Getting Started with OpenShift for Developers

Openshift 101

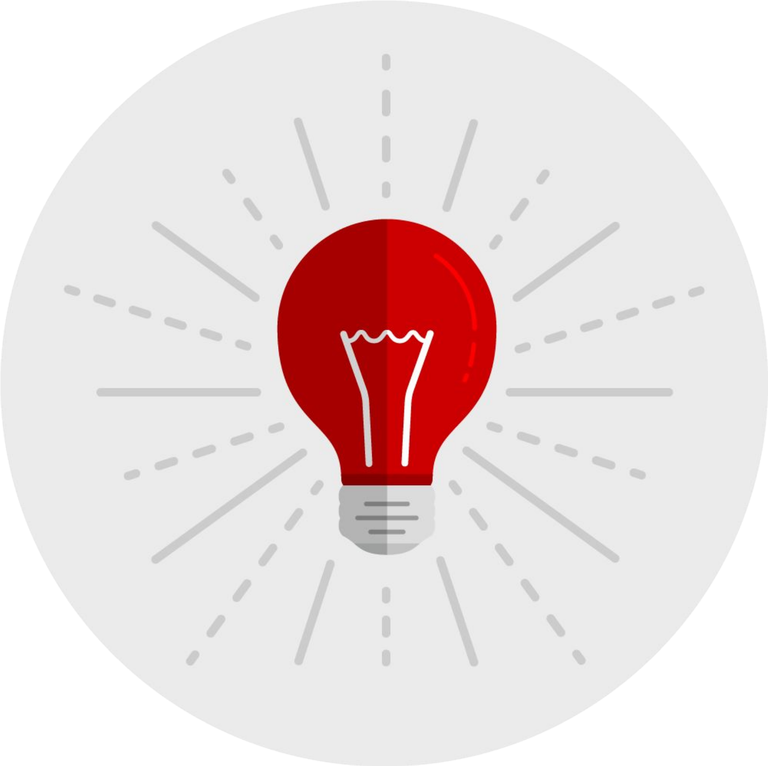
Arslan Khan
Solution Architect

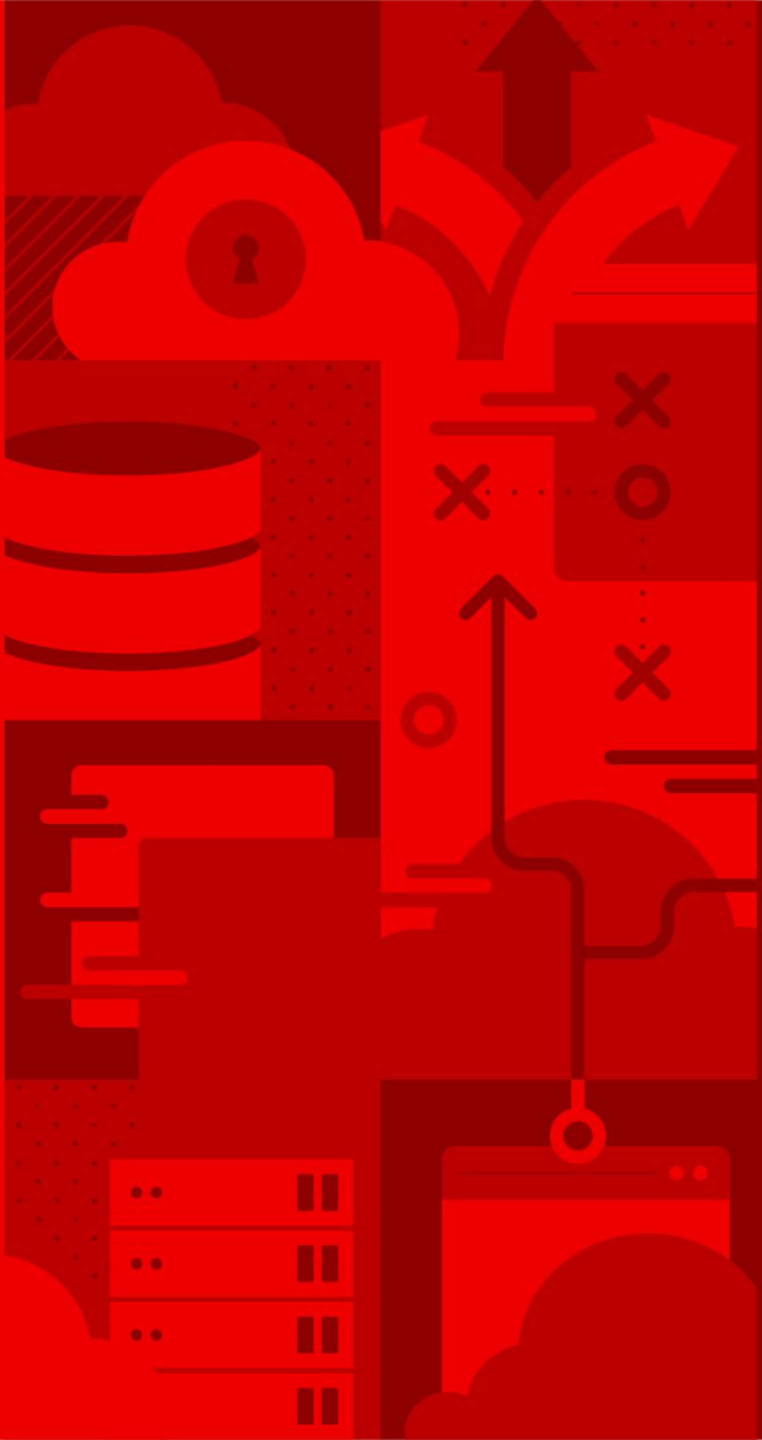
INTRODUCTION

AGENDA

13:00→ 15:00

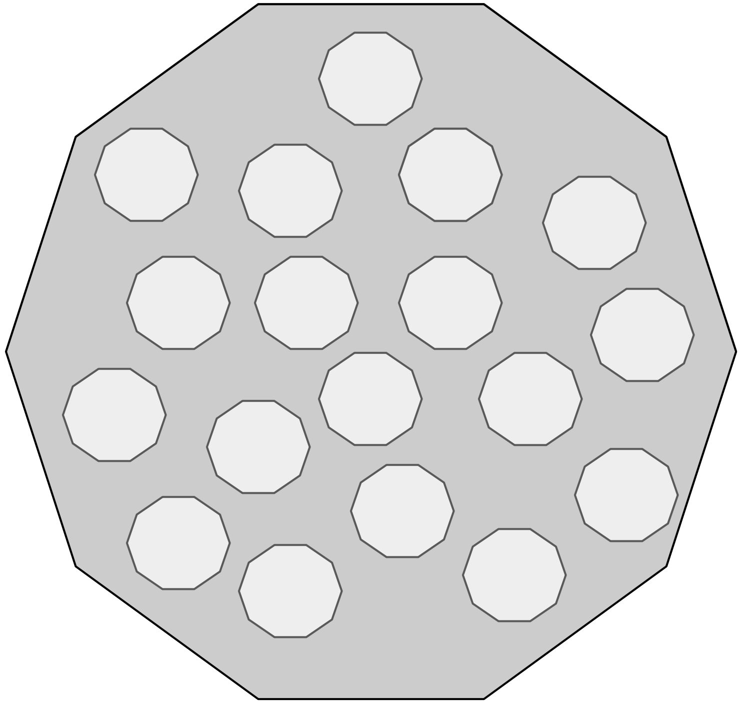
- MicroServices
- Kubernetes
- OpenShift
- Demo
  - Parkmap Application
- App delivery on OpenShift
  - OpenShift Pipelines
  - OpenShift GitOps



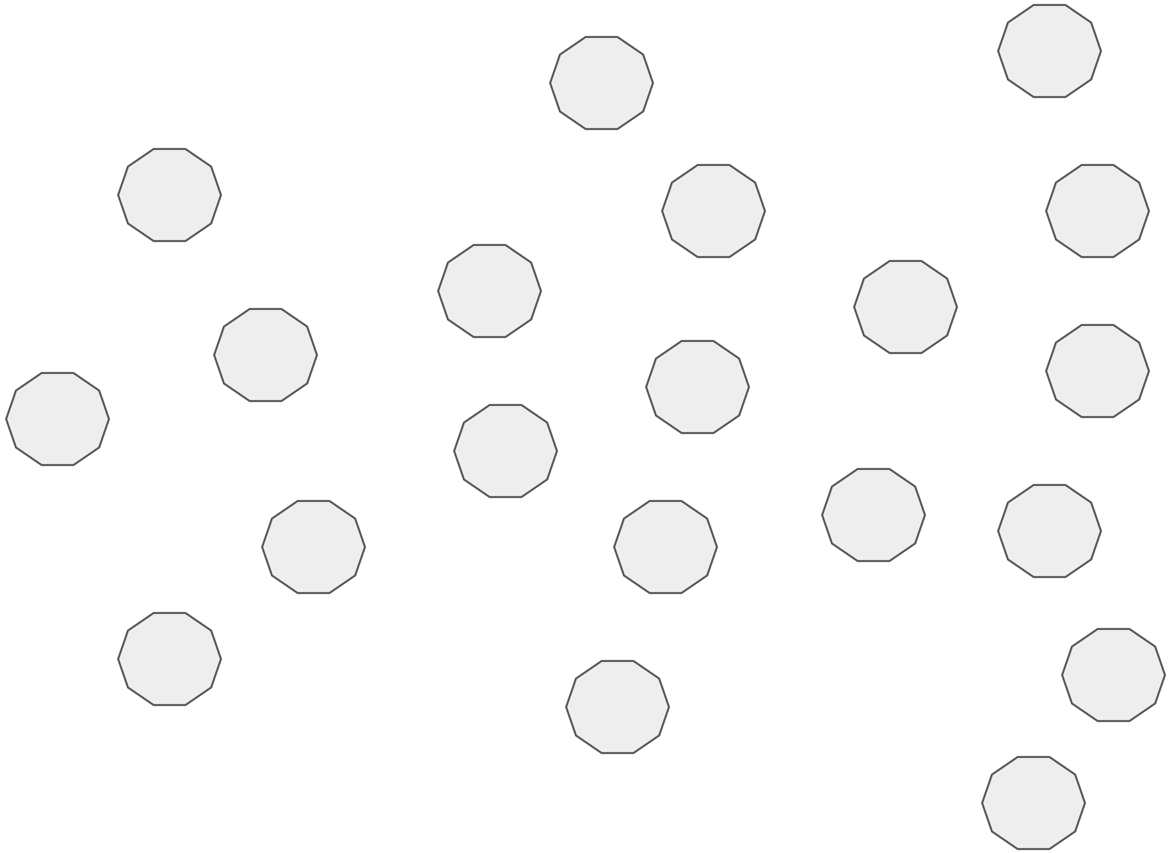


Why Microservices ?

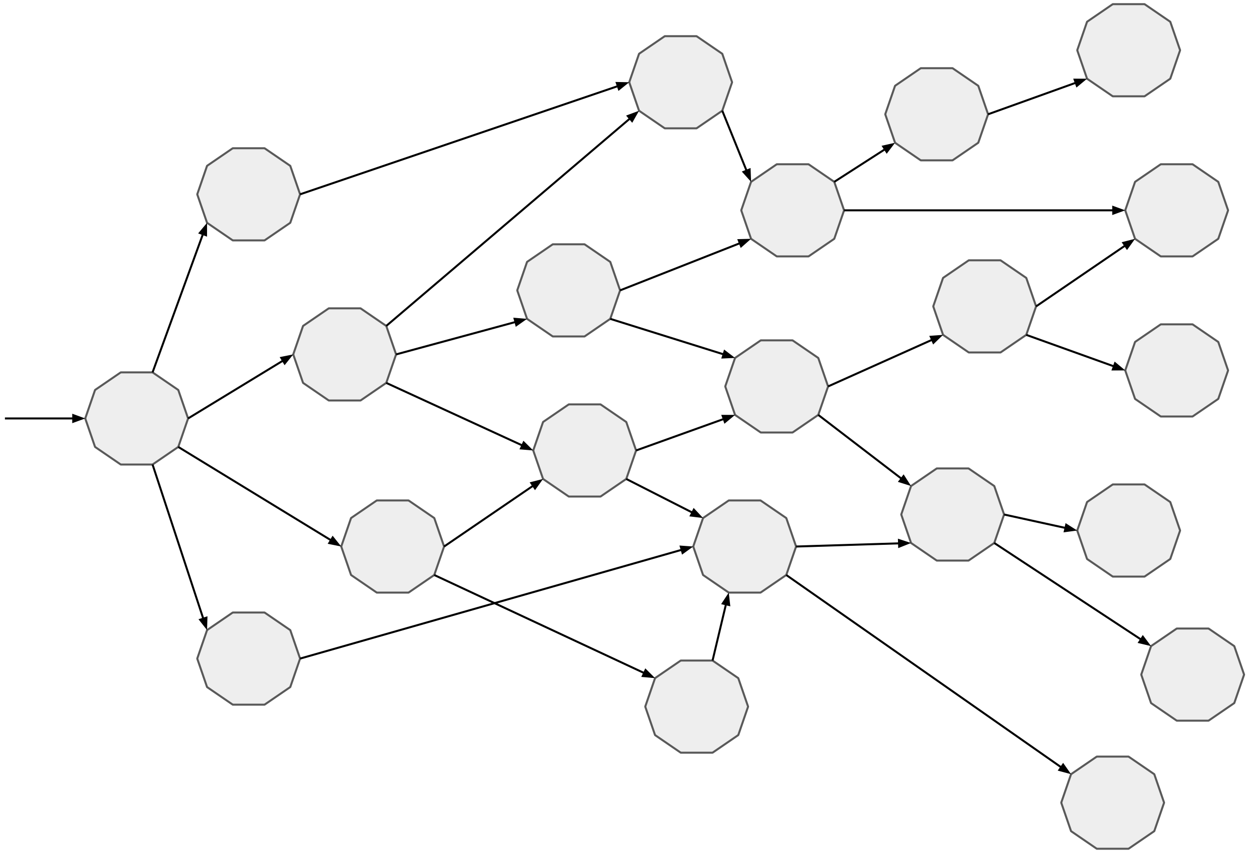
The Monolith Application



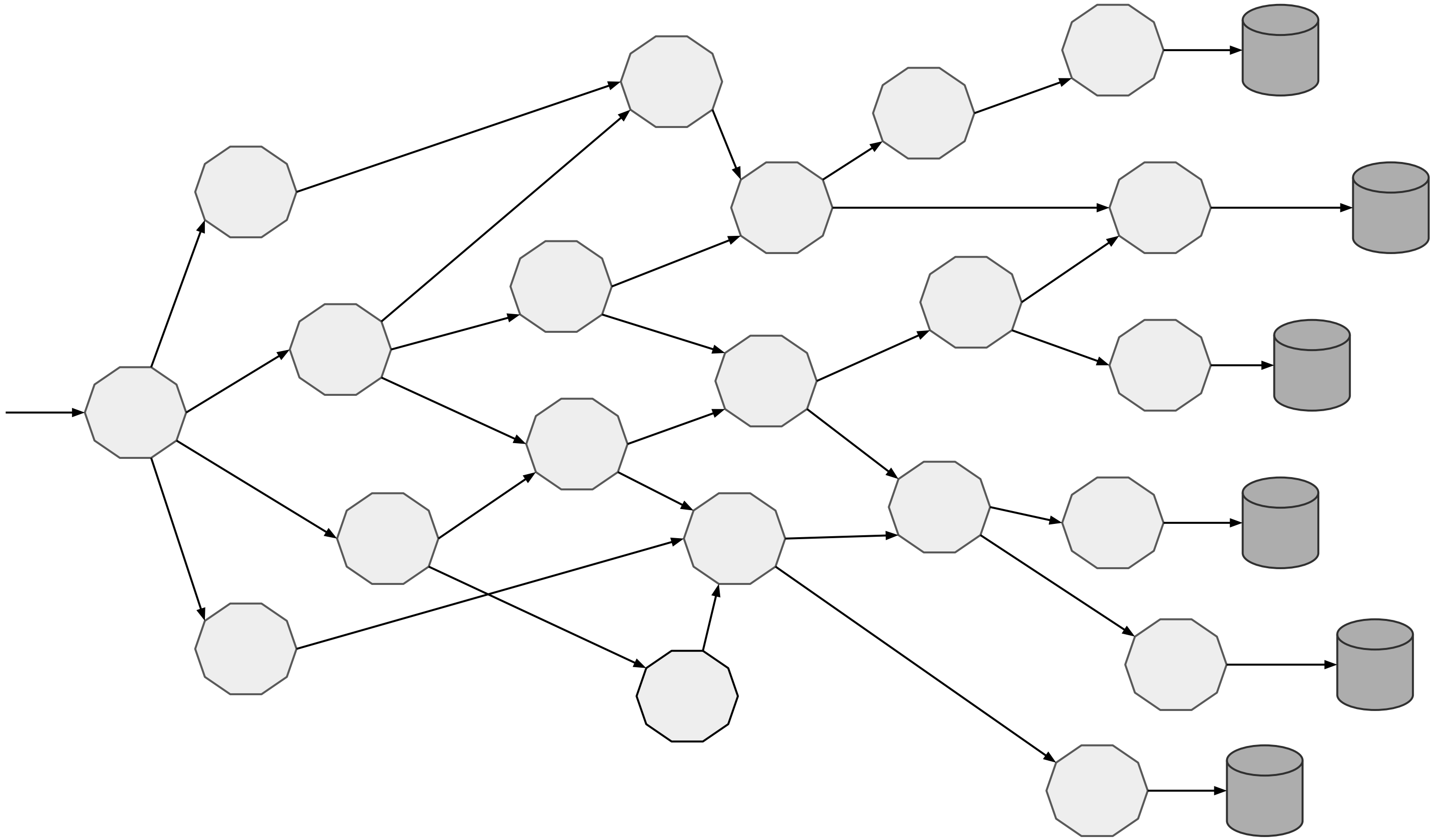
Microservices

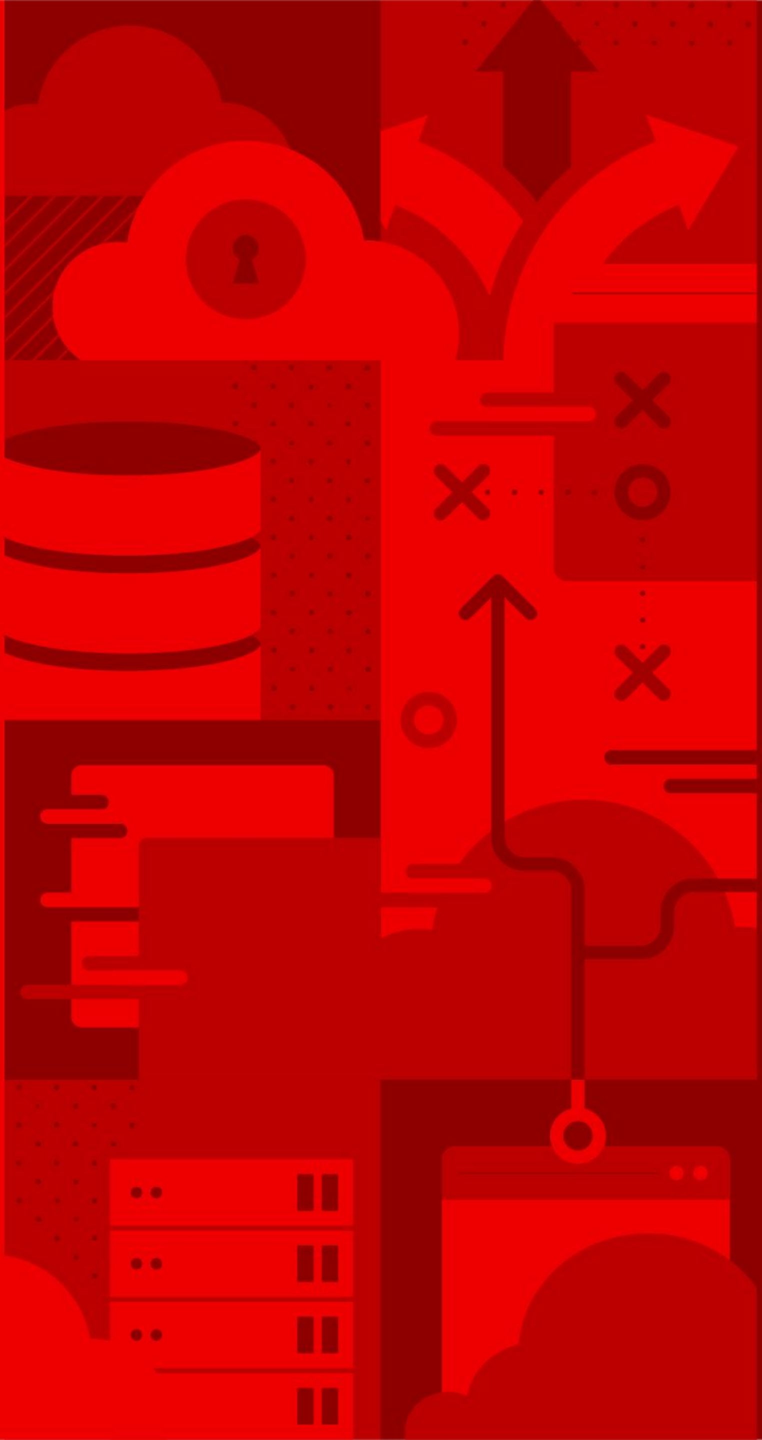


Network of Services



Microservices own their Data



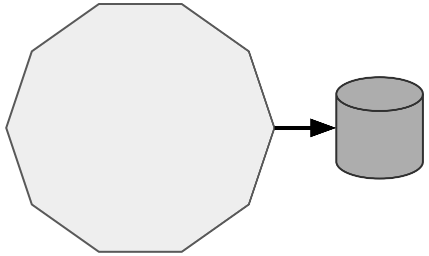


Why Kubernetes ?

