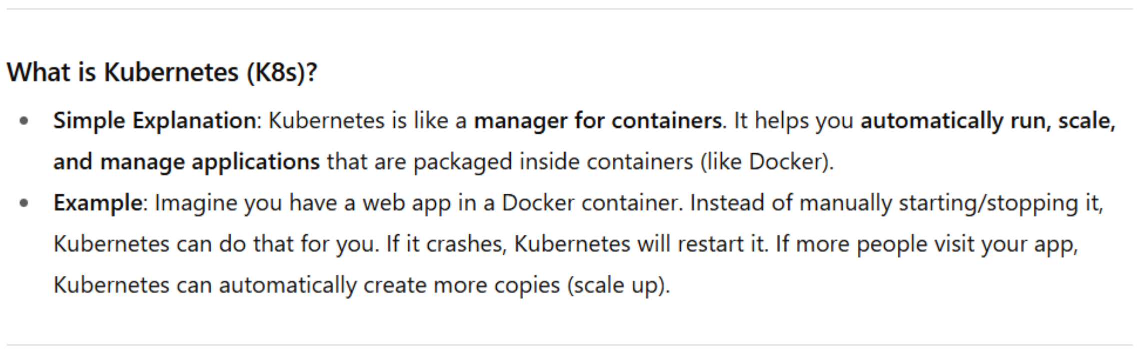
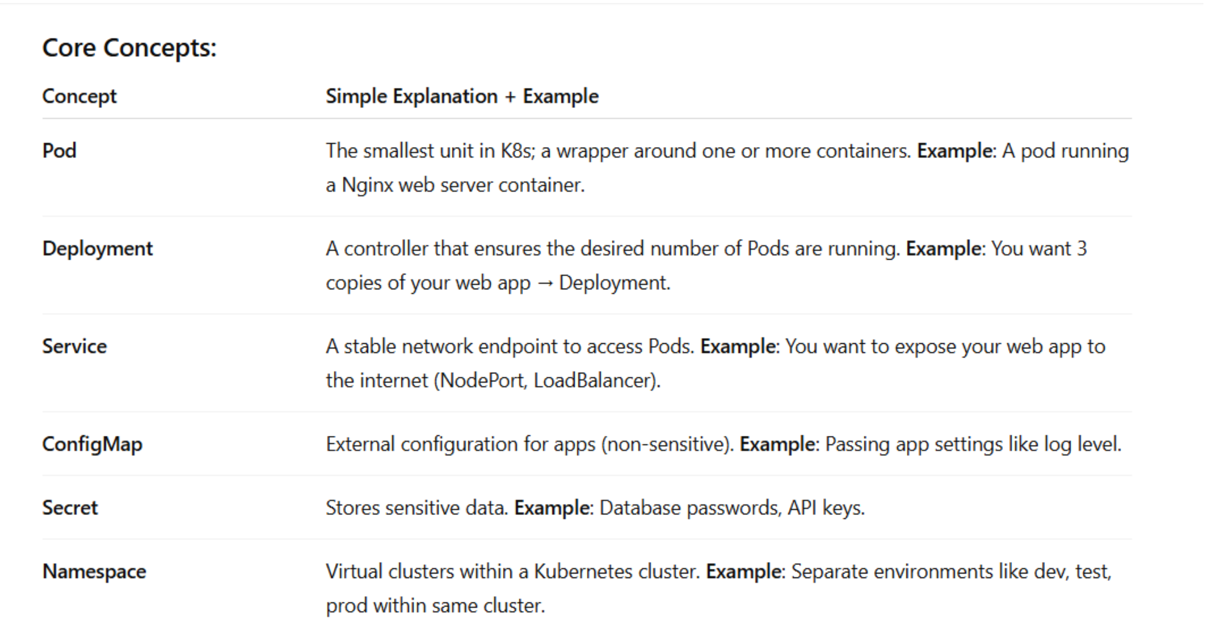
What is Kubernetes (K8s)?



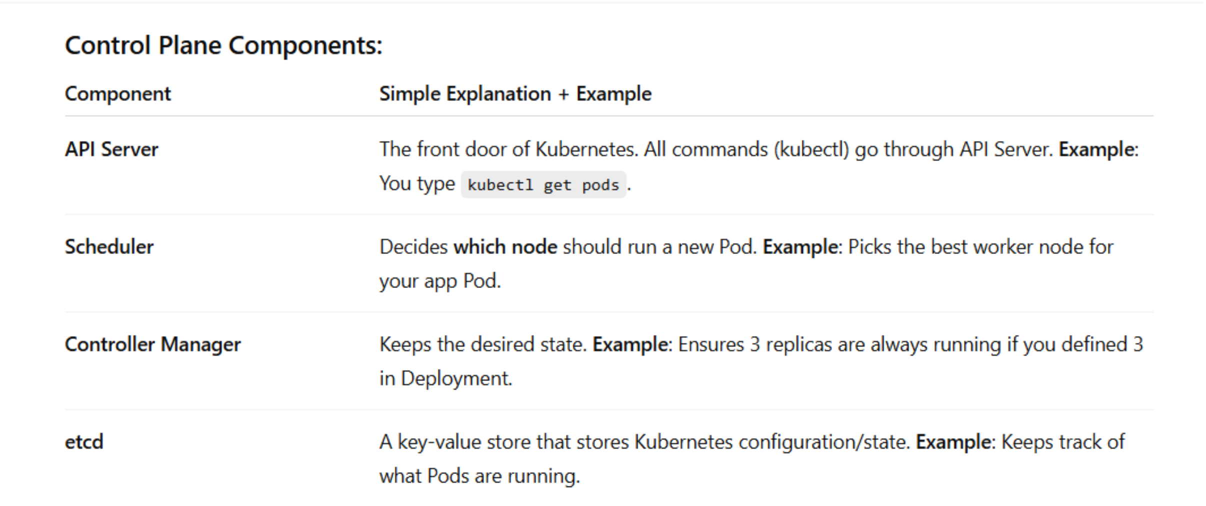
What is Kubernetes (K8s)?

