Cloud Assignment

Koganti Ramya

1. EC2 Instances

Steps:

1. Open AWS and under search box select EC2

2. Click on launch instance

3. Now enter name of your machine (Machine1-Ramya)

4. Under AMI select Amazon Linux

5. under instance type select t2.micro

6. Under key pair >> create a new key pair (TOKYO.pem)

7. Under firewall security group click on create security group

8. And click on launch instance

9. Repeat the same process and create one more instance (Machine2-Koganti)

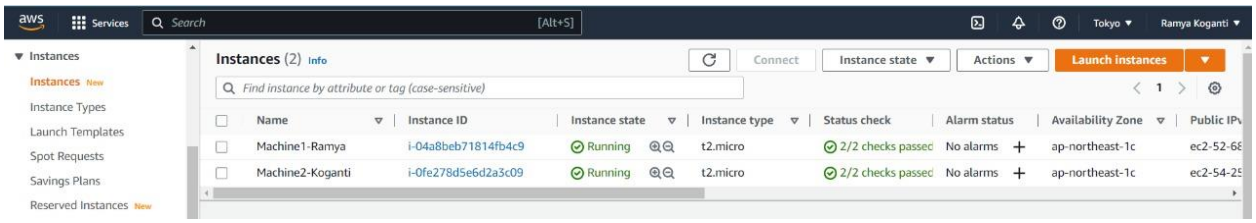


Fig1:Ec2 instances

10. Steps to connect to ec2 machines.

- Select machine1 under instances tab and click on connect.
- Under connect to instance select SSH client
- Now copy the ssh command shown under example
- Now go to the .pem file location directory and open command prompt terminal
- Now paste the ssh command and click on enter
- Type yes to connect and you can see your Machine1-Ramya instance running

Cloud Assignment

Koganti Ramya

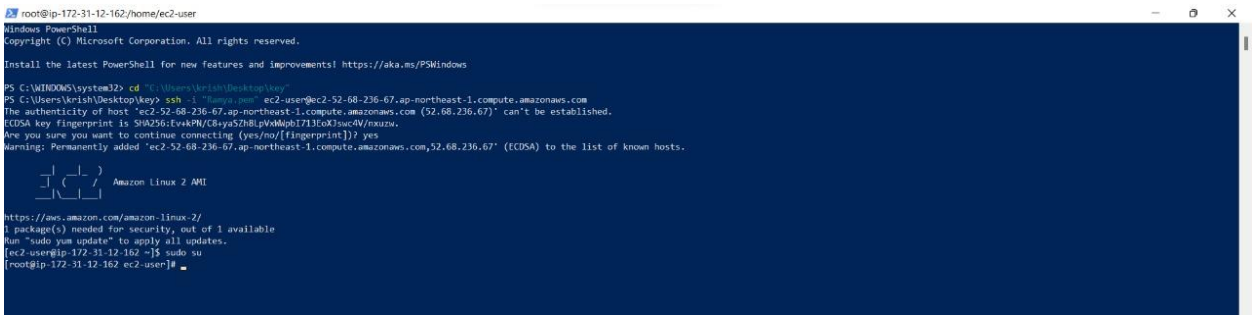


Fig2.Machine1-Ramya

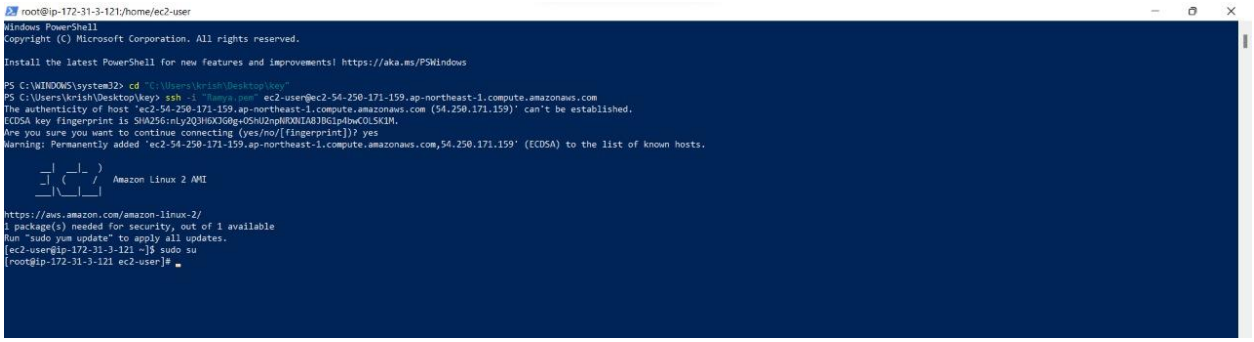


Fig3.Machine2- Koganti

2. EBS volume

Steps:

1. When Ever you want to provide an extra storage to your machine you opt for this Elastic Block store (EBS).

2. Created Two machines Ramya-Machine A and Ramya-Machine B in Oregon region.

Fig4: Instances for EBS

3 .Under EBS select volumes and you can see default storage allocated for your EC2 machines.

4. Now click on create volume

• Under volume type select any type you want (General purpose SSD (gp2))

• Under size select the amount of GB (1GB)

Cloud Assignment

Koganti Ramya

- Under Availability zone you can select available zone in which your instance got created.
- Now click on create volume.
- Now click on volumes and you can see all volumes and newly created EBS.
- Now select the EBS and click on actions and click on attach volume.
- Under Basic details select your instance and click on attach volume.

