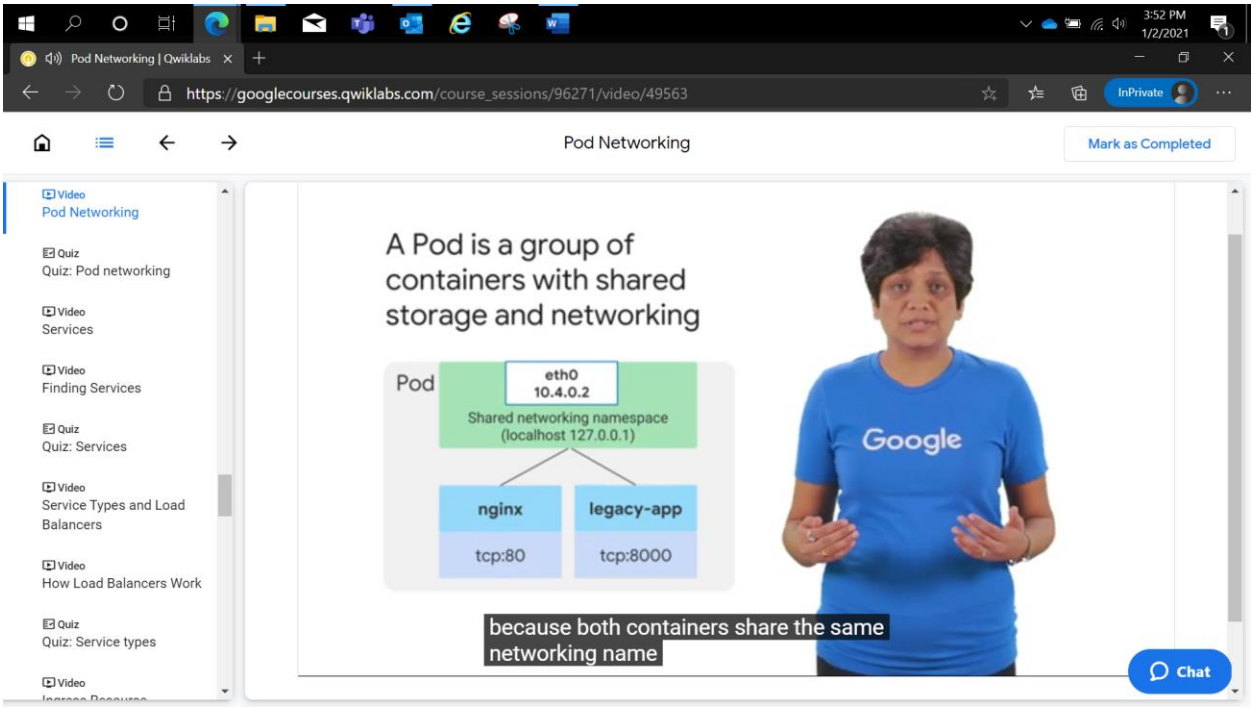
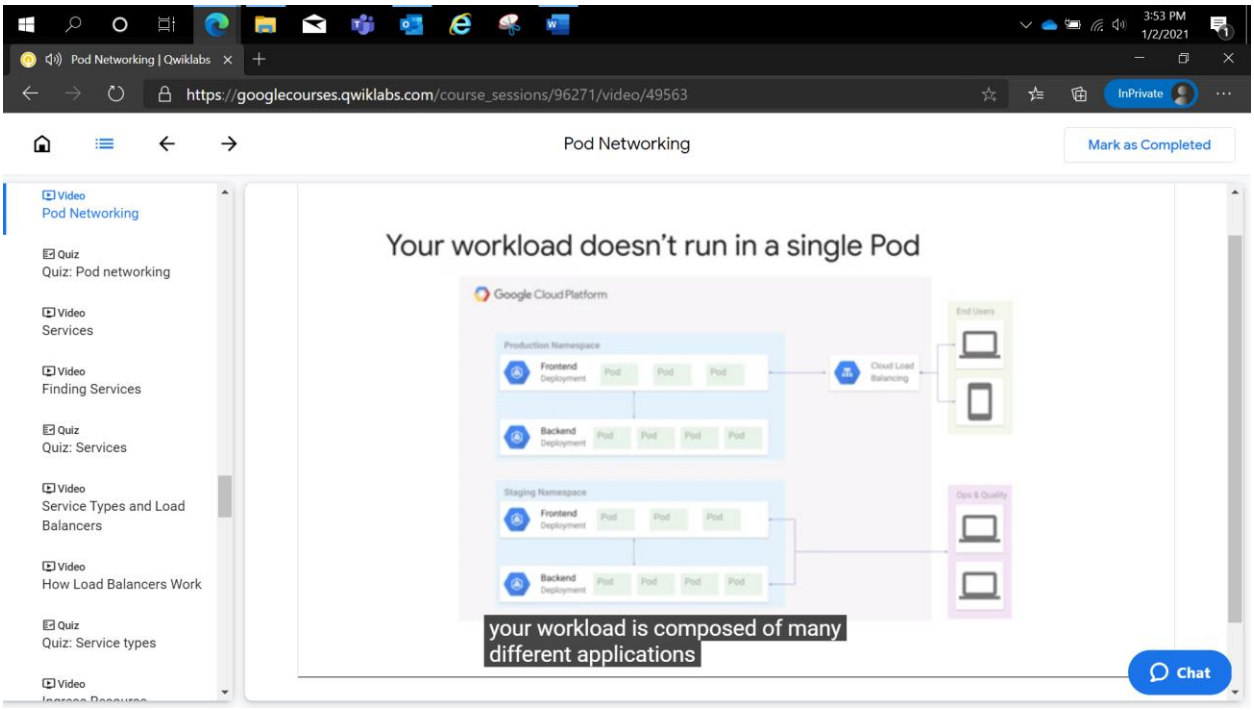
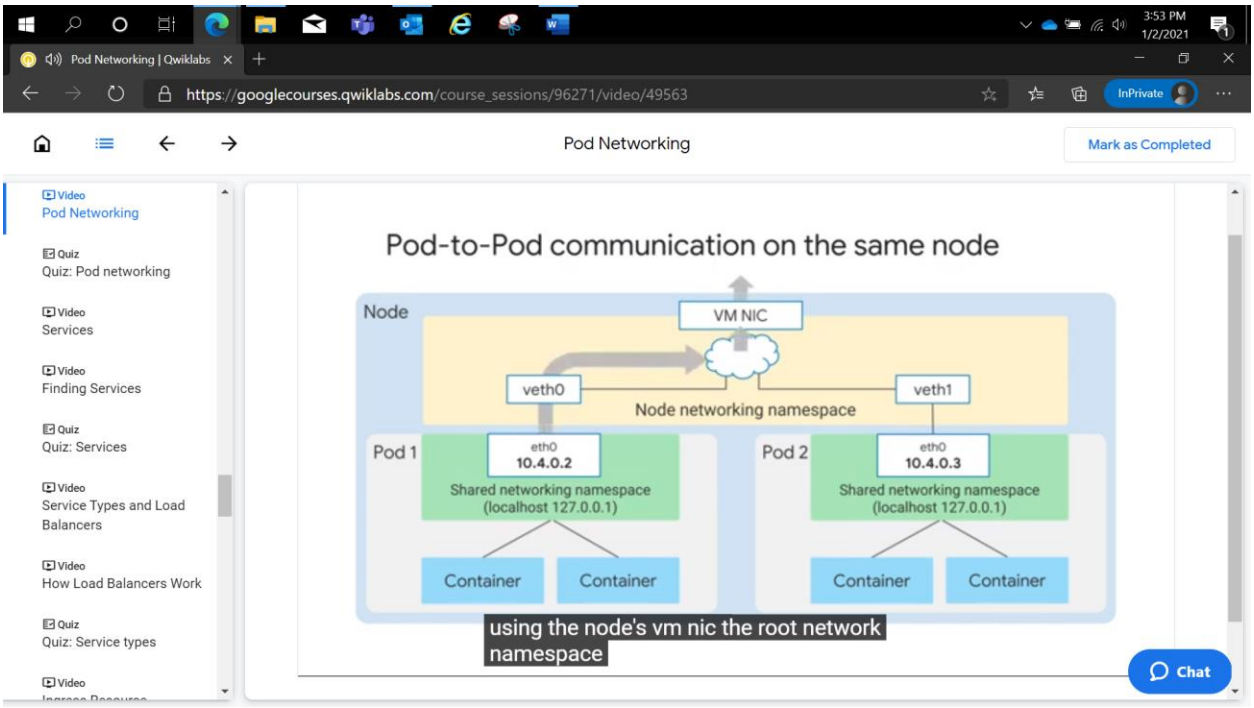
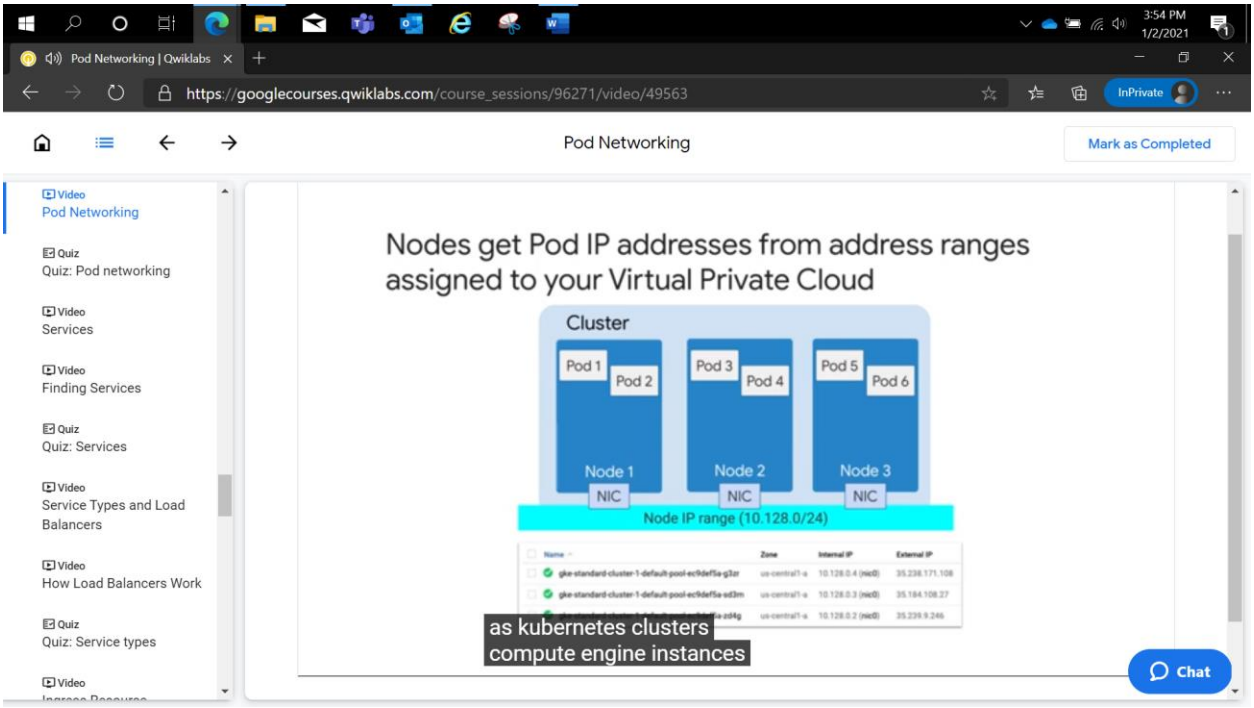
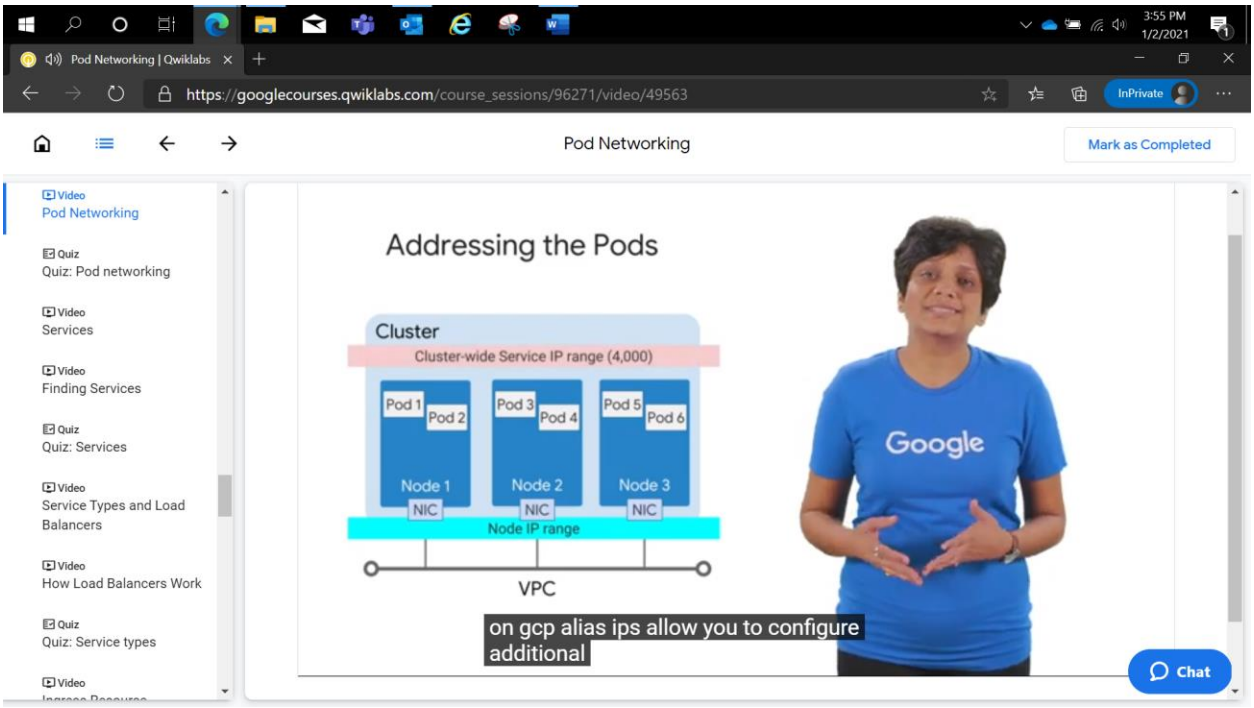
Chapter 2.2.3 GKE networking

