Minikube is a tool that makes it easy to run a single-node Kubernetes cluster on your local machine (Linux, macOS, or Windows). It's designed for:

Uses:

* Learning Kubernetes: Ideal for beginners to get hands-on experience with Kubernetes without the complexity of setting up a full-fledged cluster.
* Development: Allows developers to test their applications in a Kubernetes environment before deploying to production.
* Testing: Provides a convenient way to run automated tests against Kubernetes configurations.

Functionality:

* Cluster Creation: Sets up a minimal Kubernetes cluster on your machine within a virtual machine or container.
* Kubernetes Features: Supports many core Kubernetes features, like deploying applications, managing services, and using Persistent Volumes.
* Addons: Offers easy installation of additional Kubernetes tools and services (e.g., dashboards, ingress controllers).
* Multiple Container Runtimes: Can use various container runtimes like Docker, Containers, or CRI-O.

Key Points:

* Lightweight: Designed for local use, not for production workloads.
* Easy to Use: Simple commands for starting, stopping, and managing the cluster.
* Community-driven: Actively developed and supported by the Kubernetes community.

How it Works:

Minikube creates a virtual machine (or uses a container runtime) on your machine. Within this environment, it installs and configures all the necessary Kubernetes components (like the API server, controller manager, and kubelet) to create a functional cluster.

How to set up?

What you’ll need

* 2 CPUs or more
* 2GB of free memory
* 20GB of free disk space
* Internet connection
* Container or virtual machine manager, such as Docker, QEMU, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMware Fusion/Workstation

