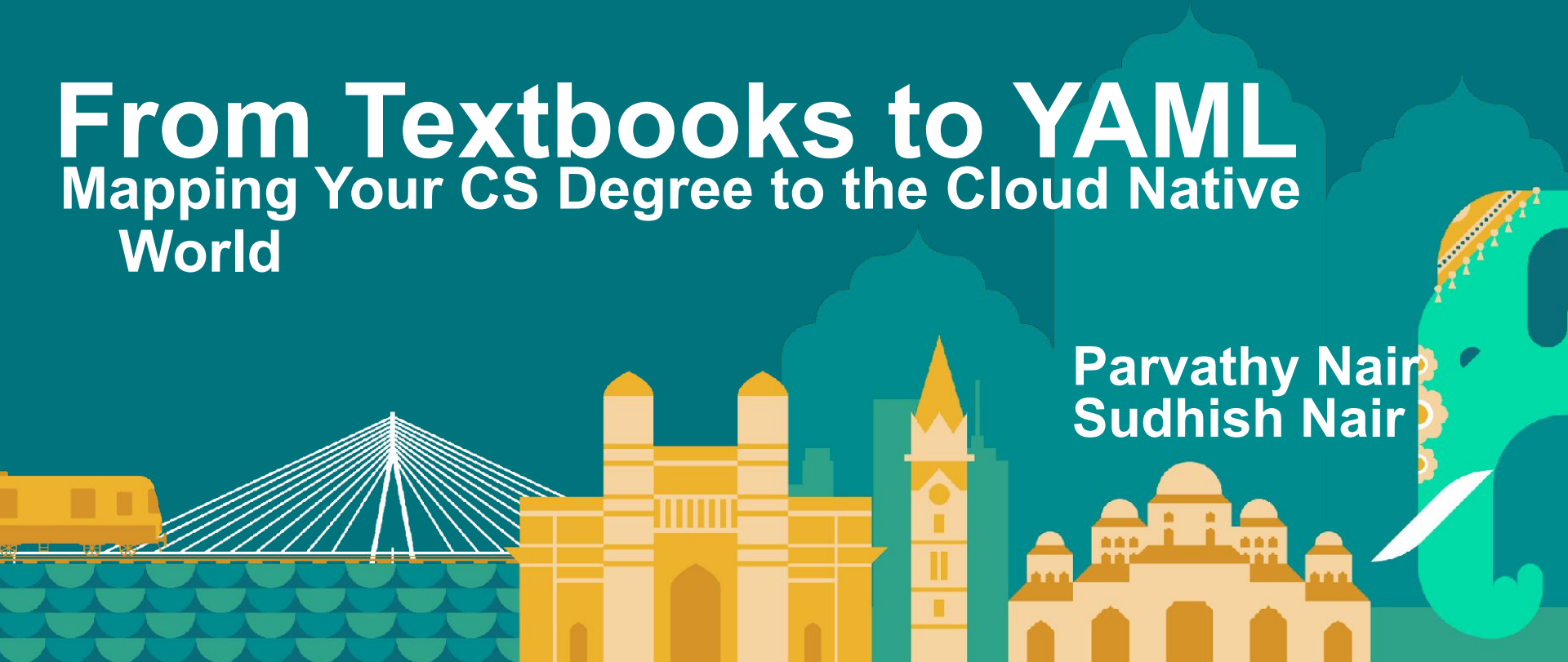
KubeCon CloudNativeCon India 2026

#KubeCon #CloudNativeCon

From Textbooks to YAML

Mapping Your CS Degree to the Cloud Native World

Parvathy Nair
Sudhish Nair



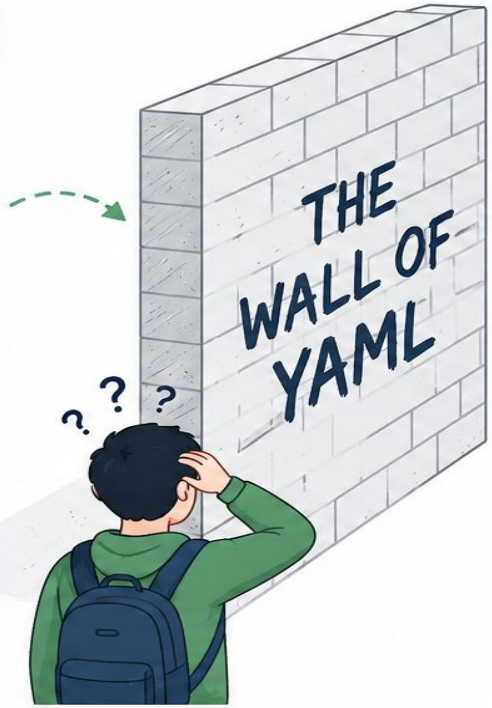
The Wall of YAML

From what we know in college to what we see in the real world

KubeCon CloudNativeCon India 2026

WHAT WE STUDY IN COLLEGE

- Operating Systems ✓
- Networking ✓
- Storage Systems ✓
- Algorithms & Data Structures ✓
- Programming Fundamentals ✓



WHAT WE SEE IN THE REAL WORLD

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: my-app
5    labels:
6      app: my-app
7  spec:
8    containers:
9    - name: app-container
10     image: my-app:1.0
11     ports:
12       - containerPort: 8080
13   restartPolicy: Always
14 .... and so on ...
```

What is apiVersion?

What is kind: Pod?

What is metadata?

Pods? Containers?

So much indentation!

CONFUSION OVERLOAD 😵

GOOD NEWS: You already know the fundamentals.
You just need a **translation** to the Cloud Native world!

Why Is Cloud-Native Everywhere?

The shift to cloud native isn't a trend – it's the new foundation

KubeCon CloudNativeCon India 2026

BY THE NUMBERS

95%+ of organizations use Kubernetes (in some form)

