Practical DevOps – CI/CD Pipeline and deploying applications on AWS EKS

1. Table of Contents

I. Introduction

1. Setting up a CI/CD Pipeline and deploying applications on AWS EKS.
2. Reuse Jenkins CI and use GitOps for the CD pipeline.
3. Setting up Monitoring Prometheus and Grafana.

II. Prerequisites

- Have an AWS account.
- AWS CLI installed.
- Terraform installed.
- Kubectl installed.
- Helm installled.

III. Git Repositories

1. Infrastructure as code

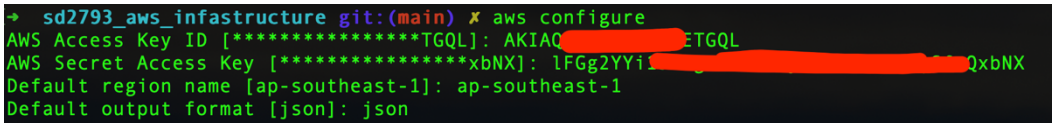
   sd2793_aws_infrastructure

2. Micro-services application

   sd2793_msa

IV. Provisioning Infrastructure by Terraform

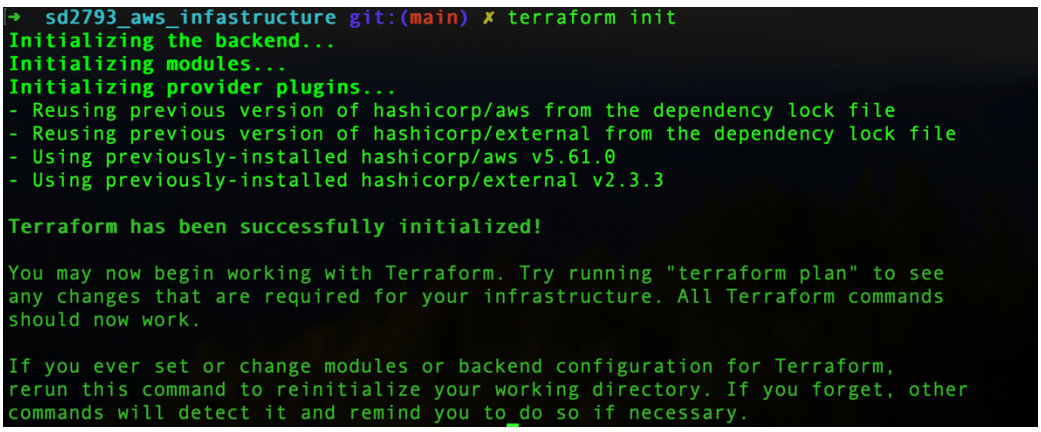
1. AWS Configure



2. Provisioning

1. Git clone from repository infrastructure as code
2. Go to folder sd2793_aws_infrastructure

   2.1. Initialize terraform terraform init



2.1. Provisioning infra terraform apply -auto-approve --var-file "terraform.tfvars"

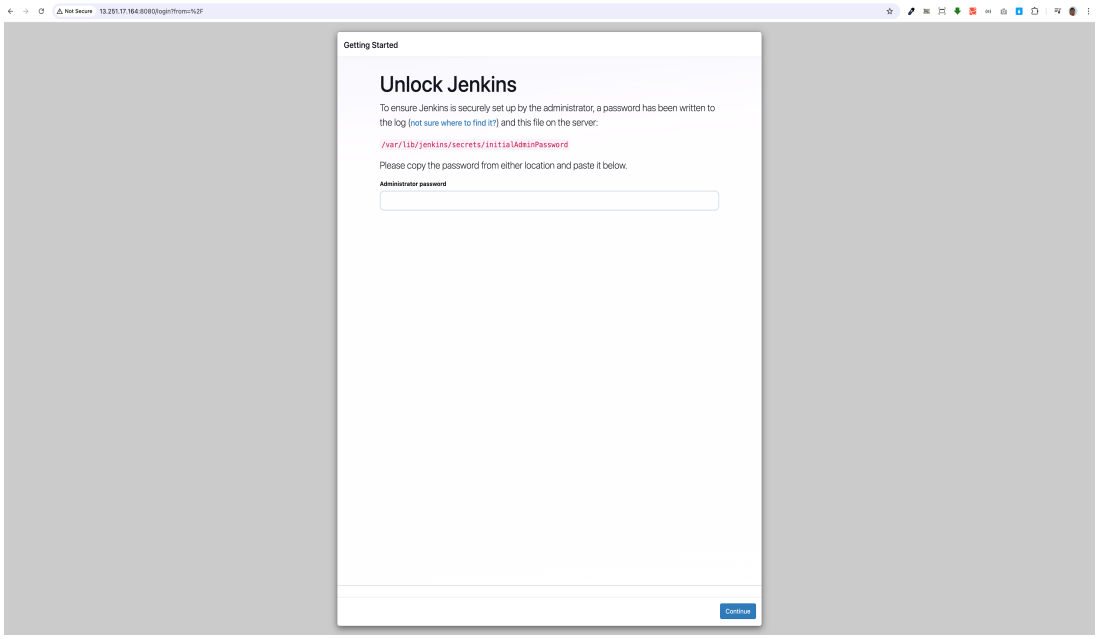
```
➜  sd2793_aws_infastructure git:(main) ✗ terraform apply -auto-approve --var-file "terraform.tfvars"
module.eks_cluster_and_worker_nodes.data.external.max_pods_calculator["one"]: Reading...
module.eks_iam_role.data.aws_iam_policy_document.assume_role_policy: Reading...
module.jenkins_iam_role.data.aws_iam_policy_document.assume_role_policy: Reading...
module.jenkins_iam_role.data.aws_iam_policy_document.eks_policy: Reading...
module.eks_iam_role.data.aws_iam_policy_document.assume_workers: Reading...
module.eks_iam_role.data.aws_iam_policy_document.assume_role_policy: Read complete after 0s [id=3552664922]
module.eks_iam_role.data.aws_iam_policy_document.assume_workers: Read complete after 0s [id=2851119427]
module.jenkins_iam_role.data.aws_iam_policy_document.eks_policy: Read complete after 0s [id=3460616156]
module.jenkins_iam_role.data.aws_iam_policy_document.ecr_policy: Reading...
module.jenkins_iam_role.data.aws_iam_policy_document.assume_role_policy: Read complete after 0s [id=2217611055]
module.jenkins_iam_role.data.aws_iam_policy_document.ecr_policy: Read complete after 0s [id=401809238]
module.eks_cluster_and_worker_nodes.data.external.max_pods_calculator["one"]: Read complete after 1s [id=-]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # module.compute.aws_eip.jenkins_eip will be created
  + resource "aws_eip" "jenkins_eip" {
      + allocation_id        = (known after apply)
      + arn                  = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = (known after apply)
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + ptr_record           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + tags_all             = (known after apply)
      + vpc                  = (known after apply)
    }

  # module.compute.aws_instance.jenkins_docker_server will be created
  + resource "aws_instance" "jenkins_docker_server" {
      + ami                                  = "ami-060e277c0d4cce553"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = "jenkins"
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t3.small"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
```

```
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [3m40s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [3m51s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m1s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m11s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m21s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m31s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m41s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [4m51s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m1s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m11s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m21s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m31s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m41s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [5m51s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m1s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m11s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m21s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m31s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m41s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [6m51s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [7m1s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [7m11s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Still creating... [7m21s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_cluster.eks_cluster: Creation complete after 7m25s [id=sd2793-devops-eks-cluster]
module.eks_cluster_and_worker_nodes.aws_launch_template.eks["one"]: Creating...
module.eks_cluster_and_worker_nodes.aws_launch_template.eks["one"]: Creation complete after 0s [id=lt-015aeac3186569f3e]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Creating...
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [10s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [20s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [30s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [40s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [50s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m0s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m10s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m20s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m30s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m40s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Still creating... [1m50s elapsed]
module.eks_cluster_and_worker_nodes.aws_eks_node_group.node_groups["one"]: Creation complete after 2m0s [id=sd2793-devops-eks-cluster:node-app]

Apply complete! Resources: 44 added, 0 changed, 0 destroyed.

Outputs:

ecr_repository_urls = [
  "010438499500.dkr.ecr.ap-southeast-1.amazonaws.com/backend",
  "010438499500.dkr.ecr.ap-southeast-1.amazonaws.com/frontend",
]
eks = {
  "cluster_endpoint" = "sd2793-devops-eks-cluster"
  "eks_cluster_sg_id" = "sg-04fe4565ae2466360"
  "max_pods_for_node_groups" = {
    "one" = tomap({
      "max_pods" = "11"
    })
  }
}
jenkins_instance_details = {
  "iam_profile" = "jenkins"
  "instance_id" = "i-042e2923a70652135"
  "jenkins_sg_id" = "sg-08ac63cabf1f60289"
  "public_id" = "13.251.17.164"
}
```

