1.Install nginx and host a simple index.html with message “hellonginx”

First of all, we install nginx as follows

```
samana@ubuntu:~$ sudo apt install nginx
[sudo] password for samana:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter
  libnginx-mod-mail libnginx-mod-stream nginx-common nginx-core
Suggested packages:
  fcgiwrap nginx-doc
The following NEW packages will be installed:
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter
  libnginx-mod-mail libnginx-mod-stream nginx nginx-common nginx-core
0 upgraded, 7 newly installed, 0 to remove and 5 not upgraded.
Need to get 0 B/603 kB of archives.
After this operation, 2,134 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Preconfiguring packages ...
```

Nginx was found to be running and it’s version was 1/18.0.

```
samana@ubuntu:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
samana@ubuntu:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset
     Active: active (running) since Wed 2021-11-17 09:02:35 +0545; 3min 31s ago
       Docs: man:nginx(8)
   Main PID: 2198 (nginx)
      Tasks: 2 (limit: 2299)
     Memory: 1.6M
     CGroup: /system.slice/nginx.service
             ├─2198 nginx: master process /usr/sbin/nginx -g daemon on; master
             └─2199 nginx: worker process
```

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.

Now in order to display our required text Hello nginx we create a config file name app1.conf

```
  GNU nano 4.8                                app1.conf
server{
listen 80;
listen [::]:80;
root /var/www/html/app1;
index index.html;
server_name localhost;
location / {
try_files $uri $uri/ =404;
}
}
```

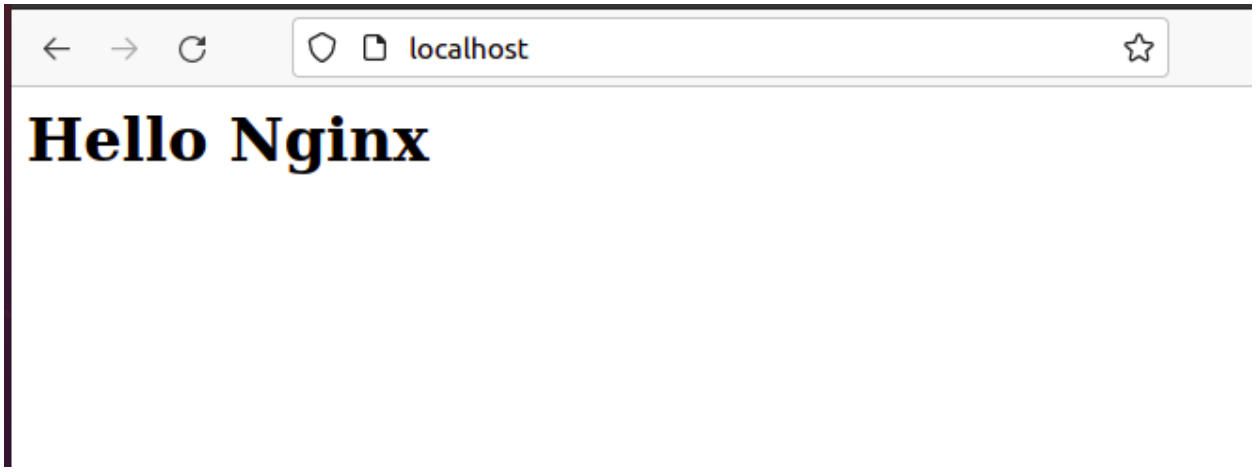
As shown above, we put the root location to /var/www/html/app1 where we have created a index file named index.html with the following contents:

```
  GNU nano 4.8                          index.html
<html>
        <head>
                <title> Hello nginx </title>
        </head>
        <body>
                <h1>Hello Nginx</h1>
        </body>
</html>
```

Now in order to make our html file hosted on nginx we create a symbolic link to sites-enabled folder from sites-available folder

```
samana@ubuntu:/etc/nginx/sites-available$ sudo ln -s /etc/nginx/sites-available
/app1.conf /etc/nginx/sites-enabled/
```

We restart the nginx server using sudo systemctl restart nginx and observe the following response in our browser.



