Certification Project – I

Solution

Problem Statement

AppleBite Co. is using Cloud for one of their products. The project uses modular components, multiple frameworks and want the components to be developed by different teams or by 3rd-party vendors.

The company's goal is to deliver the product updates frequently to production with High quality & Reliability. They also want to accelerate software delivery speed, quality and reduce feedback time between developers and testers.

As development progressed, they are facing multiple problems, because of various technologies involved in the project. Following are the problems:

- Building Complex builds is difficult
- Manual efforts to test various components/modules of the project
- Incremental builds are difficult to manage, test and deploy

To solve these problems, they need to implement Continuous Integration & Continuous Deployment with DevOps using following tools:

Git – For version control for tracking changes in the code files

Jenkins – For continuous integration and continuous deployment

Docker – For deploying containerized applications

Puppet/Ansible - Configuration management tools

Selenium - For automating tests on the deployed web application

This project will be about how to do deploy code to dev/stage/prod etc, just on a click of button.

Link for the sample PHP application: https://github.com/edureka-devops/projCert.git

Business challenge/requirement

As soon as the developer pushes the updated code on the GIT master branch, a new test server should be provisioned with all the required software. Post this, the code should be containerized and deployed on the test server.

The deployment should then be tested using a test automation tool, and if the build is successful, it should be pushed to the prod server.

All this should happen automatically and should be triggered from a push to the GitHub master branch.

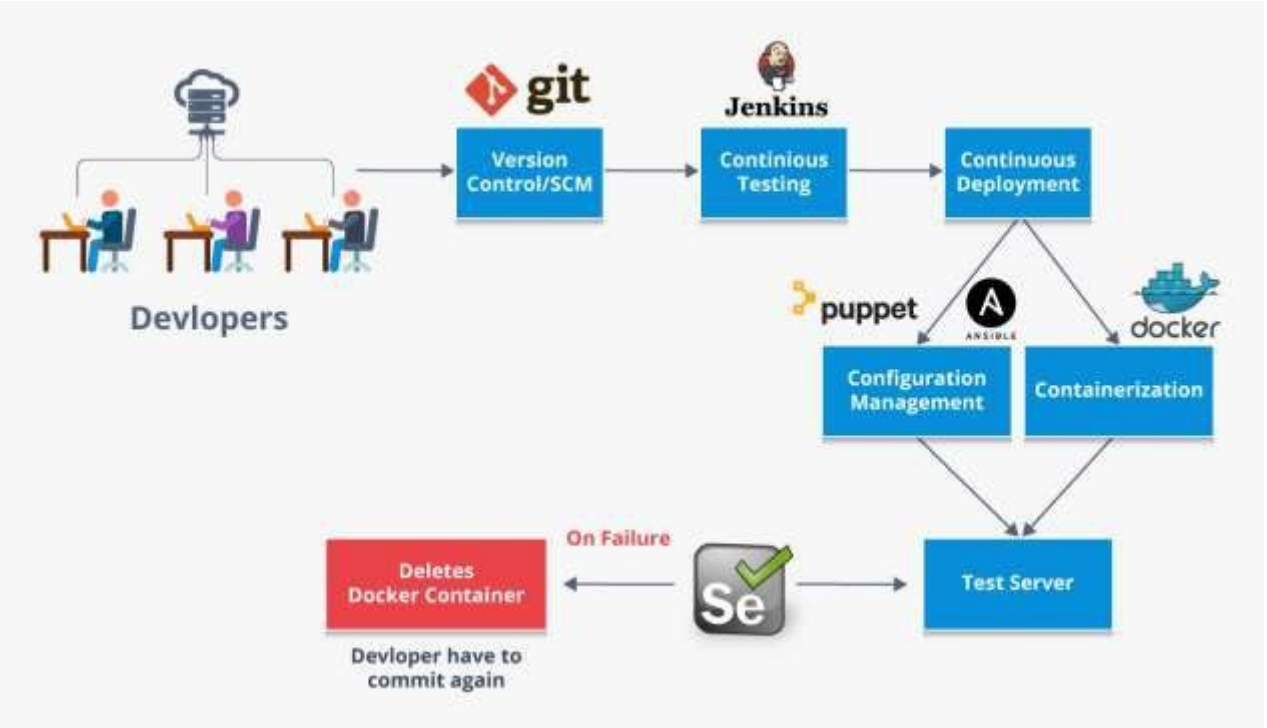
Steps for executing the solution:

- Use the Master VM for Jenkins, Ansible, Puppet, GIT etc.
- Use the Clean Ubuntu VM image provided in the "Edureka Setup Guide" for Jenkins Slave Node (Test Server)
- Change the IP address of the VMs accordingly
- Add Build Pipeline Plugin and Post-build task plugin to Jenkins on the master VM
- Install python, openssh-server and git on the slave node manually
- Set up the necessary tools such as git, chromedriver(selenium), chromium browser(selenium) on the slave node through Ansible
- Use the image devopsedu/webapp and add your PHP website to it using a Dockerfile
- Create a Selenium Test for your PHP website. It should click on "About" and verify the text written in it. This will conclude the website is deployed and is running fine.
- Push the PHP website, Dockerfile and Selenium JAR to a git repository

Below tasks should be automated through Jenkins by creating a pipeline:

1. Install and configure puppet agent on the slave node (Job 1)
2. Sign the puppet certificate on master using Jenkins (Job 2)
3. Trigger the puppet agent on test server to install docker (Job 3)
4. Pull the PHP website, Dockerfile and Selenium JAR from your git repo and build and deploy your PHP docker container. After this test the deployment using Selenium JAR file. (Job 4)
5. If Job 4 fails, delete the running container on Test Server

NOTE: Jenkins may show Job 3 as Failed, even though Console Output is successful. Enable the next job to run even if this job fails, as a workaround.

