

## 1. Create one Declarative pipeline job

**Source Code:** <https://github.com/betawins/spring3-mvc-maven-xml-hello-world-1.git>

Click on create new job name as declarative-pipeline and type as pipeline.

← → ↻ Not secure 54.198.54.149:8080/view/all/newJob

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / All ▾ / New Item

### New Item

Enter an item name

declarative-pipeline

Select an item type

- Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, follows steps like archiving artifacts and sending email notifications.
- Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (form 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 folder creates a separate namespace, so you can have multiple things of the same name as long as they are folders.

OK

In that job click on git SCM and provide repo url and save it.

← → ↻ ⚠ Not secure 54.198.54.149:8080/job/declarative-pipeline/configure

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / declarative-pipeline ▾ / Configuration

### Configure

- ⚙ General
- 🕒 Triggers
- 🔧 Pipeline**
- 🔗 Advanced

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git

Credentials ?

- none - ▾ +

Advanced ▾

+ Add Repository

Branches to build ?

**Save** Apply

Go to manage Jenkins scroll down click on maven

← → ↻ ⚠ Not secure 54.198.54.149:8080/manage/configureTools/

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / Manage Jenkins ▾ / Tools

+ Add Maven

≡ **Maven**

Name

Maven3

☒ Install automatically ?

≡ **Install from Apache**

Version

3.9.12

+ Add Installer

+ Add Maven

**Save** Apply

## Add jdk in manage Jenkins,tools

JDK installations

JDK installations ^ Edited

+ Add JDK

**JDK**

Name

JDK8

JAVA\_HOME

/usr/lib/jvm/java-1.8.0-openjdk

☐ Install automatically ?

## Click on build now

← → ↻ Not secure 54.198.54.149:8080/job/declarative-pipeline/ ☆ 📁

📧 Gmail 📺 YouTube 📍 Maps 📁 All

**Jenkins** / declarative-pipeline 🔍 ⚙️

Status declarative-pipeline Add description

📄 Changes 🏗️ Build Now ⚙️ Configure 🗑️ Delete Pipeline 🔍 Full Stage View 📅 Stages ✎ Rename ? Pipeline Syntax

Last Successful Artifacts  
ncodeit-hello-world-3.0.war 3.97 MiB view

Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	Checkout	Build	Test	Package	Archive Artifact	Declarativ Post Actio
Average stage times: (full run time: ~12s)	278ms	278ms	219ms	2s	1s	1s	97ms	87ms
#7 Jan 09 15:54 No Changes	259ms	84ms	211ms	2s	4s	4s	185ms	61ms
#6 Jan 09 15:50 1 commit	271ms	69ms	211ms	465ms failed	56ms failed	55ms failed	52ms failed	110ms
#5 Jan 09 15:39 No Changes	306ms	681ms	235ms	5s	85ms	57ms	55ms	91ms

Builds

Filter /

Today

- 🟢 #7 10:24 AM
- 🔴 #6 10:20 AM

## 3. Create one Scripted pipeline job

**Source Code:** <https://github.com/betawins/spring3-mvc-maven-xml-hello-world-1.git>

Click on build new job and name it as scripted-pipeline and select type as pipeline.

← → ↻ Not secure 54.198.54.149:8080/view/all/newJob

Gmail YouTube Maps

Jenkins / All ▾ / New Item

### New Item

Enter an item name

scripted-pipeline

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.

OK

Go to gitscm and give your repository and Jenkins file path.

← → ↻ Not secure 54.198.54.149:8080/job/scripted-pipeline/configure

📧 Gmail 📺 YouTube 📍 Maps

 **Jenkins** / scripted-pipeline ▾ / Configuration

### Configure

- ⚙️ General
- 🕒 Triggers
- 📄 Pipeline**
- 🔧 Advanced

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git


Credentials ?

- none - ▾ +

Advanced ▾

← → ↻ Not secure 54.198.54.149:8080/job/scripted-pipeline/configure

📧 Gmail 📺 YouTube 📍 Maps

 **Jenkins** / scripted-pipeline ▾ / Configuration

### Configure

- ⚙️ General
- 🕒 Triggers
- 📄 Pipeline**
- 🔧 Advanced

Branches to build ?

Branch Specifier (blank for 'any') ?

\*/master

+ Add Branch

Repository browser ?

(Auto)

Additional Behaviours

+ Add

Script Path ?

Jenkinsfile\_scriptbased

☒ Lightweight checkout ?

Save Apply

Click on build now.

The screenshot shows the Jenkins interface for a job named 'scripted-pipeline'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area displays the 'Stage View' with a table of stage times and a 'Permalinks' section below it.

	Setup ENV	Checkout	Build	Test	Package	Archive	Cleanup
Average stage times: (full run time: ~11s)	89ms	271ms	4s	3s	3s	63ms	53ms
#1 Jan 09 16:06 No Changes	89ms	271ms	4s	3s	3s	63ms	53ms

The 'Permalinks' section shows a list of builds, with the first build (#1) at 10:36 AM.

