Move & Modernize apps - Docker,k8s,OpenShift

Developer Advocate
Teck talk : Ishani Pandey
Lab : Mamatha Busi

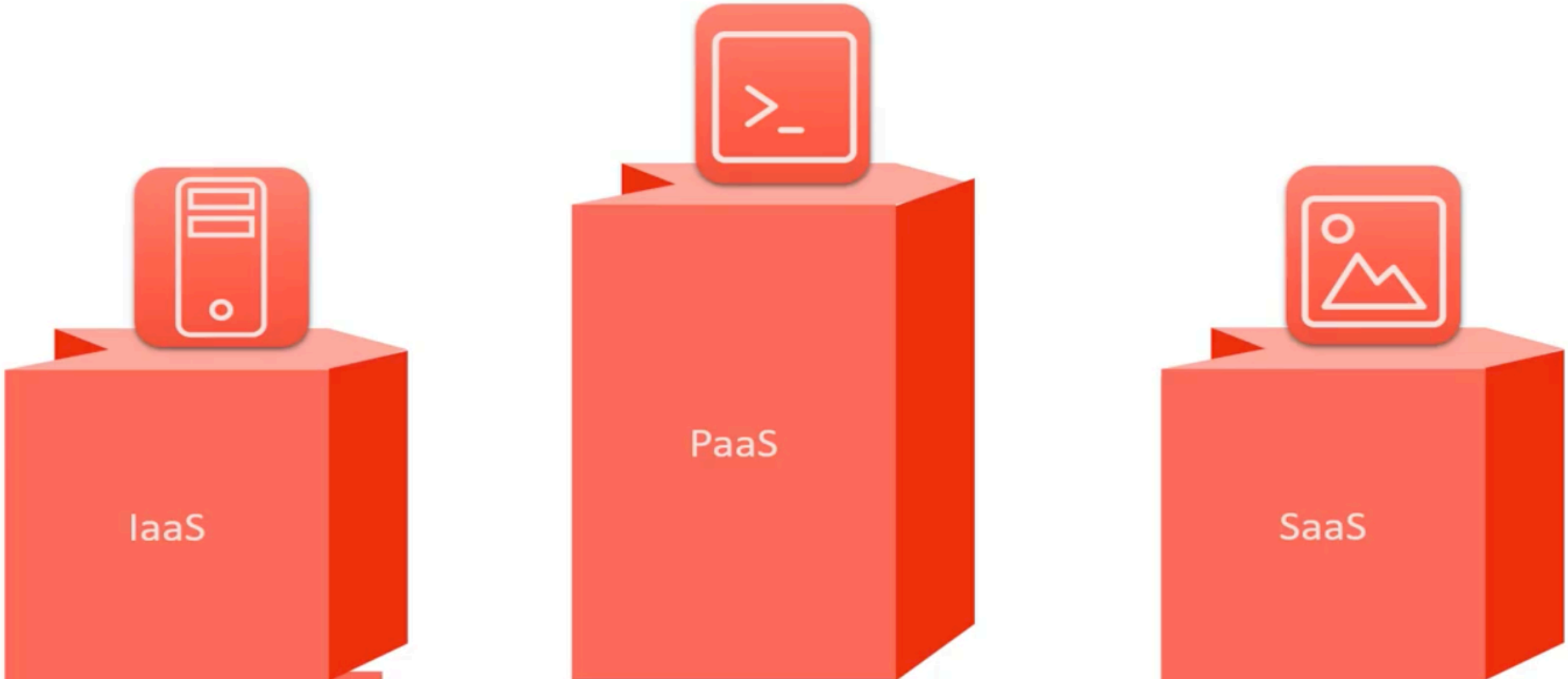
Agenda:

1) OpenShift – What?
2) Flavors of OpenShift
3) Architecture of Installing & working with OpenShift
4) Containers
5) Why do we need Containers
6) What containers can do
7) Difference between Containers & VM
8) Docker
9) Docker Components
10) OpenShift Components
11) Kubernetes & OpenShift : What is the difference
12) OpenShift Web Console walkthrough
13) OpenShift CI walkthrough
14) Projects
15) Routes- Services
16) Scale


OPENSHIFT®
by Red Hat®


IaaS
PaaS
SaaS

OPENSHIFT
by Red Hat



OpenShift

HOSTED SERVICES

Red Hat OpenShift Dedicated

- Private, high-availability Red Hat OpenShift clusters hosted on Amazon Web Services
- Delivered as a hosted service and supported by Red Hat

✔ Supported by Red Hat

Learn more

Red Hat Microsoft Azure

- Highly available Red Hat OpenShift clusters hosted on Microsoft Azure
- Delivered as a hosted service jointly engineered, operated, and supported by Red Hat and Microsoft

✔ Supported by Red Hat and Microsoft

Learn more

Red Hat IBM

- A flexible, fully-managed service of OpenShift on IBM's public cloud
- Delivered as a hosted service and supported by IBM

✔ Supported by Red Hat and IBM

Learn more

SELF-MANAGED

Red Hat OpenShift Container Platform

- A Kubernetes platform on your own infrastructure designed with security in mind
- Build, deploy and manage your container-based applications consistently across cloud and on-premises infrastructure

✔ Supported by Red Hat

Learn more

Red Hat OpenShift Online

Get started developing and deploying in the public cloud

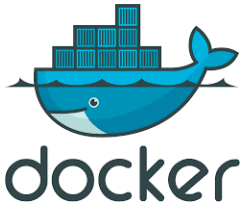
Learn more

What is OpenShift?

- **OpenShift** is based on top of **CRI-O/Docker Containers** and the **Kubernetes** Cluster Manager, with added developer and Operation Centric tools.
- These tools enable rapid application Development , Deployment And Life Cycle Management.

Tools

Kubernetes



Docker

OPENSHIFT

docker

kubernetes

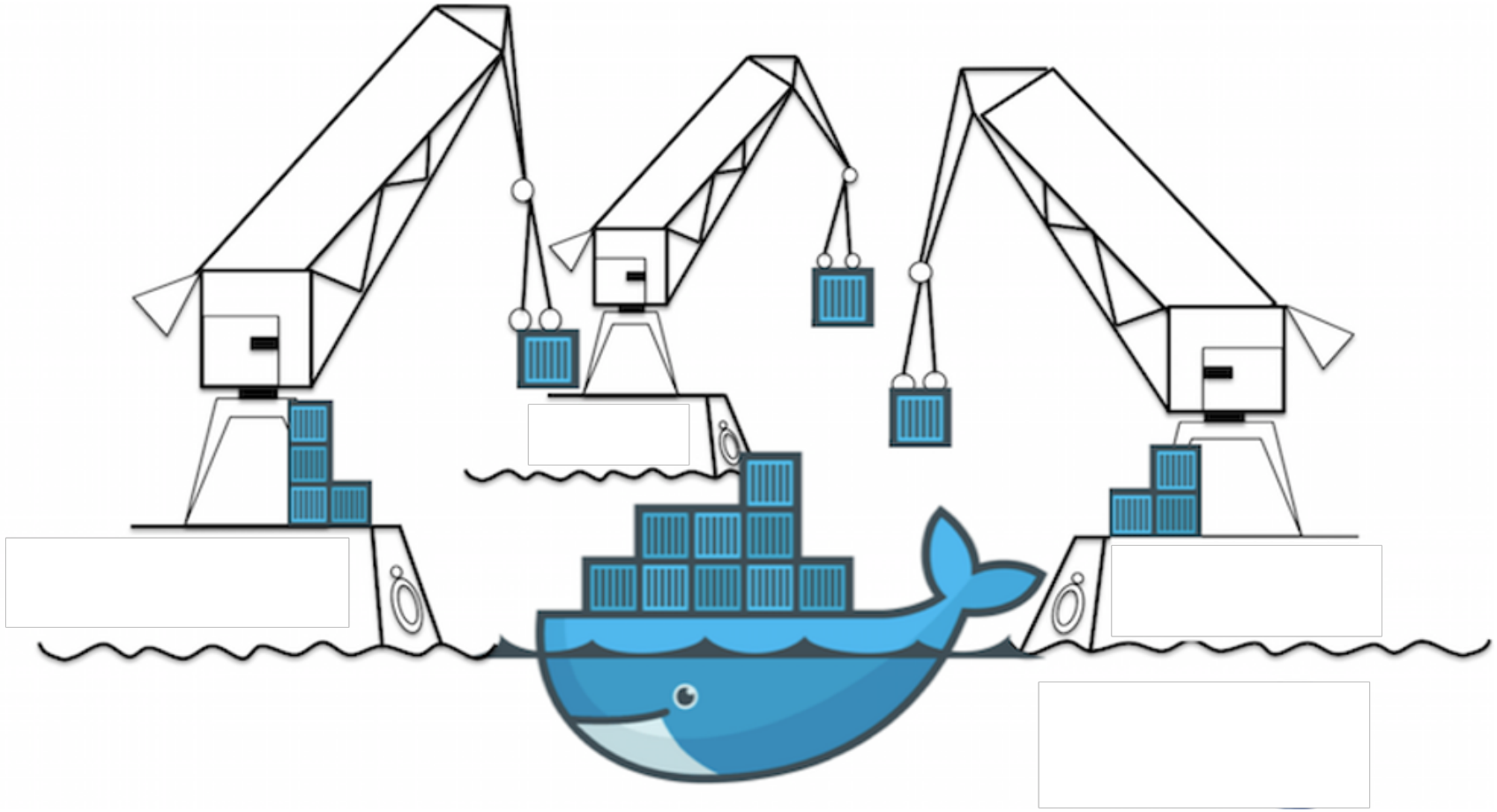
Tools


SCM
Pipeline
Registry
Software
Defined
Networking
API
Governance

Kubernetes / k8s : Pre-requisite

- Container + Orchestration

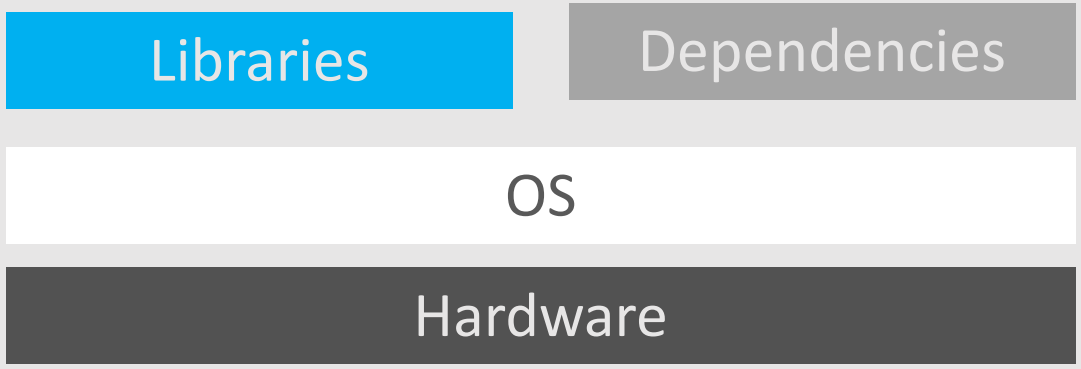
Containers



- Docker
- Containerd
- CRI-O
- Other CRI runtimes: frakt

More than 5 yrs

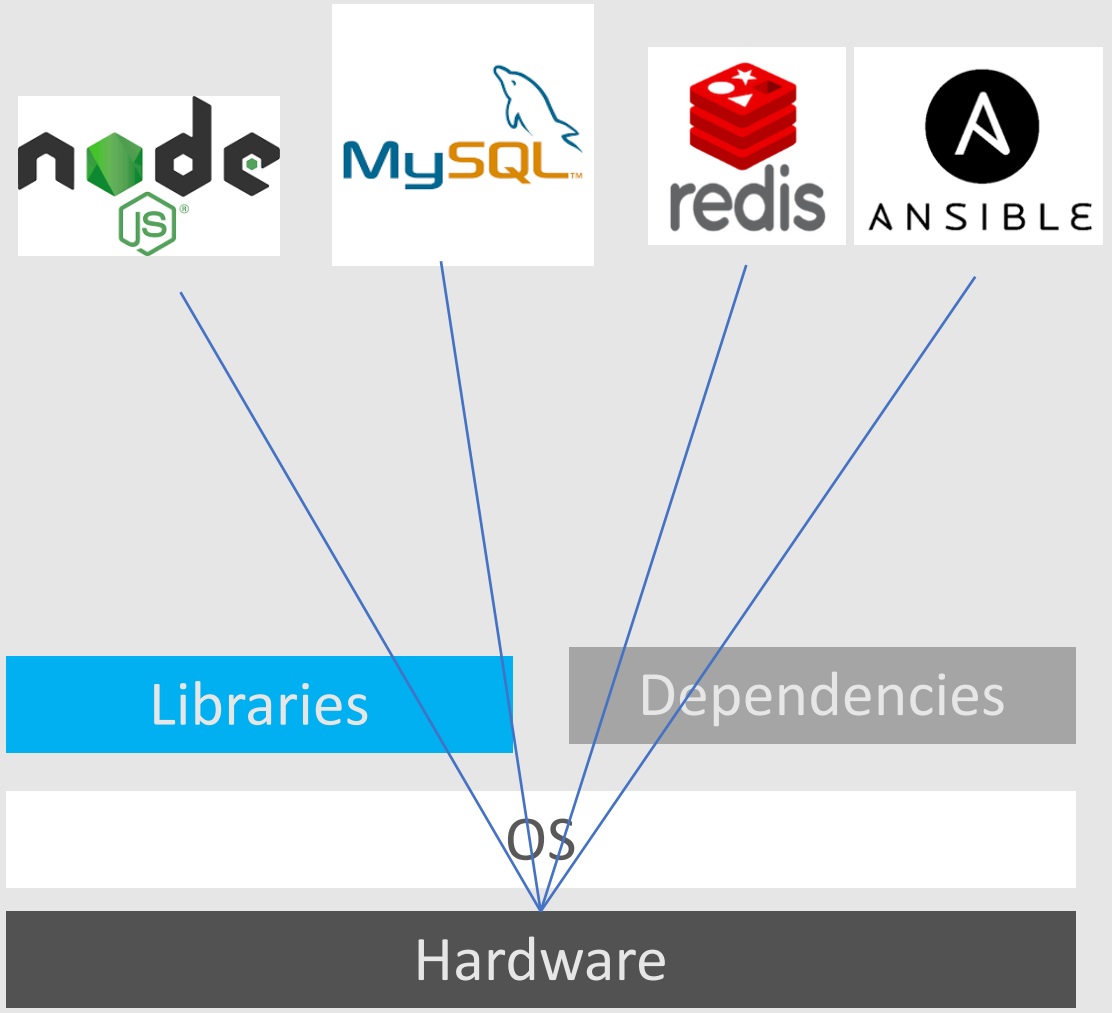
Why do you need containers?



