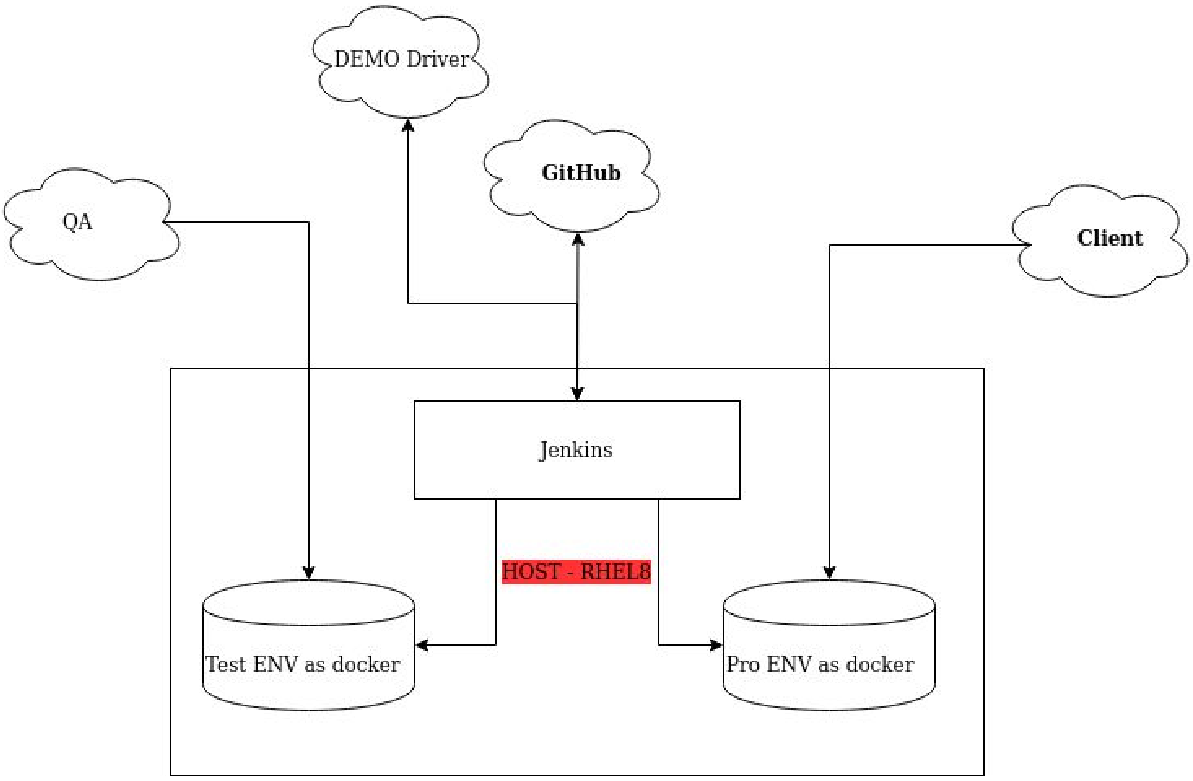
GITkens

The Idea of GITkens is to create a playground to demonstrate the power of the Integration of GIT and Jenkins.

As a first step, I created a web server demo for illustration.

Infrastructure:



TOOLS:

- GitHUb
- Jenkins
- RHEL8
- Docker

How It Works:

The Demo Driver mentioned above is you. Yes you are going to create a branch, check-in the code, deploy the code to test and finally deploy it to production with only to curl commands. Yes you read it, write with only two curl commands.

