**Docker:**

Docker is an open source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.

Installation steps for docker:

1.Download Docker:

<https://docs.docker.com/desktop/windows/install/>

2.Double –click Install Docker.

3.Follow the install wizard: accept the license, authorize the installer, and proceed with the install

4.Click finish to launch Docker.

5.Docker starts automatically.

6.Docker loads a“Welcome“windows giving you tips and access to theDocker documentation.

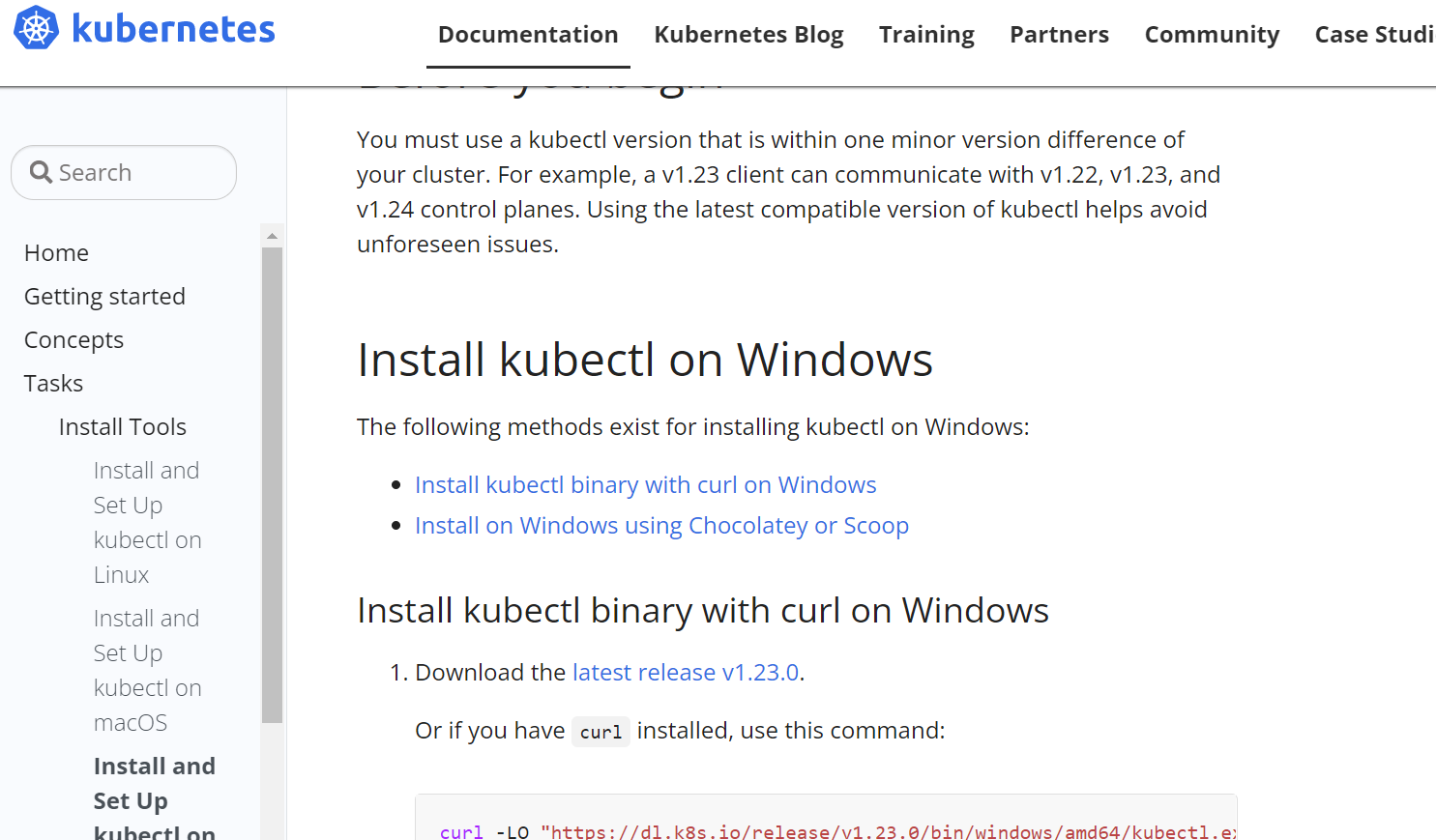
**Kubernetes:**

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation

1.To Install Kubernetes on windows go to this website:

<https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/>

2.Then click on latest version v1.23.0.to download the Kubernetes.



