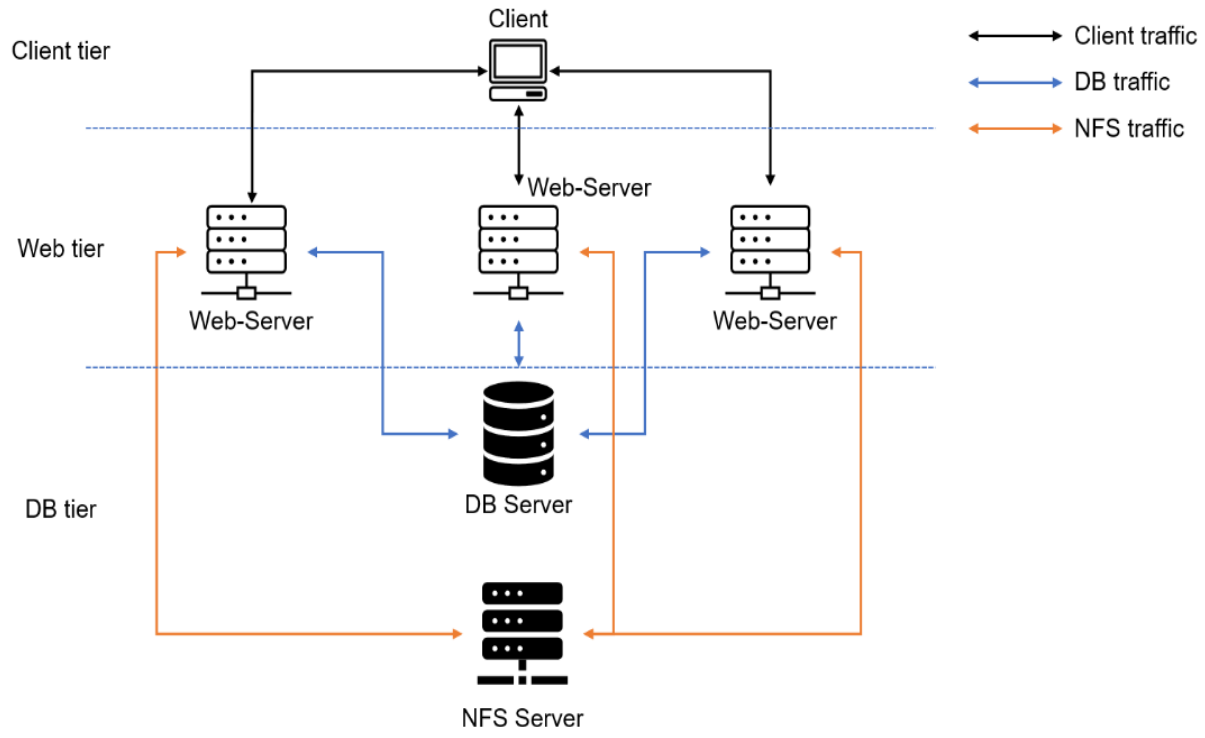
PROJECT 7: DEVOPS TOOLING WEBSITE SOLUTION

After completing Project 6, where I implemented a WordPress-based solution that is still lacking content and does not yet serve as a website or blog. I will need to increase the value of our products so that a DevOps team may use them for managing, creating, testing, deploying, and monitoring various projects daily. A single DevOps tooling solution, made up of Jenkins, Kubernetes, Jfrog Artifactory, Rancher, Grafana, Prometheus, and Kibana, will be shown on our Tooling website.

This project will give practical skills on how to launch a DevOps tools website from a GitHub repository. A MySQL database will be set up as an identity storage system, and a network file system will be set up as a centralised file storage solution. At the end of this implementation, we should have a 3-tier web application where the webservers share a common database and also access the same files using Network File System (NFS) as a shared file storage like the one in the diagram below.



The following components will be required to implement this solution:

Infrastructure: AWS
3 Webserver Linux: Red Hat Enterprise Linux 8
1 Database Server: Ubuntu 20.04 + MySQL
Storage Server: Red Hat Enterprise Linux 8 + NFS Server
Programming Language: PHP
Code Repository: GitHub

Create an EC2 instance and EBS volume

4 instances running on RHEL will be created for 1 NFS server and 3 Webervers and 1 instance running on Ubuntu for the Database Server. The volumes should have the same availability zone as the NFS server, with 10 GB and 3 Logical Volume Manager (LVM) for lv opt, lv apps, and lv logs which will be attached to the NFS Server. The exact steps to follow for this have been highlighted in Project 6.

