**Project title**

**AUTOMATED DEPLOYMENT PIPELINE**

**INTRODUCTION:**

The project aims to automate the deployment process of web applications using Jenkins, a popular continuous integration and continuous deployment (CI/CD) tool, and Nginx, a high-performance web server. This documentation outlines the steps required to set up and execute an automated deployment pipeline using Jenkins to deploy on an Nginx server.

**Prerequisites:**

Before setting up the Automated Deployment Pipeline with jenkins and nginx ensure the following prerequisites are met

**Tools:**

1.Ec2 instance

2. Jenkins

3.Github

4. Cloud Watch

**What is CI/CD?**

Continuous Integration/Continuous Deployment (CI/CD) is a software development practice aimed at automating the process of integrating code changes into a shared repository (CI) and rapidly deploying code to production environments (CD). In a single automated deployment pipeline, CI involves automatically triggering builds, running tests, and validating code changes whenever developers push code to a version control system. CD extends CI by automating the deployment process, enabling frequent and reliable delivery of applications to production environments. By automating tasks such as testing, building, and deployment, CI/CD pipelines streamline development workflows, improve code quality, and accelerate time-to-market, fostering collaboration and agility within development teams

There are Few steps to follow to create a CI/CD Pipeline.

1.Launch Ec2 Instance.

2.Install Jenkins

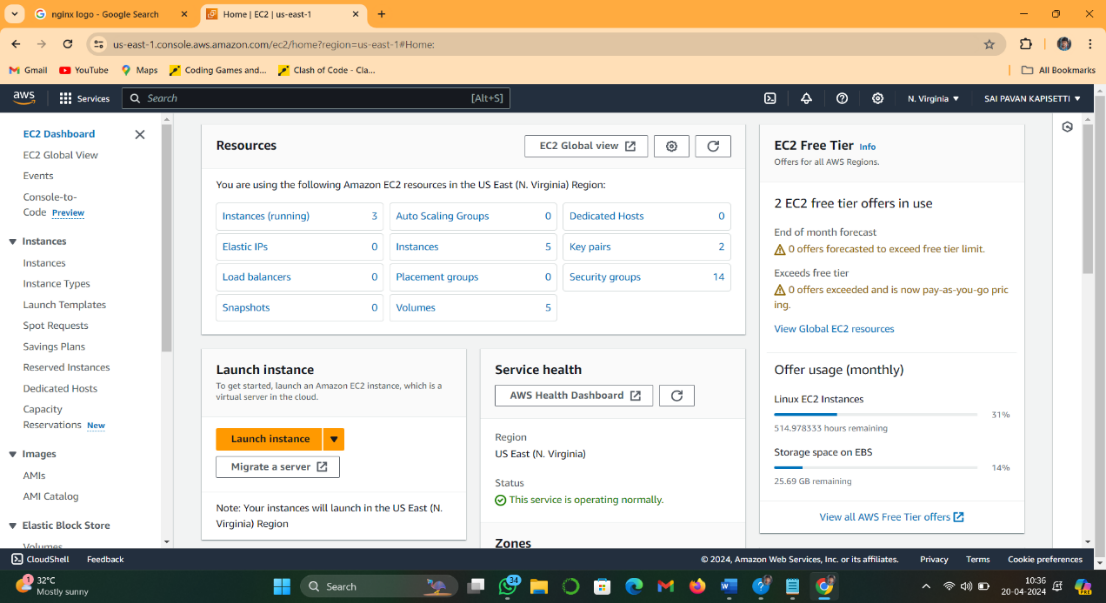
3.Deploying Application.

4.Monitoring Capabilities(cloud Watch)

**1.Launch an EC2 Instance:**

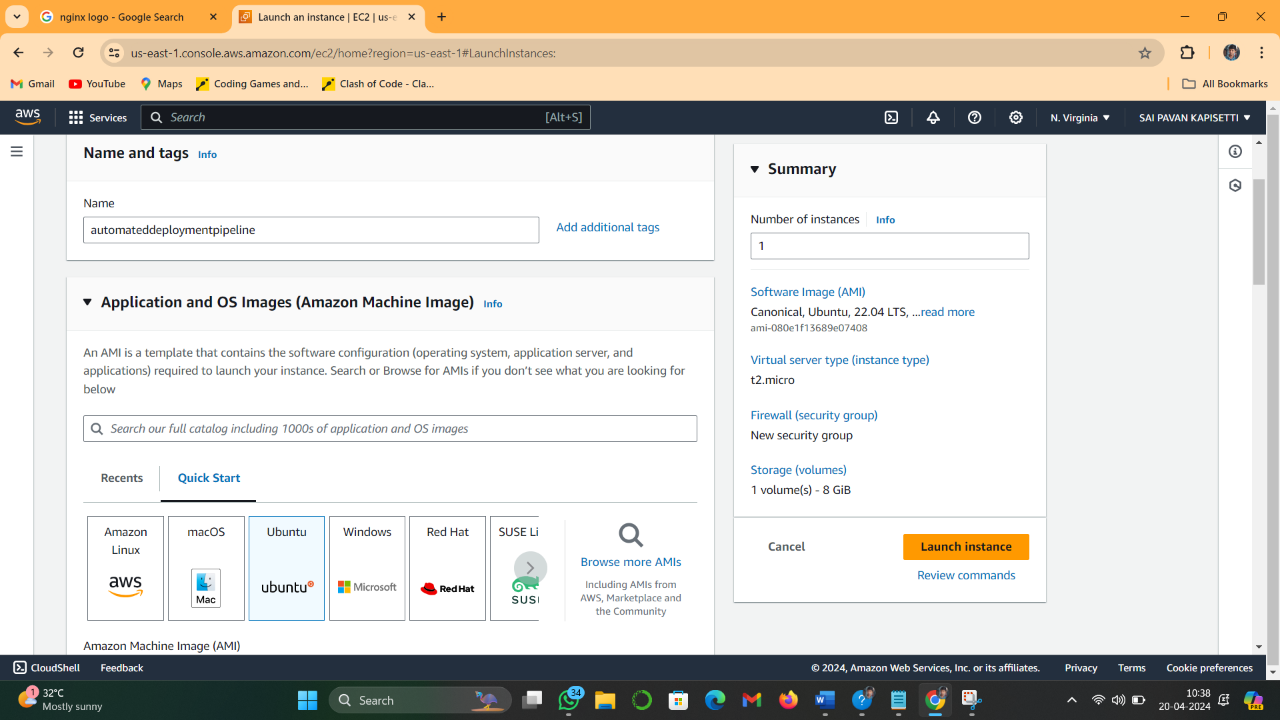
For creating an EC2 instance.Initially We have to sign in to Aws console .Then

Click on EC2 service After Click on Launch Instance.

****

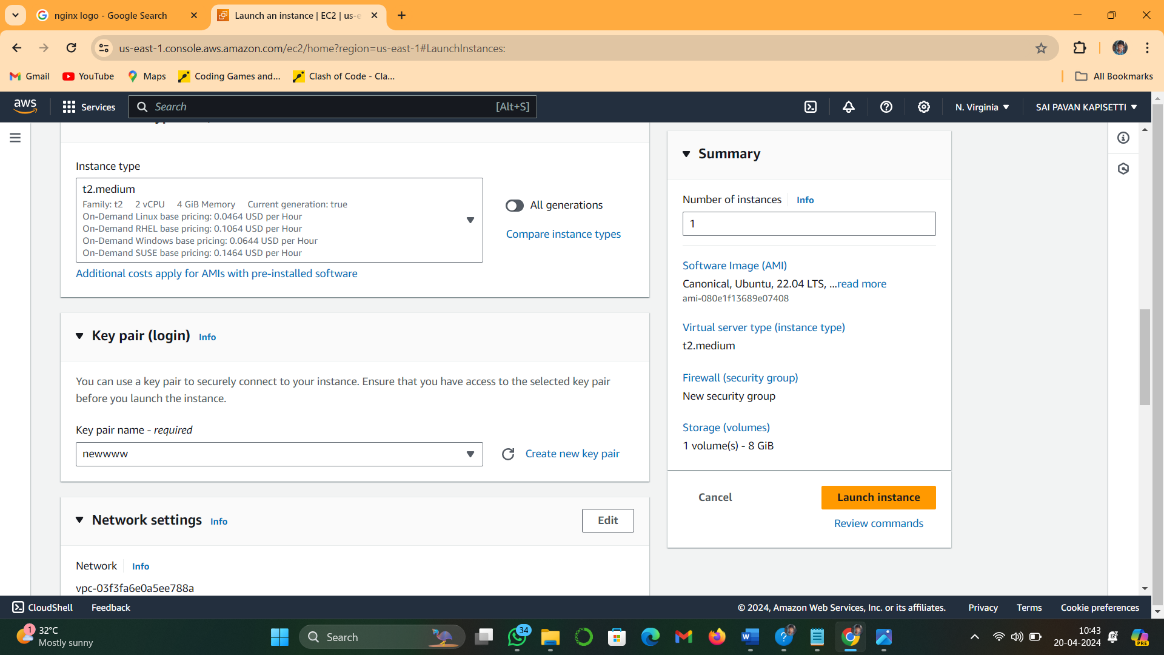
Give the Name for instance Automated Deployment pipeline and select the

