

Integration of Jenkins with GitHub

~By Neha Singh

GitHub is a multinational corporation that provides hosting for software development and version control using Git.

Prerequisites

- Jenkins running

In this module, we will see how to integrate Jenkins with GitHub.

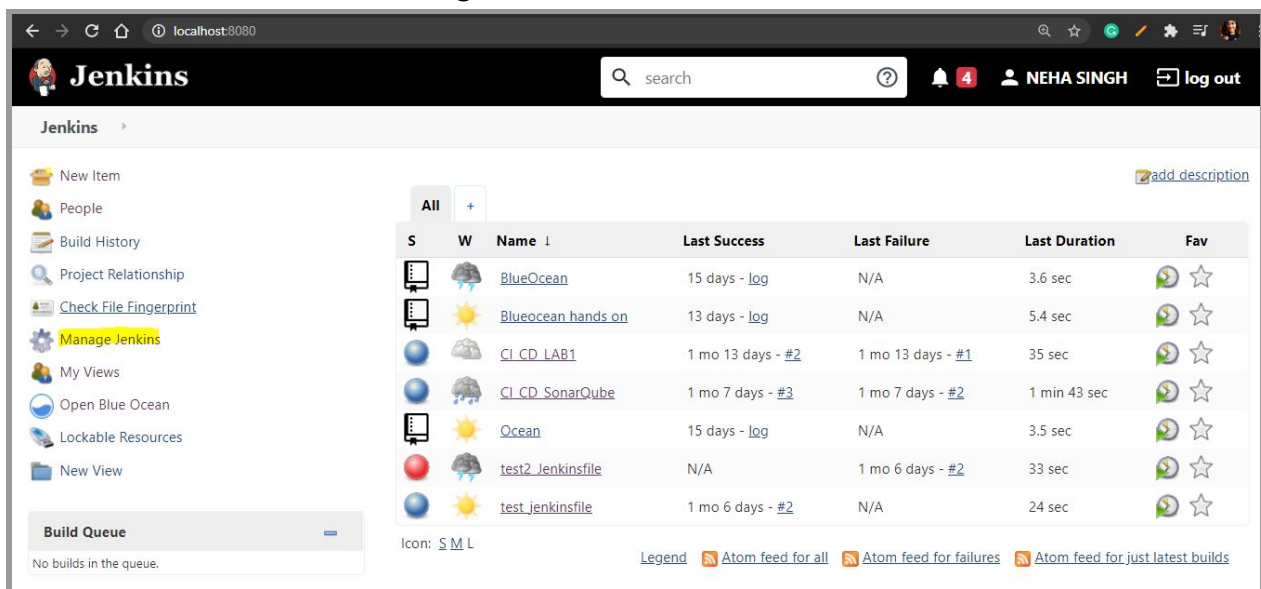
Make sure your Jenkins is up and running.

```
C:\Windows\System32\cmd.exe - java -jar jenkins.war
Microsoft Windows [Version 10.0.18362.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Neha\Downloads>java -jar jenkins.war
Running from: C:\Users\Neha\Downloads\jenkins.war
webroot: $user.home/.jenkins
2020-10-08 14:11:54.804+0000 [id=1] INFO org.eclipse.
2020-10-08 14:11:55.382+0000 [id=1] INFO winstone.Log
2020-10-08 14:11:55.580+0000 [id=1] WARNING o.e.j.s.hand
2020-10-08 14:11:55.847+0000 [id=1] INFO org.eclipse
```

Open jenkins in web browser, I will be using Chrome
“Localhost:8080”

On left-hand side, move to manage Jenkins.



