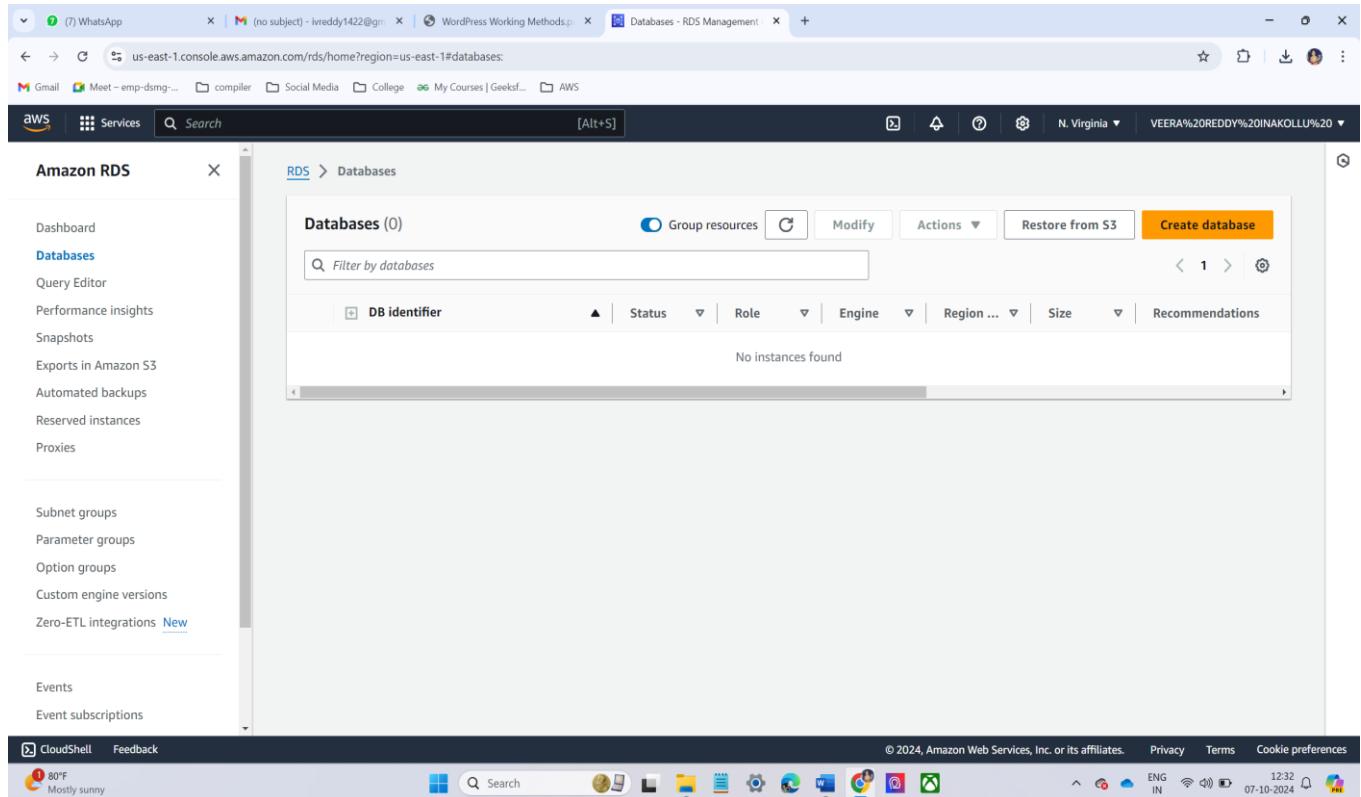


# WordPress Working Methods

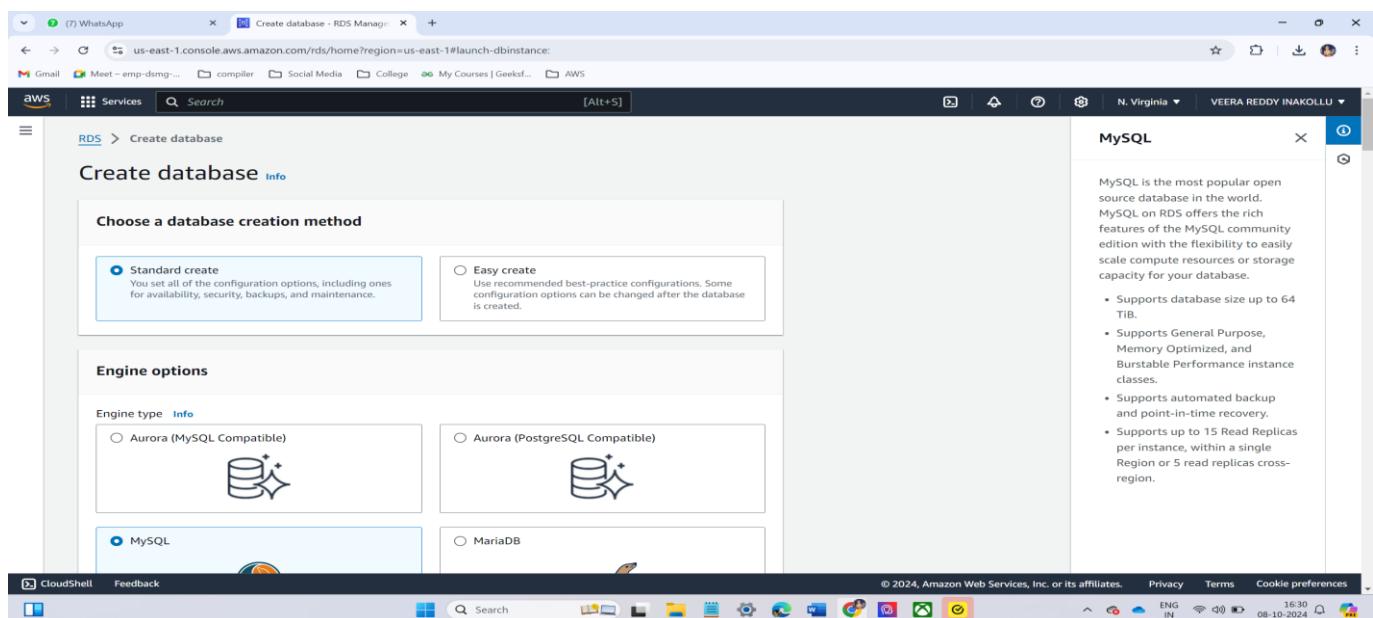
--Veera Reddy Inakollu

## Deploy WordPress web application by using AWS RDS(MYSQL) service (manually).

- ◆ Now first go to aws account and log into aws and search RDS and click on create database.



- ◆ Now select the database creation method (select standard method) because by selecting easy creation methods then its disables free tier template.



- Now select the database engine with version but I select the MySQL database engine because it's a huge usage in real time organizations.

**Engine type**

- Aurora (MySQL Compatible)
- Aurora (PostgreSQL Compatible)
- MySQL**
- MariaDB
- PostgreSQL
- Oracle
- Microsoft SQL Server
- IBM Db2

**MySQL**

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

- Now select the template as free tier and by selecting the free tier it disables the availability zone section options.

**Templates**

Choose a sample template to meet your use case.

- Production
- Dev/Test
- Free tier**

**MySQL**

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

- ◆ Now give some name to your database and give username and password as credentials for your databases.

The screenshot shows the 'Create database - RDS Manager' wizard. In the 'Settings' step, the DB instance identifier is set to 'wordpress'. Under 'Credentials Settings', the master username is also 'wordpress'. The 'Self managed' option for credentials is selected. A note indicates that the master password must be at least 8 characters long and contain uppercase letters, lowercase letters, numbers, and symbols. The MySQL sidebar on the right provides general information about MySQL and its features.

- ◆ Now select the storage type as general purpose (SSD (gp2)) and enter the storage values as maximum (1000gb) and minimum (20gb).

The screenshot shows the 'Storage' configuration step. The storage type is set to 'General Purpose SSD (gp3)'. The allocated storage is set to 20 GiB. A note states that after modifying storage, the DB instance will be in storage-optimization mode. The MySQL sidebar continues to provide information about MySQL's capabilities.

- ◆ Now select your created VPC or select default VPC and it automatically selects the database subnet group.

**Virtual private cloud (VPC) Info**  
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.  
**Default VPC (vpc-07c472b50c6ec4ff6)**  
6 Subnets, 6 Availability Zones

**DB subnet group Info**  
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.  
**default-vpc-07c472b50c6ec4ff6**  
6 Subnets, 6 Availability Zones

**Public access Info**  
 Yes  
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.  
 No  
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

**VPC security group (firewall) Info**  
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.  
 Choose existing  
Choose existing VPC security groups  
 Create new  
Create new VPC security group

**MySQL**  
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

- ◆ Now give the name of the database which you give at the stage of DB instance identifier enter same name here.
- ◆ Now click on the create database button and will create the MySQL database.

**Databases (1)**

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations
wordpress	Available	Instance	MySQL Co...	us-east-1f	db.t4g.mi...	

**Create database**

- Now create the ec2 instance by selecting ec2 instance and launch the instance by selecting Amazon Linux -2 version and giving security group with ssh(22) and http(80).

The screenshot shows the AWS Management Console with the EC2 service selected. The main pane displays the 'Instances (1/1) Info' table. One instance is listed:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
wordpress	i-0a6a7034d49a7faa0	Running	t2.micro	Initializing		us-east-1b

Below the table, the instance details for 'i-0a6a7034d49a7faa0 (wordpress)' are shown, including its Public IPv4 address (52.205.210.182) and Private IPv4 address (172.31.32.40).

- Now connect the virtual server through the git bash as shown in fig.

```

Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jawah\cd Socialprachar
C:\Users\jawah\Socialprachar>cd 'veera reddy'
C:\Users\jawah\Socialprachar\veera reddy>cd keys

C:\Users\jawah\Socialprachar\veera reddy>keys>ssh -i "devops.pem" ec2-user@ec2-52-205-210-182.compute-1.amazonaws.com
The authenticity of host 'ec2-52-205-210-182.compute-1.amazonaws.com (60:ff9b::34cd:d2b6)' can't be established.
ED25519 key fingerprint is SHA256:xNxocIR+Pd8UqaxNTaiMdvw/rjR5g0mdRZh//XXIw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-205-210-182.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

      #_
     _###_      Amazon Linux 2
    _##_###_     AL2 End of Life is 2025-06-30.
   _##_ \#/_     A newer version of Amazon Linux is available!
  _##_ \#/_     Amazon Linux 2023, GA and supported until 2028-03-15.
 _##_ \#/_     https://aws.amazon.com/linux/amazon-linux-2023/
[m/]

[ec2-user@ip-172-31-32-40 ~]$ |

```

- ◆ Now update the Linux version by using command <sudo yum update -y> and install MySQL by using the command sudo yum install -y mysql2.

The screenshot shows a terminal window on an Amazon Linux 2 system. It displays the following output:

```

[ec2-user@ip-172-31-32-40 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
--> Package mariadb.x86_64 1:5.5.68-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package           Arch      Version          Repository      Size
=====
Installing:
mariadb          x86_64   1:5.5.68-1.amzn2.0.1    amzn2-core      8.8 M

Transaction Summary
=====
Install 1 Package

Total download size: 8.8 M
Installed size: 49 M
Is this ok [y/d/N]: y
Downloading packages:
mariadb-5.5.68-1.amzn2.0.1.x86_64.rpm | 8.8 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64 1/1
  Verifying  : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64 1/1

Installed:
  mariadb.x86_64 1:5.5.68-1.amzn2.0.1

Complete!
[ec2-user@ip-172-31-32-40 ~]$ 

```

The terminal window has a dark background with white text. The title bar says "ec2-user@ip-172-31-32-40 ~". The bottom right corner shows the date and time as "08-10-2024 16:42".

- ◆ Now to allow create traffic from ec2 instance into rds database for that go to rds service and select your database.
- ◆ Now go inside the created database and go to the security under this option these is security group id click on that.
- ◆ Now access the MySQL database by using the command as “export MYSQL\_HOST <endpoint address>” for that go inside the created database and go to the endpoint address and select and copy the endpoint address
- ◆ Now give the credentials of database by giving the command as “MySQL -user=<username> --password=<password> database=”name”
- ◆ There is another method command to access your database as “MySQL -h <endpoint address> -u <user> -p” press enter button it asks the password to enter it, and press enter button
- ◆ Now create a database user for your WordPress application and give it permission to access the WordPress database. By using these commands as
  - CREATE USER WordPress IDENTIFIED BY ‘WordPress-pass’
  - GRANT ALL PRIVILEGE ON WordPress.\* TO WordPress
  - FLUSH PRIVILEGES
  - EXIT

```
ec2-user@ip-172-31-32-40 ~ | 8.8 MB 00:00:00
=====
Transaction Summary
=====
Install 1 Package

Total download size: 8.8 M
Installed size: 49 M
Is this ok [y/d/N]: y
Downloading packages:
mariadb-5.5.68-1.amzn2.0.1.x86_64.rpm | 8.8 MB 00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64 1/1
  Verifying  : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64 1/1

Installed:
  mariadb.x86_64 1:5.5.68-1.amzn2.0.1

Complete!
[ec2-user@ip-172-31-32-40 ~]$ mysql -h wordpress.cpwk62gwqsg.us-east-1.rds.amazonaws.com -u wordpress -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 27
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> CREATE database wordpress;
Query OK, 1 row affected (0.01 sec)

MySQL [(none)]> CREATE USER 'wordpress'@'localhost' IDENTIFIED BY 'wordpress-pass';
Query OK, 0 rows affected (0.01 sec)

MySQL [(none)]> GRANT ALL PRIVILEGES ON wordpress.* TO 'wordpress'@'localhost';
Query OK, 0 rows affected (0.01 sec)

MySQL [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

MySQL [(none)]> EXIT;
Bye
[ec2-user@ip-172-31-32-40 ~]$
```

- ◆ To host WordPress application need a Apache web server httpd install that by giving command as <sudo yum install httpd -y> and start and enable the httpd service by giving the command as
    - a. “sudo service httpd start (or) sudo systemctl start httpd
    - b. Sudo chkconfig httpd on (or) sudo systemctl enable httpd

```
[ec2-user@ip-172-31-32-40 ~] + | v
Installing : apr-util-bdb-1.6.3-1.amzn2.0.1.x86_64
Installing : httpd-tools-2.4.62-1.amzn2.0.2.x86_64
Installing : httpd-filesystem-2.4.62-1.amzn2.0.2.noarch
Installing : generic-logos-httpd-18.0.0-4.amzn2.noarch
Installing : mailcap-2.1.41-2.amzn2.noarch
Installing : mod_http2-1.15.19-1.amzn2.0.2.x86_64
Installing : httpd-2.4.62-1.amzn2.0.2.x86_64
Verifying  : apr-util-bdb-1.6.3-1.amzn2.0.1.x86_64
Verifying  : httpd-2.4.62-1.amzn2.0.2.x86_64
Verifying  : apr-1.7.2-1.amzn2.x86_64
Verifying  : mod_http2-1.15.19-1.amzn2.0.2.x86_64
Verifying  : apr-util-1.6.3-1.amzn2.0.1.x86_64
Verifying  : mailcap-2.1.41-2.amzn2.noarch
Verifying  : generic-logos-httpd-18.0.0-4.amzn2.noarch
Verifying  : httpd-tools-2.4.62-1.amzn2.0.2.x86_64
Verifying  : httpd-filesystem-2.4.62-1.amzn2.0.2.noarch

Installed:
httpd.x86_64 0:2.4.62-1.amzn2.0.2

Dependency Installed:
apr.x86_64 0:1.7.2-1.amzn2           apr-util.x86_64 0:1.6.3-1.amzn2.0.1     apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1   generic-logos-httpd.noarch 0:18.0.0-4.amzn2
httpd-filesystem.noarch 0:2.4.62-1.amzn2.0.2    httpd-tools.x86_64 0:2.4.62-1.amzn2.0.2   mailcap.noarch 0:2.1.41-2.amzn2          mod_http2.x86_64 0:1.15.19-1.amzn2.0.2

Complete!
[ec2-user@ip-172-31-32-40 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-32-40 ~]$ sudo systemctl enable httpd
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-32-40 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2024-10-08 11:16:22 UTC; 18s ago
     Docs: man:httpd.service(8)
 Main PID: 3467 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
  CGroup: /system.slice/httpd.service
          └─3467 /usr/sbin/httpd -DFOREGROUND
              ├─3468 /usr/sbin/httpd -DFOREGROUND
              ├─3469 /usr/sbin/httpd -DFOREGROUND
              ├─3470 /usr/sbin/httpd -DFOREGROUND
              ├─3471 /usr/sbin/httpd -DFOREGROUND
              └─3472 /usr/sbin/httpd -DFOREGROUND

Oct 08 11:16:22 ip-172-31-32-40.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Oct 08 11:16:22 ip-172-31-32-40.ec2.internal systemd[1]: Started The Apache HTTP Server.
[ec2-user@ip-172-31-32-40 ~]$
```

- ◆ Now go to ec2 instance and copy the public ipv4 it on Google. Browse it and check the official page of httpd is displayed

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

**If you are a member of the general public:**

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

**If you are the website administrator:**

You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.

You are free to use the image below on web sites powered by the Apache HTTP Server:

- ◆ Now go to browse and select a download WordPress and click on proper link and select and copy the address link of WordPress. Download the file and past that along with wget command in gitbash as “wget <address. Link>”
- ◆ It gives the zip file to unzip that file by using a command as “unzip <zip file>”

```

ec2-user@ip-172-31-32-40:~ + ~
inflating: wordpress/wp-admin/js/password-toggle.min.js
inflating: wordpress/wp-admin/js/svg-painter.js
inflating: wordpress/wp-admin/js/link.js
inflating: wordpress/wp-admin/js/custom-header.js
inflating: wordpress/wp-admin/js/widgets.js
inflating: wordpress/wp-admin/js/gallery.js
inflating: wordpress/wp-admin/js/word-count.js
inflating: wordpress/wp-admin/js/accordion.min.js
inflating: wordpress/wp-admin/js/inline-edit-post.min.js
inflating: wordpress/wp-admin/js/customize-widgets.min.js
inflating: wordpress/wp-admin/js/inline-edit-post.js
inflating: wordpress/wp-admin/js/uploads.js
inflating: wordpress/wp-admin/js/media-upload.js
inflating: wordpress/wp-admin/js/media.js
inflating: wordpress/wp-admin/js/editor-expand.min.js
inflating: wordpress/wp-admin/js/media-gallery.min.js
inflating: wordpress/wp-admin/js/common.min.js
inflating: wordpress/wp-admin/js/tags-box.min.js
inflating: wordpress/wp-admin/js/svg-painter.min.js
inflating: wordpress/wp-admin/js/custom-background.js
inflating: wordpress/wp-admin/js/color-picker.min.js
inflating: wordpress/wp-admin/js/site-icon.min.js
inflating: wordpress/wp-admin/js/auth-app.js
inflating: wordpress/wp-admin/js/code-editor.js
inflating: wordpress/wp-admin/js/common.js
inflating: wordpress/wp-admin/js/set-post-thumbnail.min.js
inflating: wordpress/wp-admin/js/postbox.min.js
inflating: wordpress/wp-admin/js/color-picker.js
inflating: wordpress/wp-admin/js/password-strength-meter.js
inflating: wordpress/wp-admin/js/customize-nav-menus.js
inflating: wordpress/wp-admin/js/editor-expand.js
inflating: wordpress/wp-admin/js/code-editor.min.js
inflating: wordpress/wp-admin/js/set-post-thumbnail.js
inflating: wordpress/wp-admin/options-permalink.php
inflating: wordpress/wp-admin/widgets.php
inflating: wordpress/wp-admin/setup-config.php
inflating: wordpress/wp-admin/install.php
inflating: wordpress/wp-admin/admin-header.php
inflating: wordpress/wp-admin/post-new.php
inflating: wordpress/wp-admin/themes.php
inflating: wordpress/wp-admin/options-reading.php
inflating: wordpress/wp-trackback.php
inflating: wordpress/wp-comments-post.php
[ec2-user@ip-172-31-32-40 ~]$ ls
latest.zip  wordpress
[ec2-user@ip-172-31-32-40 ~]$ |

```

- ◆ To run WordPress web application, you must install of WordPress web application as php language with the following commands <sudo Amazon-linux-extras install -y lamp-mariadb10.2-php7.2.php7.2> otherwise update ec2 instance with the following commands <sudo yum update -y>
  - ◆ Now go inside the unzip directory by using command as “cd <unzip directory> and change the WordPress configuration file by giving command as “sudo mv wp-config-sample.php wp-config.php”

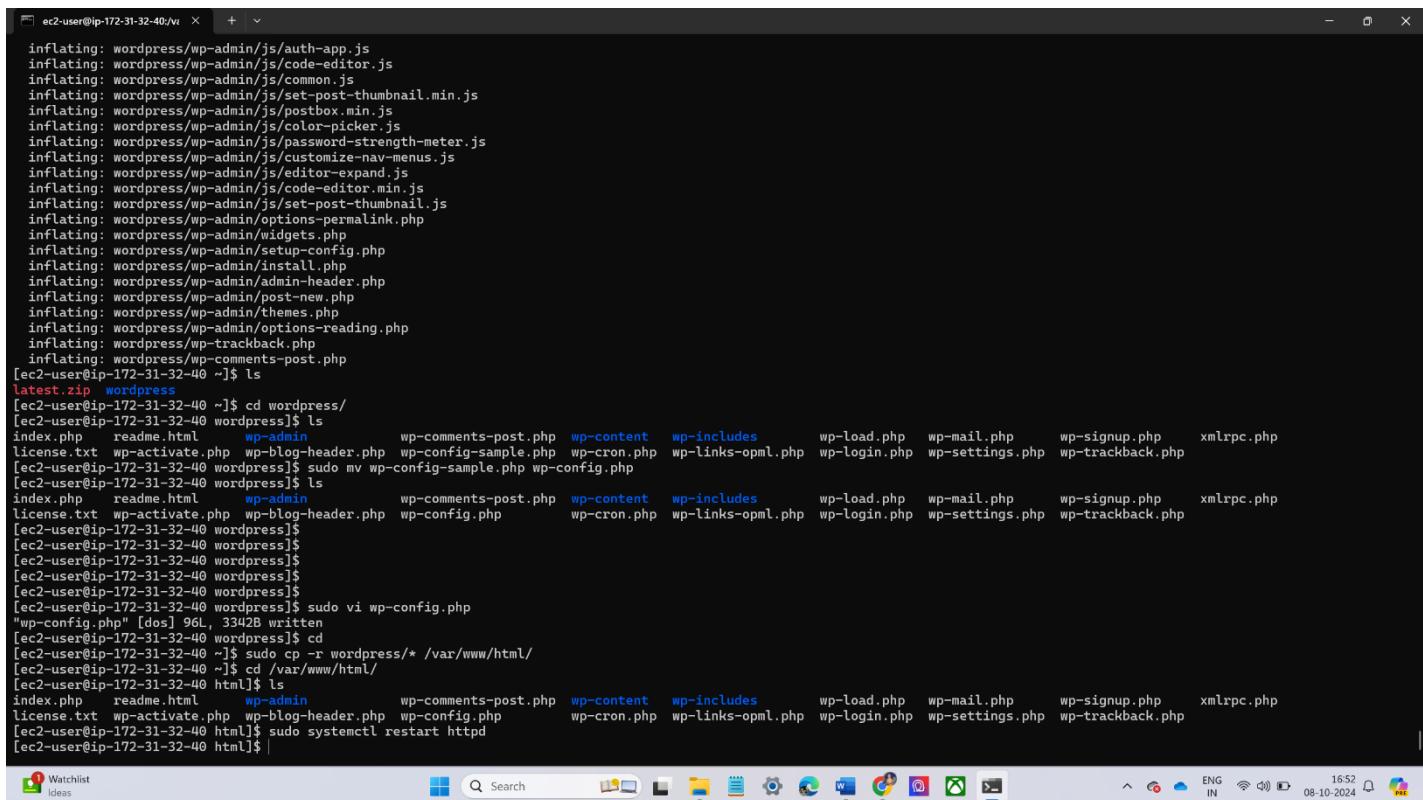
```
ec2-user@ip-172-31-32-40:~$ + 
inflating: wordpress/wp-admin/js/inline-edit-post.min.js
inflating: wordpress/wp-admin/js/customize-widgets.min.js
inflating: wordpress/wp-admin/js/inline-edit-post.js
inflating: wordpress/wp-admin/js/updates.js
inflating: wordpress/wp-admin/js/media-upload.js
inflating: wordpress/wp-admin/js/media.js
inflating: wordpress/wp-admin/js/editor-expand.min.js
inflating: wordpress/wp-admin/js/media-gallery.min.js
inflating: wordpress/wp-admin/js/common.min.js
inflating: wordpress/wp-admin/js/tags-box.min.js
inflating: wordpress/wp-admin/js/svg-painter.min.js
inflating: wordpress/wp-admin/js/custom-background.js
inflating: wordpress/wp-admin/js/color-picker.min.js
inflating: wordpress/wp-admin/js/site-icon.min.js
inflating: wordpress/wp-admin/js/auth-app.js
inflating: wordpress/wp-admin/js/code-editor.js
inflating: wordpress/wp-admin/js/common.js
inflating: wordpress/wp-admin/js/set-post-thumbnail.min.js
inflating: wordpress/wp-admin/js/postbox.min.js
inflating: wordpress/wp-admin/js/color-picker.js
inflating: wordpress/wp-admin/js/password-strength-meter.js
inflating: wordpress/wp-admin/js/customize-nav-menus.js
inflating: wordpress/wp-admin/js/editor-expand.js
inflating: wordpress/wp-admin/js/code-editor.min.js
inflating: wordpress/wp-admin/js/set-post-thumbnail.js
inflating: wordpress/wp-admin/options-permalink.php
inflating: wordpress/wp-admin/widgets.php
inflating: wordpress/wp-admin/setup-config.php
inflating: wordpress/wp-admin/install.php
inflating: wordpress/wp-admin/admin-header.php
inflating: wordpress/wp-admin/post-new.php
inflating: wordpress/wp-admin/themes.php
inflating: wordpress/wp-admin/options-reading.php
inflating: wordpress/wp-trackback.php
inflating: wordpress/wp-comments-post.php
[ec2-user@ip-172-31-32-40 ~]$ ls
latest.zip wordpress
[ec2-user@ip-172-31-32-40 ~]$ cd wordpress/
[ec2-user@ip-172-31-32-40 wordpress]$ ls
index.php readme.html wp-admin wp-comments-post.php wp-content wp-includes wp-load.php wp-mail.php wp-signup.php xmlrpc.php
license.txt wp-activate.php wp-blog-header.php wp-config-sample.php wp-cron.php wp-links-opml.php wp-login.php wp-settings.php wp-trackback.php
[ec2-user@ip-172-31-32-40 wordpress]$ sudo mv wp-config-sample.php wp-config.php
[ec2-user@ip-172-31-32-40 wordpress]$ ls
index.php readme.html wp-admin wp-comments-post.php wp-content wp-includes wp-load.php wp-mail.php wp-signup.php xmlrpc.php
license.txt wp-activate.php wp-blog-header.php wp-config.php wp-cron.php wp-links-opml.php wp-login.php wp-settings.php wp-trackback.php
[ec2-user@ip-172-31-32-40 wordpress]$
```

- ◆ Now do some configurations in the WordPress configuration file by giving database name, username, password and hostname for that execute as `sudo vi wp-config.php`" .along with these change the WordPress secret key. To get the secret key browse the Google generate WordPress secret key.

- ◆ Now copy the WordPress directory to the document root directory to host the web application of WordPress by giving as command as

Sudo cp -I <unzip file>/\* /var/www/html/

- ◆ To restart the httpd server by using command sudo systemctl restart the httpd

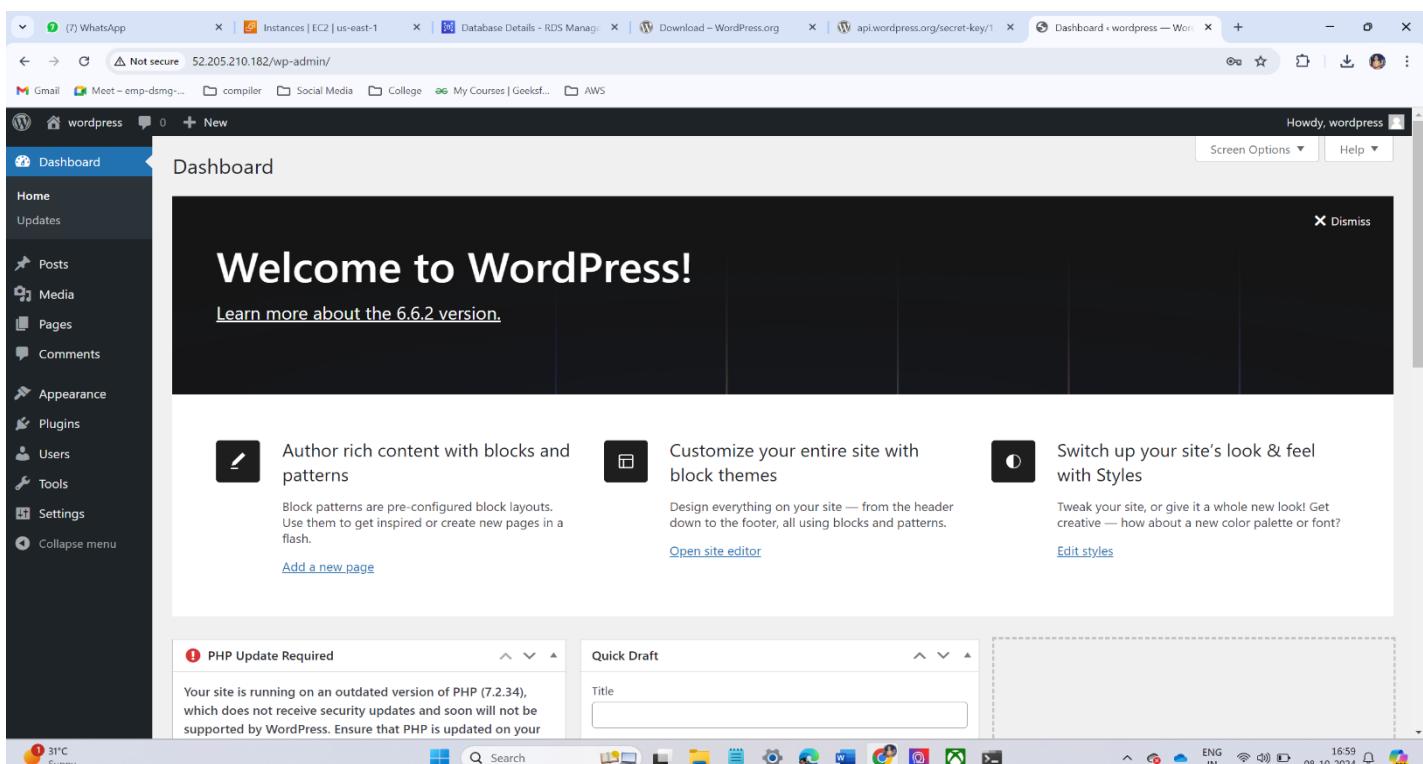


```

ec2-user@ip-172-31-32-40:~$ cd wordpress/
[ec2-user@ip-172-31-32-40 wordpress]$ ls
index.php  readme.html  wp-admin      wp-comments-post.php  wp-content    wp-includes   wp-load.php   wp-mail.php   wp-signup.php  xmlrpc.php
license.txt  wp-activate.php  wp-blog-header.php  wp-config-sample.php  wp-cron.php  wp-links-opml.php  wp-login.php  wp-settings.php  wp-trackback.php
[ec2-user@ip-172-31-32-40 wordpress]$ sudo mv wp-config-sample.php wp-config.php
[ec2-user@ip-172-31-32-40 wordpress]$ ls
index.php  readme.html  wp-admin      wp-comments-post.php  wp-content    wp-includes   wp-load.php   wp-mail.php   wp-signup.php  xmlrpc.php
license.txt  wp-activate.php  wp-blog-header.php  wp-config.php     wp-cron.php  wp-links-opml.php  wp-login.php  wp-settings.php  wp-trackback.php
[ec2-user@ip-172-31-32-40 wordpress]$ 
[ec2-user@ip-172-31-32-40 wordpress]$ sudo vi wp-config.php
"wp-config.php" [dos] 96L, 334B written
[ec2-user@ip-172-31-32-40 wordpress]$ cd
[ec2-user@ip-172-31-32-40 wordpress]$ 
[ec2-user@ip-172-31-32-40 wordpress]$ 
[ec2-user@ip-172-31-32-40 wordpress]$ 
[ec2-user@ip-172-31-32-40 wordpress]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-172-31-32-40 html]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-32-40 html]$ 

```

- ◆ Now go to ec2 instance and copy public ipv4 and paste it on google browse it and check the official page of WordPress it displays



Dashboard

# Welcome to WordPress!

Learn more about the 6.6.2 version.

