Kubernetes – **Capability Building**

Kubernetes – Learn Sources & Capability Building

Sukhvinder Singh

Kubernetes ➔ DevOps Capability Building

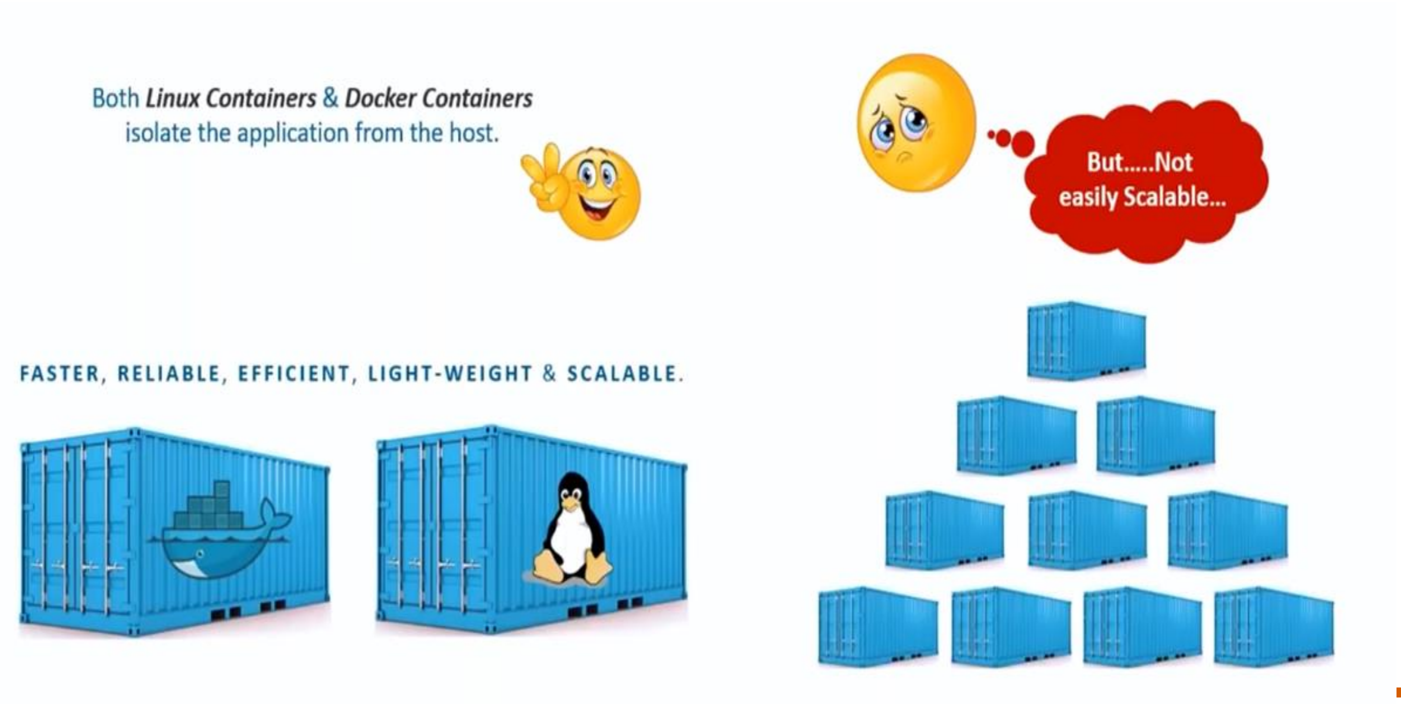
Primary Objective is to build capability building in Kubernetes

- <u>Agenda</u>
  - Containers are Good
  - Containers Problem; Problems with scaling up with containers
  - Need of Kubernetes
  - What exactly kubernetes is & what it is not
  - Kubernetes Vs Docker-Swarm
  - How Does Kubernetes work
  - Architecture of Kubernetes Container management Framework
  - Various Learn Sources, sites , online play-around environment for Kubernetes
    - Kubernetes Tutorial
    - Kubernetes Basics
    - KataCoda: Interactive Based scenarios
    - Kubernetes Playground
    - Minikube cluster
    - Udacity: Start Free course on Kubernetes
  - Hands-On:
    - ✓ Create Cluster in simple Steps
    - ✓ Use Case: Kubernetes @ Pokemon Go
    - ✓ Deployment with kubernetes

Containers ..

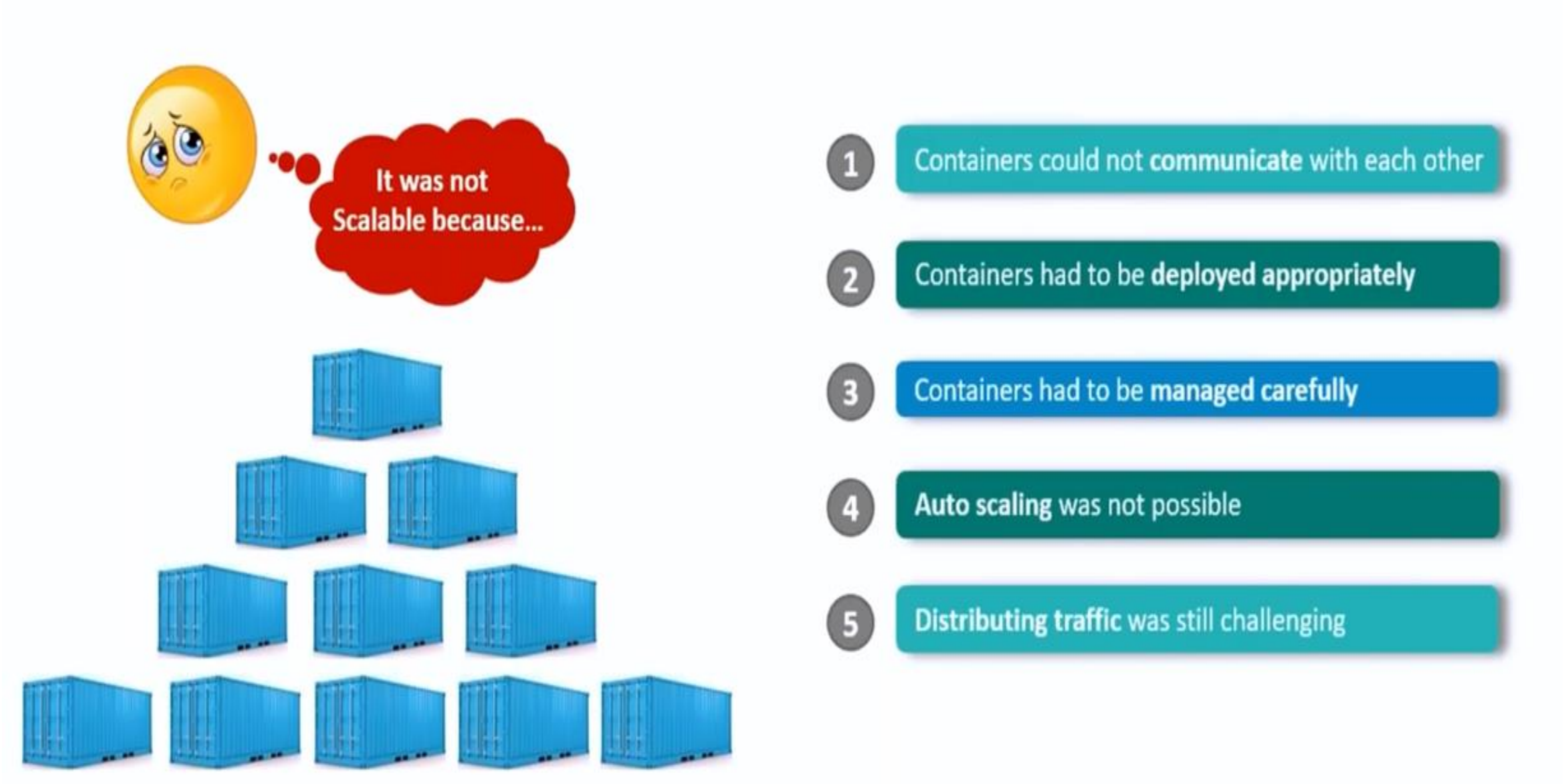
- ❑ Containers are Good..
  - ➢ Take any container, Linux container or a Docker container or even a Rocket container:
    - ✓ They all do one thing they package our application and isolate
    - ✓ Containers are fast, reliable, efficient lightweight and scalable

- ❑ Damn! Container Problems..
  - ➢ There is a problem ..not very easily scalable, scale up to 3 to 10.. 50 Lot of manual efforts
    - ✓ Manage those containers; Make sure all working; talking to each other; if not there is point of scaling
    - ✓ Its really important that containers are manageable when they scaled


Both Linux Containers & Docker Containers isolate the application from the host.
FASTER, RELIABLE, EFFICIENT, LIGHT-WEIGHT & SCALABLE.
But.....Not easily Scalable...

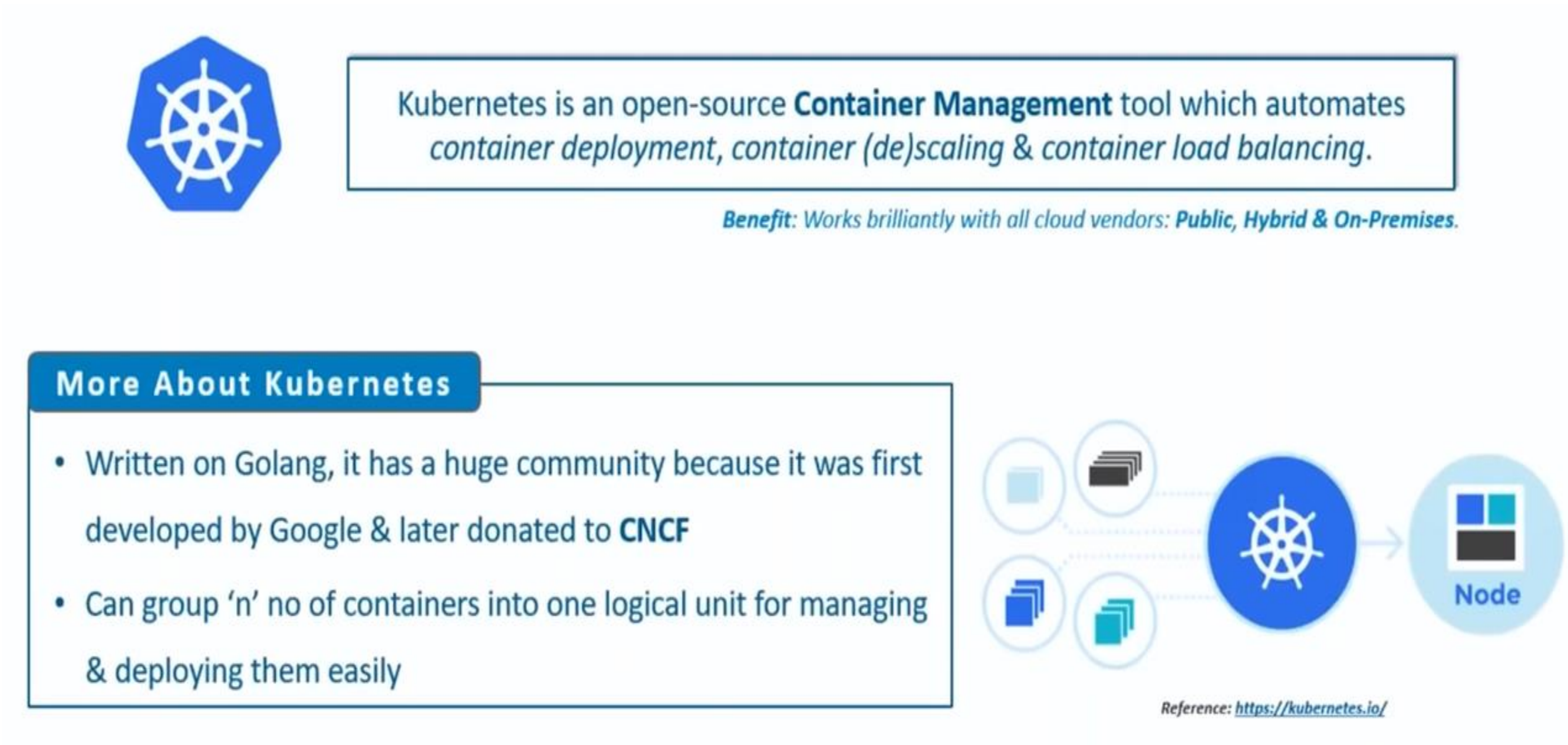
Problems with scaling up the Containers

- Communicate with each other & work together to basically host the service wrt application
  - ✓ If not able to communicate then scaling of containers is waste
  - ✓ Have to be deployed appropriately because you cannot have the containers deployed on random places
  - ✓ Auto-scaling was never the feature of container/Docker . Results to the need of Container Mangement like Kubernetes
  - ✓ Traffic reaching threshhold– Scaling up & Down based on need is a real challenge ➔ That's what kubernetes does


It was not Scalable because...
1 Containers could not **communicate** with each other
2 Containers had to be **deployed appropriately**
3 Containers had to be **managed carefully**
4 **Auto scaling** was not possible
5 **Distributing traffic** was still challenging

A Container Management Tool !!!

- **Kubernetes** (K8s) is a Container Management tool
  - ✓ an open source orchestration system for Docker containers
  - ✓ Google born product; written in Go language
  - ✓ Focuses on building a robust platform for running thousands of containers in production.
  - ✓ It simplifies DevOps tasks such as deployment, scaling, configuration, versioning, and rolling updates.



