4.6 Setting Up a Jenkins Pipeline with Docker



This section will guide you to**:**

* Build a Docker Jenkins pipeline to implement CI/CD workflow.

This guide has three subsections, namely**:**

4.6.1 Installing Docker plugin and configuring Docker cloud

4.6.2 Configuring Jenkins job

4.6.3 Pushing the code to GitHub repositories

* *Docker is already installed in your lab. (Refer MEAN: Lab Guide - Phase 4)*

**Step 4.6.1:** Installing Docker plugin and configuring Docker cloud

* Open the terminal and type the following commands:

**wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -**

**echo deb https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list**

**sudo apt-get update**

**sudo apt-get install jenkins**

**sudo systemctl start jenkins**

**sudo systemctl status jenkins**

**sudo ufw allow OpenSSH**

**sudo ufw enable**

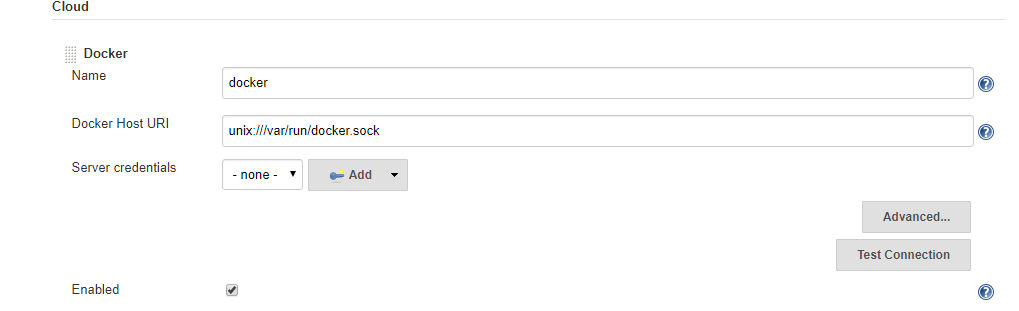
**sudo ufw allow 8080**

**sudo ufw status**

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

* Copy the password given by this command
* Type **sudo systemctl restart jenkins** and enter the copied password
* Go to the browser and type **localhost:8080**
* Copy paste the copied password
* Click on **Install Suggested Plugins**.The installation process will start.
* Add the required information in the username, full name, and email address fields
* Click on **Save and continue**
* Again click on **Save and Continue**
* Click on **Start using Jenkins**
* Add Docker cloud by accessing Manage Jenkins 🡪 Configure system. Then, you have to add Docker cloud details as shown below:

Docker Host URI: **unix:///var/run/docker.sock**



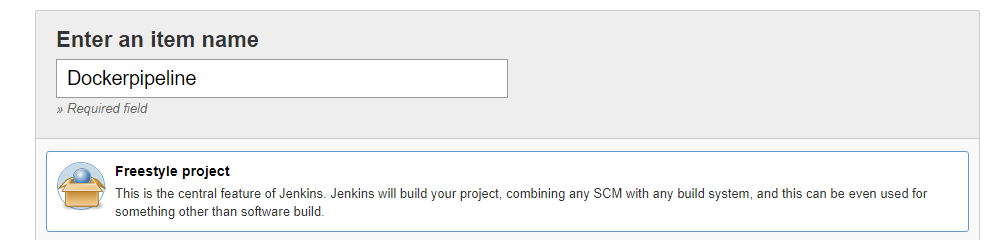
* Open the terminal and go to the /var/run path using **cd /var/run** command
* Configure the Docker cloud to give complete access to docker.sock file so that Jenkins will be able to connect to Docker process

