Module 8 –Project

Lab Guide

© 2014 Brain4ce Education Solutions Pvt. Ltd.

Version 1.0

DevOps Project

Lab Guide

Contents

edureka!

Problem Statement

→ A request comes from a client in between a software development lifecycle to make a few changes in the product

→ Even though the changes seems to be minor on product side but these changes involve huge number on major and minor changes in the current build

→ Now, this new requirement is brought to the team, the management decides that nearly 10 developers will be doing the changes simultaneously

Solution

To accomplish many changes in the build in short span

→ A Git repository was created to keep the data centralized

→ The source control was done using the Git repository

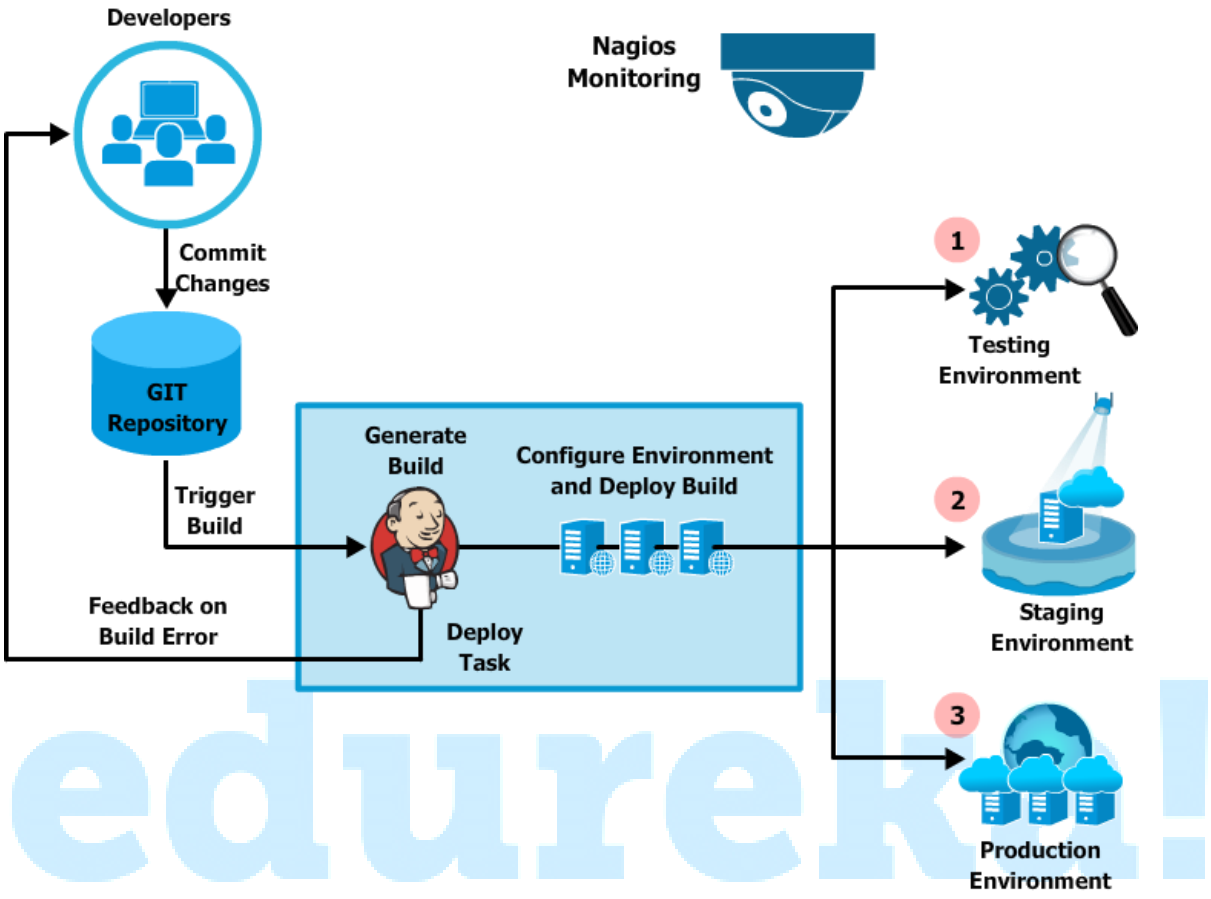
→ Nagios was configured to monitor the services and outages

→ Jenkins was configured in such a way that as soon as there is a change in the Git repository, automatically it will run a SCM(Source Code Management) job which will compile the code and make a new build

→ This way continuous Integration was achieved

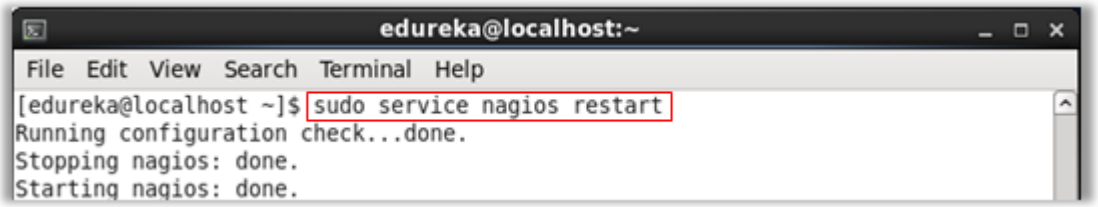
edureka!

DevOps Project Flow

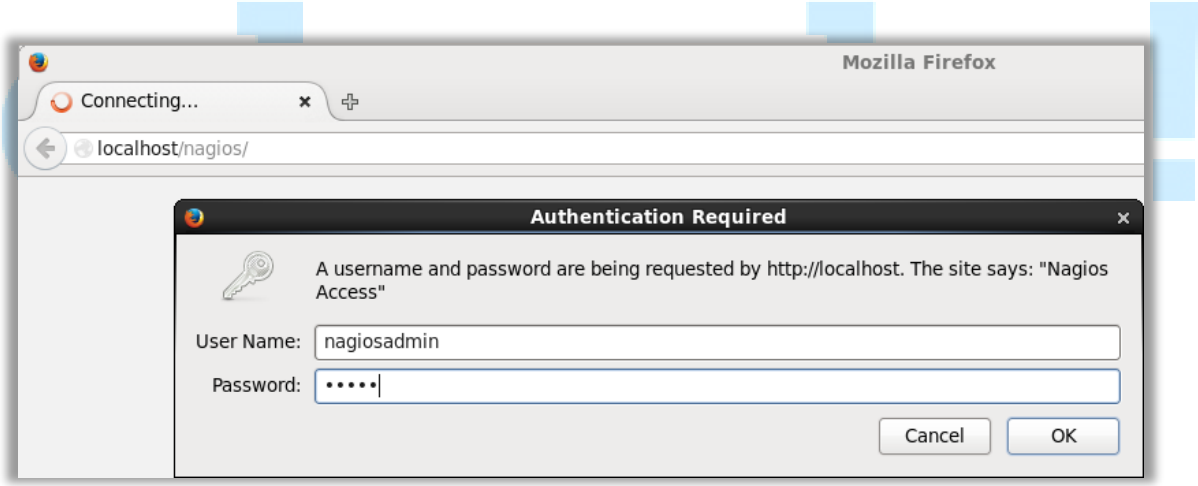


Initialising Nagios

Step 1: Starting Nagios Service from the terminal

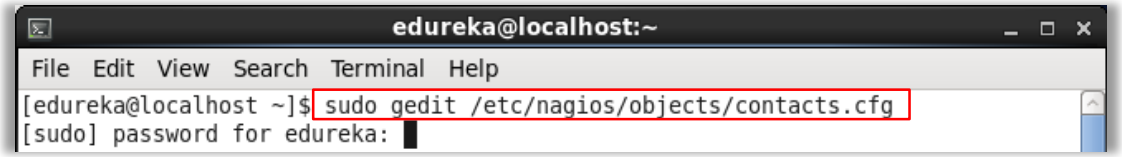


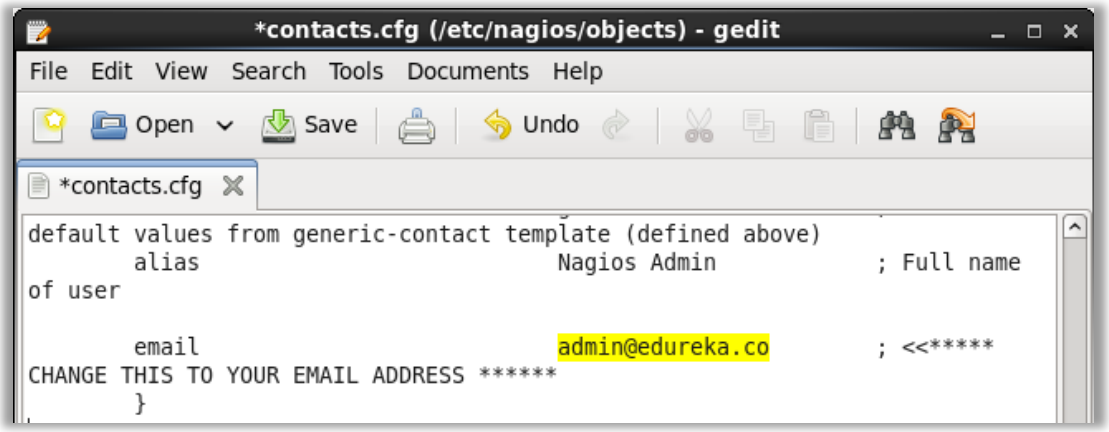
Step 2: Initiating Nagios from browser



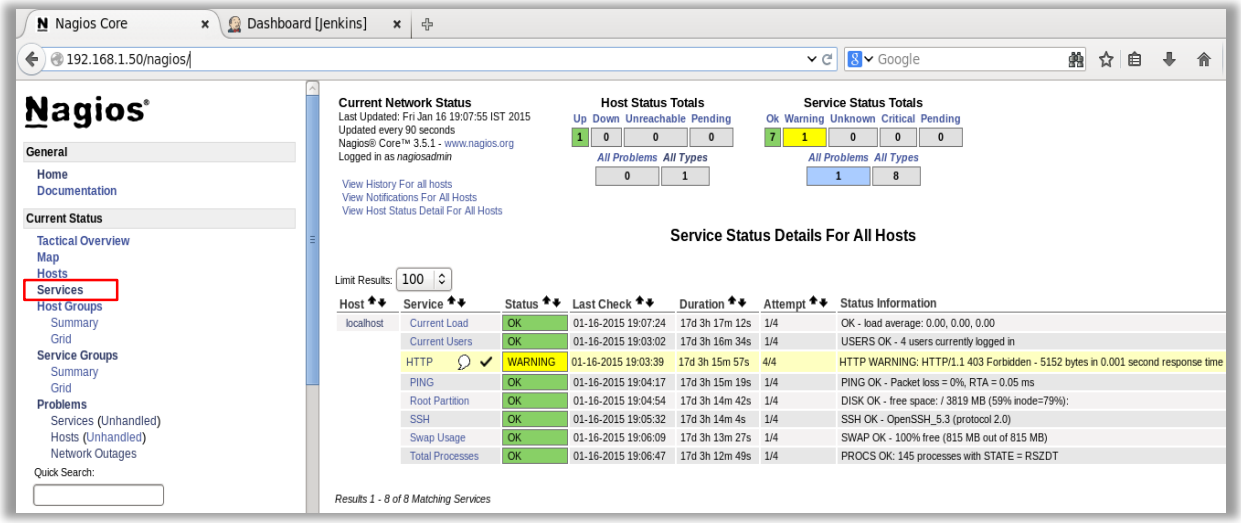
Step 3: Give the ip address/nagios to access Nagios Service

To get automatic notifications for all updates, we have to set our email id in "contacts.cfg" file. Then with every change made in the monitored services, an alert notification is sent to the email id.

