Objective

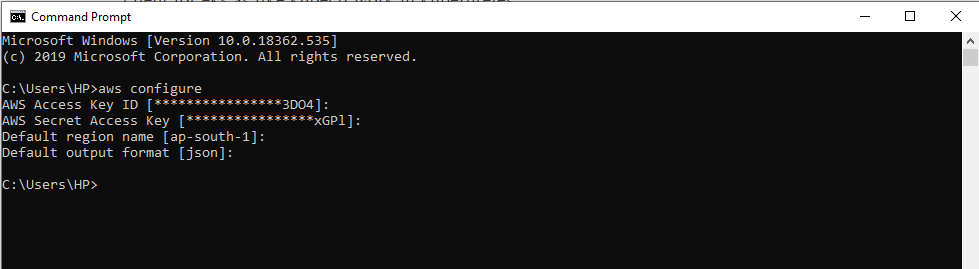
In this tasks we will see about the EKS and its uses case how it use and how it configure what will be the processor for creating such kind of cluster and how many type of cluster we can make.and then after this we will Integrate it with EBS,EFS and ELB.After doing Integration we can launch a pod that will be Wordpress with MySQL .for this site we first have to configure MySQL and then Wordpress.After this we can view the HELM usecase of HELM How to configure it and many more .After this we can launch a prometheus server on the eks and then integrate it with grafana . and at last we will about the Farget-Cluster.

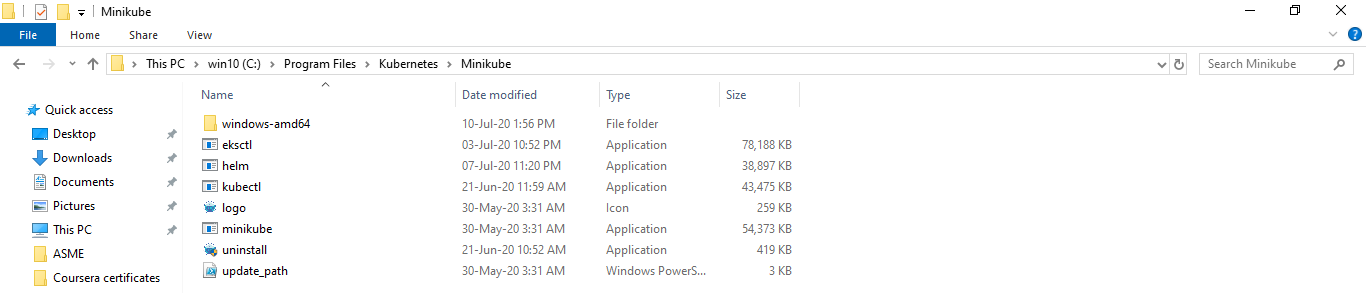
Tools Required:

* AWS CLI
* Eksctl
* Helm
* Tiller
* Kubectl

SOLUTION

First, we have to create an IAM user with AdministratorAccess and uses there credentials to login in the AWSCLI and then download EKSCTL Software to use EKSCTL command and then set its env path to our system path so we can use it from anywhere .eksctl is work as a client for eks as like kubectl work in kubernretes.





After this now we have to configure our cluster over the EKS we can use this following code to configure our EKS cluster , save the file with .yml extension

apiVersion: eksctl.io/v1alpha5

kind: ClusterConfig

metadata:

  name: vishalcluster

  region: ap-south-1

nodeGroups:

  - name: ng-1

    instanceType: t2.micro

    desiredCapacity: 2

    ssh:

          publicKeyName: mykey11

  - name: ng-2

    instanceType: t2.micro

    desiredCapacity: 2

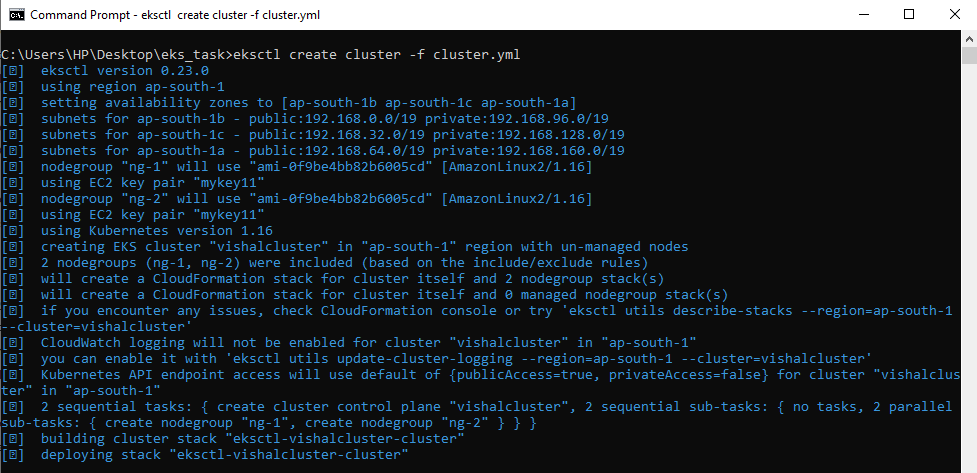
    ssh:

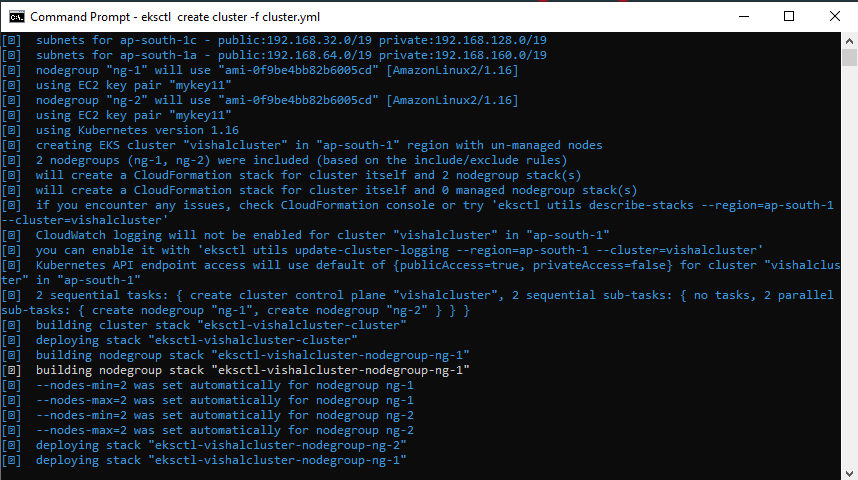
          publicKeyName: mykey11

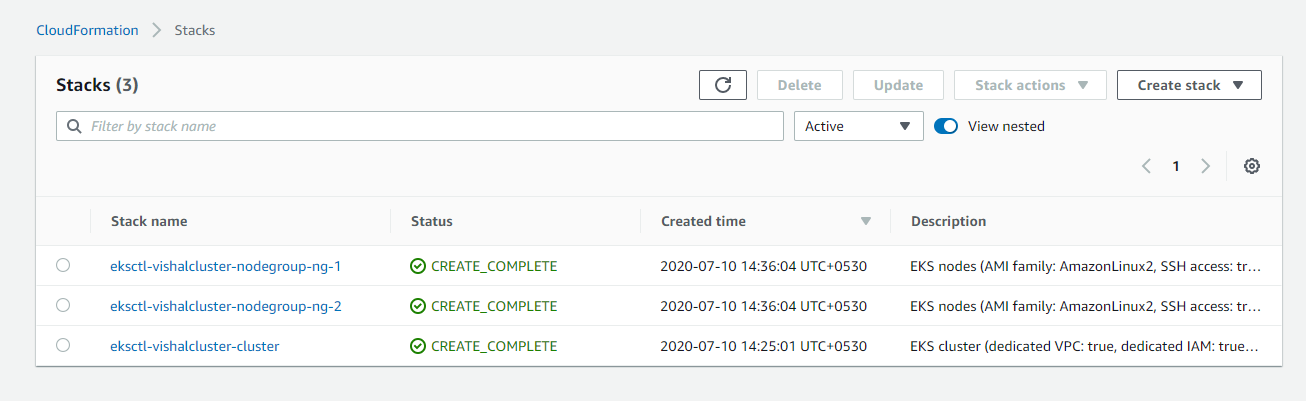
In this ABHAYCLUSTER we create 2 Nodegroup with t2,micro instance type that is free-tier.and use our key to login and perform some task on our nodes.