What is Kubernetes?

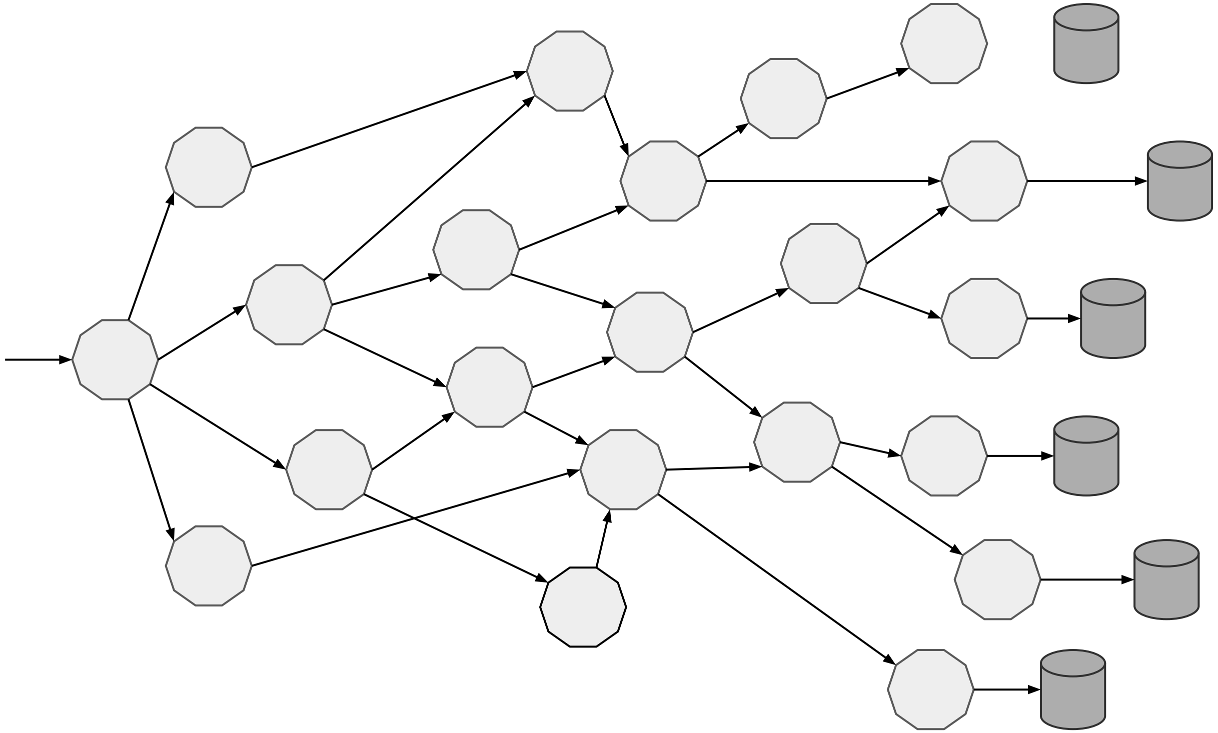
An open source orchestration system for managing containerized workloads across a cluster of nodes.

Old School

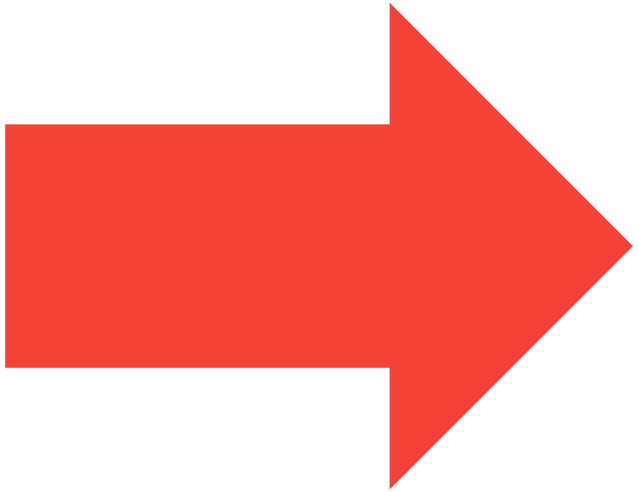


Love Thy Mono

New School



Understanding Kubernetes Objects



Kubernetes objects are persistent entities that represent the **desired state** of your cluster that you can manage with the K8s API

Understanding Kubernetes Objects

Pod

Deployment

Namespace

Service

ReplicaSet

Secret

ConfigMap

PersistentVolume

Kubernetes provides an API

API object primitives include these:

```
kind

apiVersion

metadata

spec

status
```

Deployment

- Helps you specify container runtime, in terms of pods

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: hello-k8s
  creationTimestamp:
  labels:
    run: hello-k8s
spec:
  Replicas: 2
  selector:
    matchLabels:
      run: hello-k8s
  template:
    metadata:
      creationTimestamp:
      labels:
        run: hello-k8s
    spec:
      containers:
      - name: hello-k8s
        image: jkleinert/nodejsint-workshop
        resources: {}
  strategy: {}
status: {}
```

Pod

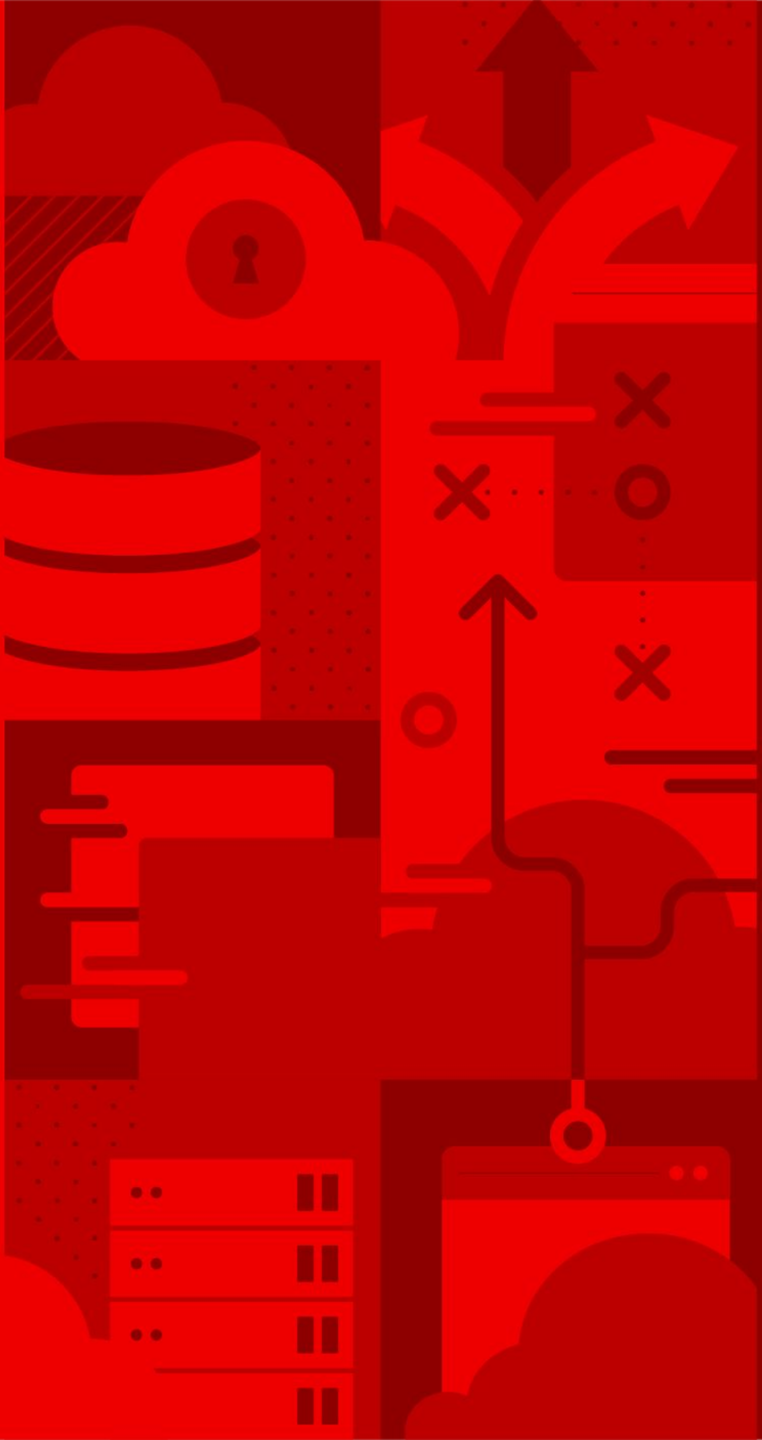
- A group of one or more co-located containers
- Minimum unit of scale

```yaml
kind: Pod
apiVersion: v1
metadata:
  creationTimestamp:
  name: hello-k8s
  labels:
    run: hello-k8s
spec:
  containers:
  - name: hello-k8s
    image: jkleinert/nodejsint-workshop
    ports:
    - containerPort: 8080
    resources: {}
```

Service

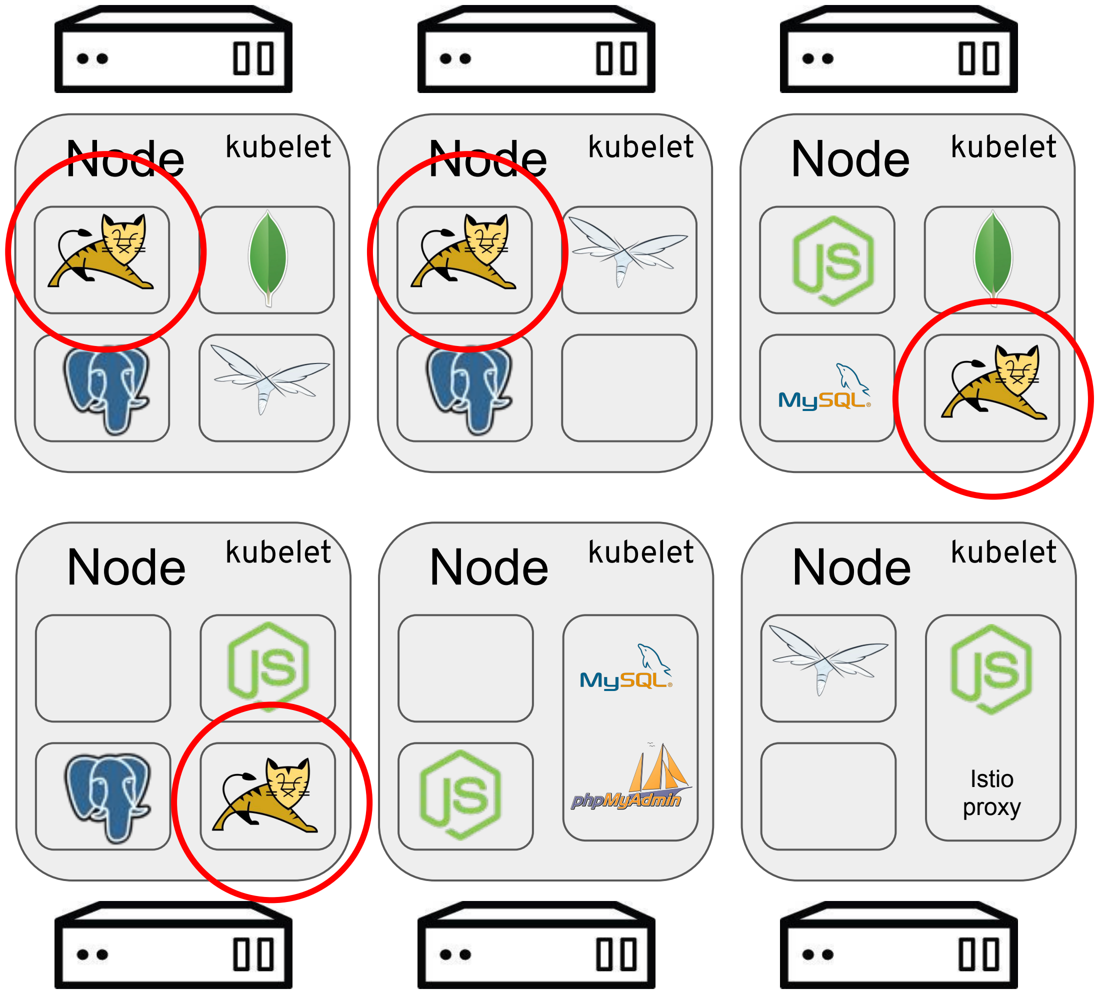
- Acts as a single endpoint for a collection of replicated pods
- Like a load balancer

