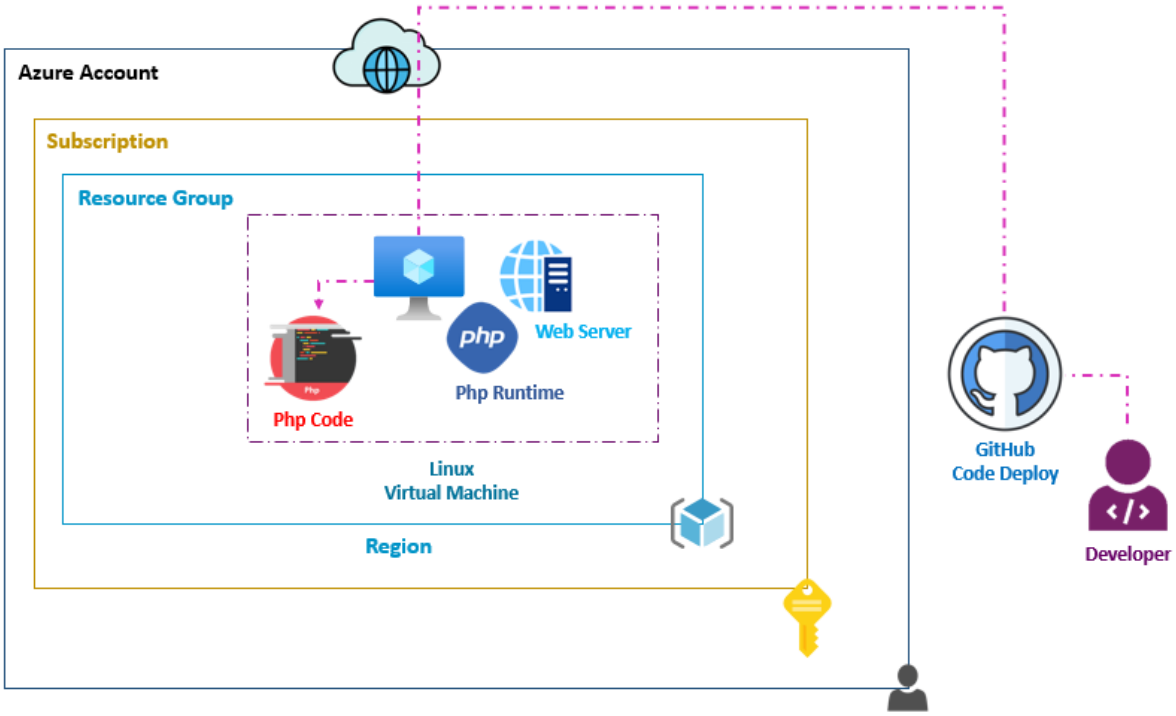
39. Open the **lab-01-01** GitHub **repository**.

    i. Click on **Code**.

    ii. **Copy** the **Clone web URL**.



Step 4: Deploy Php Code



40. Returned to **Php Application Server** (Linux virtual machine) instance terminal.
    i. **Change directory** to **/var/www/html**:
    cd /var/www/html/

    ii. **Remove** the default **index.html**:
    sudo rm index.html

iii. **Clone** the **lab-01-01** GitHub **repository**:
sudo git clone **CLONE-WEB-URL**

Note: **Replace** the **CLONE-WEB-URL** with the Github URL you have copied in the previous step.

iv. **List** the file & folders:
ls -l

Note: You can see the **lab-01-01 folder**.

v. **Change** to lab-01-01 folder:
cd lab-01-01

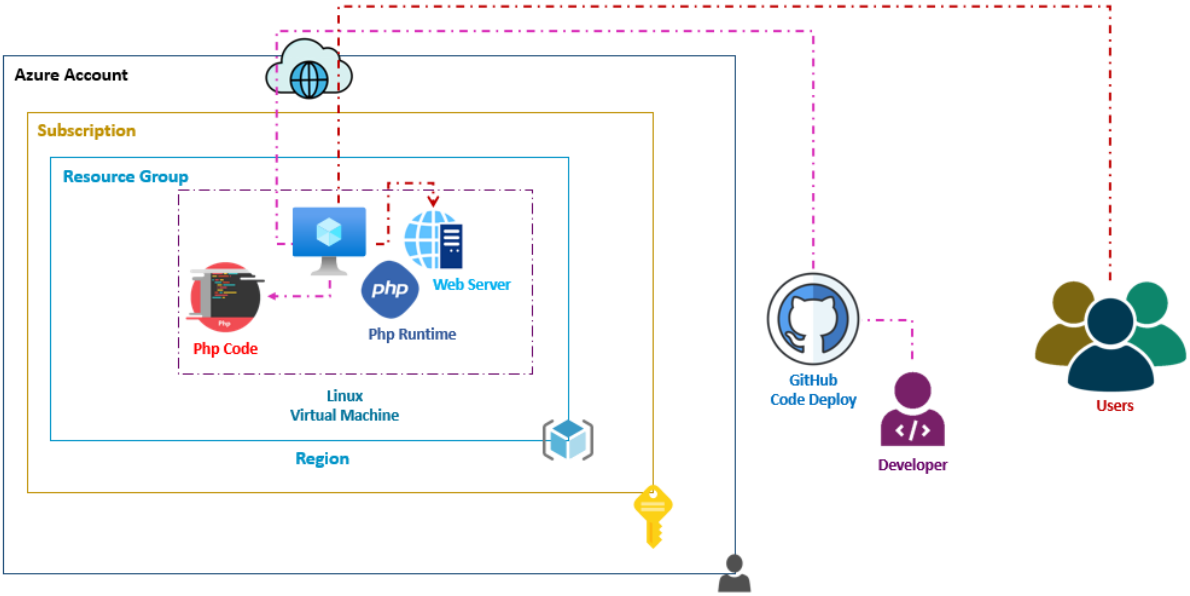
vi. **Move** all the contents to another folder:
sudo mv -v /var/www/html/lab-01-01/* /var/www/html/

vii. **Change** to parent directory:
cd ..

