WEB STACK-IMPLEMENTATION (LEMP)

The goal of this project is to describe the concepts of Continuous Integration, Continuous Delivery / Deployment and DevOps on a Lemp web stack.

if you have followed the other project one, I am sure you know what a stack is. Today we will work with the LEMP STACK.

LEMP => LINUX, NGINX, MySQL, PHP

*nginx IS PRONOUNCED "enginX"

Tldr;

#Video link

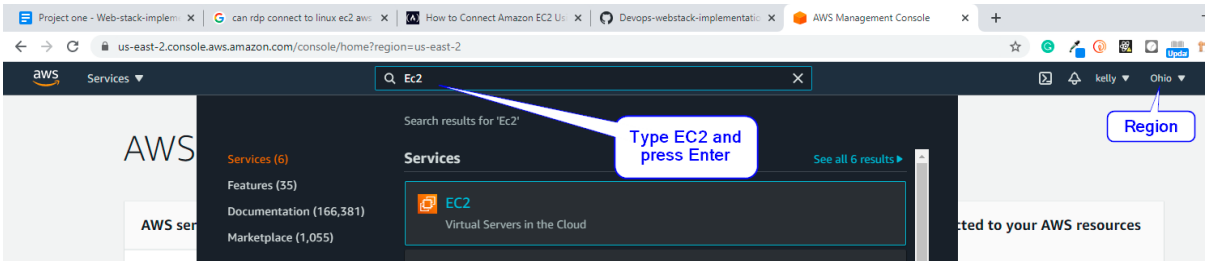
Prerequisites:

- Aws account running an EC2 instance
- Internet connection
- Fundamental Knowledge of downloading and installing
- Basics Linux skills

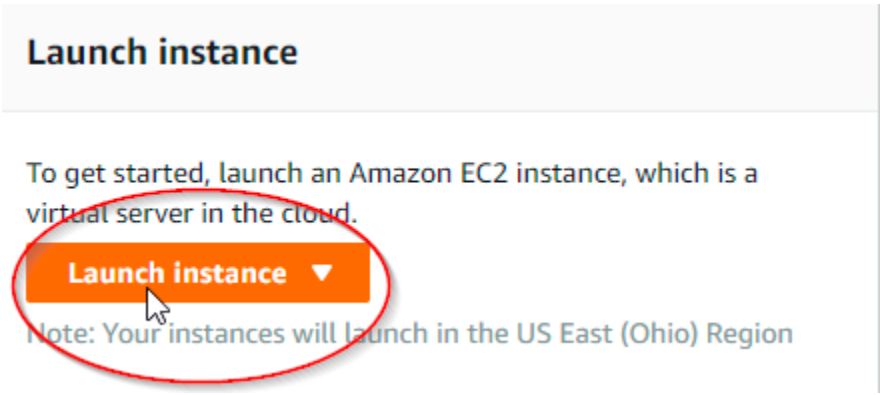
Implementation

- Open your PC browser and login to https://aws.amazon.com/

- A region is selected by default (change if necessary), from the search bar type EC2 and click.

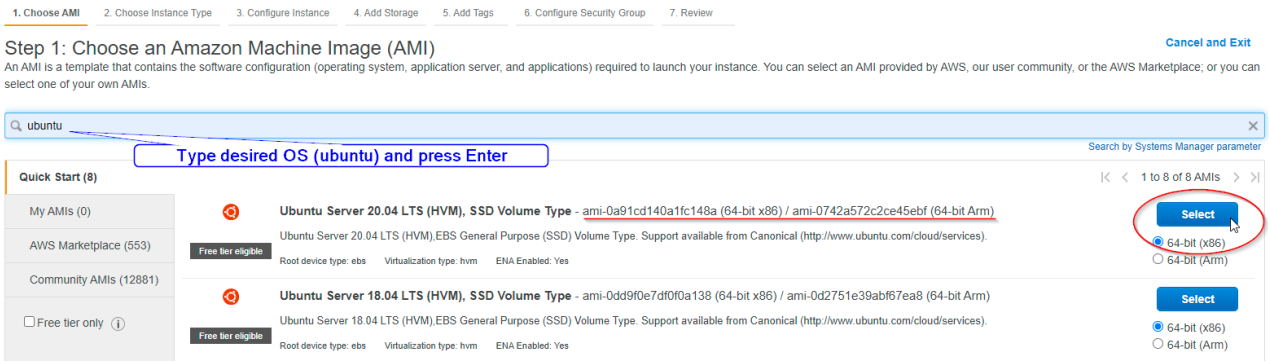


- From the Ec2 dashboard, click on the button "Launch instance" to start using a virtual server.

