

Experiment 4: Jenkin Pipeline and Maven

Aim: is to create pipeline and maven project using Jenkins

Procedure :

Steps to write here are

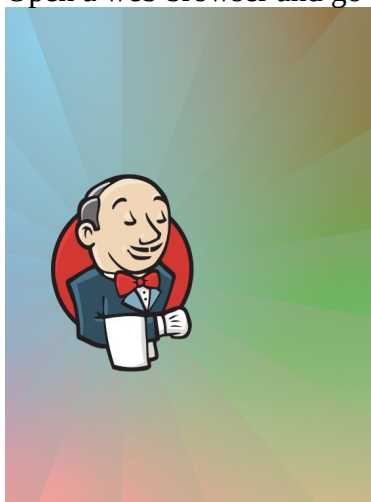
Part A - Snapshots of your project creation and execution with output generated for pipeline

1. **Jenkins installed and running** on your local machine or server. If not, you can follow the official Jenkins installation guide.
2. **Java installed** on your machine.

Step-by-Step Guide for Creating and Executing a Pipeline in Jenkins

Step 1: Log in to Jenkins

- Open a web browser and go to your Jenkins instance URL (typically <http://localhost:8080>).



Sign in to Jenkins

Username

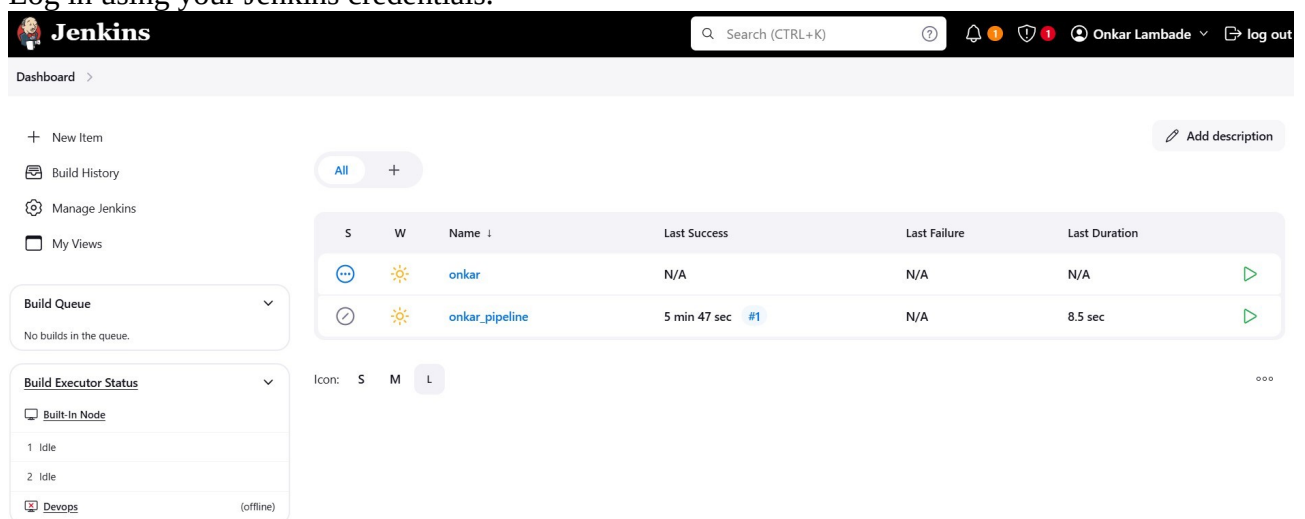
onkar

Password

☐ Keep me signed in

Sign in

- Log in using your Jenkins credentials.