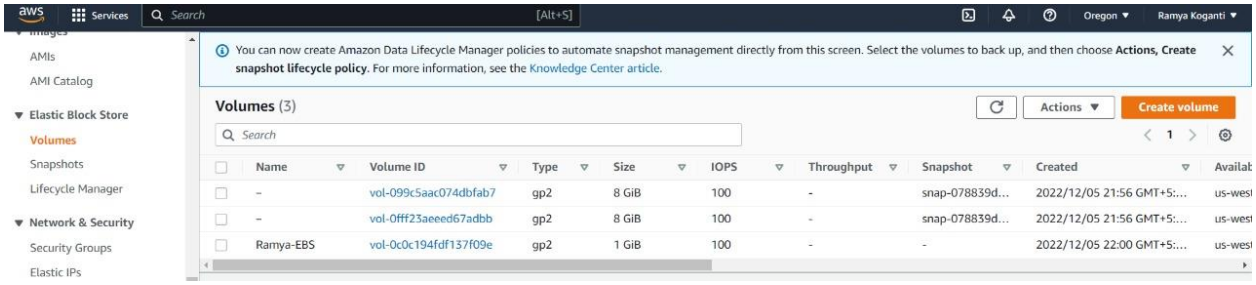


Fig5: EBS volume of 1GB

6. Now log on to Ramya-MachineA and make a file system and mount it.

- lsblk – to list all file systems
- mkdir to create a storage directory
- mkfs -t xfs /dev/sdf
- mount -t xfs /dev/sdf storage
- created a storage directory named Ramya-Storage
- mounted it to file system and created ten .txt files in it
- umounted the file system.

Cloud Assignment

Koganti Ramya

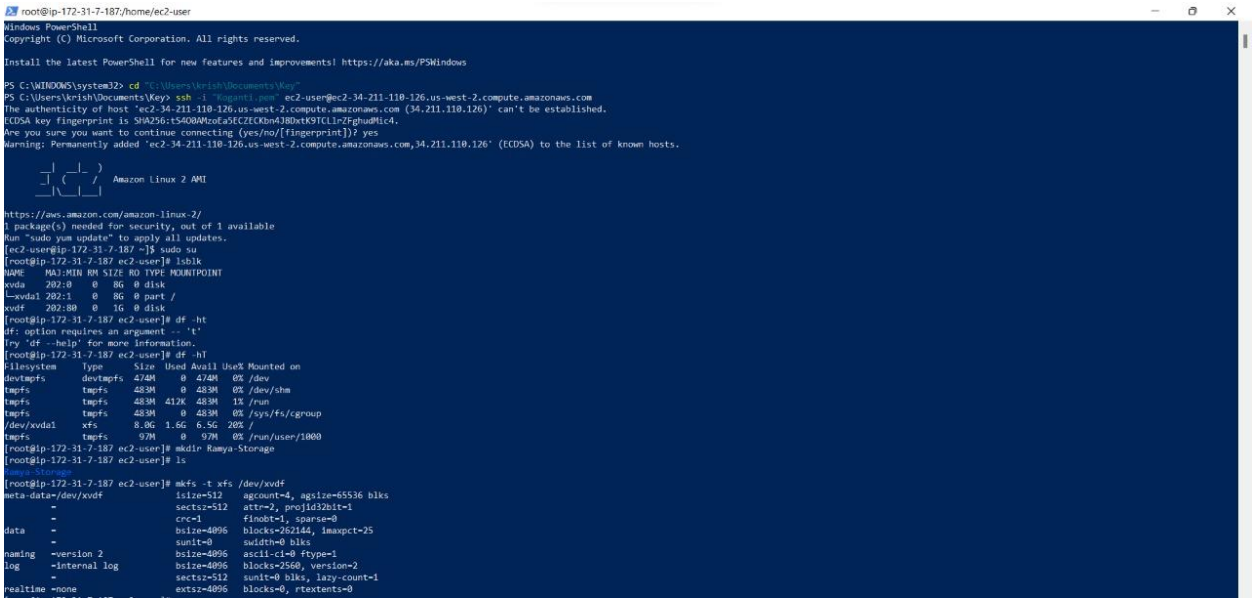


Fig 6: File system created for Ramya-MachineA and mounted it

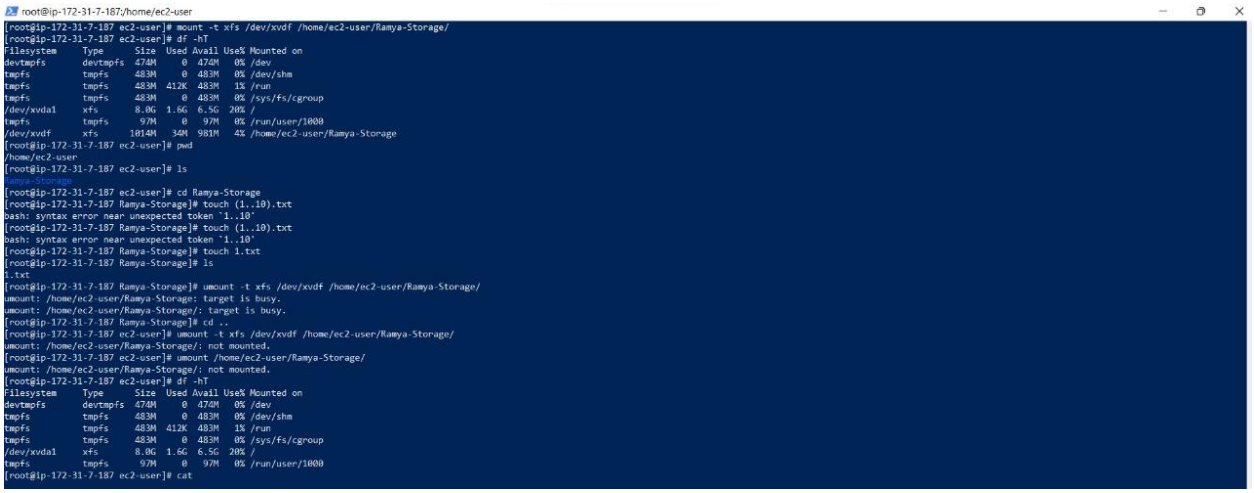


Fig7: Created 1 file in Ramya-Storage and unmounted it

7. Now detach the EBS volume from machine A and attach it to Machine B

8. Now connect to Machine B, create a new directory and mount the same to it.

9. Ramya-MachineB EBS volume contains all the one txt files.

Cloud Assignment

Koganti Ramya

Fig8:Ramya-MachineB EBS

3 . Snapshot

Steps:

1. Under EC2 Elastic Block store click on Snapshot

2. Click on create snapshot

3. Under volume id select your volume (Ramya-EBS) in Oregon region

4. Under description enter name of snapshot

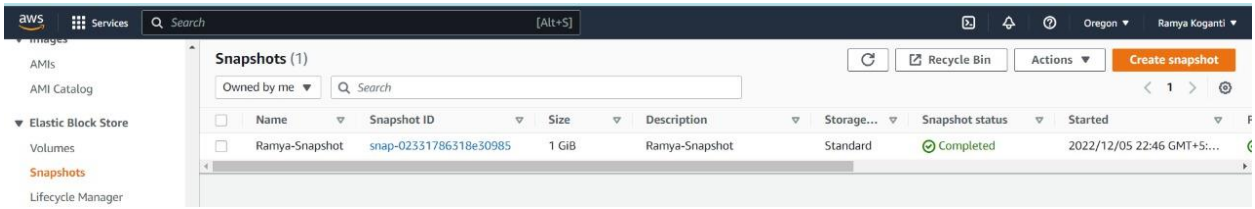
5. Now click on create snapshot

6. Now click on snapshots and you can able to see your created snapshot

7. Select your snapshot and click on actions and click on copy snapshot

8. In settings page of copy snapshot , under Destination region select the region where you want to create Tokyo (ap-northeast1)

9. Now click on copy snapshot



Cloud Assignment

Koganti Ramya

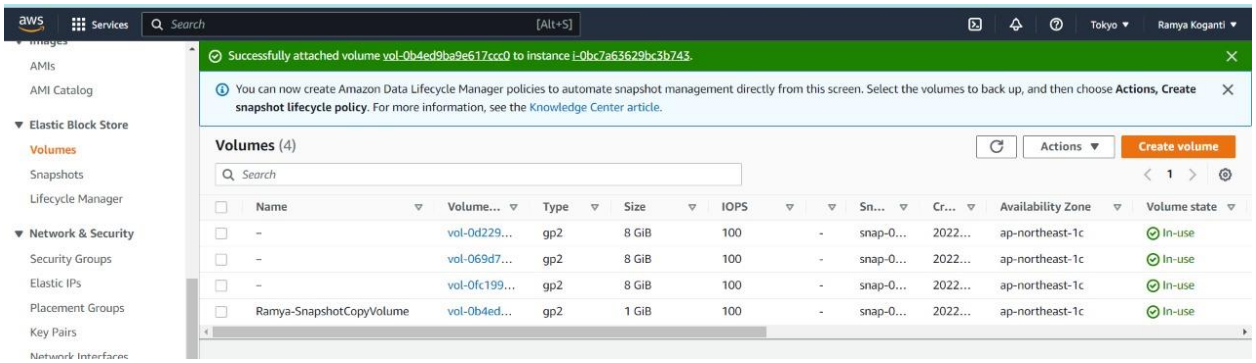


Fig 10: volume created from copy snapshot in Tokyo region

10. Create a Ramya-Machine C in Tokyo region and attach the EBS volume created from Snapshot copy
11. Now connect to Ramya-Machine C and create a new storage directory named Ramya Snaphot Volume and mount it.

12. Switch to the Ramya-SnaphotVolume directory and check the list of files in it.

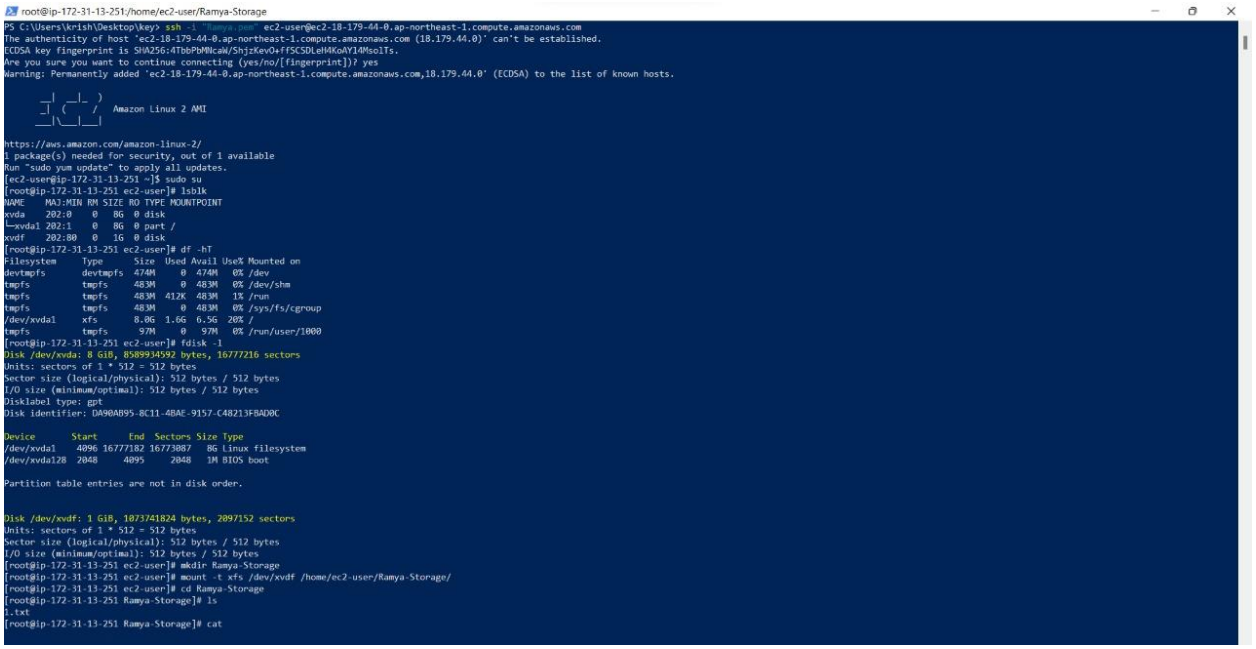


Fig 11: Ramya - Machine C Snapshot Volume

4. AMI

An Amazon Machine Image (AMI) is a template that contains a software configuration (for example, an operating system, an application server, and applications). From an AMI, you launch an instance, which is a copy of the AMI running as a virtual server in the cloud.

Steps:

Cloud Assignment

Koganti Ramya

• Created an Ramya-Machine1 Instance and in the security-groups add inbound rule http port 80 for this machine.

