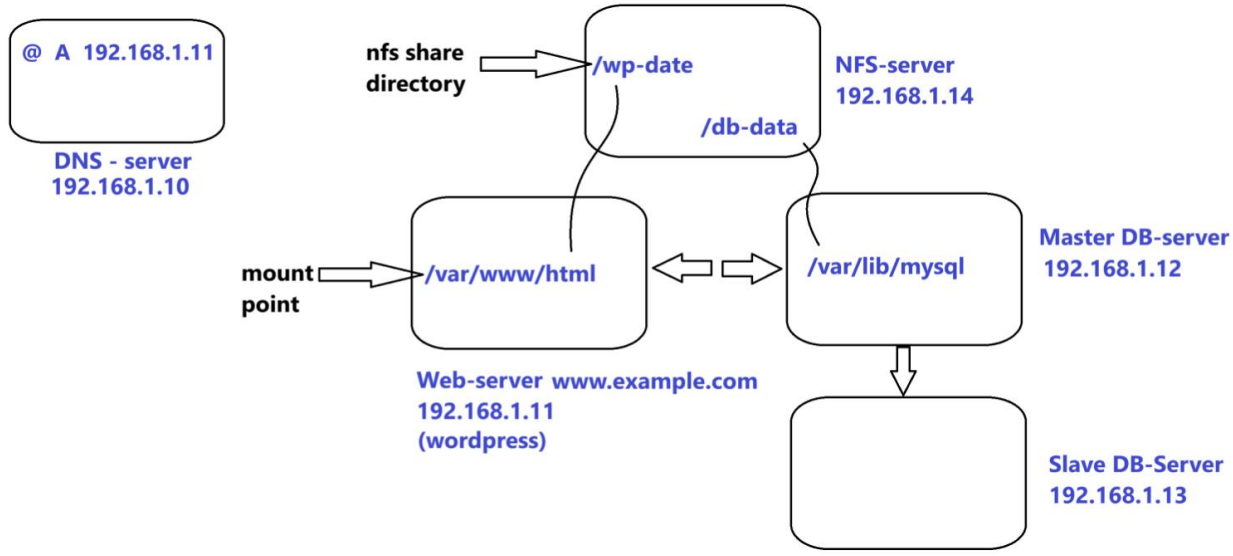
Project Title: Implementation of a Multi-Server Architecture using DNS, NFS, Web Server, and MariaDB



Objective:

To design and deploy a multi-server infrastructure that includes a DNS server, an NFS server, a web server hosting WordPress, and a MariaDB database server (Master-Slave setup). This architecture ensures centralized storage, database replication, and seamless web hosting.

Project Scope:

- **DNS Server:** Resolve domain names to IP addresses.
- **NFS Server:** Provide shared storage for web data and database storage.
- **Web Server:** Host a WordPress website with data stored on the NFS server.
- **Database Server:** Implement a Master-Slave MariaDB setup for data redundancy and high availability.

Network Topology:

1. **DNS Server:** 192.168.1.10

2. **Web Server:** 192.168.1.11 (www.example.com, WordPress hosted)
3. **Master Database Server:** 192.168.1.12
4. **Slave Database Server:** 192.168.1.13
5. **NFS Server:** 192.168.1.14

Functional Requirements:

- **DNS Server:**
  - Configure a DNS server to resolve `www.example.com` to `192.168.1.11`.
  - Ensure proper hostname resolution for all servers.
- **NFS Server:**
  - Create shared directories:
    - `/wp-data` for WordPress files (mounted on `/var/www/html` of the Web Server)
    - `/db-data` for MariaDB data (mounted on `/var/lib/mysql` of the Master DB Server)
- **Web Server:**
  - Install and configure Apache/Nginx with PHP support.
  - Mount NFS shared directory `/wp-data` to `/var/www/html`.
  - Install and set up WordPress with a database connection.
- **Database Server:**
  - Install MariaDB on Master and Slave servers.
  - Set up Master-Slave replication for database consistency.
  - Mount `/db-data` from the NFS server to store database files.

Expected Outcomes:

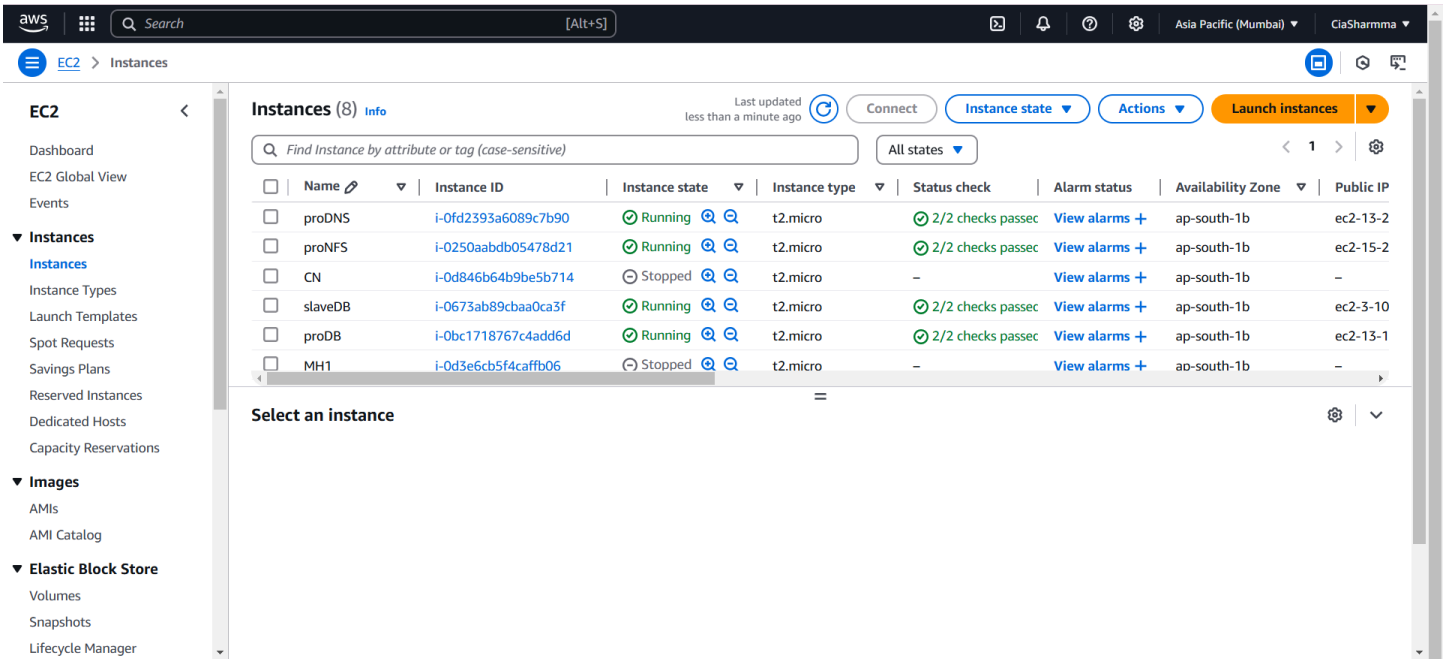
- A fully functional multi-server environment.
- Centralized storage management with NFS.
- Scalable and redundant database system using Master-Slave replication.
- Seamless WordPress hosting with persistent storage.

