

Name : Sohail Ali Khwazada  
Roll No. : 49  
Batch : T13

## Experiment 4 : Continuous Integration - Jenkins

### Aim :

To understand continuous integration, install and configure Jenkins with maven / ant / gradle to set up a build job.

### Theory :

#### 1. Continuous Integration (CI)

**Continuous Integration (CI)** is a software development practice where code changes are automatically integrated, tested, and deployed frequently, often multiple times a day. The goal of CI is to improve code quality, reduce integration issues, and speed up the development lifecycle.

##### Key Concepts of CI:

- **Automated Testing:** Code is automatically tested every time it is committed to the repository to catch bugs early.
- **Frequent Integration:** Developers integrate their code changes into a shared repository regularly to avoid conflicts.
- **Continuous Feedback:** Developers receive quick feedback about the health of their codebase, enabling them to fix issues promptly.
- **Build Automation:** The process of compiling code, running tests, and deploying applications is automated.

##### Benefits of Continuous Integration:

1. **Early Detection of Errors:** Automated tests catch bugs soon after they are introduced, reducing the cost of fixing them.

2. **Improved Code Quality:** Continuous testing ensures that only high-quality code is merged into the main codebase.
3. **Faster Development Cycles:** Developers can work in parallel without waiting for integration, enabling faster feature releases.
4. **Reduced Manual Effort:** Automating builds and tests reduces manual errors and saves time.

#### Key CI Practices:

- **Version Control System (VCS) Integration:** CI tools integrate with Git, SVN, etc., to monitor code changes.
- **Automated Builds:** Code is automatically compiled and built into deployable artifacts.
- **Automated Testing:** Unit tests, integration tests, and UI tests run automatically after every code change.
- **Deployment Automation:** Code can be automatically deployed to staging or production environments.

## 2. Jenkins: A Leading CI/CD Tool

**Jenkins** is an open-source automation server commonly used to implement CI/CD pipelines. It automates parts of software development related to building, testing, and deploying code.

#### Key Features of Jenkins:

- **Extensibility:** Jenkins has a rich ecosystem of plugins to support building, deploying, and automating projects for multiple languages and technologies.
- **Distributed Builds:** Jenkins can distribute tasks across multiple machines for faster processing.
- **Pipeline as Code:** Using Jenkins Pipeline DSL (Domain-Specific Language), you can define CI/CD workflows as code in a **Jenkinsfile**.

- **Integration with VCS:** Jenkins integrates seamlessly with Git, SVN, and other version control systems.

### 3. Jenkins Architecture

Jenkins operates on a master-slave (or controller-agent) architecture:

- **Master (Controller):** The central server that manages the CI/CD environment, schedules jobs, and monitors their execution.
- **Agents (Slaves):** Machines that perform the actual build and test tasks. Jenkins can distribute workloads to multiple agents for scalability.

### 4. Jenkins Components

- **Jobs:** Individual tasks that Jenkins can execute, such as building code, running tests, or deploying applications.
- **Builds:** The process of compiling and packaging the code. Each build can be triggered manually or automatically.
- **Pipelines:** A series of automated steps defined in a **Jenkinsfile** that describe the CI/CD process.
- **Plugins:** Extend Jenkins' functionality, allowing integration with various tools like Git, Docker, Maven, etc.

## 5. Setting Up Jenkins

### Step 1: Install Jenkins

#### On Ubuntu:

```
sudo apt update
sudo apt install openjdk-11-jdk
wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo
apt-key add -
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/
> /etc/apt/sources.list.d/jenkins.list'
sudo apt update
sudo apt install jenkins
sudo systemctl start jenkins
sudo systemctl status jenkins
```

- **On Windows:**

Download the Jenkins installer from the Jenkins official website and follow the installation wizard.

### Step 2: Access Jenkins Web Interface

- Open your browser and go to: <http://localhost:8080>

Retrieve the initial admin password:

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

- Unlock Jenkins and install the suggested plugins.

---

## 6. Configuring Jenkins for CI

### Step 1: Create a New Job

1. Click “**New Item**” in the Jenkins dashboard.
2. Enter a name for the job.

### 3. Choose the job type:

- **Freestyle Project:** Basic job configuration.
- **Pipeline:** Advanced, code-defined CI/CD workflow.

### Step 2: Configure Source Code Repository

- **For GitHub Integration:**

- In the job configuration, go to “**Source Code Management**”.
- Select **Git** and enter the repository URL.
- Add credentials if necessary.

### Step 3: Add Build Triggers

- Configure triggers to automate builds:
  - **Poll SCM:** Periodically check for changes in the repository.
  - **GitHub Hook Trigger:** Trigger builds automatically when changes are pushed to GitHub.

### Step 4: Add Build Steps

- **Execute Shell:** Run shell commands or scripts.
- **Invoke Ant/Maven/Gradle:** Run build tools for Java projects.
- **Execute Windows Batch Command:** Run batch files on Windows systems.

### Step 5: Add Post-Build Actions

- **Archive Artifacts:** Save build outputs for later use.
- **Send Notifications:** Email alerts or Slack messages after builds.
- **Deploy Applications:** Automate deployment to servers.

## 7. Jenkins Pipeline (Declarative & Scripted)

### Declarative Pipeline (Recommended for Most Cases):

A declarative pipeline is defined in a **Jenkinsfile** using a simplified syntax.

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                echo 'Building the project...'
                sh 'mvn clean install'
            }
        }
        stage('Test') {
            steps {
                echo 'Running tests...'
                sh 'mvn test'
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying application...'
                sh 'scp target/app.jar
user@server:/path/to/deploy'
            }
        }
    }
}
```

### Scripted Pipeline (More Flexible):

Scripted pipelines offer more control but require a deeper understanding of Groovy.

```
node {
    stage('Build') {
        sh 'mvn clean install'
    }
    stage('Test') {
```

```
        sh 'mvn test'
    }
    stage('Deploy') {
        sh 'scp target/app.jar user@server:/path/to/deploy'
    }
}
```

## 8. Common Jenkins Plugins

- **Git Plugin:** Integrates Jenkins with Git repositories.
- **Pipeline Plugin:** Enables the use of Jenkins Pipelines.
- **Docker Pipeline:** Manages Docker containers in CI/CD workflows.
- **Blue Ocean:** A modern UI for visualizing Jenkins pipelines.
- **Slack Notification Plugin:** Sends notifications to Slack channels.

## 9. Advanced Jenkins Concepts

### 9.1. Jenkinsfile as Code

Defining your pipeline in a **Jenkinsfile** allows you to version control your CI/CD process alongside your application code.

### 9.2. Parallel Execution

You can run multiple stages in parallel to speed up your pipeline:

```
pipeline {
    agent any
    stages {
        stage('Parallel Stages') {
            parallel {
                stage('Unit Test') {
                    steps {
                        sh 'mvn test'
                    }
                }
            }
        }
    }
}
```

```

    }
    stage('Integration Test') {
        steps {
            sh 'mvn verify'
        }
    }
}

```

### 9.3. Parameterized Builds

Allow users to pass parameters to Jenkins jobs:

