

# Containerization with Docker

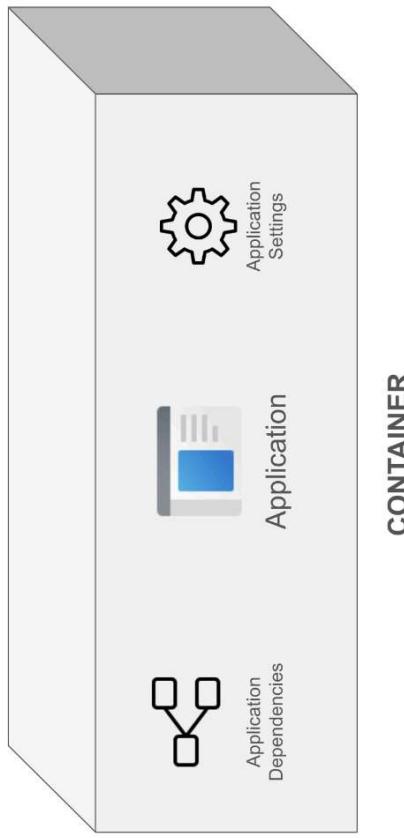
CONTAINERIZATION AND VIRTUALIZATION CONCEPTS



**Julia Ostheimer**  
Freelance AI Consultant

# Recap: Definition of a container

- Containers
  - Isolated environment
  - Includes application and all dependencies



<sup>1</sup> Icons by [icons8.com](https://icons8.com)



# Introducing Docker

- The go-to containerization tool
- Open-source & large user base
- One of the most used and popular tools!



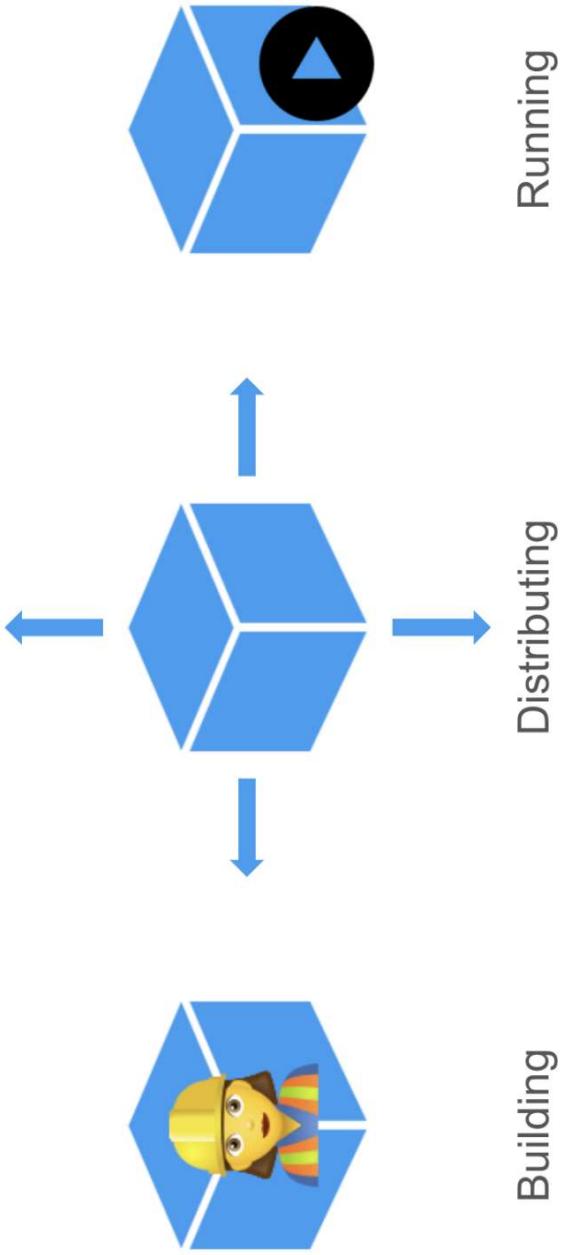
<sup>1</sup> Stack Overflow Developer Survey from 2023 <sup>2</sup> Logo by Docker Inc. from 2024

datacamp

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Introducing Docker

- Managing the lifecycle of containers



<sup>1</sup> Icons by icons8.com

datacamp

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Overview of Docker components

- Most important Docker components:
  - **Docker Desktop**
  - **Docker Engine**
    - Docker Client
    - Docker Daemon
  - **Docker Objects**
    - Docker Images
    - Docker Containers
  - **Docker Registries**

# Installing Docker via Docker Desktop

The screenshot shows the Docker Desktop application interface. At the top, there's a navigation bar with icons for Docker Desktop, Update to latest, Docker Compose, Docker Swarm, Docker Machine, Docker Cloud, Docker Images, Docker Volumes, Docker Env, Docker Scout, Docker Center, and Docker Help. Below the navigation bar is a search bar with placeholder text "Search for images, containers, volumes, extensions and more...".

The main area is titled "Containers" and displays a list of running containers. There are two filter buttons: "Only show running containers" (which is selected) and "Only show stopped containers". A "Delete" button is located at the top right of the container list.

Each container entry includes a thumbnail icon, the container name, the image name, status (e.g., Exited or Running), CPU usage (CPU (%)), port mapping (Port(s)), and the last start time. A "Actions" column contains a dropdown menu with options like "Edit", "Logs", "Attach", "Copy", "Delete", and "Recreate".

On the left side of the main content area, there are sections for "Container memory usage" (with a note: "No containers are running."), "Container GPU usage" (with a note: "No containers are running."), and "Container network usage" (with a note: "No containers are running."). There are also links for "Search", "Give feedback", and "Sign in".

At the bottom right of the container list, it says "Selected 1 of 5".

