2020

CI/CD Pipeline

JENKINS – DOCKER – SLACK

YARA AMR-ALLAH

Workflow Documentation:

1. Forked repository's link: https://github.com/YaraAmrallah/Booster_CI_CD_Project
2. Docker file to:
    a. Image from ubuntu: FROM ubuntu
    b. Install python 3.6: RUN apt-get -y install python3.6
    c. Install pip3: RUN apt-get -y install python3-pip3
    d. Copy the source code files of the app to the image: ADD . /simpleApp
    e. Set the configurations to start the Django server
        i. Install required packages: pip install -r requirements.txt
        ii. Make migrations for DB: python3.6 manage.py makemigrations
        iii. Apply the migrations: python3.6 manage.py migrate
        iv. Start the server: python3.6 manage.py runserver 0.0.0.0:8000

    Docker File:

```
FROM ubuntu
RUN apt-get update -qq
RUN apt-get -y install python3.6
RUN apt-get -y install python3-pip3
ADD . /simpleApp
WORKDIR /simpleApp
RUN pip install -r requirements.txt
RUN python3.6 manage.py makemigrations
RUN python3.6 manage.py migrate
EXPOSE 8000
CMD ["python3.6 ","manage.py ", "runserver ", "0.0.0.0:8000 "]
```

3. Configure container to act as a Jenkins slave:
    a. Create the docker image providing Jenkins and docker capabilities

    Docker File:

```
FROM ubuntu
USER root
RUN apt-get update -qq
# create jenkins directory and user
RUN mkdir jenkins_home
RUN chmod 777 jenkins_home
RUN useradd -ms /bin/bash jenkins
# install docker client and add jenkins to the docker group
RUN apt-get install -qqy apt-transport-https ca-certificates curl gnupg-agent software-properties-common
RUN curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -
RUN add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  focal \
  stable"
RUN apt-get update -qq
RUN apt-get install -y docker-ce docker-ce-cli containerd.io
RUN usermod -aG docker jenkins
# install Jenkins dependencies and SSH
RUN apt-get install openjdk-8-jdk -qq
RUN apt-get install openssh-server -qq
# switch user to jenkins and go to the jenkins working directory
USER jenkins
WORKDIR jenkins_home
```

b. Build docker image:

```
docker build -t "slave-image" .
```

c. On the host generate a public key via SSH:

   i. Generate the SSH key pair then press ENTER three times

```
ssh-keygen -t rsa -C "Jenkins slaves access key"
```

   ii. Keep the public key to place it inside the container

```
cat ~/.ssh/id_rsa.pub
```

d. Run docker container locally in interactive mode:

```
docker run -it --name slave-container -v /var/run/docker.sock:/var/run/docker.sock
```

e. While maintaining the user as ‘jenkins’, add the public key to the SSH authorized keys:

   i. Create a .ssh directory and cd into it

```
mkdir ~/.ssh
cd ~/.ssh
```

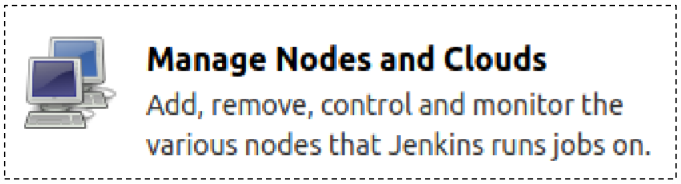
   ii. Add the public key to the authorized_keys file

```
echo "COPIED_PUBLIC_KEY" > ~/.ssh/authorized_keys
```

f. Go back to the host machine to copy the private key

```
cat ~/.ssh/id_rsa
```

