Architecting Jenkins Pipeline

<u>Project Agenda:</u>

To build a Jenkins Pipeline to Implement CI/CD Workflow

<u>Description</u>:

Creating simple DevOps project to show how use Jenkins to set up a pipeline that will compile and test a Maven project .

<u>Tools required:</u>

GitHub – Git – Jenkins – Spring boot - Maven

<u>Background of the problem statement::</u>

Creating a GitHub repository, clone the GitHub repository, create Maven app with Spring boot and installed project file locally, Editing the project code using Git and push them to GitHub repository, creating pipeline in Jenkins .

<u>Developed By:</u>

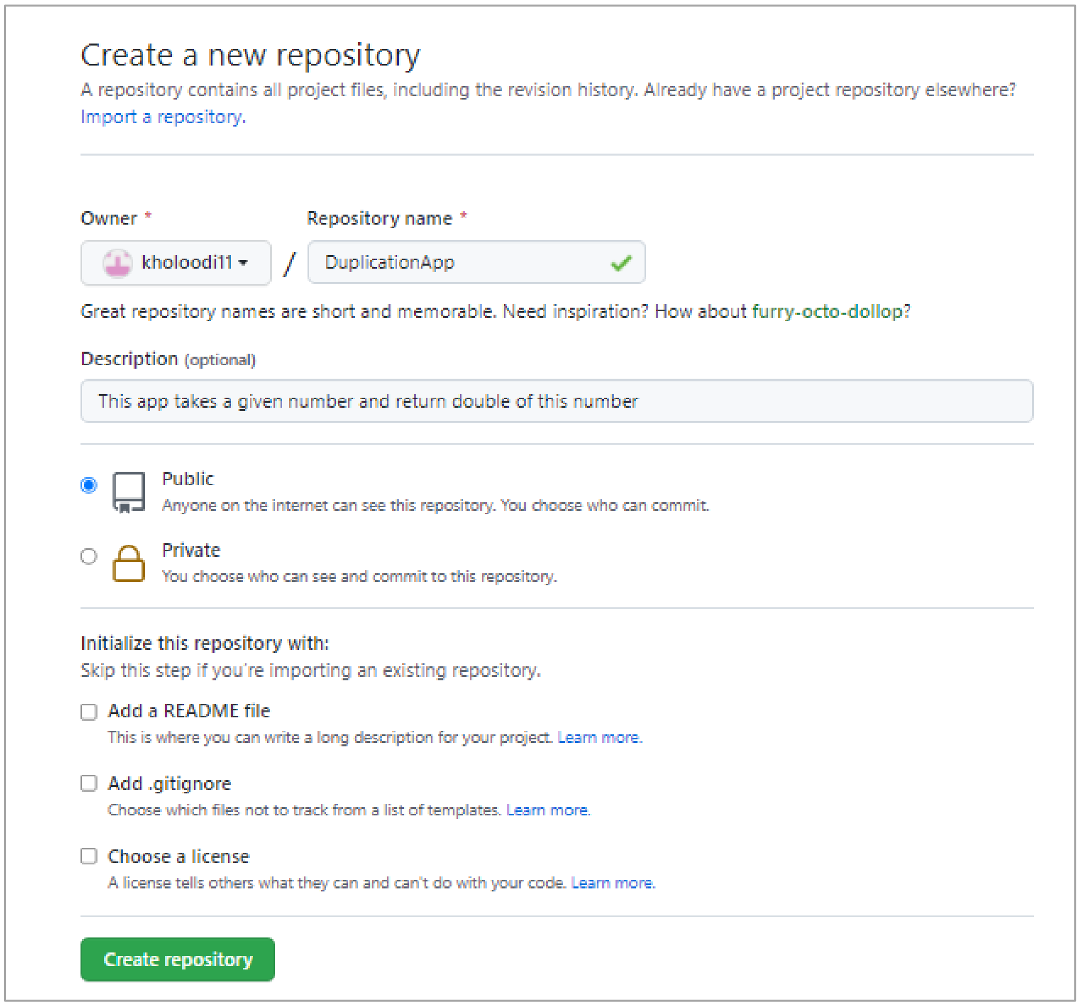
KHOLOOOD IBRAHEM

Steps to be followed:

1. Creating a Git repository for the project.
2. Generating a spring boot project.
3. Adding the code for word count to the repository.
4. Creating and committing a Jenkinsfile.
5. Creating a multistage pipeline in Jenkins

Step 1: Creating a Git repository for the project

- Login to your **Github** account
- Click on the plus icon next to the profile picture and select ***New repository*** from the drop-down menu



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * kholoodi11 / Repository name * DuplicationApp ✓

Great repository names are short and memorable. Need inspiration? How about furry-octo-dollop?

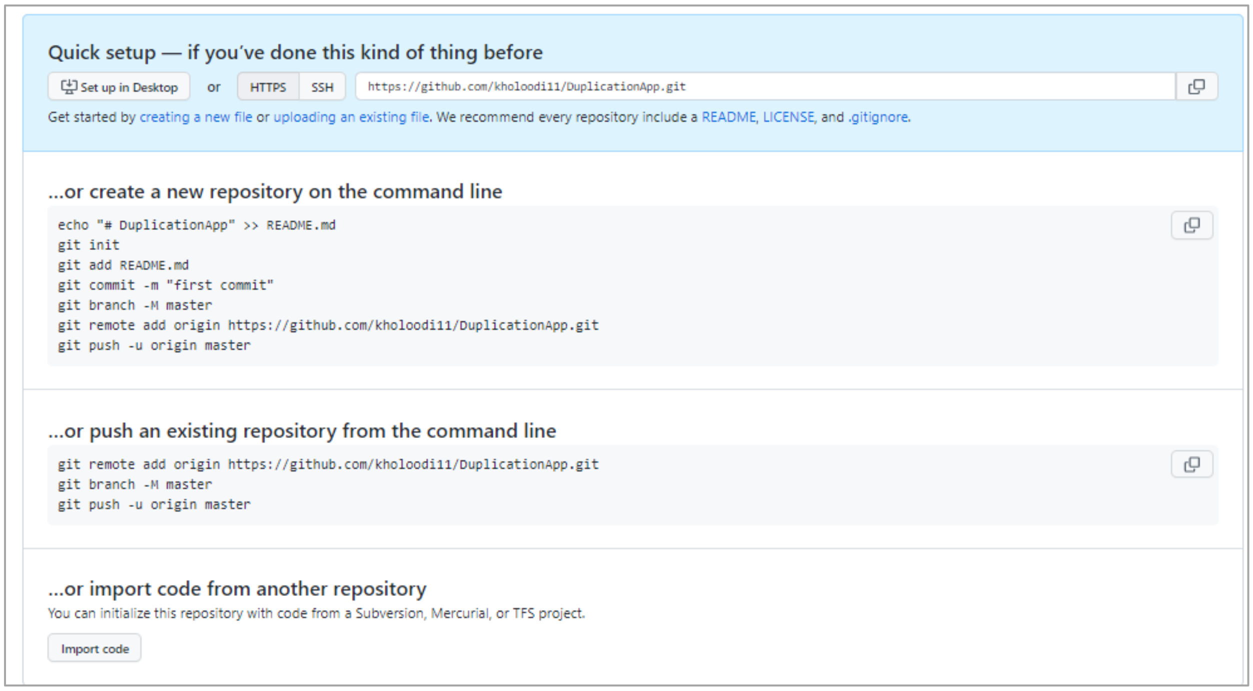
Description (optional): This app takes a given number and return double of this number

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private — You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

- [ ] Add a README file — This is where you can write a long description for your project. Learn more.
- [ ] Add .gitignore — Choose which files not to track from a list of templates. Learn more.
- [ ] Choose a license — A license tells others what they can and can't do with your code. Learn more.