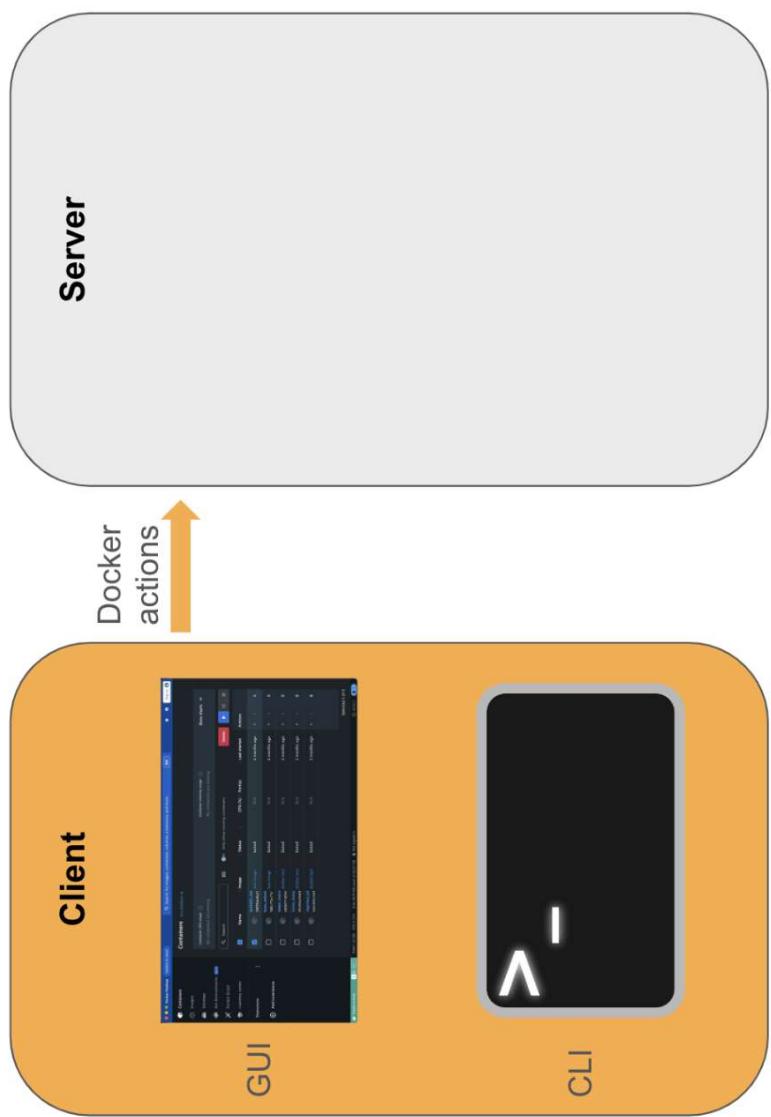
<sup>1</sup> Screenshot from Docker Desktop Mac application



## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Client-server architecture of Docker Engine

## DOCKER ARCHITECTURE



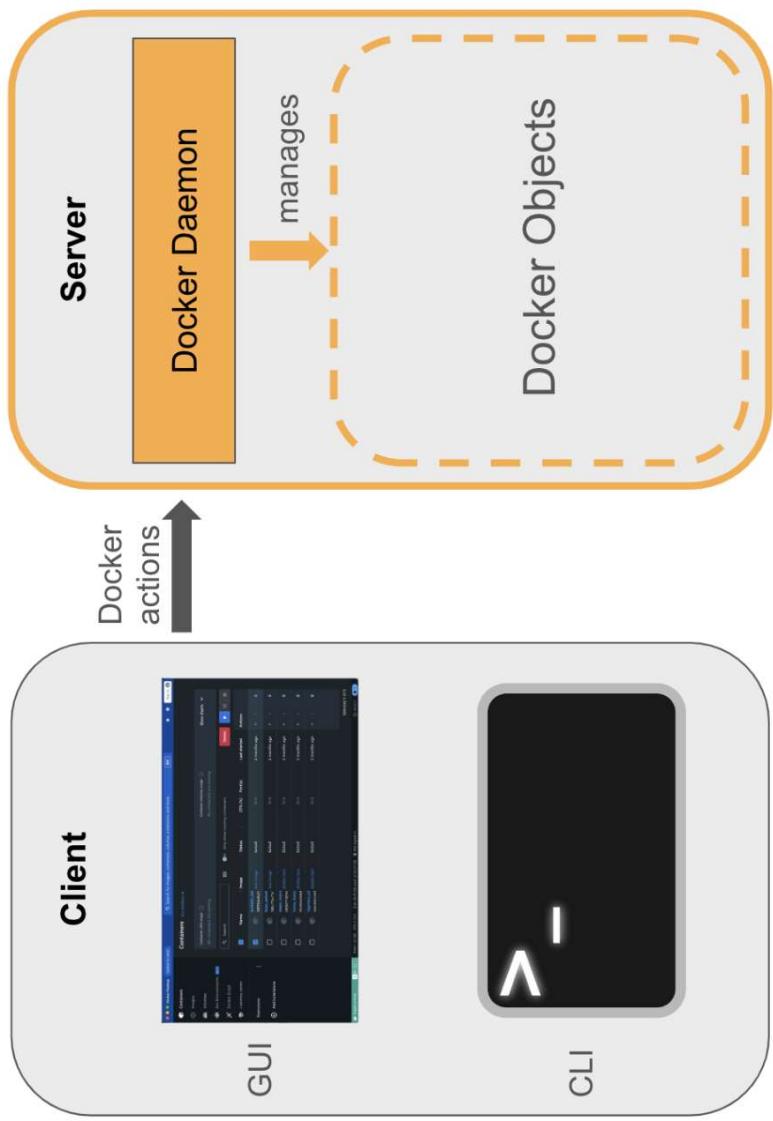
<sup>1</sup> Icons by [icons8.com](https://icons8.com)



CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Client-server architecture of Docker Engine

