

Dang Quoc Khoa

SD4591

PI-SHAPED

Just inform that in this section, we have some lines of codes that this document does not provide in detail because we have done it on the first program called "*Practical DevOps for Devs*". This document will focus on the latest changes.

1. Manage K8s Application

DevOps for Devs:

You would need to have a repository for the MSA application.

You can clone the source code from <https://github.com/nashtech-garage/kubernetes/tree/master/src>

App repository pattern: https://github.com/<your-github-name>/<staffcode_msa>

Example: https://github.com/hoanglecao/sd0660_msa

You also need another repository for infrastructure. Provision codes and manifest files will be stored in this repository.

App repository pattern: https://github.com/<your-github-name>/<staffcode_aws_infrastructure>

Example:

https://github.com/hoanglecao/sd0660_aws_infrastructure

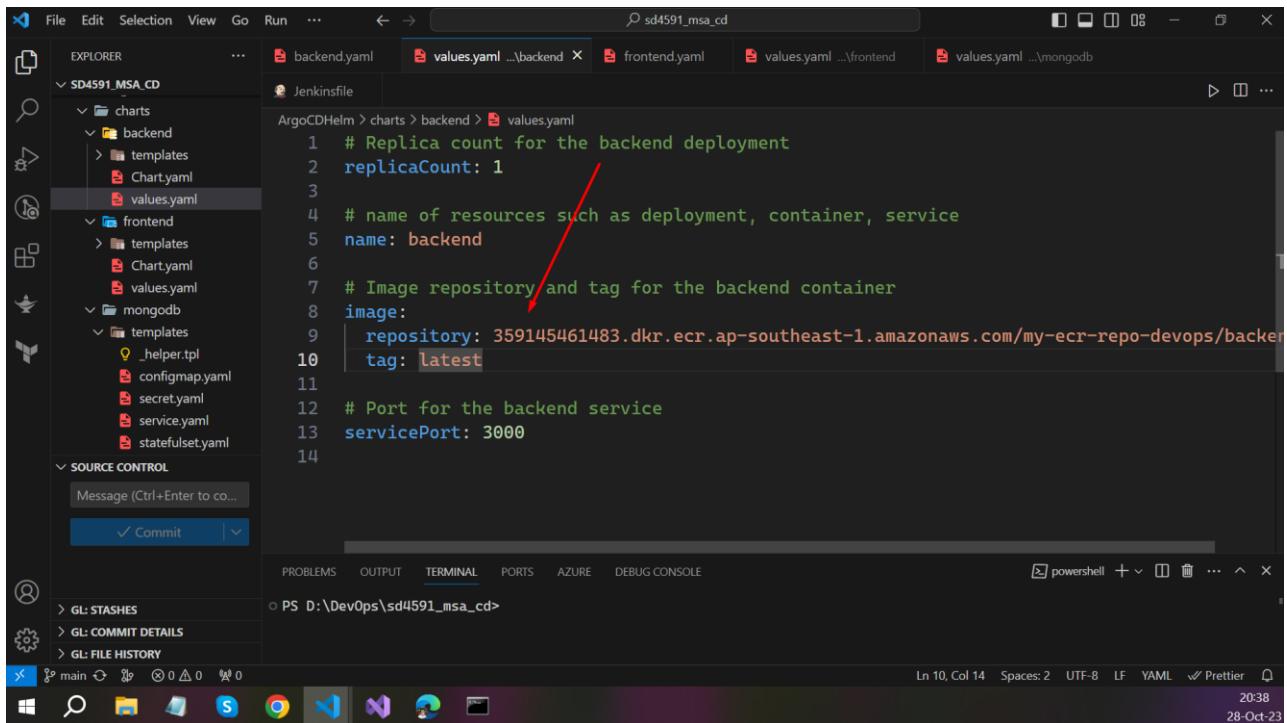
DevOps for Devs:

Create K8s manifest.

Pi-sharp:

Use helm-charts to manage K8s manifest.

After building and pushing the image to ECR (I have done requirements of 2-3-4-5 and then back to this requirement), we apply helm to project and update accordingly image information for each service.

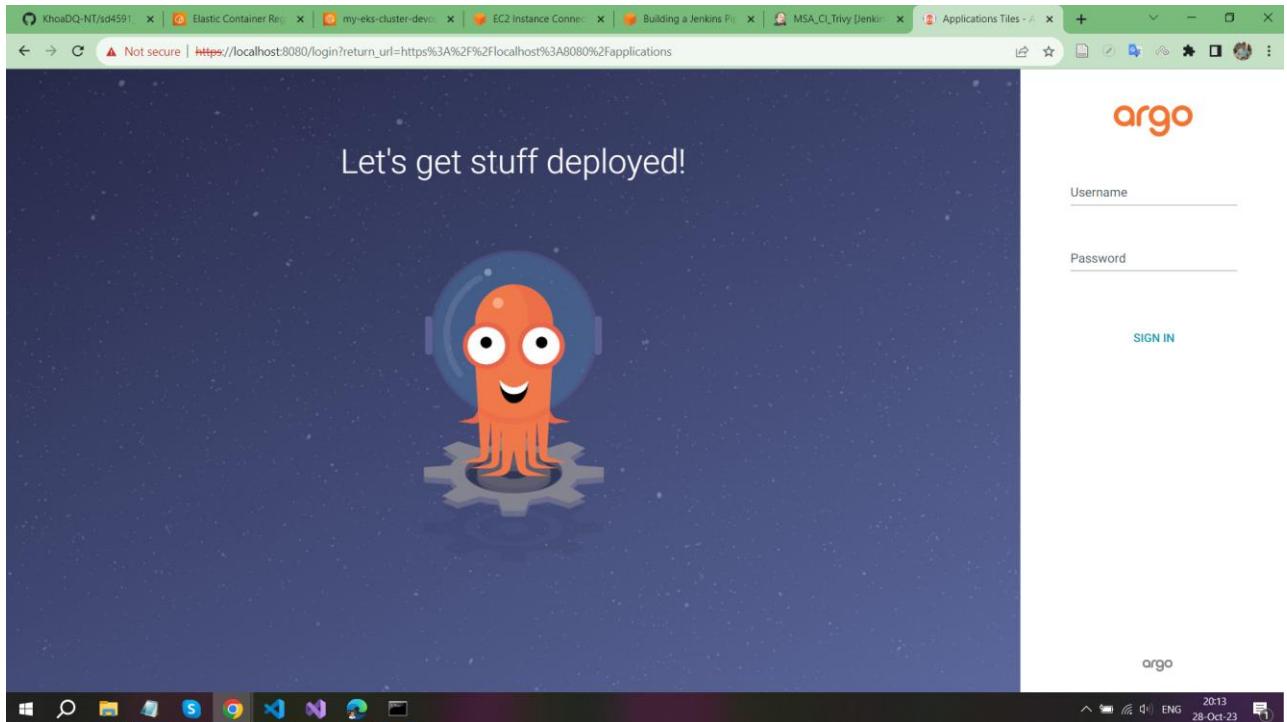


```
# Replica count for the backend deployment
replicaCount: 1

# name of resources such as deployment, container, service
name: backend

# Image repository and tag for the backend container
image:
  repository: 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/backend
  tag: latest

# Port for the backend service
servicePort: 3000
```



The screenshot shows the Argo UI interface. On the left, there's a sidebar with navigation links: 'Applications' (selected), 'Settings', 'User Info', and 'Documentation'. Below these are sections for 'Favorites Only', 'SYNC STATUS' (Unknown: 0, Synced: 0, OutOfSync: 1), and 'HEALTH STATUS' (Unknown: 0, Progressing: 0, Suspended: 0, Healthy: 0, Degraded: 0, Missing: 1). The main content area is titled 'Applications' and shows a single application card for 'argocd-helm-app'. The card details are as follows:

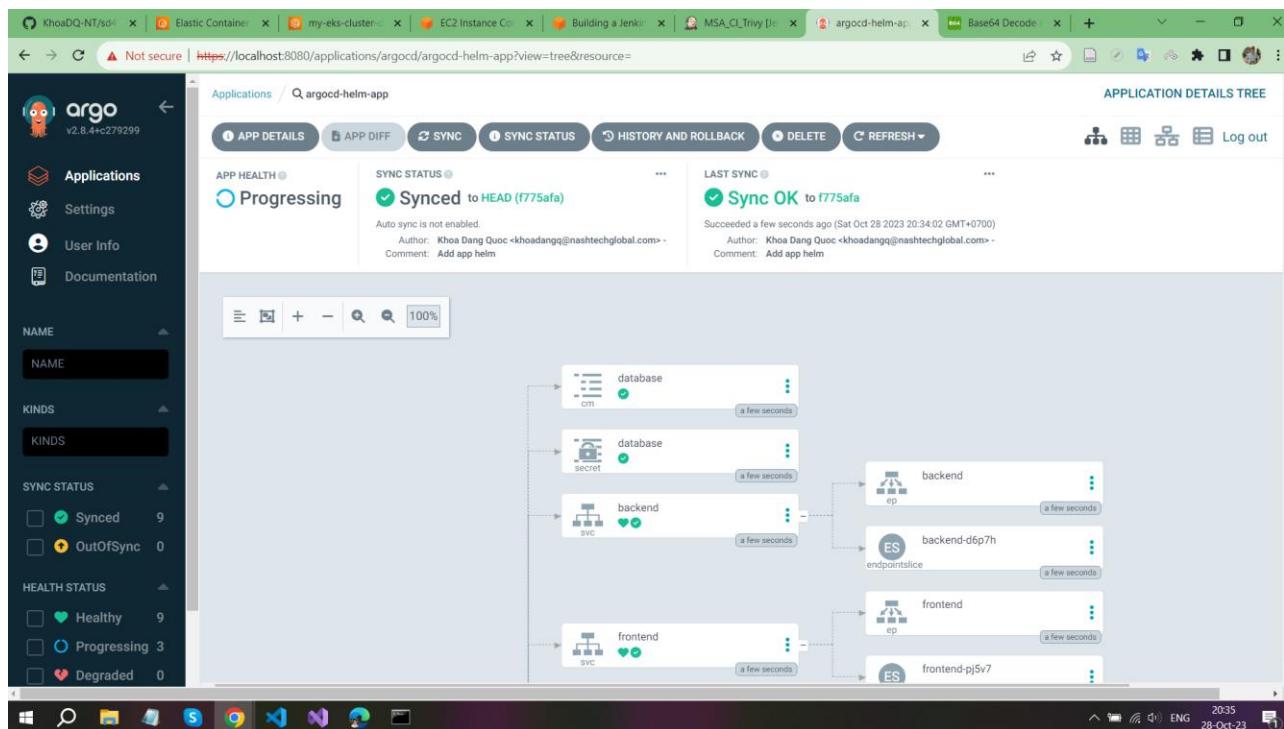
Project:	argocd-helm
Labels:	
Status:	Missing OutOfSync
Repository:	https://github.com/KhoaDQ-NT/sd4591_m...
Target Re...	HEAD
Path:	./ArgoCDHelm
Destinati...	in-cluster
Namespa...	app-argocd-helm
Created At:	10/28/2023 20:32:53 (a few seconds ago)

At the bottom of the card are three buttons: 'SYNC', 'REFRESH', and 'DELETE'.

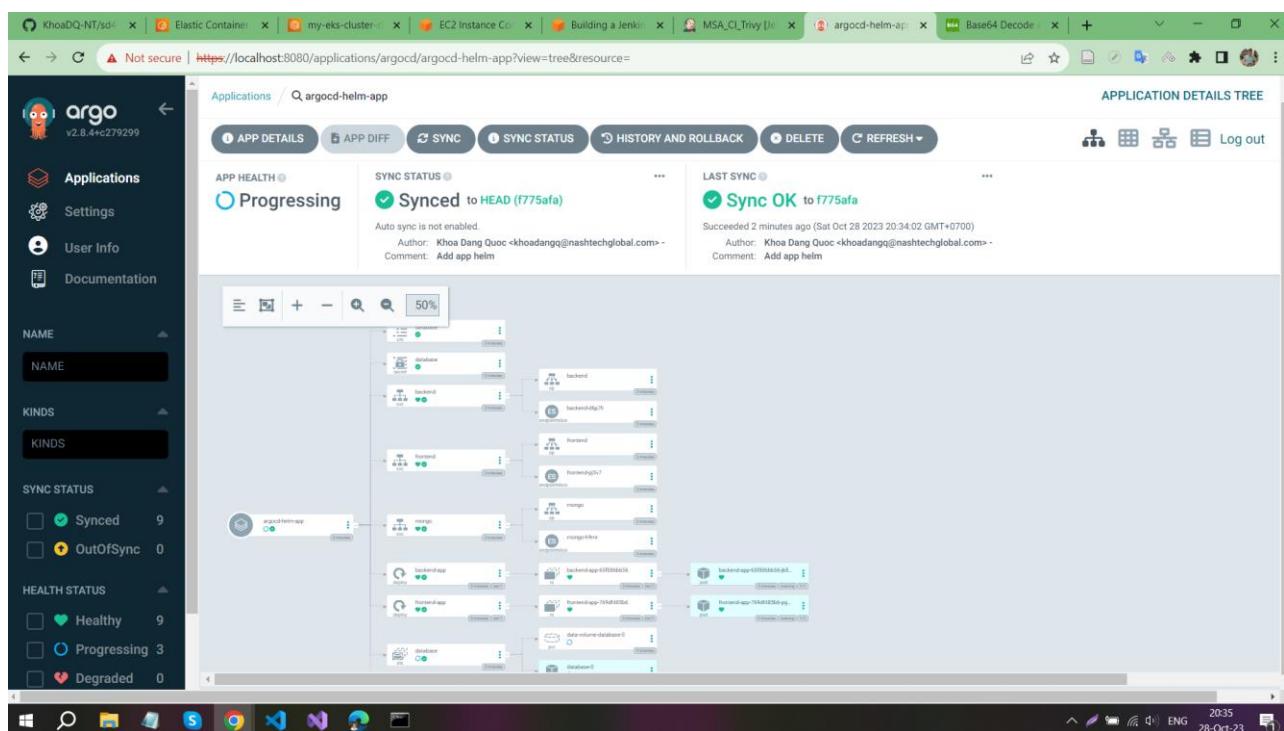
This is a detailed view of the 'argocd-helm-app' application card. The card header includes the application name, a star icon for favoriting, and a refresh icon. The card displays the following information:

Project:	argocd-helm
Labels:	
Status:	Progressing Synced
Repository:	https://github.com/KhoaDQ-NT/sd4591_m...
Target Re...	HEAD
Path:	./ArgoCDHelm
Destinati...	in-cluster
Namespa...	app-argocd-helm
Created At:	10/28/2023 20:32:53 (a minute ago)
Last Sync:	10/28/2023 20:34:02 (a few seconds ago)

At the bottom of the card are three buttons: 'SYNC', 'REFRESH', and 'DELETE'.