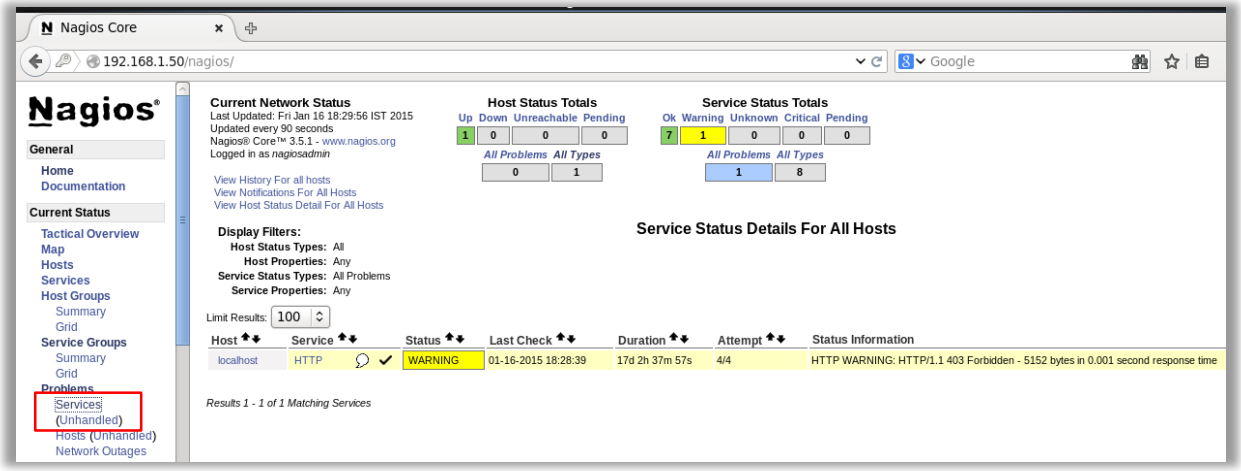




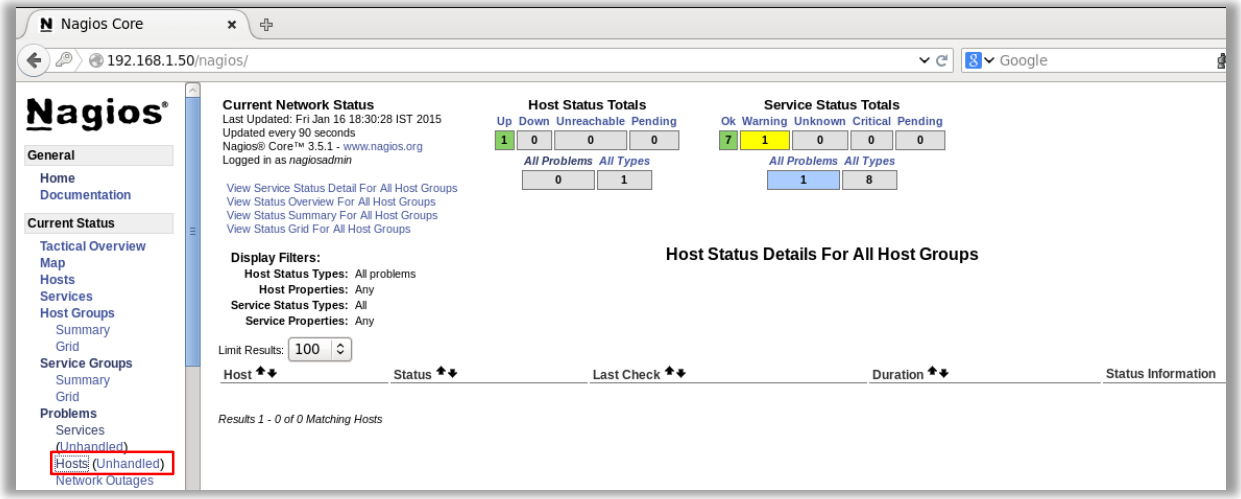
To check the Monitored services by Nagios, select the "Services" option from the menu on the left



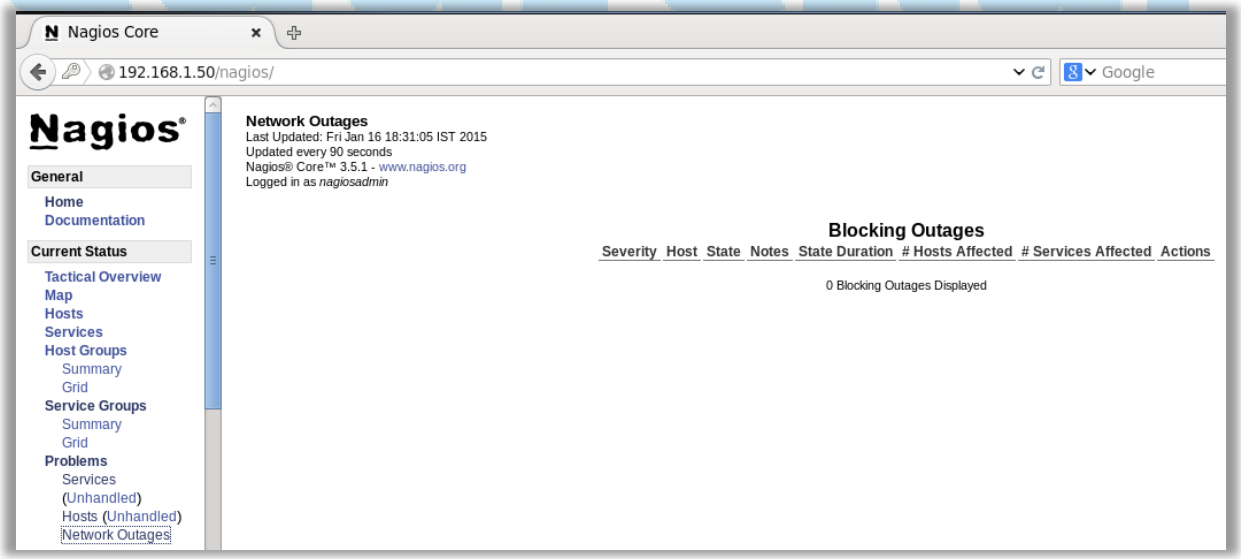
To fix the Unhandled Issues by Nagios, select the "Services (Unhandled)" option from the menu on the left



To check the hosts monitored by Nagios, select the "Hosts (Unhandled)" option from the menu on the left



To check the outages monitored by Nagios, select the "Network Outages" option from the menu on the left



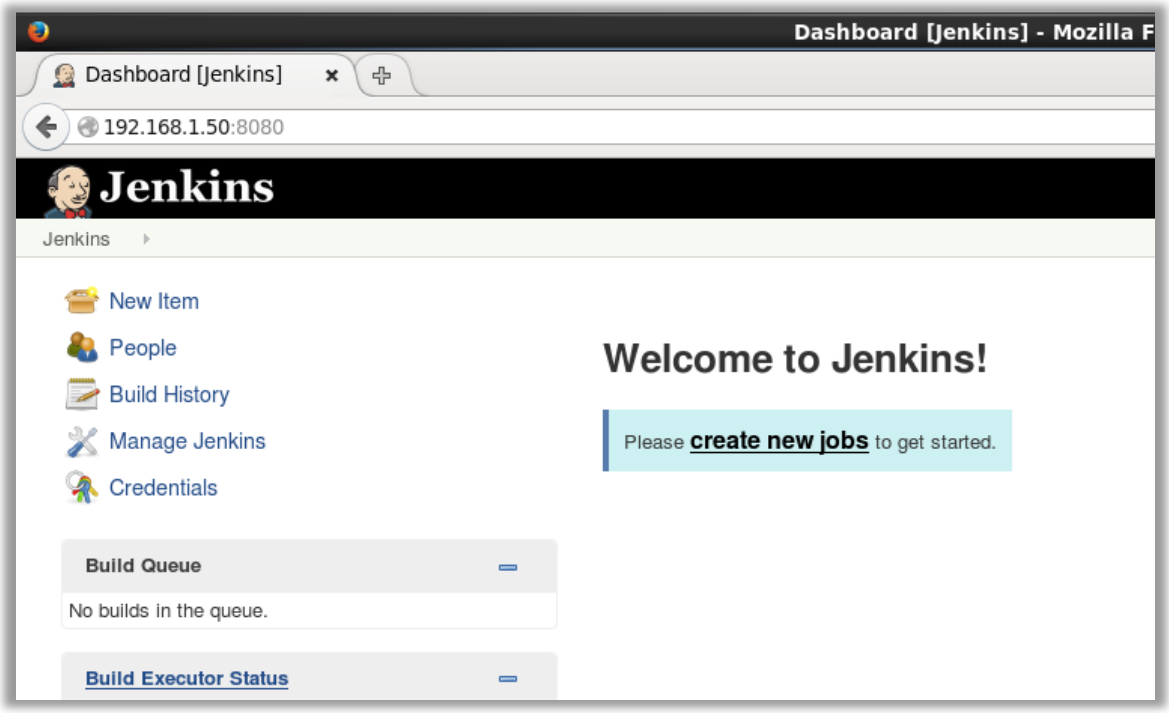
Working with Jenkins

Step 1: Check for Jenkins services form the terminal if it is working or not and restart the service

```
edureka@localhost:~
File  Edit  View  Search  Terminal  Help
[edureka@localhost ~]$ sudo chkconfig jenkins on
[edureka@localhost ~]$
```

```
edureka@localhost:~
File  Edit  View  Search  Terminal  Help
[edureka@localhost ~]$ sudo service jenkins start
Starting Jenkins                                           [  OK  ]
[edureka@localhost ~]$
```

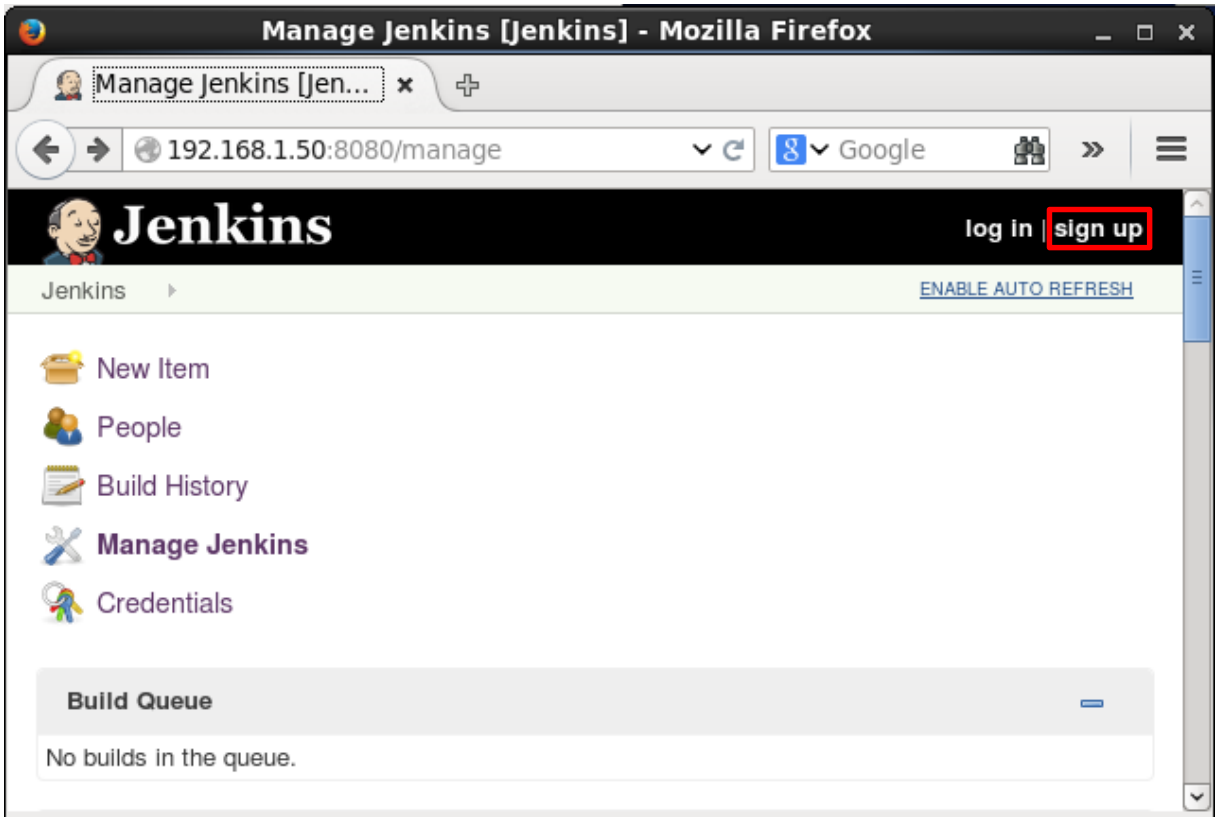
Take a new browser and open Jenkins (Give the IP address with the port number)



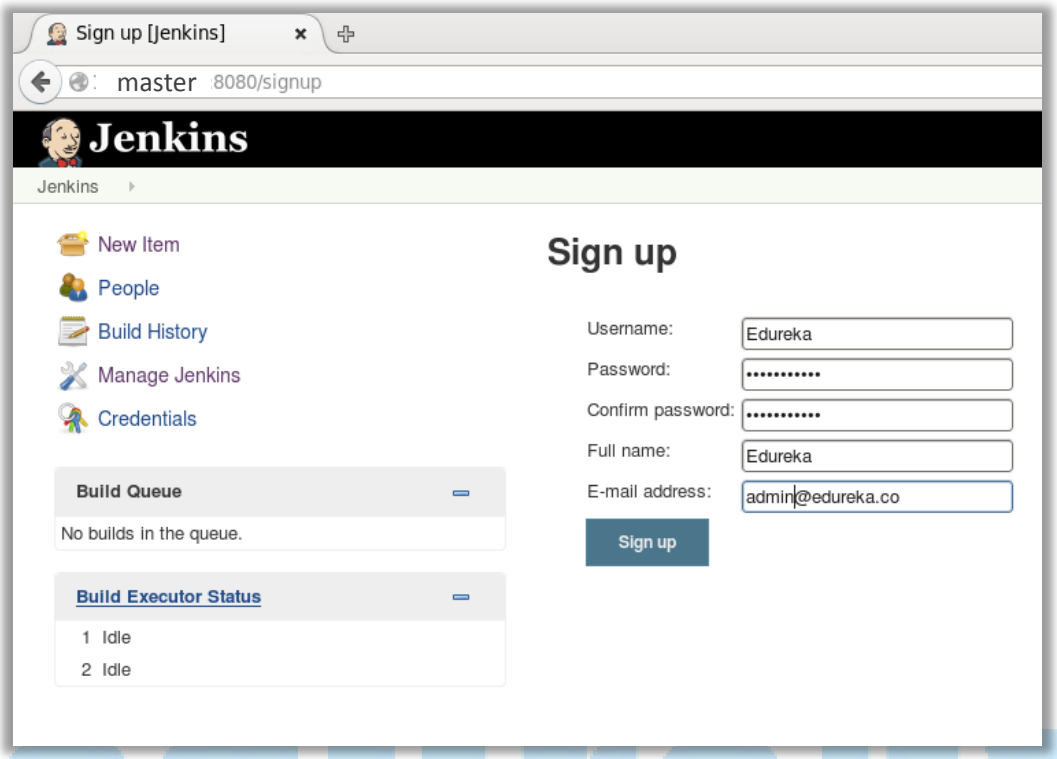
Configuring Jenkins

Sign up to create your Account for Jenkins.

