




CI/CD WITH JENKINS



Presented By : 김대현
010-5251-3547
anyware1001@gmail.com

CONTENTS

- ❑ Introduction to Jenkins
- ❑ History
- ❑ Architecture
- ❑ Features
- ❑ Plugins
- ❑ Jenkins Pipelines and Jenkinsfile
- ❑ Jenkins vs other CI/CD tools
- ❑ Advantages
- ❑ Installing and Configuring
- ❑ Demo - CI/CD of React Application



Open Source Continuous Integration Server



INTRODUCTION TO JENKINS

- ❑ Jenkins is an open source continuous integration/continuous delivery and deployment (CI/CD) automation software DevOps tools written in the Java programming language.
- ❑ It is used to implement CI/CD workflows, called pipelines.
- ❑ Jenkins is a self-contained , open source automation that can be used to automate all sorts of tasks related to building , testing and deploying software.
- ❑ It uses the plugins for building and testing the project code continuously .
- ❑ Jenkins can be installed through native system packages, Docker or even run standalone by an machine with a Java Runtime Environment (JRE) installed.

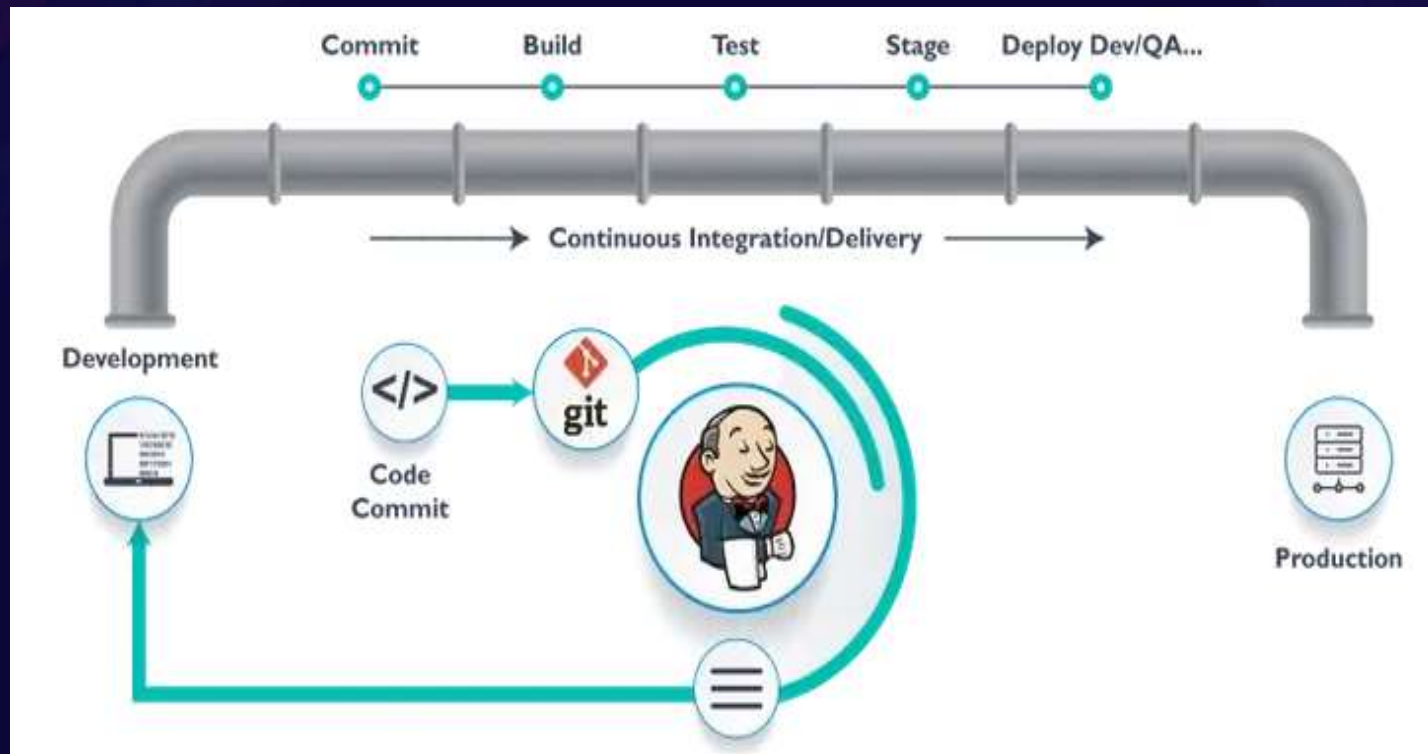