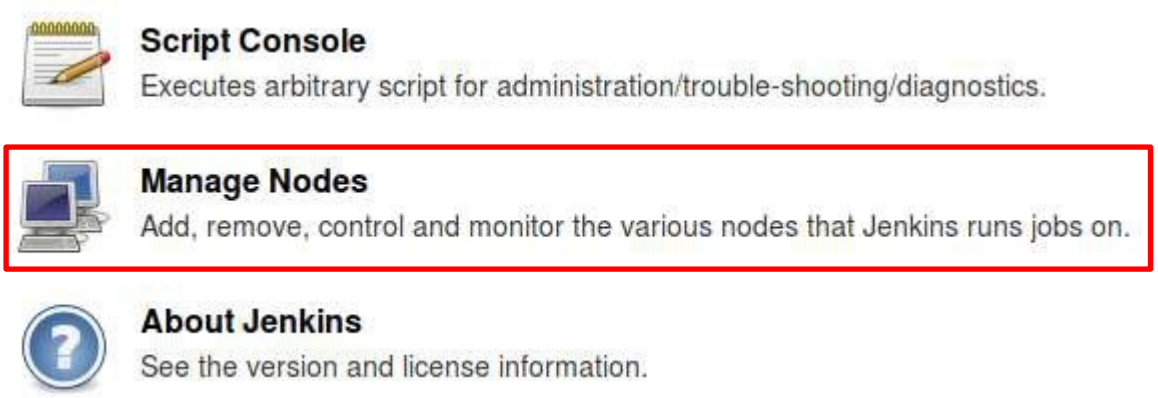


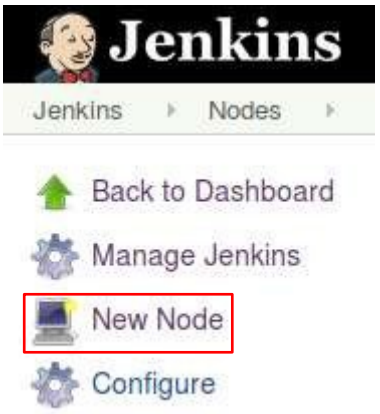
Solution

Setting up Remote Node on Jenkins

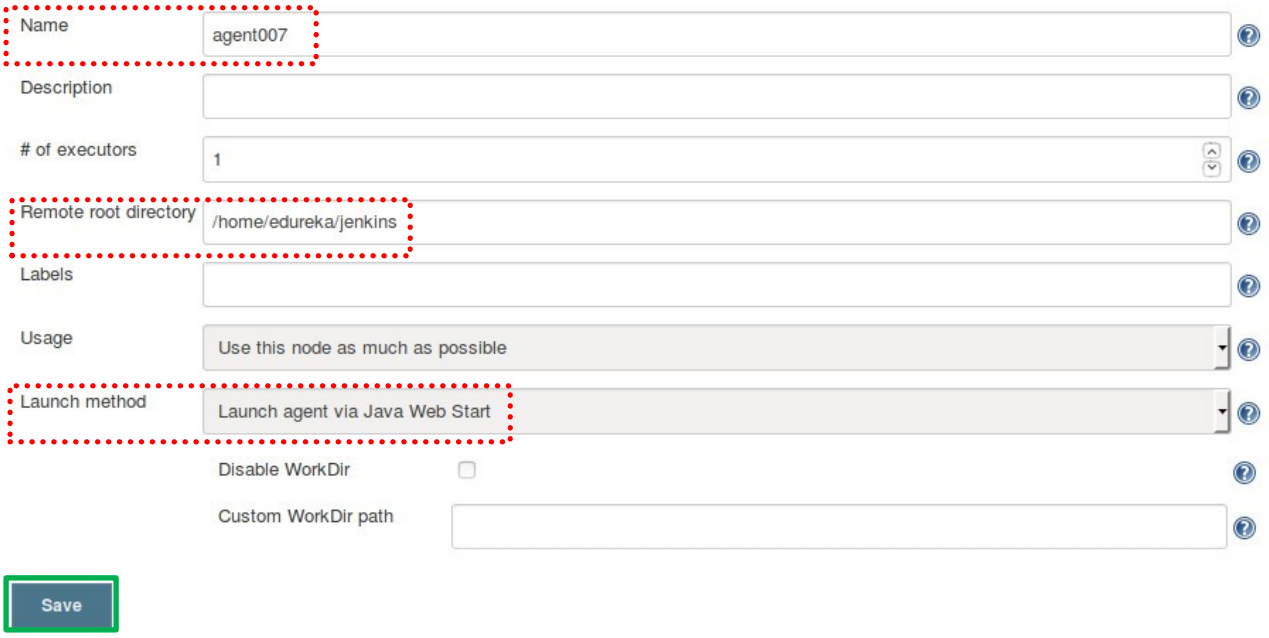
1. Go to Manage Jenkins-> Configure Global Security and under Agents set the TCP port for JNLP agents to Random

2. Go to Manage Jenkins -> Manage Nodes and create a new node



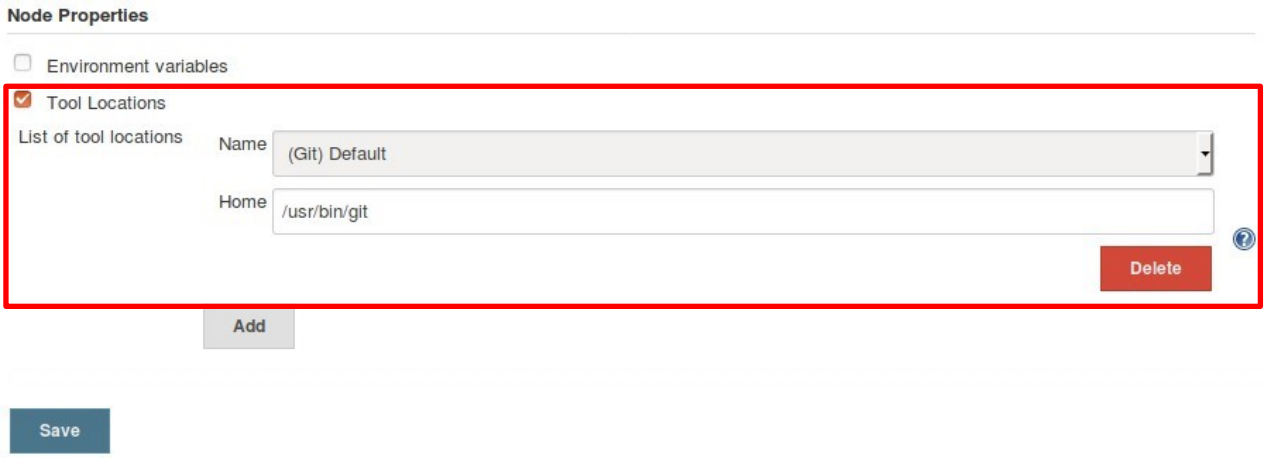


3. Now, set the “name”, “Remote root directory” and choose Launch agent via Java Web Start in Launch Method



And under Node Properties check Tool Locations and give path to git directory and click on

save



4. Now Select the newly created node and download the agent.jar file in your master node

5. Copy the above highlighted code in your agent node's terminal and replace the localhost with Master's IP address

Configuring Remote machine using Ansible

6. Create an ansible playbook to configure the remote machine for the initial setup

```
---
- hosts: agent
  become: true
  vars:
   ansible_become_pass: edureka
  tasks:
  - name: Install Git
    package:
     name: git
     state: present

  - name: Run update
    apt:
     update_cache: true

  - name: Install jdk
    package:
     name: default-jdk
     state: present

  - name: Copy chromedriver
    copy:
     src: /home/edureka/chromedriver
     dest: /home/edureka/

  - name: Install chromium browser
    package:
     name: chromium-browser
     state: present
```

```
- name: Install chromium driver
  package:
   name: chromium-chromedriver
   state: present

- name: Copy agent.jar file
  copy:
   src: /home/edureka/Downloads/agent.jar
   dest: /home/edureka

- name: Run update
  apt:
   update_cache: yes
```