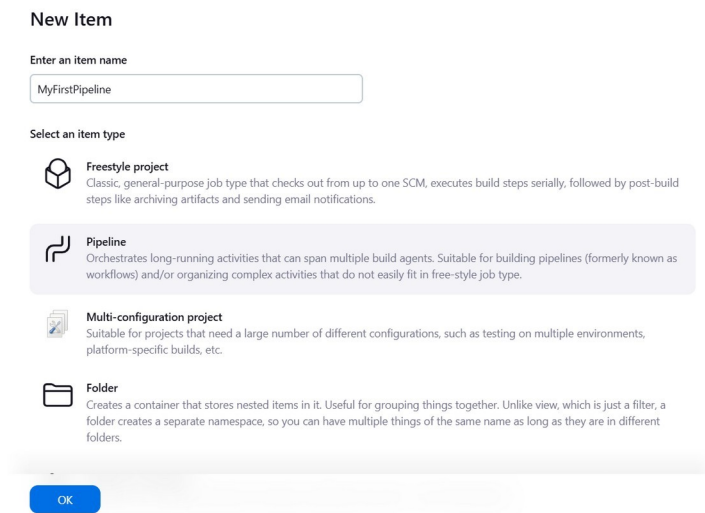
The screenshot shows the Jenkins dashboard with the following components:

- Header:** Jenkins logo, search bar (Search (CTRL+K)), and user profile (Onkar Lambade) with a log out button.
- Left Sidebar:** Dashboard, New Item, Build History, Manage Jenkins, My Views.
- Main Content Area:**
 - Build Queue:** No builds in the queue.
 - Build Executor Status:** 1 Idle, 2 Idle, Devops (offline).
 - Build History Table:**

S	W	Name ↓	Last Success	Last Failure	Last Duration
...	☀	onkar	N/A	N/A	N/A
✓	☀	onkar_pipeline	5 min 47 sec #1	N/A	8.5 sec

Step 2: Create a New Pipeline Project

1. From the Jenkins dashboard, click **“New Item”** on the left-hand menu.
2. Give your project a name, for example, MyFirstPipeline.
3. Select **“Pipeline”** as the project type.
4. Click **"OK"** to proceed.





New Item


Enter an item name


MyFirstPipeline

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.

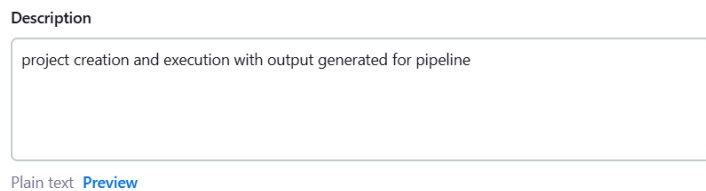
OK

Step 3: Configure the Pipeline

1. On the project configuration page, you can fill in the following fields:

- o **Description:** Describe what the pipeline does (optional).

General



Description

project creation and execution with output generated for pipeline

Plain text [Preview](#)

- o **Discard old builds:** You can check this option to limit the number of builds to keep.

☒ Discard old builds ?

Strategy

Log Rotation

Days to keep builds

if not empty, build records are only kept up to this number of days

5

Max # of builds to keep

if not empty, only up to this number of build records are kept

5

Advanced ▾

2. Scroll down to the "Pipeline" section:

- o **Definition:** Choose **"Pipeline script"** from the dropdown menu.

Definition

Pipeline script ▾

Script ?

```

1 pipeline {
2   agent any
3
4   stages {
5     stage('Checkout') {
6       steps {
7         echo 'Checking out code from SCM...'
8       }
9     }
10    stage('Build') {
11      steps {
12        echo 'Building the project...'
13      }
14    }
15    stage('Test') {
16      steps {
17        echo 'Running tests...'