These are the Nodes of our cluster our master is managed by our AWS EKS here we will create our slave or nodes where our client can launch their apps.

We need to understand that behind the scene eksctl will take the help of cloud formation so there will be some stack created for to launch nodes in the AWS.

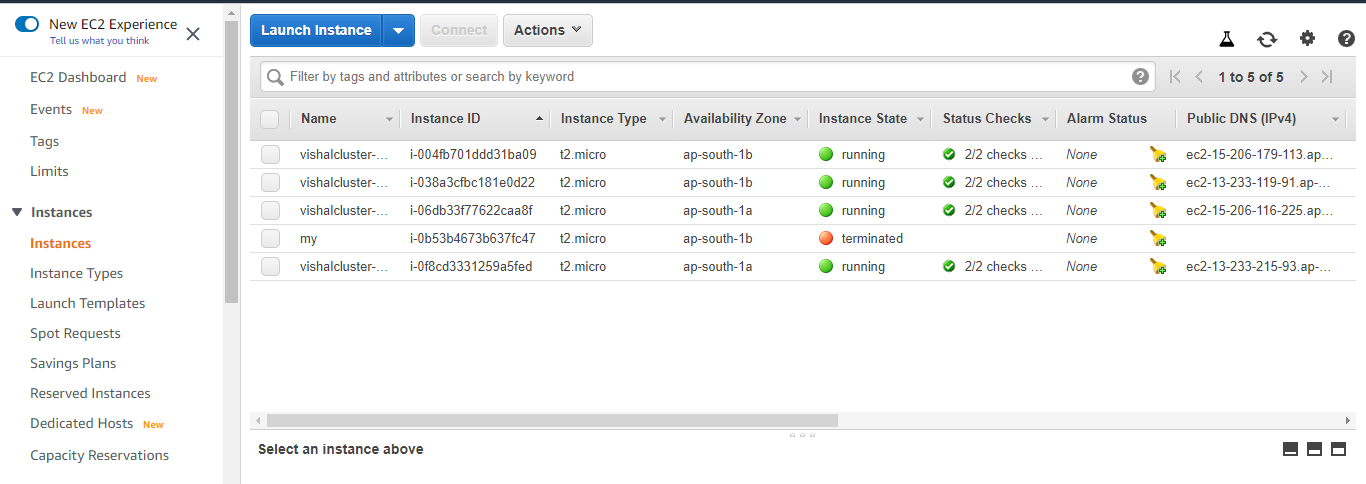




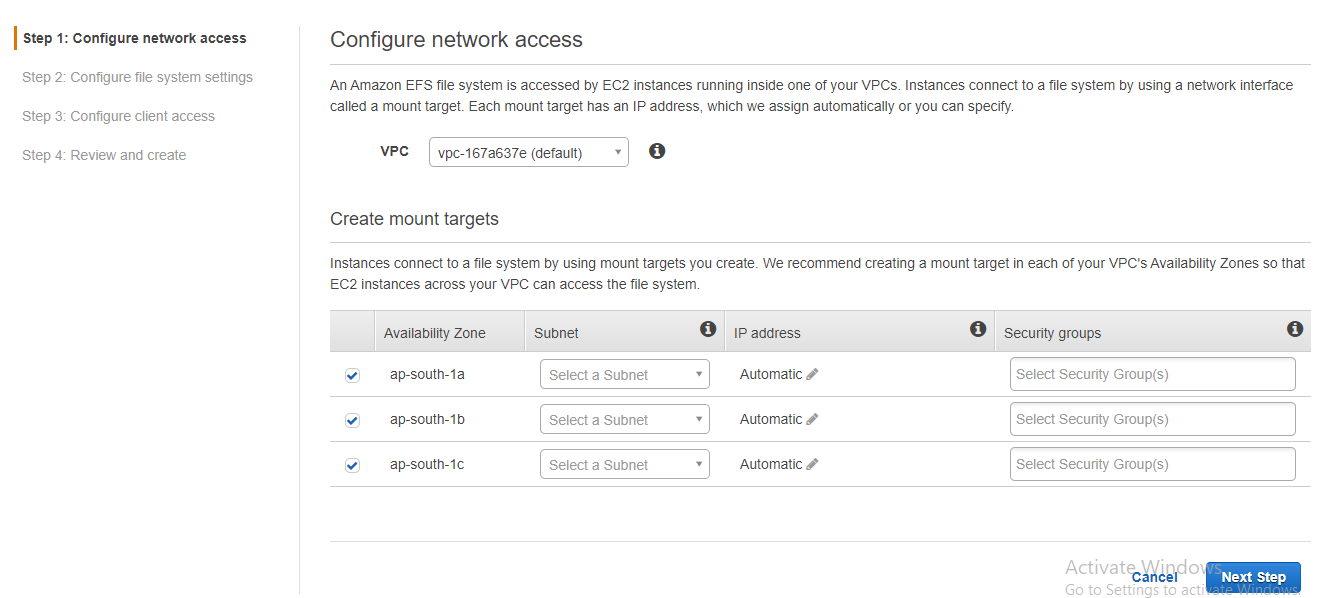


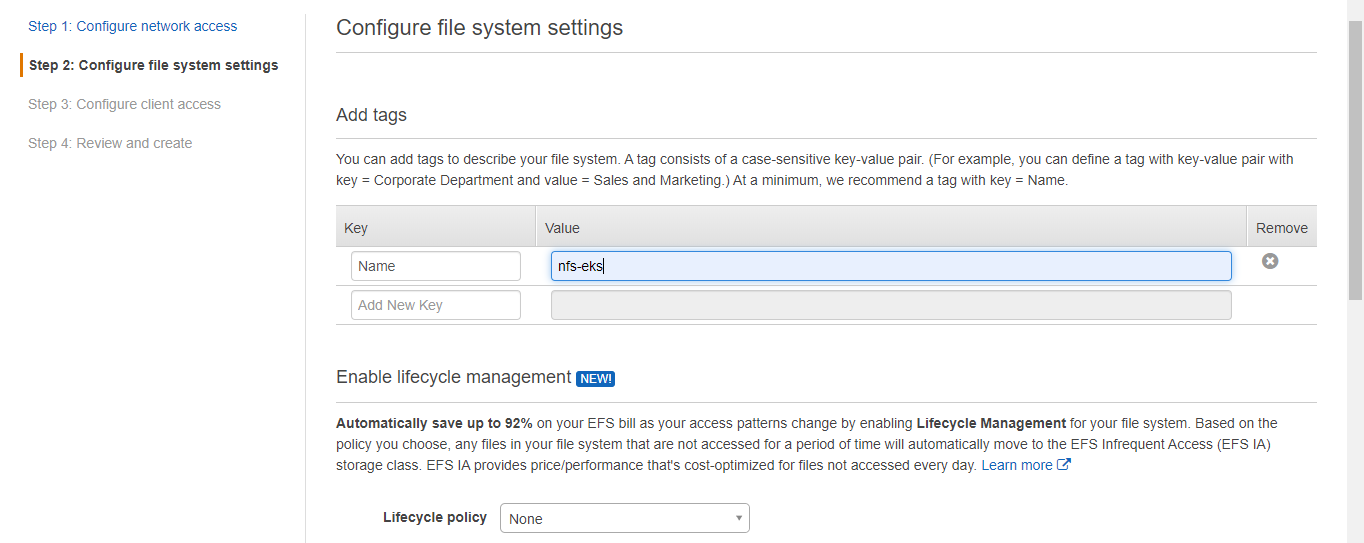
After Successfully creating clusters we have to update our kubeconfig file so that it can use the AWS EKS cluster instead of our minikube.

