

## WEB STACK IMPLEMENTATION (LEMP STACK)

**Definition:** LEMP (Linux, Nginx, MySQL, PHP or Python, or Perl) A technology stack is a set of technologies that are stacked together to build any application. Popularly known as a technology infrastructure or solutions stack, technology stack has become essential for building easy-to-maintain, scalable web applications.

The technology stack determines the type of applications you can build, the level of customizations you can perform, and the resources you need to develop your application.

### Implementation:

**Step 1 Launch Instance:** Go to your AWS console, Launch an EC2 Instance, the Ubuntu Free tier, create a key pair,

**Name and tags** Info

Name

LEMP STACK SERVER

Add additional tags

**Application and OS Images (Amazon Machine Image)** Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type  
ami-04e601abe3e1a910f (64-bit (x86)) / ami-0329d3839379bdf15 (64-bit (Arm))  
Virtualization: hvm    ENA enabled: true    Root device type: ebs

Free tier eligible

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-05-16

Architecture

AMI ID

64-bit (x86)

ami-04e601abe3e1a910f

Verified provider

**Instance type** Info

Instance type

t2.micro

Family: t2    1 vCPU    1 GiB Memory    Current generation: true  
On-Demand Windows pricing: 0.018 USD per Hour  
On-Demand Linux pricing: 0.0134 USD per Hour  
On-Demand SUSE pricing: 0.0134 USD per Hour  
On-Demand RHEL pricing: 0.0734 USD per Hour

Free tier eligible

All generations

Compare instance types

**Summary**

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...read more  
ami-04e601abe3e1a910f

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

Launch instance

Review commands

then go to network settings select edit, check create new security groups. Give it a name, in this case will call mine lempSG, by default you should have both ssh and http(but if there's no http, you can select the add security group rule to select http)

▼ Network settings Info

VPC - required Info

vpc-09c514bbc4f1ef994172.31.0.0/16(default)↻

Subnet Info

No preference↻

↻ Create new subnet ↗

Auto-assign public IP Info

Enable↻

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group☐ Select existing security group

Security group name - required

lempSG

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and .\_-:/()#,@[]+=&;!\$\*

Description - required Info

lempSG, for lemp stack server

Inbound security groups rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)Remove

Type Info <div>ssh↻</div>	Protocol Info <div>TCP</div>	Port range Info <div>22</div>
Source type Info <div>Anywhere↻</div>	Source Info <div>Q Add CIDR, prefix list or security0.0.0.0/0✕</div>	Description - optional Info <div>e.g. SSH for admin desktop</div>

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.✕

Add security group rule

**Inbound security groups rules**

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) Remove

Type [Info](#) Protocol [Info](#) Port range [Info](#)

ssh TCP 22

Source type [Info](#) Source [Info](#) Description - optional [Info](#)

Anywhere  e.g. SSH for admin desktop

0.0.0.0/0 ✕

▼ Security group rule 2 (TCP, 80) Remove

Type [Info](#) Protocol [Info](#) Port range [Info](#)

HTTP TCP 80

Source type [Info](#) Source [Info](#) Description - optional [Info](#)

Custom  e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. ✕

Add security group rule

Then proceed to launch the instance.

After confirmation of instance up and running as shown below, then ssh into the instance.

ID	Instance state ▼	Instance type ▼	Status check	Alarm status	Availability Zone ▼
c000a27ed36	Running	t2.micro	2/2 checks passed	No alarms +	eu-central-1b

## Step 2 Installing the Nginx Web Server:

We then setup the nginx server by running the following commands accordingly:

```
$sudo apt update
```

```
$sudo apt install nginx
```

To verify that nginx was successfully installed and is running as a service in Ubuntu as shown below, run:

```
$sudo systemctl status nginx
```

```
ubuntu@ip-172-31-19-183:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-06-08 00:11:17 UTC; 1min 18s ago
     Docs: man:nginx(8)
   Process: 2215 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 2216 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Main PID: 2309 (nginx)
    Tasks: 2 (limit: 1141)
   Memory: 4.4M
      CPU: 23ms
   CGroup: /system.slice/nginx.service
           └─2309 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─2312 "nginx: worker process"

Jun 08 00:11:17 ip-172-31-19-183 systemd[1]: Starting A high performance web server and a reverse proxy server...
Jun 08 00:11:17 ip-172-31-19-183 systemd[1]: Started A high performance web server and a reverse proxy server.
lines 1-16/16 (END)
```

