# Introduction

## Microservices Workshop

# Agenda

- Introduction
- Labs Infrastructure
- Meeting the Application
- Pipeline Design
- Managing Services
- Summary

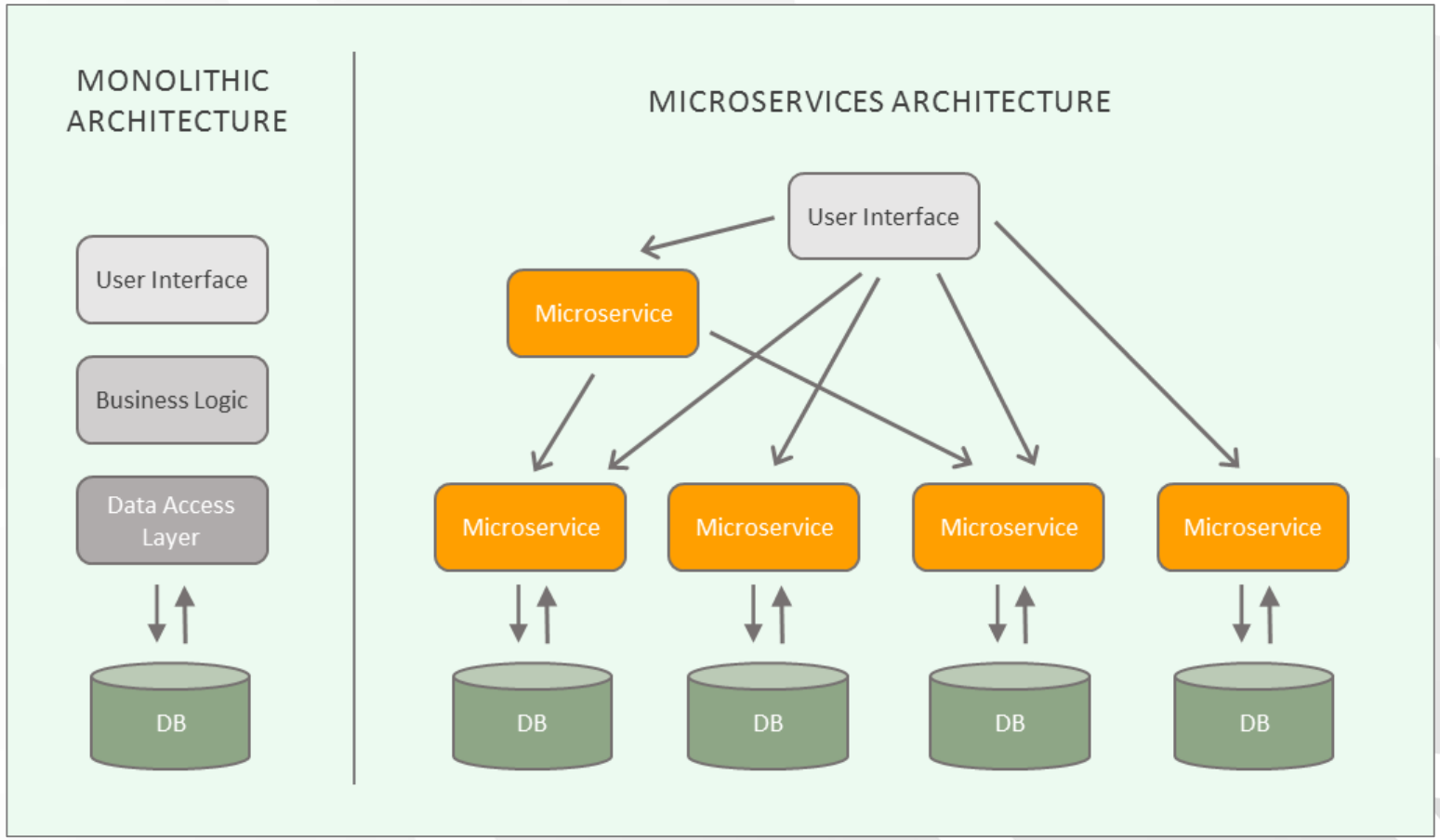
# Agenda

- Lab 01: Setting up the required infrastructure with Docker
- Lab 02: Meeting the application
- Lab 03: Create a CI/CD pipeline for a microservices application
- Lab 04: Updating a single service

# Microservices Architecture

- Microservices architecture is a distinctive method of developing software systems in a modular way.
- Allow developers to build their applications from various independent components which can easily be changed, removed or upgraded without affecting the whole application.
- These services are created to serve only one specific business function.
- Services are independent of each other (they can be written in different programming languages and use different data storages.