Manage Plugins



The screenshot shows the Jenkins 'System Configuration' page. The browser address bar indicates 'localhost:8080/manage'. The page title is 'Jenkins'. Under the 'System Configuration' heading, there are five options:

- Configure System**: Configure global settings and paths. (Gear icon)
- Global Tool Configuration**: Configure tools, their locations and automatic installers. (Wrench icon)
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. (Puzzle piece icon). A red bell icon indicates 'There are updates available'.
- Manage Nodes and Clouds**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on. (Computer icon)
- Install as Windows Service**: Installs Jenkins as a Windows service to this system, so that Jenkins starts automatically when the machine boots. (CD/DVD icon)

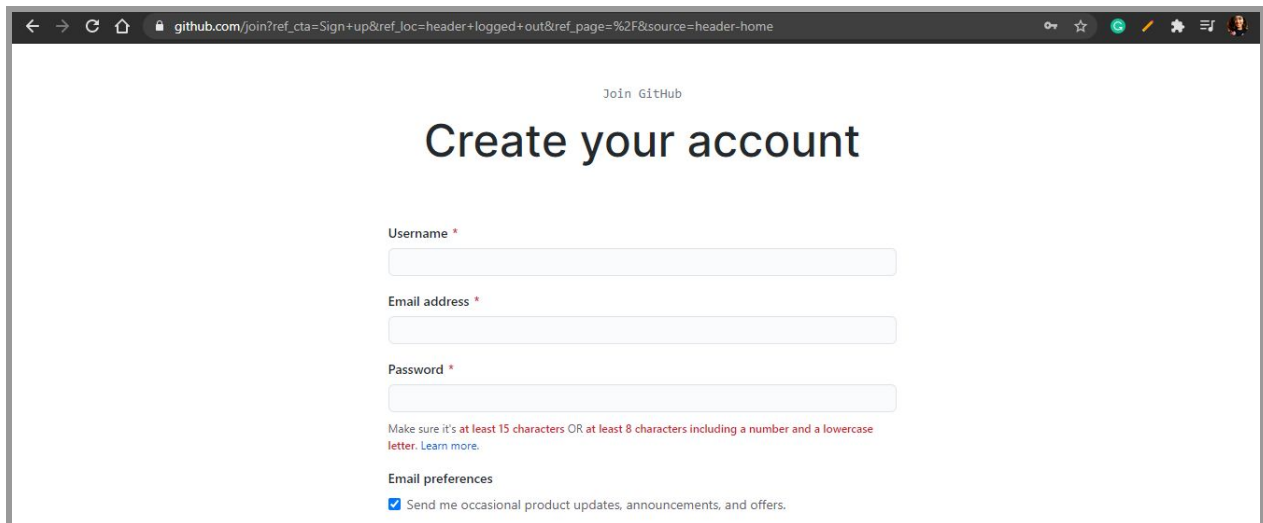
Now, you can install **Github Plugins**



The screenshot shows the Jenkins 'Plugin Manager' page. The browser address bar indicates 'localhost:8080/manage'. The page title is 'Jenkins > Plugin Manager'. A table lists available plugins:

Plugin Name	Version	Description
<input checked="" type="checkbox"/> Git	4.3.0	This plugin integrates Git with Jenkins.

If you do not have GitHub Account, move to <https://github.com/>
Enter credentials



The screenshot shows the GitHub 'Create your account' page. The browser address bar indicates 'github.com/join?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=header-home'. The page title is 'Join GitHub'. The main heading is 'Create your account'.

Form fields:

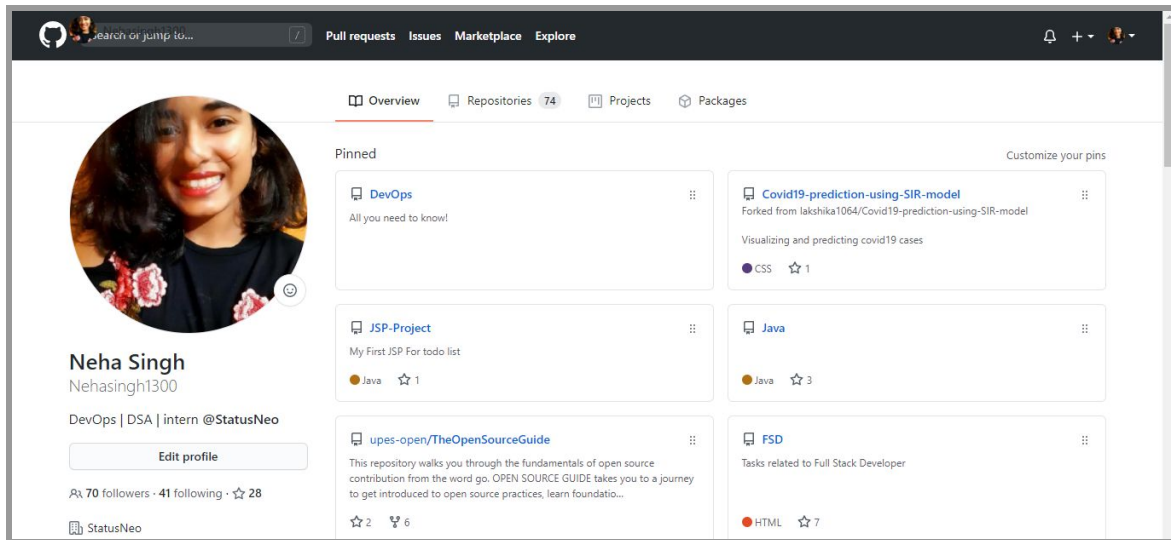
- Username *
- Email address *
- Password *