HISTORY

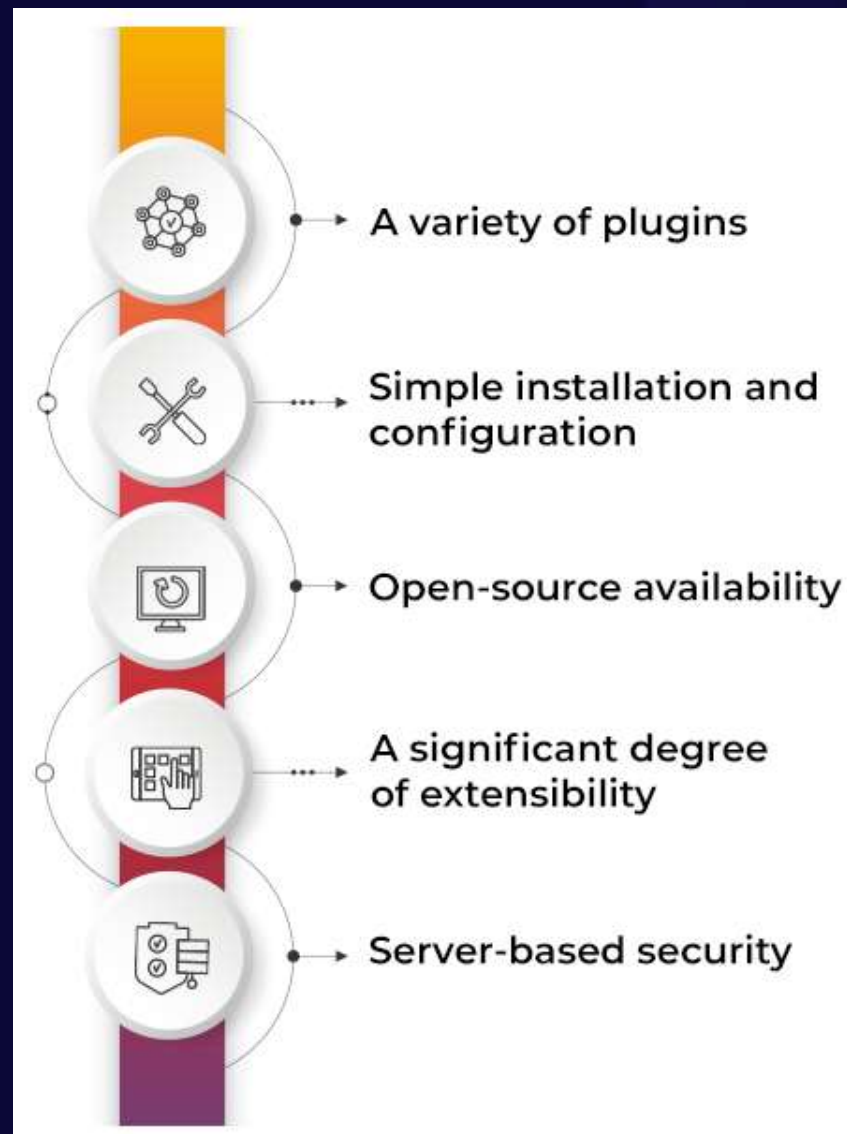
- ❑ Jenkins is a fork of a project called Hudson, which was trademarked by Oracle.
- ❑ Kohsuke first developed Hudson 2004 while working at Sun Microsystems. When Oracle acquired Sun Microsystems in 2010, there was a dispute between Oracle and the Hudson community with the respect to infrastructure used.
- ❑ Kohsuke wanted to create a method to perform continuous integration, the idea was to test the code before committing to avoid the breaking builds.
- ❑ On Jan 11, 2011, a call for votes made to change the project name from Hudson to Jenkins.
- ❑ On Jan 29, 2011 , creating the first Jenkins project.



ARCHITECTURE







FEATURES OF JENKINS




PLUGINS

- ❑ A plugin is an enhancement to the Jenkins systems. They help extend Jenkins capabilities and integrated Jenkins with other software.
- ❑ Plugins can be downloaded from the online Jenkins plugin repository and loaded using the Jenkins Web UI or CLI.
- ❑ Currently, the Jenkins community claims over 1500+ plugins available for a wide range of uses.