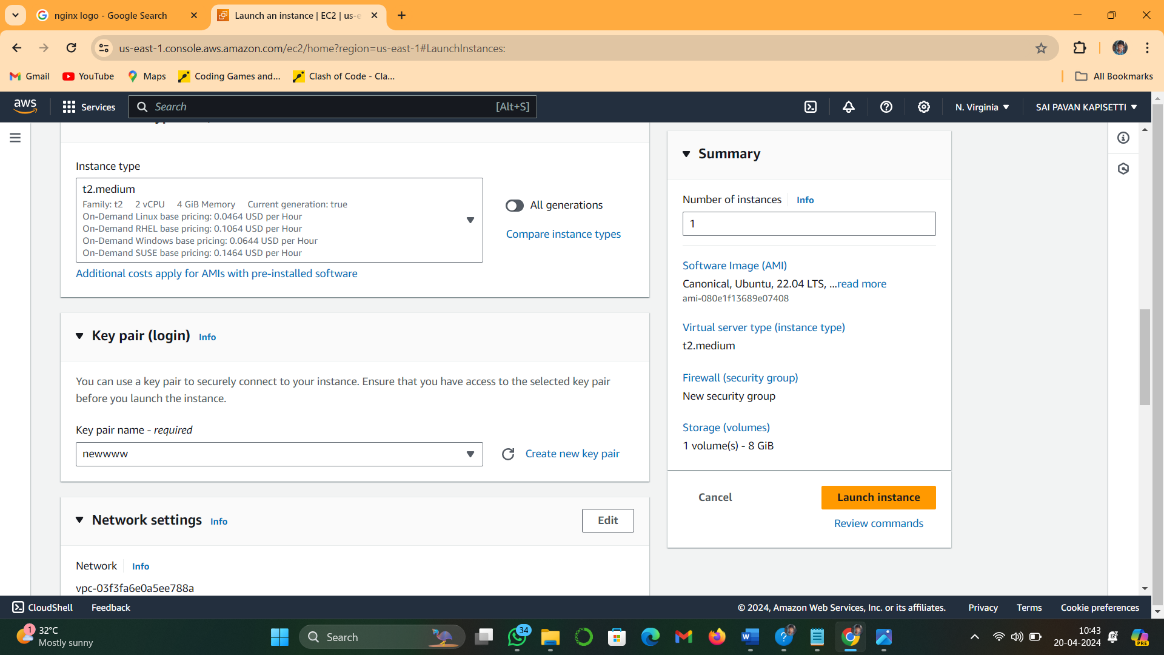
Ubuntu os



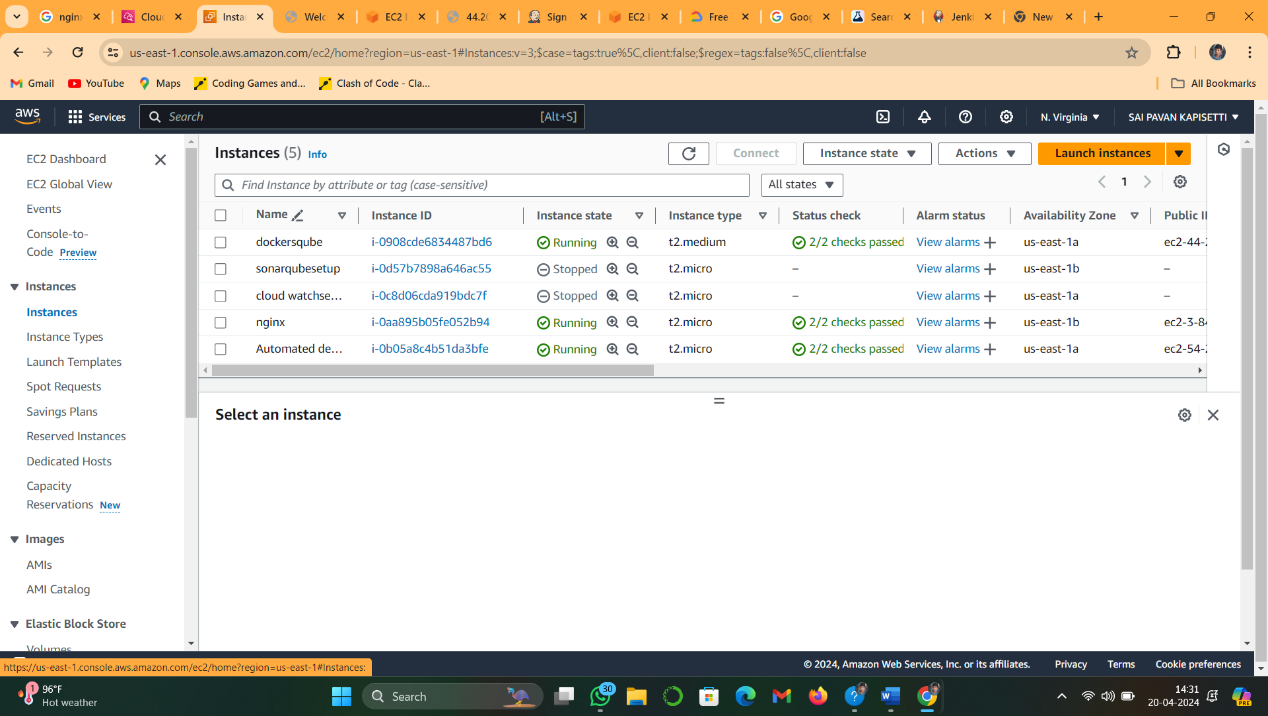
And Select the instance type As Per Requirement and Select a key pair new or

Existing



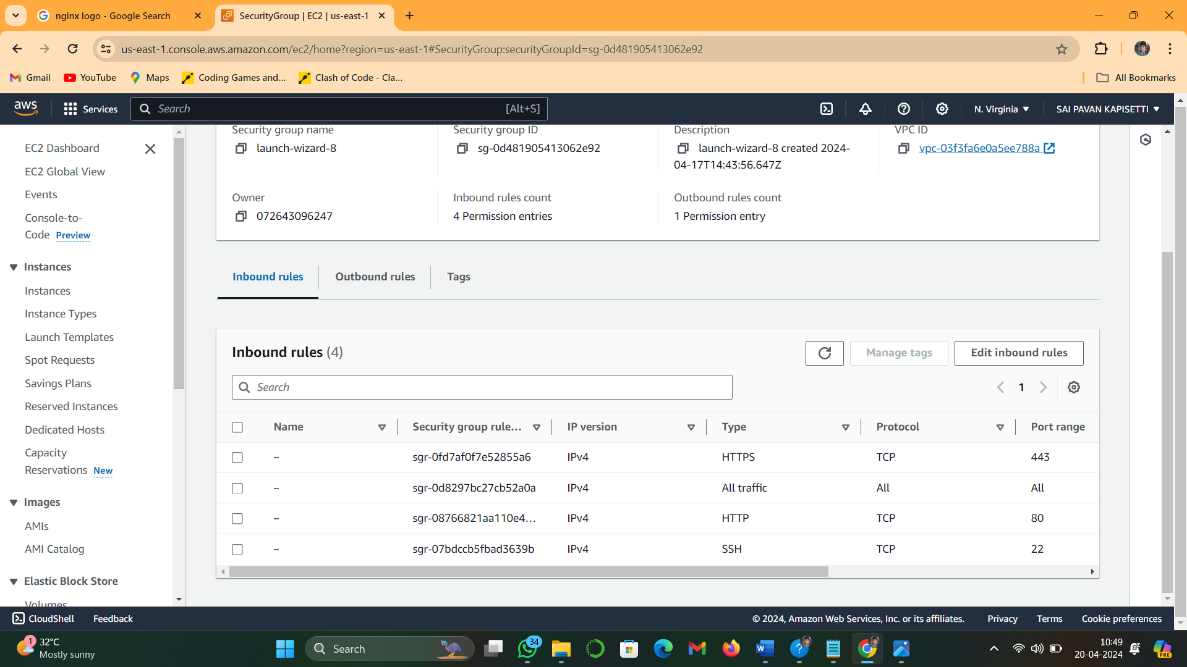
 Click Launch instnace.

