Mobile and Cloud Computing #3

Containers Services

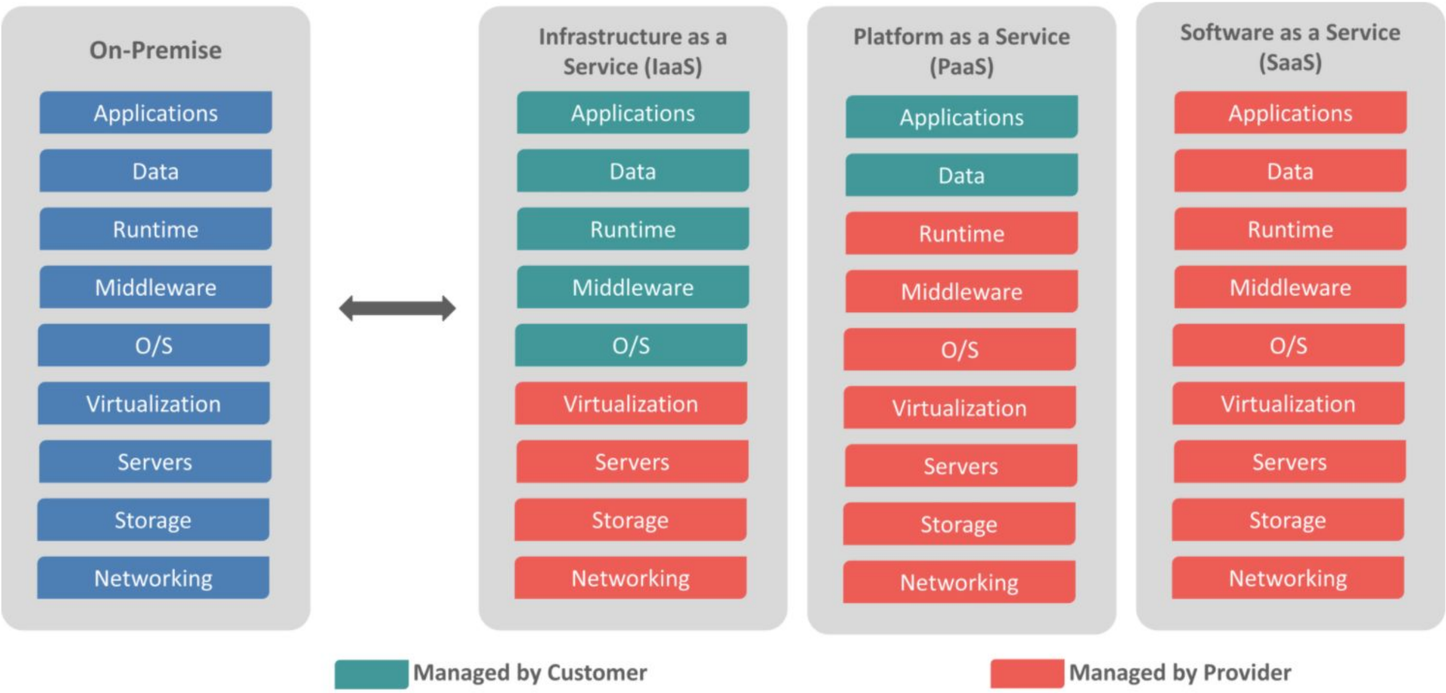
Overview

1. Background
2. Running containers on VMs
3. Running containers on App Engine and Cloud Run
4. Running containers on Kubernetes Engine

What are containers?

Containers are a **method of packaging** an application executable and its dependencies (runtime, system tools, system libraries, configuration), and **running the package as a set of resource-isolated processes**

Type of Service on Cloud

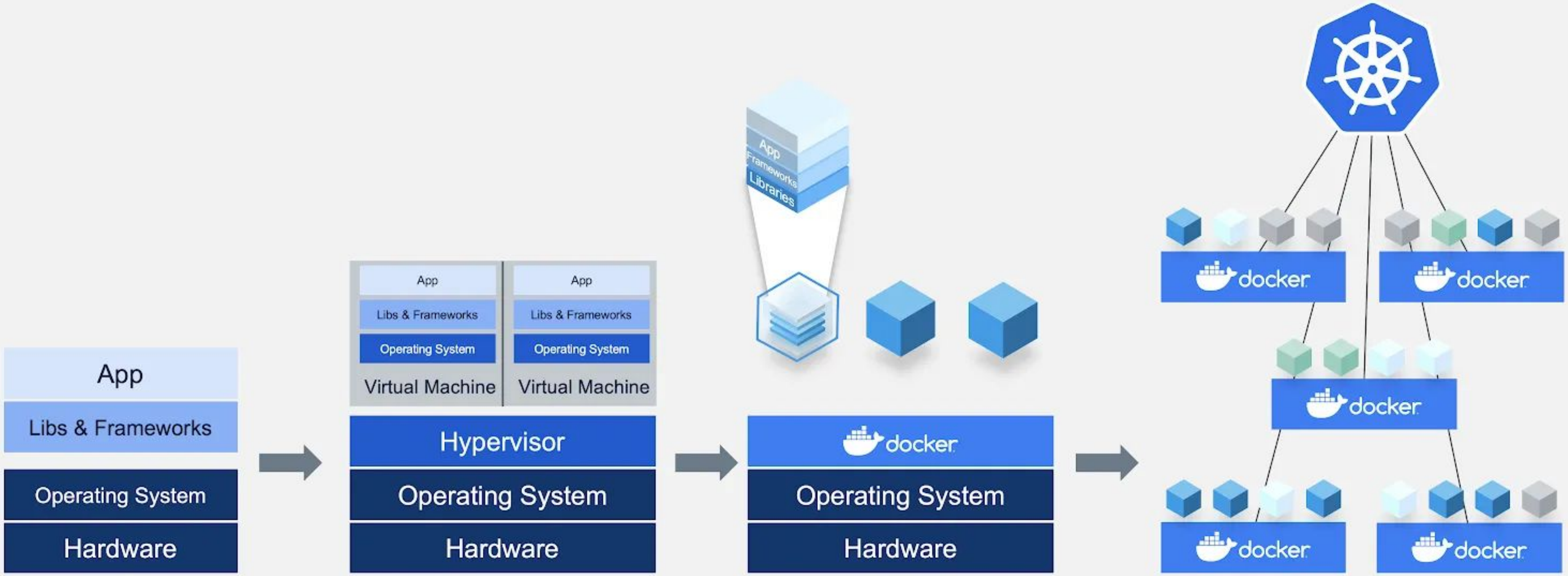


On-Premise	↔	Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
🟦 Applications		🟩 Applications	🟩 Applications	🟥 Applications
🟦 Data		🟩 Data	🟩 Data	🟥 Data
🟦 Runtime		🟩 Runtime	🟥 Runtime	🟥 Runtime
🟦 Middleware		🟩 Middleware	🟥 Middleware	🟥 Middleware
🟦 O/S		🟩 O/S	🟥 O/S	🟥 O/S
🟦 Virtualization		🟥 Virtualization	🟥 Virtualization	🟥 Virtualization
🟦 Servers		🟥 Servers	🟥 Servers	🟥 Servers
🟦 Storage		🟥 Storage	🟥 Storage	🟥 Storage
🟦 Networking		🟥 Networking	🟥 Networking	🟥 Networking

🟩 Managed by Customer

🟥 Managed by Provider

Kubernetes & Docker work together to build & run containerized applications



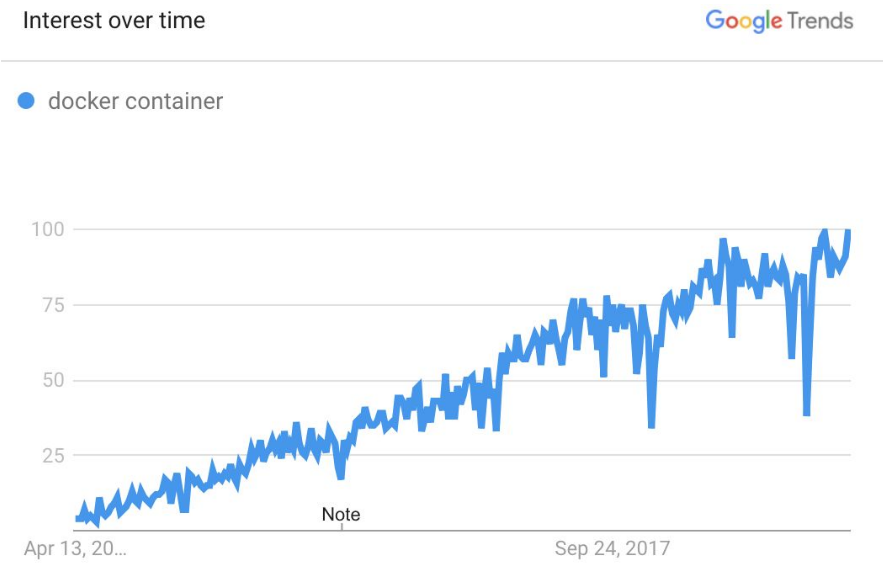
Traditional Deployment

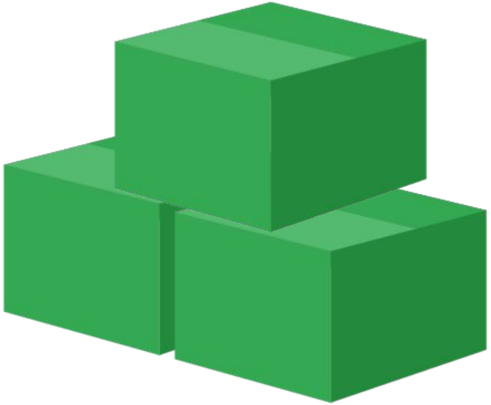
Virtualized Deployment

Container Deployment

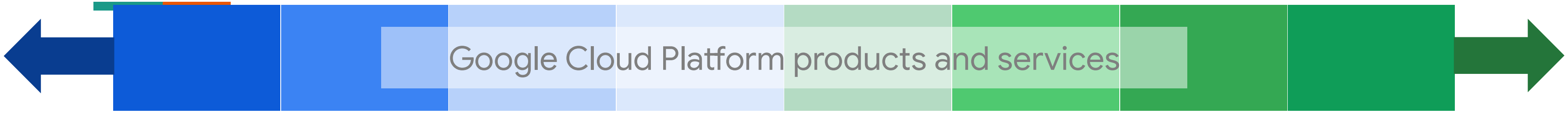
Kubernetes Deployment

The growth of containers





Complexity and control



IaaS

PaaS

SaaS

Servers,
VM instances

Clusters,
cluster management

Serverless, autoscaling

Compute Engine

Kubernetes Engine

App Engine flexible environment

App Engine

Cloud Run

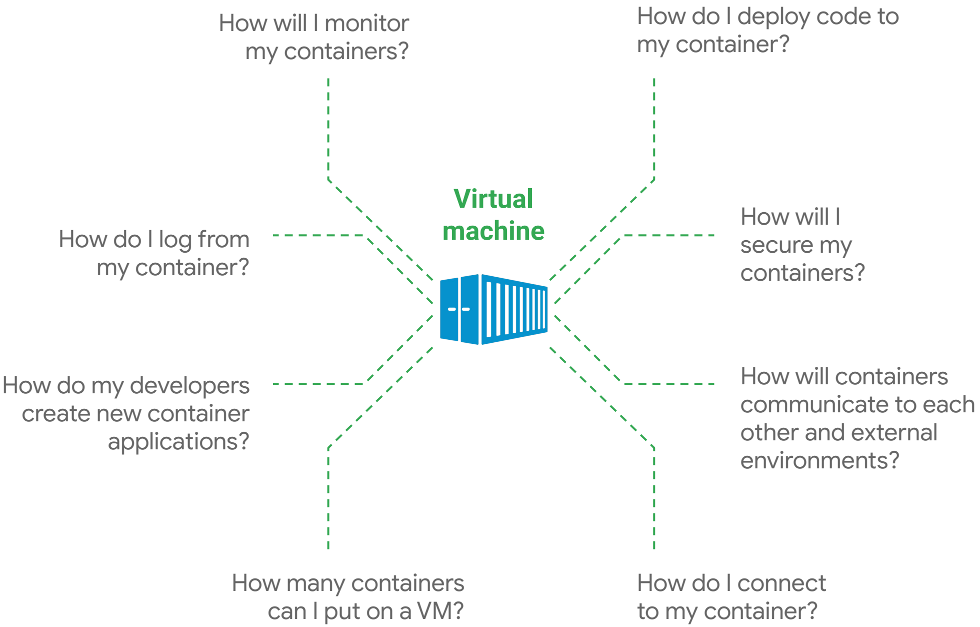
Running containers on VMs

02

Working with containers on VMs

As you start to consider what running containers on VMs might look like, you quickly realize there is a lot to think about ...

