1.Aim: Demonstrate working with GitHub using GitHub Portal.

Source Code:

GitHub is a web-based platform that uses Git for version control. It is widely used by software developers for collaborative coding and project management. Here's an overview of the basic functionalities of GitHub.

Repository (Repo): A repository is a container for a project, where all the project files, history, and documentation are stored. Repositories can be public (visible to everyone) or private (accessible only to specified collaborators).

Branching: Git enables branching, allowing developers to create different lines of development. Branches can be used for features, bug fixes, or experiments without affecting the main codebase.

Commits: A commit is a snapshot of changes made to the code.

Pull Requests: Pull Requests (PRs) allow developers to propose changes from one branch to another. They facilitate code review, discussion, and collaboration before changes are merged.

Issues: GitHub Issues are used to track tasks, enhancements, bugs, or any other kind of work. Issues can be assigned, labeled, and commented on, providing a centralized location for project communication.

Merging: Merging is the process of combining changes from one branch into another.

Forks: Forking a repository creates a copy of the original repository in the user's account.

Collaborators: Collaborators are individuals with access to a repository. They can push changes, review and merge pull requests, and manage issues.

GitHub Actions: GitHub Actions automate workflows, allowing users to define custom CI/CD (Continuous Integration/Continuous Deployment) processes directly within the repository.

Wiki: Repositories can have a wiki for documentation. It's a place to store and share information about the project, providing additional context and resources.

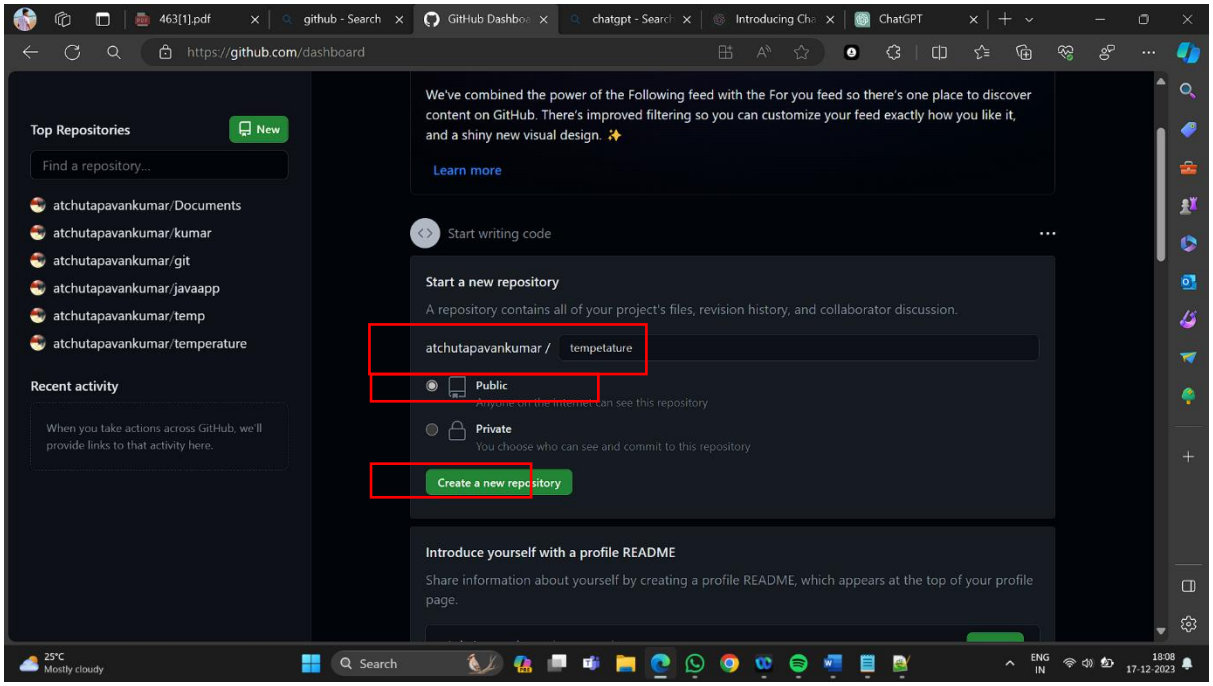
GitHub Pages: GitHub Pages enables the hosting of static websites directly from a GitHub repository.

Gists: Gists are like mini-repositories for sharing and collaborating on single files or snippets.

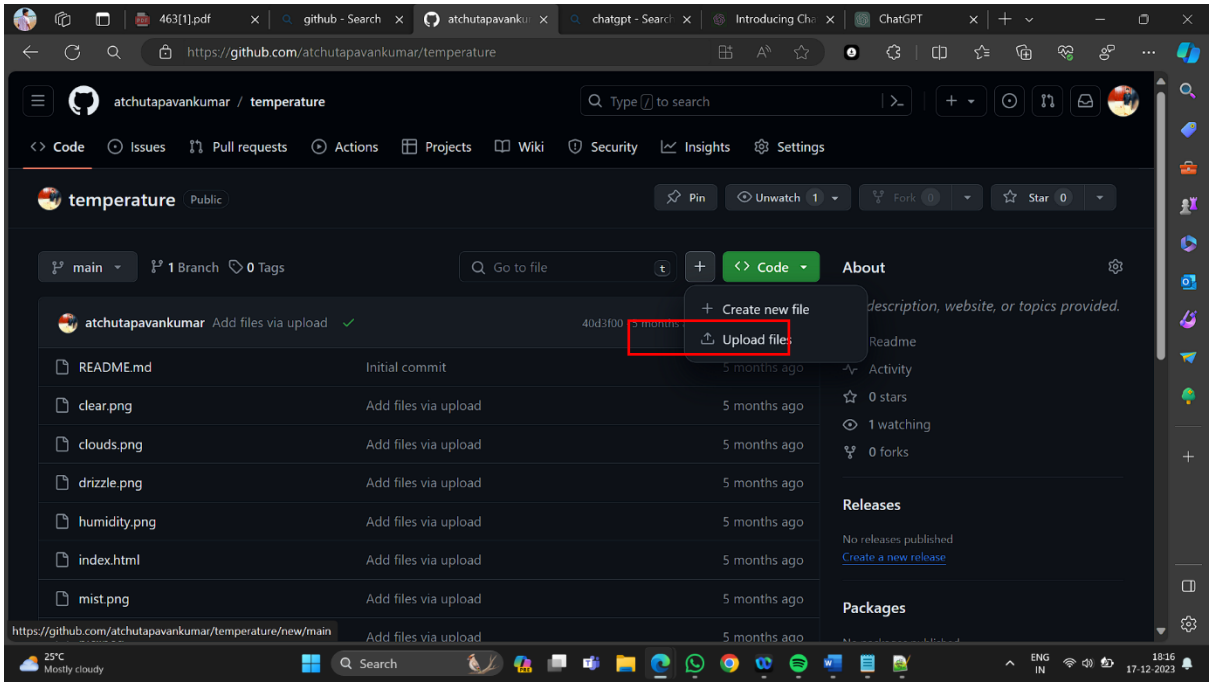
Steps to deploy a new repository on GitHub:

Step-1: Logon to your GitHub account on browser. Create one if you don’t have a GitHub account.

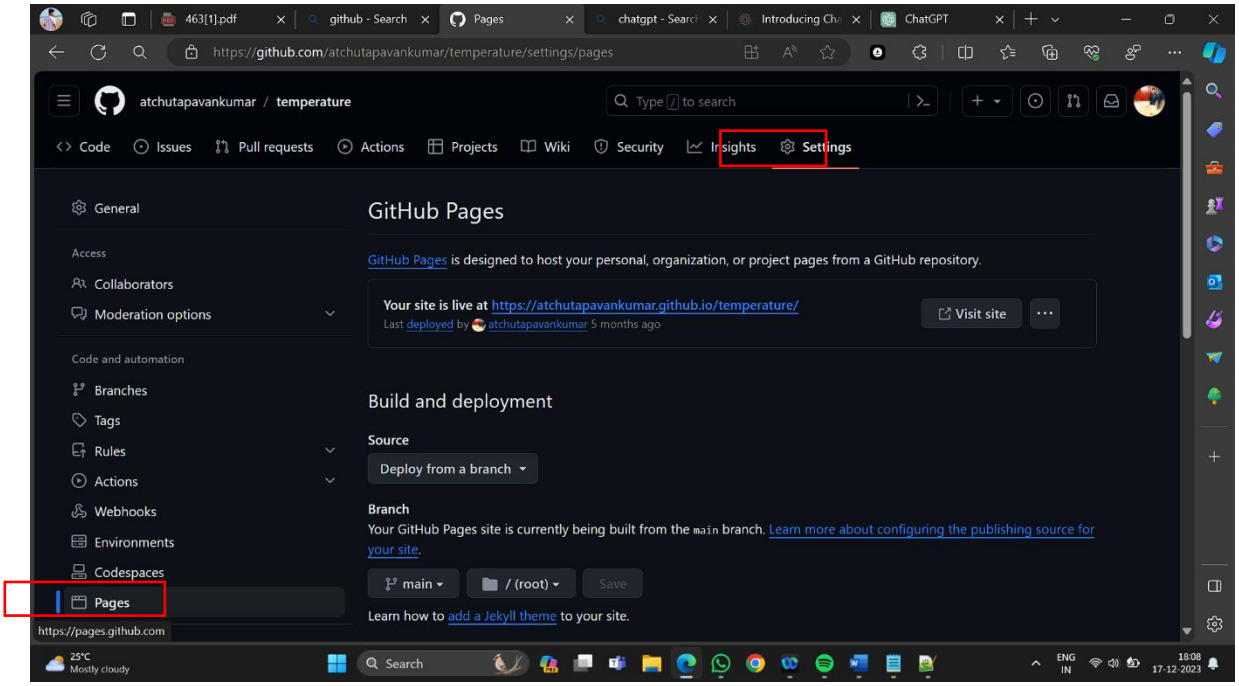
Step-2: To create a new repository, goto **Start a new repository** column and give the repository name. Make sure the repository is **Public** and add a **ReadMe file** to the repository.



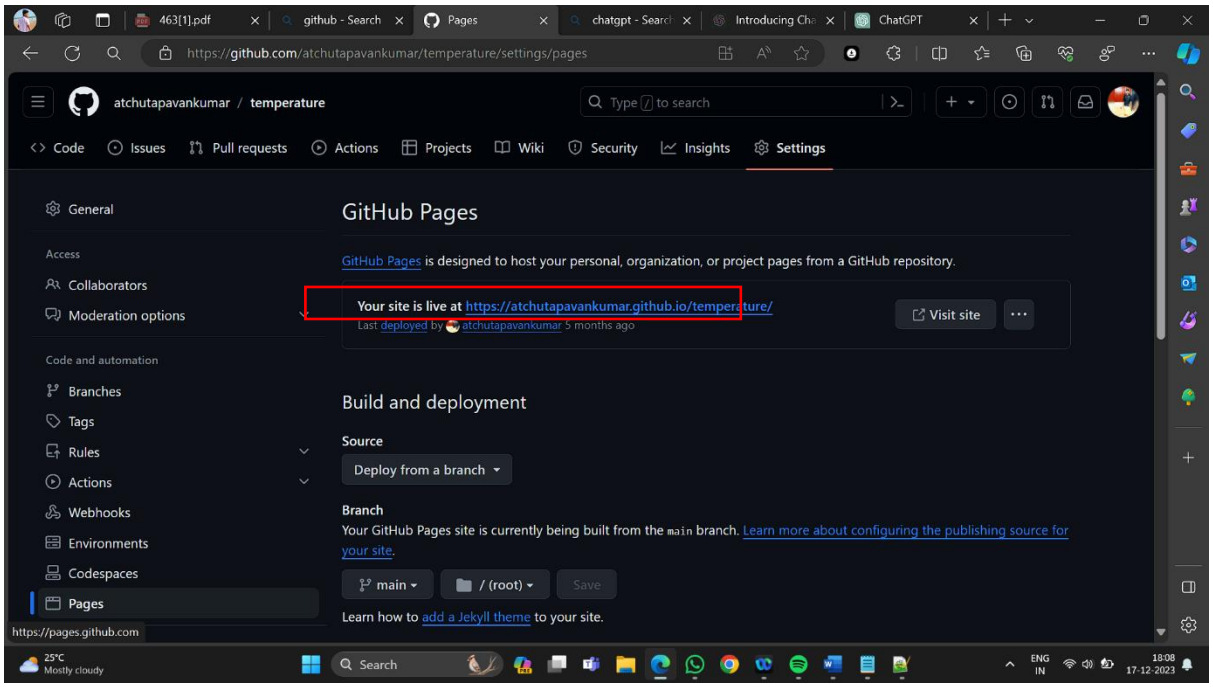
Step-3: Now open the repository and upload your files onto the repository.



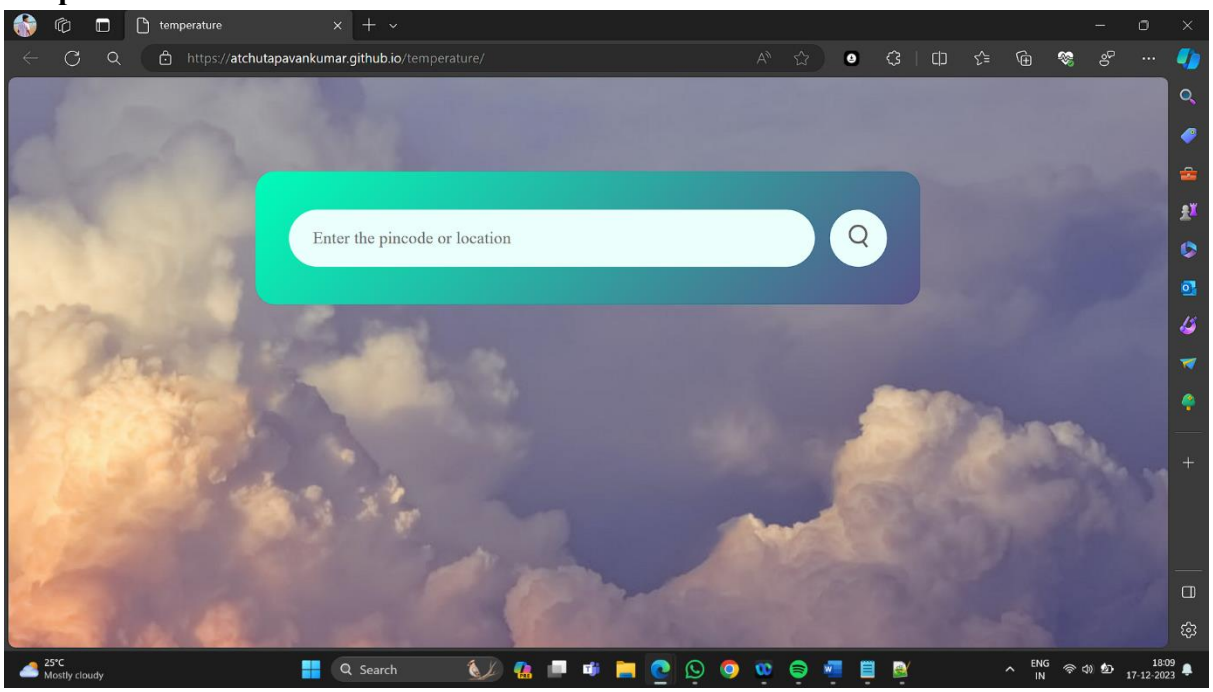
Step-4: Open the repository **settings** and goto **pages** under **code and automation** settings.



Step-5: Copy the live link of uploaded file and paste it on browser window to view your uploaded file.



Output:


temperature
https://atchutapavankumar.github.io/temperature/
Enter the pincode or location
25°C
Mostly cloudy
Search
ENG
IN
18:09
17-12-2023

2.Aim: Demonstrate working with Git Shell Commands

Source Code:

1.Git init:

To start using git,navigate to the directory where your project is located and run

```
D:\Devops>git init
Initialized empty Git repository in D:/Devops/.git/
```

2.Adding files to staging area:

You can add your files to the staging area to prepare them for a commit using the 'git add' command for example to add a file called 'myfile.txt'

```
D:\Devops>git init
Initialized empty Git repository in D:/Devops/.git/

D:\Devops>git add myfile.txt
```

3.Commiting changes:

Once you have added files to the staging area you can commit them to repository with a commit message.using 'git commit' command

```
D:\Devops>git commit -m "your commit changes"
[master (root-commit) e747d12] your commit changes
 1 file changed, 1 insertion(+)
 create mode 100644 myfile.txt
```

4.Checking Status:

To see the status of your repository and which files have been modified, added or deleted use 'git status'

```
D:\Devops>git status
On branch master
nothing to commit, working tree clean
```

5.Viewing commit history:

You can view the commit history of your repository using 'git log'

```
e747d1241f9e33ae983822c7c589ab84faef061c (HEAD -> master)

Author: Narendr <narendramurukurthi@gmail.com>

Date:  Sun Dec 3 19:26:28 2023 +053
```

6.creating and switching Branches:

To create a new branch and switch to it use 'git checkout -b'

```
D:\Devops>git checkout -b mynewbranch
Switched to a new branch 'mynewbranch'
```

7.Merging Branches:

To Merge changes from one branch to another use the ‘git merge’ command.for example to merge changes from feature-branch into another branch.

```
D:\Devops>git merge newbranch
Already up to date.
```

8.cloning a repository:

To clone a remote git repository to your local machine use the ‘git clone’
git clone https://narendra/documents

```
Cloning into 'Documents'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 37 (delta 5), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (37/37), 1.91 MiB | 2.72 MiB/s, done.
Resolving deltas: 100% (5/5), done.
```

9.pushing changes to a remote repository:

To push your local changes to a remote repository use ‘git push origin main’

```
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up D:\Devops>git push -u origin main
to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 335 bytes | 335.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/narendra/myfile.git
 44c9bc1..133e0f3  main -> main
branch 'main' set up to track 'origin/main'.
```

10.pulling changes from a remote repository:

To get the latest changes from a remote repository use the ‘git pull’ command
git pull https://narendra/documents

```
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (16/16), done.
Unpacking objects: 100% (18/18), 110.28 KiB | 1.38 MiB/s, done.
remote: Total 18 (delta 0), reused 0 (delta 0), pack-reused 0
```

3. Aim: Demonstrate using Jenkins Service with Git plugin in AWS

Source Code:

Step-1: Creating the aws EC2 instances:

1.Log in to the AWS Management Console.

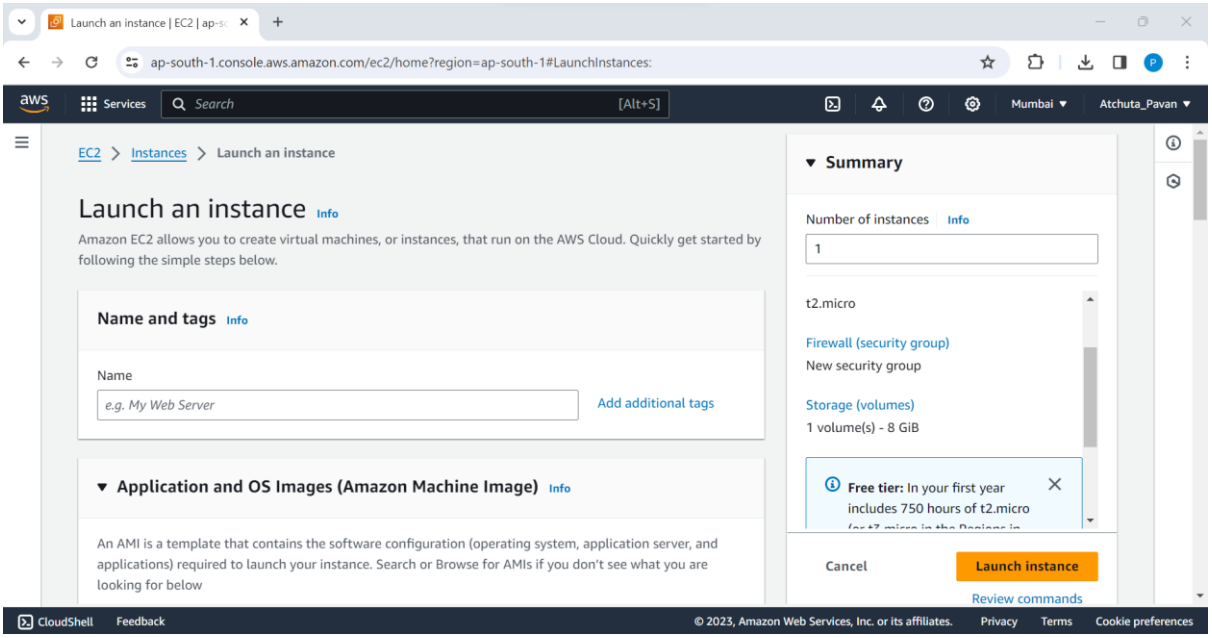
2.Navigate to the EC2 service.

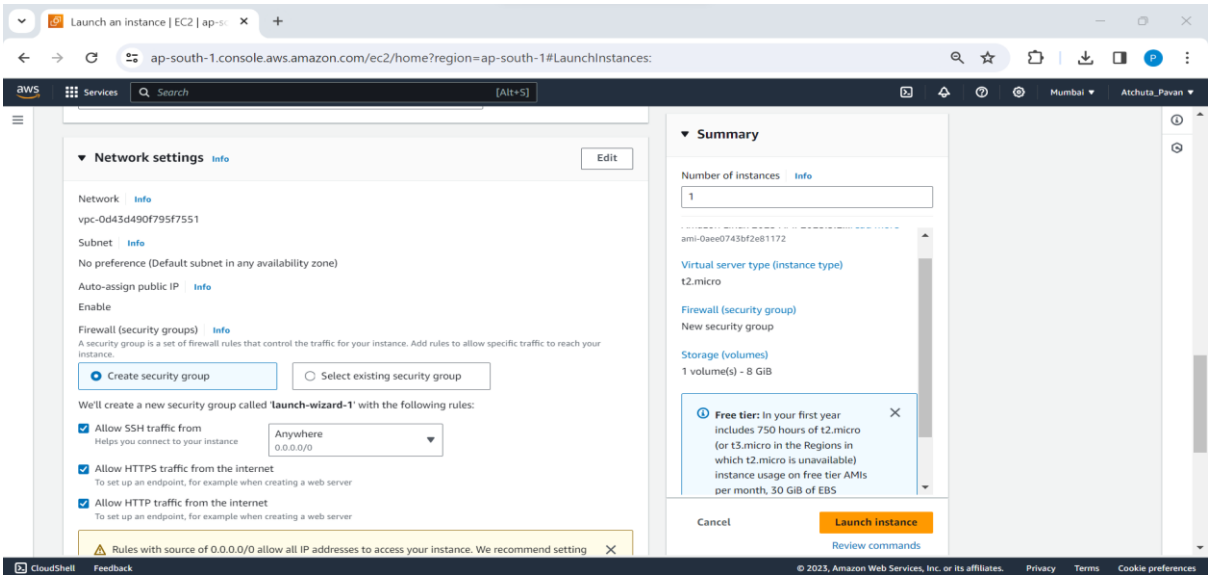
3.Click on “Launch Instance” to create a new EC2 instance.

4.Select the desired instance type, storage, and other configurations.

5.Configure the security group to allow inbound traffic on port 22 for SSH access.

6.Launch the instance and download the private key file (.pem).





Step-2:SSH into EC2 Instance, Install Java and Jenkins

SSH into the EC2 instance. After securely connecting to the instance, the first thing we have to do is to install the Java application onto the instance. Jenkins runs natively runs

on the Java application so it has to be properly installed in order for the Jenkins server to be operational.

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

```
[ec2-user@ip-172-31-13-239 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2023-12-17 07:10:03--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/jenkins.r 100%[=====================================>]      85  --.-KB/s    in 0s

2023-12-17 07:10:03 (5.74 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-13-239 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-13-239 ~]$ sudo yum upgrade
Jenkins-stable                                                  264 kB/s |  28 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
```

2. sudo dnf install java-17-amazon-corretto -y

```
[ec2-user@ip-172-31-13-239 ~]$ sudo dnf install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:06 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package                    Arch       Version                    Repository   Size
================================================================================
Installing:
 java-17-amazon-corretto    x86_64     1:17.0.9+8-1.amzn2023.1    amazonlinux  188 k
Installing dependencies:
 alsa-lib                   x86_64     1.2.7.2-1.amzn2023.0.2     amazonlinux  504 k
 cairo                      x86_64     1.17.6-2.amzn2023.0.1      amazonlinux  684 k
 dejavu-sans-fonts          noarch     2.37-16.amzn2023.0.2       amazonlinux  1.3 M
 dejavu-sans-mono-fonts     noarch     2.37-16.amzn2023.0.2       amazonlinux  467 k
 dejavu-serif-fonts         noarch     2.37-16.amzn2023.0.2       amazonlinux  1.0 M
 fontconfig                 x86_64     2.13.94-2.amzn2023.0.2     amazonlinux  273 k
 fonts-filesystem           noarch     1:2.0.5-12.amzn2023.0.2    amazonlinux  9.5 k
 freetype                   x86_64     2.13.0-2.amzn2023.0.1      amazonlinux  422 k
 giflib                     x86_64     5.2.1-9.amzn2023.0.1       amazonlinux   49 k
 google-noto-fonts-common   noarch     20201206-2.amzn2023.0.2    amazonlinux   15 k
 google-noto-sans-vf-fonts  noarch     20201206-2.amzn2023.0.2    amazonlinux  492 k
 graphite2                  x86_64     1.3.14-7.amzn2023.0.2      amazonlinux   97 k
 harfbuzz                   x86_64     7.0.0-2.amzn2023.0.1       amazonlinux  868 k
 java-17-amazon-corretto-headless
                            x86_64     1:17.0.9+8-1.amzn2023.1    amazonlinux   91 M
 javapackages-filesystem    noarch     6.0.0-7.amzn2023.0.6       amazonlinux   12 k
 langpacks-core-font-en     noarch     3.0-21.amzn2023.0.4        amazonlinux   10 k
 libICE                     x86_64     1.0.10-6.amzn2023.0.2      amazonlinux   71 k
 libSM                      x86_64     1.2.3-8.amzn2023.0.2       amazonlinux   42 k
 libX11                     x86_64     1.7.2-3.amzn2023.0.4       amazonlinux  657 k
 libX11-common              noarch     1.7.2-3.amzn2023.0.4       amazonlinux  152 k
 libXau                     x86_64     1.0.9-6.amzn2023.0.2       amazonlinux   31 k
 libXext                    x86_64     1.3.4-6.amzn2023.0.2       amazonlinux   41 k
```

3. sudo yum install jenkins -y

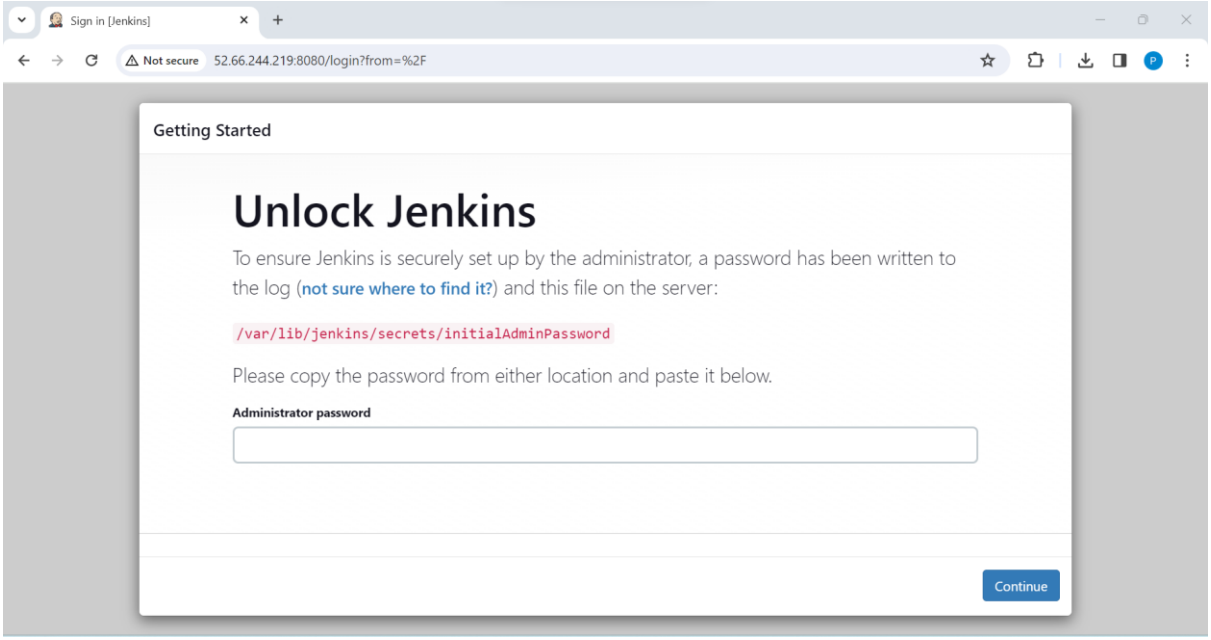
```
[ec2-user@ip-172-31-13-239 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:00:22 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package              Architecture          Version                  Repository              Size
================================================================================
Installing:
 jenkins              noarch                2.426.2-1.1              jenkins                 85 M

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

Total download size: 85 M
Installed size: 85 M
Downloading Packages:
jenkins-2.426.2-1.1.noarch.rpm                                  7.7 MB/s |  85 MB     00:11
--------------------------------------------------------------------------------
Total                                                           7.7 MB/s |  85 MB     00:11
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Installing       : jenkins-2.426.2-1.1.noarch                             1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Verifying        : jenkins-2.426.2-1.1.noarch                             1/1

Installed:
  jenkins-2.426.2-1.1.noarch
```

4. sudo systemctl enable jenkins
   sudo systemctl start jenkins

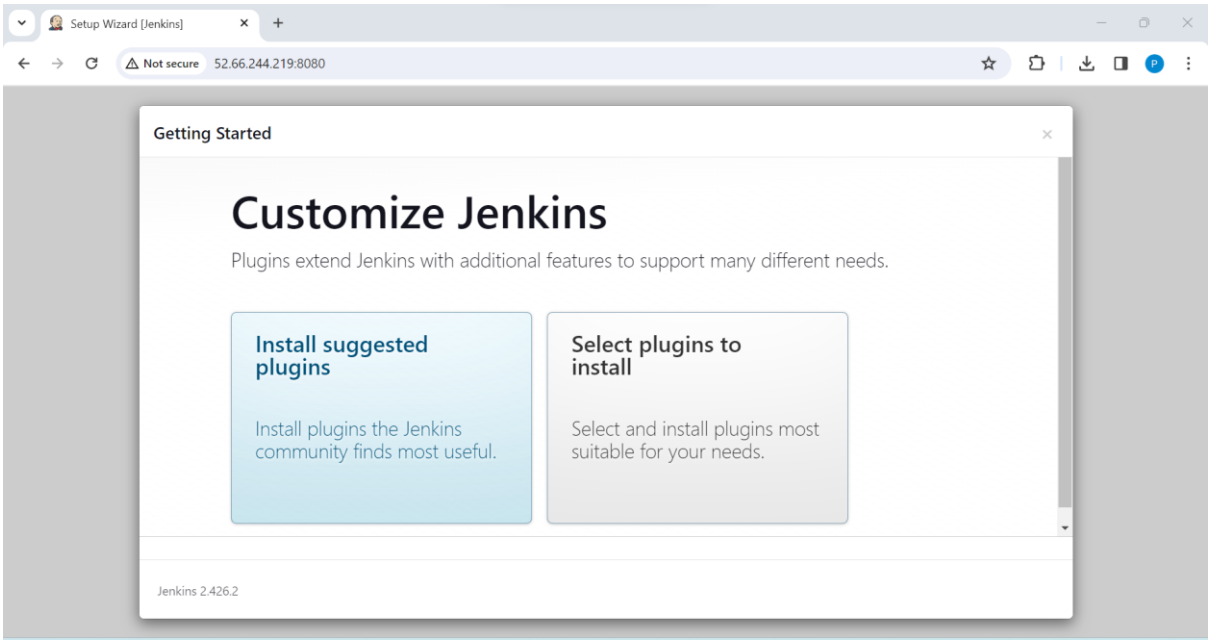
```
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jen
kins.service.
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl start jenkins
```

Step-3: Setting Up Jenkins Server

Now that the Jenkins server has been officially started on the instance, head over to the AWS Management console, grab the public IP address of the instance and paste that into your browser address bar followed by :8080




```
[ec2-user@ip-172-31-13-239 ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
baaef975c09e468a977274b59ed26a8e
[ec2-user@ip-172-31-13-239 ~]$
```



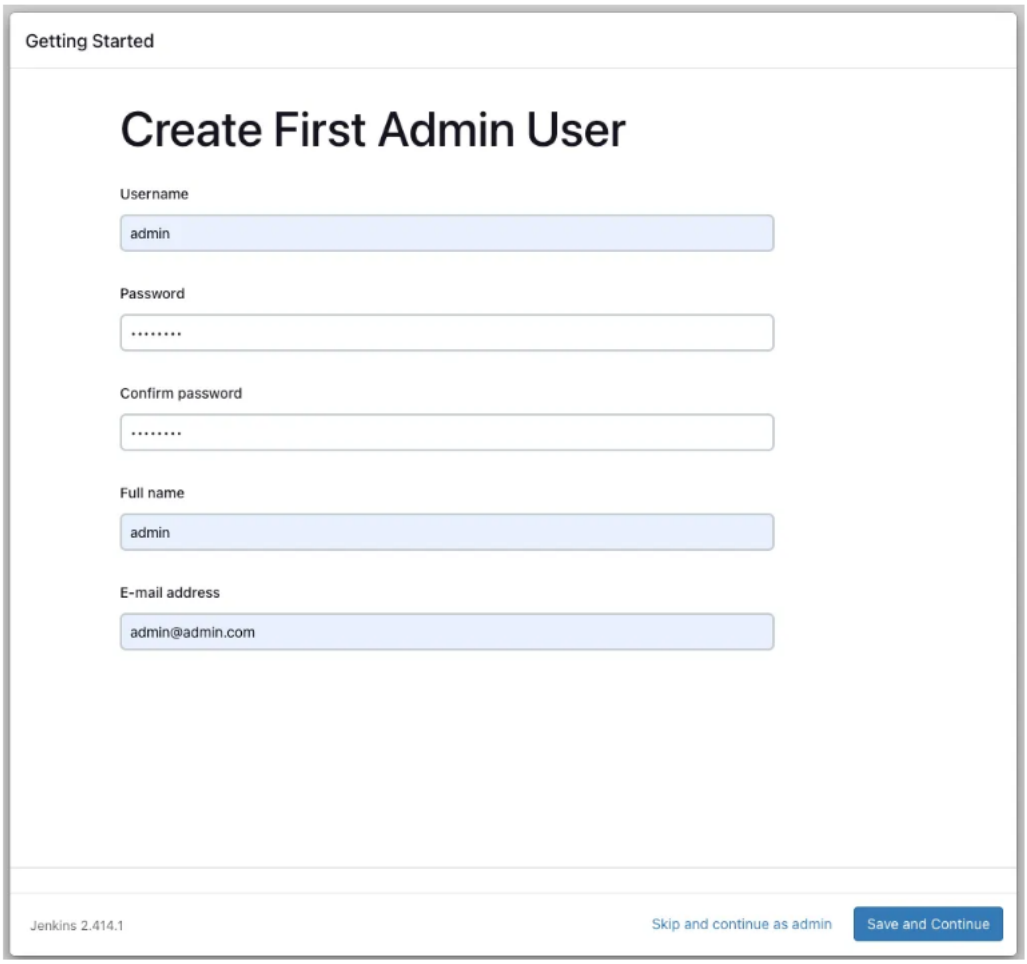
Setup an admin user. For this project we will get a dummy administrator user. In a real production you will use different credentials. Save and continue.

Username=admin

Password=Password

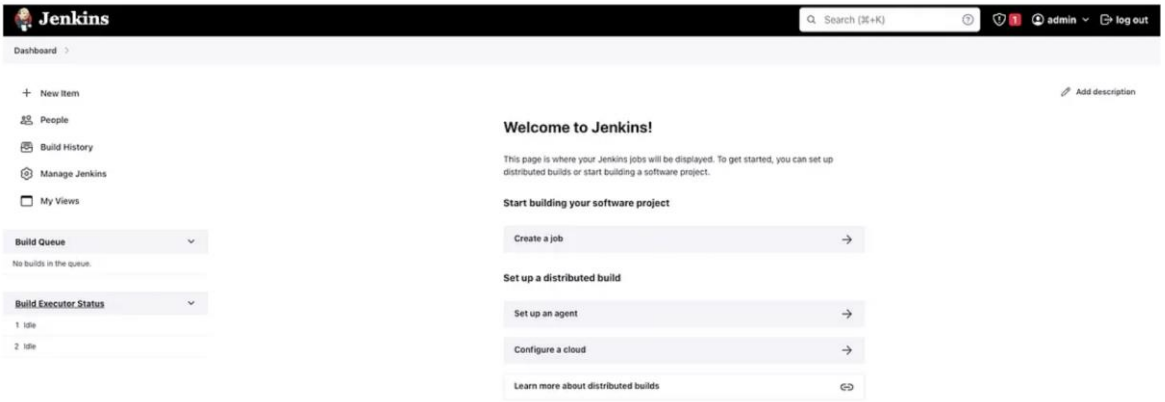
Full name= admin

Email=admin@admin.com

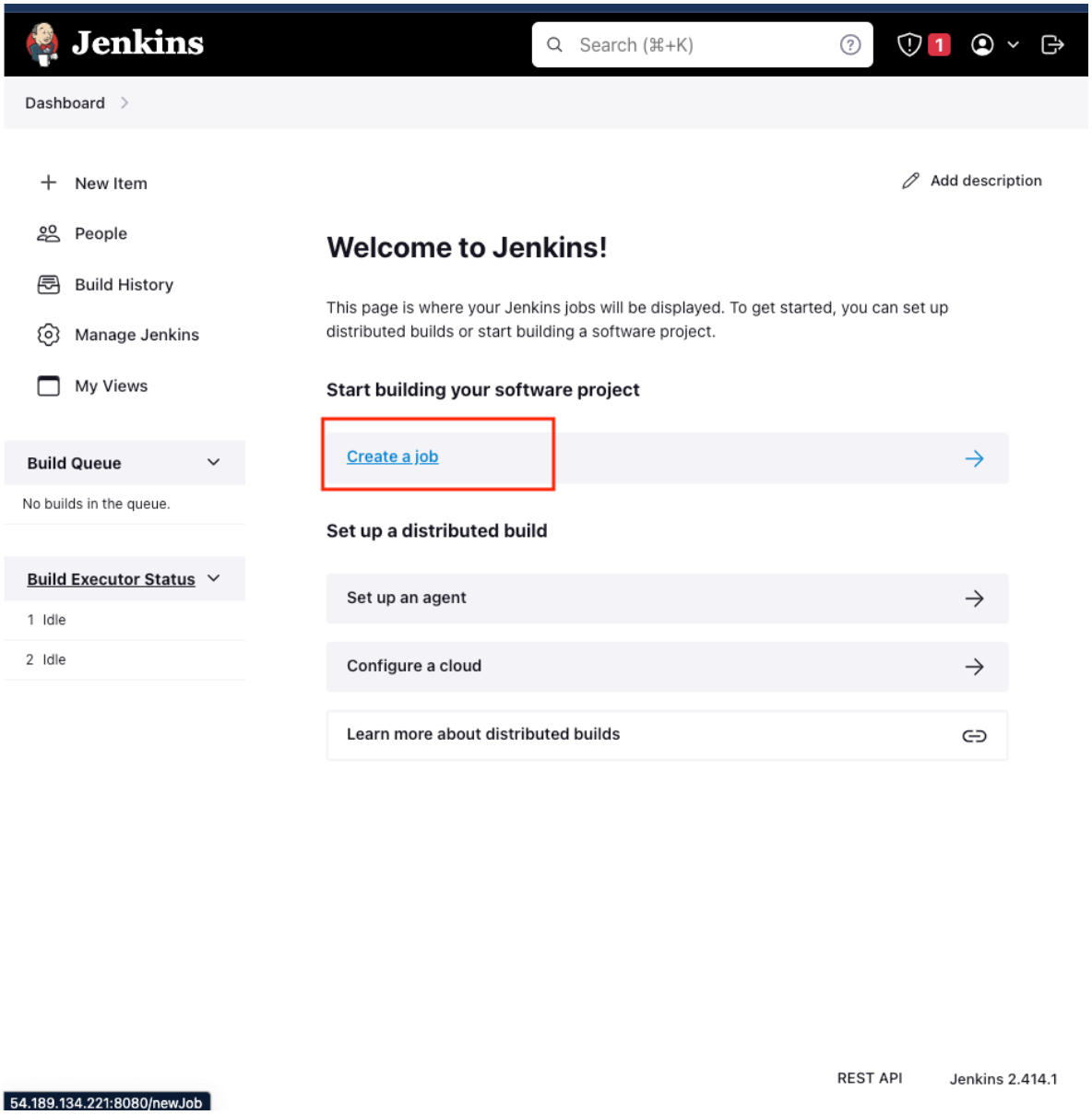


Step 4: We are Live and Jenkins Ready! Time to Build

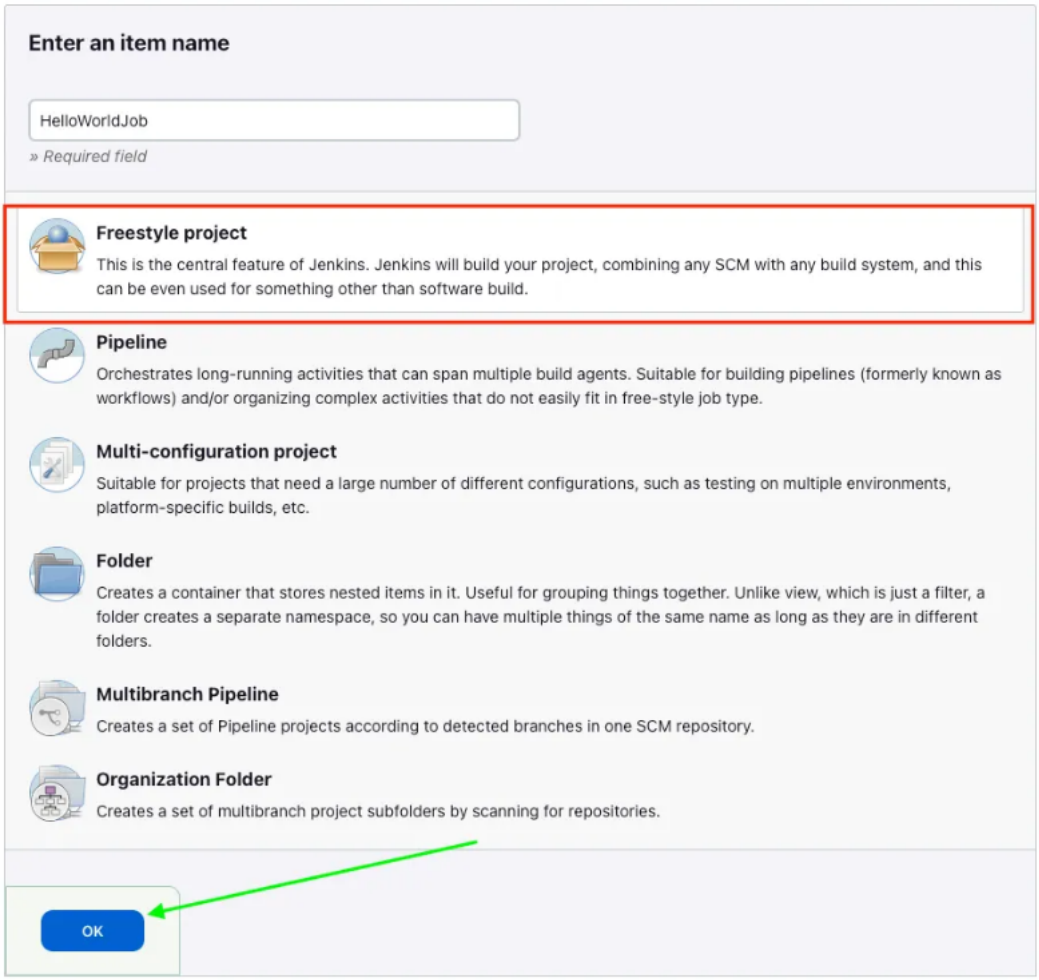
If you see the “Welcome to Jenkins” page, then you’ve successfully setup Jenkins and are ready to go to the next stage of the project, which is setting up our first Jenkins job!



