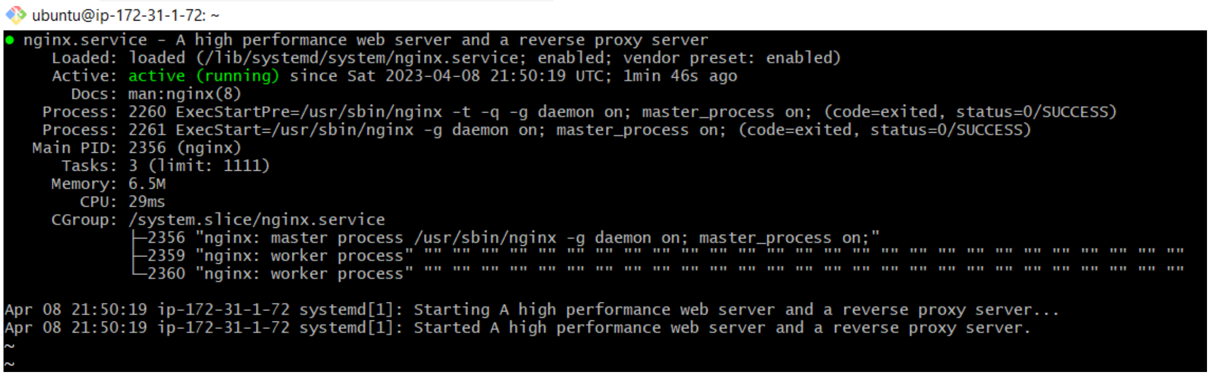
PROJECT 2: LEMP STACK IMPLEMENTATION

LEMP Stack is a combination of following components:

- Linux OS
- Nginx web server
- Mysql
- PHP

…………..INSTALLING NGINX WEB SERVER………………

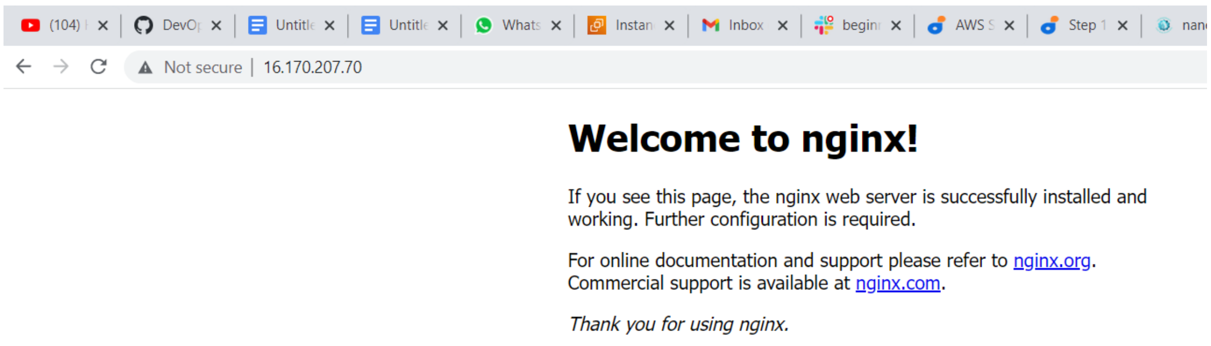
- `sudo apt update`
- `sudo apt install nginx`
- `sudo systemctl status nginx`



```
ubuntu@ip-172-31-1-72: ~
● 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 Sat 2023-04-08 21:50:19 UTC; 1min 46s ago
       Docs: man:nginx(8)
    Process: 2260 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 2261 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 2356 (nginx)
      Tasks: 3 (limit: 1111)
     Memory: 6.5M
        CPU: 29ms
     CGroup: /system.slice/nginx.service
             ├─2356 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─2359 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
             └─2360 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""

Apr 08 21:50:19 ip-172-31-1-72 systemd[1]: Starting A high performance web server and a reverse proxy server...
Apr 08 21:50:19 ip-172-31-1-72 systemd[1]: Started A high performance web server and a reverse proxy server.
~
~
```

To check if we can access it locally:

