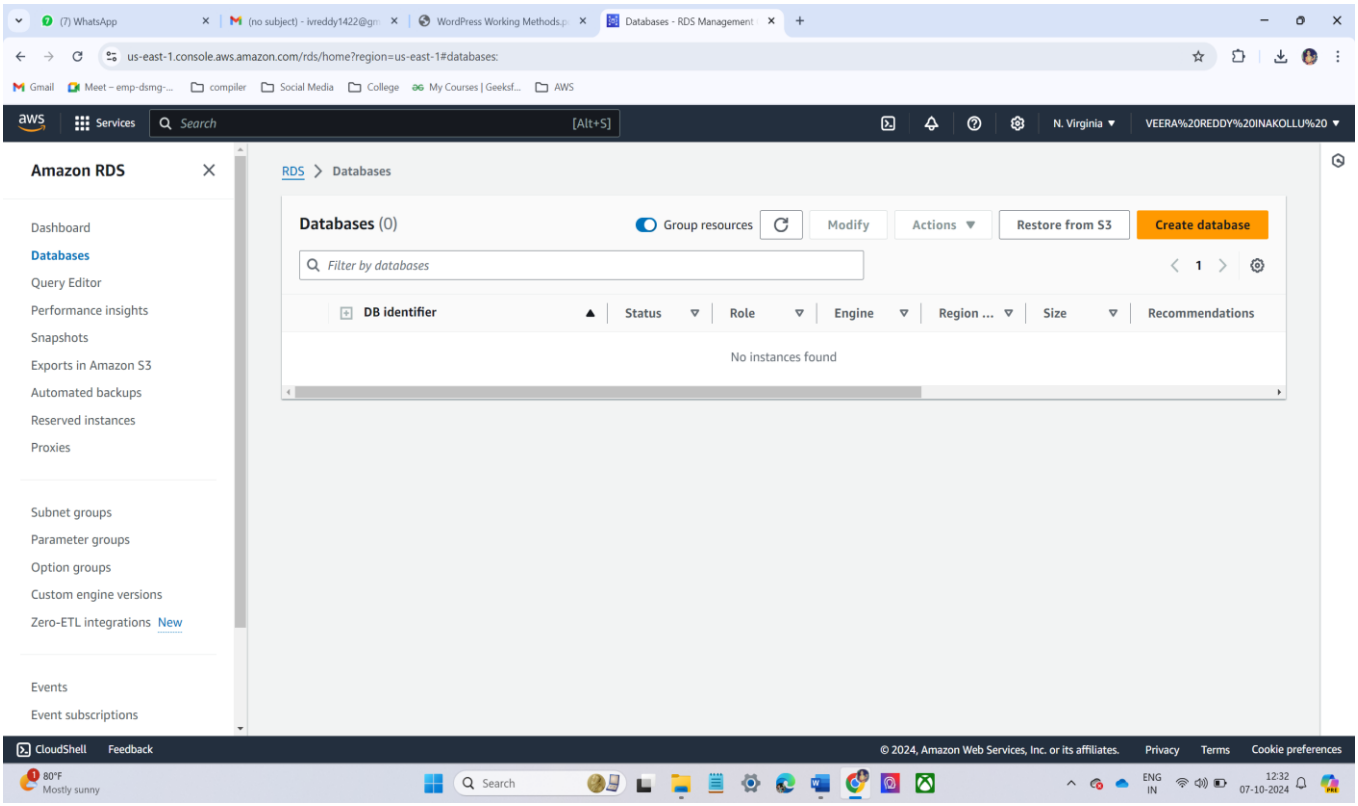
WordPress Working Methods

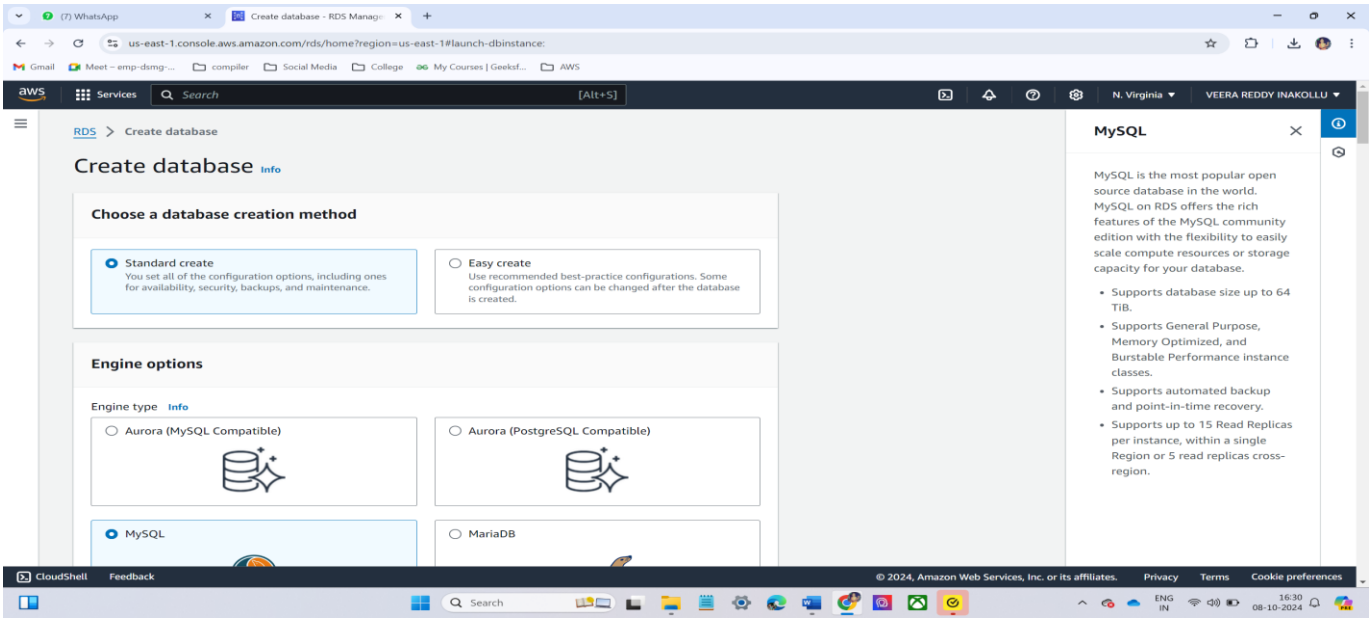
--Veera Reddy Inakollu

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

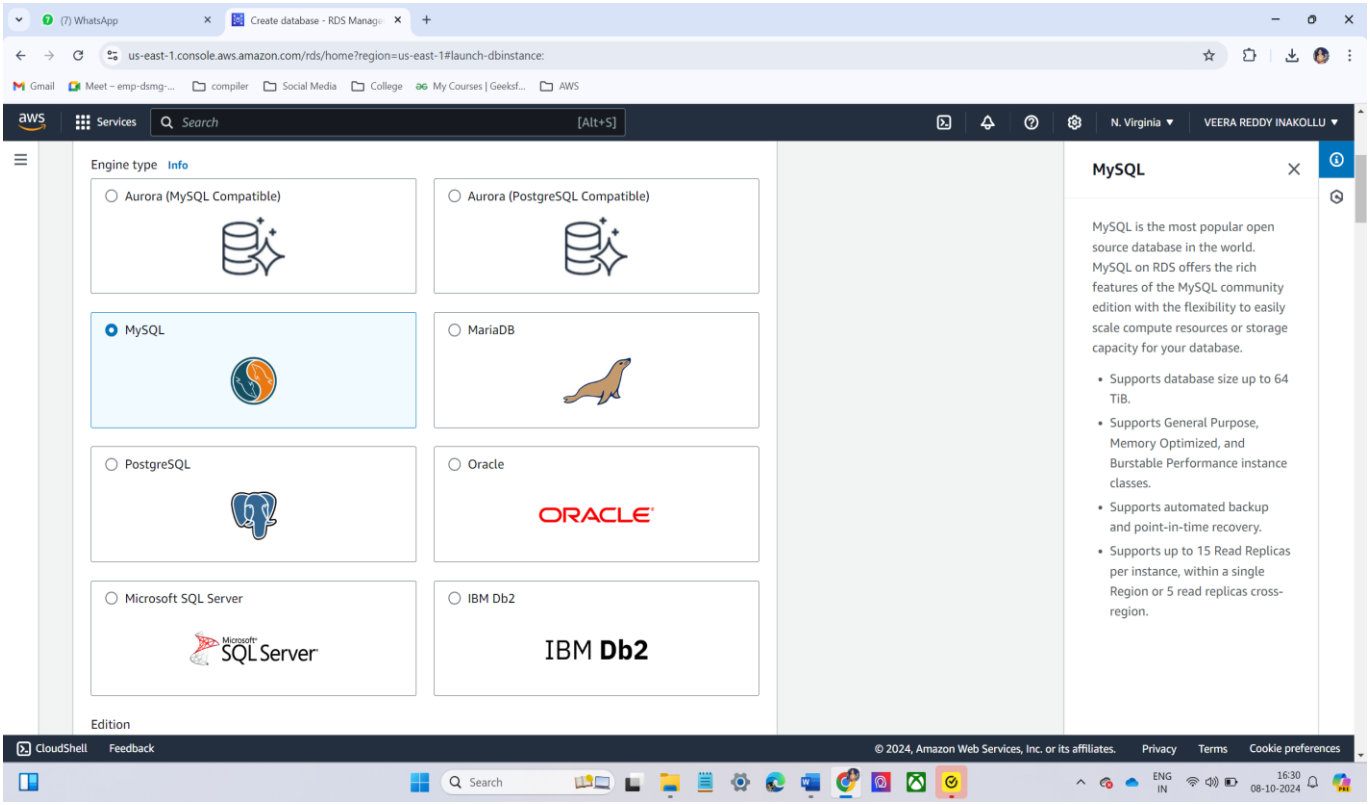
- Now first go to aws account and log into aws and search RDS and click on create database.



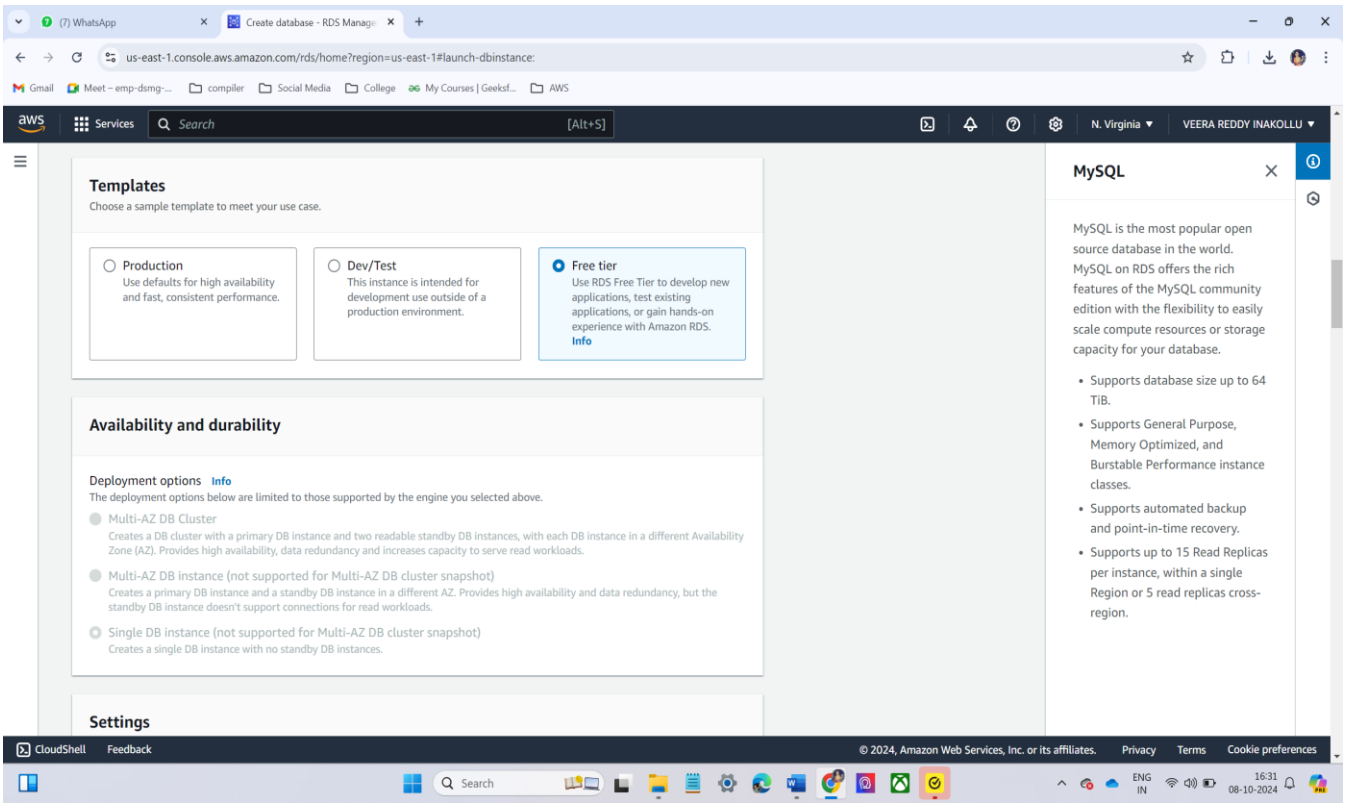
- Now select the database creation method (select standard method) because by selecting easy creation methods then its disables free tier template.



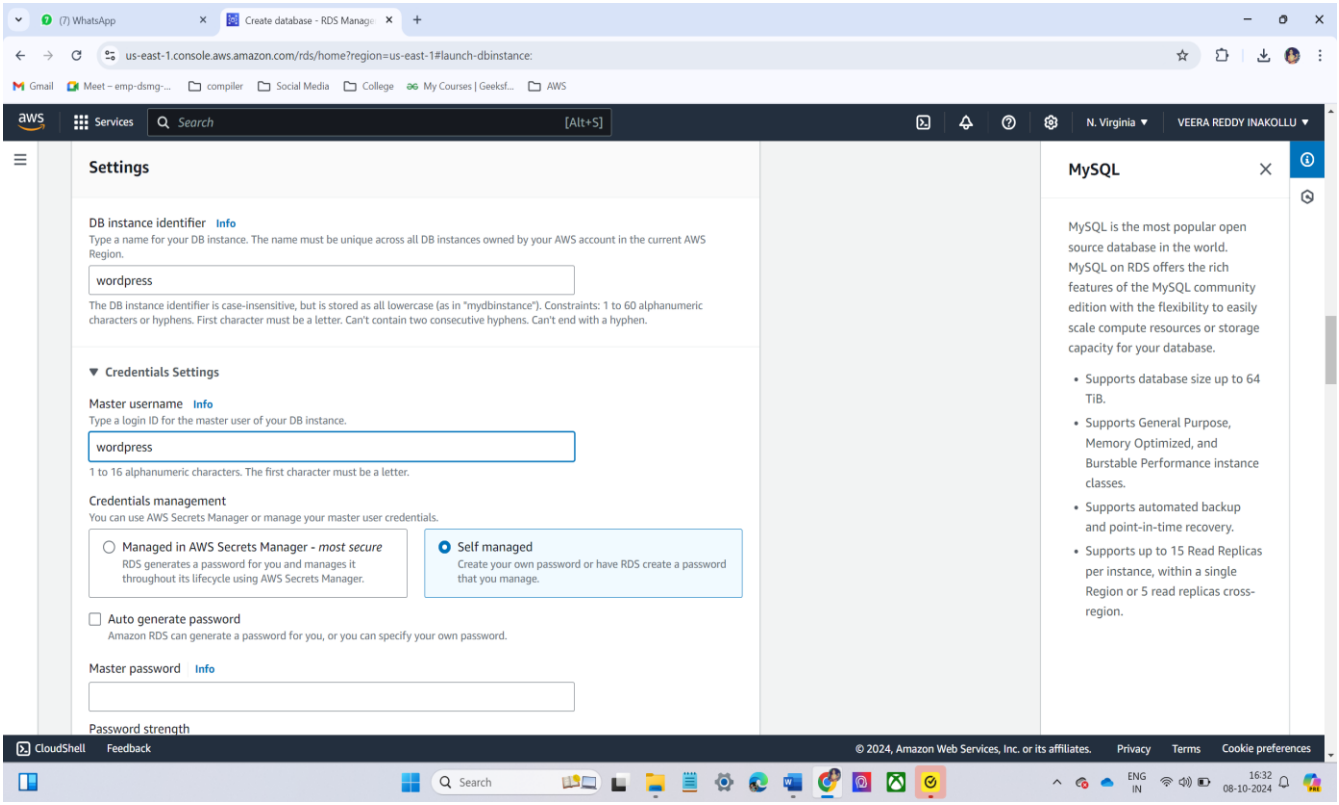
- Now select the database engine with version but I select the MySQL database engine because it's a huge usage in real time organizations.



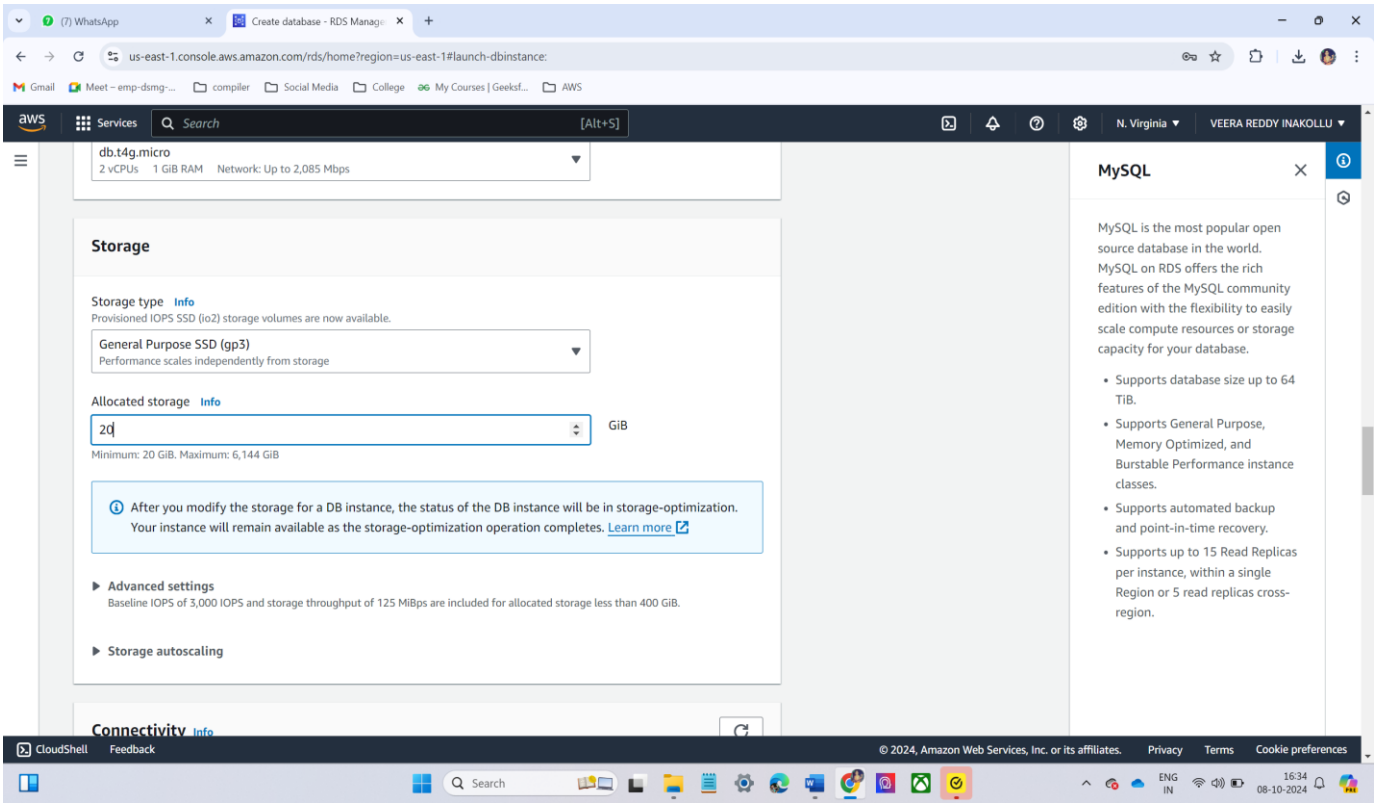
- Now select the template as free tier and by selecting the free tier it disables the availability zone section options.



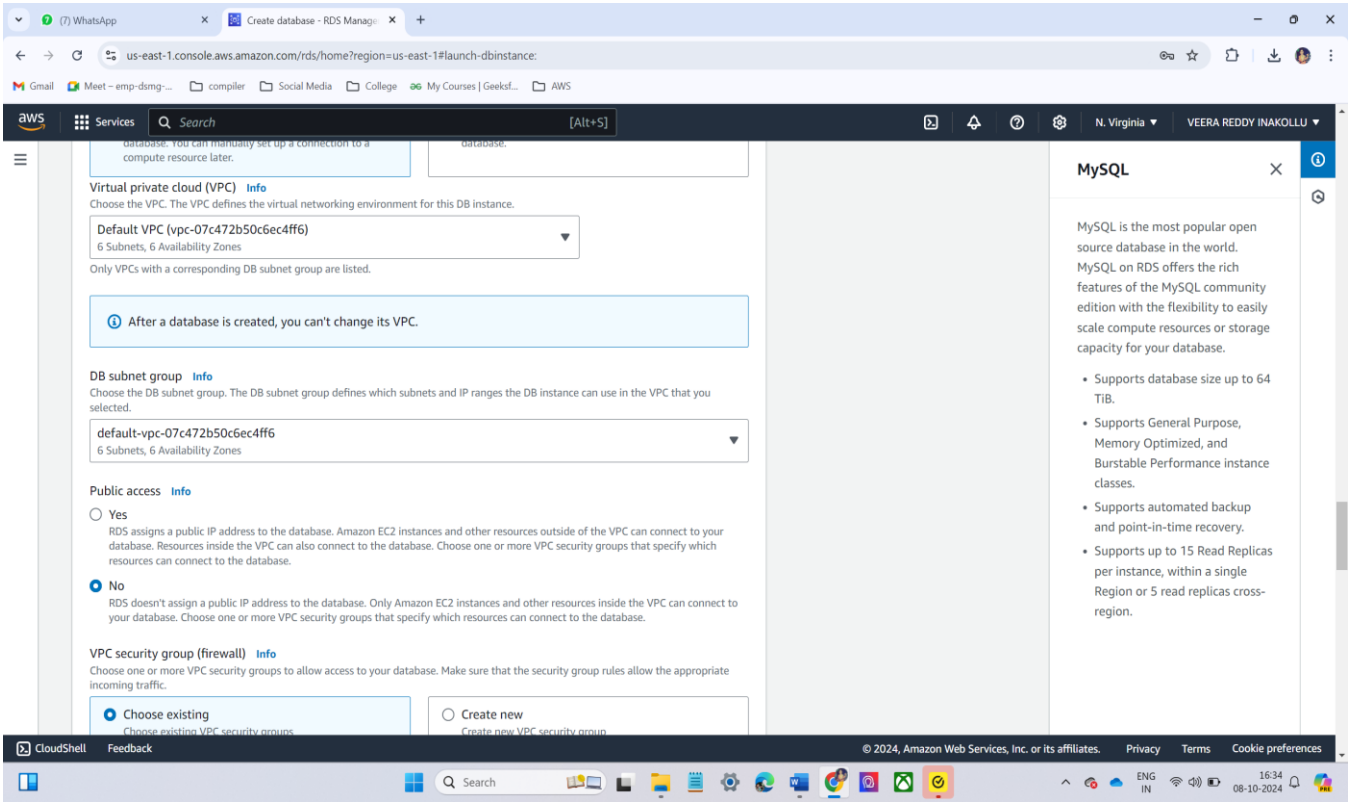
- Now give some name to your database and give username and password as credentials for your databases.



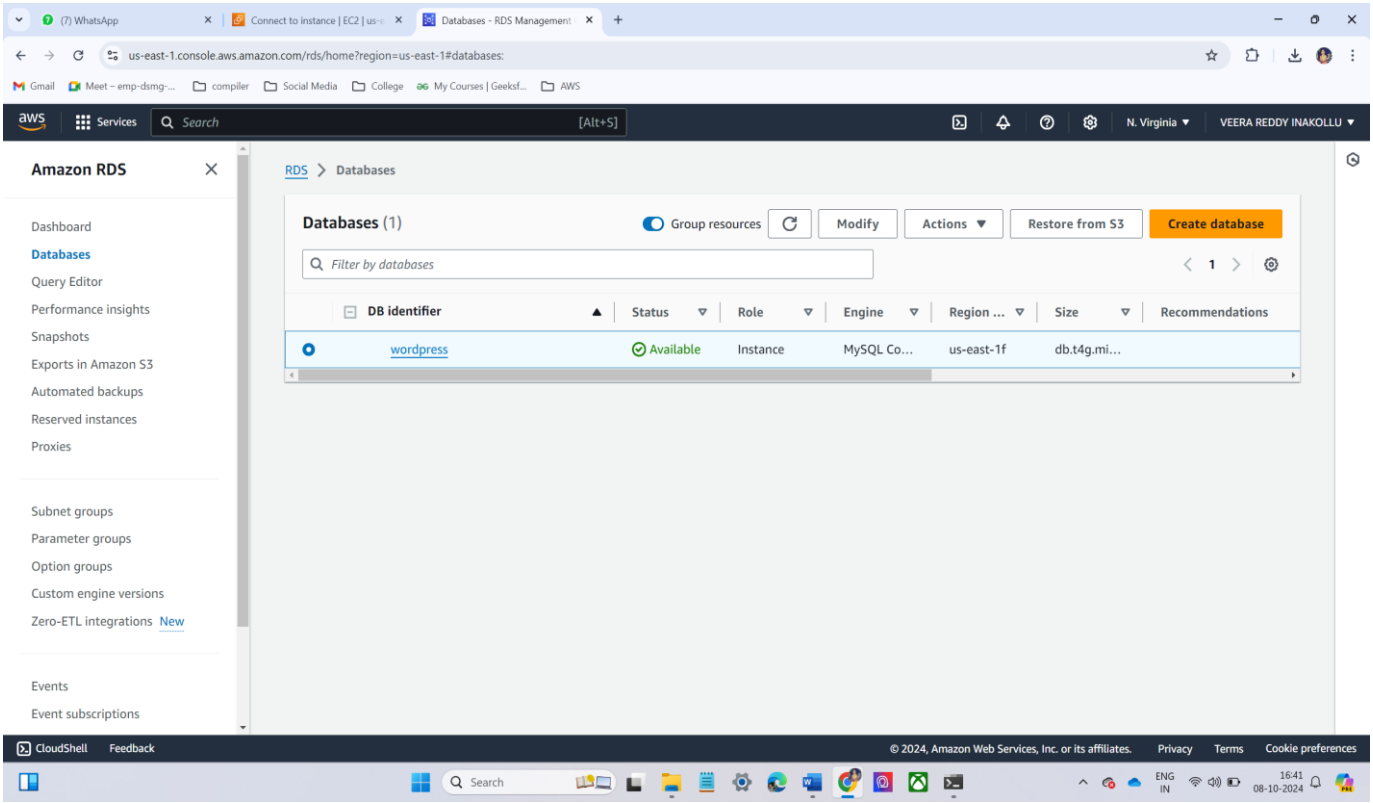
- Now select the storage type as general purpose (SSD (gp2)) and enter the storage values as maximum (1000gb) and minimum (20gb).



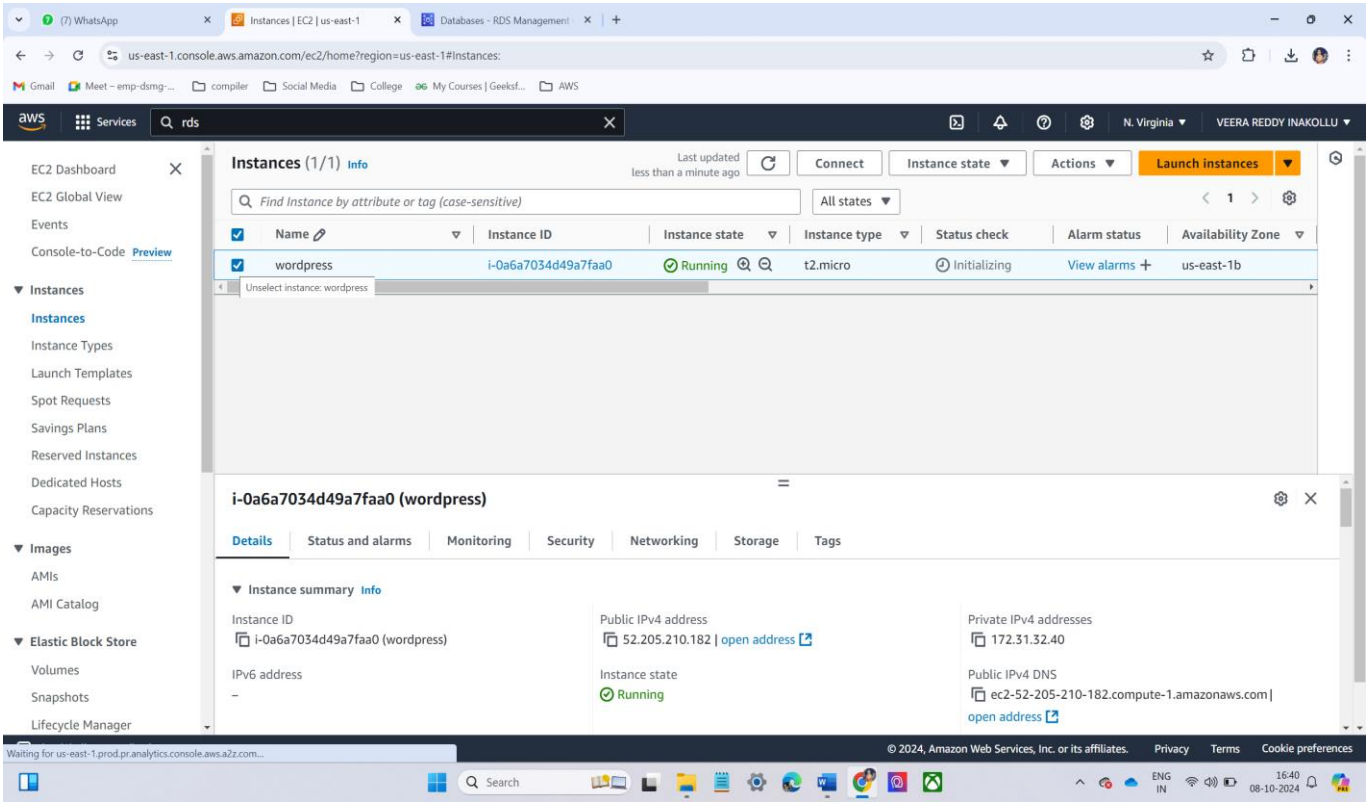
- Now select your created vpc or select default vpc and it automatically selects the database subnet group.



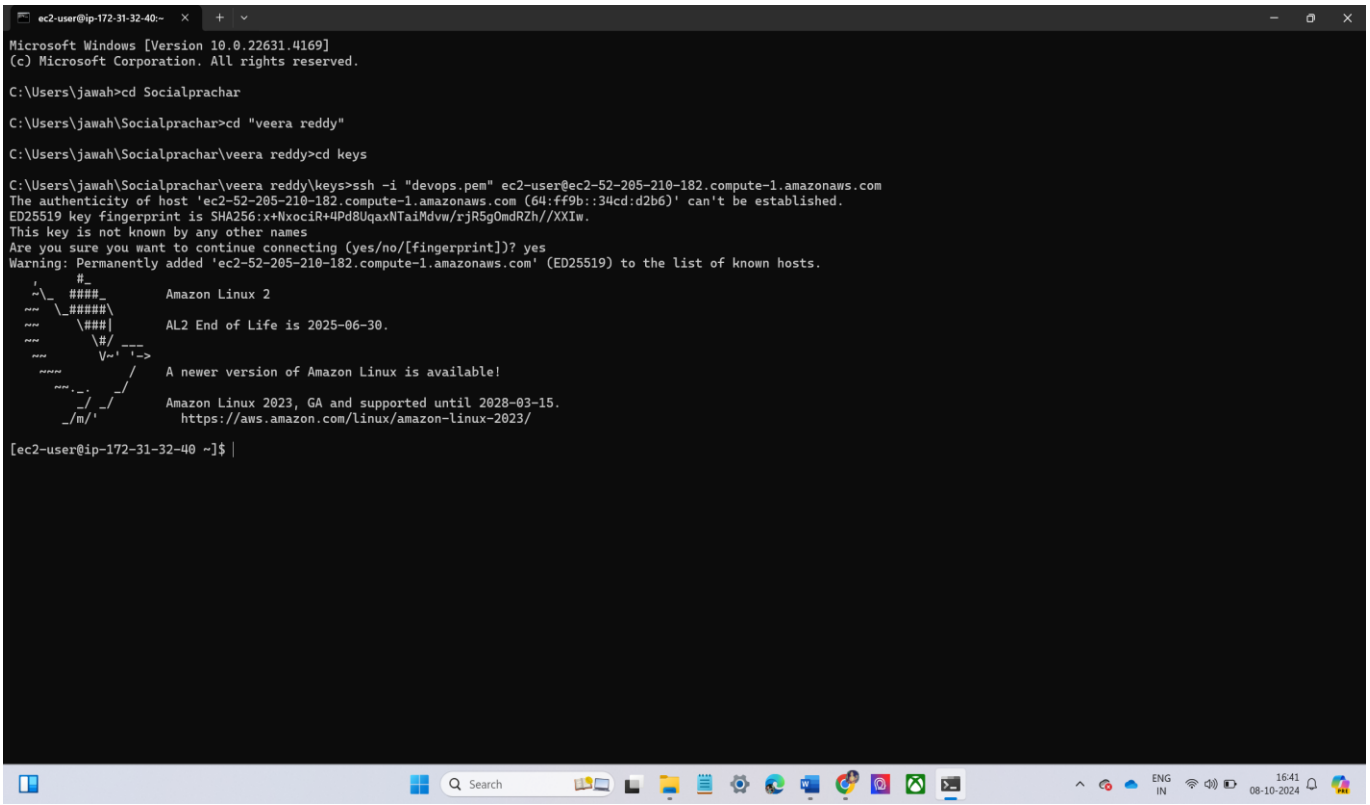
- Now give the name of the database which you give at the stage of DB instance identifier enter same name here.
- Now click on the create database button and will create the MYSQL database.



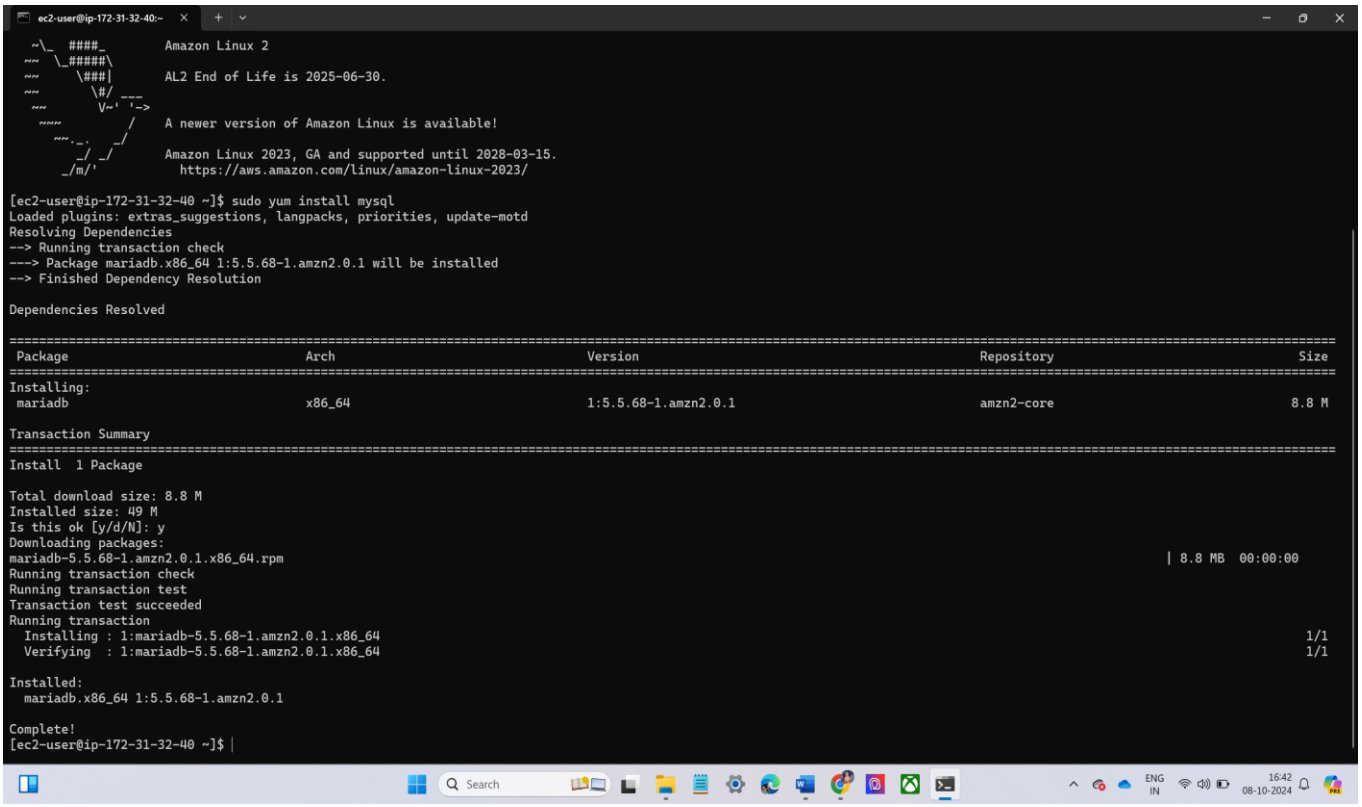
- Now create the ec2 instance by selecting ec2 instance and launch the instance by selecting Amazon Linux -2 version and giving security group with ssh(22) and http(80).



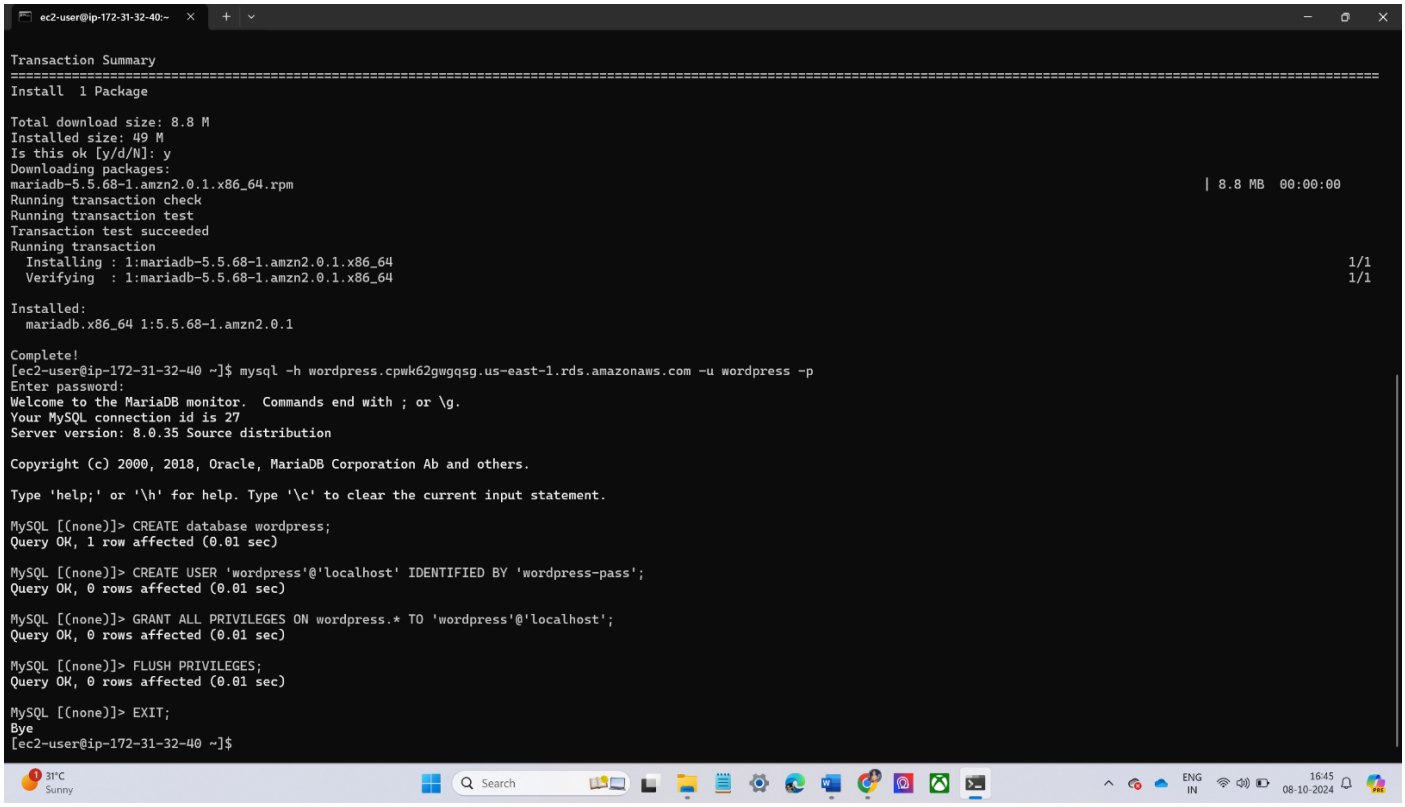
- Now connect the virtual server through the git bash as shown in fig.



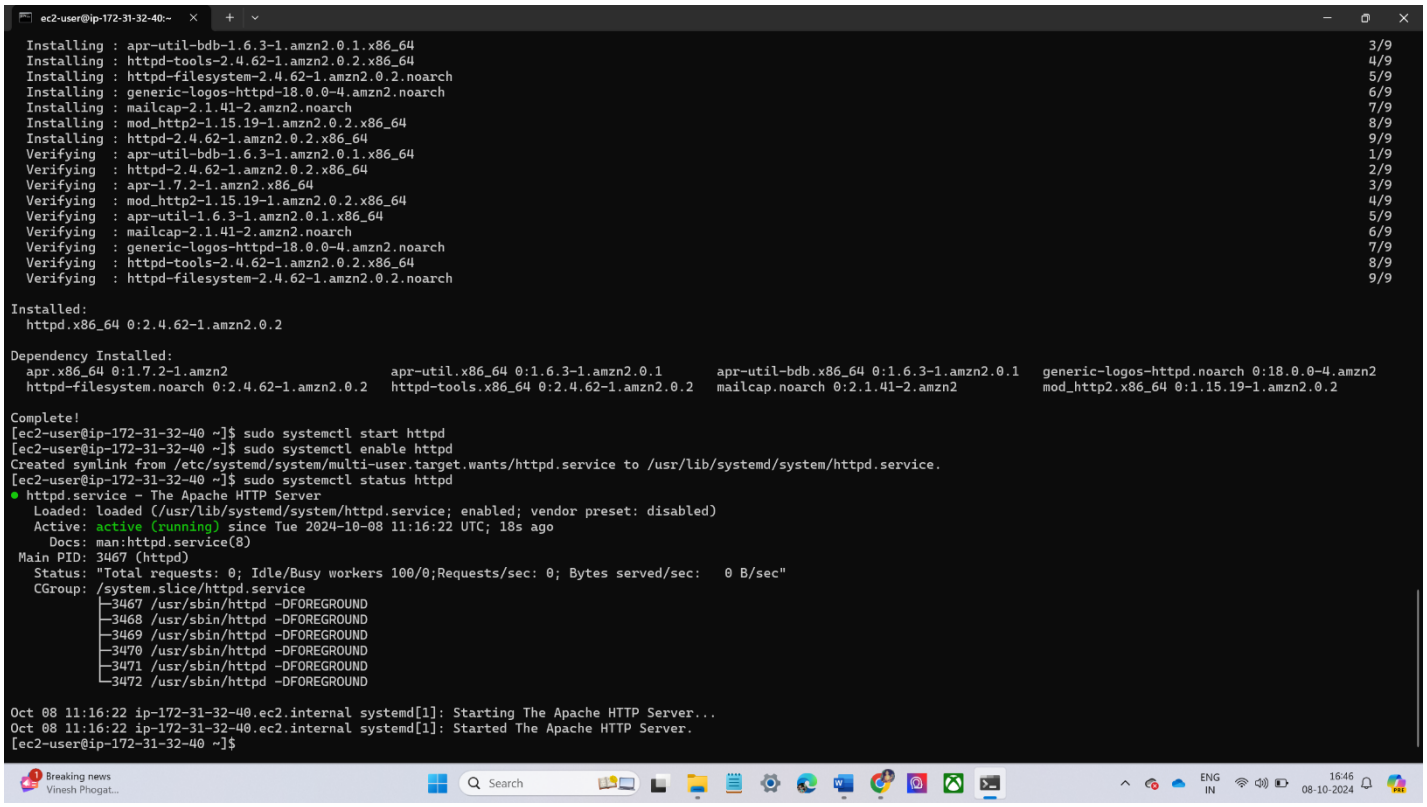
- Now update the Linux version by using command <sudo yum update -y> and install MySQL by using the command sudo yum install -y mysql2.



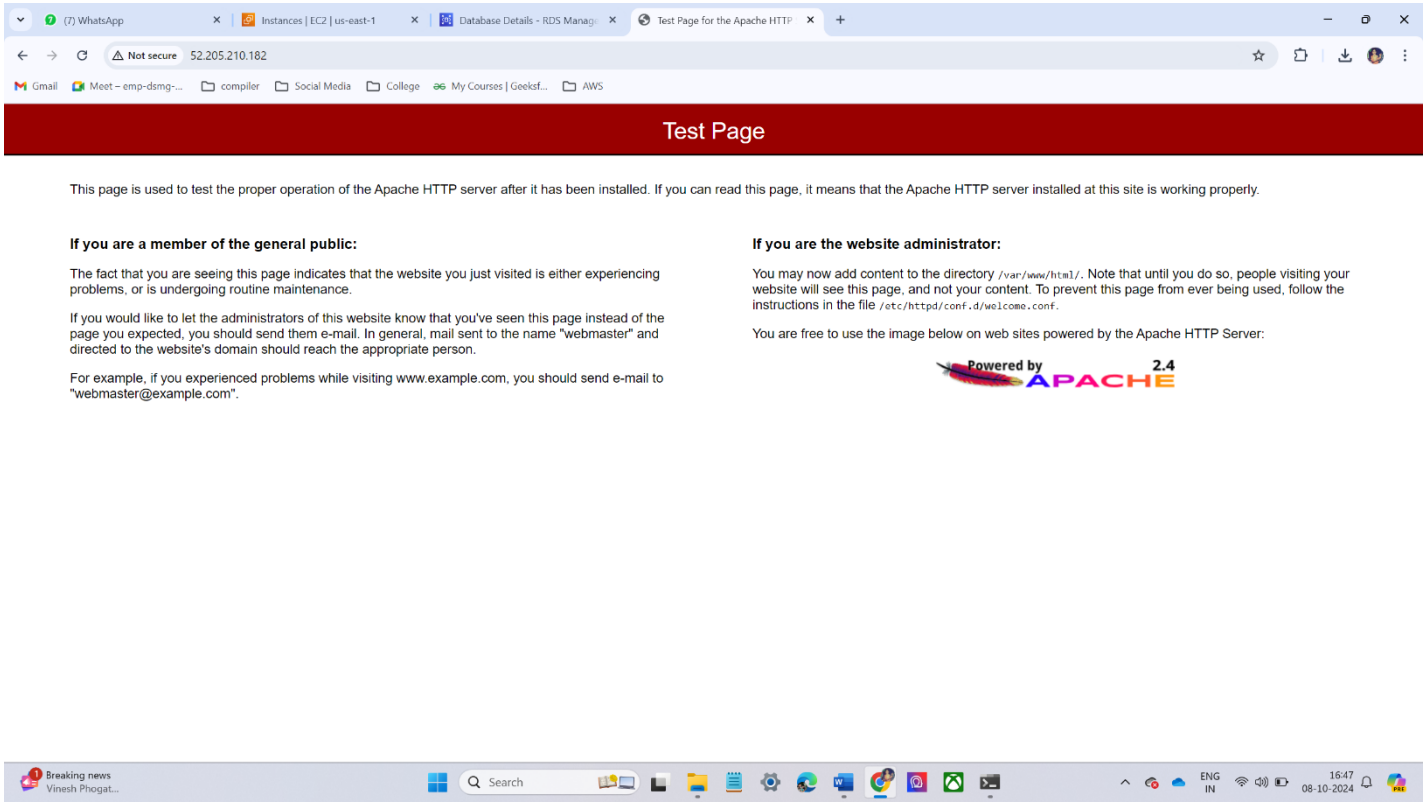
- Now to allow create traffic from ec2 instance into rds database for that go to rds service and select your database.
- Now go inside the created database and go to the security under this option these is security group id click on that.
- Now access the MySQL database by using the command as “export 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
  a. CREATE USER WordPress IDENTIFIED BY ‘WordPress-pass’
  b. GRANT ALL PRIVILEGE ON WordPress. * TO WordPress
  c. FLUSH PRIVILEGES
  d. EXIT



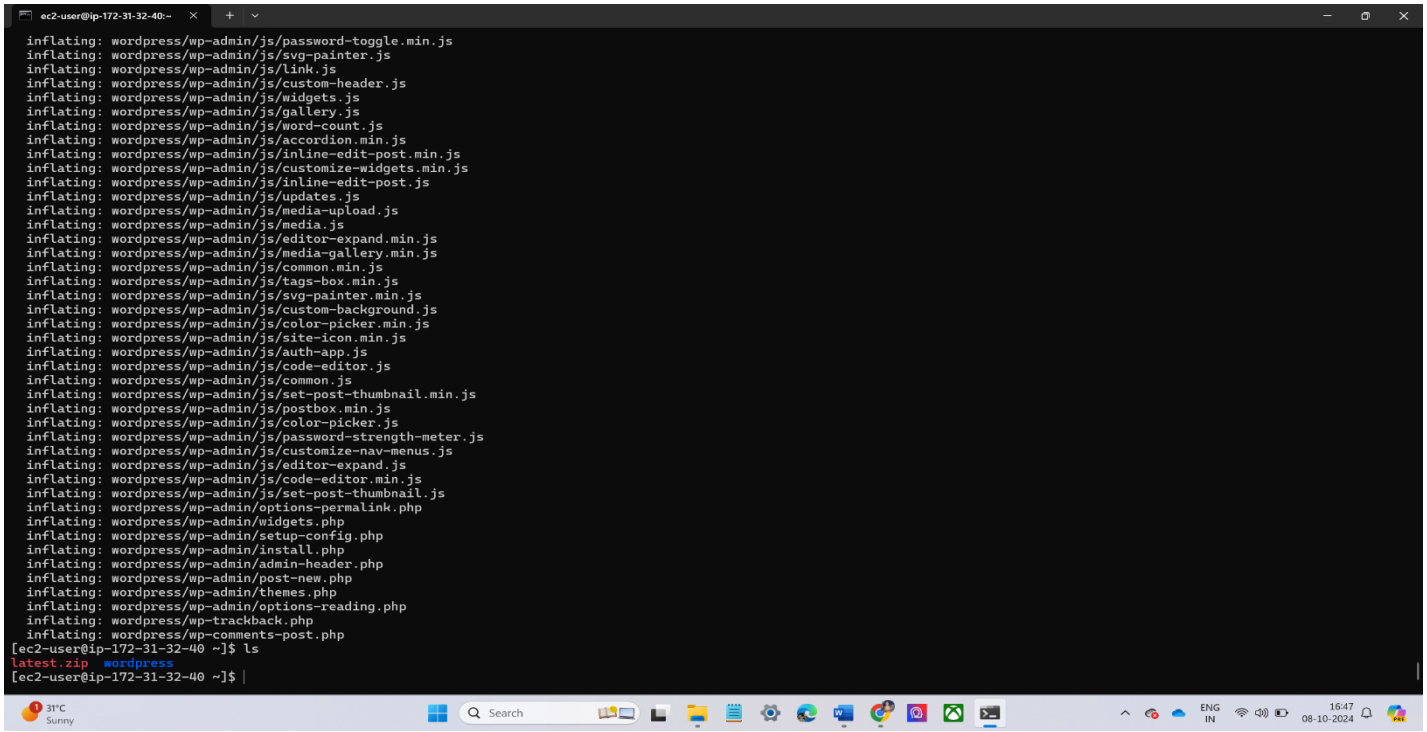
- To host WordPress application need a Apache web server httpd install that by giving command as <sudo yum install httpd -y> and start and enable the httpd service by giving the command as
  a. “sudo service httpd start (or) sudo systemctl start httpd
  b. Sudo chkconfig httpd on (or) sudo systemctl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



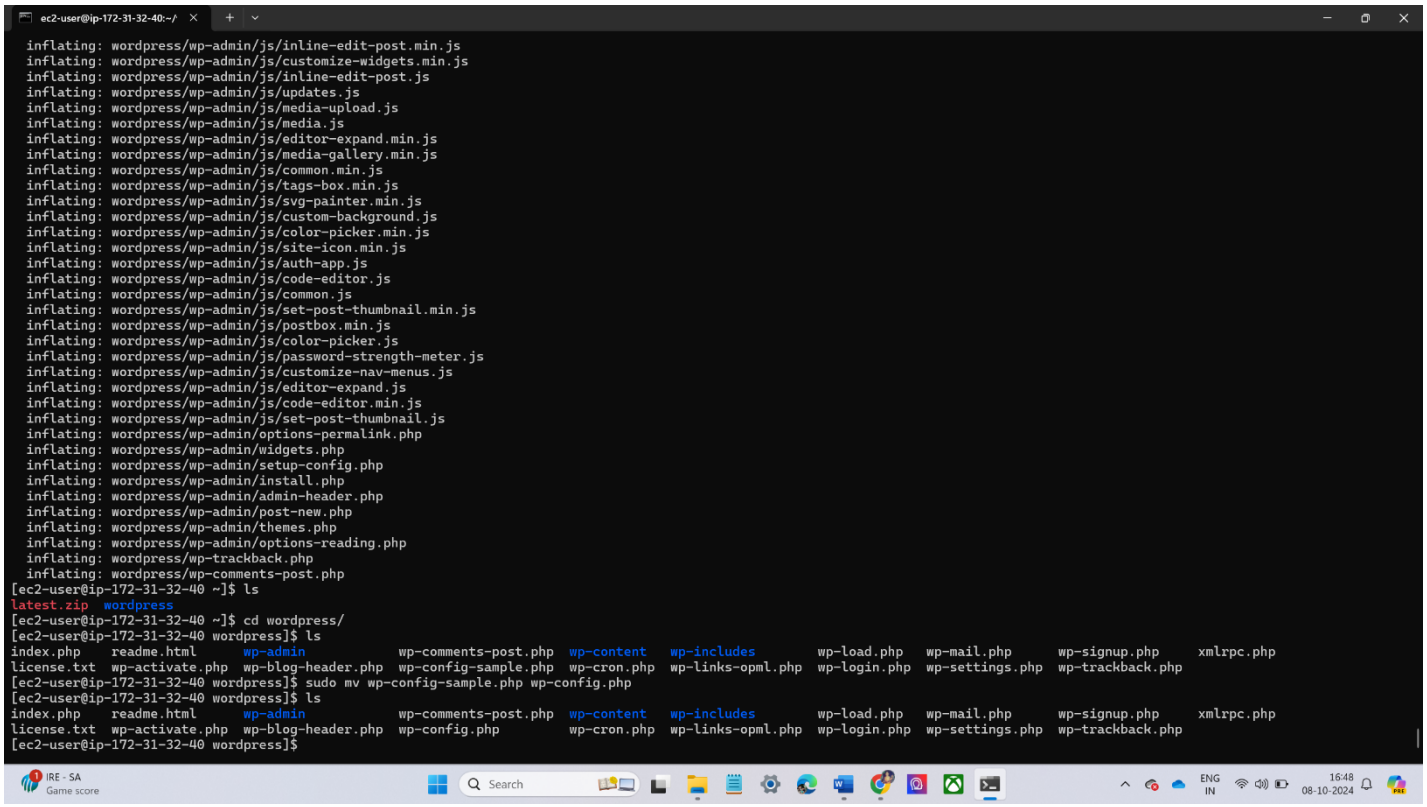
- Now go to ec2 instance and copy the public ipv4 it on Google. Browse it and check the official page of httpd is displayed



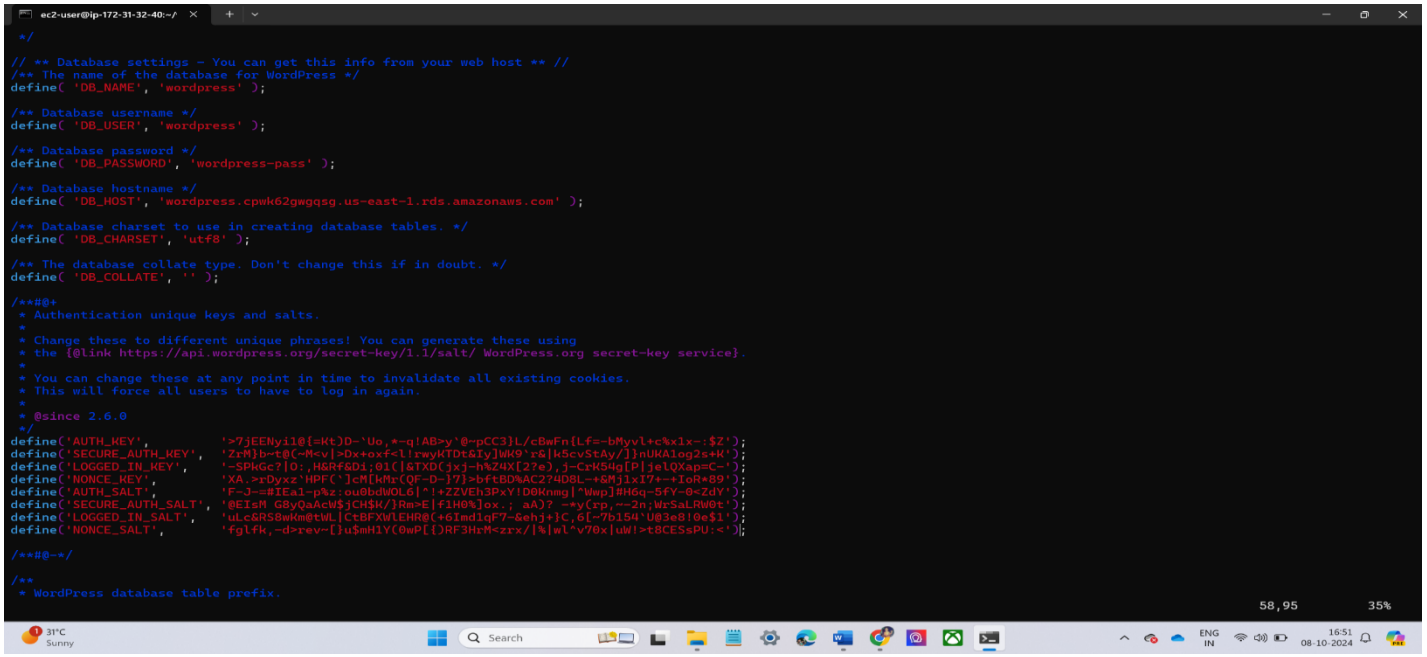
- Now go to browse and select a download WordPress and click on proper link and select and copy the address link of WordPress. Download the file and past that along with wget command in gitbash as “wget <address. Link>”
- It gives the zip file to unzip that file by using a command as “unzip <zip file>”



- 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”



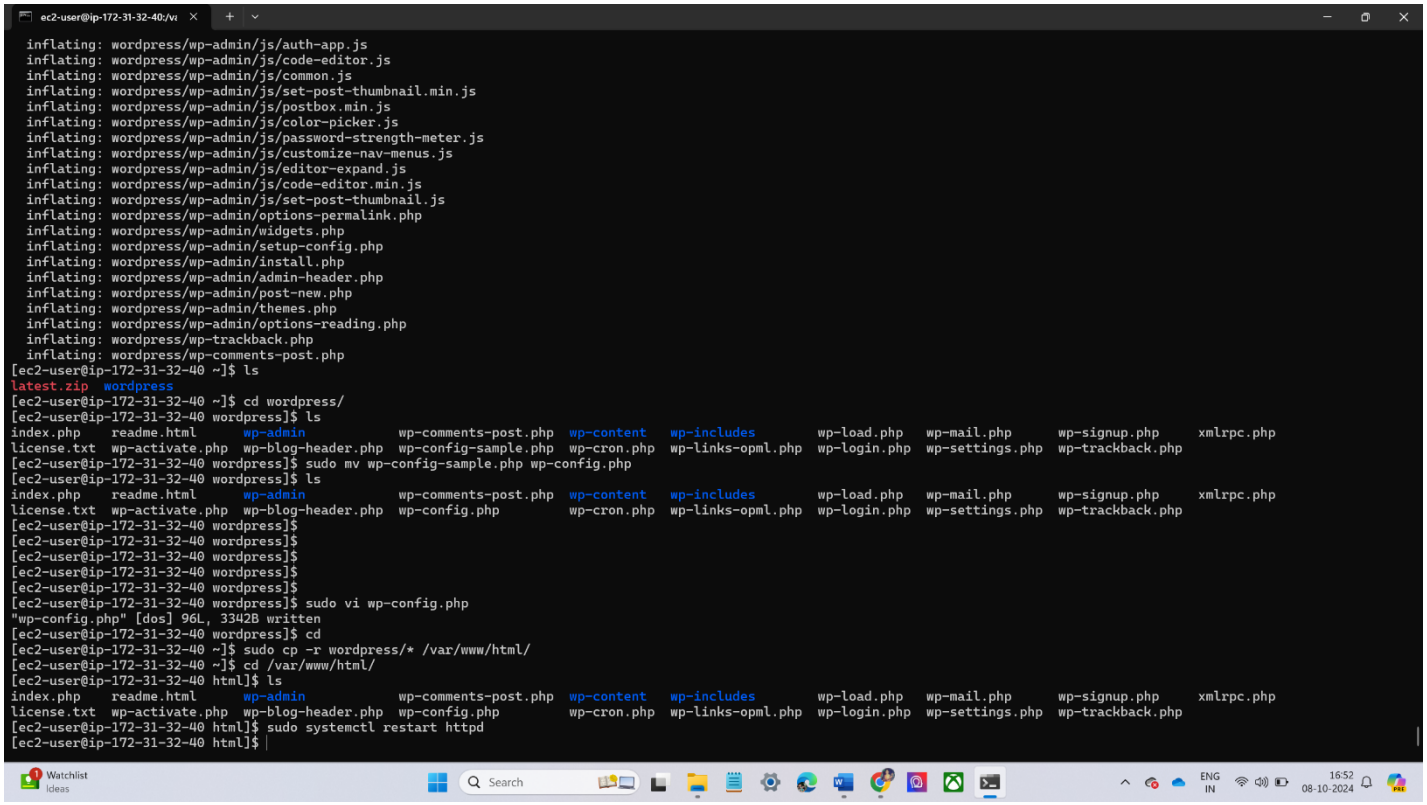
- Now do some configurations in the WordPress configuration file by giving database name, username, password and hostname for that execute as sudo vi wp-config.php” .along with these change the WordPress secret key. To get the secret key browse the Google generate WordPress secret key.