Sucessfully we created the instnce. Wait for instance state to be as running

 State as shown below.

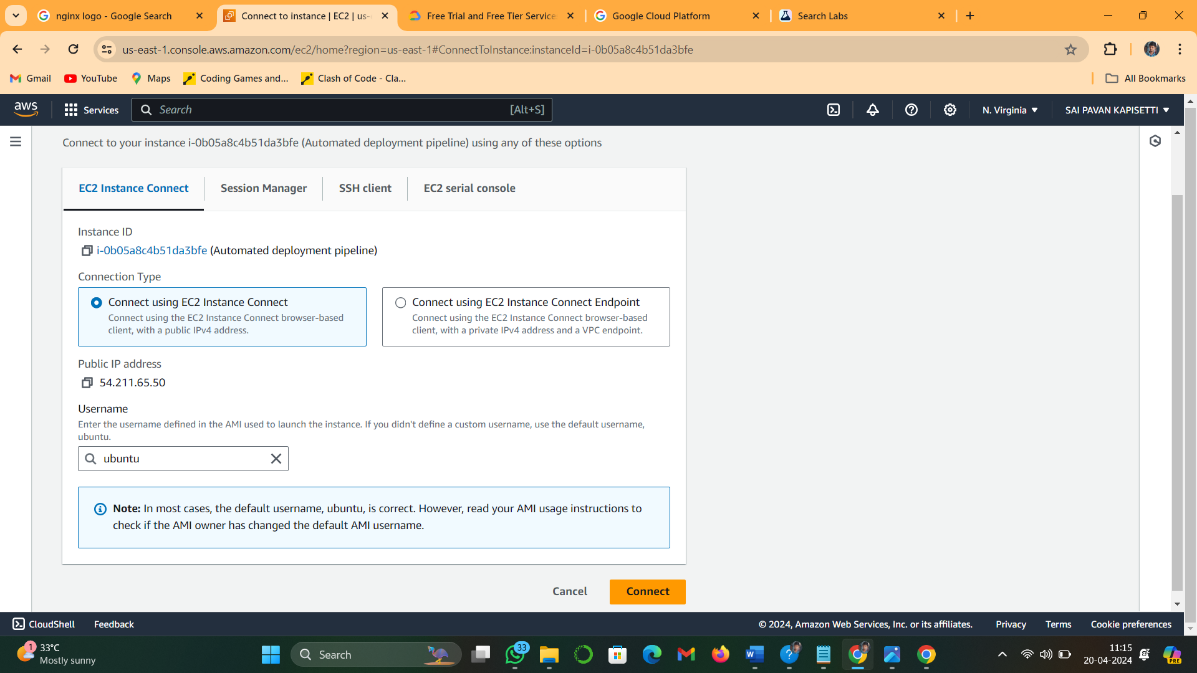
Now open the Instance and click on security and then security group edit

the inbound rules.



**2.Installing Jenkins In EC2:**

Connect through ssh or ec2 connect .

 After Connecting suucessfully.first we need to login as root user by using command

sudo-i .after we need to install java . for installation of java by following the below command.

sudo apt update

sudo apt install openjdk-11-jre

After we need to install jenkins follow the below commands.

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \

/usr/share/keyrings/jenkins-keyring.asc **>** /dev/null

echo deb **[**signed-by**=**/usr/share/keyrings/jenkins-keyring.asc] \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

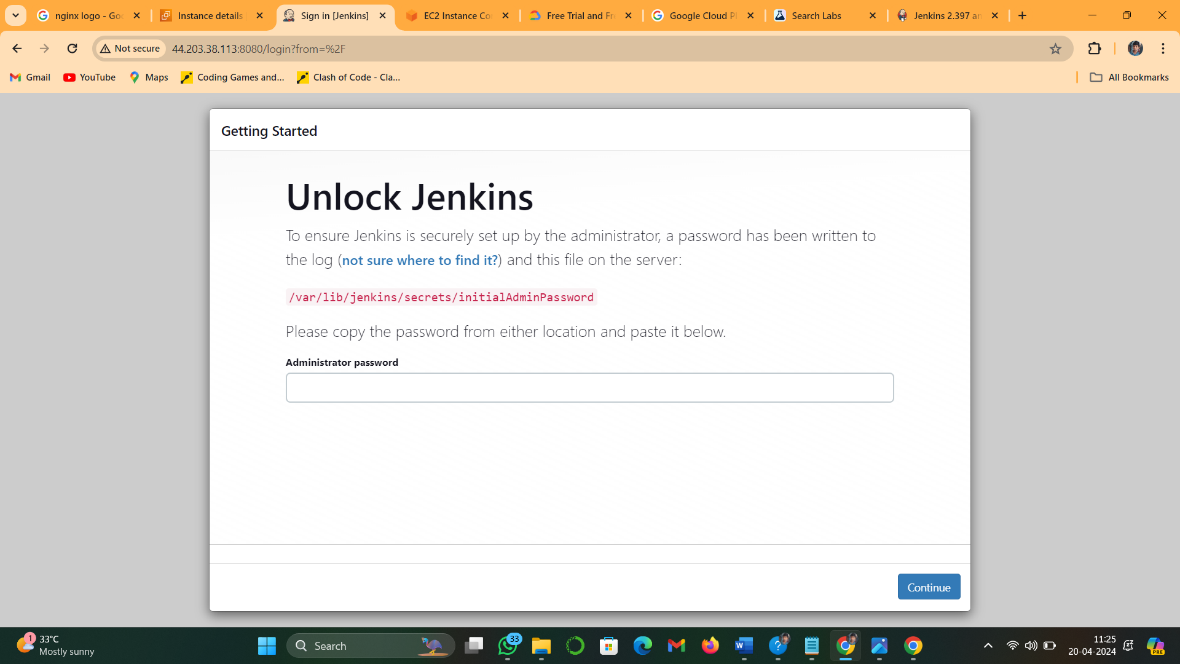
/etc/apt/sources.list.d/jenkins.list **>** /dev/null

After sucesfully installed the jenkins.