## DOCKER ARCHITECTURE

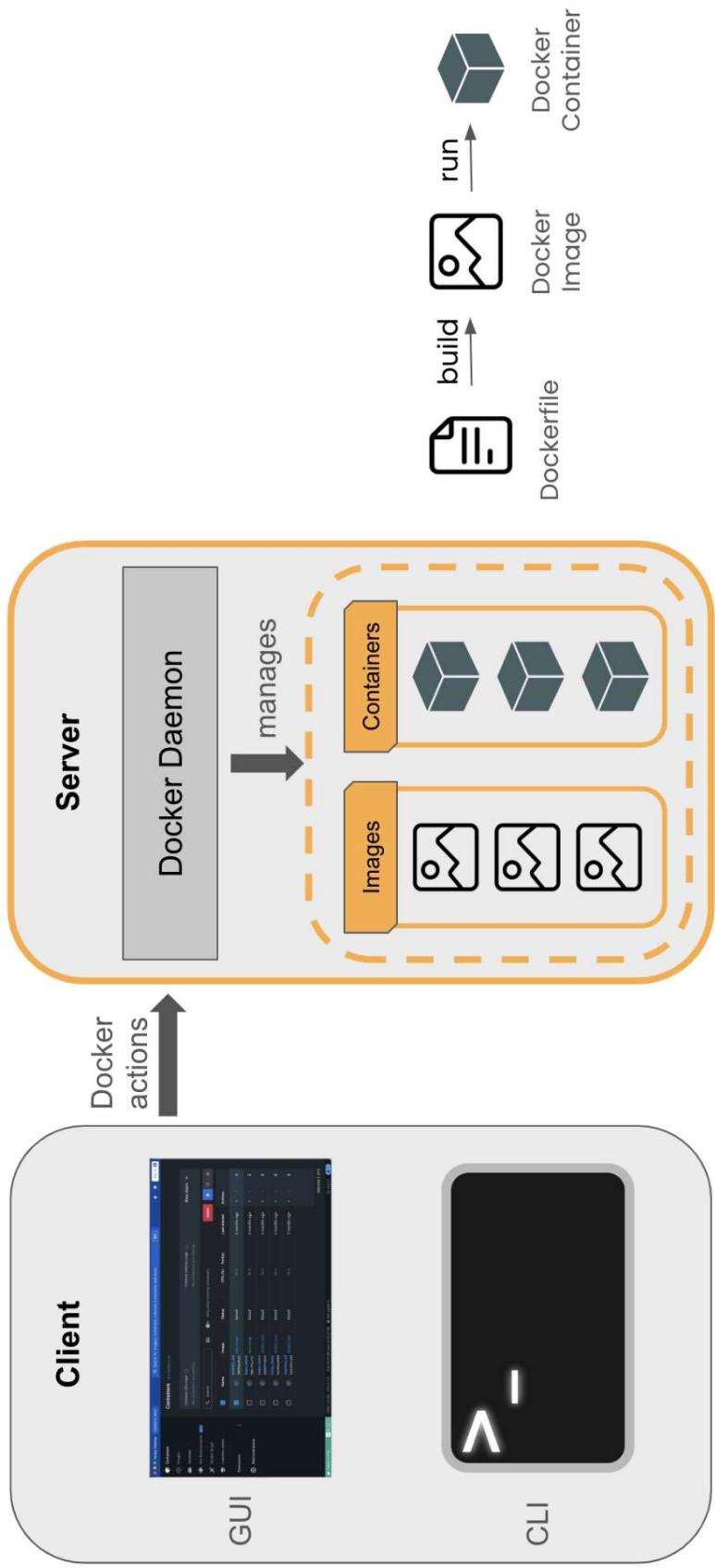


<sup>1</sup> Icons by icons8.com



# Overview of Docker objects

## DOCKER ARCHITECTURE



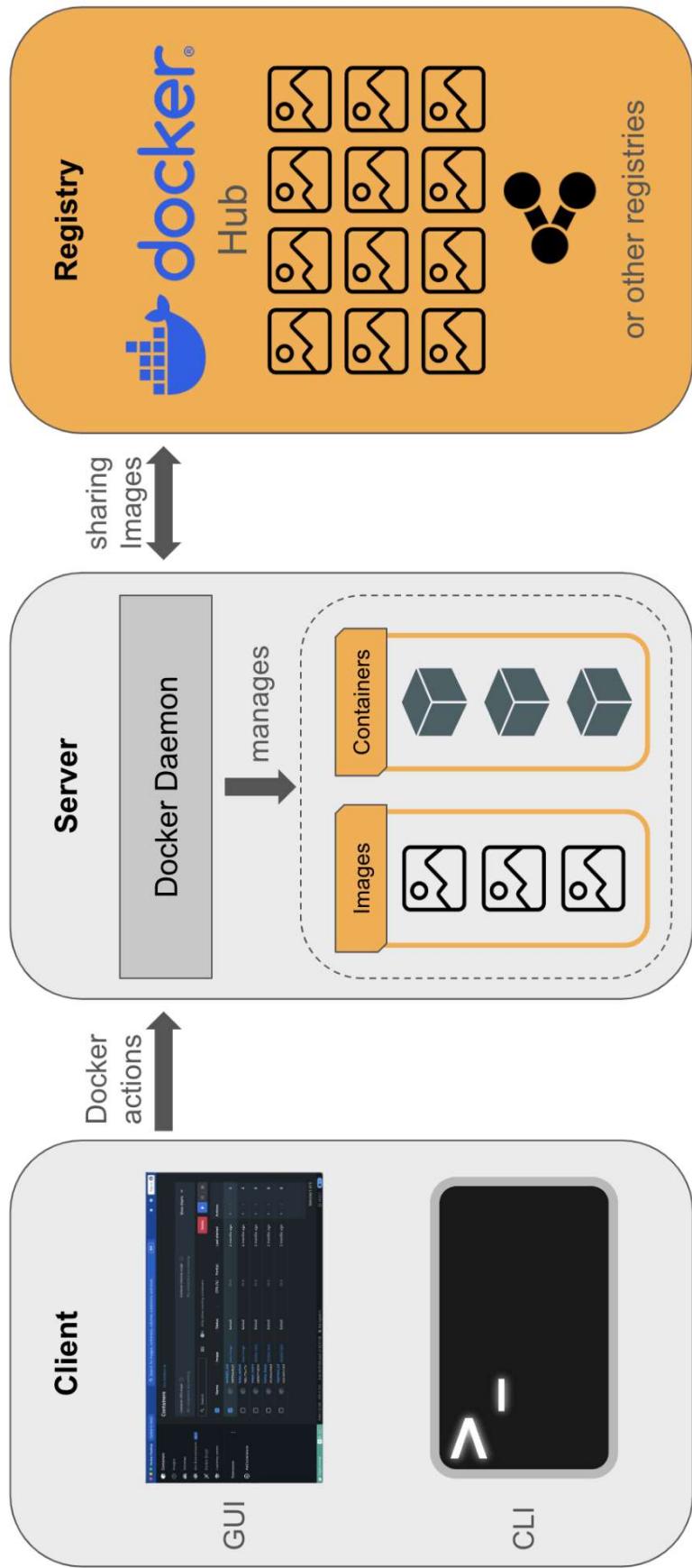
<sup>1</sup> Icons by [icons8.com](https://icons8.com)



## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Sharing containers via registries

## DOCKER ARCHITECTURE



<sup>1</sup> Icons by icons8.com



## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Let's practice!