- Container availability and fault tolerance
- Logging and monitoring
- Security and networking
- Development and deployment
- Performance and scalability
- Available resources and skill sets



Virtual machine
How will I monitor my containers?
How do I deploy code to my container?
How do I log from my container?
How will I secure my containers?
How do my developers create new container applications?
How will containers communicate to each other and external environments?
How many containers can I put on a VM?
How do I connect to my container?

Google Cloud

Infrastructure considerations

How do you support all these considerations when running your infrastructure on VMs?

- Container availability and fault tolerance
- Logging and monitoring
- Security and networking
- Development and deployment
- Performance and scalability
- Available resources and skill sets

Build or modify applications to support these concerns? **Code**

Are the resources and skill sets available to tackle these challenges? **Staff**

Impact to VM management and infrastructure? **VMs**

Buy container-specific software tools to address these concerns? **Tools**

Google Cloud

Getting help from the cloud

The major players in the cloud platform space all have container services that come with orchestration tools already integrated to work in a VM environment and have tackled some of these more difficult issues, like ....

- Elasticity of the container infrastructure is in sync with the elasticity of the VMs
- Managing state by offering mechanisms to persist data in external storage
- Communication between the container and the VM using software-defined networking

Google Cloud Platform

Azure

amazon web services™

Pros and cons

✓

- Custom control of your container infrastructure
- No vendor lock-in
- Optimization of the infrastructure for you specific requirements and constraints
- Easier to meet compliance requirements as container infrastructures could be privately hosted

✗

- Increased complexity that needs to be addressed
- Time and effort to keep the infrastructure up and running efficiently
- Timeliness of implementing changes
- Level of expertise in running container infrastructures
- Managing staff retention of highly skilled resources

QWIKLABS

GSP282

A Tour to QWIKLABS and Google Cloud

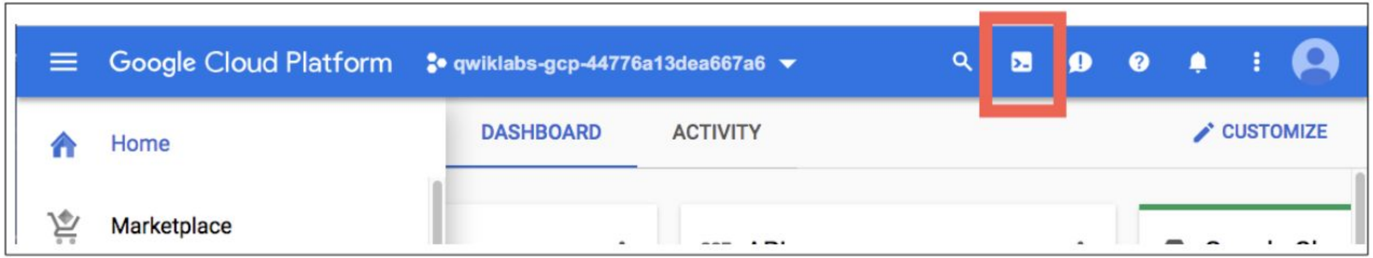
45 minutes Free Rate Lab

Google Cloud

Cloud Ace

Lab : Docker

1. Go to Cloud Console https://console.cloud.google.com/



2. Check current directory
~$ pwd

Lab : Docker play with Docker

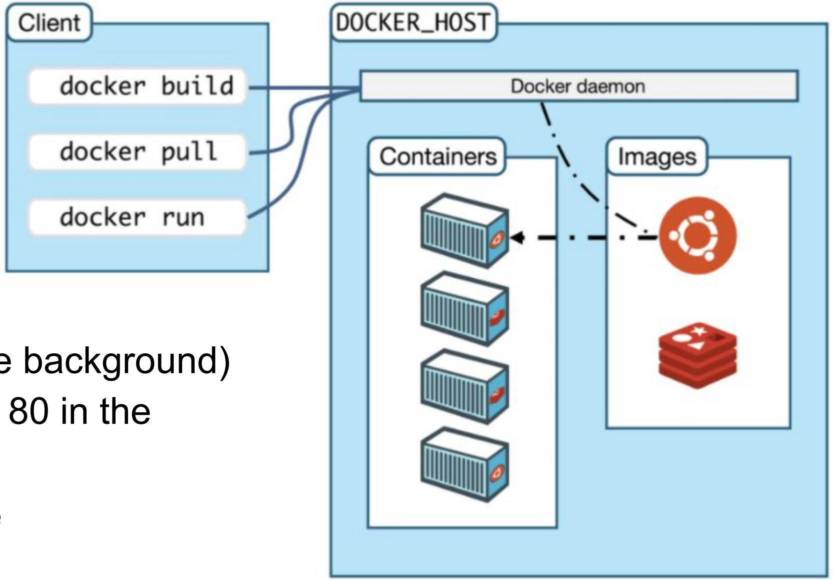
3. Check Docker version with this command :

```
~$ docker --version
```

4. Run Docker container with this command :

```
~$ docker run -dp 8080:80 docker/getting-started
```

- -d - run the container in detached mode (in the background)
- -p 8080:80 - map port 8080 of the host to port 80 in the container
- dockersamples/101-tutorial - the image to use



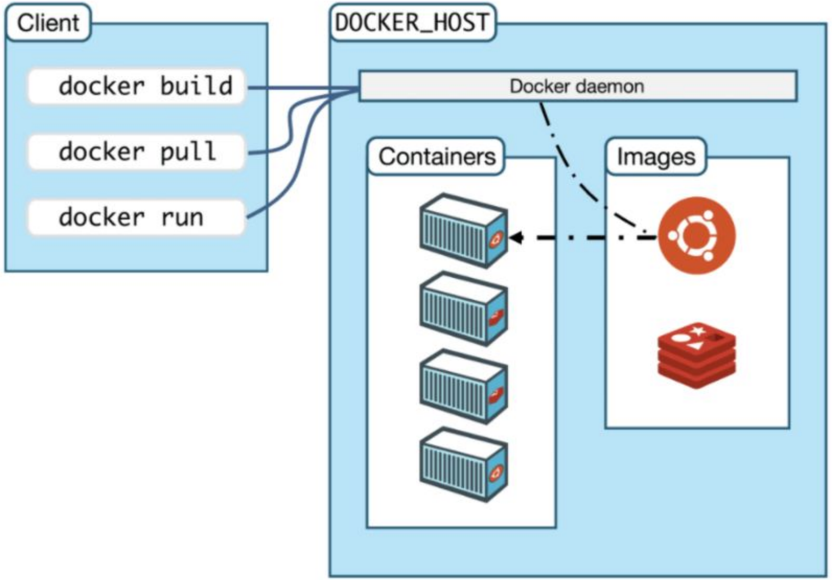
Lab : Docker play with Docker

5. Check Docker container with this command :

```
~$ docker ps
```

6. Check Docker images with this command :

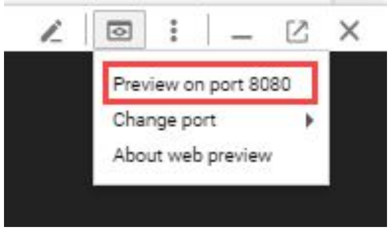
```
~$ docker images
```



Lab : Play with Docker

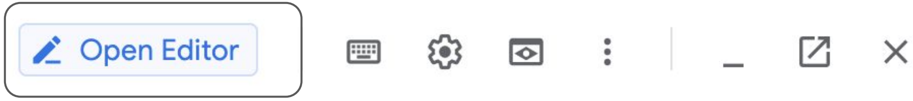
Clone Source Repository

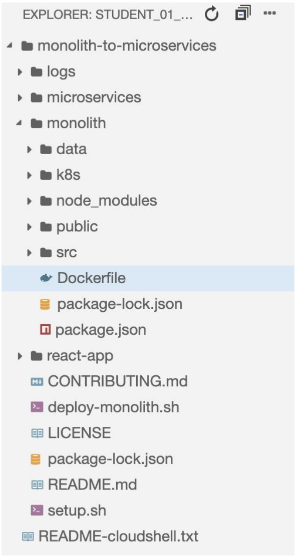
```
1.    git clone https://github.com/googlecodelabs/monolith-to-microservices.git
2.    cd ~/monolith-to-microservices
3.    ./setup.sh
4.    cd ~/monolith-to-microservices/monolith
5.    npm start
```

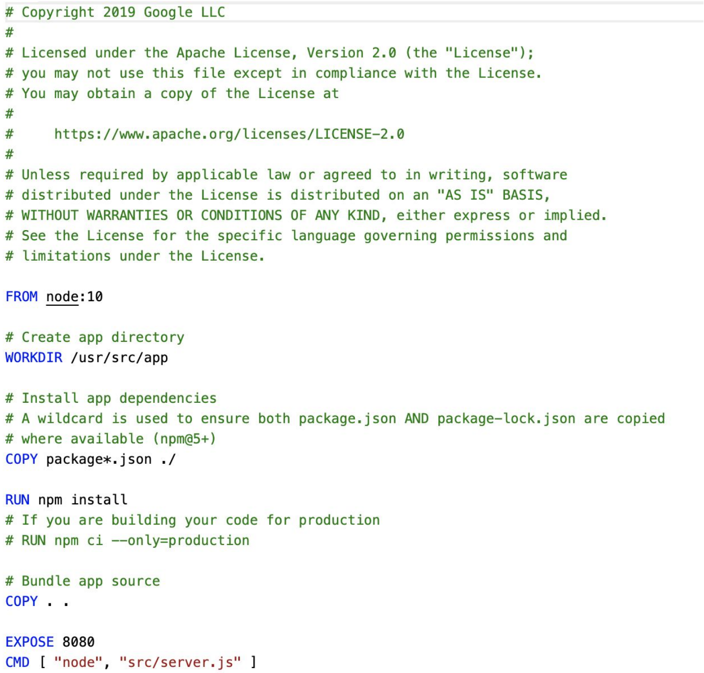


Lab : Dockerfile

1. Open Editor
2. Browse to Dockerfile





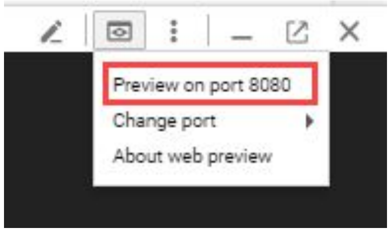


```
# Copyright 2019 Google LLC
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     https://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

FROM node:10

# Create app directory
WORKDIR /usr/src/app

# Install app dependencies
# A wildcard is used to ensure both package.json AND package-lock.json are copied
# where available (npm@5+)
COPY package*.json ./

RUN npm install
# If you are building your code for production
# RUN npm ci --only=production

# Bundle app source
COPY . .

EXPOSE 8080
CMD [ "node", "src/server.js" ]
```

Lab : Docker Build and Docker Run

```
1.    docker ps
2.    docker images
3.    docker build -t monolith:v1 .
4.    docker ps
5.    docker images
6.    docker run -d -it  --name monolith -p 8080:8080 monolith:v1
7.    docker ps
8.    docker images
9.    docker logs [container id]
```

10. Goto Preview on port 8080 then capture screen for submit quiz



Google Cloud

Cloud Ace

Break

5 mins
https://www.youtube.com/watch?v=iDnkegx5EBg

03

Running containers on App Engine

Google Cloud

What is App Engine?

An app-centric view of the world

- You want to **focus on writing code** and never touch a server, cluster, or infrastructure
- Building **quickly** and **time to market** are highly valued
- You want to sleep at night and **not worry about a pager going off,** or **5xx errors**
- You **expect your app to have high availability** without a complex architecture
- You can perform **no downtime upgrades** and **A/B test** via split traffic feature

Note: This presentation will focus solely on the App Engine flexible environment option, as it is the recommended approach

App Engine

A flexible, **zero ops** platform for building highly available apps

Google Cloud

What is App Engine flexible environment?

Based on Google Compute Engine, the App Engine flexible environment automatically scales your app up and down while balancing the load. Microservices, authorization, SQL and NoSQL databases, traffic splitting, logging, versioning, security scanning, and content delivery networks are all supported natively. In addition, the App Engine flexible environment allows you to customize the runtime and even the operating system of your virtual machine using Dockerfiles.

It mixes the best of App Engine–managed platform with the flexibility of containers, enabling mix-and-match different versions for various programming languages.

What workloads are ideal?

App Engine's benefits make it ideally suited for building

Mobile backends, especially social and casual games

Software as a Service (SaaS) applications that can disrupt stagnant industries

Internal IT apps that improve productivity and revenue (think Googelplex)

Internet of Things (IoT) front end and back end workloads.

Any web front end (Are you running Tomcat or nginx? Stop.)

Google Cloud

What if I might have to run Swift, Perl, or .NET Core?

“We don’t need to replatform!