viii. **List** the file & folders:
ls -l

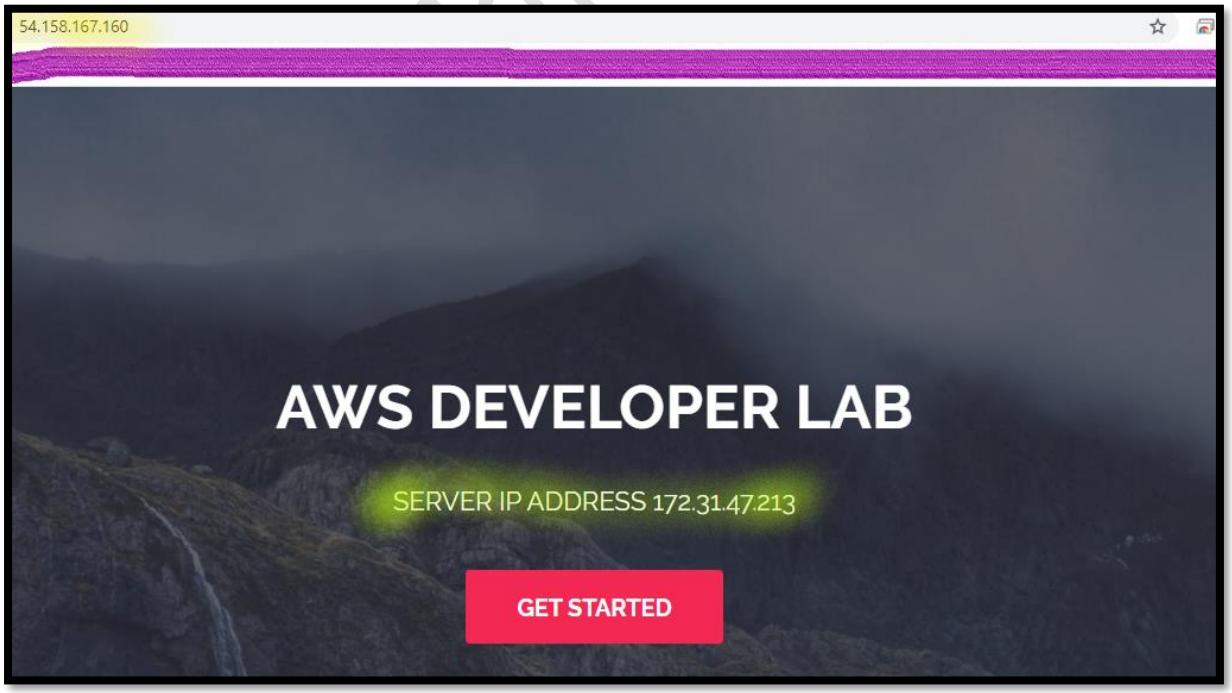
Note: You can see the **Php web application** code.

Step 5: Access the Php App Server



41. **Returned** to the **Web browser** and **Refresh** the web page.

Note: You will see the Php Application web page.

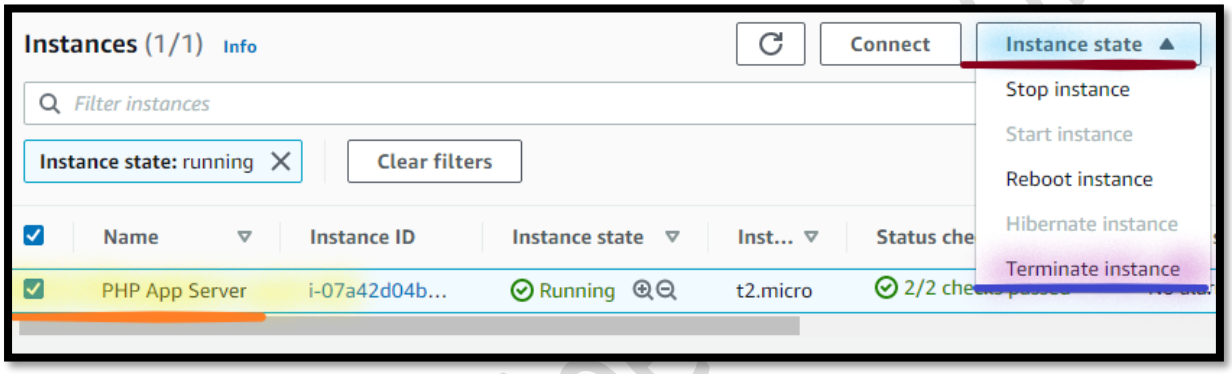


Note: Php Application web page also display the **Php Application Server** (Linux virtual machine) Private IP address.

Task 5: Clean up the Environment

Step 1: Terminate EC2 Instances

42. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
43. Click **Instances**.
44. Select **PHP App Server**.
    i. Click on **Instance state**.
    ii. Select **Terminate instance**.



iii. Select **Terminate**.