# Monolithic VS Microservices

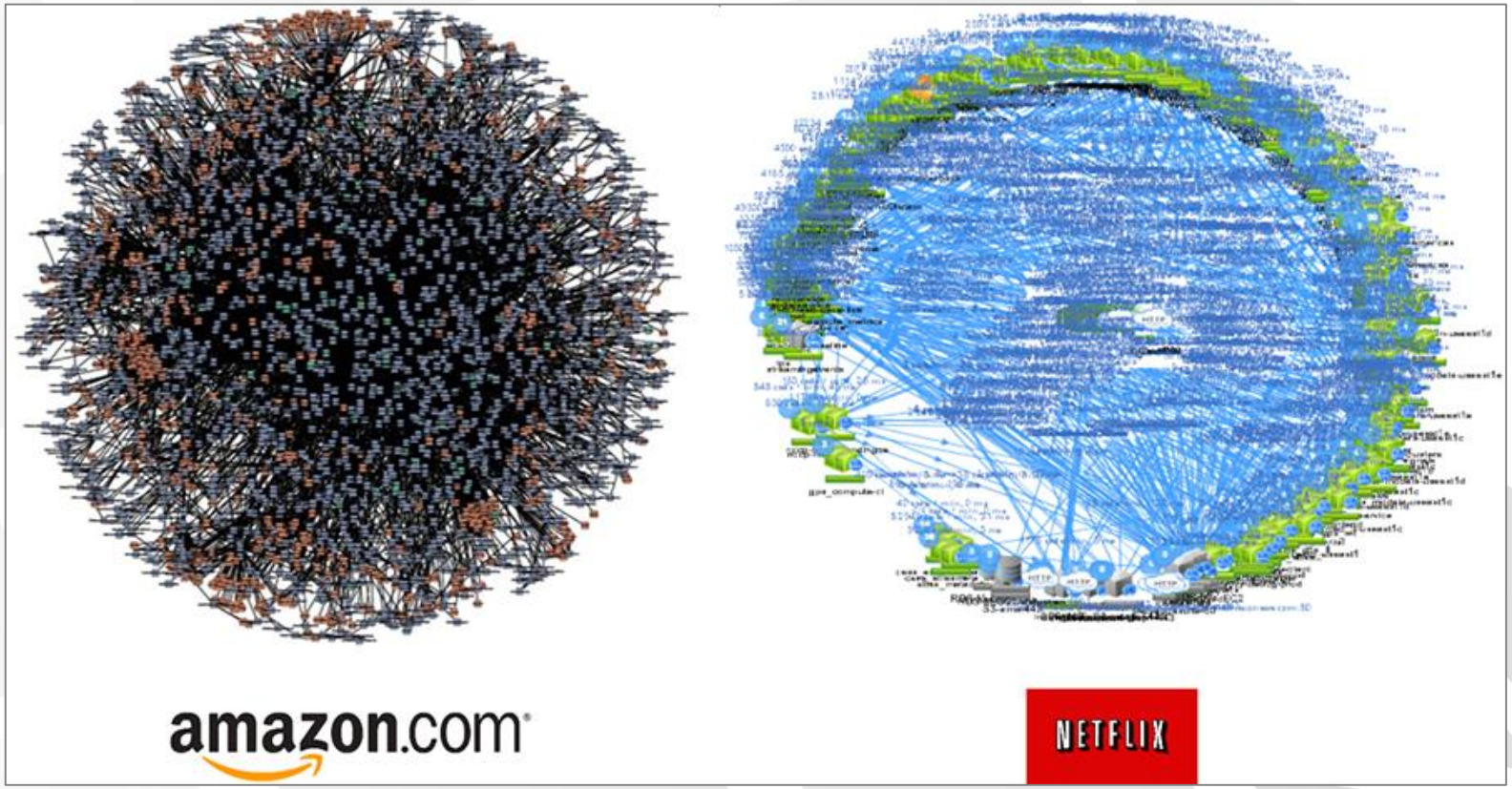


# Microservices Benefits

- Deployability (increased agility, everything “moves faster”)
- Reliability (when monolithic service fail, the entire monolith may fail)
- Availability (microservices “requires” very little downtime in upgrades)
- Scalability (microservices can be scaled independently)
- Modifiability (microservices are modular)
- Management (smaller and independently teams)