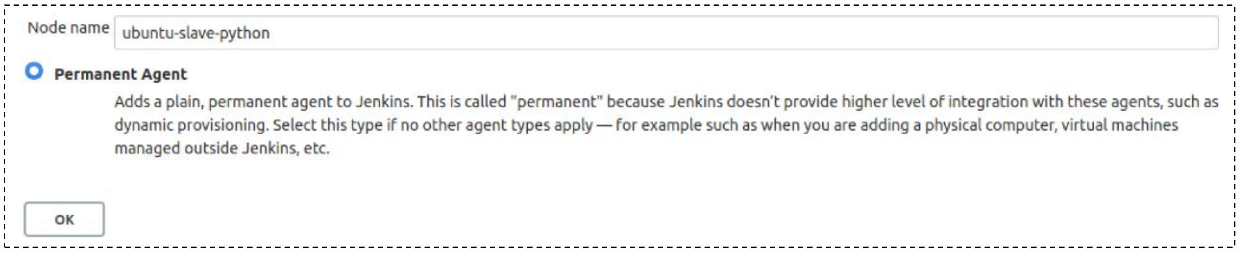
g. On the Jenkins dashboard go to “Manage Jenkins”, then go to “Manage Nodes and Clouds” under “System Configurations” section



Manage Nodes and Clouds

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

   i. Select “New Node”, add a name and press OK as follows



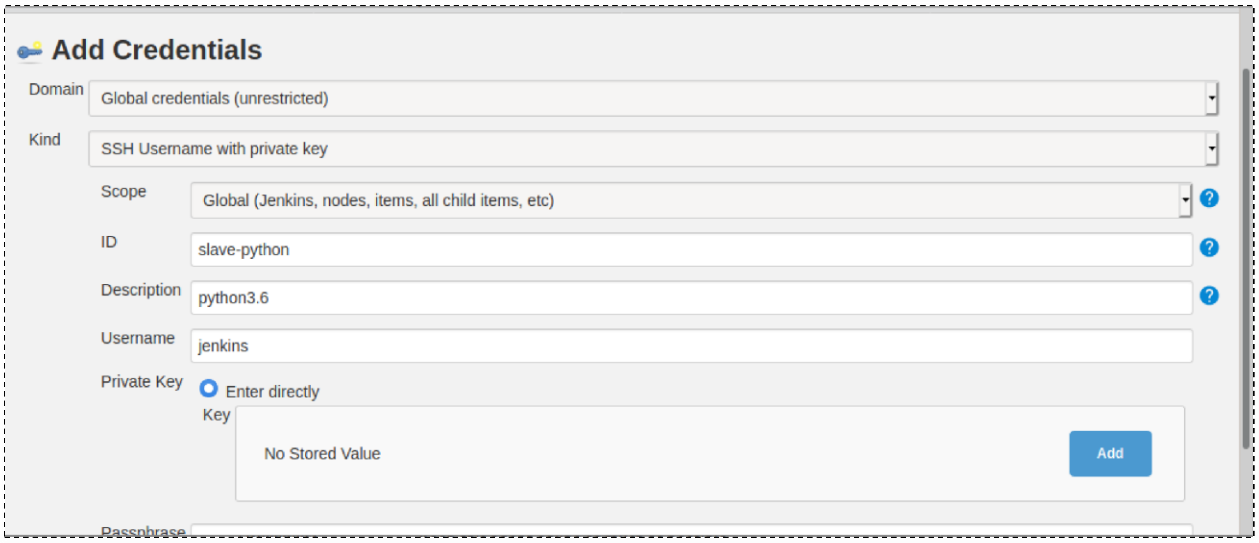
Node name: ubuntu-slave-python

● **Permanent Agent**

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

   ii. Give the suitable description and tags, then select the “launch method” as “Launch agents via SSH”

