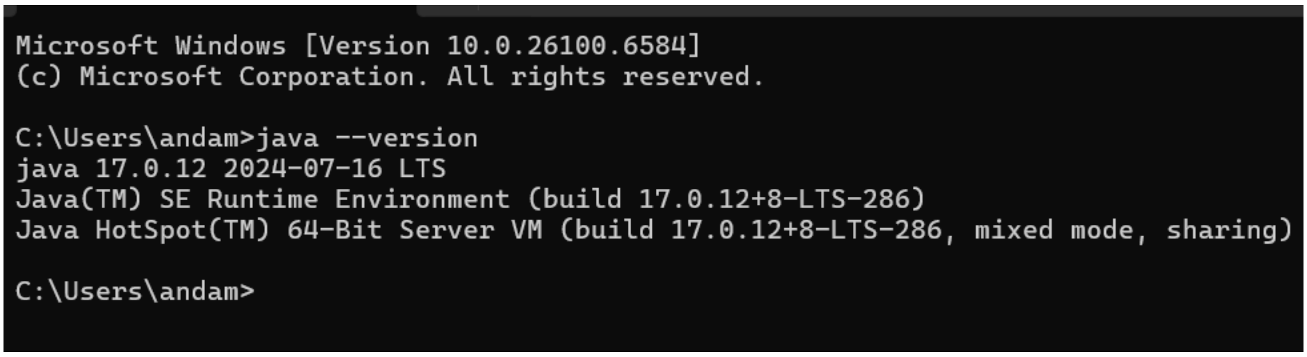
A.Bhavagna-23BD1A050C

Week 8: Jenkins Automation

Task -1 MAVEN INSTALLATION

Step1: Prerequisites (java version)



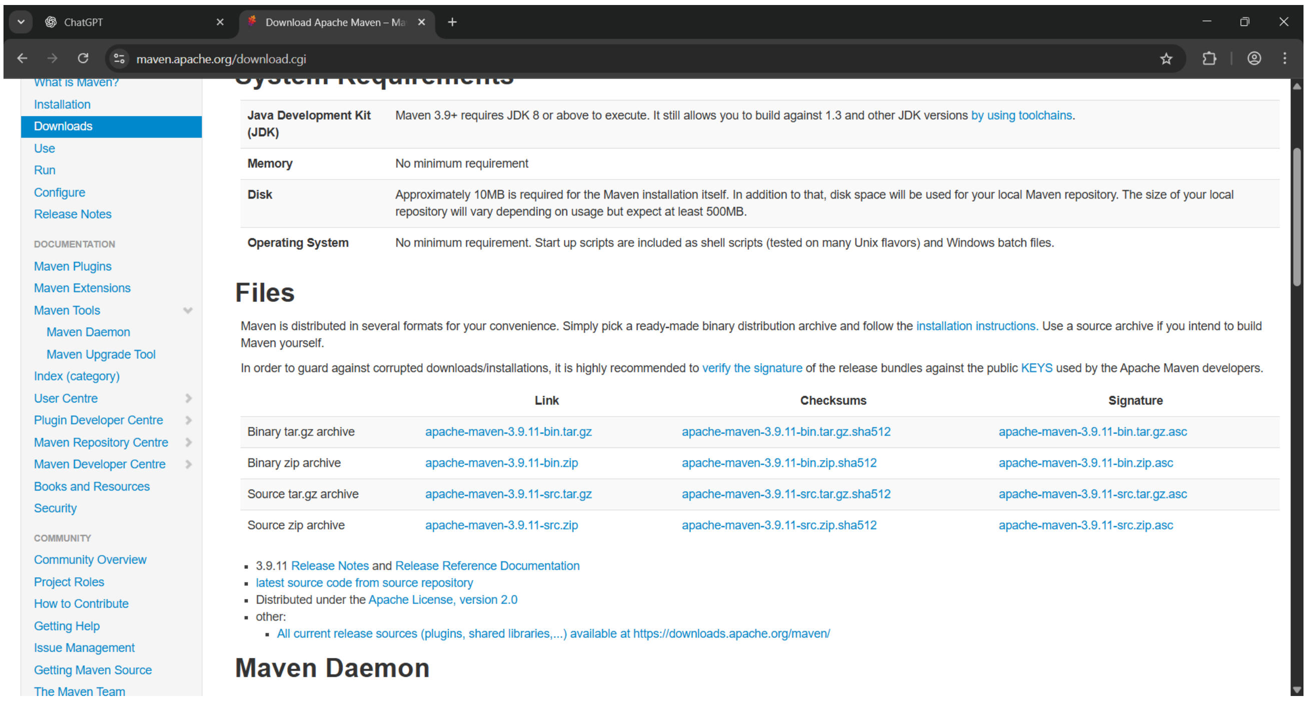
```
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\andam>java --version
java 17.0.12 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 17.0.12+8-LTS-286)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.12+8-LTS-286, mixed mode, sharing)

C:\Users\andam>
```

Step 2: Download Maven

Version 3.9.11



Java Development Kit (JDK)	Maven 3.9+ requires JDK 8 or above to execute. It still allows you to build against 1.3 and other JDK versions by using toolchains.
Memory	No minimum requirement
Disk	Approximately 10MB is required for the Maven installation itself. In addition to that, disk space will be used for your local Maven repository. The size of your local repository will vary depending on usage but expect at least 500MB.
Operating System	No minimum requirement. Start up scripts are included as shell scripts (tested on many Unix flavors) and Windows batch files.

Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the installation instructions. Use a source archive if you intend to build Maven yourself.

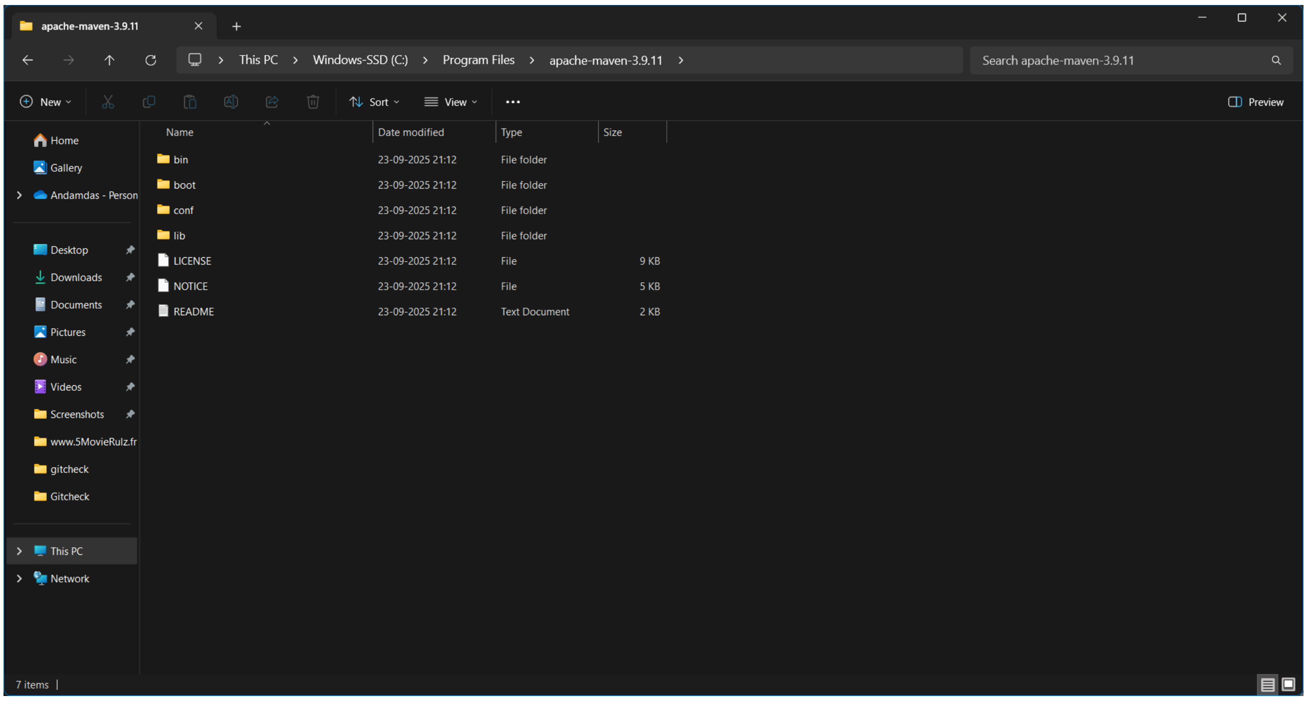
In order to guard against corrupted downloads/installations, it is highly recommended to verify the signature of the release bundles against the public KEYS used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.11-bin.tar.gz	apache-maven-3.9.11-bin.tar.gz.sha512	apache-maven-3.9.11-bin.tar.gz.asc
Binary zip archive	apache-maven-3.9.11-bin.zip	apache-maven-3.9.11-bin.zip.sha512	apache-maven-3.9.11-bin.zip.asc
Source tar.gz archive	apache-maven-3.9.11-src.tar.gz	apache-maven-3.9.11-src.tar.gz.sha512	apache-maven-3.9.11-src.tar.gz.asc
Source zip archive	apache-maven-3.9.11-src.zip	apache-maven-3.9.11-src.zip.sha512	apache-maven-3.9.11-src.zip.asc

- 3.9.11 Release Notes and Release Reference Documentation
- latest source code from source repository
- Distributed under the Apache License, version 2.0
- other:
  - All current release sources (plugins, shared libraries,...) available at https://downloads.apache.org/maven/

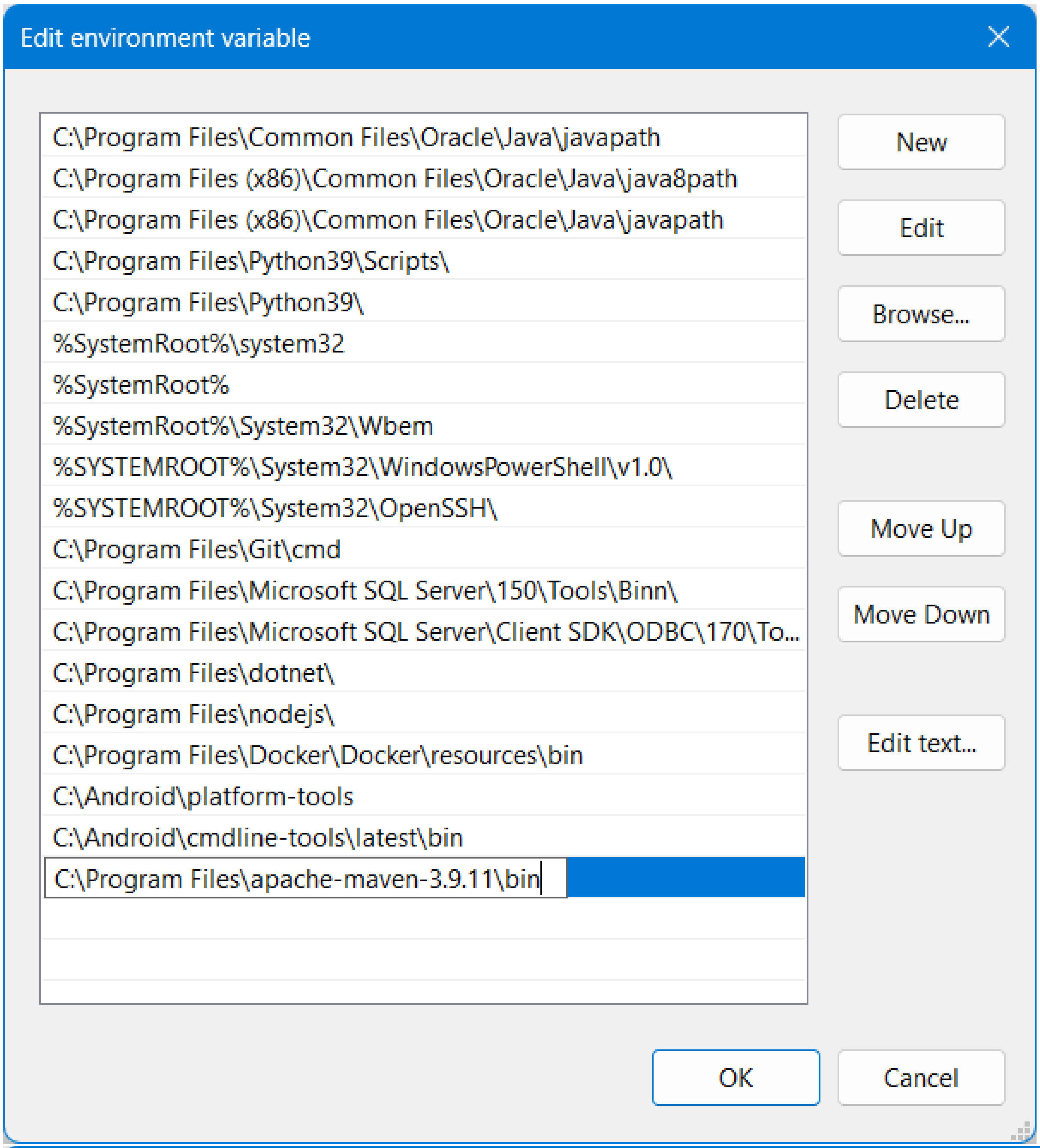
Maven Daemon

Step 3: Extract Maven

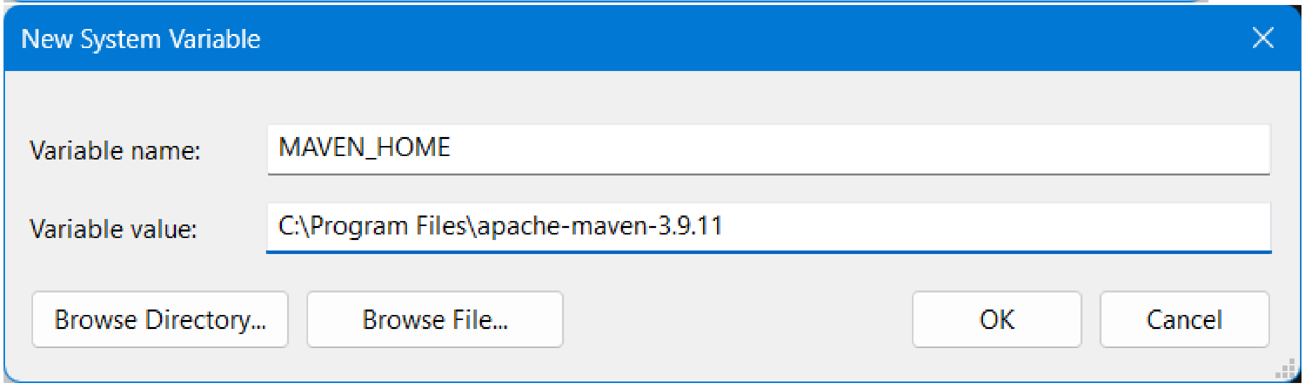


Name	Date modified	Type	Size
bin	23-09-2025 21:12	File folder	
boot	23-09-2025 21:12	File folder	
conf	23-09-2025 21:12	File folder	
lib	23-09-2025 21:12	File folder	
LICENSE	23-09-2025 21:12	File	9 KB
NOTICE	23-09-2025 21:12	File	5 KB
README	23-09-2025 21:12	Text Document	2 KB

Step 4: Set Environment Variables



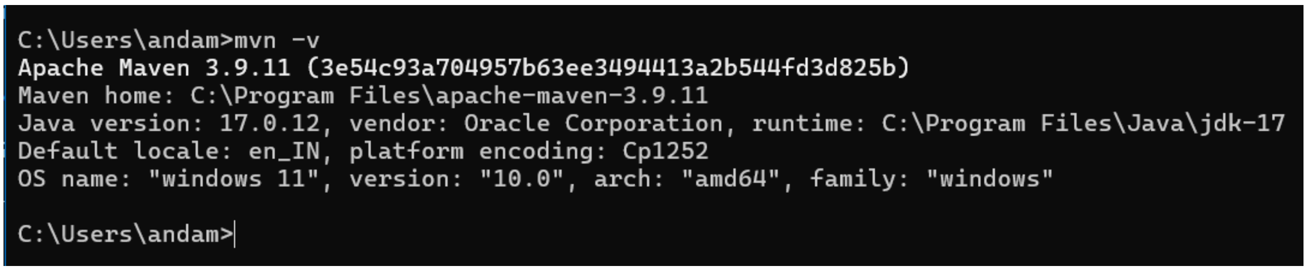
```
C:\Program Files\Common Files\Oracle\Java\javapath
C:\Program Files (x86)\Common Files\Oracle\Java\java8path
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
C:\Program Files\Python39\Scripts\
C:\Program Files\Python39\
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Git\cmd
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\To...
C:\Program Files\dotnet\
C:\Program Files\nodejs\
C:\Program Files\Docker\Docker\resources\bin
C:\Android\platform-tools
C:\Android\cmdline-tools\latest\bin
C:\Program Files\apache-maven-3.9.11\bin
```



Variable name: MAVEN_HOME

Variable value: C:\Program Files\apache-maven-3.9.11

Step 5: Verify Installation

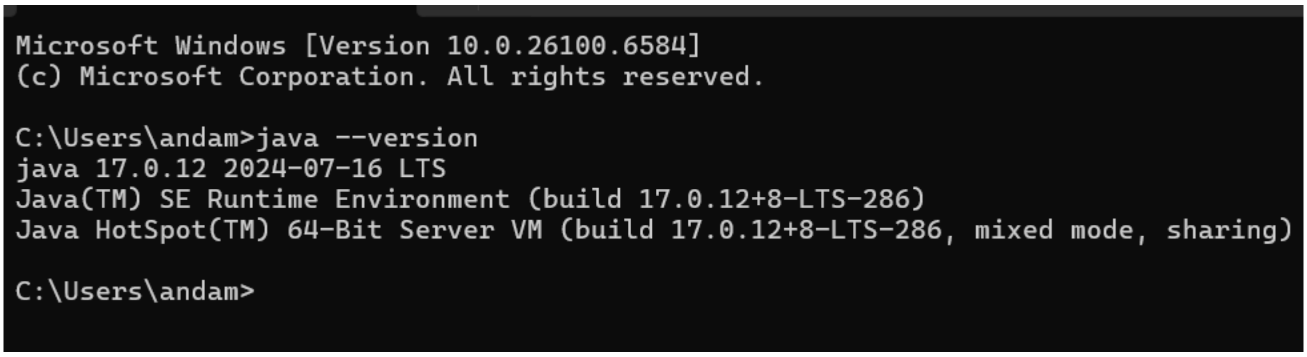


```
C:\Users\andam>mvn -v
Apache Maven 3.9.11 (3e54c93a704957b63ee3494413a2b544fd3d825b)
Maven home: C:\Program Files\apache-maven-3.9.11
Java version: 17.0.12, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17
Default locale: en_IN, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"

C:\Users\andam>
```

Task-2 JENKINS INSTALLATION

Step 1: Prerequisites

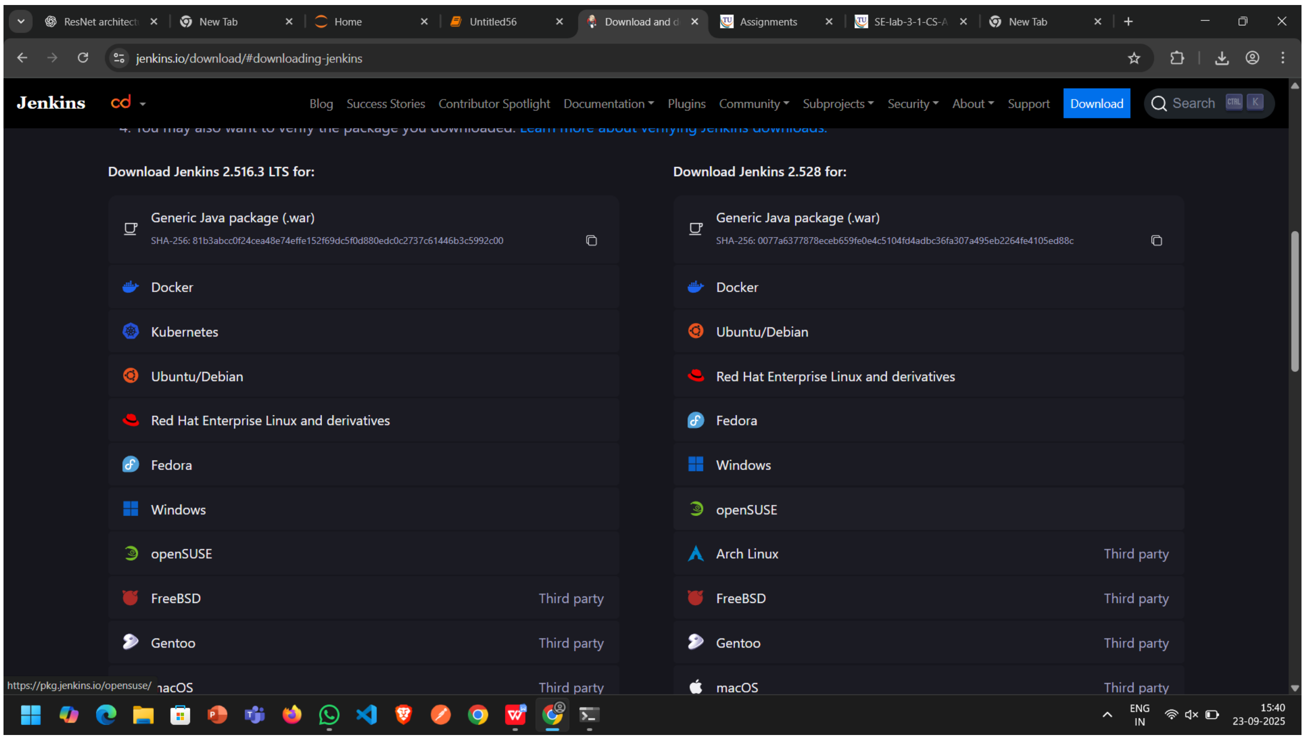


```
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\andam>java --version
java 17.0.12 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 17.0.12+8-LTS-286)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.12+8-LTS-286, mixed mode, sharing)

C:\Users\andam>
```

