Integrate, Automate, and Deploy web applications using Terraform, and Jenkins in AWS

What is Terraform?

Terraform is an open-source infrastructure as a code (IAC) tool that allows to creation, manage & deployment of the production-ready environment. Terraform codifies cloud APIs into declarative configuration files. Terraform can manage both existing service providers and custom in-house solutions.

Prerequisites:

* Basic knowledge of AWS & Terraform
* AWS Account
* IAM User
* GitHub Account
* AWS Access & Secret Key
* Jenkins Tool
* List of steps in the Pipeline:

Step 1: Create a file for the VPC

Step 2: Create a file for the Subnet

Step 3: Create a file for the Internet Gateway

Step 4: Create a file for the Route table

Step 5: Create a file for EC2 instances

Step 6: Create a file for Security Group for the Front-end tier

Step 7: Create a file for Security the Group for the Database tier

Step 8: Create a file Application Load Balancer

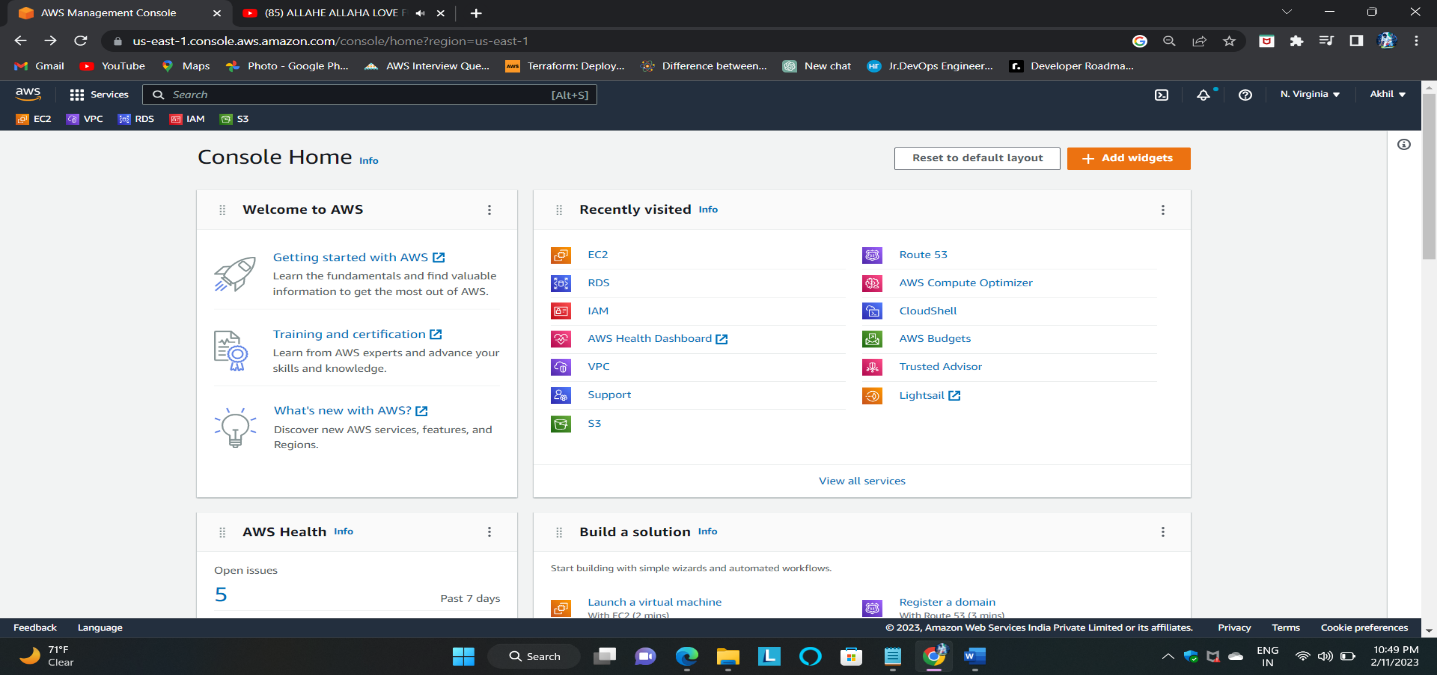
Step 9: Create a file for the RDS instance

Step 10: Create a file for outputs

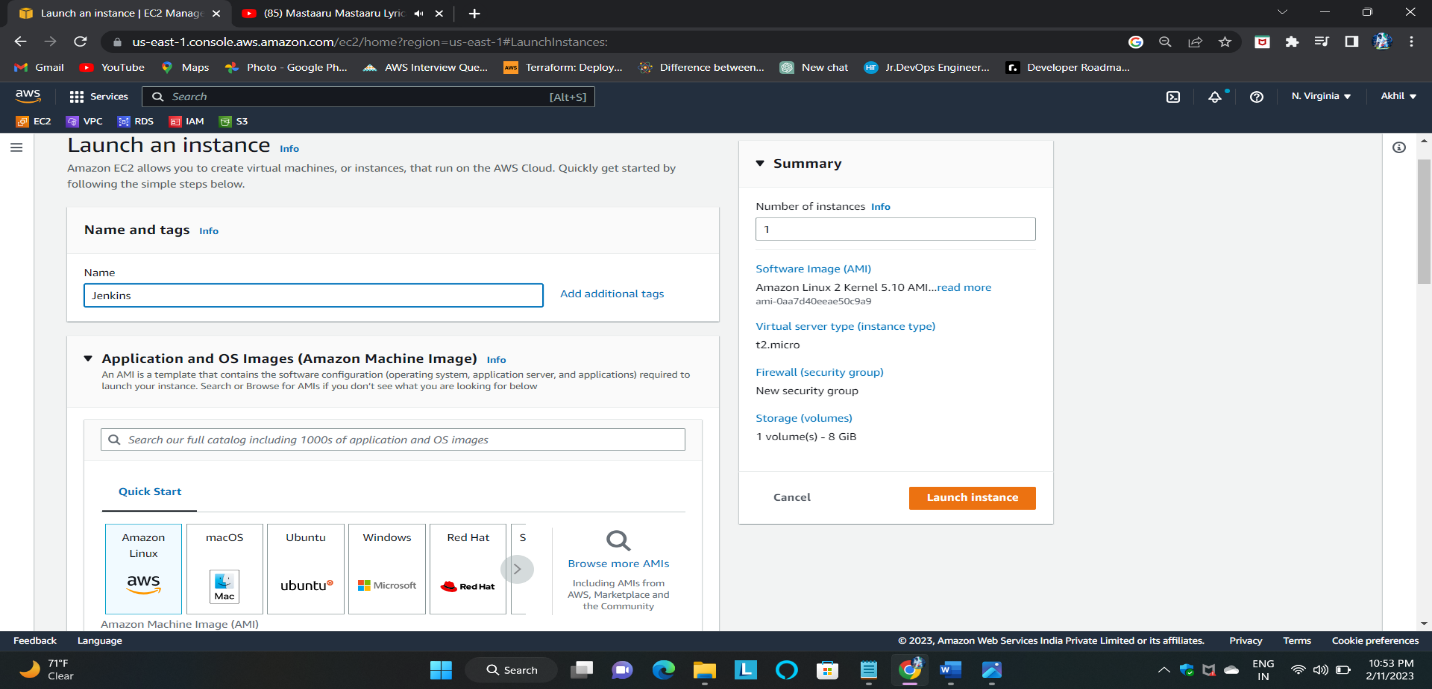
Step 11: Create a file for variable

Step 12: Create a file for user data

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

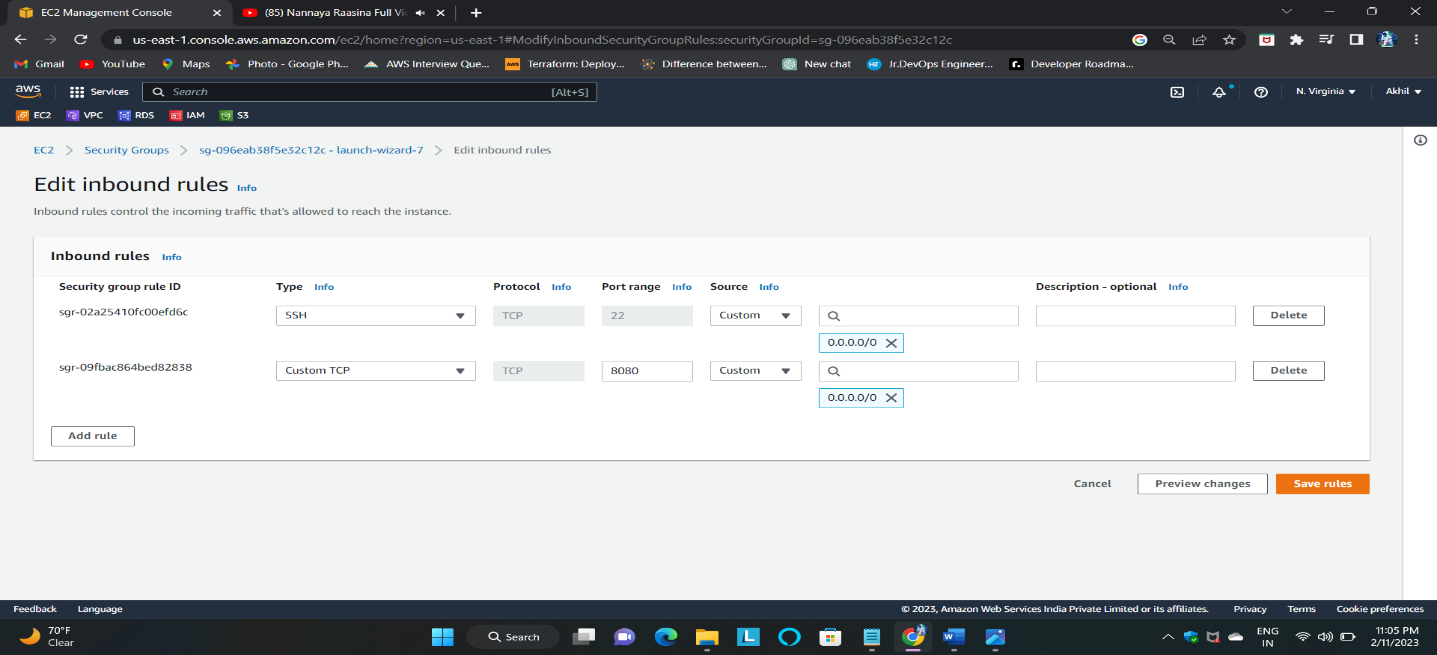
**EC2: Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware upfront, so you can develop and deploy applications faster**

# 2. Choosing an AMI – Amazon Linux 2 AMI

* An AMI is a virtual image used to create a virtual machine within an EC2 instance.
* You can also create multiple instances using a single AMI when you need instances with the same configuration.
* You can also create multiple instances using different AMI when you need instances with a different configuration

