Lab Cycle: 3
Experiment No.: 2
Date: 7-06-2022

Aim : Illustrate the steps involved in installing the LAMP Stack on a Linux machine. Deploy phpMyadmin.

Steps to install Lamp Stack

A. Steps to installing apache2 server, mariadb and php.

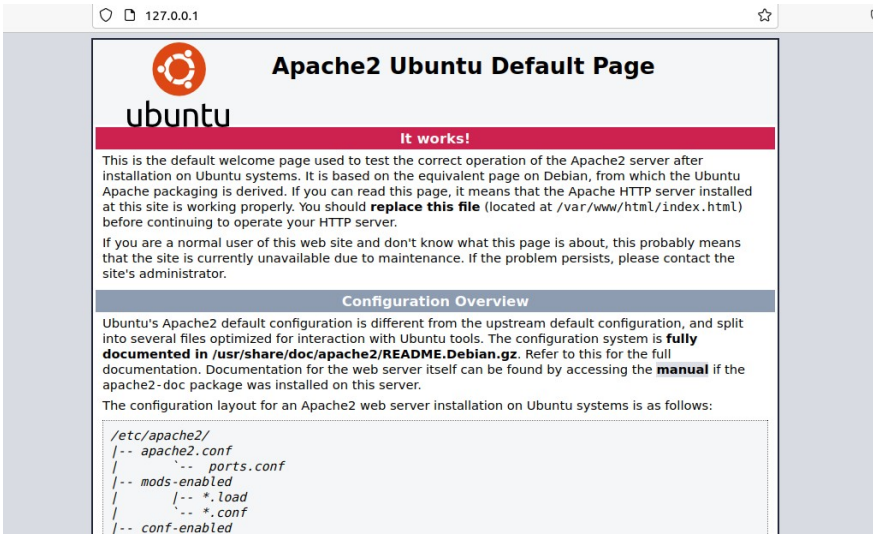
1. Initially update the repositories information
   sudo apt update

2. Install apache2 Web Server
   sudo apt install apache2

```
user@user-VirtualBox:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
0 upgraded, 9 newly installed, 0 to remove and 67 not upgraded.
Need to get 1,820 kB of archives.
After this operation, 7,945 kB of additional disk space will be used.
```

```
user@user-VirtualBox:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Thu 2022-06-09 13:58:36 IST; 1min 15s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 3419 (apache2)
      Tasks: 55 (limit: 5854)
     Memory: 4.7M
     CGroup: /system.slice/apache2.service
             ├─3419 /usr/sbin/apache2 -k start
             ├─3421 /usr/sbin/apache2 -k start
             └─3422 /usr/sbin/apache2 -k start

Jun 09 13:58:36 user-VirtualBox systemd[1]: Starting The Apache HTTP Server...
Jun 09 13:58:36 user-VirtualBox apachectl[3418]: AH00558: apache2: Could not re>
Jun 09 13:58:36 user-VirtualBox systemd[1]: Started The Apache HTTP Server.
lines 1-15/15 (END)
```

3. Check 127.0.0.1 from the browser, we can see the start page of Apache.



4. Install Mariadb Server and Client

sudo apt install mariadb-server mariadb-client

```
user@user-VirtualBox:~$ sudo apt install mariadb-server mariadb-client
[sudo] password for user:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
  libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libfcgi-perl
  libhtml-template-perl libreadline5 libsigsegv2 libsnappy1v5
  libterm-readkey-perl mariadb-client-10.3 mariadb-client-core-10.3
  mariadb-common mariadb-server-10.3 mariadb-server-core-10.3 socat
Suggested packages:
  gawk-doc libclone-perl libmldbm-perl libnet-daemon-perl
  libsql-statement-perl libipc-sharedcache-perl mailx mariadb-test tinyca
The following NEW packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
  libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libfcgi-perl
  libhtml-template-perl libreadline5 libsigsegv2 libsnappy1v5
  libterm-readkey-perl mariadb-client mariadb-client-10.3
  mariadb-client-core-10.3 mariadb-common mariadb-server mariadb-server-10.3
  mariadb-server-core-10.3 socat
0 upgraded, 22 newly installed, 0 to remove and 67 not upgraded.
Need to get 20.2 MB of archives.
After this operation, 167 MB of additional disk space will be used.
```

5. Make Mariadb Secure by setting password, disabling anonymous users etc.

sudo mysql_secure_installation

```
user@user-VirtualBox:~$ sudo mysql_secure_installation
[sudo] password for user:

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.
```

```
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!
```