• Connect to the above instance and perform the below commands

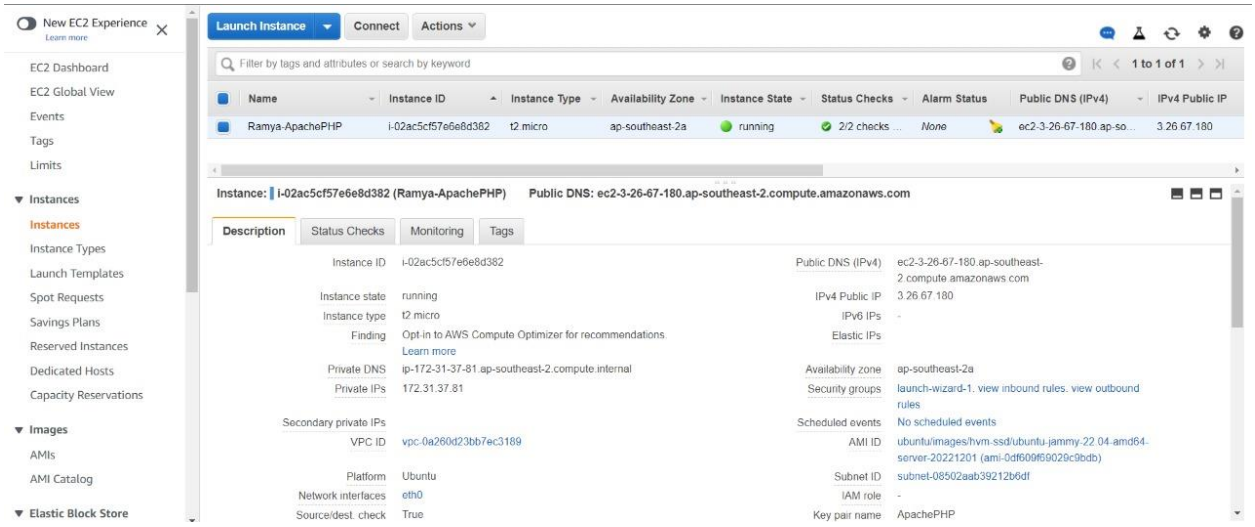


Fig12: Ramya-Machine1

• Preparing your Ubuntu server

1. sudo apt update
2. sudo ufw allow ssh
3. sudo ufw allow 80
4. sudo ufw allow 443
5. sudo ufw enable