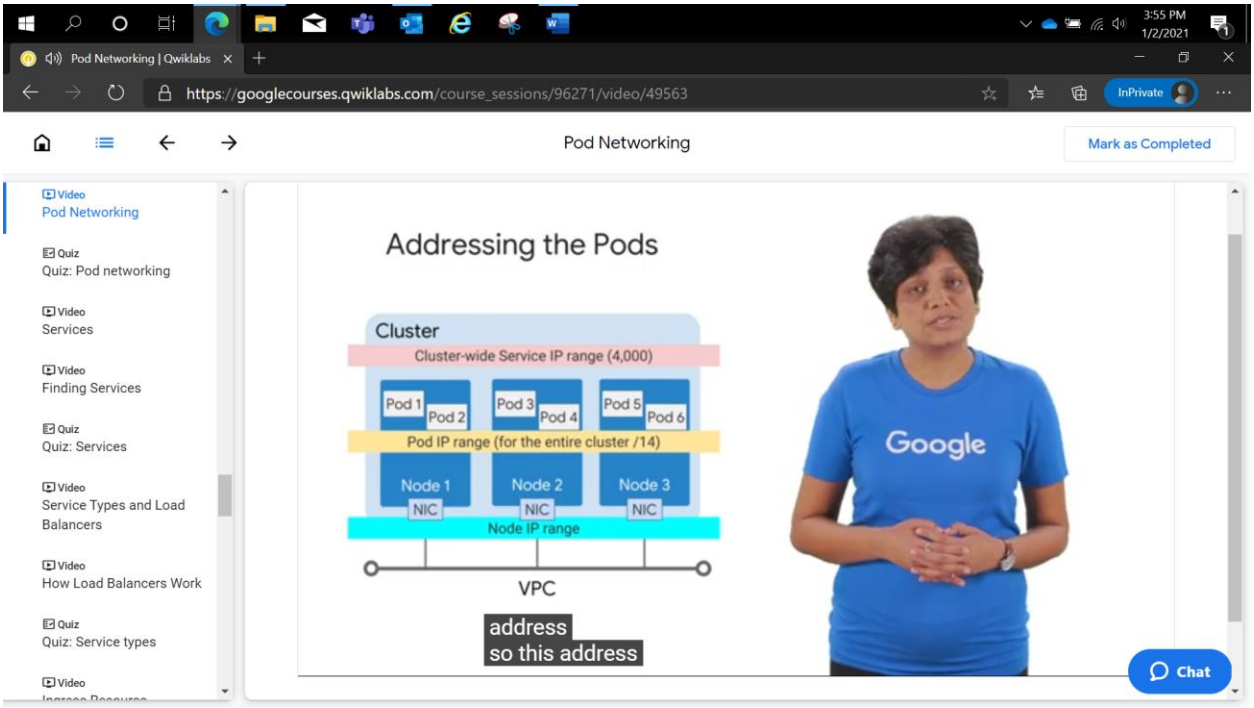


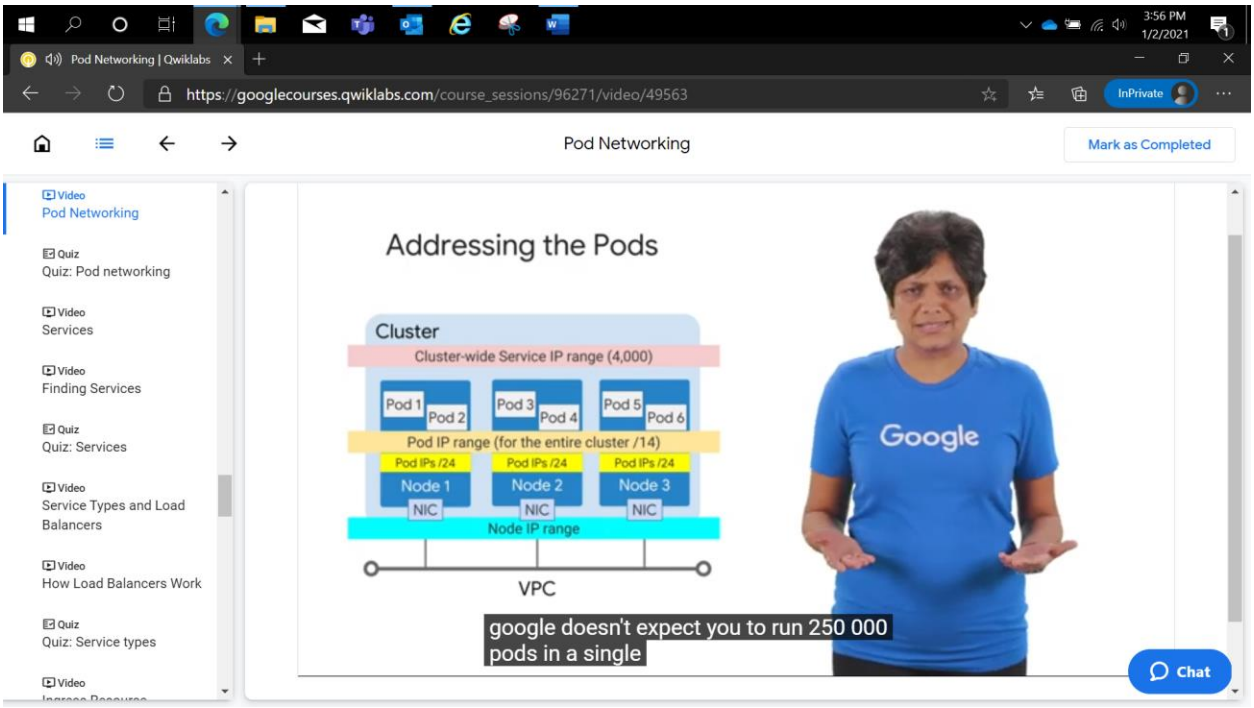


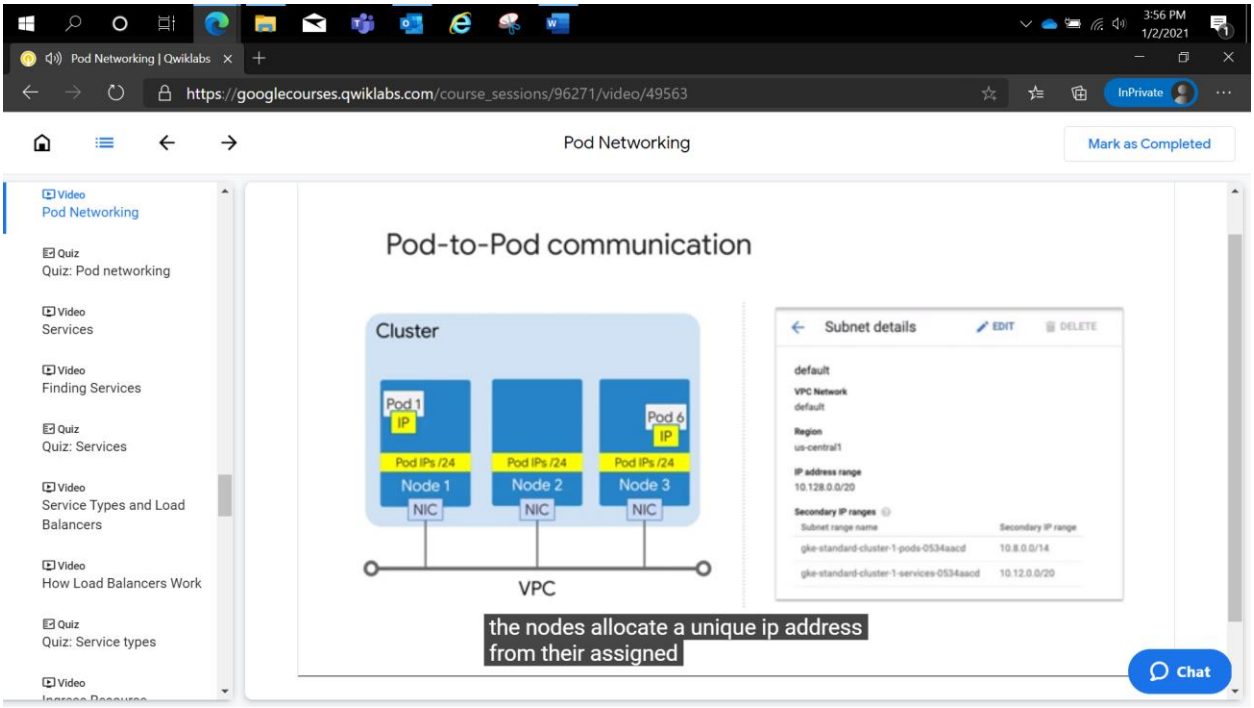

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:53 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Pod-to-Pod communication on the same node
Node
VM NIC
veth0
veth1
Node networking namespace
Pod 1
eth0
10.4.0.2
Shared networking namespace
(localhost 127.0.0.1)
Container
Container
Pod 2
eth0
10.4.0.3
Shared networking namespace
(localhost 127.0.0.1)
Container
Container
using the node's vm nic the root network namespace
Chat

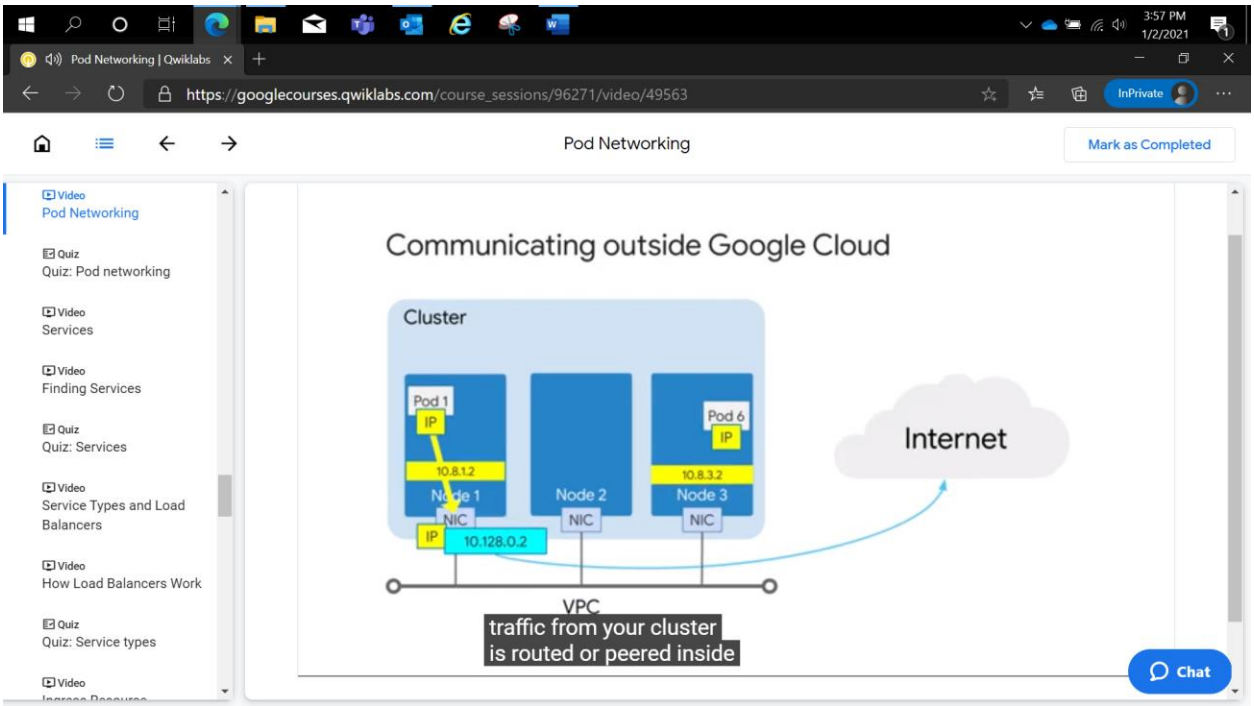

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:54 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Nodes get Pod IP addresses from address ranges assigned to your Virtual Private Cloud
Cluster
Pod 1
Pod 2
Pod 3
Pod 4
Pod 5
Pod 6
Node 1
Node 2
Node 3
NIC
NIC
NIC
Node IP range (10.128.0/24)
as kubernetes clusters compute engine instances
Chat

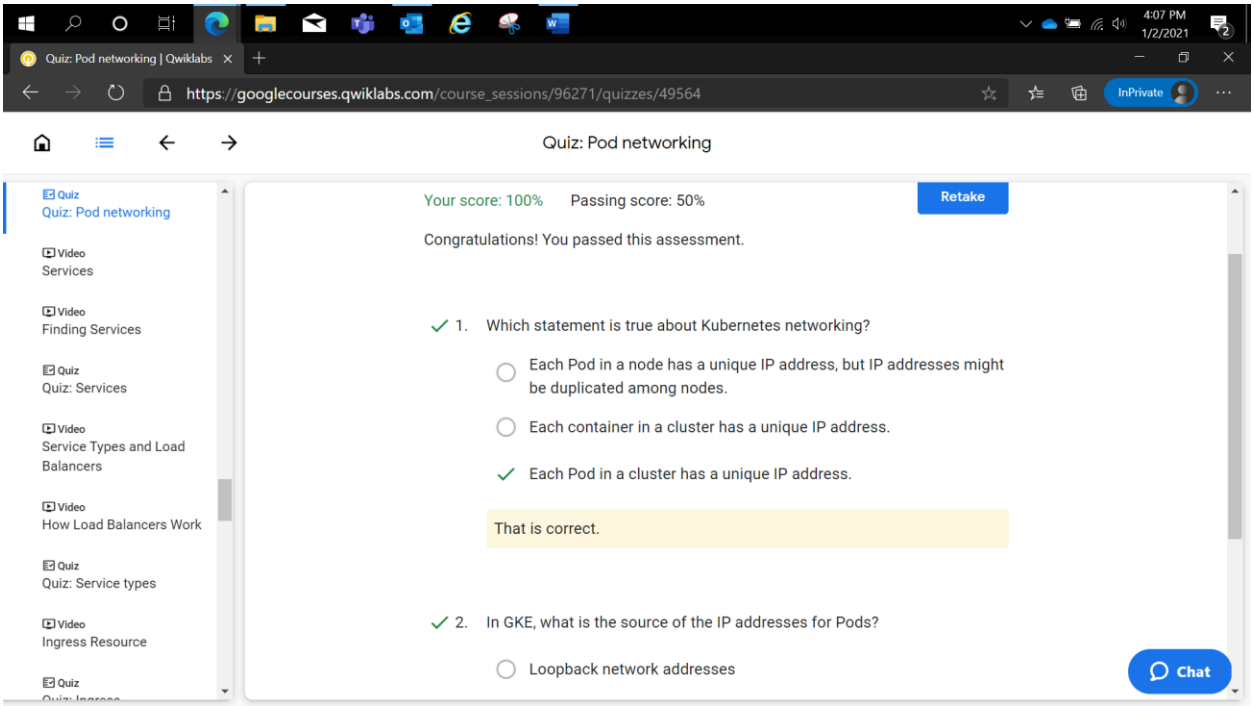

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:55 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Addressing the Pods
Cluster
Cluster-wide Service IP range (4,000)
Pod 1
Pod 2
Pod 3
Pod 4
Pod 5
Pod 6
Node 1
Node 2
Node 3
NIC
Node IP range
VPC
Google
on gcp alias ips allow you to configure
additional
Chat

