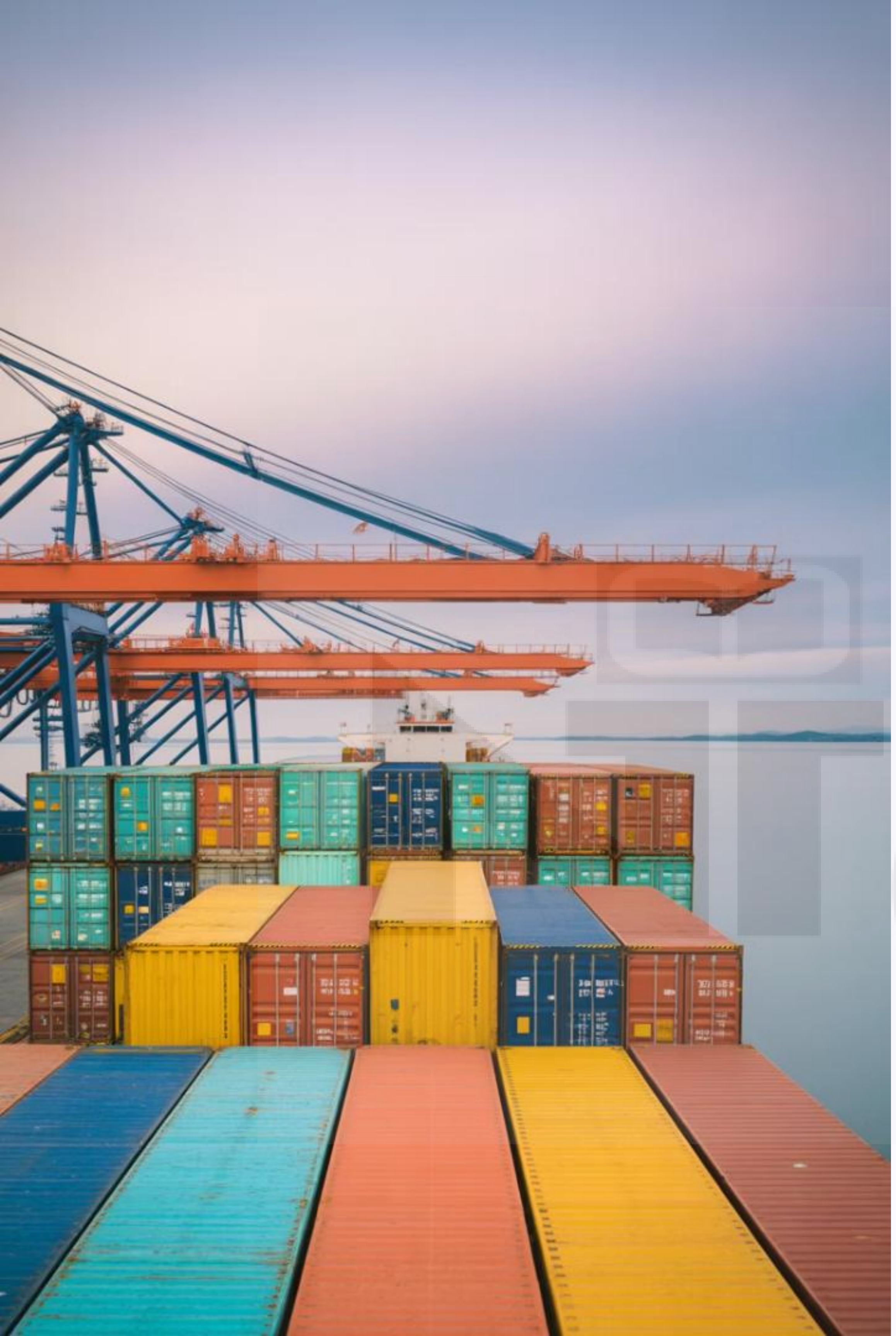


Getting Started with Docker: Simplifying Development with Containers

A beginner's guide to understanding Docker

by mukesh kumar

Our Journey Today

Why Docker matters

Solving dev problems

Core concepts

Images, containers, Dockerfiles

Practical examples

Commands and workflows

Getting started

Your Docker journey begins

The Problem We're Solving

"Works on my machine"

Code runs locally but fails elsewhere

Setup headaches

Hours wasted configuring environments

Dependency conflicts

Different versions breaking your app

Inconsistent environments

Dev, test, and production mismatches

What is Docker?