Cloud Assignment

Koganti Ramya

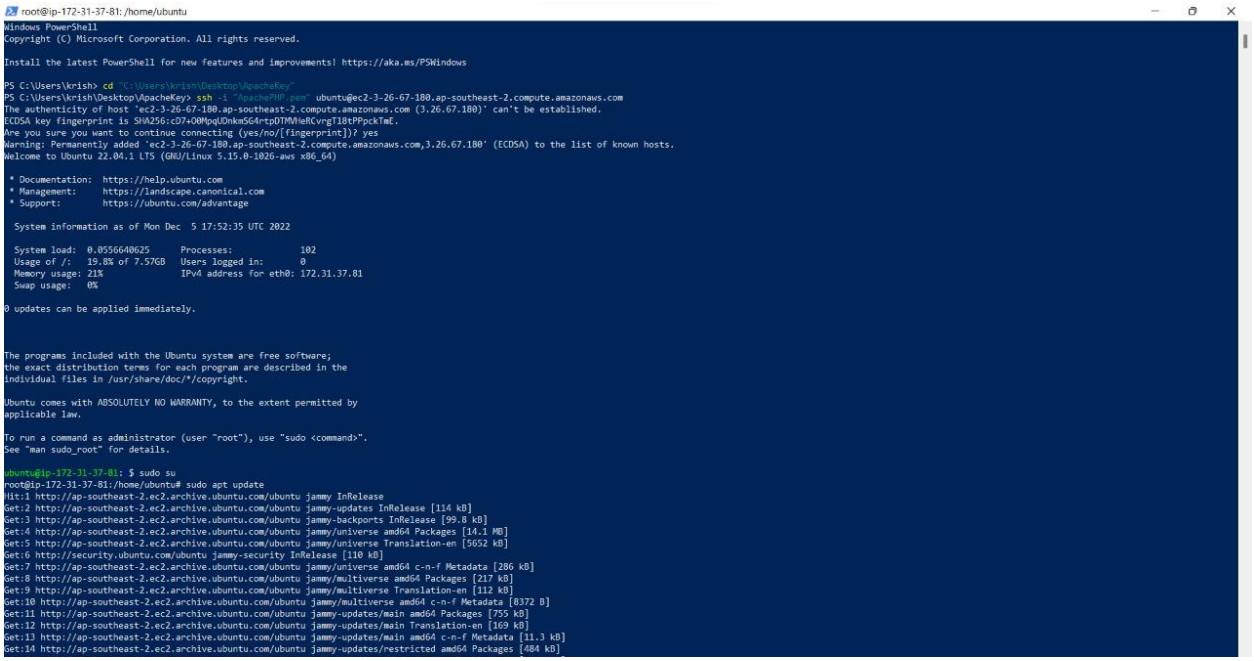


Fig13: Preparing your Ubuntu server

- Installing and testing Apache2

    1. sudo apt install apache2

    2. sudo systemctl status apache2

    3. http://YOURSERVERIPADDRESS/

Cloud Assignment

Koganti Ramya

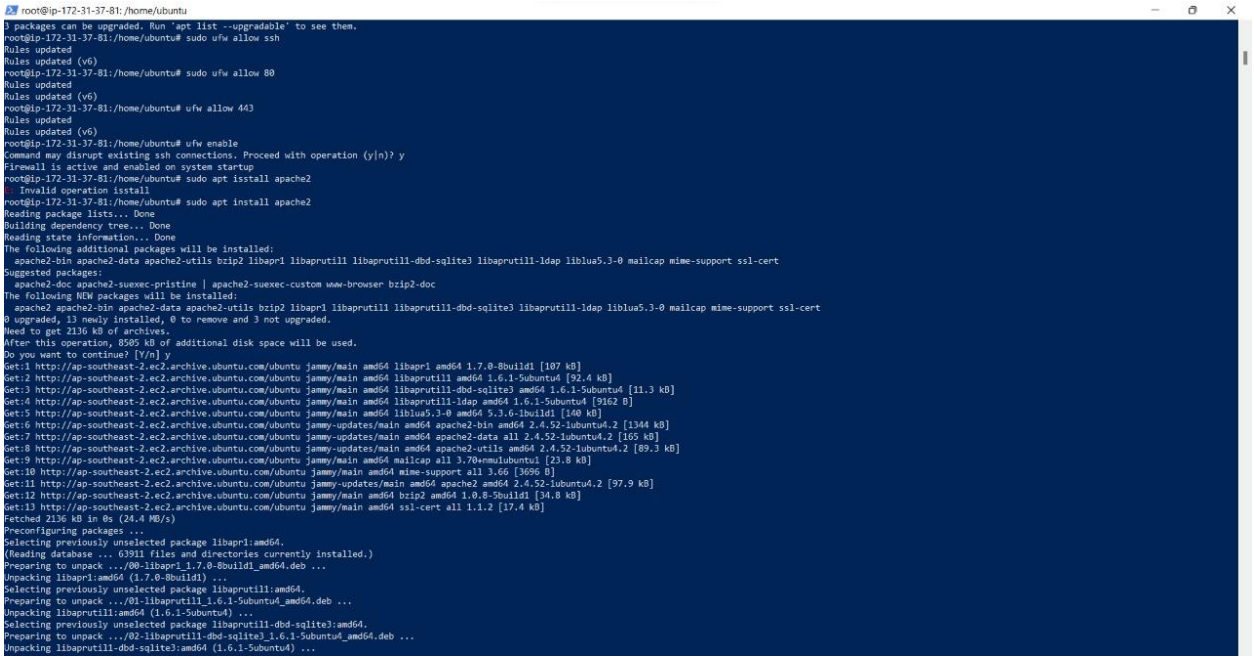


Fig 14: connecting to apache2

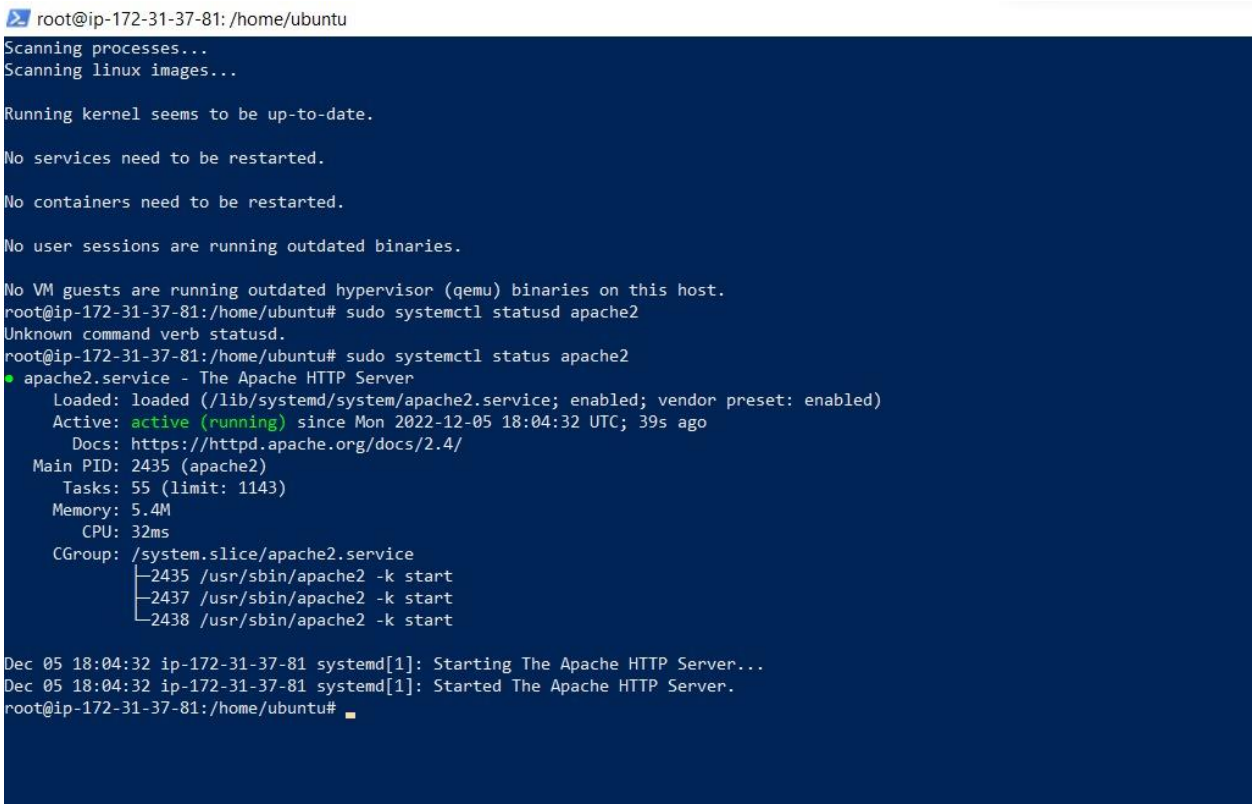


Fig 15: Testing apache2

Cloud Assignment

Koganti Ramya

Fig 16: Testing apache2 on browser

Installing and testing PHP

1. sudo apt install php8.1
2. php --version
3. sudo systemctl restart apache2
4. echo '' | sudo tee -a /var/www/html/phpinfo.php > /dev/null
5. http://YOURSERVERIPADDRESS/phpinfo.php

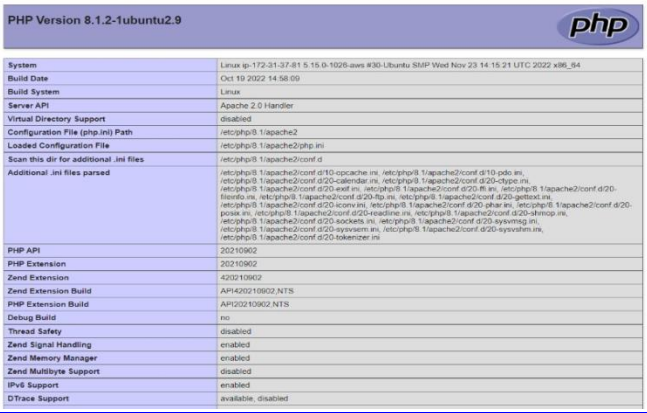


Fig 17: Testing PHP on browser

Now create AMI from existing instance

Cloud Assignment

Koganti Ramya

- From this AMI create a new instance and in the security-group add inbound rule for http port 80