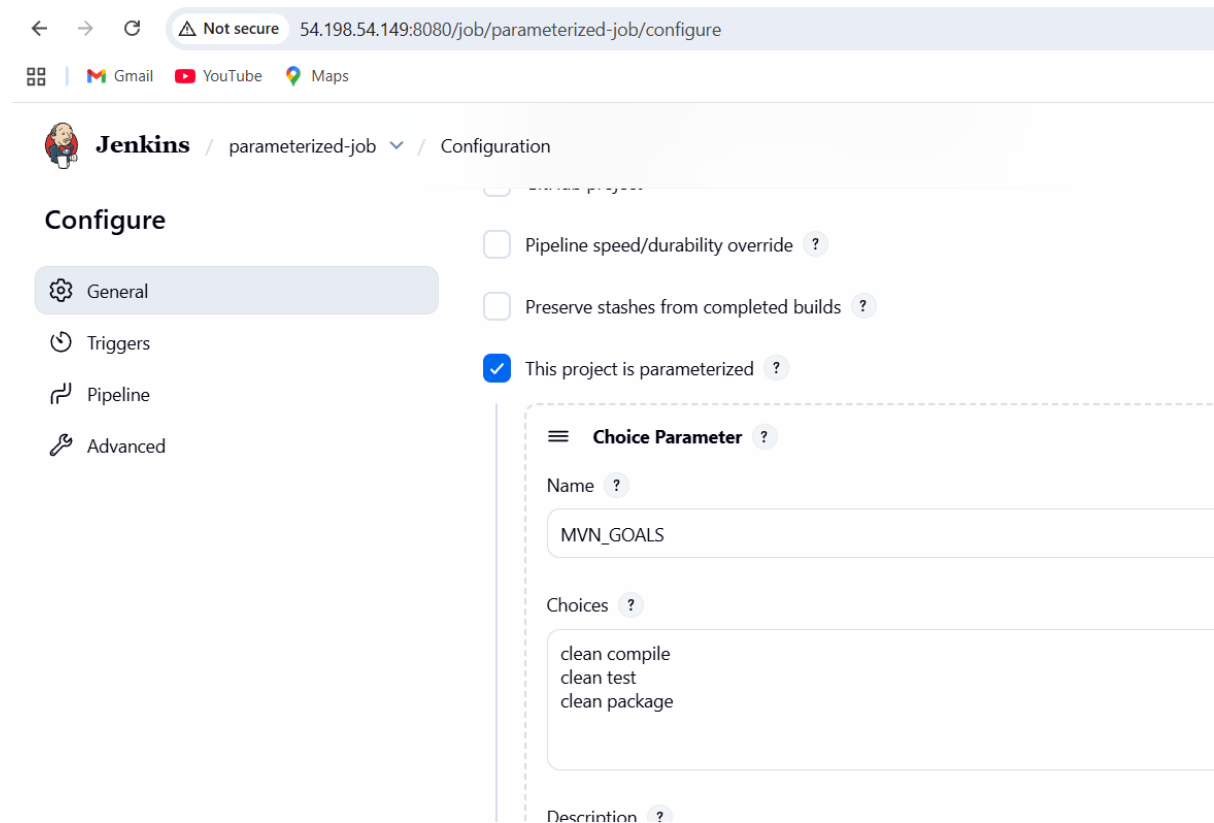
### 3. Create one Parameterized job

**Source Code:** <https://github.com/betawins/spring3-mvc-maven-xml-hello-world-1.git>

Create a new job for parameterized pipeline named as parameterized-job

The screenshot shows the Jenkins 'New Item' form. The 'Enter an item name' field contains 'parameterized-job'. The 'Select an item type' section has four options: Freestyle project, Pipeline, Multi-configuration project, and Folder. The 'Pipeline' option is selected, highlighted in blue. The 'OK' button is at the bottom.

Select this project is parameterized and select choices and give values.



The screenshot shows the Jenkins Configuration page for a job named 'parameterized-job'. The browser address bar indicates the URL is '54.198.54.149:8080/job/parameterized-job/configure'. The left sidebar shows the 'Configure' section with tabs for 'General', 'Triggers', 'Pipeline', and 'Advanced'. The 'General' tab is selected. In the main configuration area, the checkbox 'This project is parameterized' is checked. Below this, a 'Choice Parameter' is configured with the name 'MVN\_GOALS' and three choices: 'clean compile', 'clean test', and 'clean package'. There is also a 'Description' field.

← → ↻ Not secure 54.198.54.149:8080/job/parameterized-job/configure

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / parameterized-job / Configuration

**Configure**

- General
- Triggers
- Pipeline
- Advanced

☐ Pipeline speed/durability override ?

☐ Preserve stashes from completed builds ?

☒ This project is parameterized ?

**Choice Parameter** ?

Name ?

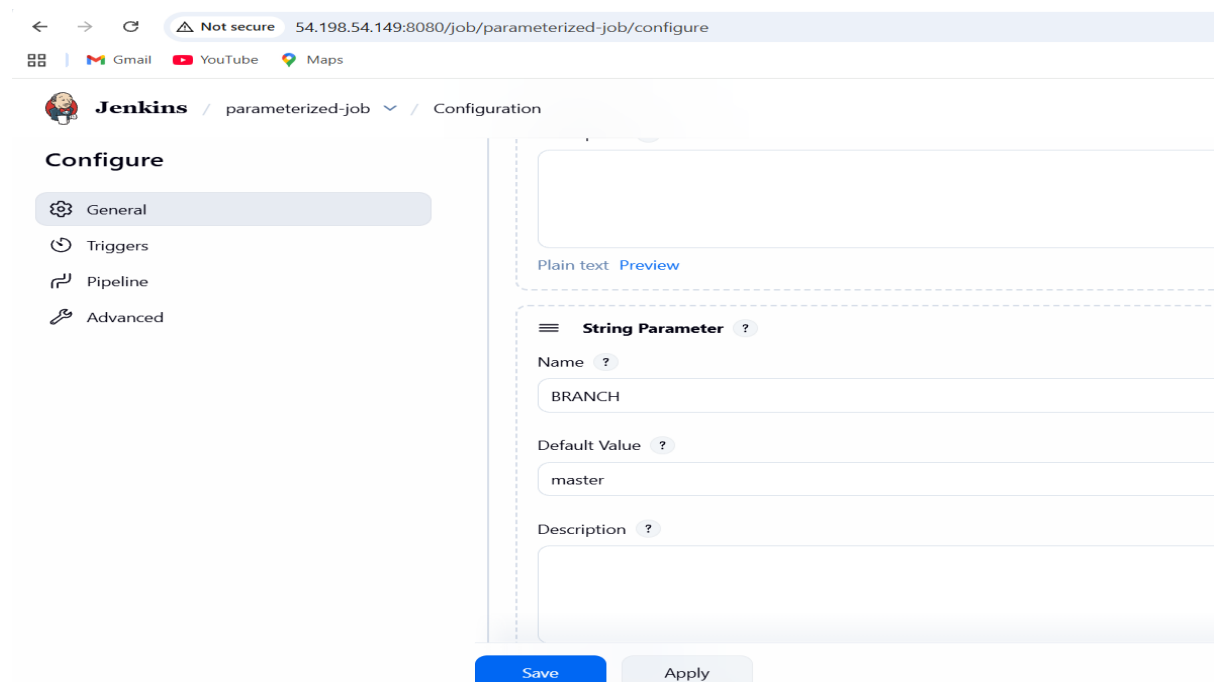
MVN\_GOALS

Choices ?

clean compile  
clean test  
clean package

Description ?

Add another parameter as string give name as branch , value as master.



The screenshot shows the same Jenkins Configuration page, but now a 'String Parameter' has been added. The 'Choice Parameter' is still visible. The 'String Parameter' has the name 'BRANCH' and a default value of 'master'. There is also a 'Description' field. At the bottom of the page, there are 'Save' and 'Apply' buttons.

← → ↻ Not secure 54.198.54.149:8080/job/parameterized-job/configure

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / parameterized-job / Configuration

**Configure**

- General
- Triggers
- Pipeline
- Advanced

☐ Pipeline speed/durability override ?

☐ Preserve stashes from completed builds ?

☒ This project is parameterized ?

**Choice Parameter** ?

Name ?

MVN\_GOALS

Choices ?

clean compile  
clean test  
clean package

Description ?

**String Parameter** ?

Name ?

BRANCH

Default Value ?

master

Description ?

Save Apply

Click on git SCM and give your repo name

The screenshot shows the Jenkins Configuration page for a job named 'parameterized-job'. The left sidebar has a 'Configure' section with tabs for 'General', 'Triggers', 'Pipeline', and 'Advanced'. The 'Pipeline' tab is selected. The main content area is titled 'Define your Pipeline using Groovy directly or pull it from source control.' and has a 'Definition' section with a text input field containing 'Pipeline script from SCM'. Below this is the 'SCM' section, which is set to 'Git'. Under 'Git', there is a 'Repositories' section with a 'Repository URL' field containing 'https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git'. There is also a 'Credentials' dropdown menu set to '- none -' and an 'Advanced' dropdown menu. At the bottom of the configuration area are 'Save' and 'Apply' buttons.