Icons	Description
	Kubernetes plugin is great for automating build agents on a Kubernetes cluster
	Git plugin allows jobs to connect to remote repositories
	EC2 plugin is used for Jenkins to automatically provision AWS
	JUnit plugin provides graphical visualizations





PLUGINS MANAGER


 **Jenkins**


Search (CTRL+K) ? Arjit Chauhan log out


Dashboard > Manage Jenkins > Plugin Manager

 Updates

 Available plugins





 Installed plugins

 Advanced settings

 Download progress

Plugins

Search installed plugins

Name ↓	Enabled
Ant Plugin 481.v7b_09e538fccca Adds Apache Ant support to Jenkins Report an issue with this plugin	 
Apache HttpComponents Client 4.x API Plugin 4.5.13-138.v4e7d9a_7b_a_e61 Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins. Report an issue with this plugin	 
<div>This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.</div>	
Authentication Tokens API Plugin 1.4	

JENKINS PIPELINE AND JENKINSFILE

- ❑ The role of Jenkins in DevOps is primarily due to the pipeline-as-code concept followed by Jenkins.
- ❑ The continuous pipeline is an automated process for obtaining software from version control to users and customers.
- ❑ Jenkins Pipeline is defined using a text file called the Jenkinsfile.
- ❑ The pipeline implements as code using Groovy Domain-specific language through an editor or the configuration page on Jenkins instance.
- ❑ Jenkinsfile give leverage to the developer to easily access or edit or check the code anytime.



JENKINSFILE

- ❑ Two types of syntax using which we can define a Jenkinsfile:

- > Declarative Pipeline syntax
- > Scripted Pipeline syntax

- ❑ Declarative Pipeline syntax ->

The declarative syntax is a new feature that used code for the pipeline. It provides a limited pre-defined structure. Thereby, it offers an easy and simple continuous delivery pipeline.

(We will create Jenkinsfile in SCM)

- ❑ Scripted Pipeline syntax ->

The scripted pipeline syntax is the old traditional way to write the Jenkinsfile on Jenkins web UI. Moreover it follows the groovy syntax and helps to develop a complex pipeline as code.

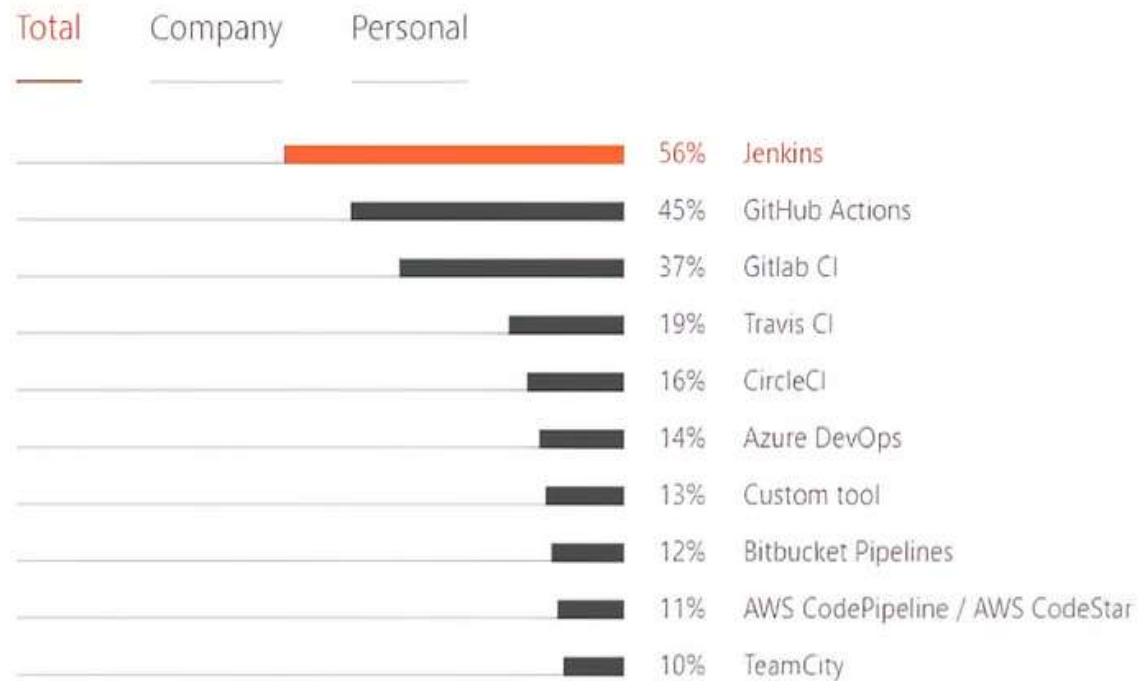
(It is written in Jenkins dashboard)



