PROJECT-3

DEPLOYING WORDPRESS WEB APPLICATION USING DOCKER IN AMAZON WEB SERVICES

INTRODUCTION :

WordPress is the easiest way to manage and create content. Its flexibility is loved by authors: with a couple of plugins, you can do everything from hosting a cute kitten's photo gallery to hosting an e-commerce site.

We'll deploy **WordPress** via **docker-composer** onto the AWS EC2 instance (**t2.micro**) and access it with a domain name defined in **Route53**.

We'll try to use as many AWS-managed services as we can to be able to offload boring and dangerous tasks.

PRE-REQUISITES :

- Amazon Web Services
- Account GitBash Tool
- GitHub Account

Steps to creating the infrastructure in this pipeline/module :

- Creating and launching an EC2 Instance with AMI – Amazon
- Linux 2 Installing GIT, Docker, and related repos
- Creating Docker images with help of YAML
- scripting Creating EIP for launching the output of the project statically

AWS-Amazon web services :

Amazon Web Services (AWS) is the world's most comprehensive and broadly adopted cloud platform, offering over 200 fully featured services from data centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.

Wordpress :

WordPress is a free and open-source content management system written in PHP and paired with a MySQL or MariaDB database with supported HTTPS. Features include a plugin architecture and a template system, referred to within WordPress as Themes. WordPress is a content management system (CMS) that allows you to host and build websites. WordPress contains plugin architecture and a template system, so you can customize any website to fit your business, blog, portfolio.

GitHub :

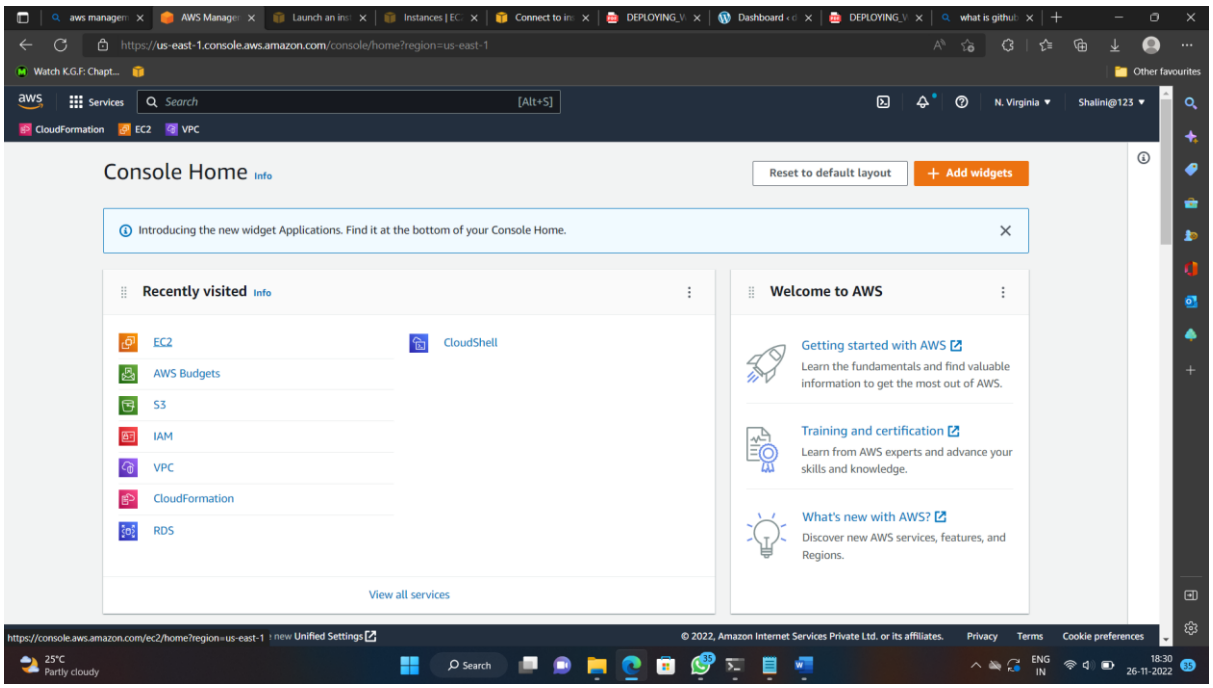
GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. This tutorial teaches you GitHub essentials like repositories, branches, commits, and pull requests.

Gitash/Terminal :

Git Bash is an application for Microsoft Windows environments which provides an emulation layer for a Git command line experience. Bash is an acronym for Bourne Again Shell. A shell is a terminal application used to interface with an operating system through written commands.

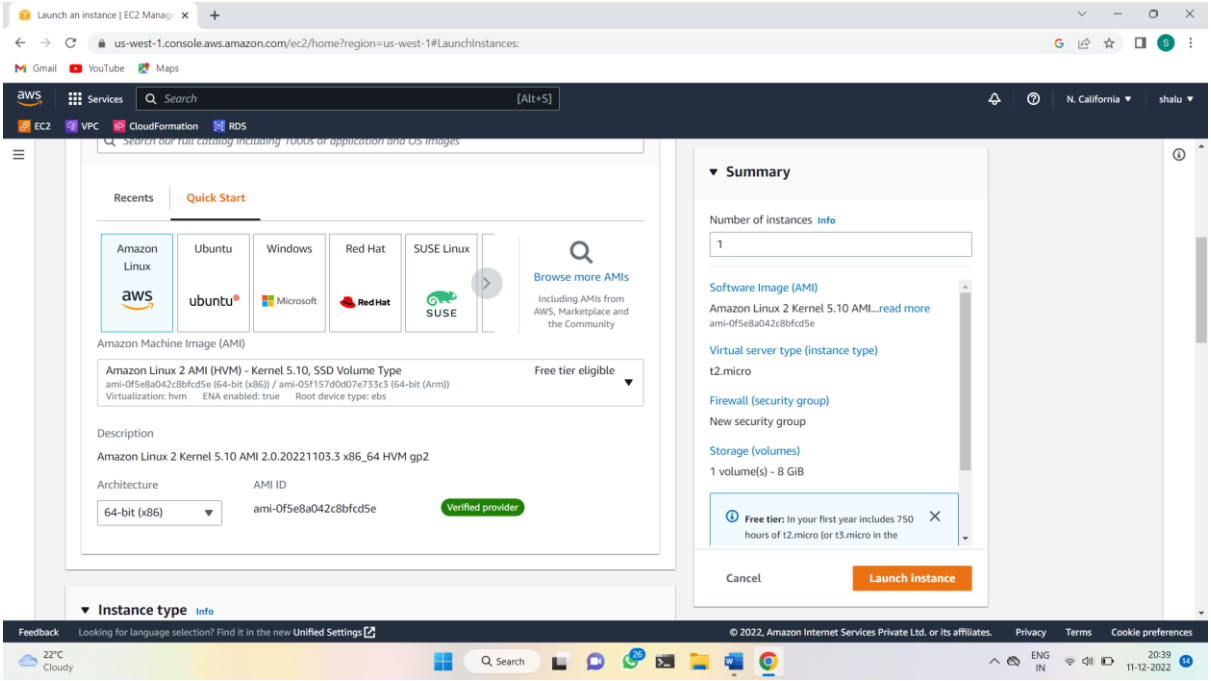
Module – 1: Now Creating and launching an Amazon Linux EC2 instance