Install the NFS Server

The *lsblk* command can be used to list all the blocks attached to this server. Next will be to create a partition on the disk by running.

```
[ec2-user@ip-172-31-10-140 ~]$ lsblk
NAME                      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
xvda                      202:0    0   10G  0 disk
├─xvda1                   202:1    0    1M  0 part
├─xvda2                   202:2    0  200M  0 part /boot/efi
├─xvda3                   202:3    0  500M  0 part /boot
└─xvda4                   202:4    0  9.3G  0 part /
xvdf                      202:80   0   10G  0 disk
└─xvdf1                   202:81   0   10G  0 part
  └─webdata--vg-lv--opt   253:2    0    9G  0 lvm
xvdg                      202:96   0   10G  0 disk
└─xvdg1                   202:97   0   10G  0 part
  └─webdata--vg-lv--logs  253:1    0    9G  0 lvm
xvdh                      202:112  0   10G  0 disk
└─xvdh1                   202:113  0   10G  0 part
  └─webdata--vg-lv--apps  253:0    0    9G  0 lvm
```

sudo gdisk /dev/xvdf
sudo gdisk /dev/xvdg

sudo gdisk /dev/xvdh

The output should appear as below where - p - prints the partition table, - n - adds a new partition, - w - writes the table to the disk and exit/save and -y- saves the changes. This process will need to be repeated across xvdg and xvdh.

```
[ec2-user@ip-172-31-33-186 ~]$ sudo gdisk /dev/xvdf
GPT fdisk (gdisk) version 1.0.7

Partition table scan:
  MBR: not present
  BSD: not present
  APM: not present
  GPT: not present

Creating new GPT entries in memory.

Command (? for help): p
Disk /dev/xvdf: 20971520 sectors, 10.0 GiB
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): E0820AF2-334E-46A4-858F-44706E216B78
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 20971486
Partitions will be aligned on 2048-sector boundaries
Total free space is 20971453 sectors (10.0 GiB)

Number  Start (sector)    End (sector)  Size       Code  Name

Command (? for help): n
Partition number (1-128, default 1): 1
First sector (34-20971486, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-20971486, default = 20971486) or {+-}size{KMGTP}:
Current type is 8300 (Linux filesystem)
Hex code or GUID (L to show codes, Enter = 8300): 8300
Changed type of partition to 'Linux filesystem'

Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/xvdf.
The operation has completed successfully.
```

Install logical volume manager and check for available partitions with *sudo lvmdiskscan*

```
[ec2-user@ip-172-31-33-186 ~]$ sudo yum install lvm2 -y
```

```
[ec2-user@ip-172-31-33-186 ~]$ sudo lvmdiskscan
  /dev/xvda2 [     200.00 MiB]
  /dev/xvda3 [     500.00 MiB]
  /dev/xvda4 [       9.31 GiB]
  /dev/xvdf1 [     <10.00 GiB]
  /dev/xvdg1 [     <10.00 GiB]
  /dev/xvdh1 [     <10.00 GiB]
  0 disks
  6 partitions
  0 LVM physical volume whole disks
  0 LVM physical volumes
```

Create physical volume by running the pvcreate command to mark each of the 3 disks as physical volumes (PVs) to be used by LVM with output as physical volume successfully created.

sudo pvcreate /dev/xvdf1

sudo pvcreate /dev/xvdg1

sudo pvcreate /dev/xvdh1

```
  Physical volume "/dev/xvdf1" successfully created.
  Creating devices file /etc/lvm/devices/system.devices
  Physical volume "/dev/xvdg1" successfully created.
  Physical volume "/dev/xvdh1" successfully created.
```

Create volume group

```
[ec2-user@ip-172-31-33-186 ~]$ sudo vgcreate webdata-vg /dev/xvdh1  /dev/xvdg1 /dev/xvdf1
```

Next, use *sudo vgs* to check for the created volume group

```
[ec2-user@ip-172-31-33-186 ~]$ sudo vgs
  VG         #PV #LV #SN Attr   VSize   VFree
  webdata-vg   3   0   0 wz--n- <29.99g <29.99g
```