My application is pending for a long time because the EBS does not have permission on volume, after investigating more on EBS driver and configuring exactly policy, it works OK



 helm-app	☆
Project:	argocd-helm
Labels:	
Status:	❤️ Healthy ✅ Synced
Repository:	https://github.com/KhoaDQ-NT/sd4591_m...
Target Re...	HEAD
Path:	ArgoCDHelm
Destinati...	in-cluster
Namespa...	argocd
Created At:	10/28/2023 21:52:10 (27 minutes ago)
Last Sync:	10/28/2023 22:12:42 (6 minutes ago)
 SYNC	 REFRESH
	 DELETE

The screenshot shows the ArgoCD interface for the 'helm-app' repository. The top navigation bar includes tabs for 'APP DETAILS', 'APP DIFF', 'SYNC', 'SYNC STATUS', 'HISTORY AND ROLLBACK', 'DELETE', and 'REFRESH'. The 'SYNC STATUS' tab is active, displaying a green 'Synced' status with a timestamp of 'Sync OK to f775afe' and a note: 'Succeeded 7 minutes ago (Sat Oct 28 2023 22:12:02 GMT+0700)'. Below this, the 'Auto sync is enabled' and 'Author: Khoa Dang Doan - khoadangq@nashtechglobal.com' are listed. A comment 'Add app helm' is also present.

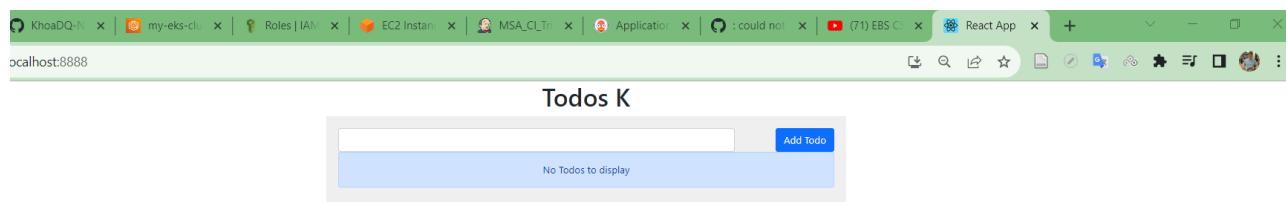
The main area displays the application tree under the 'helm-app' repository. The tree structure includes:

- secret**: Synced 27 minutes ago.
- backend**: Synced 27 minutes ago. Contains an endpoint named **backend-6c27t**.
- frontend**: Synced 27 minutes ago. Contains two endpoints: **frontend-f62h6** and **mongo**.
- mongo**: Synced 27 minutes ago. Contains an endpoint named **mongo-sa5dm**.
- helm-app**: Synced 27 minutes ago. Contains:
 - backend-app**: Deployed 27 minutes ago (rev:1). Contains a replica set named **backend-app-65f86bbb56**.
 - Frontend-app**: Deployed 27 minutes ago (rev:1). Contains a replica set named **frontend-app-769df485b6**.
 - sts**: Synced 27 minutes ago. Contains a deployment named **database**.
 - database**: Synced 27 minutes ago. Contains a persistent volume claim named **pvc** and a pod named **database-0**.
- controllerRevision**: Synced 27 minutes ago.

Each node in the tree has a green heart icon indicating it is healthy. The entire sync process took 7 minutes.

```
C:\Users\khoadangq>kubectl get services -n argocd
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP  PORT(S)          AGE
argocd-applicationset-controller  ClusterIP  172.20.132.147 <none>        7000/TCP,8080/TCP  131m
argocd-dex-server    ClusterIP  172.20.19.37  <none>        5556/TCP,5557/TCP,5558/TCP  131m
argocd-metrics     ClusterIP  172.20.86.187 <none>        8082/TCP          131m
argocd-notifications-controller-metrics ClusterIP  172.20.201.16 <none>        9001/TCP          131m
argocd-redis       ClusterIP  172.20.207.214 <none>        6379/TCP          131m
argocd-repo-server  ClusterIP  172.20.234.23  <none>        8081/TCP,8084/TCP  131m
argocd-server       ClusterIP  172.20.1.208  <none>        80/TCP,443/TCP    131m
argocd-server-metrics ClusterIP  172.20.82.210 <none>        8083/TCP          131m
backend          ClusterIP  172.20.218.205 <none>        3000/TCP          27m
frontend          ClusterIP  172.20.27.35  <none>        3000/TCP          27m
mongo            ClusterIP  172.20.203.48  <none>        27017/TCP         27m
```

After all services were “running” we need to port forward to using application on local



2. Provision AWS resources

Use <https://github.com/nashtech-garage/terraform-demo> as a reference.

Use Terraform to provision VPC, EC2, ECR and EKS on AWS [DevOps for Devs:](#)

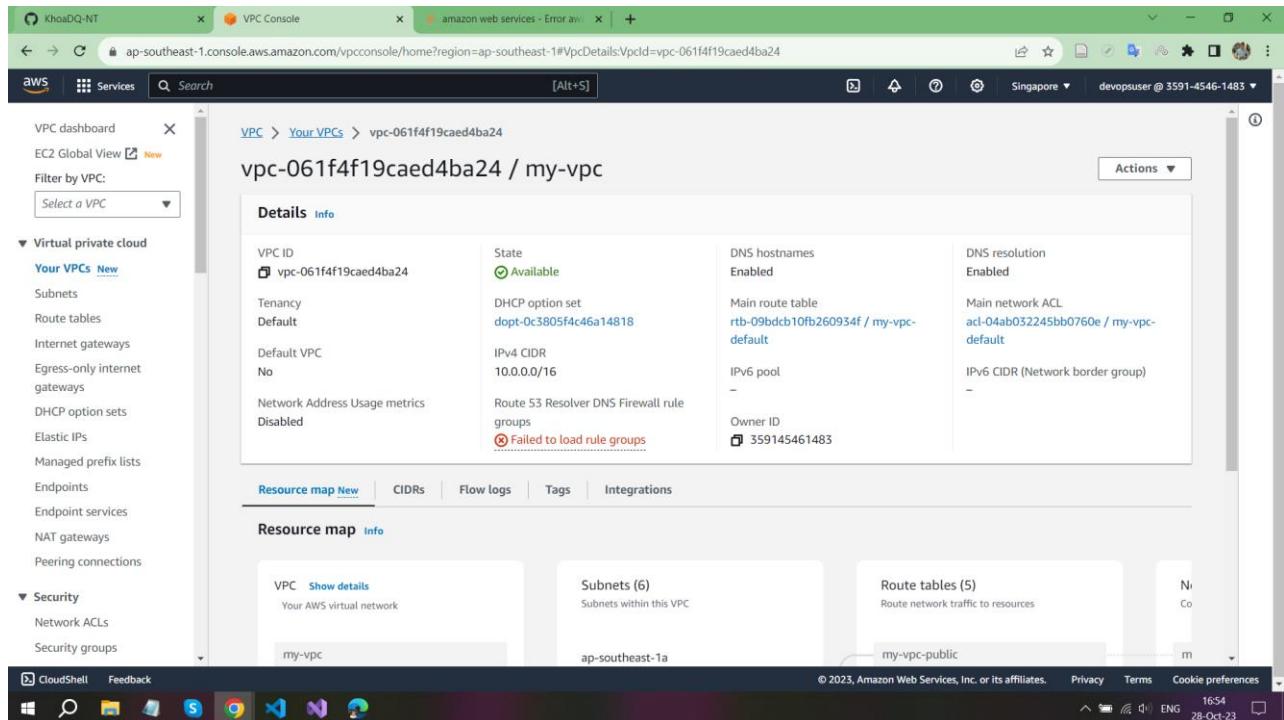
- EKS can created in one zone and
- Can use default VPC for worker nodes

[Pi-sharp:](#)

- EKS must be created in High Availability (Use Multi-AZs)
- Create your VPC and use it for worker nodes

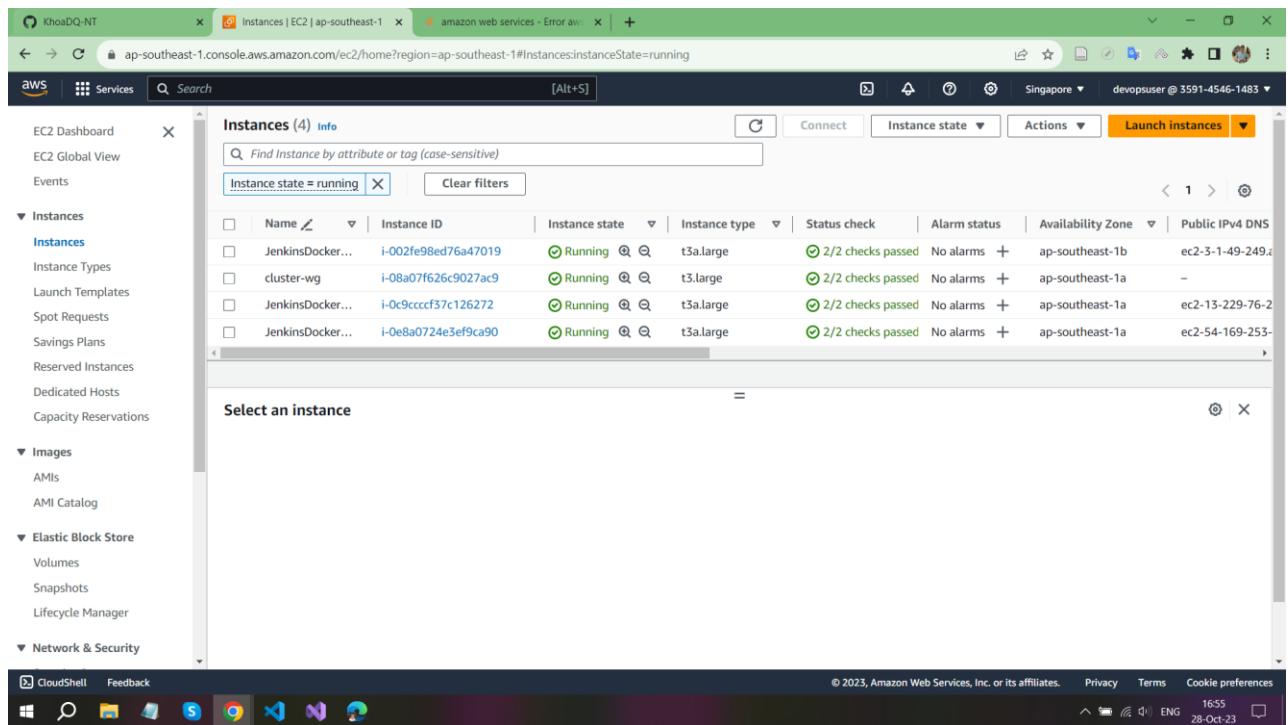
I need to create a custom VPC as requirement, so that I have made some changes on my Infrastructure source code for using Terraform to deploy all necessary AWS services

