## **Step 1 — Installing and configuring Apache**

* sudo apt update
* Sudo apt upgrade
* sudo apt install apache2
* sudo ufw app list
* sudo ufw allow 'Apache Full'
* sudo ufw status
* Sudo ufw allow ssh
* Sudo ufw allow Openssh
* Sudo ufw allow 22

After the above steps, go to the /etc/apache2/sites-available.

Copy 000-default.conf to project.conf

Then run below commands

* a2ensite project.conf
* a2dissite 000-default.conf
* Systemctl restart apache2
* a2enmod rewrite

**Below is the sample configuration file of apache .conf**

<VirtualHost \*:80>

# The ServerName directive sets the request scheme, hostname and port that

# the server uses it to identify itself. This is used when creating

# redirection URLs. In the context of virtual hosts, the ServerName

# specifies what hostname must appear in the request's Host: header to

# match this virtual host. For the default virtual host (this file) this

# value is not decisive as it is used as a last resort host regardless.

# However, you must set it for any further virtual host explicitly.

ServerName www.staging.employmentcare.io

ServerAdmin webmaster@localhost

DocumentRoot /var/www/html/code/employment\_care/public

<Directory /var/www/html/code/employment\_care/public>

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,

# error, crit, alert, emerg.

# It is also possible to configure the loglevel for particular

# modules, e.g.

#LogLevel info ssl:warn

ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

# For most configuration files from conf-available/, which are

# enabled or disabled at a global level, it is possible to

# include a line for only one particular virtual host. For example the

# following line enables the CGI configuration for this host only

# after it has been globally disabled with "a2disconf".

#Include conf-available/serve-cgi-bin.conf

RewriteEngine on

RewriteCond %{SERVER\_NAME} =staging.employmentcare.io

RewriteRule ^ https://%{SERVER\_NAME}%{REQUEST\_URI} [END,NE,R=permanent]

</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

If we need to create **.htaccess file** from our side then the below is the code which we will write in PHP code directory after creating **.htaccess** file :

RewriteEngine On

RewriteRule .\* - [E=HTTP\_AUTHORIZATION:%{HTTP:Authorization}]

RewriteBase /

RewriteRule ^index\.html$ - [L]

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . /index.html [L]

after creating the **.htaccess** file in the project folder then edit the **.conf** file for override , like we already did in the above configuration file, this is just to be aware.

<Directory /var/www/>

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

then,

The below is the process to redirect links as a proxy redirection which we write in **.conf** file of apache:

<VirtualHost \*:80>

ProxyPreserveHost On

ProxyPass / http://127.0.0.1:5000/

ProxyPassReverse / http://127.0.0.1:5000/

</VirtualHost>

Then we usually run below commands after exiting the above file:

sudo en2mod proxy

sudo a2enmod proxy\_http

sudo systemctl restart apache2

## **Step 2 — Installing MySQL**

* sudo apt update
* sudo apt install mysql-server
* sudo systemctl start mysql.service

# **How to Change MySQL Root Password in Ubuntu 20.04**

* mysql --[version](https://dev.mysql.com/doc/refman/%35%2E%31/en/information-functions.html)
* sudo systemctl stop mysql.service
* sudo systemctl [status](https://search.oracle.com/search/search?group=MySQL&q=STATUS) mysql.service
* sudo systemctl set-environment MYSQLD\_OPTS="--skip-networking --skip-grant-tables"
* sudo systemctl [start](https://search.oracle.com/search/search?group=MySQL&q=START) mysql.service
* sudo systemctl [status](https://search.oracle.com/search/search?group=MySQL&q=STATUS) mysql.service
* sudo mysql -u root

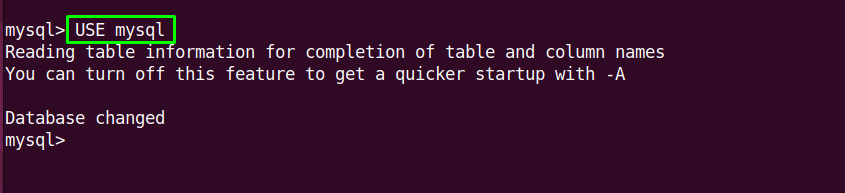
**Step 7: Alter the root password**

Now, flush the privileges first:

mysql> flush [privileges](https://search.oracle.com/search/search?group=MySQL&q=PRIVILEGES);

Next step is below

mysql> [USE](https://search.oracle.com/search/search?group=MySQL&q=USE) mysql



mysql> [ALTER](https://search.oracle.com/search/search?group=MySQL&q=ALTER) [USER](https://dev.mysql.com/doc/refman/%35%2E%31/en/information-functions.html) 'root'@'localhost' IDENTIFIED BY 'the-new-password';

mysql> quit;