- `$ curl http://localhost:80`
- `curl http://127.0.0.1:80`



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

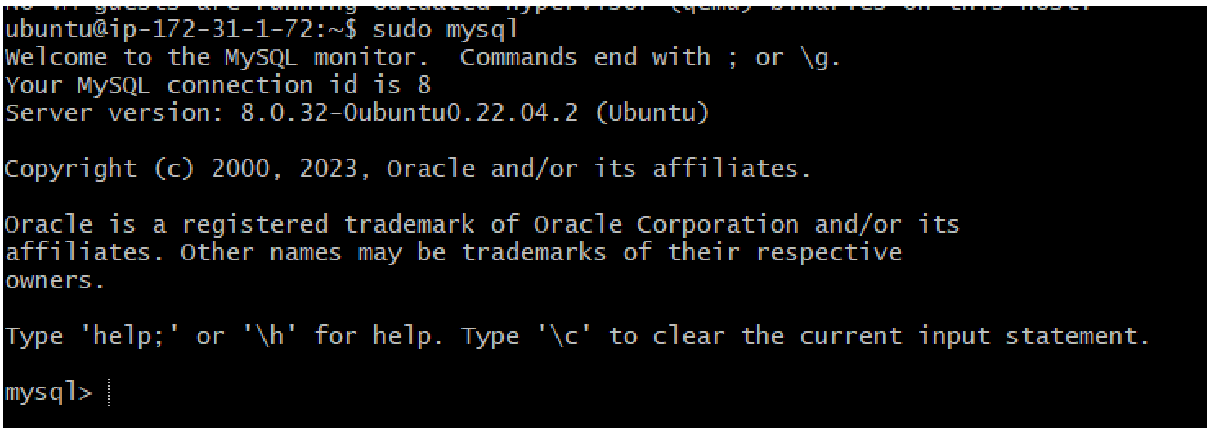
Thank you for using nginx.

…………………………..INSTALLING MYSQL……………………………..

- **`sudo apt install mysql-server`**

……………..To login……………………………….

- **`sudo mysql`**


```
ubuntu@ip-172-31-1-72:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.32-0ubuntu0.22.04.2 (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>
```

…………………………….INSTALLING PHP……………………

PHP is installed to process the code and generate dynamic content for the Webserver. Nginx requires an external program to handle PHP processing and act as a bridge between the PHP interpreter itself and the web server.

You'll need to install `php-fpm`, which stands for "PHP fastCGI process manager", and tell Nginx to pass PHP requests to this software for processing. Additionally, you'll need `php-mysql`, a PHP module that allows PHP to communicate with MySQL-based databases.

To install these 2 packages at once, run:

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

CONFIGURING NGINX TO USE PHP PROCESSOR

When using Enginx, we can create server blocks similar to virtual hosts in Apache to encapsulate configuration details and be able to host more than one Domain in a single server.

We are using the project daimain name called projectLEMP.

NB: On Ubuntu 20.04, Nginx has one server block enabled by default and is configured to serve documents out of a directory at
`/var/www/html.`

However, this works well for hosting a single site on Enginx webserver, it does not scale well for hosting multiple sites. Therefor for multiple sites hosting, we creat directory structure within `/var/www` for the **your_domain** website, leaving `/var/www/html` in place as the default directory to be served if a client request does not match any other sites.

Create the root web directory for projectLEMP as follows:

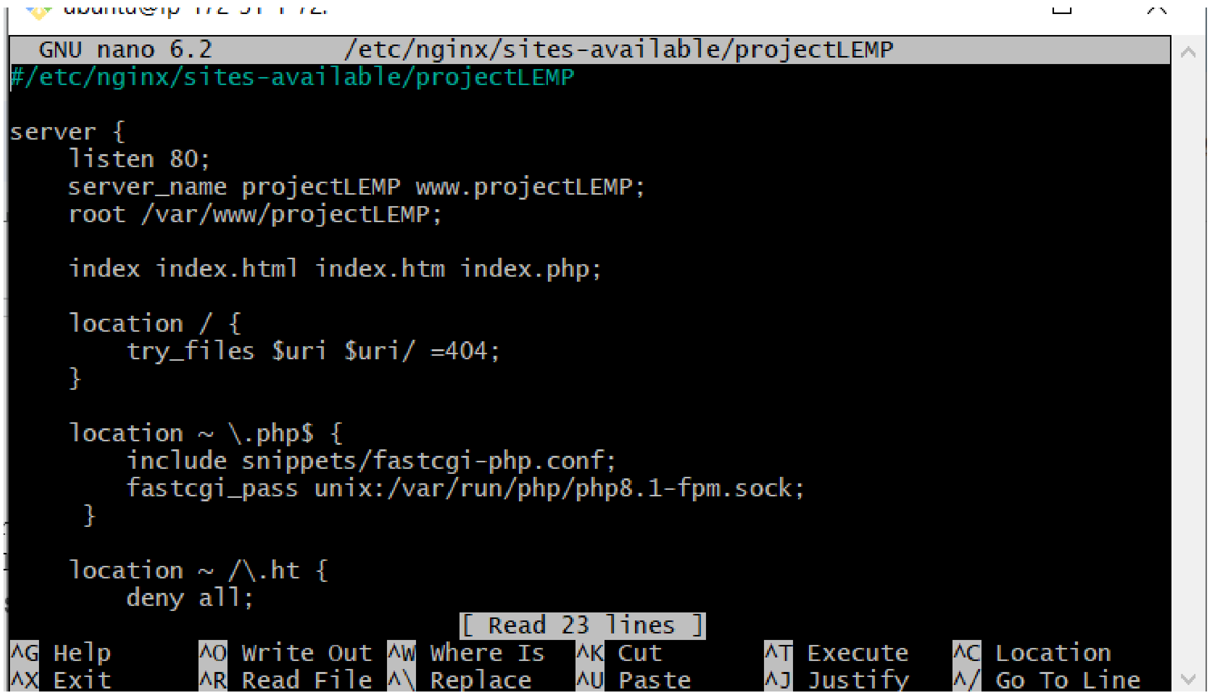
- `sudo mkdir /var/www/projectLEMP`

Assign ownership of the directory with the $USER environment variable, which will reference your current system user:

Then, open a new configuration file in Nginx's `sites-available` directory using your preferred command-line editor. Here, we'll use `nano`:

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

This created a blank file



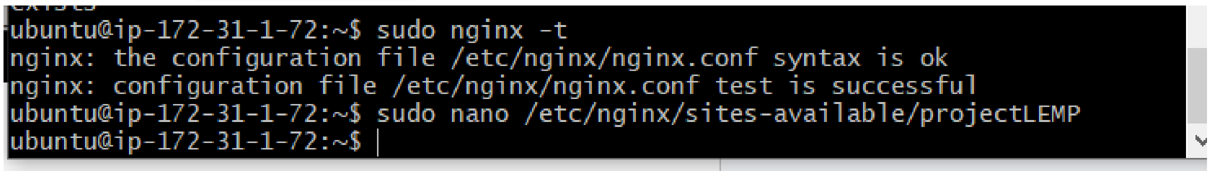
```
  GNU nano 6.2            /etc/nginx/sites-available/projectLEMP
#/etc/nginx/sites-available/projectLEMP

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;
                               [ Read 23 lines ]
^G Help        ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit        ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

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/`

You can test your configuration for syntax errors by typing:

- `sudo nginx -t`