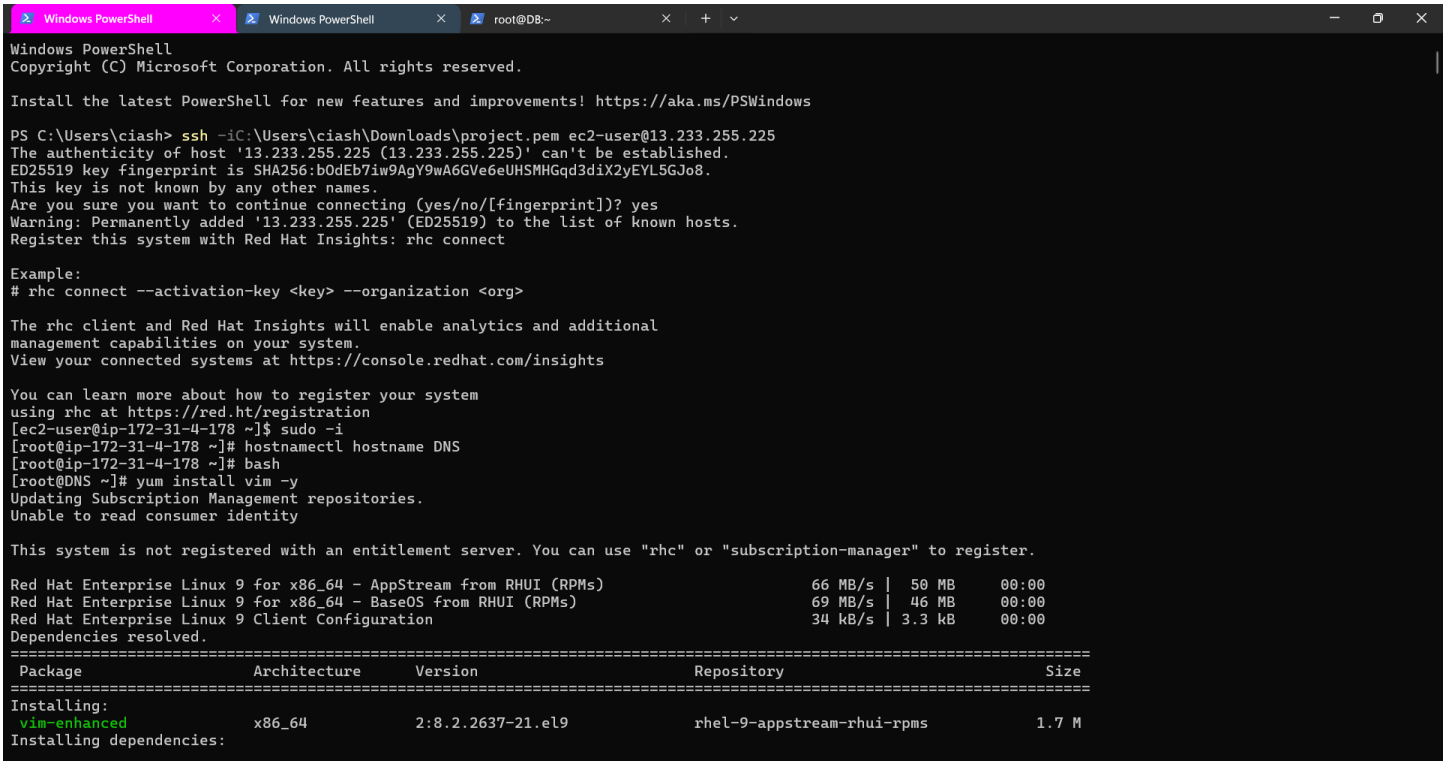
Note: Students need to research and implement the configuration steps for each component.

project 1:

1)creating 4 instances one for DNS configuration ,wordpress and webserver configuration ,one for NFS ,and 2 for replication of database