```

pipeline {
    agent any
    parameters {
        string(name: 'ENV', defaultValue: 'staging',
description: 'Deployment Environment')
    }
    stages {
        stage('Deploy') {
            steps {
                sh "deploy.sh ${params.ENV}"
            }
        }
    }
}

```

## 10. Jenkins Security Best Practices

- **Use Role-Based Access Control:** Limit permissions based on user roles.
- **Secure Credentials:** Store sensitive data (like API keys) in Jenkins Credentials Manager.



- **Keep Jenkins Updated:** Regularly update Jenkins and plugins to fix security vulnerabilities.

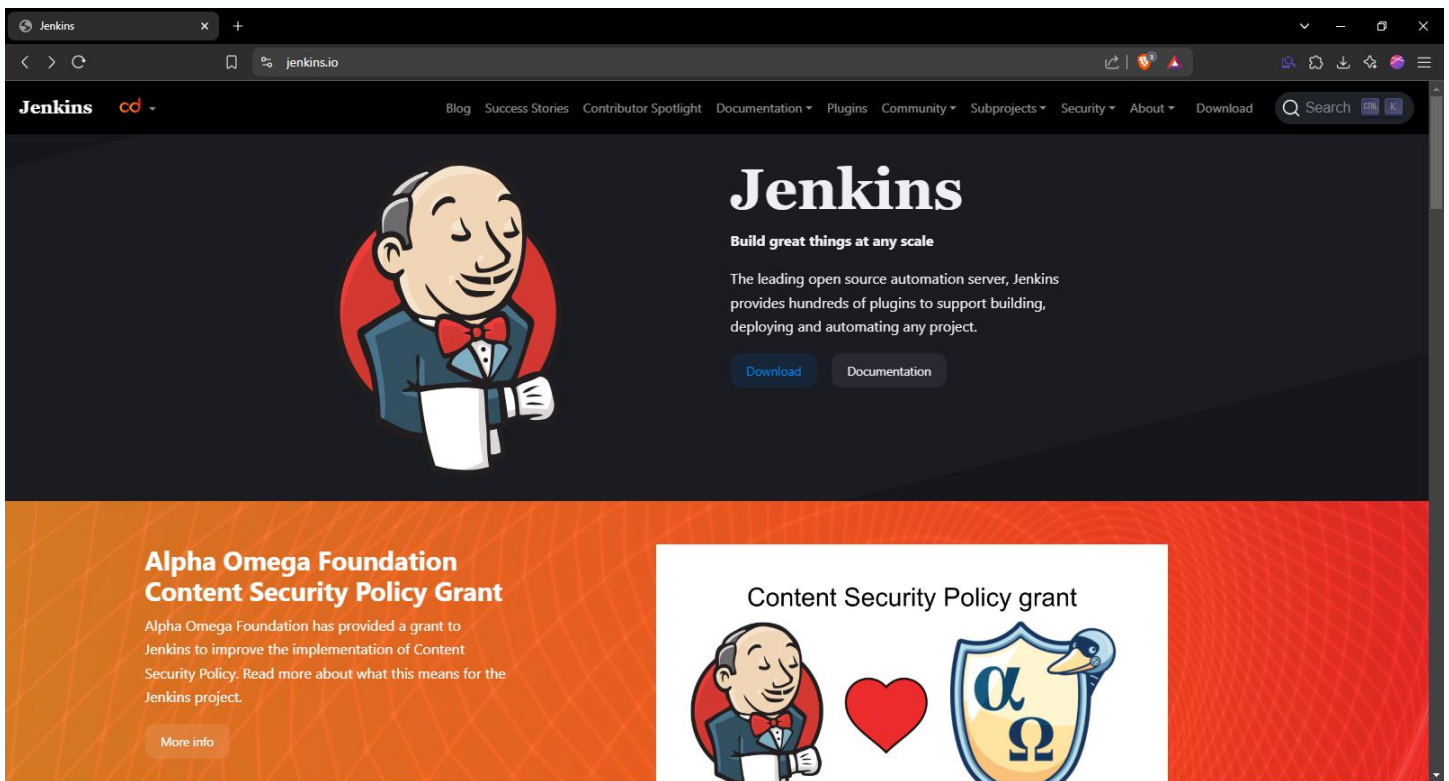
## 11. Troubleshooting Jenkins Issues

- **Build Fails:** Check the console output for error logs.
- **Permissions Issues:** Verify user permissions in Jenkins settings.
- **Plugin Conflicts:** Update or reinstall problematic plugins.

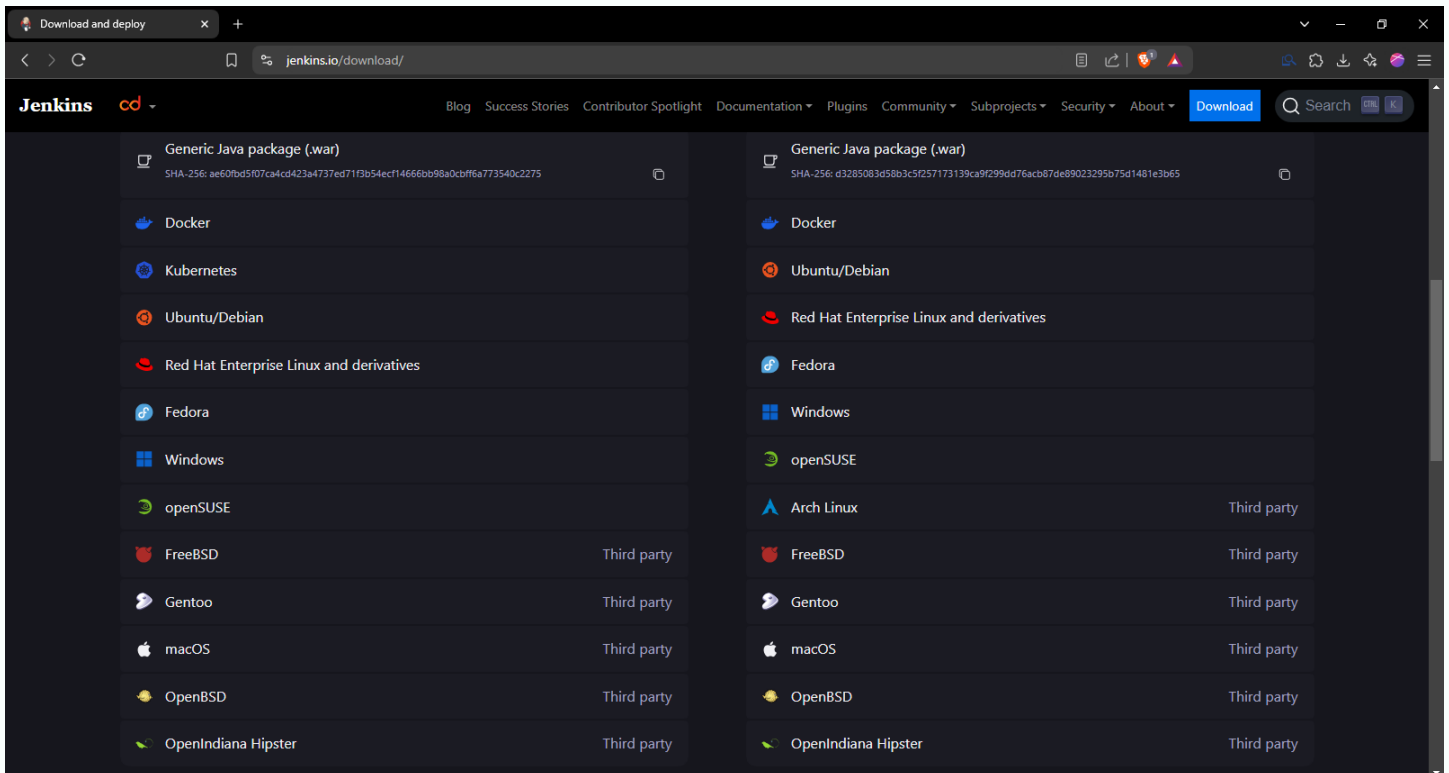
## 12. Summary of Key Jenkins Commands

Operation	Command/Action
Start Jenkins	<code>sudo systemctl start jenkins</code>
Stop Jenkins	<code>sudo systemctl stop jenkins</code>
Restart Jenkins	<code>sudo systemctl restart jenkins</code>
Check Jenkins Status	<code>sudo systemctl status jenkins</code>