First, let us try to check how we can access it locally in our Ubuntu shell, run the command:

```
$ curl http://localhost:80
```

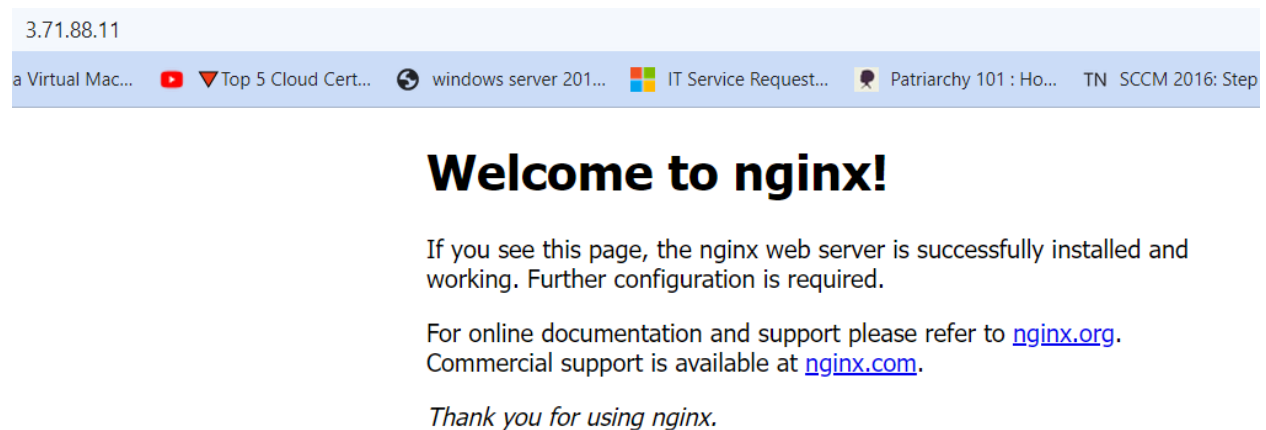
Then the image below is what you should get

```
ubuntu@ip-172-31-19-183:~$ curl http://localhost:80
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Another way to find out is to go to your AWS console, from the instance copy the public ipv4 ip address and paste in your browser, please remember its http not https before the ip address as its http port the we allowed access from.



**STEP 3** Installing SQL then we have to install SQL using the commands

- `sudo apt install mysql-server`
- `sudo MySQL`

Then the screenshot below confirms sql has been installed

Then run this command **mysql> exit**

To return to the cli

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-19-183:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> |
```

**STEP 4** Installing php is what next we have to do and that is accomplished by using the command: **sudo apt install php-fpm php-mysql**

```
/ Creating config file /etc/php/8.1/cli/php.ini with new version
Setting up php8.1-fpm (8.1.2-1ubuntu2.11) ...
S
S Creating config file /etc/php/8.1/fpm/php.ini with new version
Created symlink /etc/systemd/system/multi-user.target.wants/php8.1-fpm.service → /lib/systemd/system/php8.1-fpm.service.
S Setting up php-fpm (2:8.1+92ubuntu1) ...
Processing triggers for man-db (2.10.2-1) ...
S Processing triggers for php8.1-cli (8.1.2-1ubuntu2.11) ...
Processing triggers for php8.1-fpm (8.1.2-1ubuntu2.11) ...
S Scanning processes...
Scanning linux images...
S
S Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-19-183:~$
```

**STEP 5** Configuring Nginx To Use Php Processor by running the command in bold.

Create the root web directory.

**sudo mkdir /var/www/projectLEMP**

This assigns ownership of the directory with the \$USER environment variable, which will reference your current system user

**sudo chown -R \$USER:\$USER /var/www/projectLEMP**

This opens a new configuration file in Nginx's sites-available directory, This will create a new blank file.