```
kind: Service
apiVersion: v1
metadata:
  name: hello-k8s
  creationTimestamp:
  labels:
    run: hello-k8s
spec:
  ports:
  - protocol: TCP
    port: 8080
    targetPort: 8080
  selector:
    run: hello-k8s
  type: NodePort
status:
  loadBalancer: {}
```

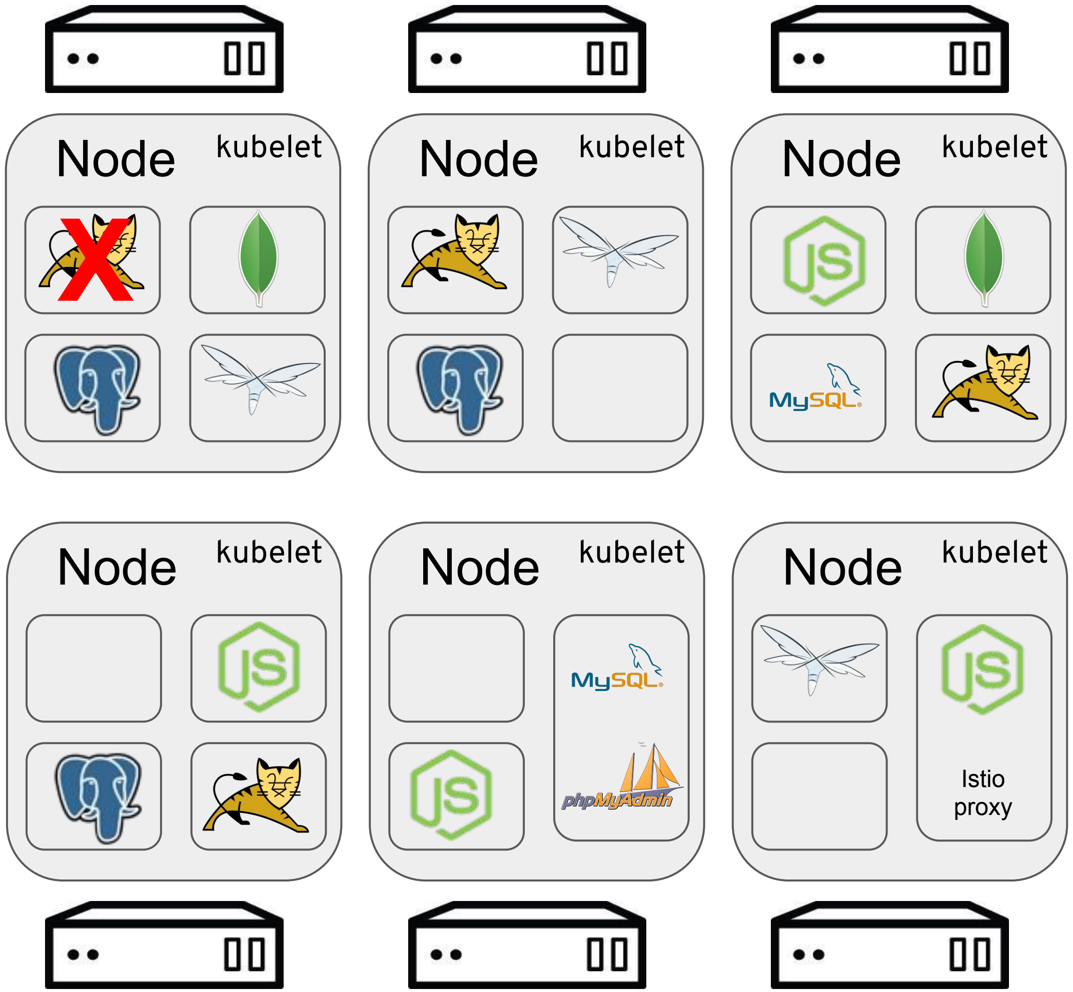


Self Healing

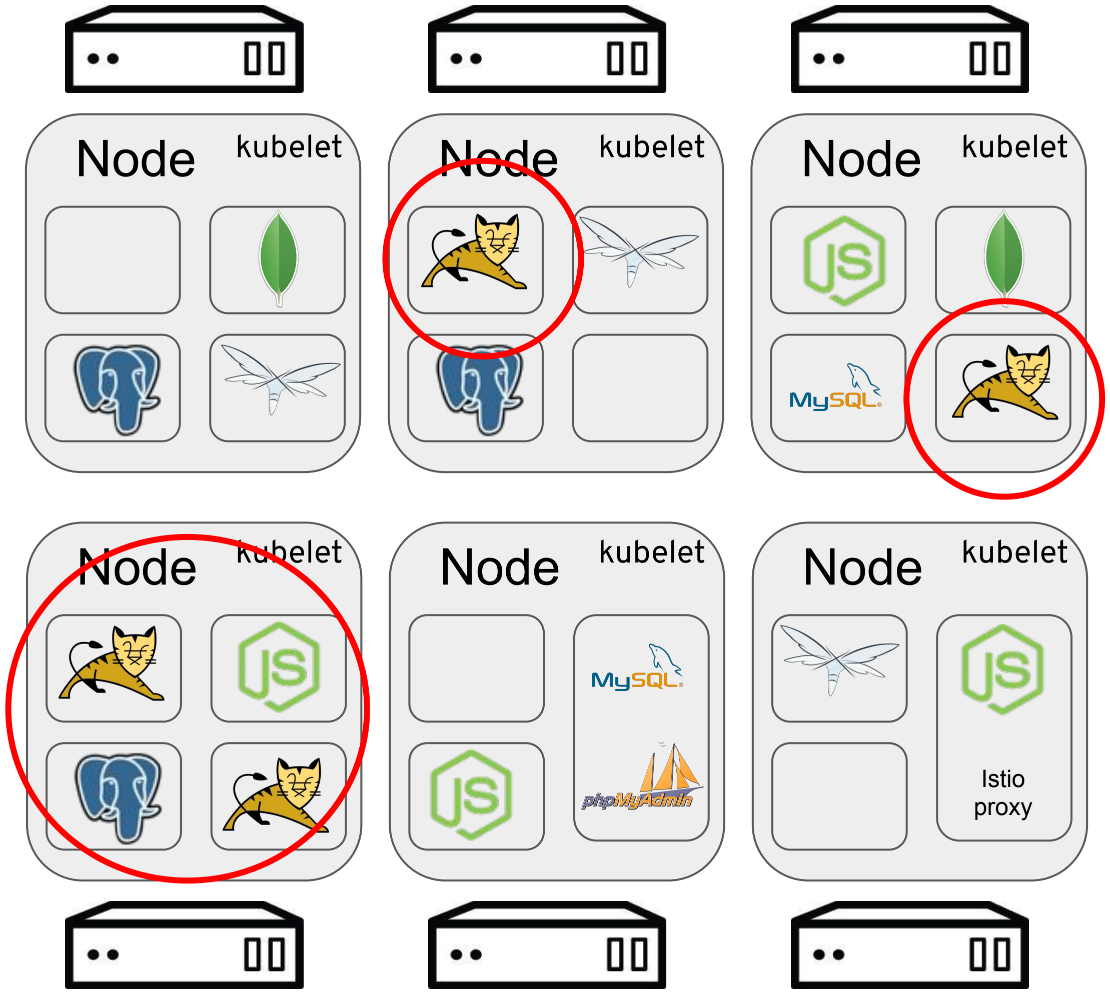
Kubernetes Cluster - 4 Tomcats



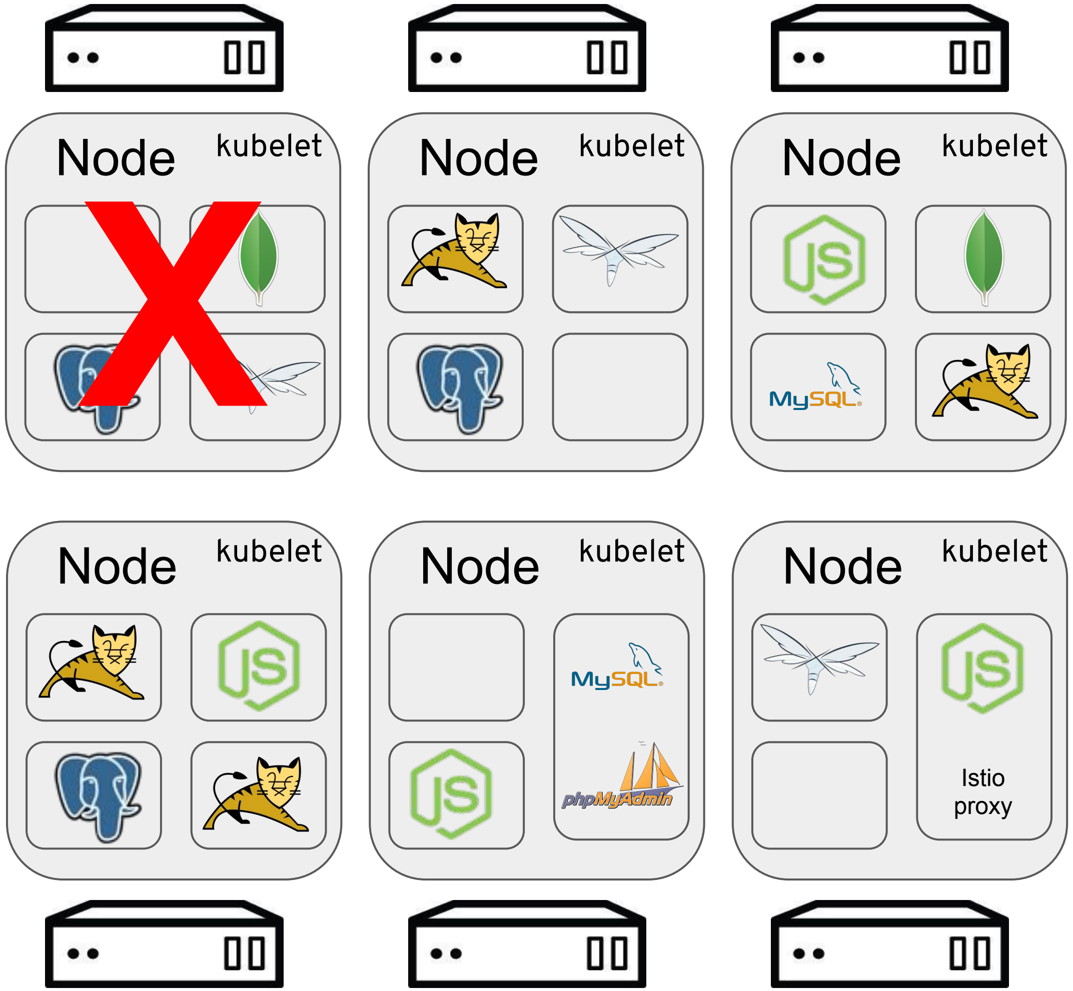
Kubernetes Cluster - Pod Fail



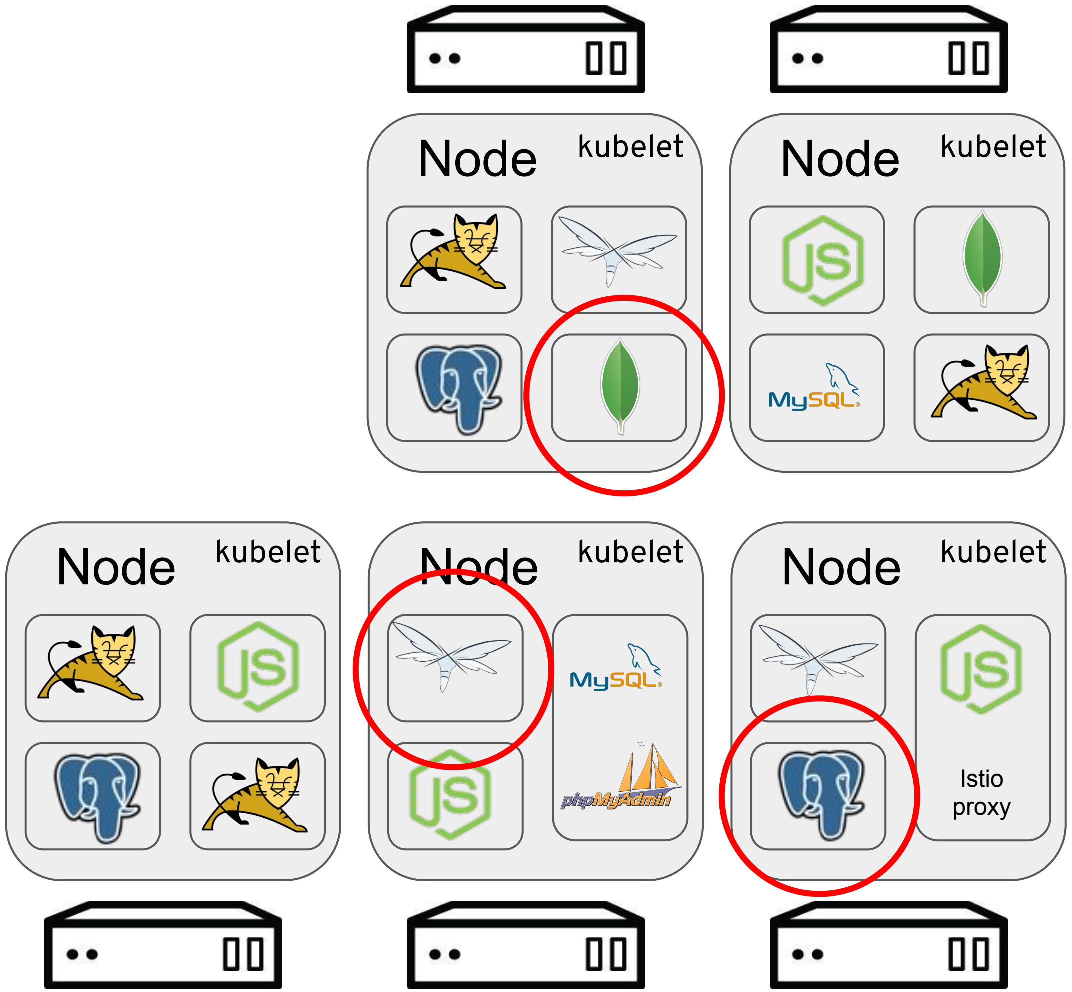
Kubernetes Cluster - Correcting

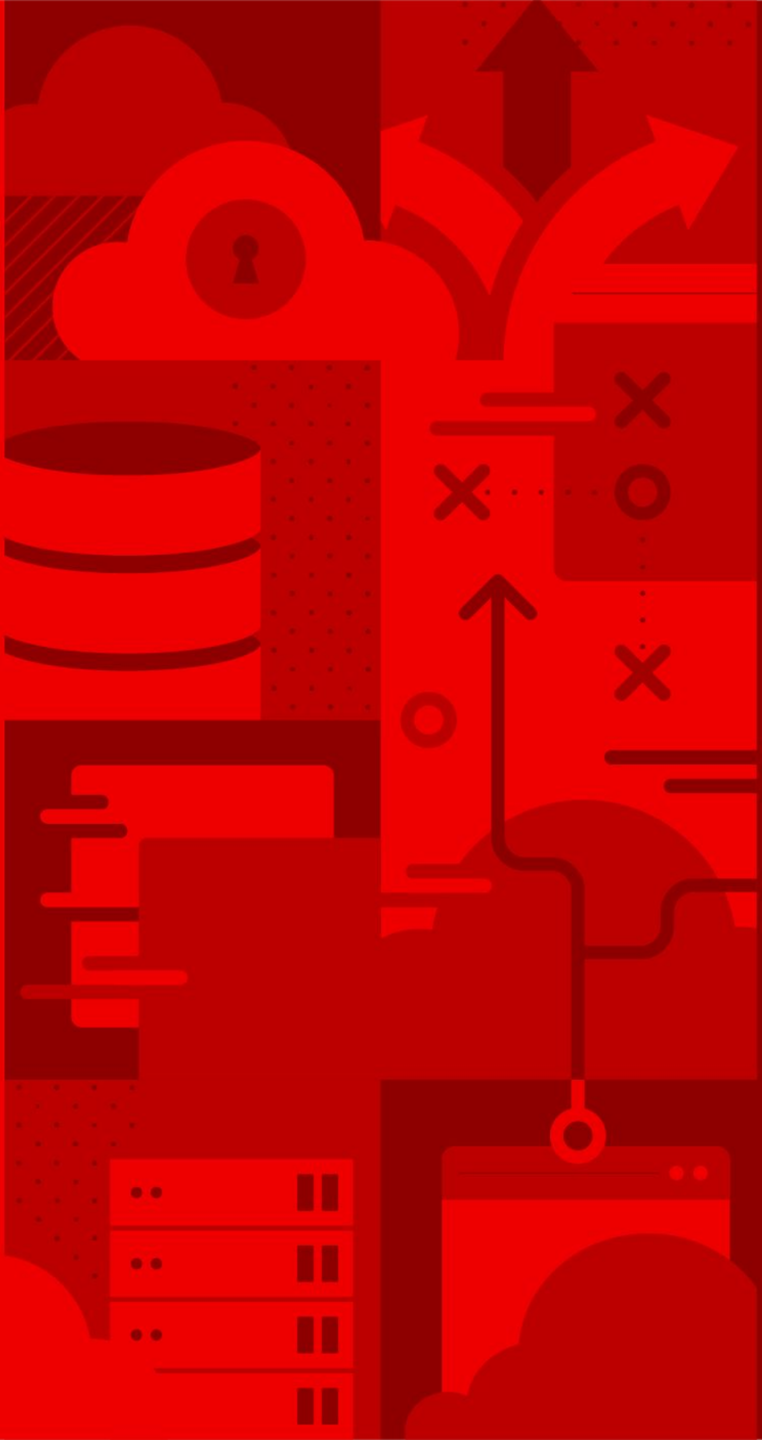


Kubernetes Cluster – Node Fail



Kubernetes Cluster - Pods Replaced



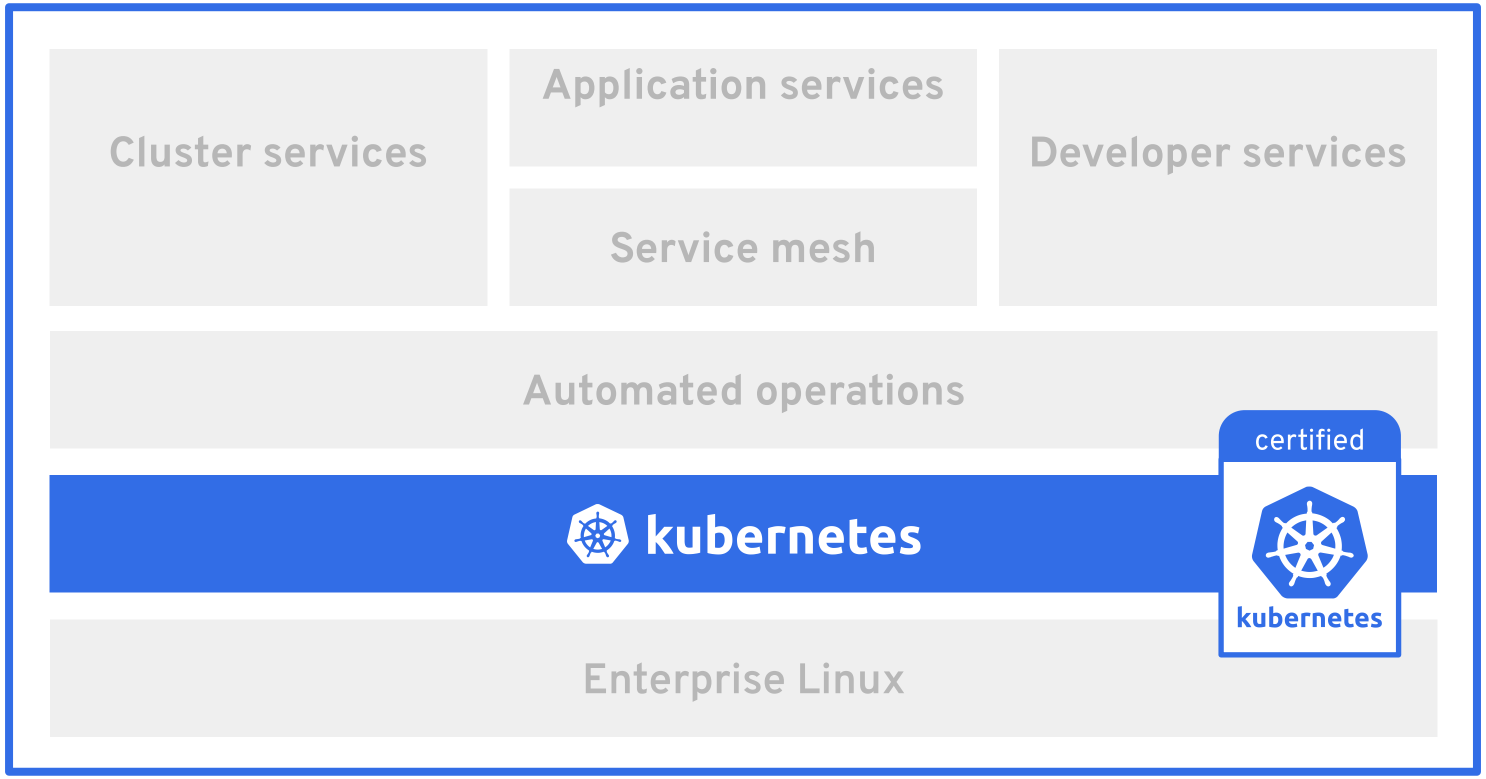


What is OpenShift?

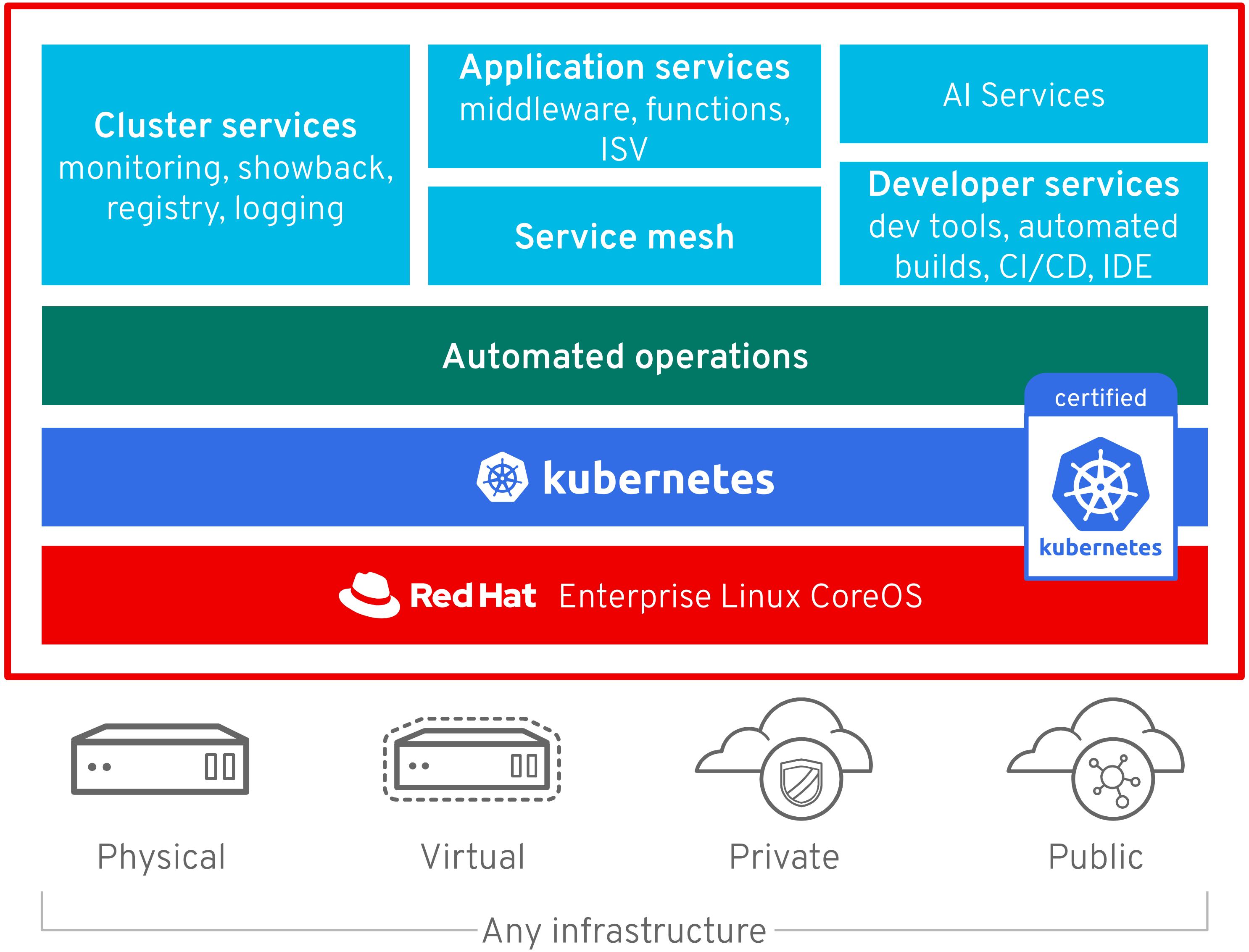
kubernetes

certified

kubernetes


Cluster services
Application services
Service mesh
Developer services
Automated operations
certified
kubernetes
kubernetes
Enterprise Linux

Red Hat OpenShift



Cluster services monitoring, showback, registry, logging

Application services middleware, functions, ISV

Service mesh

AI Services

Developer services dev tools, automated builds, CI/CD, IDE

Automated operations

certified kubernetes

kubernetes

Red Hat Enterprise Linux CoreOS

Physical

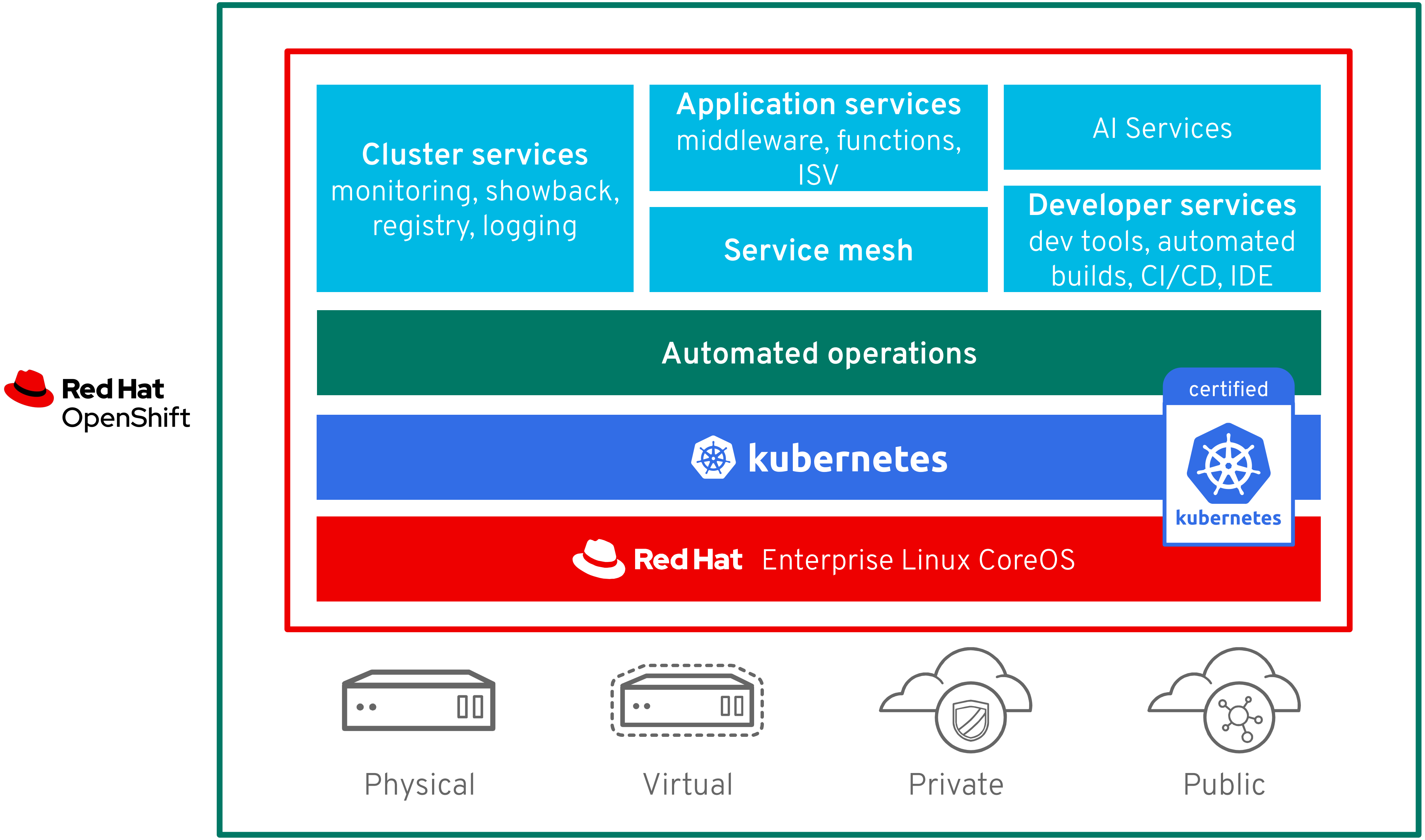
Virtual

Private

Public

Any infrastructure

Enterprise Support and Security



Red Hat OpenShift

Cluster services
monitoring, showback, registry, logging

Application services
middleware, functions, ISV

Service mesh

AI Services

Developer services
dev tools, automated builds, CI/CD, IDE

Automated operations

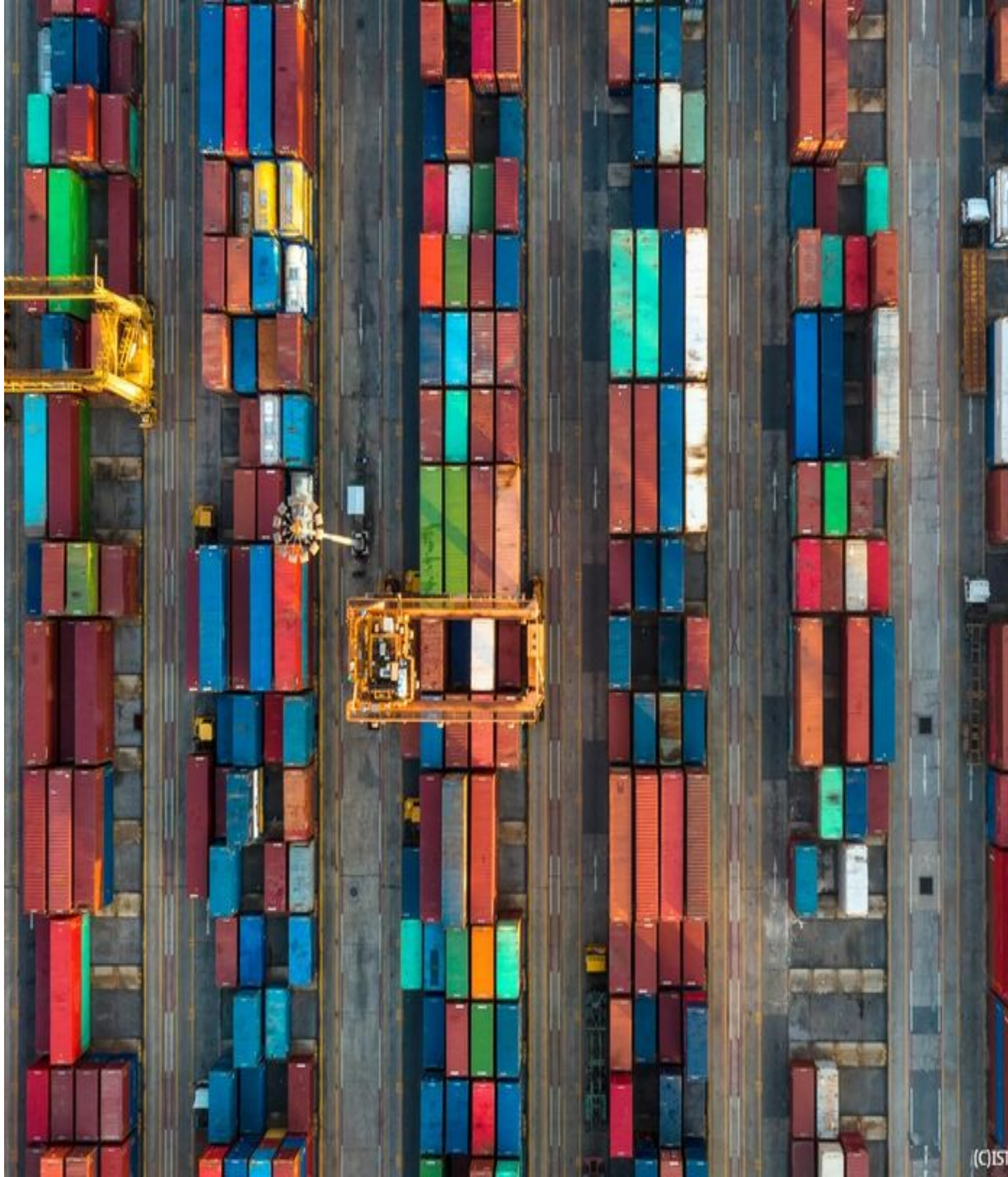
kubernetes

certified kubernetes

Red Hat Enterprise Linux CoreOS

What is a Container ?

We will be using Containers in our demo



a container is the smallest compute unit

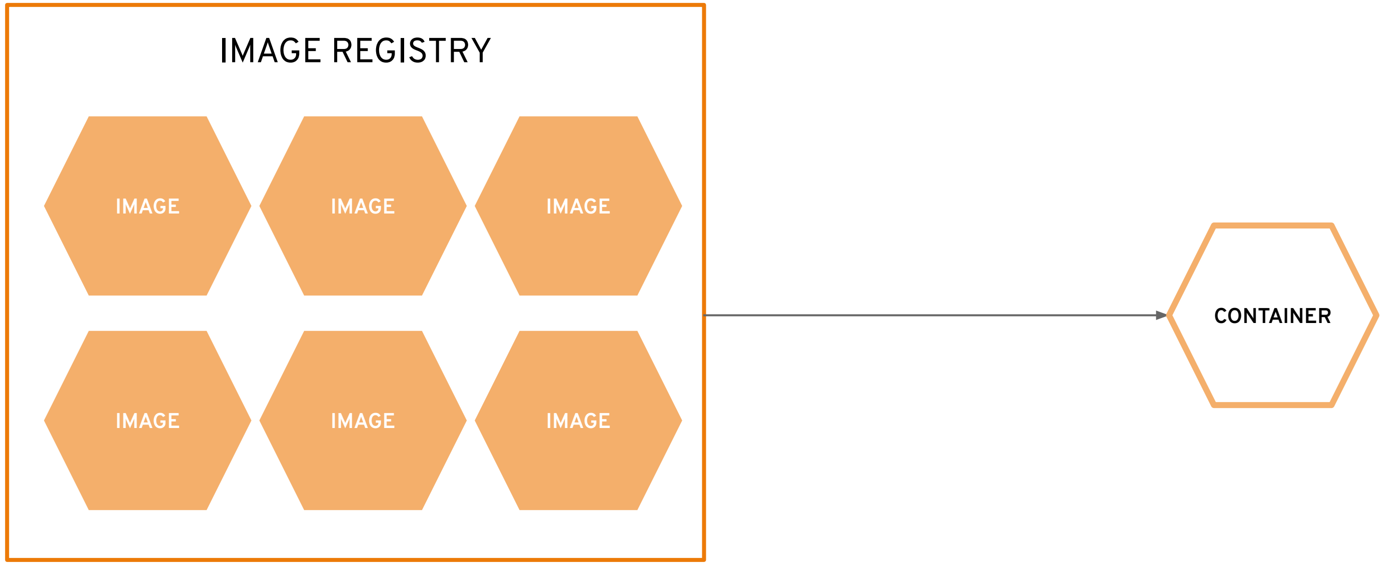
Anatomy of a Dockerfile

