

Jenkins Automation with Gitlab

1.Update the pakages

```
apt-get update
```

2.Install Required Java

Jenkins is a Java-based application, so you need to install Java on your system. You can install OpenJDK using the following command:

```
apt-get install openjdk-11-jdk -y
```

Verify the installation by checking the Java version

```
java -version
```

3.Add Jenkins Repository and Key

```
https://www.jenkins.io/doc/book/installing/linux/#debianubuntu
```

Go to jenkins website in browser, --> "Linux Section" --> "Debian/Ubuntu" --> " Copy the Long Term Support release" --> Do that steps

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins -y
```

4.Start the Jenkins service and enable it to start on boot

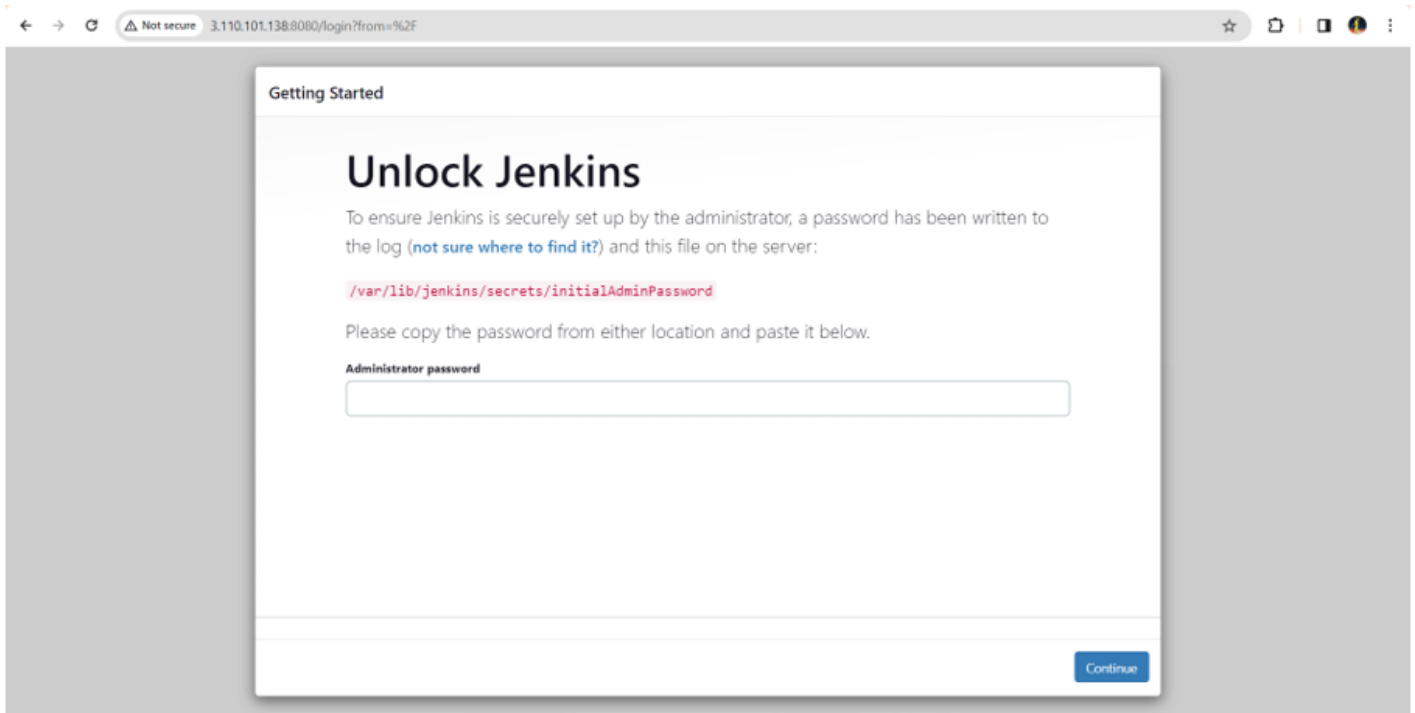
```
sudo systemctl start jenkins
```

```
sudo systemctl enable jenkins
```

5. Access Jenkins Web Interface

By default, Jenkins runs on port 8080. Open your web browser and navigate to

```
http://your_server_ip_or_domain:8080
```