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Container orchestration

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS



**Julia Ostheimer**  
Freelance AI Consultant

# Definition of container orchestration

- **Orchestration:**
  - The automated management of multiple components
- **Orchestrator:**
  - The tool used for orchestration
- **Container Orchestration:**
  - Orchestration of containers

# Purpose of container orchestration

- Simplifies management of many containers
- Ensuring that multiple containers interact **effectively and efficiently**

# Declarative programming in container orchestration

- Declarative programming:
  - Defining the desired output instead of describing the steps to reach it



<sup>1</sup> Icons by Icons8.com

# Benefits of container orchestration

- Easy scaling of containers
  - Horizontal scaling: Adding/Removing containers
  - Vertical scaling: In-/Decreasing computing resources of specific containers
- Automation of operations
  - Time savings
  - Improved developer productivity
  - Cost savings
- Better performance of application

# Applications of container orchestration

- Microservices architecture
- Application scaling
- Automation of pipelines

# Container orchestration tools



Swarm mode



HashiCorp  
**Nomad**



OpenShift



**kubernetes**

<sup>1</sup> Logos by Docker Inc., Red Hat Inc., HashiCorp Inc., and The Linux Foundation

# Let's practice!

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Container orchestration with Kubernetes

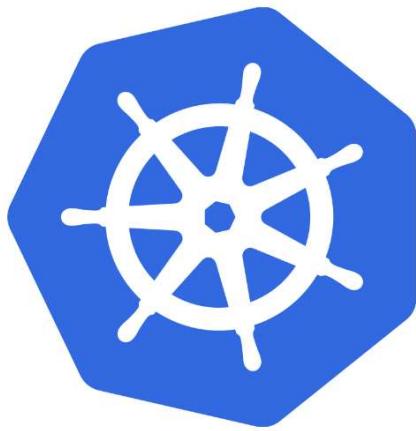
CONTAINERIZATION AND VIRTUALIZATION CONCEPTS



**Julia Ostheimer**  
Freelance AI Consultant

# Introducing Kubernetes

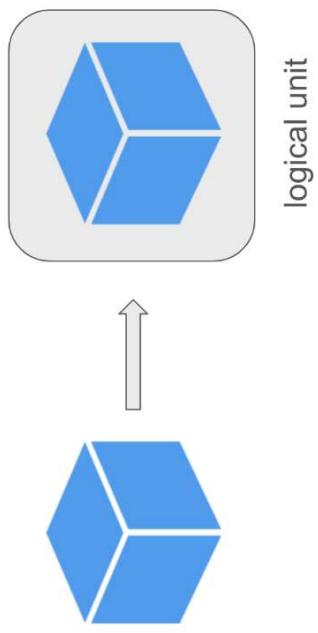
- Abbreviation: K8s
- Developed by Google, open-sourced in 2014
- 96% of organizations use/evaluate using Kubernetes



<sup>1</sup> Cloud Native Computing Foundation (CNCF) Annual Survey in 2022<sup>2</sup> Logo by The Linux Foundation

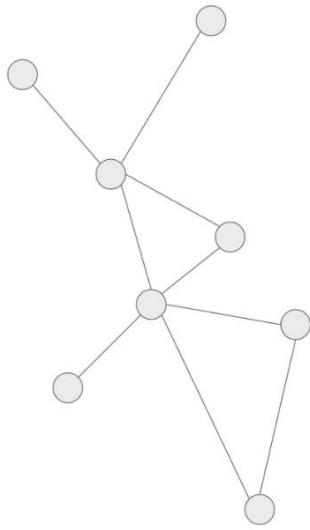
# Introducing Kubernetes

- Grouping containers into logical units



logical unit

- Distributed system



<sup>1</sup> Icons by Icons8.com

datacamp

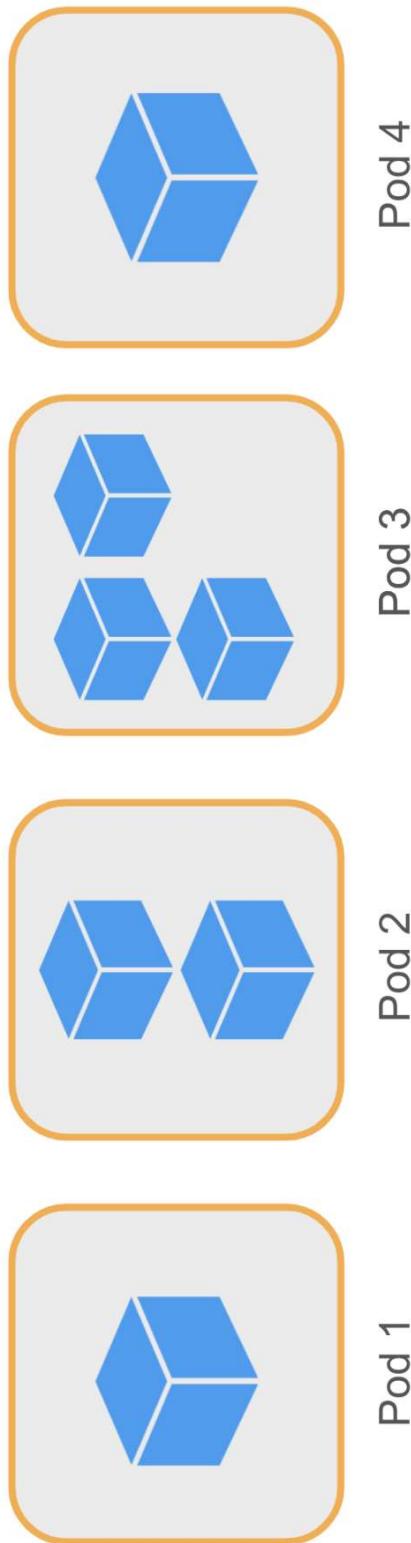
## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Overview of Kubernetes components

- Most important Kubernetes components:

- Pods
- Nodes
- Control Plane
- Cluster

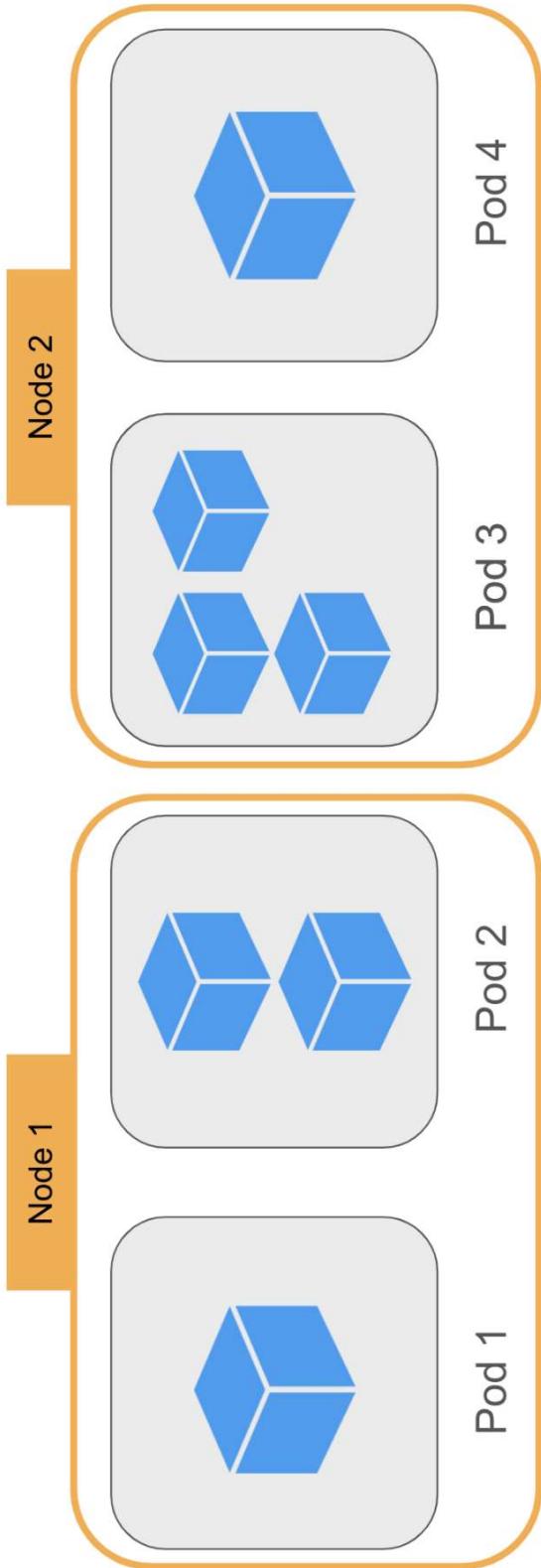
# Pods as smallest deployable unit



<sup>1</sup> Icons by Icons8.com



# Nodes as smallest hardware unit

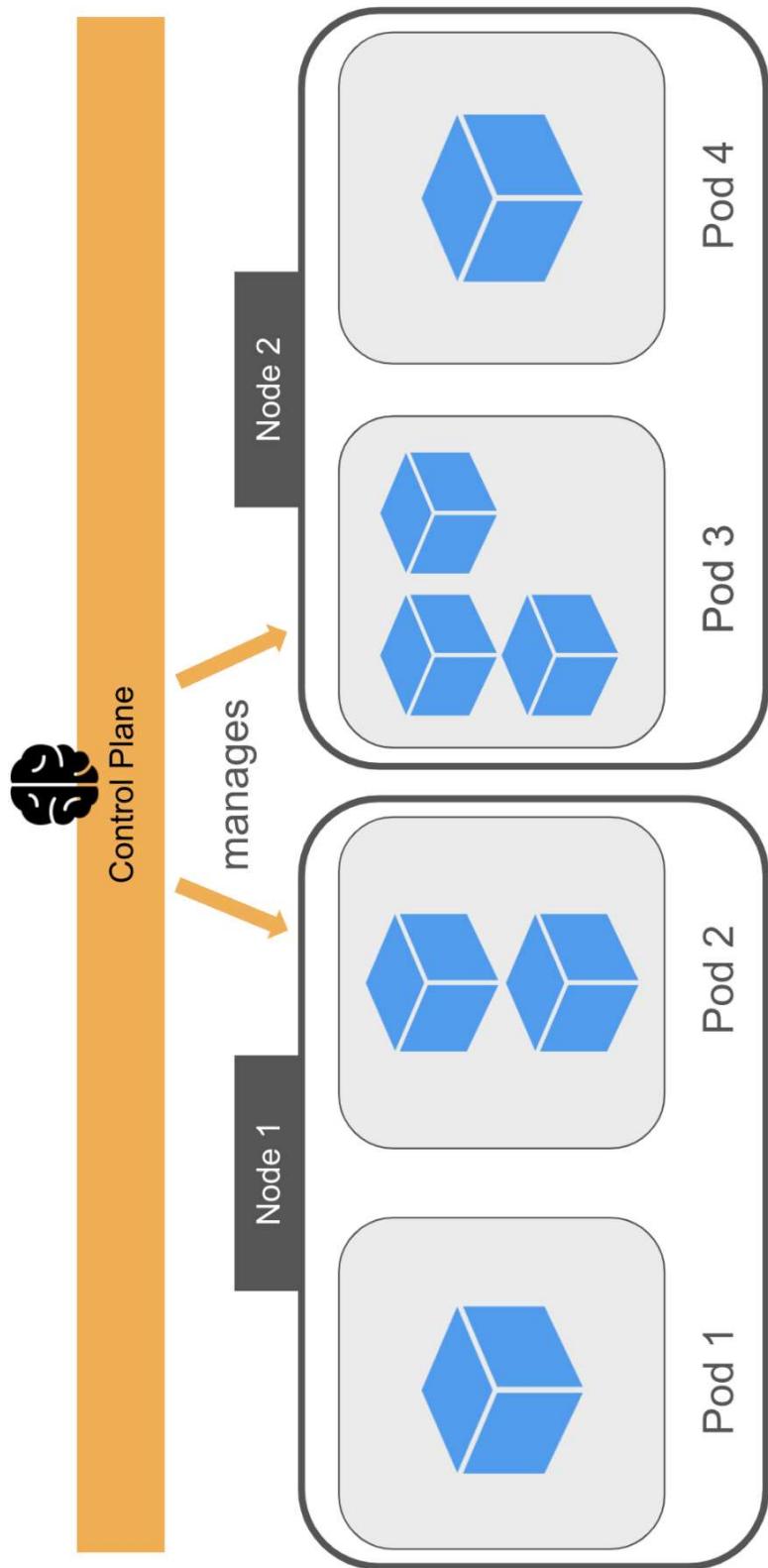


<sup>1</sup> Icons by Icons8.com



## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

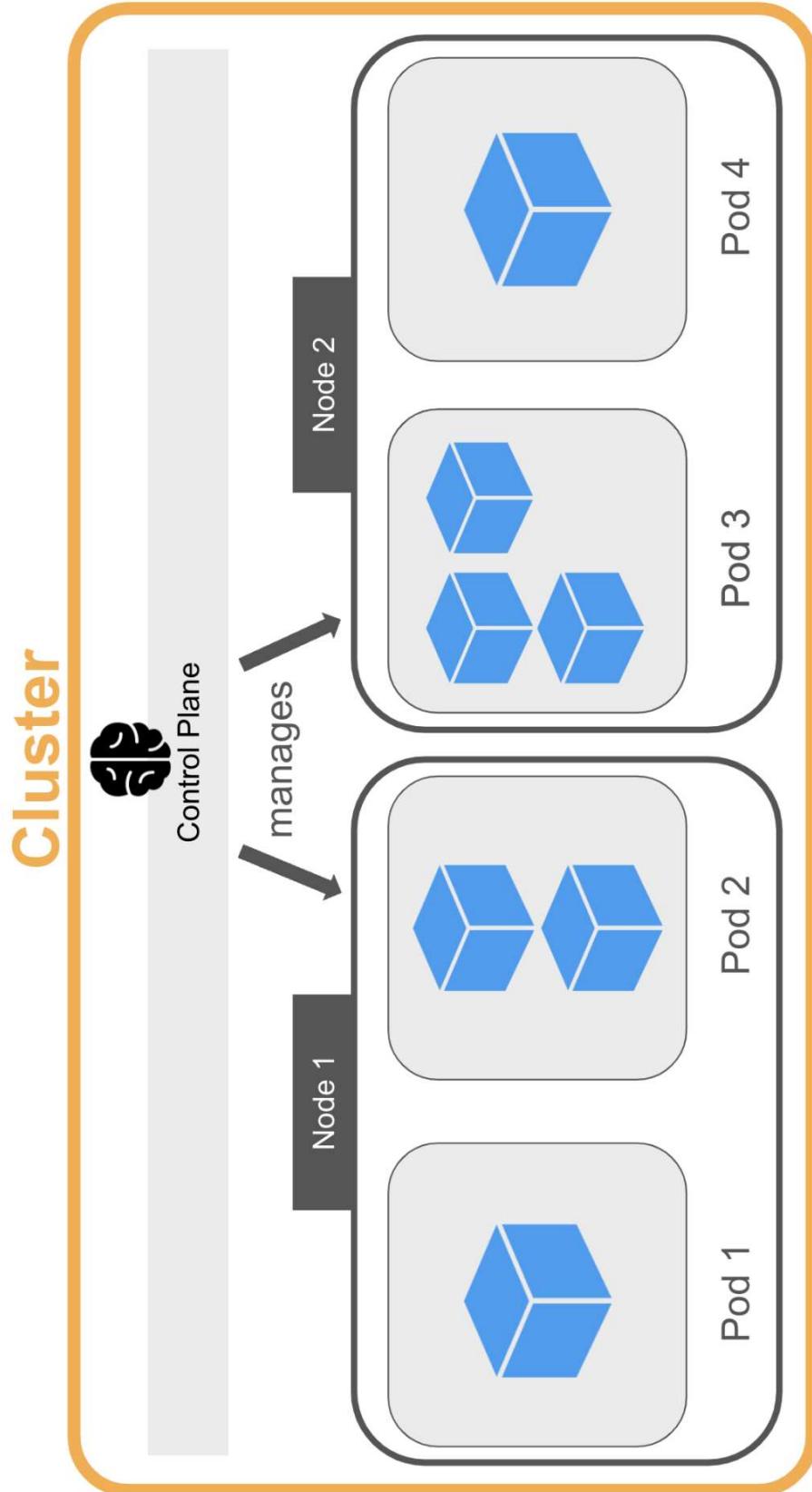
# Node management via control plane



<sup>1</sup> Icons by Icons8.com



# Grouping nodes in a cluster

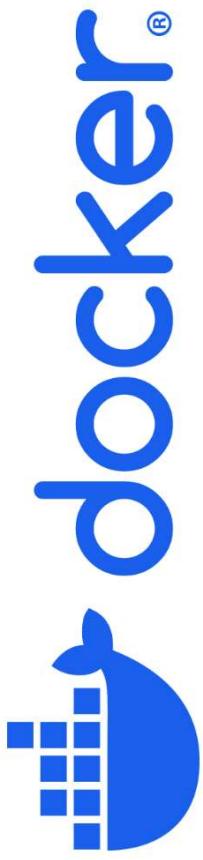


<sup>1</sup> Icons by Icons8.com



# Docker and Kubernetes

- Docker: Dealing with **one or few containers**