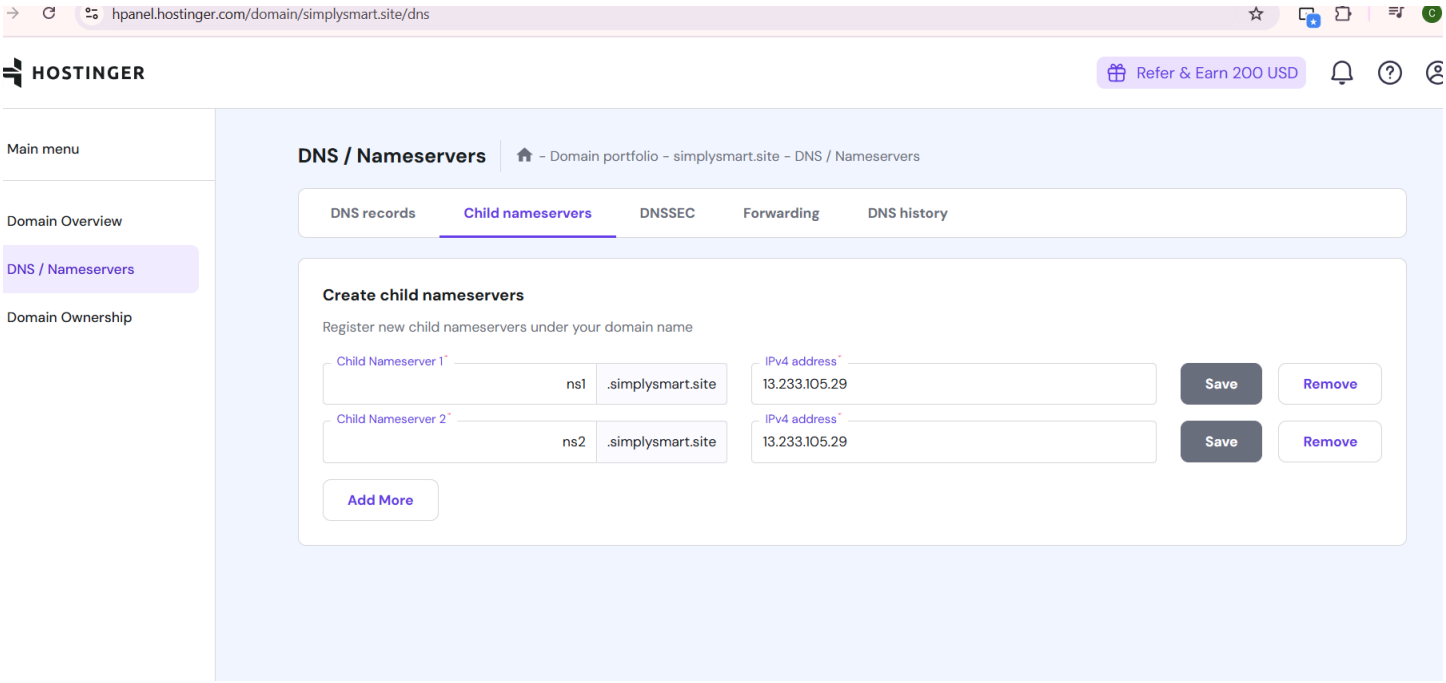
->logging into all the instance with basic service and packages like vim,wget,bash-completion and firewall





now first configuring the DNS for which zone files needs to be updated as well as named configuration files coming along bind and bind-utils packages and updating DNS server IP’s