JENKINS VS OTHER CI/CD TOOLS

❑ Jenkins : the most-used CI/CD solutions.

- No expenses required
- Limitless integrations
- Active community



ADVANTAGES

- ❑ It is open source and it is user-friendly, easy to install and does not required additional installations or components. It is free of cost.
- ❑ Easily Configurable: Jenkins can be easily modified and extended. It deploys code instantly, generates test reports.
- ❑ Platform independent: Jenkins is available for all the platform and different operating system.
- ❑ Rich Plugins ecosystem: The extensive pool of plugins makes Jenkins flexible and allow building, deploying and automating across various platform.
- ❑ Issue are detected and resolve almost right way which keeps the software in a state where it can be released at any time safely.
- ❑ Most of the integration work in automated. Hence fewer integration issues. This save both time and money over the lifespan of a project.



INSTALLING AND CONFIGURING JENKINS

❑ Prerequisites of Jenkins Installation

- Ubuntu server with 18.04, 20.04 or 22.04
- 256 MB of RAM
- 1 GB of drive space for solo use. However, no less than 10 GB is recommended if jenkins run inside a Docker container
- 4GB+ of RAM
- 50GB+ of drive space
- Oracle JDK 8 or 11
- Jenkins by default runs on port 8080



➤ STEP -1 Installing Java Development kit

- Sudo apt-get install openjdk-11-jdk

➤ Step -2 Installing Jenkins

- `curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee \`
`/usr/share/keyrings/jenkins-keyring.asc > /dev/null`
- `echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \ https://pkg.jenkins.io/debian-stable`
`binary/ | sudo tee \ /etc/apt/sources.list.d/jenkins.list > /dev/null`
- `sudo apt-get update`
- `sudo apt-get install jenkins`
- `sudo systemctl start jenkins.service`