Quick setup — if you've done this kind of thing before

```
https://github.com/kholoodi11/DuplicationApp.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# DuplicationApp" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M master
git remote add origin https://github.com/kholoodi11/DuplicationApp.git
git push -u origin master
```

…or push an existing repository from the command line

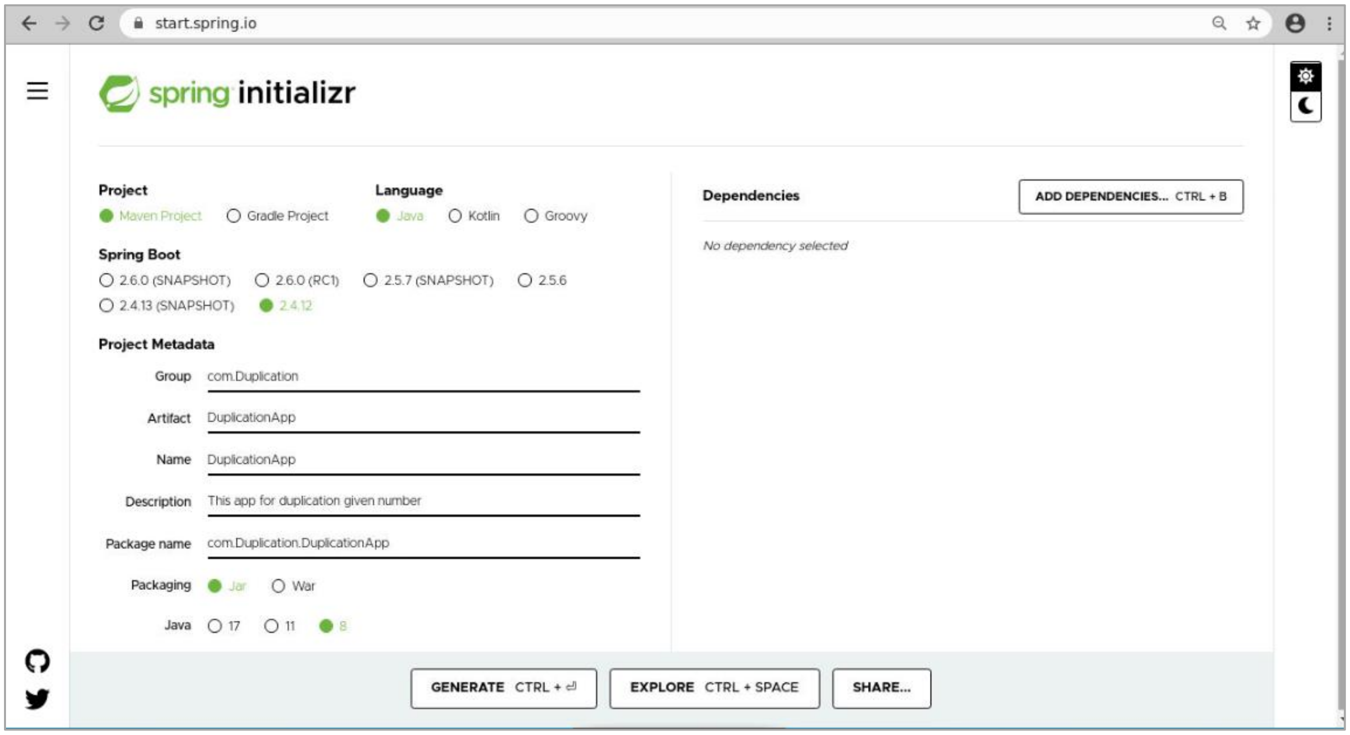
```
git remote add origin https://github.com/kholoodi11/DuplicationApp.git
git branch -M master
git push -u origin master
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Step 2: Generating a spring boot project

- Go to start.spring.io/
- Select Maven as the project type
- Fill Group and Artifact with appropriate values. For example, *com.Duplication* and *DuplicationApp*
- Add **Spring Web** to Dependencies
- Select Packaging: Jar
- Select Java: 8



Project: [✓ Maven Project] Gradle Project

Language: [✓ Java] Kotlin Groovy

Spring Boot: 2.6.0 (SNAPSHOT) 2.6.0 (RC1) 2.5.7 (SNAPSHOT) 2.5.6 2.4.13 (SNAPSHOT) [✓ 2.4.12]

Project Metadata

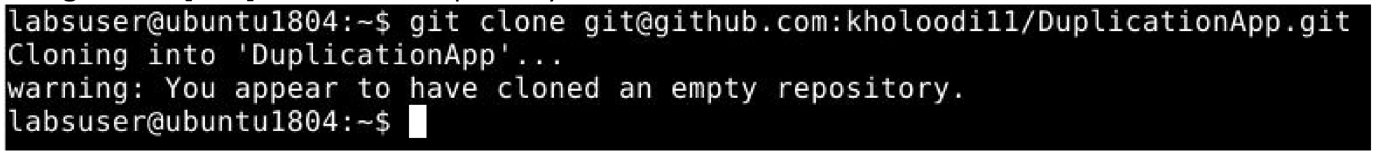
Group	com.Duplication
Artifact	DuplicationApp
Name	DuplicationApp
Description	This app for duplication given number
Package name	com.Duplication.DuplicationApp
Packaging	[✓ Jar] War
Java	17 11 [✓ 8]

Dependencies

No dependency selected

Step 3: Adding the code for word count to the repository

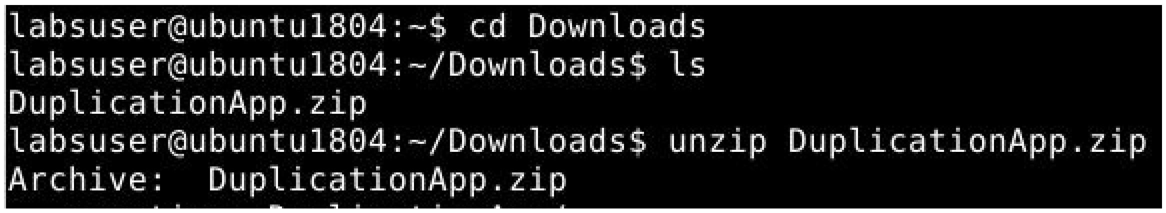
- Open the terminal
- Run **git clone [URL]** to clone the repository



```
labsuser@ubuntu1804:~$ git clone git@github.com:kholoodi11/DuplicationApp.git
Cloning into 'DuplicationApp'...
warning: You appear to have cloned an empty repository.
labsuser@ubuntu1804:~$
```

- Unzip the downloaded spring boot project to the cloned repository
  cd Downloads

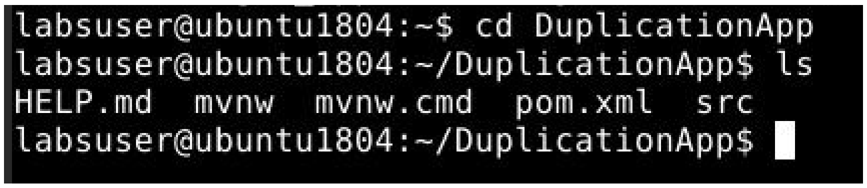
  unzip DuplicationApp.zip



```
labsuser@ubuntu1804:~$ cd Downloads
labsuser@ubuntu1804:~/Downloads$ ls
DuplicationApp.zip
labsuser@ubuntu1804:~/Downloads$ unzip DuplicationApp.zip
Archive:  DuplicationApp.zip
```

- Copy the contents of **DuplicationApp** folder present in downloads and paste it into repository folder
- On executing the following commands to see the contents of repository :
  cd DuplicationApp

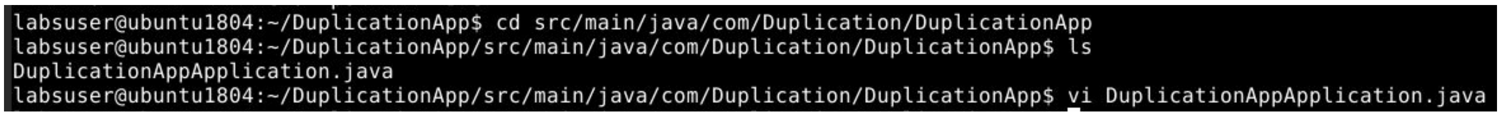
  ls



```
labsuser@ubuntu1804:~$ cd DuplicationApp
labsuser@ubuntu1804:~/DuplicationApp$ ls
HELP.md  mvnw  mvnw.cmd  pom.xml  src
labsuser@ubuntu1804:~/DuplicationApp$
```

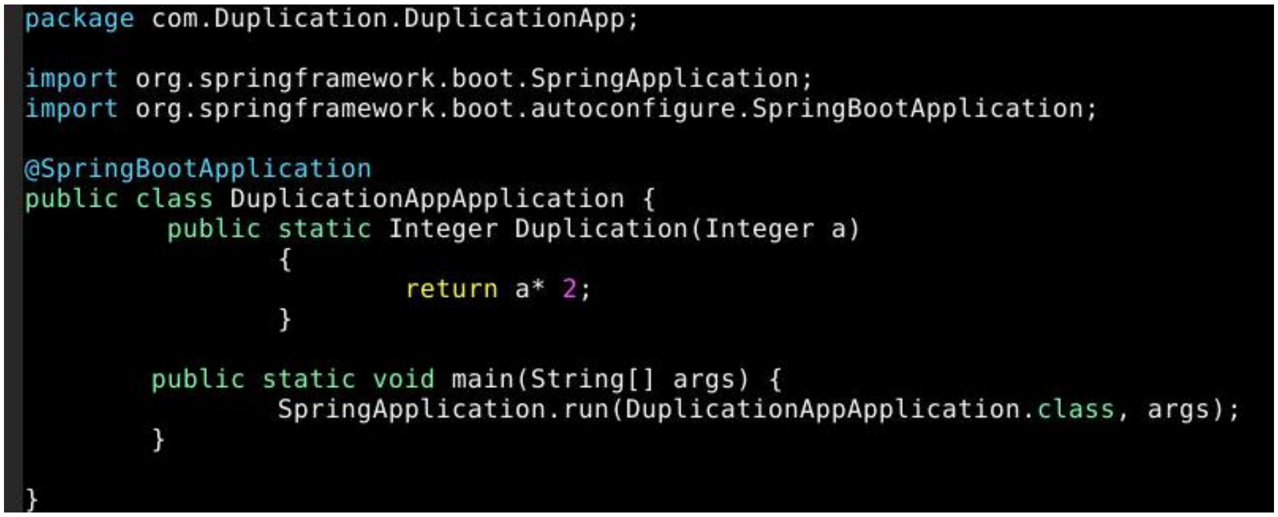
- Navigate to the *DuplicationApp* folder within the ***src*** folder
  cd src/main/java/com/Duplication/DuplicationApp

- Open *the DuplicationAppApplication.java* in a text editor
  vi DuplicationAppApplication.java



