kubernetes

Kubernetes

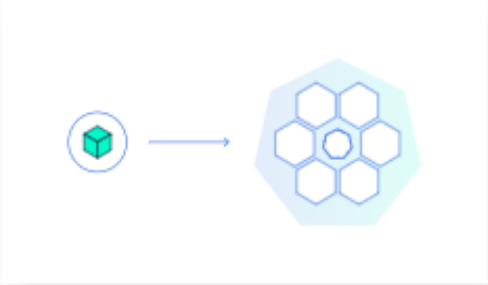
Kubernetes is a production-ready, open source platform designed with Google's accumulated experience in container orchestration. The platform manages, deploys and scales containerized applications.

Kubernetes helps you make sure those containerized applications run where and when you want, and helps them find the resources and tools they need to work.

The Kubernetes service also has advanced capabilities around simplified cluster management, container security and isolation policies, the ability to design your own cluster, and integrated operational tools for consistency in deployment.

Kubernetes Basics Modules

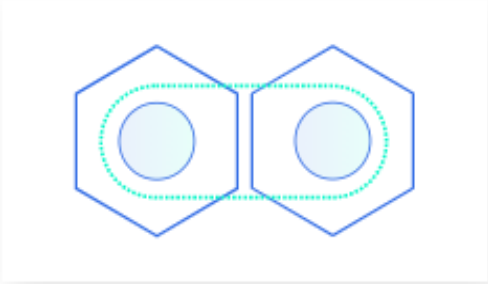
1. Create a Kubernetes cluster



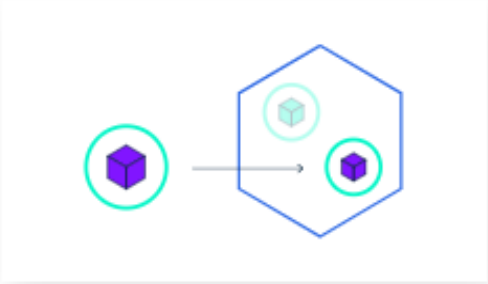
2. Deploy an app

3. Explore your app

4. Expose your app publicly



5. Scale up your app



6. Update your app

Docker Swarm

1. No Auto Scaling
2. Good community
3. Easy to start a cluster
4. Limited to the Docker API's capabilities
5. Does not have as much experience with production deployments at scale