```
ubuntu@ip-172-31-1-72:~$ 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-172-31-1-72:~$ sudo nano /etc/nginx/sites-available/projectLEMP
ubuntu@ip-172-31-1-72:~$ |
```

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

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

Reload Nginx to apply the changes:

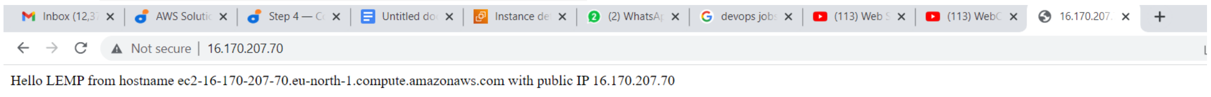
- `sudo systemctl reload nginx`

The new website is now active, but the web root /var/www/projectLEMP is still empty. 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`

Now from your browser, try to open your website URL using IP address:

- `http://<Public-IP-Address>:80`



Hello LEMP from hostname ec2-16-170-207-70.eu-north-1.compute.amazonaws.com with public IP 16.170.207.70

..................TESTING PHP WITH NGINX..................

The LEMP stack should now be completely set up.

We can test it to validate that Nginx can correctly hand `.php` files off to the PHP processor.

This can be done by creating a test PHP file in your document root. Open a new file called `info.php` within the document root in your text editor:

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

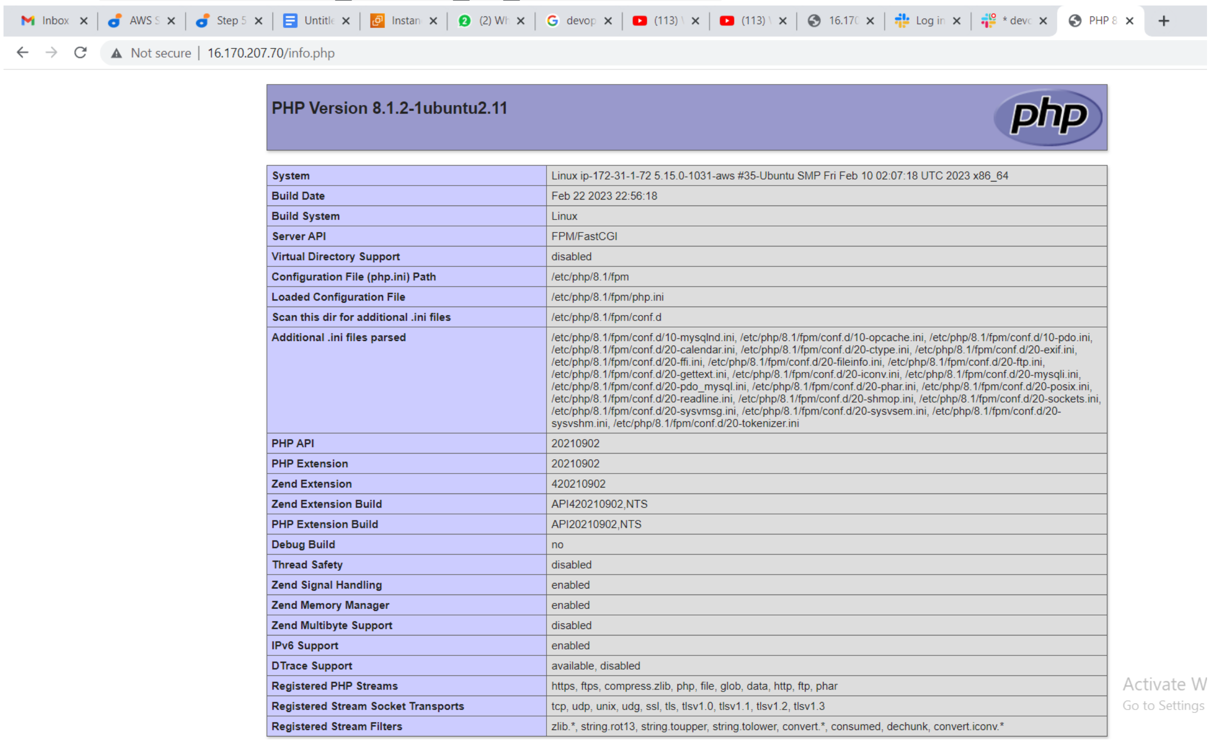
paste the following lines into the new file. This is valid PHP code that will return information about your server:

```
<?php
```

```
phpinfo();
```

This page can be accessed via the web browser by visiting the domain name or public IP address we've set up in our Nginx configuration file, followed by `/info.php`:

- `http://`server_domain_or_IP`/info.php`



PHP Version 8.1.2-1ubuntu2.11

System	Linux ip-172-31-1-72 5.15.0-1031-aws #35-Ubuntu SMP Fri Feb 10 02:07:18 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-mysqlnd.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
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, convert.iconv.*

We need to remove this file created as it contains vital info about the server:

- `sudo rm /var/www/your_domain/info.php`

RETRIEVING DATA FROM MYSQL DATABASE WITH PHP

AIM: To create a test database (DB) with 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.

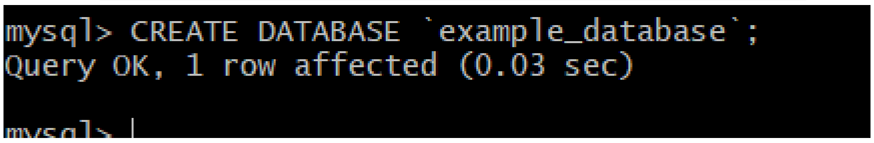
We created a database named **example_database** and a user named **example_user**, but you can replace these names with different values.

First, connect to the MySQL console using the **root** account:

- `sudo mysql`

To create a new database, run the following command from your MySQL console:

- `mysql> CREATE DATABASE `example_database`;`

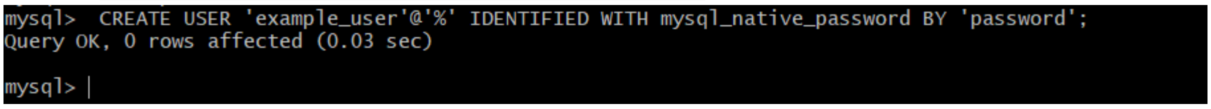