**Author rich content with blocks and patterns**

Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.

[Add a new page](#)

**Customize your entire site with block themes**

Design everything on your site — from the header down to the footer, all using blocks and patterns.

[Open site editor](#)

**Switch up your site's look & feel with Styles**

Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?

[Edit styles](#)

**PHP Update Required**

Your site is running on an outdated version of PHP (7.2.34), which does not receive security updates and soon will not be supported by WordPress. Ensure that PHP is updated on your

# Deploy WordPress Web Application By Using Docker Compose File

- ◆ First login to the aws account with credentials.

The screenshot shows the AWS Console Home page. At the top, there's a navigation bar with tabs like WhatsApp, Console Home, Databases - RDS Management, Download - WordPress.org, and api.wordpress.org/secret-key/. Below the navigation bar, the URL is us-east-1.console.aws.amazon.com/console/home?region=us-east-1. The main content area has several sections:

- Recently visited:** RDS, EC2, Billing and Cost Management, IAM, Elastic Container Registry, and Elastic Kubernetes Service.
- Applications:** Shows 0 applications in the US East (N. Virginia) region. It includes a "Create application" button and a "Find applications" search bar.
- Welcome to AWS:** Includes "Getting started with" and "Open issues" sections.
- AWS Health:** Shows current month costs and cost (\$) metrics.
- Cost and usage:** Shows current month costs and cost (\$) metrics.

The bottom of the screen shows the AWS footer with links for CloudShell, Feedback, and various AWS services like Lambda, S3, and CloudWatch. The date and time are 08-10-2024 at 17:13.

- ◆ Launch the instance allow port 80 and 22.

The screenshot shows the EC2 Launch an instance page. The URL is us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:. The main content area shows a green success message: "Successfully initiated launch of instance (i-0270b1b0e5b57a221)". Below this, there's a "Launch log" link. A "Next Steps" section contains the following options:

- Create billing and free tier usage alerts
- Connect to your instance
- Connect an RDS database
- Create EBS snapshot policy

Each option has a corresponding "Create [service] alerts" or "Create [service] database" button. The bottom of the screen shows the AWS footer with links for CloudShell, Feedback, and various AWS services like Lambda, S3, and CloudWatch. The date and time are 08-10-2024 at 17:15.

- ◆ Connect instance to the gitbash and install git and docker

- ◆ Now give permissions to limited Linux user account to docker group by using command `sudo chmod 666 /var/run/docker.sock`

```
ec2-user@ip-172-31-32-207:~ + - x 2/5
Installing : containerd=1.7.22-1.amzn2.0.2.x86_64 3/5
Installing : libcgroup=0.41-21.amzn2.x86_64 4/5
Installing : pigz=2.3.4-1.amzn2.0.1.x86_64 5/5
Installing : docker=25.0.6-1.amzn2.0.2.x86_64 1/5
Verifying : runc=1.1.13-1.amzn2.x86_64 2/5
Verifying : docker=25.0.6-1.amzn2.0.2.x86_64 3/5
Verifying : pigz=2.3.4-1.amzn2.0.1.x86_64 4/5
Verifying : containerd=1.7.22-1.amzn2.0.2.x86_64 5/5
Verifying : libcgroup=0.41-21.amzn2.x86_64

Installed:
  docker.x86_64 0:25.0.6-1.amzn2.0.2

Dependency Installed:
  containerd.x86_64 0:1.7.22-1.amzn2.0.2           libcgroup.x86_64 0:0.41-21.amzn2.0.1          pigz.x86_64 0:2.3.4-1.amzn2.0.1          runc.x86_64 0:1.1.13-1.amzn2

Complete!
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
chmod: cannot access '/var/run/docker.sock': No such file or directory
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
  Active: active (running) since Tue 2024-10-08 11:49:22 UTC; 18s ago
    Docs: https://docs.docker.com
 Main PID: 3519 (dockerd)
   CGroup: /system.slice/docker.service
           └─3519 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 08 11:49:21 ip-172-31-32-207.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.801616270Z" level=info msg="Starting up"
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.848651512Z" level=info msg="Loading containers: start."
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.072520665Z" level=info msg="Loading containers: done."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.086772937Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.087196159Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.087529464Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.087950413Z" level=info msg="Daemon has completed initialization"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.130317891Z" level=info msg="API listen on /run/docker.sock"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal systemd[1]: Started Docker Application Container Engine.

Hint: Some lines were ellipsized. Use -l to show in full.
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
docker.pid  docker.sock
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-32-207 ~]$
```

◆ Now start and enable the docker.

```

Transaction test succeeded
Running transaction
  Installing : runc-1.1.13-1.amzn2.x86_64 1/5
  Installing : containerd-1.7.22-1.amzn2.0.2.x86_64 2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64 3/5
  Installing : pigz-2.3-4.1.amzn2.0.1.x86_64 4/5
  Installing : docker-25.0.6-1.amzn2.0.2.x86_64 5/5
  Verifying : runc-1.1.13-1.amzn2.x86_64 1/5
  Verifying : docker-25.0.6-1.amzn2.0.2.x86_64 2/5
  Verifying : containerd-1.7.22-1.amzn2.0.2.x86_64 3/5
  Verifying : libcgroup-0.41-21.amzn2.x86_64 4/5
  Complete!
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
chmod: cannot access '/var/run/docker.sock': No such file or directory
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
  Active: active (running) since Tue 2024-10-08 11:49:22 UTC; 18s ago
    Docs: https://docs.docker.com
   Main PID: 3519 (dockerd)
    CGroup: /system.slice/docker.service
           └─3519 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 08 11:49:21 ip-172-31-32-207.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.801616270Z" level=info msg="Starting up"
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.848651512Z" level=info msg="Loading containers: start."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0725206652Z" level=info msg="Loading containers: done."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0867729377Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0871961592Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0875294642Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0879504132Z" level=info msg="Daemon has completed initialization"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.1303178912Z" level=info msg="API listen on /run/docker.sock"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-32-207 ~]$ 
```

◆ Now install docker compose by using google.

```

Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
  Active: active (running) since Tue 2024-10-08 11:49:22 UTC; 18s ago
    Docs: https://docs.docker.com
   Main PID: 3519 (dockerd)
    CGroup: /system.slice/docker.service
           └─3519 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 08 11:49:21 ip-172-31-32-207.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.801616270Z" level=info msg="Starting up"
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.848651512Z" level=info msg="Loading containers: start."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0725206652Z" level=info msg="Loading containers: done."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0867729377Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0871961592Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0875294642Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0879504132Z" level=info msg="Daemon has completed initialization"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-32-207 ~]$ client_loop: send disconnect: Connection reset

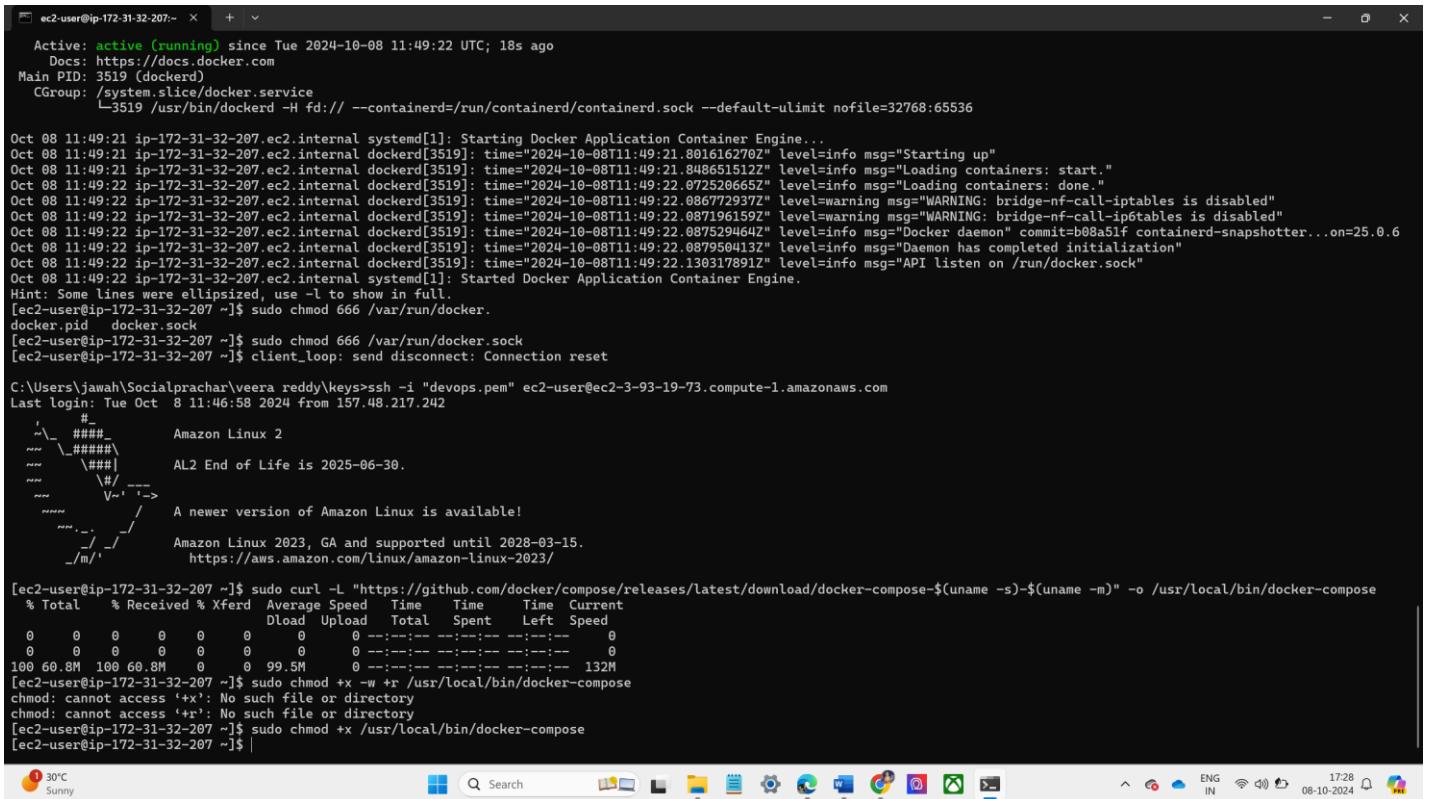
C:\Users\jawah\Socialprachar\veera reddy\keys>ssh -i "devops.pem" ec2-user@ec2-3-93-19-73.compute-1.amazonaws.com
Last login: Tue Oct  8 11:46:58 2024 from 157.48.217.242
#
  _\_\_ #####
  ~~ \#####
  ~~ \|##|
  ~~ \|#|
  ~~ \|#
  ~~ \|-'-->
  ~~ \|_/
  A newer version of Amazon Linux is available!
  ~~ \|_/_/
  Amazon Linux 2023, GA and supported until 2028-03-15.
  ~~ \|_/_/
  https://aws.amazon.com/linux/amazon-linux-2023/
  /m/| 
```

[ec2-user@ip-172-31-32-207 ~]\$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-\$(uname -s)-\$(uname -m)" -o /usr/local/bin/docker-compose

Total	% Received	% Xferd	Average Speed	Time	Time	Time	Current		
				Dload	Upload	Total	Spent	Left	Speed
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
100.60.8M	100.60.8M	0	0	99.5M	0	0	132M		

[ec2-user@ip-172-31-32-207 ~]\$ |

- ♦ Apply excitable permissions to the by using command sudo chmod +x-w +r /usr/local/bin/docker-compose



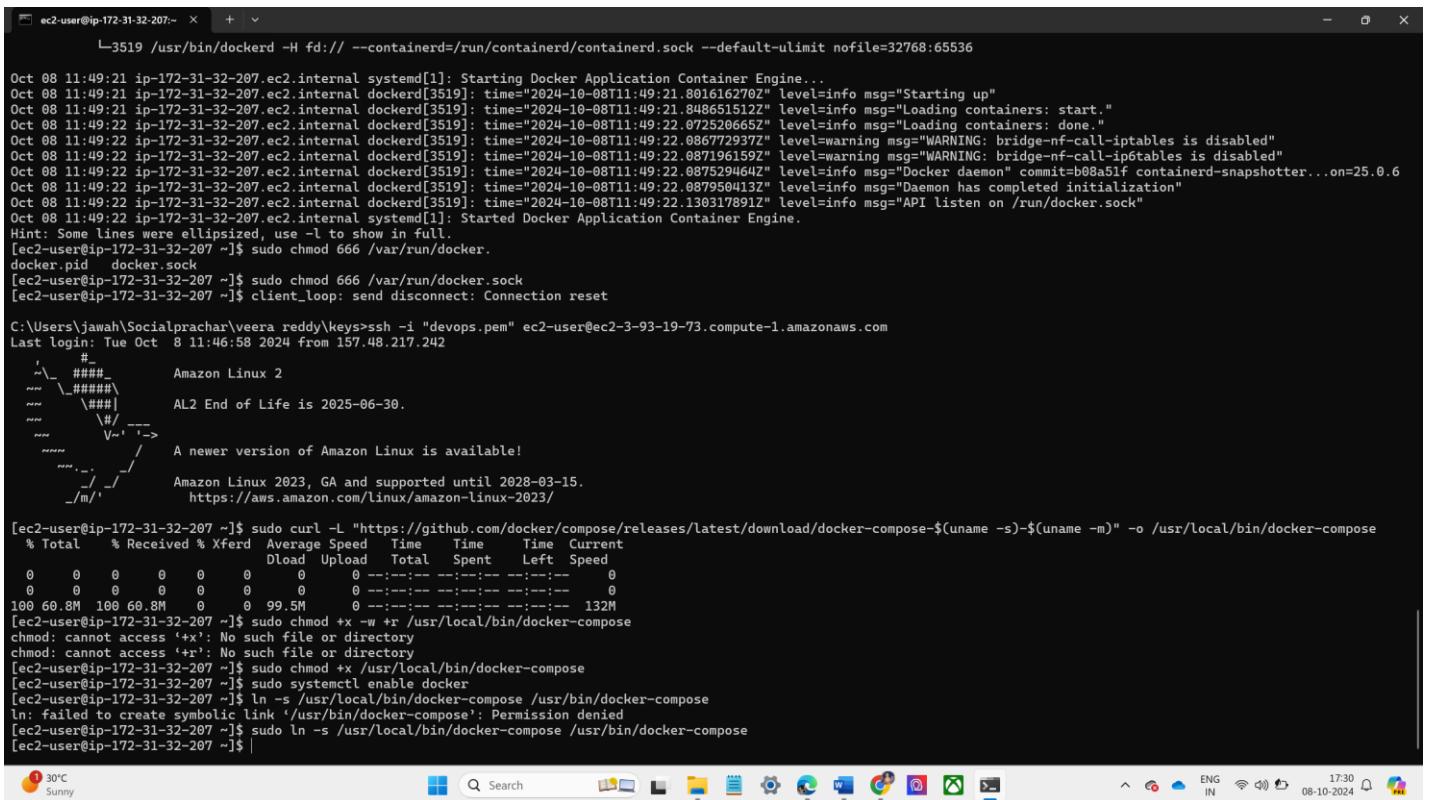
```

ec2-user@ip-172-31-32-207:~ % 
Active: active (running) since Tue 2024-10-08 11:49:22 UTC; 18s ago
  Docs: https://docs.docker.com
Main PID: 3519 (dockerd)
CGroup: /system.slice/docker.service
└─3519 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 08 11:49:21 ip-172-31-32-207.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.801616270Z" level=info msg="Starting up"
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.848651512Z" level=info msg="Loading containers: start."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0725206652Z" level=info msg="Loading containers: done."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0867729372Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0871961592Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0875294647Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0879504137Z" level=info msg="Daemon has completed initialization"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.1303178912Z" level=info msg="API listen on /run/docker.sock"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.
docker.pid  docker.sock
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-32-207 ~]$ client_loop: send disconnect: Connection reset

C:\Users\jawah\Socialprachar\veera reddy>keys>ssh -i "devops.pem" ec2-user@ec2-3-93-19-73.compute-1.amazonaws.com
Last login: Tue Oct 8 11:46:58 2024 from 157.48.217.242
'`#`#
`~` \####_      Amazon Linux 2
`~` \####`#
`~` \##`#
`~` \#/`#
`~` V~`-->
`~~` /`          A newer version of Amazon Linux is available!
`~~` .-`/`        Amazon Linux 2023, GA and supported until 2028-03-15.
`~`/`/`          https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-32-207 ~]$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left  Speed
0       0     0     0       0      0 0:00:00:00:00:00:00:00
0       0     0     0       0      0 0:00:00:00:00:00:00:00
100  60.8M  100 60.8M  0       0 99.5M 0:00:00:00:00:00:00:00 132M
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod +x -w+r /usr/local/bin/docker-compose
chmod: cannot access '+x': No such file or directory
chmod: cannot access '+r': No such file or directory
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod +x /usr/local/bin/docker-compose
[ec2-user@ip-172-31-32-207 ~]$ |
```

- ♦ Apply create symbolic link using command as <ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose



```

ec2-user@ip-172-31-32-207:~ % 
└─3519 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 08 11:49:21 ip-172-31-32-207.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.801616270Z" level=info msg="Starting up"
Oct 08 11:49:21 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:21.848651512Z" level=info msg="Loading containers: start."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0725206652Z" level=info msg="Loading containers: done."
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0867729372Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0871961592Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0875294647Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.0879504137Z" level=info msg="Daemon has completed initialization"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal dockerd[3519]: time="2024-10-08T11:49:22.1303178912Z" level=info msg="API listen on /run/docker.sock"
Oct 08 11:49:22 ip-172-31-32-207.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.
docker.pid  docker.sock
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-32-207 ~]$ client_loop: send disconnect: Connection reset

C:\Users\jawah\Socialprachar\veera reddy>keys>ssh -i "devops.pem" ec2-user@ec2-3-93-19-73.compute-1.amazonaws.com
Last login: Tue Oct 8 11:46:58 2024 from 157.48.217.242
'`#`#
`~` \####_      Amazon Linux 2
`~` \####`#
`~` \##`#
`~` \#/`#
`~` V~`-->
`~~` /`          A newer version of Amazon Linux is available!
`~~` .-`/`        Amazon Linux 2023, GA and supported until 2028-03-15.
`~`/`/`          https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-32-207 ~]$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left  Speed
0       0     0     0       0      0 0:00:00:00:00:00:00:00
0       0     0     0       0      0 0:00:00:00:00:00:00:00
100  60.8M  100 60.8M  0       0 99.5M 0:00:00:00:00:00:00:00 132M
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod +x -w+r /usr/local/bin/docker-compose
chmod: cannot access '+x': No such file or directory
chmod: cannot access '+r': No such file or directory
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod +x /usr/local/bin/docker-compose
[ec2-user@ip-172-31-32-207 ~]$ ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
ln: failed to create symbolic link '/usr/bin/docker-compose': Permission denied
[ec2-user@ip-172-31-32-207 ~]$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
[ec2-user@ip-172-31-32-207 ~]$ |
```

- ◆ Now create docker-compose.yaml file to pull the image from docker hub

```

version: '3.3'

services:
  db:
    image: mysql:8.0.19
    command: '--default-authentication-plugin=mysql_native_password'
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD=wordpress
      - MYSQL_DATABASE=veera
      - MYSQL_USER=reddy
      - MYSQL_PASSWORD=susmitha

  wordpress:
    image: wordpress:latest
    ports:
      - "80:80"
    restart: always
    environment:
      - WORDPRESS_DB_HOST=db
      - WORDPRESS_DB_USER=reddy
      - WORDPRESS_DB_PASSWORD=susmitha
      - WORDPRESS_DB_NAME=veera

volumes:
  db_data:

```

:wq!

- ◆ Now to pull the image of MySQL and WordPress. We have executed the create docker compose. yaml by using the command <docker-compose up -d>

```

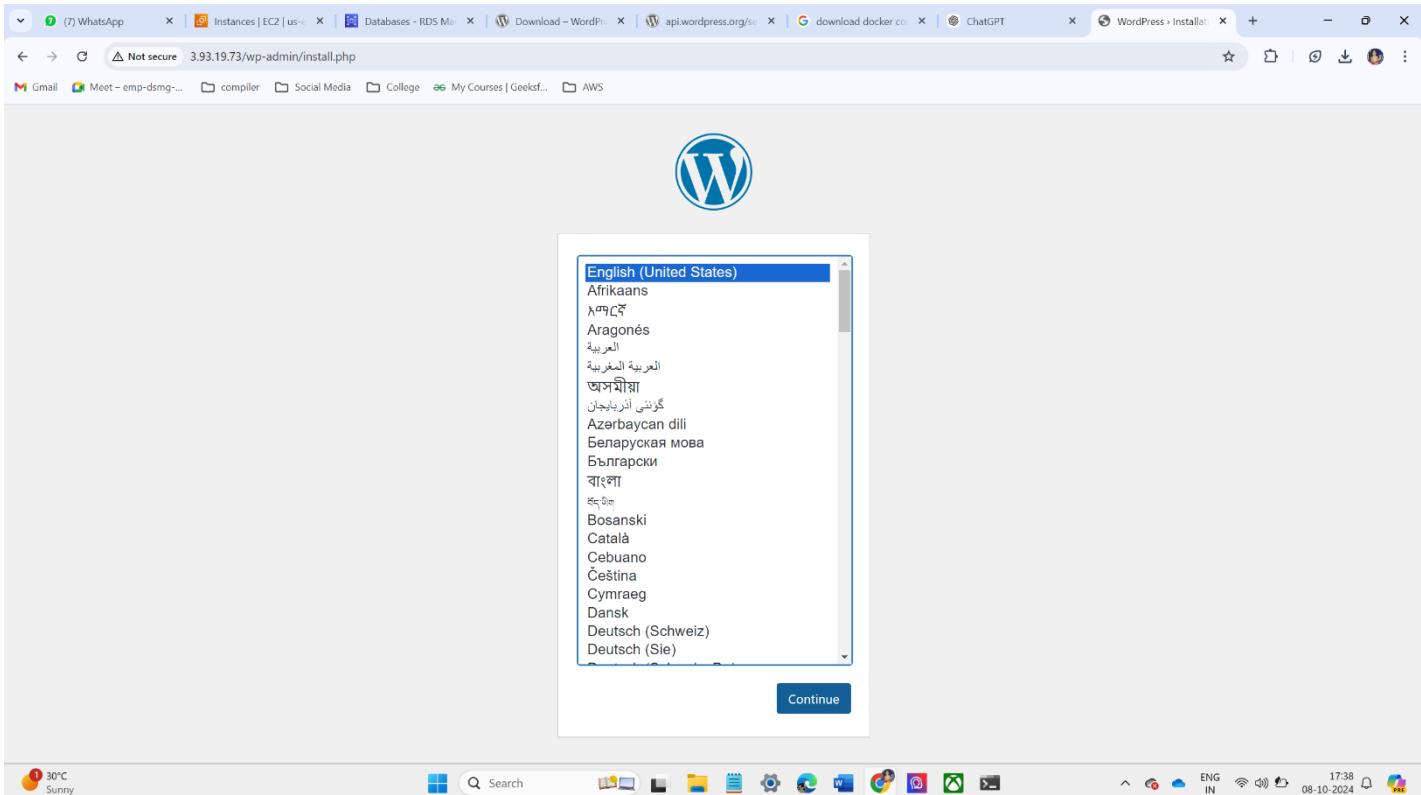
[ec2-user@ip-172-31-32-207 ~]$ sudo chmod +x /usr/local/bin/docker-compose
[ec2-user@ip-172-31-32-207 ~]$ sudo systemctl enable docker
[ec2-user@ip-172-31-32-207 ~]$ ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
ln: failed to create symbolic link '/usr/bin/docker-compose': Permission denied
[ec2-user@ip-172-31-32-207 ~]$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
[ec2-user@ip-172-31-32-207 ~]$ sudo vi docker-compose.yaml
[New] 29L, 590B written
[ec2-user@ip-172-31-32-207 ~]$ docker-compose up -d
WARN[0000] /home/ec2-user/docker-compose.yaml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 31/35
  ▪ wordpress [██████████] 216.4MB / 245.9MB Pulling
    ✓ 302e3ee49865 Pull complete
    ✓ 07fc08906857 Pull complete
    : 141aa7d58c57 Extracting [=====] 89.69MB/104.3MB
    ✓ 2720dd4bc8b3 Download complete
    ✓ 82deca51468c Download complete
    ✓ dec741dfa526 Download complete
    ✓ e284b0efab94 Download complete
    ✓ 0a9b8325ee85 Download complete
    ✓ ef45e9da2633 Download complete
    ✓ f2b46378d521 Download complete
    ✓ 5c104459dad Download complete
    ✓ 282dbb78d4dd Download complete
    ✓ 4a1f827cc210 Download complete
    ✓ 16a044b841d0 Download complete
    ✓ 5aae6c13158b Download complete
    ✓ aafbf1137210 Download complete
    ✓ fae7038f494a Download complete
    ✓ 4306628e8a77 Download complete
    ✓ 5d2474f1d0a5 Download complete
    ✓ 5e101cb8455d Download complete
    ✓ 5db2b612f33a Download complete
  ▪ db [██████████] 152.1MB / 159.1MB Pulling
    ✓ 54fec2fa59d0 Pull complete
    ✓ bcc6c6145912 Pull complete
    ✓ 951c3d959c9d Pull complete
    ✓ 05de4d0e206e Pull complete
    ✓ 319f0394ef42 Pull complete
    ✓ d9185034607b Pull complete
    ✓ 013a9c64dadc Pull complete
    ✓ 96d4c3d3lf9f Pull complete
    : 785bc90808da Extracting [=====] 60.16MB/113MB
    ✓ 1339cf094729 Download complete
    ✓ bee8bf531c37 Download complete
    ✓ 2b40c9f6a918 Download complete

```

- ♦ Now see the container of the pulled images of mysql and WordPress by using command docker ps

```
[+] Running 35/35
  ✓ wordpress Pulled
  ✓ 382e3ee49805 Pull complete
  ✓ 07fc0890b857 Pull complete
  ✓ 141aa7d58c57 Pull complete
  ✓ 2720dd4bcab8b3 Pull complete
  ✓ 82deca51468c Pull complete
  ✓ dec741dfa526 Pull complete
  ✓ e204bb0efab94 Pull complete
  ✓ 0a9b8825e85 Pull complete
  ✓ ef45e9da2633 Pull complete
  ✓ f2b46378d521 Pull complete
  ✓ 5c1044459ddad Pull complete
  ✓ 282db878d4dd Pull complete
  ✓ 4a1f827cc210 Pull complete
  ✓ 16a044b841d6 Pull complete
  ✓ 5aae6c13158b Pull complete
  ✓ aafbf1137210 Pull complete
  ✓ fae7038f494a Pull complete
  ✓ 4306628e8a77 Pull complete
  ✓ 5d2474f1d0a5 Pull complete
  ✓ e5101cb8455d Pull complete
  ✓ 5db2b012f33a Pull complete
  ✓ db Pulled
  ✓ 54fec2fa59d0 Pull complete
  ✓ bcc6c6145912 Pull complete
  ✓ 951c3d959c9d Pull complete
  ✓ 05de4d0e206e Pull complete
  ✓ 319f0394ef42 Pull complete
  ✓ d9185034607b Pull complete
  ✓ 013a9c64dadc Pull complete
  ✓ 96d4c3d31f9f Pull complete
  ✓ 785bc90880da Pull complete
  ✓ 1339cf094729 Pull complete
  ✓ bebb8ff531c37 Pull complete
  ✓ 2b40c9f6a918 Pull complete
[+] Running 4/4
  ✓ Network ec2-user_default      Created
  ✓ Volume "ec2-user_db_data"     Created
  ✓ Container ec2-user-wordpress-1 Started
  ✓ Container ec2-user-db-1       Started
[ec2-user@ip-172-31-32-207 ~]$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED          STATUS              PORTS                         NAMES
0742d1b9a8a        wordpress:latest   "docker-entrypoint.s..."   40 seconds ago   Up 38 seconds      0.0.0.0:80->80/tcp, :::80->80/tcp   ec2-user-wordpress-1
dc8a2b6b301f       mysql:8.0.19      "docker-entrypoint.s..."   40 seconds ago   Up 38 seconds      3306/tcp, 33060/tcp                ec2-user-db-1
[ec2-user@ip-172-31-32-207 ~]$ |
```

- Successfully deployed WordPress application using docker-compose



The screenshot shows the WordPress dashboard with a dark theme. At the top, there is a banner with the text "Welcome to WordPress!" and a link to "Learn more about the 6.6.2 version". Below the banner, there are three main sections: "Author rich content with blocks and patterns", "Customize your entire site with block themes", and "Switch up your site's look & feel with Styles". Each section includes a small icon and a brief description. At the bottom of the dashboard, there are two widgets: "Site Health Status" (showing "No information yet...") and "Quick Draft" (with a title input field). The browser's address bar shows the URL "Not secure 3.93.19.73/wp-admin/". The status bar at the bottom right indicates the date and time as "08-10-2024 17:40".

Welcome to WordPress!

Learn more about the 6.6.2 version.

Author rich content with blocks and patterns

Customize your entire site with block themes

Switch up your site's look & feel with Styles

Site Health Status

No information yet...