- App Engine flexible environment offers the flexibility we need
- We can add other Cloud Platform products to our architecture”

Total control and power

- Use any language, framework, or library, even SWIFT, **Perl, .NET Core Runtime** via custom runtime on App Engine flexible environment
- Can write to local disks
- SSH into VM for diagnostics
- Configure lower-level infrastructure services as necessary: Cloud Load Balancer, Cloud Autoscaler, IAM

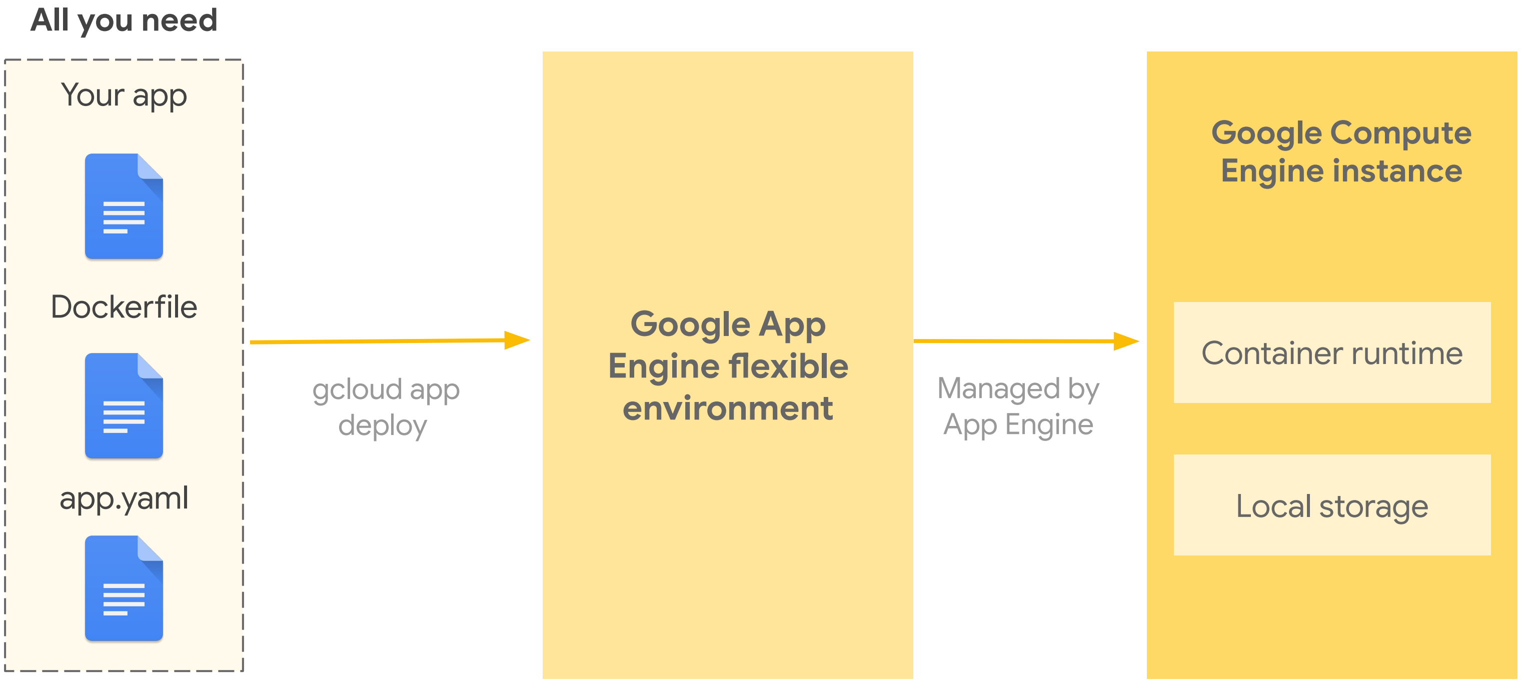
One network

- Single, secure network
- VPN, direct connect, carrier peering options

Big data

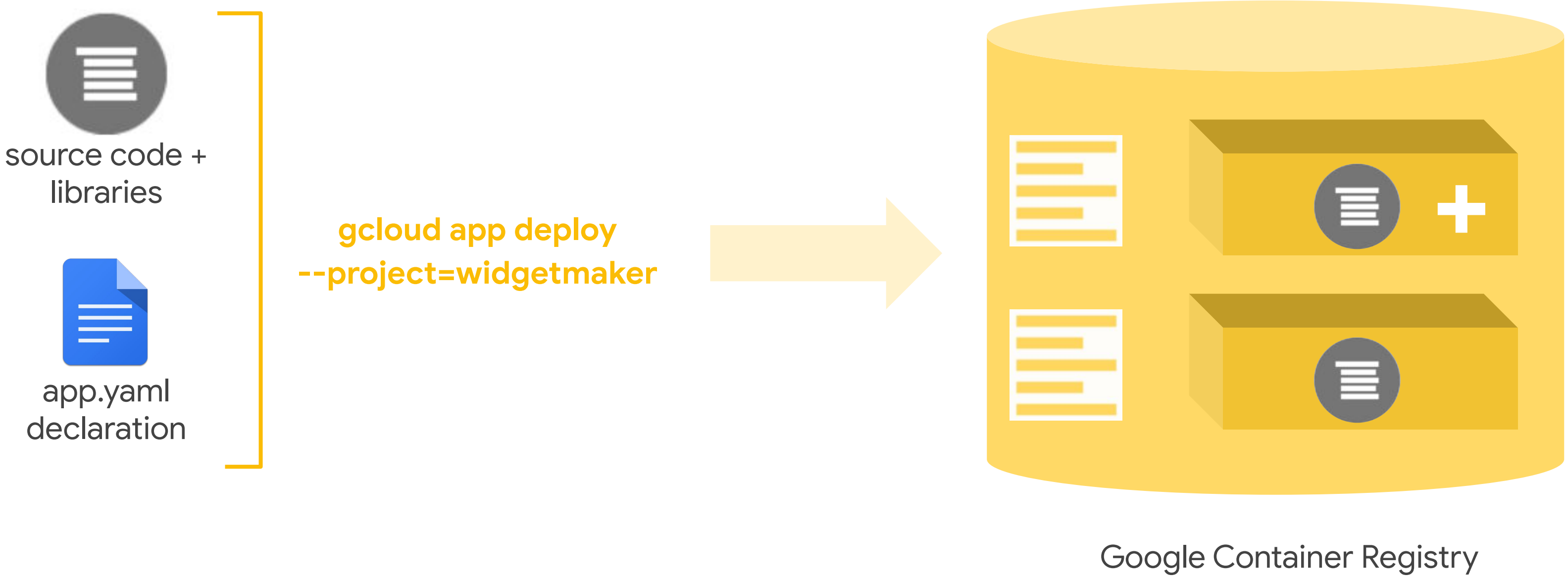
- Pipeline your logs and events to Google's leading data services

Containers in App Engine flexible environment



Google Cloud

Building containers in App Engine flexible environment



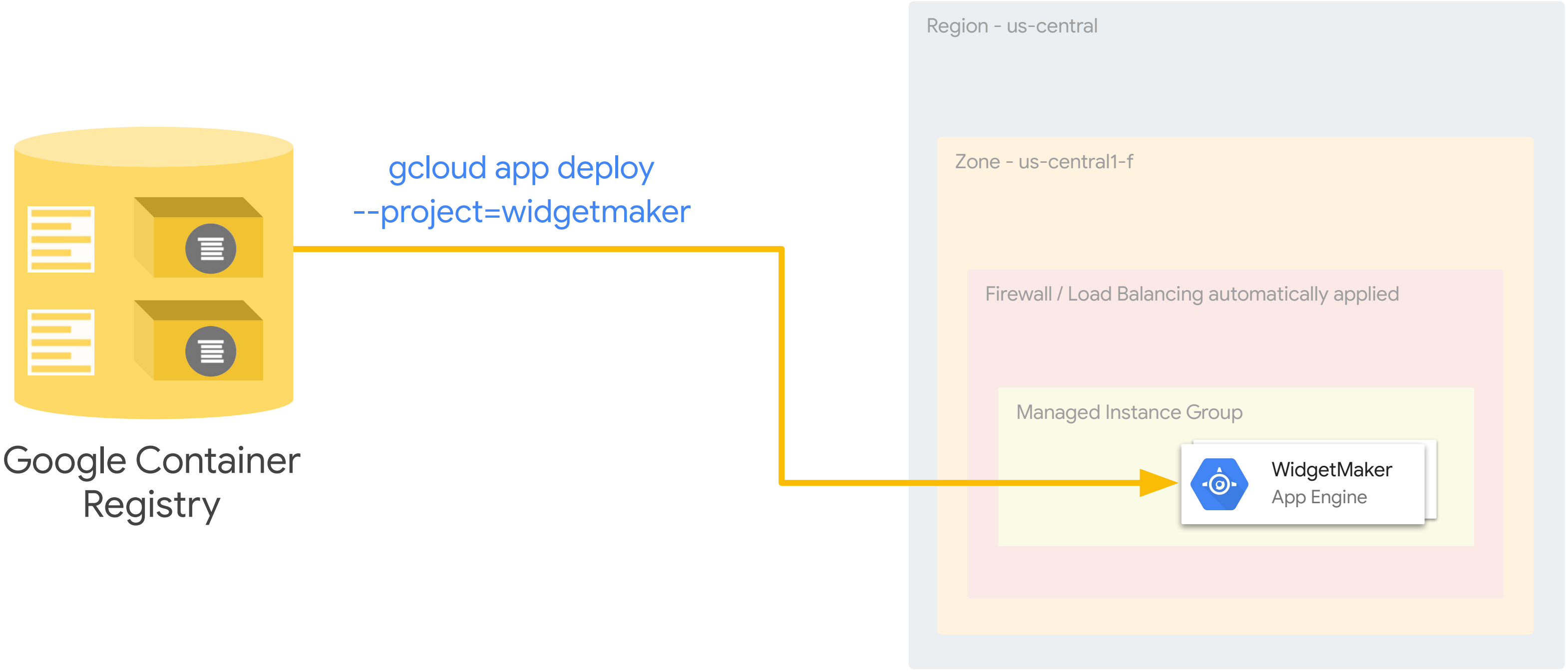
source code + libraries

app.yaml declaration

gcloud app deploy
--project=widgetmaker

Google Container Registry

Using Containers in App Engine flexible environment



Google Cloud

Key decisions

1 How much control do I need over my container environment?

2 Do I have complex security, networking, or management requirements?

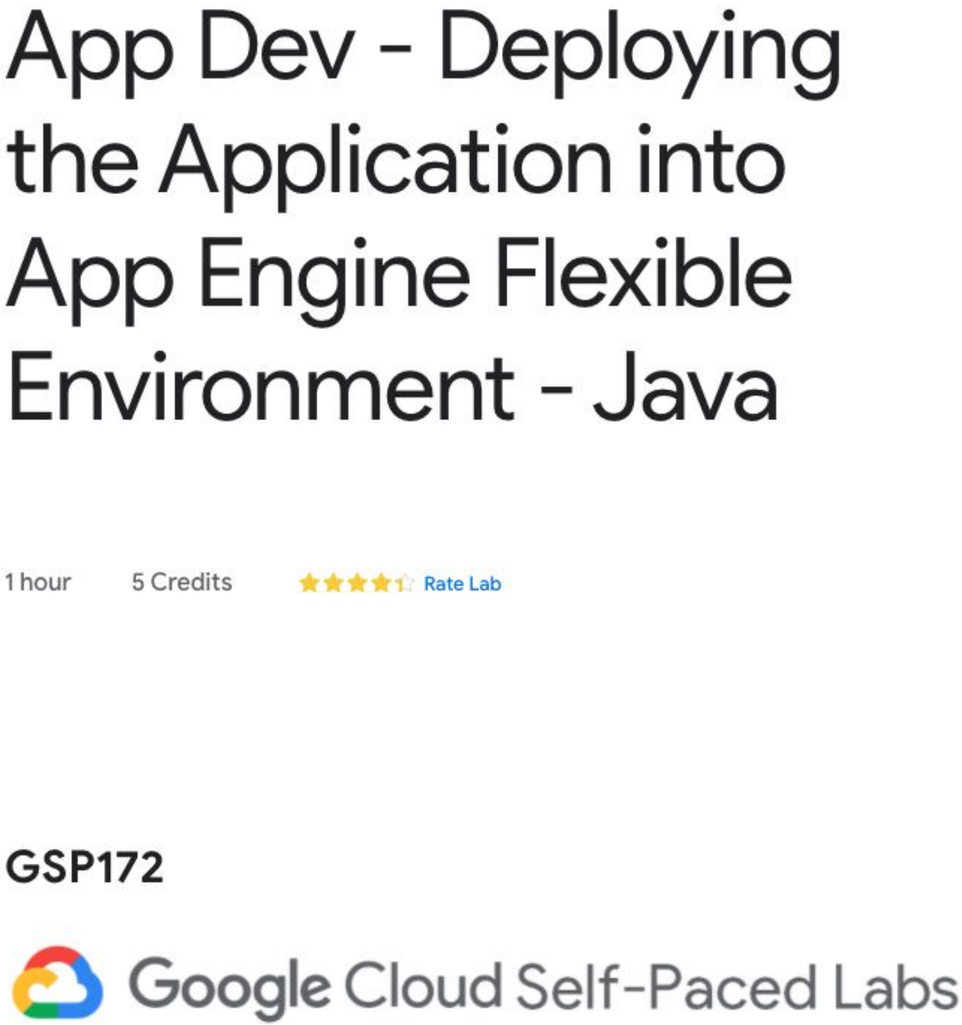
3 Is the language and runtime for my application supported in App Engine flexible environment? Do I want to maintain a custom runtime?