```
labsuser@ubuntu1804:~/DuplicationApp$ cd src/main/java/com/Duplication/DuplicationApp
labsuser@ubuntu1804:~/DuplicationApp/src/main/java/com/Duplication/DuplicationApp$ ls
DuplicationAppApplication.java
labsuser@ubuntu1804:~/DuplicationApp/src/main/java/com/Duplication/DuplicationApp$ vi DuplicationAppApplication.java
```

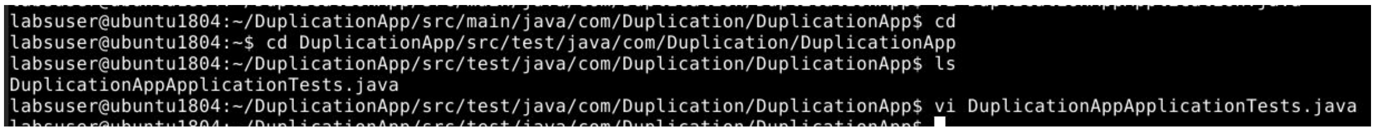
- Delete the existing content and add the following code to the file



```
package com.Duplication.DuplicationApp;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DuplicationAppApplication {
         public static Integer Duplication(Integer a)
                {
                        return a* 2;
                }

        public static void main(String[] args) {
                SpringApplication.run(DuplicationAppApplication.class, args);
        }

}
```

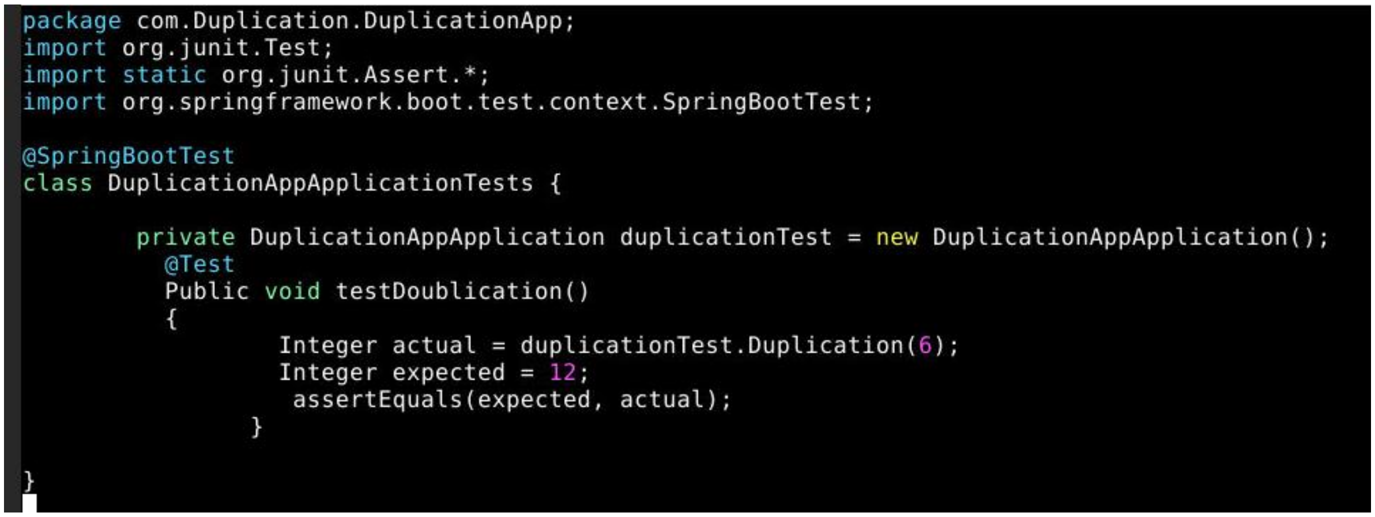
```
package com.Duplication.DuplicationApp;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication

public class DuplicationAppApplication {

             public static Integer Duplication(Integer a)

                 {

                   return a* 2;

                   }

     public static void main(String[] args) {

            SpringApplication.run(DuplicationAppApplication.class, args);

     }


}
```

- Save the file and exit using the command **[esc] shift+:wq**
- Navigate to the *DuplicationApp* folder within the ***test*** folder
  cd DuplicationApp/src/test/java/com/Duplication/DuplicationApp
- Open the *DuplicationAppApplicationTests.java* in a text editor
  vi DuplicationAppApplicationTests.java



```
labsuser@ubuntu1804:~/DuplicationApp/src/main/java/com/Duplication/DuplicationApp$ cd
labsuser@ubuntu1804:~$ cd DuplicationApp/src/test/java/com/Duplication/DuplicationApp
labsuser@ubuntu1804:~/DuplicationApp/src/test/java/com/Duplication/DuplicationApp$ ls
DuplicationAppApplicationTests.java
labsuser@ubuntu1804:~/DuplicationApp/src/test/java/com/Duplication/DuplicationApp$ vi DuplicationAppApplicationTests.java
```

- Delete the existing content and add the following code to the file



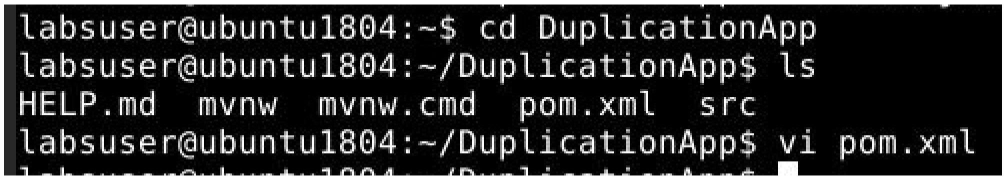
```
package com.Duplication.DuplicationApp;
import org.junit.Test;
import static org.junit.Assert.*;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class DuplicationAppApplicationTests {

        private DuplicationAppApplication duplicationTest = new DuplicationAppApplication();
          @Test
          Public void testDoublication()
          {
                  Integer actual = duplicationTest.Duplication(6);
                  Integer expected = 12;
                   assertEquals(expected, actual);
                }

}
```

```
  package com.Duplication.DuplicationApp;

  import org.junit.Test;

  import static org.junit.Assert.*;

  import org.springframework.boot.test.context.SpringBootTest;

  @SpringBootTest

  class DuplicationAppApplicationTests {

       private DuplicationAppApplication duplicationTest = new
DuplicationAppApplication();

         @Test

         Public void testDoublication()

         {

                Integer actual = duplicationTest.Duplication(6);

                Integer expected = 12;

                 assertEquals(expected, actual);

                  }

  }
```

- Save the file and exit the text editor using the command **[esc] shift+:wq**

- Run the following command to navigate to the pom file:

  cd DuplicationApp
  vi pom.xml



```
labsuser@ubuntu1804:~$ cd DuplicationApp
labsuser@ubuntu1804:~/DuplicationApp$ ls
HELP.md  mvnw  mvnw.cmd  pom.xml  src
labsuser@ubuntu1804:~/DuplicationApp$ vi pom.xml
```

- Add the following dependency in the <dependencies> section of the **pom.xml**