Kubernetes is an open-source **Container Management** tool which automates *container deployment, container (de)scaling & container load balancing.*

***Benefit**: Works brilliantly with all cloud vendors: **Public, Hybrid & On-Premises**.*

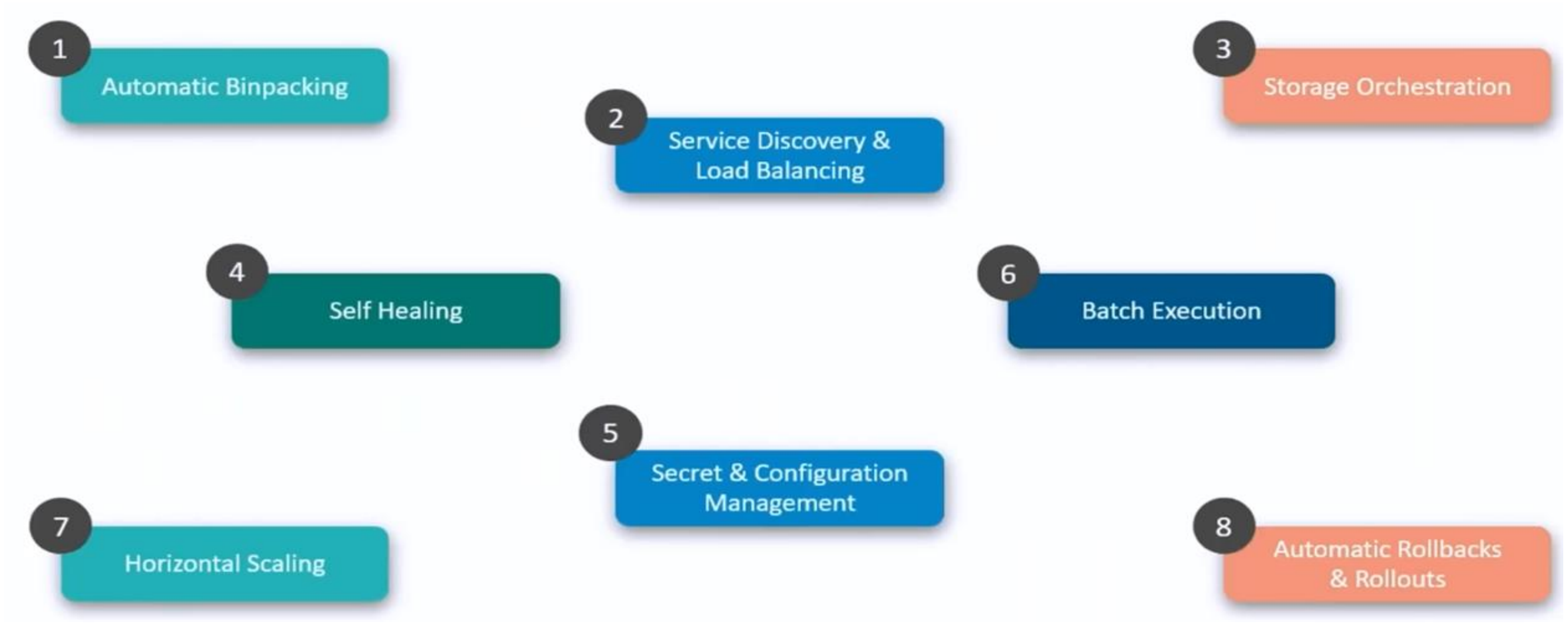
More About Kubernetes

- Written on Golang, it has a huge community because it was first developed by Google & later donated to **CNCF**
- Can group 'n' no of containers into one logical unit for managing & deploying them easily

Reference: https://kubernetes.io/

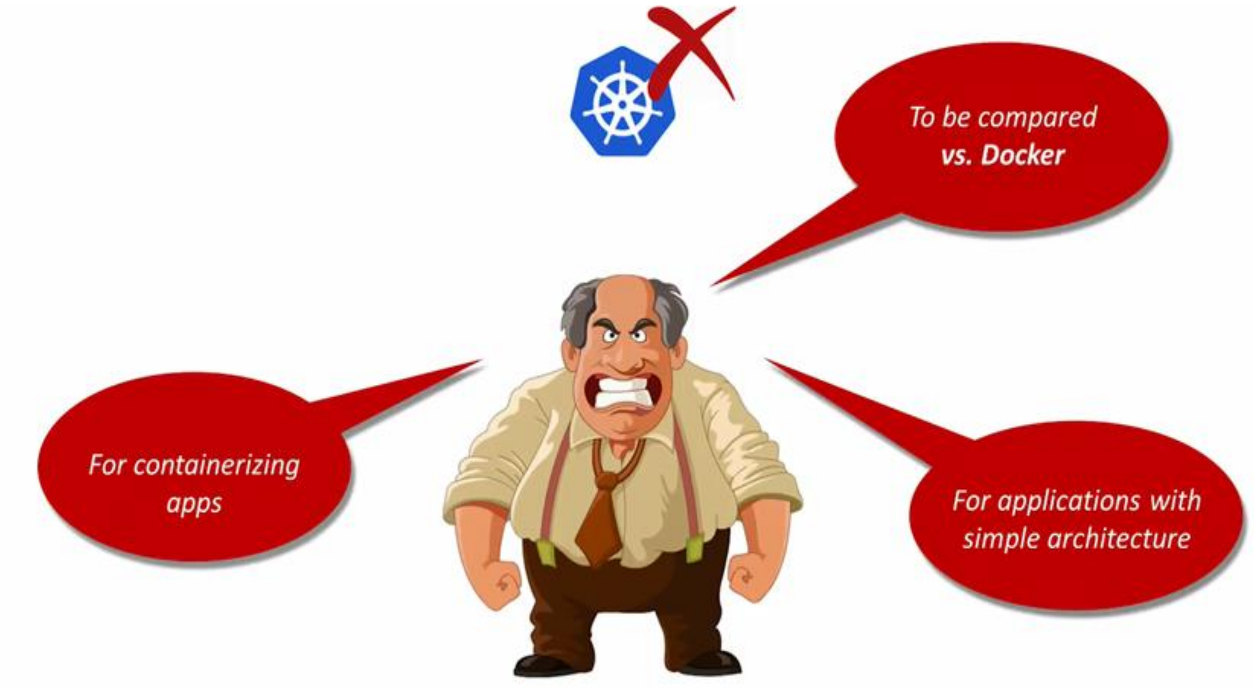
Key Features of the Kubernetes

- Following is the key features of Kubernetes:
  - ✓ kubernetes packages your application and it automatically places containers based on their requirements
  - ✓ Don't need to worry about networking and communication because it does automatically IP Address
  - ✓ With kubernetes we can **automatically Mount storage system of our choice**
  - ✓ Self-healing is best feature of kubernetes, restart containers if fails or all the containers/pods of node fails
  - ✓ Manage batch and CI workloads which is more of a devops
  - ✓ Can easily do horizontal scaling using GUI/ kubernetes dashboard: Automatic scale-up & down
  - ✓ Whenever there is an update in application , kubernetes progressively rose out these changes to automatic rollbacks & rollouts


1
Automatic Binpacking
2
Service Discovery & Load Balancing
3
Storage Orchestration
4
Self Healing
6
Batch Execution
5
Secret & Configuration Management
7
Horizontal Scaling
8
Automatic Rollbacks & Rollouts

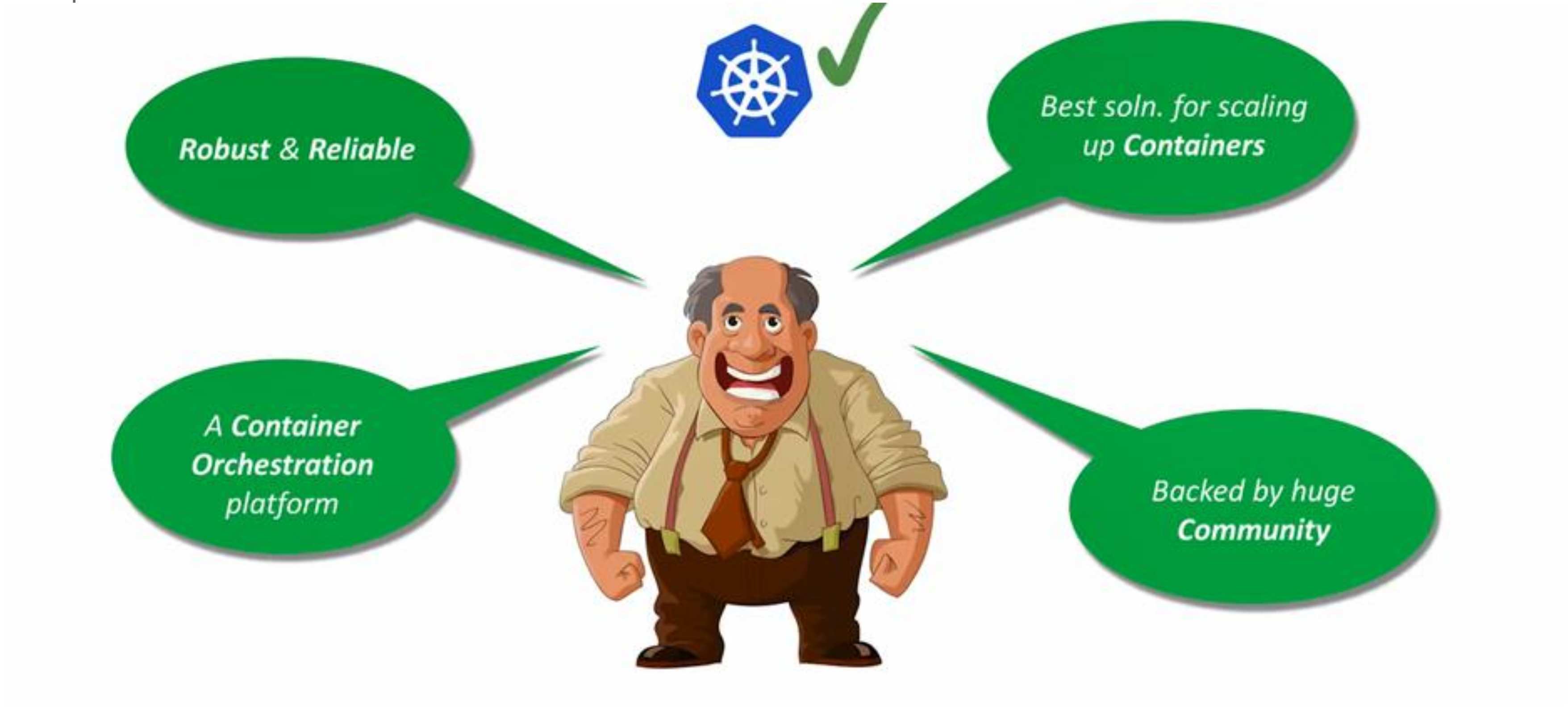
Myths wrt Kubernetes

- Kubernetes is like Docker. To be compared vs Docker
  - ✓ is not to be compared with Docker because it's not the right set of parameters against docker which is a containerization platform


To be compared vs. Docker
For containerizing apps
For applications with simple architecture

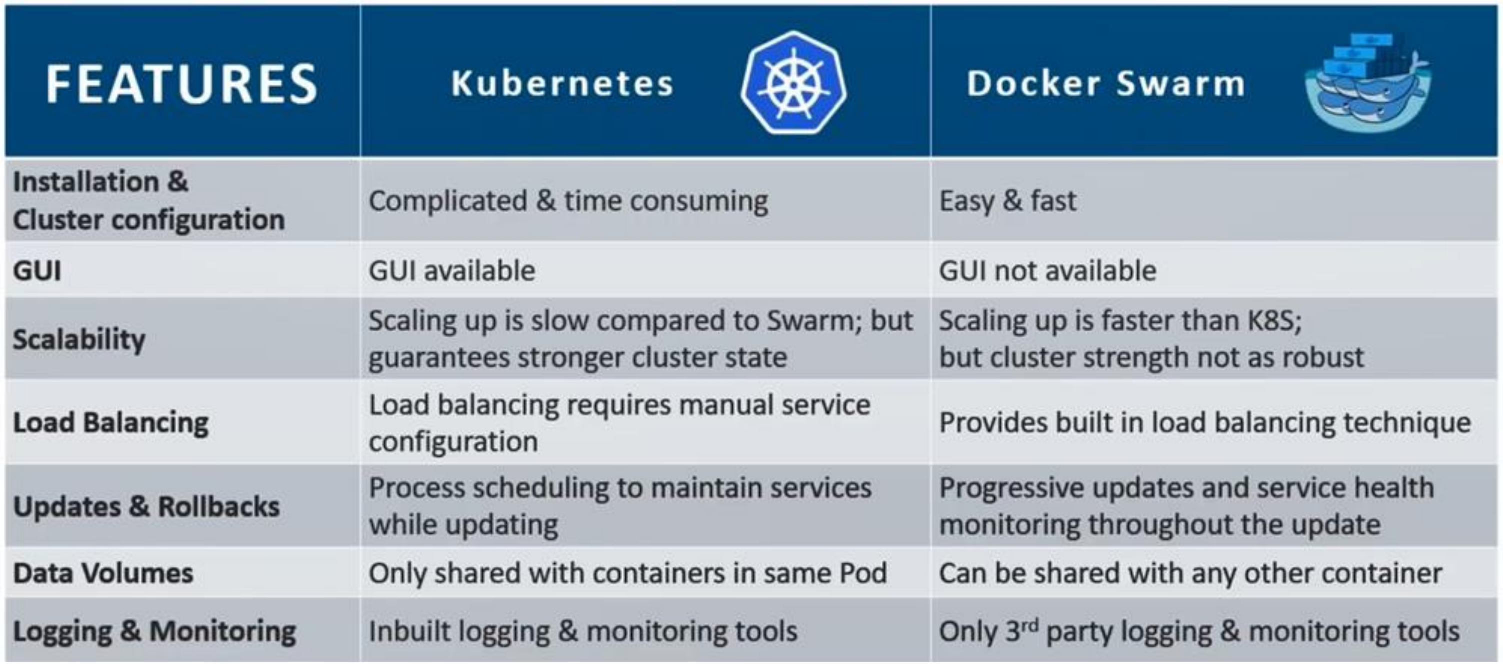
What Kubernetes is ?