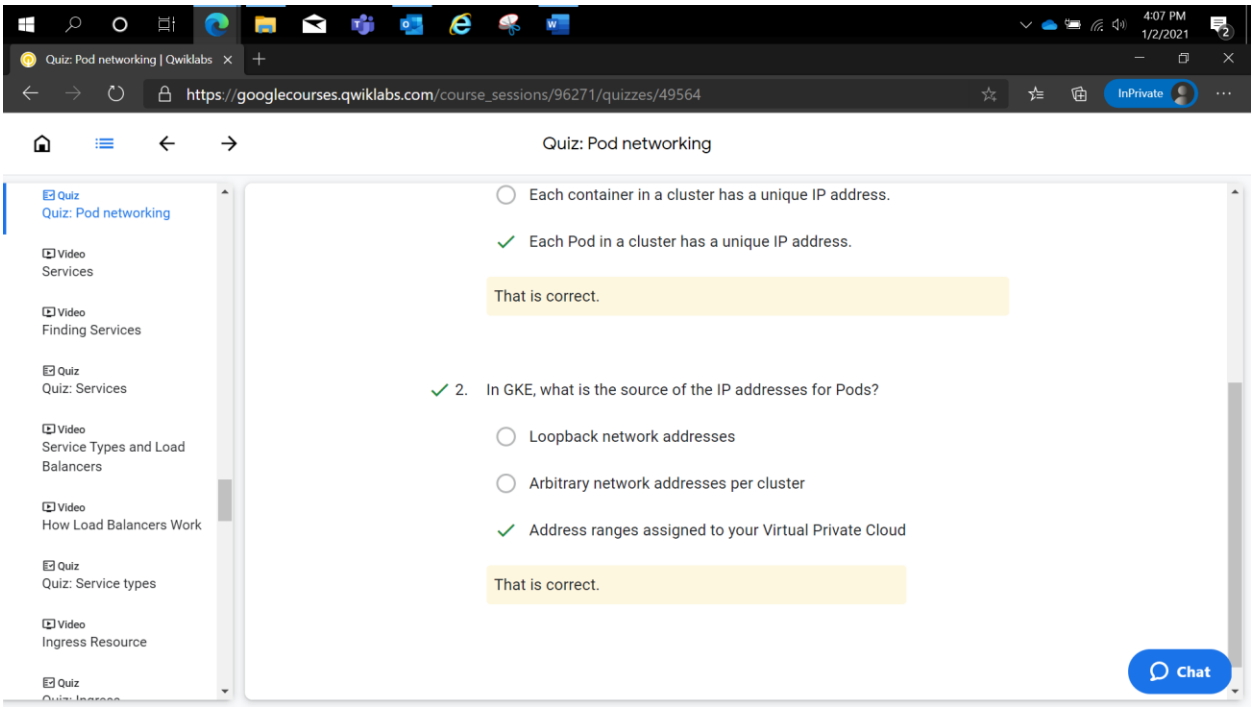

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:55 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Addressing the Pods
Cluster
Cluster-wide Service IP range (4,000)
Pod 1
Pod 2
Pod 3
Pod 4
Pod 5
Pod 6
Pod IP range (for the entire cluster /14)
Node 1
Node 2
Node 3
NIC
Node IP range
VPC
Google
address
so this address
Chat

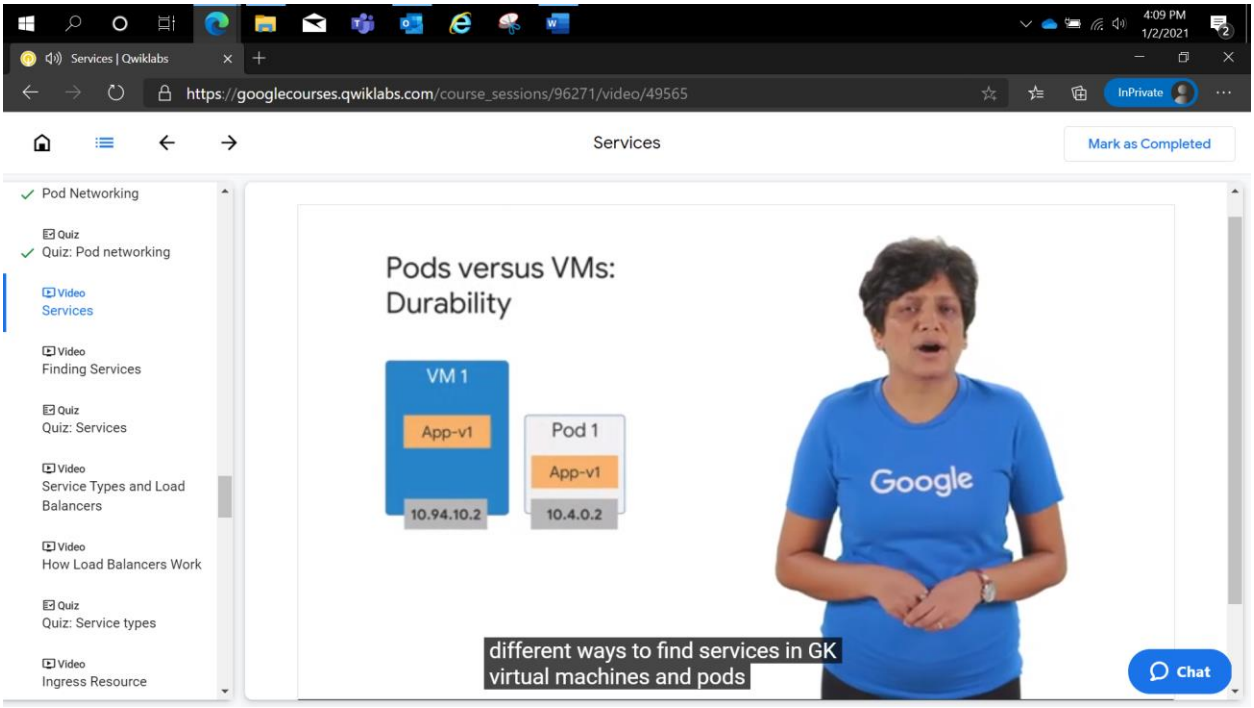

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:56 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Addressing the Pods
Cluster
Cluster-wide Service IP range (4,000)
Pod 1
Pod 2
Pod 3
Pod 4
Pod 5
Pod 6
Pod IP range (for the entire cluster /14)
Pod IPs /24
Pod IPs /24
Pod IPs /24
Node 1
Node 2
Node 3
NIC
NIC
NIC
Node IP range
VPC
Google
google doesn't expect you to run 250 000 pods in a single
Chat

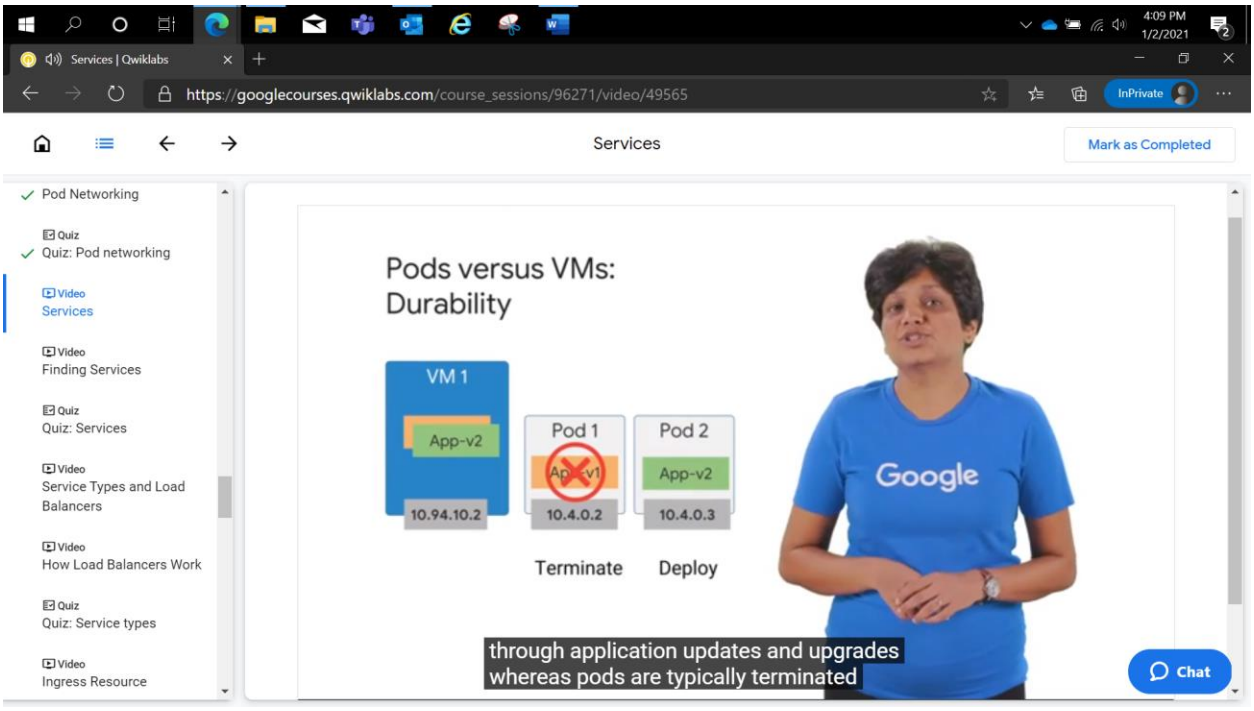

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:56 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Pod-to-Pod communication
Cluster
Pod 1
IP
Pod 6
IP
Pod IPs /24
Pod IPs /24
Pod IPs /24
Node 1
Node 2
Node 3
NIC
NIC
NIC
VPC
Subnet details
EDIT
DELETE
default
VPC Network
default
Region
us-central1
IP address range
10.128.0.0/20
Secondary IP ranges
Subnet range name
Secondary IP range
gke-standard-cluster-1-pods-0534aacd
10.8.0.0/14
gke-standard-cluster-1-services-0534aacd
10.12.0.0/20
the nodes allocate a unique ip address from their assigned
Chat

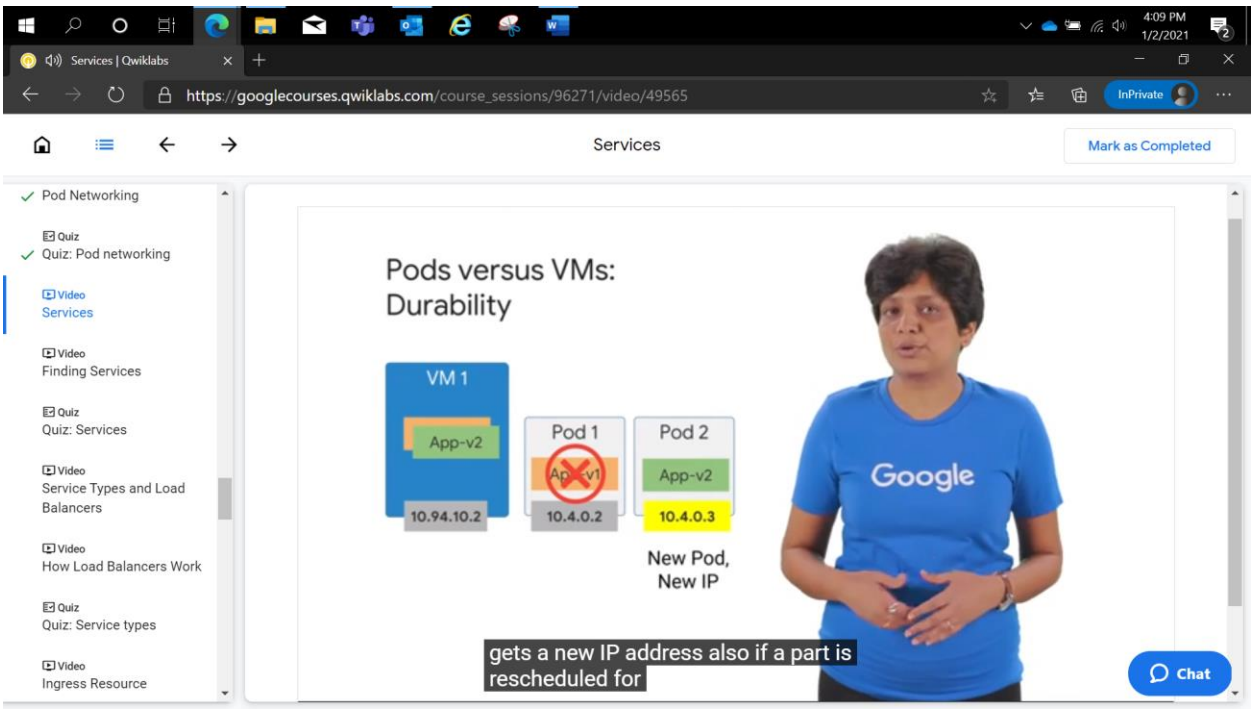

Pod Networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49563
InPrivate
3:57 PM
1/2/2021
Pod Networking
Mark as Completed
Video
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Communicating outside Google Cloud
Cluster
Pod 1
IP
10.8.1.2
Node 1
NIC
IP
10.128.0.2
Node 2
NIC
Pod 6
IP
10.8.3.2
Node 3
NIC
Internet
VPC
traffic from your cluster
is routed or peered inside
Chat