Monolith env.

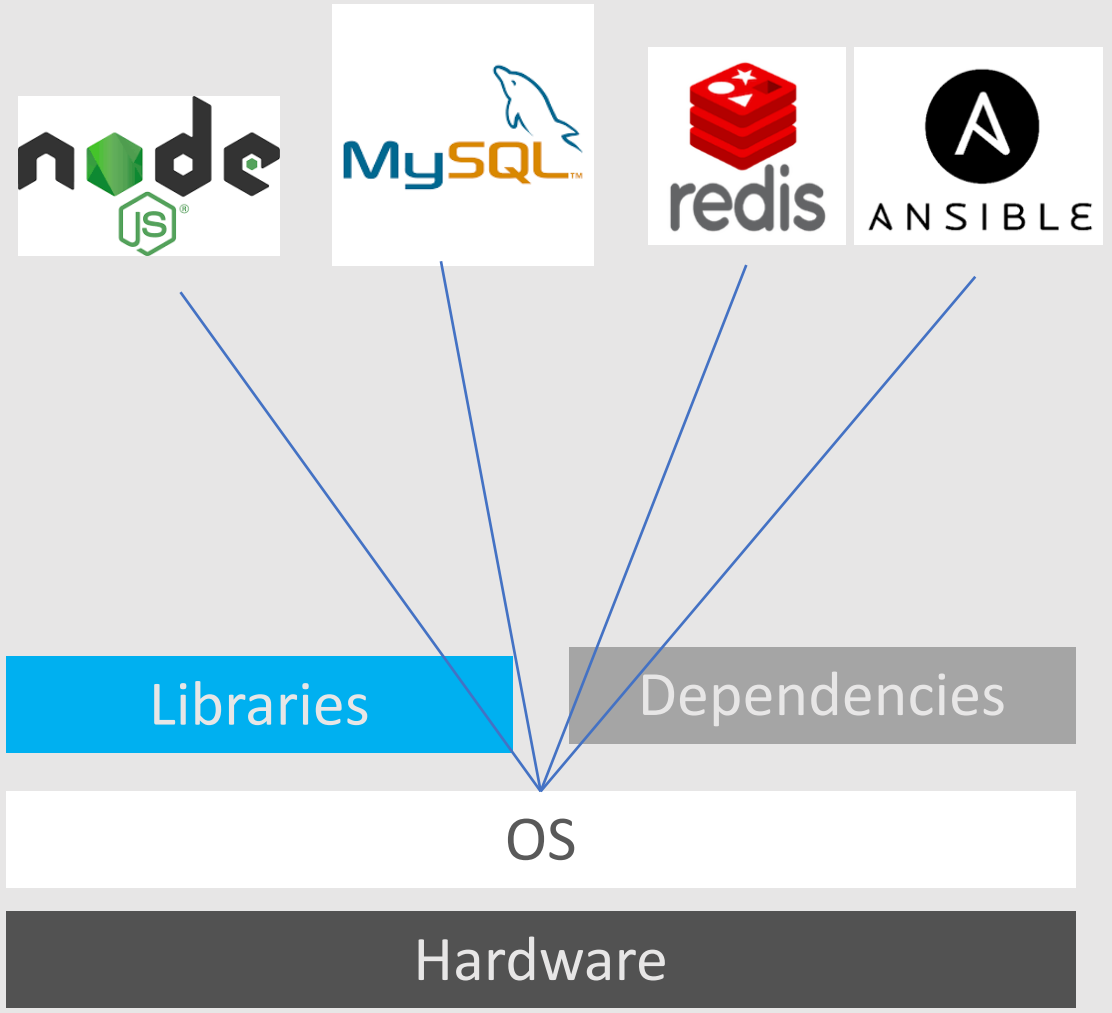
Why do you need containers?

- Right – Required Hardware
- Compatibility/ Dependency
- Long setup time



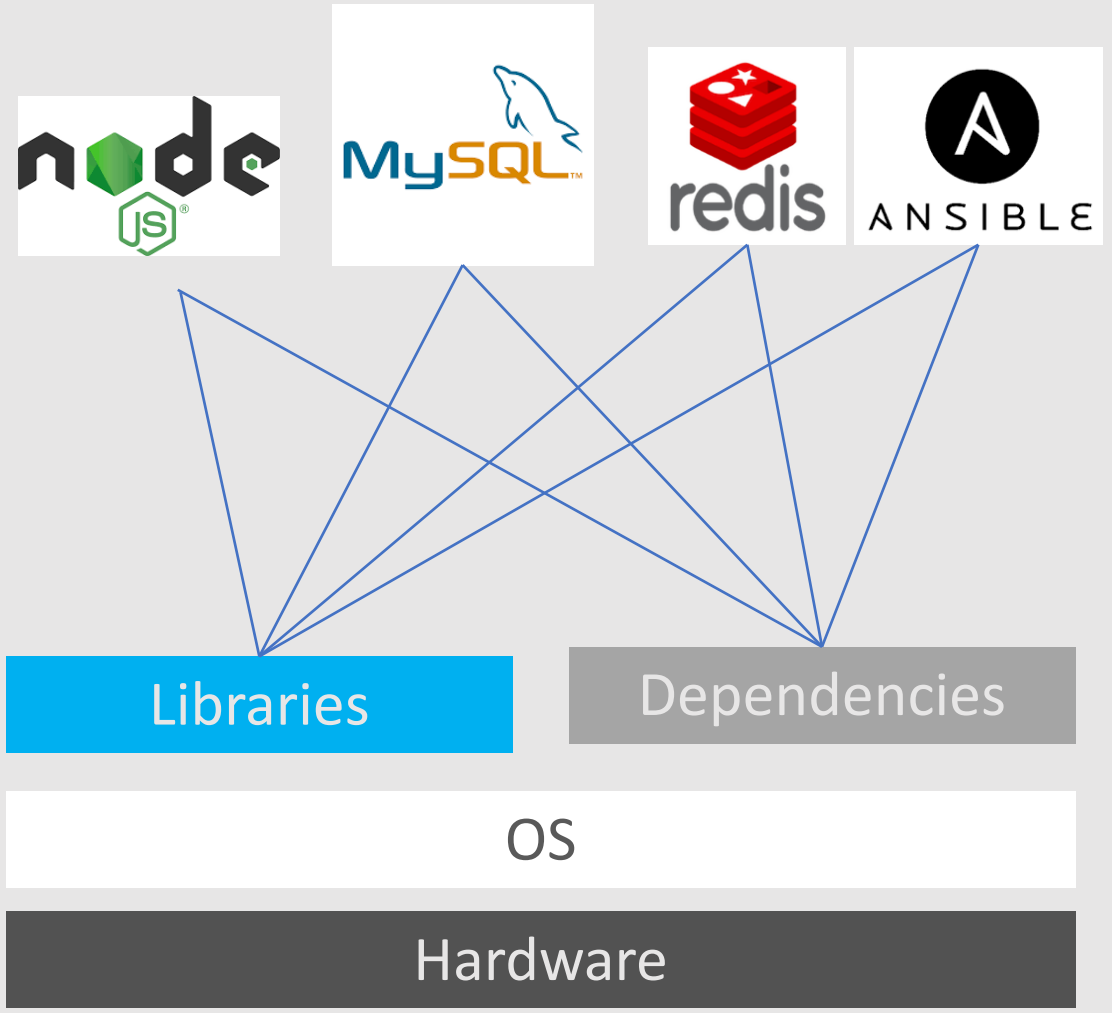
Why do you need containers?

- Compatibility with OS



Why do you need containers?

- Compatibility to work together
- Dependency on particular version supported etc
- Any Upgrade/Change



Why do you need containers?

Disadvantages of the monolith environment:

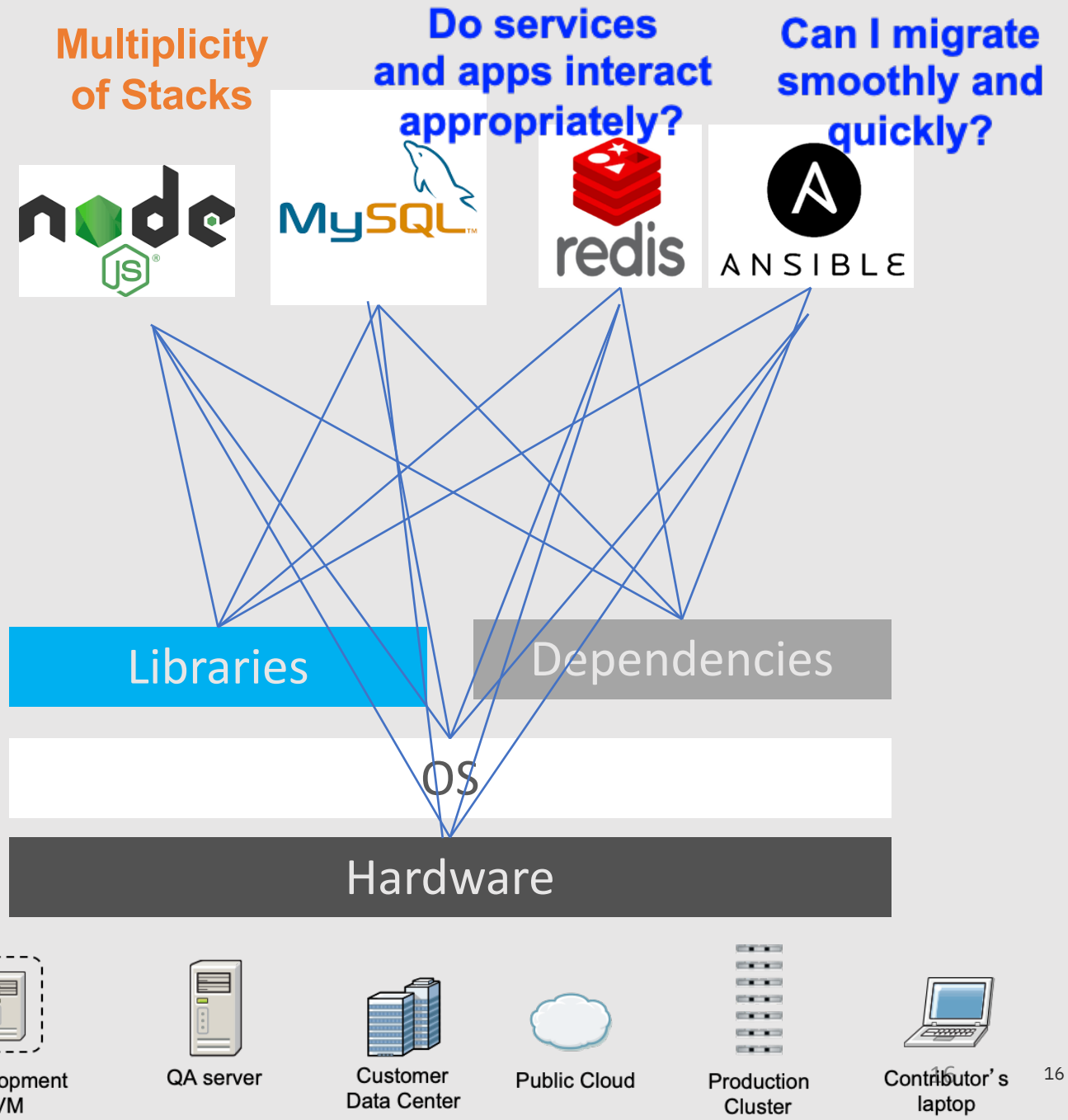
- Compatibility/ Dependency
- Long setup time
- Any Upgrade/Change
- Different Dev/Test/Prod

