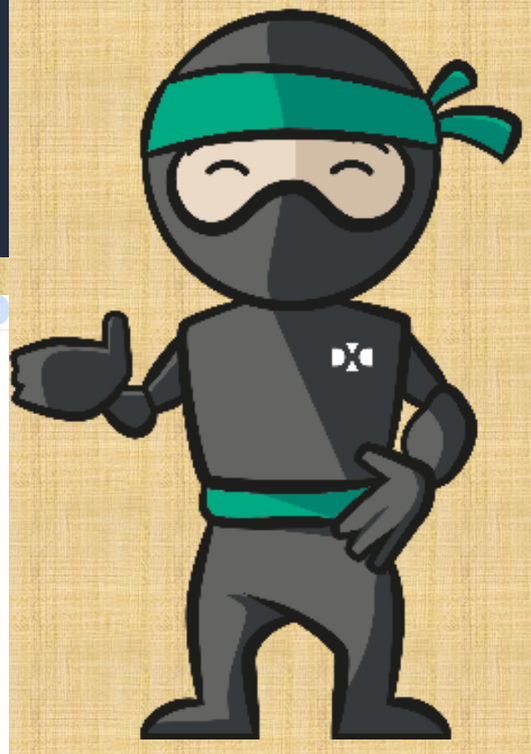
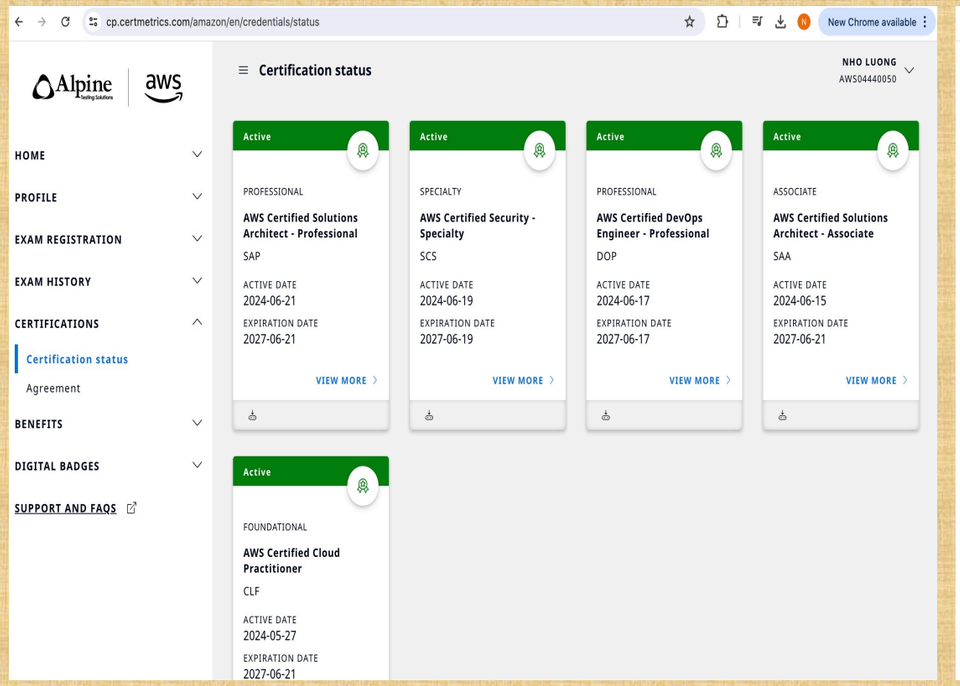
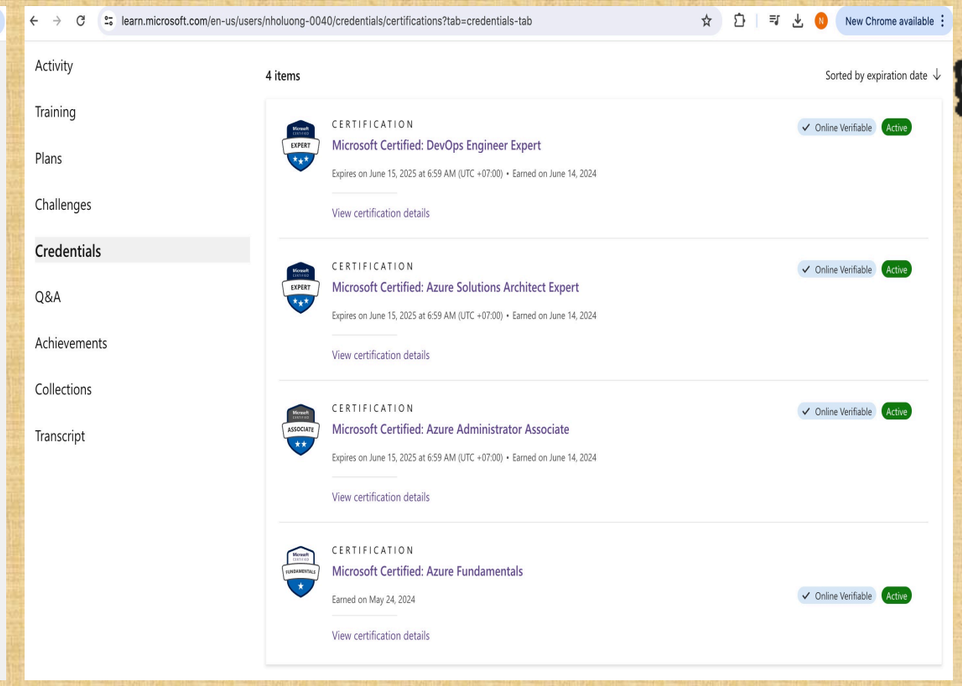
Kubernetes for Beginners on AWS Cloud