Platform

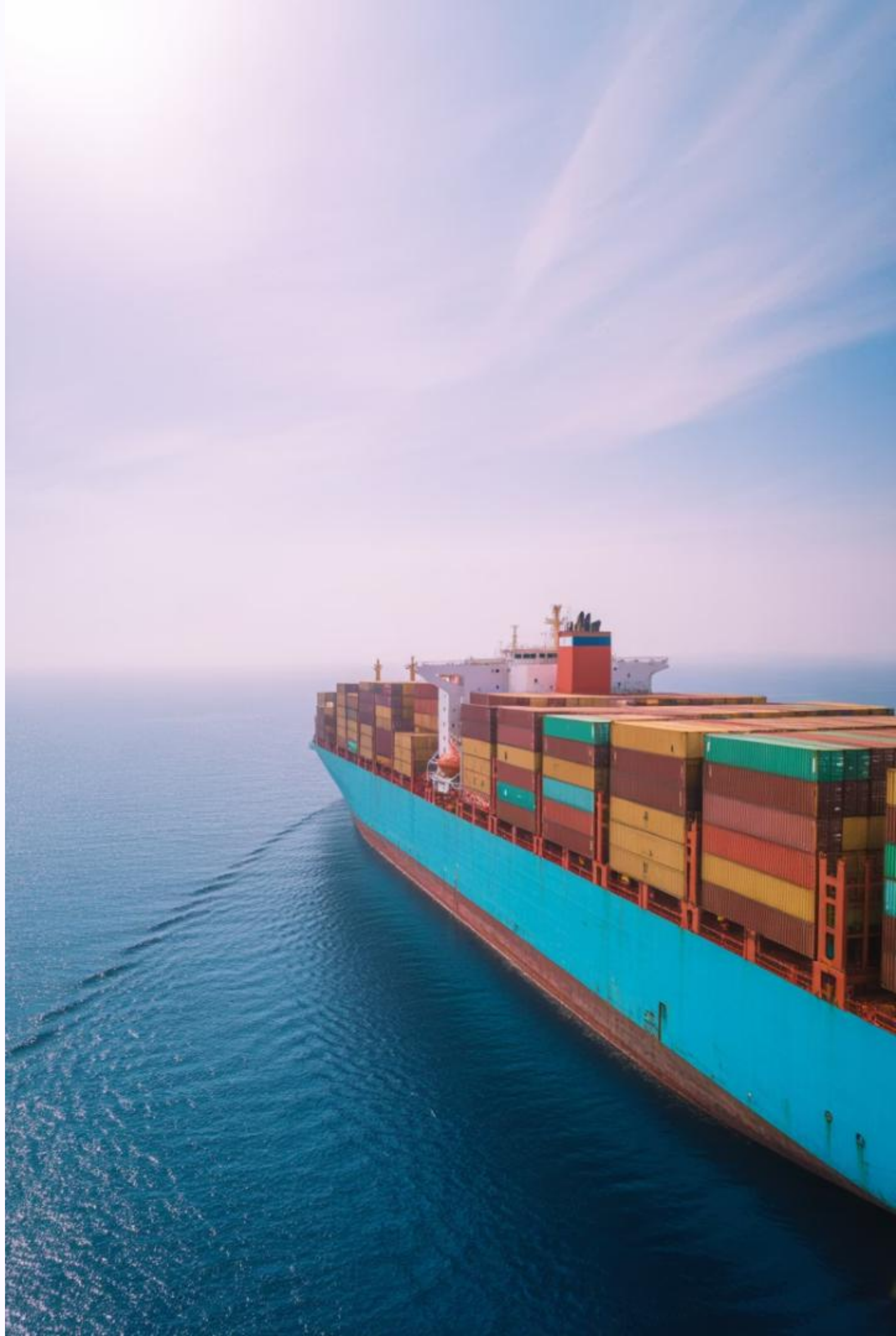
Tools to build, ship, run apps

Container system

Everything packaged together

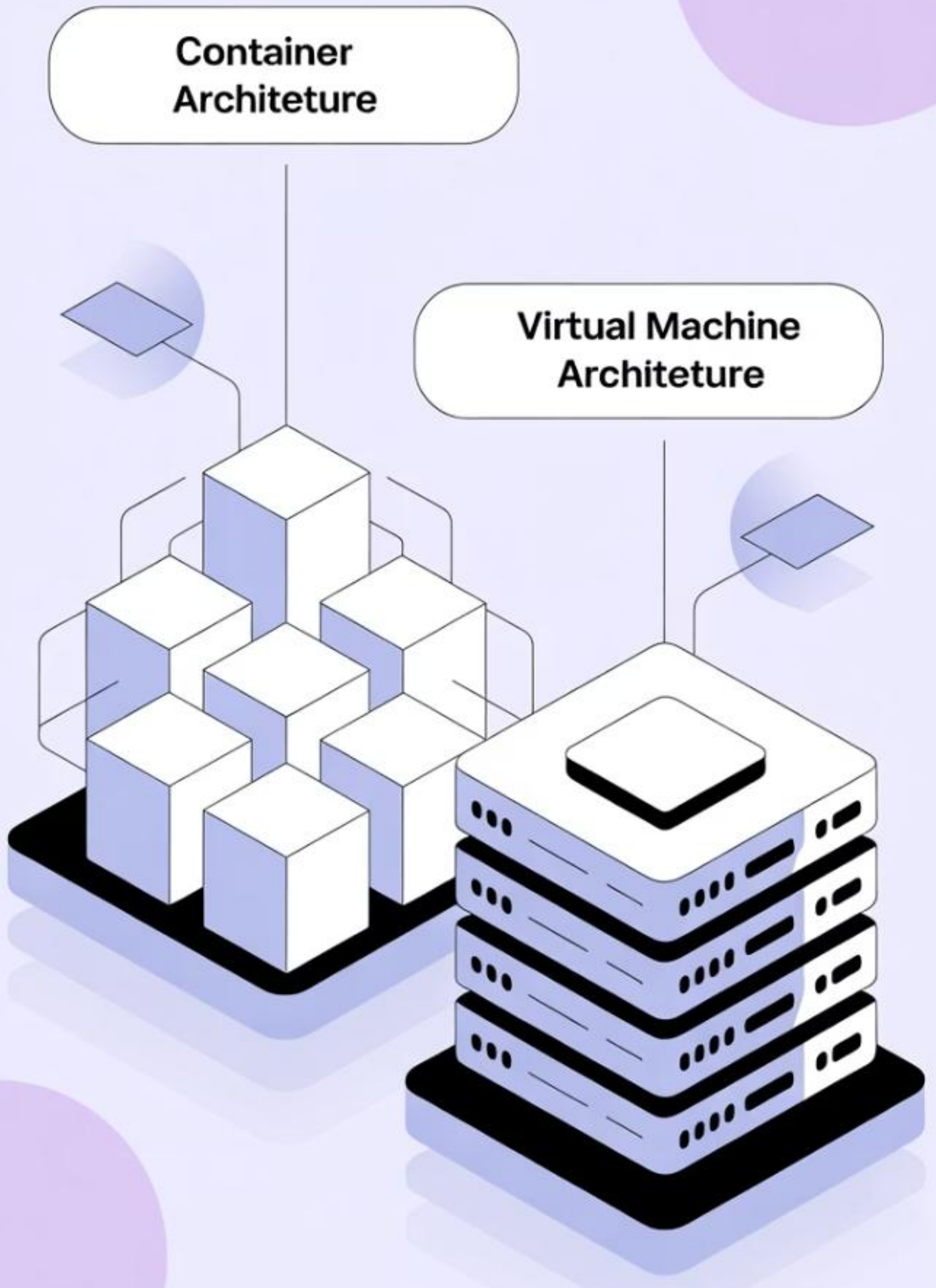
Consistent environments

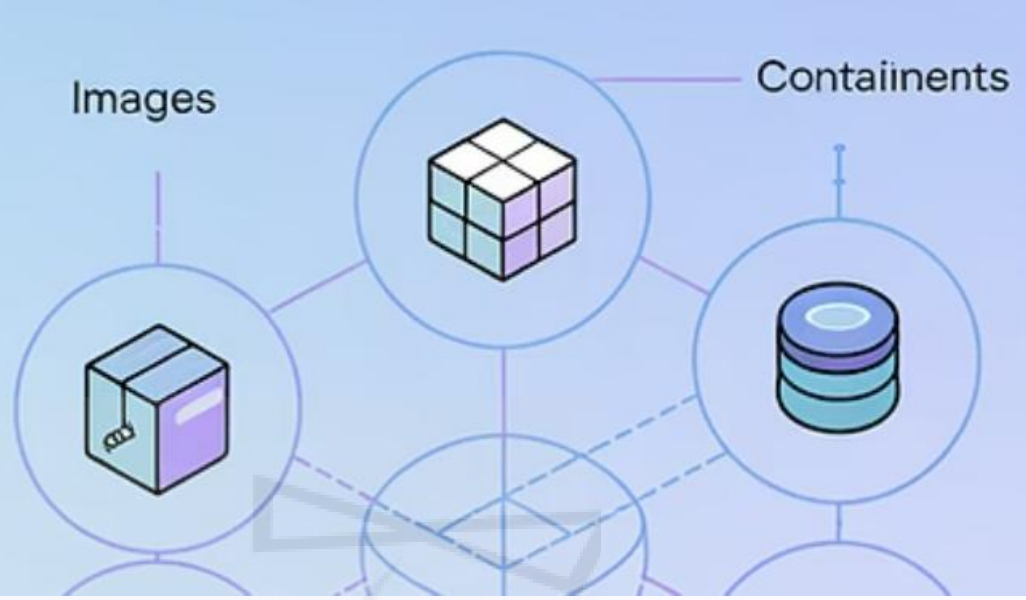
Same setup everywhere



Containers vs. Virtual Machines

Feature	Containers	Virtual Machines
Size	Lightweight (MBs)	Heavy (GBs)
Startup	Seconds	Minutes
Resources	Low usage	High usage
Isolation	Process-level	Full OS-level





Key Docker Concepts

Image

Blueprint for your app

Container

Running instance of image

Dockerfile

Instructions to build image

Docker Hub

Public registry for images

Docker Architecture

Docker Client

CLI you interact with

Docker Daemon

Builds and runs containers

Registry

Stores Docker images

Docker Workflow

Write Dockerfile

Define your app environment

Build Image

docker build -t myapp .