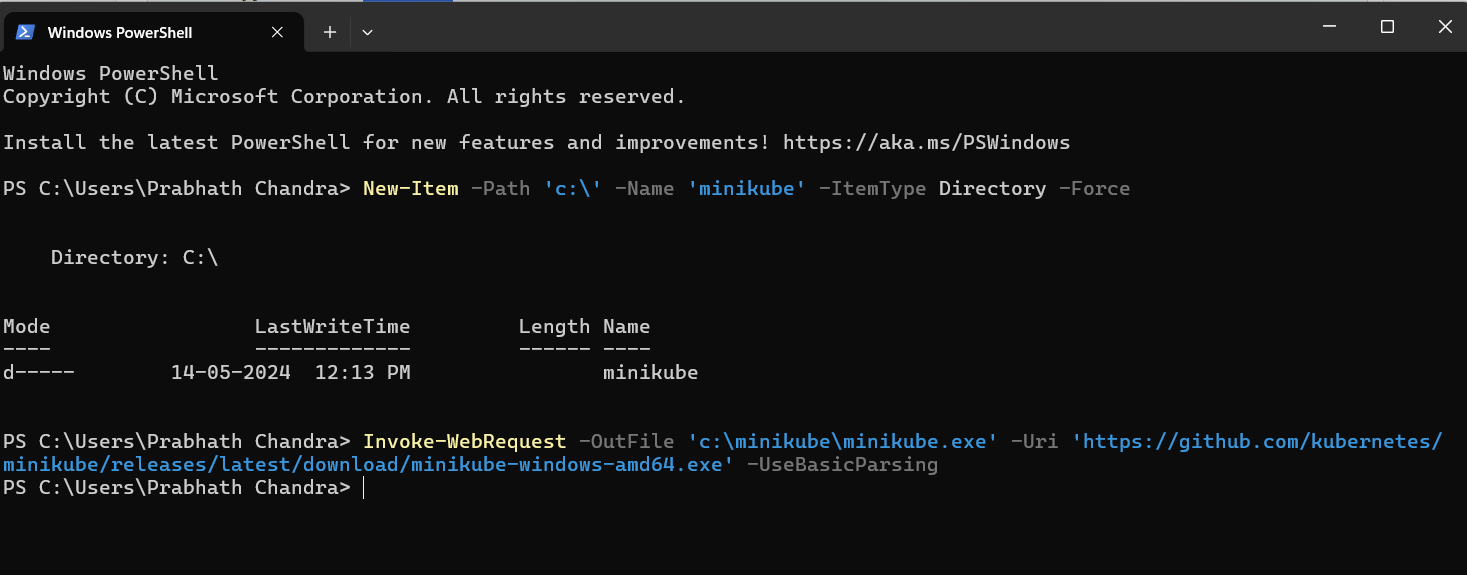
1) **Installation:**

I. **D**ownload and run the installer for the latest release.

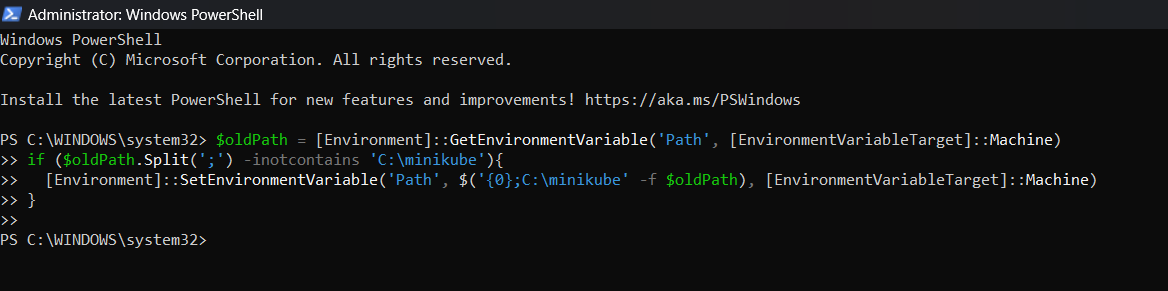
Or if using PowerShell, use this command:

“New-Item -Path 'c:\' -Name 'minikube' -ItemType Directory -Force

Invoke-WebRequest -OutFile 'c:\minikube\minikube.exe' -Uri 'https://github.com/kubernetes/minikube/releases/latest/download/minikube-windows-amd64.exe' -UseBasicParsing”



II. Add the minikube.exe binary to your PATH. Make sure to run PowerShell as Administrator.



2) **Start your cluster:**

From a terminal with administrator access (but not logged in as root), run:

“minikube start”

This will create a single-node Kubernetes cluster on your local machine.

**3) Creating a Pod:**

**Pods:**

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

A Pod (as in a pod of whales or pea pod) is a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers. A Pod's contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific "logical host": it contains one or more application containers that are relatively tightly coupled. In non-cloud contexts, applications executed on the same physical or virtual machine are analogous to cloud applications executed on the same logical host.

1. Create a YAML File (e.g; simple-pod.yaml)

“apiVersion: v1

kind: Pod

metadata:

name: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80”

1. Deploy the Pod:

“kubectl apply -f simple-pod.yaml”

1. Verify Pod Creation:

“kubectl get pods”

You should see your pod listed with a status of "Running" or "Pending."

**4) Kubectl Commands:**

Basic Commands:

* kubectl get nodes: List nodes in the cluster.
* kubectl get pods: List pods.
* kubectl describe pod <pod-name>: Show details about a pod.
* kubectl logs <pod-name>: View logs of a pod.
* kubectl exec -it <pod-name> -- bash: Execute a shell in a pod.
* kubectl delete pod <pod-name>: Delete a pod.

Deployment Commands:

* kubectl create deployment <deployment-name> --image=<image>: Create a deployment.
* kubectl get deployments: List deployments.
* kubectl rollout status deployment/<deployment-name>: Check deployment status.
* kubectl scale deployment/<deployment-name> --replicas=<num>: Scale a deployment.
* kubectl delete deployment <deployment-name>: Delete a deployment.

Service Commands:

* kubectl expose deployment <deployment-name> --type=NodePort --port=<port>: Expose a deployment as a service.
* kubectl get services: List services.
* kubectl describe service <service-name>: Show details about a service.
* kubectl delete service <service-name>: Delete a service.

Other Commands:

* kubectl get all: List all resources (pods, deployments, services, etc.).
* kubectl cluster-info: Display cluster information.
* kubectl version: Display client and server versions.