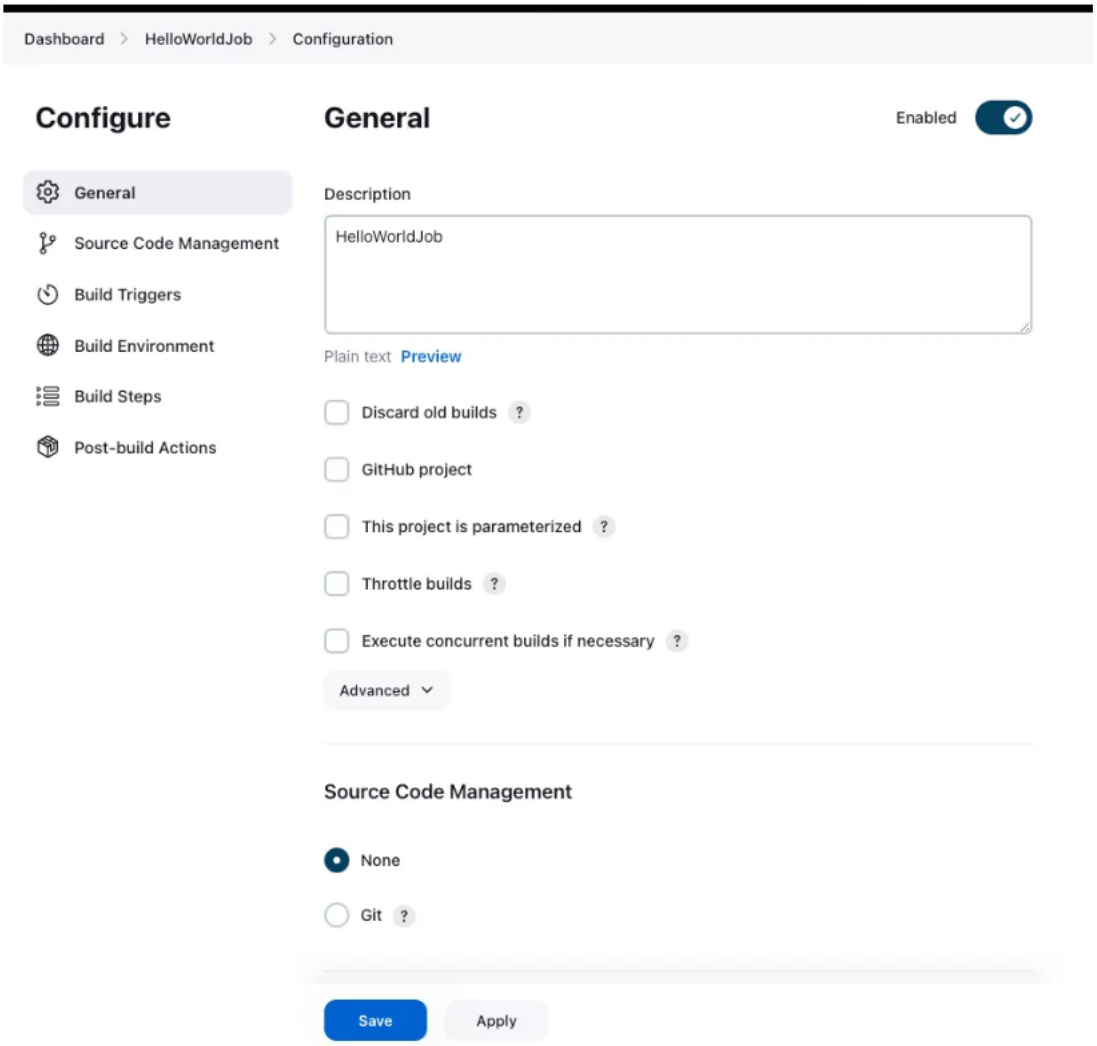
Creating our first jenkin job



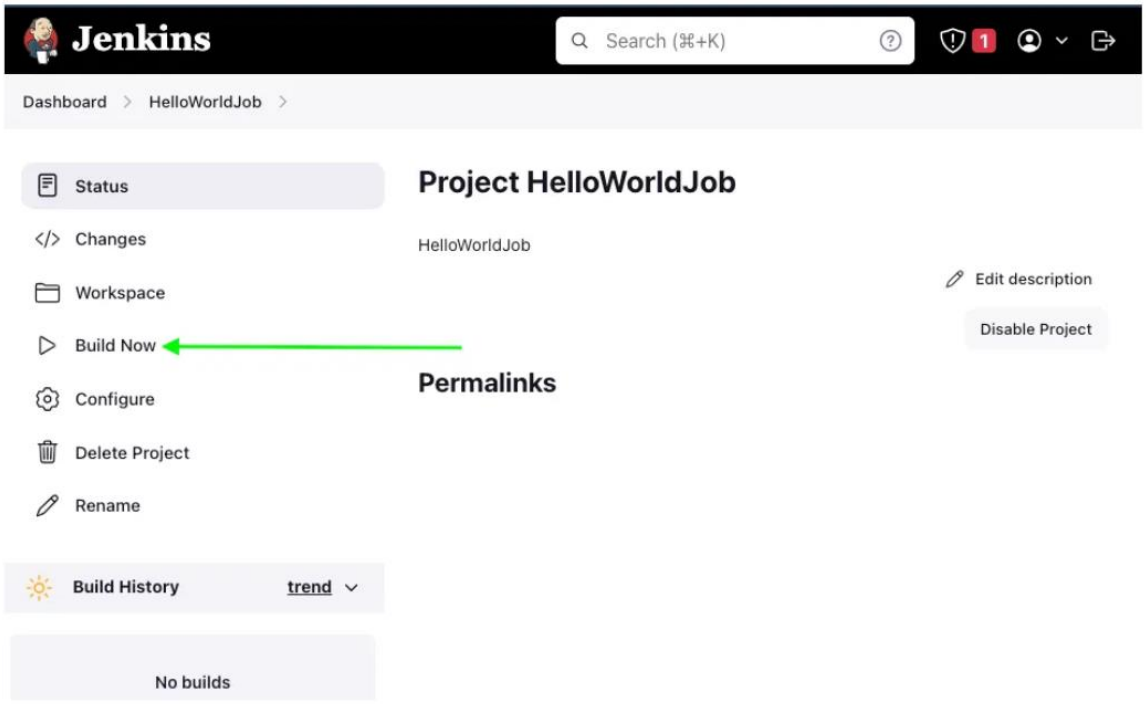
For our first job we are going to give it the name "**HelloWorldJob**" and select the option for "**Freestyle project**". Click ok to move to the next step.

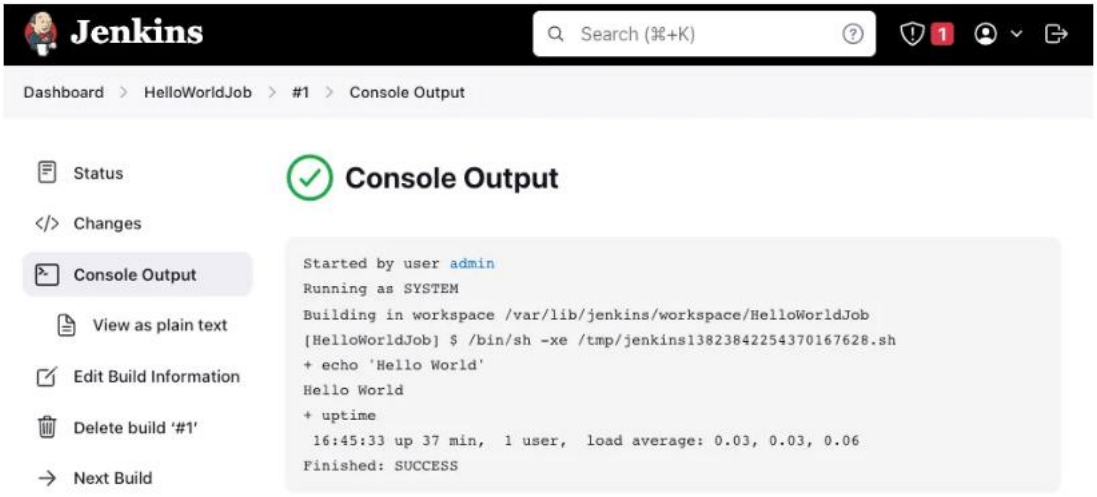


Step-5: Under the general configuration settings
In the description box, enter the text “HelloWorldJob”
Under the “Source Code Management” tab select “None” option

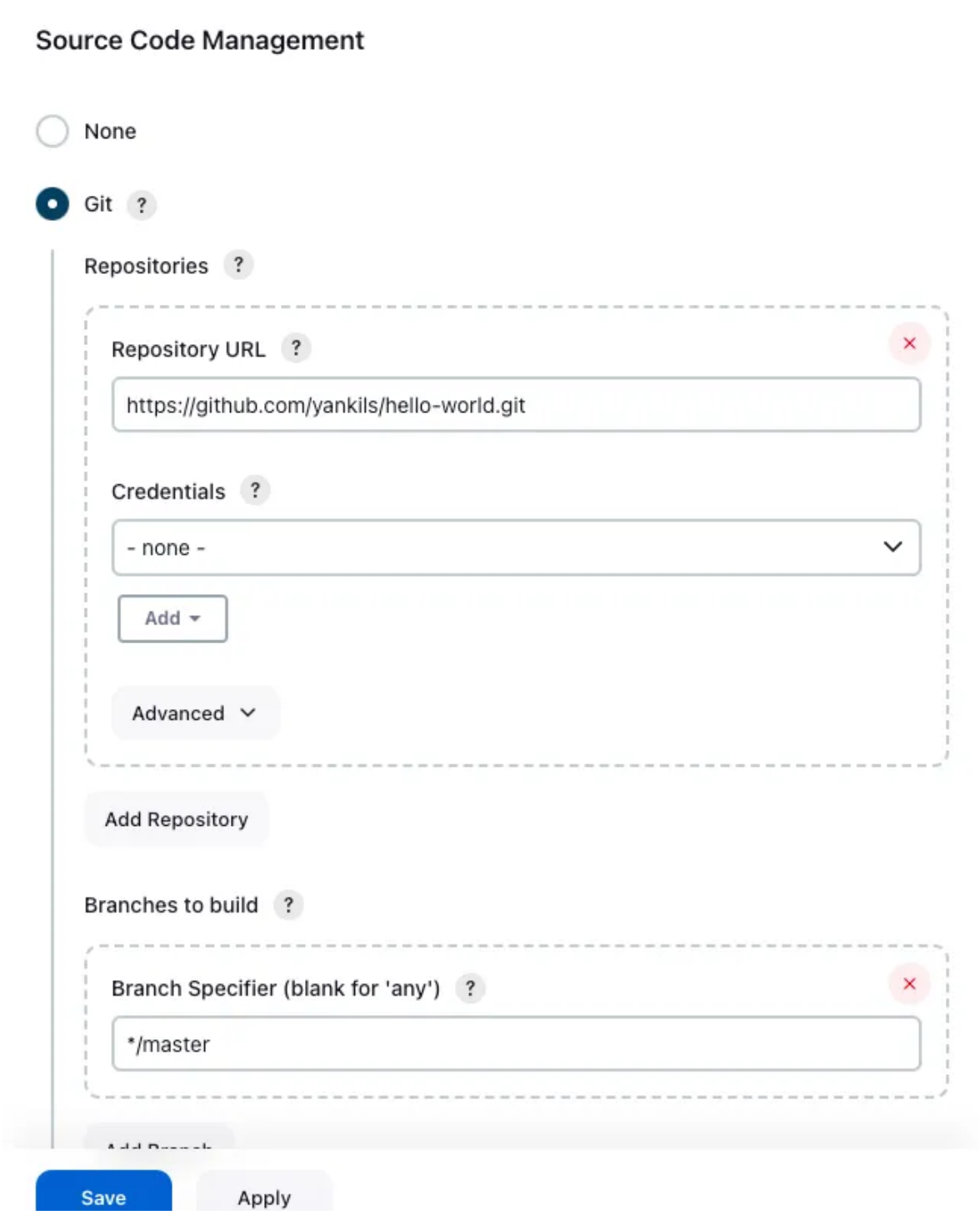


- Navigate back to the Dashboard and you will see our 1st job in queue ready for the build stage.
- Click on “Build Now”
- After a few seconds you will see the the build output under the “build history”
- The console ouput should display a green checkmark indicating that the build was SUCCESSFUL.
- You can see that the echo “Hello World” and uptime commands were properly executed!


Jenkins
Search (⌘+K)
Dashboard > HelloWorldJob >
Status
Changes
Workspace
Build Now
Configure
Delete Project
Rename
Build History trend
No builds
Project HelloWorldJob
HelloWorldJob
Edit description
Disable Project
Permalinks

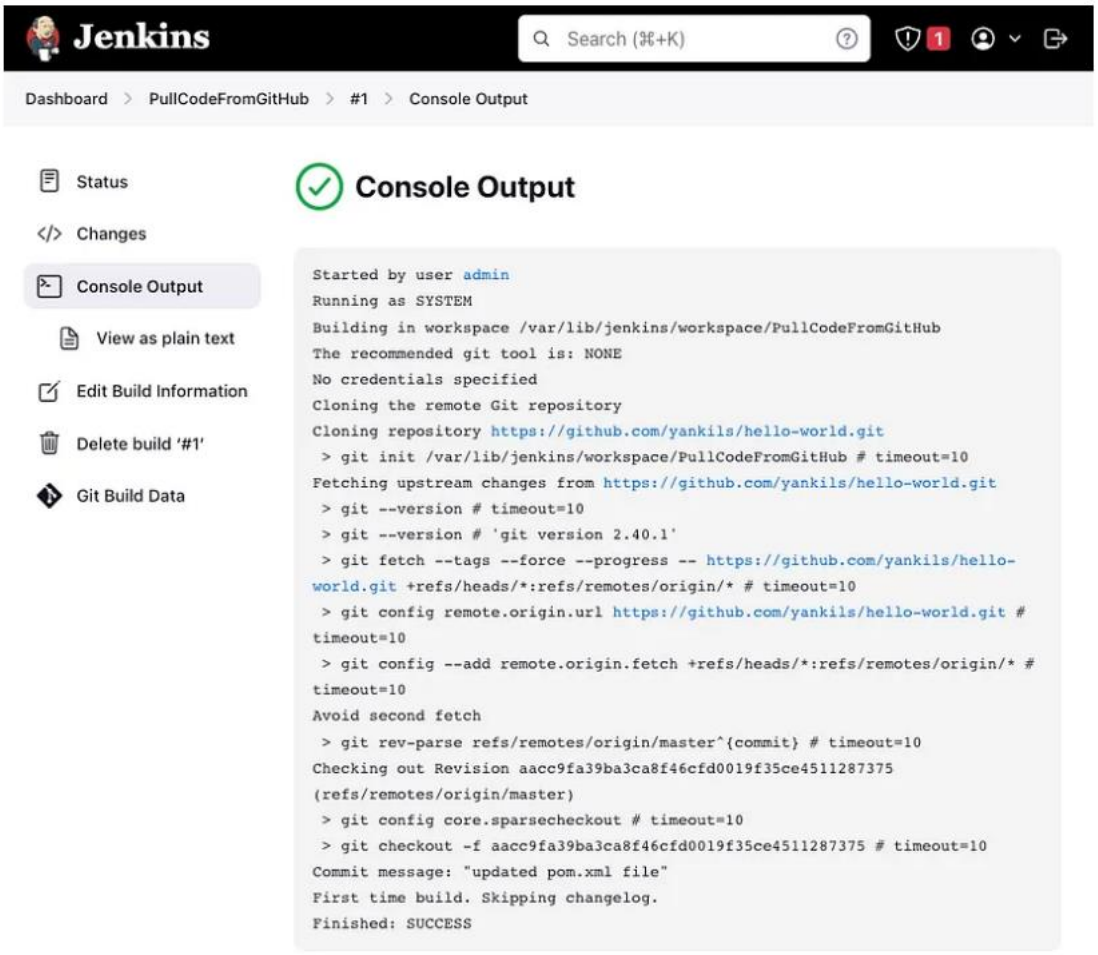

Jenkins
Search (⌘+K)
Dashboard > HelloWorldJob > #1 > Console Output
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#1'
Next Build
Console Output
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/HelloWorldJob
[HelloWorldJob] $ /bin/sh -xe /tmp/jenkins1382384225437016762 8.sh
+ echo 'Hello World'
Hello World
+ uptime
16:45:33 up 37 min, 1 user, load average: 0.03, 0.03, 0.06
Finished: SUCCESS

Step-6: Integrating git



Under Dashboard, you will see a new job for Git.

Click on “Build Now” if the build output shows a green checkmark indicator, that means our build was a SUCCESS!

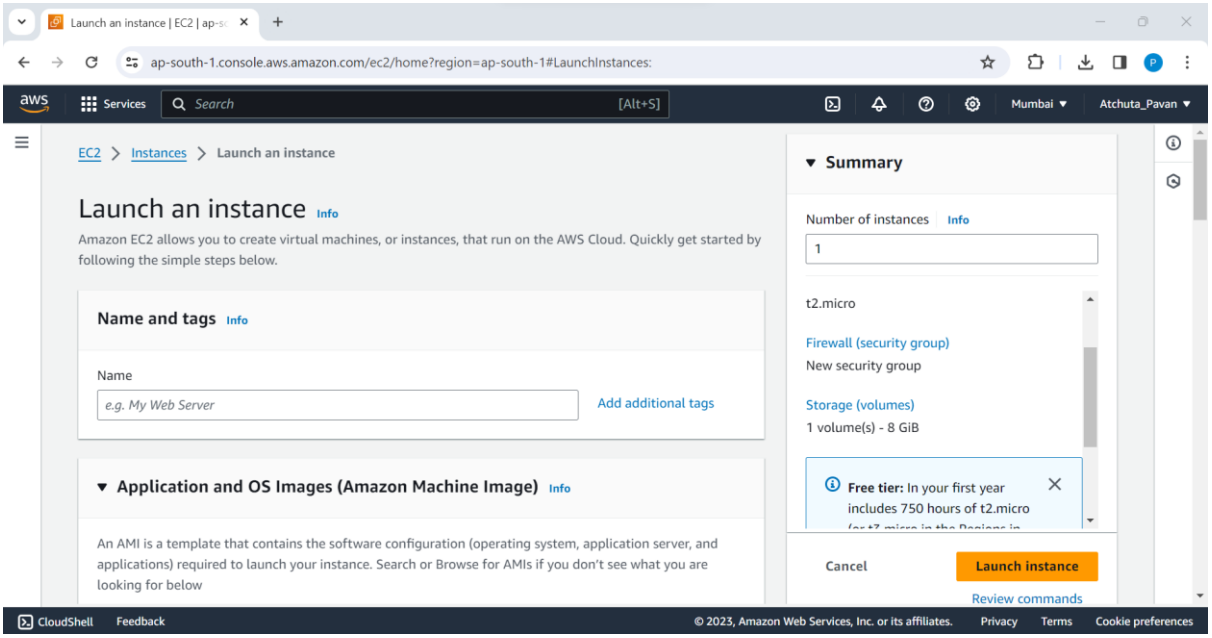

Jenkins
Search (⌘+K)
1
Dashboard > PullCodeFromGitHub > #1 > Console Output
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#1'
Git Build Data
Console Output
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/PullCodeFromGitHub
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/yankils/hello-world.git
> git init /var/lib/jenkins/workspace/PullCodeFromGitHub # timeout=10
Fetching upstream changes from https://github.com/yankils/hello-world.git
> git --version # timeout=10
> git --version # 'git version 2.40.1'
> git fetch --tags --force --progress -- https://github.com/yankils/hello-world.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/yankils/hello-world.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision aacc9fa39ba3ca8f46cfd0019f35ce4511287375 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f aacc9fa39ba3ca8f46cfd0019f35ce4511287375 # timeout=10
Commit message: "updated pom.xml file"
First time build. Skipping changelog.
Finished: SUCCESS

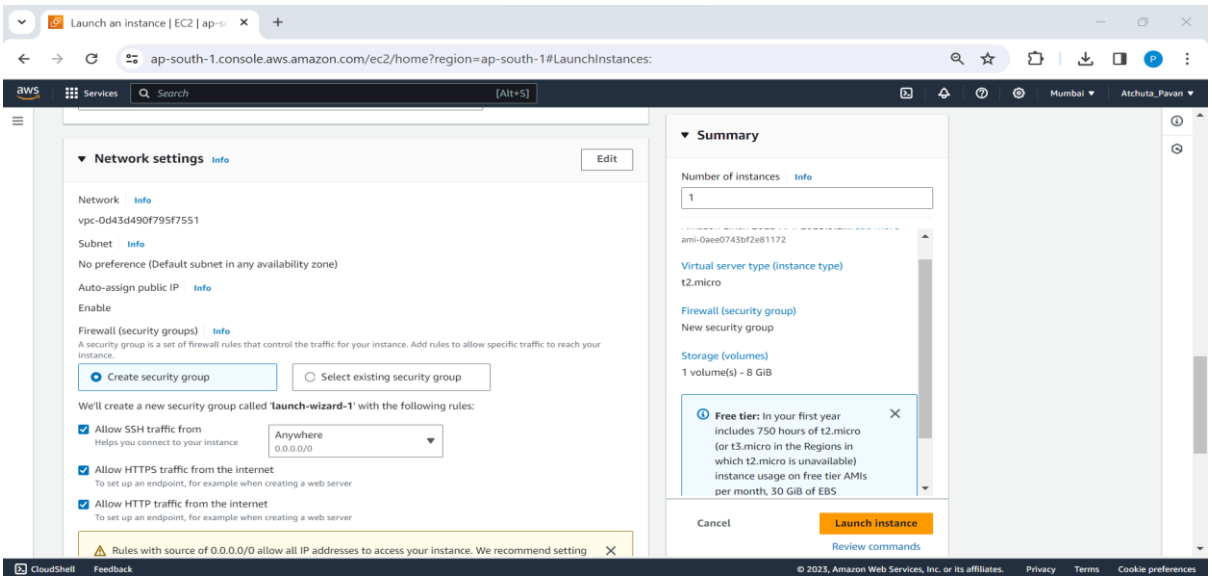
4. Aim: Demonstrate using Jenkins Service with Maven plugin in AWS and build a sample project.

Source Code:

Step-1: Creating the aws EC2 instances:

1. Log in to the AWS Management Console.
2. Navigate to the EC2 service.
3. Click on “Launch Instance” to create a new EC2 instance.
4. Select the desired instance type, storage, and other configurations.
5. Configure the security group to allow inbound traffic on port 22 for SSH access.
6. Launch the instance and download the private key file (.pem).





Step-2:SSH into EC2 Instance, Install Java and Jenkins

SSH into the EC2 instance. After securely connecting to the instance, the first thing we have to do is to install the Java application onto the instance. Jenkins runs natively runs on the Java application so it has to be properly installed in order for the Jenkins server to be operational.

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

```
[ec2-user@ip-172-31-13-239 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2023-12-17 07:10:03--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/jenkins.r 100%[=====================================>]      85  --.-KB/s    in 0s

2023-12-17 07:10:03 (5.74 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-13-239 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-13-239 ~]$ sudo yum upgrade
Jenkins-stable                                                  264 kB/s |  28 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
```

2. sudo dnf install java-17-amazon-corretto -y

```
[ec2-user@ip-172-31-13-239 ~]$ sudo dnf install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:06 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package                         Arch       Version                  Repository      Size
================================================================================
Installing:
 java-17-amazon-corretto         x86_64     1:17.0.9+8-1.amzn2023.1  amazonlinux    188 k
Installing dependencies:
 alsa-lib                        x86_64     1.2.7.2-1.amzn2023.0.2   amazonlinux    504 k
 cairo                           x86_64     1.17.6-2.amzn2023.0.1    amazonlinux    684 k
 dejavu-sans-fonts               noarch     2.37-16.amzn2023.0.2     amazonlinux    1.3 M
 dejavu-sans-mono-fonts          noarch     2.37-16.amzn2023.0.2     amazonlinux    467 k
 dejavu-serif-fonts              noarch     2.37-16.amzn2023.0.2     amazonlinux    1.0 M
 fontconfig                      x86_64     2.13.94-2.amzn2023.0.2   amazonlinux    273 k
 fonts-filesystem                noarch     1:2.0.5-12.amzn2023.0.2  amazonlinux    9.5 k
 freetype                        x86_64     2.13.0-2.amzn2023.0.1    amazonlinux    422 k
 giflib                          x86_64     5.2.1-9.amzn2023.0.1     amazonlinux     49 k
 google-noto-fonts-common        noarch     20201206-2.amzn2023.0.2  amazonlinux     15 k
 google-noto-sans-vf-fonts       noarch     20201206-2.amzn2023.0.2  amazonlinux    492 k
 graphite2                       x86_64     1.3.14-7.amzn2023.0.2    amazonlinux     97 k
 harfbuzz                        x86_64     7.0.0-2.amzn2023.0.1     amazonlinux    868 k
 java-17-amazon-corretto-headless x86_64    1:17.0.9+8-1.amzn2023.1  amazonlinux     91 M
 javapackages-filesystem         noarch     6.0.0-7.amzn2023.0.6     amazonlinux     12 k
 langpacks-core-font-en          noarch     3.0-21.amzn2023.0.4      amazonlinux     10 k
 libICE                          x86_64     1.0.10-6.amzn2023.0.2    amazonlinux     71 k
 libSM                           x86_64     1.2.3-8.amzn2023.0.2     amazonlinux     42 k
 libX11                          x86_64     1.7.2-3.amzn2023.0.4     amazonlinux    657 k
 libX11-common                   noarch     1.7.2-3.amzn2023.0.4     amazonlinux    152 k
 libXau                          x86_64     1.0.9-6.amzn2023.0.2     amazonlinux     31 k
 libXext                         x86_64     1.3.4-6.amzn2023.0.2     amazonlinux     41 k
```

3. sudo yum install jenkins -y

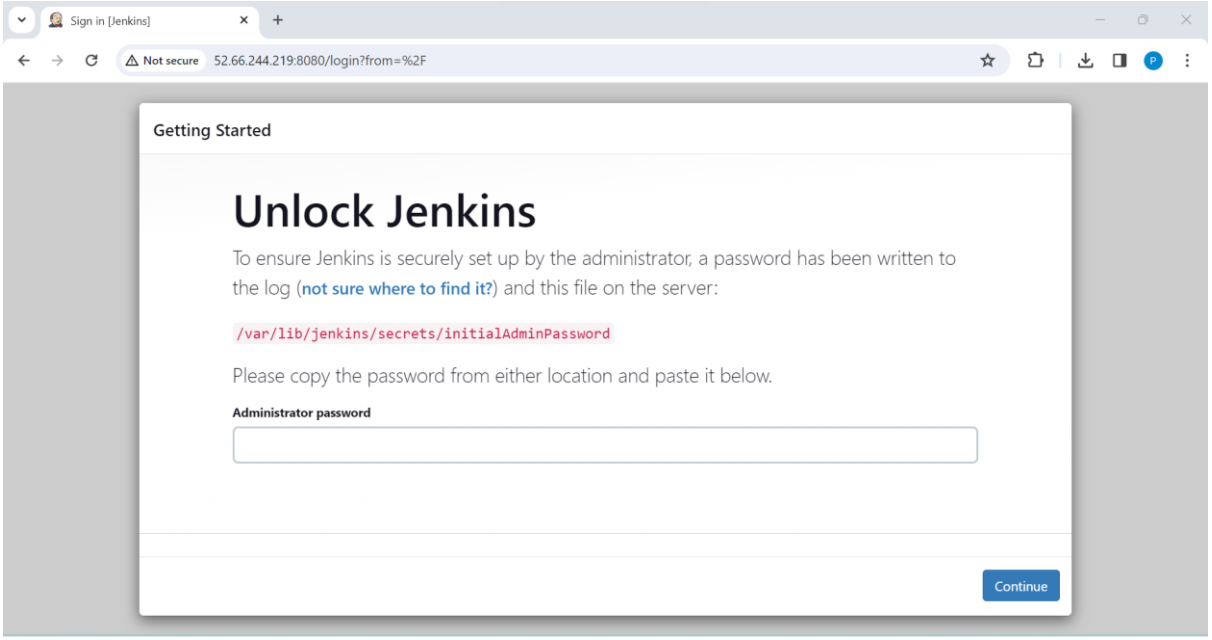
```
[ec2-user@ip-172-31-13-239 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:00:22 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package              Architecture          Version                  Repository              Size
================================================================================
Installing:
 jenkins              noarch                2.426.2-1.1              jenkins                 85 M

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

Total download size: 85 M
Installed size: 85 M
Downloading Packages:
jenkins-2.426.2-1.1.noarch.rpm                                  7.7 MB/s |  85 MB     00:11
--------------------------------------------------------------------------------
Total                                                           7.7 MB/s |  85 MB     00:11
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Installing       : jenkins-2.426.2-1.1.noarch                             1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Verifying        : jenkins-2.426.2-1.1.noarch                             1/1

Installed:
  jenkins-2.426.2-1.1.noarch
```

4. sudo systemctl enable jenkins
   sudo systemctl start jenkins

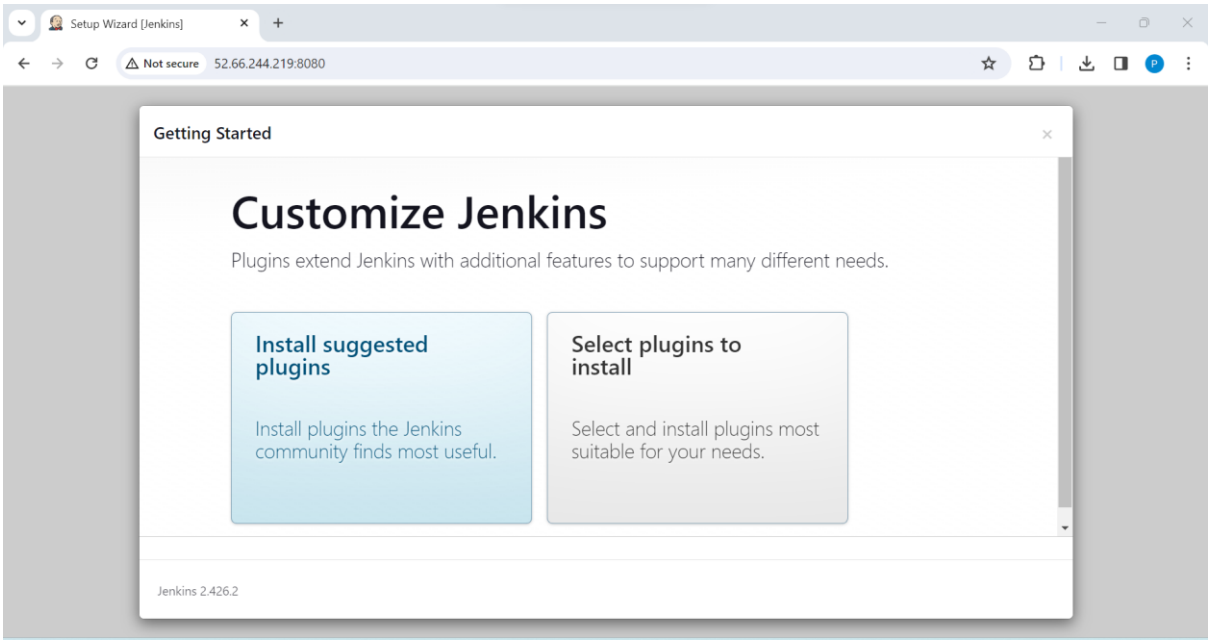
```
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jen
kins.service.
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl start jenkins
```

Step-3: Setting Up Jenkins Server

Now that the Jenkins server has been officially started on the instance, head over to the AWS Management console, grab the public IP address of the instance and paste that into your browser address bar followed by :8080




```
[ec2-user@ip-172-31-13-239 ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
baaef975c09e468a977274b59ed26a8e
[ec2-user@ip-172-31-13-239 ~]$
```



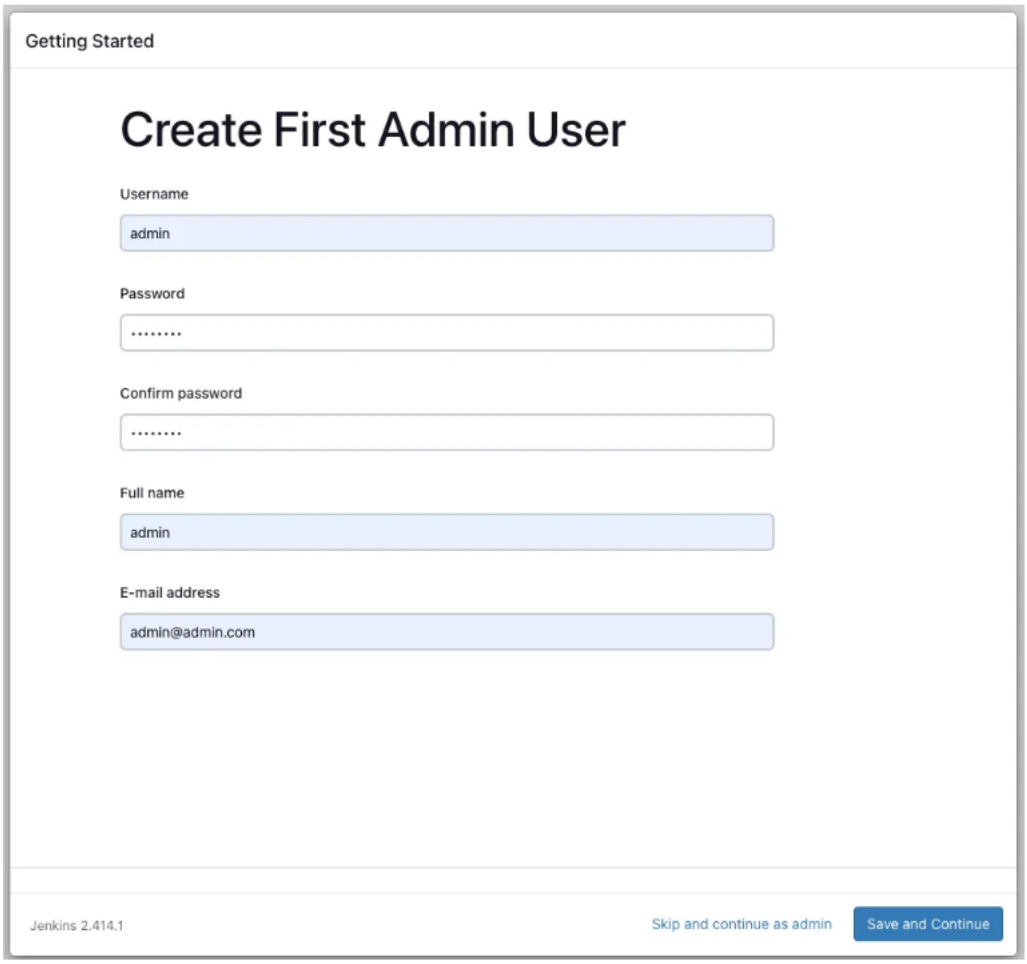
Setup an admin user. For this project we will get a dummy administrator user. In a real production you will use different credentials. Save and continue.