2.What are nginx header security and its uses. And also implement in the test.conf file.

In nginx several security headers can be implemented which are intended to protect against various known attacks and also track the users or systems performing such attacks. Below are some headers and their use:

X-XSS-Protection :This header protects against cross site scripting attacks. When set to 0 this is of no use but it can be set to 1 to filter out scripts that may be potentially

malicious. Also, it can be set to block mode if any script or code is found being placed in the server.

X-Frame-Options:
This header protects against clickjacking/UI redressing attacks(when our application is framed by some other application's user interface that may trick users to perform unintended action)

X-Content-Type-Options: This header can be set to nosniff which instructs browsers to disable content or MIME sniffing

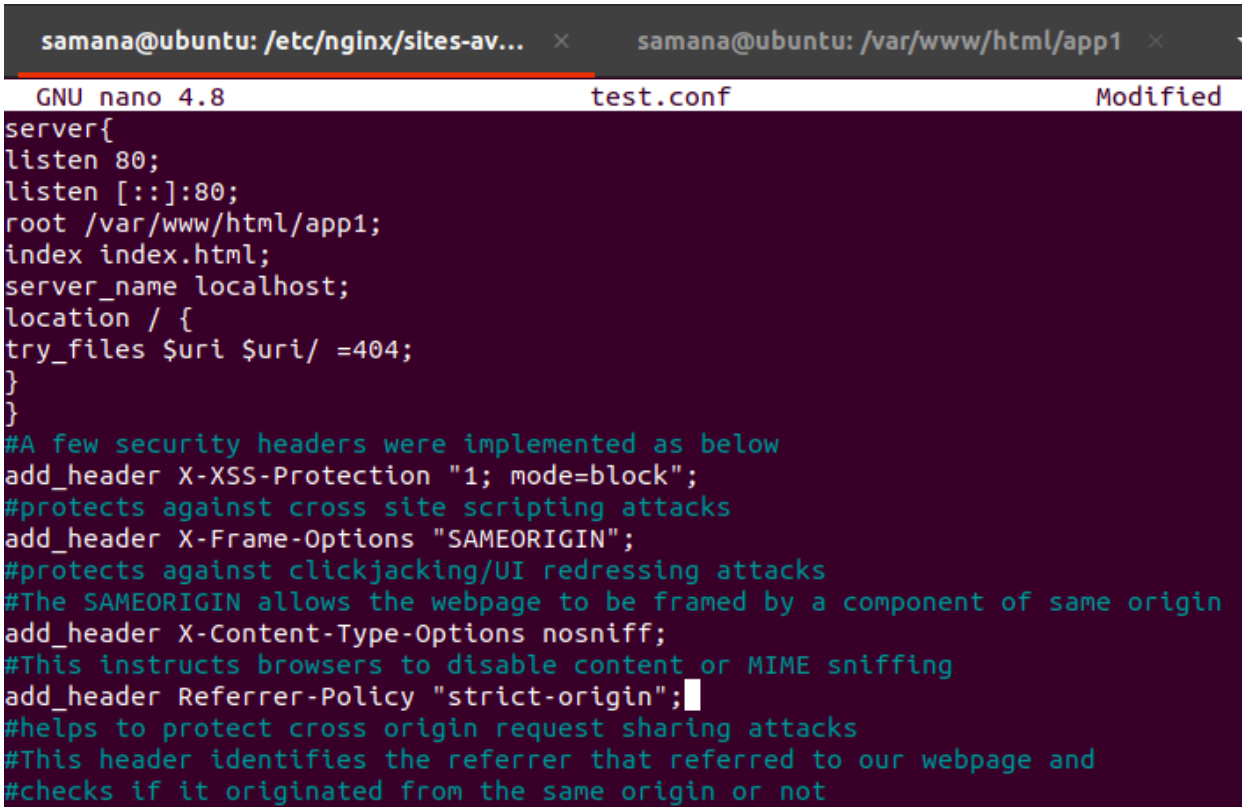
Referrer-Policy: This header helps to protect cross origin request sharing attacks It identifies the referrer that referred to our webpage and checks if it originated from the same origin or not

HTTP Strict Transport Security (HSTS): It tells the user's browser to use only https which prevents the communication from being captured.