Author: Nho Luong

Skill: DevOps Engineer Lead



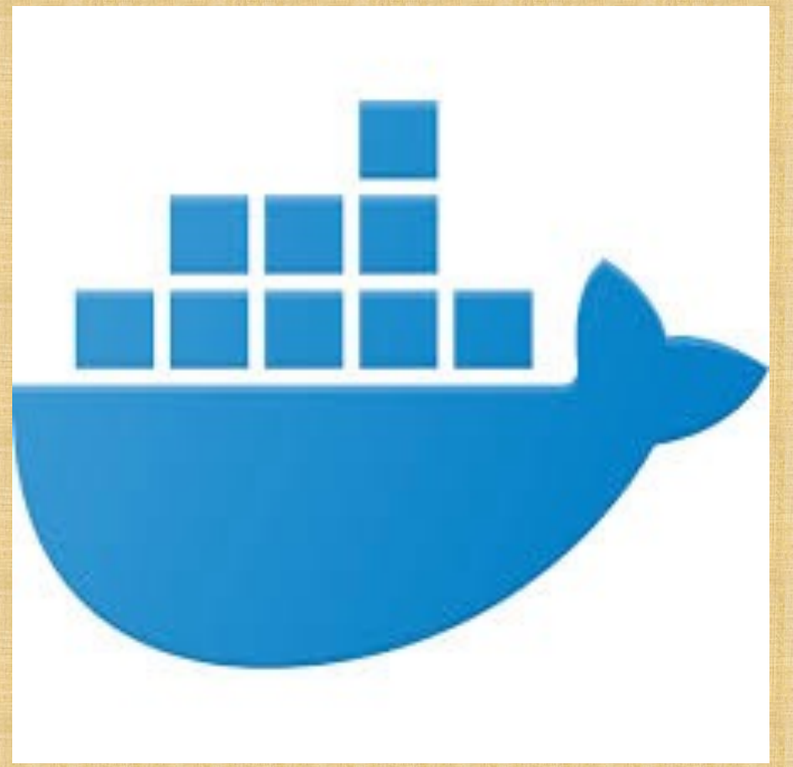




Author: Nho Luong

Skill: DevOps Engineer Lead

Docker Architecture



Author: Nho Luong

Skill: DevOps Engineer Lead

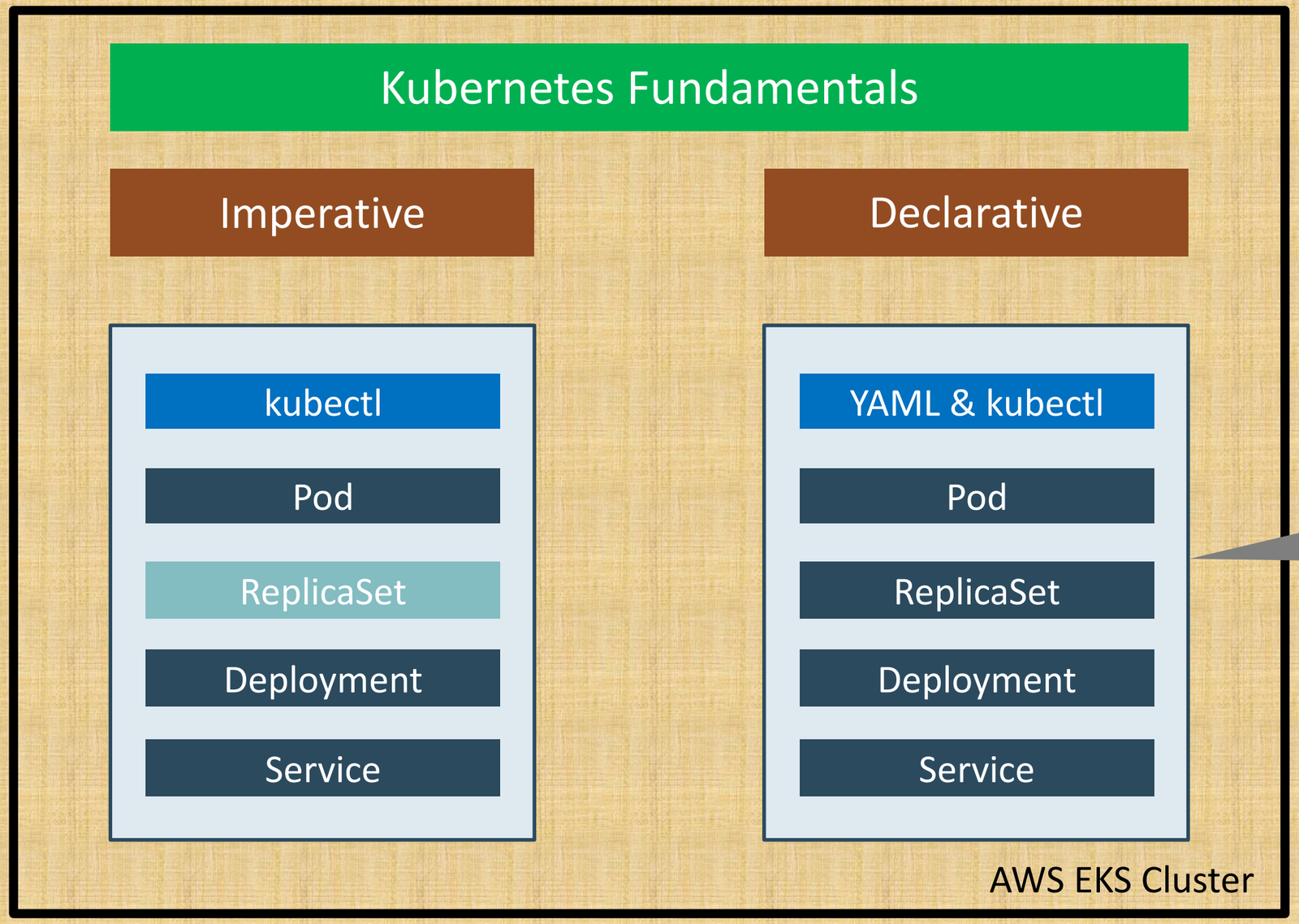
Kubernetes on AWS Cloud

Course Outline

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes on AWS Cloud



Author: Nho Luong

Skill: DevOps Engineer Lead

Free Courses – 2 Hours limitation on Udemy

Kubernetes for Absolute Beginners on AWS Cloud | Part-1

Kubernetes for Absolute Beginners on AWS Cloud | Part-2

Author: Nho Luong

Skill: DevOps Engineer Lead

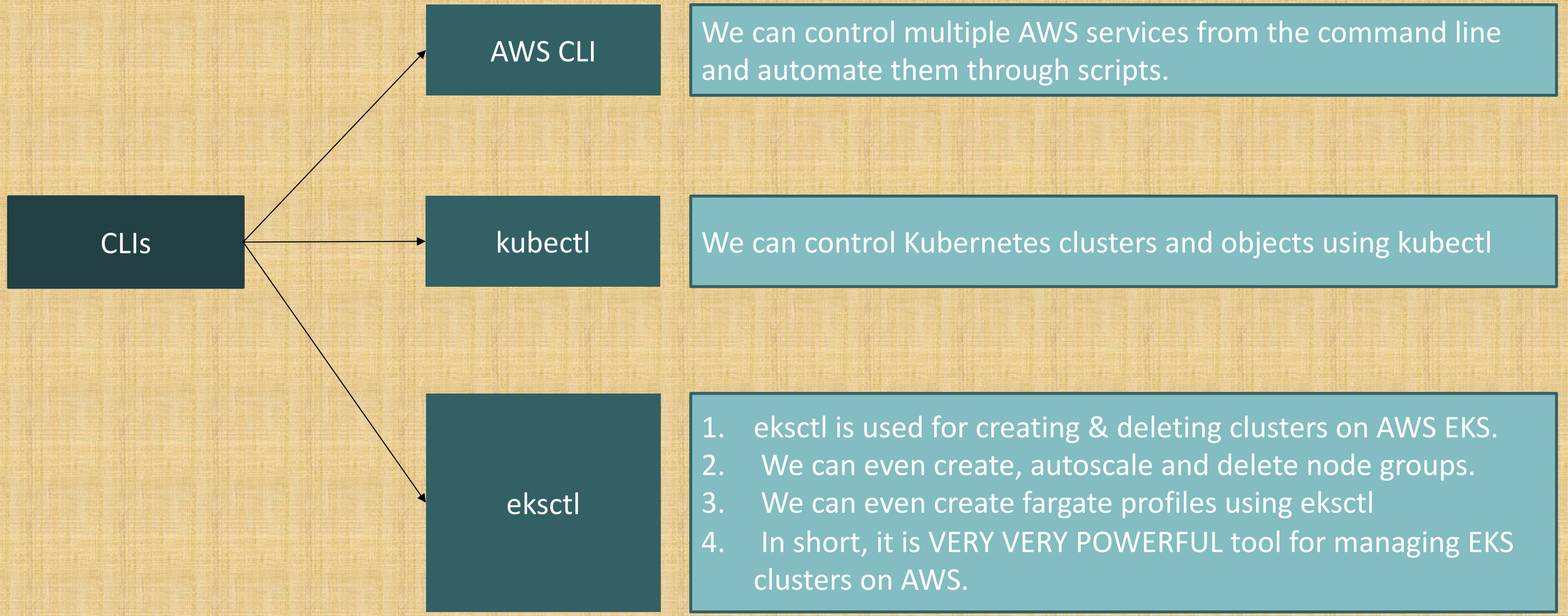
AWS EKS

CLIs

Author: Nho Luong

Skill: DevOps Engineer Lead

AWS EKS Cluster - CLIs



Author: Nho Luong

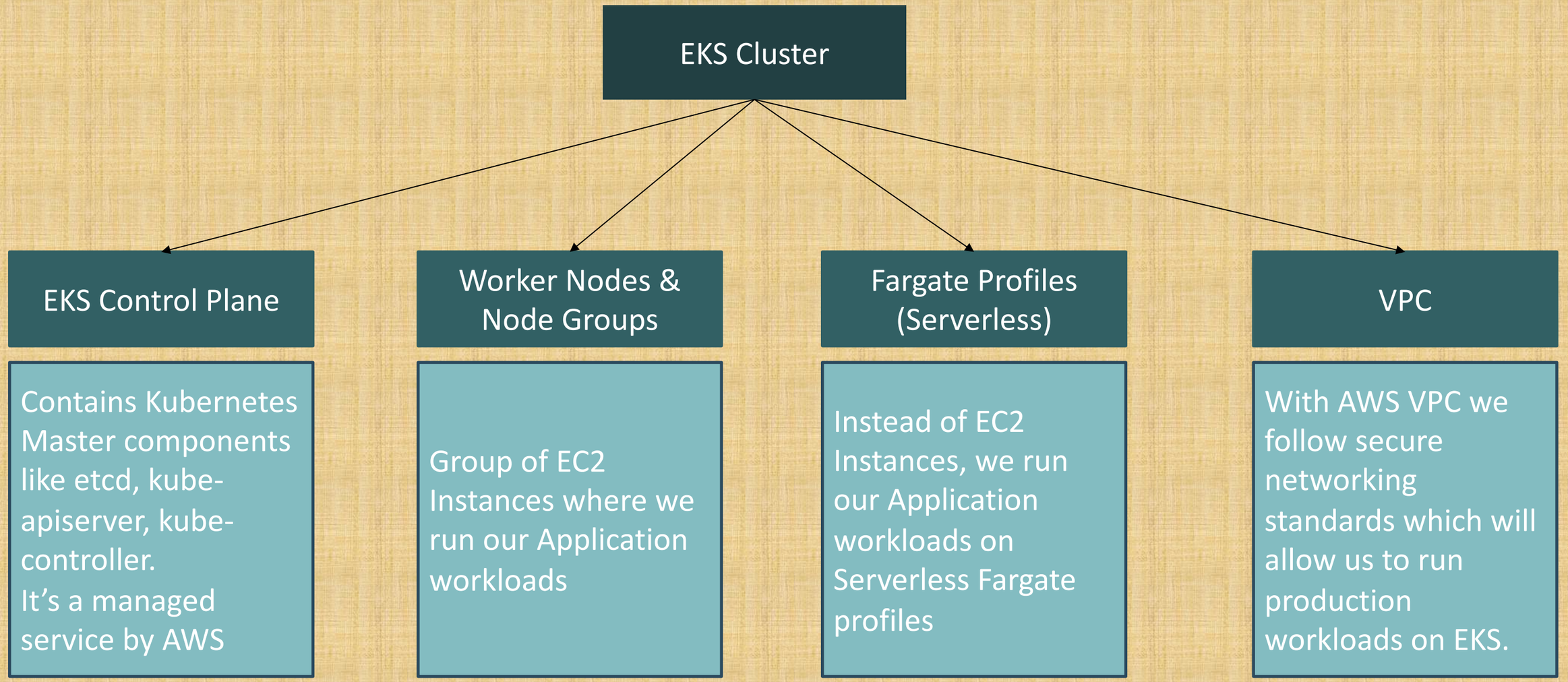
Skill: DevOps Engineer Lead

AWS EKS
Cluster

Author: Nho Luong

Skill: DevOps Engineer Lead

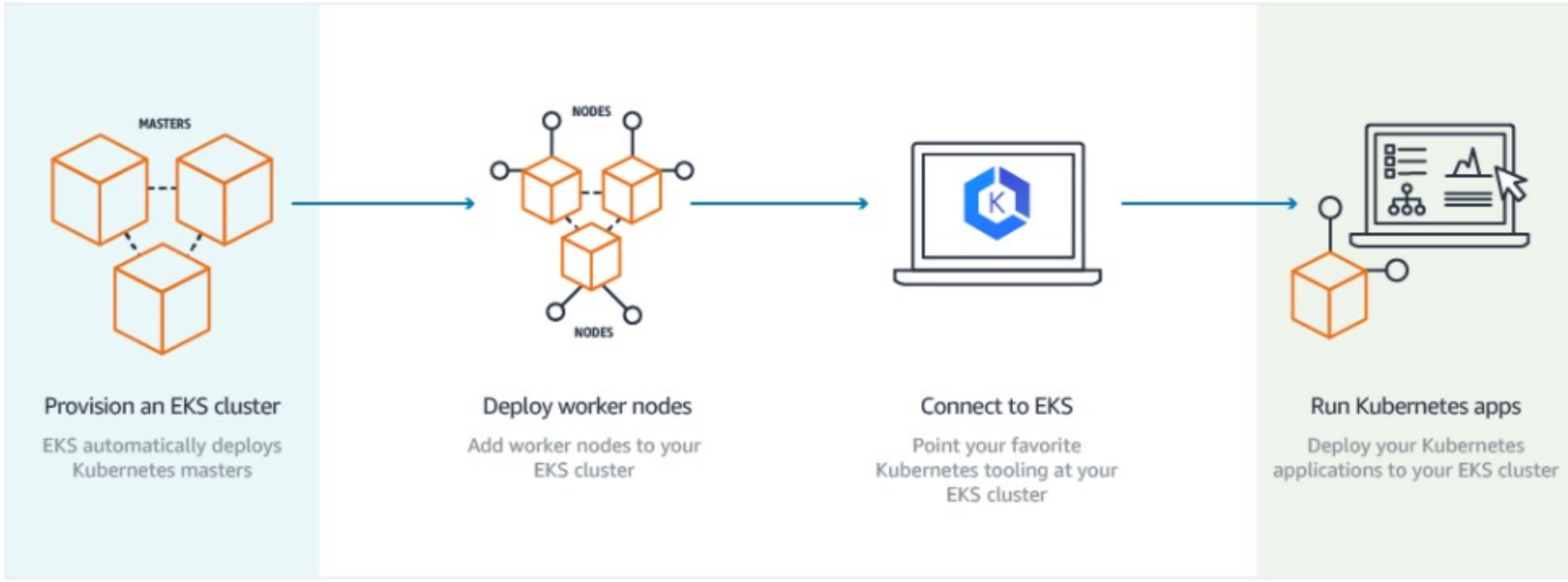
AWS EKS – Core Objects



Author: Nho Luong

Skill: DevOps Engineer Lead

How does EKS work?



Author: Nho Luong

Skill: DevOps Engineer Lead

EKS Cluster – Core Objects Detailed

EKS Control Plane

1. EKS runs a single tenant Kubernetes control plane for each cluster, and control plane infrastructure is not shared across clusters or AWS accounts.
2. This control plane consists of at least two API server nodes and three etcd nodes that run across three Availability Zones within a Region
3. EKS automatically detects and replaces unhealthy control plane instances, restarting them across the Availability Zones within the Region as needed.

Worker Nodes & Node Groups

1. Worker machines in Kubernetes are called nodes. These are EC2 Instances
2. EKS worker nodes run in our AWS account and connect to our cluster's control plane via the cluster API server endpoint.
3. A node group is one or more EC2 instances that are deployed in an EC2 Autoscaling group.
4. All instances in a node group must
    1. Be the same instance type
    2. Be running the same AMI
    3. Use the same EKS worker node IAM role

Author: Nho Luong

Skill: DevOps Engineer Lead

EKS Cluster – Core Objects Detailed

Fargate Profiles	1. AWS Fargate is a technology that provides on-demand, right-sized compute capacity for containers 2. With Fargate, we no longer have to provision, configure, or scale groups of virtual machines to run containers. 3. Each pod running on Fargate has its own isolation boundary and does not share the underlying kernel, CPU resources, memory resources, or elastic network interface with another pod. 4. AWS specially built Fargate controllers that recognizes the pods belonging to fargate and schedules them on Fargate profiles. 5. We will see more in our Fargate learning section.
VPC	1. EKS uses AWS VPC network policies to restrict traffic between control plane components to within a single cluster. 2. Control plane components for a EKS cluster cannot view or receive communication from other clusters or other AWS accounts, except as authorized with Kubernetes RBAC policies. 3. This secure and highly-available configuration makes EKS reliable and recommended for production workloads.