3.Add the path in Environment variables.

Graphical user interface, text, application

Description automatically generated

4.Test to ensure the version of kubectl is the same as downloaded.

$kubectl version –client

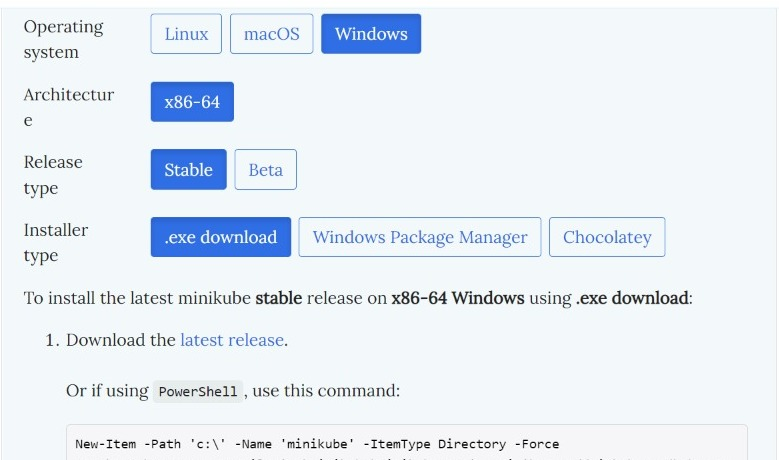
**MINIKUBE:**

Like kind ,minikube is a tool that lets you run Kubernetes locally. minikube runs a single-node Kubernetes cluster on your personal computer (including Windows, macOS and Linux PCs) so that you can try out Kubernetes, or for daily development work.

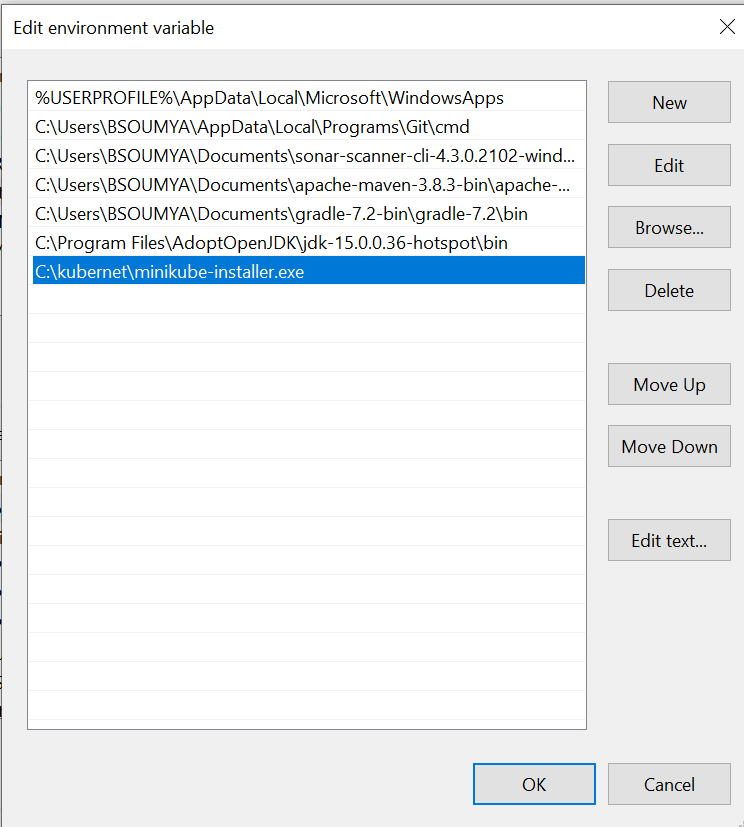
1.To Install minikube on windows go to this website:. To Install minikube on windows go to this website:

[**https://minikube.sigs.k8s.io/docs/start/**](https://minikube.sigs.k8s.io/docs/start/)

2.Then click on latest release and download.



3.Add the path in Environment variables.



4.Start your cluster by using below command.

$ minikube start

5.Then it will be shown below.