Creating Puppet Module to set up Docker on remote machine:

7. Generate a Puppet Module for docker

```
root@master:/home/edureka/Documents# /opt/puppetlabs/bin/puppet module generate edu-docker
We need to create a metadata.json file for this module.  Please answer the
following questions; if the question is not applicable to this module, feel free
to leave it blank.

Puppet uses Semantic Versioning (semver.org) to version modules.
What version is this module?  [0.1.0]
```

8. Create an Install.pp Manifest to set up docker

```
       class docker::install {


              package {'curl':
                      ensure  => present,
              }


              exec {'apt-update':
                      command => '/usr/bin/apt-get update'
              }


              exec {'download_docker_key':
                      command => '/usr/bin/curl -fsSL
       https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -'
              }


       exec {'add_docker_repo':
                command => '/usr/bin/add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"',
                require => Exec['apt-update']
       }


       exec {'docker_cache':
                command => '/usr/bin/apt-cache policy docker-ce'
       }


       exec {'install_docker':
                command => '/usr/bin/apt-get install -y docker-ce'
       }


}
```

9. Call this manifest inside init.pp file in the manifests directory

```
class docker {
        class {'docker::install':}
}
```

10. Now Build and Install this Puppet Module

```
root@master:/home/edureka/Documents# /opt/puppetlabs/bin/puppet module build docker
Notice: Building /home/edureka/Documents/docker for release
Module built: /home/edureka/Documents/docker/pkg/edu-docker-0.1.0.tar.gz
root@master:/home/edureka/Documents# /opt/puppetlabs/bin/puppet module install /home/edureka/Document
s/docker/pkg/edu-docker-0.1.0.tar.gz
Notice: Preparing to install into /etc/puppetlabs/code/environments/production/modules ...
Notice: Downloading from https://forgeapi.puppet.com ...
Notice: Installing -- do not interrupt ...
/etc/puppetlabs/code/environments/production/modules
└─┬ edu-docker (v0.1.0)
  └── puppetlabs-stdlib (v4.25.1)
```

11. Call this Module inside the main Puppet manifest(i.e. site.pp)

```
class {'docker':}
```

12. Create a docker file to run the php application

```
-FROM devopsedu/webapp

-

-ADD proj /var/www/html

-

-RUN rm /var/www/html/index.html

-

-CMD apachectl -D FOREGROUND
```

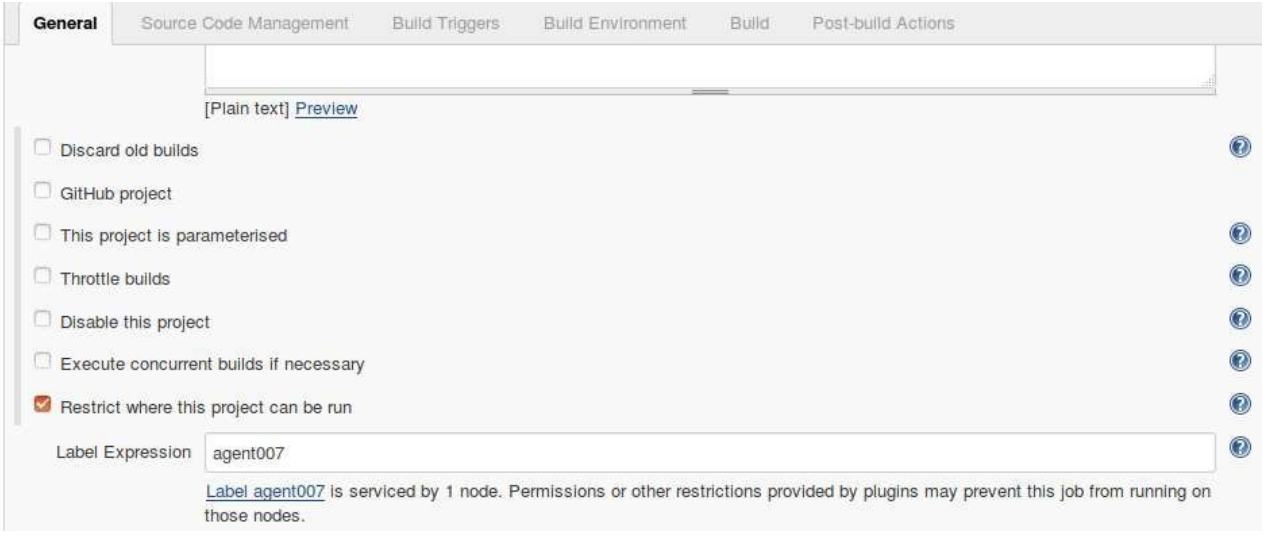
13. Write the following Selenium code and create a .jar file after compiling the code in eclipse(or any other java IDE

```
import static org.testng.Assert.assertEquals;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;
import org.openqa.selenium.chrome.ChromeOptions;

public class App
{
@Test

public static void main(String[] args) {


System.setProperty("webdriver.chrome.driver","/home/edureka/chromedriver");
ChromeOptions chromeOptions = new ChromeOptions();
chromeOptions.addArguments("--headless");
chromeOptions.addArguments("--no-sandbox");
WebDriver driver = new ChromeDriver(chromeOptions);
chromeOptions.addArguments("--headless");



driver.get("http://localhost:8081");

driver.manage().timeouts().implicitlyWait(3, TimeUnit.SECONDS);
driver.findElement(By.id("About Us")).click();

String test = driver.findElement(By.id("PID-ab2-pg")).getText(); assertEquals(test,
"This is about page. Lorem Ipsum is simply dummy text of the printing and typesetting
industry. Lorem Ipsum has been the industry's standard dummy text ever since the
1500s, when an unknown printer took a galley of type and
scrambled it to make a type specimen book. It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged. It
was popularised in the 1960s with the release of Letraset sheets containing Lorem
Ipsum passages, and more recently with desktop publishing software like Aldus
PageMaker including versions of Lorem Ipsum.");
System.out.println("Test Succeeded!!");
driver.quit();



}
}
```