Step 2: Download Jenkins



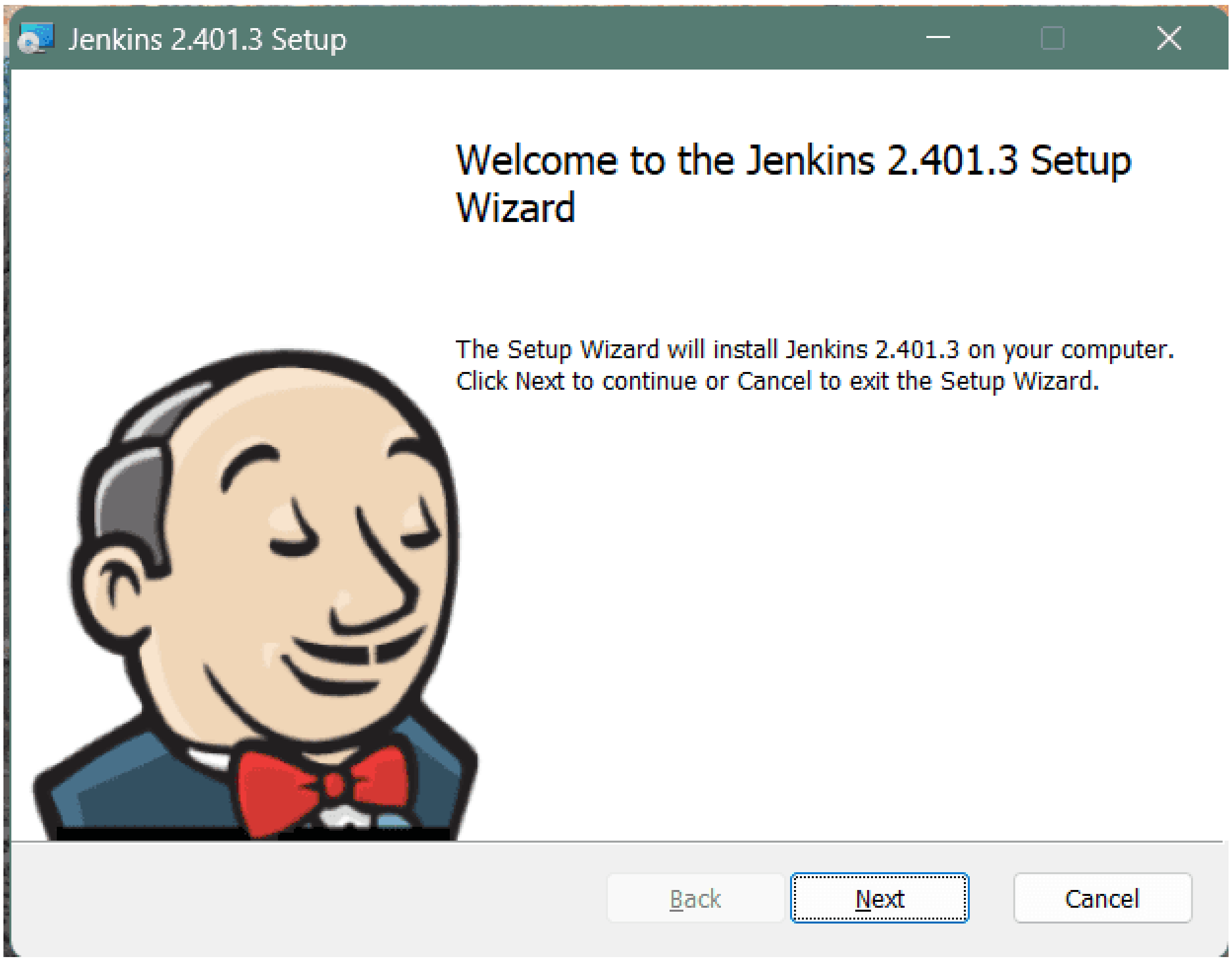
Download Jenkins 2.516.3 LTS for:

- Generic Java package (.war)
  SHA-256: 81b3abcc0f24cea48e74effe152f69dc5f0d880edc0c2737c61446b3c5992c00
- Docker
- Kubernetes
- Ubuntu/Debian
- Red Hat Enterprise Linux and derivatives
- Fedora
- Windows
- openSUSE
- FreeBSD — Third party
- Gentoo — Third party
- macOS — Third party

Download Jenkins 2.528 for:

- Generic Java package (.war)
  SHA-256: 0077a6377878eceb659fe0e4c5104fd4adbc36fa307a495eb2264fe4105ed88c
- Docker
- Ubuntu/Debian
- Red Hat Enterprise Linux and derivatives
- Fedora
- Windows
- openSUSE
- Arch Linux — Third party
- FreeBSD — Third party
- Gentoo — Third party
- macOS — Third party

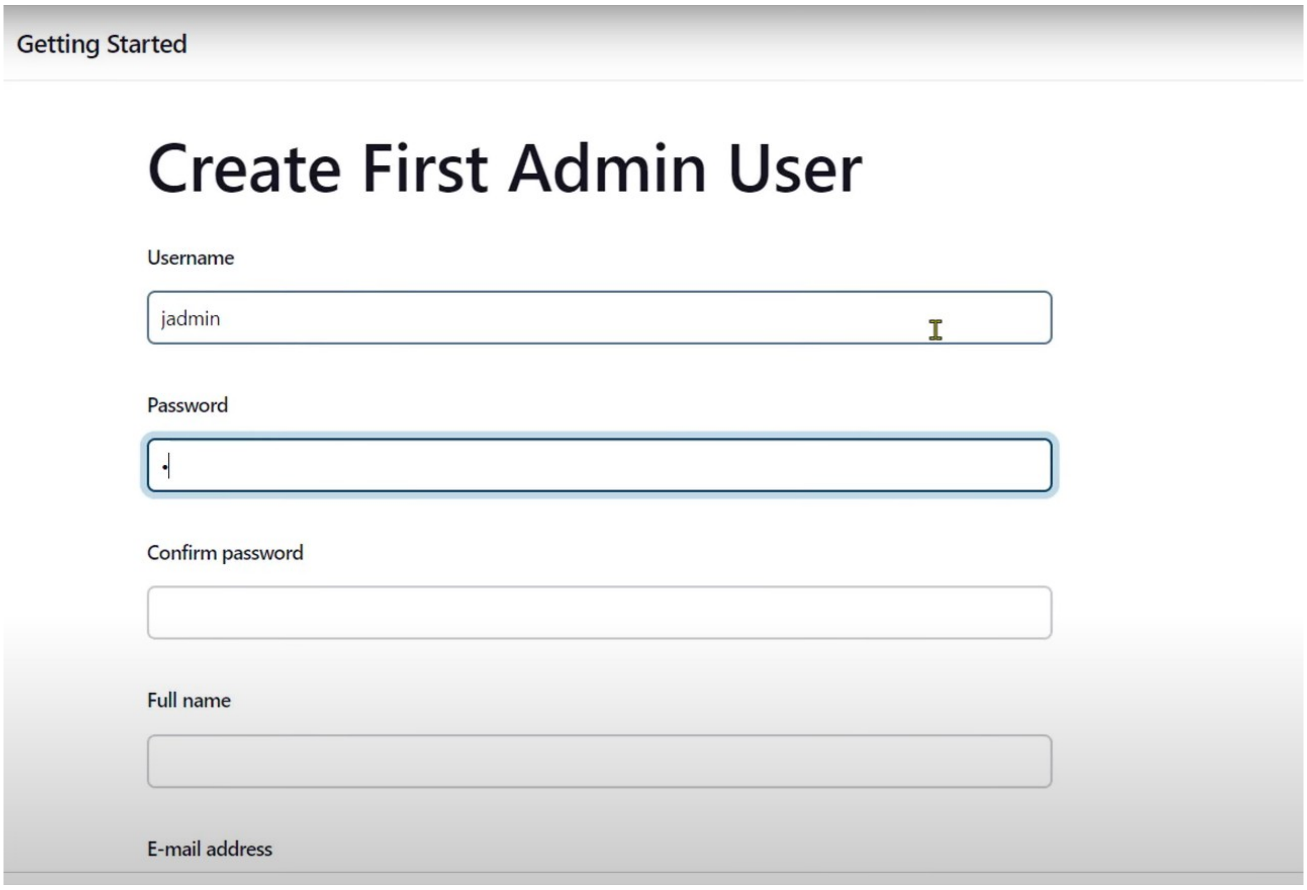
Step 3: Installation


Jenkins 2.401.3 Setup

Welcome to the Jenkins 2.401.3 Setup Wizard

The Setup Wizard will install Jenkins 2.401.3 on your computer.
Click Next to continue or Cancel to exit the Setup Wizard.

Step 4: Set User Details


Getting Started

Create First Admin User

Username

jadmin

Password

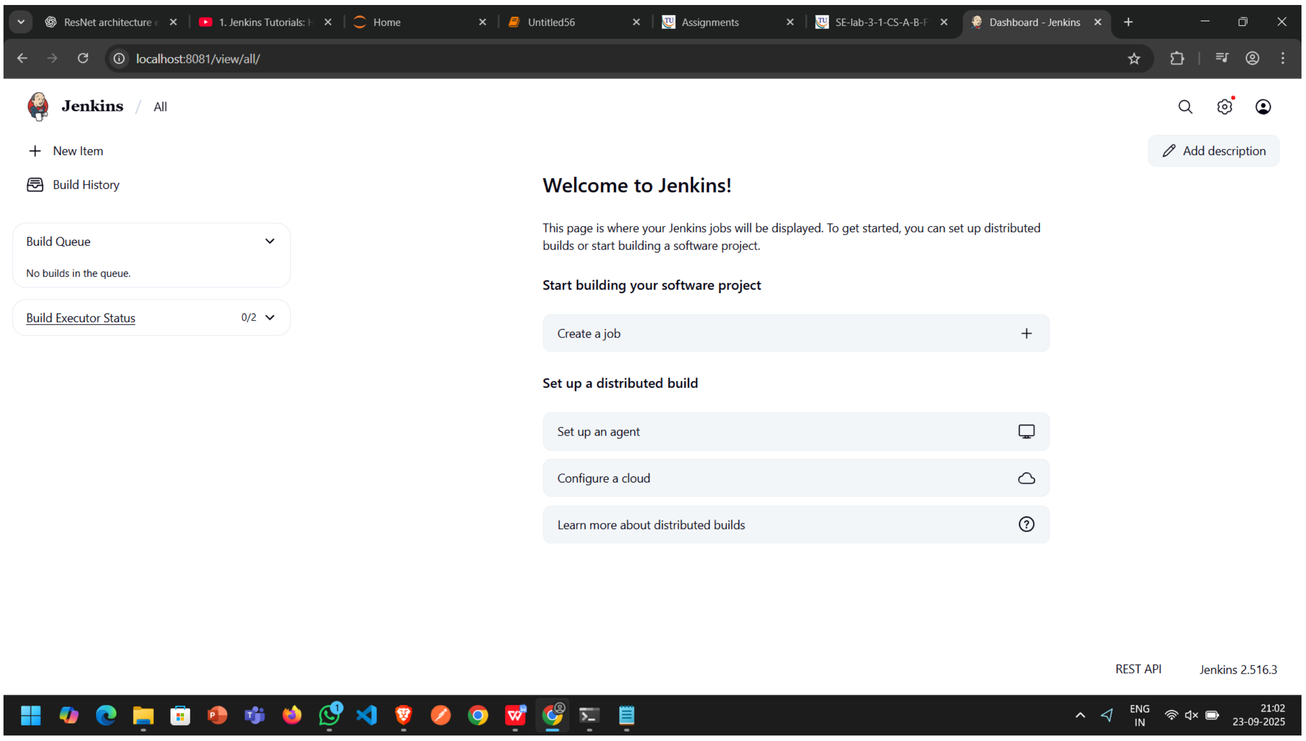
•

Confirm password

Full name

E-mail address

Step 5: Verify Installation


localhost:8081/view/all/

Jenkins / All

New Item

Build History

Build Queue

No builds in the queue.

Build Executor Status 0/2

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

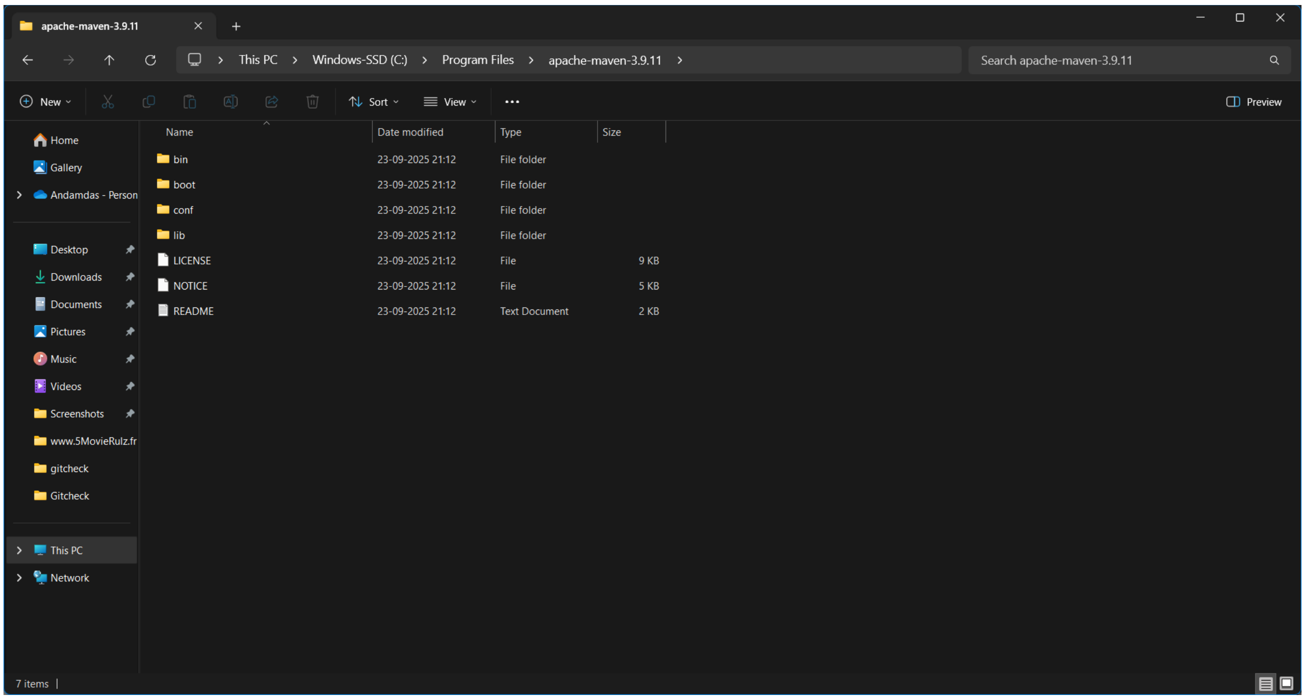
Configure a cloud

Learn more about distributed builds

REST API Jenkins 2.516.3

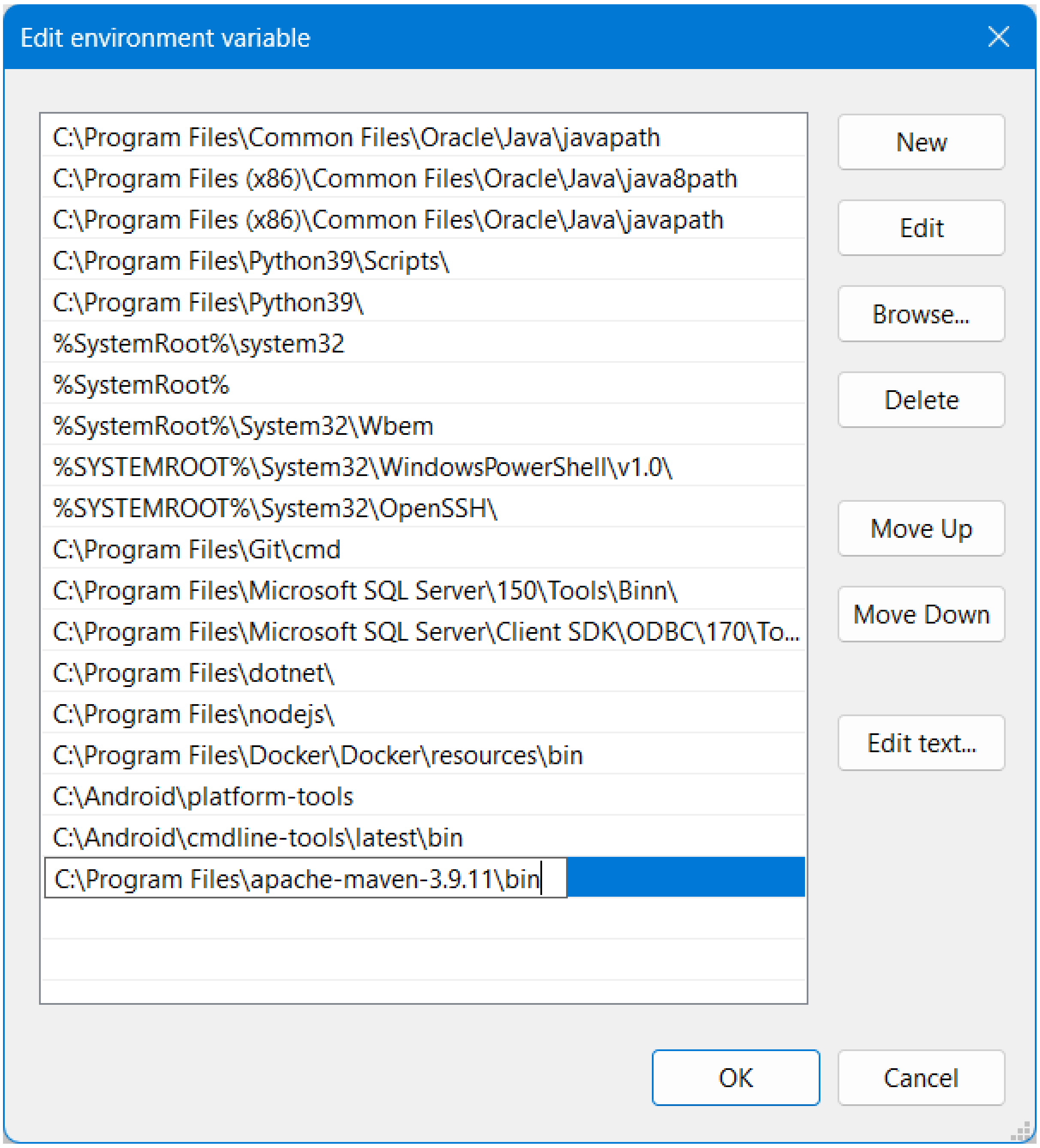
Task-3 Add Maven to Jenkins

Step 1: Copy bin path

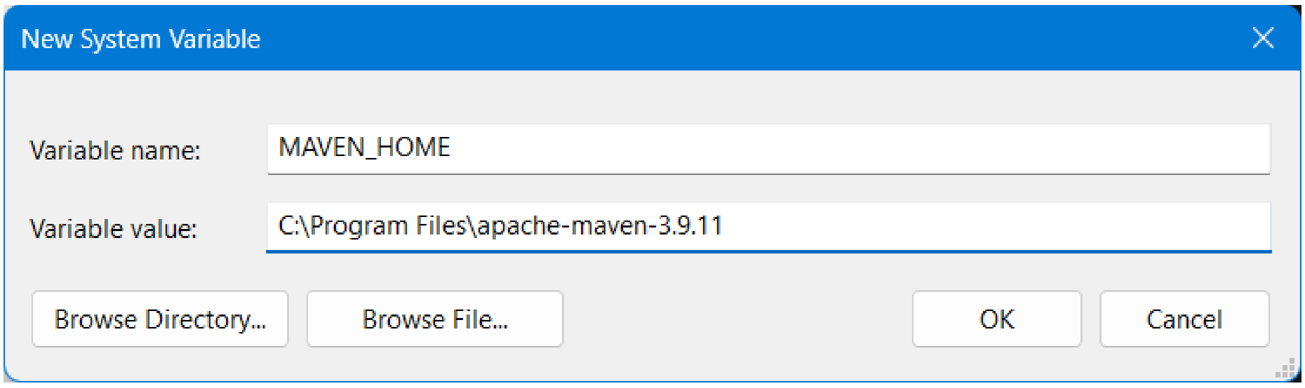


Name	Date modified	Type	Size
bin	23-09-2025 21:12	File folder	
boot	23-09-2025 21:12	File folder	
conf	23-09-2025 21:12	File folder	
lib	23-09-2025 21:12	File folder	
LICENSE	23-09-2025 21:12	File	9 KB
NOTICE	23-09-2025 21:12	File	5 KB
README	23-09-2025 21:12	Text Document	2 KB

Step 2: Set Environment Variables



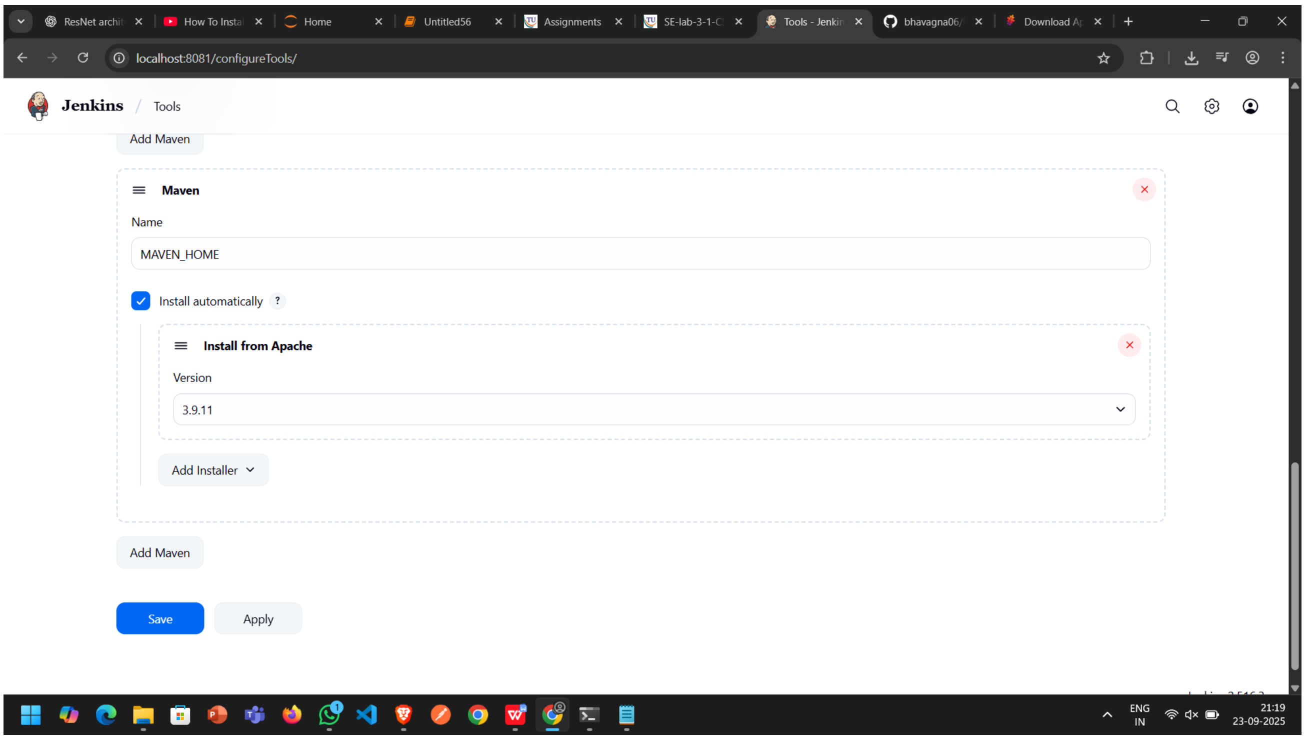
```
C:\Program Files\Common Files\Oracle\Java\javapath
C:\Program Files (x86)\Common Files\Oracle\Java\java8path
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
C:\Program Files\Python39\Scripts\
C:\Program Files\Python39\
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Git\cmd
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\To...
C:\Program Files\dotnet\
C:\Program Files\nodejs\
C:\Program Files\Docker\Docker\resources\bin
C:\Android\platform-tools
C:\Android\cmdline-tools\latest\bin
C:\Program Files\apache-maven-3.9.11\bin
```



