

ASSIGNMENT 2

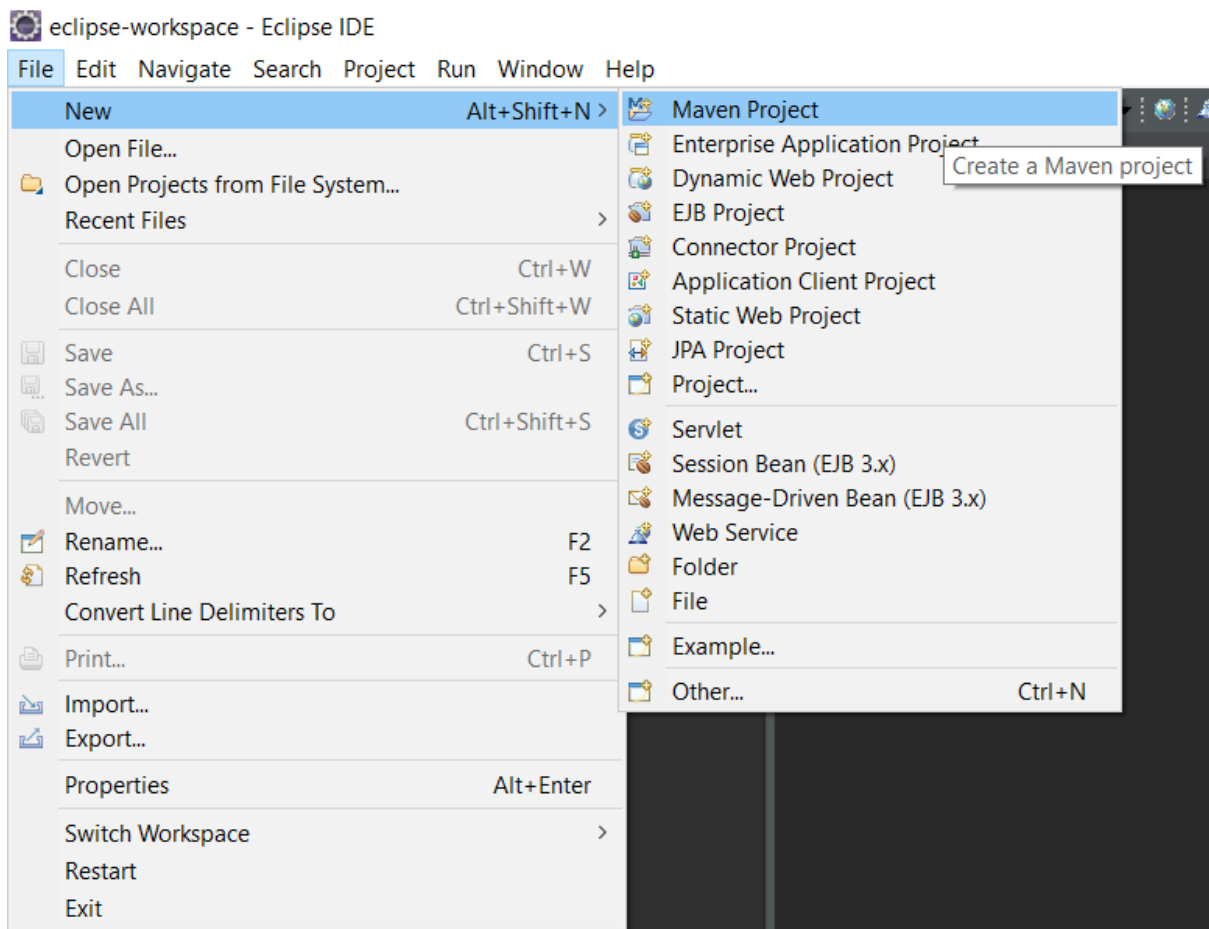
Submitted by: Hridyanshu
Roll Number: R171218047
SAP ID: 500068500