```
mysql> CREATE DATABASE `example_database`;
Query OK, 1 row affected (0.03 sec)

mysql> |
```

creating a new user and grant him full privileges on the database you have just created.

The following command creates a new user named `example_user`, using mysql_native_password as default authentication method. We're defining this user's password as `password`, but you should replace this value with a secure password of your own choosing.

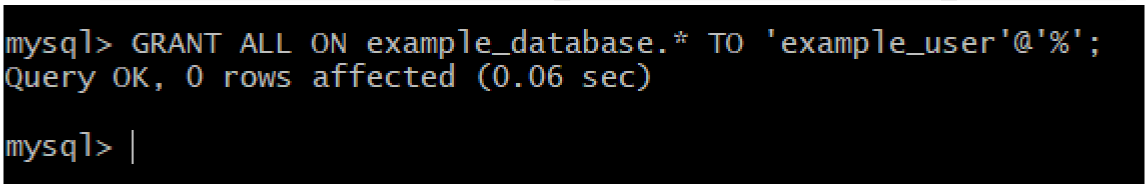
- `mysql>  CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_password BY 'password';`



```
mysql>  CREATE USER 'example_user'@'%' IDENTIFIED WITH mysql_native_password BY 'password';
Query OK, 0 rows affected (0.03 sec)

mysql> |
```

Now we need to give this user permission over the `example_database` database:

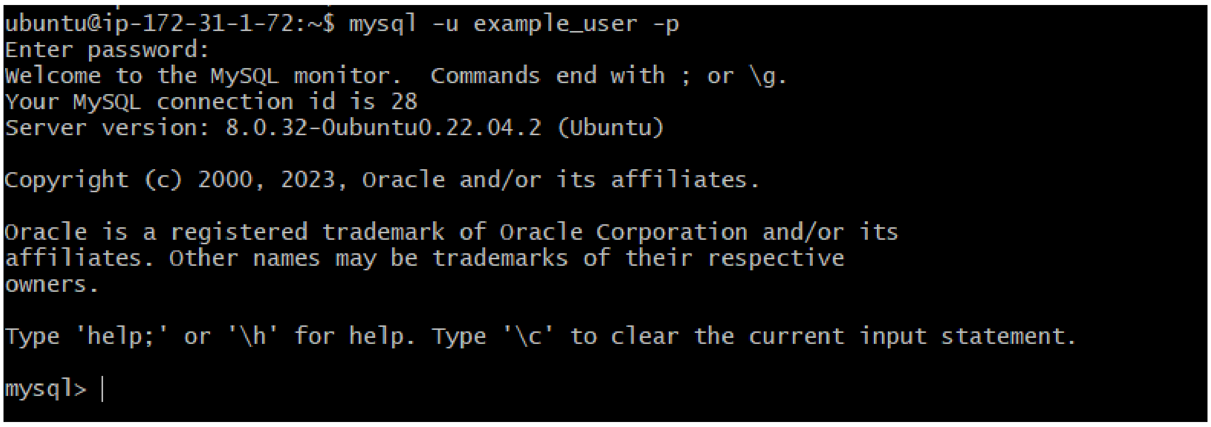
- `mysql> GRANT ALL ON example_database.* TO 'example_user'@'%';`



```
mysql> GRANT ALL ON example_database.* TO 'example_user'@'%';
Query OK, 0 rows affected (0.06 sec)

mysql> |
```

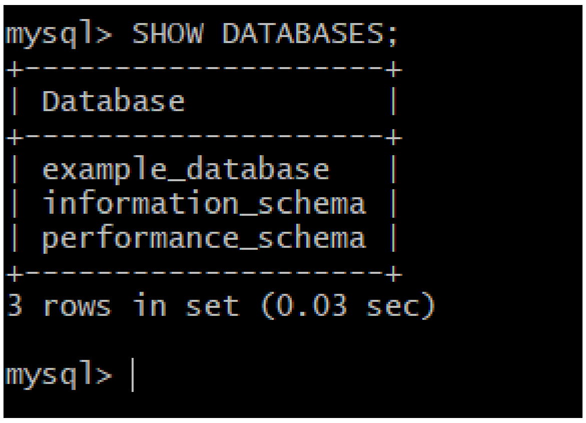
This will give the **example_user** user full privileges over the **example_database** database, while preventing this user from creating or modifying other databases on your server.

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-172-31-1-72:~$ mysql -u example_user -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 8.0.32-0ubuntu0.22.04.2 (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> |
```