**Text

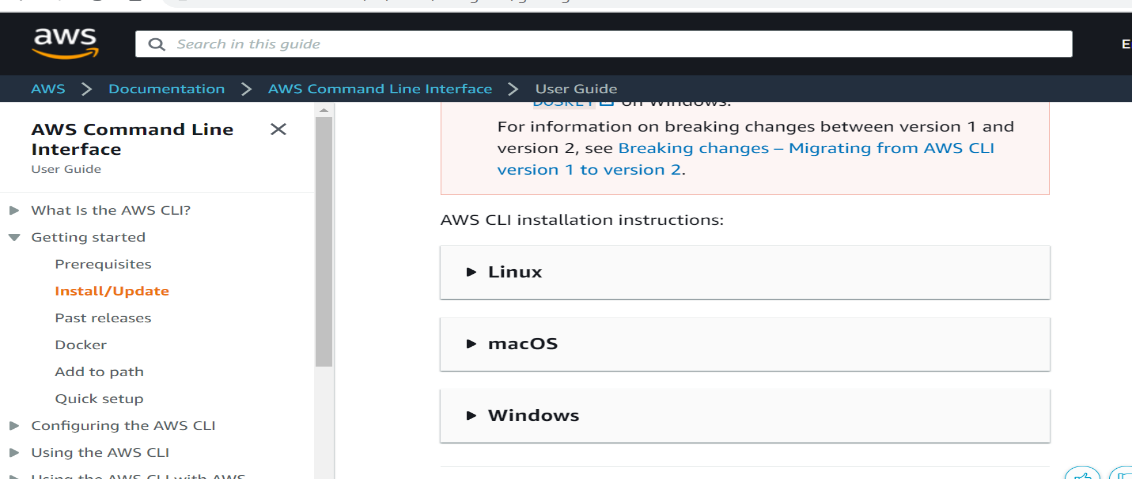
Description automatically generated**

AWSCLI:

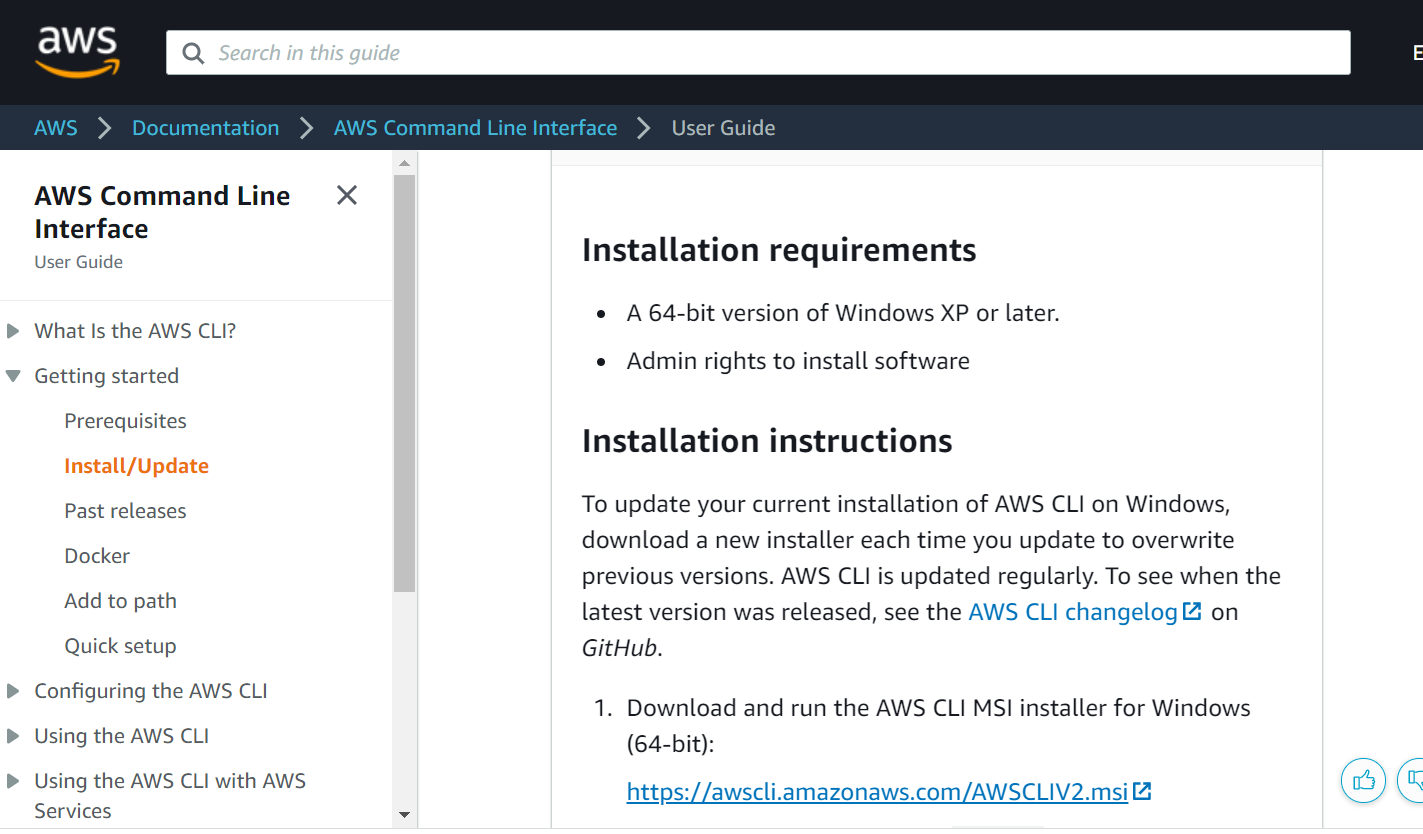
1. To Install AWSCLI on windows go to this website:

<https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting->

2.Then goto install and click on windows.



3.click on link to download the awscli



4.Follow the install wizard : accept the license, authorize the installer, andproceed with the install

5..Click finish to launch AWSCLI.

6.To confirm the installation, open the Start menu, search for cmd to open a command prompt window, and at the command prompt use the aws --version command.

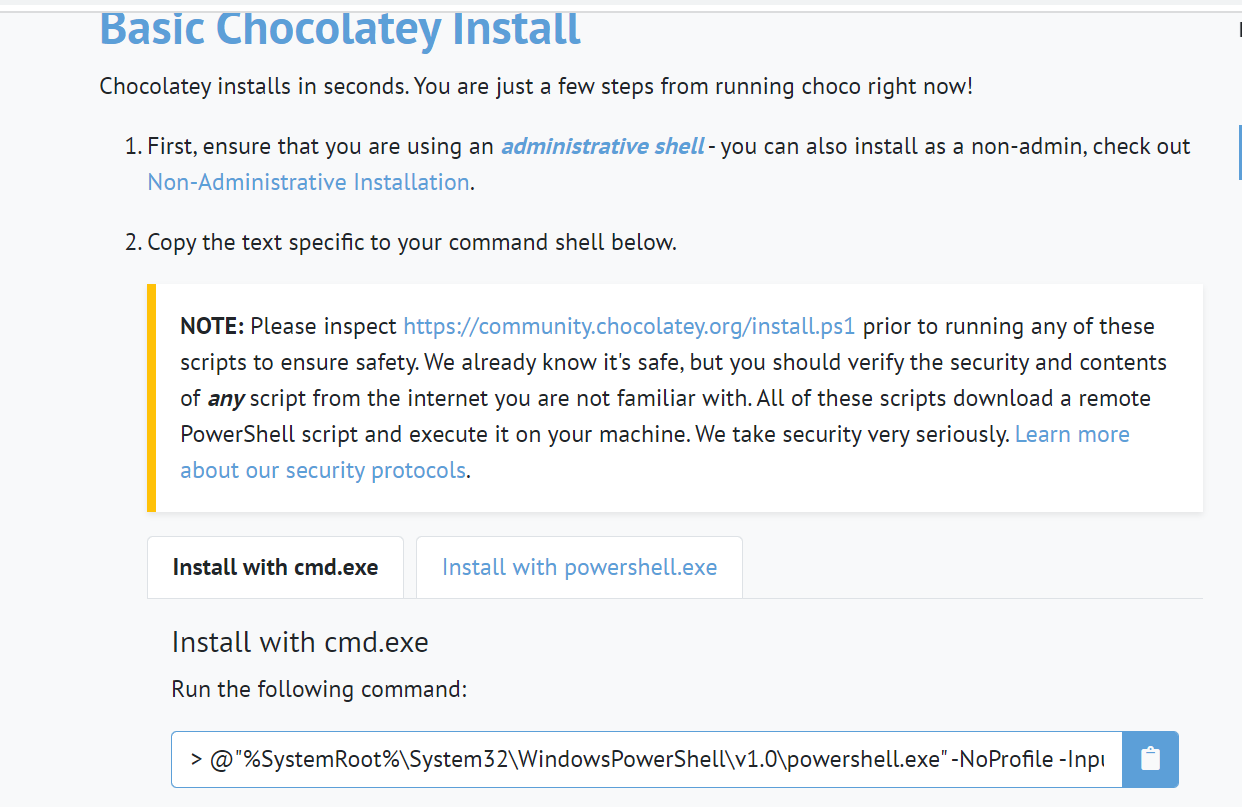
**CHOCOLATEY:**

Chocolatey is a software management solution that gives you the freedom to create a simple software package and then deploy it anywhere you have Windows using any of your familiar configuration or system management tools

1.Goto chocolatey website:

https://chocolatey.org/install

2.click on take the installation course.