Select the Sign Up option on the right

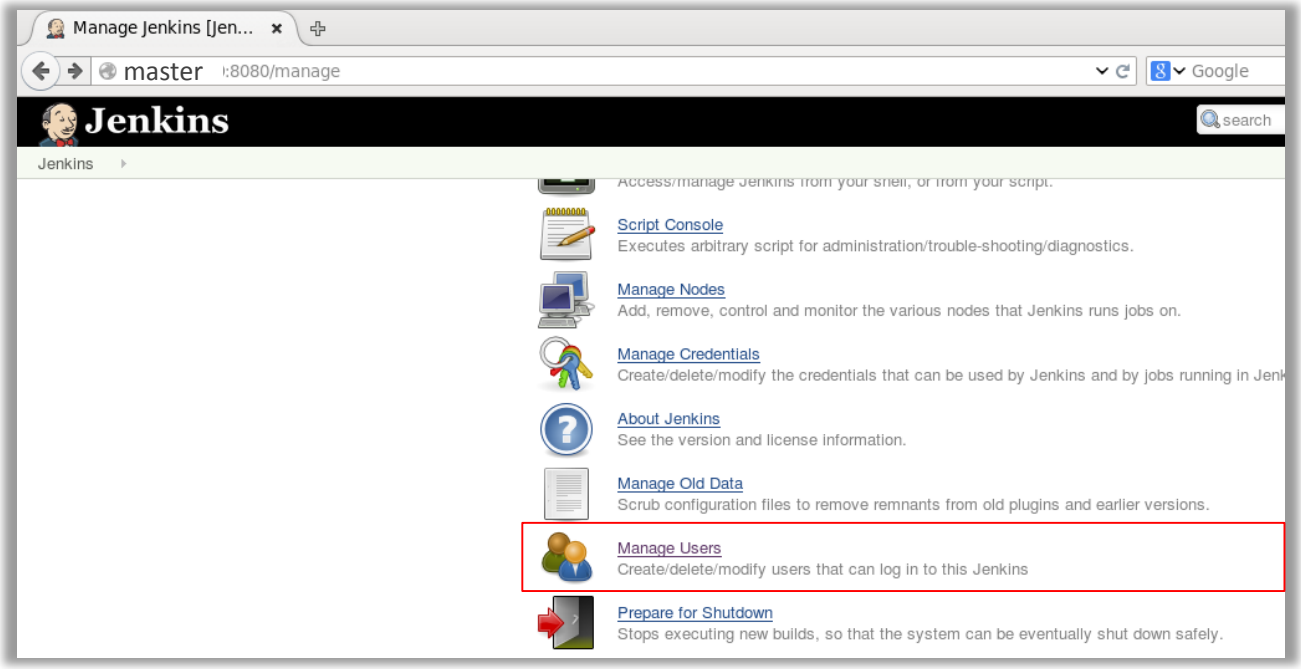


Give your credentials and create user.