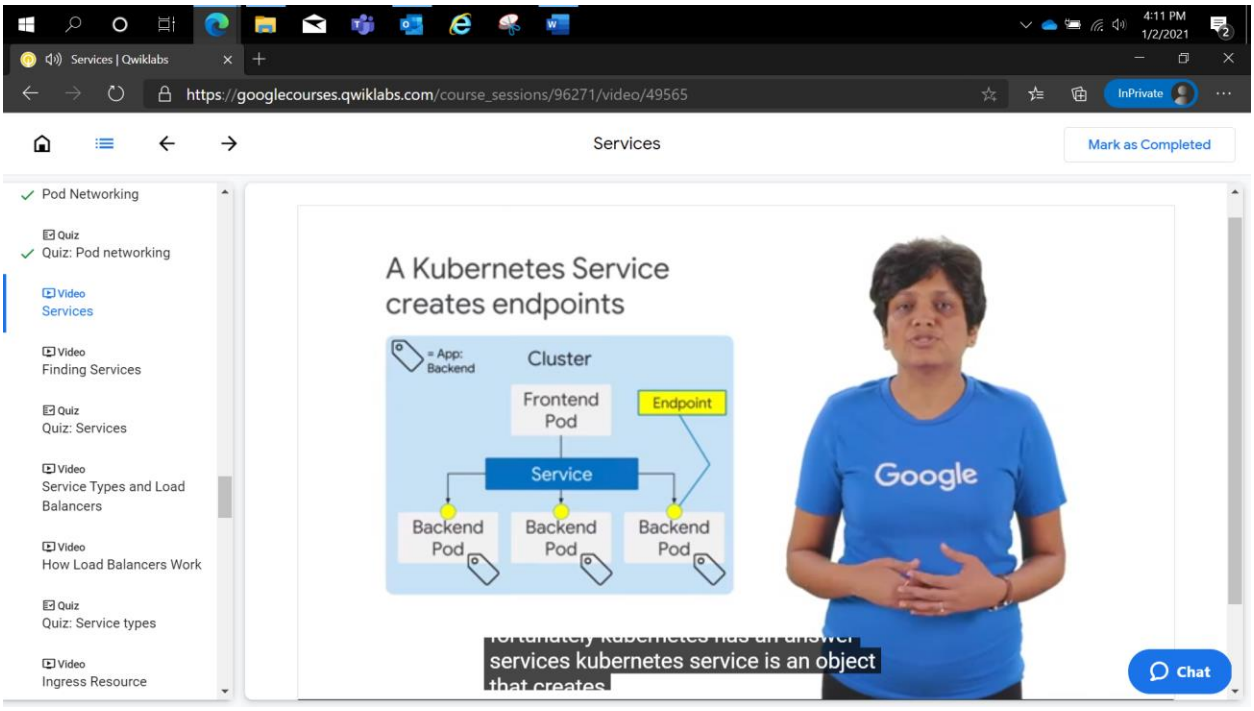

Quiz: Pod networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49564
InPrivate
4:07 PM
1/2/2021
Quiz: Pod networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Your score: 100%
Passing score: 50%
Retake
Congratulations! You passed this assessment.
1. Which statement is true about Kubernetes networking?
Each Pod in a node has a unique IP address, but IP addresses might be duplicated among nodes.
Each container in a cluster has a unique IP address.
Each Pod in a cluster has a unique IP address.
That is correct.
2. In GKE, what is the source of the IP addresses for Pods?
Loopback network addresses
Chat

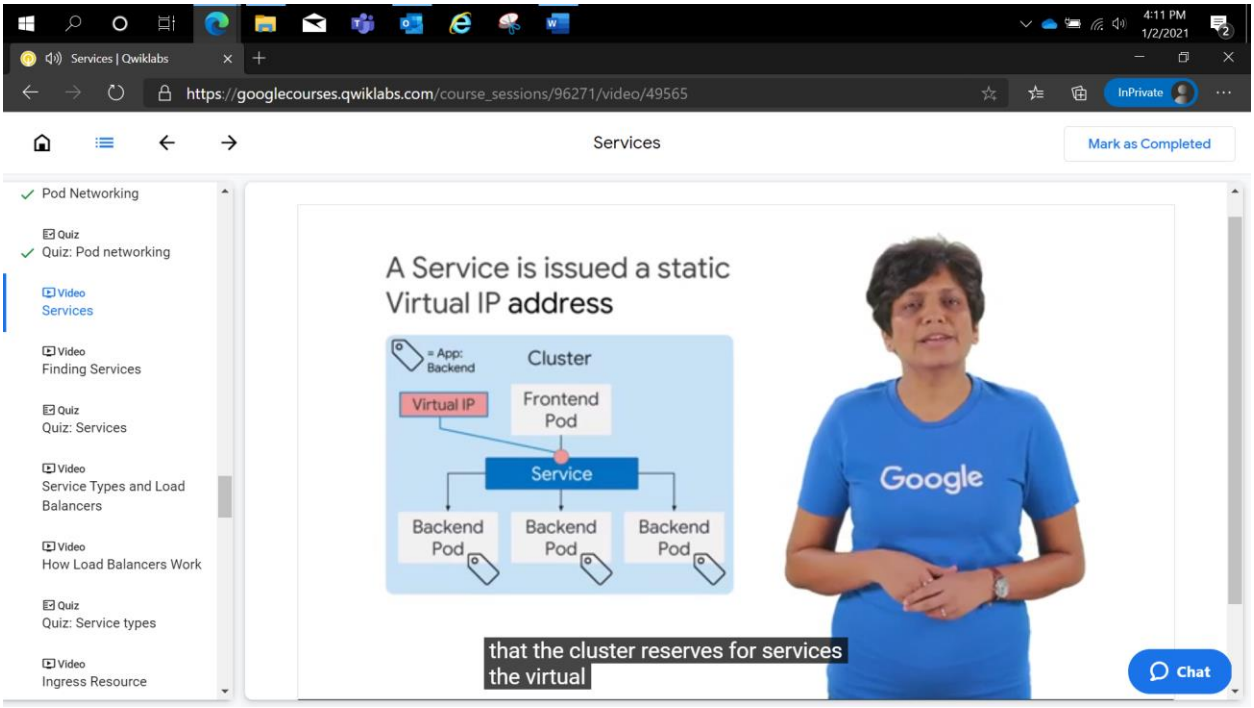

Quiz: Pod networking | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49564
InPrivate
4:07 PM
1/2/2021
Quiz: Pod networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Each container in a cluster has a unique IP address.
Each Pod in a cluster has a unique IP address.
That is correct.
2. In GKE, what is the source of the IP addresses for Pods?
Loopback network addresses
Arbitrary network addresses per cluster
Address ranges assigned to your Virtual Private Cloud
That is correct.
Chat

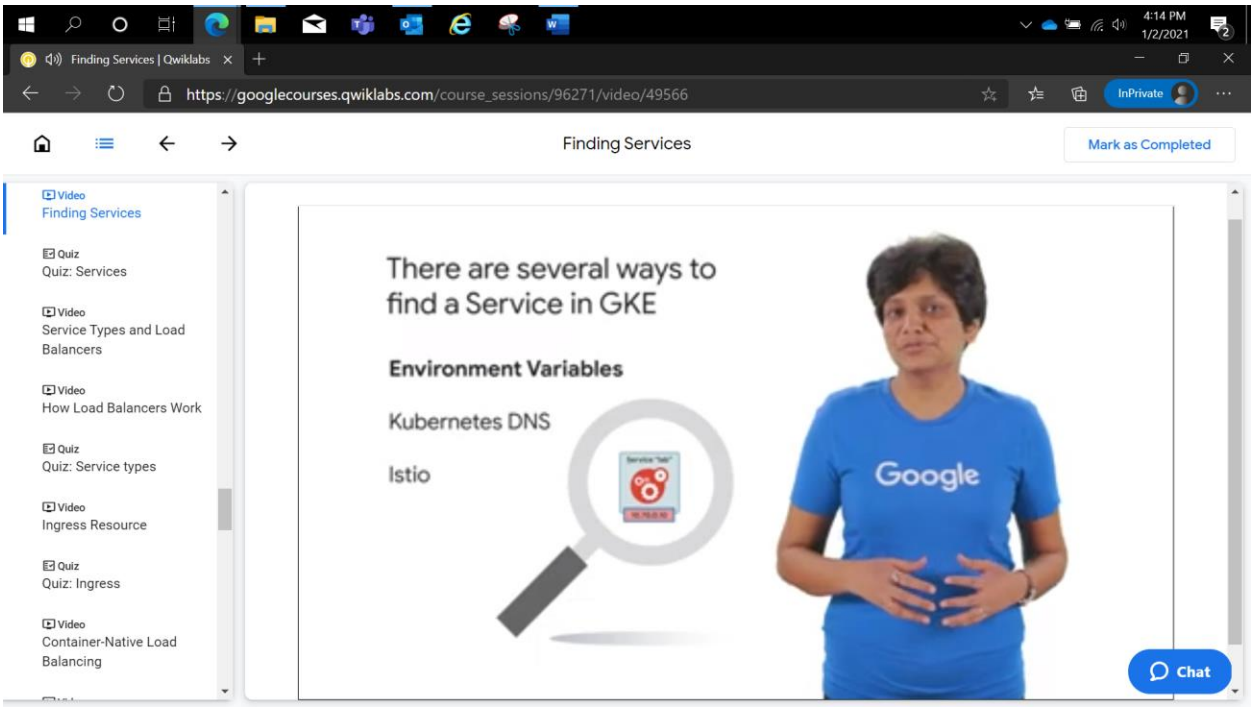

Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49565
InPrivate
4:09 PM
1/2/2021
Services
Mark as Completed
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Pods versus VMs: Durability
VM 1
App-v1
10.94.10.2
Pod 1
App-v1
10.4.0.2
Google
different ways to find services in GK
virtual machines and pods
Chat

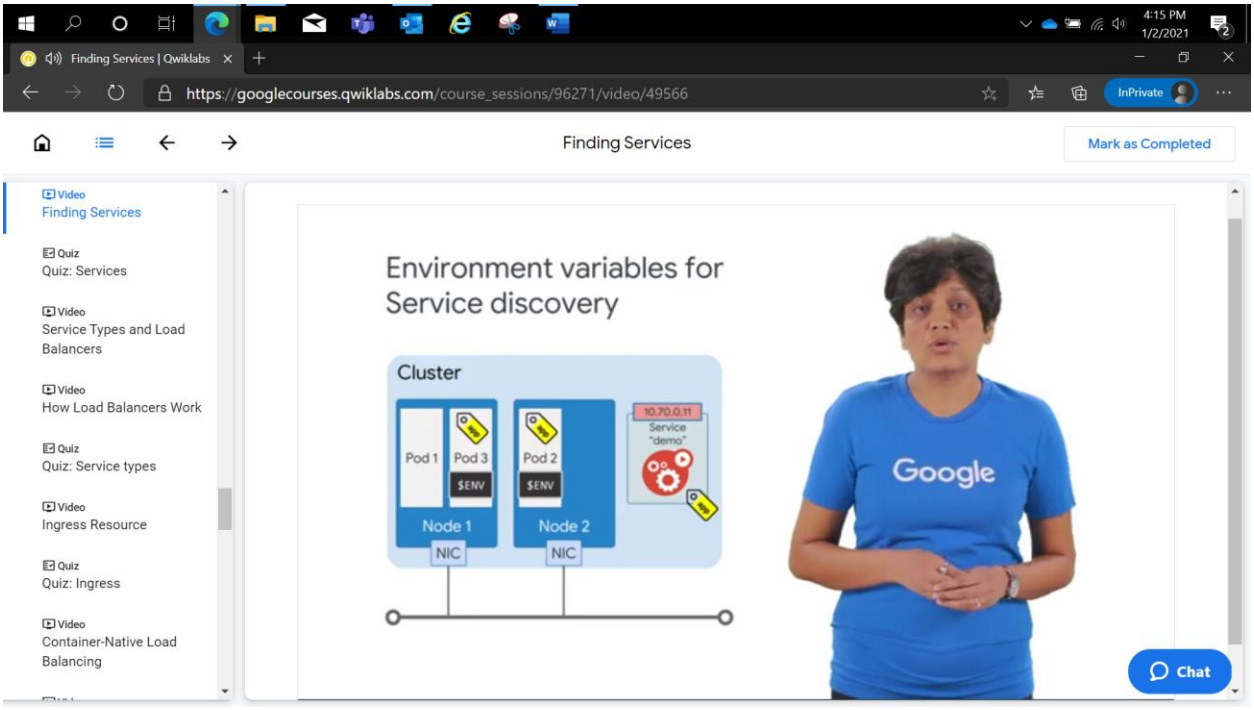

Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49565
InPrivate
4:09 PM
1/2/2021
Services
Mark as Completed
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Pods versus VMs:
Durability
VM 1
App-v2
10.94.10.2
Pod 1
10.4.0.2
Pod 2
App-v2
10.4.0.3
Terminate
Deploy
Google
through application updates and upgrades
whereas pods are typically terminated
Chat