Share (Optional)

Push to Docker Hub

Run Container

docker run -p 8000:8000 myapp

Anatomy of a Dockerfile

FROM python:3.9

Start with base image

COPY . /app

Add your code

WORKDIR /app

Set working directory

RUN pip install

Install dependencies

CMD ["python", "app.py"]

Command to run app

Essential Docker Commands

docker build -t myapp .

Create image from Dockerfile

docker run -p 5000:5000 myapp

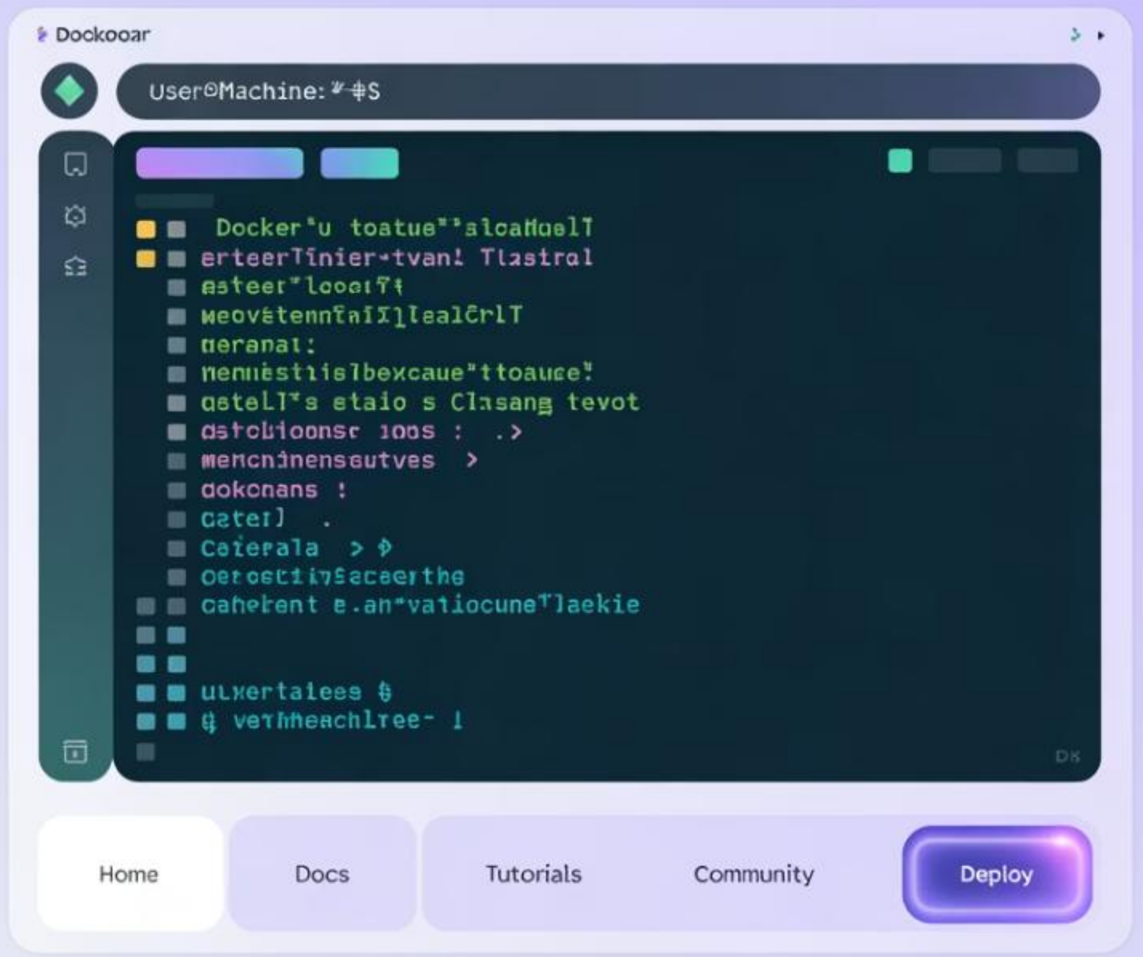
Start container from image

docker ps

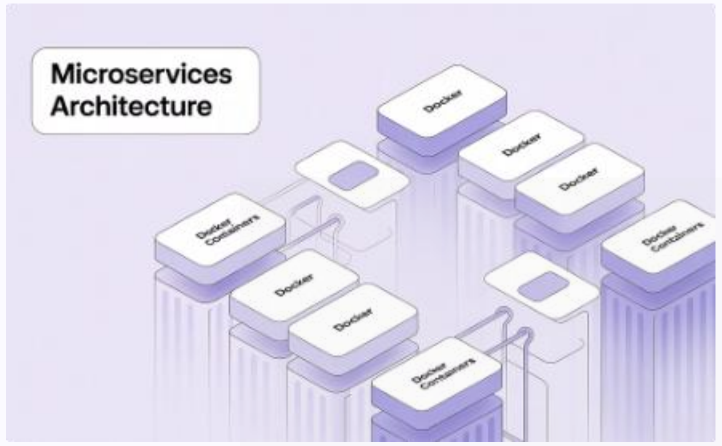
List running containers

docker stop [id]