1. Go to aws console click on ec2.



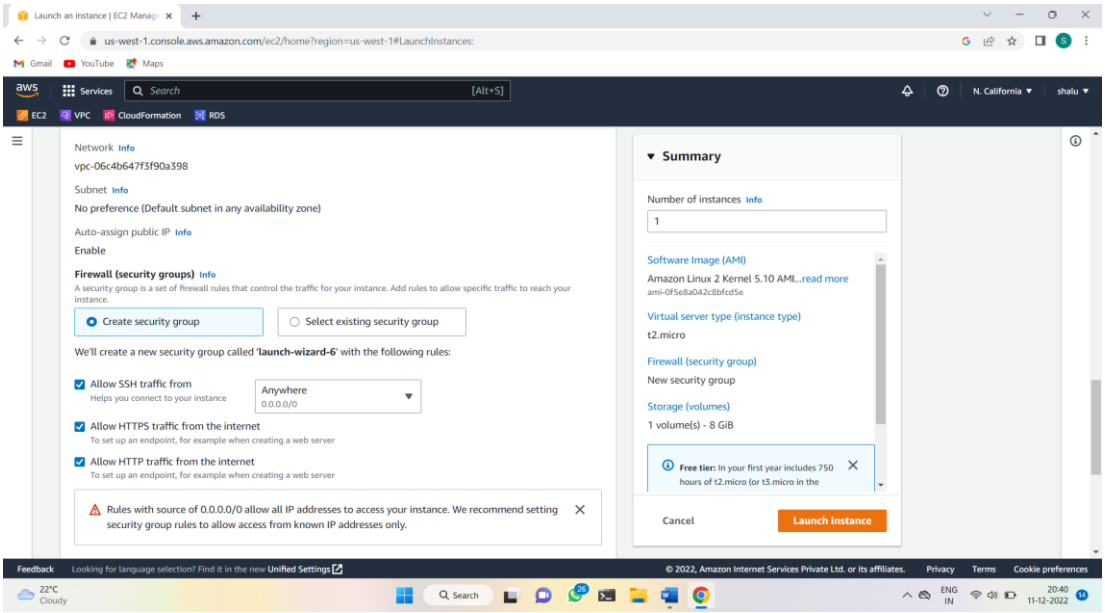
2. Choosing an AMI – Amazon Linux 2 AMI:

1. Select the amazon linux mechine

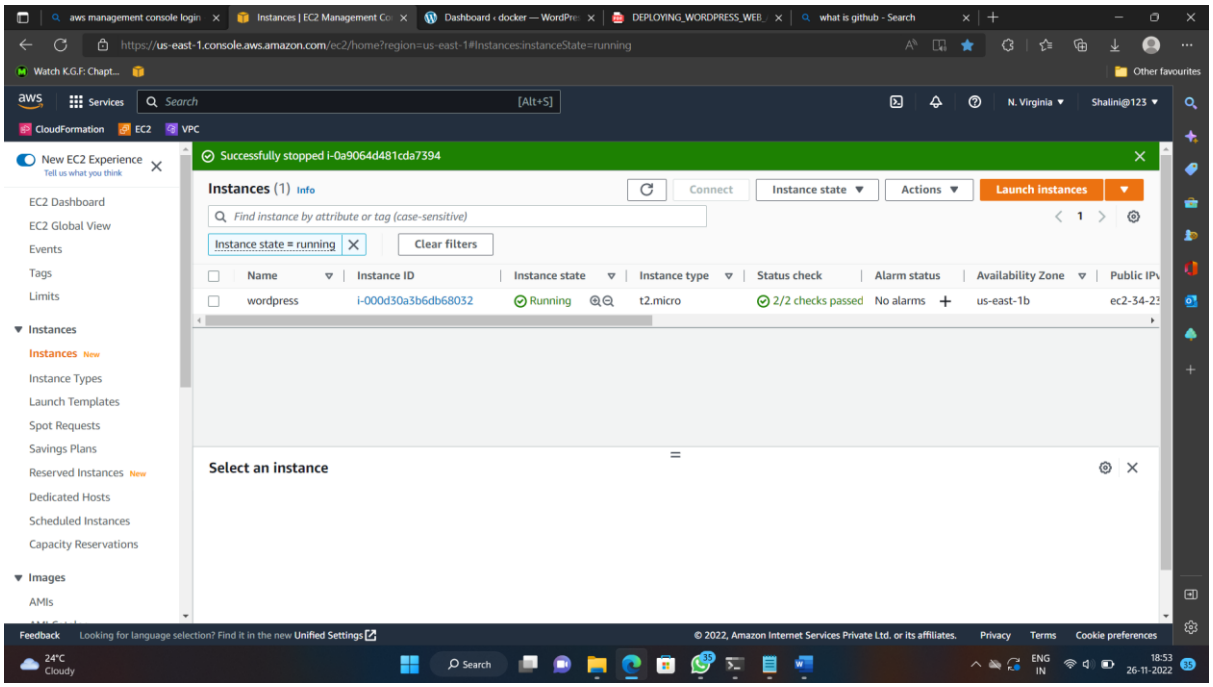


Configure the Security Group

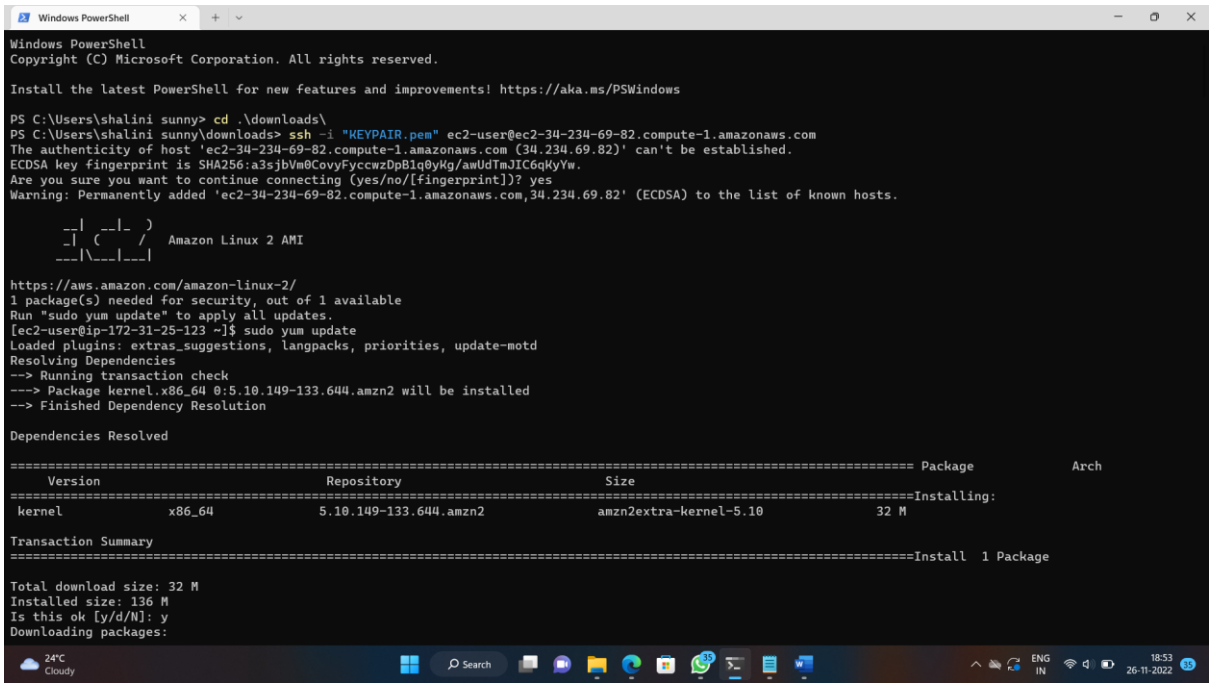
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Here, I am assigning the port numbers for the inbound rule SSH-22, HTTP-80, HTTPS-443.



Finally, Instance launched a WordPress-Server Instance



2. b y using ssh command we are accessing the command line interface in terminal.



Module – 2: Installing GIT, Docker, and related repos

1. we have to install the git and docker in our instance by using below commands.
   - **sudo yum -y install git**
   - **sudo yum -y install docker**

```
Windows PowerShell

Installed:
  kernel.x86_64 0:5.10.149-133.644.amzn2

Complete!
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$ sudo yum -y install git
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package git.x86_64 0:2.37.1-1.amzn2.0.1 will be installed
--> Processing Dependency: perl-Git = 2.37.1-1.amzn2.0.1 for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core-doc = 2.37.1-1.amzn2.0.1 for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core = 2.37.1-1.amzn2.0.1 for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Term::ReadKey) for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git::I18N) for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git) for package: git-2.37.1-1.amzn2.0.1.x86_64
--> Running transaction check
---> Package git-core.x86_64 0:2.37.1-1.amzn2.0.1 will be installed
---> Package git-core-doc.noarch 0:2.37.1-1.amzn2.0.1 will be installed
---> Package perl-Git.noarch 0:2.37.1-1.amzn2.0.1 will be installed
--> Processing Dependency: perl(Error) for package: perl-Git-2.37.1-1.amzn2.0.1.noarch
---> Package perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2 will be installed
--> Running transaction check
---> Package perl-Error.noarch 1:0.17020-2.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================ Package                         Arch
                 Version                          Repository                Size
================================================================================================================Installing:
 git                          x86_64              2.37.1-1.amzn2.0.1                amzn2-core                71 k
Installing for dependencies:
 git-core                     x86_64              2.37.1-1.amzn2.0.1                amzn2-core               6.4 M
 git-core-doc                 noarch              2.37.1-1.amzn2.0.1                amzn2-core               2.8 M
 perl-Error                   noarch              1:0.17020-2.amzn2                 amzn2-core                32 k
 perl-Git                     noarch              2.37.1-1.amzn2.0.1                amzn2-core                46 k
```

```
Windows PowerShell

Installed:
  git.x86_64 0:2.37.1-1.amzn2.0.1

Dependency Installed:
  git-core.x86_64 0:2.37.1-1.amzn2.0.1 git-core-doc.noarch 0:2.37.1-1.amzn2.0.1    perl-Error.noarch 1:0.17020-2.amzn2
  perl-Git.noarch 0:2.37.1-1.amzn2.0.1 perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2

Complete!
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$
[ec2-user@ip-172-31-25-123 ~]$ sudo yum -y install docker
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package docker.x86_64 0:20.10.17-1.amzn2.0.1 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.17-1.amzn2.0.1.x86_64
--> Processing Dependency: libcgroup >= 0.40.rc1-5.15 for package: docker-20.10.17-1.amzn2.0.1.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.17-1.amzn2.0.1.x86_64
--> Processing Dependency: pigz for package: docker-20.10.17-1.amzn2.0.1.x86_64
--> Running transaction check
---> Package containerd.x86_64 0:1.6.6-1.amzn2.0.2 will be installed
---> Package libcgroup.x86_64 0:0.41-21.amzn2 will be installed
---> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
---> Package runc.x86_64 0:1.1.3-1.amzn2.0.2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================ Package                         Arch
           Version                          Repository                Size
================================================================================================================Installing:
 docker                       x86_64              20.10.17-1.amzn2.0.1              amzn2extra-docker         39 M
Installing for dependencies:
 containerd                   x86_64              1.6.6-1.amzn2.0.2                 amzn2extra-docker         27 M
 libcgroup                    x86_64              0.41-21.amzn2                     amzn2-core                66 k
 pigz                         x86_64              2.3.4-1.amzn2.0.1                 amzn2-core                81 k
 runc                         x86_64              1.1.3-1.amzn2.0.2                 amzn2extra-docker        2.9 M

Transaction Summary
```

2. give the permission to add the limited linux user account to be docker group by using below commands start the docker & configure the docker.