```
FROM registry.access.redhat.com/ubi8/ubi

ENV foo=text

RUN dnf install -y java-11-openjdk

ADD my-app.jar /home/my-app.jar

EXPOSE 8080

CMD java -jar /home/my-app.jar
```

1. Inherit from a base image
2. Parameters as environment variables
3. Install dependencies (tooling from base image)
4. Add your app as a new Layer
5. Expose the port your app will use
6. Run the app

Example for Java app

container images are stored in an image registry



an image repository contains all versions of an image in the image registry

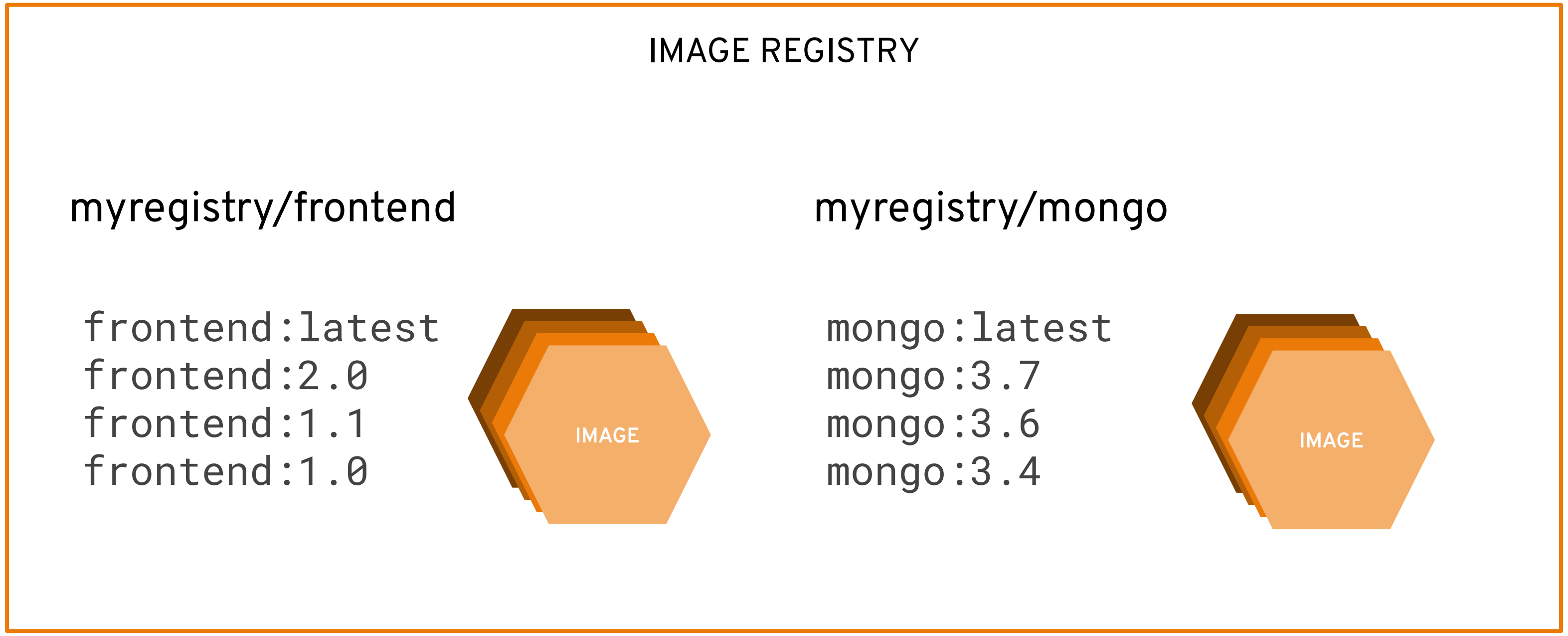


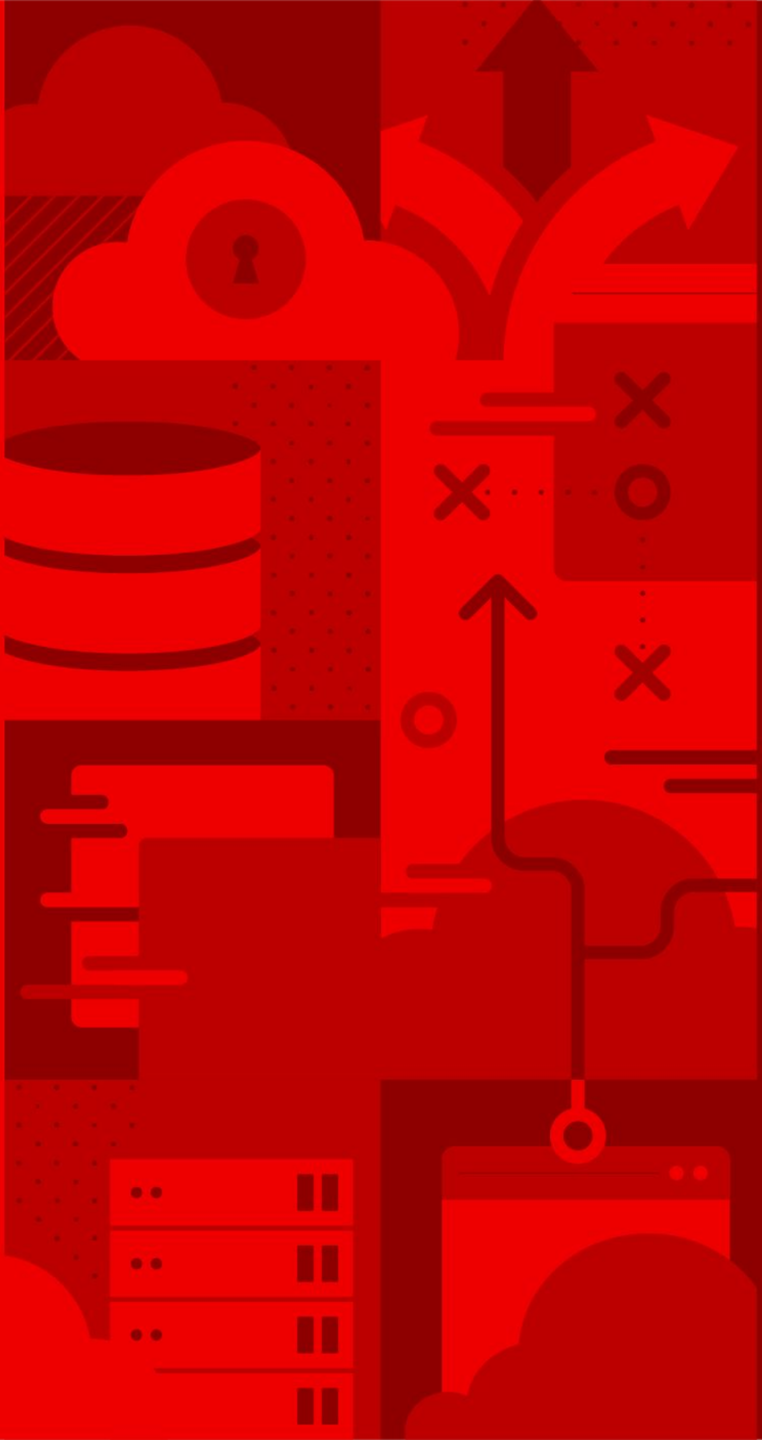
IMAGE REGISTRY

myregistry/frontend

```
frontend:latest
frontend:2.0
frontend:1.1
frontend:1.0
```

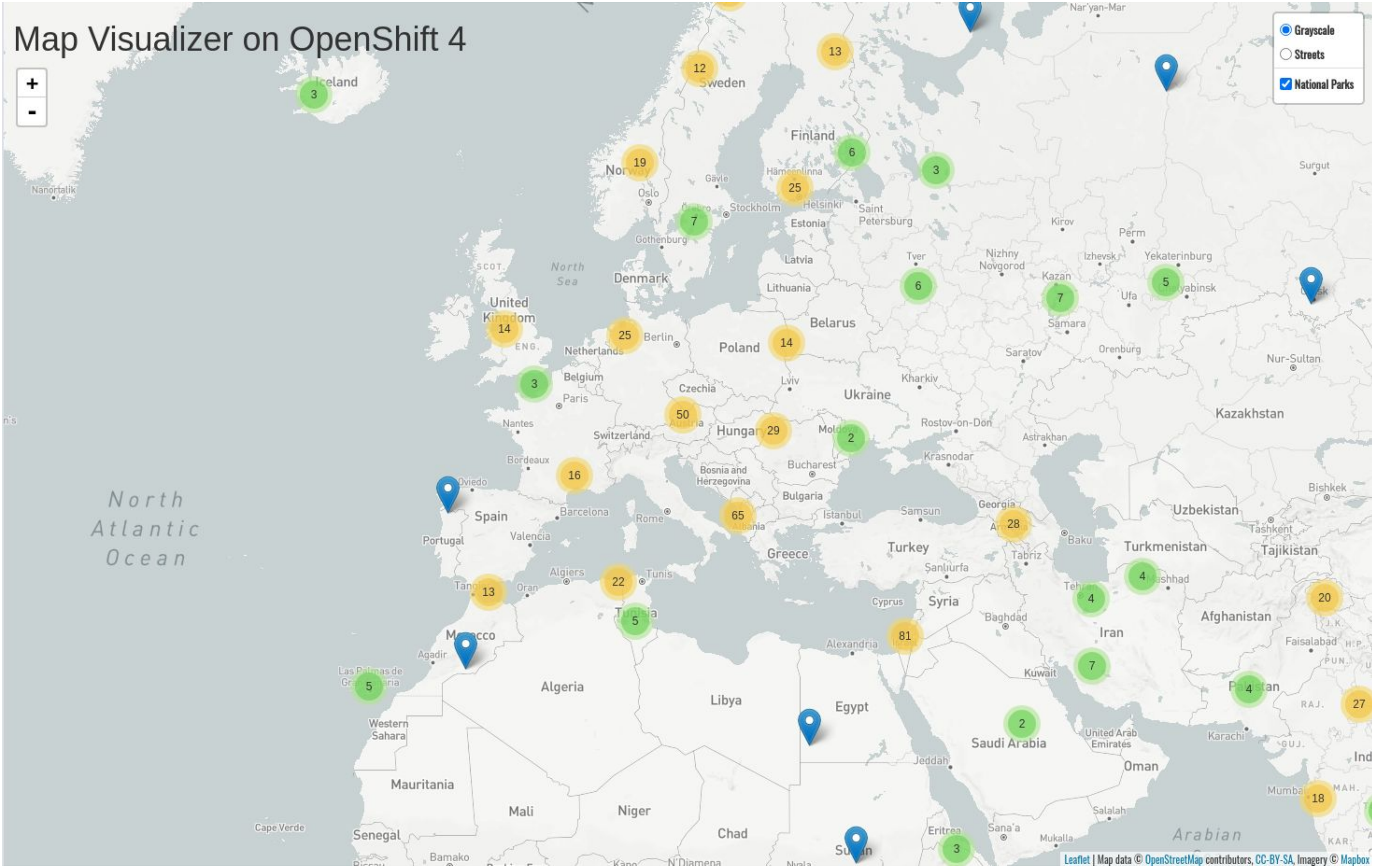
myregistry/mongo