Variable name: MAVEN_HOME

Variable value: C:\Program Files\apache-maven-3.9.11

Step 3: Set maven in Jenkins Configure Tools



Maven

Name: MAVEN_HOME

☑ Install automatically

Install from Apache

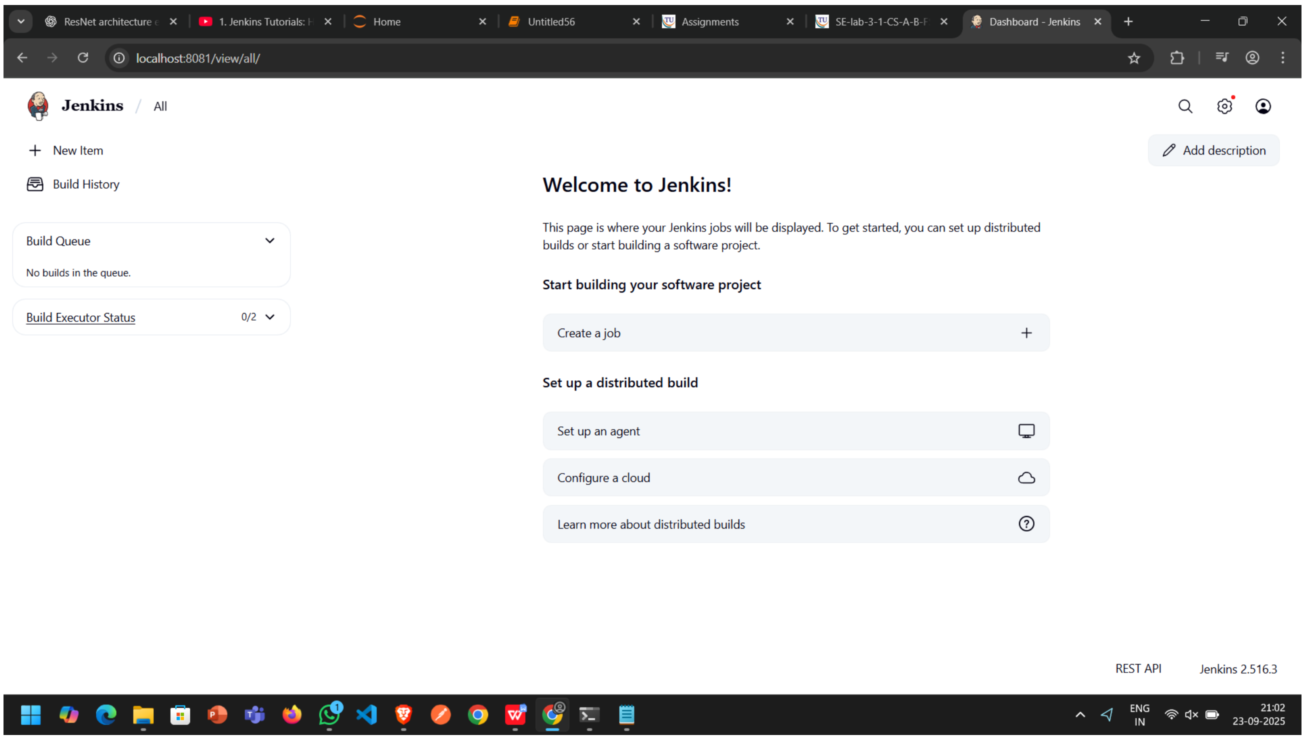
Version: 3.9.11

Apply and Save

Task-4 Maven java Automation

I. Maven Java Automation Steps:

Step 1: Open Jenkins (localhost:8081)



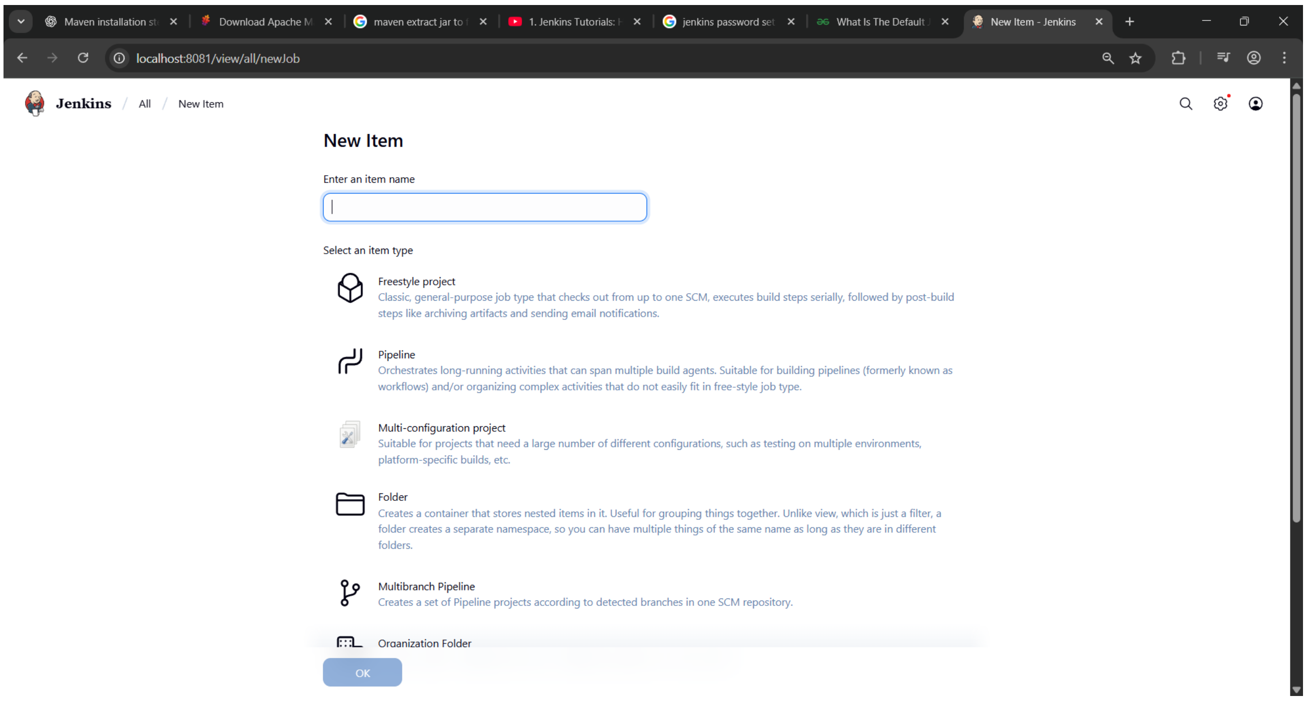
Welcome to Jenkins!

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

Set up a distributed build

Click on "New Item" (left side menu)



Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

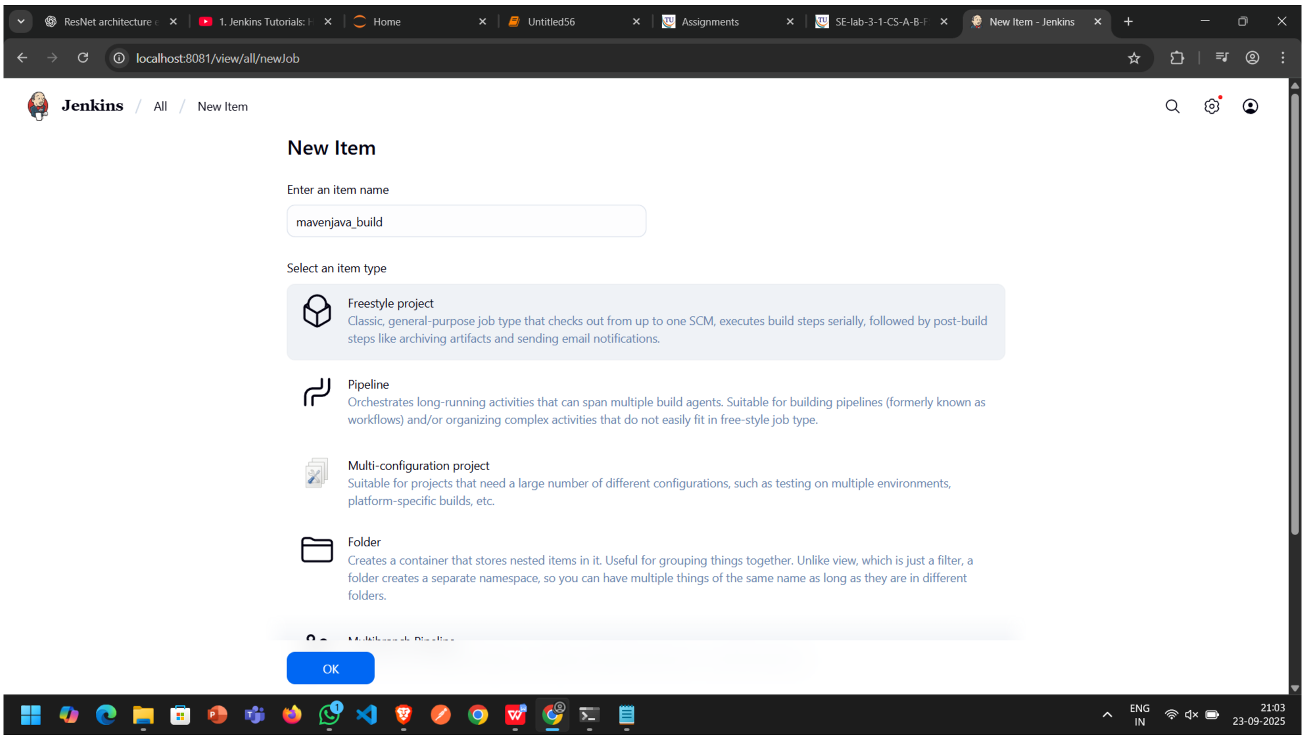
Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder

Step 2: Create Freestyle Project (e.g., MavenJava_Build)

Enter project name (e.g., MavenJava_Build)

Click "OK"



localhost:8081/view/all/newJob

Jenkins / All / New Item

New Item

Enter an item name

mavenjava_build

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

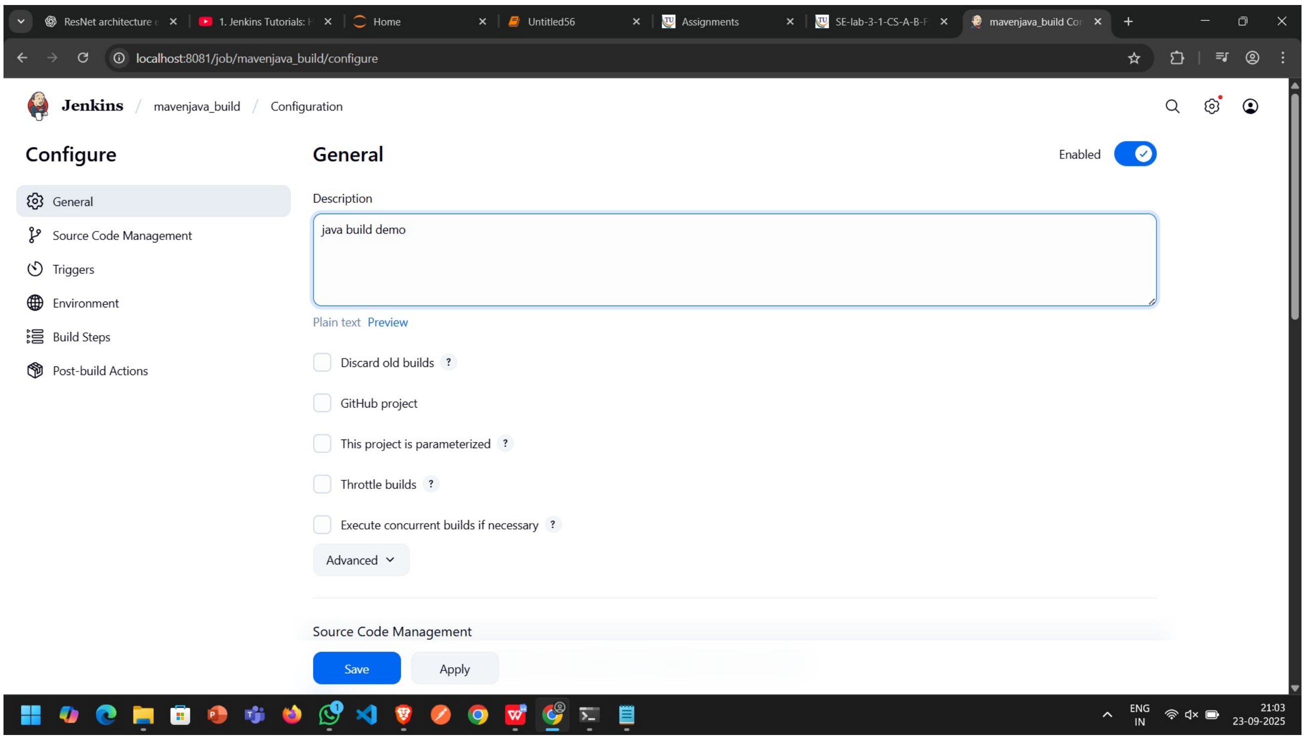
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Configure the project:

Description: "Java Build demo"



localhost:8081/job/mavenjava_build/configure

Jenkins / mavenjava_build / Configuration

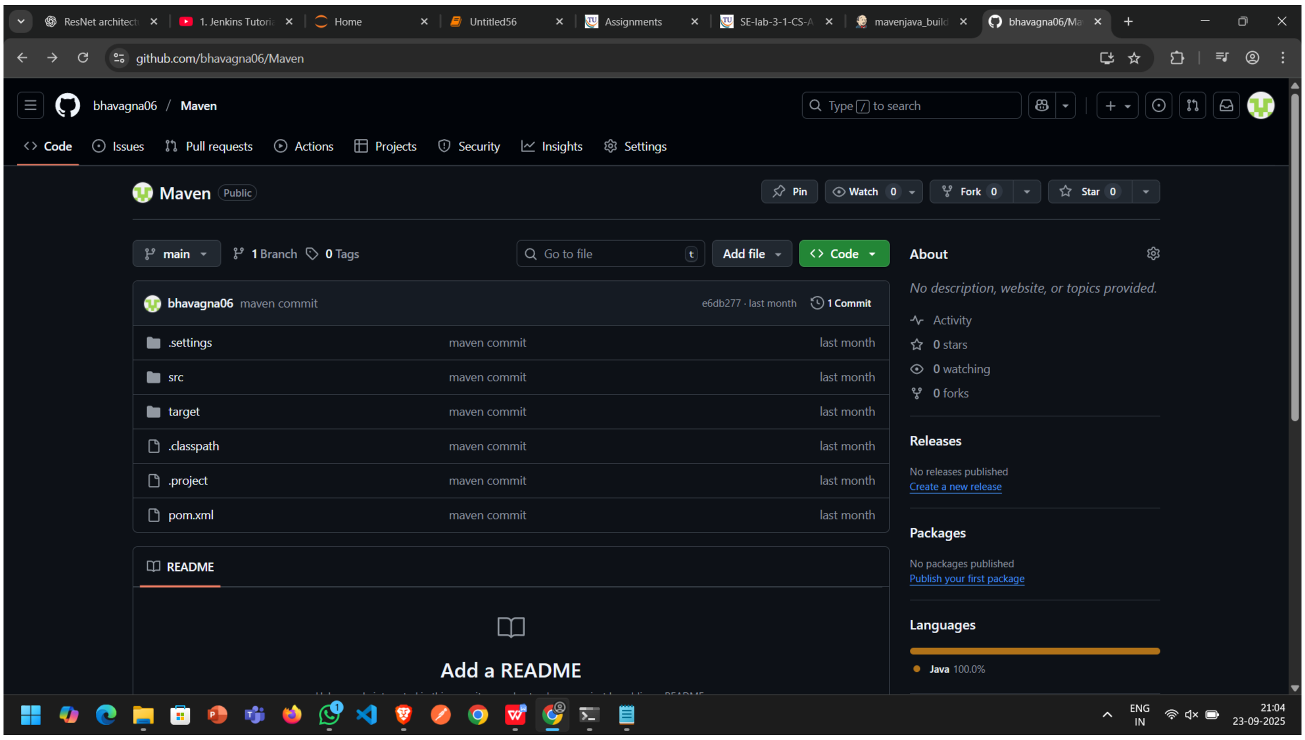
Configure

General | Enabled

Description

java build demo

Source Code Management: Git repository URL: [https://github.com/bhavagna06/Maven.git]


github.com/bhavagna06/Maven

bhavagna06 / Maven

Maven Public

main · 1 Branch · 0 Tags

bhavagna06 maven commit · e6db277 · last month · 1 Commit

.settings	maven commit	last month
src	maven commit	last month
target	maven commit	last month
.classpath	maven commit	last month
.project	maven commit	last month
pom.xml	maven commit	last month

README

Add a README

About

No description, website, or topics provided.

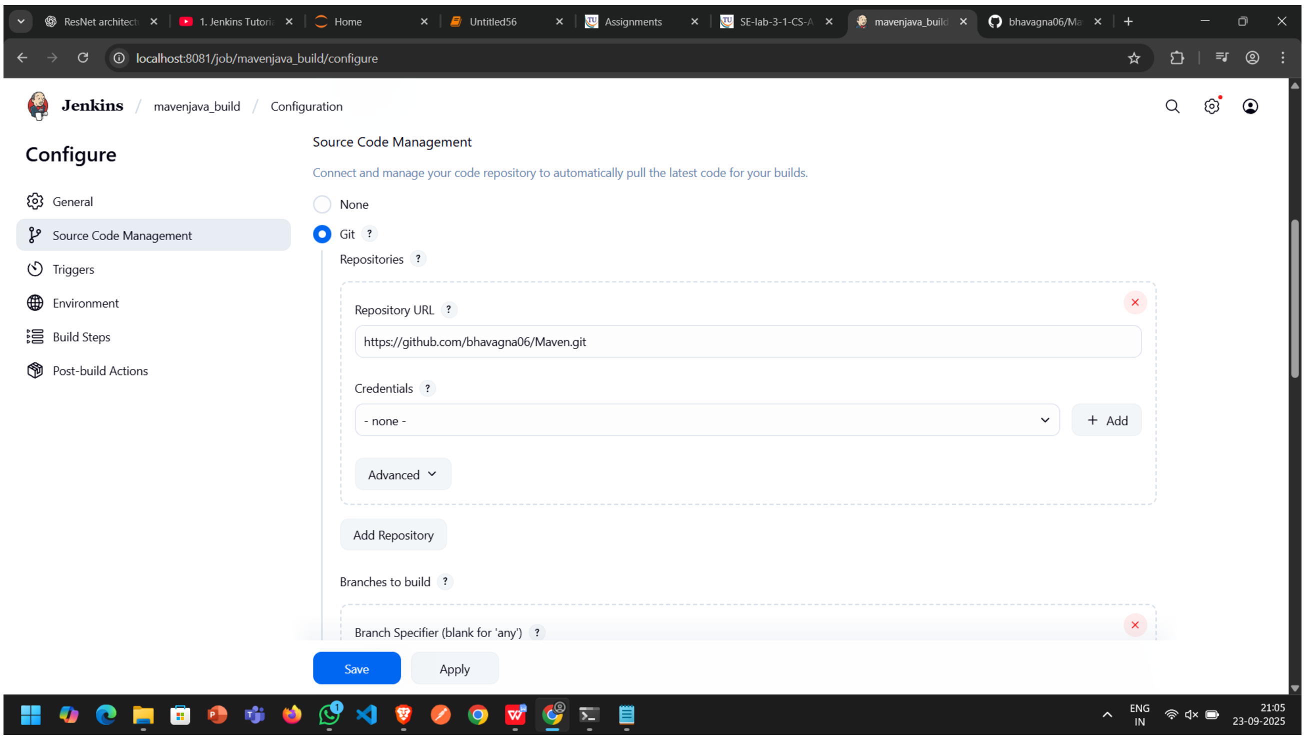
Activity · 0 stars · 0 watching · 0 forks

Releases: No releases published

Packages: No packages published

Languages: Java 100.0%

Branches to build: */Main   or */master


localhost:8081/job/mavenjava_build/configure

Jenkins / mavenjava_build / Configuration

Configure

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

Repositories

Repository URL: https://github.com/bhavagna06/Maven.git

Credentials: - none -

Advanced

Add Repository

Branches to build

Branch Specifier (blank for 'any')

Save Apply

Build Steps: Add Build Step -> "Invoke top-level Maven targets"