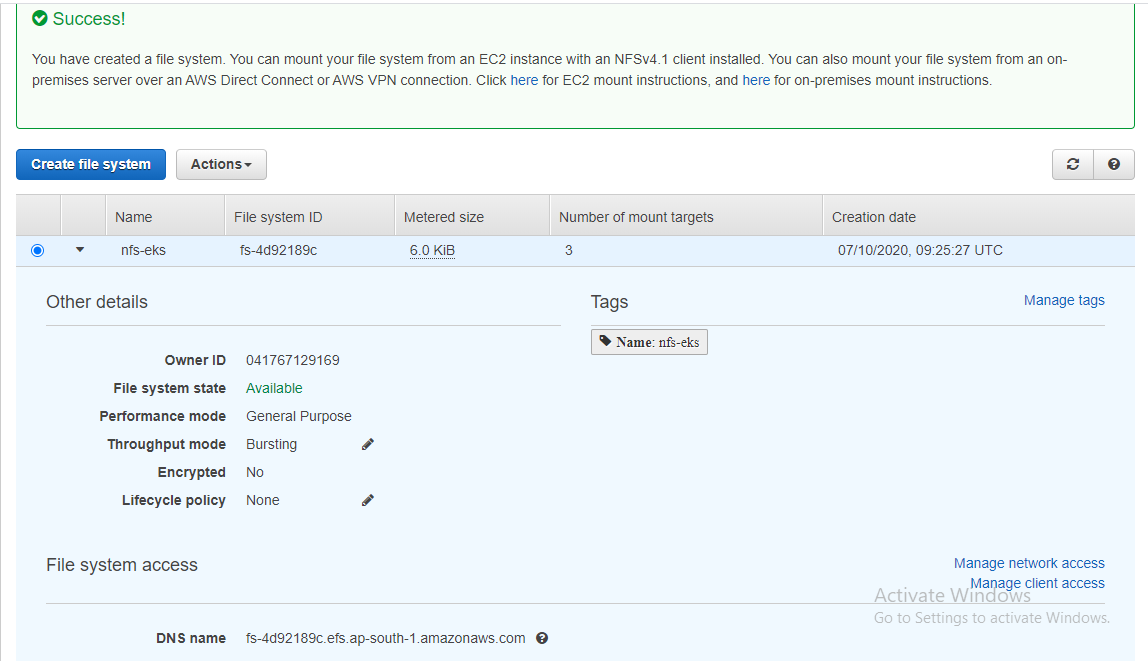
aws eks update-kubeconfig  --name vishalcluster



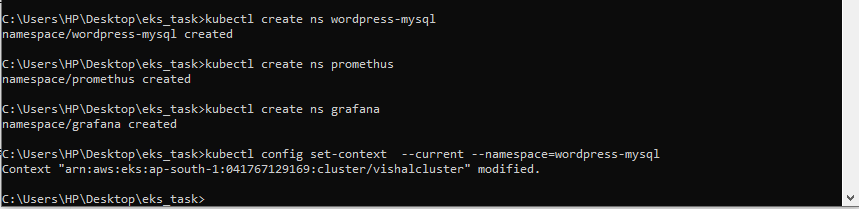
Now, we need to create one AWS Elastic file system. Go to your AWS console -> EFS and then create one file system.







Now just check that our command is working or not run "kubectl get pods" , if it works give output without any error it means it's okay. Now I am gonna launch 3 things (WORDPRESS , MYSQL) , Prometheus , Grafana so i am creating 3 different namespaces for them its better for management. I have created "wordpress-mysql" as my current namesapce



Use EFS provisioner to create deployment of pods. The code for deployment is

kind: Deployment

apiVersion: apps/v1

metadata:

  name: efs-provisioner

spec:

  selector:

    matchLabels:

      app: efs-provisioner

  replicas: 1

  strategy:

    type: Recreate

  template:

    metadata:

      labels:

        app: efs-provisioner

    spec:

      containers:

        - name: efs-provisioner

          image: quay.io/external\_storage/efs-provisioner:v0.1.0

          env:

            - name: FILE\_SYSTEM\_ID

              value: fs-4d92189c

            - name: AWS\_REGION

              value: ap-south-1

            - name: PROVISIONER\_NAME

              value: gau-prov/aws-efs

          volumeMounts:

            - name: pv-volume

              mountPath: /persistentvolumes

      volumes:

        - name: pv-volume

          nfs:

            server: fs-4d92189c.efs.ap-south-1.amazonaws.com

            path: /

Save this code in a file with .yml extension and run the following command in the cmd

kubectl create -f efs.yml

After this, create one ClusterRoleBinding file with .yml extension.