# Who is using Microservices?

- Netflix
- eBay
- Amazon
- Twitter
- PayPal
- Uber
- Groupon

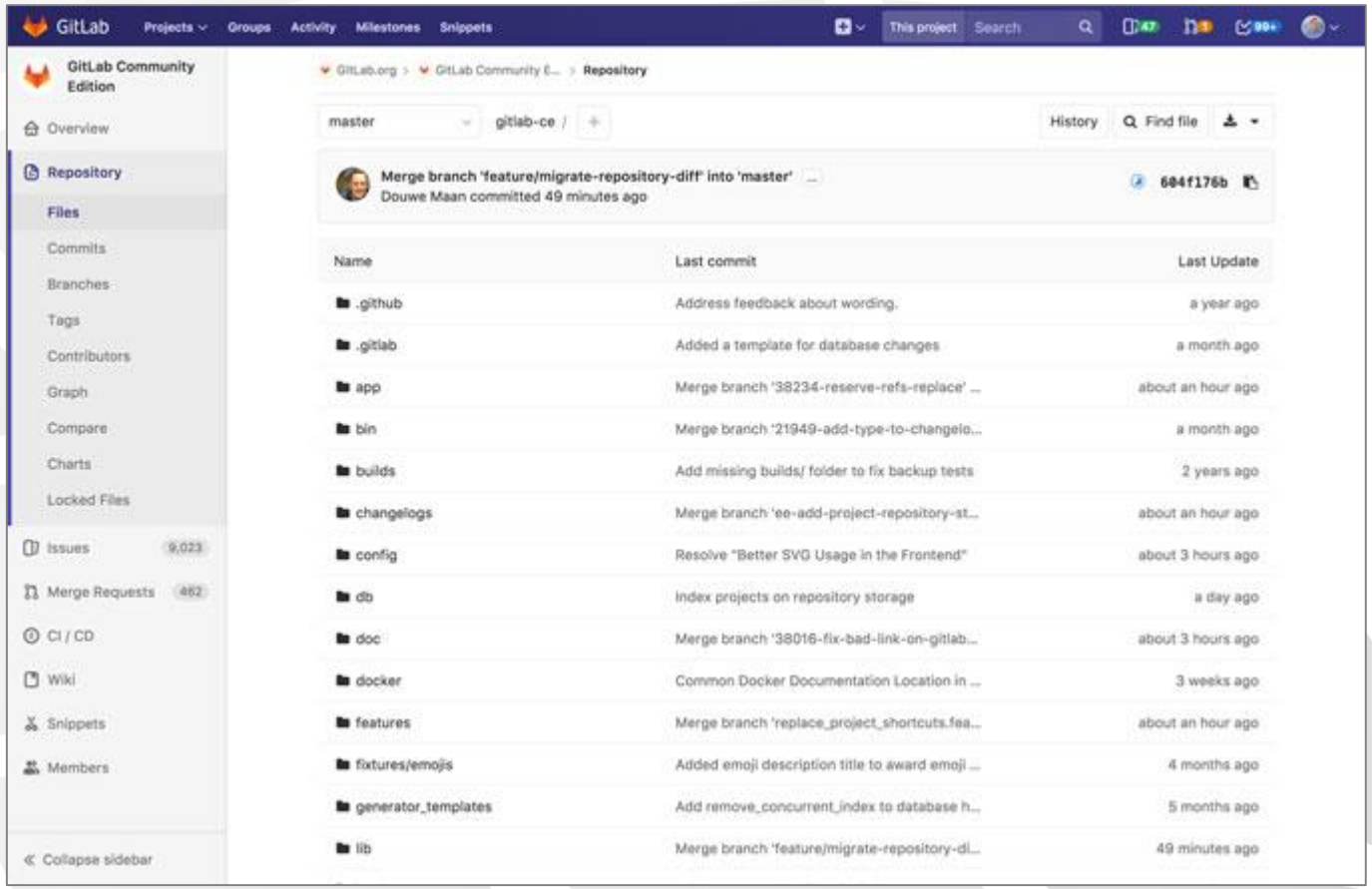


# So, Why Microservices?

The goal of microservices is to ease the building, maintaining and managing of an application by breaking it down into smaller, composeable pieces which