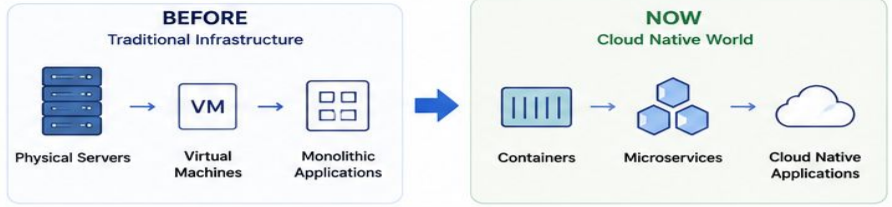
78% of organizations are running containerized applications in production

90% of workloads will be in the cloud by 2027

$67.6B Cloud Native market size by 2027 (CAGR 22.7%)

Sources: CNCF 2023 Survey, Gartner, Statista, MarketsandMarkets

THE INDUSTRY SHIFT

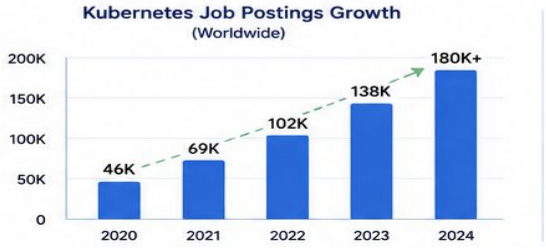


MOST IN-DEMAND CLOUD NATIVE SKILLS

Kubernetes · Containers · CI/CD · DevOps · GitOps · Observability

These skills are among the highest paying and fastest growing across tech companies worldwide.

THE TALENT OPPORTUNITY



Cloud Native skills open doors to roles like:

- Cloud Engineer
- DevOps Engineer
- Platform Engineer
- Site Reliability Engineer
- Solutions Architect

The future is cloud native. The time to build your skills is now.

What is Cloud Native?

Building and running applications that are ready for the cloud era

KubeCon CloudNativeCon India 2026

Cloud Native is an approach to building and running applications that are containerized, dynamically managed, and designed to thrive in modern, distributed environments.

THE CORE PRINCIPLES

Containerized
Package application and its dependencies together.

Dynamically Managed
Automate deployment, scaling, and operations.