Dev

Build

Ship

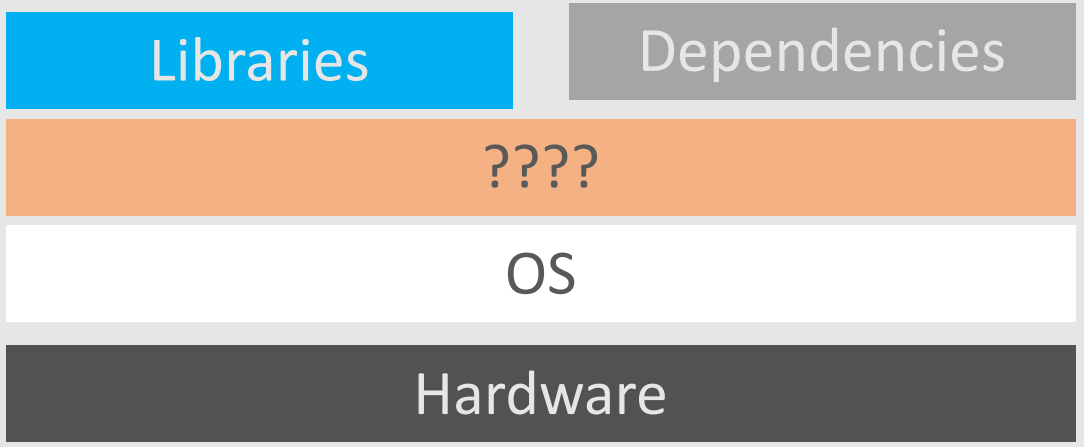
Multiplicity of hardware environments



16

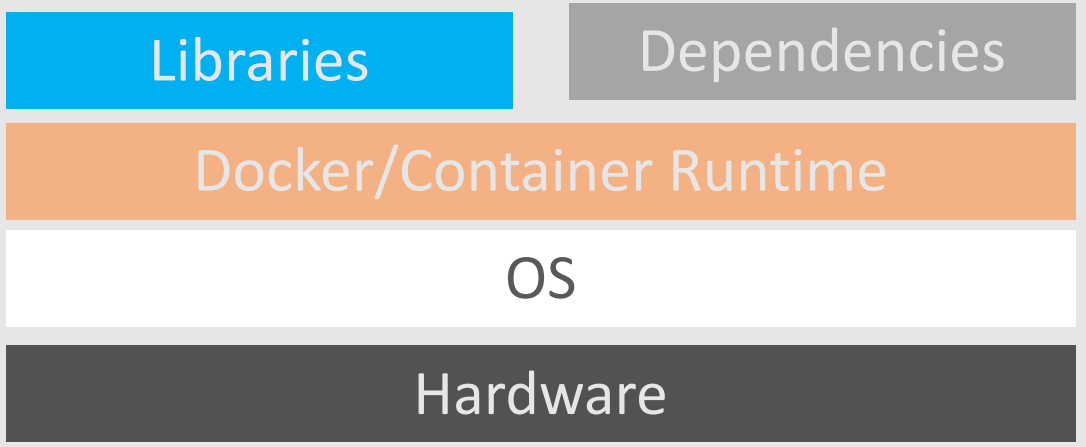
Add layer

- Remove Compatibility/ Dependency
- Without modifying underline OS



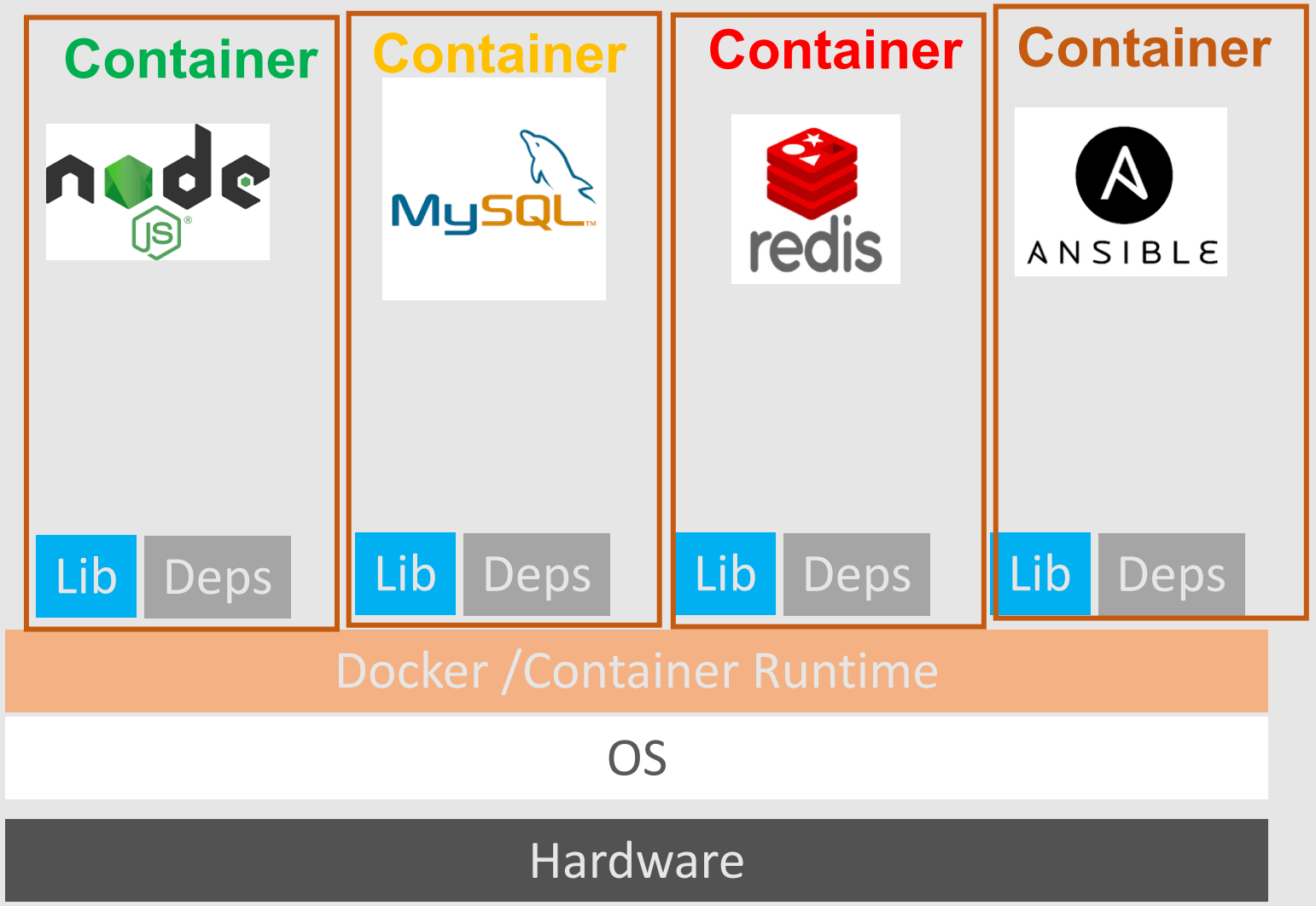
Container

- Remove Compatibility/ Dependency

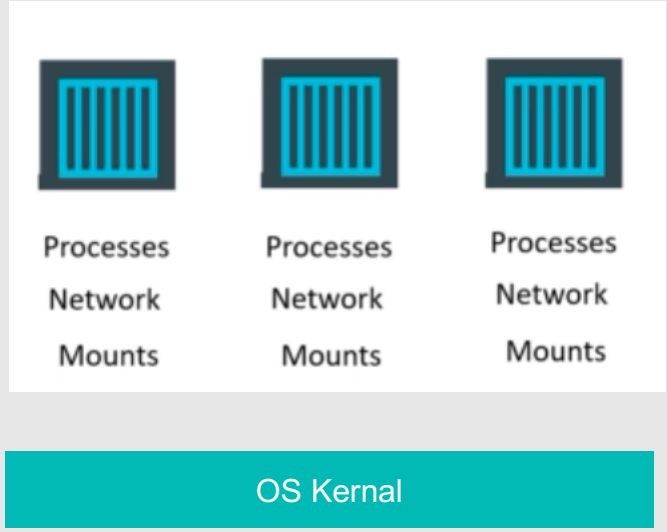


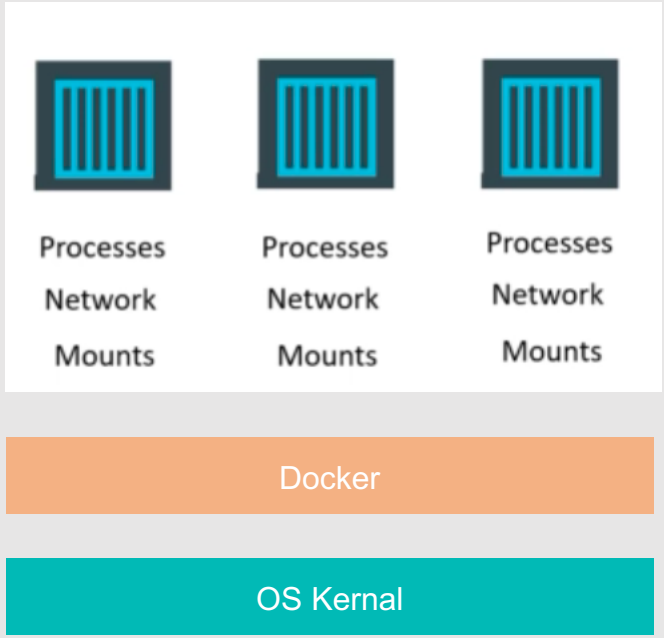
What Container can do?

- Remove Compatibility/ Dependency
- Different Dev/Test/Prod
- Scalability
- Polygot Programming



What are containers?





Lxc, lxd, lxcfs

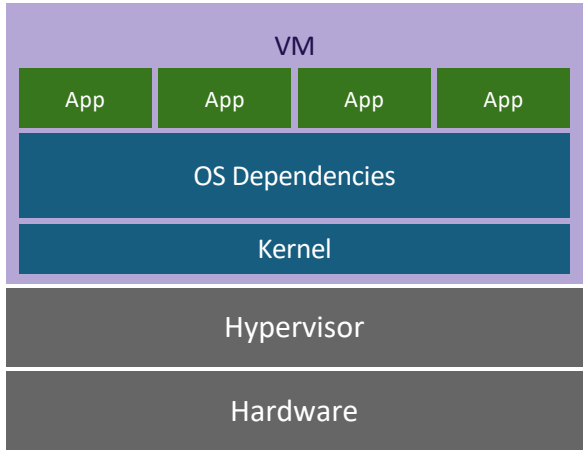
PID - process IDs
USER - user and group IDs
UTS - hostname and domain name
NS - mount points
NET - Network devices, stacks, ports
IPC - inter-process communications, message queues
cgroups - controls limits and monitoring of resources

How Containers differ from VMs ?

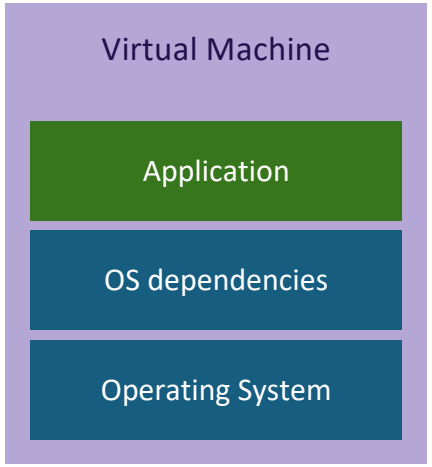
Size Bootup

Utilization

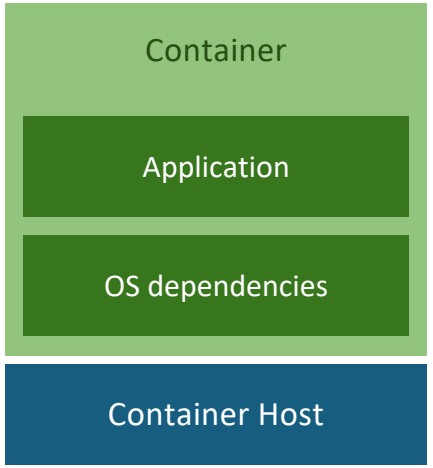
VIRTUAL MACHINES



VM isolates the hardware

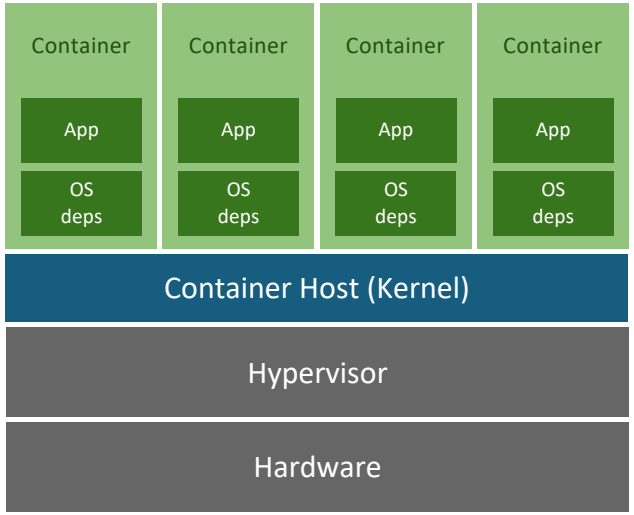


- ✚ VM Isolation
- ➖ Complete OS
- ➖ Static Compute
- ➖ Static Memory
- ➖ High Resource Usage



- ✚ Container Isolation
- ✚ Shared Kernel
- ✚ Burstable Compute
- ✚ Burstable Memory
- ✚ Low Resource Usage