apiVersion: rbac.authorization.k8s.io/v1beta1

kind: ClusterRoleBinding

metadata:

  name: nfs-provisioner-role-binding

subjects:

  - kind: ServiceAccount

    name: default

    namespace: wordpress-mysql

roleRef:

  kind: ClusterRole

  name: cluster-admin

  apiGroup: rbac.authorization.k8s.io

Now, create a storage file with .yml extension

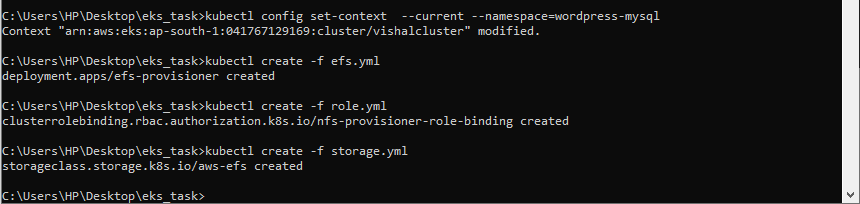
kind: StorageClass

apiVersion: storage.k8s.io/v1

metadata:

  name: aws-efs

provisioner: gau-prov/aws-efs



EKS is integrated with EFS!

Now, we create persistent volume claim(PVC) using following code

kind: PersistentVolumeClaim

apiVersion: v1

metadata:

  name: efs-wordpress

  annotations:

    volume.beta.kubernetes.io/storage-class: "aws-efs"

spec:

  accessModes:

    - ReadWriteMany

  resources:

    requests:

      storage: 10Gi

---

kind: PersistentVolumeClaim

apiVersion: v1

metadata:

  name: efs-mysql

  annotations:

    volume.beta.kubernetes.io/storage-class: "aws-efs"

spec:

  accessModes:

    - ReadWriteMany

  resources:

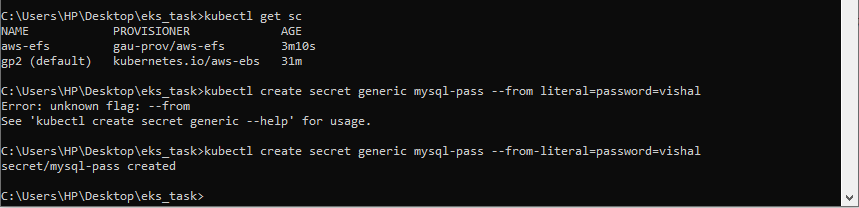
    requests:

      storage: 10Gi

Run this command in cmd

kubectl create -f pvc.yml

Now, we will create a secret for mysql and wordress



Now can deploy my MySQL server

apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2

kind: Deployment

metadata:

  name: wordpress-mysql

  labels:

    app: wordpress

spec:

  selector:

    matchLabels:

      app: wordpress

      tier: mysql

  strategy:

    type: Recreate

  template:

    metadata:

      labels:

        app: wordpress

        tier: mysql

    spec:

      containers:

      - image: mysql:5.6

        name: mysql

        env:

        - name: MYSQL\_ROOT\_PASSWORD

          valueFrom:

            secretKeyRef:

              name: mysql-pass

              key: password

        ports:

        - containerPort: 3306

          name: mysql

        volumeMounts:

        - name: mysql-persistent-storage

          mountPath: /var/lib/mysql

      volumes:

      - name: mysql-persistent-storage

        persistentVolumeClaim:

          claimName: efs-mysql

Using this code we can create a MySQL pod or server by using deployment and use MySQL version 5.6 and get password of MySQL from secret and mount our pvc to its path.

Now we can create service of MySQL

apiVersion: v1

kind: Service

metadata:

  name: wordpress-mysql

  labels:

    app: wordpress

spec:

  ports:

    - port: 3306

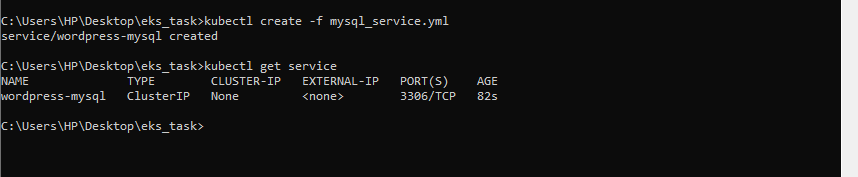
  selector:

    app: wordpress

    tier: mysql

  clusterIP: None

here we can make a service of MySQL and give a port 3306



Now we need to deploy wordpress

apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2

kind: Deployment

metadata:

  name: wordpress

  labels:

    app: wordpress

spec:

  selector:

    matchLabels:

      app: wordpress

      tier: frontend

  strategy:

    type: Recreate

  template:

    metadata:

      labels:

        app: wordpress

        tier: frontend

    spec:

      containers:

      - image: wordpress:4.8-apache

        name: wordpress

        env:

        - name: WORDPRESS\_DB\_HOST

          value: wordpress-mysql

        - name: WORDPRESS\_DB\_PASSWORD

          valueFrom:

            secretKeyRef:

              name: mysql-pass

              key: password

        ports:

        - containerPort: 80

          name: wordpress

        volumeMounts:

        - name: wordpress-persistent-storage

          mountPath: /var/www/html

      volumes:

      - name: wordpress-persistent-storage

        persistentVolumeClaim:

          claimName: efs-wordpress

 We launch Wordpress which can join with MySQL database and use deployment kind and version of wordpress is 4.8 code, rum " kubectl create -f wordpress.yml " command in cmd

Now we create wordpress service

apiVersion: v1

kind: Service

metadata:

  name: wordpress

  labels:

    app: wordpress

spec:

  ports:

    - port: 80

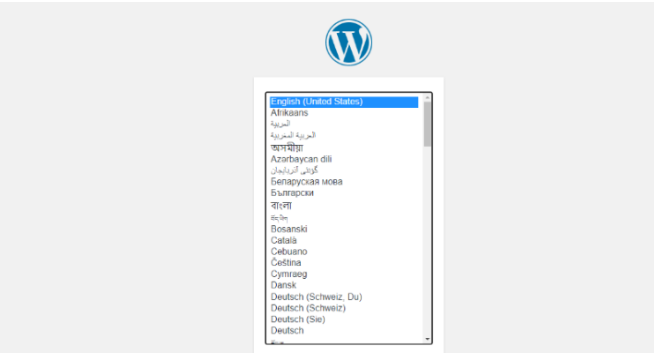
  selector:

    app: wordpress

    tier: frontend

  type: LoadBalancer

now we can see the Wordpress site First page



Now the WORDPRESS is created and running now we are gonna use helm to launch Prometheus and Grafana so first we need to setup helm for that we need the two software one helm and one triller just copy them somewhere and set PATH in environment variables and then go to command line and run

kubectl -n kube-system create serviceaccount tiller

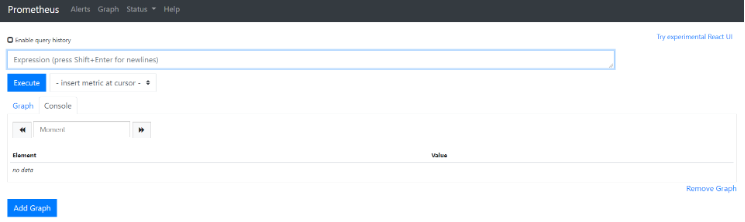
kubectl create cluster rolebinding tiller --clusterrole cluster-admin  --serviceaccount=kube-system:tiller