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Architecture

Author: Nho Luong

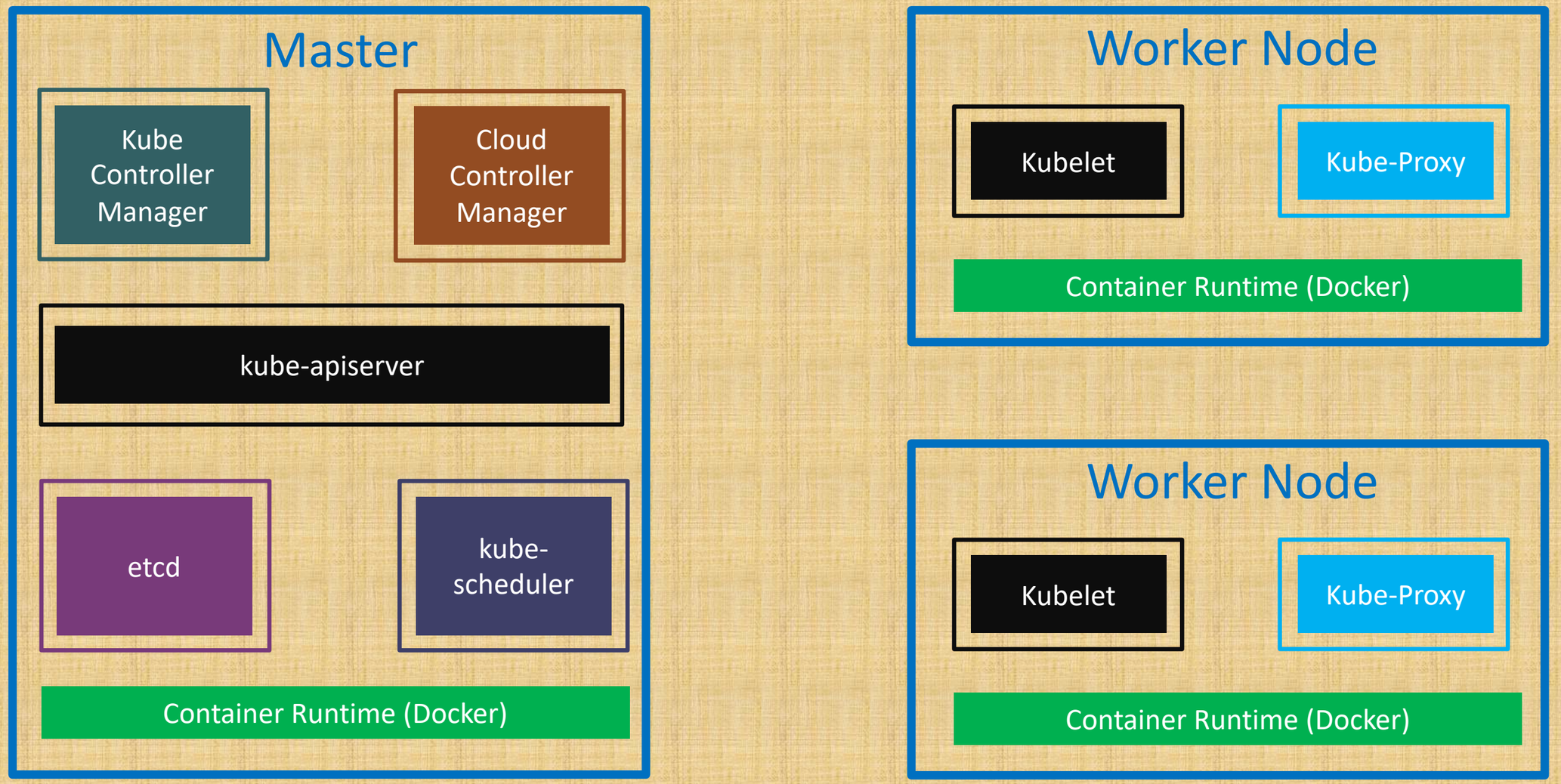
Skill: DevOps Engineer Lead

Kubernetes Architecture

Author: Nho Luong

Skill: DevOps Engineer Lead

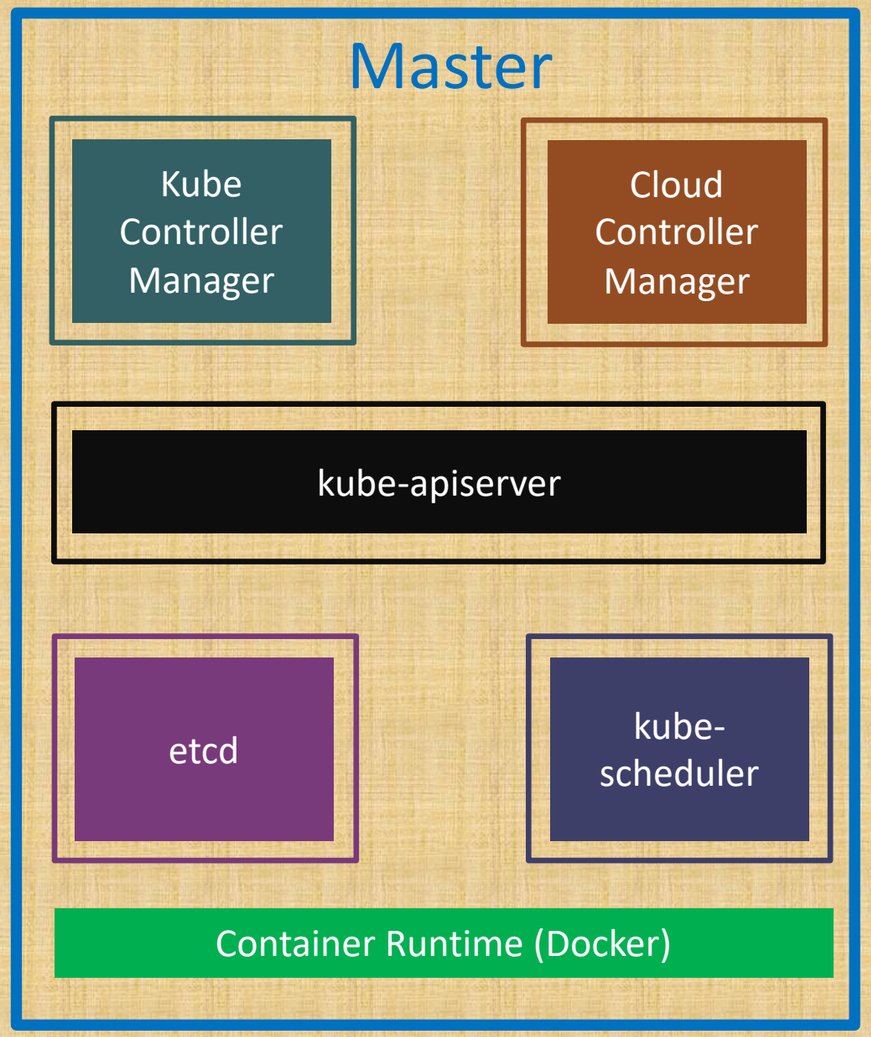
Kubernetes - Architecture



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Architecture - Master

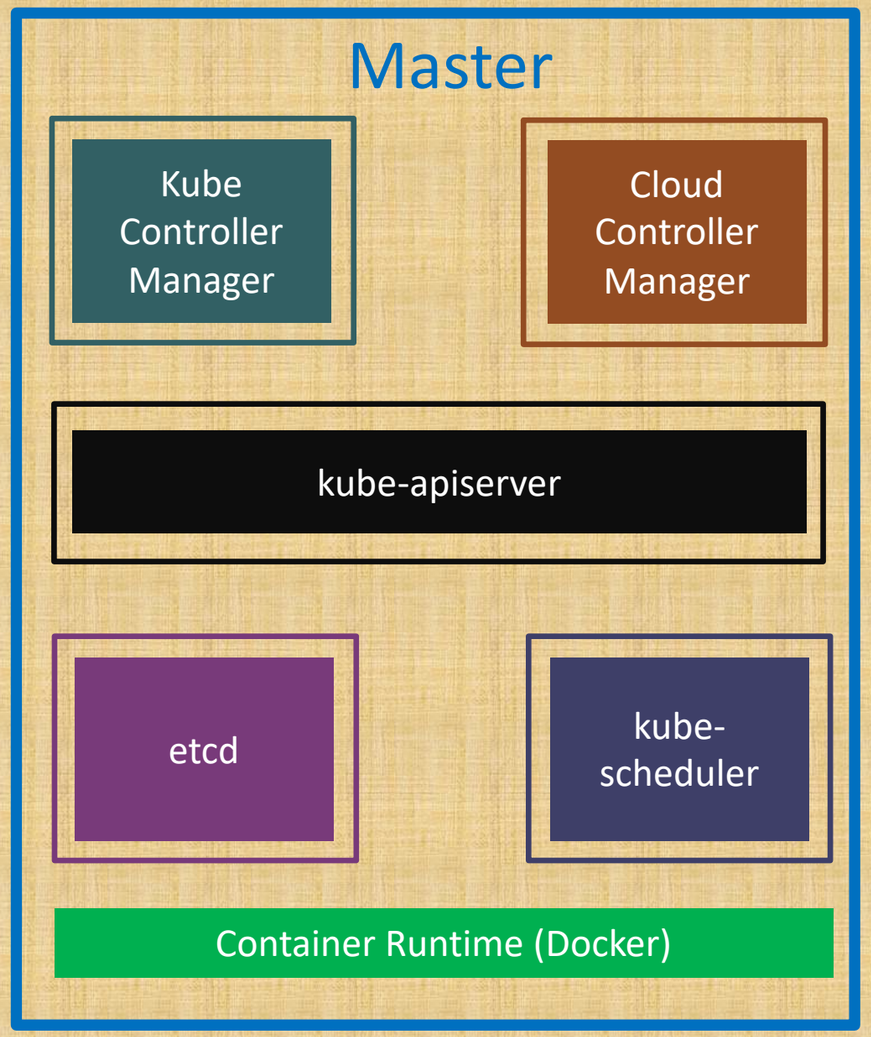


- kube-apiserver
  - It acts as front end for the Kubernetes control plane. It exposes the Kubernetes API
  - Command line tools (like kubectl), Users and even Master components (scheduler, controller manager, etcd) and Worker node components like (Kubelet) everything talk with API Server.
- etcd
  - Consistent and highly-available key value store used as Kubernetes' backing store for all cluster data.
  - It stores all the masters and worker node information.
- kube-scheduler
  - Scheduler is responsible for distributing containers across multiple nodes.
  - It watches for newly created Pods with no assigned node, and selects a node for them to run on.

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Architecture - Master

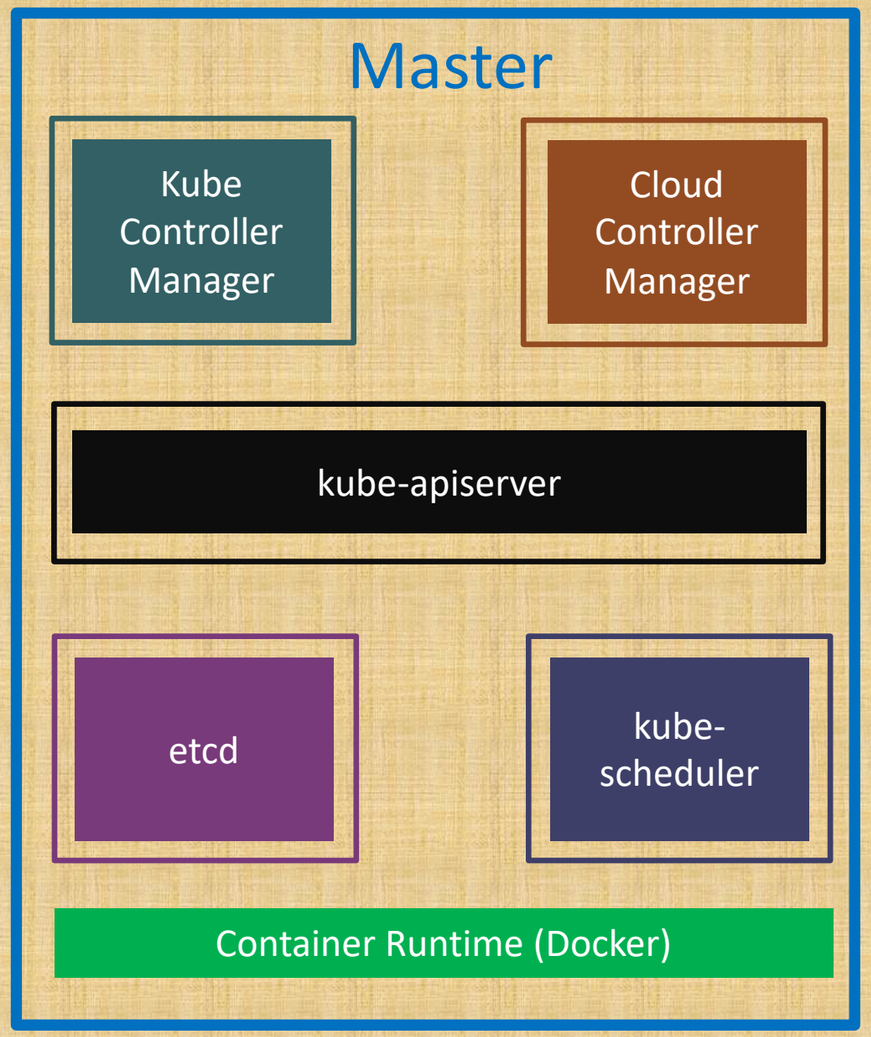


- kube-controller-manager
  - Controllers are responsible for noticing and responding when nodes, containers or endpoints go down. They make decisions to bring up new containers in such cases.
  - Node Controller: Responsible for noticing and responding when nodes go down.
  - Replication Controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.
  - Endpoints Controller: Populates the Endpoints object (that is, joins Services & Pods)
  - Service Account & Token Controller: Creates default accounts and API Access for new namespaces.

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Architecture - Master

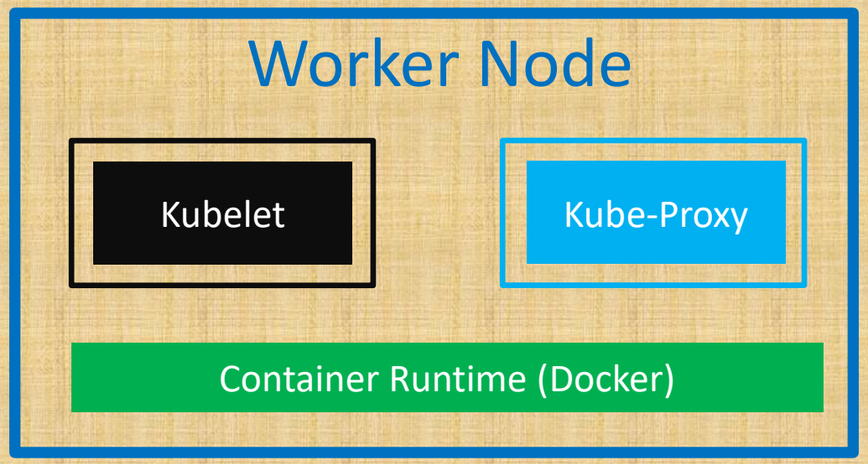


- cloud-controller-manager
  - A Kubernetes control plane component that embeds cloud-specific control logic.
  - It only runs controllers that are specific to your cloud provider.
  - On-Premise Kubernetes clusters will not have this component.
  - Node controller: For checking the cloud provider to determine if a node has been deleted in the cloud after it stops responding
  - Route controller: For setting up routes in the underlying cloud infrastructure
  - Service controller: For creating, updating and deleting cloud provider load balancer

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Architecture – Worker Nodes



- Container Runtime
  - Container Runtime is the underlying software where we run all these Kubernetes components.
  - We are using Docker, but we have other runtime options like rkt, container-d etc.