The image below shows you my custom VPC



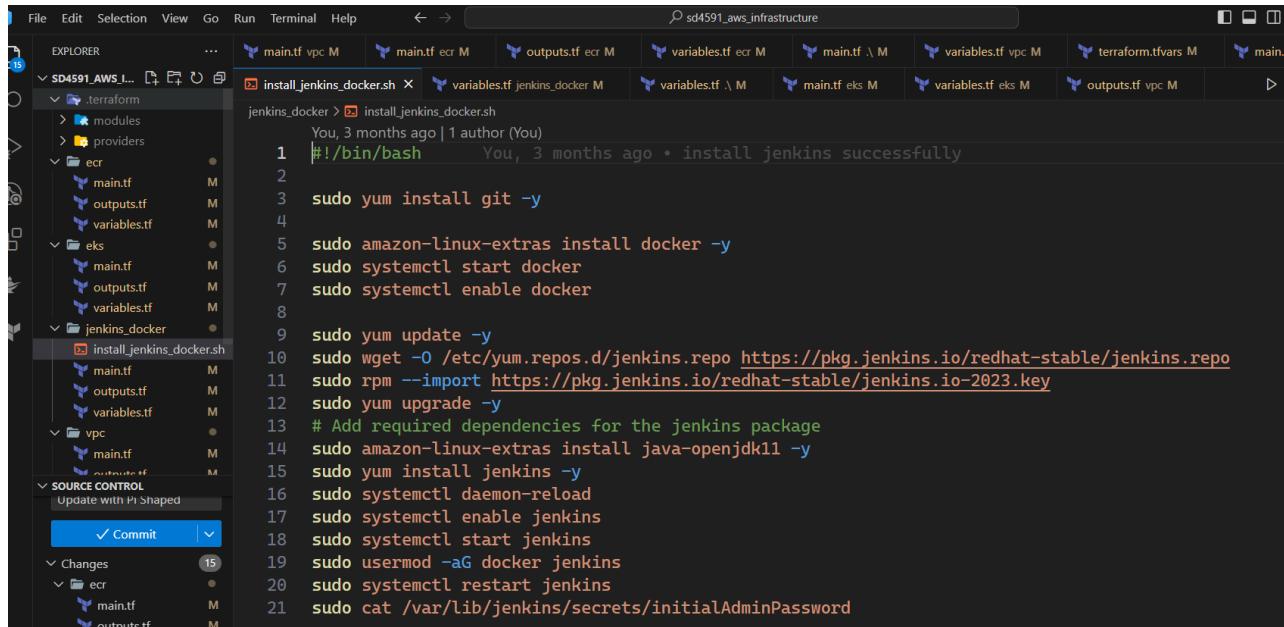
The EC2 instances will be deployed in multiple zones within a region, that will make our instances more “High availability”, the ability to operate continuously without failing for a designated period of time. The main code changes on creating EC2 is showed below

My subnets on multiple zones



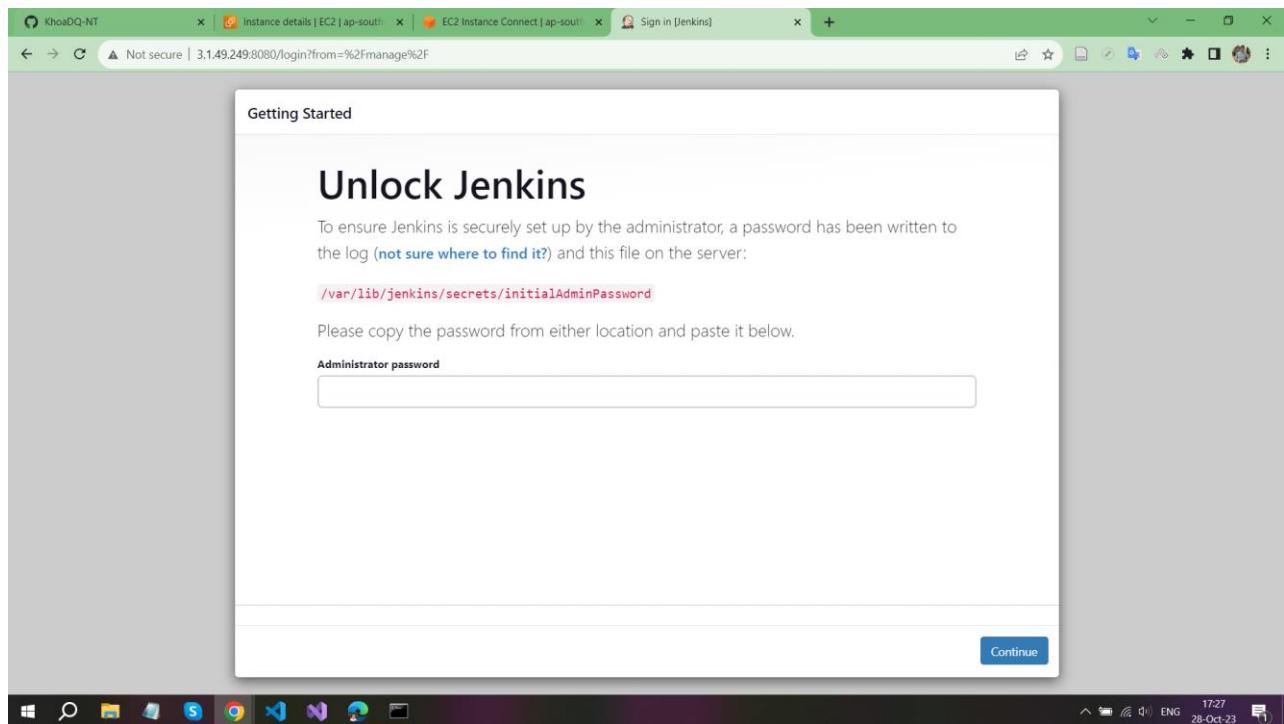
3. Installation

You need to have Docker and Jenkins servers installed on EC2.



The terminal window shows the execution of a Jenkins Docker installation script named `install_jenkins_docker.sh`. The script contains the following commands:

```
#!/bin/bash
sudo yum install git -y
sudo amazon-linux-extras install docker -y
sudo systemctl start docker
sudo systemctl enable docker
sudo yum update -y
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
sudo yum upgrade -y
# Add required dependencies for the jenkins package
sudo amazon-linux-extras install java-openjdk11 -y
sudo yum install jenkins -y
sudo systemctl daemon-reload
sudo systemctl enable jenkins
sudo systemctl start jenkins
sudo usermod -aG docker jenkins
sudo systemctl restart jenkins
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



The browser window displays the Jenkins 'Unlock Jenkins' setup page. The page title is 'Getting Started' and the main heading is 'Unlock Jenkins'. It instructs the user to copy the password from the file `/var/lib/jenkins/secrets/initialAdminPassword`. A text input field is provided for the administrator password, and a 'Continue' button is at the bottom right.

The screenshot shows a browser window with multiple tabs at the top: 'KhoaDQ-NT', 'Instance details | EC2 | ap-sout...', 'VPC Console', 'EC2 Instance Connect | ap-sout...', and 'Dashboard [Jenkins]'. The 'Dashboard [Jenkins]' tab is active. The address bar shows 'Not secure | 3.149.249:8080'. The main content area is the Jenkins dashboard. On the left, there's a sidebar with links: '+ New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. Below these are two dropdown menus: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area has a heading 'Welcome to Jenkins!'. It says, 'This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.' It features three main buttons: 'Create a job', 'Set up a distributed build', and 'Start building your software project'. Under 'Set up a distributed build', there are three sub-options: 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. At the bottom right, it says 'REST API Jenkins 2.414.3' and shows system status icons.

After Jenkins have been installed, we need to add some needed blugins and create some necessary credentials.

4. Setup Jenkins pipeline for CI

Use <https://github.com/nashtech-garage/devops-ci-cd> as a reference.

DevOps for Devs:

- You can use normal Jenkins pipeline or create Jenkins pipeline using Groovy.

Pi-sharp:

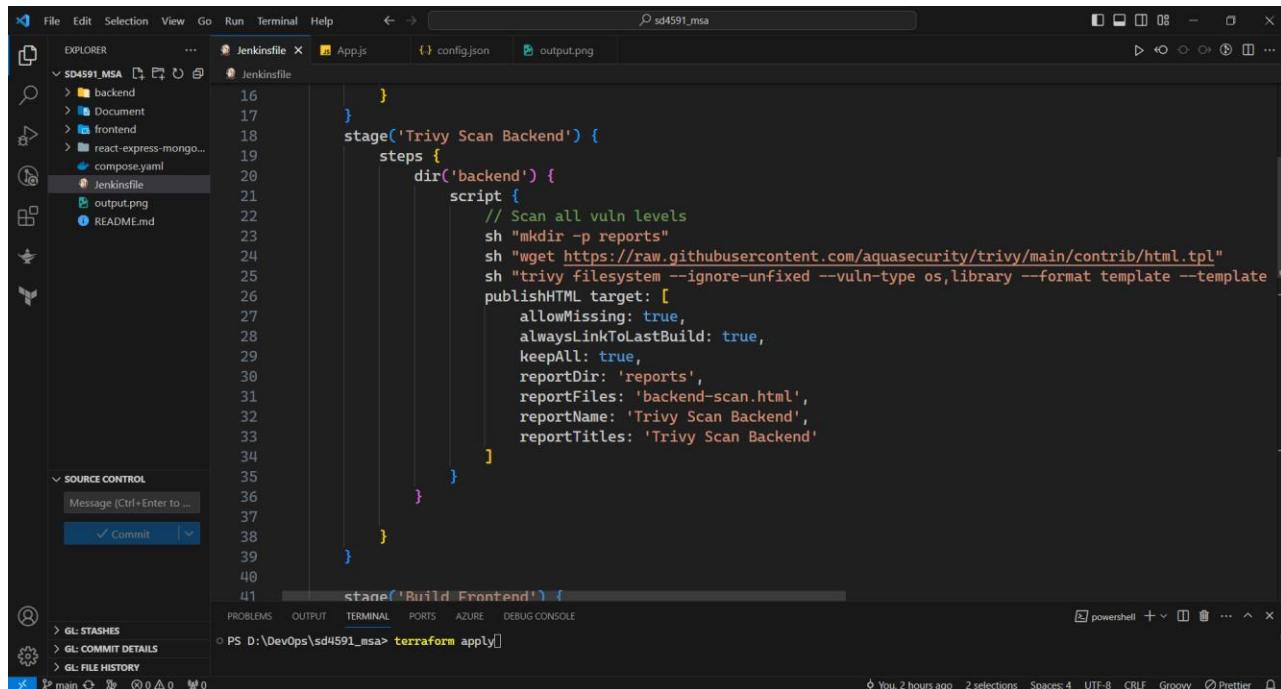
- Use Trivy in Jenkins CI pipeline.

Trivy should be installed before running the Jenkins CI pipeline

```
curl -sfL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/install.sh |  
sudo sh -s -- -b /usr/local/bin v0.18.3
```

```
curl -sfL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl >  
html.tpl
```

Besides, I have changed MSA source code to add more stages related to Trivy and use it on Jenkins CI



The screenshot shows the Jenkinsfile code in a VS Code editor. The code defines a pipeline with two main stages: 'Trivy Scan Backend' and 'Build Frontend'. The 'Trivy Scan Backend' stage uses the Trivy tool to scan the backend directory for vulnerabilities. It creates a 'reports' directory, downloads a template, runs the Trivy command, and publishes the results as an HTML report. The 'Build Frontend' stage uses Terraform to apply changes to the infrastructure. The Jenkinsfile also includes imports for 'app' and 'trivy' and defines some global variables.

```
import('app').load()  
import('trivy').load()  
  
stage('Trivy Scan Backend') {  
    steps {  
        dir('backend') {  
            script {  
                // Scan all vuln levels  
                sh "mkdir -p reports"  
                sh "wget https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl"  
                sh "trivy filesystem --ignore-unfixed --vuln-type os,library --format template --template publishHTML target: [  
                    allowMissing: true,  
                    alwaysLinkToLastBuild: true,  
                    keepAll: true,  
                    reportDir: 'reports',  
                    reportFiles: 'backend-scan.html',  
                    reportName: 'Trivy Scan Backend',  
                    reportTitles: 'Trivy Scan Backend'  
                ]  
            }  
        }  
    }  
}  
  