Username=admin

Password=Password

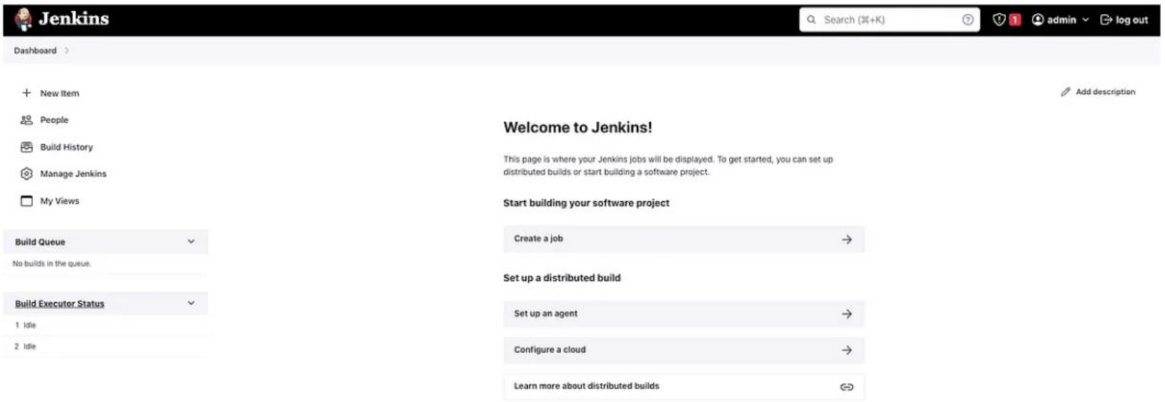
Full name= admin

Email=admin@admin.com

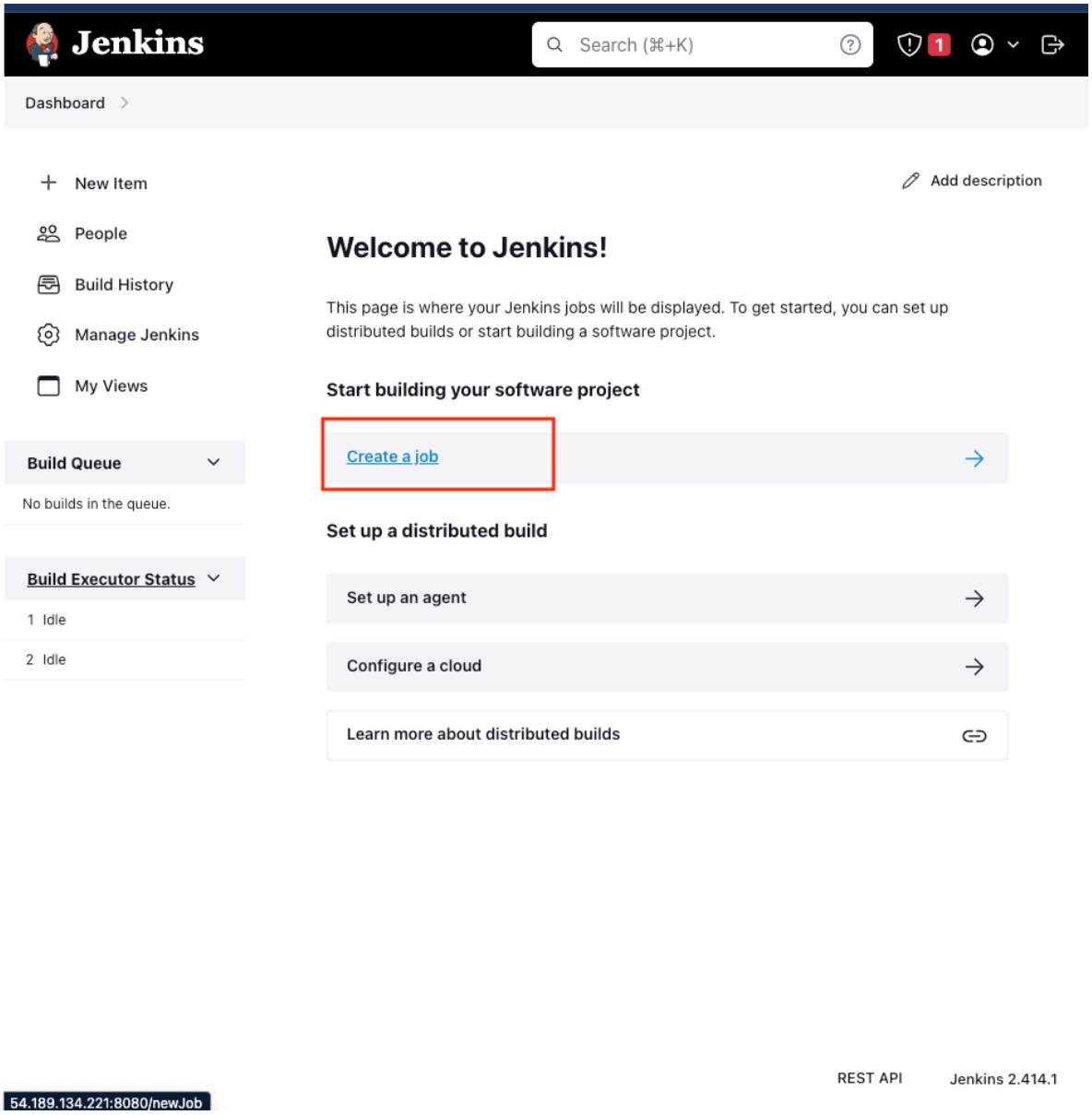


Step 4: We are Live and Jenkins Ready! Time to Build

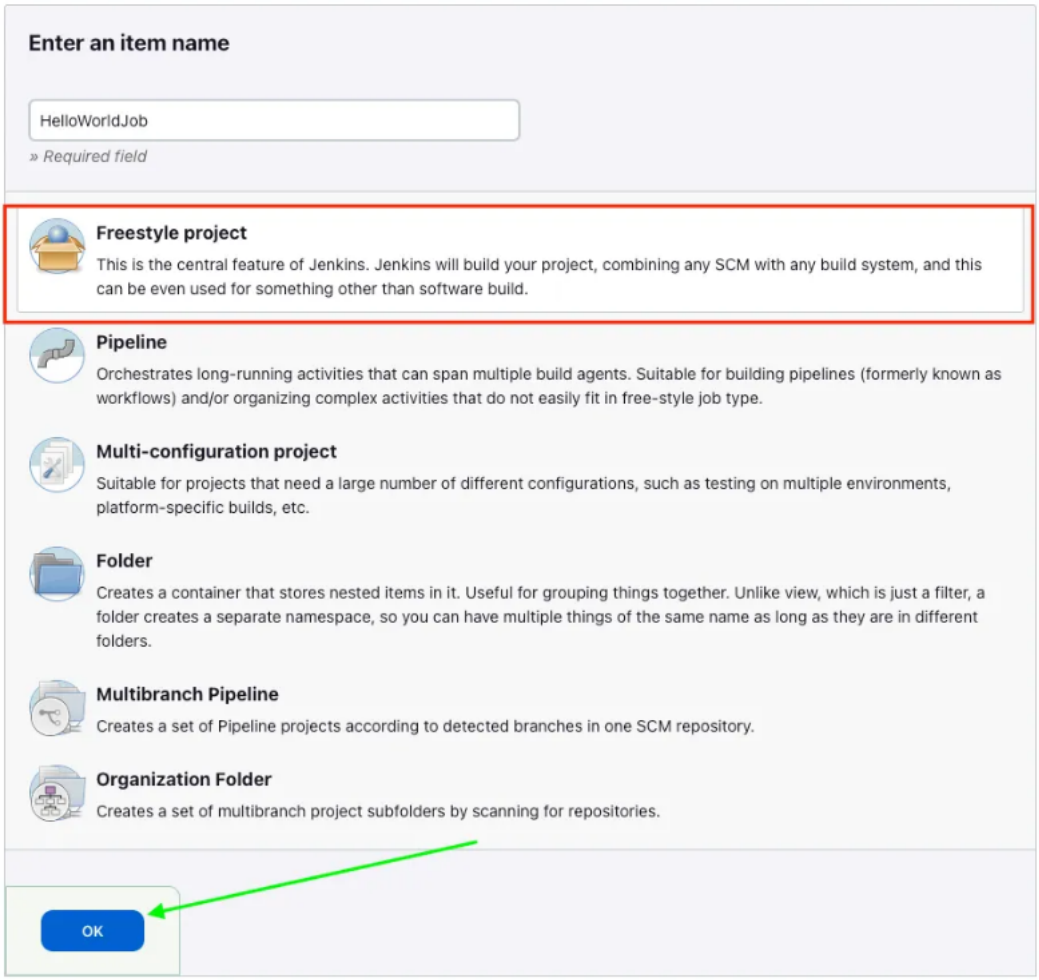
If you see the “Welcome to Jenkins” page, then you’ve successfully setup Jenkins and are ready to go to the next stage of the project, which is setting up our first Jenkins job!



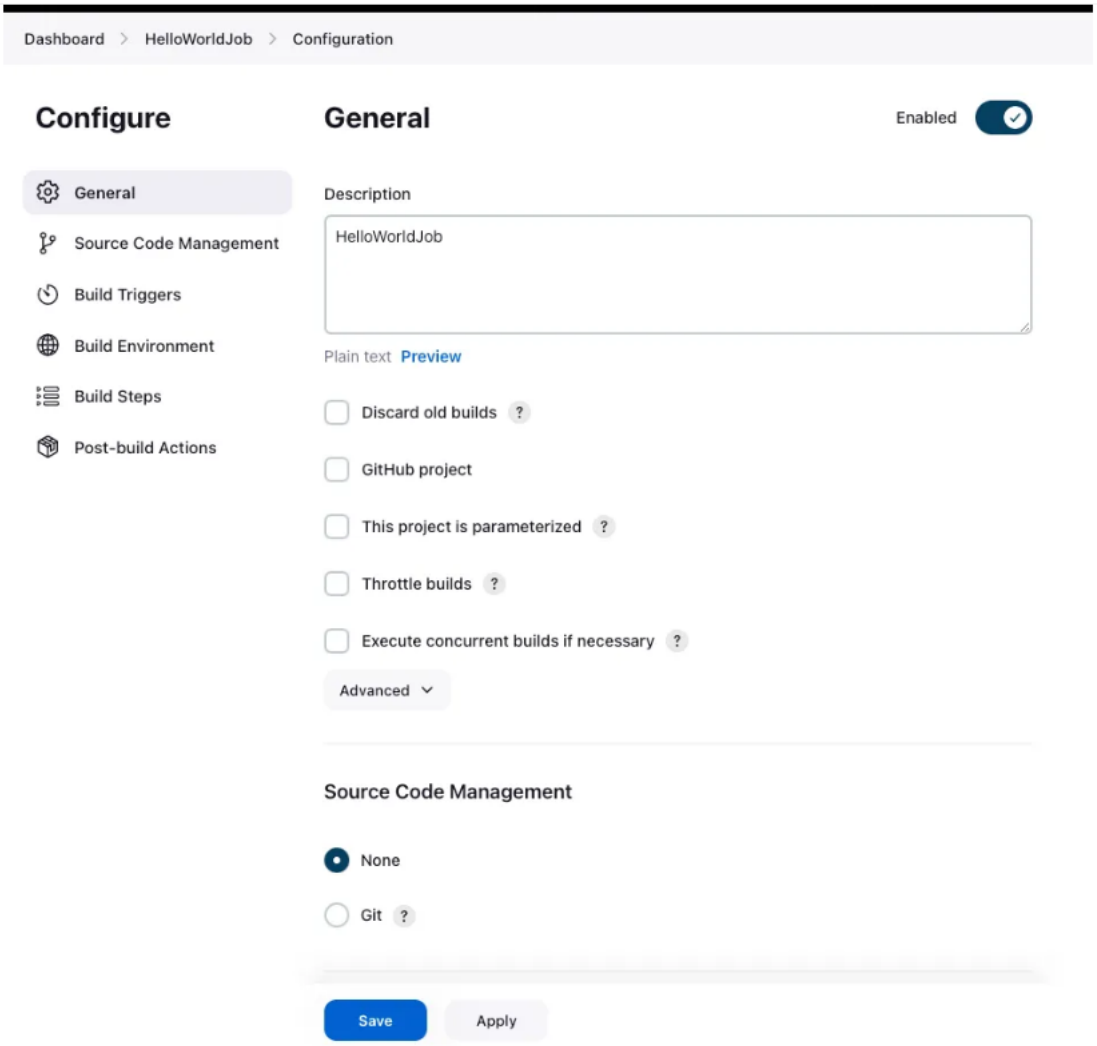
Creating our first jenkin job



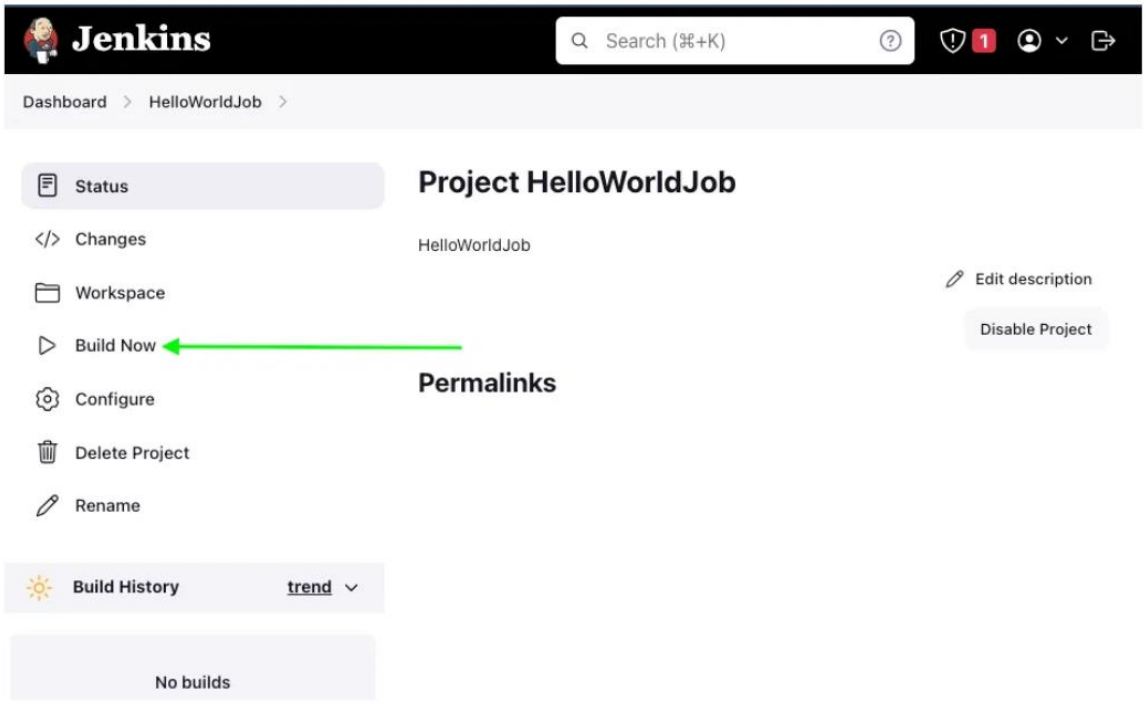
For our first job we are going to give it the name "**HelloWorldJob**" and select the option for "**Freestyle project**". Click ok to move to the next step.

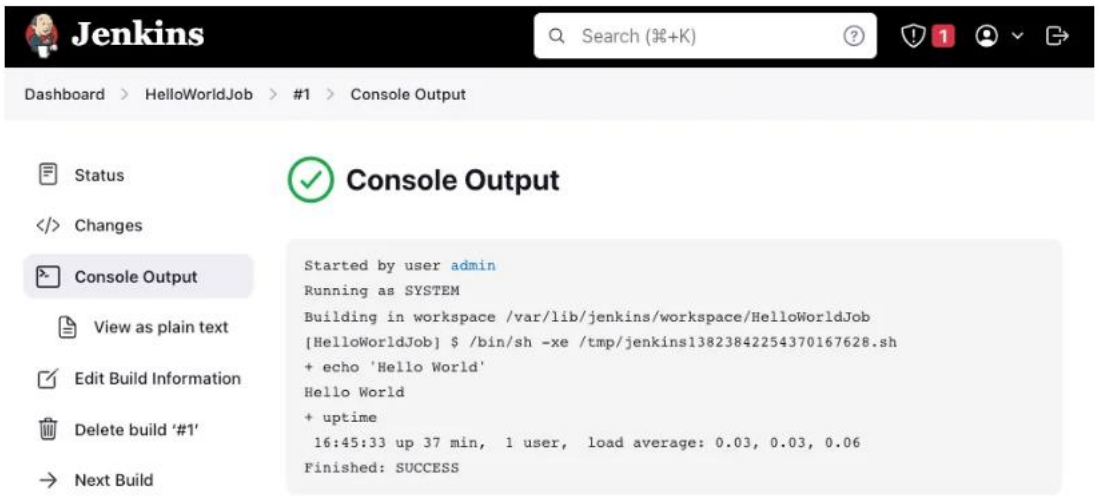


Step-5: Under the general configuration settings
In the description box, enter the text “HelloWorldJob”
Under the “Source Code Management” tab select “None” option



- Navigate back to the Dashboard and you will see our 1st job in queue ready for the build stage.
- Click on “Build Now”
- After a few seconds you will see the the build output under the “build history”
- The console ouput should display a green checkmark indicating that the build was SUCCESSFUL.
- You can see that the echo “Hello World” and uptime commands were properly executed!





Step 6: Installing and Integrating Maven to Jenkins

Maven automates the process of building and managing code in both software development as well as in DevOps. Maven manages the build, testing, and packaging of code, making it easy for DevOps teams to produce consistent, reliable, and repeatable builds.

```
[root@ip-172-31-23-228 ~]# cd /opt
[root@ip-172-31-23-228 opt]# wget https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.tar.gz
--2023-08-28 17:18:40--  https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9336368 (8.9M) [application/x-gzip]
Saving to: 'apache-maven-3.9.4-bin.tar.gz'

100%[=====================================================>] 9,336,368   --.-K/s   in 0.1s

2023-08-28 17:18:40 (81.3 MB/s) - 'apache-maven-3.9.4-bin.tar.gz' saved [9336368/9336368]
```

After installation, verify that Maven was indeed installed by list the contains of the directory: If you see the maven dependency files listed then you know it was successfully installed.

```
[root@ip-172-31-23-228 opt]# ll
total 9120
-rw-r--r-- 1 root root 9336368 Aug  3 07:56 apache-maven-3.9.4-bin.tar.gz
drwxr-xr-x 4 root root      33 Aug 22 18:26 aws
drwxr-xr-x 2 root root       6 Aug 16  2018 rh
[root@ip-172-31-23-228 opt]# tar -xvzf apache-maven-3.9.4-bin.tar.gz
apache-maven-3.9.4/README.txt
apache-maven-3.9.4/LICENSE
apache-maven-3.9.4/NOTICE
apache-maven-3.9.4/lib/
apache-maven-3.9.4/lib/aopalliance.license
apache-maven-3.9.4/lib/commons-cli.license
apache-maven-3.9.4/lib/commons-codec.license
apache-maven-3.9.4/lib/commons-lang3.license
apache-maven-3.9.4/lib/failureaccess.license
apache-maven-3.9.4/lib/guava.license
apache-maven-3.9.4/lib/guice.license
apache-maven-3.9.4/lib/httpclient.license
apache-maven-3.9.4/lib/httpcore.license
apache-maven-3.9.4/lib/jansi.license
apache-maven-3.9.4/lib/javax.annotation-api.license
apache-maven-3.9.4/lib/javax.inject.license
apache-maven-3.9.4/lib/jcl-over-slf4j.license
apache-maven-3.9.4/lib/org.eclipse.sisu.inject.license
apache-maven-3.9.4/lib/org.eclipse.sisu.plexus.license
apache-maven-3.9.4/lib/plexus-cipher.license
apache-maven-3.9.4/lib/plexus-component-annotations.license
apache-maven-3.9.4/lib/plexus-interpolation.license
apache-maven-3.9.4/lib/plexus-sec-dispatcher.license
apache-maven-3.9.4/lib/plexus-utils.license
apache-maven-3.9.4/lib/slf4j-api.license
apache-maven-3.9.4/boot/
apache-maven-3.9.4/boot/plexus-classworlds.license
apache-maven-3.9.4/lib/jansi-native/
apache-maven-3.9.4/lib/jansi-native/Windows/
apache-maven-3.9.4/lib/jansi-native/Windows/x86/
apache-maven-3.9.4/lib/jansi-native/Windows/x86_64/
apache-maven-3.9.4/lib/jansi-native/Windows/x86/jansi.dll
apache-maven-3.9.4/lib/jansi-native/Windows/x86_64/jansi.dll
apache-maven-3.9.4/bin/m2.conf
apache-maven-3.9.4/bin/mvn.cmd
apache-maven-3.9.4/bin/mvnDebug.cmd
apache-maven-3.9.4/bin/mvn
apache-maven-3.9.4/bin/mvnDebug
```

You can choose to keep the default maven file name that is highlighted in blue or change the name of the file. I'm going to change the name to just "maven" to keep things simple. To change the name use the "mv" command.

```
[root@ip-172-31-23-228 opt]# mv apache-maven-3.9.4 maven
[root@ip-172-31-23-228 opt]# ll
total 9120
-rw-r--r-- 1 root root 9336368 Aug  3 07:56 apache-maven-3.9.4-bin.tar.gz
drwxr-xr-x 4 root root      33 Aug 22 18:26 aws
drwxr-xr-x 6 root root      99 Aug 28 17:20 maven
drwxr-xr-x 2 root root       6 Aug 16  2018 rh
[root@ip-172-31-23-228 opt]# cd maven
[root@ip-172-31-23-228 maven]# ll
total 36
drwxr-xr-x 2 root root    97 Aug 28 17:20 bin
drwxr-xr-x 2 root root    76 Aug 28 17:20 boot
drwxr-xr-x 3 root root    63 Jul 26 09:37 conf
drwxr-xr-x 4 root root  4096 Aug 28 17:20 lib
-rw-r--r-- 1 root root 18945 Jul 26 09:37 LICENSE
-rw-r--r-- 1 root root  5034 Jul 26 09:37 NOTICE
-rw-r--r-- 1 root root  2533 Jul 26 09:37 README.txt
```

Execute Maven server

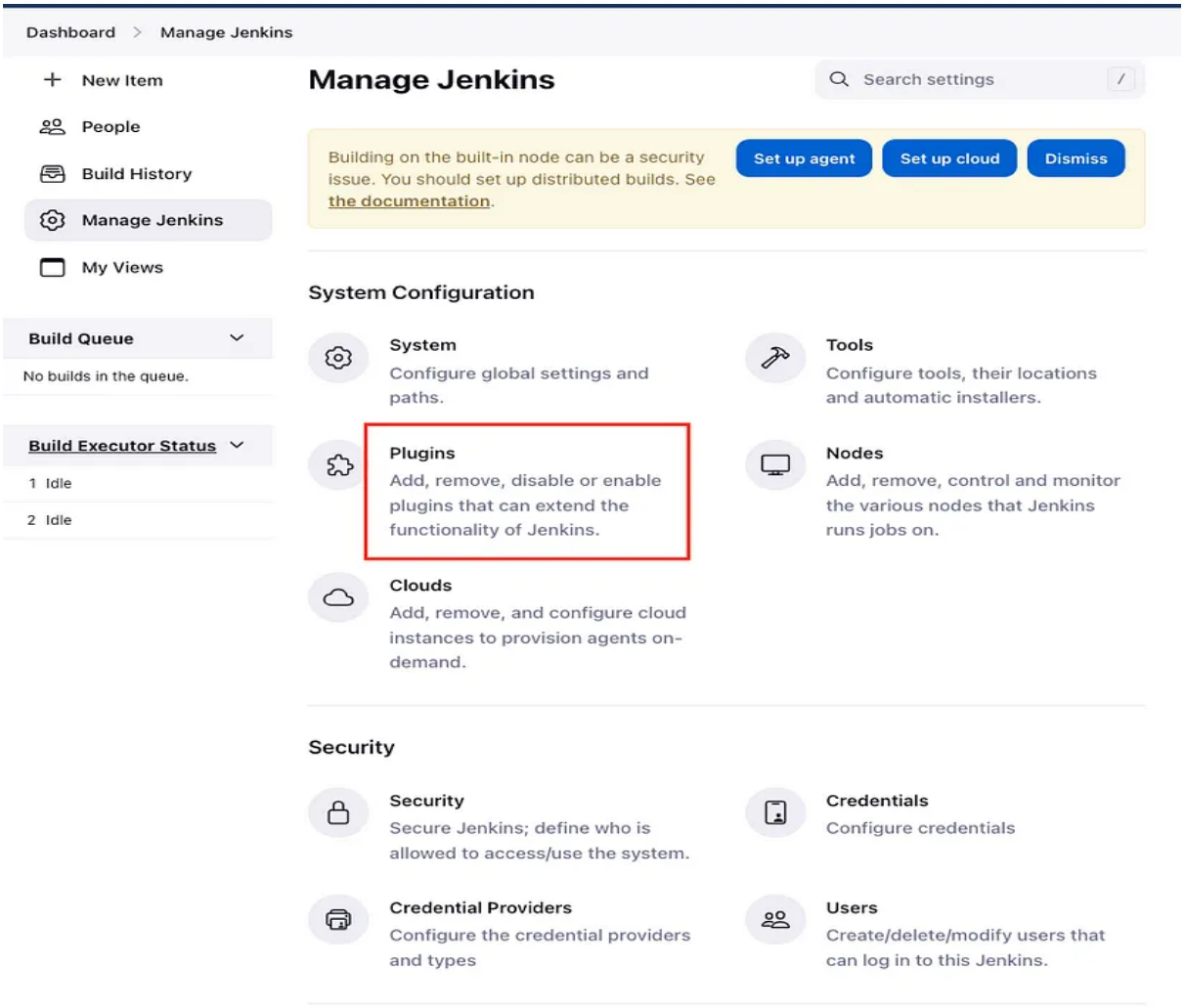
Check maven version

mvn – v

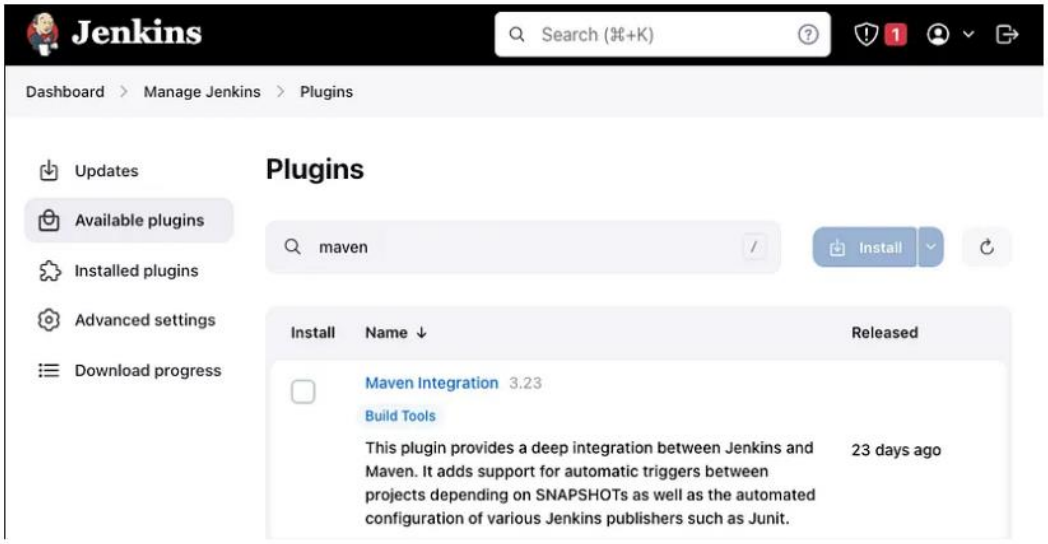
```
...c2-54-189-134-221.us-west-2.compute.amazonaws.com   ...4-189-134-221.us-west-2.compute.amazonaws.com
[root@ip-172-31-23-228 ~]# mvn -v
Apache Maven 3.9.4 (dfbb324ad4a7c8fb0bf182e6d91b0ae20e3d2dd9)
Maven home: /opt/maven
Java version: 11.0.20, vendor: Red Hat, Inc., runtime: /usr/lib/jvm/java-11-openjdk-11.0.20.0.8
-1.amzn2.0.1.x86_64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.10.186-179.751.amzn2.x86_64", arch: "amd64", family: "unix"
[root@ip-172-31-23-228 ~]#
```

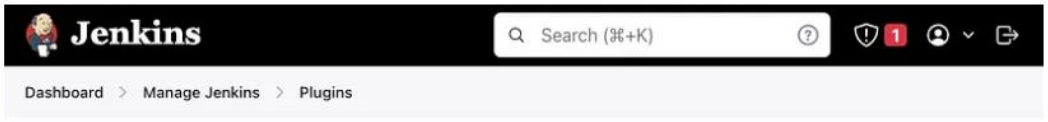
Step-7: Maven Plugin for Jenkins:

- With Maven server installed, it is time to setup the Maven plugin in the Jenkins server to complete the integration
- Navigate back to the Jenkins Dashboard, select “Manage Jenkins” option, and select “Plugins”.


Dashboard > Manage Jenkins
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Manage Jenkins
Search settings /
Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.
Set up agent
Set up cloud
Dismiss
System Configuration
System
Configure global settings and paths.
Tools
Configure tools, their locations and automatic installers.
Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
Clouds
Add, remove, and configure cloud instances to provision agents on-demand.
Security
Security
Secure Jenkins; define who is allowed to access/use the system.
Credentials
Configure credentials
Credential Providers
Configure the credential providers and types
Users
Create/delete/modify users that can log in to this Jenkins.

- Type "**Maven**" and select Maven Integration. Click on install the plugin.





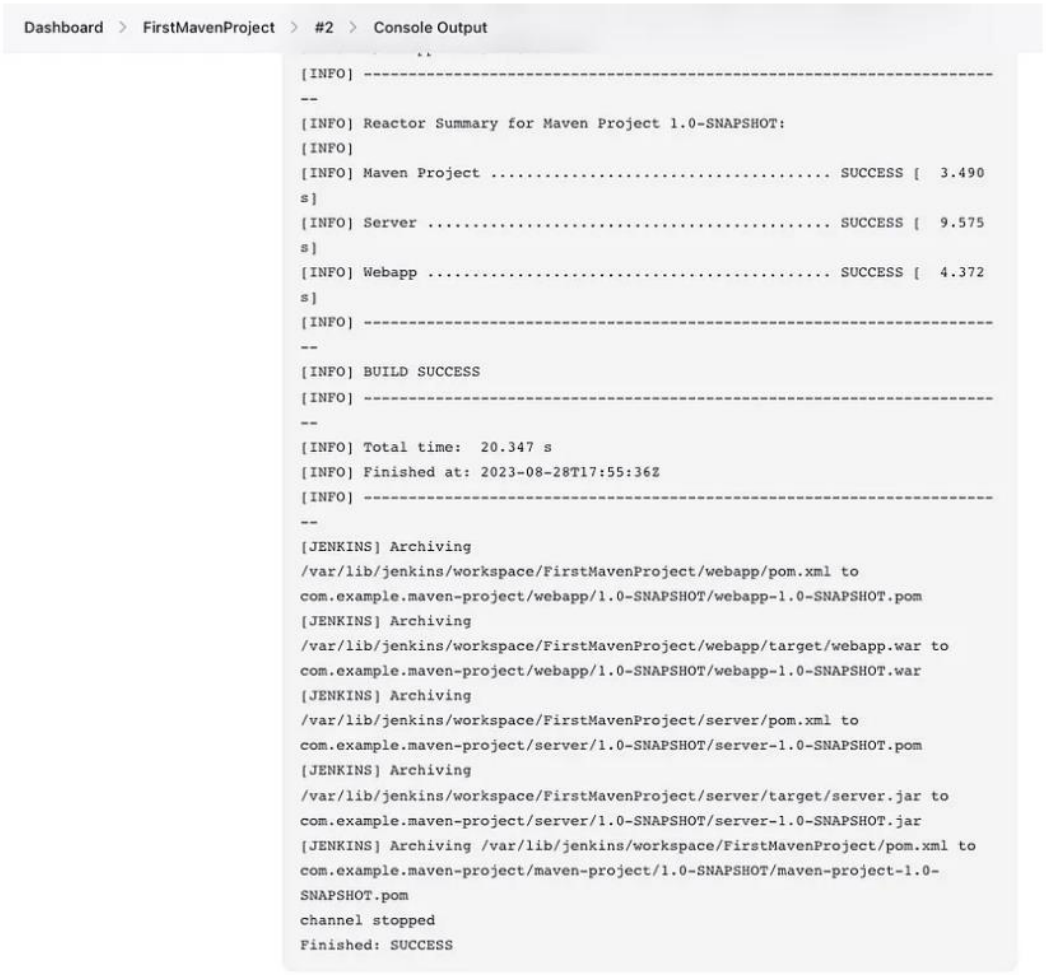
Under Mange Jenkins, select the "Global Configuration" tab. Since we successfully installed JDK and Maven, you will see the options referencing JDK and Maven.
Name-java-11
JAVA_HOME= /usr/lib/jvm/java-11-openjdk-11.0.20.0.8–1amzn2.0.1x86_64 (Java home directory)
Deselect the "install automatically" option. We already have Java installed.
Step-8: Navigate to the Jenkins dashboard, select "create item". We are going to test our CI/CD build by setting our first Maven project.

- Type "FirstMavenProject" in the name box
- Select "Maven project". Click ok.
- Enter a description for the project build.
- Select "Git" under the Source Code Management. Copy and paste the folllowing git repo URL into the box: https://github.com/GeorgeBaidooJr9/hello-world
- Under Build "Goal and Options" type "clean install". This will result in clean copy of the git repository onto Maven.
- Apply and save

Let's OFFICIALLY TEST THE BUILD!

- Select "Build" under configure. This build will take a few minutes since there is lots of data stored in the repo.

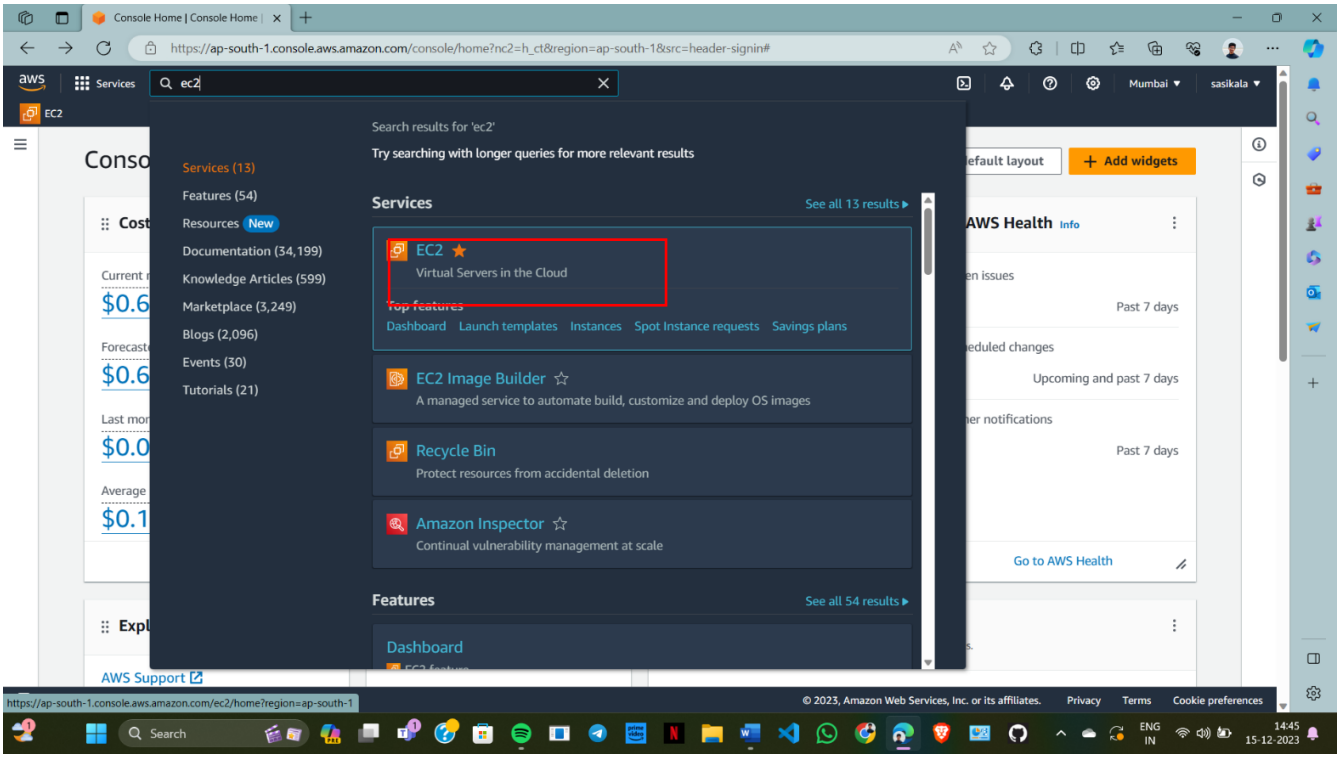
- After a few minutes you will see details of the build.



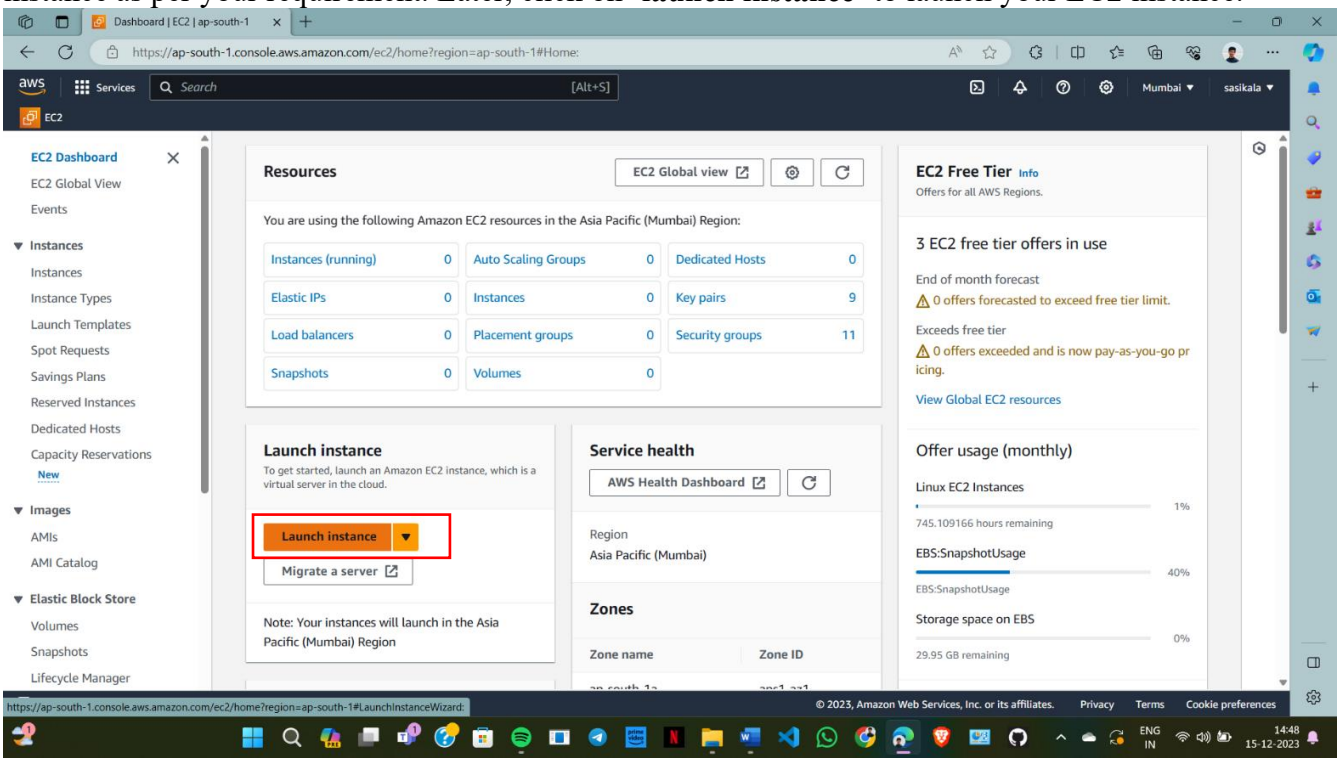
5. Deploy a Java web app with Github, Jenkins, Maven, Tomcat on AWS.