Make sure it's **at least 15 characters** OR **at least 8 characters including a number and a lowercase letter**. [Learn more](#).

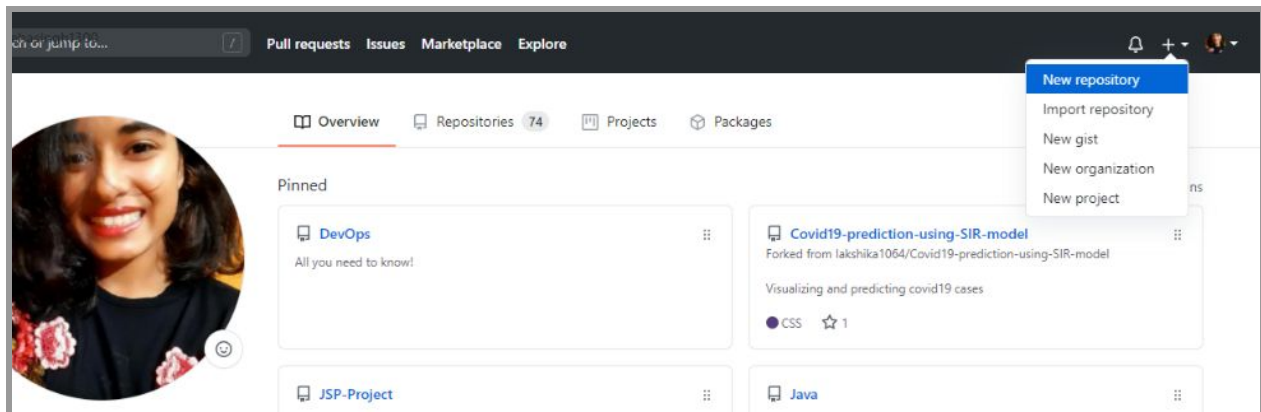
Email preferences

☒ Send me occasional product updates, announcements, and offers.

