

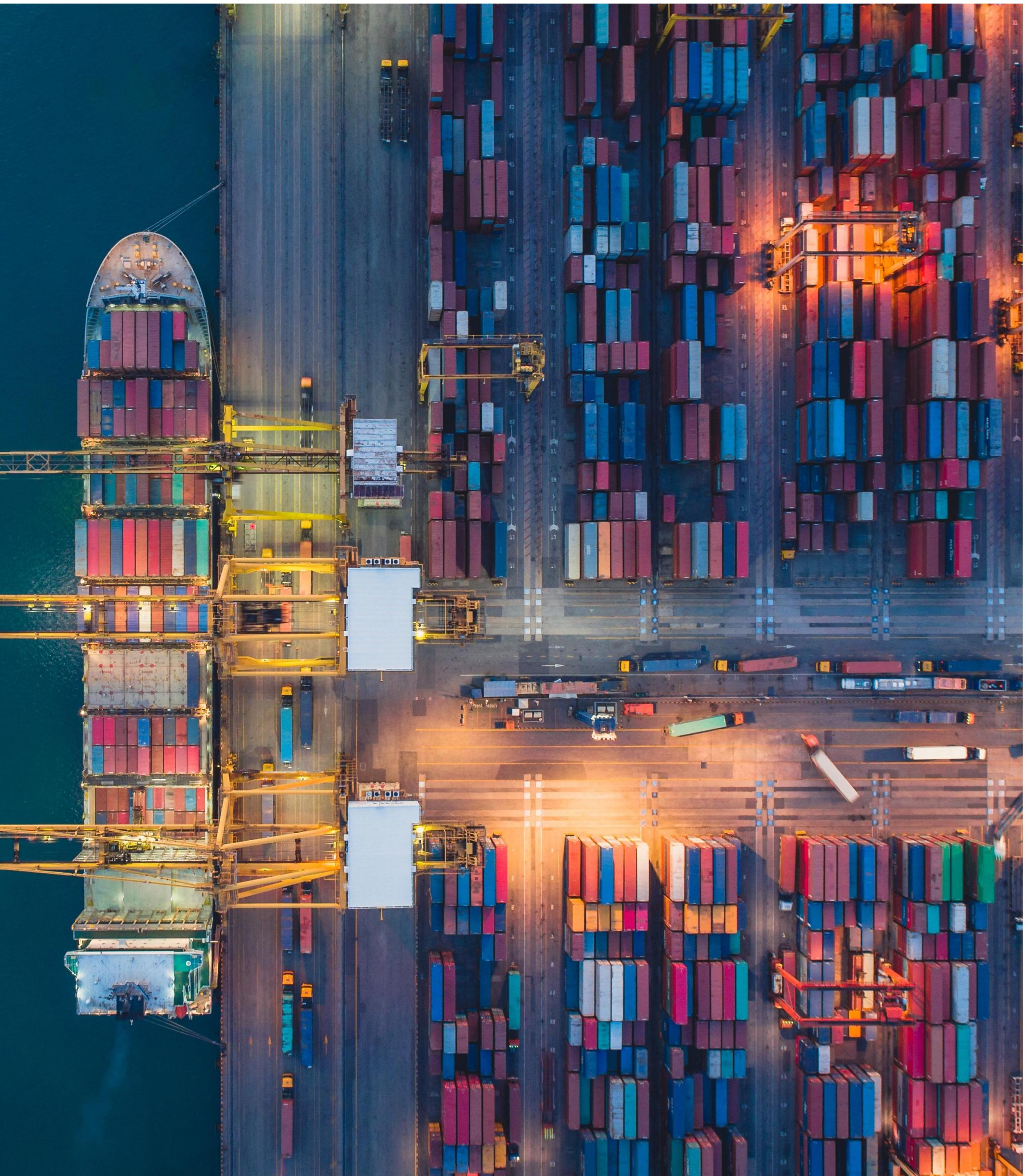
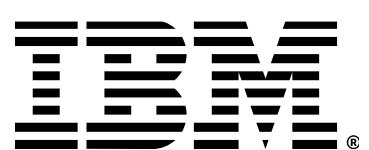
®

GRUNDING – IBM Tag 2

Platform as a Service

Alex Studer
Master@IBM Software Engineer

Alexander L. Schorno
Cross Software Partner Technical Specialist



Who are we?



Alex Studer

Master@IBM Software Engineer

alex.studer@ibm.com



Alexander L. Schorno

Cross Software Partner Technical Specialist

alexander.schorno@ibm.com

Agenda

Tag 1 – 17.10.
Cloud Computing

Onboarding IBM Cloud

Cloud Computing Value Proposition

Hands-on: Getting Started mit Rancher Desktop und Docker

Hands-on: Simple Python Webserver

Tag 2 – 24.10.
Platform as a Service

Einführung in Red Hat OpenShift als Platform as a Service Lösung

Hands-on: Mario Bros. on the Cloud

Hands-on: Simple Python Webserver

Hands-on: Shared Counter with Websockets

Tag 3 – 31.10.
Software as a Service

Recap: Software as a Service

Wie entwickelt man verantwortungsvolle KI im grossen Massstab

Hands-on: Watsonx assistant

Leistungskontrolle / Quiz

Unterlagen



Plattform as a Service

PaaS

Platform as a Service (PaaS)