**Login to the Jenkins server:**

To login to the jenkins server.through port 8080

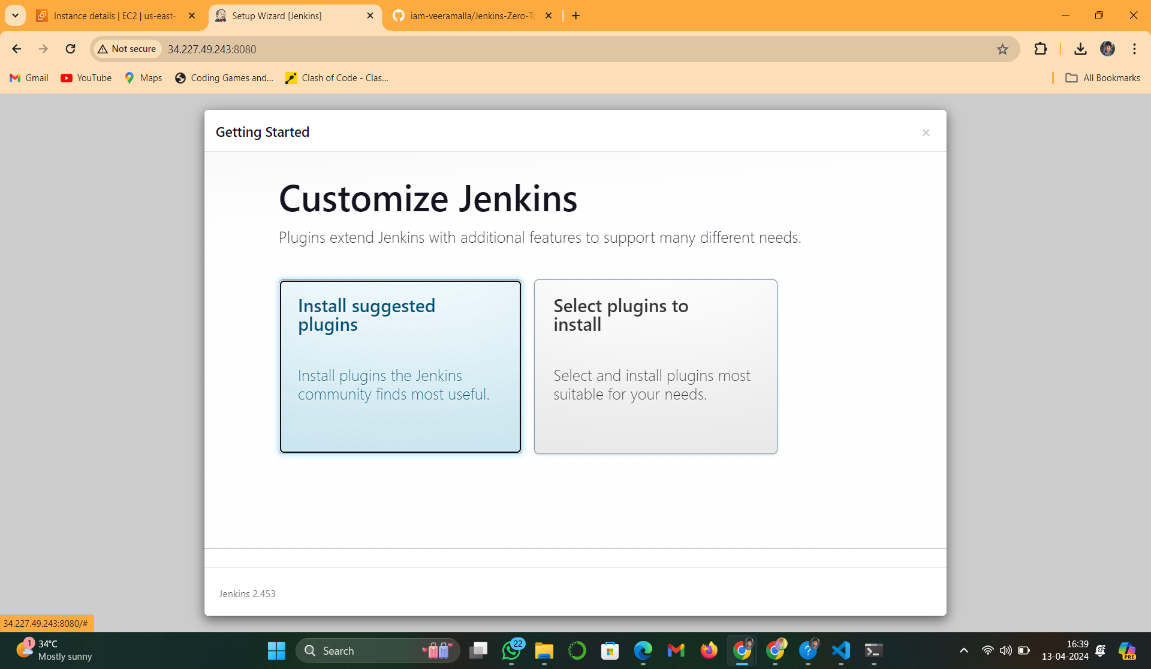
**http://ip-address:8080**(You can get the ec2-instance-public-ip-address from your AWS EC2 console page)

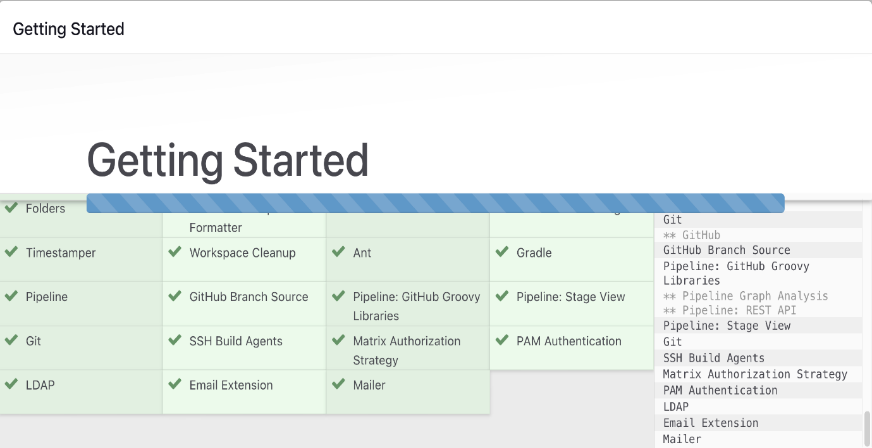
Now we will be acess to the jenkins page

initially We need to enter the adminstrator password to login.for adminstrator password

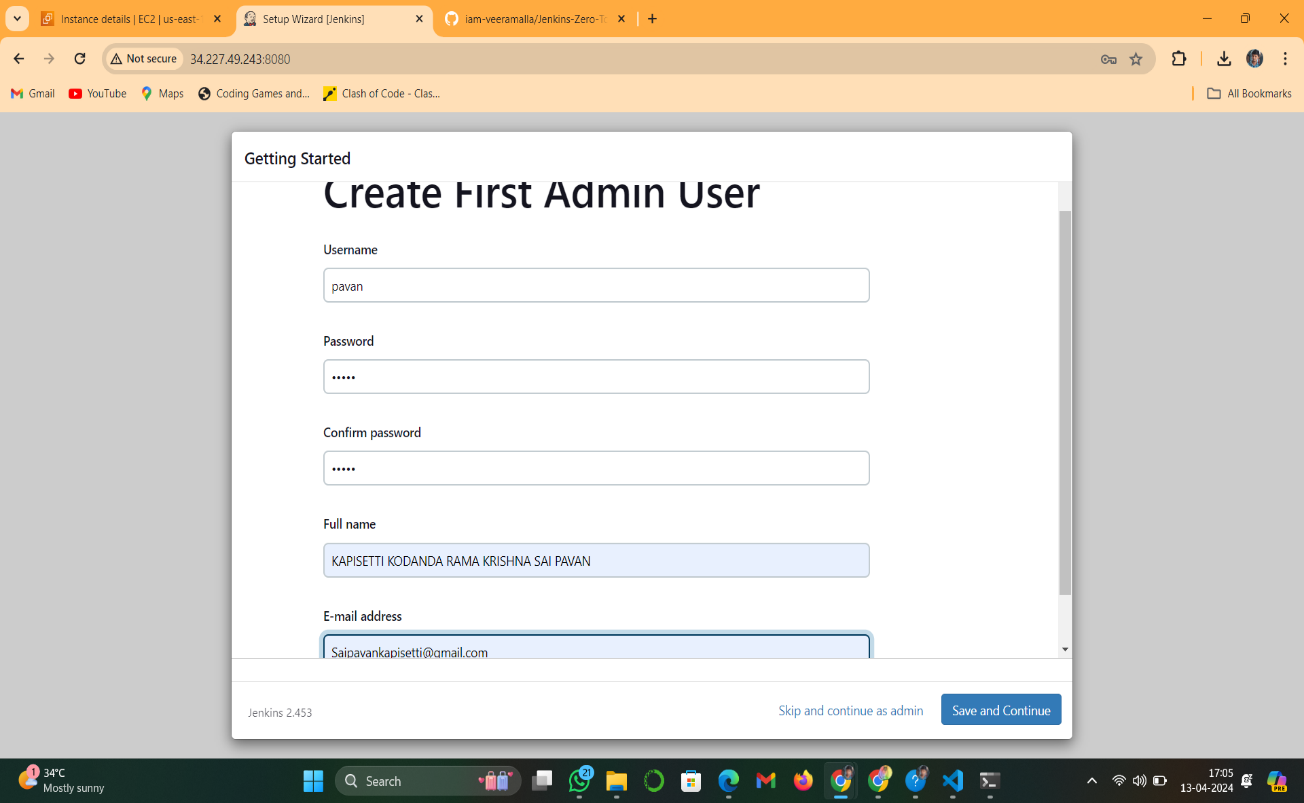
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

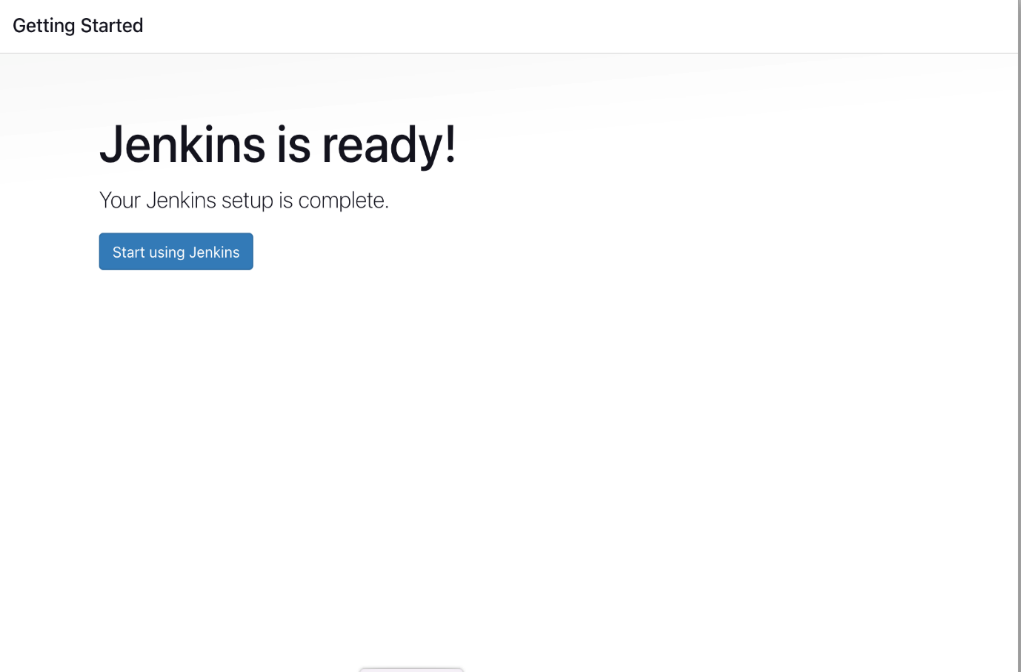
after enter the adminstrator password and install all suggested plugins