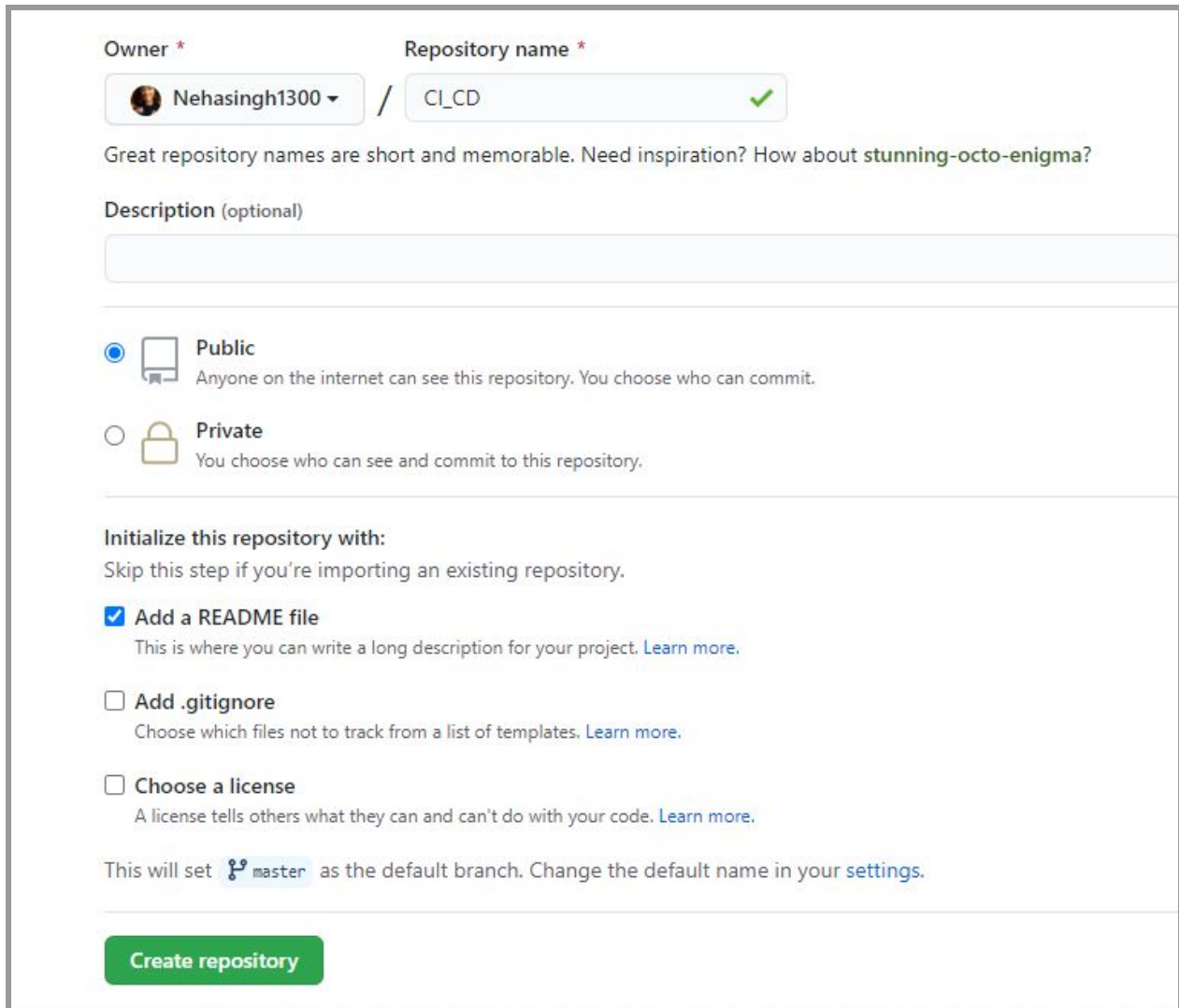
You will see account like this but without repositories.



Now, you can move to the upper right-hand side and click on + sign to make new repo.



You can enter any name for your repository, and description if you want. Then, go and click **Create Repository**



The screenshot shows the GitHub 'Create repository' form. At the top, there are two input fields: 'Owner' with a dropdown menu showing 'Nehasingh1300' and 'Repository name' with a text input 'CI_CD' and a green checkmark. Below these is a hint: 'Great repository names are short and memorable. Need inspiration? How about [stunning-octo-enigma?](#)'. There is a 'Description (optional)' text area. Under 'Visibility', the 'Public' option is selected with a radio button and a lock icon, with the text 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is unselected with a radio button and a lock icon, with the text 'You choose who can see and commit to this repository.' Under 'Initialize this repository with:', there is a checkbox for 'Add a README file' which is checked, with the text 'This is where you can write a long description for your project. [Learn more.](#)'. There are also unselected checkboxes for 'Add .gitignore' (with text 'Choose which files not to track from a [list of templates.](#) [Learn more.](#)') and 'Choose a license' (with text 'A license tells others what they can and can't do with your code. [Learn more.](#)'). At the bottom, it says 'This will set [master](#) as the default branch. Change the default name in your [settings.](#)'. A green 'Create repository' button is at the bottom.