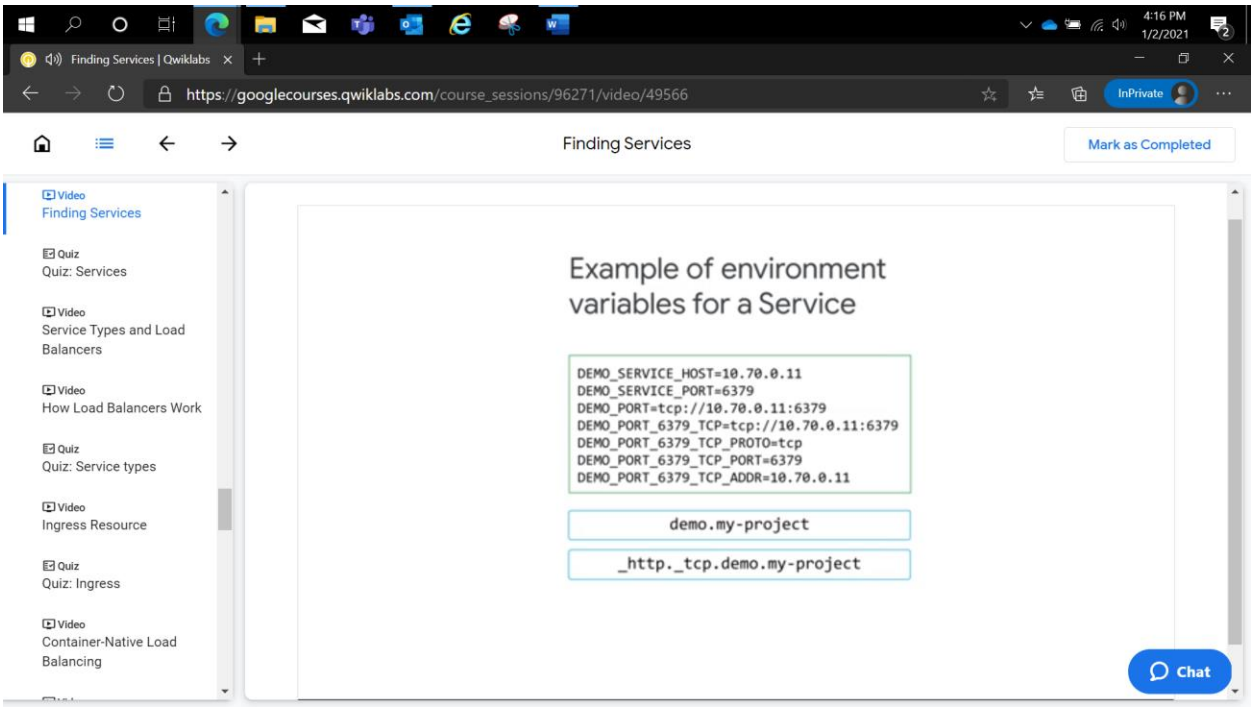

Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49565
InPrivate
4:09 PM
1/2/2021
Services
Mark as Completed
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Pods versus VMs:
Durability
VM 1
App-v2
10.94.10.2
Pod 1
10.4.0.2
Pod 2
App-v2
10.4.0.3
New Pod,
New IP
Google
gets a new IP address also if a part is
rescheduled for
Chat

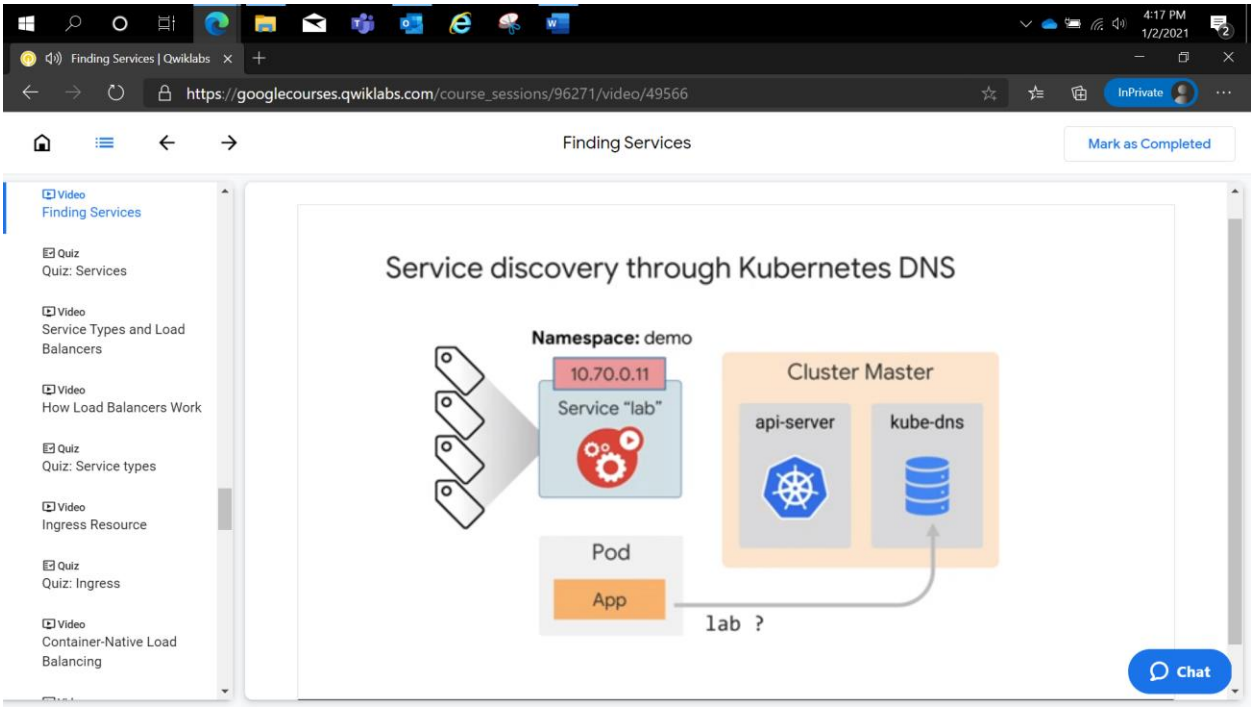

Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49565
InPrivate
4:11 PM
1/2/2021
Services
Mark as Completed
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
A Kubernetes Service creates endpoints
= App: Backend
Cluster
Frontend Pod
Endpoint
Service
Backend Pod
Backend Pod
Backend Pod
Google
services kubernetes service is an object
Chat

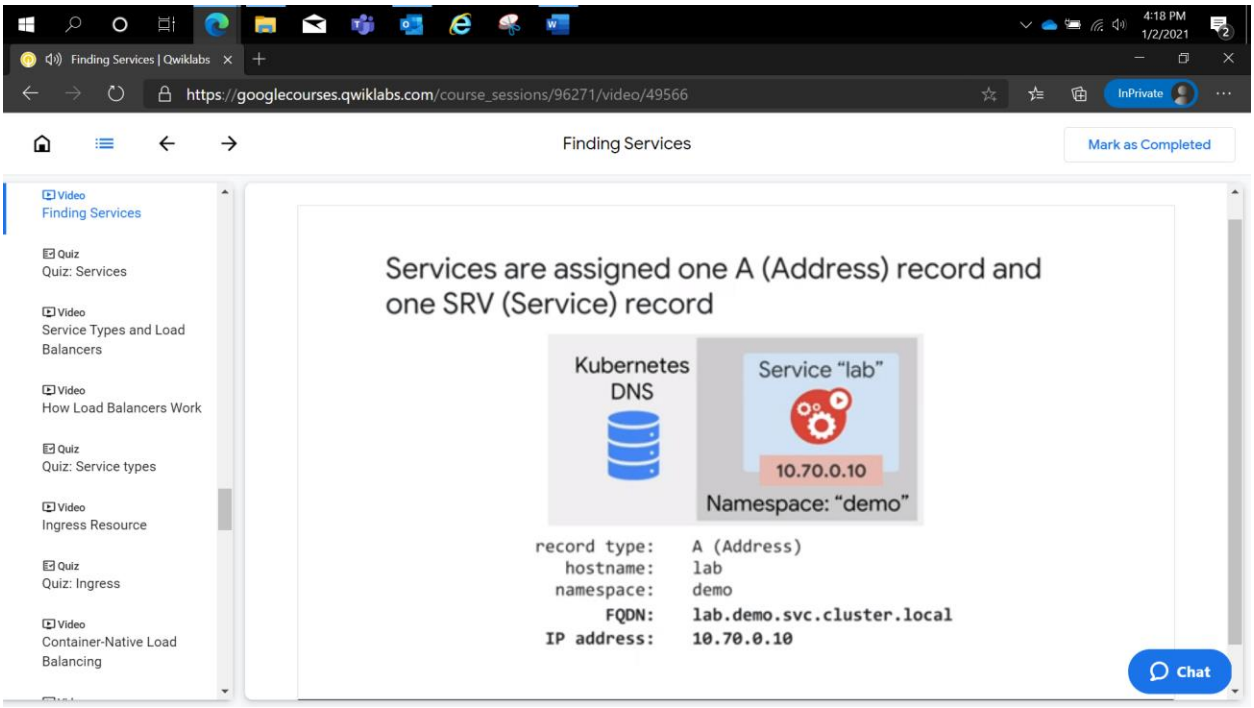

Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49565
InPrivate
4:11 PM
1/2/2021
Services
Mark as Completed
Pod Networking
Quiz
Quiz: Pod networking
Video
Services
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
A Service is issued a static Virtual IP address
= App: Backend
Cluster
Virtual IP
Frontend Pod
Service
Backend Pod
Backend Pod
Backend Pod
Google
that the cluster reserves for services
the virtual
Chat

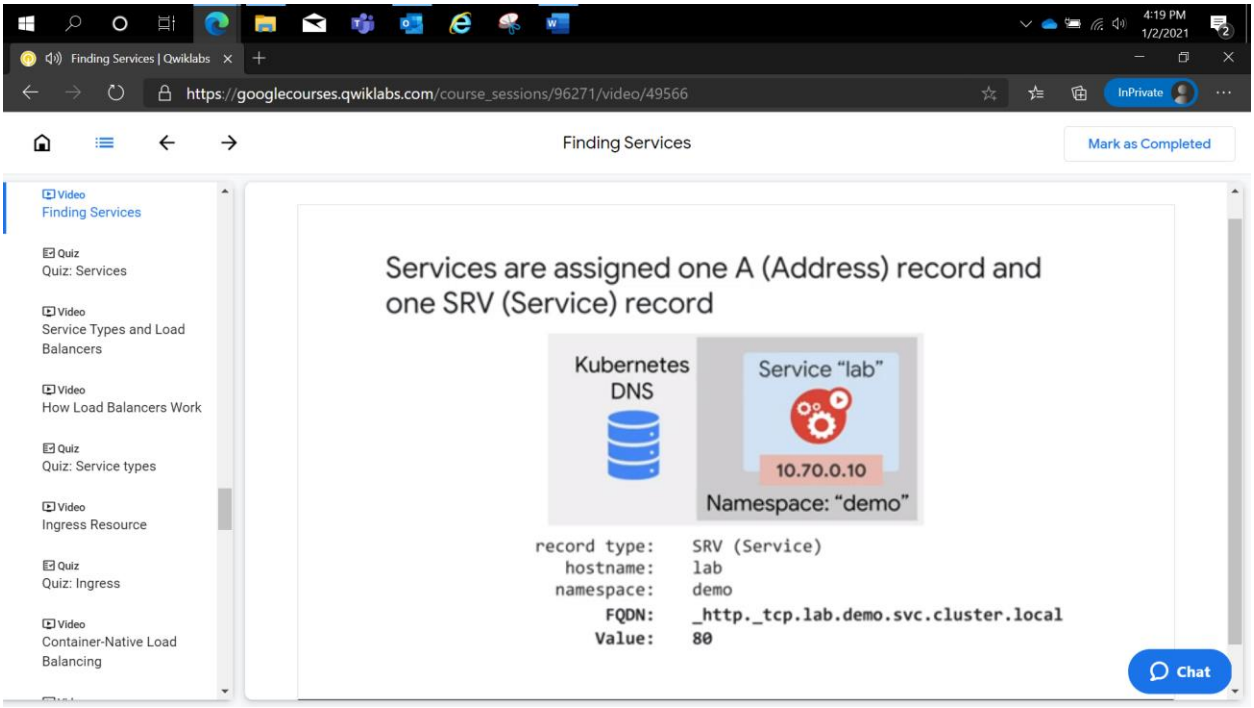

Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
4:14 PM
1/2/2021
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
There are several ways to find a Service in GKE
Environment Variables
Kubernetes DNS
Istio
Google
Chat

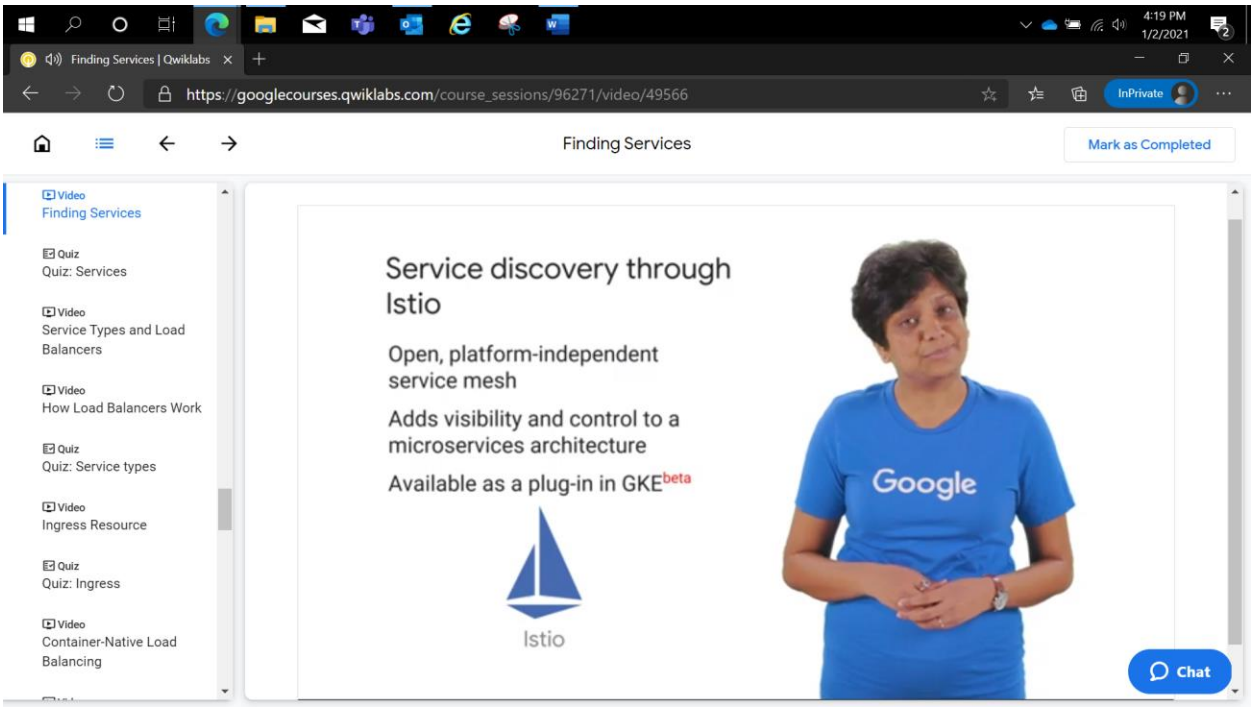

Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
4:15 PM
1/2/2021
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Environment variables for Service discovery
Cluster
Pod 1
Pod 3
$ENV
Pod 2
$ENV
Node 1
Node 2
NIC
NIC
10.70.0.11
Service "demo"
Google
Chat