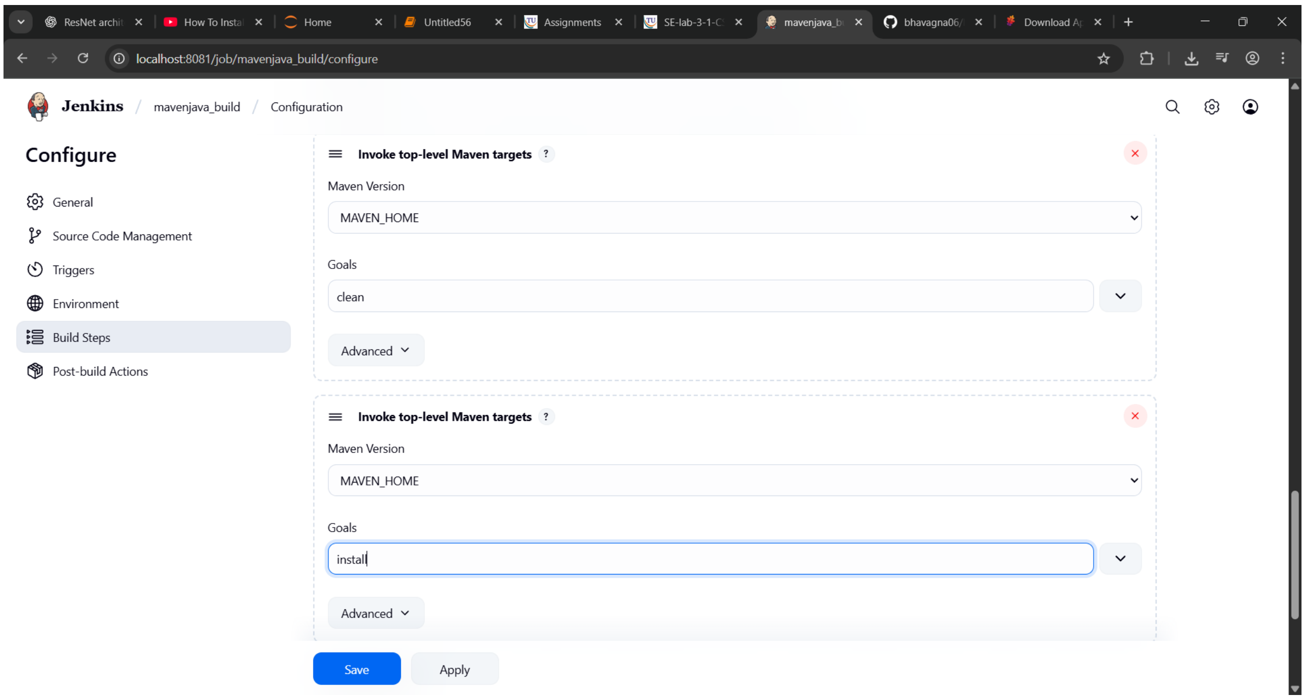
Maven version: MAVEN_HOME

Goals: clean

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: install



Invoke top-level Maven targets

Maven Version: MAVEN_HOME

Goals: clean

Invoke top-level Maven targets

Maven Version: MAVEN_HOME

Goals: install

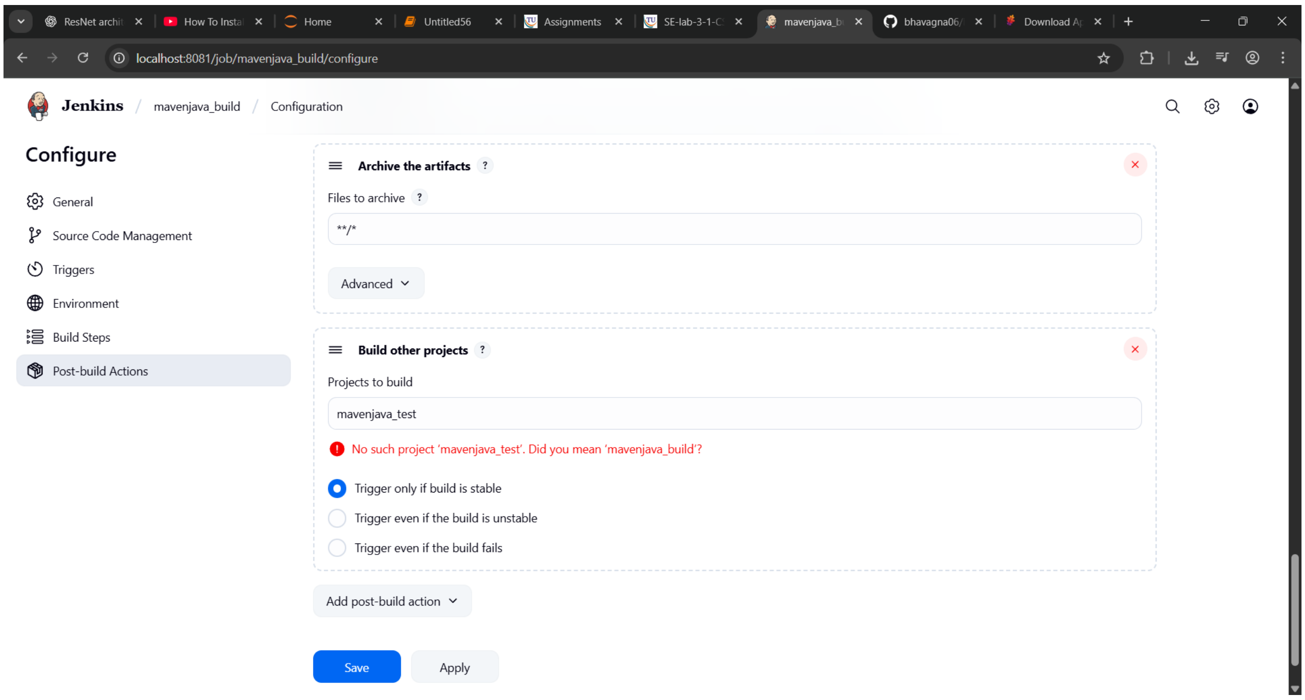
Post-build Actions: Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*

Add Post Build Action -> "Build other projects"

Projects to build: MavenJava_Test

Trigger: Only if build is stable


Jenkins / mavenjava_build / Configuration

Configure

Archive the artifacts

Files to archive: **/*

Build other projects

Projects to build: mavenjava_test

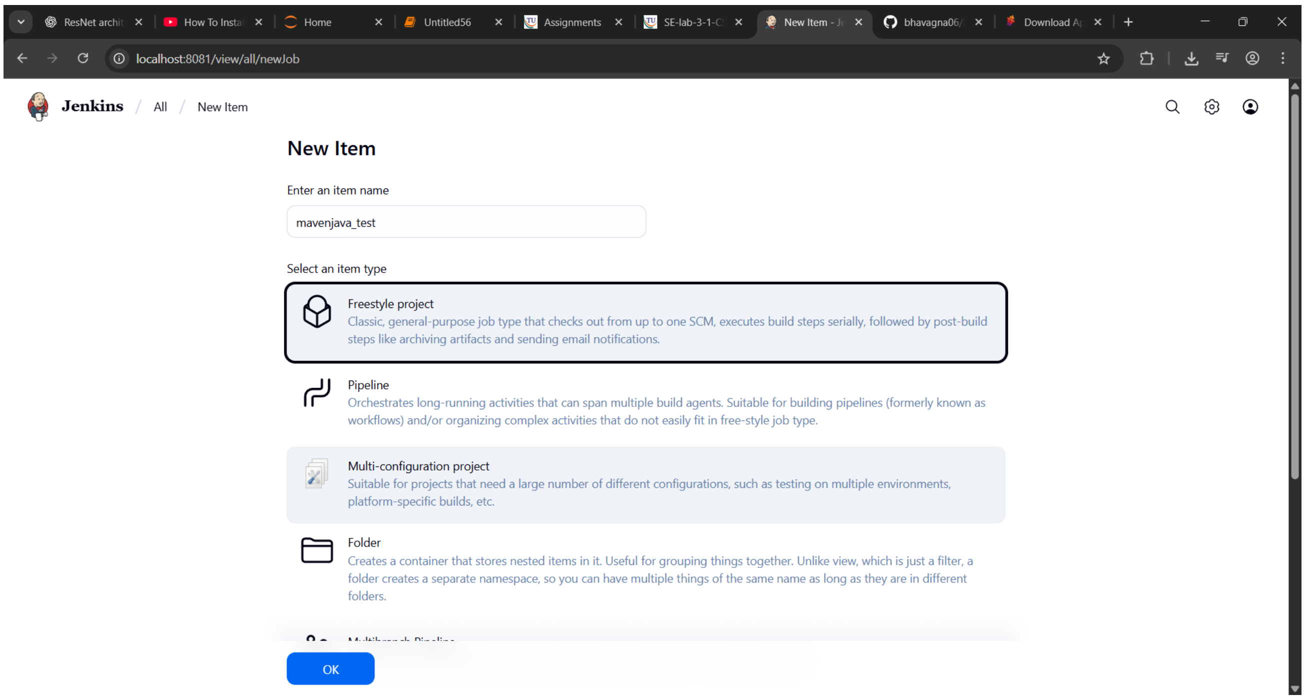
No such project 'mavenjava_test'. Did you mean 'mavenjava_build'?

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

Apply and Save

Step 3: Create Freestyle Project (e.g., MavenJava_Test)

Enter project name (e.g., MavenJava_Test) ├── Click "OK"


Jenkins / All / New Item

New Item

Enter an item name: mavenjava_test

Select an item type

[✓ Freestyle project]
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

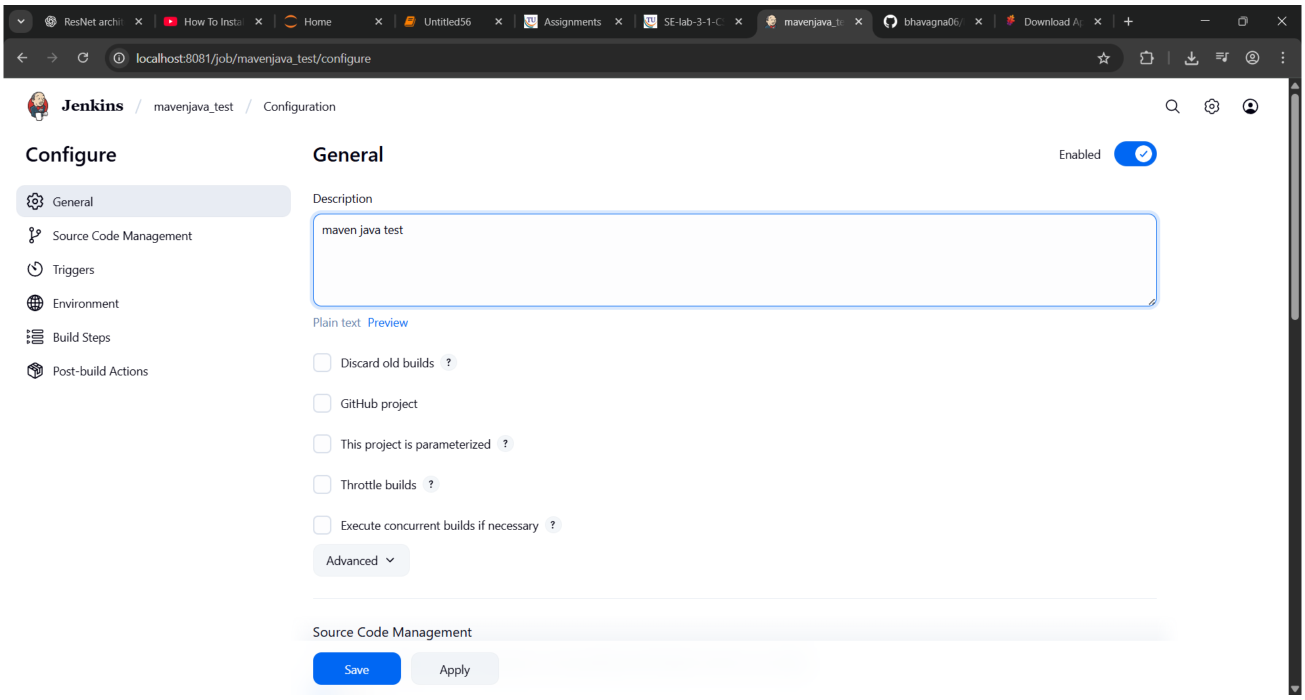
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Configure the project:

Description: "Test demo"



Jenkins / mavenjava_test / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

General — Enabled

Description: maven java test

Plain text Preview

- [ ] Discard old builds
- [ ] GitHub project
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary

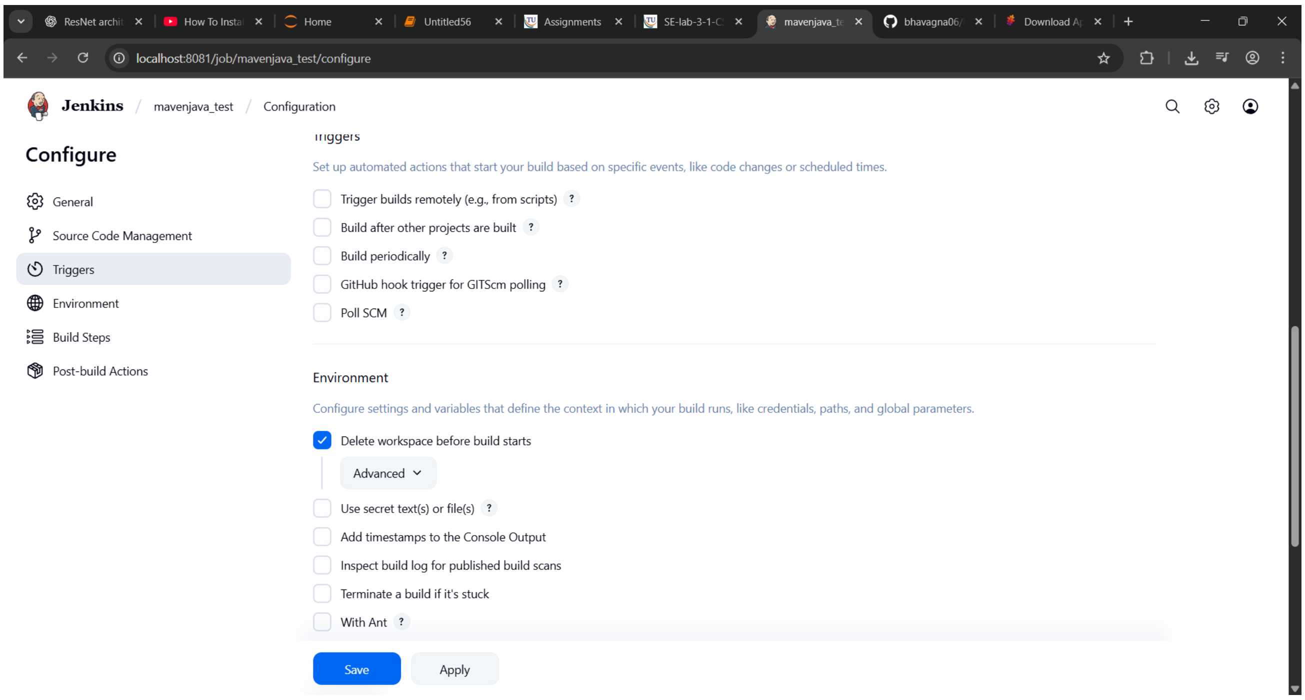
Advanced

Source Code Management

Save Apply

Build Environment:

Check: "Delete the workspace before build starts"



Jenkins / mavenjava_test / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [x] Delete workspace before build starts
  - Advanced
- [ ] 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

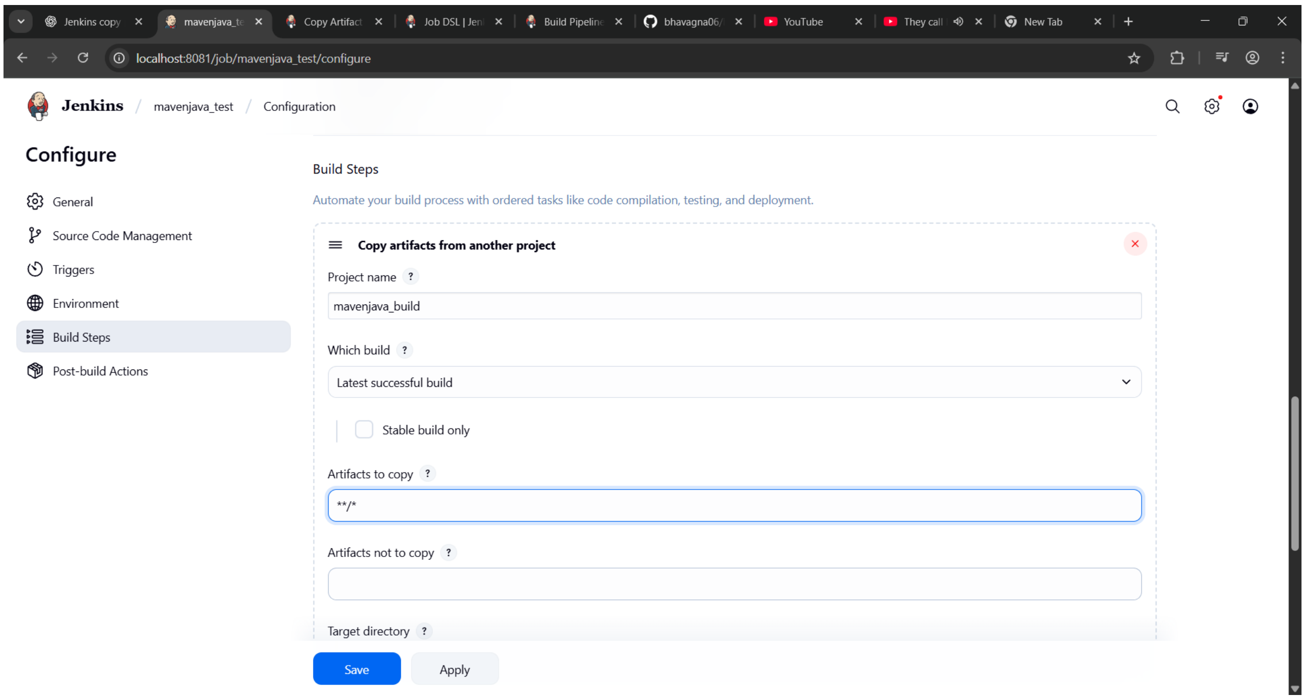
Save Apply

Add Build Step -> "Copy artifacts from another project"

Project name: MavenJava_Build

Build: Stable build only

Artifacts to copy: **/*


Jenkins / mavenjava_test / Configuration

Configure

Build Steps

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

Copy artifacts from another project

Project name: mavenjava_build

Which build: Latest successful build

Stable build only

Artifacts to copy: **/*

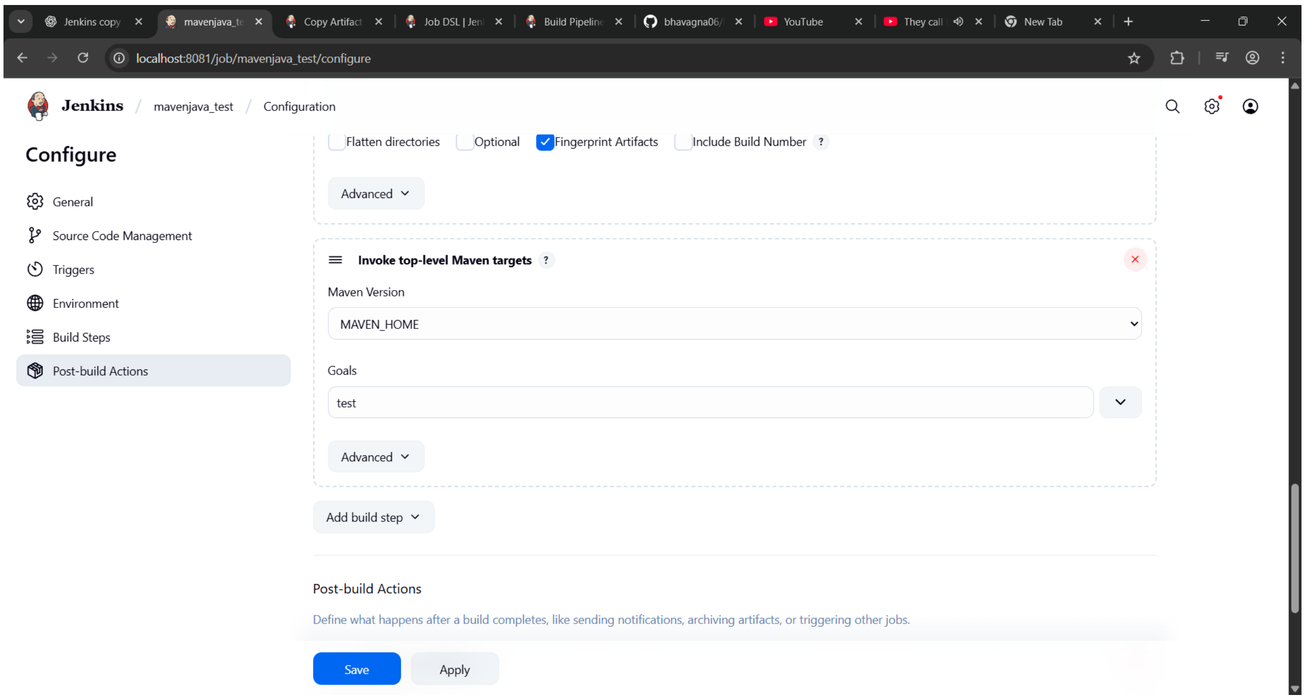
Artifacts not to copy:

Target directory

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: test


Jenkins / mavenjava_test / Configuration

Configure

Flatten directories | Optional | [✓ Fingerprint Artifacts] | Include Build Number