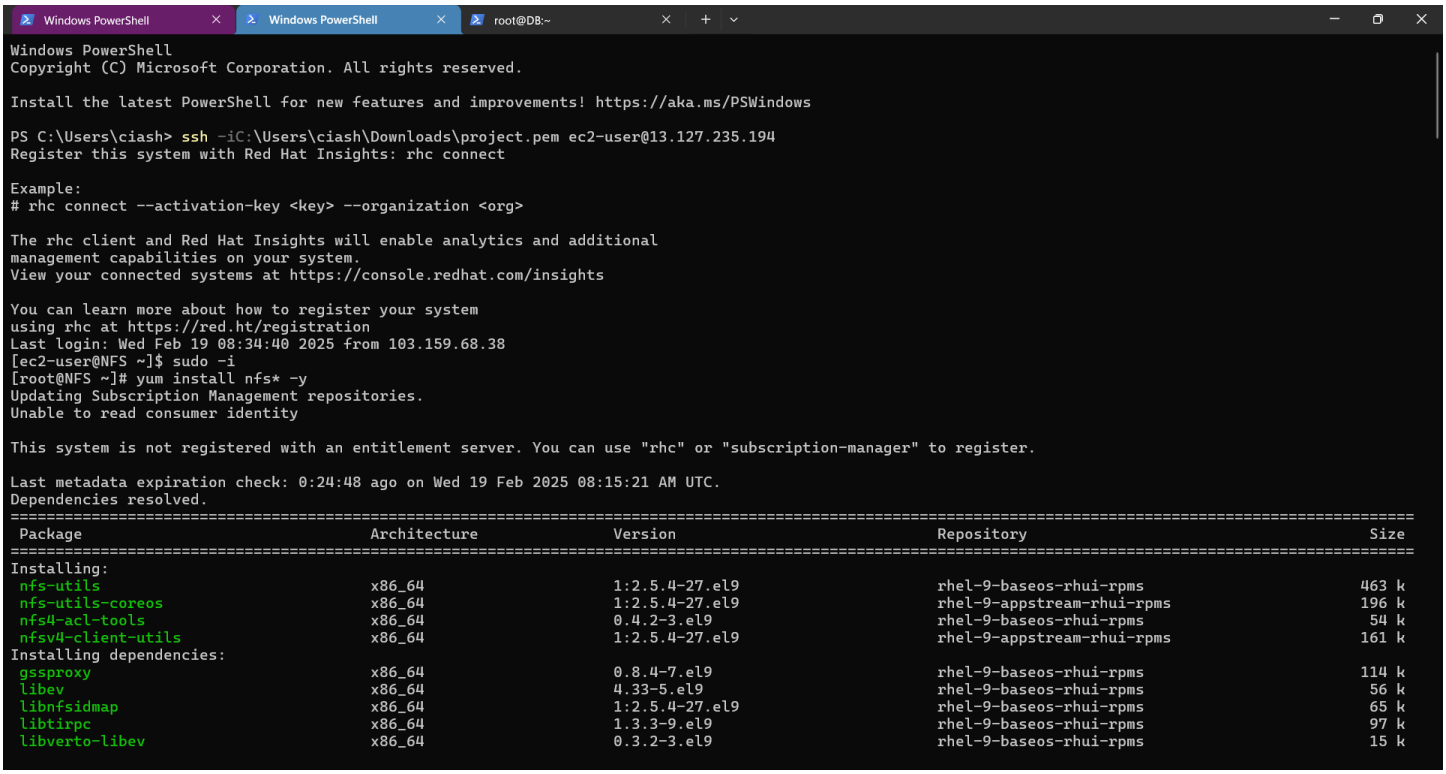
at Childservers of hostinger



now trying to check by ping or nslookup or simply seraching at brower where our site is surfing or not ,

Now,coming to NFS server by we need to establish NFS between /wp-data and /var/www/html

first in NFS we need to install and enable NFS service them make a directory by



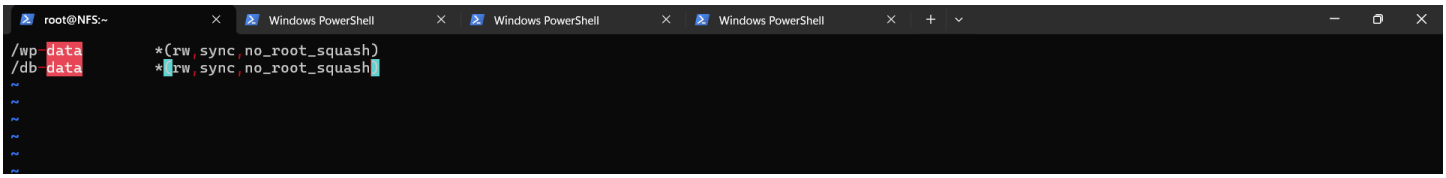
mkdir /wp-data and by changing it's to NFS we use command 'semanage fcontext -a -t nfs_t /wp-data'

then restorecon -R /wp-data/

ls -ldZ /wp-data/

systemctl restart nfs-server.service

```
Last login: Wed Feb 26 14:38:42 2025 from 157.38.79.10
[ec2-user@NFS ~]$ sudo -i
[root@NFS ~]# ls -ldZ /wp-data
drwxr-xr-x. 5 48 48 unconfined_u:object_r:nfs_t:s0 4096 Feb 26 15:33 /wp-data
```



vim /etc/exports

now on your DNS/webserver server update /etc/fstab with the ip of NFS server and give the mount point of webserver as /var/www/html

now system deamon reload and ‘mount -a’ ,hence NFS is eastablished doing the same for DATABASE as well