```
  <dependency>
                <groupId>junit</groupId>
                 <artifactId>junit</artifactId>
                  <version>${junit.version}</version>
                  <scope>test</scope>
    </dependency>
```

- Add the jacoco plugin to **pom.xml** with the following xml code in the <plugins> section:

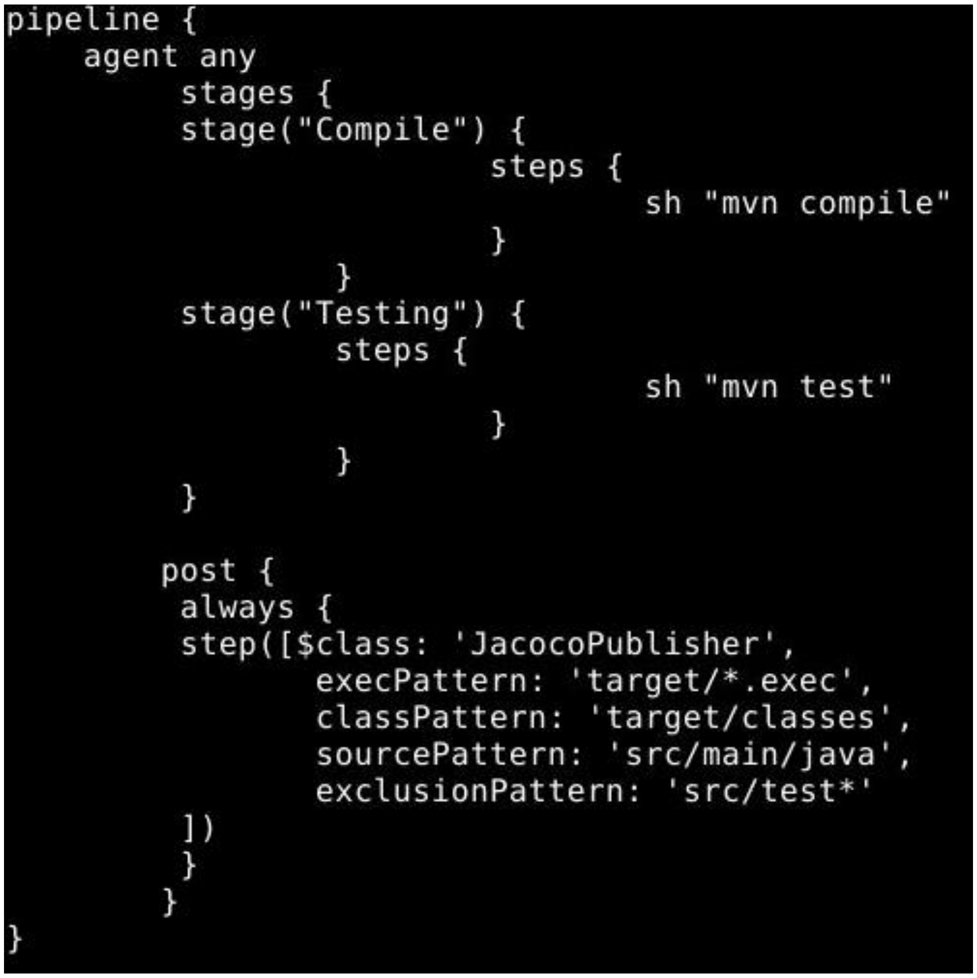
```
<plugin>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-compiler-plugin</artifactId>
                    <version>3.6.1</version>
                    <configuration>
                          <skipMain>true</skipMain>
                          <skip>true</skip>
                          <source>1.8</source>
                          <target>1.8</target>
                    </configuration>
              </plugin>
<plugin>
                    <groupId>org.jacoco</groupId>
                    <artifactId>jacoco-maven-plugin</artifactId>
                    <version>${jacoco.version}</version>
                    <executions>
```

```
                  <execution>
                        <id>prepare-agent</id>
                        <goals>
                              <goal>prepare-agent</goal>
                        </goals>
                  </execution>
                  <execution>
                        <id>report</id>
                        <phase>prepare-package</phase>
                        <goals>
                              <goal>report</goal>
                        </goals>
                  </execution>
                  <execution>
                        <id>post-unit-test</id>
                        <phase>test</phase>
                        <goals>
                              <goal>report</goal>
                        </goals>
                        <configuration>
<!-- Sets the path to the file which contains the execution data. -->
                              <dataFile>target/jacoco.exec</dataFile>
      <!-- Sets the output directory for the code coverage report. -->
                  <outputDirectory>target/jacoco-ut</outputDirectory>
                        </configuration>
                  </execution>
            </executions>
            <configuration>
            <systemPropertyVariables>
      <jacoco-agent.destfile>target/jacoco.exec</jacoco-agent.destfile>
                  </systemPropertyVariables>
            </configuration>
      </plugin>
```

- Save the file and exit the text editor using the command **[esc] shift+:wq**

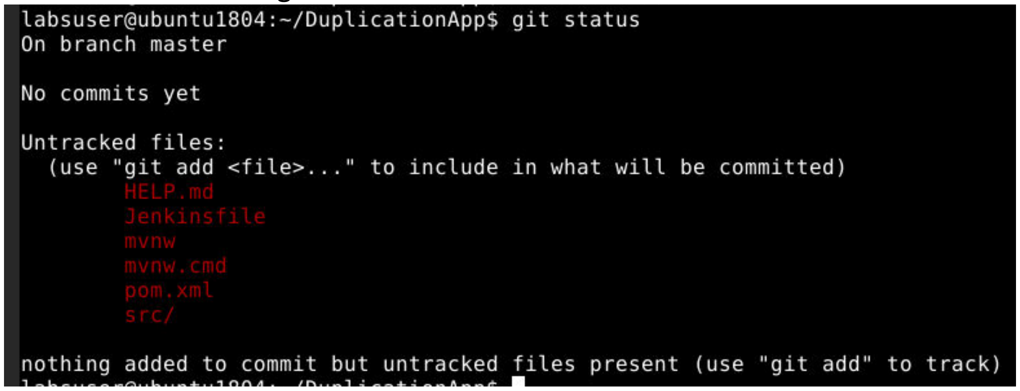
Step 4: Creating and committing a Jenkinsfile

- Navigate to the *DuplicationApp* root directory where the pom.xml is located **cd DuplicationApp**
- Open a new text file **vi Jenkinsfile** and add the following script to it.



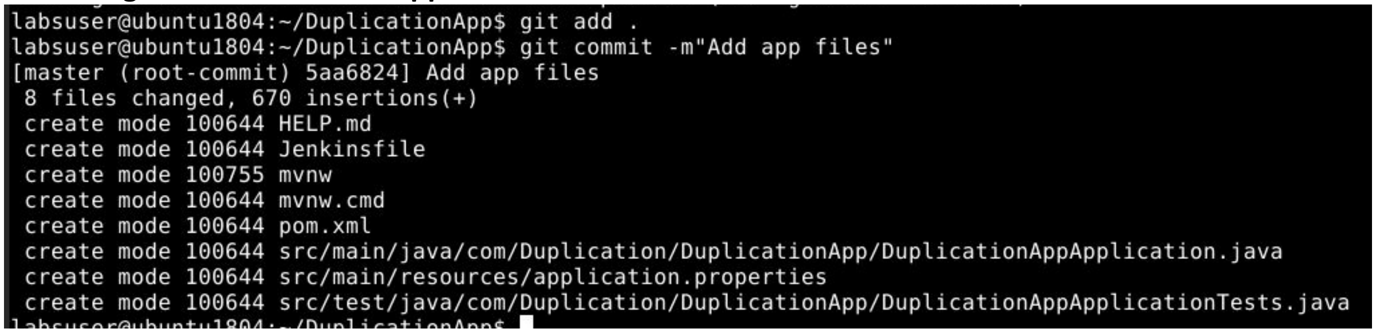
```
pipeline {
    agent any
         stages {
         stage("Compile") {
                         steps {
                                 sh "mvn compile"
                         }
                 }
         stage("Testing") {
                 steps {
                                 sh "mvn test"
                         }
                 }
         }