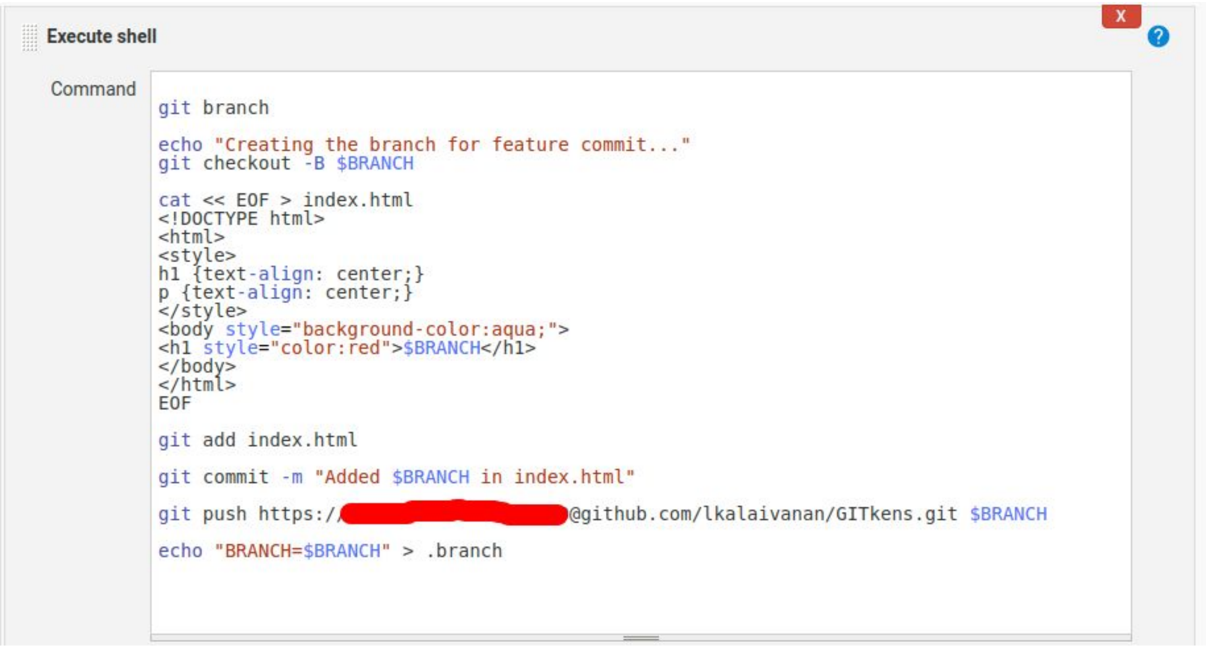
Yes now let's explain how these two curl commands deploy all the steps.

Phase 1) Develop and Deploy code to Test Environment

- Execute the command, curl -I -u <Jenkins-Username>:<Jenkins-Password> http://3ba39d2e.ngrok.io/job/remote-trigger-test/buildWithParameters?token=redhat\&BRANCH=<YourName> (Please don't forget to fill the name)

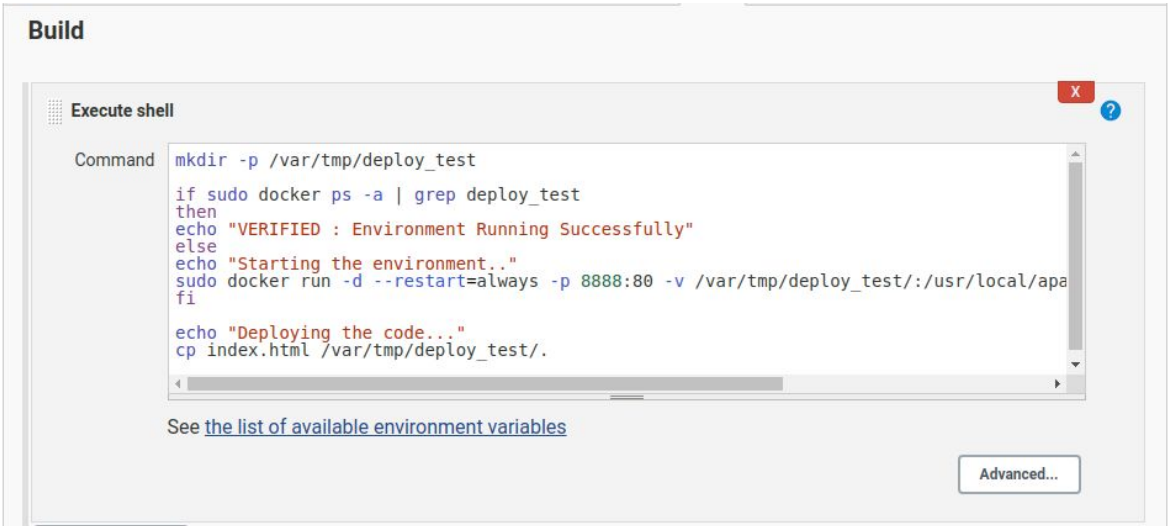
```
$ curl I -u <Jenkins-Username>:<Jenkins-Password>
http://3ba39d2e.ngrok.io/job/remote-trigger-test/buildWithParameters?token=redhat\&BRANCH=kalai

HTTP/1.1 201 Created
Date: Thu, 07 May 2020 08:54:37 GMT
X-Content-Type-Options: nosniff
Location: http://192.168.1.7:8080/queue/item/89/
Content-Length: 0
Server: Jetty(9.4.27.v20200227)
```