V. Jenkins

1. Setup

1. Get the **public_ip** in jenkins_instance_details.public_ip output from terraform
2. Open the browser http://jenkins_instance_details.public_ip:8080

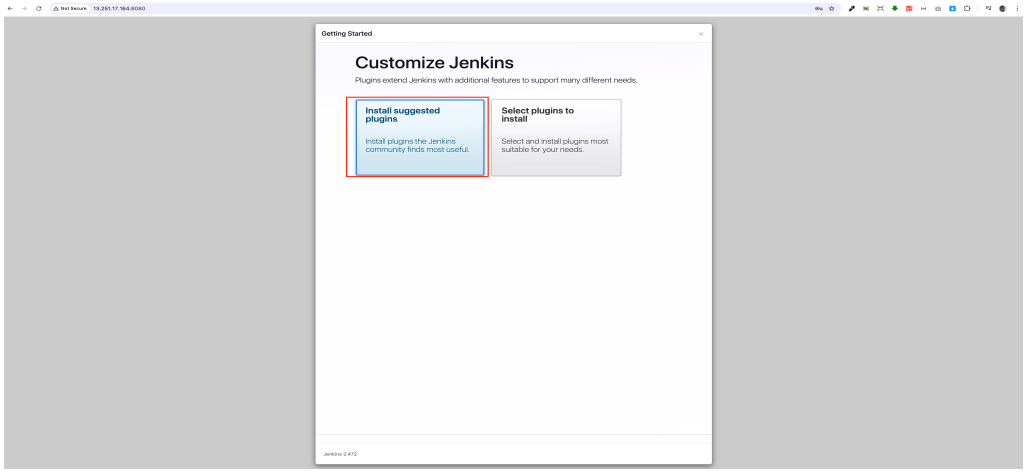


3. Access SSH to server and get the Admin password

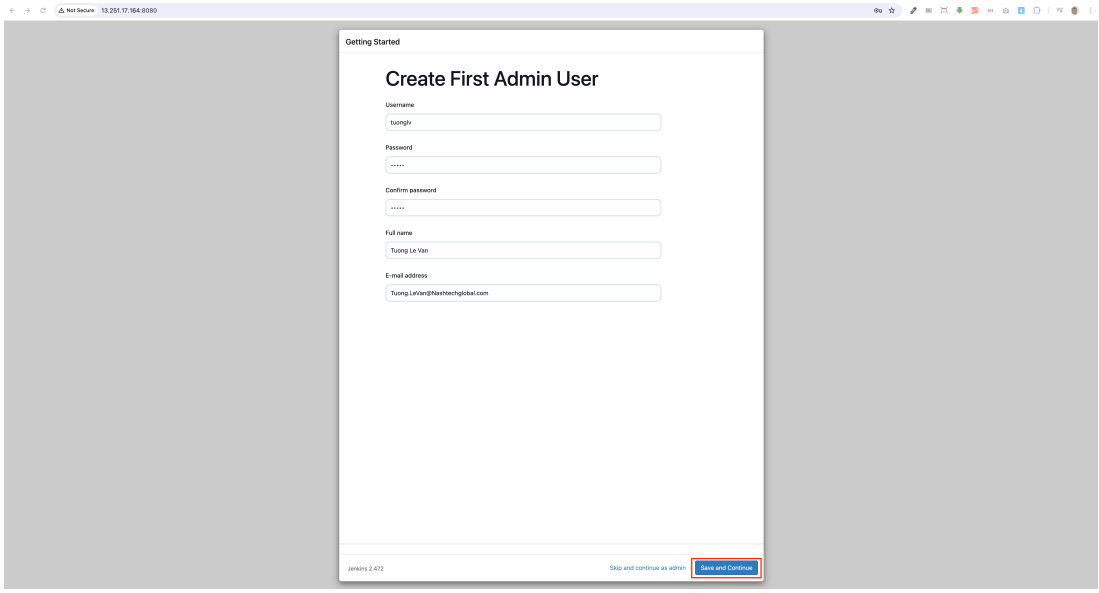
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

```
ubuntu@ip-10-0-4-146:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
666a07f314c344f59e174d954d7a21ca
```

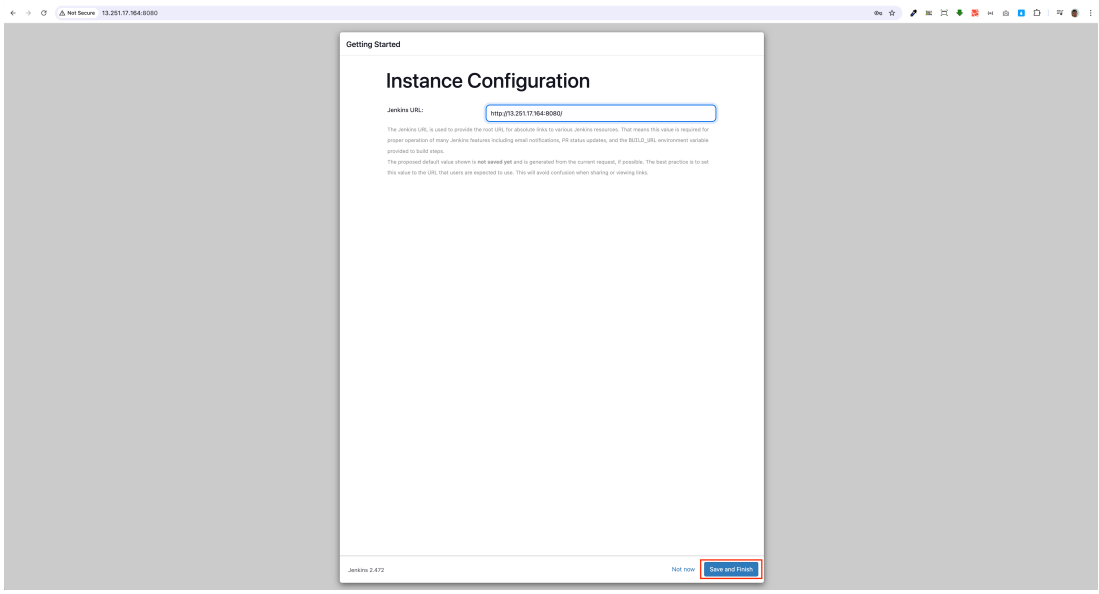
4. Copy the Admin Password output and enter to Administrator Password > Click **Continue** button
5. Choose **Install suggested plugins**



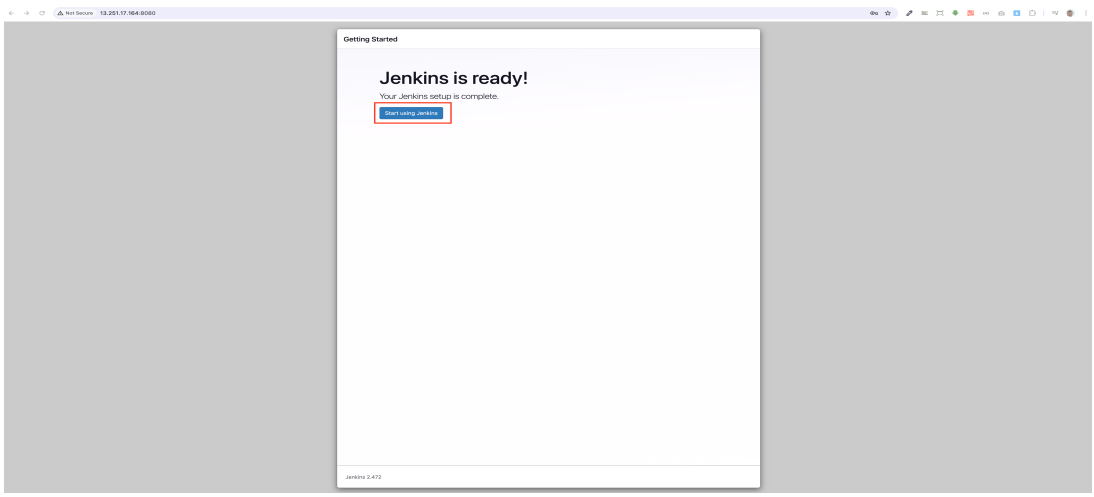
6. Create a new admin user and click **Save and Continue** button