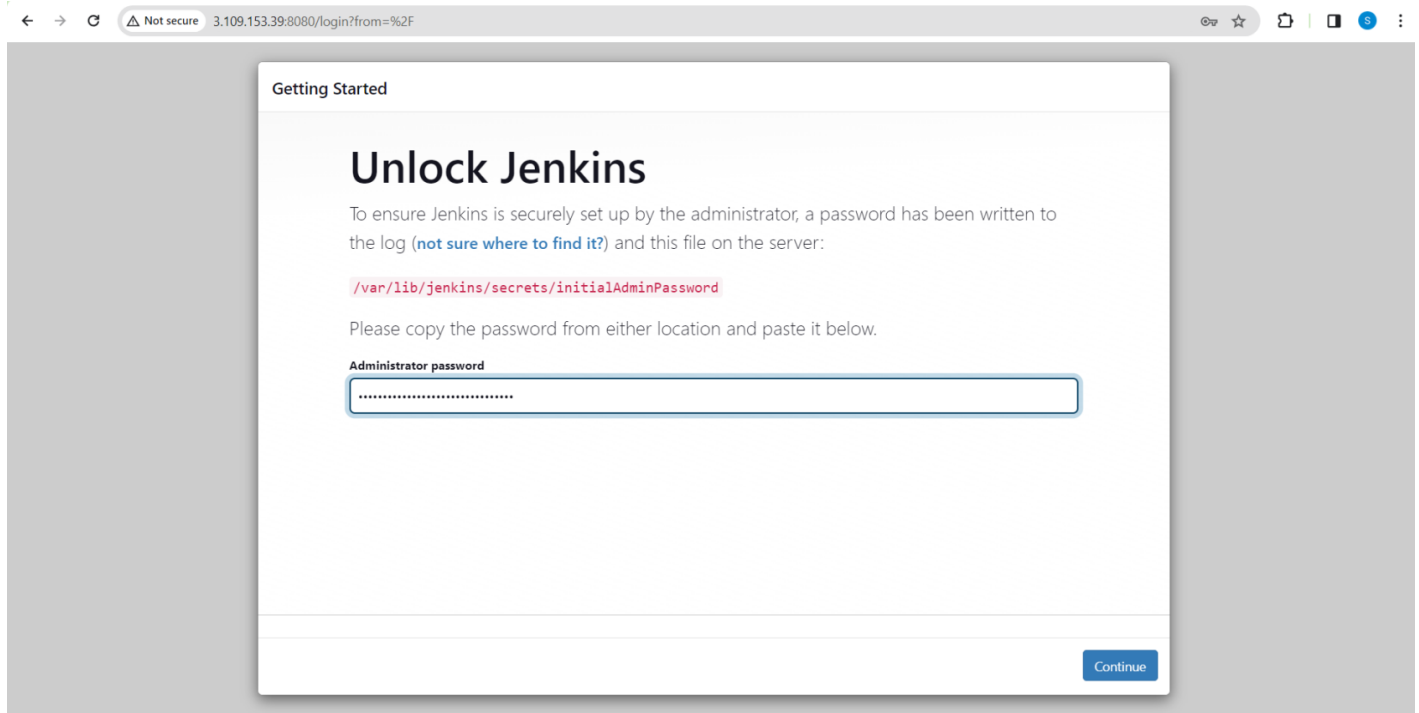
6. You will be prompted to unlock Jenkins.

Retrieve the initial admin password from the following location:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

```
root@ip-172-31-1-36:/home/ubuntu# cat /var/lib/jenkins/secrets/initialAdminPassword
