Automate php/mysql project deploy with docker.

The idea here is to run php application with docker after git push to repository.

The pipeline is as following:

1) Developer has working copy of git repository.

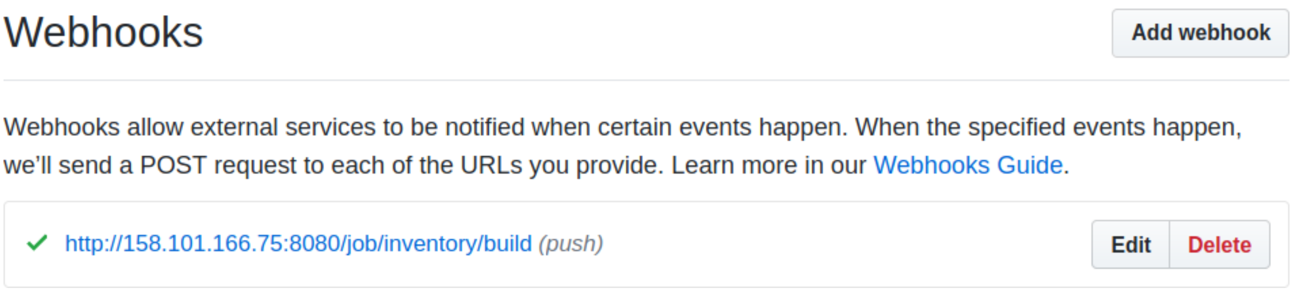
```
$ pwd
./src/inventory
```

```
$ ls -1 htdoc/
add.php
bootstrap.min.js
calibration.php
connector.php
depr.php
docker-compose.yml
edit.php
filter.php
index.html
item.php
jquery.min.js
main.php
mysql
my_style.css
phpdocker
process.php
search.php
setupdb.php
typeahead.min.js
uploads
$ git status
On branch master
nothing to commit, working tree clean
```

2) There is repository on Github.

https://github.com/ppoektos/inventory/tree/master/htdoc

It has web hook that points to Jenkins build:



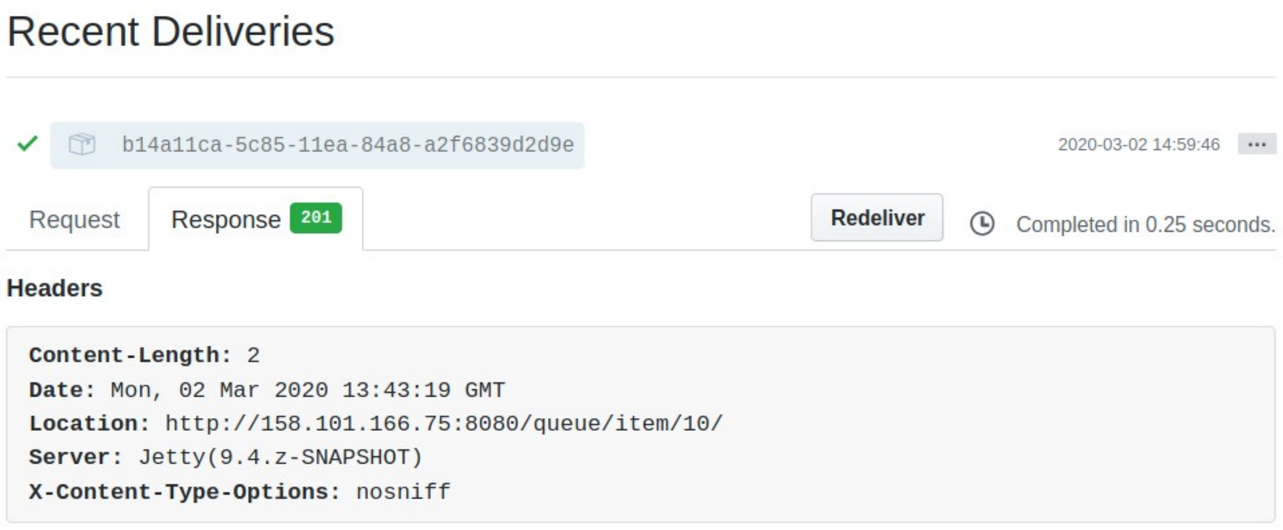
Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ http://158.101.166.75:8080/job/inventory/build *(push)*

3) Developer executes git push and Github sends POST request to Jenkins to initiate build.

```
$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 330 bytes | 330.00 KiB/s,
done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2
local objects.
To github.com:ppoektos/inventory.git
   ed41569..95138be master -> master
```



Recent Deliveries

✓ b14a11ca-5c85-11ea-84a8-a2f6839d2d9e 2020-03-02 14:59:46

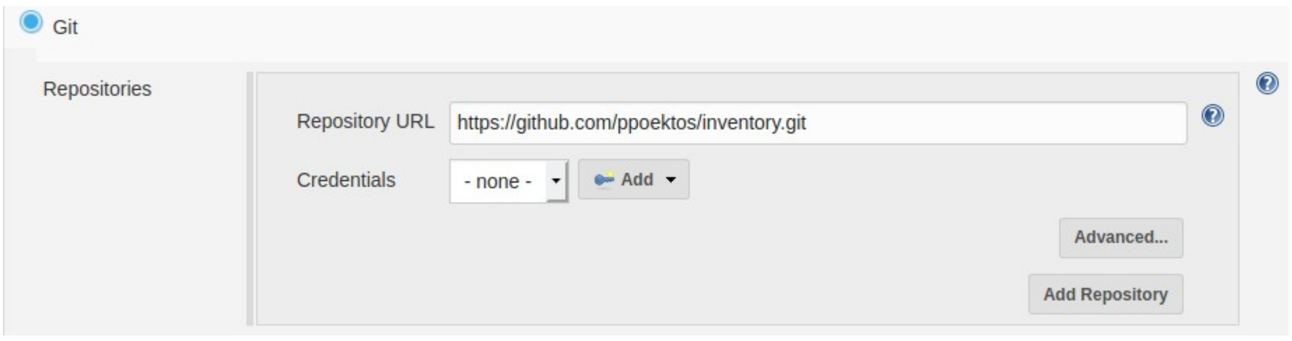
Completed in 0.25 seconds.

Headers

```
Content-Length: 2
Date: Mon, 02 Mar 2020 13:43:19 GMT
Location: http://158.101.166.75:8080/queue/item/10/
Server: Jetty(9.4.z-SNAPSHOT)
X-Content-Type-Options: nosniff
```

4) Jenkins is configured to **Allow anonymous read access** and feature to **Prevent Cross Site Request Forgery exploits** is disabled. This is quick solution to allow Webhooks from Github.

5) There is freestyle projects which has setting to discard old builds (keep only last five). SCM is configured to fetch source from Github:



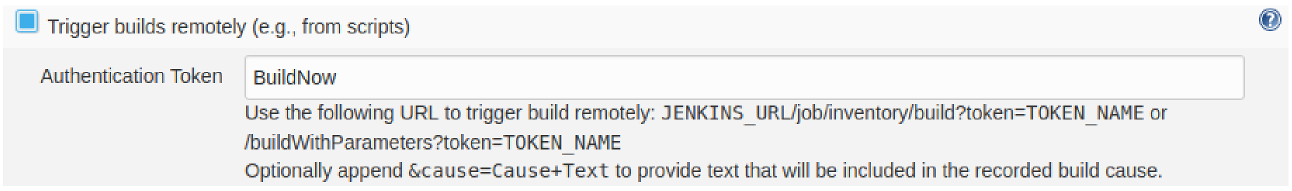
● Git

Repositories

Repository URL: https://github.com/ppoektos/inventory.git

Credentials: - none -

Trigger builds remotely option is enabled:



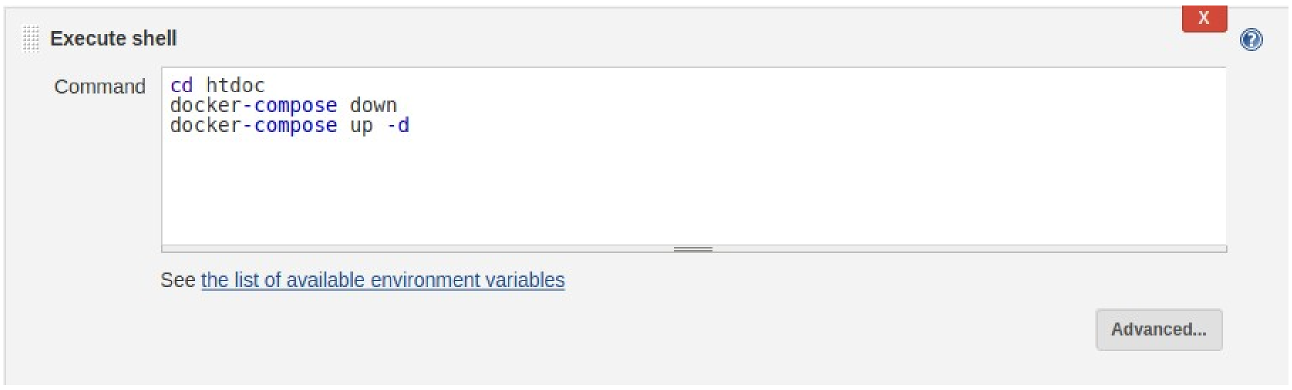
✓ Trigger builds remotely (e.g., from scripts)

Authentication Token: BuildNow

Use the following URL to trigger build remotely: JENKINS_URL/job/inventory/build?token=TOKEN_NAME or /buildWithParameters?token=TOKEN_NAME

Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

There is **Execute shell** build step to run docker-compose:



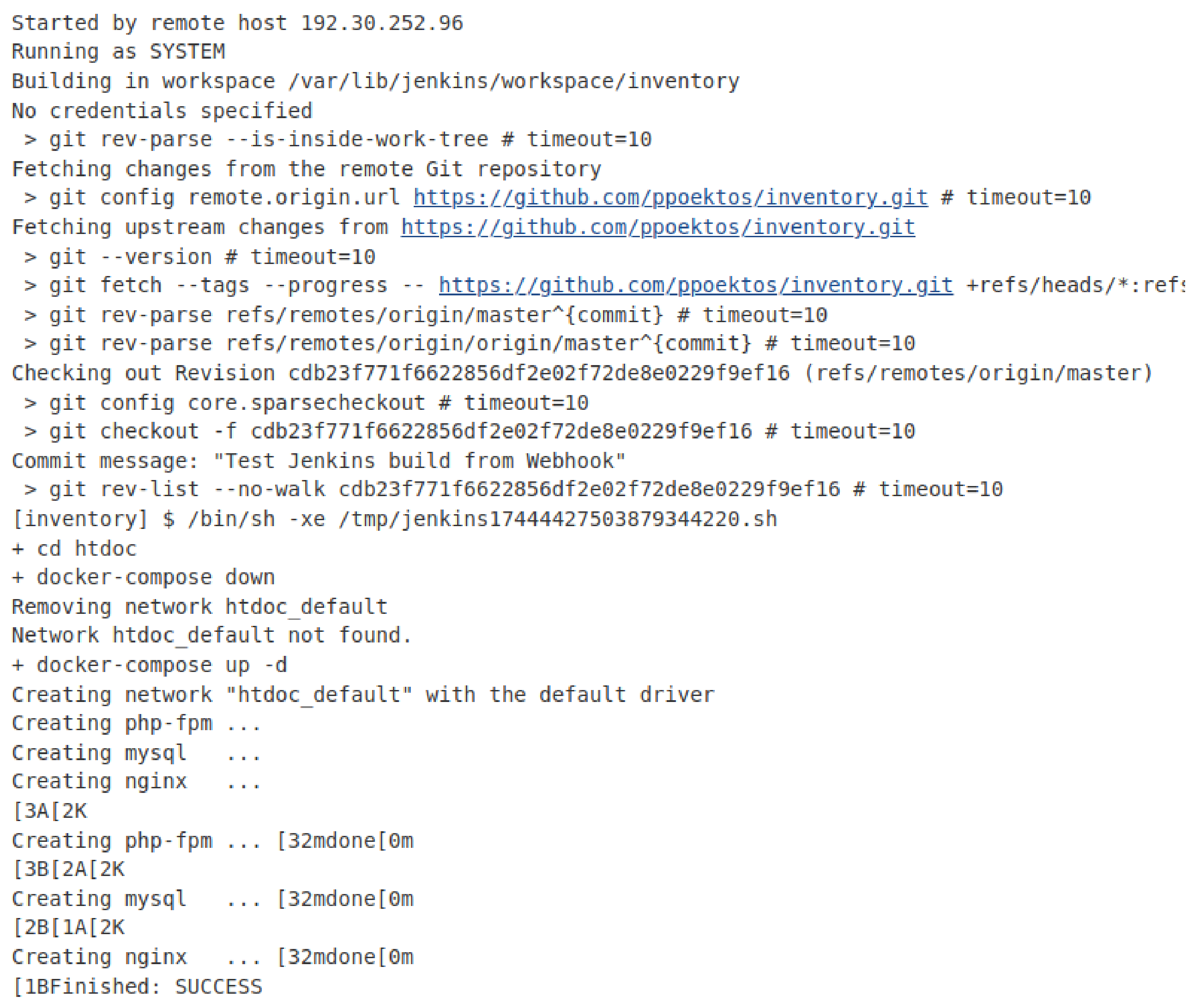
Execute shell

Command

```
cd htdoc
docker-compose down
docker-compose up -d
```

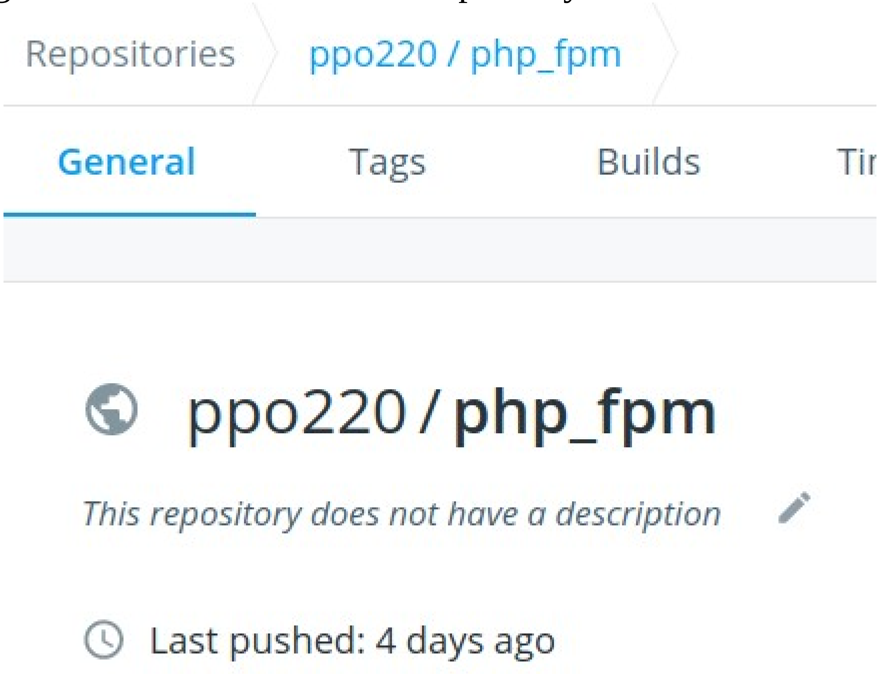
See the list of available environment variables

6) Once webhook is made the build is started. It looks for **docker-compose.yml** file in the project root, downloads images and runs containers.



```
Started by remote host 192.30.252.96
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/inventory
No credentials specified
 > git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/ppoektos/inventory.git # timeout=10
Fetching upstream changes from https://github.com/ppoektos/inventory.git
 > git --version # timeout=10
 > git fetch --tags --progress -- https://github.com/ppoektos/inventory.git +refs/heads/*:ref
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision cdb23f771f6622856df2e02f72de8e0229f9ef16 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f cdb23f771f6622856df2e02f72de8e0229f9ef16 # timeout=10
Commit message: "Test Jenkins build from Webhook"
 > git rev-list --no-walk cdb23f771f6622856df2e02f72de8e0229f9ef16 # timeout=10
[inventory] $ /bin/sh -xe /tmp/jenkins17444427503879344220.sh
+ cd htdoc
+ docker-compose down
Removing network htdoc_default
Network htdoc_default not found.
+ docker-compose up -d
Creating network "htdoc_default" with the default driver
Creating php-fpm ...
Creating mysql   ...
Creating nginx   ...
[3A[2K
Creating php-fpm ... [32mdone[0m
[3B[2A[2K
Creating mysql   ... [32mdone[0m
[2B[1A[2K
Creating nginx   ... [32mdone[0m
[1BFinished: SUCCESS
```

There are three images defined in **docker-compose.yml**: official mysql, nginx and php-fpm image from the own docker-hub repository:

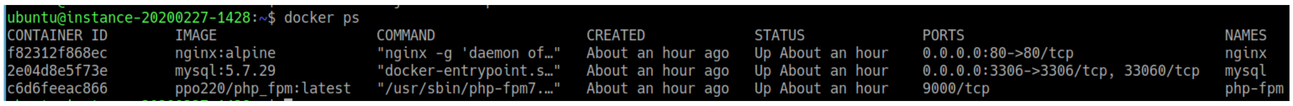


ppo220 / php_fpm

This repository does not have a description

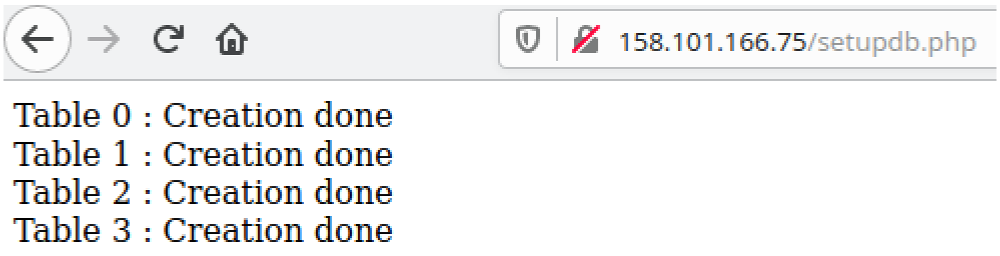
Last pushed: 4 days ago

7) On the server we can see status of containers:



```
ubuntu@instance-20200227-1428:~$ docker ps
CONTAINER ID        IMAGE                 COMMAND                  CREATED             STATUS              PORTS                               NAMES
f82312f868ec        nginx:alpine          "nginx -g 'daemon of…"   About an hour ago   Up About an hour    0.0.0.0:80->80/tcp                  nginx
2e04d8e5f73e        mysql:5.7.29          "docker-entrypoint.s…"   About an hour ago   Up About an hour    0.0.0.0:3306->3306/tcp, 33060/tcp   mysql
c6d6feeac866        ppo220/php_fpm:latest "/usr/sbin/php-fpm7.…"   About an hour ago   Up About an hour    9000/tcp                            php-fpm
```

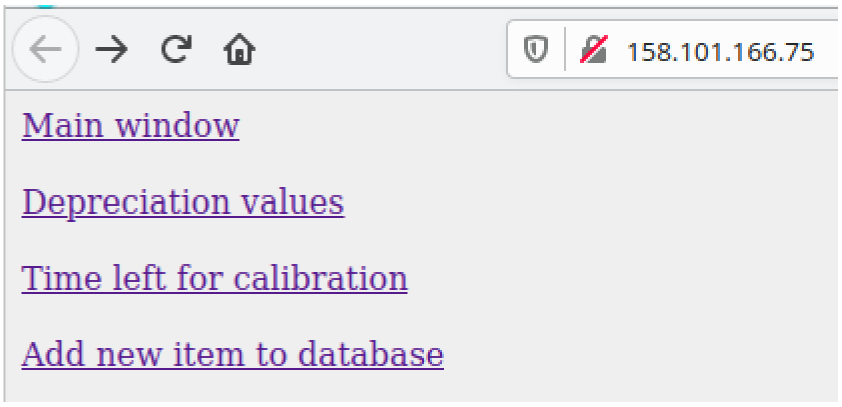
8) There is optional step to create database schema. However it could be improved by adding wget or curl command to the build step in Jenkins.



158.101.166.75/setupdb.php

Table 0 : Creation done
Table 1 : Creation done
Table 2 : Creation done
Table 3 : Creation done

9) Application is now available.



158.101.166.75

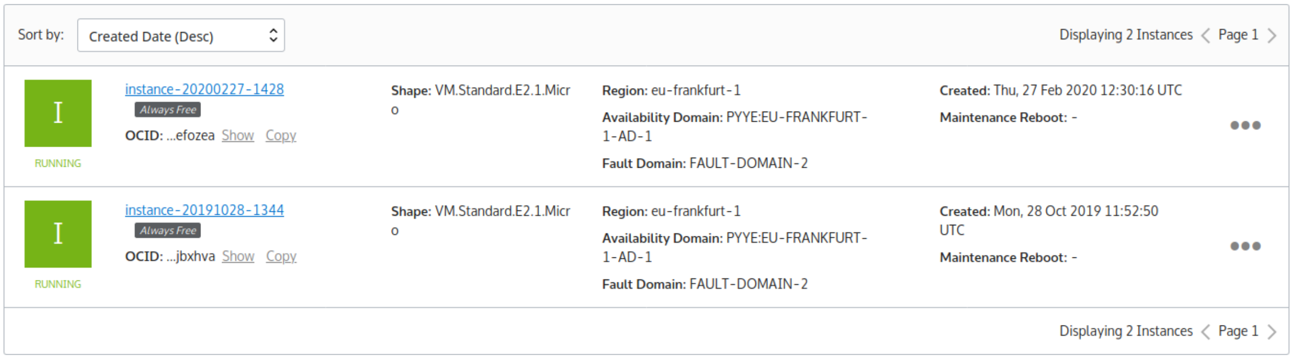
Main window

Depreciation values

Time left for calibration

Add new item to database

10) To perform all steps above I've used two Oracle Cloud instances.



Sort by: Created Date (Desc)   Displaying 2 Instances   Page 1

I RUNNING	instance-20200227-1428 *Always Free* OCID: ...efozea Show Copy	Shape: VM.Standard.E2.1.Micro	Region: eu-frankfurt-1 Availability Domain: PYYE:EU-FRANKFURT-1-AD-1 Fault Domain: FAULT-DOMAIN-2	Created: Thu, 27 Feb 2020 12:30:16 UTC Maintenance Reboot: -
I RUNNING	instance-20191028-1344 *Always Free* OCID: ...jbxhva Show Copy	Shape: VM.Standard.E2.1.Micro	Region: eu-frankfurt-1 Availability Domain: PYYE:EU-FRANKFURT-1-AD-1 Fault Domain: FAULT-DOMAIN-2	Created: Mon, 28 Oct 2019 11:52:50 UTC Maintenance Reboot: -

Displaying 2 Instances   Page 1

One for testing purpose and all screenshot here are from it.
Other is for "Production" use and application is available there by domain name with SSL enabled all time:

https://tb.ektos.net/inventory/