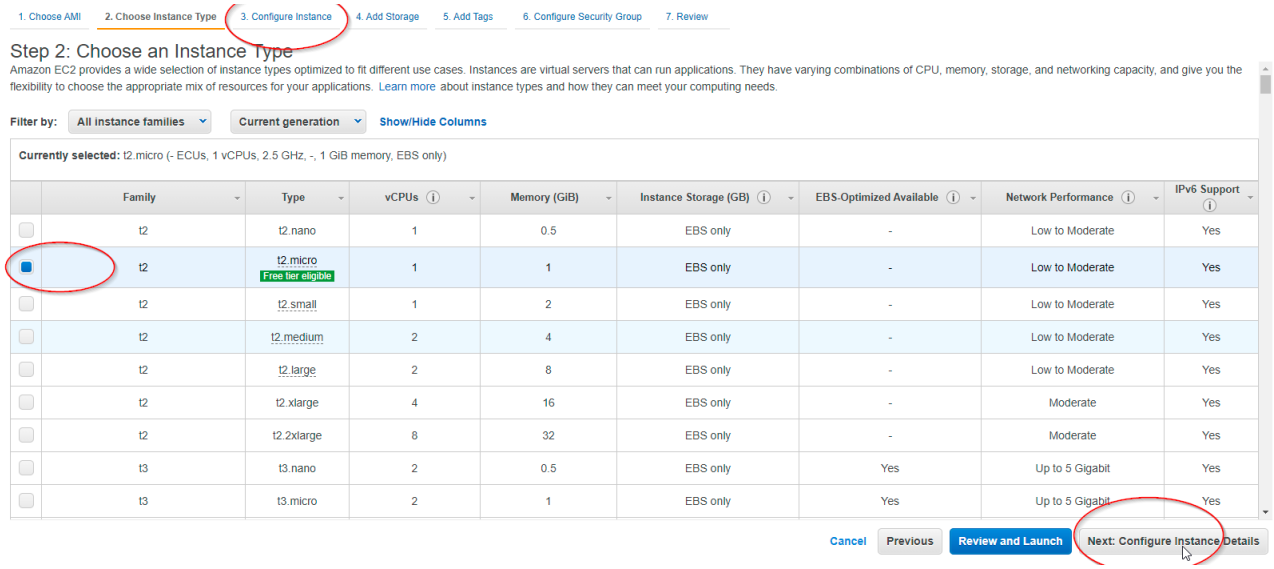


- An AMI window displays, type "Ubuntu" on the search bar and hit enter, or scroll down to select "Ubuntu Server 20.04 LTS (HVM), SSD Volume Type" based on your system architecture.

Note: the AMI (Amazon machine image) is always different from user to user



- The next step of configuring our EC2 is to select the instance type, preferably a **t2 micro - Free tier.** Then click (3) configure instance showing at the top or click next configuration details at the bottom.

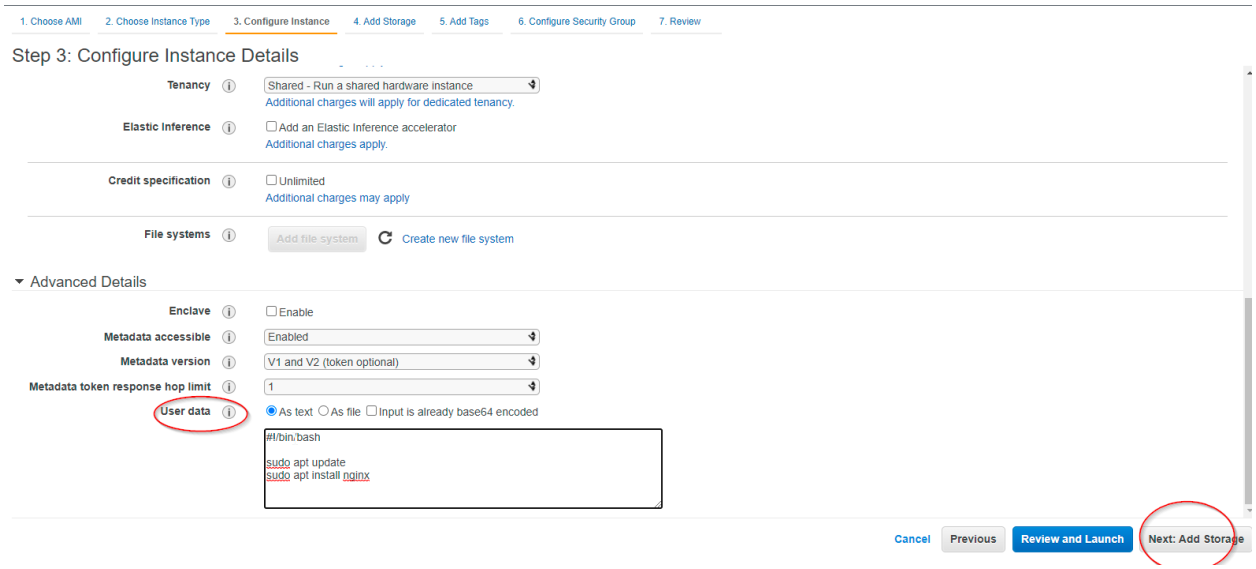


Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
t2	t2.xlarge	4	16	EBS only	-	Moderate	Yes
t2	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
t3	t3.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes
t3	t3.micro	2	1	EBS only	Yes	Up to 5 Gigabit	Yes

Move to next step

- To configure the instance, we will leave all default but scroll to the bottom and on the advanced details section, in the user data column add below script as shown on the screenshot.