- Kubelet
  - Kubelet is the agent that runs on every node in the cluster
  - This agent is responsible for making sure that containers are running in a Pod on a node.
- Kube-Proxy
  - It is a network proxy that runs on each node in your cluster.
  - It maintains network rules on nodes
  - In short, these network rules allow network communication to your Pods from network sessions inside or outside of your cluster.

Author: Nho Luong

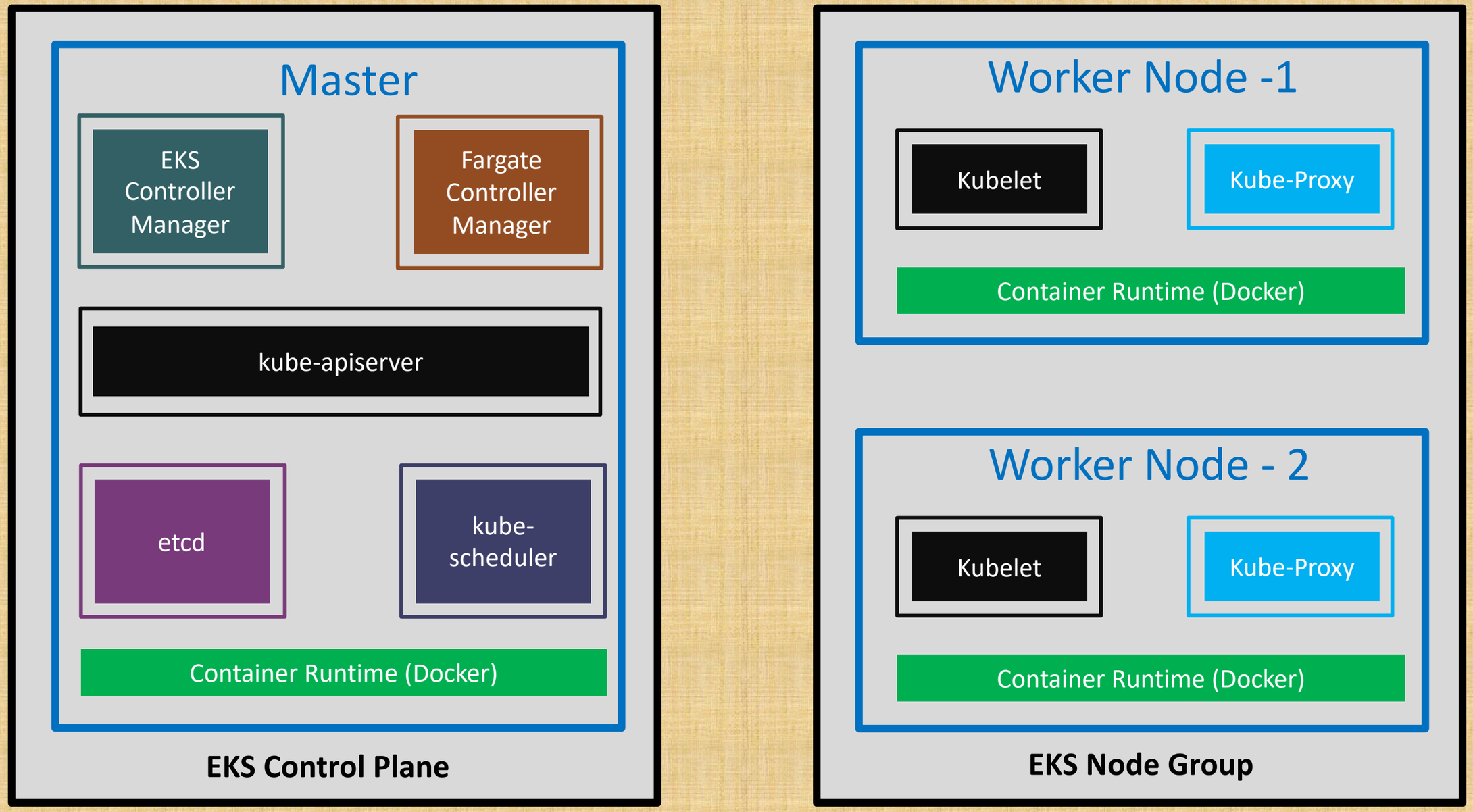
Skill: DevOps Engineer Lead

AWS EKS
Cluster

Author: Nho Luong

Skill: DevOps Engineer Lead

EKS Kubernetes - Architecture



Author: Nho Luong

Skill: DevOps Engineer Lead

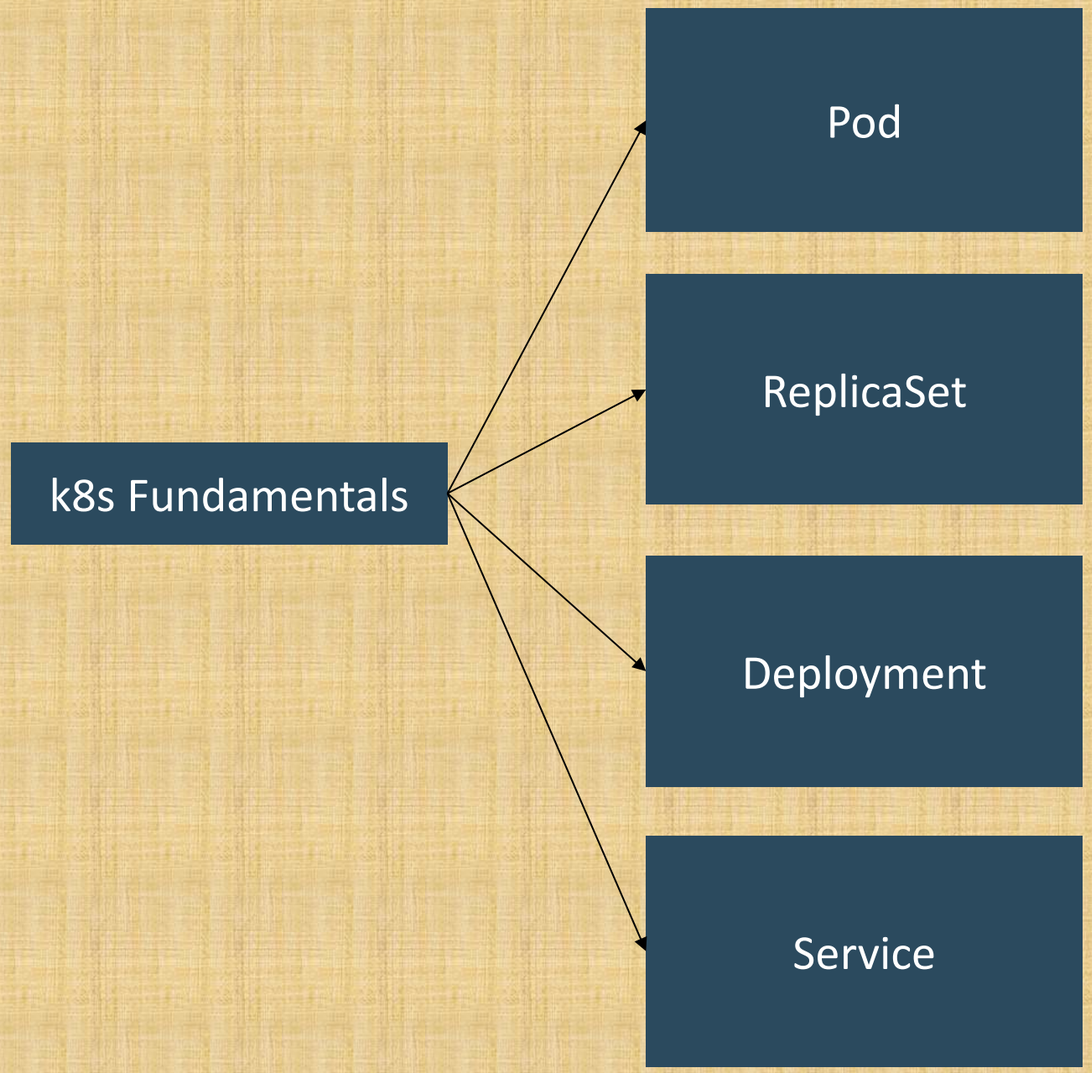
Kubernetes Fundamentals

Pod, ReplicaSet, Deployment & Service

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - Fundamentals



A POD is a single instance of an Application.
A POD is the smallest object, that you can create in Kubernetes.

A ReplicaSet will maintain a stable set of replica Pods running at any given time.
In short, it is often used to guarantee the availability of a specified number of identical Pods

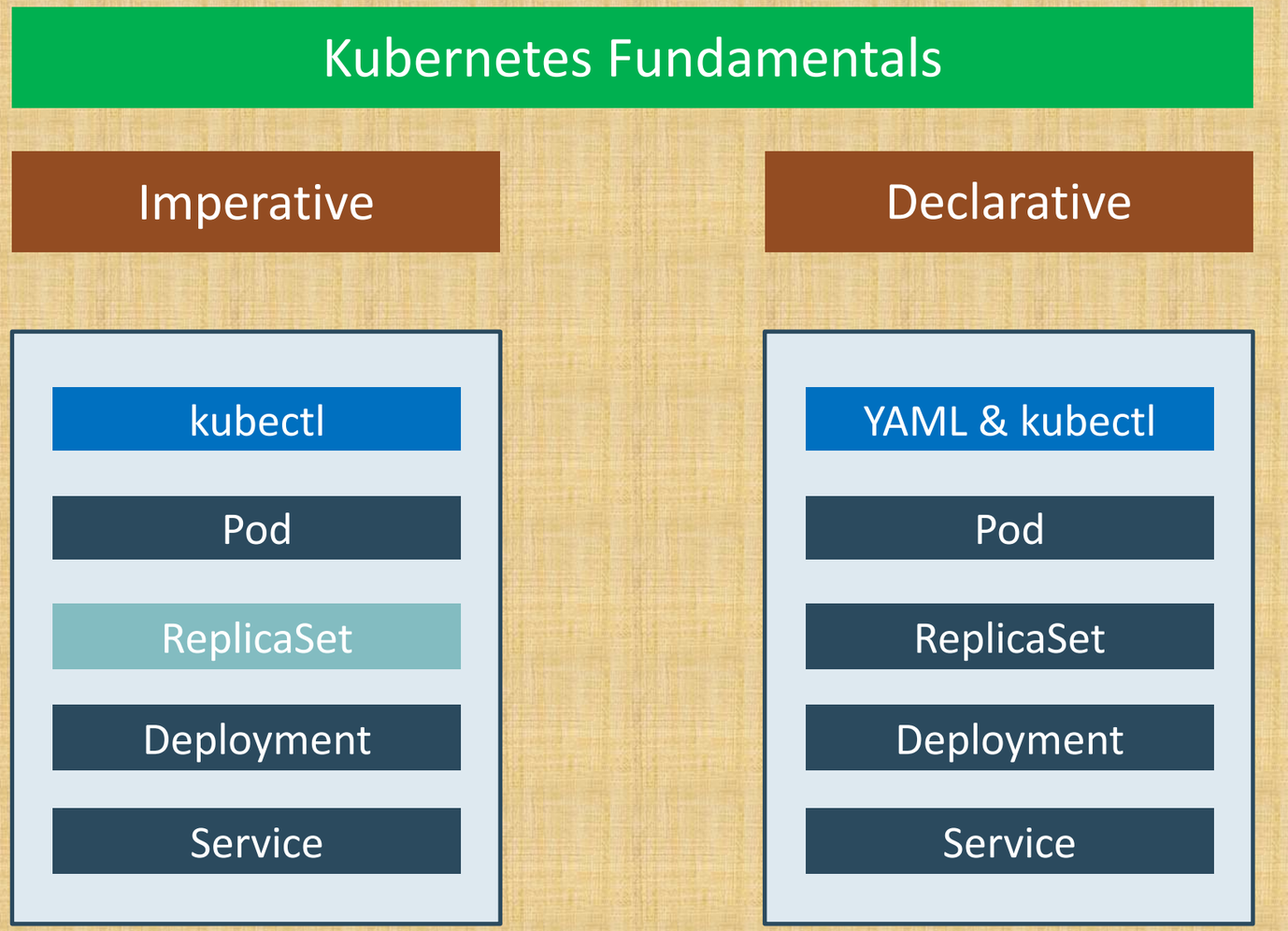
A Deployment runs multiple replicas of your application and automatically replaces any instances that fail or become unresponsive. Rollout & rollback changes to applications. Deployments are well-suited for stateless applications.