The screenshot shows the Jenkins Configuration page for the same job, but with the 'Branches to build' section expanded. The 'Branches to build' section has a 'Branch Specifier (blank for 'any')' field containing '\*/master' and an '+ Add Branch' button. Below this is the 'Repository browser' section, which is set to '(Auto)'. There is also an 'Additional Behaviours' section with an '+ Add' button. The 'Script Path' section is set to 'Jenkinsfile\_parameterized'. At the bottom, there is a checkbox for 'Lightweight checkout' which is checked. The 'Save' and 'Apply' buttons are at the bottom.

Click on build with parameters you can build which environment you need with the same script.



← → ↻ ⚠ Not secure 54.198.54.149:8080/job/parameterized-job/build?delay=0sec

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / parameterized-job ▾

- Status
- Changes
- Build with Parameters
- Configure
- Delete Pipeline
- Full Stage View
- Stages
- Rename
- Pipeline Syntax

## Pipeline parameterized-job

This build requires parameters:

MVN\_GOALS  
clean package

BRANCH  
master

**Build** Cancel

← → ↻ ⚠ Not secure 54.198.54.149:8080/job/parameterized-job/

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / parameterized-job

- Status
- Changes
- Build with Parameters
- Configure
- Delete Pipeline
- Full Stage View
- Stages
- Rename
- Pipeline Syntax

**parameterized-job**

Last Successful Artifacts  
ncodeit-hello-world-3.0.war 3.97 MiB view

## Stage View

	Init	Checkout	Build	Archive	Cleanup
Average stage times: (full run time: ~4s)	306ms	154ms	2s	51ms	51ms
#8 Jan 09 16:29 No Changes	666ms	152ms	2s	59ms	51ms
#7 Jan 09 16:28 No Changes	635ms	274ms	1s	44ms failed	
#6 Jan 09 16:27 No Changes	640ms	38ms failed			
#5 Jan 09 16:26 No Changes	103ms				

**Builds**

Filter /

Today

- #8 10:59 AM
- #7 10:58 AM
- #6 10:57 AM

## 4. Write sample skeleton for Declarative and Scripted pipelines

Declarative pipeline:

```
pipeline {
    agent any
```

```
tools {  
    jdk 'JDK8'      // Optional  
    maven 'MAVEN3'  // Optional  
}
```

```
environment {  
    APP_NAME = "sample-app"  
    BUILD_ENV = "dev"  
}
```

```
parameters {  
    string(name: 'BRANCH', defaultValue: 'main',  
description: 'Git branch')  
}
```

```
stages {  
  
    stage('Checkout') {  
        steps {  
            git branch: "${params.BRANCH}",
```

```
        url: 'https://github.com/org/repo.git'
    }
}
```

```
stage('Build') {
    steps {
        sh 'mvn clean compile'
    }
}
```

```
stage('Test') {
    steps {
        sh 'mvn test'
    }
}
```

```
stage('Package') {
    steps {
        sh 'mvn package'
    }
}
```

```
stage('Archive Artifacts') {  
    steps {  
        archiveArtifacts artifacts: 'target/*.jar', fingerprint:  
true  
    }  
}  
  
post {  
    success {  
        echo 'Pipeline executed successfully'  
    }  
    failure {  
        echo 'Pipeline failed'  
    }  
    always {  
        cleanWs()  
    }  
}  
}
```

Scripted pipeline:

```
node {
```

```
    def appName = "sample-app"
```

```
    def branchName = "main"
```

```
try {
```

```
    stage('Checkout') {
```

```
        git branch: branchName,
```

```
            url: 'https://github.com/org/repo.git'
```

```
    }
```

```
    stage('Build') {
```

```
        sh 'mvn clean compile'
```

```
    }
```

```
    stage('Test') {
```

```
        sh 'mvn test'
```

```
    }
```

```
stage('Package') {  
    sh 'mvn package'  
}  
  
stage('Archive Artifacts') {  
    archiveArtifacts artifacts: 'target/*.jar', fingerprint:  
true  
}  
  
echo "Pipeline completed successfully"  
  
} catch (err) {  
    echo "Pipeline failed"  
    throw err  
} finally {  
    cleanWs()  
}  
}
```

## 5 .Create one Declarative job

SourceCode: [https://github.com/betawins/sabear\\_simplecucumberapp/tree/feature-1.1](https://github.com/betawins/sabear_simplecucumberapp/tree/feature-1.1)

Build a new job.

← → ↺ ⚠ Not secure 54.198.54.149:8080/view/all/newJob

📧 Gmail 📺 YouTube 📍 Maps

**Jenkins** / All ▾ / New Item

### New Item

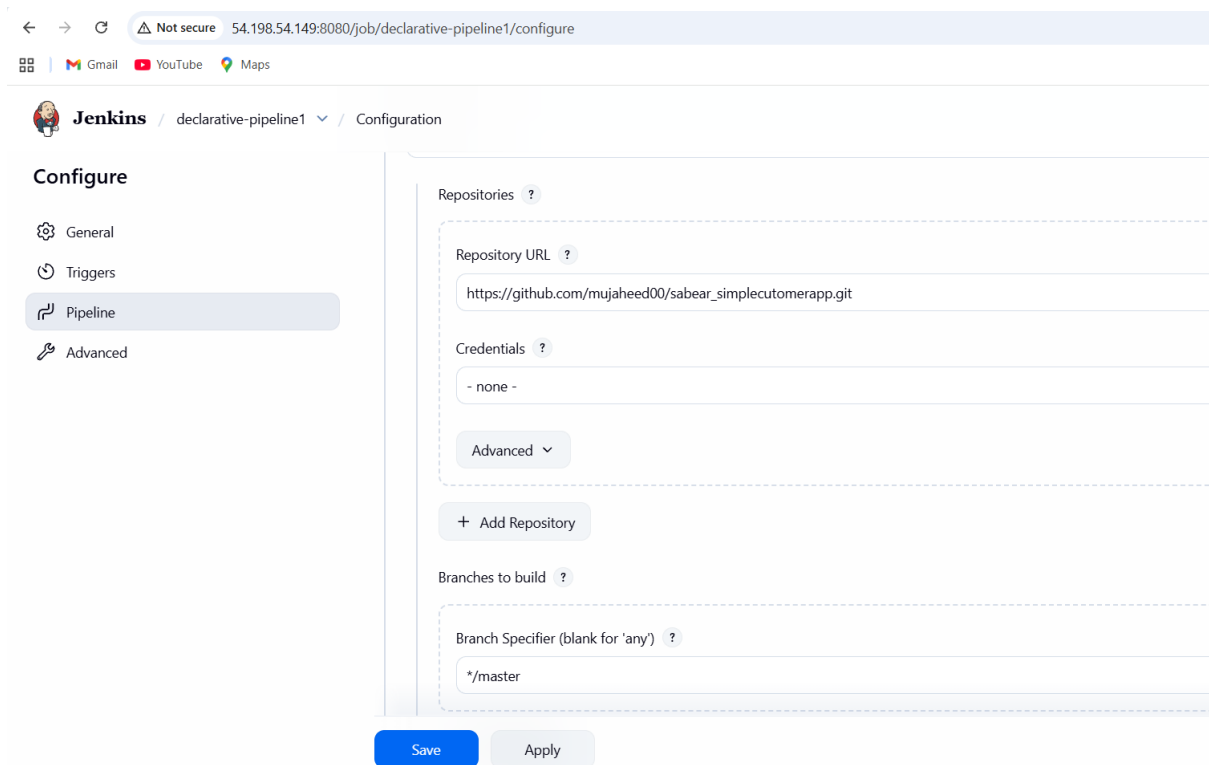
Enter an item name

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.