➤ sudo systemctl status jenkins

```
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-094f081f574529f5c&osUser=ubuntu&ss...

aws Services Search [Alt+S] N. Virginia Karan Singh

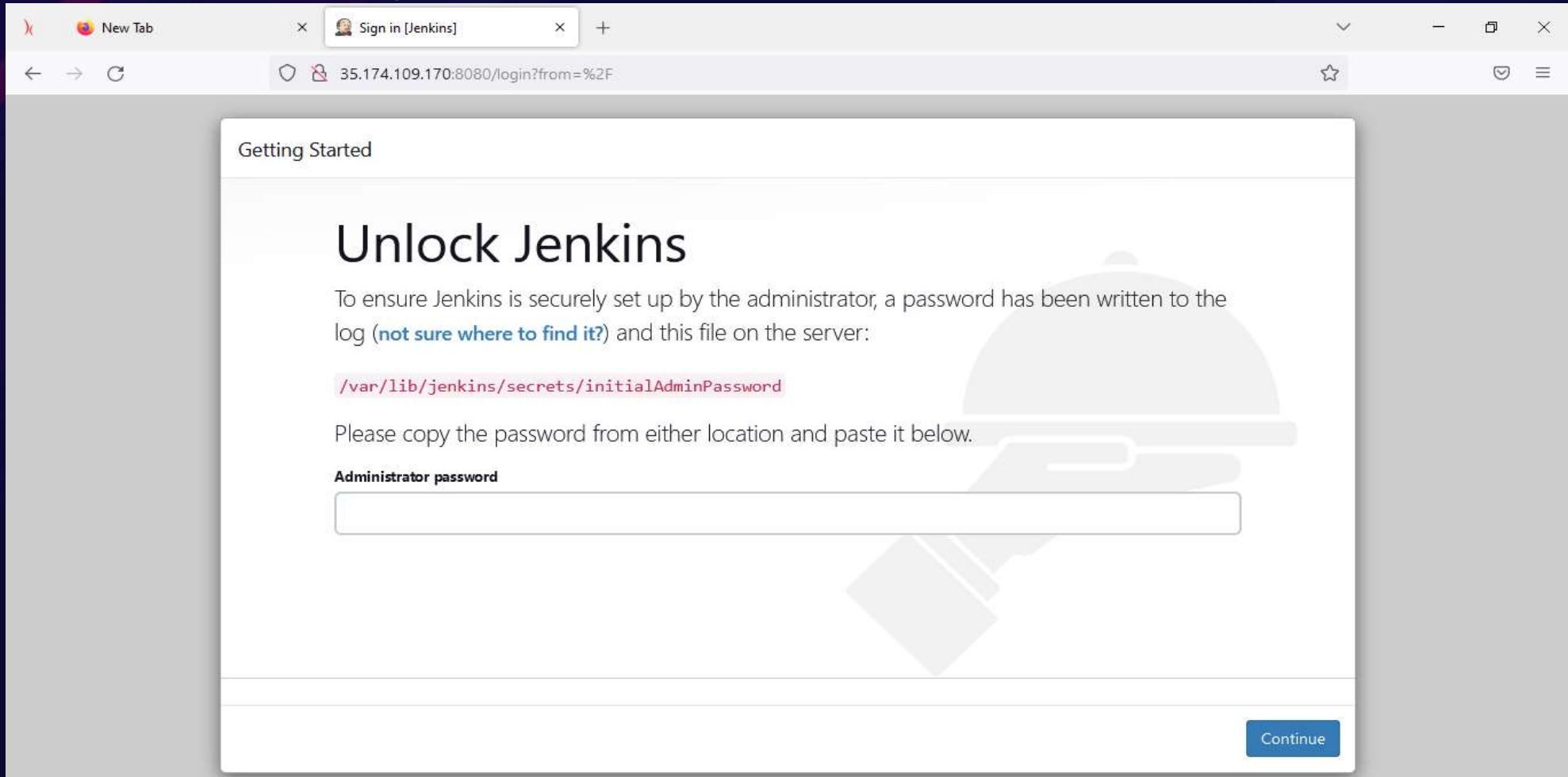
CloudFormation CodePipeline EC2 CodeBuild

Unpacking jenkins (2.375.1) ...
Setting up net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Setting up jenkins (2.375.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.19) ...
root@ip-172-31-81-13:/home/ubuntu# sudo systemctl start jenkins.service
root@ip-172-31-81-13:/home/ubuntu# sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2022-12-16 09:53:07 UTC; 1min 27s ago
     Main PID: 5838 (java)
       Tasks: 36 (limit: 1143)
      Memory: 312.3M
     CGroup: /system.slice/jenkins.service
            └─5838 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

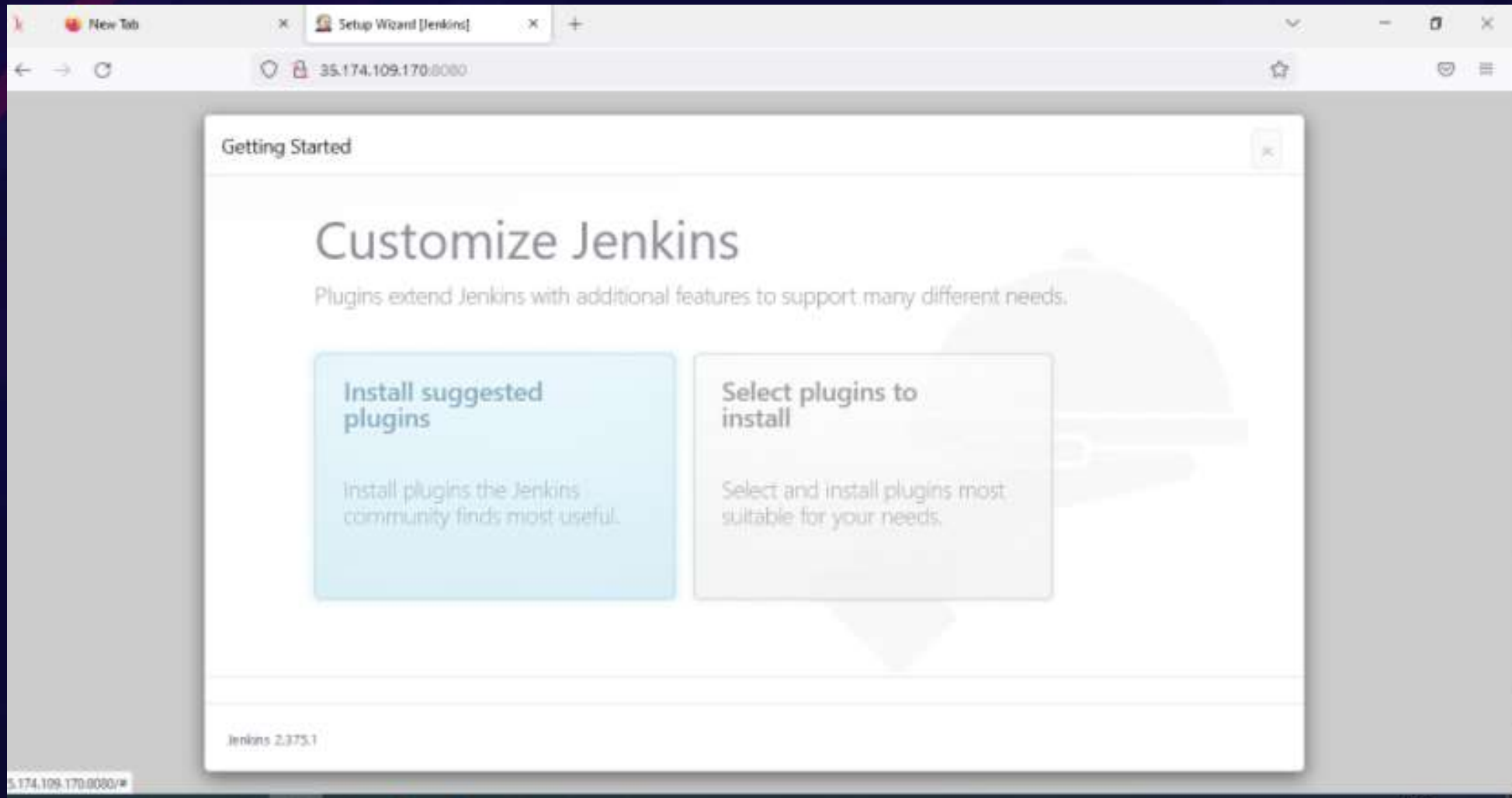
Dec 16 09:52:35 ip-172-31-81-13 jenkins[5838]: 4a35067e346d402480cb721cefb7c72b7