Applications

Data

Runtime

Middleware

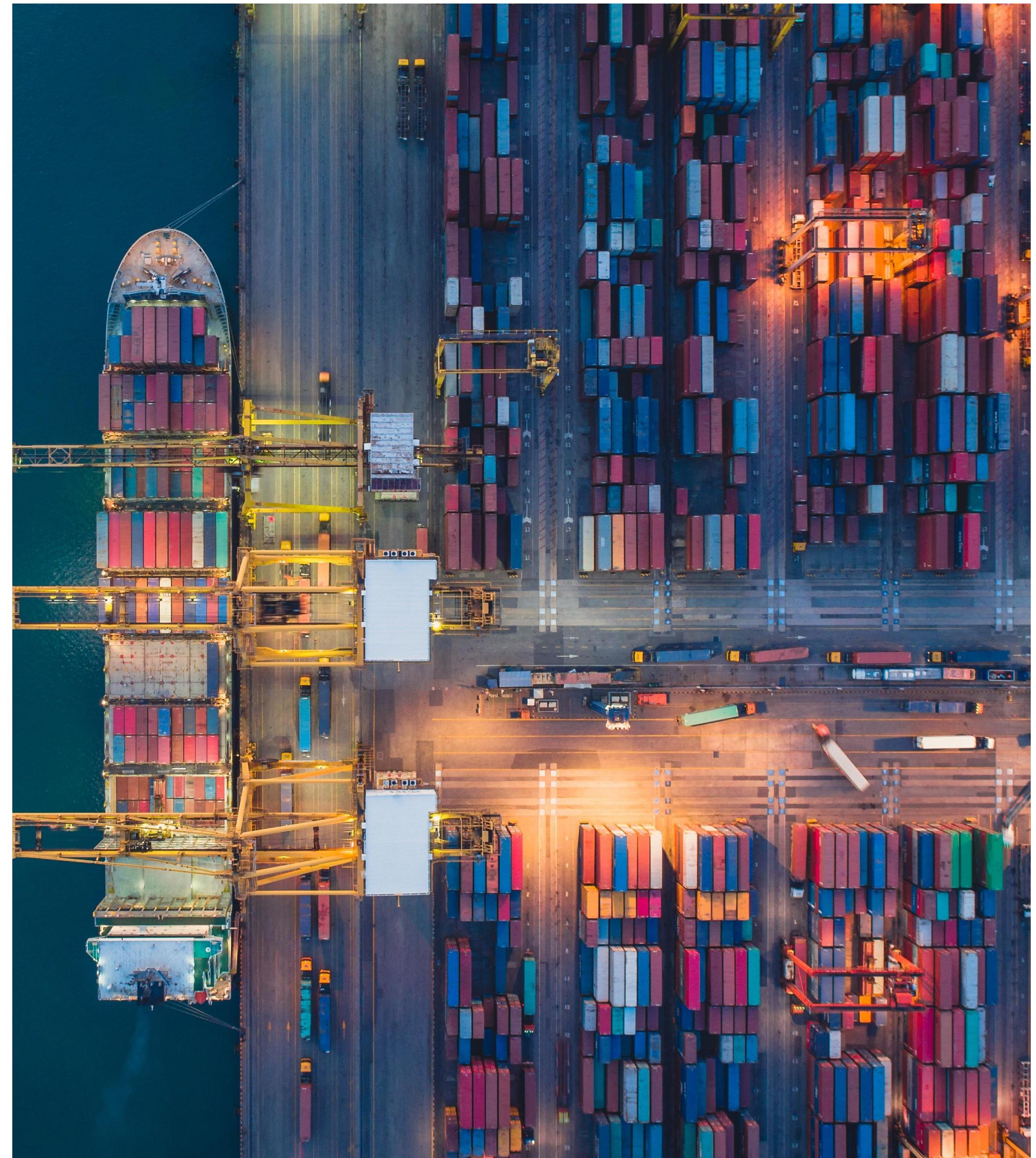
Operating System

Virtualization

Servers

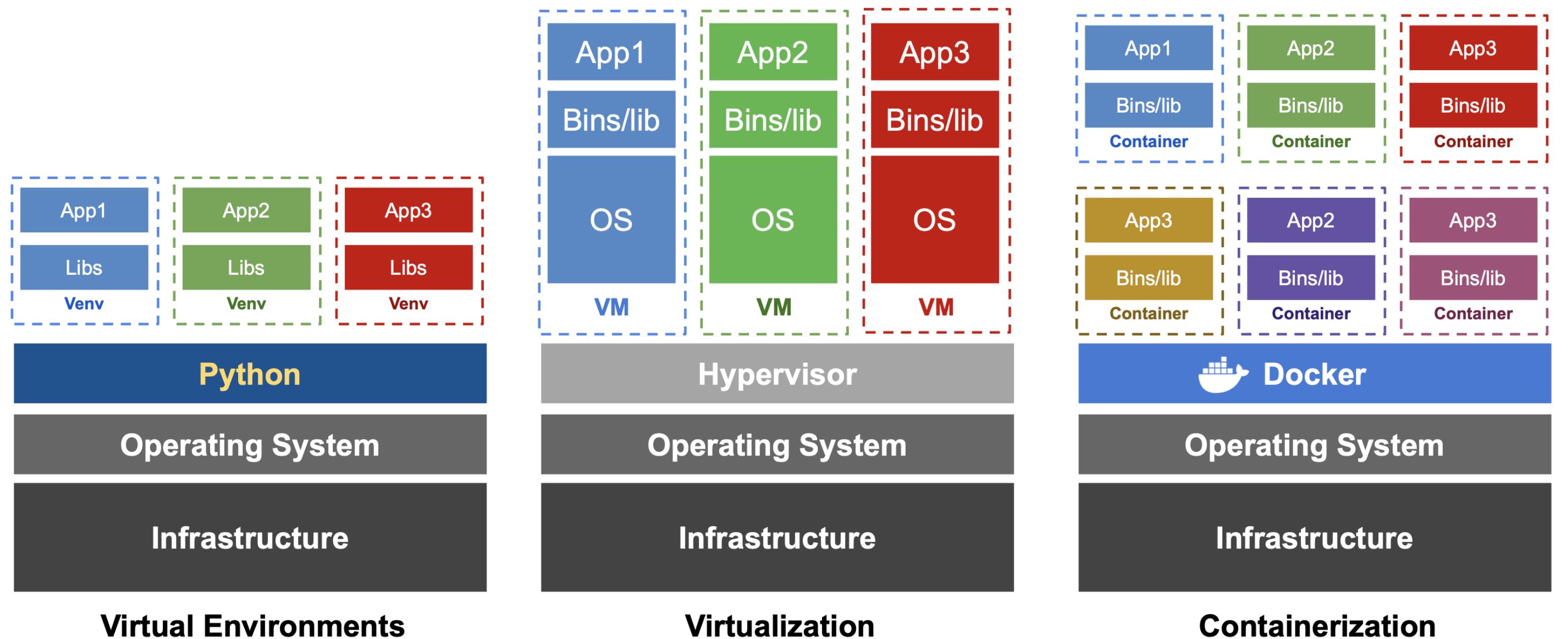
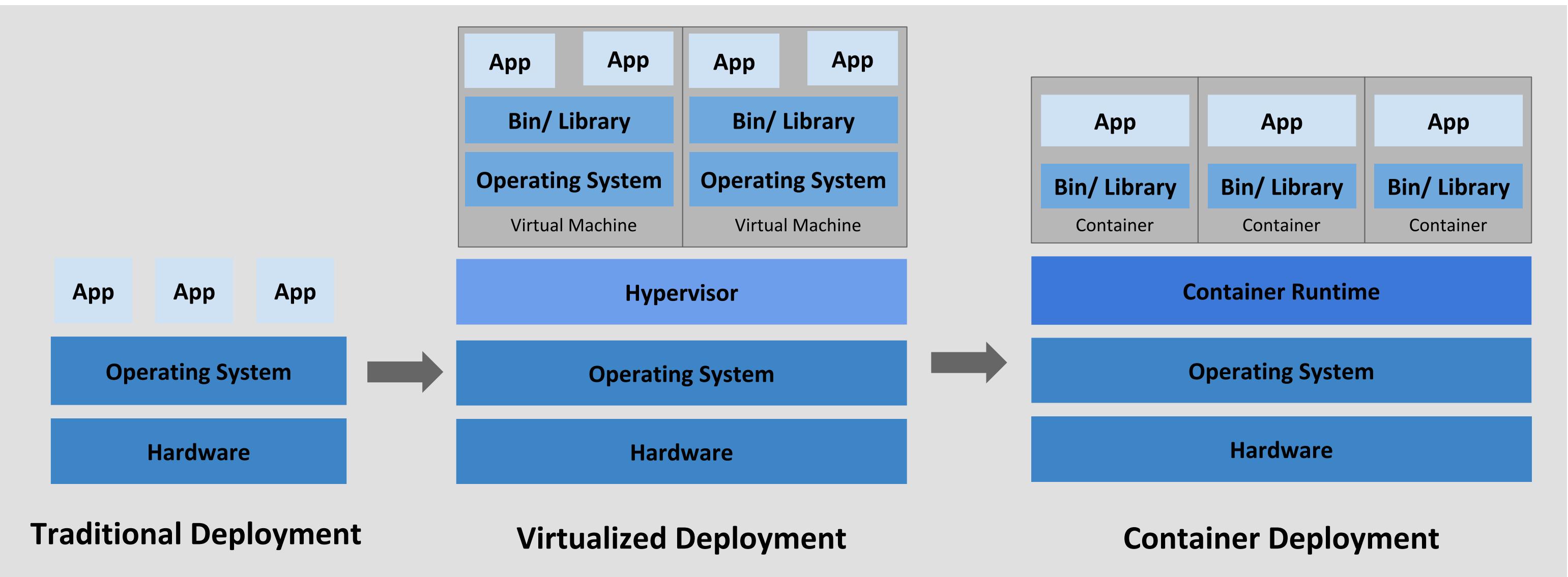
Storage