You should have a pom.xml in the main folder and then add this code inside that file.

```
<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">
  <modelVersion>4.0.0</modelVersion>
  <groupId>test.jenkinsfile</groupId>
  <artifactId>test.jenkinsfile</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <properties>
    <maven.compiler.source>1.6</maven.compiler.source>
    <maven.compiler.target>1.6</maven.compiler.target>
  </properties>
```

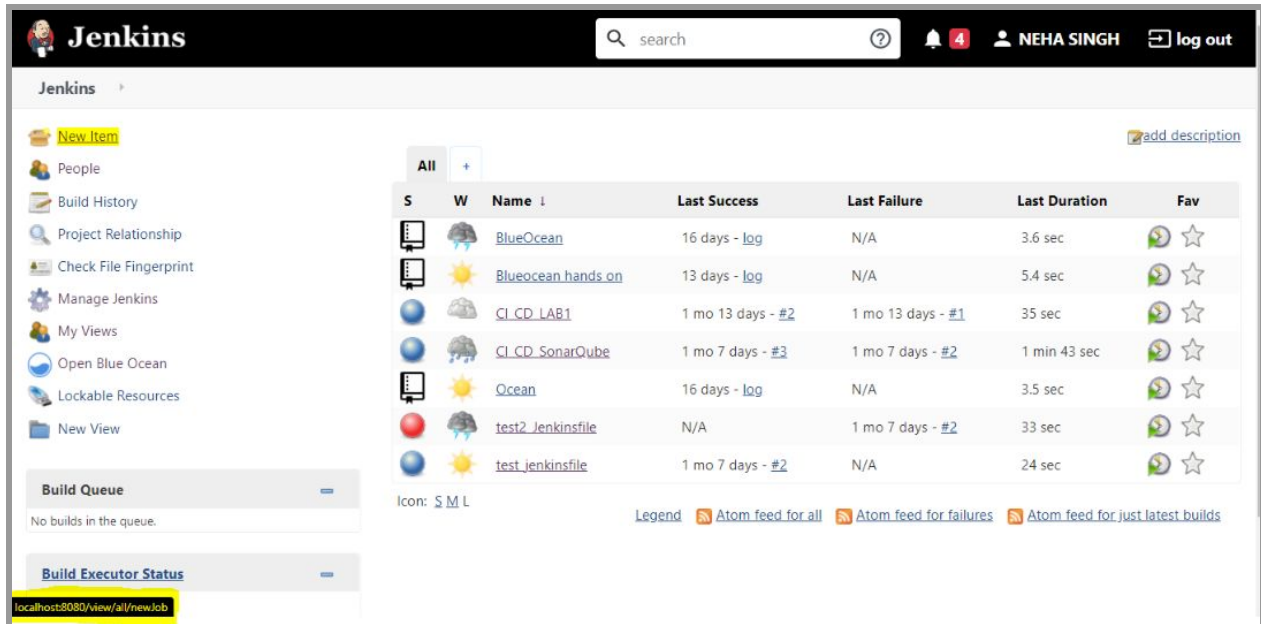
```

<distributionManagement>
<snapshotRepository>
  <id>cicd</id>
  <name>cicd</name>
  <url>http://localhost:8081/repository/cicd/</url>
</snapshotRepository>
</distributionManagement>
</project>

```

Enter your code in GitHub. You can see all the changes in your repository.

Move to new items



The screenshot shows the Jenkins dashboard interface. The top navigation bar includes the Jenkins logo, a search bar, and user information for NEHA SINGH. The left sidebar contains various navigation links such as 'New Item', 'People', 'Build History', and 'Project Relationship'. The main content area displays a table of jobs with columns for status, name, last success, last failure, last duration, and favorite status. The jobs listed are BlueOcean, Blueocean.hands.on, CI_CD_LAB1, CI_CD_SonarQube, Ocean, test2_Jenkinsfile, and test_jenkinsfile. The bottom of the dashboard shows the 'Build Queue' and 'Build Executor Status' sections.

S	W	Name	Last Success	Last Failure	Last Duration	Fav
		BlueOcean	16 days - log	N/A	3.6 sec	
		Blueocean.hands.on	13 days - log	N/A	5.4 sec	
		CI_CD_LAB1	1 mo 13 days - #2	1 mo 13 days - #1	35 sec	
		CI_CD_SonarQube	1 mo 7 days - #3	1 mo 7 days - #2	1 min 43 sec	
		Ocean	16 days - log	N/A	3.5 sec	
		test2_Jenkinsfile	N/A	1 mo 7 days - #2	33 sec	
		test_jenkinsfile	1 mo 7 days - #2	N/A	24 sec	

Icon: S M L

Legend Atom feed for all Atom feed for failures Atom feed for just latest builds

Build Queue: No builds in the queue.