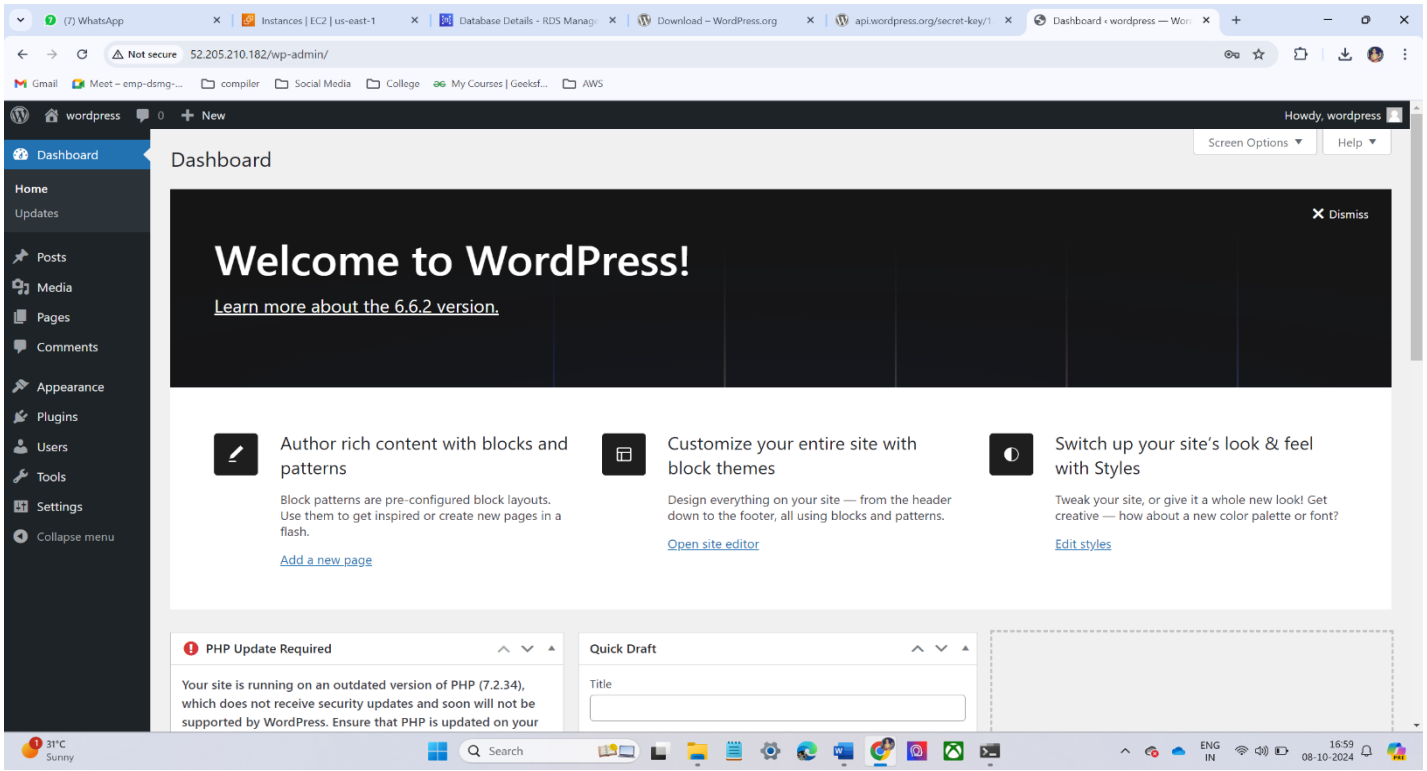
- Now copy the WordPress directory to the document root directory to host the web application of WordPress by giving as command as

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

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

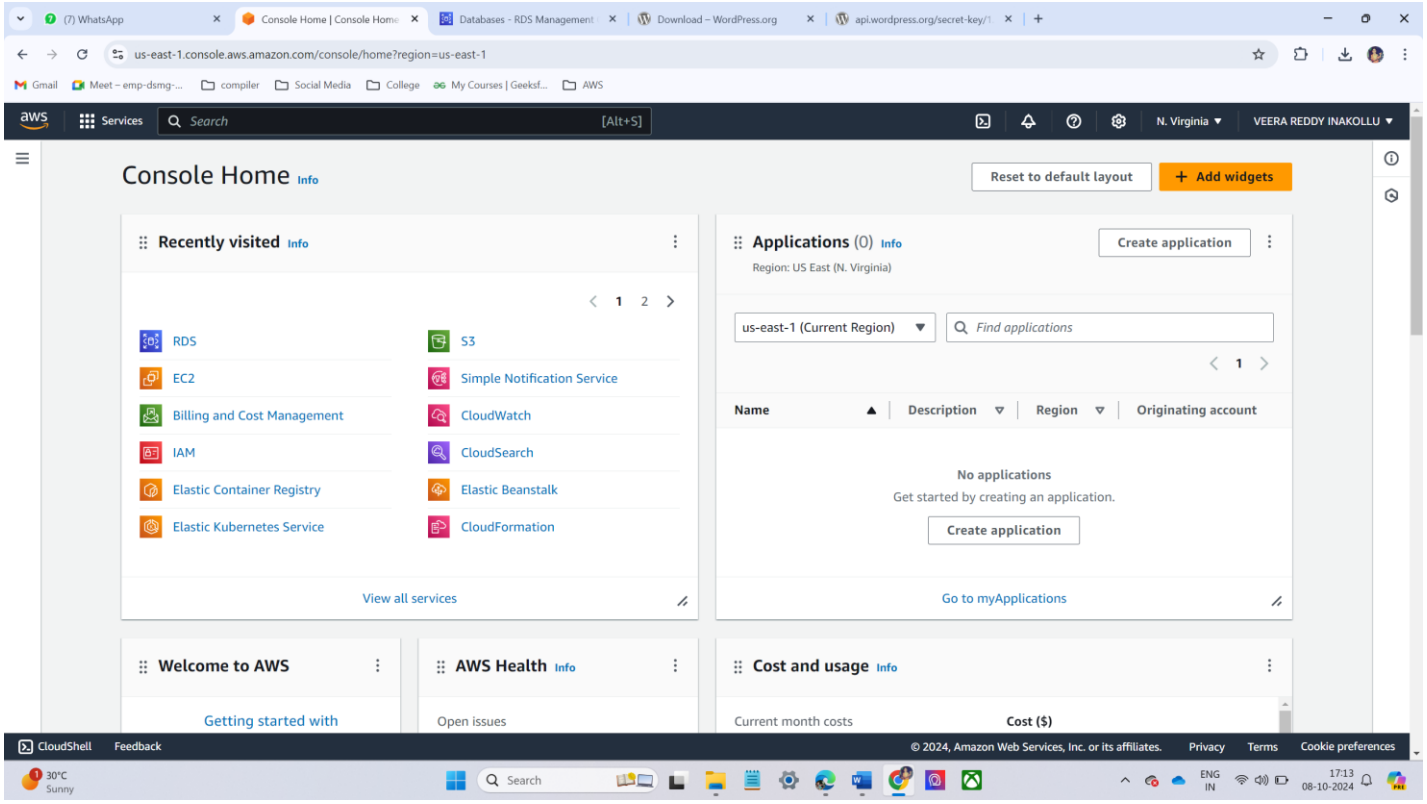


- Now go to ec2 instance and copy public ipv4 and paste it on google browse it and check the official page of WordPress it displays

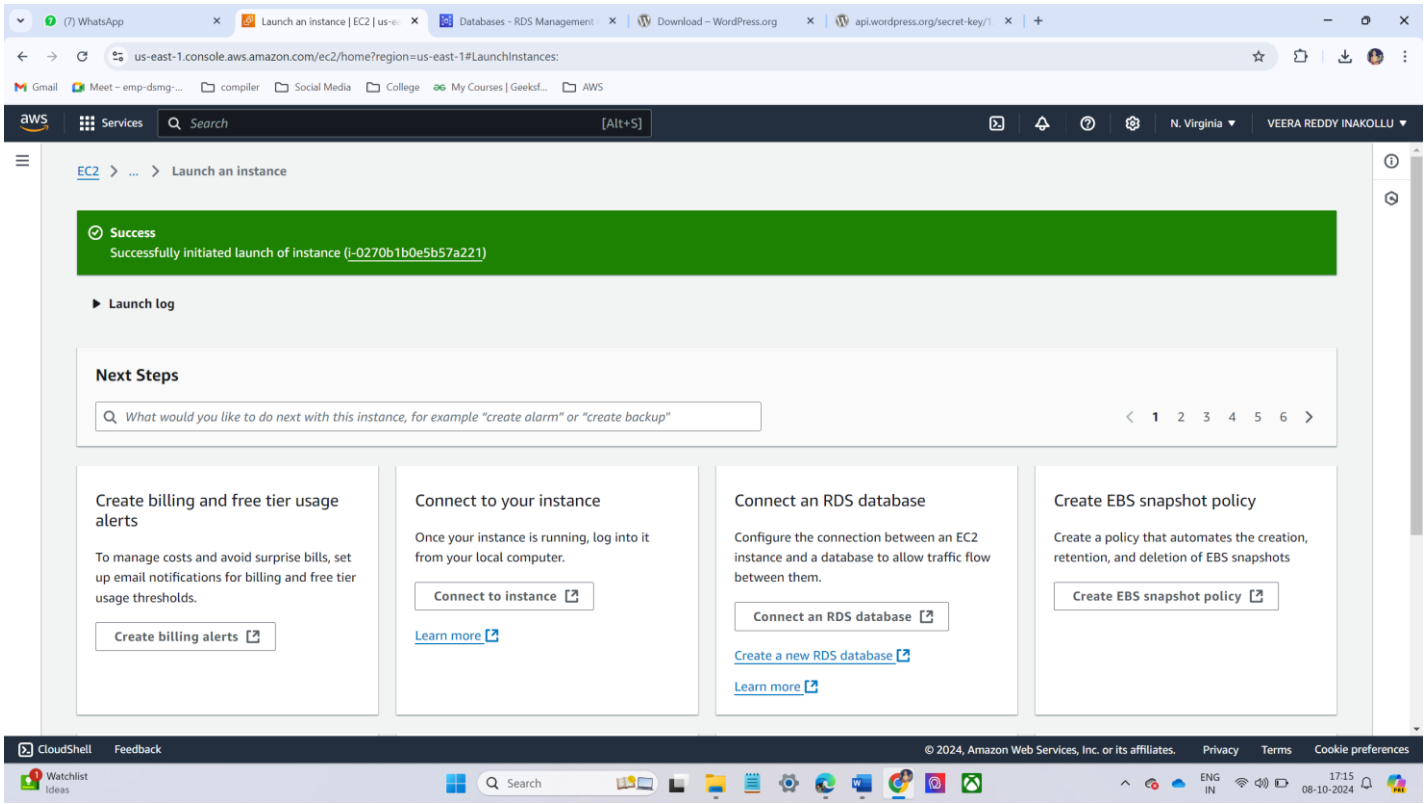


Deploy WordPress Web Application By Using Docker Compose File

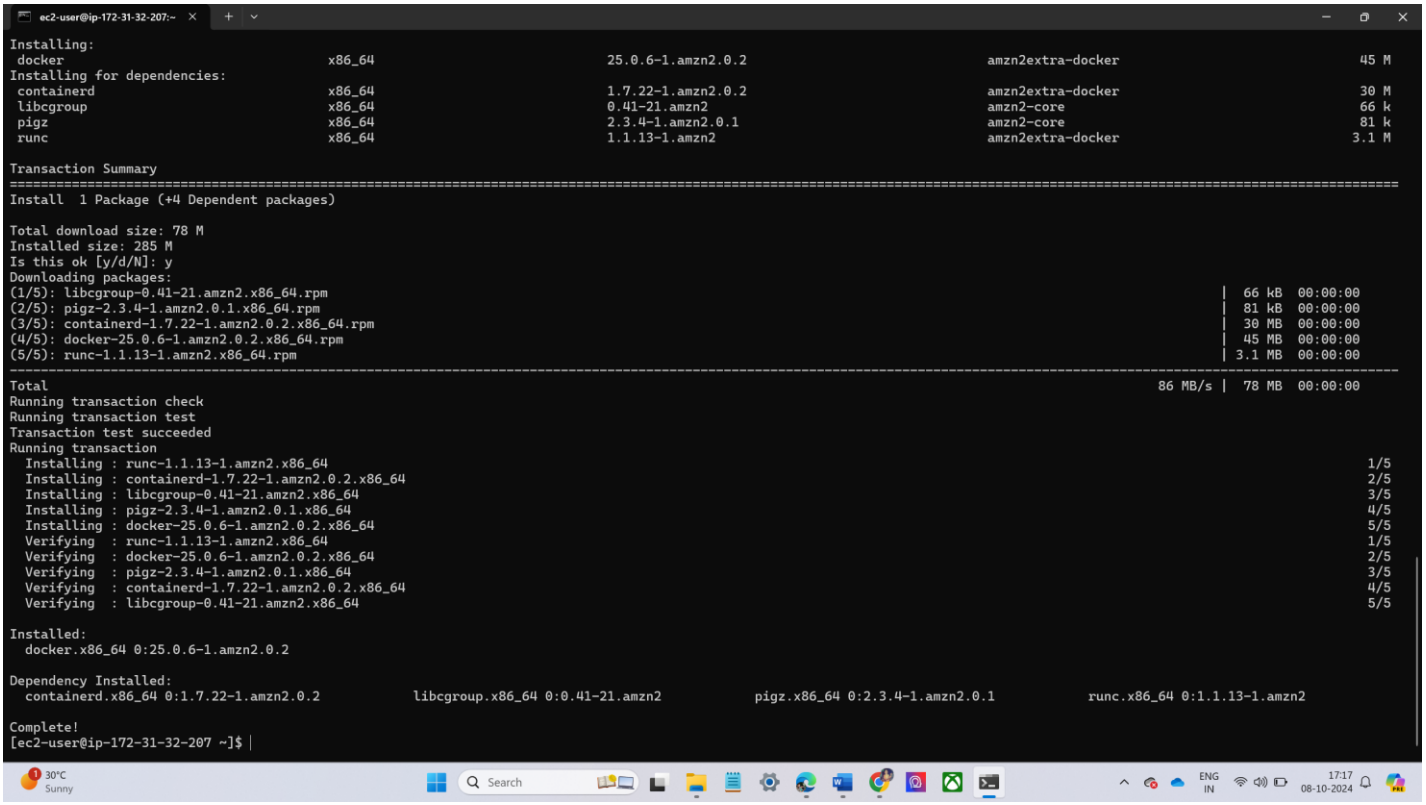
- First login to the aws account with credentials.



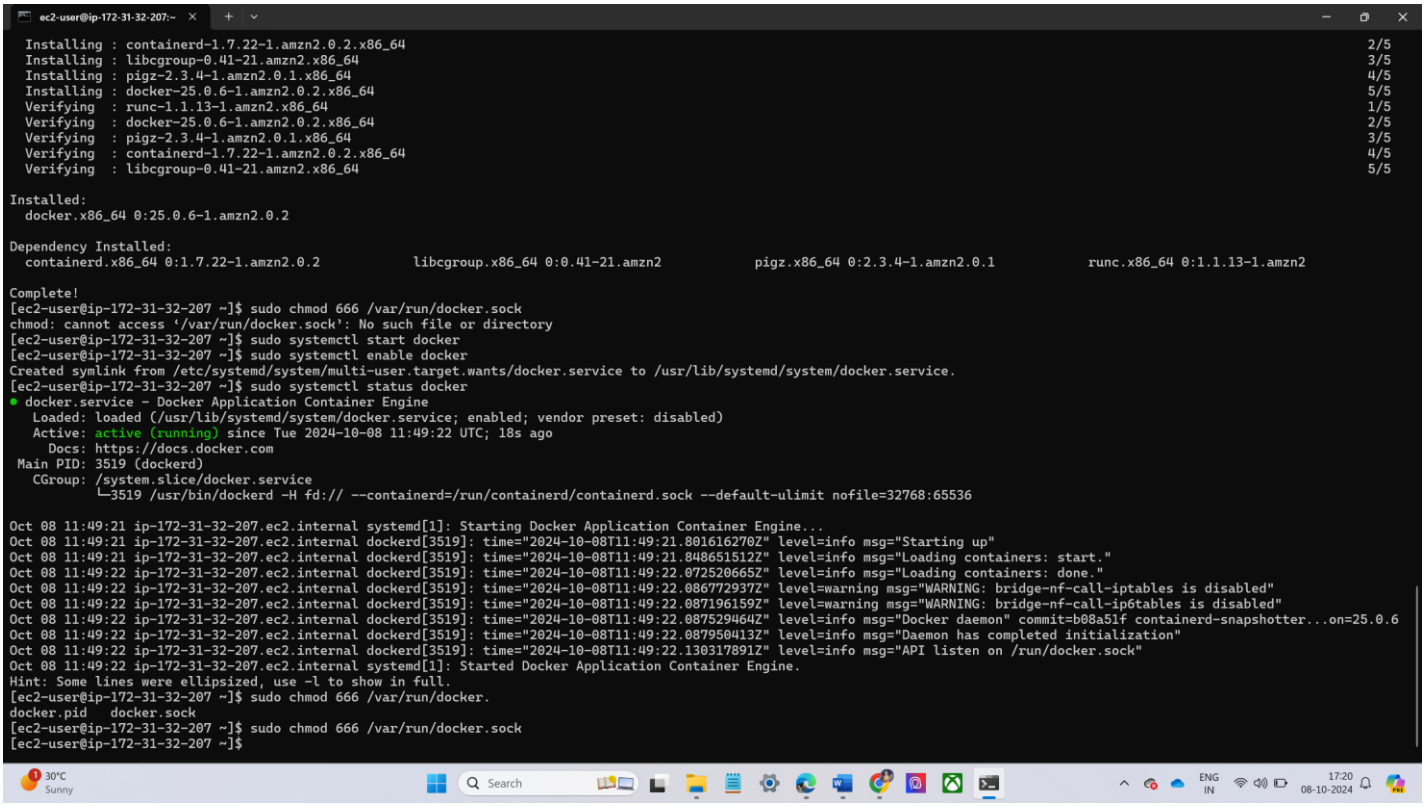
- Launch the instance allow port 80 and 22.



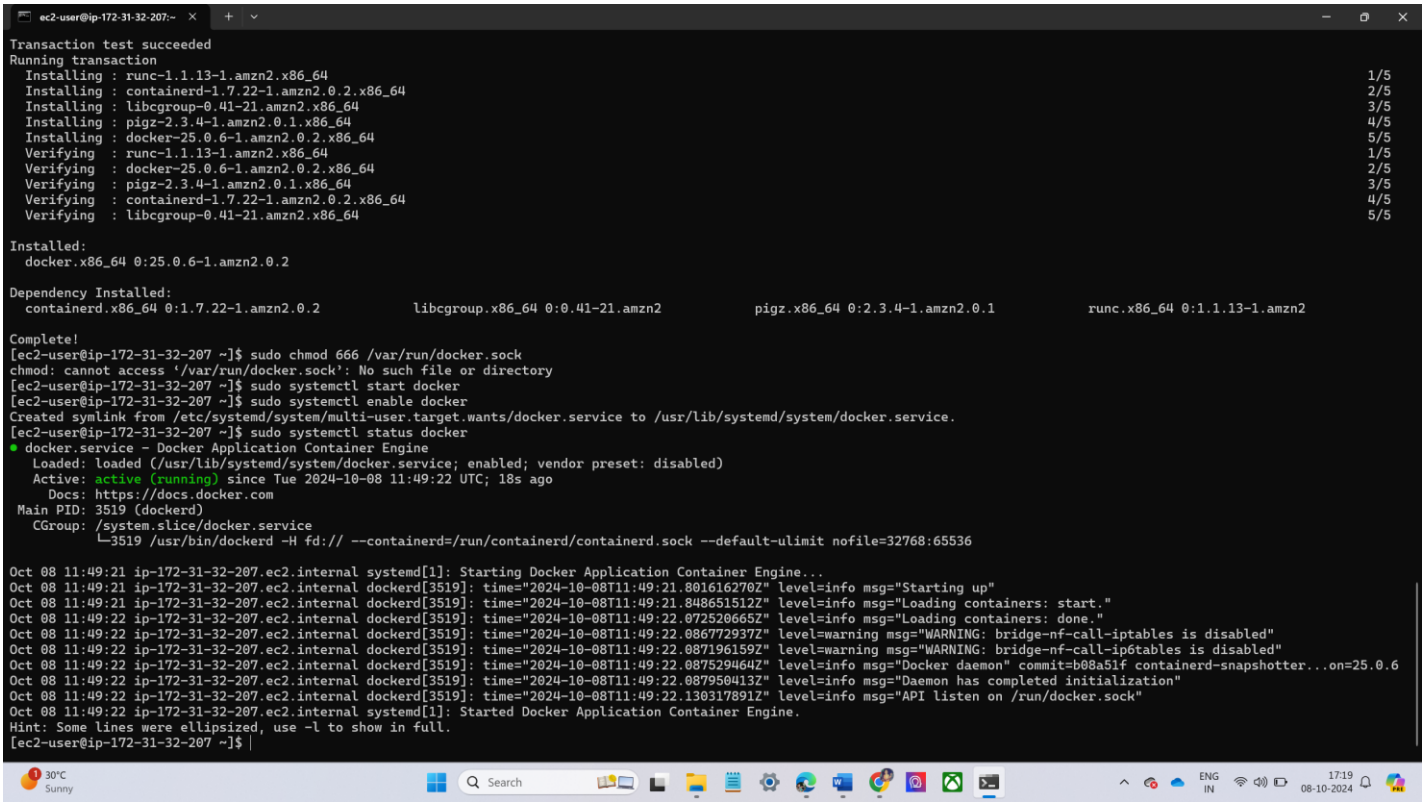
- Connect instance to the gitbash and install git and docker



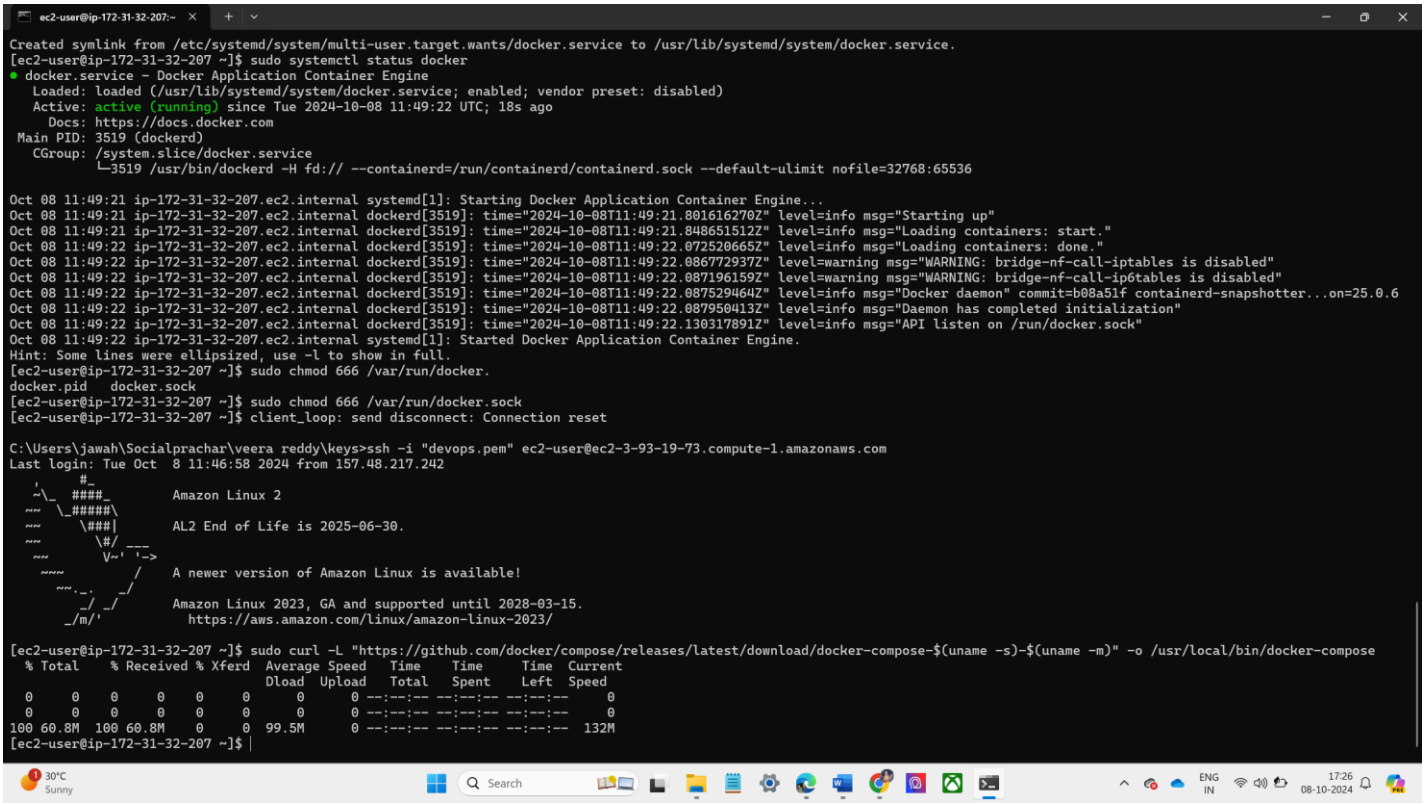
- Now give permissions to limited Linux user account to docker group by using command sudo chmod 666 /var/run/docker.sock



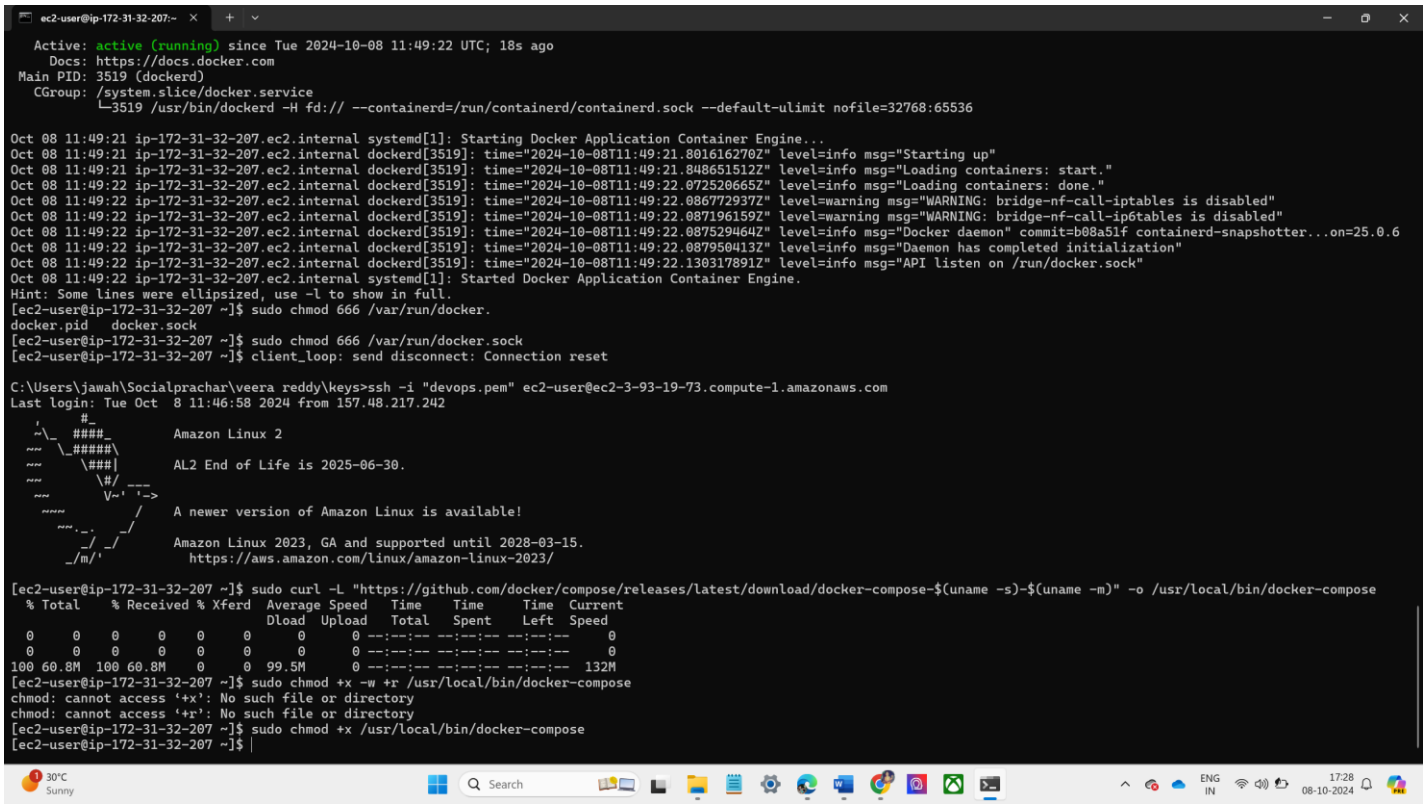
- Now start and enable the docker.



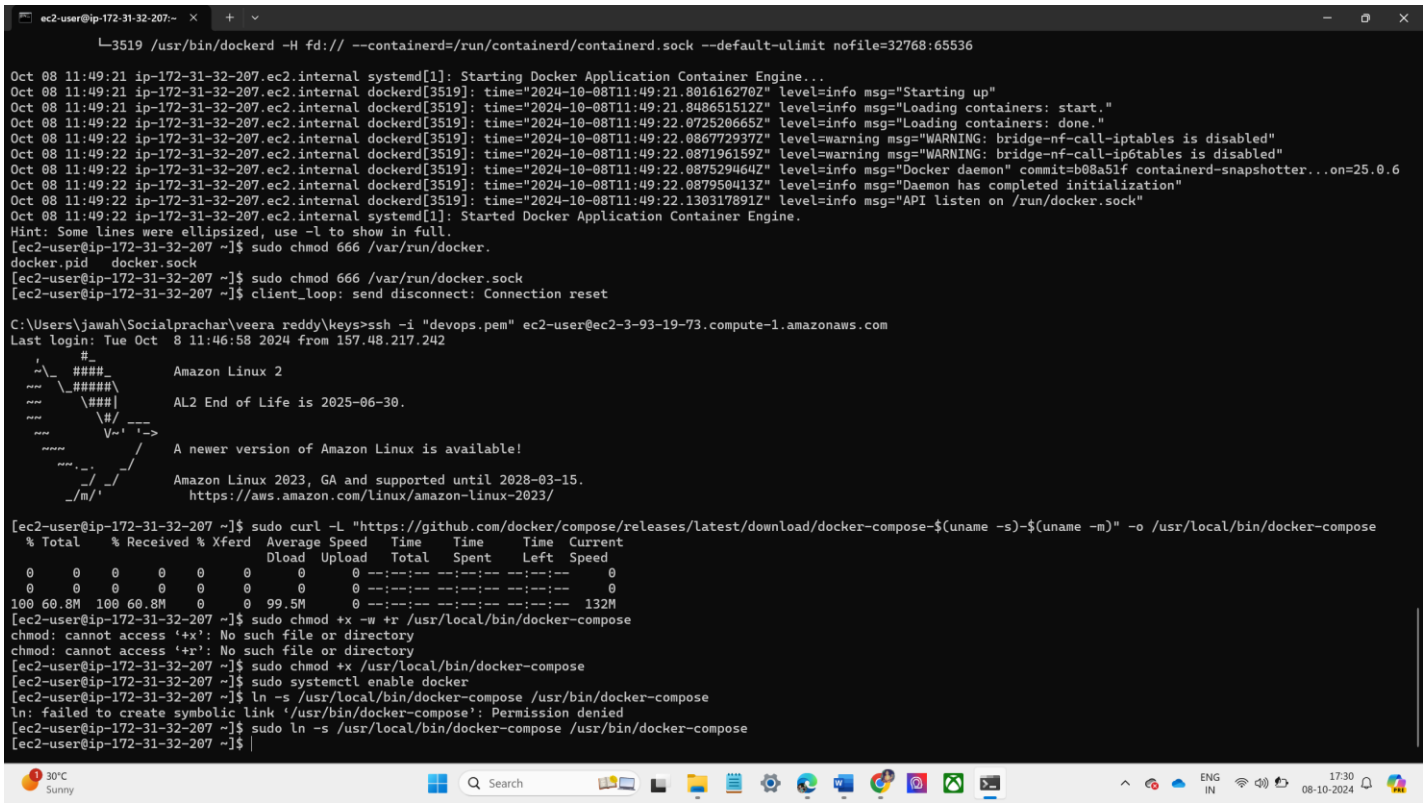
- Now install docker compose by using google.



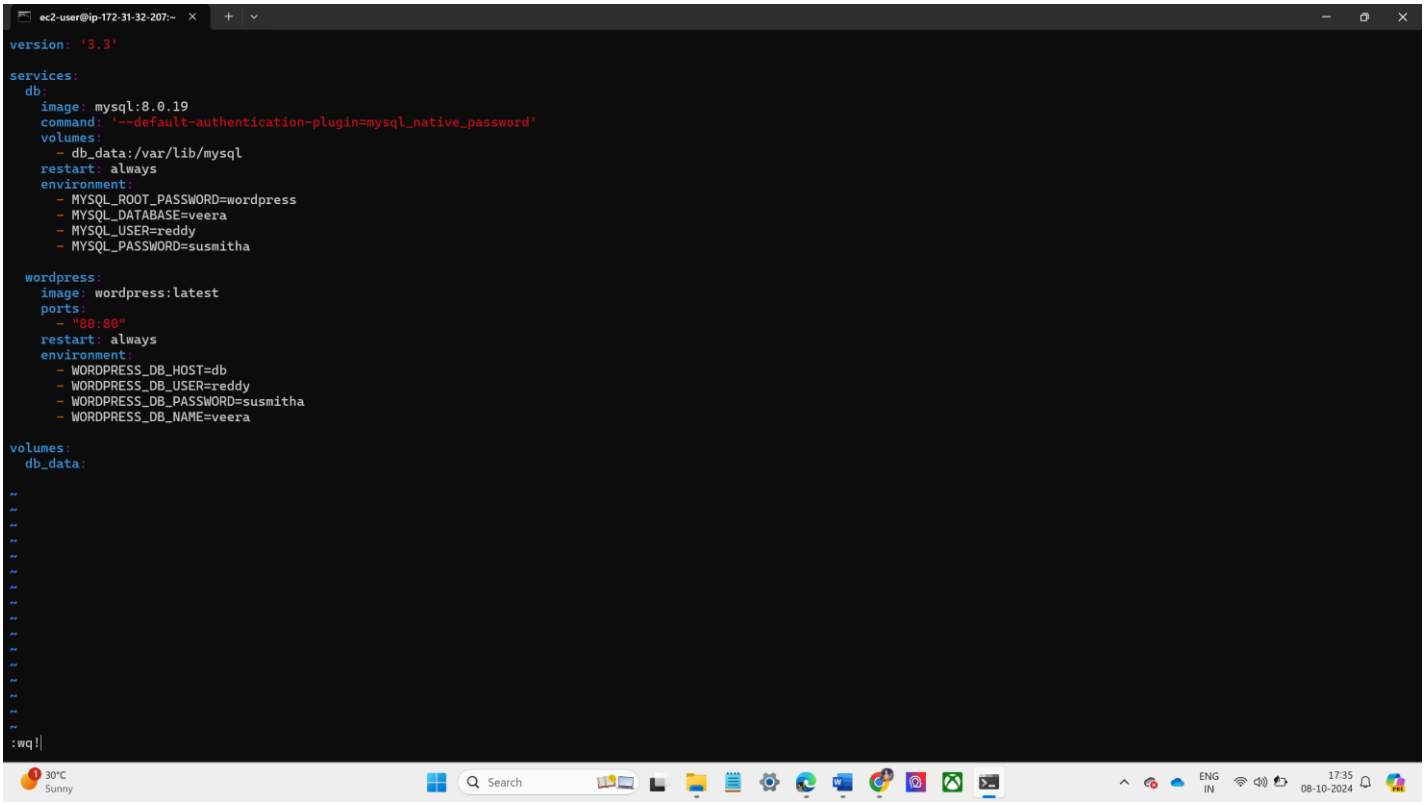
- Apply excitable permissions to the by using command sudo chmod +x-w +r /usr/local/bin/docker-compose



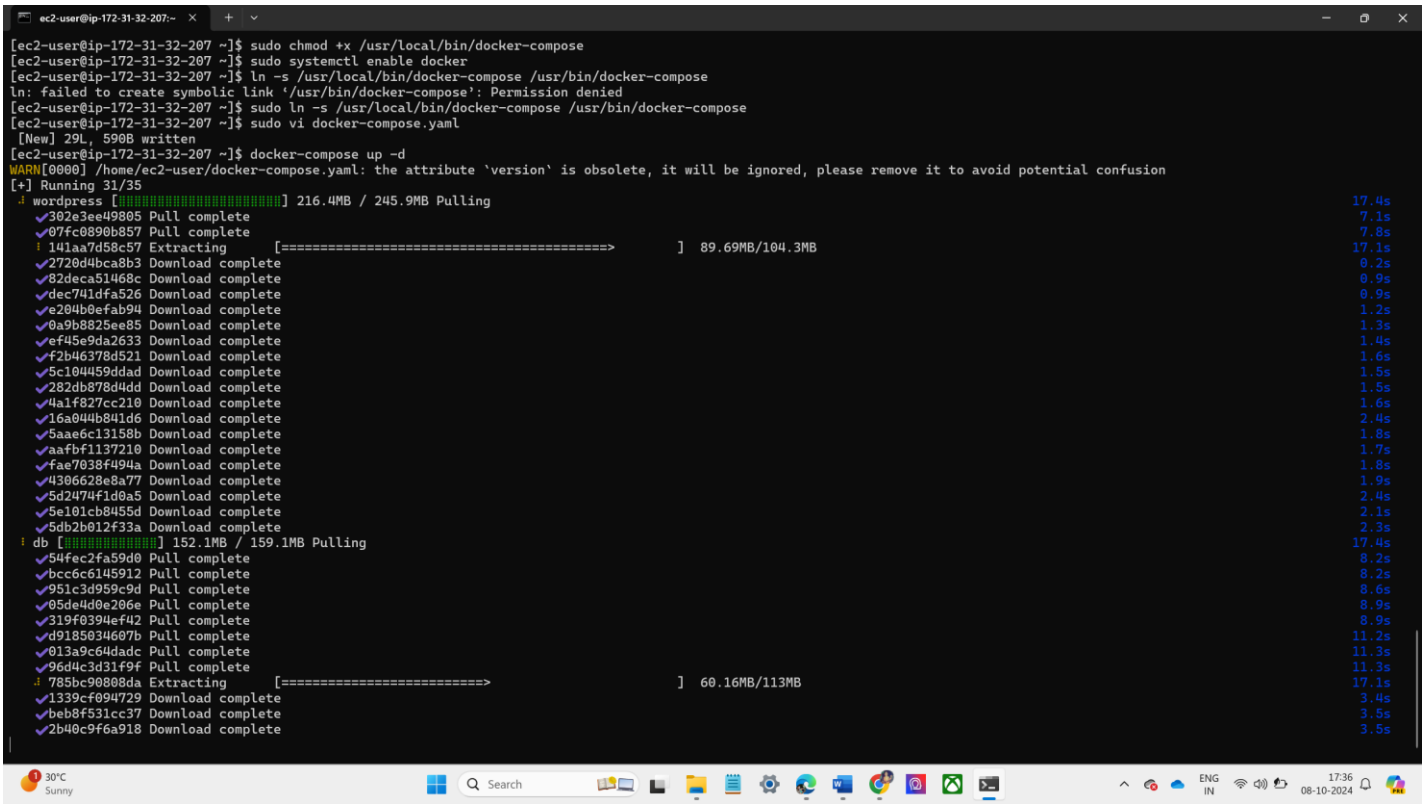
- Apply create symbolic link using command as <ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose



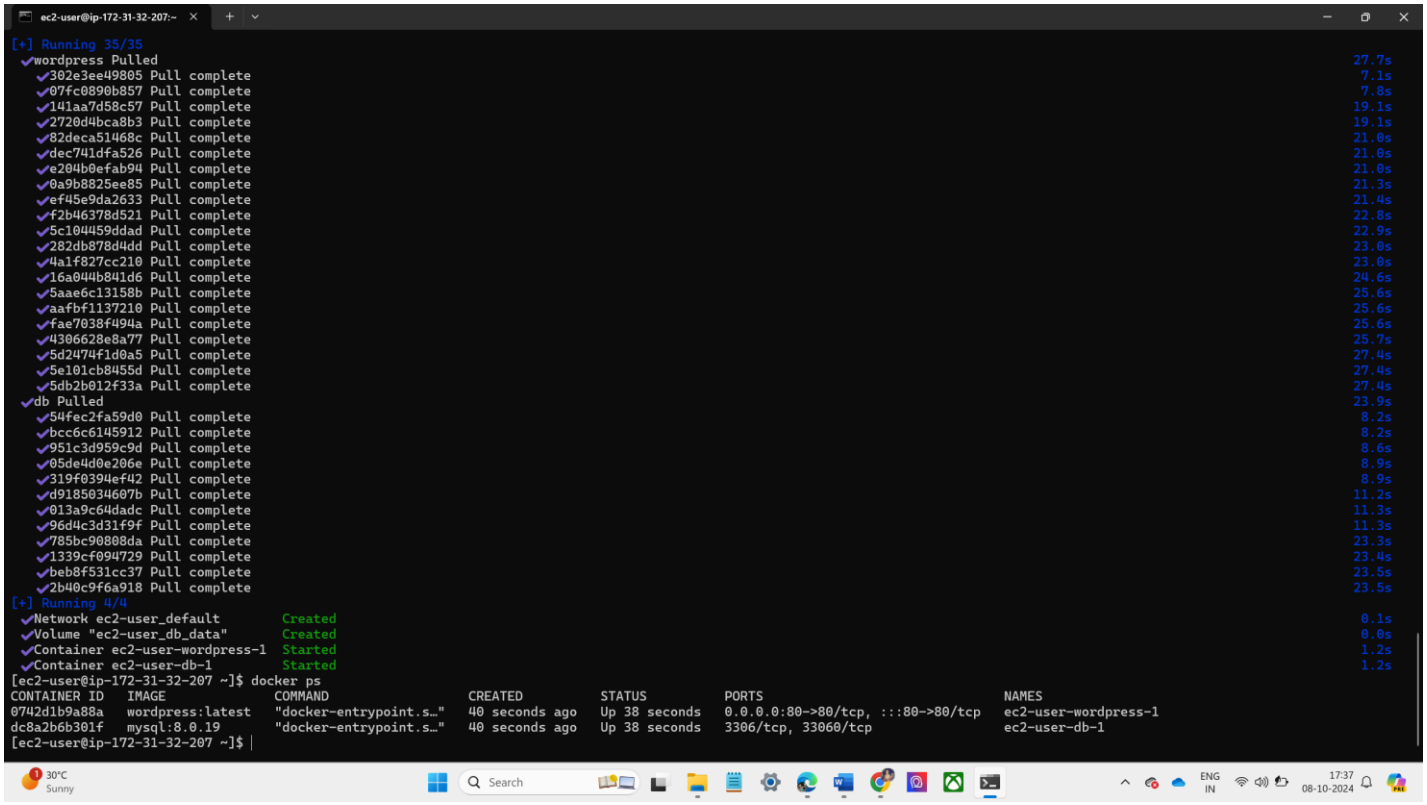
- Now create docker-compose.yaml file to pull the image form docker hub



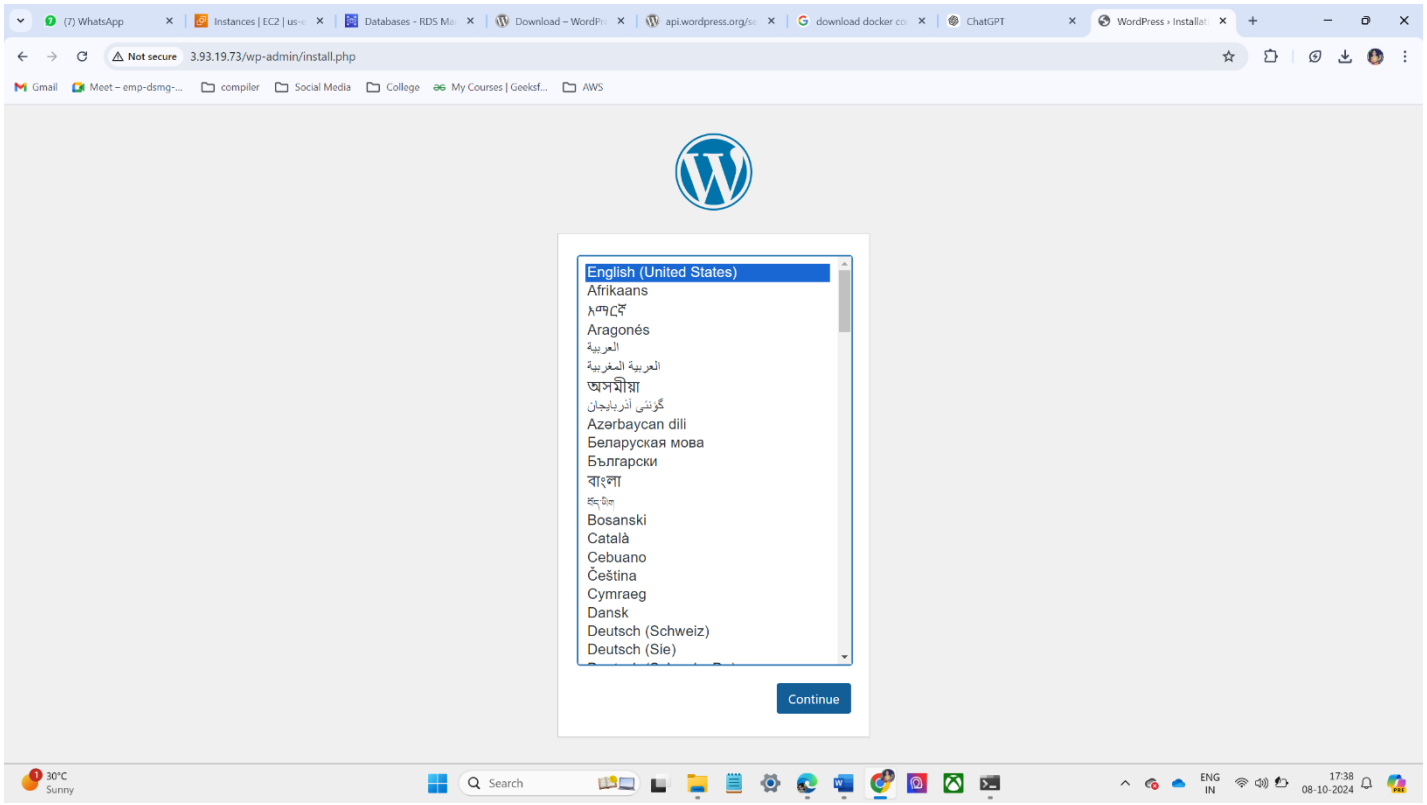
- Now to pull the image of MySQL and WordPress. We have executed the create docker compose. yaml by using the command <docker-compose up -d>

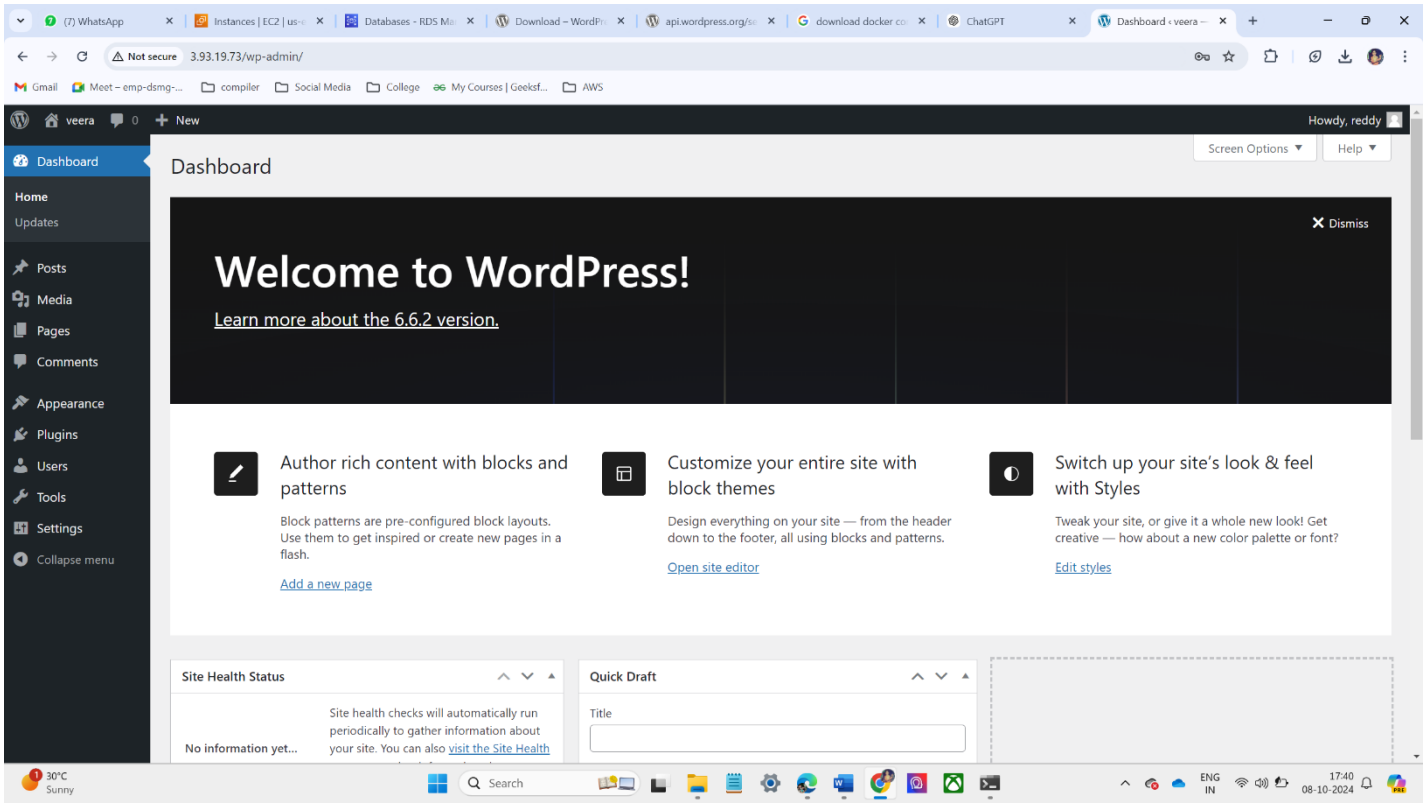


- Now see the container of the pulled images of mysql and WordPress by using command docker ps



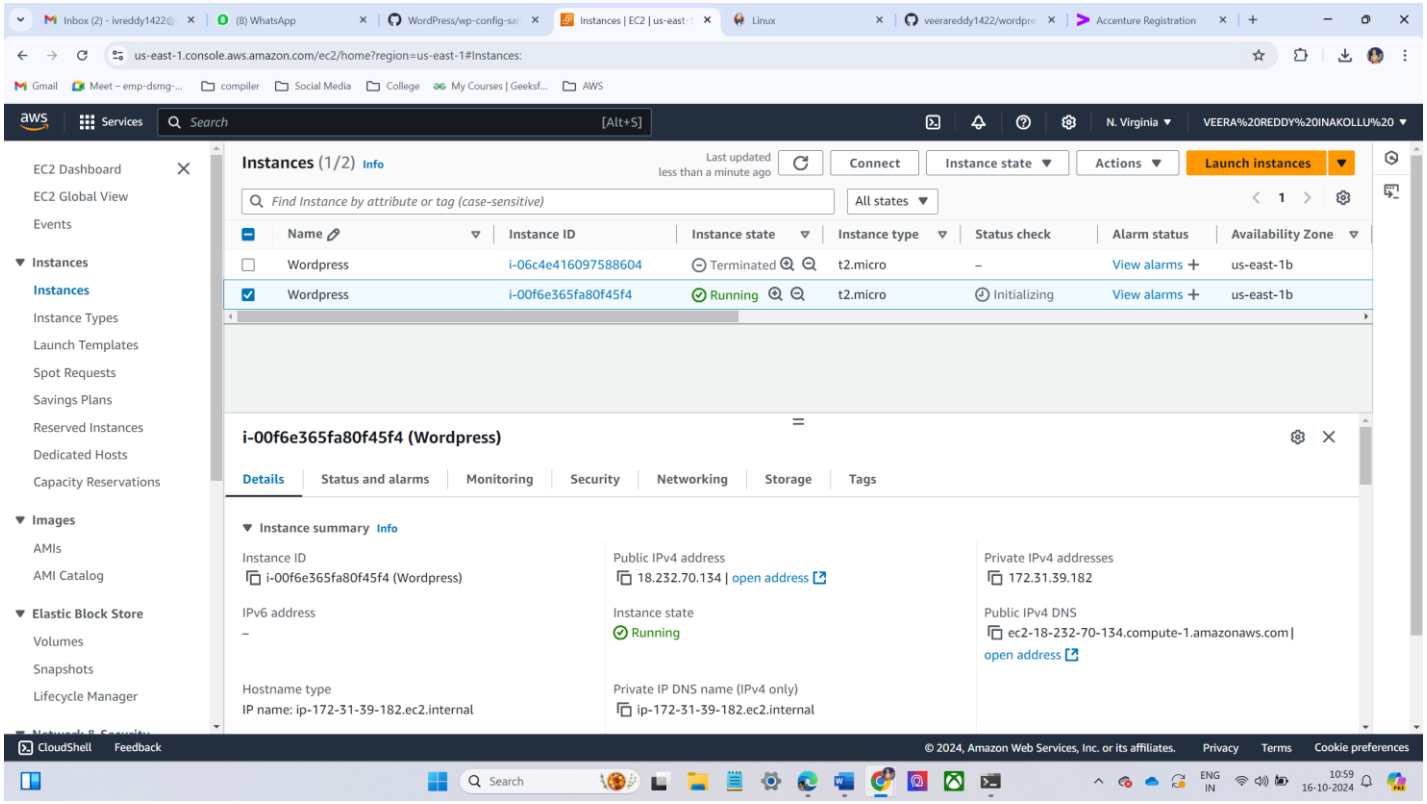
- Successfully deployed WordPress application using docker-compose



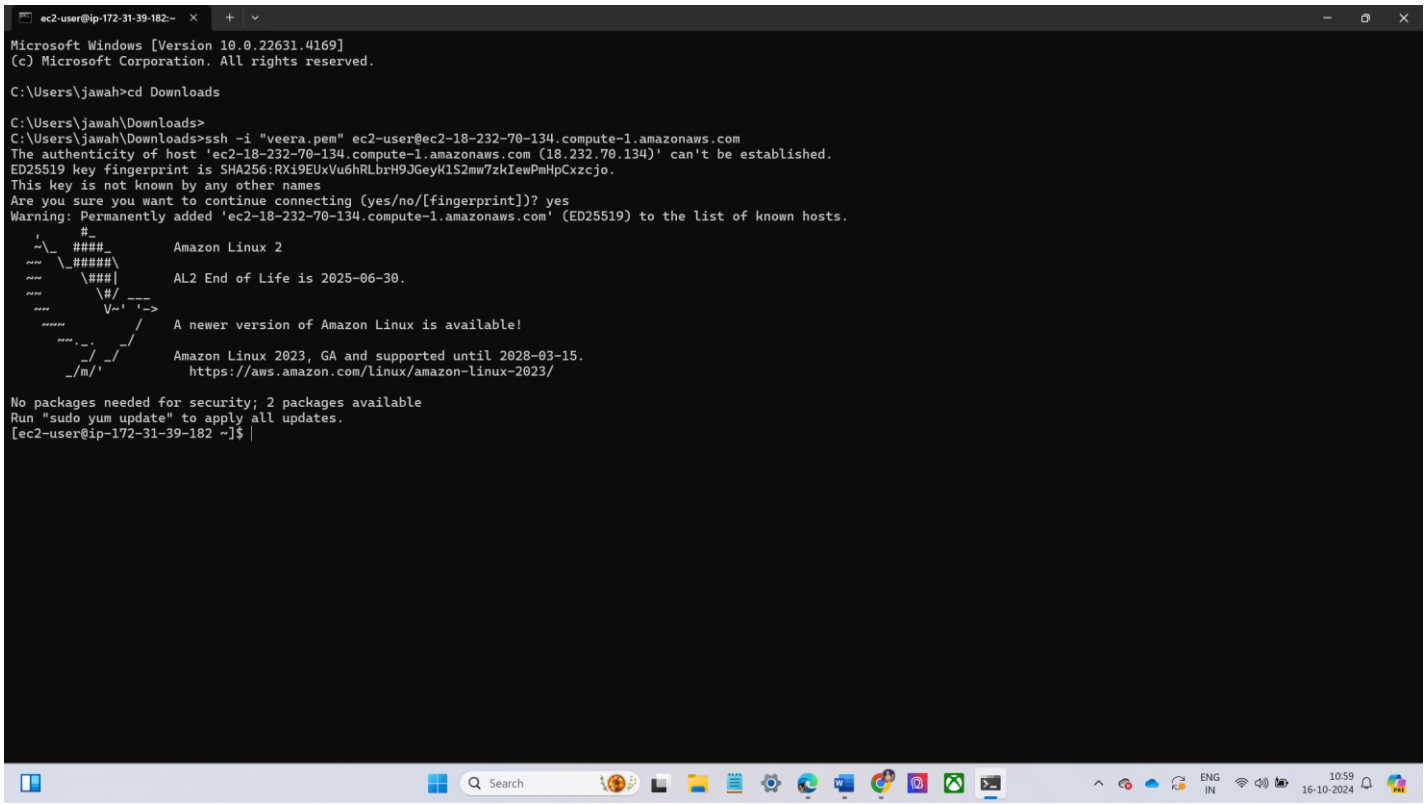

3.93.19.73/wp-admin/
veera
New
Howdy, reddy
Dashboard
Screen Options
Help
Home
Updates
Posts
Media
Pages
Comments
Appearance
Plugins
Users
Tools
Settings
Collapse menu
Welcome to WordPress!
Learn more about the 6.6.2 version.
Dismiss
Author rich content with blocks and patterns
Block patterns are pre-configured 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
No information yet...
Site health checks will automatically run periodically to gather information about your site. You can also visit the Site Health
Quick Draft
Title
30°C
Sunny
Search
ENG
IN
17:40
08-10-2024

Deploy WordPress web application by using git and jenkins?