```

try sample Pipeline... ▾

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

- o **Script:** In this field, you will define your pipeline script.

```

pipeline {
agent any

stages {
stage('Checkout') {
steps {
echo 'Checking out code from SCM...'
}
}
stage('Build') {
steps {
echo 'Building the project...'
}
}
stage('Test') {
steps {
echo 'Running tests...'
}
}
stage('Package') {

```

```

    steps {
        echo 'Packaging the application...'
    }
}
stage('Deploy') {
    steps {
        echo 'Deploying the application...'
    }
}
}
}

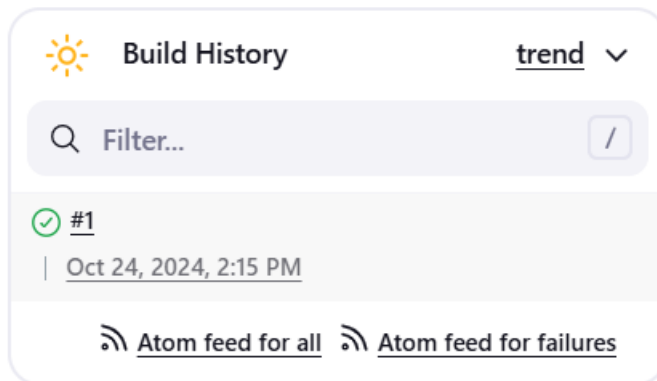
```

Step 5: Save and Build the Pipeline

1. Click on **“Save”** at the bottom of the configuration page.
2. On the project’s main page, you should see a **“Build Now”** option on the left sidebar. Click on it to trigger the pipeline.

Step 6: Check the Output

1. Once the build is triggered, you can click on the build number under the **“Build History”** section to see the build details.



2. Click **“Console Output”** to see the real-time logs of your pipeline execution. You should see the output messages:

✓ Console Output

```
Started by user Onkar Lambade
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/MyFirstPipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout)
[Pipeline] echo
Checking out code from SCM...
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Building the project...
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
Running tests...
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Package)
[Pipeline] echo
```

```
Packaging the application...
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] echo
Deploying the application...
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

3. If you see this output, congratulations! Your pipeline is working.

Part B -Snapshots of your project creation and execution with output generated for marven

1.Jenkins installed and running on your local machine or server. If not, you can follow the official Jenkins installation guide.

```
onkar@DESKTOP-D1SJIU7:~$ jenkins --version
2.462.2
```

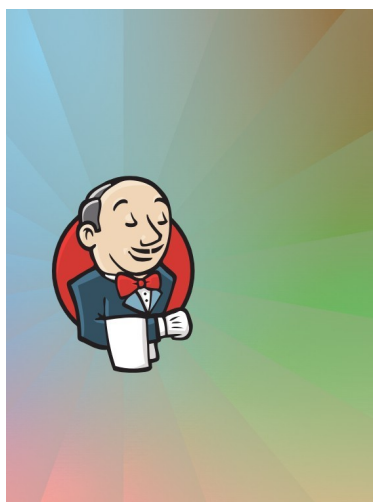
2.Java installed on your machine.