Quick Draft

Title

30°C Sunny

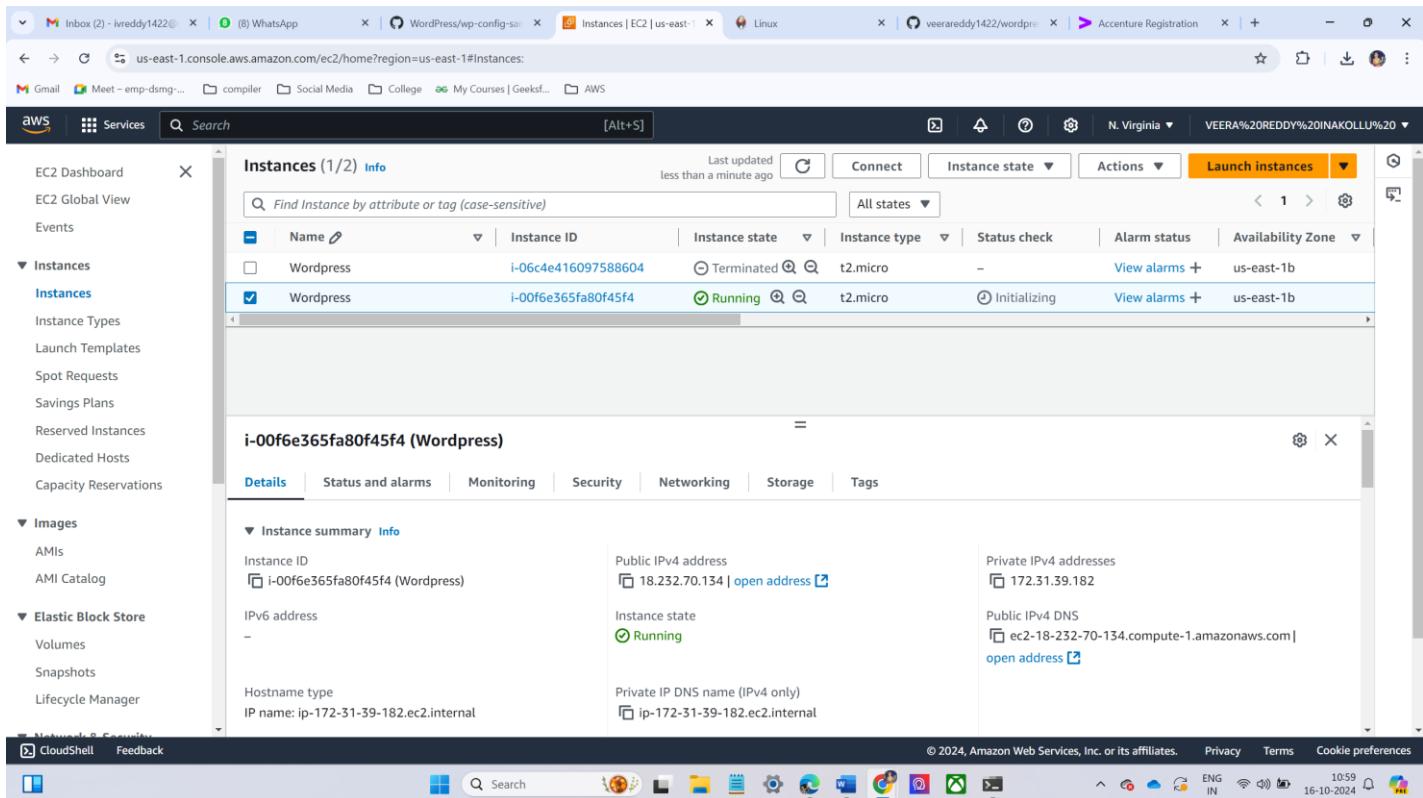
Search

ENG IN

08-10-2024 17:40

## Deploy WordPress web application by using git and jenkins?

- ◆ Login AWS Account and launch EC2 instance



- ◆ Connect EC2 instance to the command prompt

```
ec2-user@ip-172-31-39-182:~ + Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jawah\Downloads>
C:\Users\jawah\Downloads>ssh -i "veera.pem" ec2-user@ec2-18-232-70-134.compute-1.amazonaws.com
The authenticity of host 'ec2-18-232-70-134.compute-1.amazonaws.com (18.232.70.134)' can't be established.
ED25519 key fingerprint is SHA256:RX19EUxVu6RLbrH9JGeyk1S2mw7kiewPmHpcXzcjo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-232-70-134.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

          #_
          _\_ #####      Amazon Linux 2
          _\_\#####\_
          \###|      AL2 End of Life is 2025-06-30.
          \#/ ,-->
          \#v,-->
          _/   / A newer version of Amazon Linux is available!
          _/ ,-->
          _/ ,--> Amazon Linux 2023, GA and supported until 2028-03-15.
          _/m'      https://aws.amazon.com/linux/amazon-linux-2023/
No packages needed for security; 2 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-39-182 ~]$ |
```

- ◆ Install Jenkins and also set up the Jenkins

```
ec2-user@ip-172-31-39-182:~ + - x Transaction Summary ====== Install 1 Package Total download size: 89 M Installed size: 89 M Is this ok [y/d/N]: y Downloading packages: jenkins-2.462.3-1.1.noarch.rpm | 89 MB 00:00:09 Running transaction check Running transaction test Transaction test succeeded Running transaction
  Installing : jenkins-2.462.3-1.1.noarch
  Verifying : jenkins-2.462.3-1.1.noarch

Installed:
  jenkins.noarch 0:2.462.3-1.1

Complete!
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2024-10-16 05:35:58 UTC; 15s ago
     Main PID: 7209 (java)
        Tasks: 47
       Memory: 351.7M
      CGroup: /system.slice/jenkins.service
           └─7209 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

Oct 16 05:35:53 ip-172-31-39-182.ec2.internal jenkins[7209]: fbf51989bdd74c89b1d3d4c44eb0c05d
Oct 16 05:35:53 ip-172-31-39-182.ec2.internal jenkins[7209]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Oct 16 05:35:53 ip-172-31-39-182.ec2.internal jenkins[7209]: ****
Oct 16 05:35:58 ip-172-31-39-182.ec2.internal jenkins[7209]: 2024-10-16 05:35:58.263+0000 [id=33] INFO jenkins.InitReactorRunner$1#onAttained: Completed ...alization
Oct 16 05:35:58 ip-172-31-39-182.ec2.internal jenkins[7209]: 2024-10-16 05:35:58.291+0000 [id=25] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is ful...d running
Oct 16 05:35:58 ip-172-31-39-182.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Oct 16 05:35:58 ip-172-31-39-182.ec2.internal jenkins[7209]: 2024-10-16 05:35:58.356+0000 [id=48] INFO h.m.DownloadService$Downloadable#load: Obtained th...Installer
Oct 16 05:35:58 ip-172-31-39-182.ec2.internal jenkins[7209]: 2024-10-16 05:35:58.357+0000 [id=48] INFO hudson.util.Retriger#start: Performed the action ch...tempt #1
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-39-182 ~]$ | 1/1 1/1
```

- ◆ Launch the Jenkins

The screenshot shows the Jenkins dashboard with the following details:

- Header:** Shows multiple browser tabs including "Inbox (2) - ivreddy1422", "WhatsApp", "WordPress/wp-conf...", "Instances | EC2 | us-east-1", "Linux", "veerareddy1422/wo...", "Accenture Registrati...", "Dashboard [Jenkins]", and several AWS services like Gmail, Meet, compiler, Social Media, College, My Courses | GeeksforGeeks, and AWS.
- Top Bar:** Includes a search bar ("Search (CTRL+K)"), help icon, notifications (1), security shield (1), user profile ("veera reddy inakollu"), and a "log out" button.
- Left Sidebar:** Features links for "New Item", "Build History", "Manage Jenkins", and "My Views".
- Build Queue:** A box stating "No builds in the queue."
- Build Executor Status:** A box showing "1 Idle" and "2 Idle".
- Welcome Section:** Displays "Welcome to Jenkins!" and a message about starting a software project.
- Central Area:** Includes a "Create a job" button, a "Set up a distributed build" section with "Set up an agent" and "Configure a cloud" options, and a "Learn more about distributed builds" link.
- Bottom Right:** Shows "REST API" and "Jenkins 2.46.3".
- Taskbar:** Shows the Windows taskbar with various pinned icons.

## ◆ Install docker and set up the docker

```

Installing : runc-1.1.14-1.amzn2.x86_64 1/5
Installing : containerd-1.7.22-1.amzn2.0.2.x86_64 2/5
Installing : libcgroup-0.41-21.amzn2.x86_64 3/5
Installing : pigz-2.3.4-1.amzn2.0.1.x86_64 4/5
Installing : docker-25.0.6-1.amzn2.0.2.x86_64 5/5
Verifying : docker-25.0.6-1.amzn2.0.2.x86_64 1/5
Verifying : pigz-2.3.4-1.amzn2.0.1.x86_64 2/5
Verifying : runc-1.1.14-1.amzn2.x86_64 3/5
Verifying : containerd-1.7.22-1.amzn2.0.2.x86_64 4/5
Verifying : libcgroup-0.41-21.amzn2.x86_64 5/5

Installed:
  docker.x86_64 0:25.0.6-1.amzn2.0.2

Dependency Installed:
  containerd.x86_64 0:1.7.22-1.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 0:1.1.14-1.amzn2

Complete!
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl start enable docker
Failed to start enable.service: Unit not found.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl startdockerd
Unknown operation 'startdockerd'.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
     Active: active (running) since Wed 2024-10-16 05:31:07 UTC; 25s ago
       Docs: https://docs.docker.com
      Main PID: 3507 (dockerd)
         CGroup: /system.slice/docker.service
             └─3507 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 16 05:31:07 ip-172-31-39-182.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.671763108Z" level=info msg="Starting up"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.742175442Z" level=info msg="Loading containers: start."
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.948940133Z" level=info msg="Loading containers: done."
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961213093Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961526457Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961736335Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961959262Z" level=info msg="Daemon has completed initialization"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.995682372Z" level=info msg="API listen on /run/docker.sock"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-39-182 ~]$ 
```

## ◆ Install the docker-compose and enable the permissions

```

Installed:
  docker.x86_64 0:25.0.6-1.amzn2.0.2

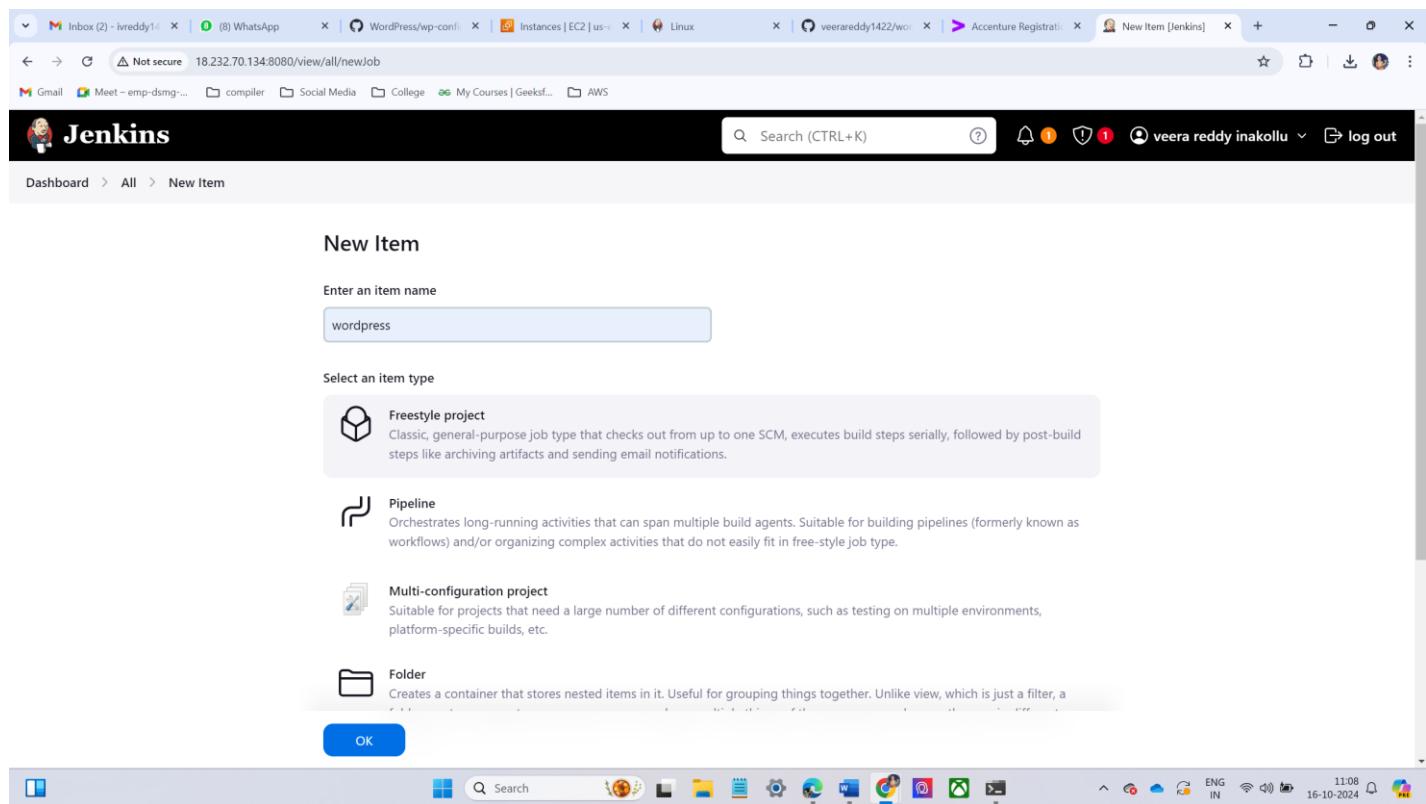
Dependency Installed:
  containerd.x86_64 0:1.7.22-1.amzn2.0.2          libcgroup.x86_64 0:0.41-21.amzn2          pigz.x86_64 0:2.3.4-1.amzn2.0.1          runc.x86_64 0:1.1.14-1.amzn2

Complete!
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl start enable docker
Failed to start enable.service: Unit not found.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl startdockerd
Unknown operation 'startdockerd'.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-39-182 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
     Active: active (running) since Wed 2024-10-16 05:31:07 UTC; 25s ago
       Docs: https://docs.docker.com
      Main PID: 3507 (dockerd)
         CGroup: /system.slice/docker.service
             └─3507 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 16 05:31:07 ip-172-31-39-182.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.671763108Z" level=info msg="Starting up"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.742175442Z" level=info msg="Loading containers: start."
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.948940133Z" level=info msg="Loading containers: done."
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961213093Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961526457Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961736335Z" level=info msg="Docker daemon" commit=b08a51f containerd-snapshotter...on=25.0.6
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.961959262Z" level=info msg="Daemon has completed initialization"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal dockerd[3507]: time="2024-10-16T05:31:07.995682372Z" level=info msg="API listen on /run/docker.sock"
Oct 16 05:31:07 ip-172-31-39-182.ec2.internal systemd[1]: Started Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-39-182 ~]$ sudo rm /usr/local/bin/docker-compose
rm: cannot remove '/usr/local/bin/docker-compose': No such file or directory
[ec2-user@ip-172-31-39-182 ~]$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload Upload Total Spent   Left Speed
0       0       0       0       0       0 0:00:00:00:00:00:00:00
0       0       0       0       0       0 0:00:00:00:00:00:00:00
100  60.8M 100  60.8M 0       0 81.4M 0:00:00:00:00:00:102M
[ec2-user@ip-172-31-39-182 ~]$ sudo chmod +x /usr/local/bin/docker-compose
[ec2-user@ip-172-31-39-182 ~]$ docker-compose --version
Docker Compose version v2.29.7


```

- ◆ Create a New Freestyle Project in Jenkins
  - ◆ Log in to Jenkins.
  - ◆ Click on "New Item".
  - ◆ Enter a name for your project (e.g., WordPress-Deployment).
  - ◆ Select "Freestyle project" and click **OK**.



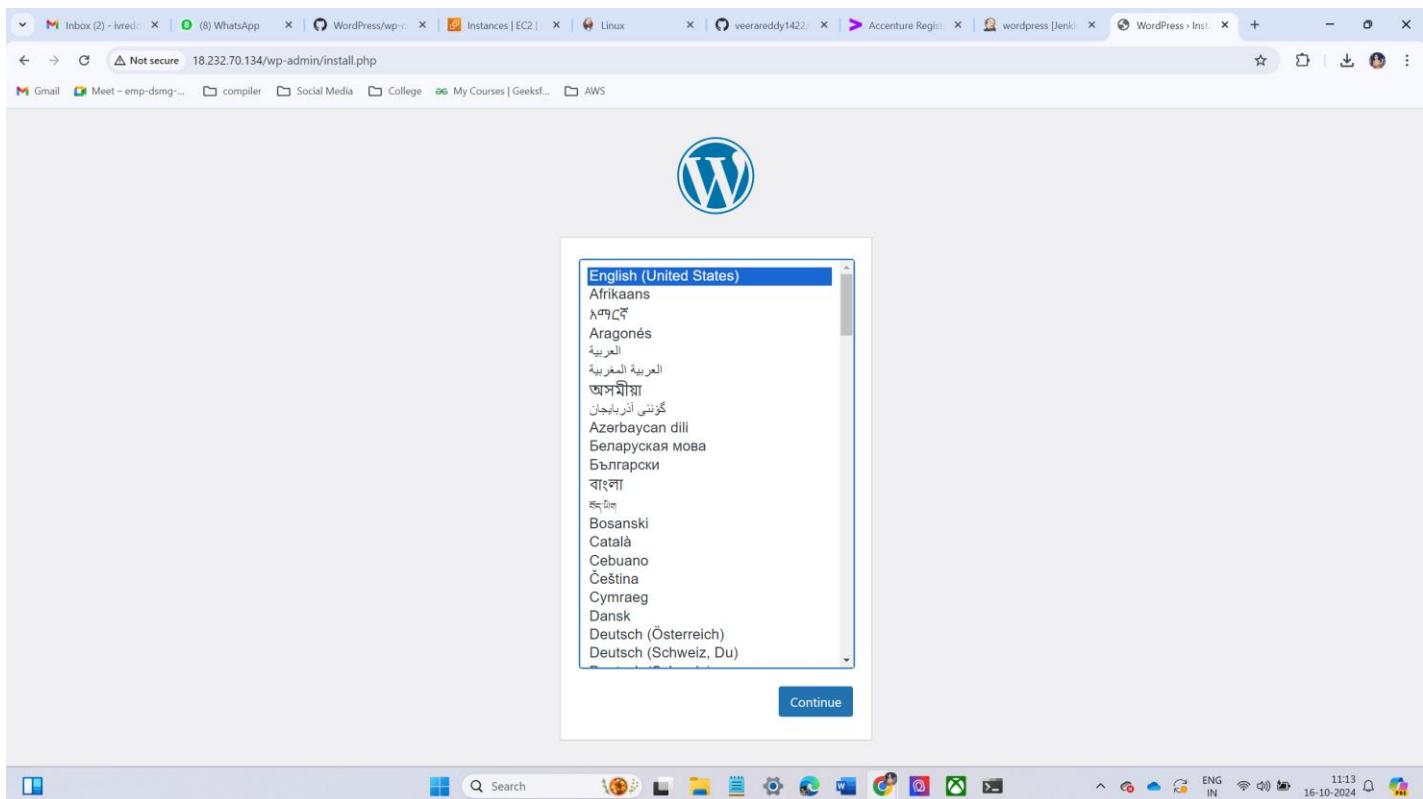
- ◆ Configure Source Code Management (SCM)
- ◆ In the project configuration page, scroll down to the "**Source Code Management**" section.
- ◆ Select "**Git**".
- ◆ Enter the **Repository URL** of your GitHub repository

The screenshot shows the Jenkins project configuration interface for a 'wordpress' job. The 'Source Code Management' section is selected under the 'Configure' tab. The 'Repository URL' field contains 'https://github.com/veerareddy1422/wordpress.git'. The 'Branch Specifier' field contains '/main'. The 'Save' button is visible at the bottom of the form.

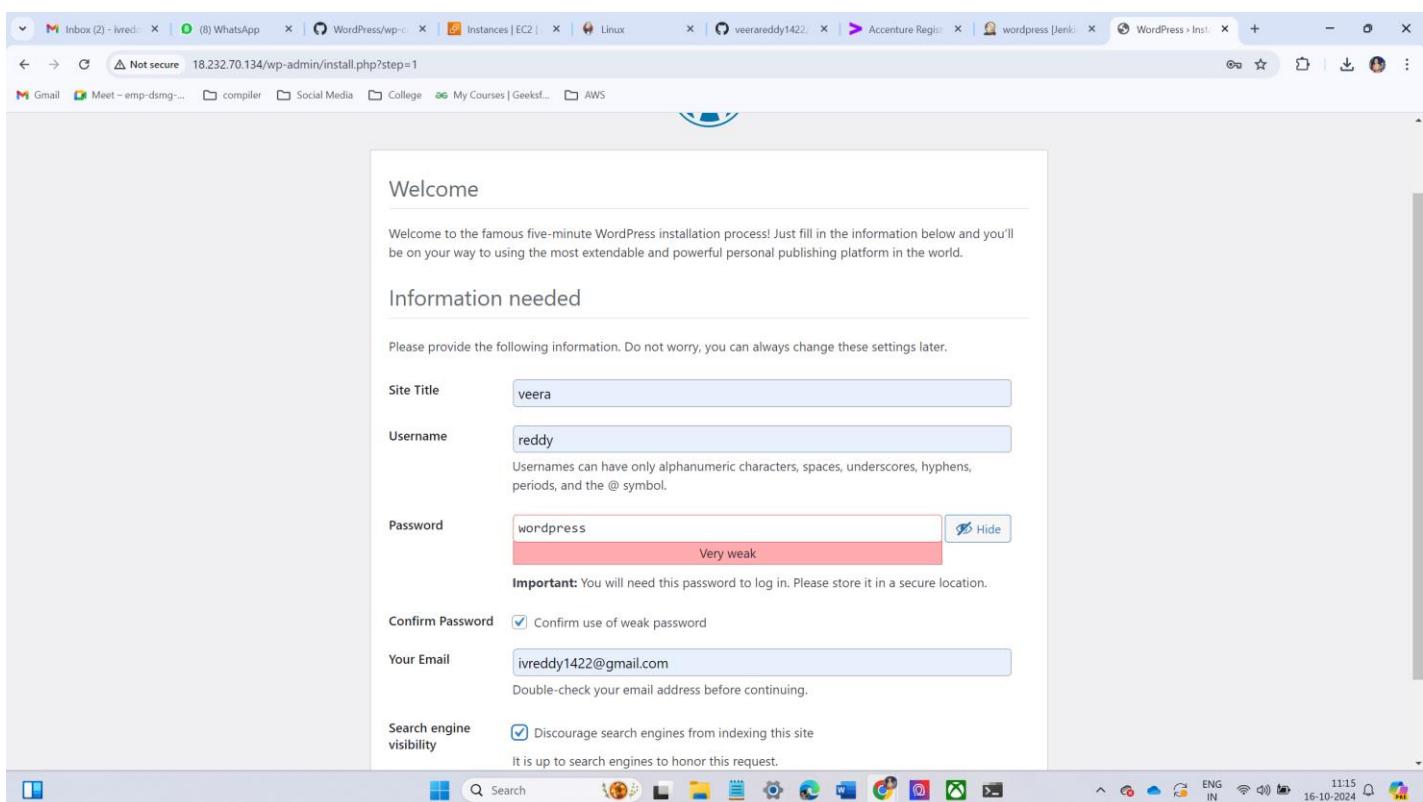
- ◆ Under build steps write script on execute shell script

The screenshot shows the Jenkins project configuration interface for a 'wordpress' job. The 'Build Steps' section is selected under the 'Configure' tab. An 'Execute shell' step is added with the command 'docker-compose up -d'. The 'Save' button is visible at the bottom of the form.

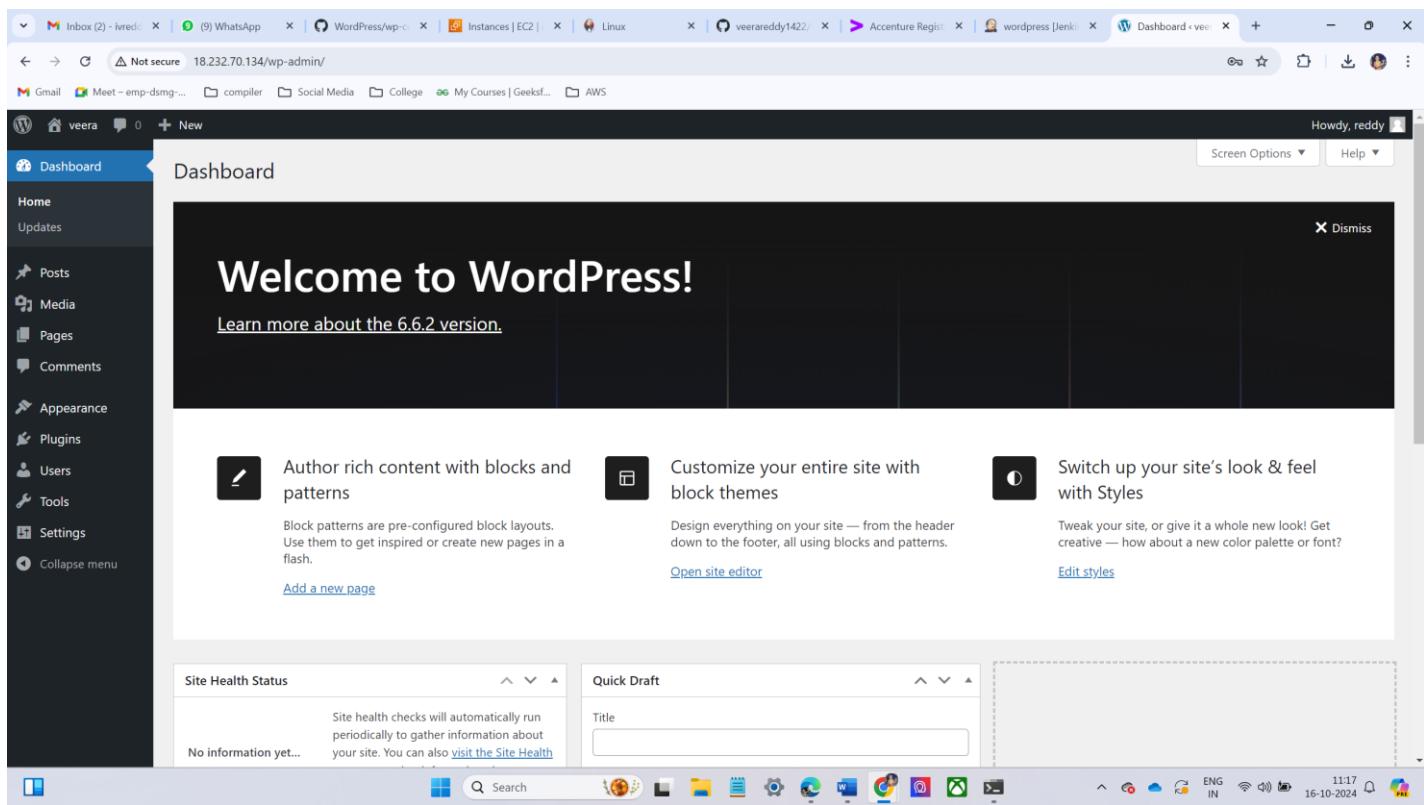
- ◆ Save and build the job
- ◆ Now browse the EC2 instance ipv4 address along with port 80 it will shows like.



- ◆ Now create the wordpress account and fill the details and install the wordpress



- ◆ Successfully host the wordpress application using git and Jenkins



## Deploy WordPress web application by using userdata of EC2 instance

- ◆ Login to AWS Account and Launch EC2 instance and attach security groups port number 20 and 80.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and AMI Catalog. The main area displays 'Instances (1/1) Info'. A table lists one instance: Name (wordpress), Instance ID (i-03de5fa27a2607ad7), Instance state (Running), Instance type (t2.micro), Status check (Initializing), and Availability Zone (us-east-1b). Below the table, a detailed view for instance i-03de5fa27a2607ad7 (wordpress) is shown under the 'Details' tab. It includes sections for Instance summary, Public IPv4 address (3.84.96.227), Instance state (Running), Private IP4 addresses (172.31.38.207), Public IPv4 DNS (ec2-3-84-96-227.compute-1.amazonaws.com), and Private IP DNS name (ip-172-31-38-207.ec2.internal). At the bottom right of the main window, there are links for CloudShell and Feedback, along with system icons like search, file, and settings.

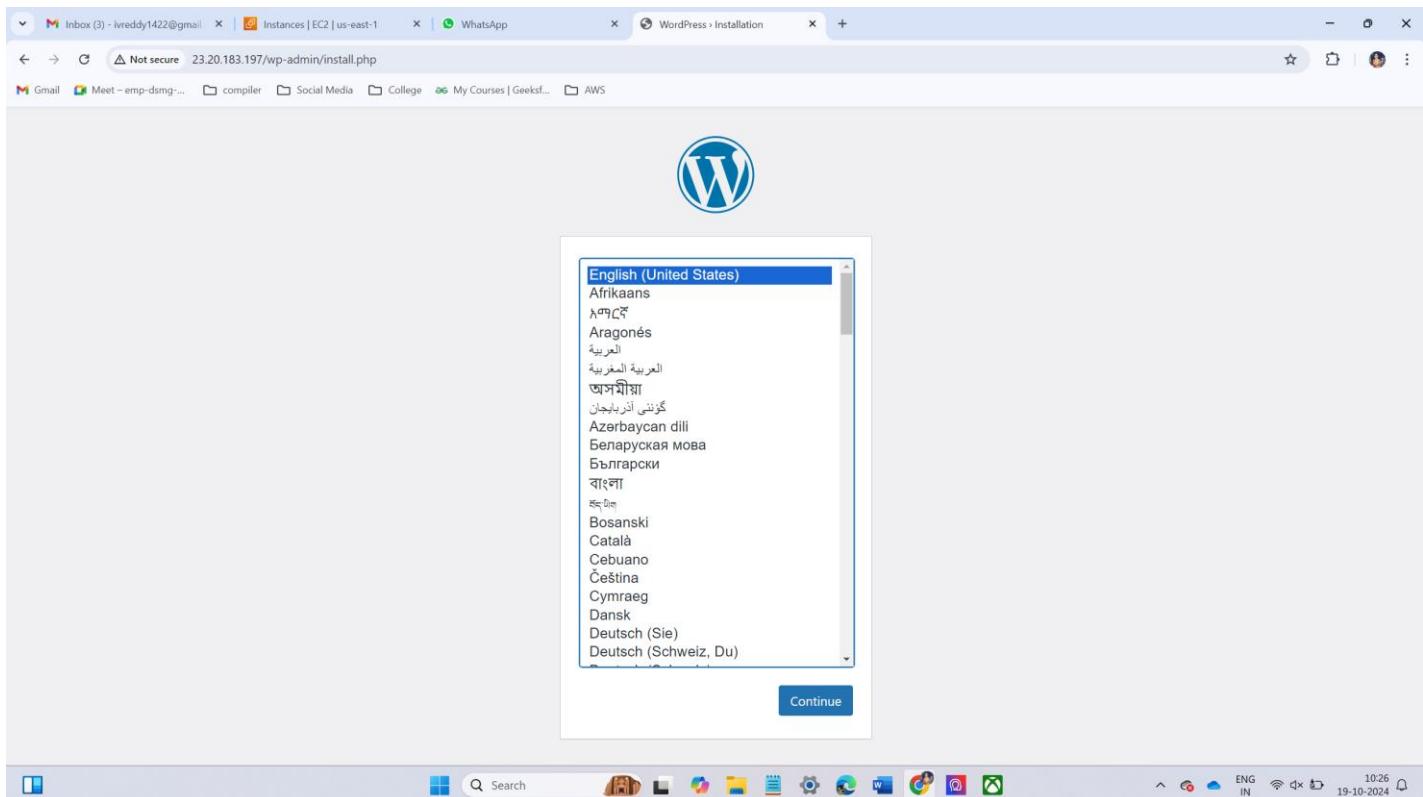
- ◆ Under Advanced details write user data

The screenshot shows the 'Launch an instance' wizard. In the 'User data - optional' section, there is a text input field containing the following shell script:

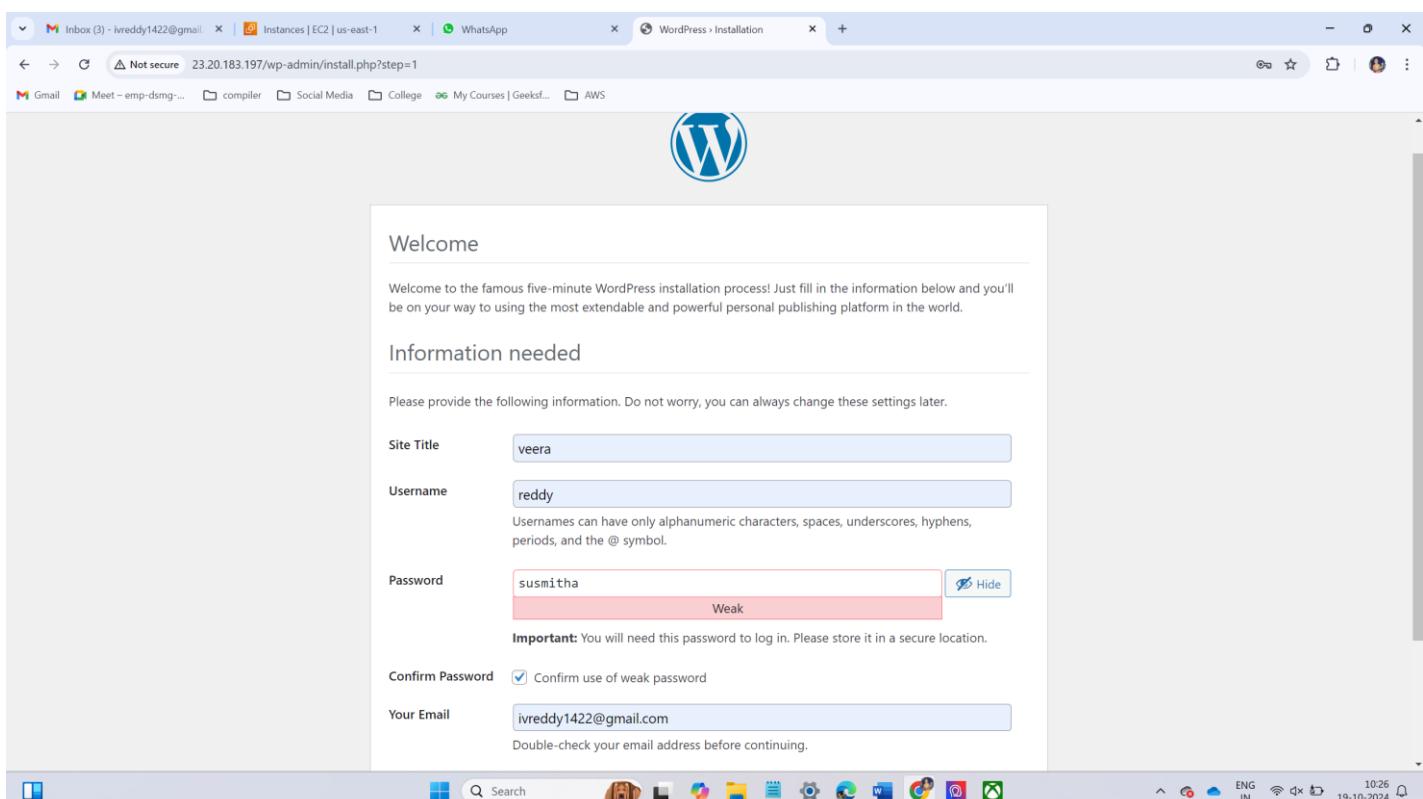
```
#!/bin/bash
sudo yum -y install git
sudo yum -y install docker
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
sudo curl -L
https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
git clone https://github.com/veerareddy1422/wordpress.git
cd wordpress/
docker-compose up -d
```

Below the user data field, there is a checkbox labeled 'User data has already been base64 encoded'. To the right, the 'Summary' section shows the configuration: Number of instances (1), Software Image (AMI) (Amazon Linux 2 Kernel 5.10 AMI...), Virtual server type (instance type) (t2.micro), Firewall (security group) (New security group), and Storage (volumes) (1 volume(s) - 8 GiB). At the bottom right, there are 'Cancel', 'Launch instance' (highlighted in orange), and 'Preview code' buttons. The status bar at the bottom indicates the date and time as 19-10-2024 10:22.