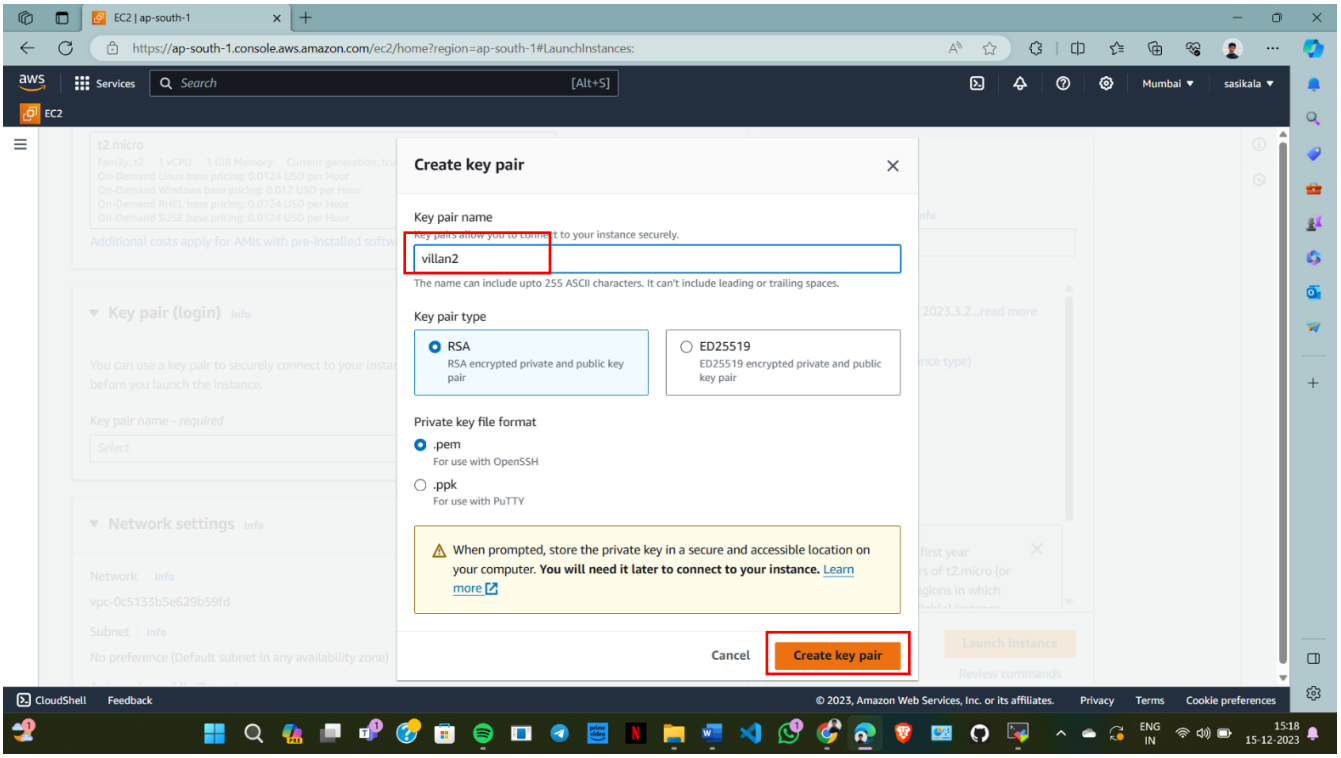
Source Code:-

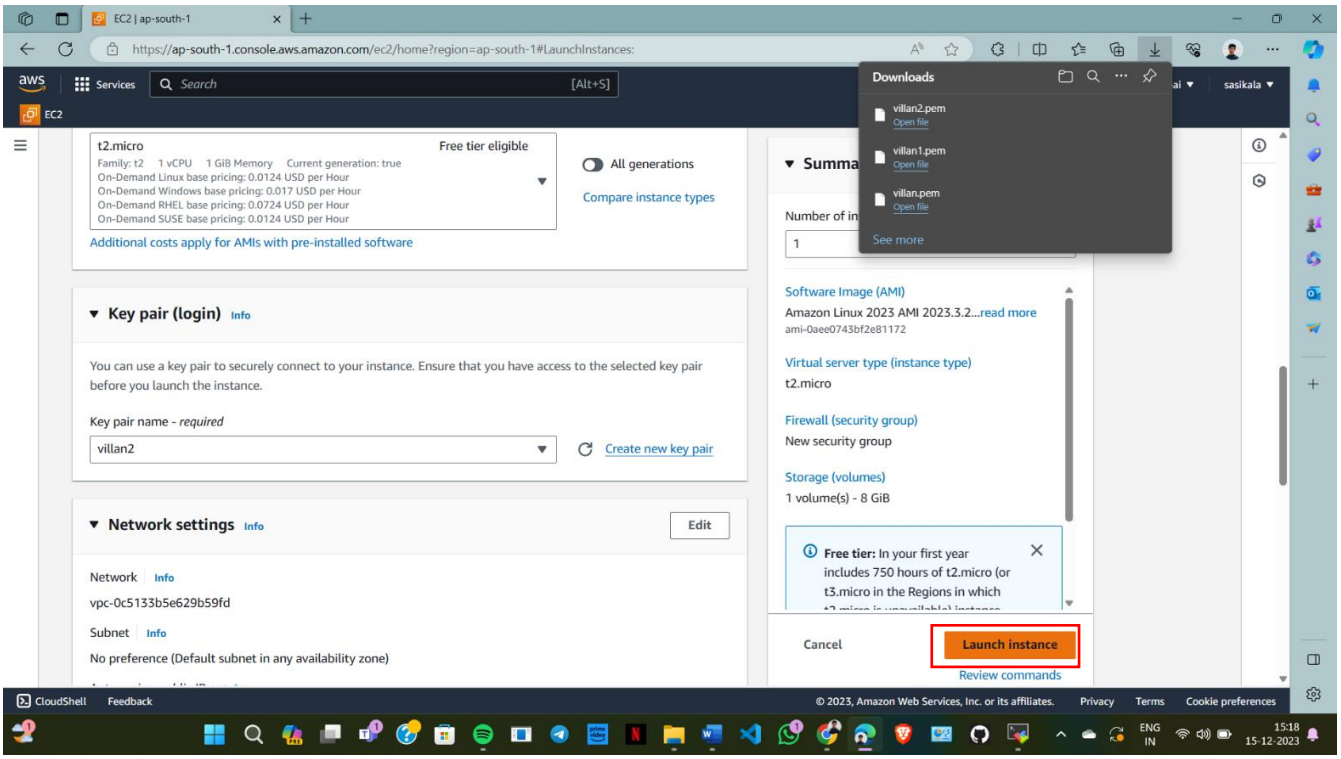
Step1: Logon to your AWS management console from your web browser.

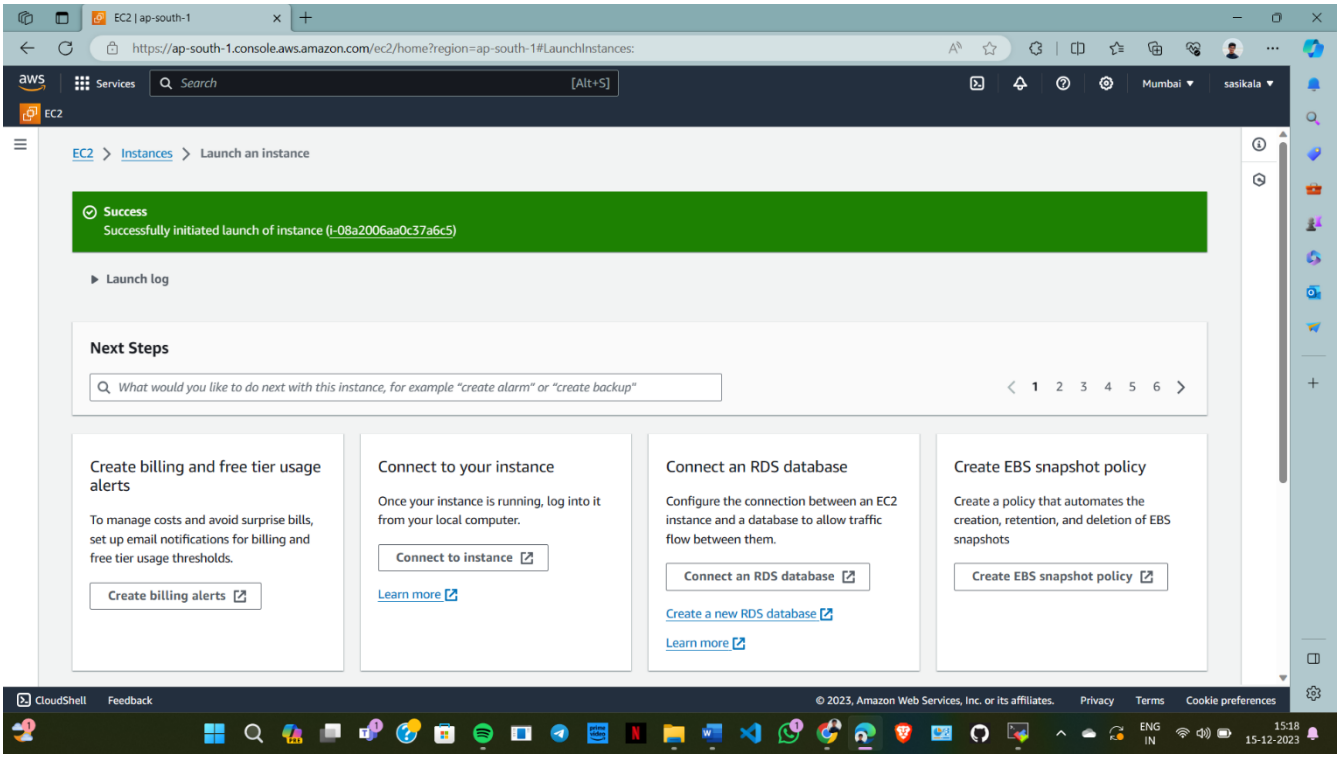


Step-2: Search for EC2 and create a Linux EC2 instance on your AWS management console by clicking on **launch instance**. Take a secured PEM key pair and configure the rest of additional configurations of instance as per your requirement. Later, click on **'launch instance'** to launch your EC2 instance.

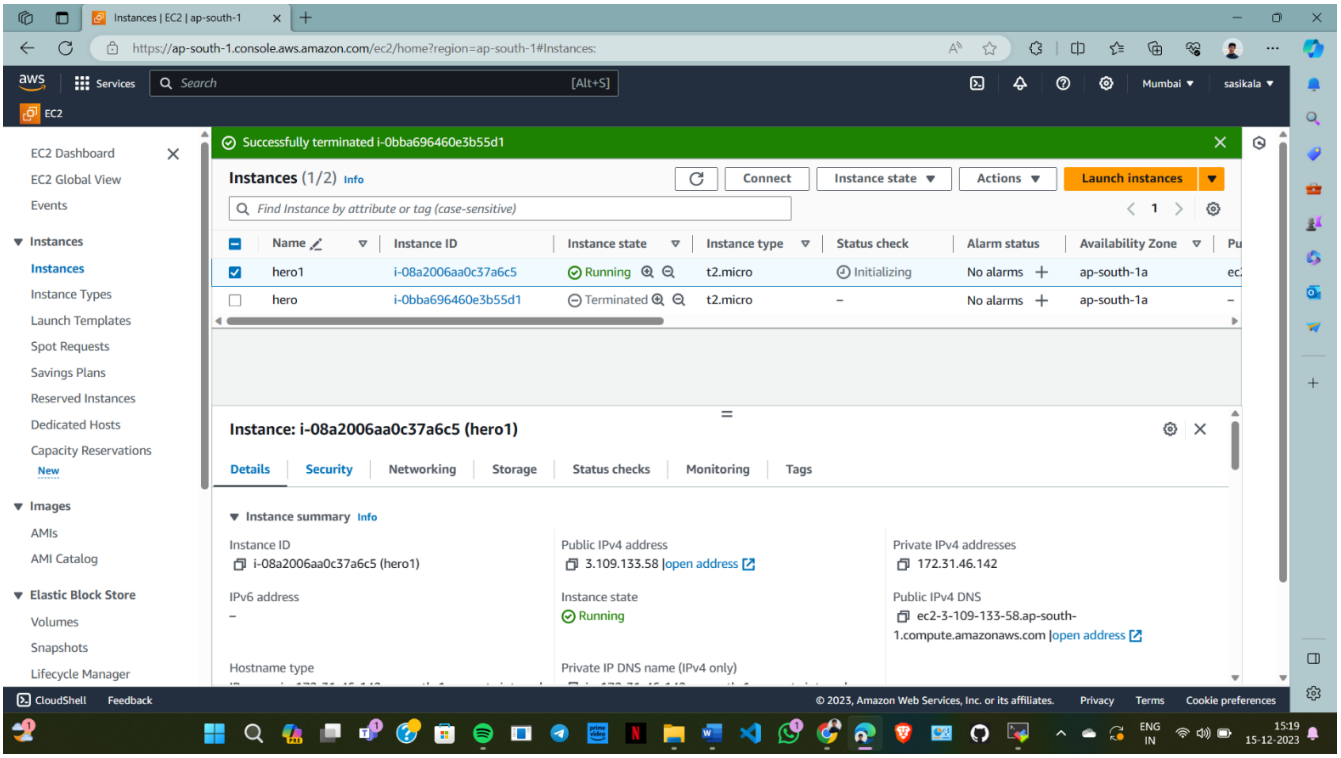


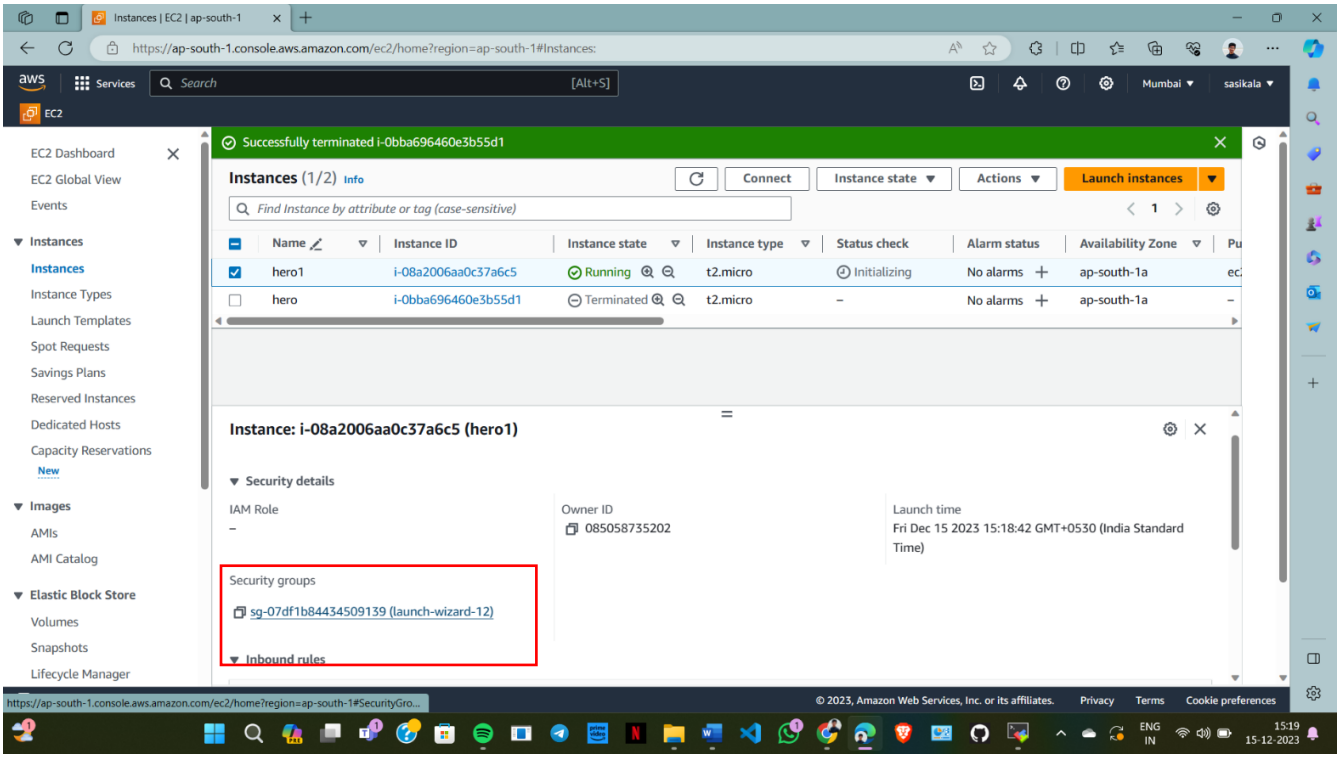

Create key pair
Key pair name
villan2
The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.
Key pair type
RSA
RSA encrypted private and public key pair
ED25519
ED25519 encrypted private and public key pair
Private key file format
.pem
For use with OpenSSH
.ppk
For use with PuTTY
When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. Learn more
Cancel
Create key pair


Downloads
villan2.pem
villan1.pem
villan.pem
Open file
See more
t2.micro
Free tier eligible
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
villan2
Create new key pair
Network settings Info
Edit
Network Info
vpc-0c5133b5e629b59fd
Subnet Info
No preference (Default subnet in any availability zone)
Software Image (AMI)
Amazon Linux 2023 AMI 2023.3.2...read more
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which
Cancel
Launch instance
Review commands

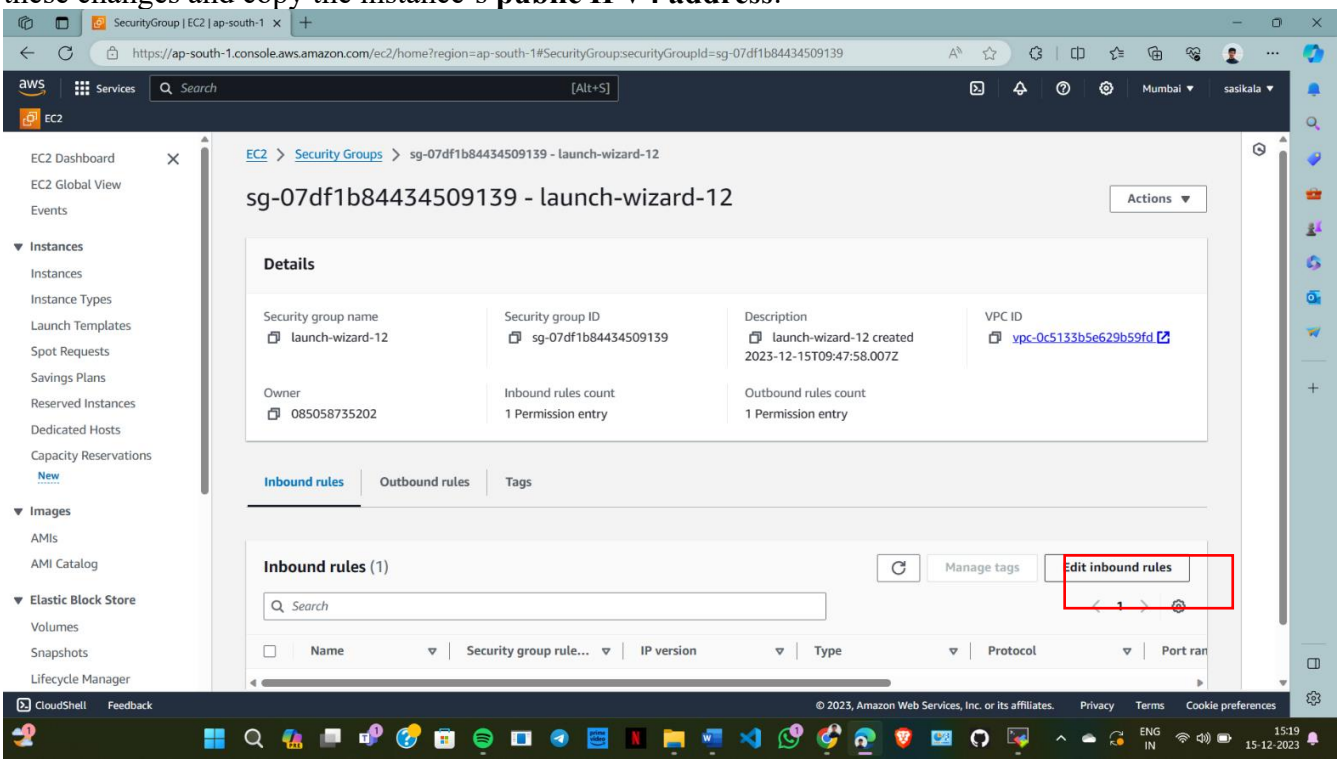


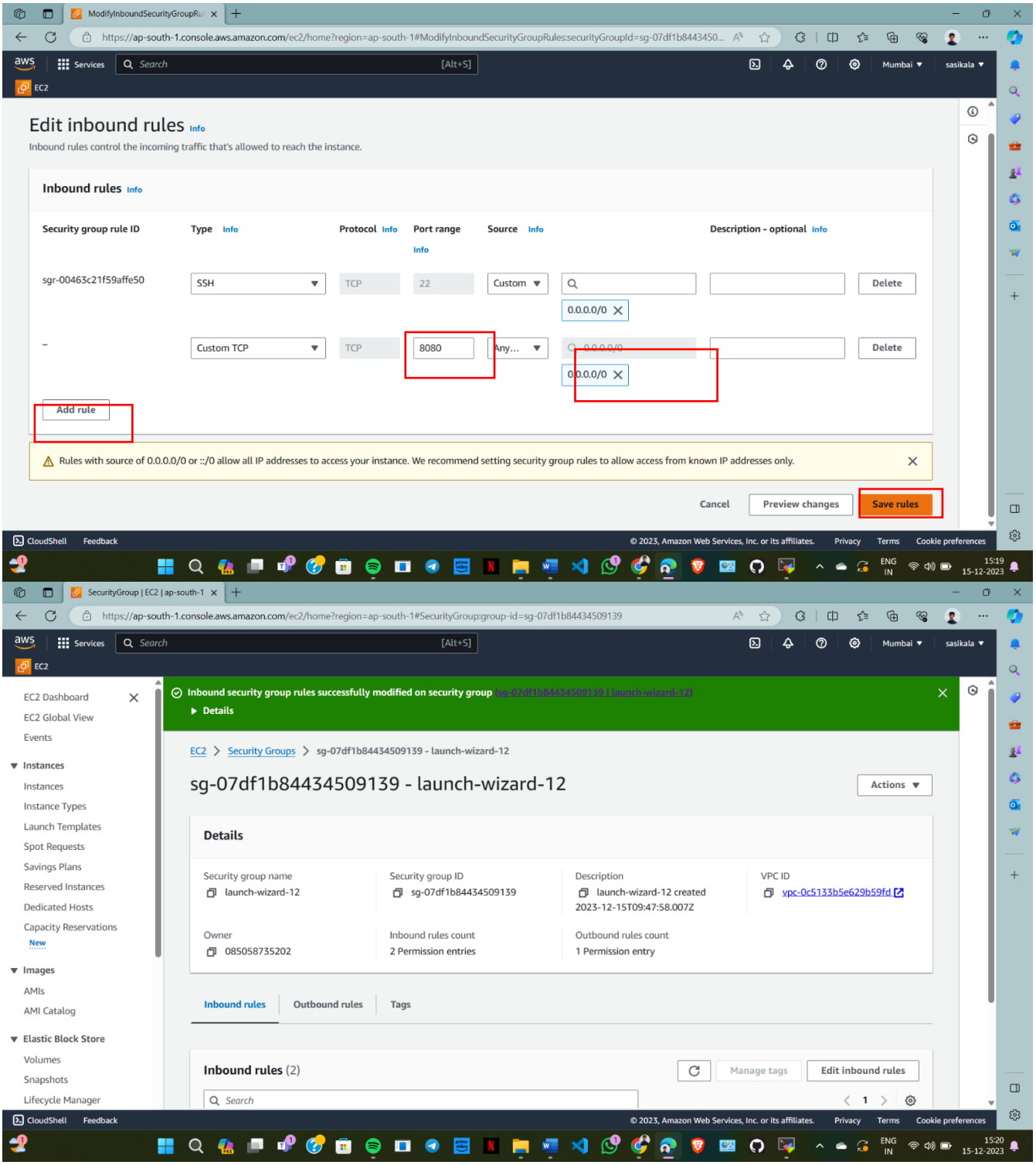
Step-3: After successfully launching the instance, edit the instance's security inbound rules by clicking on **security** and then **security groups**.

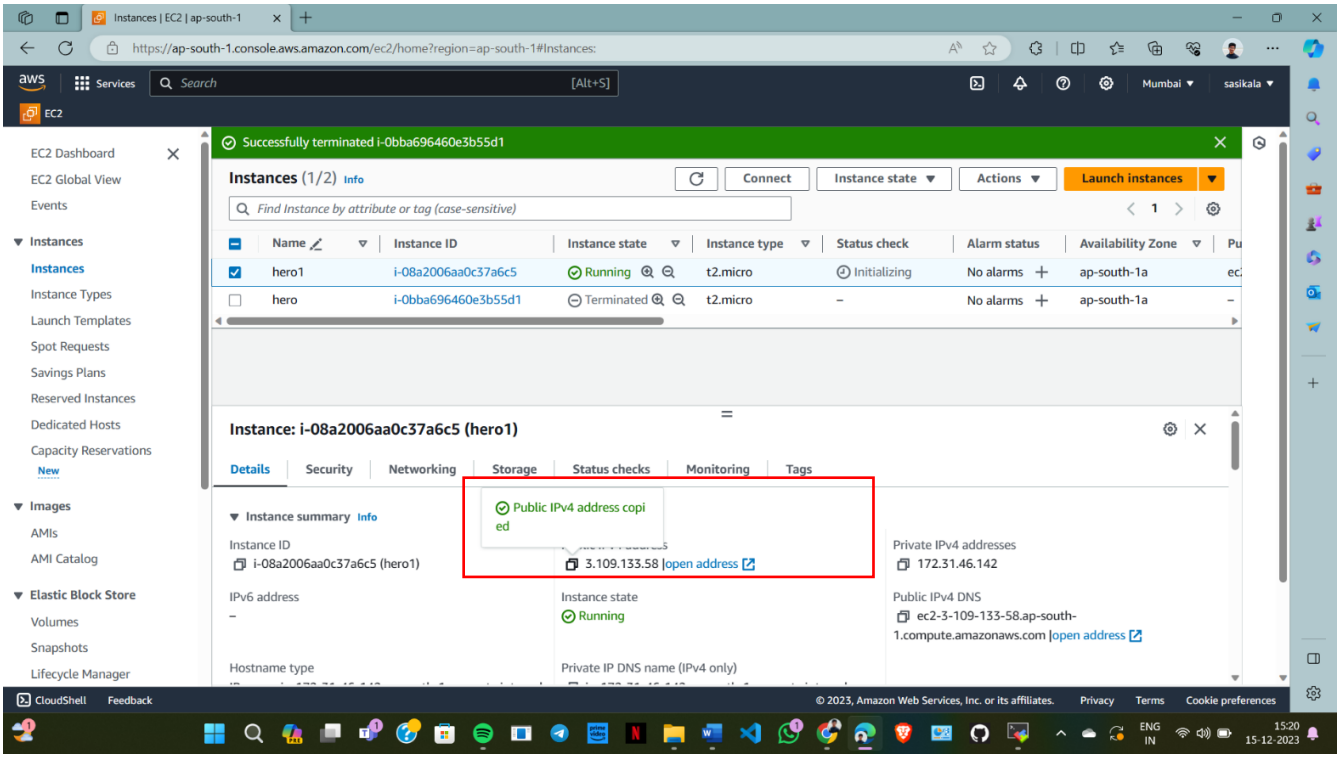




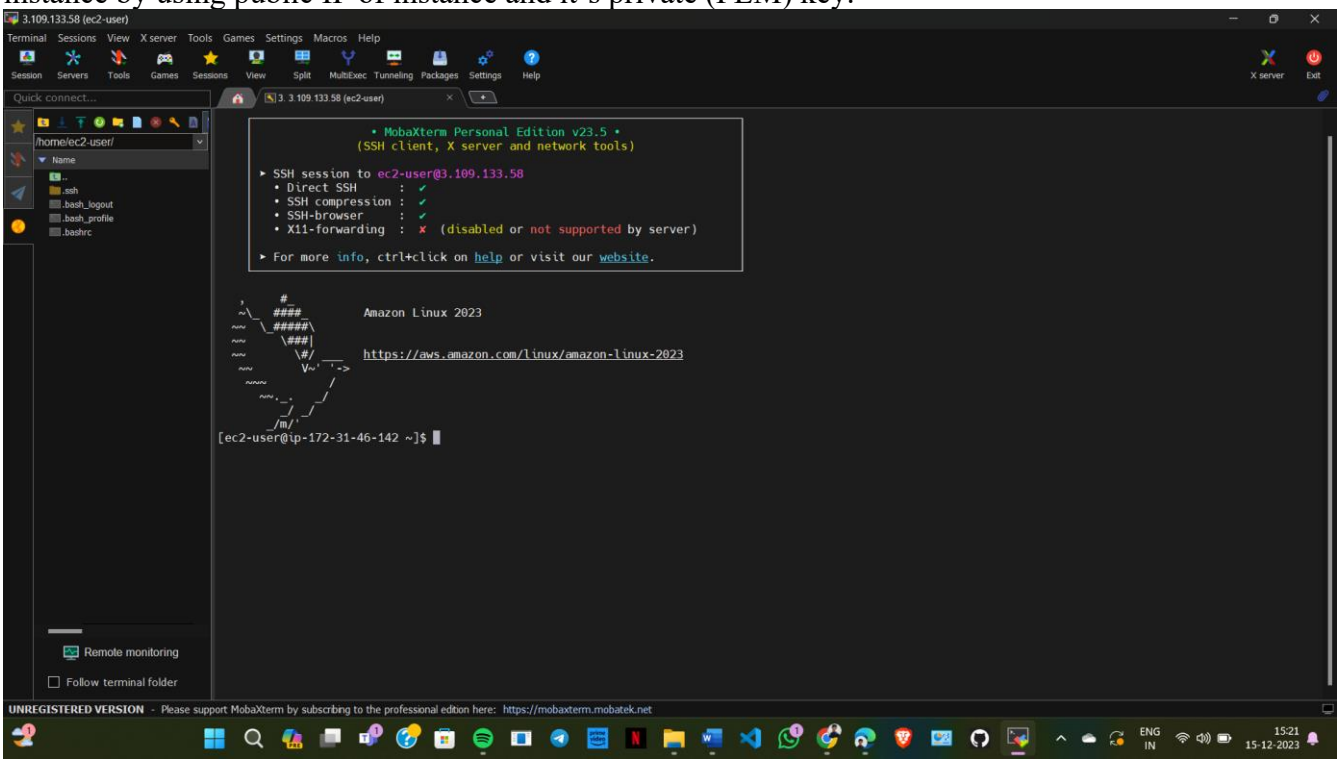
Step-4: Click on **Edit inbound rules**. Add a **custom TCP** type and select **8080** as Port Range. Save these changes and copy the instance's **public IPV4 address**.




Edit inbound rules Info
Inbound rules control the incoming traffic that's allowed to reach the instance.
Inbound rules Info
Security group rule ID
Type Info
Protocol Info
Port range Info
Source Info
Description - optional Info
sgr-00463c21f59affe50
SSH
TCP
22
Custom
0.0.0.0/0
Delete
Custom TCP
TCP
8080
Any...
0.0.0.0/0
Delete
Add rule
Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Cancel
Preview changes
Save rules
Inbound security group rules successfully modified on security group
Details
EC2 > Security Groups > sg-07df1b84434509139 - launch-wizard-12
sg-07df1b84434509139 - launch-wizard-12
Actions
Details
Security group name
launch-wizard-12
Security group ID
sg-07df1b84434509139
Description
launch-wizard-12 created 2023-12-15T09:47:58.007Z
VPC ID
vpc-0c5133b5e629b59fd
Owner
085058735202
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager

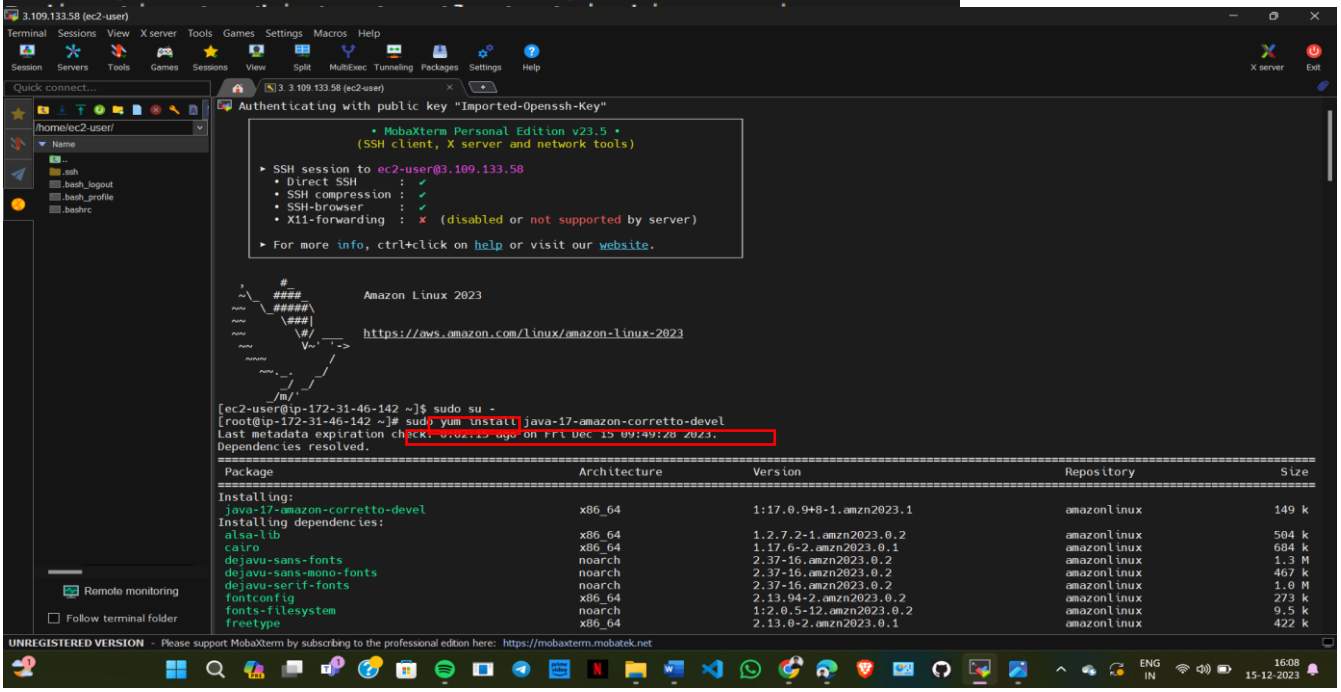


Step-5: Download and open MobaXTerm. Select **SSH terminal** and connect the SSH terminal to your instance by using public IP of instance and it's private (PEM) key.



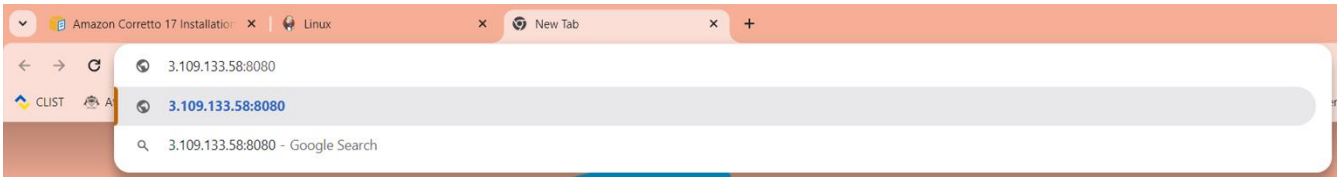
Step-6: Run the **$ sudo su -** command to enter "super user" mode. Now the run the following commands to install and start Jenkins server:

```
[root@ip-172-31-46-142 ~]# sudo wget -0 /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
[root@ip-172-31-46-142 ~]# sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[root@ip-172-31-46-142 ~]# yum install jenkins
[root@ip-172-31-38-223 ~]# yum install git
[root@ip-172-31-46-142 ~]# service jenkins start
```



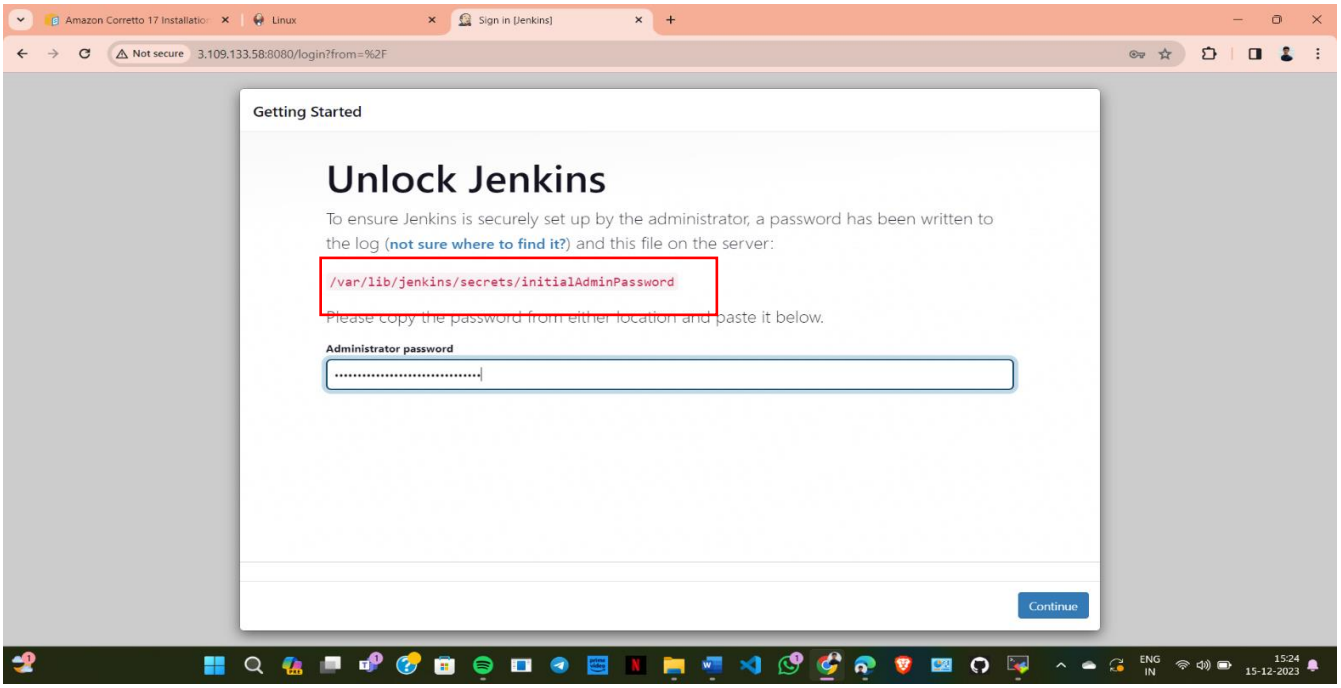
Step-7: To open Jenkins, paste the EC2 linux instance's public IP along with your port range(8080) on your browser tab.