```
onkar@DESKTOP-D1SJIU7:~$ java --version
openjdk 11.0.24 2024-07-16
OpenJDK Runtime Environment (build 11.0.24+8-post-Ubuntu-1ubuntu322.04)
OpenJDK 64-Bit Server VM (build 11.0.24+8-post-Ubuntu-1ubuntu322.04, mixed mode, sharing)
```

Step-by-Step Guide for Creating and Executing a Pipeline in Jenkins:-

Step 1: Log in to Jenkins

- Open a web browser and go to your Jenkins instance URL (typically <http://localhost:8080>).



Sign in to Jenkins

Username

onkar

Password

.....

☐ Keep me signed in

Sign in

Log in using your Jenkins credentials.

The screenshot shows the Jenkins Dashboard. At the top, there's a search bar and a user profile for 'Onkar Lambade'. The main content area displays a table of builds. The table has columns for 'S' (Success), 'W' (Warning), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. There are two builds listed: 'onkar' and 'onkar_pipeline'. The 'onkar_pipeline' build shows a duration of 5 min 47 sec and 8.5 sec. On the left sidebar, there are links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. Below these, there's a 'Build Queue' section showing 'No builds in the queue.' and a 'Build Executor Status' section showing 'Built-In Node' with '1 Idle' and '2 Idle' executors, and 'Devops' (offline).

S	W	Name	Last Success	Last Failure	Last Duration
...	☀	onkar	N/A	N/A	N/A
🕒	☀	onkar_pipeline	5 min 47 sec #1	N/A	8.5 sec

Step 2: Install the Maven Plugin (if not already installed)

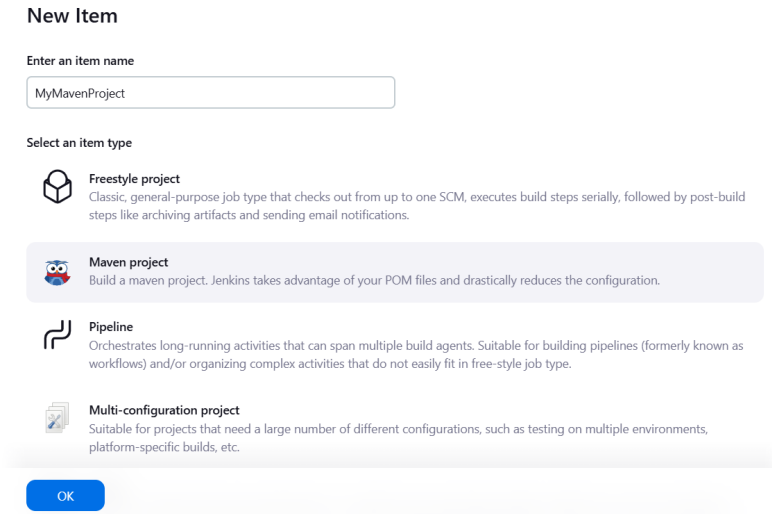
1. Go to "Manage Jenkins" from the dashboard.
2. Click on "Manage Plugins".
3. In the "Available" tab, search for "Maven Integration" or "Maven Plugin".
4. Install the plugin if it's not already installed. Restart Jenkins if prompted.

The screenshot shows the 'Manage Plugins' page in Jenkins. The 'Available plugins' tab is selected. A search bar at the top contains 'Maven plugin'. Below the search bar, there's a table of available plugins. The table has columns for 'Install', 'Name', and 'Released'. Two plugins are listed: 'Maven Integration 3.24' and 'Pipeline Maven Plugin API 1457.vf7a_de13b_c0d4'. The 'Maven Integration' plugin is checked for installation. The 'Pipeline Maven Plugin API' plugin is also checked for installation. The 'Released' column shows the time since the plugin was released: '15 hr ago' for 'Maven Integration' and '22 days ago' for 'Pipeline Maven Plugin API'.

Install	Name	Released
<input checked="" type="checkbox"/>	Maven Integration 3.24 Build Tools This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTS as well as the automated configuration of various Jenkins publishers such as Junit.	15 hr ago
<input checked="" type="checkbox"/>	Pipeline Maven Plugin API 1457.vf7a_de13b_c0d4 pipeline Maven Pipeline Maven Plugin API	22 days ago

Step 3: Create a New Maven Project

1. From the Jenkins dashboard, click on “New Item”.
2. Enter a name for your project, e.g., MyMavenProject.
3. Select “Maven Project” and click “OK”.





New Item


Enter an item name


MyMavenProject

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.

 **Maven project**
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

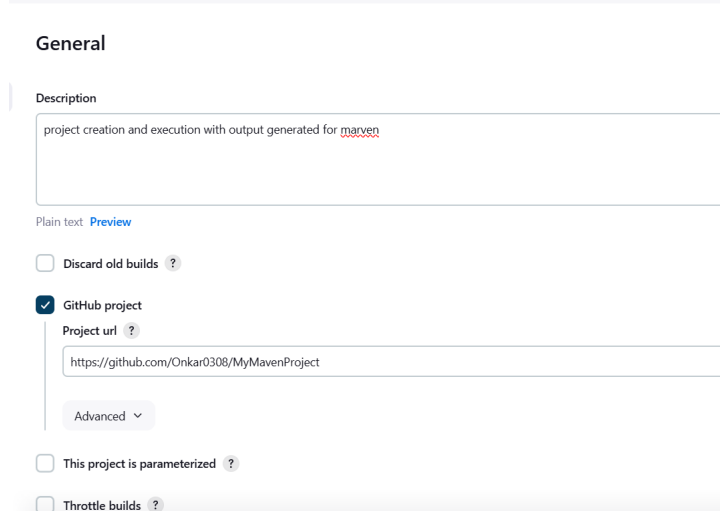
 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

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

OK

Step 4: Configure the Maven Project

1. **Description: Optionally provide a description of your project.**



General

Description

project creation and execution with output generated for [maven](#)

Plain text [Preview](#)

☐ Discard old builds ?

☒ **GitHub project**

Project url ?

<https://github.com/Onkar0308/MyMavenProject>

Advanced ▾

☐ This project is parameterized ?

☐ Throttle builds ?

2. **GitHub project: If your Maven project is hosted on GitHub, you can provide the URL here.**

IF you don't Have Github repository follow this steps:-

Step 1: Set Up Your Local Maven Project

1. **Install Maven (if you haven't already):**
 - o **Ensure that you have Maven installed on your machine. You can check by running `mvn -v` in your command line or terminal.**