7. Click **Save and Finish** button

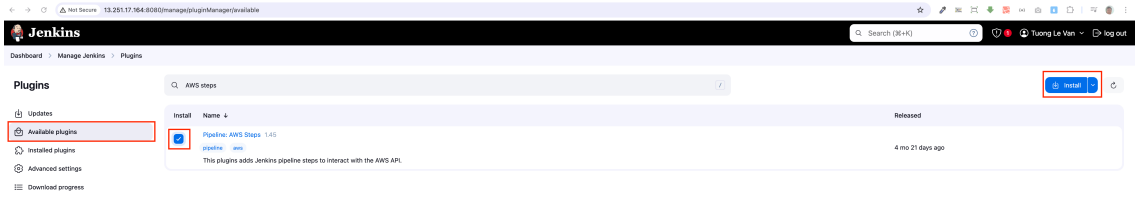


8. Click **Start using Jenkins** button



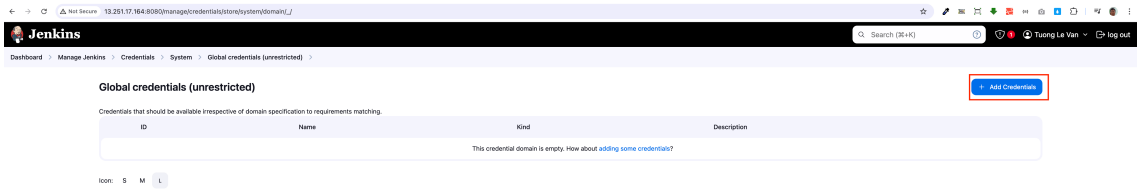
2. Install Plugins Pipeline: AWS Steps

1. Go to Dashboard > Manager Jenkins > Plugins
2. Available Plugins > Enter **AWS Steps** > Checked Install and click **Install** button

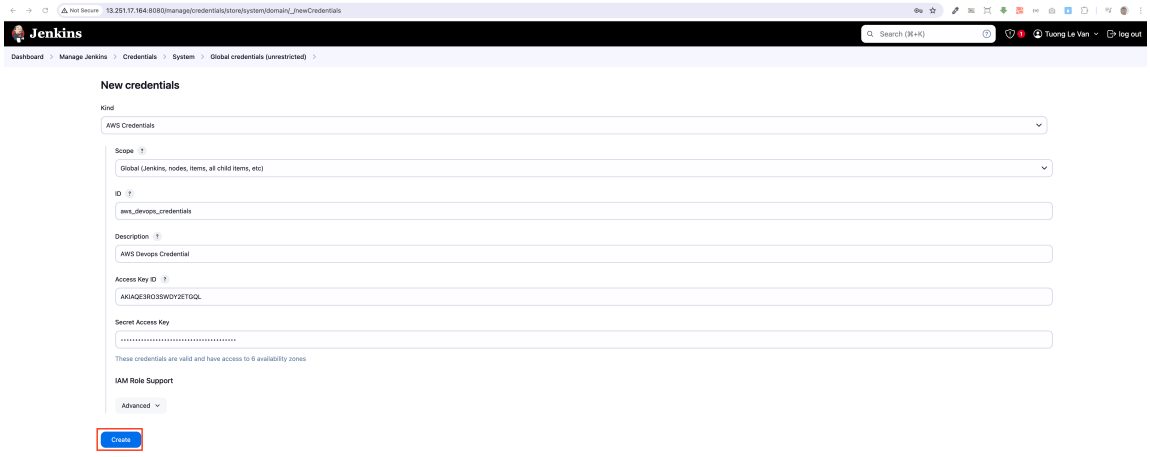


3. Setup AWS Credential Global

1. Go to Dashboard > Manager Jenkins > Credentials > System > Global Credentials > Click **Add credentials** button

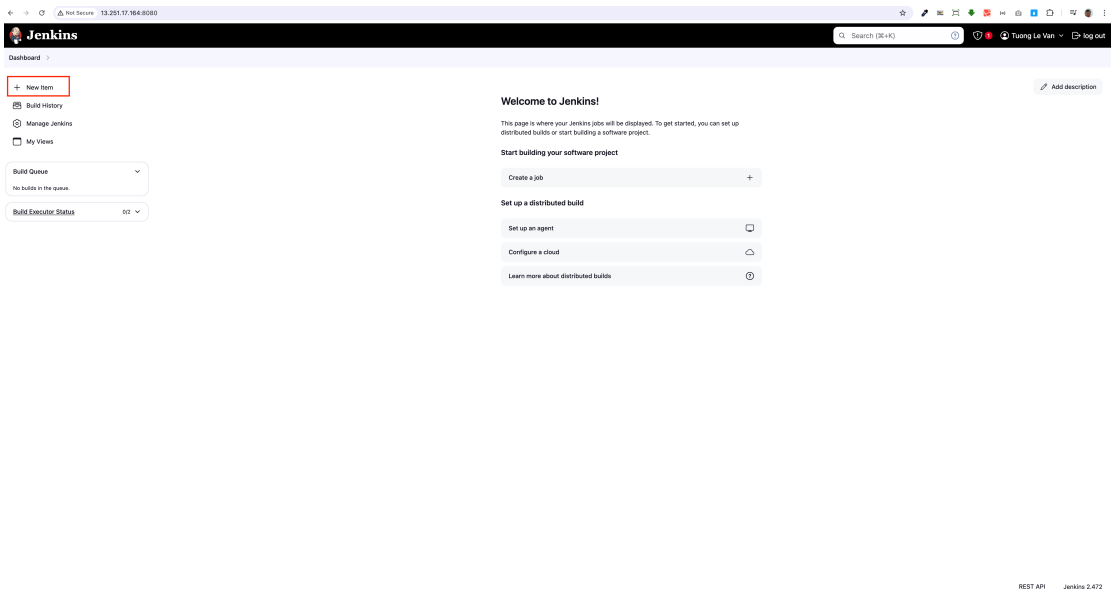


2. Enter the credentials and click **Create** button

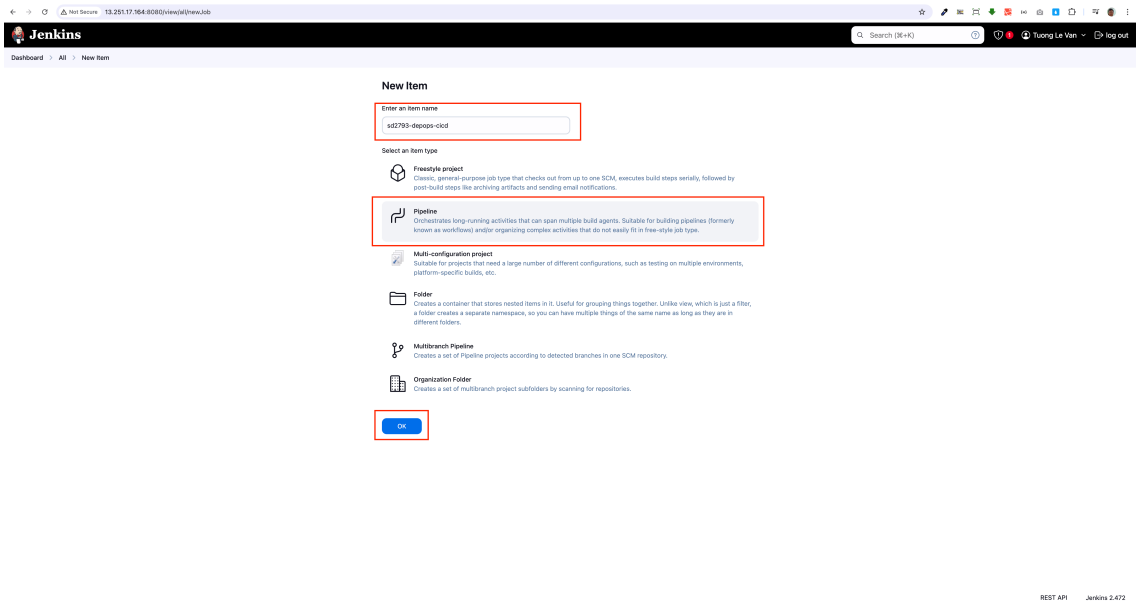


4. Setup CI-CD

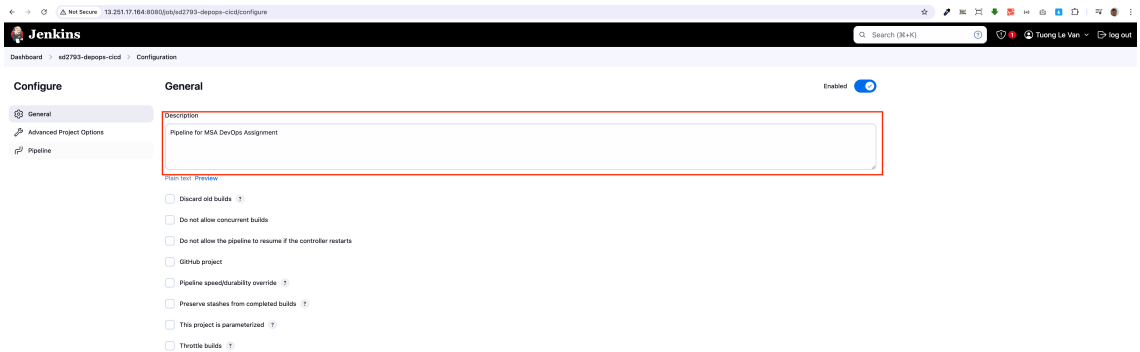
1. Go to Dashboard navigation and click **New Item** button