iii. Add SSH credentials, and paste the copied host's private key as follows



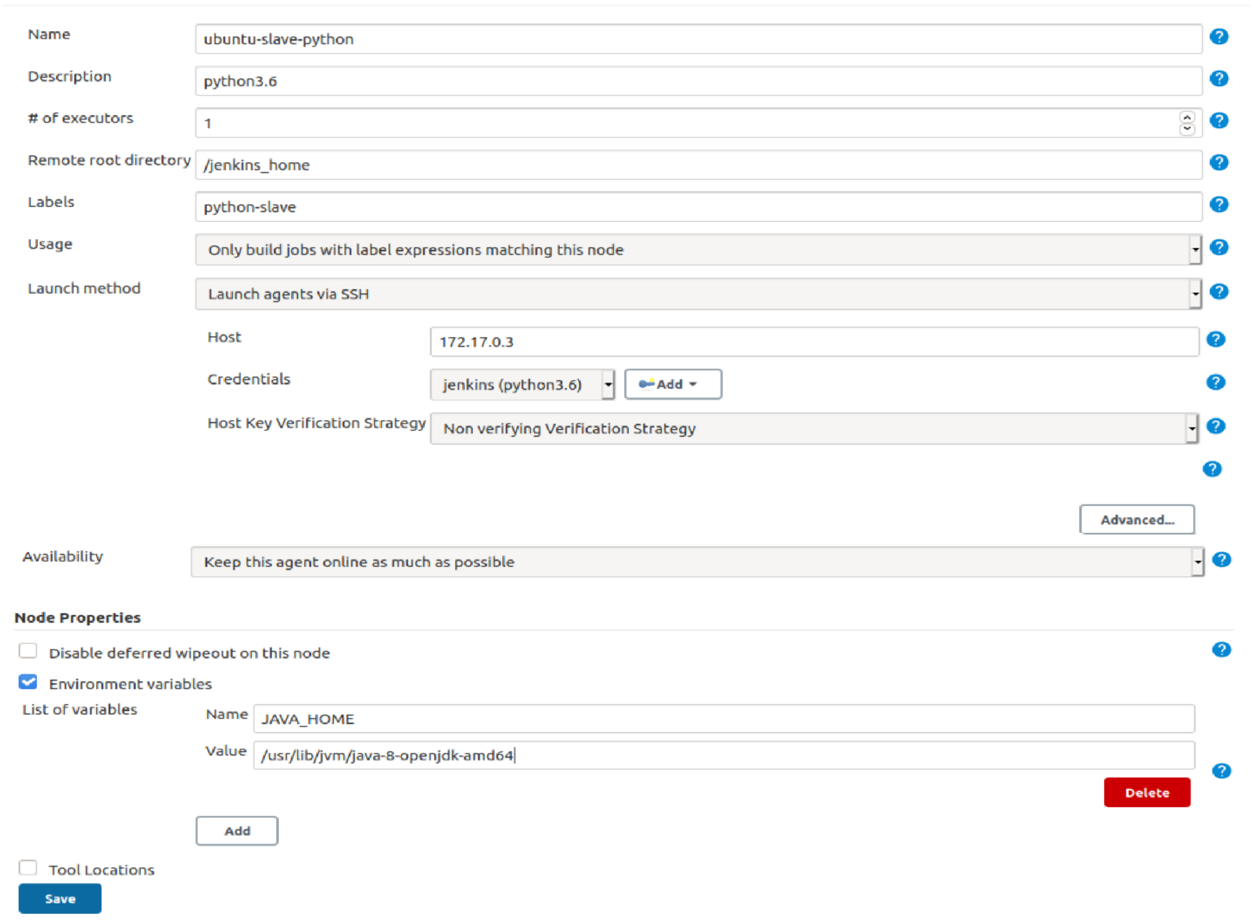
Add Credentials

Domain	Global credentials (unrestricted)
Kind	SSH Username with private key
Scope	Global (Jenkins, nodes, items, all child items, etc)
ID	slave-python
Description	python3.6
Username	jenkins
Private Key	● Enter directly
Key	No Stored Value
Passphrase	

iv. Get container's IP address from the host and paste it in the node's SSH settings

```
docker inspect -f "{{ .NetworkSettings.IPAddress }}" slave-container
```

v. Get the slave's JAVA_HOME environment variable path and add it to the node's configuration: /usr/lib/jvm/java-8-openjdk-amd64



Name	ubuntu-slave-python
Description	python3.6
# of executors	1
Remote root directory	/jenkins_home
Labels	python-slave
Usage	Only build jobs with label expressions matching this node
Launch method	Launch agents via SSH
Host	172.17.0.3
Credentials	jenkins (python3.6)
Host Key Verification Strategy	Non verifying Verification Strategy
Availability	Keep this agent online as much as possible