- kubernetes first point is **robust and reliable**
  - Its cannot be broken easily the reason being configuration what we specify …at any point of time if any container fails, a new container would come up or that whole container would be restarted
- is a container orchestration platform. **kubernetes actually is the best solution for scaling of containers** best in today's market .. the two biggest players in current market Docker Swarm and kubernetes
- **Backed by huge community;** put your error there then you will have a lot of people on github.com and answer your queries and on stackoverflow


Robust & Reliable
Best soln. for scaling up Containers
A Container Orchestration platform
Backed by huge Community

Kubernetes Vs Docker-Swarm

- First parameter: docker Swarm comes on top because its little easier we have around two or three command by which our cluster start up & running
  - ✓ *kubernetes its more complicated than the DockerSwarm. But once cluster is ready that time kubernetes is the winner because the flexibility the rigidness and robustness that kubernetes provides*
- Once cluster is ready you can **use GUI with kubernetes for deploying applications**, monitoring & managing
- Wrt Load-balancing its shortfall of kubernetes because with docker Swarm there is inbuilt load balancing technique
- When something goes wrong kubernetes does the extra mile of doing the rollback and putting you back to the previous version
- Data volumes in kubernetes can be shared with other containers <u>but only within the same pod</u>. Concept of Pods
- W hen it comes logging and monitoring kubernetes provides inbuilt tools for this purpose where as docker Swarm have to install third party tools

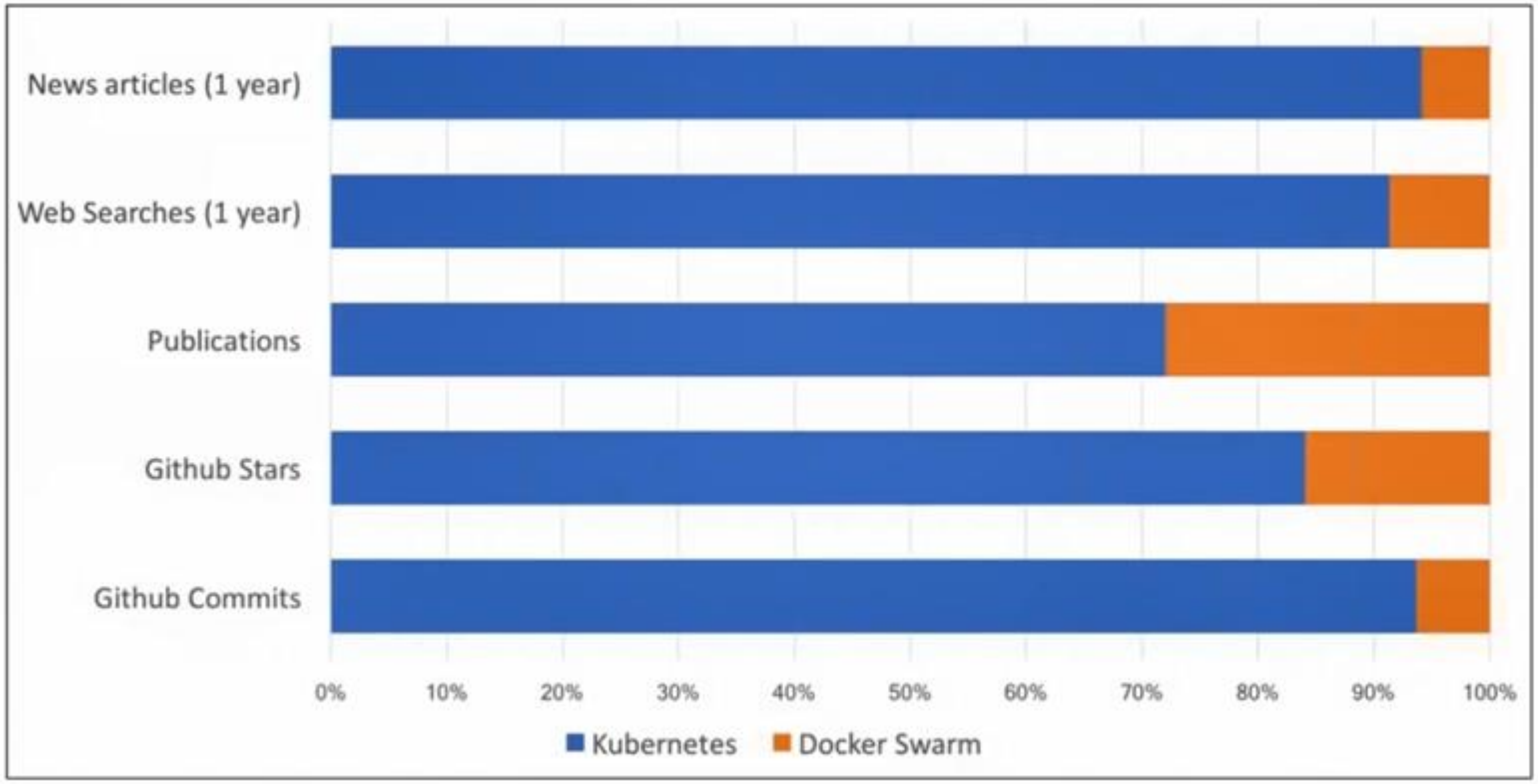


FEATURES	Kubernetes	Docker Swarm
Installation & Cluster configuration	Complicated & time consuming	Easy & fast
GUI	GUI available	GUI not available
Scalability	Scaling up is slow compared to Swarm; but guarantees stronger cluster state	Scaling up is faster than K8S; but cluster strength not as robust
Load Balancing	Load balancing requires manual service configuration	Provides built in load balancing technique
Updates & Rollbacks	Process scheduling to maintain services while updating	Progressive updates and service health monitoring throughout the update
Data Volumes	Only shared with containers in same Pod	Can be shared with any other container
Logging & Monitoring	Inbuilt logging & monitoring tools	Only 3rd party logging & monitoring tools

OPTUM™

Kubernetes Vs Docker-Swarm Mind share

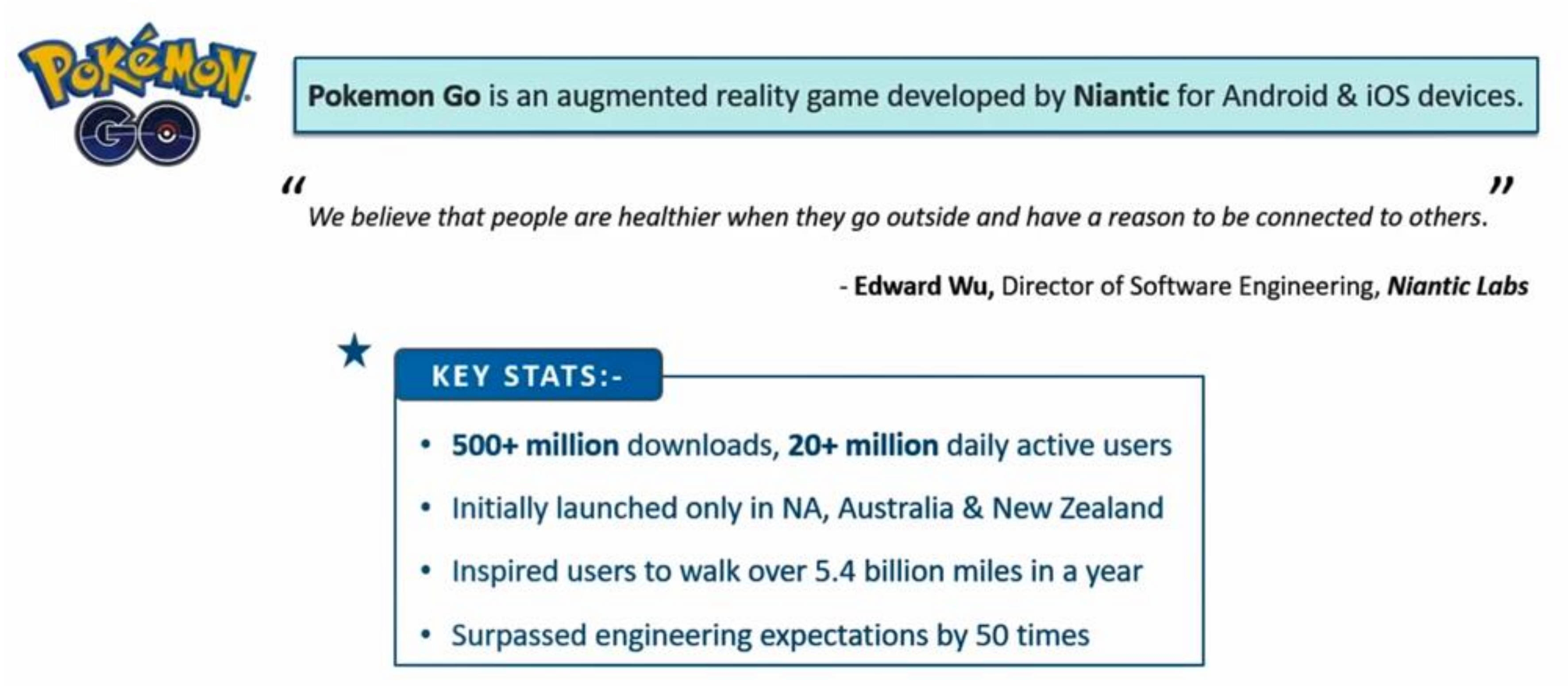
- Statistics published by paltform9 company.
  - ✓ *Numberof news articles that are produced one particular year at 90% of those cover on kubernetes compared to the 10% on docker Swarm*
  - ✓ Big difference that means for every 1 blog written for docker-swarm vs every 9 articles written on kubernetes
  - ✓ Similarly for web searches, GitHub for that's what is kubernetes is 90



Reference: https://platform9.com/blog/kubernetes-docker-swarm-compared/

Pokemon Go Using Kubernetes: Use-Case

- Kubernetes @ Pokemon Go
  - ✓ *Amazing game **Pokemon Go was powered with the help of kubernetes***
  - ✓ *Pokemon go is an Augmented reality game developed by Niantic for Android and for iOS devices. Key stats that they have 500 million + downloads overall and 20 million plus daily active users, now that is massive*



Pokemon Go is an augmented reality game developed by **Niantic** for Android & iOS devices.

"We believe that people are healthier when they go outside and have a reason to be connected to others."

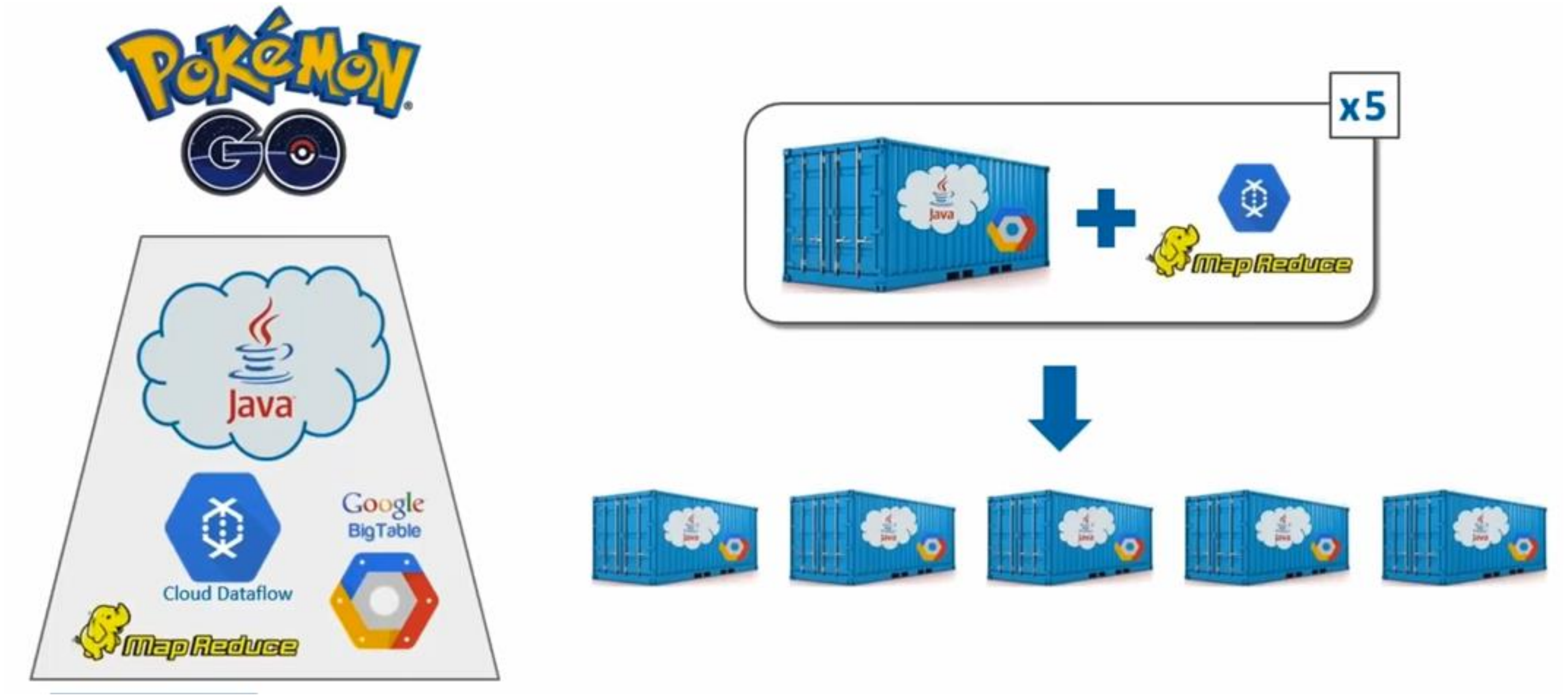
- **Edward Wu,** Director of Software Engineering, ***Niantic Labs***

KEY STATS:-

- **500+ million** downloads, **20+ million** daily active users
- Initially launched only in NA, Australia & New Zealand
- Inspired users to walk over 5.4 billion miles in a year
- Surpassed engineering expectations by 50 times

Pokemon: Ease Scaling of Containers using Kubernetes

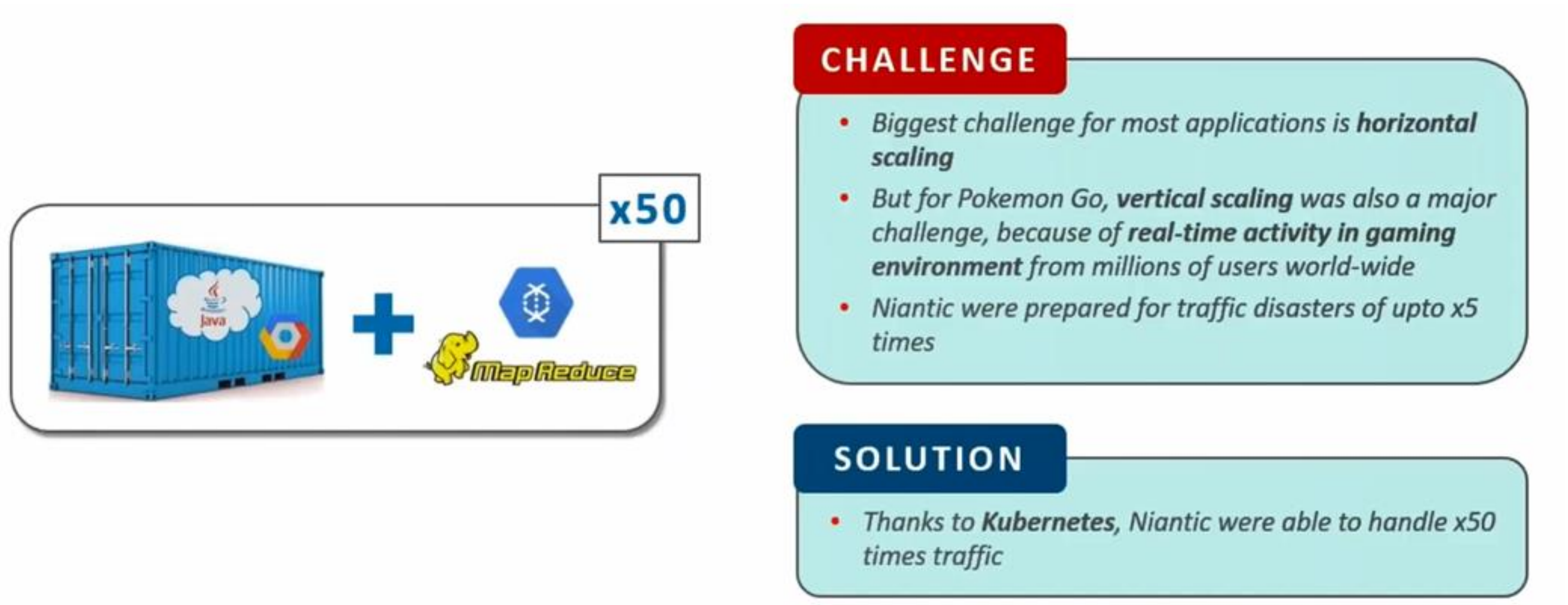
- Most interesting part is Backend architecture of Pokemon Go
  - ✓ *Pokemon go container which **had two primary component a) one is Google big table** which is main database from where everything is going and coming out b) and other is **program** running on Java cloud*.
  - ✓ Mapreduce and cloud data flow; it was used for scaling up ..so it's not just the container scaling up but it's with respect to the application how the programmer react when they have these increase number of users and how to handle increase number of requests.