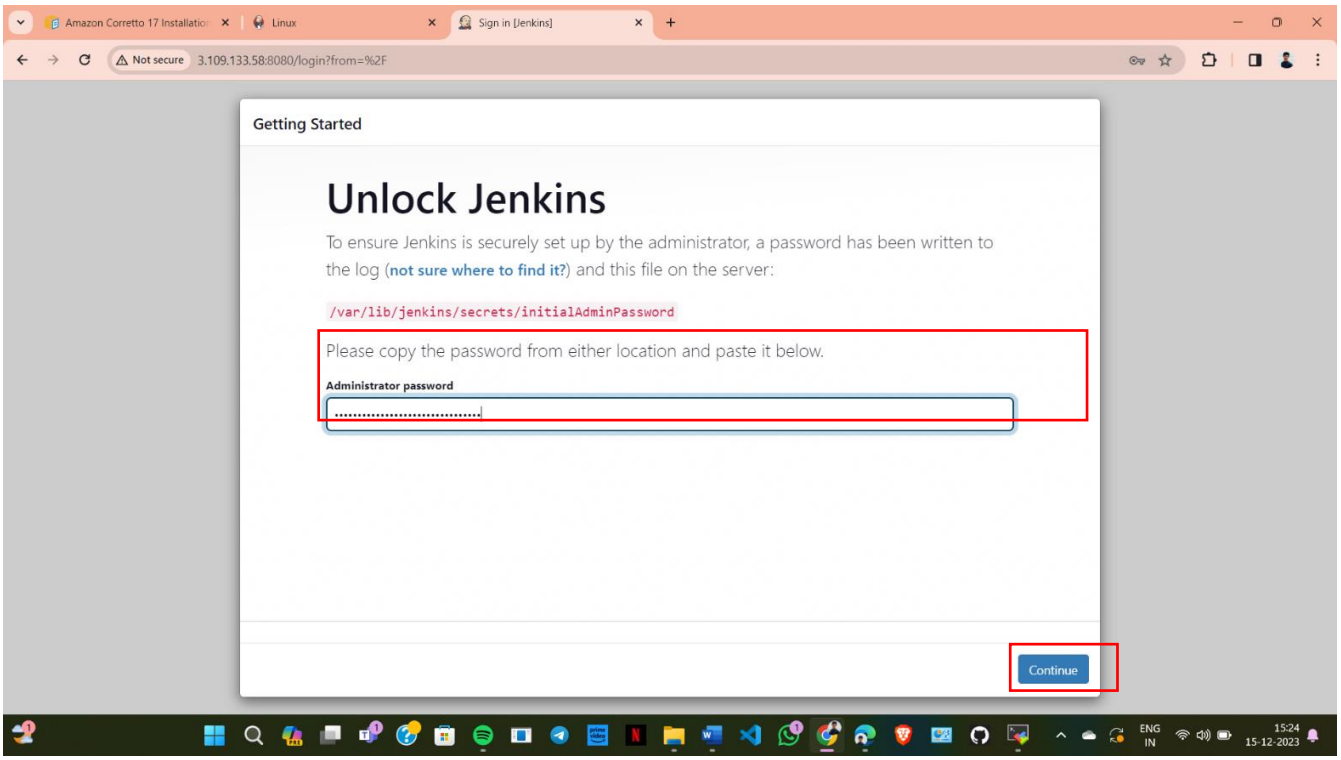
3.109.133.58:8080

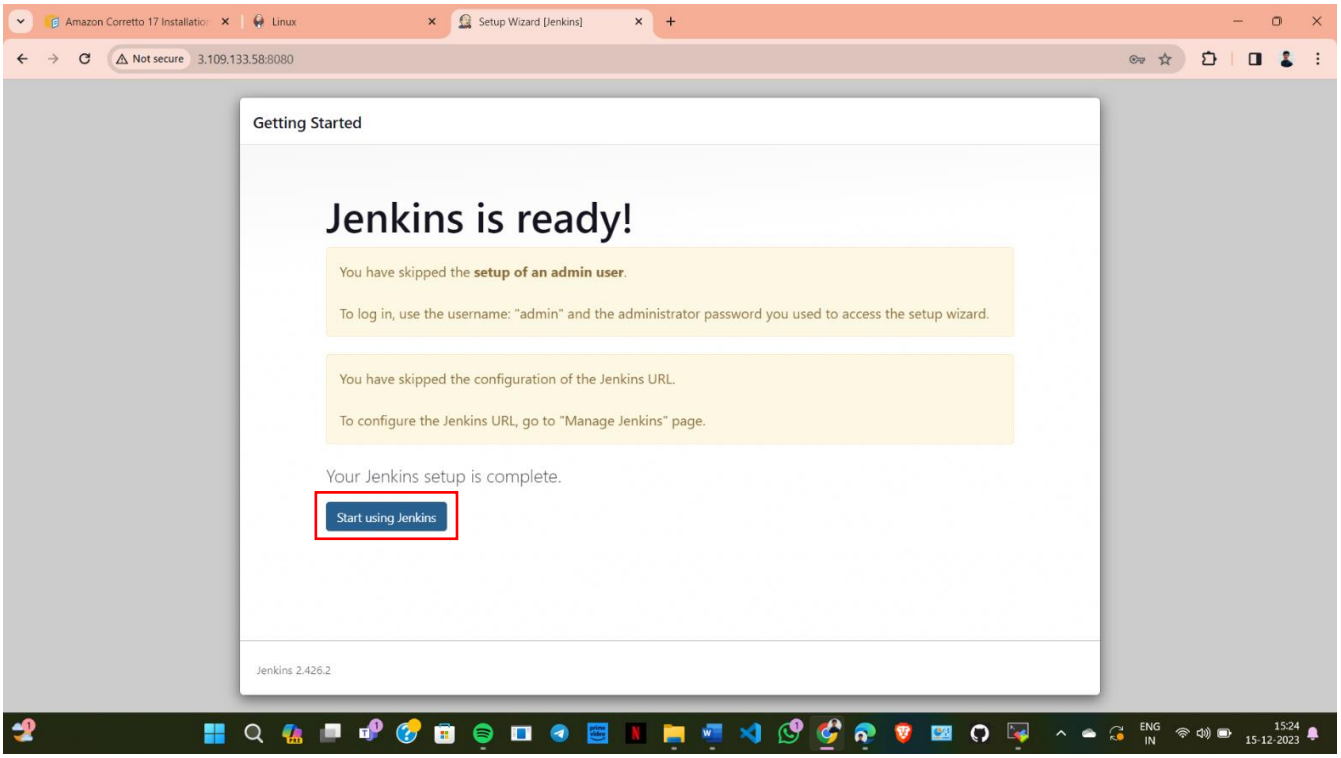


Step-8: To get the password for opening Jenkins, copy the displaying url and run the following command on your SSH terminal

```
[root@ip-172-31-46-142 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
24b19efd4b2e43a69b59ca4b2d97d619
```

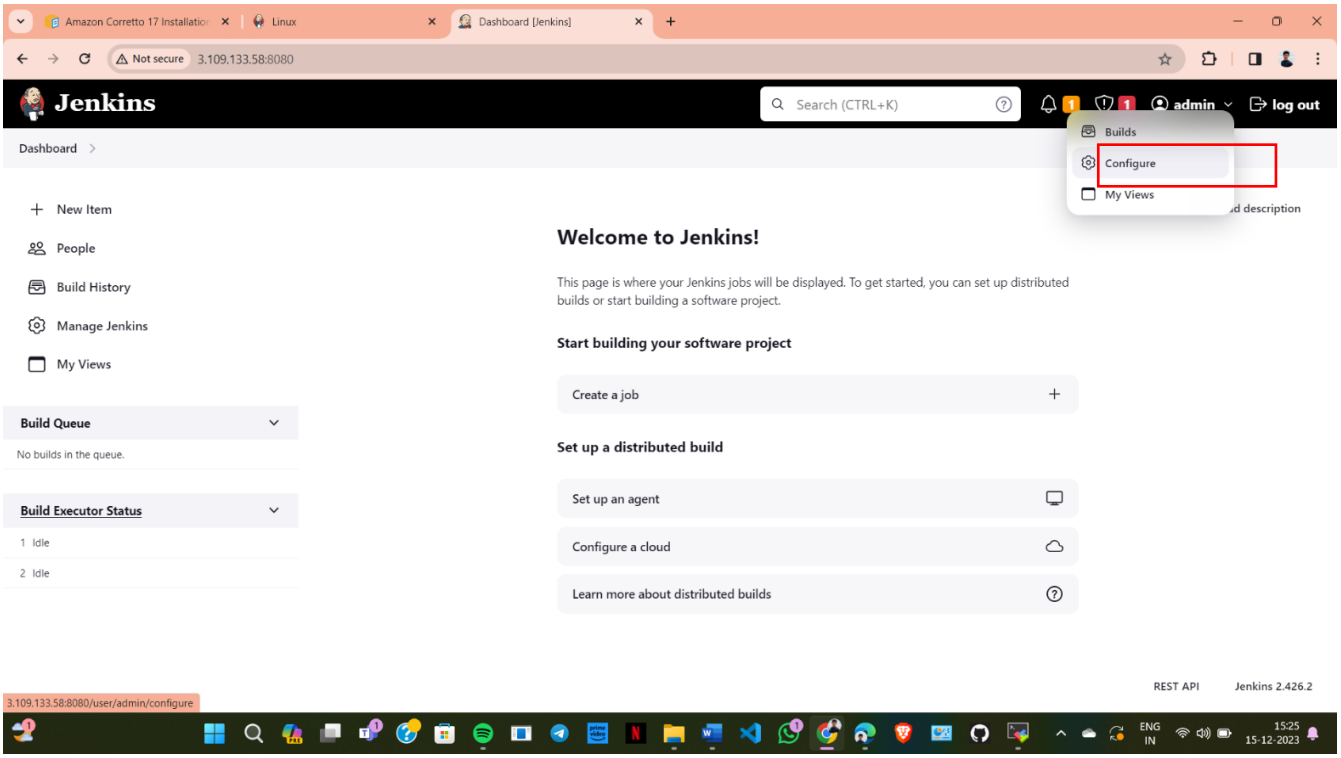

Amazon Corretto 17 Installatio
Linux
Sign in [Jenkins]
Not secure
3.109.133.58:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/var/lib/jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue

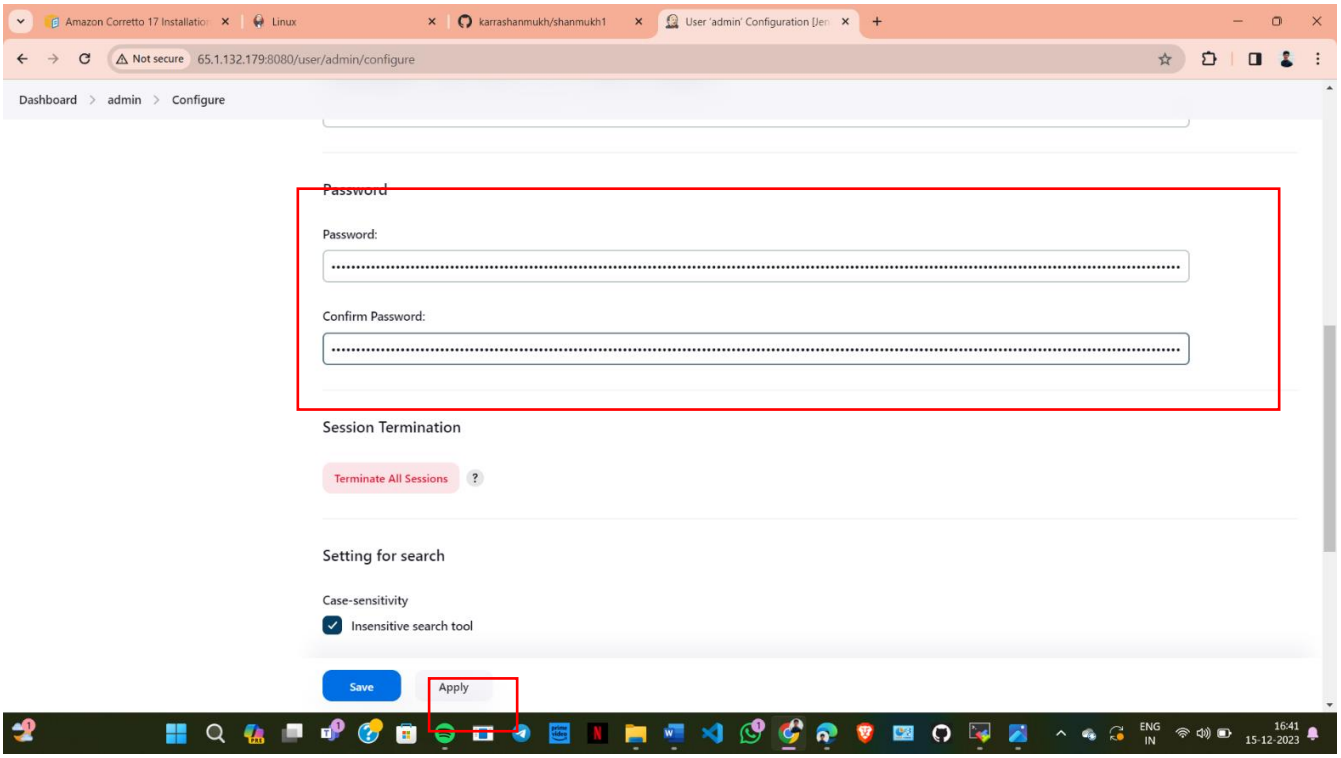

Amazon Corretto 17 Installatio
Linux
Sign in [Jenkins]
Not secure
3.109.133.58:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/var/lib/jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue



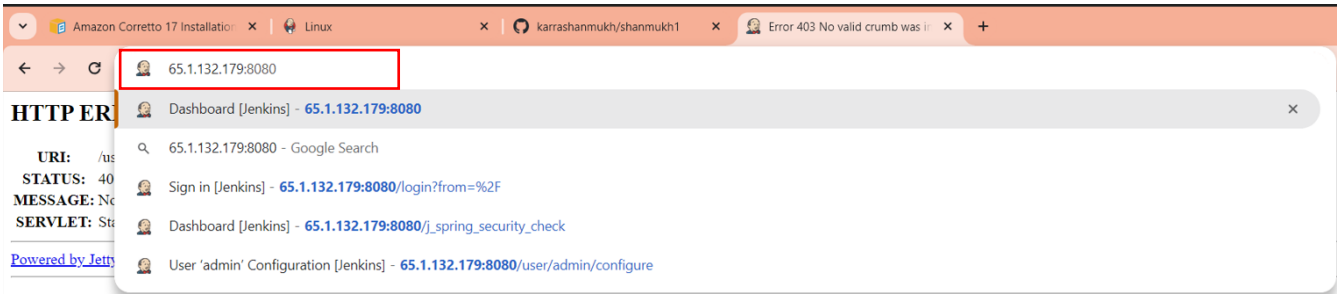
Step- 9: Now in Jenkins, goto **configure** settings and change the account password. Apply and save the changes.

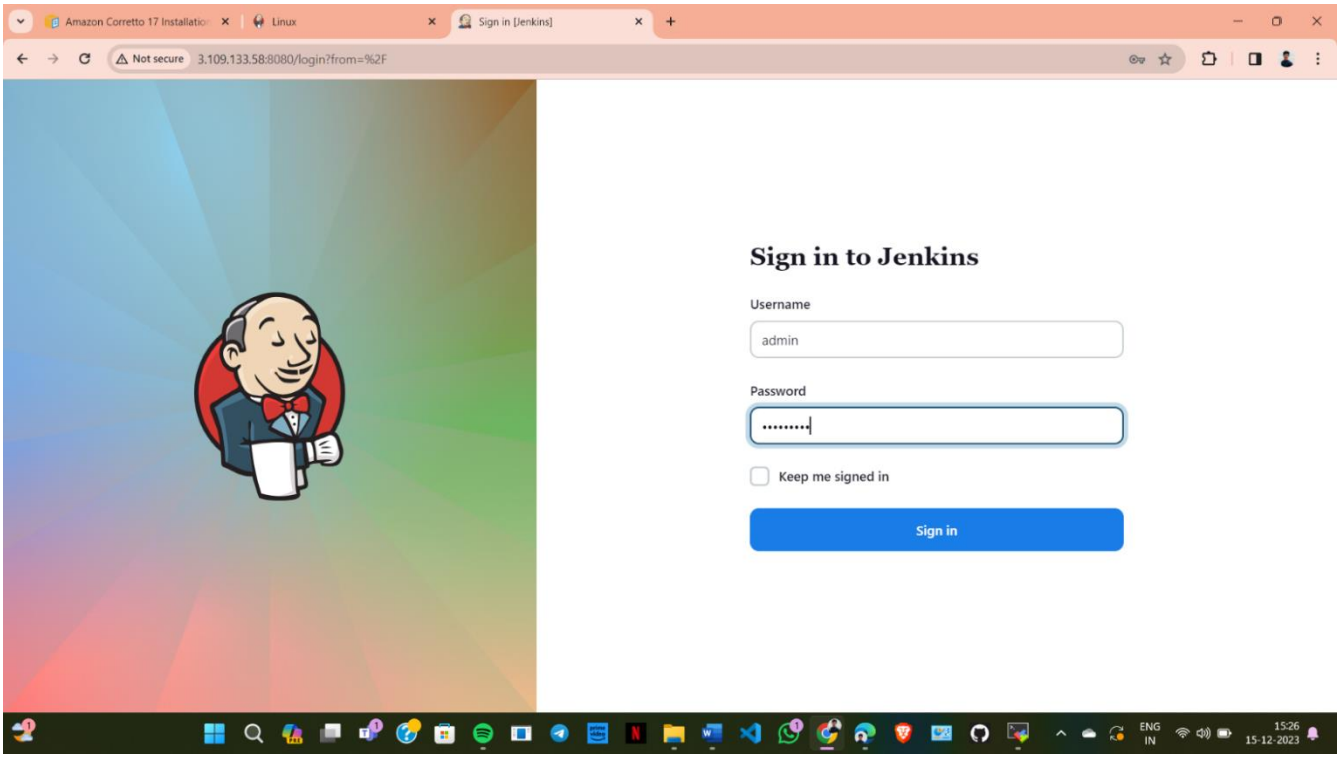
3.109.133.58:8080



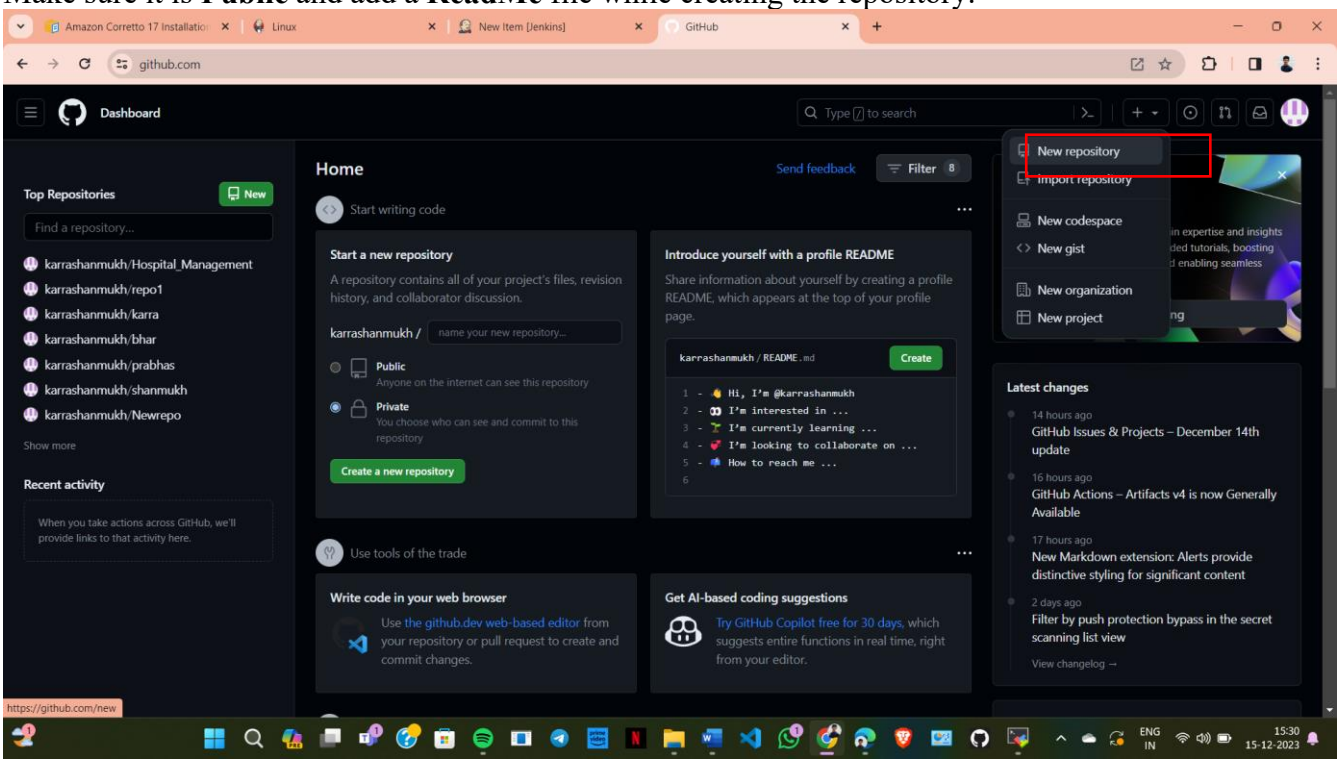


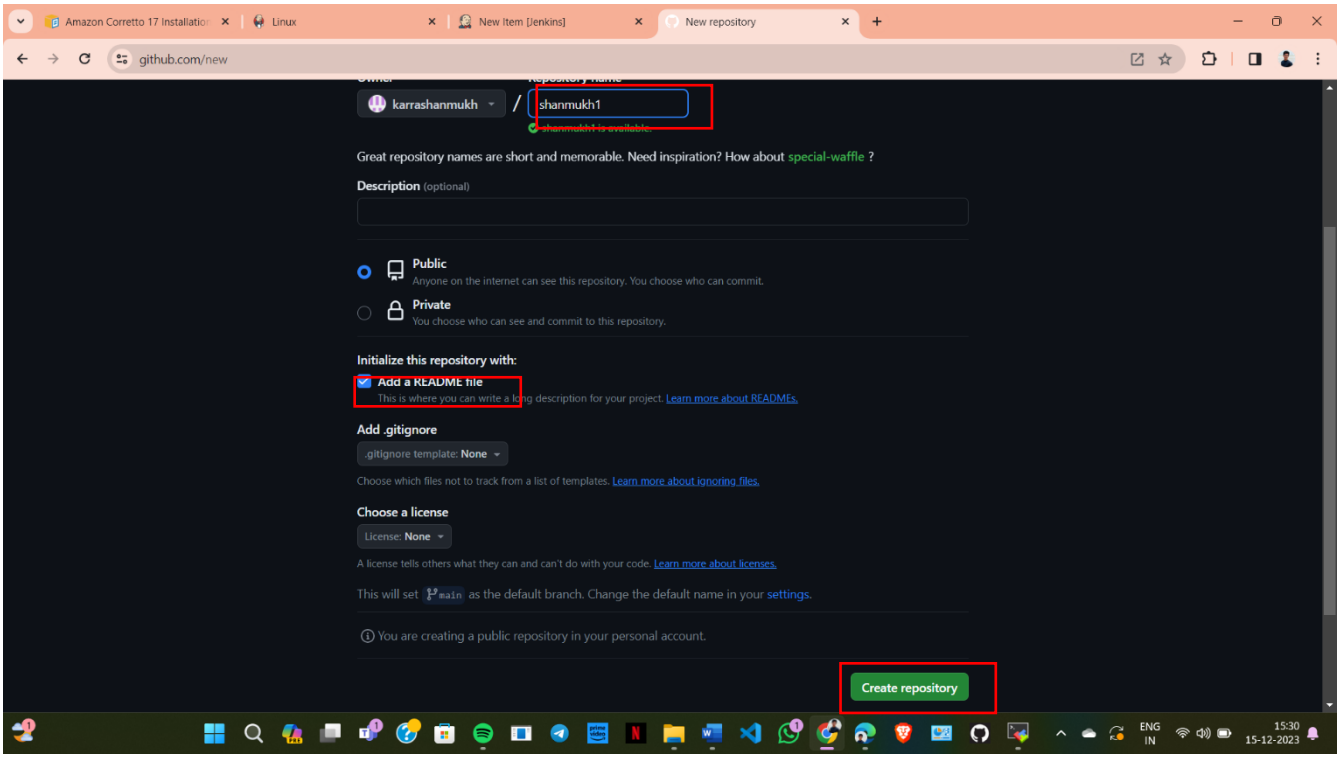
Step-10: The site now shows error after changing the password. Clear the web url and re-enter the public IP of EC2 instance along with port range(8080). Now you will be asked to enter username and password. Use **admin** as username and provide your changed password.



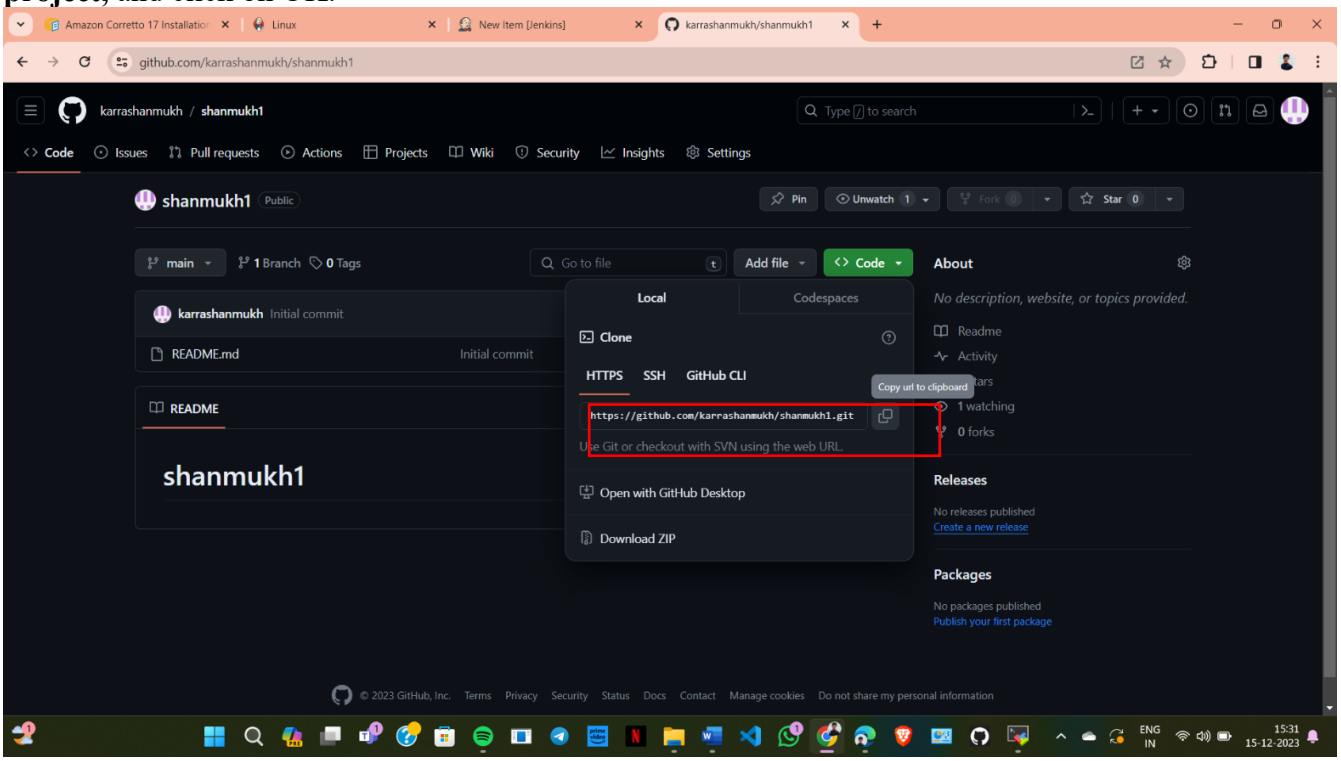


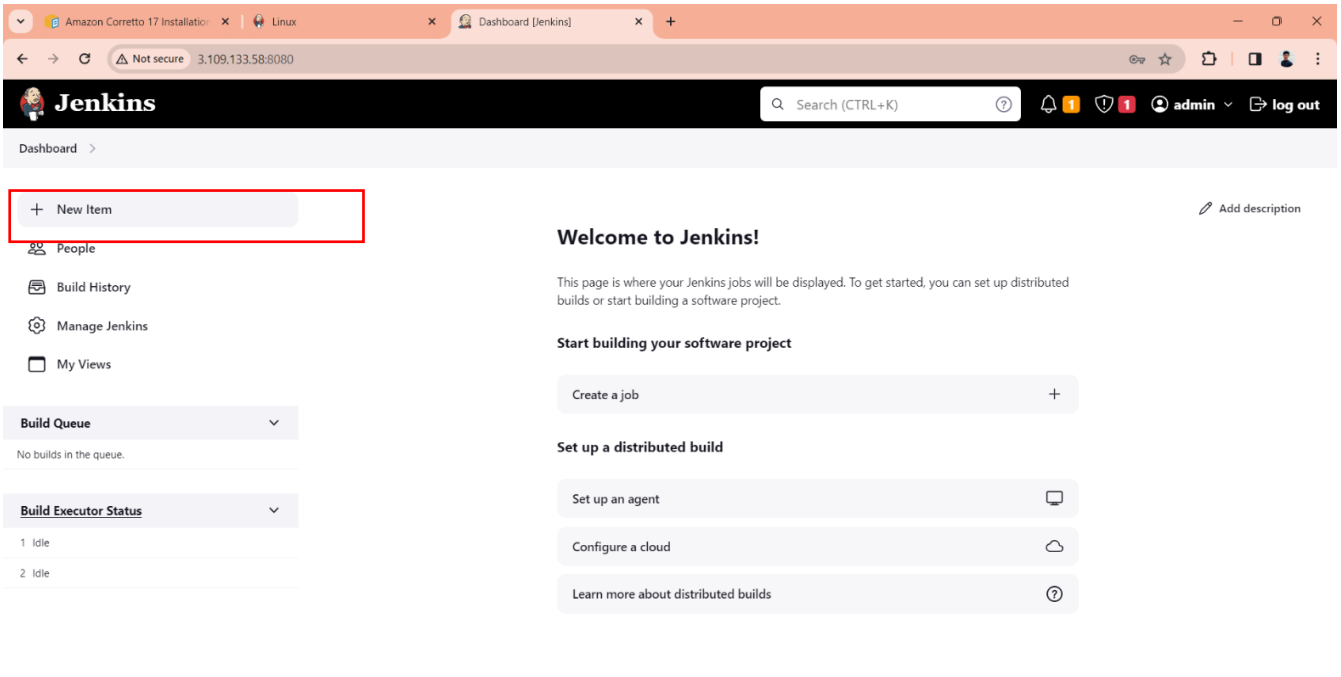
Step-11: Now open and logon to your GitHub account. Create a new repository and give it a name. Make sure it is **Public** and add a **ReadMe** file while creating the repository.

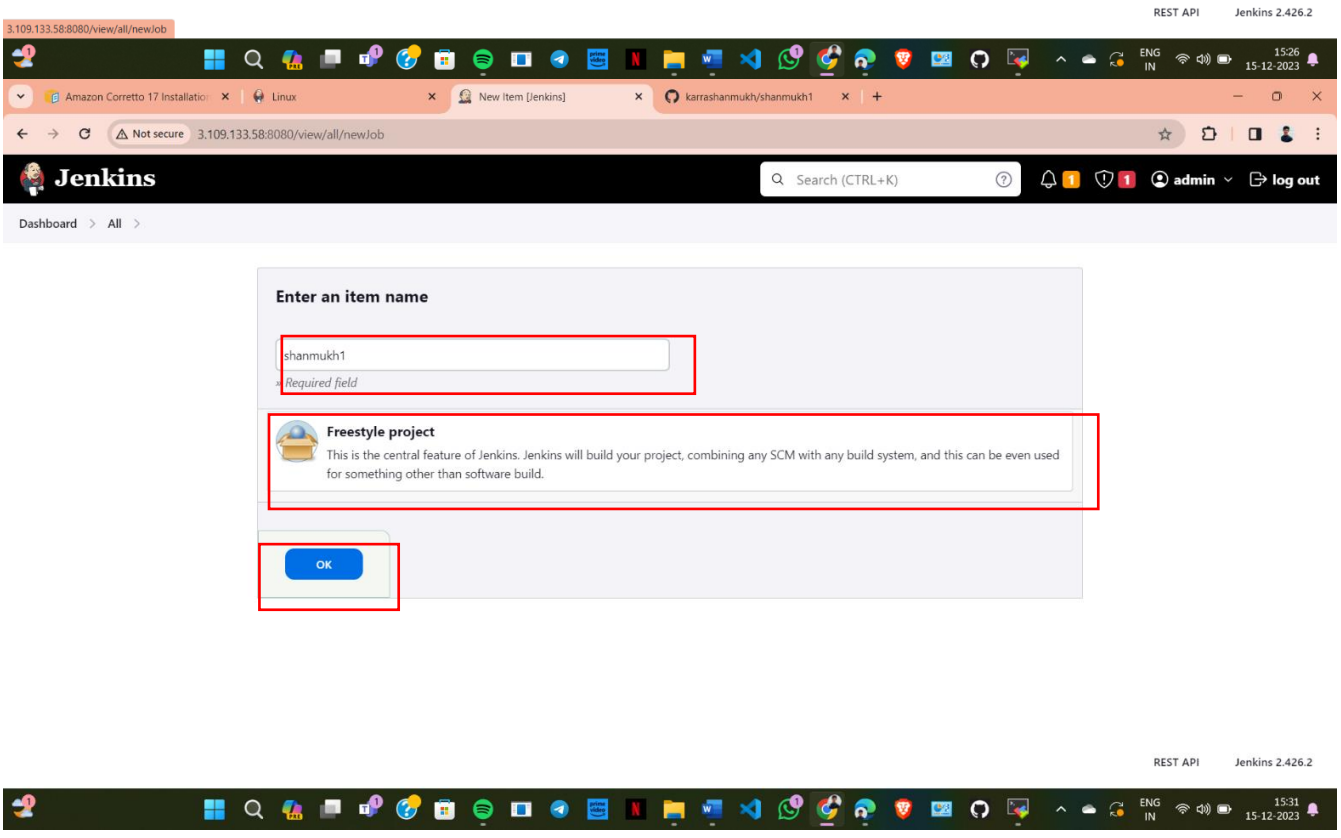




Step-12: Copy the repository's url. Goto Jenkins and click on new item, give it a name, select **free style project,** and click on OK.

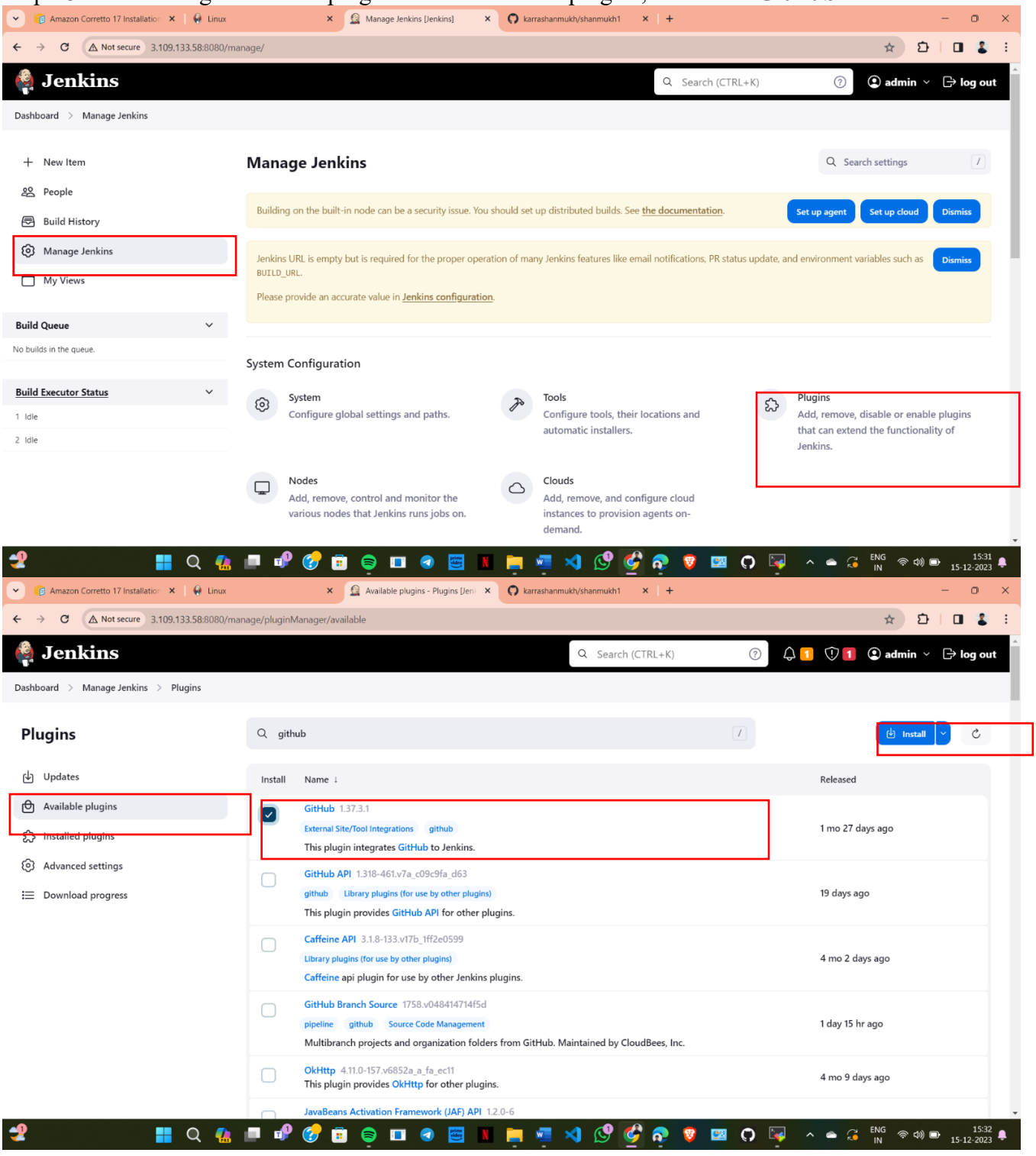


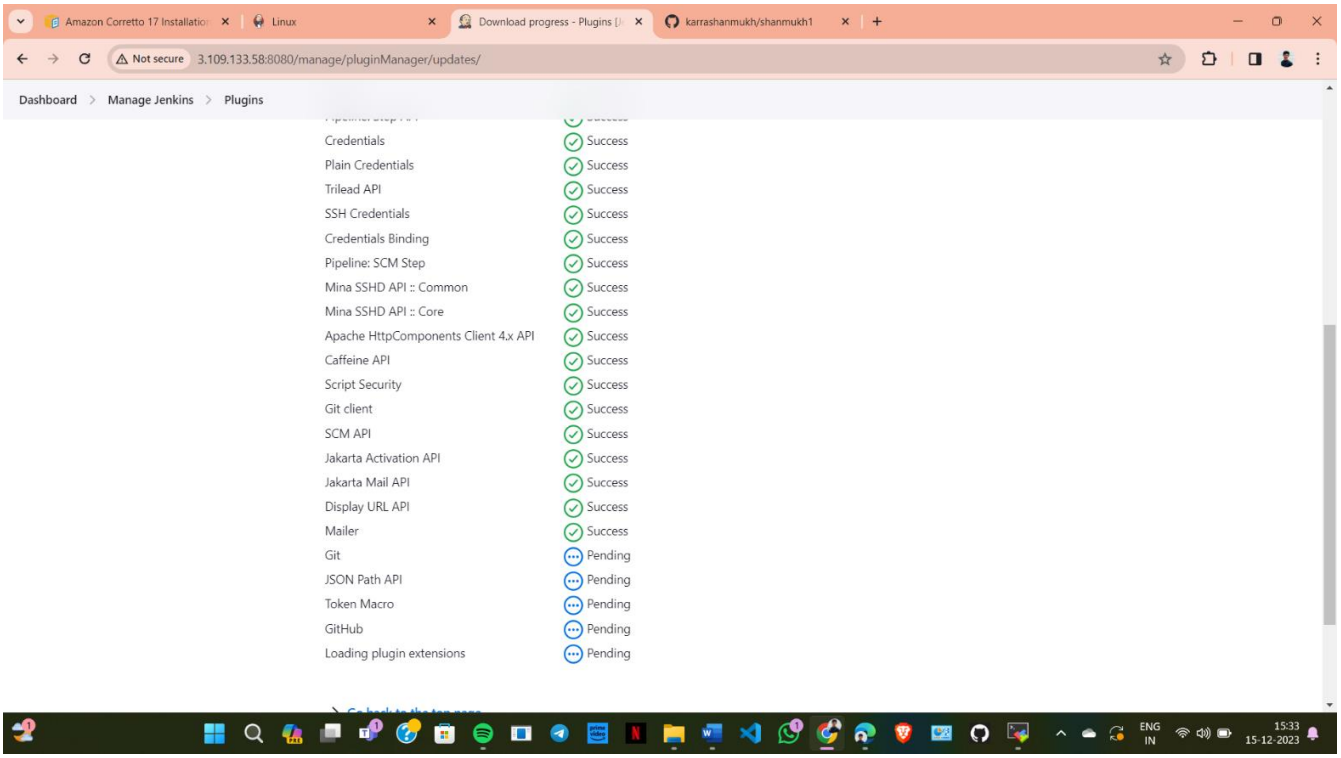

Jenkins
Search (CTRL+K)
admin
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds


REST API
Jenkins 2.426.2
Jenkins
Search (CTRL+K)
admin
log out
Dashboard
All
Enter an item name
shanmukh1
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
OK

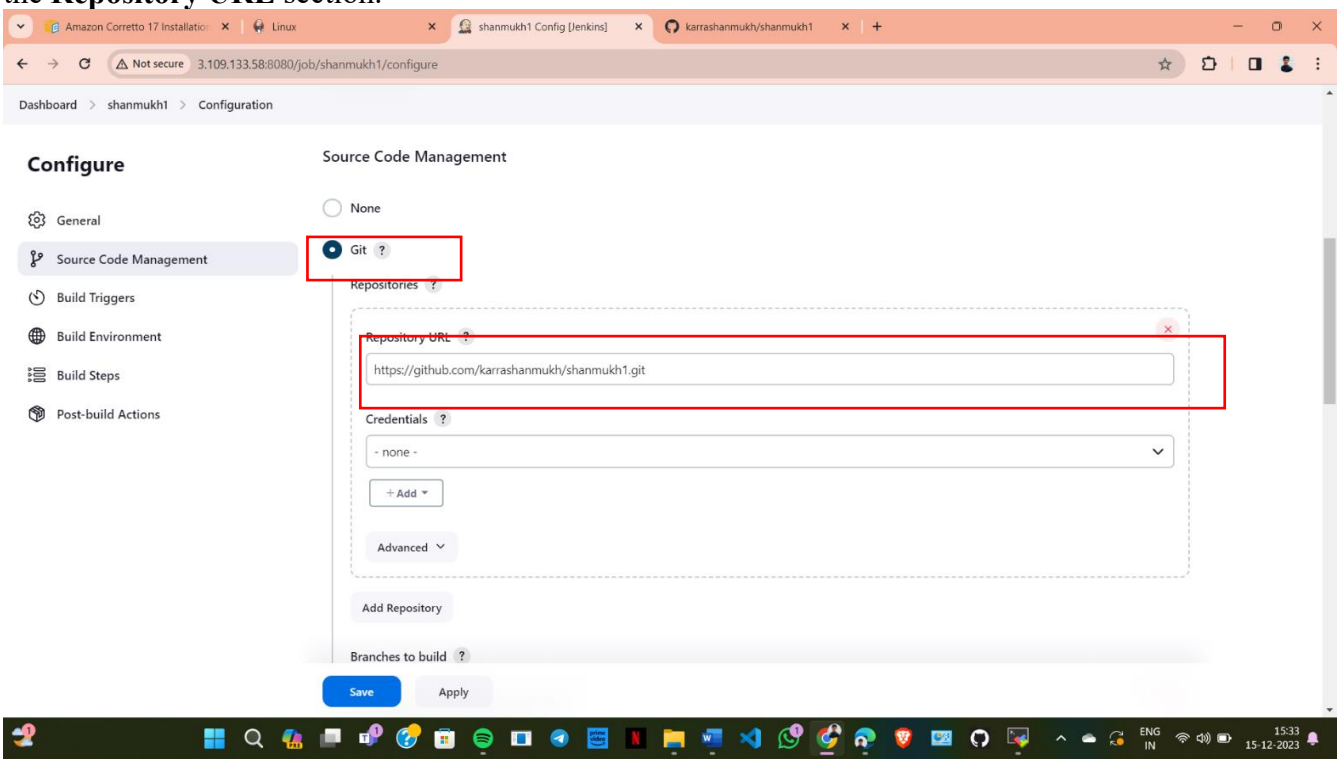
REST API
Jenkins 2.426.2

Step-13: Goto manage Jenkins > plugins. In the available plugins, search for **GitHub** and install.

