Capstone Project

Cloud DevOps Nanodegree

Prepared by: Mohamed Nabil

Table of Contents

Udacity Capstone Cloud DevOps Nanodegree Project

Introduction

In this project we have been asked to apply the skills and knowledge which were developed throughout the Cloud DevOps Nanodegree program. These include:

- Working in AWS
- Using Jenkins to implement Continuous Integration and Continuous Deployment
- Building pipelines
- Working with Ansible and CloudFormation to deploy clusters
- Building Kubernetes clusters
- Building Docker containers in pipelines

As a capstone project, the directions are rather more open-ended than they were in the previous projects in the program. We were allowed to make some of our own choices in this capstone, for the type of deployment we implement, which services we will use, and the nature of the application we develop. We were asked to develop a CI/CD pipeline for micro services applications with either blue/green deployment or rolling deployment. We were also asked to develop our Continuous Integration steps as we deemed fit but must at least include a typographical checking (aka “linting”).

Once we complete our Continuous Integration, we need to set up Continuous Deployment, which will include:

- Pushing the built Docker container(s) to the Docker repository (we can use AWS ECR, create our own custom Registry within your cluster, or another 3rd party Docker repository); and
- Deploying these Docker container(s) to a small Kubernetes cluster. For our Kubernetes cluster we can either use AWS Kubernetes as a Service, or build our own Kubernetes cluster. To deploy our Kubernetes cluster, we could use either Ansible or CloudFormation. Preferably, run these from within Jenkins as an independent pipeline.

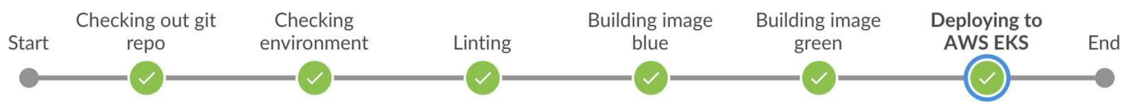
My GitHub Repo

https://github.com/mnabil1989/cloud-devops-capstone-project5-moenabil

Step 1: Propose and Scope the Project

- Plan what your pipeline will look like.

  My pipeline would look like this



- Decide which options you will include in your Continuous Integration phase.

  I decided to have checking the github repo and linting of the docker images

- Pick either Jenkins or Jenkins X to use.

  I decided to use Jenkins on a EC2 instance on AWS https://linuxize.com/post/how-to-install-jenkins-on-ubuntu-18-04/

- Pick a deployment type - either rolling deployment or blue/green deployment.

  I picked a blue/green deployment type https://medium.com/@andresaaap/simple-blue-green-deployment-in-kubernetes-using-minikube-b88907b2e267

- For the Docker application you can either use an application which you come up with, or use an open-source application pulled from the Internet, or if you have no idea, you can use an Nginx "Hello World, my name is (student name)" application

  I picked a simple application Hello World ("Hello World, my name is Moe Nabil") https://medium.com/@andresaaap/simple-blue-green-deployment-in-kubernetes-using-minikube-b88907b2e267

Step 2: Use Jenkins or Jenkins X, and implement blue/green or rolling deployment.

- Create your Jenkins master box with either Jenkins or Jenkins X and install the plugins you will need.

  Installed Jenkins on EC2 instance
  https://linuxize.com/post/how-to-install-jenkins-on-ubuntu-18-04/

  Installed AWS, Blue-Ocean and other required plugins

- Set up your environment to which you will deploy code.

  Setup authentication with

  1. Dockerhub (https://medium.com/@gustavo.guss/jenkins-building-docker-image-and-sending-to-registry-64b84ea45ee9)

  2. AWS

Step 3: Pick AWS Kubernetes as a Service, or build your own Kubernetes cluster.

- Use Ansible or CloudFormation to build your “infrastructure”; i.e., the Kubernetes Cluster.

  Used CloudFormation template to create the EKS cluster. All files related to same can be found here. I found it easy to create EKS using simple shell script also included on the same location

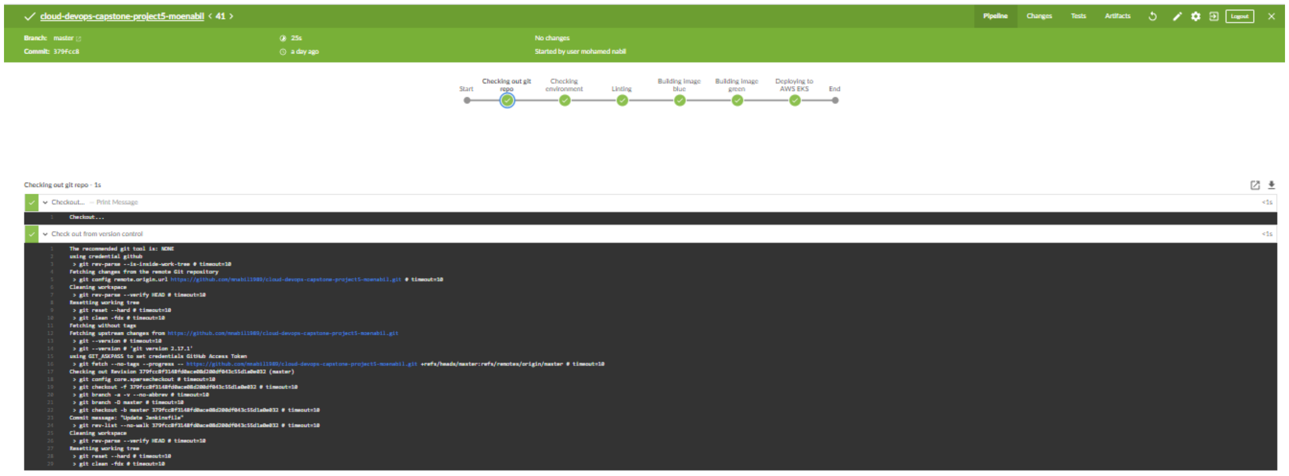
  https://github.com/mnabil1989/cloud-devops-capstone-project5-moenabil/tree/master/aws
- It should create the EC2 instances (if you are building your own), set the correct networking settings, and deploy software to these instances.

- As a final step, the Kubernetes cluster will need to be initialized. The Kubernetes cluster initialization can either be done by hand, or with Ansible/Cloudformation at the student’s discretion.

  The EKS cluster was manually created by me.

Step 4: Build your pipeline

Stage: Check out GitHub Repo



Branch: master
Commit: 379fcc8
25s
a day ago
No changes
Started by user mohamed nabil

Checking out git repo - 1s

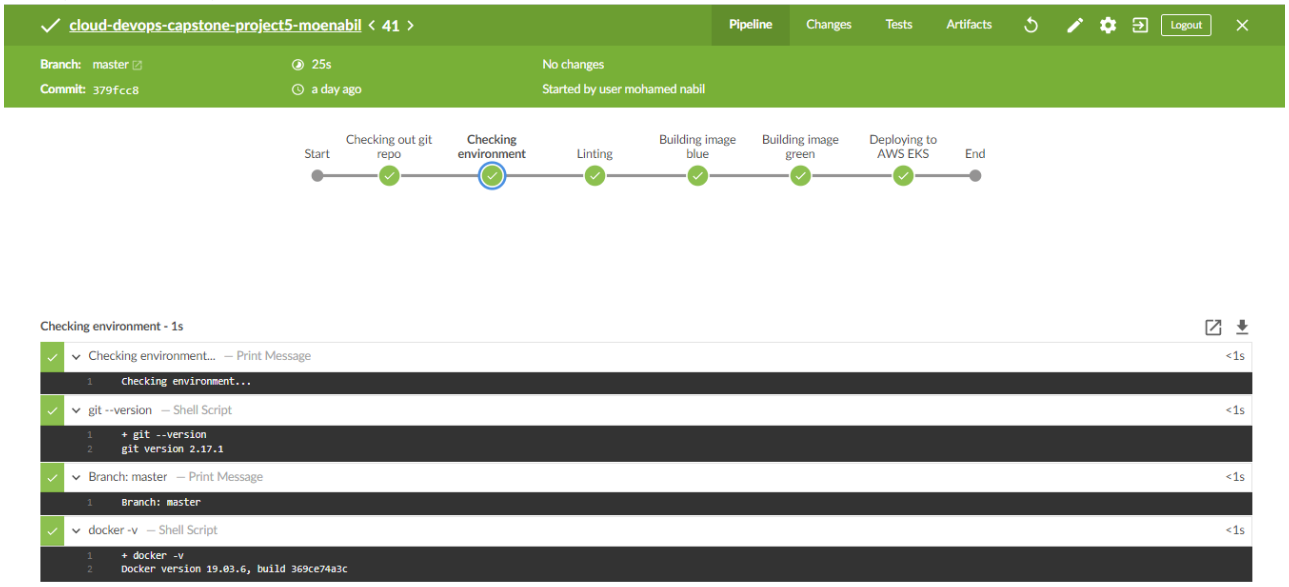
Checkout... — Print Message <1s

```
Checkout...
```

Check out from version control <1s

```
The recommended git tool is: NONE
using credential github
 > git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/mnabil1989/cloud-devops-capstone-project5-moenabil.git # timeout=10
Cleaning workspace
 > git rev-parse --verify HEAD # timeout=10
Resetting working tree
 > git reset --hard # timeout=10
 > git clean -fdx # timeout=10
Fetching without tags
Fetching upstream changes from https://github.com/mnabil1989/cloud-devops-capstone-project5-moenabil.git
 > git --version # timeout=10
 > git --version # 'git version 2.17.1'
using GIT_ASKPASS to set credentials GitHub Access Token
 > git fetch --no-tags --progress -- https://github.com/mnabil1989/cloud-devops-capstone-project5-moenabil.git +refs/heads/master:refs/remotes/origin/master # timeout=10
Checking out Revision 379fcc8f3148fd0ace08d200df043c55d1a0e032 (master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 379fcc8f3148fd0ace08d200df043c55d1a0e032 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git branch -D master # timeout=10
 > git checkout -b master 379fcc8f3148fd0ace08d200df043c55d1a0e032 # timeout=10
Commit message: "Update Jenkinsfile"
 > git rev-list --no-walk 379fcc8f3148fd0ace08d200df043c55d1a0e032 # timeout=10
Cleaning workspace
 > git rev-parse --verify HEAD # timeout=10
Resetting working tree
 > git reset --hard # timeout=10
 > git clean -fdx # timeout=10
```

Stage: Checking Environment



Branch: master
Commit: 379fcc8
25s
a day ago
No changes
Started by user mohamed nabil

Checking environment - 1s

Checking environment... — Print Message <1s

```
Checking environment...
```

git --version — Shell Script <1s

```
+ git --version
git version 2.17.1
```

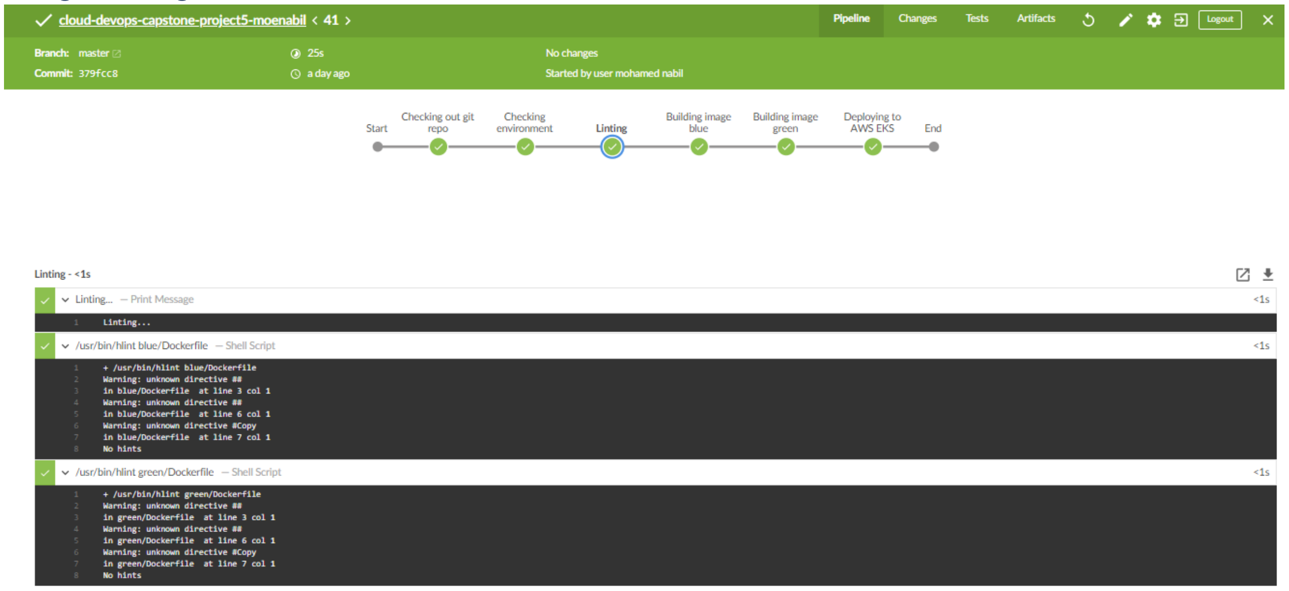
Branch: master — Print Message <1s

```
Branch: master
```

docker -v — Shell Script <1s

```
+ docker -v
Docker version 19.03.6, build 369ce74a3c
```

Stage: Linting



Linting - <1s

Linting... — Print Message <1s

```
Linting...
```

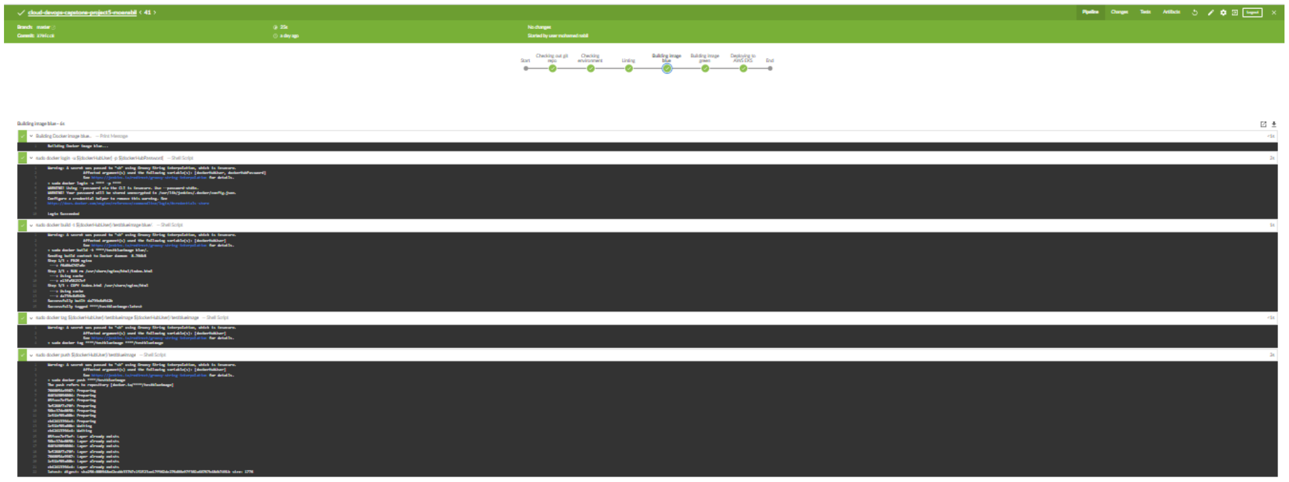
/usr/bin/hlint blue/Dockerfile — Shell Script <1s

```
+ /usr/bin/hlint blue/Dockerfile
Warning: unknown directive ##
in blue/Dockerfile  at line 3 col 1
Warning: unknown directive ##
in blue/Dockerfile  at line 6 col 1
Warning: unknown directive #Copy
in blue/Dockerfile  at line 7 col 1
No hints
```

/usr/bin/hlint green/Dockerfile — Shell Script <1s

```
+ /usr/bin/hlint green/Dockerfile
Warning: unknown directive ##
in green/Dockerfile  at line 3 col 1
Warning: unknown directive ##
in green/Dockerfile  at line 6 col 1
Warning: unknown directive #Copy
in green/Dockerfile  at line 7 col 1
No hints
```

Stage: Build Blue Image



Building image blue - 6s

Building Docker image blue... — Print Message

```
Building Docker image blue...
```

sudo docker login -u ${dockerHubUser} -p ${dockerHubPassword} — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser, dockerHubPassword]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker login -u **** -p ****
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

sudo docker build -t ${dockerHubUser}/testblueimage blue/. — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker build -t ****/testblueimage blue/.
Sending build context to Docker daemon  [illegible]
Step 1/3 : FROM nginx
 ---> [illegible]
Step 2/3 : RUN rm /usr/share/nginx/html/index.html
 ---> Using cache
 ---> [illegible]
Step 3/3 : COPY index.html /usr/share/nginx/html
 ---> Using cache
 ---> [illegible]
Successfully built [illegible]
Successfully tagged ****/testblueimage:latest
```

sudo docker tag ${dockerHubUser}/testblueimage ${dockerHubUser}/testblueimage — Shell Script

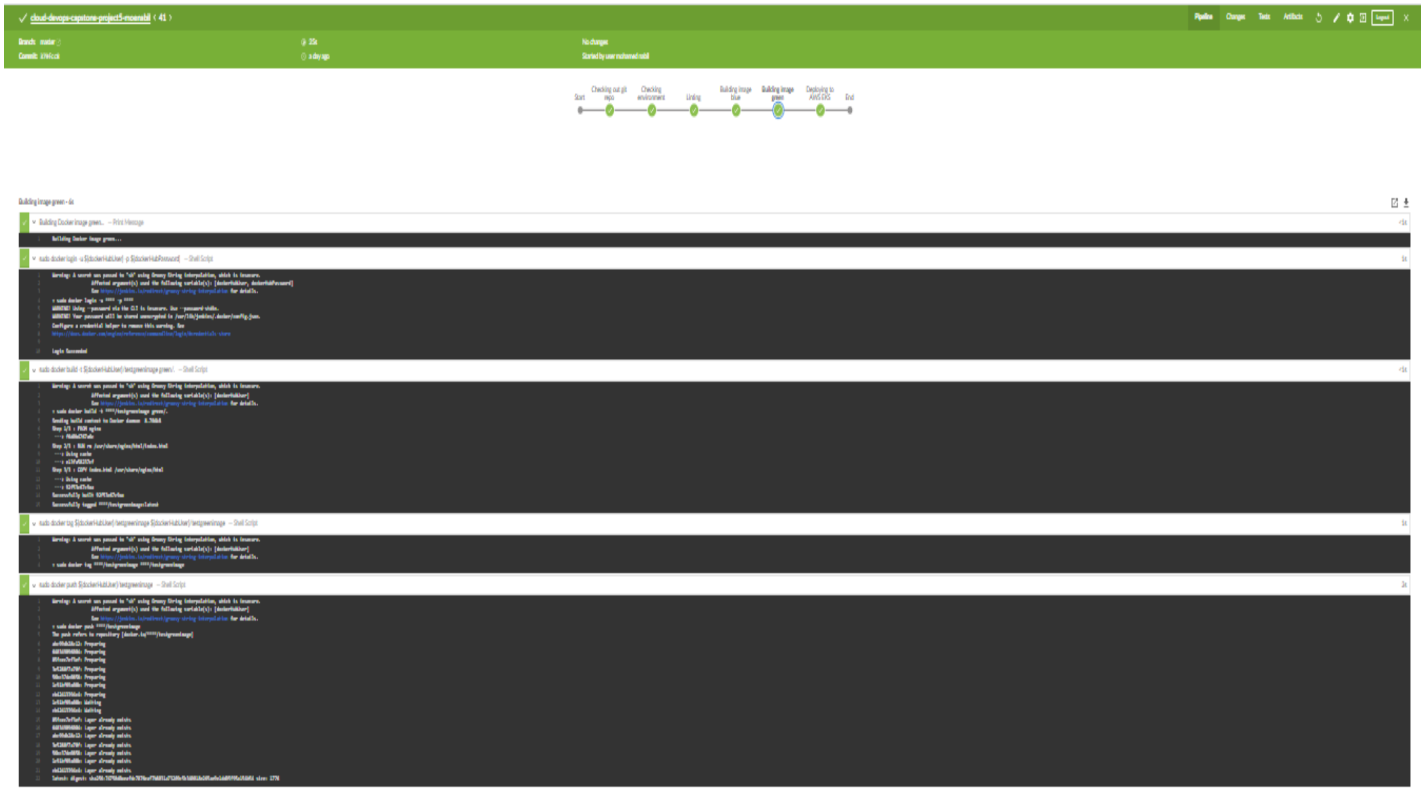
```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker tag ****/testblueimage ****/testblueimage
```

sudo docker push ${dockerHubUser}/testblueimage — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker push ****/testblueimage
The push refers to repository [docker.io/****/testblueimage]
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Waiting
[illegible]: Waiting
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
latest: digest: sha256:[illegible] size: 1778
```

Mohamed Nabil

Stage: Build Green Image



✓ cloud-devops-capstone-project5-monabil ‹ 41 ›

Branch: master
Commit: [illegible]

25s
a day ago

No changes
Started by user mohamed nabil

Building image green - 6s

✓ Building Docker image green... — Print Message

```
Building Docker image green...
```

✓ sudo docker login -u ${dockerHubUser} -p ${dockerHubPassword} — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser, dockerHubPassword]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker login -u **** -p ****
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

✓ sudo docker build -t ${dockerHubUser}/testgreenimage green/. — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker build -t ****/testgreenimage green/.
Sending build context to Docker daemon  [illegible]
Step 1/3 : FROM nginx
 ---> [illegible]
Step 2/3 : RUN rm /usr/share/nginx/html/index.html
 ---> Using cache
 ---> [illegible]
Step 3/3 : COPY index.html /usr/share/nginx/html
 ---> Using cache
 ---> [illegible]
Successfully built [illegible]
Successfully tagged ****/testgreenimage:latest
```

✓ sudo docker tag ${dockerHubUser}/testgreenimage ${dockerHubUser}/testgreenimage — Shell Script

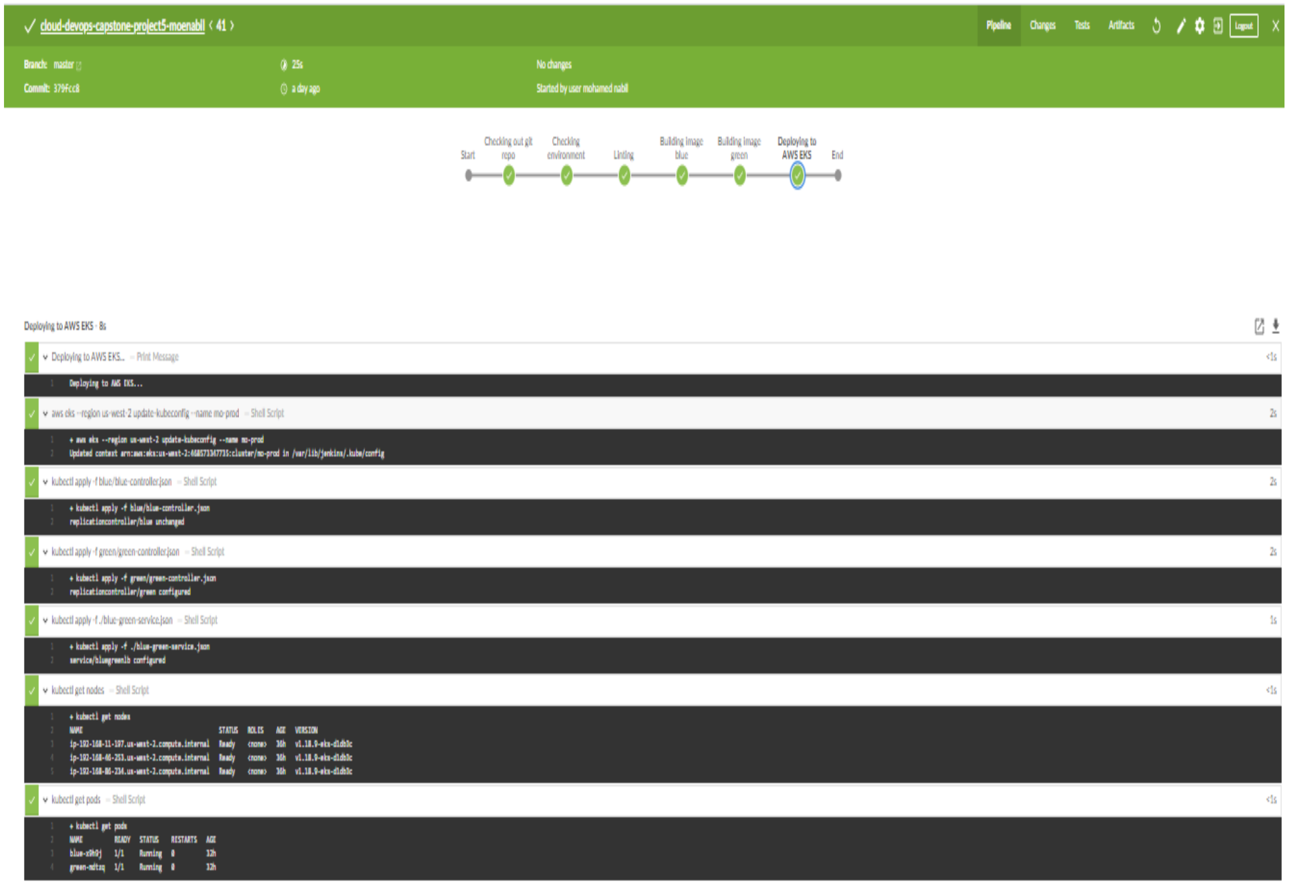
```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker tag ****/testgreenimage ****/testgreenimage
```

✓ sudo docker push ${dockerHubUser}/testgreenimage — Shell Script

```
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubUser]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ sudo docker push ****/testgreenimage
The push refers to repository [docker.io/****/testgreenimage]
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Preparing
[illegible]: Waiting
[illegible]: Waiting
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
[illegible]: Layer already exists
latest: digest: sha256:[illegible] size: 1776
```

Mohamed Nabil

Stage: Deploying to AWS EKS



Deploying to AWS EKS - 8s

✓ Deploying to AWS EKS.. — Print Message <1s

```
Deploying to AWS EKS...
```

✓ aws eks --region us-west-2 update-kubeconfig --name mo-prod — Shell Script 2s

```
+ aws eks --region us-west-2 update-kubeconfig --name mo-prod
Updated context arn:aws:eks:us-west-2:468573347735:cluster/mo-prod in /var/lib/jenkins/.kube/config
```

✓ kubectl apply -f blue/blue-controller.json — Shell Script 2s

```
+ kubectl apply -f blue/blue-controller.json
replicationcontroller/blue unchanged
```

✓ kubectl apply -f green/green-controller.json — Shell Script 2s

```
+ kubectl apply -f green/green-controller.json
replicationcontroller/green configured
```

✓ kubectl apply -f ./blue-green-service.json — Shell Script 1s

```
+ kubectl apply -f ./blue-green-service.json
service/bluegreenlb configured
```

✓ kubectl get nodes — Shell Script <1s

```
+ kubectl get nodes
NAME                                         STATUS   ROLES    AGE   VERSION
ip-192-168-11-197.us-west-2.compute.internal Ready    <none>   36h   v1.18.9-eks-d1db3c
ip-192-168-46-253.us-west-2.compute.internal Ready    <none>   36h   v1.18.9-eks-d1db3c
ip-192-168-86-234.us-west-2.compute.internal Ready    <none>   36h   v1.18.9-eks-d1db3c
```

✓ kubectl get pods — Shell Script <1s

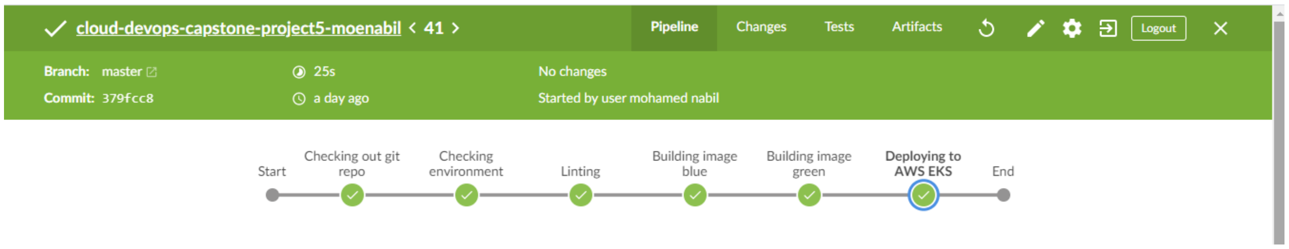
```
+ kubectl get pods
NAME         READY   STATUS    RESTARTS   AGE
blue-x9h9j   1/1     Running   0          32h
green-mdtzq  1/1     Running   0          32h
```

Mohamed Nabil

Step 5: Test your pipeline

- Perform builds on your pipeline.

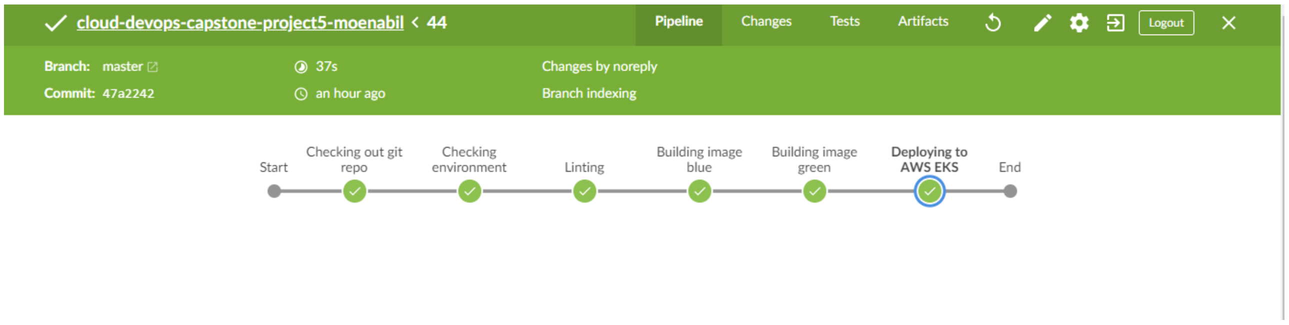
Works fine



Branch: master
Commit: 379fcc8
25s
a day ago
No changes
Started by user mohamed nabil

- Verify that your pipeline works as you designed it.

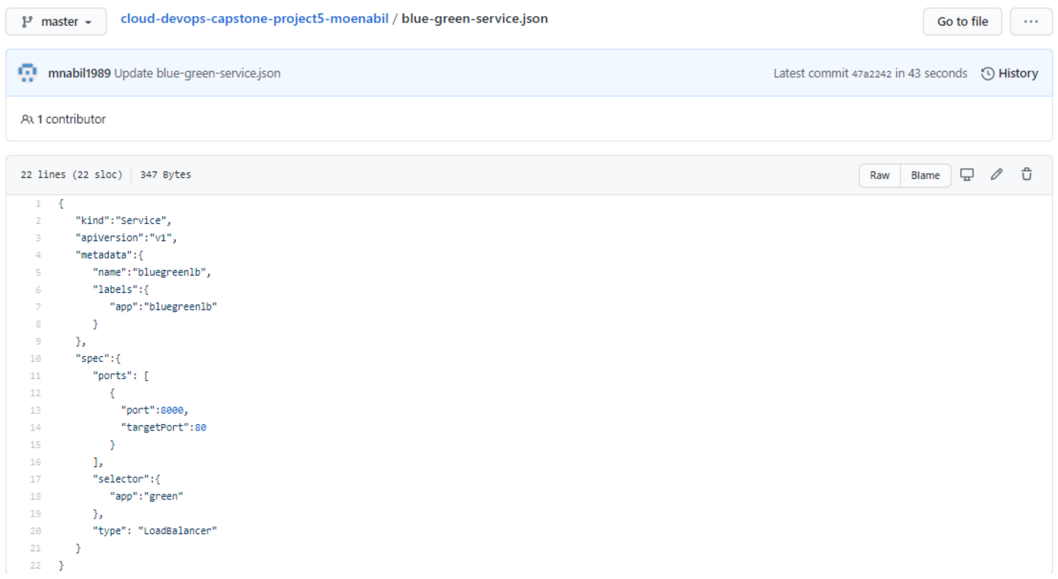
Works fine



Branch: master
Commit: 47a2242
37s
an hour ago
Changes by noreply
Branch indexing

- Take a screenshot of the Jenkins pipeline showing deployment and a screenshot of your AWS EC2 page showing the newly created (for blue/green) or modified (for rolling) instances. Make sure you name your instances differently between blue and green deployments.

Green Deployment



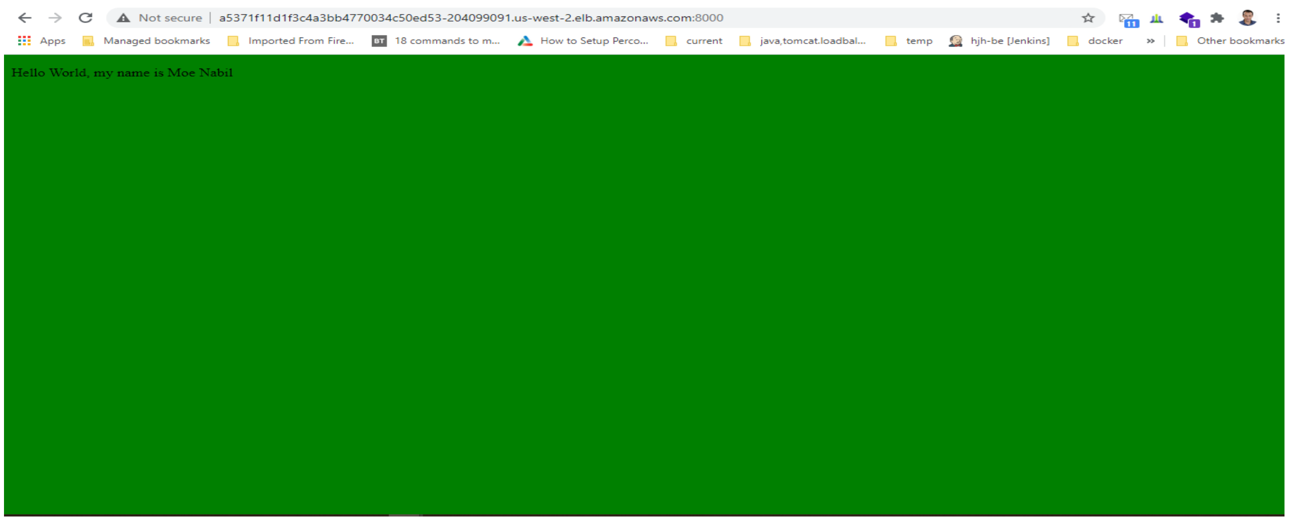
cloud-devops-capstone-project5-moenabil / blue-green-service.json

mnabil1989 Update blue-green-service.json   Latest commit 47a2242 in 43 seconds

1 contributor

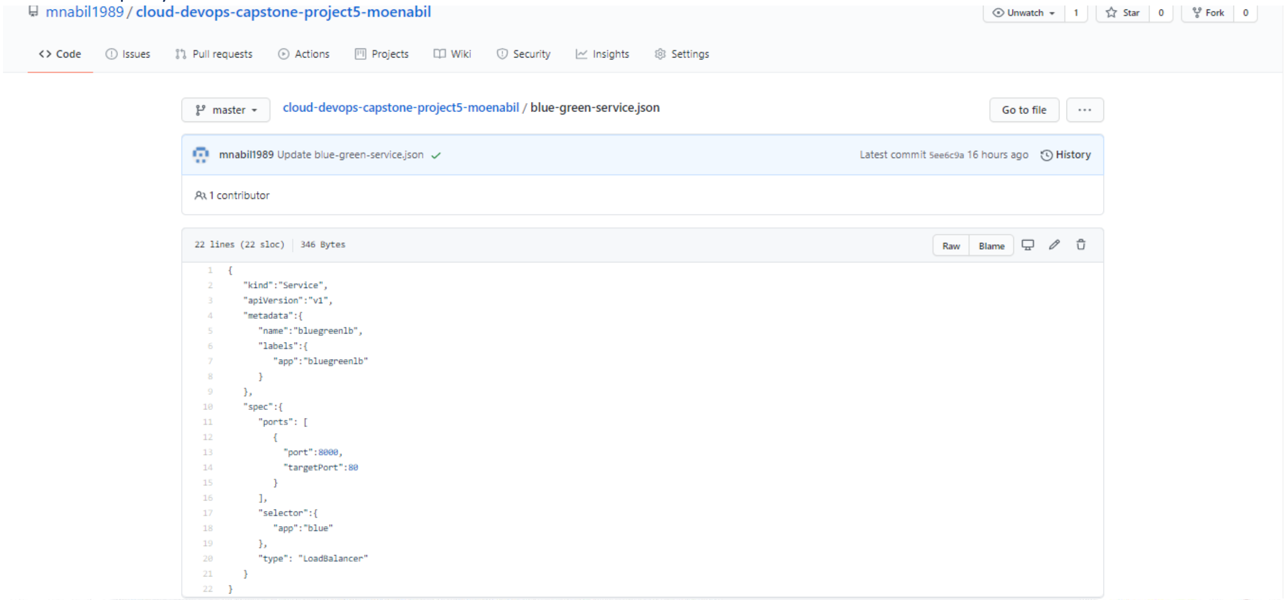
22 lines (22 sloc) | 347 Bytes

```
{
   "kind":"Service",
   "apiVersion":"v1",
   "metadata":{
      "name":"bluegreenlb",
      "labels":{
         "app":"bluegreenlb"
      }
   },
   "spec":{
      "ports": [
         {
           "port":8000,
           "targetPort":80
         }
      ],
      "selector":{
         "app":"green"
      },
      "type": "LoadBalancer"
   }
}
```



Hello World, my name is Moe Nabil

Blue-Deployment



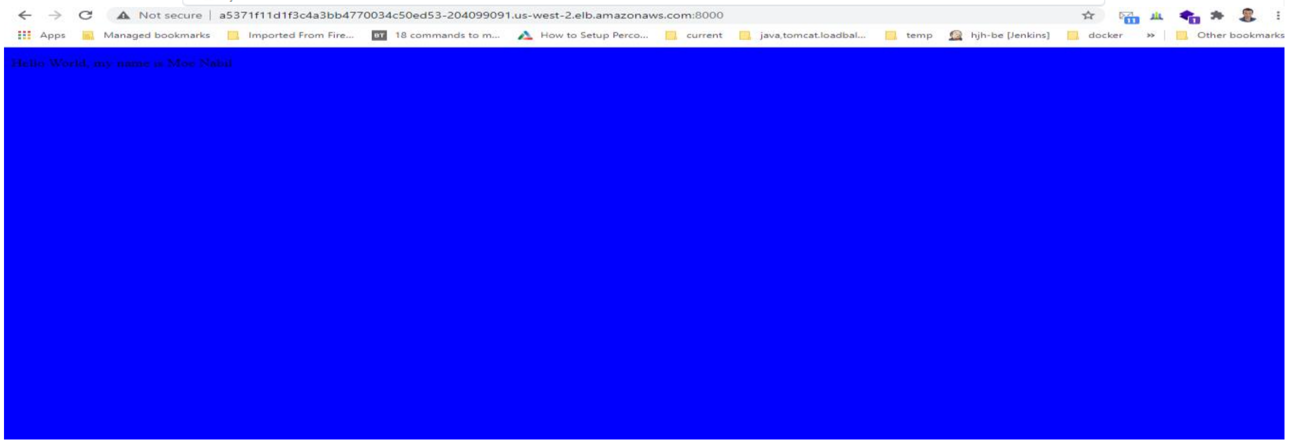
cloud-devops-capstone-project5-moenabil / blue-green-service.json

mnabil1989 Update blue-green-service.json ✓ Latest commit 5ee6c9a 16 hours ago

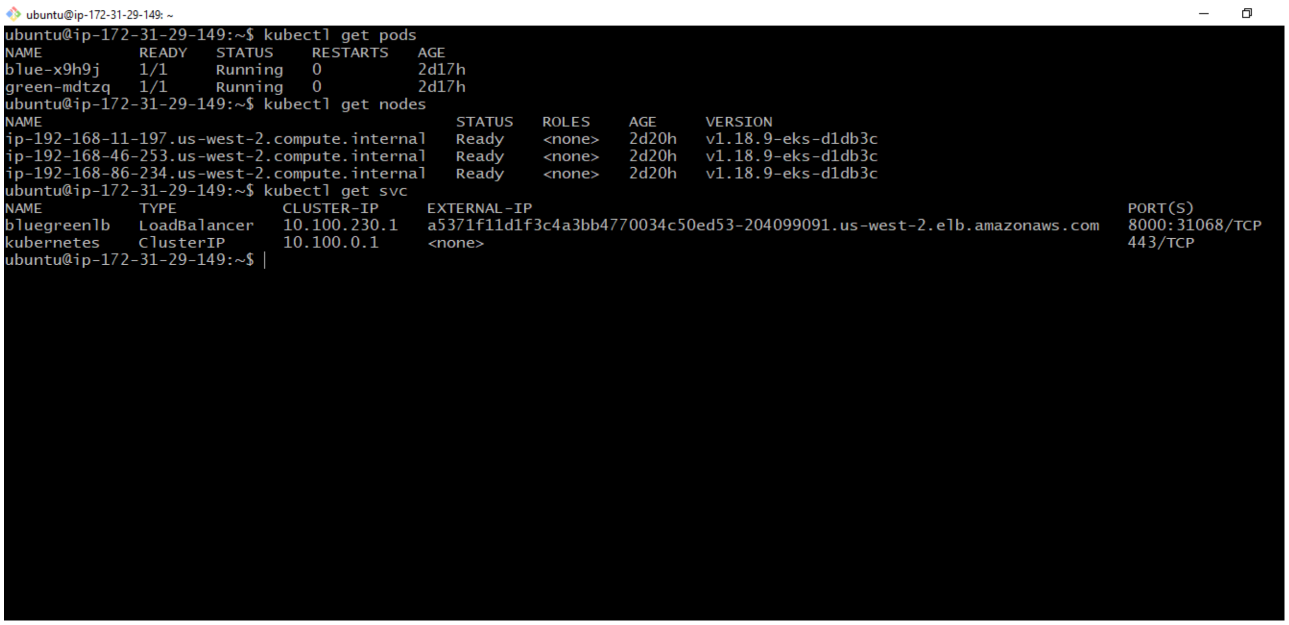
1 contributor

22 lines (22 sloc) | 346 Bytes

```
{
   "kind":"Service",
   "apiVersion":"v1",
   "metadata":{
      "name":"bluegreenlb",
      "labels":{
         "app":"bluegreenlb"
      }
   },
   "spec":{
      "ports": [
         {
           "port":8000,
           "targetPort":80
         }
      ],
      "selector":{
         "app":"blue"
      },
      "type": "LoadBalancer"
   }
}
```



Hello World, my name is Moe Nabil



```
ubuntu@ip-172-31-29-149:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
blue-x9h9j    1/1     Running   0          2d17h
green-mdtzq   1/1     Running   0          2d17h
ubuntu@ip-172-31-29-149:~$ kubectl get nodes
NAME                                         STATUS   ROLES    AGE     VERSION
ip-192-168-11-197.us-west-2.compute.internal   Ready    <none>   2d20h   v1.18.9-eks-d1db3c
ip-192-168-46-253.us-west-2.compute.internal   Ready    <none>   2d20h   v1.18.9-eks-d1db3c
ip-192-168-86-234.us-west-2.compute.internal   Ready    <none>   2d20h   v1.18.9-eks-d1db3c
ubuntu@ip-172-31-29-149:~$ kubectl get svc
NAME          TYPE           CLUSTER-IP     EXTERNAL-IP                                                               PORT(S)
bluegreenlb   LoadBalancer   10.100.230.1   a5371f11d1f3c4a3bb4770034c50ed53-204099091.us-west-2.elb.amazonaws.com   8000:31068/TCP
kubernetes    ClusterIP      10.100.0.1     <none>                                                                    443/TCP
ubuntu@ip-172-31-29-149:~$ |
```