create db-data

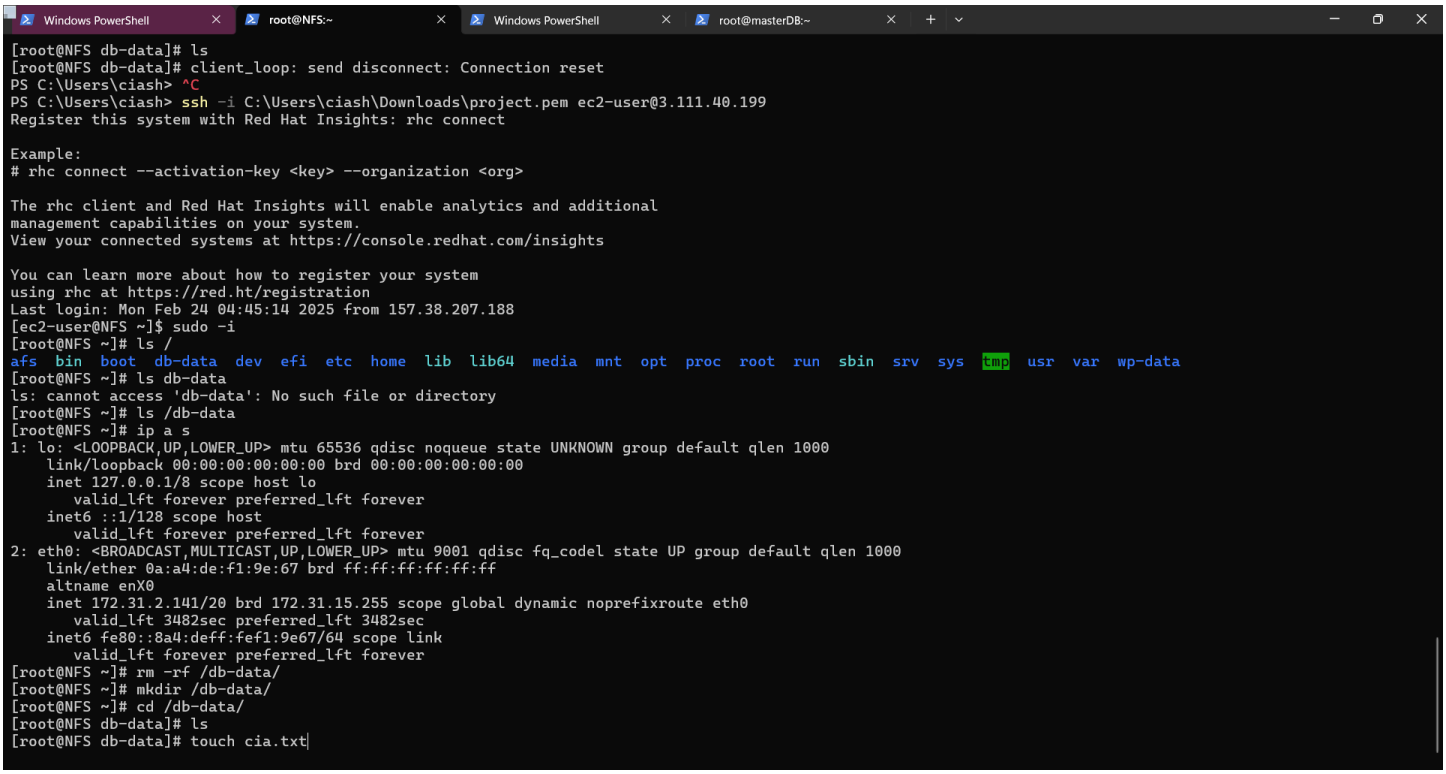
mkdir /db-data

semanage fcontext -a -t nfs_t /db-data

restorecon -R /db-data/

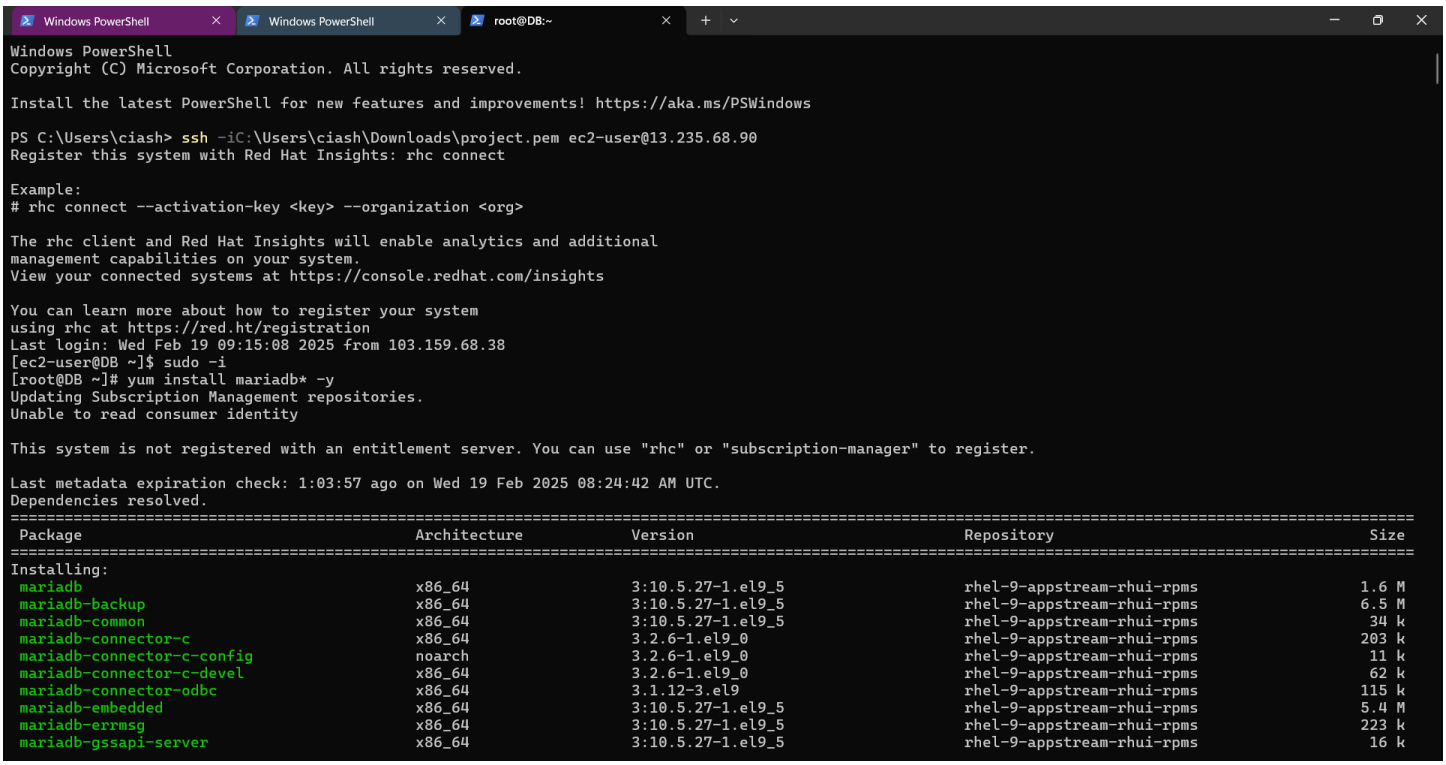
ls -ldZ /db-data/

systemctl restart nfs-server.service



proof that both the files as Sycn with NFS by creating a file EX cia.txt on any one server displays on another

Now, Creating a master slave or replication database first log in into both master and slave data base install basic packages + mariabd*



now log in into the master server and create a Databse as wp_DB and a usr named as replicator have given all privilesges on master data base

*Here i’ve created a DB as wp_DB and ‘create user ’replicator’@’%’ identified by ‘c’;

then Flush privileges;

and show master status;

MariaDB [(none)]> show master status;