- This Curl command triggers the JOB1 in the jenkins, which will take input as your name and create a branch in github in your name and edit the index.html and fill your name on it.



```
git branch
echo "Creating the branch for feature commit..."
git checkout -B $BRANCH
cat << EOF > index.html
<!DOCTYPE html>
<html>
<style>
h1 {text-align: center;}
p {text-align: center;}
</style>
<body style="background-color:aqua;">
<h1 style="color:red">$BRANCH</h1>
</body>
</html>
EOF
git add index.html
git commit -m "Added $BRANCH in index.html"
git push https:/[redacted]@github.com/lkalaivanan/GITkens.git $BRANCH
echo "BRANCH=$BRANCH" > .branch
```

- After successful completion of JOB1, it will trigger a JOB2 which will deploy the code to the Test environment

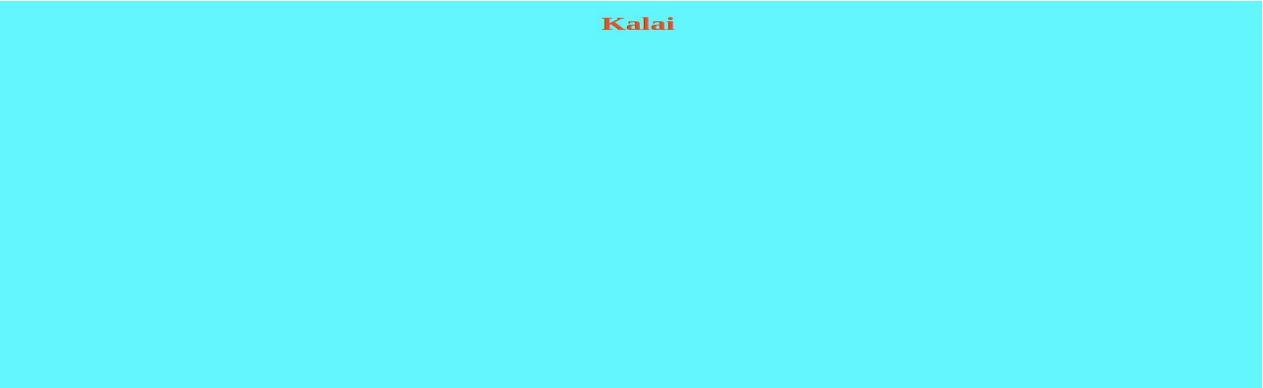


Command

```
mkdir -p /var/tmp/deploy_test

if sudo docker ps -a | grep deploy_test
then
echo "VERIFIED : Environment Running Successfully"
else
echo "Starting the environment.."
sudo docker run -d --restart=always -p 8888:80 -v /var/tmp/deploy_test/:/usr/local/apa
fi

echo "Deploying the code..."
cp index.html /var/tmp/deploy_test/.
```