OK

Select git SCM and give your repo url

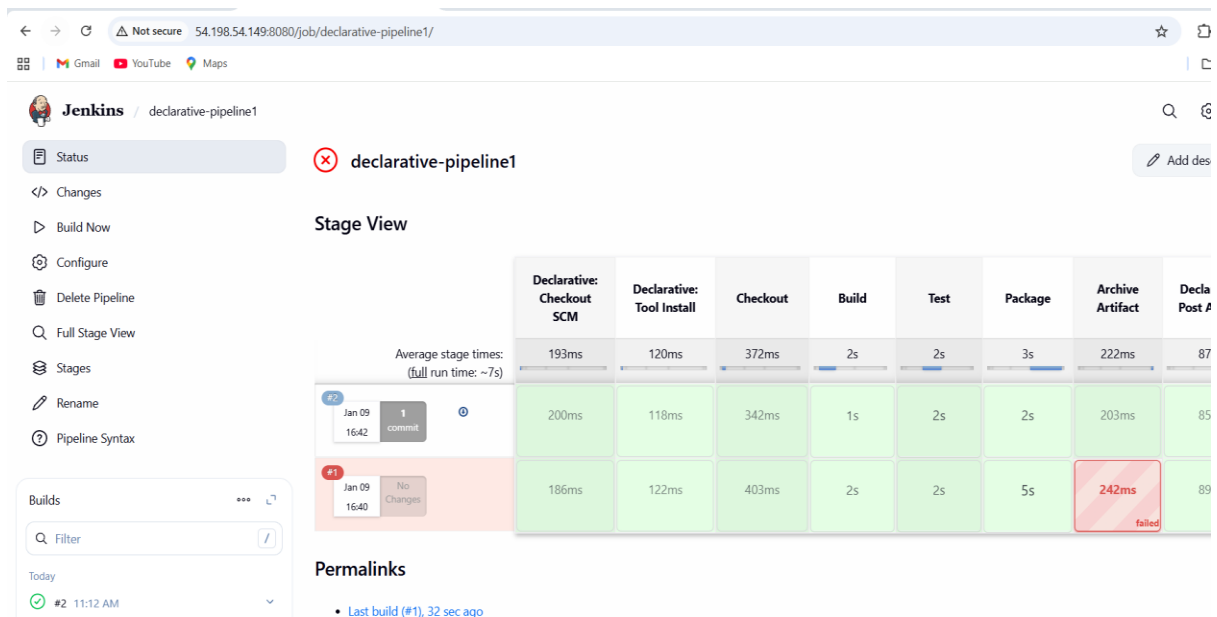


The screenshot shows the Jenkins 'Configure' page for a declarative pipeline. The left sidebar has options: General, Triggers, Pipeline (selected), and Advanced. The main area is titled 'Configuration' and contains the following fields:

- Repositories**: A section with a 'Repository URL' field containing 'https://github.com/mujaheed00/sabear\_simplecutomerapp.git', a 'Credentials' dropdown set to 'none', and an 'Advanced' dropdown.
- Branches to build**: A 'Branch Specifier (blank for \'any\')' field containing '\*/master'.

At the bottom are 'Save' and 'Apply' buttons.

Click on build now.



The screenshot shows the Jenkins 'declarative-pipeline1' run view. The left sidebar includes 'Status' (selected), 'Changes', 'Build Now', 'Configure', 'Delete Pipeline', 'Full Stage View', 'Stages', 'Rename', and 'Pipeline Syntax'. The main area shows a 'Stage View' table with stage times for two builds.

**Stage View Table:**

	Declarative: Checkout SCM	Declarative: Tool Install	Checkout	Build	Test	Package	Archive Artifact	Declarative Post Build
<b>Average stage times:</b> (full run time: ~7s)	193ms	120ms	372ms	2s	2s	3s	222ms	87
<b>#2</b> Jan 09 16:42 1 commit	200ms	118ms	342ms	1s	2s	2s	203ms	85
<b>#1</b> Jan 09 16:40 No Changes	186ms	122ms	403ms	2s	2s	5s	242ms failed	89