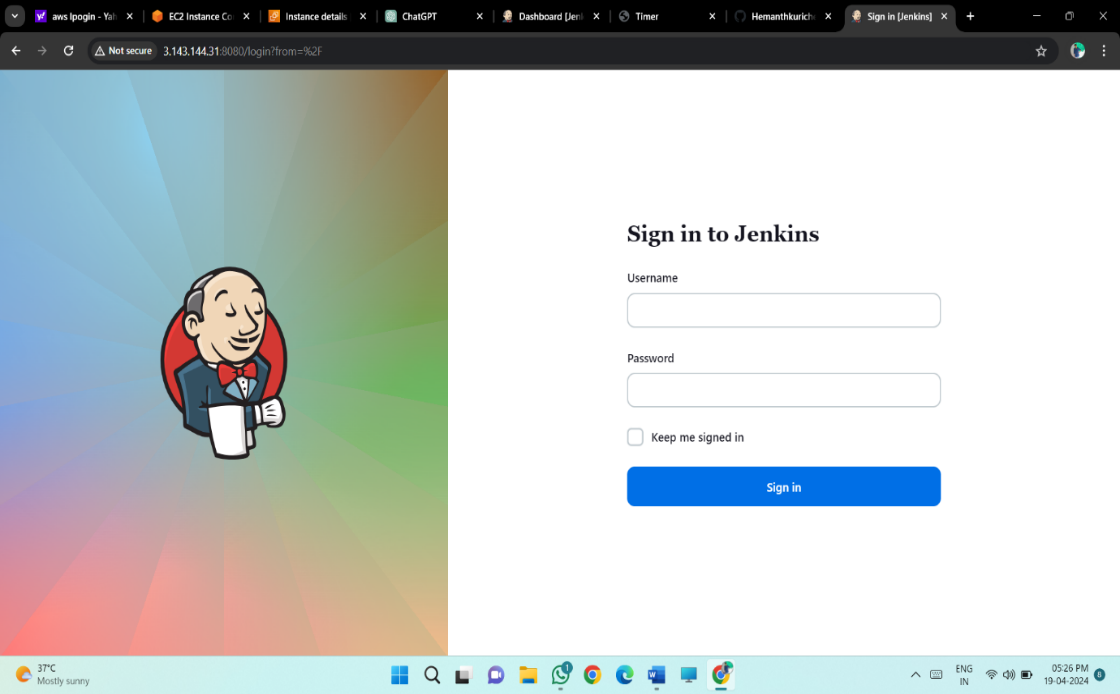


After installing all the plugins then create your login credentials

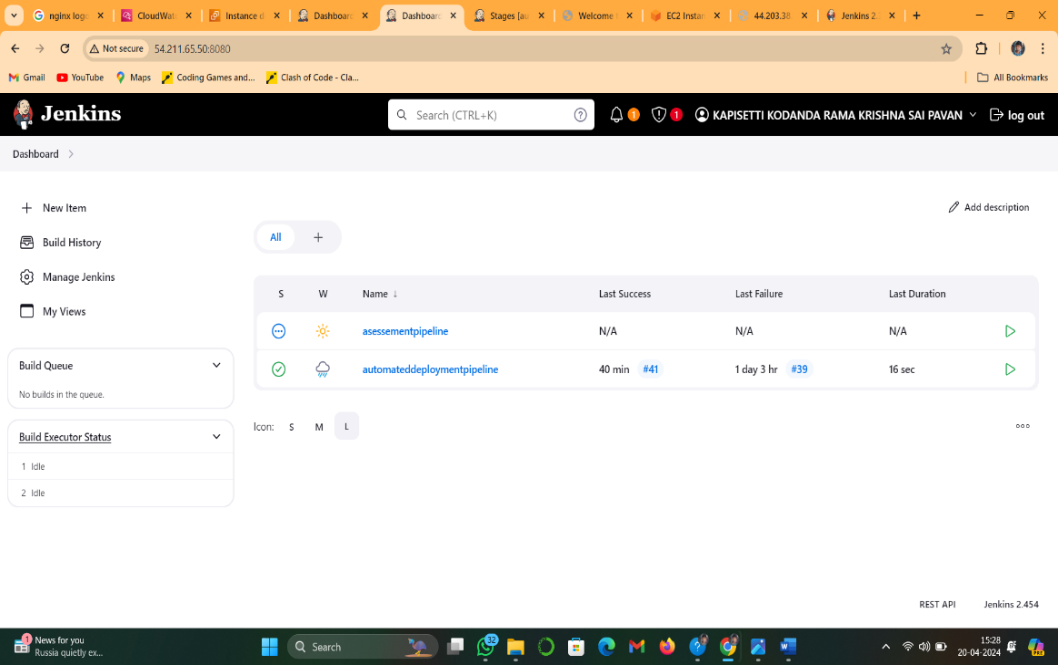


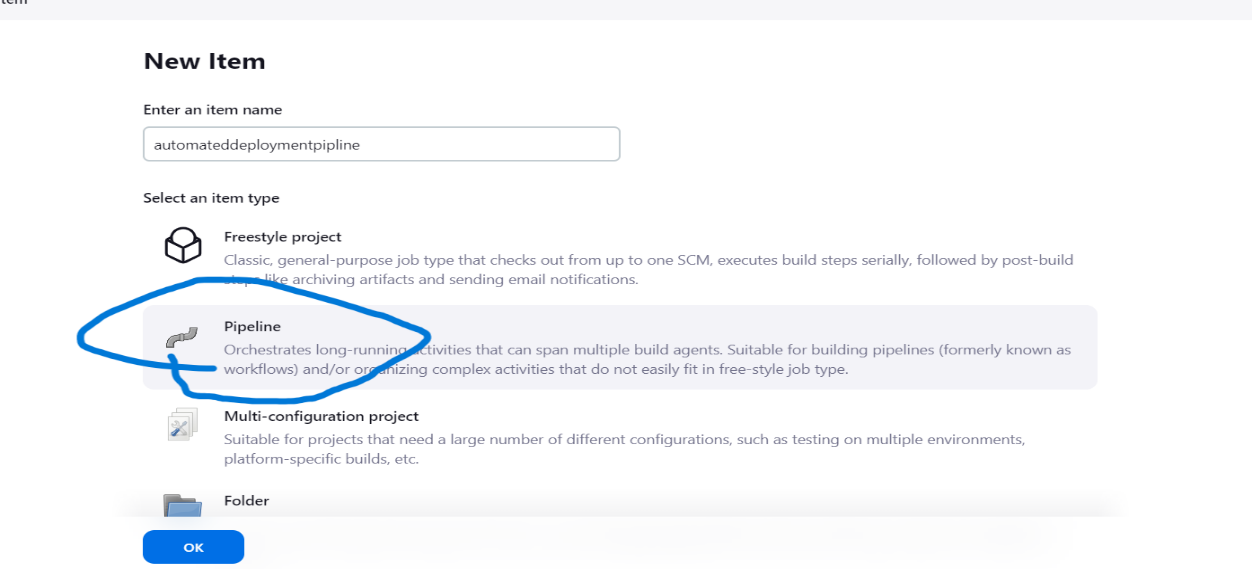


Click on start using Jenkins.Then login to your jenkins page

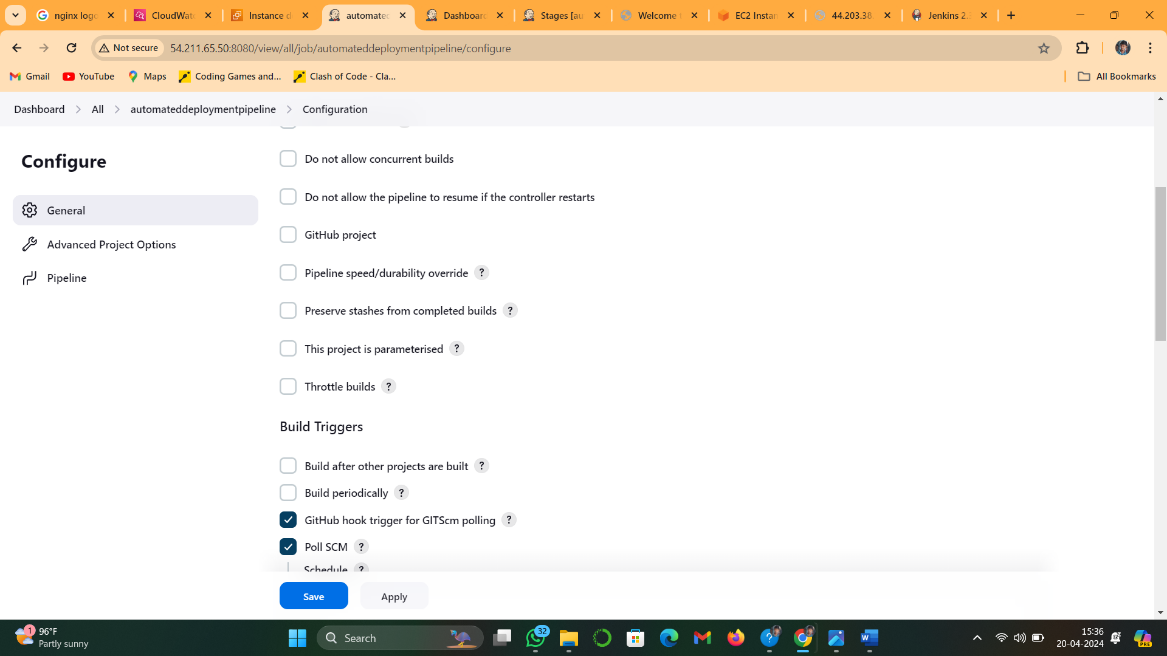