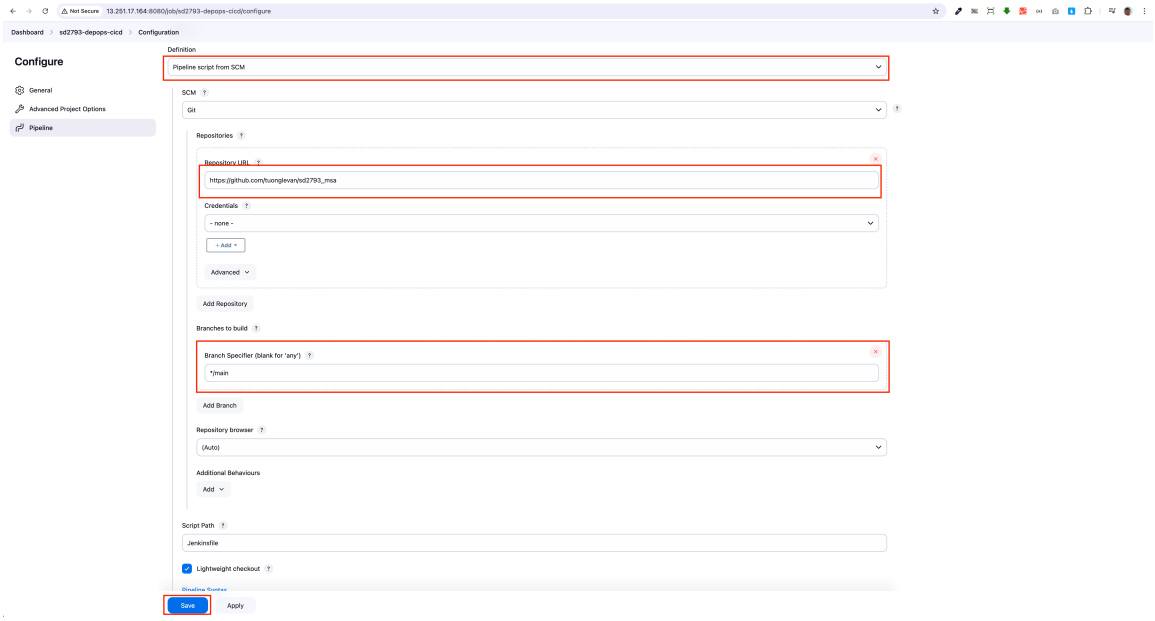
2. Enter the name of pipeline > Choose Pipeline and Click **Ok** button



3. Configure > General tab > Generate > Enter the Description name



4. Configure > Pipeline > Setup Pipeline SCM > Click **Save** button

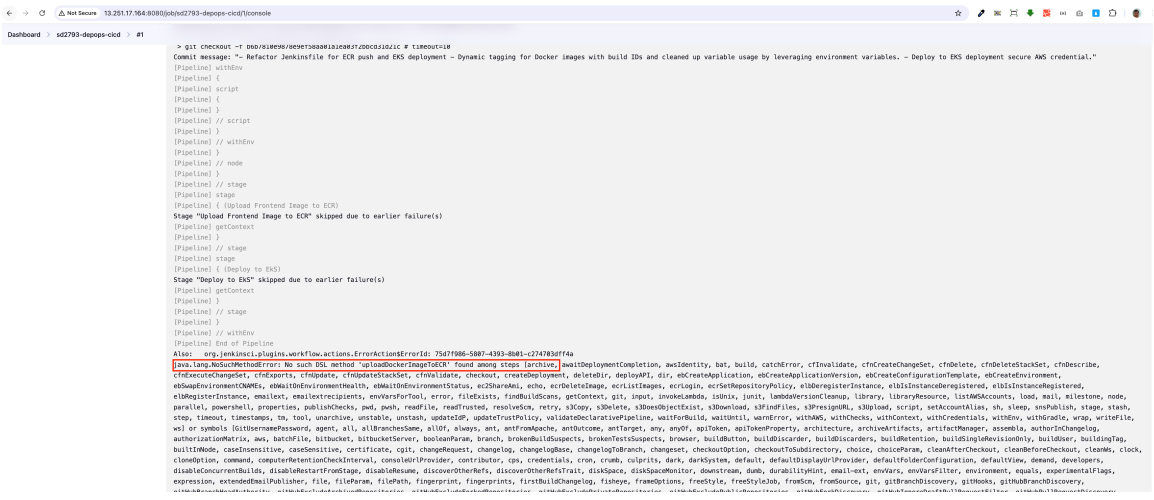


5. Build Image and Push to ECR

1. Go to Dashboard > Choose project name > Click button **Build Now**

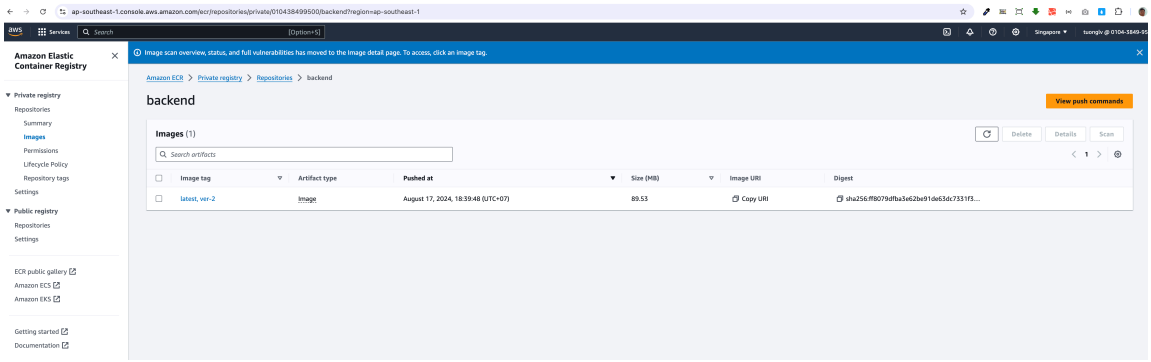
2. Check Build > Read the **Console Output** of the build number

    2.1. If have some issues, need to fixed and make another build

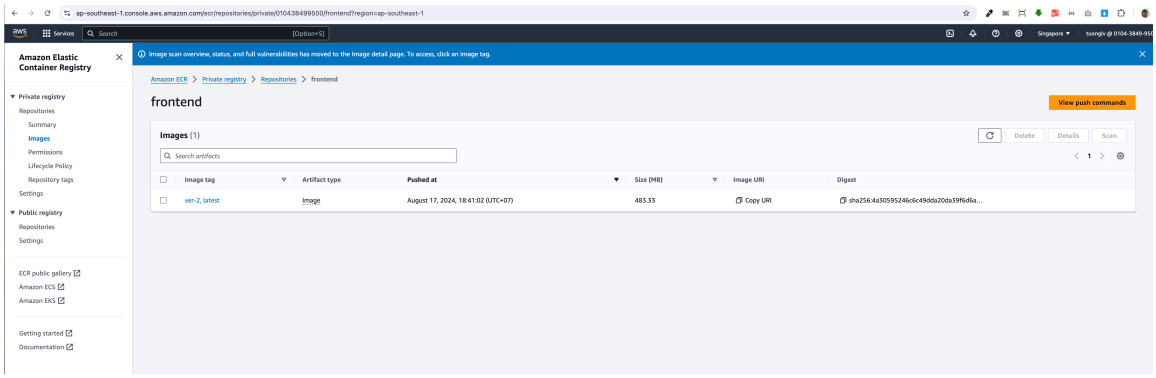


3. Check ECR > Login AWS console > Amazon ECR > Private Registry > Repositories

3.1. backend



3.2. frontend



4. Verify Pipeline

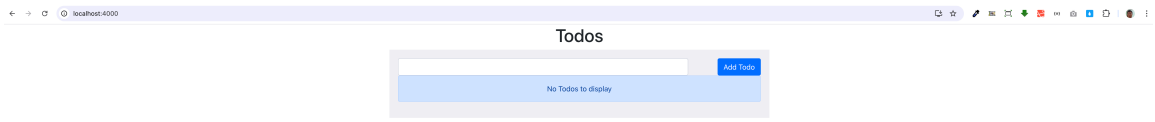
5. Verify EKS

```
➜  sd2793_aws_infastructure git:(main) ✗ aws eks update-kubeconfig --name sd2793-devops-eks-cluster
Updated context arn:aws:eks:ap-southeast-1:010438499500:cluster/sd2793-devops-eks-cluster in /Users/tuonglv/.kube/config
```

```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
backend-984f8b86d-zxwft     1/1     Running   0          6m43s
frontend-646f49fb59-9sqmr   1/1     Running   0          6m42s
mongo-0                     0/1     Pending   0          6m44s
```