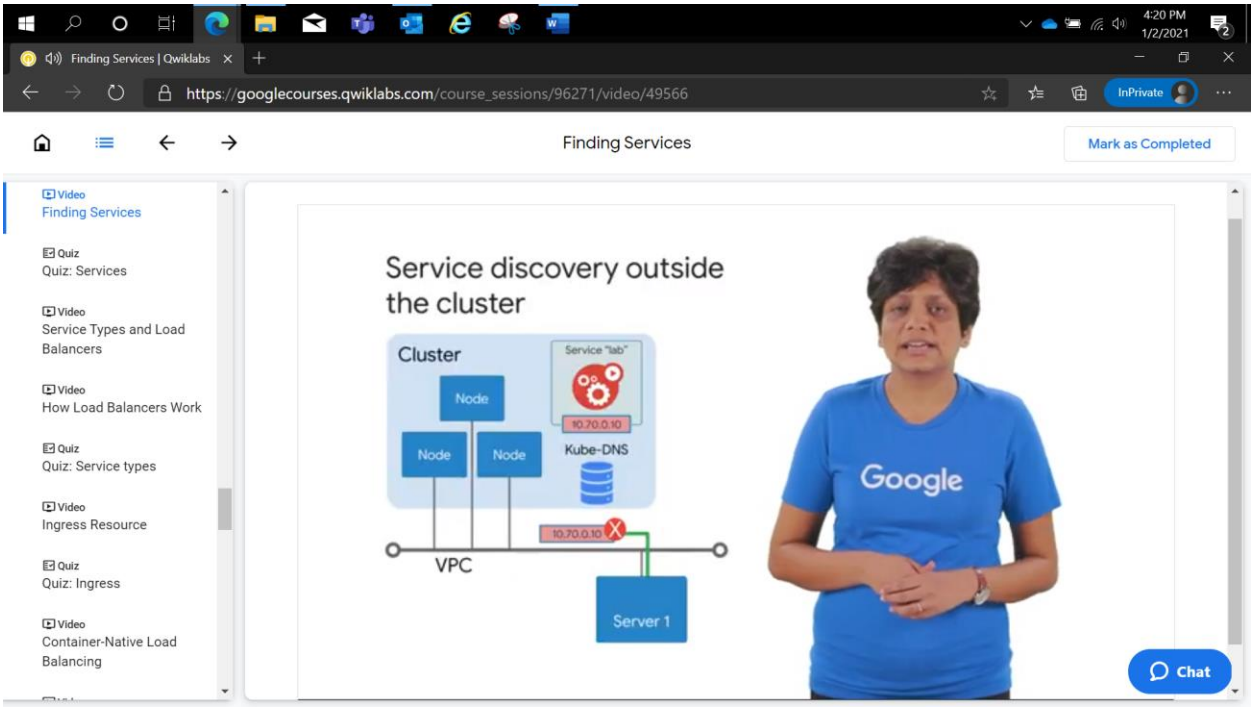

Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
4:16 PM
1/2/2021
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Example of environment variables for a Service
DEMO_SERVICE_HOST=10.70.0.11
DEMO_SERVICE_PORT=6379
DEMO_PORT=tcp://10.70.0.11:6379
DEMO_PORT_6379_TCP=tcp://10.70.0.11:6379
DEMO_PORT_6379_TCP_PROTO=tcp
DEMO_PORT_6379_TCP_PORT=6379
DEMO_PORT_6379_TCP_ADDR=10.70.0.11
demo.my-project
_http._tcp.demo.my-project
Chat

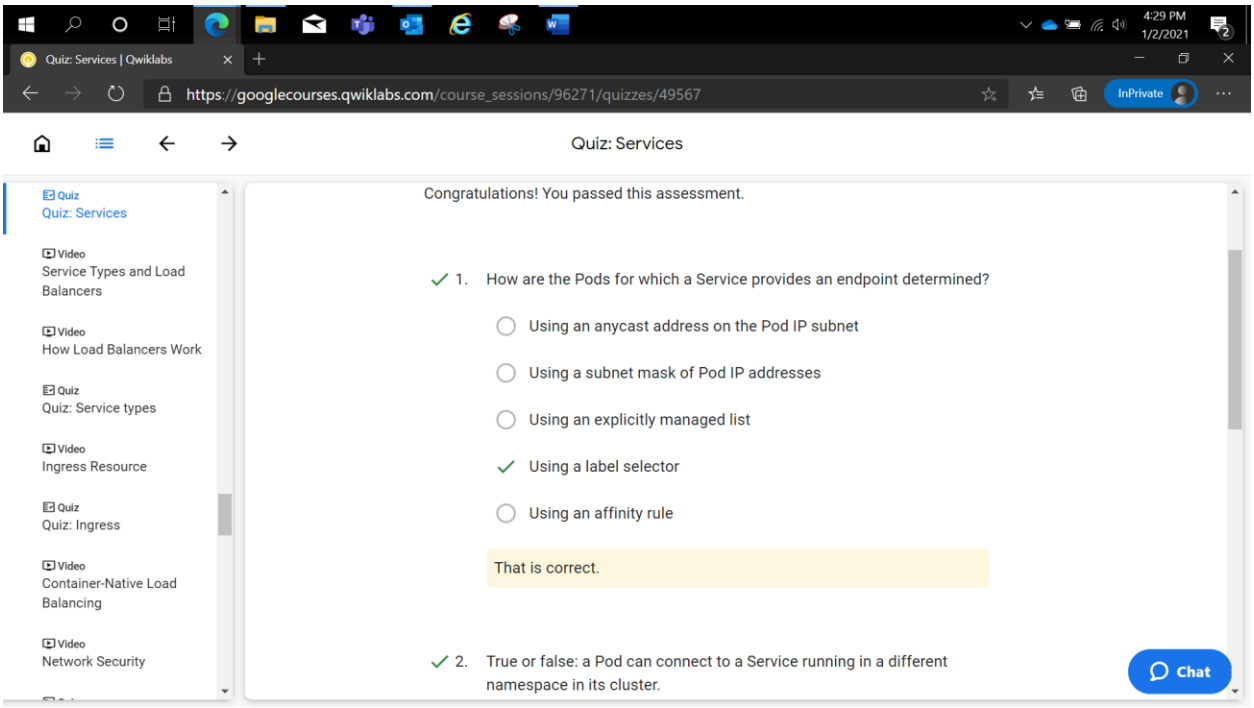

Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
4:17 PM
1/2/2021
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Service discovery through Kubernetes DNS
Namespace: demo
10.70.0.11
Service "lab"
Cluster Master
api-server
kube-dns
Pod
App
lab ?
Chat

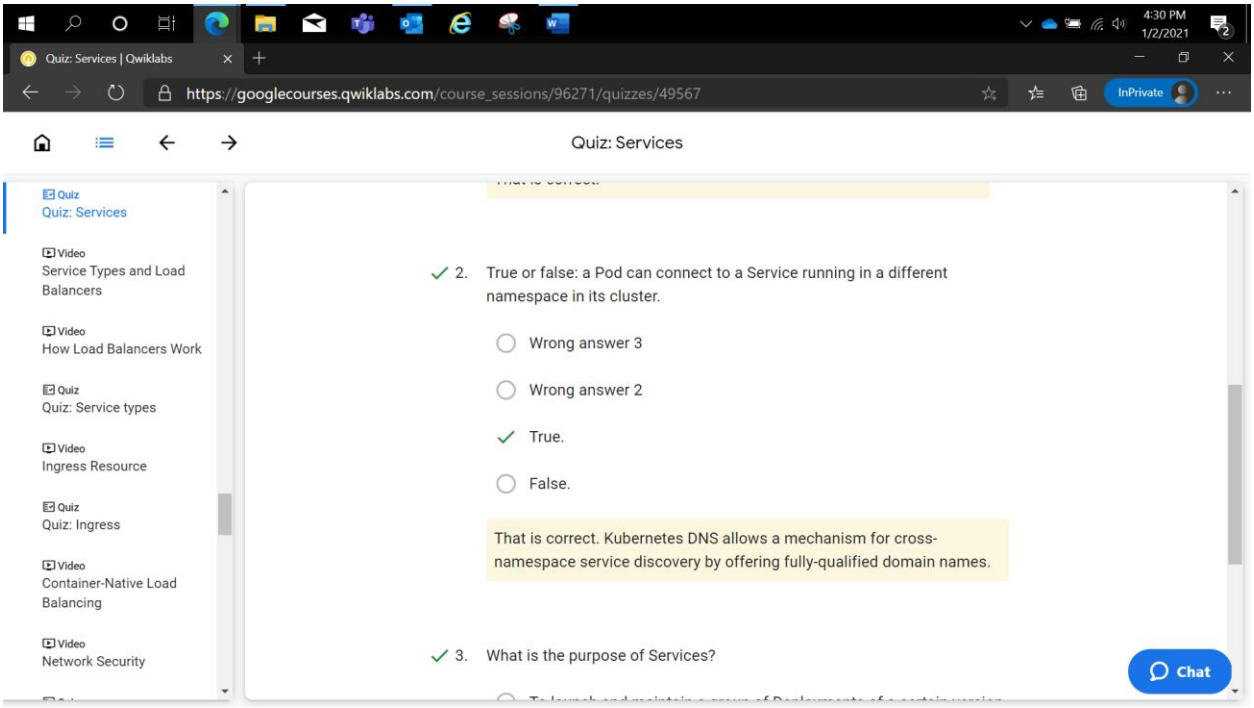

4:18 PM
1/2/2021
Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Services are assigned one A (Address) record and one SRV (Service) record
Kubernetes DNS
Service "lab"
10.70.0.10
Namespace: "demo"
record type: A (Address)
hostname: lab
namespace: demo
FQDN: lab.demo.svc.cluster.local
IP address: 10.70.0.10
Chat

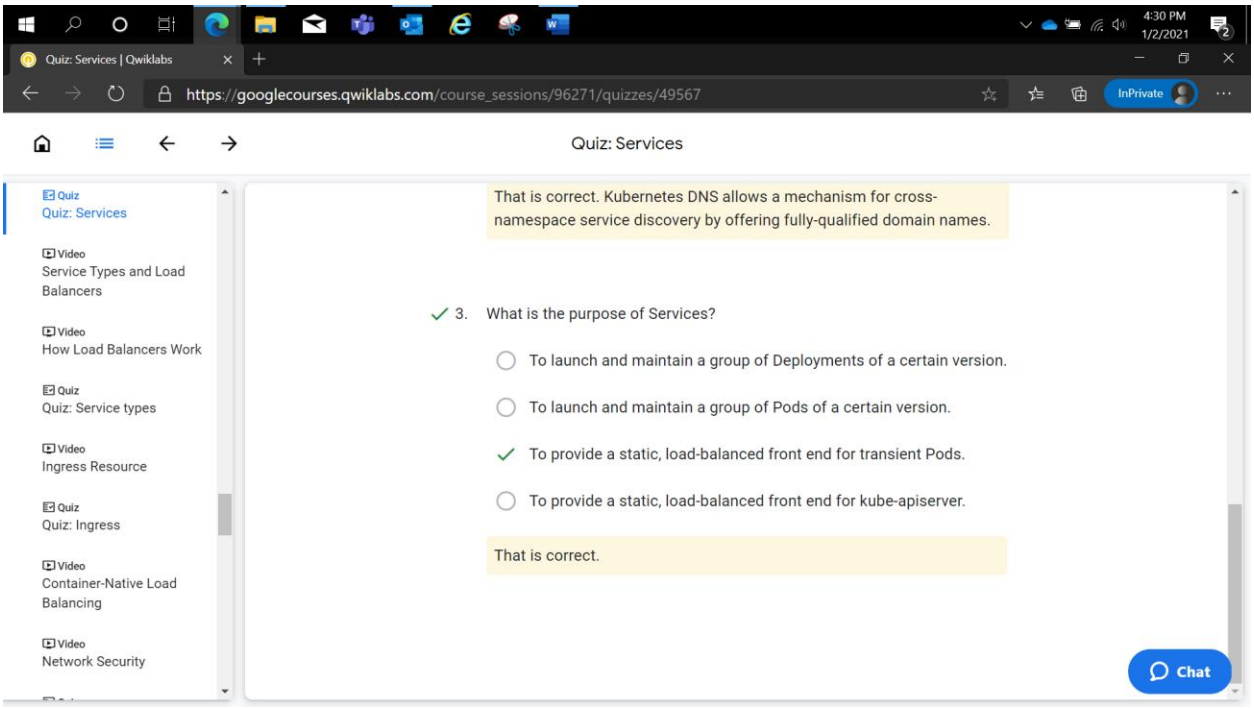

4:19 PM
1/2/2021
Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Services are assigned one A (Address) record and one SRV (Service) record
Kubernetes DNS
Service "lab"
10.70.0.10
Namespace: "demo"
record type: SRV (Service)
hostname: lab
namespace: demo
FQDN: _http._tcp.lab.demo.svc.cluster.local
Value: 80
Chat