**3.Deploying Application:**

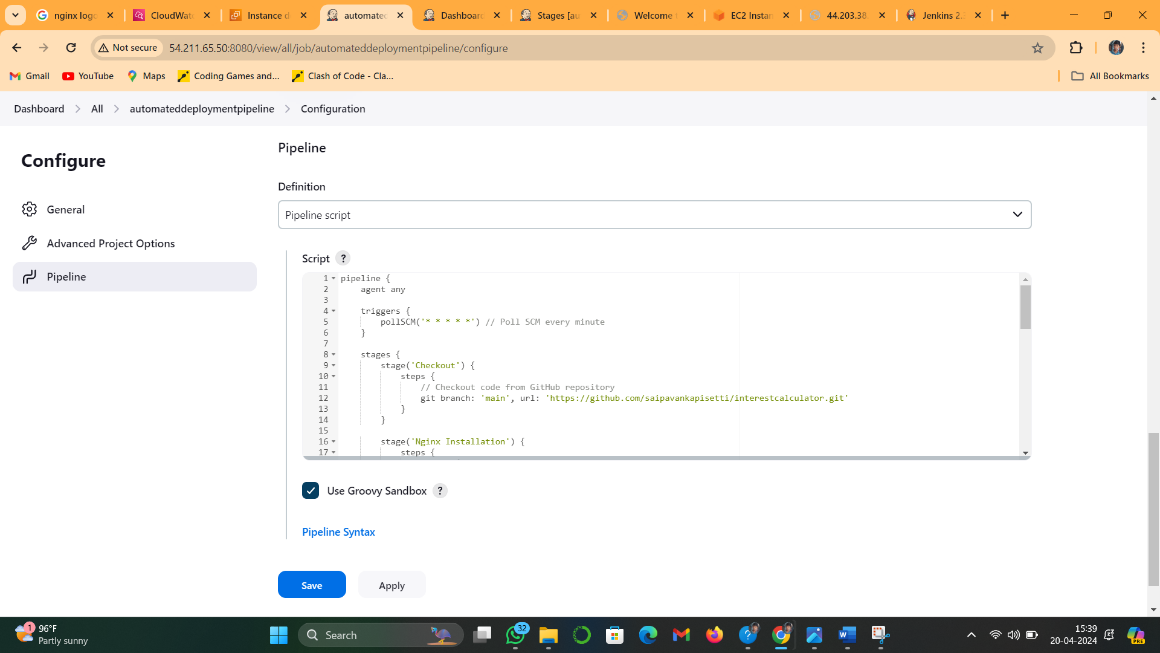
After login sucessfully .click on new item



And then click on pipeline project and give the project name.

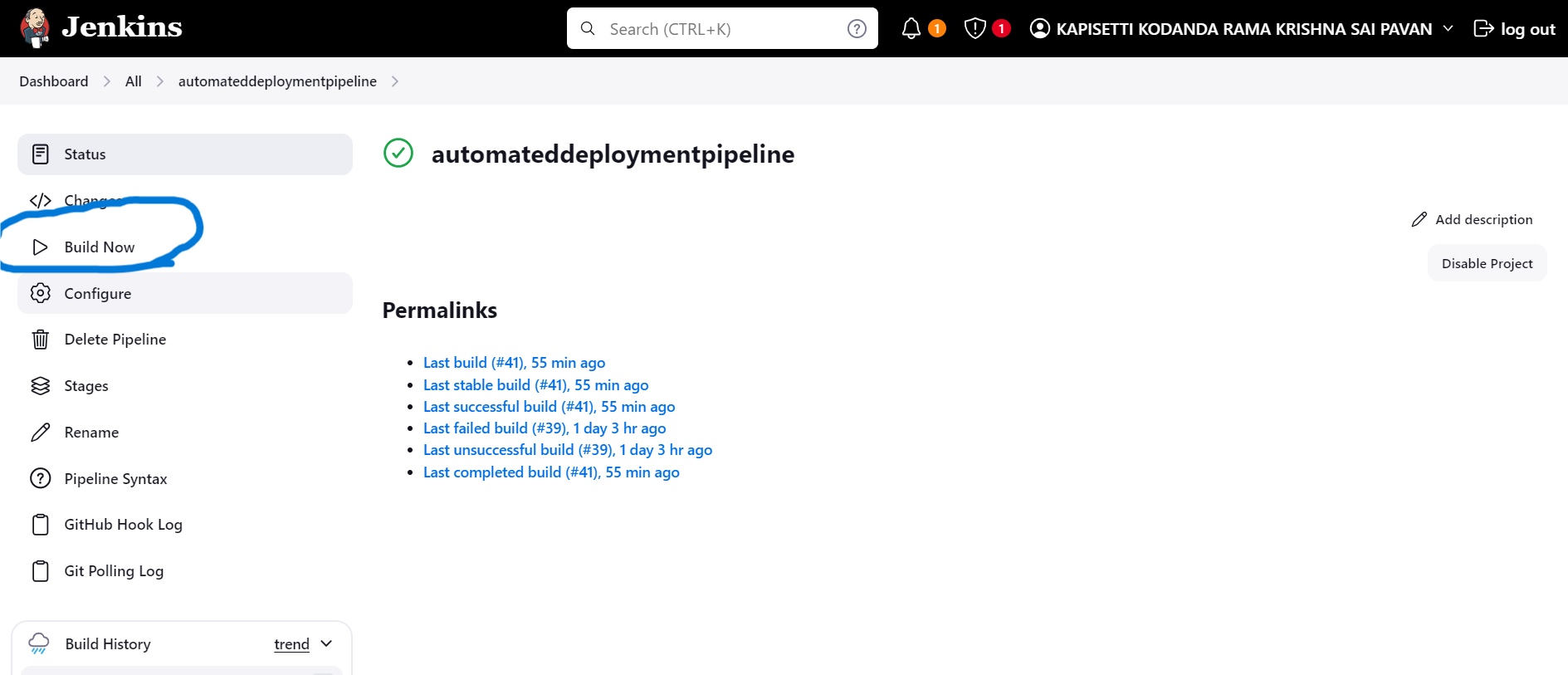
After creating sucessfully open a project .click on configrue and Enable Github hook trigger for GITSCM polling.