Access Jenkins Dashboard	<code>http://localhost:8080</code>
Install Plugins	Manage Jenkins → Manage Plugins
Run Pipeline Manually	Click <b>“Build Now”</b>
View Build Logs	Click on a build → <b>Console Output</b>

# Output :



The screenshot shows the Jenkins homepage at [jenkins.io](https://jenkins.io). The header includes the Jenkins logo and navigation links: Blog, Success Stories, Contributor Spotlight, Documentation, Plugins, Community, Subprojects, Security, About, and Download. A search bar is located on the right. The main content area features a large illustration of the Jenkins mascot, a man in a tuxedo, holding a document. To the right of the mascot, the text reads: "Jenkins", "Build great things at any scale", and "The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project." Below this text are two buttons: "Download" and "Documentation". A prominent orange banner at the bottom of the page announces the "Alpha Omega Foundation Content Security Policy Grant". The banner text states: "Alpha Omega Foundation has provided a grant to Jenkins to improve the implementation of Content Security Policy. Read more about what this means for the Jenkins project." and includes a "More info" button. To the right of the text is an illustration of the Jenkins mascot, a red heart, and the Alpha Omega symbol (α Ω) inside a shield.



The screenshot shows the Jenkins download page at [jenkins.io/download/](https://jenkins.io/download/). The header is identical to the homepage, but the "Download" button is highlighted in blue. The main content area displays a list of operating systems and distributions supported by Jenkins, categorized under "Generic Java package (.war)". The list is divided into two columns. The left column lists: Docker, Kubernetes, Ubuntu/Debian, Red Hat Enterprise Linux and derivatives, Fedora, Windows, openSUSE, FreeBSD, Gentoo, macOS, OpenBSD, and OpenIndiana Hipster. The right column lists: Docker, Ubuntu/Debian, Red Hat Enterprise Linux and derivatives, Fedora, Windows, openSUSE, Arch Linux, FreeBSD, Gentoo, macOS, OpenBSD, and OpenIndiana Hipster. Each entry includes a SHA-256 hash and a "Third party" label.

Operating System / Distribution	SHA-256 Hash	Third party
Generic Java package (.war)	SHA-256: ae60bd5d07ca4cd423a4737ed71f3b54ecf14666bb98a0cbff6a773540c2275	
Docker		
Kubernetes		
Ubuntu/Debian		
Red Hat Enterprise Linux and derivatives		
Fedora		
Windows		
openSUSE		
FreeBSD		Third party
Gentoo		Third party