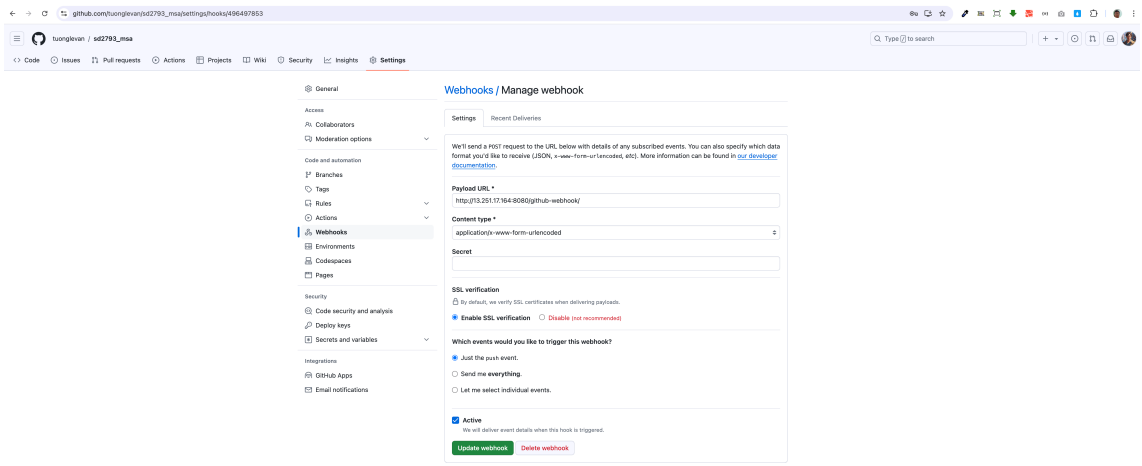
5.1. Verify Frontend App

```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl get services
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)     AGE
backend      ClusterIP   172.20.146.210   <none>        3000/TCP    12m
frontend     ClusterIP   172.20.195.210   <none>        3000/TCP    12m
kubernetes   ClusterIP   172.20.0.1       <none>        443/TCP     3h3m
mongo        ClusterIP   172.20.153.124   <none>        27017/TCP   12m
➜  sd2793_aws_infastructure git:(main) ✗ kubectl port-forward service/frontend 4000:3000
Forwarding from 127.0.0.1:4000 -> 3000
Forwarding from [::1]:4000 -> 3000
```

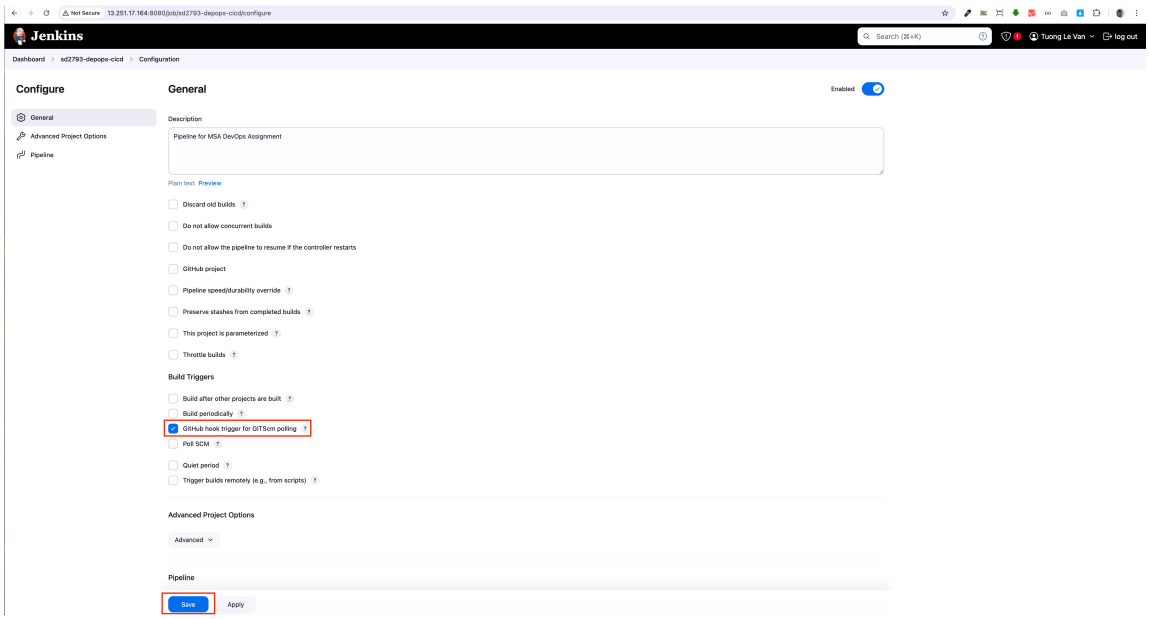


6. Config Webhook

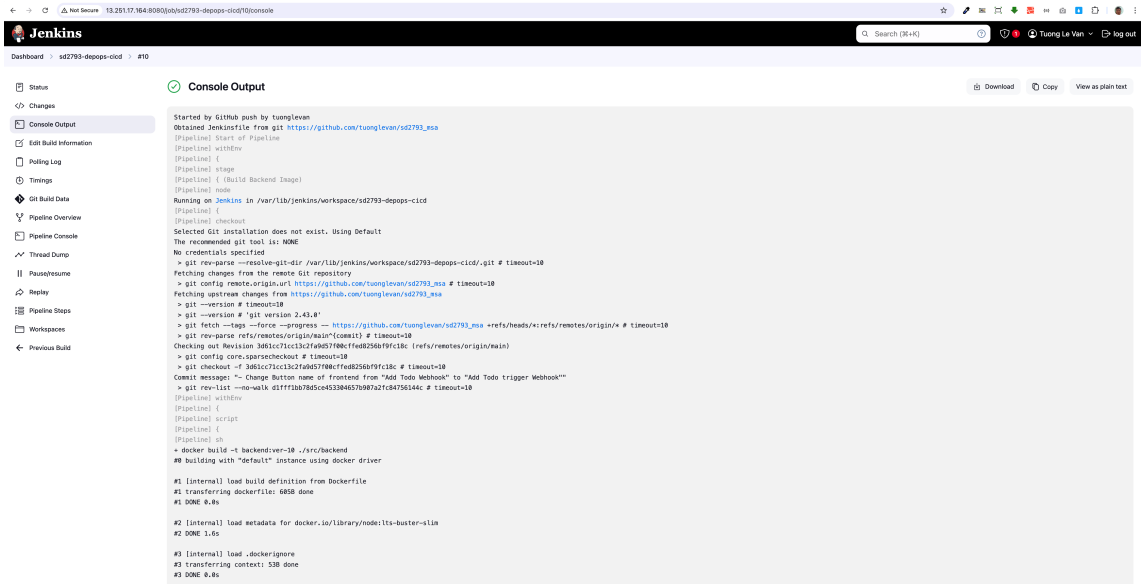
1. Go to repository sd2793_msa > Setting > Webhooks > click **Add webhook** button



2. Go to Dashboard in jenkins > Choose pipeline project > configure > general > checked **Github hook** … in Build Triggers > click **Save** button