```
onkar@DESKTOP-D1SJI07:~$ mvn -v
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.24, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.15.153.1-microsoft-standard-wsl2", arch: "amd64", family: "unix"
```

2. Create a New Maven Project:

- o Open your command line or terminal.
- o Navigate to the directory where you want to create your project:

```
onkar@DESKTOP-D1SJIU7:~$ mkdir mavenprojects
onkar@DESKTOP-D1SJIU7:~$ ls
1.py 172.25.90.203 main.tf mavenprojects my-terraform-project onkar_devops snap
onkar@DESKTOP-D1SJIU7:~$ cd mavenprojects/
```

Use the following Maven command to create a new project:

bash

Copy code

```
mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenProject -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

- ▮ Replace com.example with your desired group ID.
- ▮ Replace MyMavenProject with your desired artifact ID.

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ mvn archetype:generate -DgroupId=com.onkar -DartifactId=FirstMavenProject -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

3. Navigate to Your Project Directory:

bash

4.

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ ls
FirstMavenProject
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ cd FirstMavenProject/
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.onkar:FirstMavenProject >-----
[INFO] Building FirstMavenProject 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
```

Build Your Project (optional):

- o You can build your Maven project to ensure it's set up correctly:

bash

Copy code

```
mvn clean install
```

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ ls
FirstMavenProject
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ cd FirstMavenProject/
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.onkar:FirstMavenProject >-----
[INFO] Building FirstMavenProject 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
```

Step 2: Create a GitHub Repository

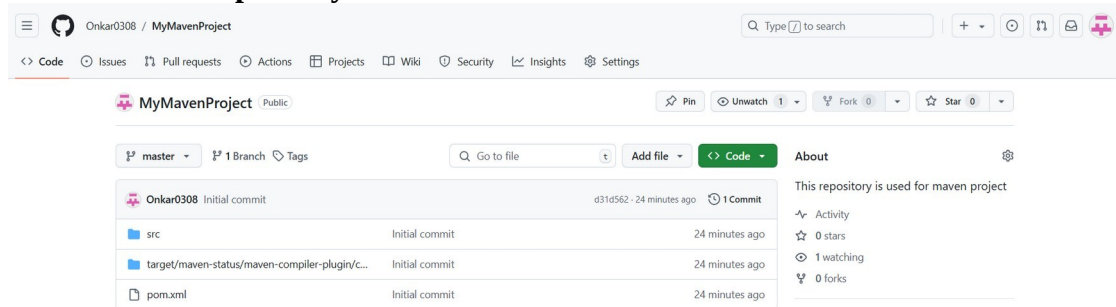
1. Log in to GitHub:

- o Open a web browser and go to [GitHub](#).
- o Log in to your account (or create one if you don't have an account).