```
mongo:latest
mongo:3.7
mongo:3.6
mongo:3.4
```



Demo OpenShift

Map that shows National Parks



Map Visualizer on OpenShift 4

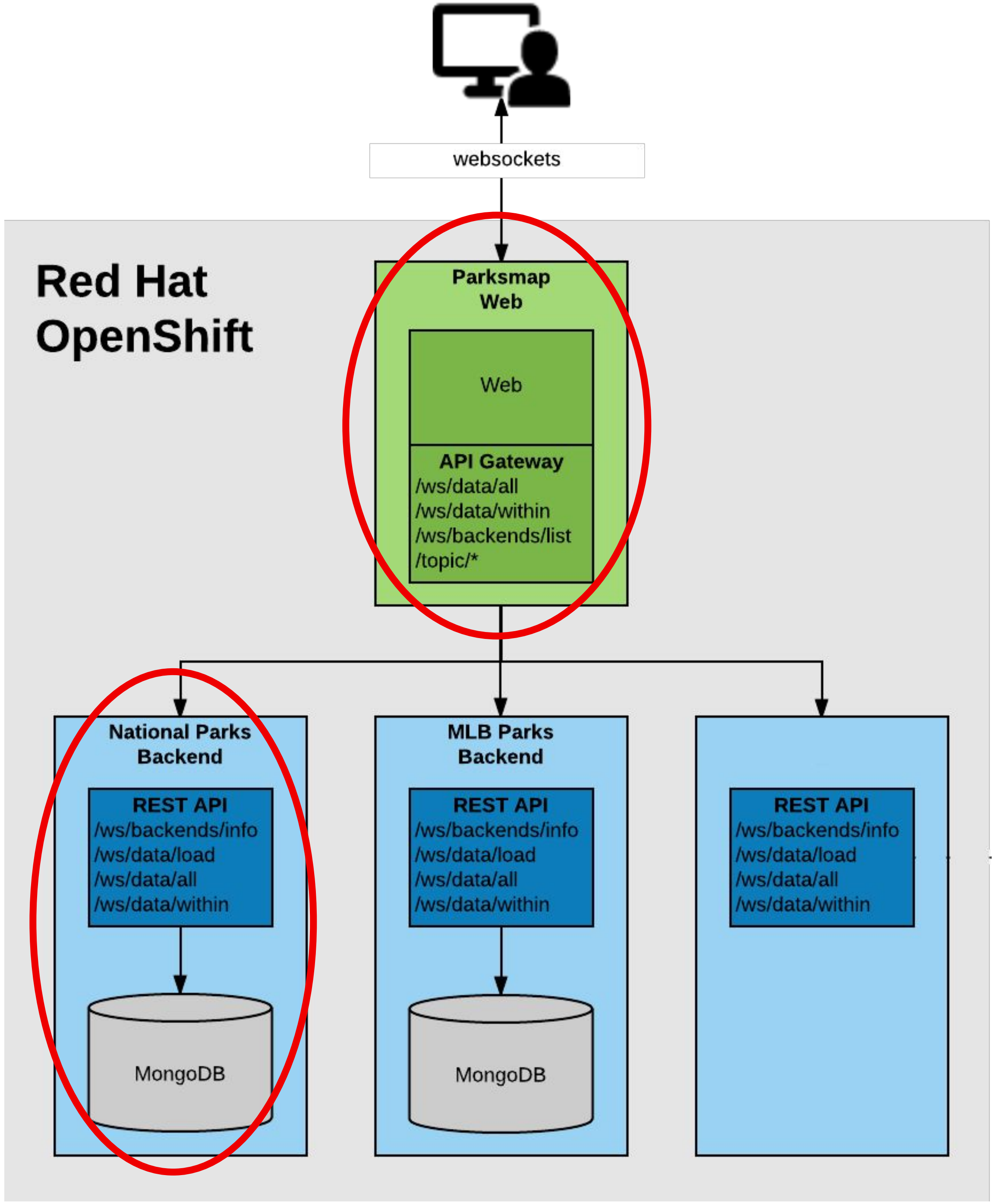
● Grayscale
○ Streets
☑ National Parks

Leaflet | Map data © OpenStreetMap contributors, CC-BY-SA, Imagery © Mapbox

Parksmap Architecture

Has 3 components:

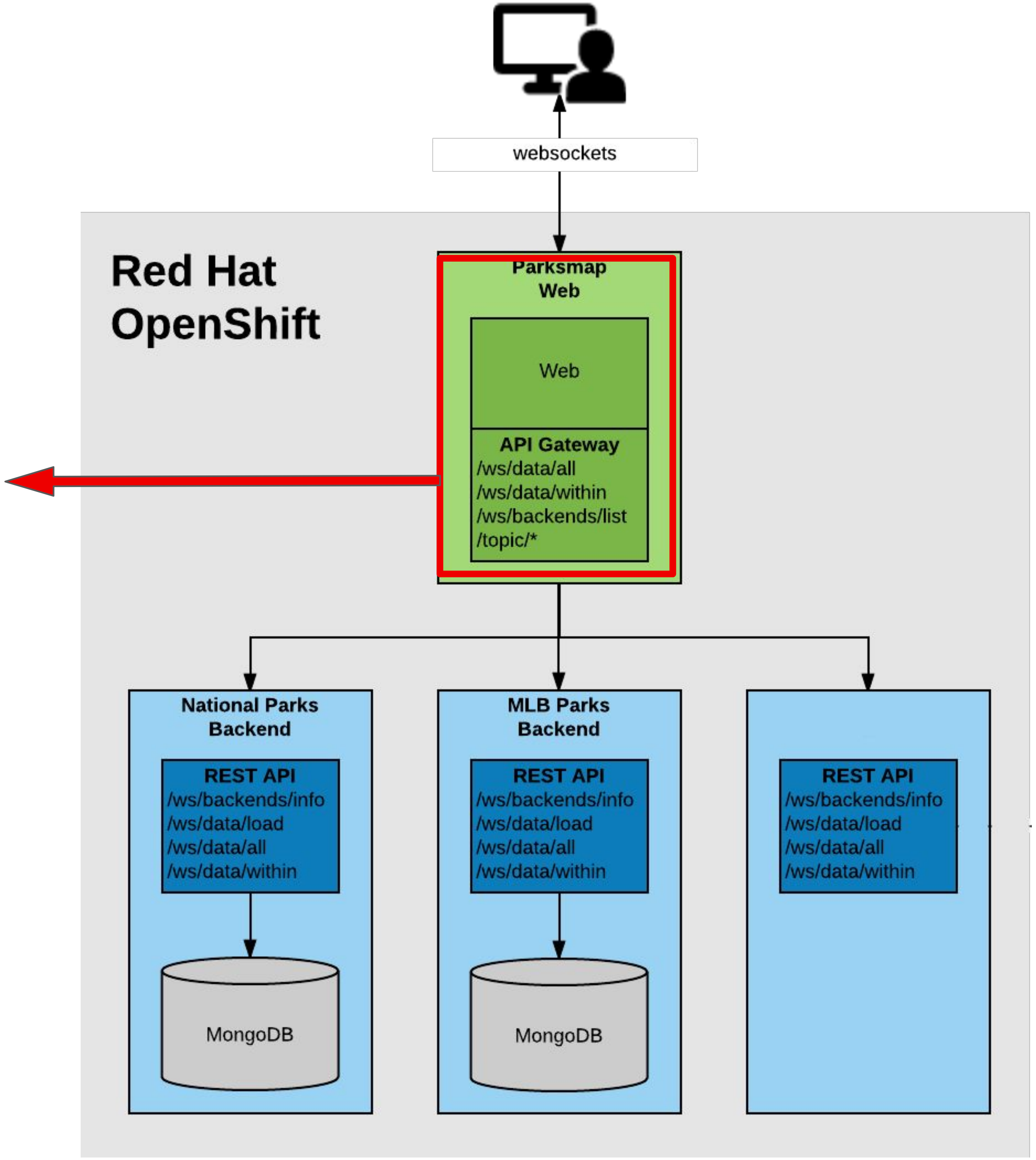
1. Frontend
2. Backend
3. Database



Parksmap Architecture

1- Frontend:

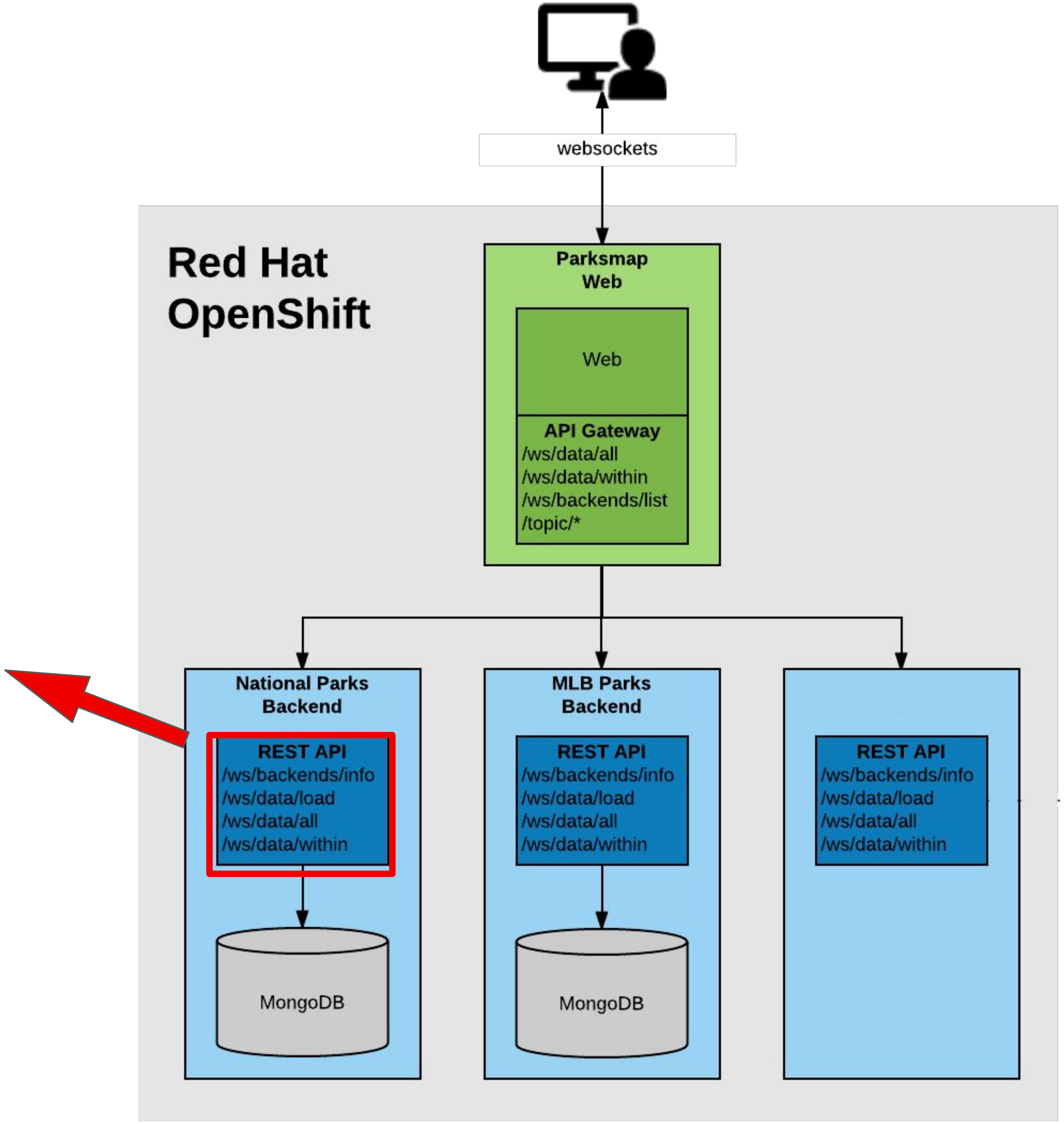
- Spring boot frontend using Mapbox Javascript API to display a World map with data points
- Labels:
  - app=workshop
  - component=parksmap
  - role=frontend



Parksmap Architecture

2 - Backend:

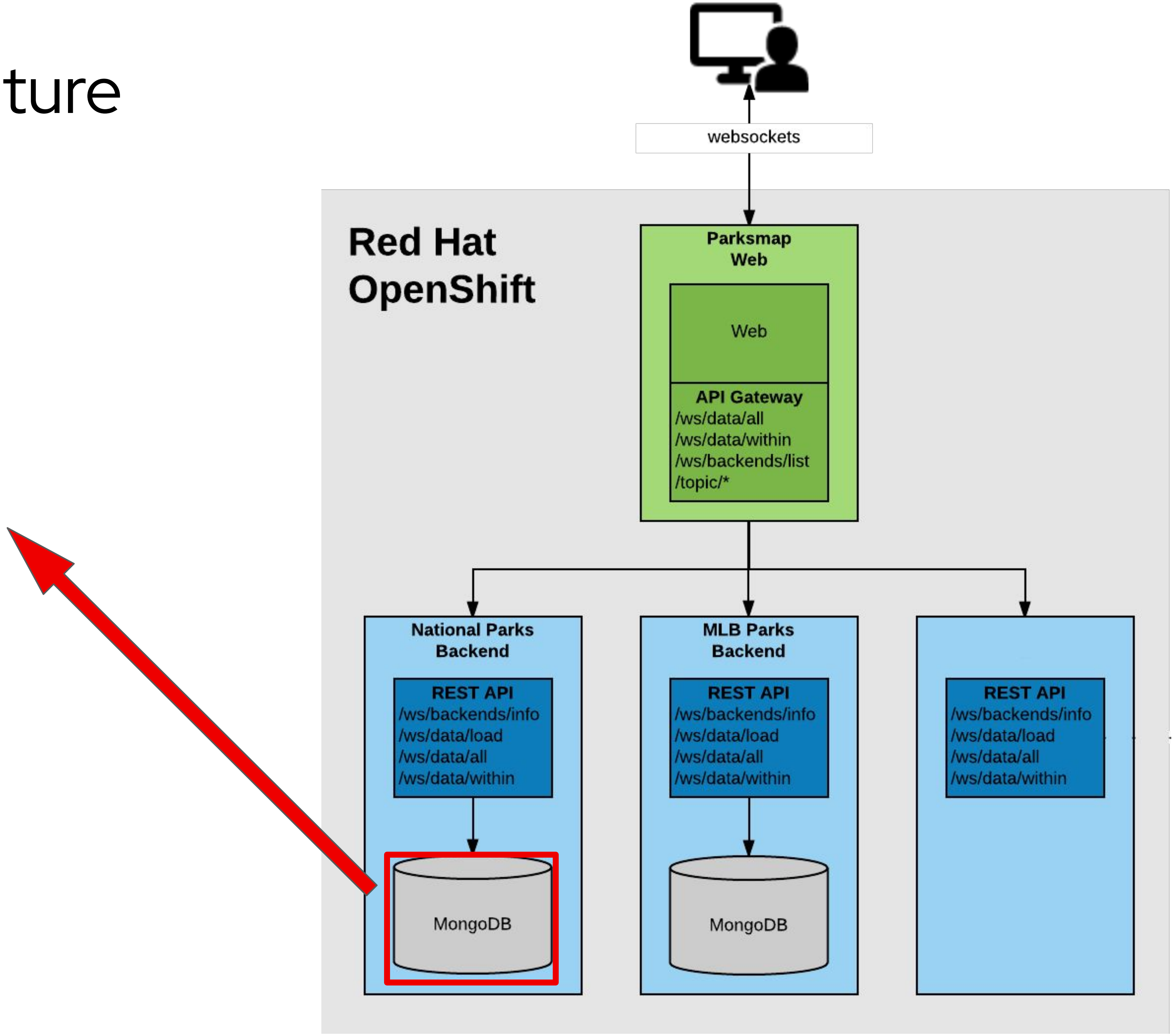
- Backend to show worldwide National Parks
- Using MongoDB Database to save and retrieve data as geo locations
- Exposes REST APIs for Parksmap frontend
- Labels
  - app=workshop
  - component=nationalparks
  - role=backend



Parksmap Architecture

2 - Database

- Mongodb
- Labels
  - app=workshop
  - component=nationalparks
  - role=database



Frontend

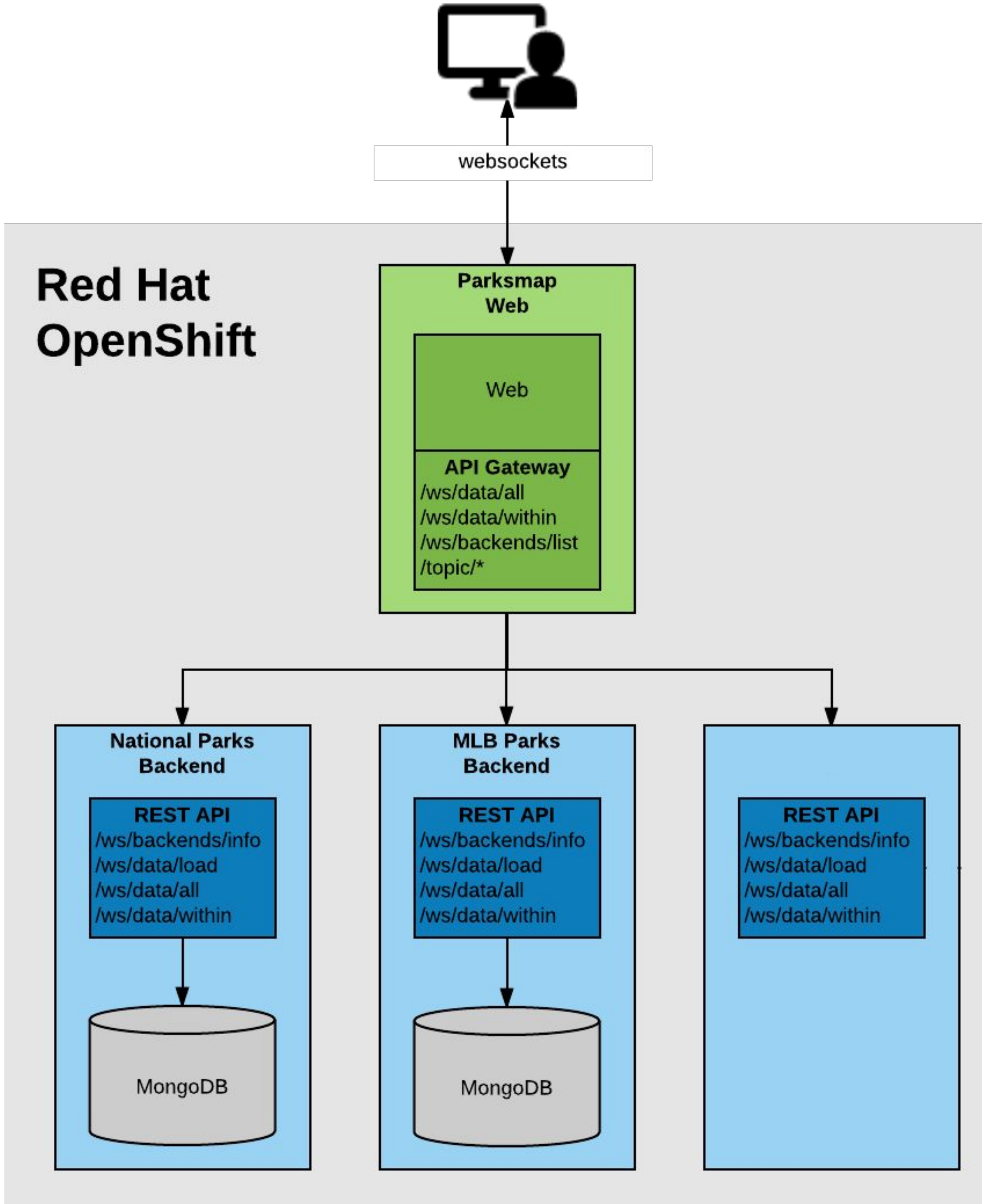
- Create from **Container image**
- Scale the application
- Self Healing
- Route
- Logs
- Permissions
- Connecting to a Container

Backend

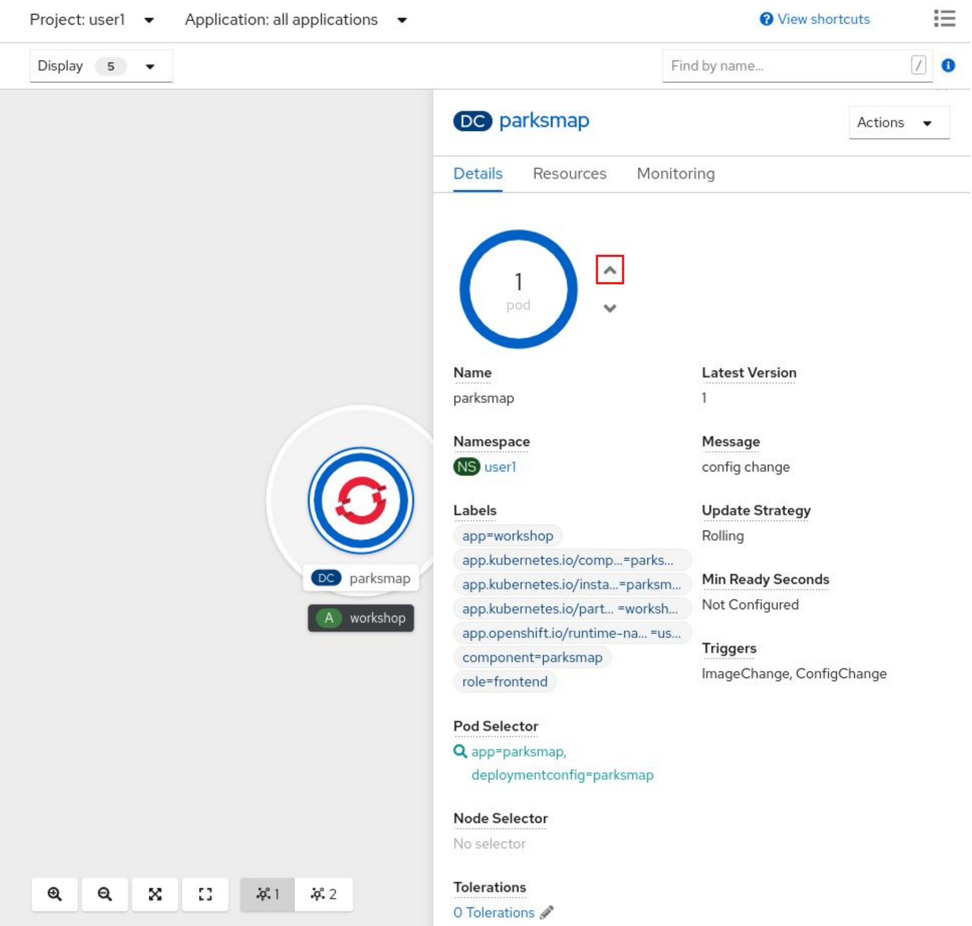
- Create from **Source to Image (s2i)**

Database

- Create from **Container image**
- Create a database user with the proper settings and roles
- Create a Secret
- Update Environment Variables for the backend
- Fix Mongodb and Backend labels
- Load data into the Db



Parksmap: Exploring OpenShift



Name
parksmap

Latest Version
1

Namespace
NS user1

Message
config change

Labels
app=workshop
app.kubernetes.io/comp...=parks...
app.kubernetes.io/insta...=parksm...
app.kubernetes.io/part... =worksh...
app.openshift.io/runtime-na... =us...
component=parksmap
role=frontend

Update Strategy
Rolling

Min Ready Seconds
Not Configured

Triggers
ImageChange, ConfigChange

Pod Selector
app=parksmap,
deploymentconfig=parksmap

Node Selector
No selector

Tolerations
0 Tolerations

- Scaling Apps
- Logging
- Labels
- Permissions
- Accessing and debugging Containers

Frontend

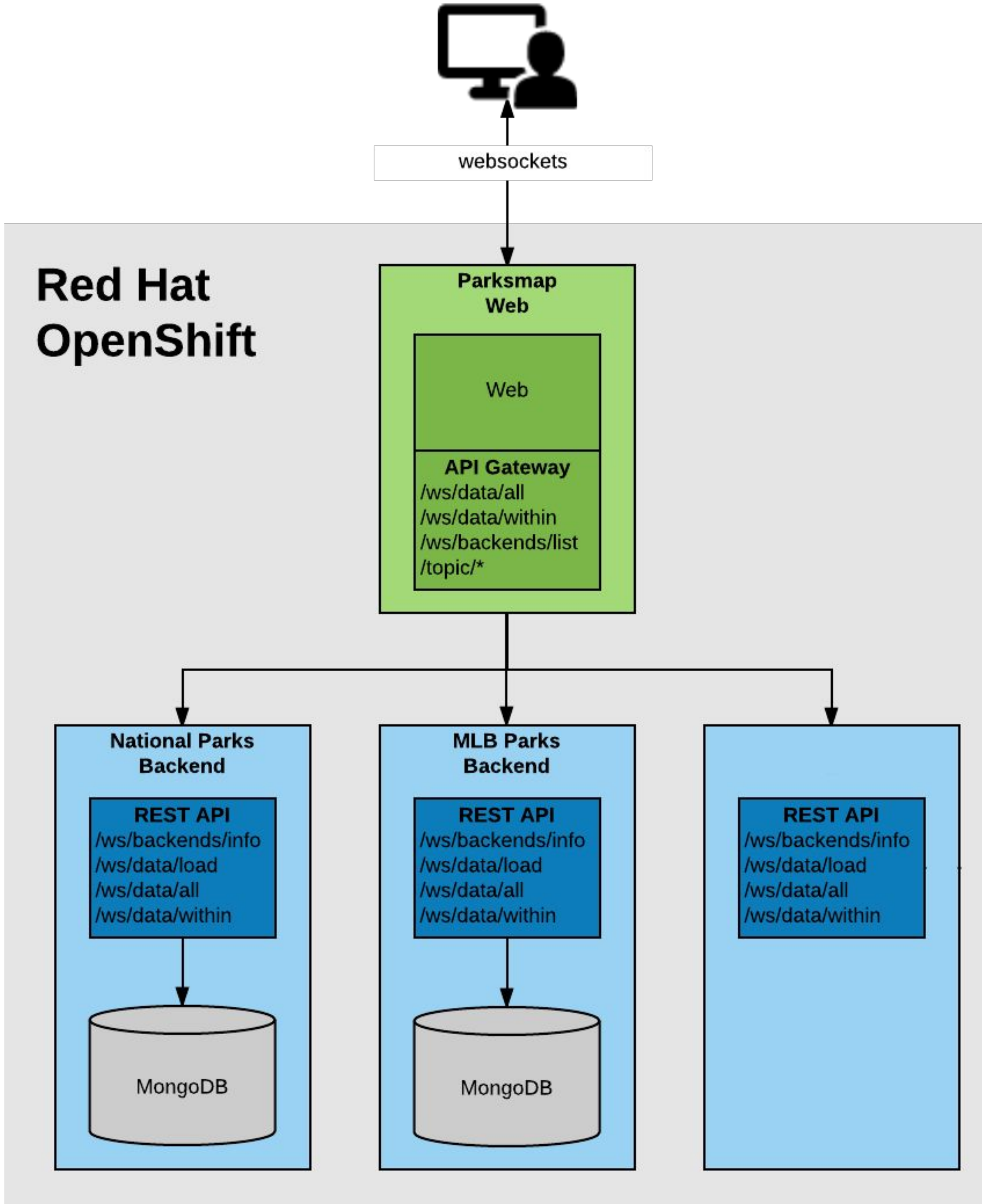
- Create from **Container image**
- Scale the application
- Self Healing
- Route
- Logs
- Permissions