3.Paste the copied text into your shell and press Enter.

4.Wait a few seconds for the command to complete.

5.Then upgrade by using the below command:

$ choco upgrade chocolatey

**EKSCTL:**

Eksctl is a tool jointly developed by AWS and Weaveworks that automates much of the experience of creating EKS clusters. In this module, we will use eksctl to launch and configure our EKS cluster and nodes

1.To install eksctlgoto these AWS website and open Amazon EMR on EKS Development Guide.

2.If you do not already have Chocolatey installed on your Windows system, see [Installing Chocolatey](https://chocolatey.org/install)

3.Install or upgrade eksctl

$ choco install -y eksctl

4.If they are already installed, run the following command to upgrade:

$ choco upgrade -y eksctl

5.Test that your installation was successful with the following command.

$ eksctl version

**Document - Employee Management System**

Procedure:

1. Steps To create Docker file:
   1. Install Docker on machine.
   2. Create project.
   3. Create a file called Docker File.
   4. Build your Docker File using properties.
   5. Save the file.
2. Steps to create docker image:
   1. Create a Base Container
   2. Inspect Images
   3. Inspect Containers
   4. Start the Container
   5. Modify the Running Container
   6. Create an Image from a Container by using below command.

* **docker build -t employee-management-system .**
  1. Tag the Image
* **docker tag employee-management-system nandini15/employee-management-system:0.0.1**
  1. push that image into docker hub using below command.
* **docker push nandini15/employee-management-system:0.0.1**

**Graphical user interface, text, application, email

Description automatically generated**

1. Steps to create docker-compose file:
2. create the docker-compose.yml
3. Define services in a Compose file
4. Run the application with compose using the below command

* **docker-compose up**

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

Check whether the application is working with the port in chrome.

Graphical user interface, text, application

Description automatically generated

1. Create the manifest files(yaml):
2. **change the application properties**
3. **write yaml files**

* **deployment.yaml**
* **postgres-credentials.yaml**
* **postgres-configmap.yaml**
* **postgres-deployment.yaml**

1. **Again build the jar**
2. **In cmd start the minikube**
3. Deploy the application on Kubernetes environment:

Create an image employee-management-postgres and push the image.

* **docker build -t employee-management-postgres.**
* **docker tag employee-management-system nandini15/employee-management-postgres**
* **docker push nandini15/employee-management-postgres**

Change the directory using the command **cd k8s** ( In folder k8s yaml files are created)

1. write the below deployment commands in cmd:

* **kubectl create -f deployment.yaml**
* **kubectl create -f postgres-credentials.yaml**
* **kubectl create -f postgres-configmap.yaml**
* **kubectl create -f postgres-deployment.yaml**

**Graphical user interface, text, application, email

Description automatically generated**

1. After succesful deployment ,forward the port

* **kubectl port-forward svc/employee-management-system 9094:8080**

**Text

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

1. Deploy the application on EKS cluster:
   1. Create a cluster in EKS with eksctl command

* **eksctl create cluster --name employee-management-system --version 1.21 --region ap-south-1 --nodegroup-name employee- management-system -node-group --node-type t2.micro --nodes 2**
* **awseks --region ap-south-1 update-kubeconfig --name employee-management-system**

**Graphical user interface, text

Description automatically generated**

**Text

Description automatically generated**

* 1. Create the docker image and push on docker hub
  2. Deploying the application on eks cluster using the following commands

Change the directory using the command **cd test** ( In folder test yaml files are created)

* **kubectl apply -f postgres-storage.yml**
* **kubectl apply -f postgres-secrets.yml**
* **kubectl apply -f postgres-deployment.yml**
* **kubectl apply -f postgres-service.yml**

Text

Description automatically generated

* **kubectl get all**
  1. Set the config map
  2. Get the Postgres Host IP Address:
* **kubectl get svc postgres -o jsonpath="{.spec.clusterIP}"**
  1. get the IP Address and put in the below command
* **kubectl create configmap hostname-config –from literal=postgres\_host=10.100.7.34**
* **kubectl apply -f springboot-deployment.yml**
* **kubectl apply -f springboot-service.yml**

**EKS Cluster Link:-**

<http://add70ac801ba243aeb316d625f7c4af3-1949687877.ap-south-1.elb.amazonaws.com:8080/employee/getAll>

Graphical user interface, text, application

Description automatically generated