Node Properties

- ☐ Disable deferred wipeout on this node
- ☑ Environment variables

List of variables

Name	JAVA_HOME
Value	/usr/lib/jvm/java-8-openjdk-amd64

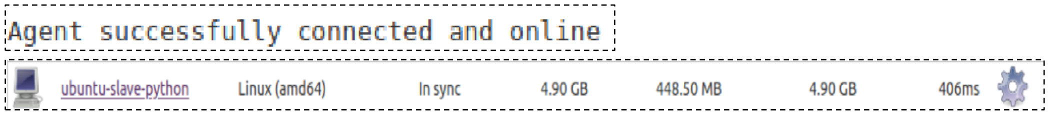
- ☐ Tool Locations

vi. Exit to/switch to the slave's root user and start the SHH service

```
# exit
docker exec -it -u root slave-container bash
service ssh start
```

vii. Save the node's configurations

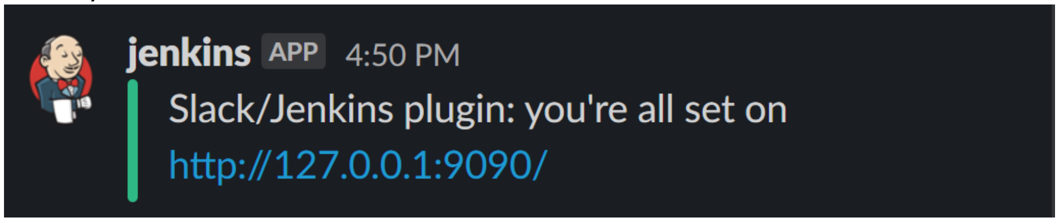
viii. Go back to the Agent's dashboard and make sure it has connected successfully



```
Agent successfully connected and online
```

ubuntu-slave-python	Linux (amd64)	In sync	4.90 GB	448.50 MB	4.90 GB	406ms

4. Slack configuration
   a. In the slack "App Directory" search for "Jenkins CI": YOUR_WORKSPACE_URLNAME.slack.com/apps
   b. Click on "Add to Slack"
   c. Assign a workspace channel to the jenkins user
   d. Follow the setup instructions on Jenkins and add your credentials (given authentication code) as a secret text
   e. Test your connection and make sure that slack is connected. You will find a similar message sent to your slack channel



jenkins APP 4:50 PM

Slack/Jenkins plugin: you're all set on http://127.0.0.1:9090/

   f. Use the following ink to know the slack functions and syntax options: https://www.jenkins.io/doc/pipeline/steps/slack/
      Example:
      slackSend(color: '#00FF00', message: "SUCCESSFUL: Job '${env.JOB_NAME} [${env.BUILD_NUMBER}]' (${env.BUILD_URL}console)")

5. Install Audit Log plugin to track builds and create statistics
   a. Manage Jenkins → Manage Plugins → Available → Download and Install "Audit Log"
   b. You can now track logs by clicking on "Audit Logs" on the left-hand side

6. Jenkins file with the following stages:
   a. Preparation: checkout the code
   b. Build image: build image using the docker file
   c. Push image: push the built image to docker hub
   d. Deploy: deploy a container from the pushed image
   e. Notification: send slack notification with the build status