Pokemon: Challenges solved using Kubernetes

- Pokemon go on releasing in just 3 different geographies .. became **so much popular that it was not a member of 5x** times which was the original service capability.. but **the traffic that they got was up to 50 times more than what they expected**
- Suddenly traffic request start coming in are so much that reaches 50 X.**so that's where kubernetes comes in and they overcome all the challenges** ..

- **kubernetes can do both vertical scaling and horizontal scaling ..** .. they figured out the way to actually scale up to 50 time in a very short time



CHALLENGE

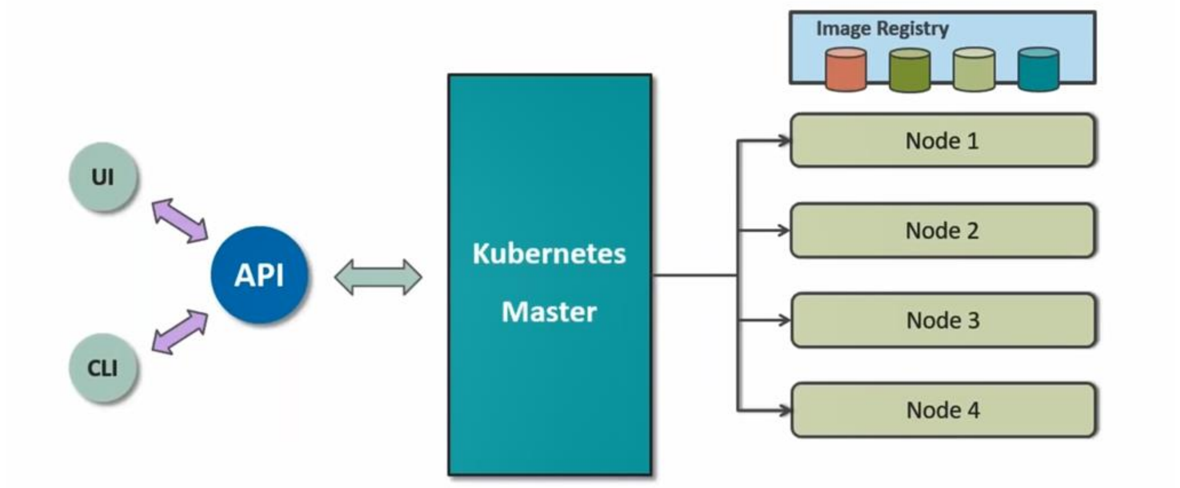
- *Biggest challenge for most applications is **horizontal scaling***
- *But for Pokemon Go, **vertical scaling** was also a major challenge, because of **real-time activity in gaming environment** from millions of users world-wide*
- *Niantic were prepared for traffic disasters of upto x5 times*

SOLUTION

- *Thanks to **Kubernetes**, Niantic were able to handle x50 times traffic*

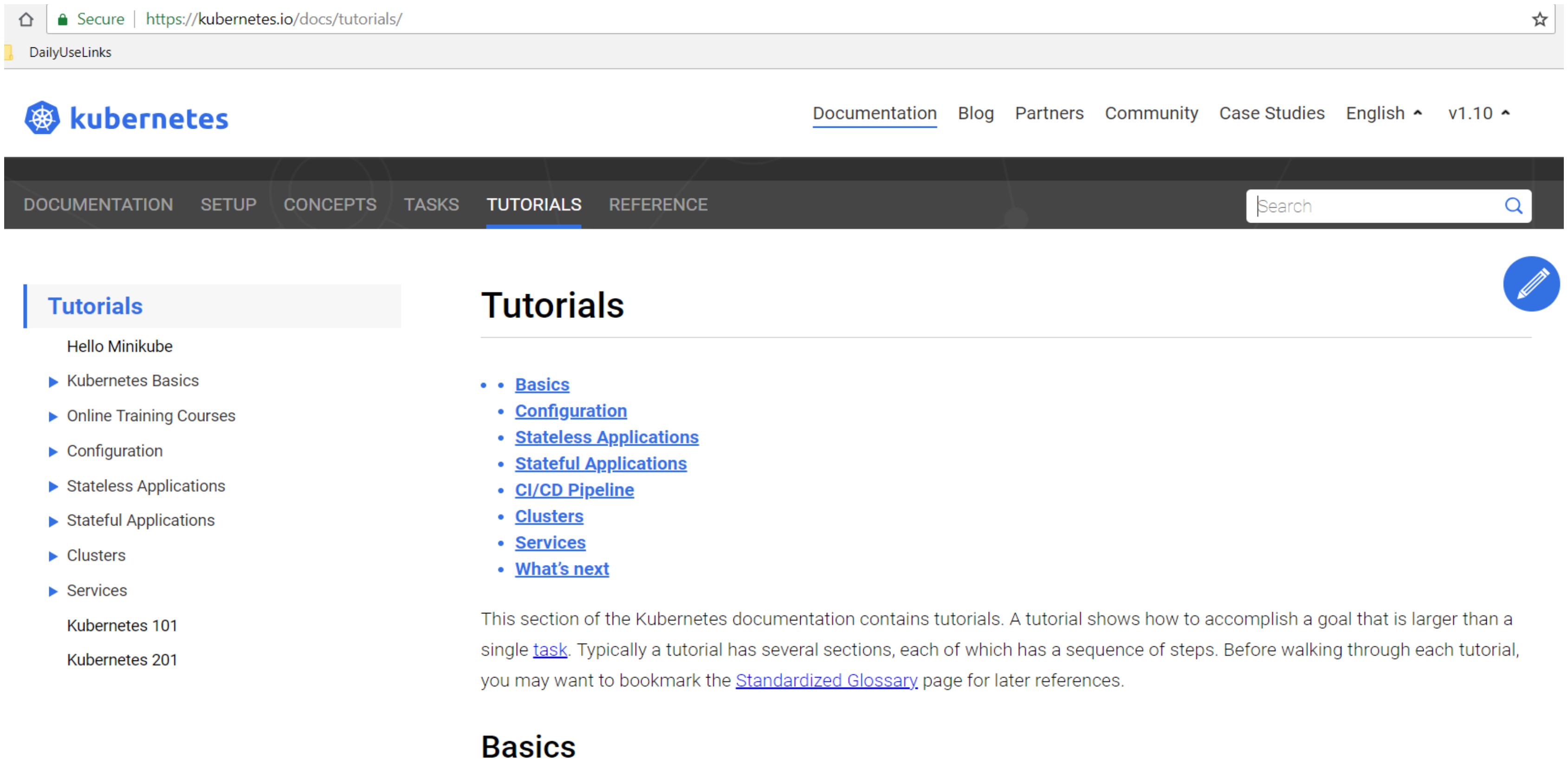
Architecture of Kubernetes

- **Kubernetes cluster consists of at least one master and multiple compute nodes:**
  - ✓ **Master**
    - Control services in a Kubernetes cluster
    - Master is responsible for exposing the application program interface (API)
    - In-charge of the cluster and monitor the cluster, scheduling the deployments and managing the overall cluster
  - ✓ **Node**
    - Each node runs a container runtime, such as Docker or rkt, along with an agent that communicates with the master.
    - Nodes are the workhorses of a Kubernetes cluster.
    - node also runs additional components for logging, monitoring, service discovery
  - ✓ **Pod :** A pod is a collection of one or more containers, it serves as Kubernetes' core unit of management
  - ✓ **Replica sets :** deliver the required scale and availability by maintaining a pre-defined set of pods at all times
  - ✓ **Services:** Single pod or a replica set can be exposed to the internal or external consumers via services



Kubernetes Learn Sources: https://kubernetes.io/docs/tutorials/

- Kubernetes Tutorial:
  - ✓ Provides Kubernetes Basics
  - ✓ Provides online environment for hands-on excersices
  - ✓ Detailed information on Kubectl CLI and Pods



Tutorials
- Hello Minikube
- Kubernetes Basics
- Online Training Courses
- Configuration
- Stateless Applications
- Stateful Applications
- Clusters
- Services
- Kubernetes 101
- Kubernetes 201

Tutorials

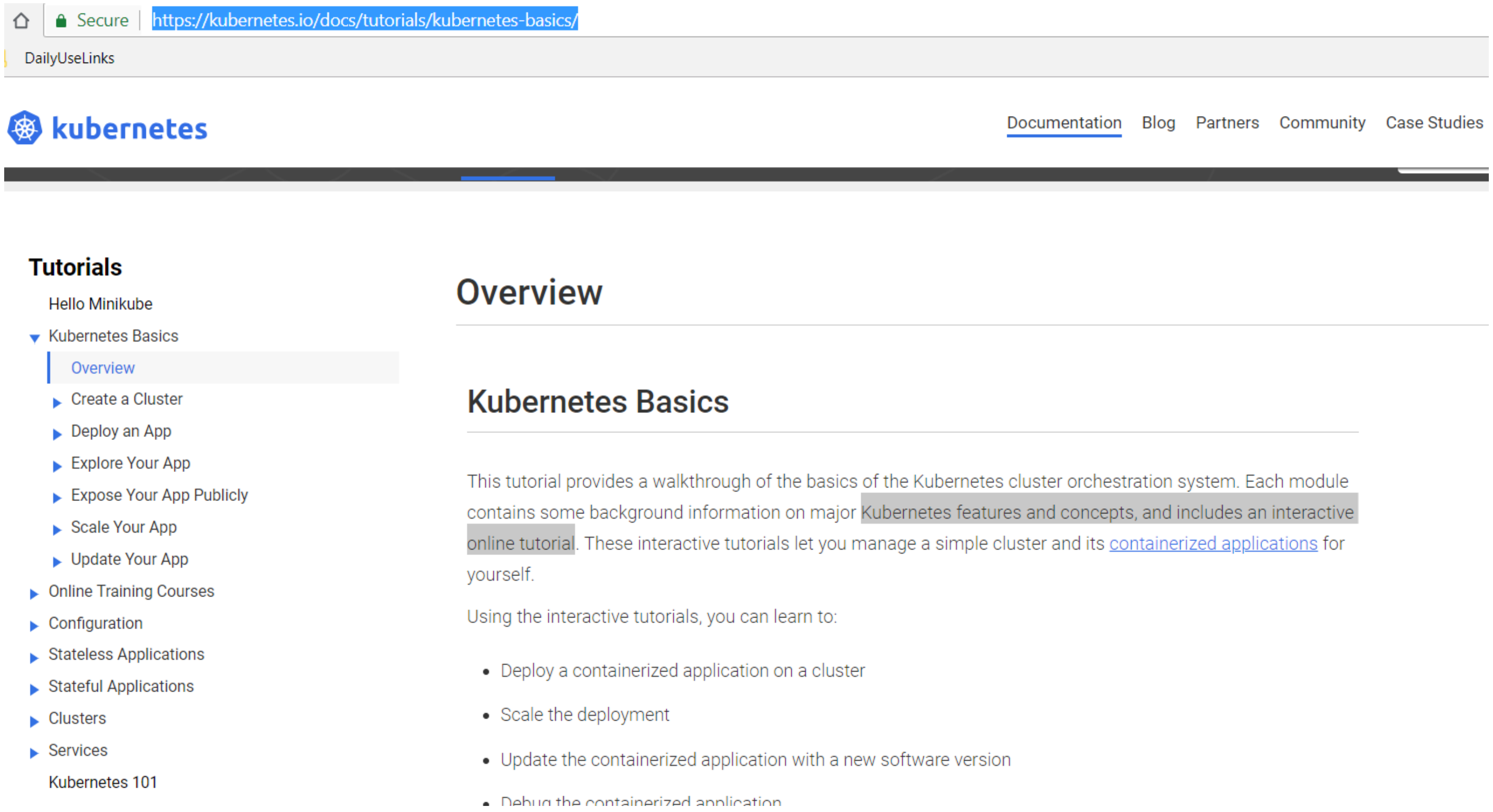
- Basics
- Configuration
- Stateless Applications
- Stateful Applications
- CI/CD Pipeline
- Clusters
- Services
- What's next

This section of the Kubernetes documentation contains tutorials. A tutorial shows how to accomplish a goal that is larger than a single task. Typically a tutorial has several sections, each of which has a sequence of steps. Before walking through each tutorial, you may want to bookmark the Standardized Glossary page for later references.