2. Create a New Repository:

- o Click on the "+" icon in the top right corner and select "New repository".
- o Fill in the details:
 - ▮ Repository name: Enter a name for your repository, e.g., MyMavenProject.
 - ▮ Description: Add an optional description.
 - ▮ Public/Private: Choose whether you want your repository to be public or private.
- o Do not initialize with a README: Since you will be pushing an existing project.

- o Click "Create repository".



Step 3: Initialize Git in Your Local Project

1. Initialize Git:

- o In your command line or terminal, still within the project directory (MyMavenProject), run:

2.

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/onkar/mavenprojects/FirstMavenProject/.git/
```

Add Remote Repository:

- o Add the GitHub repository as a remote:

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git remote add origin https://github.com/Onkar0308/MyMavenProject.git
```

Replace your username with your GitHub username and adjust the URL according to the repository name.

3. Stage Your Files:

- o Stage all files in your project for commit:

4.

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git add .
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git commit -m "Initial commit"
[master (root-commit) d31d562] Initial commit
5 files changed, 70 insertions(+)
create mode 100644 pom.xml
create mode 100644 src/main/java/com/onkar/App.java
create mode 100644 src/test/java/com/onkar/AppTest.java
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst
```

Commit Your Changes:

- o Commit the staged files:

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git add .
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git commit -m "Initial commit"
[master (root-commit) d31d562] Initial commit
5 files changed, 70 insertions(+)
create mode 100644 pom.xml
create mode 100644 src/main/java/com/onkar/App.java
create mode 100644 src/test/java/com/onkar/AppTest.java
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst
create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst
```

Step 4: Push Your Project to GitHub

1. Push to GitHub:

- o Push your local commits to the GitHub repository:

```

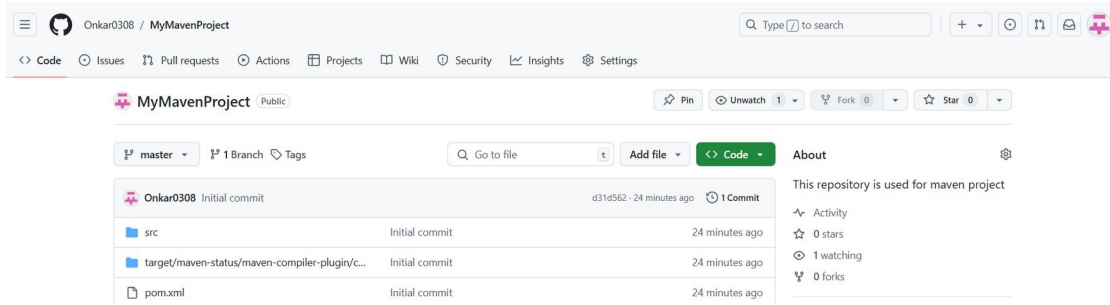
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git push -u origin master
Username for 'https://github.com': Onkar0308
Password for 'https://Onkar0308@github.com':
Enumerating objects: 21, done.
Counting objects: 100% (21/21), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (21/21), 1.74 KiB | 592.00 KiB/s, done.
Total 21 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Onkar0308/MyMavenProject.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$

```

Step 5: Verify Your Project on GitHub

1. Go back to GitHub:

- o Refresh your repository page. You should see your Maven project files uploaded to GitHub.



SS

3. Source Code Management:

- o Choose “Git” if your project is in a Git repository.
- o Enter the repository URL and credentials if needed.

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/Onkar0308/MyMavenProject

Credentials ?

- none -

+ Add

Advanced

Add Repository

Branches to build ?

4. Build Triggers:

- o You can check options like “Poll SCM” or “Build periodically” depending on your needs.

Build Triggers

☒ Build whenever a SNAPSHOT dependency is built ?

☐ Schedule build when some upstream has no successful builds ?

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

☒ Build periodically ?

Schedule ?