- Kubernetes: Dealing with **many containers**



# Let's practice!

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

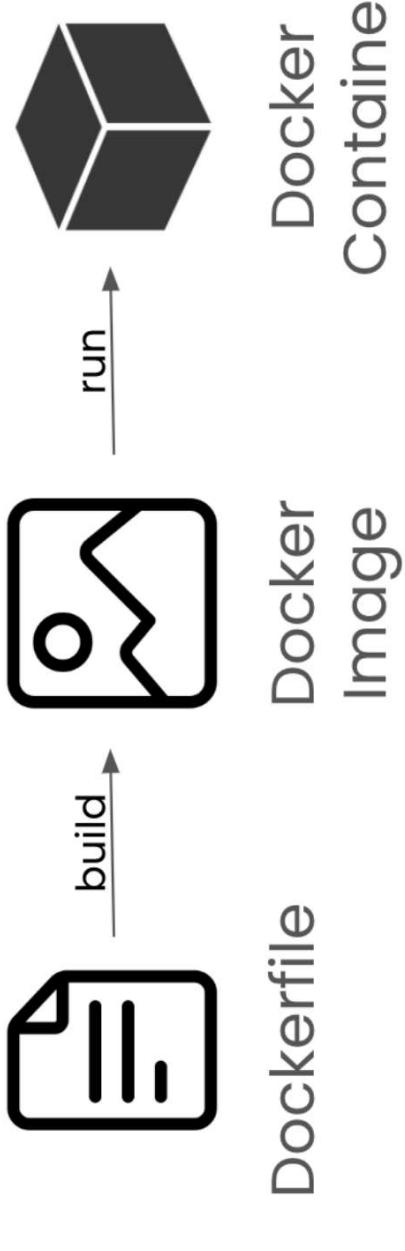
# Reading Dockerfiles and running containers

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS



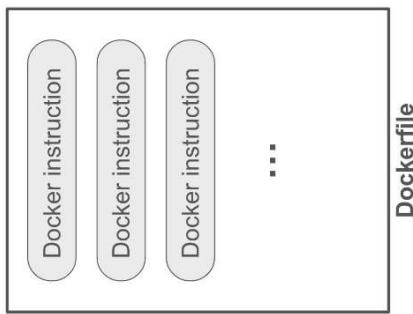
**Julia Ostheimer**  
Freelance AI Consultant

# Recap of Docker terms



# Docker instructions vs. Docker commands

- Docker instructions detail how to build a Docker image



- Docker commands: Commands via Command Line Interface (CLI)



# Format of a Dockerfile

comment

```
# Define the image on which to build
FROM python:3.10
```

...