Build Executor Status: [localhost:8080/view/all/newJob](#)

Select Maven Project and enter the name of the project.



Jenkins

CI_CD_LAB1

General

Source Code Management

Build Triggers

Build Environment

Pre Steps

Build

Post Steps

Build Settings

Post-build Actions

Description

[Plain text] [Preview](#)

Jira site

☐ Discard old builds☐ GitHub project☐ This build requires lockable resources☐ This project is parameterized☐ Throttle builds☐ Disable this project☐ Execute concurrent builds if necessary

Advanced...

Source Code Management

☐ None☒ Git

Repositories

Repository URL 

Credentials

Advanced...

Save

Apply

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/master



Jenkins

CI_CD_LAB1

Repository browser

(Auto)



Additional Behaviours

Add ▾

☐ Mercurial☐ Subversion

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☐ GitHub hook trigger for GITScm polling☐ Poll SCM

Build Environment

☐ Delete workspace before build starts☐ Use secret text(s) or file(s)☐ Abort the build if it's stuck☐ Add timestamps to the Console Output☐ Generate Release Notes☐ Inspect build log for published Gradle build scans☐ With Ant

Pre Steps

Add pre-build

Save

Apply

Build

Jenkins

CI_CD_LAB1

Goals and options

[Advanced...](#)

Post Steps

☐ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☒ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

[Add post-build step ▾](#)

Build Settings

☐ E-mail Notification

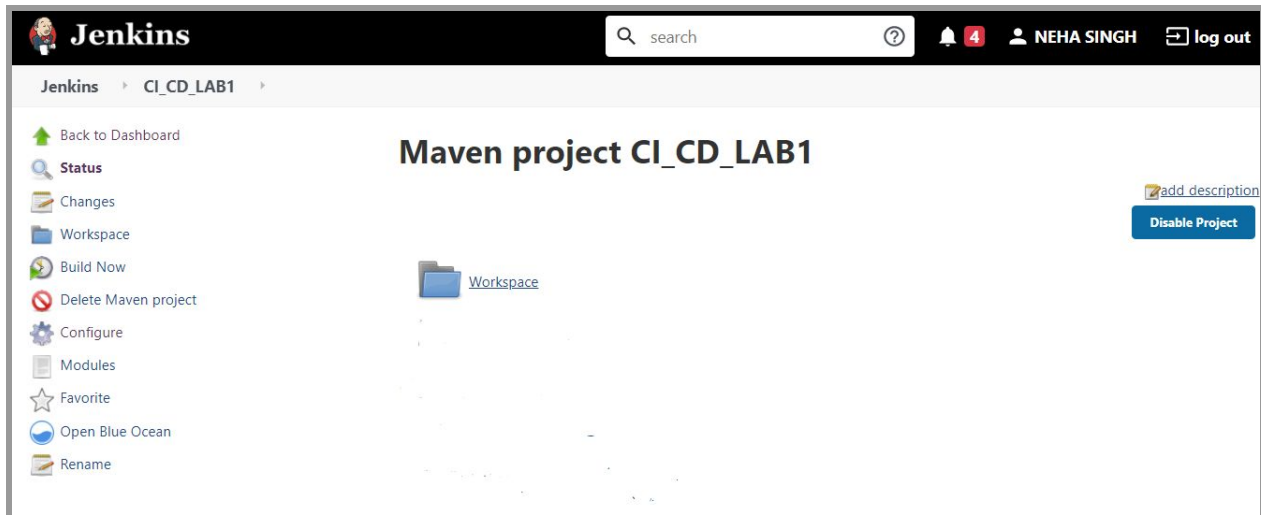
Post-build Actions

[Add post-build action ▾](#)