        post {
         always {
         step([$class: 'JacocoPublisher',
                execPattern: 'target/*.exec',
                classPattern: 'target/classes',
                sourcePattern: 'src/main/java',
                exclusionPattern: 'src/test*'
         ])
         }
        }
}
```

- Save the file as **Jenkinsfile** using the command **[esc] shift+:wq**
- Now check untracking files



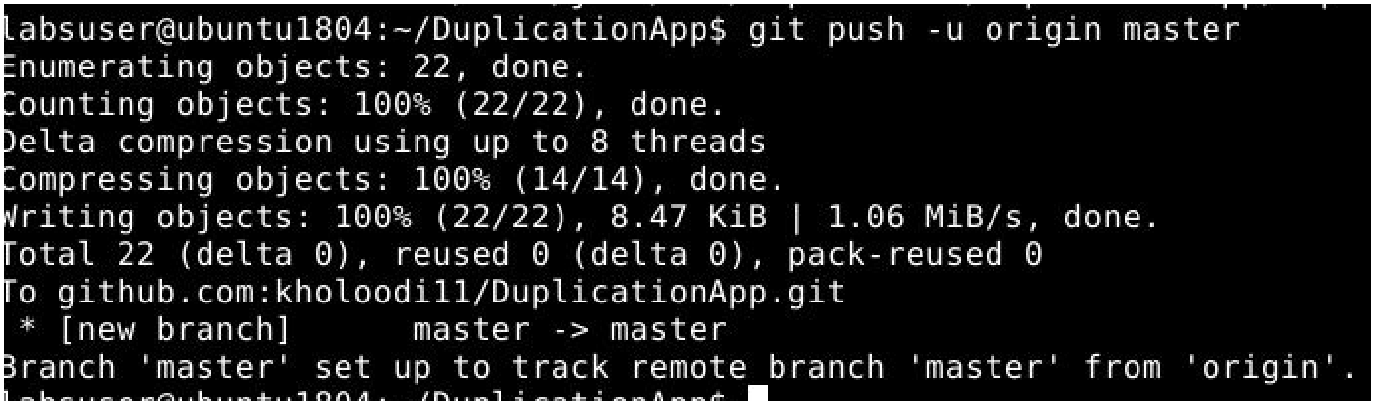
```
labsuser@ubuntu1804:~/DuplicationApp$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        HELP.md
        Jenkinsfile
        mvnw
        mvnw.cmd
        pom.xml
        src/

nothing added to commit but untracked files present (use "git add" to track)
```

- Commit the changes to the remote SCM
- Run **git add .**
- Run **git commit -m “Add App files”**

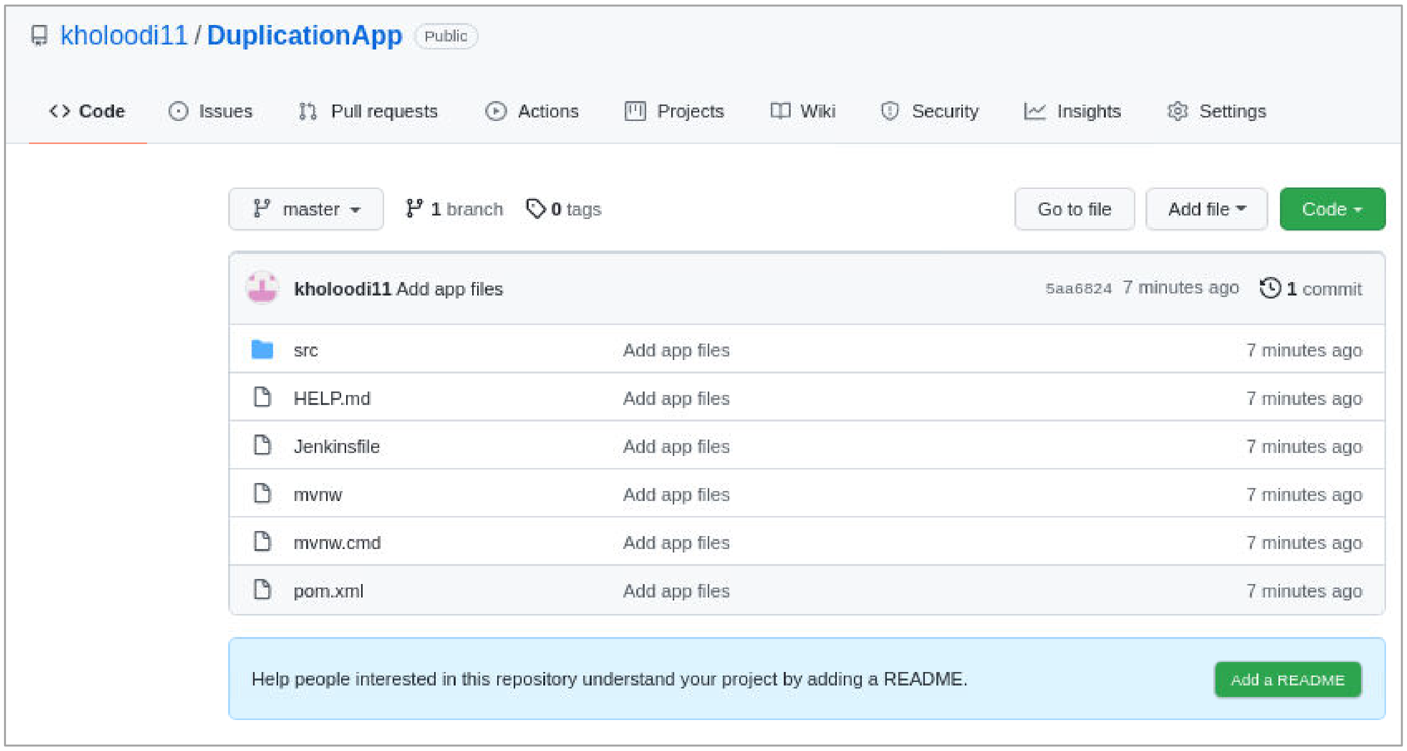


```
labsuser@ubuntu1804:~/DuplicationApp$ git add .
labsuser@ubuntu1804:~/DuplicationApp$ git commit -m"Add app files"
[master (root-commit) 5aa6824] Add app files
 8 files changed, 670 insertions(+)
 create mode 100644 HELP.md
 create mode 100644 Jenkinsfile
 create mode 100755 mvnw
 create mode 100644 mvnw.cmd
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/Duplication/DuplicationApp/DuplicationAppApplication.java
 create mode 100644 src/main/resources/application.properties
 create mode 100644 src/test/java/com/Duplication/DuplicationApp/DuplicationAppApplicationTests.java
```

- Run **git push -u origin master**



```
labsuser@ubuntu1804:~/DuplicationApp$ git push -u origin master
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (22/22), 8.47 KiB | 1.06 MiB/s, done.
Total 22 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:kholoodi11/DuplicationApp.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```



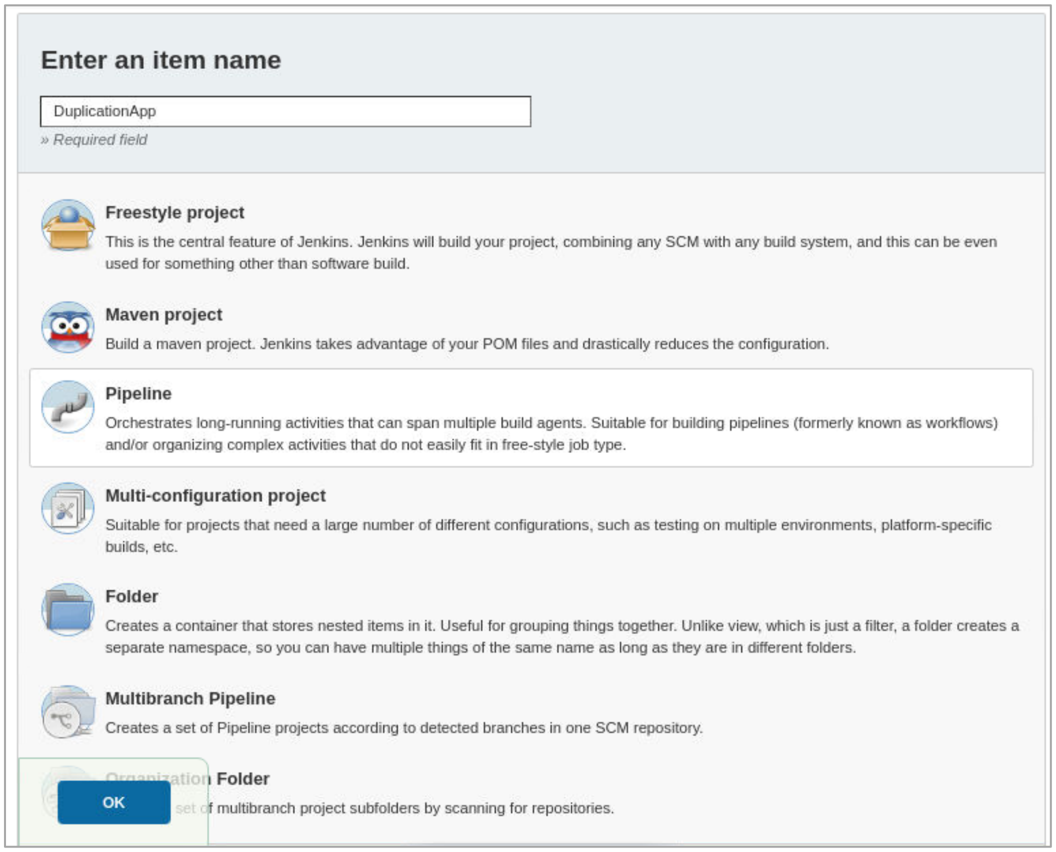
kholoodi11 Add app files 5aa6824 7 minutes ago 1 commit

src	Add app files	7 minutes ago
HELP.md	Add app files	7 minutes ago
Jenkinsfile	Add app files	7 minutes ago
mvnw	Add app files	7 minutes ago
mvnw.cmd	Add app files	7 minutes ago
pom.xml	Add app files	7 minutes ago

Help people interested in this repository understand your project by adding a README. Add a README

Step 5: Creating a multistage pipeline in Jenkins

- Go to Jenkins **dashboard**
- Click on ***New Item***
- Enter a **name** for your build job (Ex: review-analyser)
- Select ***Pipeline*** as the build job type



Enter an item name

DuplicationApp

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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.