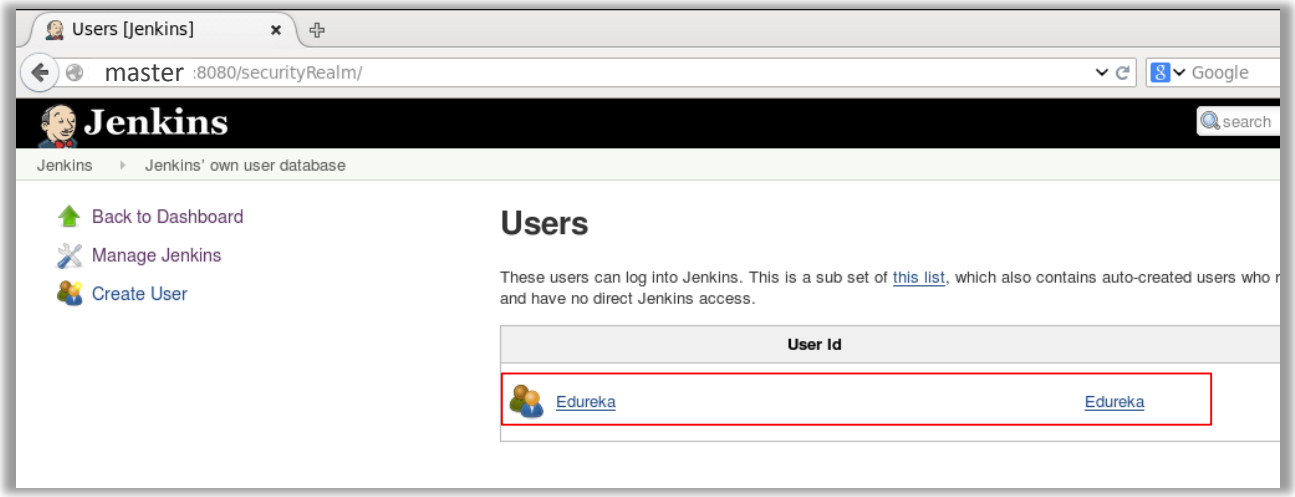


To check for the user, Select manage Jenkins form the menu on the left

Now, Select Manage Users form the menu displayed

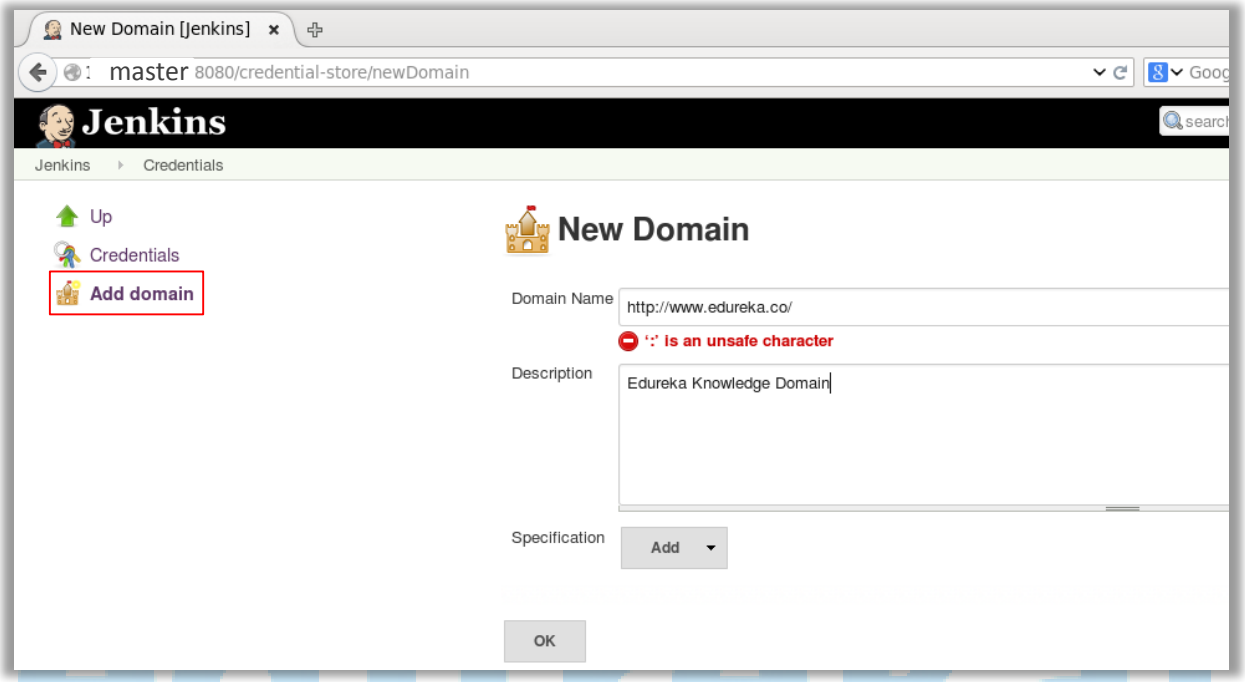


Check for the User

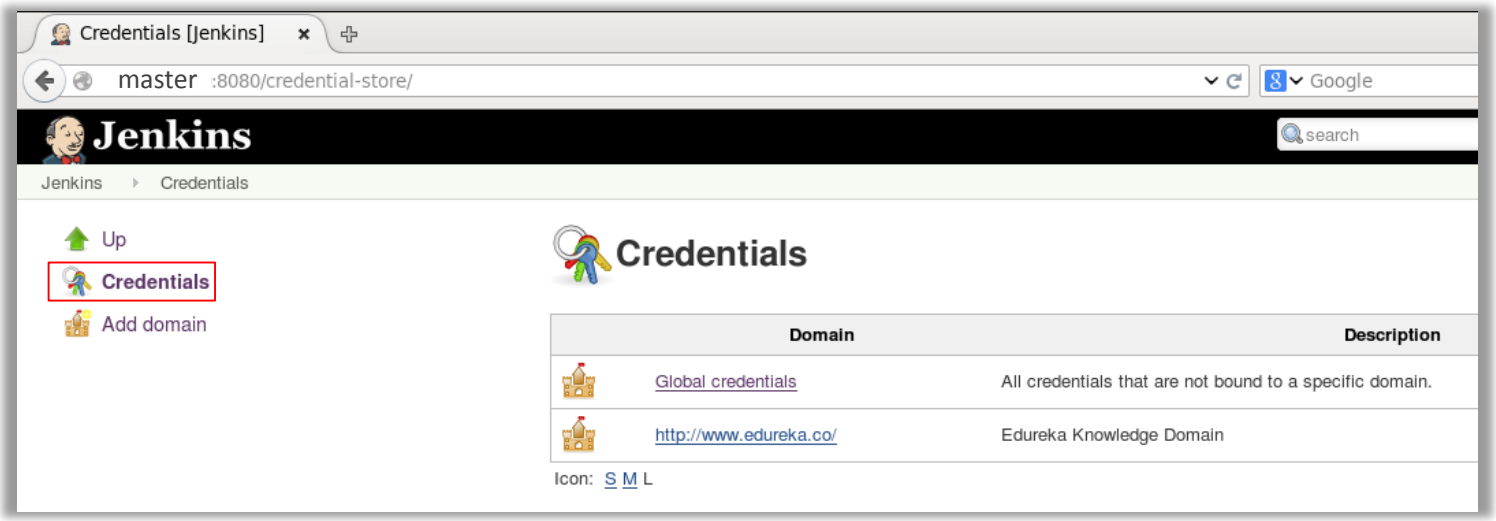


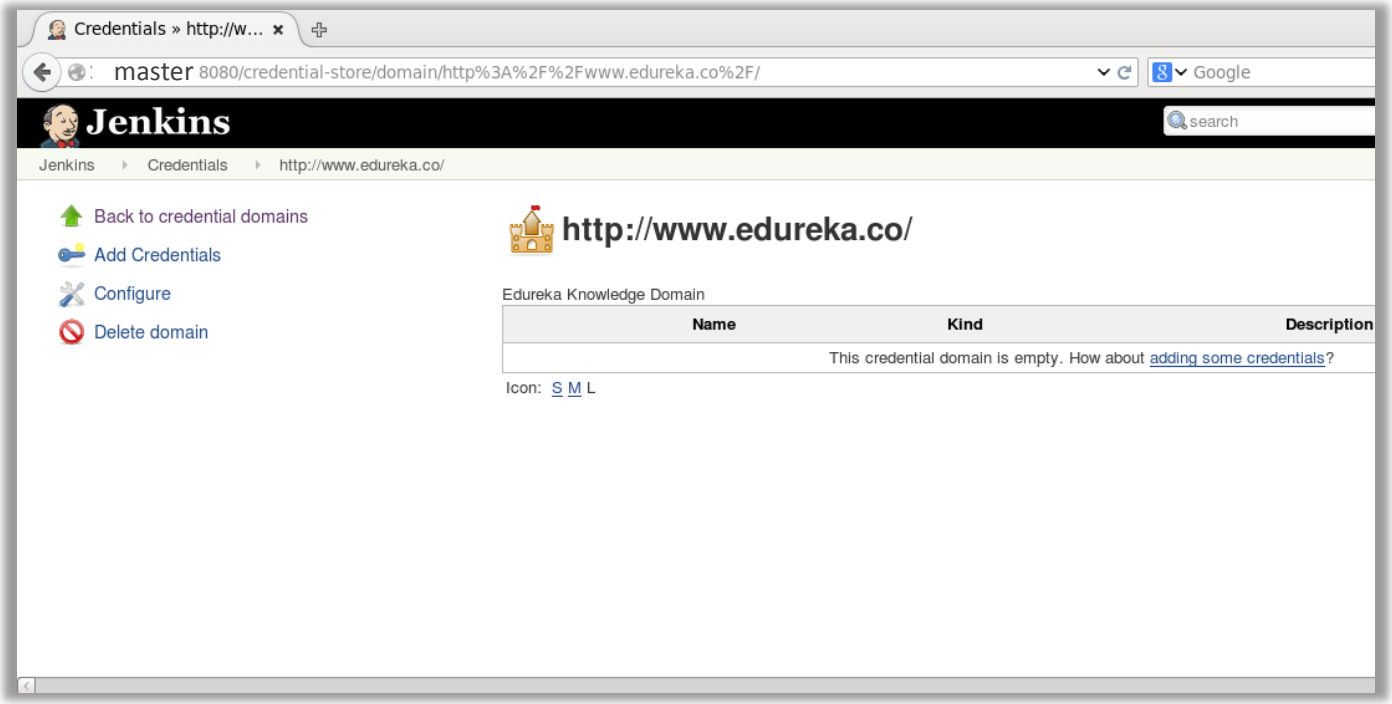
Now that you have signed up.

We have to set the domain for Users. Click on the Add domain option from the menu on the left



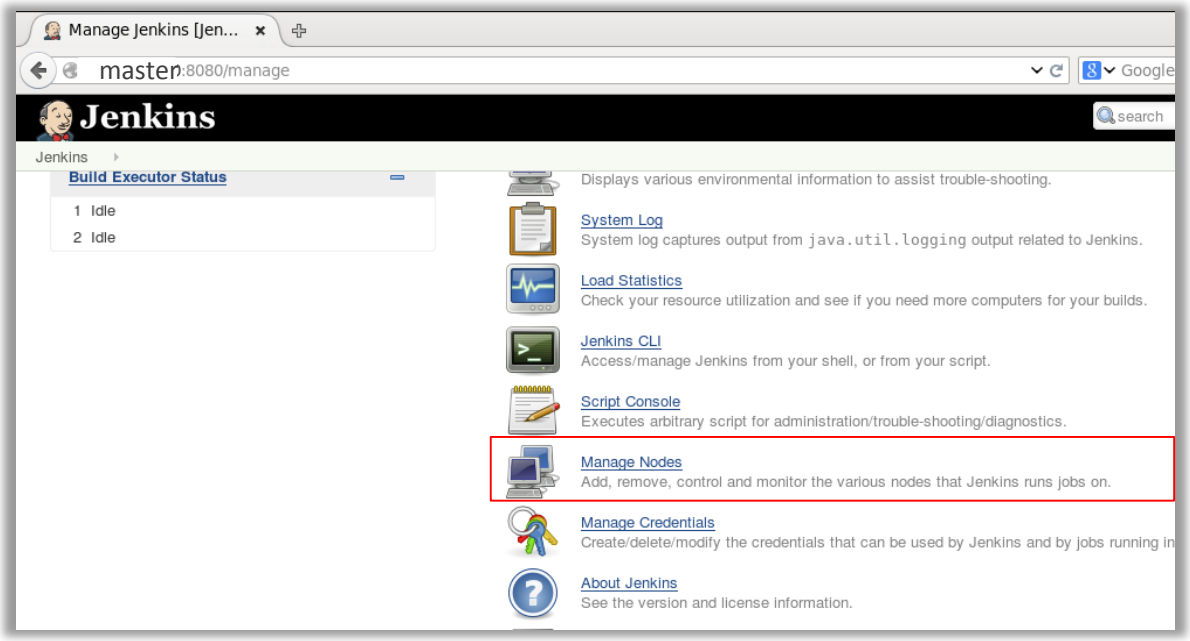
We can also set the desired credential for every user as per our requirement. To set the credential select Credentials Option from the left menu.



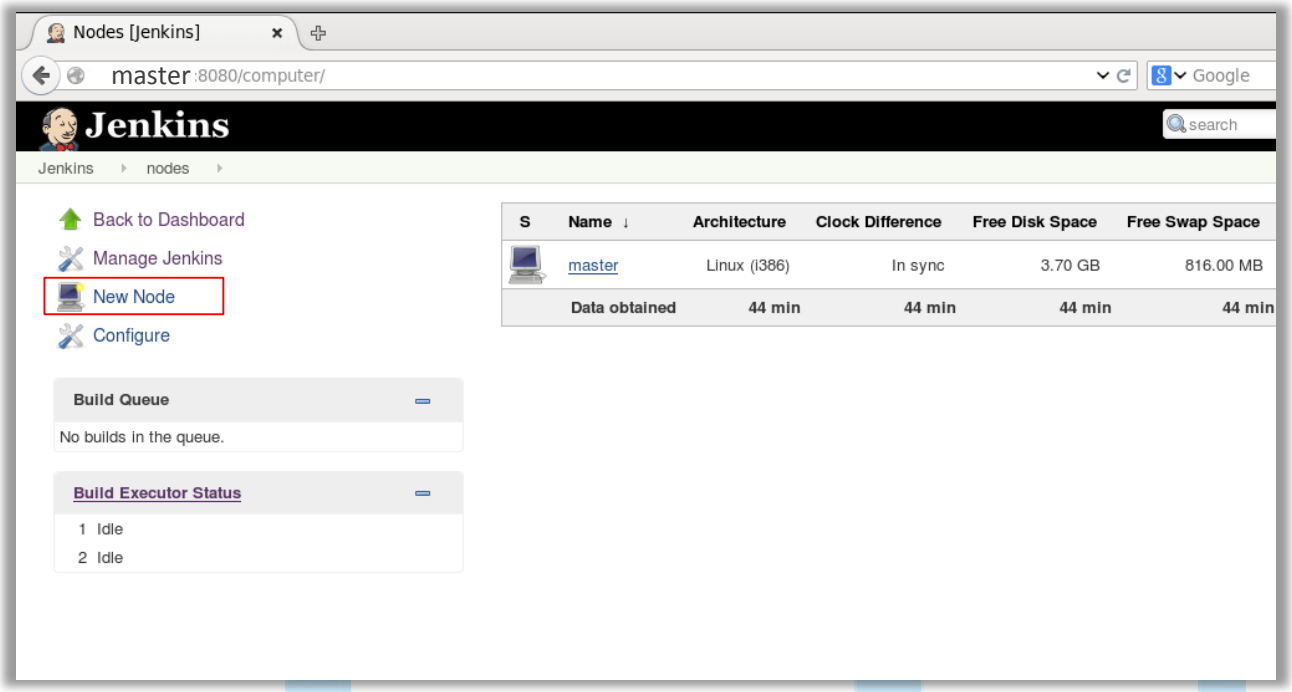


Many a times, the unit of work which we are working on cannot be handled by only one node. Also, if the node gets hung, all the work will stop.

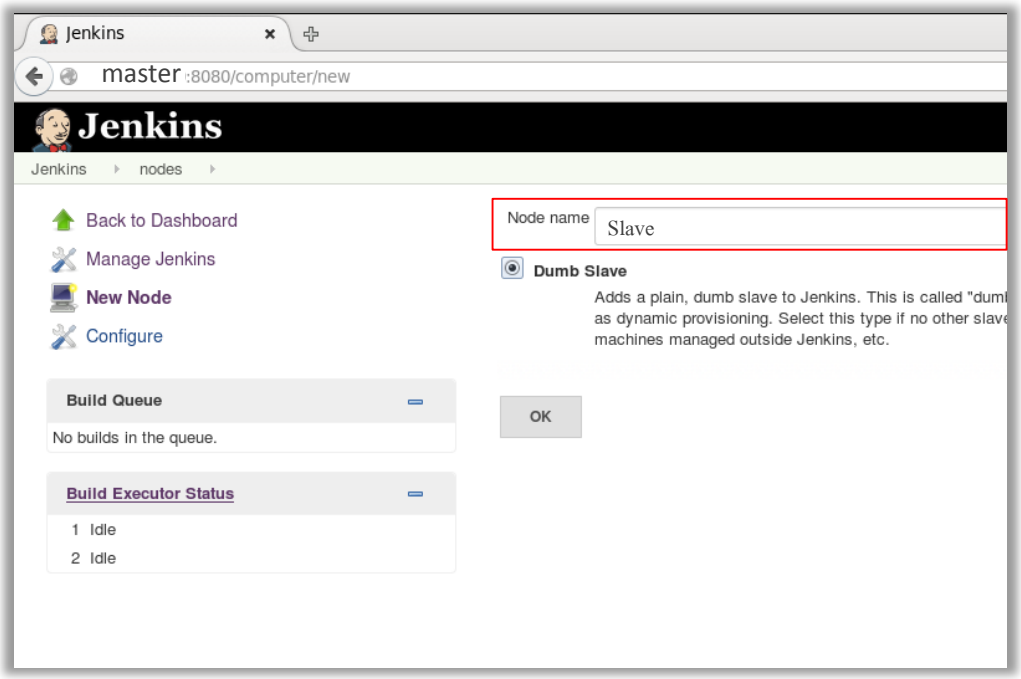
To resolve this, we have an option to Manage Nodes to distribute the load



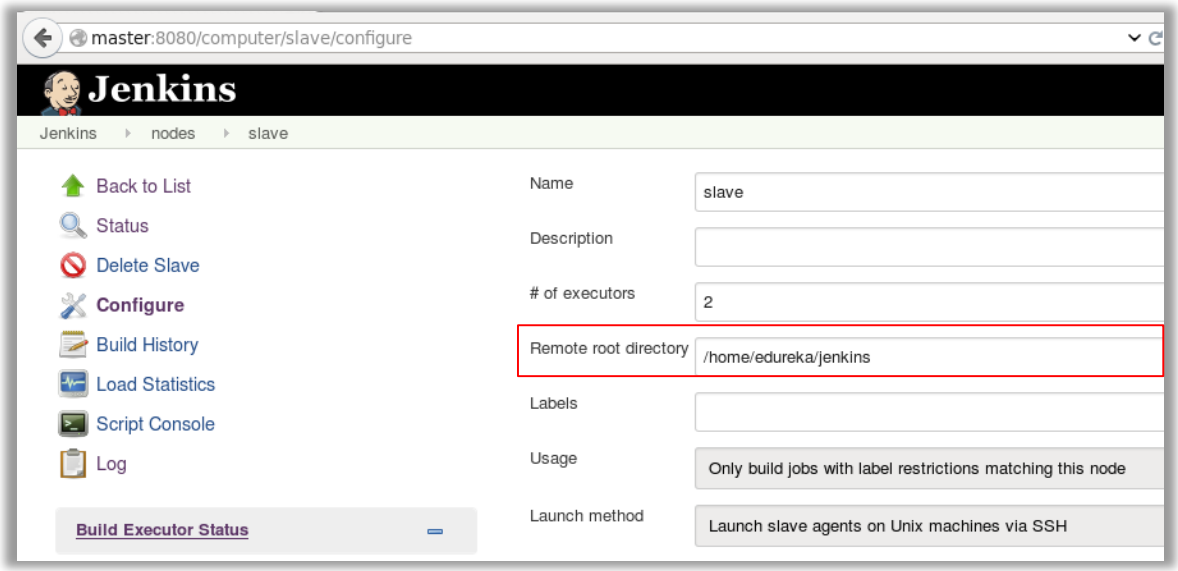
Select the New Node option.