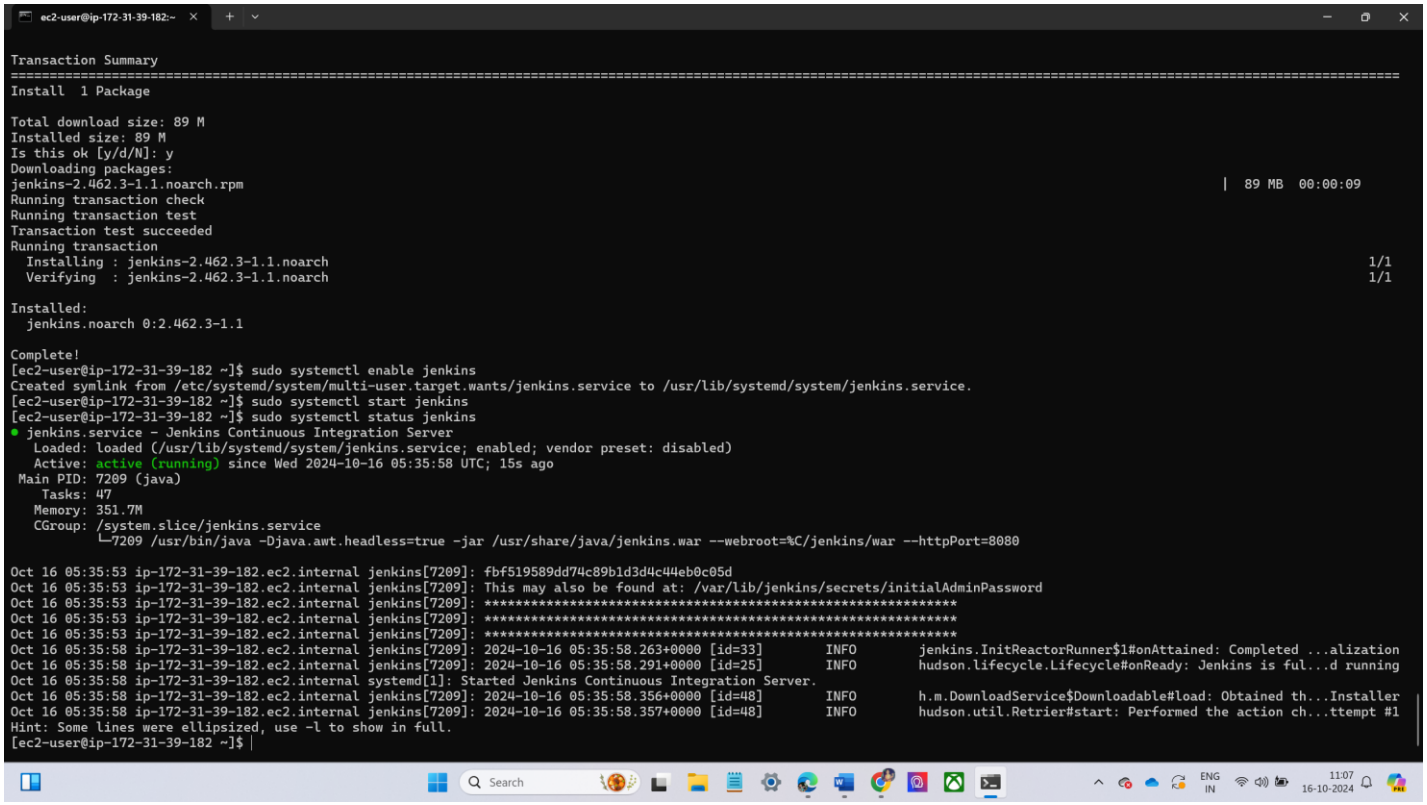
- Login AWS Account and launch EC2 instance



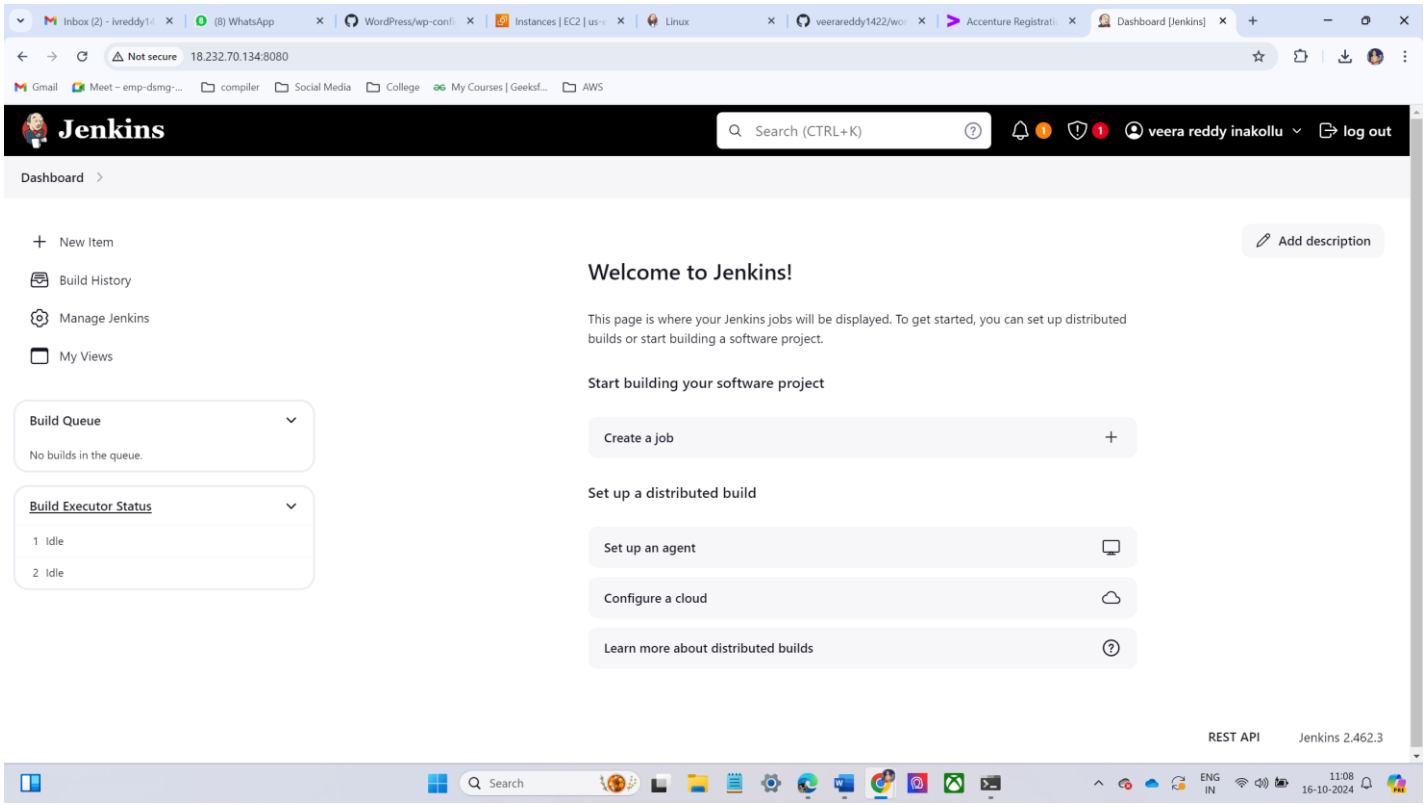
- Connect EC2 instance to the command prompt



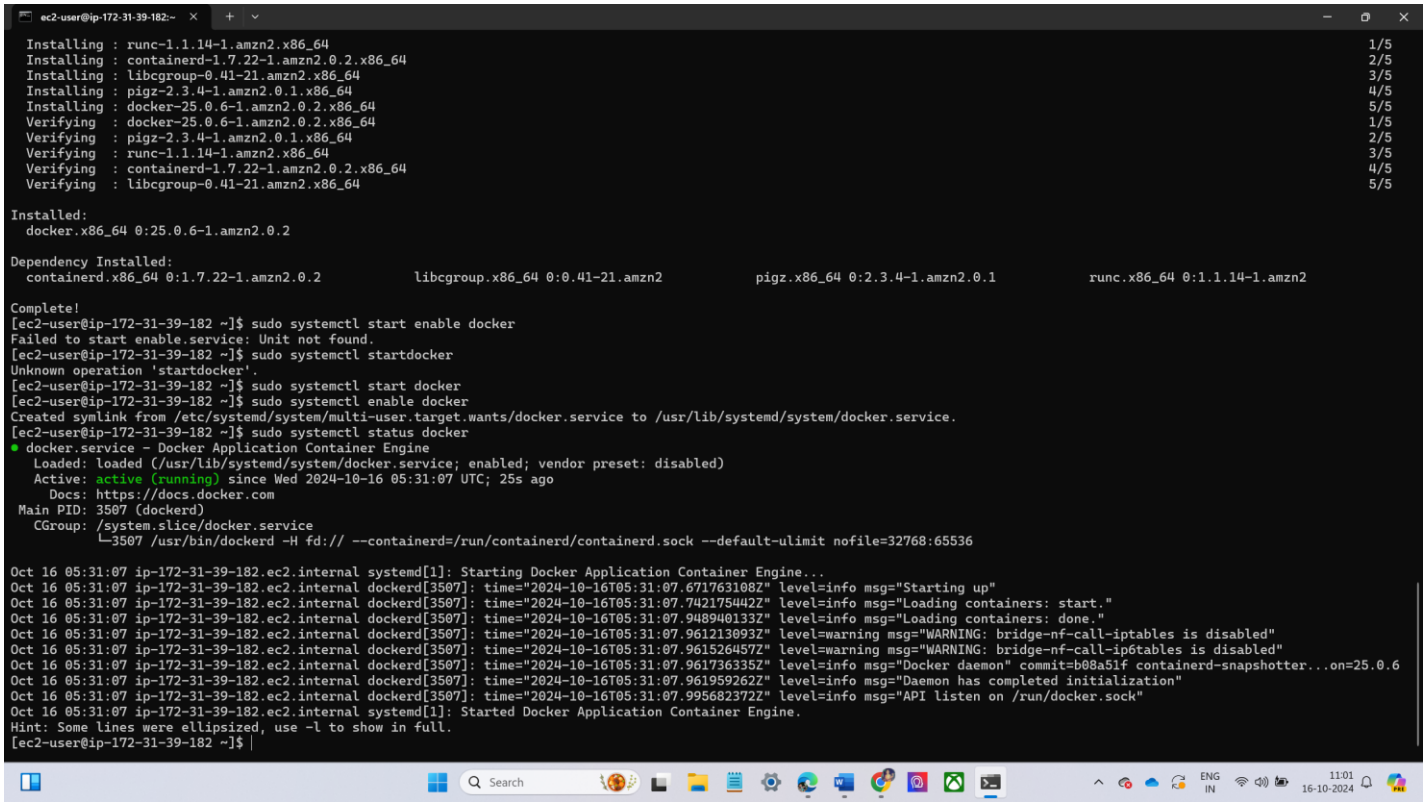
- Install Jenkins and also set up the Jenkins



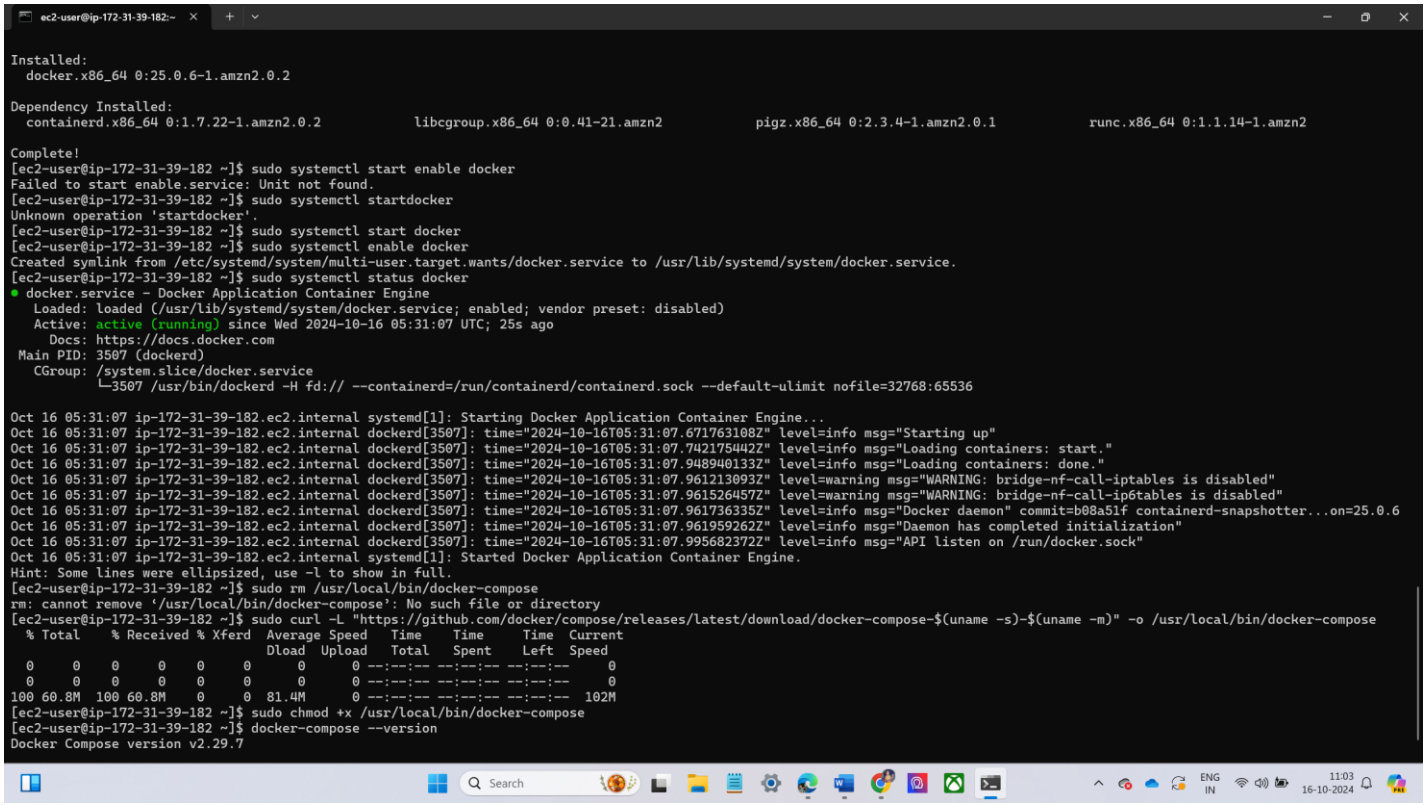
- Launch the Jenkins



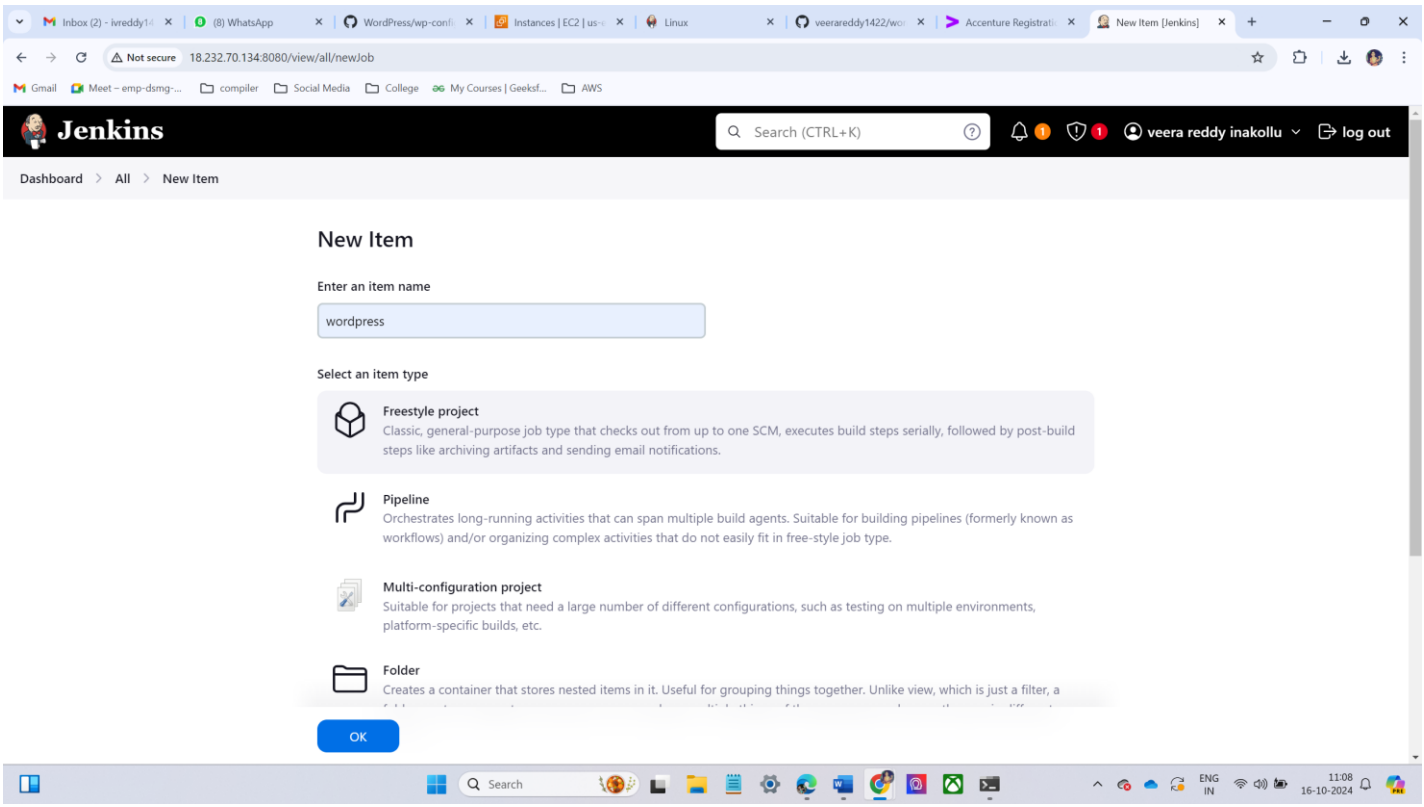
- Install docker and set up the docker



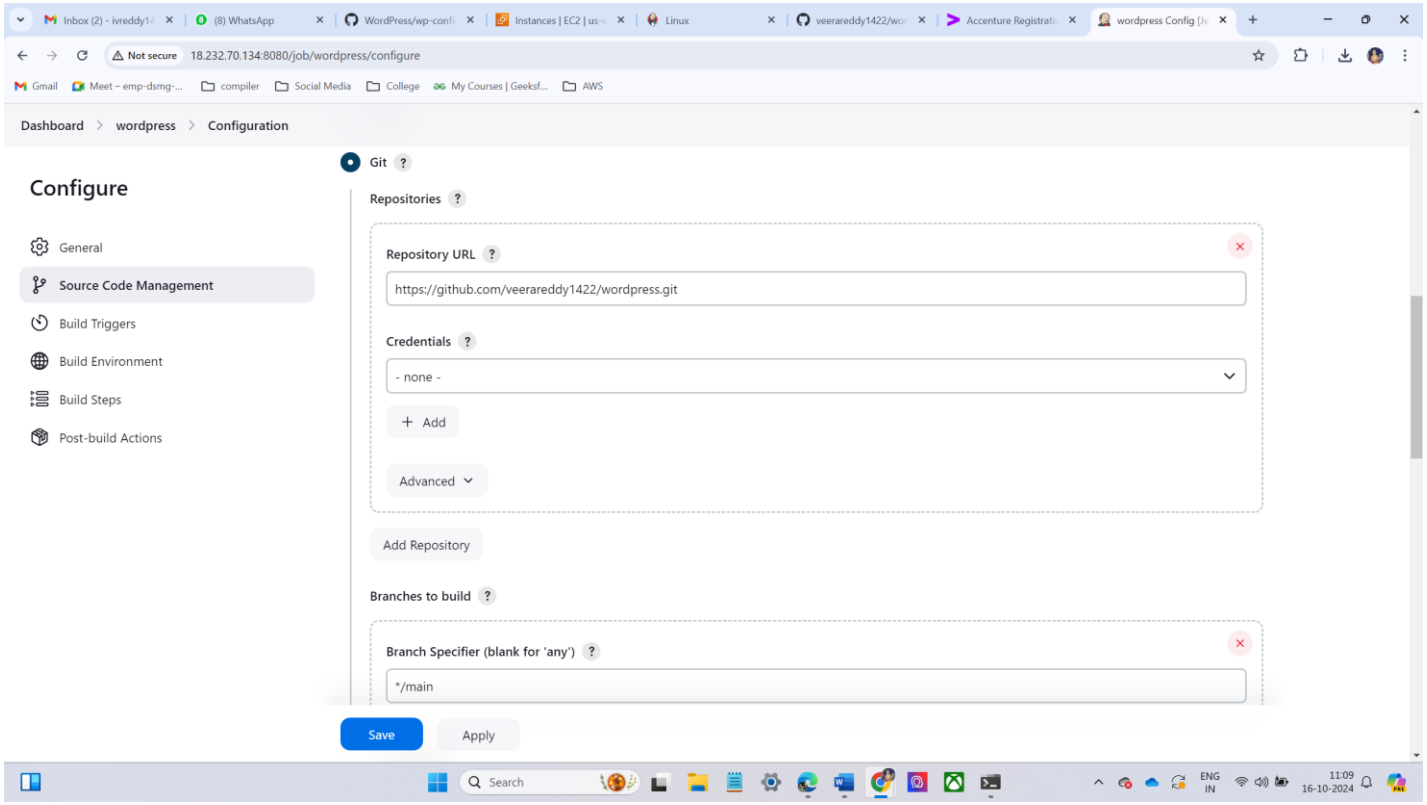
- Install the docker-compose and enable the permissions



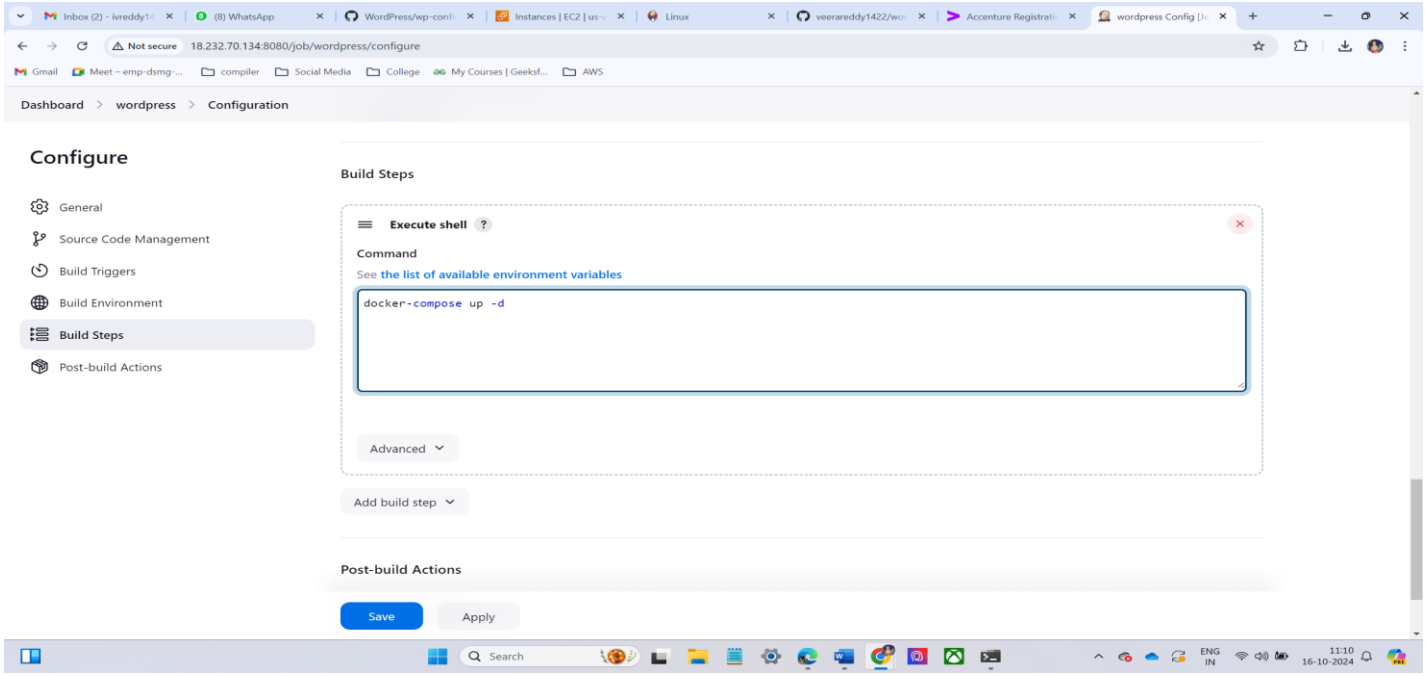
- Create a New Freestyle Project in Jenkins
- Log in to Jenkins.
- Click on **"New Item"**.
- Enter a name for your project (e.g., WordPress-Deployment).
- Select **"Freestyle project"** and click **OK**.



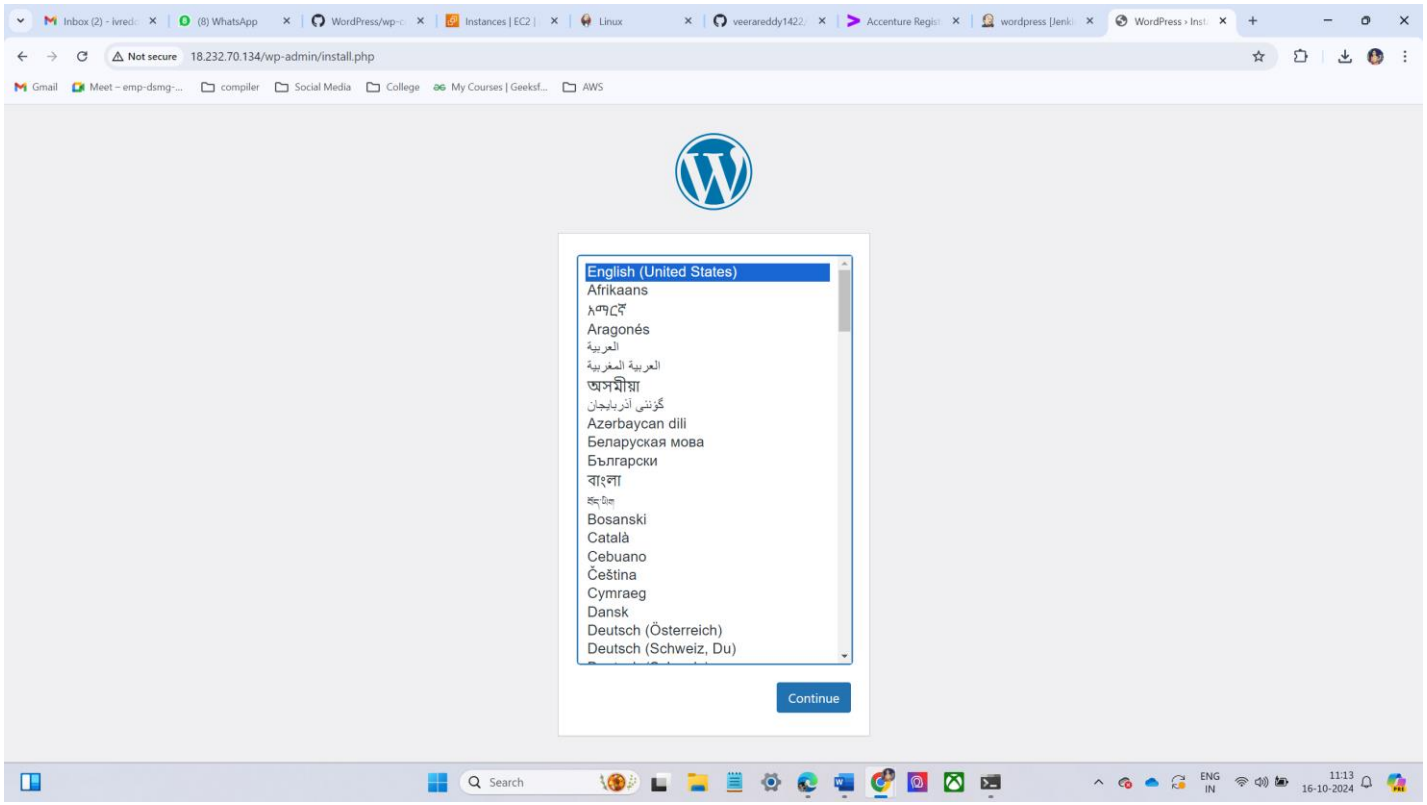
- Configure Source Code Management (SCM)
- In the project configuration page, scroll down to the **"Source Code Management"** section.
- Select **"Git"**.
- Enter the **Repository URL** of your GitHub repository



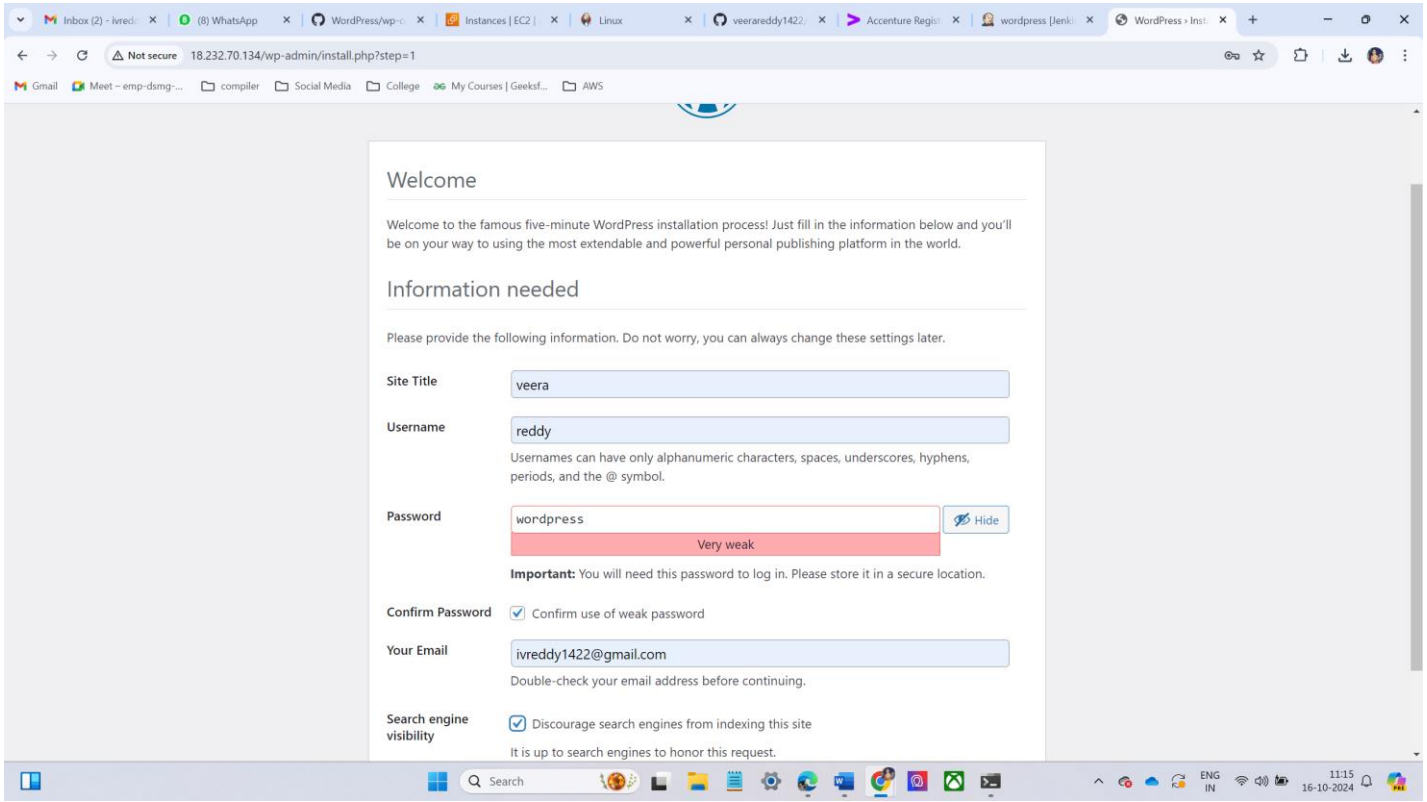
- Under build steps write script on execute shell script



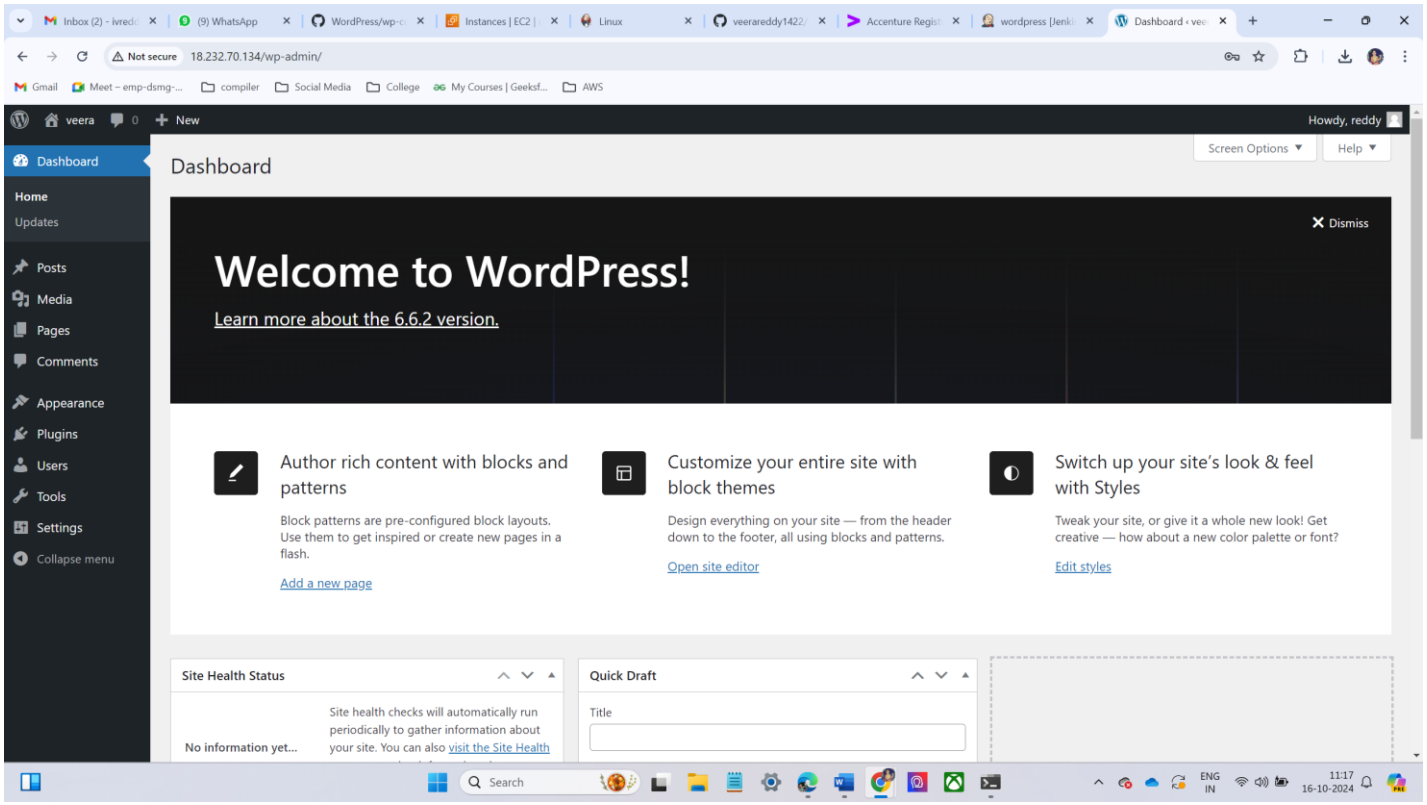
- Save and build the job
- Now browse the EC2 instace ipv4 address along with port 80 it will shows like.



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

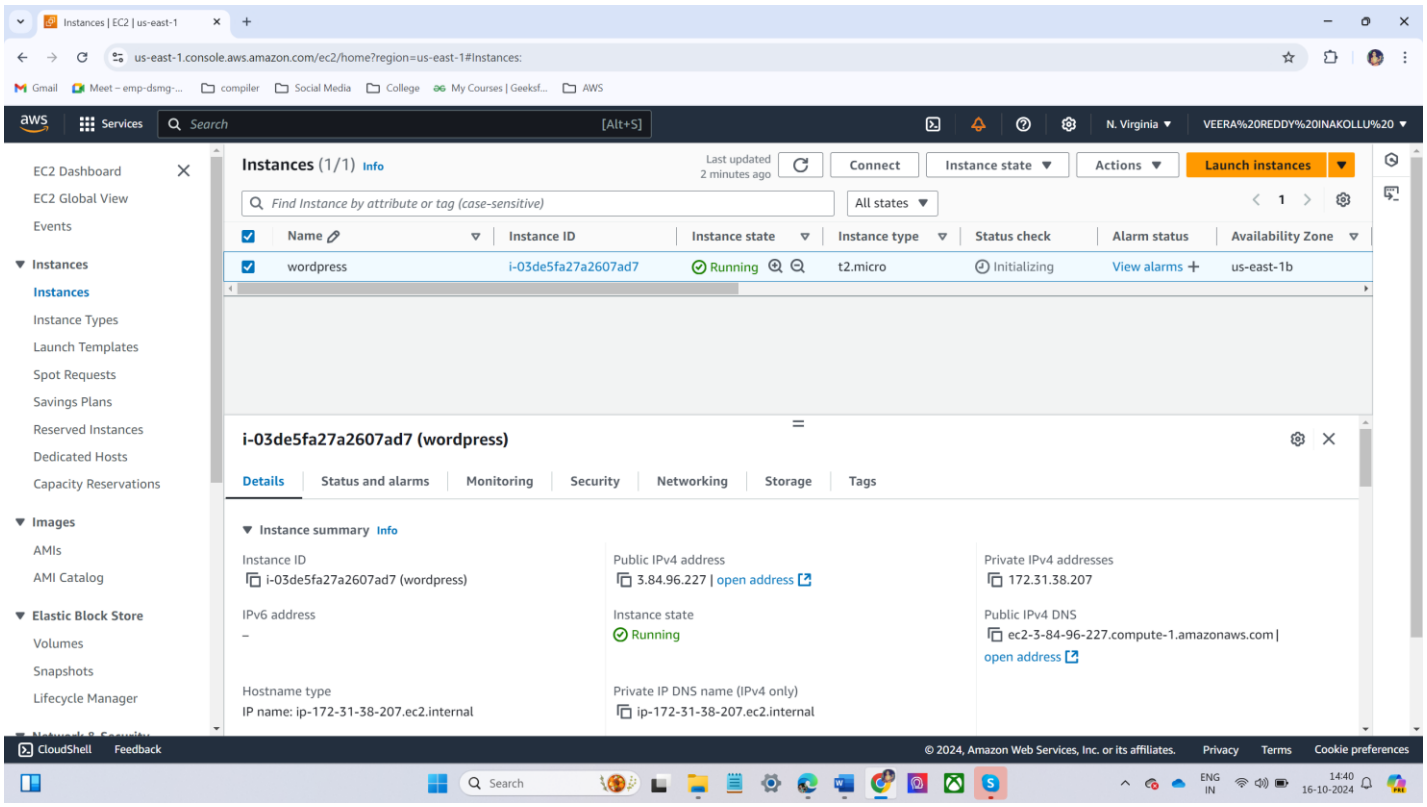


- Succssfuly host the wordpress application using git and Jenkins

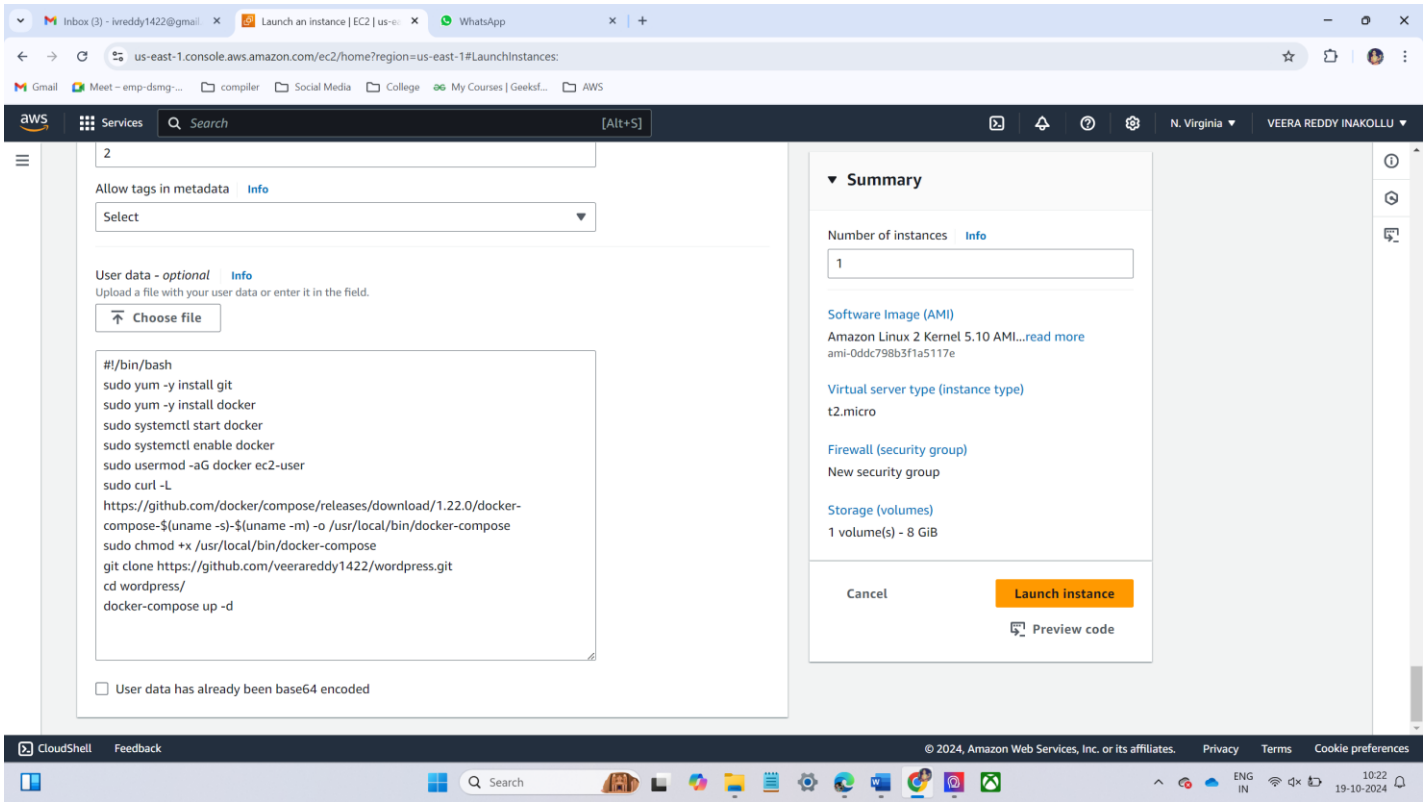


Deploy WordPress web application by using userdata of EC2 instance

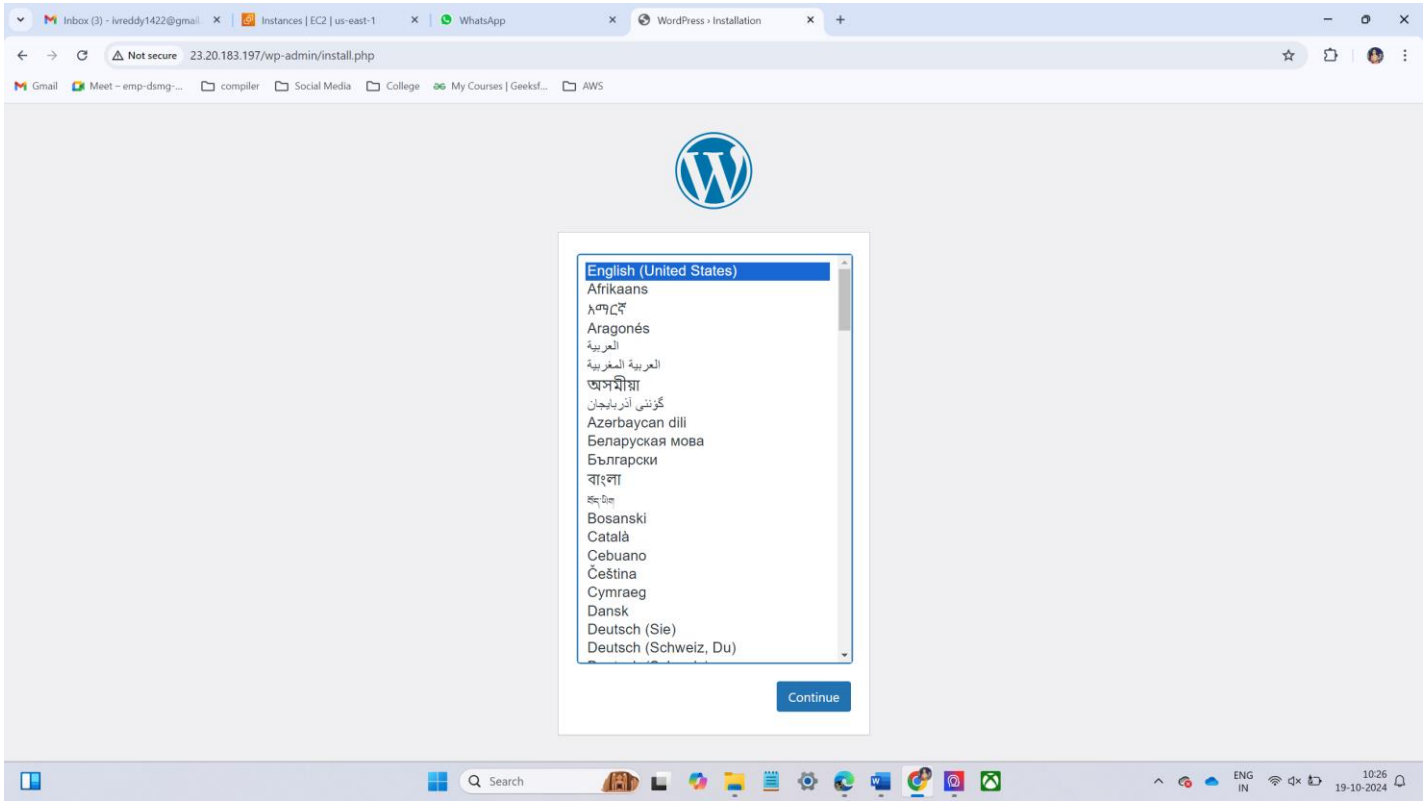
- Login to AWS Account and Launch EC2 instance and attach security groups port number 20 and 80.