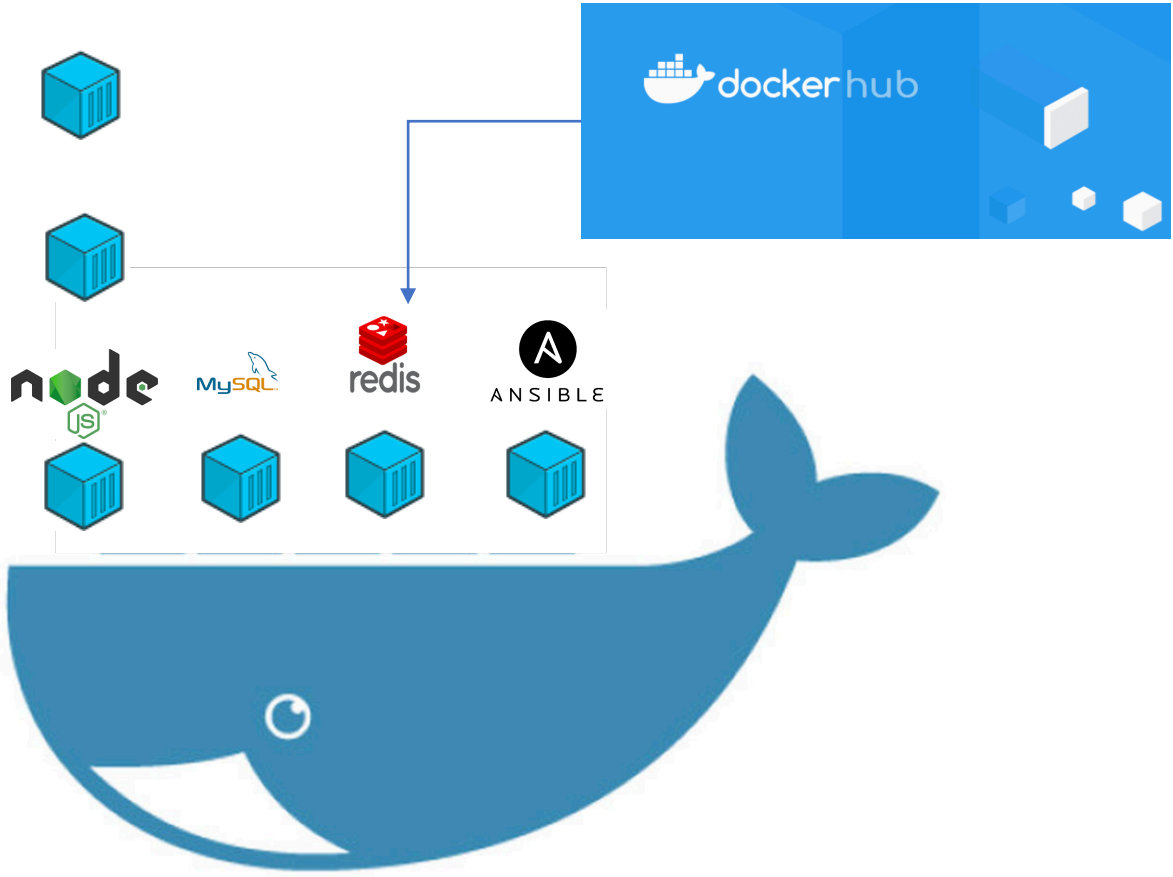
CONTAINERS



Container isolates the process

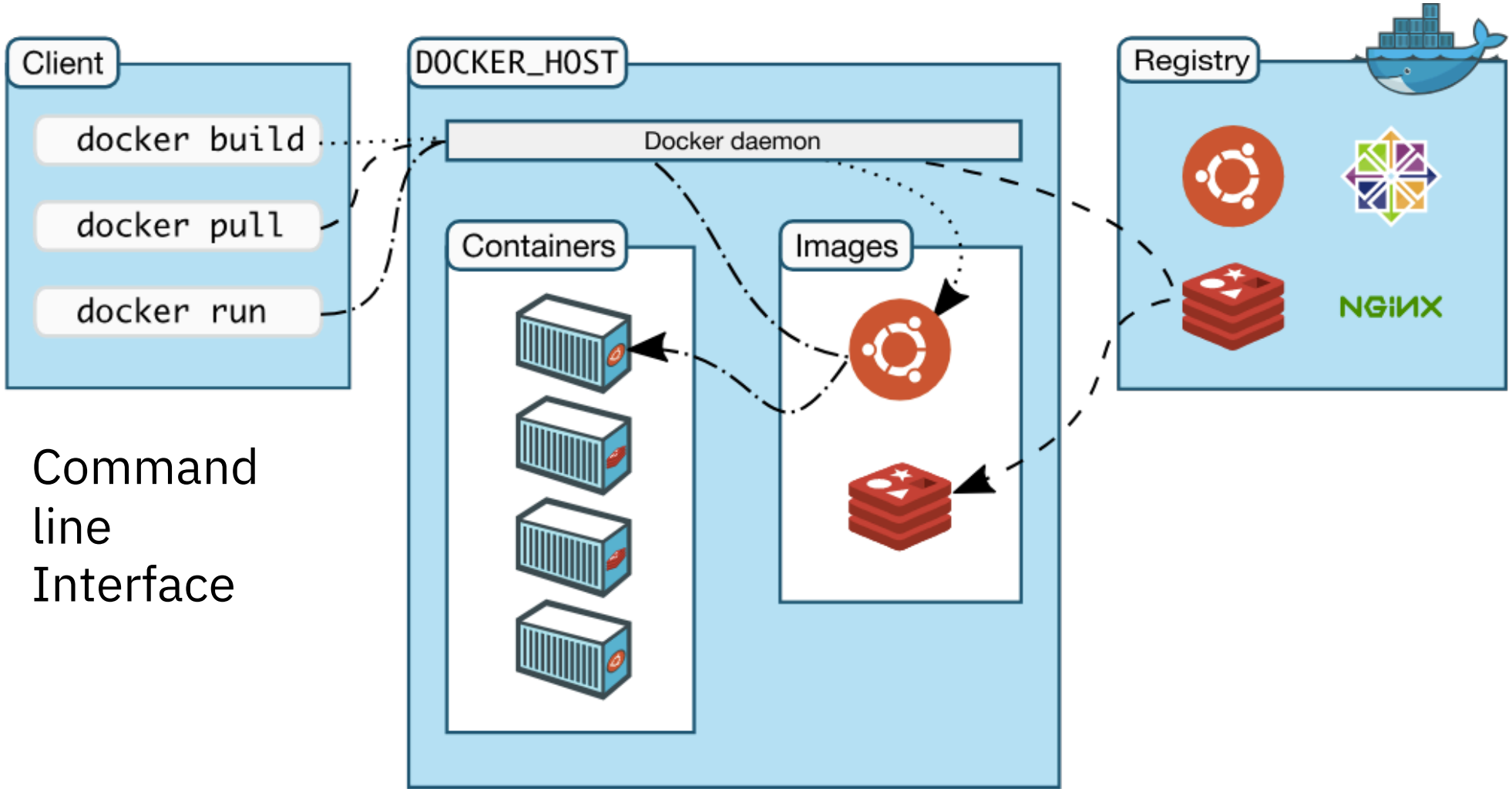
Docker

- docker run ansible
- docker run nodejs
- docker run ansible
- docker run redis



- docker run nodejs
- docker run nodejs

Docker Components



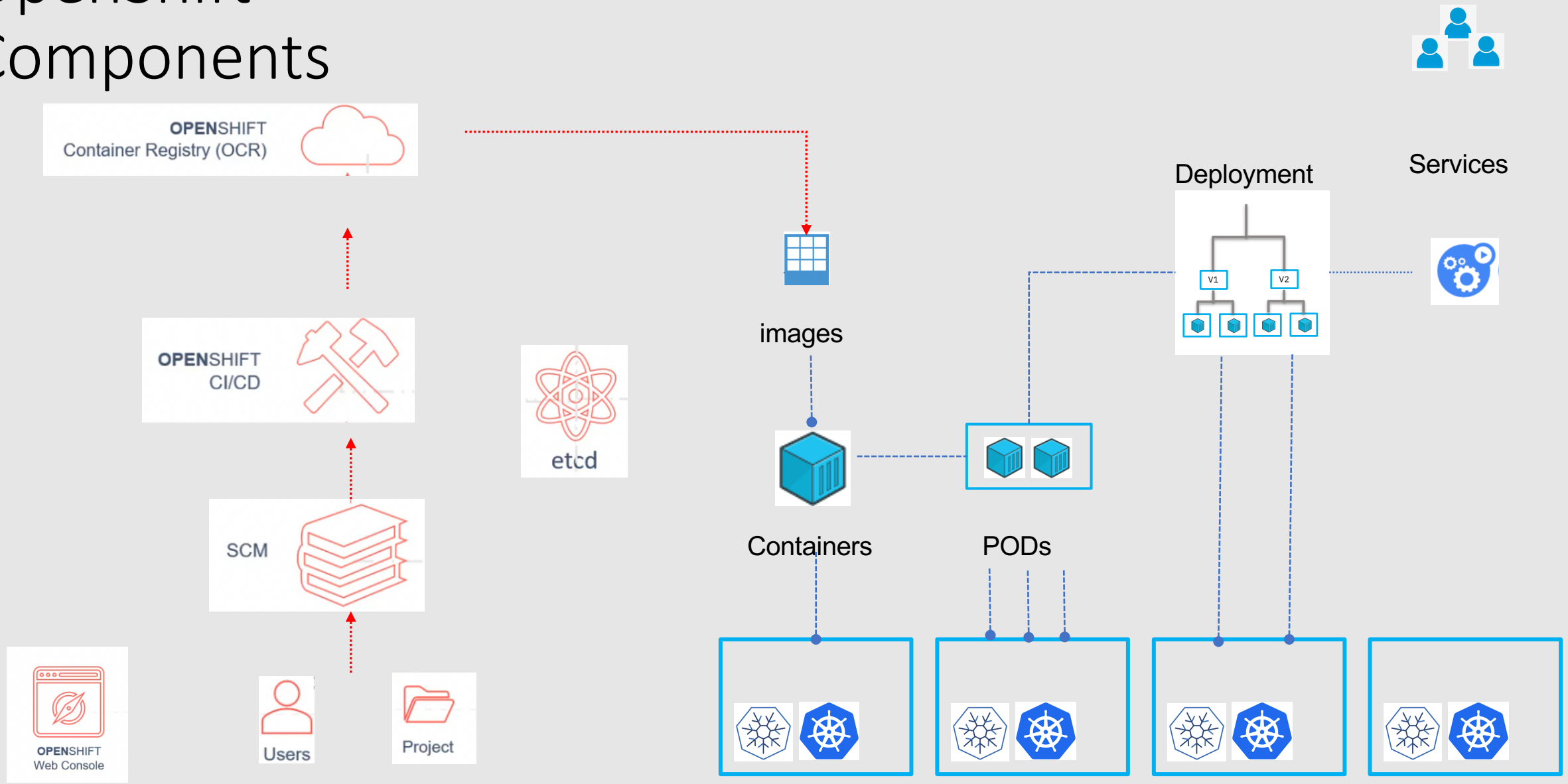
Command line Interface

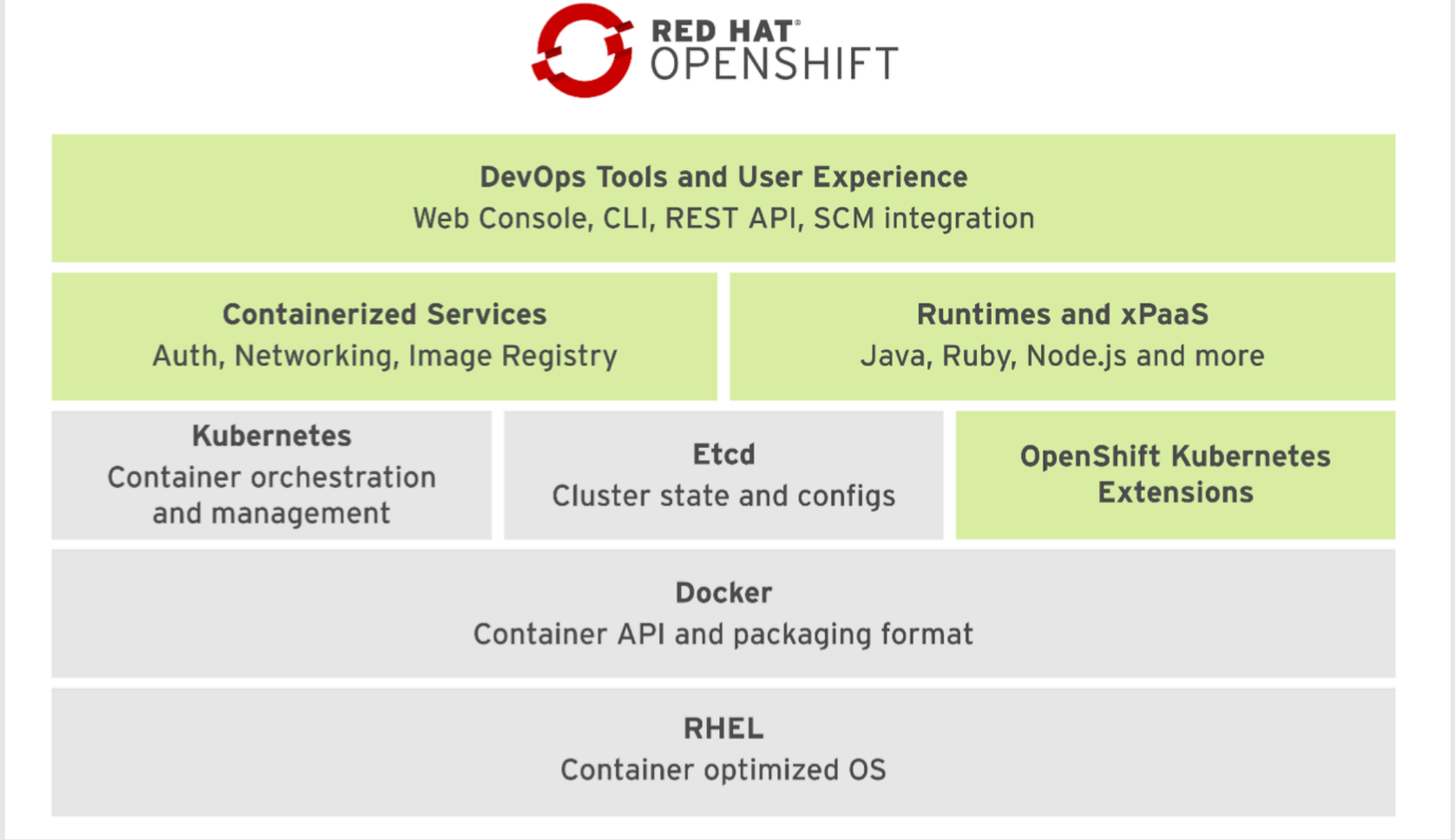
Docker Engine

What is K8s?