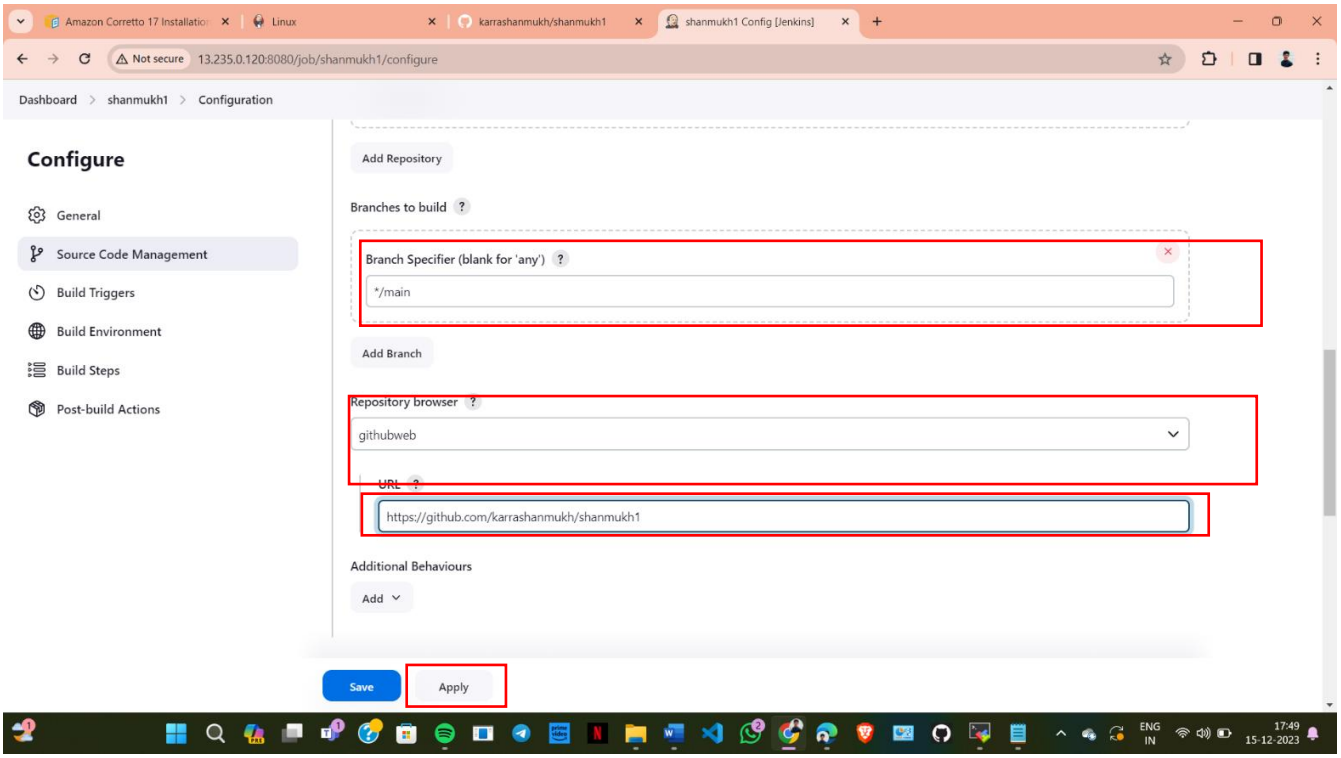




Step-14: Goto **Source Code Management** and select **git** , paste the copied github repository's url in the **Repository URL** section.



Step-15: In the **Branch Specifier** section, enter ***/main** , and in **Repository browser** section, enter githubweb. Enter the repository's URL in the **URL** section by removing ".git" at the end of URL. Apply and save the changes. Click on Build.



Maven:

Maven installation commands:

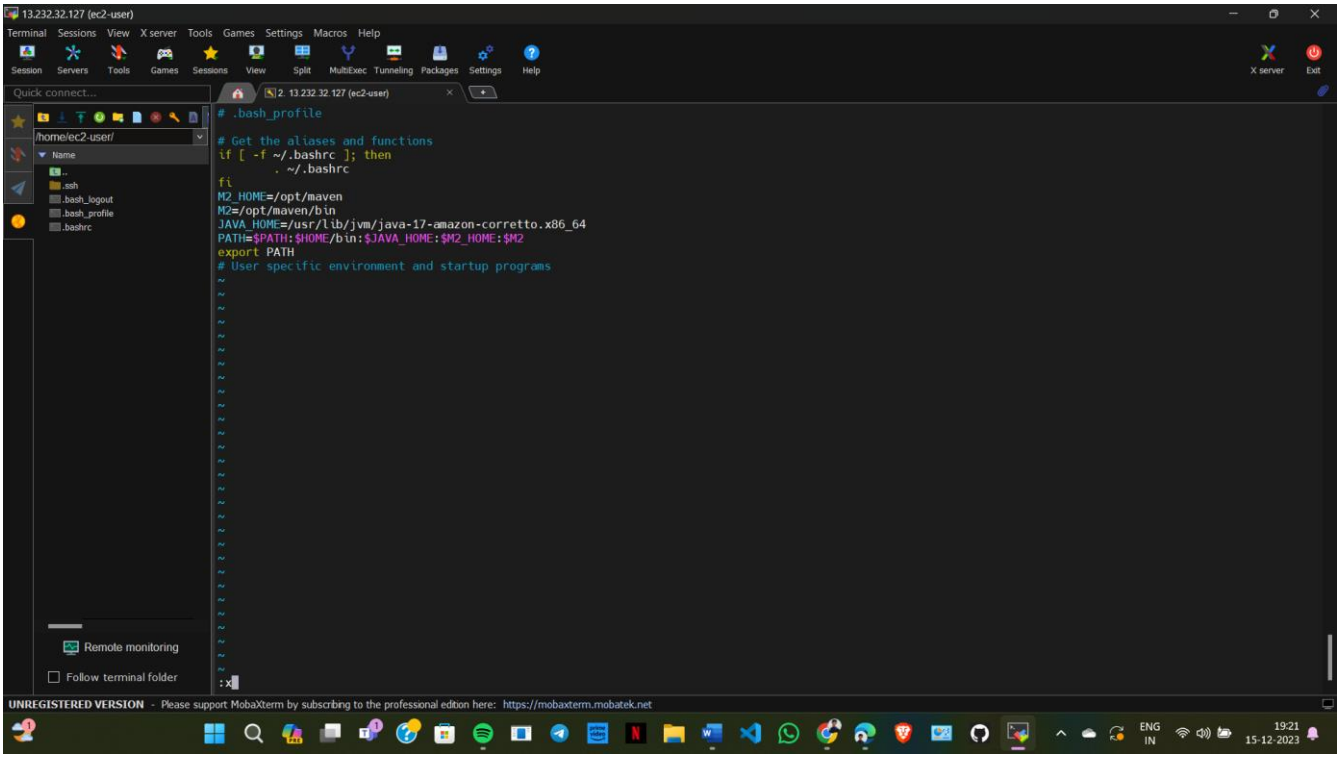
[root@ip-172-31-14-200 ~]# sudo su -
[root@ip-172-31-14-200 ~]# cd /opt/
[root@ip-172-31-14-200 opt]# ll
[root@ip-172-31-14-200 opt]# wget https://dlcdn.apache.org/maven/maven-3/3.9.6/binaries/apache-maven-3.9.6-bin.tar.gz
[root@ip-172-31-14-200 opt]# tar -xzvf apache-maven-3.9.6-bin.tar.gz
[root@ip-172-31-14-200 opt]# ll
[root@ip-172-31-14-200 opt]# mv apache-maven-3.9.6 maven
[root@ip-172-31-14-200 opt]# ll
[root@ip-172-31-14-200 opt]# cd maven
[root@ip-172-31-14-200 maven]# ll
[root@ip-172-31-14-200 maven]# cd bin
[root@ip-172-31-14-200 bin]# ./mvn -v

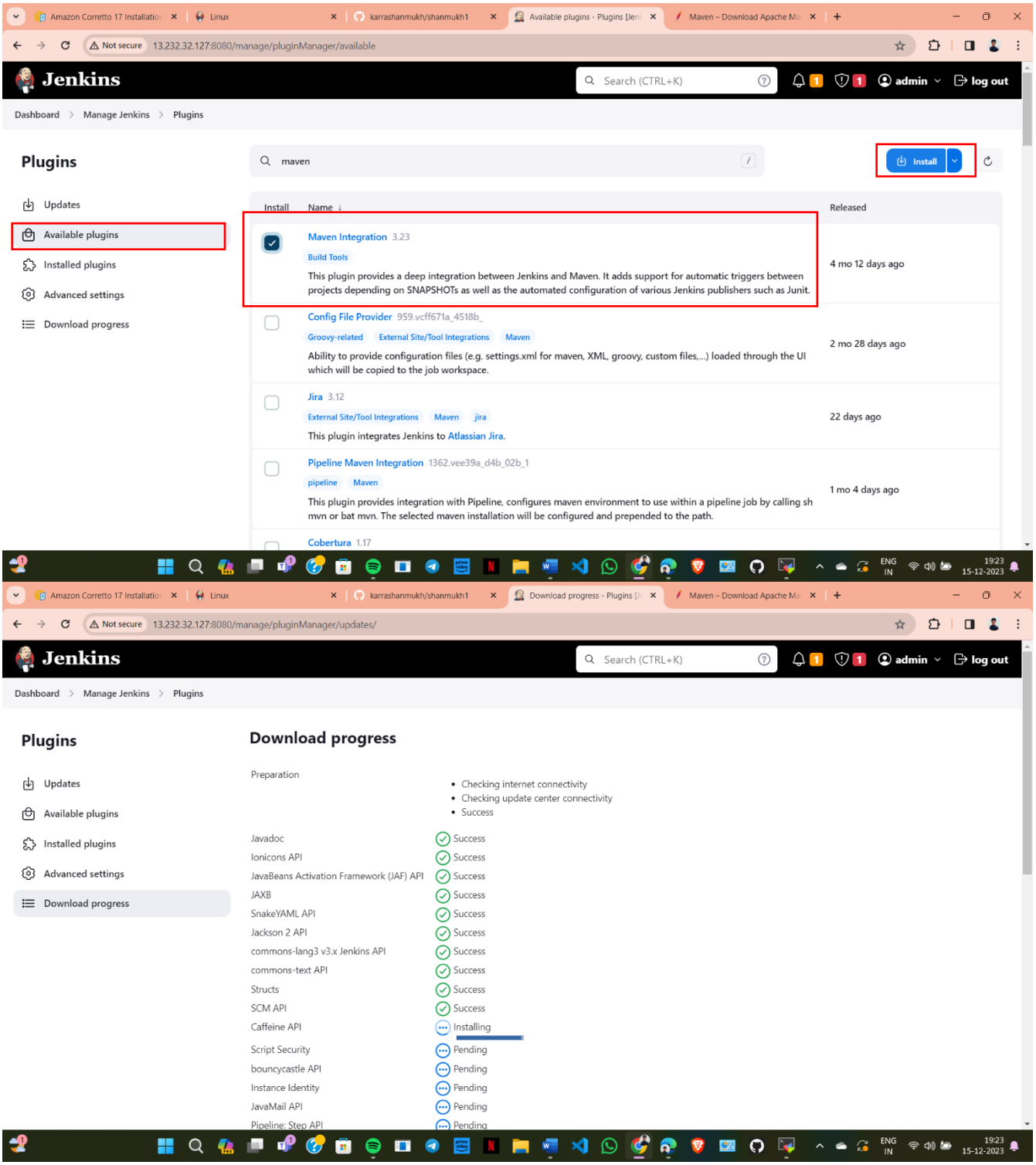
output:

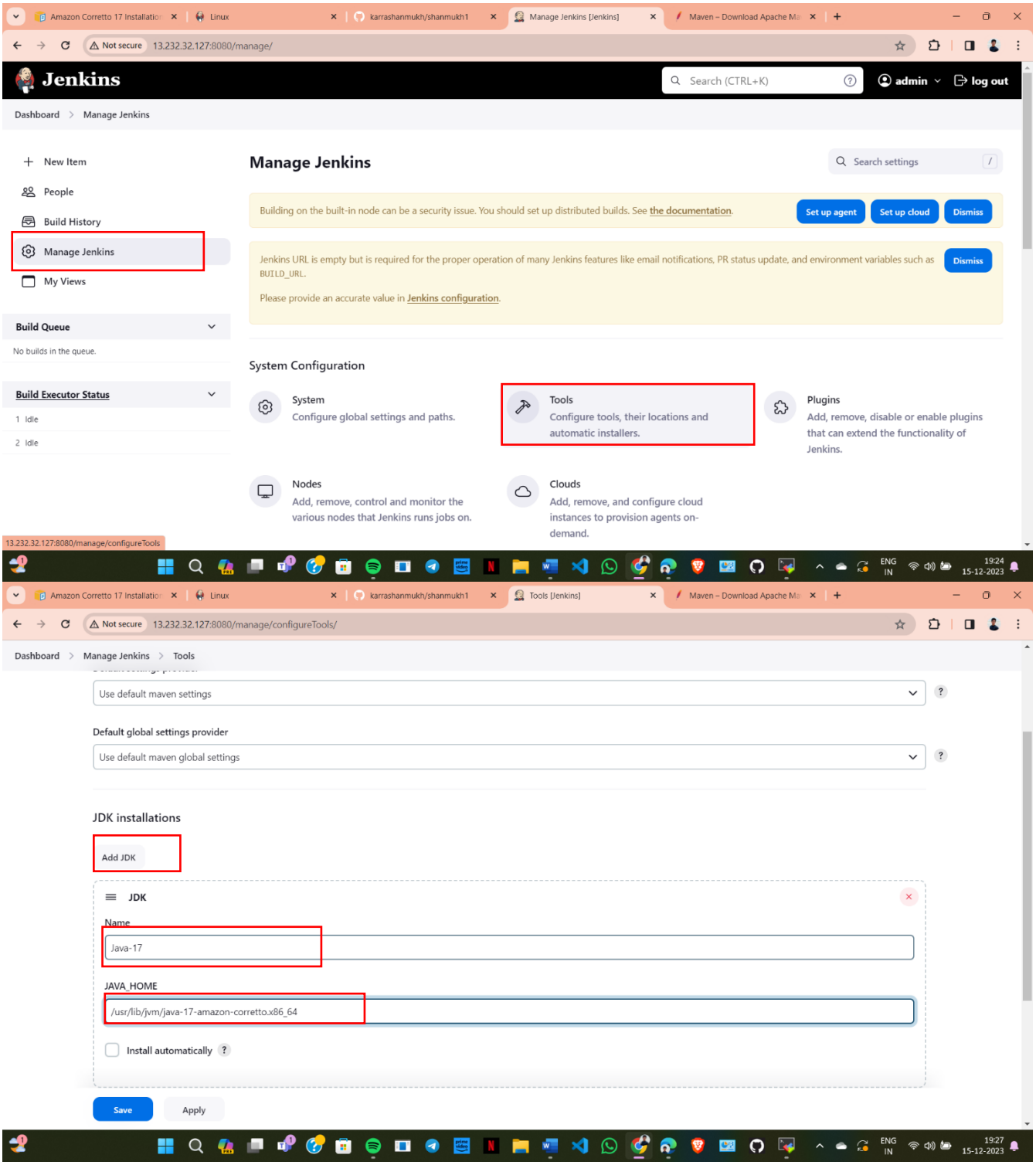
Apache Maven 3.9.6 (bc0240f3c744dd6b6ec2920b3cd08dcc295161ae)
Maven home: /opt/maven
Java version: 17.0.9, vendor: Amazon.com Inc., runtime: /usr/lib/jvm/java-17-amazon-corretto.x86_64
Default locale: en, platform encoding: UTF-8

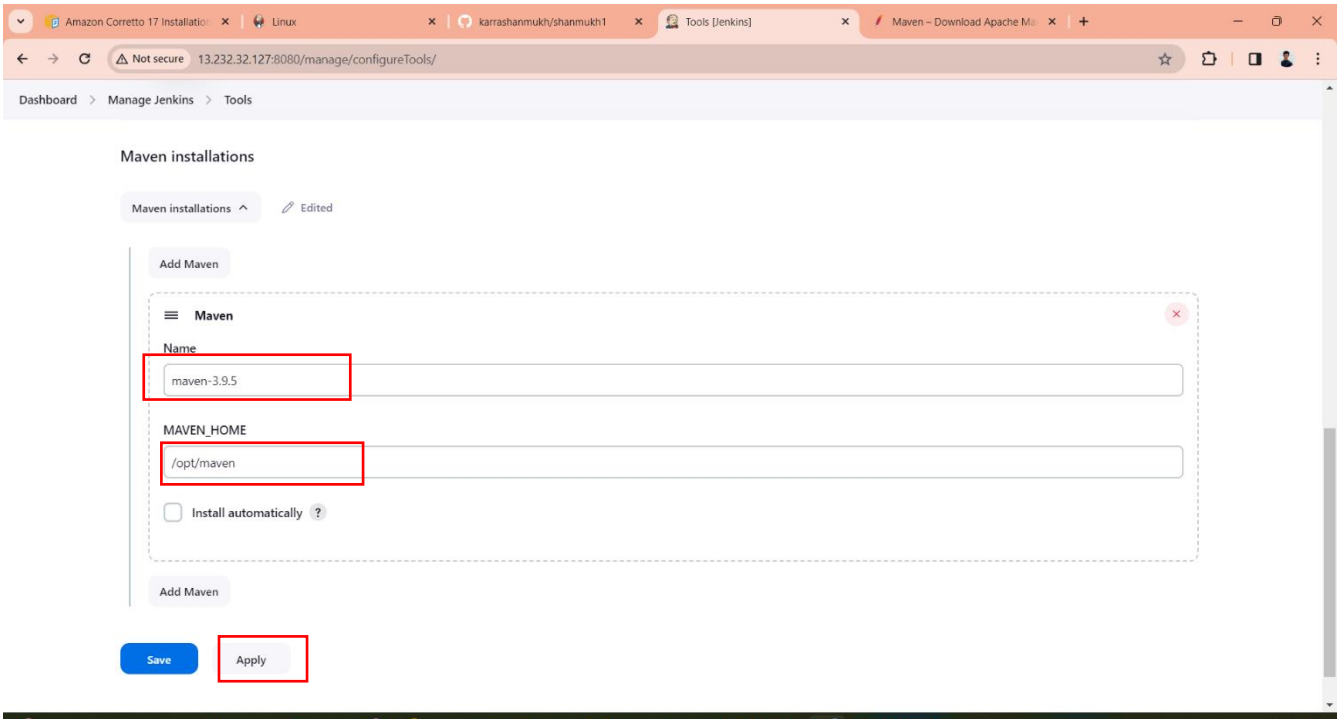
OS name: "linux", version: "6.1.66-91.160.amzn2023.x86_64", arch: "amd64", family: "unix"

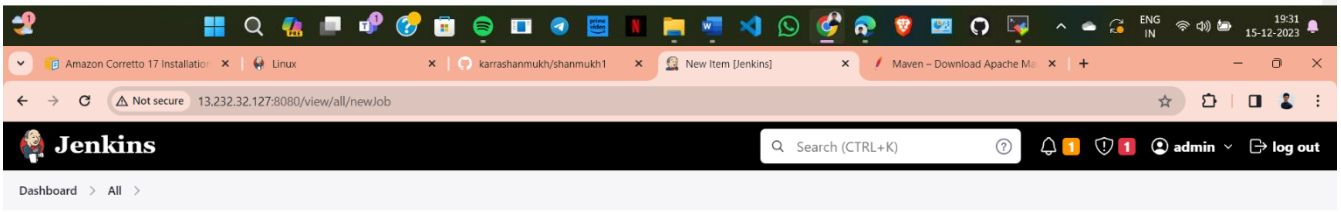
```
[root@ip-172-31-14-200 bin]# cd ~
[root@ip-172-31-14-200 ~]# pwd
/root
[root@ip-172-31-14-200 ~]# vi .bash_profile
```



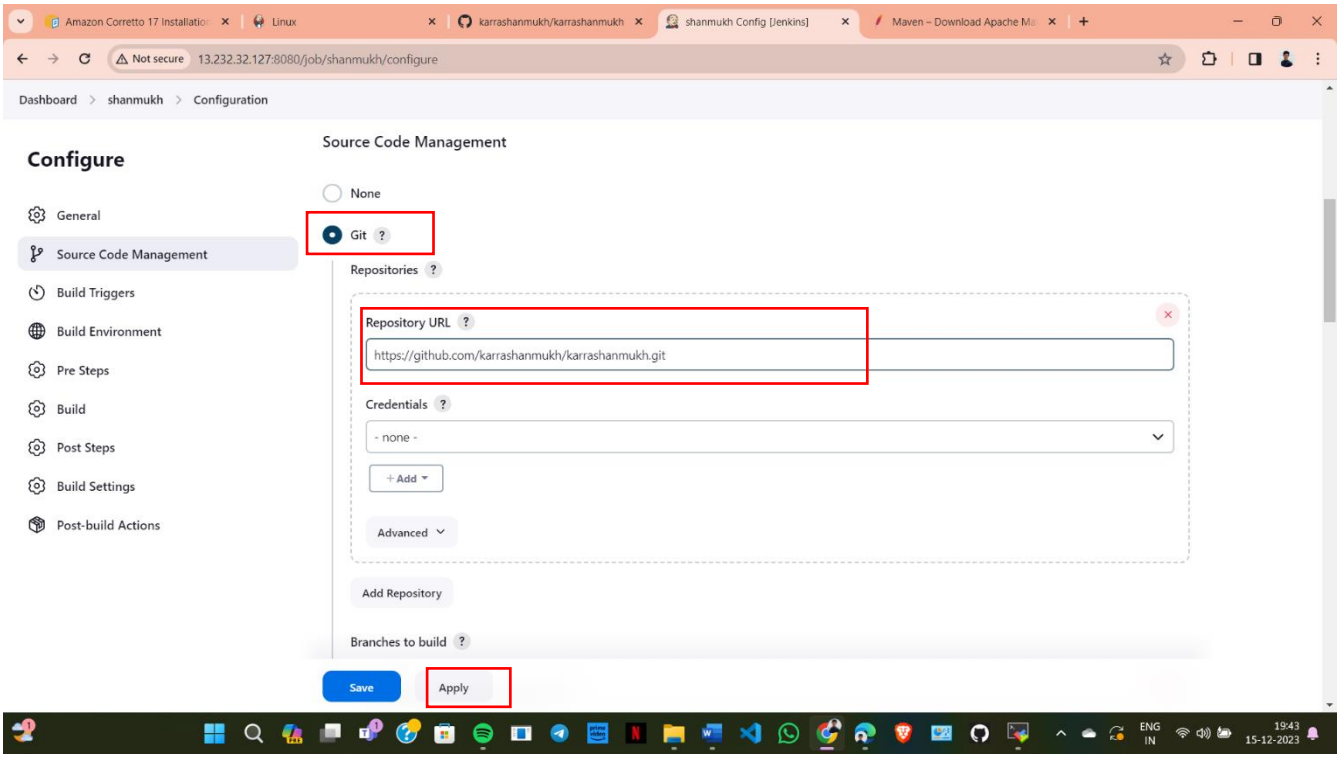

Jenkins
Search (CTRL+K)
admin
log out
Dashboard > Manage Jenkins > Plugins
Plugins
maven
Install
Updates
Available plugins
Installed plugins
Advanced settings
Download progress
Install
Name
Released
Maven Integration 3.23
Build Tools
This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit.
4 mo 12 days ago
Config File Provider 959.vcff671a_4518b_
Groovy-related
External Site/Tool Integrations
Maven
Ability to provide configuration files (e.g. settings.xml for maven, XML, groovy, custom files,...) loaded through the UI which will be copied to the job workspace.
2 mo 28 days ago
Jira 3.12
External Site/Tool Integrations
Maven
jira
This plugin integrates Jenkins to Atlassian Jira.
22 days ago
Pipeline Maven Integration 1362.vee39a_d4b_02b_1
pipeline
Maven
This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path.
1 mo 4 days ago
Cobertura 1.17
Jenkins
Search (CTRL+K)
admin
log out
Dashboard > Manage Jenkins > Plugins
Plugins
Download progress
Updates
Available plugins
Installed plugins
Advanced settings
Download progress
Preparation
Checking internet connectivity
Checking update center connectivity
Success
Javadoc Success
Ionicons API Success
JavaBeans Activation Framework (JAF) API Success
JAXB Success
SnakeYAML API Success
Jackson 2 API Success
commons-lang3 v3.x Jenkins API Success
commons-text API Success
Structs Success
SCM API Success
Caffeine API Installing
Script Security Pending
bouncycastle API Pending
Instance Identity Pending
JavaMail API Pending
Pipeline: Step API Pending


Jenkins
Search (CTRL+K)
admin
log out
Dashboard > Manage Jenkins
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Manage Jenkins
Search settings
Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.
Set up agent
Set up cloud
Dismiss
Jenkins URL is empty but is required for the proper operation of many Jenkins features like email notifications, PR status update, and environment variables such as BUILD_URL.
Please provide an accurate value in Jenkins configuration.
Dismiss
System Configuration
System
Configure global settings and paths.
Tools
Configure tools, their locations and automatic installers.
Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
Clouds
Add, remove, and configure cloud instances to provision agents on-demand.
13.232.32.127:8080/manage/configureTools
Dashboard > Manage Jenkins > Tools
Use default maven settings
Default global settings provider
Use default maven global settings
JDK installations
Add JDK
JDK
Name
Java-17
JAVA_HOME
/usr/lib/jvm/java-17-amazon-corretto.x86_64
Install automatically
Save
Apply


Not secure 13.232.32.127:8080/manage/configureTools/
Dashboard > Manage Jenkins > Tools
Maven installations
Maven installations
Edited
Add Maven
Maven
Name
maven-3.9.5
MAVEN_HOME
/opt/maven
Install automatically
Add Maven
Save
Apply



Not secure 13.232.32.127:8080/view/all/newJob
Jenkins
Search (CTRL+K)
admin
log out
Dashboard > All >
Enter an item name
shanmukh
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
OK
REST API
Jenkins 2.426.2



Step3:create another instances for tomcat

➔ To build maven application using Jenkins and deploy it Tomcat.

➔ First Go to AWS webpage and Login into console. Then search EC2 click on it.

➔Before, Tomcat server create Jenkins server and install maven in it and

➔Create another Instance for Tomcat Server. Create a session and paste public IP address in Mobaxterm.

➔Download tomcat by copy link address from Tomcat official Website. And unzip the zip file and rename it to tomcat

```
[ec2-user@ip-172-31-27-252 ~]$ sudo su -
[root@ip-172-31-27-252 ~]# cd /opt
[root@ip-172-31-27-252 opt]# wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.83/bin/apache-tomcat-9.0.83.zip
--2023-12-11 16:09:50--  https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.83/bin/apache-tomcat-9.0.83.zip
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12344614 (12M) [application/zip]
Saving to: 'apache-tomcat-9.0.83.zip'

apache-tomcat-9.0.83.zip              100%[=========================================================================>]

2023-12-11 16:09:50 (229 MB/s) - 'apache-tomcat-9.0.83.zip' saved [12344614/12344614]

[root@ip-172-31-27-252 opt]# unzip apache-tomcat-9.0.83.zip
Archive:  apache-tomcat-9.0.83.zip
```

```
[root@ip-172-31-27-252 opt]# ll -a
total 12072
drwxr-xr-x.  4 root root       77 Dec 11 16:10 .
dr-xr-xr-x. 18 root root      237 Nov 10 19:53 ..
drwxr-xr-x.  9 root root    16384 Nov  9 20:57 apache-tomcat-9.0.83
-rw-r--r--.  1 root root 12344614 Nov  9 21:47 apache-tomcat-9.0.83.zip
drwxr-xr-x.  4 root root       33 Nov 10 19:54 aws
[root@ip-172-31-27-252 opt]# mv apache-tomcat-9.0.83 tomcat
[root@ip-172-31-27-252 opt]# cd tomcat
[root@ip-172-31-27-252 tomcat]# chmod -R u=rwx bin
[root@ip-172-31-27-252 tomcat]# ln -s /opt/tomcat/bin/startup.sh /usr/local/bin/tomcatup
[root@ip-172-31-27-252 tomcat]# ln -s /opt/tomcat/bin/shutdown.sh /usr/local/bin/tomcatdown
[root@ip-172-31-27-252 tomcat]# tomcatup
Neither the JAVA_HOME nor the JRE_HOME environment variable is defined
At least one of these environment variable is needed to run this program
[root@ip-172-31-27-252 tomcat]# sudo yum install java-17-amazon-corretto-devel
Last metadata expiration check: 0:31:50 ago on Mon Dec 11 15:41:59 2023.
Dependencies resolved.
================================================================================
 Package                                 Architecture          Version
================================================================================
Installing:
 java-17-amazon-corretto-devel           x86_64                1:17.0.9+8-1.amzn2023.1
```

➔ Change the permissions for bin in tomcat. And Install JDK – 17 into the instance.

➔ And change the startup.sh and shutdown.sh to the tomcatup and tomcatdown.

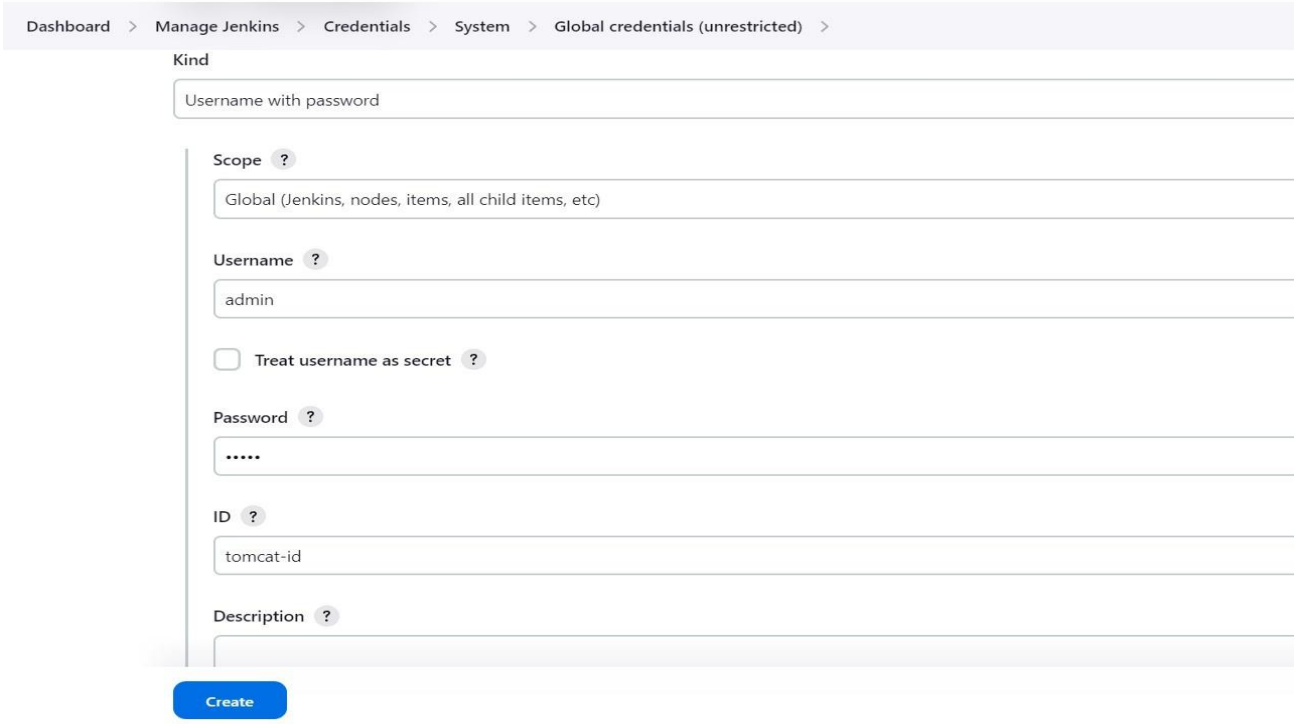
➔ Now, Start Tomcat Server and Modify the context.xml in the host-manager and manager. By commenting the valve tag

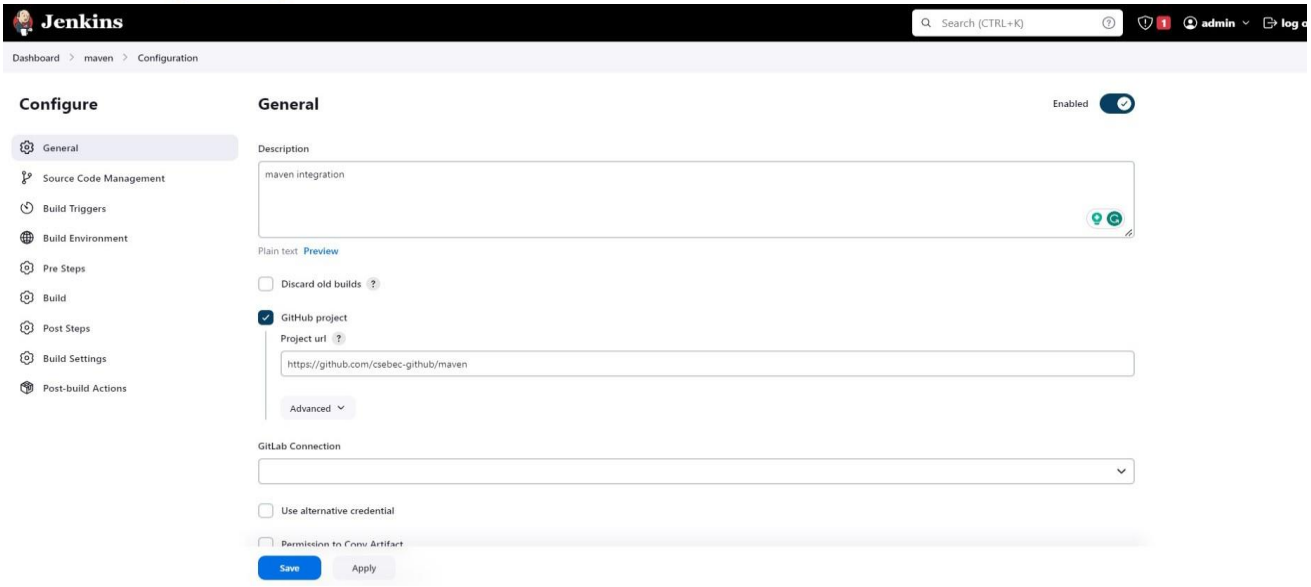
```
Complete!
[root@ip-172-31-27-252 tomcat]# tomcatup
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
[root@ip-172-31-27-252 tomcat]# find / -name context.xml
/opt/tomcat/conf/context.xml
/opt/tomcat/webapps/docs/META-INF/context.xml
/opt/tomcat/webapps/examples/META-INF/context.xml
/opt/tomcat/webapps/host-manager/META-INF/context.xml
/opt/tomcat/webapps/manager/META-INF/context.xml
[root@ip-172-31-27-252 tomcat]# vi /opt/tomcat/webapps/host-manager/META-INF/context.xml
[root@ip-172-31-27-252 tomcat]# vi /opt/tomcat/webapps/manager/META-INF/context.xml
[root@ip-172-31-27-252 tomcat]# cd conf
[root@ip-172-31-27-252 conf]# vi tomcat-users.xml
[root@ip-172-31-27-252 conf]#
```

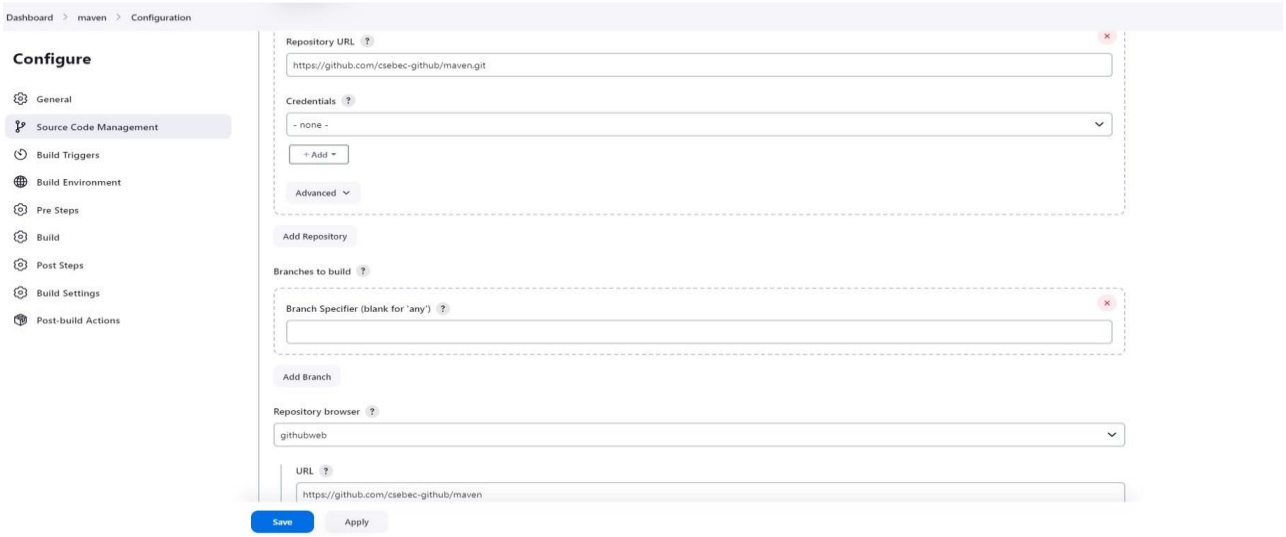
➔ Move to conf and modify tomcat-users.xml by adding roles and users

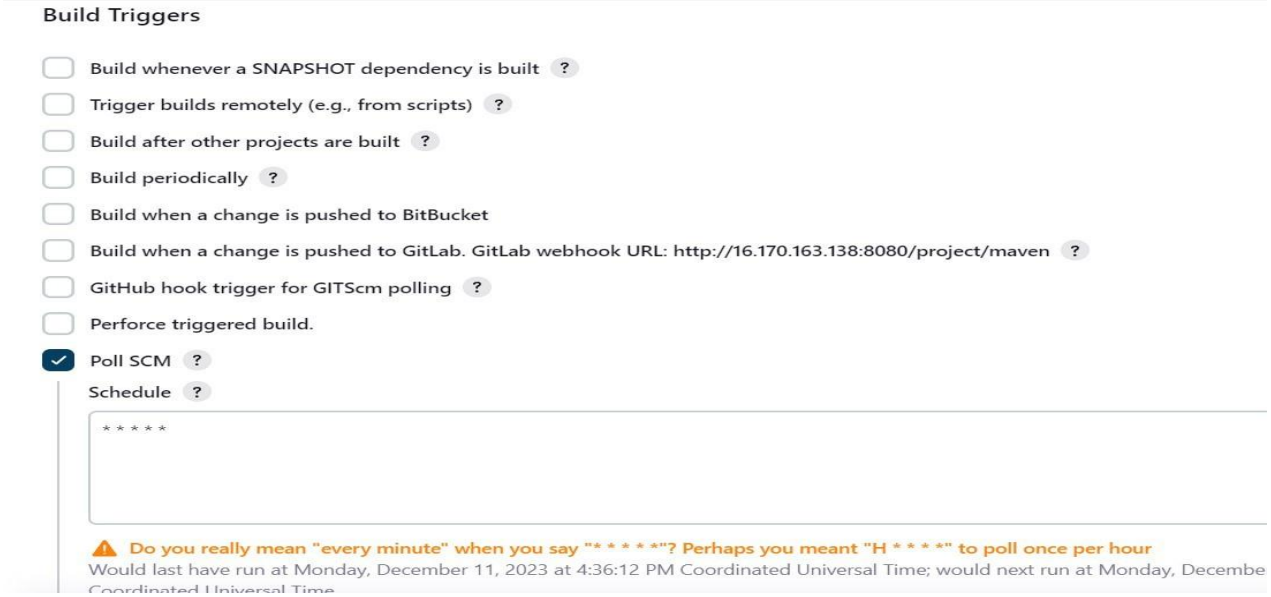
```
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
<role rolename="manager-gui"/>
 <role rolename="manager-script"/>
 <role rolename="manager-jmx"/>
 <role rolename="manager-status"/>
  <user username="admin" password="admin" roles="manager-gui,manager-script,manager-jmx,manager-status"/>
  <user username="deployer" password="deployer" roles="manager-script"/>
  <user username="user" password="user" roles="manager-gui"/>
```

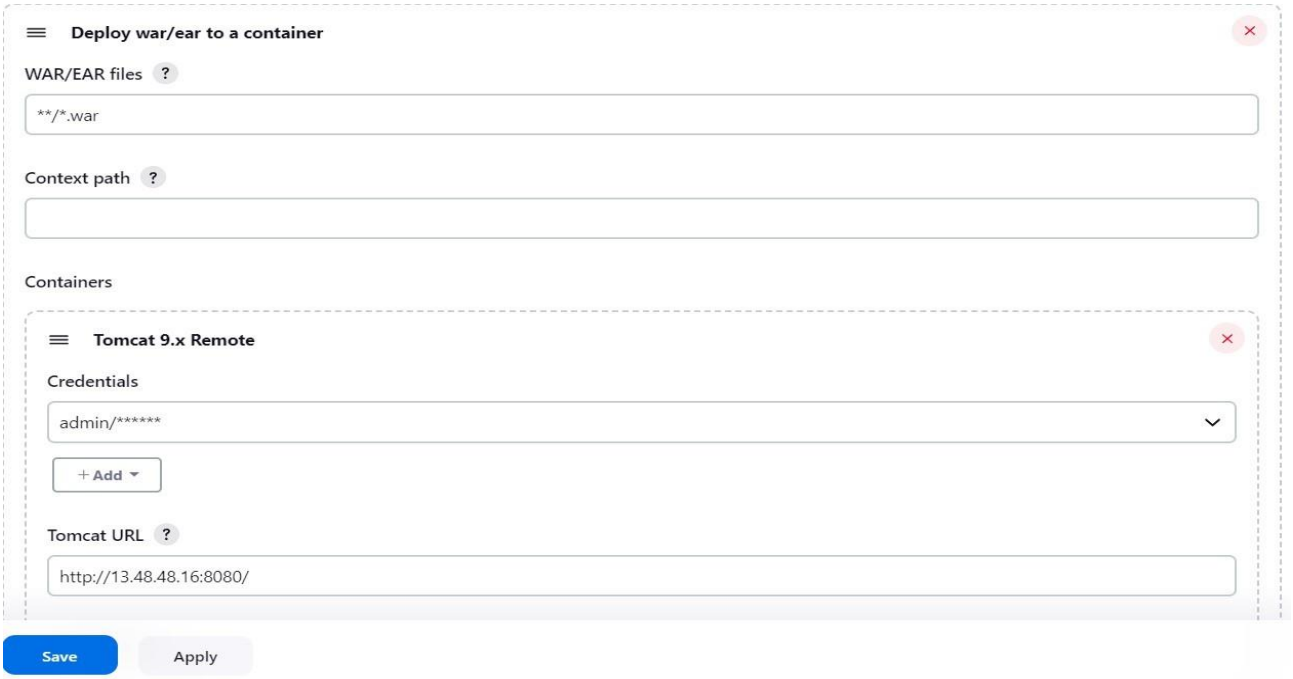
➔ Install plugin of Deploy to container. Add credentials of tomcat to Jenkins global credentials.