- ◆ Now browse the instance ipv4 address in google



- ◆ Create WordPress account



- ◆ Create the WordPress account and login using credentials

The screenshot shows the WordPress dashboard with a "Welcome to WordPress!" message. A Google Password Manager notification is displayed, asking if the user wants to save their password for the site. The dashboard includes sections for rich content, customization, and styles. Site health status and a quick draft are also shown.

**Welcome to WordPress!**  
Learn more about the 6.6.2 version.

**Author rich content with blocks and patterns**  
Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.  
[Add a new page](#)

**Customize your entire site with block themes**  
Design everything on your site — from the header down to the footer, all using blocks and patterns.  
[Open site editor](#)

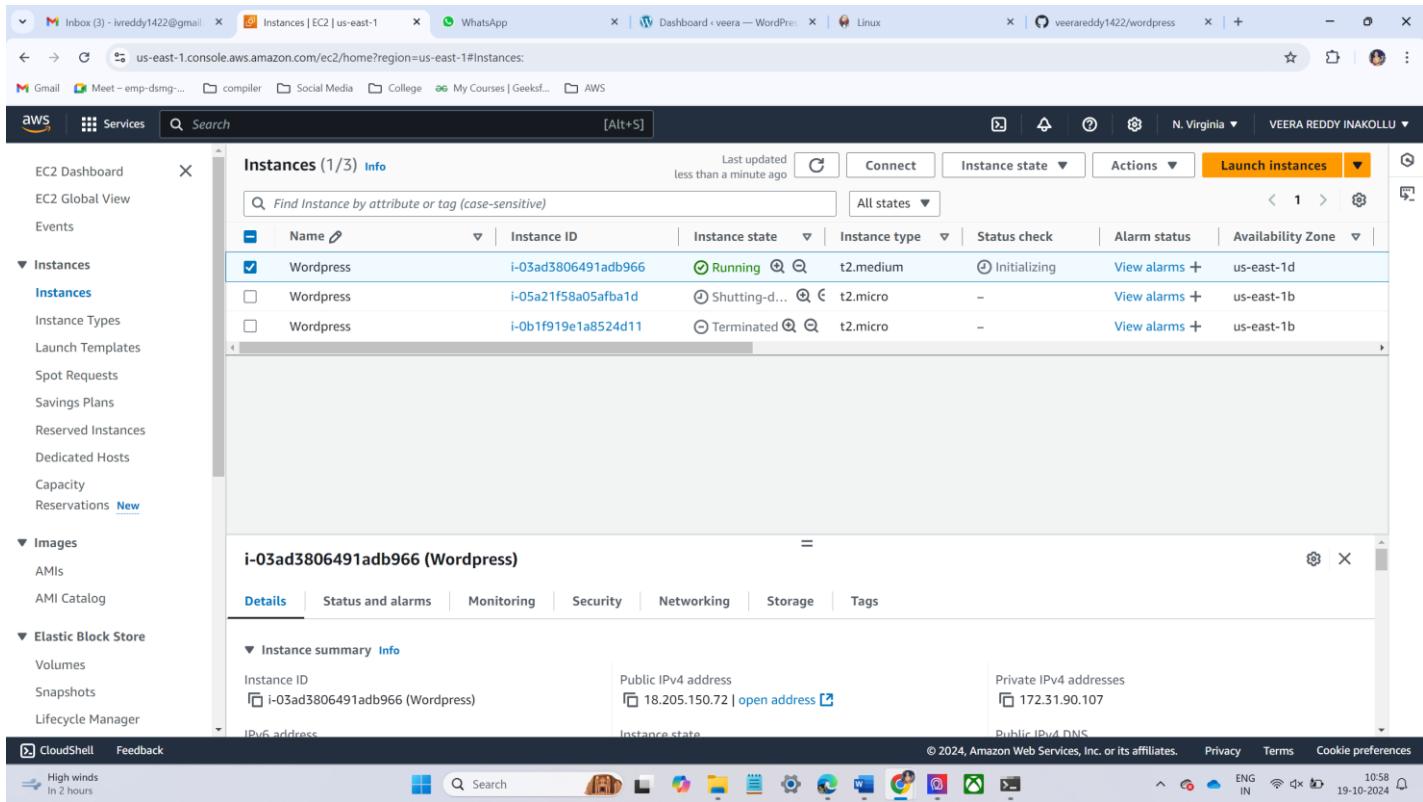
**Switch up your site's look & feel with Styles**  
Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?  
[Edit styles](#)

**Site Health Status**  
Site health checks will automatically run periodically to gather information about your site. You can also [visit the Site Health](#).  
No information yet...

**Quick Draft**  
Title  
[Empty text input field]

## Deploy WordPress web application by using git and jenkins execute shell (bash script)

- ◆ Login to aws account and launch ec2 instance



- ◆ Connect ec2 instance to the gitbash.

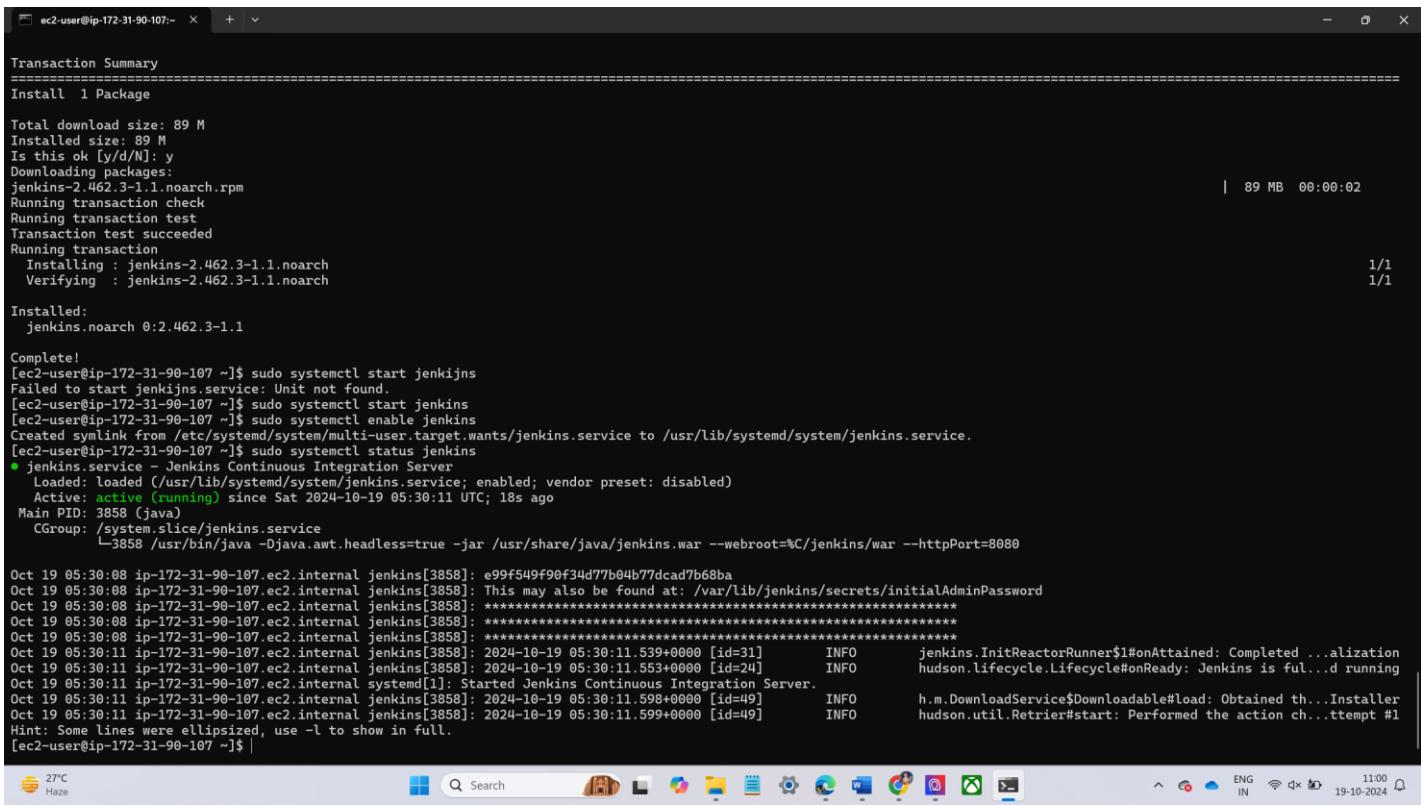
```
ec2-user@ip-172-31-90-107:~ + Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jawah>cd Downloads

C:\Users\jawah\Downloads>ssh -i "veera.pem" ec2-user@ec2-18-205-150-72.compute-1.amazonaws.com
The authenticity of host 'ec2-18-205-150-72.compute-1.amazonaws.com (18.205.150.72)' can't be established.
ED25519 key fingerprint is SHA256:9hMFHj4sr4bLXndisEM/NpP05vUNSUGJgieSbsmNuU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-205-150-72.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#_
  _\###_      Amazon Linux 2
  _\######_\
  \##|       AL2 End of Life is 2025-06-30.
  #/ ---_
  V=+-->
  /     A newer version of Amazon Linux is available!
  ./. /    Amazon Linux 2023, GA and supported until 2028-03-15.
  ./m/   https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-90-107 ~]$ |
```

## ◆ Now install Jenkins and set up Jenkins



```
Transaction Summary
=====
Install 1 Package

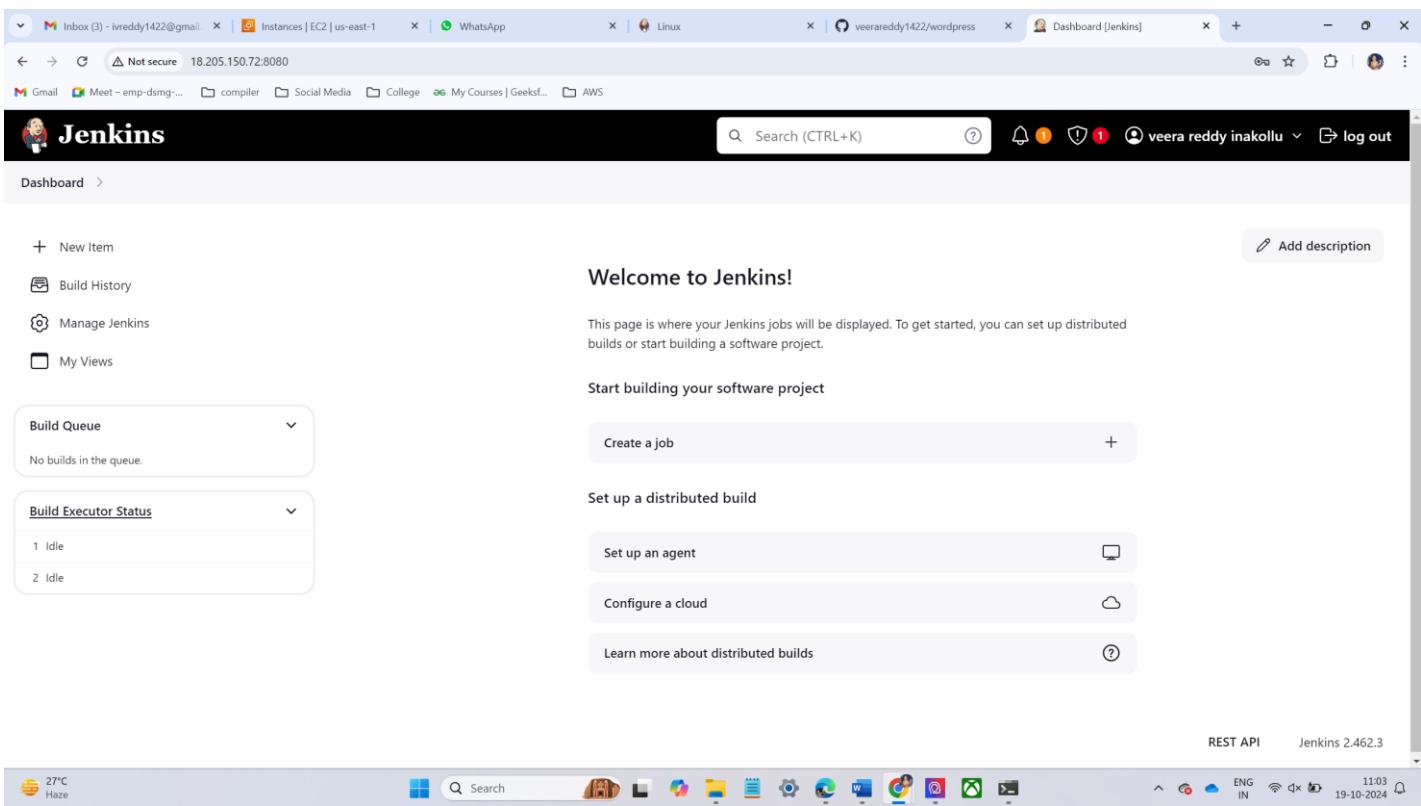
Total download size: 89 M
Installed size: 89 M
Is this ok [y/d/N]: y
Downloading packages:
jenkins-2.462.3-1.1.noarch.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.462.3-1.1.noarch
  Verifying   : jenkins-2.462.3-1.1.noarch
                                           1/1
                                           1/1

Installed:
  jenkins.noarch 0:2.462.3-1.1

Complete!
[ec2-user@ip-172-31-90-107 ~]$ sudo systemctl start jenkins
Failed to start jenkins.service: Unit not found.
[ec2-user@ip-172-31-90-107 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-90-107 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-90-107 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
    Active: active (running) since Sat 2024-10-19 05:30:11 UTC; 18s ago
      Main PID: 3858 (java)
        CGroup: /system.slice/jenkins.service
               └─3858 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

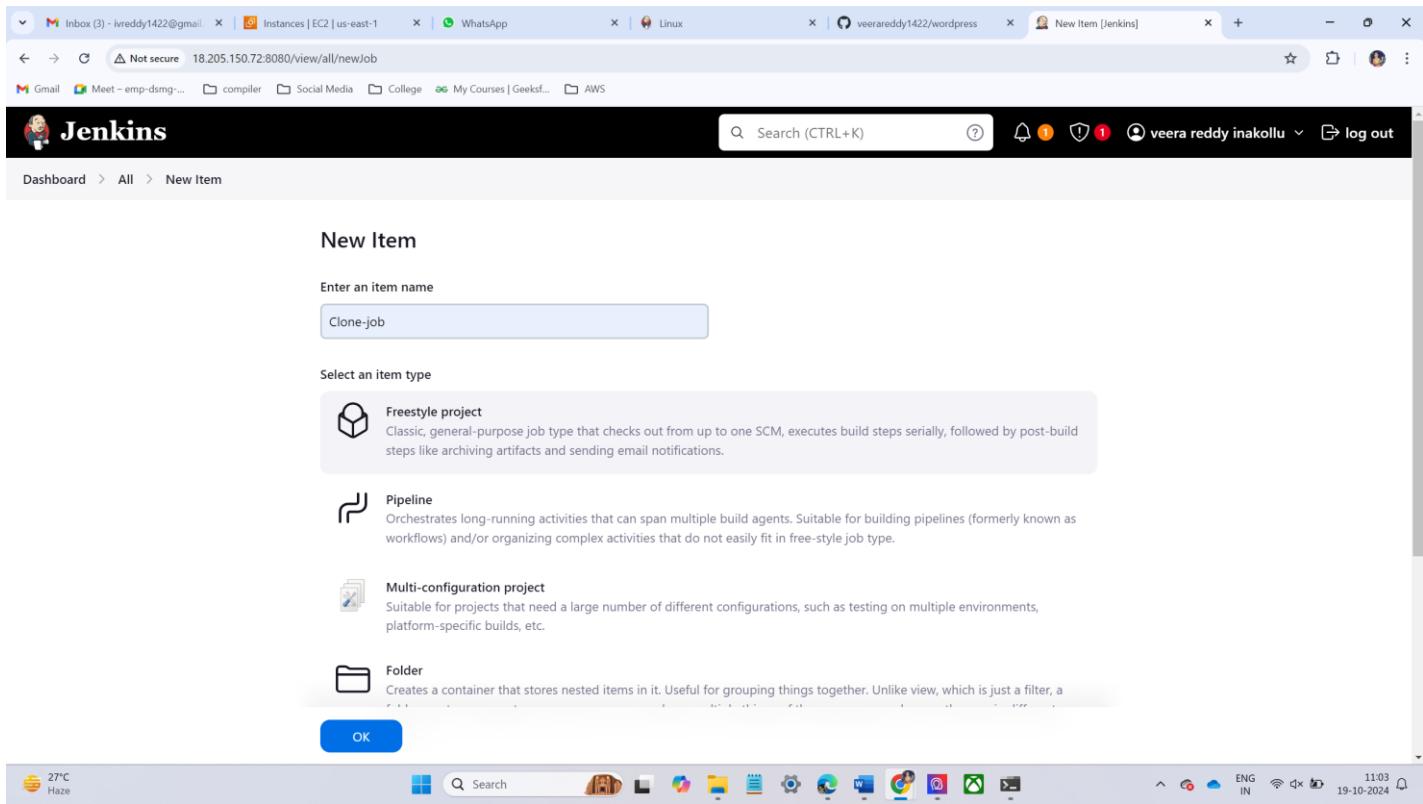
Oct 19 05:30:08 ip-172-31-90-107.ec2.internal jenkins[3858]: e99f549f90f34d77b04b77dcad7b68ba
Oct 19 05:30:08 ip-172-31-90-107.ec2.internal jenkins[3858]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Oct 19 05:30:08 ip-172-31-90-107.ec2.internal jenkins[3858]: ****
Oct 19 05:30:08 ip-172-31-90-107.ec2.internal jenkins[3858]: ****
Oct 19 05:30:08 ip-172-31-90-107.ec2.internal jenkins[3858]: ****
Oct 19 05:30:11 ip-172-31-90-107.ec2.internal jenkins[3858]: 2024-10-19 05:30:11.539+0000 [id=31]      INFO      jenkins.InitReactorRunner$1#onAttained: Completed ...alization
Oct 19 05:30:11 ip-172-31-90-107.ec2.internal jenkins[3858]: 2024-10-19 05:30:11.553+0000 [id=24]      INFO      hudson.lifecycle.Lifecycle#onReady: Jenkins is ful...d running
Oct 19 05:30:11 ip-172-31-90-107.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Oct 19 05:30:11 ip-172-31-90-107.ec2.internal jenkins[3858]: 2024-10-19 05:30:11.598+0000 [id=49]      INFO      h.m.DownloadService$Downloadable#load: Obtained th...Installer
Oct 19 05:30:11 ip-172-31-90-107.ec2.internal jenkins[3858]: 2024-10-19 05:30:11.599+0000 [id=49]      INFO      hudson.util.Retriger#start: Performed the action ch...attempt #1
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-90-107 ~]$ |
```

## ◆ Now launch the Jenkins sever using ipv4 address along with port number 8080.



The screenshot shows the Jenkins dashboard at <http://18.205.150.72:8080>. The page title is "Jenkins". The dashboard features a "Welcome to Jenkins!" message and a "Start building your software project" button. On the left, there are links for "New item", "Build History", "Manage Jenkins", and "My Views". Below these are two dropdown menus: "Build Queue" (empty) and "Build Executor Status" (showing 1 Idle and 2 Idle). The main right-hand area contains sections for "Set up a distributed build" with options for "Set up an agent" and "Configure a cloud", and a link to "Learn more about distributed builds". At the bottom right, it says "REST API Jenkins 2.462.3".

- ◆ Create job for clone the code
- ◆ Click on "New Item".
- ◆ Enter a name for your project (e.g., WordPress-Deployment).
- ◆ Select "Freestyle project" and click OK.



- ◆ Configure Source Code Management (SCM)
- ◆ In the project configuration page, scroll down to the "**Source Code Management**" section.
- ◆ Select "**Git**".
- ◆ Enter the **Repository URL** of your GitHub repository

The screenshot shows the Jenkins 'Configure' screen for a job named 'Clone-job'. On the left, a sidebar lists 'General', 'Source Code Management' (which is selected), 'Build Triggers', 'Build Environment', 'Build Steps', and 'Post-build Actions'. The main area is titled 'Repositories' and contains a 'Repository URL' field with the value 'https://github.com/veerareddy1422/wordpress.git'. Below it is a 'Credentials' dropdown set to '- none -'. A 'Branches to build' section shows a 'Branch Specifier' field with the value '/main'. At the bottom are 'Save' and 'Apply' buttons.

- ◆ Create job for build the code
- ◆ Click on "New Item".
- ◆ Enter a name for your project (e.g., WordPress-Deployment).
- ◆ Select "Freestyle project" and click OK.

The screenshot shows the Jenkins 'New Item' creation interface. The title bar says 'New Item [Jenkins]'. The main area has a 'New Item' heading and a 'Search (CTRL+K)' input field. A 'Build-job' entry is in a text input field. Below it, a 'Select an item type' section shows three options: 'Freestyle project' (selected), 'Pipeline', and 'Multi-configuration project'. Each option has a description and a small icon. At the bottom is an 'OK' button.

- ◆ Configure Build steps
- ◆ In the project configuration page, scroll down to the "Build steps" section.
- ◆ Select "Execute shell".
- ◆ Write the bash script

The screenshot shows the Jenkins interface for configuring a build job. The left sidebar has 'Build Steps' selected. The main area shows an 'Execute shell' step with the following command:

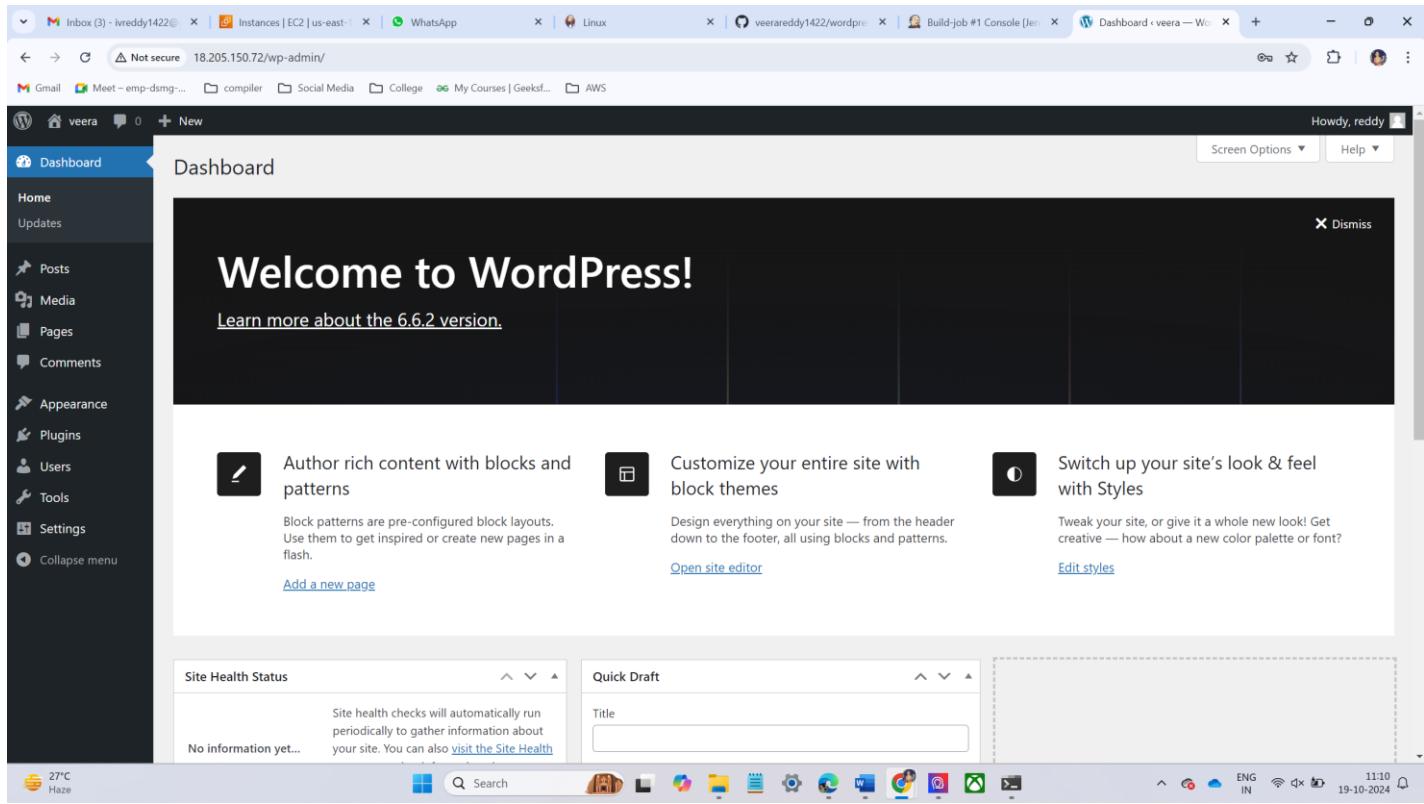
```
#!/bin/bash
sudo yum -y install git
sudo yum -y install docker
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
sudo usermod -aG docker jenkins
sudo chmod 666 /var/run/docker.sock
sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
git clone https://github.com/veerareddy1422/wordpress.git
cd wordpress/
docker-compose up -d
```

At the bottom, there are 'Save' and 'Apply' buttons.

- ◆ Now browse the public ipv4 address on new tab. The word press application will launch

The screenshot shows a web browser displaying the WordPress installation process. A dropdown menu lists various languages, with 'English (United States)' selected. Other options include Afrikaans, Aragonés, العربية, العربية المغربية, অসমীয়া, گۈنئى ئۇيغۇرچىسى, Azerbaijani, Bosnian, Català, Cebuano, Čeština, Cymraeg, Dansk, Deutsch (Sie), and Deutsch (Schweiz, Du). A 'Continue' button is visible at the bottom of the dropdown.

- ◆ WordPress application successfully deployed using Jenkins bash script



# Deploy WordPress web application by using git and jenkins execute shell (bash script) create jenkins pipeline add build periodically and poll scm to initial job of pipeline and check the changes happened or not which are made in github repo?

- ◆ Create pipeline for WordPress application for continue in the previous method
- ◆ Install Jenkins pipeline plugin

The screenshot shows the Jenkins Plugins page. On the left, there's a sidebar with links: Updates, Available plugins, Installed plugins, Advanced settings, and Download progress (which is selected). The main area is titled "Download progress" and shows the status of several plugins:

Plugin	Status
Preparation	Success
Parameterized Trigger	Success
JavaMail API	Success
Oracle Java SE Development Kit Installer	Success
SSH server	Success
Command Agent Launcher	Success
jQuery	Success
Build Pipeline	Success
Loading plugin extensions	Success

At the bottom, there are two buttons: "Go back to the top page" and "Restart Jenkins when installation is complete and no jobs are running".

- ◆ Now create pipeline for wordpress application
- ◆ Goto dashboard → under dashboard click + icon
- ◆ Enter pipeline name
- ◆ Select build pipeline view

New view

Name: wordpress

Type: Build Pipeline View

Build Queue: No builds in the queue.

Build Executor Status: 1 Idle, 2 Idle

Create

- ◆ Configuration the pipeline
- ◆ Under pipeline flow select your initial job

Pipeline Flow

Layout: Based on upstream/downstream relationship

Upstream / downstream config

Select Initial Job: Clone-job

Trigger Options

Build Cards: Standard build card

OK Apply

- ◆ Under display options select **No Of displayed build** (your wish)

The screenshot shows the 'Display Options' configuration page in Jenkins. The 'No Of Displayed Builds' dropdown is set to 3. Other settings include 'Row Headers' (set to 'Just the pipeline number') and 'Column Headers' (set to 'No header'). The 'Refresh frequency (in seconds)' is also set to 3. There is a 'URL for custom CSS files' input field, an 'OK' button, and an 'Apply' button.

- ◆ Now configure the clone job
- ◆ Under clone job configuration go to post-build actions
- ◆ Under post build action select build other project
- ◆ Now select the build job

The screenshot shows the 'Clone-job Configuration' page in Jenkins. The 'Build Steps' section is selected in the sidebar. In the main area, under 'Post-build Actions', there is a 'Build other projects' section. It contains a 'Projects to build' input field with 'Build-job,' and three trigger options: 'Trigger only if build is stable' (selected), 'Trigger even if the build is unstable', and 'Trigger even if the build fails'. There is also an 'Add post-build action' dropdown and 'Save' and 'Apply' buttons at the bottom.

- ◆ Now configure the build job
- ◆ Under build job configuration go to build triggers
- ◆ Under build triggers select build after other projects are built
- ◆ Now select the clone job

The screenshot shows the Jenkins 'Build-job Configuration' page for a job named 'Clone-job'. The 'Build Triggers' section is active. Under 'Build Triggers', the 'Build after other projects are built' checkbox is checked. In the 'Projects to watch' dropdown, 'Clone-job' is selected. Below this, the 'Trigger only if build is stable' radio button is selected. There are also other trigger options like 'Trigger even if the build is unstable', 'Trigger even if the build fails', and 'Always trigger, even if the build is aborted'. Other trigger options like 'Build periodically', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' are available but not checked.

- ◆ Now apply build trigger action on clone job

The screenshot shows the Jenkins 'Clone-job Configuration' page. The 'Build Triggers' section is active. Under 'Build Triggers', the 'Build periodically' checkbox is checked. In the 'Schedule' dropdown, there is a single asterisk (\*) entered. A warning message '⚠ No schedules so will never run' is displayed below the schedule input field. Other trigger options like 'Trigger builds remotely', 'Build after other projects are built', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' are available but not checked.

- ◆ Now go to pipeline check build pipeline actions

- ◆ It runs every minute

The screenshot shows the Jenkins Build Pipeline interface. There are three parallel pipelines listed vertically:

- Pipeline #5:** Contains a green box for '#5 Clone-job' (Oct 19, 2024 6:02:00 AM, 0.14 sec) and a green box for '#8 Build-job' (Oct 19, 2024 6:02:17 AM, 4.3 sec, user ivreddy1422).
- Pipeline #4:** Contains a green box for '#4 Clone-job' (Oct 19, 2024 6:01:00 AM, 0.13 sec) and a green box for '#6 Build-job' (Oct 19, 2024 6:01:27 AM, 4.7 sec, user ivreddy1422).
- Pipeline #3:** Contains a green box for '#3 Clone-job' (Oct 19, 2024 6:00:00 AM, 0.14 sec) and a green box for '#4 Build-job' (Oct 19, 2024 6:00:07 AM, 4.6 sec, user ivreddy1422).

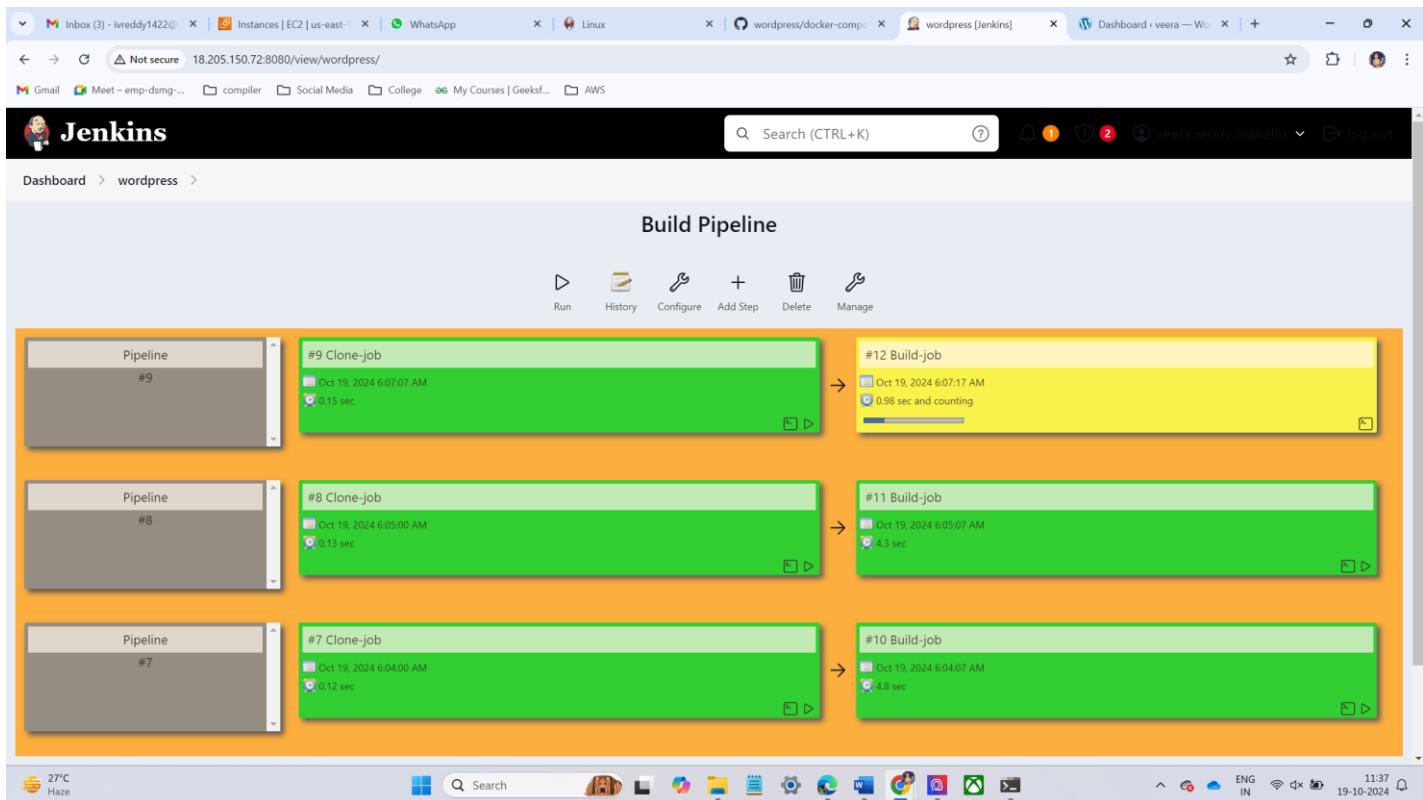
Each pipeline has a sidebar on the left labeled with its name and number. The Jenkins logo is at the top left, and a search bar is at the top right.

