

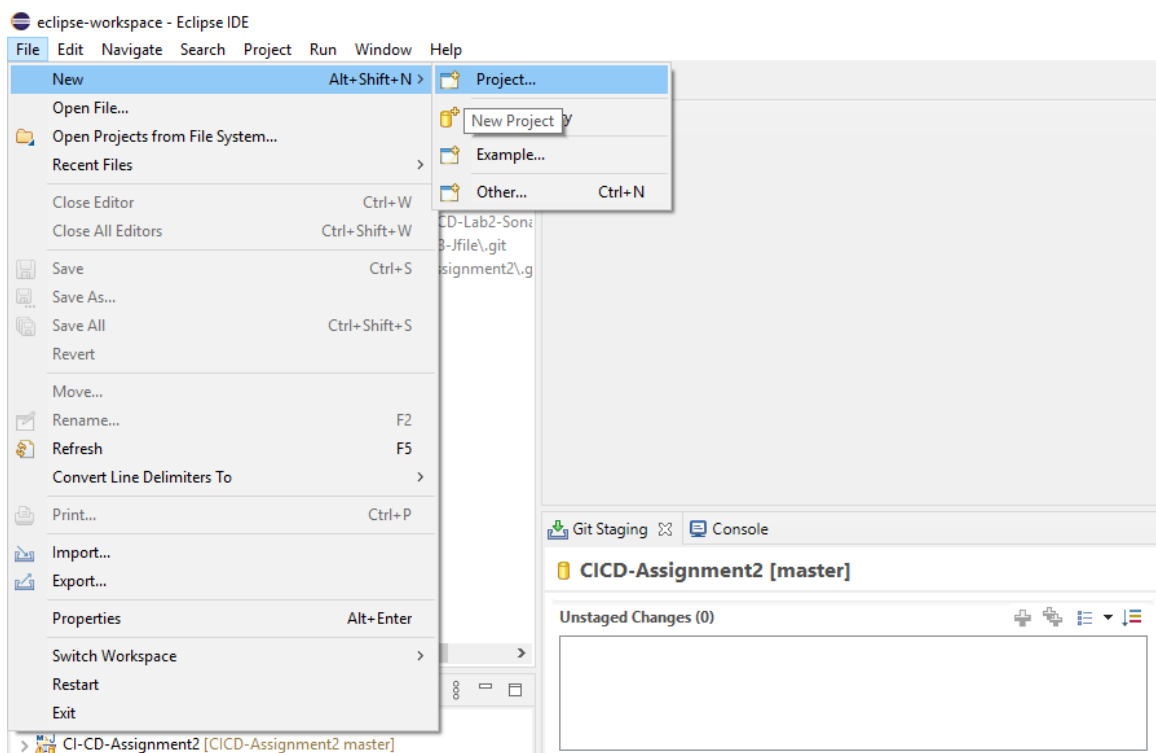
Name: Prashasti Gupta

Roll No: R171218077

SAP ID: 500068493

Assignment: 02

1. On the Eclipse IDE, create a new Maven Project.



2. Give a Group ID and a artifact ID to the project. Click on Finish.

New Maven Project

New Maven project
Configure project

Artifact

Group Id:

Artifact Id:

Version:

Packaging:

Name:

Description:

Parent Project

Group Id:

Artifact Id:

Version:

Advanced

3. Now, right click on the project, click on New and then select Class. After that, you will be prompted to give a name to the class. Finally, click on Finish.

New Java Class

Java Class

The use of the default package is discouraged.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