8489304cef0a4e61ab39a7eccae1836f
root@ip-172-31-1-36:/home/ubuntu# |
```

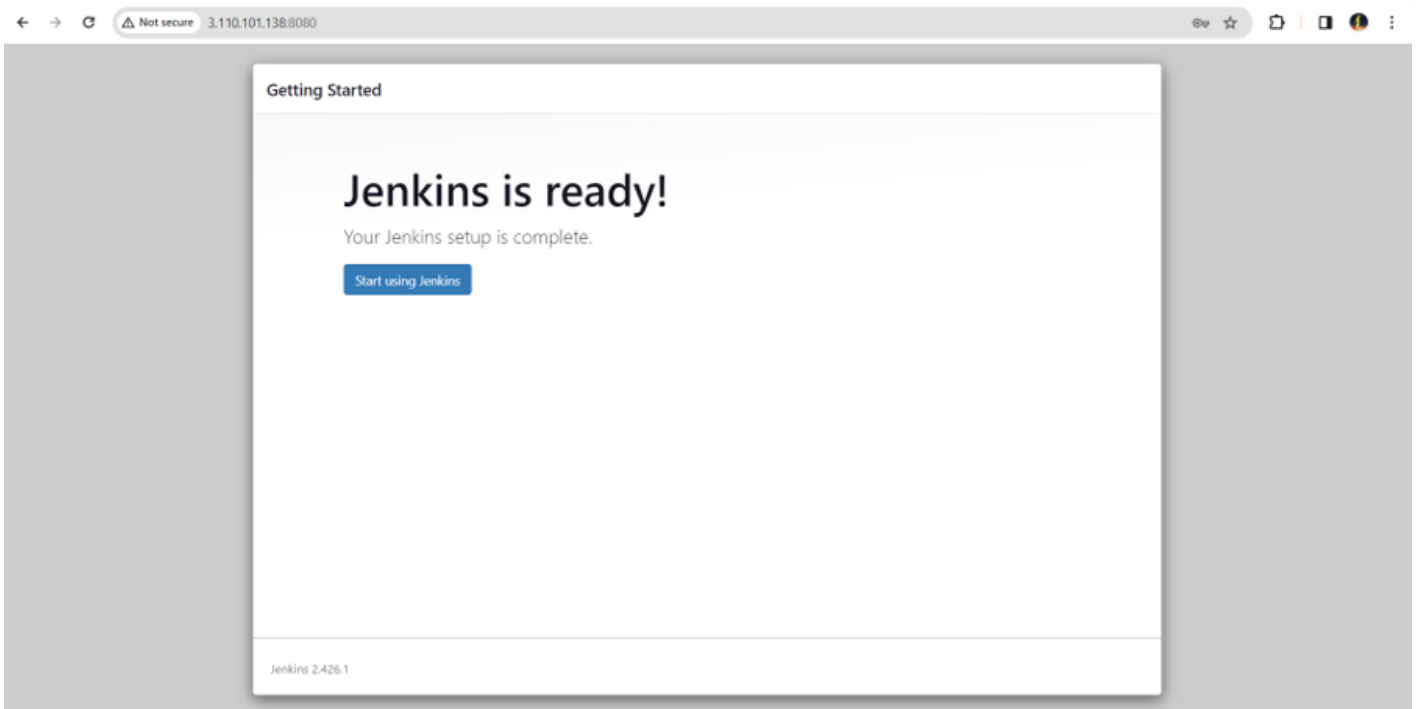
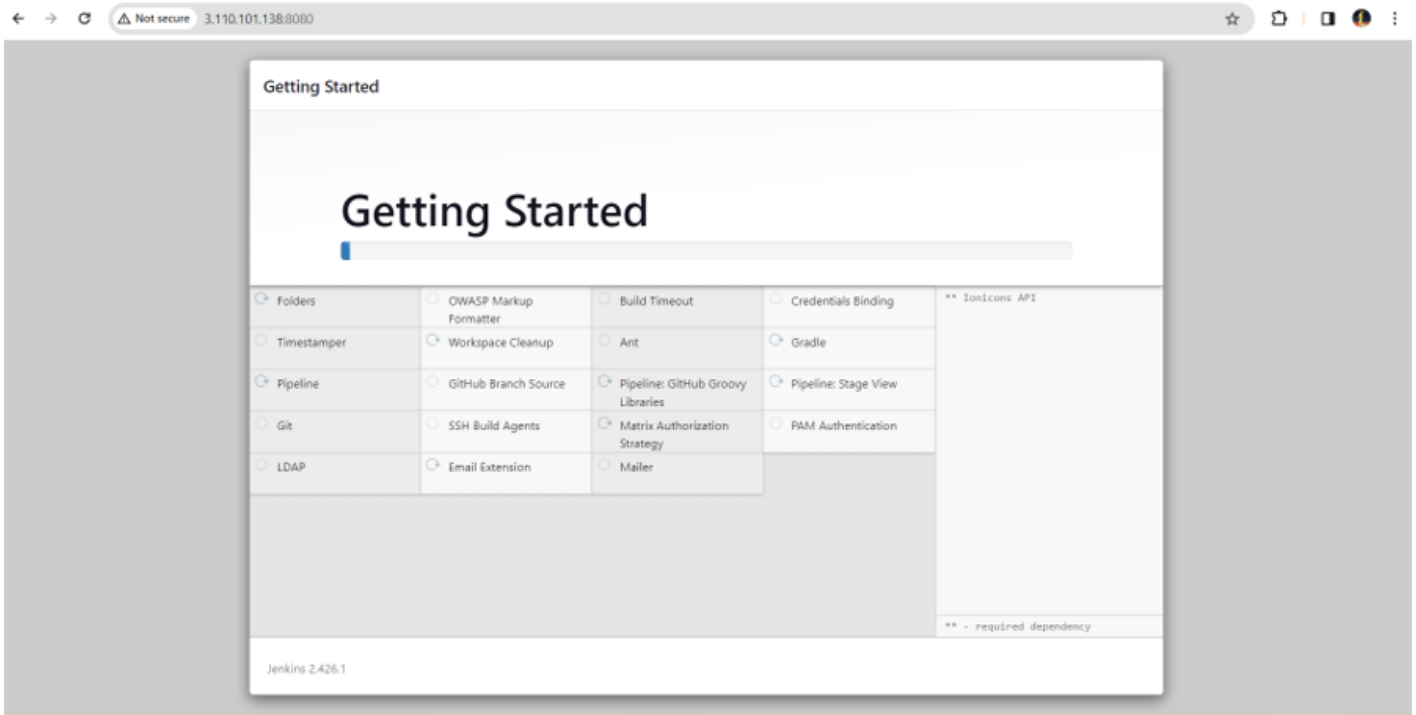
Copy and paste this password into the Jenkins web interface to unlock it.



7. Install Plugins

After unlocking Jenkins, you can choose to install recommended plugins or select specific plugins based on your requirements.

- Create an admin user and provide the necessary details.
- Specify the Jenkins URL and click "Save and Finish."
- After completing the setup, you can start using Jenkins.