- **Pushing a Maven project build to the Nexus repository.**

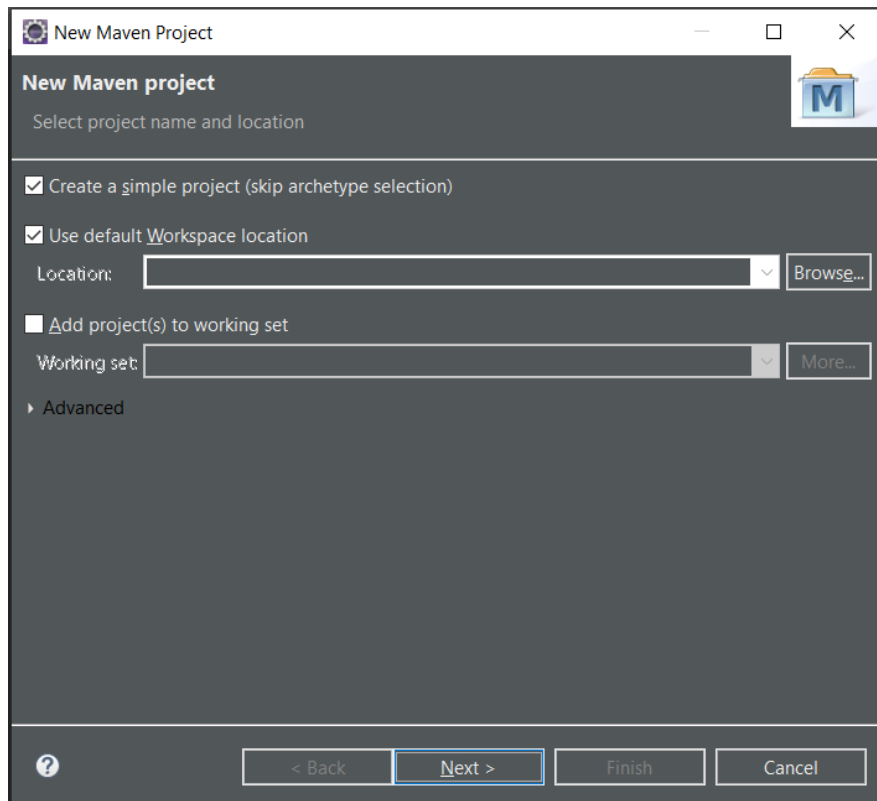
The goal here is to create a Maven project using Eclipse IDE and then finally pushing its build to the Nexus repository using Jenkinsfile.

To achieve the goal, the steps that are needed to be followed are:

1. Create a new Maven Project using Eclipse IDE.

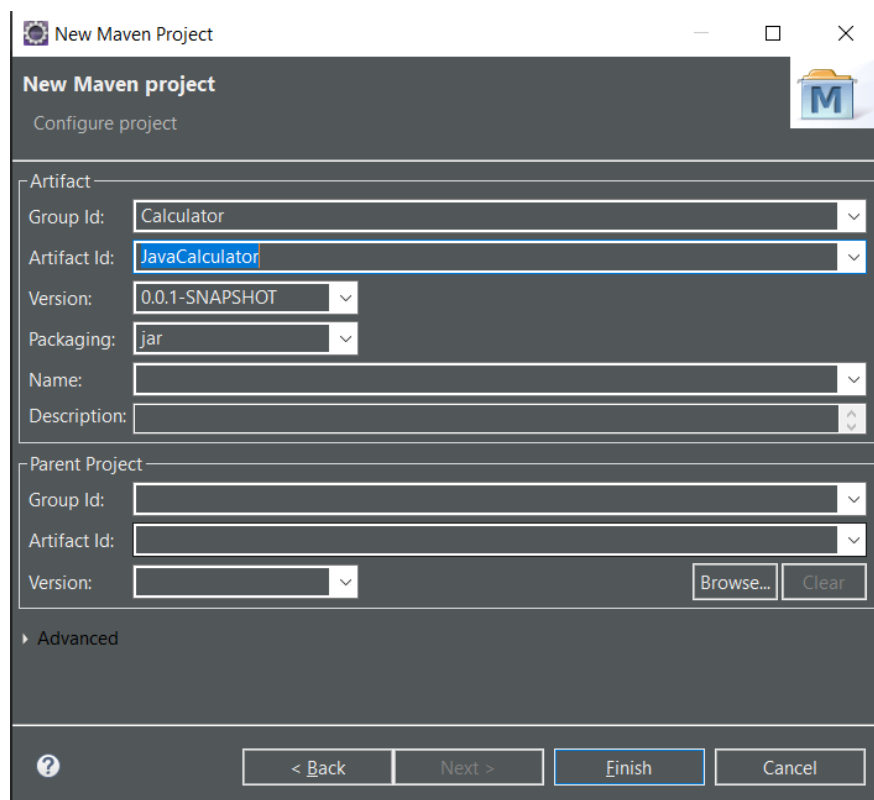


Check the “create a simple project” checkbox and click Next.