3. Changes code in MSA repo and checked Pipeline auto trigger



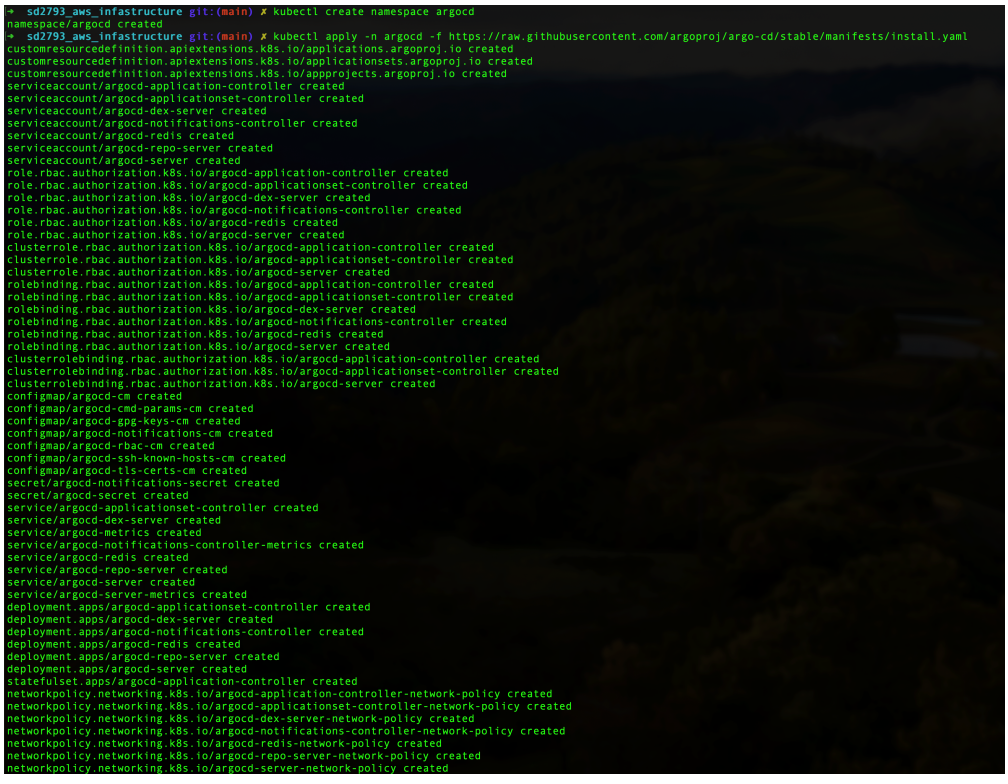
VI. ArgoCD

1. Install

1. Open terminal and run commands

1.1. kubectl create namespace argocd

1.2. kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml



2. Port Forward and Access ArgoCD UI

1. Port forward

    1.1. kubectl port-forward svc/argocd-server -n argocd 8080:443

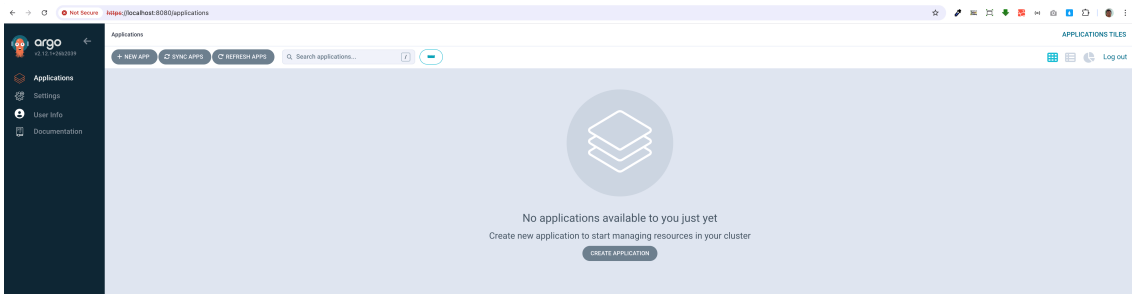
```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl port-forward svc/argocd-server -n argocd 8080:443
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
```

2. Get password admin

    2.1. kubectl -n argocd get secret argocd-initial-admin-secret -ojsonpath={.data.password} | base64 –decode

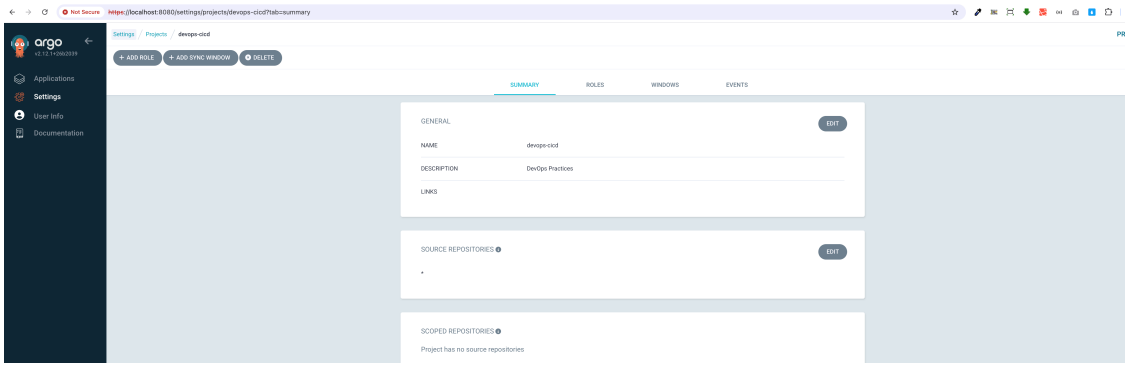
```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl -n argocd get secret argocd-initial-admin-secret -ojsonpath={.data.password} | base64 --decode
QpLaWYKkbP-rm75X%
```

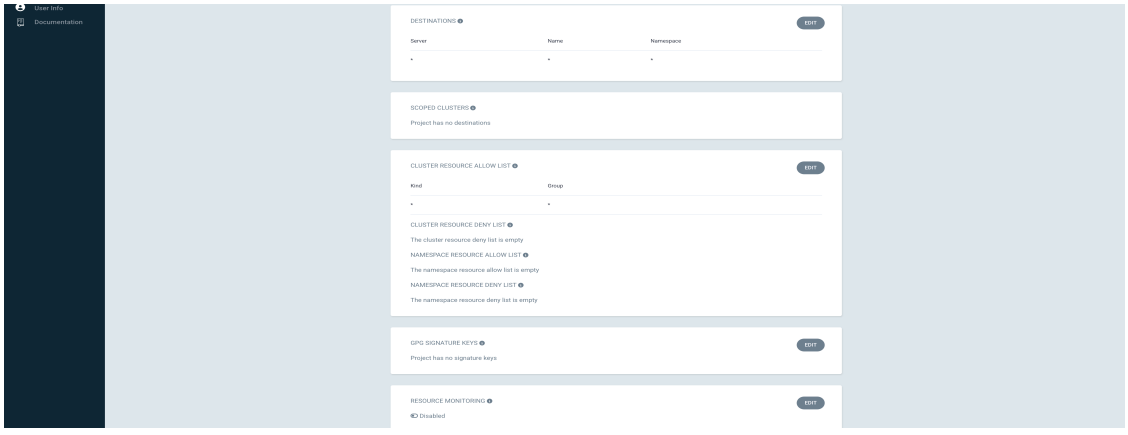
3. Open browser and access to https://localhost:8080