stage('Build Frontend') {  
    steps {  
        sh "terraform apply"  
    }  
}
```

KhoaDQ-NT/sd4591_msa | Elastic Container Registry | VPC Console | EC2 Instance Connect | ap | Building a Jenkins Pipeline | MSA_CI_Trivy #14 Console

Not secure | 3.1.49.249.8080/job/MSA_CI_Trivy/14/console

Dashboard > MSA_CI_Trivy > #14

Status

Changes

Console Output

View as plain text

Edit Build Information

Polling Log

Timings

Git Build Data

Trivy Scan Backend

Trivy Scan Frontend

Thread Dump

Pause/resume

Replay

Pipeline Steps

Workspaces

Previous Build

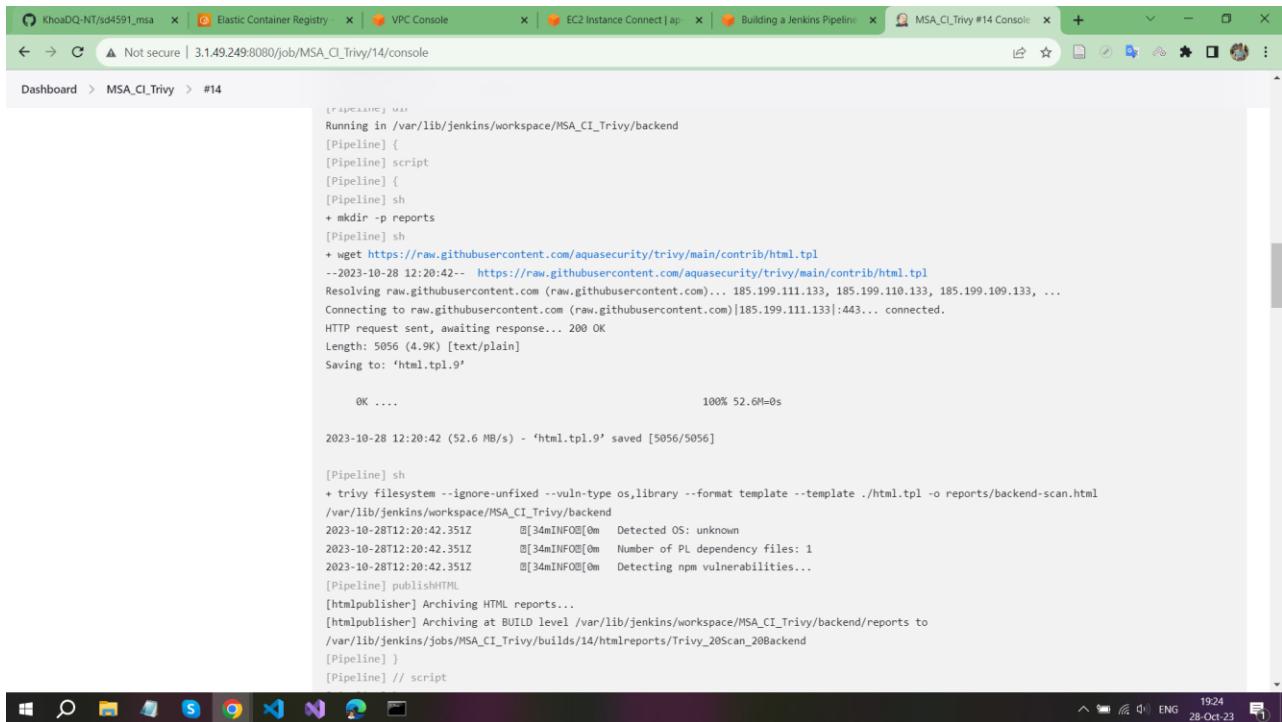
Console Output

```
Started by GitHub push by KhoaDQ
Obtained Jenkinsfile from git https://github.com/KhoaDQ-NT/sd4591\_msa
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/MSA_CI_Trivy
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
The recommended git tool is: git
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/MSA_CI_Trivy/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/KhoaDQ-NT/sd4591\_msa # timeout=10
Fetching upstream changes from https://github.com/KhoaDQ-NT/sd4591\_msa
> git --version # timeout=10
> git --version # 'git' version 2.40.1'
> git rev-parse refs/remotes/origin/main{commit} # timeout=10
Checking out Revision 000192cf4bb86c574197d27b2e79a48c663efa39 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 000192cf4bb86c574197d27b2e79a48c663efa39 # timeout=10
Commit message: "fix"
> git rev-list --no-walk 8dca880fb8118fac7eeeeeefddbe3abb52d0ffadf # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
```

Build backend

```
Running in /var/lib/jenkins/workspace/MSA_CI_Trvy/backend
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ docker build -t 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/backend:latest .
Sending build context to Docker daemon 142.8kB
```

Use Trivy scanner on Backend



```
Running in /var/lib/jenkins/workspace/MSA_CI_Trvy/backend
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ mkdir -p reports
[Pipeline] sh
+ wget https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl
--2023-10-28 12:20:42-- https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.110.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5056 (4.9K) [text/plain]
Saving to: 'html.tpl.9'

OK ....
100% 52.6M=0s

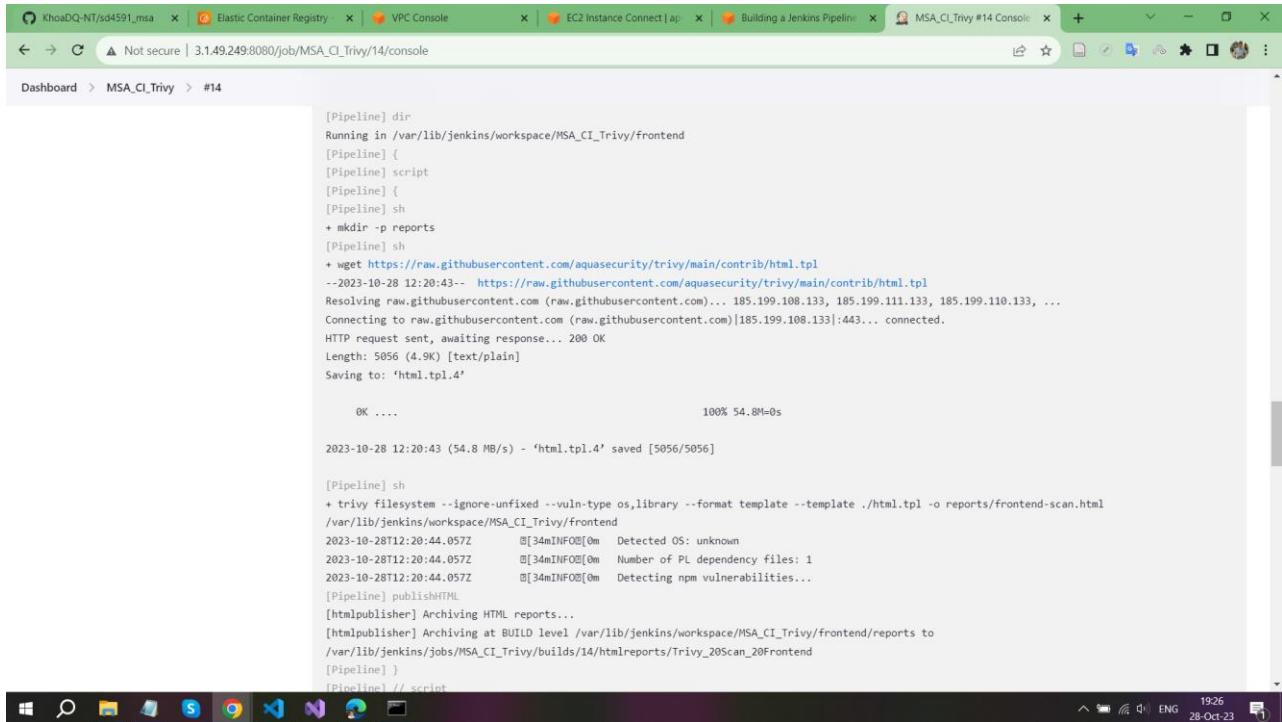
2023-10-28 12:20:42 (52.6 MB/s) - 'html.tpl.9' saved [5056/5056]

[Pipeline] sh
+ trivy filesystem --ignore-unfixed --vuln-type os,library --format template --template ./html.tpl -o reports/backend-scan.html
/var/lib/jenkins/workspace/MSA_CI_Trvy/backend
2023-10-28T12:20:42.351Z    [34mINFO[0m] Detected OS: unknown
2023-10-28T12:20:42.351Z    [34mINFO[0m] Number of PL dependency files: 1
2023-10-28T12:20:42.351Z    [34mINFO[0m] Detecting npm vulnerabilities...
[Pipeline] publishHTML
[htmlpublisher] Archiving HTML reports...
[htmlpublisher] Archiving at BUILD level /var/lib/jenkins/workspace/MSA_CI_Trvy/backend/reports to
/var/lib/jenkins/jobs/MSA_CI_Trvy/builds/14/htmlreports/Trivy_20Scan_20Backend
[Pipeline] }
[Pipeline] // script
```

Build frontend

```
Running in /var/lib/jenkins/workspace/MSA_CI_Trivy/frontend
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ docker build -t 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/frontend:latest .
Sending build context to Docker daemon 600.6kB
```

User Trivy scanner on frontend



```
[Pipeline] dir
Running in /var/lib/jenkins/workspace/MSA_CI_Trivy/frontend
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ mkdir -p reports
[Pipeline] sh
+ wget https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl
--2023-10-28 12:20:43- https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/html.tpl
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.111.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5056 (4.9K) [text/plain]
Saving to: 'html.tpl.4'

OK ....
100% 54.8M=0s

2023-10-28 12:20:43 (54.8 MB/s) - 'html.tpl.4' saved [5056/5056]

[Pipeline] sh
+ trivy filesystem --ignore-unfixed --vuln-type os,library --format template --template ./html.tpl -o reports/frontend-scan.html
/var/lib/jenkins/workspace/MSA_CI_Trivy/frontend
2023-10-28T12:20:44.057Z 34mINFO[0m Detected OS: unknown
2023-10-28T12:20:44.057Z 34mINFO[0m Number of PL dependency files: 1
2023-10-28T12:20:44.057Z 34mINFO[0m Detecting npm vulnerabilities...
[Pipeline] publishHTML
[htmlpublisher] Archiving HTML reports...
[htmlpublisher] Archiving at BUILD level /var/lib/jenkins/workspace/MSA_CI_Trivy/frontend/reports to
/var/lib/jenkins/jobs/MSA_CI_Trivy/builds/14/htmlreports/Trivy_20Scan_20Frontend
[Pipeline] }
[Pipeline] // script
```

Push image into ECR and complete CI pipeline

```
Constructing AWS CredentialsSetting AWS region ap-southeast-1
[Pipeline] {
[Pipeline] sh
+ aws ecr get-login-password --region ap-southeast-1
+ docker login --username AWS --password-stdin 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/backend
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
[Pipeline] sh
+ docker push 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/backend:latest
The push refers to repository [359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/backend]
20c1f73a02c5: Preparing

1c90t40000084: Pushed
3142bf4692e1: Pushed
latest: digest: sha256:660b924c9a3dd462ea5b3df1d61a235ab0bda5674acfc55160c31516fb66e12d size: 2411
[Pipeline] sh
+ docker login --username AWS --password-stdin 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/frontend
+ aws ecr get-login-password --region ap-southeast-1
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
[Pipeline] sh
+ docker push 359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/frontend:latest
The push refers to repository [359145461483.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-devops/frontend]
84a4c41623c5: Preparing
--> 84a4c41623c5: Pushed
a59att4ta5/i: Pushed
6a24eb7db86f: Pushed
latest: digest: sha256:6a985261d54223c8b8309e06bcaa83f56d0a7b61fa489993a71f77c7a17272fc size: 3051
[Pipeline] }
[Pipeline] // withAWS
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

KhoaDQ-NT/sd4591_msa | Elastic Container Registry | VPC Console | EC2 Instance Connect | ap | Building a Jenkins Pipeline | MSA_CI_Trivy [Jenkins] | +

Not secure | 3.1.49.249:8080/job/MSA_CI_Trivy/

Jenkins

Search (CTRL+K) | Dang Quoc Khoa | log out

Dashboard > MSA_CI_Trivy >

Pipeline MSA_CI_Trivy

Status Changes Build Now Configure Delete Pipeline Full Stage View GitHub Trivy Scan Backend Trivy Scan Frontend Rename Pipeline Syntax GitHub Hook Log Build History trend

Average stage times:
(Average full run time: ~46s)

Stage View

Declarative: Checkout SCM	Build Backend	Trivy Scan Backend	Build Frontend	Trivy Scan Frontend	Push to ECR
942ms	830ms	1s	435ms	623ms	8s
#15 Thg 10 28 19:35 No Changes	733ms	700ms	1s	665ms	970ms
#16 Thg 10 28 19:20 1 commit	1s	946ms	1s	675ms	1s 1min 13s
#17 Thg 10 28 19:19 No Changes	796ms	747ms	1s	726ms	1s 1s failed

Windows taskbar: File Explorer, Task View, Taskbar settings, Taskbar icons, Taskbar search, Taskbar pinned items, Taskbar status bar: ENG 28-Oct-23 19:37