- **Simple Explanation**: Kubernetes is like a **manager for containers**. It helps you **automatically run, scale, and manage applications** that are packaged inside containers (like Docker).
- **Example**: Imagine you have a web app in a Docker container. Instead of manually starting/stopping it, Kubernetes can do that for you. If it crashes, Kubernetes will restart it. If more people visit your app, Kubernetes can automatically create more copies (scale up).



Core Concepts:

Concept	Simple Explanation + Example
Pod	The smallest unit in K8s; a wrapper around one or more containers. **Example**: A pod running a Nginx web server container.
Deployment	A controller that ensures the desired number of Pods are running. **Example**: You want 3 copies of your web app → Deployment.
Service	A stable network endpoint to access Pods. **Example**: You want to expose your web app to the internet (NodePort, LoadBalancer).
ConfigMap	External configuration for apps (non-sensitive). **Example**: Passing app settings like log level.
Secret	Stores sensitive data. **Example**: Database passwords, API keys.
Namespace	Virtual clusters within a Kubernetes cluster. **Example**: Separate environments like dev, test, prod within same cluster.



Control Plane Components:

Component	Simple Explanation + Example
API Server	The front door of Kubernetes. All commands (kubectl) go through API Server. **Example**: You type `kubectl get pods`.
Scheduler	Decides **which node** should run a new Pod. **Example**: Picks the best worker node for your app Pod.
Controller Manager	Keeps the desired state. **Example**: Ensures 3 replicas are always running if you defined 3 in Deployment.
etcd	A key-value store that stores Kubernetes configuration/state. **Example**: Keeps track of what Pods are running.

Worker Node Components:

Component	Simple Explanation + Example
Kubelet	Agent that runs on each worker node. Talks to API Server and manages Pods on that node.
Kube-proxy	Manages networking rules so Pods can talk to each other.
Container Runtime	The actual software that runs containers (Docker, containerd). **Example**: Docker runs your app container.

Networking Basics:

Concept	Simple Explanation + Example
Pod Networking (CNI)	Ensures every Pod gets an IP and can talk to others. Example: Two Pods communicate over their IP addresses.
Service Discovery (DNS)	Resolves Service names to Pod IPs. Example: App connects to `db-service` instead of an IP.
Ingress Controller	Manages external access (HTTP/HTTPS) to Services. Example: Nginx Ingress routes www.example.com to your app.

Persistent Storage:

Concept	Simple Explanation + Example
Volume	Storage attached to a Pod. Example: A folder that stores uploaded files.
PersistentVolume (PV)	A piece of storage in the cluster (like EBS in AWS).
PersistentVolumeClaim (PVC)	A request for storage by a Pod. Example: Your app Pod asks for 5GB of storage through a PVC.
StorageClass	Defines types of storage (fast SSDs, normal HDDs). Example: Choose StorageClass for performance needs.