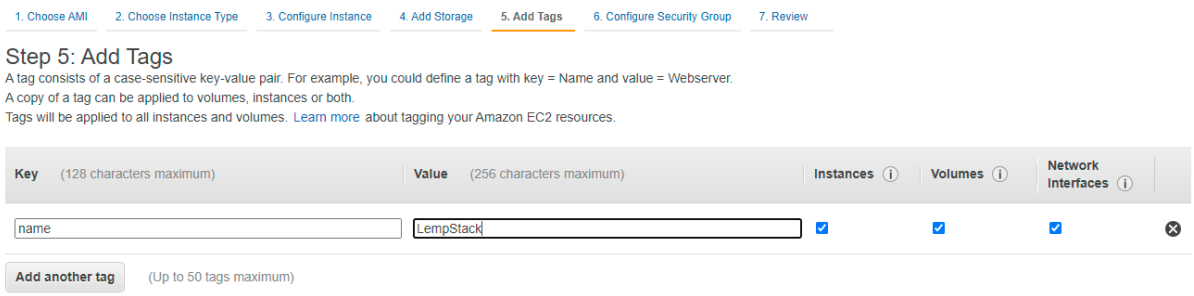
```
#!/bin/bash

sudo apt update

sudo apt install nginx
```



Move to next step

- Move to tab 5 to Add tags to our EC2 instance, I have deliberately skipped tab 4 to choose the default storage volume given by AWS.

Tags are key-value paired fields and help to categorize your AWS resource, now click ADD TAG to assign a unique name and move to next.



Move to Next

- We will modify the default security group later to give access to **port 80, 22 and**.

Reason: The security Group are set of firewall rules which denies and grant access to our EC2 instance,

To access the EC2 instance with a console, we need port 22 opened accessed via SSH and through a browser externally we need port 80 opened

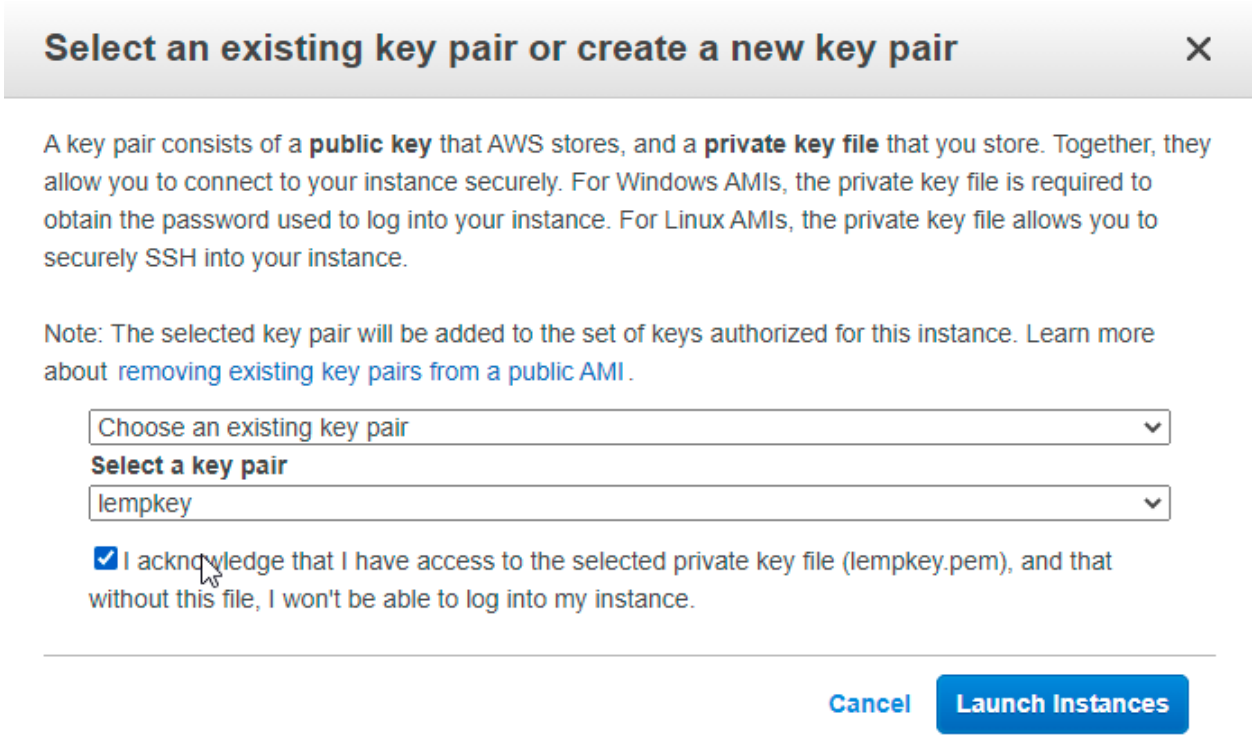
You may add descriptions on the last column.