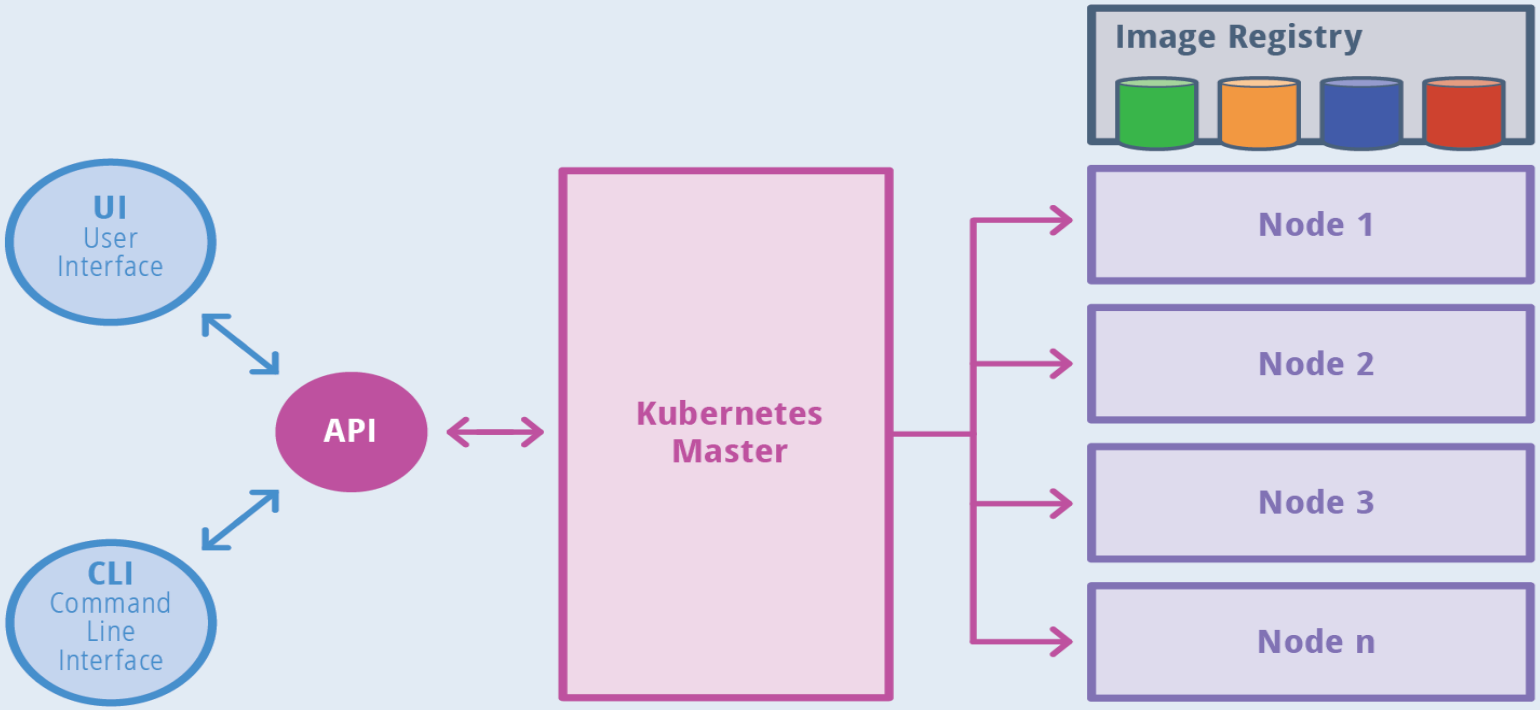
Kubernetes

1. Auto Scaling
2. Great active community
3. Difficult to start a cluster
4. Can overcome constraints of Docker and Docker API
5. Deployed at scale more often among organizations

Features	Kubernetes	Docker Swarm
Installation & Cluster Configuration	Installation is complicated; but once setup, the cluster is very strong	Installation is very simple; but cluster is not very strong
GUI	GUI is the Kubernetes Dashboard	There is no GUI
Scalability	Highly scalable & scales fast	Highly scalable & scales 5x faster than Kubernetes
Auto-Scaling	Kubernetes can do auto-scaling	Docker Swarm cannot do auto-scaling
Load Balancing	Manual intervention needed for load balancing traffic between different containers in different Pods	Docker Swarm does auto load balancing of traffic between containers in the cluster
Rolling Updates & Rollbacks	Can deploy Rolling updates & does automatic Rollbacks	Can deploy Rolling updates, but not automatic Rollbacks
Data Volumes	Can share storage volumes only with other containers in same Pod	Can share storage volumes with any other container
Logging & Monitoring	In-built tools for logging & monitoring	3rd party tools like ELK should be used for logging & monitoring