**Permalinks:**

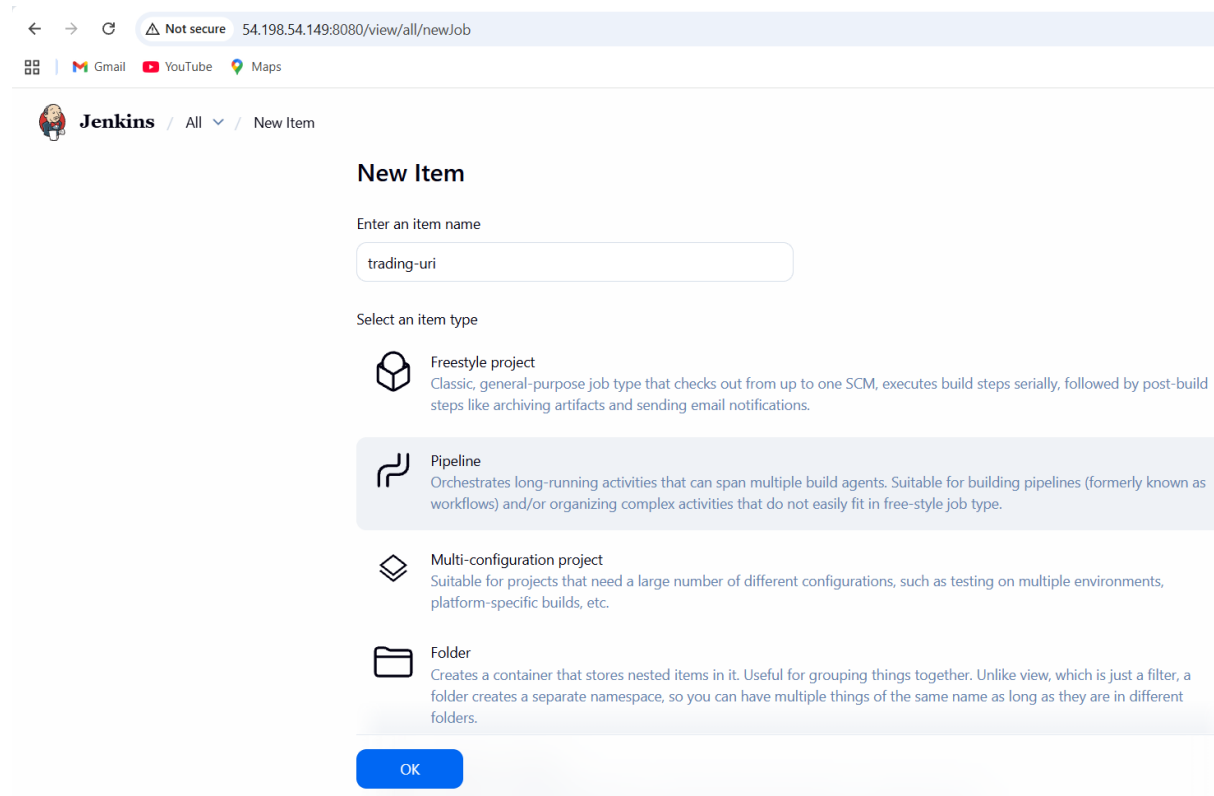
- Last build (#1), 32 sec ago

## 6. Create one Declarative job

**Source Code:** <https://github.com/betawins/Trading-UI.git>



# Create new job





The screenshot shows the Jenkins 'New Item' page in a web browser. The address bar indicates a 'Not secure' connection to 54.198.54.149:8080/view/all/newJob. The Jenkins logo and navigation links are visible. The main heading is 'New Item'. Below it, there is a text input field for 'Enter an item name' with the value 'trading-uri'. Underneath, a section titled 'Select an item type' lists four options: 'Freestyle project' (described as a classic, general-purpose job type), 'Pipeline' (described as orchestrating long-running activities), 'Multi-configuration project' (suitable for projects with many configurations), and 'Folder' (creates a container for nested items). The 'Pipeline' option is highlighted with a light blue background. At the bottom, there is a blue 'OK' button.


Enter an item name


trading-uri

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.

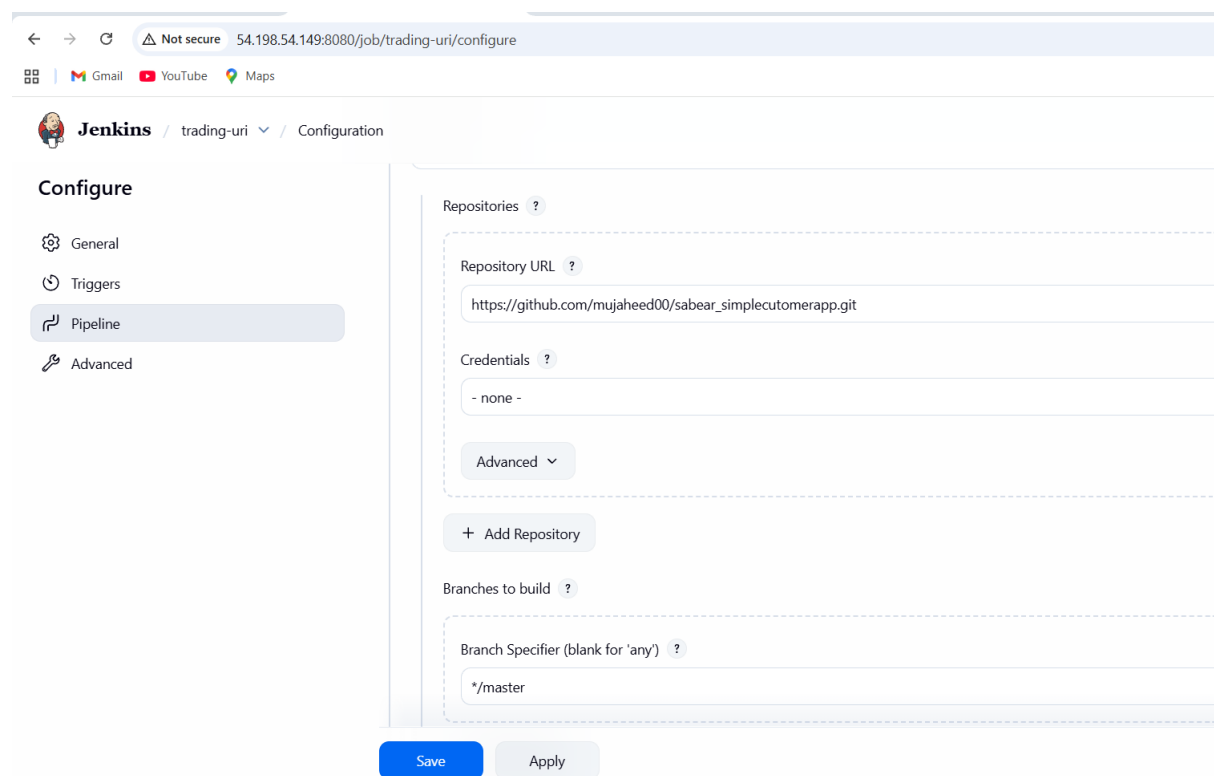
 **Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **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.

OK

# Click on git SCM and give your repository url



The screenshot shows the Jenkins 'Configure' page for a job named 'trading-uri'. The address bar shows the URL 54.198.54.149:8080/job/trading-uri/configure. On the left, a sidebar titled 'Configure' has four tabs: 'General', 'Triggers', 'Pipeline' (which is selected and highlighted), and 'Advanced'. The main content area is for configuring the 'Pipeline' job. It has a section for 'Repositories' with a 'Repository URL' field containing 'https://github.com/mujaheed00/sabear\_simplecutomerapp.git'. Below this is a 'Credentials' dropdown menu set to '- none -' and an 'Advanced' dropdown. There is a '+ Add Repository' button. Below that is a 'Branches to build' section with a 'Branch Specifier (blank for 'any')' field containing '\*/master'. At the bottom, there are 'Save' and 'Apply' buttons.

Configure

General

Triggers

**Pipeline**

Advanced

Repositories ?

Repository URL ?

https://github.com/mujaheed00/sabear\_simplecutomerapp.git

Credentials ?

- none -

Advanced ▾

+ Add Repository

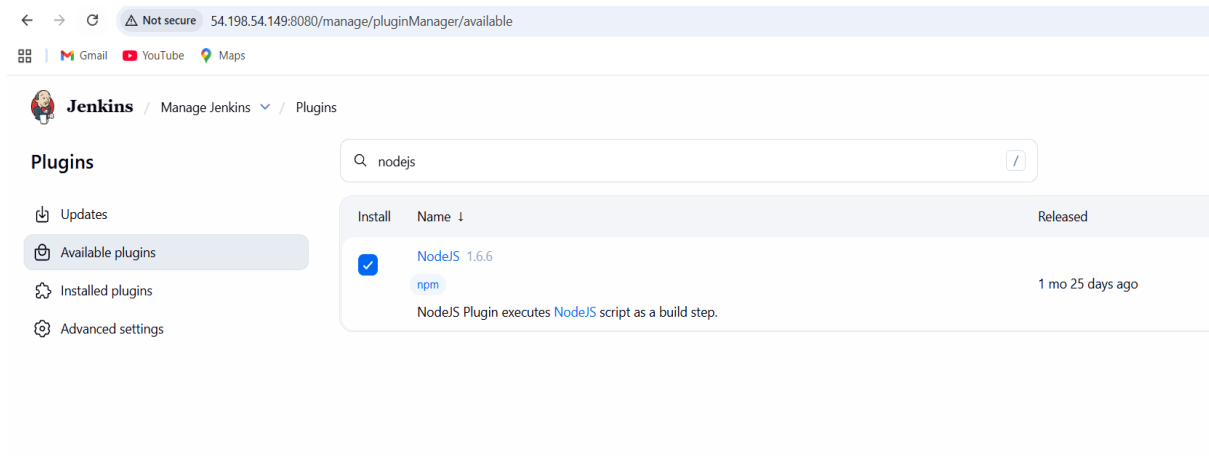
Branches to build ?

