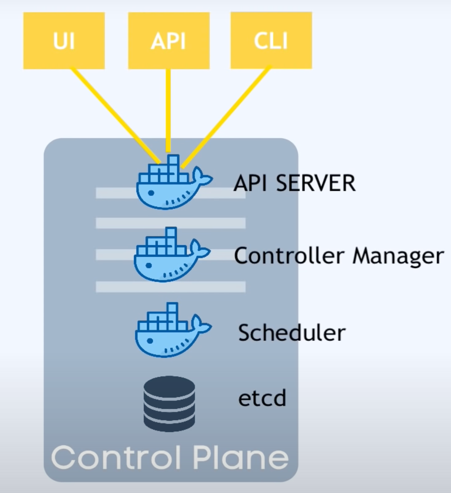
**Kubernetes**: Open-source container orchestration tool, developed by Google. Helps manage containerized applications in different deployment environments.

* Need for container orchestration tool
  + Trend from Monolith to Microservices
  + Increased usage of containers
  + Demand for a proper way of managing those hundreds of containers.
  + Difficult to manage all the containers of various environments using scripts
* Features provided by Kubernetes for contest orchestration:
  + ﻿﻿High Availability or no downtime
  + ﻿﻿Scalability or high performance
  + Disaster recovery - backup and restore

**Kubernetes Architecture:**

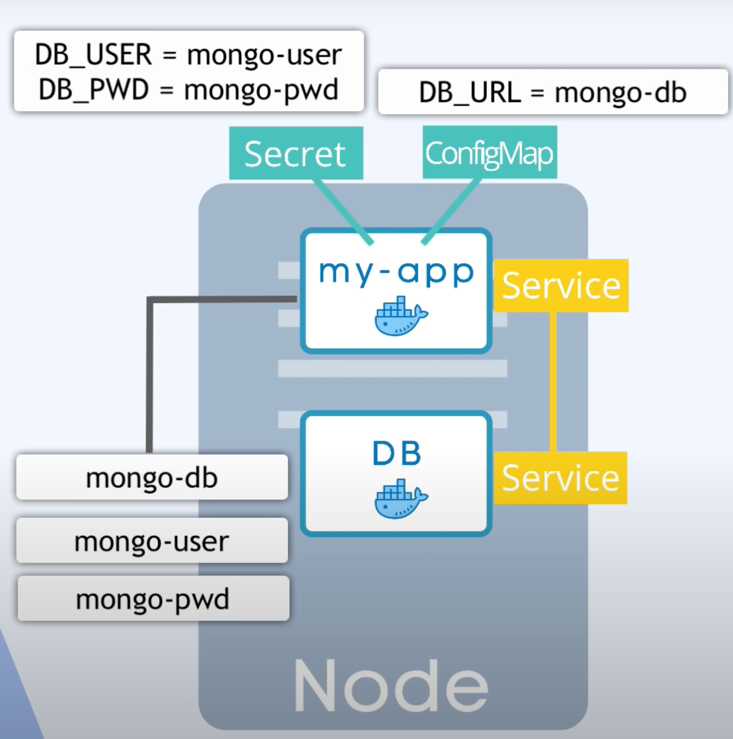
* Master Slave architecture
* Each node will have the **kubelet** which is primary agent responsible for communication across the nodes.
* Containers will run on the nodes

Master Node:

* API Server = Entrypoint to K8s cluster
* Controller Manager = keeps track of whats happening in the cluster
* Scheduler = ensures Pods placement
* etcd = Kubernetes backing store

**Main Kubernetes Components**

|  |  |
| --- | --- |
| Pod | ConfigMap |
| StatefulSet | DaemonSet |
| Service | Secret |
| Ingress | Deployment |

* **Pods**:
  + Smallest unit in Kubernetes
  + Abstraction over container, helps in isolating the container runtime and technology
  + Usually 1 Application per Pod, can also have dependent containers together
  + Each Pod gets its own IP address, these IP addresses are used for communication among the pods.
  + ﻿﻿New IP address on re-creation, this will make the dependent containers to map to new IP address for communication.
* **Service:**
  + Permanent IP address, overcomes the issues in pods
  + ﻿﻿Lifecycle of Pod and Service are independent
  + **External** Service: To access from web browser, but it’ll have the url as {nodenumber: port} without domain.
  + **Internal** Service: Allows components of cluster to access but not publicly exposed (Ex: Database).
* **Ingress**: It’ll provide the domain name to the IP addresses in Service
  + App requests are made to Ingress
  + Followed by services
* **ConfigMap**:
  + Whenever an environment variable is changed then we should update it, followed by build the image, push and re-build.
    - Example: Database url is changed, usually URL’s will be in the build application and must manually update the url.
  + Instead, a config map will maintain these environment variables and update them in the pod without rebuilding it.
* **Secret**:
  + Instead of Storing the data in plain text, they’re encrypted using third parties.
  + Used for storing credentials or other important data.
* **Vol:**
  + Data stored in pod’s cannot be persisted and will be lost upon restart / crashing.
  + Therefore, vol’s are used to store it in either:
    - Storage on local machine (in node)
    - Or remote, outside of the K8s cluster (in cloud)
* **Deployment**:
  + To create replicas of the application (pods), thus leverages distributed system and enables zero down time
  + Deployments are blueprint for Pods that tells structure of the application and can be easily replicated.
  + It is Abstraction of Pods