**sudo nano /etc/nginx/sites-available/projectLEMP**

Paste in the following bare-bones configuration:

***#/etc/nginx/sites-available/projectLEMP***

***server {***

***listen 80;***

***server\_name projectLEMP www.projectLEMP;***

***root /var/www/projectLEMP;***

***index index.html index.htm index.php;***

***location / {***

***try\_files \$uri \$uri/ =404;***

```

}
location ~ /\.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
}
location ~ /\.ht {
    deny all;
}
}

```

Activate your configuration by linking to the config file from Nginx's sites-enabled directory:

```
sudo ln -s /etc/nginx/sites-available/projectLEMP /etc/nginx/sites-enabled/
```

- Test your configuration for syntax errors by typing:

**sudo nginx -t**

```

ubuntu@ip-172-31-19-183:~$ sudo mkdir /var/www/projectLEMP
ubuntu@ip-172-31-19-183:~$ sudo chown -R $USER:$USER /var/www/projectLEMP
ubuntu@ip-172-31-19-183:~$ sudo nano /etc/nginx/sites-available/projectLEMP
ubuntu@ip-172-31-19-183:~$ sudo vim /etc/nginx/sites-available/projectLEMP
ubuntu@ip-172-31-19-183:~$ sudo ln -s /etc/nginx/sites-available/projectLEMP /etc/nginx/sites-enabled/
ubuntu@ip-172-31-19-183:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

```

We need to disable the default Nginx host that is currently configured to listen on port 80, for this run:

```
sudo unlink /etc/nginx/sites-enabled/default
```

When you are ready, reload Nginx to apply the changes:

```
sudo systemctl reload nginx
```

Create an index.html file in that location so that we can test that your new server block works as expected:

```

sudo echo 'Hello LEMP from hostname' $(curl -s http://169.254.169.254/latest/meta-data/public-hostname)
'with public IP' $(curl -s http://169.254.169.254/latest/meta-data/public-ipv4) >
/var/www/projectLEMP/index.html

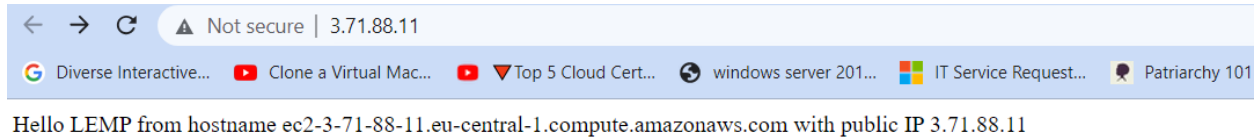
```

```

ubuntu@ip-172-31-19-183:~$ sudo unlink /etc/nginx/sites-enabled/default
ubuntu@ip-172-31-19-183:~$ sudo systemctl reload nginx
ubuntu@ip-172-31-19-183:~$ sudo echo 'Hello LEMP from hostname' $(curl -s http://169.254.169.254/latest/meta-data/public
-hostname) 'with public IP' $(curl -s http://169.254.169.254/latest/meta-data/public-ipv4) > /var/www/projectLEMP/index.
html
ubuntu@ip-172-31-19-183:~$ |

```

Test your website URL using IP address from the ec2 instance, as shown below (in some cases you might have to put:80 after the ip address to test):



## STEP 6 Testing Php With Nginx

At this point, your LAMP stack is completely installed, but we need to test it to validate that Nginx can correctly hand .php files off to your PHP processor by using the command below

**`sudo nano /var/www/projectLEMP/info.php`**

Type or paste the following lines into the new file. This is valid PHP code that will return information about your server:

**`<?php`**

**`phpinfo();`**

You can now access this page in your web browser by going to our browser

**`http://`ec2-ipaddress`/info.php`**



## PHP Version 8.1.2-1ubuntu2.11



System	Linux ip-172-31-19-183 5.19.0-1025-aws #26~22.04.1-Ubuntu SMP Mon Apr 24 01:58:15 UTC 2023 x86_64
Build Date	Feb 22 2023 22:56:18
Build System	Linux
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/fpm
Loaded Configuration File	/etc/php/8.1/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/fpm/conf.d
Additional .ini files parsed	/etc/php/8.1/fpm/conf.d/10-mysqld.ini, /etc/php/8.1/fpm/conf.d/10-opcache.ini, /etc/php/8.1/fpm/conf.d/10-pdo.ini, /etc/php/8.1/fpm/conf.d/20-calendar.ini, /etc/php/8.1/fpm/conf.d/20-ctype.ini, /etc/php/8.1/fpm/conf.d/20-exif.ini, /etc/php/8.1/fpm/conf.d/20-ffi.ini, /etc/php/8.1/fpm/conf.d/20-fileinfo.ini, /etc/php/8.1/fpm/conf.d/20-ftp.ini, /etc/php/8.1/fpm/conf.d/20-gettext.ini, /etc/php/8.1/fpm/conf.d/20-iconv.ini, /etc/php/8.1/fpm/conf.d/20-mysqli.ini, /etc/php/8.1/fpm/conf.d/20-pdo_mysql.ini, /etc/php/8.1/fpm/conf.d/20-phar.ini, /etc/php/8.1/fpm/conf.d/20-posix.ini, /etc/php/8.1/fpm/conf.d/20-readline.ini, /etc/php/8.1/fpm/conf.d/20-shmop.ini, /etc/php/8.1/fpm/conf.d/20-sockets.ini, /etc/php/8.1/fpm/conf.d/20-sysvmsg.ini, /etc/php/8.1/fpm/conf.d/20-sysvsem.ini, /etc/php/8.1/fpm/conf.d/20-sysvshm.ini, /etc/php/8.1/fpm/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3