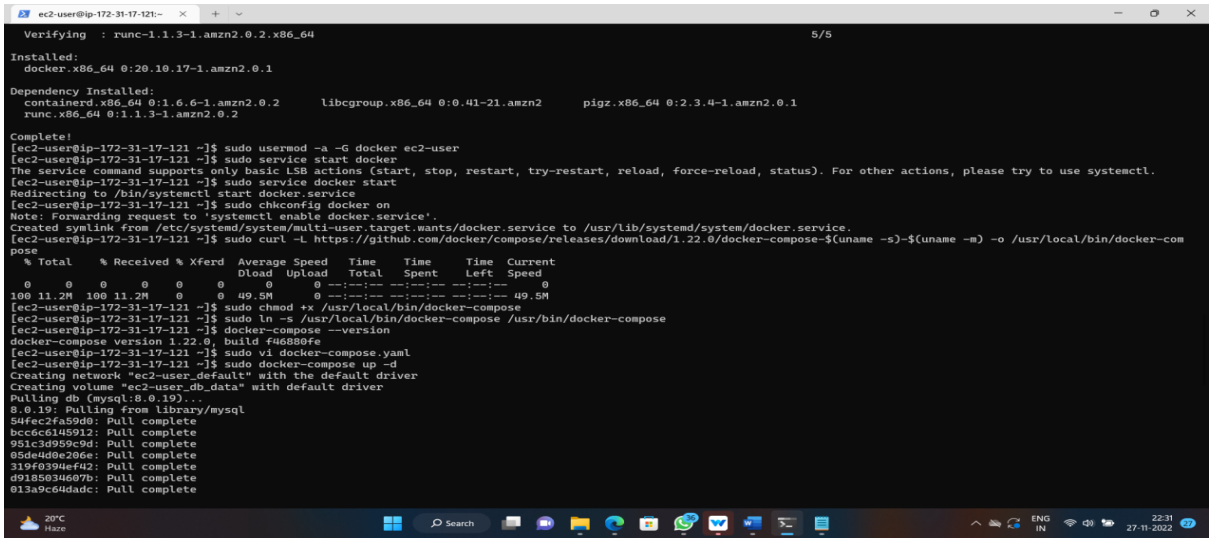
- **sudo usermod -a -G docker ec2-user**

3. to start the docker service the command

- **sudo service docker start**

4. to get docker service up after reboot run below command

➢ **sudo chkconfig docker on**



- Install Docker Compose

1. Download the latest version of Docker Compose (Incommand to download the current stable release of Docker Compose by using below command.

sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) - o /usr/local/bin/docker-compose

2. Apply excutable permissions to the binary.

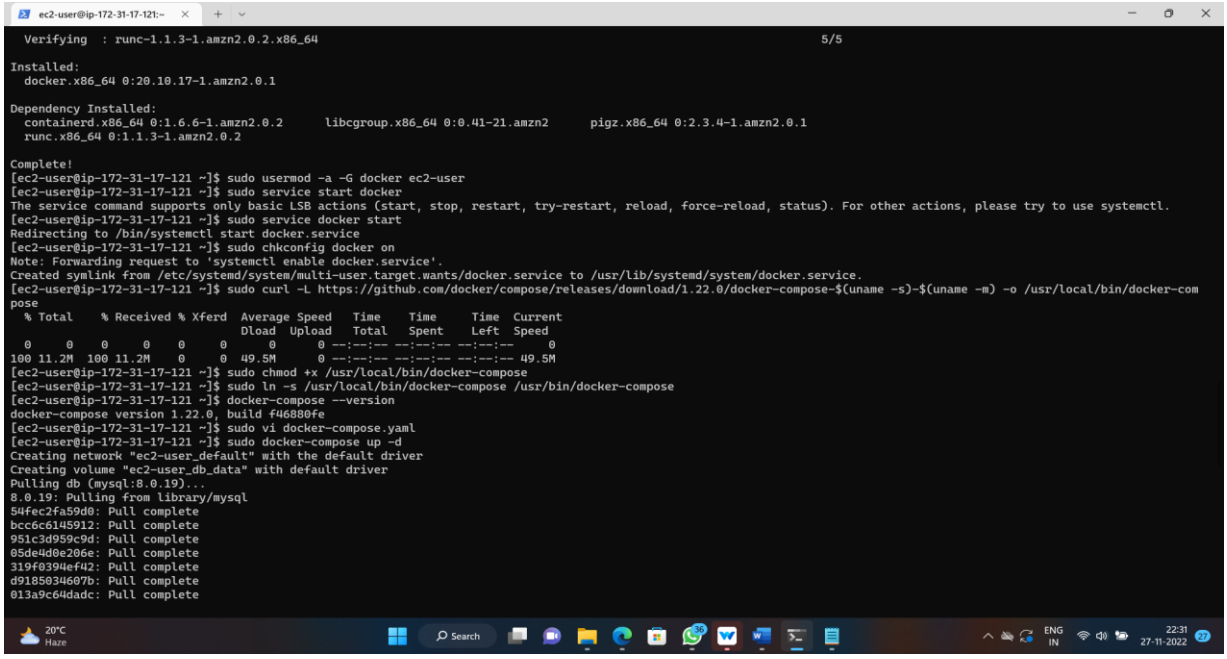
➢ **Sudo usermod -a -G ec2-user**

3. Create a symbolic link.

➢ **ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose**

4. to check the installed docker-compose verion.