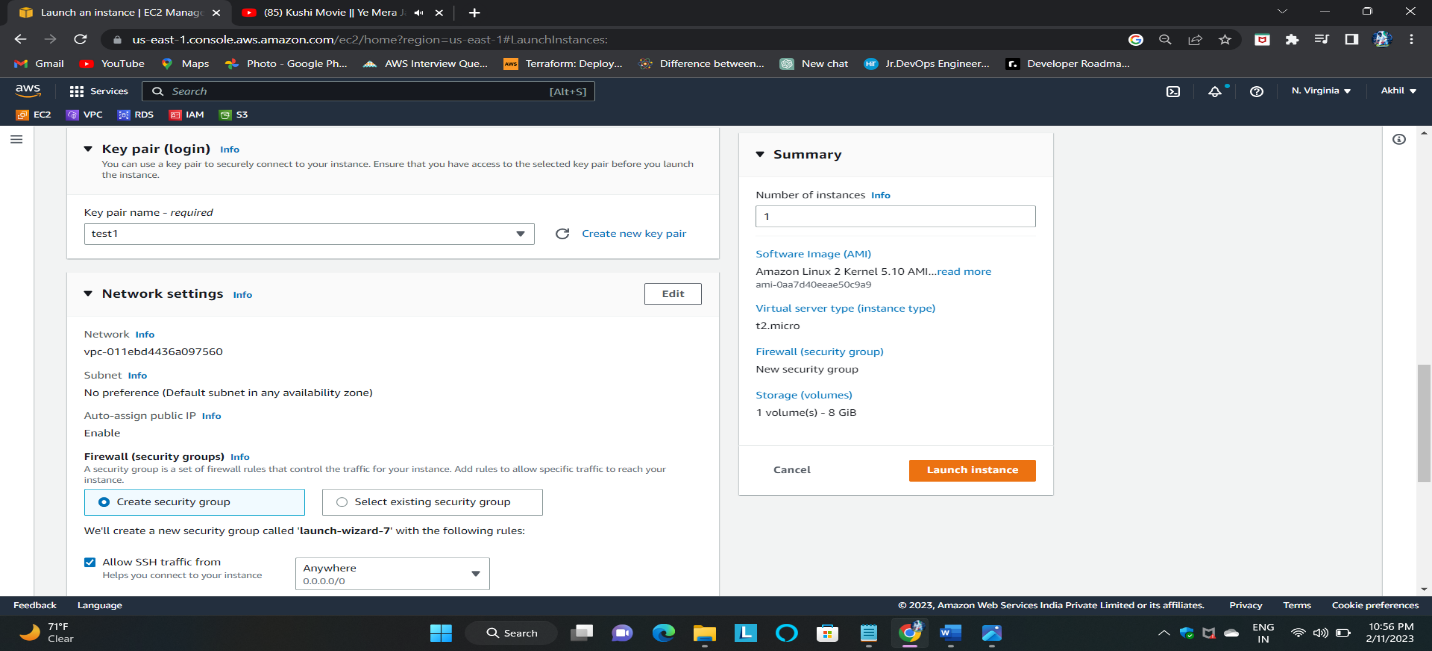
# 3. Configure the Security Group

**A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below.**

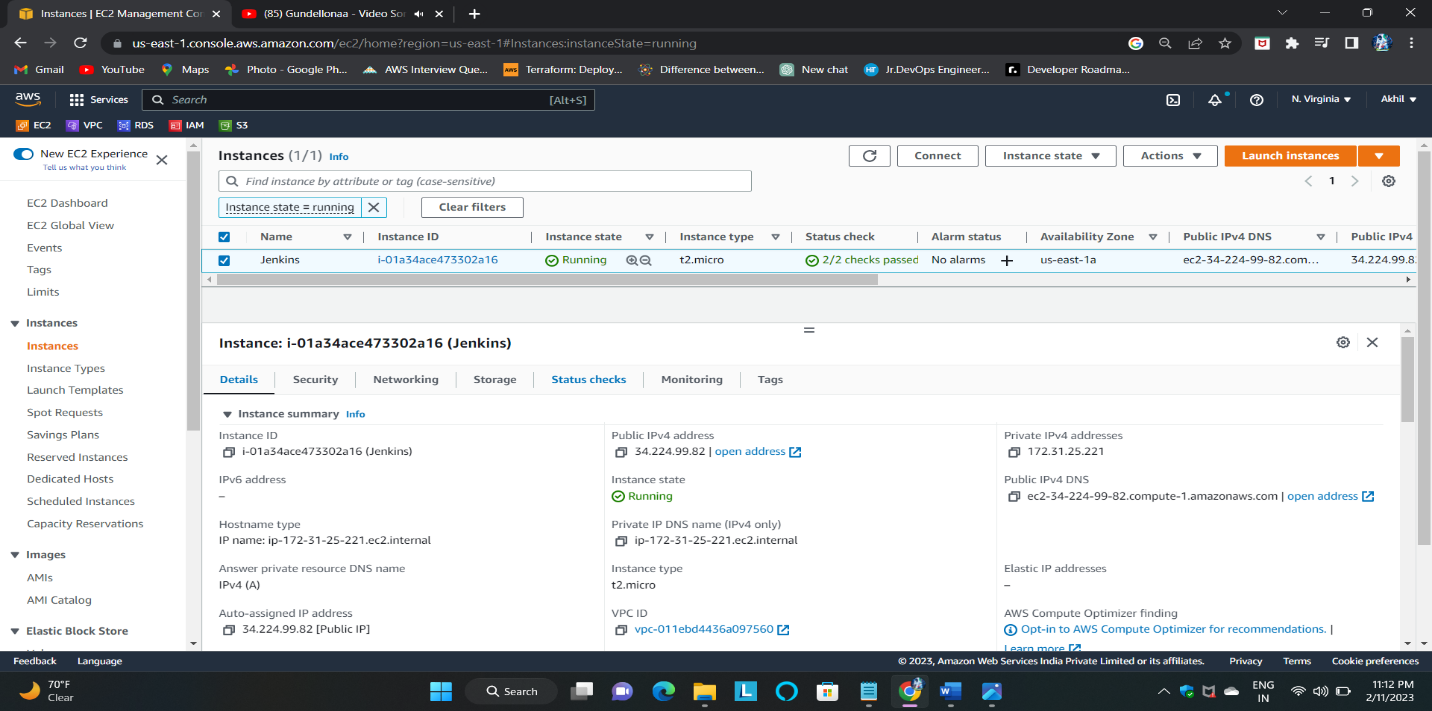
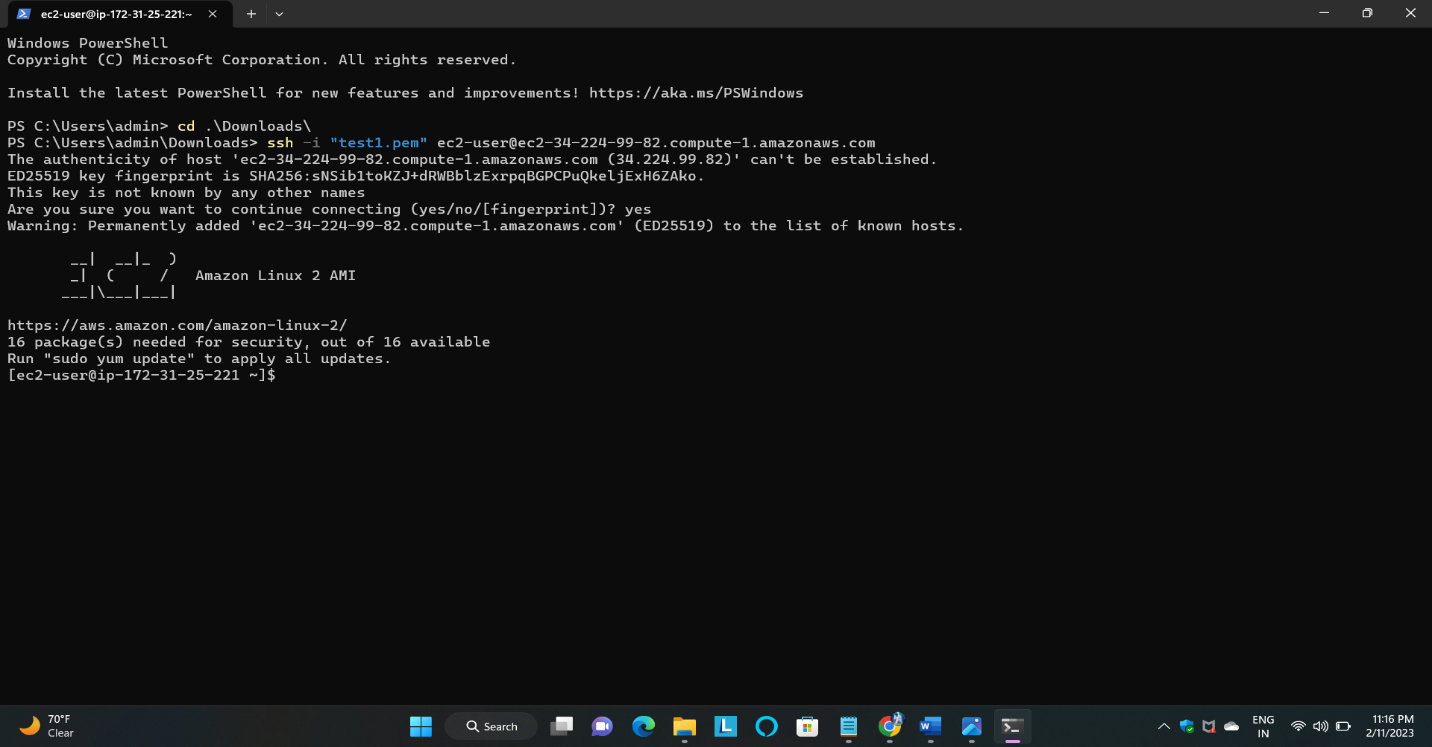
Here, I am assigning the port numbers for the inbound rule SSH-22, 8080, and All Traffic for the outbound rule.

**4. Select an Existing keypair or create a new keypair**

A key pair consists of a public key that AWS stores, and a private key file that you store.