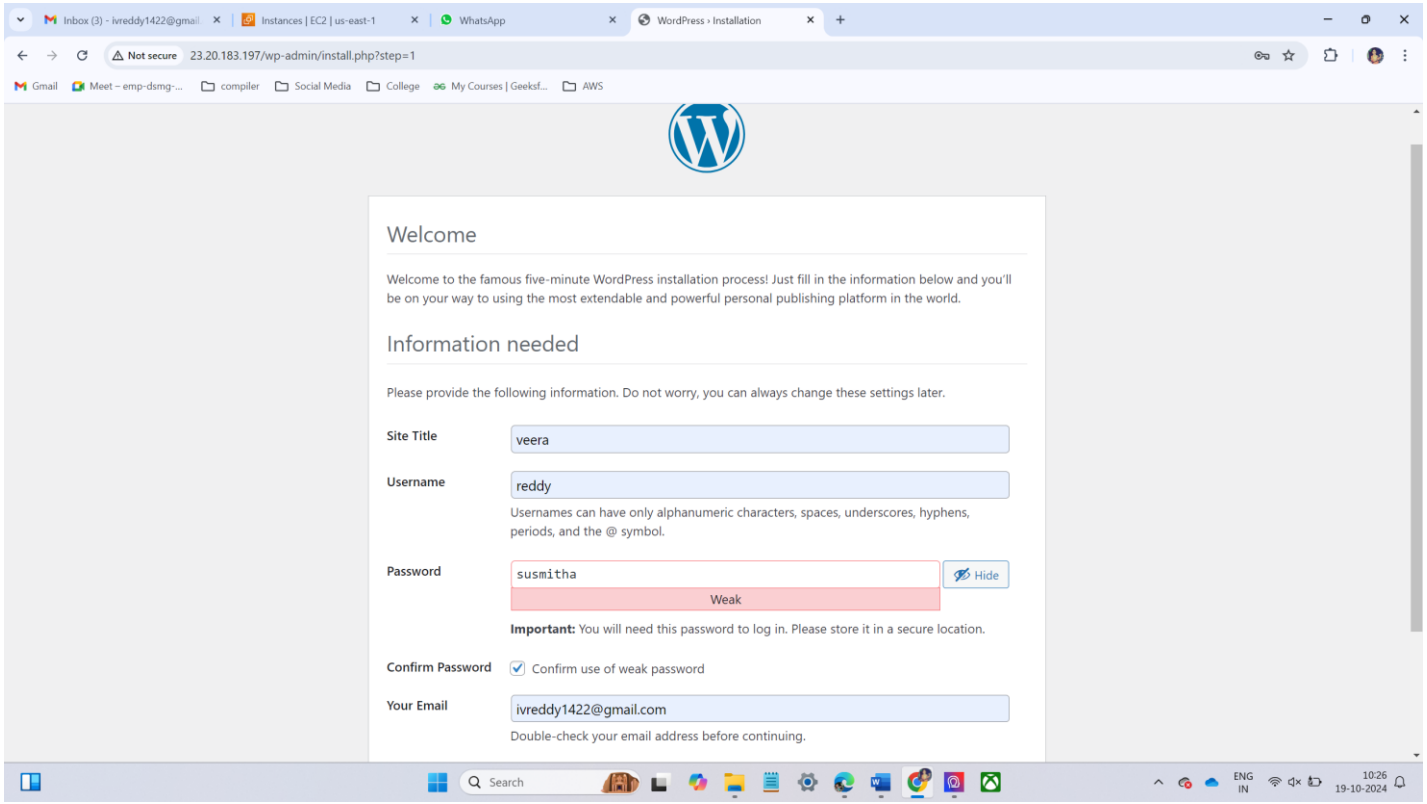
- Under **Advanced details** write user data



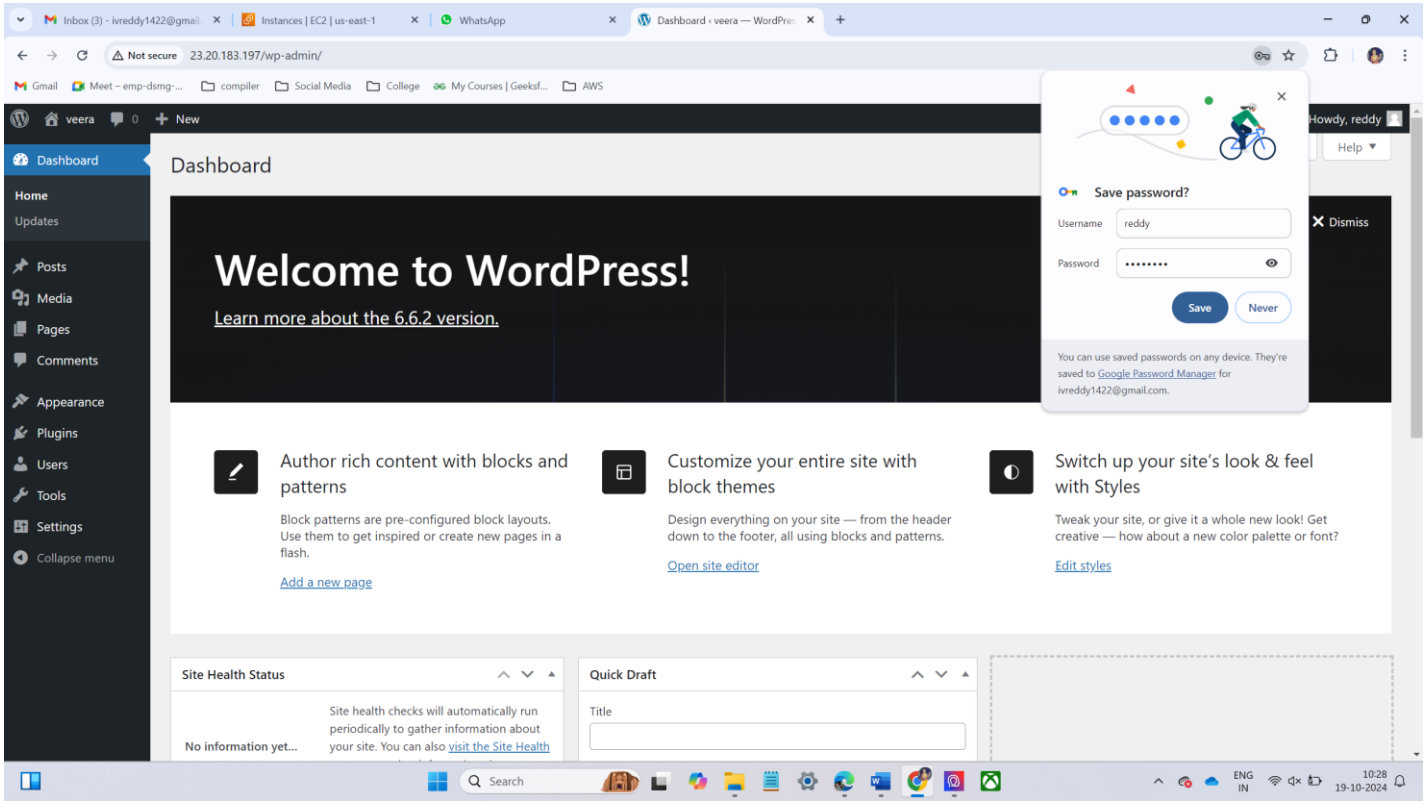
- Now browse the instance ipv4 address in google



- Create WordPress account

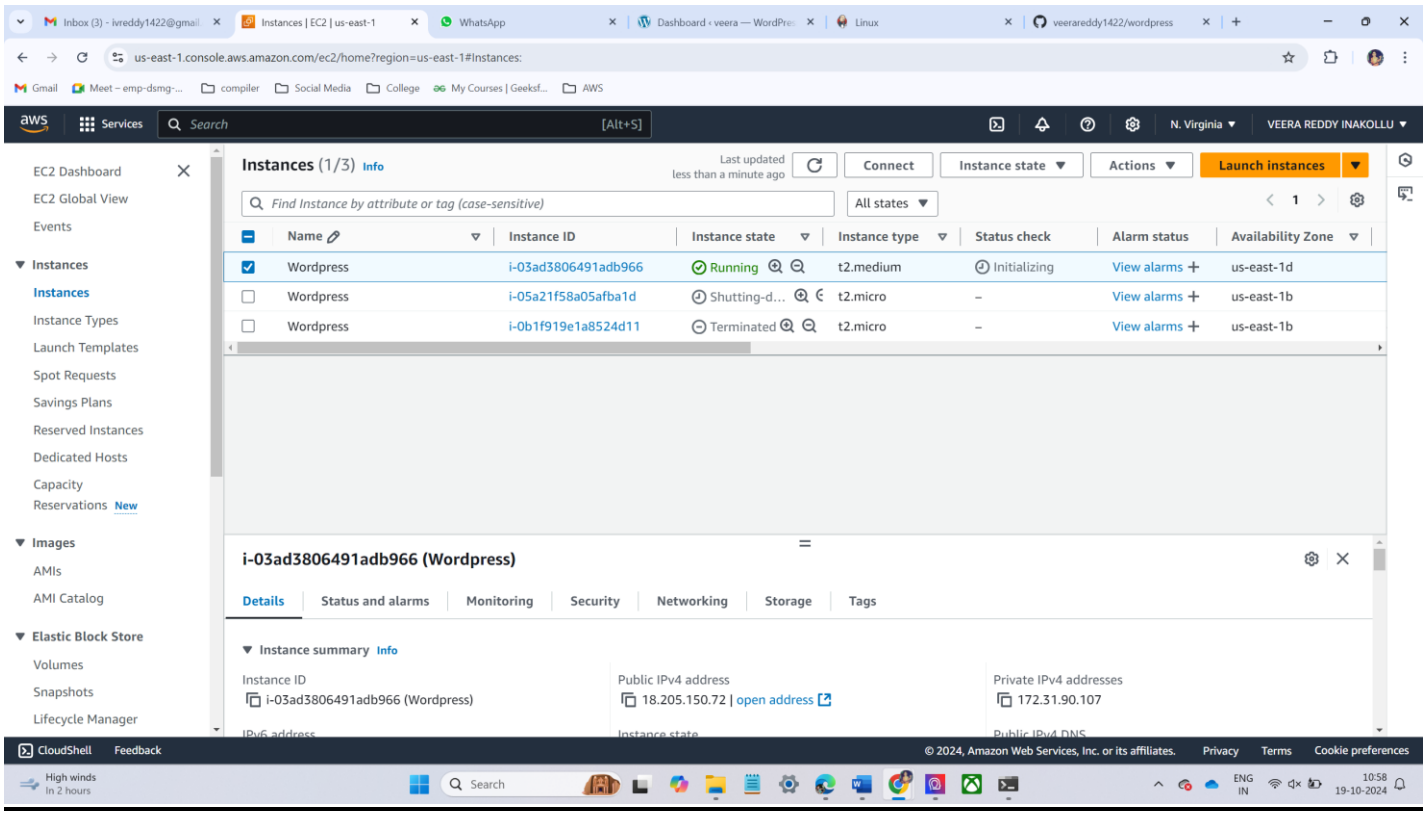


- Create the WordPress account and login using credentials

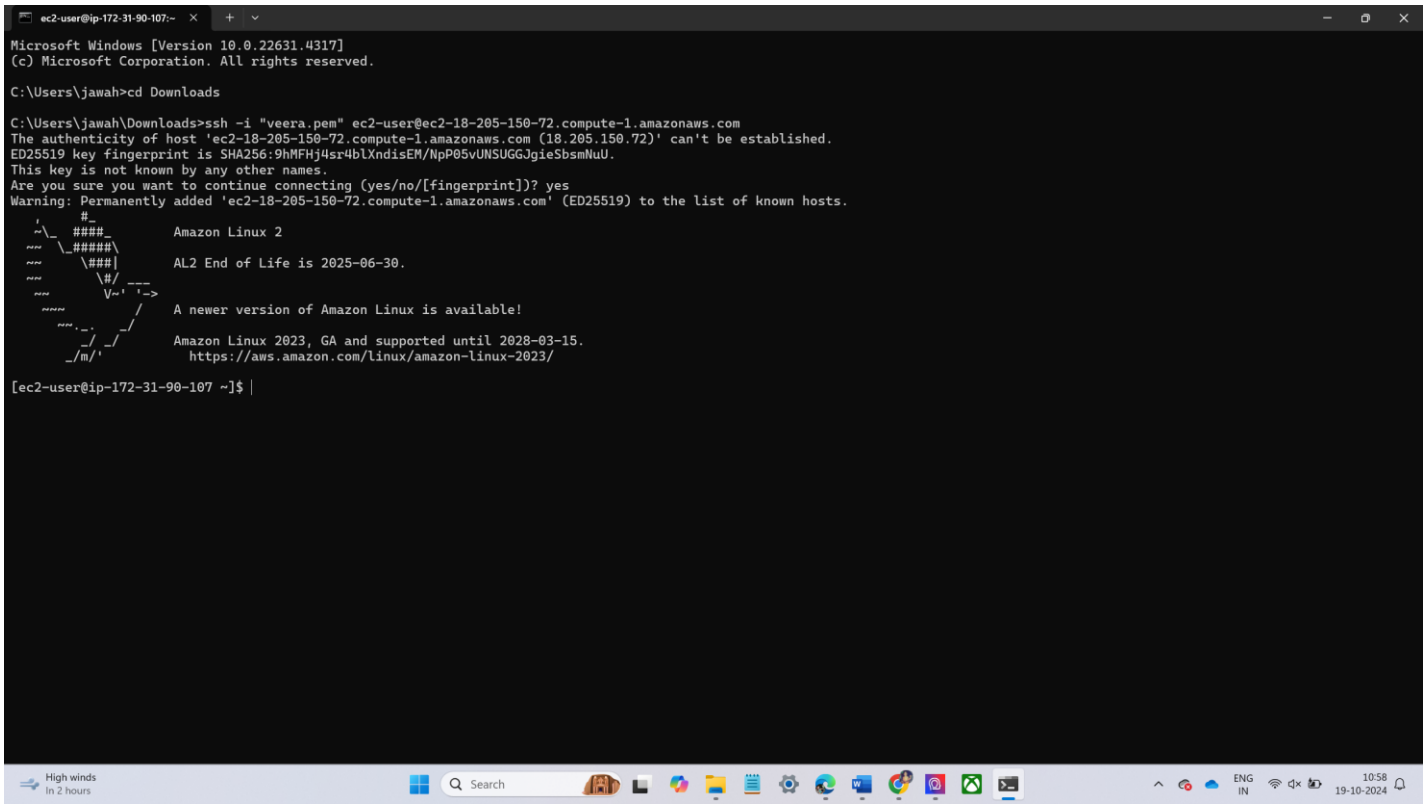


Deploy WordPress web application by using git and jenkins execute shell (bash script)

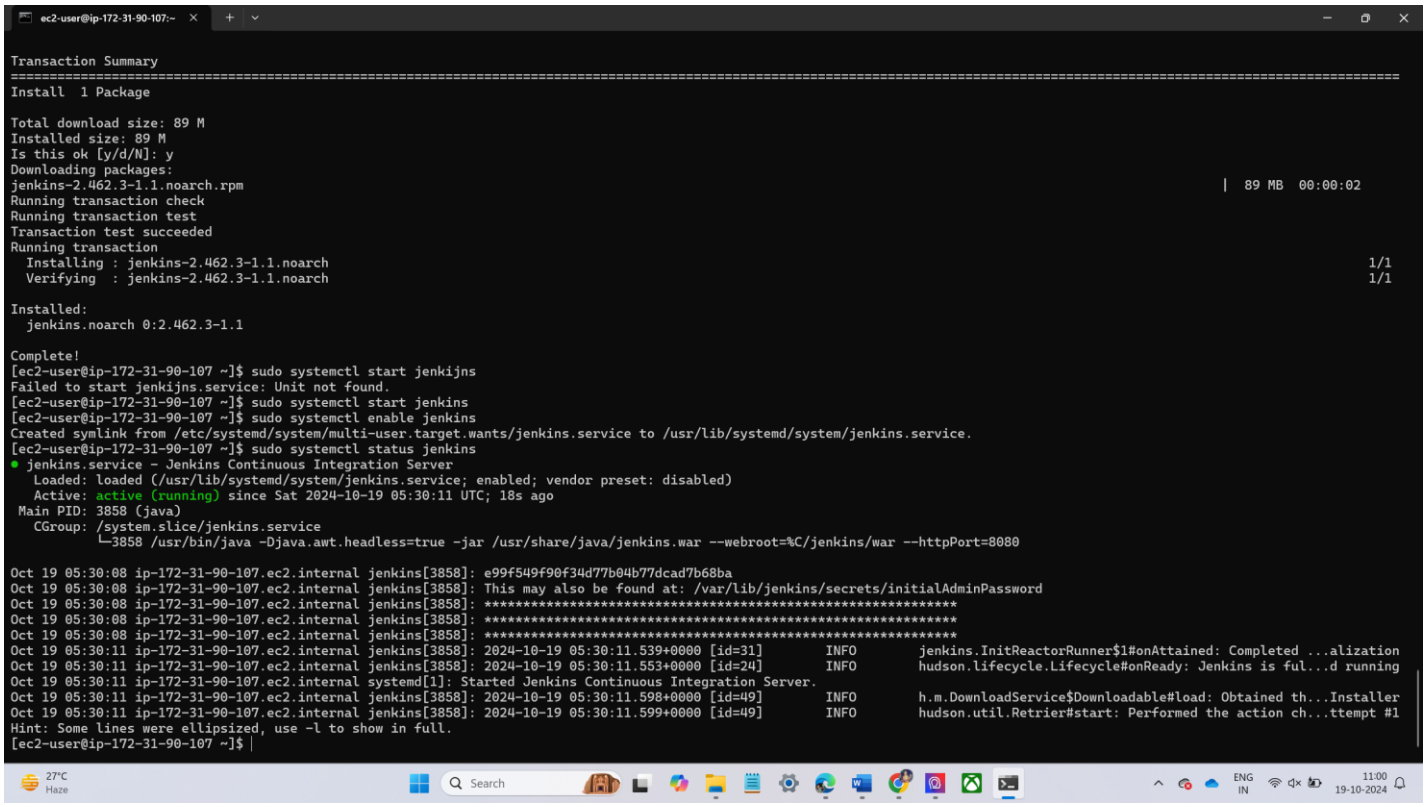
- Login to aws account and launch ec2 instance



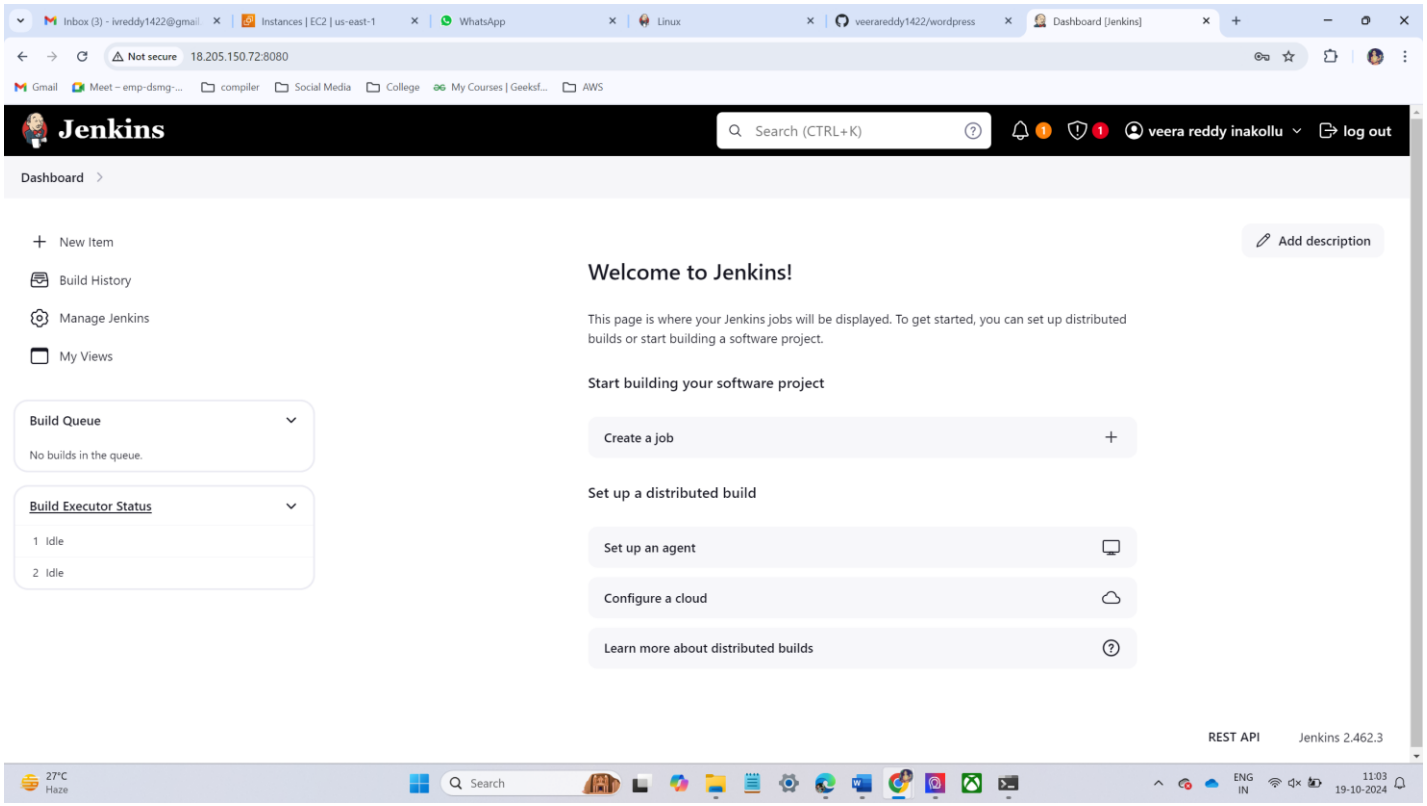
- Connect ec2 instance to the gitbash.



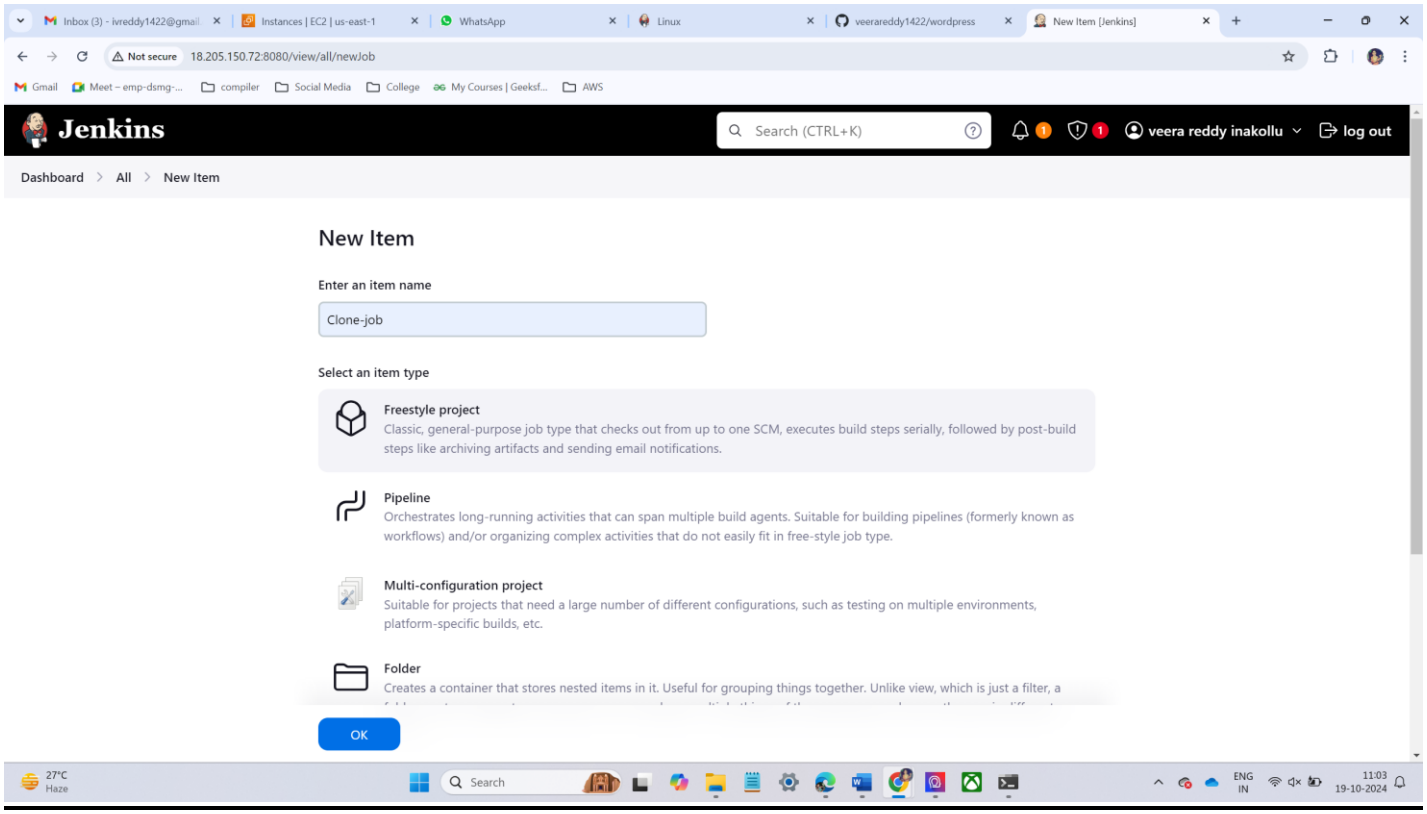
- Now install Jenkins and set up Jenkins



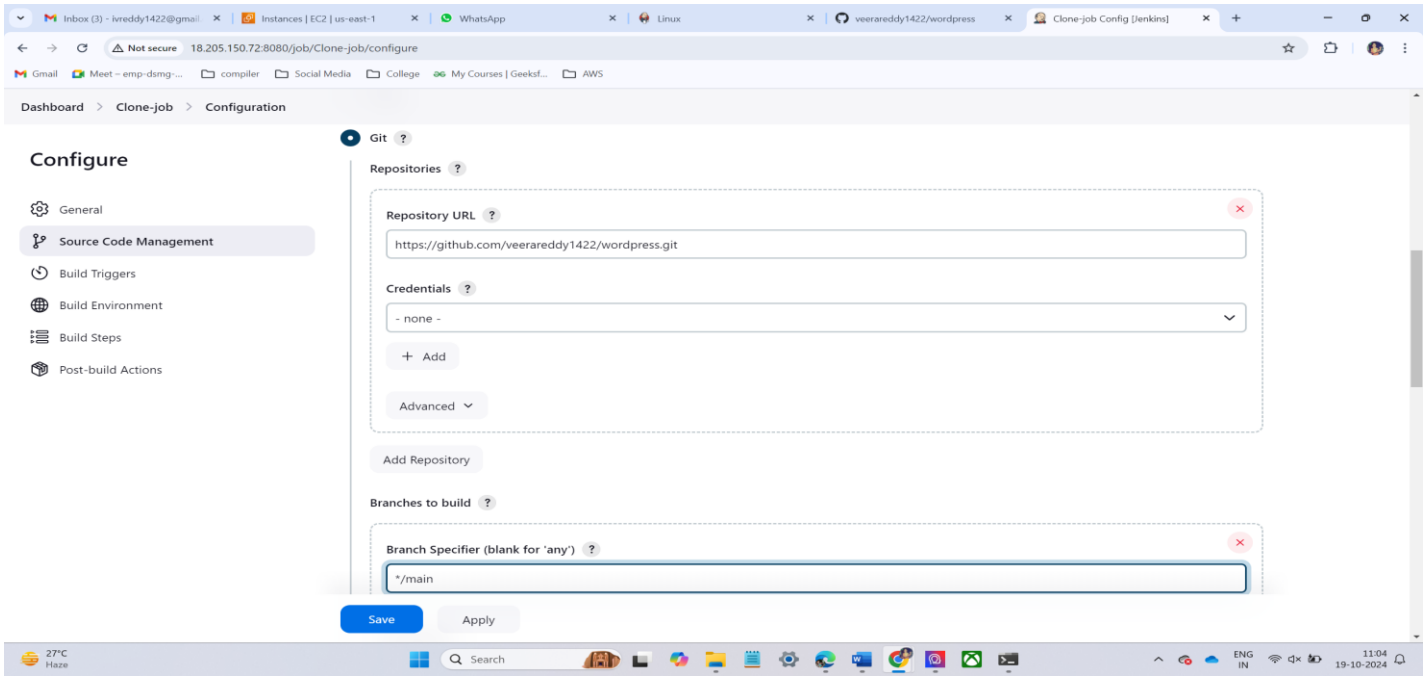
- Now launch the Jenkins sever using ipv4 address along with port number 8080.



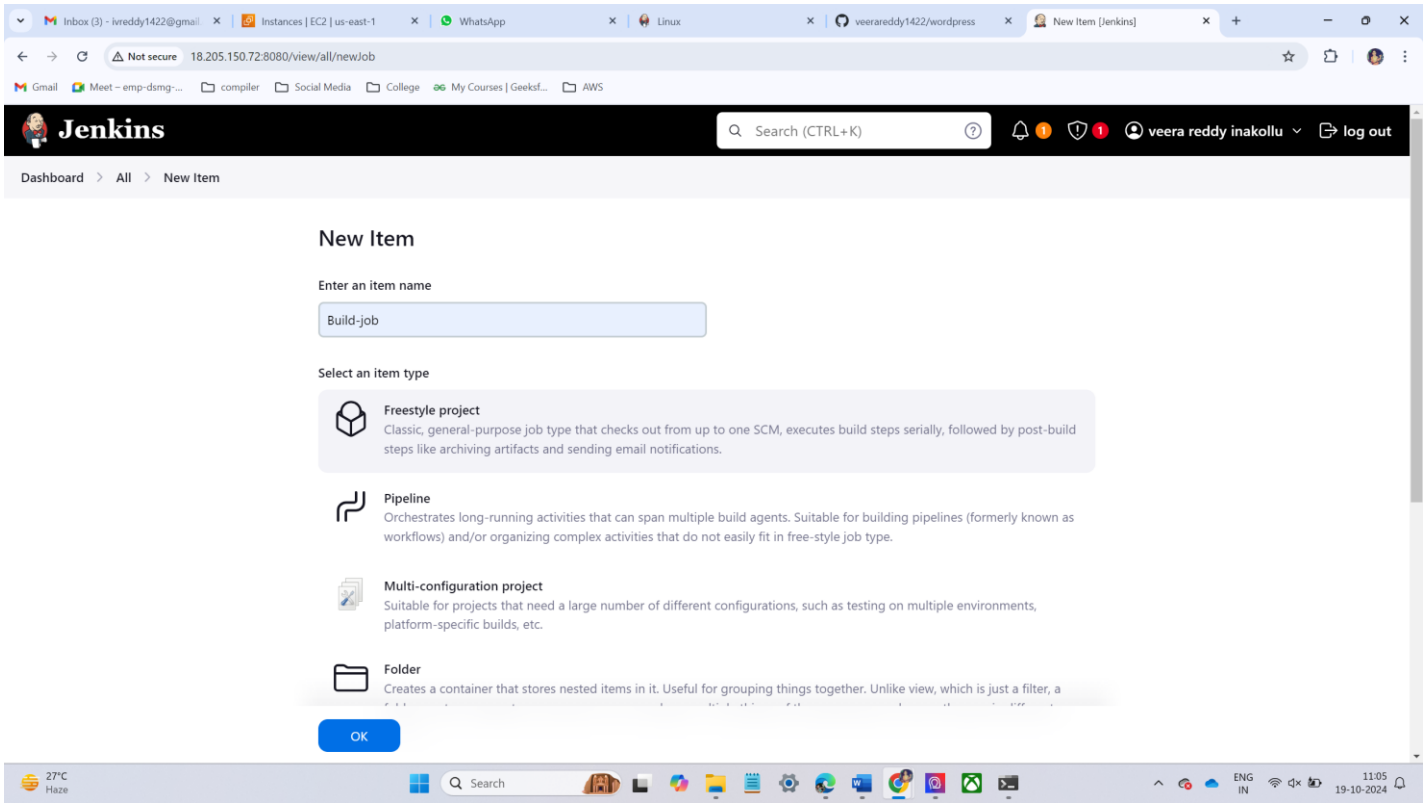
- Create job for clone the code
- Click on **"New Item"**.
- Enter a name for your project (e.g., WordPress-Deployment).
- Select **"Freestyle project"** and click **OK**.



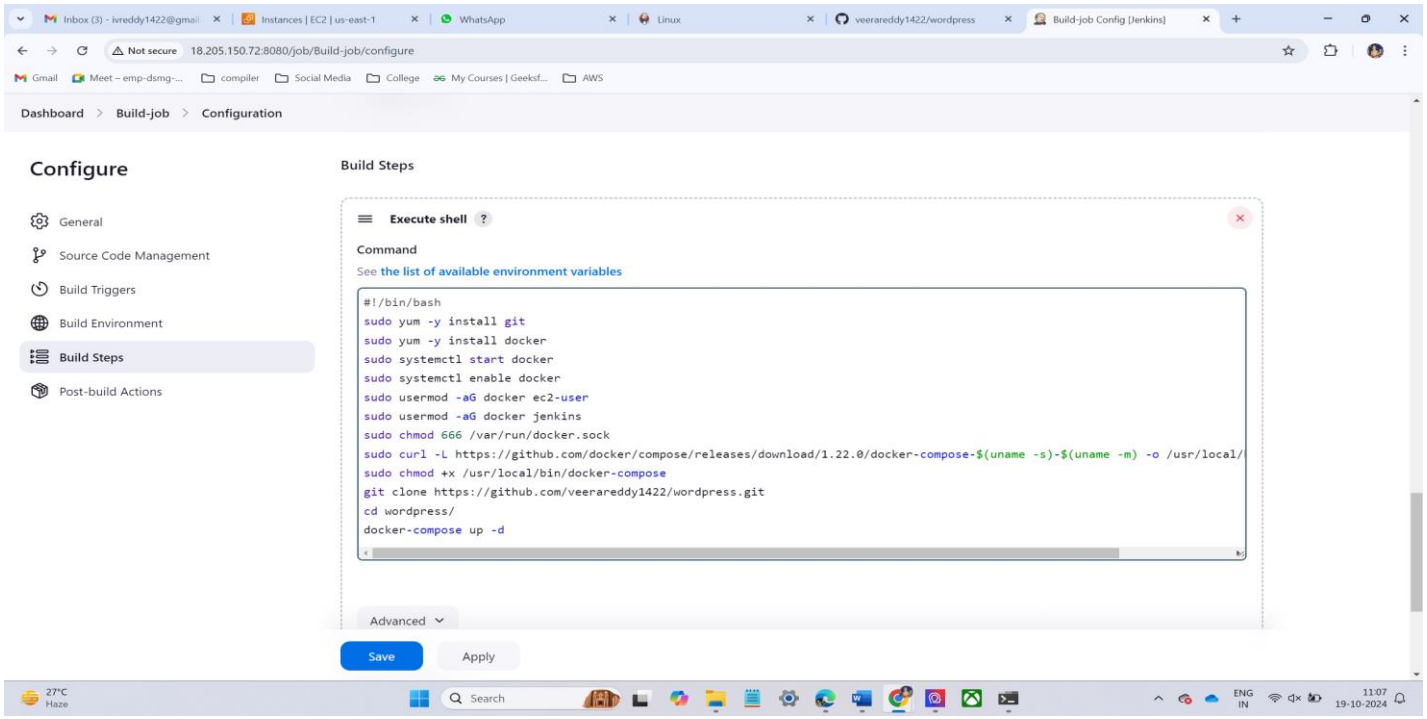
- Configure Source Code Management (SCM)
- In the project configuration page, scroll down to the **"Source Code Management"** section.
- Select **"Git"**.
- Enter the **Repository URL** of your GitHub repository



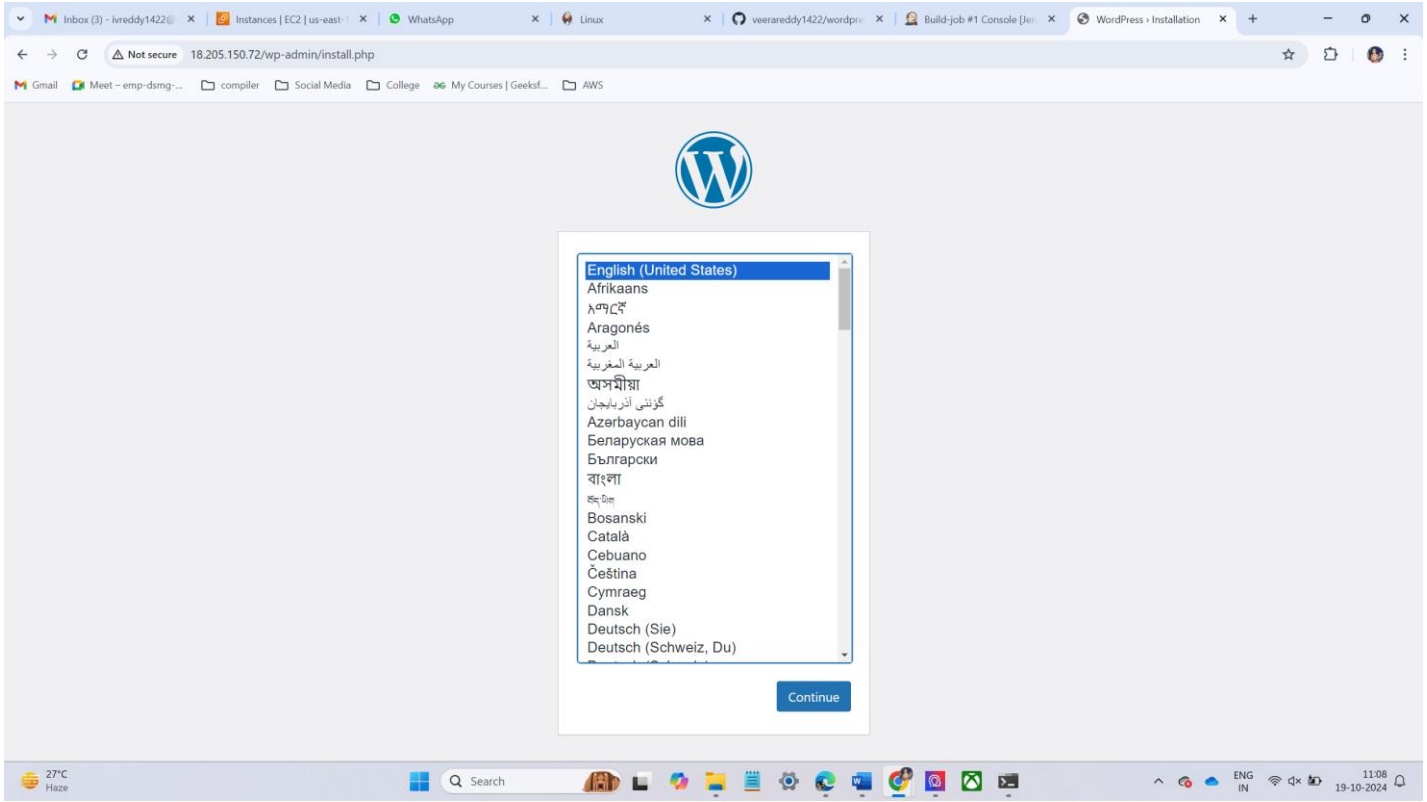
- Create job for build the code
- Click on **"New Item"**.
- Enter a name for your project (e.g., WordPress-Deployment).
- Select **"Freestyle project"** and click **OK**.



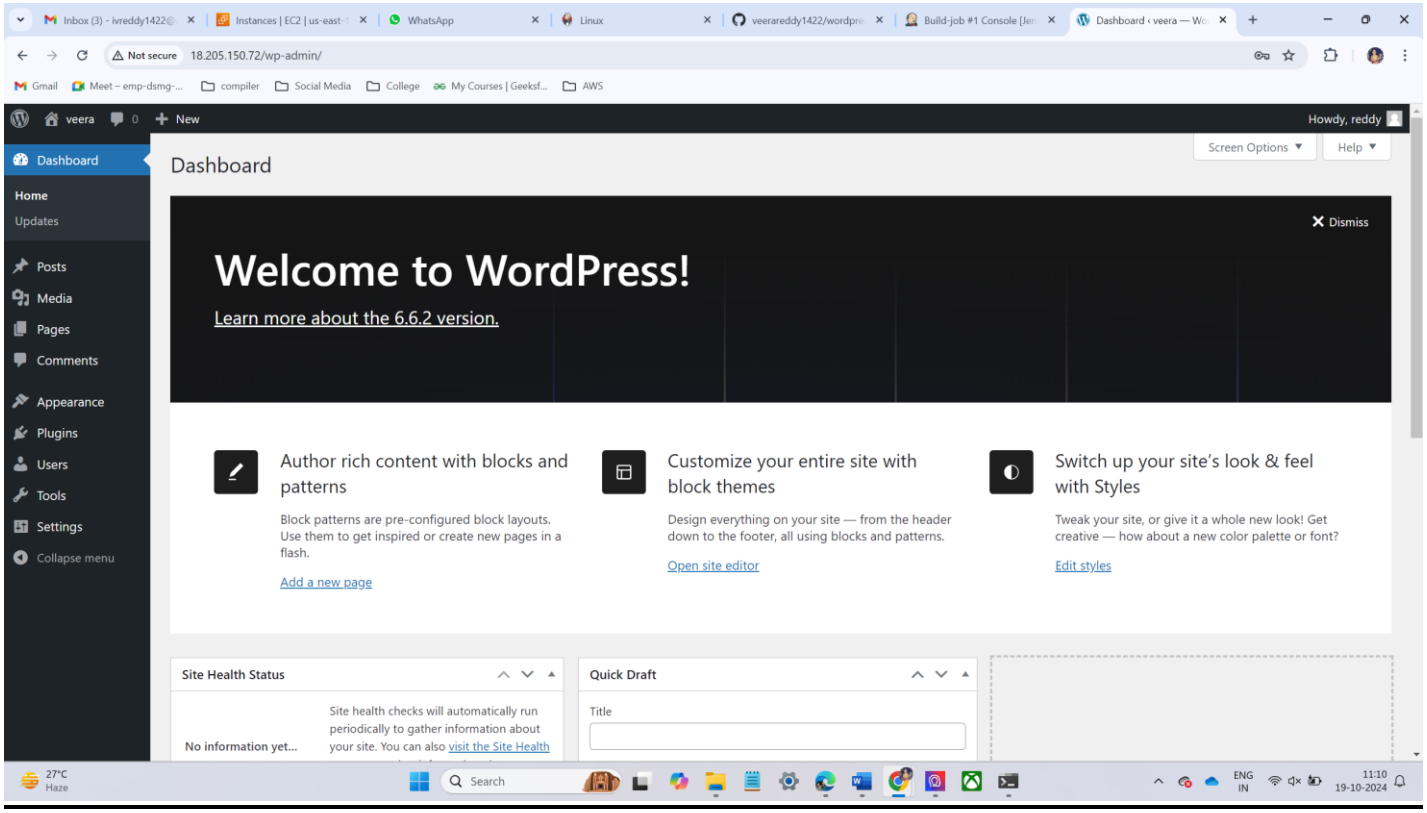
- Configure Build steps
- In the project configuration page, scroll down to the **"Build steps"** section.
- Select **"Execute shell"**.
- Write the **bash script**



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

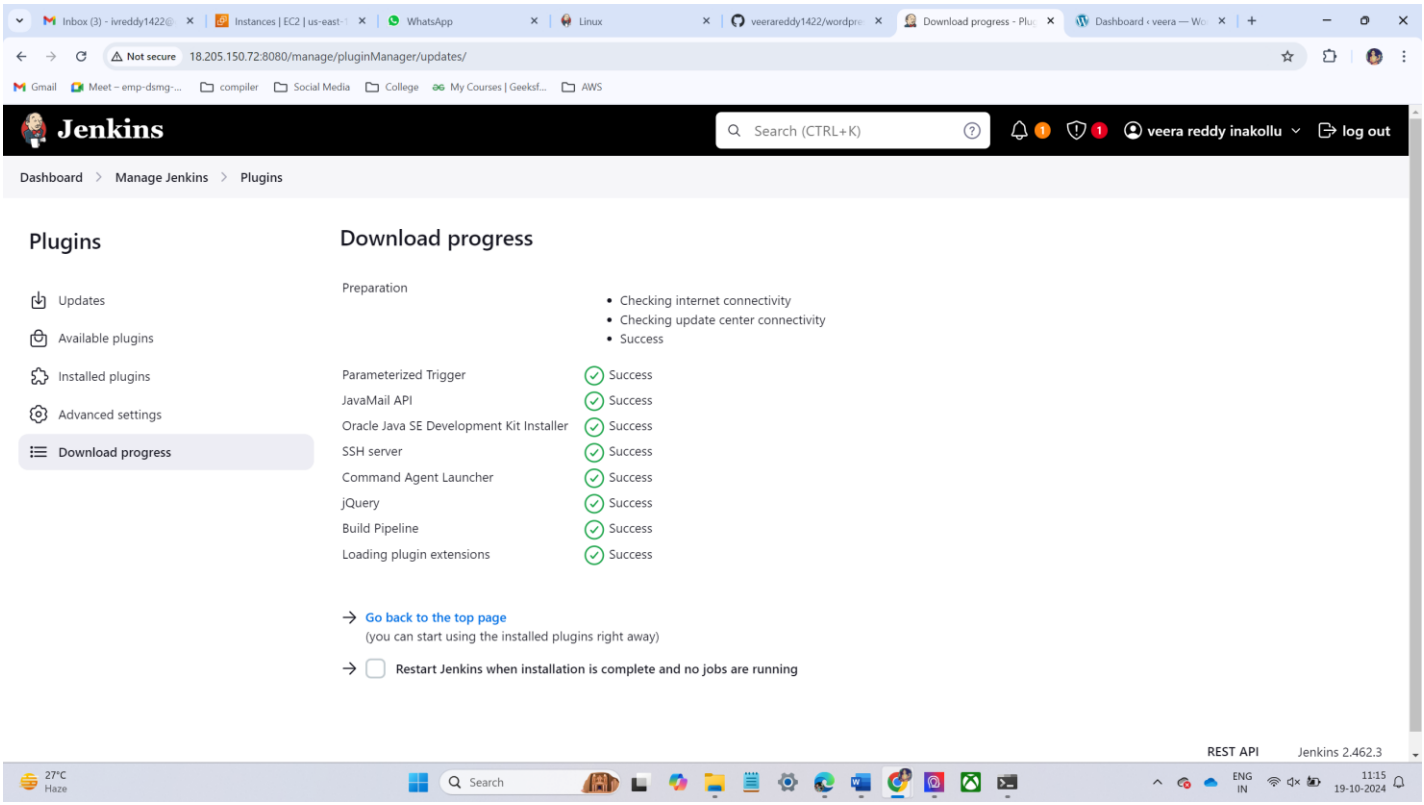


- WordPress application successfully deployed using Jenkins bash script

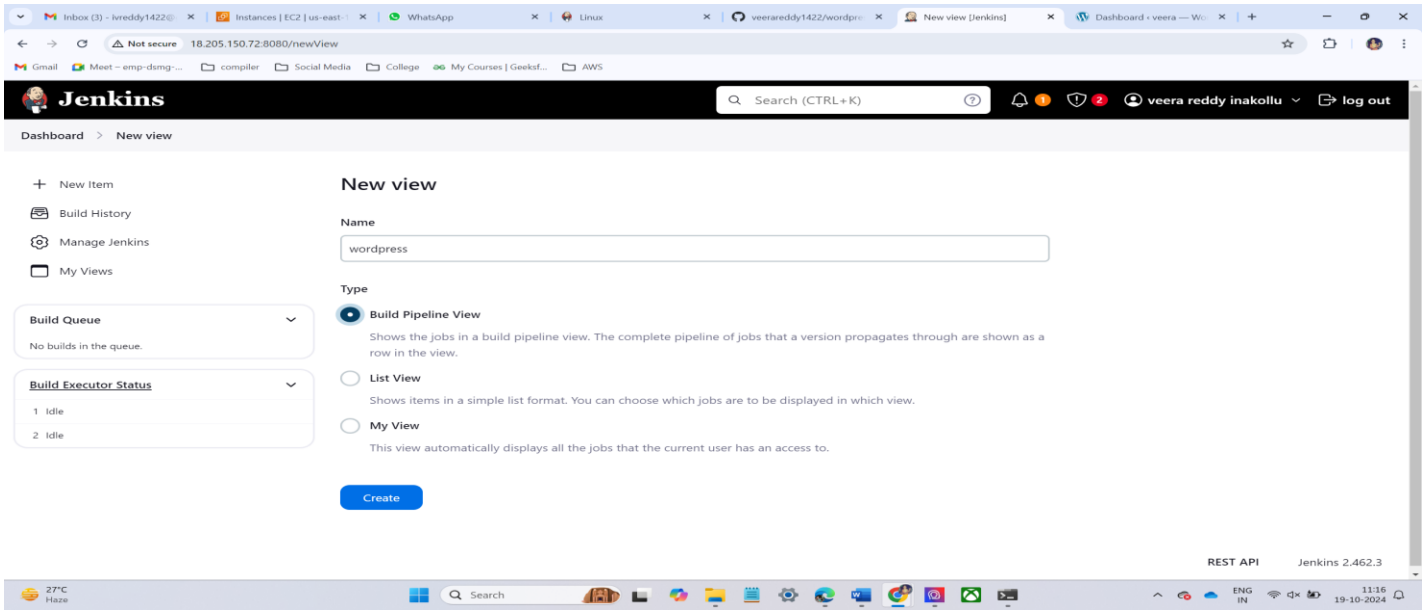


Deploy WordPress web application by using git and jenkins execute shell (bash script) create jenkins pipeline add build periodically and poll scm to initial job of pipeline and check the changes happened or not which are made in github repo?

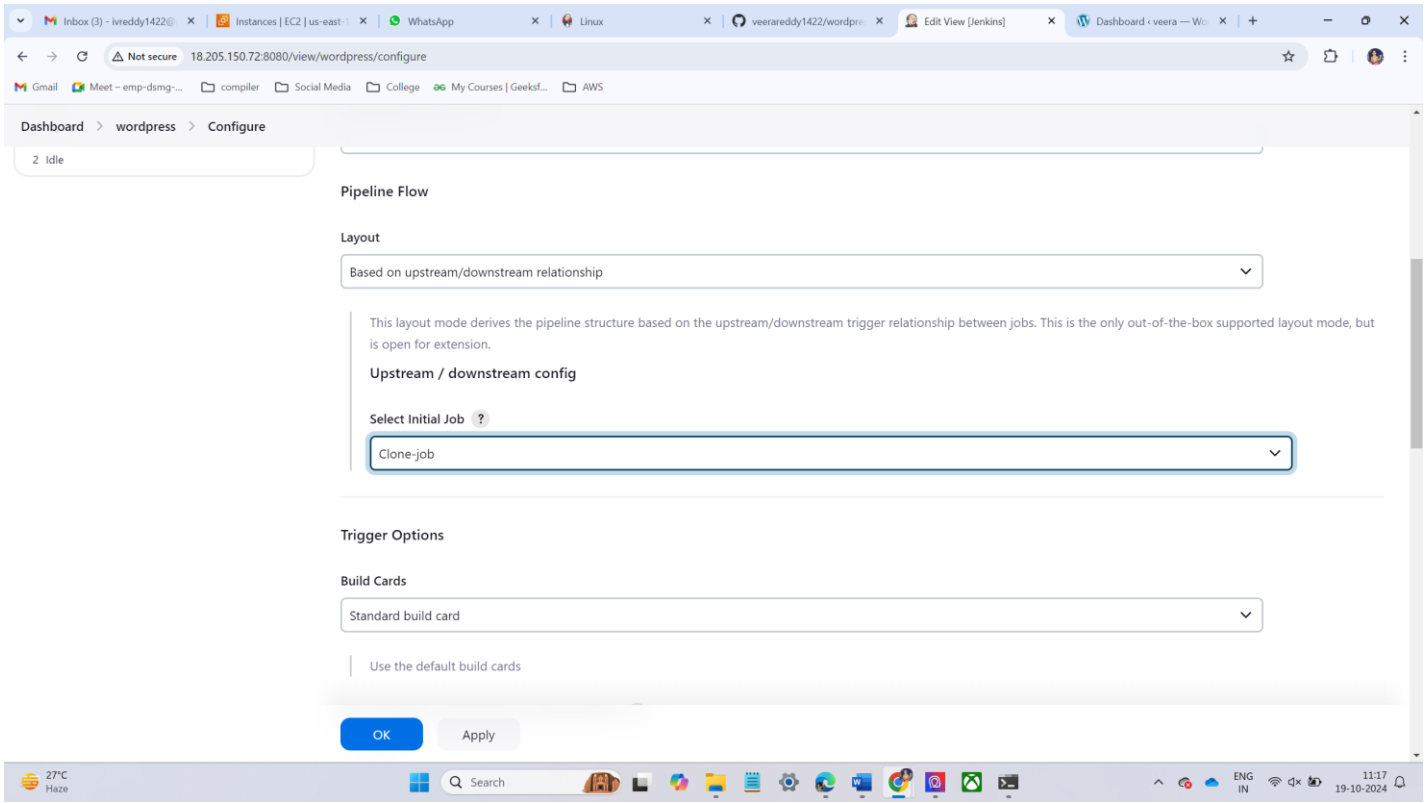
- Create pipeline for WordPress application for continue in the previous method
- Install Jenkins pipeline plugin



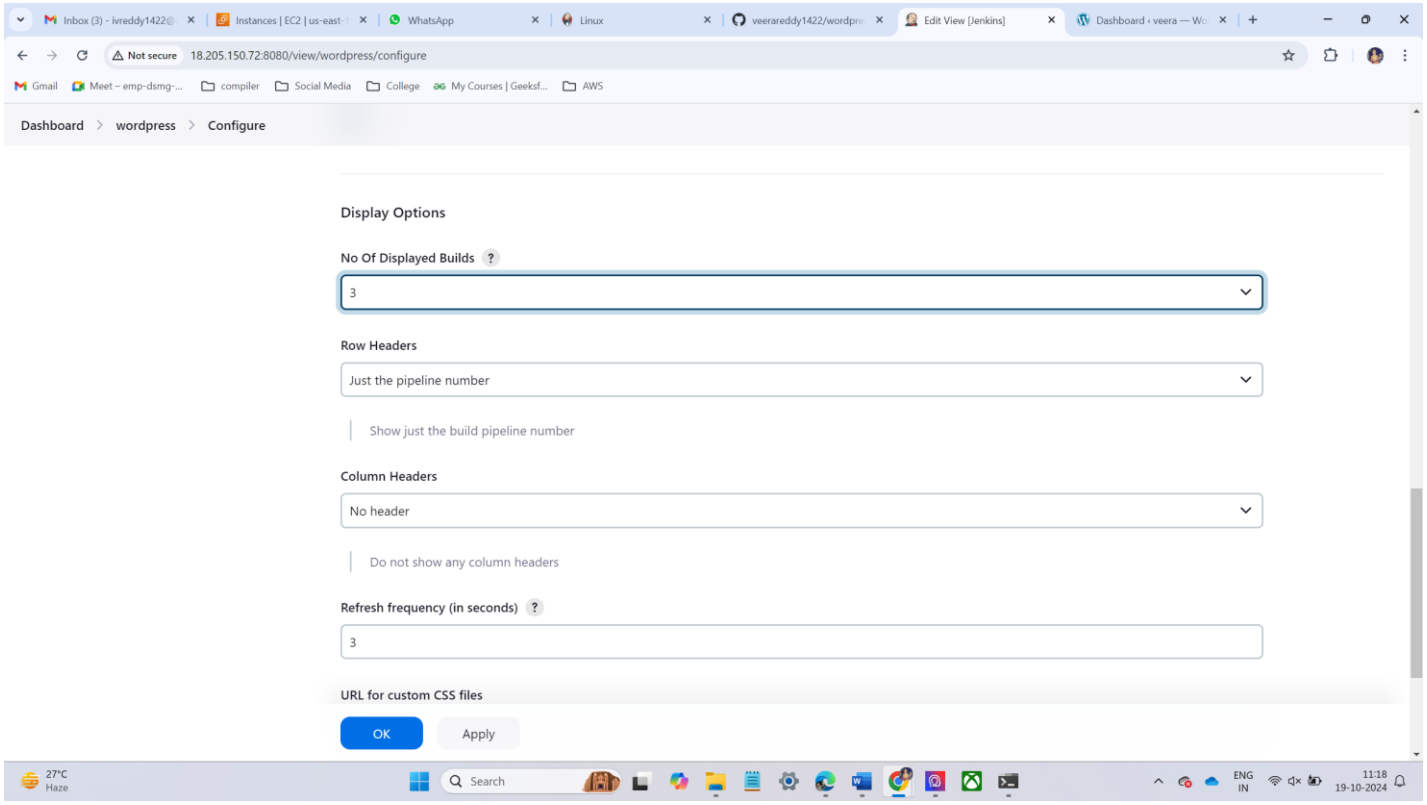
- Now create pipeline for wordpress application
- Goto dashboard→ under dashboard click + icon
- Enter pipeline name
- Select build pipeline view



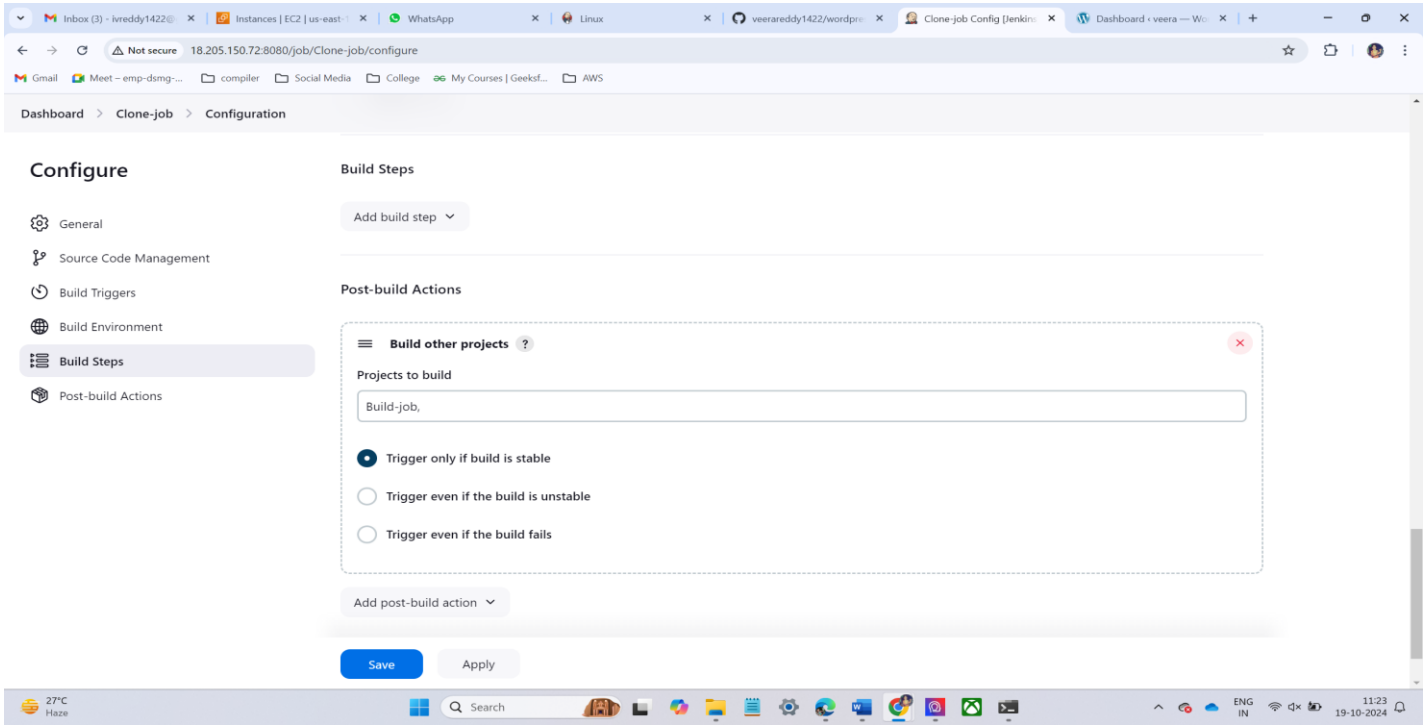
- Configuration the pipeline
- Under pipeline flow select **your initial job**