Basics

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

- Kubernetes Basics:
  - ✓ Provides a walkthrough of the basics of the Kubernetes cluster orchestration system
  - ✓ Kubernetes features and concepts, and includes an interactive online tutorial



Tutorials

- Hello Minikube
- Kubernetes Basics
  - Overview
  - Create a Cluster
  - Deploy an App
  - Explore Your App
  - Expose Your App Publicly
  - Scale Your App
  - Update Your App
- Online Training Courses
- Configuration
- Stateless Applications
- Stateful Applications
- Clusters
- Services
- Kubernetes 101

Overview

Kubernetes Basics

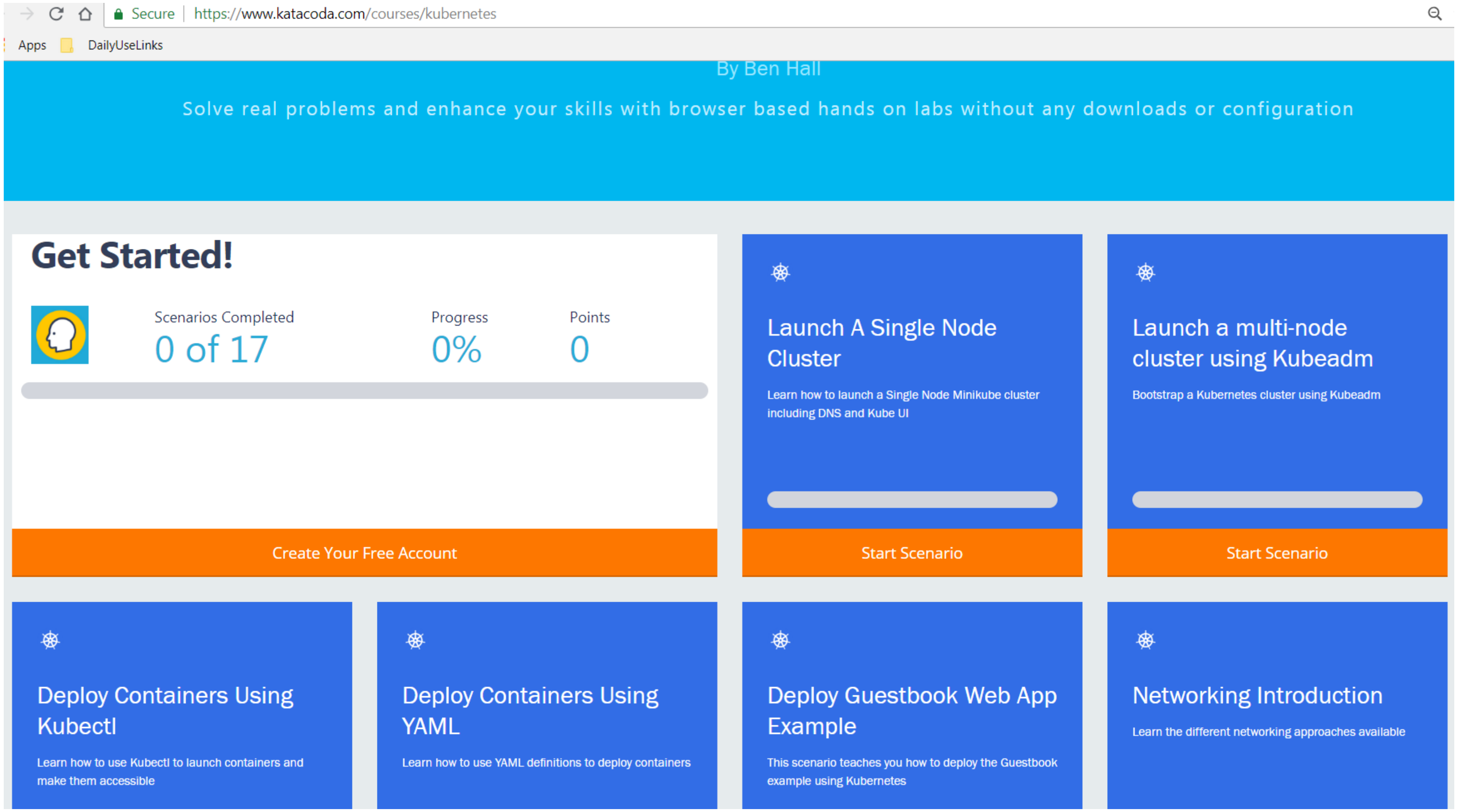
This tutorial provides a walkthrough of the basics of the Kubernetes cluster orchestration system. Each module contains some background information on major Kubernetes features and concepts, and includes an interactive online tutorial. These interactive tutorials let you manage a simple cluster and its containerized applications for yourself.

Using the interactive tutorials, you can learn to:

- Deploy a containerized application on a cluster
- Scale the deployment
- Update the containerized application with a new software version
- Debug the containerized application

Katacoda: Learn Kubernetes using Interactive Browser-Based Scenarios

- **Katacoda Scenarios**:
  - ✓ Katacoda have scenarios how to deploy containers , how to create cluster, multimode cluster
  - ✓ Deploy web app example, Running stateful services on Kubernetes



Secure | https://www.katacoda.com/courses/kubernetes

Apps DailyUseLinks

By Ben Hall

Solve real problems and enhance your skills with browser based hands on labs without any downloads or configuration

Get Started!

Scenarios Completed	Progress	Points
0 of 17	0%	0

Create Your Free Account

Launch A Single Node Cluster
Learn how to launch a Single Node Minikube cluster including DNS and Kube UI
Start Scenario

Launch a multi-node cluster using Kubeadm
Bootstrap a Kubernetes cluster using Kubeadm
Start Scenario

Deploy Containers Using Kubectl
Learn how to use Kubectl to launch containers and make them accessible

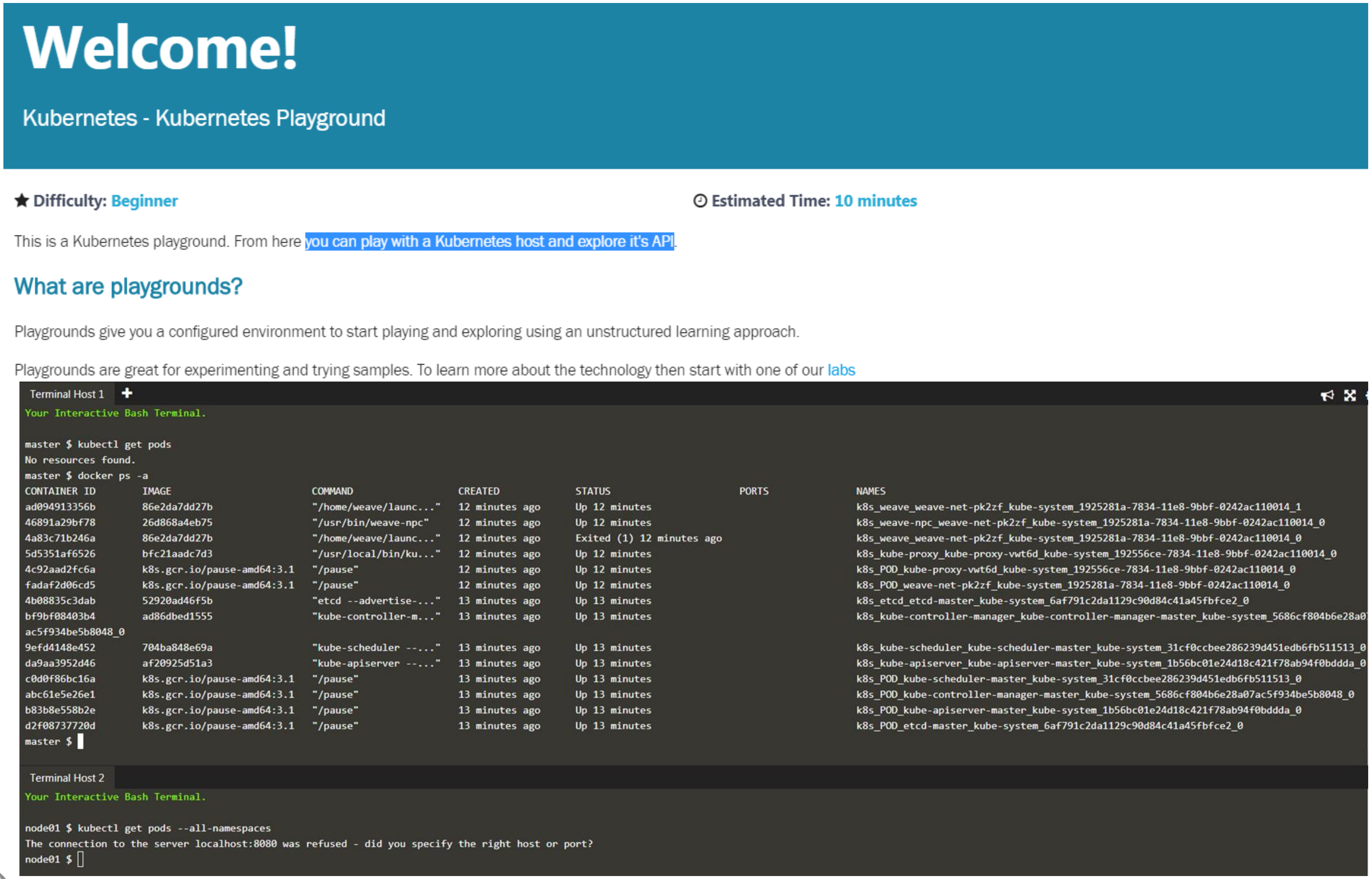
Deploy Containers Using YAML
Learn how to use YAML definitions to deploy containers

Deploy Guestbook Web App Example
This scenario teaches you how to deploy the Guestbook example using Kubernetes

Networking Introduction
Learn the different networking approaches available

Kubernetes Playground: https://www.katacoda.com/courses/kubernetes/playground

- Kubernetes Playground:
  - ✓ We can play with a Kubernetes host and explore it's API



Welcome!

Kubernetes - Kubernetes Playground

★ **Difficulty:** Beginner　　　　**Estimated Time:** 10 minutes

This is a Kubernetes playground. From here you can play with a Kubernetes host and explore it's API.

What are playgrounds?

Playgrounds give you a configured environment to start playing and exploring using an unstructured learning approach.

Playgrounds are great for experimenting and trying samples. To learn more about the technology then start with one of our labs

Terminal Host 1

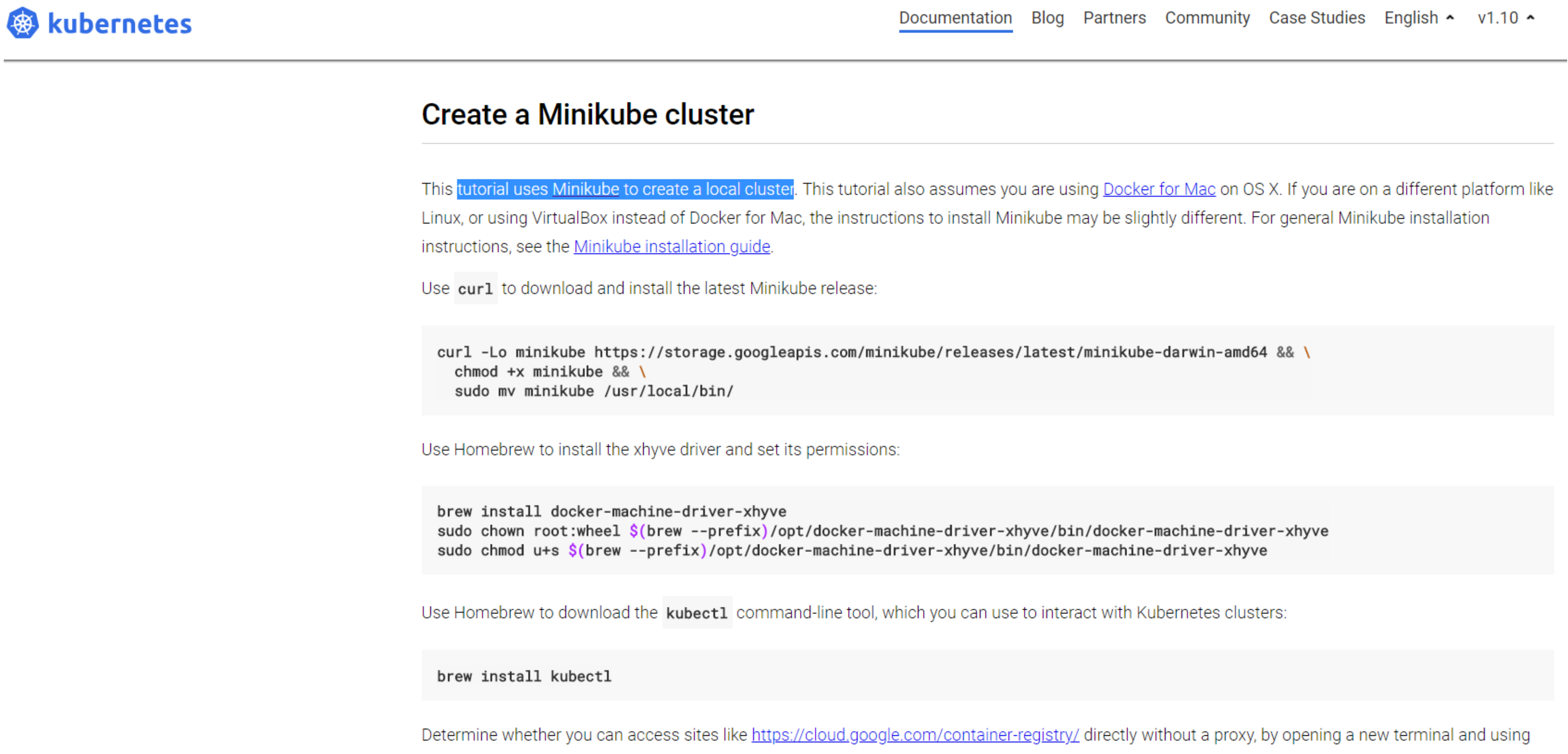
```
Your Interactive Bash Terminal.

master $ kubectl get pods
No resources found.
master $ docker ps -a
CONTAINER ID        IMAGE                         COMMAND                  CREATED             STATUS                        PORTS               NAMES
ad094913356b        86e2da7dd27b                  "/home/weave/launc..."   12 minutes ago      Up 12 minutes                                     k8s_weave_weave-net-pk2zf_kube-system_1925281a-7834-11e8-9bbf-0242ac110014_1
46891a29bf78        26d868a4eb75                  "/usr/bin/weave-npc"     12 minutes ago      Up 12 minutes                                     k8s_weave-npc_weave-net-pk2zf_kube-system_1925281a-7834-11e8-9bbf-0242ac110014_0
4a83c71b246a        86e2da7dd27b                  "/home/weave/launc..."   12 minutes ago      Exited (1) 12 minutes ago                         k8s_weave_weave-net-pk2zf_kube-system_1925281a-7834-11e8-9bbf-0242ac110014_0
5d5351af6526        bfc21aadc7d3                  "/usr/local/bin/ku..."   12 minutes ago      Up 12 minutes                                     k8s_kube-proxy_kube-proxy-vwt6d_kube-system_192556ce-7834-11e8-9bbf-0242ac110014_0
4c92aad2fc6a        k8s.gcr.io/pause-amd64:3.1    "/pause"                 12 minutes ago      Up 12 minutes                                     k8s_POD_kube-proxy-vwt6d_kube-system_192556ce-7834-11e8-9bbf-0242ac110014_0
fadaf2d06cd5        k8s.gcr.io/pause-amd64:3.1    "/pause"                 12 minutes ago      Up 12 minutes                                     k8s_POD_weave-net-pk2zf_kube-system_1925281a-7834-11e8-9bbf-0242ac110014_0
4b08835c3dab        52920ad46f5b                  "etcd --advertise-..."   13 minutes ago      Up 13 minutes                                     k8s_etcd_etcd-master_kube-system_6af791c2da1129c90d84c41a45fbfce2_0
bf9bf08403b4        ad86dbed1555                  "kube-controller-m..."   13 minutes ago      Up 13 minutes                                     k8s_kube-controller-manager_kube-controller-manager-master_kube-system_5686cf804b6e28a0
ac5f934be5b8048_0
9efd4148e452        704ba848e69a                  "kube-scheduler --..."   13 minutes ago      Up 13 minutes                                     k8s_kube-scheduler_kube-scheduler-master_kube-system_31cf0ccbee286239d451edb6fb511513_0
da9aa3952d46        af20925d51a3                  "kube-apiserver --..."   13 minutes ago      Up 13 minutes                                     k8s_kube-apiserver_kube-apiserver-master_kube-system_1b56bc01e24d18c421f78ab94f0bddda_0
c0d0f86bc16a        k8s.gcr.io/pause-amd64:3.1    "/pause"                 13 minutes ago      Up 13 minutes                                     k8s_POD_kube-scheduler-master_kube-system_31cf0ccbee286239d451edb6fb511513_0
abc61e5e26e1        k8s.gcr.io/pause-amd64:3.1    "/pause"                 13 minutes ago      Up 13 minutes                                     k8s_POD_kube-controller-manager-master_kube-system_5686cf804b6e28a07ac5f934be5b8048_0
b83b8e558b2e        k8s.gcr.io/pause-amd64:3.1    "/pause"                 13 minutes ago      Up 13 minutes                                     k8s_POD_kube-apiserver-master_kube-system_1b56bc01e24d18c421f78ab94f0bddda_0
d2f08737720d        k8s.gcr.io/pause-amd64:3.1    "/pause"                 13 minutes ago      Up 13 minutes                                     k8s_POD_etcd-master_kube-system_6af791c2da1129c90d84c41a45fbfce2_0
master $
```