Use lvcreate command to create 3 logical volumes: lv-apps, lv-logs, and lv-opt. lv-apps will be used to store data for the Website, lv-logs will be used to store data for logs and lv-opt will be used by Jenkins server in Project 8

sudo lvcreate -n lv-apps -L 9G webdata-vg
sudo lvcreate -n lv-logs -L 9G webdata-vg
*sudo lvcreate -n lv-opt -L 9G webdata-v*g

```
[ec2-user@ip-172-31-33-186 ~]$ sudo lvcreate -n lv-apps -L 9G webdata-vg
sudo lvcreate -n lv-logs -L 9G webdata-vg
sudo lvcreate -n lv-opt -L 9G webdata-vg
  Logical volume "lv-apps" created.
  Logical volume "lv-logs" created.
  Logical volume "lv-opt" created.
```

To view the entire setup use vgdisplay

```
[ec2-user@ip-172-31-33-186 ~]$ sudo vgdisplay -v
  --- Volume group ---
  VG Name               webdata-vg
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               0
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               <29.99 GiB
  PE Size               4.00 MiB
  Total PE              7677
  Alloc PE / Size       6912 / 27.00 GiB
  Free  PE / Size       765 / <2.99 GiB
  VG UUID               4Qhfsg-yPXM-8FKl-lcO7-5QM4-U0uD-lL2pcC

  --- Logical volume ---
  LV Path                /dev/webdata-vg/lv-apps
  LV Name                lv-apps
  VG Name                webdata-vg
  LV UUID                JC4QrF-2kBz-KH28-xhhA-aK2m-CpUz-y2ewcs
  LV Write Access        read/write
  LV Creation host, time ip-172-31-33-186.eu-west-2.compute.internal, 2023-06-16 21:59:25 +0000
  LV Status              available
  # open                 0
  LV Size                9.00 GiB
  Current LE             2304
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:0
```

To format the disk as xfs and not ext4 used in Project 6, run each of the below commands:

sudo mkfs -t xfs /dev/webdata-vg/lv-apps

sudo mkfs -t xfs /dev/webdata-vg/lv-logs

sudo mkfs -t xfs /dev/webdata-vg/lv-opt

```
meta-data=/dev/webdata-vg/lv-apps isize=512    agcount=4, agsize=589824 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1
data     =                       bsize=4096   blocks=2359296, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
meta-data=/dev/webdata-vg/lv-logs isize=512    agcount=4, agsize=589824 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1
data     =                       bsize=4096   blocks=2359296, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
meta-data=/dev/webdata-vg/lv-opt isize=512    agcount=4, agsize=589824 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1
data     =                       bsize=4096   blocks=2359296, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Now create mount points with and confirm it's been created by listing the items in the directory

sudo mkdir /mnt/apps

sudo mkdir /mnt/logs

sudo mkdir /mnt/opt

```
[ec2-user@ip-172-31-33-186 ~]$ ls /mnt
apps  logs  opt
```

Now mount paths

sudo mount /dev/webdata-vg/lv-apps /mnt/apps
sudo mount /dev/webdata-vg/lv-logs /mnt/logs
sudo mount /dev/webdata-vg/lv-opt /mnt/opt

Install NFS Server

As usual, it is important to check for updates with *sudo yum -y update* before making any installation on the server. Then start the NFS server service and then check the status to ensure it is active.

```
[ec2-user@ip-172-31-33-186 ~]$ sudo yum install nfs-utils -y
```

```
[ec2-user@ip-172-31-33-186 ~]$ sudo systemctl start nfs-server.service
[ec2-user@ip-172-31-33-186 ~]$ sudo systemctl enable nfs-server.service
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service → /usr/lib/systemd/system/nfs-server.service.
[ec2-user@ip-172-31-33-186 ~]$ sudo systemctl status nfs-server.service
● nfs-server.service - NFS server and services
     Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; preset: disa>     Active: active (exited) since Fri 2023-06-16 22:00:46 UTC; 18s a
go
   Main PID: 15150 (code=exited, status=0/SUCCESS)
        CPU: 28ms

