

# Deploying a Website on LAMP Server (Amazon EC2)

Submitted By

Rizve Ahmed

## Introduction:

This documentation provides a detailed guide to deploying a simple PHP file on a LAMP (Linux, Apache, MySQL, PHP) server hosted on

an Amazon EC2 instance. The LAMP stack is commonly used for hosting dynamic web applications.

## Prerequisites:

- An Amazon Web Services (AWS) account.
- Basic understanding of AWS services and concepts.
- Familiarity with Linux command line usage.
- A running Amazon EC2 instance.

## EC2 Instance Create Steps:

### 1. Accessing the EC2 Instance:

- Log in to your AWS Management Console.
- Navigate to the EC2 Dashboard.
- Click on "Launch Instances" to create a new EC2 instance.

### 2. Choosing an Amazon Machine Image (AMI):

Select an AMI based on your requirements. For a LAMP stack, you can search for an "Amazon Linux 2023" or "Ubuntu Server" AMI.

### 3. Choosing an Instance Type:

Choose an instance type based on needs. For this purpose, a t2.micro instance is suitable.

#### **4. Configure Instance:**

Configure instance settings as needed, such as network, subnet, auto-assign public IP, and storage.

#### **5. Add Storage:**

Specify the size of the root volume based on your requirements.

#### **6. Add Tags:**

Add tags to instance for easy identification.

#### **7. Configure Security Group:**

Configure security group settings to allow SSH access (port 22) and HTTP access (port 80).

#### **8. Review Instance Launch:**

Review instance configuration and click "Launch."

#### **9. Select Key Pair:**

Choose an existing key pair or create a new one. This key pair will be used to connect to EC2 instance via SSH.

#### **10. Launch Status:**

Once instance is launched, note its public IP address.

#### **11. Accessing the EC2 Instance:**

Open a terminal on local machine.

Use the following command to connect to your EC2 instance using SSH:

```
ssh -i your-key.pem ec2-user@your-instance-ip
```

#### **12. Setting Up the LAMP Stack:**

Follow AWS documentation to set up a LAMP stack on your EC2 instance: AWS LAMP Server Setup.

---

After successfully setting up the lamp server we need to test our lamp server

## Test my LAMP server:

To check LAMP server


1.Create a PHP file in the Apache document root:

```
[ec2-user ~]$ echo "<?php phpinfo(); ?>/var/www/html/phpinfo.php"
```

In a web browser, type the URL of the file that you just created. This URL is the public DNS address of your instance followed by a forward slash and the file name. For example:

<http://my.public.dns.amazonaws.com/phpinfo.php>

After add my public dns i can see my phpinfo.php file

PHP Version 8.1.7	
	
System	Linux ip-172-31-16-77.ec2.internal 5.15.57-28.127.amzn2022.aarch64 #1 SMP Thu Aug 4 17:06:57 UTC 2022 aarch64
Build Date	Jun 7 2022 16:21:36
Build System	Linux
Build Provider	Amazon Linux
Compiler	gcc (GCC) 11.3.1 20220421 (Red Hat 11.3.1-2)
Architecture	aarch64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/10-opcache.ini, /etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gd.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-mbstring.ini, /etc/php.d/20-mysqli.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-simplexml.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/20-xml.ini, /etc/php.d/20-xmlwriter.ini, /etc/php.d/20-xsl.ini, /etc/php.d/30-mysql.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini, /etc/php.d/30-xmlreader.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API20210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, compress.bzip2, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, bzip2.*, convert.iconv.*

## Creating a php file in lamp server:

Creating a PHP file within a LAMP server environment involves a series of steps that seamlessly combine to bring your code to life. Here's a step-by-step breakdown of the process, designed to illuminate every detail:

### **Navigating to the Web Directory:**

- Open a terminal on the server.
- Use the command `cd /var/www/html` to navigate to the web directory where your web content is stored. This directory serves as the root for your website.

### **Generating the PHP File:**

- Utilize the command `sudo nano filename.php` to create a new PHP file named `filename.php`. In our case, I want to create a file named `riju.php`.
- By including `sudo` before the command, ensure that i have the necessary permissions to create and edit files in the directory.

### **Editing the PHP File:**

- The terminal will now open a text editor called Nano, displaying a blank canvas for your PHP file.
- Begin by adding the PHP opening tag `<?php` to signify the start of the PHP code.
- With the Nano editor, i can now input your desired code within the PHP tags. This could range from simple text outputs to complex calculations, depending on your project's requirements.

### **Saving and Exiting Nano:**

- Once i have added code, press `Ctrl + O` to save the file.
- It will be prompted to confirm the filename. Press `Enter` to confirm, and then press `Ctrl + X` to exit Nano.

### **Viewing the PHP Output:**

- With my PHP file created and code entered, its ready to witness the result. Open a web browser and enter the URL to my PHP file. Enter `http://My-server-ip/riju.php`.

### **Experiencing the Result:**

- When i visit the URL, your browser communicates with the LAMP server. The PHP file is executed on the server, and the output is sent back to the browser.
- In this case, the browser will display some information about Inflexionpoint Technologies BD LTD.

Here you can see the Output of my riju.php file:



## **Conclusion:**

In closing, this documentation has guided through the process of deploying a PHP file on an Amazon EC2 instance using the LAMP stack. By following the steps outlined here, I have gained practical experience in cloud hosting and web development.

**The End**