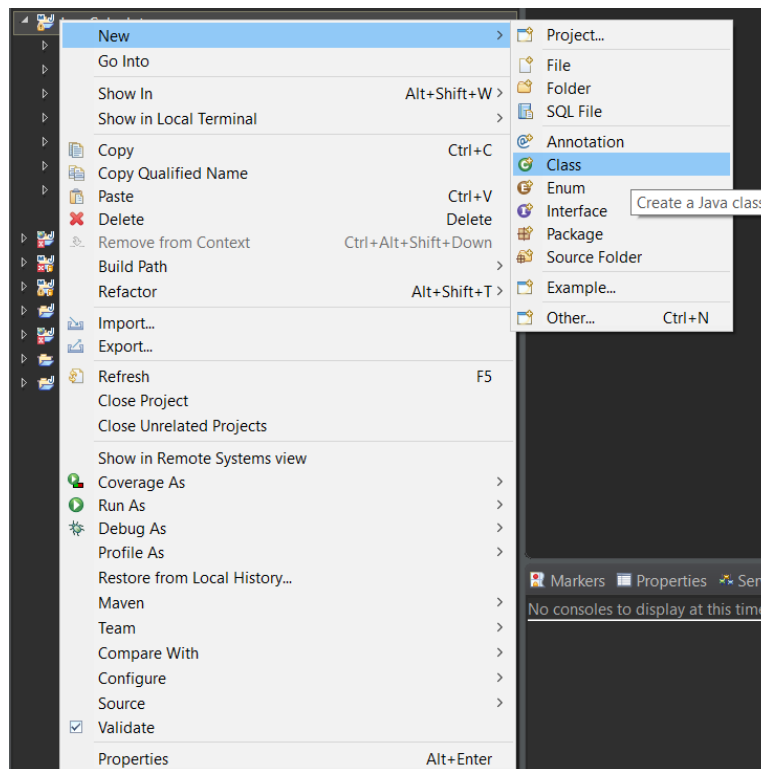
The image shows the 'New Maven Project' dialog box in an IDE. The title bar says 'New Maven Project'. The main heading is 'New Maven project' with a subtitle 'Select project name and location'. There is a Maven logo icon in the top right. The dialog has several options: 'Create a simple project (skip archetype selection)' is checked, 'Use default Workspace location' is checked, and 'Add project(s) to working set' is unchecked. Below these are input fields for 'Location' and 'Working set', each with a 'Browse...' button. An 'Advanced' section is collapsed. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a blue border.

Next you will be asked to provide the artifact ID and Group ID for the project. Provide the names and then click Finish. Here, I am building a simple four function calculator using Java language.