work together and can be independently deployed, upgraded, removed or scaled whenever the need arises.

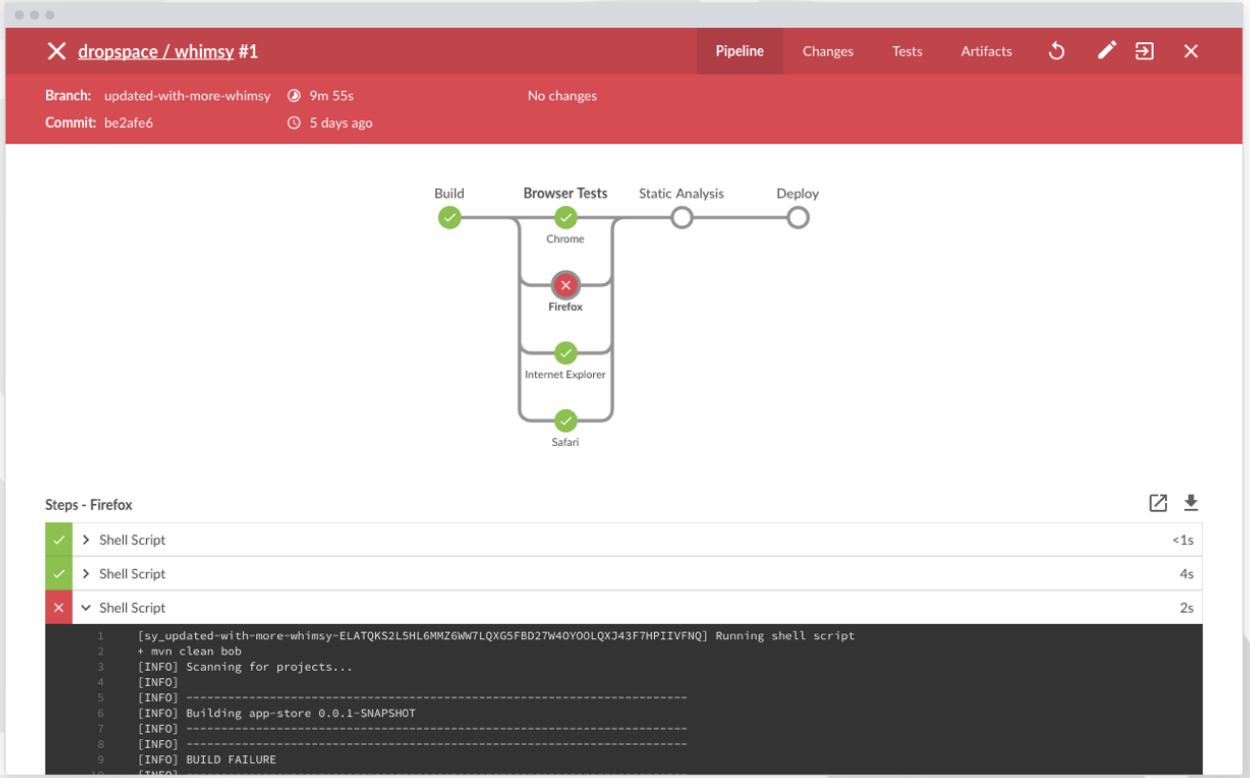
# GitLab CE for Version Control

- Open source
- Easy configuration
- Docker installation
- Free private repositories
- Most used Git Manager
- Many set of features