Invoke top-level Maven targets

Maven Version: MAVEN_HOME

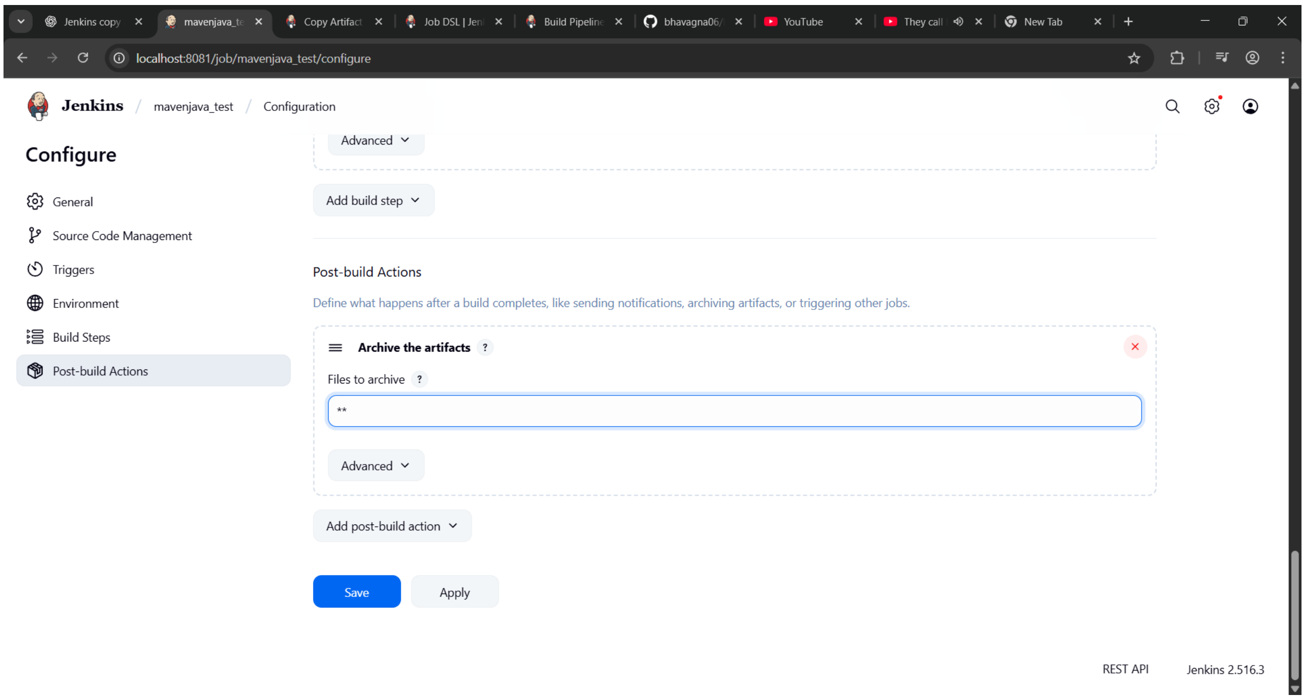
Goals: test

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Post-build Actions: Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*


localhost:8081/job/mavenjava_test/configure

Jenkins / mavenjava_test / Configuration

Configure

General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions

Advanced

Add build step

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Archive the artifacts

Files to archive: **

Advanced

Add post-build action

Save   Apply

REST API   Jenkins 2.516.3

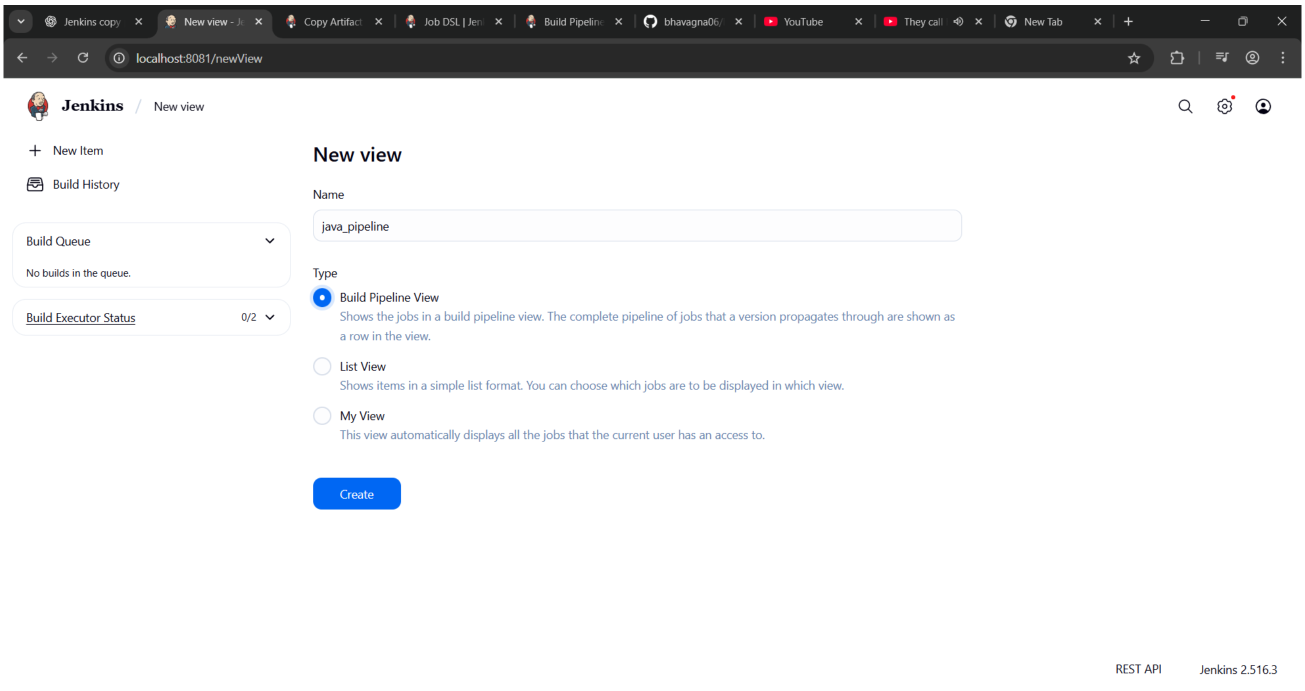
Apply and Save

Step 4: Create Pipeline View for Maven Java project

Click "+" beside "All" on the dashboard

Enter name: Java_Pipeline

Select "Build pipeline view"


localhost:8081/newView

Jenkins / New view

New Item
Build History

Build Queue
No builds in the queue.

Build Executor Status 0/2

New view

Name: java_pipeline

Type

[✓ Build Pipeline View]
Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.

List View
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

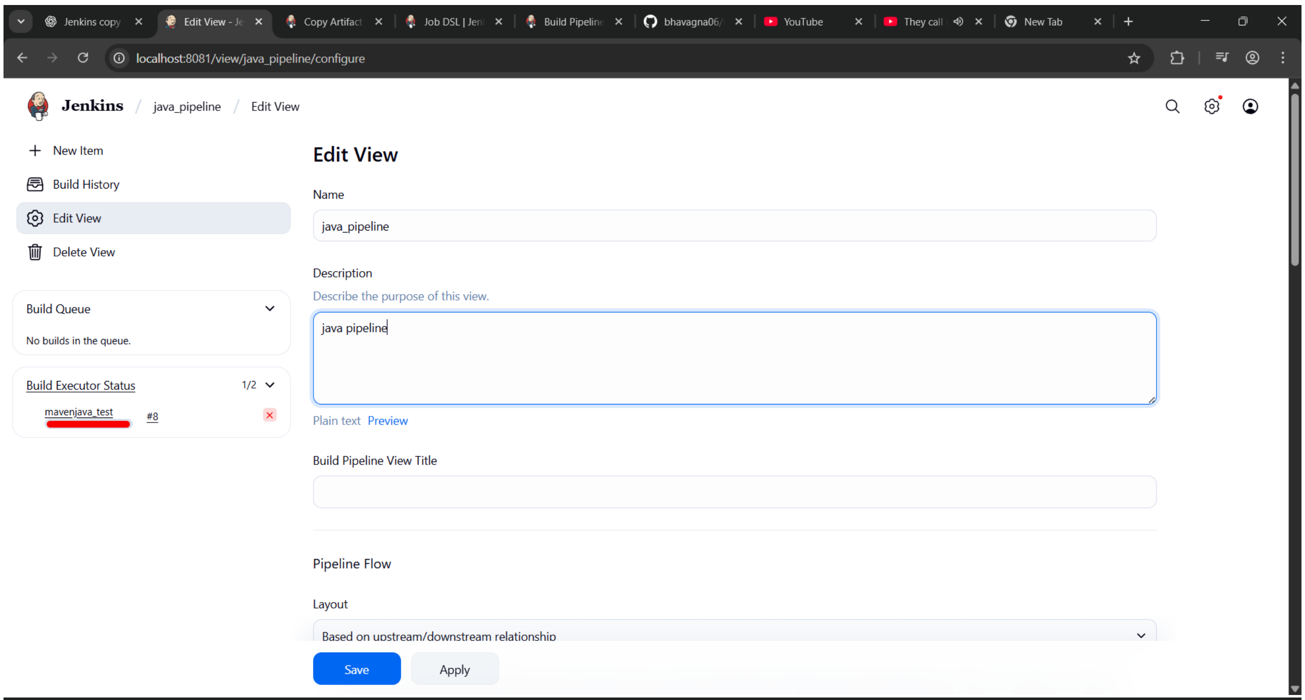
My View
This view automatically displays all the jobs that the current user has an access to.

Create

REST API   Jenkins 2.516.3

create Pipeline Flow: Layout: Based on upstream/downstream relationship

Initial job: MavenJava_Build


localhost:8081/view/java_pipeline/configure

Jenkins / java_pipeline / Edit View

Edit View

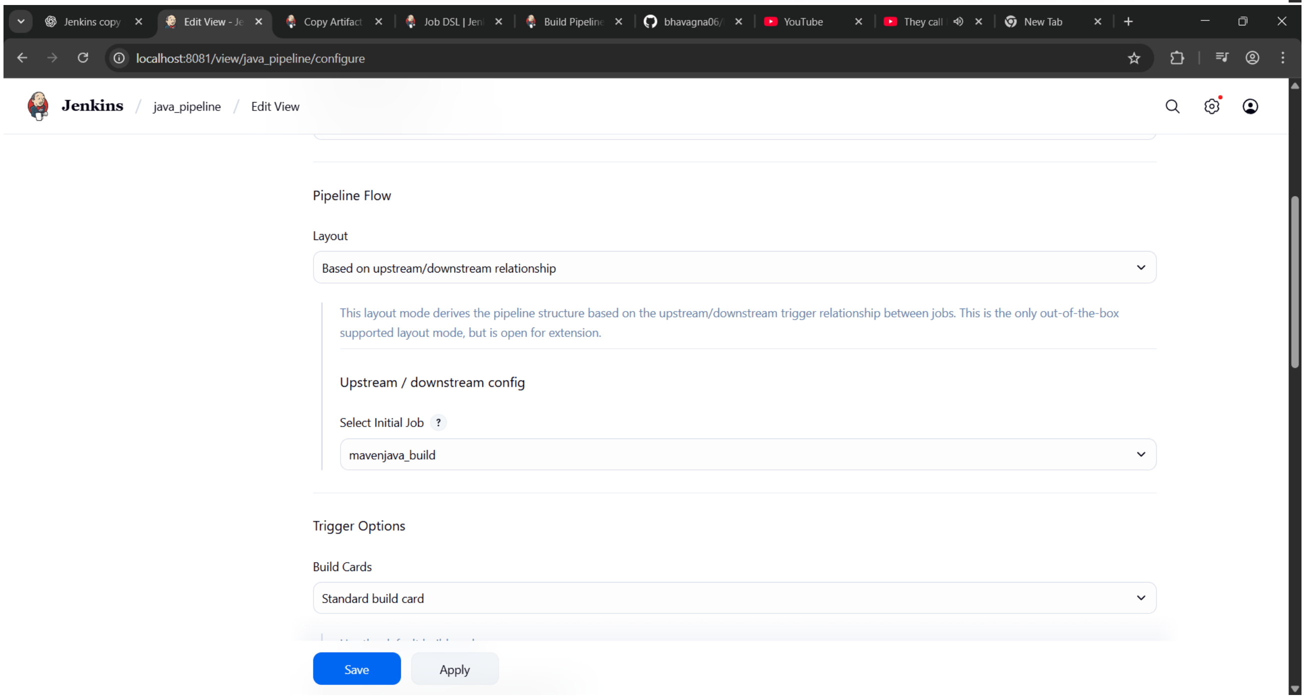
Name: java_pipeline

Description: java pipeline

Build Pipeline View Title:

Pipeline Flow

Layout: Based on upstream/downstream relationship


localhost:8081/view/java_pipeline/configure

Pipeline Flow

Layout: Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job: mavenjava_build

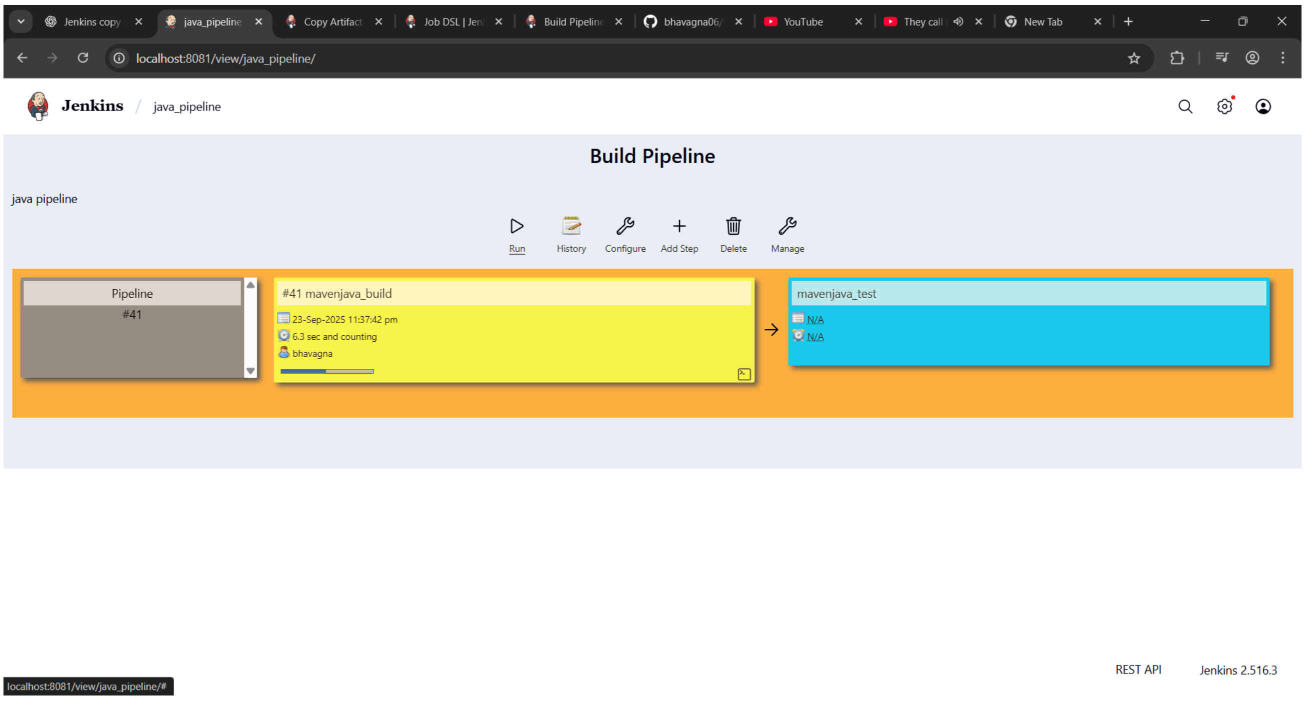
Trigger Options

Build Cards: Standard build card

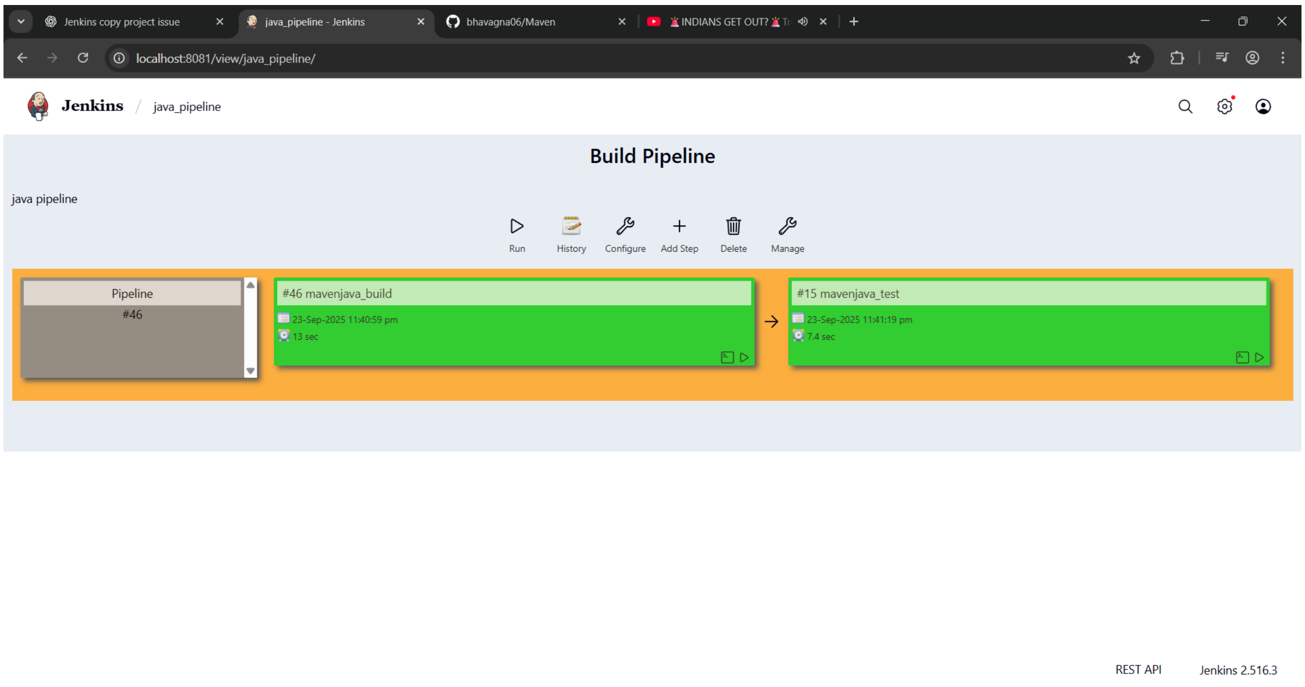
Apply and Save OK

Step 5: Run the Pipeline and Check Output

Click on the trigger to run the pipeline



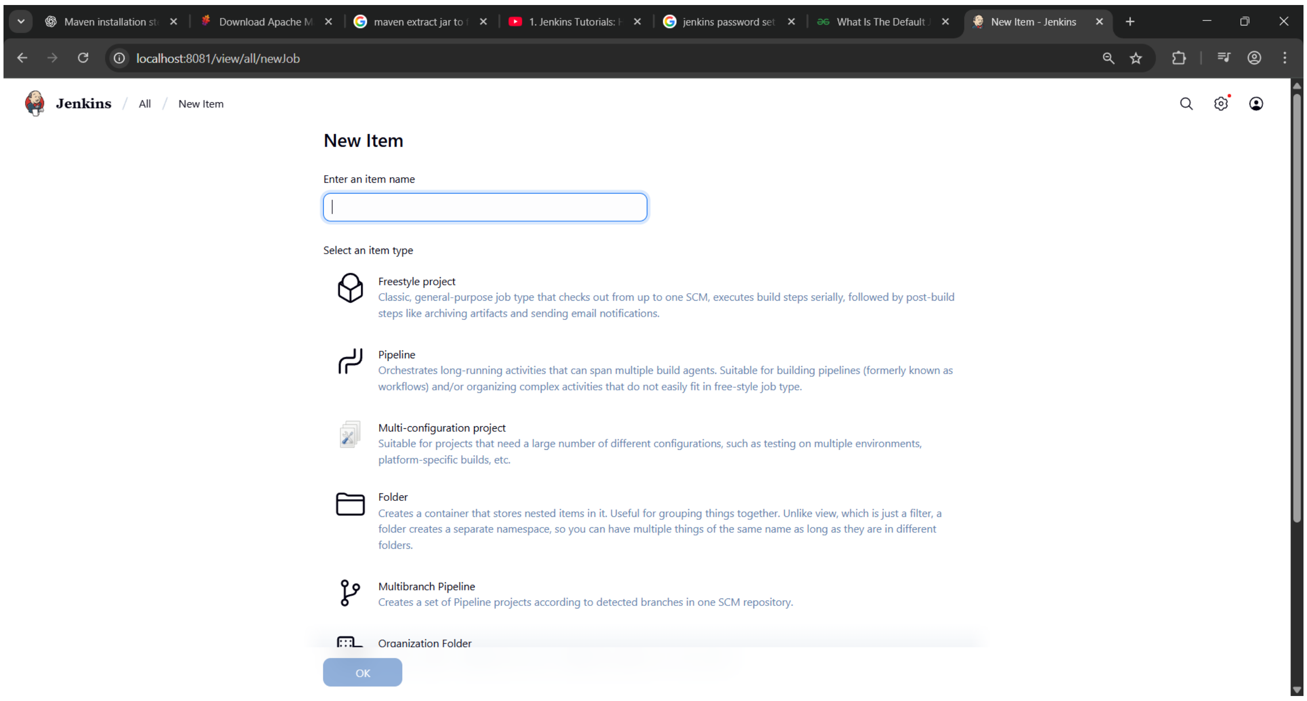
click on the small black box to open the console to check if the build is success



<u>II. Maven Web Automation Steps:</u>

Step 1: Open Jenkins (localhost:8081)

├── Click on "New Item" (left side menu)



New Item

Enter an item name

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

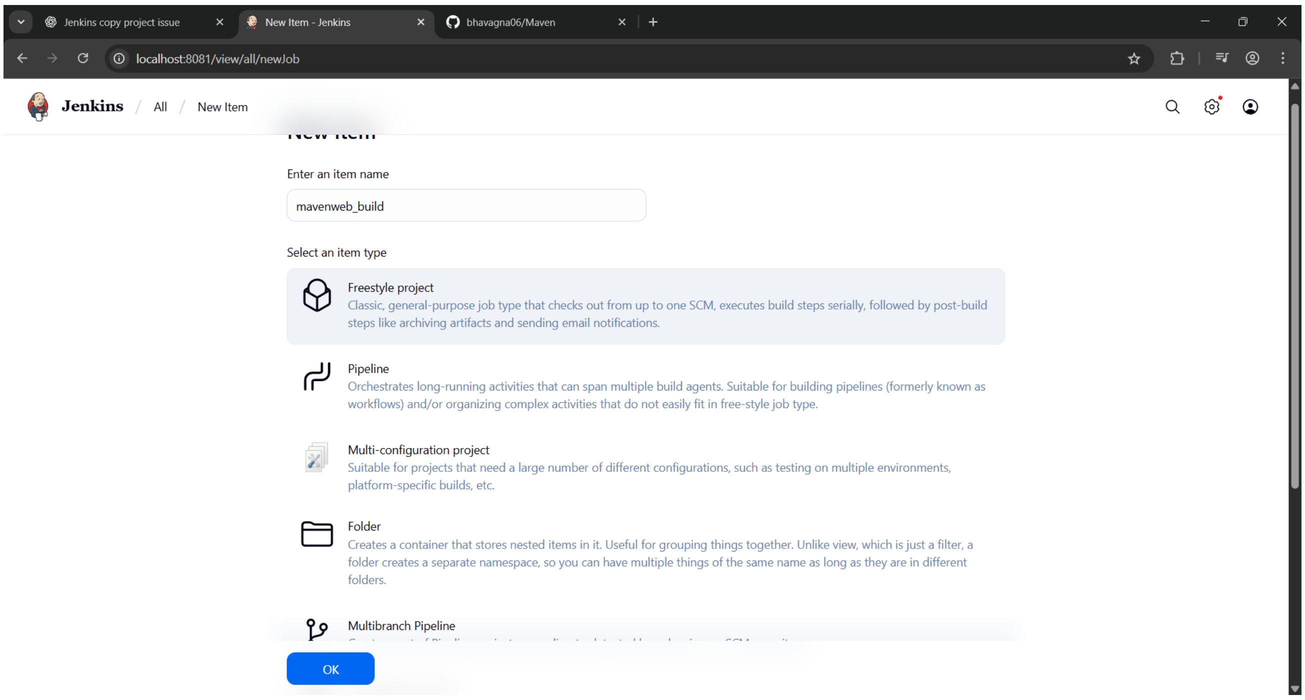
Organization Folder

OK

Step 2: Create Freestyle Project (e.g., MavenWeb_Build)