6. Install PHP and commonly used packages

sudo apt install php libapache2-mod-php php-opcache php-cli php-gd php-curl php-mysql

```
user@user-VirtualBox:~$ sudo apt install php libapache2-mod-php php-opcache php-cli php-gd php
-curl php-mysql
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'php7.4-opcache' instead of 'php-opcache'
The following additional packages will be installed:
```

B. Steps to install phpMyAdmin

1. First login to mysql.

```
user@user-VirtualBox:~$ sudo mysql -uroot
[sudo] password for user:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.3.34-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]>
```

2. Create user ‘phpmyadmin’ and grant privileges to phpmyadmin.

CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY 'phpmyadminpassword';
GRANT ALL PRIVILEGES ON *.* TO 'phpmyadmin'@'localhost';
FLUSH PRIVILEGES;

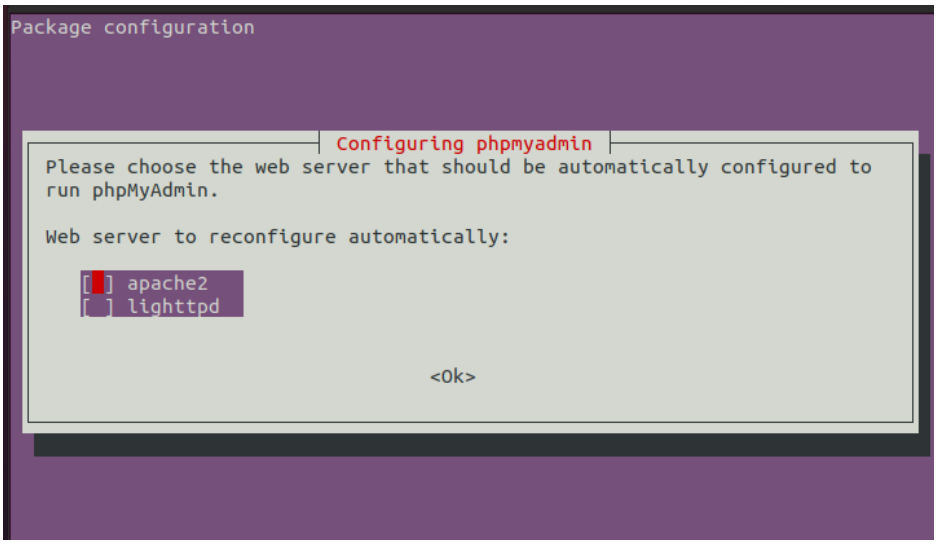
```
MariaDB [(none)]> CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY 'phpmyadminpassword';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'phpmyadmin'@'localhost';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> exit;
Bye
```

3. Install phpmyadmin package

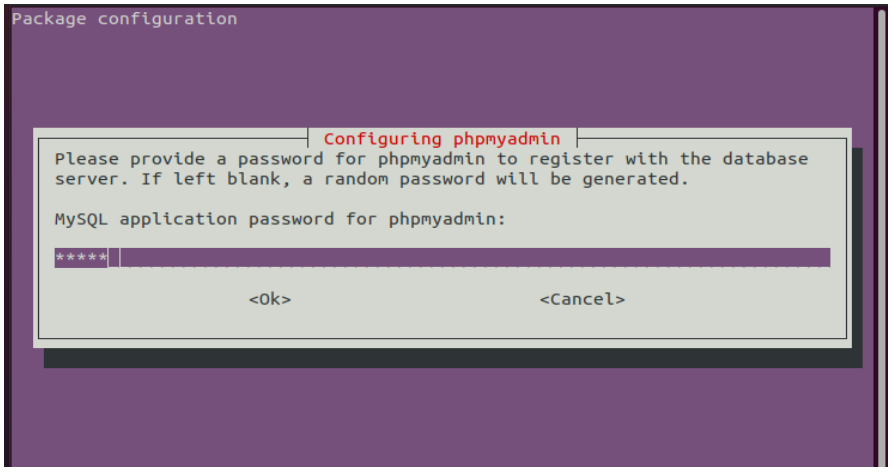
sudo apt install phpmyadmin php-mbstring php-zip php-json