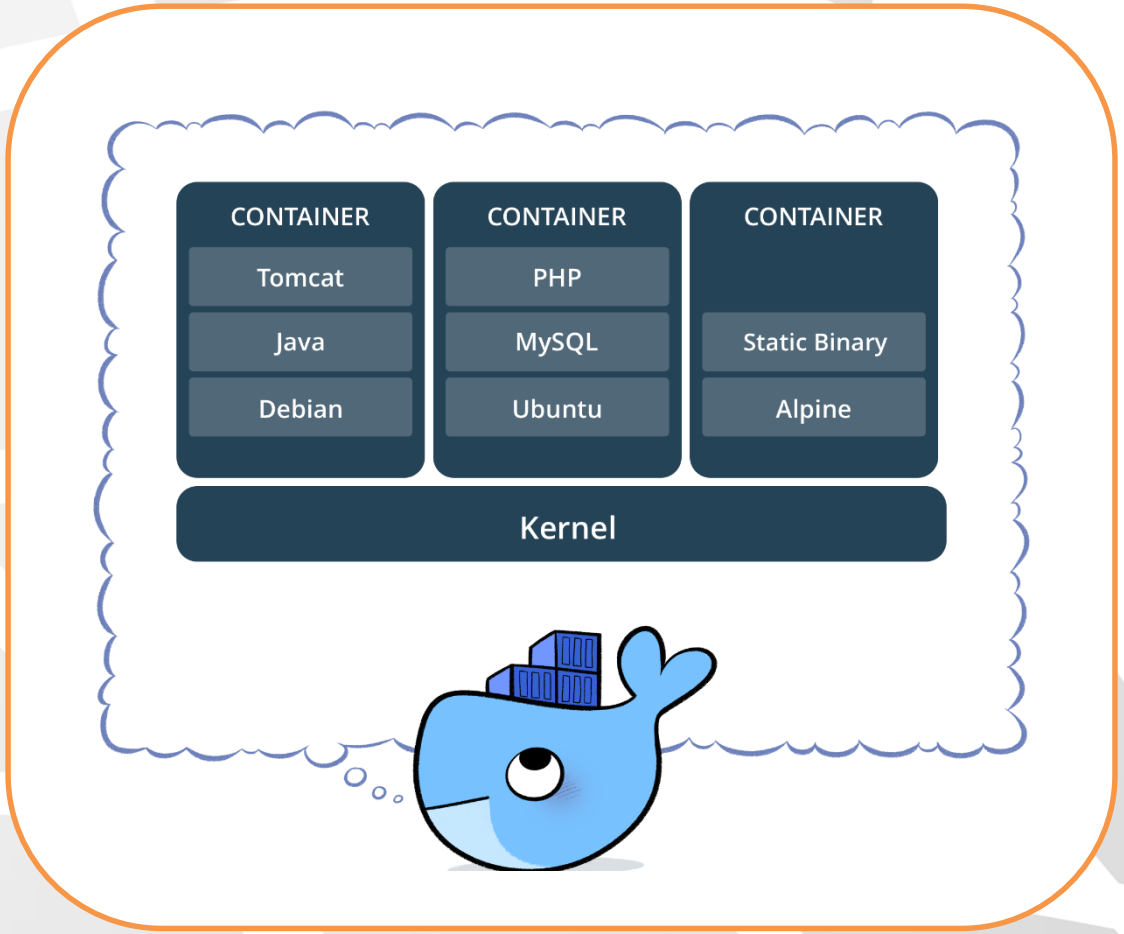
# Jenkins (Blue Ocean) for CI/CD

- Free and open source
- Can automate everything
- A lot of available plugins
- Simple configuration
- Infrastructure as code
- Great traceability and debug
- Many documentation in the web