Terminal Host 2

```
Your Interactive Bash Terminal.

node01 $ kubectl get pods --all-namespaces
The connection to the server localhost:8080 was refused - did you specify the right host or port?
node01 $
```

Kubernetes Minikube cluster: https://kubernetes.io/docs/tutorials/hello-minikube/

- Create a Minikube cluster:
  - ✓ Tutorial uses Minikube to create a local kubernetes cluster
  - ✓ With Minikube, Its easy to run Kubernetes locally without any efforts


kubernetes
Documentation Blog Partners Community Case Studies English v1.10

Create a Minikube cluster

This tutorial uses Minikube to create a local cluster. This tutorial also assumes you are using Docker for Mac on OS X. If you are on a different platform like Linux, or using VirtualBox instead of Docker for Mac, the instructions to install Minikube may be slightly different. For general Minikube installation instructions, see the Minikube installation guide.

Use `curl` to download and install the latest Minikube release:

```
curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-amd64 && \
  chmod +x minikube && \
  sudo mv minikube /usr/local/bin/
```

Use Homebrew to install the xhyve driver and set its permissions:

```
brew install docker-machine-driver-xhyve
sudo chown root:wheel $(brew --prefix)/opt/docker-machine-driver-xhyve/bin/docker-machine-driver-xhyve
sudo chmod u+s $(brew --prefix)/opt/docker-machine-driver-xhyve/bin/docker-machine-driver-xhyve
```

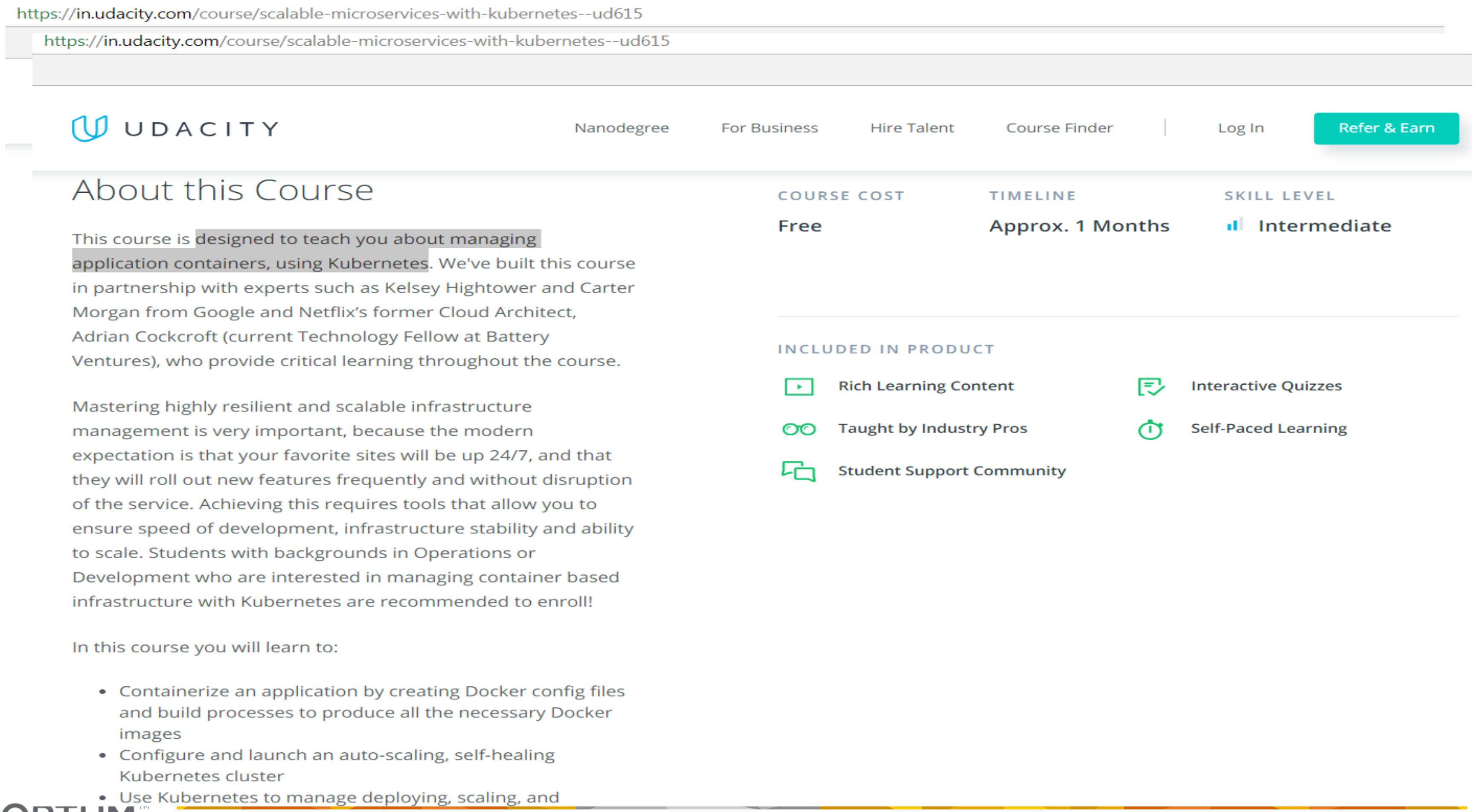
Use Homebrew to download the `kubectl` command-line tool, which you can use to interact with Kubernetes clusters:

```
brew install kubectl
```

Determine whether you can access sites like https://cloud.google.com/container-registry/ directly without a proxy, by opening a new terminal and using

Udacity: Start Free course on Kubernetes

- Start Free Couse on Udacity:
  - ✓ Designed Course to teach about managing application containers, using Kubernetes
  - ✓ Detailed Course Contents:
    - https://in.udacity.com/course/scalable-microservices-with-kubernetes--ud615


https://in.udacity.com/course/scalable-microservices-with-kubernetes--ud615
https://in.udacity.com/course/scalable-microservices-with-kubernetes--ud615
UDACITY
Nanodegree
For Business
Hire Talent
Course Finder
Log In
Refer & Earn

About this Course

This course is designed to teach you about managing application containers, using Kubernetes. We've built this course in partnership with experts such as Kelsey Hightower and Carter Morgan from Google and Netflix's former Cloud Architect, Adrian Cockcroft (current Technology Fellow at Battery Ventures), who provide critical learning throughout the course.

Mastering highly resilient and scalable infrastructure management is very important, because the modern expectation is that your favorite sites will be up 24/7, and that they will roll out new features frequently and without disruption of the service. Achieving this requires tools that allow you to ensure speed of development, infrastructure stability and ability to scale. Students with backgrounds in Operations or Development who are interested in managing container based infrastructure with Kubernetes are recommended to enroll!

In this course you will learn to:

- Containerize an application by creating Docker config files and build processes to produce all the necessary Docker images
- Configure and launch an auto-scaling, self-healing Kubernetes cluster
- Use Kubernetes to manage deploying, scaling, and

COURSE COST	TIMELINE	SKILL LEVEL
Free	Approx. 1 Months	Intermediate

INCLUDED IN PRODUCT

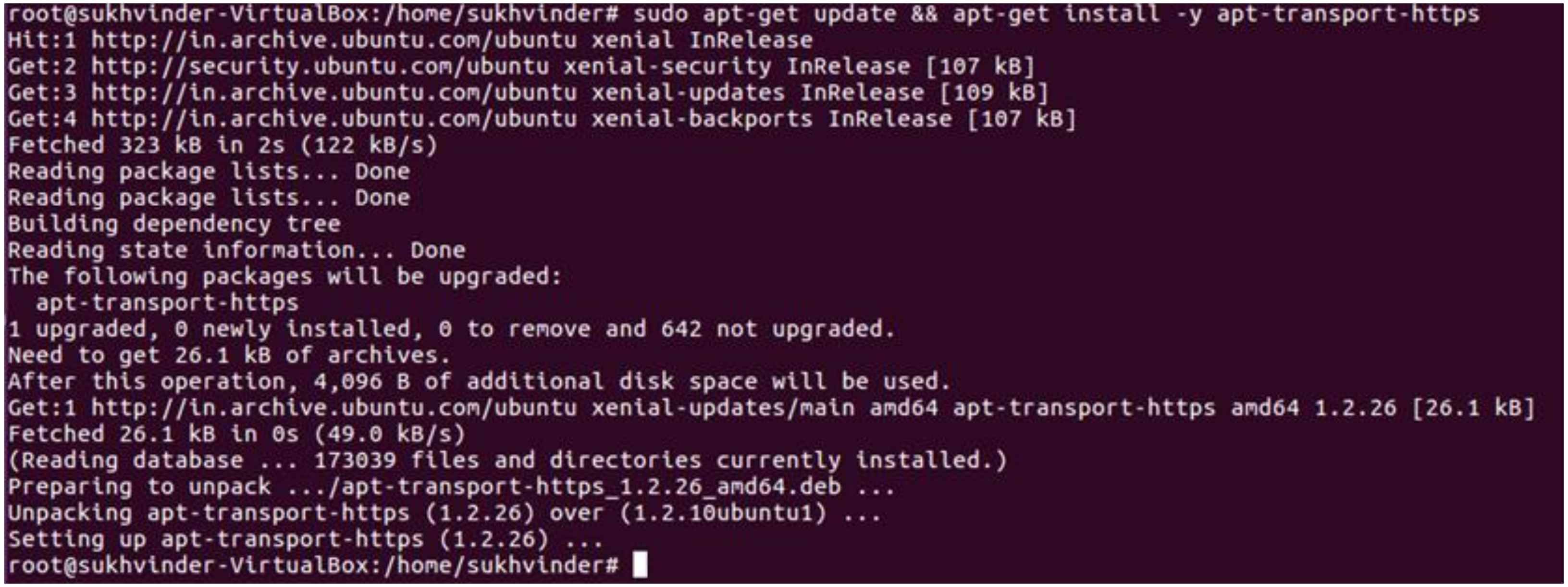
- Rich Learning Content
- Interactive Quizzes
- Taught by Industry Pros
- Self-Paced Learning
- Student Support Community

For developers : Create cluster from scratch in simple steps

- Start building your own cluster on fresh linux box:

Installing dependencies:

- ✓ The first piece to be install is apt-transport-https (a package that allows using https as well as http in apt repository sources)

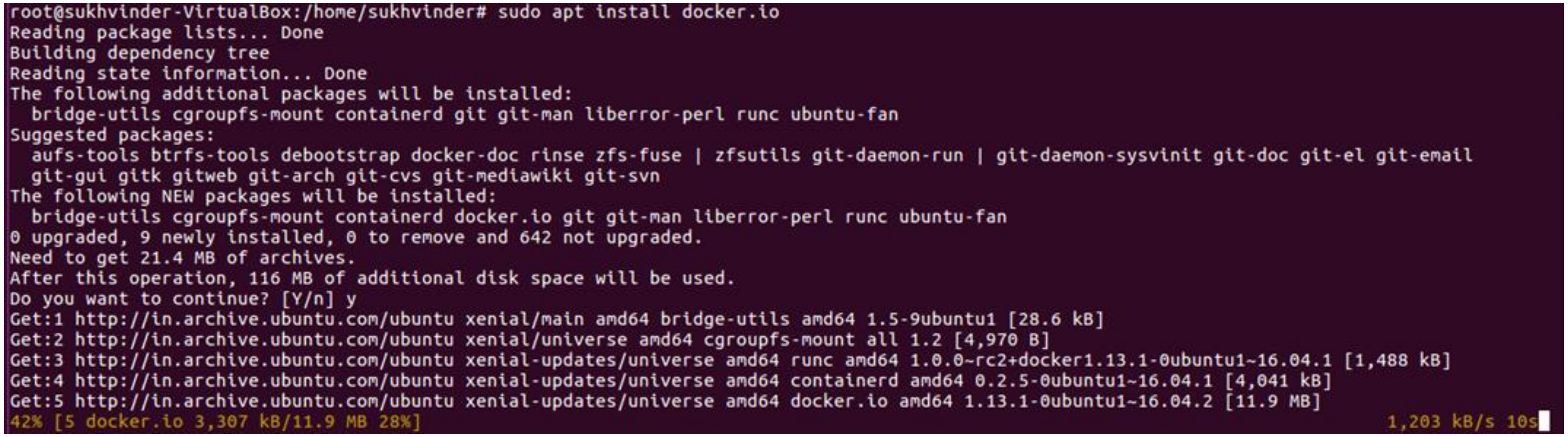


```
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo apt-get update && apt-get install -y apt-transport-https
Hit:1 http://in.archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://security.ubuntu.com/ubuntu xenial-security InRelease [107 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Fetched 323 kB in 2s (122 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be upgraded:
  apt-transport-https
1 upgraded, 0 newly installed, 0 to remove and 642 not upgraded.
Need to get 26.1 kB of archives.
After this operation, 4,096 B of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu xenial-updates/main amd64 apt-transport-https amd64 1.2.26 [26.1 kB]
Fetched 26.1 kB in 0s (49.0 kB/s)
(Reading database ... 173039 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_1.2.26_amd64.deb ...
Unpacking apt-transport-https (1.2.26) over (1.2.10ubuntu1) ...
Setting up apt-transport-https (1.2.26) ...
root@sukhvinder-VirtualBox:/home/sukhvinder# █
```

Create cluster from scratch in simple steps cont..