Jun 16 22:00:46 ip-172-31-33-186.eu-west-2.compute.internal systemd[1]: Starting NFS s>Jun 16 22:00:46 ip-172-31-33-186.eu-west-2.compute.internal systemd[1
```

To ensure that we set up permission that will allow our Web servers to read, write and execute files on NFS, I will need to change ownership and modify the permissions before restarting the NFS server again and checking the status to ensure it is active

```
[ec2-user@ip-172-31-10-140 ~]$ sudo chown -R nobody: /mnt/apps
[ec2-user@ip-172-31-10-140 ~]$ sudo chown -R nobody: /mnt/logs
[ec2-user@ip-172-31-10-140 ~]$ sudo chown -R nobody: /mnt/opt
```

```
[ec2-user@ip-172-31-10-140 ~]$ sudo chmod -R 777 /mnt/apps
[ec2-user@ip-172-31-10-140 ~]$ sudo chmod -R 777 /mnt/logs
[ec2-user@ip-172-31-10-140 ~]$ sudo chmod -R 777 /mnt/opt
```

sudo systemctl restart nfs-server.service
sudo systemctl status nfs-server.service

Expose the mount using the mount for the webserver CIDR -172.31.32.0/20, then configure its access to NFS and then save these mount details.

```
[ec2-user@ip-172-31-33-186 ~]$ sudo vi /etc/exports
```

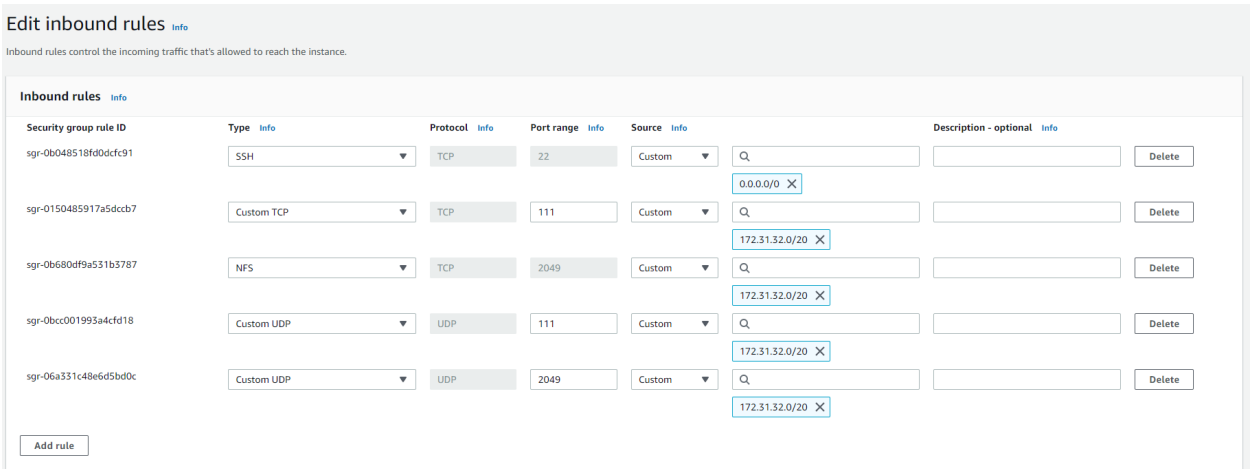
```
/mnt/apps 172.31.32.0/20(rw,sync,no_all_squash,no_root_squash)
/mnt/logs 172.31.32.0/20(rw,sync,no_all_squash,no_root_squash)
/mnt/opt 172.31.32.0/20(rw,sync,no_all_squash,no_root_squash)
```

To allow the webserver to see the mount points when it wants to connect run *sudo exportfs -arv*

```
[ec2-user@ip-172-31-33-186 ~]$ sudo exportfs -arv
exporting 172.31.32.0/20:/mnt/opt
exporting 172.31.32.0/20:/mnt/logs
exporting 172.31.32.0/20:/mnt/apps
```

rpcinfo -p | *grep nfs* command will allow you to check which port is used by the NFS and then I had to open it using Security Groups on AWS. Ensure to use the CIDR range (172.31.32.0/20) for the webserver.

```
[ec2-user@ip-172-31-33-186 ~]$ rpcinfo -p | grep nfs
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100227    3   tcp   2049  nfs_acl
```



Set up the Database server

Run the application update before installing the package and then connect to the MySQL database.