A service is an abstraction for pods, providing a stable, so called virtual IP (VIP) address.
In simple terms, service sits Infront of a POD and acts as a load balancer.

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - Imperative & Declarative



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes

POD

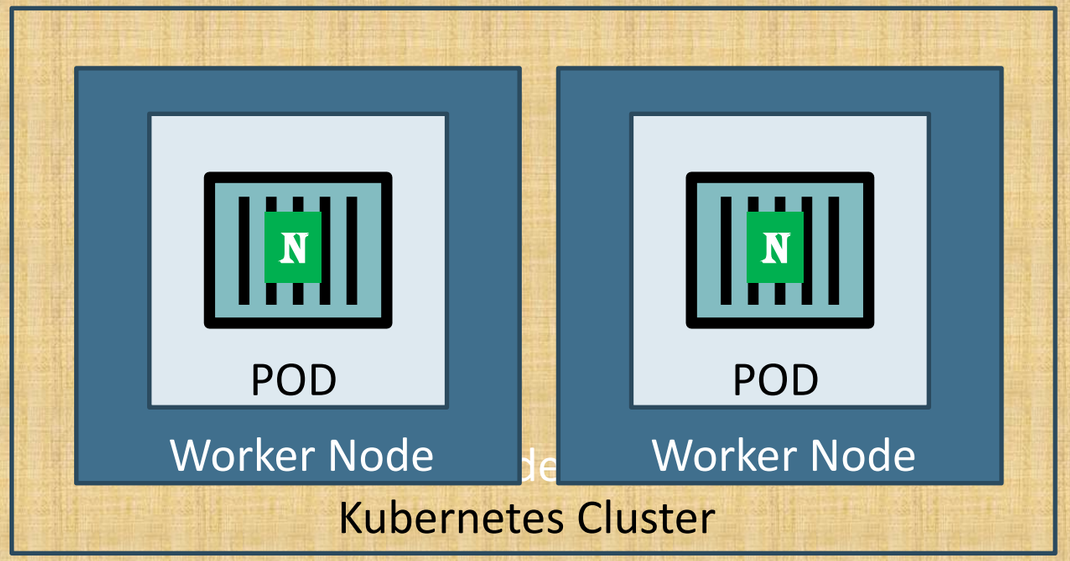
Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - POD

- With Kubernetes our core goal will be to deploy our applications in the form of containers on worker nodes in a k8s cluster.
- Kubernetes does not deploy containers directly on the worker nodes.
- Container is encapsulated in to a Kubernetes Object named POD.
- A POD is a single instance of an application.
- A POD is the smallest object that we can create in Kubernetes.

Nginx Container Image

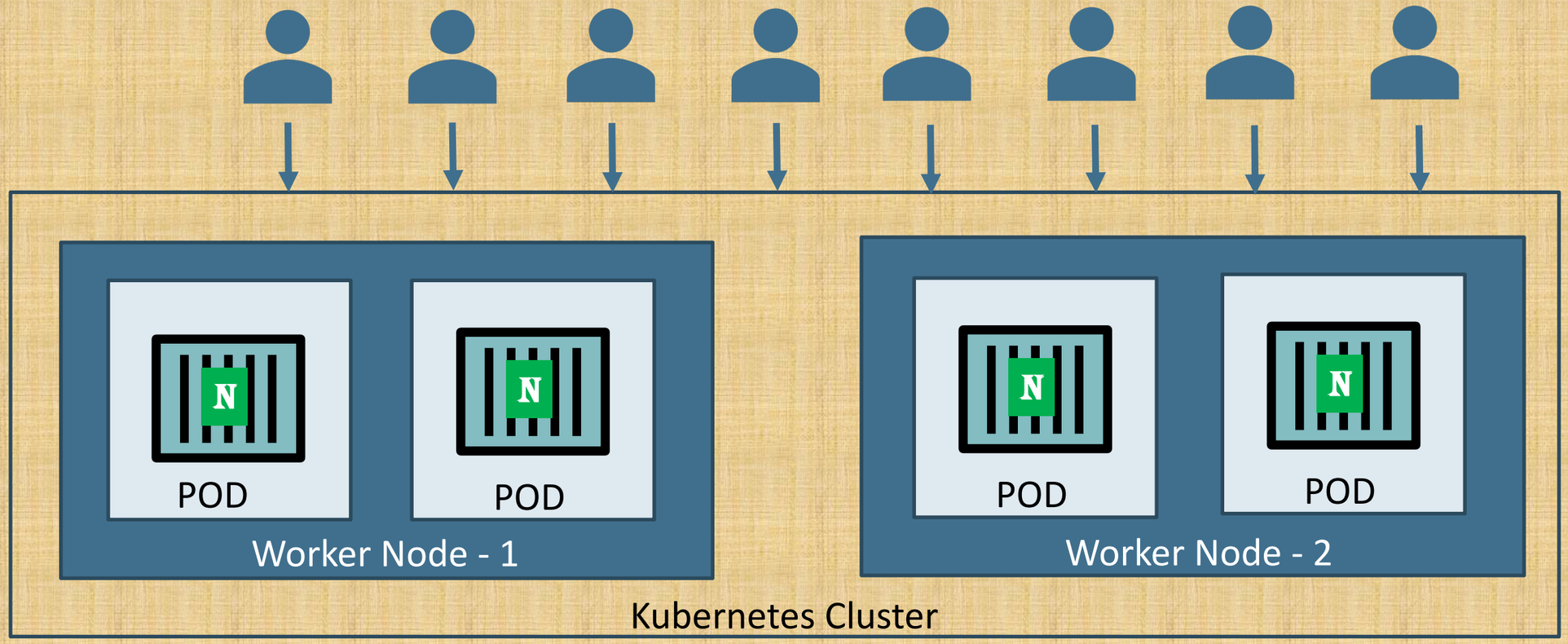


Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - POD

- PODs generally have one to one relationship with containers.
- To scale up we create new POD and to scale down we delete the POD.

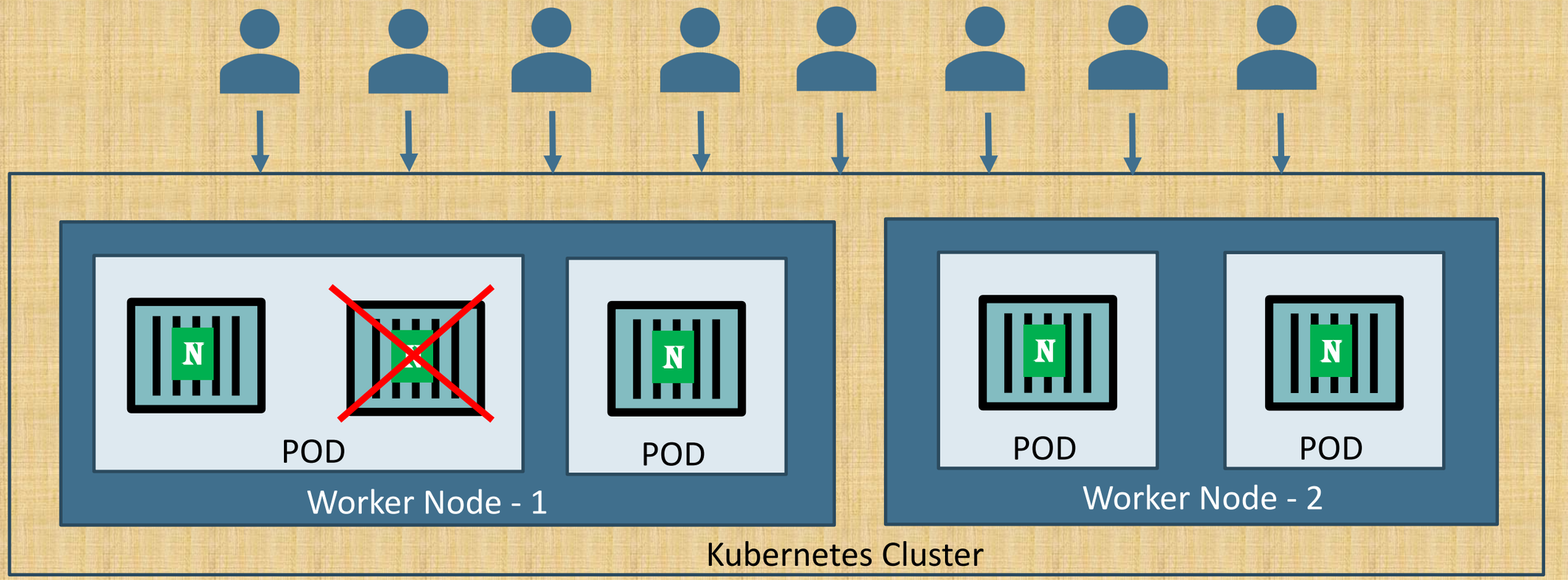


Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – PODs

- We cannot have multiple containers of same kind in a single POD.
- Example: Two NGINX containers in single POD serving same purpose is not recommended.

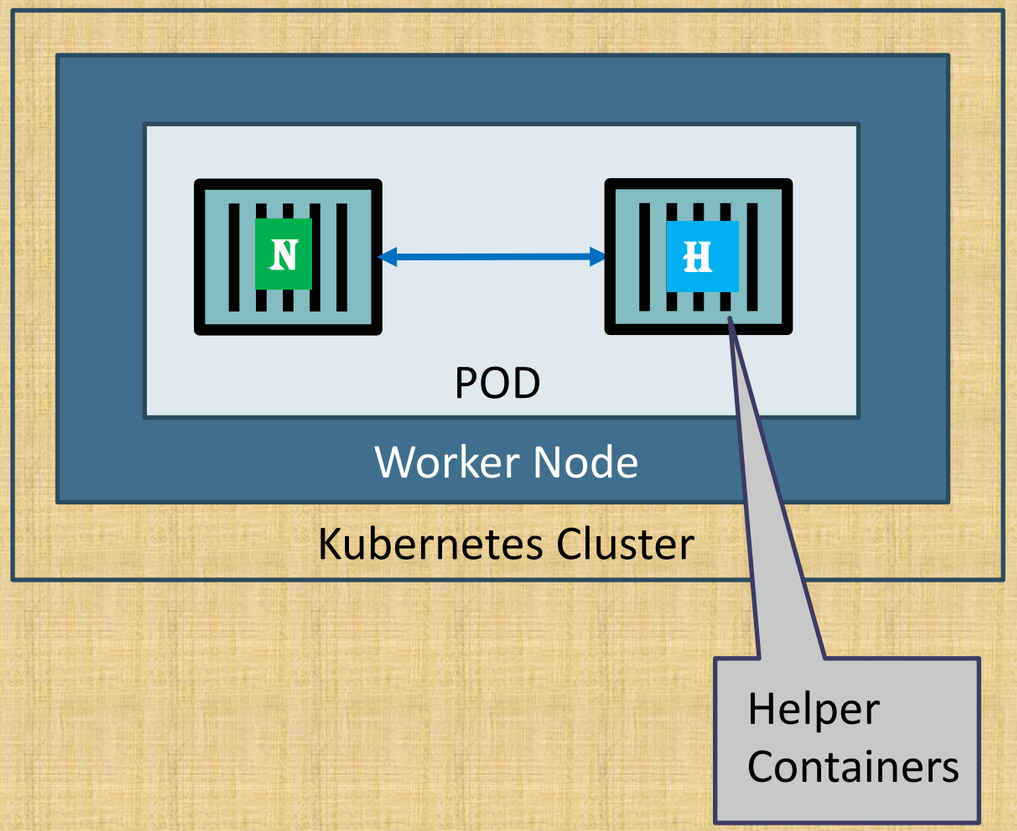


Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – Multi-Container Pods

- We can have multiple containers in a single POD, provided they are not of same kind.
- Helper Containers (Side-car)
  - Data Pullers: Pull data required by Main Container
  - Data pushers: Push data by collecting from main container (logs)
  - Proxies: Writes static data to html files using Helper container and Reads using Main Container.
- Communication
  - The two containers can easily communicate with each other easily as they share same network space.
  - They can also easily share same storage space.
- Multi-Container Pods is a rare use-case and we will try to focus on core fundamentals.