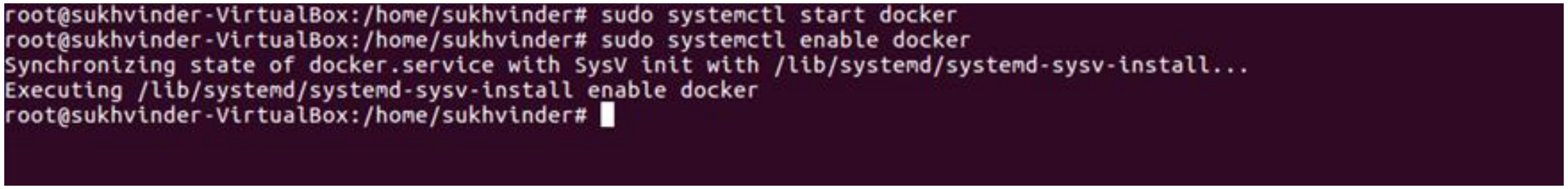
Installing dependencies:

- ✓ Our next dependency is Docker. Our Kubernetes installation will depend upon this, so install it with:



```
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo apt install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils cgroupfs-mount containerd git git-man liberror-perl runc ubuntu-fan
Suggested packages:
  aufs-tools btrfs-tools debootstrap docker-doc rinse zfs-fuse | zfsutils git-daemon-run | git-daemon-sysvinit git-doc git-el git-email
  git-gui gitk gitweb git-arch git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  bridge-utils cgroupfs-mount containerd docker.io git git-man liberror-perl runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 642 not upgraded.
Need to get 21.4 MB of archives.
After this operation, 116 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 bridge-utils amd64 1.5-9ubuntu1 [28.6 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu xenial/universe amd64 cgroupfs-mount all 1.2 [4,970 B]
Get:3 http://in.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 runc amd64 1.0.0~rc2+docker1.13.1-0ubuntu1~16.04.1 [1,488 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 containerd amd64 0.2.5-0ubuntu1~16.04.1 [4,041 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 docker.io amd64 1.13.1-0ubuntu1~16.04.2 [11.9 MB]
42% [5 docker.io 3,307 kB/11.9 MB 28%]                                                                          1,203 kB/s 10s
```

- ✓ Once that completes, start and enable the Docker service

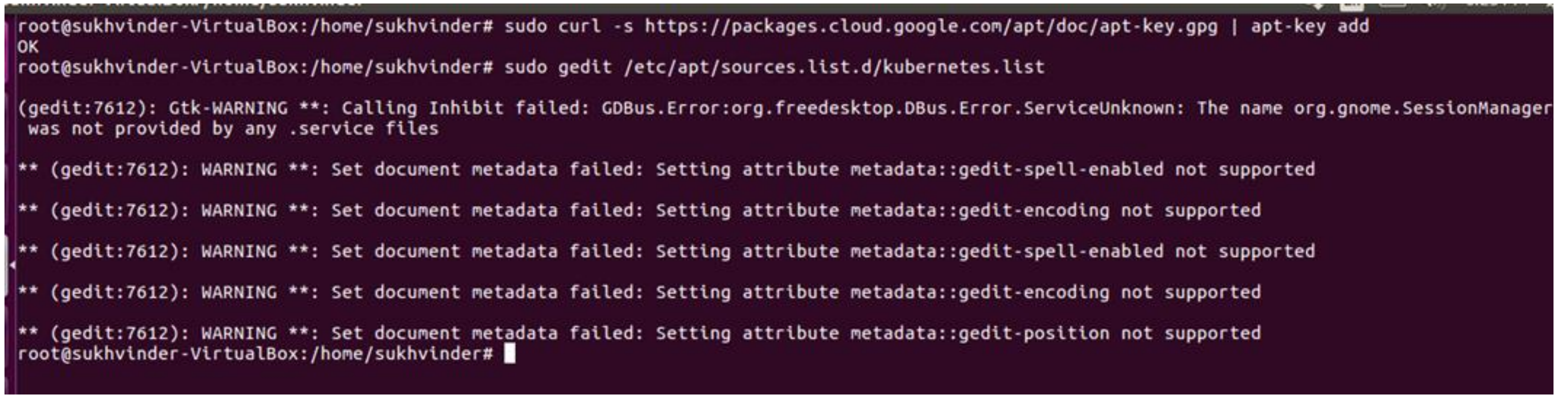


```
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo systemctl start docker
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo systemctl enable docker
Synchronizing state of docker.service with SysV init with /lib/systemd/systemd-sysv-install...
Executing /lib/systemd/systemd-sysv-install enable docker
root@sukhvinder-VirtualBox:/home/sukhvinder#
```

For developers : Create cluster from scratch in simple steps cont..

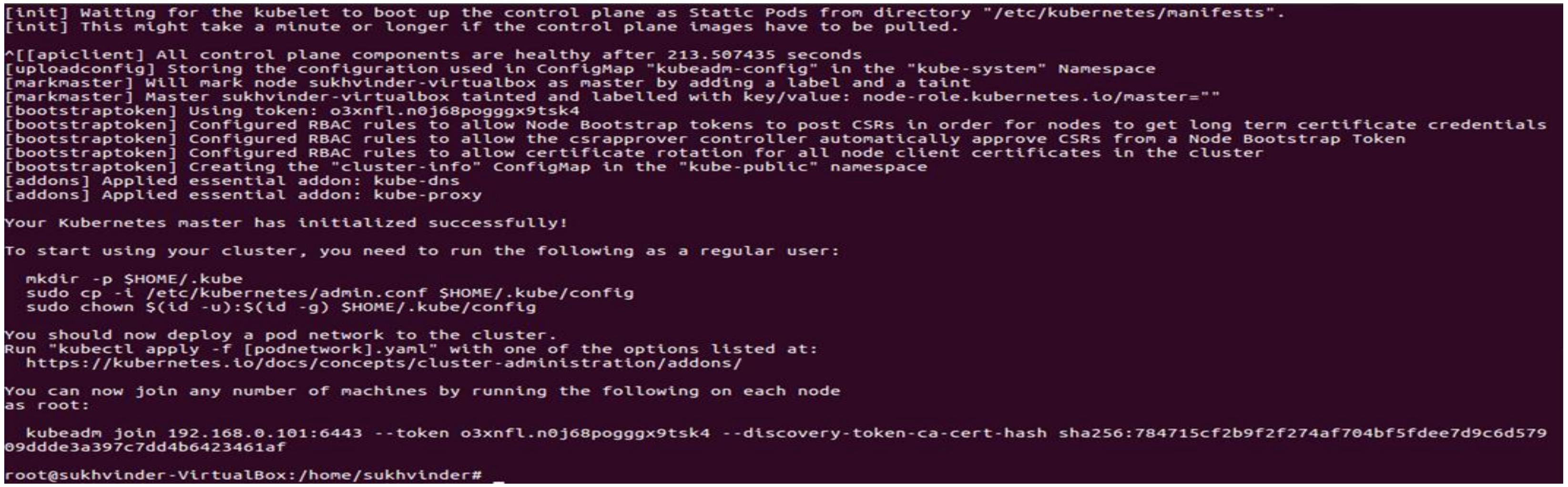
Installing Kubernetes:

- ✓ Our first step is to download and add the key for the Kubernetes install:
- ✓ sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add



```
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add
OK
root@sukhvinder-VirtualBox:/home/sukhvinder# sudo gedit /etc/apt/sources.list.d/kubernetes.list

(gedit:7612): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager
 was not provided by any .service files

** (gedit:7612): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-spell-enabled not supported

** (gedit:7612): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-encoding not supported

** (gedit:7612): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-spell-enabled not supported

** (gedit:7612): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-encoding not supported

** (gedit:7612): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-position not supported
root@sukhvinder-VirtualBox:/home/sukhvinder# 
```

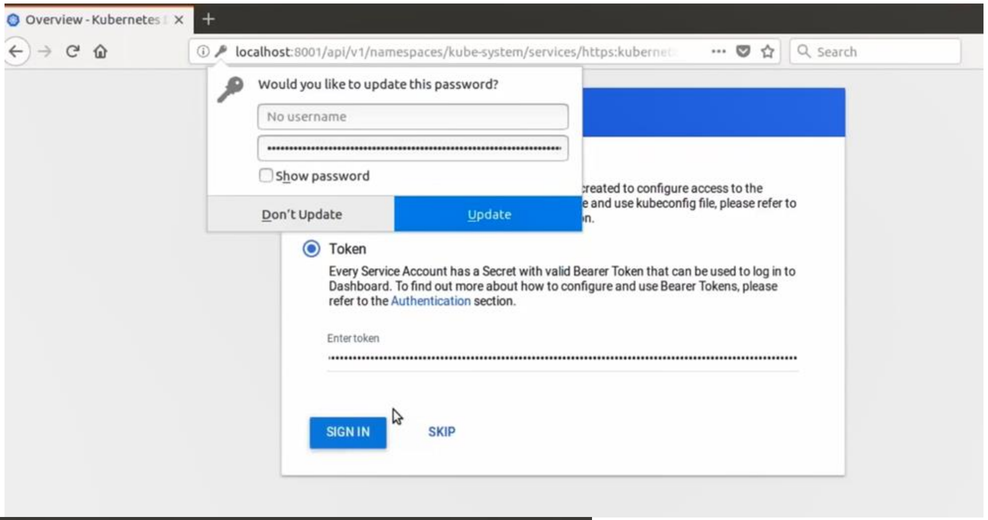
- ✓ Initialize your master, With everything installed: -sudo kubeadm init



```
[init] Waiting for the kubelet to boot up the control plane as Static Pods from directory "/etc/kubernetes/manifests".
[init] This might take a minute or longer if the control plane images have to be pulled.

^[[apiclient] All control plane components are healthy after 213.507435 seconds
[uploadconfig] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[markmaster] Will mark node sukhvinder-virtualbox as master by adding a label and a taint
[markmaster] Master sukhvinder-virtualbox tainted and labelled with key/value: node-role.kubernetes.io/master=""
[bootstraptoken] Using token: o3xnfl.n0j68pogggx9tsk4
[bootstraptoken] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstraptoken] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstraptoken] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstraptoken] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[addons] Applied essential addon: kube-dns
[addons] Applied essential addon: kube-proxy

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join 192.168.0.101:6443 --token o3xnfl.n0j68pogggx9tsk4 --discovery-token-ca-cert-hash sha256:784715cf2b9f2f274af704bf5fdee7d9c6d579
09ddde3a397c7dd4b6423461af

root@sukhvinder-VirtualBox:/home/sukhvinder# _
```

Kubernetes -Dashboard

Enabling kubernetes Dashboard:


Overview - Kubernetes

localhost:8001/api/v1/namespaces/kube-system/services/https:kuberne

Search

Would you like to update this password?

No username

Show password

Don't Update | Update

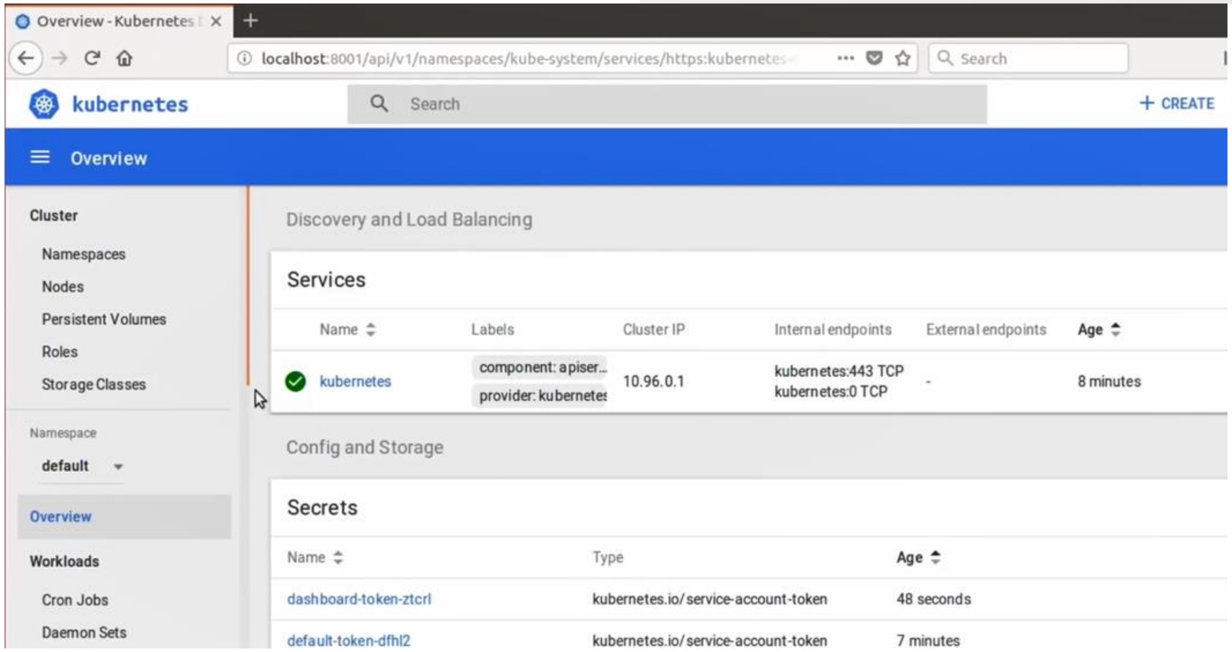
reated to configure access to the
e and use kubeconfig file, please refer to
n.

Token

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the Authentication section.