See the list of available environment variables

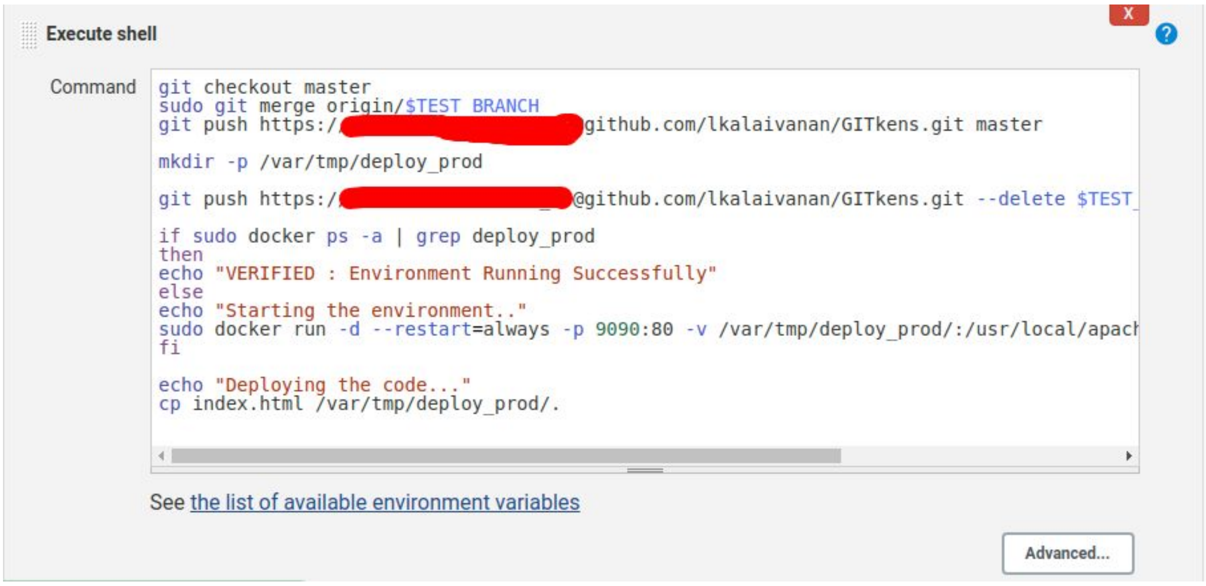
- Open “https://71a86056.ngrok.io/” to check the code deployed



- Once you are happy with the page run command “curl -I -u <Jenkins-Username>:<Jenkins-Password> http://3ba39d2e.ngrok.io/job/Deploy-Prod/buildWithParameters?token=deploy\&TEST_BRANCH=<YourName>” to deploy the same on the production env

```
$ curl -I -u <Jenkins-Username>:<Jenkins-Password>
http://3ba39d2e.ngrok.io/job/Deploy-Prod/buildWithParameters?token=deploy\&TEST_BR
ANCH=kalai
HTTP/1.1 201 Created
Date: Thu, 07 May 2020 09:12:20 GMT
X-Content-Type-Options: nosniff
Location: http://3ba39d2e.ngrok.io/queue/item/97/
Content-Length: 0
Server: Jetty(9.4.27.v20200227)
```

- The above mentioned curl command will trigger a JOB3) which will consider your input as a Tested Branch and merge the branch with master.



```
git checkout master
sudo git merge origin/$TEST BRANCH
git push https:/[redacted]github.com/lkalaivanan/GITkens.git master

mkdir -p /var/tmp/deploy_prod

git push https:/[redacted]@github.com/lkalaivanan/GITkens.git --delete $TEST_

if sudo docker ps -a | grep deploy_prod
then
echo "VERIFIED : Environment Running Successfully"
else
echo "Starting the environment.."
sudo docker run -d --restart=always -p 9090:80 -v /var/tmp/deploy_prod/:/usr/local/apach
fi

echo "Deploying the code..."
cp index.html /var/tmp/deploy_prod/.
```

See the list of available environment variables

- After Merging it will deploy the code on the production environment
- Open “https://a4aa61c0.ngrok.io/” to verify

Kalai