Microservices Oriented
Build applications as small, independent services.

Scalable
Scale up or down based on demand instantly.

Resilient
Design for failure and recover automatically.

Automated
Use automation and CI/CD for faster delivery.

CLOUD NATIVE IS NOT JUST ABOUT TECHNOLOGY, IT'S ABOUT A BETTER WAY TO BUILD SOFTWARE

People
Cross-functional teams and collaboration

Process
Agile, DevOps, and continuous improvement

Technology
Containers, Kubernetes, CI/CD, Observability and more

Observability
Monitor, log and trace everything in real-time

Security
Built-in security from code to runtime

Cloud Native = Containers + Automation + Scalability + Resilience
It's not a single product, it's an approach.

THE EVOLUTION

Traditional Applications
- Monolithic
- Tightly Coupled
- Hard to Scale
- Manual Operations

Virtualized Applications
VM VM VM
- Better Isolation
- Higher Overhead
- Slow Provisioning

Containerized Applications
- Lightweight
- Portable
- Faster Deployment

Cloud Native Applications
- Elastic & Scalable
- Resilient
- Automated
- Built for the Cloud

Cloud Native is the foundation.
Kubernetes is one of the key enablers.

What is Kubernetes?

The cloud – native operating systems for your applications

KubeCon CloudNativeCon India 2026

Kubernetes is an open-source platform that **automates deploying, scaling,** and **operating containerized applications** across a cluster of machines.

WHAT KUBERNETES DOES

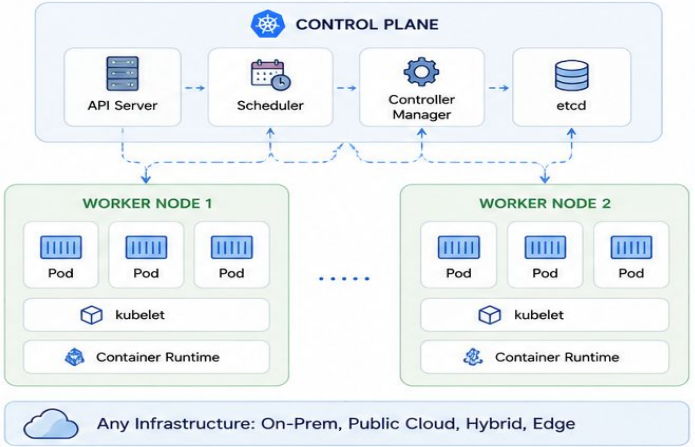
DEPLOY
Deploy applications consistently every time.

SCALE
Scale up or down based on demand automatically.

HEAL
Detect and recover from failures automatically.

MANAGE
Manage applications, resources, and updates declaratively.

KUBERNETES CLUSTER AT A GLANCE



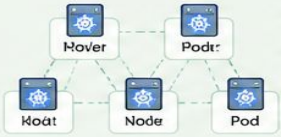
THE BIG ANALOGY

OPERATING SYSTEM
(On a Single Machine)

- Manages Processes
- Manages Memory
- Manages Filesystem
- Manages Devices

=

KUBERNETES
(Across Many Machines)



- Manages Containers (like Processes)
- Manages Storage (like Filesystem)
- Manages Networking (like Network Stack)
- Manages Nodes & Resources (like Hardware)

Kubernetes doesn't run your application.

It runs the platform that runs your application.

Why Do Students Feel Overwhelmed?

It's not that you don't know the concepts. It's that the vocabulary is different.

KubeCon CloudNativeCon India 2026

WHAT STUDENTS LEARN IN COLLEGE	WHAT INDUSTRY USES (CLOUD NATIVE WORLD)
Process A program in execution with its own memory and resources.	**Container** A lightweight, isolated runtime for your application.
Process Group A collection of related processes working together.	**Pod** A group of containers that share network and storage.
Scheduler Decides which process gets CPU time.	**Kubernetes Scheduler** Decides which node runs your pod.
Executable A binary file that the OS can run.	**Container Image** A packaged application with all dependencies.