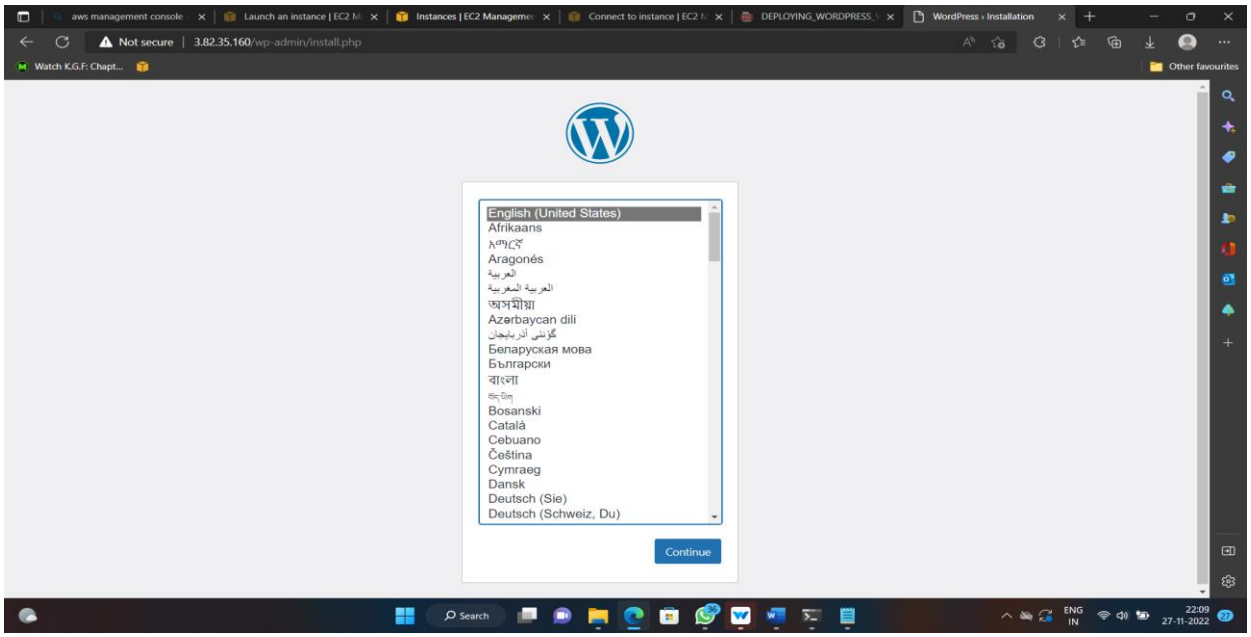
➢ **docker-compose --version**



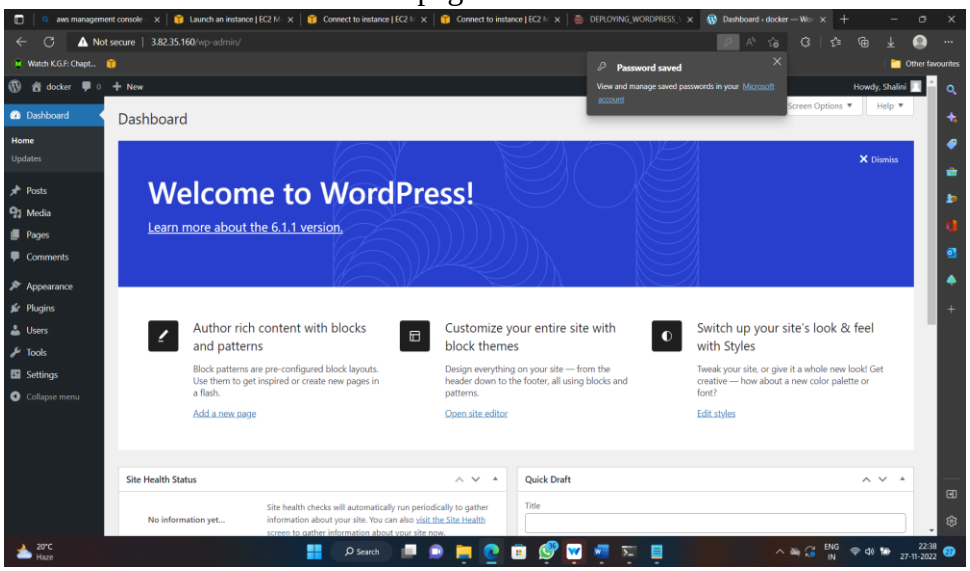
```
ec2-user@ip-172-31-17-121:~
  Verifying  : runc-1.1.3-1.amzn2.0.2.x86_64                                    5/5

Installed:
  docker.x86_64 0:20.10.17-1.amzn2.0.1

Dependency Installed:
  containerd.x86_64 0:1.6.6-1.amzn2.0.2      libcgroup.x86_64 0:0.41-21.amzn2      pigz.x86_64 0:2.3.4-1.amzn2.0.1
  runc.x86_64 0:1.1.3-1.amzn2.0.2

Complete!
[ec2-user@ip-172-31-17-121 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-17-121 ~]$ sudo service start docker
The service command supports only basic LSB actions (start, stop, restart, try-restart, reload, force-reload, status). For other actions, please try to use systemctl.
[ec2-user@ip-172-31-17-121 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-17-121 ~]$ sudo chkconfig docker on
Note: Forwarding request to 'systemctl enable docker.service'.
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-17-121 ~]$ sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-com
pose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 11.2M  100 11.2M    0     0  49.5M      0 --:--:-- --:--:-- --:--:-- 49.5M
[ec2-user@ip-172-31-17-121 ~]$ sudo chmod +x /usr/local/bin/docker-compose
[ec2-user@ip-172-31-17-121 ~]$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
[ec2-user@ip-172-31-17-121 ~]$ docker-compose --version
docker-compose version 1.22.0, build f46880fe
[ec2-user@ip-172-31-17-121 ~]$ sudo vi docker-compose.yaml
[ec2-user@ip-172-31-17-121 ~]$ sudo docker-compose up -d
Creating network "ec2-user_default" with the default driver
Creating volume "ec2-user_db_data" with default driver
Pulling db (mysql:8.0.19)...
8.0.19: Pulling from library/mysql
54fec2fa59d0: Pull complete
bcc6c6145912: Pull complete
951c3d959c9d: Pull complete
05de4d0e206e: Pull complete
319f0394ef42: Pull complete
d9185034607b: Pull complete
013a9c64dadc: Pull complete
```

And then lastly, I had a look to see that this was running correctly.

And then lastly, I had a look to see that this was running correctly.



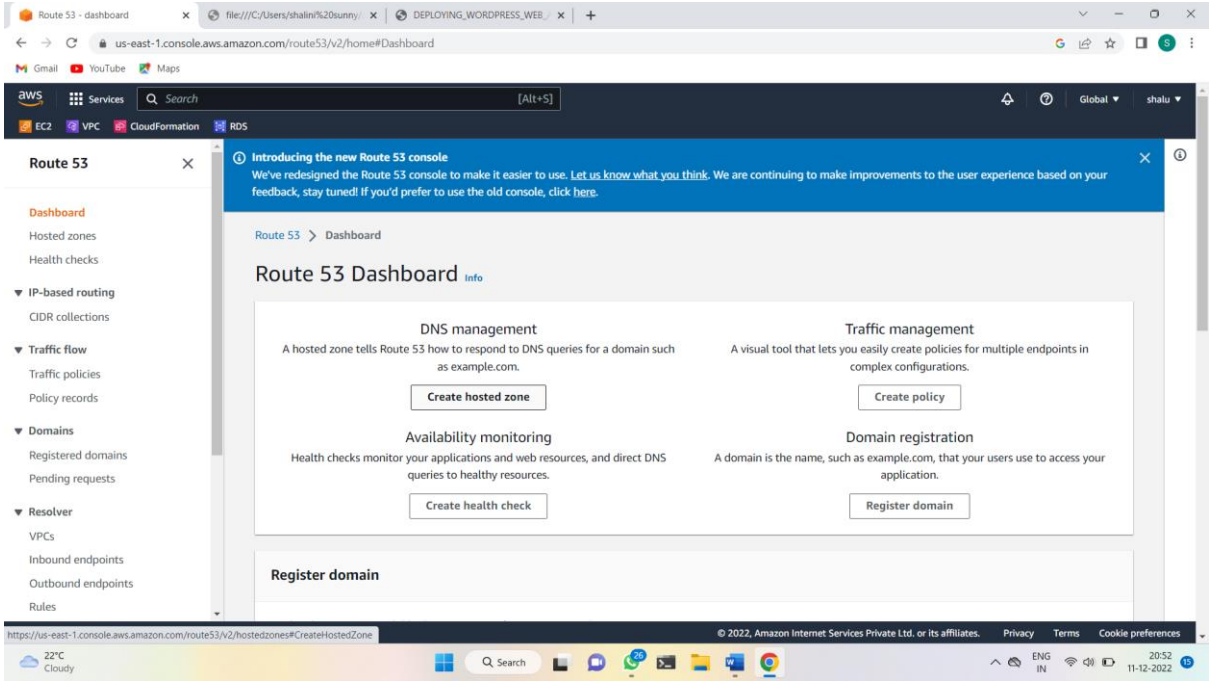
This is the WordPress welcome page.



