

## 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.

The screenshot shows the Jenkins 'New Item' creation interface. At the top, the URL is 54.198.54.149:8080/view/all/newJob. Below the URL, there are links for Gmail, YouTube, and Maps. The main title is 'New Item'. A text input field is labeled 'Enter an item name' and contains 'declarative-pipeline'. Below this, a section titled 'Select an item type' lists four options: 'Freestyle project', 'Pipeline', 'Multi-configuration project', and 'Folder'. The 'Pipeline' option is highlighted with a light gray background. The 'Pipeline' description states: '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.' At the bottom of the form is a blue 'OK' button.

In that job click on gitscm 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

This screenshot shows the Jenkins declarative pipeline configuration page. The left sidebar has tabs for General, Triggers, Pipeline (which is selected), and Advanced. The main area is titled 'SCM' and shows 'Git' as the provider. It includes fields for 'Repository URL' containing 'https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git', 'Credentials' set to '- none -', and an 'Advanced' dropdown. There's also a '+ Add Repository' button and a 'Branches to build' section.

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

This screenshot shows the Jenkins 'Manage Jenkins' tools configuration page. It features a 'Maven' section with a 'Name' field set to 'Maven3'. A checkbox for 'Install automatically' is checked. Below it is a 'Install from Apache' section with a 'Version' field set to '3.9.12'. There are buttons for '+ Add Maven' and '+ Add Installer'. At the bottom are 'Save' and 'Apply' buttons.

## Add jdk in manage Jenkins, tools

JDK installations

JDK installations ^ Edited

+ Add JDK

**JDK**

Name:

JAVA\_HOME:

Install automatically ?

## Click on build now

Not secure 54.198.54.149:8080/job/declarative-pipeline/ All

**Jenkins / declarative-pipeline** Status declarative-pipeline Add descrip

</> Changes Last Successful Artifacts ncodeit-hello-world-3.0.war 3.97 MiB view

▷ Build Now Configure Delete Pipeline

Q Full Stage View Stages

Rename Pipeline Syntax

Builds \*\*\* Filter

Today #7 10:24 AM #6 10:20 AM

Average stage times: (full run time: ~12s)

	Declarative: Checkout SCM	Declarative: Tool Install	Checkout	Build	Test	Package	Archive Artifact	Declarative Post Action
#7 Jan 09 15:54	278ms	278ms	219ms	2s	1s	1s	97ms	87ms
#6 Jan 09 15:50	259ms	84ms	211ms	2s	4s	4s	185ms	61ms
#5 Jan 09 15:39	271ms	69ms	211ms	465ms failed	56ms failed	55ms failed	52ms failed	110ms
	306ms	681ms	235ms	5s	85ms	57ms	55ms	91ms

## 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.

The screenshot shows the Jenkins 'New Item' creation interface. At the top, the URL is 54.198.54.149:8080/view/all/newJob. Below the URL, there's a search bar with the text 'scripted-pipeline'. Under the heading 'Select an item type', there are four options: 'Freestyle project' (selected), 'Pipeline' (highlighted with a blue border), 'Multi-configuration project', and 'Folder'. Each option has a brief description below it. At the bottom of the form is a blue 'OK' button.

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**

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 ▾

This screenshot shows the Jenkins Pipeline configuration page. The left sidebar has 'Pipeline' selected. The main area is titled 'Definition' and shows 'Pipeline script from SCM'. Under 'SCM', 'Git' is selected. The 'Repository URL' is set to 'https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git'. There are no credentials defined. An 'Advanced' dropdown is present.

← → ⌛ 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

This screenshot shows the Jenkins Pipeline configuration page. The left sidebar has 'Pipeline' selected. The main area shows 'Branches to build' with '\*/master' specified. It includes a '+ Add Branch' button. Below that is 'Repository browser' set to '(Auto)'. Under 'Additional Behaviors', there is a '+ Add' button. The 'Script Path' is set to 'Jenkinsfile\_scriptbased'. A 'Lightweight checkout' checkbox is checked. At the bottom are 'Save' and 'Apply' buttons.