├── Enter project name (e.g., MavenWeb_Build)

├── Click "OK"



Enter an item name

mavenweb_build

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

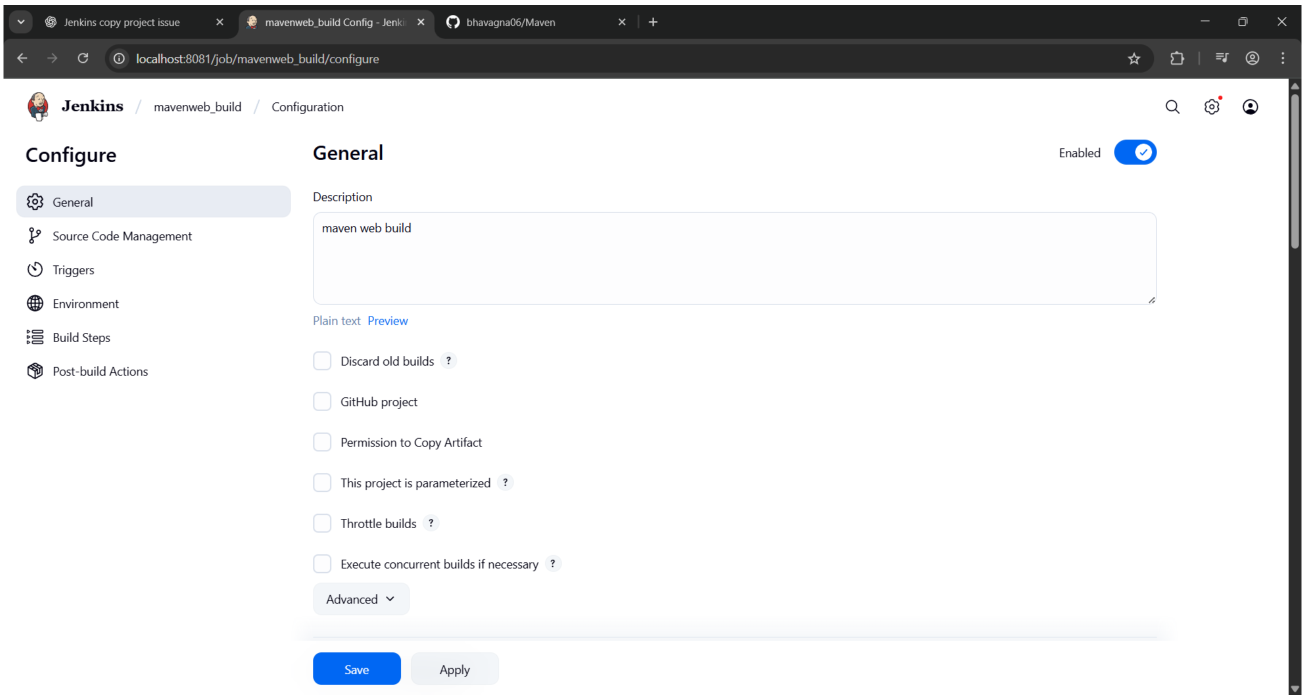
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline

OK

Configure the project:

├── **Description**: "Web Build demo"


localhost:8081/job/mavenweb_build/configure

Jenkins / mavenweb_build / Configuration

Configure

General

Enabled

Description

maven web build

Discard old builds

GitHub project

Permission to Copy Artifact

This project is parameterized

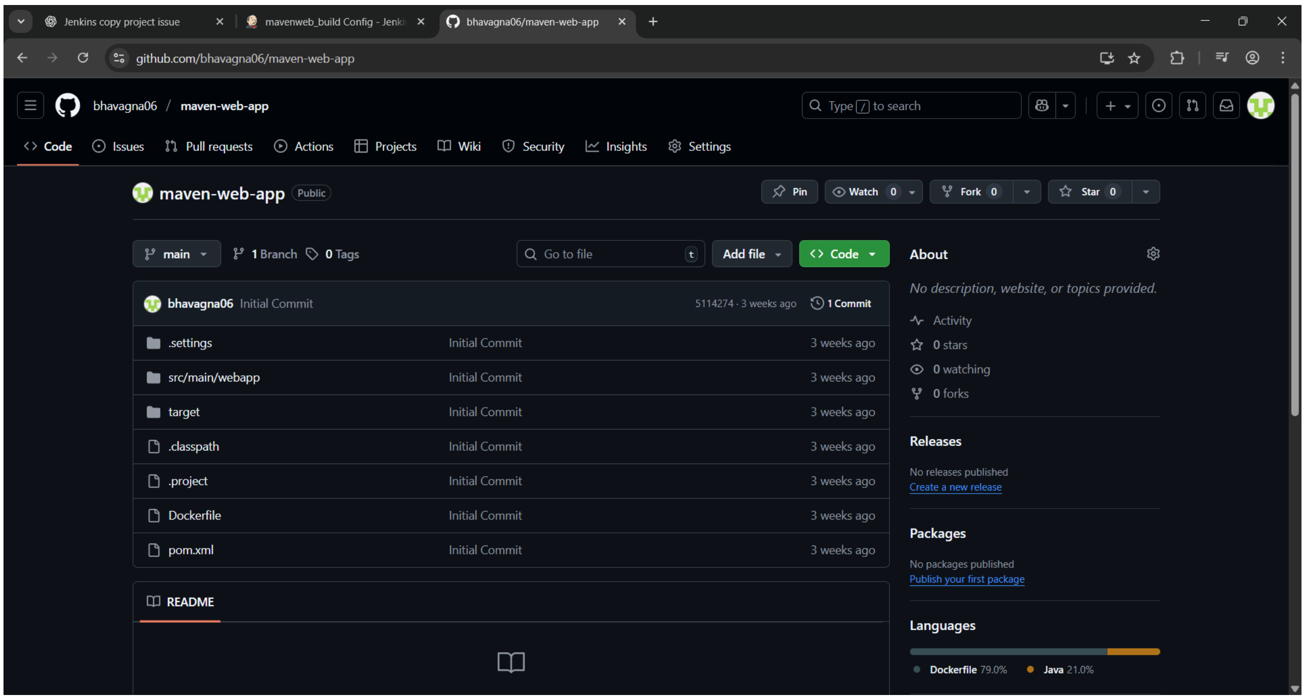
Throttle builds

Execute concurrent builds if necessary

Source Code Management:

└── Git repository URL: [https://github.com/bhavagna06/maven-web-app.git]

├── *Branches to build*: */Main or master


github.com/bhavagna06/maven-web-app

bhavagna06 / maven-web-app

maven-web-app Public

main · 1 Branch · 0 Tags

bhavagna06 Initial Commit 5114274 · 3 weeks ago 1 Commit

.settings	Initial Commit	3 weeks ago
src/main/webapp	Initial Commit	3 weeks ago
target	Initial Commit	3 weeks ago
.classpath	Initial Commit	3 weeks ago
.project	Initial Commit	3 weeks ago
Dockerfile	Initial Commit	3 weeks ago
pom.xml	Initial Commit	3 weeks ago

About

No description, website, or topics provided.

Activity
0 stars
0 watching
0 forks

Releases

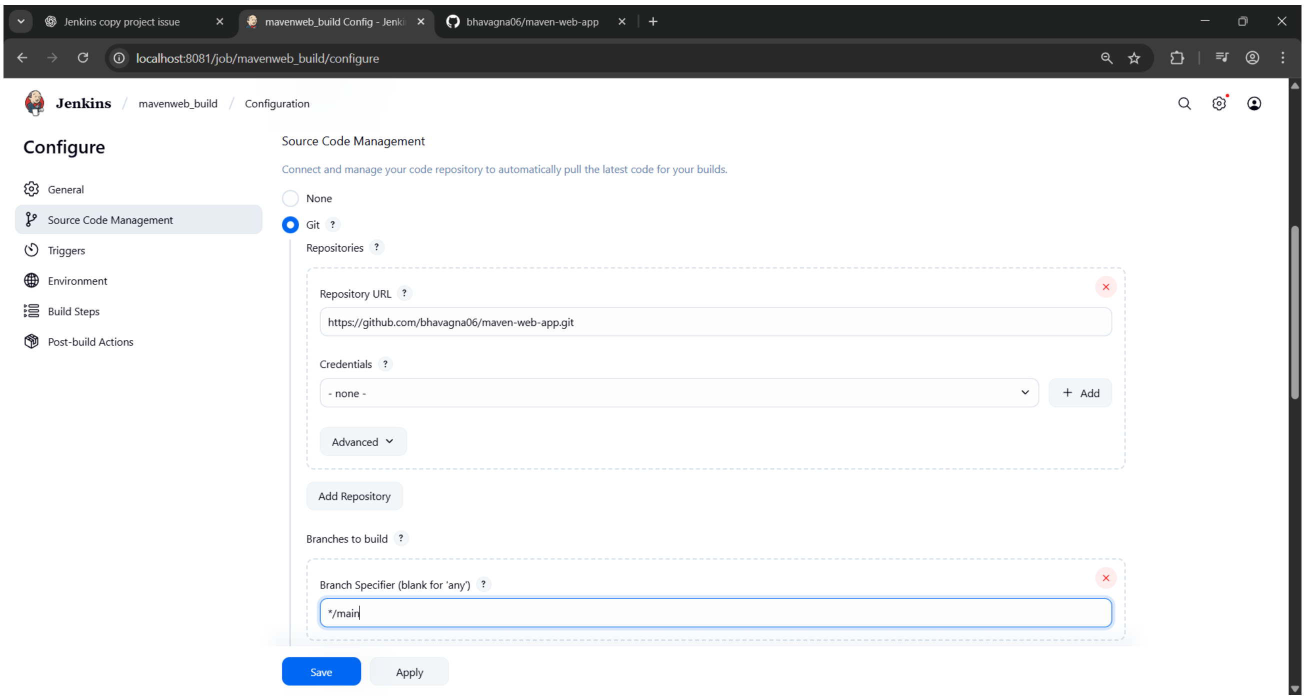
No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

Dockerfile 79.0% Java 21.0%



Jenkins copy project issue
mavenweb_build Config - Jenkins
bhavagna06/maven-web-app
localhost:8081/job/mavenweb_build/configure
Jenkins / mavenweb_build / Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Save
Apply

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

Repositories

Repository URL: https://github.com/bhavagna06/maven-web-app.git

Credentials: - none -

Advanced

Add Repository

Branches to build

Branch Specifier (blank for 'any'): */main

Build Steps:

├── **Add Build Step** -> "Invoke top-level Maven targets"

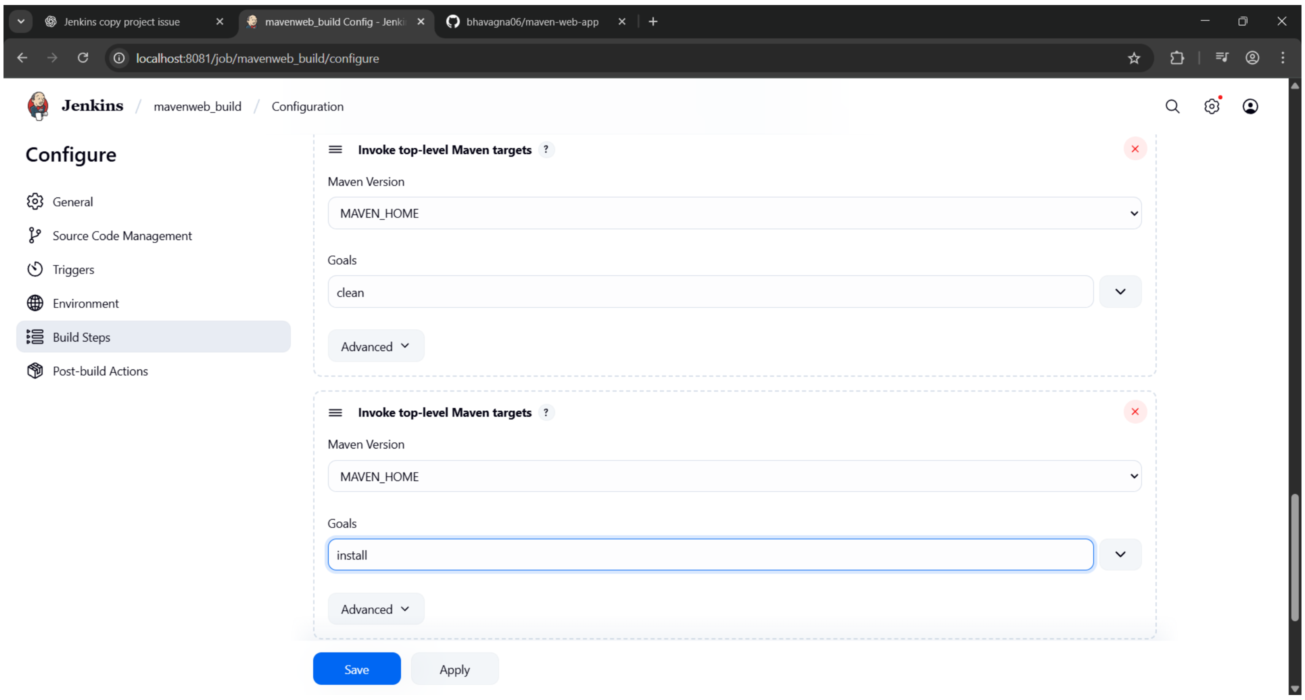
└── Maven version: MAVEN_HOME

└── Goals: clean

├── **Add Build Step** -> "Invoke top-level Maven targets"

└── Maven version: MAVEN_HOME

└── Goals: install



Jenkins copy project issue
mavenweb_build Config - Jenkins
bhavagna06/maven-web-app
localhost:8081/job/mavenweb_build/configure
Jenkins / mavenweb_build / Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Advanced
Save
Apply

Invoke top-level Maven targets

Maven Version: MAVEN_HOME

Goals: clean

Invoke top-level Maven targets

Maven Version: MAVEN_HOME

Goals: install

Post-build Actions:

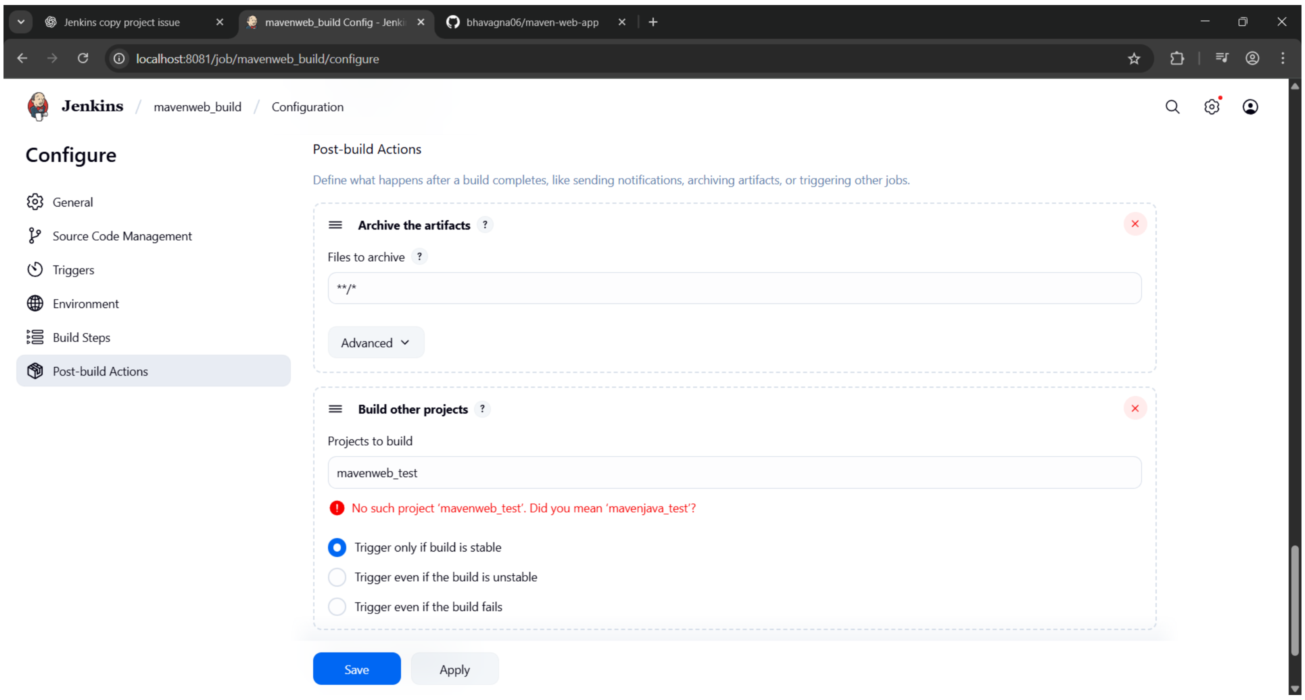
├── **Add Post Build Action** -> "Archive the artifacts"

└── Files to archive: **/*

├── **Add Post Build Action** -> "Build other projects"

└── Projects to build: MavenWeb_Test

└── Trigger: Only if build is stable


Jenkins copy project issue
mavenweb_build Config - Jenkins
bhavagna06/maven-web-app
localhost:8081/job/mavenweb_build/configure
Jenkins / mavenweb_build / Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Advanced
Save
Apply

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Archive the artifacts

Files to archive: **/*

Build other projects

Projects to build: mavenweb_test

No such project 'mavenweb_test'. Did you mean 'mavenjava_test'?

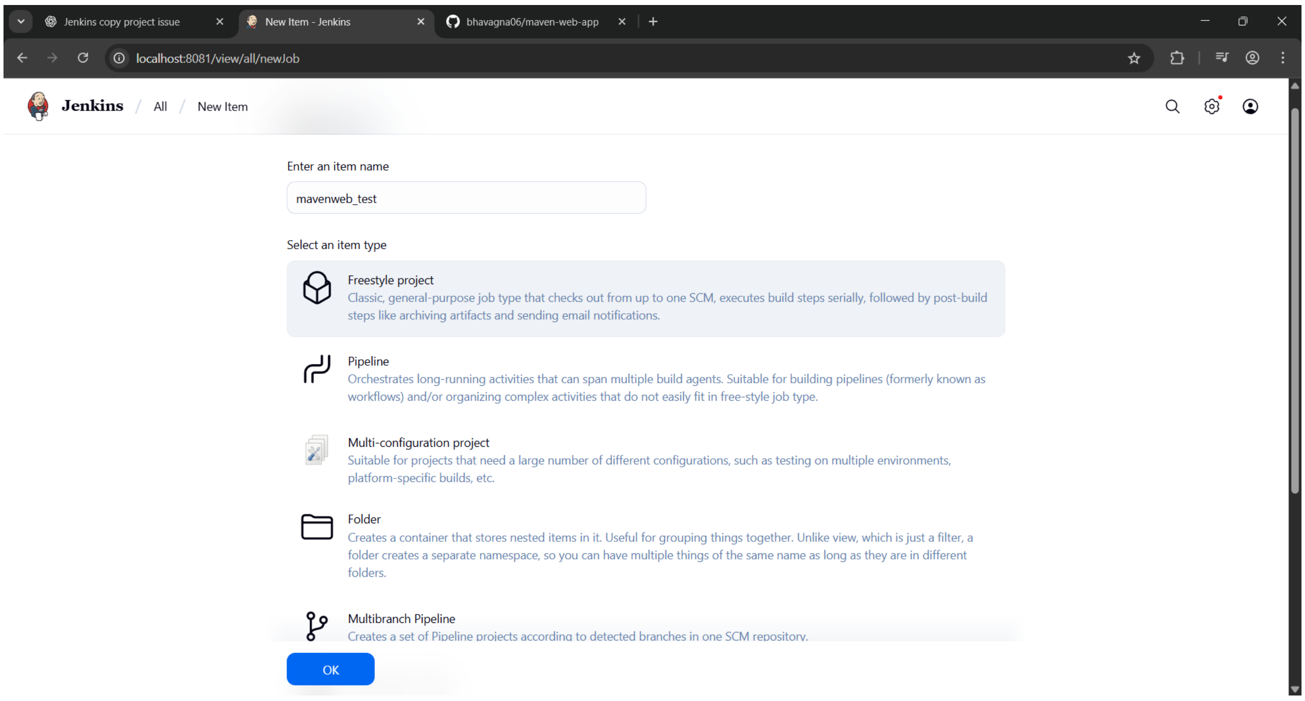
- [✓ Trigger only if build is stable]
- Trigger even if the build is unstable
- Trigger even if the build fails

└── Apply and Save

Step 3: Create Freestyle Project (e.g., MavenWeb_Test)

├── Enter project name (e.g., MavenWeb_Test)

├── Click "OK"