Route53:

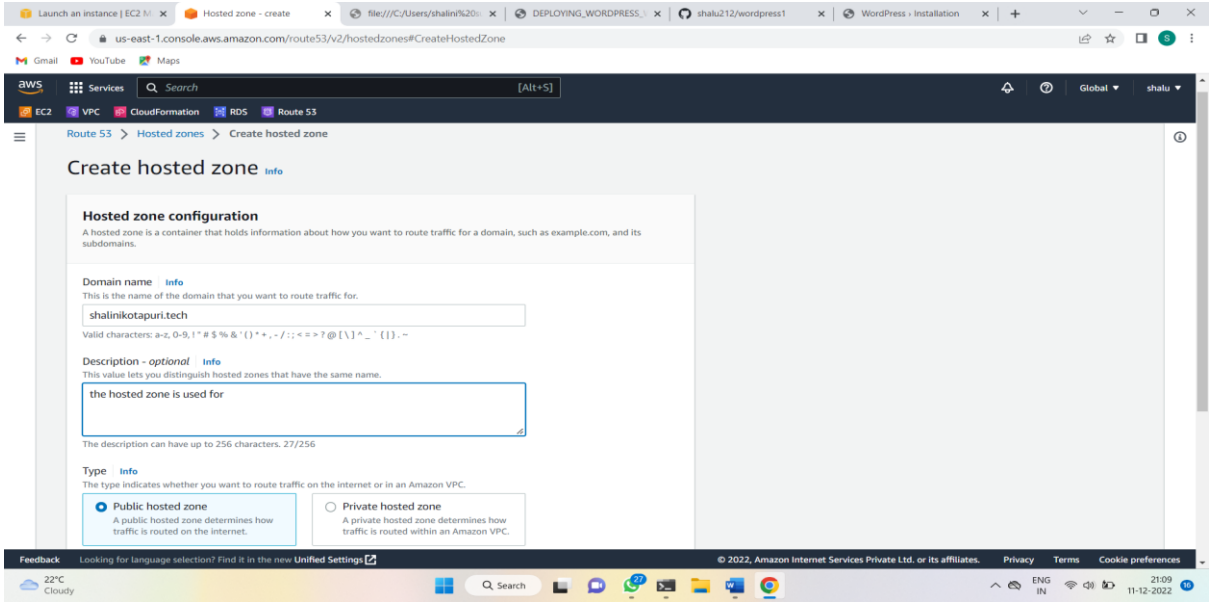
- Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service.
- Route 53 connects user requests to internet applications running on AWS or on-premises.
- GOm to Route53 Dashoard, and click on DNS Management system(create hosted zone).

- Now, click on create hosted zone.



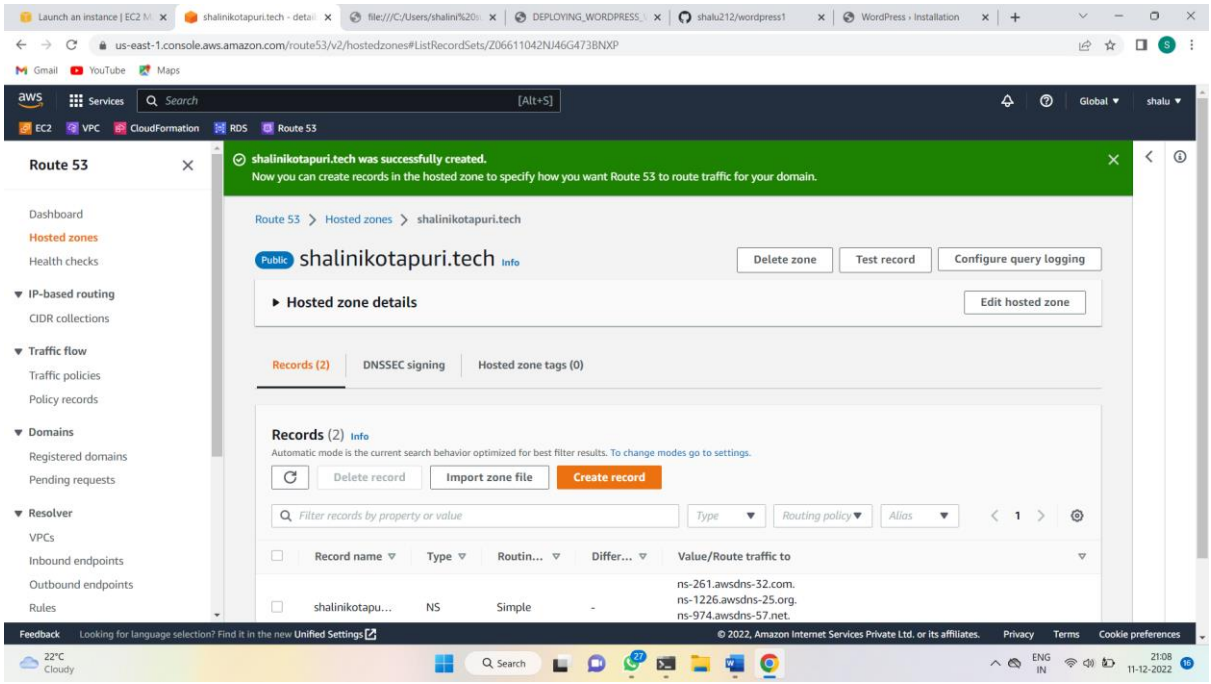
- Now fill up the hosted zone configuration details.

- Provide Domain name,Description ad type should be public hosted zone.