Networking



Eine kleine Geschichte der Virtualisierung

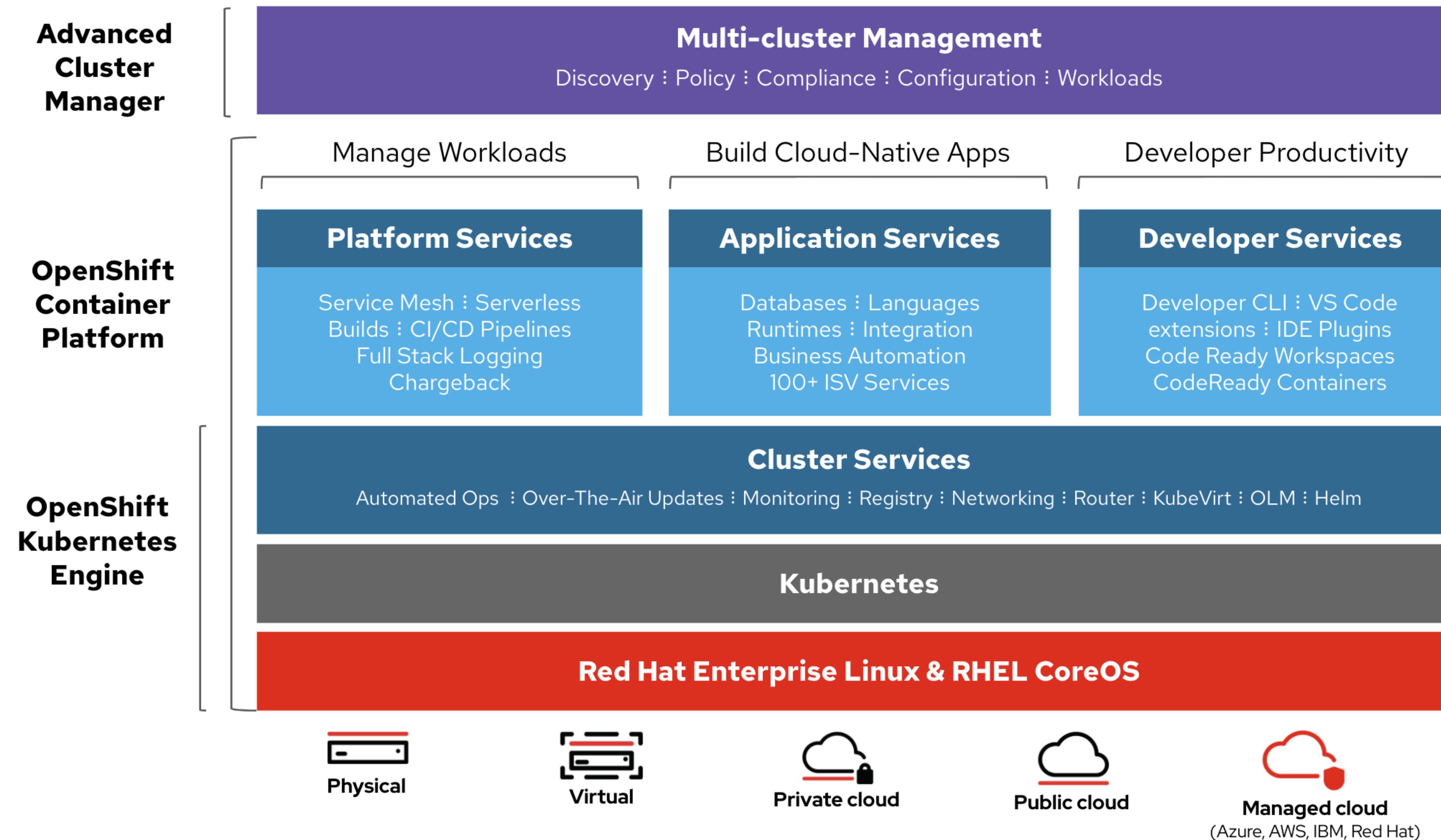
- 1960s: Time sharing on mainframes
- 1990s/2000s: Wide-spread adoption of hardware virtualization (server virtualization, virtual machines)
- 2008: Release Linux Containers (LXC)
- 2013: Release Docker
- 2017: Release OCI Image Spec