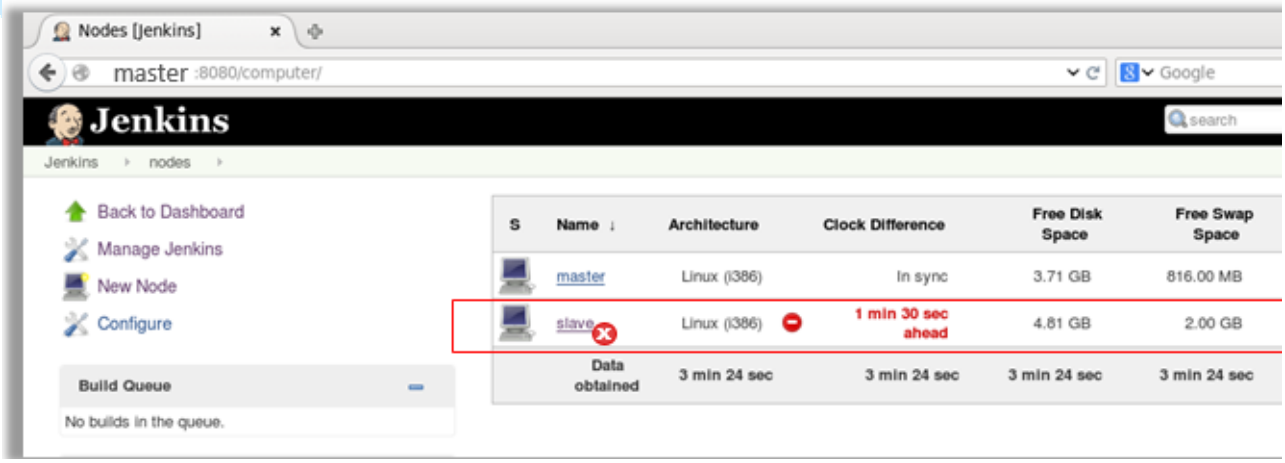
Give the Name for the node and select ok.



It will come to a page for further details. Set the Remote Root directory and Save

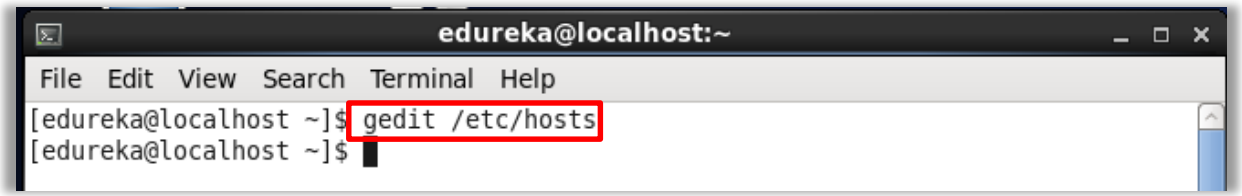


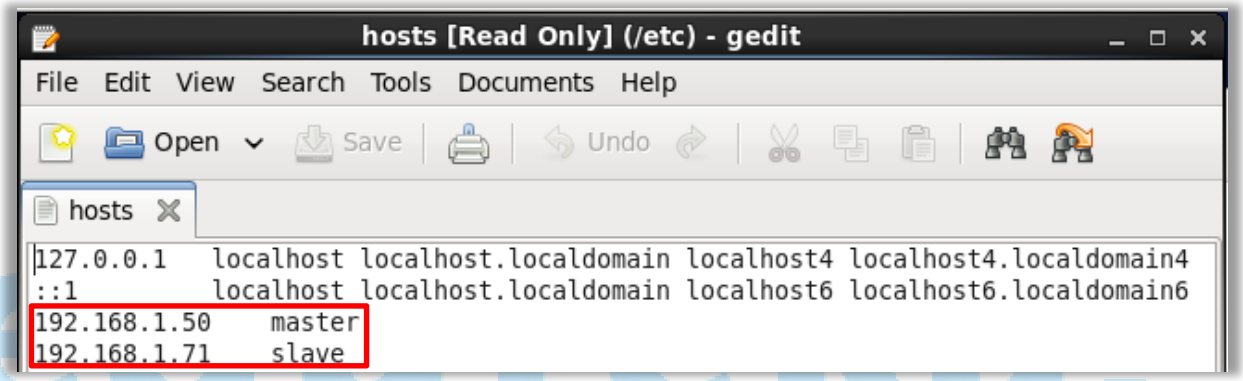
Now when we see the Slave node, it comes with a cross mark. To activate the slave node. We need to Launch the slave from the slave terminal

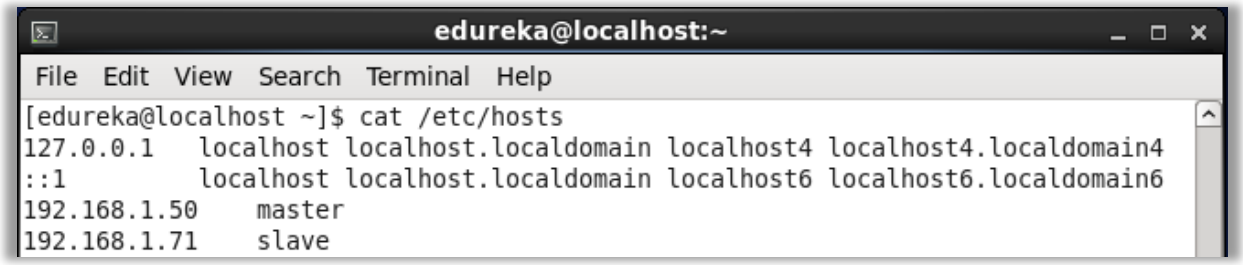


Before launching the Slave terminal.

Go to master and slave terminal separately, check for the IP with the help of ifconfig command and edit the host file with their IP. Execute the following command.



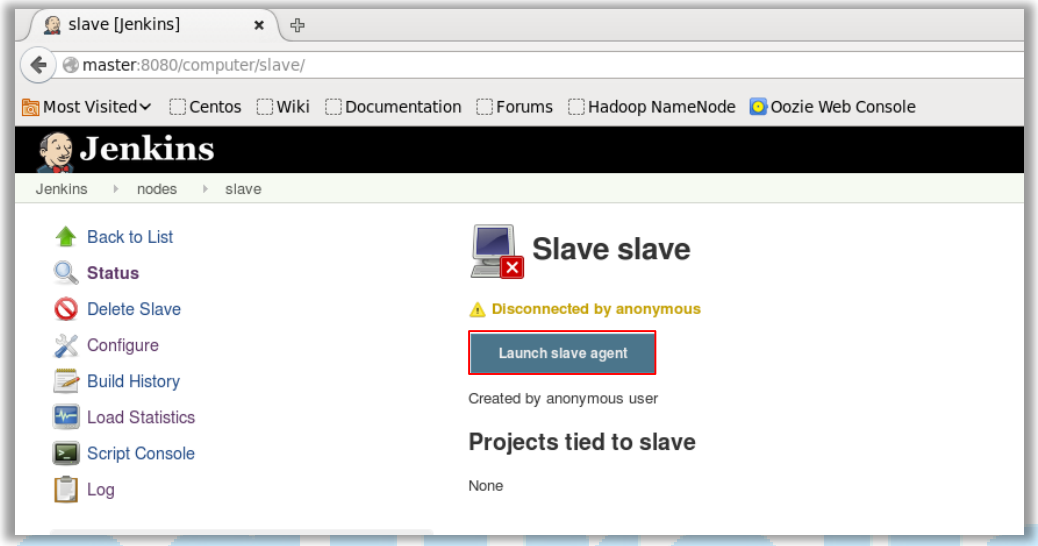




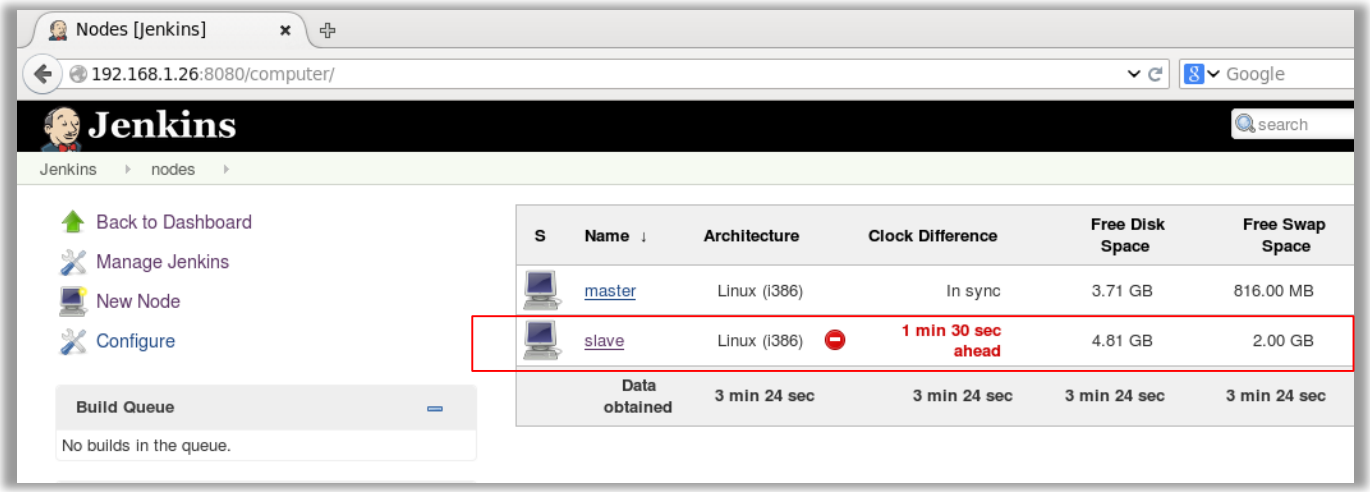
Note: The Ip of the system keeps on changing, if at times you are unable to access the master or the slave node, please recheck for the IP and the hosts file

Go to the Slave Terminal and select Launch slave Option.

Note: To access the Launch Slave option, the Slave node must also have Jenkin installed in it



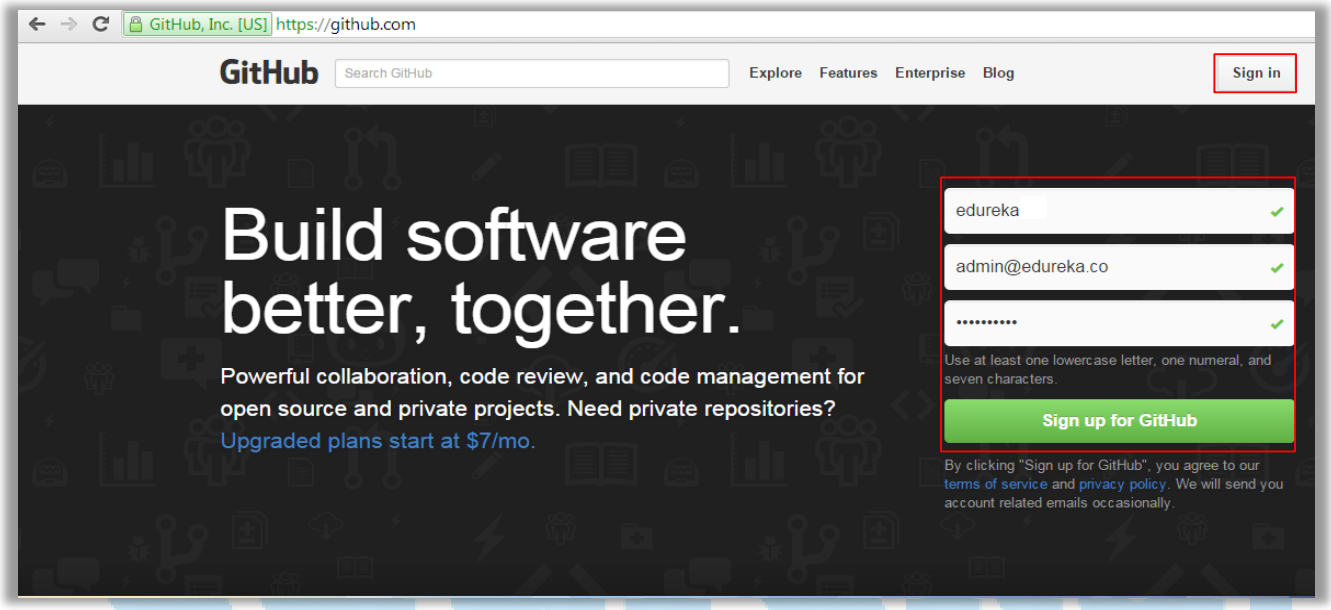
Now, when we check for the slave node, it is accessible