5. Monitoring

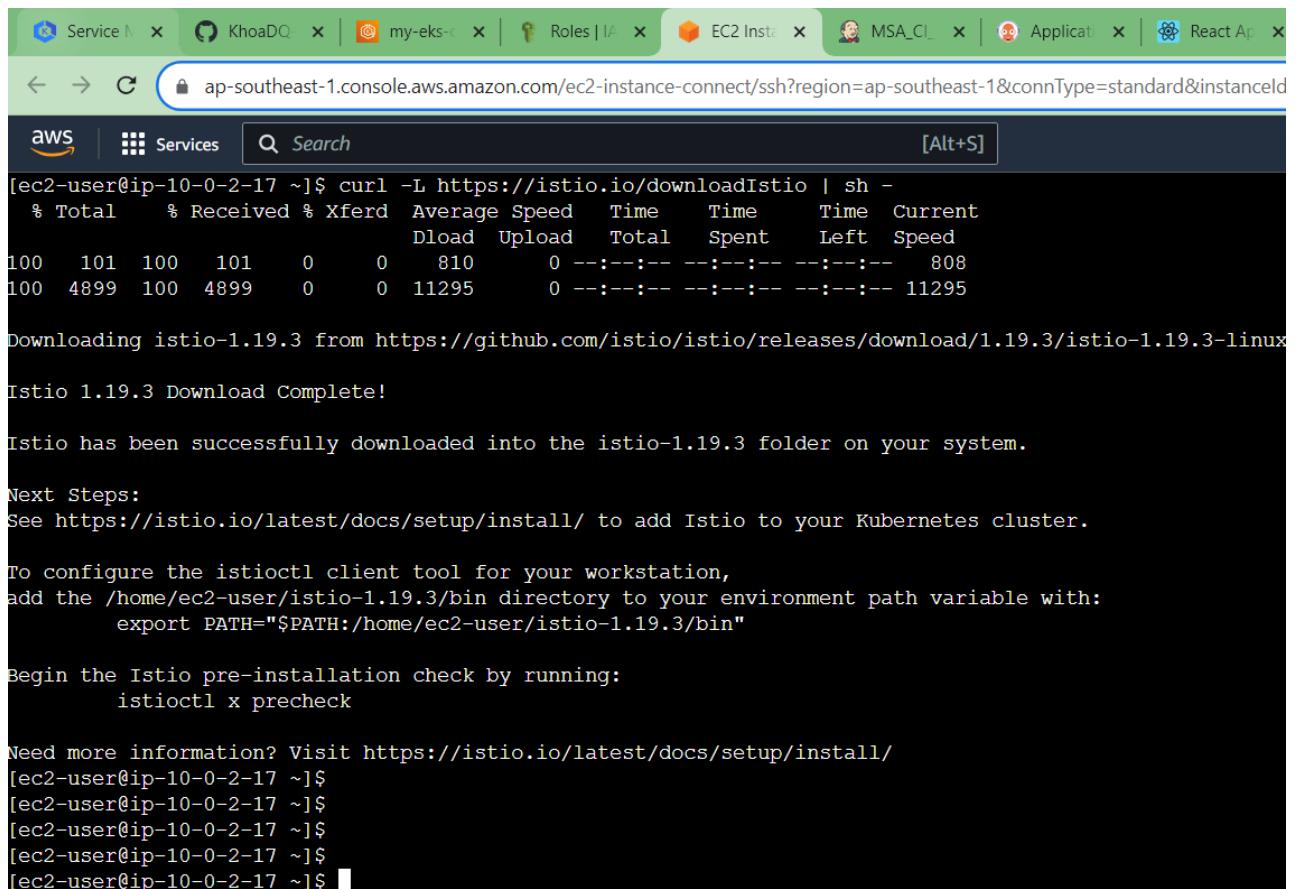
DevOps for Devs:

Set up Prometheus and Grafana to monitor EKS resources and default EKS resource metric.

Pi-sharp:

Use Istio to provide observability metrics, which can be visualized and collected using Grafana and Prometheus.

Download Istio Deployment Files



The screenshot shows a terminal window within the AWS CloudShell interface. The terminal title is '[ec2-user@ip-10-0-2-17 ~]\$'. The user is executing a curl command to download Istio 1.19.3 from its GitHub release page. The output shows the progress of the download, including file sizes and speeds. After the download completes, the user runs 'istioctl x precheck' to perform a pre-installation check. The terminal also provides links to the Istio documentation for setup and installation.

```
[ec2-user@ip-10-0-2-17 ~]$ curl -L https://istio.io/downloadIstio | sh -
% Total    % Received % Xferd  Average Speed   Time     Time      Current
          Dload  Upload   Total   Spent    Left  Speed
100  101  100  101    0      0  810      0 --:--:-- --:--:-- --:--:--  808
100  4899  100  4899    0      0 11295      0 --:--:-- --:--:-- --:--:-- 11295

Downloading istio-1.19.3 from https://github.com/istio/istio/releases/download/1.19.3/istio-1.19.3-linux-amd64.tgz
Istio 1.19.3 Download Complete!

Istio has been successfully downloaded into the istio-1.19.3 folder on your system.

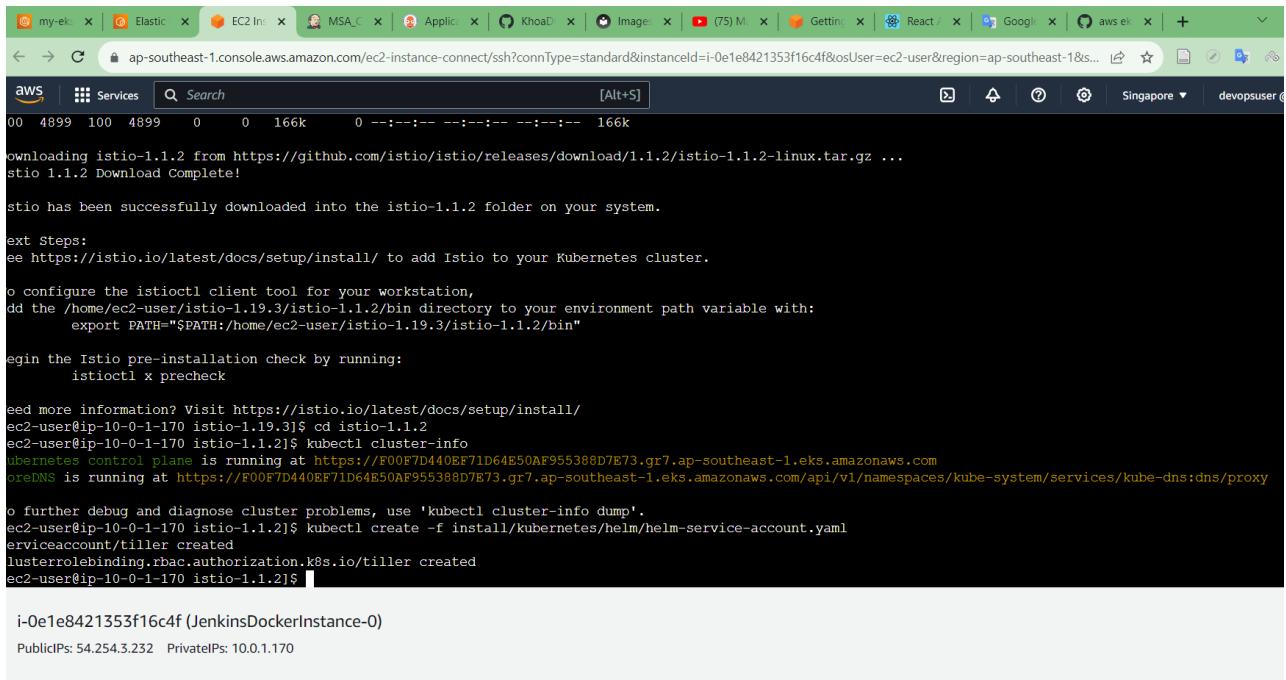
Next Steps:
See https://istio.io/latest/docs/setup/install/ to add Istio to your Kubernetes cluster.

To configure the istioctl client tool for your workstation,
add the /home/ec2-user/istio-1.19.3/bin directory to your environment path variable with:
export PATH="$PATH:/home/ec2-user/istio-1.19.3/bin"

Begin the Istio pre-installation check by running:
istioctl x precheck

Need more information? Visit https://istio.io/latest/docs/setup/install/
[ec2-user@ip-10-0-2-17 ~]$
```

Configure Helm



```
00 4899 100 4899 0 0 166k 0 --::-- --::-- --::-- 166k
ownloading istio-1.1.2 from https://github.com/istio/istio/releases/download/1.1.2/istio-1.1.2-linux.tar.gz ...
stio 1.1.2 Download Complete!

istio has been successfully downloaded into the istio-1.1.2 folder on your system.

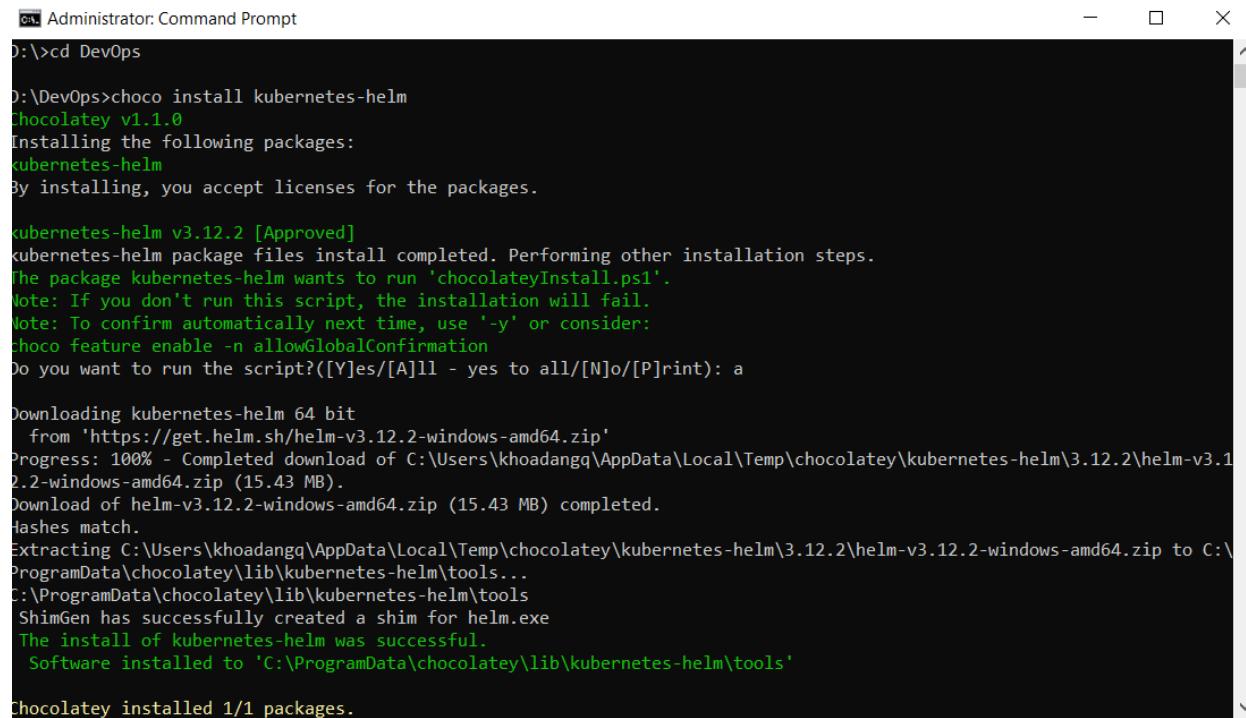
Next Steps:
ee https://istio.io/latest/docs/setup/install/ to add Istio to your Kubernetes cluster.

o configure the istioctl client tool for your workstation,
dd the /home/ec2-user/istio-1.1.2/bin directory to your environment path variable with:
    export PATH="$PATH:/home/ec2-user/istio-1.1.2/bin"

Begin the Istio pre-installation check by running:
    istioctl x precheck

eed more information? visit https://istio.io/latest/docs/setup/install/
ec2-user@ip-10-0-1-170 istio-1.1.2]$ cd istio-1.1.2
ec2-user@ip-10-0-1-170 istio-1.1.2]$ kubectl cluster-info
ubernetes control plane is running at https://F00F7D440E871D64E50AF955388D7E73.gr7.ap-southeast-1.eks.amazonaws.com
oreDNS is running at https://F00F7D440E871D64E50AF955388D7E73.gr7.ap-southeast-1.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-dns:proxy

o further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
ec2-user@ip-10-0-1-170 istio-1.1.2]$ kubectl create -f install/kubernetes/helm/helm-service-account.yaml
erviceaccount/tiller created
usterrolebinding.rbac.authorization.k8s.io/tiller created
ec2-user@ip-10-0-1-170 istio-1.1.2]$ i-0e1e8421353f16c4f (JenkinsDockerInstance-0)
PublicIPs: 54.254.3.232 PrivateIPs: 10.0.1.170
```



```
D:\>cd DevOps
D:\DevOps>choco install kubernetes-helm
Chocolatey v1.1.0
Installing the following packages:
kubernetes-helm
By installing, you accept licenses for the packages.

kubernetes-helm v3.12.2 [Approved]
kubernetes-helm package files install completed. Performing other installation steps.
The package kubernetes-helm wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N)o/[P]rint): a

Downloading kubernetes-helm 64 bit
  from 'https://get.helm.sh/helm-v3.12.2-windows-amd64.zip'
Progress: 100% - Completed download of C:\Users\khoadang\AppData\Local\Temp\chocolatey\kubernetes-helm\3.12.2\helm-v3.12.2-windows-amd64.zip (15.43 MB).
Download of helm-v3.12.2-windows-amd64.zip (15.43 MB) completed.
Hashes match.
Extracting C:\Users\khoadang\AppData\Local\Temp\chocolatey\kubernetes-helm\3.12.2\helm-v3.12.2-windows-amd64.zip to C:\ProgramData\chocolatey\lib\kubernetes-helm\tools...
C:\ProgramData\chocolatey\lib\kubernetes-helm\tools
| ShimGen has successfully created a shim for helm.exe
| The install of kubernetes-helm was successful.
| Software installed to 'C:\ProgramData\chocolatey\lib\kubernetes-helm\tools'

Chocolatey installed 1/1 packages.
```