- Based on Google's Borg & Omega - Open Source - Container Orchestrator
- Open Governance
  Cloud Native Compute Foundation
- Adoption by Enterprise
  RedHat, Microsoft, IBM and Amazon
- Help to automate DevOps – Deployment, scaling and management of containerized apps
- Helps organize container in logical units(pods, nodes)
- Helps in resource monitoring and logging

OpenShift Components

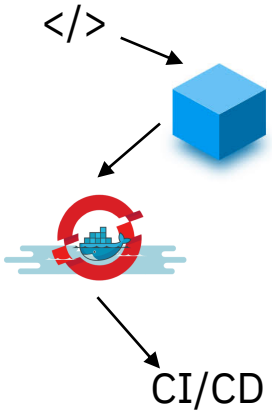



RED HAT
OPENSHIFT
DevOps Tools and User Experience
Web Console, CLI, REST API, SCM integration
Containerized Services
Auth, Networking, Image Registry
Runtimes and xPaaS
Java, Ruby, Node.js and more
Kubernetes
Container orchestration and management
Etcd
Cluster state and configs
OpenShift Kubernetes Extensions
Docker
Container API and packaging format
RHEL
Container optimized OS

Kubernetes & Openshift : What’s the difference

KUBERNETES / OPENSHIFT

DEPLOY

K8S	OPENSHIFT
	PROJECT
CI/CD	✓

MANAGE

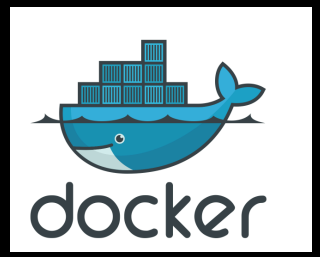
K8S	OPENSHIFT
	
GRAFANA	EFK
ISTIO ...	✓

SECURITY

K8S	OPENSHIFT
RBAC	PROJECT
IAM	✓
AUTH	

OPENSHIFT

OKD K8S



Shared resource types:

- Kubernetes and Openshift share some resource types-
  - Pods
  - Namespaces
  - Deployment config
  - Services
  - Persistent Volumes and Volume claims
  - Secrets

Openshift resource types:

- Some resource types are unique to OpenShift
    - Image
    - Image streams
    - templates

OpenShift Explore

Web Console walkthrough

oc CLI walkthroguh

What is a project?

What is a Route? Creating a Route

Scaling an app, self-healing

Logging applications

Web Console Walkthrough

Search Catalog

Browse Catalog

Deploy Image | Import YAML / JSON | Select from Project

All | Languages | Databases | Middleware | CI/CD | Other

Filter **40 Items**

- .NET Core
- .NET Core + PostgreSQL (Persistent)
- .NET Core Example
- .NET Core Runtime Example
- 3scale-gateway
- amp-apicast-wildcard-router
- amp-pvc
- Apache HTTP Server
- Apache HTTP Server (httpd)
- CakePHP + MySQL
- CakePHP + MySQL (Ephemeral)
- Dancer + MySQL
- Dancer + MySQL (Ephemeral)
- Django + PostgreSQL
- Django + PostgreSQL (Ephemeral)

Display a menu

My Projects

+ Create Project

5 of **28** Projects View All

blog
blog – created by IAM#ispandey@in.ibm.com
2 days ago
testing HCL

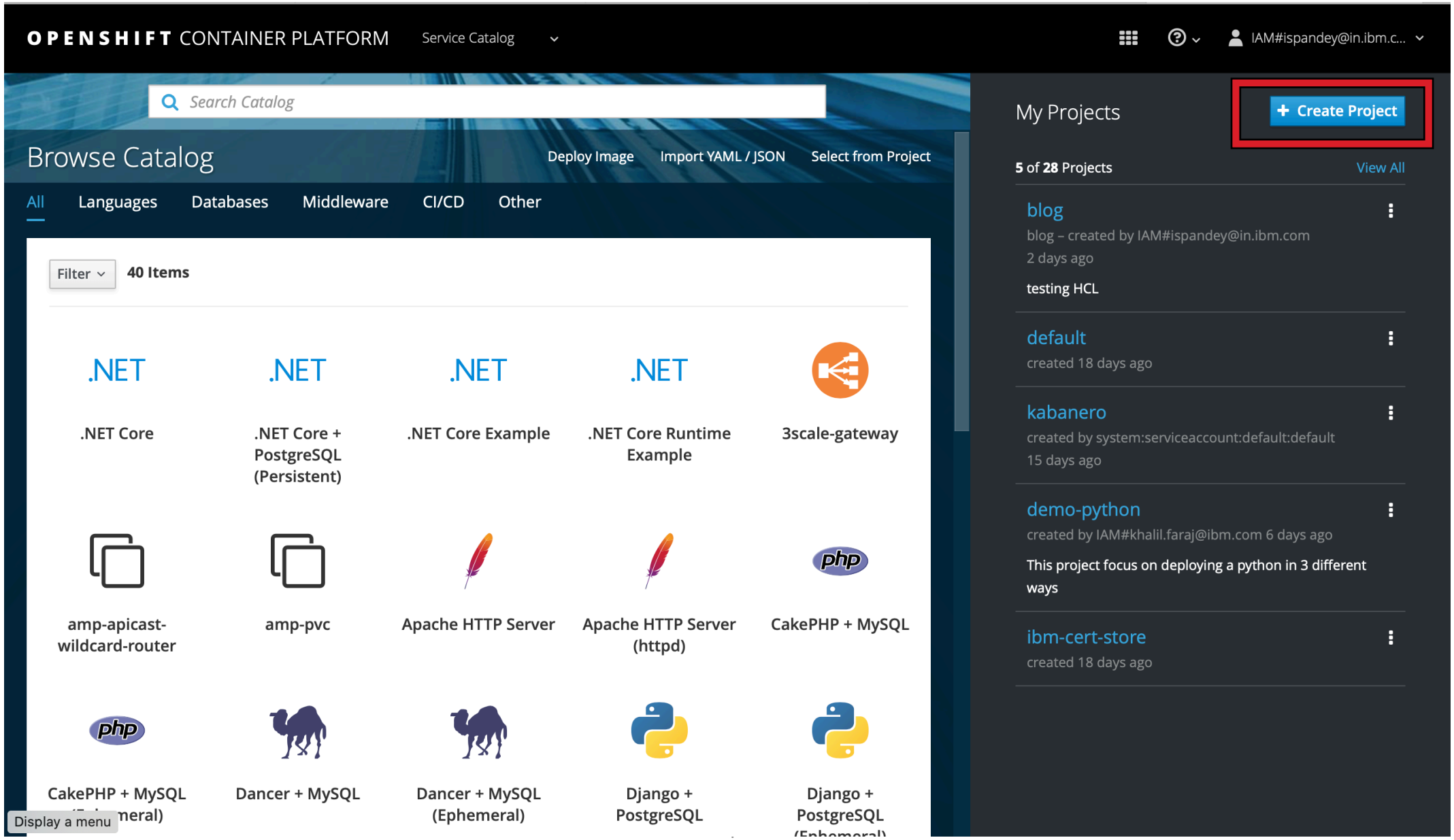
default
created 18 days ago

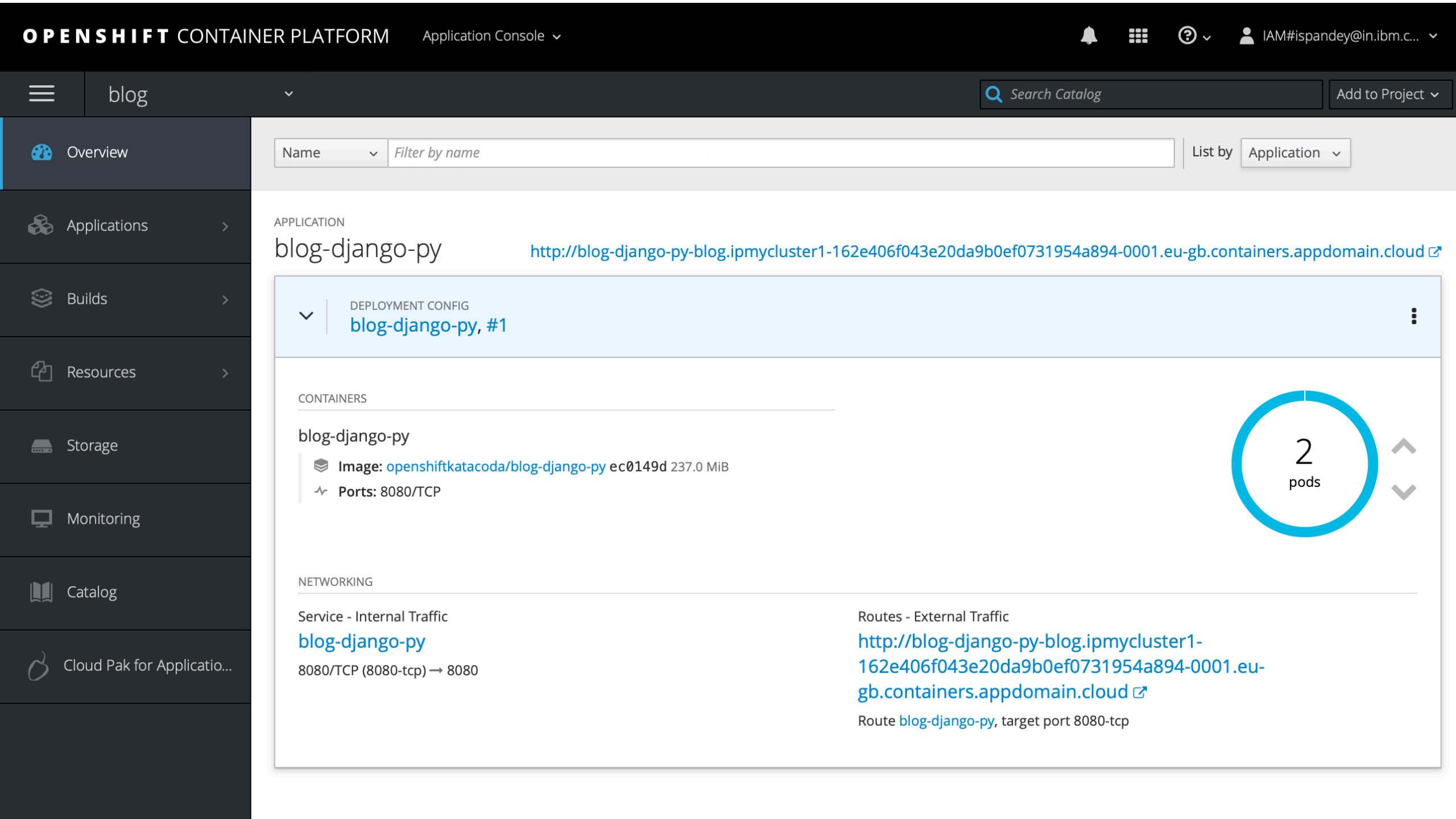
kabanero
created by system:serviceaccount:default:default
15 days ago

demo-python
created by IAM#khalil.faraj@ibm.com 6 days ago
This project focus on deploying a python in 3 different ways