Machine Physical/virtual machine where OS runs.	**Node** A machine (VM or bare metal) managed by Kubernetes.

THE REALITY

Same fundamentals.
Different names.

OUR GOAL

Translate what you know in Computer Science to what the industry uses in Kubernetes.

You already know the building blocks.
Let's connect the dots.

COMPUTER SCIENCE CONCEPTS → TRANSLATION LAYER → KUBERNETES WORLD

The Goal of This Session

We are not here to learn anything new. We're here to connect what you already know.

KubeCon CloudNativeCon India 2026

OUR JOURNEY TOGETHER

WHAT YOU ALREADY KNOW
(Computer Science Fundamentals)

COMPUTE
Processes, Scheduling, Concurrency, OS

NETWORKING
TCP/IP, Sockets, DNS, Routing, Load Balancing

STORAGE
Disks, Filesystems, Mounting, I/O

TRANSLATION LAYER
(This Session)

We'll map core CS concepts to Kubernetes building blocks.

Different names.
Same fundamentals.

KUBERNETES WORLD
(Cloud Native Reality)

COMPUTE
Containers, Pods, Nodes, Scheduler

NETWORKING
Services, Ingress, CNI, DNS

STORAGE
PV, PVC, CSI, Storage Classes

BY THE END OF THIS SESSION, YOU WILL...

- See Kubernetes through your CS lens.
- Understand how Compute, Networking and Storage work in Kubernetes.
- Feel confident to start your Cloud Native journey.

You don't need to learn everything new.
You just need a new vocabulary for what you already know.

How Compute Works in Operating Systems

From program to running process on the CPU

THE BIG IDEA

The OS takes a program, creates a process, and makes sure it gets CPU time. Many processes share the CPU safely and efficiently.

KEY CONCEPTS

Program
A passive set of instructions on disk

Process
A program in execution with its own memory and resources

CPU Scheduling
The OS decides which process gets CPU time and for how long

Context Switching
The OS saves the state of one process and switches to another

THE JOURNEY OF COMPUTE IN AN OS

1 PROGRAM
You write a program

program.c

Stored on disk

2 PROCESS CREATION
OS creates a process (via fork/exec)

PID: 4242

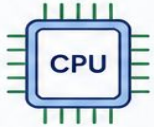
Process has its own memory, PID, and resources

3 CPU SCHEDULING
OS scheduler decides who runs next

Ready Queue

Processes wait their turn in the ready queue

4 RUNNING ON CPU
Process gets CPU time to execute

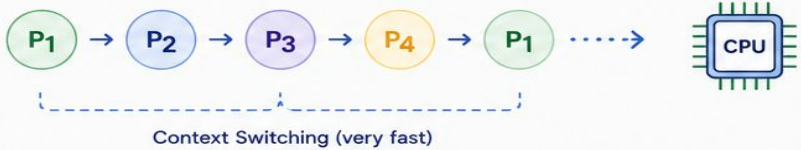


Running

Instructions are executed until time slice ends or process blocks/exits

MANY PROCESSES, ONE CPU

The OS gives each process a small slice of CPU time (time slice) in a very fast cycle.



THE TAKEAWAY

The OS abstracts the hardware and gives the illusion that each process has the CPU to itself, even though they share it.

This is the foundation for how Kubernetes thinks about compute.

KubeCon CloudNativeCon India 2026

How Memory & Storage Work in Operating Systems

From saving data to finding it back

MEMORY: FAST BUT VOLATILE

The OS manages main memory (RAM) for running processes.

1. **Process Requests Memory** — A process asks the OS for memory to use.
2. **OS Allocates Memory** — OS finds free memory and allocates it.
3. **Process Uses Memory** — Process reads/writes data in memory.
4. **OS Protects & Isolates** — OS ensures one process can't access another's memory.

KEY CONCEPTS

- **Virtual Memory** — OS uses disk as an extension of RAM when needed.
- **Protection** — Each process gets its own isolated memory space.