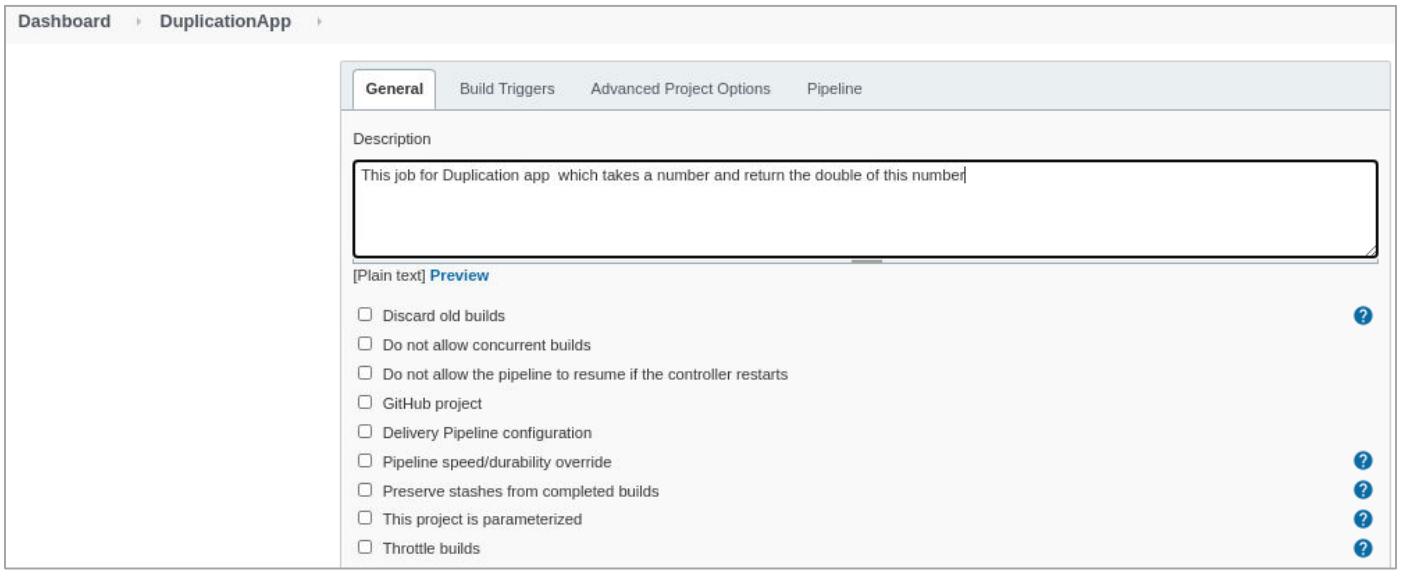
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

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

Organization Folder
set of multibranch project subfolders by scanning for repositories.

OK

- In **General** section add the description of project



Dashboard › DuplicationApp ›

General | Build Triggers | Advanced Project Options | Pipeline

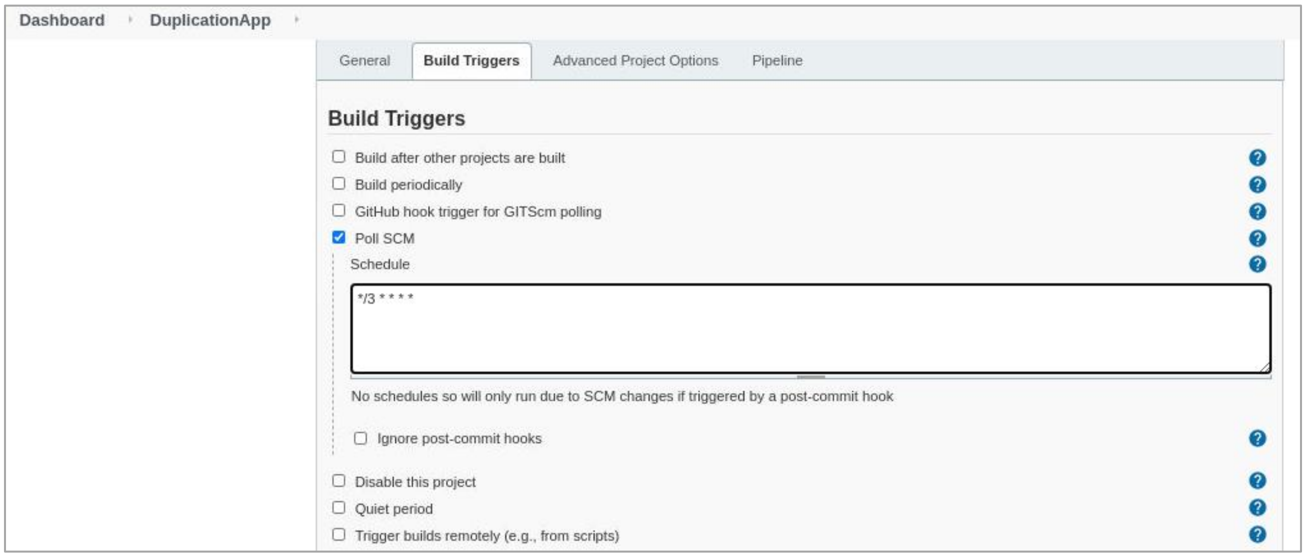
Description

This job for Duplication app which takes a number and return the double of this number

[Plain text] Preview

- ☐ Discard old builds
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts
- ☐ GitHub project
- ☐ Delivery Pipeline configuration
- ☐ Pipeline speed/durability override
- ☐ Preserve stashes from completed builds
- ☐ This project is parameterized
- ☐ Throttle builds