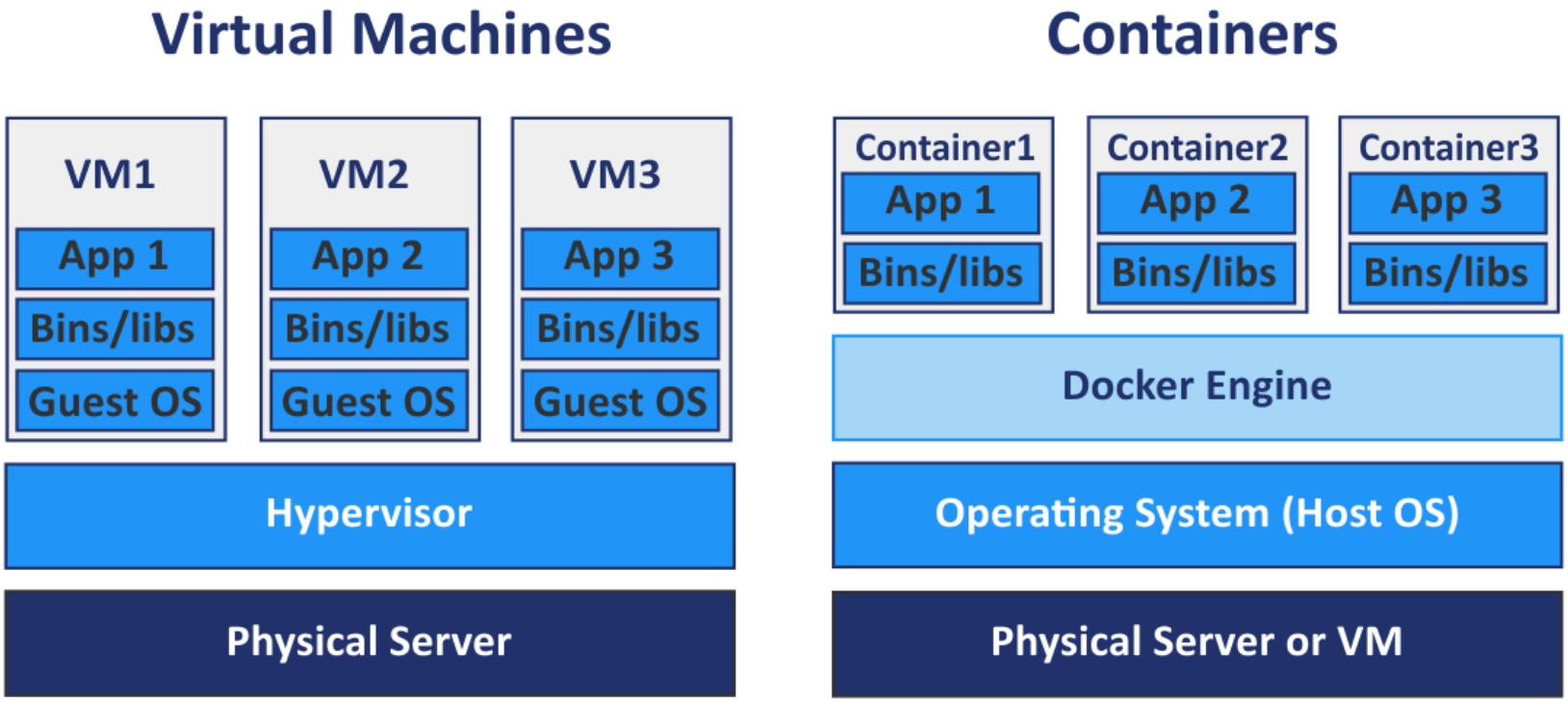
	Docker Swarm	Kubernetes	AWS ECS
Management Tier	A 3-5 node manager tier responds to requests on a single ELB, delegates to N worker nodes	▪ A 3-5 node master tier responds to API calls and web requests, delegates to N minion nodes. ▪ Services can leverage individual ELBs directly, or define NodePorts that are routed through the master tier.	▪ Control plane fully managed by AWS ▪ ALB and ELB route traffic to appropriate containers
Management Tier Failure	Manager nodes must maintain a quorum, but a failed manager will continue to run services	Master nodes must maintain a quorum, and a failed master tier will cause most services to fail	No single point of failure in managed control plane.
Worker Nodes Replaced?	Lost worker nodes automatically replaced	Lost minion nodes automatically replaced	Worker nodes are easily added, replaced, or removed
Updates	Cluster can be upgraded in-place	In-place cluster upgrades still maturing, 3rd-party distributions may differ	Agent upgrades can be performed in-place