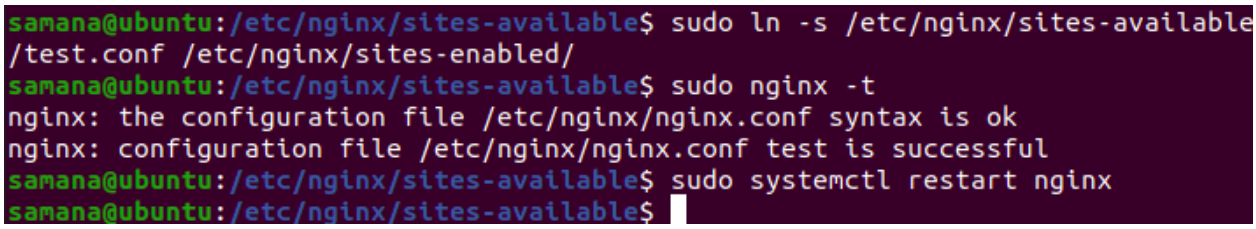
Content-Security-Policy(CSP): It is a newer header that is used to prevent from xss, clickjacking as well as few injection attacks.

Permissions-Policy: It is a new header that allows site to control which APIs or features can be used in the browser.

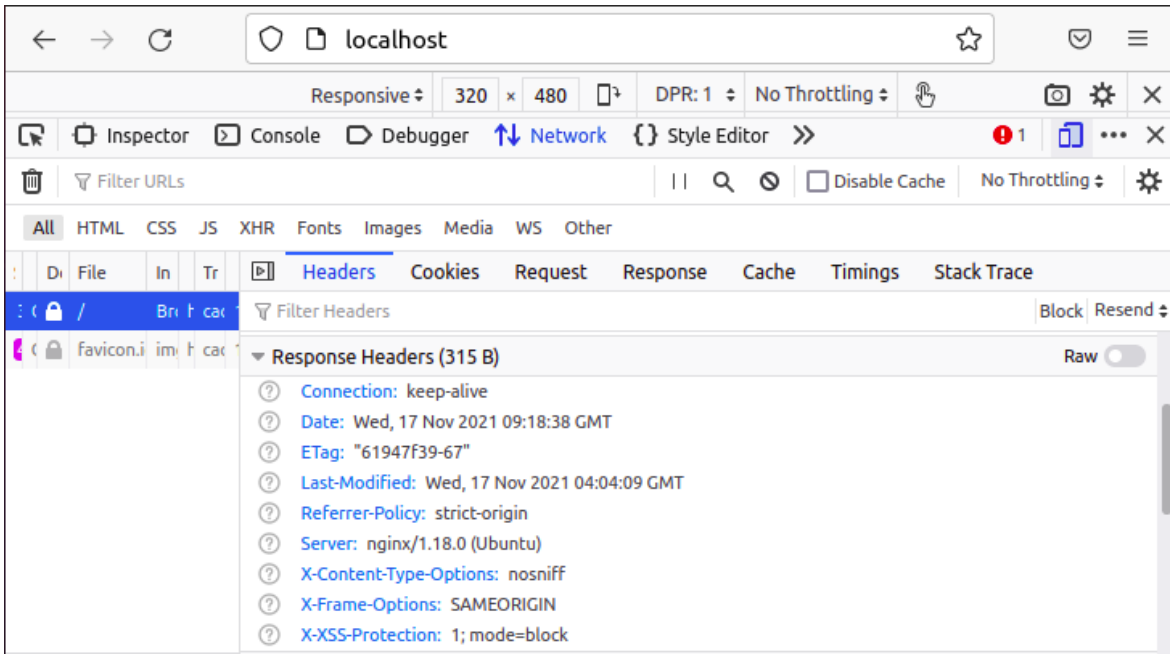
We implemented few security headers by creating a test.conf file as follows



Now we create a symbolic link for sites-enabled folder



After restarting the service we were able to see that the headers were successfully implemented.