Container Management and Container Orchestration

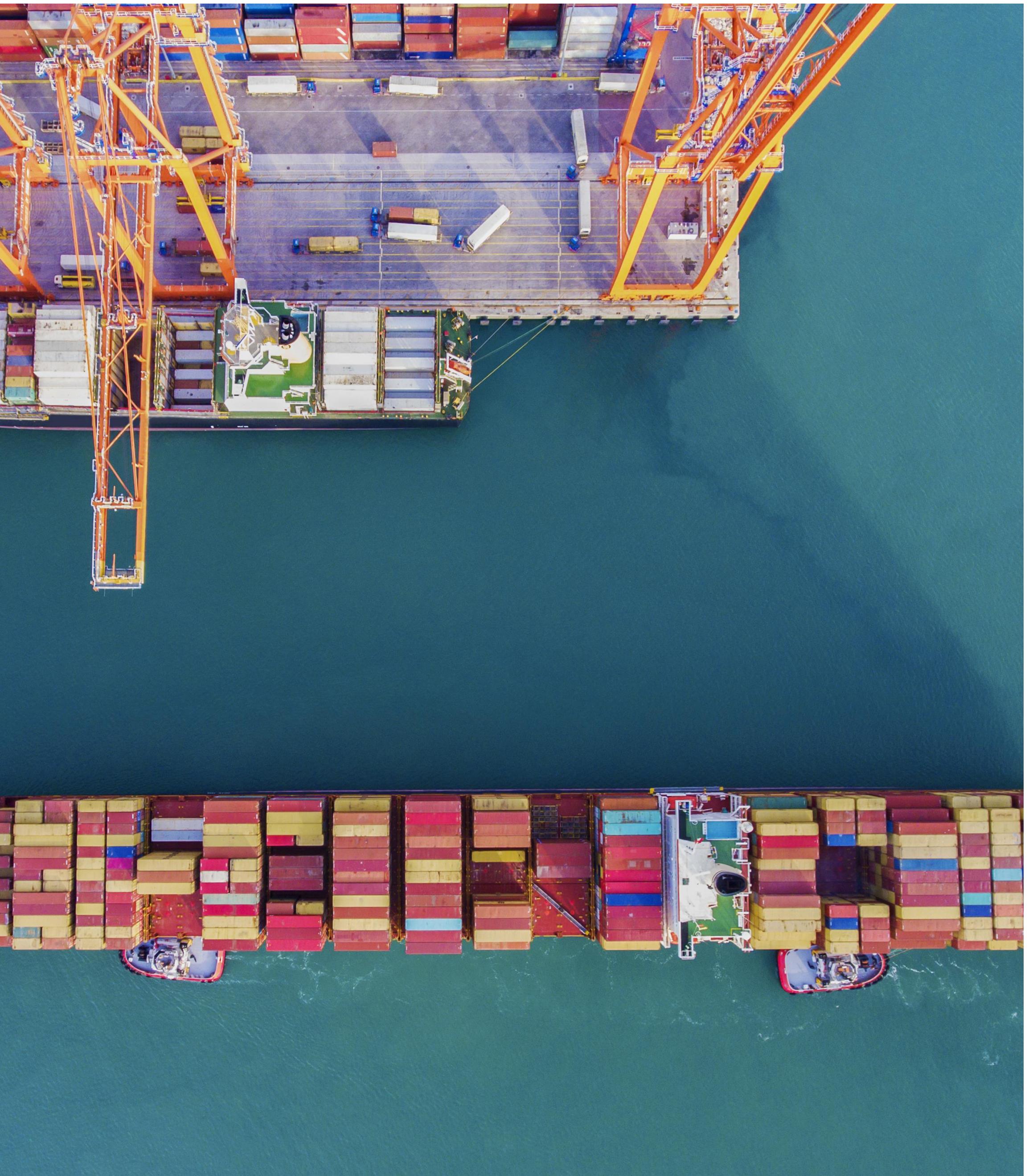


Red Hat OpenShift



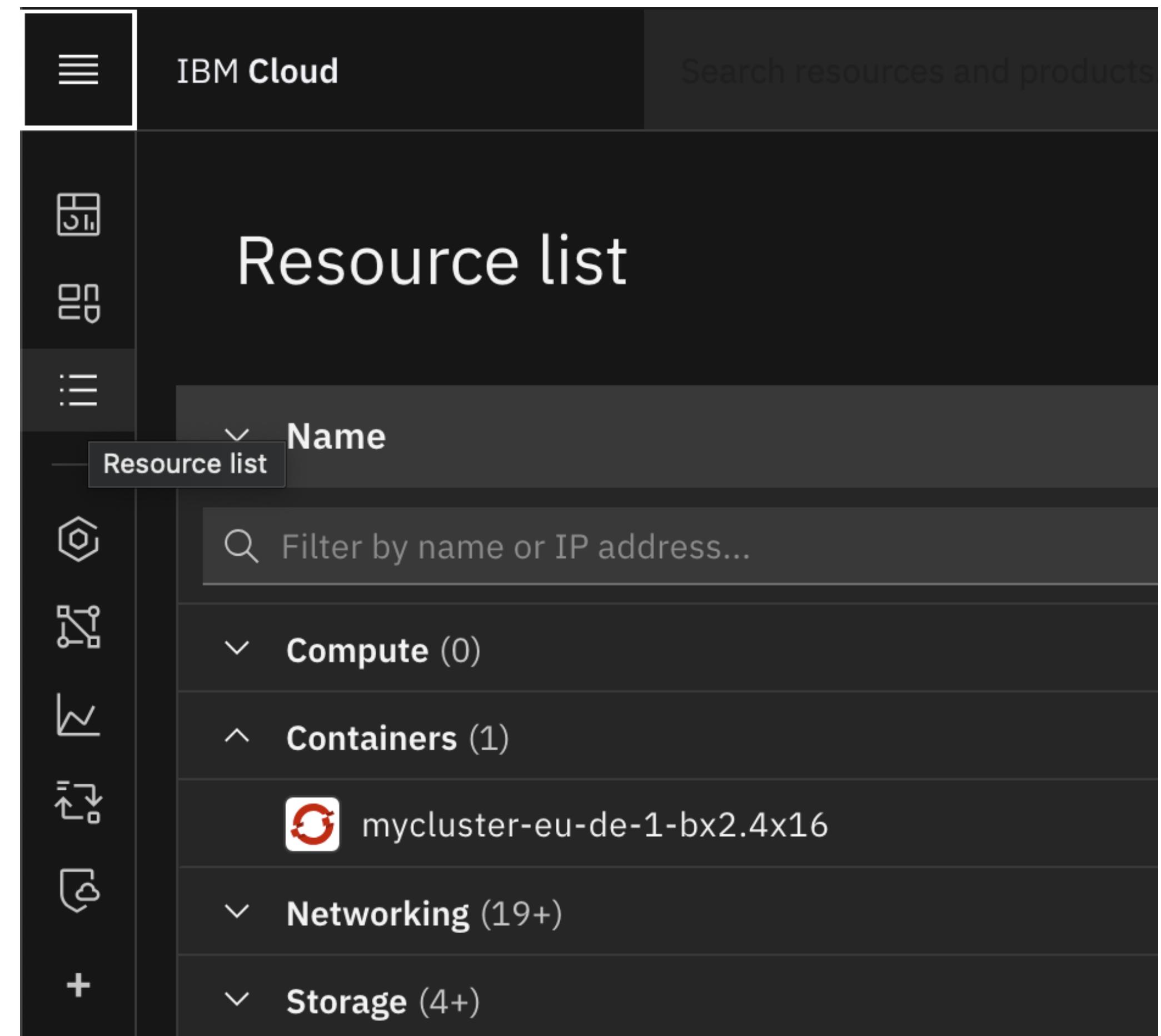
Hands-on PaaS

with Red Hat OpenShift



Open Red Hat OpenShift

- Öffnet die Resourcen Liste und navigiert zur OpenShift Platform «mycluster-eu-de-1-bx2.4x16» unter Containers (siehe Bild rechts)
- Wenn der Service geöffnet ist, könnt ihr via der blauen Schaltfläche «OpenShift web consol» das GUI von OpenShift öffnen.



This screenshot shows the OpenShift Container Management interface. The top navigation bar includes links for "Containers", "Cluster management", and "Clusters". The main content area displays a cluster named "mycluster-eu-de-1-bx2.4x16". Below the cluster name are status indicators: a green checkmark for "Normal" and a link to "Add tags". On the right side, there are buttons for "Help", "OpenShift web console" (which is highlighted in blue), and "Actions".

Red Hat OpenShift

- Wechselt in OpenShift auf die «Developer» Ansicht
- Mit dem drop-down Menu «Project» erreicht ihr den Button «Create Project» mit dem ihr ein neues Projekt anlegen könnt (siehe Bild rechts)
- Bennent euer Projekt nach dem folgenden Muster «student-vorname-nachname» und erstellt dieses (siehe Bild rechts)
- Navigiert mit dem Project drop-down Menü zu eurem projekt.

Create Project

An OpenShift project is an alternative representation of a Kubernetes namespace.

[Learn more about working with projects ↗](#)

Name * ⓘ

Display name

Description

[Cancel](#) [Create](#)

Project: All Projects ▾

Select project...