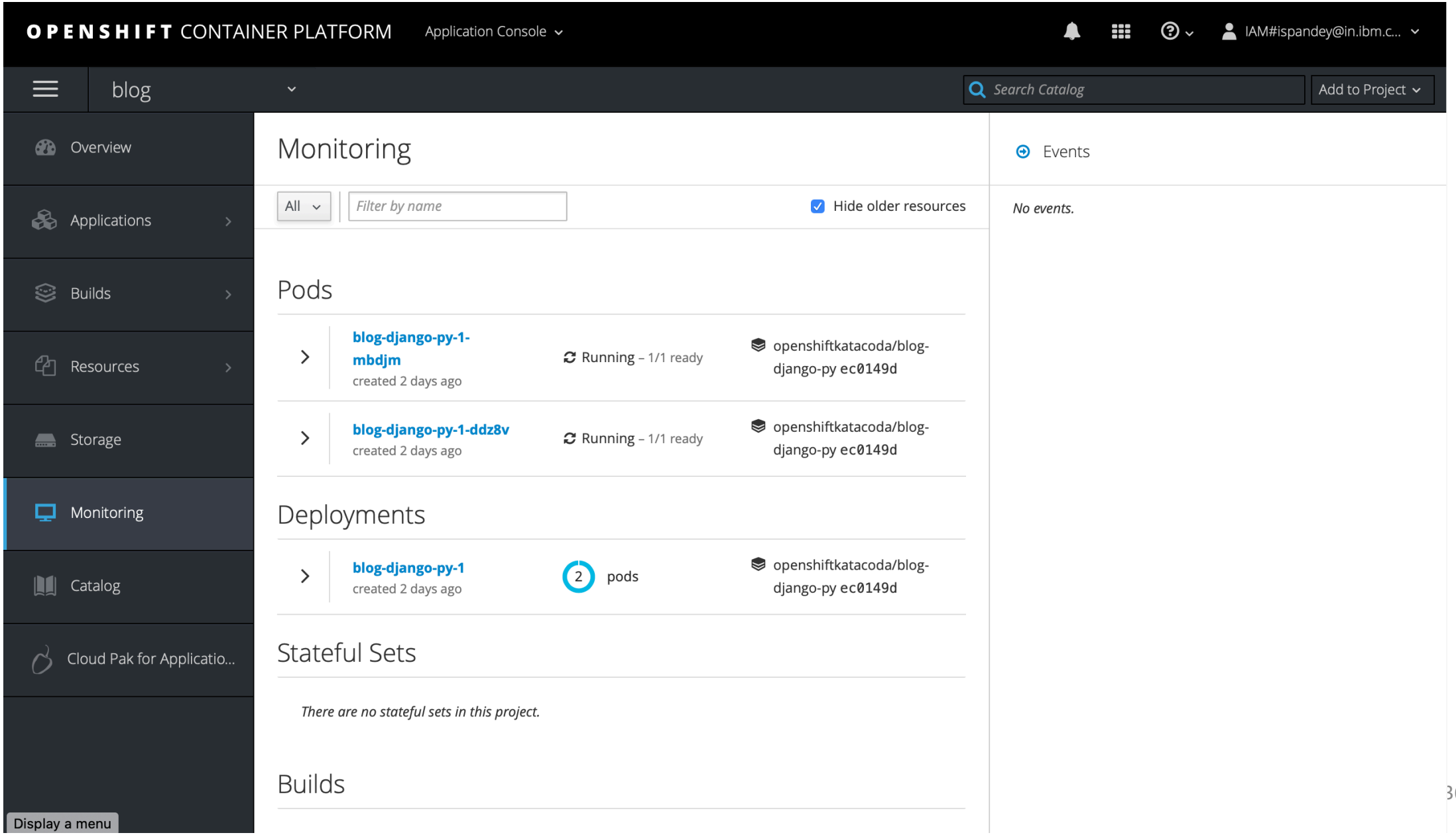
ibm-cert-store
created 18 days ago

Creating a Project

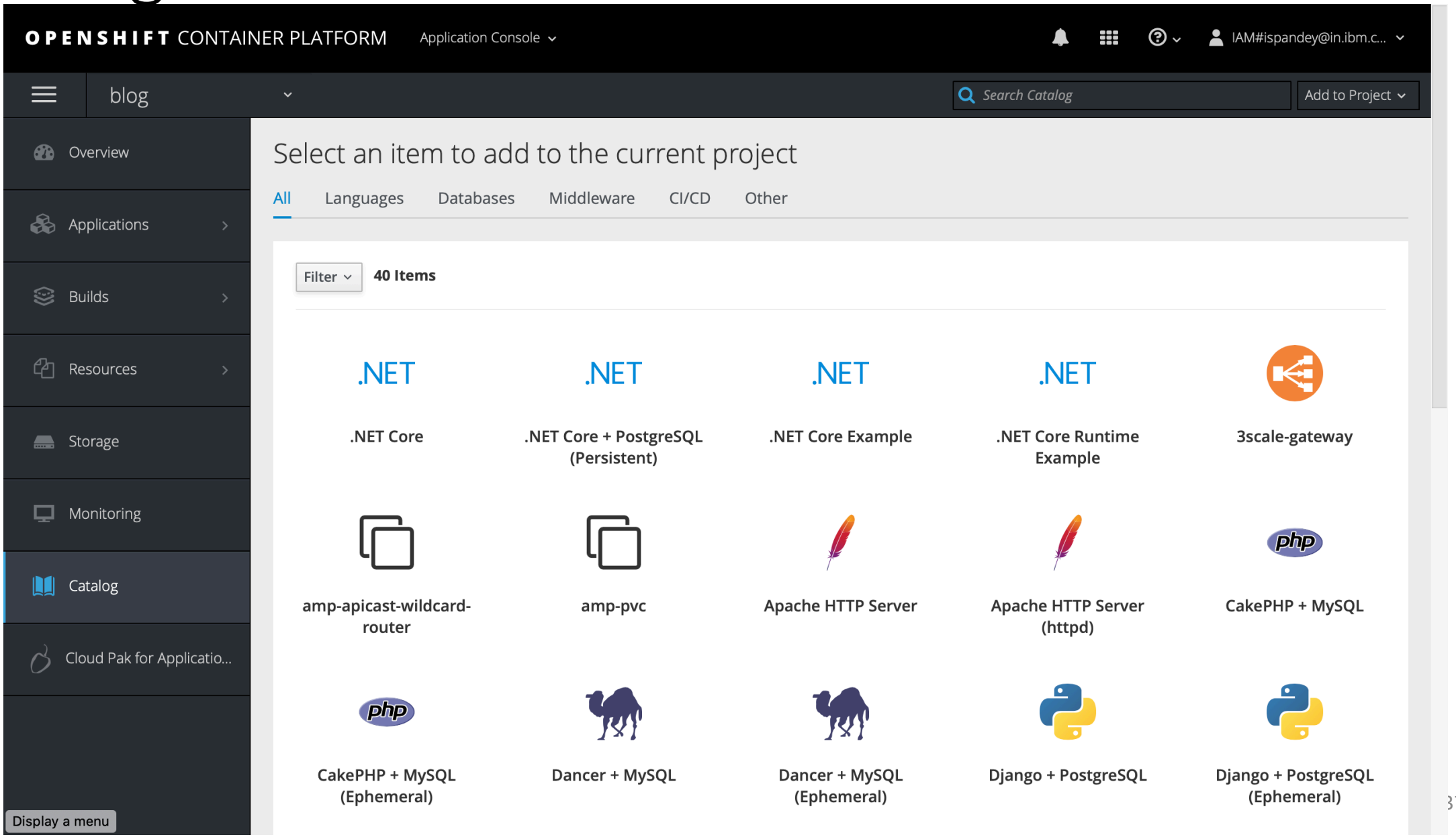



OPENSHIFT CONTAINER PLATFORM
Application Console
IAM#ispandey@in.ibm.c...
blog
Search Catalog
Add to Project
Overview
Applications
Builds
Resources
Storage
Monitoring
Catalog
Cloud Pak for Applicatio...
Name
Filter by name
List by
Application
APPLICATION
blog-django-py
http://blog-django-py-blog.ipmycluster1-162e406f043e20da9b0ef0731954a894-0001.eu-gb.containers.appdomain.cloud
DEPLOYMENT CONFIG
blog-django-py, #1
CONTAINERS
blog-django-py
Image: openshiftkatacoda/blog-django-py ec0149d 237.0 MiB
Ports: 8080/TCP
2
pods
NETWORKING
Service - Internal Traffic
blog-django-py
8080/TCP (8080-tcp) → 8080
Routes - External Traffic
http://blog-django-py-blog.ipmycluster1-162e406f043e20da9b0ef0731954a894-0001.eu-gb.containers.appdomain.cloud
Route blog-django-py, target port 8080-tcp

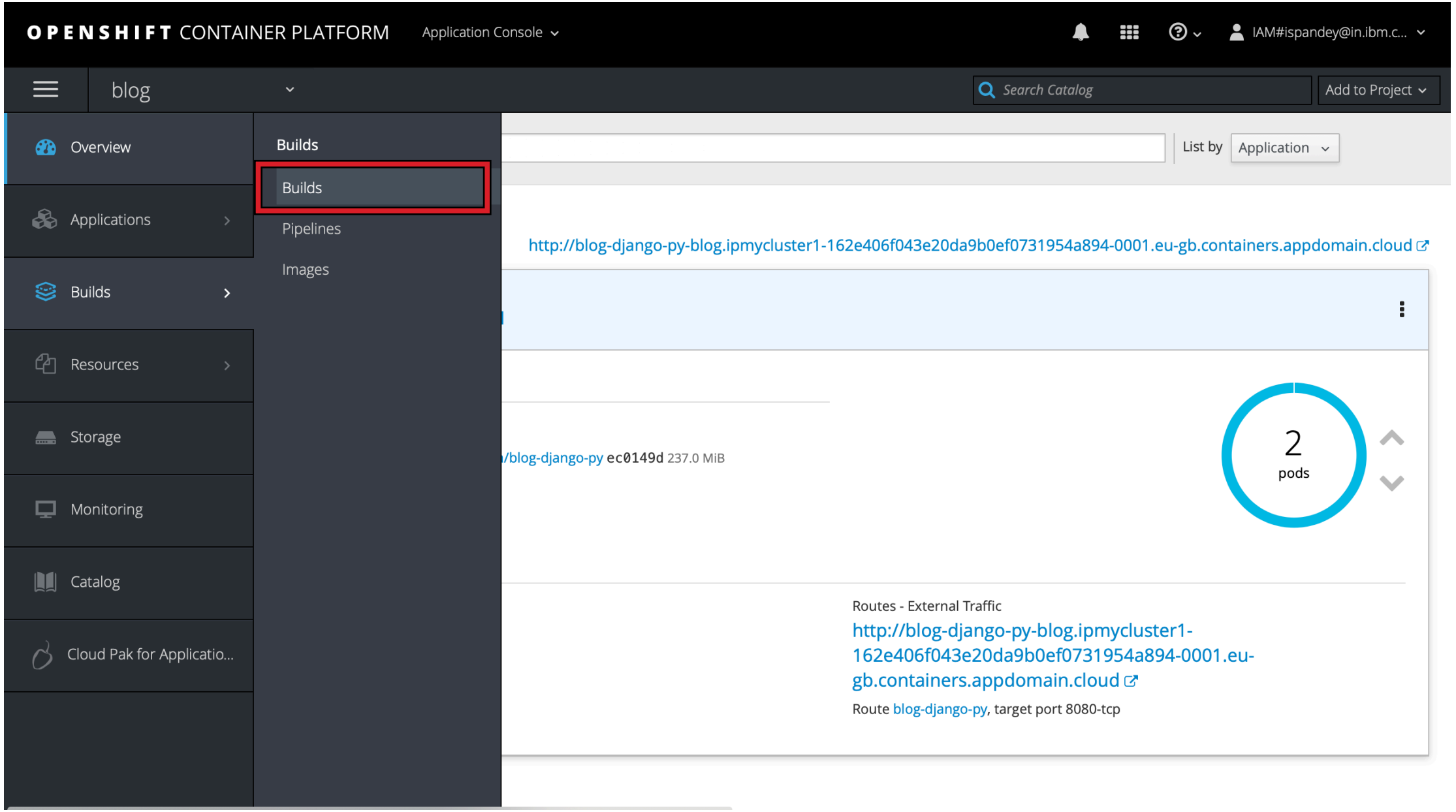
Monitoring


OPENSHIFT CONTAINER PLATFORM
Application Console
IAM#ispandey@in.ibm.c...
blog
Search Catalog
Add to Project
Overview
Applications
Builds
Resources
Storage
Monitoring
Catalog
Cloud Pak for Applicatio...
Display a menu
Monitoring
All
Filter by name
Hide older resources
Pods
blog-django-py-1-mbdjm
created 2 days ago
Running – 1/1 ready
openshiftkatacoda/blog-django-py ec0149d
blog-django-py-1-ddz8v
created 2 days ago
Running – 1/1 ready
openshiftkatacoda/blog-django-py ec0149d
Deployments
blog-django-py-1
created 2 days ago
2 pods
openshiftkatacoda/blog-django-py ec0149d
Stateful Sets
There are no stateful sets in this project.
Builds
Events
No events.

Catalog


OPENSHIFT CONTAINER PLATFORM
Application Console
IAM#ispandey@in.ibm.c...
blog
Search Catalog
Add to Project
Overview
Applications
Builds
Resources
Storage
Monitoring
Catalog
Cloud Pak for Applicatio...
Display a menu
Select an item to add to the current project
All
Languages
Databases
Middleware
CI/CD
Other
Filter
40 Items
.NET Core
.NET Core + PostgreSQL (Persistent)
.NET Core Example
.NET Core Runtime Example
3scale-gateway
amp-apicast-wildcard-router
amp-pvc
Apache HTTP Server
Apache HTTP Server (httpd)
CakePHP + MySQL
CakePHP + MySQL (Ephemeral)
Dancer + MySQL
Dancer + MySQL (Ephemeral)
Django + PostgreSQL
Django + PostgreSQL (Ephemeral)

Manually rebuilding images



CLI Walkthrough

oc CLI can be downloaded from the OpenShift website. However it comes pre-packaged with Minishift utility.

Login using a username and password. For this use oc login command . Or

CLI

```
> oc
OpenShift Client

This client helps you develop, build, deploy, and run your applications on any
OpenShift or Kubernetes compatible
platform. It also includes the administrative commands for managing a cluster
under the 'adm' subcommand.

To create a new application, login to your server and then run new-app:

  oc login https://mycluster.mycompany.com
  oc new-app centos/ruby-22-centos7~https://github.com/openshift/ruby-ex.git
  oc logs -f bc/ruby-ex
```

Login & Logout commands :

CLI - Login

```
> oc login
OpenShift server [https://localhost:8443]: https://openshift.example.com

Username: developer
Authentication required for https://openshift.example.com (openshift)
Password: ******
Login successful.
```

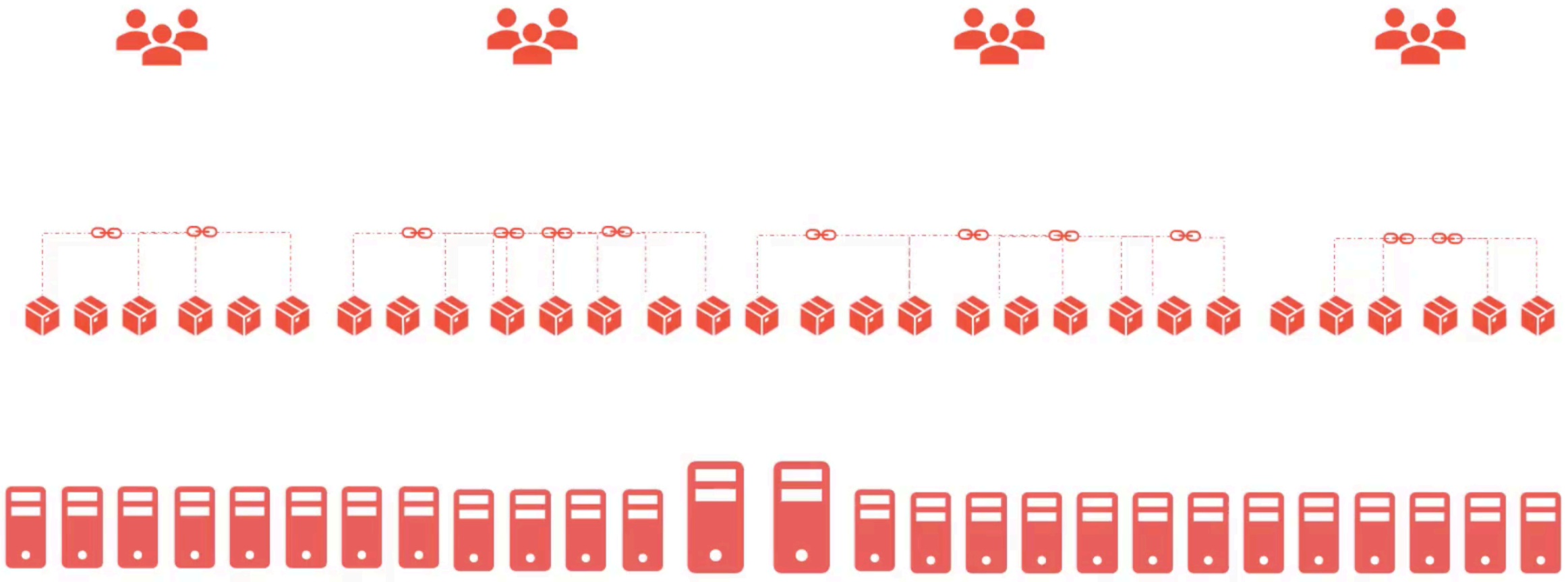
```
> oc login -u developer -p developer
Login successful.
```

```
> oc logout
User, developer, logged out of https://openshift.example.com
```