Stop a running container

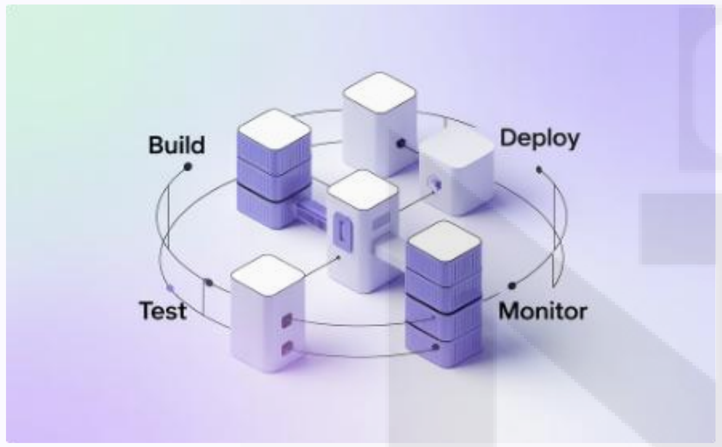


Real-World Use Cases



Microservices

Independent services in containers



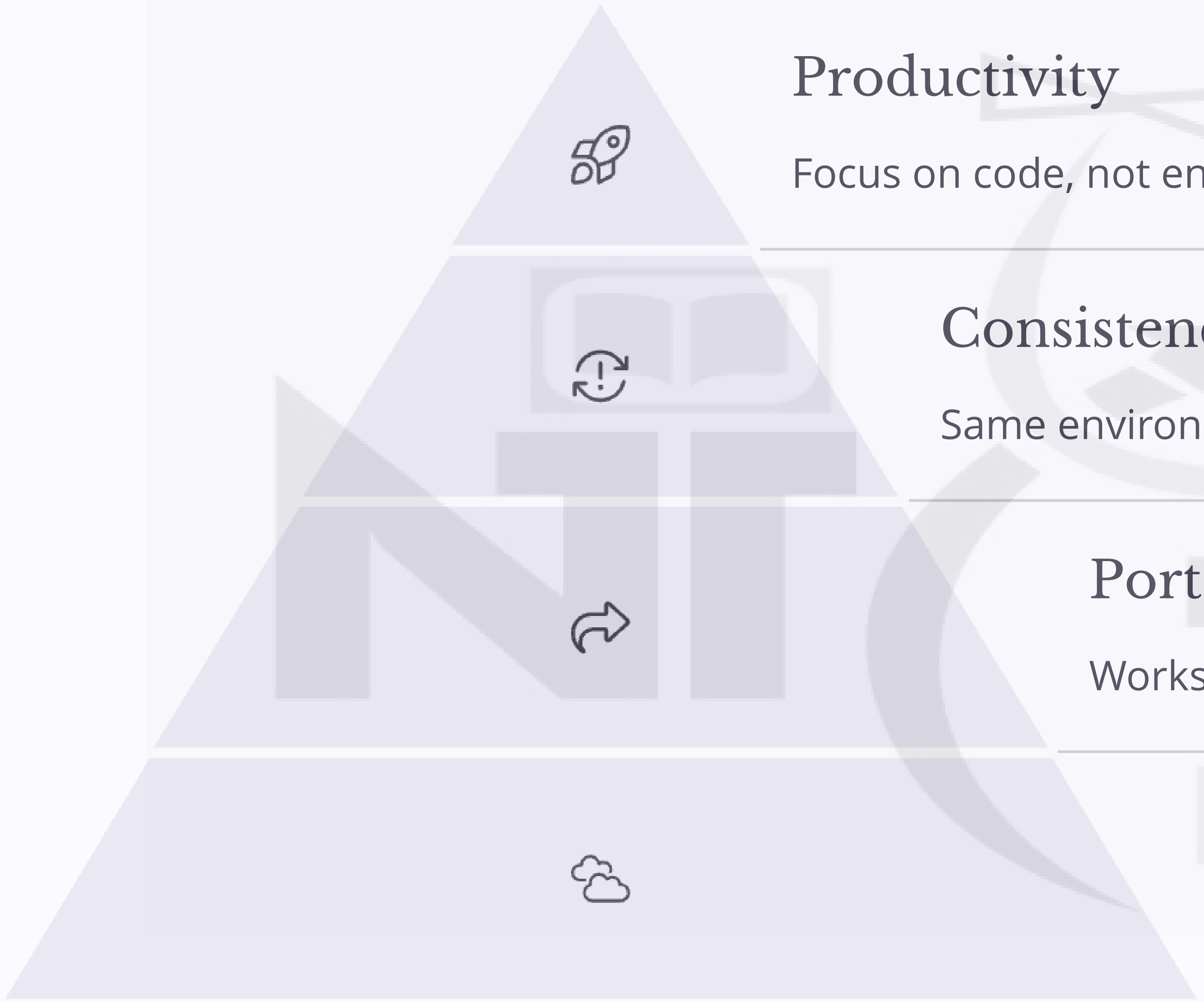
CI/CD Pipelines

Automated testing and deployment

Cloud Deployments

Consistent scaling in any cloud

Benefits of Docker



Productivity

Focus on code, not environment

Consistency

Same environment everywhere

Portability

Works on any system

Scalability

Easy to grow with demand

Common Misconceptions

Myth	Reality
• Docker replaces VMs completely	• Different tools for different jobs
• Only for production deployment	• Incredibly useful during development
• Too complex for beginners	• Basic usage is straightforward
• Only useful for large teams	• Valuable even for solo developers