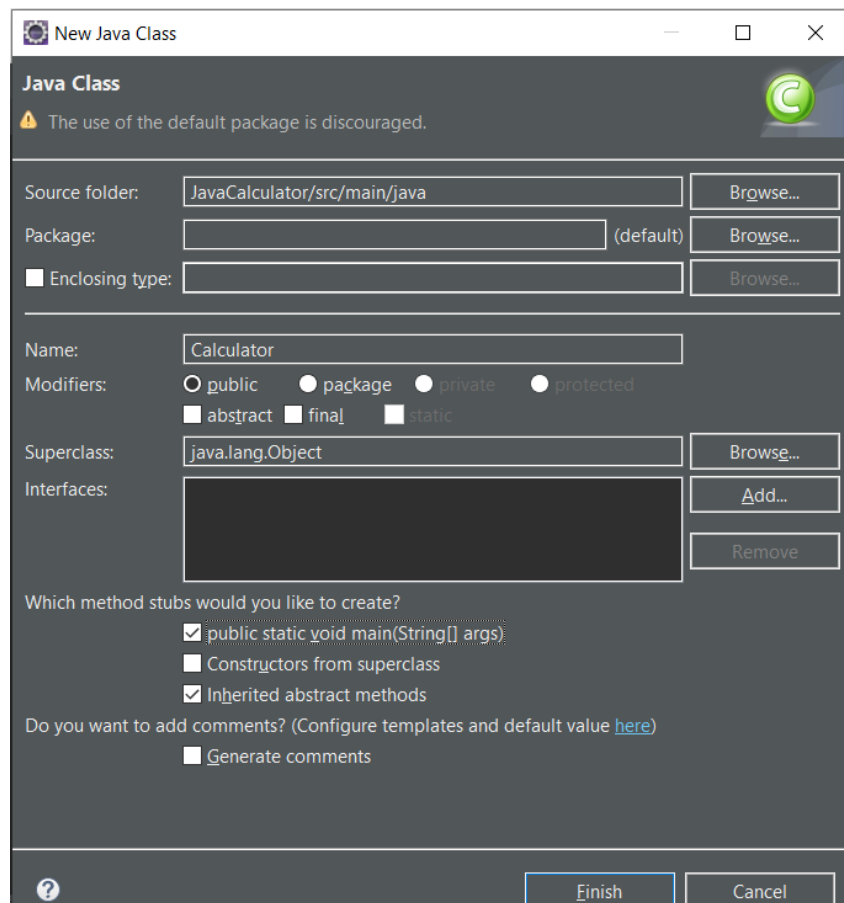


The image shows the 'New Maven Project' dialog box in the 'Configure project' step. The title bar says 'New Maven Project'. The main heading is 'New Maven project' with a subtitle 'Configure project'. There is a Maven logo icon in the top right. The 'Artifact' section contains fields for 'Group Id' (set to 'Calculator'), 'Artifact Id' (set to 'JavaCalculator'), 'Version' (set to '0.0.1-SNAPSHOT'), and 'Packaging' (set to 'jar'). There are also empty fields for 'Name' and 'Description'. The 'Parent Project' section has fields for 'Group Id', 'Artifact Id', and 'Version', along with 'Browse...' and 'Clear' buttons. An 'Advanced' section is collapsed. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Finish' button is highlighted with a blue border.

2. Add a new class to the project. To do so, right click on the project then click on New and then Class.



Provide the name of the class and then click on finish. Now add the code in the class.



3. After writing code file, the project is ready. Now create a repository on GitHub to store the project source code.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



Hridyanshu

Repository name *

/ Assignment2



Great repository names are short and memorable. Need inspiration? How about [bug-free-octo-meme?](#)

Description (optional)



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.](#)

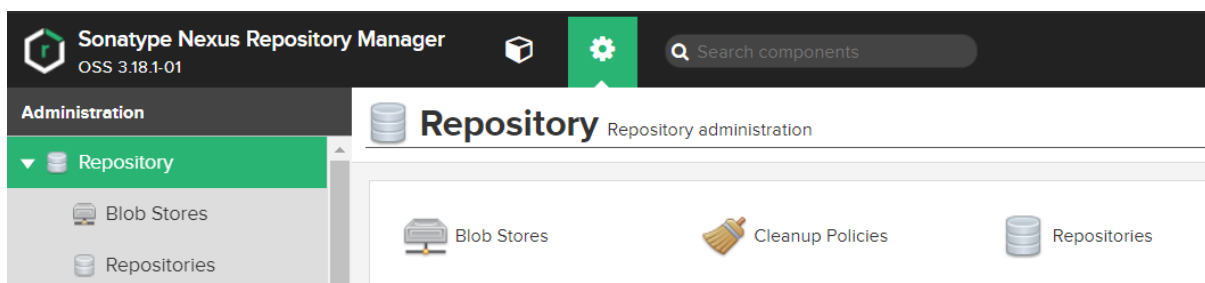
☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

4. Now start your nexus server by using the following command: `nexus.exe /run`

5. Create a new maven hosted repository on nexus.



Repositories / Select Recipe / Create Repository: maven2 (hosted)

Snapshot

Layout policy:

Validate that all paths are maven artifact or metadata paths

Strict

Storage

Blob store:

Blob store used to store asset contents

default

Strict Content Type Validation:

☒ Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

Hosted

Deployment policy:

Controls if deployments of and updates to artifacts are allowed

Allow redeploy

Cleanup Policy

Available cleanup policies:

Select a cleanup policy

None

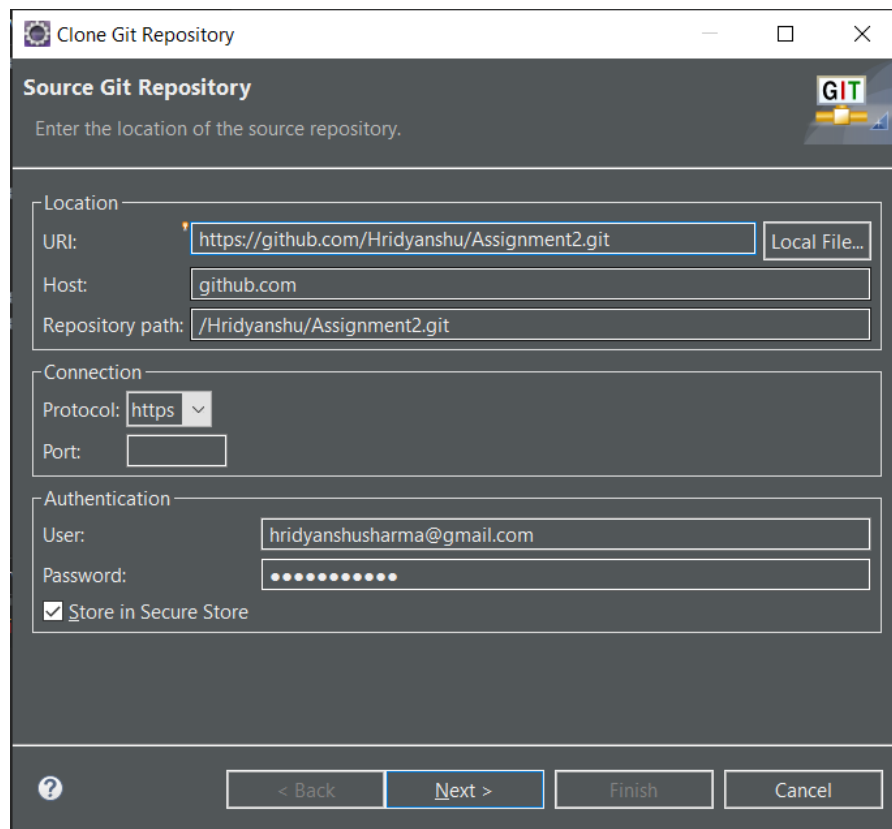
Create repository

Cancel

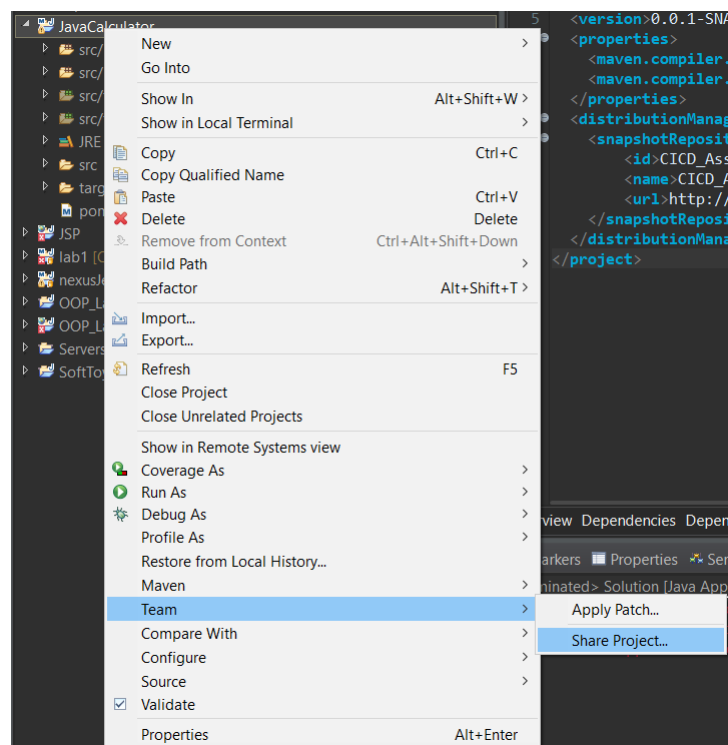
6. Add the distributionManagement tag in the pom.xml file of the project and provide the name, id and url of the Nexus repository there.

```
JavaCalculator/pom.xml x
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema"
2   <modelVersion>4.0.0</modelVersion>
3   <groupId>Calculator</groupId>
4   <artifactId>JavaCalculator</artifactId>
5   <version>0.0.1-SNAPSHOT</version>
6   <properties>
7     <maven.compiler.source>1.7</maven.compiler.source>
8     <maven.compiler.target>1.7</maven.compiler.target>
9   </properties>
10  <distributionManagement>
11    <snapshotRepository>
12      <id>CICD_Assignment2</id>
13      <name>CICD_Assignment2</name>
14      <url>http://localhost:8081/repository/CICD_Assignment2</url>
15    </snapshotRepository>
16  </distributionManagement>
17 </project>
```