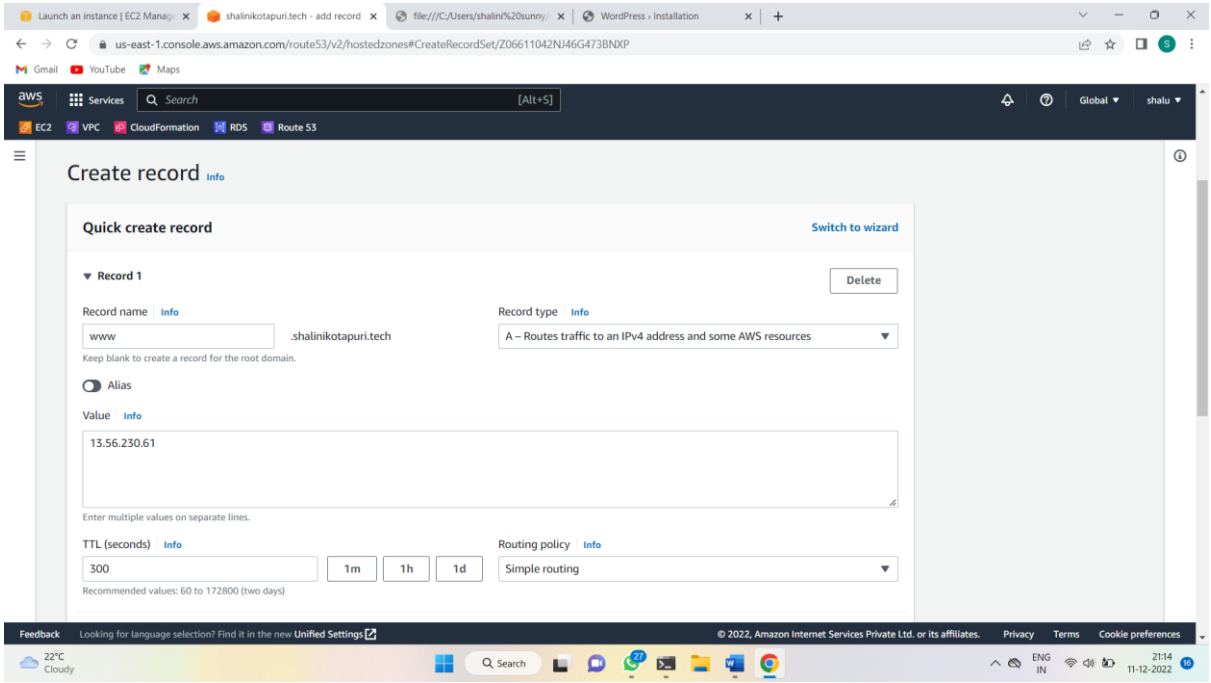


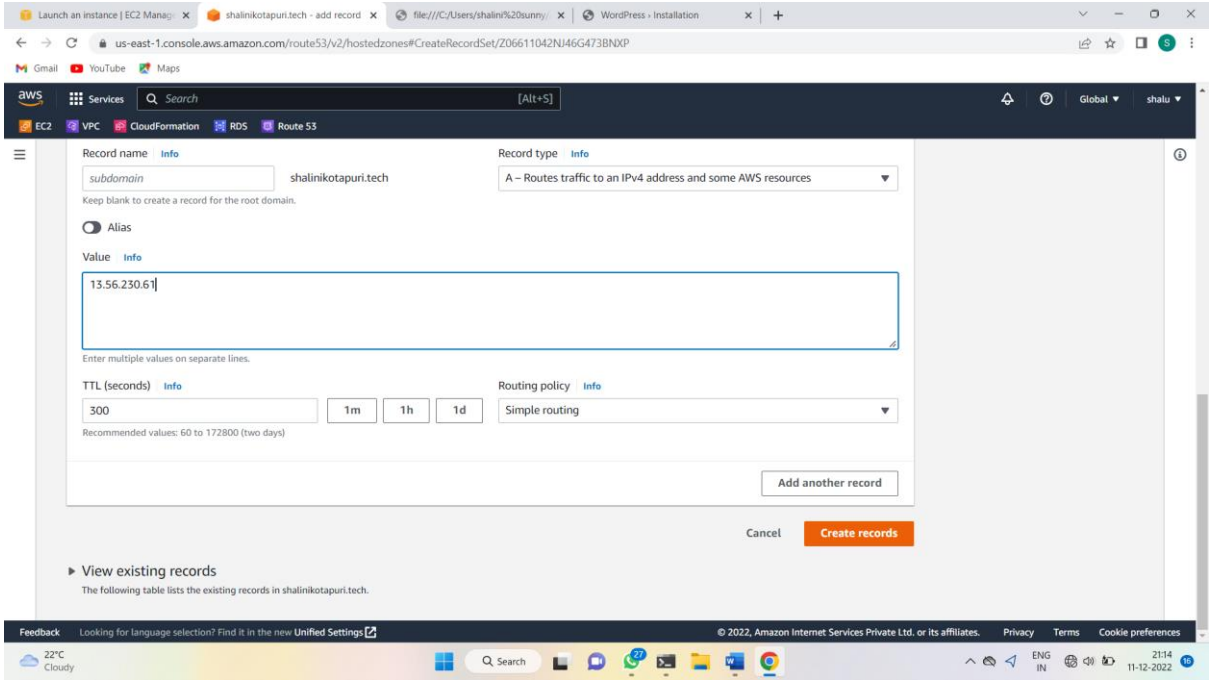
- Click on create Hosted Zone.

- The Hostedzone will be created.

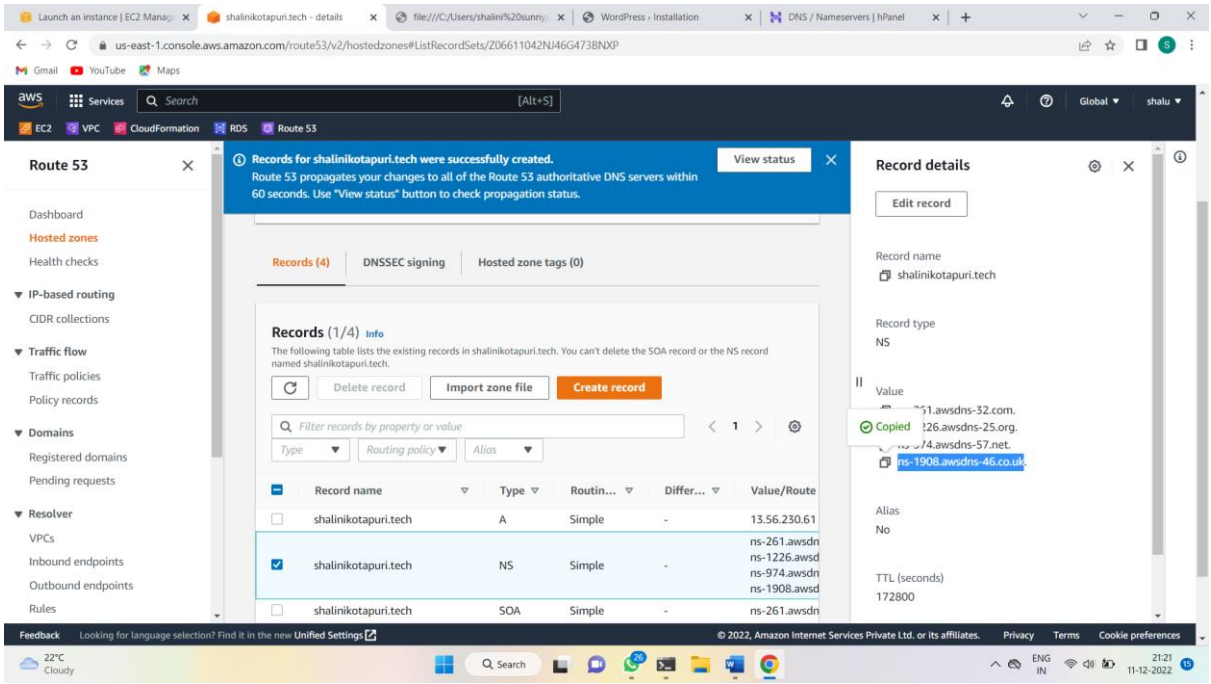


- Now create records as per requirement .
- Click on create records, In record enter record name and record type
- In value , please enter the ip adress of the instance.
- Similarly create another record with another Name.