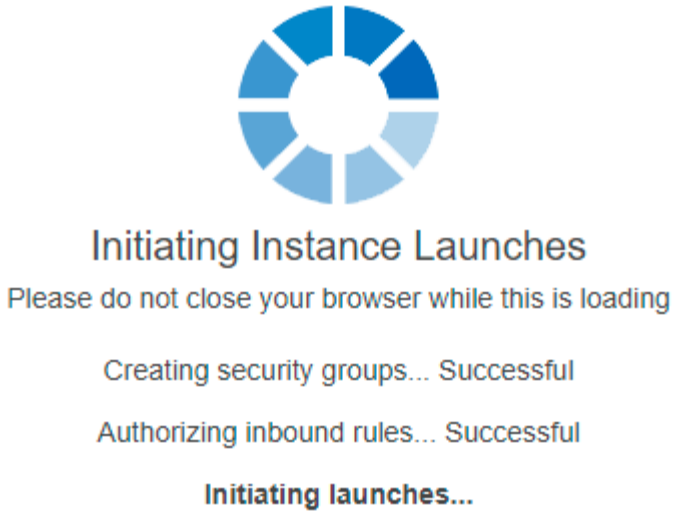
Leaving the default, Security.

- Click review and launch

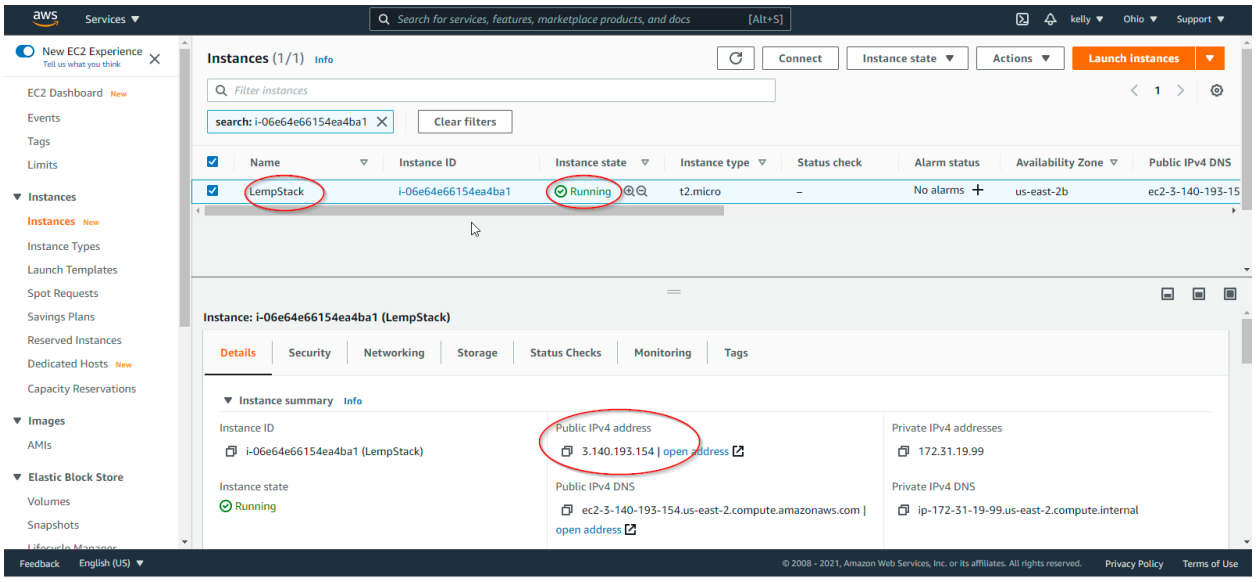
You will get a Prompt to Create a Private Key File, feel free to choose an existing one, if it already exists on the same PC.

Download the key file to a good location, to be used later, Then Launch.





Done? Good Job, let's get to business now.



Copy your own Public IP as shown on the above screenshot, nowit's time to use the console

Yay!!!

Open git bash or putty or mobaxterm, whichever console is suitable, else download.