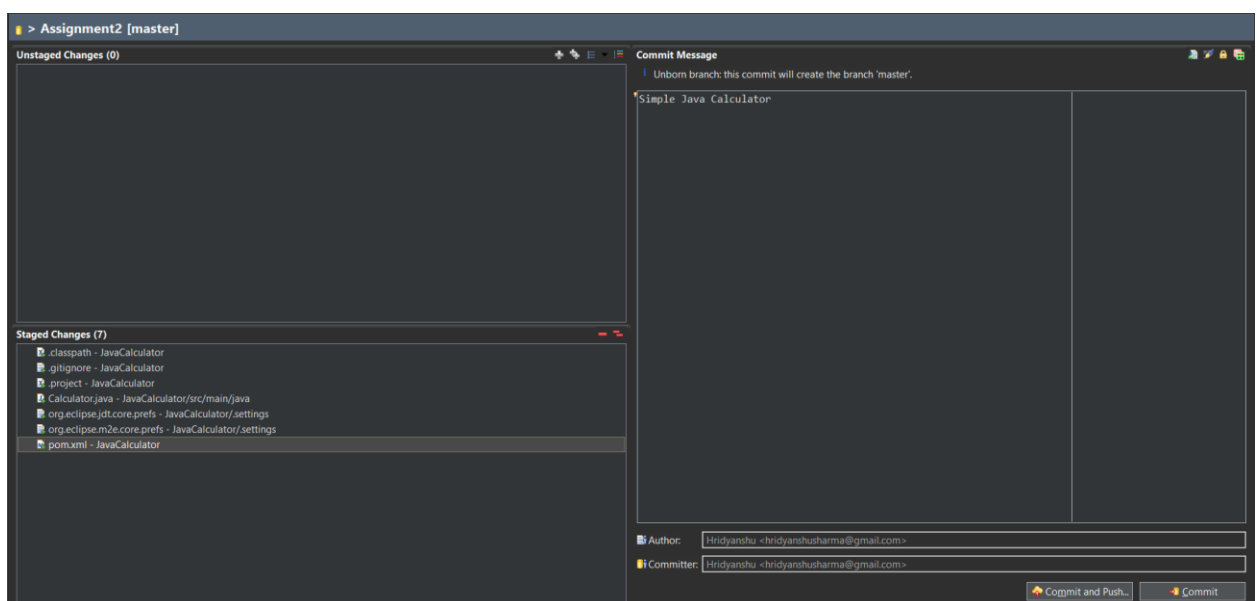
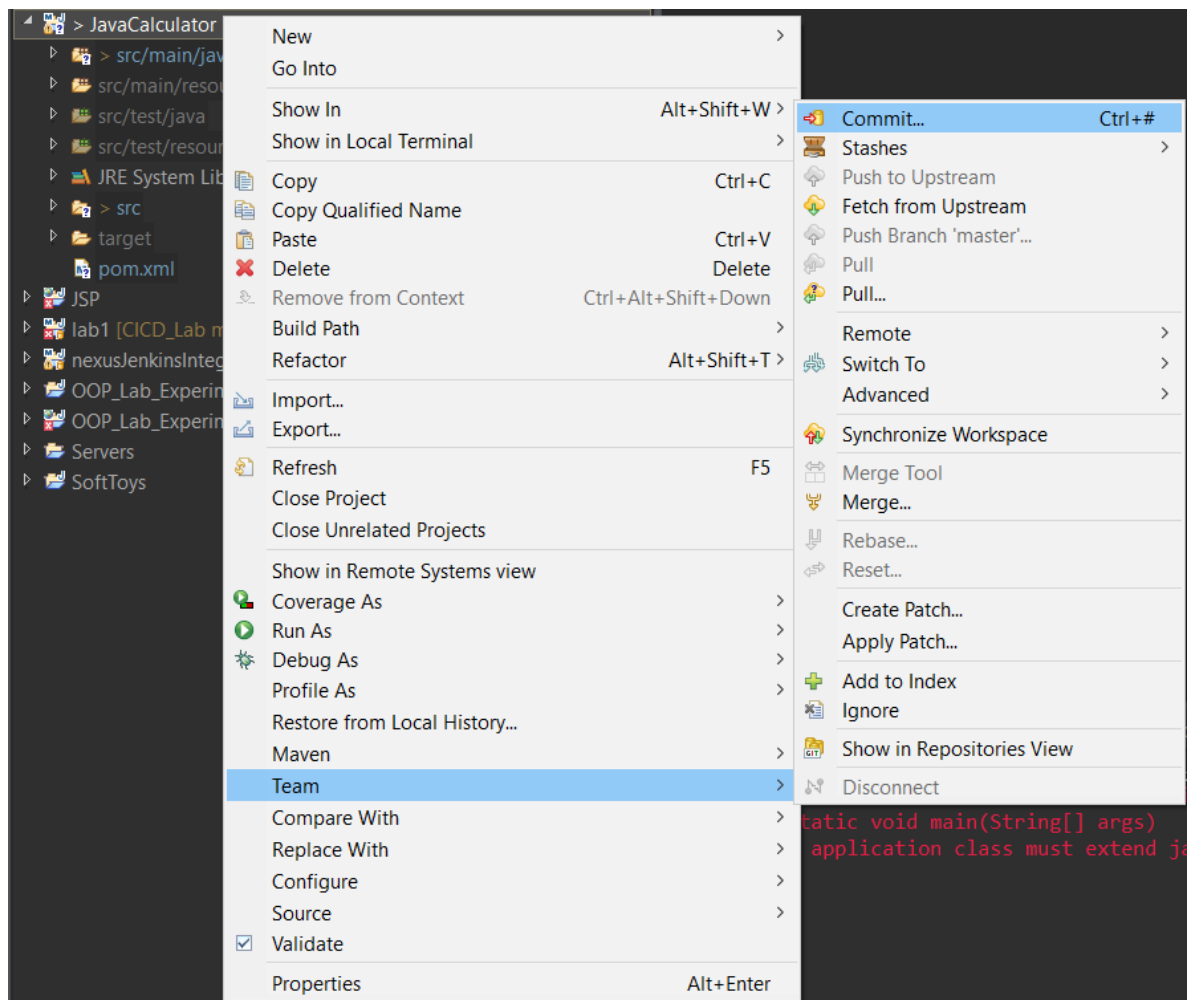
7. Clone the GitHub repository in the Eclipse IDE.