Enter token

SIGN IN | SKIP


Overview - Kubernetes

localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-

kubernetes | Search | + CREATE

Overview

Cluster
- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace: default

Overview

Workloads
- Cron Jobs
- Daemon Sets

Discovery and Load Balancing

Services

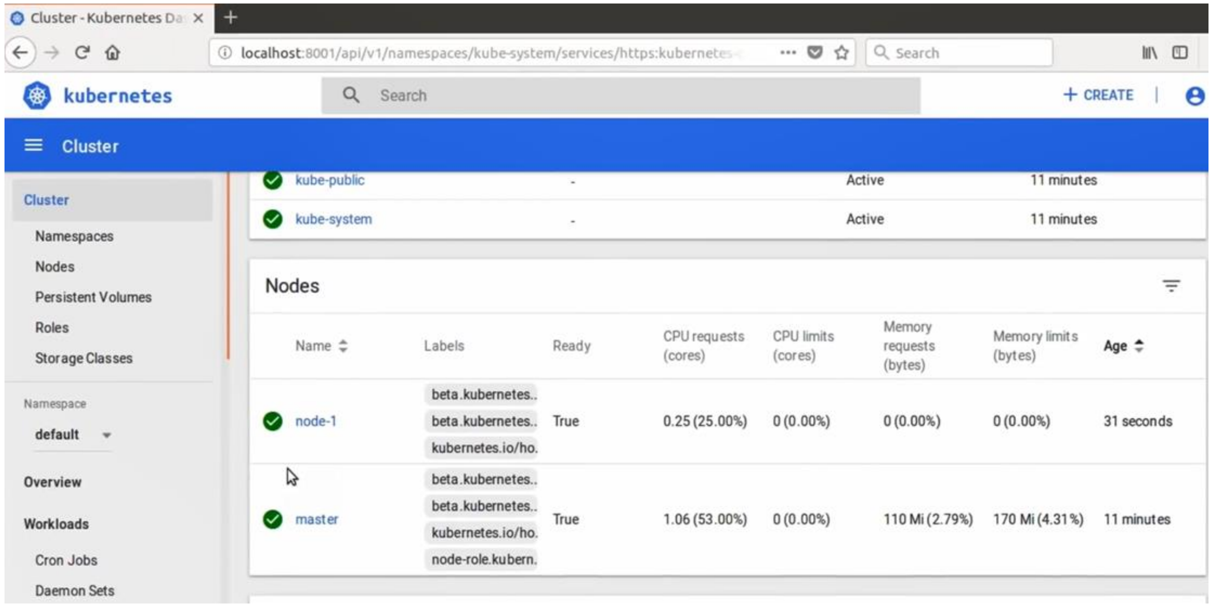
	Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
✓	kubernetes	component: apiser... provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	8 minutes

Config and Storage

Secrets

Name	Type	Age
dashboard-token-ztcrl	kubernetes.io/service-account-token	48 seconds
default-token-dfhl2	kubernetes.io/service-account-token	7 minutes

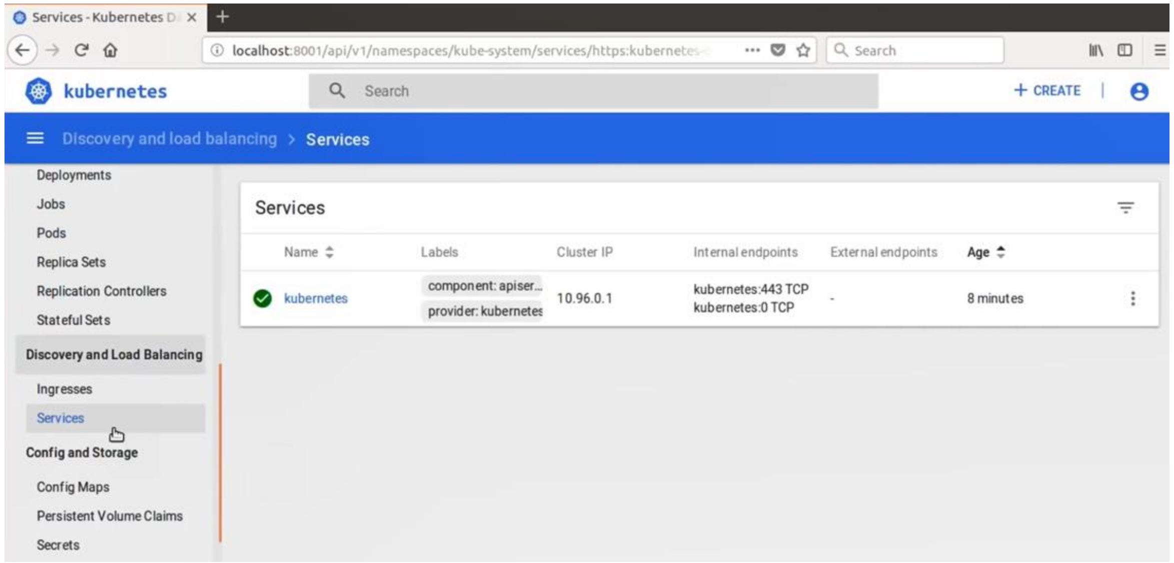
Dashboard: View Deployments, Pods, Services



kube-public	-	Active	11 minutes
kube-system	-	Active	11 minutes

Nodes

Name	Labels	Ready	CPU requests (cores)	CPU limits (cores)	Memory requests (bytes)	Memory limits (bytes)	Age
node-1	beta.kubernetes.. beta.kubernetes.. kubernetes.io/ho.	True	0.25 (25.00%)	0 (0.00%)	0 (0.00%)	0 (0.00%)	31 seconds
master	beta.kubernetes.. beta.kubernetes.. kubernetes.io/ho. node-role.kubern.	True	1.06 (53.00%)	0 (0.00%)	110 Mi (2.79%)	170 Mi (4.31%)	11 minutes

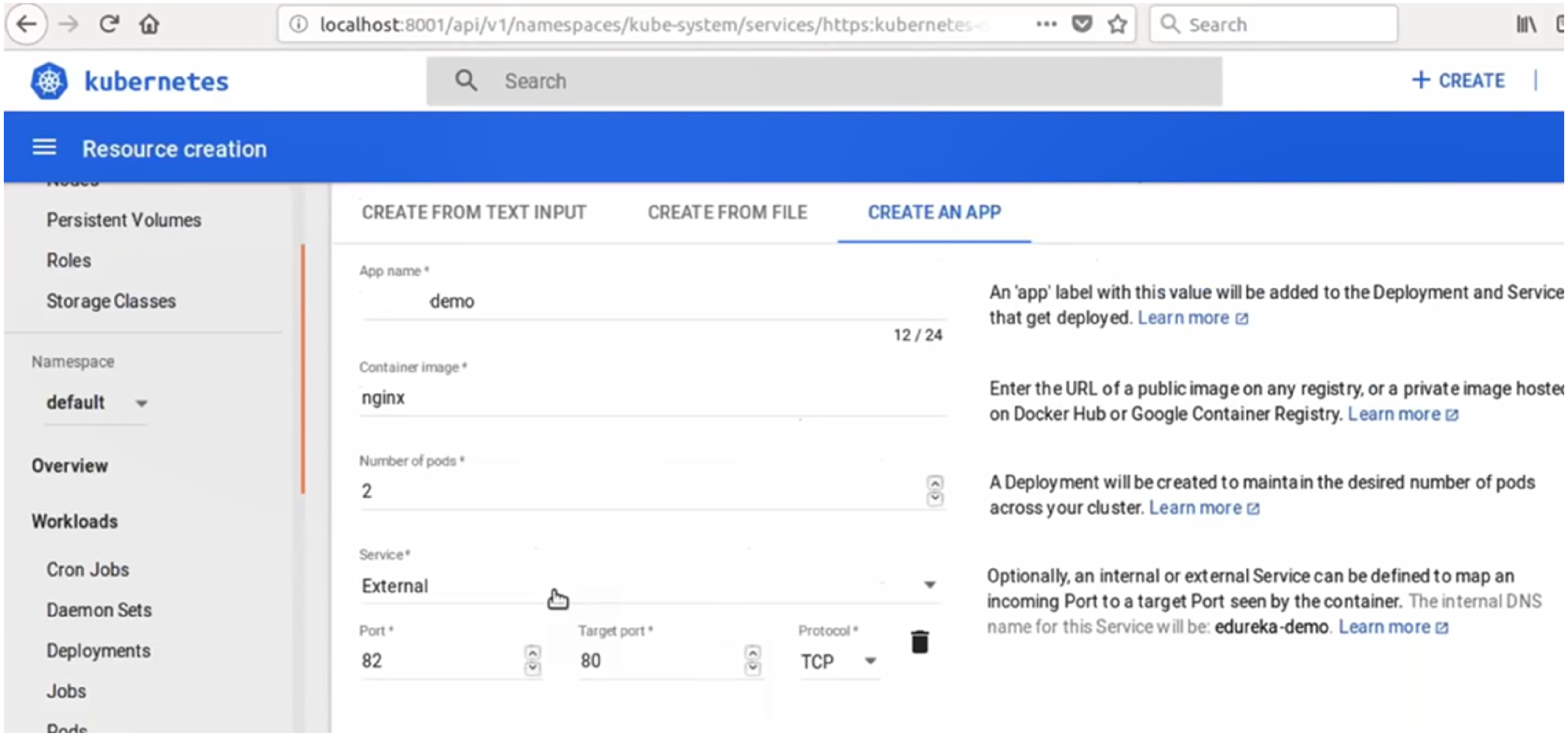


Services

Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
kubernetes	component: apiser... provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	8 minutes

Dashboard: Create an APP using GUI

- **We can set Container Image, number of Pods, Expose as Service, Port etc via GUI:**



App name * demo 12 / 24

Container image * nginx

Number of pods * 2

Service * External

Port * 82 Target port * 80 Protocol * TCP

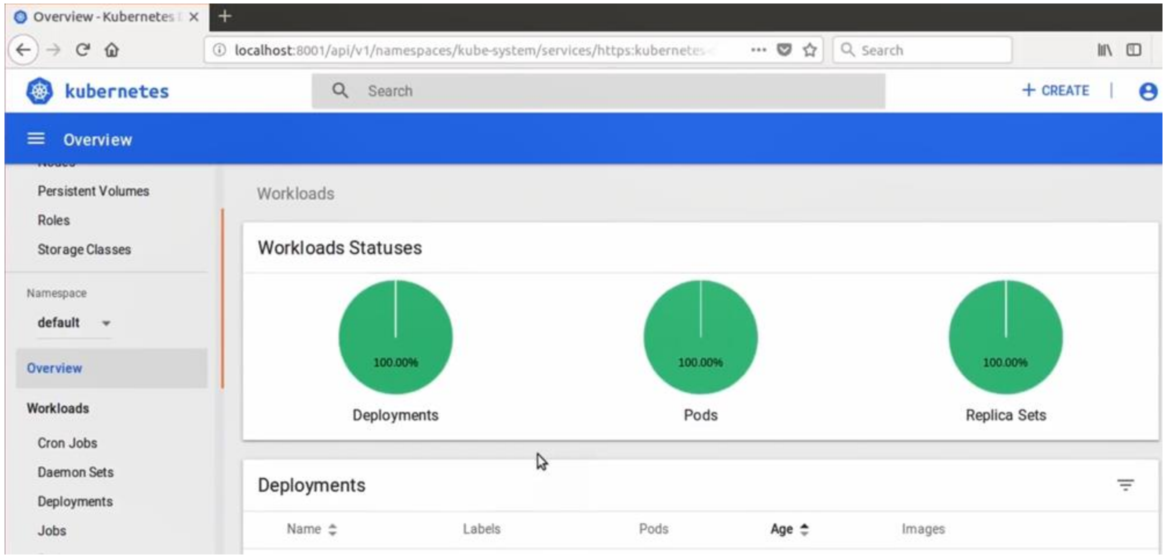
An 'app' label with this value will be added to the Deployment and Service that get deployed. Learn more

Enter the URL of a public image on any registry, or a private image hosted on Docker Hub or Google Container Registry. Learn more

A Deployment will be created to maintain the desired number of pods across your cluster. Learn more

Optionally, an internal or external Service can be defined to map an incoming Port to a target Port seen by the container. The internal DNS name for this Service will be: **edureka-demo**. Learn more

Kubernetes Dashboard: View Work Loads status, Pods status



Overview - Kubernetes

localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes

kubernetes

Search

+ CREATE

Overview

Persistent Volumes
Roles
Storage Classes

Namespace
default

Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs

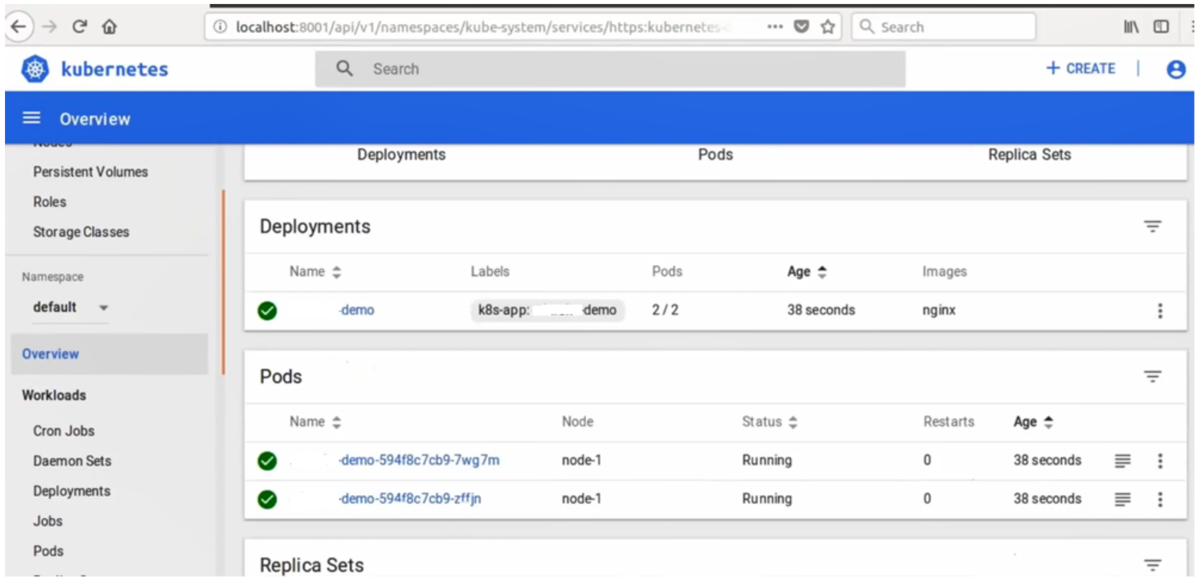
Workloads

Workloads Statuses

Deployments	Pods	Replica Sets
100.00%	100.00%	100.00%

Deployments

Name	Labels	Pods	Age	Images



localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes

kubernetes

Search

+ CREATE

Overview

Persistent Volumes
Roles
Storage Classes

Namespace
default

Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods

Deployments Pods Replica Sets

Deployments

	Name	Labels	Pods	Age	Images
✓	demo	k8s-app: demo	2 / 2	38 seconds	nginx

Pods

	Name	Node	Status	Restarts	Age
✓	demo-594f8c7cb9-7wg7m	node-1	Running	0	38 seconds
✓	demo-594f8c7cb9-zffjn	node-1	Running	0	38 seconds

Replica Sets

Thank you.