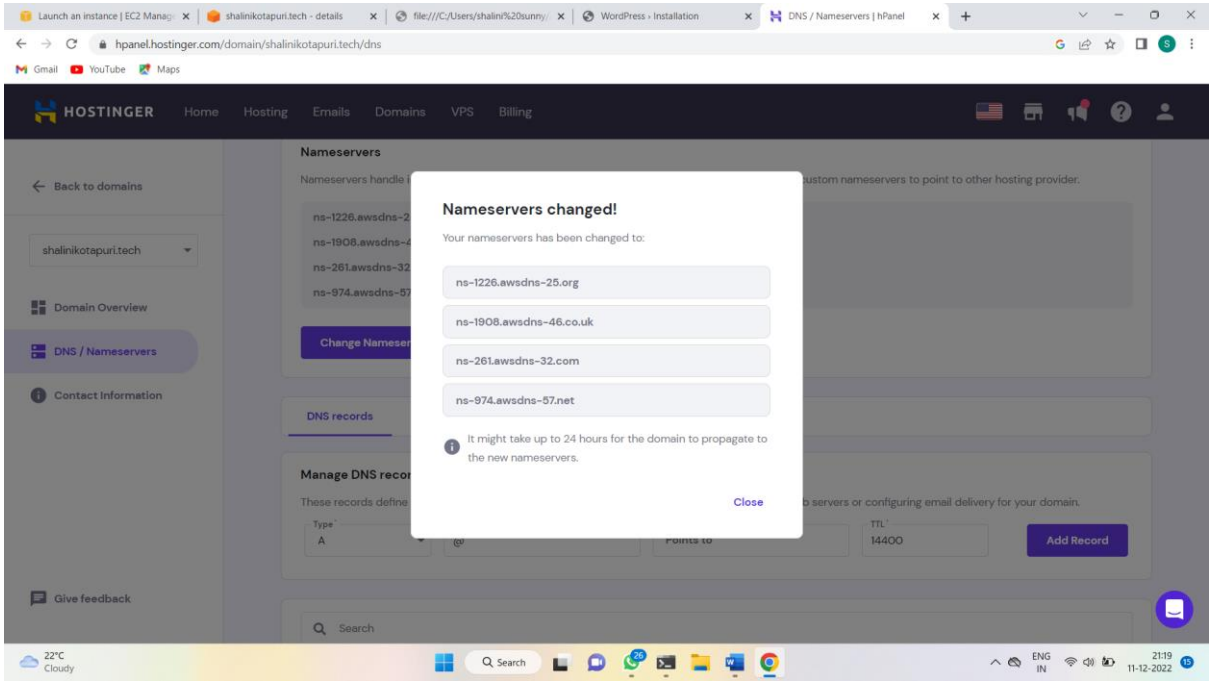
- In records, click on the Naming servers section.
- Copy all the four Naming servers .



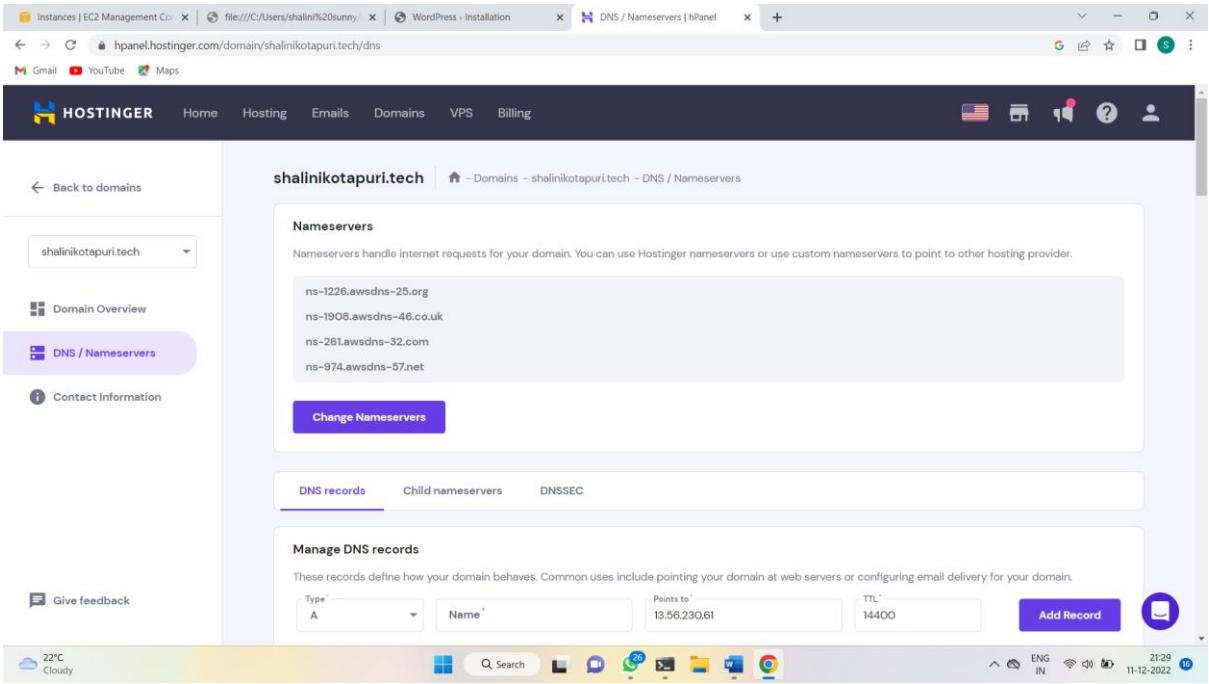
- After copying the name servers, go to Hostinger which is provided your domain name.
- Click on your domain and click on Manage servers

- Now paste the Name servers Here.

- Copy all the four Naming servers .



- Now click on the add record and Paste the IP address of the instance in the place of points to box.



- Now Save the details ad search it with your domain ame in the browser.

- It will take some time to get ready .