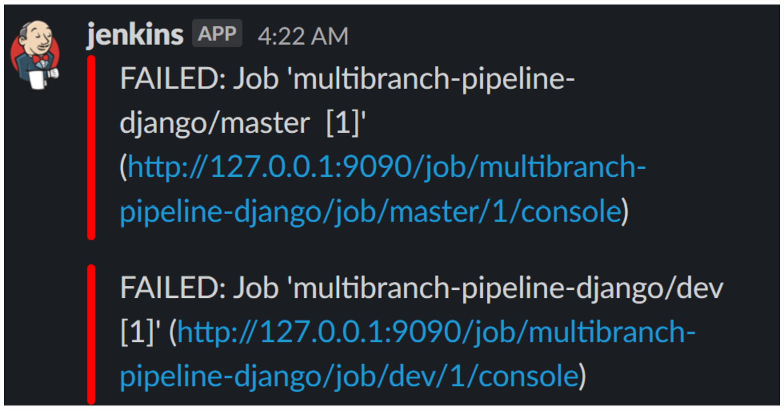
Jenkinsfile:

```
pipeline {
    agent {label 'python-slave'}
    stages {

        stage('preparation') {
            steps {
                git 'https://github.com/YaraAmrallah/Booster_CI_CD_Project'
            }
        }

        stage('build image') {
            steps {
              sh 'docker build -t yaraamrallah/django_app:v1.0 .'
            }
         }

        stage('push image') {
            steps {
withCredentials([usernamePassword(credentialsId:"docker",usernameVariable:"USERNAME",passwordVariable:"
PASSWORD")]){
              sh 'docker login --username $USERNAME --password $PASSWORD'
              sh 'docker push yaraamrallah/django_app:v1.0'
              }
            }
        }

        stage('deploy') {
          steps {
            sh 'docker run -d -p 8000:8000 yaraamrallah/django_app:v1.0'
        }
        }
    }

    post {
      success {
      slackSend (color: '#00FF00', message: "SUCCESSFUL: Job '${env.JOB_NAME}  [${env.BUILD_NUMBER}]'
(${env.BUILD_URL}console)")
      }

      failure {
      slackSend (color: '#FF0000', message: "FAILED: Job '${env.JOB_NAME}  [${env.BUILD_NUMBER}]'
(${env.BUILD_URL}console)")
      }

      aborted {
      slackSend (color: '#000000', message: "ABORTED: Job '${env.JOB_NAME}  [${env.BUILD_NUMBER}]'
(${env.BUILD_URL}console)")
      }
    }
}
```

7. Create a multibranch pipeline:
   a. Add the repository's link and include Jenkinsfile name

      N.B. When using the SCM option, the GitHub code will checkout automatically, it is better to remove the 'Preparation' step, so it will not happen twice

   b. The build kept failing, since after upgrading docker, the jenkins user cannot obtain docker permissions by any means



jenkins APP 4:22 AM

FAILED: Job 'multibranch-pipeline-django/master [1]' (http://127.0.0.1:9090/job/multibranch-pipeline-django/job/master/1/console)

FAILED: Job 'multibranch-pipeline-django/dev [1]' (http://127.0.0.1:9090/job/multibranch-pipeline-django/job/dev/1/console)

   c. I had to either:
      i. change "docker.sock" file's permission
      ii. create a user with the same UID and name of jenkins user on the host and add it to the docker group
   d. There were some dependencies issues which made me think of different approaches for building the docker file. At the end, both approaches worked successfully
      i. Master Branch Docker File:

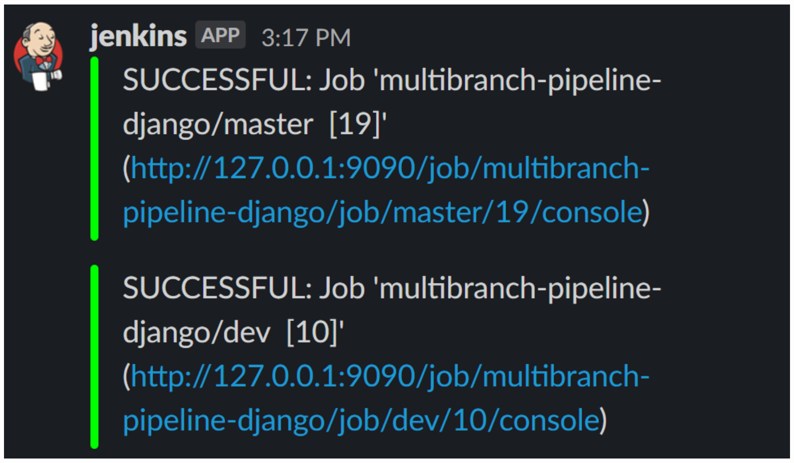
```
FROM python:3.6-buster
ADD . /simpleApp
WORKDIR /simpleApp
RUN pip install -r requirements.txt
RUN python3.6 manage.py makemigrations
RUN python3.6 manage.py migrate
COPY . .
EXPOSE 8000
CMD ["python3.6",  "manage.py ",  "runserver ",   "0.0.0.0:8000 "]
```