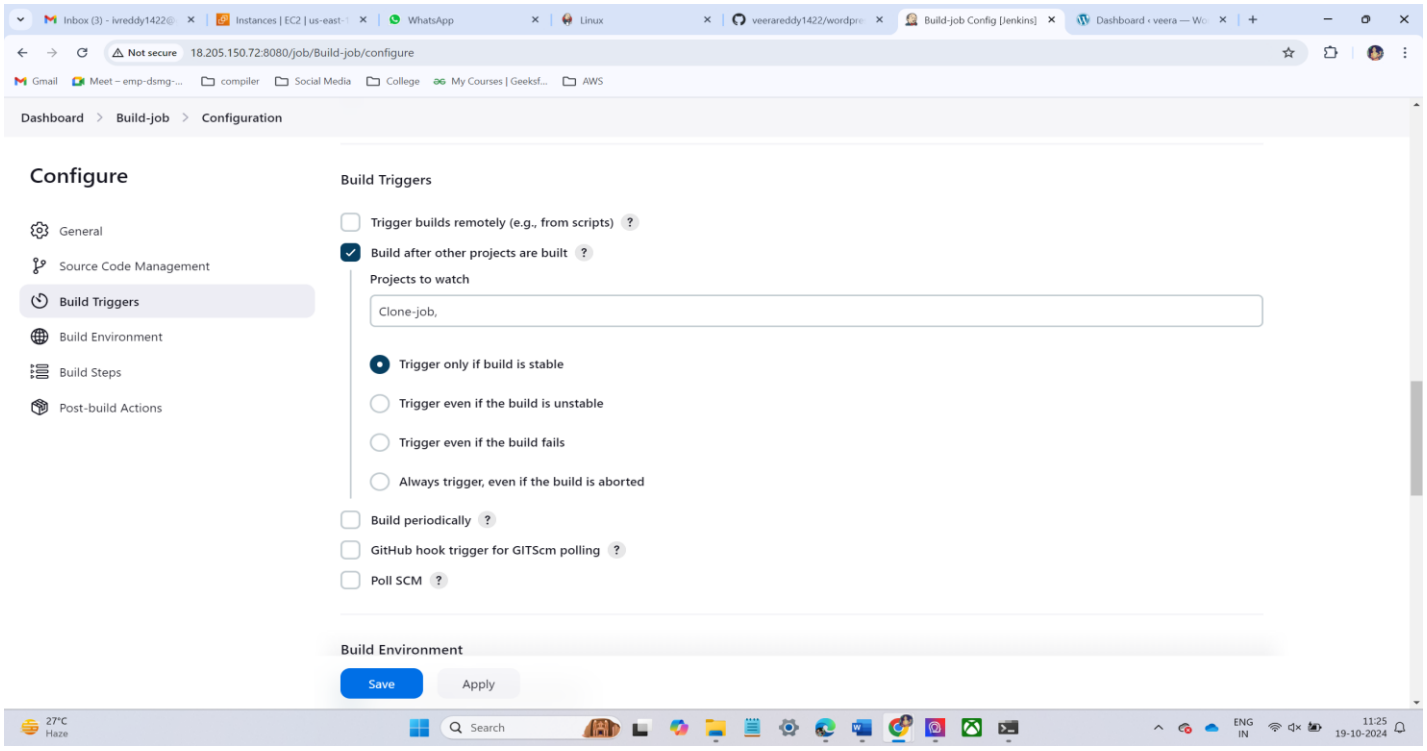
- Under display options select **No Of displayed build** (your wish)



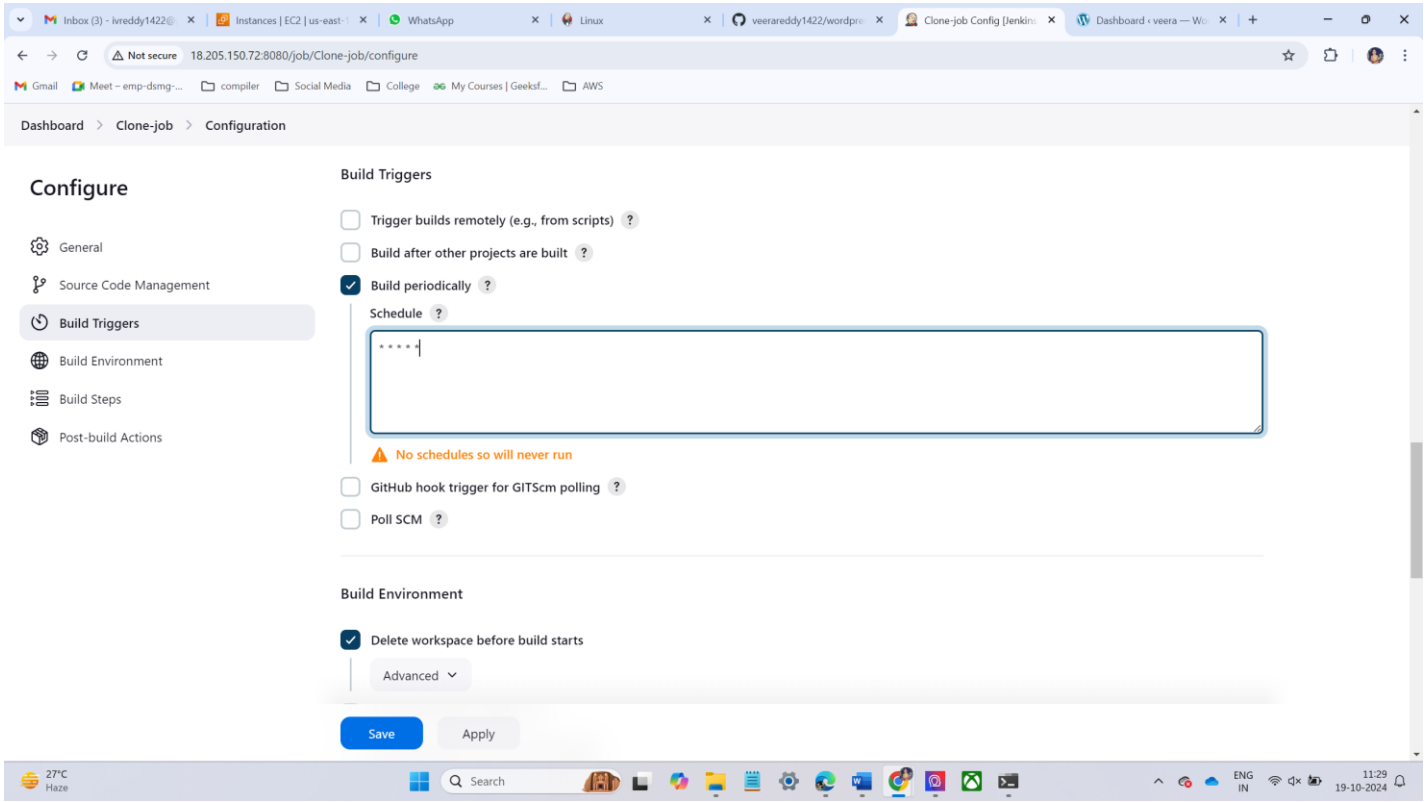
- Now configure the clone job
- Under clone job configuration go to post-build actions
- Under post build action select build other project
- Now select the build job



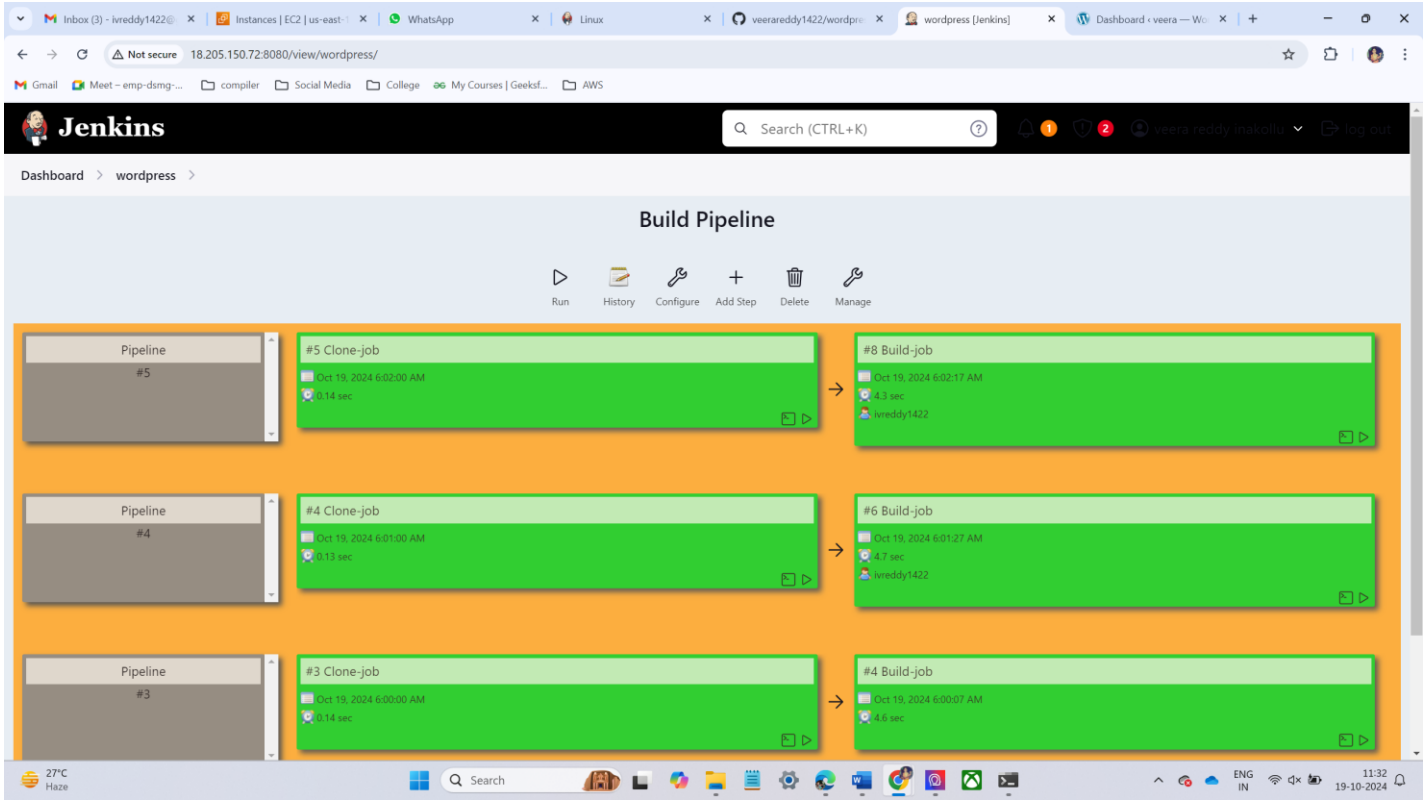
- Now configure the build job
- Under build job configuration go to build triggers
- Under build triggers select build after other projects are built
- Now select the clone job



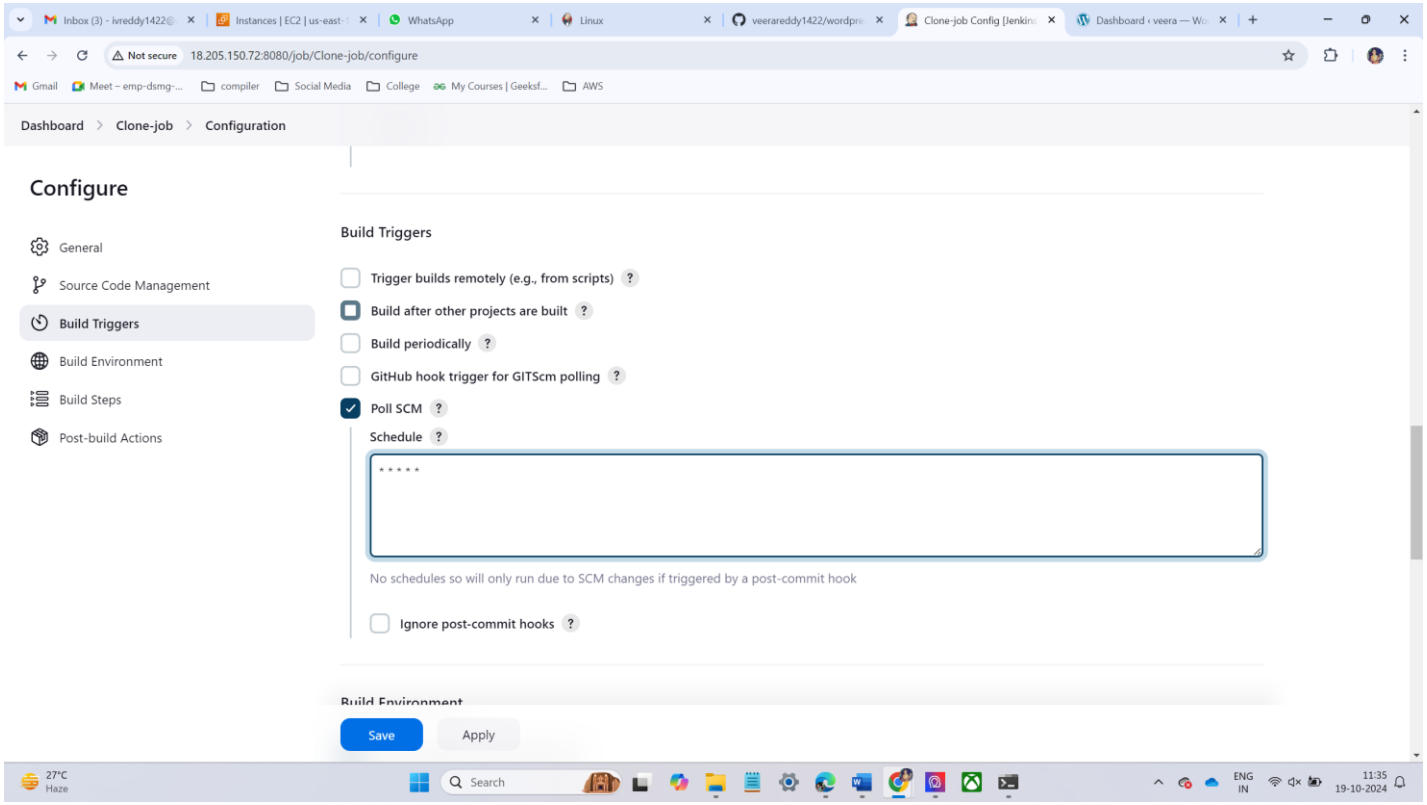
- Now apply build trigger action on clone job



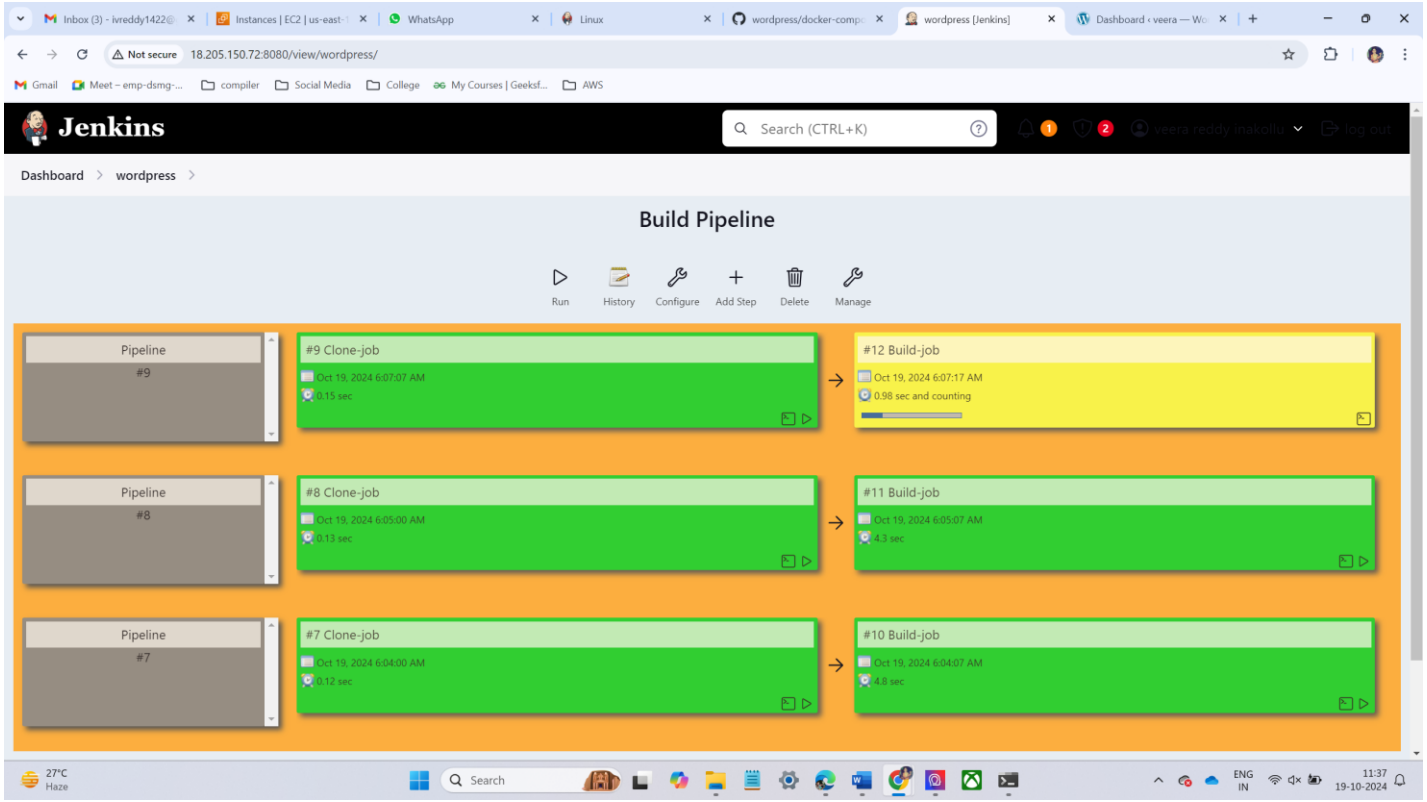
- Now go to pipeline check build pipeline actions
- It runs every minute



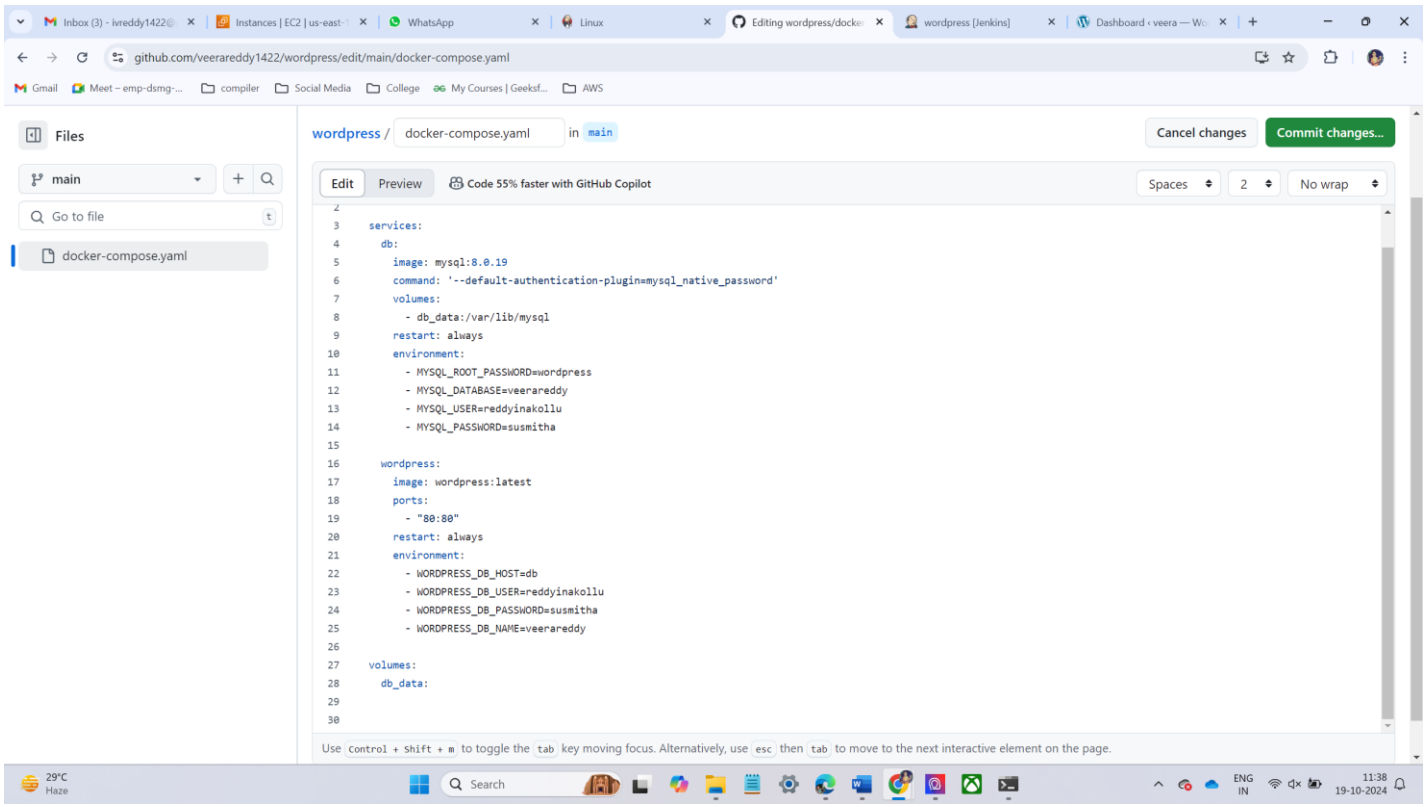
- Now apply poll SCM for clone job



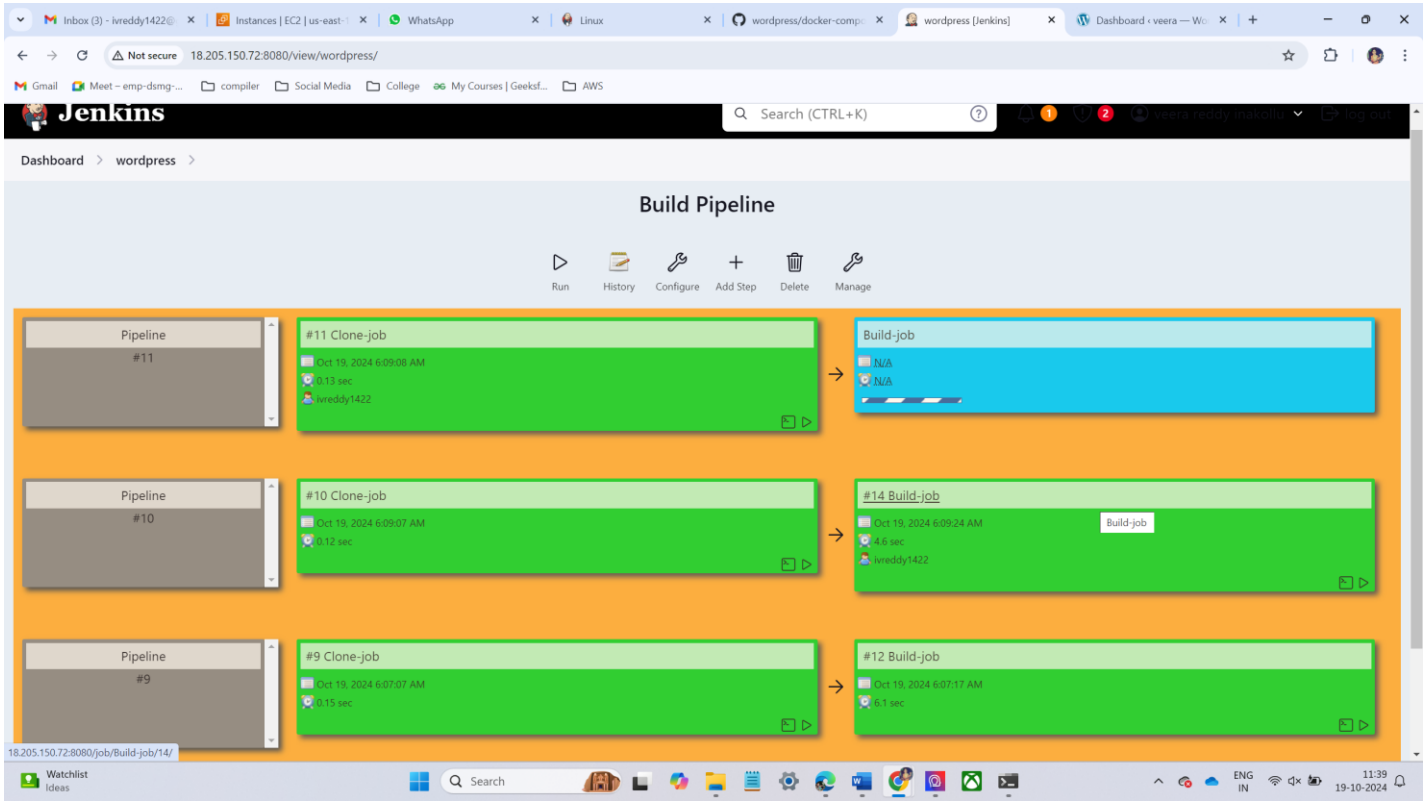
- Now go to pipeline check build pipeline actions
- When you are change the code in git hub it automatically runs



- In this time I am change the database name

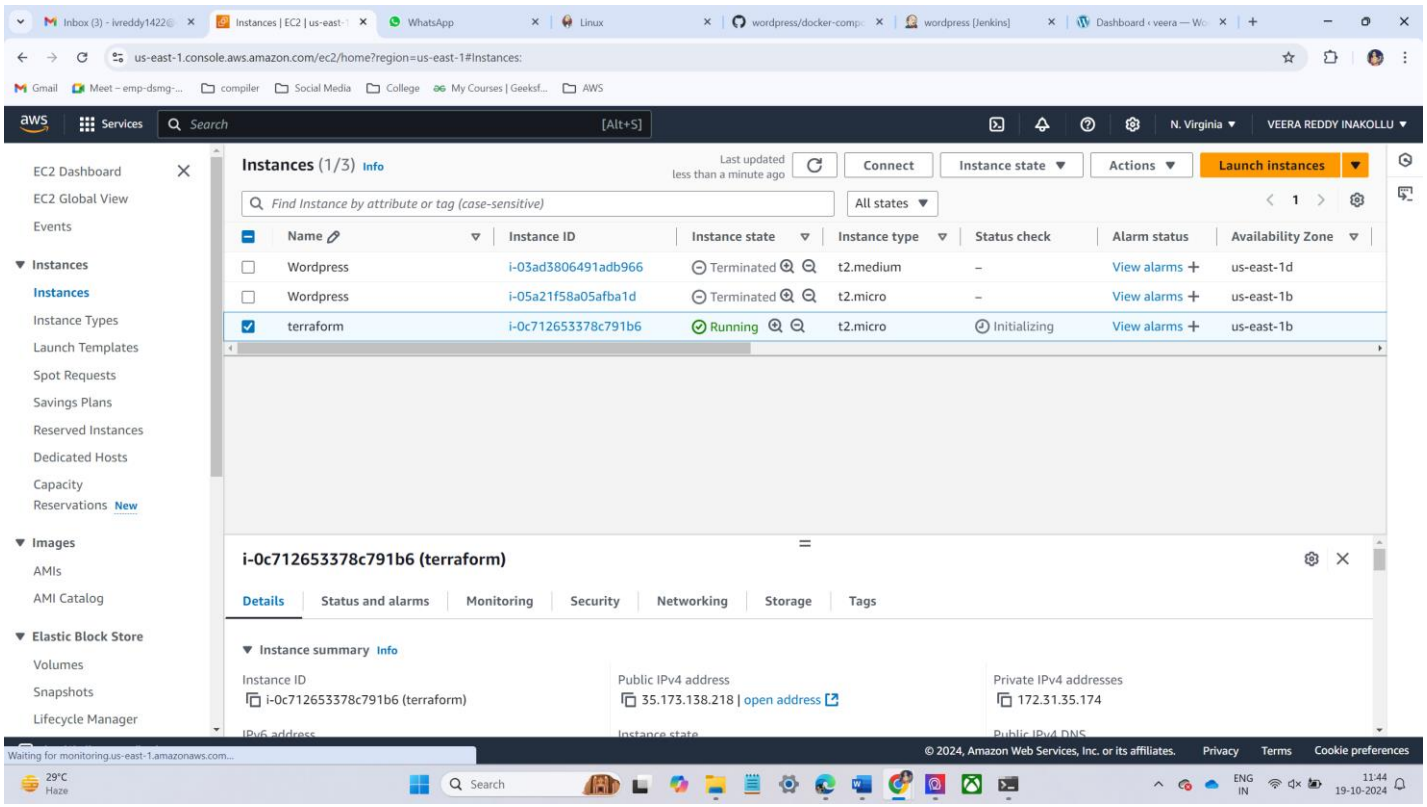


- Its runs automatically when I am changing the code

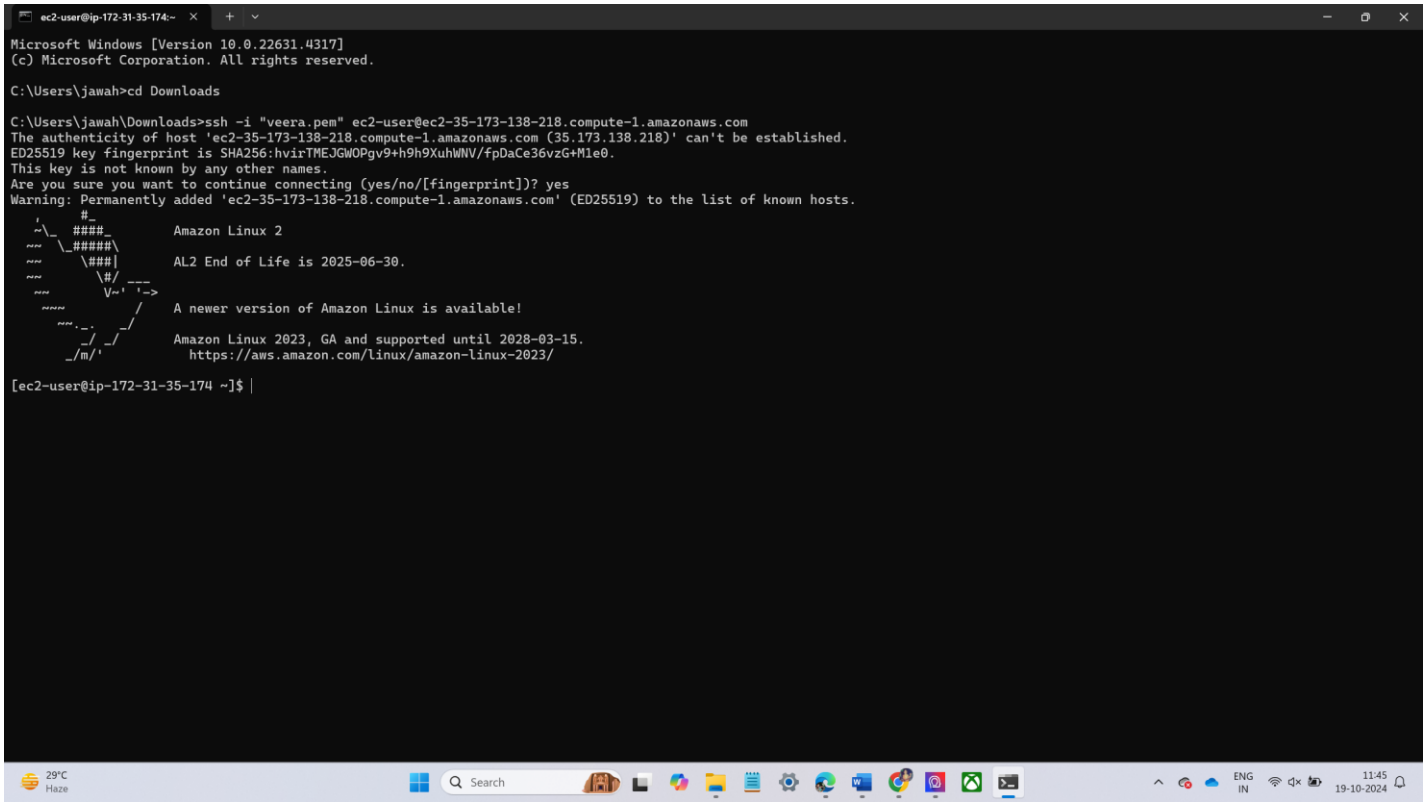


Deploy WordPress web application by using terraform (create Ec2 instance along with userdata .sh file)

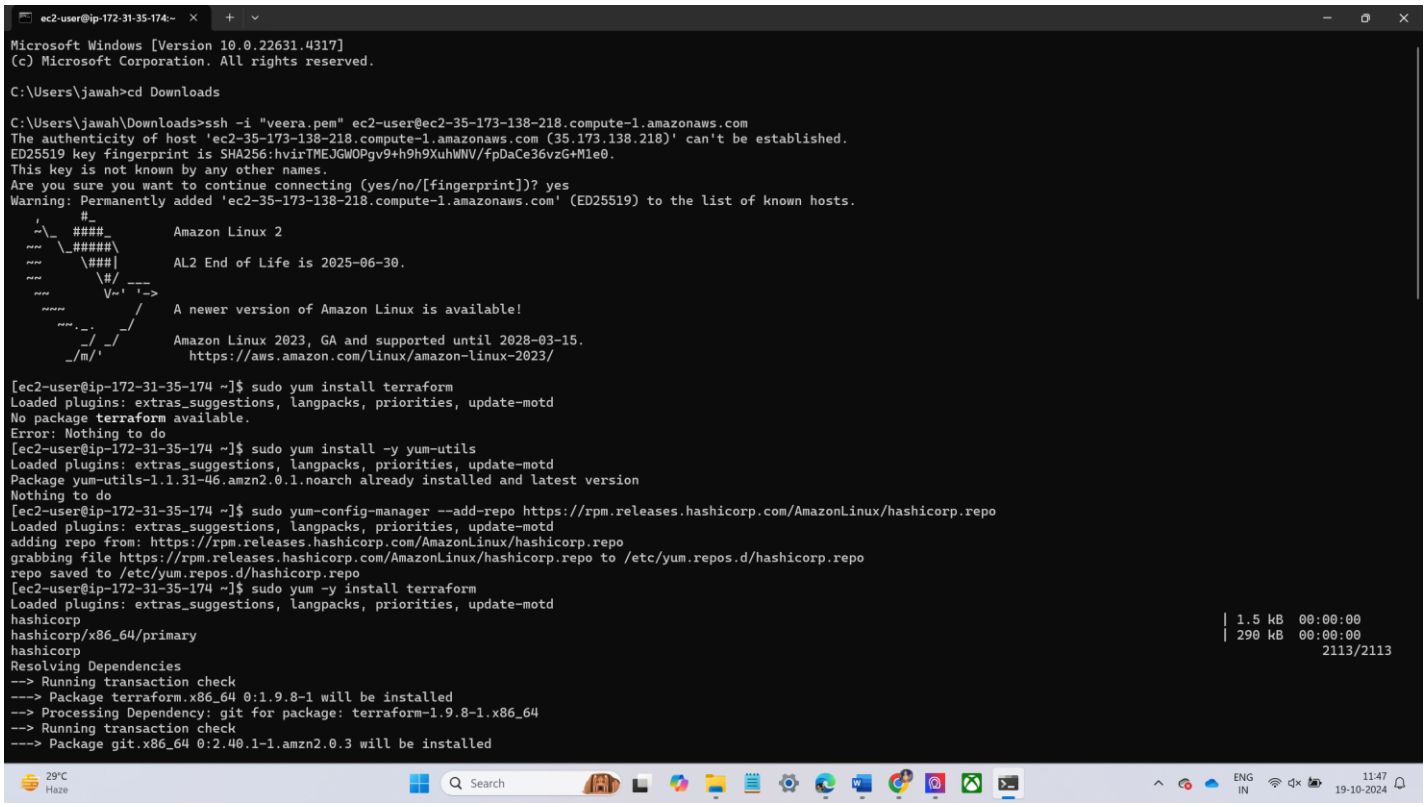
- Create ec2 instance for using terraform



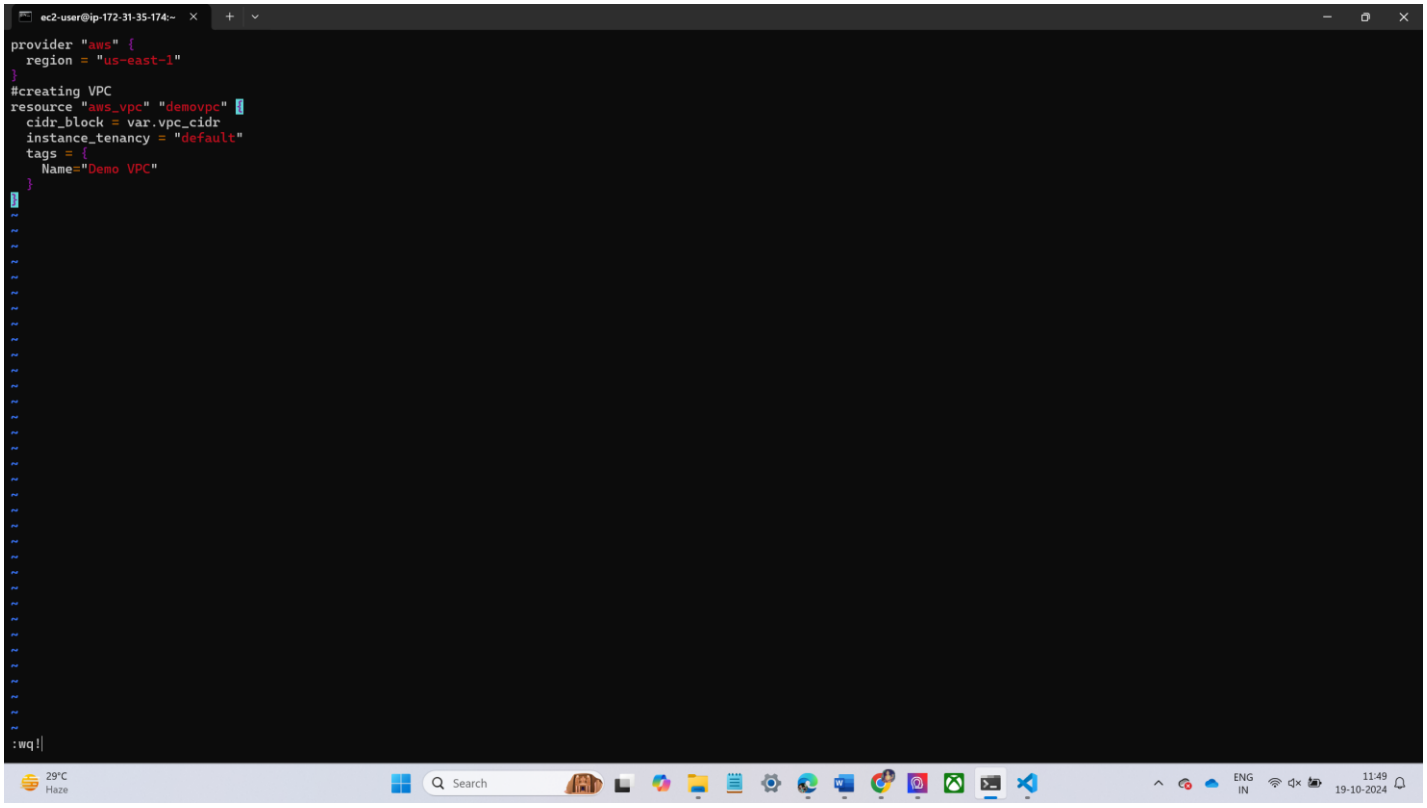
- Connect ec2 instance to the command prompt



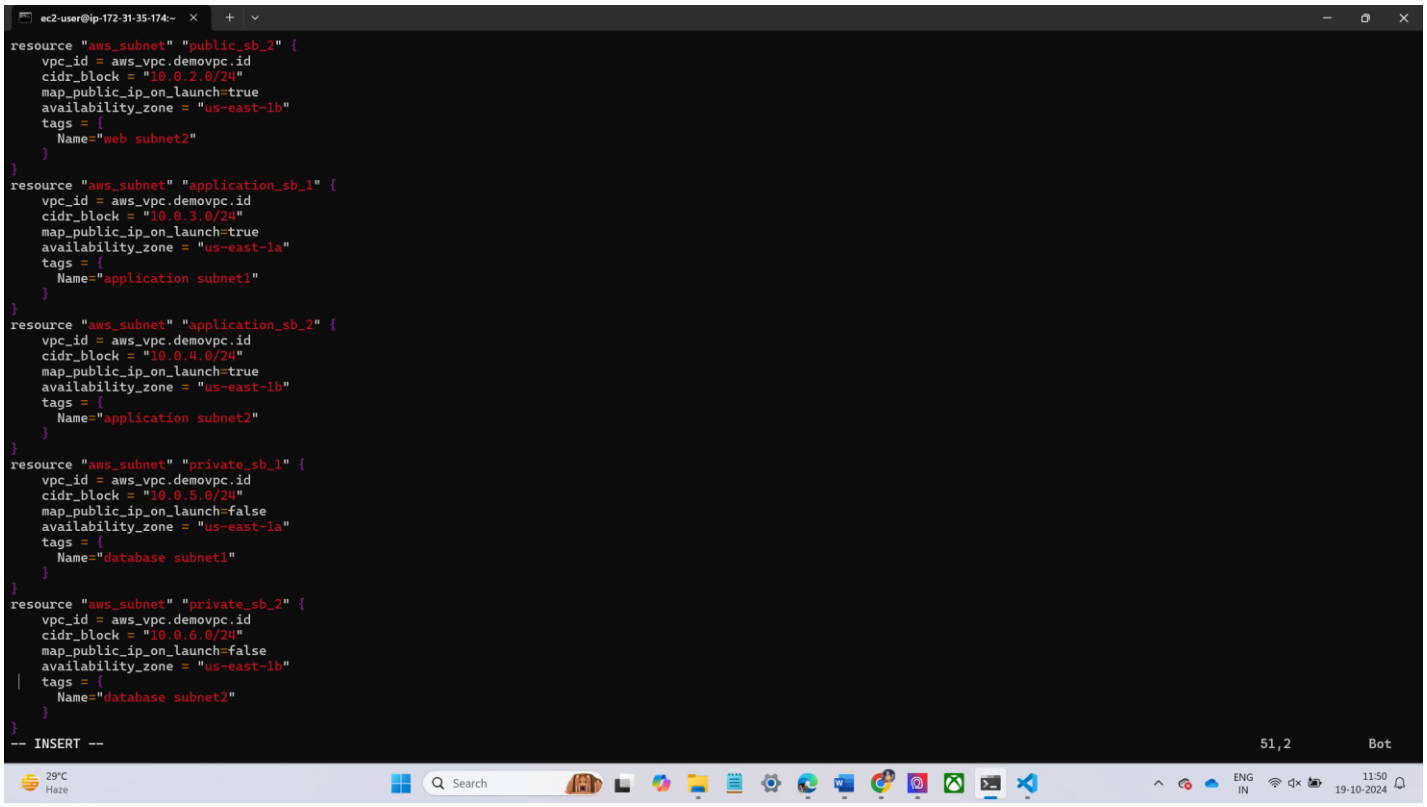
- Install terraform and set up the terraform