Creating Jenkins Jobs:

Job to set up puppet agent on Remote Machine:

14. Create a new Job on Jenkins and restrict it to run on Remote machine only



15. Now, In the build phase add a build step to execute shell commands and save the Job

```
wget https://apt.puppetlabs.com/puppetlabs-release-pc1-xenial.deb
sudo dpkg -i puppetlabs-release-pc1-xenial.deb
sudo apt-get update
sudo apt-get install -y puppet-agent
sudo systemctl start puppet
sudo systemctl start puppet
```

Job to Sign Puppet certificates of the newly connected nodes to puppet

16. Create a new Job and add a build step to execute shell command on the local master machine

```
sudo /opt/puppetlabs/bin/puppet cert sign --all
```

Job to trigger puppet agent to pull and apply the catalog from master

17. Create a new job and restrict it to execute only on the remote machine

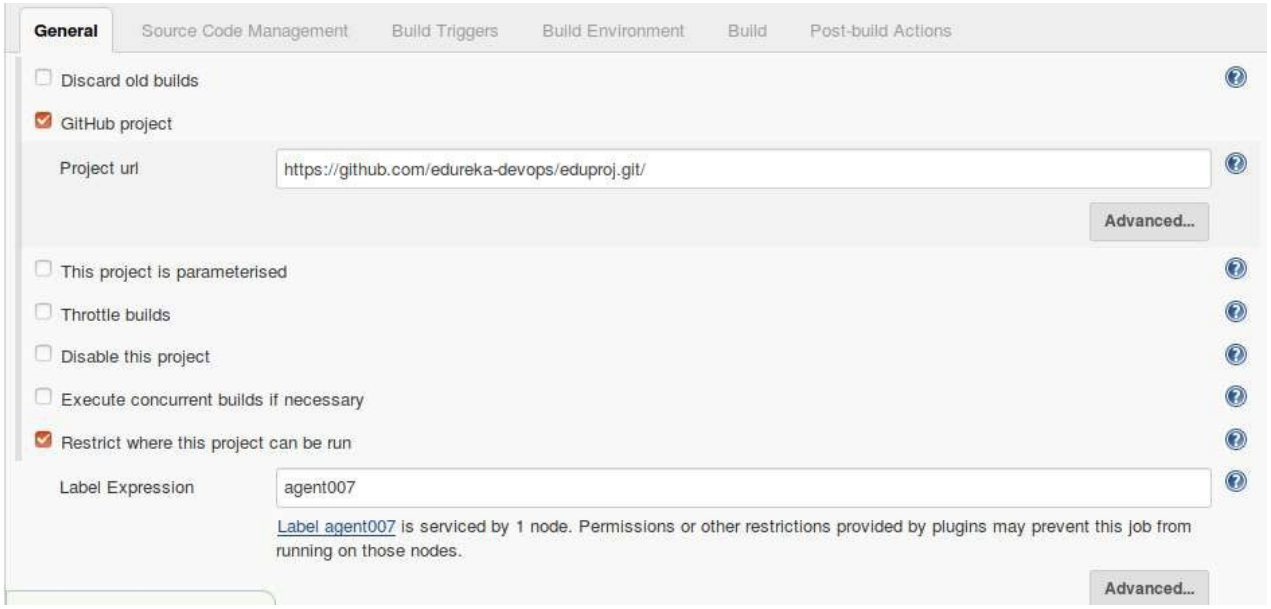
18. Add a build step to execute a shell command to trigger Puppet agent

```
sudo /opt/puppetlabs/bin/puppet agent -t
```

