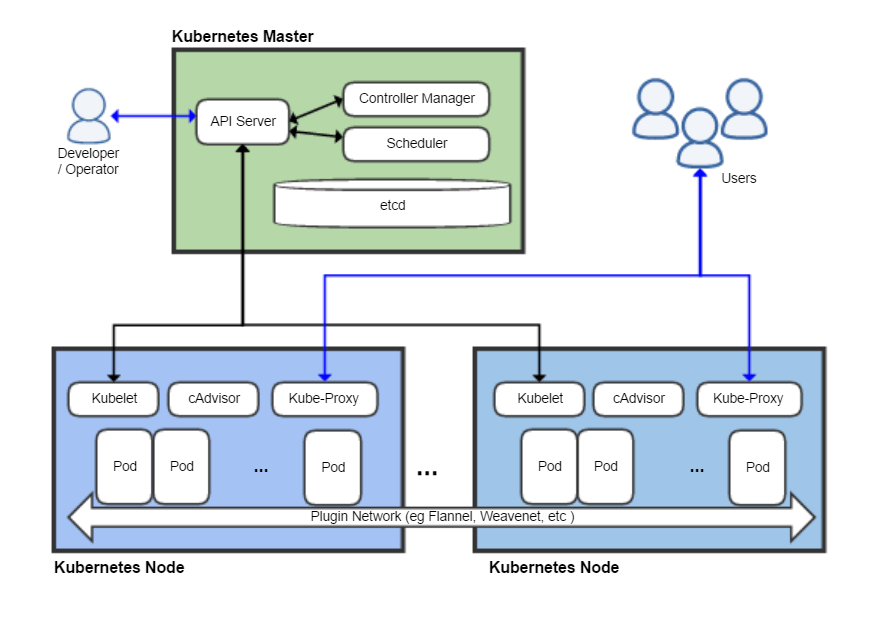
**Kubernetes Architecture - Detailed Explanation**

**1. Introduction to Kubernetes**

**What is Kubernetes?**

Kubernetes (K8s) is an open-source container orchestration platform that automates deployment, scaling, and management of containerized applications. Originally developed by Google, it is now maintained by the Cloud Native Computing Foundation (CNCF).

**Architecture Diagram**



**Importance of Container Orchestration**

Before Kubernetes, deploying applications in containers required manual intervention to manage resources, ensure availability, and handle failures. Kubernetes automates these tasks:  
✅ Efficient workload distribution across nodes.  
✅ Self-healing and automatic rescheduling of failed containers.  
✅ Load balancing and service discovery.  
✅ Rolling updates and rollback support.

**Benefits of Kubernetes**

Scalability: Automatically scales applications based on resource usage.

Fault Tolerance: Reschedules failed workloads to healthy nodes.

Self-Healing: Monitors and restarts unhealthy containers.

Automation: Handles deployment, networking, storage, and security automatically.

**2. Kubernetes Components**

**a) Control Plane Components**

The Control Plane is responsible for managing the entire cluster, making global decisions, and maintaining the desired application state.

**1. API Server (kube-apiserver)**

Acts as the entry point to the Kubernetes cluster.

Receives REST API requests (kubectl commands, external clients, etc.).

Authenticates, validates, and processes API requests.

Stores cluster state in etcd.

**2. Controller Manager (kube-controller-manager)**

Ensures that the actual cluster state matches the desired state.

**Runs multiple controllers:**

Node Controller: Monitors node health and replaces failed nodes.

Replication Controller: Ensures the correct number of pod replicas.

Service Controller: Manages Service discovery and endpoints.

**3. Scheduler (kube-scheduler)**

Assigns new workloads (Pods) to available worker nodes.

Makes scheduling decisions based on CPU, memory, and node availability.

**4. etcd (Key-Value Store)**

Stores the cluster’s state and configuration.

Highly available and distributed across multiple nodes for fault tolerance.

**b) Worker Node Components**

Worker Nodes run the actual applications in the form of Pods.

1**. Kubelet**

An agent running on each worker node.

Ensures containers are running and healthy.

Communicates with the API Server to receive commands.

**2. Kube-Proxy**

Handles networking inside the cluster.

Routes traffic between services and pods.

Implements load balancing for service requests.

**3. Container Runtime (Docker, containerd, CRI-O)**

Responsible for running containers inside the nodes.

Popular container runtimes: Docker, containerd, CRI-O.