3. Setup Application

1. Create a Project



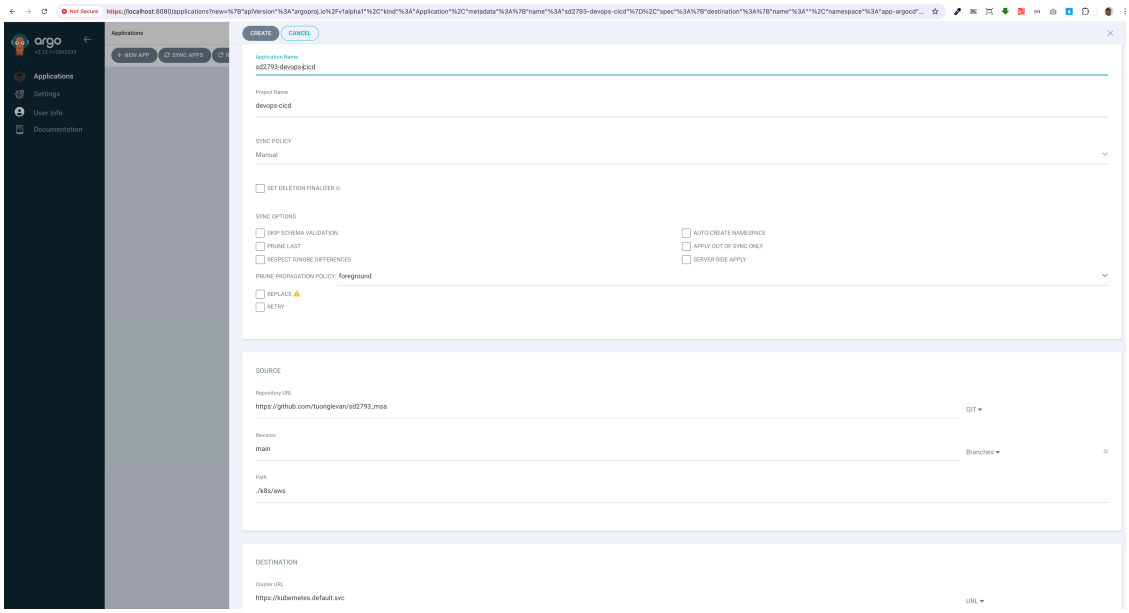


2. Create a new namespace for application

2.1. kubectl create namespace app-argocd

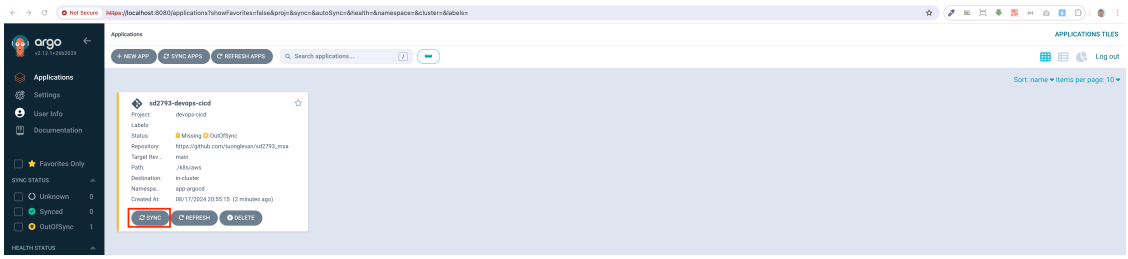
```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl create namespace app-argocd
namespace/app-argocd created
```

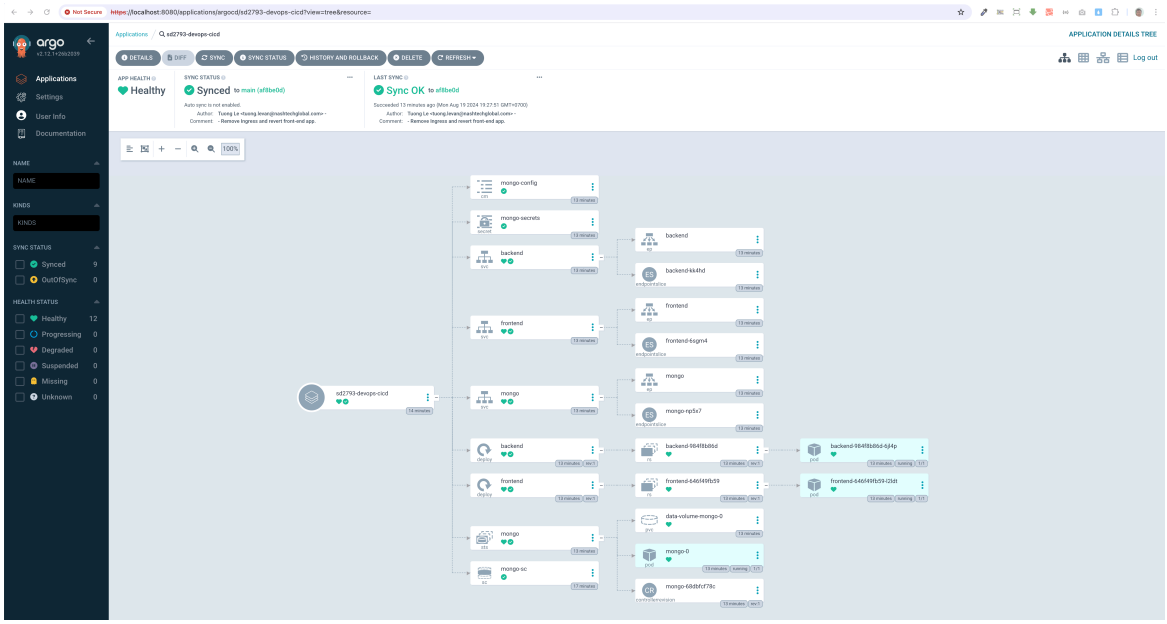
3. Create a new Application



4. Sync ArgoCD application with github

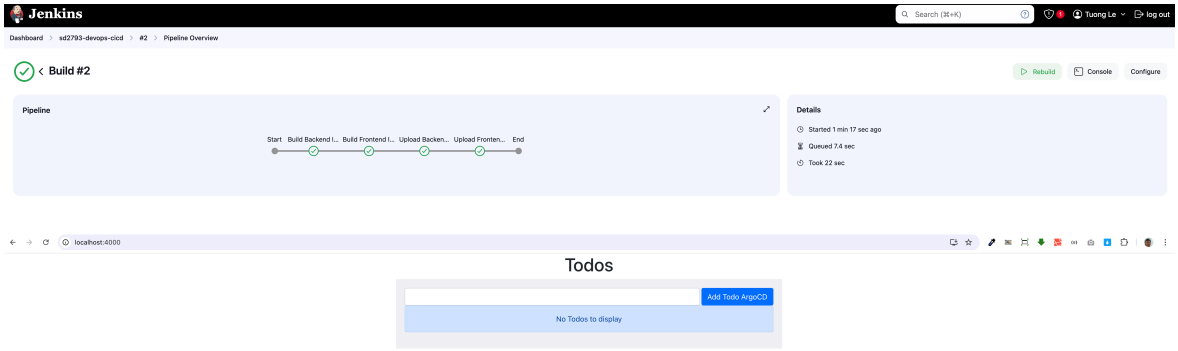
4.1. Go to Applications > choose application name > click **SYNC** button





4. Remove Stage CD in Jenkins and Update UI button

```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl port-forward svc/frontend 4000:3000 -n app-argocd
Forwarding from 127.0.0.1:4000 -> 3000
Forwarding from [::1]:4000 -> 3000
Handling connection for 4000
Handling connection for 4000
Handling connection for 4000
```



VII. Prometheus and Grafana

1. Installation

```
➜  sd2793_aws_infastructure git:(main) ✗ helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories
➜  sd2793_aws_infastructure git:(main) ✗ helm repo add stable https://charts.helm.sh/stable
"stable" has been added to your repositories
➜  sd2793_aws_infastructure git:(main) ✗ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "prometheus-community" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete. ⎈Happy Helming!⎈
➜  sd2793_aws_infastructure git:(main) ✗ helm install prometheus prometheus-community/kube-prometheus-stack
NAME: prometheus
LAST DEPLOYED: Sun Aug 18 16:24:46 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
kube-prometheus-stack has been installed. Check its status by running:
  kubectl --namespace default get pods -l "release=prometheus"

Visit https://github.com/prometheus-operator/kube-prometheus for instructions on how to create & configure Alertmanager and Prometheus instances using the Operator.
➜  sd2793_aws_infastructure git:(main) ✗
```

```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl get all
NAME                                                     READY   STATUS    RESTARTS   AGE
pod/alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running   0          2m13s
pod/backend-984f8b86d-pncl7                              1/1     Running   0          9m9s
pod/frontend-646f49fb59-hd8x9                            1/1     Running   0          9m8s
pod/mongo-0                                              0/1     Pending   0          9m10s
pod/prometheus-grafana-785b988595-5dmc7                  3/3     Running   0          2m19s
pod/prometheus-kube-prometheus-operator-bb46f9945-pm2x2  1/1     Running   0          2m19s
pod/prometheus-kube-state-metrics-688d66b5b8-vg9d9       1/1     Running   0          2m19s
pod/prometheus-prometheus-kube-prometheus-prometheus-0   2/2     Running   0          2m12s
pod/prometheus-prometheus-node-exporter-2ktkx            1/1     Running   0          2m19s
pod/prometheus-prometheus-node-exporter-c6npn            1/1     Running   0          2m19s

NAME                                                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                      AGE
service/alertmanager-operated                       ClusterIP   None             <none>        9093/TCP,9094/TCP,9094/UDP   2m13s
service/backend                                     ClusterIP   172.20.202.229   <none>        3000/TCP                     9m9s
service/frontend                                    ClusterIP   172.20.254.127   <none>        3000/TCP                     9m8s
service/kubernetes                                  ClusterIP   172.20.0.1       <none>        443/TCP                      39m
service/mongo                                       ClusterIP   172.20.252.8     <none>        27017/TCP                    9m10s
service/prometheus-grafana                          ClusterIP   172.20.60.203    <none>        80/TCP                       2m19s
service/prometheus-kube-prometheus-alertmanager     ClusterIP   172.20.76.128    <none>        9093/TCP,8080/TCP            2m19s
service/prometheus-kube-prometheus-operator         ClusterIP   172.20.135.197   <none>        443/TCP                      2m19s
service/prometheus-kube-prometheus-prometheus       ClusterIP   172.20.20.190    <none>        9090/TCP,8080/TCP            2m19s
service/prometheus-kube-state-metrics               ClusterIP   172.20.1.206     <none>        8080/TCP                     2m19s
service/prometheus-operated                         ClusterIP   None             <none>        9090/TCP                     2m12s
service/prometheus-prometheus-node-exporter         ClusterIP   172.20.64.106    <none>        9100/TCP                     2m19s

NAME                                                    DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
daemonset.apps/prometheus-prometheus-node-exporter      2         2         2       2            2           kubernetes.io/os=linux   2m19s

NAME                                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/backend                             1/1     1            1           9m9s
deployment.apps/frontend                            1/1     1            1           9m8s
deployment.apps/prometheus-grafana                  1/1     1            1           2m19s
deployment.apps/prometheus-kube-prometheus-operator 1/1     1            1           2m19s
deployment.apps/prometheus-kube-state-metrics       1/1     1            1           2m19s

NAME                                                          DESIRED   CURRENT   READY   AGE
replicaset.apps/backend-984f8b86d                             1         1         1       9m9s
replicaset.apps/frontend-646f49fb59                           1         1         1       9m8s
replicaset.apps/prometheus-grafana-785b988595                 1         1         1       2m19s
replicaset.apps/prometheus-kube-prometheus-operator-bb46f9945 1         1         1       2m19s
replicaset.apps/prometheus-kube-state-metrics-688d66b5b8      1         1         1       2m19s

NAME                                                               READY   AGE
statefulset.apps/alertmanager-prometheus-kube-prometheus-alertmanager   1/1     2m13s
statefulset.apps/mongo                                             0/1     9m10s
statefulset.apps/prometheus-prometheus-kube-prometheus-prometheus  1/1     2m12s
```