- Connecting to a Container

Backend

- Create from **Source to Image (s2i)**

Database

- Create from **Container image**
- Create a database user with the proper settings and roles
- Create a Secret
- Update Environment Variables for the backend
- Fix Mongodb and Backend labels
- Load data into the Db



Frontend

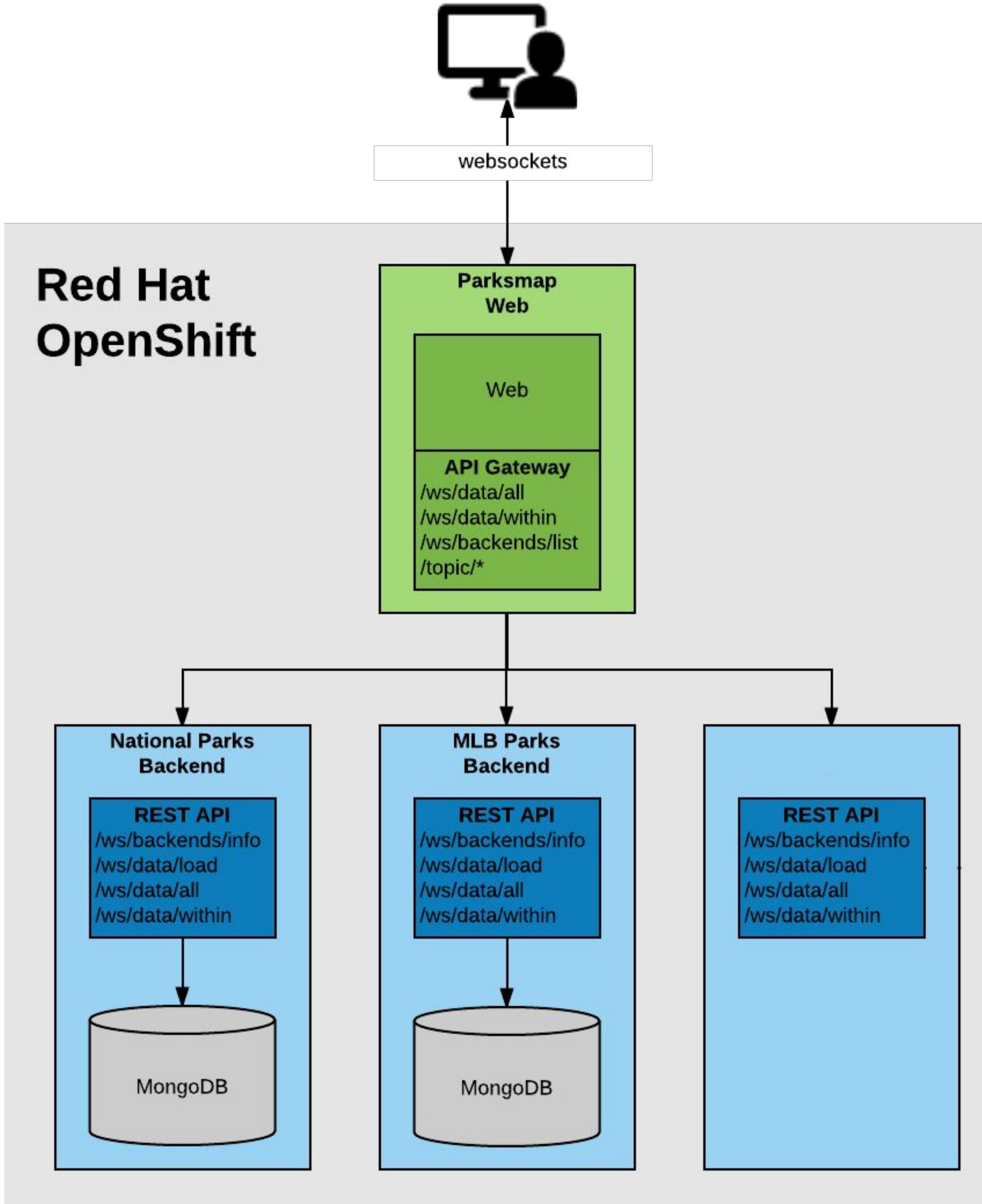
- Create from **Container image**
- Scale the application
- Self Healing
- Route
- Logs
- Permissions
- Connecting to a Container

Backend

- Create from **Source to Image (s2i)**

Database

- Create a Secret
- Create from **Container image**
- Create a database user with the proper settings and roles
- Create a Secret
- Update Environment Variables for the backend
- Fix Mongodb and Backend labels
- Load data into the Db



Frontend

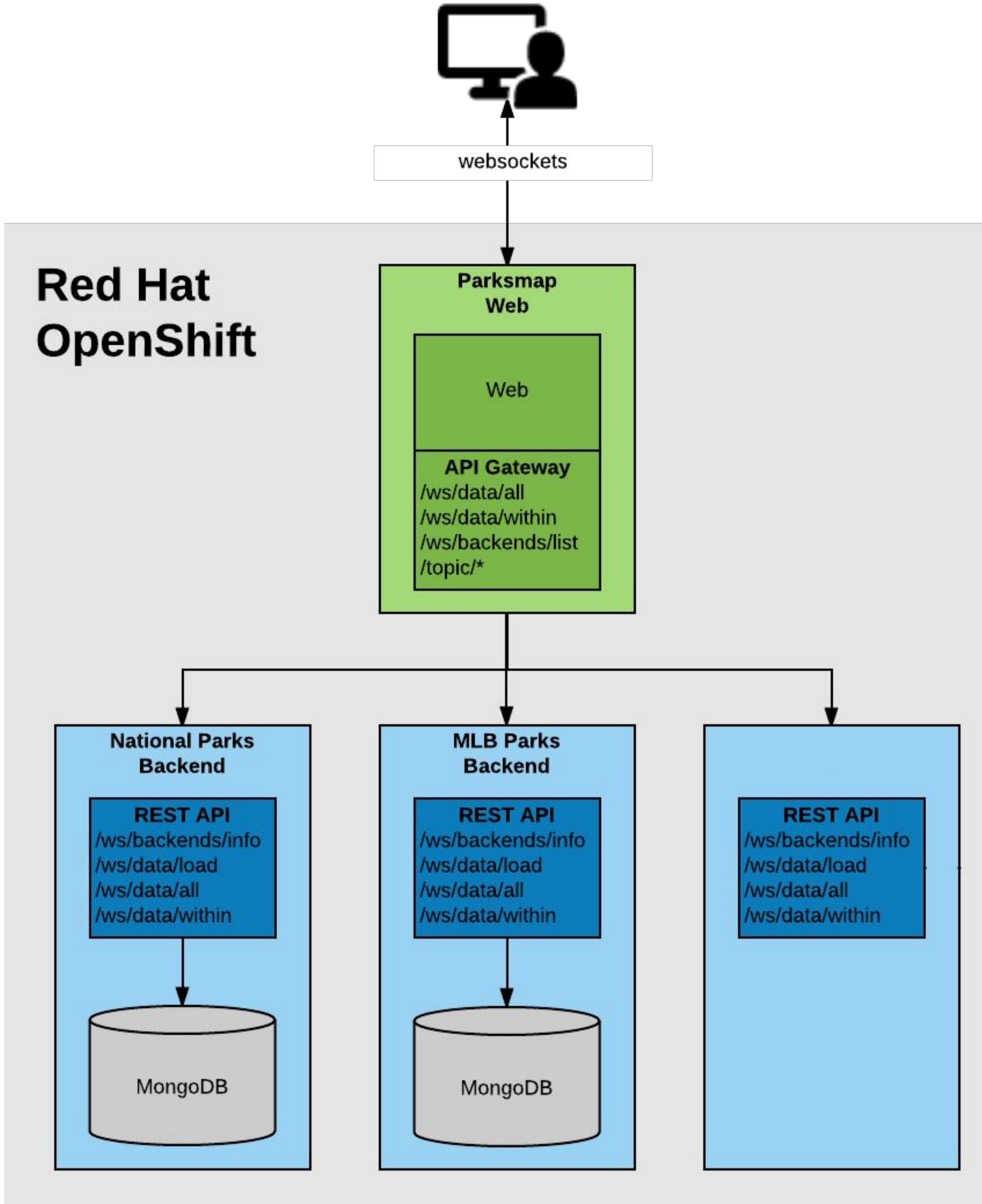
- Create from **Container image**
- Scale the application
- Self Healing
- Route
- Logs
- Permissions
- Connecting to a Container

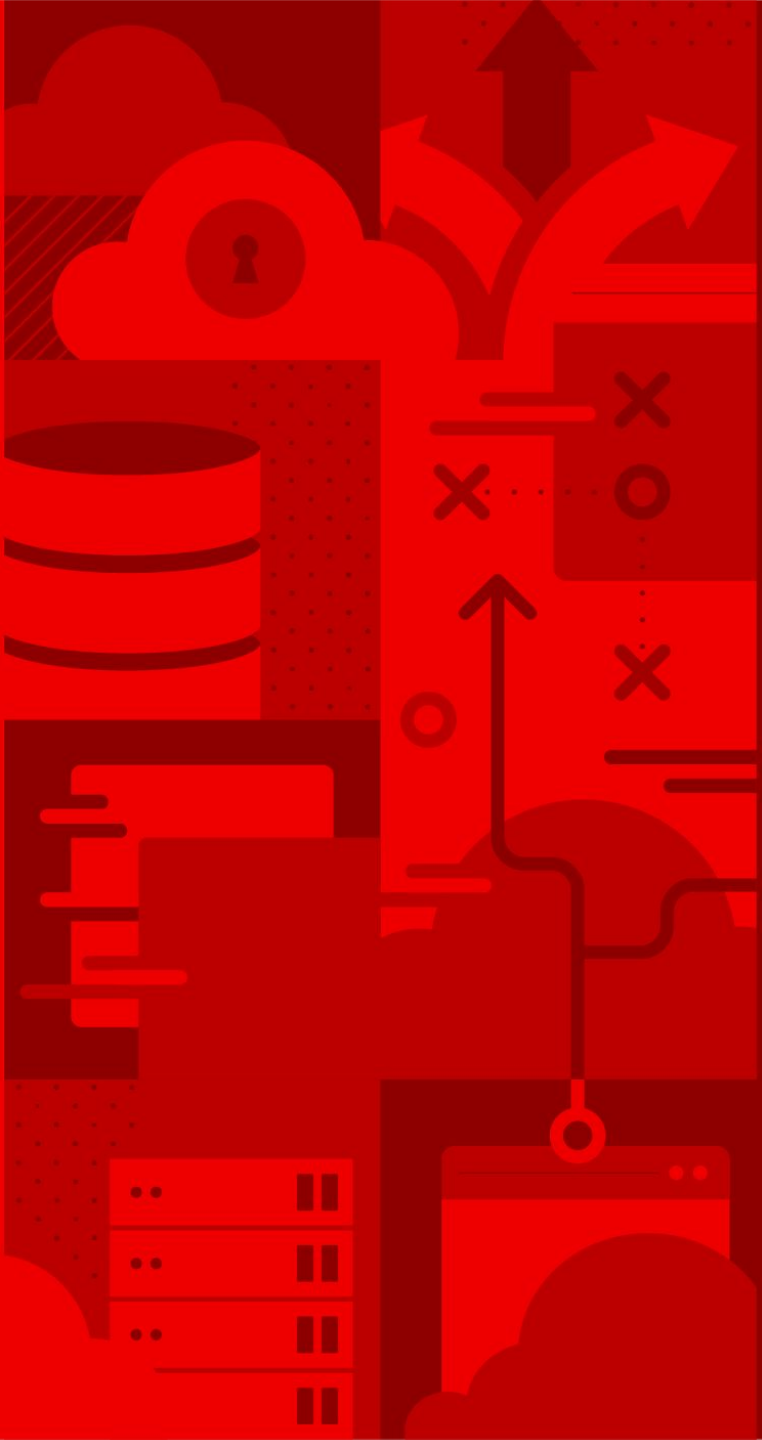
Backend

- Create from **Source to Image (s2i)**

Database

- Create from **Container image**
- Create a database user with the proper settings and roles
- Create a Secret
- Update Environment Variables for the backend
- Fix Mongodb and Backend labels
- Load data into the Db



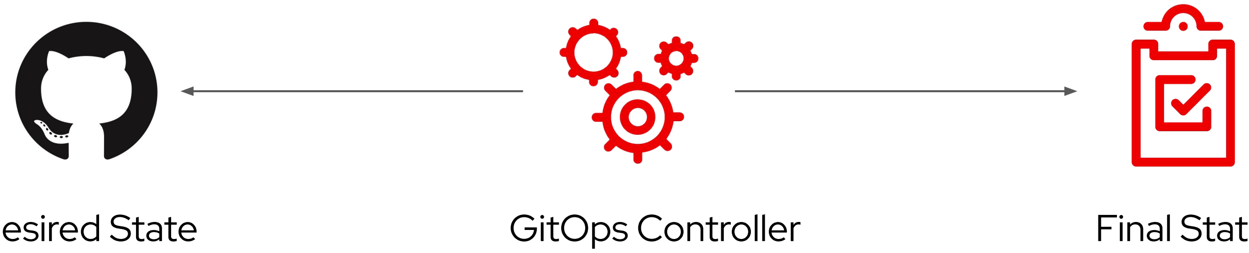


GitOps

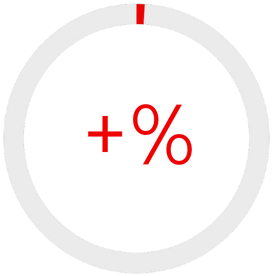
What is GitOps?

Overview

GitOps is a set of practices that leverages Git workflows to manage infrastructure and application configurations. By using Git repositories as the **source of truth**, it allows the DevOps team to store the entire state of the cluster configuration in Git so that the trail of changes are visible and auditable.



Benefits of using a GitOps Model



Deploy faster / Innovation Velocity

Developer Centric

Quick and Easy Recovery (Mean Time To Recover - MTTR)

Secure / Separation of Concerns CI - CD

Auditability / Audit Log outside of Cluster

Rollout based on PRs / Rollback with Revert

Code is Reviewed

Observability / Single Source of Truth & Detect Config Drifts

Increase Stability and Reliability

ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.

ArgoCD Kubernetes Objects Generator

Manifests and third-party integrations

Helm

Helm uses a packaging format called charts. A chart is a collection of files that describe a related set of Kubernetes resources