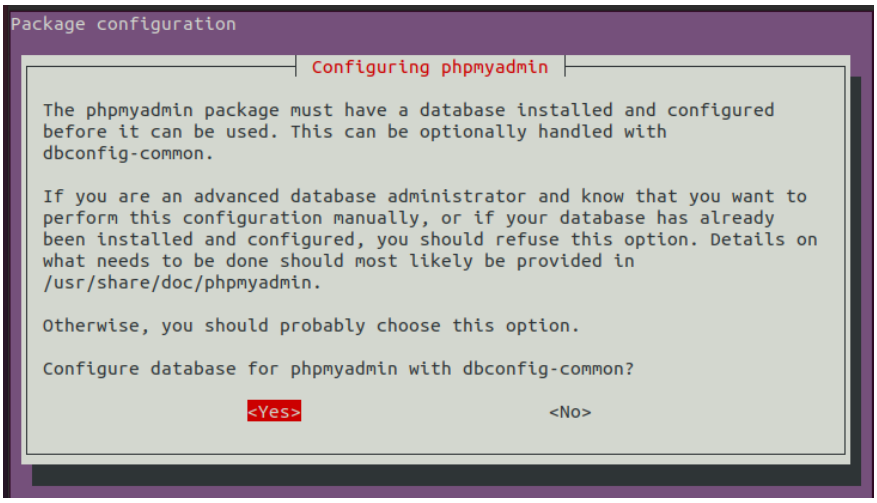
```
user@user-VirtualBox:~$ sudo apt install phpmyadmin php-mbstring php-zip php-json
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  dbconfig-common dbconfig-mysql icc-profiles-free javascript-common libjs-jquery
  libjs-openlayers libjs-sphinxdoc libjs-underscore libonig5 libzip5 php-bz2
  php-google-recaptcha php-phpmyadmin-motranslator php-phpmyadmin-shapefile
  php-phpmyadmin-sql-parser php-phpseclib php-psr-cache php-psr-container php-psr-log
  php-symfony-cache php-symfony-cache-contracts php-symfony-expression-language
  php-symfony-service-contracts php-symfony-var-exporter php-tcpdf php-twig
  php-twig-extensions php-xml php7.4-bz2 php7.4-mbstring php7.4-xml php7.4-zip
Suggested packages:
```

4. Next installation prompt to the automatic setup of the webserver to be used alongside phpMyAdmin. The choice is Apache2 .



5. Next step is the configuration of a phpMyAdmin database. Select db-configuration and set password. Choose yes to enable the database configuration steps.

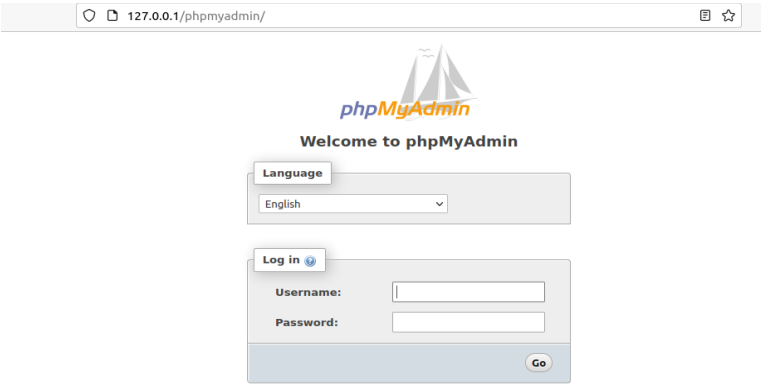




6. Reload the apache2 server.

   sudo systemctl restart apache2

7. Access phpmyadmin from browser using ‘127.0.0.1/phpmyadmin’.



8. Login to phpMyAdmin by using login credentials.

Lab Cycle: 3
Experiment No.: 3
Date: 14-06-2022

Aim : Create a Docker container of ubuntu:20.04. The container should be pre installed with nano. Use Dockerfile and build, create and start commands.

1. Install Docker Engine

```
ubuntu3@ubuntu3-VirtualBox:~$ sudo apt install docker.io
[sudo] password for ubuntu3:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd git git-man liberror-perl pigz runc ubuntu-fan
Suggested packages:
```

2.Create a Dockerfile with the following contents

```
  GNU nano 4.8
FROM ubuntu:20.04
RUN apt-get update
RUN apt -y install nano
```

3. Build the image with a suitable name.

```
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker build -t myubuntu .
Sending build context to Docker daemon  535.7MB
Step 1/3 : FROM ubuntu:20.04
20.04: Pulling from library/ubuntu
d7bfe07ed847: Pull complete
Digest: sha256:fd92c36d3cb9b1d027c4d2a72c6bf0125da82425fc2ca37c414d4f010180dc19
Status: Downloaded newer image for ubuntu:20.04
 ---> 20fffa419e3a
Step 2/3 : RUN apt-get update
 ---> Running in ba0624951a11
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [880 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [1286 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [27.5 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [1931 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [1275 kB]
```

4. Create the container with a suitable name then start and attach to the container

```
Successfully tagged myubuntu:latest
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker create --name container -i -t myubuntu
8baf59d8a3acc8de807fd33db707897d38e5d0977cbac5a6599385828f60e8aa
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker start -i -a container
```

5. Check whether nano is installed in the container, by giving the command nano.

```
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker start -i -a container
root@8baf59d8a3ac:/# nano
```