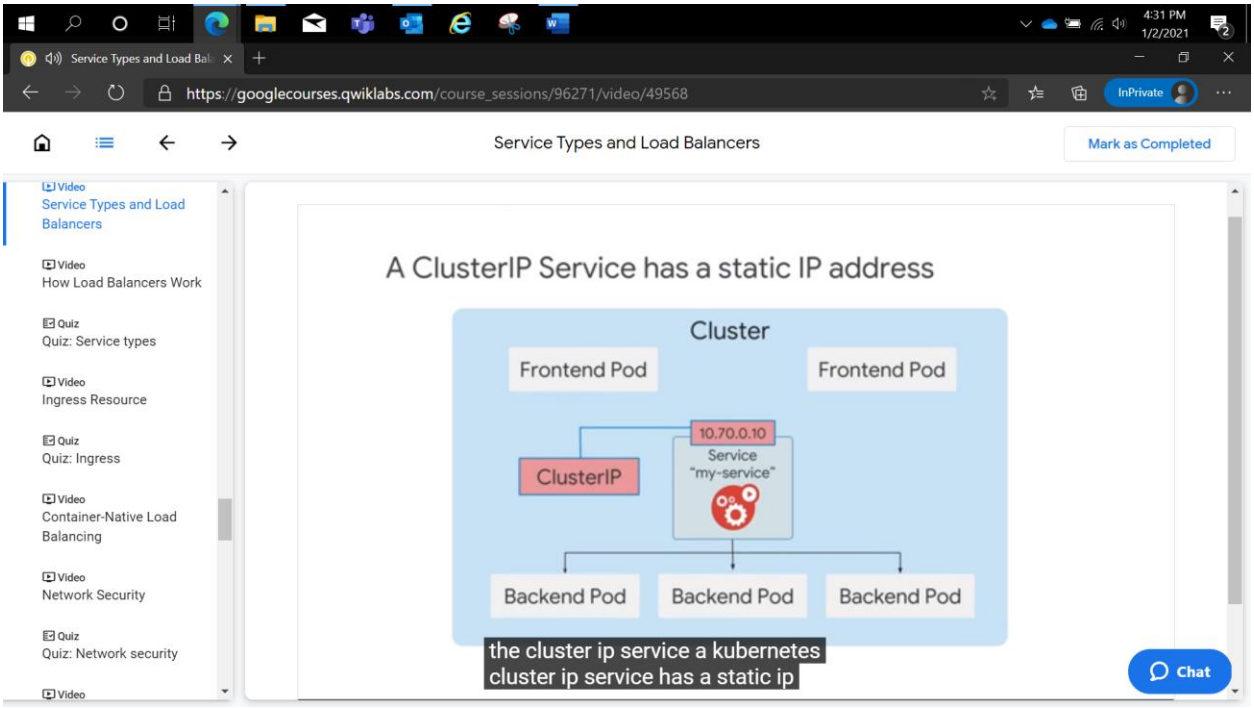

4:19 PM
1/2/2021
Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Service discovery through Istio
Open, platform-independent service mesh
Adds visibility and control to a microservices architecture
Available as a plug-in in GKE beta
Istio
Google
Chat

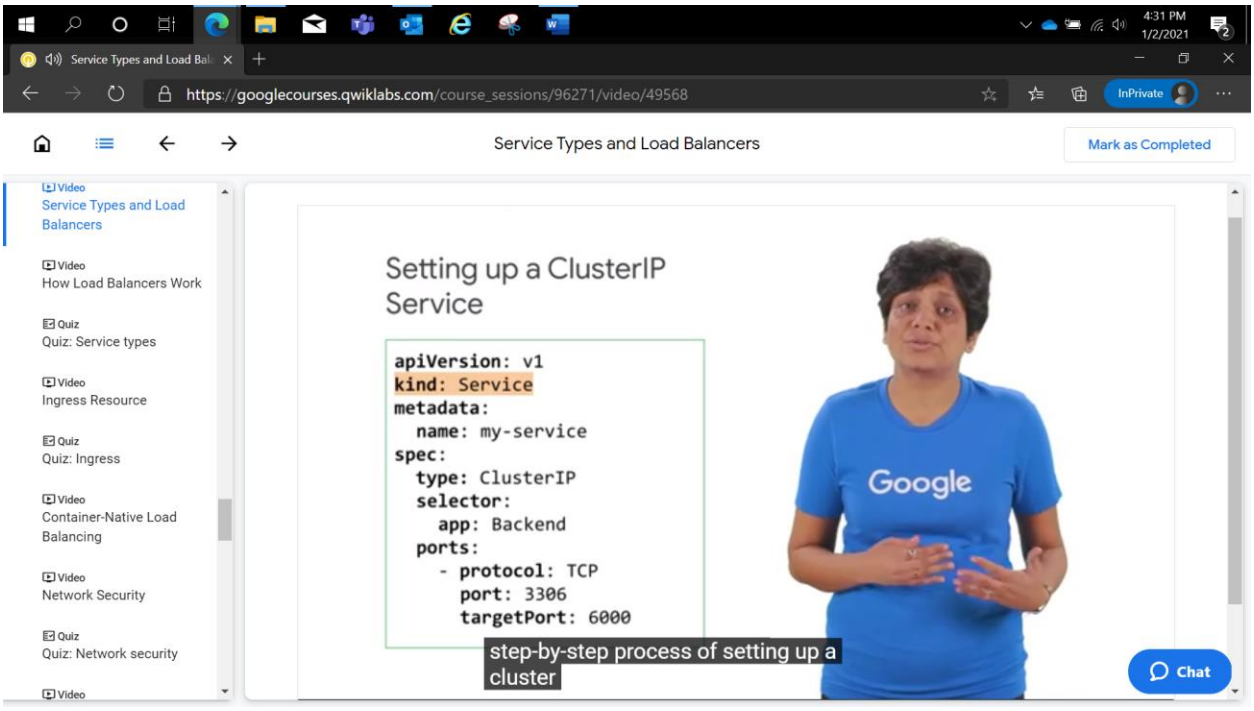

4:20 PM
1/2/2021
Finding Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49566
InPrivate
Finding Services
Mark as Completed
Video
Finding Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Service discovery outside the cluster
Cluster
Service "lab"
Node
Node
Node
10.70.0.10
Kube-DNS
10.70.0.10
VPC
Server 1
Google
Chat

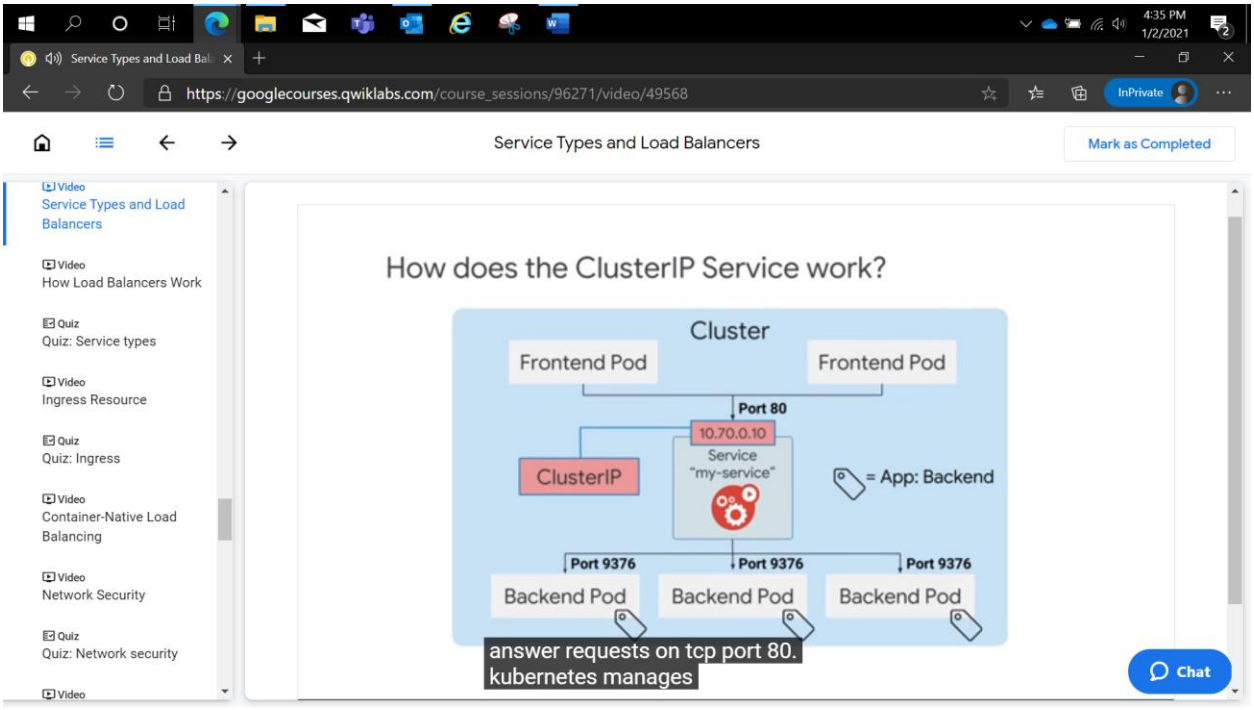

4:29 PM
1/2/2021
Quiz: Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49567
InPrivate
Quiz: Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Congratulations! You passed this assessment.
✓ 1. How are the Pods for which a Service provides an endpoint determined?
Using an anycast address on the Pod IP subnet
Using a subnet mask of Pod IP addresses
Using an explicitly managed list
✓ Using a label selector
Using an affinity rule
That is correct.
✓ 2. True or false: a Pod can connect to a Service running in a different namespace in its cluster.
Chat

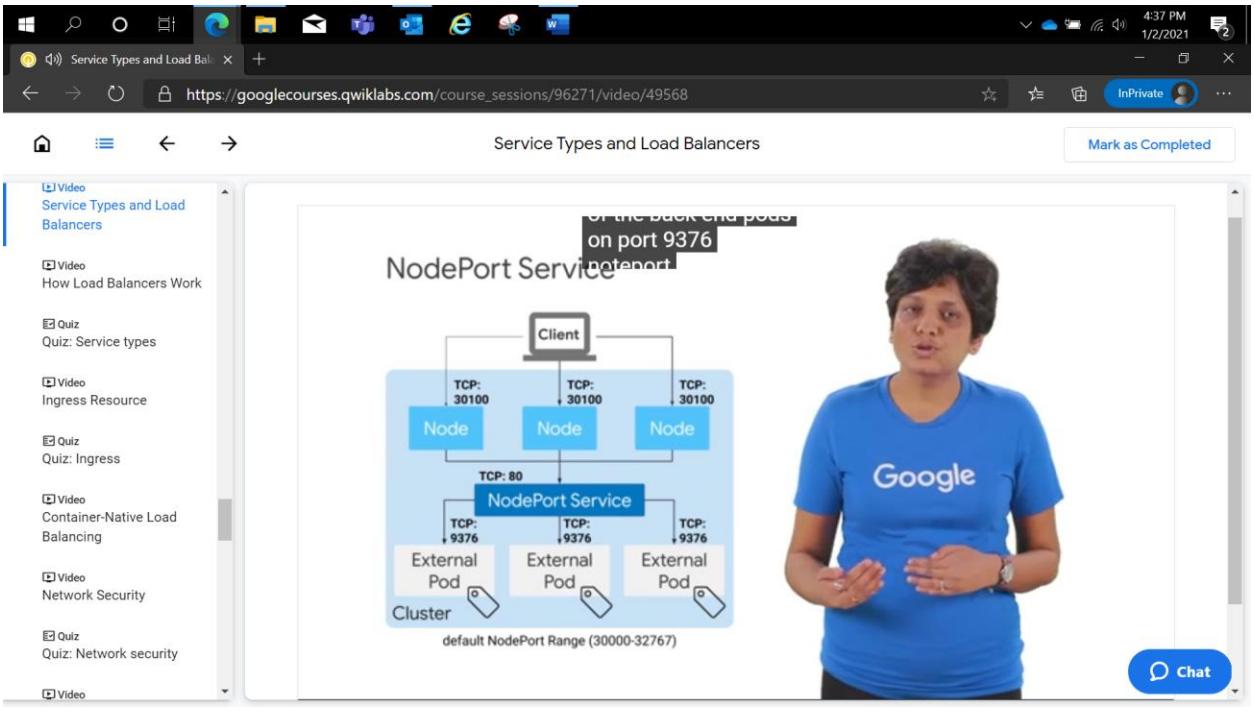

4:30 PM
1/2/2021
Quiz: Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49567
InPrivate
Quiz: Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
✓ 2. True or false: a Pod can connect to a Service running in a different namespace in its cluster.
Wrong answer 3
Wrong answer 2
✓ True.
False.
That is correct. Kubernetes DNS allows a mechanism for cross-namespace service discovery by offering fully-qualified domain names.
✓ 3. What is the purpose of Services?
Chat

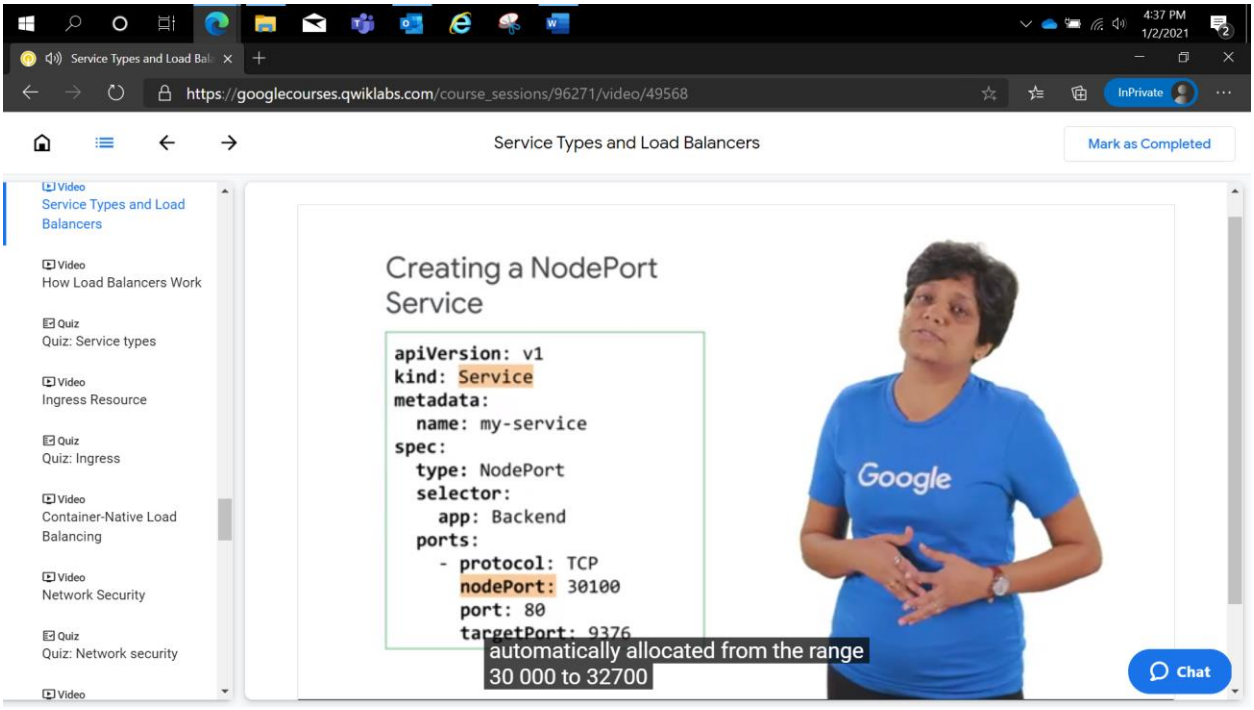

Quiz: Services | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49567
InPrivate
4:30 PM
1/2/2021
Quiz: Services
Quiz
Quiz: Services
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
That is correct. Kubernetes DNS allows a mechanism for cross-namespace service discovery by offering fully-qualified domain names.
3. What is the purpose of Services?
To launch and maintain a group of Deployments of a certain version.
To launch and maintain a group of Pods of a certain version.
To provide a static, load-balanced front end for transient Pods.
To provide a static, load-balanced front end for kube-apiserver.
That is correct.
Chat

