In this tutorial, you do the following:

Create an Kubernetes Cluster on EKS using the **eksctl** tool from AWS and make a sample application deployment using **kubectl** and expose it through a load balancer.

[**What is EKS? 1**](#_jpta0l7nhq8m)

[**Pre Requisites 2**](#_d3wa77js39j4)

[**Getting started with EKS 2**](#_za05ev0b56q)

[**Basic Kubernetes commands 4**](#_eorglo6ch2v4)

[**Deploy sample application 5**](#_638gl153gl6p)

### What is EKS?

Amazon Elastic Kubernetes Service (Amazon EKS) is a managed service that you can use to run Kubernetes on AWS without needing to install, operate, and maintain your own Kubernetes control plane or nodes. Kubernetes is an open-source system for automating the deployment, scaling, and management of containerized applications. Amazon EKS:

* Runs and scales the Kubernetes control plane across multiple AWS Availability Zones to ensure high availability.
* Automatically scales control plane instances based on load, detects and replaces unhealthy control plane instances, and it provides automated version updates and patching for them.
* Is integrated with many AWS services to provide scalability and security for your applications, including the following capabilities:
  + Amazon ECR for container images
  + Elastic Load Balancing for load distribution
  + IAM for authentication
  + Amazon VPC for isolation
* Runs up-to-date versions of the open-source Kubernetes software, so you can use all of the existing plugins and tooling from the Kubernetes community. Applications that are running on Amazon EKS are fully compatible with applications running on any standard Kubernetes environment, no matter whether they're running in on-premises data centers or public clouds. This means that you can easily migrate any standard Kubernetes application to Amazon EKS without any code modification.

### Pre Requisites

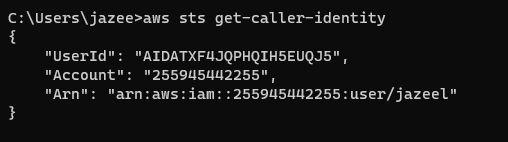
* Install kubectl on your machine:
  + Mac: brew install kubectl
  + Windows: choco install kubernetes-cli
  + After installing, run **kubectl version --client** to verify installation
* Install eksctl:
  + Mac: <https://github.com/weaveworks/eksctl/blob/main/README.md#for-macos>
  + Windows(Run powershell with administrator first): <https://github.com/weaveworks/eksctl/blob/main/README.md#for-windows-1>
  + After installing, run **eksctl version** on your terminal/powershell to verify installation

### Getting started with EKS

We would be following the instructions here to create a EKS cluster using Managed EC2 nodes:

Open your terminal / powershell and run the following command. This command will show you which account / user you’re currently in:

**aws sts get-caller-identity**

****

Create a new repo in Github and clone it to your desktop

Open terminal/powershell, cd into your repo and type code . to open in VSCode

Then create a file named cluster.yaml with the following content in it:

apiVersion: eksctl.io/v1alpha5

kind: ClusterConfig

metadata:

name: <SET YOUR OWN CLUSTER NAME HERE e.g. jazeel-eks-cluster>

region: us-east-1

vpc:

subnets:

public:

us-east-1a: { id: subnet-0860c289a56da0f08 }

us-east-1b: { id: subnet-0b442c55bfcaef886 }

us-east-1c: { id: subnet-0c236aa7db587a815 }

nodeGroups:

- name: <SET YOUR WORKER NODE GROUP NAME HERE. e.g. jazeel-worker-ng>

labels: { role: workers }

instanceType: t3.small

desiredCapacity: 1

Then go to your terminal and run the following command:

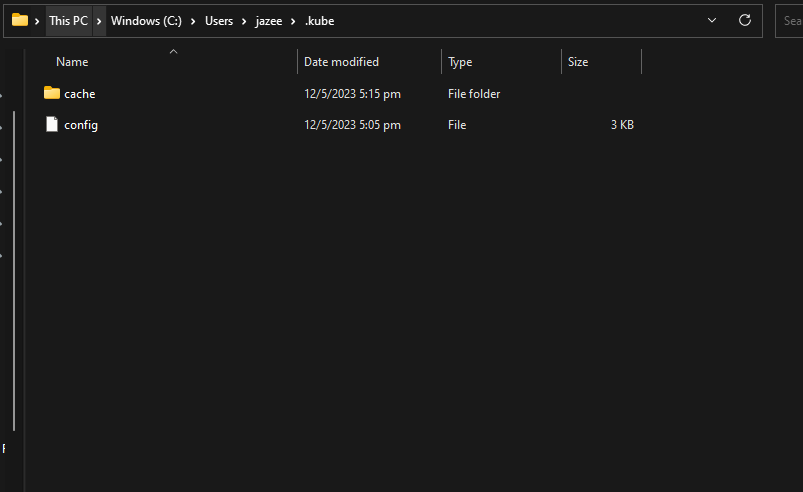
**eksctl create cluster -f cluster.yaml**

After you have finished creating you can go to your root folder to view that a kubeconfig file has been created. This is your “credential” file to connect to the kubernetes cluster.