 **the** key file is required to obtain the password used to log into your instance. **Together, they allow you to connect to your instance securely.**

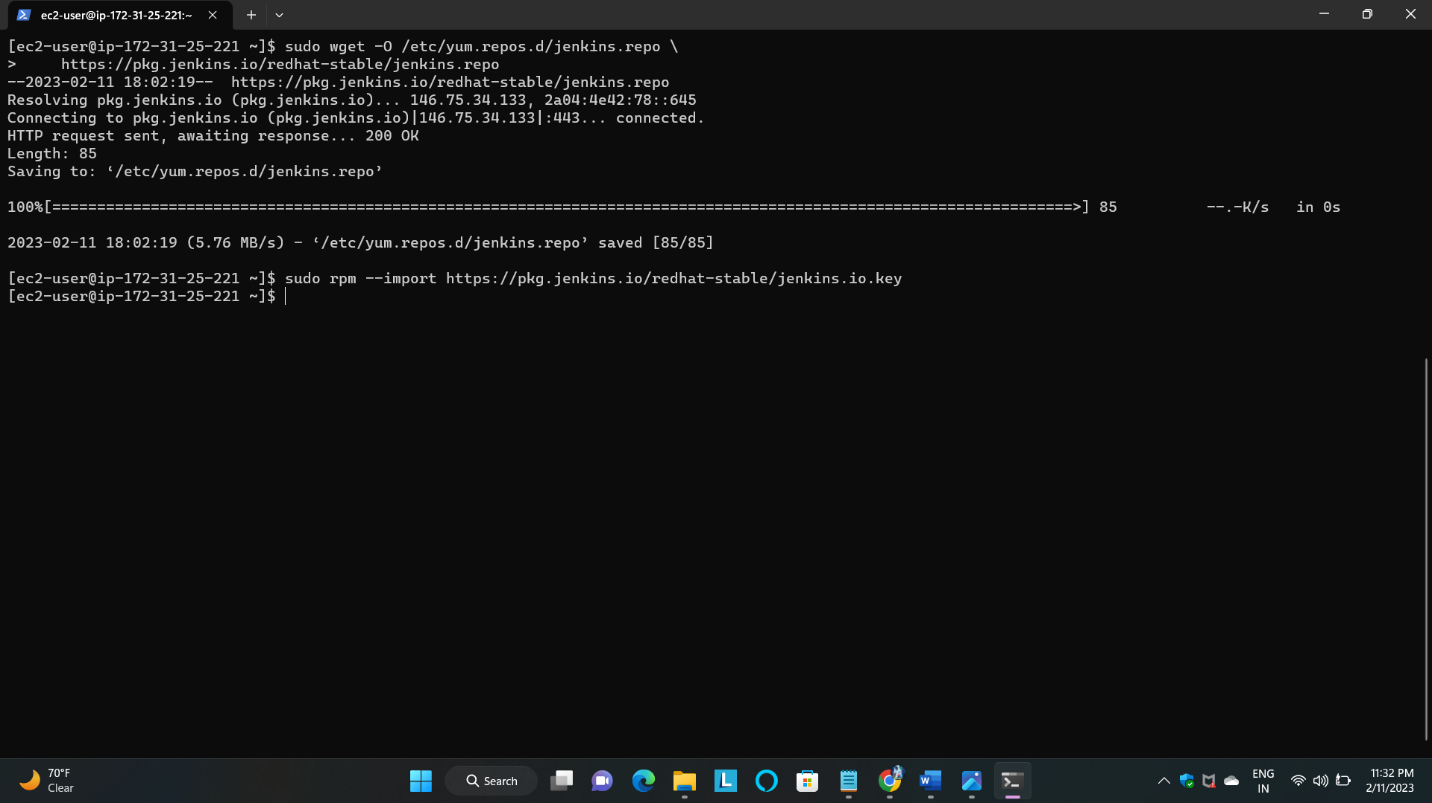
Finally, the launched a Jenkins server Instance. Now to connect an EC2 instance

* Now to the instance page and click on connect then go to SSH Client/EC2 instance connect.
* By using the SSH command we are accessing our instance in Command Line Interface Terminal

Module:2 Install Jenkins and related repos by using these commands.

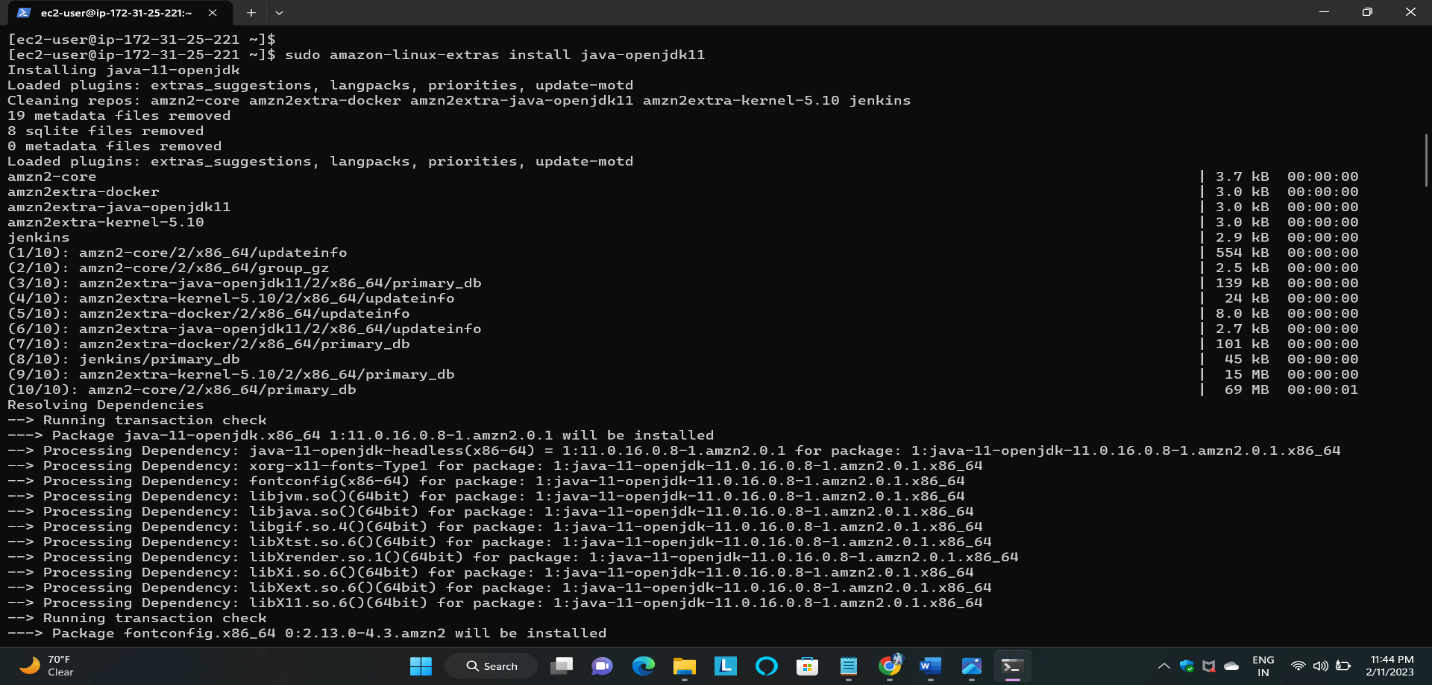
* Install related repos and packages, and import the public key of the Jenkins software repository, by using the below command.

sudo wget -O /etc/yum.repos.d/jenkins.repo <https://pkg.jenkins.io/redhat-stable/jenkins.repo>

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

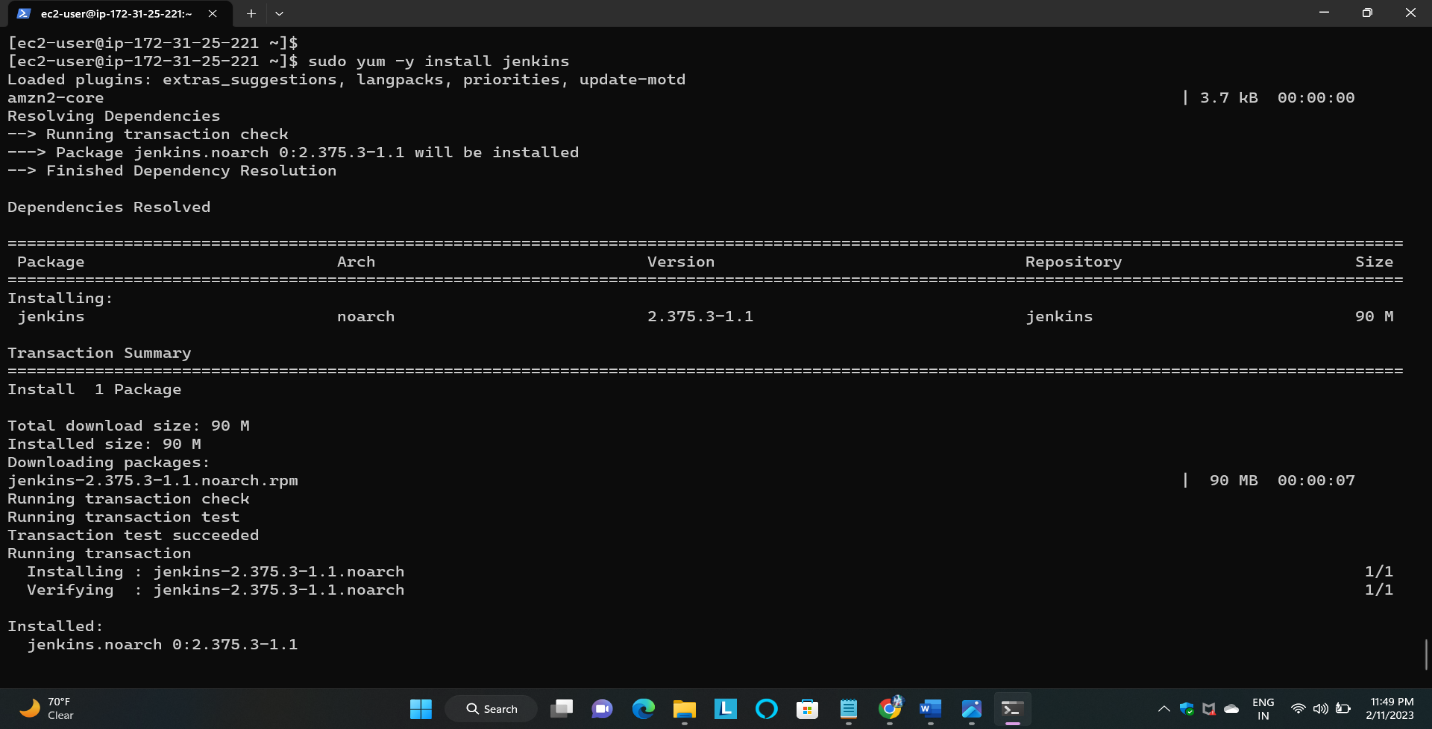
* Add required dependencies for the Jenkins package by using these commands