- ◆ Now apply poll SCM for clone job

The screenshot shows the 'Clone-job Config [Jenkins]' configuration page. The 'Build Triggers' section is active, with the 'Poll SCM' option checked. The schedule field is empty, and the note 'No schedules so will only run due to SCM changes if triggered by a post-commit hook' is displayed. Other options like 'Trigger builds remotely' and 'Build after other projects are built' are also present but unchecked.

On the left sidebar, the 'Build Triggers' option is highlighted. At the bottom, there are 'Save' and 'Apply' buttons.

- ◆ Now go to pipeline check build pipeline actions
- ◆ When you are change the code in git hub it automatically runs



- ◆ In this time I am change the database name

The screenshot shows a GitHub Copilot interface editing a `docker-compose.yaml` file for a `wordpress` project. The code is as follows:

```

version: '3.8'
services:
  db:
    image: mysql:8.0.19
    command: '--default-authentication-plugin=mysql_native_password'
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD=veerareddy
      - MYSQL_DATABASE=veerareddy
      - MYSQL_USER=reddyinakolu
      - MYSQL_PASSWORD=susmitha

  wordpress:
    image: wordpress:latest
    ports:
      - "80:80"
    restart: always
    environment:
      - WORDPRESS_DB_HOST=db
      - WORDPRESS_DB_USER=reddyinakolu
      - WORDPRESS_DB_PASSWORD=susmitha
      - WORDPRESS_DB_NAME=veerareddy

volumes:
  db_data:

```

The GitHub Copilot interface includes buttons for **Edit**, **Preview**, and **Code 55% faster with GitHub Copilot**. There are also buttons for **Cancel changes** and **Commit changes...**.

- ◆ Its runs automatically when I am changing the code

The screenshot shows the Jenkins Build Pipeline interface. At the top, there's a navigation bar with tabs like 'Dashboard' and 'wordpress'. Below it, the main title is 'Build Pipeline'. A toolbar with icons for 'Run', 'History', 'Configure', 'Add Step', 'Delete', and 'Manage' is visible.

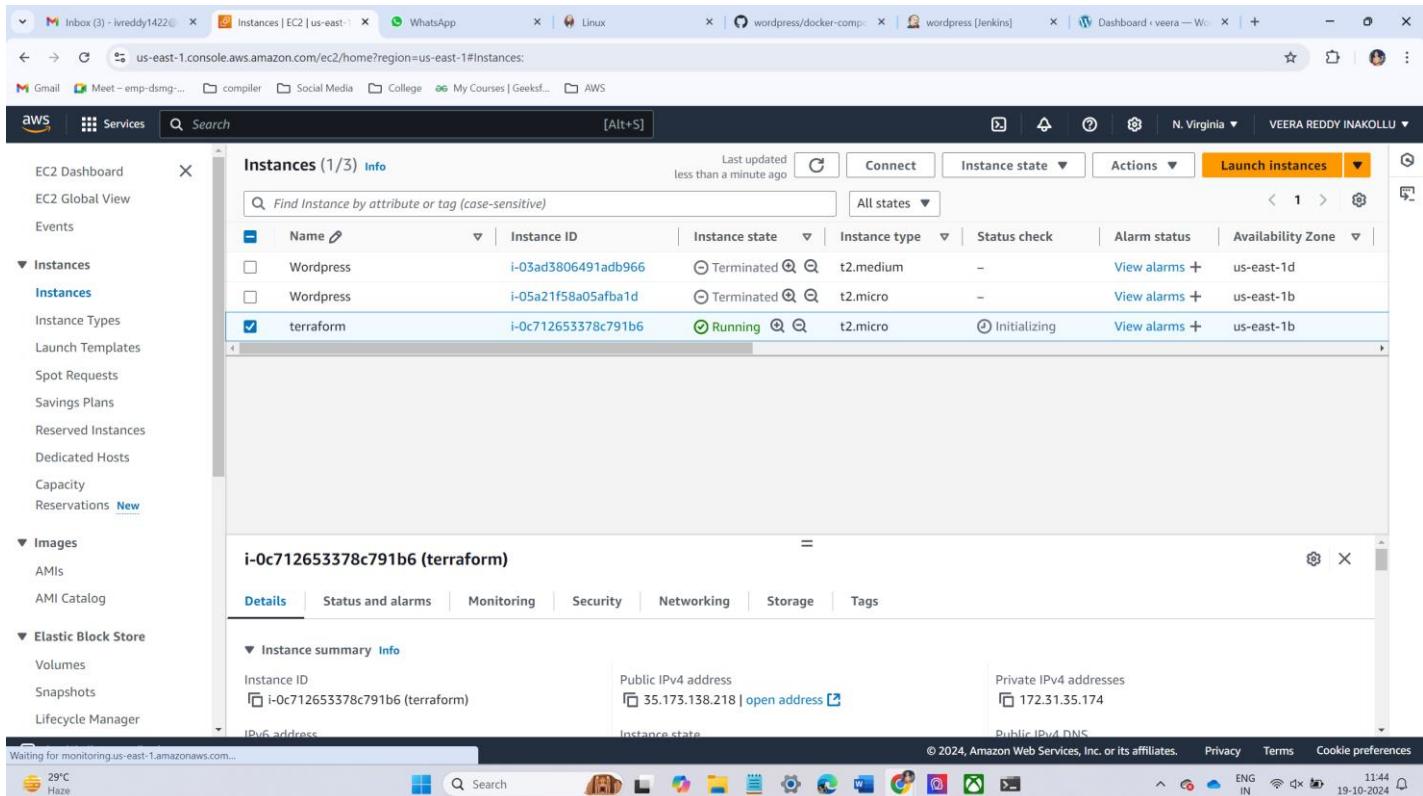
The pipeline structure is as follows:

- Pipeline #11:** Triggers a 'Clone-job' (#11) which then triggers a 'Build-job' (#11).
- Pipeline #10:** Triggers a 'Clone-job' (#10) which then triggers a 'Build-job' (#14).
- Pipeline #9:** Triggers a 'Clone-job' (#9) which then triggers a 'Build-job' (#12).

Each step in the pipeline is represented by a green rectangular card with details like date, time, duration, and user. The Jenkins interface includes a sidebar with 'Watchlist' and 'Ideas' sections, and a taskbar at the bottom with various application icons.

## Deploy WordPress web application by using terraform (create Ec2 instance along with userdata .sh file)

- ◆ Create ec2 instance for using terraform



- ◆ Connect ec2 instance to the command prompt

## ◆ Install terraform and set up the terraform

ec2-user@ip-172-31-35-174 ~ % + - x Microsoft Windows [Version 10.0.22631.4317] (c) Microsoft Corporation. All rights reserved. C:\Users\jawah>cd Downloads C:\Users\jawah\Downloads>ssh -i "veera.pem" ec2-user@ec2-35-173-138-218.compute-1.amazonaws.com The authenticity of host 'ec2-35-173-138-218.compute-1.amazonaws.com (35.173.138.218)' can't be established. ED25519 key fingerprint is SHA256:hvirTMEJGWOPgv9+h9XuhWNV/fpDaCe36vzG+Mle0. This key is not known by any other names. Are you sure you want to continue connecting (yes/no/[fingerprint])? yes Warning: Permanently added 'ec2-35-173-138-218.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

```
      #  
      _- #####_     Amazon Linux 2  
      _- \#####\|  
      _- \|###[|     AL2 End of Life is 2025-06-30.  
      _- \|#/  
      _- \|~,-_>  
      _- /| A newer version of Amazon Linux is available!  
      _- /| /| Amazon Linux 2023, GA and supported until 2028-03-15.  
      _- /| /| https://aws.amazon.com/linux/amazon-linux-2023/  
[ec2-user@ip-172-31-35-174 ~]$ sudo yum install terraform  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
No package terraform available.  
Error: Nothing to do  
[ec2-user@ip-172-31-35-174 ~]$ sudo yum install -y yum-utils  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
Package yum-utils-1.1.31-46.amzn2.0.1.noarch already installed and latest version  
Nothing to do  
[ec2-user@ip-172-31-35-174 ~]$ sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
adding repo from: https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo  
grabbing file https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo to /etc/yum.repos.d/hashicorp.repo  
repo saved to /etc/yum.repos.d/hashicorp.repo  
[ec2-user@ip-172-31-35-174 ~]$ sudo yum -y install terraform  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
hashicorp  
hashicorp/x86_64/primary  
hashicorp  
Resolving Dependencies  
--> Running transaction check  
--> Package terraform.x86_64 0:1.9.8-1 will be installed  
--> Processing Dependency: git for package: terraform-1.9.8-1.x86_64  
--> Running transaction check  
--> Package git.x86_64 0:2.40.1-1.amzn2.0.3 will be installed
```

| 1.5 kB 00:00:00  
| 290 kB 00:00:00  
2113/2113

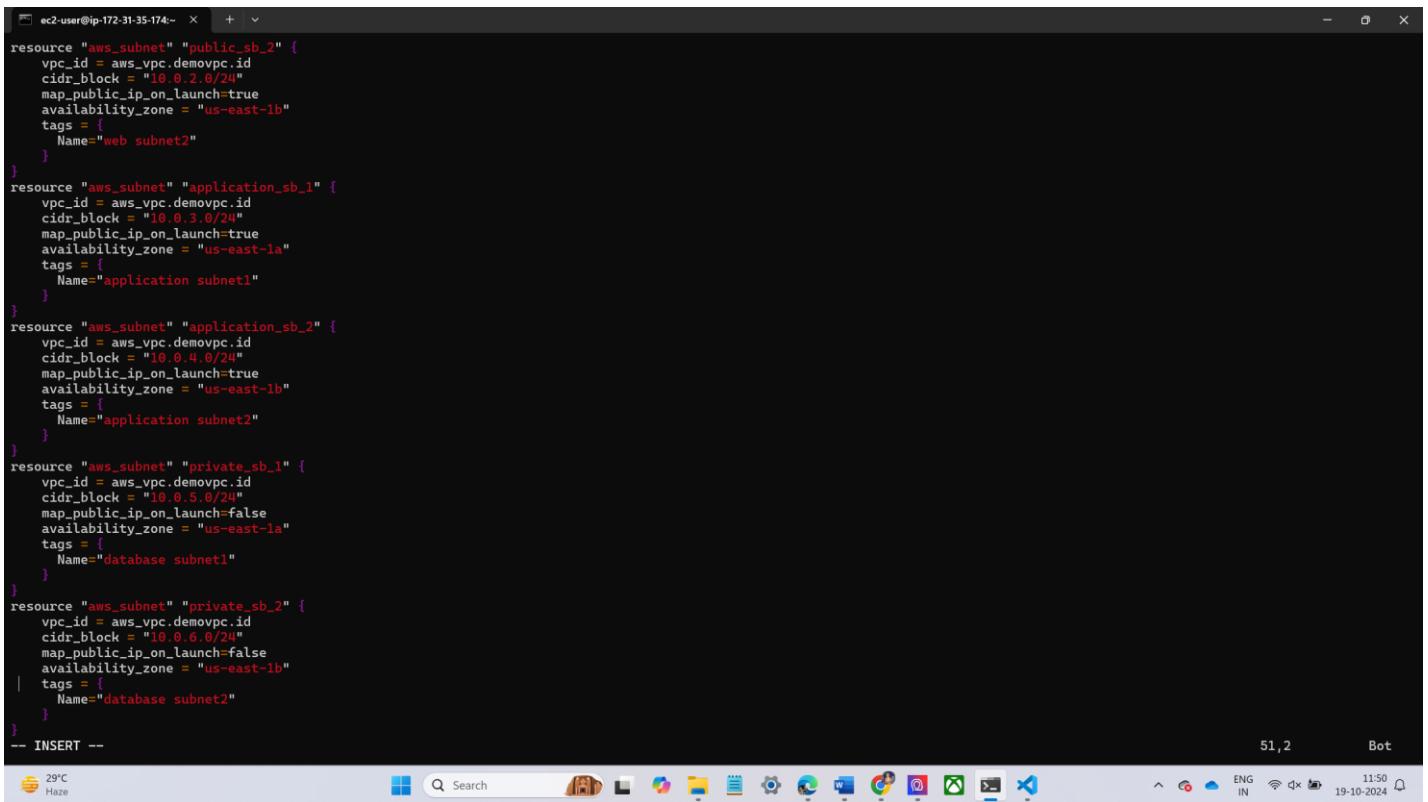
29°C Haze 11:47 ENG IN 19-10-2024

## ◆ Now write terraform script for vpc

```
provider "aws" {  
    region = "us-east-1"  
}  
#creating VPC  
resource "aws_vpc" "demovpc" {  
    cidr_block = var.vpc_cidr  
    instance_tenancy = "default"  
    tags = {  
        Name="Demo VPC"  
    }  
}  
:wq!
```

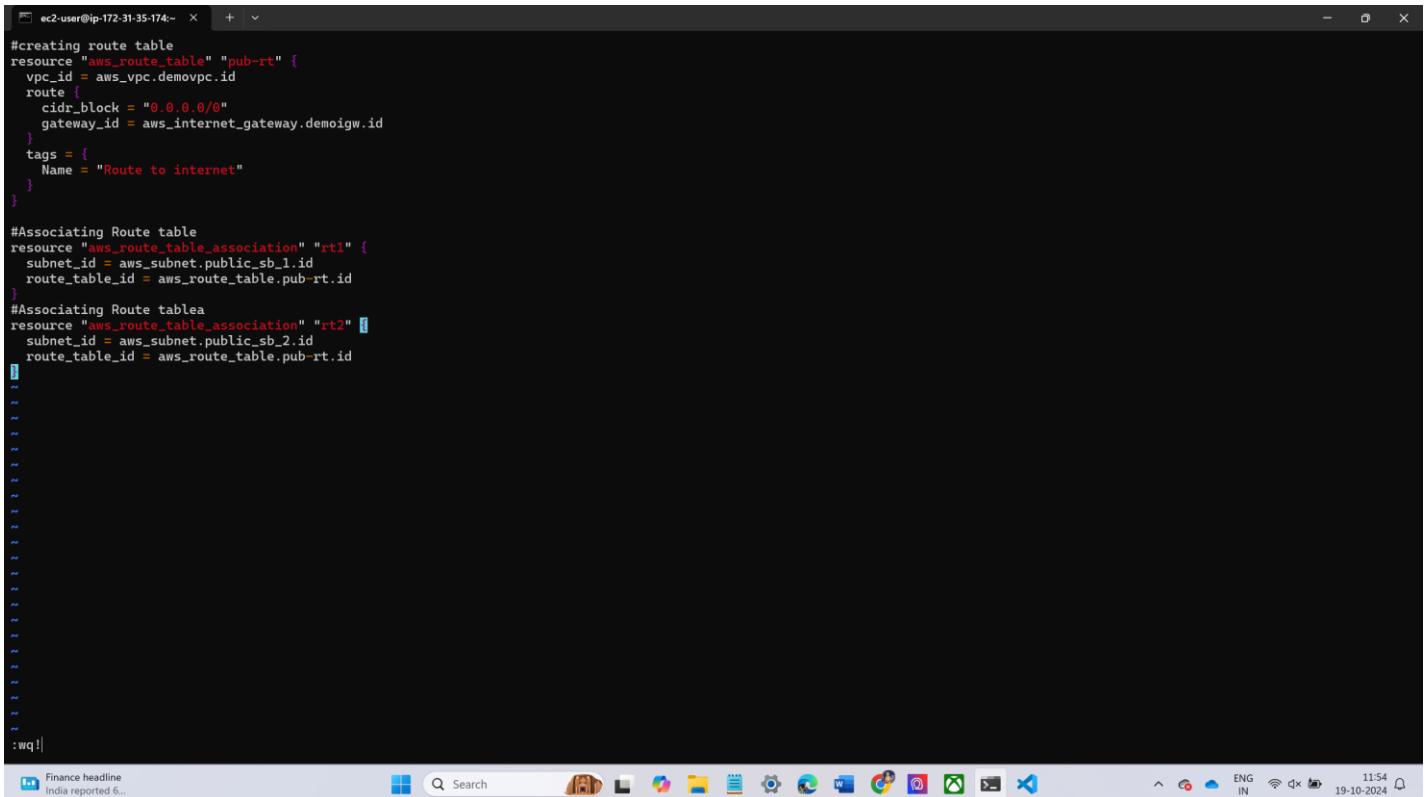
29°C Haze 11:49 ENG IN 19-10-2024

## ◆ Create subnets.tf



```
resource "aws_subnet" "public_sb_2" {
  vpc_id = aws_vpc.demovpcl.id
  cidr_block = "10.0.2.0/24"
  map_public_ip_on_launch=true
  availability_zone = "us-east-1b"
  tags = [
    Name="web subnet2"
  ]
}
resource "aws_subnet" "application_sb_1" {
  vpc_id = aws_vpc.demovpcl.id
  cidr_block = "10.0.3.0/24"
  map_public_ip_on_launch=true
  availability_zone = "us-east-1a"
  tags = [
    Name="application subnet1"
  ]
}
resource "aws_subnet" "application_sb_2" {
  vpc_id = aws_vpc.demovpcl.id
  cidr_block = "10.0.4.0/24"
  map_public_ip_on_launch=true
  availability_zone = "us-east-1b"
  tags = [
    Name="application subnet2"
  ]
}
resource "aws_subnet" "private_sb_1" {
  vpc_id = aws_vpc.demovpcl.id
  cidr_block = "10.0.5.0/24"
  map_public_ip_on_launch=false
  availability_zone = "us-east-1a"
  tags = [
    Name="database subnet1"
  ]
}
resource "aws_subnet" "private_sb_2" {
  vpc_id = aws_vpc.demovpcl.id
  cidr_block = "10.0.6.0/24"
  map_public_ip_on_launch=false
  availability_zone = "us-east-1b"
  tags = [
    Name="database subnet2"
  ]
}
-- INSERT --
```

## ◆ Create file for route table

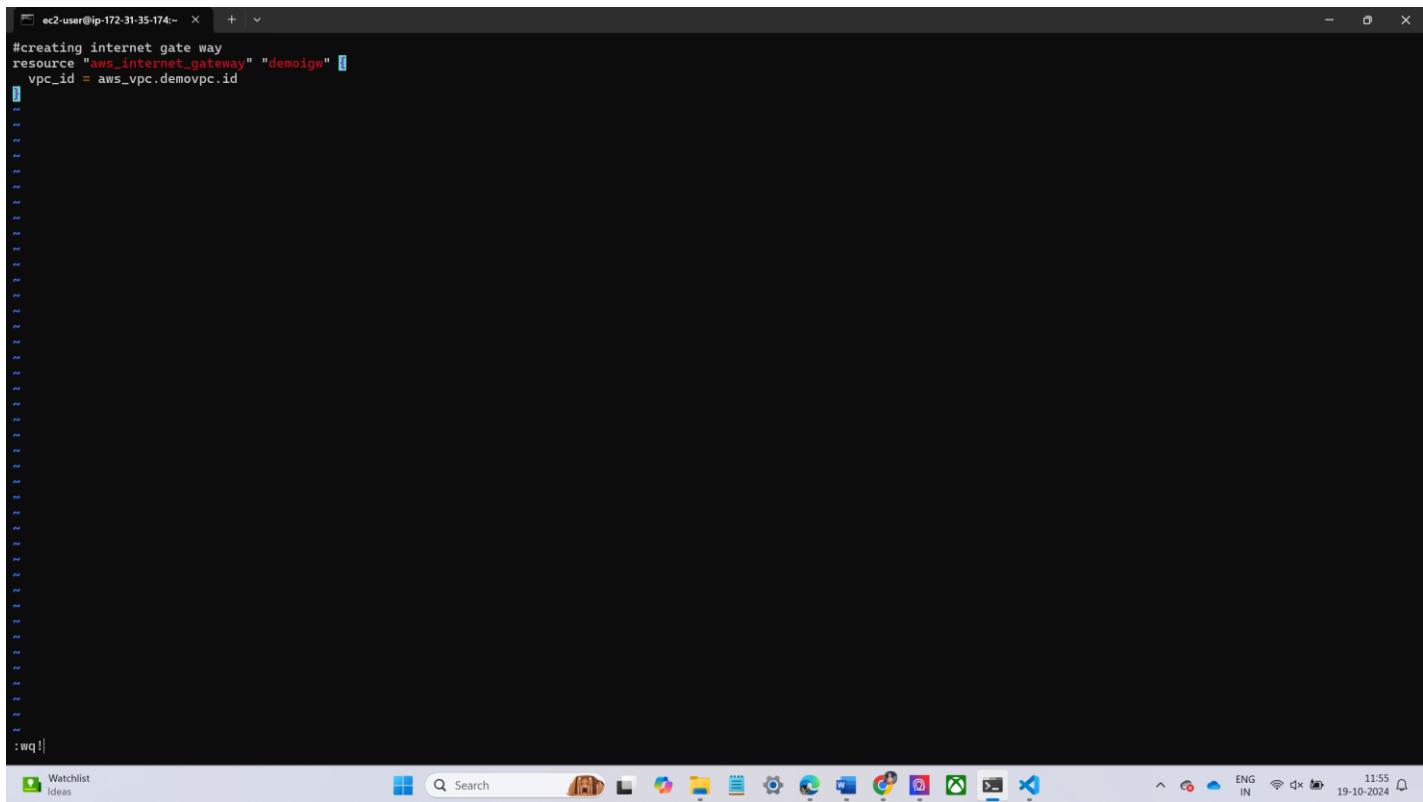


```
#creating route table
resource "aws_route_table" "pub-rt" {
  vpc_id = aws_vpc.demovpcl.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.demoigw.id
  }
  tags = [
    Name = "Route to internet"
  ]
}

#Associating Route table
resource "aws_route_table_association" "rtl" {
  subnet_id = aws_subnet.public_sb_1.id
  route_table_id = aws_route_table.pub-rt.id
}
#Associating Route tablea
resource "aws_route_table_association" "rt2" {
  subnet_id = aws_subnet.public_sb_2.id
  route_table_id = aws_route_table.pub-rt.id
}

:wq!
```

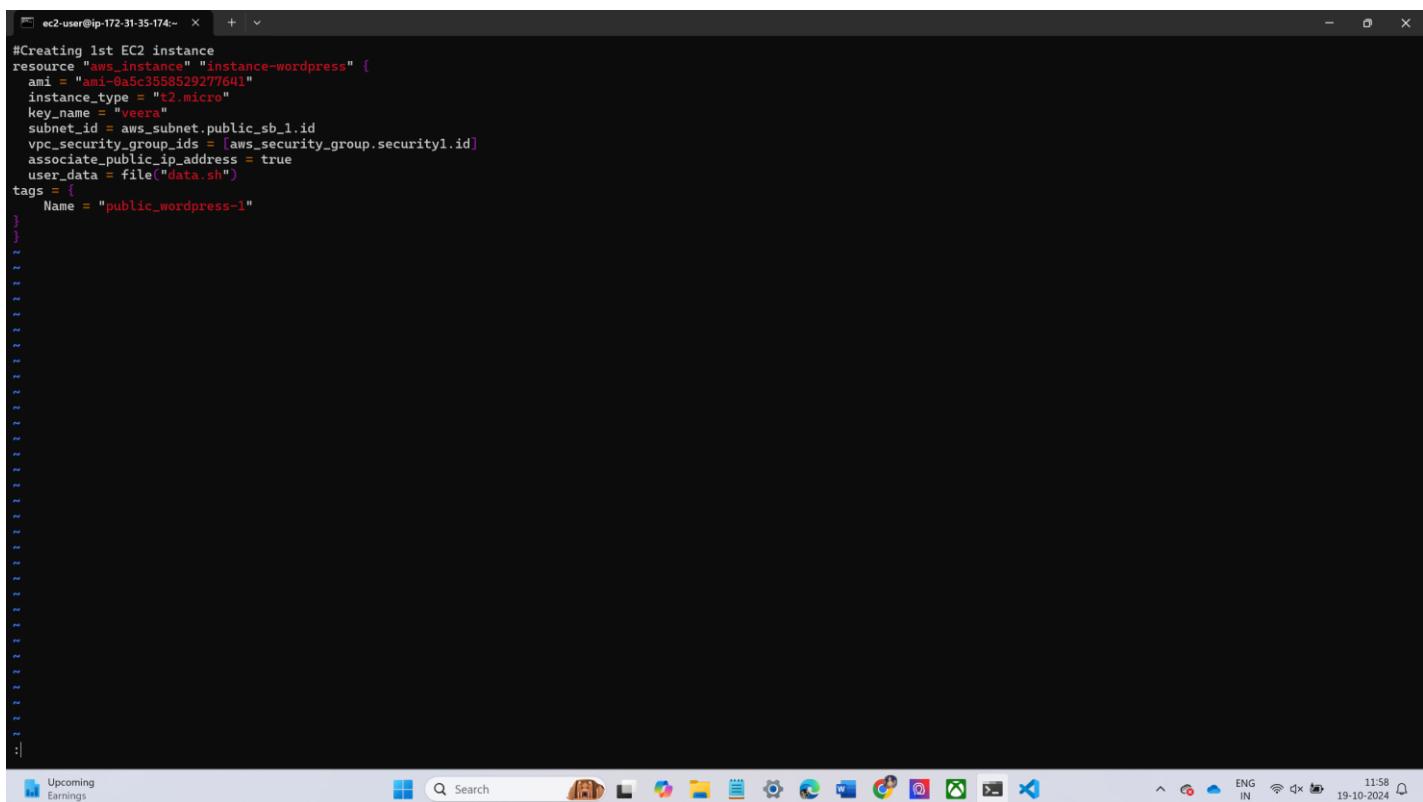
## ◆ Create file for internet gateway



```
#creating internet gate way
resource "aws_internet_gateway" "demoigw" {
  vpc_id = aws_vpc.demovpc.id
}

:wq!
```

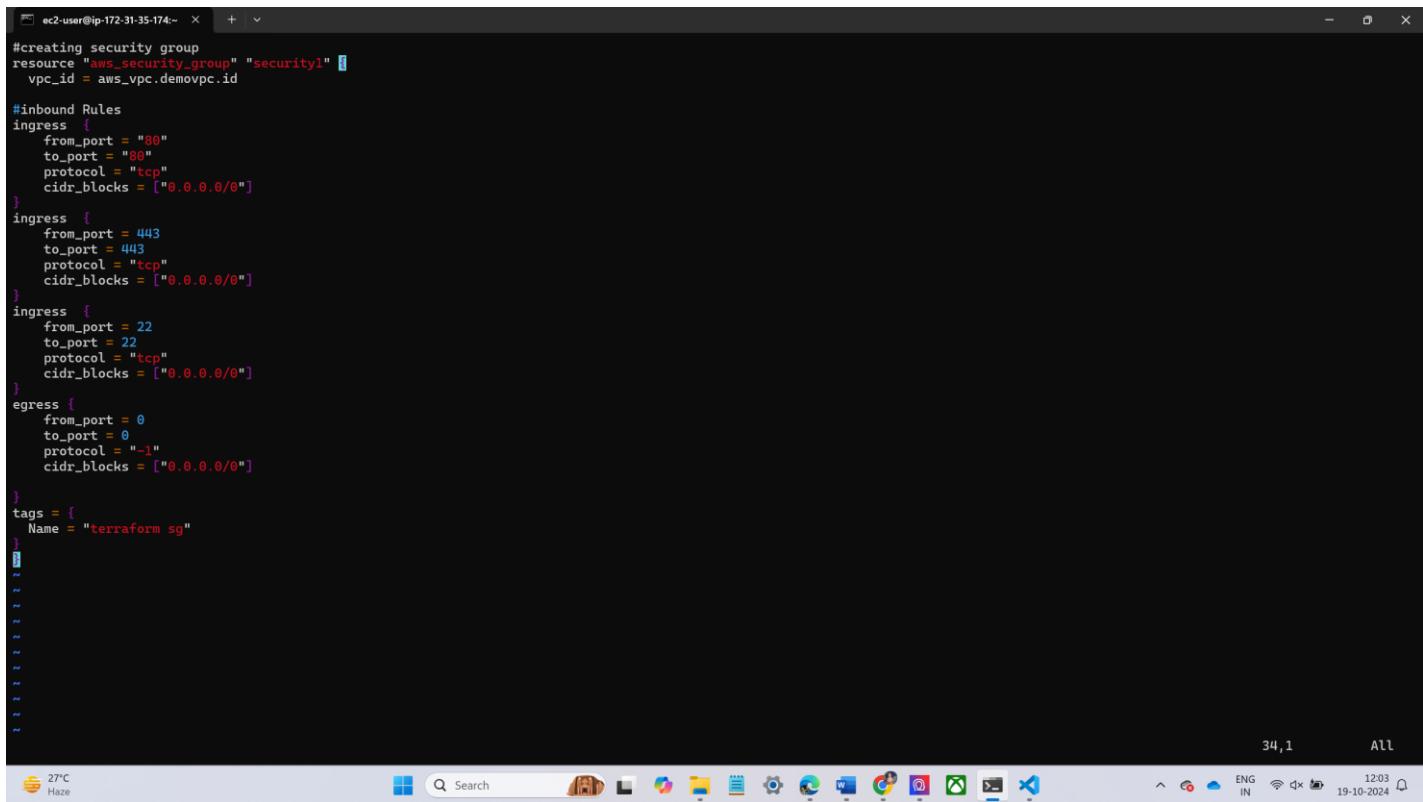
## ◆ Create file for ec2 instance



```
#Creating lsc EC2 instance
resource "aws_instance" "instance-wordpress" {
  ami = "ami-0a5c3588529277641"
  instance_type = "t2.micro"
  key_name = "vera"
  subnet_id = aws_subnet.public_sb_1.id
  vpc_security_group_ids = [aws_security_group.security1.id]
  associate_public_ip_address = true
  user_data = file("data.sh")
  tags = {
    Name = "public_wordpress-1"
  }
}

:wq!
```