Click on build now.

The screenshot shows the Jenkins interface for the 'scripted-pipeline' job. On the left, there's a sidebar with options like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area is titled 'Stage View' and displays a timeline of stages: Setup ENV (89ms), Checkout (271ms), Build (4s), Test (3s), Package (3s), Archive (63ms), and Cleanup (53ms). Below this, a specific build is shown with a timestamp of Jan 09 16:06 and a note that there were no changes. The 'Permalinks' section provides links for Builds, Today, and a specific build (#1) from 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' dialog. It starts with a search bar for 'Enter an item name' containing 'parameterized-job'. Below it, a 'Select an item type' section lists four options: 'Freestyle project', 'Pipeline', 'Multi-configuration project', and 'Folder'. The 'Pipeline' option is highlighted. At the bottom of the dialog is an 'OK' button.

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

The screenshot shows the Jenkins job configuration page for a parameterized job. The 'General' tab is selected. A checkbox labeled 'This project is parameterized' is checked. A 'Choice Parameter' section is expanded, showing a parameter named 'MVN\_GOALS' with three choices: 'clean', 'compile', 'test', and 'package'. The 'Description' field is empty.

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

The screenshot shows the Jenkins job configuration page for a parameterized job. The 'General' tab is selected. A 'String Parameter' section is expanded, showing a parameter named 'BRANCH' with a 'Default Value' of 'master'. The 'Description' field is empty. At the bottom, there are 'Save' and 'Apply' buttons.

Click on gitscm and give your repo name

The screenshot shows the Jenkins Pipeline configuration page. The left sidebar has tabs: General, Triggers, Pipeline (which is selected), and Advanced. The main area is titled 'Configure' and says 'Define your Pipeline using Groovy directly or pull it from source control.' Under 'Definition', 'Pipeline script from SCM' is selected. In the 'SCM' section, 'Git' is chosen. Below it, 'Repositories' are listed with a single repository: 'Repository URL' set to `https://github.com/mujaheed00/spring3-mvc-maven-xml-hello-world-1.git`. There are also 'Credentials' and an 'Advanced' dropdown. At the bottom are 'Save' and 'Apply' buttons.

The screenshot shows the Jenkins Pipeline configuration page. The left sidebar has tabs: General, Triggers, Pipeline (selected), and Advanced. The main area is titled 'Configure' and shows 'Branches to build'. It has a 'Branch Specifier' field containing `*/master`, a '+ Add Branch' button, and a 'Repository browser' set to '(Auto)'. Under 'Additional Behaviours', there is a '+ Add' button. At the bottom are 'Script Path' set to `Jenkinsfile_parameterized` and a checked 'Lightweight checkout' checkbox. There are 'Save' and 'Apply' buttons 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  
nocodeit-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	666ms	152ms	2s	59ms	51ms
#7 Jan 09 16:28	635ms	274ms	1s	44ms	failed
#6 Jan 09 16:27	640ms	38ms	failed		
#5 Jan 09 16:26	103ms				

**Builds**

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

Filter

## 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\\_simplecutomerapp/tree/feature-1.1](https://github.com/betawins/sabear_simplecutomerapp/tree/feature-1.1)

Build a new job.

The screenshot shows the Jenkins 'New Item' creation interface. At the top, there's a browser header with 'Not secure' and the URL '54.198.54.149:8080/view/all/newJob'. Below the header, there are links for 'Gmail', 'YouTube', and 'Maps'. The main area has a 'Jenkins' logo and a 'New Item' title. A search bar contains the text 'declarative-pipeline1'. Below the search bar, there's a section titled 'Select an item type' with four options:

- Freestyle project**: Described as a 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**: Described as orchestrating 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**: Described as suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Described as creating 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.

At the bottom of the form is a blue 'OK' button.