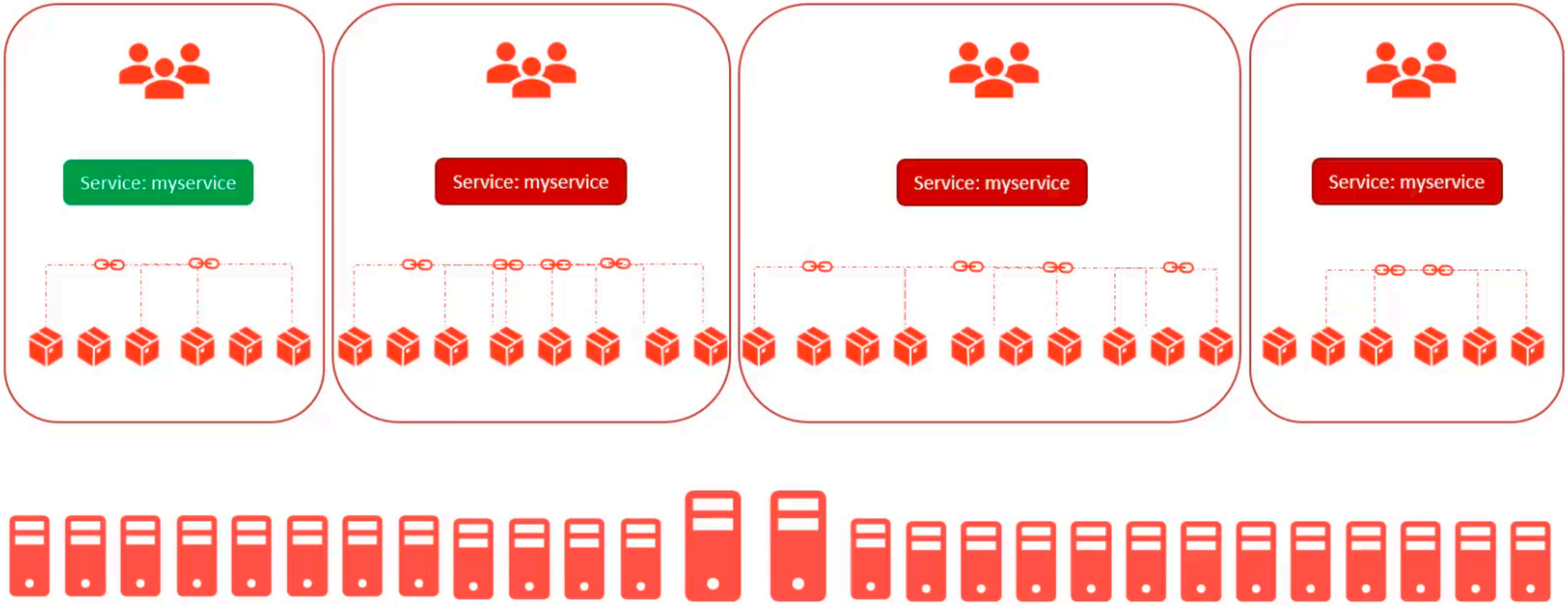
More Details

```
Usage:
  oc [flags]

Basic Commands:
  types           An introduction to concepts and types
  login           Log in to a server
  new-project     Request a new project
  new-app         Create a new application
  status          Show an overview of the current project
  project         Switch to another project
  projects        Display existing projects
  explain         Documentation of resources
  cluster         Start and stop OpenShift cluster

Build and Deploy Commands:
  rollout         Manage a Kubernetes deployment or OpenShift deployment config
  rollback        Revert part of an application back to a previous deployment
  new-build       Create a new build configuration
  start-build     Start a new build
  cancel-build    Cancel running, pending, or new builds
  import-image    Imports images from a Docker registry
  tag             Tag existing images into image streams

Application Management Commands:
  get             Display one or many resources
  describe        Show details of a specific resource or group of resources
  edit            Edit a resource on the server
  set             Commands that help set specific features on objects
  label           Update the labels on a resource
  annotate        Update the annotations on a resource
  expose          Expose a replicated application as a service or route
  delete          Delete one or more resources
  scale           Change the number of pods in a deployment
  autoscale       Autoscale a deployment config, deployment, replication controller, or replica set
  secrets         Manage secrets
  serviceaccounts Manage service accounts in your project

Troubleshooting and Debugging Commands:
  logs            Print the logs for a resource
  rsh             Start a shell session in a pod
  rsync           Copy files between local filesystem and a pod
  port-forward    Forward one or more local ports to a pod
  debug           Launch a new instance of a pod for debugging
  exec            Execute a command in a container
  proxy           Run a proxy to the Kubernetes API server
  attach          Attach to a running container
  run             Run a particular image on the cluster
```

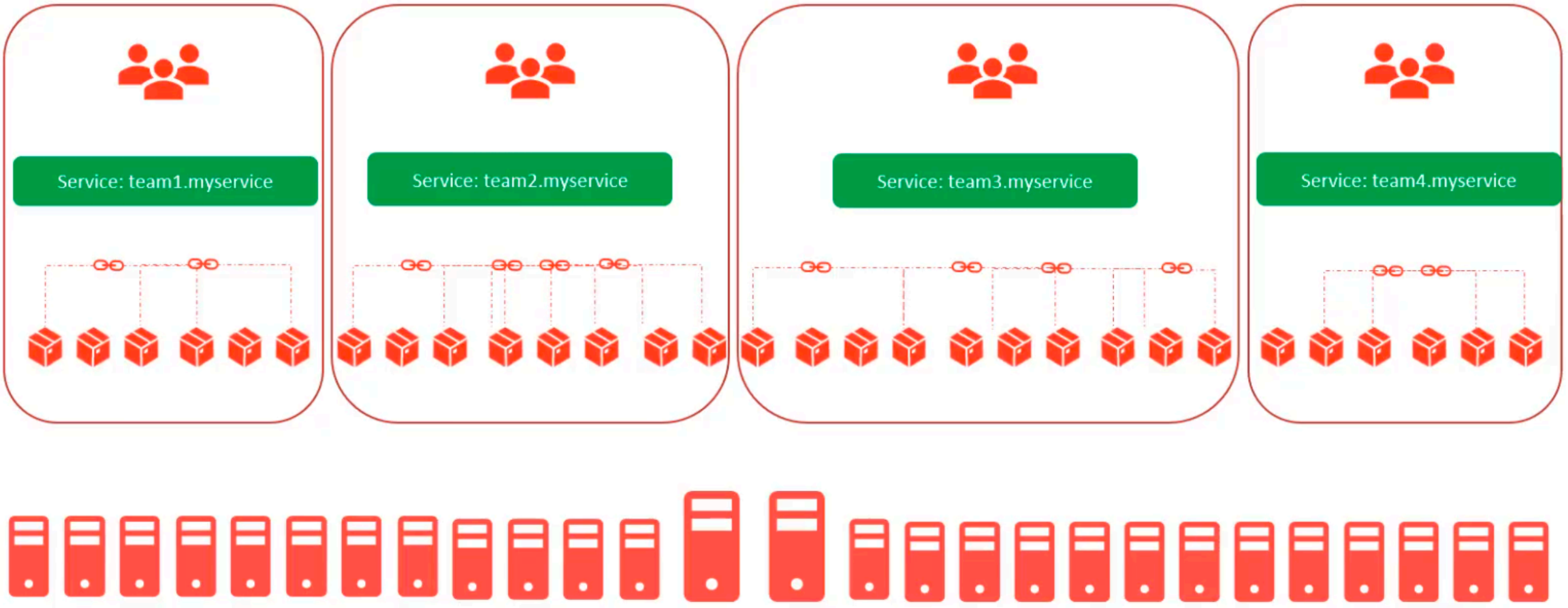
Projects



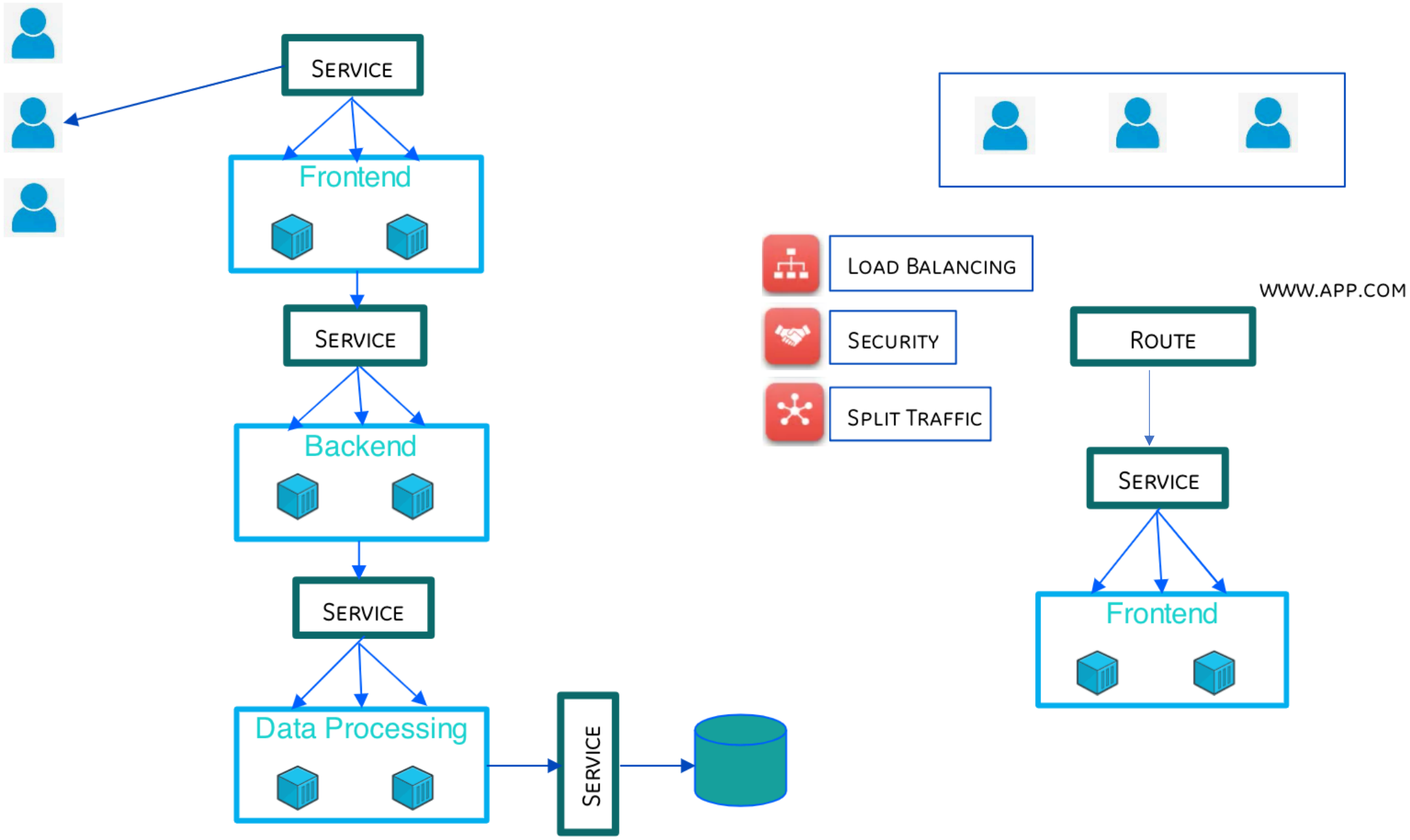
Projects


Service: myservice
Service: myservice
Service: myservice
Service: myservice

Projects


Service: team1.myservice
Service: team2.myservice
Service: team3.myservice
Service: team4.myservice