Docker Demo

1

Dockerfile

Write simple web app config

2

Build

Create container image

3

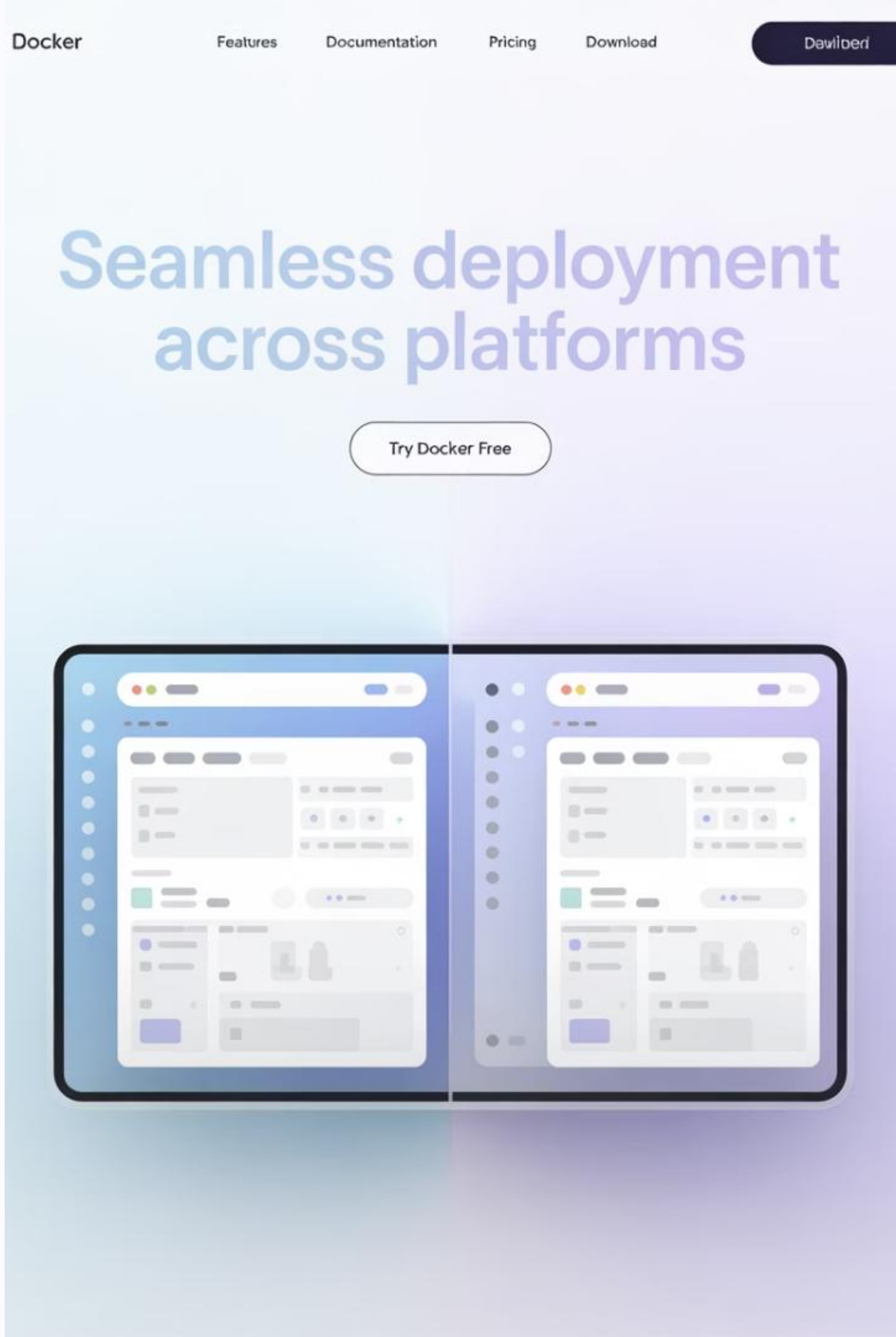
Run

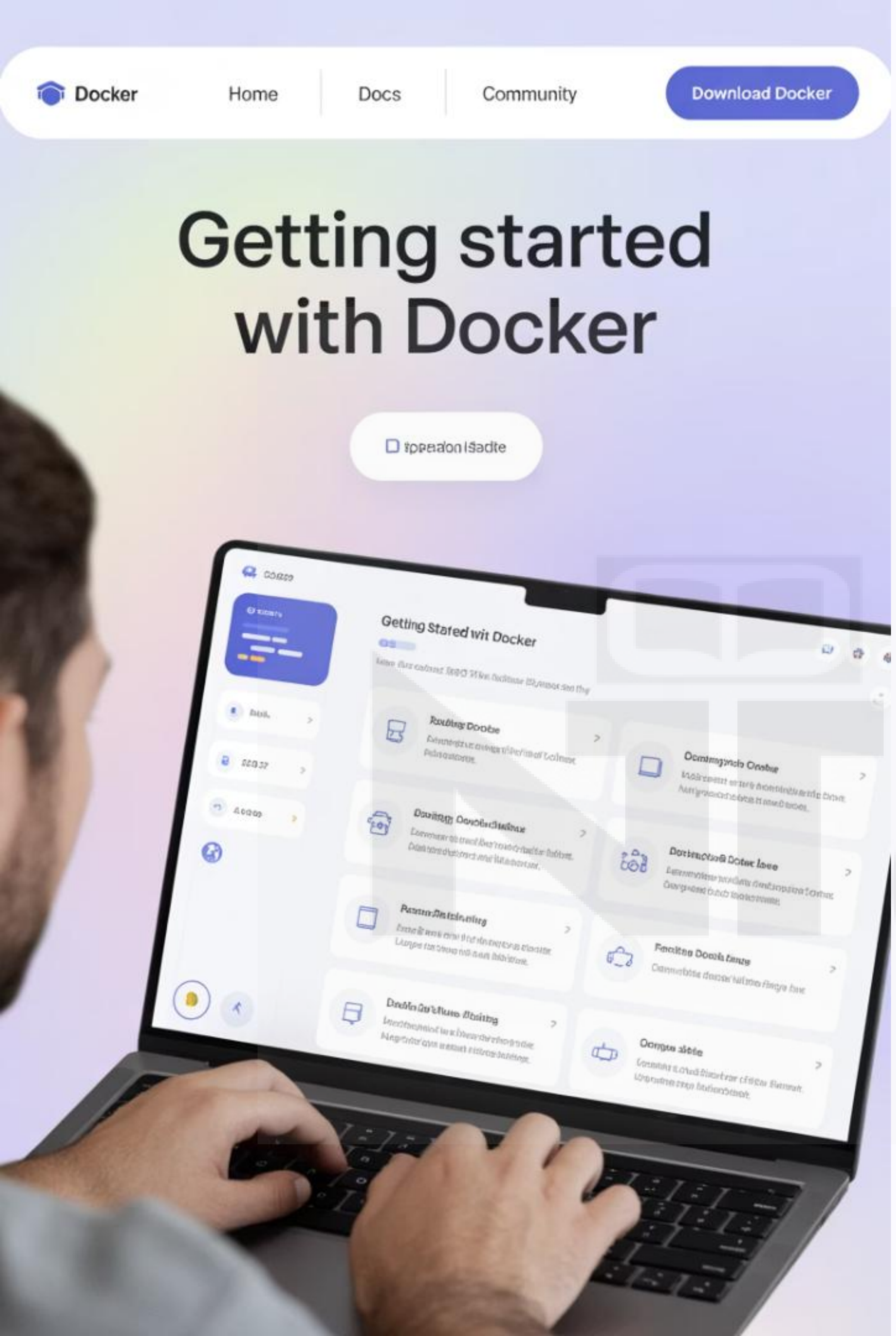
Start app in container

4

Share

Works on any machine





Getting Started Steps

Install Docker Desktop

Free for personal use

Follow official tutorials