Prometheus

```
Administrator: Command Prompt - kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/prometheus-kube-state-metrics   1/1     1           1           104s
deployment.apps/prometheus-prometheus-pushgateway   1/1     1           1           104s
deployment.apps/prometheus-server                1/1     1           1           104s

NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/prometheus-kube-state-metrics-69c8887cfd   1        1       1   104s
replicaset.apps/prometheus-prometheus-pushgateway-79ff799669   1        1       1   104s
replicaset.apps/prometheus-server-855f6c967f            1        1       1   104s

NAME          READY   AGE
statefulset.apps/prometheus-alertmanager   0/1     104s

D:\DevOps>kubectl get pods -n prometheus
NAME          READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-0      1/1     Running   0          5m50s
prometheus-kube-state-metrics-69c8887cfd-zrcnn   1/1     Running   0          5m50s
prometheus-prometheus-node-exporter-n5xc4        1/1     Running   0          5m51s
prometheus-prometheus-pushgateway-79ff799669-6h6n2  1/1     Running   0          5m50s
prometheus-server-855f6c967f-xmb6t              2/2     Running   0          5m50s

D:\DevOps>kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
Forwarding from 127.0.0.1:9090 -> 9090
Forwarding from [::1]:9090 -> 9090
```

The screenshot shows a web browser window with the URL `localhost:9090/graph`. The page is titled "Prometheus". At the top, there are several checkboxes for configuration: "Use local time", "Enable query history", "Enable autocomplete" (which is checked), "Enable highlighting" (which is checked), and "Enable linter". Below this is a search bar with the placeholder "Expression (press Shift+Enter for newlines)". To the right of the search bar are "Table" and "Graph" tabs, with "Graph" being the active tab. Underneath the tabs is a dropdown for "Evaluation time" with arrows for navigating between time points. The main content area displays the message "No data queried yet". In the bottom right corner of the panel, there is a "Remove Panel" link.

localhost:9090/targets?search=

Prometheus Targets

All scrape pools ▾ All Unhealthy Collapse All Filter by endpoint or labels Unknown Unhealthy Healthy

kubernetes-apiservers (2/2 up) [show less]

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://10.0.5.62/metrics	UP	instance="10.0.5.62:443" job="kubernetes-apiservers"	12.98s ago	117.160ms	
https://10.0.6.109/metrics	UP	instance="10.0.6.109:443" job="kubernetes-apiservers"	12.431s ago	199.654ms	

kubernetes-nodes (1/1 up) [show less]

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://kubernetes.default.svc/api/v1/nodes/ip-10-0-3-235.ap-southeast-1.compute.internal/proxy/metrics	UP	beta_kubernetes_io_arch="amd64" beta_kubernetes_io_instance_type="t3.large" beta_kubernetes_io_os="linux" eks.amazonaws.com_capacityType="SPOT" eks.amazonaws.com_nodegroup="cluster-wg-202308161103023065000000e" eks.amazonaws.com_nodegroup_image="ami-0c65faadff48b4c197" eks.amazonaws.com_sourceLaunchTemplateId="R-01c1bbf4ead2afc6b" eks.amazonaws.com_sourceLaunchTemplateVersion="1" failure_domain_beta_kubernetes_io_region="ap-southeast-1" failure_domain_beta_kubernetes_io_zone="ap-southeast-1a" instance="ip-10-0-3-235.ap-southeast-1.compute.internal" job="kubernetes-nodes" k8s.io_cloud_provider_aws="c51c948643288821e634433ab3c7cfdf"	51.244s ago	130.754ms	

Graphana

devopsuser | my-eks-cluste | EC2 Instance | MSA_CD #28 | React App | Prometheus | Prereqs :: Ama

← → C ap-southeast-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-southeast-1&connType=standard&instanceId=i-0c92556b9

aws Services Search [Alt+S]

EC2

```
kubectl get secret --namespace grafana grafana -o jsonpath=".data.admin-password" | base64 --decode ; echo
```

2. The Grafana server can be accessed via port 80 on the following DNS name from within your cluster:

```
grafana.grafana.svc.cluster.local
```

Get the Grafana URL to visit by running these commands in the same shell:
NOTE: It may take a few minutes for the LoadBalancer IP to be available.
You can watch the status of by running 'kubectl get svc --namespace grafana -w grafana'
export SERVICE_IP=\$(kubectl get svc --namespace grafana grafana -o jsonpath='(.status.loadBalancer.ingress[0].http://\$SERVICE_IP:80

3. Login with the password from step 1 and the username: admin

```
[ec2-user@ip-10-0-1-20 ~]$ ^C
[ec2-user@ip-10-0-1-20 ~]$ kubectl get all -n grafana
NAME           READY   STATUS    RESTARTS   AGE
pod/grafana-5fbb56d446-4slb8   1/1     Running   0          76s

NAME              TYPE      CLUSTER-IP      EXTERNAL-IP
service/grafana   LoadBalancer   172.20.189.82   a5445ba2134854086985bc5bfc913497-1614150311.ap-southeast-1.elb.amazonaws.com

NAME            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/grafana   1/1     1           1          77s

NAME           DESIRED   CURRENT   READY   AGE
replicaset.apps/grafana-5fbb56d446   1        1        1        77s
[ec2-user@ip-10-0-1-20 ~]$
```

Not secure | a5445ba2134854086985bc5bfc913497-1614150311.ap-southeast-1.elb.amazonaws.com/login



Documentation | Support | Community | Open Source | v10.0.3 (eb8dd72637)

Screenshot of a web browser showing the process of importing a Grafana dashboard from Grafana.com.

The top window shows the "Import dashboard" screen. It displays the following details:

- Published by:** sekkal
- Updated on:** 2018-06-07 06:51:56

Options:

- Name:** Kubernetes Cluster (Prometheus)
- Folder:** General

A note about Unique Identifier (UID) is present, explaining its use for consistency across multiple Grafana installations. A "Change uid" button is available.

The "prometheus" folder is selected in the dropdown menu.

Buttons at the bottom: **Import** (highlighted in blue), **Cancel**.

The bottom window shows the imported dashboard titled "Kubernetes Cluster (Prometheus)".

Dashboard filters: **node**: * and **namespace**: *.

Section: **Cluster Health** (disabled):

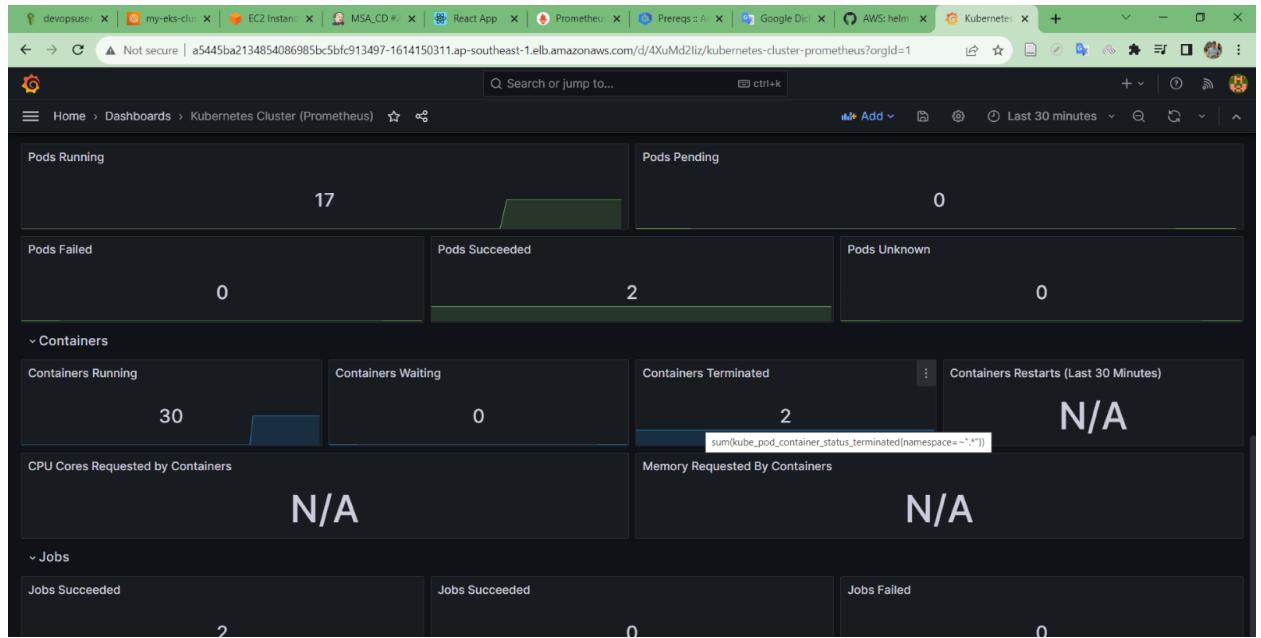
- Cluster Pod Usage: N/A
- Cluster CPU Usage: N/A
- Cluster Memory Usage: N/A
- Cluster Disk Usage: N/A

Section: **Cluster Capacity** (disabled):

- Cluster Pod Capacity: No data (Graph shows constant value around 18.5 pods from 23:30 to 23:50)
- Cluster CPU Capacity: No data (Graph shows constant value around 0.500 cores from 23:30 to 23:50)
- Cluster Mem Capacity: No data (Graph shows constant value around 0 B from 23:30 to 23:50)
- Cluster Disk Capacity: No data (Graph shows constant value around 0 B from 23:30 to 23:50)

Section: **Deployments** (disabled):

- Deployment Replicas - Up To Date: No data
- Deployment Replicas: No data
- Deployment Replicas - Updated: No data
- Deployment Replicas - Unavailable: No data



6. Use GitOps for the CD pipeline.

DevOps for Devs:

Install Argo CD on your EKS Cluster

Create an application in Argo CD (you can use GUI or CLI)

Pi-sharp:

- Install Argo CD Image Update (<https://argocd-image-updater.readthedocs.io/en/stable/install/installation/>). This will help you listen to changes the image on ECR and update the image tag in Ops repo.
- Use Blue/Green strategy for deployment.

Using ArgoCD Image Updater

Install ArgoCD Image Updater and make some necessary configurations

```
C:\Users\khoadangq>kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj-labs/argocd-image-updater/stable/manifests/install.yaml
serviceaccount/argocd-image-updater created
role.rbac.authorization.k8s.io/argocd-image-updater created
rolebinding.rbac.authorization.k8s.io/argocd-image-updater created
configmap/argocd-image-updater-config created
configmap/argocd-image-updater-ssh-config created
secret/argocd-image-updater-secret created
deployment.apps/argocd-image-updater created

C:\Users\khoadangq>
```