Kubernetes Architecture

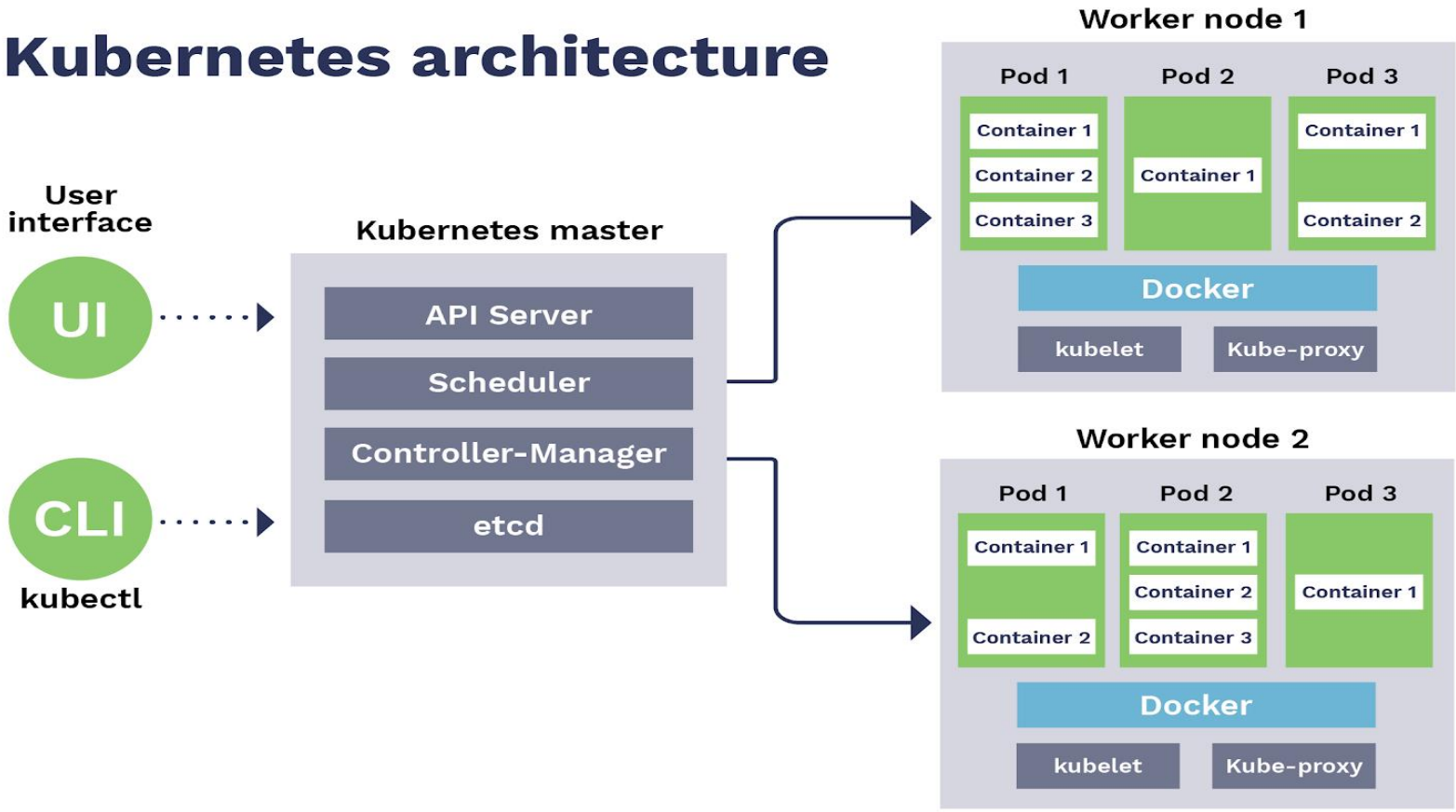


Source: Janakiram MSV

THENEWSTACK


Virtual Machines
VM1
App 1
Bins/libs
Guest OS
VM2
App 2
Bins/libs
Guest OS
VM3
App 3
Bins/libs
Guest OS
Hypervisor
Physical Server
Containers
Container1
App 1
Bins/libs
Container2
App 2
Bins/libs
Container3
App 3
Bins/libs
Docker Engine
Operating System (Host OS)
Physical Server or VM

Kubernetes architecture



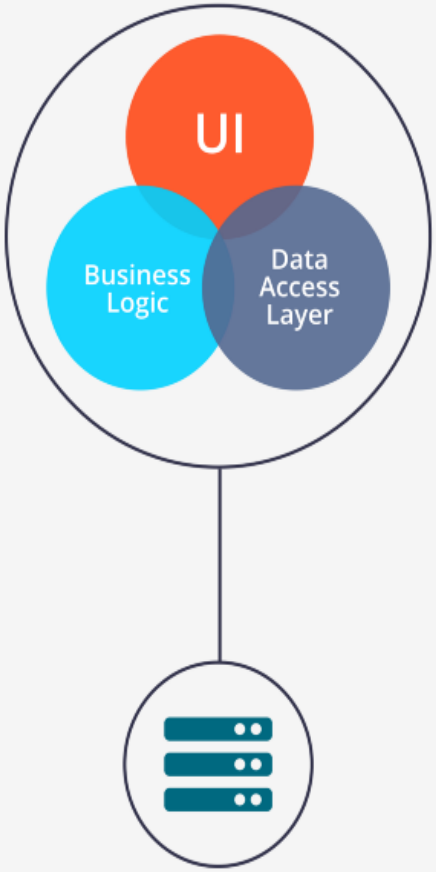
Monolithic architecture vs Microservices architecture

A monolithic application is built as a single unit. Enterprise Applications are built in three parts: a database (consisting of many tables usually in a relational database management system), a client-side user interface (consisting of HTML pages and/or JavaScript running in a browser), and a server-side application. This server-side application will handle HTTP requests, execute some domain-specific logic, retrieve and update data from the database, and populate the HTML views to be sent to the browser. It is a monolith – a single logical executable.