sudo amazon-linux-extras install java-openjdk11



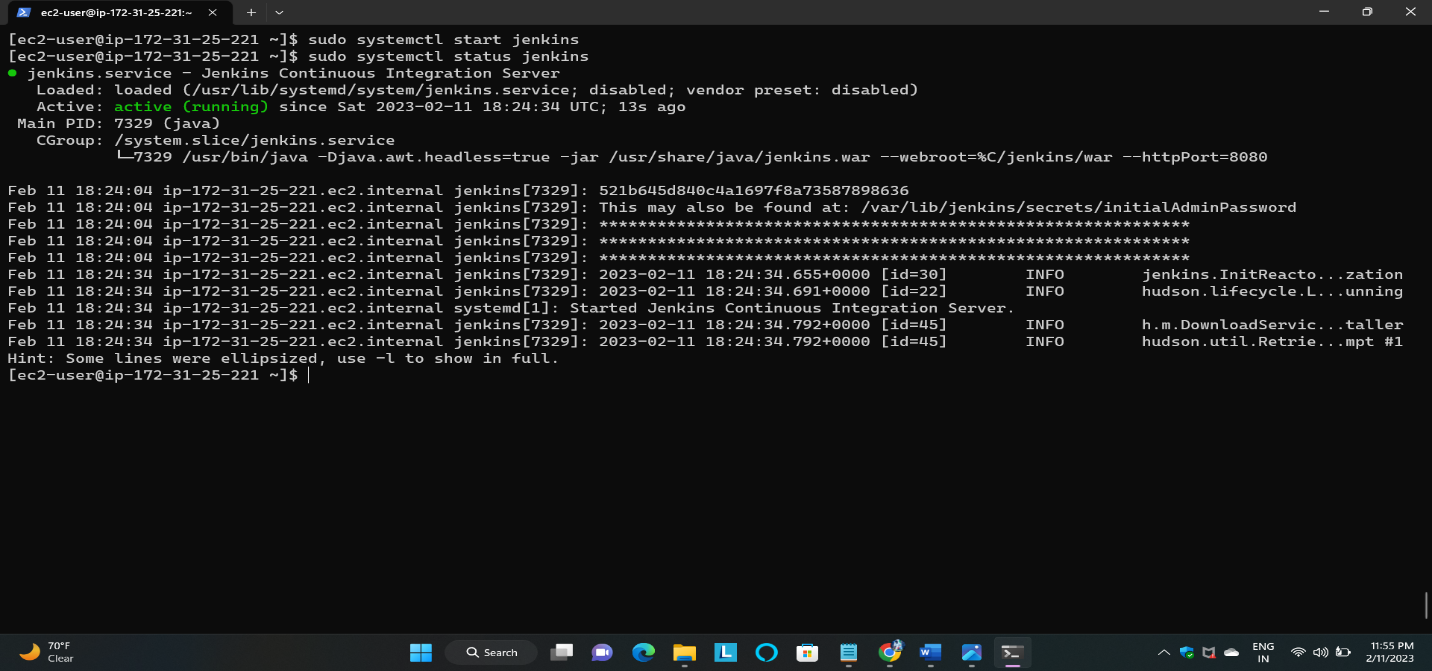
* Install Jenkins by using the below command.

sudo yum -y install Jenkins



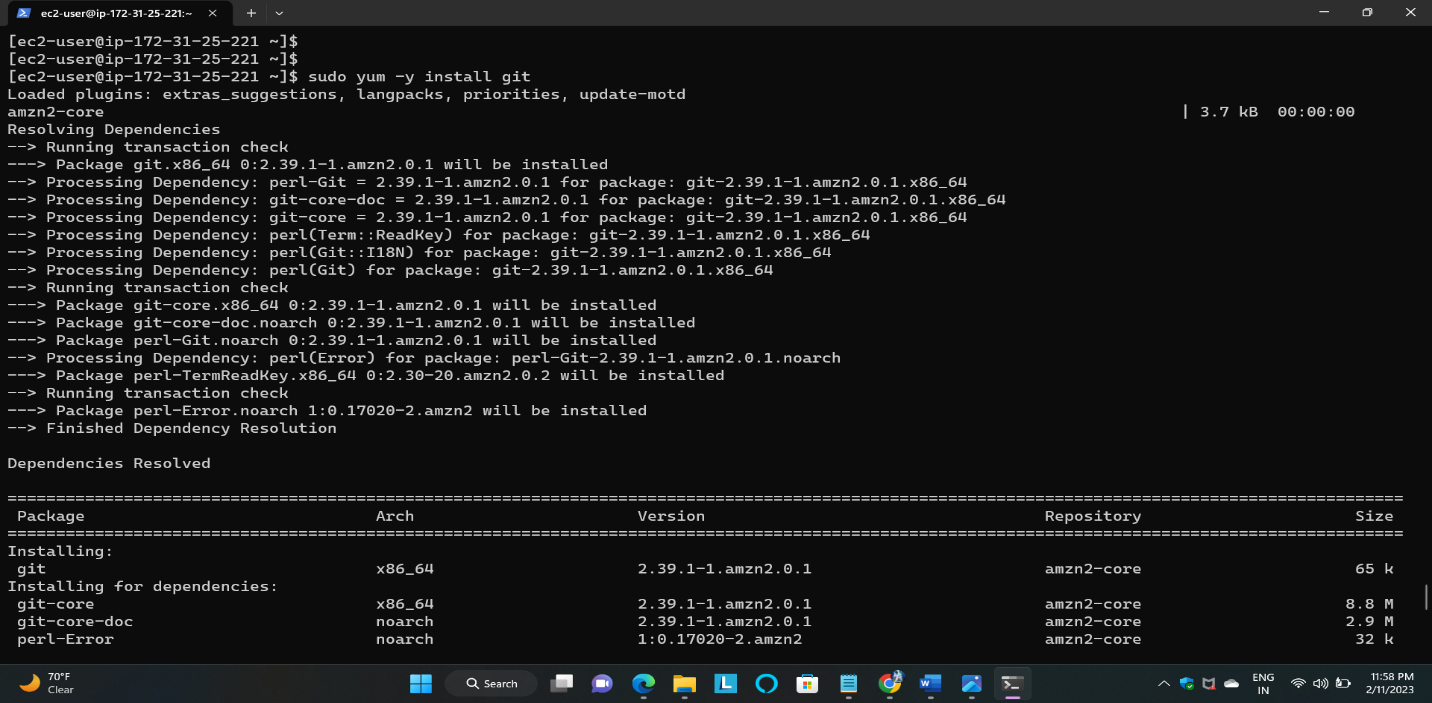
* Start Jenkins and check the status of Jenkins.

sudo systemctl start jenkins

sudo systemctl status Jenkins

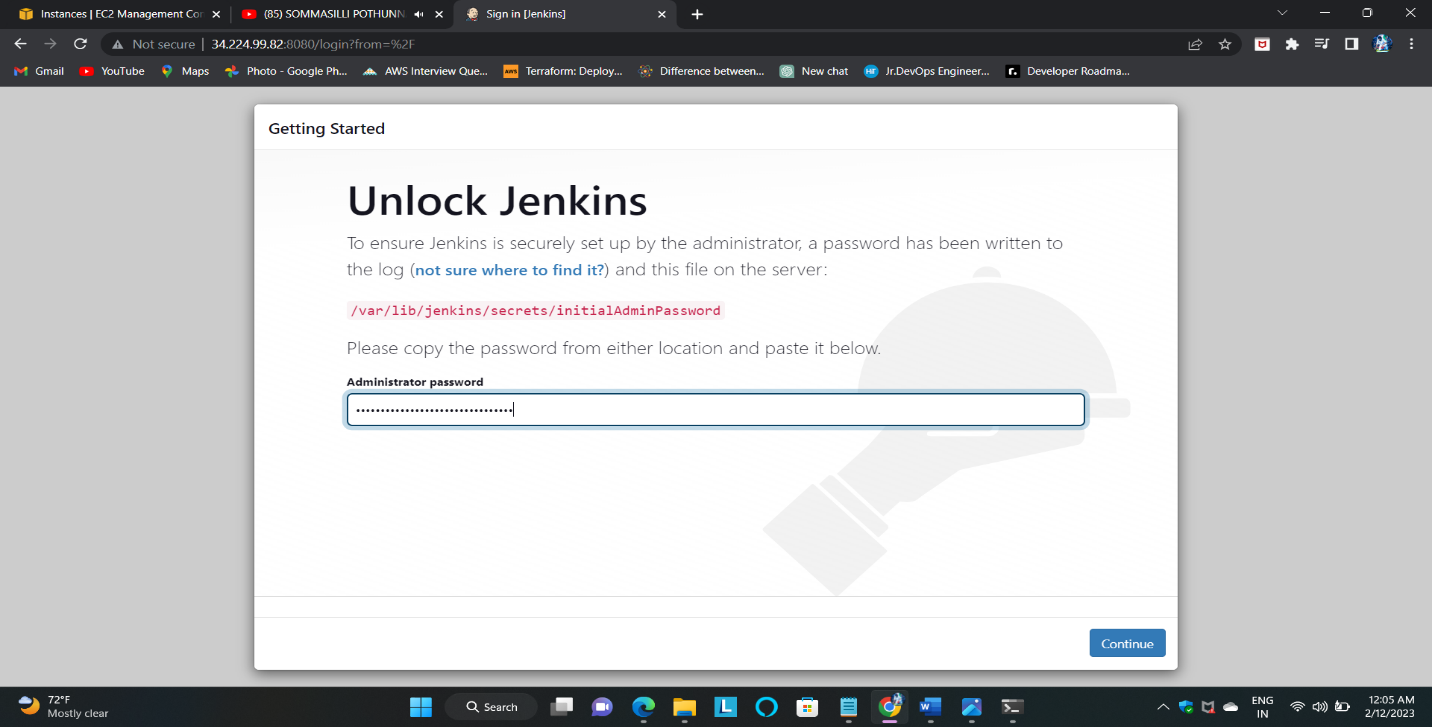
* Install using the git below command.