### Step 7: Retrieving data from mysql database using php

We will be creating a database using mysql which was installed some steps ago. To do this just type the command below

`Sudo mysql -p`

We are using -p because we have put a password on the mysql module. With out this you would get the error message you can see in the image below

```
ubuntu@ip-172-31-19-183:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> mysql> CREATE DATABASE 'example_database';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
for the right syntax to use near 'mysql> CREATE DATABASE 'example_database'' at line 1
mysql> CREATE DATABASE 'example_database1';
Query OK, 1 row affected (0.02 sec)

mysql> |
```

```

ubuntu@ip-172-31-19-183:~$ sudo mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> mysql> CREATE DATABASE 'example_database';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL s
for the right syntax to use near 'mysql> CREATE DATABASE 'example_database'' at line 1
mysql> CREATE DATABASE 'example_database1';
Query OK, 1 row affected (0.02 sec)

mysql> CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_password BY 'password';
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements
mysql> CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_password BY 'password1';
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements
mysql> CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_password BY 'password1.';
Query OK, 0 rows affected (0.00 sec)

mysql> |

```

You can test if the new user has the proper permissions by logging in to the MySQL console again, this time using the custom user credentials:

```

ubuntu@ip-172-31-19-183:~$ mysql -u example_user -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| example_database1 |
| information_schema |
| performance_schema |
+-----+
3 rows in set (0.02 sec)

mysql> |

```

To confirm that the data was successfully saved to your table, run:

```
Mysql > SELECT * FROM example_database.todo_list;
```

You'll see the following output:

```
mysql> INSERT INTO example_database1.todo_list (content) VALUES ("My first important item");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO example_database1.todo_list (content) VALUES ("My second important item");
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO example_database1.todo_list (content) VALUES ("My third important item");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO example_database1.todo_list (content) VALUES ("My fourth important item");
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM example_database1.todo_list;
+-----+-----+
| item_id | content                |
+-----+-----+
|      1 | My first important item |
|      2 | My second important item |
|      3 | My third important item  |
|      4 | My fourth important item |
+-----+-----+
4 rows in set (0.00 sec)

mysql> |
```

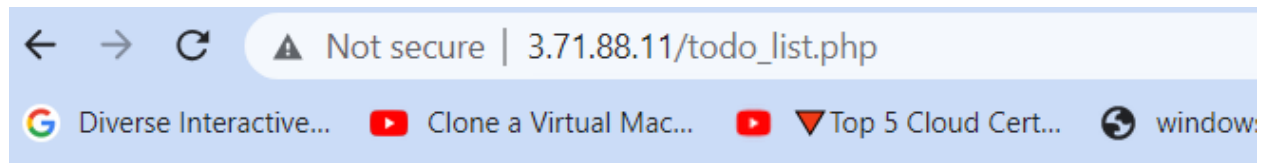
The following PHP script connects to the MySQL database and queries for the content of the todo\_list table, displays the results in a list. If there is a problem with the database connection, it will throw an exception.

Copy this content into your todo\_list.php script:

```
GNU nano 6.2 /var/www/projectLEMP/todo_list.php *
<?php
$user = "example_user";
$password = "password"1;
$database = "example_database1";
$table = "todo_list";

try {
    $db = new PDO("mysql:host=localhost;dbname=$database", $user, $password);
    echo "<h2>TODO</h2><ol>";
    foreach($db->query("SELECT content FROM $table") as $row) {
        echo "<li>" . $row['content'] . "</li>";
    }
    echo "</ol>";
} catch (PDOException $e) {
    print "Error!: " . $e->getMessage() . "<br/>";
    die();
}
```

You can now access this page in your web browser by visiting the domain name or public IP address configured for your website, followed by /todo\_list.php:



# TODO

1. My first important item
2. My second important item
3. My third important item
4. My fourth important item