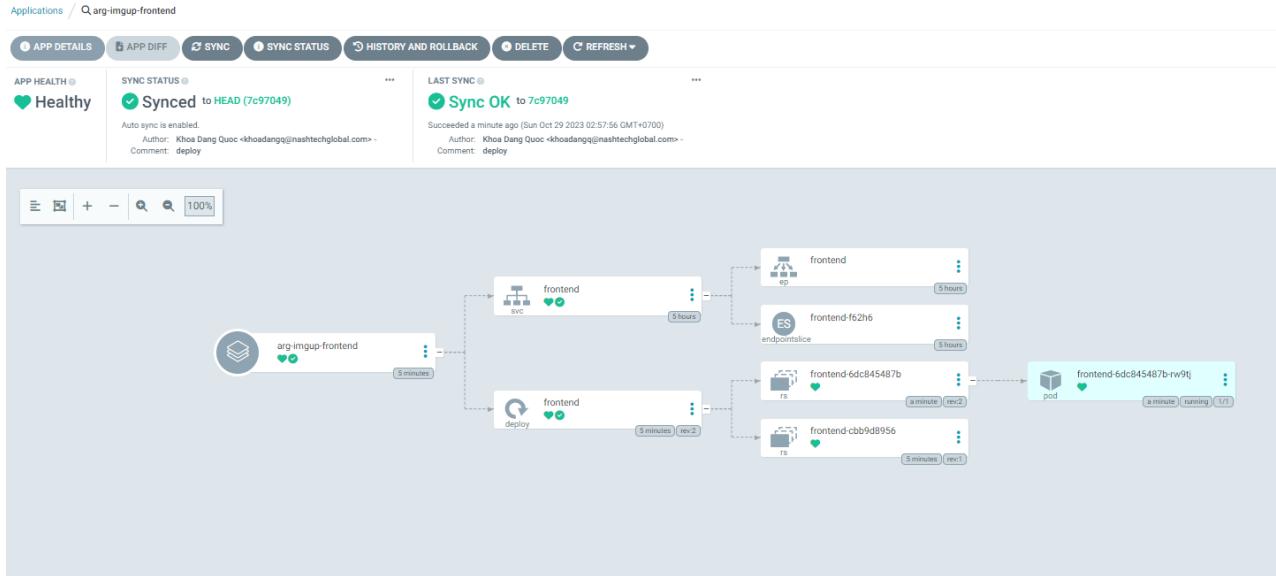
```
C:\Users\khoadangq>kubectl --namespace argocd logs --selector app.kubernetes.io/name=argocd-image-updater
time="2023-10-28T19:05:23Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
time="2023-10-28T19:05:23Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
time="2023-10-28T19:07:24Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
time="2023-10-28T19:07:24Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
time="2023-10-28T19:09:24Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
time="2023-10-28T19:09:24Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
time="2023-10-28T19:11:24Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
time="2023-10-28T19:11:24Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
time="2023-10-28T19:13:24Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
time="2023-10-28T19:13:24Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"

C:\Users\khoadangq>
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure for "sd4591_msa_cd". The "frontend" directory is selected. Inside "frontend", there are files: "application.yaml", ".deployment.yaml", ".service.yaml", "backend.yaml", "kustomization.yaml", and "mongo/mongodb.yaml".
- Editor:** The main editor area displays the content of "application.yaml". The file defines an ArgoCD application named "frontend" in the "argocd" namespace. It specifies the project as "frontend", the source repository as "https://github.com/KhoaDQ-NT/sd4591_msa_cd", and the target revision as "HEAD". The destination is set to "https://kubernetes.default.svc" in the "argocd" namespace. The sync policy is "automated" with self-healing enabled.
- Terminal:** The terminal at the bottom shows the command "PS D:\DevOps\sd4591_msa_cd>".
- Bottom Status Bar:** Displays the status "You, 15 minutes ago", line numbers "Ln 13, Col 20", spaces information "Spaces: 2", and file encoding "UTF-8".

Sort: name ▾ Items per page: 10 ▾	
<div> backend</div> <p>Project: argocd</p> <p>Labels:</p> <p>Status: Healthy Synced</p> <p>Repository: https://github.com/KhoaDQ-NT/sd4591_m...</p> <p>Target Re...: HEAD</p> <p>Path: ./ArgoCDIU/backend</p> <p>Destinati...: in-cluster</p> <p>Namespa...: argocd</p> <p>Created At: 10/29/2023 15:59:59 (2 hours ago)</p> <p>Last Sync: 10/29/2023 16:00:00 (2 hours ago)</p> <div><button> SYNC</button> <button> REFRESH</button> <button> DELETE</button></div>	
<div> frontend</div> <p>Project: argocd</p> <p>Labels:</p> <p>Status: Healthy Synced</p> <p>Repository: https://github.com/KhoaDQ-NT/sd4591_m...</p> <p>Target Re...: HEAD</p> <p>Path: ./ArgoCDIU/frontend</p> <p>Destinati...: in-cluster</p> <p>Namespa...: argocd</p> <p>Created At: 10/29/2023 16:00:49 (2 hours ago)</p> <p>Last Sync: 10/29/2023 18:00:25 (3 minutes ago)</p> <div><button> SYNC</button> <button> REFRESH</button> <button> DELETE</button></div>	
<div> mongo</div> <p>Project: argocd</p> <p>Labels:</p> <p>Status: Healthy Synced</p> <p>Repository: https://github.com/KhoaDQ-NT/sd4591_m...</p> <p>Target Re...: HEAD</p> <p>Path: ./ArgoCDIU/mongo</p> <p>Destinati...: in-cluster</p> <p>Namespa...: argocd</p> <p>Created At: 10/29/2023 15:58:37 (2 hours ago)</p> <p>Last Sync: 10/29/2023 15:58:39 (2 hours ago)</p> <div><button> SYNC</button> <button> REFRESH</button> <button> DELETE</button></div>	



I did try to find any problem, but it still did not work.

```
MINGW64:/  
time="2023-10-29T10:13:07Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"  
time="2023-10-29T10:13:07Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"  
khoadangq@VNNOTO1084 MINGW64 /  
$ aws ecr describe-images --repository-name my-ecr-repo-devops/frontend --image-ids imageTag=16 --query 'images[0].imageDigest' --output text  
None  
  
khoadangq@VNNOTO1084 MINGW64 /  
$ aws ecr describe-images --repository-name my-ecr-repo-devops/frontend --image-ids imageTag=15 --query 'images[0].imageDigest' --output text  
None  
  
khoadangq@VNNOTO1084 MINGW64 /  
$ aws ecr describe-images --repository-name my-ecr-repo-devops/frontend --image-ids imageTag=17 --query 'images[0].imageDigest' --output text  
An error occurred (ImageNotFoundException) when calling the DescribeImages operation: The image with imageId {imageDigest:'null', imageTag:'17'} does not exist v  
s/frontend in the registry with id '359145461483'  
khoadangq@VNNOTO1084 MINGW64 /  
$ aws ecr describe-images --repository-name my-ecr-repo-devops/frontend --output table  
+-----+-----+  
| DescribeImages |  
+-----+-----+  
| imageDetails |  
+-----+-----+  
| artifactMediaType | application/vnd.docker.container.image.v1+json  
| imageDigest | sha256:27231ca3327088a366cac85657250ff7d62473b6afbc44350f573b9a65bd709  
| imageManifestMediaType | application/vnd.docker.distribution.manifest.v2+json  
| imagePushedAt | 2023-10-29T16:29:28+07:00  
| imageSizeInBytes | 48196353  
| lastRecordedPullTime | 359145461483  
| registryId | my-ecr-repo-devops/frontend  
| repositoryName |  
+-----+-----+  
| imageTags |  
+-----+-----+  
| 16 |  
+-----+-----+  
| imageDetails |  
+-----+-----+  
| artifactMediaType | application/vnd.docker.container.image.v1+json  
| imageDigest | sha256:667985d1aadaa92bd3985ff7a6b7bf183a093d433d314662365c28c9e48e5fd8  
| imageManifestMediaType | application/vnd.docker.distribution.manifest.v2+json  
| imagePushedAt | 2023-10-29T15:54:35+07:00  
| imageSizeInBytes | 481962019  
| lastRecordedPullTime | 2023-10-29T16:00:51.843000+07:00  
| registryId | 359145461483  
| repositoryName | my-ecr-repo-devops/frontend  
+-----+-----+  
| imageTags |  
+-----+-----+  
| 15 |  
+-----+-----+
```

```

MINGW64/
Time="2023-10-29T09:51:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T09:51:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T09:55:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T09:57:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:01:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:01:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:01:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:01:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:05:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:05:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:07:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:07:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:09:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:09:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:11:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:11:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:13:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:13:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:15:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:15:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:17:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:17:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:19:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:19:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:21:06Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:21:06Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:23:07Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:23:07Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"
Time="2023-10-29T10:25:07Z" level=info msg="Starting image update cycle, considering 0 annotated application(s) for update"
Time="2023-10-29T10:25:07Z" level=info msg="Processing results: applications=0 images_considered=0 images_skipped=0 images_updated=0 errors=0"

hoadangq@VNNT01084 MINGW64 /
$ aws ecr describe-images --repository-name my-ecr-repo-devops/frontend --image-ids imageTag=16
{
  "imageDetails": [
    {
      "registryId": "359145461483",
      "repositoryName": "my-ecr-repo-devops/frontend",
      "imageDigest": "sha256:7231ca332f088a366cac85657250ff7d62473b6fbca44350f573b9a65bd709",
      "imageTags": [
        "16"
      ],
      "imageSizeInBytes": 481963553,
      "imagePushedAt": "2023-10-29T10:23:07Z",
      "imageManifestMediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "artifactMediaType": "application/vnd.docker.container.image.v1+json"
    }
  ]
}

hoadangq@VNNT01084 MINGW64 /
$ |

```

ArgoCD Image Updater cannot detect new images, so they do not perform any operations in GitHub Actions. Synchronization still must be done manually. I will take more time to learn later.

The screenshot shows the ArgoCD interface for the 'frontend' application. The sync status is 'Sync OK' to commit '4286ca0'. The dependency graph illustrates the deployment structure:

- A 'frontend' service (svc) is connected to an 'frontend' endpoint slice (ep).
- The endpoint slice connects to an 'frontend-tb8qt' endpoint slice.
- This leads to two 'frontend' pods: 'frontend-8c4c4845c' and 'frontend-cc449dd47'.
- 'frontend-8c4c4845c' is in a 'running' state with 1/1 replicas.
- 'frontend-cc449dd47' is in a 'terminating' state with 1/1 replicas.
- The entire process took 2 hours.

```

MINGW64:/ 
NAME                                     READY   STATUS    RESTARTS   AGE
argocd-application-controller-0           1/1     Running   0          132m
argocd-applicationset-controller-568754c579-lbw7z 1/1     Running   0          132m
argocd-dex-server-7658cdcf77-hkzmr       1/1     Running   0          132m
argocd-image-updater-88454679d-8gfd        1/1     Running   0          99m
argocd-notifications-controller-5548b96954-bdjz7 1/1     Running   0          132m
argocd-redis-6976fc7dfc-bfcks            1/1     Running   0          132m
argocd-repo-server-7594f8849c-w82rs       1/1     Running   0          132m
argocd-server-58cc545d87-hmdvj           1/1     Running   0          132m
backend-b4bd4594-f9s2k                   1/1     Running   0          124m
frontend-8c4c4845c-vzrgb                 1/1     Running   0          4m11s
mongo-0                                    1/1     Running   0          125m

khoadangq@VNNOT01084 MINGW64 / 
$ kubectl get svc -n argocd
NAME                           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)
argocd-applicationset-controller   ClusterIP  172.20.15.208 <none>        7000/TCP,8080/TCP
argocd-dex-server                ClusterIP  172.20.56.96  <none>        5556/TCP,5557/TCP,5558/TCP
argocd-metrics                  ClusterIP  172.20.183.104 <none>        8082/TCP
argocd-notifications-controller-metrics   ClusterIP  172.20.82.50  <none>        9001/TCP
argocd-redis                     ClusterIP  172.20.249.33 <none>        6379/TCP
argocd-repo-server               ClusterIP  172.20.85.220 <none>        8081/TCP,8084/TCP
argocd-server                    ClusterIP  172.20.18.132 <none>        80/TCP,443/TCP
argocd-server-metrics             ClusterIP  172.20.61.58  <none>        8083/TCP
backend                         ClusterIP  172.20.195.124 <none>        3000/TCP
frontend                        ClusterIP  172.20.219.34 <none>        3000/TCP
mongo                           ClusterIP  172.20.59.84  <none>        27017/TCP

khoadangq@VNNOT01084 MINGW64 / 
$ kubectl port-forward svc/frontend -n argocd 8081:3000
Forwarding from 127.0.0.1:8081 -> 3000
Forwarding from [::1]:8081 -> 3000

