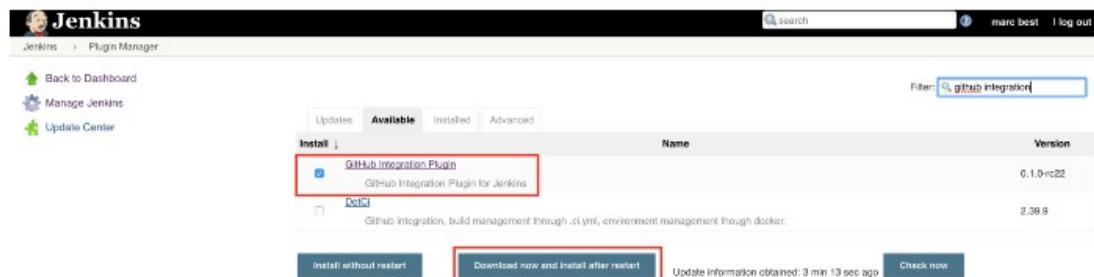


1. Install Github Integration Plugin

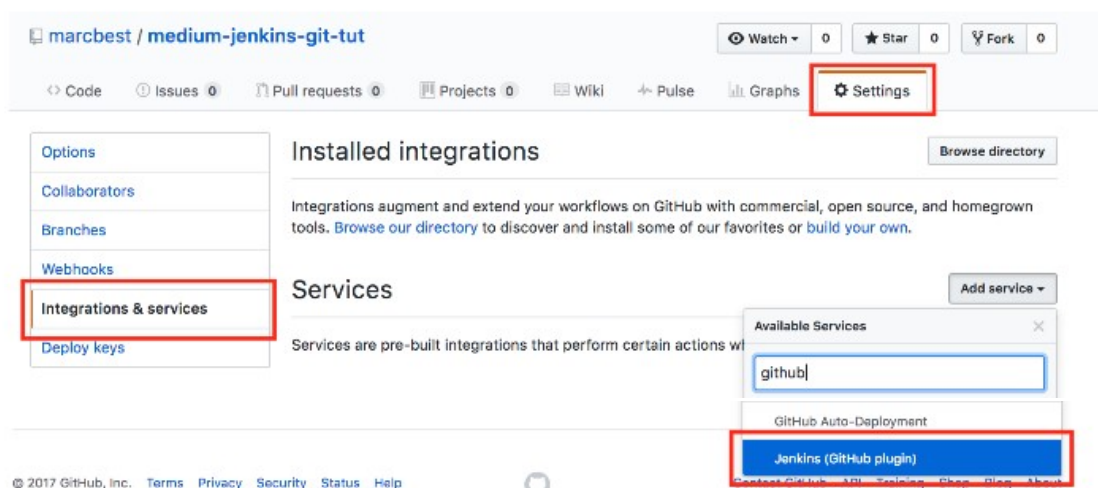
First we need to install the GitHub Integration Plugin, this will give us the ability to configure Jenkins to use our Github repository.



Installing Github integration plugin

2. Prepare Github repository

We need to add a service to call the Jenkins Github webhook on a push, to do this go to settings -> integrations & Services and add a new service. The Jenkins Github plugin service should be in the list of available services.



Enter the URL of your Jenkins instance followed by /github-webhook/

Services / **Manage Jenkins (GitHub plugin)** Test service

Jenkins is a popular continuous integration server.

Using the Jenkins GitHub Plugin you can automatically trigger build jobs when pushes are made to GitHub.

Install Notes

- "Jenkins Hook Url" is the URL of your Jenkins server's webhook endpoint. For example: `http://ci.jenkins-ci.org/github-webhook/`.

For more information see <https://wiki.jenkins-ci.org/display/JENKINS/GitHub+plugin>.

Jenkins hook url

`http://{YOUR_JENKINS_URL}/github-webhook/`

☒ **Active**
We will run this service when an event is triggered.

Update service Delete service

3. Giving the Jenkins user access to the Github repository

We need to give the Jenkins user access to our repository by adding a deploy key in the Github settings.

The first step is generating SSH keys for the Jenkins user if they do not already exist.

```
jenkins@ip:/home/ubuntu$ ssh-keygen
```

Depending on where the key was created, you need to copy the public key so that it can be added to Github

```
jenkins@ip:/home/ubuntu$ cat /var/lib/jenkins/.ssh/id_rsa.pub
```

Add the key copied in the previous step to Github. To do this head to the repository settings -> Deploy keys

marcbest / medium-jenkins-git-tut

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Options Collaborators Branches Webhooks Integrations & services Deploy keys