Dec 16 09:52:35 ip-172-31-81-13 jenkins[5838]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Dec 16 09:52:35 ip-172-31-81-13 jenkins[5838]: *****
Dec 16 09:52:35 ip-172-31-81-13 jenkins[5838]: *****
Dec 16 09:52:35 ip-172-31-81-13 jenkins[5838]: *****
Dec 16 09:53:07 ip-172-31-81-13 jenkins[5838]: 2022-12-16 09:53:07.474+0000 [id=29] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
Dec 16 09:53:07 ip-172-31-81-13 jenkins[5838]: 2022-12-16 09:53:07.501+0000 [id=22] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and runni>
Dec 16 09:53:07 ip-172-31-81-13 systemd[1]: Started Jenkins Continuous Integration Server.
Dec 16 09:53:07 ip-172-31-81-13 jenkins[5838]: 2022-12-16 09:53:07.621+0000 [id=44] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data >
Dec 16 09:53:07 ip-172-31-81-13 jenkins[5838]: 2022-12-16 09:53:07.622+0000 [id=44] INFO hudson.util.Retrier#start: Performed the action check updates ser>
lines 1-19/19 (END)

i-094f081f574529f5c (jenkinsupload)
Public IPs: 35.174.109.170 Private IPs: 172.31.81.13
```