      ii. Dev Branch Docker File:

```
FROM ubuntu
RUN apt-get update -qq
RUN apt-get -y install python3.6
RUN apt-get -y install python3-pip3
ADD . /simpleApp
WORKDIR /simpleApp
RUN pip3 install -r requirements.txt
RUN python3.6 manage.py makemigrations
RUN python3.6 manage.py migrate
EXPOSE 8000
CMD ["python3.6 ","manage.py ", "runserver ", "0.0.0.0:8000 "]
```

e. Finally the build was successful

   i. Slack Notification

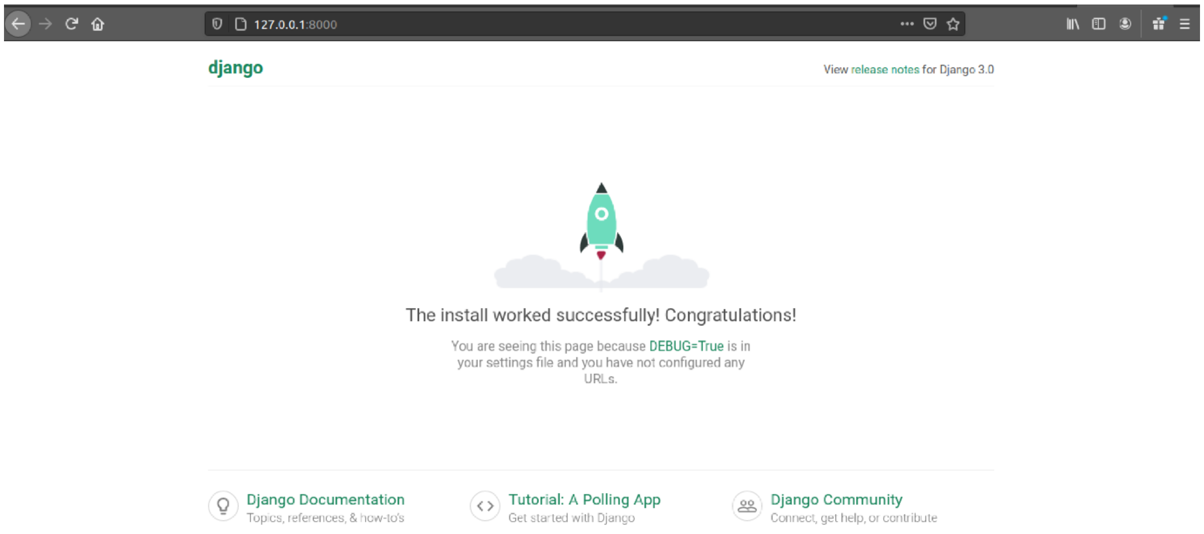


jenkins APP 3:17 PM

SUCCESSFUL: Job 'multibranch-pipeline-django/master [19]' (http://127.0.0.1:9090/job/multibranch-pipeline-django/job/master/19/console)

SUCCESSFUL: Job 'multibranch-pipeline-django/dev [10]' (http://127.0.0.1:9090/job/multibranch-pipeline-django/job/dev/10/console)