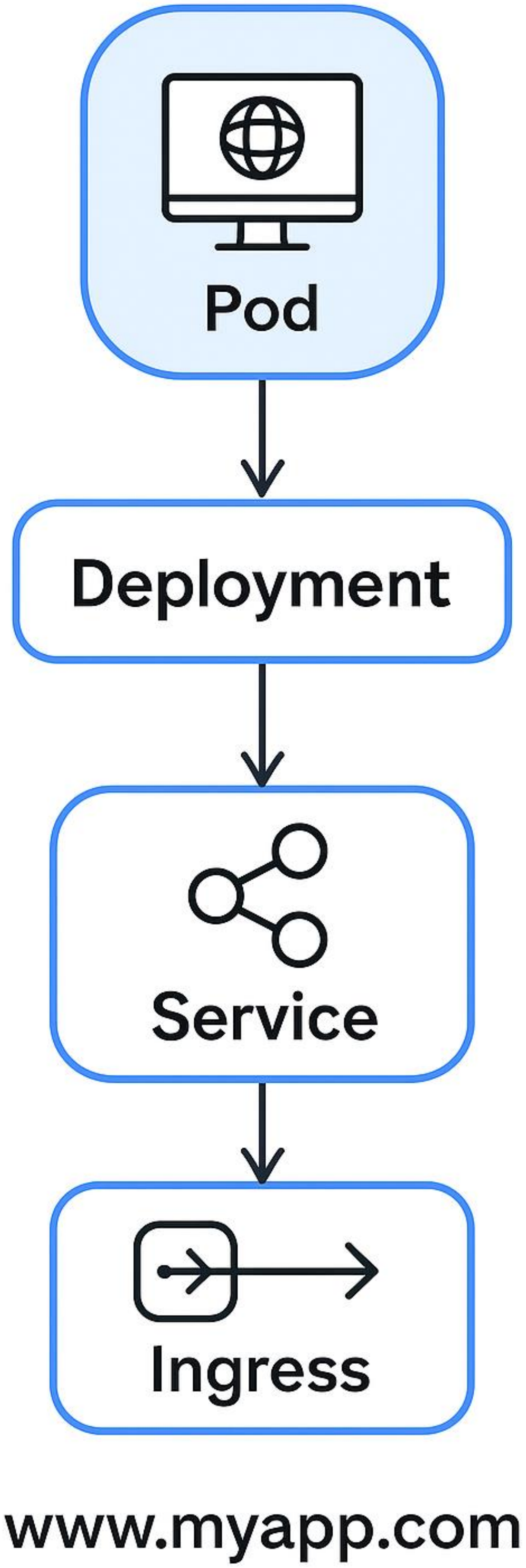
RBAC (Role-Based Access Control):

Concept	Simple Explanation + Example
User	The person or system accessing Kubernetes.
Role	Set of permissions within a namespace. Example: Role to allow viewing Pods.
ClusterRole	Permissions that span the entire cluster. Example: Admin rights across all namespaces.
RoleBinding	Assigns a Role to a User in a namespace. Example: Give developer access to only the "dev" namespace.

Visual Example Scenario:

You deploy a **web application** on Kubernetes:

1. You create a **Deployment** with 3 replicas of your app container.
2. You expose it using a **Service (LoadBalancer)**.
3. You use a **ConfigMap** for app configurations.
4. Store secrets like DB passwords in a **Secret**.
5. Users access your app via an **Ingress** at `www.myapp.com`.
6. Logs are stored in a **PersistentVolume**.
7. You use **RBAC** to give developers access only to "dev" namespace.



Pod

Deployment

Service

Ingress

www.myapp.com

What is a Pod?

- A **Pod** is like a **box** that runs your application.
- It can contain **one or more containers** (usually one).
- Example: You have a website running in a **Nginx container** → Kubernetes puts it inside a **Pod**.

What is a Service?

- A **Service** is like a **permanent address (IP & Port)** for your Pods.
- Pods can **come and go** (they die and restart), but the Service IP stays the same.
- The Service makes sure traffic always reaches the correct Pod(s), even if Pods change.

Types of Services:

Type	Simple Example
ClusterIP	Access the app **inside the cluster only**. Example: Backend service used by other apps in cluster.
NodePort	Opens a port on **all nodes** so you can access the app from outside. Example: Testing app on port 30080.
LoadBalancer	Gets a **public IP** (in cloud) to expose app to the internet. Example: www.myapp.com.

Example Scenario:

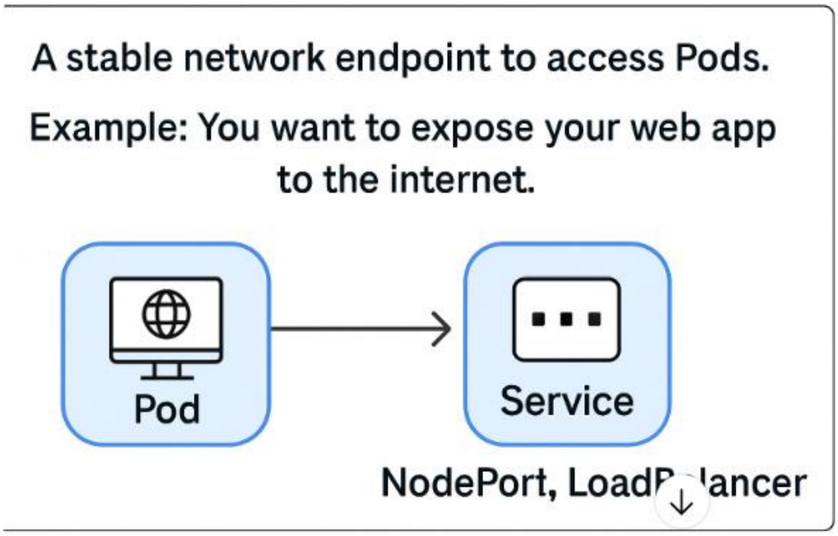
You have a **web app**:

1. You put it in a **Pod**.
2. You want users to access it from a browser.
3. You create a **Service (LoadBalancer)**.
4. Users visit `www.myapp.com` → Request goes to **Service IP** → Service routes it to the **Pod**.

I'll now generate a **diagram showing:**

- A Pod (with app)
- A Service (LoadBalancer)
- User accessing it via browser (www.myapp.com)

Generating Diagram...



A stable network endpoint to access Pods.

Example: You want to expose your web app to the internet.