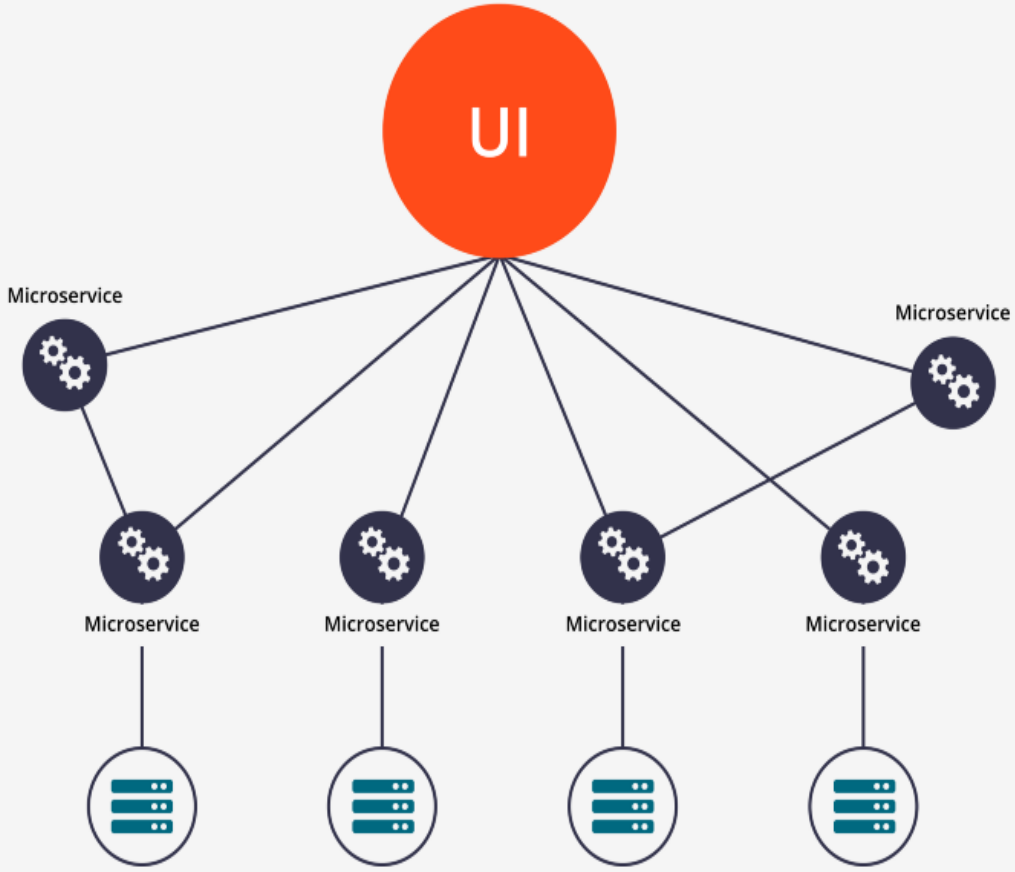
Monolithic architecture vs Microservices architecture

- While a monolithic application is a single unified unit, a **microservices architecture** breaks it down into a collection of smaller independent units. These units carry out every application process as a separate service. So all the services have their own logic and the database as well as perform the specific functions.

- In short, the microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.