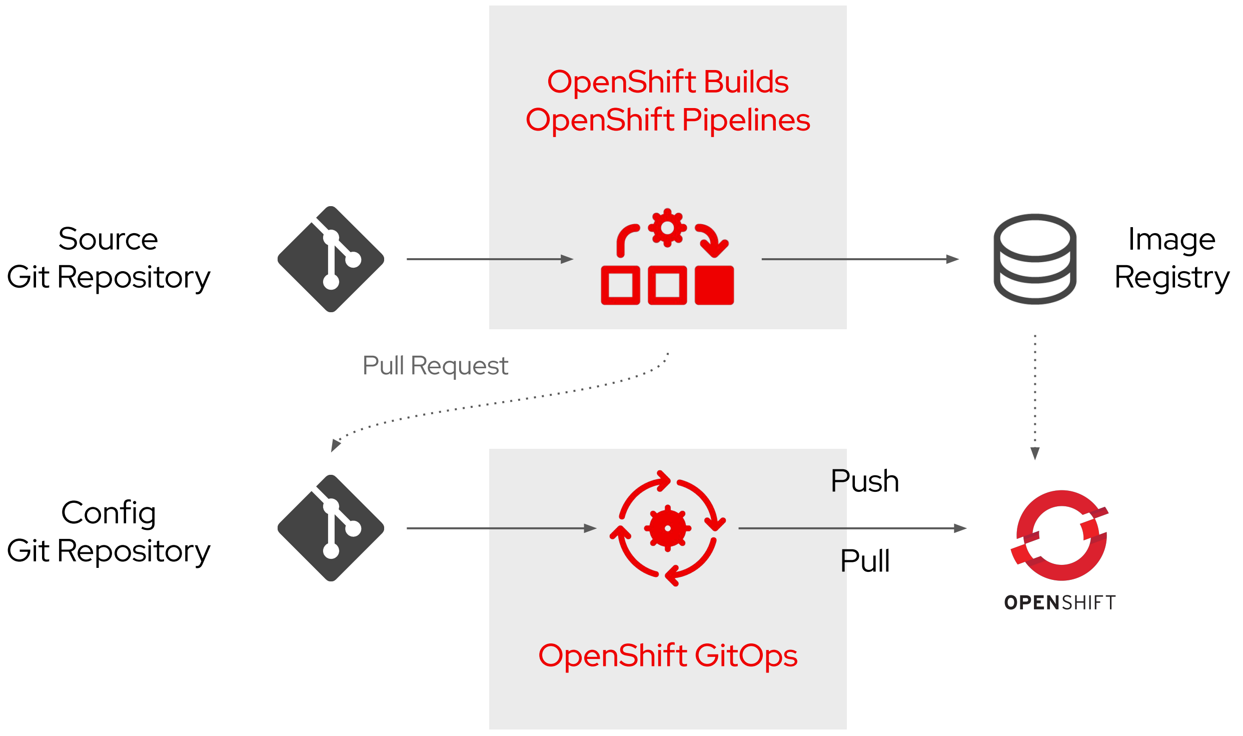
Kustomize

Template-free way to customize application configuration that simplifies the use of off-the-shelf applications

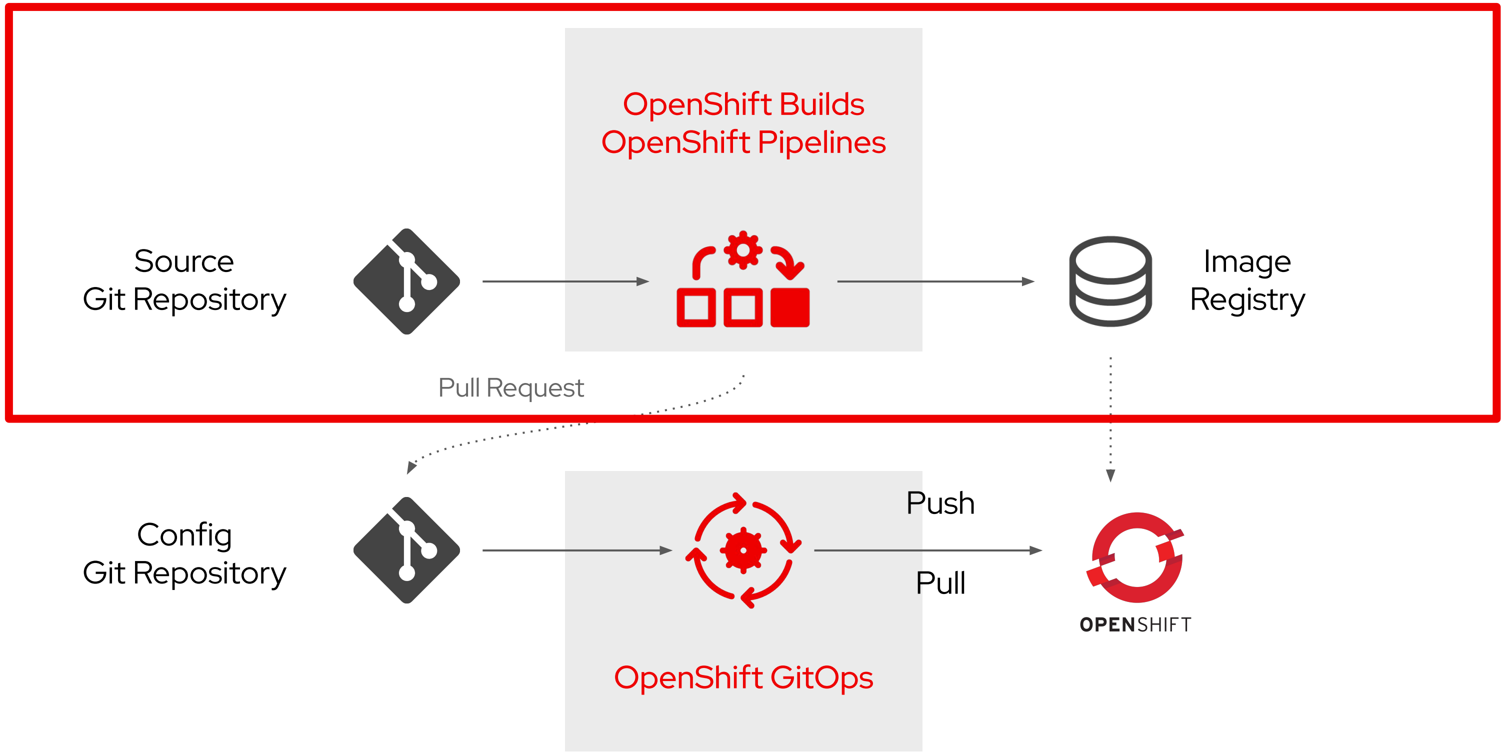
Kubernetes Manifests

Plain text kubernetes object located in YAML or JSON format

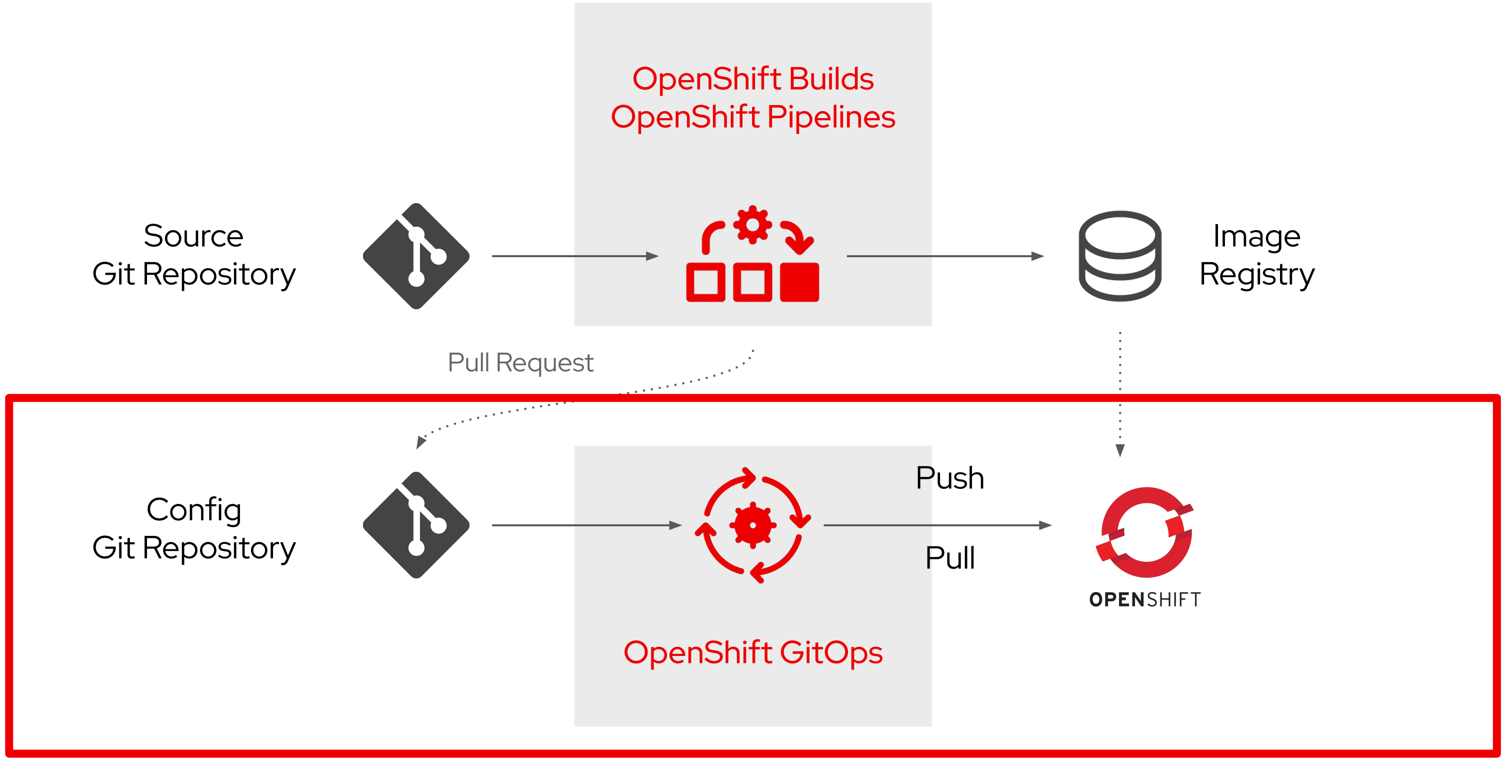
Application Delivery Model on OpenShift

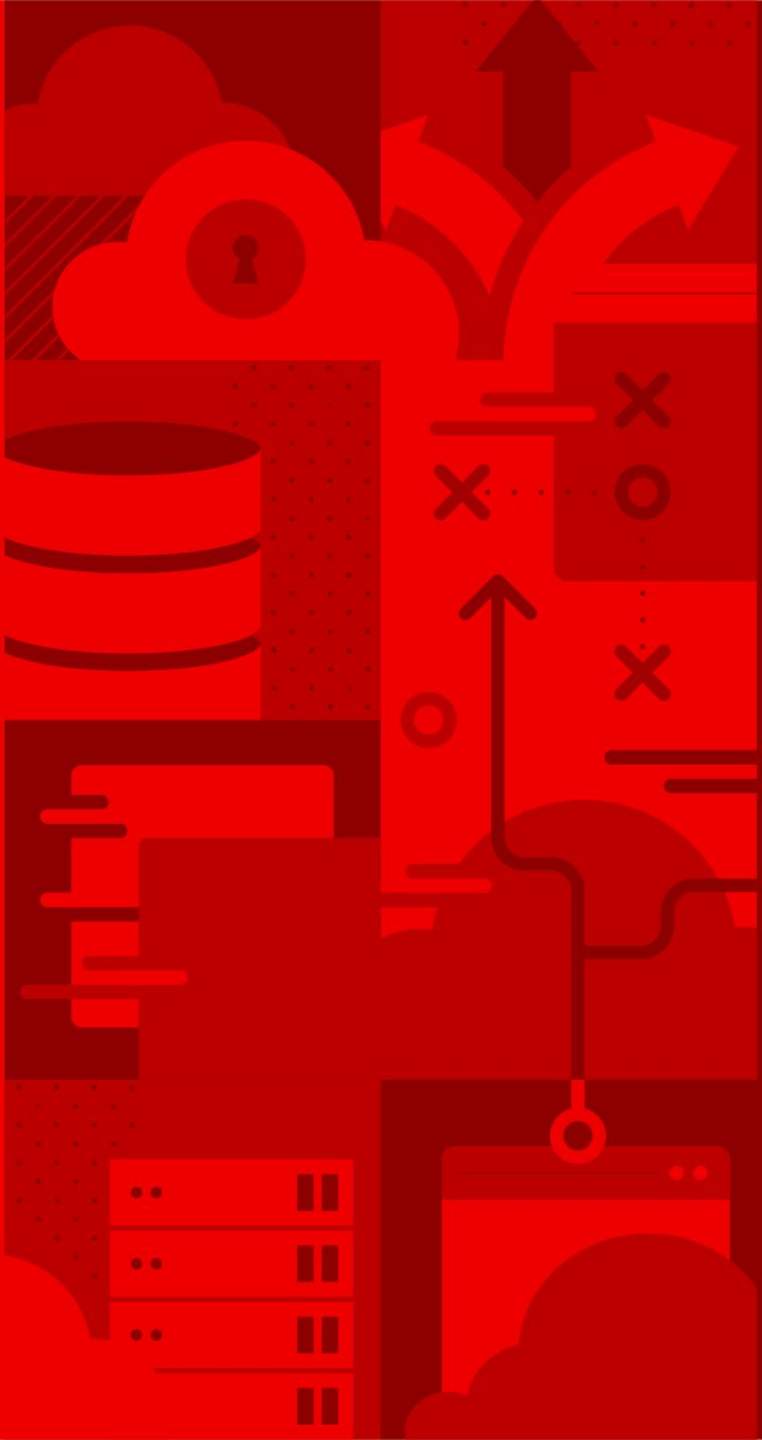


Continuous Integration



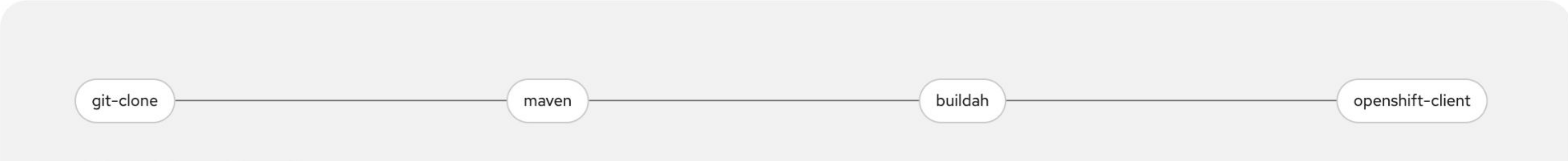
Continuous Deployment



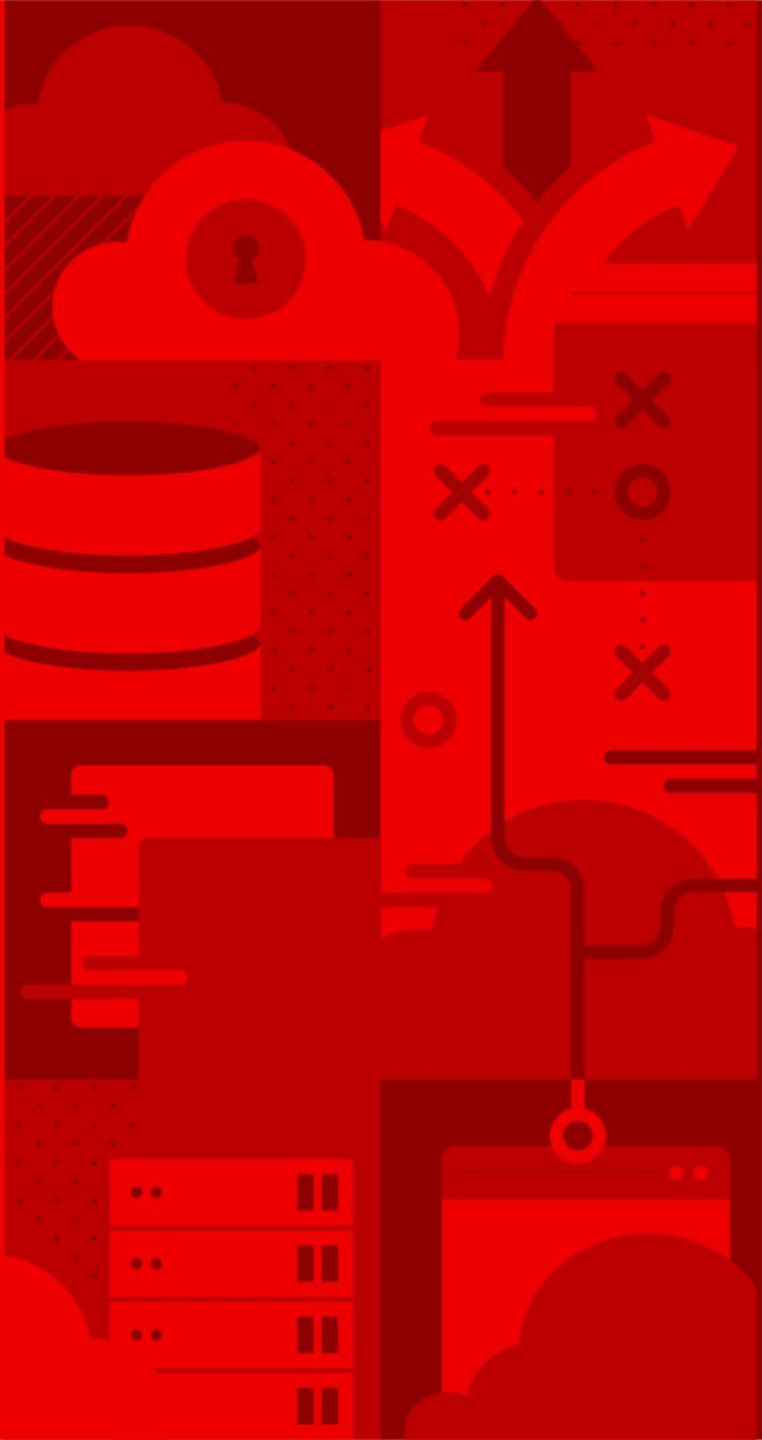


Quickly show a Pipeline

Tekton Pipeline - Build and Deploy

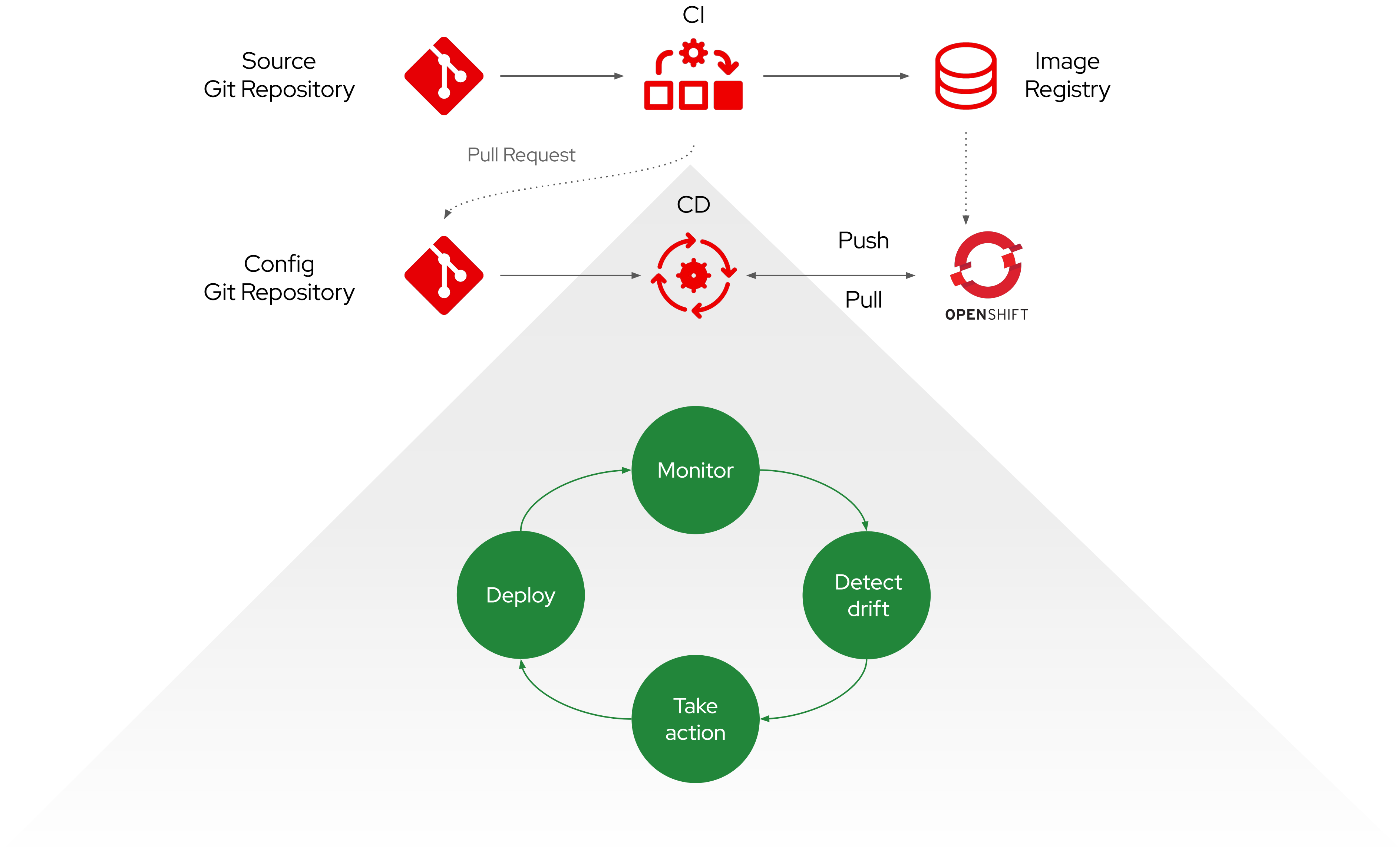


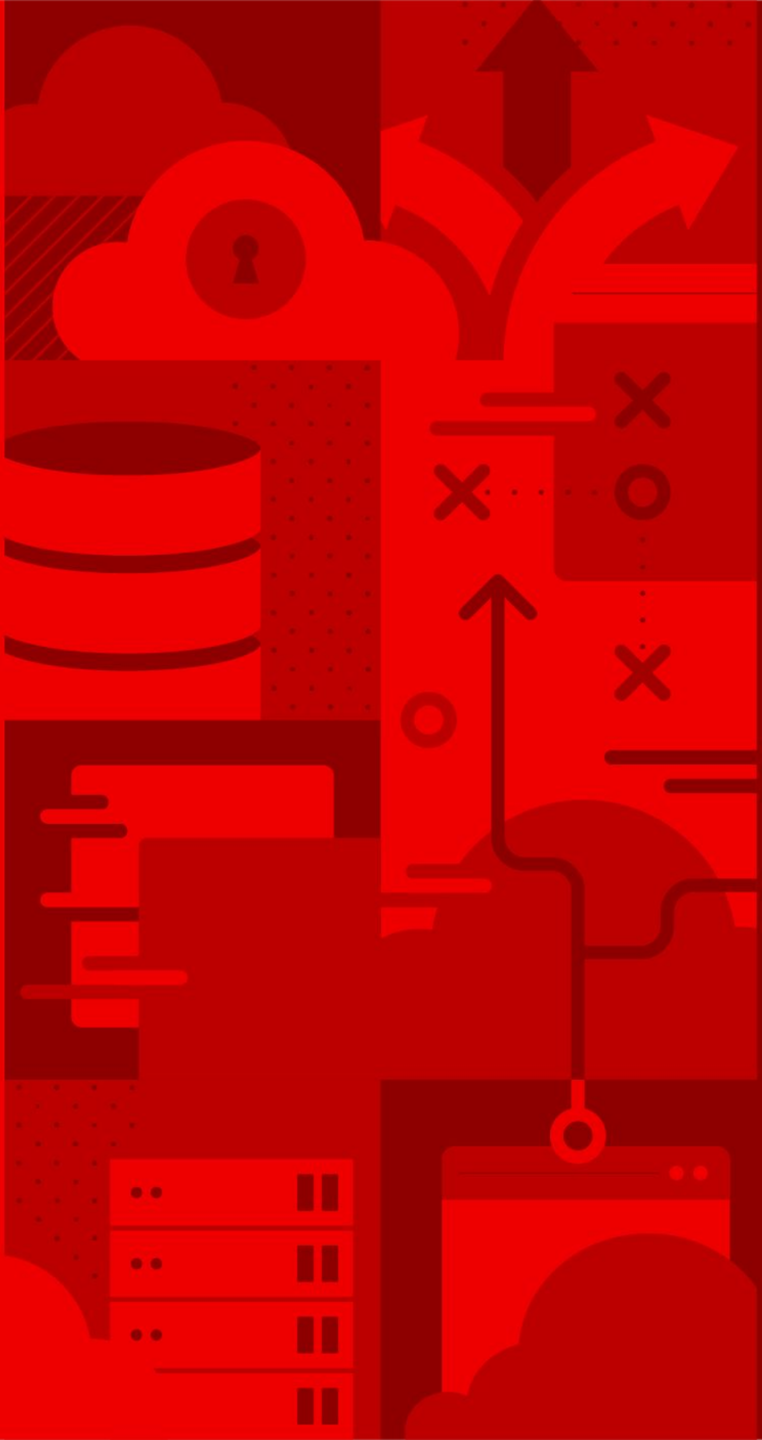
1. Clone the Repo
2. Build the app
3. Create a container & push to internal Openshift repository
4. Deploy the container



Back to GitOps

The GitOps Application Delivery Model





Demo GitOps

GitOps to deploy Everything

Frontend

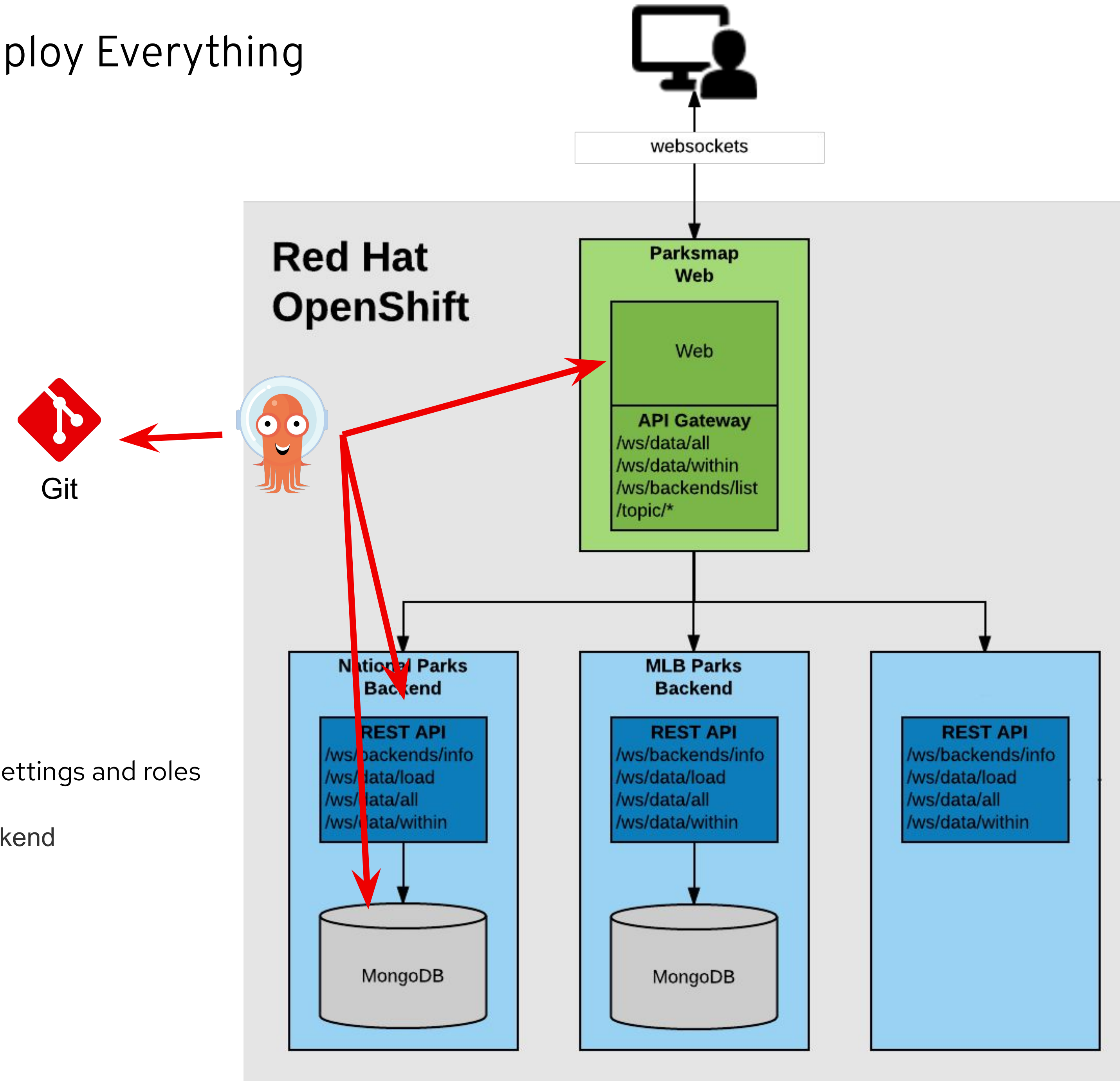
- Create from **Container image**
- Scale the application
- Self Healing
- Route
- Logs
- Permissions
- Connecting to a Container

Backend

- Create from **Source to Image (s2i)**

Database

- Create from **Container image**
- Create a database user with the proper settings and roles
- Create a Secret
- Update Environment Variables for the backend
- Fix Mongodb and Backend labels
- Load data into the Db



Summary

OpenShift

- Dashboard
- Create from
  - **Container image**
  - **S2i**
- Scale the application
- Self Healing
- Route
- Logs
- Service Accounts
- Connecting to a Container terminal
- Labels

Pipelines: Tekton

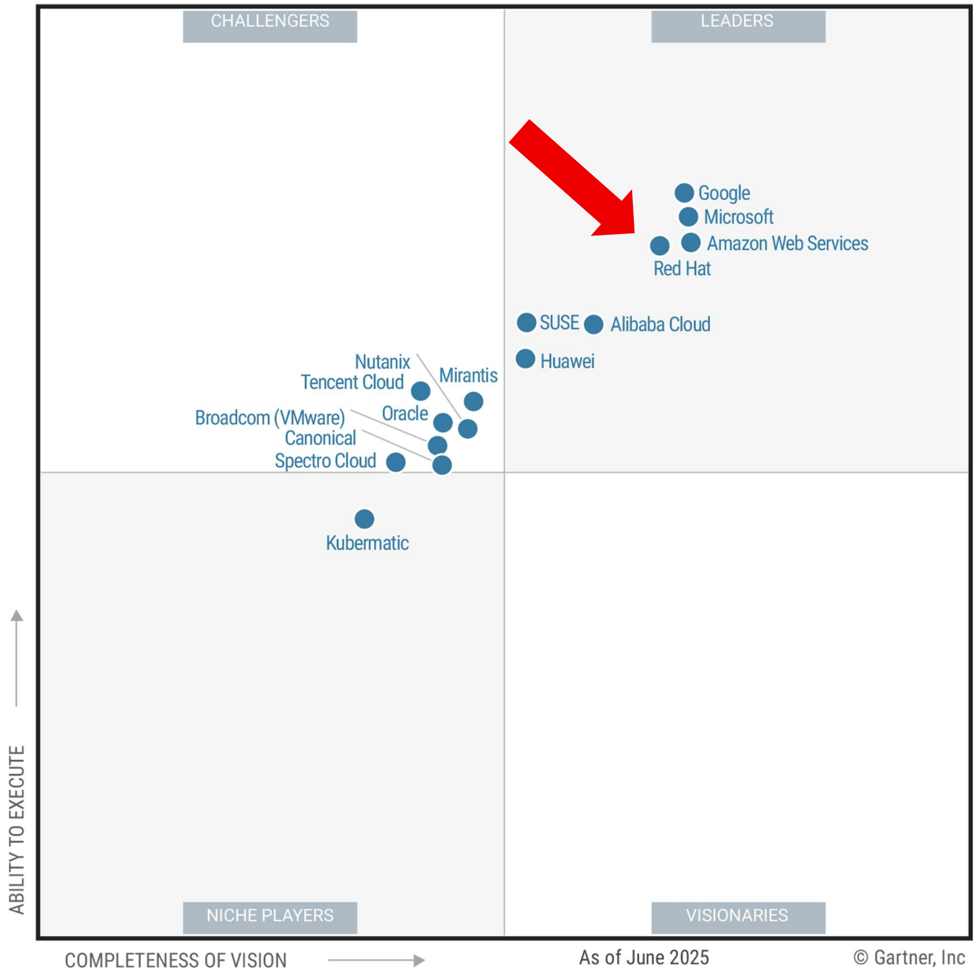
- Continuous Integration

GitOps: ArgoCD

- Continuous Deployment

Red Hat is recognized as a Leader in the 2025 Gartner® Magic Quadrant™ for Container Management for the third year in a row

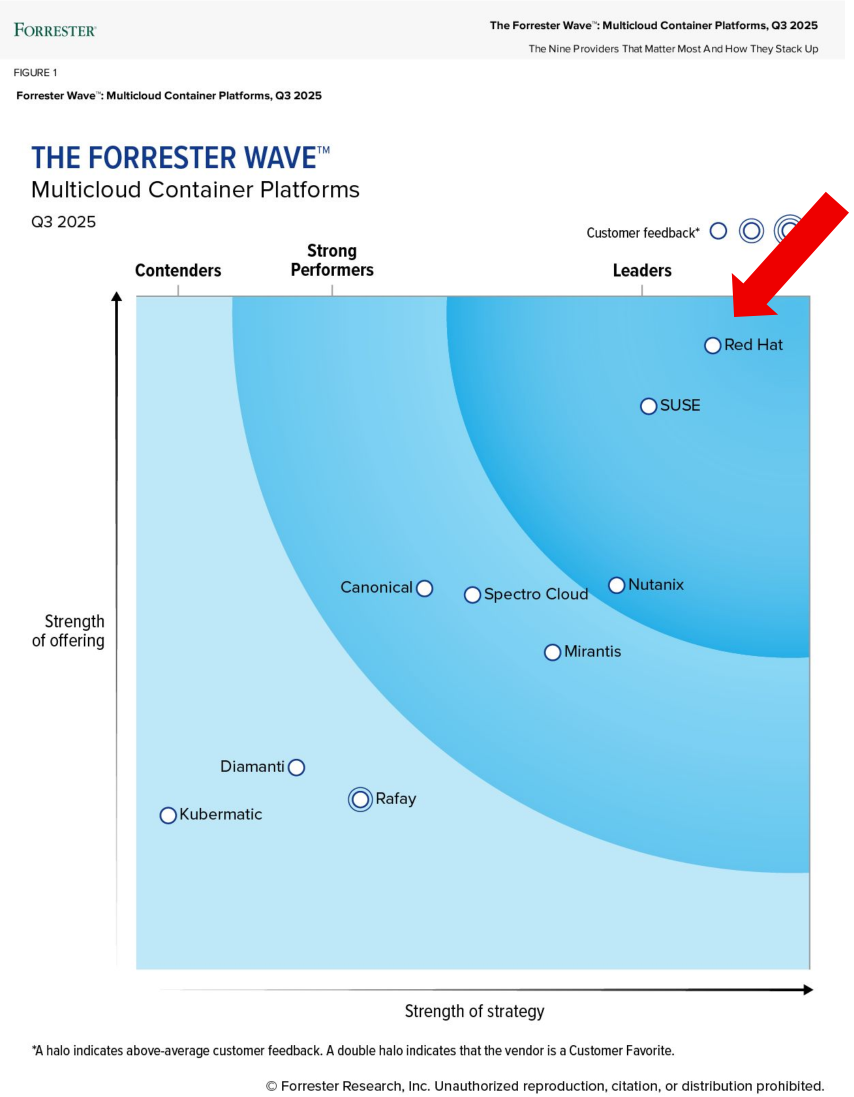
Figure 1: Magic Quadrant for Container Management



- “By 2028, 95% of new **AI** deployments will use Kubernetes, up from less than 30% today.”

Gartner, “Magic Quadrant for Container Management,” Dennis Smith, Tony Iams, Wataru Katsurashima, Michael Warrilow, Richard Watson, 6 August 2025.

The Forrester Wave™: Multicloud Container Platforms, Q3 2025



FORRESTER

The Forrester Wave™: Multicloud Container Platforms, Q3 2025