Jenkins / All / New Item

Enter an item name

mavenweb_test

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

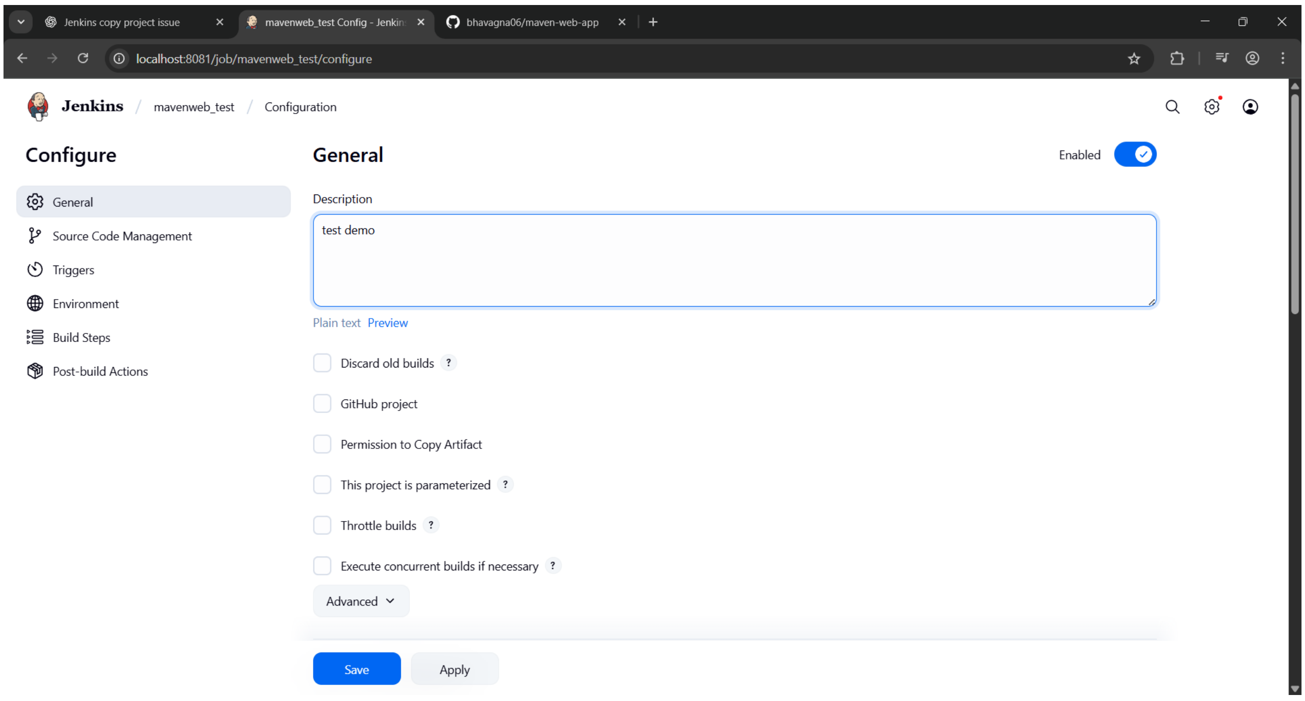
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK

Configure the project:

├── **Description:** "Test demo"


Jenkins / mavenweb_test / Configuration

Configure: General, Source Code Management, Triggers, Environment, Build Steps, Post-build Actions

General — Enabled

Description

test demo

Plain text Preview

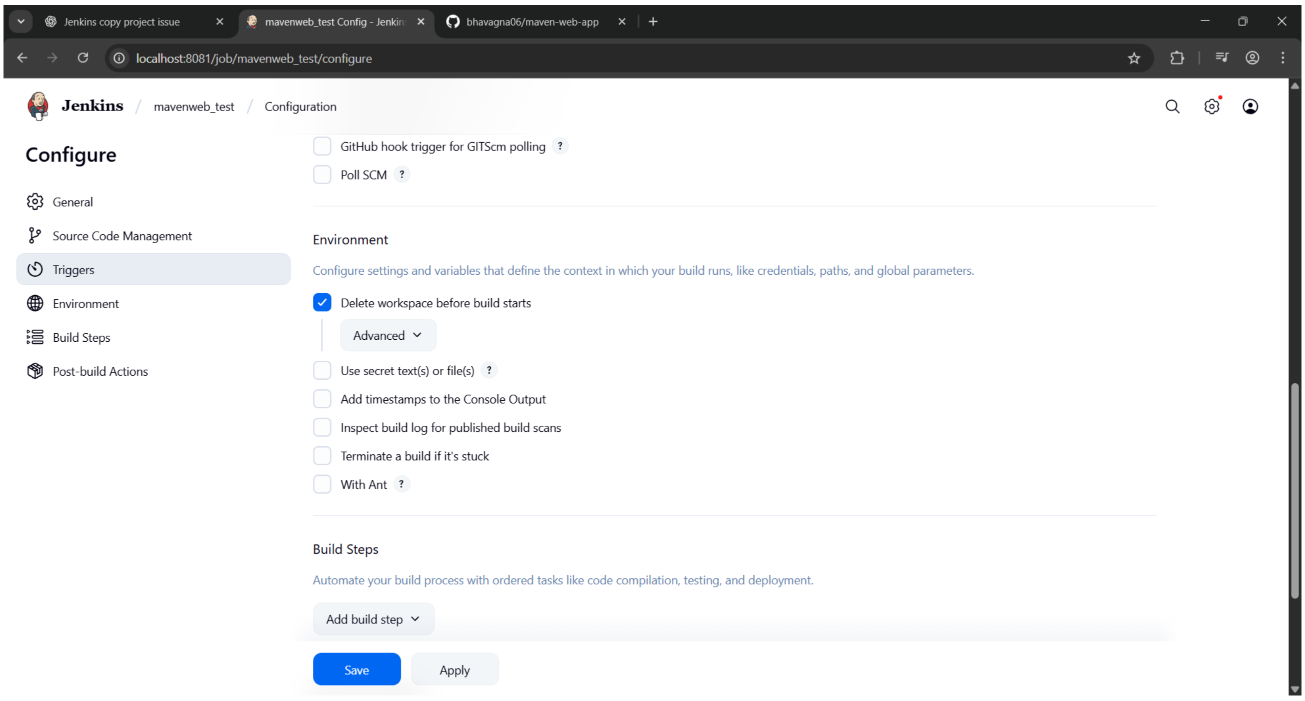
- [ ] Discard old builds
- [ ] GitHub project
- [ ] Permission to Copy Artifact
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary

Advanced

Save Apply

Build Environment:

└── Check: "Delete the workspace before build starts"



Jenkins / mavenweb_test / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

- [ ] GitHub hook trigger for GITScm polling ?
- [ ] Poll SCM ?

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [x] Delete workspace before build starts

Advanced

- [ ] 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.

Add build step

Save Apply

Add Build Step -> "Copy artifacts from another project"

└── Project name: MavenWeb_Build

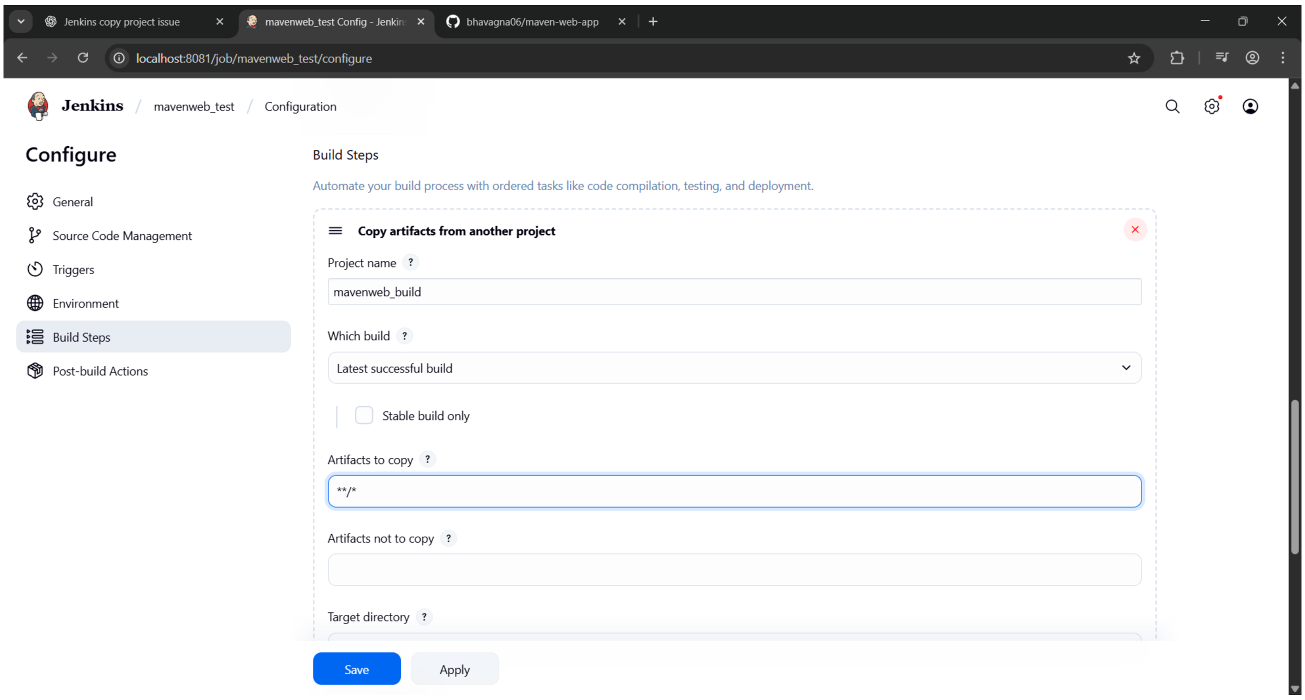
└── Build: Stable build only

└── Artifacts to copy: **/*

Add Build Step -> "Invoke top-level Maven targets"

└── Maven version: MAVEN_HOME

└── Goals: test



Jenkins / mavenweb_test / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Build Steps

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

Copy artifacts from another project

Project name ?
mavenweb_build

Which build ?
Latest successful build

- [ ] Stable build only

Artifacts to copy ?
**/*

Artifacts not to copy ?

Target directory ?

Save Apply

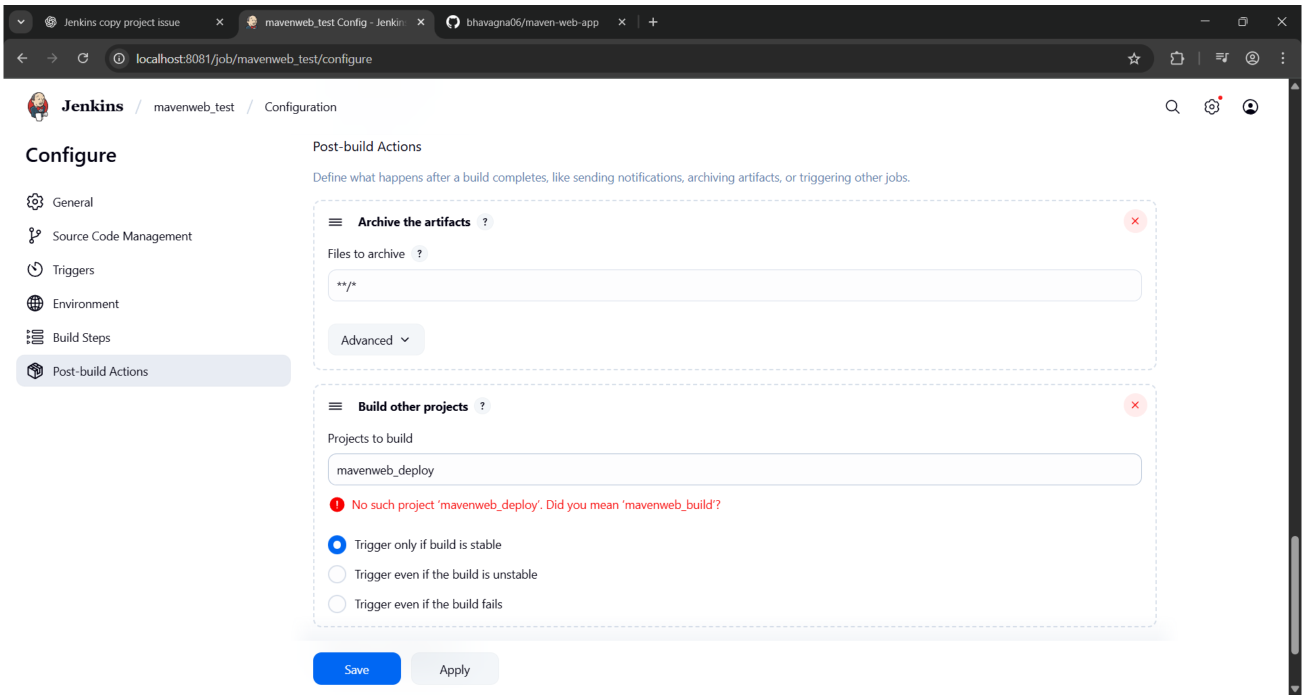
Post-build Actions:

├── **Add Post Build Action** -> "Archive the artifacts"

└── Files to archive: **/*

├── **Add Post Build Action** -> "Build other projects"

└── Projects to build: MavenWeb_Deploy



Jenkins copy project issue | mavenweb_test Config - Jenkins | bhavagna06/maven-web-app

localhost:8081/job/mavenweb_test/configure

Jenkins / mavenweb_test / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Archive the artifacts ?

Files to archive ?

**/*

Advanced

Build other projects ?

Projects to build

mavenweb_deploy

No such project 'mavenweb_deploy'. Did you mean 'mavenweb_build'?

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

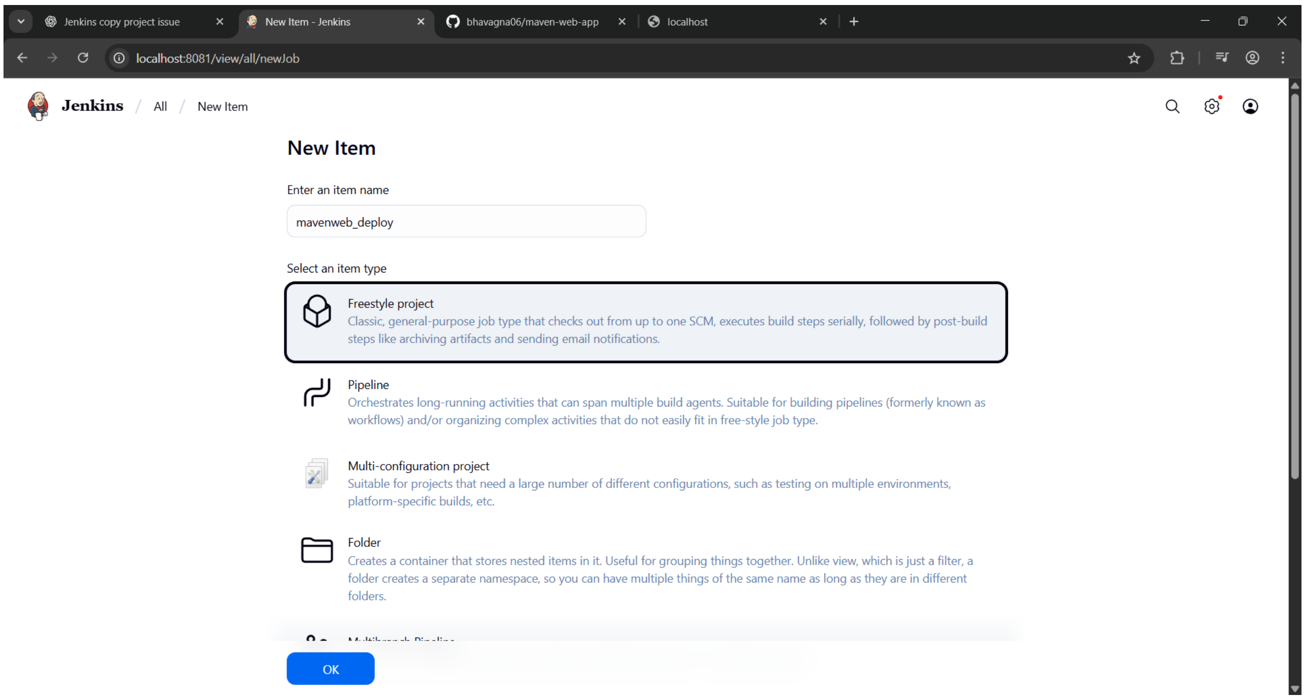
Save | Apply

Apply and Save

Step 4: Create Freestyle Project (e.g., MavenWeb_Deploy)

├── Enter project name (e.g., MavenWeb_Deploy)

├── Click "OK"



New Item

Enter an item name

mavenweb_deploy

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

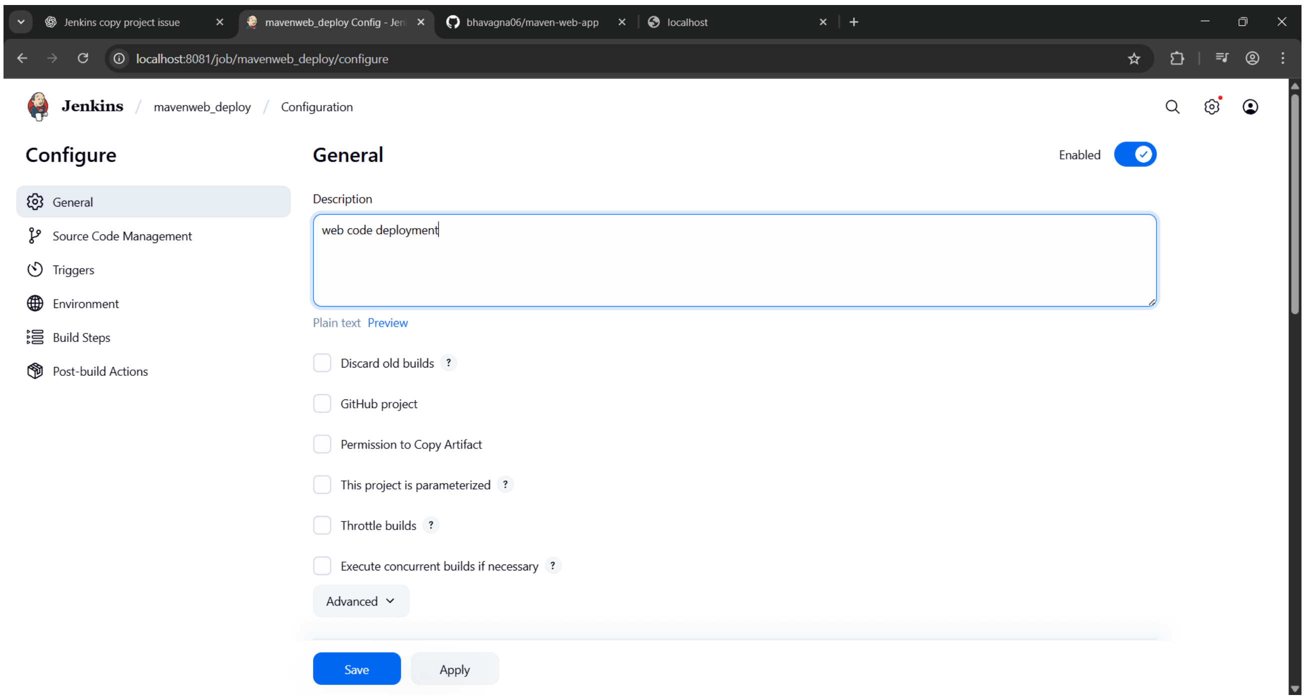
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Configure the project:

├── **Description**: "Web Code Deployment"


Jenkins copy project issue
mavenweb_deploy Config - Jen
bhavagna06/maven-web-app
localhost
localhost:8081/job/mavenweb_deploy/configure
Jenkins / mavenweb_deploy / Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions

General

Enabled [✓]

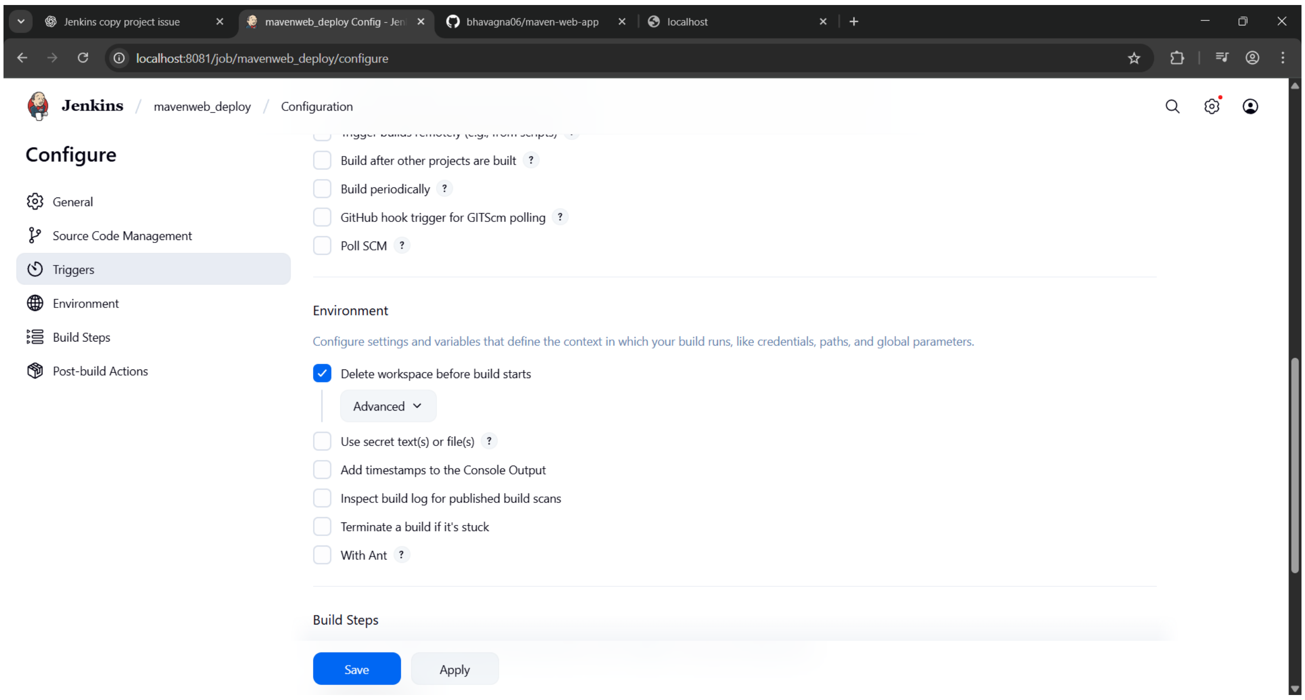
Description

web code deployment

- [ ] Discard old builds
- [ ] GitHub project
- [ ] Permission to Copy Artifact
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary

Build Environment:

└── Check: "Delete the workspace before build starts"


Jenkins / mavenweb_deploy / Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [x] Delete workspace before build starts
- [ ] 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