4 Do I need to control how containers are deployed in relationship to either nodes or other containers?

5 Do I need more fine-grained security or network management?

GSP172

Google Cloud



App Dev - Deploying the Application into App Engine Flexible Environment - Java

1 hour 5 Credits ★★★★½ Rate Lab

GSP172

Google Cloud Self-Paced Labs

Cloud Run

Serverless but with Containers

Cloud Run

Container to production in seconds

Natively Serverless

One experience, where you want it

ONE

Container to production in seconds

Containers

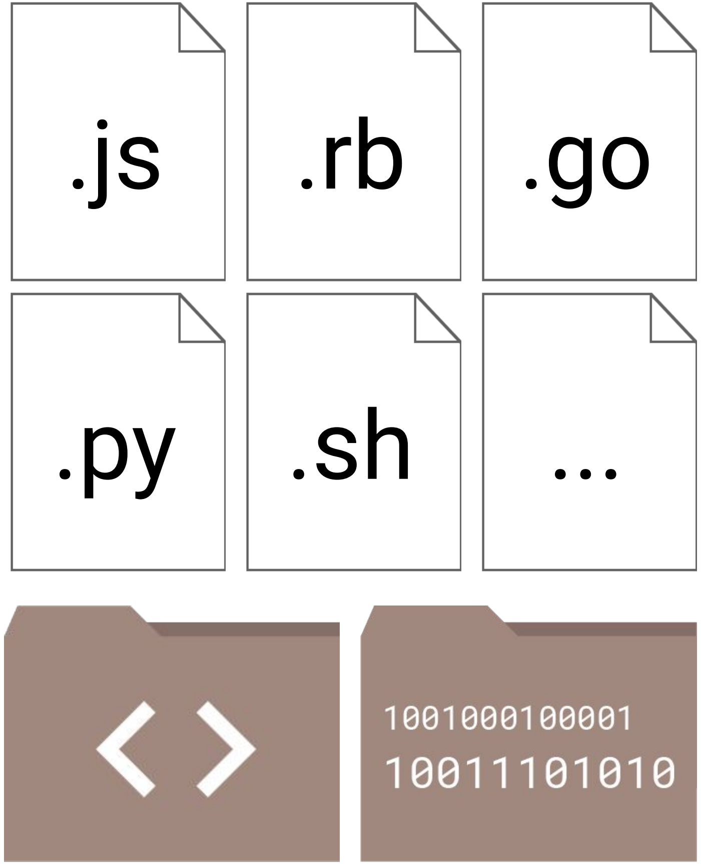
Any language

Any library

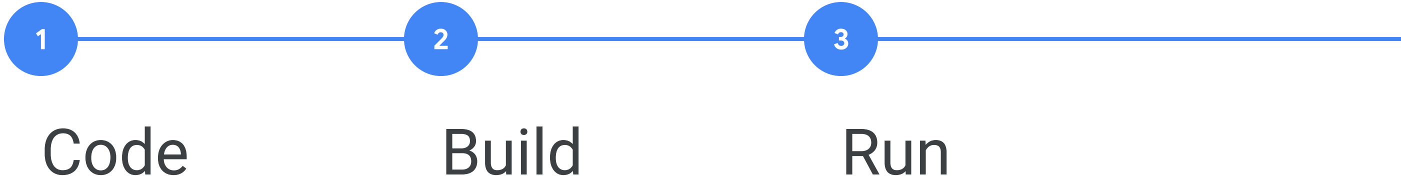
Any binary

Ecosystem of base images

Industry standard



Steps



1 Code

2 Build

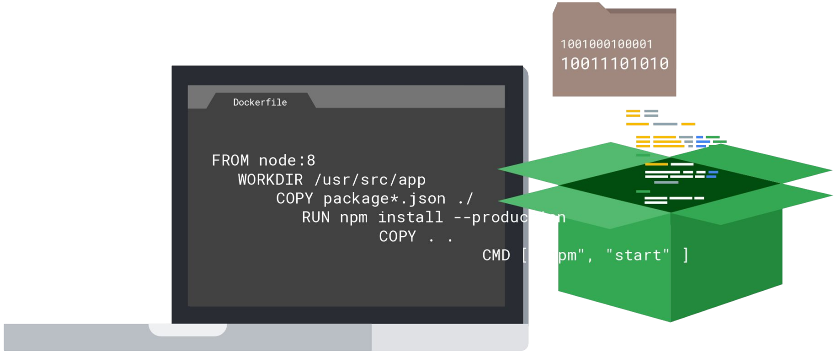
3 Run

Container runtime contract

HTTP

~~State~~

Build



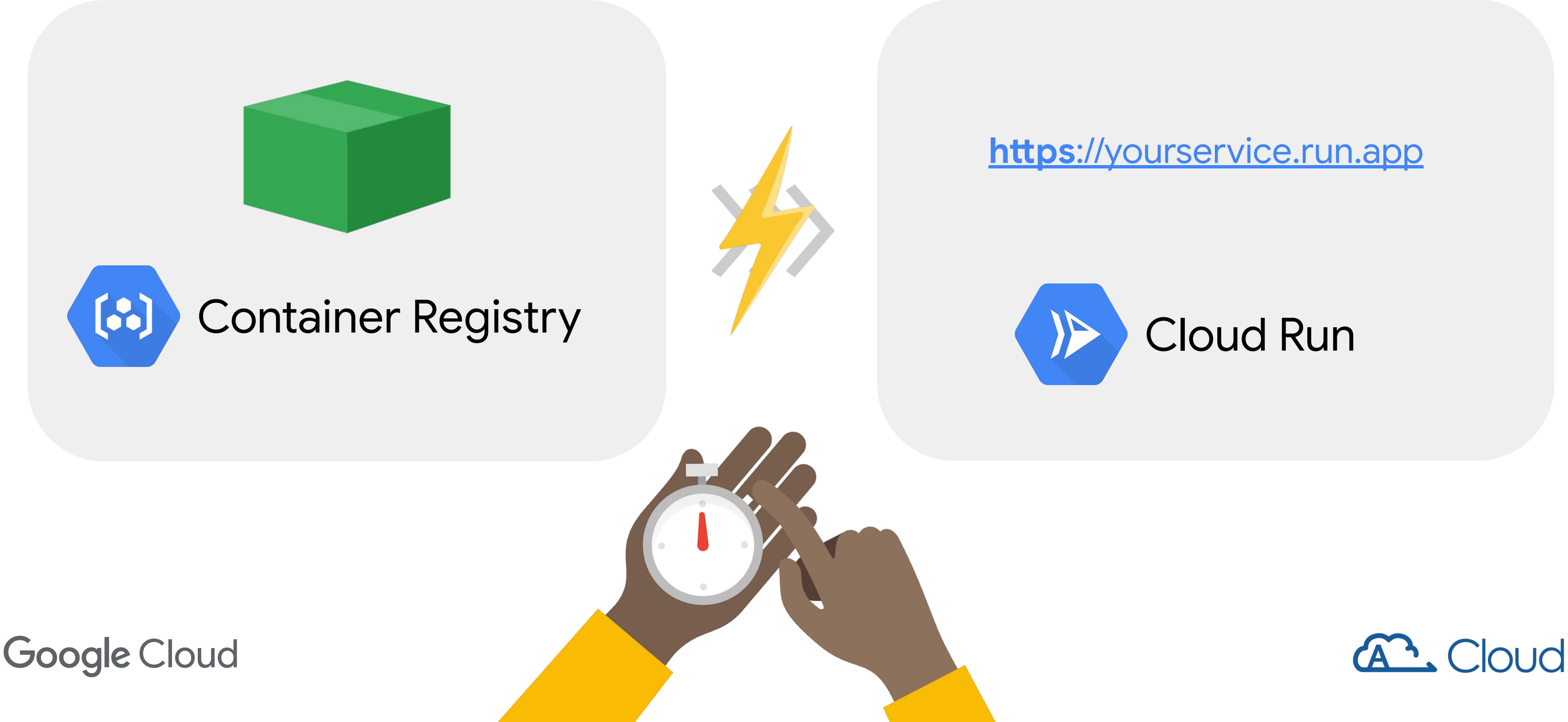
```
FROM node:8
   WORKDIR /usr/src/app
       COPY package*.json ./
          RUN npm install --production
                  COPY . .
                             CMD [ "npm", "start" ]
```

Docker

Cloud Build

CI/CD

Run



Container Registry

https://yourservice.run.app

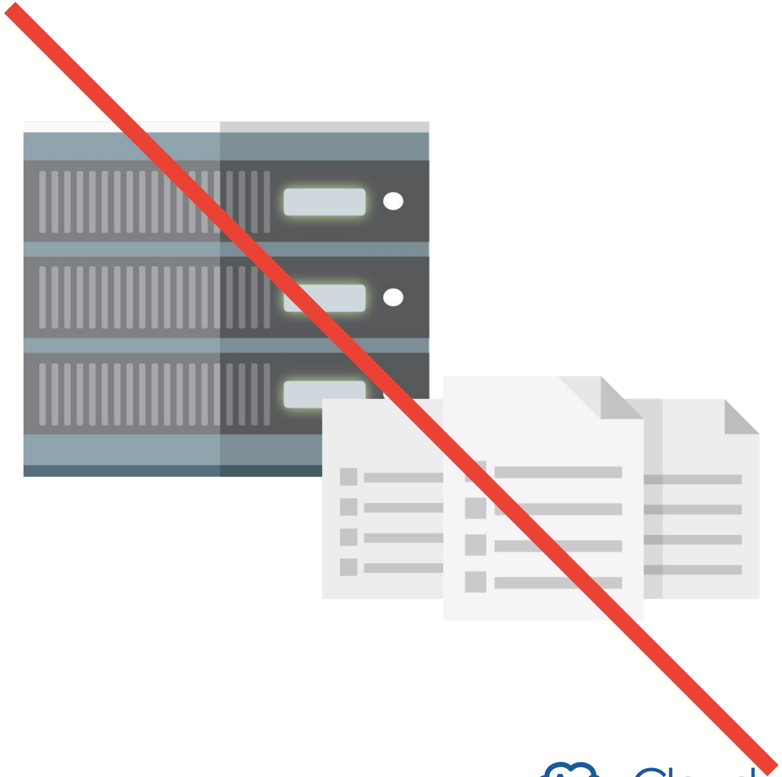
Cloud Run

TWO

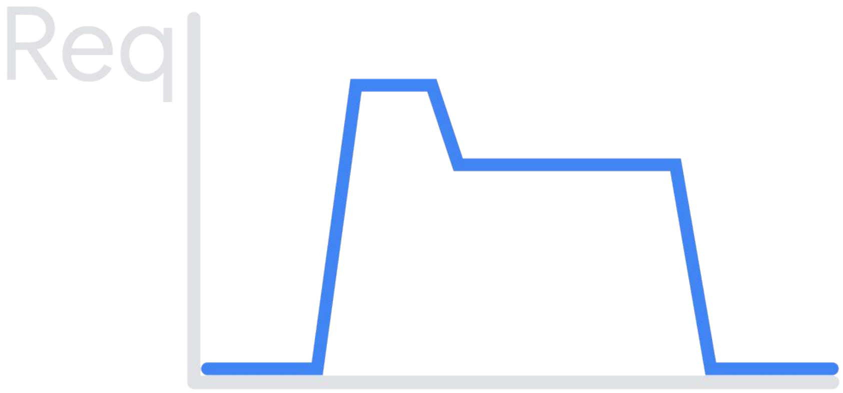
Natively Serverless

Cloud Run is Serverless

- Focus on your code
- No infrastructure to manage
- Managed URLs and TLS certificates
- Redundant, automatic failover
- Simple developer experience
- Scales with you
- Pay per use



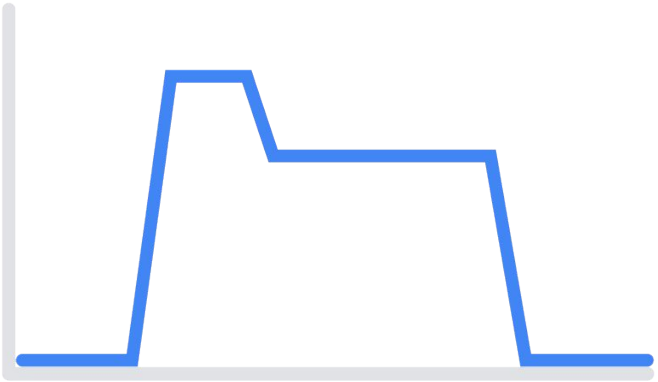
Automatically scaled for requests



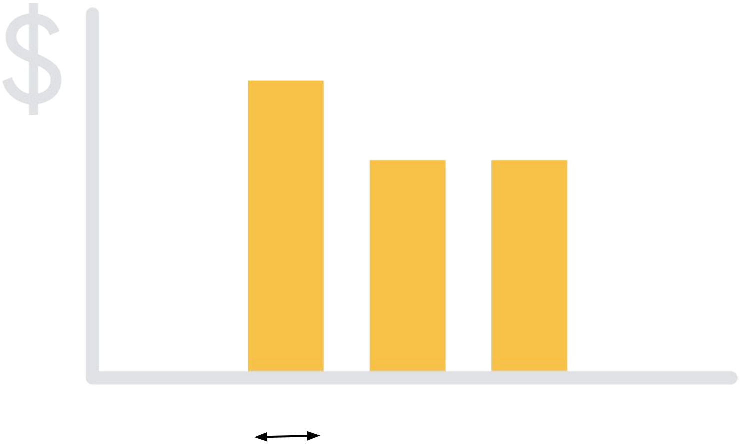
Scales up fast

Scales down to zero

Cloud Run: Pay-per-use



CPU / Memory / Requests



100ms

THREE

One experience, where you want it

Introducing Cloud Run on GKE

Same great Cloud Run, but on Kubernetes

More flexibility and control, operator required.