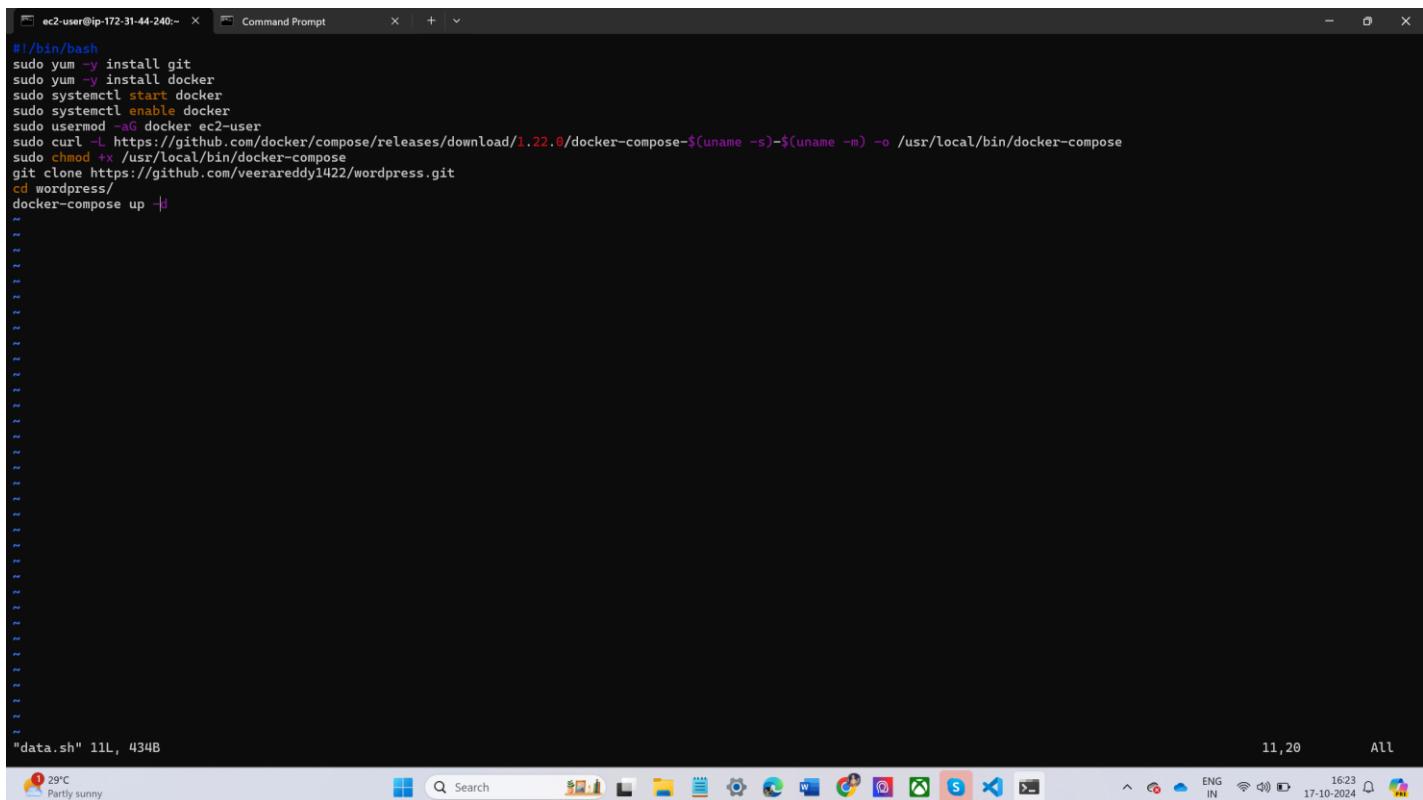
## ◆ create file for security groups



```
#creating security group
resource "aws_security_group" "security1" {
  vpc_id = aws_vpc.demovpc.id

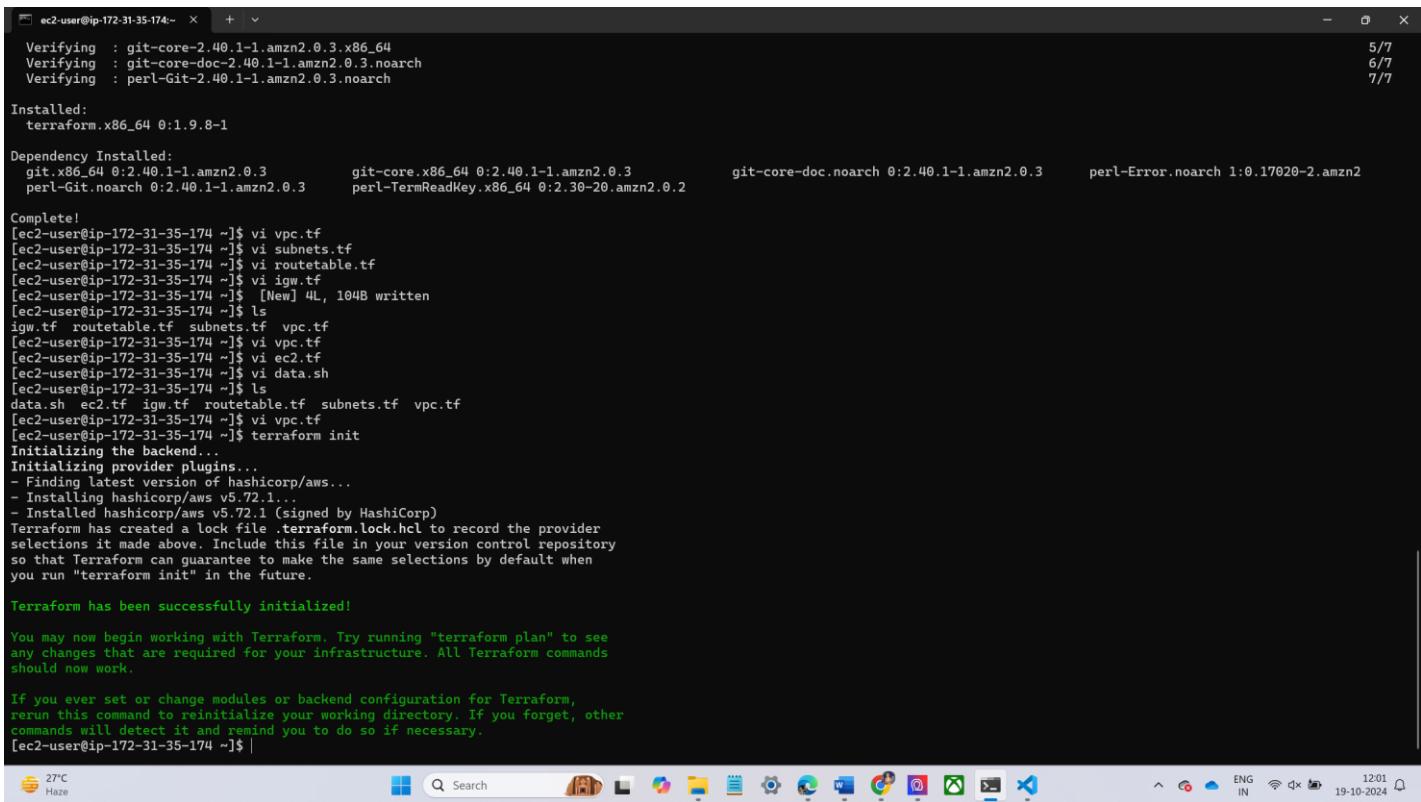
  #inbound Rules
  ingress {
    from_port = "80"
    to_port = "80"
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port = 443
    to_port = 443
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port = 22
    to_port = 22
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port = 0
    to_port = 0
    protocol = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags = {
    Name = "terraform_sg"
  }
}
```

## ◆ create file for data script



```
#!/bin/bash
sudo yum -y install git
sudo yum -y install docker
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
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
sudo chmod +x /usr/local/bin/docker-compose
git clone https://github.com/veerareddy1422/wordpress.git
cd wordpress/
docker-compose up -d
```

- ◆ After creation of all files apply terraform init command



```

[ec2-user@ip-172-31-35-174 ~]$ terraform init
Verifying : git-core-2.40.1-1.amzn2.0.3.x86_64 5/7
Verifying : git-core-doc-2.40.1-1.amzn2.0.3.noarch 6/7
Verifying : perl-Git-2.40.1-1.amzn2.0.3.noarch 7/7

Installed:
  terraform.x86_64 0:1.9.8-1

Dependency Installed:
  git.x86_64 0:2.40.1-1.amzn2.0.3      git-core.x86_64 0:2.40.1-1.amzn2.0.3      git-core-doc.noarch 0:2.40.1-1.amzn2.0.3      perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2      perl-Error.noarch 1:0.17020-2.amzn2

Complete!
[ec2-user@ip-172-31-35-174 ~]$ vi vpc.tf
[ec2-user@ip-172-31-35-174 ~]$ vi subnets.tf
[ec2-user@ip-172-31-35-174 ~]$ vi routetable.tf
[ec2-user@ip-172-31-35-174 ~]$ vi igw.tf
[ec2-user@ip-172-31-35-174 ~]$ [New] 4L, 104B written
[ec2-user@ip-172-31-35-174 ~]$ ls
igw.tf routetable.tf subnets.tf vpc.tf
[ec2-user@ip-172-31-35-174 ~]$ vi vpc.tf
[ec2-user@ip-172-31-35-174 ~]$ vi ec2.tf
[ec2-user@ip-172-31-35-174 ~]$ vi data.sh
[ec2-user@ip-172-31-35-174 ~]$ ls
data.sh ec2.tf igw.tf routetable.tf subnets.tf vpc.tf
[ec2-user@ip-172-31-35-174 ~]$ vi vpc.tf
[ec2-user@ip-172-31-35-174 ~]$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.72.1...
- Installed hashicorp/aws v5.72.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

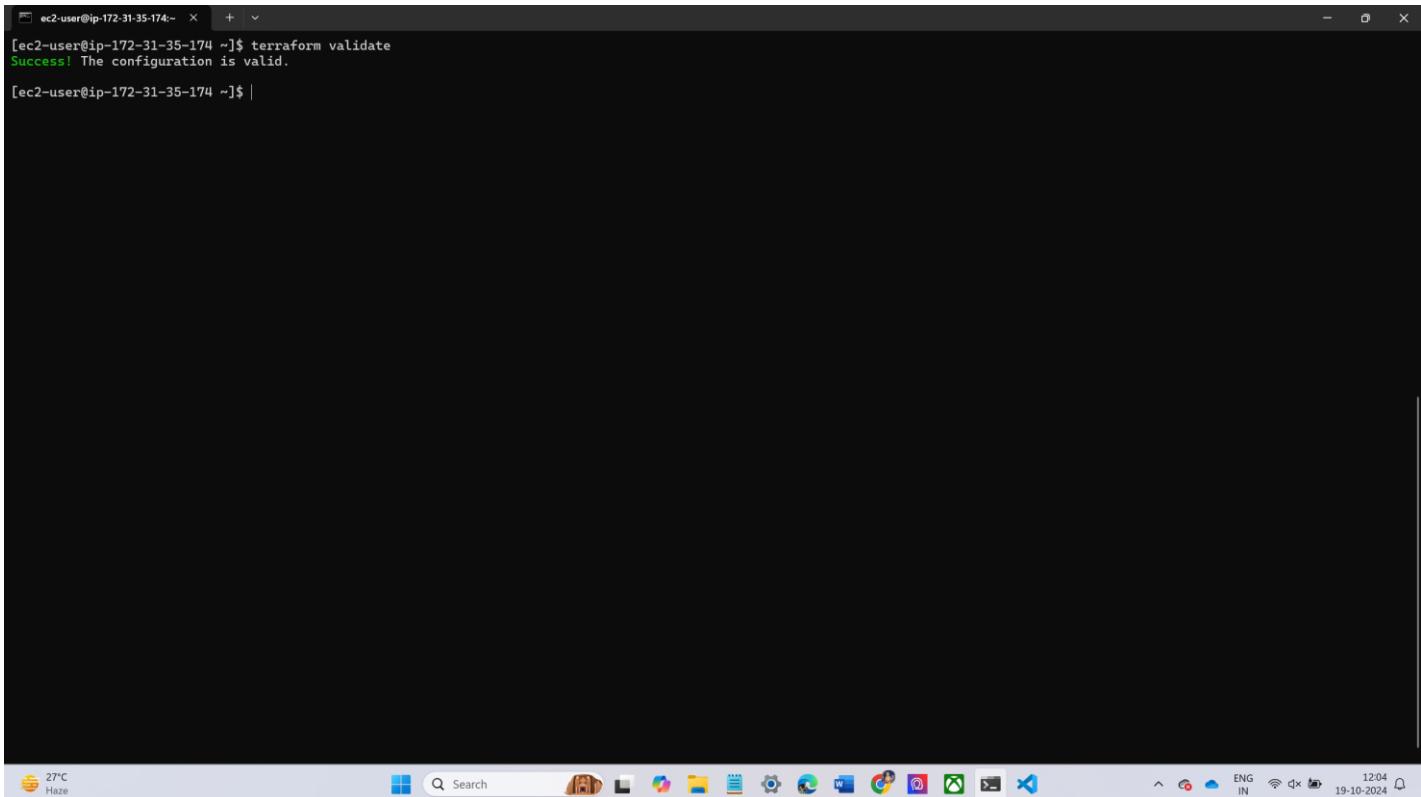
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.
[ec2-user@ip-172-31-35-174 ~]$ |

```

- ◆ Now check the errors for using terraform validate



```

[ec2-user@ip-172-31-35-174 ~]$ terraform validate
Success! The configuration is valid.

[ec2-user@ip-172-31-35-174 ~]$ |

```

- ◆ Now create the resource for using terraform apply command

```

ec2-user@ip-172-31-35-174: ~ + 
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id
+ owner_id
+ tags
+ tags_all
+ tags_all["Name"] = "Demo VPC"
}

plan: 13 to add, 0 to change, 0 to destroy.

aws_vpc.demoVPC: Creating...
aws_vpc.demoVPC: Creation complete after 1s [id=vpc-034f294206593520d]
aws_subnet.public_sb_2: Creating...
aws_subnet.private_sb_2: Creating...
aws_subnet.private_sb_1: Creating...
aws_subnet.application_sb_1: Creating...
aws_security_group.security1: Creating...
aws_subnet.public_sb_1: Creating...
aws_internet_gateway.demoIGW: Creating...
aws_subnet.application_sb_2: Creating...
aws_internet_gateway.demoIGW: Creation complete after 0s [id=igw-0074618009df903a09]
aws_route_table.pub-rt: Creating...
aws_subnet.private_sb_2: Creation complete after 1s [id=subnet-072e569d590959c8]
aws_subnet.private_sb_1: Creation complete after 1s [id=subnet-0c9c21c20bddf89c8]
aws_route_table.pub-rt: Creation complete after 1s [id=rtb-0acb13912a444da1b]
aws_security_group.security1: Creation complete after 3s [id=sg-0f2908ee669c7a5b1]
aws_subnet.public_sb_2: Still creating... [10s elapsed]
aws_subnet.application_sb_1: Still creating... [10s elapsed]
aws_subnet.public_sb_1: Still creating... [10s elapsed]
aws_subnet.application_sb_2: Still creating... [10s elapsed]
aws_subnet.public_sb_2: Creation complete after 11s [id=subnet-09ef6e6ff5feb925]
aws_subnet.application_sb_2: Creation complete after 11s [id=subnet-0e953ad8774c07827]
aws_route_table_association.rt2: Creating...
aws_subnet.application_sb_1: Creation complete after 11s [id=subnet-0c6cea5b79660fd0a]
aws_route_table_association.rt2: Creation complete after 0s [id=rtbassoc-058357f7871bf01b9]
aws_subnet.public_sb_1: Creation complete after 11s [id=subnet-05723aa59a6f9d25d]
aws_instance.instance-wordpress: Creating...
aws_route_table_association.rt1: Creating...
aws_route_table_association.rt1: Creation complete after 1s [id=rtbassoc-078d35f8822e25853]
aws_instance.instance-wordpress: Still creating... [10s elapsed]
aws_instance.instance-wordpress: Creation complete after 13s [id=i-0ec10ccff1fe547b5]

Apply complete! Resources: 13 added, 0 changed, 0 destroyed.
[ec2-user@ip-172-31-35-174 ~]$ 

```

27°C Haze 12:05 IN 19-10-2024

- ◆ Now check the aws ec2 console.

- ◆ New ec2 instance was created

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Wordpress	i-03ad3806491adb966	Terminated	t2.medium	-	<a href="#">View alarms</a>	us-east-1d
public_wordpress-1	i-0ec10ccff1fe547b5	Running	t2.micro	Initializing	<a href="#">View alarms</a>	us-east-1a
terraform	i-0c712653378c791b6	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1b

**i-0ec10ccff1fe547b5 (public\_wordpress-1)**

**Details** | Status and alarms | Monitoring | Security | Networking | Storage | Tags

**Instance summary**

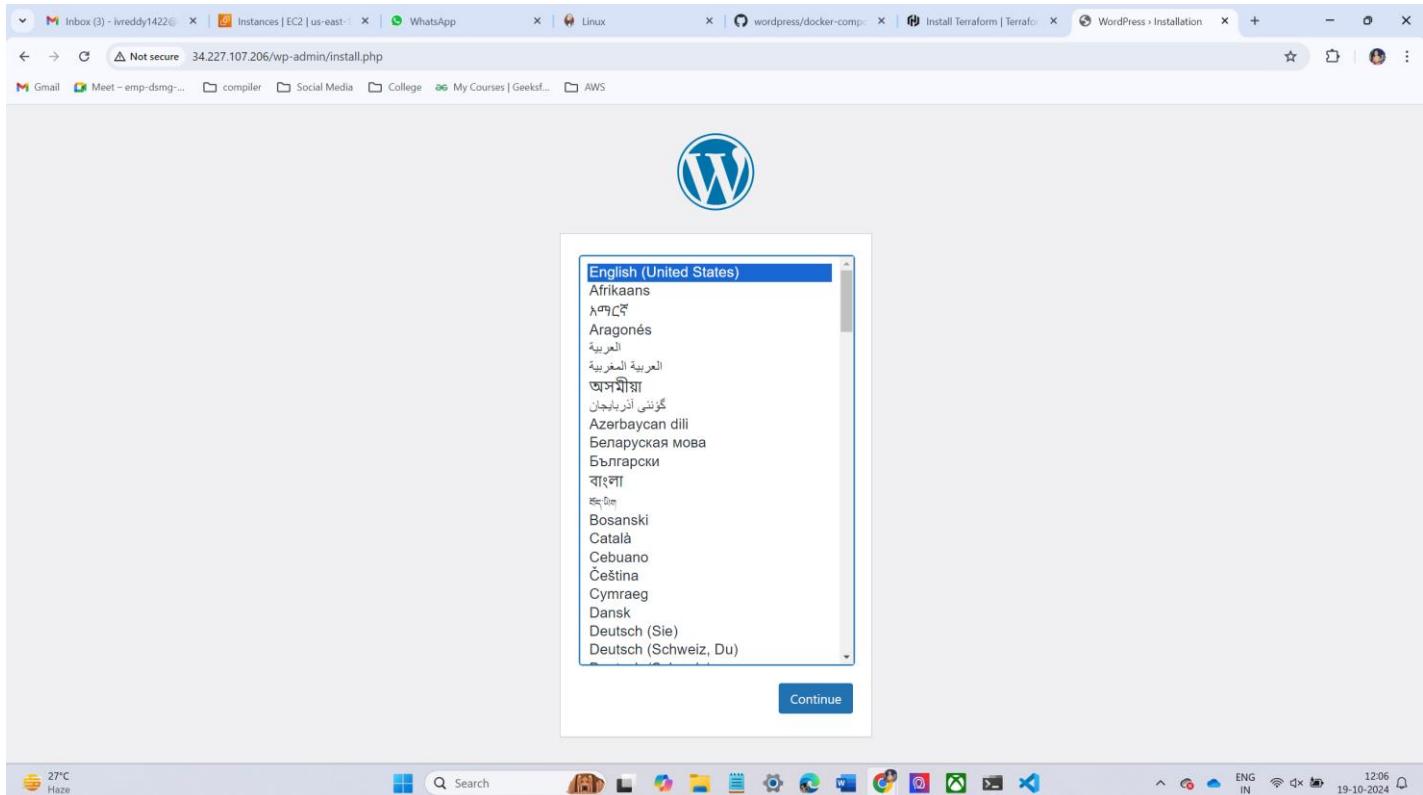
Instance ID: i-0ec10ccff1fe547b5 (public\_wordpress-1)

Public IPv4 address copied: 34.227.107.206 | [open address](#)

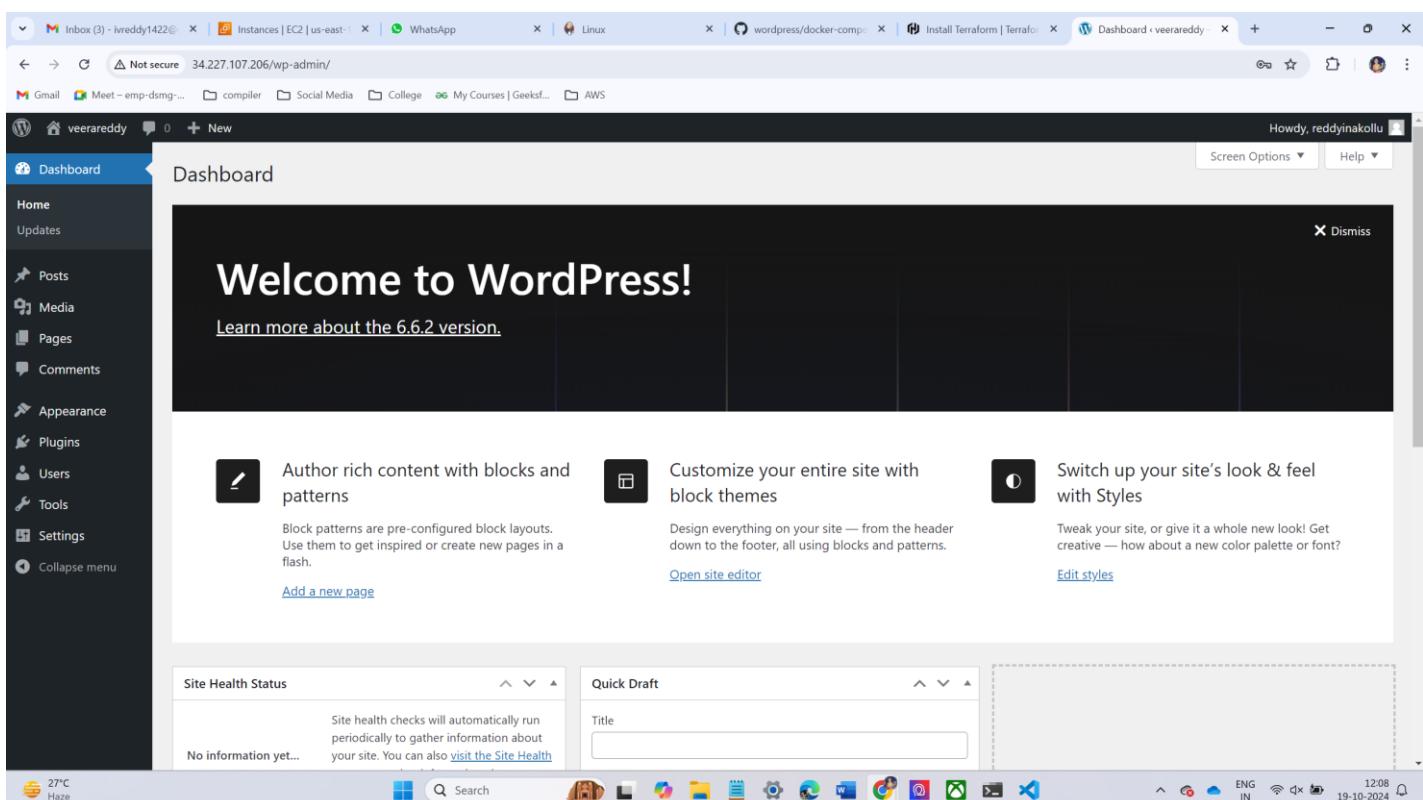
Private IPv4 addresses: 10.0.1.96

Public IPv4 DNS:

- ◆ Now browse the public ipv4 address of new ec2 instance it will displays wordpress application

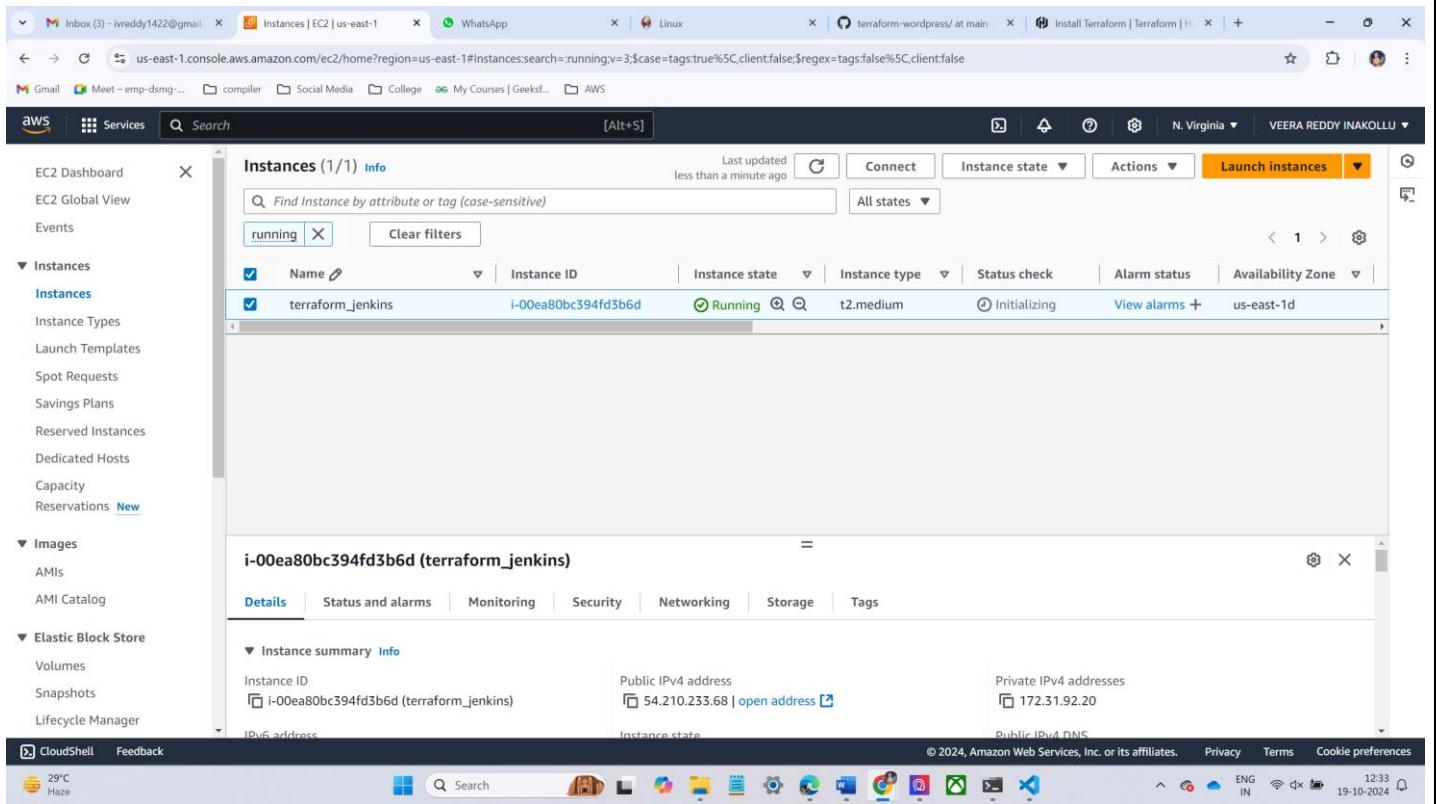


- ◆ Successfully deployed the WordPress application using terraform



# Deploy WordPress web application by using git (clone terraform script which helps to deploy WordPress web application), jenkins (in execute shell install terraform, init, fmt, validate and apply with automatic command as terraform apply --auto-approve) and terraform.

- ◆ Login to AWS account and launch EC2 instance



- ◆ Connect ec2 instance to the command prompt

```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jawah>cd Downloads

C:\Users\jawah\Downloads>ssh -i "veera.pem" ec2-user@ec2-54-210-233-68.compute-1.amazonaws.com
The authenticity of host 'ec2-54-210-233-68.compute-1.amazonaws.com (54.210.233.68)' can't be established.
ED25519 key fingerprint is SHA256:qzOoQgpGzCujPFUSSZY9hQW+z73vYHmbSN56Ks0EI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-210-233-68.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#_ _ _ _ _ Amazon Linux 2
#_ _ \_ _ _ _ AL2 End of Life is 2025-06-30.
#_ _ \_ _ #_ _ A newer version of Amazon Linux is available!
#_ _ \_ /_ / Amazon Linux 2023, GA and supported until 2028-03-15.
#_ /_ / https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-92-26 ~]$ |
```

## ◆ Now install Jenkins and setup Jenkins

```
ec2-user@ip-172-31-92-20:~ + - x
Running transaction
  Installing : jenkins-2.462.3-1.1.noarch
  Verifying  : jenkins-2.462.3-1.1.noarch
1/1
Installed:
  jenkins.noarch 0:2.462.3-1.1

Complete!
[ec2-user@ip-172-31-92-20 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-92-20 ~]$ sudo systemctl startclient_loop: send disconnect: Connection reset
C:\Users\jawah\Downloads>ssh -i "veera.pem" ec2-user@ec2-54-210-233-68.compute-1.amazonaws.com
Last login: Sat Oct 19 07:05:11 2024 from 103.160.27.56
#
# Amazon Linux 2
~~ \##### AL2 End of Life is 2025-06-30.
~~ \# AL2 End of Life is 2025-06-30.
~~ \# A newer version of Amazon Linux is available!
~~ \# / Amazon Linux 2023, GA and supported until 2028-03-15.
~~ \# https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-92-20 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-92-20 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
     Active: active (running) since Sat 2024-10-19 07:07:29 UTC; 2min 2s ago
       Main PID: 3781 (java)
      CGroup: /system.slice/jenkins.service
              └─3781 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

Oct 19 07:07:27 ip-172-31-92-20.ec2.internal jenkins[3781]: ad5b8bdedcc442928c264afe6ccf31e
Oct 19 07:07:27 ip-172-31-92-20.ec2.internal jenkins[3781]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Oct 19 07:07:27 ip-172-31-92-20.ec2.internal jenkins[3781]: ****
Oct 19 07:07:29 ip-172-31-92-20.ec2.internal jenkins[3781]: 2024-10-19 07:07:29.483+0000 [id=31]           INFO          jenkins.InitReactorRunner$1#onAttained: Completed initialization
Oct 19 07:07:29 ip-172-31-92-20.ec2.internal jenkins[3781]: 2024-10-19 07:07:29.500+0000 [id=24]           INFO          hudson.lifecycle.Lifecycle$onReady: Jenkins is full...d running
Oct 19 07:07:29 ip-172-31-92-20.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Oct 19 07:07:29 ip-172-31-92-20.ec2.internal jenkins[3781]: 2024-10-19 07:07:29.518+0000 [id=49]           INFO          h.m.DownloadService$Downloadable#load: Obtained the...Installer
Oct 19 07:07:29 ip-172-31-92-20.ec2.internal jenkins[3781]: 2024-10-19 07:07:29.518+0000 [id=49]           INFO          hudson.util.Retriger#start: Performed the action che...attempt #1
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-92-20 ~]$
```

29°C Haze 12:39 19-10-2024

## ◆ Now launch Jenkins server on new tab using public ipv4 address

Inbox (2) - ivreddy1422 | Instances | EC2 | us-east-1 | WhatsApp | Linux | terraform-wordpress/ at | Install Terraform | Terraform | Dashboard [Jenkins]

Jenkins

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Build Queue

No builds in the queue.