Start with Docker getting started guide

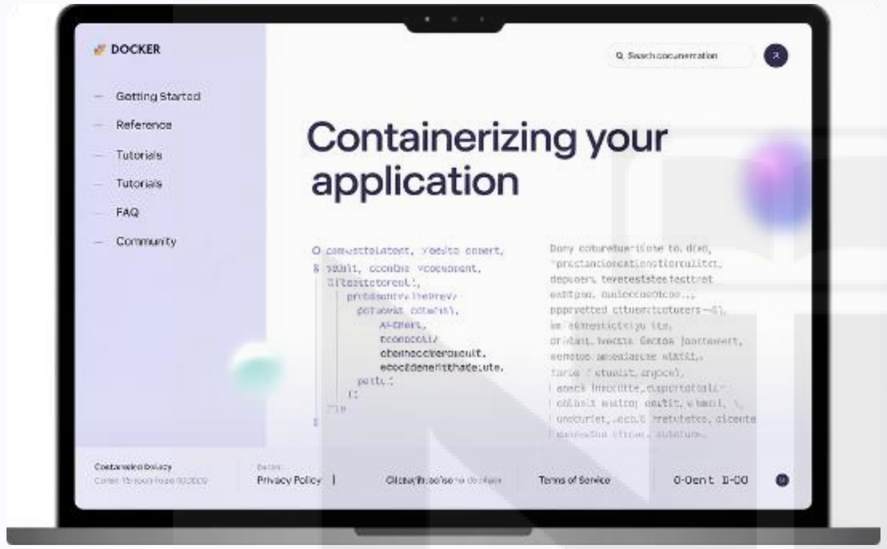
Try simple project

Containerize a basic web app

Join community

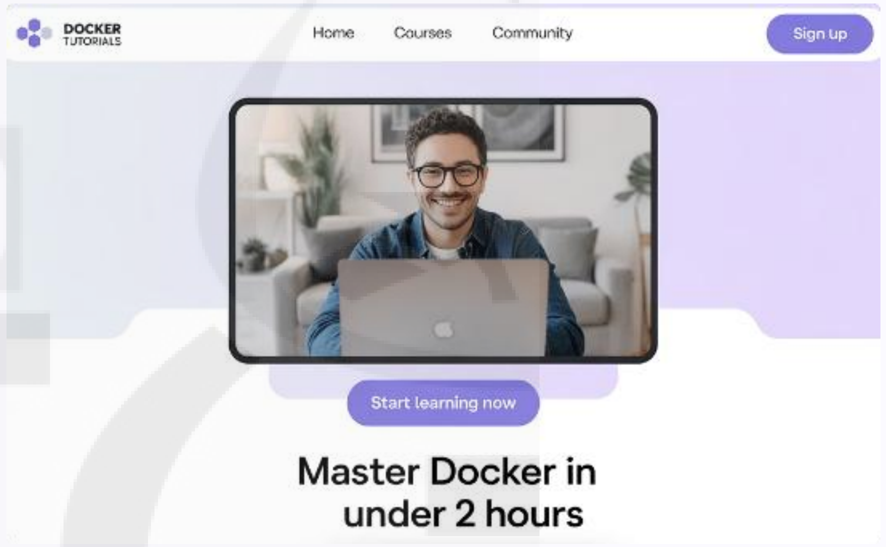
Docker forums and Stack Overflow

Learning Resources



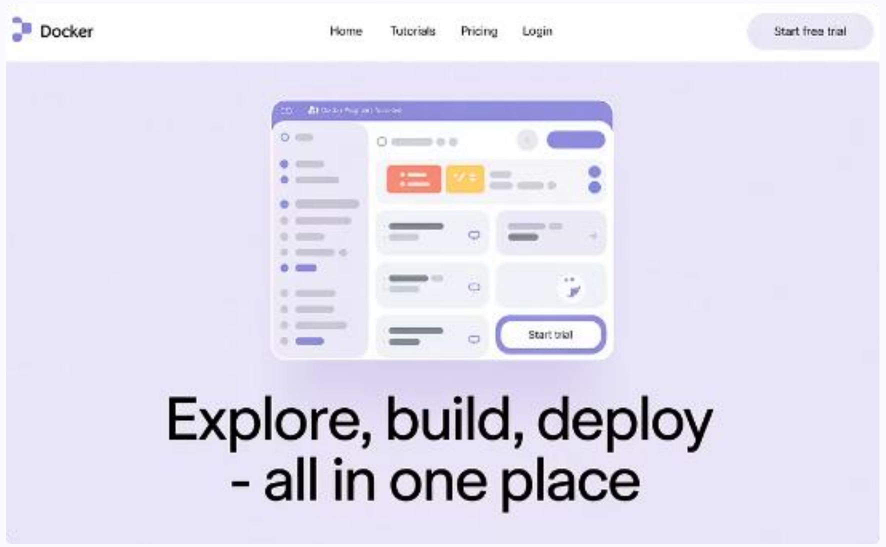
Official Docs

docs.docker.com



Video Tutorials

YouTube has great free content



Docker Playground

Try without installing

Key Takeaways

Containers simplify development

Eliminate "works on my machine"

Basic commands are enough

Build, run, stop, push

Valuable career skill

In-demand for most tech roles

Learning curve is manageable

Start small, build gradually