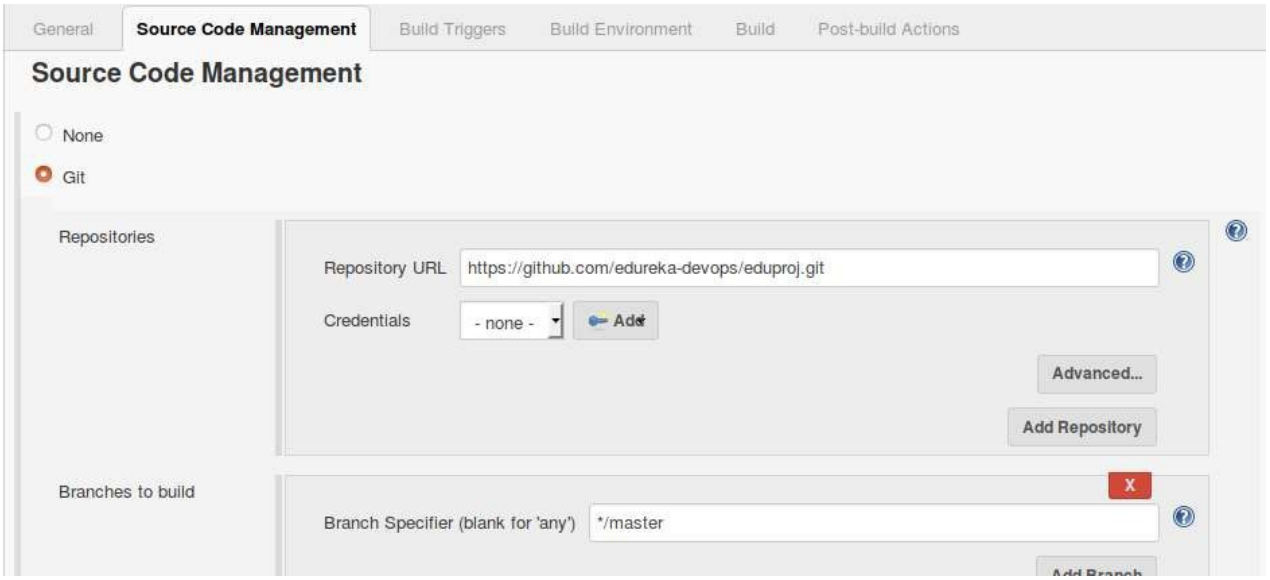
Job to create a docker container, run the container and execute Selenium test case on it

19. Create a new job and restrict it to execute only on the remote machine

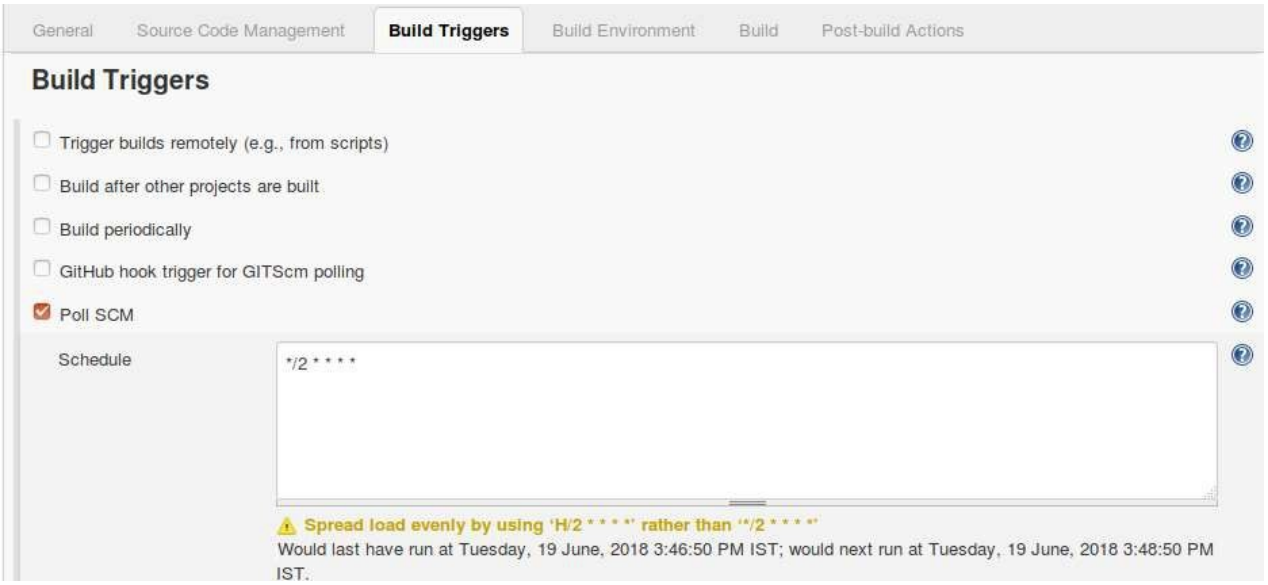
20. Add Your GitHub link in the GitHub Project and also make this job restricted to remote machine only