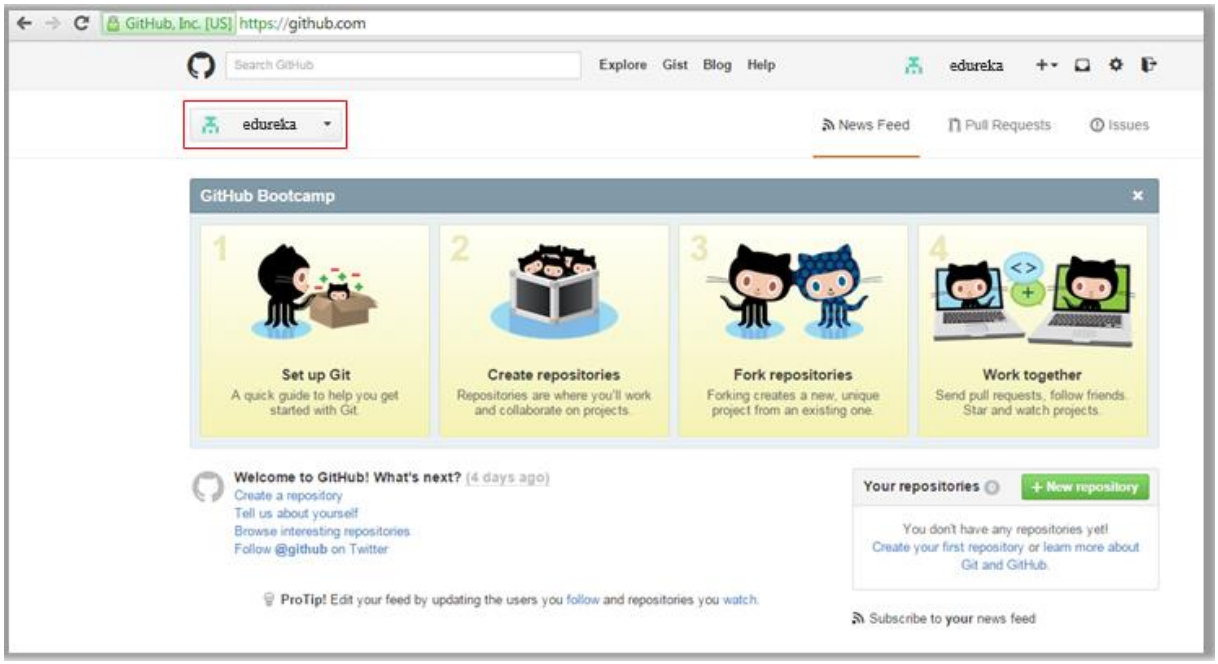


Setting up Git Account

To create a project, it is must to have a Git repository. Let's see how to set up a Git Repository

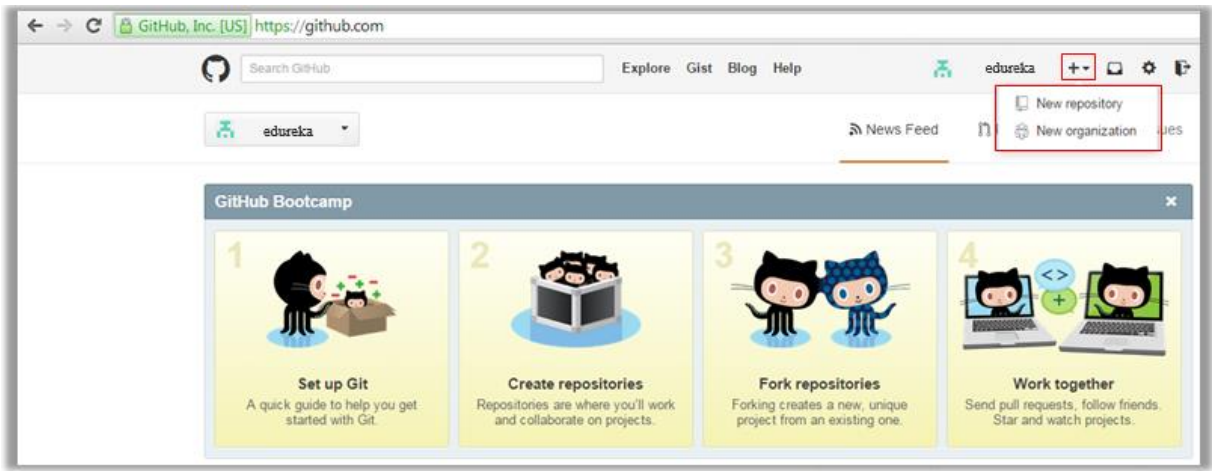
Go to **http://github.com**, and create your own account



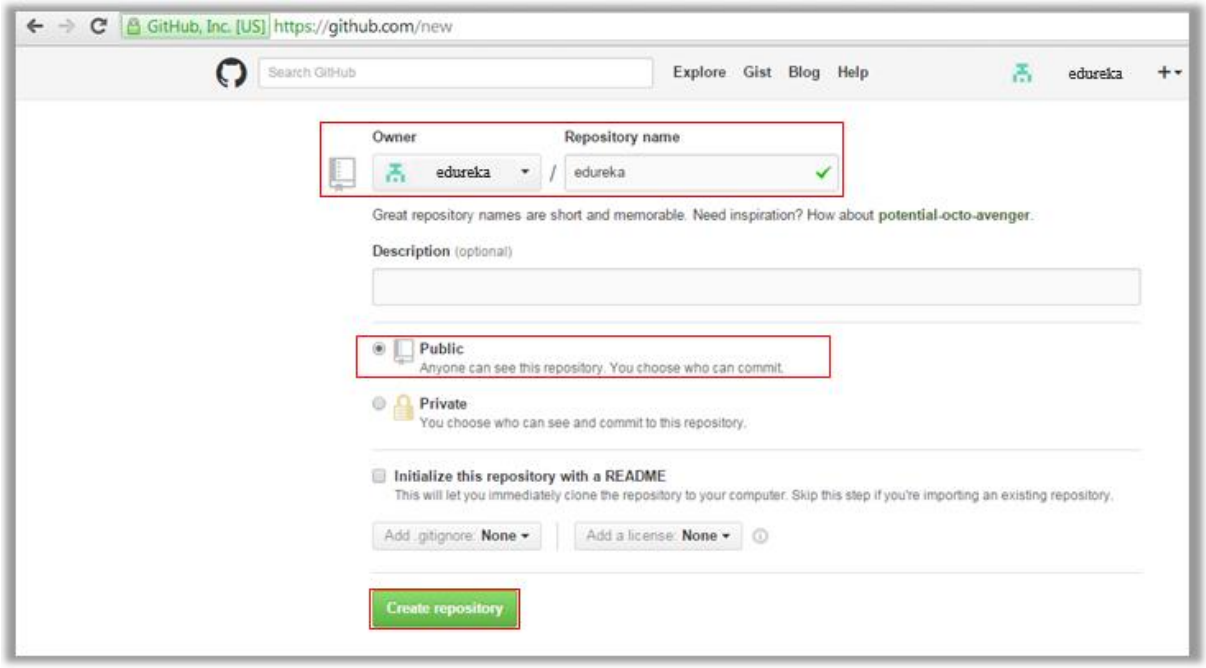


Now, we need to set a new repository.

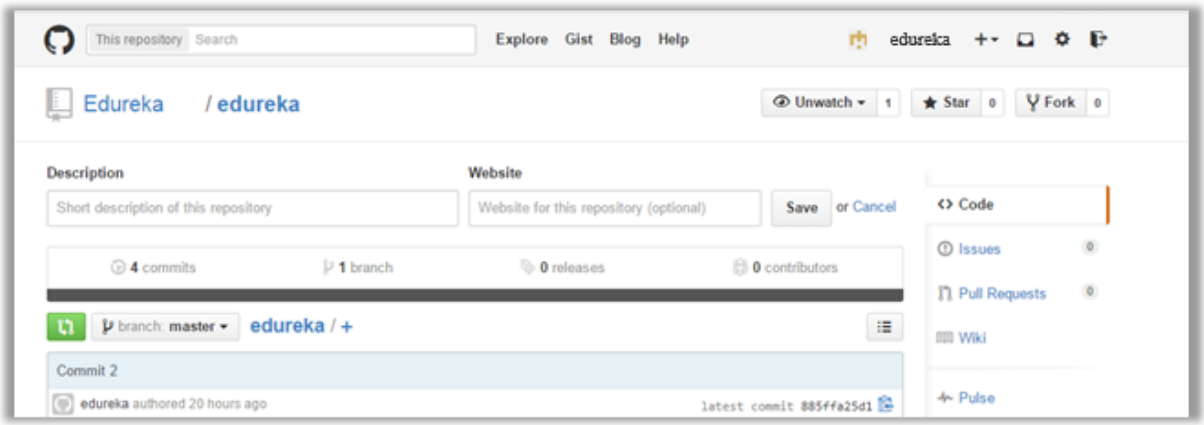
Click on the "+" and select New repository



Give the name of the repository, set it as Public and then click on Create Repository



Now we have a separate repository of our own

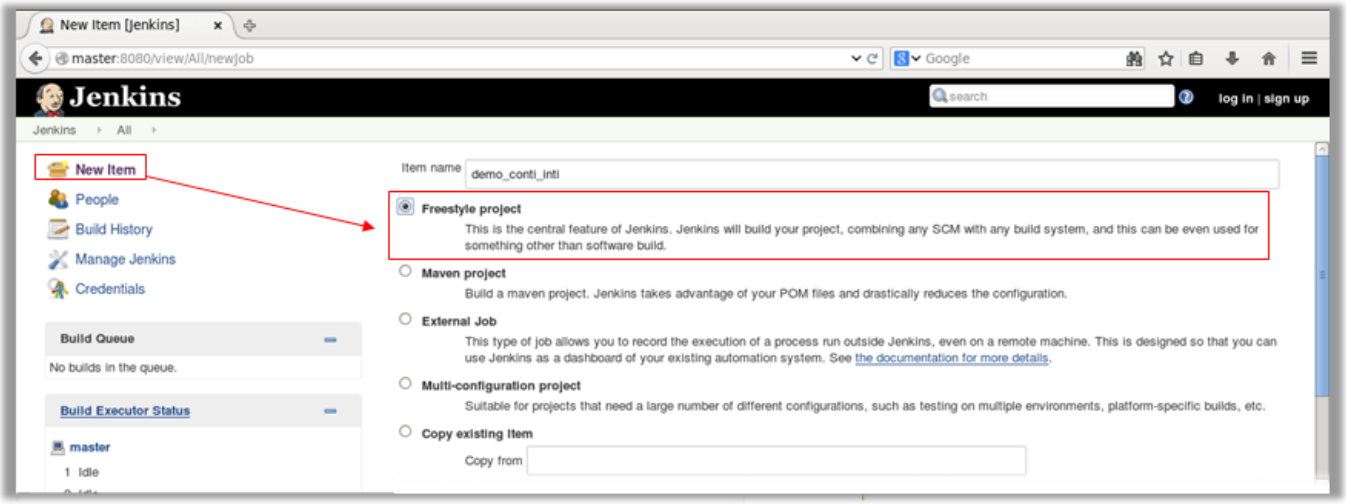


Create a Project

Let's see how we can create a Jenkins Project.

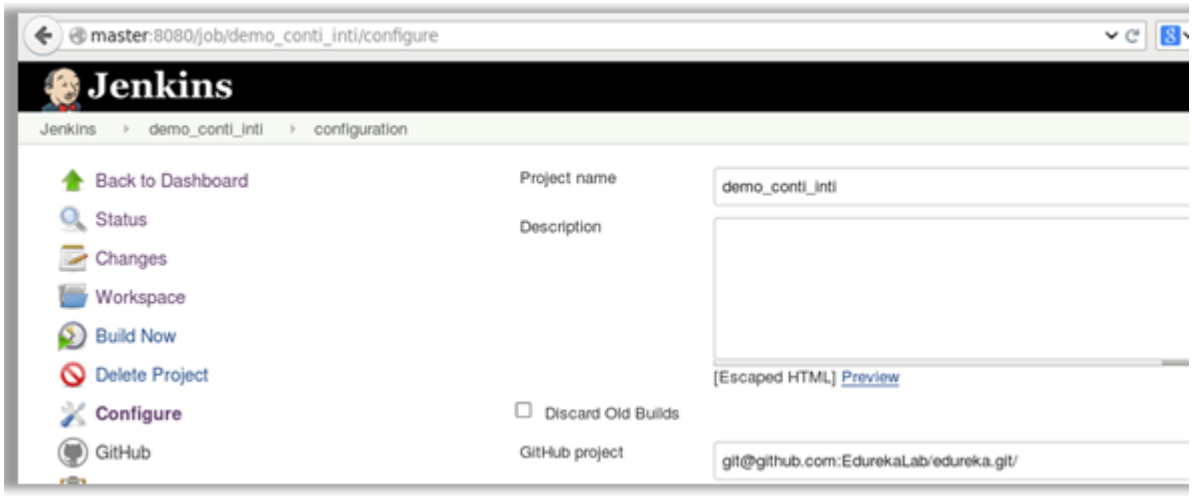
Go to your master node.