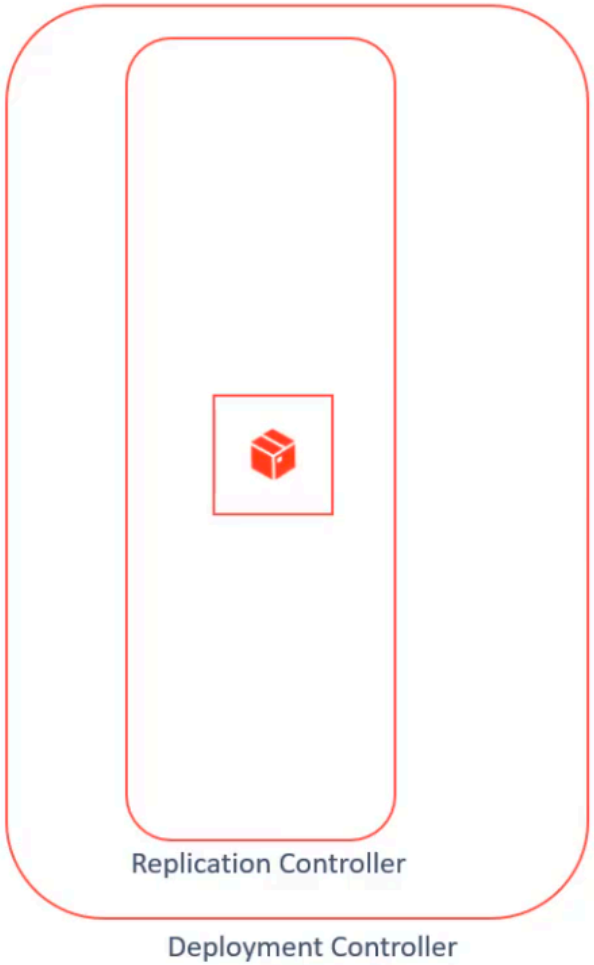
Routes - Services



Understanding Routes

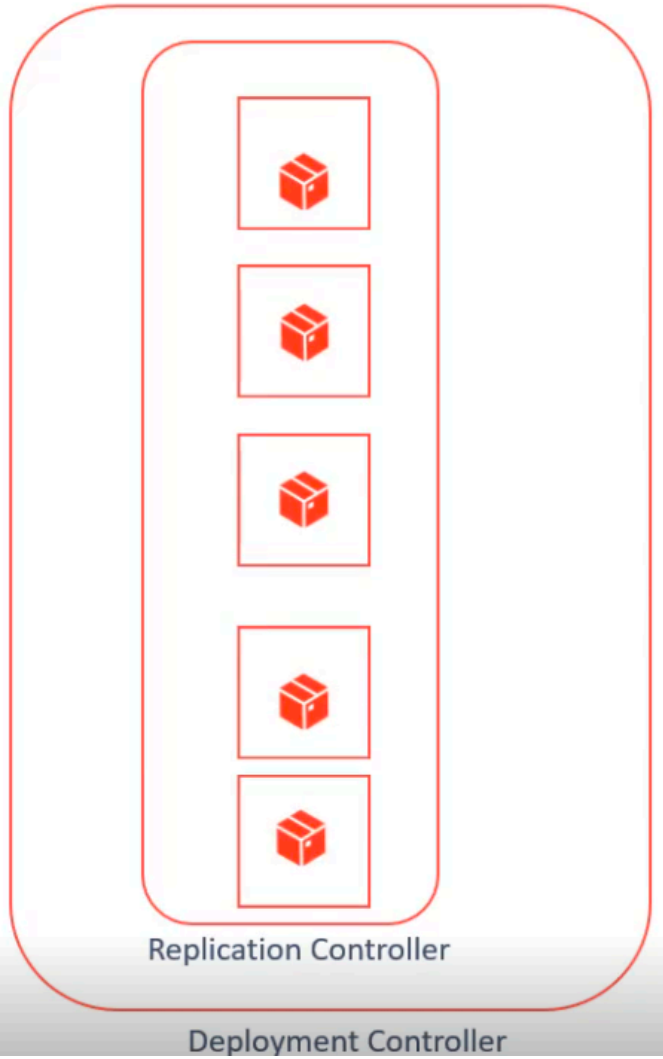
- OpenShift routes allow network access to pods from outside the Openshift environment.
- A dedicated router pod is used to load-balance traffic between the target pods.
- The router pods uses HAProxy and can be scaled itself.
- The router exposes a public facing IP address and DNS hostname to the internal pod networking.
- Routers connect directly to the pods, the service is used for pod lookup only, but not involved in the actual traffic.

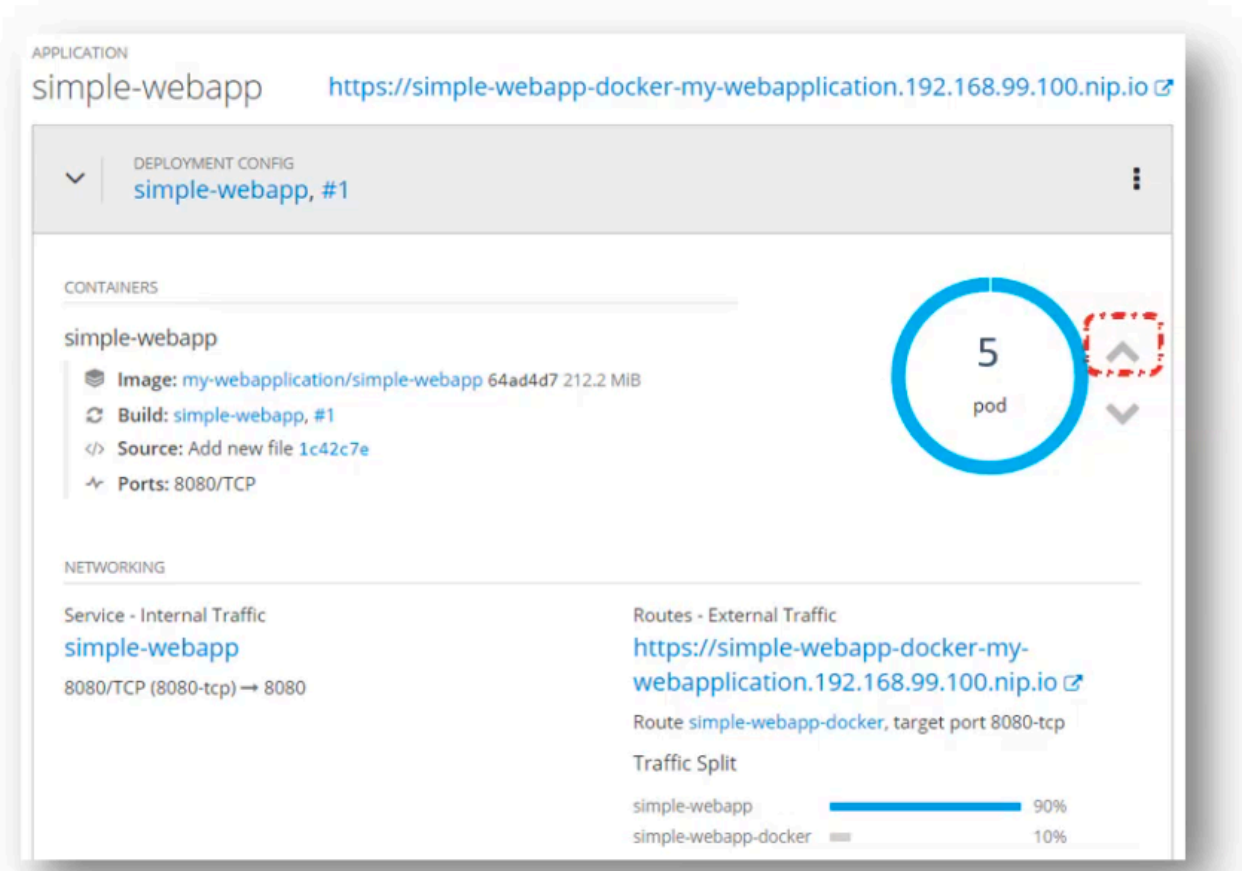
Scale




```
spec:
  replicas: 1
```

Scale (Contd..)


Replication Controller
Deployment Controller


APPLICATION
simple-webapp
https://simple-webapp-docker-my-webapplication.192.168.99.100.nip.io
DEPLOYMENT CONFIG
simple-webapp, #1
CONTAINERS
simple-webapp
Image: my-webapplication/simple-webapp 64ad4d7 212.2 MiB
Build: simple-webapp, #1
Source: Add new file 1c42c7e
Ports: 8080/TCP
5
pod
NETWORKING
Service - Internal Traffic
simple-webapp
8080/TCP (8080-tcp) → 8080
Routes - External Traffic
https://simple-webapp-docker-my-webapplication.192.168.99.100.nip.io
Route simple-webapp-docker, target port 8080-tcp
Traffic Split
simple-webapp 90%
simple-webapp-docker 10%

Agenda:

1) Cloud Pak
2) Types of Cloud Paks
3) Cloud Pak For Applications
4) Cloud Foundry Lab deployment

Agenda

10:00:am – 10:15 am	Introduction and General Information sharing	Shivanand Shenoi
10:15:am - 11:15am	Move & Modernize apps -Docker, k8s, OpenShift	Ishani Pandey
11:15am – 11:20am	5 mins break	
11:20am - 11:50 am	Introduction to IBM Cloud Pak for Applications + Cloud Foundry deployment on IBM Cloud	Ishani Pandey
11:50am - 12:00pm	10 mins	
12:00 pm 12:45 pm	Workshop	Mamatha Busi
12:45pm - 01:00pm	Q&A	

IBM Containerized Software's on OpenShift Container Platform

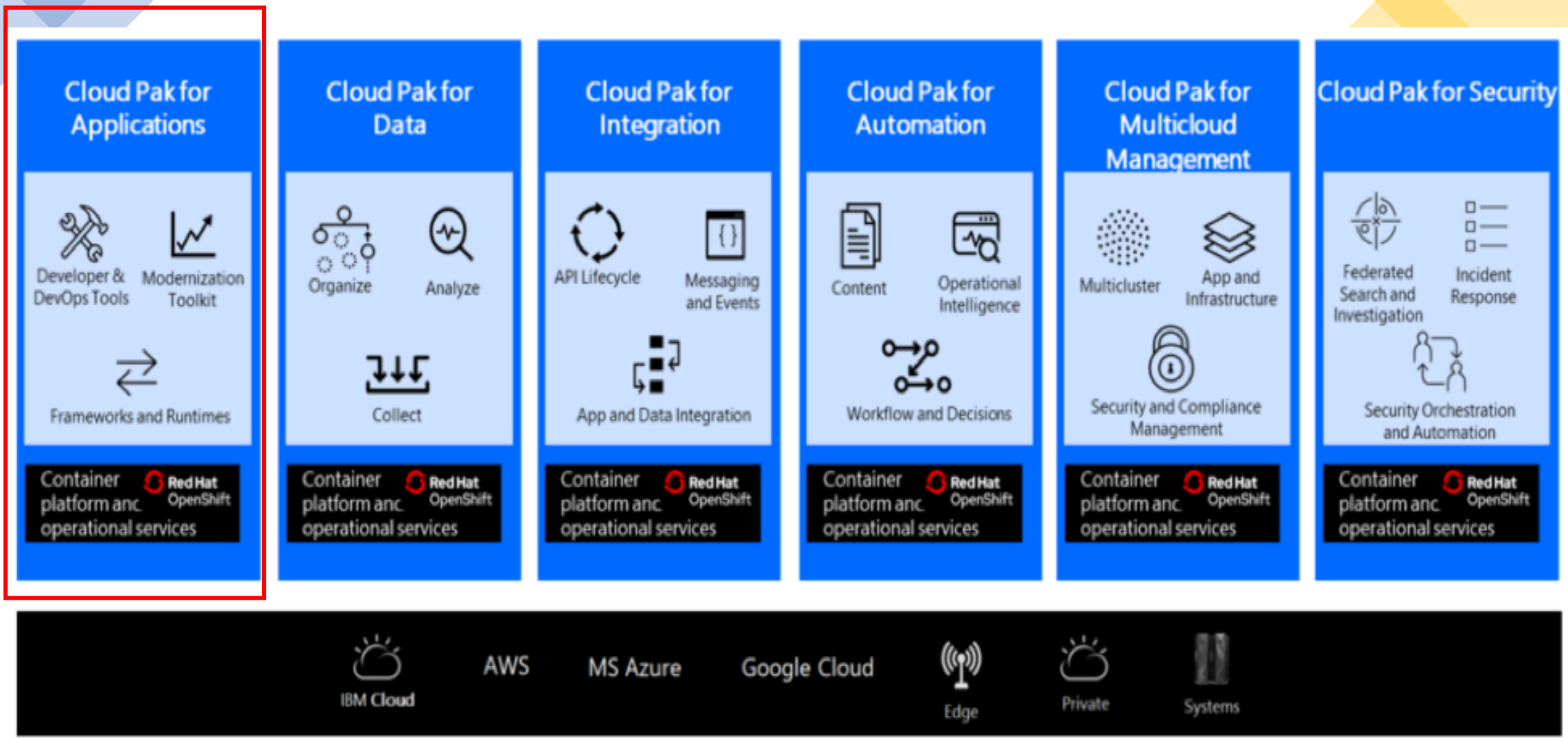
Cloud Paks – *Enterprise-ready Containerized Software*

A faster, more secure way to move your core business applications to any cloud through enterprise-ready containerized software solutions

Complete yet simple
Application, data and AI services, fully modular and easy to consume

IBM certified
Full software stack support, and ongoing security, compliance and version compatibility

Run anywhere
On-premises, on private and public clouds, and in pre-integrated systems


Cloud Pak for Applications
Developer & DevOps Tools
Modernization Toolkit
Frameworks and Runtimes
Container platform and operational services
Red Hat OpenShift
Cloud Pak for Data
Organize
Analyze
Collect
Container platform and operational services
Red Hat OpenShift
Cloud Pak for Integration
API Lifecycle
Messaging and Events
App and Data Integration
Container platform and operational services
Red Hat OpenShift
Cloud Pak for Automation
Content
Operational Intelligence
Workflow and Decisions
Container platform and operational services
Red Hat OpenShift
Cloud Pak for Multicloud Management
Multicluster
App and Infrastructure
Security and Compliance Management
Container platform and operational services
Red Hat OpenShift
Cloud Pak for Security
Federated Search and Investigation
Incident Response
Security Orchestration and Automation
Container platform and operational services
Red Hat OpenShift
IBM Cloud
AWS
MS Azure
Google Cloud
Edge
Private
Systems

Cloud Pak for Applications | 3 customer needs with 1 offering

Flexibly rebalance entitlement over time: from what you need today, to what you need tomorrow

Run existing apps | Continue to run your apps, where they are.

Modernize existing apps | When apps need to move, IBM has the most experience, tools, and experts to move them

Building new apps | New apps are automatically ready for hybrid-cloud deployment, using the best of open source, fully supported

Cloud Pak for Applications

Build, deploy and run applications in a modern, microservice-based framework

Key capabilities

Use cases

- Move and modernize applications - using insights about your current infrastructure to appropriately refactor, optimize resources and costs, reduce complexity and deliver apps on multiple clouds
- Develop cloud native apps with containers, starting with open source, common services, developer tools of choice, and integrated DevOps

Competitive differentiation

- Optimized set of frameworks and runtimes for cloud native and traditional
- Accelerate development with governance, supported by IBM enterprise expertise
- Portability across multiple cloud environments, avoiding vendor lock-in
- Investment protection as you modernize at your pace

#IBMDeveloper