21. Under Source Code Management in the git section again add your GitHub url



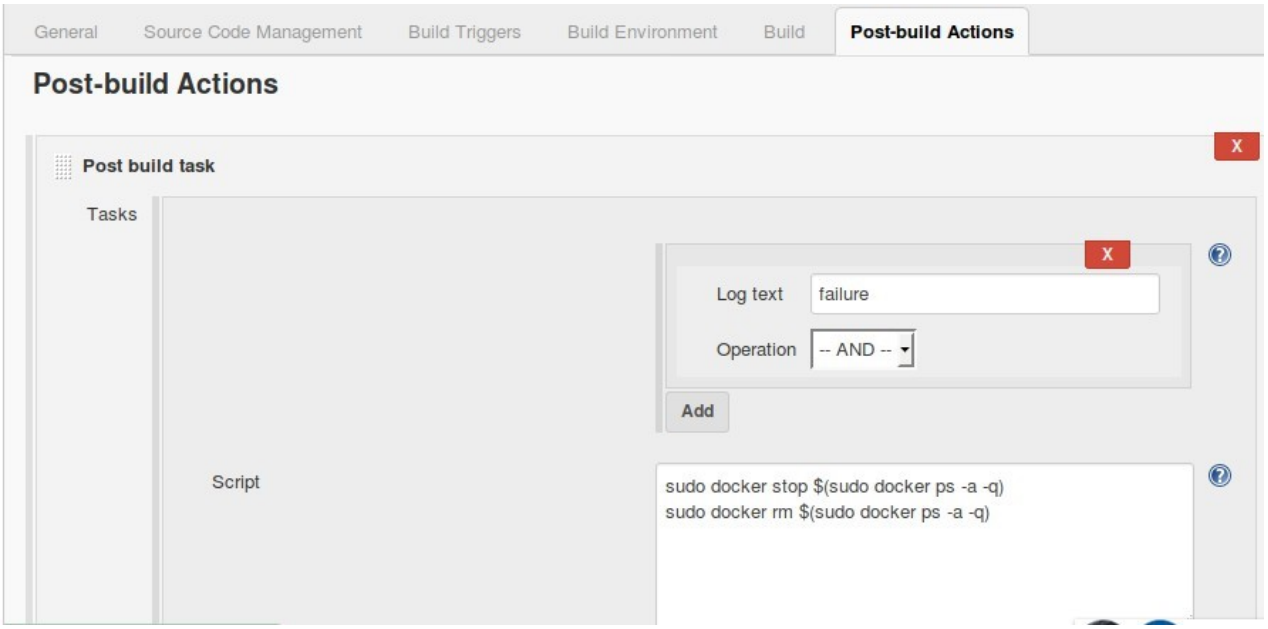
22. Under Build Triggers check poll SCM and give it appropriate time interval to poll GiHiub for changes



23. Now, add a build step to execute the following shell commands on the remote machine

```
sudo docker run busybox
sudo docker stop $(sudo docker ps -a -q)
sudo docker rm $(sudo docker ps -a -q)
sudo docker build . -t appmain:latest
sudo docker run -it -p 8081:80 -d appmain:latest
sudo java -jar final11.jar
```

24. Add a Post-build Actions -> Post build task to ensure that if the Selenium test fails, the docker containers are deleted. Here keep the Log text as failure and add the docker container removing commands in the script section



Build a Jenkins Delivery Pipeline to run all the jobs:

Note: While creating the Puppetagent trigger job Set the Post Build Action to Trigger even if the build fails. This is because for some reason Jenkins is considering Puppet agent Successful build to be a failure.