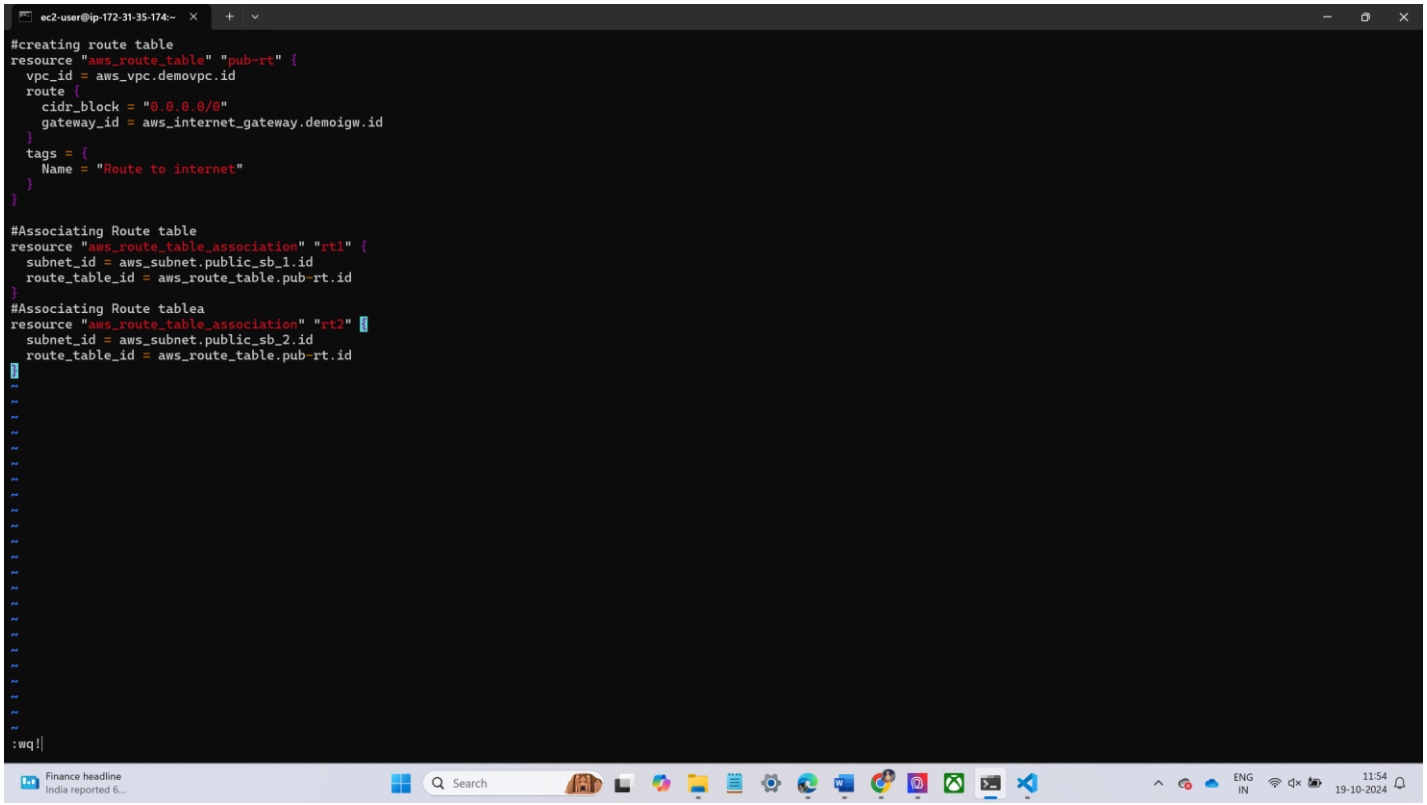
- Now write terraform script for vpc



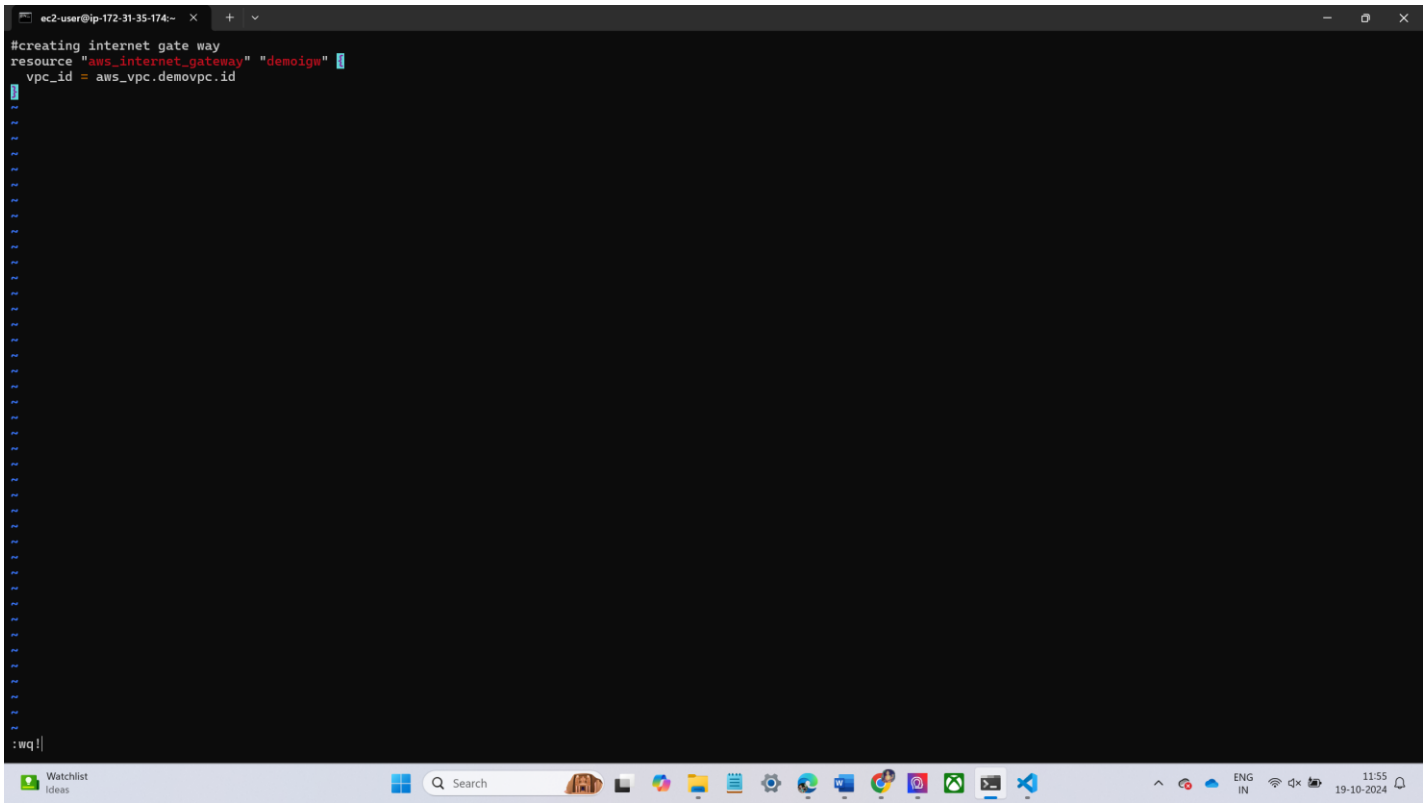
- Create subnets.tf



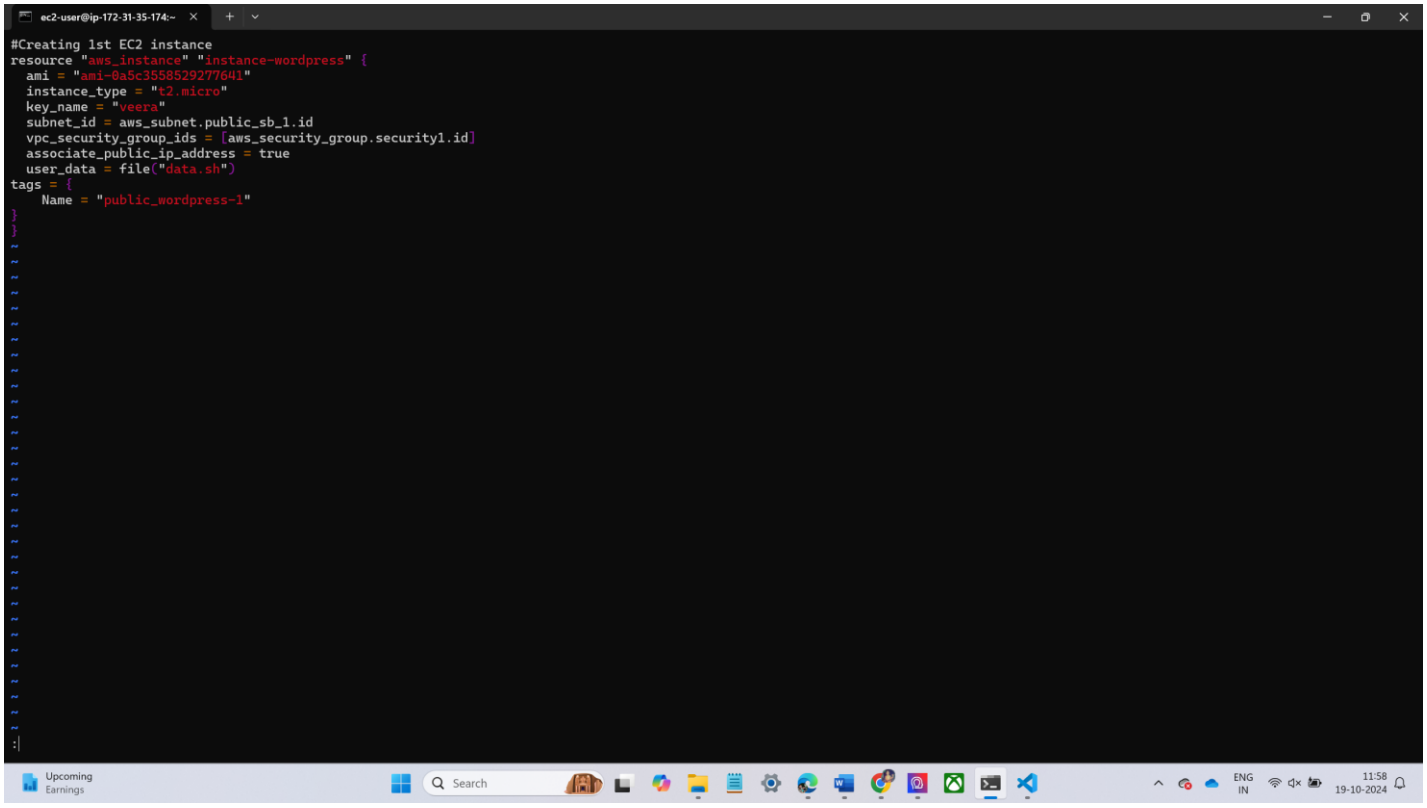
- Create file for route table



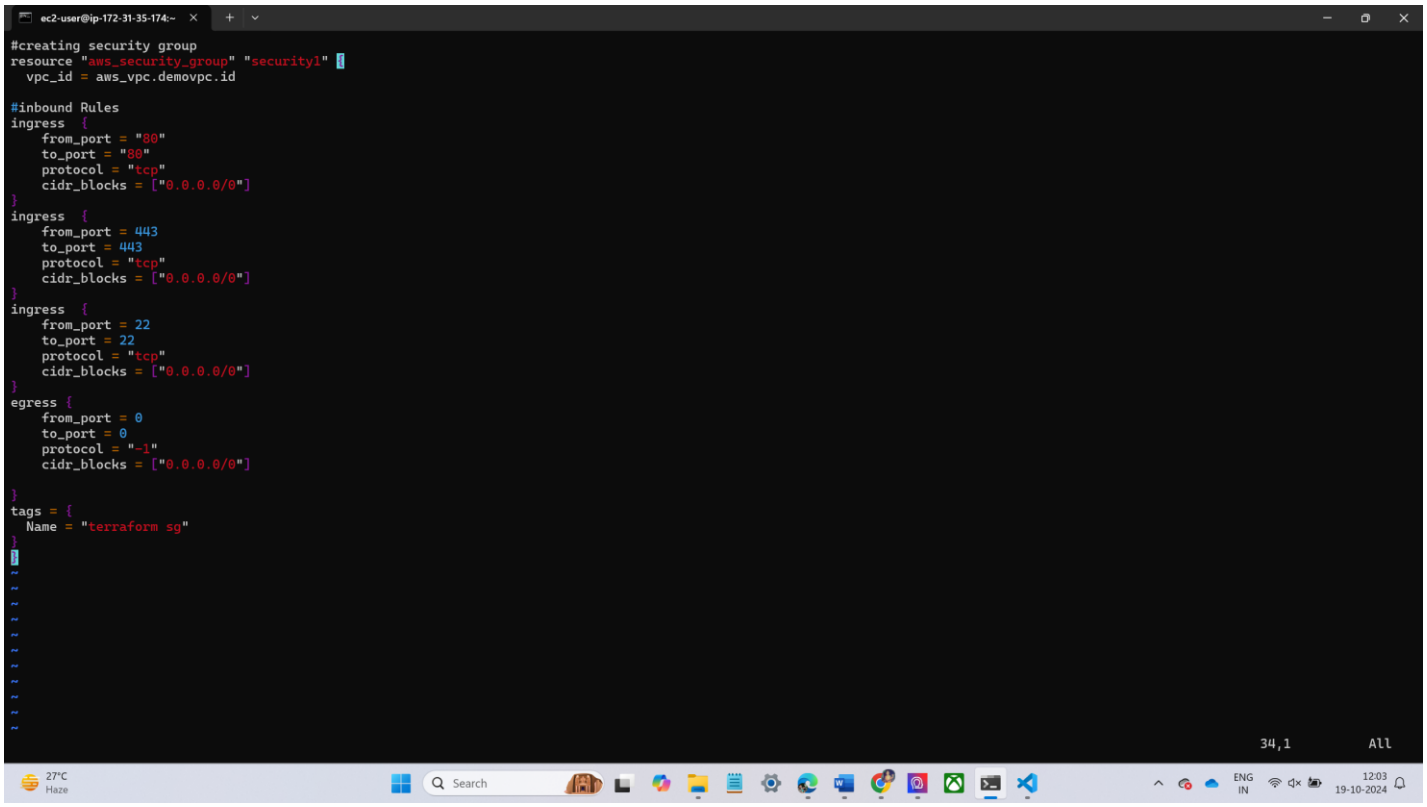
- Create file for internet gateway



- Create file for ec2 instance

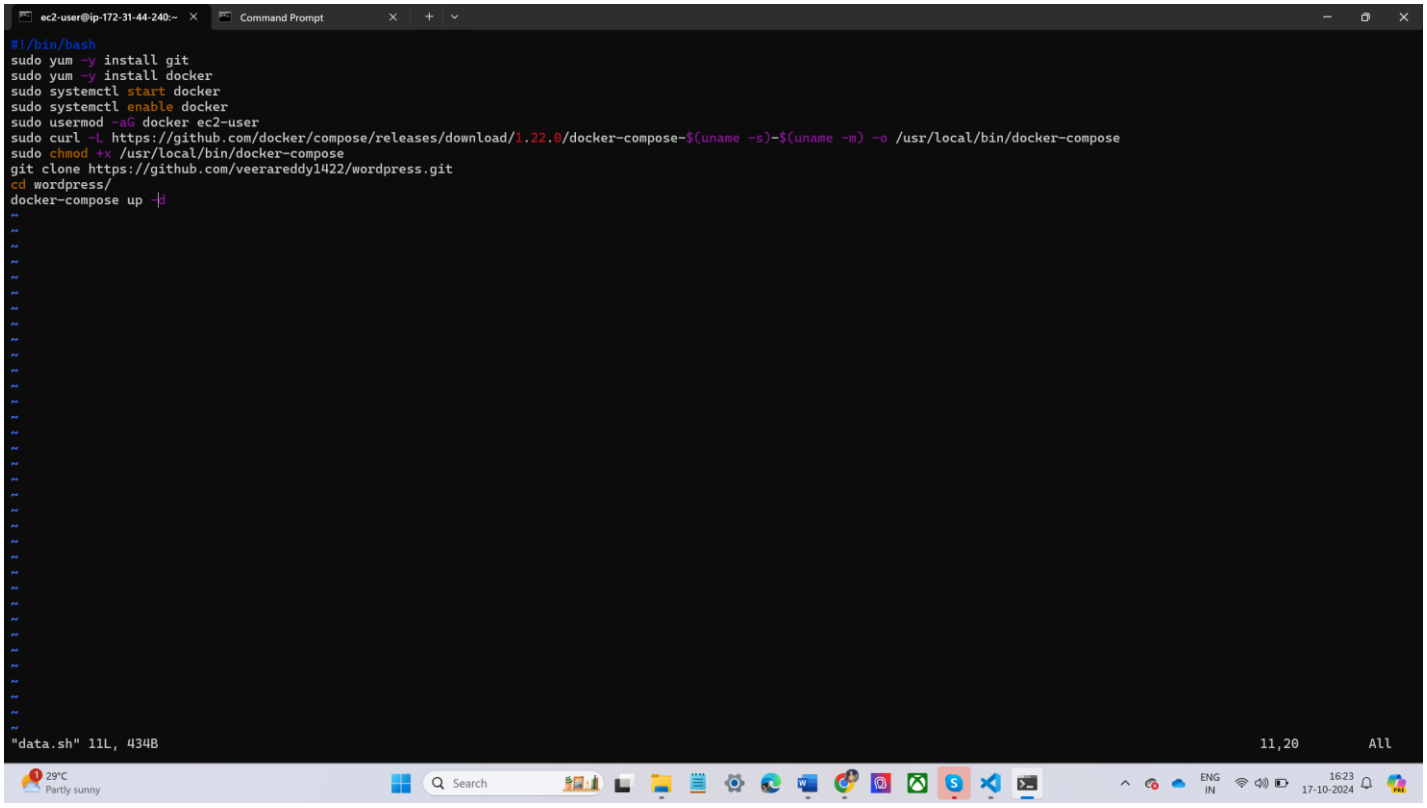


- create file for security groups



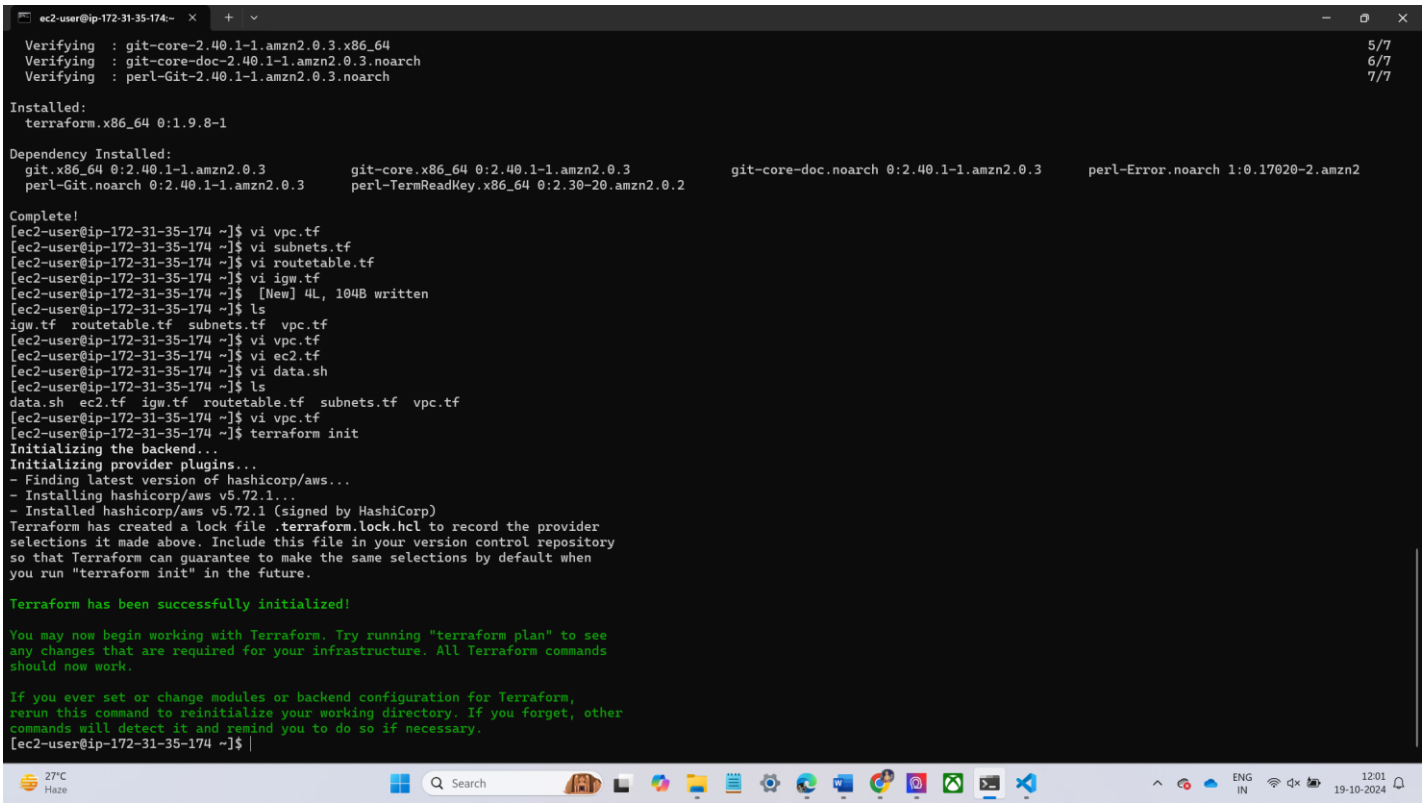
```
#creating security group
resource "aws_security_group" "security1" {
  vpc_id = aws_vpc.demovpc.id

#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

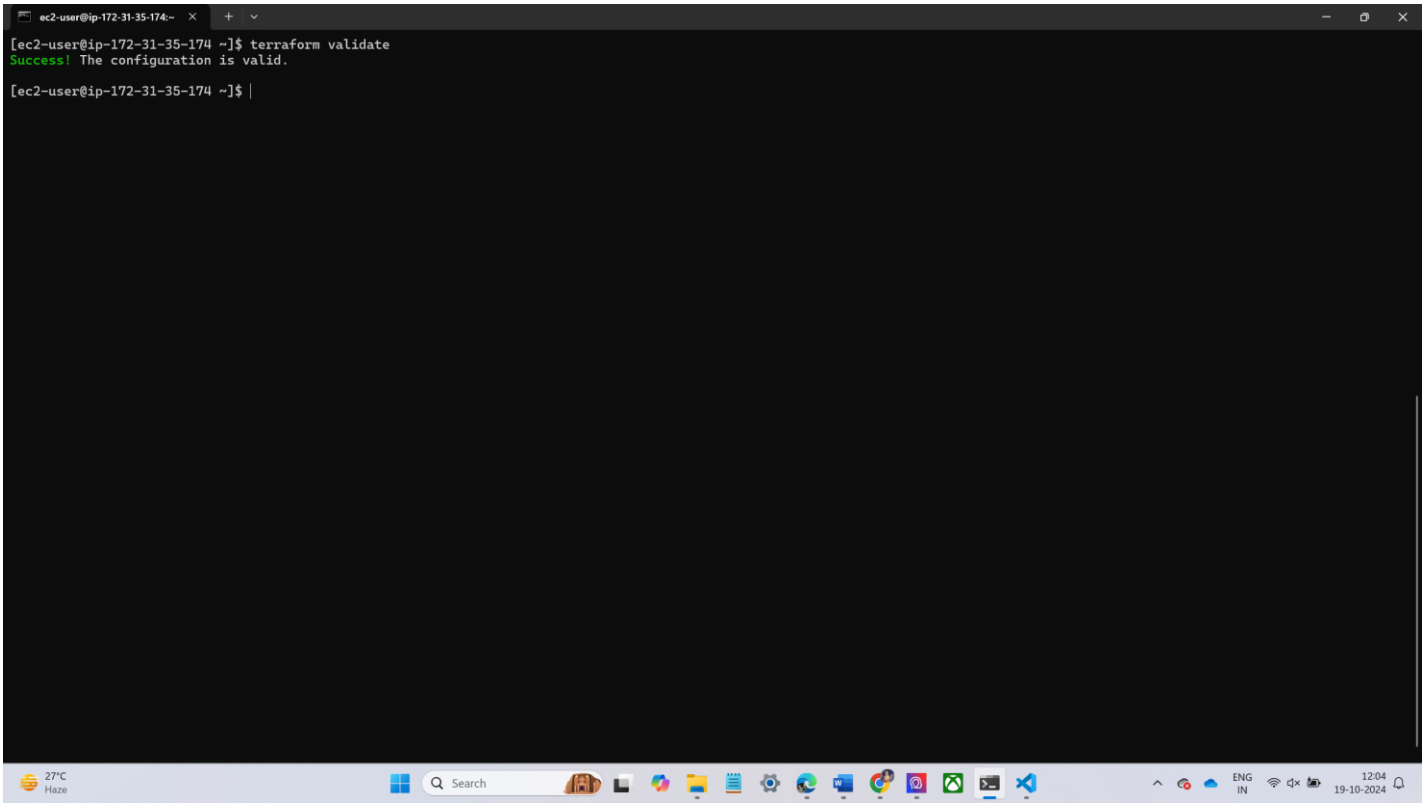


```
#!/bin/bash
sudo yum -y install git
sudo yum -y install docker
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose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

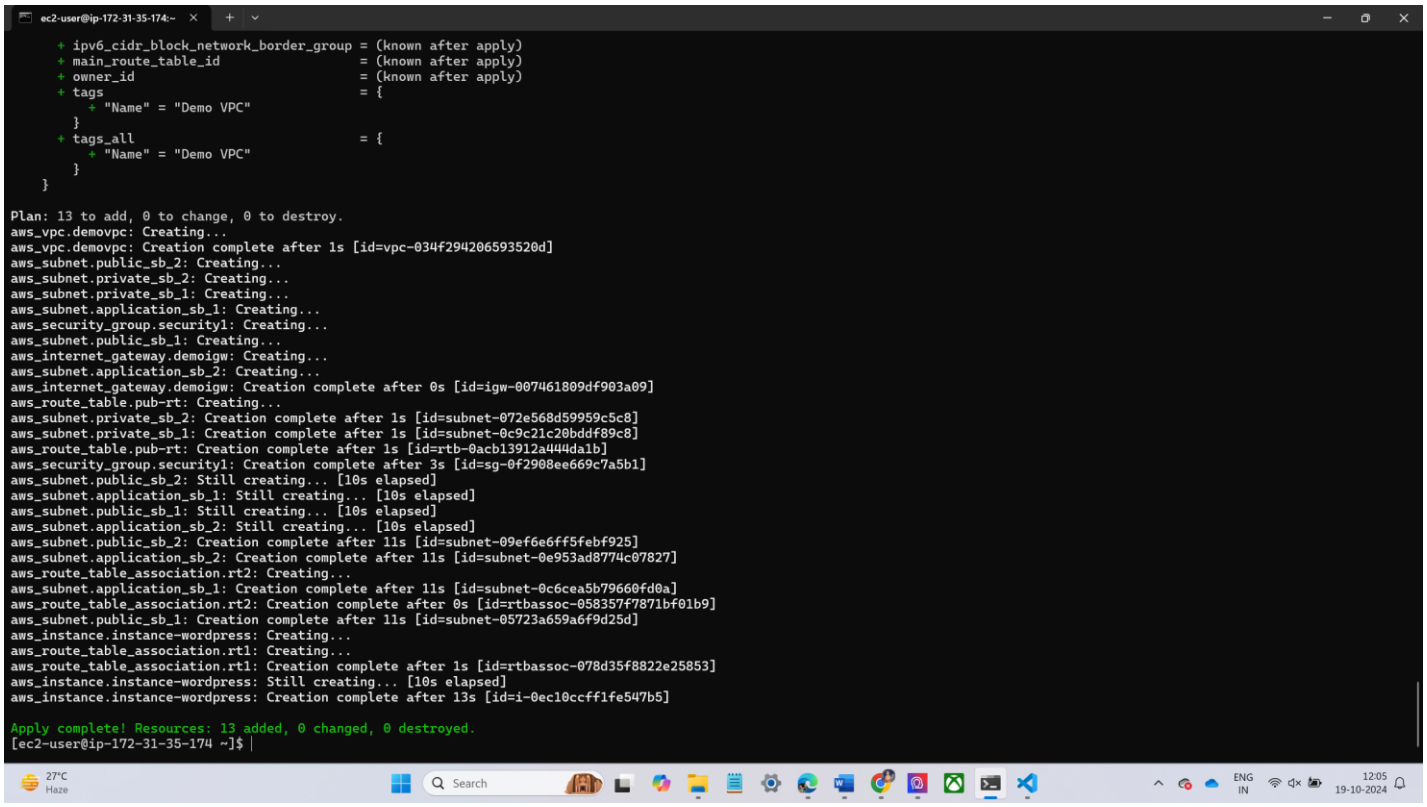
- After creation of all files apply terraform init command



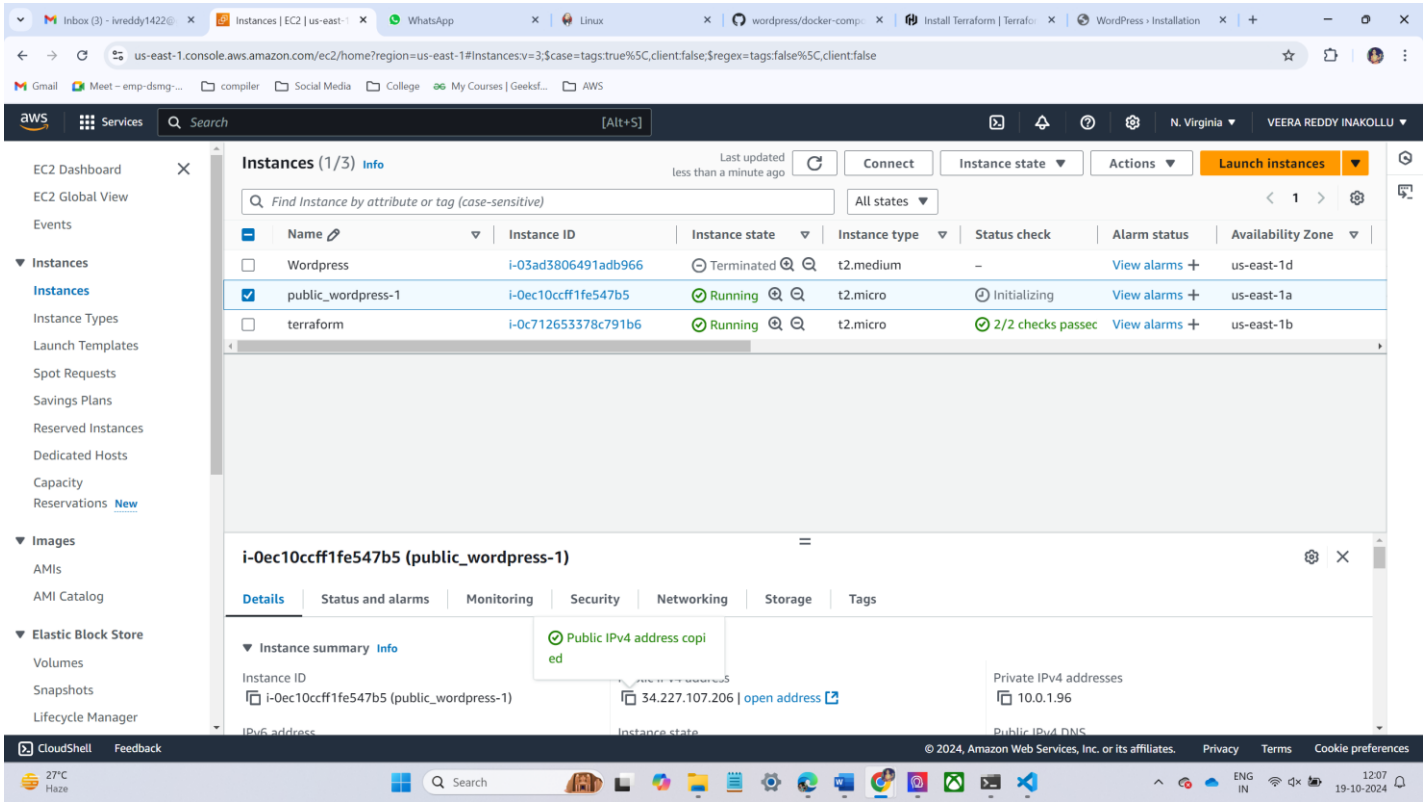
- Now check the errors for using terraform validate



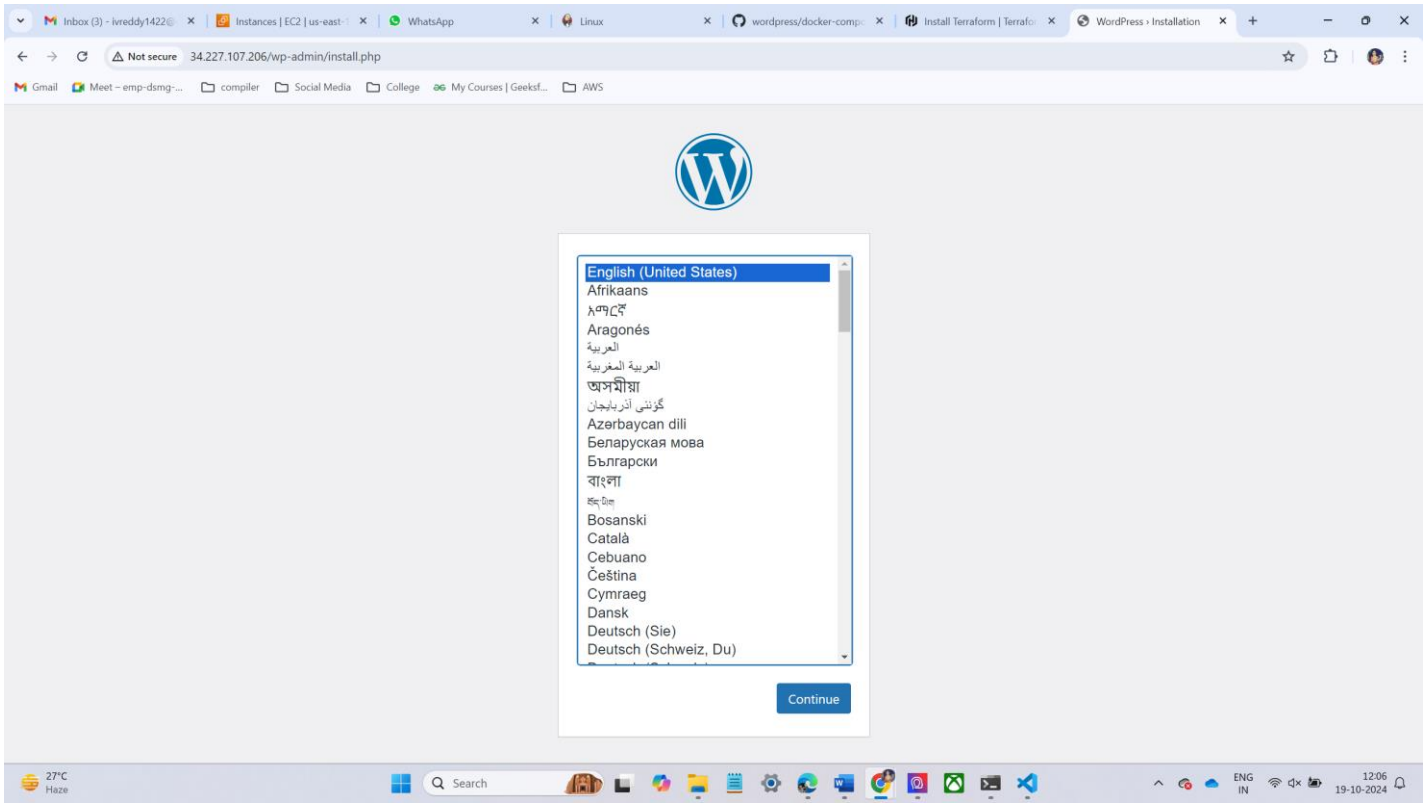
- Now create the resource for using terraform apply command



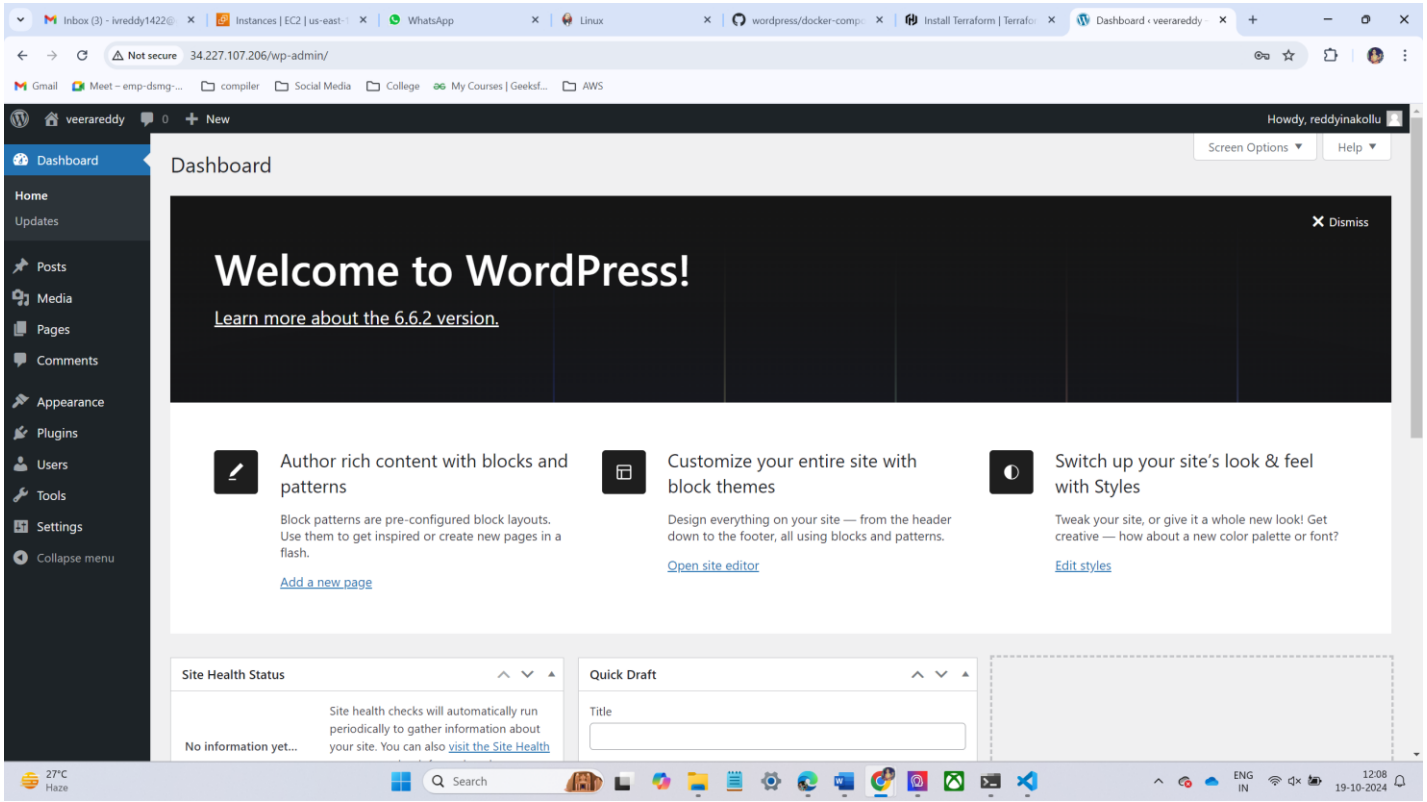
- Now check the aws ec2 console.
- New ec2 instance was created



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

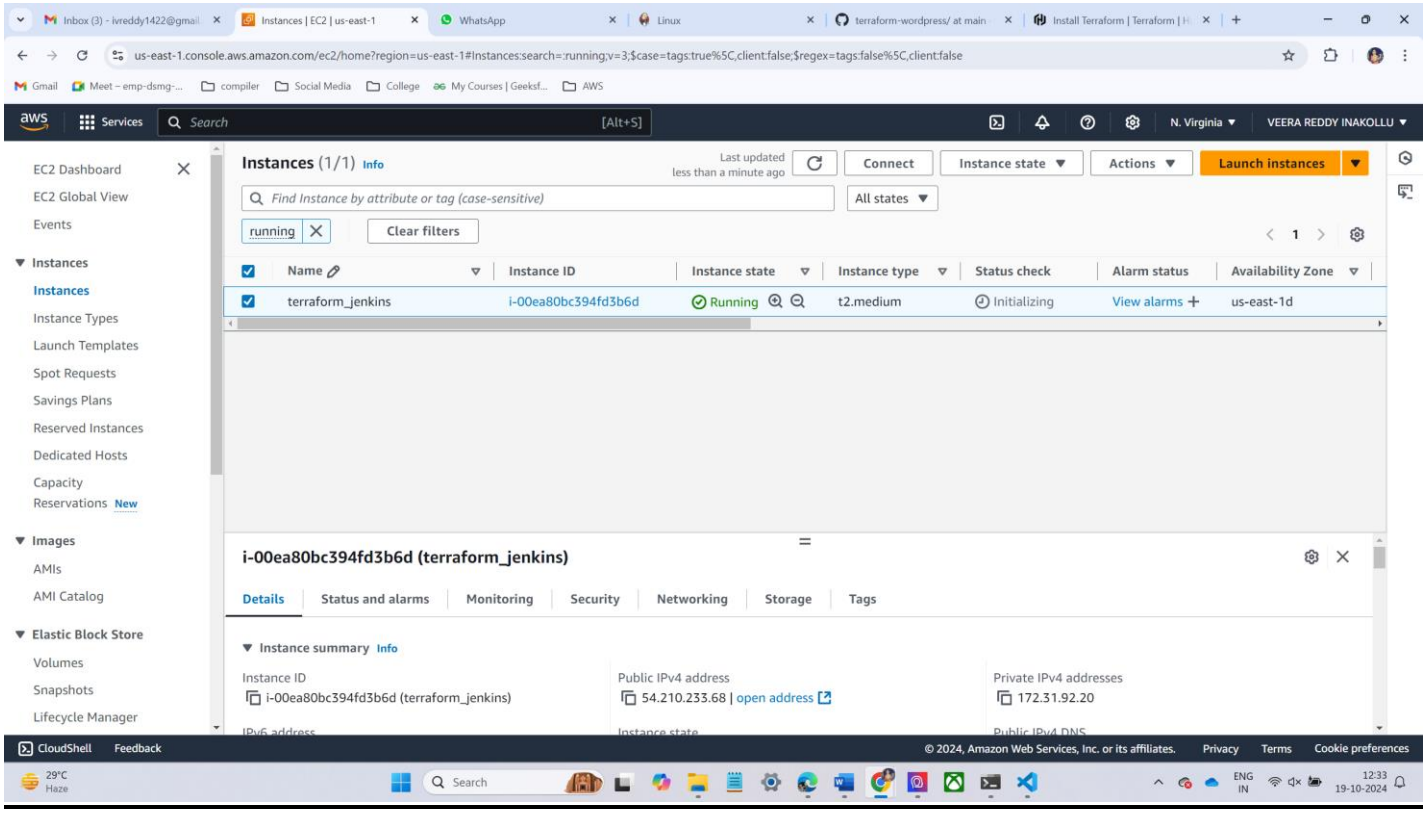


- Successfully deployed the WordPress application using terraform

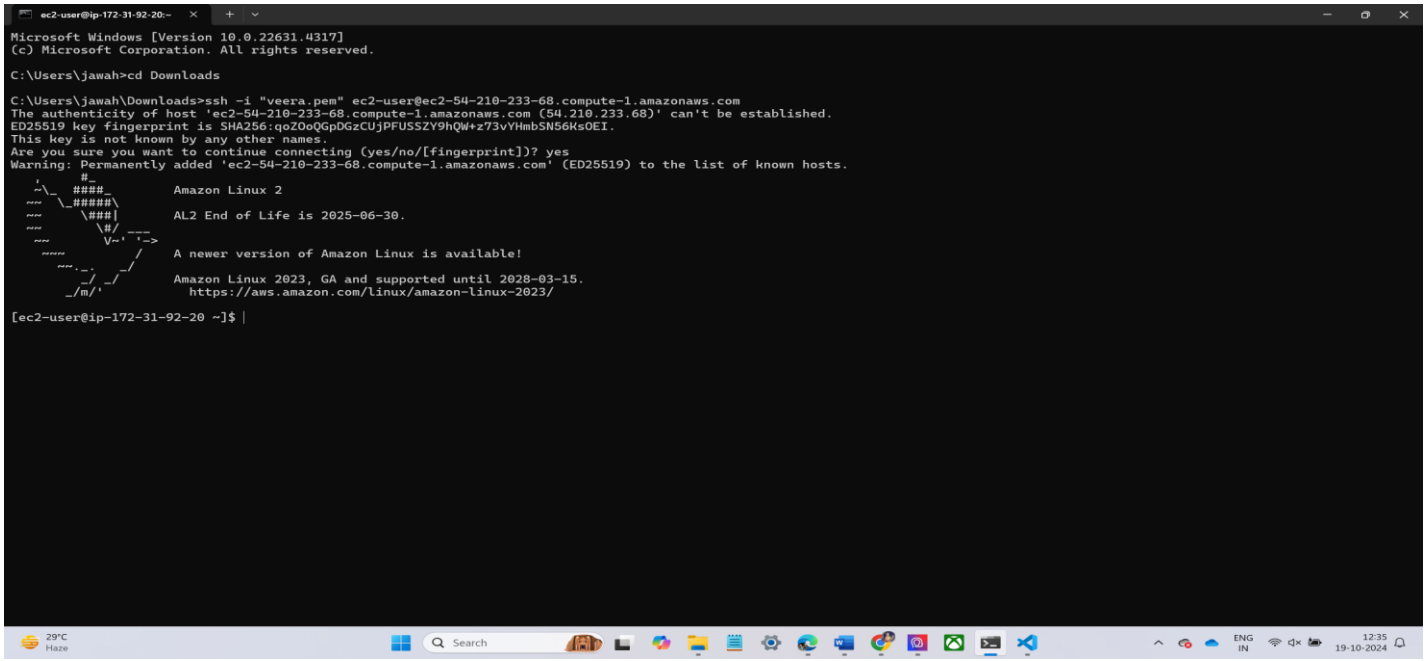


Deploy WordPress web application by using git (clone terraform script which helps to deploy WordPress web application), jenkins (in execute shell install terraform, init, fmt, validate and apply with automatic command as terraform apply --auto-approve) and terraform.

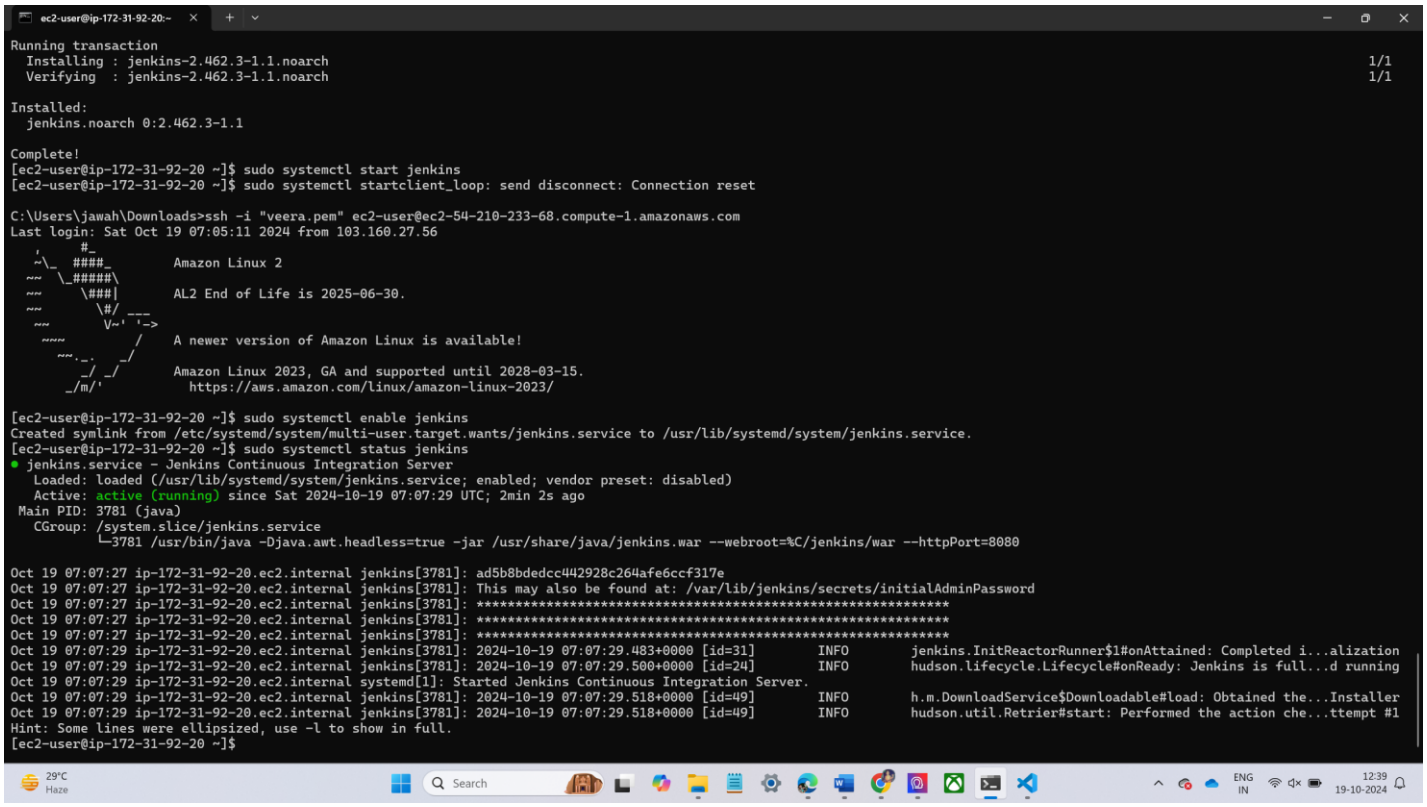
- Login to AWS account and launch EC2 instance



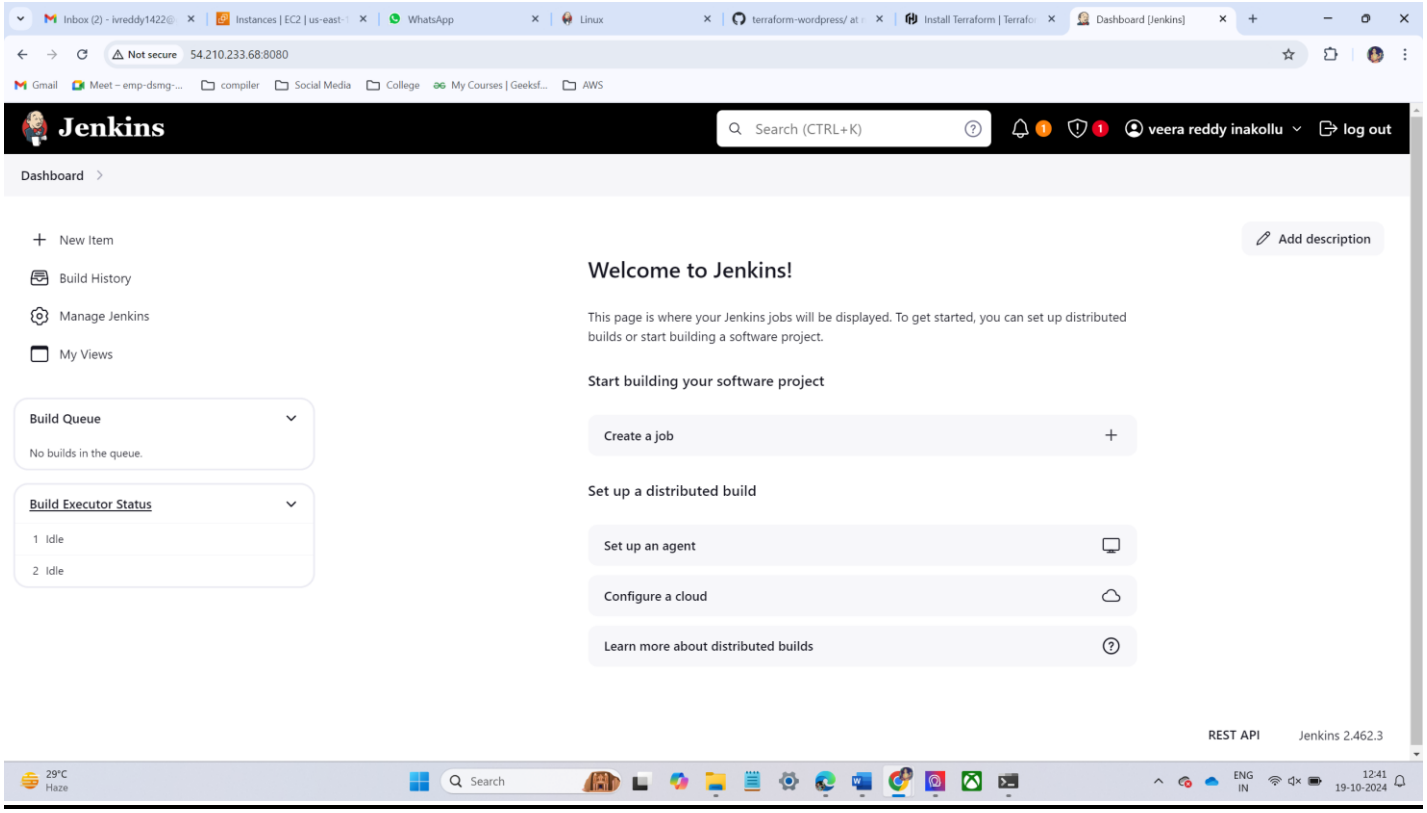
- Connect ec2 instance to the command prompt



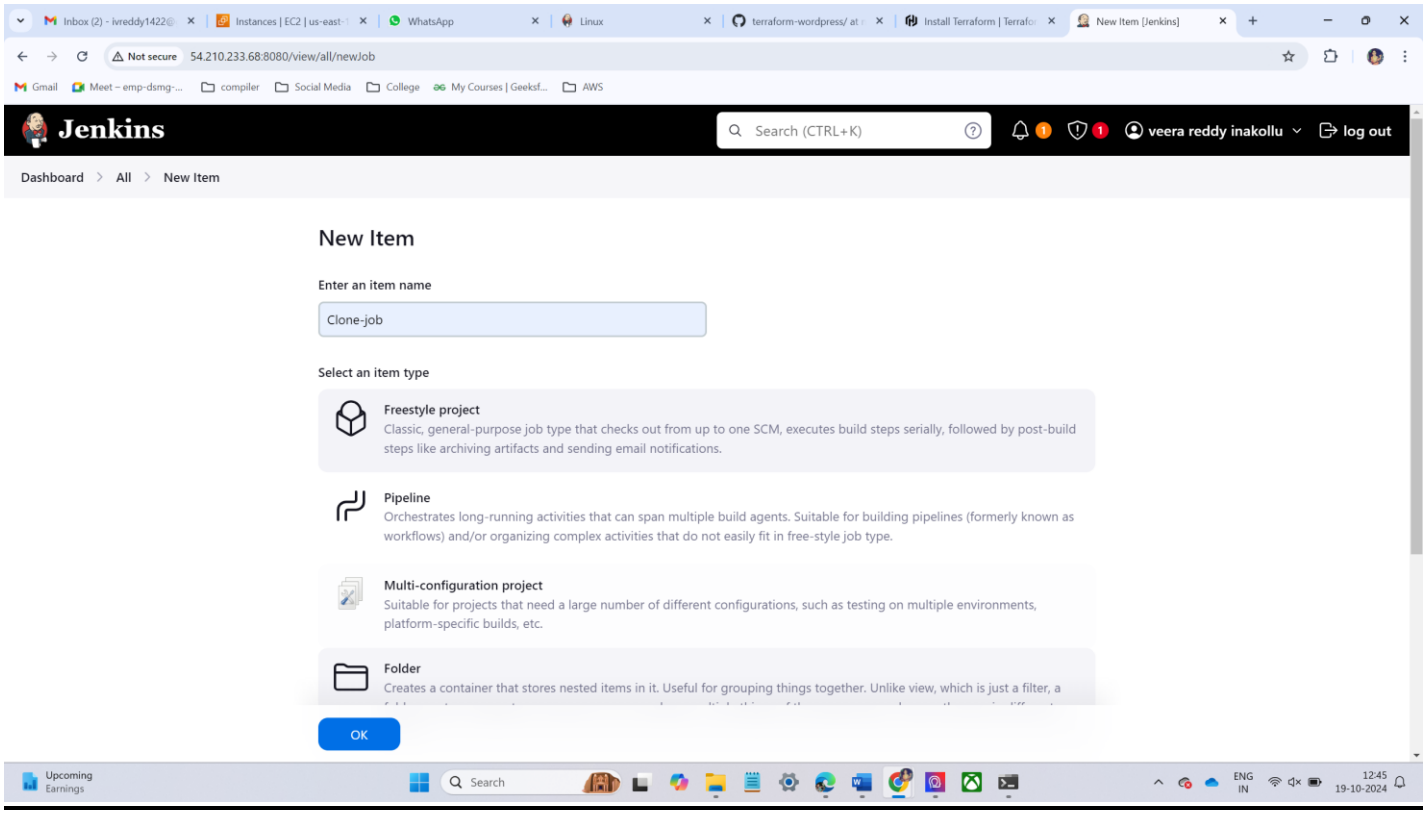
- Now install Jenkins and setup Jenkins



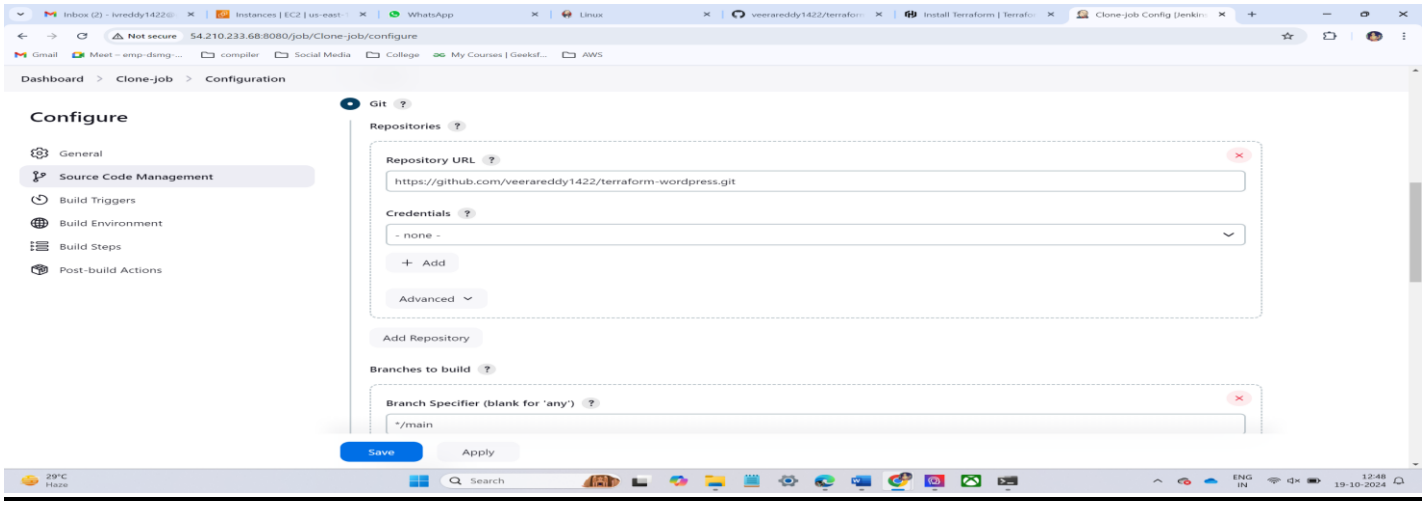
- Now launch Jenkins server on new tab using public ipv4 address



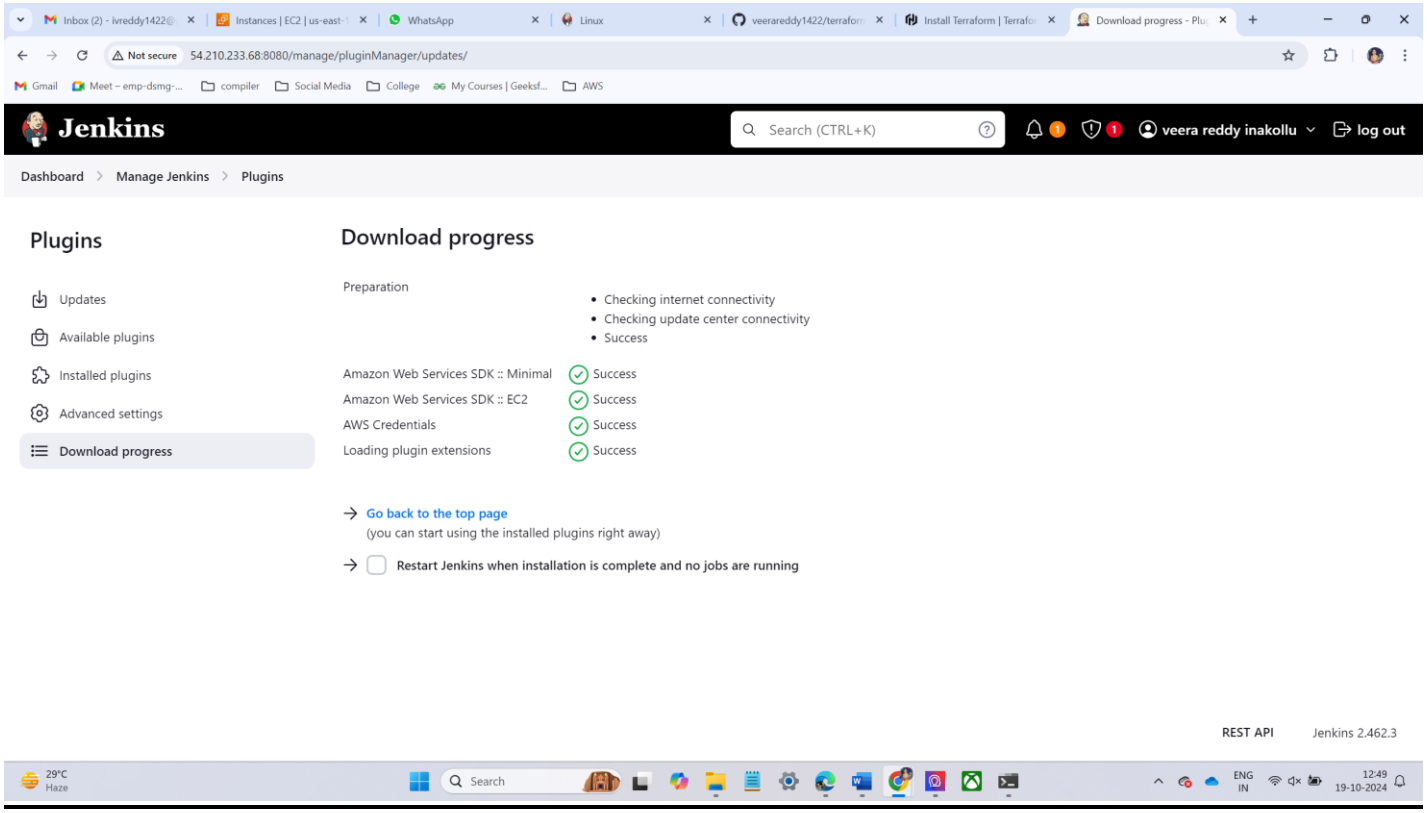
- Create job for clone the code
- Click on **"New Item"**.
- Enter a name for your project (e.g., WordPress-Deployment).
- Select **"Freestyle project"** and click **OK**.