3.Nginx Reverse proxy all http requests to nodejs api.

First of all we start the node api1.js application from pm2 which is running at port 6080.

```
samana@ubuntu:/var/www/nodeapp1$ ls
api1.js  package.json
samana@ubuntu:/var/www/nodeapp1$ sudo pm2 start api1.js
[sudo] password for samana:
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /var/www/nodeapp1/api1.js in fork_mode (1 instance)
[PM2] Done.
```

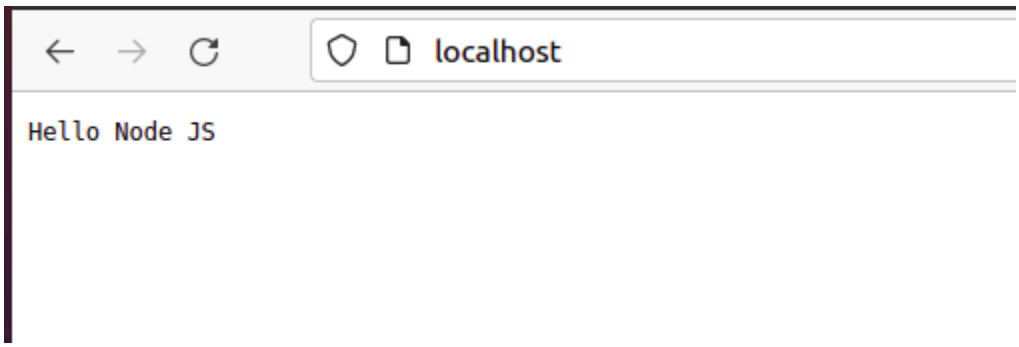
id	name	mode	↺	status	cpu	memory
0	api1	fork	0	online	0%	30.9mb

Then we create a config file named node_reverse_proxy.conf

```
  GNU nano 4.8                    node_reverse_proxy.conf
server{
listen 80;
server_name localhost;

location / {
        proxy_pass http://localhost:6080;
}
}
```

Hence all the http requests are proxied to node js api.



4.Create a test2.conf and listen on port 82 and to “ location /test/”with message “ test is successful”

First, we create a index.html file under the test directory under the location /var/www/html/app1

```
samana@ubuntu:/var/www/html/app1$ cd test/
samana@ubuntu:/var/www/html/app1/test$ sudo nano index.html
```

```
  GNU nano 4.8                          index.html
<html>
        <head>
                <title> Hello test </title>
        </head>
        <body>
                <h1>test is successful</h1>
        </body>
</html>
```

Now in sites-available directory we create a test2.conf file as follows:

```
samana@ubuntu:/etc/nginx/sites-available$ sudo nano test2.conf
[sudo] password for samana:
```

```
  GNU nano 4.8                          test2.conf
server{
listen 82;
listen [::]:80;
root /var/www/html/app1;
index index.html;
server_name localhost;
location /test/ {
index index.html;
}
location / {
try_files $uri $uri/ =404;
}
}
```

Now we create a symbolic link, run nginx test and restart nginx

```
samana@ubuntu:/etc/nginx/sites-available$ sudo ln -s /etc/nginx/sites-available
/test2.conf /etc/nginx/sites-enabled/
samana@ubuntu:/etc/nginx/sites-available$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
samana@ubuntu:/etc/nginx/sites-available$ sudo systemctl restart nginx
samana@ubuntu:/etc/nginx/sites-available$
```

Finally we can see that on pointing the url to /test/ location our desired text is displayed.

5.Reverse proxy all http traffic of port 82 to port 85.