We are using git bash here:

```
MINGW64:/c/Users/USER/Downloads

USER@LENOVO MINGW64 ~/Downloads
$ ssh -i ./keys/lempkey.pem ubuntu@3.140.193.154
The authenticity of host '3.140.193.154 (3.140.193.154)' can't be established.
ECDSA key fingerprint is SHA256:kOONTGxqgF5I6MwyhGkz6W+LuYaV8OkIJOw7m11VFmU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Type YES, to connect.

You have now connected to the EC2 instance via SSH

Type clear, to have a neat console and proceed.

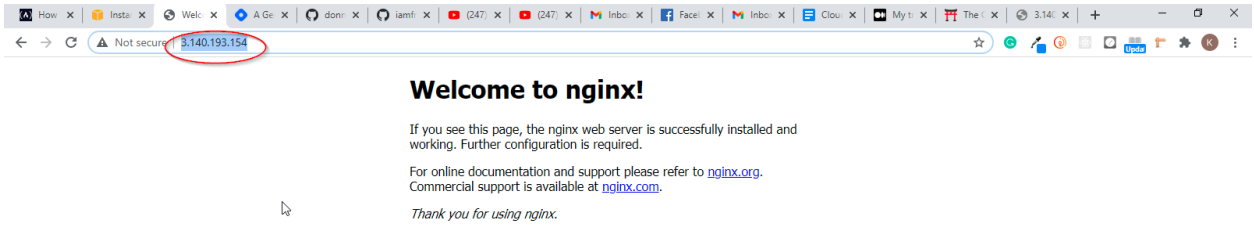
We will now check if our userdata scripts were loaded after getting our Public IP from below command.

Type: curl http://169.254.169.254/latest/user-data

By default EC2 user is given sudo privilege

```
ubuntu@ip-172-31-19-99:~$ curl localhost
<!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>
ubuntu@ip-172-31-19-99:~$
```

To check if nginx installation also worked, visit <http://public ip> on browser to confirm.



Great!

So from the LEMP stack, we have got Linux and Nginx ready, let's get MySQL running now.

Run this code:

```
$ sudo apt install MySQL-server –y
```

```
Unpacking libevent-core-2.1-7:amd64 (2.1.11-stable-1) ...
Selecting previously unselected package libevent-pthreads-2.1-7:amd64.
Preparing to unpack .../4-libevent-pthreads-2.1-7_2.1.11-stable-1_amd64.deb ...
Unpacking libevent-pthreads-2.1-7:amd64 (2.1.11-stable-1) ...
Selecting previously unselected package libmecab2:amd64.
Preparing to unpack .../5-libmecab2_0.996-10build1_amd64.deb ...
Unpacking libmecab2:amd64 (0.996-10build1) ...
Selecting previously unselected package mysql-server-core-8.0.
Preparing to unpack .../6-mysql-server-core-8.0_8.0.23-0ubuntu0.20.04.1_amd64.deb ...
Unpacking mysql-server-core-8.0 (8.0.23-0ubuntu0.20.04.1) ...

Progress: [ 13%] [###########.....................................................................]
```

Next, we need to configure MySQL to secure authentication.

Run this code:

$ sudo mysql_secure_installation

```
ubuntu@ip-172-31-19-99: ~
ubuntu@ip-172-31-19-99:~$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: y

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary                  file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG:
```

Type yes to continue, on the next prompt choose 0 or 2 and enter desired password to continue, then type y or yes to continue for ALL prompt.

Great MYSQL is installed and configured for use, we could test by running below code:

$ sudo MySQL

```
ubuntu@ip-172-31-19-99:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.23-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, 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>
```

```
Success.

All done!
ubuntu@ip-172-31-19-99:~$
```

Type exit to leave MySQL console editor.

MySQL> exit

Linux, Nginx and MySQL checked, now let's install PHP and required dependencies using below command:

PHP INSTALLATION

Run below code

```
$ sudo apt install php-fpm php-mysql -y
```

```
Preparing to unpack .../8-php7.4-mysql_7.4.3-4ubuntu2.4_amd64.deb ...
Unpacking php7.4-mysql (7.4.3-4ubuntu2.4) ...
Selecting previously unselected package php-mysql.
Preparing to unpack .../9-php-mysql_2%3a7.4+75_all.deb ...
Unpacking php-mysql (2:7.4+75) ...
Setting up php-common (2:75) ...
Created symlink /etc/systemd/system/timers.target.wants/phpsessionclean.timer → /lib/systemd/system
ean.timer.

Progress: [ 51%] [##################################################.....................................
```

Now it's installed, check PHP version using this command:

```
php -version
```

Next, we **Configuring Nginx to Use PHP Processor**

Next step,let's make a dir. for our site directory, Run below Command

```
$ sudo mkdir /var/www/projectLEMP

$ sudo chown -R $USER:$USER /var/www/projectLEMP
```

Run below command, this is done to edit the new site directory:

```
$ sudo vim /etc/nginx/sites-available/projectLEMP
```

Type "I" without the quotes to type the below virtual host file in the config created, then press ESC and exit with " :wq " command

```
#/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/php7.4-fpm.sock;
     }

    location ~ /\.ht {
        deny all;
    }

}
```

Next we activate the configuration by linking to the Nginx config file using below command

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

$ sudo nginx -t
```

```
ubuntu@ip-172-31-19-99: ~
ubuntu@ip-172-31-19-99:~$ sudo mkdir /var/www/projectLEMP
ubuntu@ip-172-31-19-99:~$ sudo chown -R $USER:$USER /var/www/projectLEMP
ubuntu@ip-172-31-19-99:~$ sudo vi /etc/nginx/sites-available/projectLEMP
ubuntu@ip-172-31-19-99:~$ sudo ln -s /etc/nginx/sites-available/projectLEMP /etc/nginx/sites-enabled/
ubuntu@ip-172-31-19-99:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-172-31-19-99:~$
```