- Connect to this instance and try to access the ubuntu and php on browser

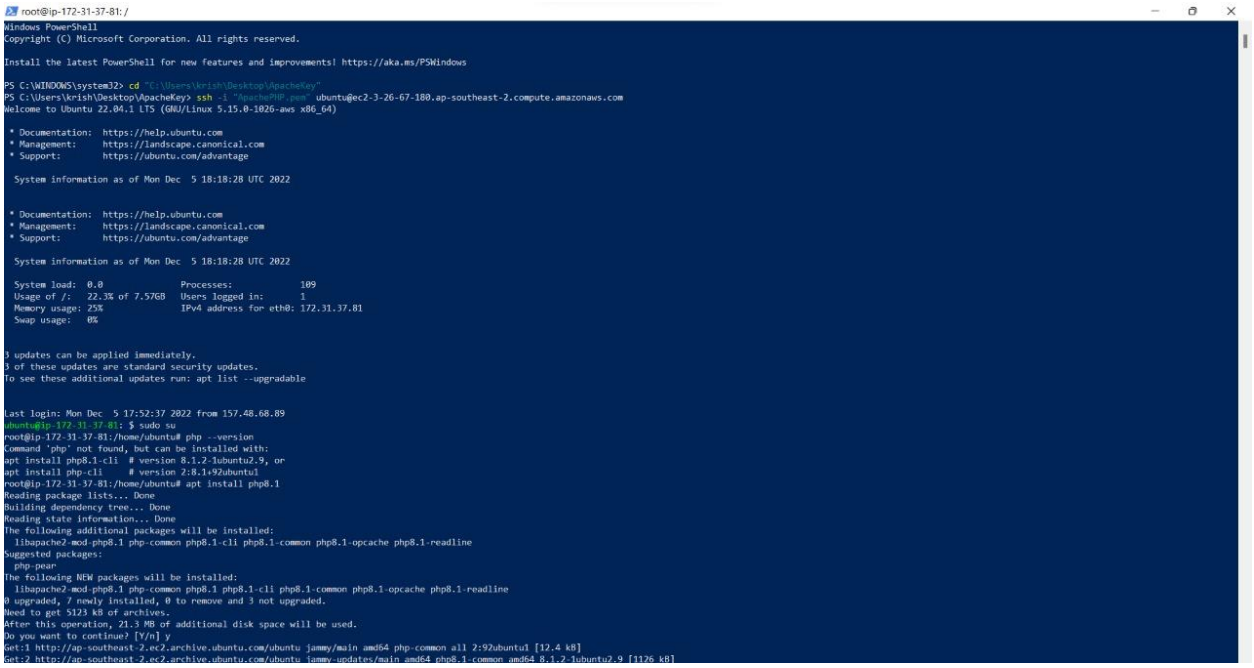


Fig 18: Connecting to PHP

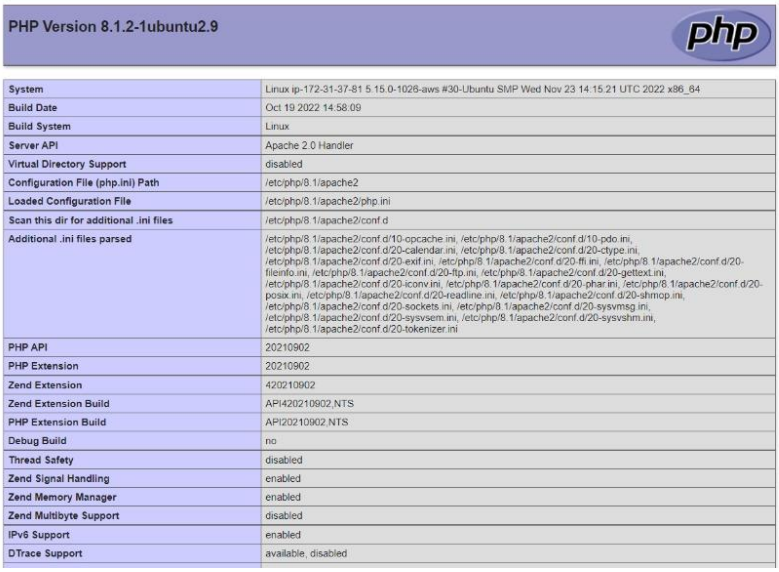
Cloud Assignment

Koganti Ramya

```
root@ip-172-31-37-81: /
Creating config file /etc/php/8.1/mods-available/sysvshm.ini with new version

Creating config file /etc/php/8.1/mods-available/tokenizer.ini with new version
Setting up php8.1-readline (8.1.2-1ubuntu2.9) ...

Creating config file /etc/php/8.1/mods-available/readline.ini with new version
Setting up php8.1-opcache (8.1.2-1ubuntu2.9) ...

Creating config file /etc/php/8.1/mods-available/opcache.ini with new version
Setting up php8.1-cli (8.1.2-1ubuntu2.9) ...
update-alternatives: using /usr/bin/php8.1 to provide /usr/bin/php (php) in auto mode
update-alternatives: using /usr/bin/phar8.1 to provide /usr/bin/phar (phar) in auto mode
update-alternatives: using /usr/bin/phar.phar8.1 to provide /usr/bin/phar.phar (phar.phar) in auto mode

Creating config file /etc/php/8.1/cli/php.ini with new version
Setting up libapache2-mod-php8.1 (8.1.2-1ubuntu2.9) ...

Creating config file /etc/php/8.1/apache2/php.ini with new version
Module mpm_event disabled.
Enabling module mpm_prefork.
apache2_switch_mpm Switch to prefork
apache2_invoke: Enable module php8.1
Setting up php8.1 (8.1.2-1ubuntu2.9) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for php8.1-cli (8.1.2-1ubuntu2.9) ...
Processing triggers for libapache2-mod-php8.1 (8.1.2-1ubuntu2.9) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-37-81:/home/ubuntu# cd /
root@ip-172-31-37-81:/# php --version
PHP 8.1.2-1ubuntu2.9 (cli) (built: Oct 19 2022 14:58:09) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.1.2, Copyright (c) Zend Technologies
    with Zend OPcache v8.1.2-1ubuntu2.9, Copyright (c), by Zend Technologies
root@ip-172-31-37-81:/# sudo systemctl restart apache2
root@ip-172-31-37-81:/# echo '<?php phpinfo();?>' | sudo tee -a /var/www/html/phpinfo.php > /dev/null
root@ip-172-31-37-81:/#
```