2. Monitoring

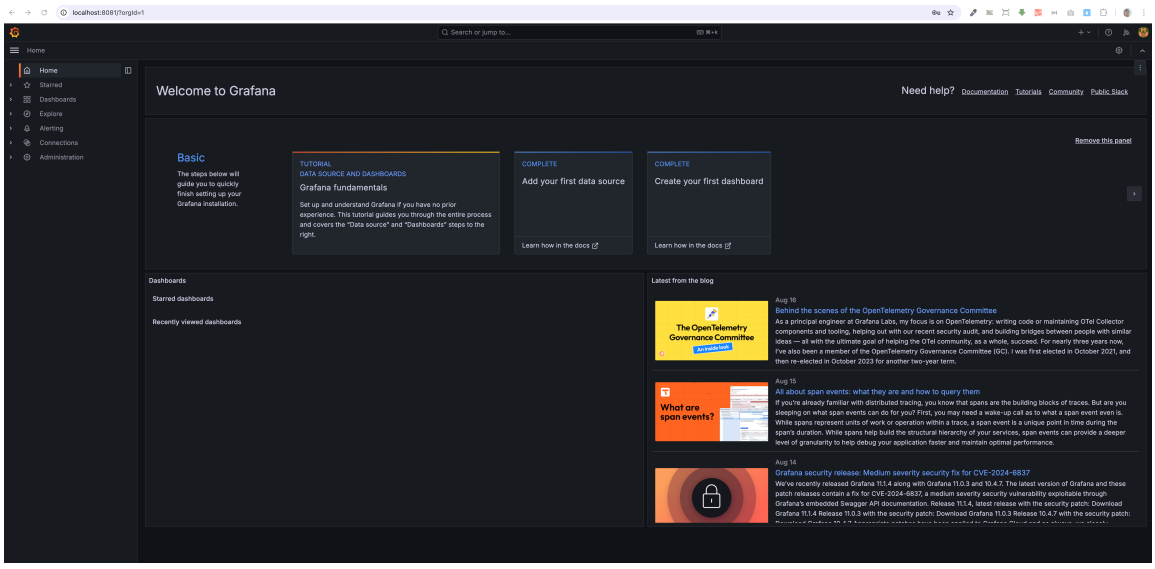
1. Port Forward and Access UI

```
➜  sd2793_aws_infastructure git:(main) ✗ kubectl port-forward service/prometheus-grafana 8081:80
Forwarding from 127.0.0.1:8081 -> 3000
Forwarding from [::1]:8081 -> 3000
```

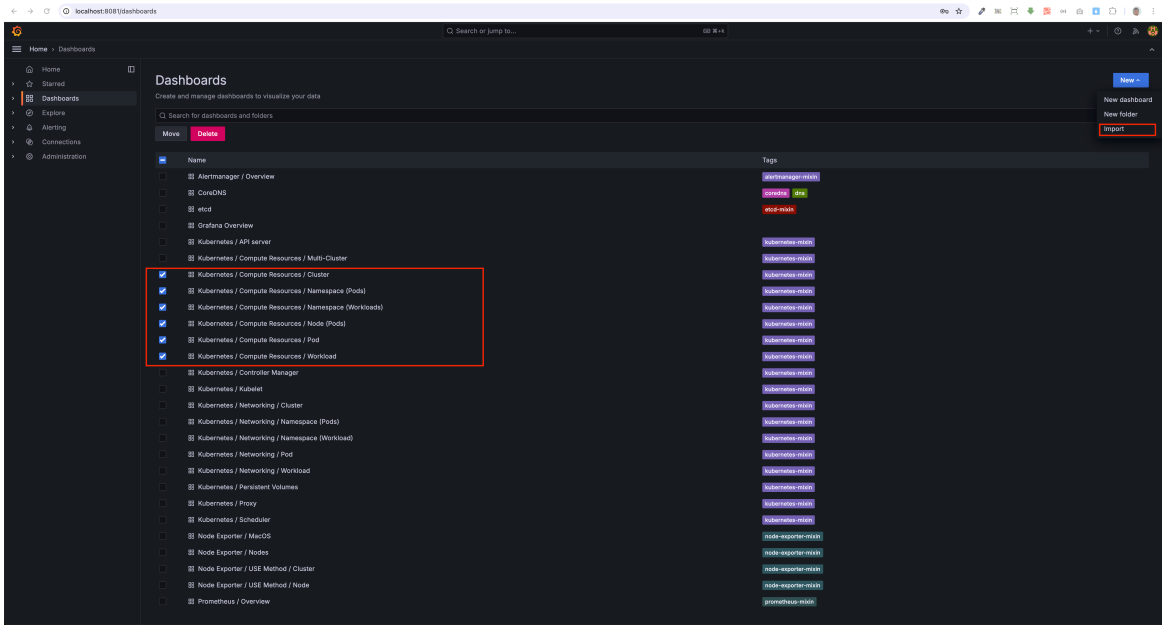
2. Open the Brow and access to http://localhost:8081/ with username and password default

    2.1. Username: **admin**

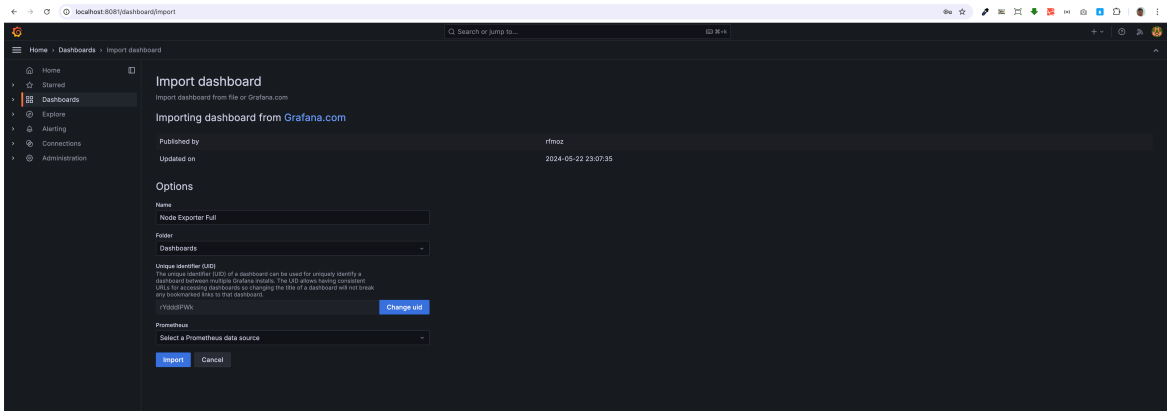
    2.2. Password: **prom-operator**



3. Import Metrics to Dashboard



Dashboard ID: 1860 [Node Exporter Full]



4. Monitoring in the Dashboard UI.