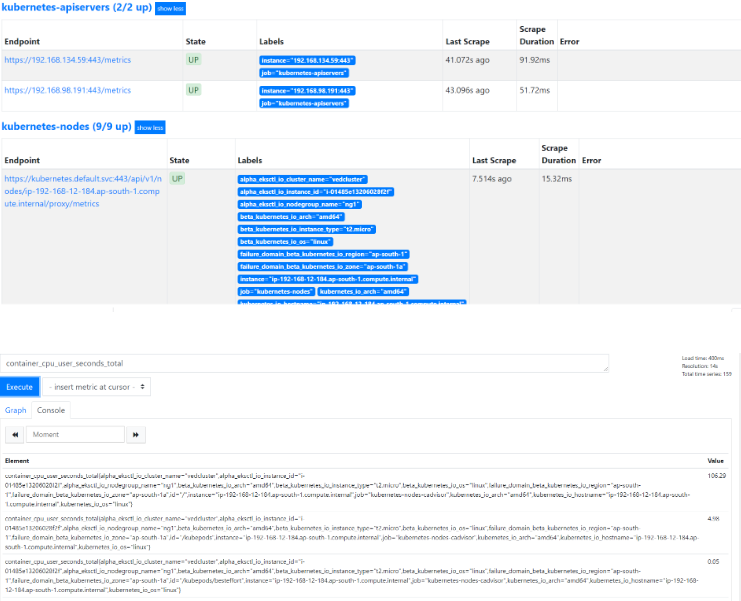
helm init --service-account tiller

helm init --service-account tiller --upgrade

These commands will configure the helm now we can use help to do our work fast . Now to install Prometheus in its namespace

helm install  stable/prometheus  --namespace prometheus  --set alertmanager.persistentVolume.storageClass="gp2"  --set server.persistentVolume.storageClass="gp2"





To use port forwarding run following commands:

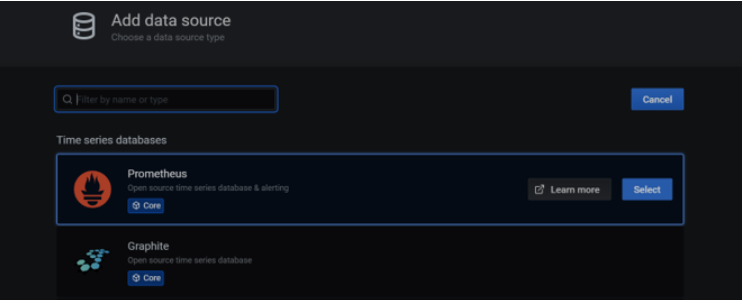
kubectl get svc -n prometheus

kubectl -n prometheus port-forward svc/dull-bumblebee-prometheus-server  88

To install grafana in its namespace:

helm install grafana/stable  --namespace grafana  --set persistence.storageClassName="gp2"  --set adminPasswod=redhat  --set service.type=LooadBalancer

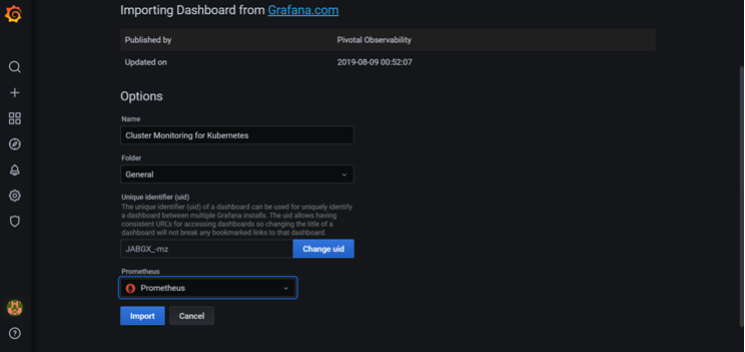




To use port forwarding in grafana:

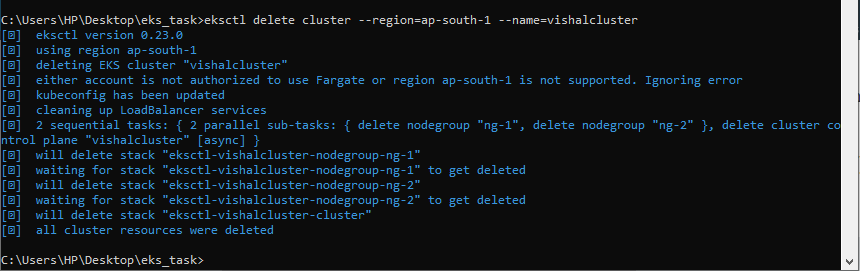
kubectl get svc -n grafana

kubectl -n grafana  port-forward svc/exasperated-seal-grafana  1234:80



this is the prove that we can launch our Prometheus with Grafana. Now everything is at its place now we can use this kind of setup to do some great things.

At last we must delete our cluster, because EKS is not a free service provided by AWS



GITHUB URL : <https://github.com/vishaldhole173/aws_eks>

THANK YOU !!!