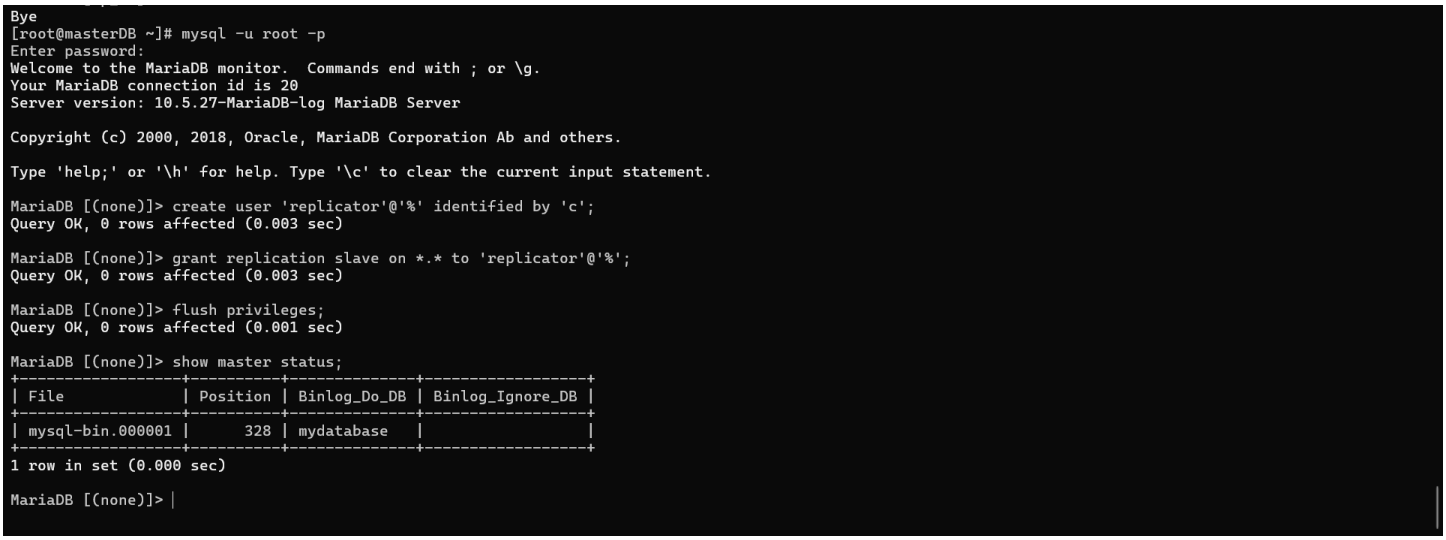
+------------------+----------+--------------+------------------+

| File | Position | Binlog_Do_DB | Binlog_Ignore_DB |

+------------------+----------+--------------+------------------+

| mysql-bin.000001 | 328 | mydatabase | |

+------------------+----------+--------------+------------------+

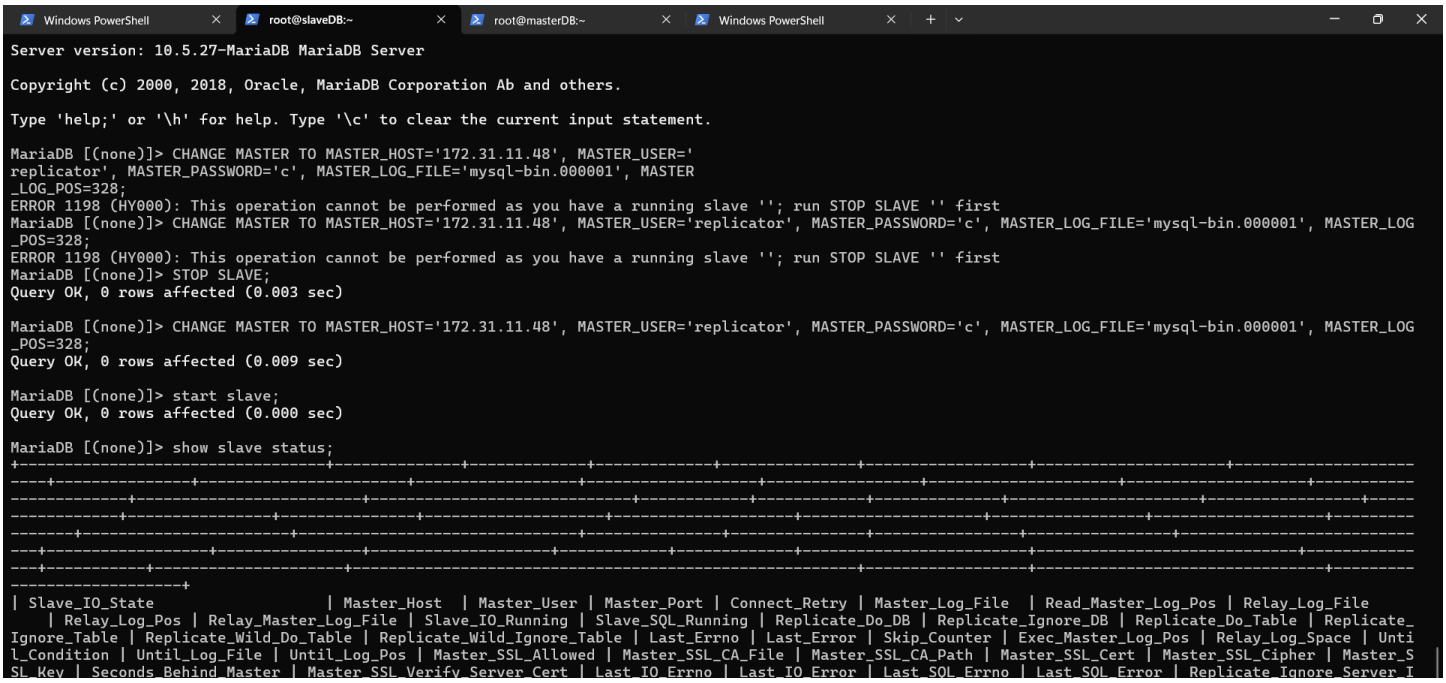


once the work is done in master server come to slave server enable here the mariadb service and log into mariadb via authorized name and master server’s private IP

mysql -u replicaotor -h (IP) -p

password - c

then give all reqired details in the maria



once the command run start slave;

and check is slave_IO_State and SQL is yes thus the slaver server has successfully established