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

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

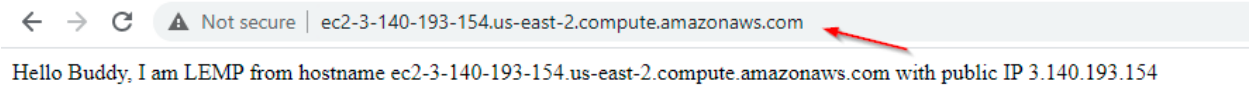
$ sudo systemctl reload nginx

To test the new server block, we will add an index file to the new web root using below command

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

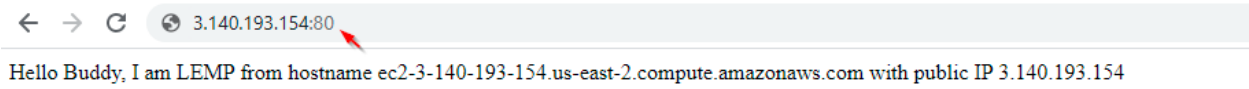
Good.

Refresh browser and check now, Hola!!!



Hello Buddy, I am LEMP from hostname ec2-3-140-193-154.us-east-2.compute.amazonaws.com with public IP 3.140.193.154

http://<public ip>:80 is set.



Hello Buddy, I am LEMP from hostname ec2-3-140-193-154.us-east-2.compute.amazonaws.com with public IP 3.140.193.154

Your LEMP stack is now fully configured. In the next step, we'll create a PHP script to test that Nginx is in fact able to handle .php files within your newly configured website.

Testing PHP with Nginx

To cap it all up, we need to serve php files, then we need to tweak a file, and make index.php the first directory index as shown below.

Run this command

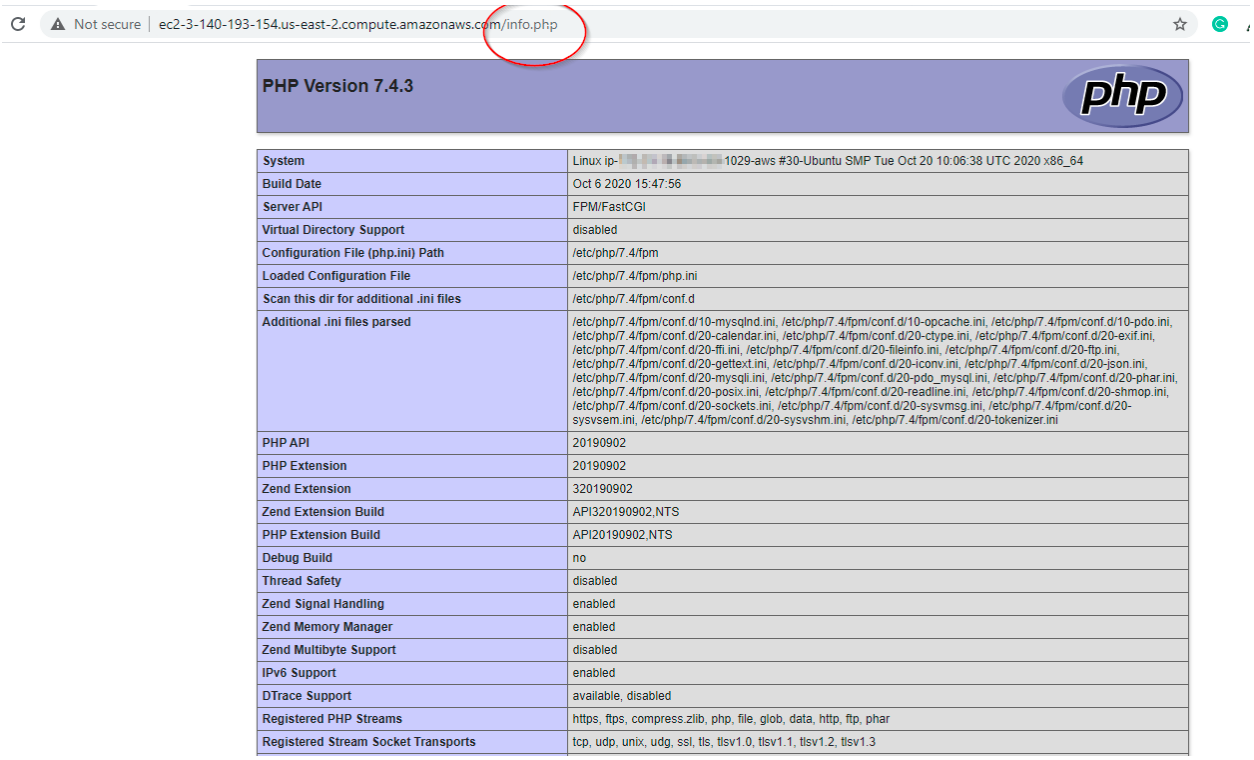
```
$ sudo vi  /var/www/projectLEMP/info.php
```