Integrates with k8s-based policy, control & mgmt

Custom nodes, hardware accelerators, VPC

Build on your existing investment in Kubernetes

Serverless containers, where you want them

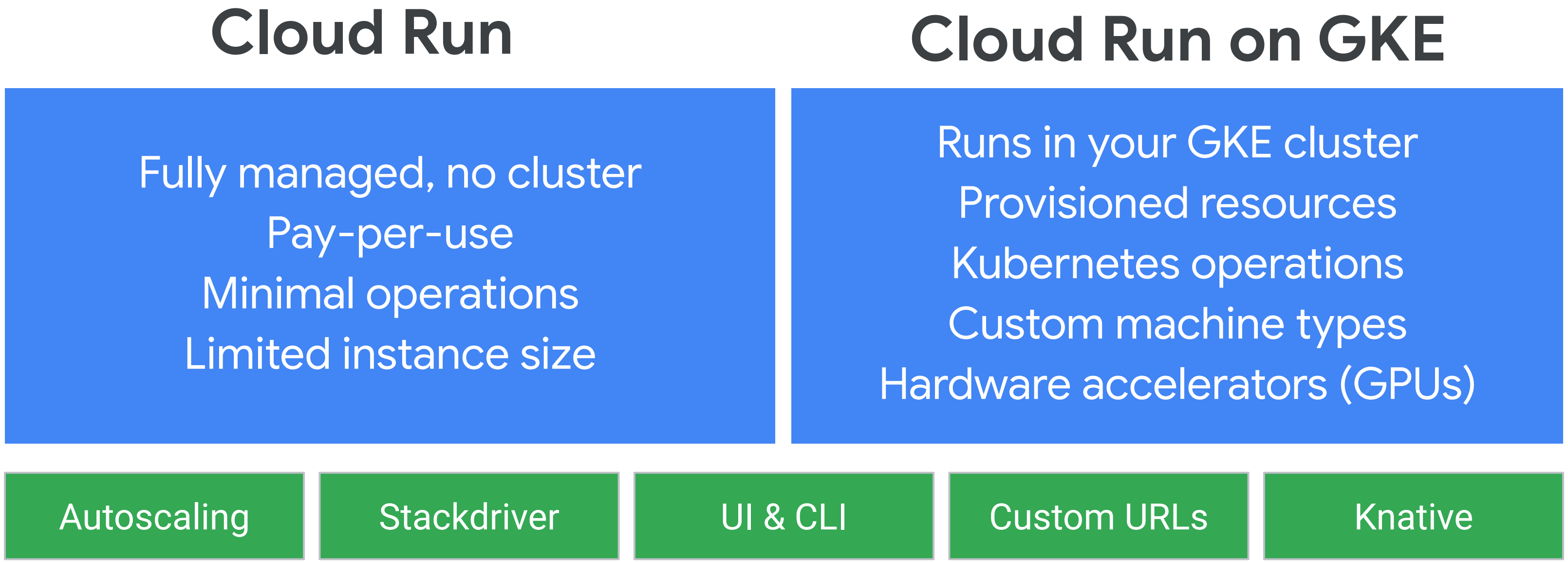
Cloud Run

Fully serverless
No cluster to manage
Pay for what you use

Cloud Run on GKE

Serverless developer experience
Runs in your GKE cluster

Choose the platform for you



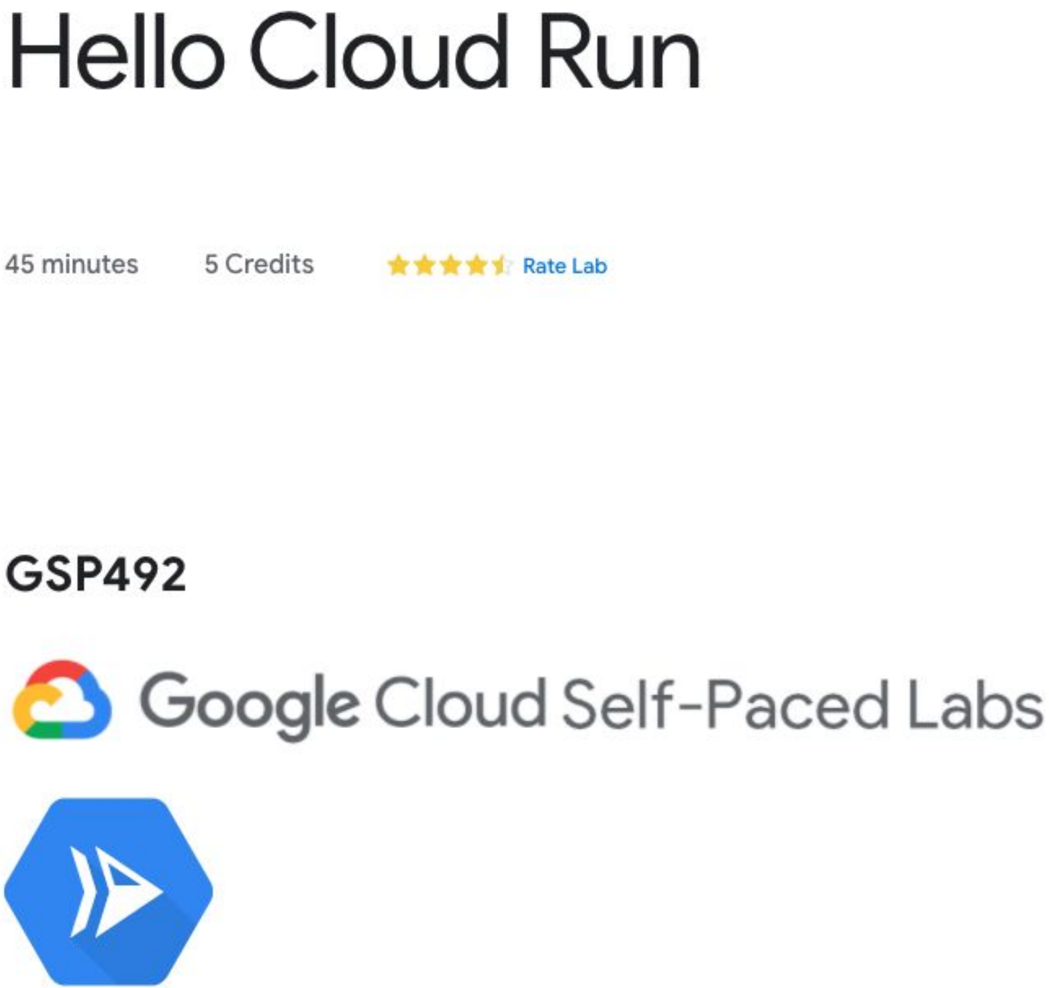
Cloud Run

Fully managed, no cluster
Pay-per-use
Minimal operations
Limited instance size

Cloud Run on GKE

Runs in your GKE cluster
Provisioned resources
Kubernetes operations
Custom machine types
Hardware accelerators (GPUs)

Autoscaling | Stackdriver | UI & CLI | Custom URLs | Knative



Hello Cloud Run

45 minutes 5 Credits ★★★★☆ Rate Lab

GSP492

Google Cloud Self-Paced Labs

Break

5 mins

https://www.youtube.com/watch?v=I0vvfEwI2Og

Running containers on Kubernetes Engine

04

Using orchestration

Container orchestration tools provide a rich set of features for a container infrastructure

Orchestration tools can manage how multiple containers get created and updated, and provide high availability, networking, fault tolerance, and more

Orchestration tools can take you a long way but there are still some bridges to cross with regard to integration of the VM and the container environment

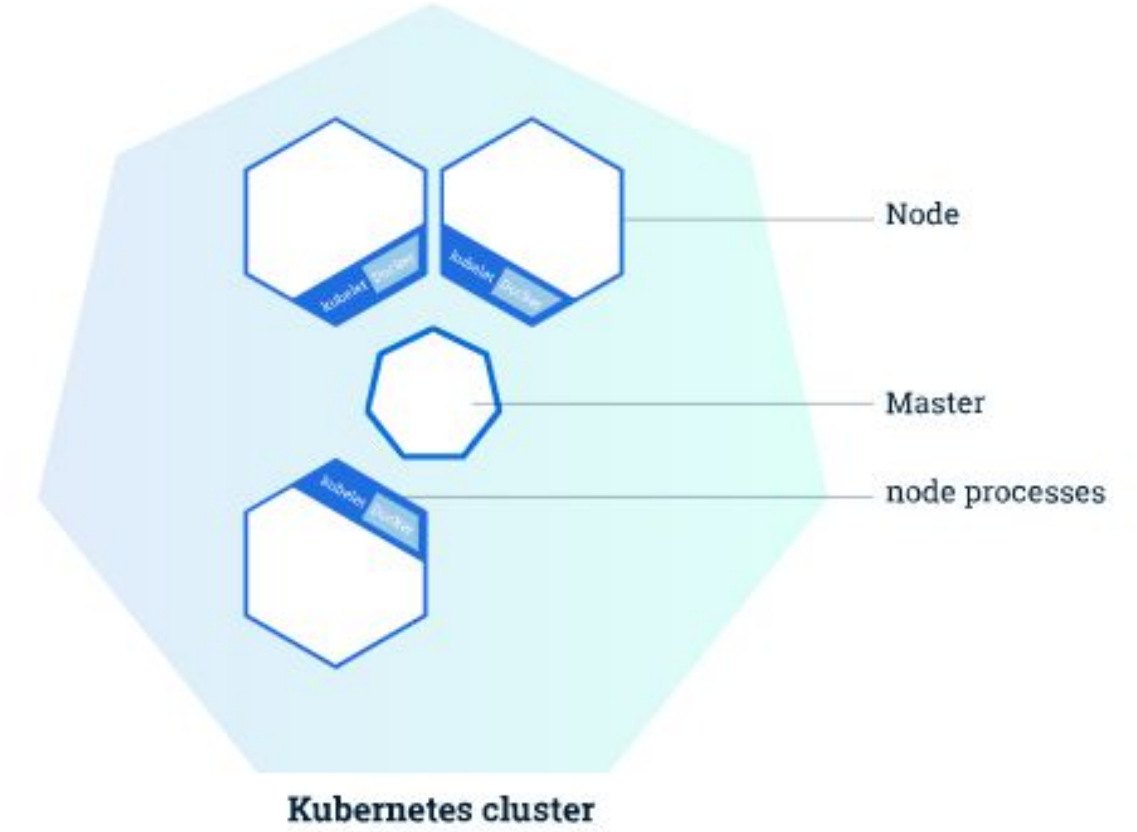
kubernetes

docker SWARM

HashiCorp Nomad

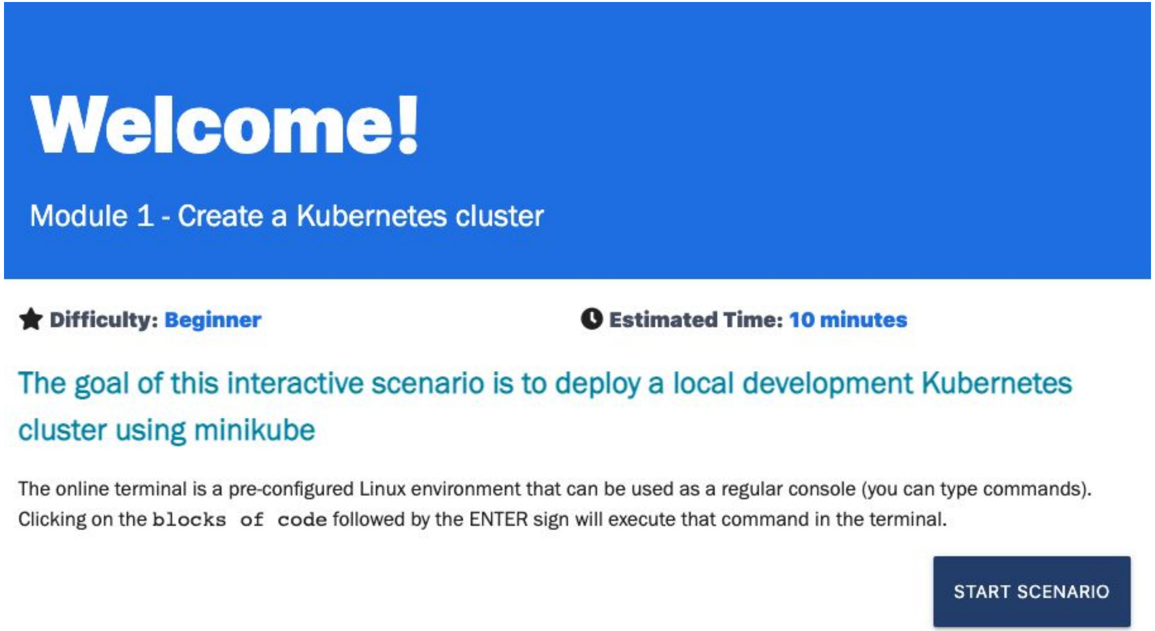
Apache MESOS™

Kubernetes Cluster



Kubernetes Cluster Lab

https://kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-interactive/



Welcome!

