**Model Deployment & Orchestration**

**Objective**

Deploy a machine learning model and orchestrate its operations using Kubernetes.

**Amazon Elastic Kubernetes Service(EKS)**

For the M4 task, we decided to use Amazon Elastic Kubernetes Service(EKS) over AWS or Google Cloud Platform for below reasons.

* **Integration**: EKS integrates seamlessly with AWS services, making it a good choice for businesses that already use AWS
* **Security**: EKS benefits from AWS's security features, including IAM, VPC, and Amazon GuardDuty
* **Scalability**: EKS is designed to scale efficiently, making it suitable for small and large applications
* **Customization**: EKS offers a wide range of customization options
* **Community support**: EKS has a large community and extensive third-party tool support

**EKS Initial Setup**

Setting up Amazon Elastic Kubernetes Service (EKS) begins with creating an EKS cluster, which acts as the control plane for Kubernetes workloads. It requires an AWS account, configure AWS CLI, and install tools like kubectl and eksctl to interact with EKS. Using eksctl, we can easily create the cluster by defining the desired configurations, such as the number of nodes and the region. After the cluster is created, you will configure kubectl to communicate with the EKS cluster using a kubeconfig file. Finally, you can deploy applications to the cluster and manage workloads by utilizing Kubernetes resources such as pods, services, and deployments.

**Kubernetes Engine**

* **Cluster Creation**

Cluster creation in EKS involves using tools like eksctl or AWS Management Console to define the desired configuration, such as the VPC, node types, and scaling options. After provisioning, EKS sets up the control plane, and you can connect kubectl to manage the cluster for deploying and scaling applications.

**A black text on a white background

Description automatically generated**

This command creates an eks cluster below

**A screenshot of a computer

Description automatically generated**

* **Deployment (using Deployment.yaml)**

In EKS, a deployment YAML file defines the desired state for a Kubernetes application, including the number of replicas, container images, and ports. This file is applied using kubectl apply -f to create and manage the deployment in the EKS cluster**.**

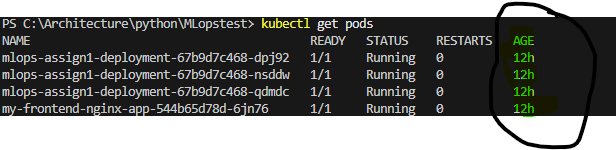
**A screen shot of a computer

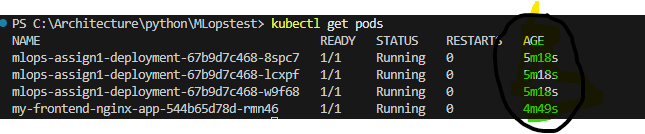
Description automatically generated**

**A screen shot of a computer

Description automatically generated**

On each CD trigger the changed/updated application will be deployed into EKS





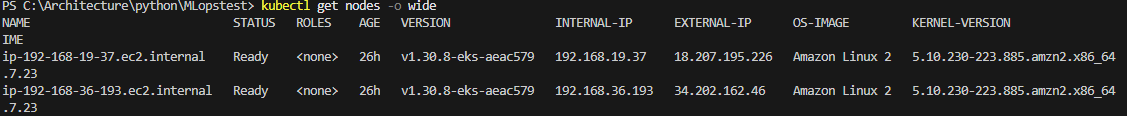
* **Testing the deployed app**

Once the deployment is completed the applications can be see with kubectl commands kubectl get pods

A screenshot of a computer screen

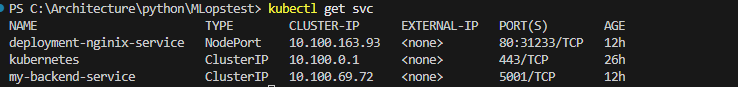
Description automatically generated

These podes are running under nodes



* **Link to the deployed endpoint**

The application is accessed using a service resource in the EKS which exposed via a reverse proxy(nginx)in this case.



The end point URL : http://18.207.195.226:31233/mlops/predict\_diabetes