Build Executor Status

1 Idle  
2 Idle

Set up a distributed build

Create a job

Set up an agent

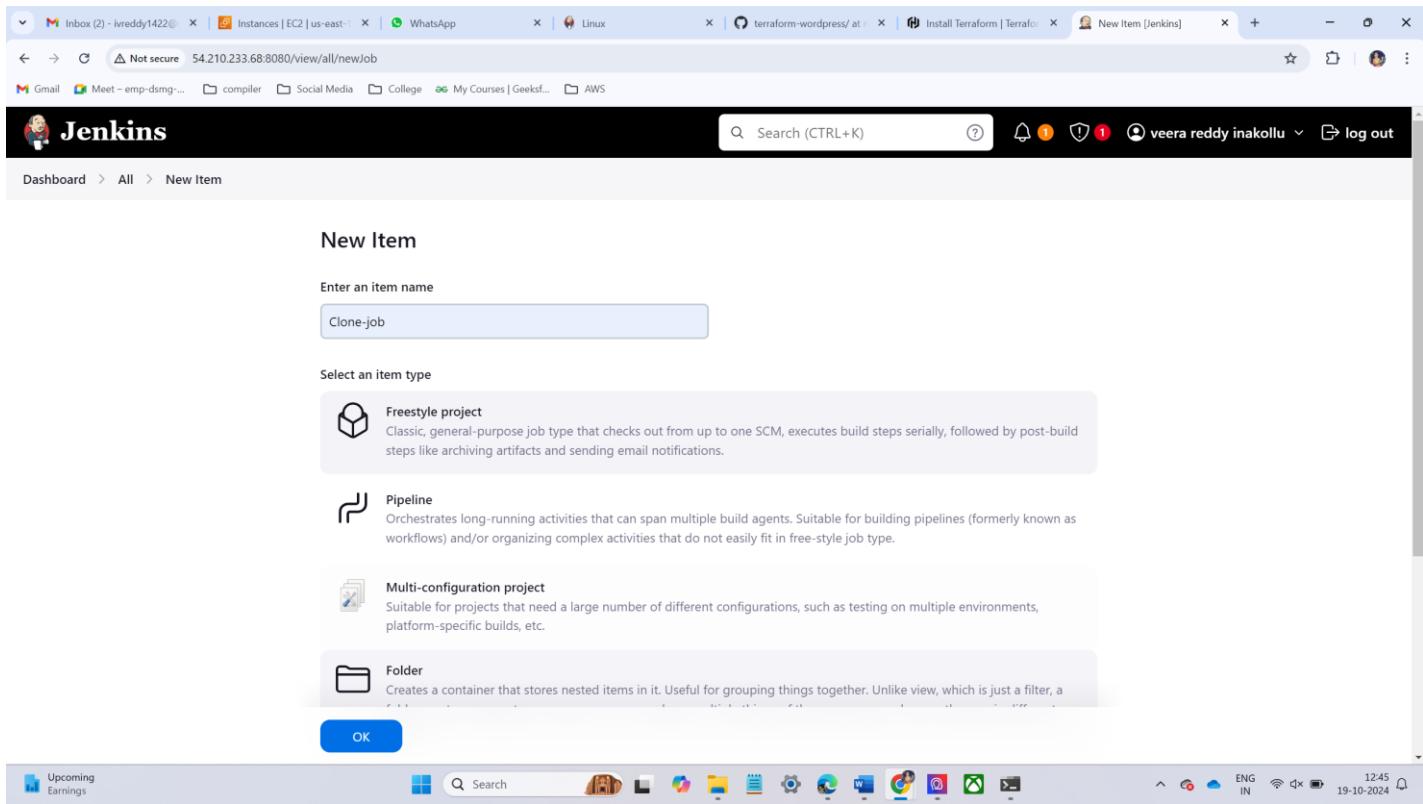
Configure a cloud

Learn more about distributed builds

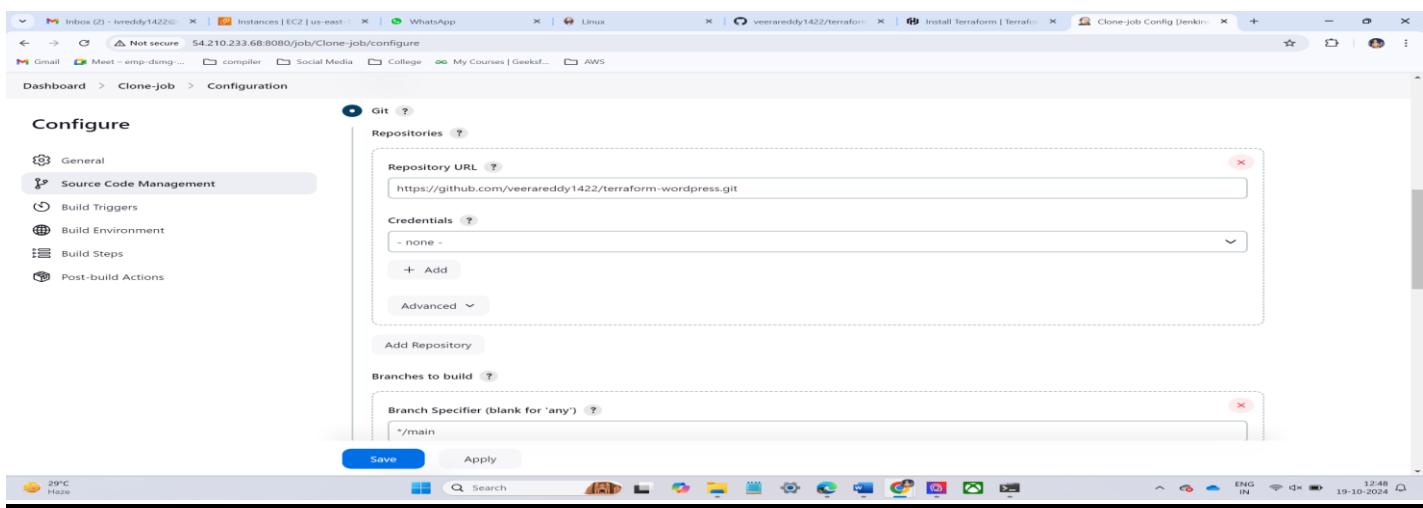
REST API Jenkins 2.462.3

29°C Haze 12:41 19-10-2024

- ◆ Create job for clone the code
- ◆ Click on "New Item".
- ◆ Enter a name for your project (e.g., WordPress-Deployment).
- ◆ Select "Freestyle project" and click OK.



- ◆ Configure Source Code Management (SCM)
- ◆ In the project configuration page, scroll down to the "**Source Code Management**" section.
- ◆ Select "**Git**".
- ◆ Enter the **Repository URL** of your GitHub repository



- ◆ For Access key and secret key purpose we can install AWS Credentials plugin

The screenshot shows the Jenkins 'Download progress' page. On the left, there's a sidebar with links like 'Updates', 'Available plugins', 'Installed plugins', 'Advanced settings', and 'Download progress'. The 'Download progress' link is highlighted. The main area is titled 'Download progress' and shows the following status:

Preparation	
• Checking internet connectivity	Success
• Checking update center connectivity	Success
• Success	Success

Below this, it lists installed plugins with green checkmarks:

- Amazon Web Services SDK :: Minimal: Success
- Amazon Web Services SDK :: EC2: Success
- AWS Credentials: Success

At the bottom, there are two buttons: 'Go back to the top page' and 'Restart Jenkins when installation is complete and no jobs are running'.

REST API Jenkins 2.462.3

- ◆ Create job for build the code
- ◆ Click on "New Item".
- ◆ Enter a name for your project (e.g., WordPress-Deployment).
- ◆ Select "Freestyle project" and click OK

The screenshot shows the Jenkins 'New Item' creation page. The user has entered 'Build-job' as the item name. Under 'Select an item type', the 'Freestyle project' option is selected, described as a 'Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.' Other options shown include 'Pipeline' and 'Multi-configuration project'. At the bottom, there is an 'OK' button.

29°C Haze 12:49 19-10-2024

- ◆ Under build environment select use secret text or files and enter access key and secret key

The screenshot shows the Jenkins configuration interface for a build job. The left sidebar has sections: General, Source Code Management, Build Triggers, **Build Environment**, Build Steps, and Post-build Actions. The 'Build Environment' section is selected. The main area is titled 'Build Environment' and contains a checkbox for 'Delete workspace before build starts' (unchecked), a checked checkbox for 'Use secret text(s) or file(s)', and a 'Bindings' section. The 'Bindings' section includes fields for 'Access Key Variable' (AWS\_ACCESS\_KEY\_ID) and 'Secret Key Variable' (AWS\_SECRET\_ACCESS\_KEY). Below these are 'Credentials' settings, with 'Specific credentials' selected and a dropdown menu showing 'AKIAUEMGED77WFCXQNLB'. At the bottom are 'Save' and 'Apply' buttons.

- ◆ Configure Build Steps
- ◆ In the project configuration page, scroll down to the "**Build Steps**" section.
- ◆ Select "**Execute shell**".
- ◆ Enter the **bash script**

The screenshot shows the Jenkins configuration interface for a build job. The left sidebar has sections: General, Source Code Management, Build Triggers, Build Environment, **Build Steps**, and Post-build Actions. The 'Build Steps' section is selected. The main area is titled 'Build Steps' and contains a section for 'Execute shell'. The 'Command' field contains the following bash script:

```
#!/bin/bash
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
terraform init
terraform fmt
terraform validate
terraform apply --auto-approve
```

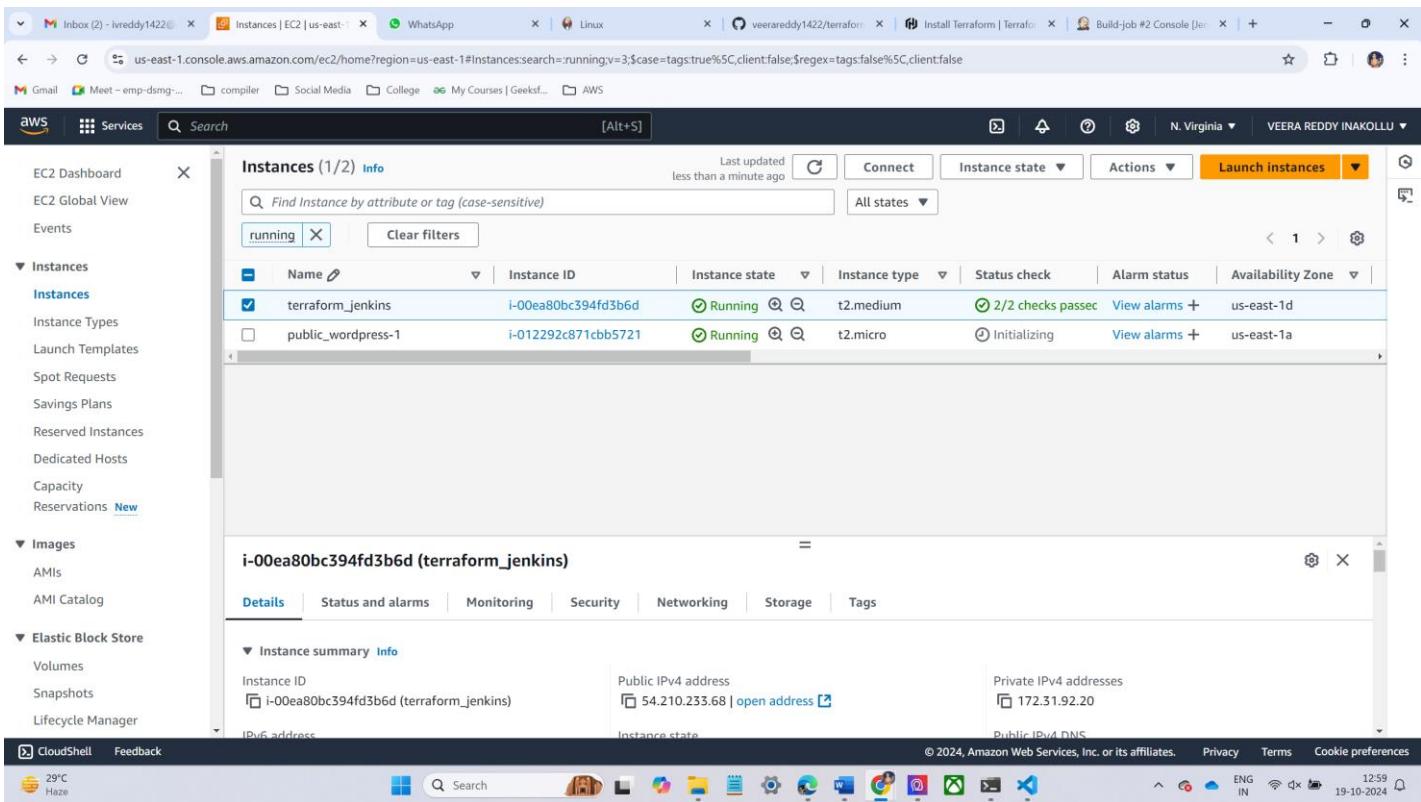
At the bottom are 'Advanced' and 'Save' buttons.

## ◆ Check the Build details of Build job

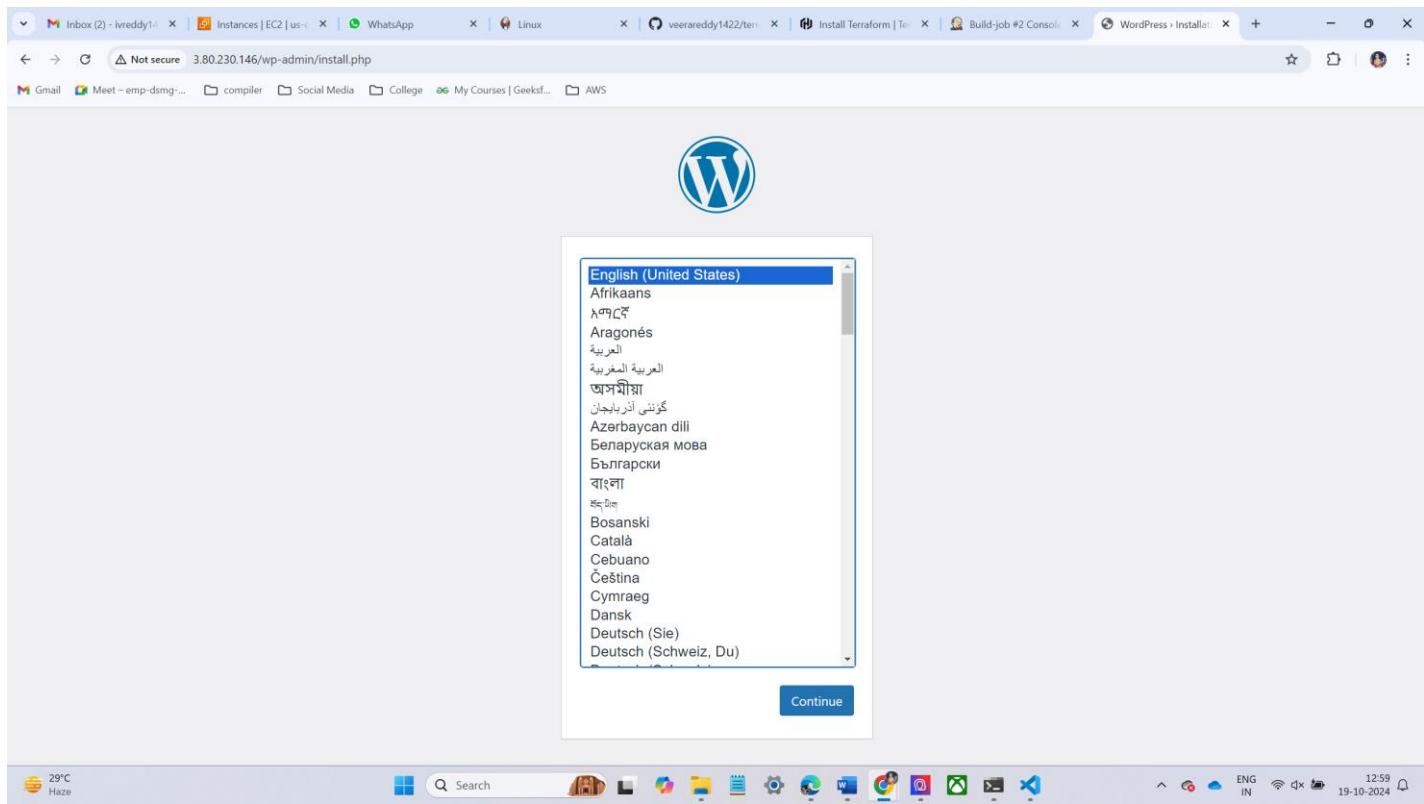
```

Started by user veera reddy inakollu
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Build-job
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Build-job/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/veerareddy1422/terraform-wordpress.git # timeout=10
Fetching upstream changes from https://github.com/veerareddy1422/terraform-wordpress.git
> git --version # timeout=10
> git --version # 'git version 2.40.1'
> git fetch --tags --force --progress -- https://github.com/veerareddy1422/terraform-wordpress.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision db405eb693a1c01ebfbcdcd6fcfa2751b11dc4d86 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f db405eb693a1c01ebfbcdcd6fcfa2751b11dc4d86 # timeout=10
Commit message: "Update subnets.tf"
> git rev-list --no-walk 3b424ab7681a9d36f6aa1c33c2003cf980fd9e9 # timeout=10
[Build-job] $ /bin/bash /tmp/jenkins11192476726506581000.sh
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package yum-utils-1.1.31-46.amzn2.0.1.noarch already installed and latest version
Package 2:shadow-utils-4.1.5.1-24.amzn2.0.3.x86_64 already installed and latest version
Nothing to do
  
```

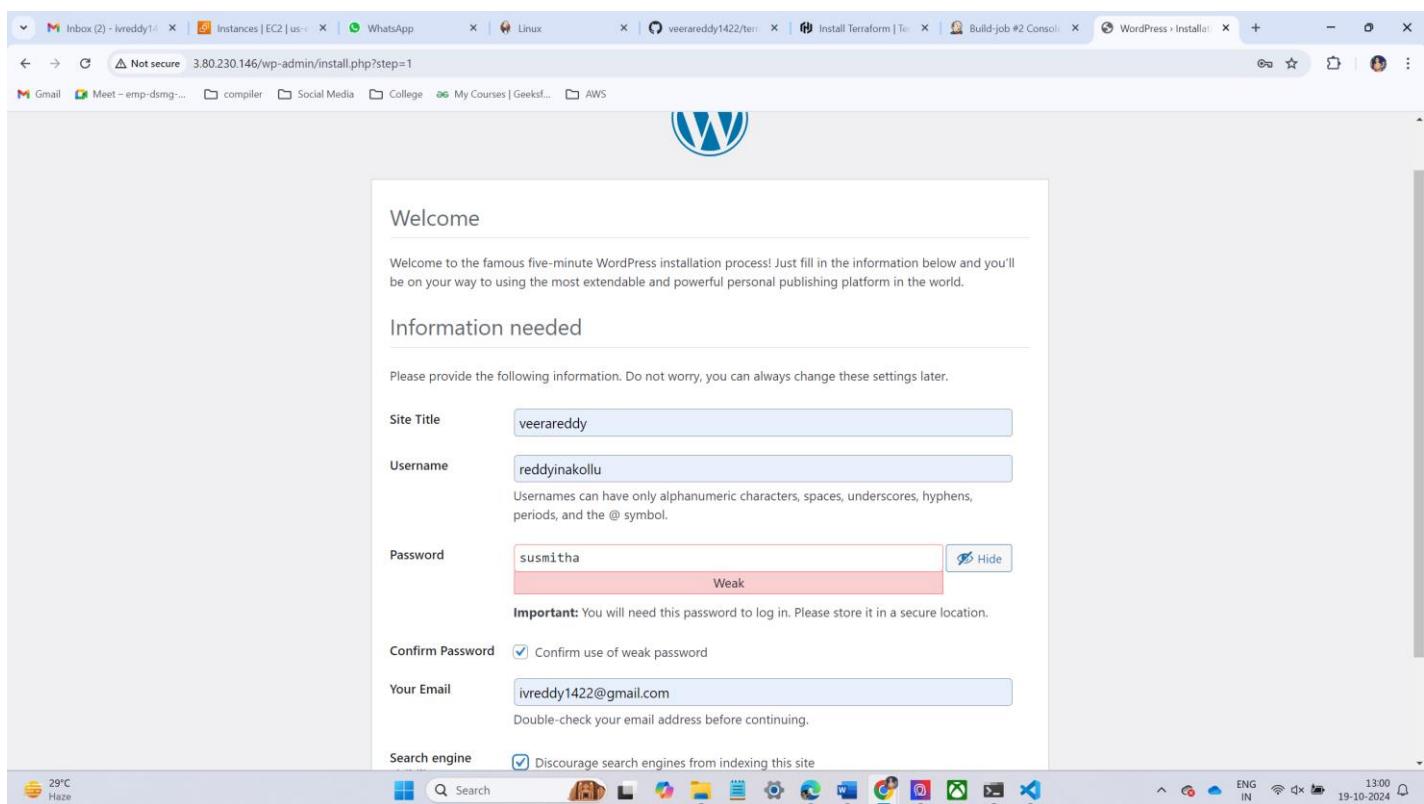
## ◆ Now login to aws account and under ec2 console one newly created ec2 instance is displayed.



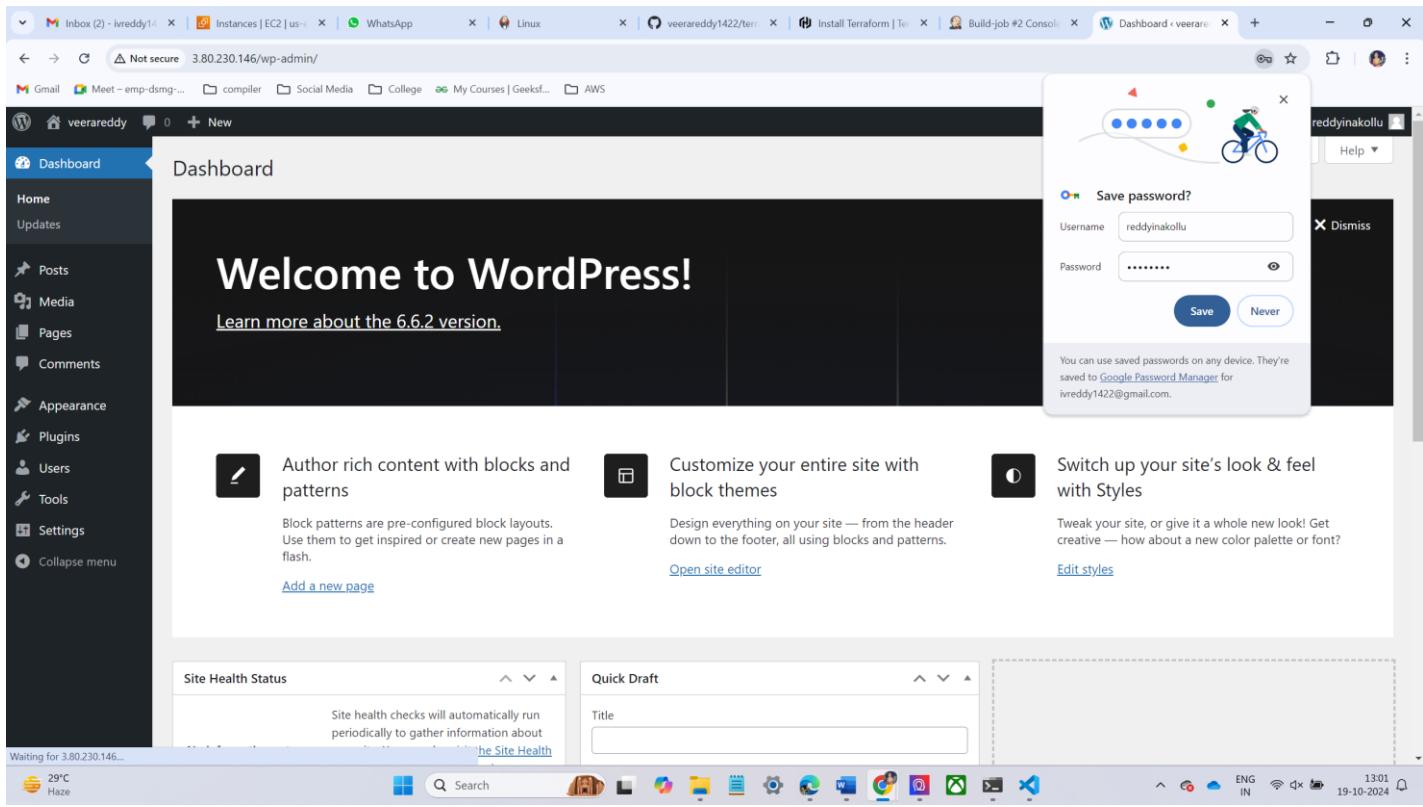
- ◆ Copy and paste the new ec2 instance ipv4 address in new tab



- ◆ Create the WordPress account



- ◆ Login WordPress account
- ◆ Successfully deployed the WordPress application using terraform and Jenkins for automation process



Deploy WordPress web application by using git (clone terraform script which helps to deploy WordPress web application), jenkins (in execute shell install terraform, init, fmt, validate and apply with automatic command as terraform apply --auto-approve) and terraform and create jenkins pipeline and add build periodically and poll scm to initial job of pipeline and check the changes happened or not which are made in github repo.

- ◆ Creating Jenkins pipeline for continues process for previous method
- ◆ For the creation of Jenkins pipeline we need to install build pipeline plugin.

The screenshot shows a browser window with multiple tabs open. The active tab is 'Manage Jenkins > Plugins'. On the left, there's a sidebar with options like 'Updates', 'Available plugins', 'Installed plugins', 'Advanced settings', and 'Download progress' (which is currently selected). The main content area is titled 'Download progress' and shows a table of download tasks. Most tasks have a green checkmark next to them, indicating success. The tasks listed are:

Task	Status
Checking internet connectivity	Success
Checking update center connectivity	Success
Success	Success
Amazon Web Services SDK :: Minimal	Success
Amazon Web Services SDK :: EC2	Success
AWS Credentials	Success
Loading plugin extensions	Success
Parameterized Trigger	Success
JavaMail API	Success
Oracle Java SE Development Kit Installer	Success
SSH server	Success
Command Agent Launcher	Success
jQuery	Success
Build Pipeline	Success
Loading plugin extensions	Success

At the bottom of the page, there's a link to 'Go back to the top page' with the note '(you can start using the installed plugins right away)'. The browser's status bar at the bottom shows system information like weather (29°C Haze), date (19-10-2024), and time (13:04).

- ◆ Goto dashboard under this section select + icon
- ◆ Now enter pipeline name and select the build pipeline view

New view

Name: wordpress\_terraform

Type:  Build Pipeline View

List View  
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

My View  
This view automatically displays all the jobs that the current user has an access to.

Create

- ◆ Now configure the pipeline
- ◆ Under pipeline flow choose your initial job

Build Pipeline View Title

Pipeline Flow

Select Initial Job:

Trigger Options

Build Cards

OK Apply

- ◆ Under display options choose No of Displayed Builds (your wish)

The screenshot shows the Jenkins configuration interface for a pipeline named "wordpress\_terraform". The "Configure" screen is displayed, specifically the "Display Options" section. The "No of Displayed Builds" dropdown is set to "3". Other settings include "Row Headers" set to "Just the pipeline number" and "Column Headers" set to "No header". The "Refresh frequency (in seconds)" is set to "3". At the bottom are "OK" and "Apply" buttons.

- ◆ Successfully creating pipeline for clone job only.

The screenshot shows the Jenkins dashboard with the "Build Pipeline" section selected. On the left, there is a sidebar titled "Pipeline" with "#1". In the main area, a green card displays the details of a build: "#1 Clone-job", "Oct 19, 2024 7:17:15 AM", "3.3 sec", and "ivreddy1422". The Jenkins version "Jenkins 2.462.3" and the date "19-10-2024" are visible at the bottom right.

- ◆ Now create the pipeline for all jobs
- ◆ For creating pipeline configure the clone job
- ◆ In clone job under post-build action select the build other projects
- ◆ Select build job

**Configure**

**Post-build Actions**

Build other projects

Projects to build: Build-job

Trigger only if build is stable (selected)

Trigger even if the build is unstable

Trigger even if the build fails

Add post-build action

Save Apply

- ◆ Configure the build job
- ◆ Under build triggers select build after other projects are built
- ◆ Select clone job

**Configure**

**Build Triggers**

Build after other projects are built

Projects to watch: Clone-job

Trigger only if build is stable (selected)

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build periodically

GitHub hook trigger for GITScm polling

Poll SCM

Save Apply

## ◆ Now apply build periodically in clone job

The screenshot shows the Jenkins 'Configure' screen for a 'Clone-job'. The 'Build Triggers' section is active, displaying several options:

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?

The 'Build periodically' option is selected, and its 'Schedule' field contains the value '\*\*\*\*\*'. A warning message below the schedule field states: '⚠ No schedules so will never run'.

Other sections visible include 'Build Environment' (with options like 'Delete workspace before build starts' and 'Use secret text(s) or file(s) ?') and 'Build Steps'.

At the bottom, there are 'Save' and 'Apply' buttons.

## ◆ Observe the build pipeline

The screenshot shows the Jenkins 'Build Pipeline' interface. It displays three parallel pipelines, each consisting of a 'Clone-job' and a 'Build-job' stage.

- Pipeline #3:** Contains a green '#3 Clone-job' stage (Oct 19, 2024 7:40:00 AM, 0.12 sec) and a yellow '#6 Build-job' stage (Oct 19, 2024 7:40:06 AM, 0.41 sec and counting).
- Pipeline #2:** Contains a green '#2 Clone-job' stage (Oct 19, 2024 7:39:00 AM, 0.15 sec) and a green '#5 Build-job' stage (Oct 19, 2024 7:39:13 AM, 20 sec).
- Pipeline #1:** Contains a green '#1 Clone-job' stage (Oct 19, 2024 7:17:15 AM, 33 sec, user ivreddy1422) and a green '#4 Build-job' stage (Oct 19, 2024 7:38:51 AM, 21 sec, user ivreddy1422).

The interface includes a toolbar with 'Run', 'History', 'Configure', 'Add Step', 'Delete', and 'Manage' buttons. The status bar at the bottom shows system information like temperature (29°C), date (19-10-2024), and time (13:10).

## ◆ Now apply the Poll SCM

The screenshot shows the Jenkins 'Configure' screen for a job named 'Clone-job'. On the left, a sidebar lists 'General', 'Source Code Management', 'Build Triggers' (which is selected and highlighted in blue), 'Build Environment', 'Build Steps', and 'Post-build Actions'. The 'Build Triggers' section contains several options: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' (which is checked). Below this is a 'Schedule' field containing '\*\*\*\*\*' and a note: 'No schedules so will only run due to SCM changes if triggered by a post-commit hook'. There is also an unchecked checkbox for 'Ignore post-commit hooks'. At the bottom of the configuration screen, there are 'Save' and 'Apply' buttons.

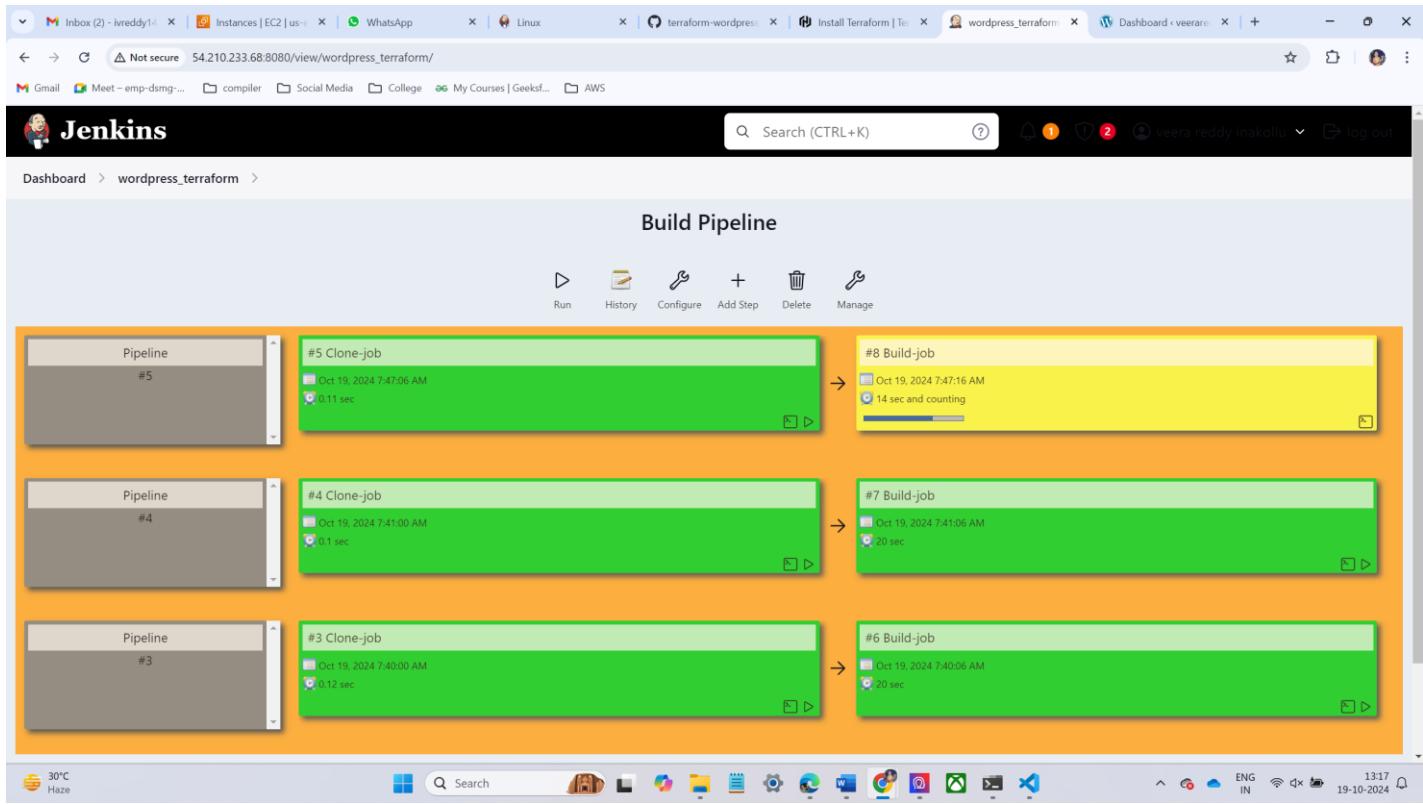
## ◆ We can change the code in git hub

The screenshot shows the GitHub code editor interface for a repository named 'terraform-wordpress'. The left sidebar shows files like 'main', 'data.sh', 'ec2.tf', 'igw.tf', 'routetable.tf', 'security.tf' (which is selected and highlighted in blue), 'subnets.tf', and 'vpc.tf'. The main editor area displays the 'security.tf' file content:

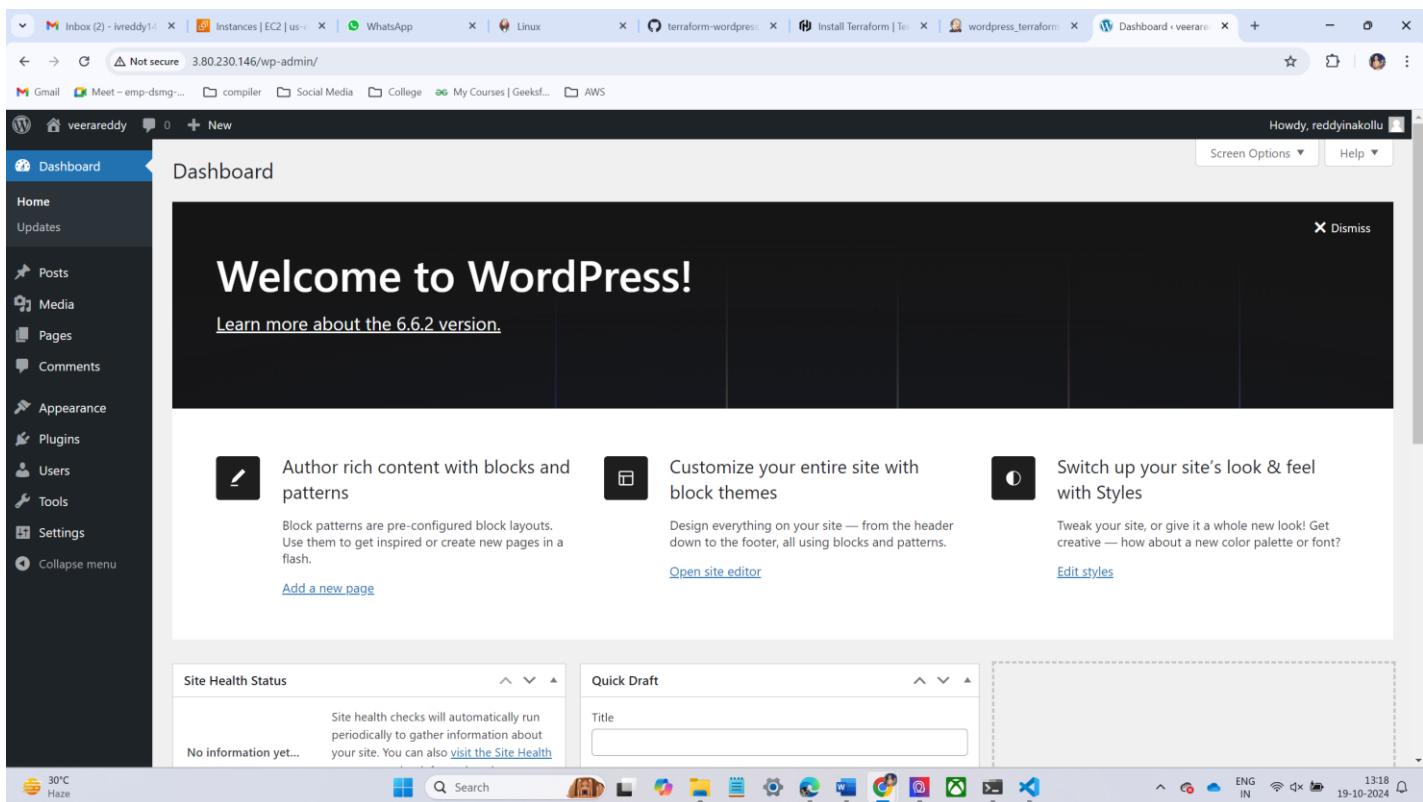
```
1 #creating security group
2 resource "aws_security_group" "security1" {
3   vpc_id = aws_vpc.demoVPC.id
4
5   #inbound Rules
6   ingress {
7     from_port = "80"
8     to_port = "80"
9     protocol = "tcp"
10    cidr_blocks = ["0.0.0.0/0"]
11  }
12  ingress {
13    from_port = 22
14    to_port = 22
15    protocol = "tcp"
16    cidr_blocks = ["0.0.0.0/0"]
17  }
18  egress {
19    from_port = 0
20    to_port = 0
21    protocol = "-1"
22    cidr_blocks = ["0.0.0.0/0"]
23 }
```

At the top right of the editor, there are 'Cancel changes' and 'Commit changes...' buttons. Below the editor, a note says: 'Use `Control + Shift + m` to toggle the `tab` key moving focus. Alternatively, use `esc`, then `tab`, to move to the next interactive element on the page.' The bottom of the screen shows the Windows taskbar with various pinned icons.