⚠ No schedules so will never run

☐ GitHub hook trigger for GITScm polling ?

☒ Poll SCM ?

Schedule ?

5. Build:

- o In the “Goals and options” section, enter the Maven goals, for example, clean install to clean and build the project.

Build

Root POM ?

pom.xml

Goals and options ?

clean install

Advanced ▾

Post Steps

☐ Run only if build succeeds

☐ Run only if build succeeds or is unstable

- o Optionally specify other parameters, such as -DskipTests to skip tests.

6. Advanced Settings: Click on “Advanced” if you need to set up specific Maven settings or profiles.

Step 5: Save the Configuration

- ▮ Click “Save” at the bottom of the page to store your project settings.

Step 6: Build the Maven Project

1. On the project’s main page, you will see a “Build Now” option on the left sidebar. Click it to trigger the Maven build.
2. You will see a build history entry with a timestamp.

```
Started by user Onkar Lambade
Running as SYSTEM
Building on the built-in node in workspace /var/lib/jenkins/workspace/MyMavenProject
Unpacking https://repo.maven.apache.org/maven2/org/apache-maven/apache-maven/3.9.9/apache-maven-3.9.9-bin.zip to
/var/lib/jenkins/tools/hudson.tasks.Maven_MavenInstallation/maven on Jenkins
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Onkar0308/MyMavenProject
> git init /var/lib/jenkins/workspace/MyMavenProject # timeout=10
Fetching upstream changes from https://github.com/Onkar0308/MyMavenProject
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- https://github.com/Onkar0308/MyMavenProject +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Onkar0308/MyMavenProject # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision d31d56247394d7ceb95e65ac9d1363c8eae2b25d (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f d31d56247394d7ceb95e65ac9d1363c8eae2b25d # timeout=10
Commit message: "Initial commit"
First time build. Skipping changelog.
Parsing POMs
Discovered a new module com.onkar:FirstMavenProject FirstMavenProject
Modules changed, recalculating dependency graph
Established TCN root on 27287
```

```

commons-1.14.jar 37207
<===[JENKINS REMOTING CAPACITY]===>channel started
Executing Maven: -B -f /var/lib/jenkins/workspace/MyMavenProject/pom.xml clean install
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.onkar:FirstMavenProject >-----
[INFO] Building FirstMavenProject 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] -----[ jar ]-----
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.pom
(5.3 kB at 17 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/35/maven-plugins-35.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/35/maven-plugins-35.pom (9.9 kB at 367
kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/35/maven-parent-35.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/35/maven-parent-35.pom (45 kB at 1.0 MB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/25/apache-25.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/25/apache-25.pom (21 kB at 642 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.jar
(36 kB at 776 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-3.3.1.pom (8.2 kB at 272 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom (8.1 kB at 476

[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-1.9.18.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-1.9.18.pom (2.7 kB at 167 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-util/1.9.18/maven-resolver-util-1.9.18.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-util/1.9.18/maven-resolver-util-1.9.18.jar (196 kB at 5.2 MB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-1.9.18.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-1.9.18.jar (157 kB at 4.2 MB/s)
[INFO] Installing /var/lib/jenkins/workspace/MyMavenProject/pom.xml to /var/lib/jenkins/.m2/repository/com/onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.pom
[INFO] Installing /var/lib/jenkins/workspace/MyMavenProject/target/FirstMavenProject-1.0-SNAPSHOT.jar to /var/lib/jenkins/.m2/repository/com/onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 12.695 s
[INFO] Finished at: 2024-10-24T14:43:22+05:30
[INFO] -----
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/lib/jenkins/workspace/MyMavenProject/pom.xml to com.onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/MyMavenProject/target/FirstMavenProject-1.0-SNAPSHOT.jar to com.onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.jar
channel stopped
Finished: SUCCESS

```

Conclusion : Frame your conclusion here

References: Include your references here

Rubrics: 5 marks for 1st part and 5 mark for second part - 100- 80 % - 5, 80- 60%- 4 and so on