# Format of a Dockerfile

```
# Define the image on which to build
FROM python:3.10
    ↪ Docker instruction
...

```



# Format of a Dockerfile

```
# Define the image on which to build
FROM python:3.10
...
... COMMAND
```



# Format of a Dockerfile

```
# Define the image on which to build
FROM python:3.10
...
...  
argument
```



# Sequential order in Dockerfiles

- Execution in sequential order
- Start of a Dockerfile:
  - Metadata
  - Comments
  - Arguments
  - `FROM` instruction

# Overview of Docker instructions

- Important Docker instructions

- FROM
- COPY
- RUN
- ENTRYPOINT



# FROM instruction

- Specifies an existing Docker image
- Defines the image we are building on
  - "Starting point"

**Syntax:**

```
FROM <name_of_image>
```

**Example:**

```
# Define the image on which to build  
FROM python:3.10
```



# COPY instruction

- Copies files or directories
  - From source (<source>) to destination (<destination>)
  - Files that are needed in following Docker instructions

**Syntax:**

```
COPY <source> <destination>
```

**Example:**

```
# Copy files/folders to the main folder of the container  
COPY . .
```

# RUN instruction

- Runs a command within a container
  - Can be any command that could be run in a CLI

## Syntax:

```
RUN <command>
```

## Example:

```
# Install the application's dependencies  
RUN pip install -r requirements.txt
```



# ENTRYPOINT instruction

- Defines the container's default behavior
  - Specifies command to run at initiation
  - The primary purpose of the container

## Syntax:

```
ENTRYPOINT ["command", "argument"]
```

## Example:

```
# Run the script when the container starts  
ENTRYPOINT ["python", "hello_world.py"]
```

# Assembling instructions to Dockerfile

```
# Define the image on which to build  
FROM python:3.10
```



# Assembling instructions to Dockerfile

```
# Define the image on which to build
FROM python:3.10

# Copy files/folders to the main folder of the container
COPY . .
```



# Assembling instructions to Dockerfile

```
# Define the image on which to build
FROM python:3.10

# Copy files/folders to the main folder of the container
COPY . .

# Install the application's dependencies
RUN pip install -r requirements.txt
```

# Assembling instructions to Dockerfile

```
# Define the image on which to build
FROM python:3.10

# Copy files/folders to the main folder of the container
COPY . .

# Install the application's dependencies
RUN pip install -r requirements.txt

# Run the script when the container starts
ENTRYPOINT ["python", "hello_world.py"]
```

# Docker build command

- Builds Docker image from a Dockerfile
  - Dockerfile needs to be located in build's context (<context>)
  - Executed as command with Docker client via CLI

Syntax:

```
docker build <context>
```

# Docker run command

- Creates and runs Docker container from Docker image
  - Docker image needs to be specified as argument (<name\_of\_image>)
  - Executed as command with Docker client via CLI

Syntax:

```
docker run <name_of_image>
```

# Let's practice!

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Wrap-up

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS



**Julia Ostheimer**  
Freelance AI Consultant



# Recap of course goals

- Chapter 1
  - Define virtualization
  - Define containerization
  - Comparing containerization and virtualization
- Chapter 2
  - Explain containerization with Docker
    - Define container orchestration
    - Explain container orchestration with Kubernetes
    - Hands-on with Docker

# Recap: Virtualization vs. containerization

## Virtualization

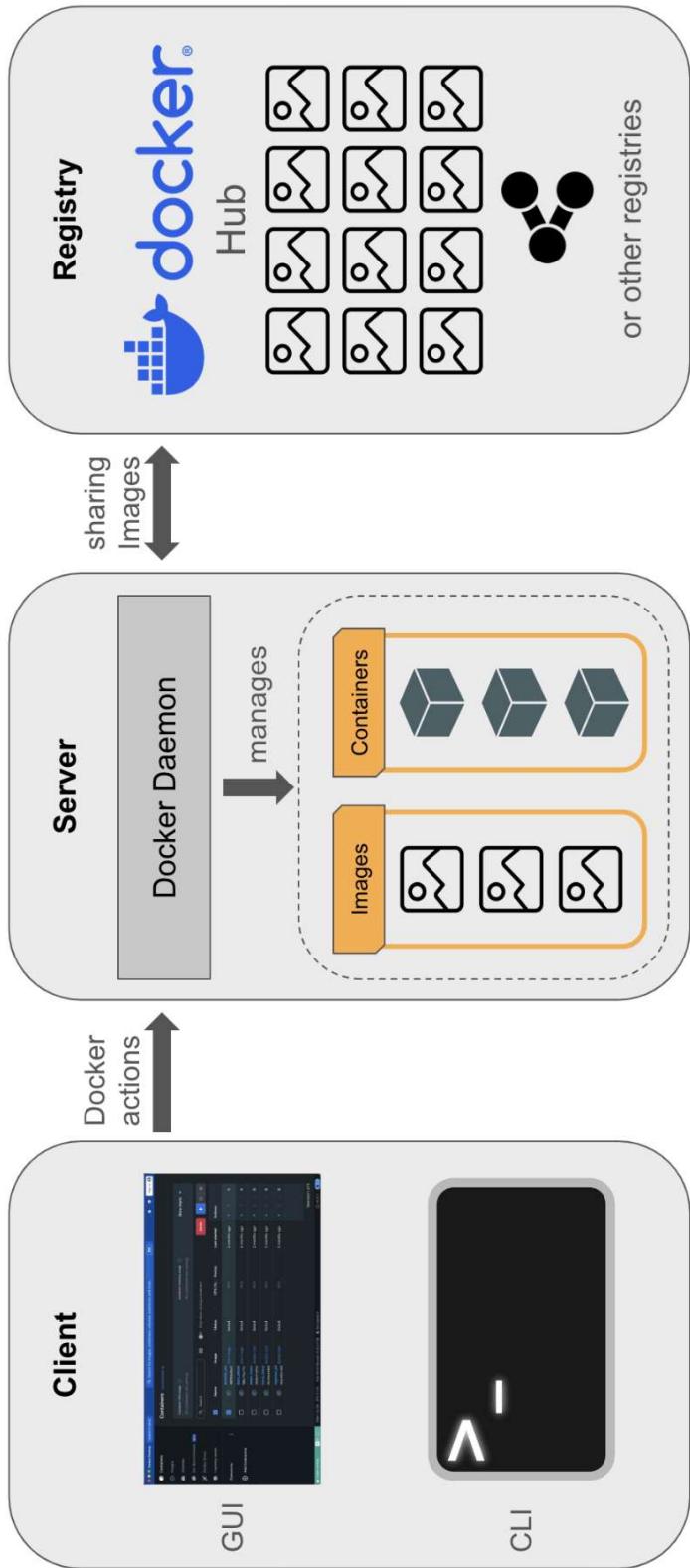
- Creates a virtual version of a computing resource
- Full virtualization
- VM: Simulated computer system inside another computer