Show default projects

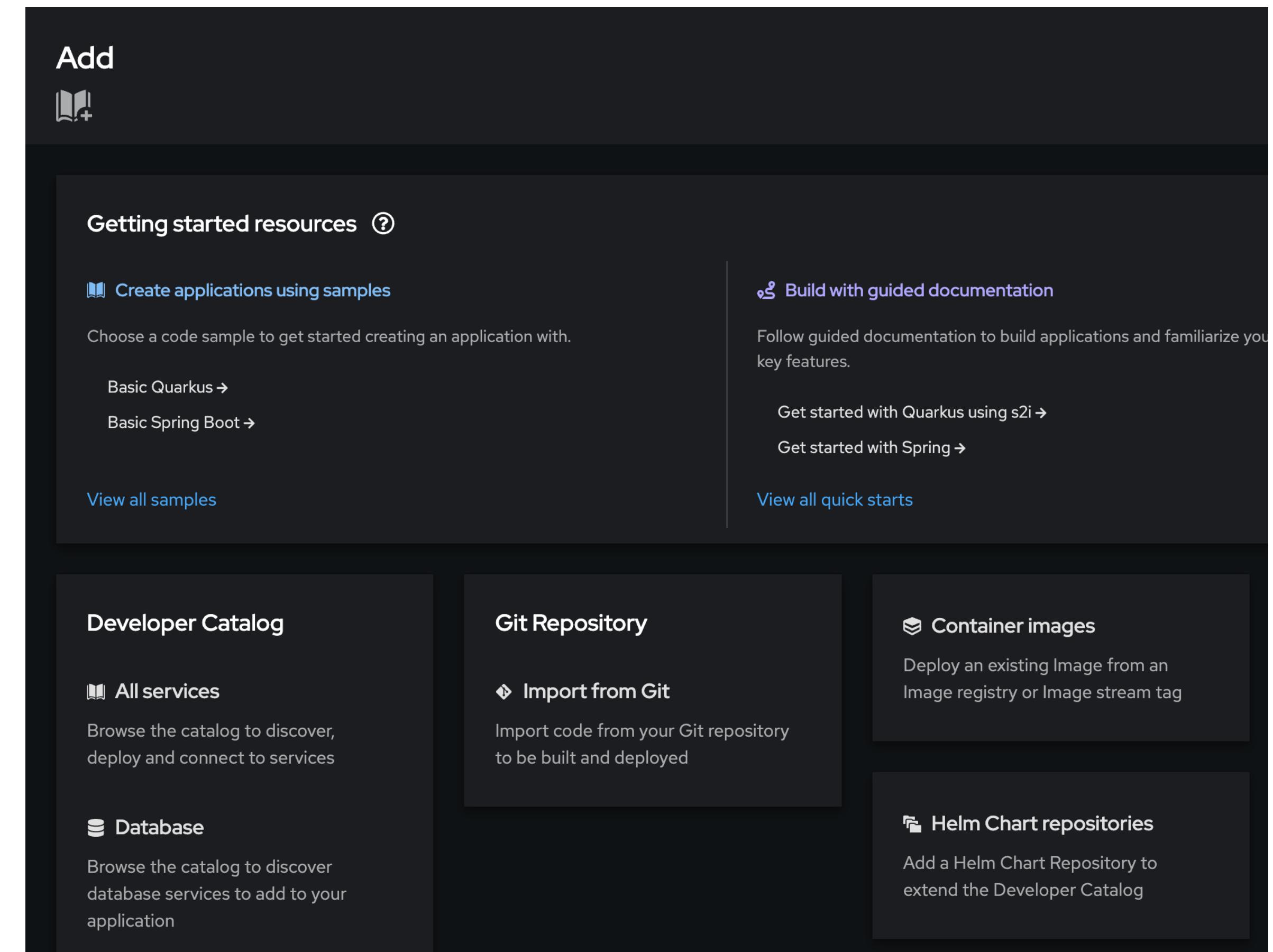
Projects	Star
All Projects	★
calico-system	★
demo-game	★
ibm-cert-store	★
ibm-odf-validation-webhook	★
ibm-system	★
student-alex-studer	★
student-alexander-schorno	★
tigera-operator	★

[Create Project](#)

Exercise 1

Mario Bros. on the cloud

- Bleibt in der «Developer» Ansicht für dieses und alle folgenden Übungen
- Wechselt zur «Add» Oberfläche über die Navigation auf der Linkenseite des GUIs
- Wählt die Schaltfläche «Container Images» aus



Exercise 1

Mario Bros. on the cloud

- Fügt den Namen des Container Images ein (analog zum Bild rechts) -> pengbai/docker-supermario
- Alles andere sollte dann automatisch ergänzt werden.
- Drückt auf «Create» und wechselt anschliessend auf die «Topology» Ansicht.

Deploy Image

Image

Deploy an existing Image from an Image Stream or Image registry.

Image name from external registry

`pengbai/docker-supermario` 

Validated

To deploy an Image from a private registry, you must [create an Image pull secret](#) with your Image registry credentials.

Allow Images from insecure registries

Image stream tag from internal registry

Runtime icon

 openshift 

The icon represents your Image in Topology view. A label will also be added to the resource defining the icon.

General

Application name

`docker-supermario-app`

A unique name given to the application grouping to label your resources.

Name *

`docker-supermario`

A unique name given to the component that will be used to name associated resources.

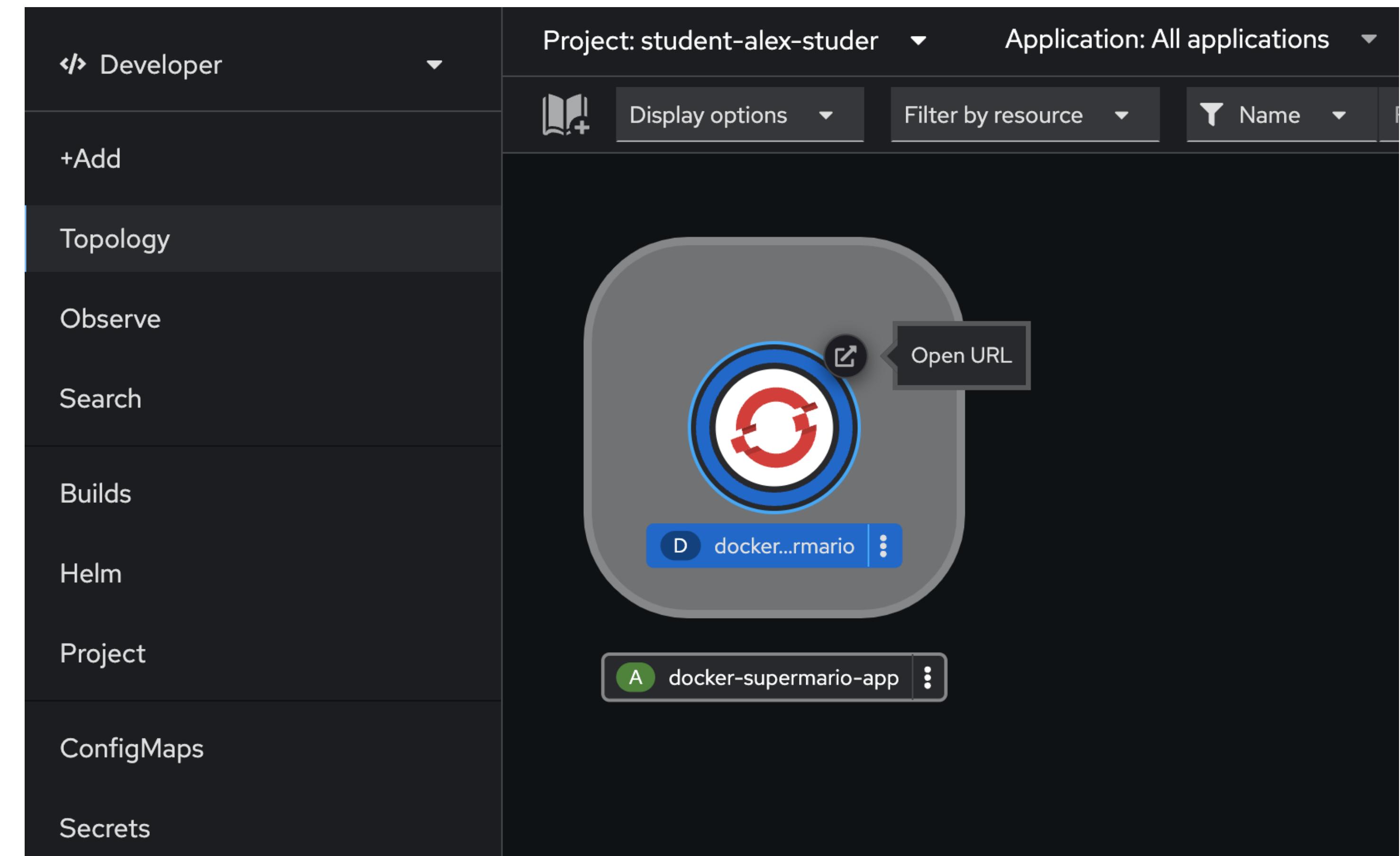
Resource type

Create **Cancel**

Exercise 1

Mario Bros. on the cloud

- Sobald der Ring um das OpenShift Logo dunkelblau eingefärbt ist, solltet ihr den «Open URL» Button sehen.
- Wenn ihr diesen drückt, sollte in einem neuen Fester Supermario laufen.
- Öffnet die selbe URL auf eurem Mobiltelefon oder einem andere Gerät.
Habt ihr zugriff darauf?