Now, the last step for the project is remaining which is to create a master DB user and taking it as a details for the wordpress service

so ,create a user as wp_user and give it the privileges on wp_DB and access it as in DNS server/wordpress server

at webserver install mariadb*

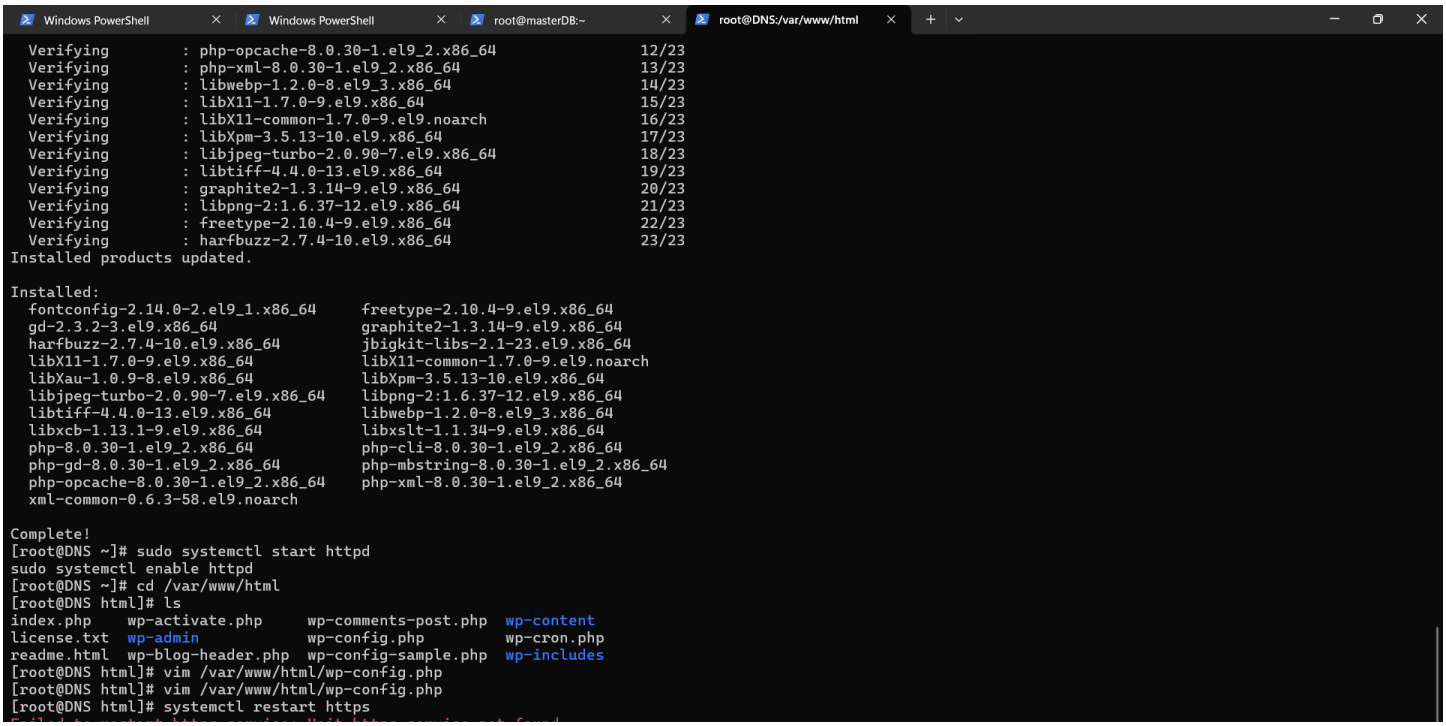
and login by mysql -u wp_user -h (master server pvt IP) -p and password

check databases is available ;

download the wordpress via ‘wget https://wordpress.org/latest.tar.gz

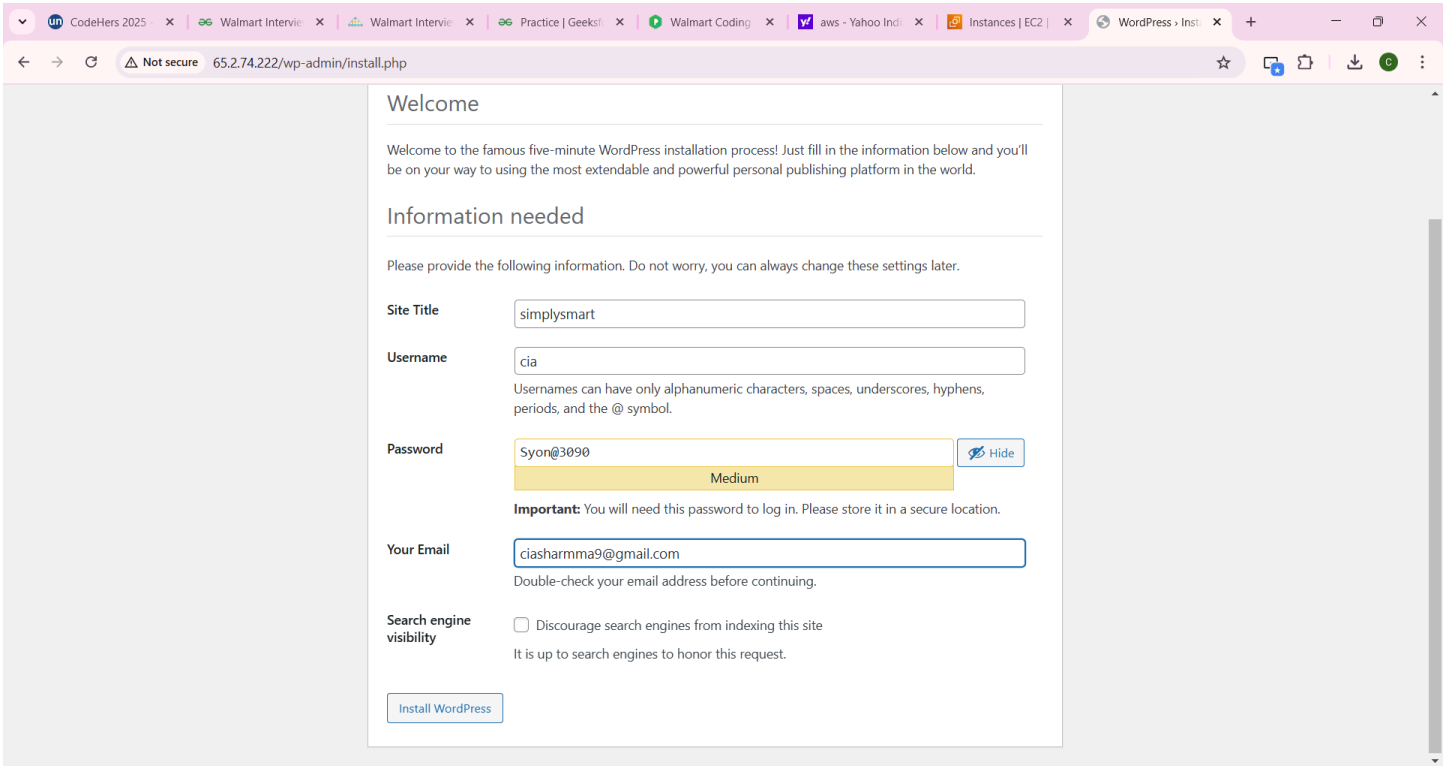
unzip the file and copy it’s content to /var/www/html file