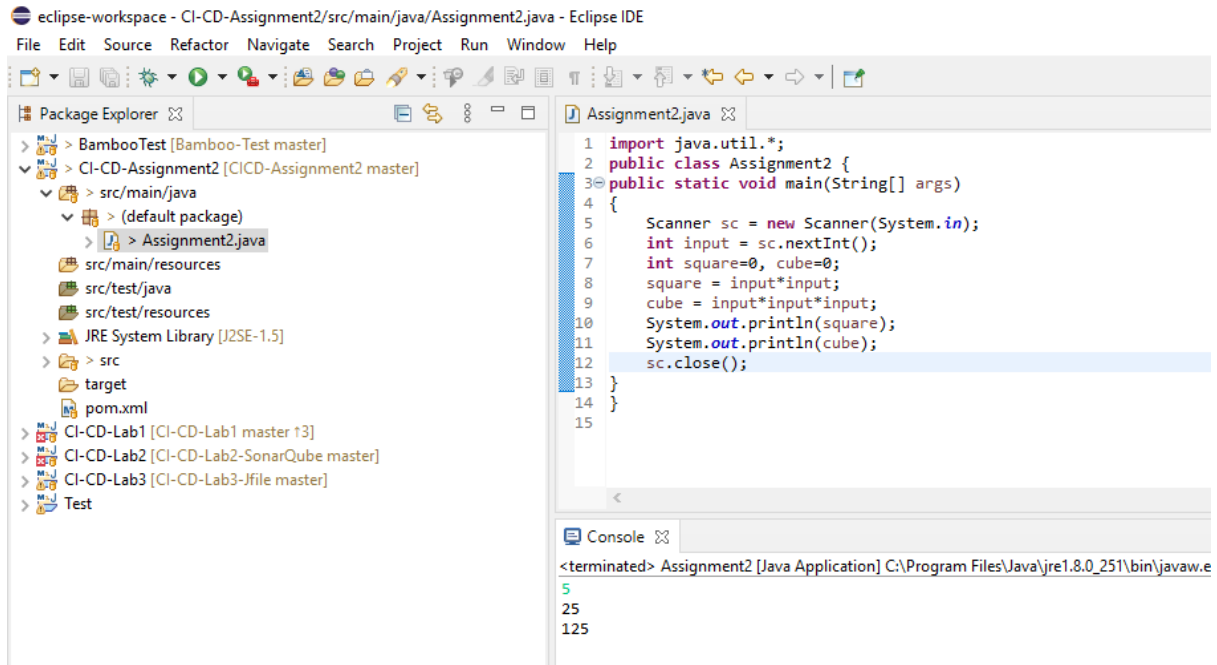
☐ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

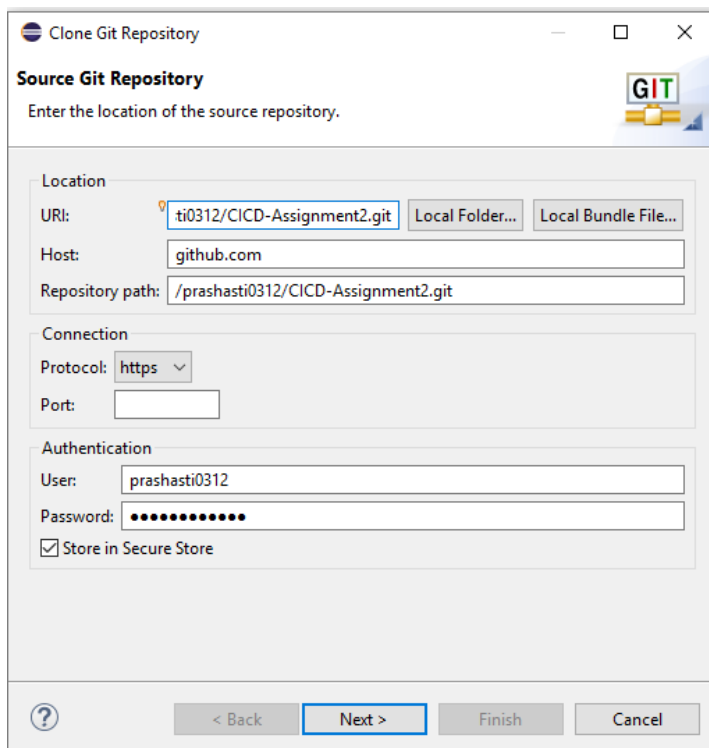
4. Add the Java code to the above created class.

Here, I have created a program in which user will give a number as an input. The square and cube of that number will be calculated and displayed on the screen.