macOS		Third party
OpenBSD		Third party
OpenIndiana Hipster		Third party

Thank you for downloading Windows Stable installer

Download hasn't started? [Click this link](#)

### Changing boot configuration

By default, your Jenkins runs at <https://localhost:8080/>. This can be changed by editing `jenkins.xml`, which is located in your installation directory. This file also contains boot configuration parameters, such as JVM options, HTTPS setup, etc.

### Starting/stopping the service

Jenkins is installed as a Windows service, and it is configured to start automatically upon boot. To start/stop them manually, use the service manager from the control panel, or the `sc` command line tool.

### Inheriting your existing Jenkins installation

If you'd like your new installation to take over your existing Jenkins data, copy your old data directory into the new `JENKINS_HOME` directory.

### See Also

- [Running Jenkins behind Internet Information Server \(IIS\)](#)
- [Running Jenkins behind nginx](#)
- [Running Jenkins behind Apache](#)

[Report an Infra Issue](#)

[Resources](#) [Project](#) [Community](#) [Other](#)

Recent download history

- jenkins.msi  
98.1 MB • Done
- SEPM\_Exp03\_T11\_03.pdf  
480 KB • 13 minutes ago
- SEPM\_Exp02\_T11\_03.pdf  
790 KB • 40 minutes ago
- SEPM\_Exp02\_T11\_03.pdf  
790 KB • 41 minutes ago

[Full download history](#)



Jenkins 2.492.2 Setup

**Destination Folder**

Click Next to install to the default folder or click Change to choose another.

Install Jenkins 2.492.2 to:

C:\Program Files\Jenkins\

Change...

Back Next Cancel

Jenkins 2.492.2 Setup

**Service Logon Credentials**

Enter service credentials for the service.

Jenkins 2.492.2 installs and runs as an independent Windows service. To operate in this manner, you must supply the user account credentials for Jenkins 2.492.2 to run successfully.

**Logon Type:**

☒ Run service as LocalSystem (not recommended)

☐ Run service as local or domain user:

Account:

Password:

Test Credentials

Back Next Cancel

Jenkins 2.492.2 Setup


**Port Selection**

Choose a port for the service.

Please choose a port.

**Port Number (1-65535):**

8080

Test Port 

It is recommended that you accept the selected default port.

Back Next Cancel

Jenkins 2.492.2 Setup

Jenkins 2.492.2 Setup

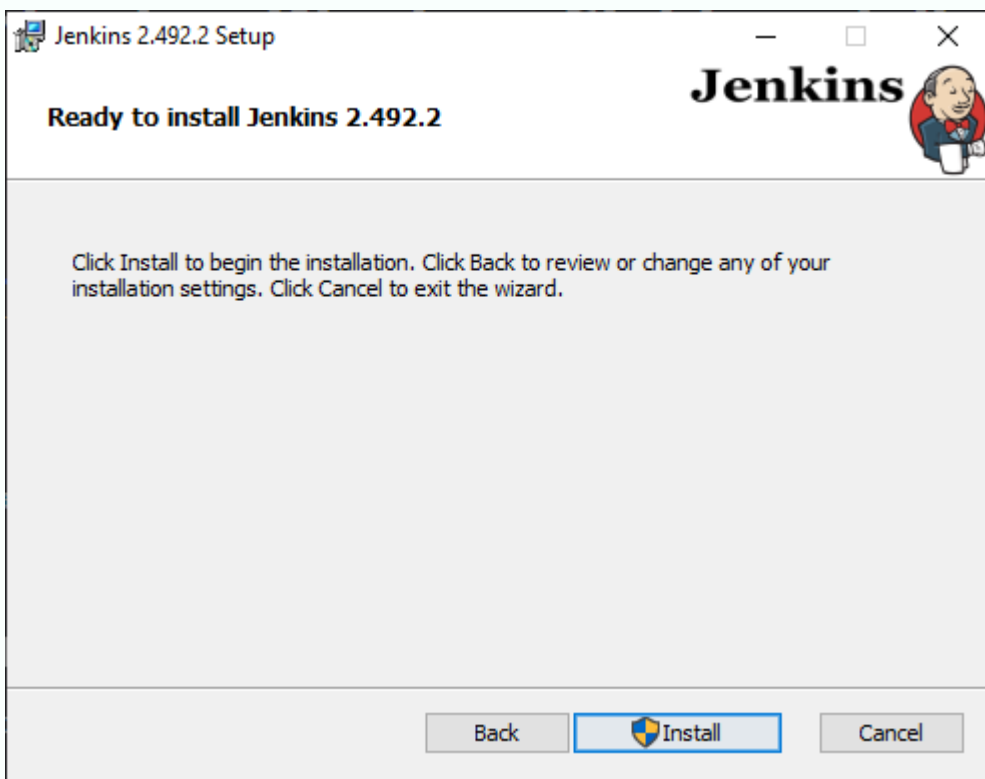
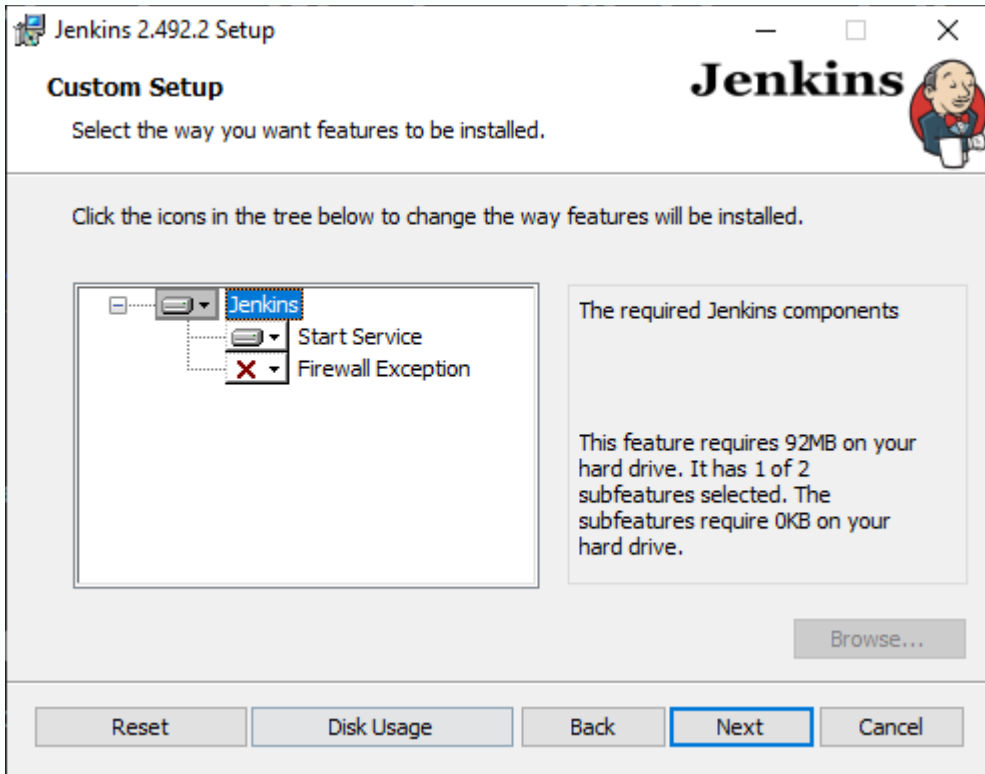
Select Java home directory (JDK or JRE)

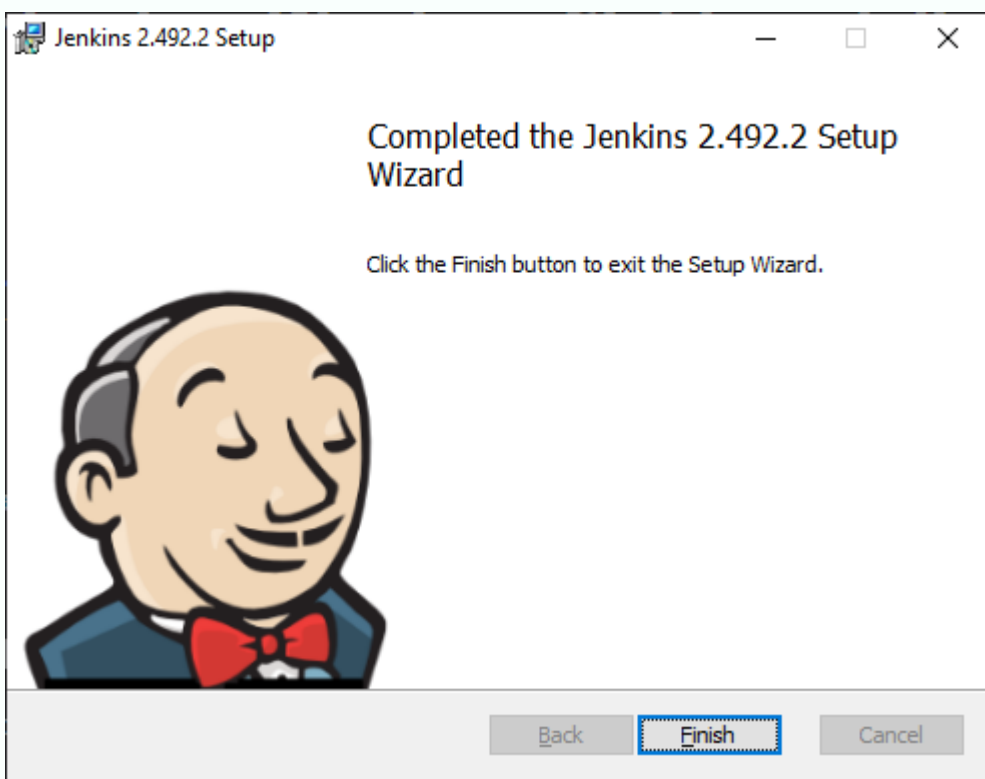