And scroll down to and add the pipeline script.

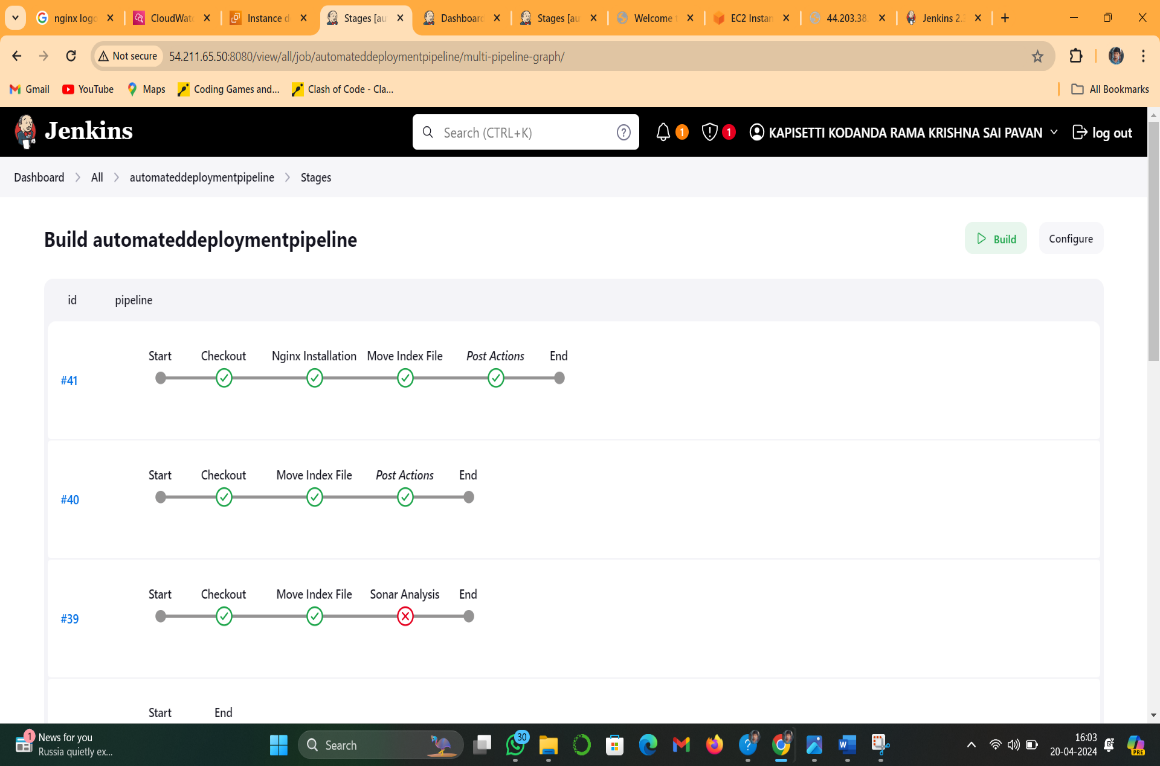


after writing a pipeline script and click on apply and save it .

after sucessfully configrue the build steps now click on build now

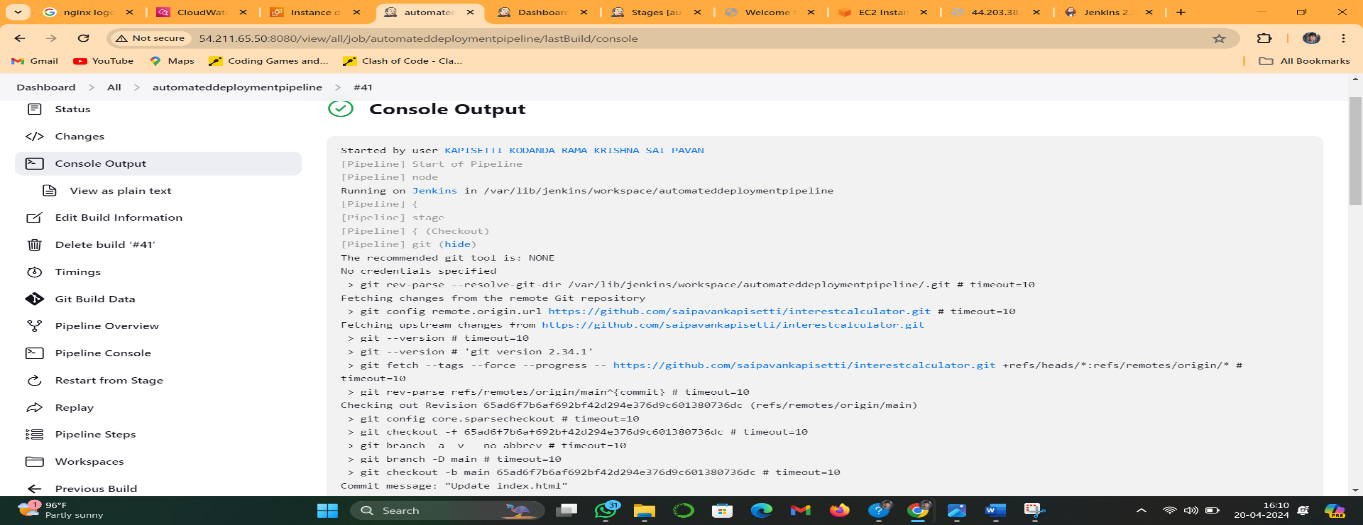


After build sucessfully . View stages



Sucessfully build the pipeline

if pipeline was failed you need to check errors in console output



**The pipeline Script used in my project:**

pipeline {

agent any

triggers {

pollSCM('\* \* \* \* \*')

}

stages {

stage('Checkout') {

steps {

git branch: 'main', url: 'https://github.com/saipavankapisetti/interestcalculator.git'

}

}

stage('Nginx Installation') {

steps {

script {

try {

sh 'sudo apt update && sudo apt install nginx -y'

sh 'sudo systemctl start nginx'

sh 'sudo systemctl status nginx'

} catch (Exception e) {

currentBuild.result = 'FAILED'

error "Failed to install and start Nginx: ${e.message}"

}

}

}

}

stage('Move Index File') {

steps {

script {

try {

sh 'sudo mv index.html /var/www/html/index.nginx-debian.html'

currentBuild.result = 'SUCCESS'

} catch (Exception e) {

echo "Failed to move index.html: ${e.message}"

currentBuild.result = 'FAILURE'

error "Failed to move index.html"

}

}

}

}

}

post {

success {

echo "Pipeline completed successfully!"

}

failure {

echo "Pipeline failed!"