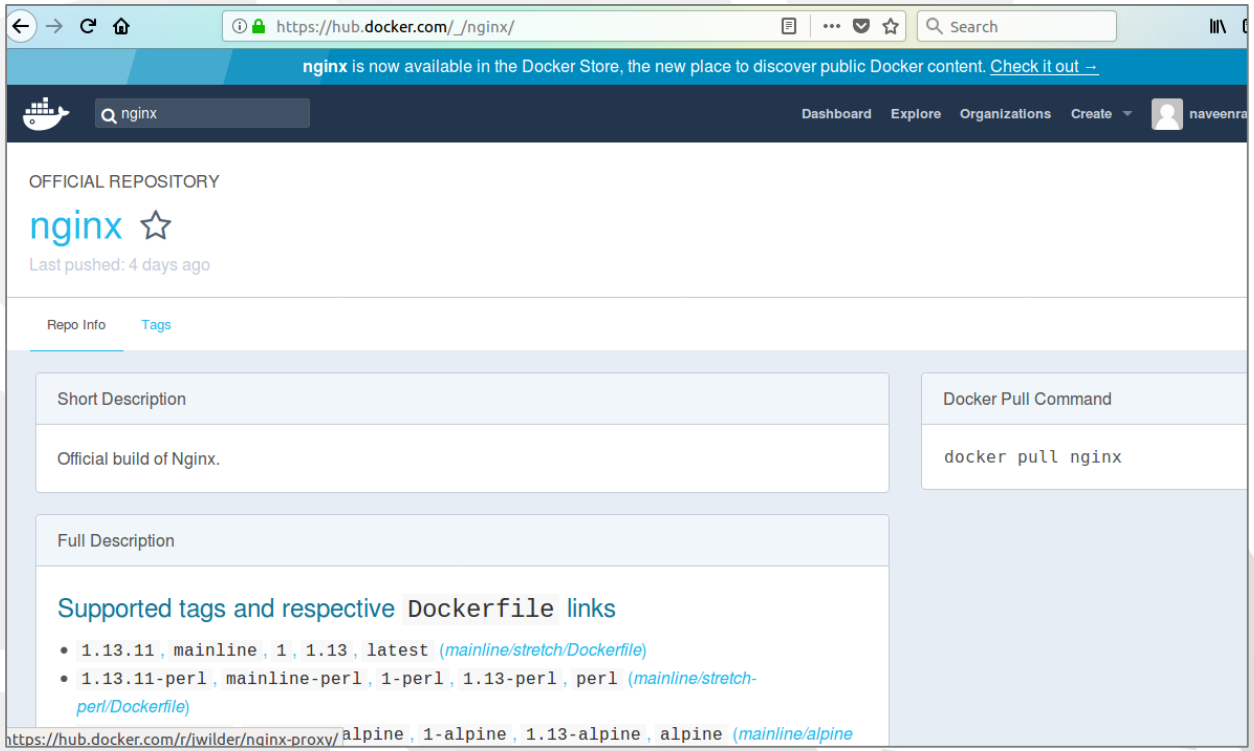
# Docker for Infrastructure

- Standardization
- CI Efficiency
- Compatibility
- Simple and faster configuration
- Rapid deployments
- Hybrid platforms
- Isolation and security