Select New Item option from left menu and Freestyle Project from Right



Give a proper name for the Project

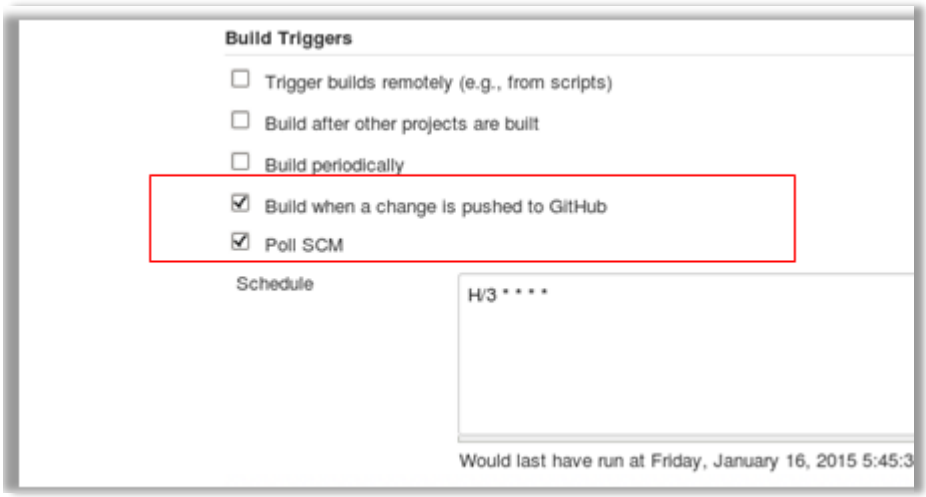
In GitHub project option: copy the SSH link from Git Account and paste there



For the Build Triggers option,

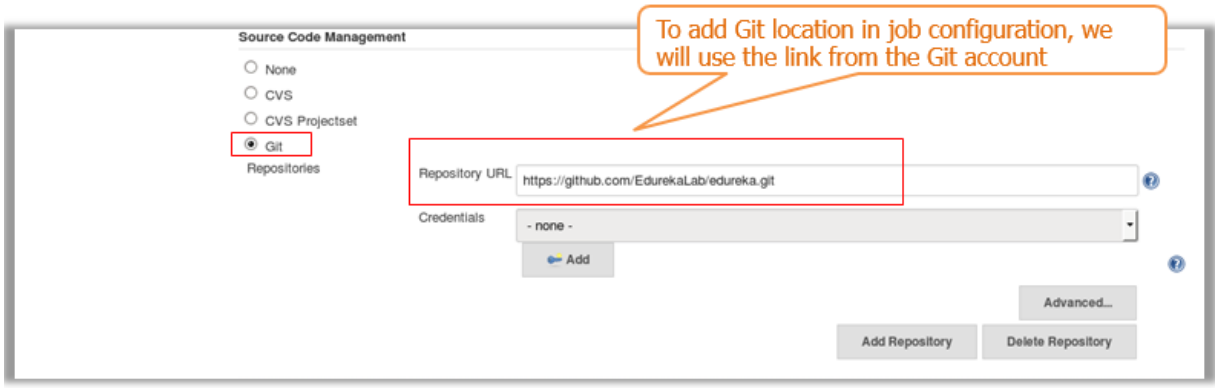
Select "Build when a change is published to GitHub" and "Poll SCM".

Schedule a periodic time frame for Schedule option.



Take the https link form Git Account and Paste in the Repository URL option

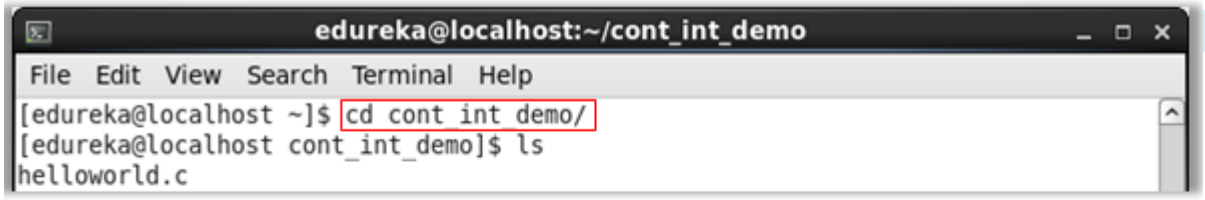
Finally save it

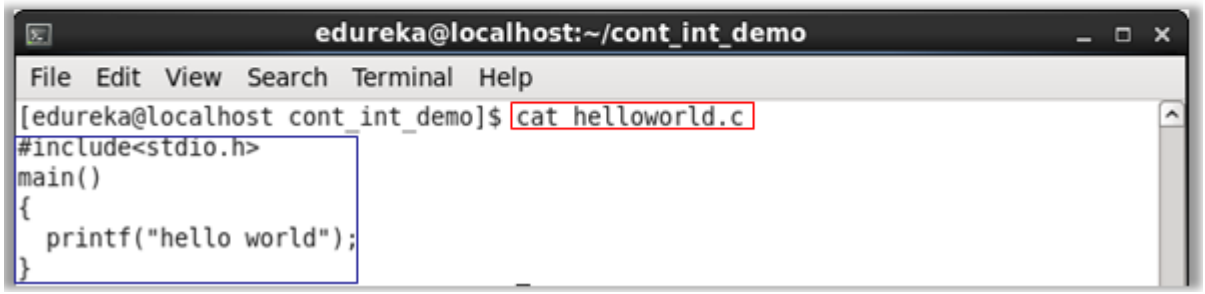


Now we will create our Source code.

Go to the terminal, and inside the project (Created in Jenkins), write the program to be executed.

Say "helloworld .c"





Now, check for the source code which need to be complied with the following command.

Then initialise the Git account

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ make helloworld
cc     helloworld.c   -o helloworld
[edureka@localhost cont_int_demo]$ ls
helloworld  helloworld.c
[edureka@localhost cont_int_demo]$
```

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git init
Initialized empty Git repository in /home/edureka/cont_int_demo/.git/
[edureka@localhost cont_int_demo]$
```

Now, when we check for the status of Git, we see that the programs have not been added.

So, we will add the program to it.

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git status -s
?? helloworld
?? helloworld.c
```

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git add helloworld.c
[edureka@localhost cont_int_demo]$ git status -s
A  helloworld.c
?? helloworld
```

File added

After making the changes to the repository, it is necessary to commit the changes.

To commit the changes made, execute the below command

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git commit -m 'Initial commit'
[master (root-commit) b8210c6] Initial commit
 Committer: edureka <edureka@localhost.localdomain>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

If the identity used for this commit is wrong, you can fix it with:

    git commit --amend --author='Your Name <you@example.com>'

 1 files changed, 5 insertions(+), 0 deletions(-)
 create mode 100644 helloworld.c
```

Now, when we go inside the Git folder and check folder, we will see that the remote location of repository has not been added

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ cd .git/
[edureka@localhost .git]$ ls
branches        config       HEAD   index  logs     refs
COMMIT_EDITMSG  description  hooks  info   objects
```

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[edureka@localhost .git]$ vi config
```

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
```

So we will add the remote location of out Git repository with the following command and then we check the config file.

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git remote add origin git remote add origin g
it@github.com:EdurekaLab/edureka.git
```

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[edureka@localhost .git]$ vi config
```

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = git@github.com:EdurekaLab/edureka.git
        fetch = +refs/heads/*:refs/remotes/origin/*
```

We could see that the location has been added

Let's push the code now, but before pushing the code to Git, we need to generate the SSH key in our terminal and add it to the repository

Let's see how to add Ssh Key

Execute the below command to generate the key.

```
edureka@localhost:~
File  Edit  View  Search  Terminal  Help
[edureka@localhost ~]$ ssh-keygen -t rsa -C   admin@edureka.co"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/edureka/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/edureka/.ssh/id_rsa.
Your public key has been saved in /home/edureka/.ssh/id_rsa.pub.
The key fingerprint is:
68:42:89:2a:ce:39:b5:c0:56:68:98:eb:04:ef:3d:a3 prachi@edureka.co
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|.... .           |
|+o..o            |
|++..   .         |
|+=... o S        |
|B.+..o           |
| *..+            |
|  .. o           |
|  E              |
+-----------------+
```

Provide your own Git email Id

Check for the key and then, copy the key

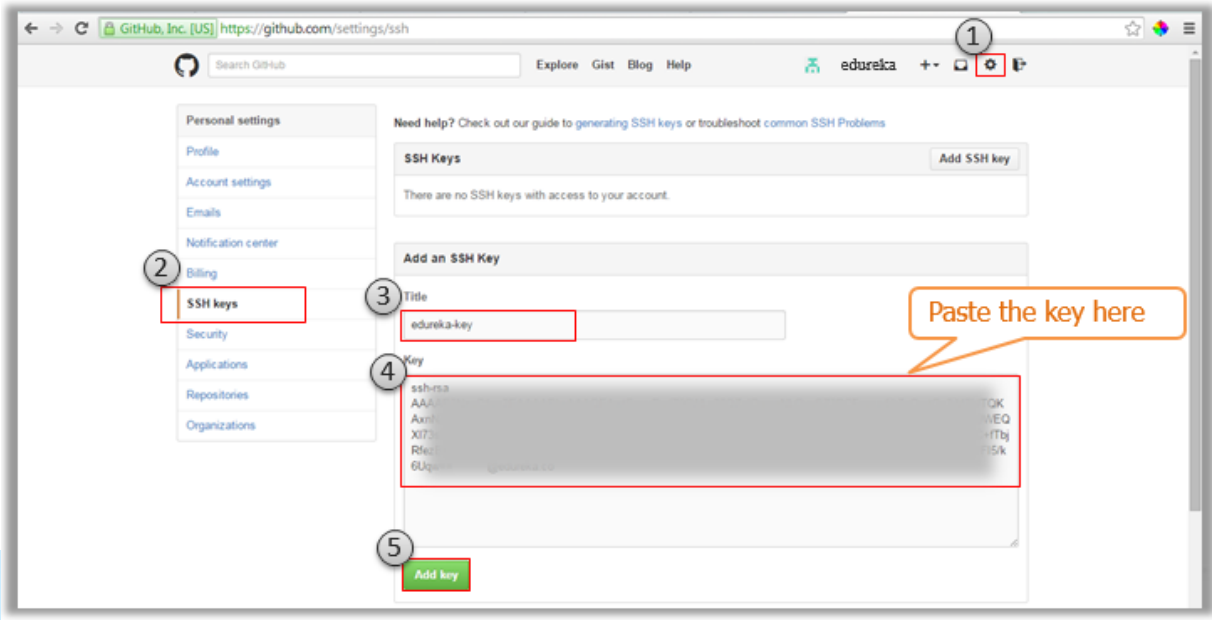
```
edureka@localhost:~
File  Edit  View  Search  Terminal  Help
[edureka@localhost ~]$ cat ~/.ssh/id_rsa.pub
ssh-rsa [illegible] 4jh7
gOmtGc [illegible] tJ42
XeJheo [illegible] +FYW
lC23Xp [illegible] TmNu
kuxq33 [illegible] admin@edureka.co
```

Now, to configure the Git account,

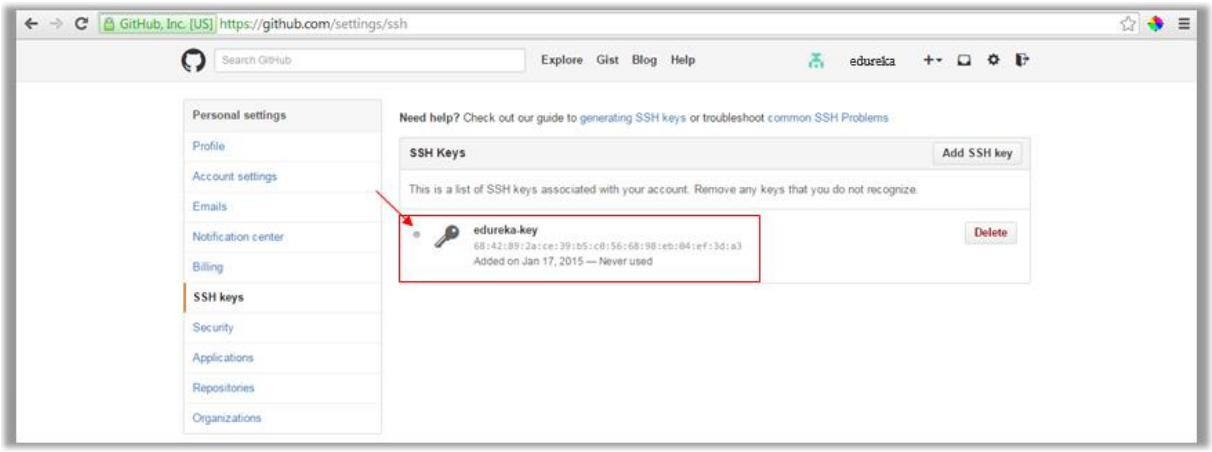
Go to your Git Account and Select Settings icon in the Top right

Then, select SSH keys option from the left menu.

Give the Title for the key, paste the key in the key option and then click on Add key



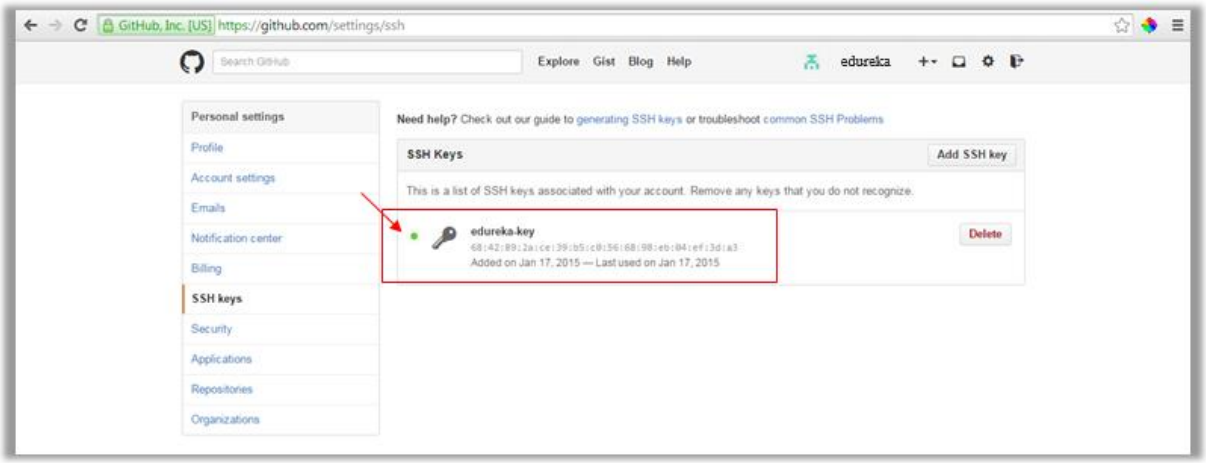
The key has been successfully added but is unused now



Execute the following command to use the key.