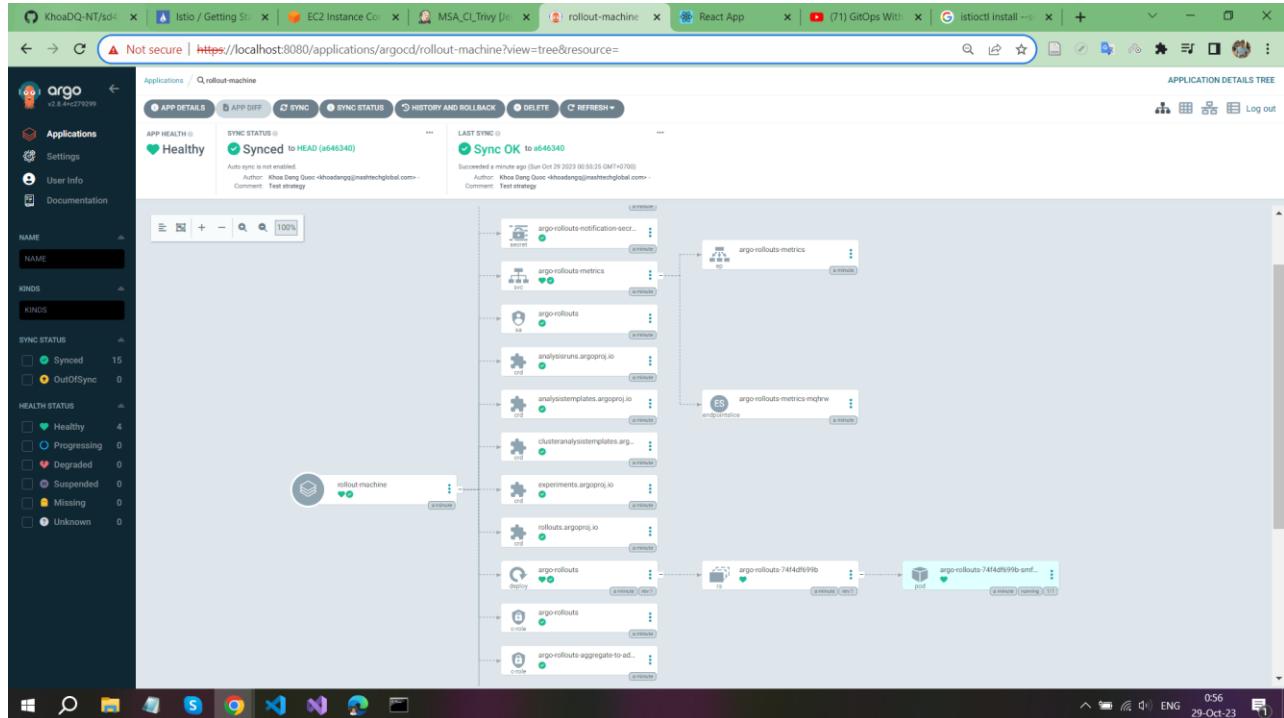
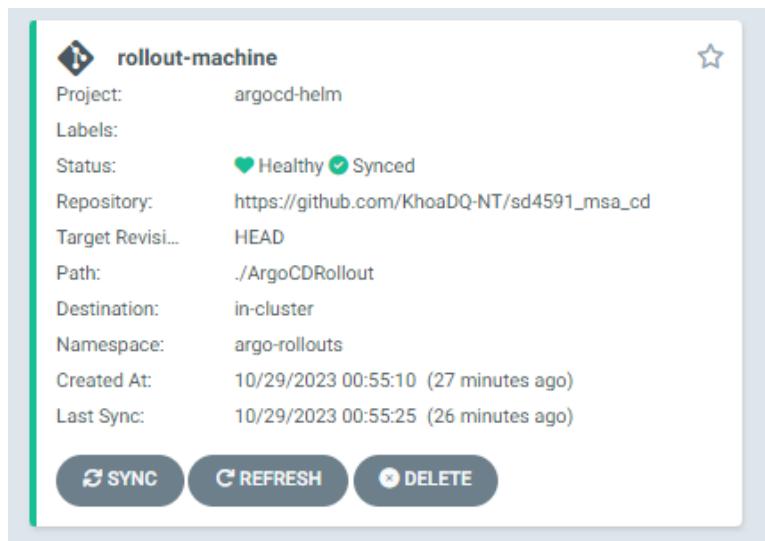
```

The screenshot shows a browser window with the title 'Todos K 17'. Below the title, there is a text input field and a blue 'Add Todo' button. Underneath the input field, a light blue box contains the text 'No Todos to display'.

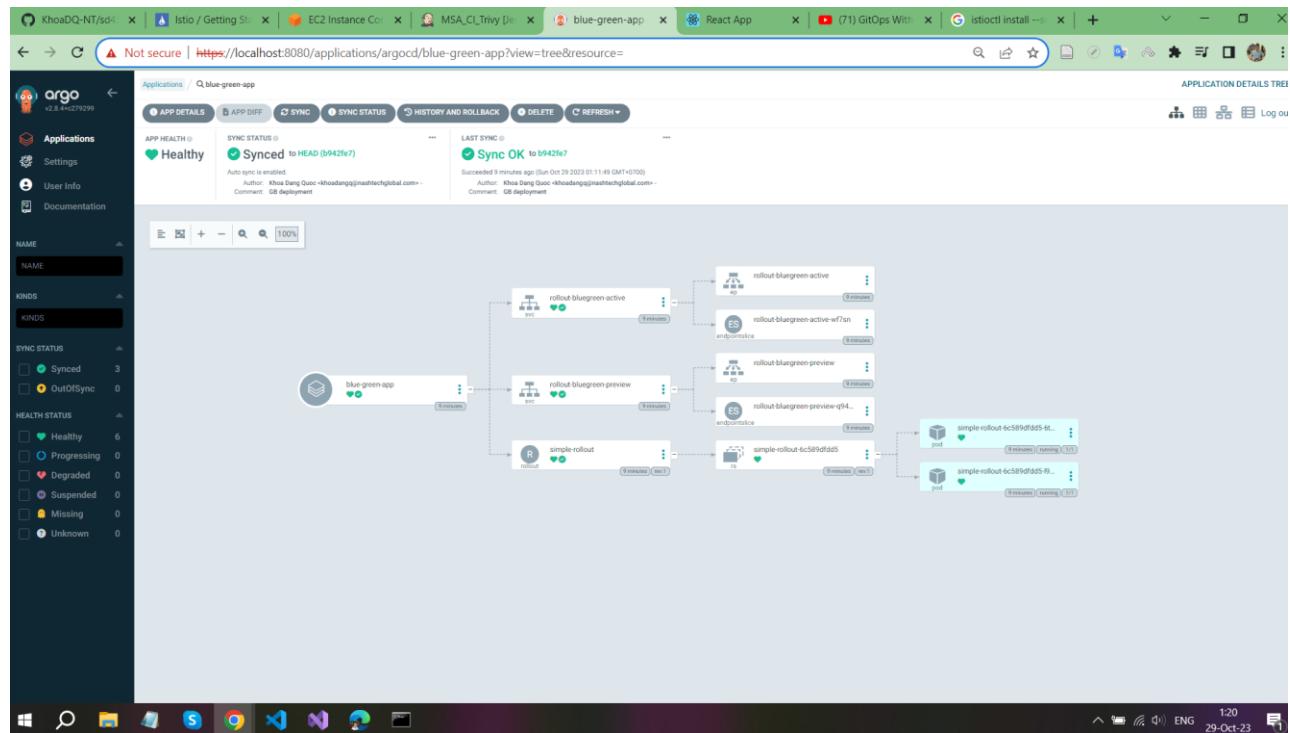
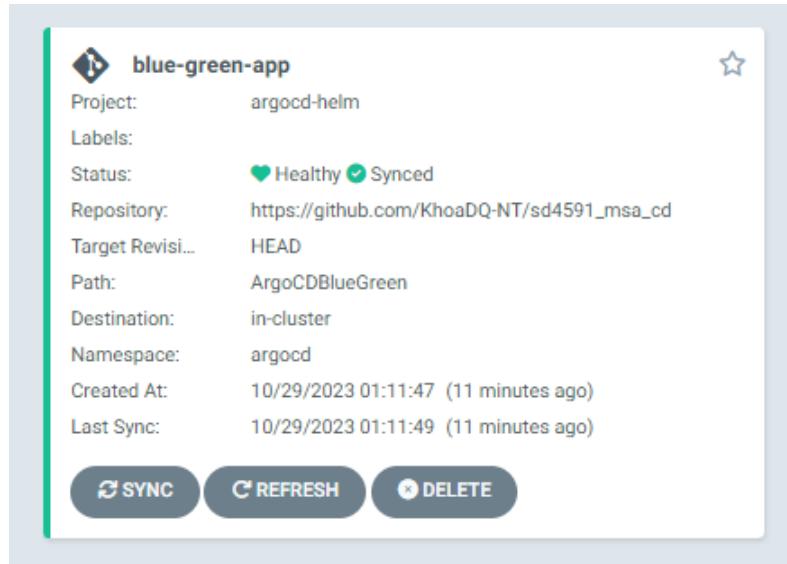


Use Blue/Green strategy

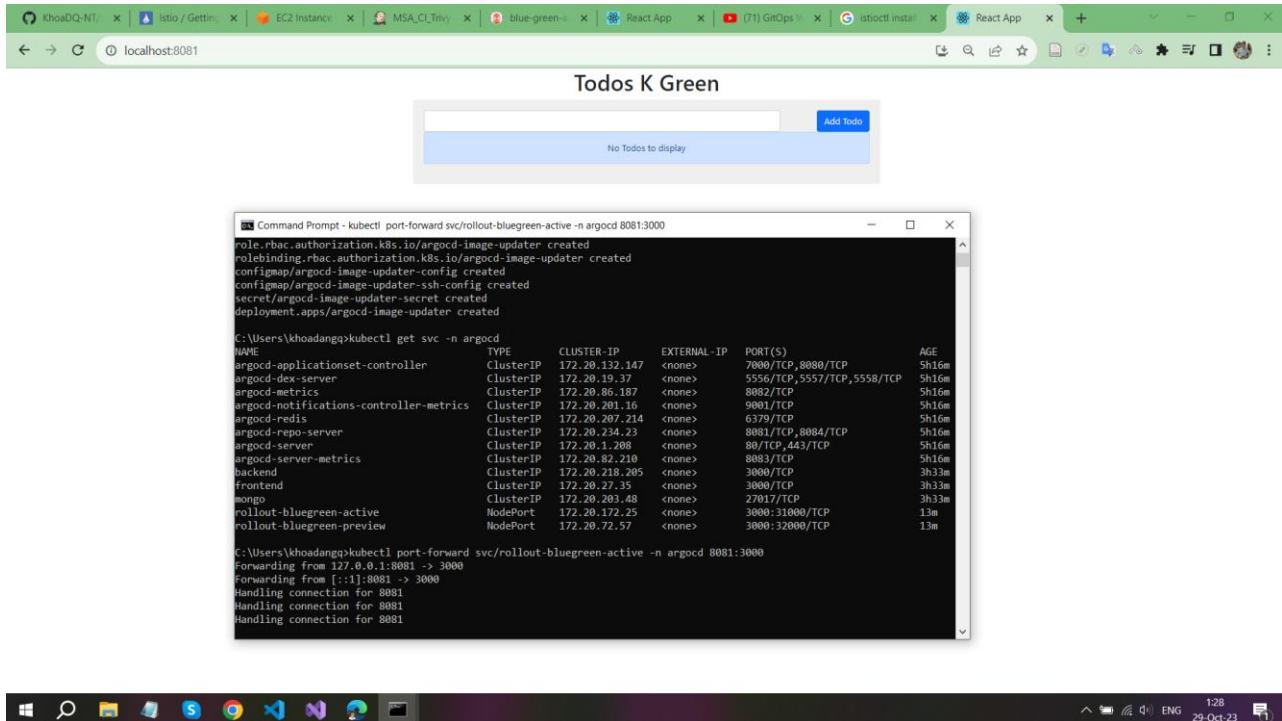
Install rollout-machine to switch between blue and green version of our application



Create blue-green-app that will get deployment based on image version change



Check first deployment



After changing the image, some actions will be processed:

- ArgoCD trigger the changes and have a replica version of application
- Argo Rollouts creates another replica with the new version
- The existing version still works and gets live traffic
- ArgoCD will mark our application as out-of-sync
- Until the application blue-green-app is synchronized and the new version is completely initialized. Our web application will be switched to new version for using

After the changes were applied, forward port if it's necessary and continue to use the application

The Argo UI dashboard shows three applications:

- blue-green-app**: Project: argocephl, Status: Progressing Synced, Repository: https://github.com/KhoaDQ-NT/sd4591_msa_cd, Target Revise...: HEAD, Path: ArgoCDBlueGreen, Destination: in-cluster, Namespace: argocephl, Created At: 10/29/2023 01:11:47 (22 minutes ago), Last Sync: 10/29/2023 01:33:31 (a few seconds ago).
- helm-app**: Project: argocephl, Status: Healthy Synced, Repository: https://github.com/KhoaDQ-NT/sd4591_msa_cd, Target Revise...: HEAD, Path: ArgoCDHelm, Destination: in-cluster, Namespace: argocephl, Created At: 10/28/2023 21:52:10 (4 hours ago), Last Sync: 10/28/2023 22:12:42 (3 hours ago).
- rollout-machine**: Project: argocephl, Status: Healthy Synced, Repository: https://github.com/KhoaDQ-NT/sd4591_msa_cd, Target Revise...: HEAD, Path: /ArgoCDRollout, Destination: in-cluster, Namespace: argo-rollsouts, Created At: 10/29/2023 00:55:10 (38 minutes ago), Last Sync: 10/29/2023 00:55:25 (38 minutes ago).

A red arrow points from the 'blue-green-app' section to the 'Todos K Blue' application window below.

Todos K Blue

No Todos to display

Add Todo

Command Prompt - kubectl port-forward svc/rollout-bluegreen-active -n argocephl 8081:3000

```

ClusterIP 172.20.207.214 <none> 6379/TCP 5h16m
ClusterIP 172.20.234.23 <none> 8081/TCP,8084/TCP 5h16m
ClusterIP 172.20.1.208 <none> 80/TCP,443/TCP 5h16m
ClusterIP 172.20.82.210 <none> 8083/TCP 5h16m
ClusterIP 172.20.218.205 <none> 3000/TCP 3h33m
ClusterIP 172.20.1.27.35 <none> 3000/TCP 3h33m
ClusterIP 172.20.1.48 <none> 27017/TCP 3h33m
NodePort 172.20.172.25 <none> 3000:31000/TCP 13m
NodePort 172.20.72.57 <none> 3000:32000/TCP 13m

```

E1029 01:36:39.370241 10140 portforward.go:409] an error occurred forwarding 8081 -> 3000: error forwarding port 3000 to pod 98d02face0d8ceaa9d4a146e24efde3a3817ccbe385012cce9234e989161f, uid : failed to find sandbox "98d02face0d8ceaa9d4a146e24efde3a3817ccbe385012cce9234e989161f" in store: not found
error: lost connection to pod

C:\Users\khoadang\kubectl port-forward svc/rollout-bluegreen-active -n argocephl 8081:3000

```

Forwarding from 127.0.0.1:8081 -> 3000
Forwarding from [::]:8081 -> 3000
Handling connection for 8081
E1029 01:36:39.370241 10140 portforward.go:409] an error occurred forwarding 8081 -> 3000: error forwarding port 3000 to pod 98d02face0d8ceaa9d4a146e24efde3a3817ccbe385012cce9234e989161f, uid : failed to find sandbox "98d02face0d8ceaa9d4a146e24efde3a3817ccbe385012cce9234e989161f" in store: not found
error: lost connection to pod

C:\Users\khoadang\kubectl port-forward svc/rollout-bluegreen-active -n argocephl 8081:3000
Forwarding from 127.0.0.1:8081 -> 3000
Forwarding from [::]:8081 -> 3000
Handling connection for 8081
Handling connection for 8081
Handling connection for 8081

```