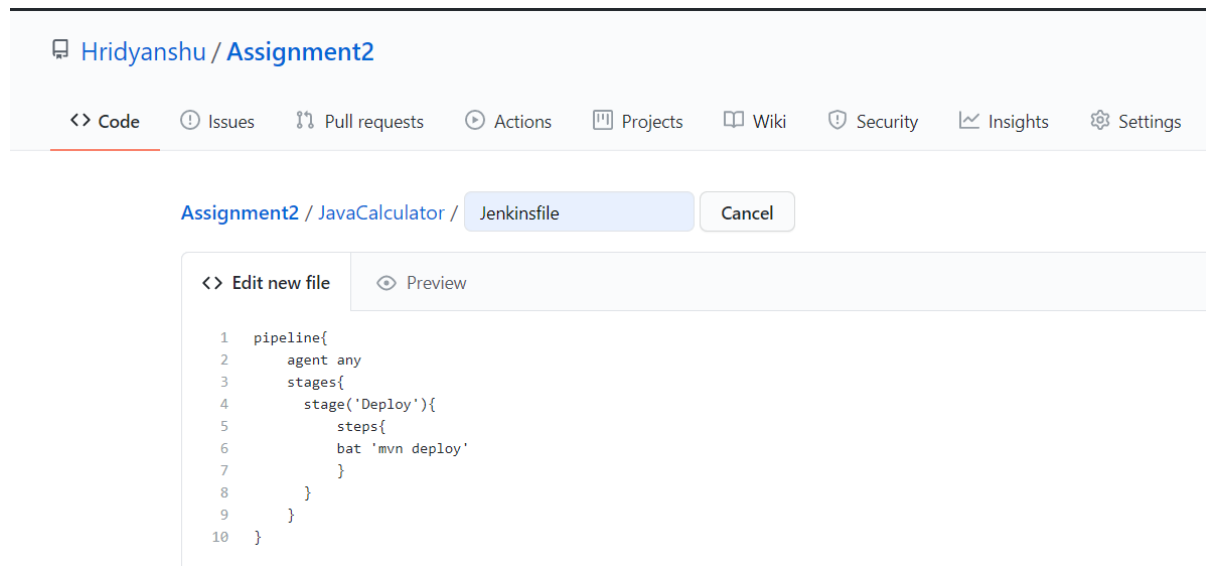
8. Link the project with the cloned GitHub repository.