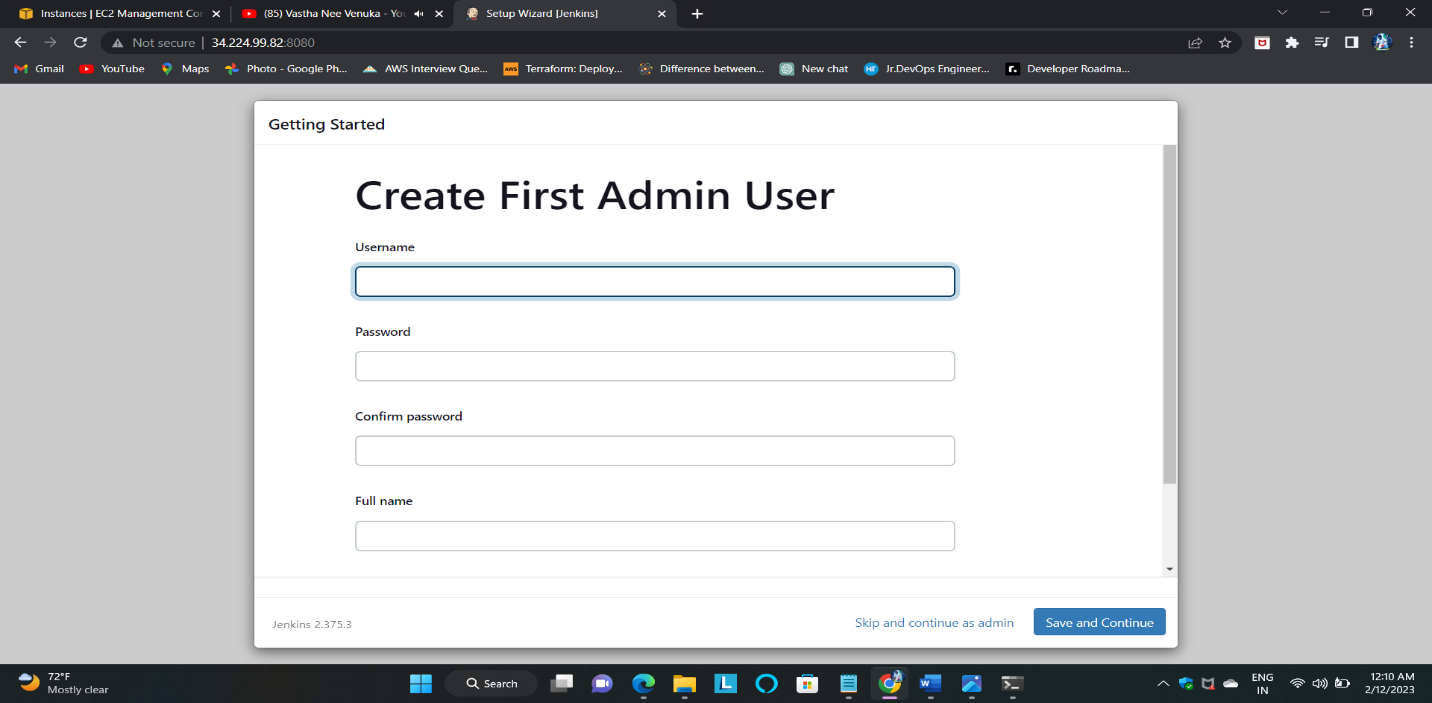
sudo yum -y install git



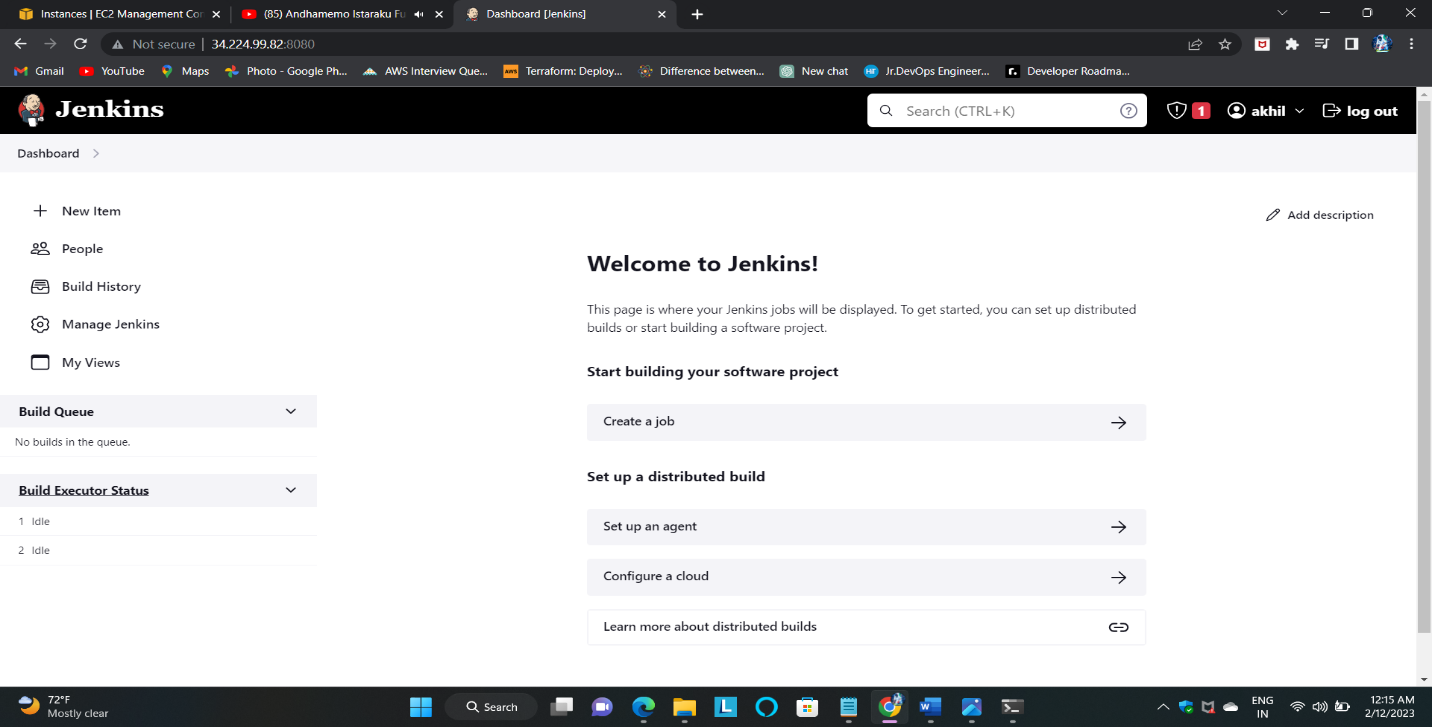
Module – 3: Setup the Jenkins with related repos. And create jobs.

Unlock the Jenkins by using this command

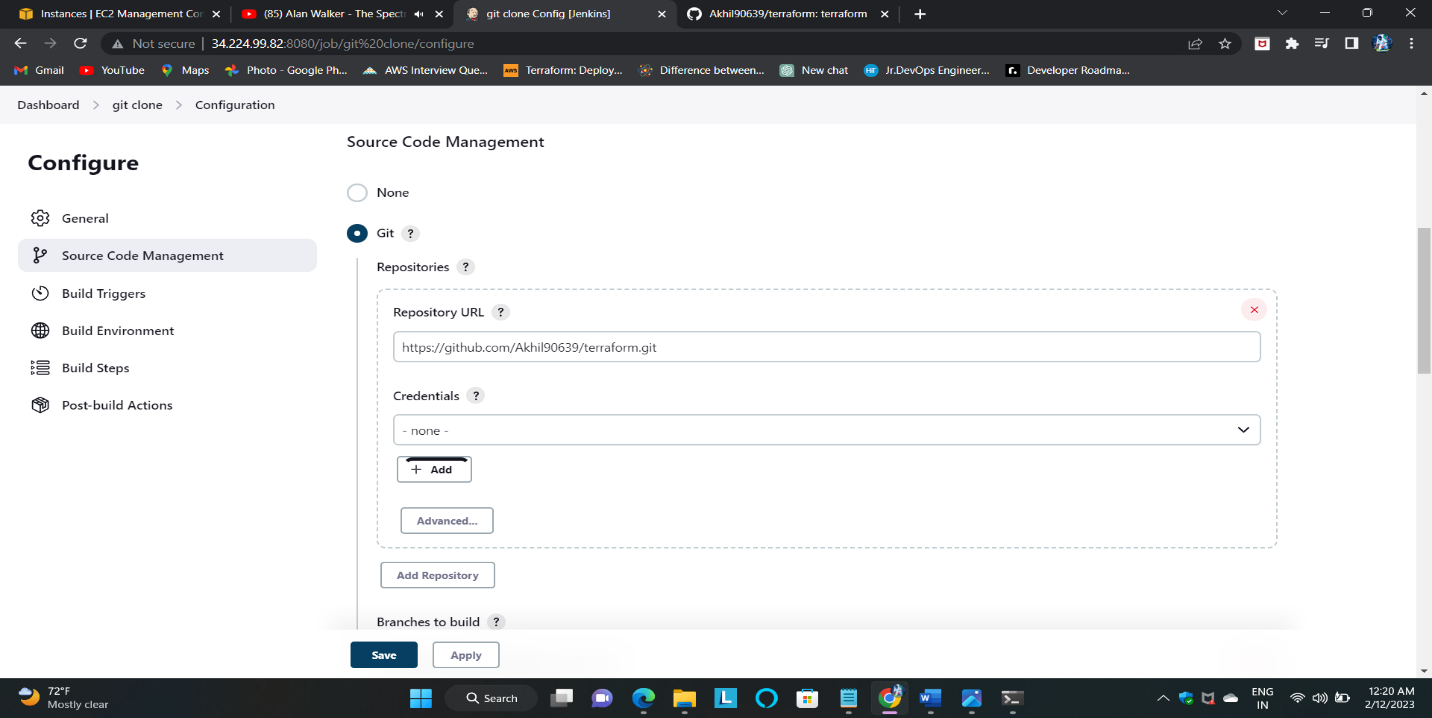
• sudo cat /var/lib/jenkins/secrets/initialAdminPassword

• Set up the configuration with user details and save the details.

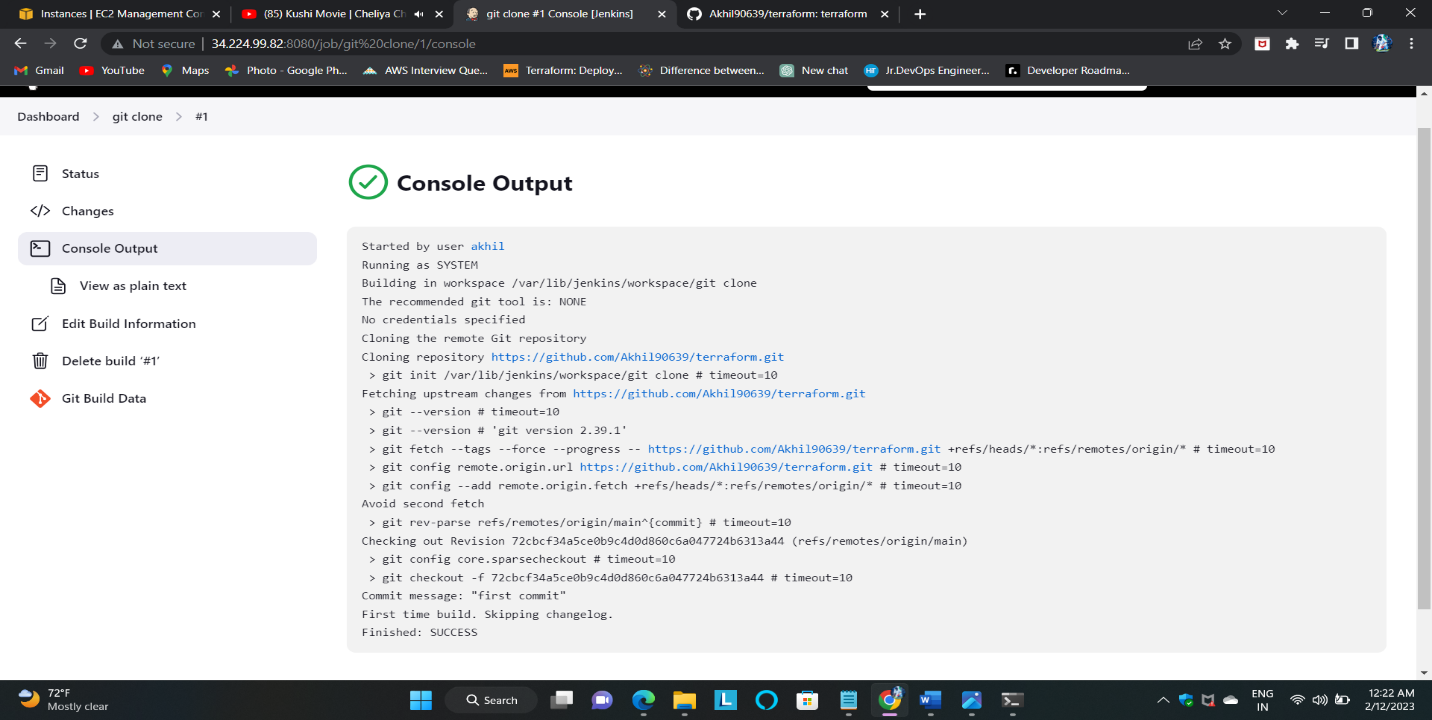
1. And then lastly, I had a look to see that this was running correctly.