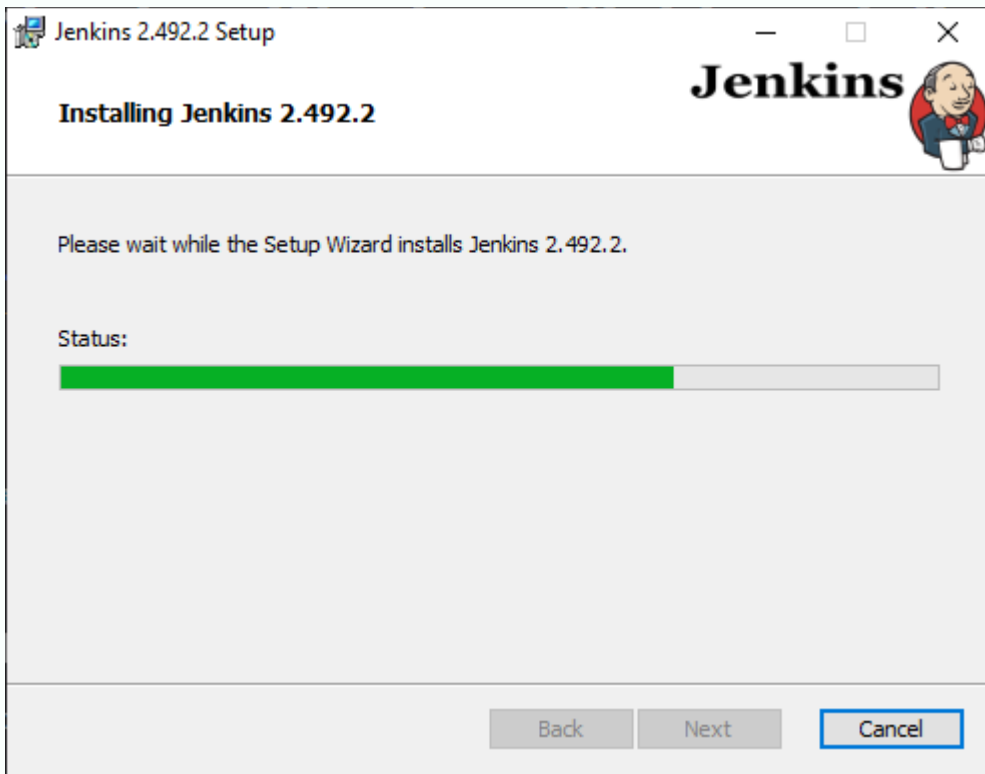
Please select the path of a Java Development Kit or Java Runtime Environment.  
Only Java 17 and 21 are supported by Jenkins.

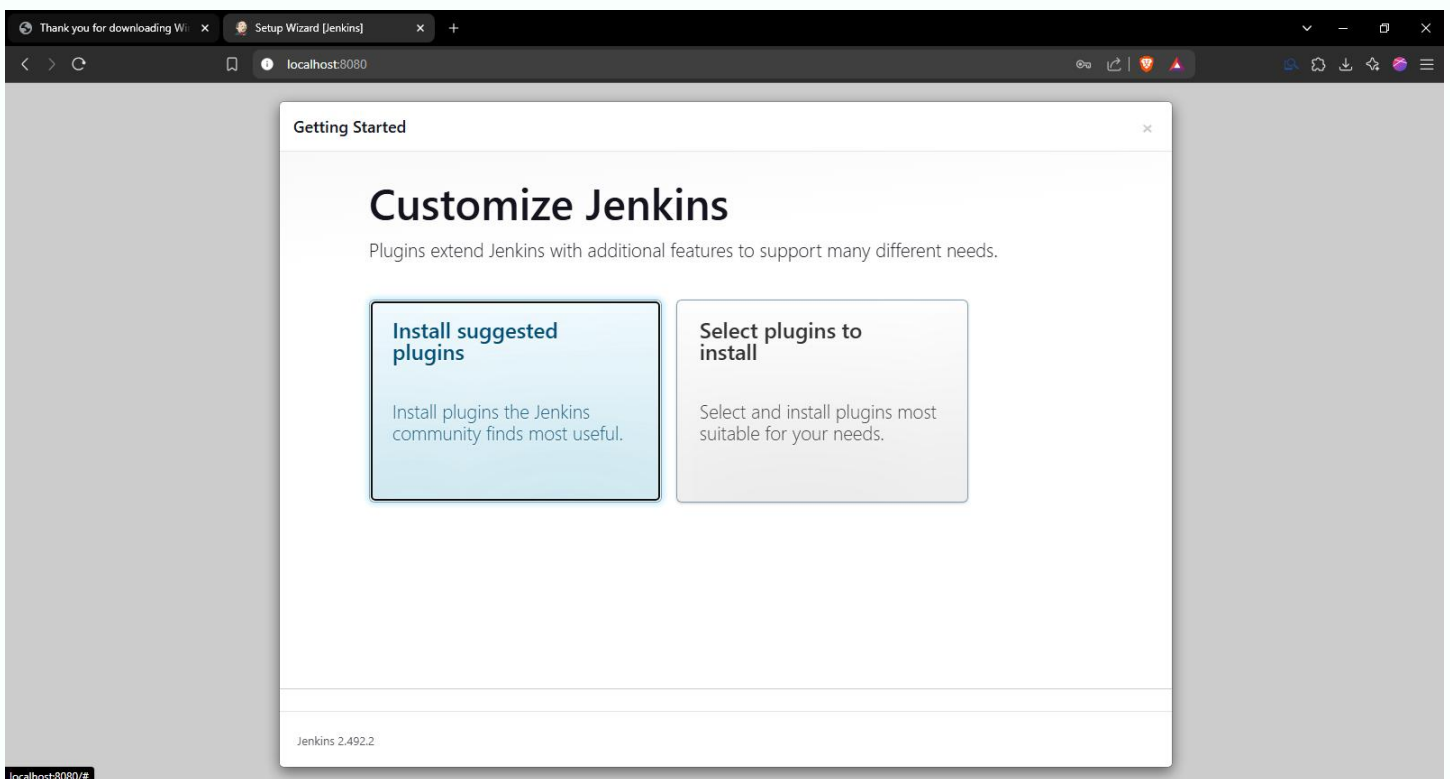
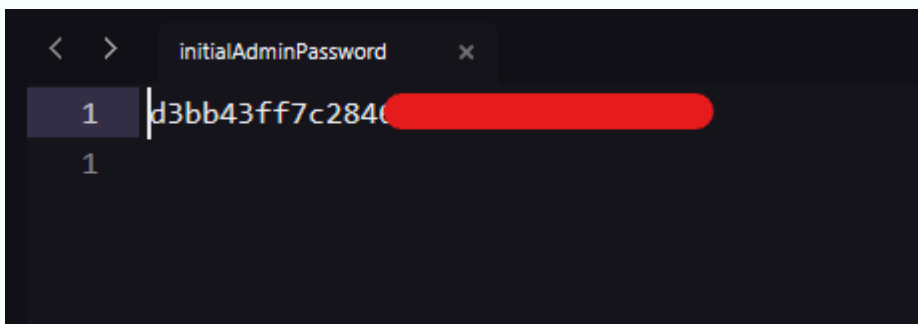
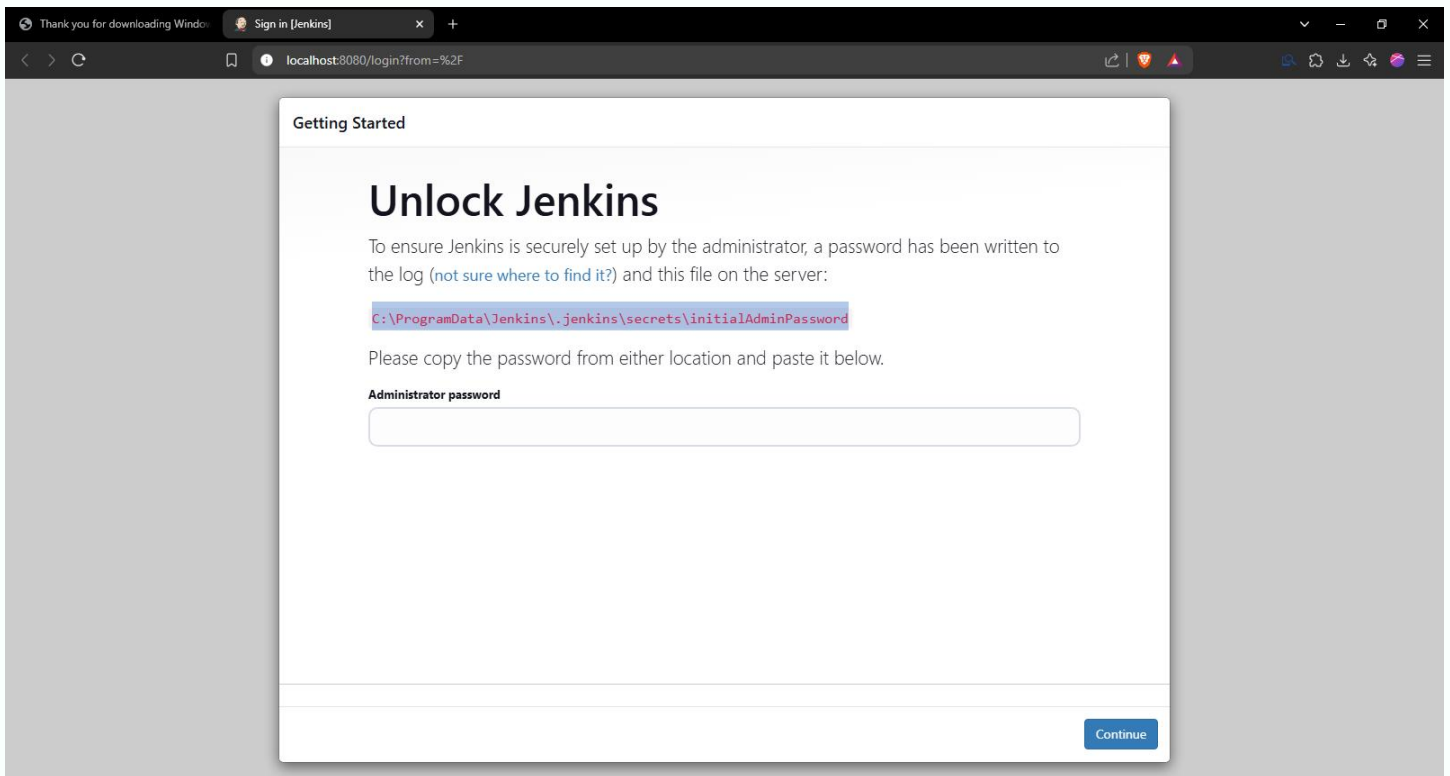
C:\Program Files\Amazon Corretto\jdk21.0.5\_11\

Change...

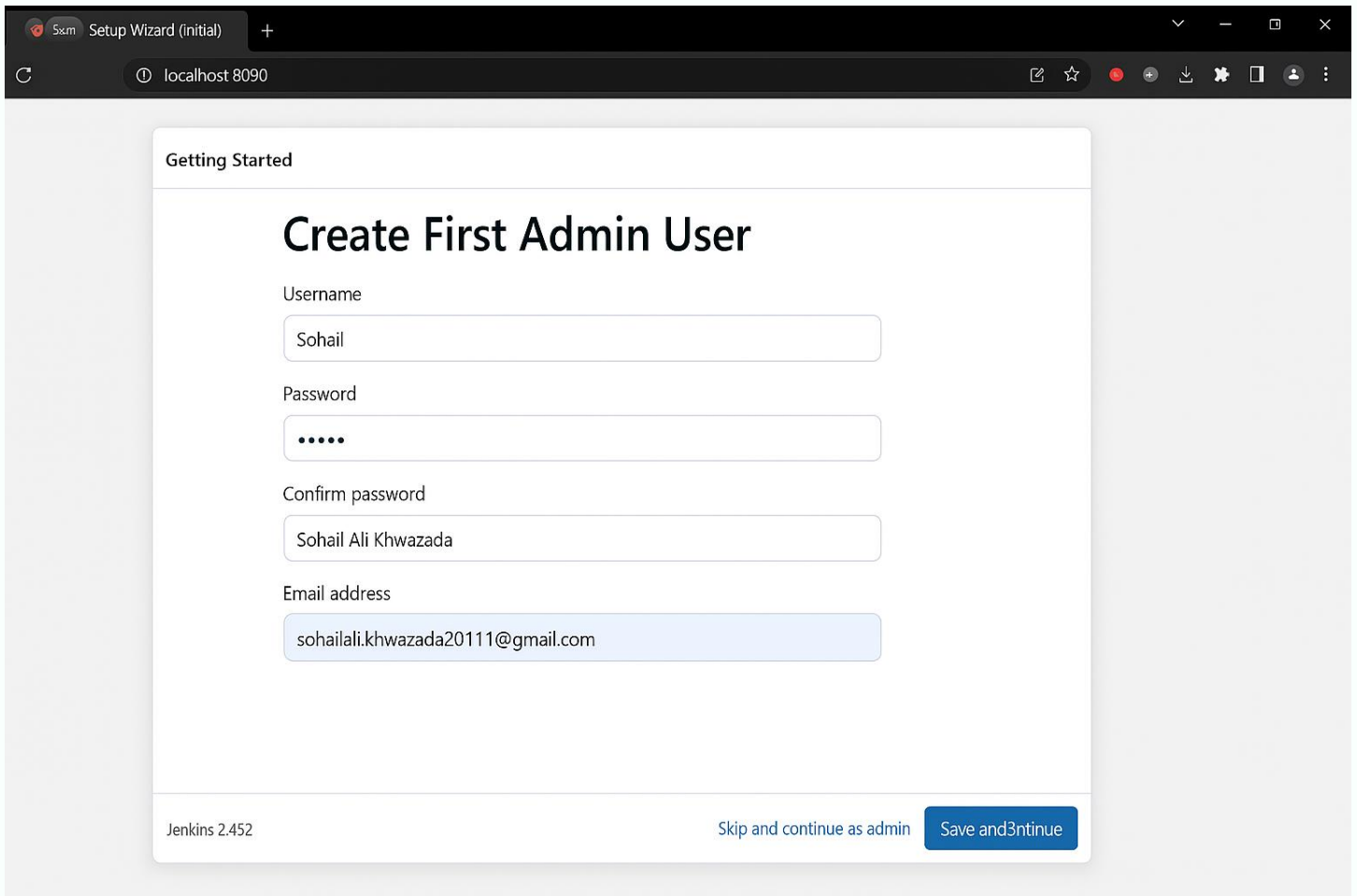
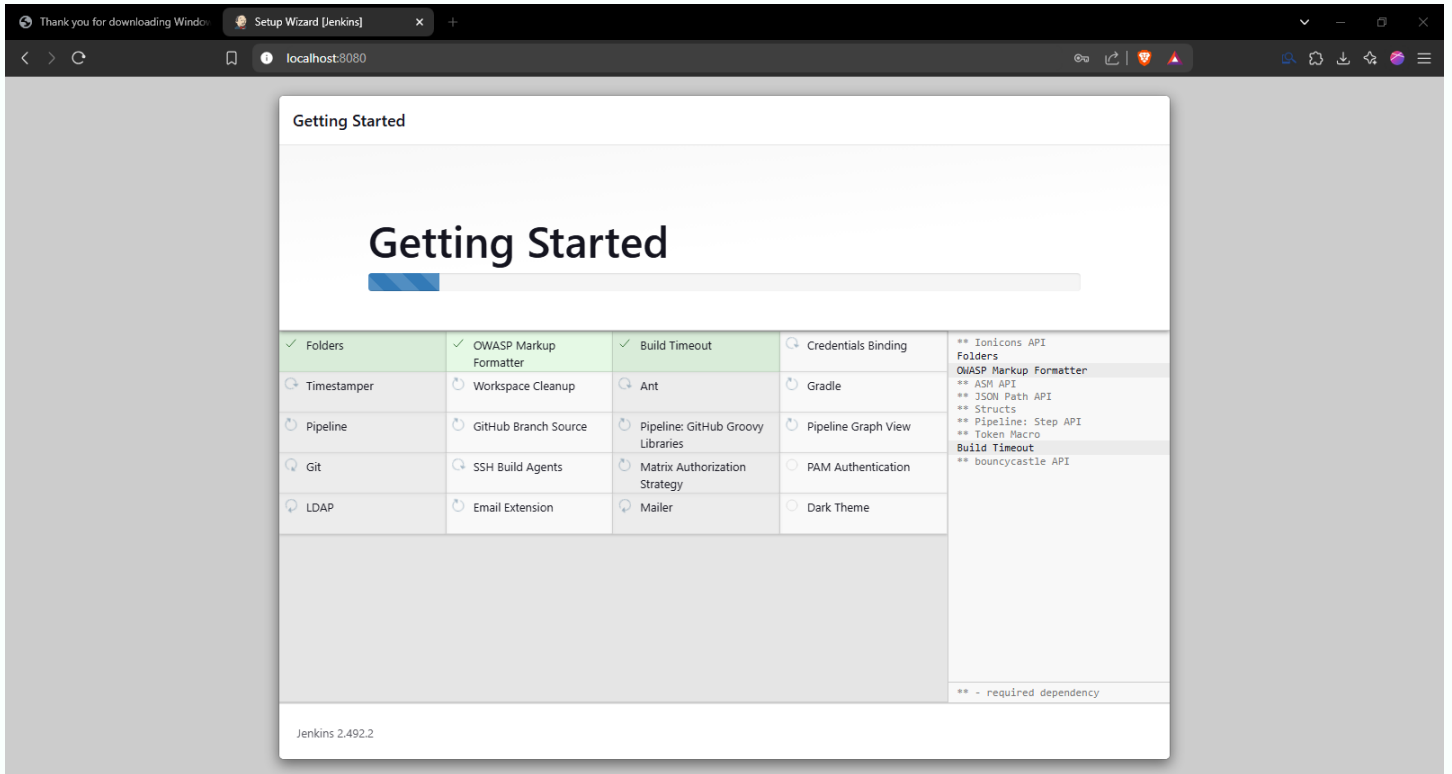
Back Next Cancel

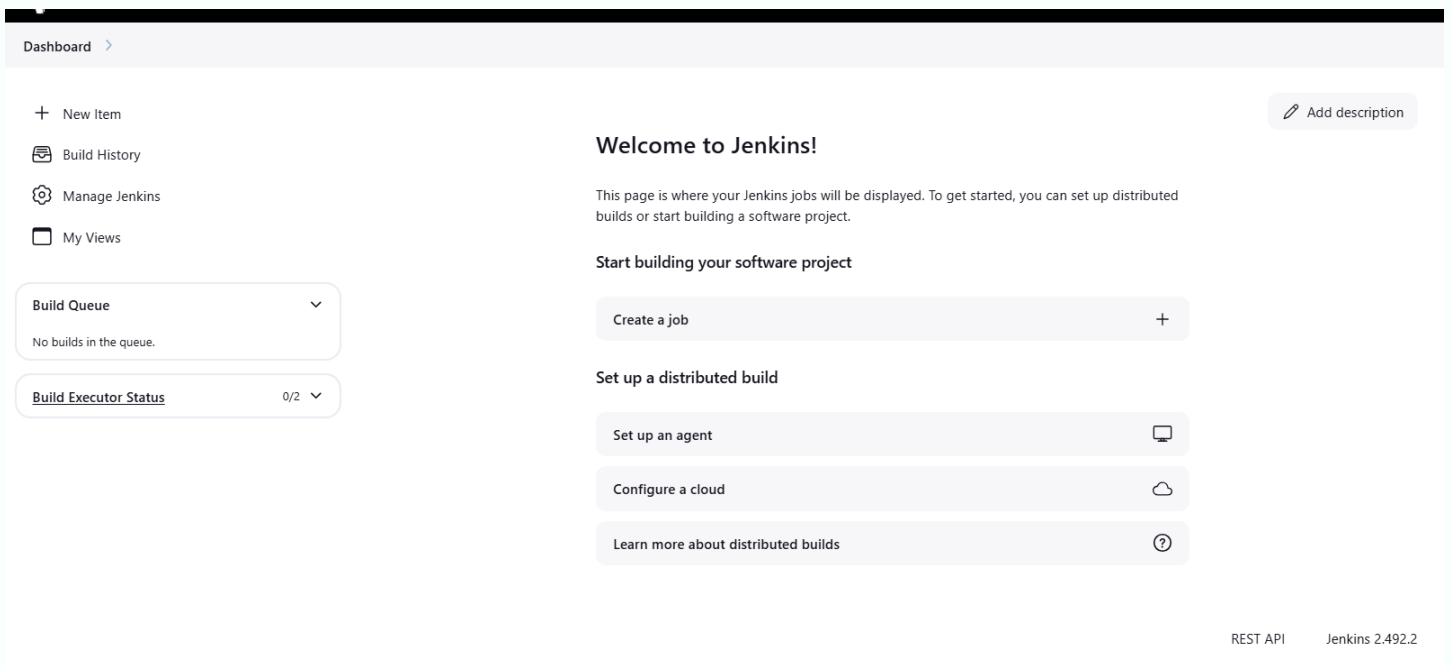
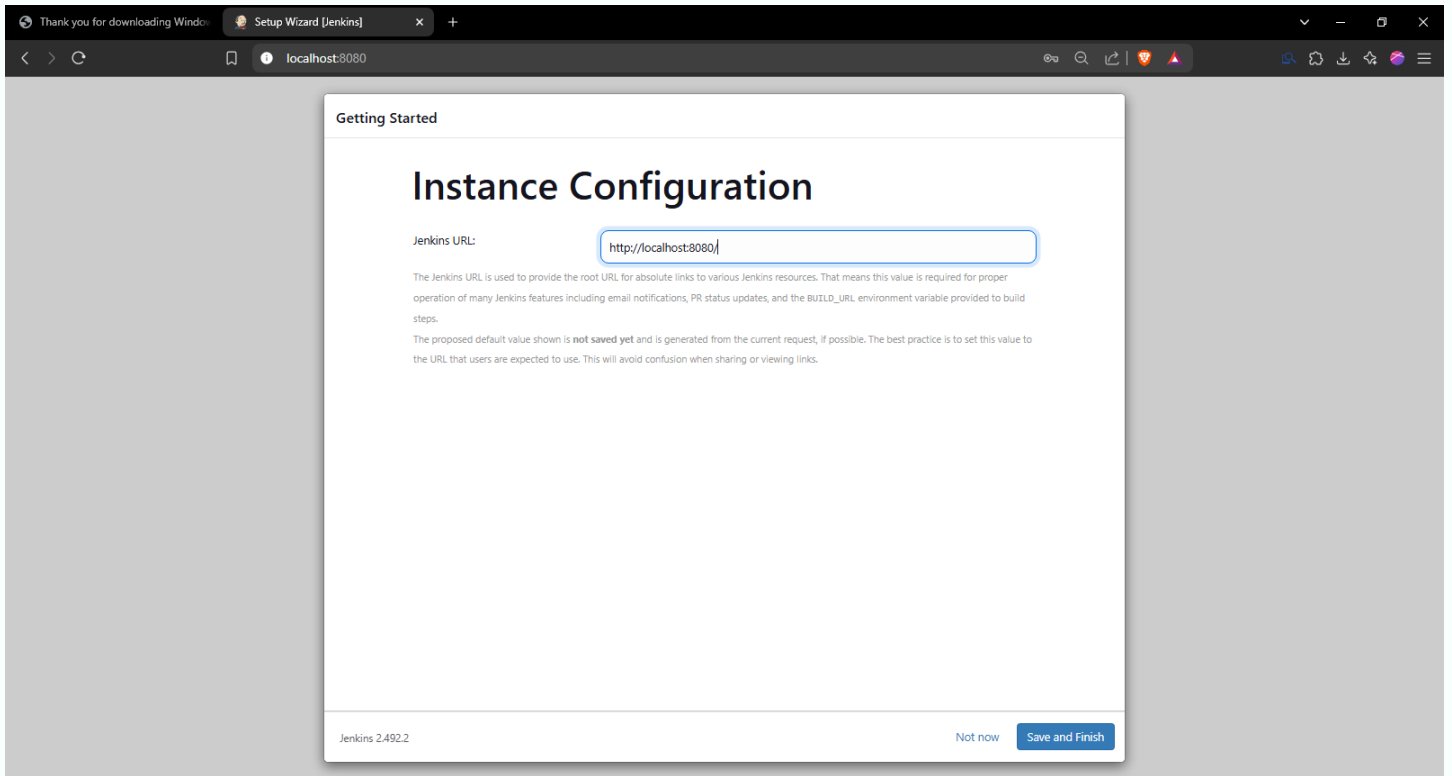


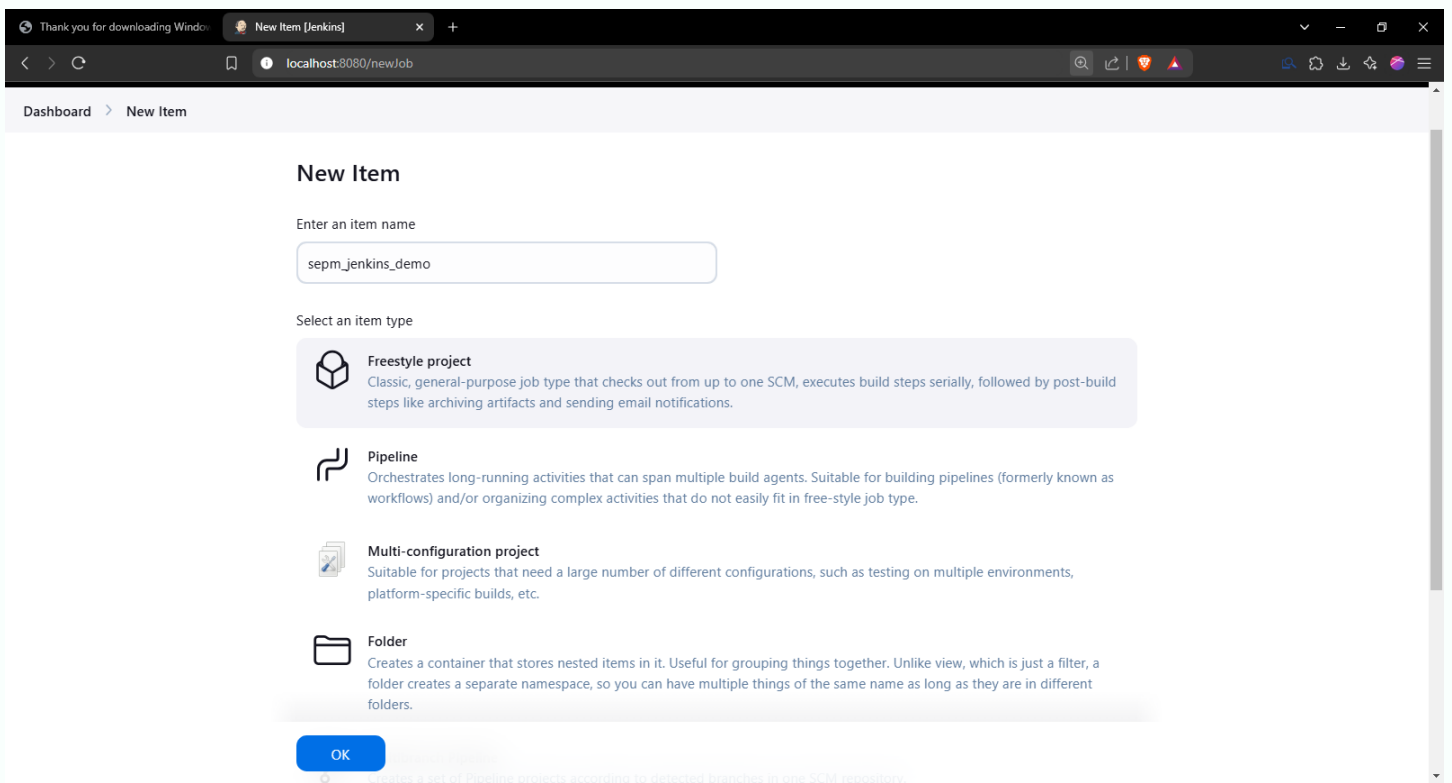
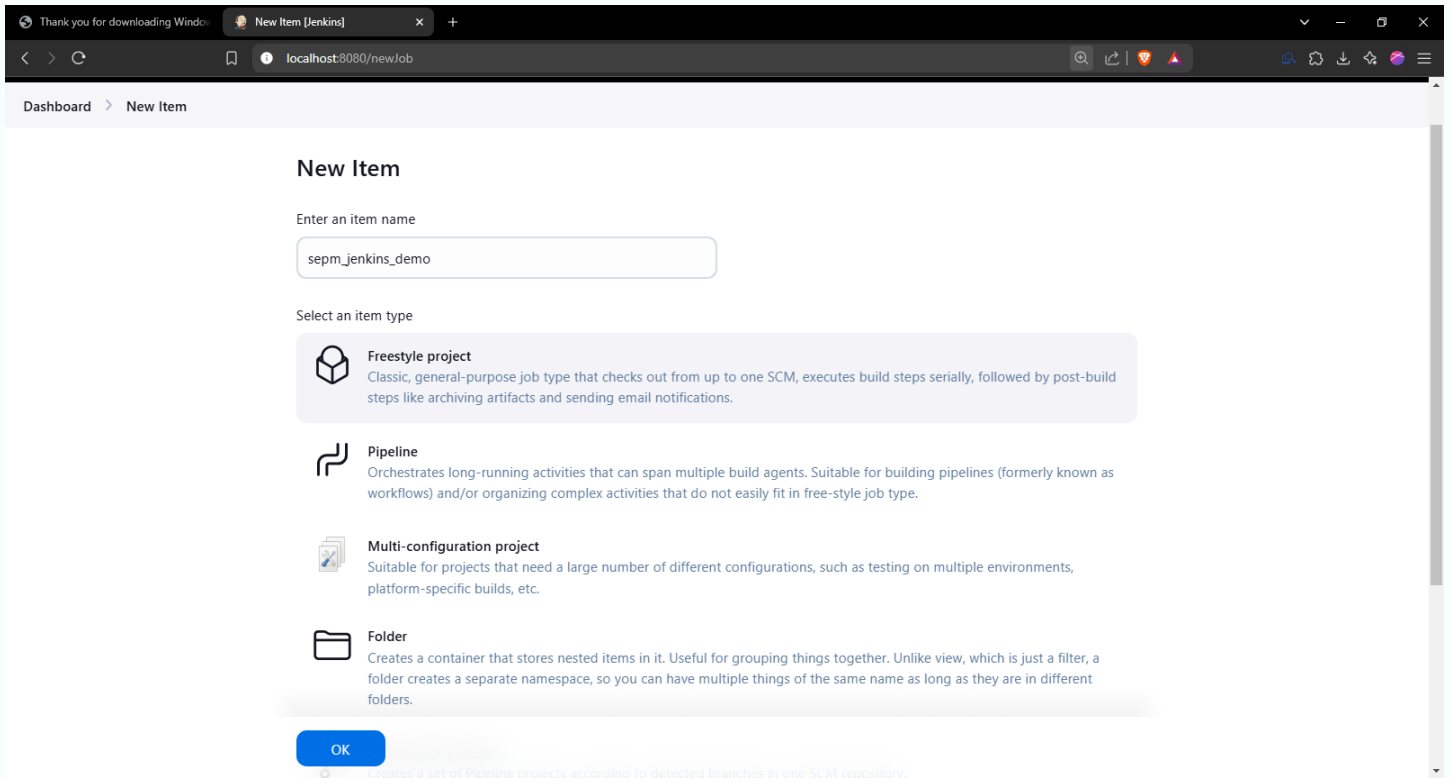












Thank you for downloading Windows

sepm\_jenkins\_demo Config [U] x +

localhost:8080/job/sepm\_jenkins\_demo/configure

Dashboard > sepm\_jenkins\_demo > Configuration

Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

Use secret text(s) or file(s) ?

Add timestamps to the Console Output

Inspect build log for published build scans

Terminate a build if it's stuck

With Ant ?

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Execute Windows batch command ?

Command

See [the list of available environment variables](#)

echo "Hello, World!"

Advanced

Save

Apply

Dashboard > sepm\_jenkins\_demo > #1