- **Performance** — Memory is fast, but limited and volatile.

STORAGE: PERSISTENT BUT SLOWER

The OS manages data on disks for long-term storage.

1. **Process Requests to Save Data** — A process asks the OS to save data.
2. **OS Writes to Storage** — OS finds space on disk and writes the data.
3. **Data is Stored Persistently** — Data remains even after the process or system restarts.
4. **OS Manages Everything** — OS handles file systems, space management, caching, and more.

KEY CONCEPTS

- **File Systems** — OS organizes data into files and directories.
- **Persistence** — Data survives restarts and failures.
- **Abstraction** — OS hides disk complexity from applications.

CORE IDEAS TO REMEMBER

- **Memory (RAM)** — Fast, limited, and volatile.
- **Storage (Disk)** — Slower, larger, and persistent.
- **The OS** — Manages both efficiently and securely.

Everything you save and everything you run depends on these fundamentals.

KubeCon | CloudNativeCon India 2026

THE TAKEAWAY

The OS gives every process the illusion of having lots of fast, private memory and unlimited, reliable storage — even though the hardware is limited.

MEMORY + STORAGE → OPERATING SYSTEM

How Networking Works in Operating Systems

From sending a message to reaching the right place.

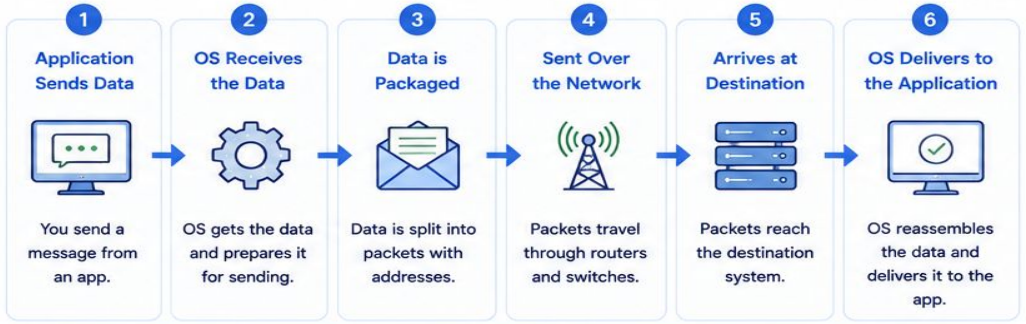
THE BIG IDEA

The OS helps your data travel across networks reliably and securely.

WHAT THE OS DOES

- Chooses the best path (routing)
- Reliable delivery (handling errors, retries)
- Security (firewalls, encryption)
- Efficient communication (sockets, buffering)

THE JOURNEY OF DATA (SENDING A MESSAGE)



The OS handles all the complex steps in between — you just use simple APIs!

KEY CONCEPTS

Socket
An endpoint for communication (IP + Port).

Protocol
Rules that define how data is formatted, sent, and received.

IP Address
Unique identifier for devices on a network.

Firewall
Protects your system by controlling what traffic is allowed.

Buffering
Temporarily stores data to handle speed differences.

THE NETWORKING STACK

Closer to You ↑ Closer to the Hardware ↓

Application Layer	Your apps (e.g., browser, chat) use network services.
Transport Layer	Ensures delivery between applications (TCP / UDP).
Network Layer	Finds the best path (IP routing).
Data Link Layer	Moves data between devices on the same network.
Physical Layer	Sends raw bits over cables, Wi-Fi, or other media.

NETWORK INTERFACES: THE BRIDGES

The OS uses network interfaces to connect to different networks.

Wi-Fi
Connects to wireless networks.

Ethernet
Connects to networks using cables.

Bluetooth
Connects to nearby devices.

Cellular
Connects via mobile data networks.

VPN
Creates secure, private tunnels over networks.

KubeCon CloudNativeCon India 2026

THE TAKEAWAY

Networking is not magic — it's a set of steps the OS handles for you.
You send data. The OS makes sure it reaches the right place, safely and reliably.