Node

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes
PODs
Demo

Author: Nho Luong

Skill: DevOps Engineer Lead

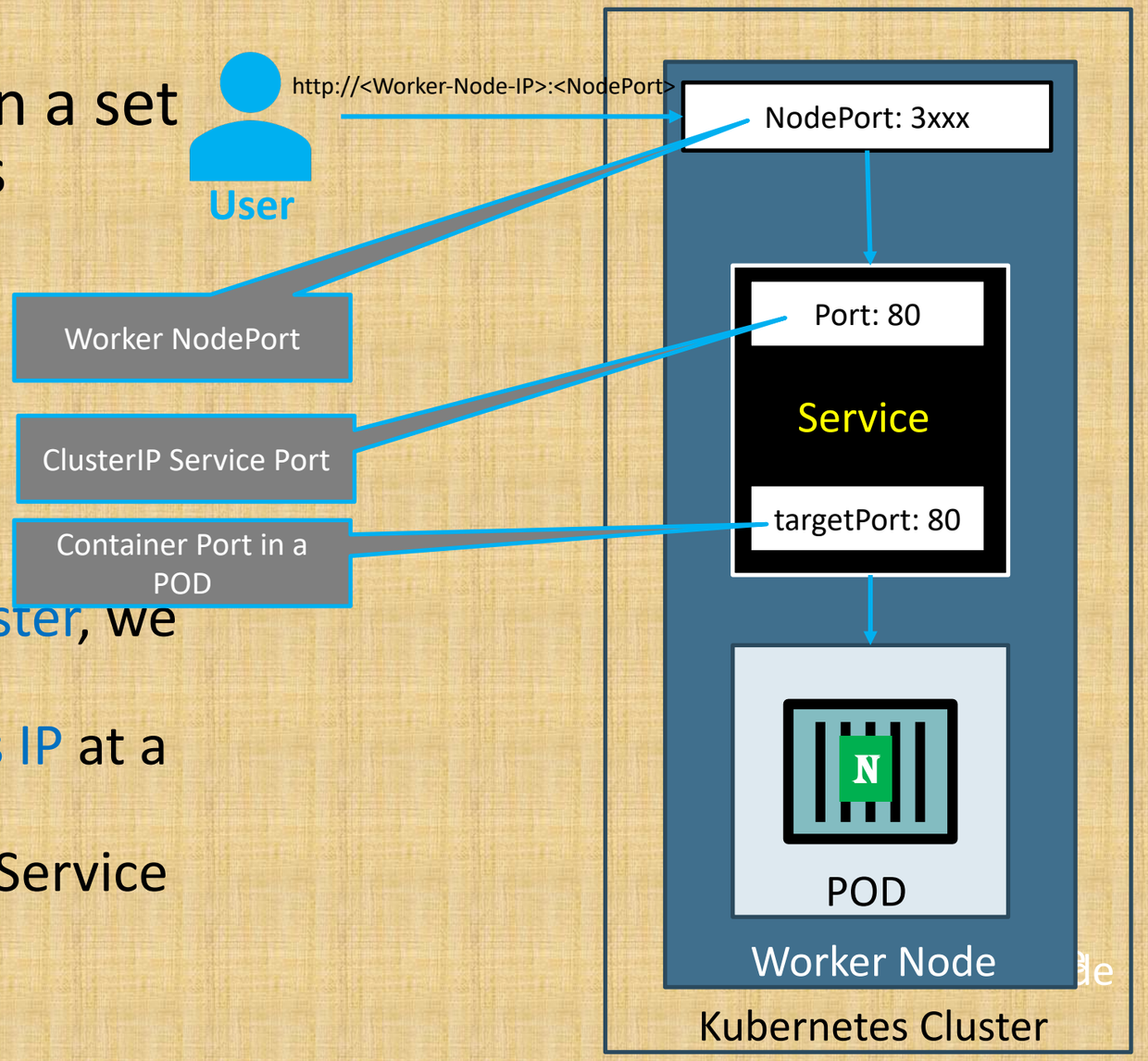
Kubernetes
Services - NodePort

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – Service - NodePort

- We can expose an application running on a set of PODs using different types of Services available in k8s.
  - ClusterIP
  - NodePort
  - LoadBalancer
- NodePort Service
  - To access our application outside of k8s cluster, we can use NodePort service.
  - Exposes the Service on each Worker Node's IP at a static port (nothing but NodePort).
  - A ClusterIP Service, to which the NodePort Service routes, is automatically created.
  - Port Range 30000-32767



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes
POD & NodePort Service
Demo

Author: Nho Luong

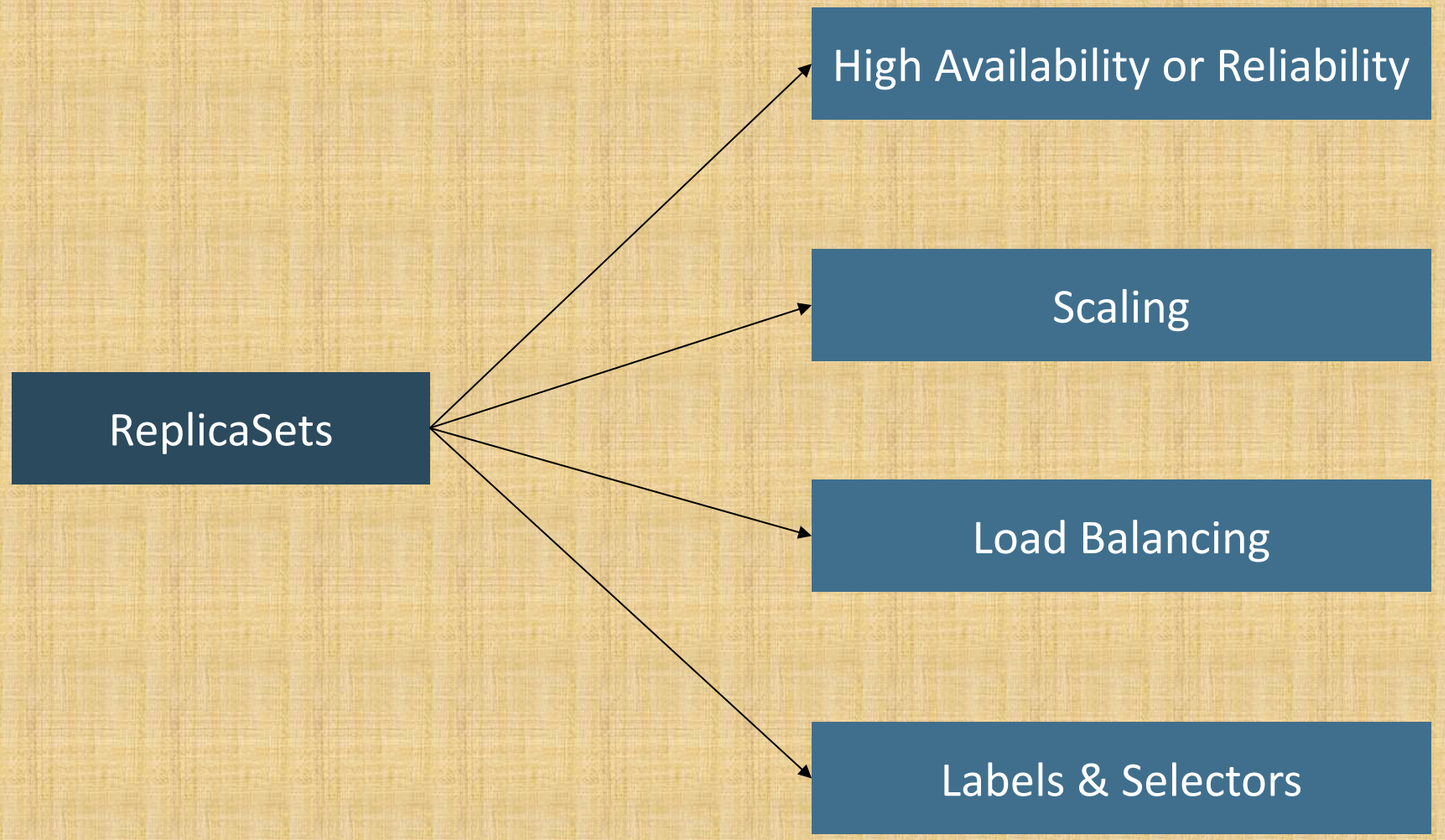
Skill: DevOps Engineer Lead

Kubernetes
ReplicaSets

Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - ReplicaSets



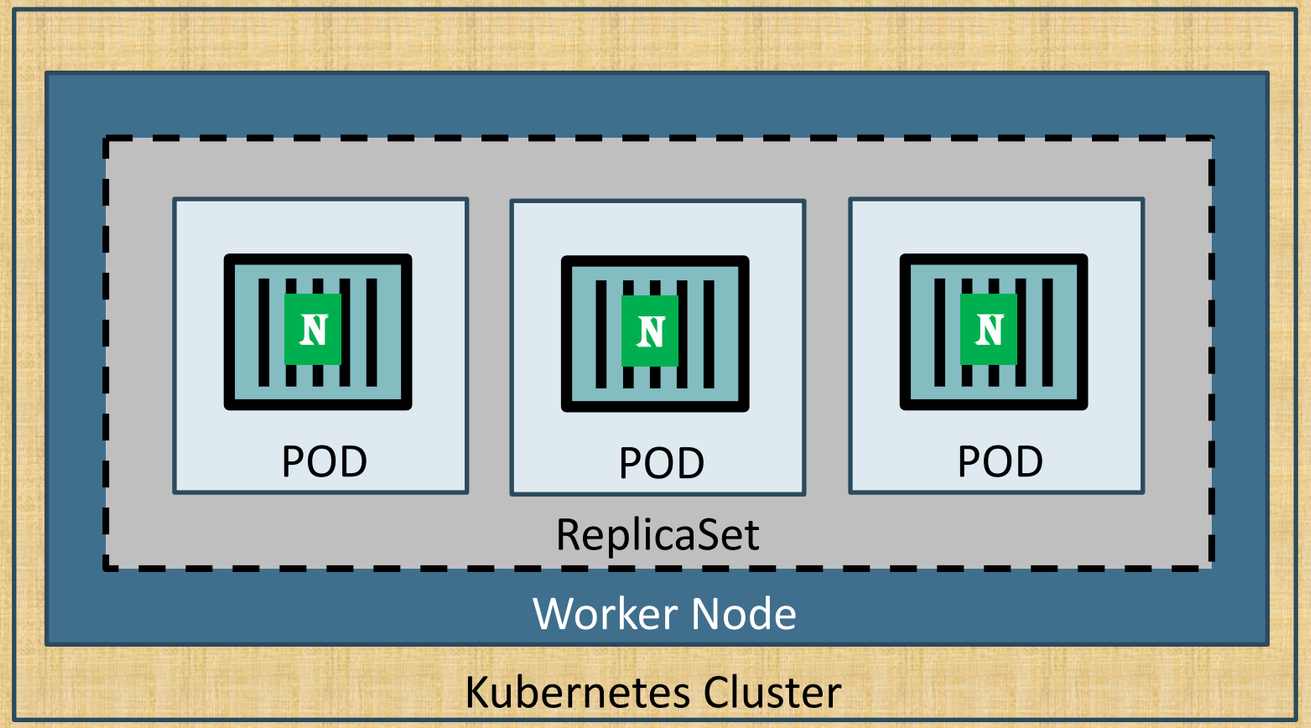
Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – ReplicaSet

- A ReplicaSet's purpose is to maintain a stable set of replica Pods running at any given time.
- If our application crashes (any pod dies), replicaset will recreate the pod immediately to ensure the configured number of pods running at any given time.