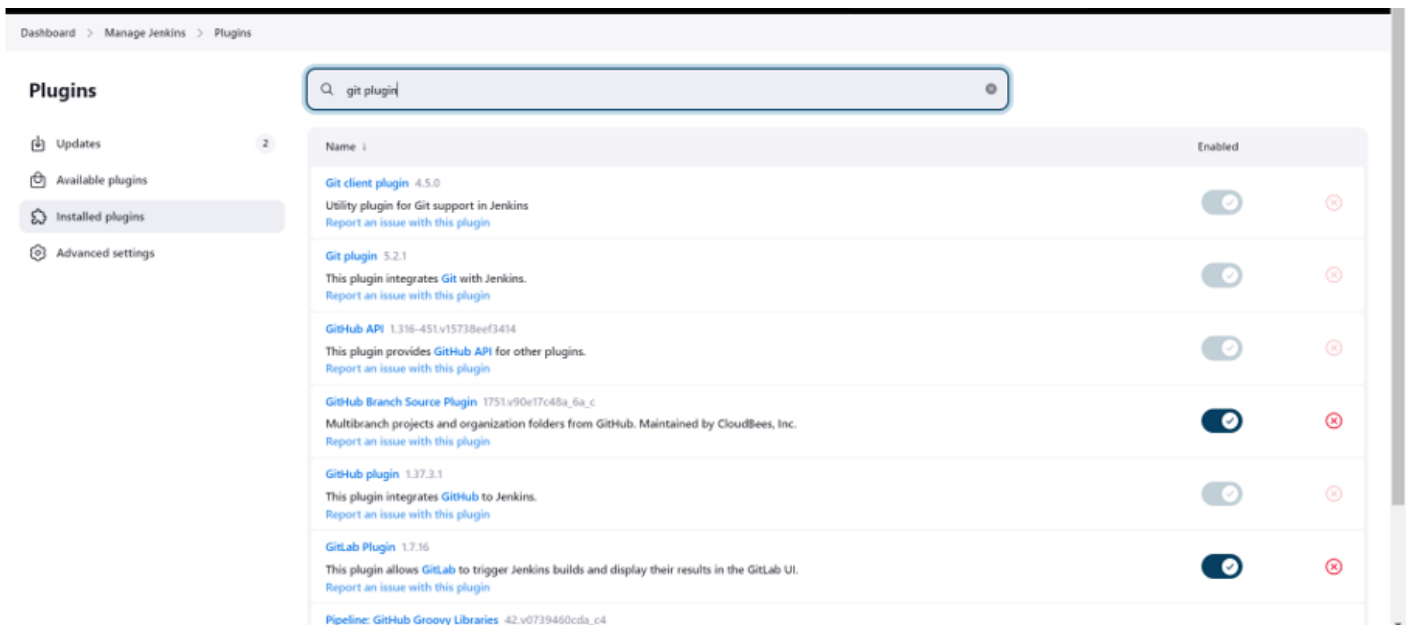


Configure the Jenkins server

Install and configure the Jenkins plugin to authorize the connection to GitLab.

1. On the Jenkins server, select "**Manage Jenkins**" -> "**Manage Plugins**".

2. Select the Available tab. Search for `gitlab-plugin` and select it to install.



3. Select **Manage Jenkins** -> **Configure System**.

4. In the GitLab section, select **Enable authentication for '/project' end-point**.

5. Select **Add**, then choose **Jenkins Credential Provider**.

6. Select **GitLab API token** as the token type.

7. In **API Token**, paste the access token value you copied from GitLab and select **Add**.

8. Enter the GitLab server's URL in **GitLab host URL**.

9. To test the connection, select **Test Connection**

Gitlab

Gitlab host URL	<input type="text" value="https://gitlab.example.com"/>
API Token	<input type="text" value="dwkXdHB-Xznc7_4uxg2H"/>
Ignore SSL Certificate Errors	<input type="checkbox"/>

The complete URL to the Gitlab server (i.e. http://gitlab)

API Token for accessing Gitlab

Test Connection

Configure the Jenkins project

Set up the Jenkins project you intend to run your build on.