Branch Specifier (blank for 'any') ?

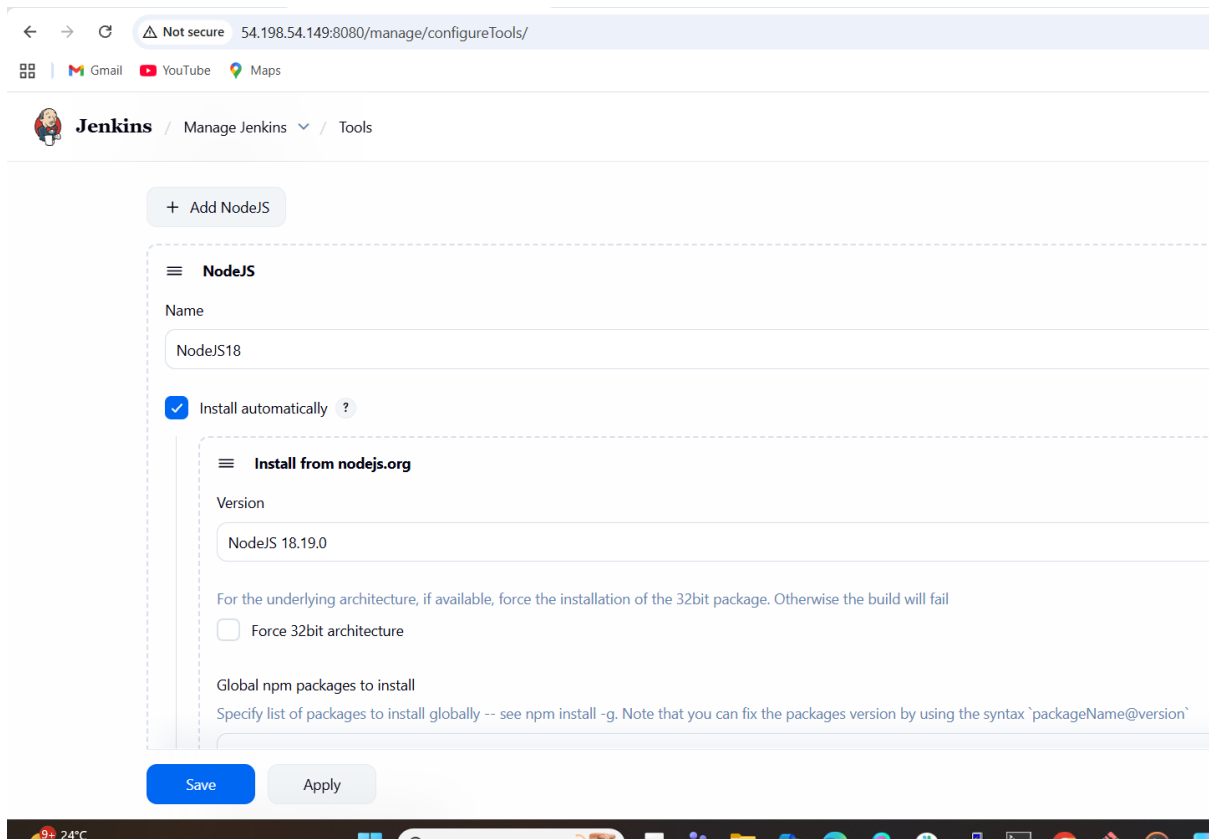
\*/master

Save Apply

# Install plugin nodejs



## Go to manage Jenkins, tools and give nodejs version



Click on build now.

The screenshot shows the Jenkins interface for a job named 'trading-uri'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area displays the 'Stage View' for the 'trading-uri' job. It includes a table of stage times and a list of builds.

	Declarative: Checkout SCM	Checkout	Install Dependencies	Build	Archive Artifacts	Declarative: Post Actions
Average stage times: (full run time: ~1s)	205ms	181ms	53ms	49ms	63ms	76ms
#7 Jan 09 17:40 1 commit	213ms	135ms	50ms	46ms	76ms	78ms
#6 Jan 09 17:38 3 commits	198ms	228ms	56ms	53ms	50ms	74ms

The table shows stage times for two builds. Build #6 is highlighted in red, indicating it failed. The 'Checkout' stage for build #6 failed with a time of 228ms. The 'Install Dependencies', 'Build', and 'Archive Artifacts' stages for build #6 also failed with times of 56ms, 53ms, and 50ms respectively. The 'Declarative: Post Actions' stage for build #6 completed with a time of 74ms.

Below the table, there is a section for 'Permalinks' with the following links:

- Last build (#7), 7.9 sec ago
- Last stable build (#7), 7.9 sec ago
- Last successful build (#7), 7.9 sec ago

## 7. Execute parallel stages using Jenkins pipeline for any sample job

Click on create new job with the name parallel-pipeline-demo and select pipeline as type.

The screenshot shows the Jenkins 'New Item' form. The 'Enter an item name' field contains 'parallel-pipeline-demo'. The 'Select an item type' section shows four options: Freestyle project, Pipeline, Multi-configuration project, and Folder. The 'Pipeline' option is selected and highlighted in blue.

**New Item**

Enter an item name

parallel-pipeline-demo

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 steps like archiving artifacts and sending email notifications.
- Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly 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 folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

In that job go to pipeline and select pipeline script and give this script.

```
pipeline {
```

```
    agent any
```

```
    stages {
```

```
        stage('Checkout') {
```

```
            steps {
```

```
                echo "Checking out source code..."
```

```
            }
```

```
        }
```

```
        stage('Parallel Stage') {
```

```
            parallel {
```

```
                stage('Build') {
```

```
                    steps {
```

```
                        echo "Building application..."
```

```
                        sh 'sleep 5'
```

```
                    }
```

```
}
```

```
stage('Test') {  
  steps {  
    echo "Running tests..."  
    sh 'sleep 5'  
  }  
}
```

```
stage('Scan') {  
  steps {  
    echo "Running code scan..."  
    sh 'sleep 5'  
  }  
}  
  
}  
  
}
```

```
stage('Deploy') {  
  steps {  
    echo "Deploying application..."
```

```
}  
  
}  
  
}  
  
}
```

Jenkins / parallel-pipeline-demo / Configuration

## Configure

- General
- Triggers
- Pipeline
- Advanced

Define your Pipeline using Groovy directly or pull it from source control.

### Definition

Pipeline script

#### Script ?

```
31             echo "Running code scan..."  
32             sh 'sleep 5'  
33         }  
34     }  
35 }  
36 }  
37  
38 stage('Deploy') {  
39     steps {  
40         echo "Deploying application..."  
41     }  
42 }  
43 }  
44 }  
45 }
```

Save

Apply

Save and click on build now.