Add the content as shown below

ubuntu@ip-172-31-19-99: ~

```
<?php
phpinfo();
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Then reload browser to check the ip with a route **of** **/info.php** to



Not secure | ec2-3-140-193-154.us-east-2.compute.amazonaws.com/info.php

PHP Version 7.4.3

System	Linux ip-[redacted]1029-aws #30-Ubuntu SMP Tue Oct 20 10:06:38 UTC 2020 x86_64
Build Date	Oct 6 2020 15:47:56
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/fpm
Loaded Configuration File	/etc/php/7.4/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/fpm/conf.d
Additional .ini files parsed	/etc/php/7.4/fpm/conf.d/10-mysqlnd.ini, /etc/php/7.4/fpm/conf.d/10-opcache.ini, /etc/php/7.4/fpm/conf.d/10-pdo.ini, /etc/php/7.4/fpm/conf.d/20-calendar.ini, /etc/php/7.4/fpm/conf.d/20-ctype.ini, /etc/php/7.4/fpm/conf.d/20-exif.ini, /etc/php/7.4/fpm/conf.d/20-ffi.ini, /etc/php/7.4/fpm/conf.d/20-fileinfo.ini, /etc/php/7.4/fpm/conf.d/20-ftp.ini, /etc/php/7.4/fpm/conf.d/20-gettext.ini, /etc/php/7.4/fpm/conf.d/20-iconv.ini, /etc/php/7.4/fpm/conf.d/20-json.ini, /etc/php/7.4/fpm/conf.d/20-mysqli.ini, /etc/php/7.4/fpm/conf.d/20-pdo_mysql.ini, /etc/php/7.4/fpm/conf.d/20-phar.ini, /etc/php/7.4/fpm/conf.d/20-posix.ini, /etc/php/7.4/fpm/conf.d/20-readline.ini, /etc/php/7.4/fpm/conf.d/20-shmop.ini, /etc/php/7.4/fpm/conf.d/20-sockets.ini, /etc/php/7.4/fpm/conf.d/20-sysvmsg.ini, /etc/php/7.4/fpm/conf.d/20-sysvsem.ini, /etc/php/7.4/fpm/conf.d/20-sysvshm.ini, /etc/php/7.4/fpm/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,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

We will remove the info file because it contains sensitive information about your PHP environment and your Ubuntu server. You can use rm to remove that file:

```
$ sudo rm /var/www/projectLEMP/info.php
```

Finally - **Retrieving data from MySQL database with PHP**

We will create a database with sample details and also a simple "To do list" and configure access to it, so the Nginx website would be able to query data from the DB and display it.

Now run below command:

```
sudo mysql

mysql> CREATE DATABASE `stores`;

mysql>  CREATE USER 'user1'@'%' IDENTIFIED WITH mysql_native_password BY 'userpass';

mysql> GRANT ALL ON stores.* TO 'user1'@'%';

exit

mysql -u user1 -p
```

```
mysql> CREATE DATABASE `stores`;
Query OK, 1 row affected (0.00 sec)

mysql>
```

```
mysql> CREATE USER 'user1'@'%' IDENTIFIED WITH mysql_native_password BY 'userpass';
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql>
mysql>
mysql> GRANT ALL ON stores.* TO 'user1'@'%';
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Having done above, we need to check the database and create a table to add content.

```
ubuntu@ip-172-31-19-99: ~
ubuntu@ip-172-31-19-99:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.23-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, 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           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.01 sec)

mysql> |
```

Run below code

```
CREATE TABLE stores.todolist (

mysql>    itemId INT auto_increment,

mysql>    content varchar(255),

mysql>     primary key(itemId)

mysql> );
```

```
mysql> create table stores.todolist(
    -> itemId int auto_increment,
    -> content varchar(255),
    -> primary key(itemId)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql>
```

Insert Content to Created Table:

```
mysql> INSERT INTO stores.todolist (content) VALUES ("Laptop");

INSERT INTO stores.todolist (content) VALUES ("Ear Buds");

INSERT INTO stores.todolist (content) VALUES ("Wristwatch");

INSERT INTO stores.todolist (content) VALUES ("Bible");

INSERT INTO stores.todolist (content) VALUES ("PS5");
```

```
mysql>
mysql> INSERT INTO stores.todolist (content) VALUES ("E
ar Buds");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO stores.todolist (content) VALUES ("W
ristwatch");
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO stores.todolist (content) VALUES ("B
ible");
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO stores.todolist (content) VALUES ("P
s5");
Query OK, 1 row affected (0.00 sec)

mysql> select * from stores.todolist;
+--------+------------+
| itemId | content    |
+--------+------------+
|      1 | Laptop     |
|      2 | Ear Buds   |
|      3 | Wristwatch |
|      4 | Bible      |
|      5 | PS5        |
+--------+------------+
5 rows in set (0.00 sec)

mysql>
```

To see the above content of the Table in the DB run the select ALL (*) command

```
mysql>  SELECT * FROM stores.todolist;
```

Now EXIT MySQL

We need to create a PHP script that will connect to MySQL and query for the content.