Module 1 - Create a Kubernetes cluster

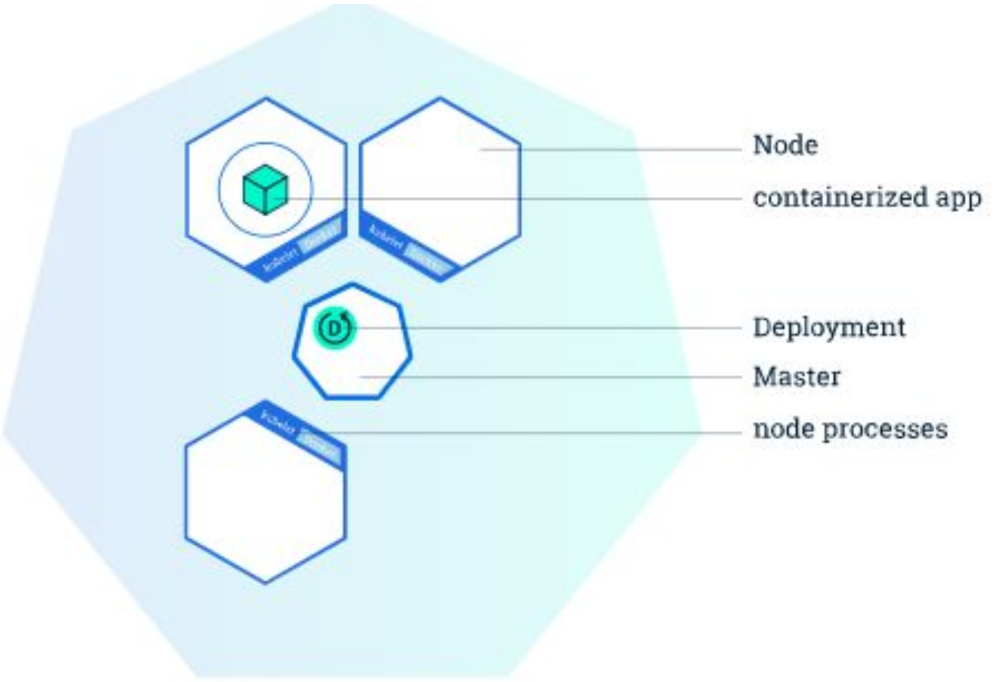
★ **Difficulty: Beginner**

Estimated Time: 10 minutes

The goal of this interactive scenario is to deploy a local development Kubernetes cluster using minikube

The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the `blocks of code` followed by the ENTER sign will execute that command in the terminal.

Deploy app to Kubernetes



Node

containerized app

Deployment

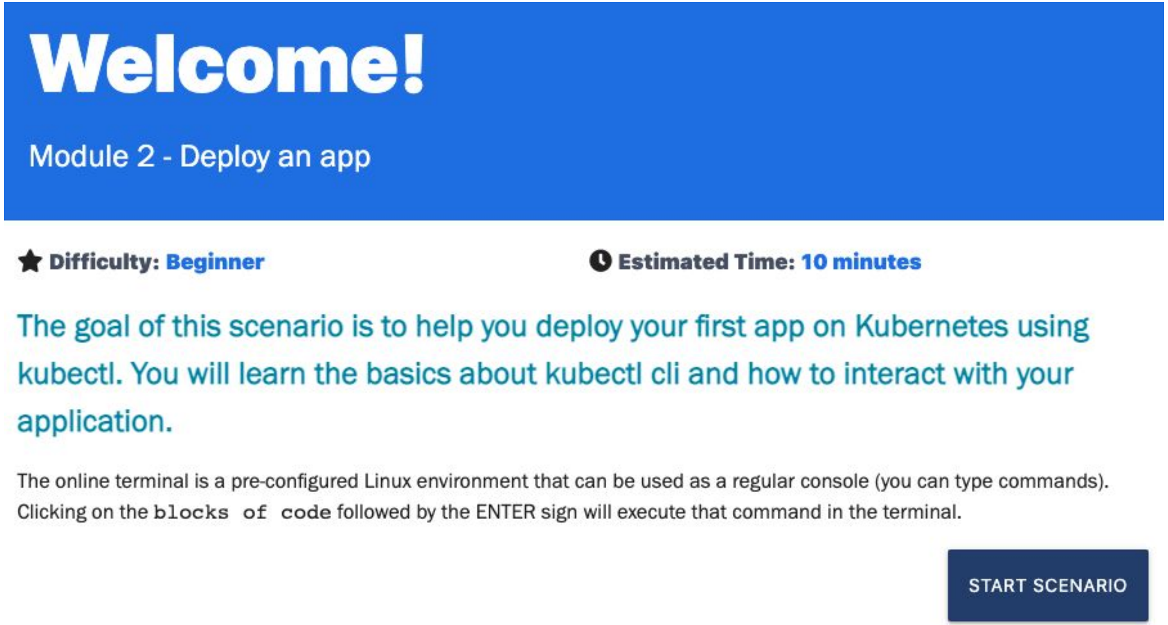
Master

node processes

Kubernetes Cluster

Kubernetes Deployment Lab

https://kubernetes.io/docs/tutorials/kubernetes-basics/deploy-app/deploy-interactive/



Welcome!

Module 2 - Deploy an app

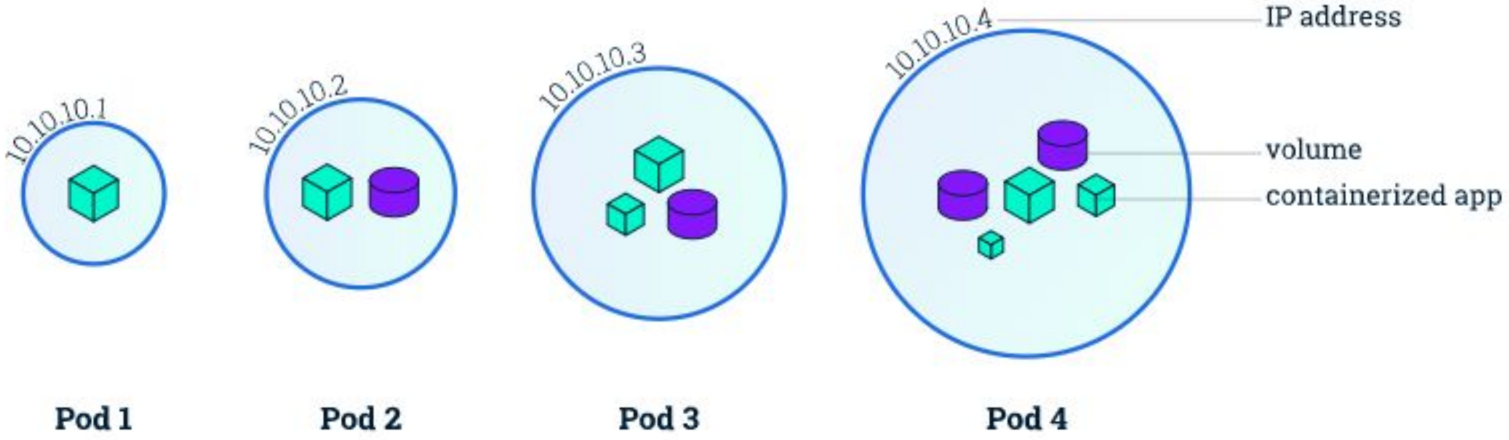
★ Difficulty: Beginner

Estimated Time: 10 minutes

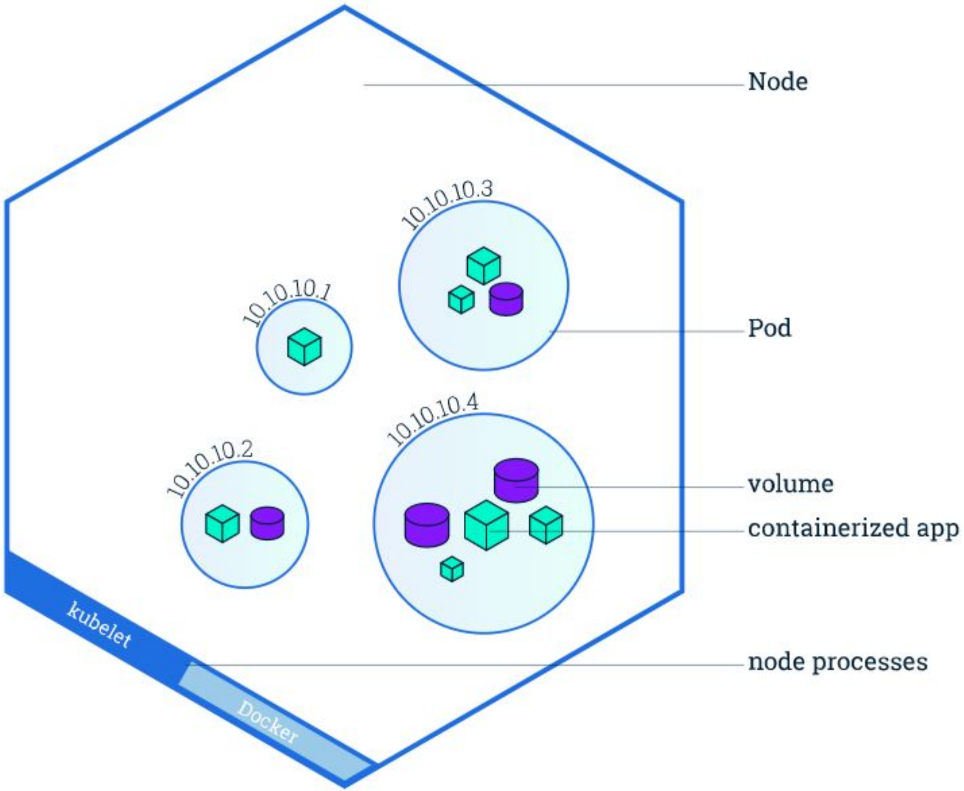
The goal of this scenario is to help you deploy your first app on Kubernetes using kubectl. You will learn the basics about kubectl cli and how to interact with your application.

The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the `blocks of code` followed by the ENTER sign will execute that command in the terminal.