## Containerization

- Packages application and dependencies into isolated environment
- OS-level virtualization
- Container: Isolated application environment

# Docker architecture

## DOCKER ARCHITECTURE

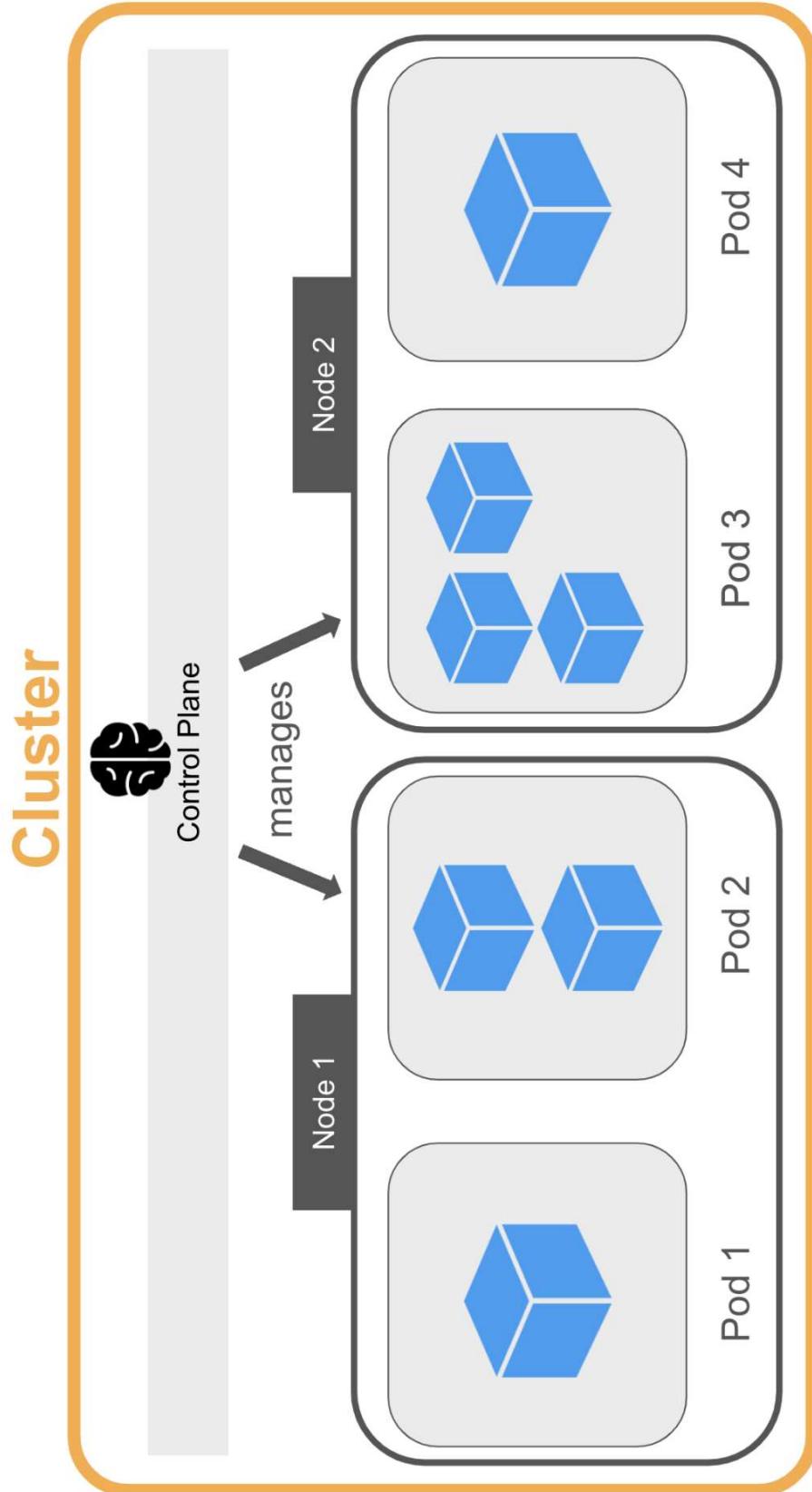


<sup>1</sup> Icons by Icons8.com



## CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Kubernetes architecture



<sup>1</sup> Icons by Icons8.com

datacamp

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Docker instructions and commands

Docker instruction	Description
FROM	Defines the image to build on.
COPY	Copies files or directories into the container.
RUN	Runs a command inside the container.
ENTRYPOINT	Defines the default behavior of the container.

Docker command	Description
docker build <context>	Builds a Docker image based on Dockerfile.
docker run <name_of_image>	Runs a Docker container based on Docker image.

# Hungry for more?

- Understanding computing in the cloud

INTERACTIVE COURSE

## Understanding Cloud Computing

[Start Course](#) [Bookmark](#)

- Dealing with CLI

INTERACTIVE COURSE

## Introduction to Shell

[Start Course](#) [Bookmark](#)



CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

# Hungry for more?

- Continue learning about Docker & Kubernetes!

1. DONE Introduction to Containerization and Virtualization
2. UPCOMING Introduction to Docker
3. Intermediate Docker
4. Introduction to Kubernetes

# Congratulations!

CONTAINERIZATION AND VIRTUALIZATION CONCEPTS