```
ubuntu@ip-172-31-35-205:~$ sudo apt update
```

```
ubuntu@ip-172-31-35-205:~$ sudo apt install mysql-server -y
```

```
ubuntu@ip-172-31-35-205:~$ sudo mysql
```

I will need to create the database as tooling, create the database user, and grant privileges

```
mysql> create database tooling;
Query OK, 1 row affected (0.00 sec)

mysql> create user 'webaccess'@'172.31.32.0/20' identified by '[redacted]';
Query OK, 0 rows affected (0.02 sec)

mysql> grant all privileges on tooling.* to 'webaccess'@'172.31.32.0/20';
Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)
```

You should now be able to see the created database with show databases;

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| tooling            |
+--------------------+
5 rows in set (0.01 sec)
```

There is a need to change the bind address and MySQL bind address to 0.0.0.0 with the below steps.

```
ubuntu@ip-172-31-35-205:~$ cd /etc/mysql/mysql.conf.d/
```

```
ubuntu@ip-172-31-35-205:/etc/mysql/mysql.conf.d$ sudo vi mysqld.cnf
```

```
ubuntu@ip-172-31-35-205:/etc/mysql/mysql.conf.d$ sudo systemctl restart mysql.service
```

Install Web Server

Repeat the steps across all 3 web servers. Run an update and then install the NFS Client.

```
[ec2-user@ip-172-31-39-161 ~]$ sudo yum update -y
```

```
[ec2-user@ip-172-31-39-161 ~]$ sudo yum install nfs-utils nfs4-acl-tools -y
```

Mount var/www and target the server's export for apps using the NFS Server Private IP Address

```
[ec2-user@ip-172-31-39-161 ~]$ sudo mkdir /var/www
[ec2-user@ip-172-31-39-161 ~]$ sudo mount -t nfs -o rw,nosuid 172.31.33.186:/mnt/apps /var/www
```

To view the disk available on the computer, use *df -h*

```
[ec2-user@ip-172-31-39-161 ~]$ df -h
Filesystem                Size  Used Avail Use% Mounted on
devtmpfs                  4.0M     0  4.0M   0% /dev
tmpfs                     385M     0  385M   0% /dev/shm
tmpfs                     154M  4.4M  150M   3% /run
/dev/xvda4                9.4G  1.3G  8.1G  14% /
/dev/xvda3                495M  153M  343M  31% /boot
/dev/xvda2                200M  8.0K  200M   1% /boot/efi
tmpfs                      77M     0   77M   0% /run/user/1000
172.31.33.186:/mnt/apps  9.0G   97M  8.9G   2% /var/www
```

To retain the information on the web server edit the /etc/fstab configuration and add the Private IP of the NFS server.

```
[ec2-user@ip-172-31-39-161 ~]$ sudo vi /etc/fstab
```

172.31.44.176:/mnt/apps /var/www nfs defaults 0 0

Reload daemon if required - *sudo systemctl daemon-reload*

Install PHP from Remi's repository on the web server

sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm

sudo dnf install dnf-utils http://rpms.remirepo.net/enterprise/remi-release-8.rpm

sudo dnf module reset php

sudo dnf module enable php -y

sudo dnf install php php-opcache php-gd php-curl php-mysqlnd -y

sudo systemctl start php-fpm

sudo systemctl enable php-fpm

sudo systemctl status php-fpm - - -press q to step out

sudo setsebool -P httpd_execmem 1

Fork the tooling source code from Darey.io Github Account (https://github.com/darey-io/tooling.git) to my GitHub account. You need to ensure you sign in first into GitHub, select fork, click on the code and then copy the HTTPS path then go back to the web server. It should look like this https://github.com/enyemg/tooling.git

I installed git on the web server by running and then cloned the data in the URL before using ls to ensure I had cloned tooling successfully.

```
[ec2-user@ip-172-31-39-161 ~]$ sudo yum install git -y
```

```
[ec2-user@ip-172-31-39-161 ~]$ sudo git clone https://github.com/enyemg/tooling.git
Cloning into 'tooling'...
remote: Enumerating objects: 234, done.
remote: Total 234 (delta 0), reused 0 (delta 0), pack-reused 234
Receiving objects: 100% (234/234), 282.72 KiB | 2.98 MiB/s, done.
Resolving deltas: 100% (130/130), done.
[ec2-user@ip-172-31-39-161 ~]$ ls
tooling
```

Now use *sudo cp -R html/. /var/www/html* to move the html folder to var/www/html. Ensure port 80 is opened in EC2 on the web server and the source will be anywhere.