How Context Switching Works

Saving here, loading there – that's how the OS judges many things at once.

THE BIG IDEA

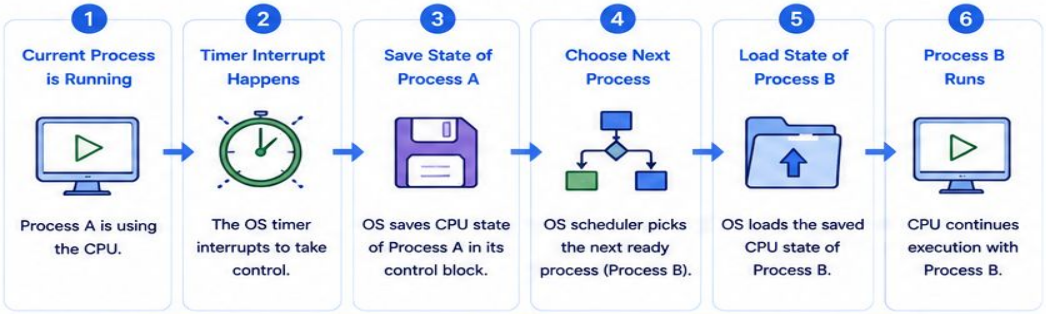
The OS can switch the CPU from one process to another so fast that it feels like everything is running at the same time.

WHAT GETS SAVED?

For each process, the OS saves the CPU's state so it can resume later.

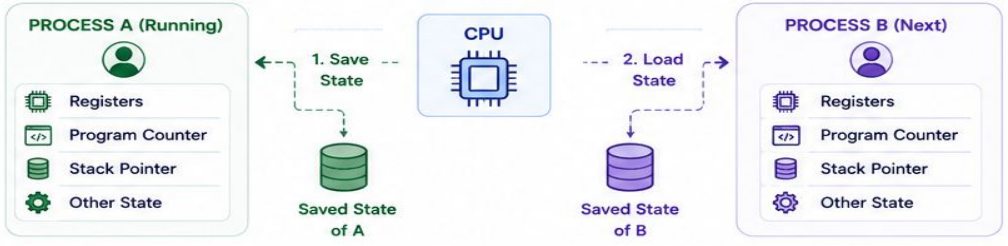
- **Registers** — Values in CPU registers (e.g., general purpose, flags)
- **Program Counter** — The address of the next instruction to execute
- **Stack Pointer** — Points to the top of the process stack
- **Other State** — CPU mode, open files, memory mappings, etc.

THE JOURNEY OF A CONTEXT SWITCH



It happens in microseconds — millions of times a second!

CPU SWITCHES FROM PROCESS A TO PROCESS B



KEY CONCEPTS

- **Preemption** — The OS can interrupt a running process to switch.
- **Timer** — Generates interrupts regularly to keep things fair.
- **Scheduler** — Decides which process runs next.
- **Context Switch** — The act of saving one state and loading another.
- **Overhead** — Context switching takes time, but it enables multitasking.

THE TAKEAWAY

Context switching is how the OS gives each process a turn with the CPU.
Save the state. Load the state. Repeat — fast!

KubeCon CloudNativeCon India 2026

How Kubernetes Brings It All Together

From infrastructure to intelligent orchestration.

THE BIG IDEA

Kubernetes orchestrates containers across a cluster of machines so your applications are reliable, scalable, and resilient.

WHAT KUBERNETES DOES

- Automates deployment and scaling
- Ensures high availability and self-healing
- Efficiently uses resources
- Provides a consistent platform everywhere
- Built-in security and isolation

THE KUBERNETES ARCHITECTURE

CONTROL PLANE (The Brain)

Makes global decisions about the cluster.

- API Server
- Scheduler
- Controller Manager
- etcd (Data Store)

- Exposes the Kubernetes API
- Schedules pods to the right nodes
- Watches the state and makes corrections
- Stores cluster data reliably

WORKER NODES (The Muscle)

Runs your containers.