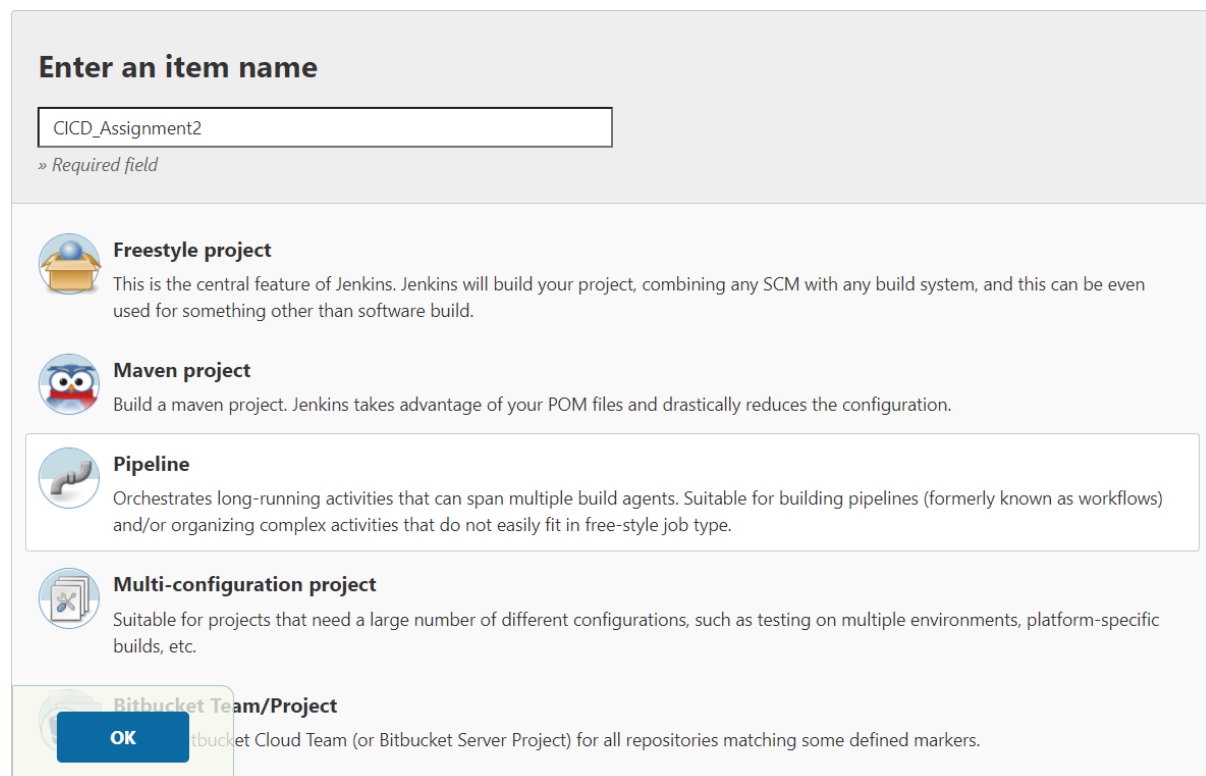
9. Push the project to GitHub repository.



10. Create a Jenkinsfile in the Github repository to with a deploy stage as follows:



11. Create a new pipeline project in Jenkins and link that project to the above made GitHub repository. Provide the path to the Jenkinsfile as well.



General
Build Triggers
Advanced Project Options
Pipeline

Definition
Pipeline script from SCM

SCM
Git

Repositories

Repository URL
https://github.com/Hridyanshu/Assign

Credentials
- none -
Add

Advanced...

Add Repository

Branches to build

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

Add Branch

Repository browser
(Auto)

Additional Behaviours
Add

Script Path
JavaCalculator/Jenkinsfile

Lightweight checkout
☒

Save
Apply

```

0.0.1-SNAPSHOT.pom
[INFO]
[INFO] --- maven-deploy-plugin:2.7:deploy (default-deploy) @ JavaCalculator ---
Downloading from CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/maven-metadata.xml
Uploading to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/JavaCalculator-0.0.1-20201119.182605-1.jar
Progress (1): 2.0 kB

Uploaded to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/JavaCalculator-0.0.1-20201119.182605-1.jar (2.0 kB at 60 B/s)
Uploading to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/JavaCalculator-0.0.1-20201119.182605-1.pom
Progress (1): 757 B

Uploaded to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/JavaCalculator-0.0.1-20201119.182605-1.pom (757 B at 87 B/s)
Downloading from CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/maven-metadata.xml
Uploading to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/maven-metadata.xml
Progress (1): 774 B

Uploaded to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/0.0.1-SNAPSHOT/maven-metadata.xml (774 B at 179 B/s)
Uploading to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/maven-metadata.xml
Progress (1): 284 B

Uploaded to CICD_Assignment2: http://localhost:8081/repository/CICD_Assignment2/Calculator/JavaCalculator/maven-metadata.xml (284 B at 110 B/s)
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:10 min
[INFO] Finished at: 2020-11-19T23:56:53+05:30
[INFO] -----
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

After successful build, the artifact is pushed to Nexus repository.