Install Apache

sudo yum install httpd -y

sudo setenforce 0

sudo vi /etc/sysconfig/selinux

- - change SELINUX to disabled

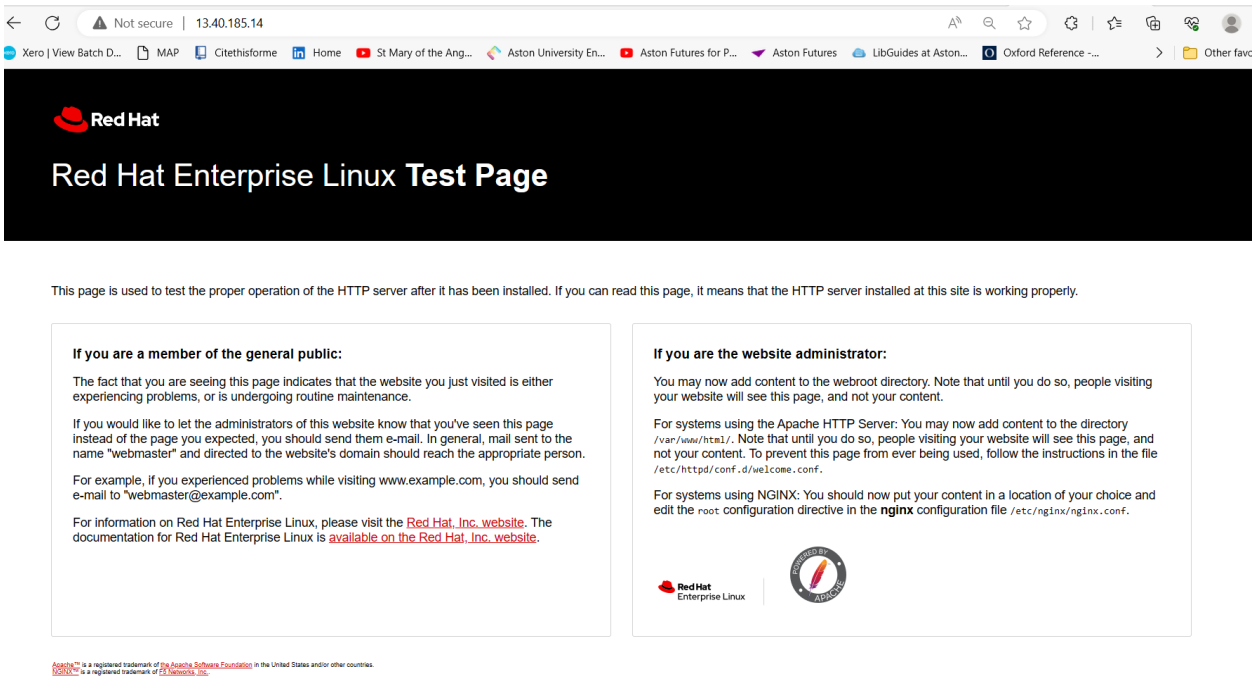
sudo systemctl restart httpd

sudo systemctl start httpd

sudo systemctl status httpd

```
[ec2-user@ip-172-31-39-161 ~]$ sudo systemctl status php-fpm
● php-fpm.service - The PHP FastCGI Process Manager
     Loaded: loaded (/usr/lib/systemd/system/php-fpm.service; enabled; preset: disable>     Active: active (running)
 since Fri 2023-06-16 22:27:12 UTC; 15s ago
   Main PID: 17508 (php-fpm)
     Status: "Processes active: 0, idle: 5, Requests: 0, slow: 0, Traffic: 0req/sec"
      Tasks: 6 (limit: 4421)
     Memory: 13.1M
        CPU: 58ms
     CGroup: /system.slice/php-fpm.service
             ├─17508 "php-fpm: master process (/etc/php-fpm.conf)"
             ├─17509 "php-fpm: pool www"
             ├─17510 "php-fpm: pool www"
             ├─17511 "php-fpm: pool www"
             ├─17512 "php-fpm: pool www"
             └─17513 "php-fpm: pool www"
```

The web page should display as the below.