Reliability
Or
High Availability

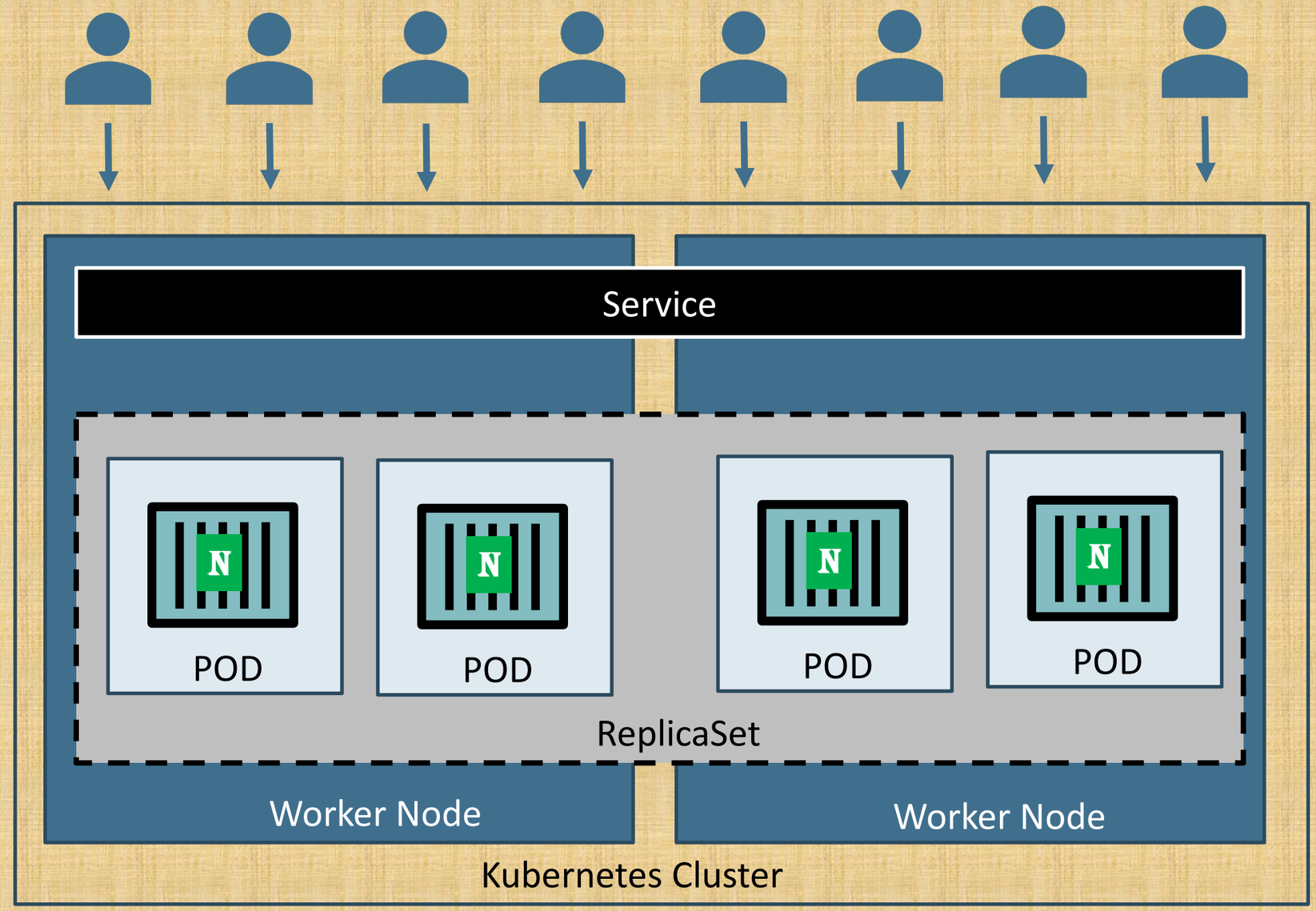


Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – ReplicaSet

- Load Balancing
- To avoid overloading of traffic to single pod we can use load balancing.
- Kubernetes provides pod load balancing out of the box using Services for the pods which are part of a ReplicaSet
- Labels & Selectors are the key items which ties all 3 together (Pod, ReplicaSet & Service), we will know in detail when we are writing YAML manifests for these objects

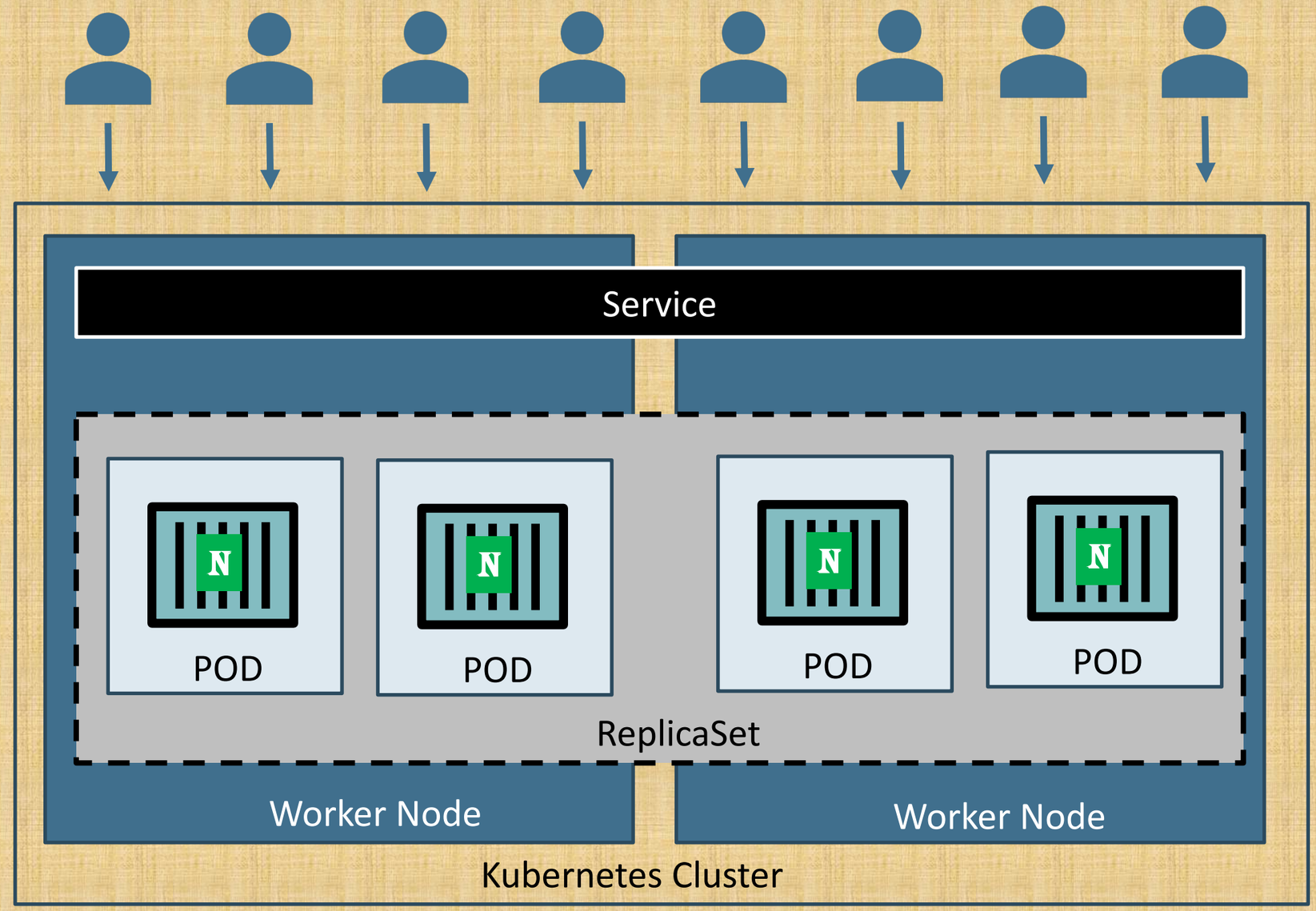


Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes – ReplicaSet

- Scaling
- When load become too much for the number of existing pods, Kubernetes enables us to easily scale up our application, adding additional pods as needed.
- This is going to be seamless and super quick.