How?

First, we Create a new PHP file in your custom web root directory using your preferred editor. We'll use vi for that:

```
$ vi /var/www/projectLEMP/todolist.php
```

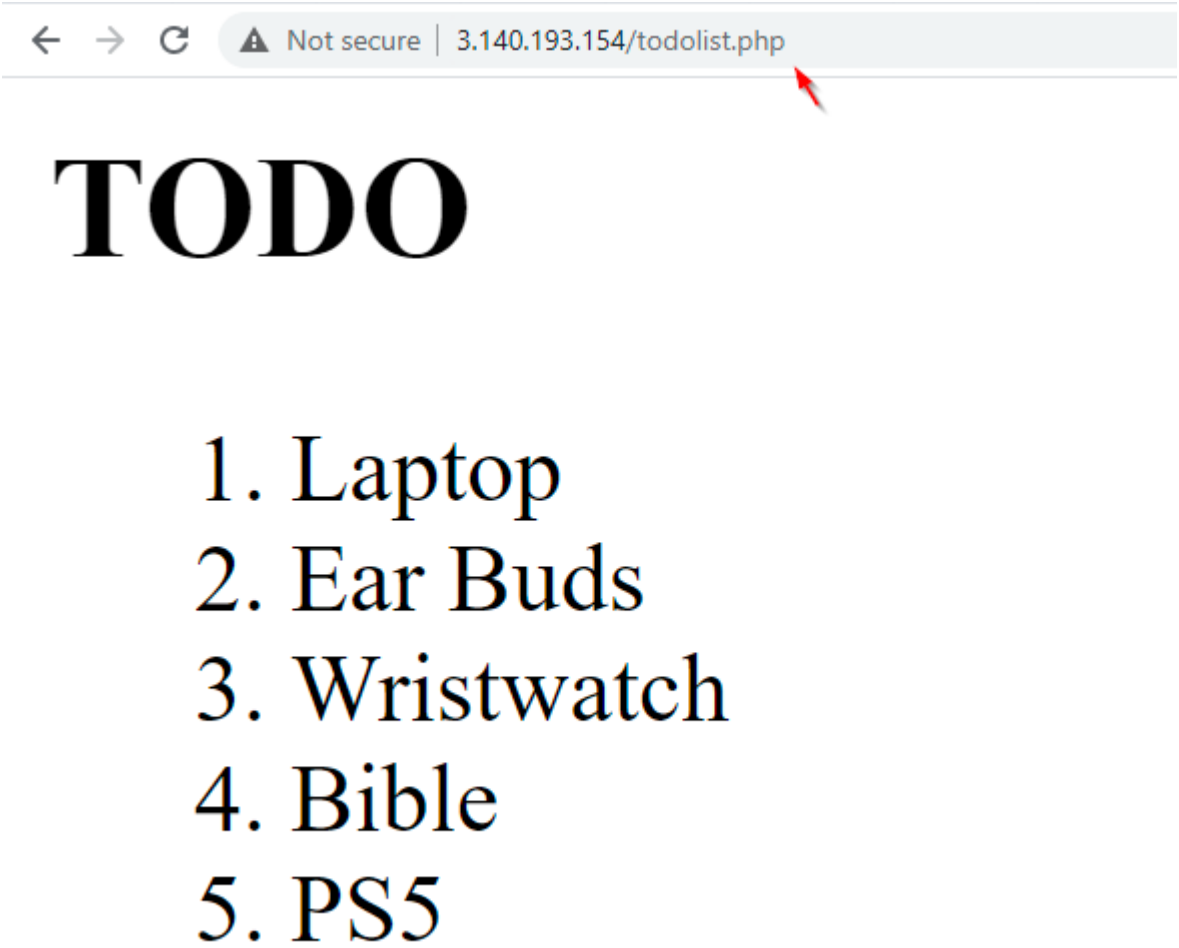
```
<?php

$user = "user1";

$password = "userpass";

$database = "stores";

$table = "todolist";



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();

}
```

Save and close the file we are done!!!

Check the browser on route /todolist.php

ubuntu@ip-172-31-19-99: ~

```php
<?php
$user = "user1";
$password = "userpass";
$database = "stores";
$table = "todolist";

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();
}

~
~
~
~
~
~
~
~
~
~
~
:wq
```



Congratulations, this is the requirement to set up an AWS instance with Linux, NginX, MySQL and PHP for a web project.

Hope this was informative.

PS: Remember to terminate your EC2 instance.