To set up the HTML directory for the webservers

```
[ec2-user@ip-172-31-39-161 ~]$ sudo vi /var/www/html/functions.php
```

You will then need to populate the below information in functions.php

mysqli_connect('db private ip', 'webaccess', 'database password', 'tooling');

To install MySQL on the web server so that I can ssh into it, I will need to also open port 3306 for My SQL and add the CIDR of the webserver in the Security Group on the DBserver on EC2.

```
[ec2-user@ip-172-31-39-161 ~]$ sudo yum install mysql -y
```

Now *cd tooling* or ensure you are in the tooling directory and then run *mysql -h <databse-private-ip> -u <db-username> -p <db-pasword> < tooling-db.sql*

```
[ec2-user@ip-172-31-39-161 tooling]$  mysql -h 172.31.35.205 -u webaccess -p tooling < tooling-db.sql
Enter password:
```

To disable the welcome page on the web server use the below and ensure that the service is running afterwards.

```
[ec2-user@ip-172-31-39-161 tooling]$ sudo mv /etc/httpd/conf.d/welcome.conf /etc/httpd/conf.d/welcome.backup
[ec2-user@ip-172-31-39-161 tooling]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-39-161 tooling]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)    Drop-In: /usr/lib/systemd
/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Fri 2023-06-16 22:35:40 UTC; 6s ago
       Docs: man:httpd.service(8)
   Main PID: 17793 (httpd)
     Status: "Started, listening on: port 80"
      Tasks: 213 (limit: 4421)
     Memory: 23.1M
        CPU: 73ms
```

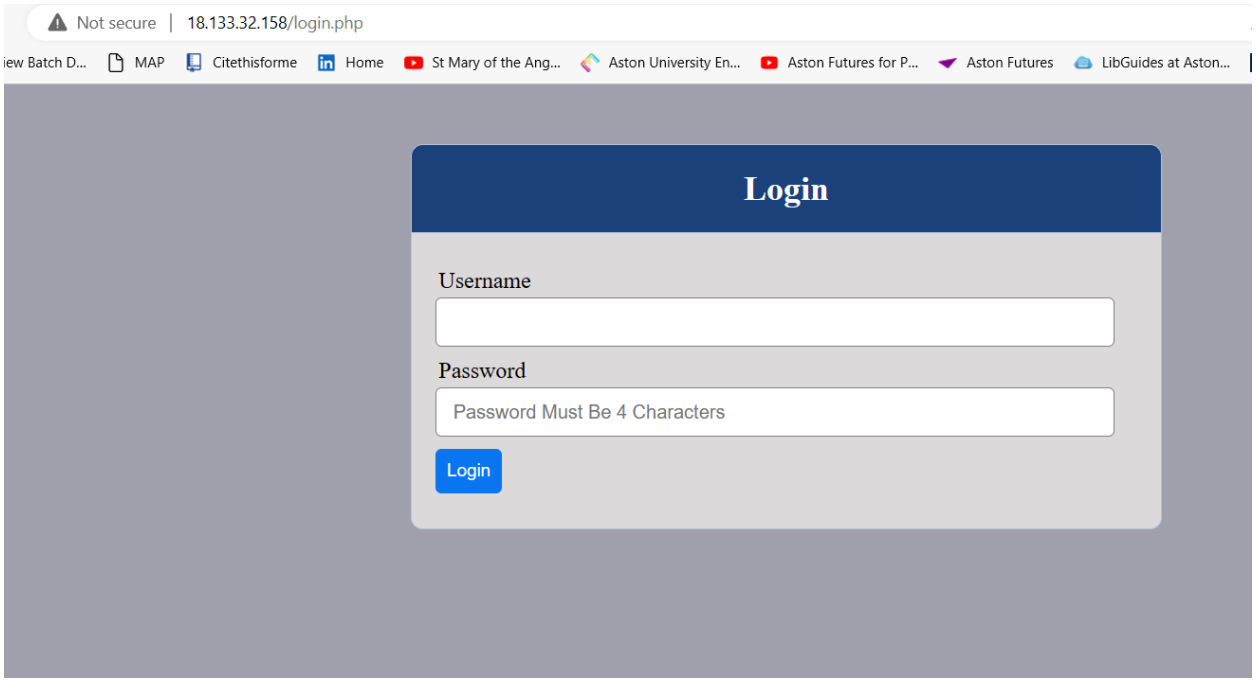
Revert to the database and you should now see more results from each command run.