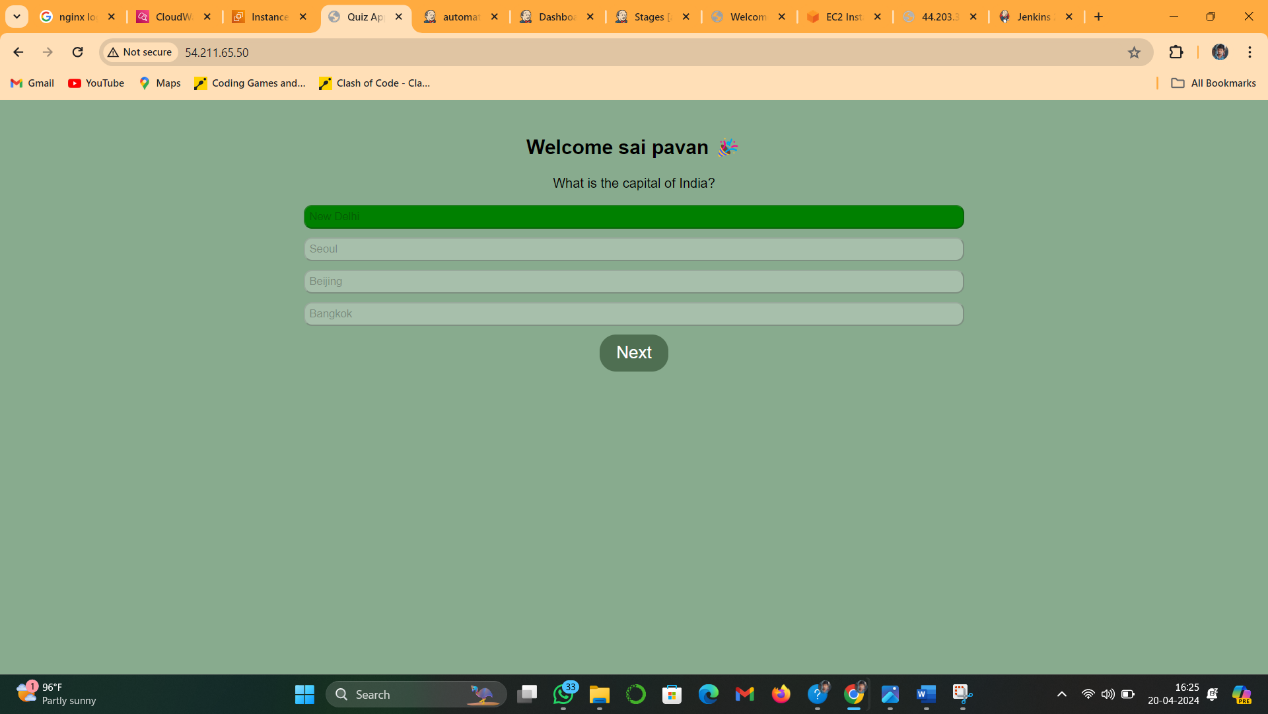
}

}

}

Our application was sucessfully deployed. For acess the application

Go to the instance and copy the public ipv4 address and open in a browser to verify the output.



Sucessfully run the application.

**4.Monitoring Capabilities:**

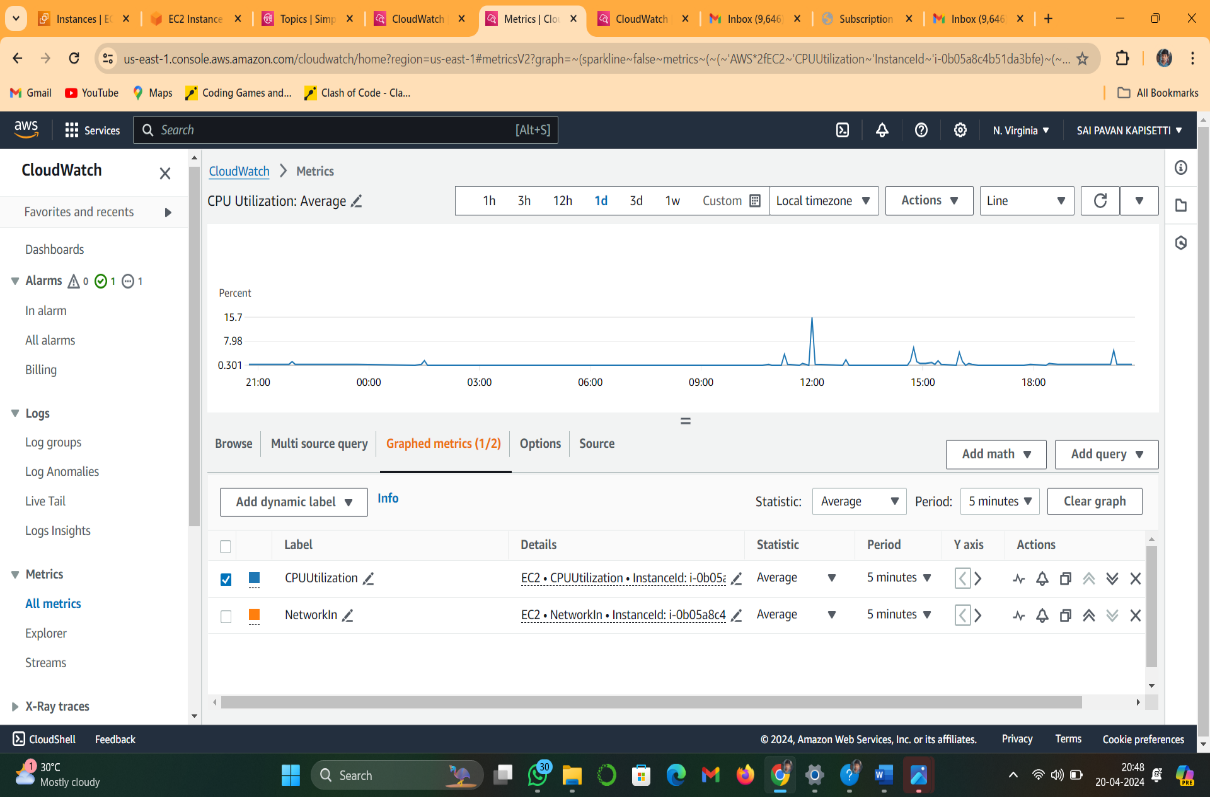
For monitoring cpu utilization and RAM utilization using cloud watch

Open the cloud watch and click on metrics section and monitor the graphs about cpu utilization ,RAM utilization and network in etc.

Create a Alarm for required

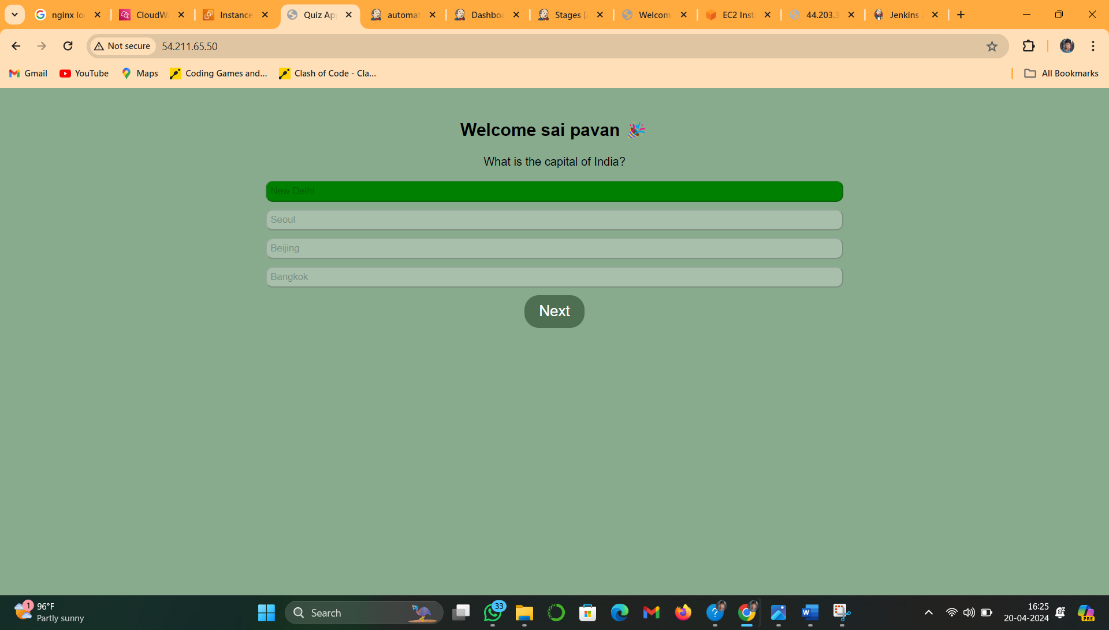
In my case I create an alarm for when cpu utilization is threshold value is greater than 200 percent.and then alarm will be activated.

Cpu utilization Graphs are shown below



**OUTPUT:**

Here is the final output.



The application deployed on nginx server using Ec2 instance ,it contains

Html,Css,Javascript