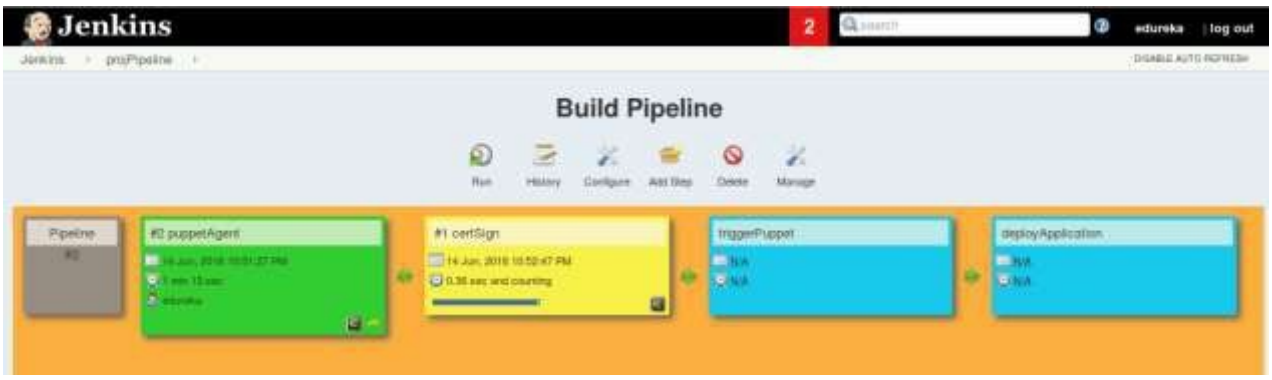

```
Building remotely on agent007 in workspace /home/edureka/jenkins/workspace/triggerPuppet
[triggerPuppet] $ /bin/sh -xe /tmp/jenkins8243335251037546 76.sh
+ sudo /opt/puppetlabs/bin/puppet agent -t
[0;32mInfo: Using configured environment 'production'[0m
[0;32mInfo: Retrieving pluginfacts[0m
[0;32mInfo: Retrieving plugin[0m
[0;32mInfo: Loading facts[0m
[0;32mInfo: Caching catalog for agent[0m
[0;32mInfo: Applying configuration version '1528997688'[0m
[mNotice: /Stage[main]/Docker::Install/Exec[apt-update]/returns: executed successfully[0m
[mNotice: /Stage[main]/Docker::Install/Exec[download_docker_key]/returns: executed successfully[0m
[mNotice: /Stage[main]/Docker::Install/Exec[add_docker_repo]/returns: executed successfully[0m
[mNotice: /Stage[main]/Docker::Install/Exec[docker_cache]/returns: executed successfully[0m
[mNotice: /Stage[main]/Docker::Install/Exec[install_docker]/returns: executed successfully[0m
[mNotice: Applied catalog in 22.63 seconds[0m
Build step 'Execute shell' marked build as failure
Triggering a new build of deployApplication
Finished: FAILURE
```

Execution

1. Execute the Ansible Playbook for initial setup of the agent node

```
edureka@master:/etc/ansible$ ansible-playbook nodeSetup.yml

PLAY [agent] ***********************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************
ok: [agent]

TASK [Install Git] *****************************************************************************************************
changed: [agent]

TASK [Run update] ******************************************************************************************************
changed: [agent]

TASK [Install jdk] *****************************************************************************************************
changed: [agent]

TASK [Copy chromedriver] ***********************************************************************************************
changed: [agent]

TASK [Install chromium browser] ****************************************************************************************
changed: [agent]

TASK [Install chromium driver] *****************************************************************************************
changed: [agent]

TASK [Copy agent.jar file] *********************************************************************************************
changed: [agent]

TASK [Run update] ******************************************************************************************************
changed: [agent]

TASK [Run update] ******************************************************************************************************
changed: [agent]

PLAY RECAP *************************************************************************************************************
agent                      : ok=9    changed=8    unreachable=0    failed=0
```

2. Execute the Jenkins Pipeline




Jenkins
edureka
log out
Build Pipeline
Run
History
Configure
Add Step
Delete
Manage
Pipeline
#2 puppetAgent
#1 certSign
#1 triggerPuppet
deployApplication


Jenkins
edureka
log out
Build Pipeline
Run
History
Configure
Add Step
Delete
Manage
Pipeline
#2 puppetAgent
#1 certSign
#3 triggerPuppet
#5 deployApplication


Jenkins
edureka
log out
Build Pipeline
Run
History
Configure
Add Step
Delete
Manage
Pipeline
#2 puppetAgent
#1 certSign
#3 triggerPuppet
#5 deployApplication

END