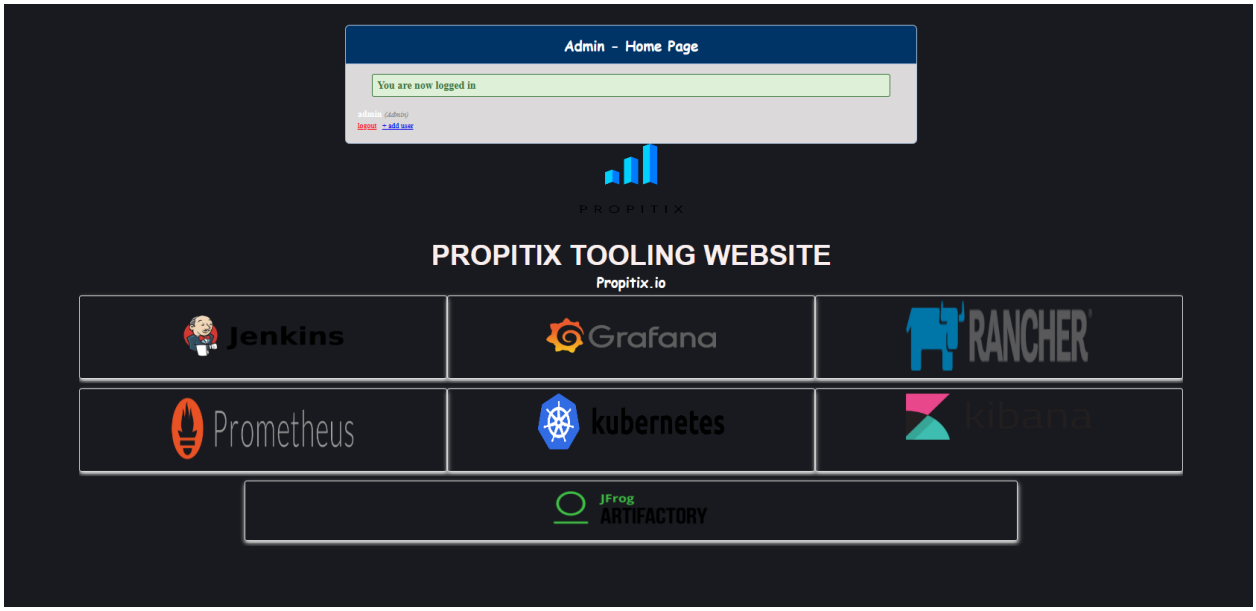
```
mysql> use tooling;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-------------------+
| Tables_in_tooling |
+-------------------+
| users             |
+-------------------+
1 row in set (0.00 sec)

mysql> select * from users;
+----+----------+----------------------------------+---------------+-----------+--------+
| id | username | password                         | email         | user_type | status |
+----+----------+----------------------------------+---------------+-----------+--------+
|  1 | admin    | 21232f297a57a5a743894a0e4a801fc3 | dare@dare.com | admin     | 1      |
+----+----------+----------------------------------+---------------+-----------+--------+
1 row in set (0.00 sec)
```

Browse the webservers with their respective IPs or DNS and it should display as the Propitix Tooling website upon successful login. This means that I have now implemented a web solution for a DevOps team using a LAMP stack with a remote Database and NFS servers.


Not secure | 18.133.32.158/login.php
MAP
Citethisforme
Home
St Mary of the Ang...
Aston University En...
Aston Futures for P...
Aston Futures
LibGuides at Aston...
Login
Username
Password
Password Must Be 4 Characters
Login


Admin - Home Page
You are now logged in
logout
+ add user
PROPITIX
PROPITIX TOOLING WEBSITE
Propitix.io
Jenkins
Grafana
RANCHER
Prometheus
kubernetes
kibana
JFrog
ARTIFACTORY

Source: https://www.dareyio.com/docs/devops-tooling-website-solution/