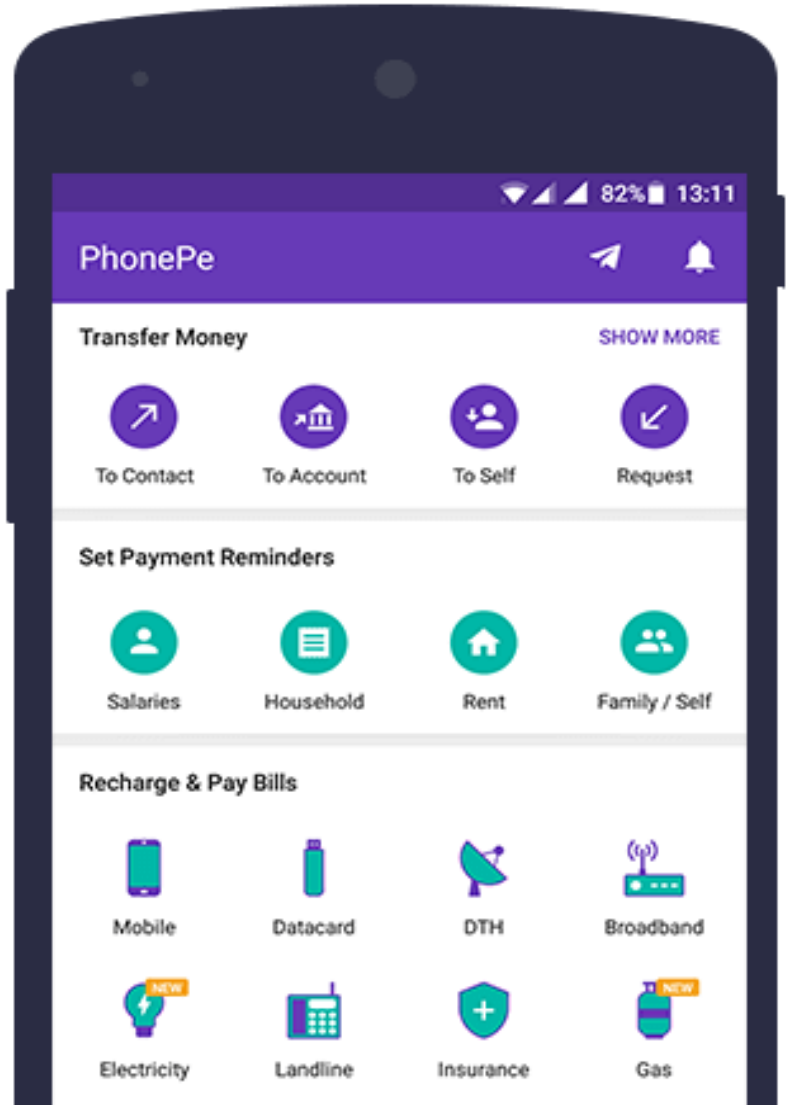


Monolithic Architecture



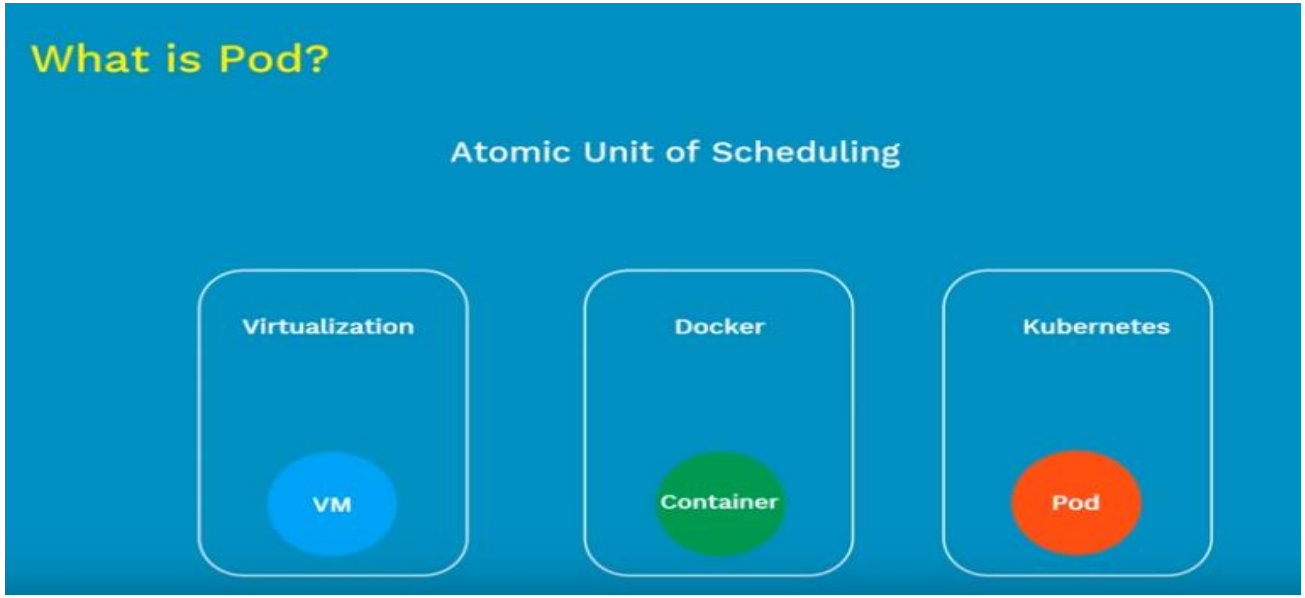
Microservice Architecture


Restaurants
Grocery Stores
Autorickshaws
Petrol Pumps
Milk Booths
Chemists
Paytm
Accepted Here
Parking
Street Food
Electricity
Movies
Paytm
Your Paytm Wallet is here!
Pay
Add Money
Passbook
Request


82% 13:11
PhonePe
Transfer Money
SHOW MORE
To Contact
To Account
To Self
Request
Set Payment Reminders
Salaries
Household
Rent
Family / Self
Recharge & Pay Bills
Mobile
Datacard
DTH
Broadband
NEW
Electricity
Landline
Insurance
NEW
Gas