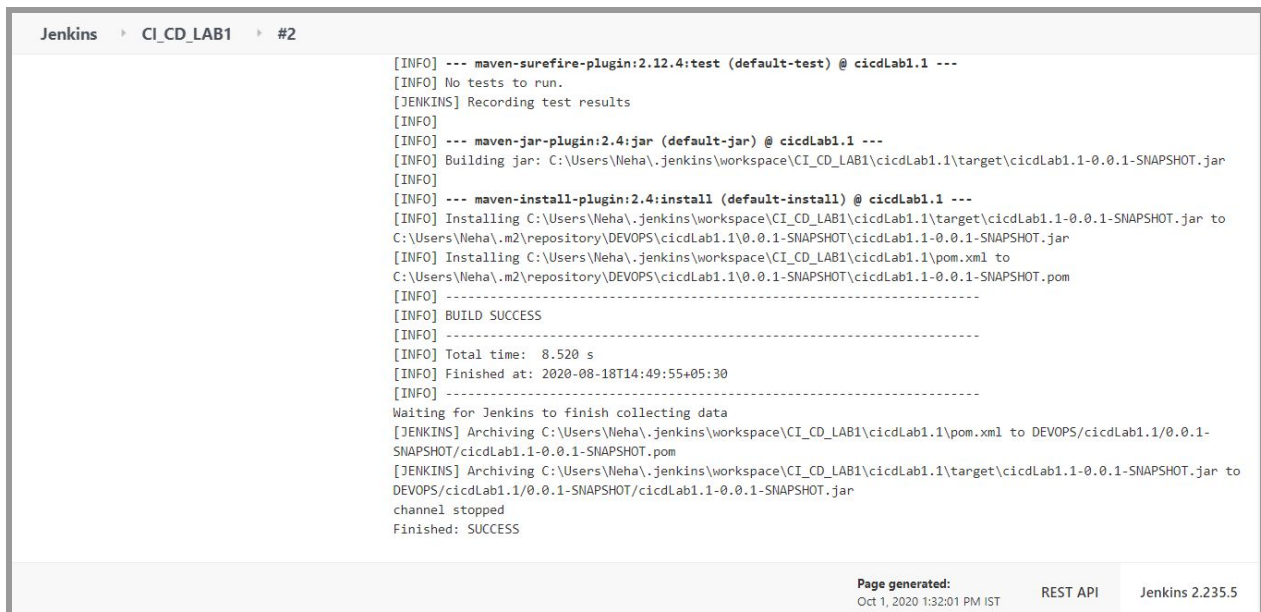
Jenkins 2.235.5

[Save](#)[Apply](#)

Now, you can see your project. Click on Build now



You will be able to see Build Success if your project has no error, else go to the error message and make suitable changes.



You can see the build artifact



The image shows a Jenkins build page for the project 'cicdLab1.1'. The breadcrumb navigation at the top reads 'Jenkins > CI_CD_LAB1 > #2 > cicdLab1.1'. On the left sidebar, there are links for 'Back to Project', 'Status', 'Changes', 'Console Output', 'Edit Build Information', 'Delete build 'cicdLab1.1'', 'Executed Mojos', 'See Fingerprints', 'Redeploy Artifacts', and 'Open Blue Ocean'. The main content area has a title 'Build cicdLab1.1 (Aug 18, 2020 2:49:2...)' with a blue sphere icon. Below the title is an 'add description' link. A 'Build Artifacts' section shows two files: 'cicdLab1.1-0.0.1-SNAPSHOT.jar' (1.85 KB) and 'cicdLab1.1-0.0.1-SNAPSHOT.pom' (359 B), each with a 'view' link. Below the artifacts is a 'No changes.' message with a notepad icon.

Jenkins > CI_CD_LAB1 > #2 > cicdLab1.1

Back to Project

Status

Changes

Console Output

Edit Build Information

Delete build 'cicdLab1.1'

Executed Mojos

See Fingerprints

Redeploy Artifacts

Open Blue Ocean

Build cicdLab1.1 (Aug 18, 2020 2:49:2...

[add description](#)

Build Artifacts

 cicdLab1.1-0.0.1-SNAPSHOT.jar	1.85 KB  view
 cicdLab1.1-0.0.1-SNAPSHOT.pom	359 B  view

 No changes.

That's all for this module.