Select gitscm and give your repo url

Not secure 54.198.54.149:8080/job/declarative-pipeline1/configure

Gmail YouTube Maps

**Jenkins / declarative-pipeline1 / Configuration**

### Configure

- General
- Triggers
- Pipeline**
- Advanced

**Repositories**

Repository URL: [https://github.com/mujaheed00/sabear\\_simplecustomerapp.git](https://github.com/mujaheed00/sabear_simplecustomerapp.git)

Credentials: - none -

Advanced

+ Add Repository

**Branches to build**

Branch Specifier (blank for 'any'): \*/master

**Save** **Apply**

Click on build now.

Not secure 54.198.54.149:8080/job/declarative-pipeline1

Gmail YouTube Maps

**Jenkins / declarative-pipeline1**

Status **declarative-pipeline1** Add des

- </> Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Stages
- Rename
- Pipeline Syntax

**Stage View**

Declarative: Checkout SCM	Declarative: Tool Install	Checkout	Build	Test	Package	Archive Artifact	Declarative Post Action
Average stage times: (full run time: ~7s)	193ms	120ms	372ms	2s	2s	3s	222ms
#2 Jan 09 16:42	200ms	118ms	342ms	1s	2s	203ms	85
#1 Jan 09 16:40	186ms	122ms	403ms	2s	2s	5s	242ms failed

**Permalinks**

- Last build (#1), 32 sec ago

Builds Today #2 11:12 AM

## 6. Create one Declarative job

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

## Create new job

The screenshot shows the Jenkins 'New Item' creation interface. At the top, there's a header bar with navigation icons and the URL '54.198.54.149:8080/view/all/newJob'. Below the header, the Jenkins logo is visible, followed by the path 'All / New Item'. The main title 'New Item' is centered above a form. The first field is 'Enter an item name' with the value 'trading-uri'. The next section is 'Select an item type' with four options: 'Freestyle project', 'Pipeline', 'Multi-configuration project', and 'Folder'. The 'Pipeline' option is currently selected. A large blue 'OK' button is at the bottom.

Click on gitscm and give your repository url

The screenshot shows the Jenkins configuration page for the 'trading-uri' job. The top navigation bar includes the Jenkins logo and the path 'trading-uri / Configuration'. On the left, a sidebar lists 'General', 'Triggers', 'Pipeline' (which is selected), and 'Advanced'. The main content area is titled 'Configure' and contains sections for 'Repositories' and 'Branches to build'. In the 'Repositories' section, the 'Repository URL' is set to 'https://github.com/mujaheed00/sabear\_simplecustomerapp.git' and 'Credentials' are listed as '- none -'. There's also a '+ Add Repository' button. In the 'Branches to build' section, the 'Branch Specifier (blank for \'any\')' is set to '/master'. At the bottom, there are 'Save' and 'Apply' buttons.

## Install plugin nodejs

The screenshot shows the Jenkins 'Plugins' page. In the top navigation bar, there are links for 'Manage Jenkins' and 'Plugins'. Below the navigation, there is a search bar with the query 'nodejs'. On the left, there is a sidebar with options: 'Updates', 'Available plugins' (which is selected and highlighted in grey), 'Installed plugins', and 'Advanced settings'. The main content area displays a table with one row for the 'NodeJS 1.6.6' plugin. The table has columns for 'Install', 'Name' (with a dropdown arrow), and 'Released'. The 'Released' column shows '1 mo 25 days ago'. A note below the table states: 'NodeJS Plugin executes NodeJS script as a build step.'

Go to manage Jenkins, tools and give nodejs version

The screenshot shows the Jenkins 'Tools' configuration page. In the top navigation bar, there are links for 'Manage Jenkins' and 'Tools'. Below the navigation, there is a button '+ Add NodeJS'. The main content area is titled 'NodeJS'. It has a 'Name' field containing 'NodeJS18' and a checked checkbox 'Install automatically'. Below this, there is a section titled 'Install from nodejs.org' with a 'Version' field set to 'NodeJS 18.19.0'. There is also a checkbox 'Force 32bit architecture' which is unchecked. At the bottom, there is a section for 'Global npm packages to install' with a text input field containing 'Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`'.