- For **Build Triggers** let build ever 3 minutes



Dashboard › DuplicationApp ›

General | Build Triggers | Advanced Project Options | Pipeline

Build Triggers

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

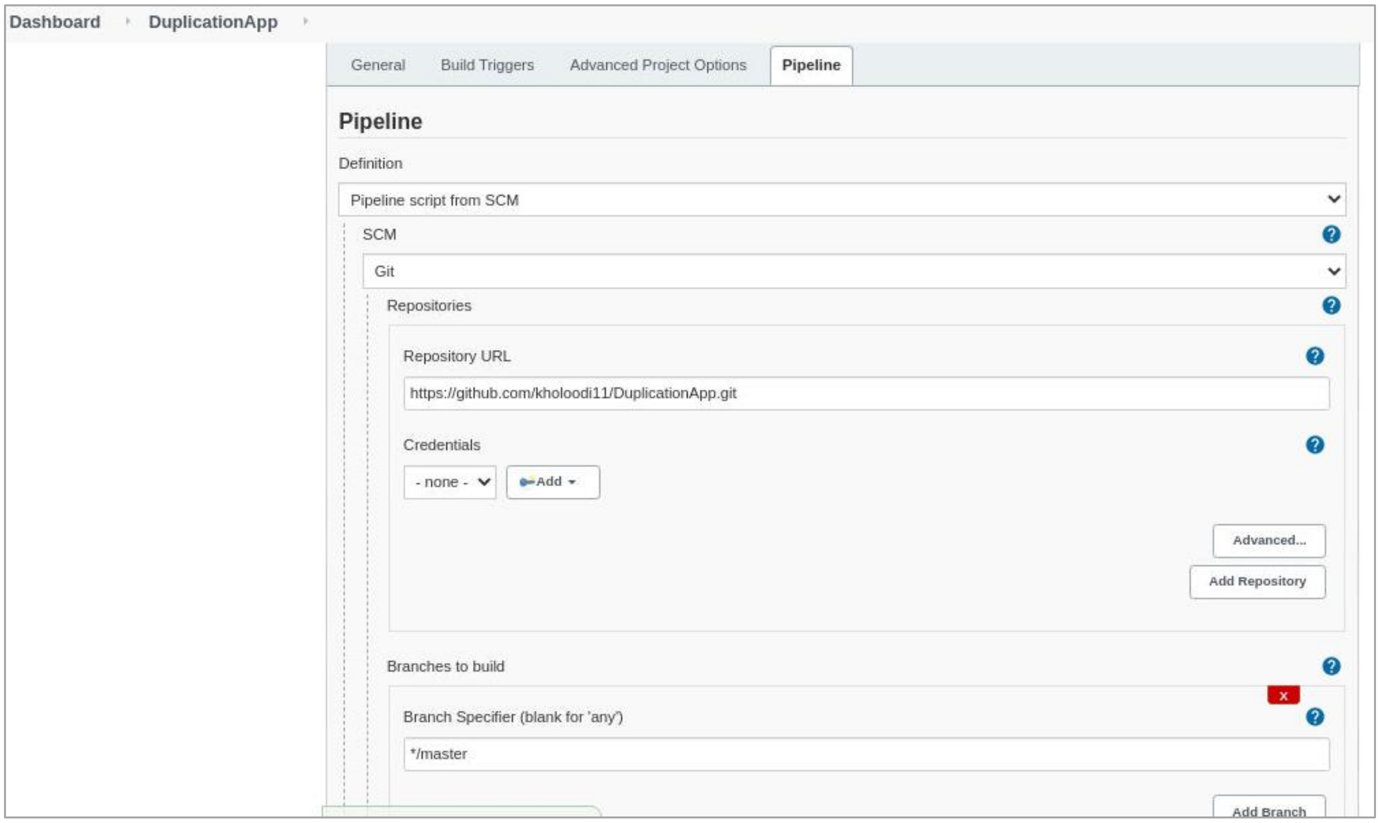
Schedule

```
*/3 * * * *
```

No schedules so will only run due to SCM changes if triggered by a post-commit hook

- [ ] Ignore post-commit hooks
- [ ] Disable this project
- [ ] Quiet period
- [ ] Trigger builds remotely (e.g., from scripts)

- In **Pipeline** section Change *Definition* from *Pipeline script* to ***Pipeline script from SCM***



Dashboard › DuplicationApp ›

General | Build Triggers | Advanced Project Options | Pipeline

Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/kholoodi11/DuplicationApp.git

Credentials: - none - | Add

Advanced... | Add Repository

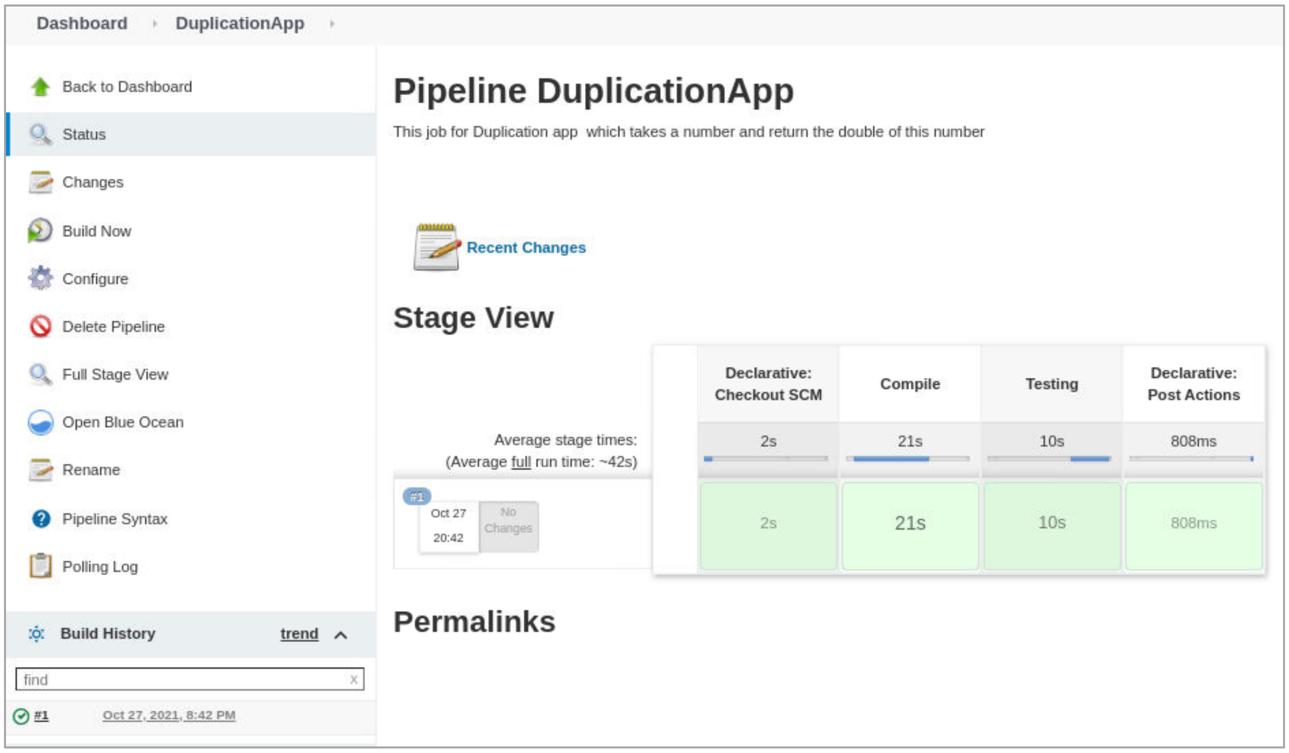
Branches to build

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

Add Branch

Click on Save

Waiting for 3 minutes for auto build



Dashboard › DuplicationApp ›

- Back to Dashboard
- Status
- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Open Blue Ocean
- Rename
- Pipeline Syntax
- Polling Log

Build History trend

find

#1 Oct 27, 2021, 8:42 PM

Pipeline DuplicationApp

This job for Duplication app which takes a number and return the double of this number

Recent Changes

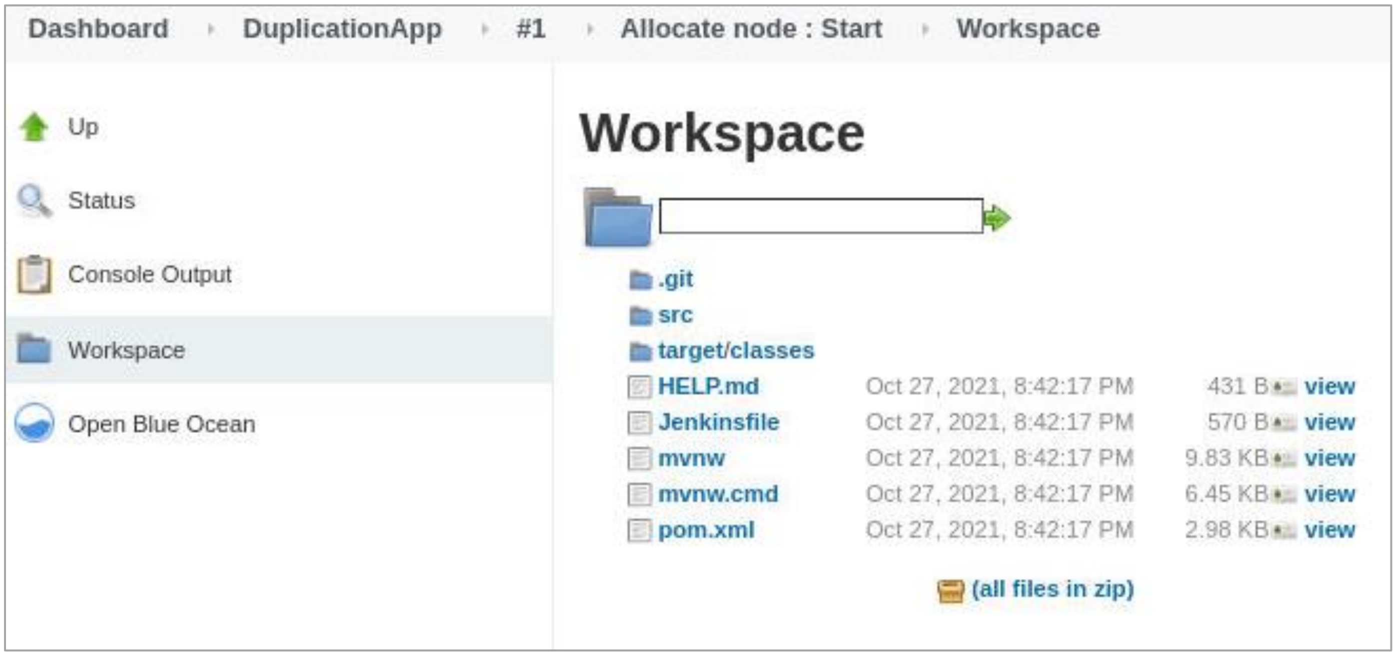
Stage View

Average stage times:
(Average full run time: ~42s)

	Declarative: Checkout SCM	Compile	Testing	Declarative: Post Actions
Average	2s	21s	10s	808ms
#1 Oct 27 20:42 No Changes	2s	21s	10s	808ms

Permalinks

- New, build succeed if we look at the **workspace,** we see the target directory is generated.