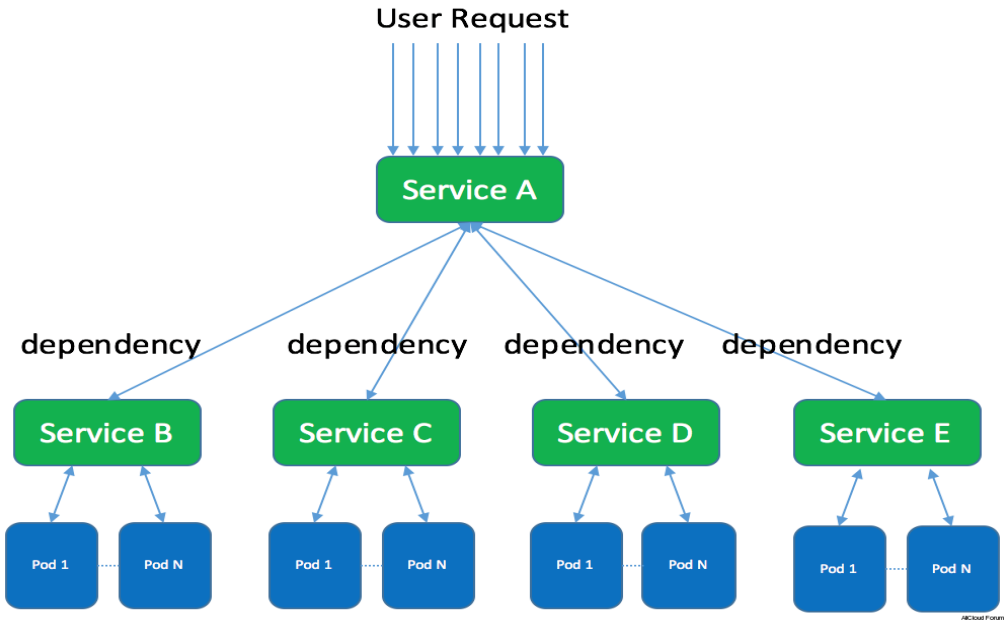
Kubernetes: Key Concepts

- Kubernetes architecture starts with containers. A container is the smallest unit in the k8 platform. It is a packaged software, usually containing one process in a self-contained unit. This unit or container has everything the process needs to run.
- **Pods**
- A pod is a group of containers that share the same networking and storage resources. One of the characteristics of the pod is that when it is deleted, it cannot be restored.



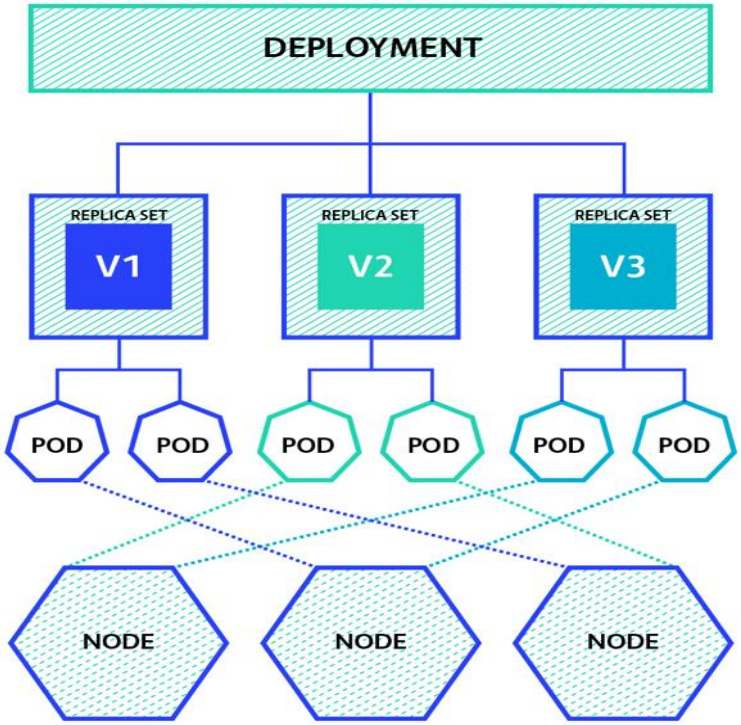
Kubernetes: Key Concepts

- **Nodes**
- A node is a server the pods run on. It represents a single machine in your cluster. It can be a physical or virtual machine.
- **Cluster**
- A cluster consists of a collection of nodes. The cluster distributes the workload to the individual nodes for you. If you remove or add new nodes, the cluster redistributes the load.