Exercise 2

Simple Python Webserver

- Wählt auf der «+Add» Ansicht «Import from Git» aus.
- Kopiert die folgende Git Repo URL (siehe Bild rechts):
`https://github.com/alexreduts/cloud-computing-lectures.git`
- Gebt an welcher Unterordner ausgewählt werden soll.
`/src/simple-python-webserver`

Git

Git Repo URL *

`https://github.com/alexreduts/cloud-computing-lectures.git` 

Validated

 Hide advanced Git options

Git reference

Optional branch, tag, or commit.

Context dir

`/src/simple-python-webserver`

Optional subdirectory for the source code, used as a context directory for build.

Source Secret

Select Secret name 

Secret with credentials for pulling your source code.

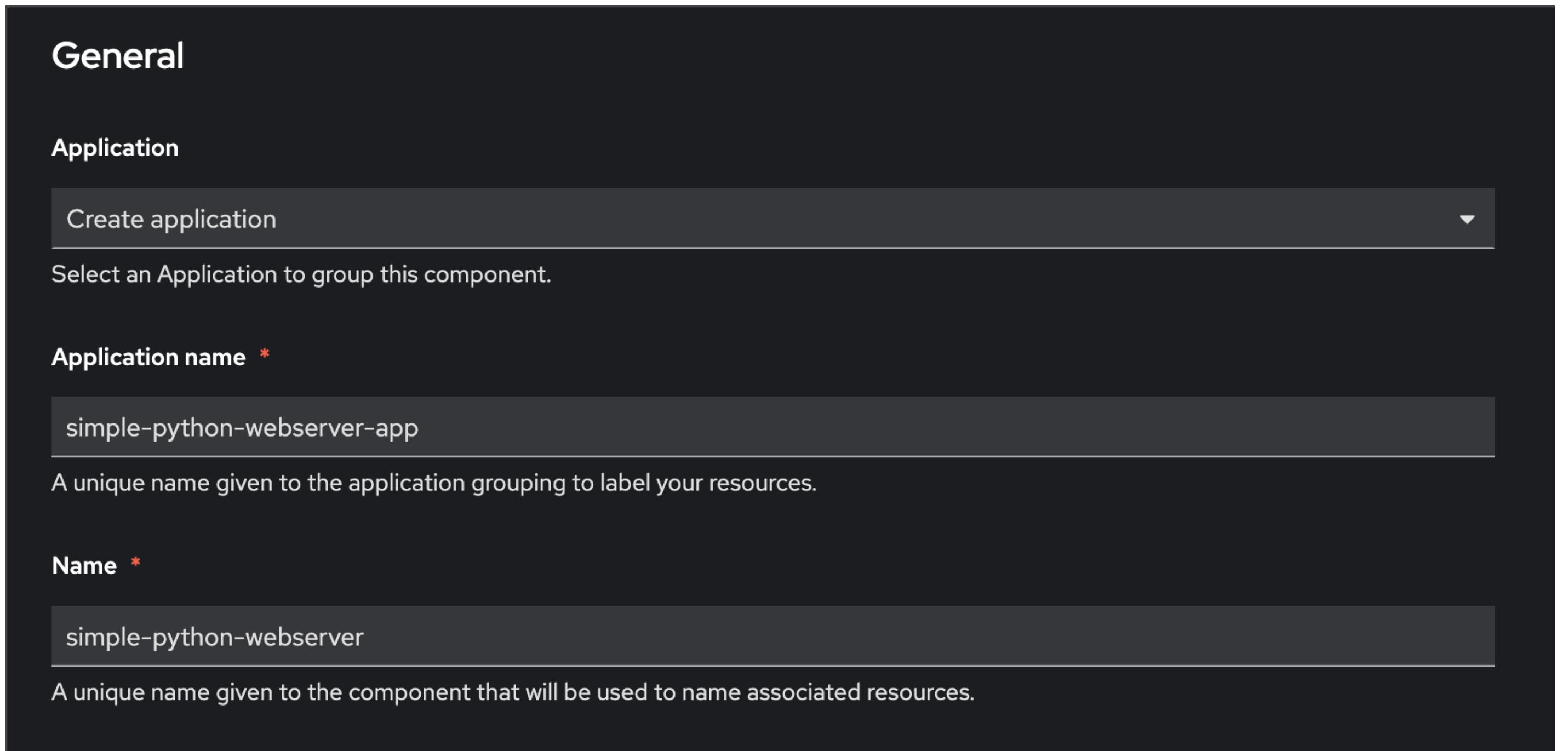
 **Multiple import strategies detected**
The Dockerfile at Dockerfile is recommended.

 Dockerfile  Edit Import Strategy

Exercise 2

Simple Python Webserver

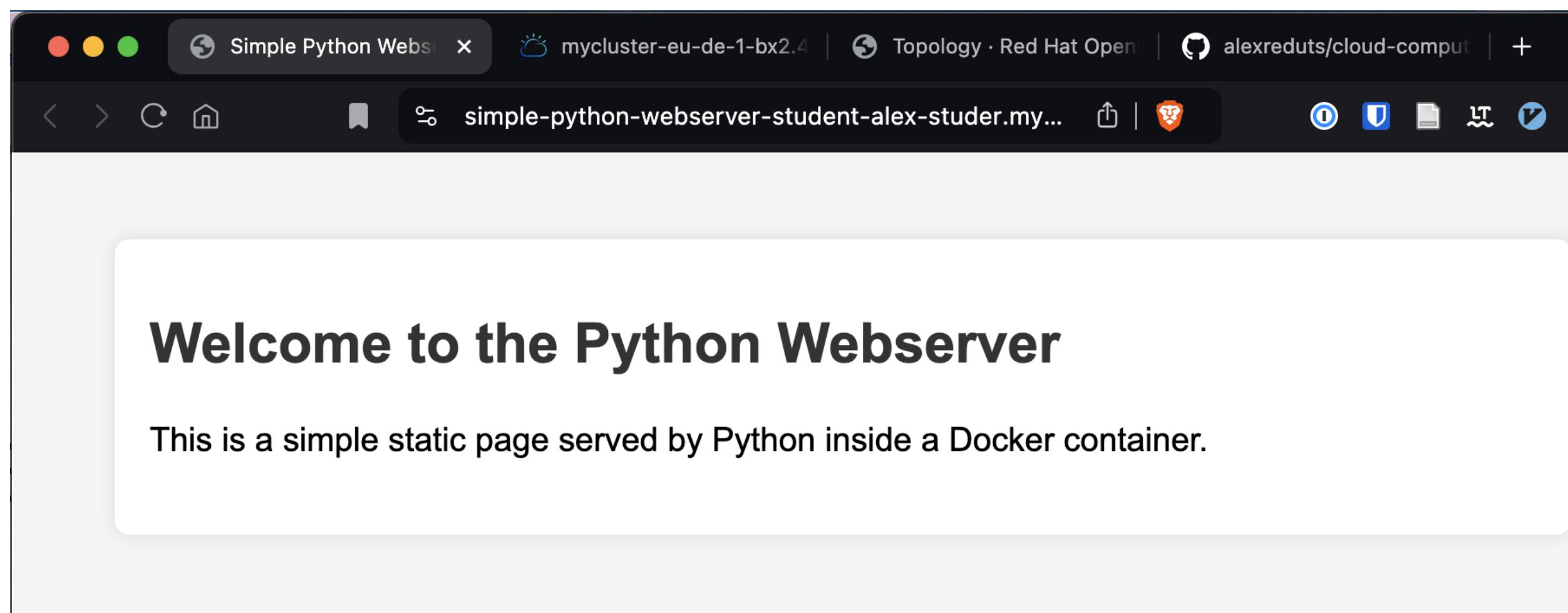
- Unter General>>Application wählt im drop-down Menu «Create application» aus. (siehe Bild rechts untern)
- Gebt der neuen Applikation einen eigenen Namen unter «simple-python-webserver-app», «simple-python-webserver»
- Den Rest lässt ihr unverändert und führt wieder «Create» aus