Status

Changes

Console Output

Edit Build Information

Delete build '#1'

Timings

✓ #1 (Apr 2, 2025, 11:28:04 AM)

Add description

Keep this build forever

Started by user [Taha Alotwala](#)

Started 15 sec ago  
Took [1.1 sec](#)

This run spent:

- 74 ms waiting;
- 1.1 sec build duration;
- 1.2 sec total from scheduled to completion.

</>

No changes.

REST API

Jenkins 2.492.2

20

Dashboard > sepm\_jenkins\_demo > #1 > Console Output

Status

</> Changes

**Console Output**

Edit Build Information

Delete build '#1'

Timings

✓ Console Output

Download Copy View as plain text

Started by user [Taha Alotwala](#)

Running as SYSTEM

Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sepm\_jenkins\_demo

[sepm\_jenkins\_demo] \$ cmd /c call C:\Windows\TEMP\jenkins5434933225770186855.bat

C:\ProgramData\Jenkins\.jenkins\workspace\sepm\_jenkins\_demo>echo "Hello, World!"

"Hello, World!"

C:\ProgramData\Jenkins\.jenkins\workspace\sepm\_jenkins\_demo>exit 0

Finished: SUCCESS

REST API Jenkins 2.492.2

## Conclusion :

We have successfully installed and configured Jenkins with Maven/Ant/Gradle to setup a build Job and learnt about the implementation of Jenkins in open source continuous integration.

21