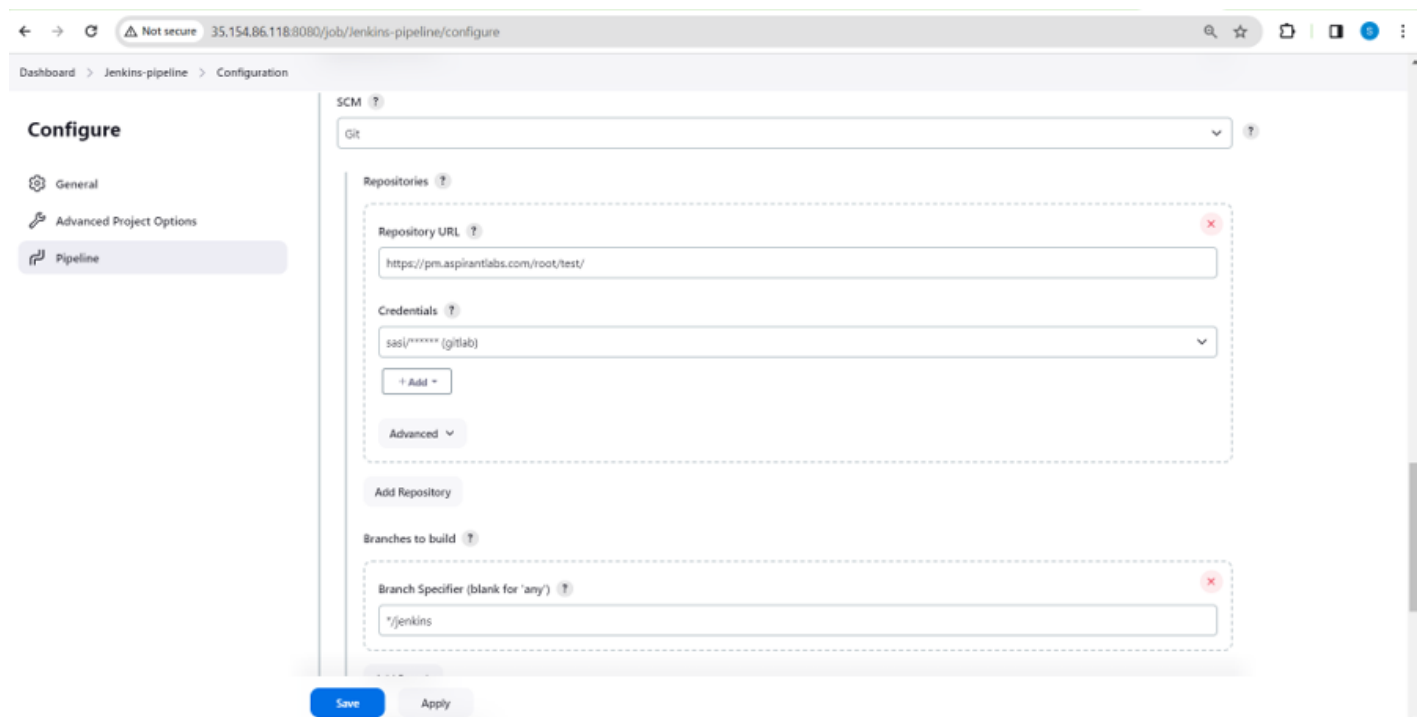
1. On your Jenkins instance, select **New Item**.