Dashboard > maven > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Repository URL
https://github.com/csebec-github/maven.git
Credentials
- none -
+ Add
Advanced
Add Repository
Branches to build
Branch Specifier (blank for 'any')
Add Branch
Repository browser
githubweb
URL
https://github.com/csebec-github/maven
Save
Apply


Build Triggers
Build whenever a SNAPSHOT dependency is built
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
Build when a change is pushed to BitBucket
Build when a change is pushed to GitLab. GitLab webhook URL: http://16.170.163.138:8080/project/maven
GitHub hook trigger for GITScm polling
Perforce triggered build.
Poll SCM
Schedule
* * * * *
Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour
Would last have run at Monday, December 11, 2023 at 4:36:12 PM Coordinated Universal Time; would next run at Monday, Decembe
Coordinated Universal Time



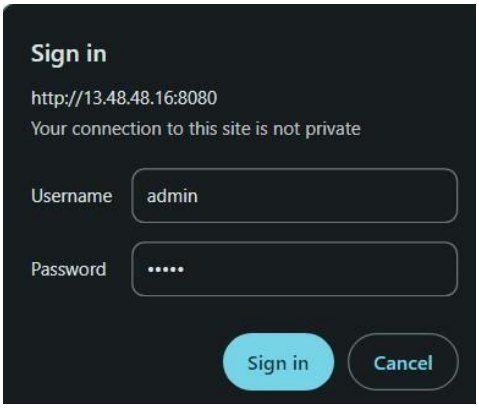
➔ After all the above steps save and apply.

```
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/maven/maven/src/main/webapp]
[INFO] Building war: /var/lib/jenkins/workspace/maven/maven/target/maven.war
[INFO]
[INFO] --- install:3.1.1:install (default-install) @ maven ---
[INFO] Installing /var/lib/jenkins/workspace/maven/maven/pom.xml to /var/lib/jenkins/.m2/repository/project/maven/0.0.1-SNAPSHOT/maven-0.0.1-
SNAPSHOT.pom
[INFO] Installing /var/lib/jenkins/workspace/maven/maven/target/maven.war to /var/lib/jenkins/.m2/repository/project/maven/0.0.1-SNAPSHOT/maven-
0.0.1-SNAPSHOT.war
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  7.083 s
[INFO] Finished at: 2023-12-11T16:39:39Z
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/lib/jenkins/workspace/maven/maven/pom.xml to project/maven/0.0.1-SNAPSHOT/maven-0.0.1-SNAPSHOT.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/maven/maven/target/maven.war to project/maven/0.0.1-SNAPSHOT/maven-0.0.1-SNAPSHOT.war
channel stopped
[DeployPublisher][INFO] Attempting to deploy 1 war file(s)
[DeployPublisher][INFO] Deploying /var/lib/jenkins/workspace/maven/maven/target/maven.war to container Tomcat 9.x Remote with context null
  [/var/lib/jenkins/workspace/maven/maven/target/maven.war] is not deployed. Doing a fresh deployment.
  Deploying [/var/lib/jenkins/workspace/maven/maven/target/maven.war]
Finished: SUCCESS
```

➔ Now, Open New Tab and search with tomcat IP :8080 and Redirect to tomcat page.

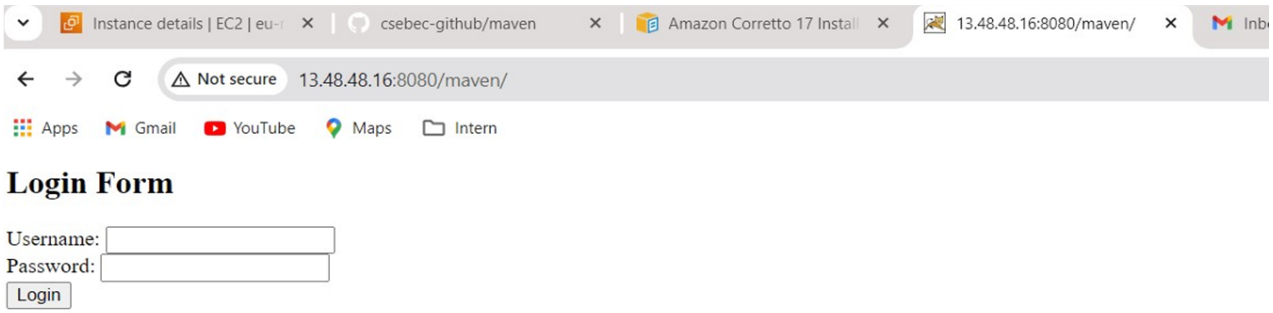
➔ There click on manager app.

➔ And Login to the tomcat using the credentials of tomcat user having deployer.



➔ After successful login. Click on the project that deployed maven project from the Jenkins after building the project.

➔ It will run project on tomcat server

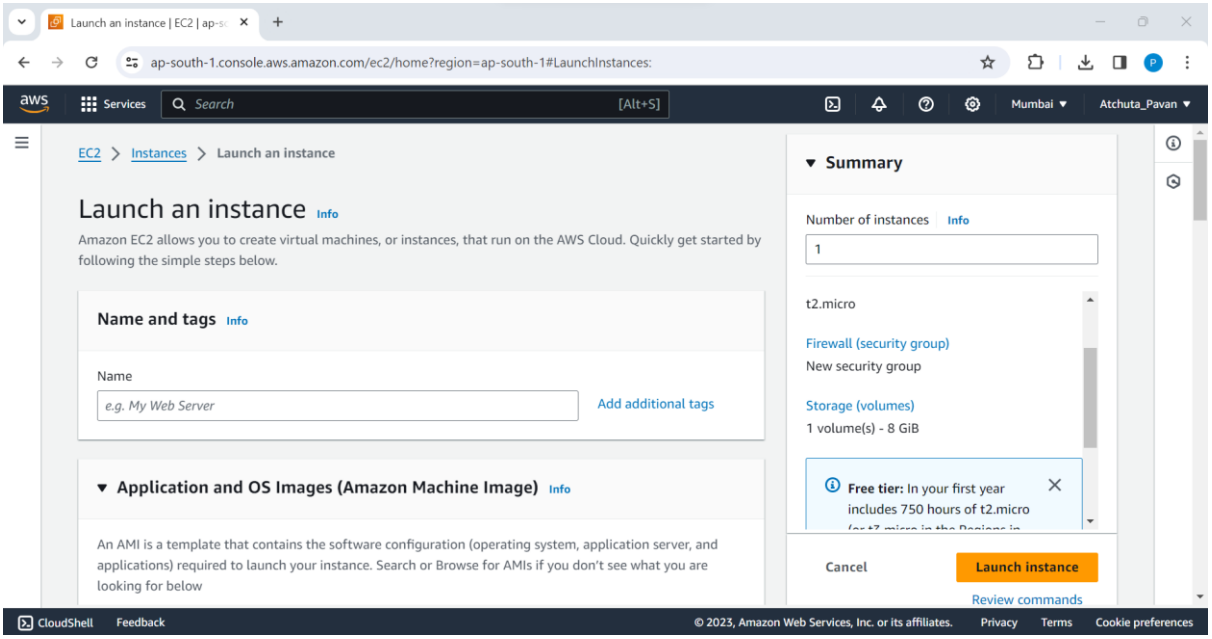


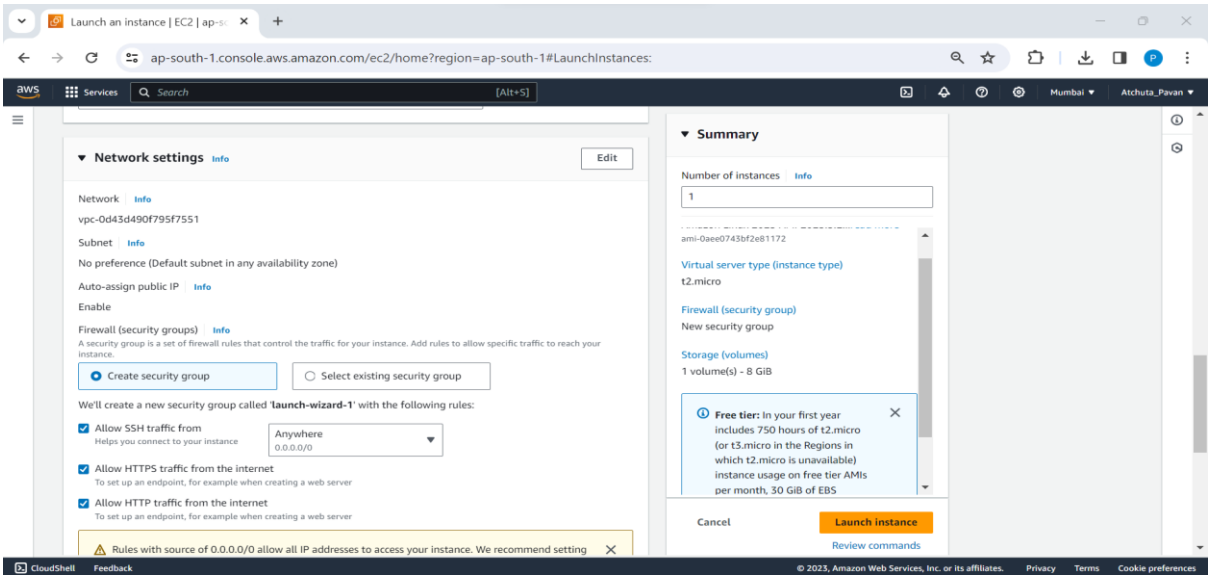
6. Aim: Deploying a Full Stack Application on AWS EC2 with Docker.

Source Code:

Step-1: Creating the aws EC2 instances:

1. Log in to the AWS Management Console.
2. Navigate to the EC2 service.
3. Click on “Launch Instance” to create a new EC2 instance.
4. Select the desired instance type, storage, and other configurations.
5. Configure the security group to allow inbound traffic on port 22 for SSH access.
6. Launch the instance and download the private key file (.pem).





Step-2: Installing Git and docker compose

1. Install Git by running the command:

```
sudo yum install git -y
```

```
Installed:
  git-2.40.1-1.amzn2023.0.1.x86_64
  git-core-2.40.1-1.amzn2023.0.1.x86_64
  git-core-doc-2.40.1-1.amzn2023.0.1.noarch
  perl-Error-1:0.17029-5.amzn2023.0.2.noarch
  perl-File-Find-1.37-477.amzn2023.0.6.noarch
  perl-Git-2.40.1-1.amzn2023.0.1.noarch
  perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64
  perl-lib-0.65-477.amzn2023.0.6.x86_64

Complete!
```

2. Install Docker by executing the following commands:

sudo yum install docker -y

sudo usermod -aG docker ec2-user

sudo service docker start

```
  Installing       : libcgroup-3.0-1.amzn2023.0.1.x86_64                                                  3/10
  Installing       : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64                                            4/10
  Installing       : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64                                   5/10
  Installing       : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64                                            6/10
  Installing       : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                                                 7/10
  Installing       : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                             8/10
  Running scriptlet: iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                             8/10
  Installing       : pigz-2.5-1.amzn2023.0.3.x86_64                                                       9/10
  Running scriptlet: docker-24.0.5-1.amzn2023.0.3.x86_64                                                 10/10
  Installing       : docker-24.0.5-1.amzn2023.0.3.x86_64                                                 10/10
  Running scriptlet: docker-24.0.5-1.amzn2023.0.3.x86_64                                                 10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

  Verifying        : pigz-2.5-1.amzn2023.0.3.x86_64                                                       1/10
  Verifying        : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64                                   2/10
  Verifying        : containerd-1.7.2-1.amzn2023.0.4.x86_64                                               3/10
  Verifying        : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                                                 4/10
  Verifying        : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                                             5/10
  Verifying        : runc-1.1.7-1.amzn2023.0.3.x86_64                                                     6/10
  Verifying        : docker-24.0.5-1.amzn2023.0.3.x86_64                                                  7/10
  Verifying        : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64                                            8/10
  Verifying        : libcgroup-3.0-1.amzn2023.0.1.x86_64                                                  9/10
  Verifying        : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64                                           10/10

Installed:
  containerd-1.7.2-1.amzn2023.0.4.x86_64                docker-24.0.5-1.amzn2023.0.3.x86_64
  iptables-libs-1.8.8-3.amzn2023.0.2.x86_64             iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
  libcgroup-3.0-1.amzn2023.0.1.x86_64                   libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64             libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64                        runc-1.1.7-1.amzn2023.0.3.x86_64

Complete!
```

```
[ec2-user@ip-172-31-36-250 ~]$ sudo usermod -aG docker ec2-user
[ec2-user@ip-172-31-36-250 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-36-250 ~]$
```

Step-3: Install Docker Compose by running the commands:

sudo curl -L "https://github.com/docker/compose/releases/download/v2.12.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

```
[ec2-user@ip-172-31-36-250 ~]$ sudo curl -L "https://github.com/docker/compose/releases/download/v2.12.2/docker-compose-$(unam
e -s)-$(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
100 42.8M  100 42.8M    0     0  26.4M      0  0:00:01  0:00:01 --:--:--  122M
[ec2-user@ip-172-31-36-250 ~]$
```

sudo mv /usr/local/bin/docker-compose /usr/bin/docker-compose

sudo chmod +x /usr/bin/docker-compose

```
[ec2-user@ip-172-31-36-250 ~]$ sudo mv /usr/local/bin/docker-compose /usr/bin/docker-compose
[ec2-user@ip-172-31-36-250 ~]$ sudo chmod +x /usr/bin/docker-compose
[ec2-user@ip-172-31-36-250 ~]$
```

Step-4: Cloning the repository

1.Change to desired path

cd /path/to/desired-directory

```
[ec2-user@ip-172-31-36-250 ~]$ sudo mkdir -p /path/to/desired-directory
[ec2-user@ip-172-31-36-250 ~]$ cd /path/to/desired-directory
[ec2-user@ip-172-31-36-250 desired-directory]$
```

2.Cloning the repository git:

git clone https://github.com/madhurajayashanka/docker-mysql-nodejs-reactjs-app.git

```
[ec2-user@ip-172-31-36-250 desired-directory]$ sudo git clone https://github.com/madhurajayashanka/docker-mysql-nodejs-reactjs
-app.git
Cloning into 'docker-mysql-nodejs-reactjs-app'...
remote: Enumerating objects: 29514, done.
remote: Counting objects: 100% (31/31), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 29514 (delta 6), reused 30 (delta 6), pack-reused 29483
Receiving objects: 100% (29514/29514), 51.13 MiB | 6.46 MiB/s, done.
Resolving deltas: 100% (5281/5281), done.
Updating files: 100% (37870/37870), done.
```

Step-5: Configuring and Running the Dockerized Application

1. Change into the project directory:

cd docker-mysql-nodejs-reactjs-app

```
[ec2-user@ip-172-31-36-250 desired-directory]$ cd docker-mysql-nodejs-reactjs-app
[ec2-user@ip-172-31-36-250 docker-mysql-nodejs-reactjs-app]$
```

2. Modify the configuration files (if necessary) for the application.

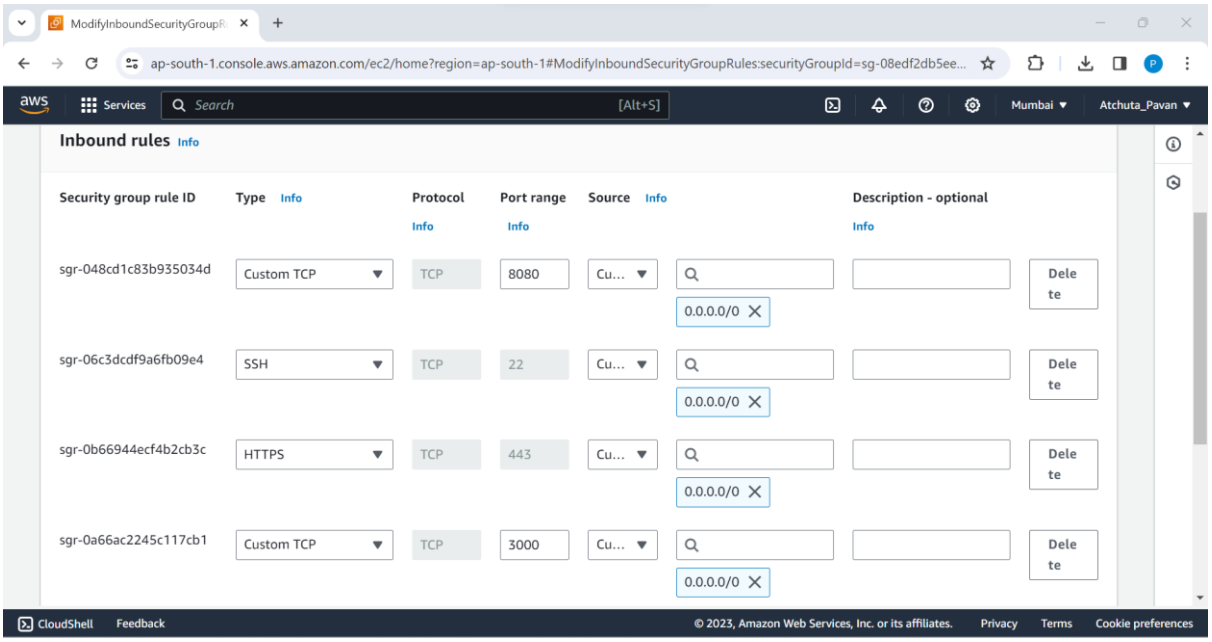
3. Build and start the containers using Docker Compose:

docker-compose up –build

```
[+] Building 259.1s (18/18) FINISHED
 => [docker-mysql-nodejs-reactjs-app-frontend internal] load build definition from Dockerfile                          0.1s
 => => transferring dockerfile: 650B                                                                                   0.1s
 => [docker-mysql-nodejs-reactjs-app-frontend internal] load .dockerignore                                             0.1s
 => => transferring context: 2B                                                                                        0.0s
 => [docker-mysql-nodejs-reactjs-app-api internal] load build definition from Dockerfile                               0.1s
 => => transferring dockerfile: 553B                                                                                   0.1s
 => [docker-mysql-nodejs-reactjs-app-api internal] load .dockerignore                                                  0.1s
 => => transferring context: 2B                                                                                        0.1s
 => [docker-mysql-nodejs-reactjs-app-frontend internal] load metadata for docker.io/library/node:20                    2.3s
 => [docker-mysql-nodejs-reactjs-app-frontend 1/5] FROM docker.io/library/node:20@sha256:445acd9b2ef7e9de665424053bf95  53.8s
 => => resolve docker.io/library/node:20@sha256:445acd9b2ef7e9de665424053bf95652e0b8995ef36500557d48faf29300170a        0.0s
 => => sha256:27e1a8ca91d35598fbae8dee7f1c211f0f93cec529f6804a60e9301c53a604d0 24.05MB / 24.05MB                       2.2s
 => => sha256:d3a767d1d12e57724b9f254794e359f3b04d4d5ad966006e5b5cda78cc382762 64.13MB / 64.13MB                       2.7s
 => => sha256:90e5e7d8b87a34877f61c2b86d053db1c4f440b9054cf49573e3be5d6a674a47 49.58MB / 49.58MB                       1.7s
 => => sha256:445acd9b2ef7e9de665424053bf95652e0b8995ef36500557d48faf29300170a 1.21kB / 1.21kB                        0.0s
 => => sha256:b41c3eccd3c2404464bdbe37f6ee28fc832b2d99b10caeaa38b250a9b9d601e8 2.00kB / 2.00kB                         0.0s
 => => sha256:cda45fab4fcae55c63c3eb4b7f52db026ed4c6d5b977b36be324af800530843a 7.53kB / 7.53kB                         0.0s
 => => extracting sha256:90e5e7d8b87a34877f61c2b86d053db1c4f440b9054cf49573e3be5d6a674a47                              14.5s
 => => sha256:22956530cc64ef2361591684e23e3b8e5bb5910da23197635a2b5b96a34b488d 3.37kB / 3.37kB                         2.7s
 => => sha256:711be5dc50448ab08ccab0b44d65962f36574d341749ab30651b78ec0d4bfd1c 211.07MB / 211.07MB                     8.3s
 => => sha256:53a3765121b18007f5a66660b95a49330a81912398db54d36cb1781639cb8ee5 47.93MB / 47.93MB                       4.7s
 => => sha256:97eaeee3704e79fa0eda21ac5525039beeacb573c32e3c95fb6255a954f8d4cc 2.21MB / 2.21MB                         3.1s
 => => sha256:f52b2977463aa441447e75b516cc29ca2f36c89f2939b44aae8bb0a7f8231e21 450B / 450B                             3.7s
 => => extracting sha256:27e1a8ca91d35598fbae8dee7f1c211f0f93cec529f6804a60e9301c53a604d0                              3.5s
 => => extracting sha256:d3a767d1d12e57724b9f254794e359f3b04d4d5ad966006e5b5cda78cc382762                             12.1s
 => => extracting sha256:711be5dc50448ab08ccab0b44d65962f36574d341749ab30651b78ec0d4bfd1c                             16.3s
 => => extracting sha256:22956530cc64ef2361591684e23e3b8e5bb5910da23197635a2b5b96a34b488d                              0.0s
 => => extracting sha256:53a3765121b18007f5a66660b95a49330a81912398db54d36cb1781639cb8ee5                              3.4s
 => => extracting sha256:97eaeee3704e79fa0eda21ac5525039beeacb573c32e3c95fb6255a954f8d4cc                              0.1s
 => => extracting sha256:f52b2977463aa441447e75b516cc29ca2f36c89f2939b44aae8bb0a7f8231e21                              0.0s
 => [docker-mysql-nodejs-reactjs-app-api internal] load build context                                                  0.1s
```

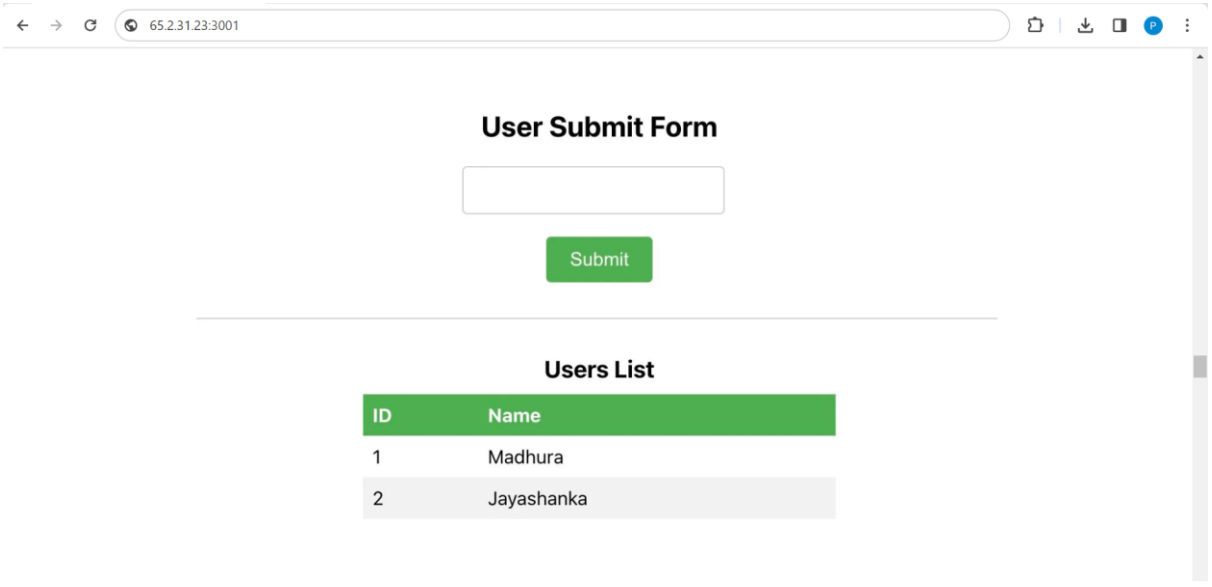
Step-6: Configure Aws Security group

- Go to the AWS Management Console and navigate to the EC2 service.
- Select your running instance and click on “Actions” > “Networking” > “Change Security Groups.”
- Create a new security group or modify the existing one.
- Add a new inbound rule to allow traffic on port 3001 (or the desired port) from anywhere (0.0.0.0/0).



Step-7: Accessing the Application

- Obtain the public IP address of your EC2 instance from the AWS Management Console.
- Open a web browser and enter the following URL: http://<public-ip-address>:3001.



7. Aim: Create a CI/CD pipeline by using Jenkins with AWS CodeBuild & AWS Code Deploy.

Source Code:

Step-1: Creating the aws EC2 instances:

1.Log in to the AWS Management Console.

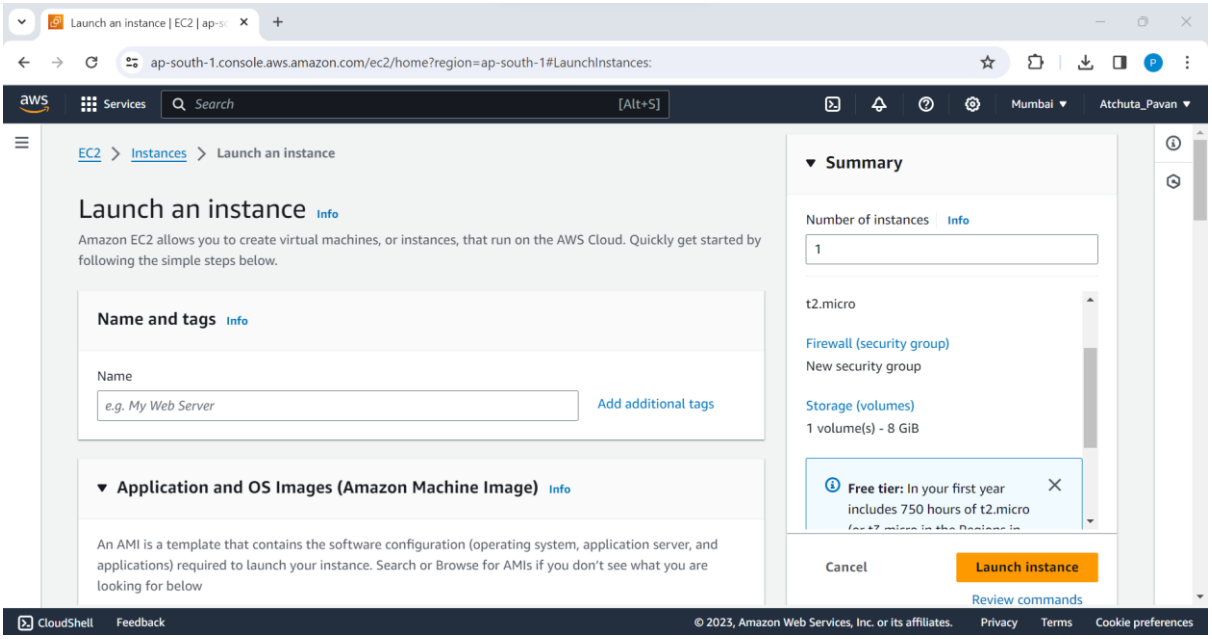
2.Navigate to the EC2 service.

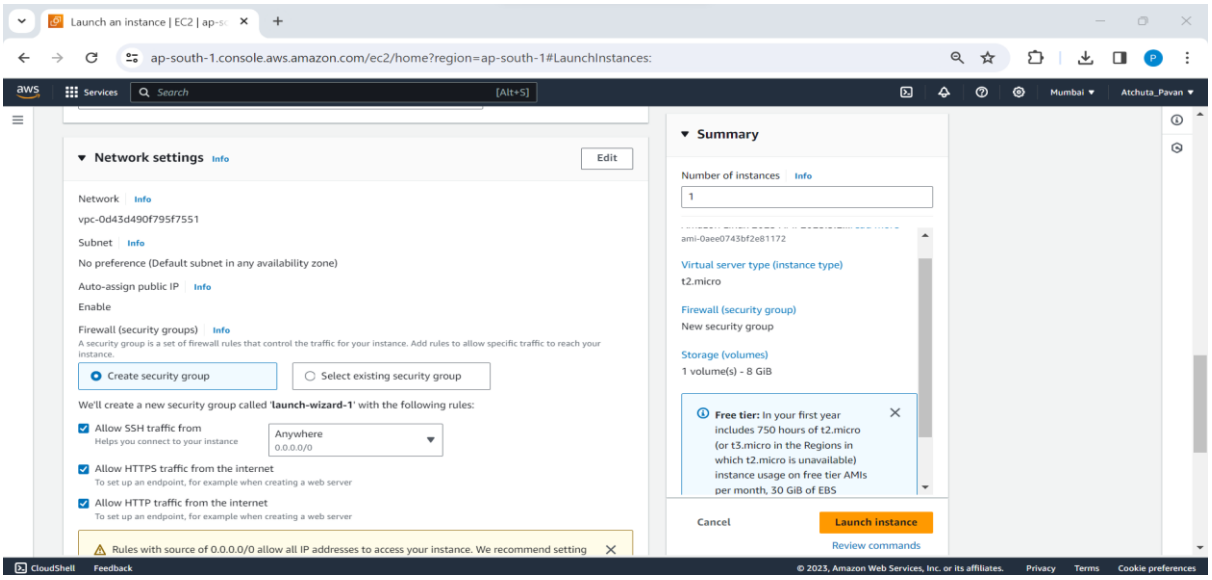
3.Click on “Launch Instance” to create a new EC2 instance.

4.Select the desired instance type, storage, and other configurations.

5.Configure the security group to allow inbound traffic on port 22 for SSH access.

6.Launch the instance and download the private key file (.pem).





Step-2:SSH into EC2 Instance, Install Java and Jenkins

SSH into the EC2 instance. After securely connecting to the instance, the first thing we have to do is to install the Java application onto the instance. Jenkins runs natively runs on the Java application so it has to be properly installed in order for the Jenkins server to be operational.

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

```
[ec2-user@ip-172-31-13-239 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2023-12-17 07:10:03--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/jenkins.r 100%[=====================================>]      85  --.-KB/s    in 0s

2023-12-17 07:10:03 (5.74 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-13-239 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-13-239 ~]$ sudo yum upgrade
Jenkins-stable                                                  264 kB/s |  28 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
```

2. sudo dnf install java-17-amazon-corretto -y

```
[ec2-user@ip-172-31-13-239 ~]$ sudo dnf install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:06 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package                         Arch       Version                  Repository       Size
================================================================================
Installing:
 java-17-amazon-corretto         x86_64     1:17.0.9+8-1.amzn2023.1  amazonlinux     188 k
Installing dependencies:
 alsa-lib                        x86_64     1.2.7.2-1.amzn2023.0.2   amazonlinux     504 k
 cairo                           x86_64     1.17.6-2.amzn2023.0.1    amazonlinux     684 k
 dejavu-sans-fonts               noarch     2.37-16.amzn2023.0.2     amazonlinux     1.3 M
 dejavu-sans-mono-fonts          noarch     2.37-16.amzn2023.0.2     amazonlinux     467 k
 dejavu-serif-fonts              noarch     2.37-16.amzn2023.0.2     amazonlinux     1.0 M
 fontconfig                      x86_64     2.13.94-2.amzn2023.0.2   amazonlinux     273 k
 fonts-filesystem                noarch     1:2.0.5-12.amzn2023.0.2  amazonlinux     9.5 k
 freetype                        x86_64     2.13.0-2.amzn2023.0.1    amazonlinux     422 k
 giflib                          x86_64     5.2.1-9.amzn2023.0.1     amazonlinux      49 k
 google-noto-fonts-common        noarch     20201206-2.amzn2023.0.2  amazonlinux      15 k
 google-noto-sans-vf-fonts       noarch     20201206-2.amzn2023.0.2  amazonlinux     492 k
 graphite2                       x86_64     1.3.14-7.amzn2023.0.2    amazonlinux      97 k
 harfbuzz                        x86_64     7.0.0-2.amzn2023.0.1     amazonlinux     868 k
 java-17-amazon-corretto-headless x86_64    1:17.0.9+8-1.amzn2023.1  amazonlinux      91 M
 javapackages-filesystem         noarch     6.0.0-7.amzn2023.0.6     amazonlinux      12 k
 langpacks-core-font-en          noarch     3.0-21.amzn2023.0.4      amazonlinux      10 k
 libICE                          x86_64     1.0.10-6.amzn2023.0.2    amazonlinux      71 k
 libSM                           x86_64     1.2.3-8.amzn2023.0.2     amazonlinux      42 k
 libX11                          x86_64     1.7.2-3.amzn2023.0.4     amazonlinux     657 k
 libX11-common                   noarch     1.7.2-3.amzn2023.0.4     amazonlinux     152 k
 libXau                          x86_64     1.0.9-6.amzn2023.0.2     amazonlinux      31 k
 libXext                         x86_64     1.3.4-6.amzn2023.0.2     amazonlinux      41 k
```

3. sudo yum install jenkins -y

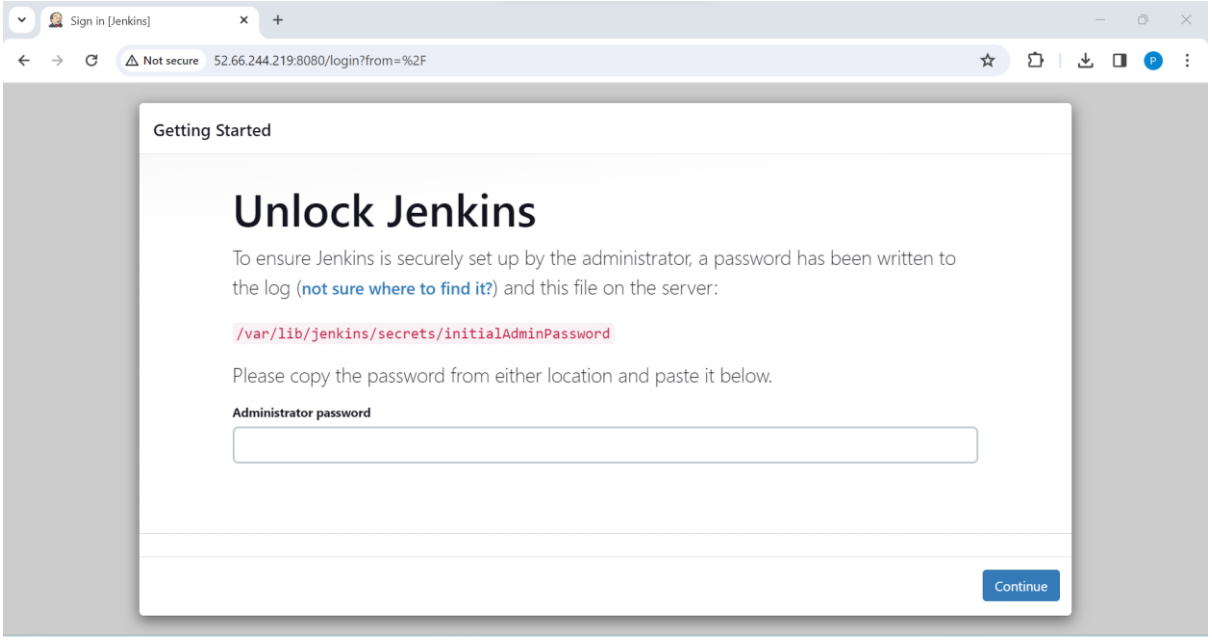
```
[ec2-user@ip-172-31-13-239 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:00:22 ago on Sun Dec 17 07:10:17 2023.
Dependencies resolved.
================================================================================
 Package              Architecture          Version                  Repository              Size
================================================================================
Installing:
 jenkins              noarch                2.426.2-1.1              jenkins                 85 M

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

Total download size: 85 M
Installed size: 85 M
Downloading Packages:
jenkins-2.426.2-1.1.noarch.rpm                                  7.7 MB/s |  85 MB     00:11
--------------------------------------------------------------------------------
Total                                                           7.7 MB/s |  85 MB     00:11
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Installing       : jenkins-2.426.2-1.1.noarch                             1/1
  Running scriptlet: jenkins-2.426.2-1.1.noarch                             1/1
  Verifying        : jenkins-2.426.2-1.1.noarch                             1/1

Installed:
  jenkins-2.426.2-1.1.noarch
```

4. sudo systemctl enable jenkins
   sudo systemctl start jenkins

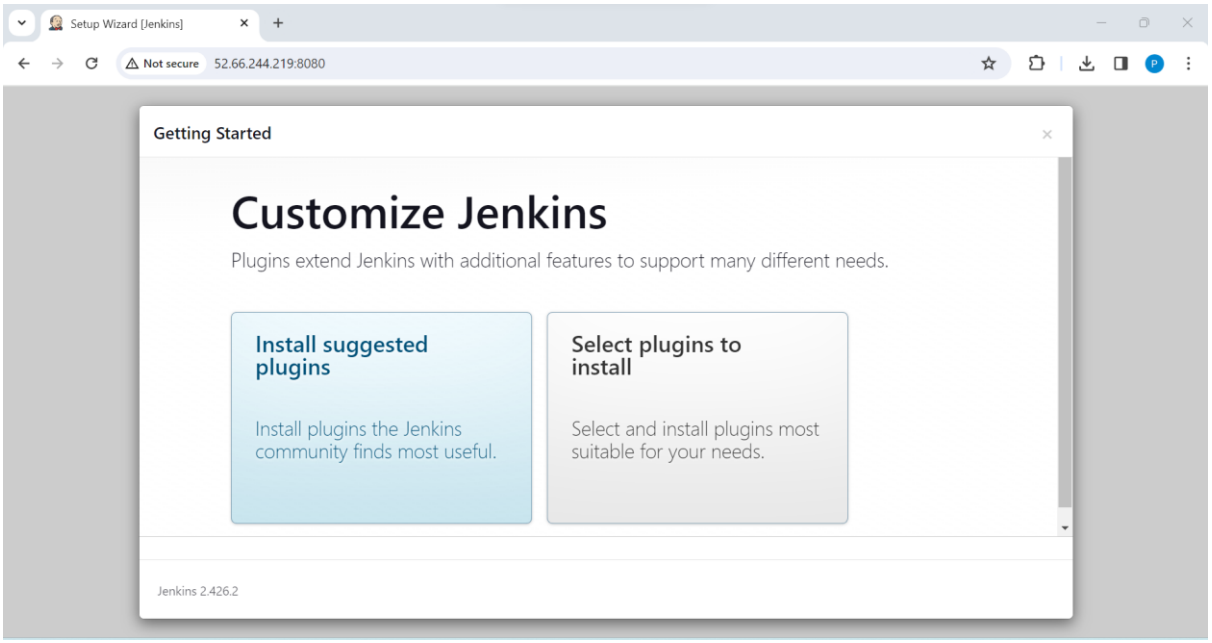
```
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jen
kins.service.
[ec2-user@ip-172-31-13-239 ~]$ sudo systemctl start jenkins
```

Step-3: Setting Up Jenkins Server

Now that the Jenkins server has been officially started on the instance, head over to the AWS Management console, grab the public IP address of the instance and paste that into your browser address bar followed by :8080




```
[ec2-user@ip-172-31-13-239 ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
baaef975c09e468a977274b59ed26a8e
[ec2-user@ip-172-31-13-239 ~]$
```



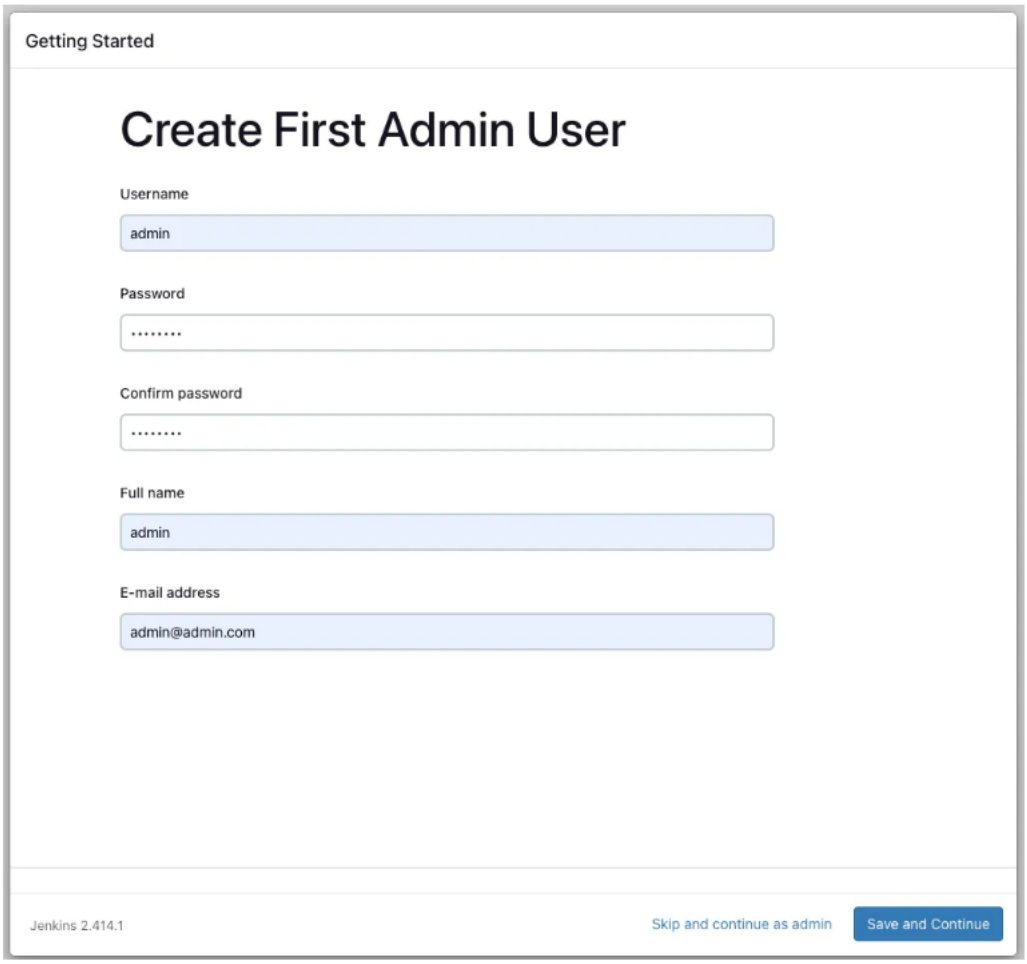
Setup an admin user. For this project we will get a dummy administrator user. In a real production you will use different credentials. Save and continue.