Dashboard › DuplicationApp › #1 › Allocate node : Start › Workspace

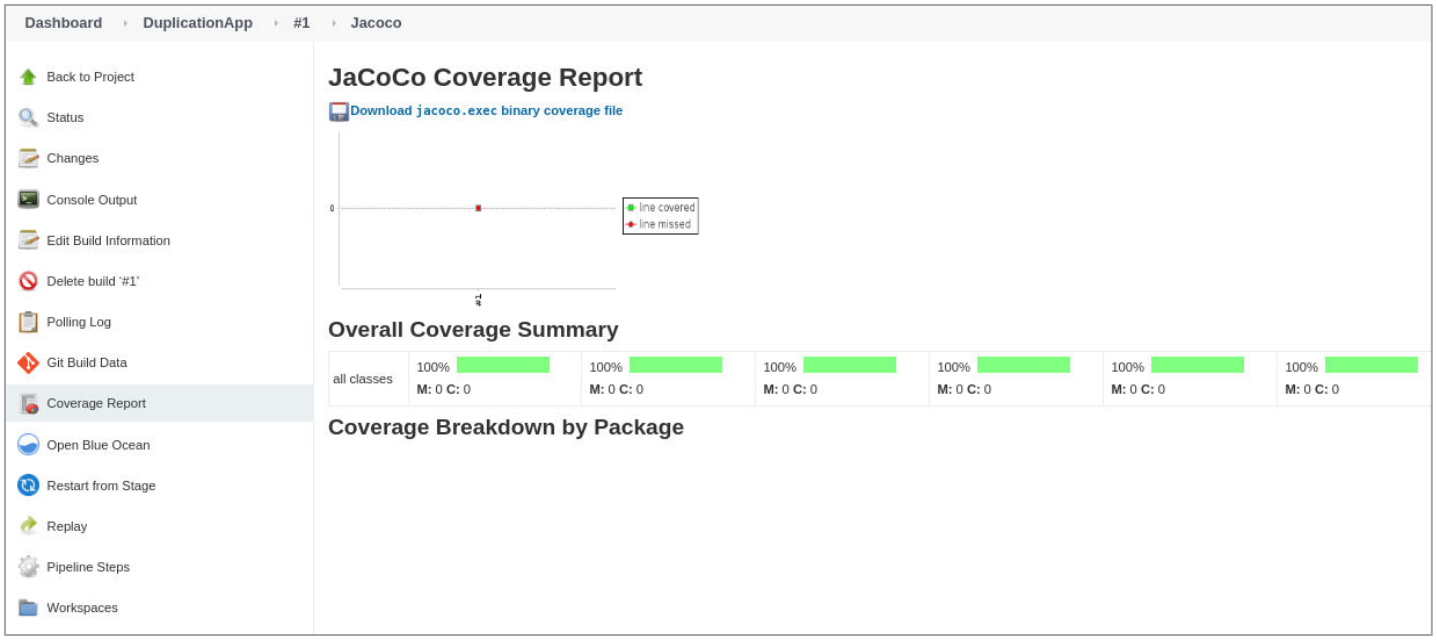
- Up
- Status
- Console Output
- Workspace
- Open Blue Ocean

Workspace

.git			
src			
target/classes			
HELP.md	Oct 27, 2021, 8:42:17 PM	431 B	view
Jenkinsfile	Oct 27, 2021, 8:42:17 PM	570 B	view
mvnw	Oct 27, 2021, 8:42:17 PM	9.83 KB	view
mvnw.cmd	Oct 27, 2021, 8:42:17 PM	6.45 KB	view
pom.xml	Oct 27, 2021, 8:42:17 PM	2.98 KB	view

(all files in zip)

- And look at to ***Coverage Report***



Dashboard › DuplicationApp › #1 › Jacoco

- Back to Project
- Status
- Changes
- Console Output
- Edit Build Information
- Delete build '#1'
- Polling Log
- Git Build Data
- Coverage Report
- Open Blue Ocean
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces

JaCoCo Coverage Report

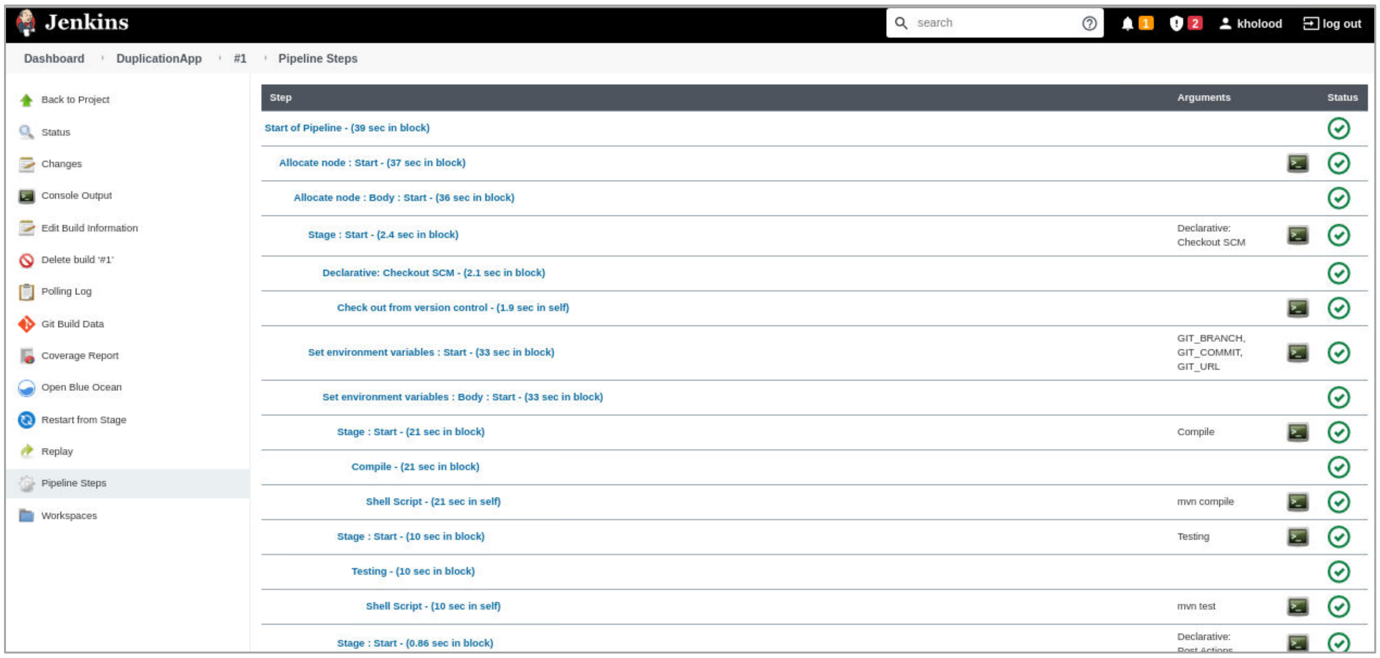
Download `jacoco.exec` binary coverage file

Overall Coverage Summary

all classes	100% M: 0 C: 0	100% M: 0 C: 0	100% M: 0 C: 0	100% M: 0 C: 0	100% M: 0 C: 0	100% M: 0 C: 0

Coverage Breakdown by Package

- And when click on ***Pipeline Steps*** we can see all pipeline steps
-



Jenkins · search · 1 · 2 · kholood · log out

Dashboard › DuplicationApp › #1 › Pipeline Steps

- Back to Project
- Status
- Changes
- Console Output
- Edit Build Information
- Delete build '#1'
- Polling Log
- Git Build Data
- Coverage Report
- Open Blue Ocean
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces

Step	Arguments	Status
Start of Pipeline - (39 sec in block)		✓
Allocate node : Start - (37 sec in block)		✓
Allocate node : Body : Start - (36 sec in block)		✓
Stage : Start - (2.4 sec in block)	Declarative: Checkout SCM	✓
Declarative: Checkout SCM - (2.1 sec in block)		✓
Check out from version control - (1.9 sec in self)		✓
Set environment variables : Start - (33 sec in block)	GIT_BRANCH, GIT_COMMIT, GIT_URL	✓
Set environment variables : Body : Start - (33 sec in block)		✓
Stage : Start - (21 sec in block)	Compile	✓
Compile - (21 sec in block)		✓
Shell Script - (21 sec in self)	mvn compile	✓
Stage : Start - (10 sec in block)	Testing	✓
Testing - (10 sec in block)		✓
Shell Script - (10 sec in self)	mvn test	✓
Stage : Start - (0.86 sec in block)	Declarative: Post Actions	✓