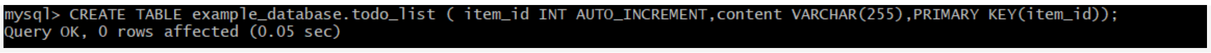
  Confirm that you have access to the `example_database` database:
  - `SHOW DATABASES;`



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| example_database   |
| information_schema |
| performance_schema |
+--------------------+
3 rows in set (0.03 sec)

mysql> |
```

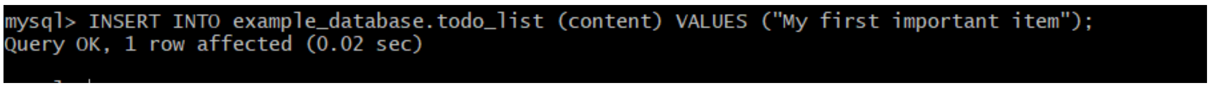
Create a test table named **todo_list**. From the MySQL console, run the following statement:

- CREATE TABLE example_database.todo_list (item_id INT AUTO_INCREMENT,content VARCHAR(255),PRIMARY KEY(item_id));



```
mysql> CREATE TABLE example_database.todo_list ( item_id INT AUTO_INCREMENT,content VARCHAR(255),PRIMARY KEY(item_id));
Query OK, 0 rows affected (0.05 sec)
```

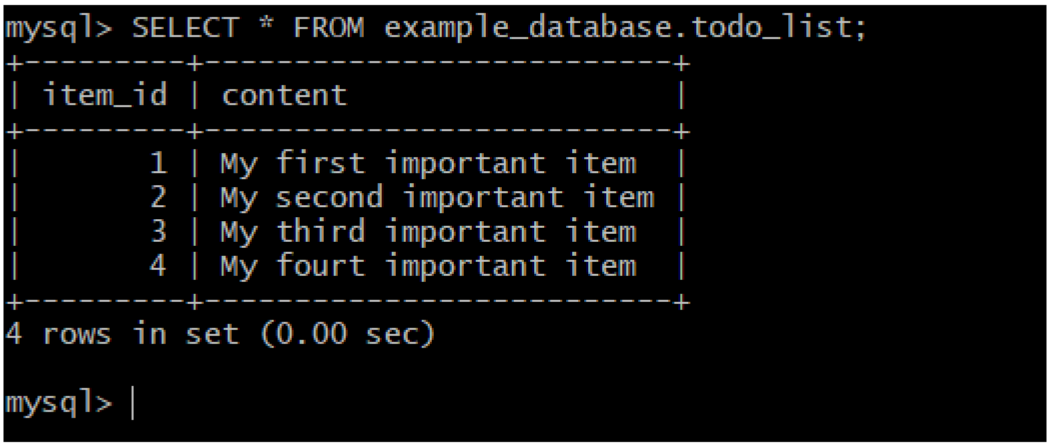
We insert a few rows of content in the test table. You might want to repeat the next command a few times, using different VALUES:



```
mysql> INSERT INTO example_database.todo_list (content) VALUES ("My first important item");
Query OK, 1 row affected (0.02 sec)
```

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

- `SELECT * FROM example_database.todo_list;`



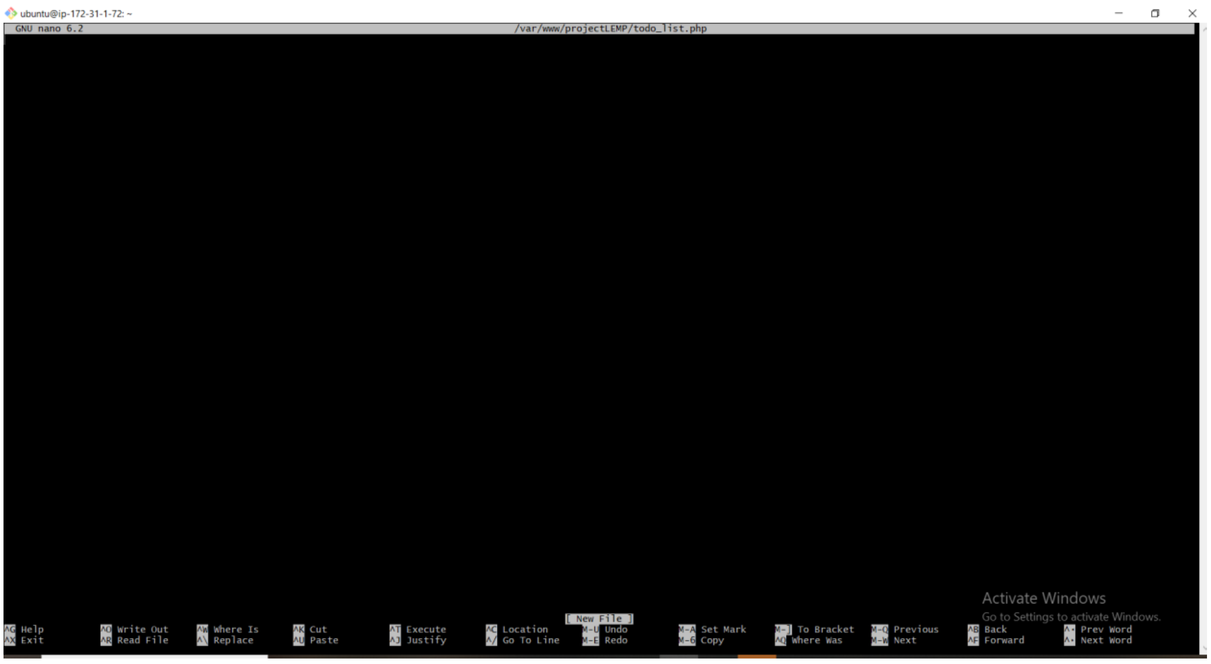
```
mysql> SELECT * FROM example_database.todo_list;
+---------+--------------------------+
| item_id | content                  |
+---------+--------------------------+
|       1 | My first important item  |
|       2 | My second important item |
|       3 | My third important item  |
|       4 | My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

mysql> |
```

After confirming that the valid data in your test table, you can exit the MySQL console:

- `mysql> exit`

Now we have to create a PHP script that will connect to MySQL and query for your content. Create a new PHP file in your custom web root directory using your preferred editor. We'll use nano for that:

- `nano /var/www/projectLEMP/todo_list.php`



```
ubuntu@ip-172-31-1-72: ~
GNU nano 6.2                    /var/www/projectLEMP/todo_list.php

                                [ New File ]
```

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:

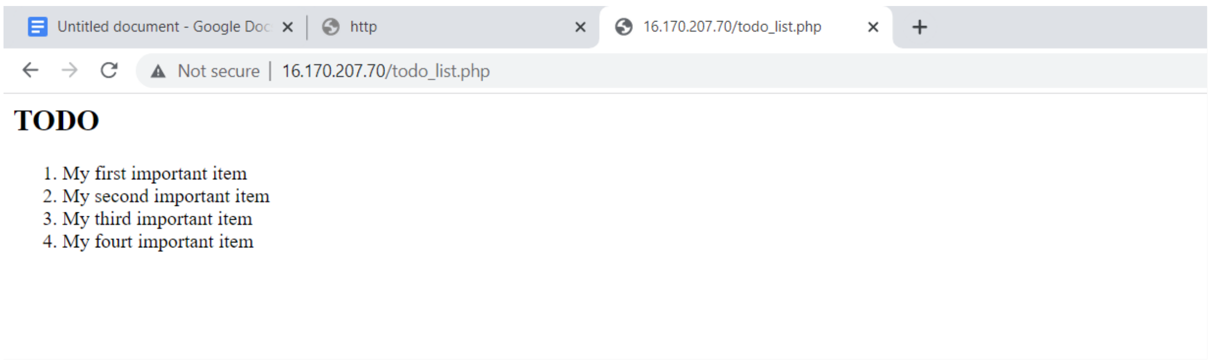
```
<?php

$user = "example_user";

$password = "password";

$database = "example_database";

$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();
}
```

Save and close the file when you are done editing.

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:`

- `http://<Public_domain_or_IP>/todo_list.php`


Untitled document - Google Doc
http
16.170.207.70/todo_list.php
Not secure | 16.170.207.70/todo_list.php

TODO

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