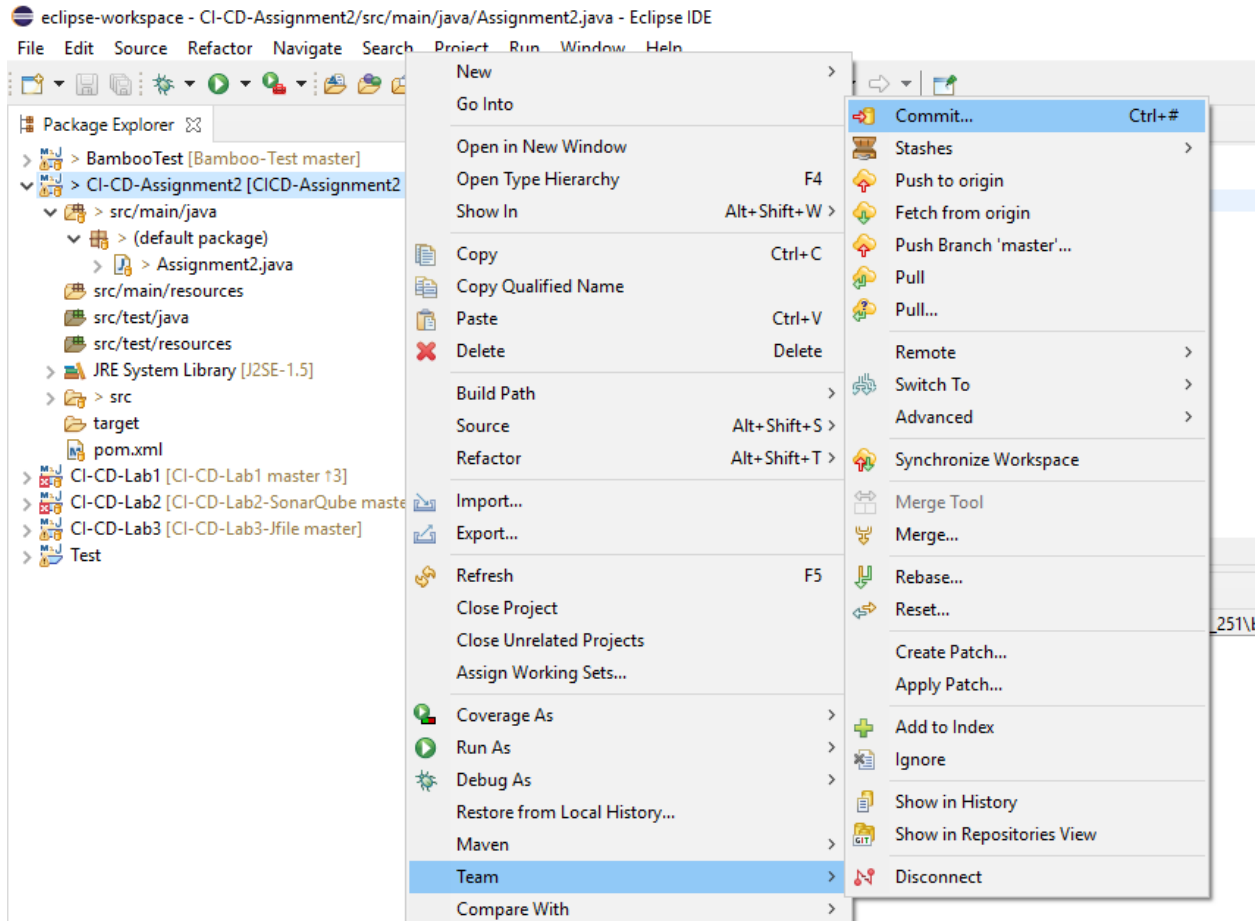


5. Create a new GitHub repository to which the maven project will be pushed.

6. Clone the GitHub repository in Eclipse IDE.



7. Now, link this maven project to GitHub repository. Move all the unstaged changes to staging area and finally click on Commit and Push.



Push Branch master

Push to branch in remote

Select a remote and the name the branch should have in the remote.