Deploy keys

Add deploy key

There are no deploy keys for this repository

Title

Jenkins

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQCoD1zZ6nhwzXyQF1p7wiHM1dBDAS1FycffGCGuh0R4dHGbbObfvl3wU3ddPk3FNbY5Khzkwc+ZEouIBZyp0wQ/ZU7Q1VPBle0N22sJZ1izNcdKu6wS51+qQizKdc+4w6nGoSf3L/q2kpg6Yb4ZYfaWTPZs50MOIBCu69/Yw+O9+rvrd4U8D3cDr/5yghbJJuvzsrHtvx9VYmRTk8nofRFom2IDbALyk4cvOAti7ofz7UVE0dkqcufoWndDa8FrDqPr/iuM8EcnUEhCo6fnALelxdPegvr0OMwLnUkpxQIBDalqfaRL2DApH5fpqf0jXiZs2vCsYCtDJ71b0p/pjenkins@ip-172-31-22-109
```

☐ Allow write access
Can this key be used to push to this repository? Deploy keys always have pull access.

Add key

The last step is to check that everything is working as expected, as the Jenkins user in your console enter the below to check the connection to Github.

```
jenkins@ip:~/.ssh$ ssh git@github.com
```

If successful you should see the following message

```
Warning: Permanently added the RSA host key for IP address  
'{YOUR_SERVER_IP}' to the list of known hosts.
```

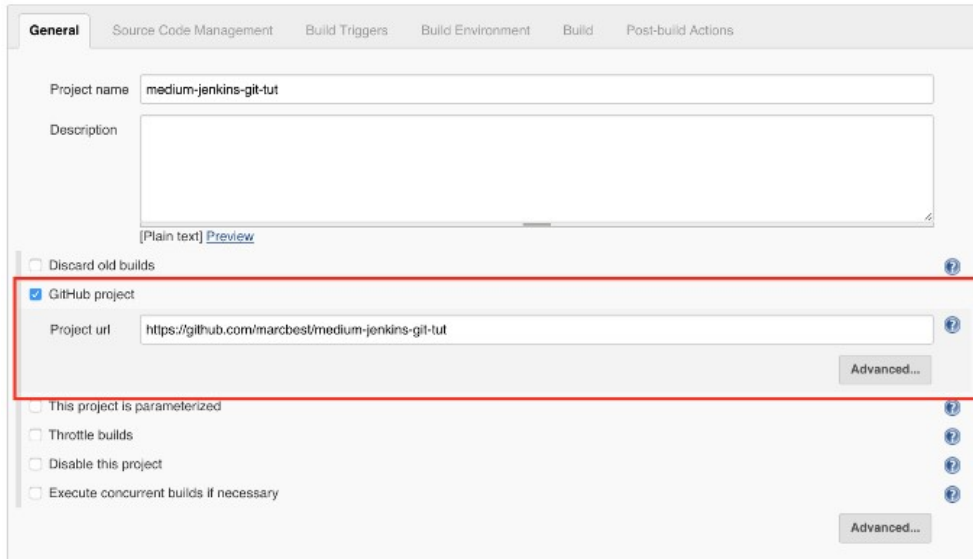
```
PTY allocation request failed on channel 0
```

```
Hi marcbest/medium-jenkins-git-tut! You've successfully  
authenticated, but GitHub does not provide shell access.
```

```
Connection to github.com closed.
```

4. Update Jenkins job with Github configuration

In the 'General' section of the job configuration check the Github project tick box and enter the URL to the repository that you configured in step 2.



The screenshot shows the Jenkins job configuration interface with the 'General' tab selected. The 'Project name' field is set to 'medium-jenkins-git-tut'. The 'Description' field is empty. Below the description, there is a 'Discard old builds' checkbox which is unchecked. The 'GitHub project' checkbox is checked and highlighted with a red rectangular box. Below this checkbox, the 'Project url' field contains the text 'https://github.com/marcbest/medium-jenkins-git-tut'. To the right of the 'Project url' field is an 'Advanced...' button. Below the 'GitHub project' section, there are four more checkboxes: 'This project is parameterized' (unchecked), 'Throttle builds' (unchecked), 'Disable this project' (unchecked), and 'Execute concurrent builds if necessary' (unchecked). To the right of these checkboxes are four help icons (question marks). At the bottom right of the configuration area is another 'Advanced...' button.

Next update the Source Code Management section, first set the repository URL (note the format `git@github.com:{YOUR_REPO}`). You can also specify the branch you would like to use.

Source Code Management

☐ None
☒ Git

Repositories

Repository URL: `git@github.com:marcbest/medium-jenkins-git-tut`

Credentials: `- none -` [Add](#)

[Advanced...](#)
[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any'): `*/master` [Add Branch](#)

The last step is to tell Jenkins to build when the Github hook is called, select the highlighted option below in the Build Triggers section.

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)
☐ Build after other projects are built
☐ Build periodically
☐ GitHub Branches
☐ GitHub Pull Requests
☒ GitHub hook trigger for GITScm polling
☐ Poll SCM

That's it! Your Jenkins build should now be triggered whenever a push is made to your repository.