Kubernetes Pods



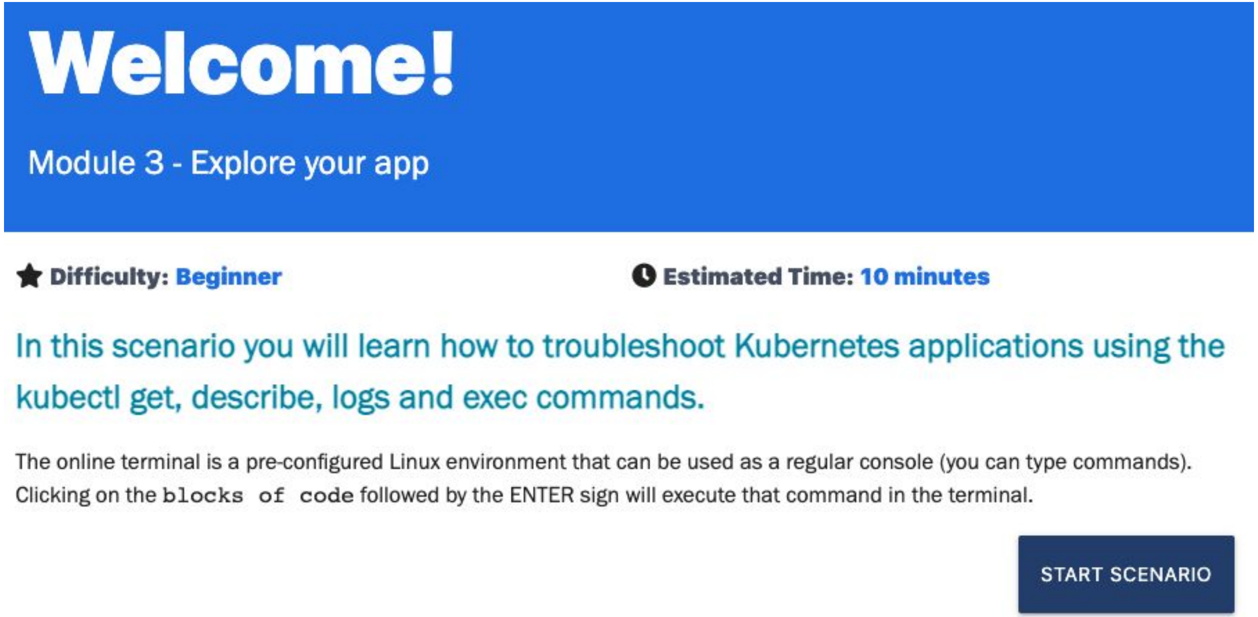
Kubernetes Node



Google Cloud

Explore app Lab

https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-interactive/



Welcome!

Module 3 - Explore your app

★ **Difficulty:** Beginner

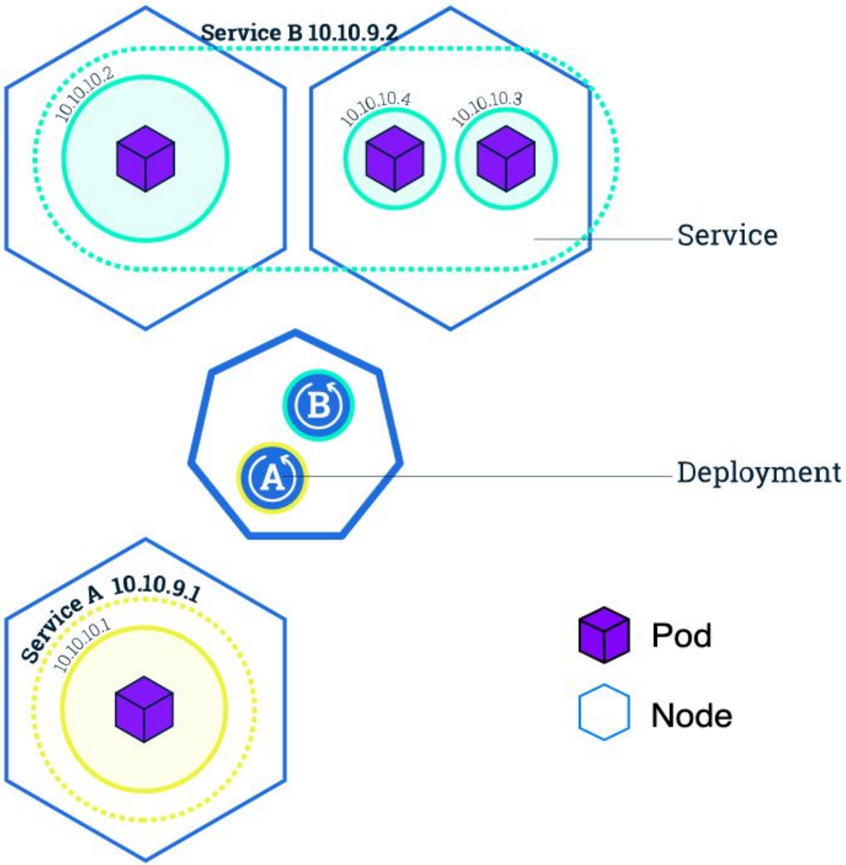
Estimated Time: 10 minutes

In this scenario you will learn how to troubleshoot Kubernetes applications using the kubectl get, describe, logs and exec commands.

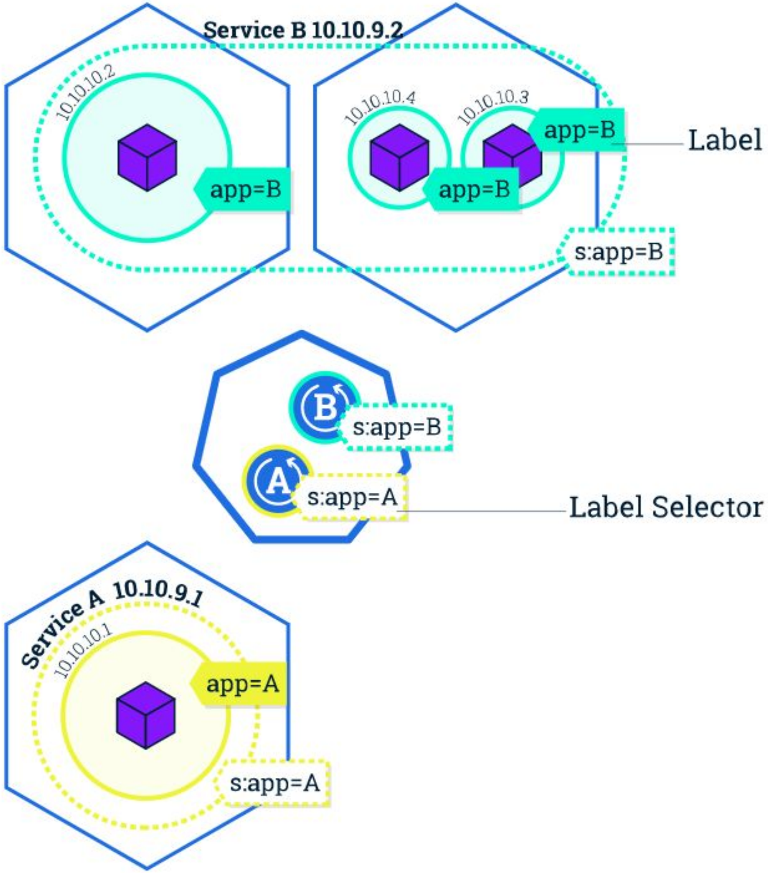
The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the `blocks of code` followed by the ENTER sign will execute that command in the terminal.

Google Cloud

Kubernetes Services



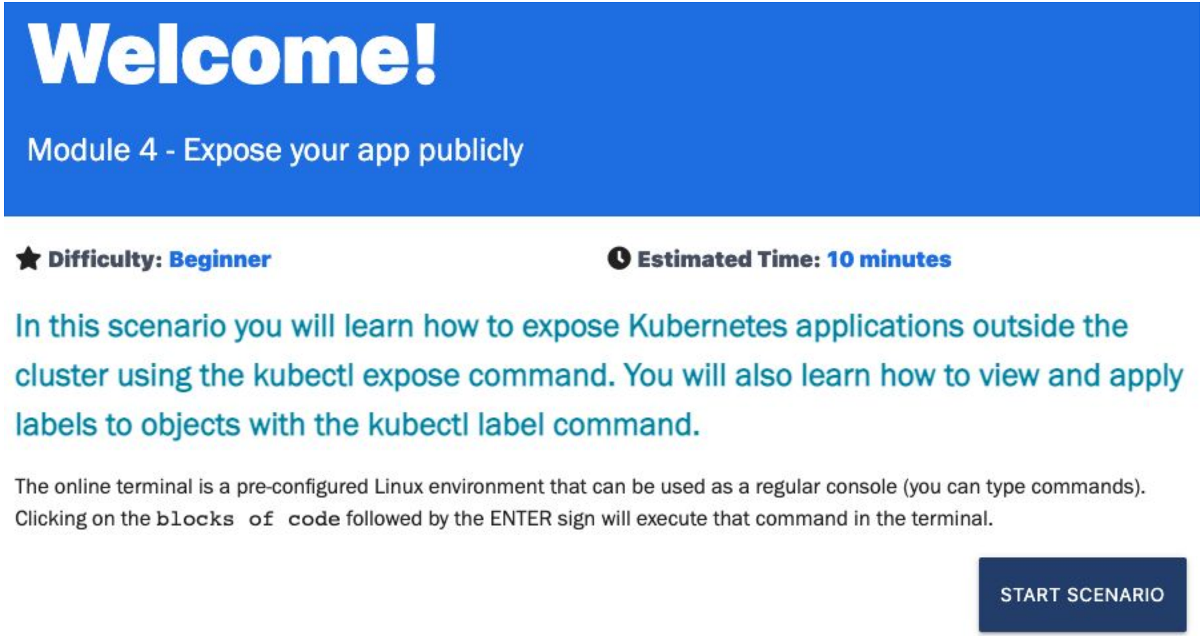
Kubernetes Labels



Google Cloud

Explore app Lab

https://kubernetes.io/docs/tutorials/kubernetes-basics/expose/expose-interactive/



Welcome!

Module 4 - Expose your app publicly

★ **Difficulty: Beginner**

Estimated Time: 10 minutes

In this scenario you will learn how to expose Kubernetes applications outside the cluster using the kubectl expose command. You will also learn how to view and apply labels to objects with the kubectl label command.

The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the `blocks of code` followed by the ENTER sign will execute that command in the terminal.

What is Kubernetes Engine?

Google Kubernetes Engine is managed Kubernetes hosted on Google Cloud.

It’s built upon open-source Kubernetes.

Backed by Google Cloud infrastructure, it’s secure, reliable, and scalable to handle massive workloads quickly and efficiently.

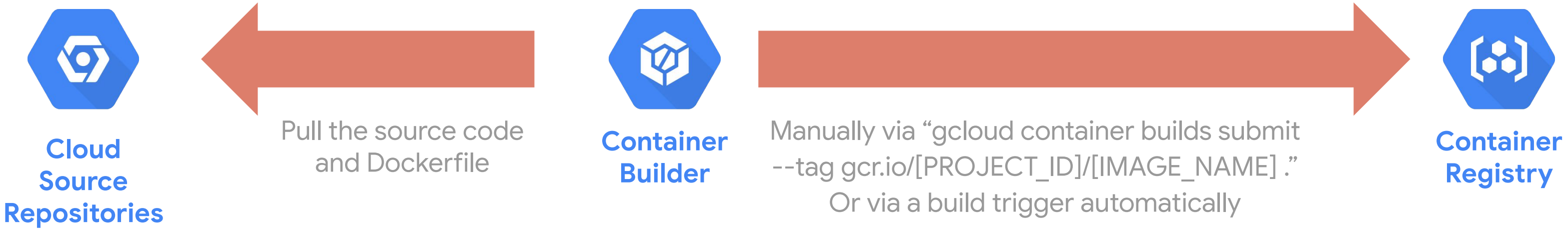
Kubernetes Engine manages time-consuming operational tasks for you such as:

- Implementing and configuring cluster networking
- Provisioning, maintaining, upgrading VMs
- Container logging, monitoring, replication, autoscaling

https://kubernetes.io/docs/tutorials/kubernetes-basics/

Building containers in Kubernetes Engine

While building containers in Kubernetes Engine, consider the following process flow:



Cloud Source Repositories

Pull the source code and Dockerfile

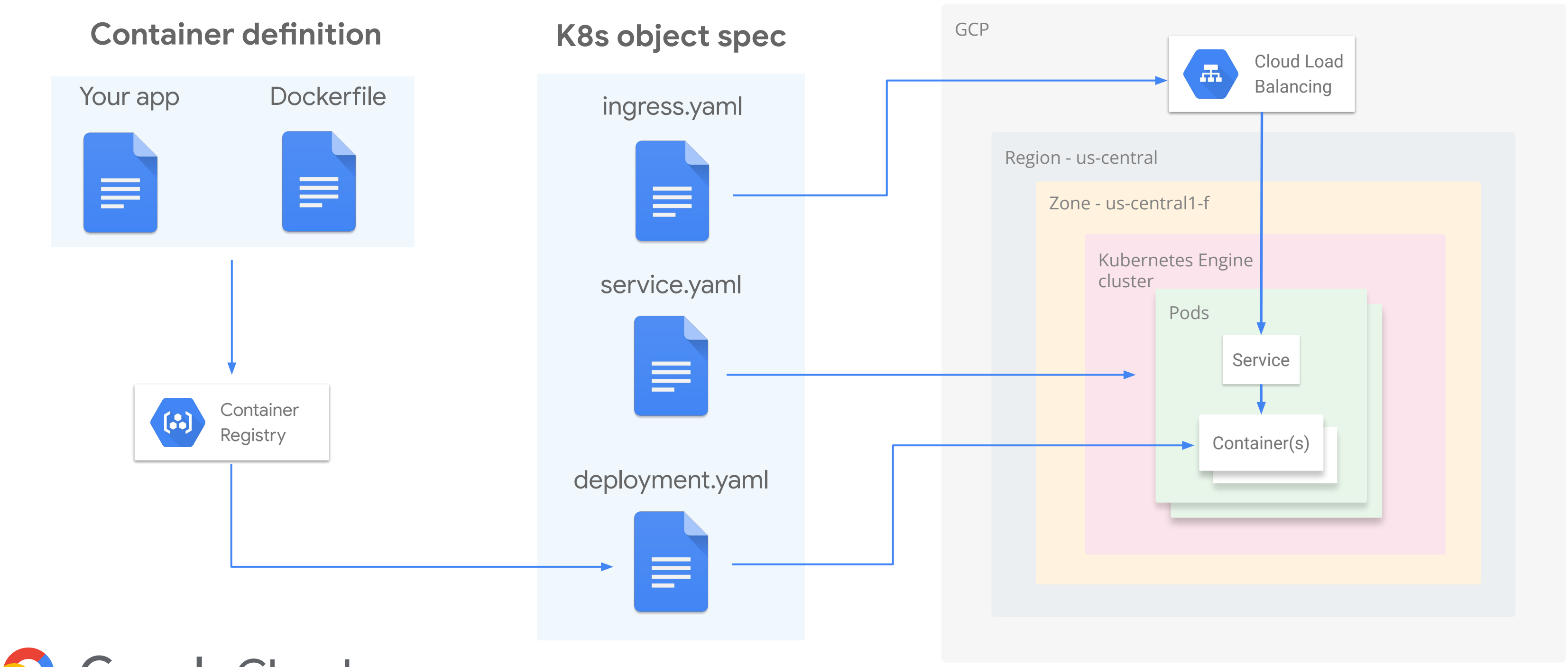
Container Builder

Manually via "gcloud container builds submit --tag gcr.io/[PROJECT_ID]/[IMAGE_NAME] ."
Or via a build trigger automatically

Container Registry

Using containers in Kubernetes Engine

While running containers in Kubernetes Engine, consider the following process flow:



Google Cloud

Kubernetes Engine pros and cons

Pros

- More control over infrastructure and architecture compared to App Engine
- Reduced operational overhead compared to running Kubernetes yourself or containers on VMs
- Workload mobility, no vendor lock-in
- Optimized resource utilization
- Access controls with GCP IAM

Cons

- Opinionated configuration and workflow
- No control over master node and control plane — managed by GCP
- Very few customization options for worker nodes
- Cannot run Windows containers

Key decisions

1 How much control do I need over my containers? Do I need orchestration?

2 How do I expose my service(s)? Load Balancer? Ingress? or NodePort?

3 How do I handle persistent data?

4 What back end should I use for logging and monitoring?

5 How do I secure my cluster? Who needs access to what?

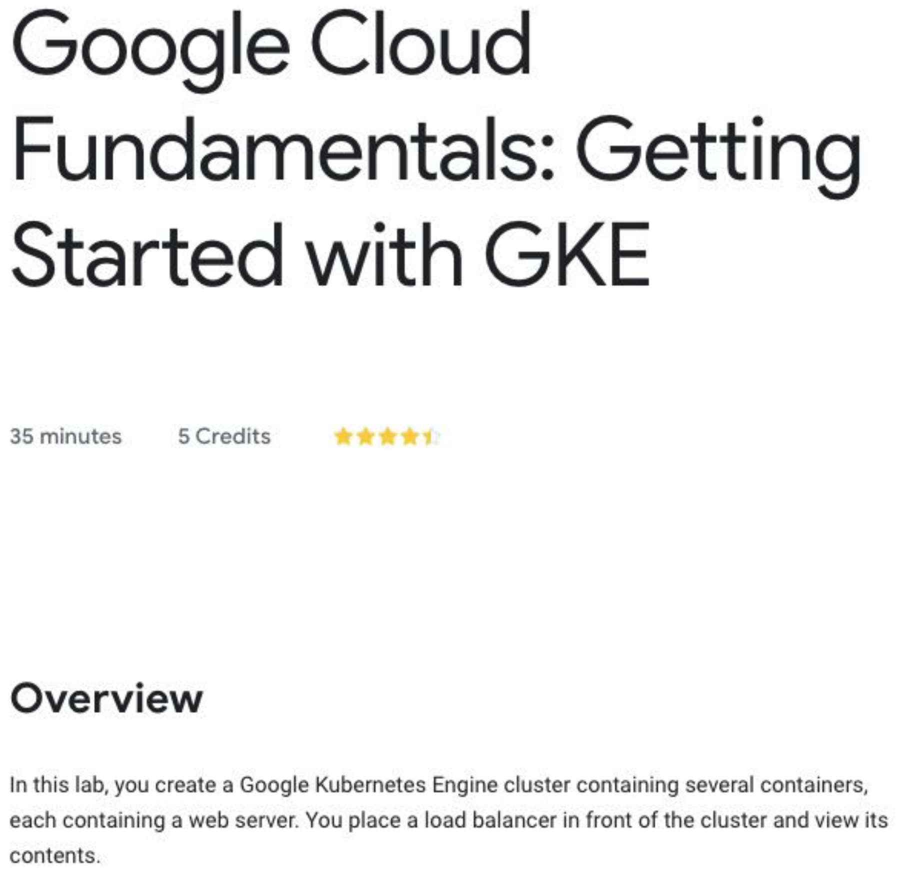
6 What is my CI/CD strategy?

AK8S-03 Creating a GKE Cluster via GCP Console

1 hour 5 Credits ★★★★½

Overview

In this lab, you use the GCP Console to build GKE clusters and deploy a sample Pod.



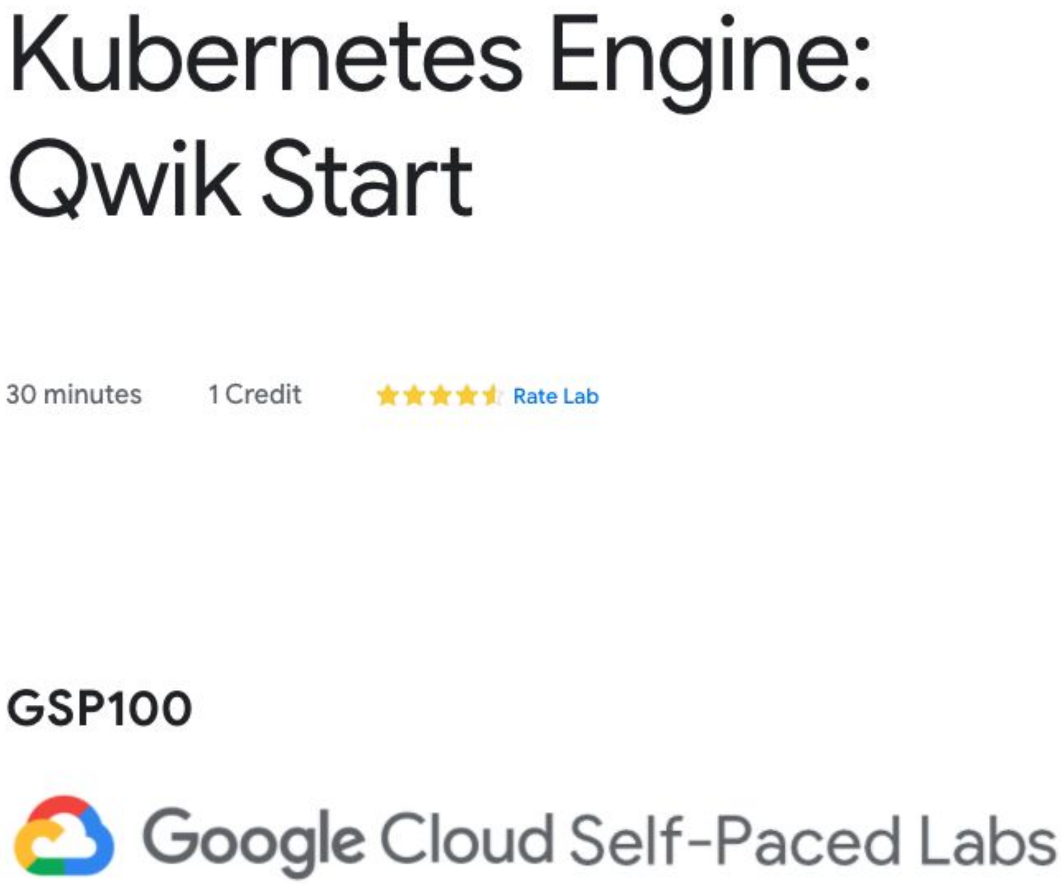
Google Cloud Fundamentals: Getting Started with GKE

35 minutes 5 Credits ★★★★½

Overview

In this lab, you create a Google Kubernetes Engine cluster containing several containers, each containing a web server. You place a load balancer in front of the cluster and view its contents.

ตัวอย่าง yaml file
https://github.com/arun-gupta/docker-kubernetes-hello-world/blob/master/deployment.yaml

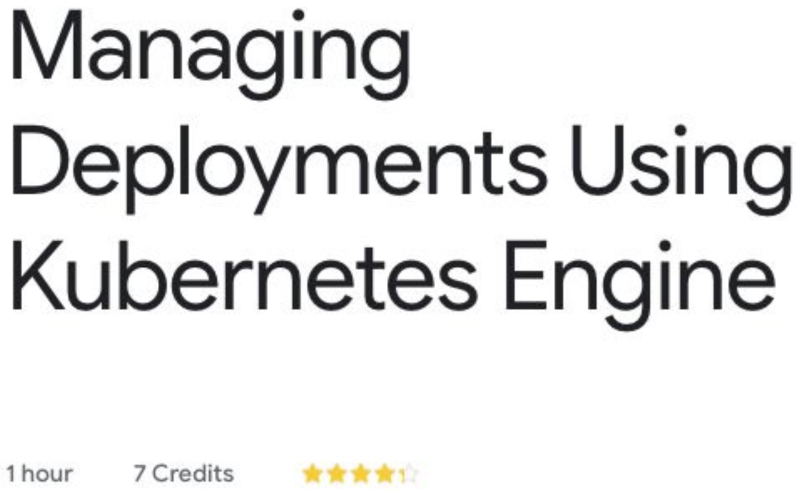


Kubernetes Engine: Qwik Start

30 minutes 1 Credit ★★★★⯪ Rate Lab

GSP100

Google Cloud Self-Paced Labs



Managing Deployments Using Kubernetes Engine

1 hour 7 Credits ★★★★☆

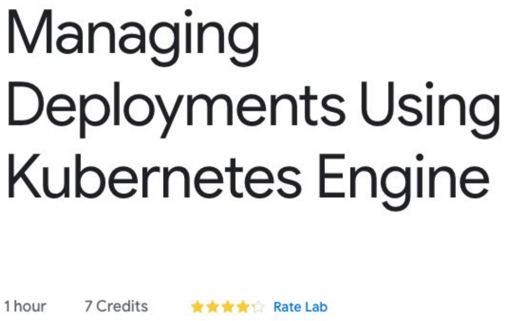
https://www.qwiklabs.com/focuses/639?parent=catalog

GSP053

Google Cloud Self-Paced Labs

Google Cloud

Homework



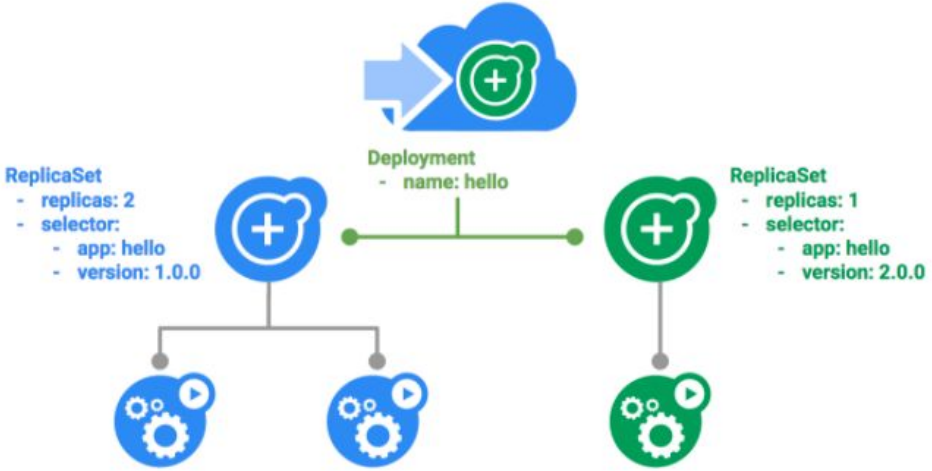
Managing Deployments Using Kubernetes Engine

1 hour 7 Credits ★★★★☆ Rate Lab

GSP053

Google Cloud Self-Paced Labs

Rolling update


ReplicaSet
- replicas: 2
- selector:
- app: hello
- version: 1.0.0
Deployment
- name: hello
ReplicaSet
- replicas: 1
- selector:
- app: hello
- version: 2.0.0