2. Enter the project's name.

3. Select **Freestyle** or **Pipeline** and select **OK**. You should select a freestyle project, because the Jenkins plugin updates the build status on GitLab. In a pipeline project, you must configure a script

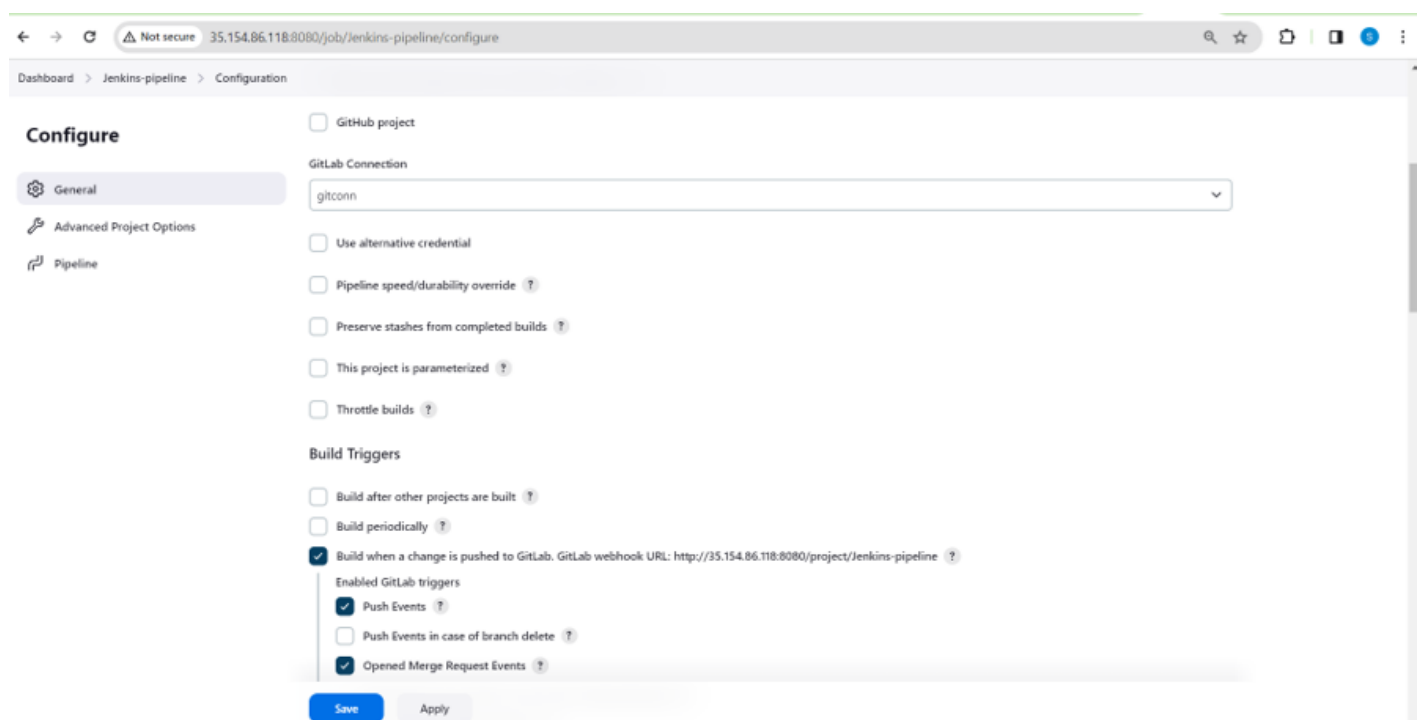
to update the status on GitLab.

4.Choose your GitLab connection from the dropdown list.



The screenshot shows the Jenkins Pipeline Configuration page. The left sidebar has a 'Configure' section with 'General', 'Advanced Project Options', and 'Pipeline' tabs. The 'General' tab is selected. The main area is titled 'SCM' and shows a dropdown menu set to 'Git'. Below this is a 'Repositories' section with a 'Repository URL' field containing 'https://pm.aspirantlabs.com/root/test/' and a 'Credentials' dropdown set to 'sas/***** (gitlab)'. There is an 'Add Repository' button and an 'Advanced' dropdown. Below that is a 'Branches to build' section with a 'Branch Specifier (blank for 'any')' field containing '*/jenkins'. At the bottom are 'Save' and 'Apply' buttons.

5.Select Build when a change is pushed to GitLab.



The screenshot shows the Jenkins Pipeline Configuration page. The left sidebar has a 'Configure' section with 'General', 'Advanced Project Options', and 'Pipeline' tabs. The 'General' tab is selected. The main area is titled 'Build Triggers'. It has a 'GitLab Connection' dropdown set to 'gitconn'. Below this are several checkboxes: 'GitHub project', 'Use alternative credential', 'Pipeline speed/durability override', 'Preserve stashes from completed builds', 'This project is parameterized', and 'Throttle builds'. The 'Build Triggers' section has a checkbox 'Build when a change is pushed to GitLab. GitLab webhook URL: http://35.154.86.118:8080/project/Jenkins-pipeline' which is checked. Below this is a 'Enabled GitLab triggers' section with checkboxes: 'Push Events' (checked), 'Push Events in case of branch delete', and 'Opened Merge Request Events' (checked). At the bottom are 'Save' and 'Apply' buttons.