The Nine Providers That Matter Most And How They Stack Up

FIGURE 1

Forrester Wave™: Multicloud Container Platforms, Q3 2025

*A halo indicates above-average customer feedback. A double halo indicates that the vendor is a Customer Favorite.

- "Red Hat's strong execution has kept it among the **top players in [the MCP] market**."
- The report calls out "…**high-value offerings like OpenShift AI and OpenShift Virtualization as a VMware alternative**."
- **"Red Hat excels in core Kubernetes areas**, offering robust operator options, powerful management, GitOps automation, and flexible interfaces via a GUI or command-line interface (CLI)."
- "OpenShift is a good fit for enterprises that prioritize **support, reliability, and advanced engineering**, particularly in regulated industries such as financial services."

The Forrester Wave™: Multicloud Container Platforms, Q3 2025

Forrester does not endorse any company, product, brand, or service included in its research publications and does not advise any person to select the products or services of any company or brand based on the ratings included in such publications. Information is based on the best available resources. Opinions reflect judgment at the time and are subject to change. For more information, read about Forrester's objectivity here .

Summary

OpenShift

- Dashboard
- Create from
  - **Container image**
  - **S2i**
- Scale the application
- Self Healing
- Route
- Logs
- Service Accounts
- Connecting to a Container terminal
- Labels

Pipelines: Tekton

- Continuous Integration

GitOps: ArgoCD

- Continuous Deployment

Summary

OpenShift

- Dashboard
- Create from
  - **Container image**
  - **S2i**
- Scale the application
- Self Healing
- Route
- Logs
- Service Accounts
- Connecting to a Container terminal
- Labels

Pipelines: Tekton

- Continuous Integration

GitOps: ArgoCD

- Continuous Deployment

Summary

OpenShift

- Dashboard
- Create from
  - **Container image**
  - **S2i**
- Scale the application
- Self Healing
- Route
- Logs
- Service Accounts
- Connecting to a Container terminal
- Labels

Pipelines: Tekton

- Continuous Integration

GitOps: ArgoCD

- Continuous Deployment

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

in linkedin.com/company/red-hat

▶ youtube.com/user/RedHatVideos

f facebook.com/redhatinc

twitter.com/RedHat

Red Hat