 2. This is the Jenkins welcome page.

3. Create the first free-style project by cloning the Git link, and saving the job



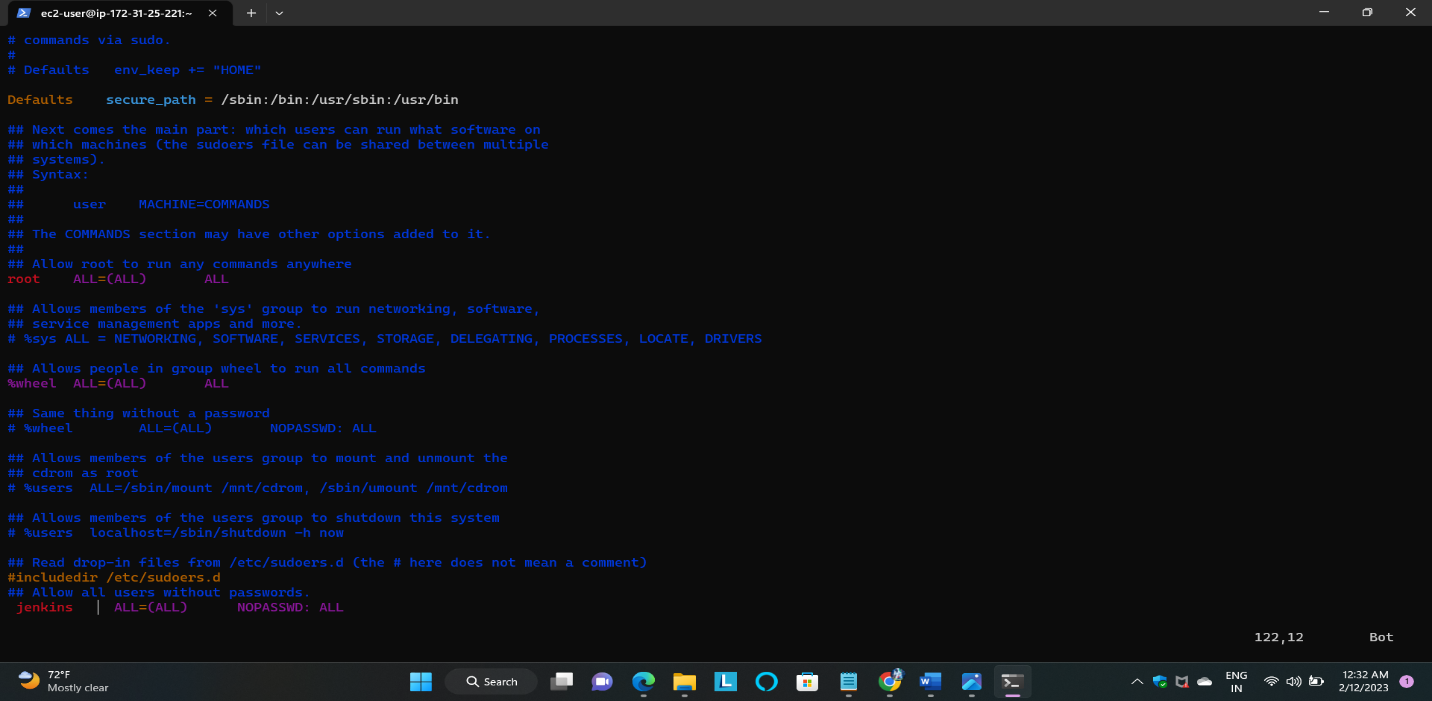
* After saving the first job build the job.

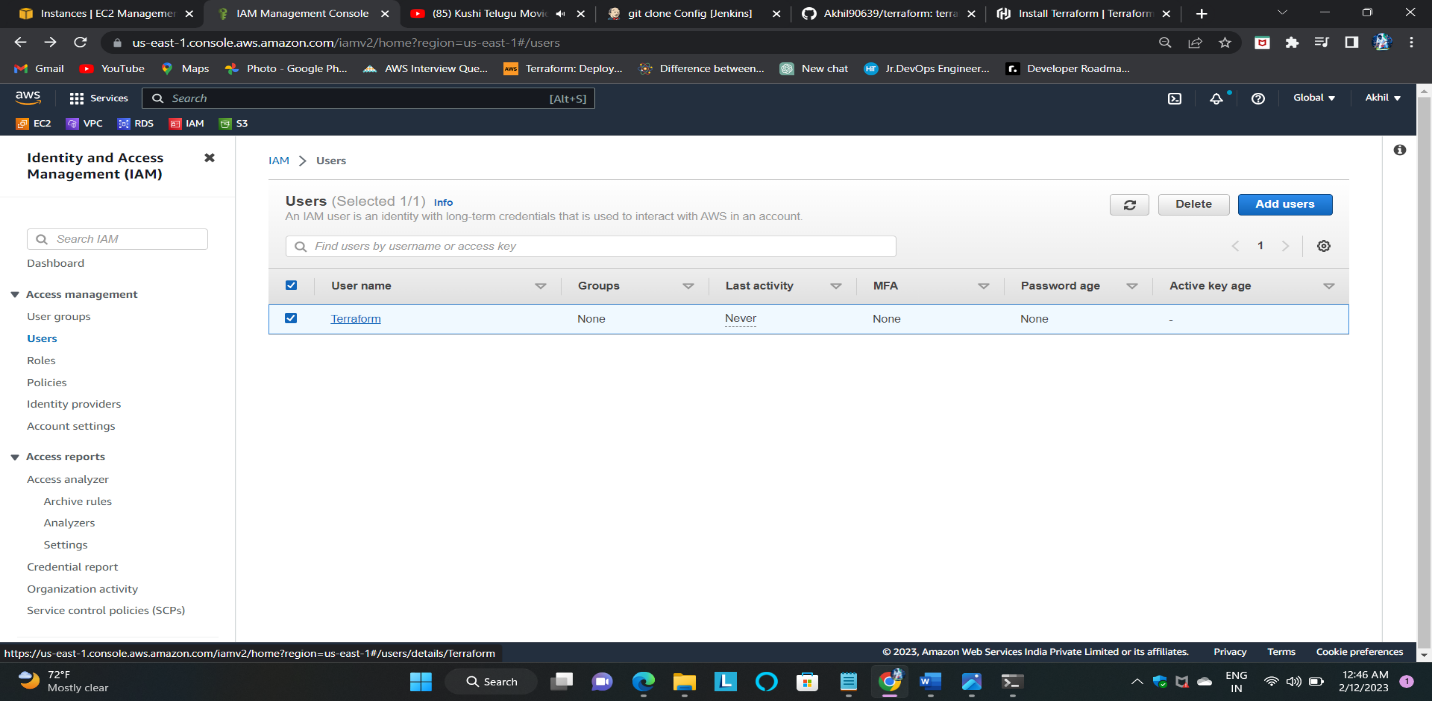
 Here is the console output

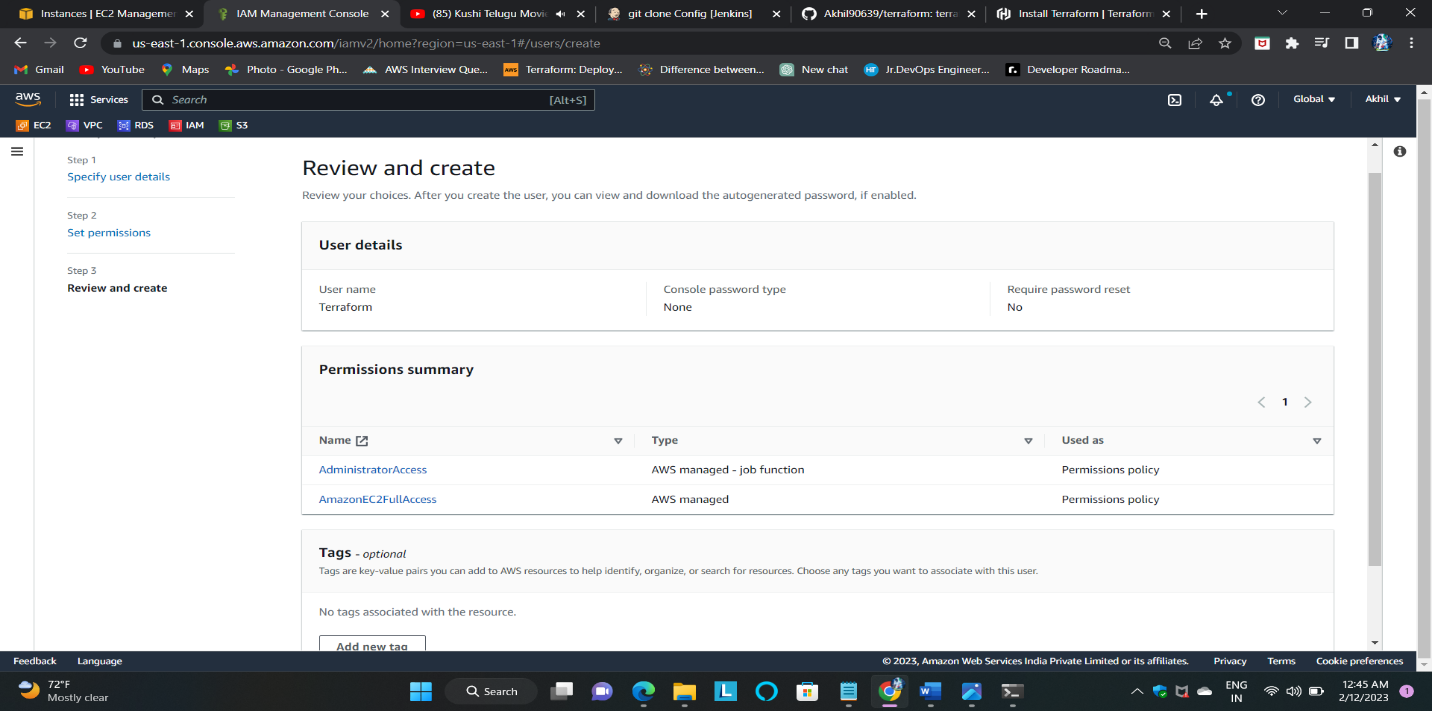
1. Give the permissions to Jenkins and docker by using these commands and restart Jenkins.