Fig 19: Ramya-Machine2

Fig 20: Testing ubuntu for Ramya-Machine2



PHP Version 8.1.2-1ubuntu2.9

System	Linux ip-172-31-37-81 5.15.0-1026-aws #30-Ubuntu SMP Wed Nov 23 14:15:21 UTC 2022 x86_64
Build Date	Oct 19 2022 14:58:09
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/apache2
Loaded Configuration File	/etc/php/8.1/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/apache2/conf.d
Additional .ini files parsed	/etc/php/8.1/apache2/conf.d/10-opcache.ini, /etc/php/8.1/apache2/conf.d/10-pdo.ini, /etc/php/8.1/apache2/conf.d/20-calendar.ini, /etc/php/8.1/apache2/conf.d/20-ctype.ini, /etc/php/8.1/apache2/conf.d/20-exif.ini, /etc/php/8.1/apache2/conf.d/20-ffi.ini, /etc/php/8.1/apache2/conf.d/20-fileinfo.ini, /etc/php/8.1/apache2/conf.d/20-ftp.ini, /etc/php/8.1/apache2/conf.d/20-gettext.ini, /etc/php/8.1/apache2/conf.d/20-iconv.ini, /etc/php/8.1/apache2/conf.d/20-phar.ini, /etc/php/8.1/apache2/conf.d/20-posix.ini, /etc/php/8.1/apache2/conf.d/20-readline.ini, /etc/php/8.1/apache2/conf.d/20-shmop.ini, /etc/php/8.1/apache2/conf.d/20-sockets.ini, /etc/php/8.1/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.1/apache2/conf.d/20-sysvsem.ini, /etc/php/8.1/apache2/conf.d/20-sysvshm.ini, /etc/php/8.1/apache2/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled

Fig 21: Testing PHP for Ramya-Machine2

5. Load Balancer

Steps:

• Create a EC2 machine (Ramya-A) and add security group with inbound rule allowing SSH and HTTP port.

• Prepare your ubuntu server and install and test apache2

• Install and test PHP8.1

• Create an AMI and create two instances from AMI with security group allowing inbound rule for SSH and HTTP port.

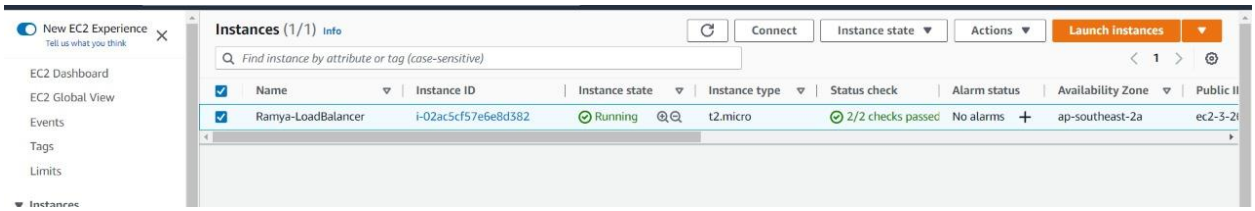


Fig22: Ramya-Load Balancer

Cloud Assignment

Koganti Ramya

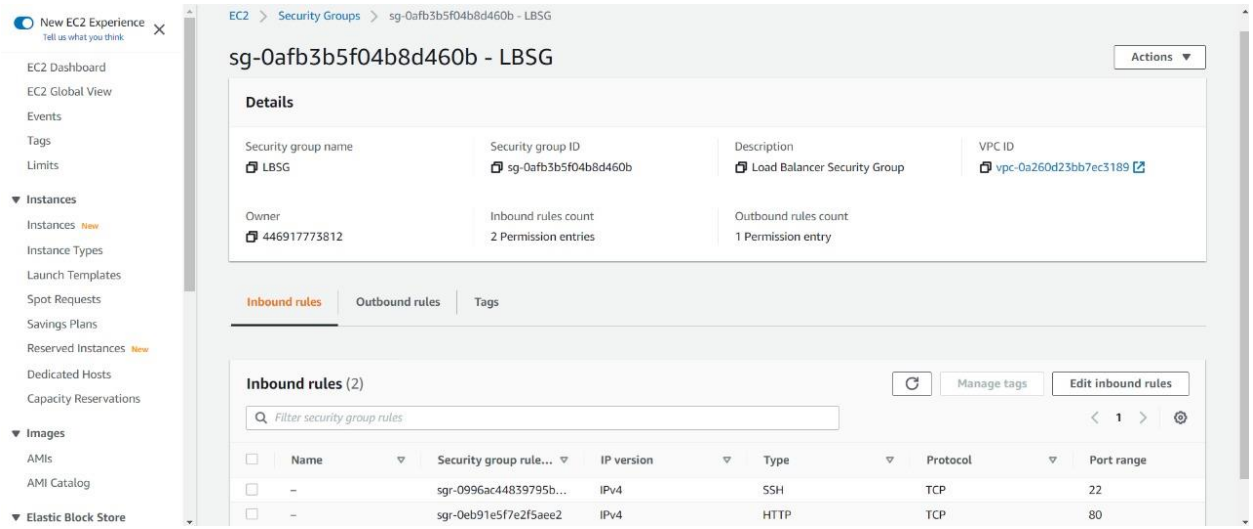


Fig23: Security Group-LBSG for Load Balancer

• Under Load balancing from EC2 service click on Load Balancer and click on create a load balancer.

• Click on create Application load balancer and Give name to your load balancer (Ramya-APLB) and select all mappings under Network Mapping.