Username=admin

Password=Password

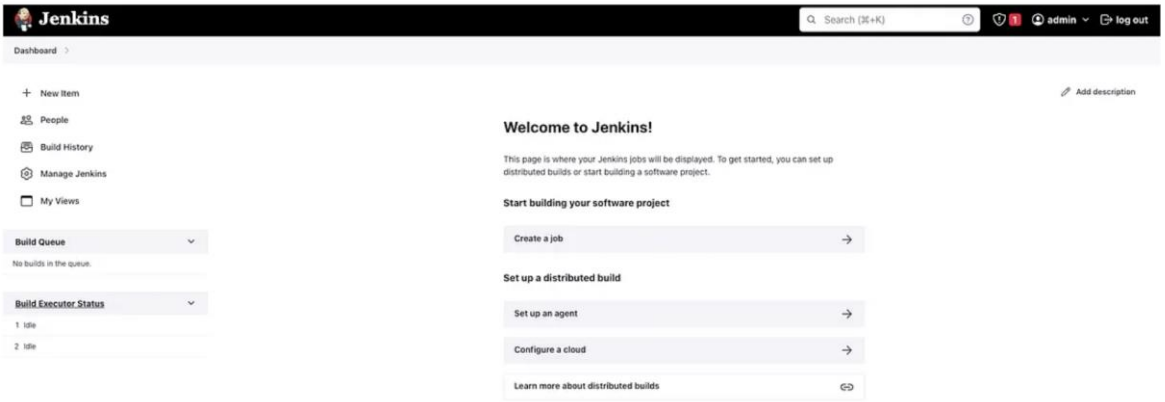
Full name= admin

Email=admin@admin.com

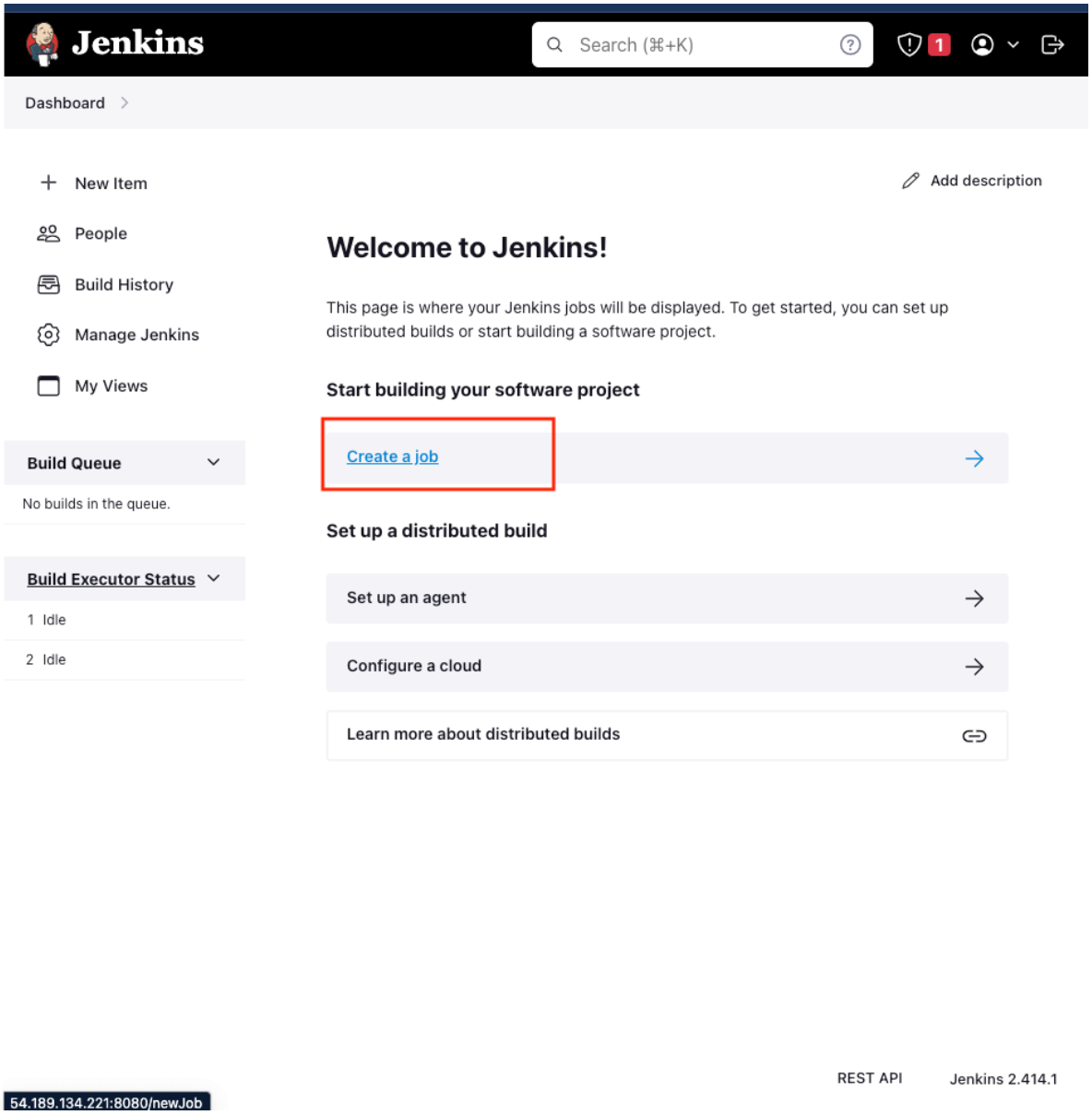


Step 4: We are Live and Jenkins Ready! Time to Build

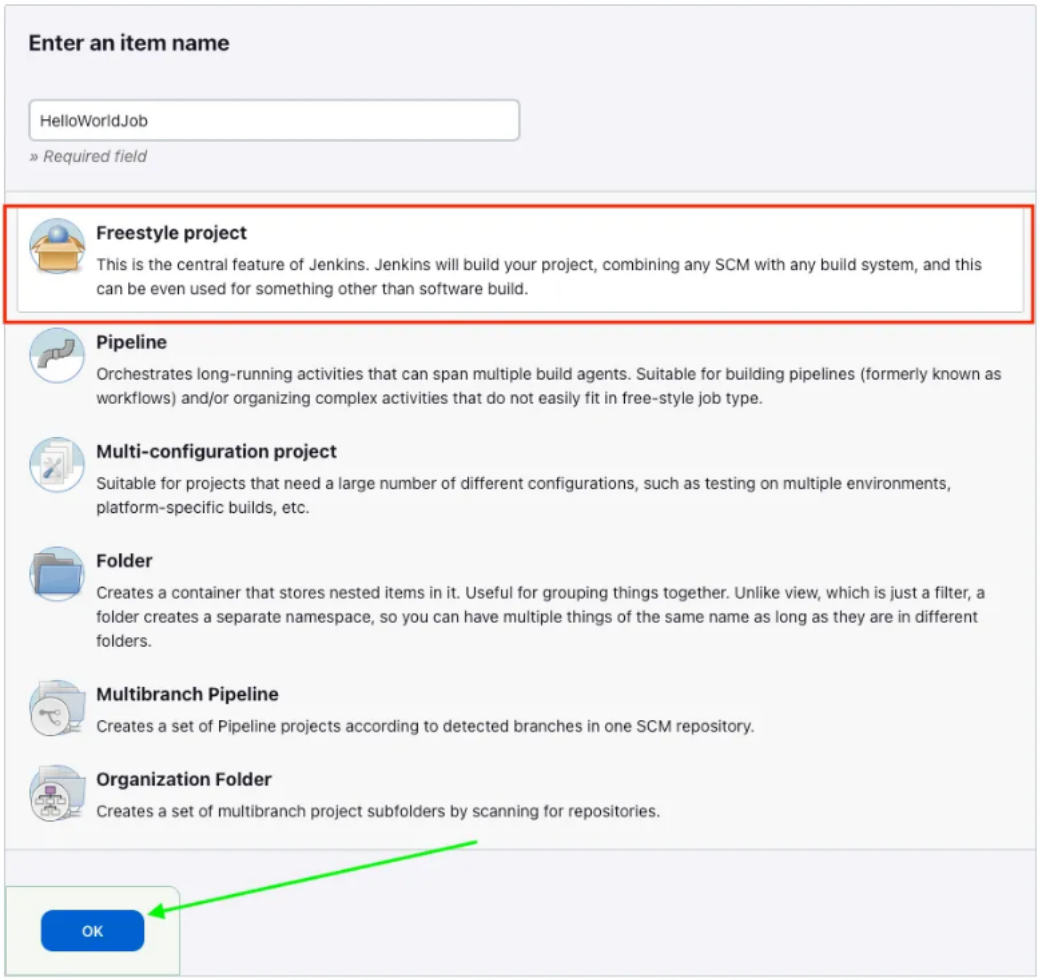
If you see the “Welcome to Jenkins” page, then you’ve successfully setup Jenkins and are ready to go to the next stage of the project, which is setting up our first Jenkins job!



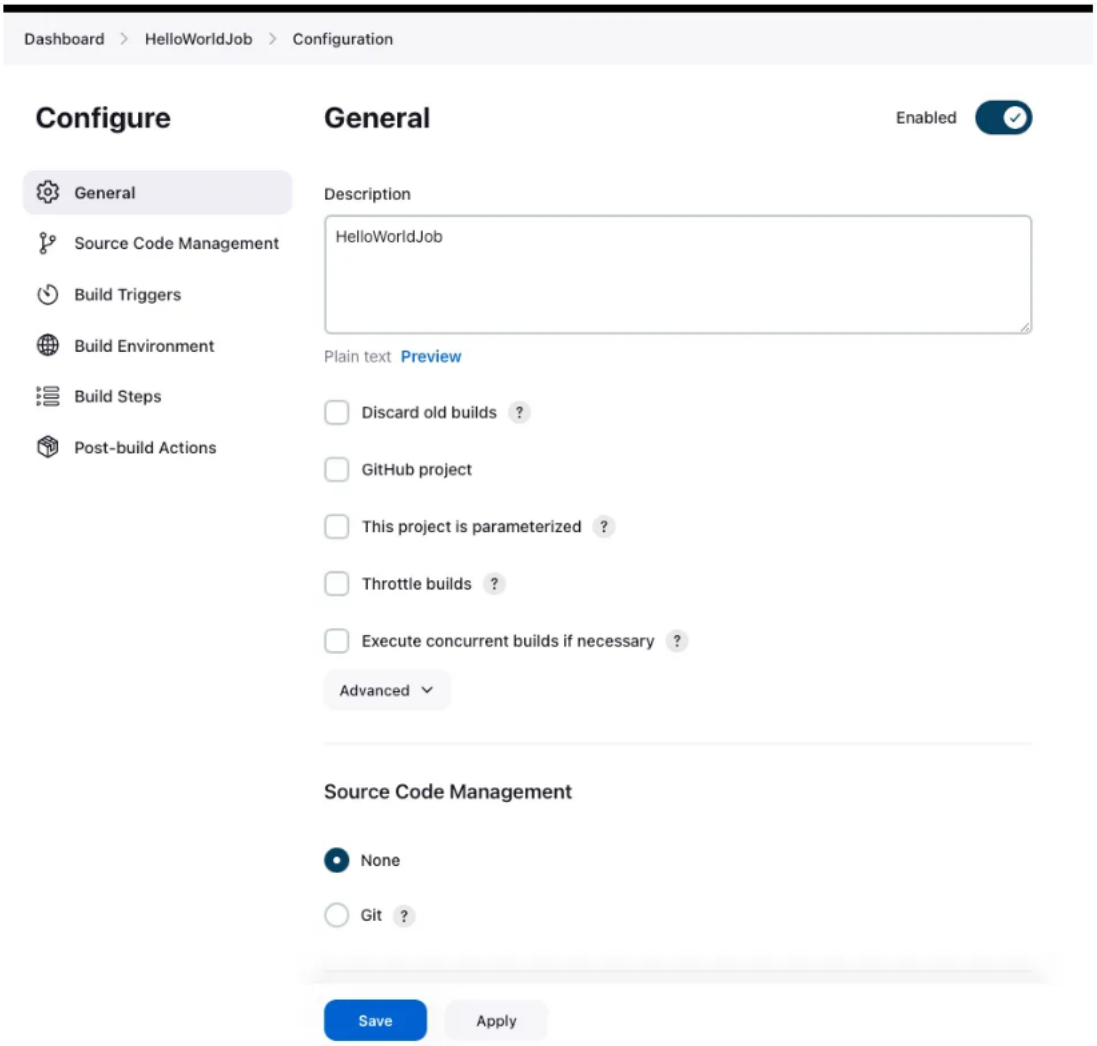
Creating our first jenkin job



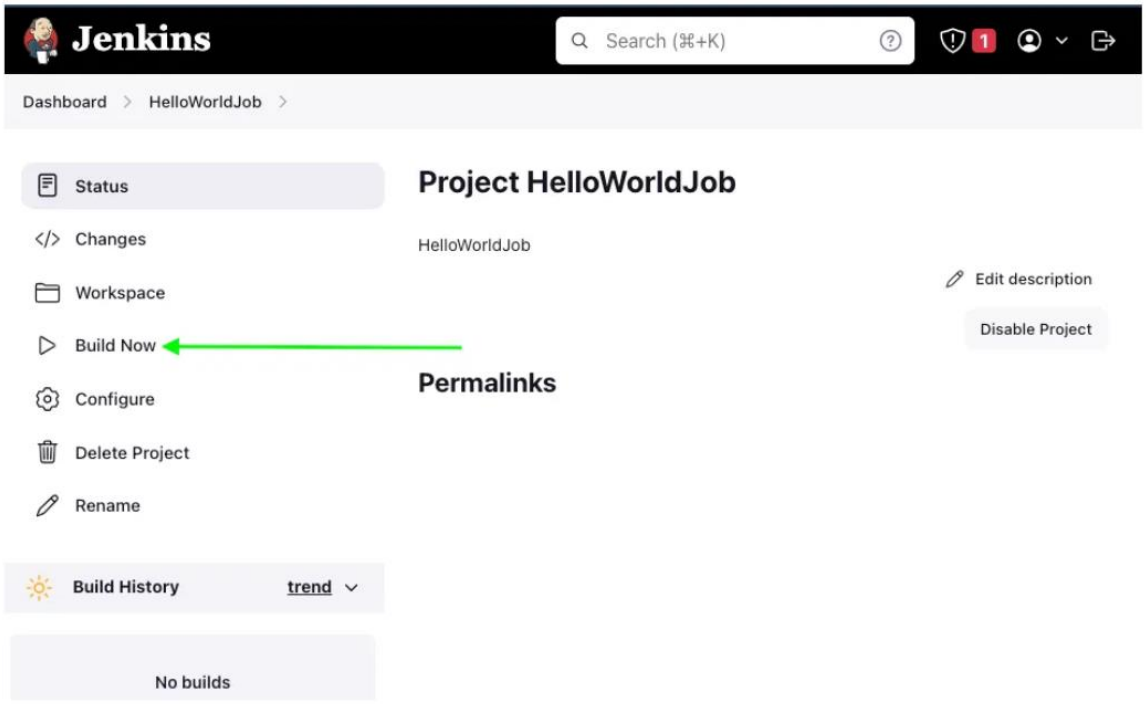
For our first job we are going to give it the name "**HelloWorldJob**" and select the option for "**Freestyle project**". Click ok to move to the next step.

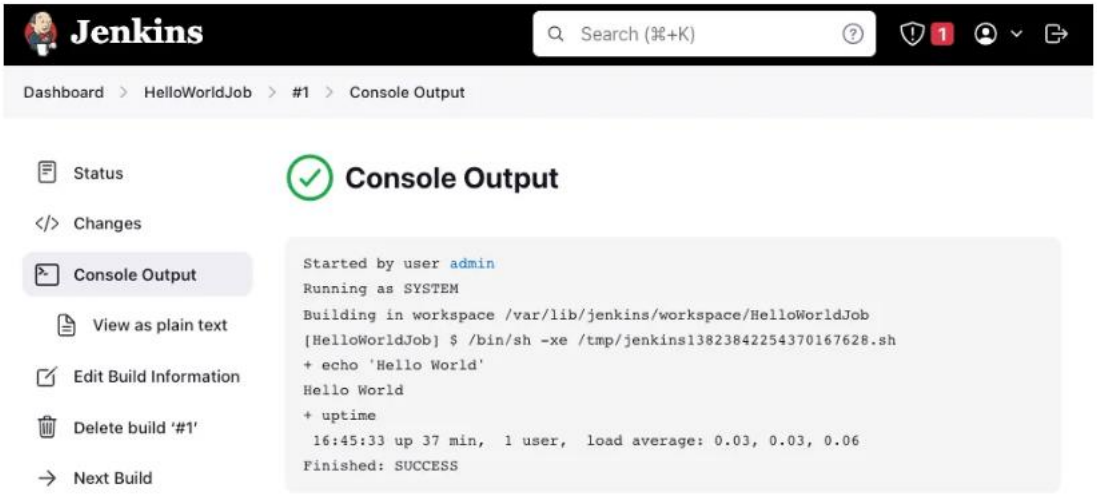


Step-5: Under the general configuration settings
In the description box, enter the text “HelloWorldJob”
Under the “Source Code Management” tab select “None” option

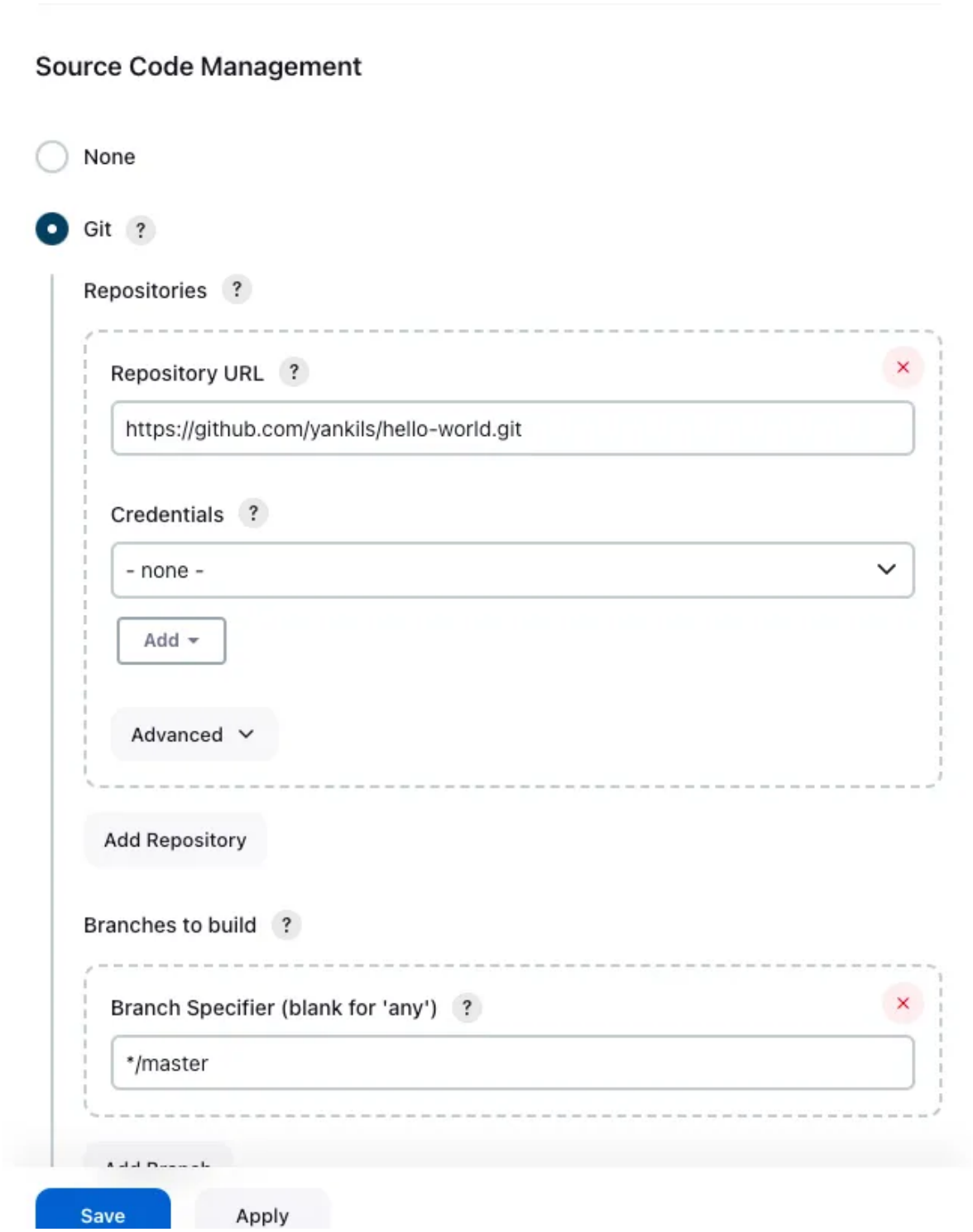


- Navigate back to the Dashboard and you will see our 1st job in queue ready for the build stage.
- Click on “Build Now”
- After a few seconds you will see the the build output under the “build history”
- The console ouput should display a green checkmark indicating that the build was SUCCESSFUL.
- You can see that the echo “Hello World” and uptime commands were properly executed!



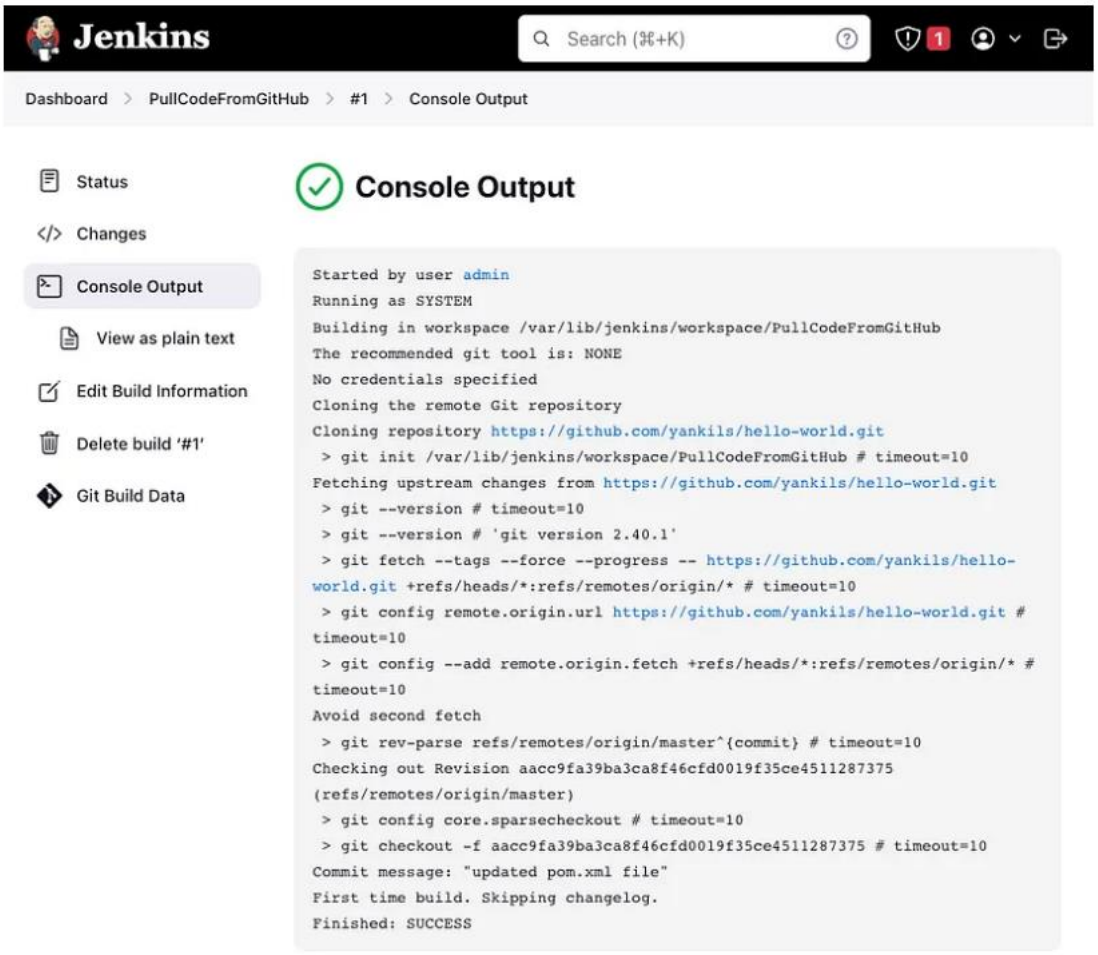


Step-6: Integrating git



Under Dashboard, you will see a new job for Git.

Click on “Build Now” if the build output shows a green checkmark indicator, that means our build was a SUCCESS!



Step 7: Installing and Integrating Maven to Jenkins

Maven automates the process of building and managing code in both software development as well as in DevOps. Maven manages the build, testing, and packaging of code, making it easy for DevOps teams to produce consistent, reliable, and repeatable builds.

```
[root@ip-172-31-23-228 ~]# cd /opt
[root@ip-172-31-23-228 opt]# wget https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.tar.gz
--2023-08-28 17:18:40--  https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9336368 (8.9M) [application/x-gzip]
Saving to: 'apache-maven-3.9.4-bin.tar.gz'

100%[=====================================================>] 9,336,368   --.-K/s   in 0.1s

2023-08-28 17:18:40 (81.3 MB/s) - 'apache-maven-3.9.4-bin.tar.gz' saved [9336368/9336368]
```

After installation, verify that Maven was indeed installed by list the contains of the directory: If you see the maven dependency files listed then you know it was successfully installed.

```
[root@ip-172-31-23-228 opt]# ll
total 9120
-rw-r--r-- 1 root root 9336368 Aug  3 07:56 apache-maven-3.9.4-bin.tar.gz
drwxr-xr-x 4 root root      33 Aug 22 18:26 aws
drwxr-xr-x 2 root root       6 Aug 16  2018 rh
[root@ip-172-31-23-228 opt]# tar -xvzf apache-maven-3.9.4-bin.tar.gz
apache-maven-3.9.4/README.txt
apache-maven-3.9.4/LICENSE
apache-maven-3.9.4/NOTICE
apache-maven-3.9.4/lib/
apache-maven-3.9.4/lib/aopalliance.license
apache-maven-3.9.4/lib/commons-cli.license
apache-maven-3.9.4/lib/commons-codec.license
apache-maven-3.9.4/lib/commons-lang3.license
apache-maven-3.9.4/lib/failureaccess.license
apache-maven-3.9.4/lib/guava.license
apache-maven-3.9.4/lib/guice.license
apache-maven-3.9.4/lib/httpclient.license
apache-maven-3.9.4/lib/httpcore.license
apache-maven-3.9.4/lib/jansi.license
apache-maven-3.9.4/lib/javax.annotation-api.license
apache-maven-3.9.4/lib/javax.inject.license
apache-maven-3.9.4/lib/jcl-over-slf4j.license
apache-maven-3.9.4/lib/org.eclipse.sisu.inject.license
apache-maven-3.9.4/lib/org.eclipse.sisu.plexus.license
apache-maven-3.9.4/lib/plexus-cipher.license
apache-maven-3.9.4/lib/plexus-component-annotations.license
apache-maven-3.9.4/lib/plexus-interpolation.license
apache-maven-3.9.4/lib/plexus-sec-dispatcher.license
apache-maven-3.9.4/lib/plexus-utils.license
apache-maven-3.9.4/lib/slf4j-api.license
apache-maven-3.9.4/boot/
apache-maven-3.9.4/boot/plexus-classworlds.license
apache-maven-3.9.4/lib/jansi-native/
apache-maven-3.9.4/lib/jansi-native/Windows/
apache-maven-3.9.4/lib/jansi-native/Windows/x86/
apache-maven-3.9.4/lib/jansi-native/Windows/x86_64/
apache-maven-3.9.4/lib/jansi-native/Windows/x86/jansi.dll
apache-maven-3.9.4/lib/jansi-native/Windows/x86_64/jansi.dll
apache-maven-3.9.4/bin/m2.conf
apache-maven-3.9.4/bin/mvn.cmd
apache-maven-3.9.4/bin/mvnDebug.cmd
apache-maven-3.9.4/bin/mvn
apache-maven-3.9.4/bin/mvnDebug
```

You can choose to keep the default maven file name that is highlighted in blue or change the name of the file. I'm going to change the name to just "maven" to keep things simple. To change the name use the "mv" command.

```
[root@ip-172-31-23-228 opt]# mv apache-maven-3.9.4 maven
[root@ip-172-31-23-228 opt]# ll
total 9120
-rw-r--r-- 1 root root 9336368 Aug  3 07:56 apache-maven-3.9.4-bin.tar.gz
drwxr-xr-x 4 root root      33 Aug 22 18:26 aws
drwxr-xr-x 6 root root      99 Aug 28 17:20 maven
drwxr-xr-x 2 root root       6 Aug 16  2018 rh
[root@ip-172-31-23-228 opt]# cd maven
[root@ip-172-31-23-228 maven]# ll
total 36
drwxr-xr-x 2 root root    97 Aug 28 17:20 bin
drwxr-xr-x 2 root root    76 Aug 28 17:20 boot
drwxr-xr-x 3 root root    63 Jul 26 09:37 conf
drwxr-xr-x 4 root root  4096 Aug 28 17:20 lib
-rw-r--r-- 1 root root 18945 Jul 26 09:37 LICENSE
-rw-r--r-- 1 root root  5034 Jul 26 09:37 NOTICE
-rw-r--r-- 1 root root  2533 Jul 26 09:37 README.txt
```

Execute Maven server

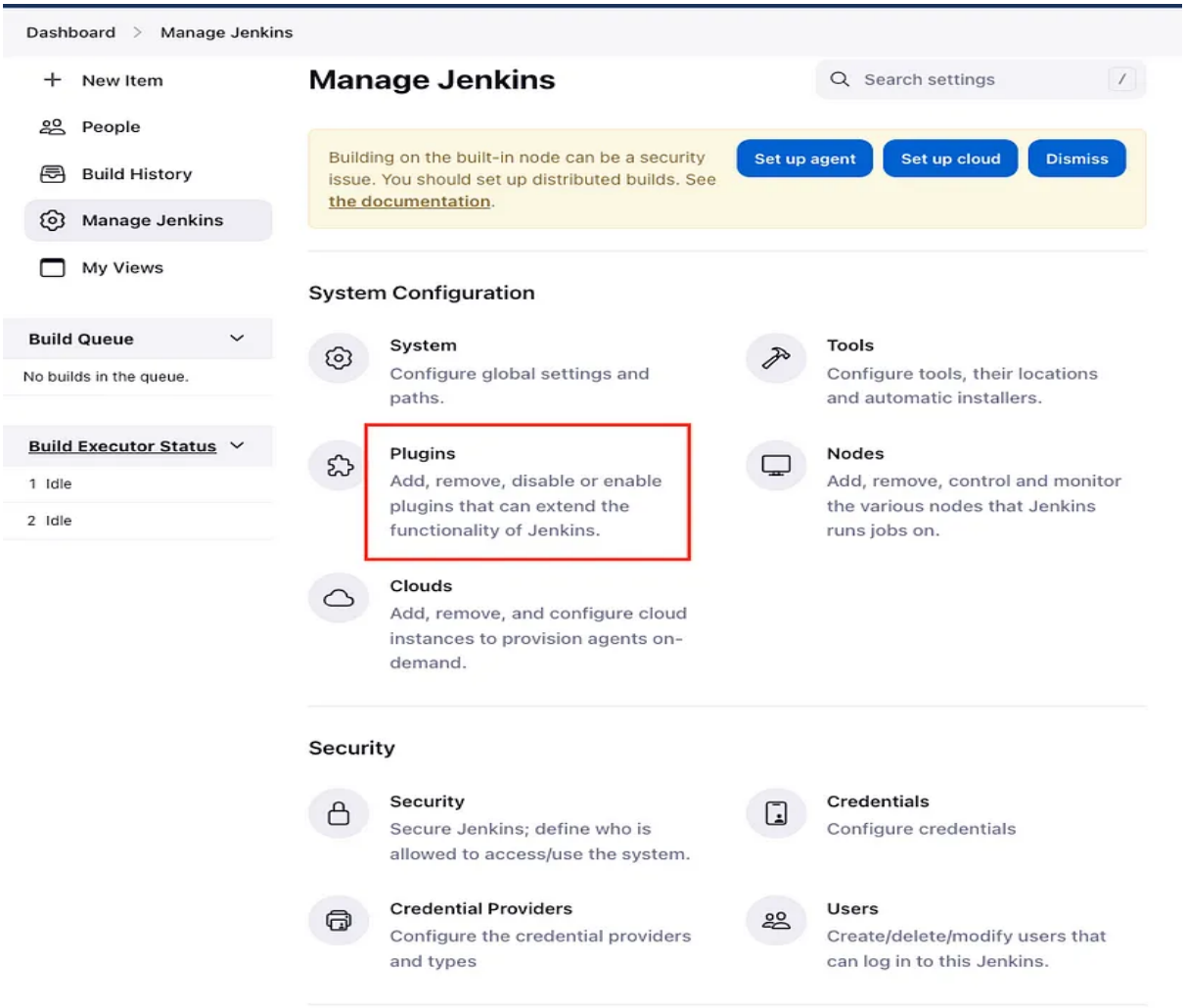
Check maven version

mvn – v

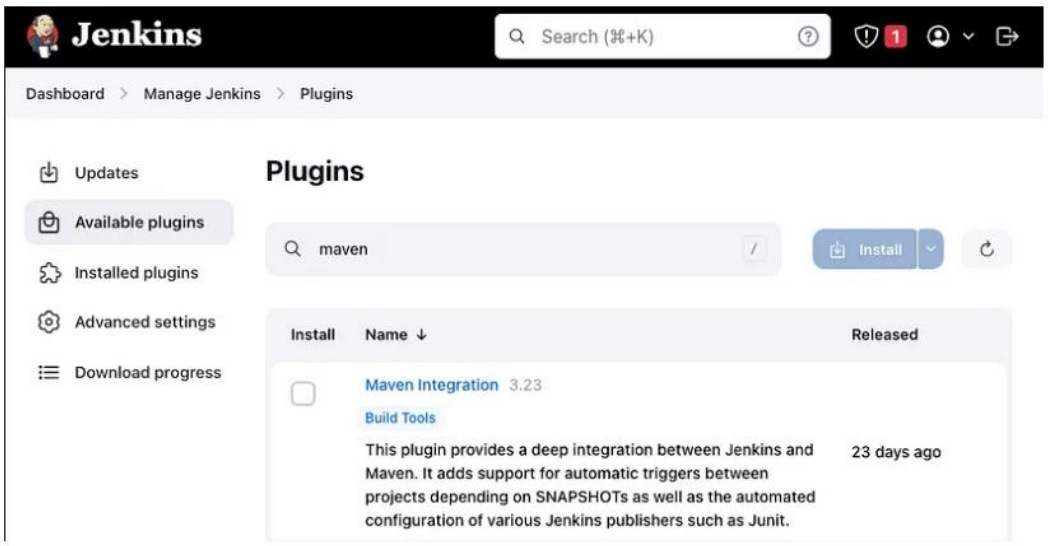
```
...c2-54-189-134-221.us-west-2.compute.amazonaws.com   ...4-189-134-221.us-west-2.compute.amazonaws.com
[root@ip-172-31-23-228 ~]# mvn -v
Apache Maven 3.9.4 (dfbb324ad4a7c8fb0bf182e6d91b0ae20e3d2dd9)
Maven home: /opt/maven
Java version: 11.0.20, vendor: Red Hat, Inc., runtime: /usr/lib/jvm/java-11-openjdk-11.0.20.0.8
-1.amzn2.0.1.x86_64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.10.186-179.751.amzn2.x86_64", arch: "amd64", family: "unix"
[root@ip-172-31-23-228 ~]#
```

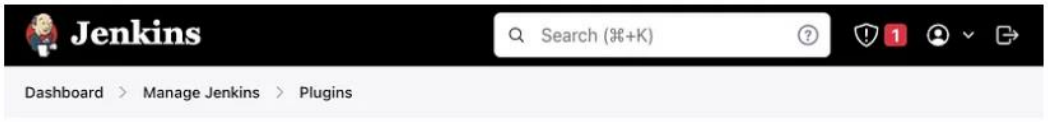
Step-8: Maven Plugin for Jenkins:

- With Maven server installed, it is time to setup the Maven plugin in the Jenkins server to complete the integration
- Navigate back to the Jenkins Dashboard, select “Manage Jenkins” option, and select “Plugins”.



- Type "**Maven**" and select Maven Integration. Click on install the plugin.





Under Mange Jenkins, select the "Global Configuration" tab. Since we successfully installed JDK and Maven, you will see the options referencing JDK and Maven.
Name-java-11
JAVA_HOME= /usr/lib/jvm/java-11-openjdk-11.0.20.0.8–1amzn2.0.1x86_64 (Java home directory)
Deselect the "install automatically" option. We already have Java installed.
Step-9: Navigate to the Jenkins dashboard, select "create item". We are going to test our CI/CD build by setting our first Maven project.

- Type "FirstMavenProject" in the name box
- Select "Maven project". Click ok.
- Enter a description for the project build.
- Select "Git" under the Source Code Management. Copy and paste the folllowing git repo URL into the box: https://github.com/GeorgeBaidooJr9/hello-world
- Under Build "Goal and Options" type "clean install". This will result in clean copy of the git repository onto Maven.
- Apply and save

Let's OFFICIALLY TEST THE BUILD!

- Select "Build" under configure. This build will take a few minutes since there is lots of data stored in the repo.

- After a few minutes you will see details of the build.