What is NGINX?

NGINX (pronounced "engine X") is a popular open-source web server software that is widely used as a reverse proxy server, load balancer, and HTTP cache. It was created by Igor Sysoev in 2004 and has gained significant popularity due to its performance, scalability, and flexibility.

Originally designed to handle high traffic websites, NGINX has evolved into a full-featured web server that can handle various tasks in modern web application architectures. Some of its key features include:

Web Server: NGINX can serve static content such as HTML, CSS, JavaScript files, and images. It efficiently handles concurrent connections and delivers content quickly.

Reverse Proxy: NGINX can act as a reverse proxy server, sitting between clients and backend servers. It distributes incoming requests to multiple backend servers and can perform load balancing to ensure efficient resource utilization.

Load Balancing: NGINX can distribute incoming requests across multiple servers to achieve high availability and scalability. It can evenly distribute the load based on various algorithms like round-robin, IP hash, least connections, etc.

Caching: NGINX includes a built-in caching mechanism that can store frequently accessed content in memory. This helps to reduce the load on backend servers and improve response times for subsequent requests.

SSL/TLS Termination: NGINX can handle SSL/TLS encryption and decryption, acting as a termination point for secure connections. This offloads the cryptographic processing from backend servers, improving their performance.

URL Rewriting: NGINX provides flexible URL rewriting capabilities, allowing you to modify and redirect URLs based on predefined rules. This is useful for creating user-friendly URLs or implementing redirections.

High Performance: NGINX is known for its high performance and efficiency in handling concurrent connections. It uses an asynchronous, event-driven architecture that allows it to handle a large number of simultaneous requests efficiently.

NGINX is widely used by many high-traffic websites, including popular platforms like Netflix, Airbnb, Pinterest, and WordPress.com. It is available for various operating systems and has a rich ecosystem of extensions and modules, making it highly customizable and adaptable to different use cases.

How to install NGINX on UBUNTU 22.04?

```
sudo su
apt update
apt install nginx
```

Once package are downloaded open your browser and type. **http://localhost** You will see the default page of nginx.

This page is coming from the below path:

```
cd /var/www/html
```

Useful commands for nginx:

```
systemctl status nginx
sudo systemctl stop nginx
sudo systemctl start nginx
sudo systemctl restart nginx
```

If you are only making configuration changes Nginx can often reload without dropping connections.

```
sudo systemctl reload nginx
```

Set up the Virtual host (DOMAIN)?

Create a new site folder under */var/www/html*

Let’s keep the name of the folder ***test.work*** and then make a ***index.html*** file under this.

Change the permission of the ***test.work*** folder

```
chmod -R 777 test.work
```

Now go to below path:

```
cd /etc/nginx/sites-available
```

Make a new file with your domain name.

```
nano test.work
```

Add the below code to the file:

```
server {
    listen 80;
    listen [::]:80;
    root /var/www/html/test.work;
    index index.html index.htm index.nginx-debian.html;
    server_name test.work www.test.work;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

You can do so by recreating the symbolic link

```
sudo ln -s /etc/nginx/sites-available/test.work /etc/nginx/sites-enabled/
```

To avoid a possible hash bucket memory problem that can arise from adding additional server names, it is necessary to adjust a single value in the /etc/nginx/nginx.conf file. Open the file: Find the server_names_hash_bucket_size directive and remove the # symbol to uncomment the line.

```
sudo nano /etc/nginx/nginx.conf
```

Test for any kind of syntax error on Nginx configuration file.

```
sudo nginx -t
```

Now add a entry to your host file.

```
nano /etc/hosts
```

Add the below code:

```
127.0.1.1 test.work
```

Now open the ***test.work*** URL on the browser and you will see code will run from */var/www/html/test.work.*

Important information:

- /var/log/nginx/access.log: Every request to your web server is recorded in this log file unless Nginx is configured to do otherwise.
- /var/log/nginx/error.log: Any Nginx errors will be recorded in this log.

- /etc/nginx: The Nginx configuration directory. All of the Nginx configuration files reside here.
- /etc/nginx/nginx.conf: The main Nginx configuration file. This can be modified to make changes to the Nginx global configuration.
- /etc/nginx/sites-available/: The directory where per-site server blocks can be stored. Nginx will not use the configuration files found in this directory unless they are linked to the sites-enabled directory. Typically, all server block configuration is done in this directory, and then enabled by linking to the other directory.
- /etc/nginx/sites-enabled/: The directory where enabled per-site server blocks are stored. Typically, these are created by linking to configuration files found in the sites-available directory.
- /etc/nginx/snippets: This directory contains configuration fragments that can be included elsewhere in the Nginx configuration. Potentially repeatable configuration segments are good candidates for refactoring into snippets.

Server Logs:

- /var/log/nginx/access.log: Every request to your web server is recorded in this log file unless Nginx is configured to do otherwise.
- /var/log/nginx/error.log: Any Nginx errors will be recorded in this log.

How to Install Mysql ?

Install mysql server

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

Create user for mysql:

```
sudo mysql -u root
USE mysql;
UPDATE user SET plugin='mysql_native_password' WHERE User='root';
FLUSH PRIVILEGES;
exit;
```

Restart:

```
sudo systemctl restart mysql.service
```

Create password for Mysql

```
sudo mysql_secure_installation
```

Now, Login to Mysql by username and password

```
mysql -u root -p
```

How to Install PHP?

Install PHP:

sudo apt install php8.1-fpm php-mysql

check php version:

php -v

Here is the path where PHP is installed:

cd /etc/php

To make the PHP working under your domain you need to update the below code under your domain configuration file.

nano /etc/nginx/sites-available/test.work

```
server {
   listen 80;
   server_name test.work www.test.work;
   root /var/www/html/test.work;
   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;
   }
}
```

systemctl restart nginx

Make **index.php** file under ***test.work*** and add the below code to make sure PHP is working.

<?php Phpinfo(); ?>

Here's what each of these directives and location blocks do:

- listen — Defines what port Nginx will listen on. In this case, it will listen on port 80, the default port for HTTP.
- root — Defines the document root where the files served by this website are stored.
- index — Defines in which order Nginx will prioritize index files for this website. It is a common practice to list index.html files with higher precedence than index.php files to allow for quickly setting up a maintenance landing page in PHP applications. You can adjust these settings to better suit your application needs.
- server_name — Defines which domain names and/or IP addresses this server block should respond for. **Point this directive to your server's domain name or public IP address.**
- location / — The first location block includes a try_files directive, which checks for the existence of files or directories matching a URL request. If Nginx cannot find the appropriate resource, it will return a 404 error.
- location ~ \.php$ — This location block handles the actual PHP processing by pointing Nginx to the fastcgi-php.conf configuration file and the php8.1-fpm.sock file, which declares what socket is associated with php8.1-fpm.
- location ~ /\.ht — The last location block deals with .htaccess files, which Nginx does not process. By adding the deny all directive, if any .htaccess files happen to find their way into the document root, they will not be served to visitors.

How to Install phpMyAdmin?

How to install the phpmyadmin:

```
sudo apt update
```

First of all you need to disable the password for Mysql:

```
mysql -u root -p
UNINSTALL COMPONENT "file://component_validate_password";
exit
```

Command to install the phpmyadmin:

```
apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl
```

Now login to Mysql and re-enable the password:

```
INSTALL COMPONENT "file://component_validate_password";
```

Create a symbolic link from the installation

```
sudo ln -s /usr/share/phpmyadmin /var/www/test.work/phpmyadmin
```

You can now access the phpmyadmin with below URL:

http://test.work/phpmyadmin