← → ↻ Not secure 107.22.119.174:8080/job/parallel-pipeline-demo/ ☆

📱 Gmail YouTube Maps

Jenkins / parallel-pipeline-demo 🔍

Status

</> Changes

▶ Build Now

⚙️ Configure

🗑️ Delete Pipeline

🔍 Full Stage View

📋 Stages

✎️ Rename

❓ Pipeline Syntax

parallel-pipeline-demo

📝 Ad

### Stage View

		Checkout	Parallel Stage	Build	Test	Scan	Deploy
Average stage times: (full run time: ~7s)		85ms	104ms	5s	5s	5s	50ms
#1 Jan 10 09:40	No Changes	85ms	104ms	5s	5s	5s	50ms

### Permalinks

Builds

Today

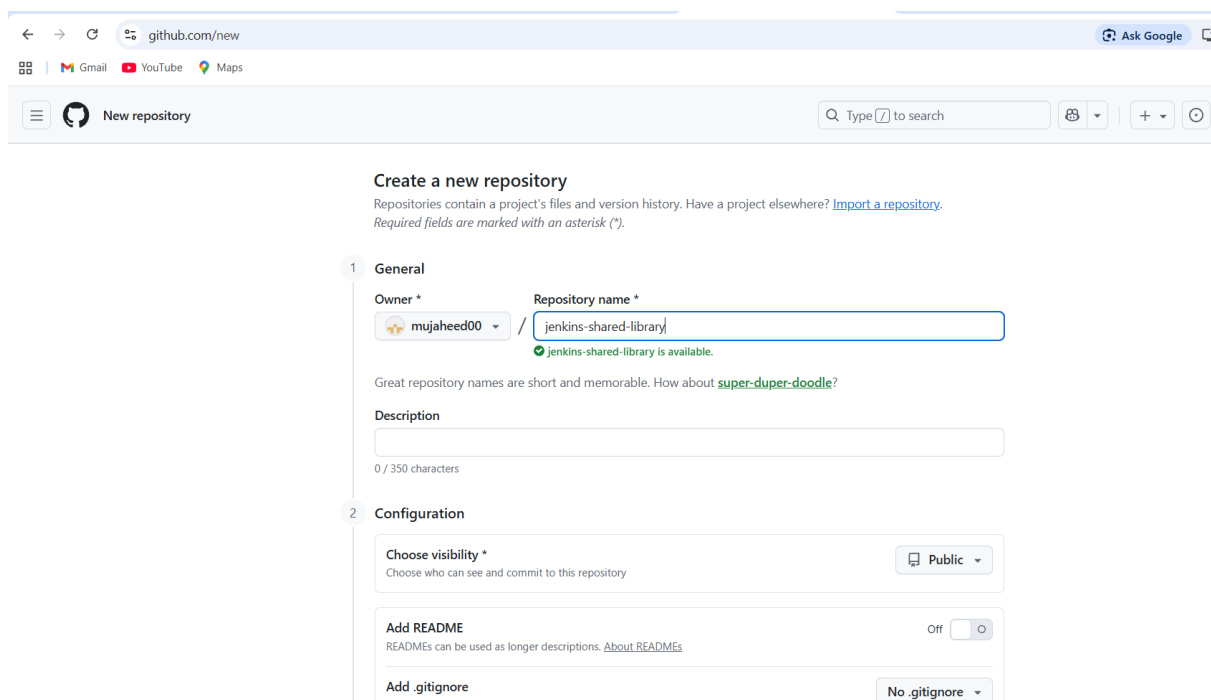
🟢 #1 4:10 AM

Build, test, scan these 3 steps will be executed parallelly with this we can optimize time.

## 8. Execute Jenkins pipeline stages using shared libraries

**Reference:** <https://phoenixnap.com/kb/jenkins-shared-library>

Create a new repository in github with the name Jenkins-shared-library



The screenshot shows the GitHub 'Create a new repository' page. The browser address bar shows 'github.com/new'. The page has a search bar and navigation links. The main content area is titled 'Create a new repository' and includes a link to 'Import a repository'. Below this, there are two tabs: 'General' and 'Configuration'. The 'General' tab is active and shows the 'Repository name' field with the value 'jenkins-shared-library'. A message below the field states 'jenkins-shared-library is available.' The 'Description' field is empty. The 'Configuration' tab is also visible and shows the 'Choose visibility' dropdown set to 'Public', the 'Add README' toggle set to 'Off', and the 'Add .gitignore' dropdown set to 'No .gitignore'.

Create a directory in that git hub named as vars so there is now specific type to create directory so, type vars/filename it will create a file inside that directory.

The screenshot shows a web browser at the URL `github.com/mujaheed00/jenkins-shared-library/new/main`. The repository name is `jenkins-shared-library`. The breadcrumb navigation shows `jenkins-shared-library / vars / buildStage.groovy` in the `main` branch. The file editor shows the following Groovy code:

```
1  def call() {
2      stage('Build') {
3          echo "Build stage from Shared Library"
4          sh 'sleep 3'
5      }
6  }
7  |
```

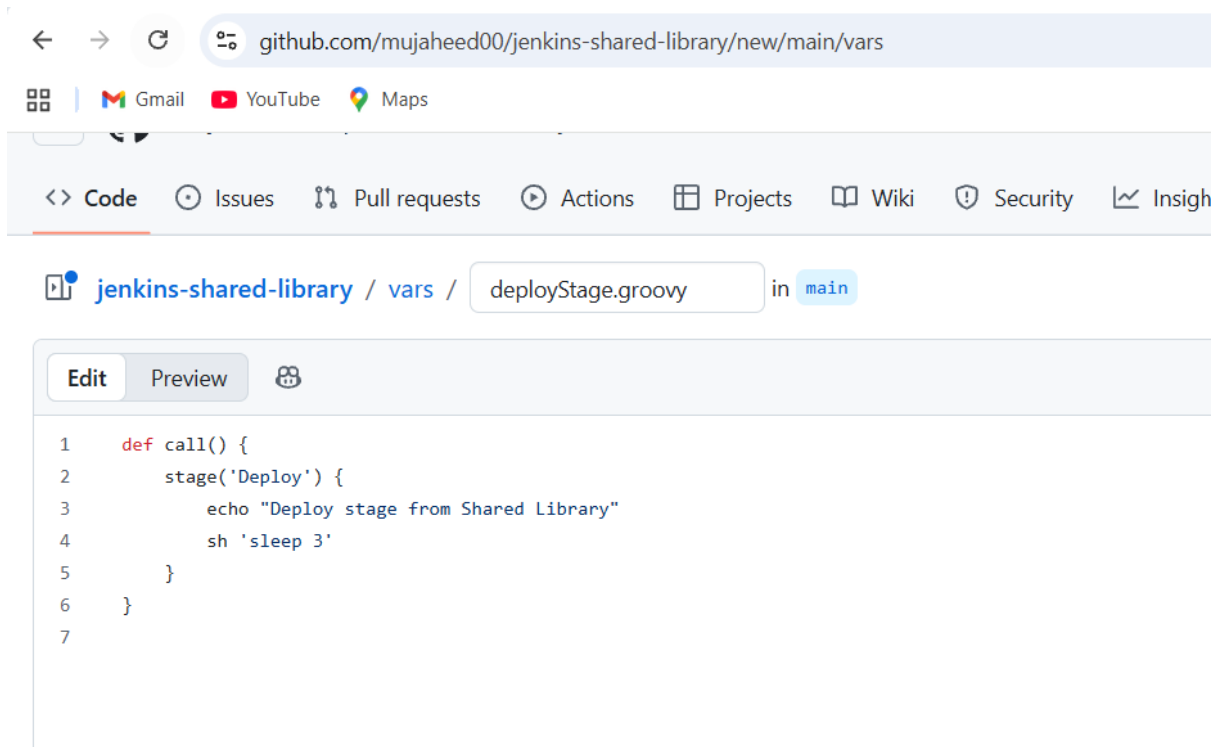
Create an another file with name var/testStage.groovy