- ◆ Whenever I am changing the code in git hub its automatically build the pipeline



- ◆ Successfully host the WordPress application using Jenkins pipeline using build periodically and poll scm.



# Deploy WordPress web application by using k8's (Declarative manifest method) with the help of docker hub images?

- Login to AWS account and launch ec2 instance

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, and Reservations. The main area displays a table titled 'Instances (1/1) Info'. It shows one instance: 'Name' is 'kubernetes', 'Instance ID' is 'i-060bf3be3cb9992ca', 'Instance state' is 'Running', 'Instance type' is 't2.medium', 'Status check' is 'Initializing', and it's in 'us-east-1d' Availability Zone. Below the table, a modal window for 'i-060bf3be3cb9992ca (kubernetes)' is open, showing the 'Details' tab with information about the instance's summary, including its Public IPv4 address (52.87.158.240) and Private IPv4 addresses (172.31.83.29).

- Now connect ec2 instance to the command prompt

The screenshot shows a Windows terminal window titled 'ec2-user@ip-172-31-83-29 ~'. The user has run the command 'ssh -i "veera.pem" ec2-user@ec2-52-87-158-240.compute-1.amazonaws.com' to connect to the instance. The terminal shows the Microsoft Windows version (10.0.22631.4317) and the user's path ('C:\Users\jawah'). The user then runs 'cd Downloads' and lists files. A warning message from Amazon Linux 2 is displayed, indicating that the end of life is June 30, 2025, and that a newer version of Amazon Linux is available. The user is connected via port 22.

- Now install kubectl by using google chrome

```
ec2-user@ip-172-31-83-29:~ + v - x

top          Display resource (CPU/memory) usage
cordond     Mark node as unschedulable
uncordond   Mark node as schedulable
drain       Drain node in preparation for maintenance
taint       Update the taints on one or more nodes

Troubleshooting and Debugging Commands:
describe    Show details of a specific resource or group of resources
logs        Print the logs for a container in a pod
attach      Attach to a running container
exec        Execute a command in a container
port-forward Forward one or more local ports to a pod
proxy       Run a proxy to the Kubernetes API server
cp          Copy files and directories to and from containers
auth        Inspect authorization
debug       Create debugging sessions for troubleshooting workloads and nodes
events      List events

Advanced Commands:
diff        Diff the live version against a would-be applied version
apply       Apply a configuration to a resource by file name or stdin
patch       Update fields of a resource
replace     Replace a resource by file name or stdin
wait        Experimental: Wait for a specific condition on one or many resources
kustomize   Build a kustomization target from a directory or URL

Settings Commands:
label       Update the labels on a resource
annotate    Update the annotations on a resource
completion  Output shell completion code for the specified shell (bash, zsh, fish, or powershell)

Subcommands provided by plugins:

Other Commands:
api-resources Print the supported API resources on the server
api-versions  Print the supported API versions on the server, in the form of "group/version"
config       Modify kubeconfig files
plugin       Provides utilities for interacting with plugins
version      Print the client and server version information

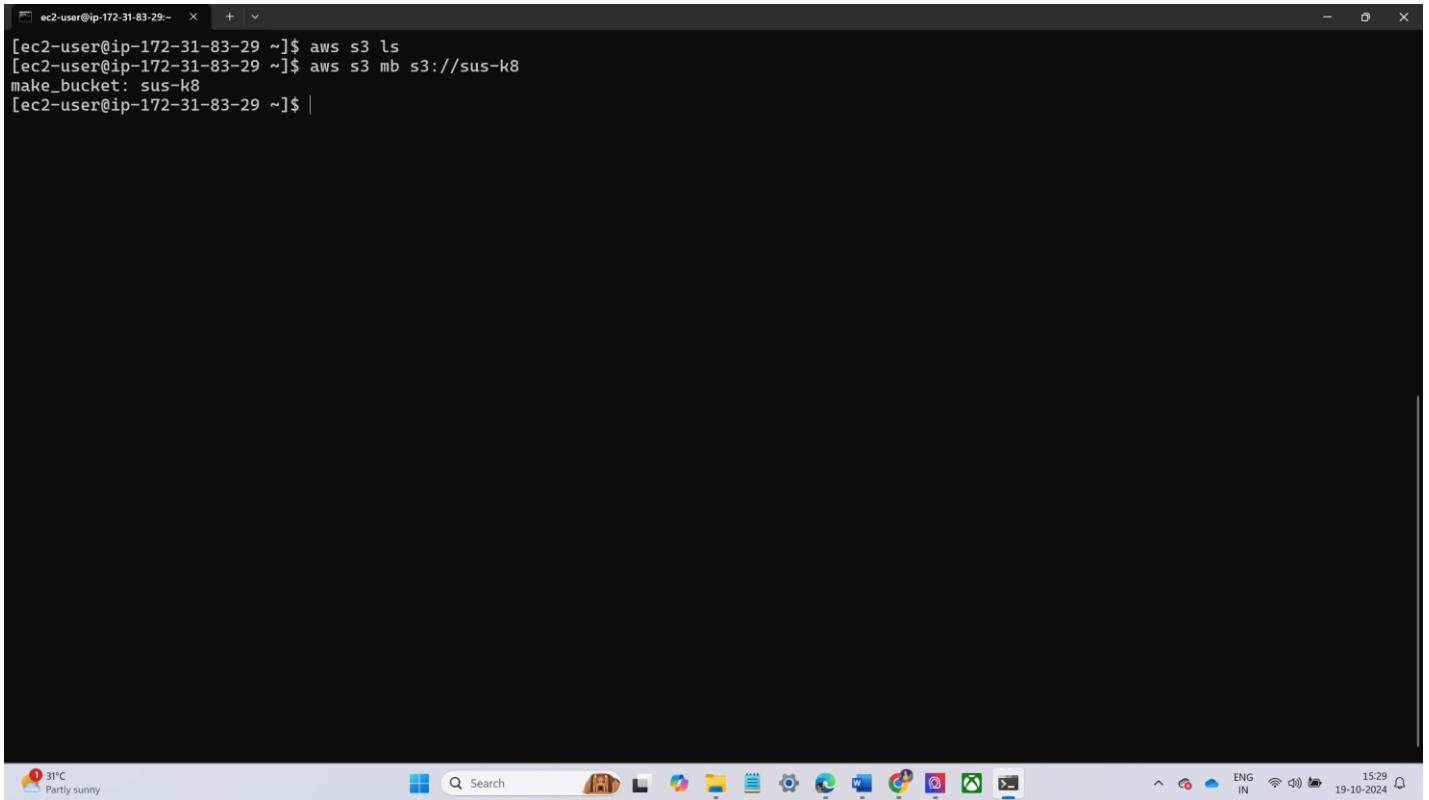
Usage:
  kubectl [flags] [options]

Use "kubectl <command> --help" for more information about a given command.
Use "kubectl options" for a list of global command-line options (applies to all commands).
[ec2-user@ip-172-31-83-29 ~]$ |
```

- After installing kubectl than install kops

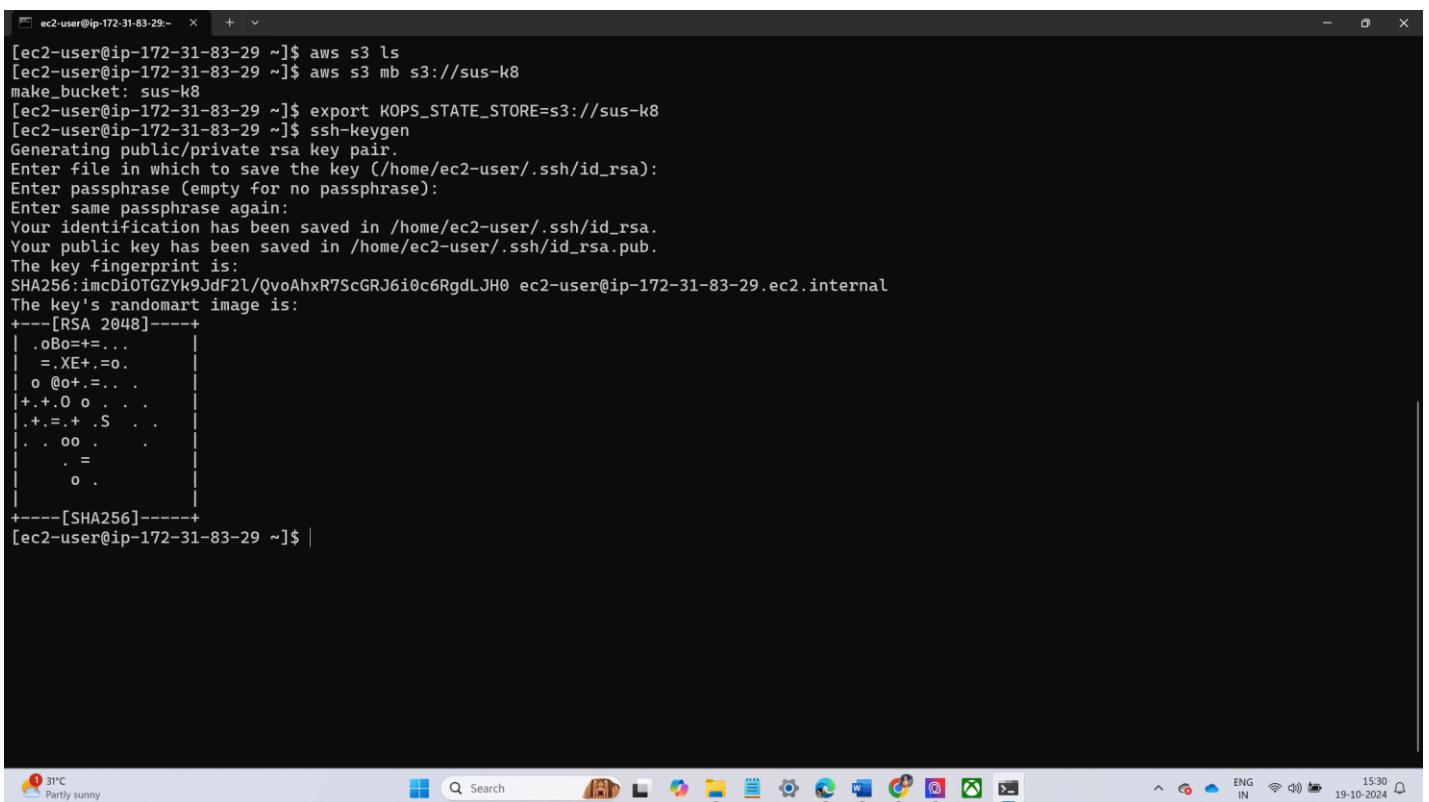
```
ec2-user@ip-172-31-83-29 ~ % name | cut -d "" -f 4)/kops-linux-amd64  
mv kops /usr/local/bin/kops  
          % Total    % Received % Xferd  Average Speed   Time   Time     Time  Current  
          Dload  Upload   Total Spent   Left Speed  
0 0 0 0 0 0 0 0 --:--:--:--:--:--:--:--:-- 0  
100 238M 100 238M 0 0 89.6M 0 0:00:02 0:00:02 --:-- 91.2M  
[ec2-user@ip-172-31-83-29 ~]$ chmod +x kops  
[ec2-user@ip-172-31-83-29 ~]$ sudo mv kops /usr/local/bin/kops  
[ec2-user@ip-172-31-83-29 ~]$ kops  
kOps is Kubernetes Operations.  
  
kOps is the easiest way to get a production grade Kubernetes cluster up and running. We like to think of it as kubectl for clusters.  
  
kOps helps you create, destroy, upgrade and maintain production-grade, highly available, Kubernetes clusters from the command line. AWS (Amazon Web Services) is currently officially supported, with Digital Ocean and OpenStack in beta support.  
  
Usage:  
  kops [command]  
  
Available Commands:  
  completion  Generate the autocompletion script for the specified shell  
  create      Create a resource by command line, filename or stdin.  
  delete      Delete clusters, instancegroups, instances, and secrets.  
  distrust    Distrust keypairs.  
  edit        Edit clusters and other resources.  
  export      Export configuration.  
  get         Get one or many resources.  
  help        Help about any command  
  promote    Promote a resource.  
  replace    Replace cluster resources.  
  rolling-update Rolling update a cluster.  
  toolbox    Miscellaneous, experimental, or infrequently used commands.  
  trust      Trust keypairs.  
  update     Update a cluster.  
  upgrade    Upgrade a kubernetes cluster.  
  validate   Validate a kOps cluster.  
  version    Print the kOps version information.  
  
Flags:  
  --config string  yaml config file (default is $HOME/.kops.yaml)  
  -h, --help        help for kops  
  --name string    Name of cluster. Overrides KOPS_CLUSTER_NAME environment variable  
  --state string   Location of state storage (kops 'config' file). Overrides KOPS_STATE_STORE environment variable  
  -v, --v Level    number for the log level verbosity  
  
Use "kops [command] --help" for more information about a command.  
[ec2-user@ip-172-31-83-29 ~]$ |
```

- Now check AWS configuration and create a s3 bucket by using command aws s3 mb s3://bucketname



```
[ec2-user@ip-172-31-83-29 ~]$ aws s3 ls
[ec2-user@ip-172-31-83-29 ~]$ aws s3 mb s3://sus-k8
make_bucket: sus-k8
[ec2-user@ip-172-31-83-29 ~]$ |
```

- Now export the bucket to the kops and give ssh keygen permissions



```
[ec2-user@ip-172-31-83-29 ~]$ aws s3 ls
[ec2-user@ip-172-31-83-29 ~]$ aws s3 mb s3://sus-k8
make_bucket: sus-k8
[ec2-user@ip-172-31-83-29 ~]$ export KOPS_STATE_STORE=s3://sus-k8
[ec2-user@ip-172-31-83-29 ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ec2-user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/id_rsa.
Your public key has been saved in /home/ec2-user/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:imcDiOTGZYk9JdF2L/QvoAhxR7ScGRJ6i0c6RgdLJH0 ec2-user@ip-172-31-83-29.ec2.internal
The key's randomart image is:
+---[RSA 2048]---+
| .oBo+=... |
| =.XE+.=o. |
| o @o+.=... |
| +.+0 o . . . |
| .+.=+ .S . . |
| .. oo . . . |
| . = . . . . |
| o . . . . . |
+---[SHA256]---+
[ec2-user@ip-172-31-83-29 ~]$ |
```

- Now create cluster by using command `kubectl create cluster --name <cluster-name> --state <bucket-name> --regions <name of regions you created> --nodes <number of nodes> --yes`

```
ec2-user@ip-172-31-83-29:~ + ~
I1019 10:06:20.081658 3577 update_cluster.go:338] Exporting kubeconfig for cluster
kOps has set your kubectl context to veerareddy.k8s.local

Cluster is starting. It should be ready in a few minutes.

Suggestions:
* validate cluster: kops validate cluster --wait 10m
* list nodes: kubectl get nodes --show-labels
* ssh to a control-plane node: ssh -i ~/.ssh/id_rsa ubuntu@
* the ubuntu user is specific to Ubuntu. If not using Ubuntu please use the appropriate user based on your OS.
* read about installing addons at: https://kops.sigs.k8s.io/addons.

[ec2-user@ip-172-31-83-29 ~]$ kops validate cluster
Using cluster from kubectl context: veerareddy.k8s.local

Validating cluster veerareddy.k8s.local

INSTANCE GROUPS
NAME          ROLE      MACHINETYPE   MIN   MAX   SUBNETS
control-plane-us-east-1a  ControlPlane  t3.medium    1     1     us-east-1a
nodes-us-east-1a       Node        t3.medium    1     1     us-east-1a
nodes-us-east-1b       Node        t3.medium    1     1     us-east-1b

NODE STATUS
NAME          ROLE      READY
i-09b1ff63aec3a5737  control-plane  True

VALIDATION ERRORS
KIND  NAME          MESSAGE
Machine i-072d0901c719c37df  machine "i-072d0901c719c37df" has not yet joined cluster
Machine i-0a864d430b4594c2d  machine "i-0a864d430b4594c2d" has not yet joined cluster
Pod    kube-system/coredns-7bfbb444bf-z6jpn  system-cluster-critical pod "coredns-7bfbb444bf-z6jpn" is pending
Pod    kube-system/coredns-autoscaler-84969b654b-g87lt  system-cluster-critical pod "coredns-autoscaler-84969b654b-g87lt" is pending

Validation Failed
Error: validation failed: cluster not yet healthy
[ec2-user@ip-172-31-83-29 ~]$ |
```

Watchlist Ideas

Search

15:42 19-10-2024 ENG IN

- Now check the created cluster list

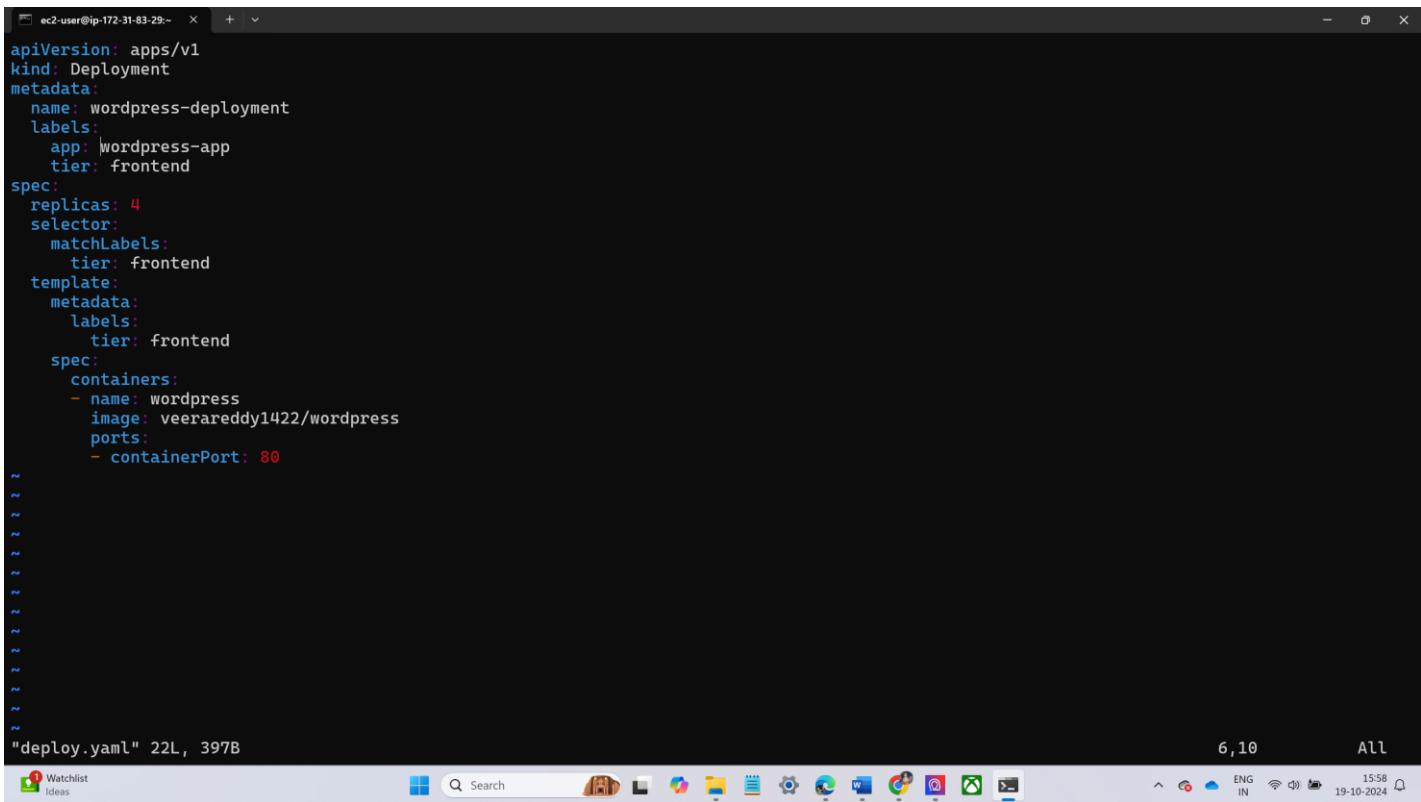
```
ec2-user@ip-172-31-83-29:~ + ~
[ec2-user@ip-172-31-83-29 ~]$ kubectl get all
NAME           TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes  ClusterIP  100.64.0.1  <none>        443/TCP  5m8s
[ec2-user@ip-172-31-83-29 ~]$ |
```

Rain coming In about 2.5 hours

Search

15:45 19-10-2024 ENG IN

- Now create the file for deployment and write script (HELM)



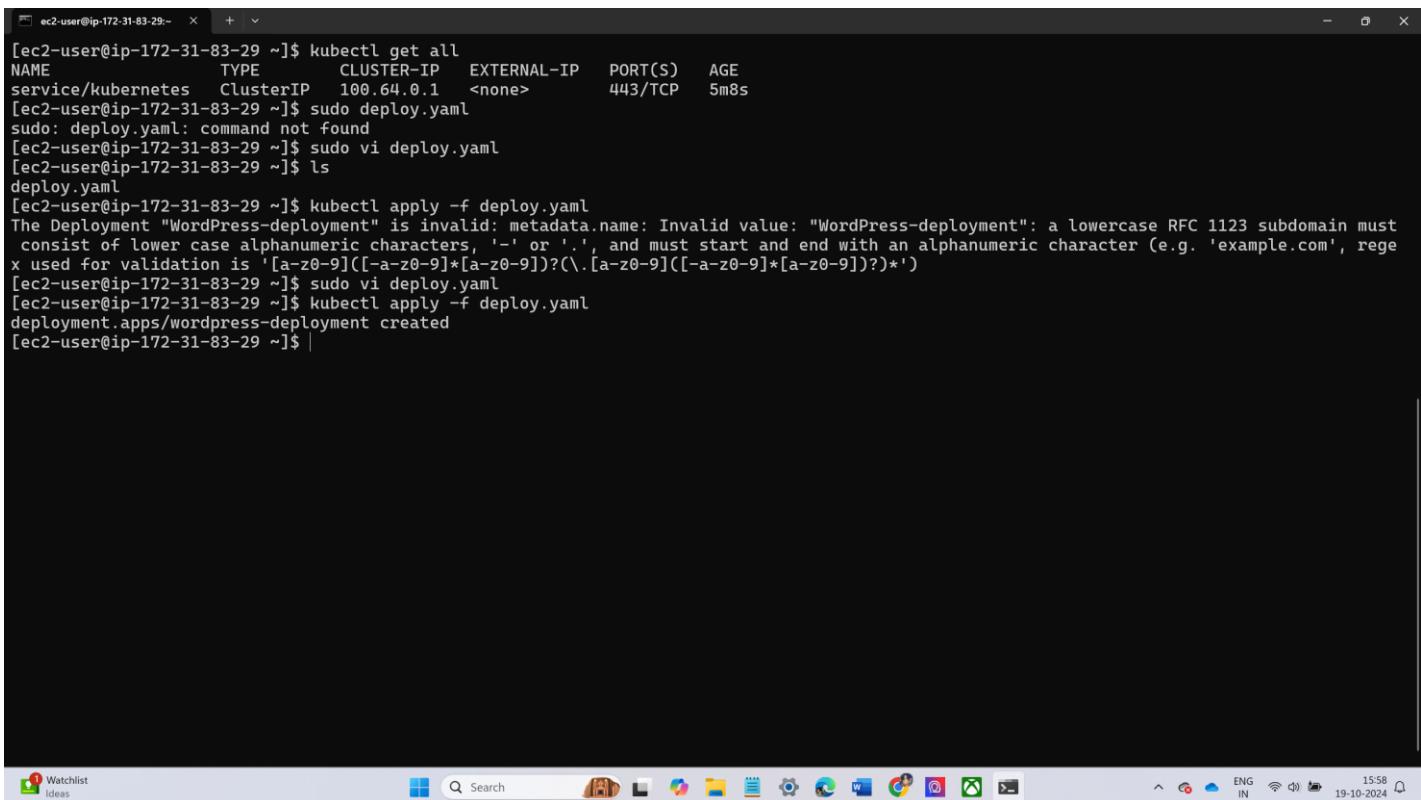
```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress-deployment
  labels:
    app: wordpress-app
    tier: frontend
spec:
  replicas: 4
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
        - name: wordpress
          image: veerareddy1422/wordpress
          ports:
            - containerPort: 80

```

"deploy.yaml" 22L, 397B

- Now create deployment pods by using command `kubectl apply -f <yaml file name>`



```

[ec2-user@ip-172-31-83-29 ~]$ kubectl get all
NAME           TYPE     CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP  100.64.0.1   <none>       443/TCP   5m8s
[ec2-user@ip-172-31-83-29 ~]$ sudo deploy.yaml
sudo: deploy.yaml: command not found
[ec2-user@ip-172-31-83-29 ~]$ sudo vi deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ ls
deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl apply -f deploy.yaml
The Deployment "WordPress-deployment" is invalid: metadata.name: Invalid value: "WordPress-deployment": a lowercase RFC 1123 subdomain must consist of lower case alphanumeric characters, '-' or '.', and must start and end with an alphanumeric character (e.g. 'example.com', regex used for validation is '[a-z0-9]([-a-z0-9]*[a-z0-9])?(.[a-z0-9]([-a-z0-9]*[a-z0-9])?)*')
[ec2-user@ip-172-31-83-29 ~]$ sudo vi deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl apply -f deploy.yaml
deployment.apps/wordpress-deployment created
[ec2-user@ip-172-31-83-29 ~]$ 

```

- now check the pods list

```
[ec2-user@ip-172-31-83-29 ~]$ kubectl get all
NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes  ClusterIP  100.64.0.1  <none>       443/TCP   5m8s

[ec2-user@ip-172-31-83-29 ~]$ sudo deploy.yaml
sudo: deploy.yaml: command not found
[ec2-user@ip-172-31-83-29 ~]$ sudo vi deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ ls
deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl apply -f deploy.yaml
The Deployment "WordPress-deployment" is invalid: metadata.name: Invalid value: "WordPress-deployment": a lowercase RFC 1123 subdomain must consist of lower case alphanumeric characters, '-' or '.', and must start and end with an alphanumeric character (e.g. 'example.com', regex used for validation is '[a-z0-9]([-a-z0-9]*[a-z0-9])?(.[a-z0-9]([-a-z0-9]*[a-z0-9])?)?')
[ec2-user@ip-172-31-83-29 ~]$ sudo vi deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl apply -f deploy.yaml
deployment.apps/wordpress-deployment created
[ec2-user@ip-172-31-83-29 ~]$ sudo vi deploy.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl get all
NAME                           READY   STATUS    RESTARTS   AGE
pod/wordpress-deployment-59ccc98fb9-4zrw7  1/1     Running   0          48s
pod/wordpress-deployment-59ccc98fb9-86qk7  1/1     Running   0          48s
pod/wordpress-deployment-59ccc98fb9-dsbxv  1/1     Running   0          48s
pod/wordpress-deployment-59ccc98fb9-xl9bx  1/1     Running   0          48s

NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes  ClusterIP  100.64.0.1  <none>       443/TCP   18m

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/wordpress-deployment  4/4      4           4           48s

NAME                           DESIRED   CURRENT   READY   AGE
replicaset.apps/wordpress-deployment-59ccc98fb9  4         4         4         48s

[ec2-user@ip-172-31-83-29 ~]$ |
```

- now create service file and write script

- Now create service pods by using kubectl apply -f <service.yaml>

```
ec2-user@ip-172-31-83-29: ~ + 
pod/wordpress-deployment-59ccc98fb9-86qk7 1/1 Running 0 48s
pod/wordpress-deployment-59ccc98fb9-dsbxv 1/1 Running 0 48s
pod/wordpress-deployment-59ccc98fb9-xl9bx 1/1 Running 0 48s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 100.64.0.1 <none> 443/TCP 18m

NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/wordpress-deployment 4/4 4 4 48s

NAME DESIRED CURRENT READY AGE
replicaset.apps/wordpress-deployment-59ccc98fb9 4 4 4 48s
[ec2-user@ip-172-31-83-29 ~]$ sudo vi service.yaml
[ec2-user@ip-172-31-83-29 ~]$ ls
deploy.yaml service.yaml
[ec2-user@ip-172-31-83-29 ~]$ kubectl apply -f service.yaml
service/wordpress-service created
[ec2-user@ip-172-31-83-29 ~]$ kubectl get all
NAME READY STATUS RESTARTS AGE
pod/wordpress-deployment-59ccc98fb9-4zrw7 1/1 Running 0 4m48s
pod/wordpress-deployment-59ccc98fb9-86qk7 1/1 Running 0 4m48s
pod/wordpress-deployment-59ccc98fb9-dsbxv 1/1 Running 0 4m48s
pod/wordpress-deployment-59ccc98fb9-xl9bx 1/1 Running 0 4m48s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 100.64.0.1 <none> 443/TCP 22m
service/wordpress-service LoadBalancer 100.66.32.163 a505f870777eb490baf387e64100fc1f-764557402.us-east-1.elb.amazonaws.com 80:3142 /TCP 16s

NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/wordpress-deployment 4/4 4 4 4m48s

NAME DESIRED CURRENT READY AGE
replicaset.apps/wordpress-deployment-59ccc98fb9 4 4 4 4m48s
[ec2-user@ip-172-31-83-29 ~]$ |
```

30°C Mostly sunny 16:03 ENG IN 19-10-2024

- After creation of all pods. Browse the load balancer URL in the browser it will displays WordPress application