Click on build now.

The screenshot shows the Jenkins Pipeline Stage View for a job named "trading-uri". On the left, there's a sidebar with various Jenkins management links like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. Below that is a "Builds" section showing the last build (#7) from today at 12:10 PM. The main area is titled "Stage View" and displays a table of execution times for different stages. The table has columns for Declarative: Checkout SCM (205ms), Checkout (181ms), Install Dependencies (53ms), Build (49ms), Archive Artifacts (63ms), and Declarative: Post Actions (76ms). Two parallel stages are shown: one with 1 commit (213ms) and another with 3 commits (198ms). The stage with 3 commits has several failed steps highlighted in red, with execution times of 228ms, 56ms, 53ms, and 50ms.

## 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" creation interface. At the top, it says "Enter an item name" with "parallel-pipeline-demo" typed in. Below that, it says "Select an item type". There are three options listed: "Freestyle project" (selected), "Pipeline", and "Multi-configuration project". Each option has a brief description. At the bottom is a blue "OK" button.

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

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

General Triggers Pipeline Advanced

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.

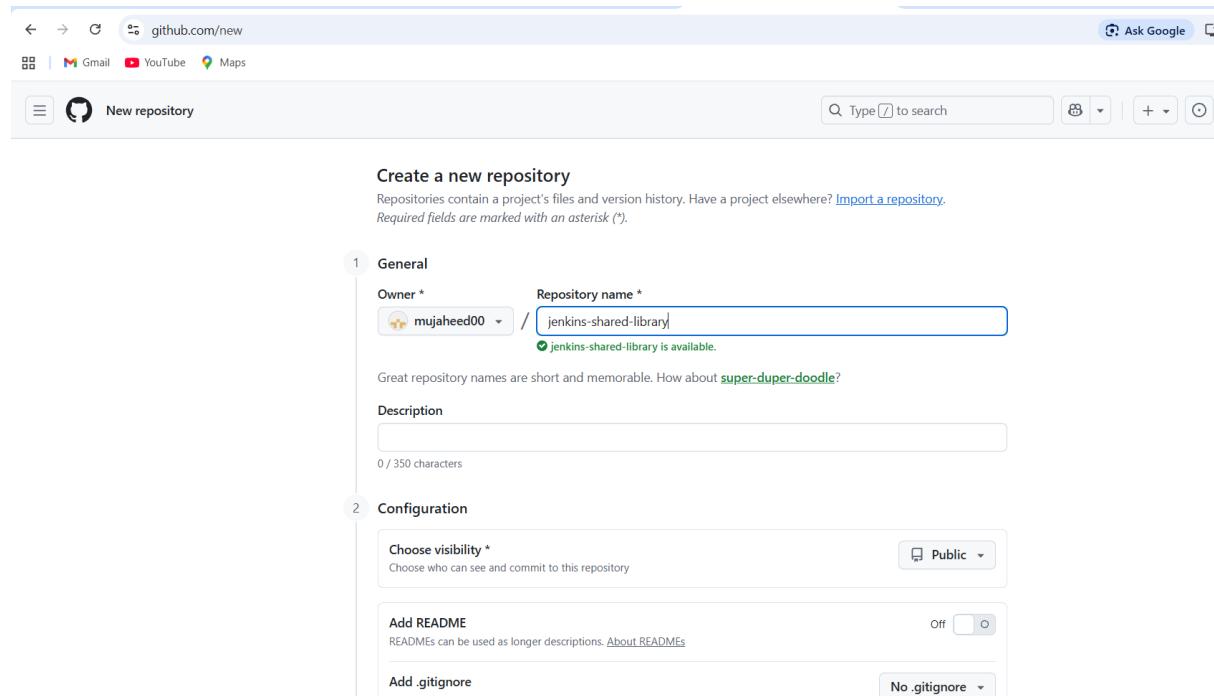
The screenshot shows the Jenkins interface for a pipeline named 'parallel-pipeline-demo'. On the left, there's a sidebar with various Jenkins management links like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area is titled 'parallel-pipeline-demo' and 'Stage View'. It displays a timeline with six stages: Checkout, Parallel Stage, Build, Test, Scan, and Deploy. Each stage has a progress bar indicating its duration: Checkout (85ms), Parallel Stage (104ms), Build (5s), Test (5s), Scan (5s), and Deploy (50ms). Below the timeline, a tooltip shows 'Average stage times: (full run time: ~7s)'. A small box indicates 'Jan 10 09:40 No Changes'. At the bottom, there's a 'Permalinks' section and a 'Builds' summary for today, showing one successful build (#1 at 4:10 AM).