- Kubelet
- Kube-proxy
- Pods
- Container Runtime

WHY IT MATTERS

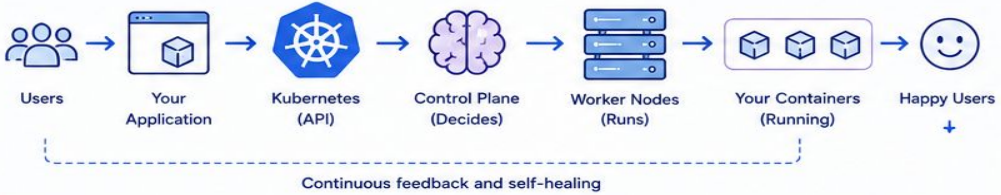
Resilience
Automatically restarts and replaces failed containers.

Scalability
Scale up or down with demand.

Portability
Run anywhere: cloud, on-prem, or hybrid.

Declarative
You declare the desired state, Kubernetes makes it real.

HOW IT ALL CONNECTS



FROM CONCEPTS TO REALITY

- Kubernetes uses the Compute, Networking, and Storage primitives you already know.
- It adds an orchestration layer on top.
- You focus on your app. Kubernetes handles the rest.

That's the power of Cloud Native.

KubeCon CloudNative India 2026

THE TAKEAWAY

Kubernetes is the conductor.
It orchestrates containers using the fundamentals of Compute, Networking, and Storage to deliver **reliable**, **scalable**, and **modern** applications.

Compute + Networking + Storage = Kubernetes

Kubernetes In Action: A Real – World Example

THE BIG IDEA

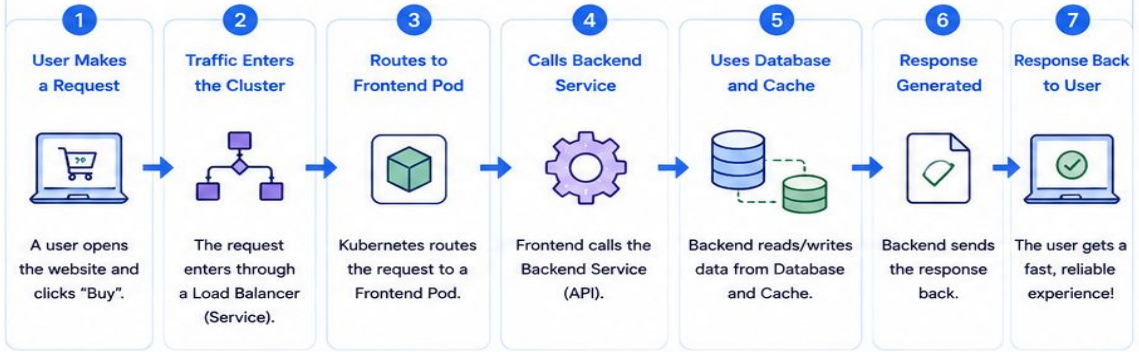
Let's see how Kubernetes runs a real application — step by step.

MEET OUR APP

An e-commerce web app with multiple parts working together.

- Frontend (Web UI)
- Backend (API)
- Database (User Data)
- Cache (Redis)

THE JOURNEY OF A REQUEST



Kubernetes ensures each step is available, scalable, and self-healing.

BUSINESS IMPACT

Reliability
High availability and fewer outages.

Scalability
Handles traffic spikes without breaking.

Speed
Faster deployments and feature releases.

Efficiency
Better resource usage = lower costs.

Focus
Teams focus on building features, not managing infrastructure.

WHAT KUBERNETES DOES BEHIND THE SCENES

Schedules Pods
Places each part of the app on the best available nodes.

Monitors Health
Continuously checks if pods and services are healthy.

Scales Automatically
Adds more pods when traffic increases and removes when it drops.

Self-Heals
Restarts or replaces failed pods instantly.

Updates Safely
Deploys updates with zero downtime.

Manages Resources
Efficiently uses CPU, memory, and other resources.

THE TAKEAWAY

Kubernetes turns a complex, distributed application into a smooth, reliable experience for your users — automatically.