**chmod 777 docker.sock**



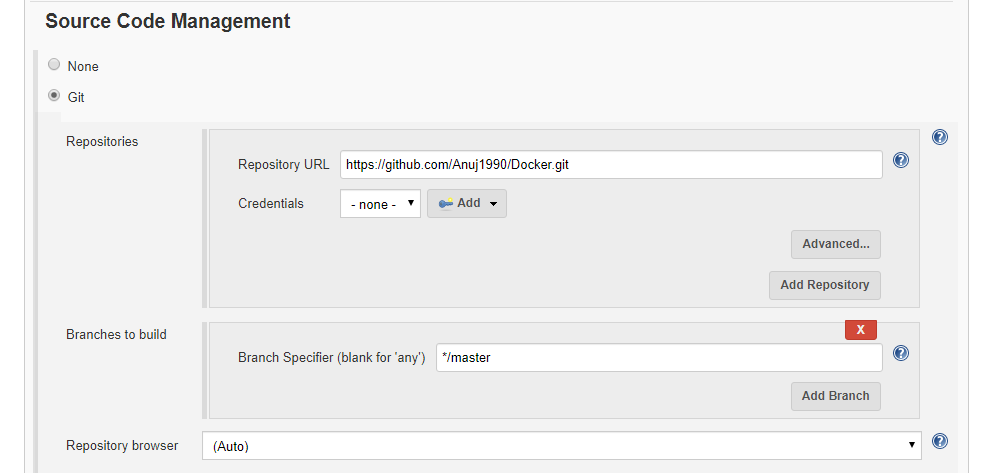
**Step 4.6.2:** Configuring Jenkins job

* Create a new Jenkins pipeline job for supporting CI/CD workflow

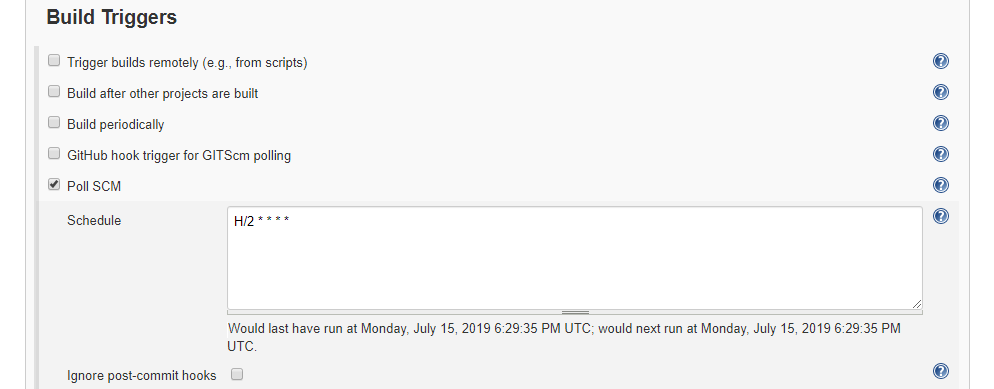


* Configure Git repository so that we can have Dockerfile to build Docker container and push it to Docker Hub

**https://github.com/Anuj1990/Docker.git**



* Configure build triggers to enable Poll SCM feature to detect if there is any push



* Configure Docker build option to configure build configurations. Some of the configurations are mentioned below:

Directory for Dockerfile . (Represents current location)

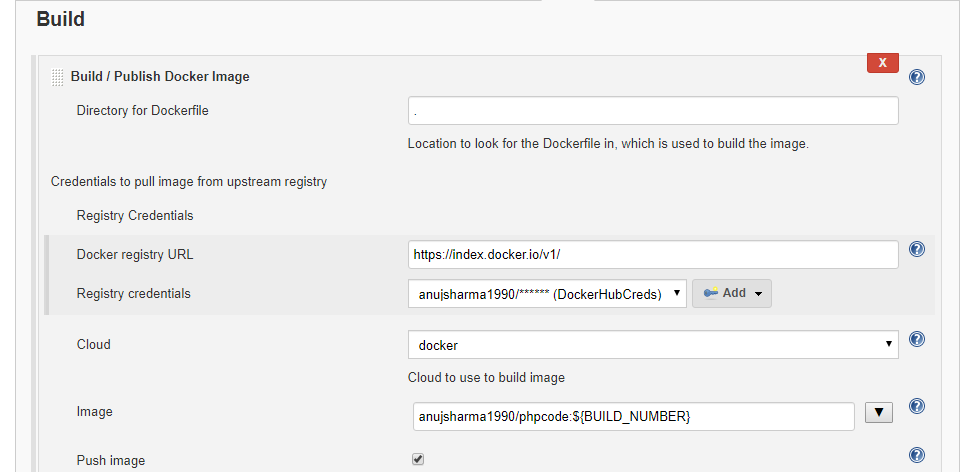
Docker Registry URL https://index.docker.io/v1/

Docker credentials Docker hub username password

Cloud Select Docker Cloud created in dropdown

Image anujsharma1990/phpcode:${BUILD\_NUMBER}

Registry Credentials Docker hub username password

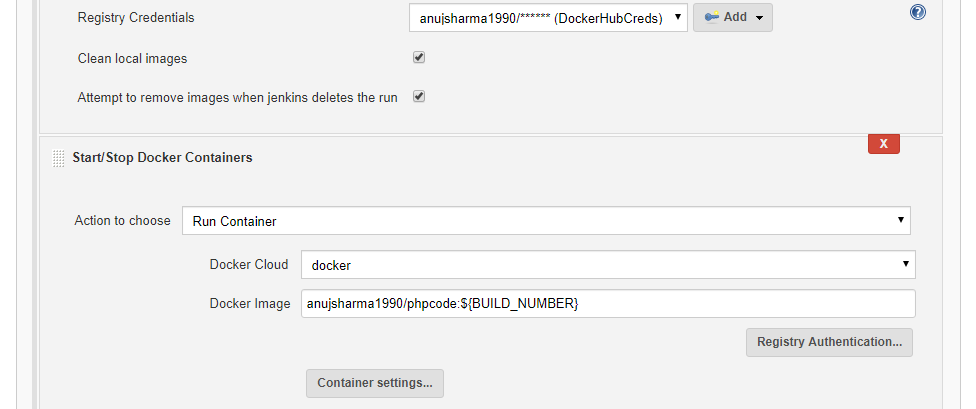


**Please Note:** We have configured how our custom Docker image will be pushed to Docker Hub. Now, we have to configure the Docker container deployment.

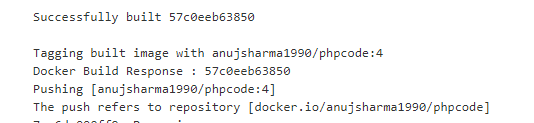
* Follow the steps mentioned below to configure how Docker containers will be deployed:

Docker cloud Select from dropdown Docker cloud details

Docker image anujsharma1990/phpcode:${BUILD\_NUMBER}

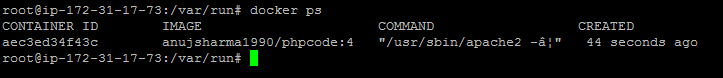


* Once job configuration is done, save the configuration and proceed with triggering build in order to build custom container and deploy the container





* Once the build is successful, validate the Docker container deployment on Docker host which will help us to implement complete CI/CD workflow for Docker container



**Step 4.6.3:** Pushing the code to GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add . 

Commit the changes using the following command:

git commit . -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master