,

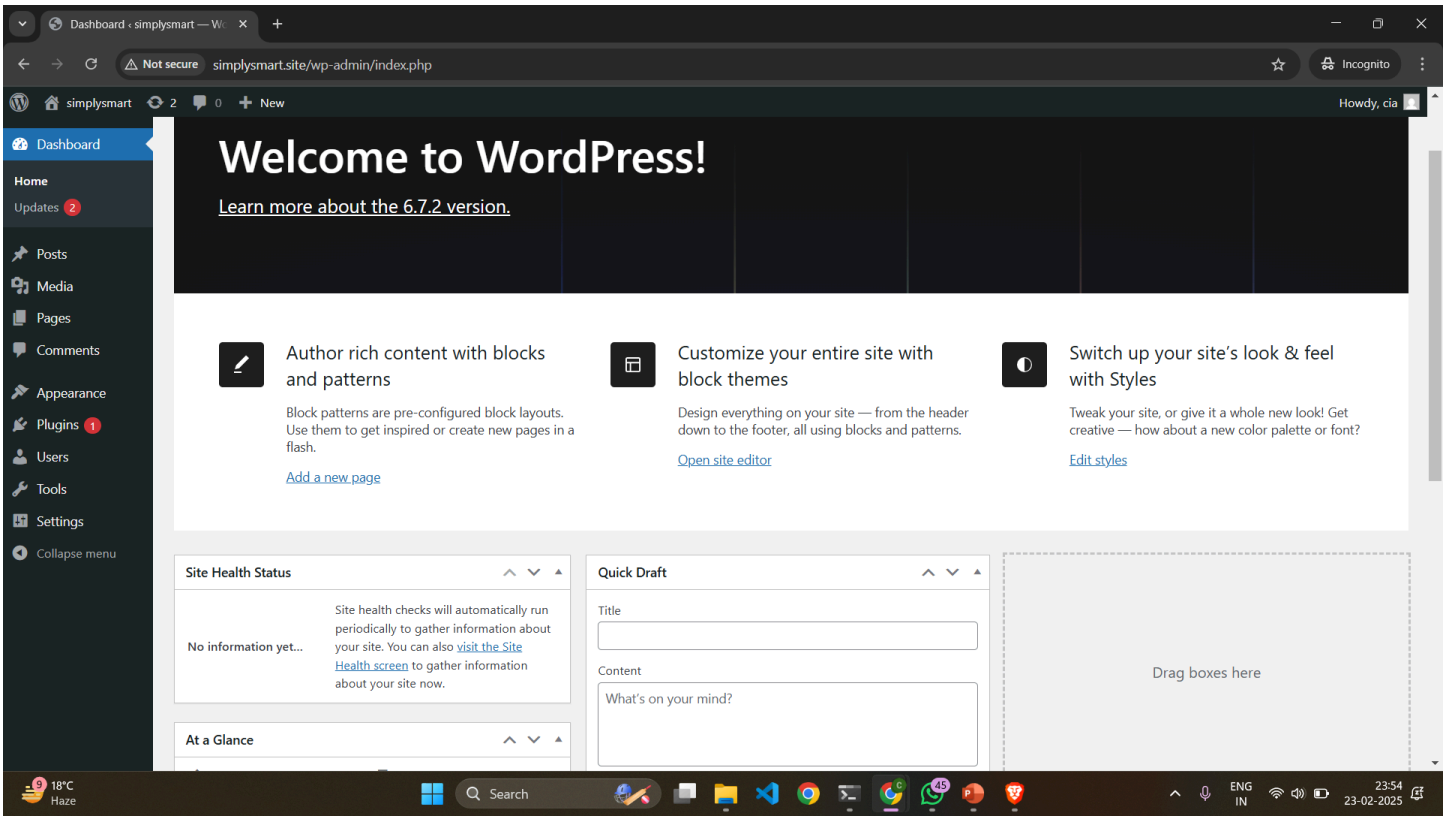


once done make modify wp-sample-config.php file to wp-config.php and open it

make all the required entries and then restart the service and check ip on google or ‘simplysmart.site ‘



fill in the details and then the wordpress server is done



Thus,

All the functionality is working properly & we get a centralized storage managment along with replication as well as NFS feature