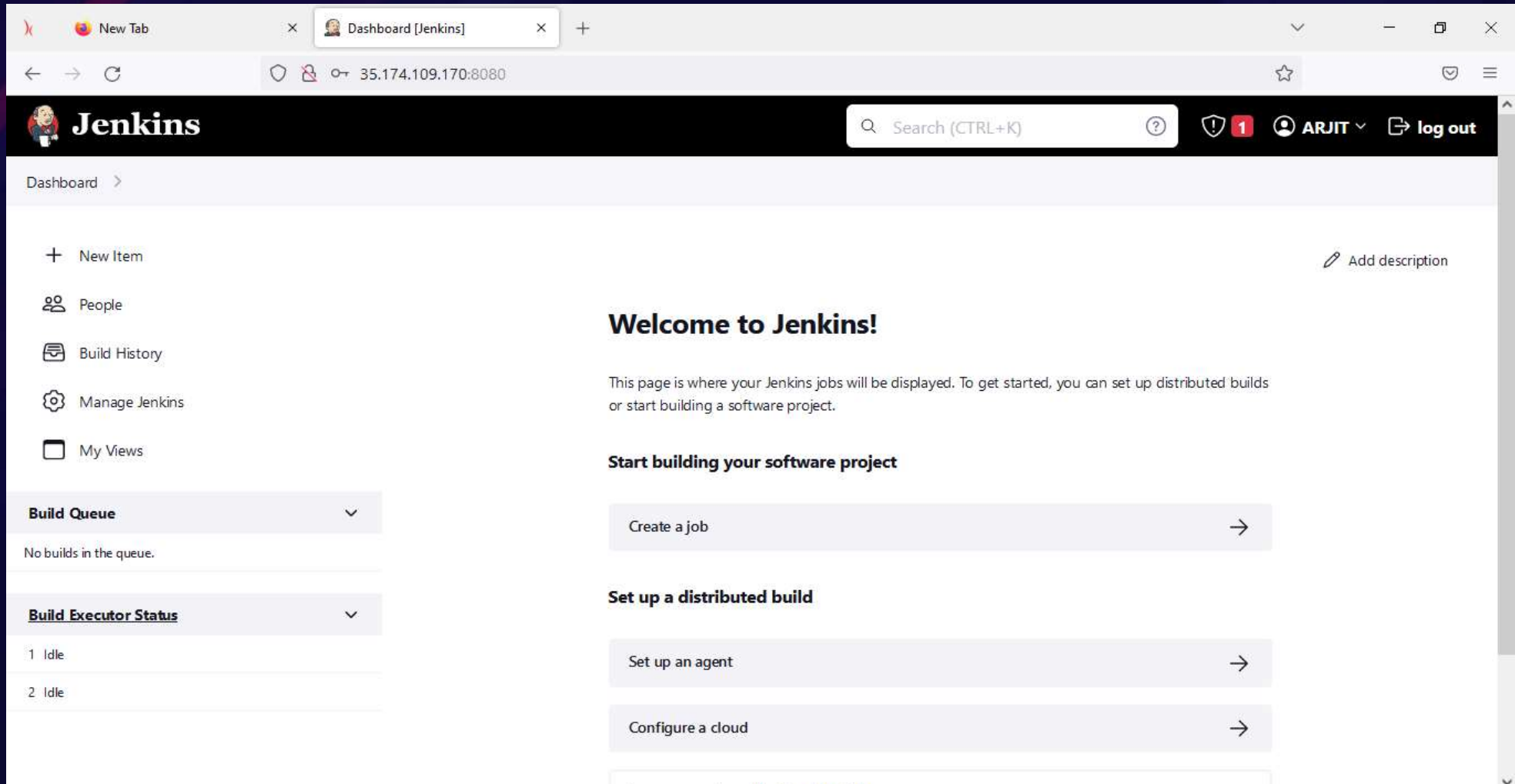
- Now copy the IP and paste in browser with port 8080
- `sudo cat /var/lib/jenkins/secrets/initialAdminPassword`



➤ Customize Jenkins



➤ Dashboard Jenkins



The screenshot shows the Jenkins Dashboard in a web browser. The browser's address bar displays the URL `35.174.109.170:8080`. The Jenkins header includes the logo, a search bar, a notification badge with the number '1', the user name 'ARJIT', and a 'log out' link. The left sidebar contains navigation links: 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area features a 'Welcome to Jenkins!' message, a brief introduction to the dashboard, and two primary action sections: 'Start building your software project' with a 'Create a job' button, and 'Set up a distributed build' with buttons for 'Set up an agent' and 'Configure a cloud'. On the left, the 'Build Queue' section is currently empty, and the 'Build Executor Status' section shows two idle executors.

Dashboard >

+ New Item Add description

People

Build History

Manage Jenkins

My Views

Build Queue ▼

No builds in the queue.

Build Executor Status ▼

1 Idle

2 Idle

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 →

DEMO -CI/CD OF Application