• Under security groups create a new security group allowing inbound rules for SSH and HTTP port.

• Under Listeners and routing, need to create a new target group (APLBTG) and include your target machines under it.

• Now connect your Target Group to your Load balancer and click on create

Fig 24: Target Group

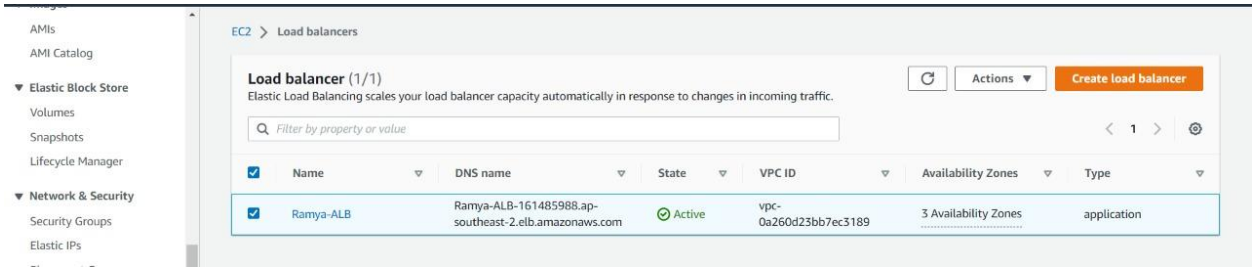


Fig25: Load Balance

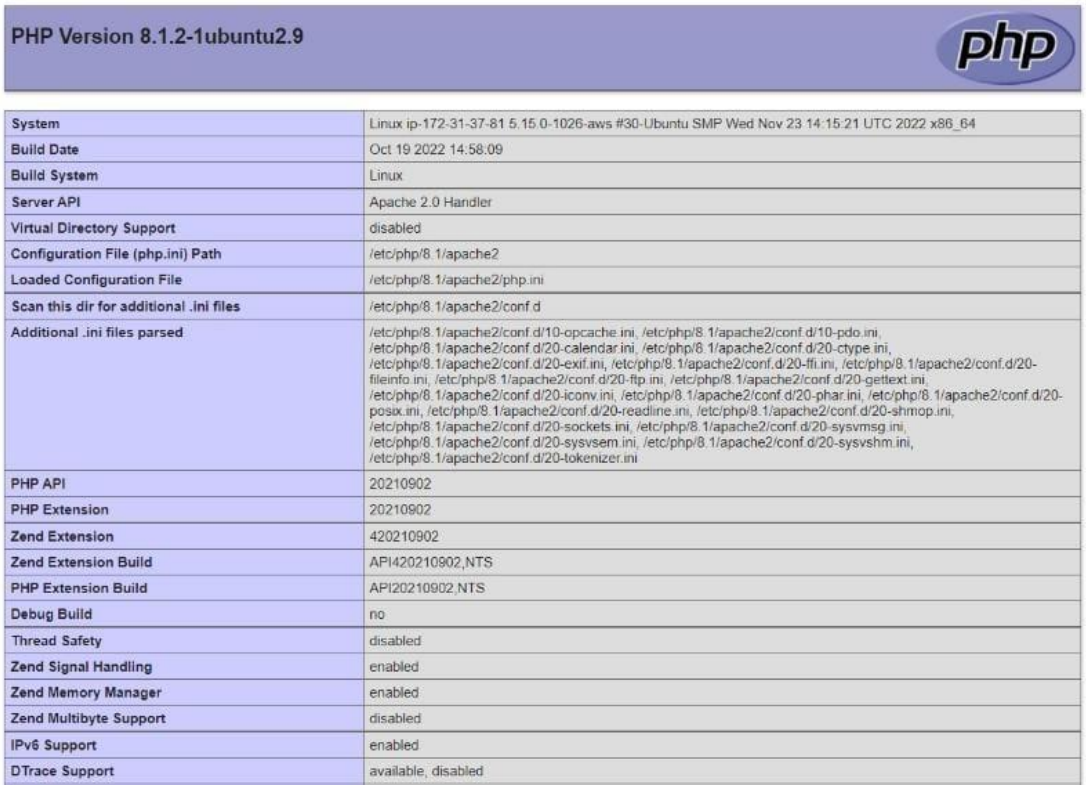
• Now connect to your Load balancer by copying the DNS name and pasting in the browser.

• You can also check to which machine it is being connected using DNS name/phpinfo.php and you can check the ip address of your machine to which it is being connected

Fig26: Connecting to UBUNTU using Load balancer

Cloud Assignment

Koganti Ramya



PHP Version 8.1.2-1ubuntu2.9

System	Linux ip-172-31-37-81 5.15.0-1026-aws #30-Ubuntu SMP Wed Nov 23 14:15:21 UTC 2022 x86_64
Build Date	Oct 19 2022 14:58:09
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/apache2
Loaded Configuration File	/etc/php/8.1/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/apache2/conf.d
Additional .ini files parsed	/etc/php/8.1/apache2/conf.d/10-opcache.ini, /etc/php/8.1/apache2/conf.d/10-pdo.ini, /etc/php/8.1/apache2/conf.d/20-calendar.ini, /etc/php/8.1/apache2/conf.d/20-ctype.ini, /etc/php/8.1/apache2/conf.d/20-exif.ini, /etc/php/8.1/apache2/conf.d/20-ffi.ini, /etc/php/8.1/apache2/conf.d/20-fileinfo.ini, /etc/php/8.1/apache2/conf.d/20-ftp.ini, /etc/php/8.1/apache2/conf.d/20-gettext.ini, /etc/php/8.1/apache2/conf.d/20-iconv.ini, /etc/php/8.1/apache2/conf.d/20-phar.ini, /etc/php/8.1/apache2/conf.d/20-posix.ini, /etc/php/8.1/apache2/conf.d/20-readline.ini, /etc/php/8.1/apache2/conf.d/20-shmop.ini, /etc/php/8.1/apache2/conf.d/20-sockets.ini, /etc/php/8.1/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.1/apache2/conf.d/20-sysvsem.ini, /etc/php/8.1/apache2/conf.d/20-sysvshm.ini, /etc/php/8.1/apache2/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled

Fig27: Connecting to PHP using Load balancer