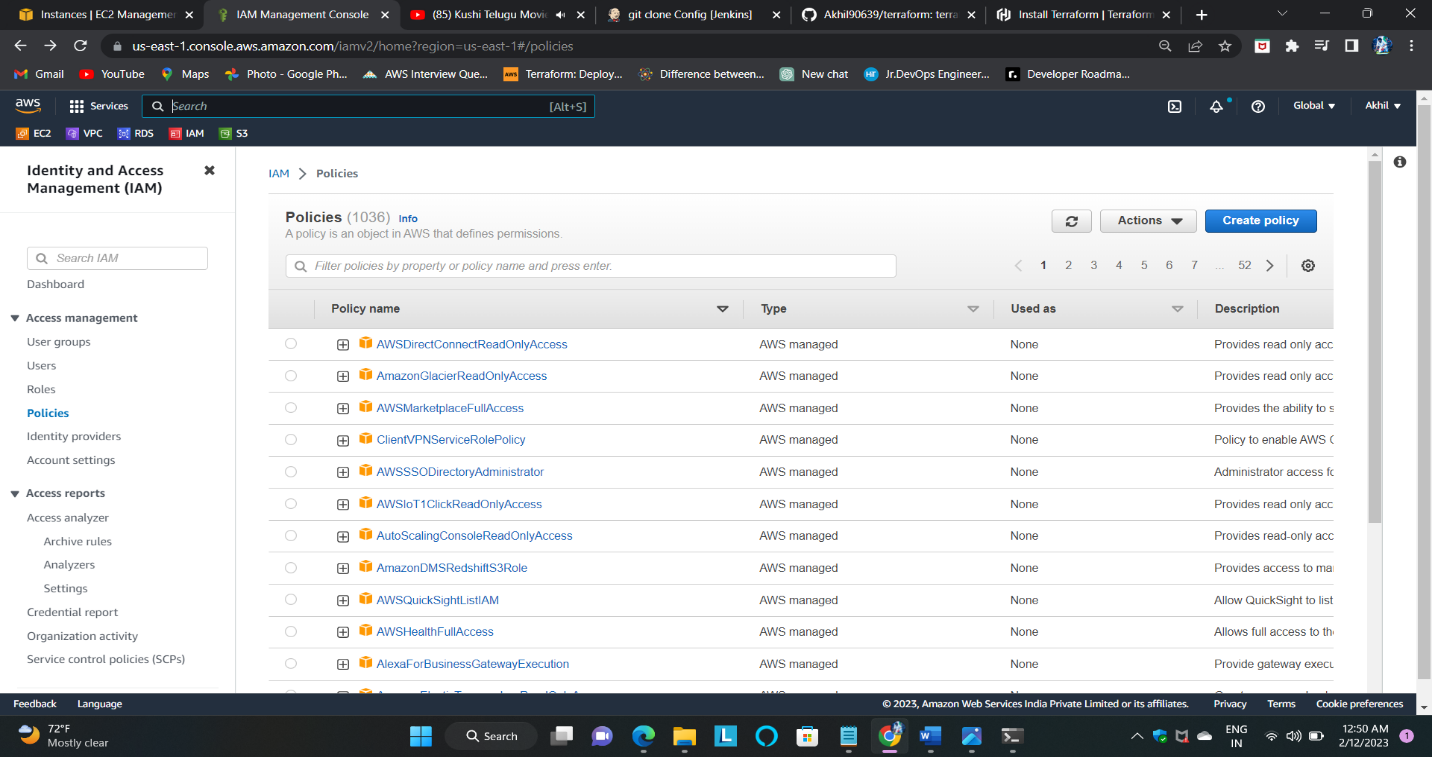
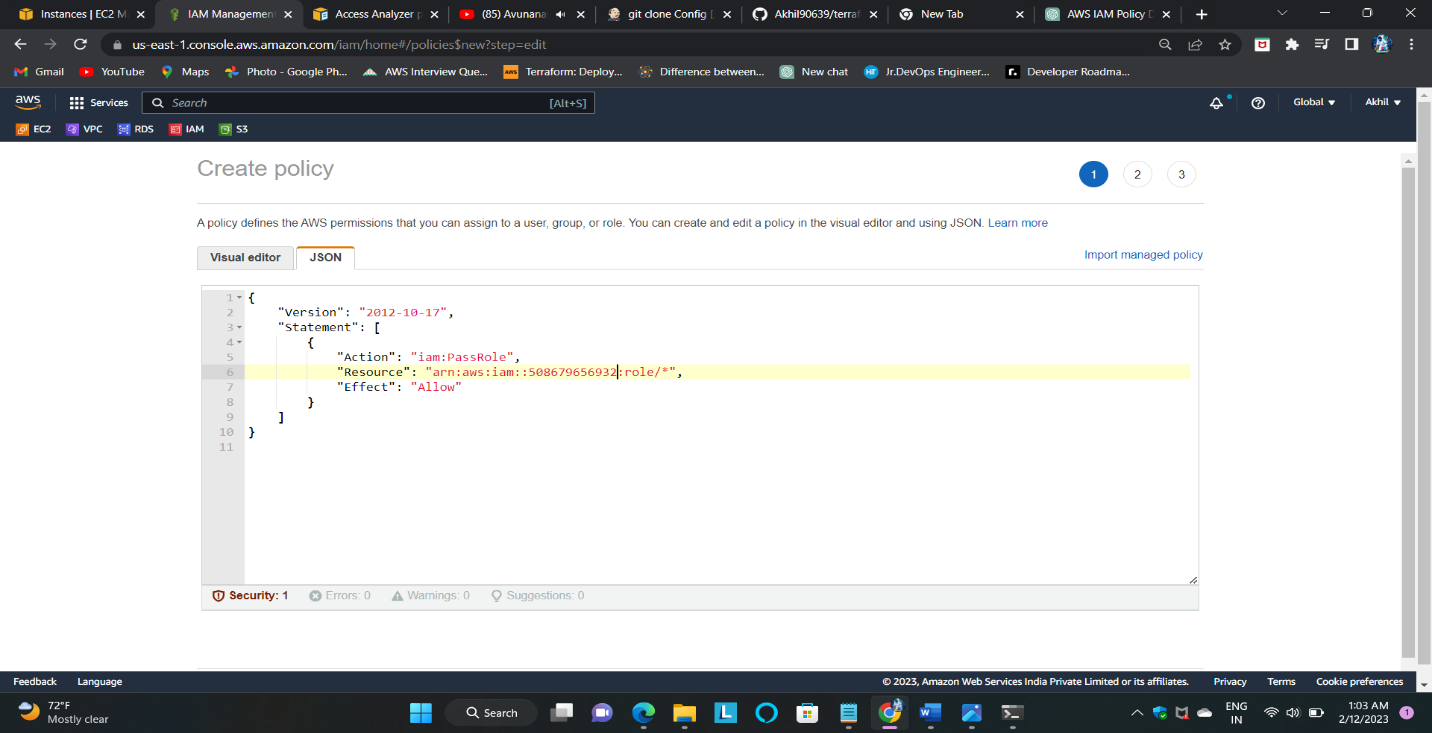
• sudo vi /etc/sudoers

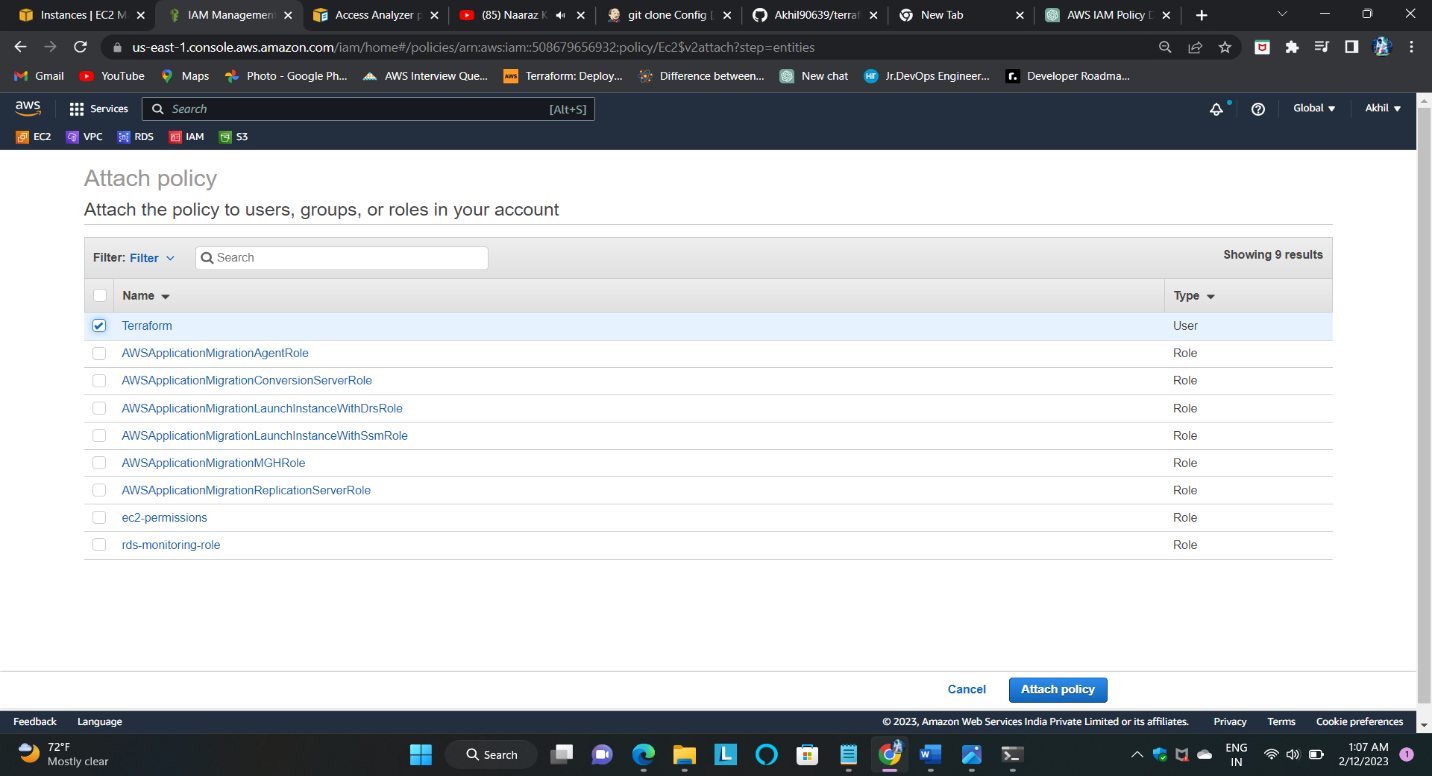
## Allow all users without passwords.