Exercise 2

Simple Python Webserver

- Wenn ihr diesmal auf das OpenShift Symbol in der «Topology» Ansicht drückt, solltet ihr eine Ansicht analog dem Bild rechts sehen. Wartet bis unter «Pods» eine Pod ressource existiert, die als «running» gekennzeichnet ist.
- Sobald der Container läuft könnt ihr auf den Link unter «Routes» clicken und ein neuer Tab sollte sich öffnen mit einer simplen Hello World Nachricht.

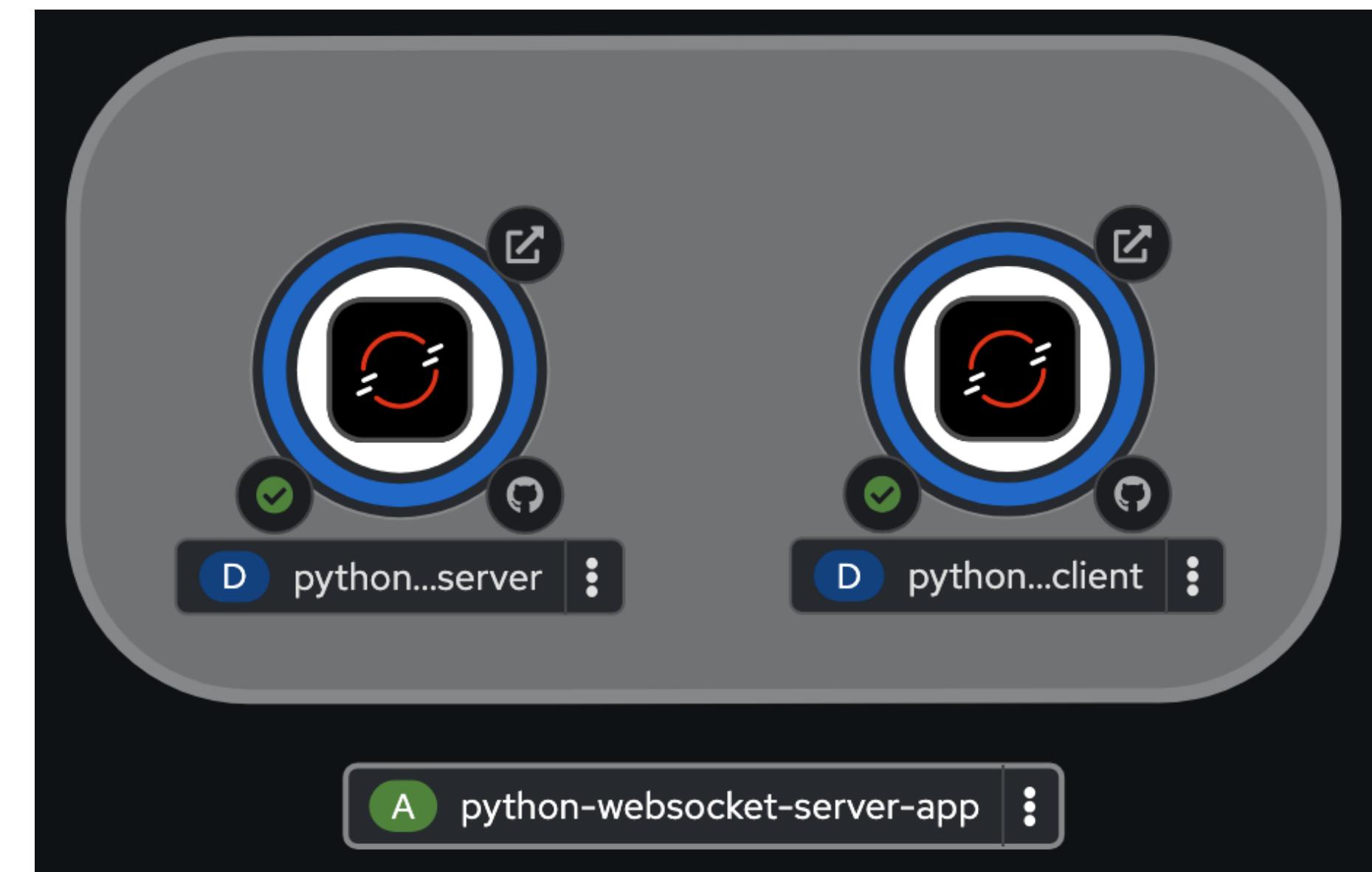
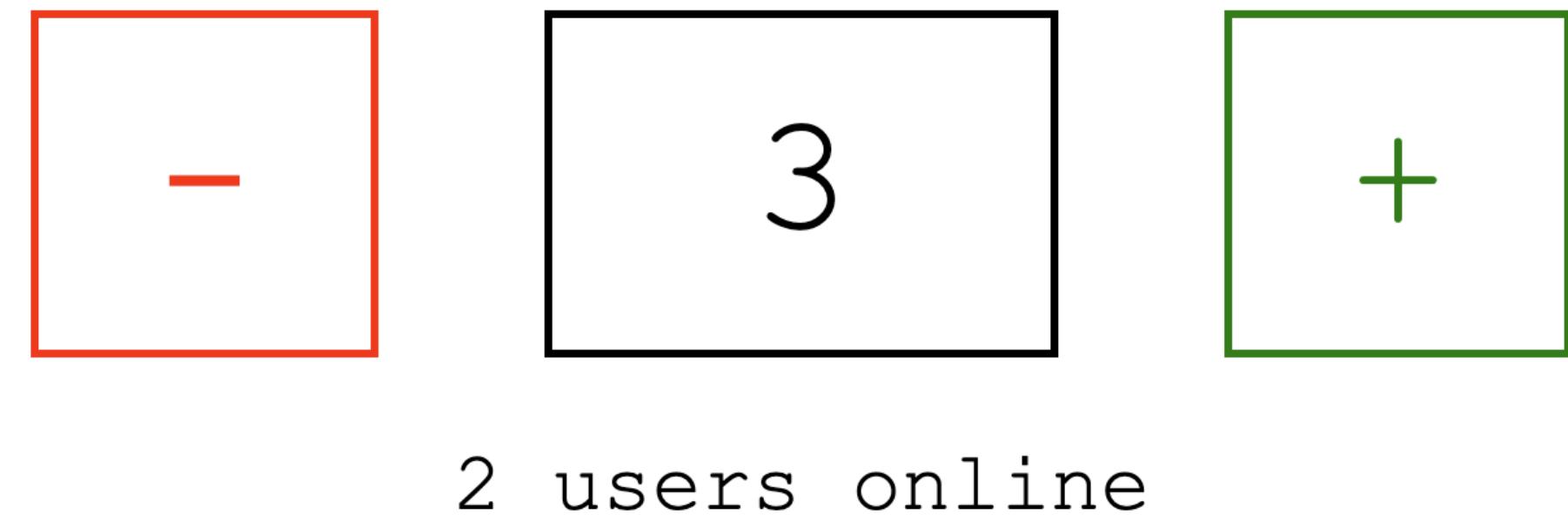
A screenshot of the OpenShift web interface. The top navigation bar shows "D simple-python-webserver" and an "Actions" dropdown. Below it, there are tabs for "Details", "Resources" (which is selected), and "Observe".