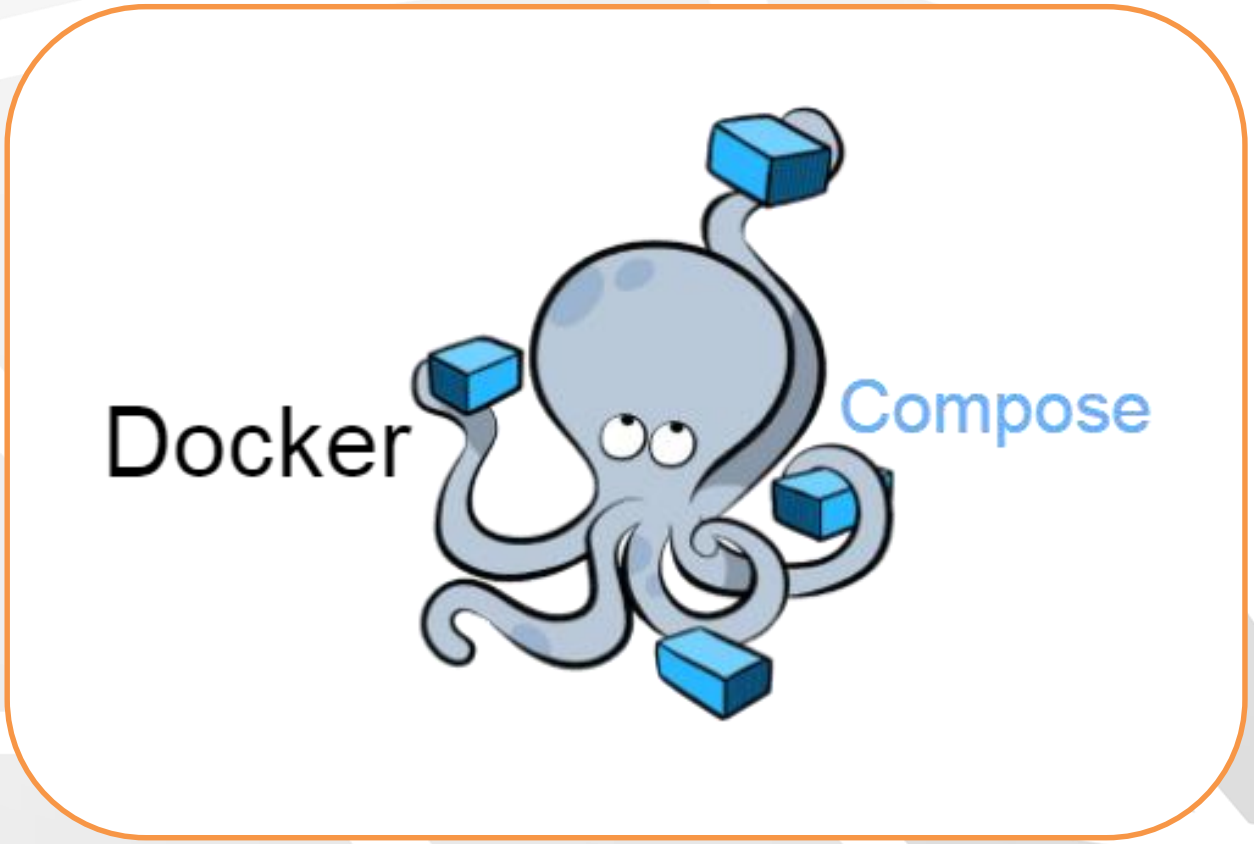
# DockerHub for container registry

- Free and online
- SaaS (no installation required)
- Easy usage
- Default Docker registry
- Most used Docker Registry

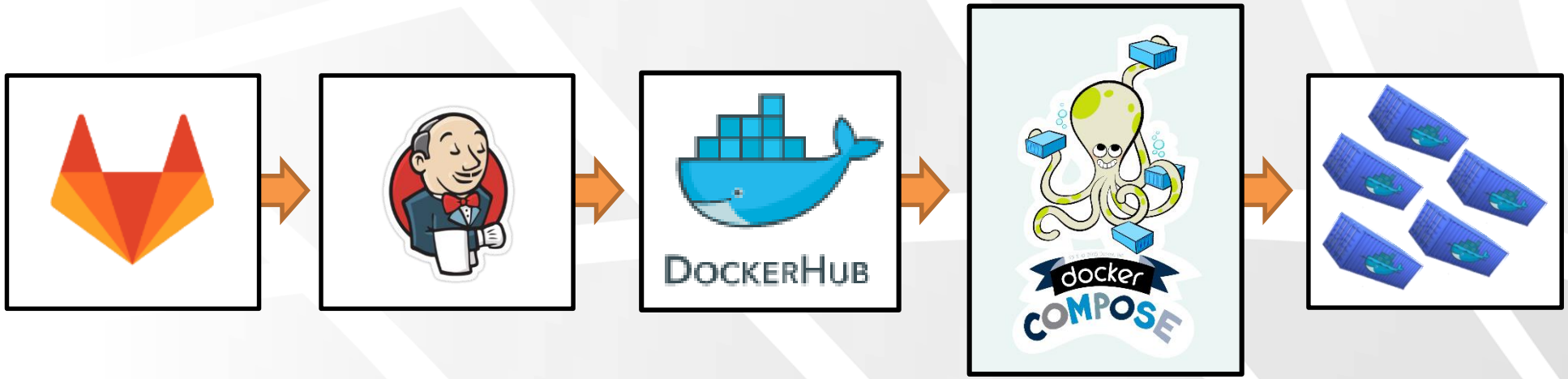


# Deployments with docker-compose

- Infrastructure as code
- Portability
- Containers management
- Volume management
- Network management
- Scalability



# Putting All Together



# Questions