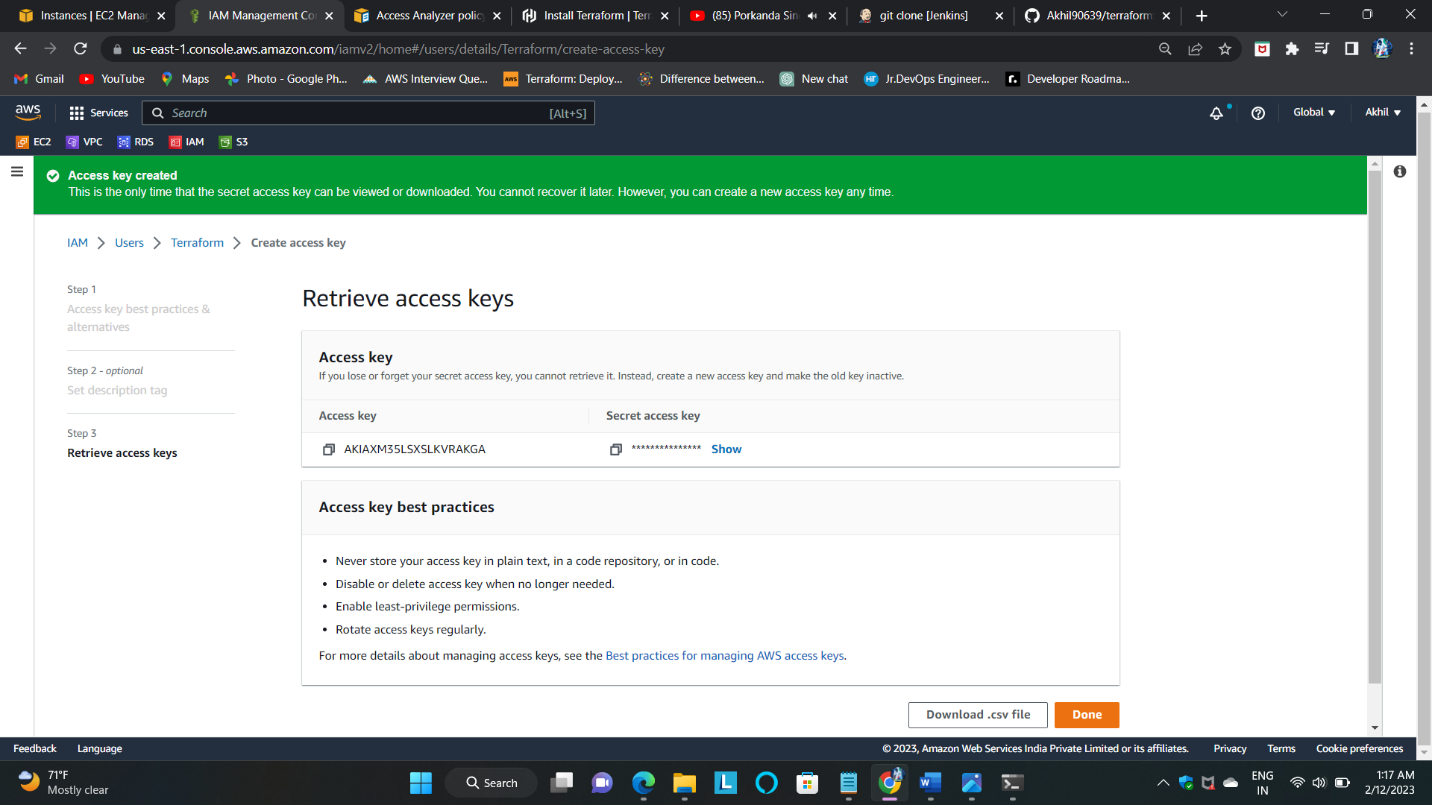
 Jenkins ALL=(ALL) NOPASSWD: ALL

1. Create the user and IAM policy
2. Click on add user then create and give the permissions
3. Give the permissions to administrative and ec2 full access in identity access and management

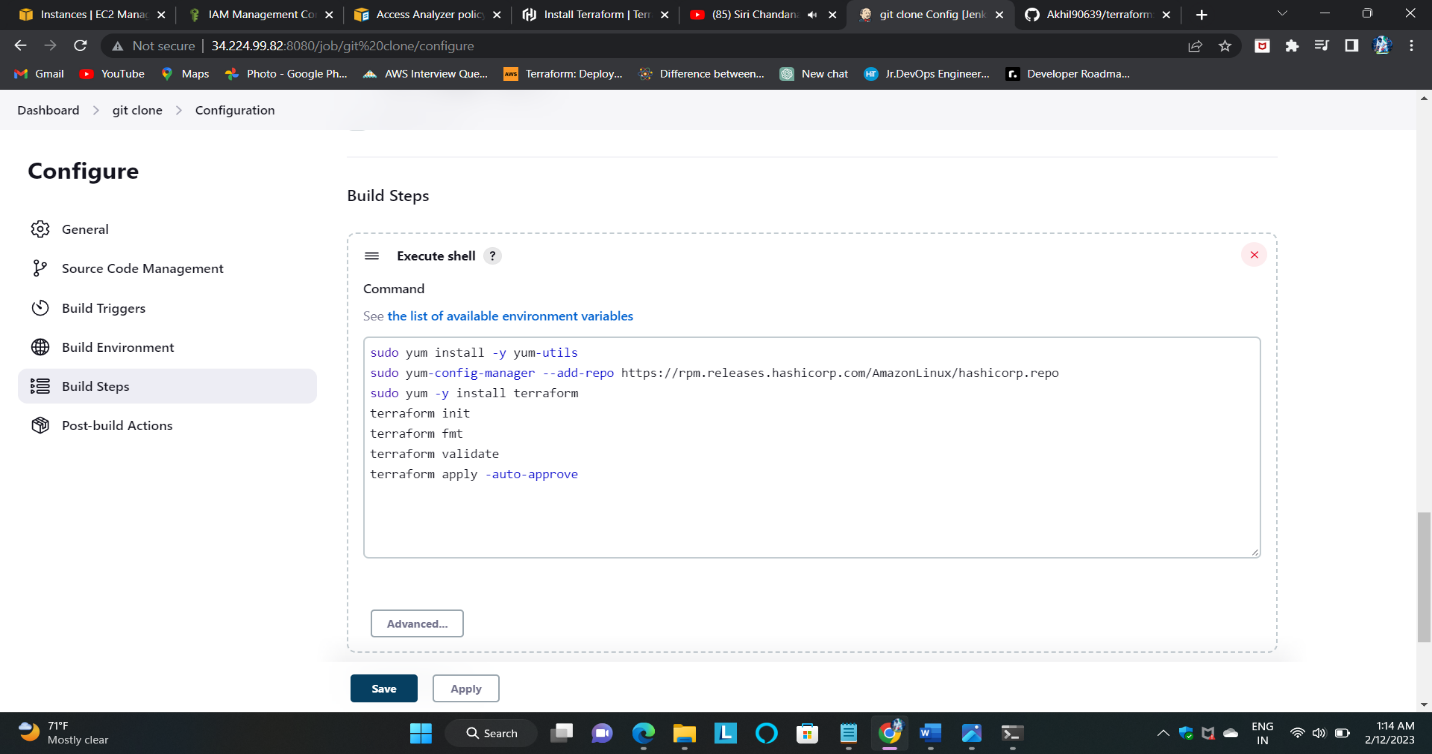


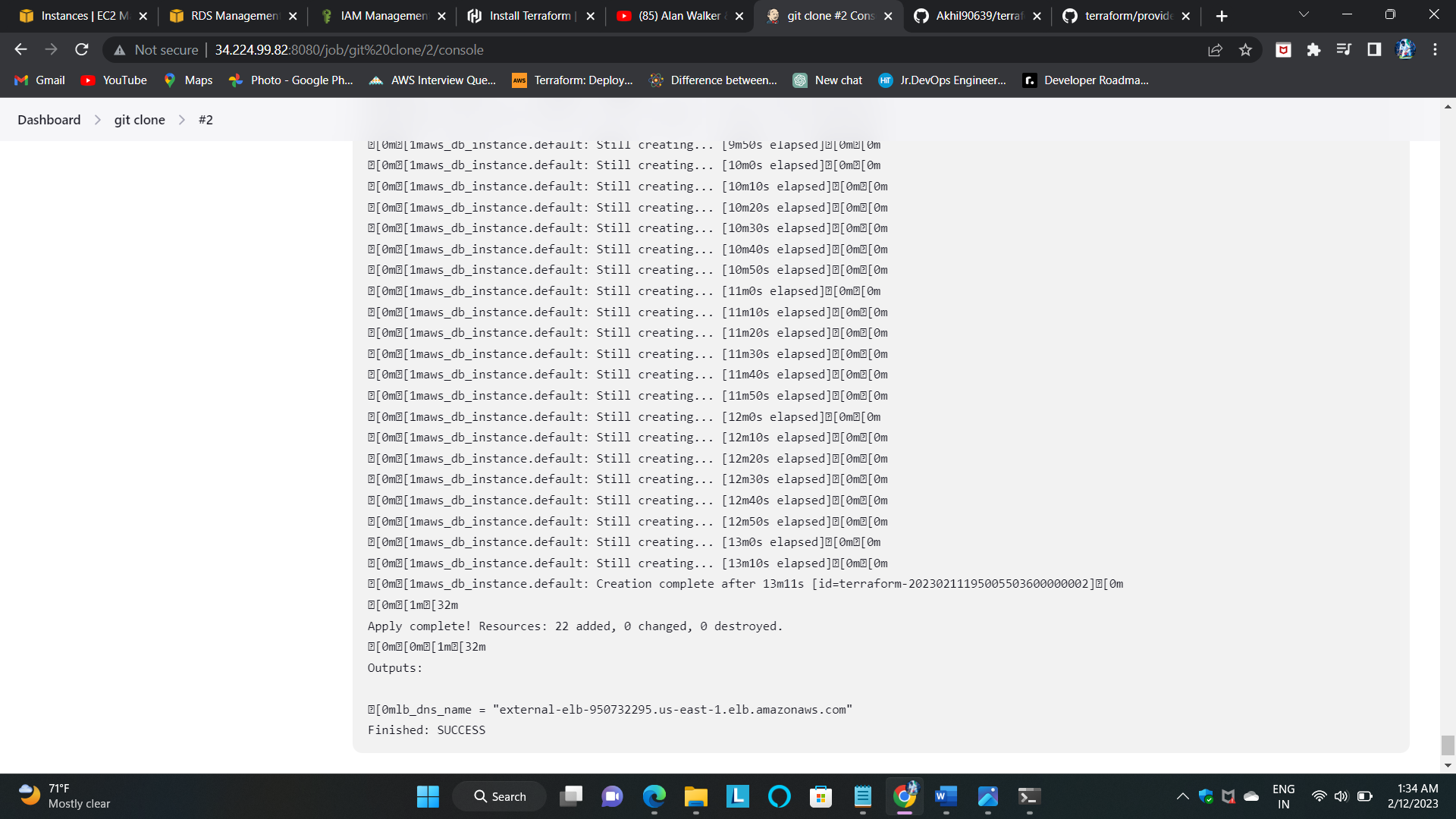
1. Click on create policy then save the policy
2. Create the IAM policy and give the script JSON format finally give the policy name then click on save.
3. Attach the policy to the user group and save it.



* Finally create the access and secret key then attach it to the provider.tf in terraform directory.

1. Create the second freestyle build job by using execute a shell.



1. Here is the final console output of creating AWS resources.