Mac Directory: ~/.kube

Windows: C:/Users/<your user>/.kube

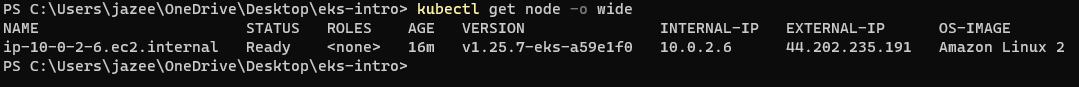
Windows example shown below:



### Basic Kubernetes commands

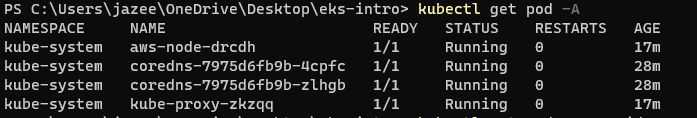
Once the cluster stack has finished creating in the terminal, you would need to verify that the worker node that you created above as part of cluster.yaml, is part of the cluster. To verify this, run the following command:

**kubectl get node -o wide**

****

You can also run the following command to view pods created across all namespaces. Since for now we have not deployed anything, you would be only able to see the control plane nodes:

**kubectl get pod -A**



You can also use the following command to find out more information about your worker node:

**kubectl describe nodes**

**kubectl describe nodes <name of node>**

### Deploy sample application

Create a file called **k8s-deployment.yaml** with the content below:

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: hello-kubernetes

spec:

replicas: 3

selector:

matchLabels:

app: hello-kubernetes

template:

metadata:

labels:

app: hello-kubernetes

spec:

containers:

- name: hello-kubernetes

image: paulbouwer/hello-kubernetes:1.5

ports:

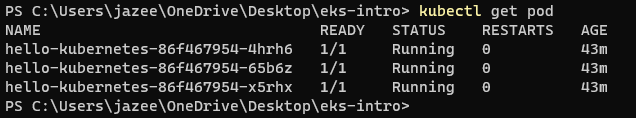
- containerPort: 8080

The code above would be used to create 3 pods(with 1 container each) running the paulbouwer/hello-kubernetes:1.5 image on port 8080

Run the following command to deploy the above manifest file:

**kubectl apply -f k8s-deployment.yaml**

Once deployed, you can also view your deployed pods::



### Expose your application publicly

Create a file called **service.yaml with the following content and run kubectl apply -f service.yaml**

**---**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: hello-kubernetes**

**spec:**

**type: LoadBalancer**

**ports:**

**- port: 80**

**targetPort: 8080**

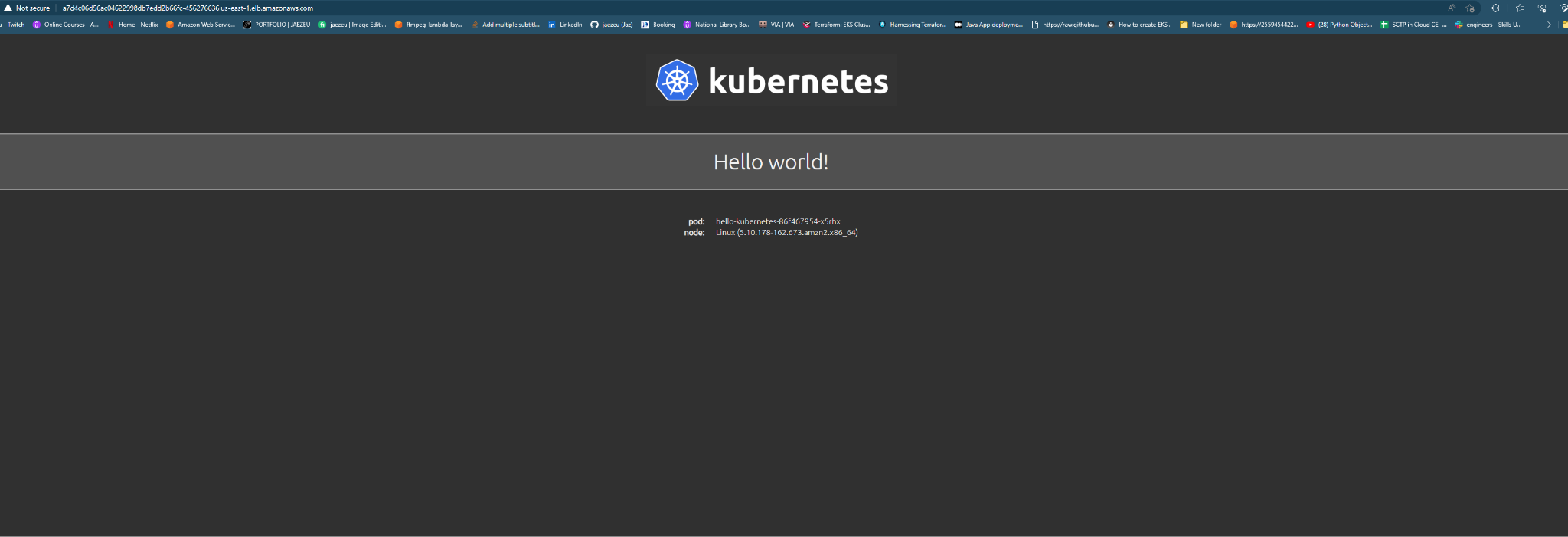
**selector:**

**app: hello-kubernetes**

**Then you can run kubectl get svc, you will be able to see a external DNS name, which is also the load balancer dns**

****

**Copy the DNS and paste into your browser to view your application live:**

****