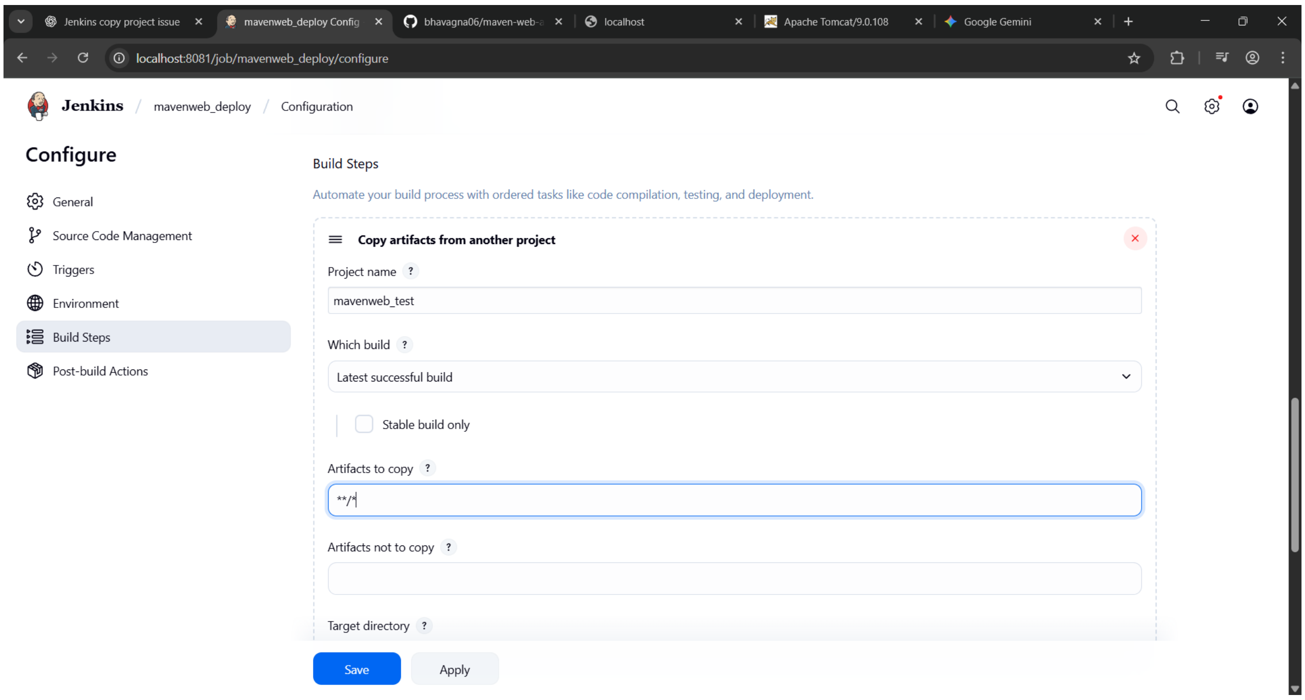
Build Steps

Add Build Step -> "Copy artifacts from another project"

└── Project name: MavenWeb_Test

└── Build: Stable build only

└── Artifacts to copy: **/*


Jenkins / mavenweb_deploy / Configuration

Configure

Build Steps

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

Copy artifacts from another project

Project name: mavenweb_test

Which build: Latest successful build

- [ ] Stable build only

Artifacts to copy: **/*

Artifacts not to copy:

Target directory

Post-build Actions:

├── **Add Post Build Action** -> "Deploy WAR/EAR to a container"

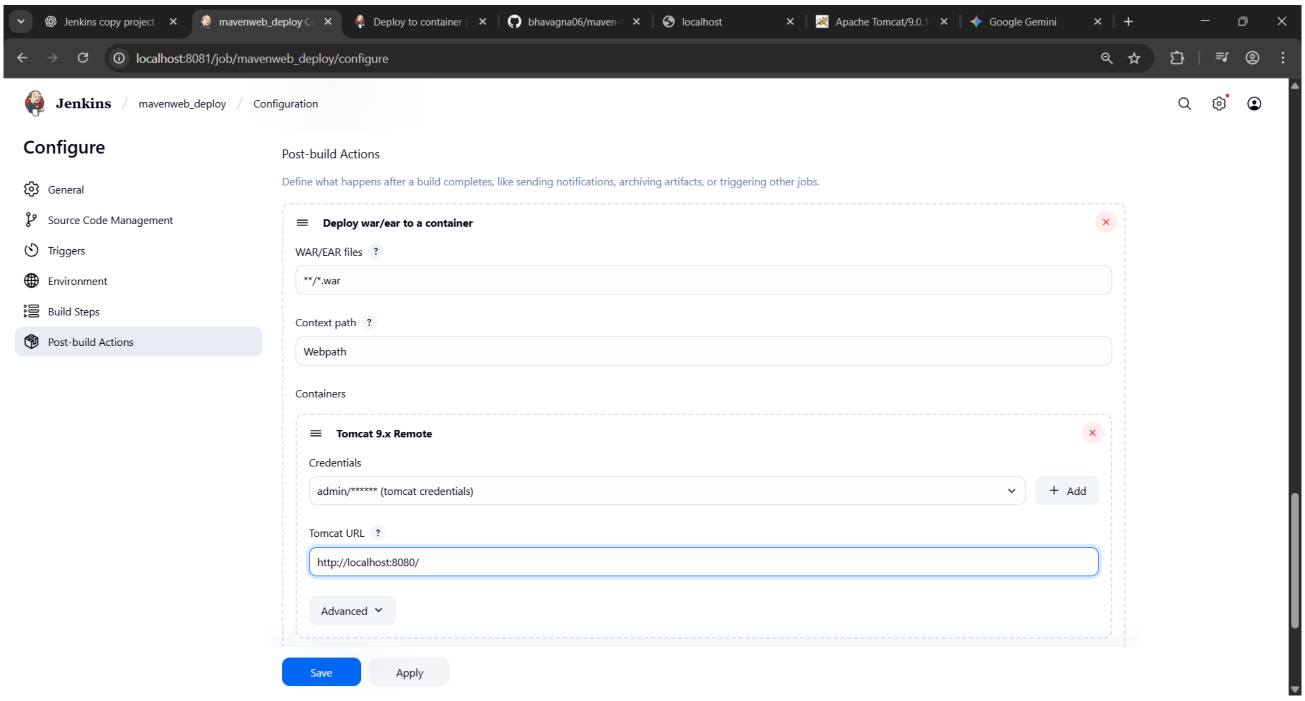
└── WAR/EAR File: **/*.war

└── Context path: Webpath

└── Add container -> Tomcat 9.x remote

└── Credentials: Username: admin, Password: 1234

— Tomcat URL: https://localhost:8080/



Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Deploy war/ear to a container

WAR/EAR files: **/*.war

Context path: Webpath

Containers

Tomcat 9.x Remote

Credentials: admin/****** (tomcat credentials)

Tomcat URL: http://localhost:8080/

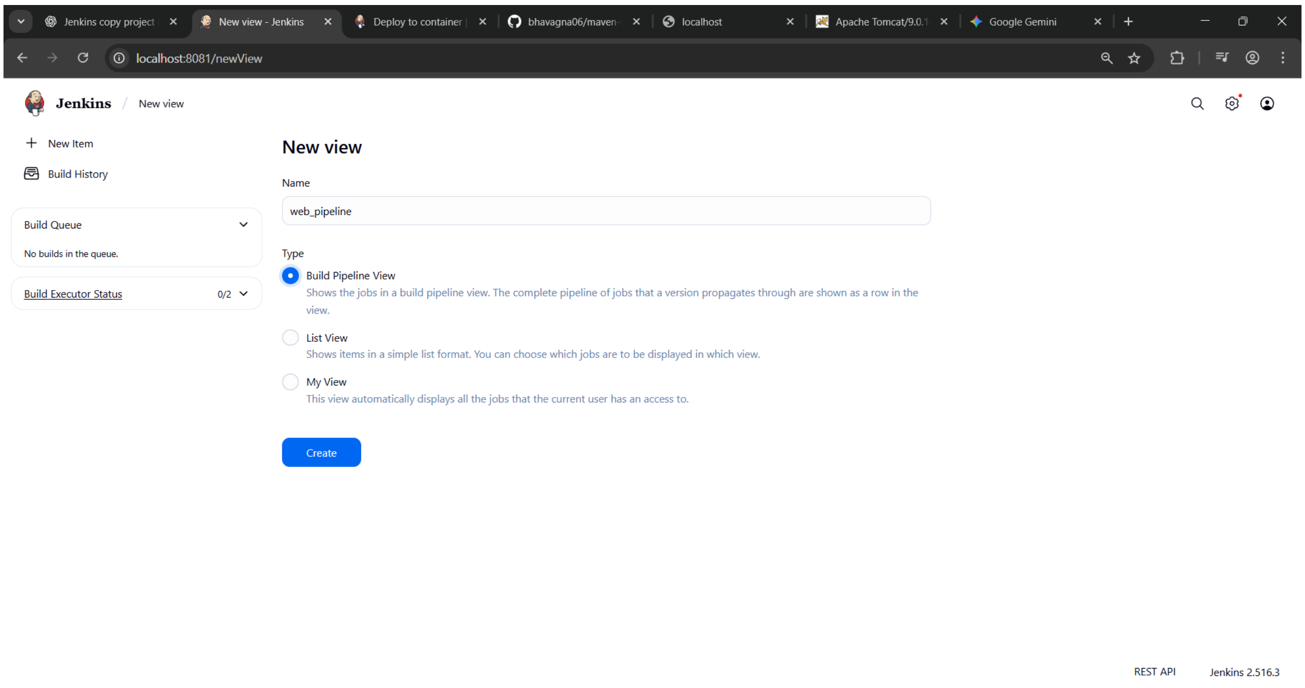
Advanced

└── Apply and Save

└── **Step 5**: Create Pipeline View for MavenWeb

├── Click "+" beside "All" on the dashboard

├── Enter name: MavenWeb_Pipeline


Jenkins / New view

New Item
Build History
Build Queue
No builds in the queue.
Build Executor Status 0/2

New view

Name
web_pipeline

Type
[✓ Build Pipeline View]
Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.
List View
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.
My View
This view automatically displays all the jobs that the current user has an access to.

Create

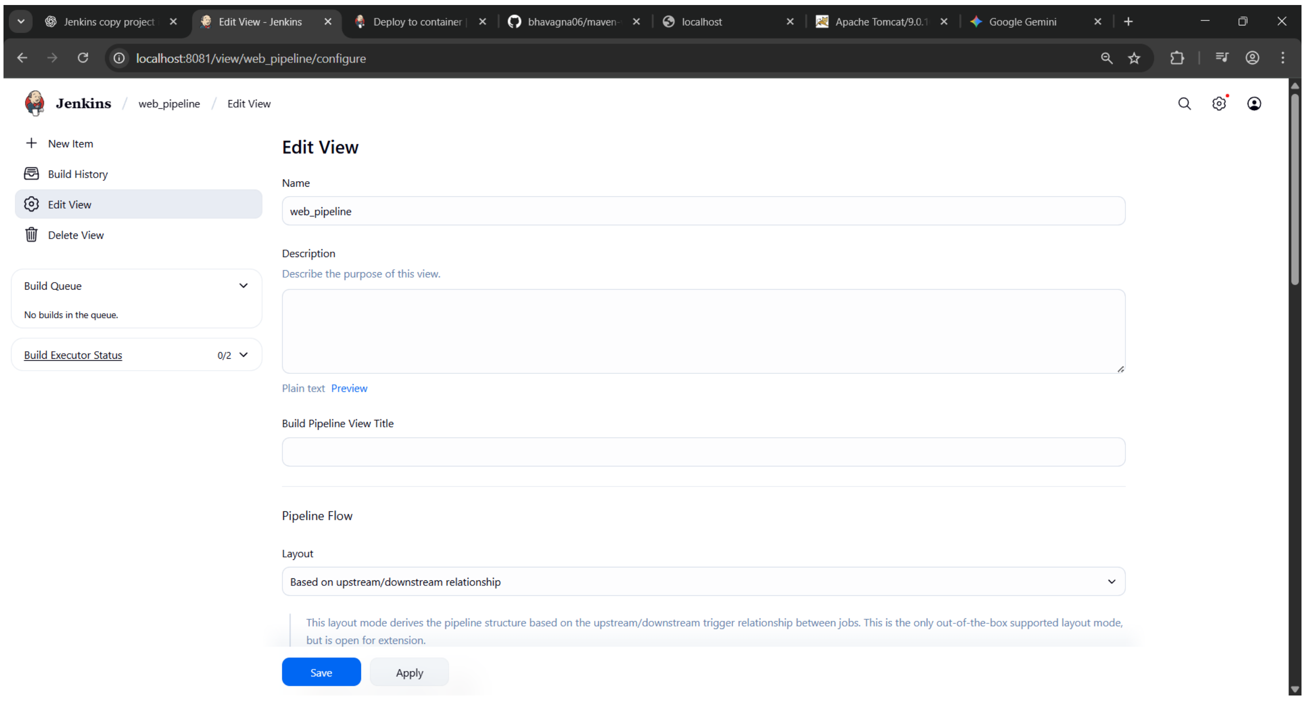
REST API Jenkins 2.516.3

Select "Build pipeline view"

└── **Pipeline Flow**:

├── **Layout**: Based on upstream/downstream relationship

├── Initial job: MavenWeb_Build


Jenkins / web_pipeline / Edit View

New Item
Build History
Edit View
Delete View
Build Queue
No builds in the queue.
Build Executor Status 0/2

Edit View

Name
web_pipeline

Description
Describe the purpose of this view.

Plain text Preview

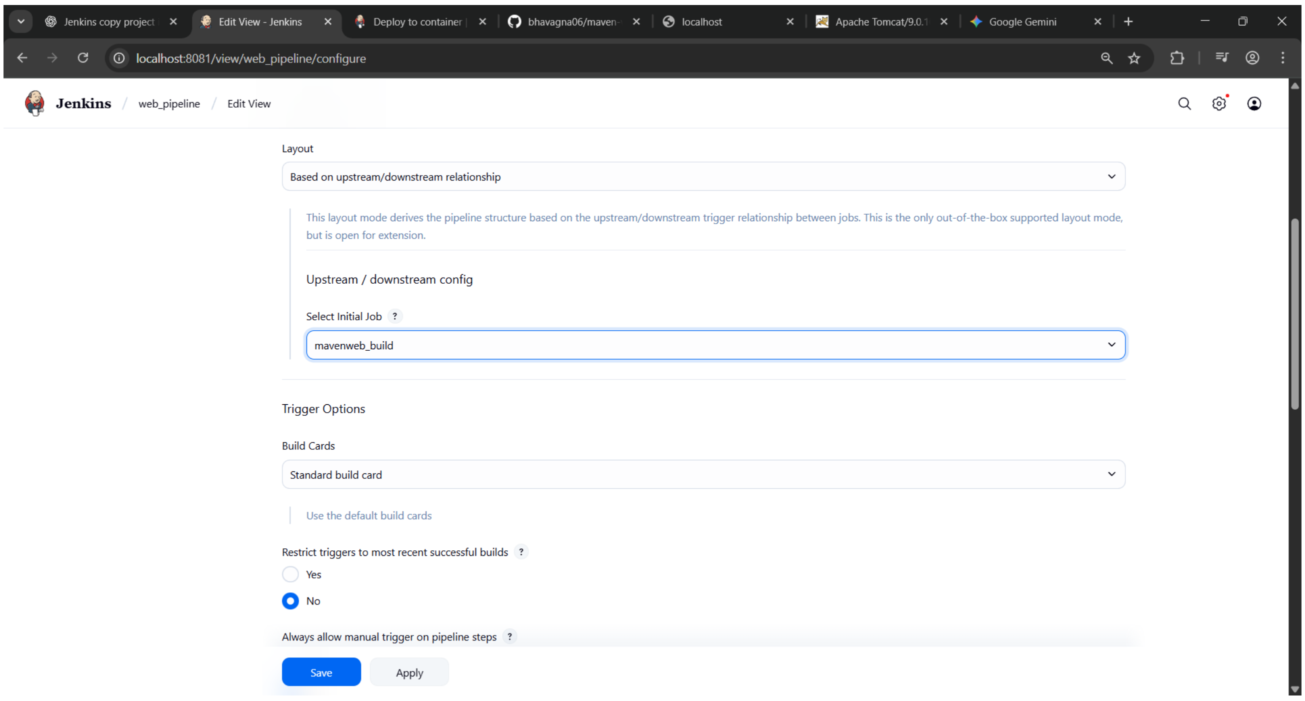
Build Pipeline View Title

Pipeline Flow

Layout
Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Save Apply


localhost:8081/view/web_pipeline/configure

Jenkins / web_pipeline / Edit View

Layout

Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job ?

mavenweb_build

Trigger Options

Build Cards

Standard build card

Use the default build cards

Restrict triggers to most recent successful builds ?

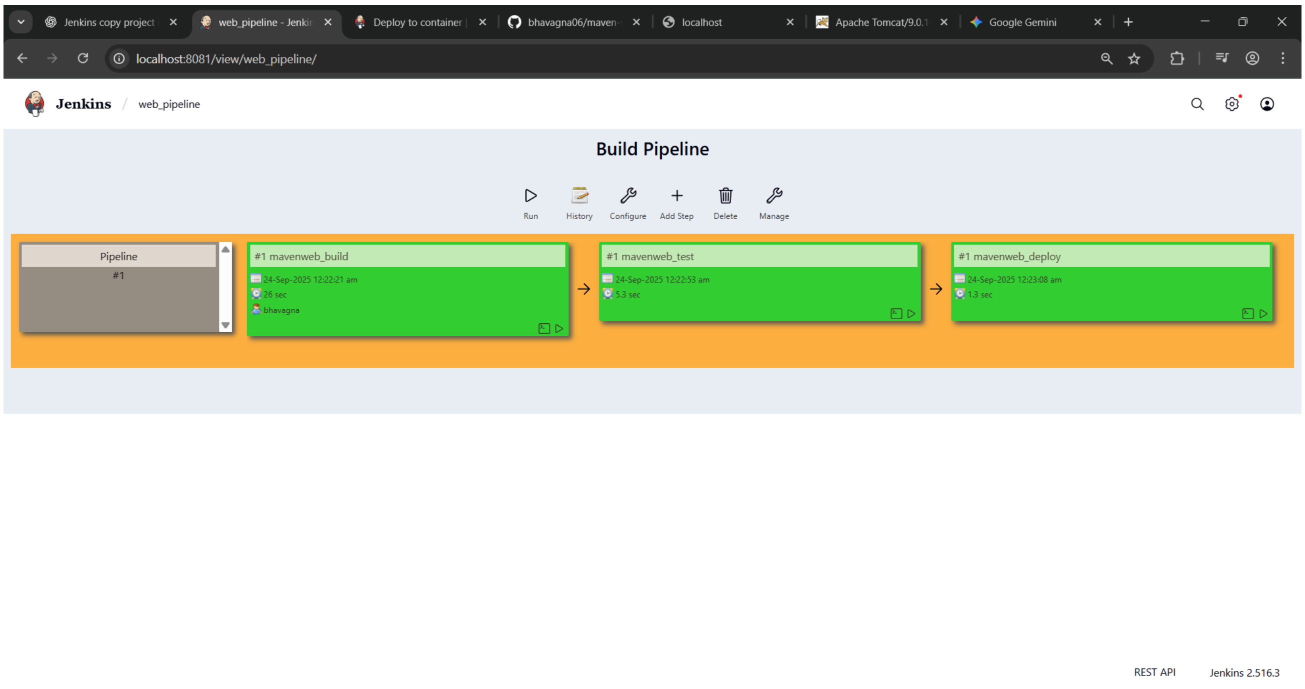
Yes [✓ No]

Always allow manual trigger on pipeline steps ?

└── Apply and Save

└── **Step 6**: Run the Pipeline and Check Output

├── Click on the trigger **"RUN"** to run the pipeline


localhost:8081/view/web_pipeline/

Jenkins / web_pipeline

Build Pipeline

Pipeline #1

#1 mavenweb_build — 24-Sep-2025 12:22:21 am — 26 sec — bhavagna

#1 mavenweb_test — 24-Sep-2025 12:22:53 am — 5.3 sec

#1 mavenweb_deploy — 24-Sep-2025 12:23:08 am — 1.3 sec

REST API Jenkins 2.516.3

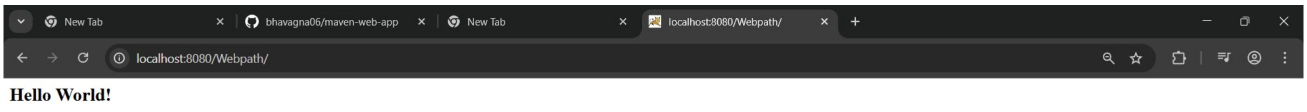
Note:

1. After Click on Run -> click on the small black box to open the console to check if the build is success

2. Now we see all the build has success if it appears in green color

├── Open Tomcat homepage in another tab

├── Click on the "/webpath" option under the manager app



Hello World!

III. Questions on Jenkins

1. What is Jenkins primarily used for?

Jenkins is used for continuous integration (CI) and continuous deployment/Delivery (CD).

2. What is feature of Jenkins?

pipeline support, extensibility

3. What is the default port on which Jenkins runs?

8080

4. What can be integrated with Jenkins for version control?

Git

5. What is the purpose of Jenkins plugins?

Plugins help in functionality like scm, email and tools

6. Which type of Jenkins job is best suited for running one-off tasks or small scripts?

Freestyle project

7. How can you manage sensitive information such as API keys in Jenkins?

It is stored in Jenkins credentials plugin

8. What does the "blue ocean" feature in Jenkins refer to?

Blue Ocean is a modern UI for Jenkins ,improves usability and user-friendly

9. What does the "blue ocean" feature in Jenkins refer to?

Blue Ocean is a modern UI for Jenkins ,improves usability and user-friendly

10. Which Jenkins component allows for distributed builds across multiple machines?

It uses Master-Agent Architecture

11. List at least five Jenkins plugins that you would consider important for a microservices-based application CI/CD pipeline. Briefly explain the purpose of each plugin.

Git Plugin → Integrates Git repositories for version control.Pipeline Plugin → Allows defining CI/CD pipelines using Groovy (Jenkinsfile).Docker Plugin → Builds, runs, and manages Docker containers for microservices

12. Explain the steps you would take to install a plugin in Jenkins through the Jenkins UI. What considerations would you keep in mind regarding plugin compatibility and updates?

In Manage Plugins we can search for plugin and install

13. Explain the steps you would take to install a plugin in Jenkins through the Jenkins UI. What considerations would you keep in mind regarding plugin compatibility and updates?

Dashboard ->Manage Jenkins->Manage plugin ->install

14. After installing a plugin, explain how you would configure it within Jenkins. For example, if you installed the Git Plugin, what steps would you take to set it up for your pipeline?

    After installation configure it with global tools and add credentials and it in pipeline

15. Discuss common issues that might arise when using Jenkins plugins, such as dependency conflicts or version compatibility problems. How would you troubleshoot these issues?

Version compatibility, Dependency conflict