Build,test,scan these 3 steps will 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' interface. It's a two-step process. Step 1: General. The 'Repository name' field contains 'jenkins-shared-library'. Step 2: Configuration. Under 'Choose visibility', 'Public' is selected. Under 'Add README', the switch is off. Under 'Add .gitignore', 'No .gitignore' is chosen. Other fields like 'Description' and 'Choose visibility' have placeholder text.

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 GitHub repository page for 'jenkins-shared-library'. The URL in the address bar is [github.com/mujaheed00/jenkins-shared-library/new/main](https://github.com/mujaheed00/jenkins-shared-library/new/main). The repository owner is 'mujaheed00'. The current file being viewed is 'vars/buildStage.groovy'. The code in the file is:

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

Create an another file with name var/testStage.groovy

The screenshot shows a GitHub repository page for 'jenkins-shared-library'. The URL in the address bar is [github.com/mujaheed00/jenkins-shared-library/new/main/vars](https://github.com/mujaheed00/jenkins-shared-library/new/main/vars). The repository owner is 'mujaheed00'. The current file being viewed is 'vars/testStage.groovy'. The code in the file is:

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

Create another file with the name var/deployStage.groovy

The screenshot shows a GitHub repository page for 'jenkins-shared-library'. The URL is [github.com/mujaheed00/jenkins-shared-library/new/main/vars](https://github.com/mujaheed00/jenkins-shared-library/new/main/vars). The page displays the 'vars' directory and the 'deployStage.groovy' file. The code in the file is:

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

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.

The screenshot shows the Jenkins 'Global Trusted Pipeline Libraries' configuration page. The URL is [Not secure 107.22.119.174:8080/manage/configure](http://107.22.119.174:8080/manage/configure). The page shows a library named 'my-shared-lib' with a default version of 'main'. The 'Load implicitly' checkbox is checked. Other options like 'Allow default version to be overridden' and 'Include @Library changes in job recent changes' are also checked. The 'Save' and 'Apply' buttons are at the bottom.

The screenshot shows the Jenkins system configuration page under 'Manage Jenkins' > 'System'. A 'Shared Library' configuration is being edited. The 'Retrieval method' is set to 'Modern SCM' (Git). The 'Project Repository' is set to 'https://github.com/mujaheed00/jenkins-shared-library.git'. Under 'Behaviors', the 'Discover branches' option is selected. At the bottom are 'Save' and 'Apply' buttons.

Click on save

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

The screenshot shows the Jenkins 'New Item' creation page. In the 'Enter an item name' field, 'shared-library' is typed. In the 'Select an item type' section, 'Pipeline' is selected. Other options shown include 'Freestyle project', 'Multi-configuration project', and 'Folder'. An 'OK' button is at the bottom.

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

Script ?

```
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
- shared-library**
- </> Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Stages
- Rename
- Pipeline Syntax

**Builds**

Q Filter /

Today

- #2 4:48 AM
- #1 4:45 AM

**Stage View**

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

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