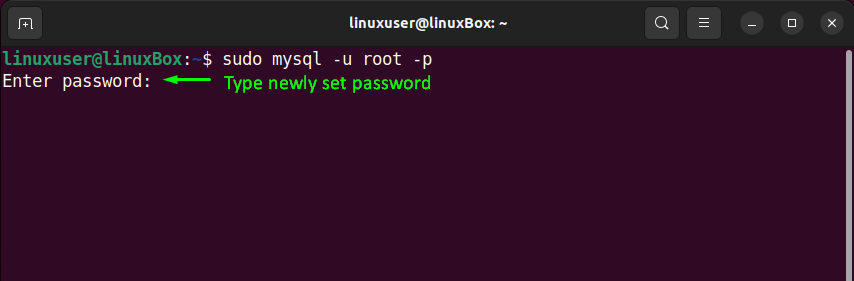
sudo systemctl unset-environment MYSQLD\_OPTS

sudo systemctl revert mysql

sudo killall -u mysql

sudo systemctl restart mysql.service

sudo mysql -u root -p



# **How To Create a New User and Grant Permissions in MySQL**

* sudo mysql
* mysql -u root -p
* CREATE USER 'admin'@'localhost' IDENTIFIED BY 'password';
* ALTER USER 'admin@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'password';
* GRANT ALL PRIVILEGES ON \*.\* TO 'admin'@'localhost' WITH GRANT OPTION;
* FLUSH PRIVILEGES;

If you need to revoke a permission, the structure is almost identical to granting it:

REVOKE type\_of\_permission ON database\_name.table\_name FROM 'username'@'host';

1. Note that when revoking permissions, the syntax requires that you use FROM, instead of TO which you used when granting the permissions.

You can review a user’s current permissions by running the SHOW GRANTS command:

SHOW GRANTS FOR 'username'@'host';

1. Just as you can delete databases with DROP, you can use DROP to delete a user:

DROP USER 'username'@'localhost';

After creating your MySQL user and granting them privileges, you can exit the MySQL client:

* exit

To login again with new user:

* mysql -u sammy -p

## **Step 3 — Installing PHP 7.4**

**To install PHP 7.4 and its packages in Ubuntu**

sudo apt update

sudo apt install software-properties-common apt-transport-https

sudo add-apt-repository ppa:ondrej/php

sudo apt update

sudo apt install php7.4

sudo apt install apache2 php7.4 php7.4 php7.4-common libapache2-mod-php7.4 php7.4-cli

sudo apt install php7.4-common php7.4-mysql php7.4-xml php7.4-xmlrpc php7.4-curl php7.4-gd php7.4-imagick php7.4-cli php7.4-dev php7.4-imap php7.4-mbstring php7.4-opcache php7.4-soap php7.4-zip php7.4-intl -y php7.4-pdo

**To install PHP 8.1 and its packages in Ubuntu**

sudo apt update

sudo apt install -y -q php8.1-{cli,fpm,mysql,gd,soap,mbstring,bcmath,common,xml,curl,imagick}

sudo apt install -y -q unzip

sudo curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=composer

**To Install Composer on ubuntu**

sudo apt update

sudo apt install php-cli unzip

cd ~

curl -sS https://getcomposer.org/installer -o /tmp/composer-setup.php

HASH=`curl -sS <https://composer.github.io/installer.sig>`

php -r "if (hash\_file('SHA384', '/tmp/composer-setup.php') === '$HASH') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP\_EOL;"

Output should be like below

Installer verified

Next step

sudo php /tmp/composer-setup.php --install-dir=/usr/local/bin --filename=composer

To test your installation, run:

composer

Now installation is completed.

Now to run the code, we will go to the code directory and will run below commands.

* composer install
* composer dumpautoload
* composer update

copy the .env.example file to .env in project directory where code is placed :

* copy .env.example .env
* php artisan cache:clear

generate application key :

* php artisan key:generate
* Edit the .env file to add your environment values for database : DB\_HOST=127.0.0.1

Migrate and seed the database :

* php artisan migrate:fresh --seed
* php artisan optimize:clear (after editing .env file)

The below command is for laraship only

* New: Laraship supports now Passport for API, you need to run the command below to install Passport

**php artisan passport:install**

**To remove PHP version from Ubuntu**

sudo apt-get purge php7.\*

sudo apt-get autoclean

sudo apt-get autoremove

## How To Clear Memory Cache And Buffer On Linux

Command to free pagecache, dentries and inodes in cache memory in Linux.

sync; echo 3 > /proc/sys/vm/drop\_caches

Second option to clear memory cache.

Command to free dentries and inodes

sync; echo 2 > /proc/sys/vm/drop\_caches

Third option to clear memory cache on Linux.

Run the following command to free pagecache only:

sync; echo 1 > /proc/sys/vm/drop\_caches

## How To Schedule Cron To Flush Cache Regularly

Run the following command to schedule cron to flush cache regularly. It makes your job easier and reduces the work pressure. Use ‘crontab -e’ command to edit cron on your Linux.

crontab -l

0 \* \* \* \* sync; echo 3 > /proc/sys/vm/drop\_caches

**To Install Certbot SSL on ubuntu with apache2**

**sudo apt update**

**sudo apt install snapd**

**sudo snap install core; sudo snap refresh core**

**sudo apt-get remove certbot**

**sudo snap install --classic certbot**

**sudo ln -s /snap/bin/certbot /usr/bin/certbot**

**sudo certbot --apache**