6.Select the following checkboxes:

Accepted Merge Request Events

Closed Merge Request Events

7.Specify how the build status is reported to GitLab:

*If you created a freestyle project, in the **Post-build Actions section**, choose **Publish build status to GitLab**.*

If you created a pipeline project, you must use a Jenkins Pipeline script to update the status on GitLab.

Dashboard > First_Job > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

☐ Build after other projects are built ?

☐ Build periodically ?

☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://3.85.201.30:8080/project/First_Job ?

Enabled GitLab triggers

- ☒ Push Events ?
- ☐ Push Events in case of branch delete ?
- ☒ Opened Merge Request Events ?
- ☐ Build only if new commits were pushed to Merge Request ?
- ☐ Accepted Merge Request Events ?
- ☐ Closed Merge Request Events ?

Rebuild open Merge Requests ?

Never

- ☒ Approved Merge Requests (EE-only) ?
- ☒ Comments ?

Comment (regex) for triggering a build ?

Save Apply

8.Generate a token

copy and save the job

9.Go to webhook paste the url and secret token

🏠

🔗

📄

🔍 Search or go to...

Issues0

Merge requests0

Manage>

Plan>

</> Code>

🔧 Build>

🔒 Secure>

🚀 Deploy>

⚙️ Operate>

🖥️ Monitor>

📊 Analyze>

⚙️ Settings▼

General

Integrations

Webhooks

🆘 Help

🔍 Search page

Webhook

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

URL

http://3.85.201.30:8080/project/First_Job

URL must be percent-encoded if it contains one or more special characters.

☒ Show full URL

☐ Mask portions of URL

Do not show sensitive data such as tokens in the UI.

Secret token

.....

Used to validate received payloads. Sent with the request in the `X-GitLab-Token` HTTP header.

Trigger

☒ Push events

☐ All branches

☒ Wildcard pattern

jenkins

Wildcards such as `*-stable` or `production/*` are supported.

☐ Regular expression

☐ Tag push events

A new tag is pushed to the repository.

☐ Comments

Webhook

URL

URL must be percent-encoded if it contains one or more special characters.

Secret token

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

- ☒ Push events
 - ☐ All branches
 - ☒ Wildcard pattern

Wildcards such as `*-stable` or `production/*` are supported.

10. Now make the changes in gitlab and check the job will or not

11.If the job will trigger , check the console output

Console Output

View as plain text

Edit Build Information

Thread Dump

Pause/resume

Replay

Pipeline Steps

Workspaces

Previous Build

```
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/runner-pip
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Status Jenkins)
[Pipeline] echo
Check service status
[Pipeline] sh
+ systemctl status nginx.service
• nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2023-11-27 08:31:03 UTC; 2h 6min ago
  Docs: man:nginx(8)
  Main PID: 523 (nginx)
  Tasks: 2 (limit: 1121)
  Memory: 3.5M
  CPU: 49ms
  OGroup: /system.slice/nginx.service
         └─523 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
           └─528 "nginx: worker process" ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Production)
[Pipeline] echo
Pulled the code
[Pipeline] sh
+ systemctl status odoo
• odoo.service - Odoo Open Source ERP and CRM
  Loaded: loaded (/lib/systemd/system/odoo.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2023-11-27 08:31:02 UTC; 2h 6min ago
  Main PID: 378 (python3.7)
  Tasks: 4 (limit: 1121)
  Memory: 4.6M
```

Revision #10

Created Fri, Jan 5, 2024 4:57 AM by [sasi](#)Updated Wed, Apr 3, 2024 7:44 AM by [sasi](#)