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes ReplicaSets Demo

Author: Nho Luong

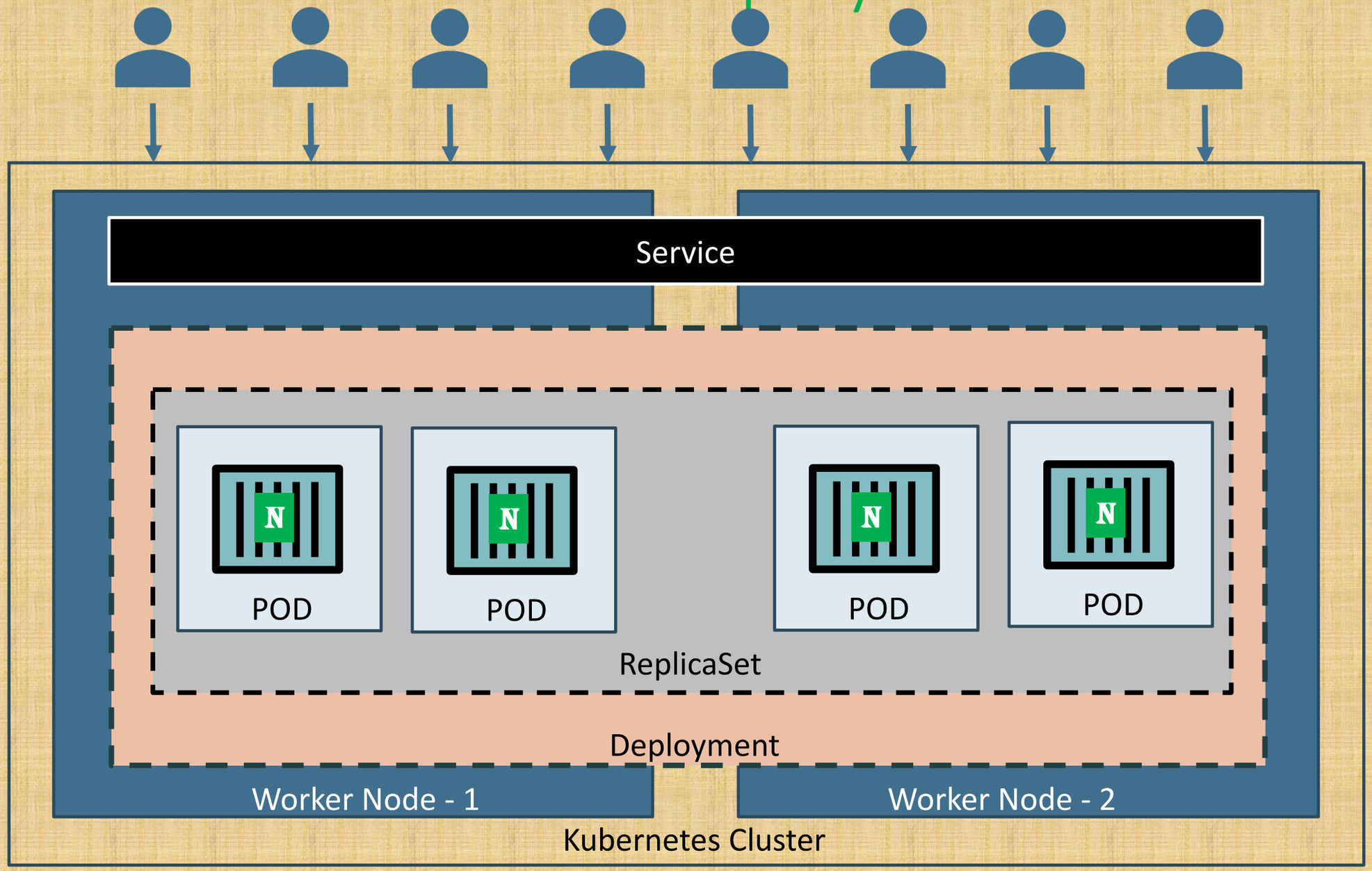
Skill: DevOps Engineer Lead

Kubernetes
Deployments

Author: Nho Luong

Skill: DevOps Engineer Lead

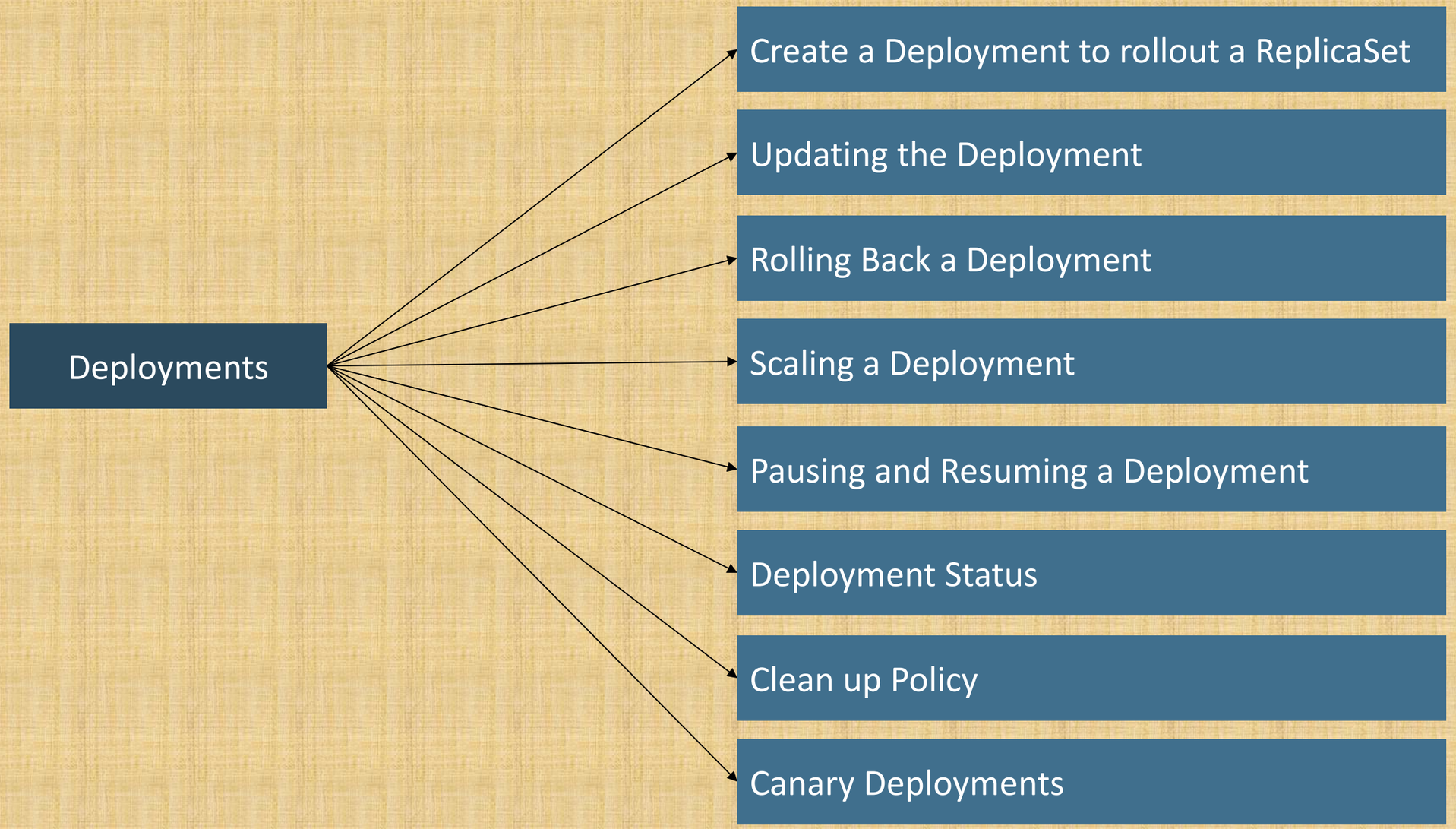
Kubernetes – Deployments



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes - Deployment



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes Deployments Demo

Author: Nho Luong

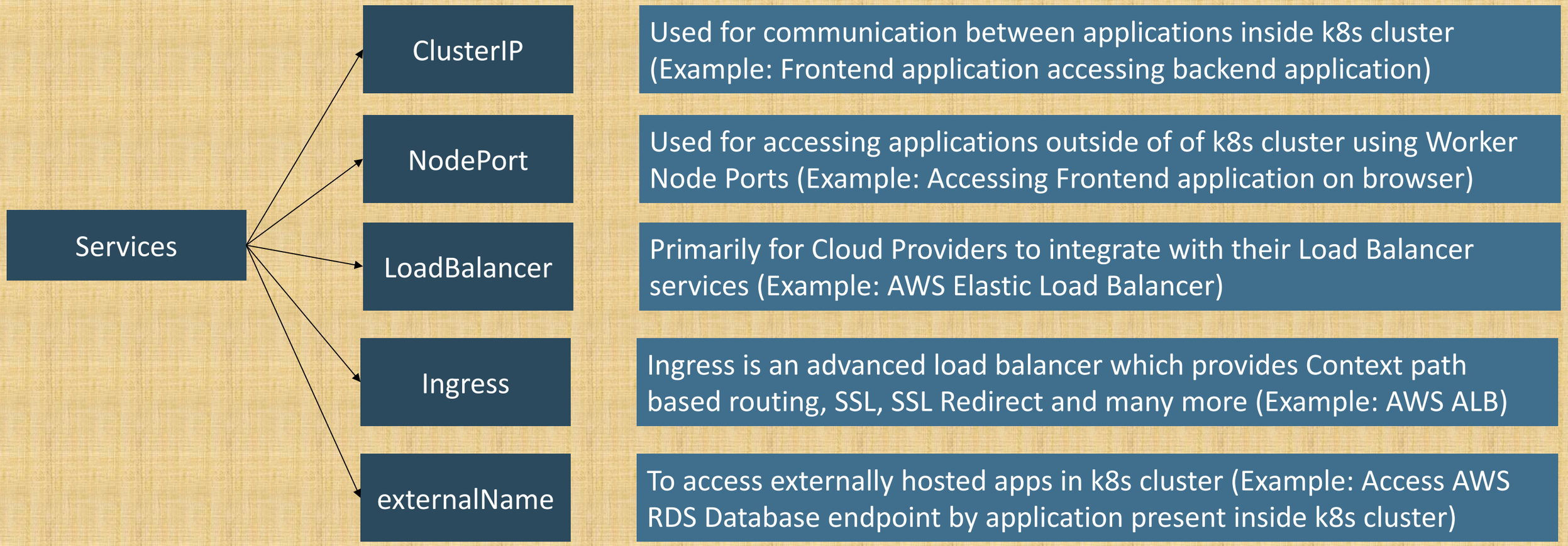
Skill: DevOps Engineer Lead

Kubernetes
Services

Author: Nho Luong

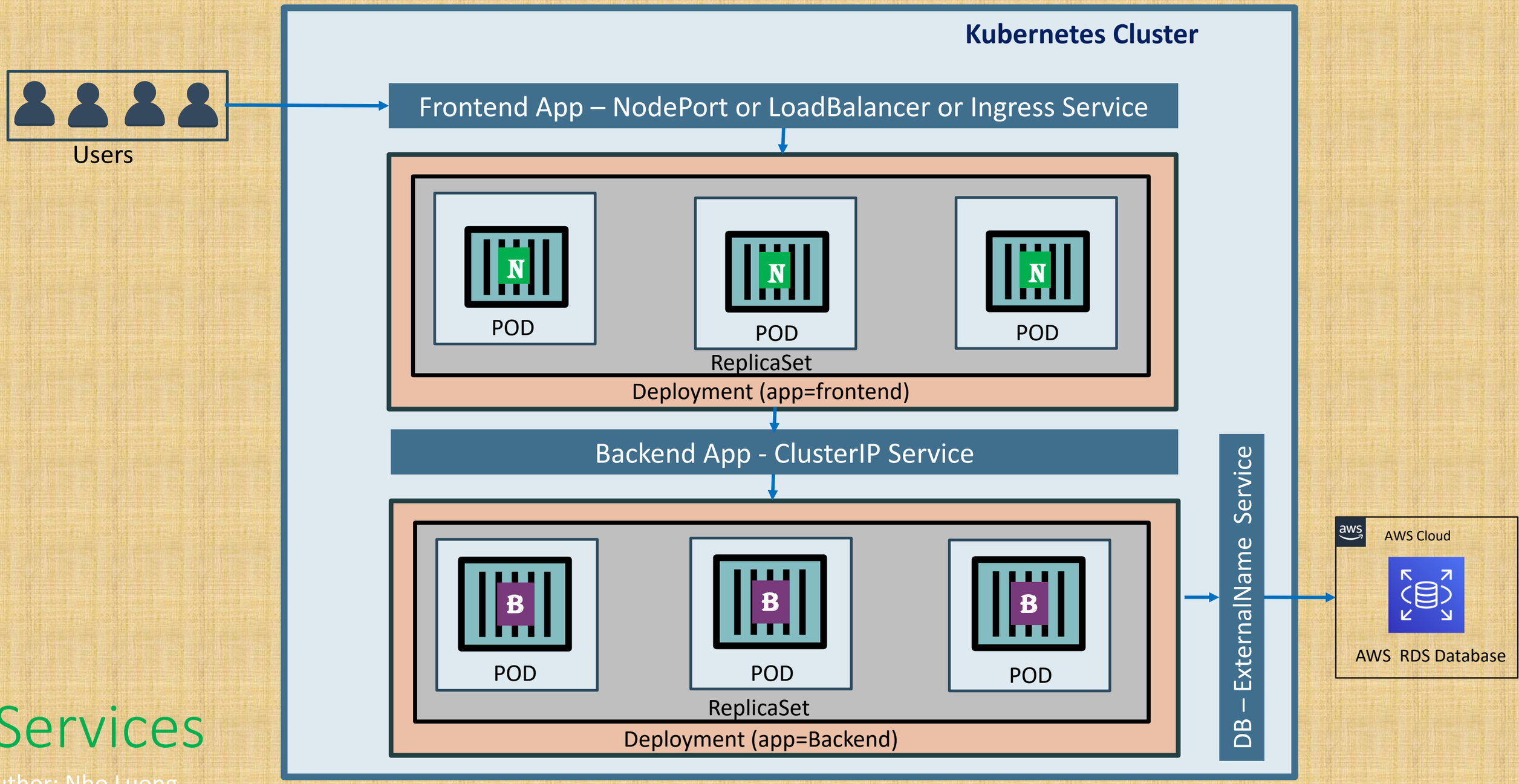
Skill: DevOps Engineer Lead

Kubernetes - Services



Author: Nho Luong

Skill: DevOps Engineer Lead



Services

Author: Nho Luong

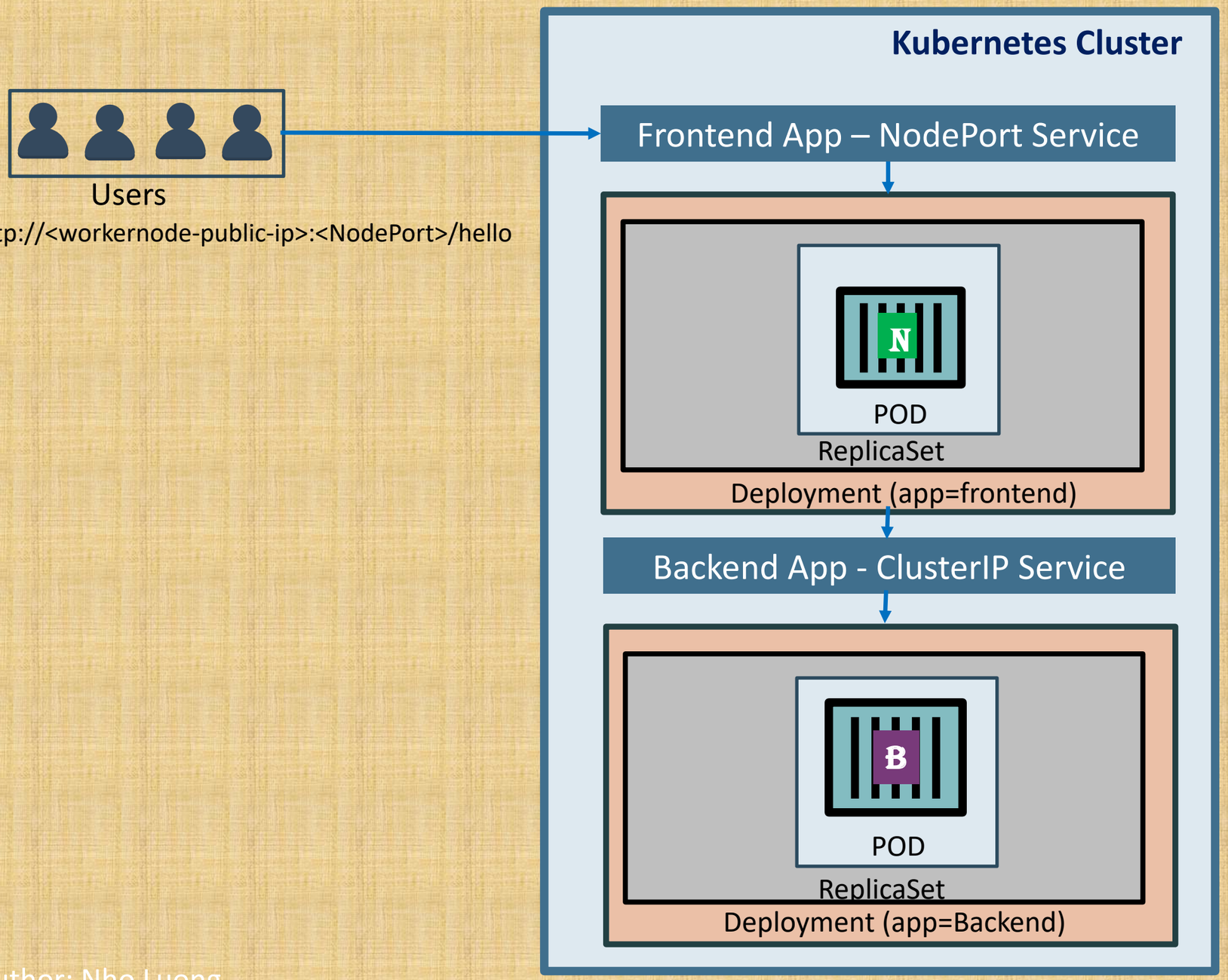
Skill: DevOps Engineer Lead

Kubernetes Services Demo

Author: Nho Luong

Skill: DevOps Engineer Lead

Services Demo



Author: Nho Luong

Skill: DevOps Engineer Lead

Kubernetes
YAML Basics

Author: Nho Luong

Skill: DevOps Engineer Lead

YAML Basics

- YAML is not a Markup Language
- YAML is used to store information about different things
- We can use YAML to define key, Value pairs like variables, lists and objects
- YAML is very similar to JSON (Javascript Object Notation)
- YAML primarily focuses on readability and user friendliness
- YAML is designed to be clean and easy to read
- We can define YAML files with two different extensions
  - abc.yml
  - abc.yaml

Author: Nho Luong

Skill: DevOps Engineer Lead

YAML Basics

- YAML Comments
- YAML Key Value Pairs
- YAML Dictionary or Map
- YAML Array / Lists
- YAML Spaces
- YAML Document Separator

Author: Nho Luong

Skill: DevOps Engineer Lead

Thank You

Author: Nho Luong

Skill: DevOps Engineer Lead