Kubernetes: Key Concepts

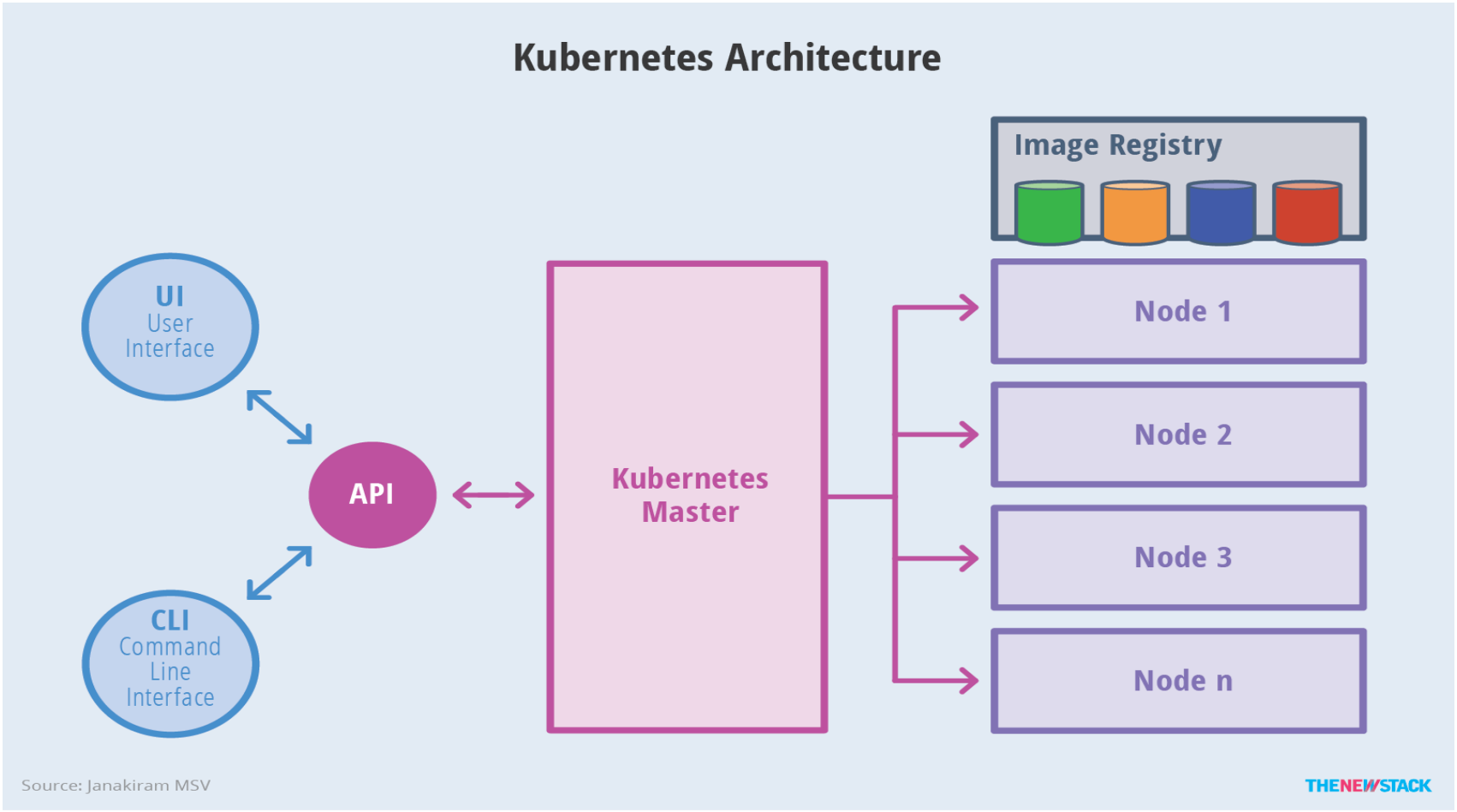
- **Deployment**
- Pods are usually managed in a deployment layer. A deployment is k8s' way to achieve high availability. While pods are mortal, you can set the number of pods you want running. Kubernetes uses deployments to maintain this number of pods.



Kubernetes Architecture Model

- K8s is based on a master-slave architecture model. A Kubernetes **master** node is a unit that controls workloads across the system. The components of a master node include Etcd storage, a controller manager, an API server, and a scheduler.

- **Worker** or slave nodes receive communications from the master node. The master node assigns resources to containers according to the schedule. The responsibilities of master nodes range from handling API requests, to scheduling and running pods in worker nodes. Master nodes also perform monitoring and networking.


Kubernetes Architecture
Image Registry
UI
User
Interface
API
CLI
Command
Line
Interface
Kubernetes
Master
Node 1
Node 2
Node 3
Node n
Source: Janakiram MSV
THENEWSTACK

Kubernetes architecture