For this we create a config file as follows

```
  GNU nano 4.8                          82_to_85.conf
server
{
        listen 85;
        server_name localhost;
        location / {
        proxy_pass http://localhost:82/;
}
}
```

Then we create symbolic link from the sites-available folder to sites-enabled folder and restart nginx after performing test

```
samana@ubuntu:/etc/nginx/sites-available$ sudo ln -s /etc/nginx/sites-available
/82_to_85.conf /etc/nginx/sites-enabled/
samana@ubuntu:/etc/nginx/sites-available$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
samana@ubuntu:/etc/nginx/sites-available$ sudo systemctl restart nginx
samana@ubuntu:/etc/nginx/sites-available$
```

Here the contents which were previously configured at port 82 were proxied to port 85.

Hello Nginx

test is successful

6.Install LEMP stack (avoid installing mysql) and open info.php onport 80 and print message info.php.

LEMP stack stands for Linux Nginx Mysql and Php. since we have a linux system,nginx we install php as follows

```
samana@ubuntu:/var/www/php$ sudo apt install php php-fpm
Reading package lists... Done
Building dependency tree
Reading state information... Done
php-fpm is already the newest version (2:7.4+75).
The following NEW packages will be installed:
  php php7.4
0 upgraded, 2 newly installed, 0 to remove and 13 not upgraded.
Need to get 12.0 kB of archives.
After this operation, 88.1 kB of additional disk space will be used.
Get:1 http://np.archive.ubuntu.com/ubuntu focal-updates/main amd64 php7.4 all 7
.4.3-4ubuntu2.7 [9,248 B]
Get:2 http://np.archive.ubuntu.com/ubuntu focal/main amd64 php all 2:7.4+75 [2,
712 B]
Fetched 12.0 kB in 1s (14.9 kB/s)
```

We can check the status and the php was found to be running.

```
samana@ubuntu:/etc/php/7.4/fpm$ sudo systemctl status php7.4-fpm.service
● php7.4-fpm.service - The PHP 7.4 FastCGI Process Manager
     Loaded: loaded (/lib/systemd/system/php7.4-fpm.service; enabled; vendor p>
     Active: active (running) since Wed 2021-11-17 18:17:55 +0545; 7s ago
       Docs: man:php-fpm7.4(8)
    Process: 17553 ExecStartPost=/usr/lib/php/php-fpm-socket-helper install /r>
   Main PID: 17550 (php-fpm7.4)
     Status: "Ready to handle connections"
      Tasks: 3 (limit: 2299)
     Memory: 6.7M
     CGroup: /system.slice/php7.4-fpm.service
             ├─17550 php-fpm: master process (/etc/php/7.4/fpm/php-fpm.conf)
             ├─17551 php-fpm: pool www
             └─17552 php-fpm: pool www

नवम्बर 17 18:17:55 ubuntu systemd[1]: Starting The PHP 7.4 FastCGI Process Mana>
नवम्बर 17 18:17:55 ubuntu systemd[1]: Started The PHP 7.4 FastCGI Process Manag>
lines 1-16/16 (END)
```

Now we created a file named info.php which had the following content in the following location

```
samana@ubuntu:/var/www$ cd php
samana@ubuntu:/var/www/php$ cat info.php
<?php
        phpinfo();
?>
samana@ubuntu:/var/www/php$
```

Now we created a php.conf file as given below inside the /etc/nginx/sites-available directory

```
 GNU nano 4.8                          php.conf
server {
        listen 80 ;
        listen [::]:80 ;

        root /var/www/php;
        index info.php;
        server_name localhost;

        location / {

                try_files $uri $uri/ =404;

        }

        # This location block handles the actual PHP processing by pointing Ng>

        location ~ \.php$ {

                include snippets/fastcgi-php.conf;

                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
```

After creating the symbolic link to folder sites-enabled, we restarted the nginx service We obtained our desired result as specified by the php.conf file which can be verified from the screenshot below: