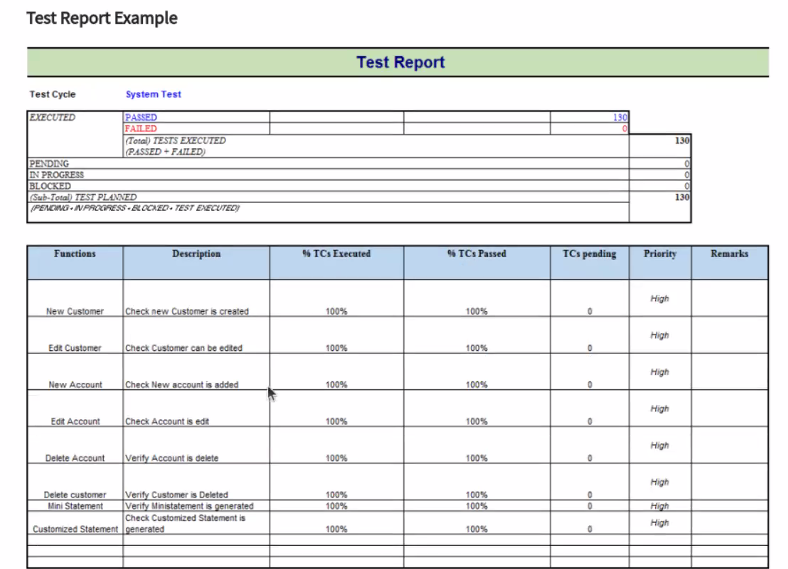
1. Adımdayız

Bu aşamada manuel testler yapılacak. Maunel testerlar bu aşamada uygulamayı manuel olarak test ediyorlar.

Yani insan gözüyle test yapılıyor. QA testerlar, sonrasında da bir rapor çıkarıyorlar.



New customer a tıkladım çalıştı mı? Account a tıkladım çalıştı mı gibi.

Nightly pipeline da her şey sıfırdan oluşuyordu. Weekly pipeline da ise oluşan cluster hep çalışıyor, petllinic weekly pipeline haftalık çalıştığında uygulamanın en güncel halini her pazar gecesi ayağa kaldırmış oluyor.

O bir hafta içinde de developlerlar hatalarını düzeltiyor, güncellemelerini yapıyor. Bu ekilde devam eden bir süreç.

Burada ECR repomuz da sürekli kalacak. Bu aşamada yeni bir şey yapmadani petclinic-nightly pipeline ı alıp ihityacımız olmayan kısmı çıkaracaz.

Jenkins arayüzünde EKS yi yönetecez burada.

git checkout dev

git branch feature/msp-19

git checkout feature/msp-19

EKS yi manuel kuramayızi otomasyon yapacaz. EKS cluster ı cli ile nasıl kuruyorduk.

eksctl create cluster 
- -name cu-cluster 
--region us-east-I 
- -zones us-east-la, us-east -1b, us -east-Ic 
-nodegroup•nMe my • nodes \ 
- type t3a. medium \ 
--nodes 2 
--nodes-min 2 
--nodes-max 3 
- -ssh-access 
• • SSh -public • key 
- -managed 
SSh/id_rsa.pub 

Bunun için de komutlarını uzun uzun yazmak yerine configürasyon file ını kullancaz.

Using Config Files 
You can create a cluster using a config file instead of flags. 
First, create cluster. yaml file: 
Esctl. 10/ v IaIpha5 
apiVerSiOn : 
kind: ClusterConfig 
metadata 
name: basic-cluster 
region: eu-north-l 
nodeGroup s : 
name: ng-l 
instance Type: m5.Iarge 
desiredCapacity: I e 
volumeSIze: 80 
ssh: 
allow: true will use 
name: ng-2 
instanceType: mS.xIarge 
desir edCapacity : 
volumeSize : 100 
ssh: 
publicKeyPath: 
2 
.ssh/id_rsa.pub as the default ssh key 
pub 

Şimdi infrastrcuture altında qa-eks-cluster

isimli bir klasör oluşturup burada cluster.yaml oluşturacaz.

apiVersion: eksctl.io/v1alpha5

kind: ClusterConfig

metadata:

name: petclinic-cluster

region: us-east-1

availabilityZones: ["us-east-1a", "us-east-1b", "us-east-1c"]

managedNodeGroups:

- name: ng-1

instanceType: t3a.medium

desiredCapacity: 2

minSize: 2

maxSize: 3

volumeSize: 8

bunu proje dosyamızda dursun diye oluşturduk. bunu burada kullanmayacaz.

git add .

git commit -m 'added cluster.yaml file'

git push --set-upstream origin feature/msp-19

git checkout dev

git merge feature/msp-19

git push origin dev

şimdi cluster oluşturacaz.

cd ile üste çıktık ( nerde olacağımızın bir önemi yok)

curl --silent --location "<https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname> -s)\_amd64.tar.gz" | tar xz -C /tmp

sudo mv /tmp/eksctl /usr/local/bin

eksctl binary sini /sr/local/bin altında gönderdik.

versionu grelim:

eksctl version

AWS nin EKS kubectl ini indiriyoruz.

curl -O <https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.4/2023-05-11/bin/linux/amd64/kubectl>

executable yetkisini veriyoruz:

chmod +x ./kubectl

kubectl binary sini /usr/local/bin altına atıyoruz:

sudo mv kubectl /usr/local/bin

bu işlemi kubectl i ya da eksctl i her kulanıcı kullansın diye yapıyoruz.

versionu görelim:

kubectl version --short --client

Kustomize önceden helm yerine kullanılıyordu, artık helm kullanılıyor.

• ec2-user@jenkins-server# kubectl version 
-short - -client 
Flag --short has been deprecated, and will be removed in the future. 
-short output will become the default. 
Client Version: VI. 26.4-eks -Øa21954 
Kustomize Version: v4.5.7 
The 

şimdi jenkins user a geçiyoruz. biz eks de hangi kullanıcı ile cluster oluşturursak o kullanıcı ilave bir configürasyon olmaksızın EKS e erişebilir.

Biz de jenkins arayüzüyle cluster ı yönetecez. bu nedenle jenkins user a geçiyoruz.

/var/lib/jenkins

altında vi ile cluster.yaml oluşturuyoruz.

[jenkins@jenkins-server pwd 
/var/lib/jenkins 

apiVersion: eksctl.io/v1alpha5

kind: ClusterConfig

metadata:

name: petclinic-cluster

region: us-east-1

availabilityZones: ["us-east-1a", "us-east-1b", "us-east-1c"]

managedNodeGroups:

- name: ng-1

instanceType: t3a.medium

desiredCapacity: 2

minSize: 2

maxSize: 3

volumeSize: 8

cluster aisim verdik, availibility zone ları veriyoruz ki yeni vpc oluşturmasın. bizim hazır vpc mizi kullansın. nodegrubun parametrelerini giriyoruz, kullanazağımızın instance ın özelliklerini yani.

tek AZ de olabilirdi.

eksctl create cluster -f cluster.yaml

cluster.yaml dosyasını kullanarak eks cluster ı oluşturuyoruz. bu komut .kube oluşturarak altındaki conf ları da yapar.

ec2-user olarak da cluster a bağlanmak istiyorsak e yapmamız lazım? .kube u ec-2 usr a da almamız lazım. ancak kullanıcı bilgilerini de chown ile değiştirmemiz gerekir.

[jenkins@jenkins-server Is -a kube 
cache config 
(jenkins@jenkins-server —IS Is 
total 20 
drwxr-xr-x. 3 jenkins j enkins 
08:55 
23 jenkins jenkins 
09:23 
drwxr-x---. 
4 jenkins jenkins 
1 jet*ins jenkins 
(jenkins@jenkins-sérver -IS I 
-al . kube 
33 Jun 
16384 Jun 
35 Jun 
24€7 Jun 
le 
08:57 cache 
08:55 config 

cluster oluşması biraz zaman alır.

bu komutu girince arkada cloudformation oluşturup işlemleri onun üstünden oluşturur.

CloudFormation > Stacks 
Stacks (1) 
Q Filter by stack name 
O 
Stack name 
eksctl-petclinic-cluster-cluster 
Status 
@ CREATE_ 
IN 
PROGRESS 
Created time 
2023-06-10 1040:45 UTC+0200 

kullandığımız terminalde cluster oluştuğu için yeni bir terminal açıyor orada devam ediyoruz. 2605 i yükleme bitince yapacaz.

ingress controller yükleme komutunu 2605. satır:

kubectl apply -f <https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.8.0/deploy/static/provider/cloud/deploy.yaml)>

20. adımdayız

git checkout dev

git branch feature/msp-20

git checkout feature/msp-20

önce create-ecr-docker-registry-for-petclinic-qa

isimli bir jenkins job la bir ECR repo oluşturacaz.

PATH="$PATH:/usr/local/bin"

APP\_REPO\_NAME="clarusway-repo/petclinic-app-qa"

AWS\_REGION="us-east-1"

aws ecr describe-repositories --region ${AWS\_REGION} --repository-name ${APP\_REPO\_NAME} || \

aws ecr create-repository \

--repository-name ${APP\_REPO\_NAME} \

--image-scanning-configuration scanOnPush=false \

--image-tag-mutability MUTABLE \

--region ${AWS\_REGION}

komutta önce describe yapıyoruz, varsa oluşturmasın diye. || olmadan create deseydik hata alırdık.

pipeline yaptığımız zaman böyle bir hata yüzünden pipeline durur.

Başka hiçbir conf yapmadan Apply/Save/Build Now diyoruz.

ECR a gidip kontrol edelim.

Create re osito 
Q Find repositories 
Repository 
name 
cla rusway- 
repo/petclinic- 
app-ga 
d, 
995714465210.dkr.ecr.us- 
east- 
1.amazonaws.com/clarusway- 
repo/petclinic-app-qa 

jenkins klasörü altında docker image lerin tagini oluşutran kodları prepare-tags-ecr-for-qa-docker-images.sh

isimli bir dosya ile kaydedelim.

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-admin-server/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_ADMIN\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:admin-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-api-gateway/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_API\_GATEWAY="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:api-gateway-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-config-server/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_CONFIG\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:config-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-customers-service/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_CUSTOMERS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:customers-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-discovery-server/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_DISCOVERY\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:discovery-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-hystrix-dashboard/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_HYSTRIX\_DASHBOARD="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:hystrix-dashboard-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-vets-service/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_VETS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:vets-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=$(. ${WORKSPACE}/spring-petclinic-visits-service/target/maven-archiver/pom.properties && echo $version)

export IMAGE\_TAG\_VISITS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:visits-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

export IMAGE\_TAG\_GRAFANA\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:grafana-service"

export IMAGE\_TAG\_PROMETHEUS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:prometheus-service"

bir önceki pipeline daki komutlarla aynı.

docker build komutarı için aynı klasörde build-qa-docker-images-for-ecr.sh

isimli bir dosya oluşturuyoruz.

docker build --force-rm -t "${IMAGE\_TAG\_ADMIN\_SERVER}" "${WORKSPACE}/spring-petclinic-admin-server"

docker build --force-rm -t "${IMAGE\_TAG\_API\_GATEWAY}" "${WORKSPACE}/spring-petclinic-api-gateway"

docker build --force-rm -t "${IMAGE\_TAG\_CONFIG\_SERVER}" "${WORKSPACE}/spring-petclinic-config-server"

docker build --force-rm -t "${IMAGE\_TAG\_CUSTOMERS\_SERVICE}" "${WORKSPACE}/spring-petclinic-customers-service"

docker build --force-rm -t "${IMAGE\_TAG\_DISCOVERY\_SERVER}" "${WORKSPACE}/spring-petclinic-discovery-server"

docker build --force-rm -t "${IMAGE\_TAG\_HYSTRIX\_DASHBOARD}" "${WORKSPACE}/spring-petclinic-hystrix-dashboard"

docker build --force-rm -t "${IMAGE\_TAG\_VETS\_SERVICE}" "${WORKSPACE}/spring-petclinic-vets-service"

docker build --force-rm -t "${IMAGE\_TAG\_VISITS\_SERVICE}" "${WORKSPACE}/spring-petclinic-visits-service"

docker build --force-rm -t "${IMAGE\_TAG\_GRAFANA\_SERVICE}" "${WORKSPACE}/docker/grafana"

docker build --force-rm -t "${IMAGE\_TAG\_PROMETHEUS\_SERVICE}" "${WORKSPACE}/docker/prometheus"

bir önceki pipeline daki komutlarla aynı.

imageleri push etmek için gerekli komutları da aynı kalsörde

push-qa-docker-images-to-ecr.sh

isimli dosyay atıyoruz.

aws ecr get-login-password --region ${AWS\_REGION} | docker login --username AWS --password-stdin ${ECR\_REGISTRY}

docker push "${IMAGE\_TAG\_ADMIN\_SERVER}"

docker push "${IMAGE\_TAG\_API\_GATEWAY}"

docker push "${IMAGE\_TAG\_CONFIG\_SERVER}"

docker push "${IMAGE\_TAG\_CUSTOMERS\_SERVICE}"

docker push "${IMAGE\_TAG\_DISCOVERY\_SERVER}"

docker push "${IMAGE\_TAG\_HYSTRIX\_DASHBOARD}"

docker push "${IMAGE\_TAG\_VETS\_SERVICE}"

docker push "${IMAGE\_TAG\_VISITS\_SERVICE}"

docker push "${IMAGE\_TAG\_GRAFANA\_SERVICE}"

docker push "${IMAGE\_TAG\_PROMETHEUS\_SERVICE}"

bir önceki pipeline daki komutlarla aynı.

(cluster hazır olduğu için bu ensada, bir önceki adımda yer alan ingress controller yükleme komutunu giriyoruz kubectl apply -f <https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.8.0/deploy/static/provider/cloud/deploy.yaml)>

şimdi ise application nu QA environmentinde deploym etmek için gerekli komutları deploy\_app\_on\_qa\_environment.sh

dosyasına atalım. dosya jenkins folder altında.

echo 'Deploying App on Kubernetes'

envsubst < k8s/petclinic\_chart/values-template.yaml > k8s/petclinic\_chart/values.yaml

sed -i s/HELM\_VERSION/${BUILD\_NUMBER}/ k8s/petclinic\_chart/Chart.yaml

AWS\_REGION=$AWS\_REGION helm repo add stable-petclinic s3://petclinic-helm-charts-<put-your-name>/stable/myapp/ || echo "repository name already exists"

AWS\_REGION=$AWS\_REGION helm repo update

helm package k8s/petclinic\_chart

AWS\_REGION=$AWS\_REGION helm s3 push --force petclinic\_chart-${BUILD\_NUMBER}.tgz stable-petclinic

kubectl create ns petclinic-qa || echo "namespace petclinic-qa already exists"

kubectl delete secret regcred -n petclinic-qa || echo "there is no regcred secret in petclinic-qa namespace"

kubectl create secret generic regcred -n petclinic-qa \

--from-file=.dockerconfigjson=/var/lib/jenkins/.docker/config.json \

--type=kubernetes.io/dockerconfigjson

AWS\_REGION=$AWS\_REGION helm repo update

AWS\_REGION=$AWS\_REGION helm upgrade --install \

petclinic-app-release stable-petclinic/petclinic\_chart --version ${BUILD\_NUMBER} \

--namespace petclinic-qa

burada bucket name i güncellliyoruz. normalde petclininc nightly ve petclinic weekley için ayrı birer bucket ve repo kullanmak best practice tir.

bunu kullanmak için de helm s3 push da --fore komutu kullanıyoruz. çünkü aynı isimde bir repo olduğu için yapmazdı.

burada

envsubst < k8s/petclinic\_chart/values-template.yaml > k8s/petclinic\_chart/values.yaml

bu komutu neden kullanıyoruz, tekrar hatırlayalım.

dinamik hale getirmek için values-template.yaml oluşturup envsubst ile aşağıdkai variable ların son halini alıyoruz.

! values-template.yaml x 
petclinic-microservices-13-TR ) k8s ) petclinic_chart ) ! values-template.yaml 
3 
4 
5 
6 
8 
9 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
IMAGE 
TAG 
TAG 
TAG 
TAG 
TAG 
TAG 
TAG 
TAG 
TAG 
TAG 
DNS NAME: 
CONFIG SERVER: "${IMAGE TAG CONFIG SERVER}" 
DISCOVERY SERVER: "S{IMAGE TAG DISCOVERY SERVER}" 
"S{IMAGE TAG CUSTOMERS SERVICE}" 
CUSTOMERS SERVICE: 
VISITS SERVICE: "S{IMAGE TAG VISITS SERVICE}" 
VETS SERVICE: •S{IMAGE TAG VETS SERVICE}" 
API GATEWAY: "S{INAGE TAG API_6ATEWAY}" 
ADMIN SERVER: "S{IMAGE TAG ADMIN SERVER}" 
"S{IMAGE TAG HYSTRIX DASHBOARD}" 
HYSTRIx DASHBOARD: 
GRAFANA SERVICE: "S{IMAGE TAG GRAFANA SERVICEP 
PROMETHEUS SERVICE: "${IMAGE TAG PROMETHEUS SERVICE}" 
"petclinictr. cla rusway. us" 

regcred secret ını deployment ta imagePullSecrets kısmı kullanıyor.

imagePul Sec rets 
- name: regcred 

MSP 20 de tamamlandı.

git add .

git commit -m 'added build scripts for QA Environment'

git push --set-upstream origin feature/msp-20

git checkout dev

git merge feature/msp-20

git push origin dev

1. adımdayız

jenkins folder altında kayıtta bulunsun diye build-and-deploy-petclinic-on-qa-env-manually.sh

isimli bir dosya oluşturup içeriğine

PATH="$PATH:/usr/local/bin:$HOME/bin"

APP\_NAME="petclinic"

APP\_REPO\_NAME="clarusway-repo/petclinic-app-qa"

AWS\_ACCOUNT\_ID=$(aws sts get-caller-identity --query Account --output text)

export AWS\_REGION="us-east-1"

ECR\_REGISTRY="${AWS\_ACCOUNT\_ID}.dkr.ecr.${AWS\_REGION}.amazonaws.com"

echo 'Packaging the App into Jars with Maven'

. ./jenkins/package-with-maven-container.sh

echo 'Preparing QA Tags for Docker Images'

. ./jenkins/prepare-tags-ecr-for-qa-docker-images.sh

echo 'Building App QA Images'

. ./jenkins/build-qa-docker-images-for-ecr.sh

echo "Pushing App QA Images to ECR Repo"

. ./jenkins/push-qa-docker-images-to-ecr.sh

echo 'Deploying App on Kubernetes Cluster'

. ./jenkins/deploy\_app\_on\_qa\_environment.sh

echo 'Deleting all local images'

docker image prune -af

kısmını yapıştırıyoruz.

git add .

git commit -m 'added script for jenkins job to build and deploy app on QA environment'

git push --set-upstream origin feature/msp-21

git checkout dev

git merge feature/msp-21

git push origin dev

şimdi bir jenkins job oluşturacaz.

bu ilk kez uygulamayı manuel kurmak için yazacağımız bir job. bundan sonra cluster hep açık kalacak ve weekly pipeline çalışacak.

bundan sonra release branchına geçiyoruz çünkü uygulama ete kemiğe büründü. dev branchı tabi ki bir yandan devam edecek.

git checkout release

git merge dev

git push origin release

bu komutla release branchı oluşturup güncelledik. şimdi build-and-deploy-petclinic-on-qa-env

isimli job ımızı oluşturalım.

bu kez release branch olacak job SCM de.

execute shell e de yukarıdaki komutu yapıştırdık.

en sondaki docker işmage prune komutu image ler birikmesin diye silmek için.

values-template.yaml içine bakalım workspace teki:

[jenkins@jenkins-server petclinic chartl$ cat values-template.yaml 
INAGE TAC 
INAGE TAG 
INAGE TAC 
IRAGE TAG 
INAGE TAG 
IRAGE TAG 
NAGE TAG 
NAGE TAG 
INAGE TAG 
TAG 
DNS NAME: 
IRAGE TAG 
HAGE TAG 
IRAGE TAC 
TAC 
IRAGE TAC 
HAGE TAC 
IRAGE TAG 
TAC 
NAGE TAC 
IRAGE TAC 
CONFIG SERVER: CONFIG SERVER}" 
-_DISCOVÉRY_SERVER: • 
ECUSTOMERS SERVICE: SERVICE}" 
_VISITS SERVICE: "${IMAGE TAG VISITS SERVICE}" 
VETS SERVICE: "S{IRAGE TÄG vÉTS SERVICE} " 
-_API_GATEWAY: "S{IRAGE tAG ÄPI GÄTEWAY}" 
_ADMIN_SERVER: 
eHYSTRIX DASHBOARD: "${IMAGE 
GRAFAWCSERVICE: "${IMAGE TÄG GRAFAUA SERVICE}" 
"${IMÄGE 
petctinic_chartlS cat values.yanl 
_ CONFIG_SERVER: 
DISCOVERY SERVER: 
CUSTWERS¯SERVICE: "046402772087.dkr.ecr.us-east-l.amazonaws.com/clarusway-repo/petclinic-app-qa:customers-service-qa-v2.1.2-bl• 
ZVISITS_SEÄVICE: "046402772087.dkr.ecr.us-east-l.amazonaws.com/clarusway- repo/petcUnic-app-qa:visits-service-qa-v2.1.2-b1" 
VETS_SERVICE: "046402772e87.dkr.ecr.us-east-1.amazonaws.com/c1arusway-rep0/petctinic-app-qa:vets-service-qa-v2.1.2-b1• 
_API GATEWAY: 
ADMiN SERVER: '046402772087.dkr.ecr.us-east-1.amazonaws.com/clarusway-repo/petclinic-app-qa:admin-server-qa-v2.1.2-bI• 
¯HYSTRiX DASHBOARD: 0@46402772087.dkr.ecr.us-east-l.anazonaws.com/clarusway- repo/petclinic-app-qa:hystrix-dashboard-qa-v2.1.2-b1• 
-_GRAFANA-SERVICE: "046402772087-dkr.ecr.us-east-l.amazonaws.com/clarusway- repo/petclinic-app-qa:grafana-service" 
PRWETHEUS SERVICE: 
petclinic_chartlS 

image lerin son halini almışız.

deleted : 
deleted : 
sha256 : 7515debb58fe219a634f97ba47489c2eb5b1b4736aa1a2fff256f1d28bfbeecb 
sha256 : f1417ff83b319fbdae6dd9cd6d8c9c88eø2dcd75ecf6ec2a1c8c6894681cf2bs 
Total reclaimed space: I. 79163 
Finished: SUCCESS 

job başarılı çalıştı.

kubectl ile ns deki objectlere baktığımızda ingress-nginx e bakıyoruz:

CLASS HOSTS 
api-gateway nginx petclinic . emreaydeniz.com 
[jenkins@jenkins-server 
@9A901 w o 
[jenkins@jenkins-server kubectl get ingress 
-n petclinic-qa 
Ln 2765, Col 20 
ADDRESS 
a4a8c28b1Ø73f43198fØdd28dffe1dca-6362øu72 
. us -east-I. elb 
. amazonaws 
. com 
PORTS 
80 
AGE 
20m 

buradaki dns name nereden geliyor?

@ README.md 
api-gateway-ingress.yaml X 
'services-with-db > k8s > petclinic_chart > templates > ! 
a pi-gateway-ingress. 
kompose . Image-pull-secret: regcred 
kompose. service. type: nodeport 
7 
8 
le 
11 
12 
13 
14 
15 
16 
17 
18 
19 
2e 
21 
kompose . service 
.Va1ues.DNS NAMI 
. expose : 
kompose . service . nodeport . port • 
. "3øoe1 " 
kompose . version : 
1.28.e (c4137Ø12e) 
creationT imestamp : 
null 
labels : 
io. kompose. service: api-gateway 
name: api -gateway 
spec : 
ingressC1assName: nginx 
rules: 
host: '{{ . Values. DNS NAME 
http : 
paths : 
backend: 

api-gatewayy-ingress.yaml dan geldi.

buradaki DNS NAme i da values-template values.yaml a overwrite olunca alıyor.

igress controller yüklediğimiz için ELB miz de oluşmuş:

EC2 > Load balancers 
Load balancers (1) 
Elastic Load Balancing scales pur load balancer capacity automatically in response to changes in incoming traffic 
C Actions 
Q Find resources by attribute or tog 
Name 
a4a8c28b1073f43198f0 
dd28dffe1dca 
O load balancers selected 
Select a load balancer above. 
DNS name 
3 a4a8c28b1073f43198f0d... 
State 

şimdi route 53 e hosted zone a giriyouz.

Create record 
Quick create record 
Record I 
Record name 
petclinic 
Info 
.emreaydeniz.com 
Switch to wizard 
Delete 
Record type Info 
A — Routes traffic to an IPv4 address and some AW... 
Keep blank to create a record for the domain. 
O Alias 
Route traffic to Info 
Alias to Application and Classic Load Balancer 
US East (N. Virginia) [us-east- 1] 
Q dualstack.a4a8c28b1073f43198f0dd28dffe1dca-636204472.us-east-1.e1b.amazonaws.corq 
Alias hosted zone ID: Z35SXDOTRQ7X7K 
Routing policy Info 
Simple routing 
Evaluate target health 
O Yes 
x 
Add another record 
Cancel 
Create reco 

buraya values-tmplate teki dns i yazmamız gerekiyor. bu şekilde record u oluşturuyoruz.

1. adıma geçtik

weekly pipeline hazırıkları bitti, uyuglamayı manuel olarak job la yaağa kaldırdık. şimdi pipeline ı oluşturabiliriz.

tekrar dev branchına gidip msp-22 branchı oluşturalım.

git checkout dev

git branch feature/msp-22

git checkout feature/msp-22

jenkins klasörü altında jenkinsfile-petclinic-weekly-qa

isimli bir jenkinsfile oluşturacaz.

pipeline {

agent any

environment {

PATH=sh(script:"echo $PATH:/usr/local/bin:$HOME/bin", returnStdout:true).trim()

APP\_NAME="petclinic"

APP\_REPO\_NAME="clarusway-repo/petclinic-app-qa"

AWS\_ACCOUNT\_ID=sh(script:'export PATH="$PATH:/usr/local/bin" && aws sts get-caller-identity --query Account --output text', returnStdout:true).trim()

AWS\_REGION="us-east-1"

ECR\_REGISTRY="${AWS\_ACCOUNT\_ID}.dkr.ecr.${AWS\_REGION}.amazonaws.com"

}

stages {

stage('Package Application') {

steps {

echo 'Packaging the app into jars with maven'

sh ". ./jenkins/package-with-maven-container.sh"

}

}

stage('Prepare Tags for Docker Images') {

steps {

echo 'Preparing Tags for Docker Images'

script {

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-admin-server/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_ADMIN\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:admin-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-api-gateway/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_API\_GATEWAY="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:api-gateway-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-config-server/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_CONFIG\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:config-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-customers-service/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_CUSTOMERS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:customers-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-discovery-server/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_DISCOVERY\_SERVER="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:discovery-server-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-hystrix-dashboard/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_HYSTRIX\_DASHBOARD="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:hystrix-dashboard-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-vets-service/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_VETS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:vets-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

MVN\_VERSION=sh(script:'. ${WORKSPACE}/spring-petclinic-visits-service/target/maven-archiver/pom.properties && echo $version', returnStdout:true).trim()

env.IMAGE\_TAG\_VISITS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:visits-service-qa-v${MVN\_VERSION}-b${BUILD\_NUMBER}"

env.IMAGE\_TAG\_GRAFANA\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:grafana-service"

env.IMAGE\_TAG\_PROMETHEUS\_SERVICE="${ECR\_REGISTRY}/${APP\_REPO\_NAME}:prometheus-service"

}

}

}

stage('Build App Docker Images') {

steps {

echo 'Building App Dev Images'

sh ". ./jenkins/build-qa-docker-images-for-ecr.sh"

sh 'docker image ls'

}

}

stage('Push Images to ECR Repo') {

steps {

echo "Pushing ${APP\_NAME} App Images to ECR Repo"

sh ". ./jenkins/push-qa-docker-images-to-ecr.sh"

}

}

stage('Deploy App on Kubernetes Cluster'){

steps {

echo 'Deploying App on Kubernetes Cluster'

sh '. ./jenkins/deploy\_app\_on\_qa\_environment.sh'

}

}

}

post {

always {

echo 'Deleting all local images'

sh 'docker image prune -af'

}

}

}

daha sonra push ediyoruz.

git add .

git commit -m 'added jenkinsfile petclinic-weekly-qa for release branch'

git push --set-upstream origin feature/msp-22

git checkout dev

git merge feature/msp-22

git push origin dev

tekrar release branchına gidip güncelleme yapıyoruz.

dosyalar hazır artık petclinic-weekly-qa

isimli pipeline oluşturacaz.

Branch Specifier (blank for •any') 
Yrelease 
Add Branch 
Repository browser 
(Auto) 
Additional Behaviours 
Add 
Script Path ? 
jenkins/jenkinsfile-petclinic-weekly-qe 

branch ve jenkinsfile path ini değiştirmeyi unutmuyoruz.

crontab ı şu şekilde oluşturduk:

Build Triggers 
o 
Build after other projects are built 
Build periodically ? 
Schedule ? 
59 23 o 
A Spread load evenly by using 'H 23 * * 
O' rather than '59 23 ** O' 
Would last have run at Sunday, June 4, 2023 at 11:59:01 PM Coordinated Universal T 
2023 at 1 1:59:01 PM Coordinated Universal Time. 

Apply/Save/Build Now

devops mühendisi sistemi kurar ve ve zamanıın çoğu troubleshooting ile geçer.

biz manuel kurulum yaptığımız job ı çalıştırmadan direk bu pipeline ı çalışıtrırsak yine de çalışırdı ancak biz gerçeği simüle etmek için ilk manuel kurulumu yaptık.

pipeline çalıştı:

Pipeline petclinic-weekly-qa 
Stage View 
Average stage times: 
(Average full run time: —5min 4s) 
Hiz ID 
Changes 
12:45 
Permalinks 
Declarative: 
Checkout SCM 
589ms 
S89ms 
Package 
Application 
3min 12s 
3min 12s 
Prepare Tags 
for Docker 
Images 
Build App 
Docker Images 
Sis 
51s 
Push Images 
to ECR Repo 
34s 
34s 
Deploy App on 
Kubernetes 
Cluster 
9s 
9s 
Declarative: 
Post Actions 

peki biz neden api-gateway-ingress.yaml da host verdik:

spec : 
ingressC1assName: nginx 
rules: 
host: '{{ . Values. DNS NAME 
http : 
paths : 
backend: 
service : 
name: api-gateway 
port : 
number : 8080 
path: / 
pathType: Prefix 
status : 
loadBa1ancer: 

host verdiğimiz için curl komutuyla elb dnsini çekemem, elb nin dns iyle curl komutu çalışmaz:

[jenkins@jenkins-server kubectl get ingress •n petclinic-qa 
CLASS HOSTS 
ADDRESS 
api -gateway nginx petclinictr. clarusway. us af140fb252d284eabb4ee7b7381f1a79-58702787. us -east-l.elb.amazonaws.cwn 
[jenkins@jenkins-server -IS curl af140fb252d284eabb4ee7b7381f1a79-587e2787.us-east-1.etb.amazonaws.corn 
Not 
ebodp 
Not 
</bodp 
1 
[jenkins@jenkins-server —IS 
PORTS 
AGE 

hostu çıkaralım:

spec : 
ingressCtassNæne: ngunx 
rules : 
- Ittp: 
paths : 
- backend: 
service : 
name: ani- oatewav 

[ j enkins@jenkins-server 
CLASS 
api -gateway nginx 
[jenkins@jenkins-server — 
kubectl get ingress -n petclinic-qa 
ADDRESS 
af14efb252d284eabb4ee7b7381f1a79-58702787. us east- 1. elb. . com 
PORTS 
8e 
AGE 

host gitti

şimdi curl ile elb dns ini girersek gelir.

curl ile petclinic.emreaydeniz.com yazarsak yine de geliyor. yani subdomaine bir etkisi yok. o zaman hostun etkisi ne?

güvenlik. yani kiimse elb nin dns iyle siteme ulaşılsın istemiyorum. sdece domain name ile ulaşılsın istiyorum bu nedenle host parametresini ingress.yaml dosyasına ekliyorum.

cluster ı kapatmak için

sudo su - jenkins

eksctl delete cluster -f cluster.yaml

bu da uzun sürecek, beklemek gerekiyor.