   ii. Master Branch output on port 8000



127.0.0.1:8000

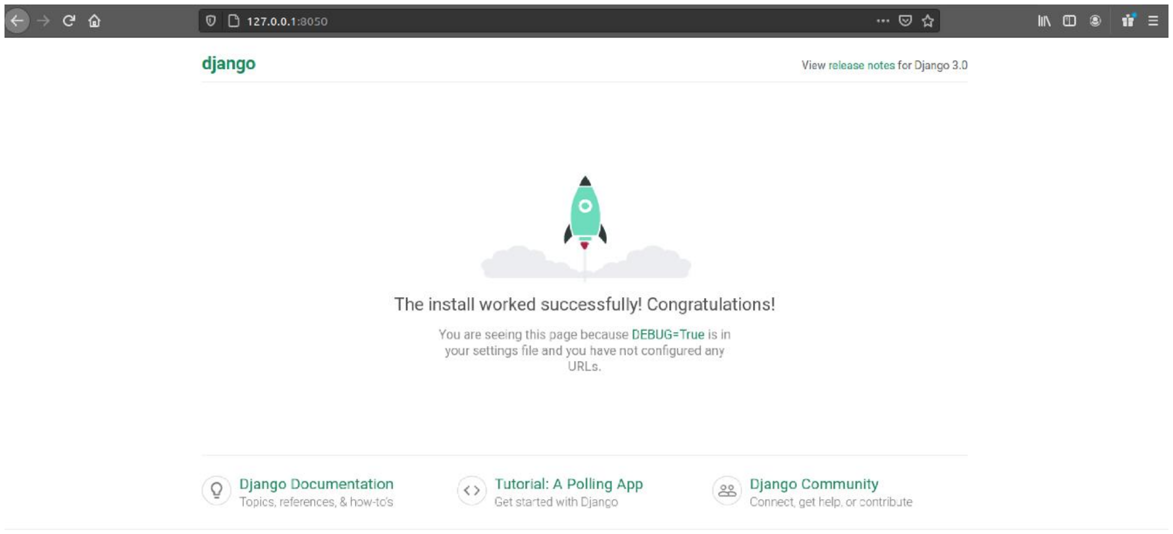
django

View release notes for Django 3.0

The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

   iii. Dev Branch output on port 8050



127.0.0.1:8050

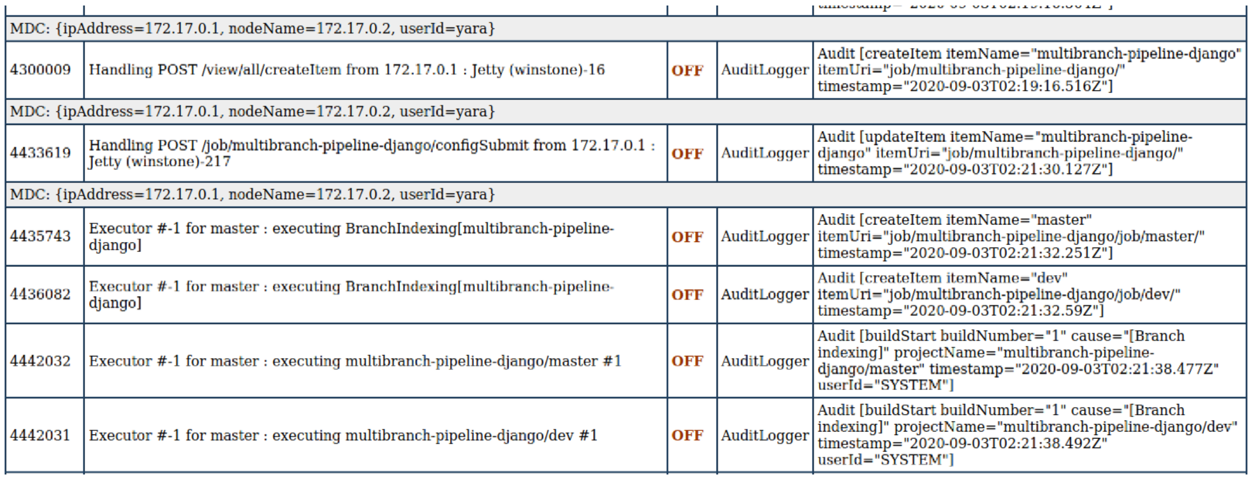
django

View release notes for Django 3.0

The install worked successfully! Congratulations!

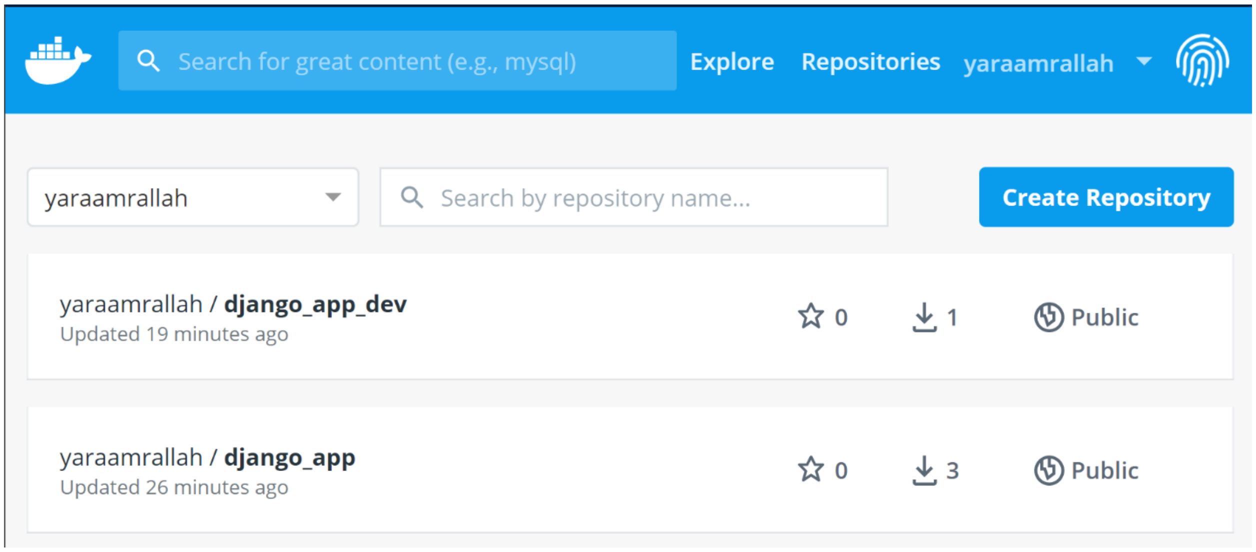
You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

iv. Some Audit Logs



MDC: {ipAddress=172.17.0.1, nodeName=172.17.0.2, userId=yara}				
4300009	Handling POST /view/all/createItem from 172.17.0.1 : Jetty (winstone)-16	OFF	AuditLogger	Audit [createItem itemName="multibranch-pipeline-django" itemUri="job/multibranch-pipeline-django/" timestamp="2020-09-03T02:19:16.516Z"]
MDC: {ipAddress=172.17.0.1, nodeName=172.17.0.2, userId=yara}				
4433619	Handling POST /job/multibranch-pipeline-django/configSubmit from 172.17.0.1 : Jetty (winstone)-217	OFF	AuditLogger	Audit [updateItem itemName="multibranch-pipeline-django" itemUri="job/multibranch-pipeline-django/" timestamp="2020-09-03T02:21:30.127Z"]
MDC: {ipAddress=172.17.0.1, nodeName=172.17.0.2, userId=yara}				
4435743	Executor #-1 for master : executing BranchIndexing[multibranch-pipeline-django]	OFF	AuditLogger	Audit [createItem itemName="master" itemUri="job/multibranch-pipeline-django/job/master/" timestamp="2020-09-03T02:21:32.251Z"]
4436082	Executor #-1 for master : executing BranchIndexing[multibranch-pipeline-django]	OFF	AuditLogger	Audit [createItem itemName="dev" itemUri="job/multibranch-pipeline-django/job/dev/" timestamp="2020-09-03T02:21:32.59Z"]
4442032	Executor #-1 for master : executing multibranch-pipeline-django/master #1	OFF	AuditLogger	Audit [buildStart buildNumber="1" cause="[Branch indexing]" projectName="multibranch-pipeline-django/master" timestamp="2020-09-03T02:21:38.477Z" userId="SYSTEM"]
4442031	Executor #-1 for master : executing multibranch-pipeline-django/dev #1	OFF	AuditLogger	Audit [buildStart buildNumber="1" cause="[Branch indexing]" projectName="multibranch-pipeline-django/dev" timestamp="2020-09-03T02:21:38.492Z" userId="SYSTEM"]

v. Images on DockerHub


Search for great content (e.g., mysql)
Explore
Repositories
yaraamrallah
yaraamrallah
Search by repository name...
Create Repository

yaraamrallah / **django_app_dev**
Updated 19 minutes ago
☆ 0 ⤓ 1 Public

yaraamrallah / **django_app**
Updated 26 minutes ago
☆ 0 ⤓ 3 Public