Bringing It All Together – The Big Picture

Every layer. Every step. All working together.

THE BIG IDEA

The OS is the invisible hero that makes everything work together seamlessly.

WHAT WE LEARNED

Hardware Resources
The foundation: CPU, memory, storage, and network.

Operating System
The manager: allocates resources and provides services.

Processes & Apps
Run on top of the OS using the resources.

Networking
Connects systems and moves data.

Kubernetes
Orchestrates containers at scale.

Users
Get reliable, fast, and secure experiences.

THE BIG PICTURE FLOW

1. **User Request** — A user makes a request (e.g., opens an app).
2. **OS Takes Over** — The OS manages CPU, memory, storage, and network.
3. **Process Runs** — The application (process) runs using allocated resources.
4. **Data Moves** — Data is stored, retrieved, and sent across networks.
5. **Kubernetes Orchestrates** — Kubernetes ensures the app stays healthy, scalable, and available.
6. **Response Delivered** — The response comes back to the user — fast, reliable, secure!

Security, monitoring, and optimizations happen at every layer.

THE LAYERS OF MODERN SYSTEMS

- **APPLICATIONS** — What users see and interact with.
- **KUBERNETES** — Orchestrates containers and manages the platform.
- **OPERATING SYSTEM** — Manages resources and provides core services.
- **HARDWARE** — Provides the physical resources.

KEY TAKEAWAYS

It's All Connected
Every component relies on others to get the job done.

The OS is Central
It's the bridge between hardware and everything that runs.

Abstraction Makes It Easy
You don't manage hardware directly — the OS and Kubernetes handle it.

Built for Scale
From a single process to thousands of containers — the principles stay the same.

User Experience is the Goal
All the complexity behind the scenes is to deliver a seamless experience.

KubeCon CloudNativeCon India 2026

THE TAKEAWAY

From hardware to happy users — it's a journey of layers, services, and smart orchestration.
Understanding this big picture helps you build, run, and troubleshoot systems with confidence.

Hardware → OS → Kubernetes → Apps → Users

Key Takeaways: What we learned

From individual components to a unified system – that's the power of understanding

THE BIG IDEA

Modern systems are complex, but they're built from simple ideas working together.

WHAT WE EXPLORED

Operating System
Manages resources and provides services.

Networking
Connects systems and moves data.

Context Switching
Lets the OS multitask efficiently.

Kubernetes
Orchestrates containers at scale.

Real-World Apps
Deliver seamless experiences to users.

THE BIG PICTURE IN ONE VIEW

User	OS	Networking	Kubernetes	Application	User
A user makes a request.	The OS manages resources and runs the application.	Data travels across networks to reach the right place.	Kubernetes keeps containers running and healthy.	The application processes the data and generates a response.	The response comes back to the user.

All working together — seamlessly, securely, and at scale.

FROM BASICS TO REAL-WORLD IMPACT

1. Foundations	2. Core Systems	3. Connectivity	4. Orchestration	5. User Impact
Hardware provides the resources.	The OS manages and abstracts.	Networks connect everything.	Kubernetes runs and scales workloads.	Great experiences, delivered reliably.

Every layer adds value. Together, they create powerful systems.

PRINCIPLES THAT POWER IT ALL

Abstraction
Hide complexity.
Focus on what matters.

Automation
Let systems handle the routine.

Scalability
Grow without breaking.
Be ready for more.

Reliability
Build systems that heal and recover.

Security
Protect every layer, every step.

WHAT YOU CAN DO NEXT

- Keep exploring and stay curious.
- Build small, learn by doing.
- Understand deeply, think in systems, and solve real problems.

You now have the map.
Keep building the journey!

KubeCon CloudNativeCon India 2026

THE TAKEAWAY

You've seen how the pieces fit — from the OS and networking to context switching and Kubernetes.
Understanding the system helps you build, scale, and innovate.

KubeCon CloudNativeCon India 2026

Any questions???


KubeCon
CloudNativeCon
India 2026
Thank You