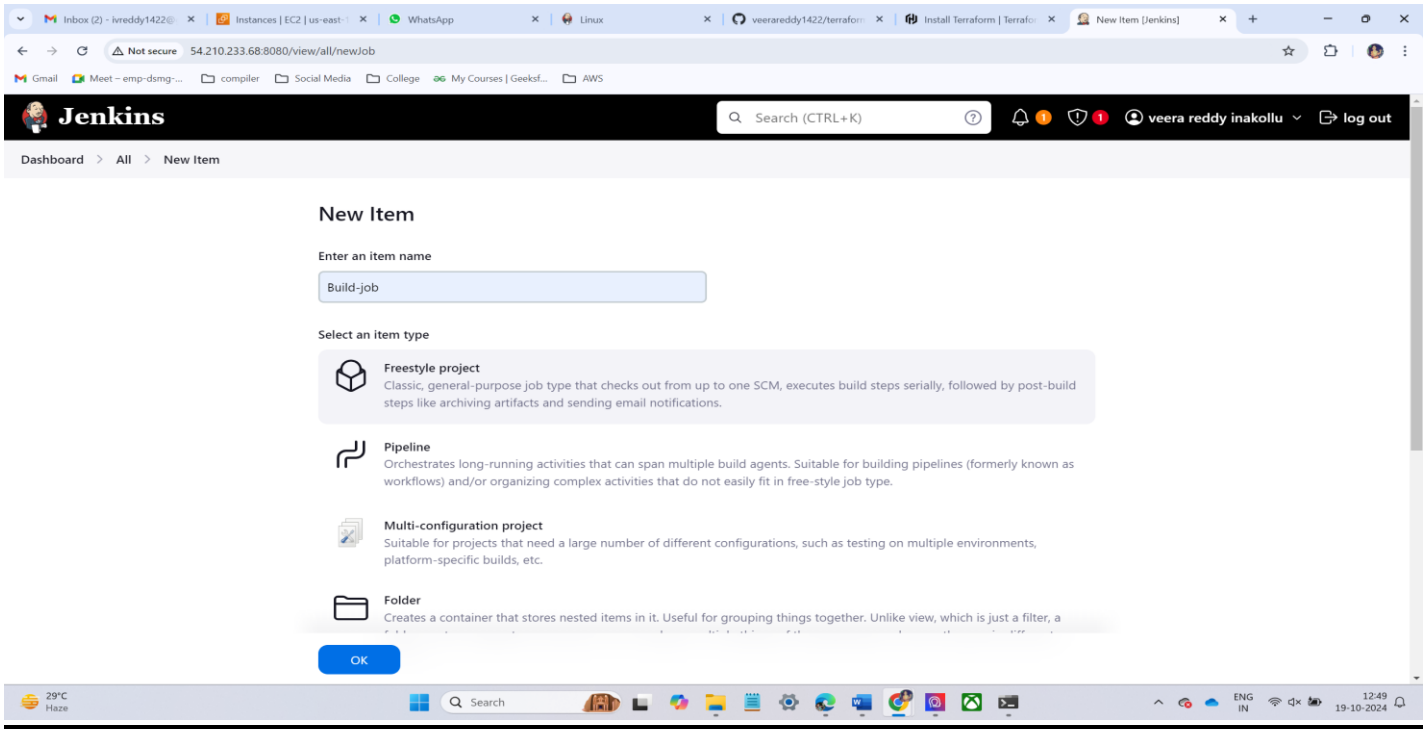
- Configure Source Code Management (SCM)
- In the project configuration page, scroll down to the **"Source Code Management"** section.
- Select **"Git"**.
- Enter the **Repository URL** of your GitHub repository



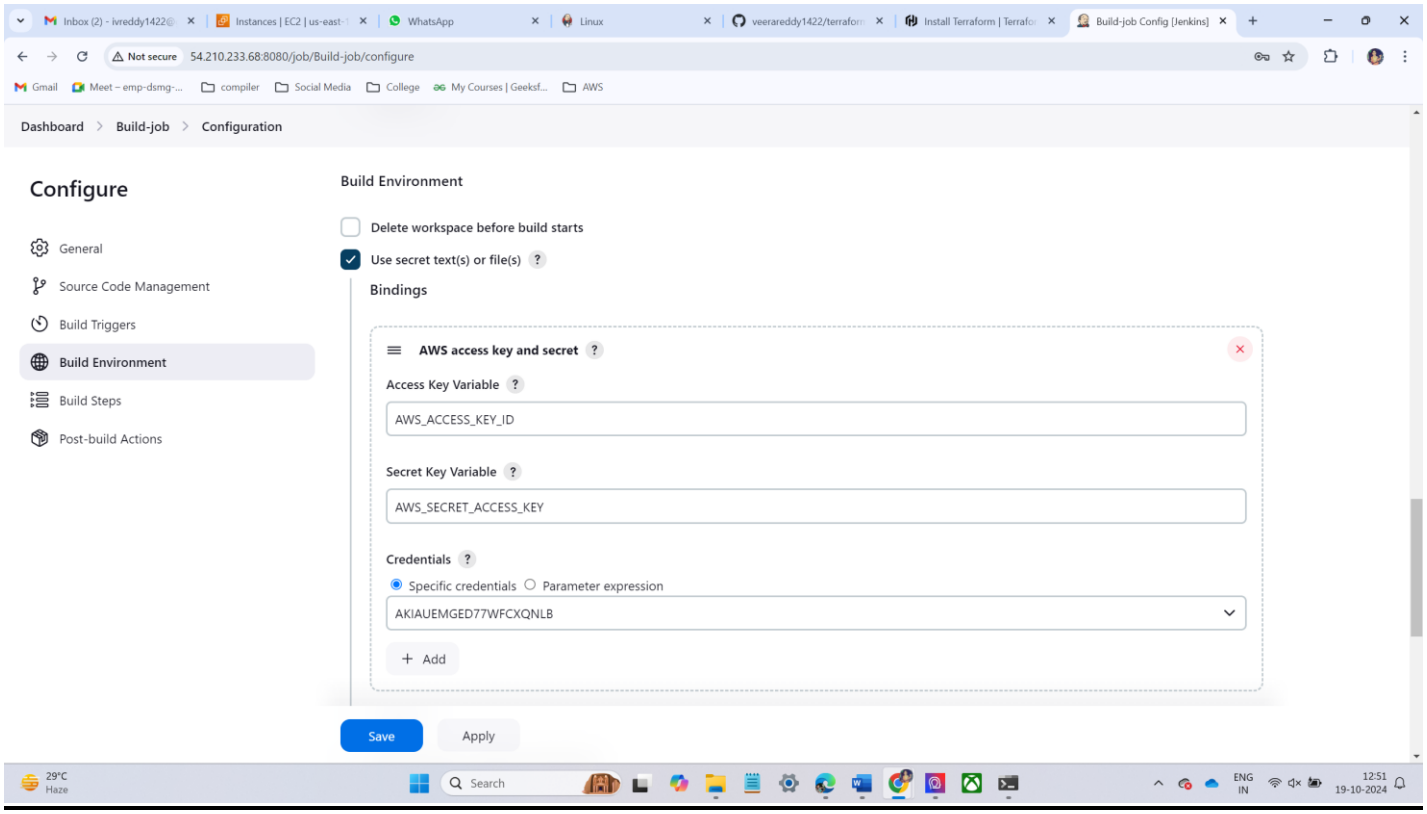
- For Access key and secret key purpose we can install AWS Credentials plugin



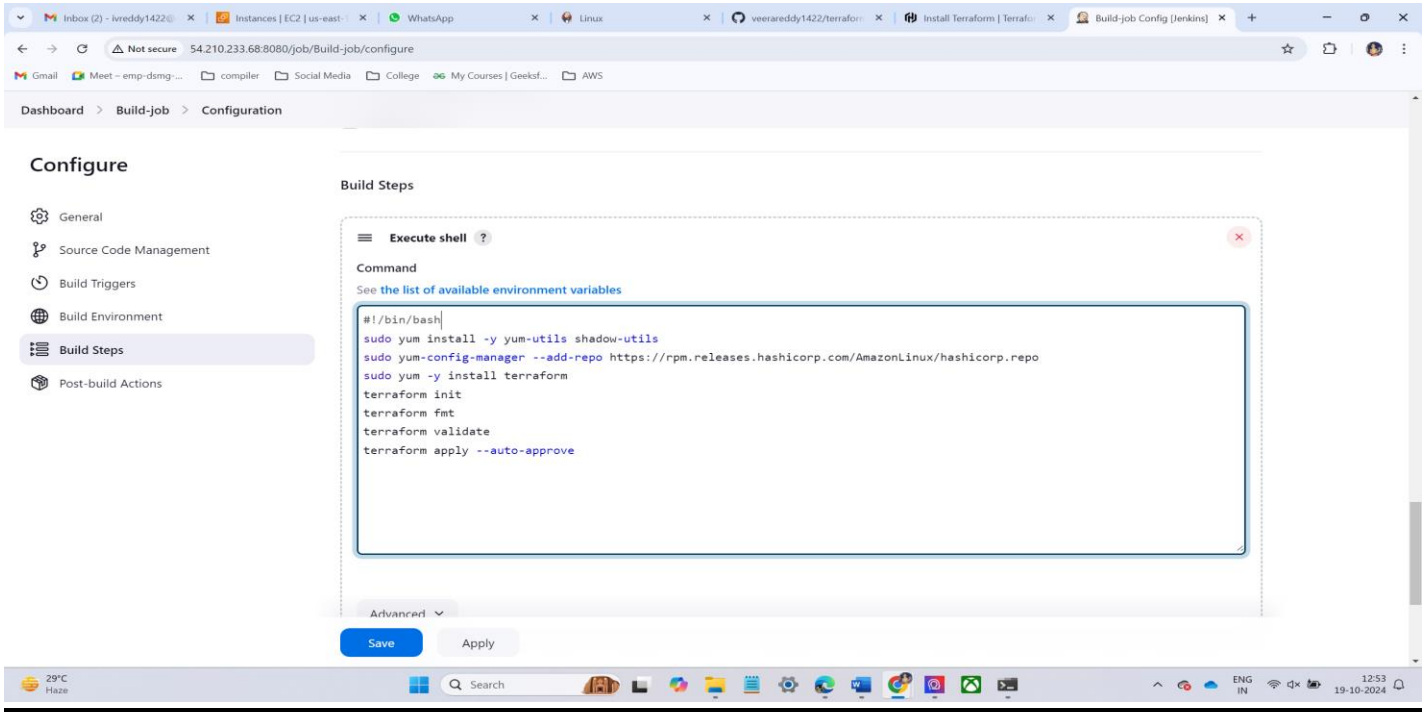
- Create job for build the code
- Click on **"New Item"**.
- Enter a name for your project (e.g., WordPress-Deployment).
- Select **"Freestyle project"** and click **OK**



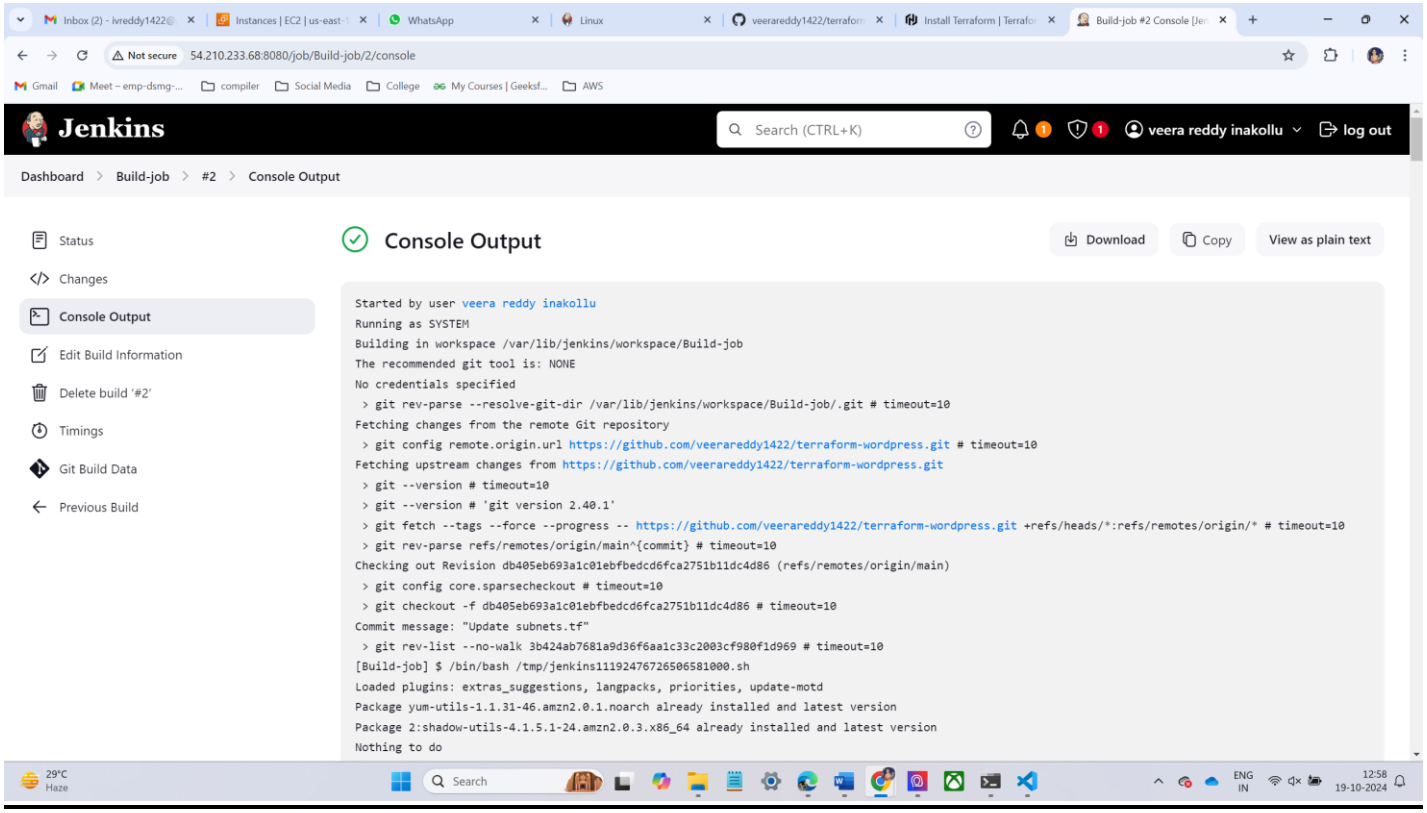
- Under build environment select use secret text or files and enter access key and secret key



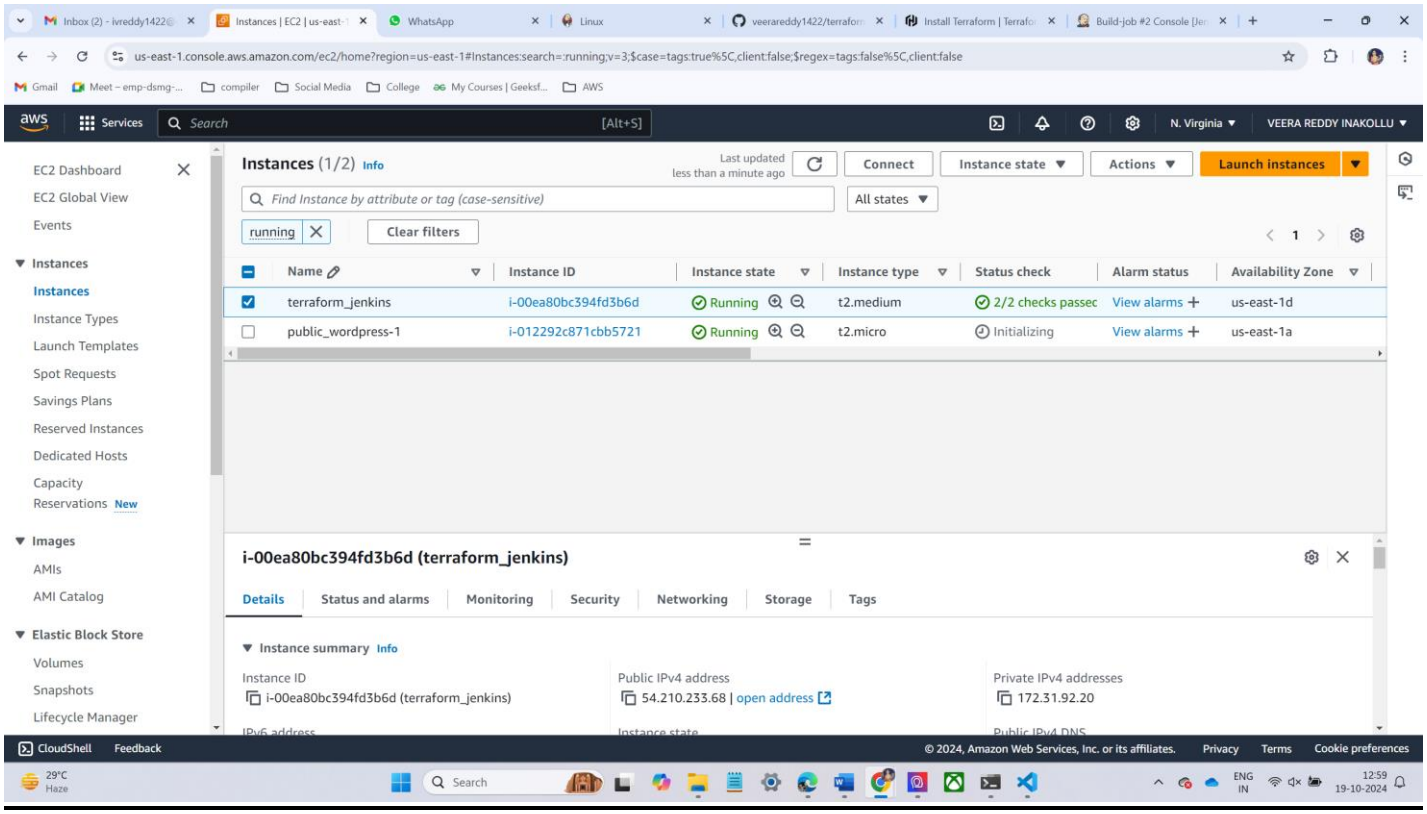
- Configure **Build Steps**
- In the project configuration page, scroll down to the **"Build Steps"** section.
- Select **"Execute shell"**.
- Enter the **bash script**



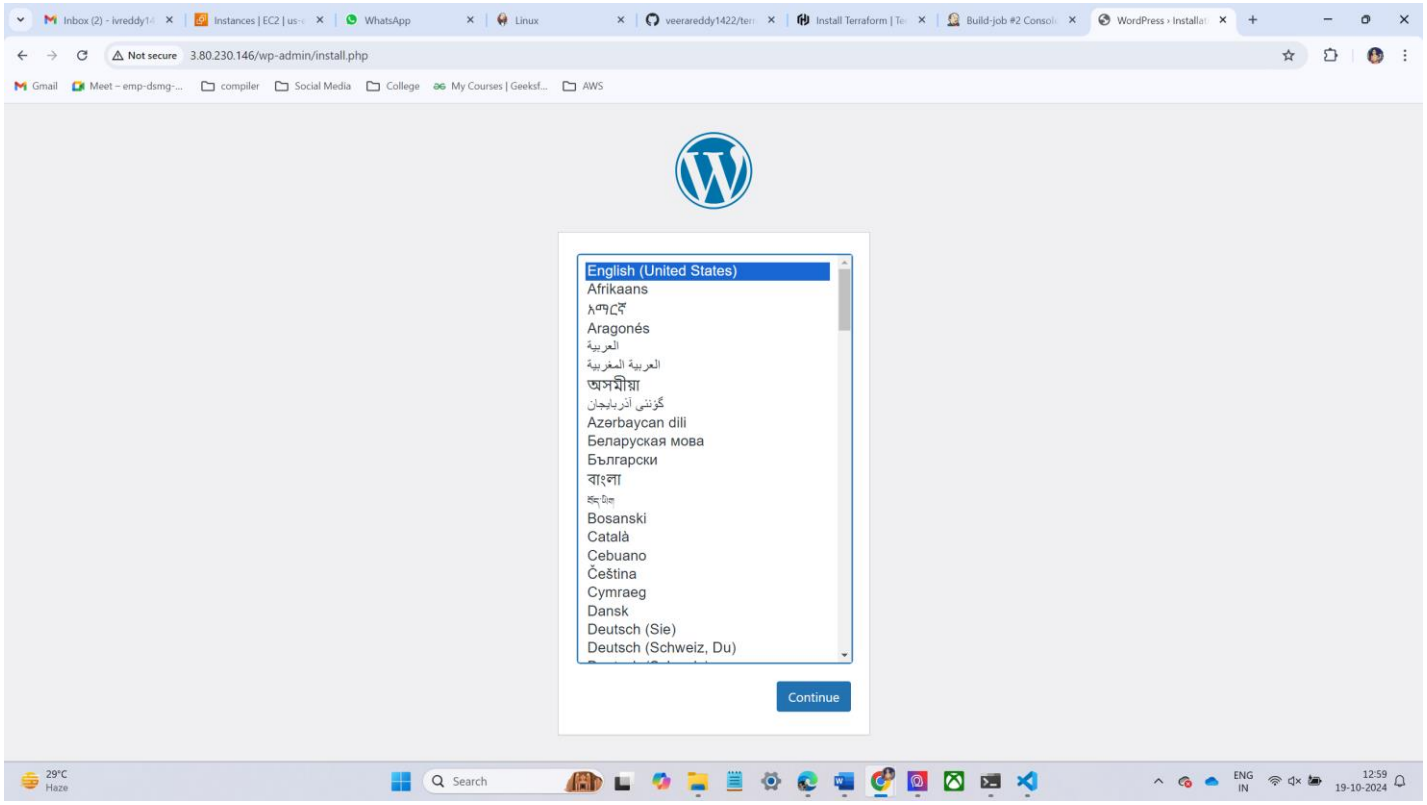
- Check the Build details of Build job



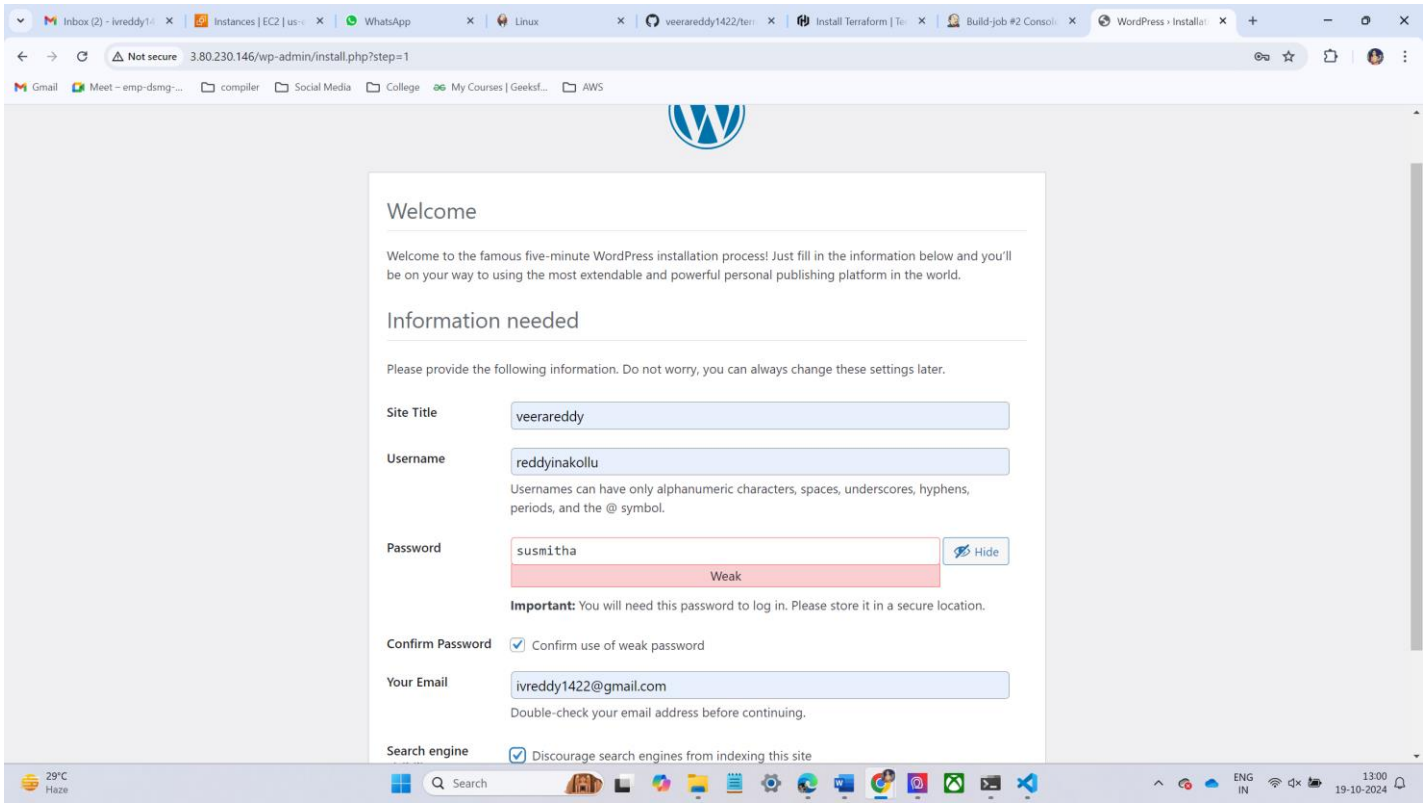
- Now login to aws account and under ec2 console one newly created ec2 instance is displayed.



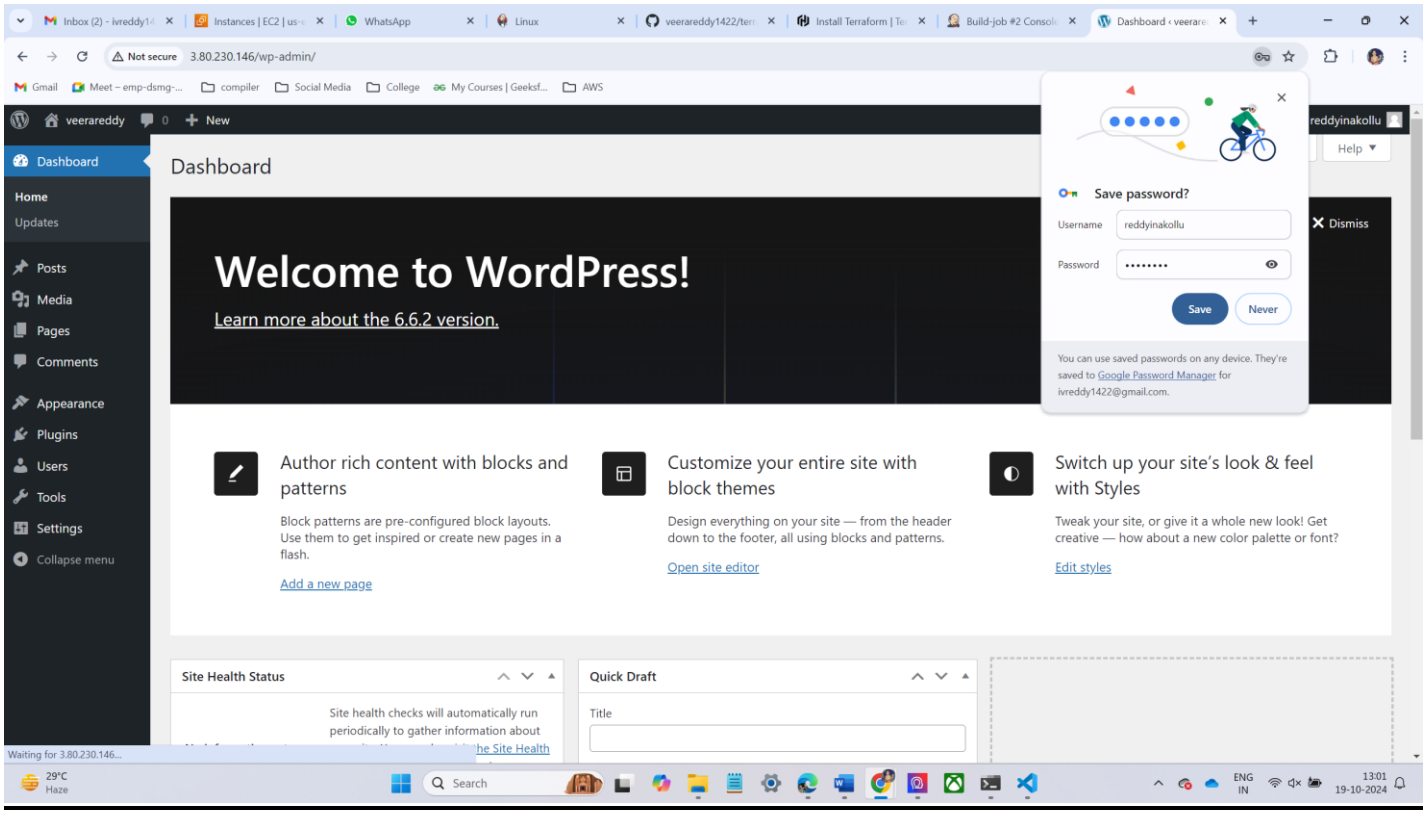
- Copy and paste the new ec2 instance ipv4 address in new tab



- Create the WordPress account

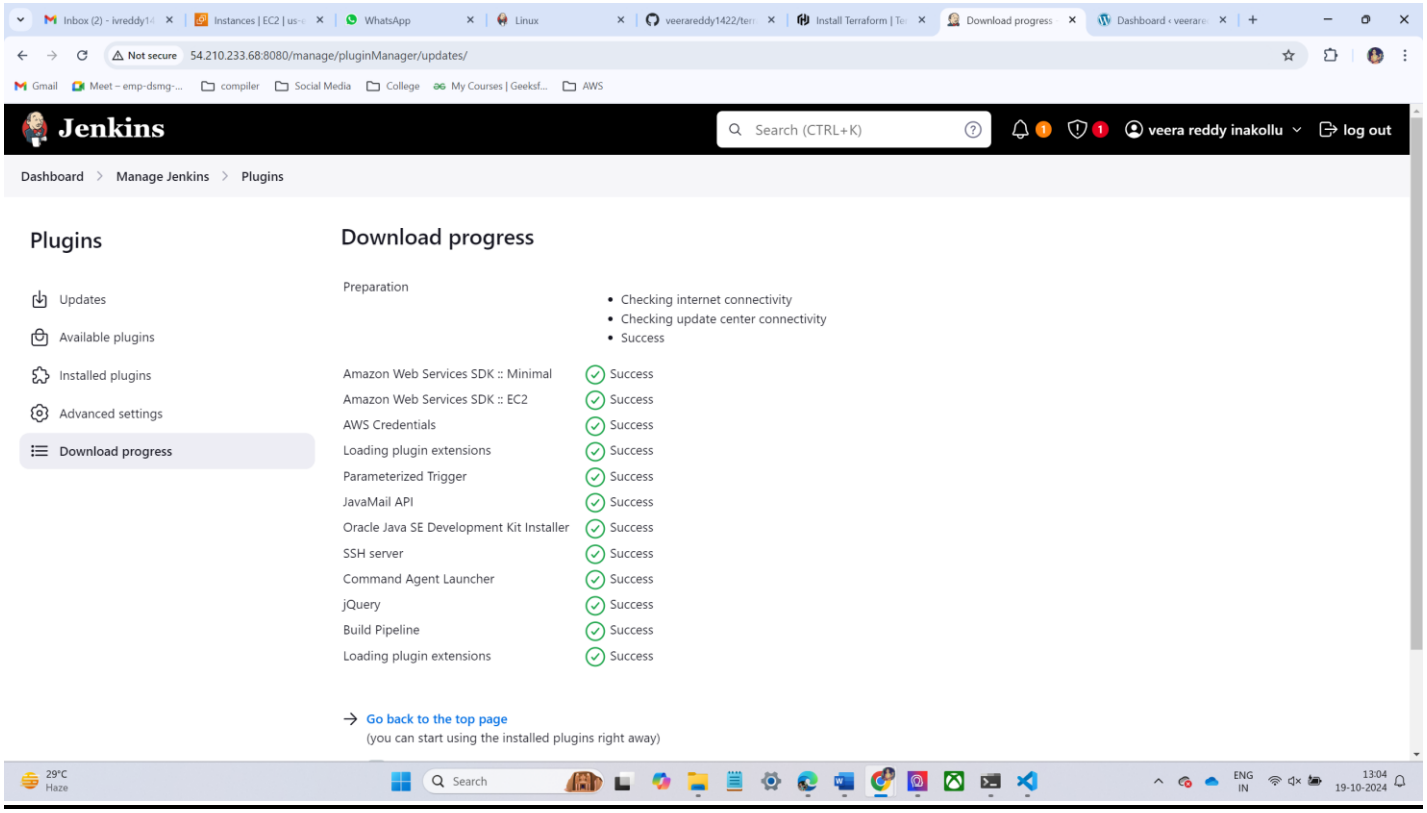


- Login WordPress account
- Successfully deployed the WordPress application using terraform and Jenkins for automation process

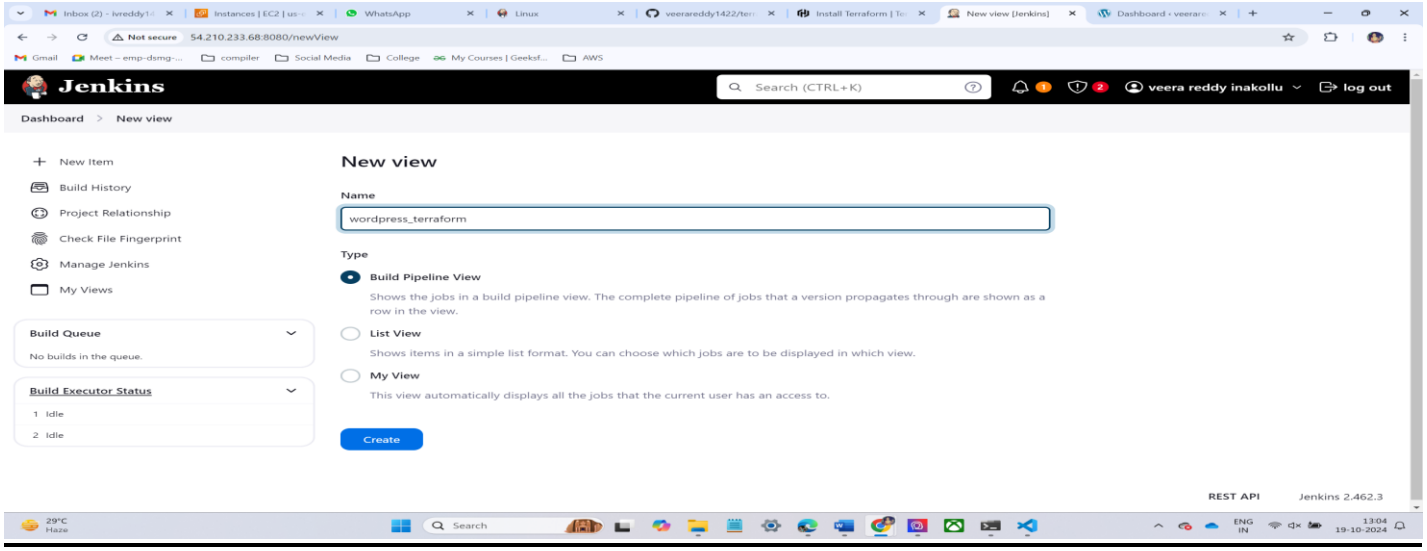


Deploy WordPress web application by using git (clone terraform script which helps to deploy WordPress web application), jenkins (in execute shell install terraform, init, fmt, validate and apply with automatic command as terraform apply --auto-approve) and terraform and create jenkins pipeline and add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

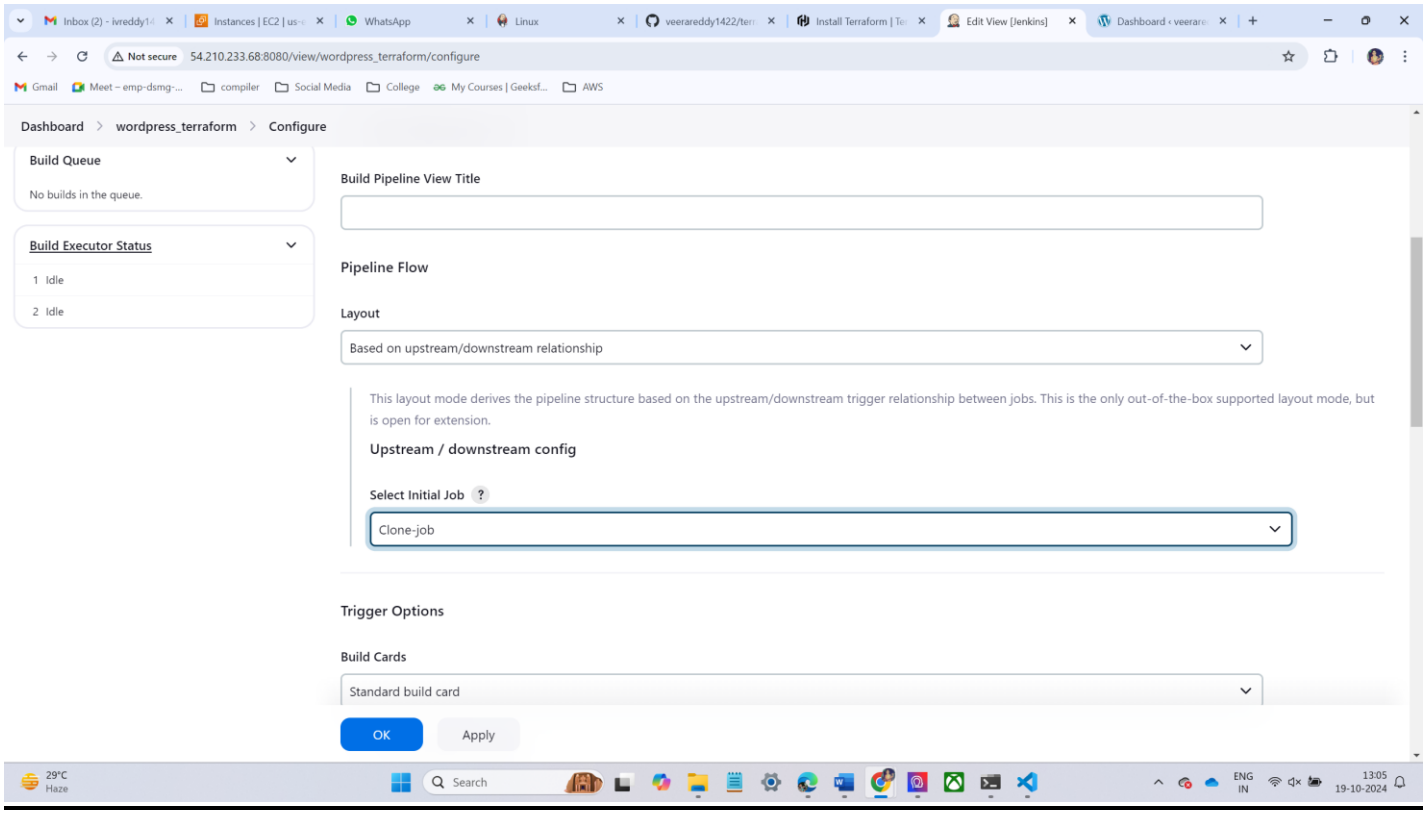
- Creating Jenkins pipeline for continues process for previous method
- For the creation of Jenkins pipeline we need to install build pipeline plugin.



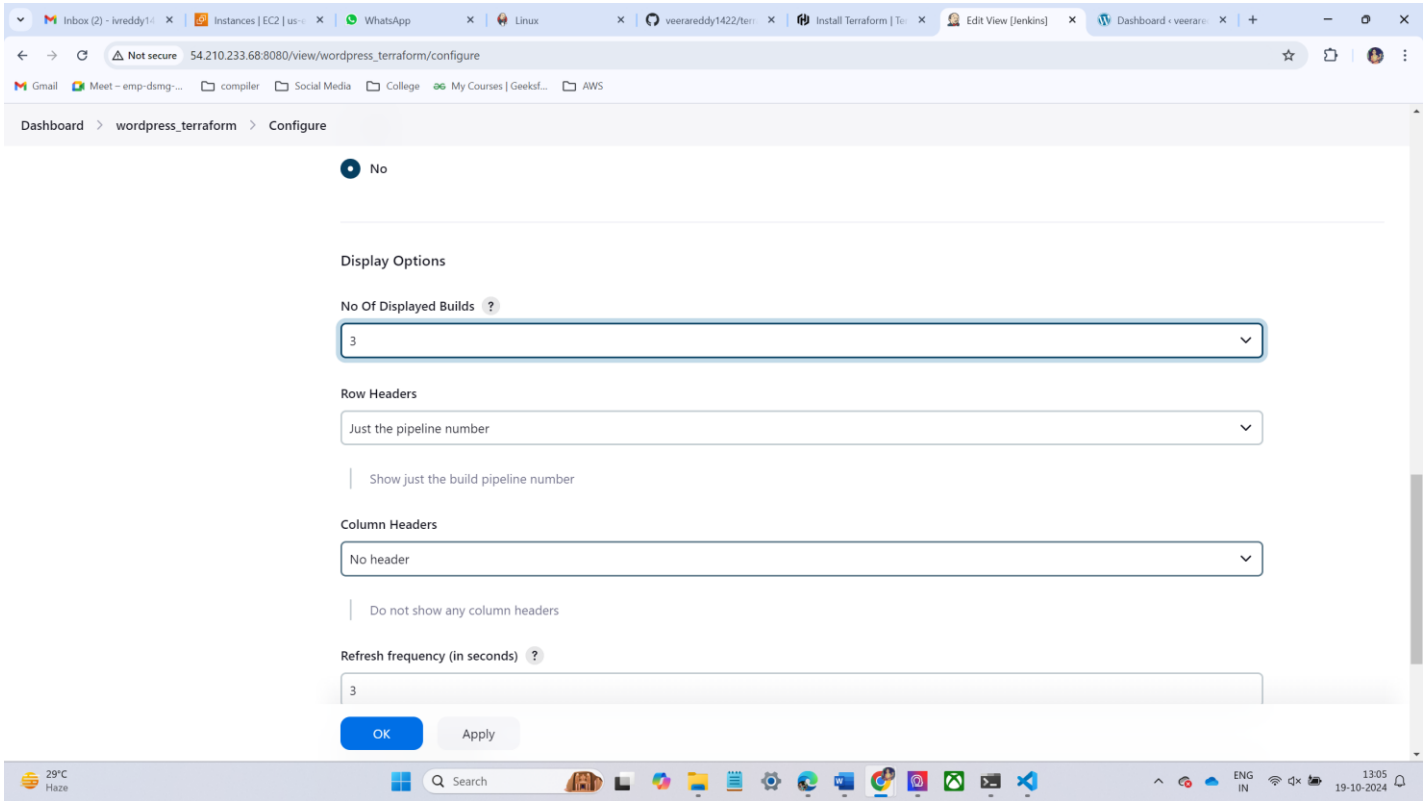
- Goto dashboard under this section select + icon
- Now enter pipeline name and select the build pipeline view



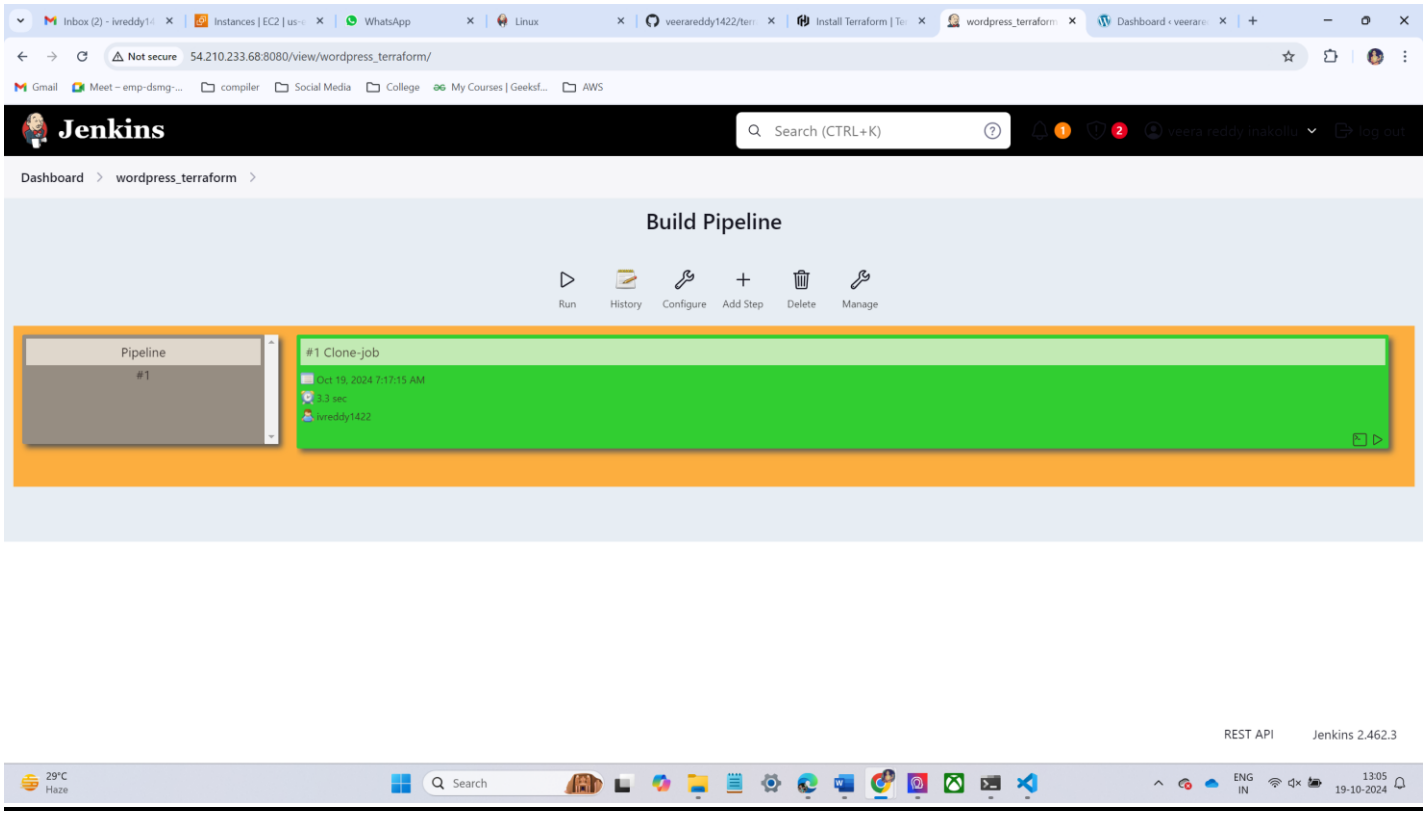
- Now configure the pipeline
- Under pipeline flow choose your initial job



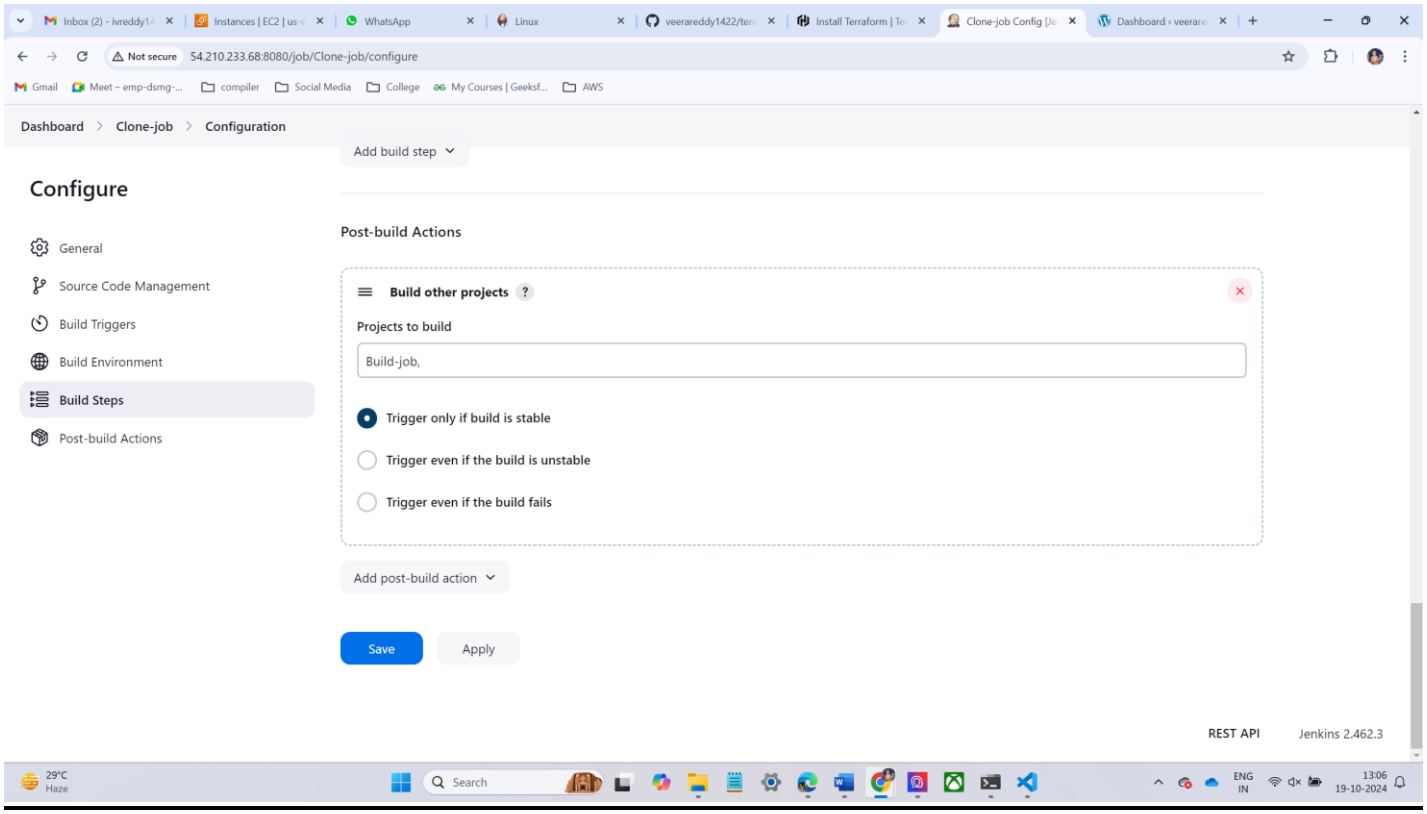
- Under display options choose No 0f Displayed Builds (your wish)



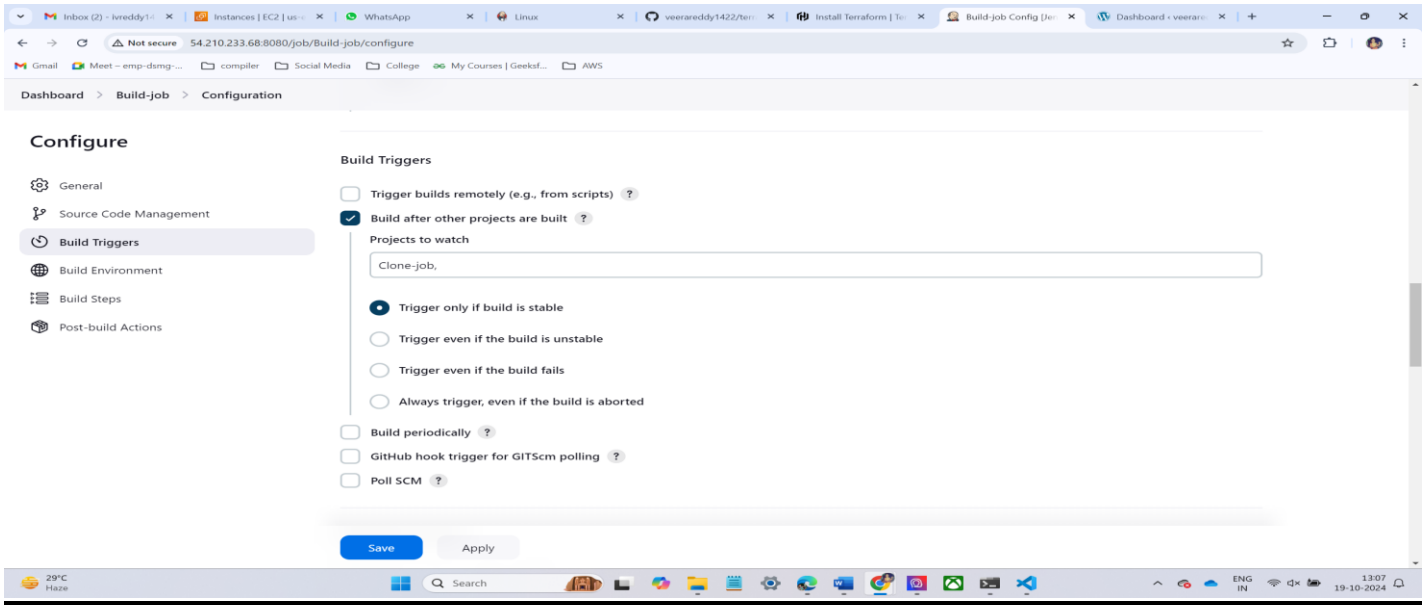
- Successfully creating pipeline for clone job only.



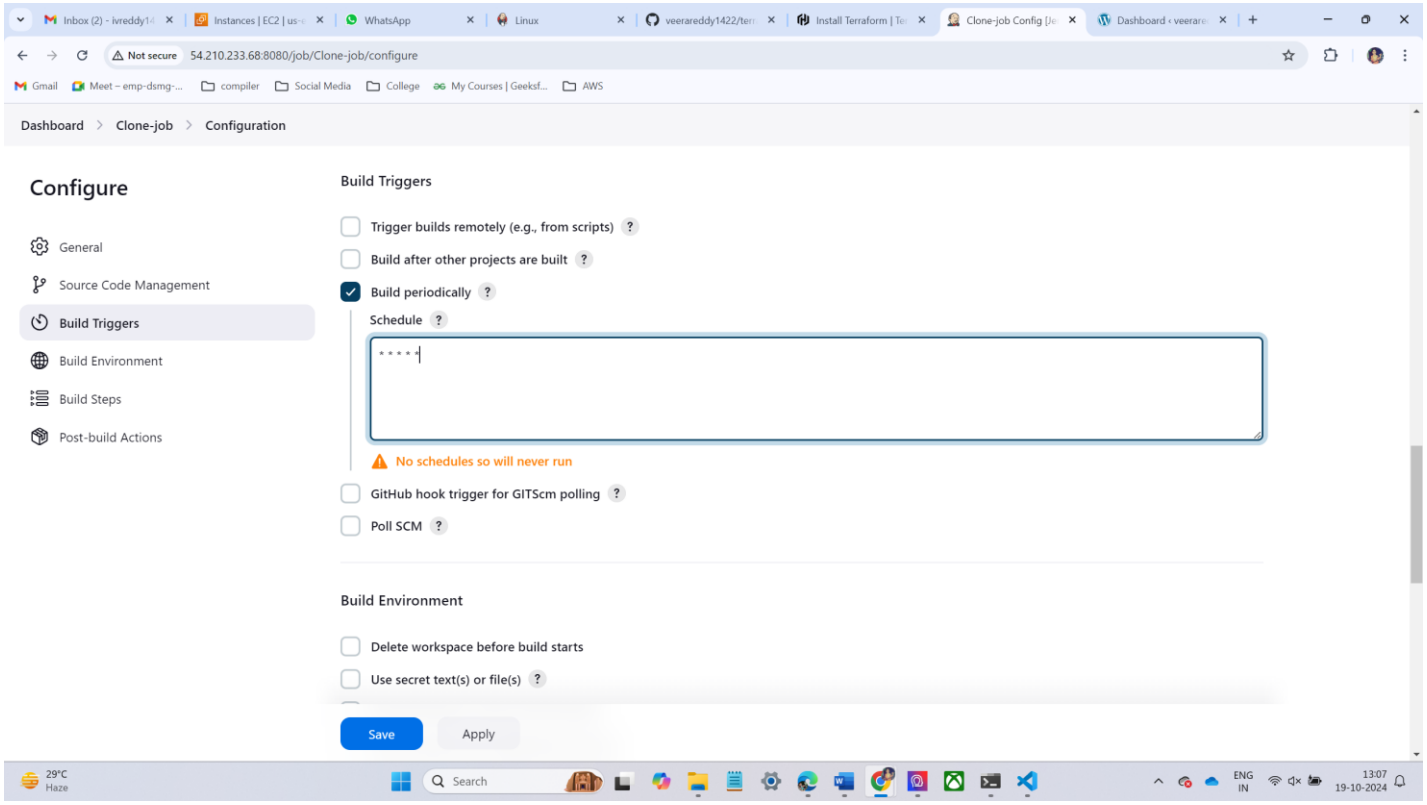
- Now create the pipeline for all jobs
- For creating pipeline configure the clone job
- In clone job under post-build action select the build other projects
- Select build job



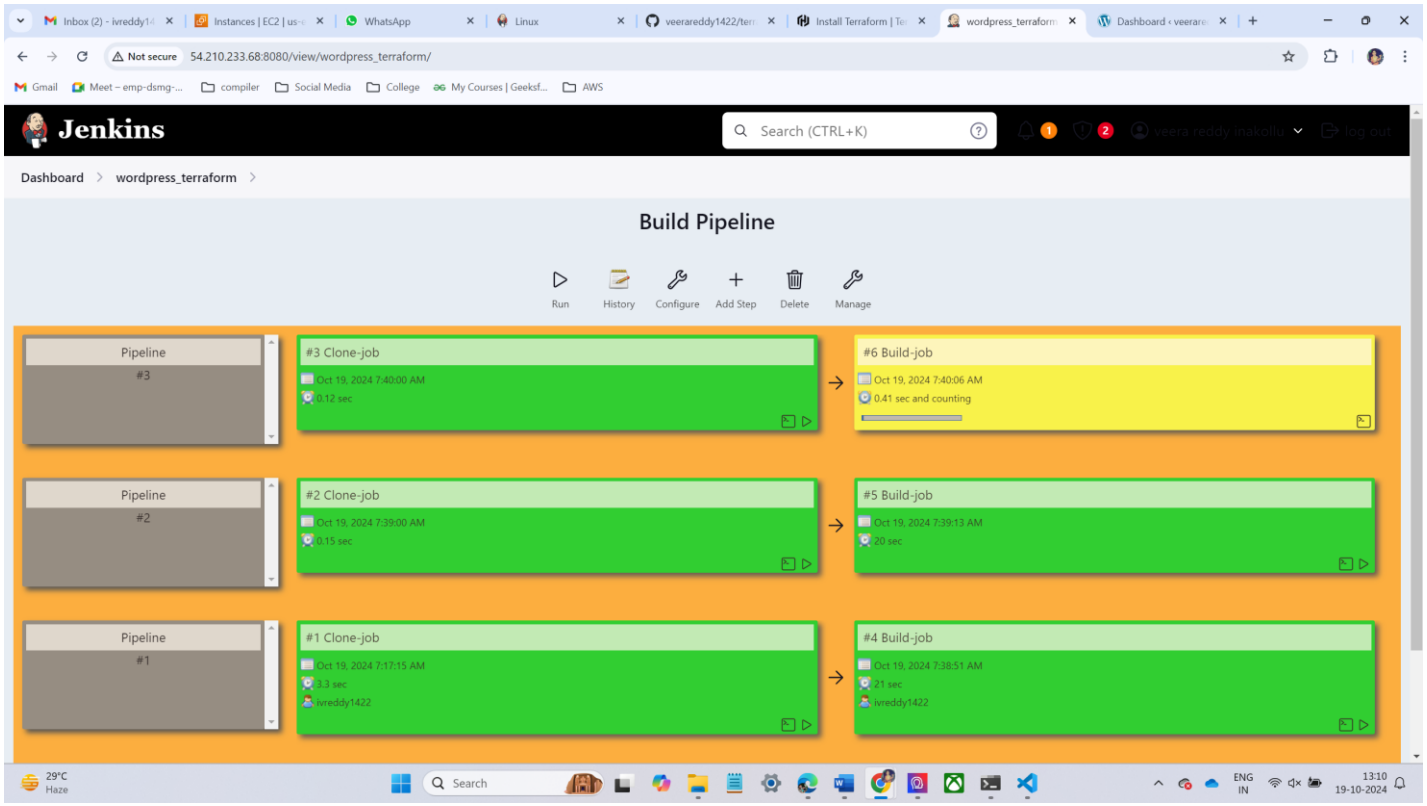
- Configure the build job
- Under build triggers select build after other projects are built
- Select clone job



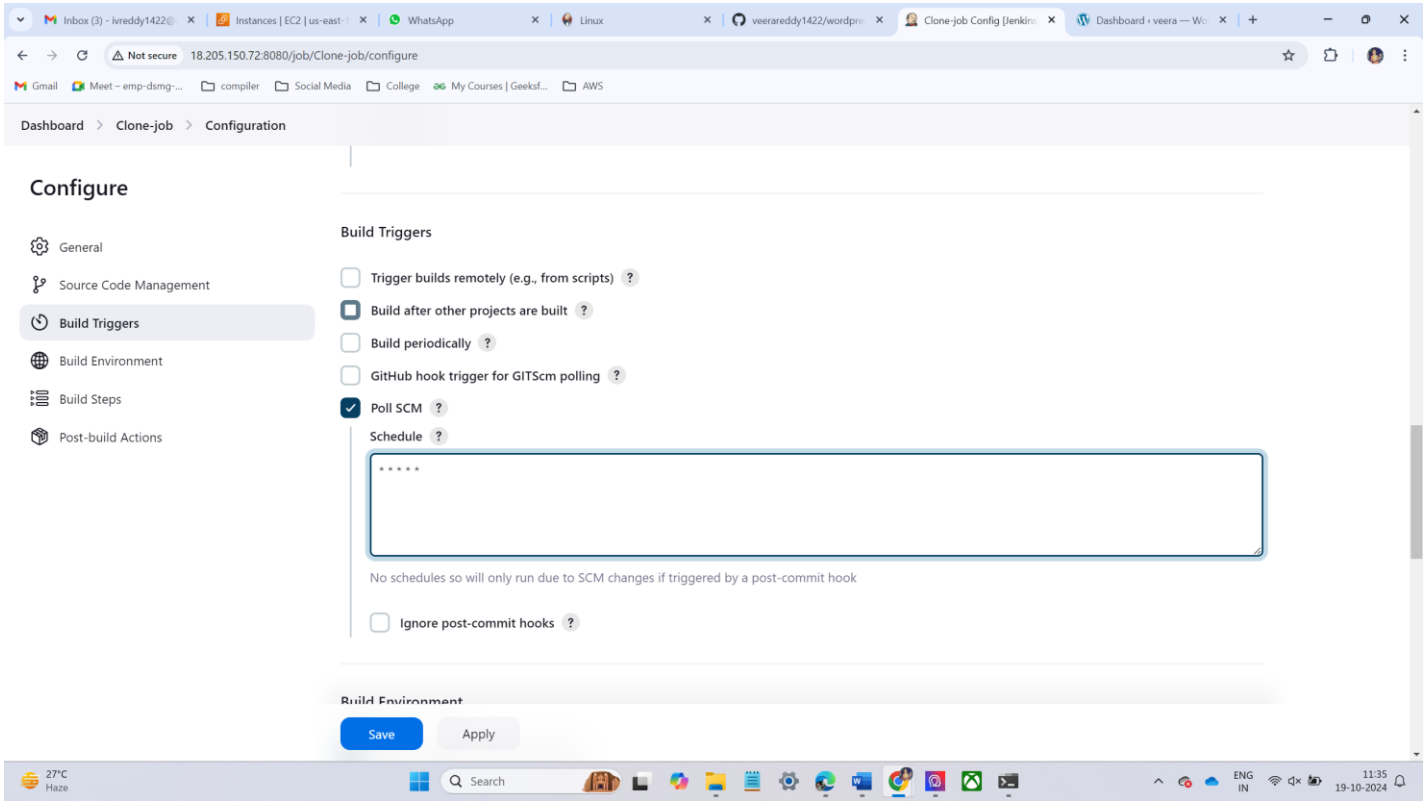
- Now apply build periodically in clone job