Source:

master
 d718762 Third Commit

Destination:

Remote:
origin: <https://github.com/prashasti0312/CICD-Assignment2.git>
New Remote...

Branch:
master

☒ Configure upstream for push and pull

When pulling:
Merge

☐ Force overwrite branch in remote if it exists and has diverged

Show [advanced push](#) dialog

?

< Back

Preview >

Push

Cancel

8. Now, create a Jenkinsfile in the GitHub repository.

master
[CICD-Assignment2](#) / [CI-CD-Assignment2](#) / Jenkinsfile

prashasti0312 Create Jenkinsfile

1 contributor

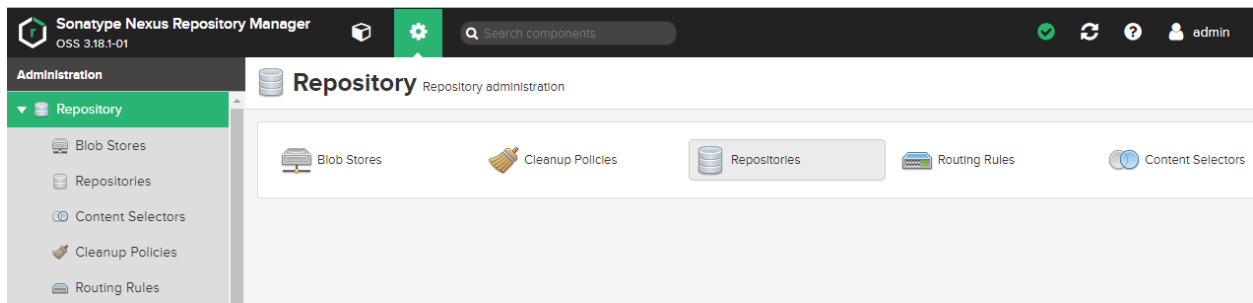
11 lines (10 sloc) | 108 Bytes

```

1 pipeline{
2   agent any
3   stages{
4     stage('Deploy'){
5       steps{
6         bat 'mvn deploy'
7       }
8     }
9   }
10 }
11

```

9. Open Command Prompt and start the nexus server by using the command - **nexus.exe /run**.
10. Create a maven hosted repository on Nexus.



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

Name: A unique identifier for this repository

Online: ☒ If checked, the repository accepts incoming requests

Maven 2

Version policy:
What type of artifacts does this repository store?

Layout policy:
Validate that all paths are maven artifact or metadata paths

Storage

Blob store:
Blob store used to store asset contents

Strict Content Type Validation:
☒ Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

11. Now, add the distributionManagement tag to the pom.xml file of the maven project. Push the changes to GitHub.

```
Assignment2.java  CI-CD-Assignment2/pom.xml
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
2   <modelVersion>4.0.0</modelVersion>
3   <groupId>CI-CD</groupId>
4   <artifactId>CI-CD-Assignment2</artifactId>
5   <version>0.0.1-SNAPSHOT</version>
6   <distributionManagement>
7     <snapshotRepository>
8       <id>Assignment2</id>
9       <name>Assignment2</name>
10      <url>http://localhost:8081/repository/Assignment2</url>
11    </snapshotRepository>
12  </distributionManagement>
13 </project>
```


12. Start the Jenkins service by using the command- **java -jar Jenkins.war.**

13. Now, create a new Pipeline in Jenkins.


Enter an item name

Assignment2


» 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

OK

14. Now, scroll down to the Pipeline section and then provide the URL of the GitHub repository as well as the path to the Jenkinsfile.

General Build Triggers Advanced Project Options **Pipeline**

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL

Credentials

Branches to build

Branch Specifier (blank for 'any')

Repository browser (Auto)

Additional Behaviours

15. Click on Build Now.

```
[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
```

16. After the build is successful, the artifact will be pushed to Nexus.



Browse



Welcome



Search



Browse



Upload



Browse / CICD_Assignment2

[HTML View](#)



Calculator



JavaCalculator



0.0.1-SNAPSHOT



maven-metadata.xml



maven-metadata.xml.md5



maven-metadata.xml.sha1