The screenshot shows the same GitHub repository, but the breadcrumb navigation now shows `jenkins-shared-library / vars / testStage.groovy`. The file editor shows the following Groovy code:

```
1  def call() {
2      stage('Test') {
3          echo "Test stage from Shared Library"
4          sh 'sleep 3'
5      }
6  }
7
```

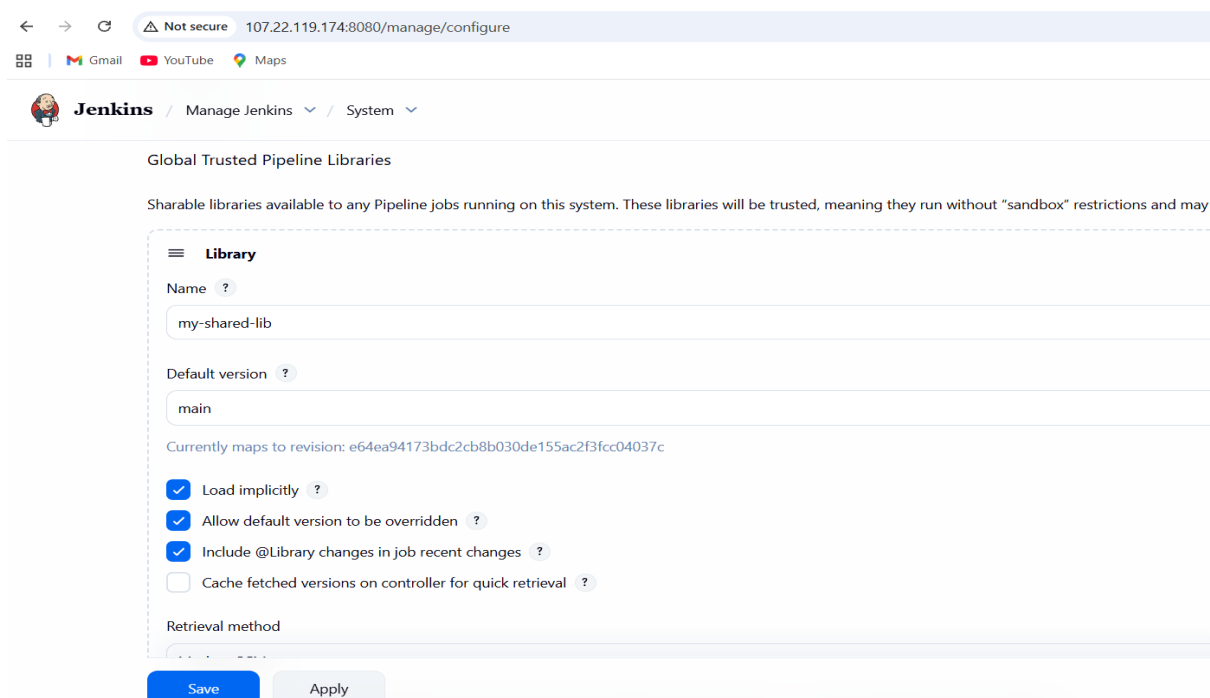
Create another file with the name var/deployStage.groovy






Go to Jenkins dashboard and go to manage Jenkins, and go to system and click on global trusted pipeline libraries click on add and give details

Name, version as branch name, select modern scm and give repo url.



← → ↻ Not secure 107.22.119.174:8080/manage/configure

🗖️ | 📧 Gmail | 📺 YouTube | 📍 Maps

 **Jenkins** / Manage Jenkins ▾ / System ▾

☐ Cache fetched versions on controller for quick retrieval ?

Retrieval method

Modern SCM

Loads a library from an SCM plugin using newer interfaces optimized for this purpose. The recommended option when available.

Source Code Management

Git

Project Repository ?

https://github.com/mujaheed00/jenkins-shared-library.git

Credentials ?

- none -

Behaviors

Discover branches ?


**Save** Apply

Click on save

Create a new job with the name shared-library and select type as pipeline.

← → ↻ Not secure 107.22.119.174:8080/view/all/newJob

🗖️ | 📧 Gmail | 📺 YouTube | 📍 Maps





 **Jenkins** / All ▾ / New Item

### New Item

Enter an item name

shared-library

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 p steps like archiving artifacts and sending email notifications.
-  **Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly k 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 environment 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 f folder creates a separate namespace, so you can have multiple things of the same name as long as they are in diff folders.

**OK**

In the pipeline give this pipeline script (about that you configured in global library in that global details you have given details about your github repository that you created your shared library).

```
@Library('my-shared-lib') _
```

```
pipeline {
```

```
    agent any
```

```
    stages {
```

```
        stage('Shared Library Execution') {
```

```
            steps {
```

```
                script {
```

```
                    buildStage()
```

```
                    testStage()
```

```
                    deployStage()
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
}
```

←

→

↺

⚠ Not secure


107.22.119.174:8080/job/shared-library/configure

📦

📧 Gmail

📺 YouTube

📍 Maps

 **Jenkins**

/ shared-library / Configuration

Configure

⚙️ General

🕒 Triggers

📜 Pipeline

🔧 Advanced

Pipeline script

Script ?

try sample

```
1 @Library('my-shared-lib') _
2
3 pipeline {
4   agent any
5
6   stages {
7     stage('Shared Library Execution') {
8       steps {
9         script {
10           buildStage()
11           testStage()
12           deployStage()
13         }
14       }
15     }
16   }
17 }
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save

Apply

Click on build now.

←

→

↺

⚠ Not secure


107.22.119.174:8080/job/shared-library/

📦

📧 Gmail

📺 YouTube

📍 Maps

 **Jenkins**

/ shared-library

Status

</> Changes

▶ Build Now

⚙️ Configure

🗑 Delete Pipeline

🔍 Full Stage View

📁 Stages

✎ Rename

🔍 Pipeline Syntax

Builds

🔍 Filter

Today

✅ #2 4:48 AM

❌ #1 4:45 AM

❌ shared-library

Stage View

Average stage times:  
(full run time: ~11s)

	Shared Library Execution	Build	Test	Deploy
#2 Jan 10 10:18 No Changes	99ms	3s	3s	3s
#1 Jan 10 10:15 No Changes				

Permalinks

- Last build (#1), 3 min 37 sec ago
- Last failed build (#1), 3 min 37 sec ago
- Last unsuccessful build (#1), 3 min 37 sec ago