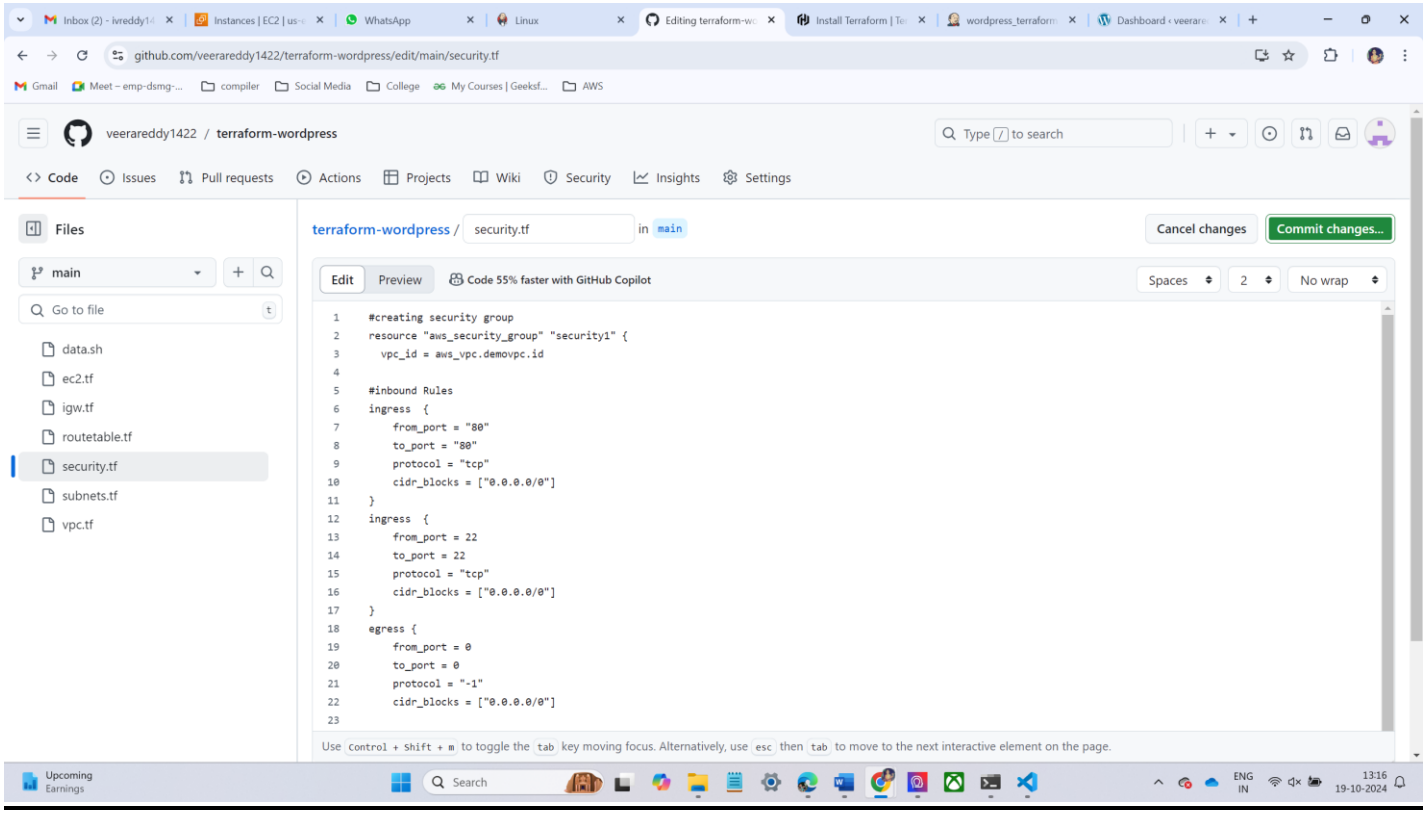
- Observe the build pipeline



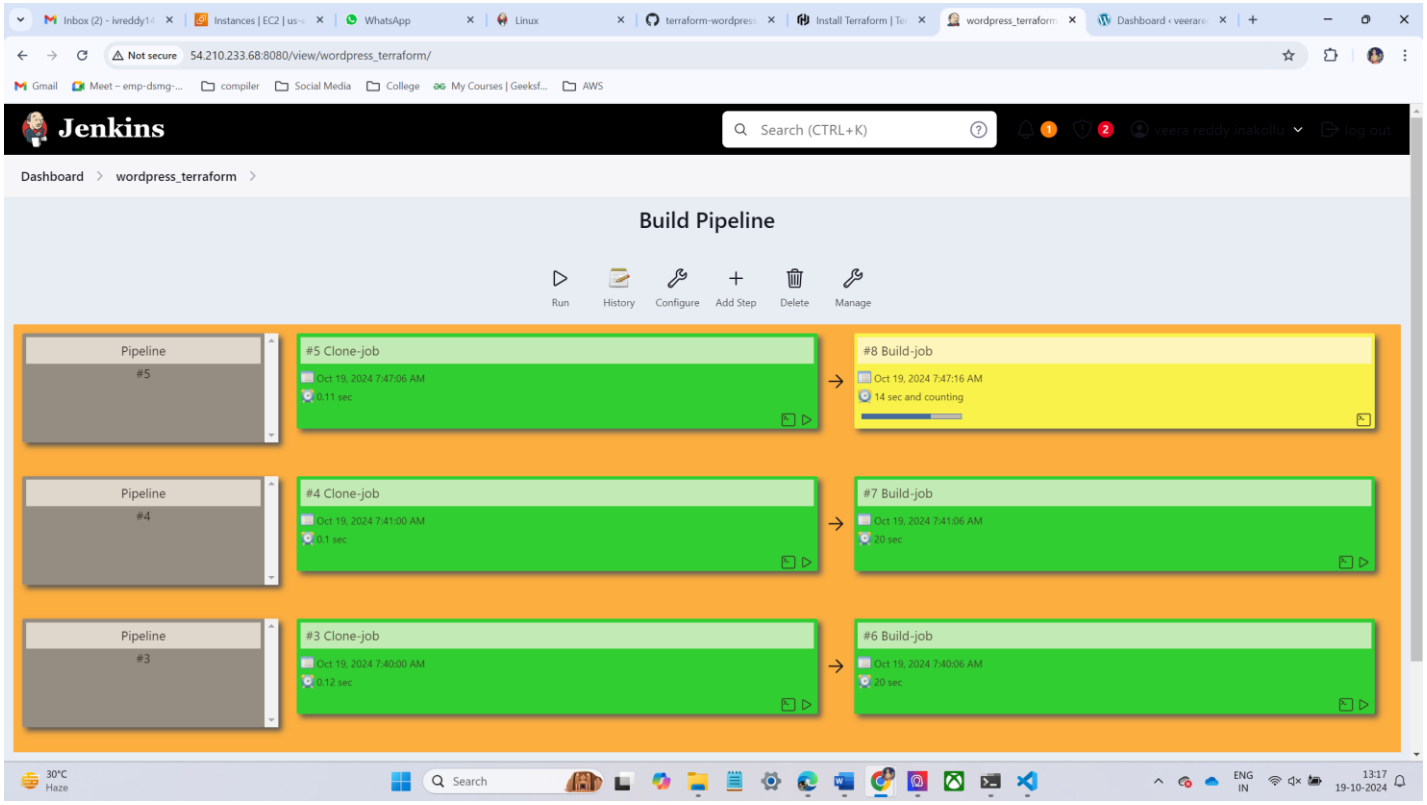
- Now apply the Poll SCM



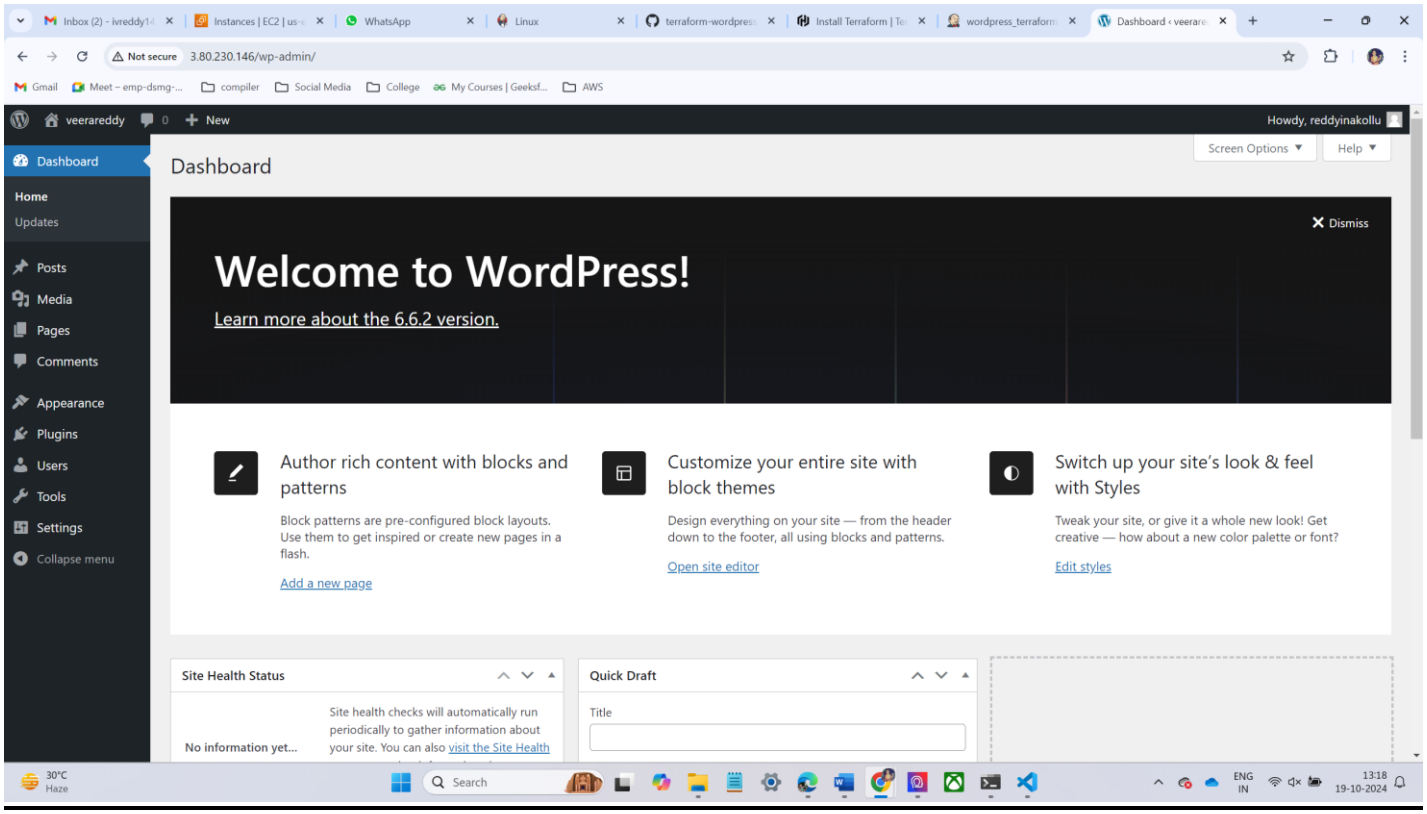
- We can the change the code in git hub



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

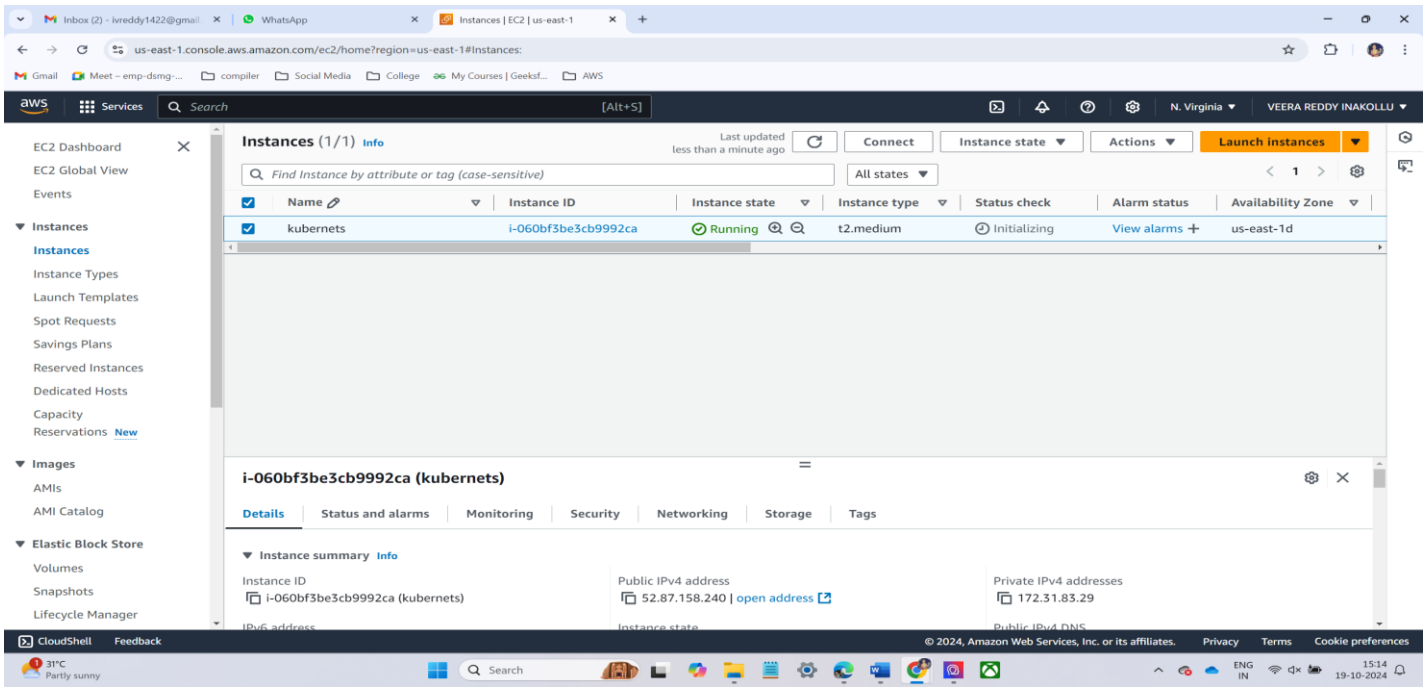


- Successfully host the WordPress application using Jenkins pipeline using build periodically and poll scm.

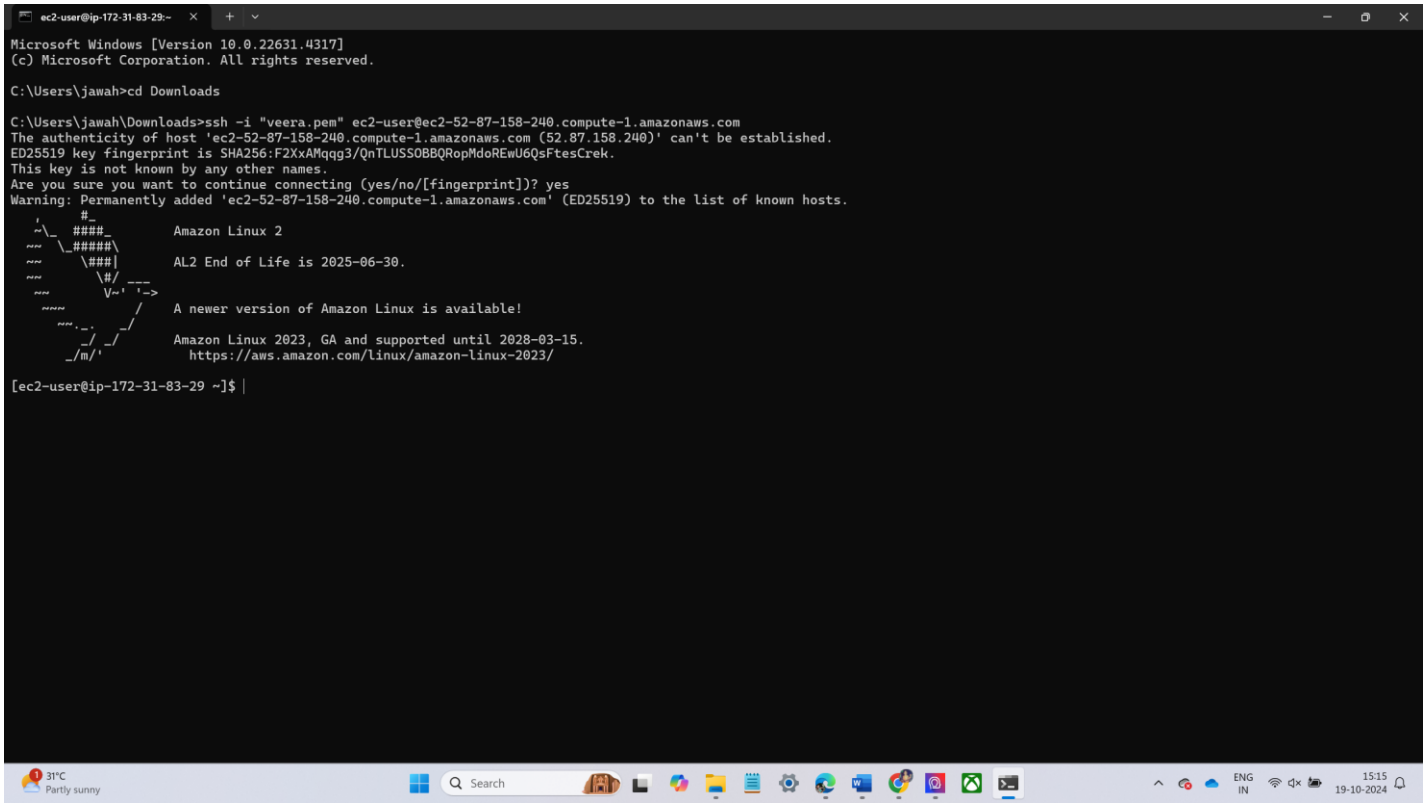


Deploy WordPress web application by using k8's (Declarative manifest method) with the help of docker hub images?

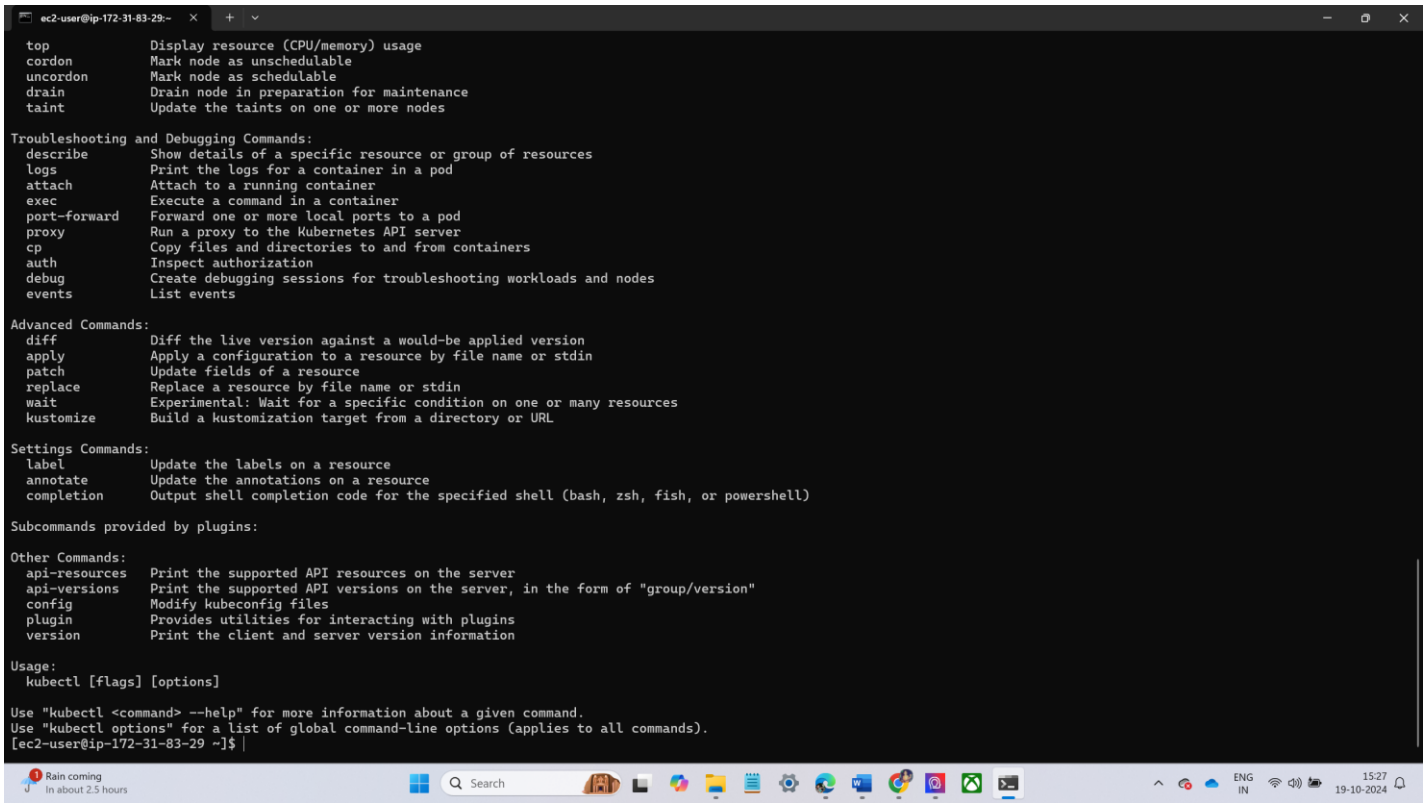
- Login to AWS account and launch ec2 instance



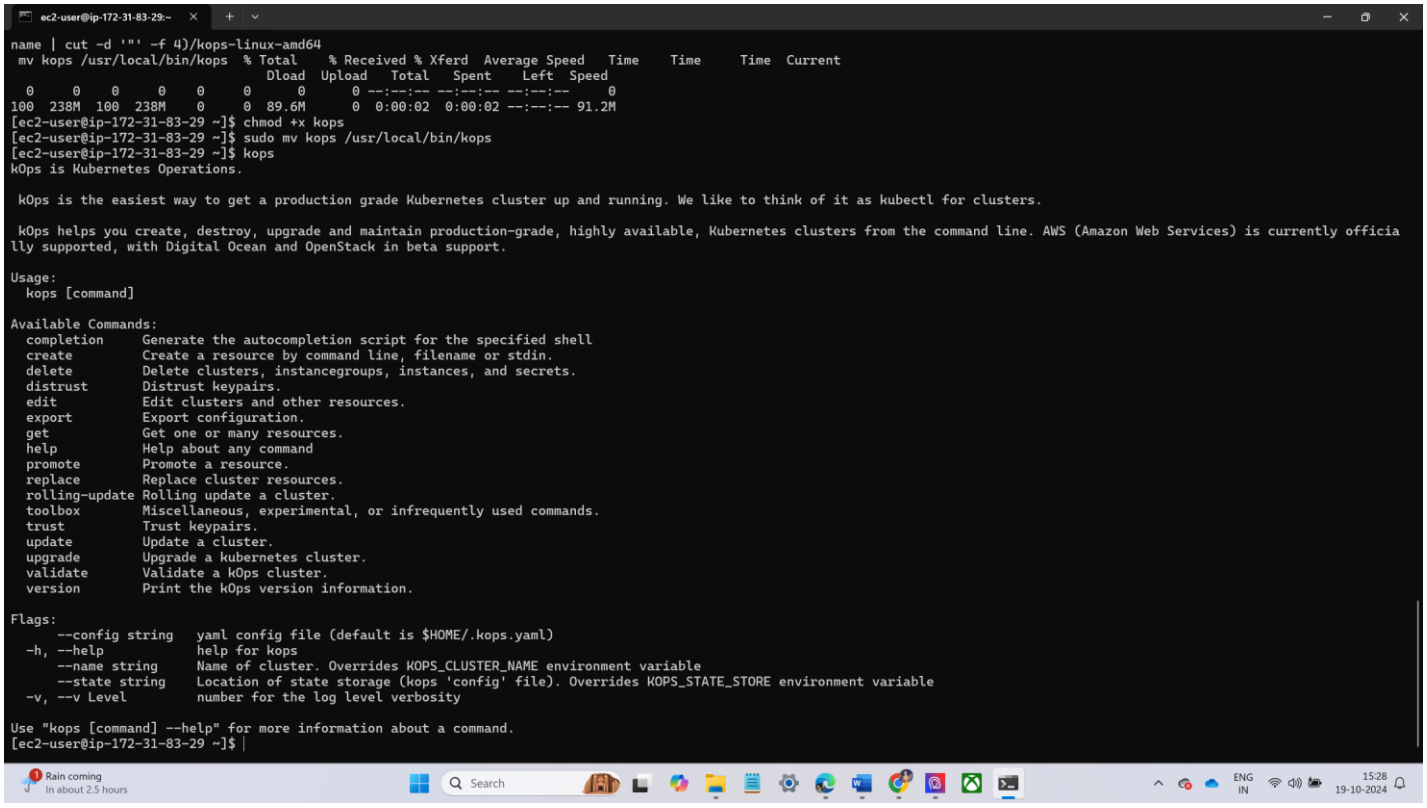
- Now connect ec2 instance to the command prompt



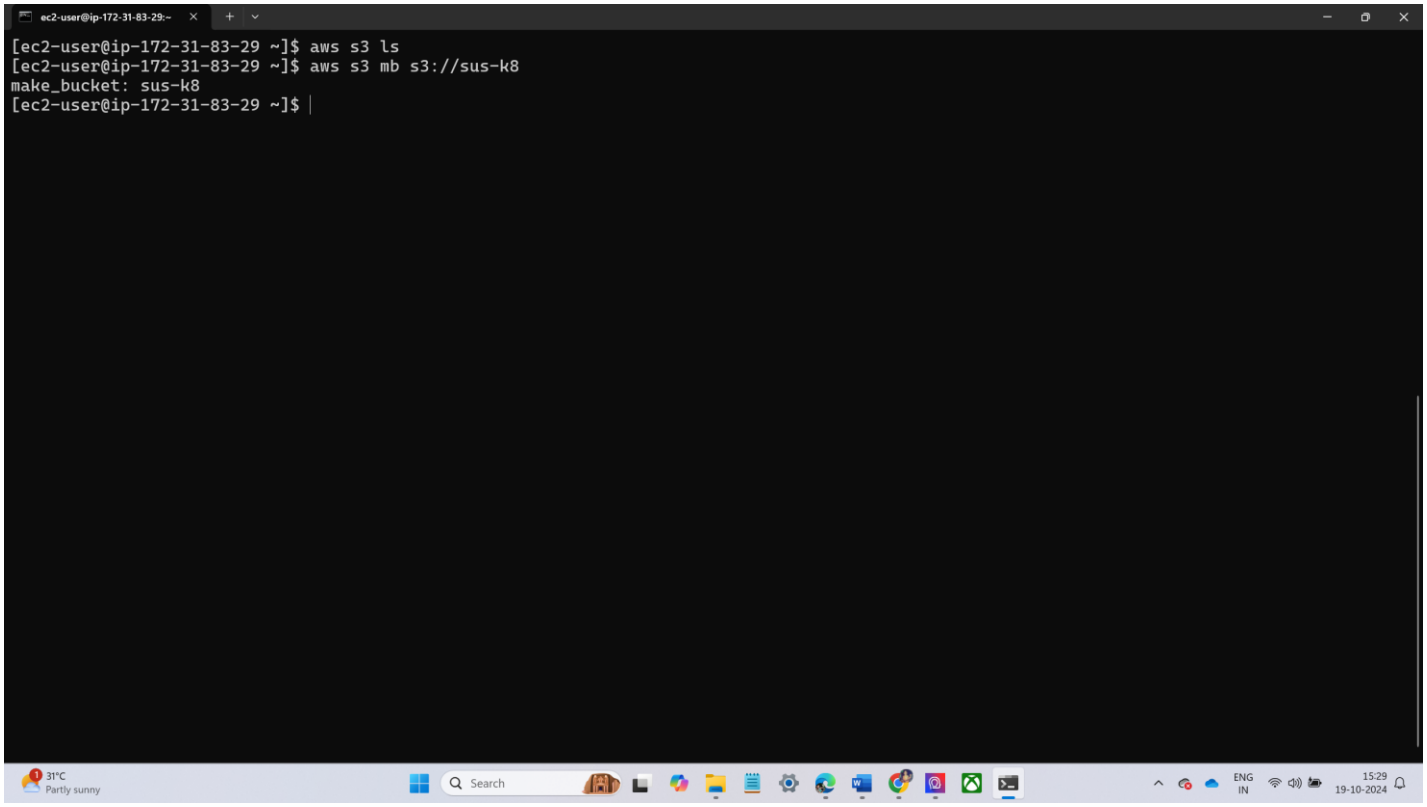
- Now install kubectl by using google chrome



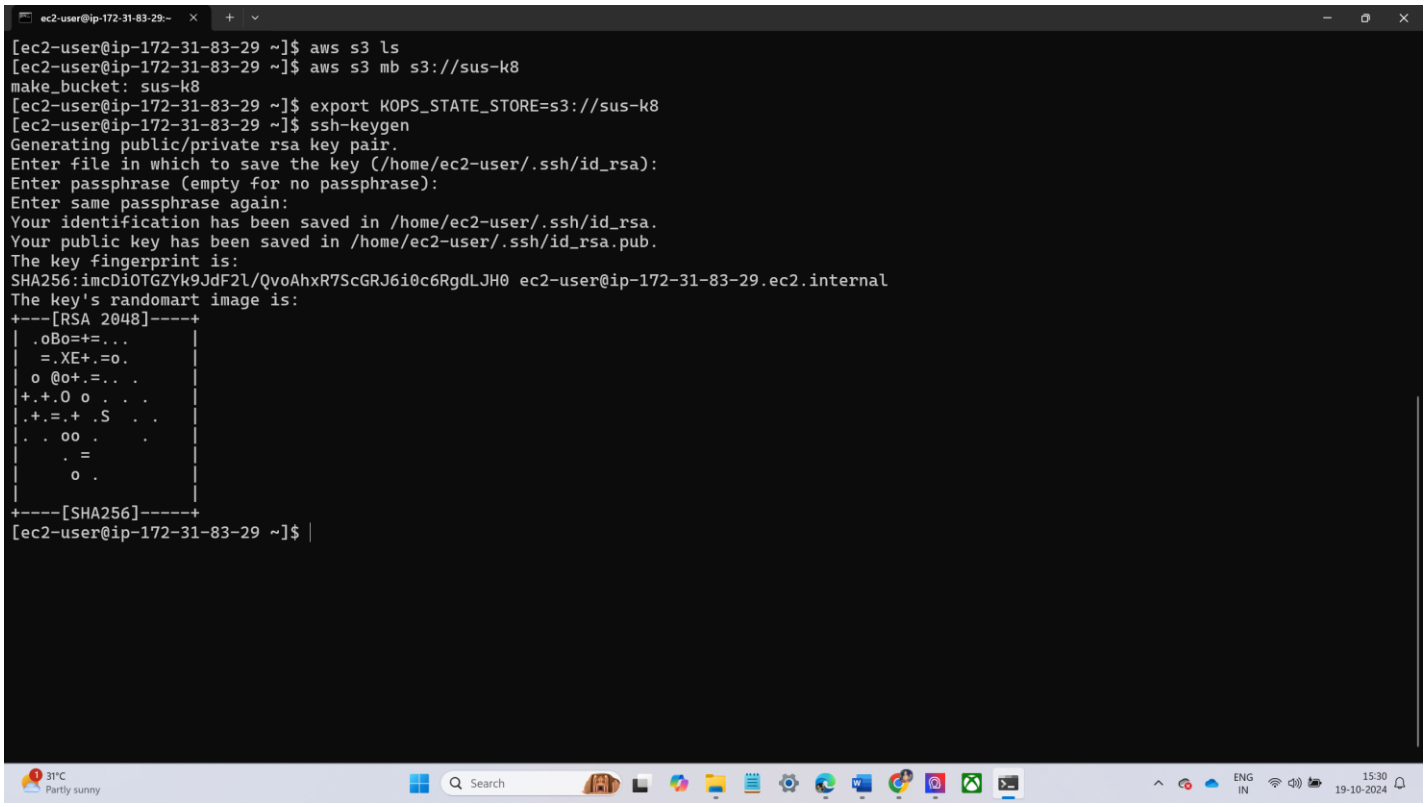
- After installing kubectl than install kops



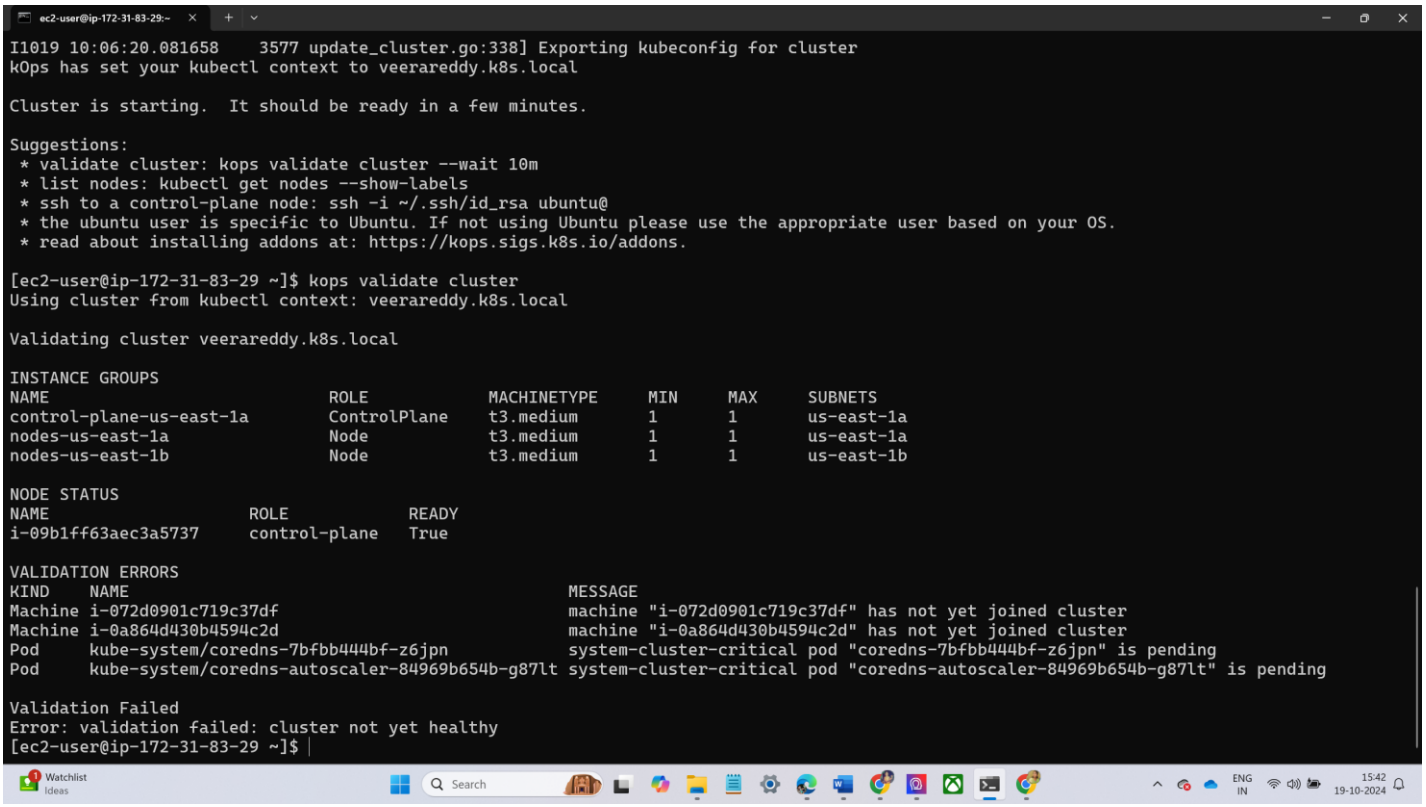
- Now check AWS configuration and create a s3 bucket by using command aws s3 mb s3://bucketname



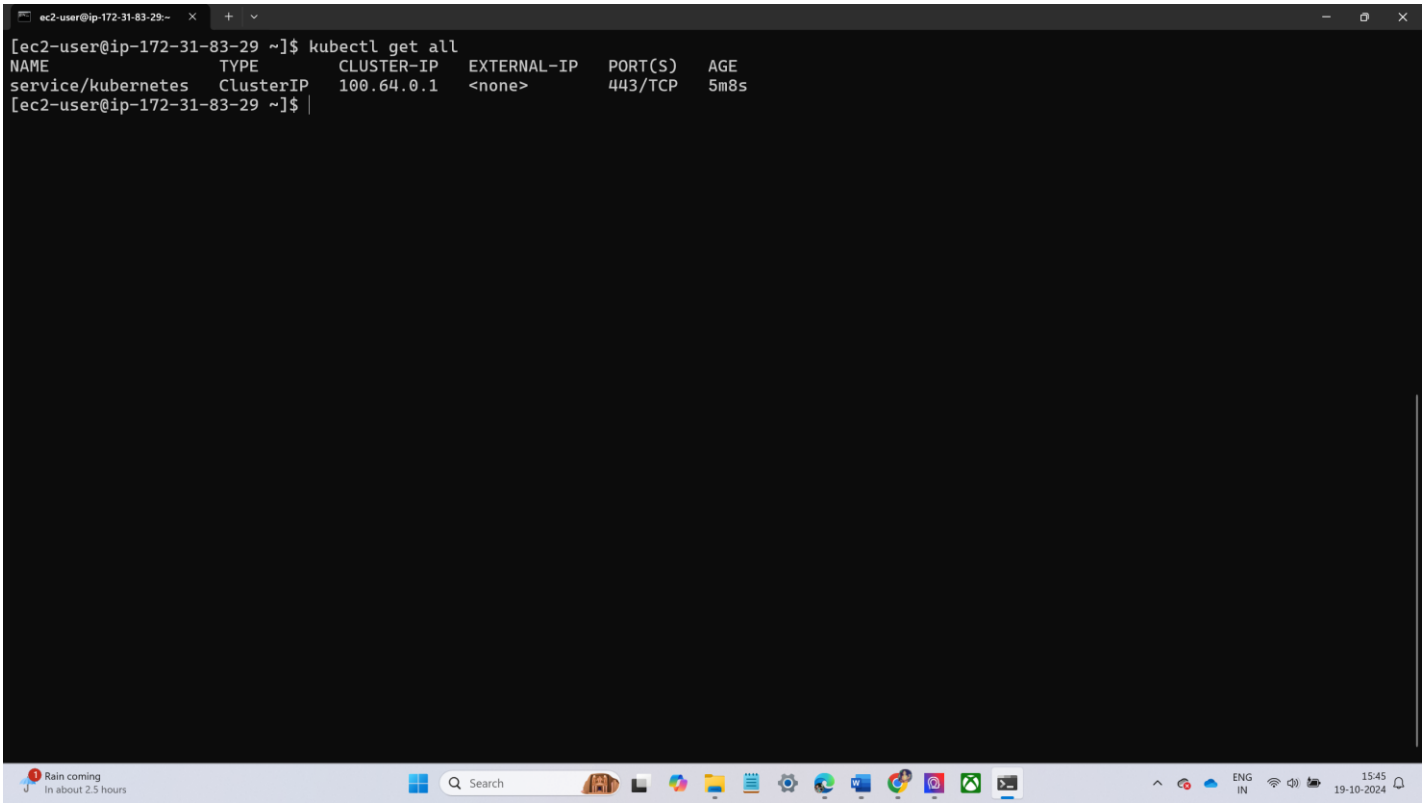
- Now export the bucket to the kops and give ssh keygen permissions



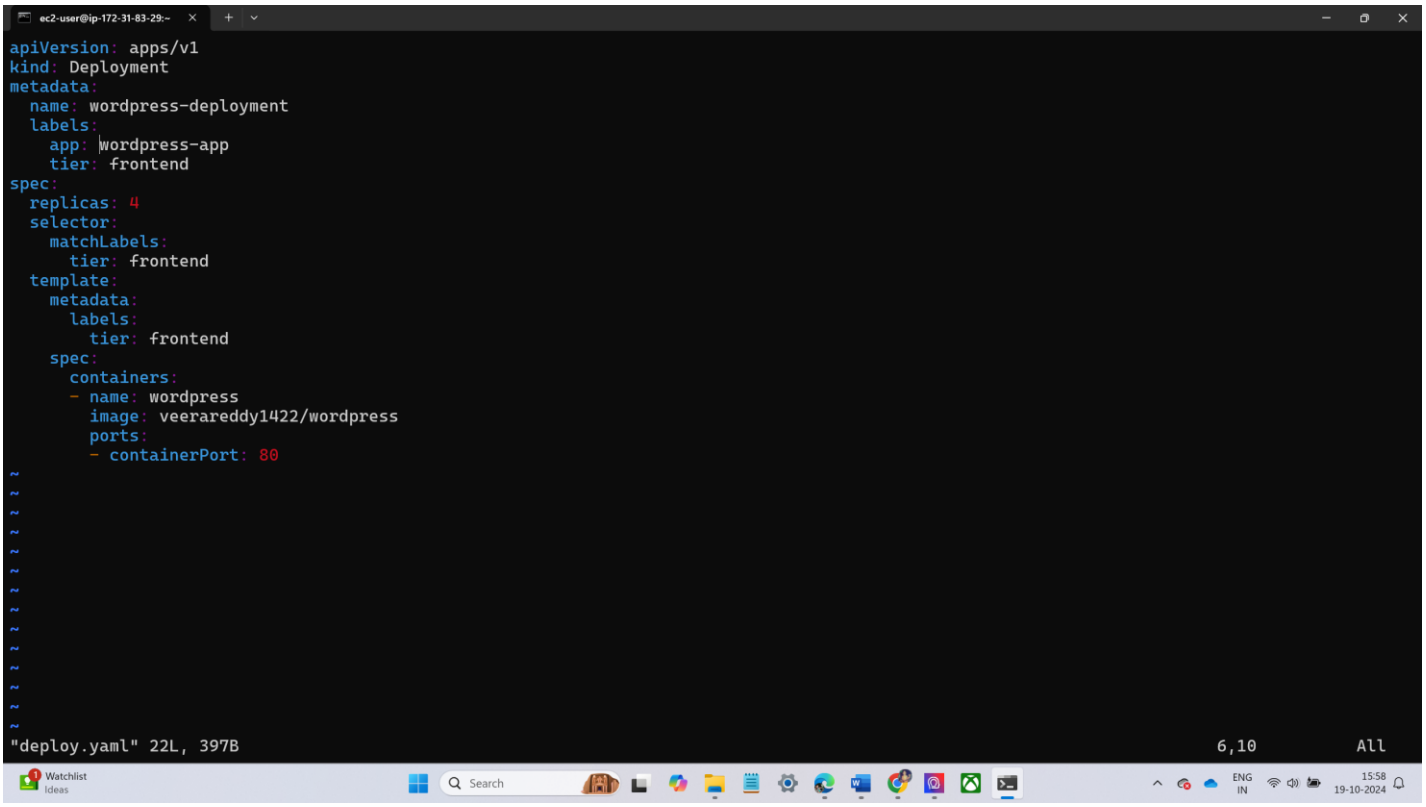
- Now create cluster by using command kubectl create cluster -- name <cluster-name> --state <bucket-name> --regions <name of regions you created> -- nodes < number of nodes> --yes



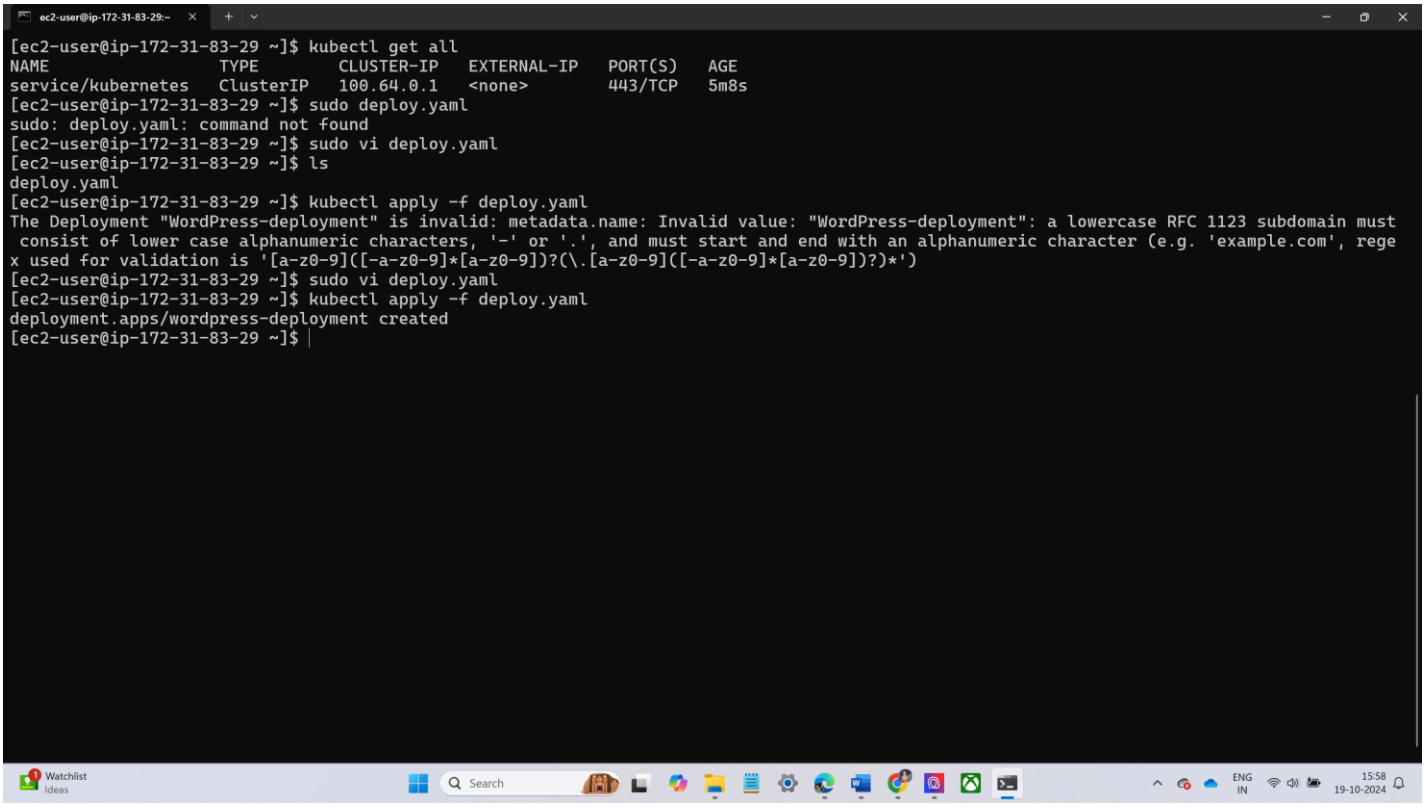
- Now check the created cluster list



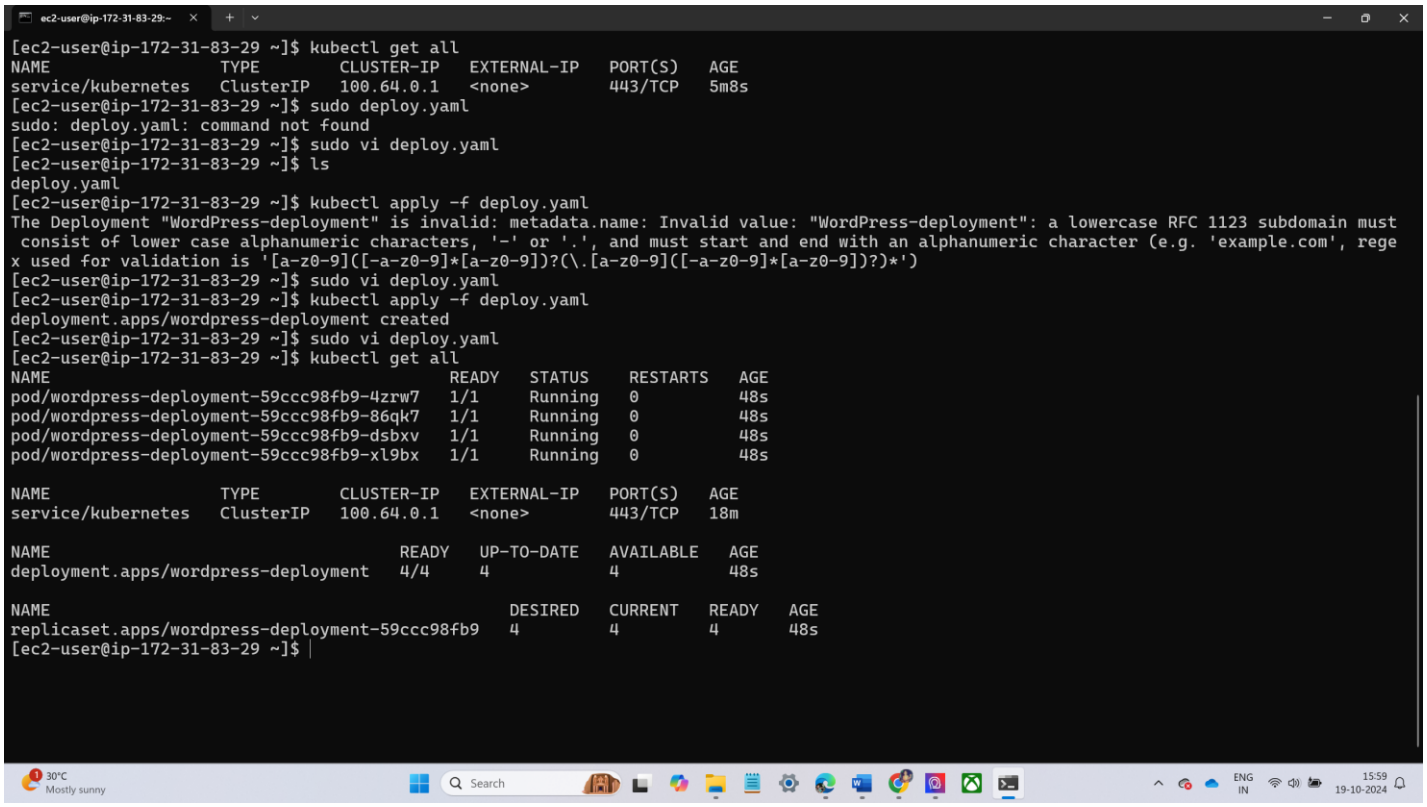
- Now create the file for deployment and write script (HELM)



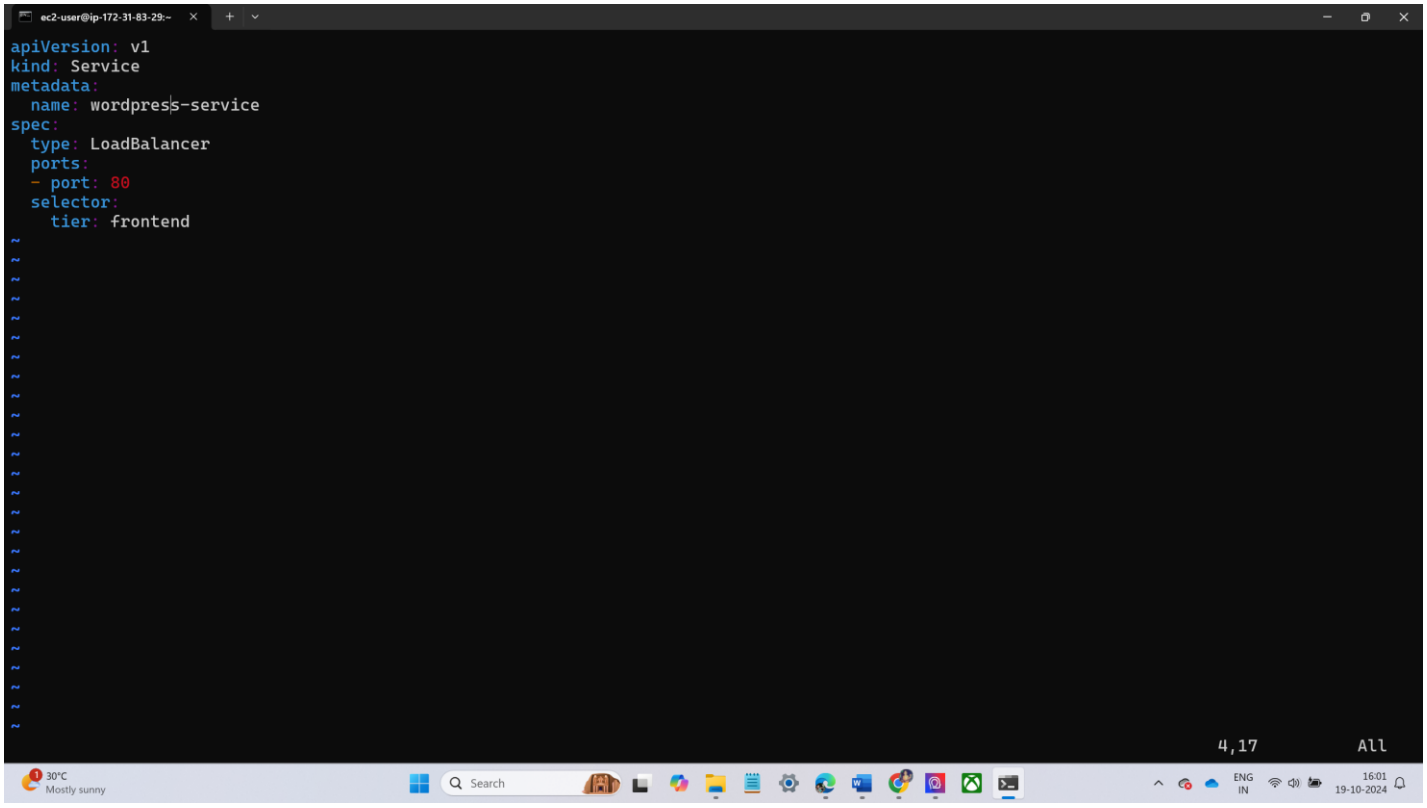
- Now create deployment pods by using command kubectl apply -f <ymal file name>



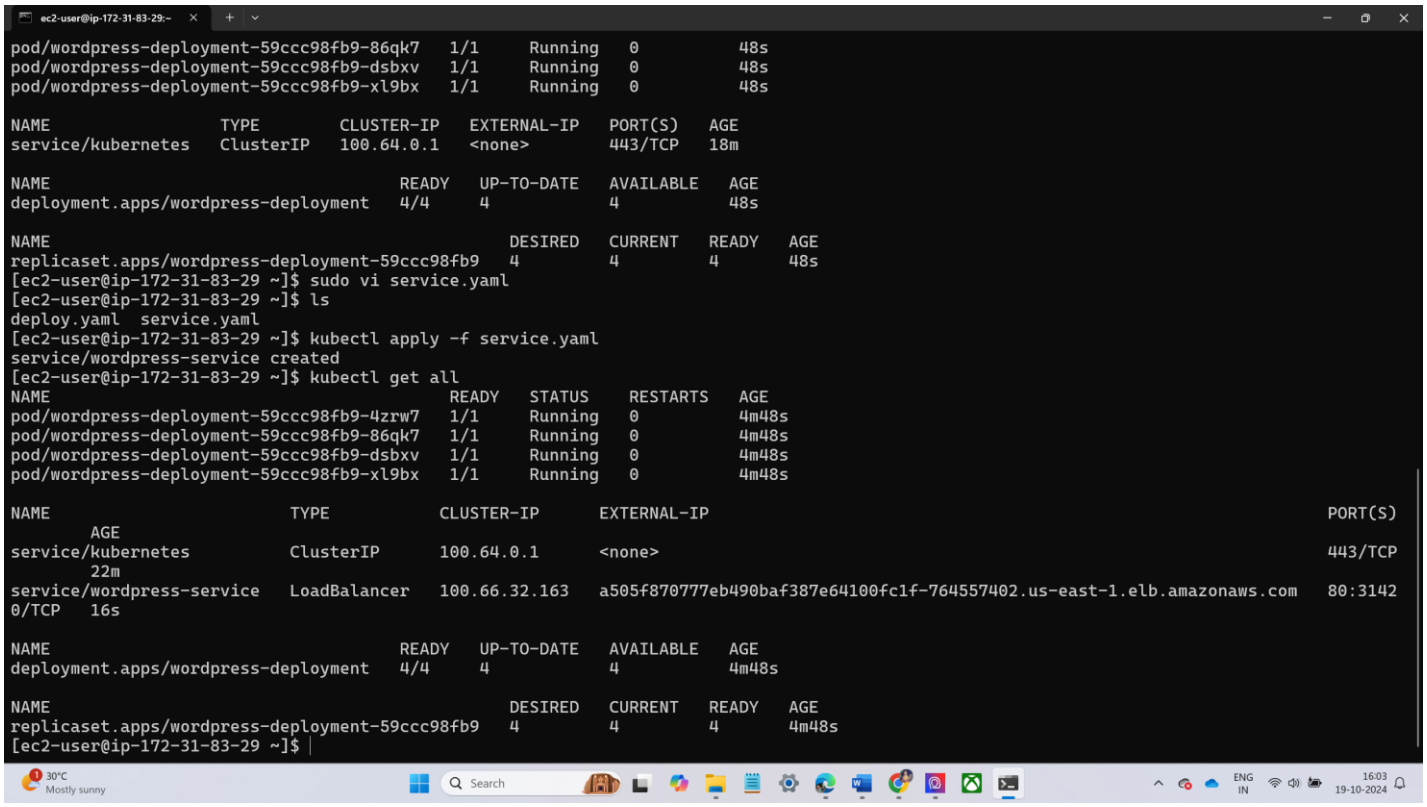
- now check the pods list



- now create service file and write script



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



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