- Pods:** A table entry for "simple-python-webserver-758cf5dc9f-s9981" shows it is "Running". There is a "View logs" button next to it.
- Builds:** A table entry for "simple-python-webserver" shows a completed build "#1" from "Just now". There is a "Start Build" button and a "View logs" button.
- Services:** A table entry for "simple-python-webserver" shows the service port "8080-tcp" mapped to the pod port "8080".
- Routes:** A table entry for "simple-python-webserver" shows the route location as "https://simple-python-webserver-student-alex-studer.mycluster-eu-de-1-244439-975dd9665934b81fa7342475b7855171-0000.eu-de.containers.appdomain.cloud". There is a "View logs" button next to it.

Exercise 3

Shared Counter with Websockets

- Diese Übung funktioniert etwas anders. Es ist nur beschrieben was ihr machen müsst aber nicht wie. Überlegt wie ihr das gelernte der vorangegangen Übungen adaptieren könnt.
- Ziel: Ihr deployed zwei Container die zur selben App gehören. Einer beinhaltet ein Websocket Server, welcher den Stand eines Zählers verwaltet. Der andere ist ein HTTP Server der eine Websocket Client zur Verfügung stellt, wenn er angefragt wird. Diese Client können alle in Echtzeit auf den Zähler zugreifen und ihn erhöhen oder erniedrigen.



Exercise 3

Shared Counter with Websockets

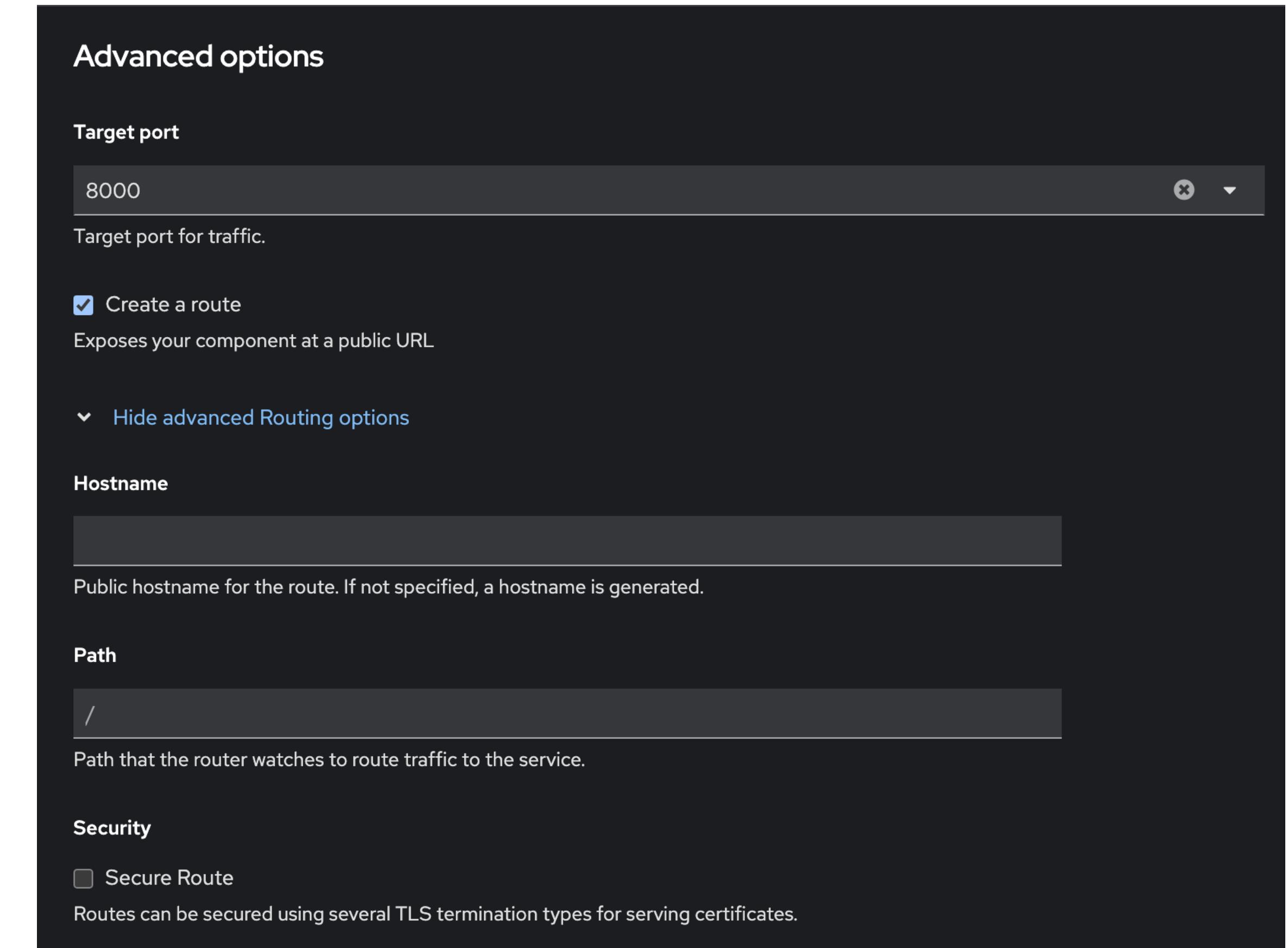
1. Erstellt einen [github.com](#) Account falls ihr noch keinen besitzt.
2. Erstellt eine **Fork** von meinem Repo (<https://github.com/alexreduts/cloud-computing-lectures.git>)
3. Findet einen Weg um euer eigenen **Fork** zu bearbeiten. (Es gibt min. 2 Möglichkeiten). Verwendet von nun an nur noch den Code in eurem eigenen **Fork**.



Exercise 3

Shared Counter with Websockets

1. Beginnt mit dem «src/server-backend» Teil
2. Beachtet dass ihr die «Advanced options» entsprechend dem Bild rechts anpasst



Exercise 3

Shared Counter with Websockets

1. Für den «src/web-frontend» Teil müsst eine Zeile Code anpassen.
Hinweis: Der Websocket Client muss wissen wo der Websocket Server zu finden ist.
2. Beachtet wieder dass ihr die «Advanced options» entsprechend dem Bild rechts anpasst.

General

Application
python-websocket-server-app

Select an Application to group this component.

Name *
python-websocket-client

A unique name given to the component that will be used to name associated resources.

Advanced options

Target port
8080

Target port for traffic.

Create a route
Exposes your component at a public URL

[Hide advanced Routing options](#)

Hostname

Public hostname for the route. If not specified, a hostname is generated.

Path
/

Path that the router watches to route traffic to the service.

Security

Secure Route

Routes can be secured using several TLS termination types for serving certificates.