```
edureka@localhost:~
File  Edit  View  Search  Terminal  Help
[edureka@localhost ~]$ ssh -T git@github.com
The authenticity of host 'github.com (192.30.252.131)' can't be established.
RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,192.30.252.131' (RSA) to the list of know
n hosts.
Hi      edureka   ! You've successfully authenticated, but GitHub does not provid
e shell access.
```

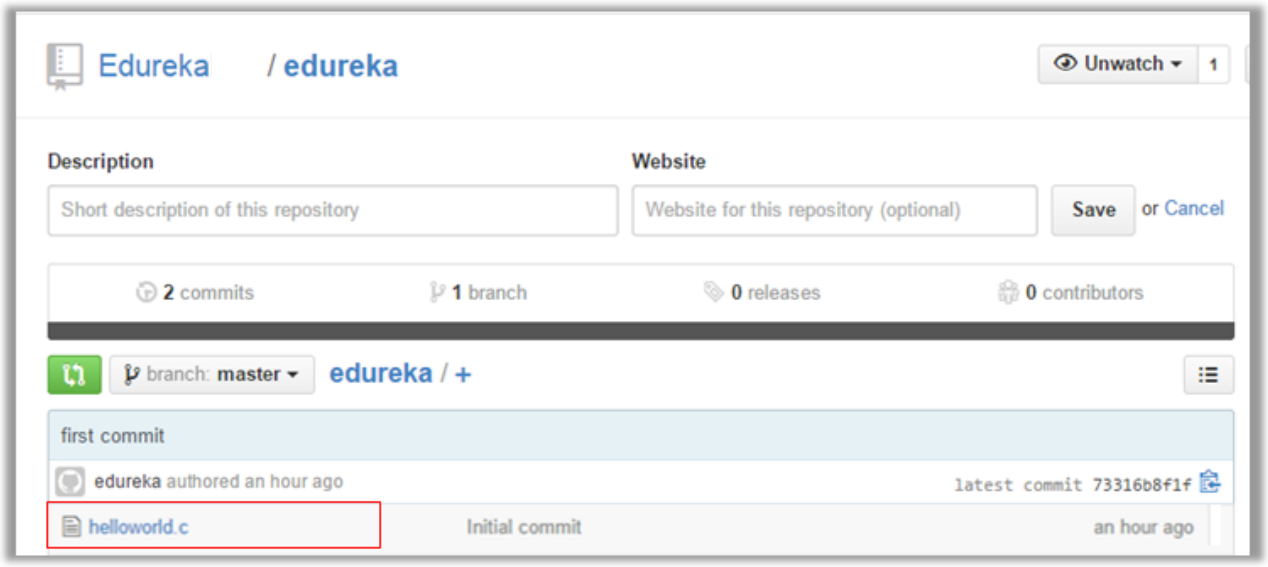
Now, we can see that the status of the key has changed to green after being used once



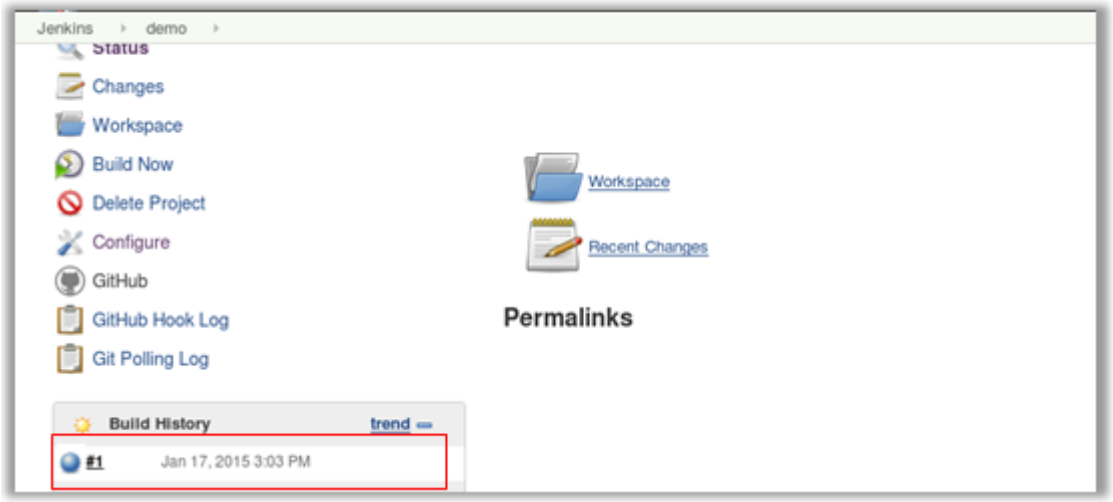
Now we can push our code the master. Execute the following command

```
edureka@localhost:~/cont_int_demo/.git
File  Edit  View  Search  Terminal  Help
[edureka@localhost .git]$ git push -u origin master
Counting objects: 6, done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 516 bytes, done.
Total 6 (delta 0), reused 0 (delta 0)
To git@github.com:EdurekaLab/edureka.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
```

We could see that the code has been pushed



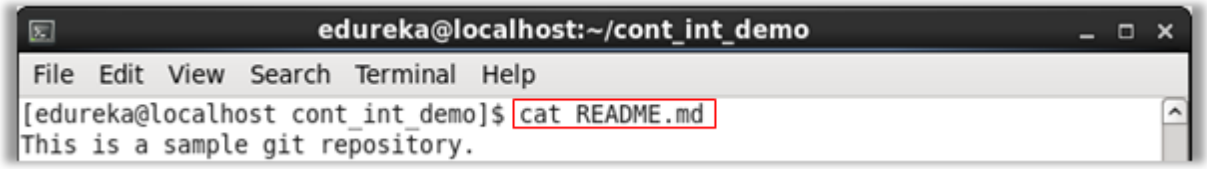
Jenkins complies the code and automatically creates a build. Below is the screen of the 1st build created



When we click on the build, we can check the details

Timestamp, error message if any, etc. all these details can be checked here

Now, let's add a new README.md file and then see how Jenkins works



Now, before pushing the new changes to the repository, we need to pull the changes from there

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git pull
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 2), reused 3 (delta 2)
Unpacking objects: 100% (6/6), done.
From github.com:EdurekaLab/edureka1
   885ffa2..1209e1c  master     -> origin/master
Updating 885ffa2..1209e1c
error: Your local changes to 'README.md' would be overwritten by merge.  Abortin
g.
Please, commit your changes or stash them before you can merge.
```

Now, let's add the README.md file

```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git add README.md
```

Now, let's commit the changes. Execute following command for the same

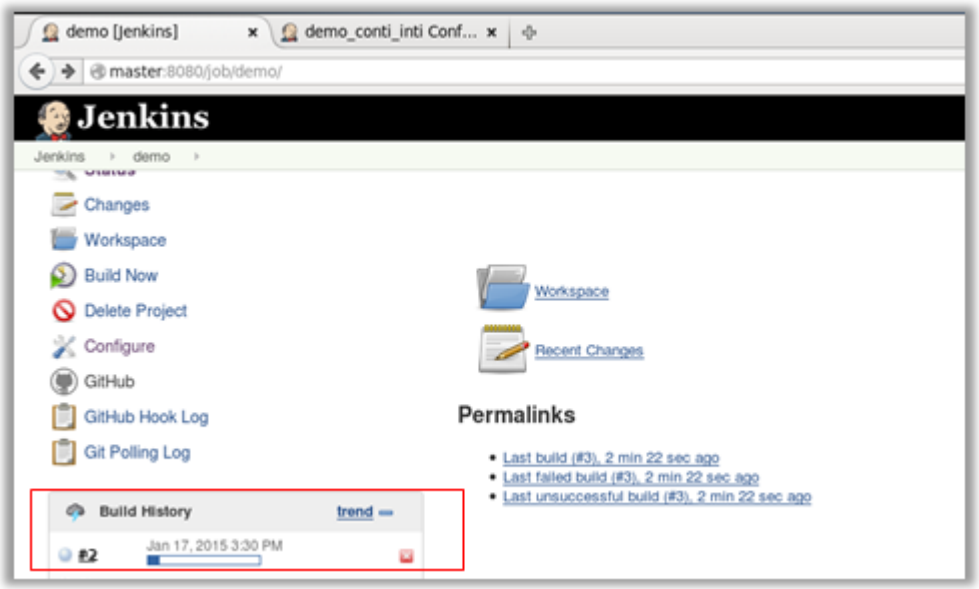
```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git commit -m "updated commit"
[master d1ed4bb] updated commit
 Committer: edureka <edureka@localhost.localdomain>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

If the identity used for this commit is wrong, you can fix it with:

    git commit --amend --author='Your Name <you@example.com>'

 1 files changed, 1 insertions(+), 0 deletions(-)
```

Now, we can push the changes. Execute following command for the same

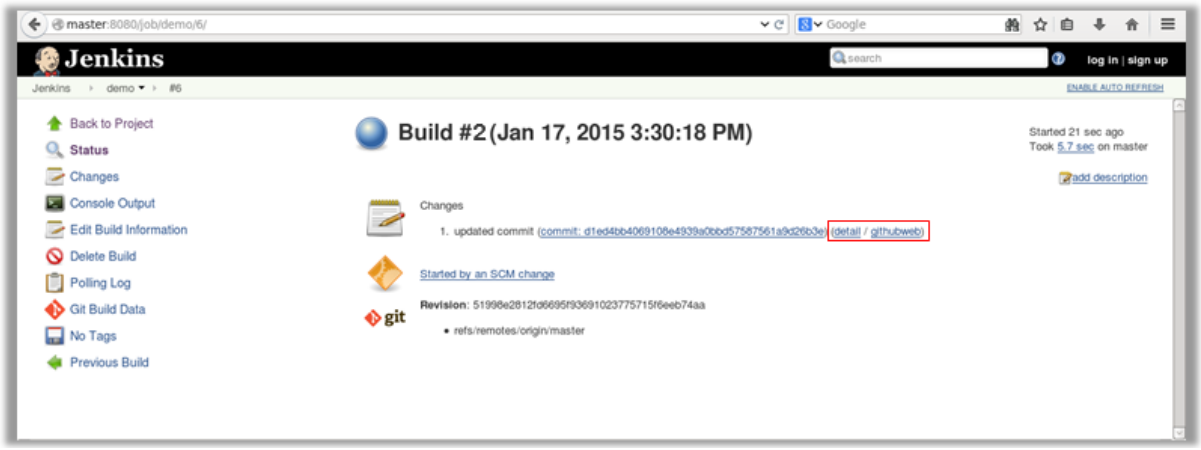
```
edureka@localhost:~/cont_int_demo
File  Edit  View  Search  Terminal  Help
[edureka@localhost cont_int_demo]$ git push -u origin master
Counting objects: 8, done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 536 bytes, done.
Total 5 (delta 1), reused 0 (delta 0)
To git@github.com:EdurekaLab/edureka1.git
   1209e1c..51998e2  master -> master
Branch master set up to track remote branch master from origin.
```

Jenkins has been configured to check for changes after every 3 minutes

As soon as the new code is pushed, Jenkins automatically compiles the code and creates a build



Build Successfully Created



We can check for the changes made in the new build. Select the build and then changes option from the left menu