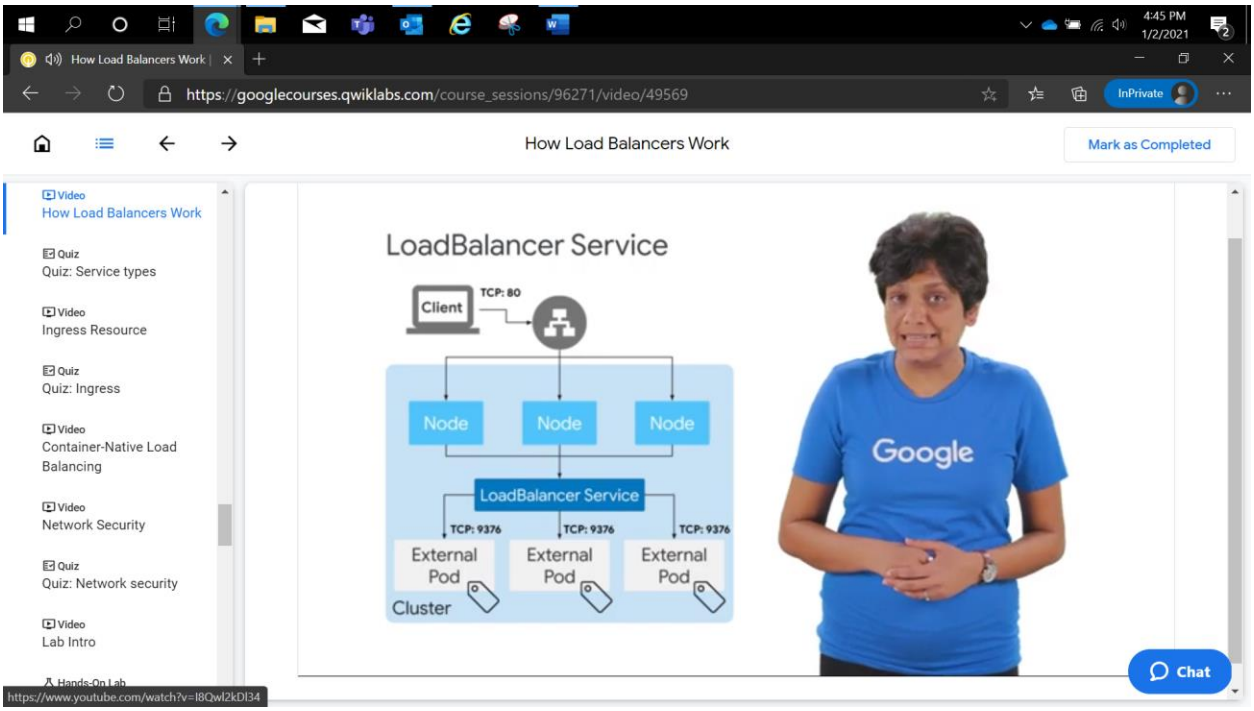

Service Types and Load Bal
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49568
InPrivate
4:31 PM
1/2/2021
Service Types and Load Balancers
Mark as Completed
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
A ClusterIP Service has a static IP address
Cluster
Frontend Pod
Frontend Pod
10.70.0.10
Service
"my-service"
ClusterIP
Backend Pod
Backend Pod
Backend Pod
the cluster ip service a kubernetes
cluster ip service has a static ip
Chat

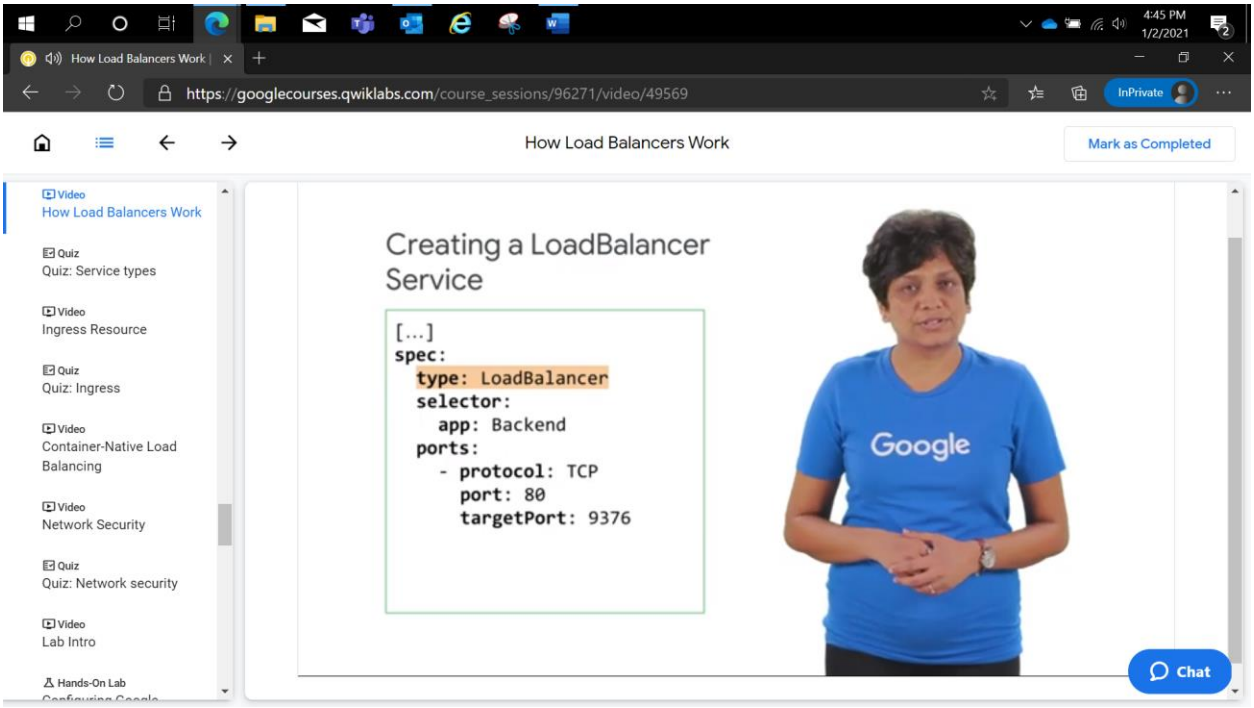

Service Types and Load Bal
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49568
InPrivate
4:31 PM
1/2/2021
Service Types and Load Balancers
Mark as Completed
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Setting up a ClusterIP Service
apiVersion: v1
kind: Service
metadata:
name: my-service
spec:
type: ClusterIP
selector:
app: Backend
ports:
- protocol: TCP
port: 3306
targetPort: 6000
Google
step-by-step process of setting up a cluster
Chat

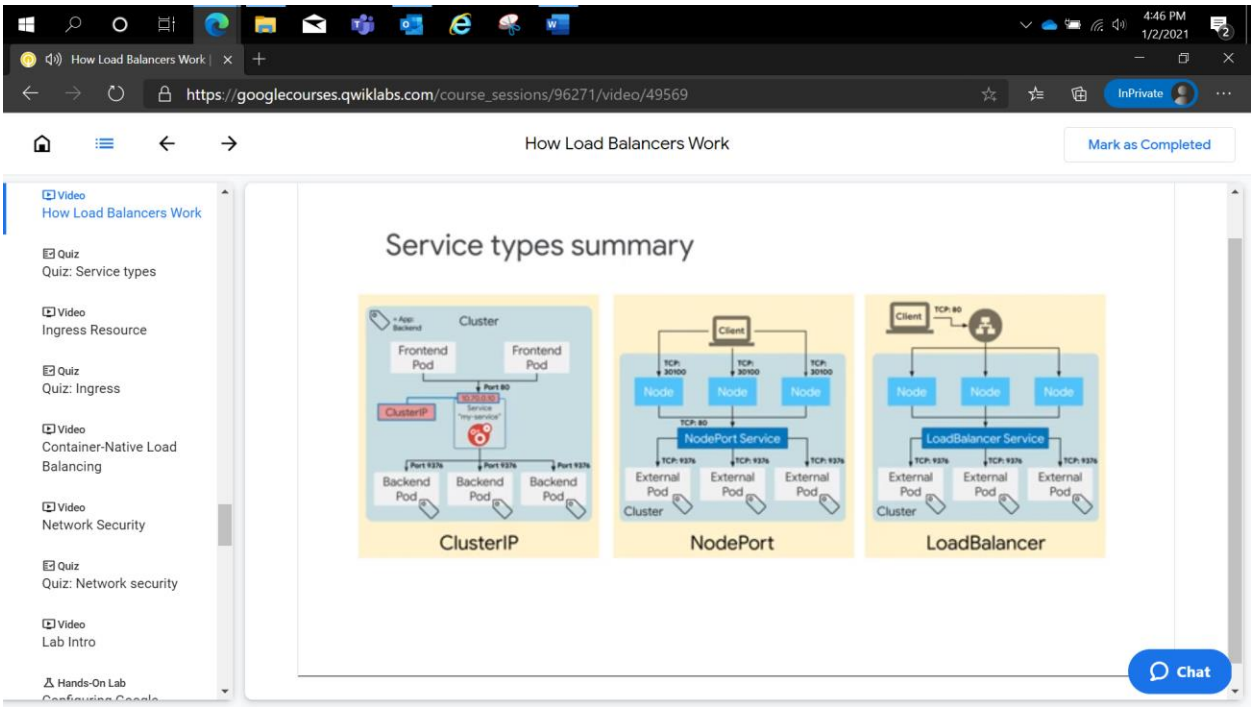

Service Types and Load Bal
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49568
InPrivate
4:35 PM
1/2/2021
Service Types and Load Balancers
Mark as Completed
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
How does the ClusterIP Service work?
Cluster
Frontend Pod
Frontend Pod
Port 80
10.70.0.10
Service "my-service"
ClusterIP
= App: Backend
Port 9376
Port 9376
Port 9376
Backend Pod
Backend Pod
Backend Pod
answer requests on tcp port 80.
kubernetes manages
Chat

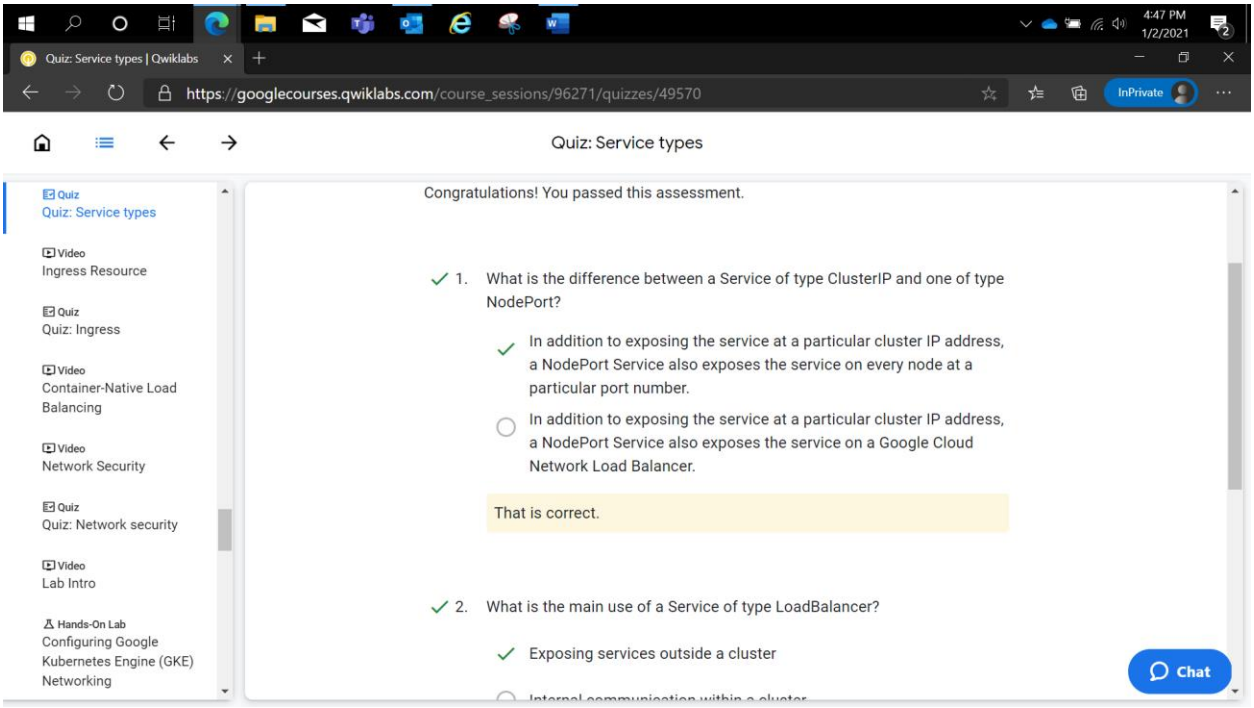

Service Types and Load Bal
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49568
InPrivate
4:37 PM
1/2/2021
Service Types and Load Balancers
Mark as Completed
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
on port 9376
NodePort Service
Client
TCP: 30100
Node
TCP: 80
NodePort Service
TCP: 9376
External Pod
Cluster
default NodePort Range (30000-32767)
Google
Chat

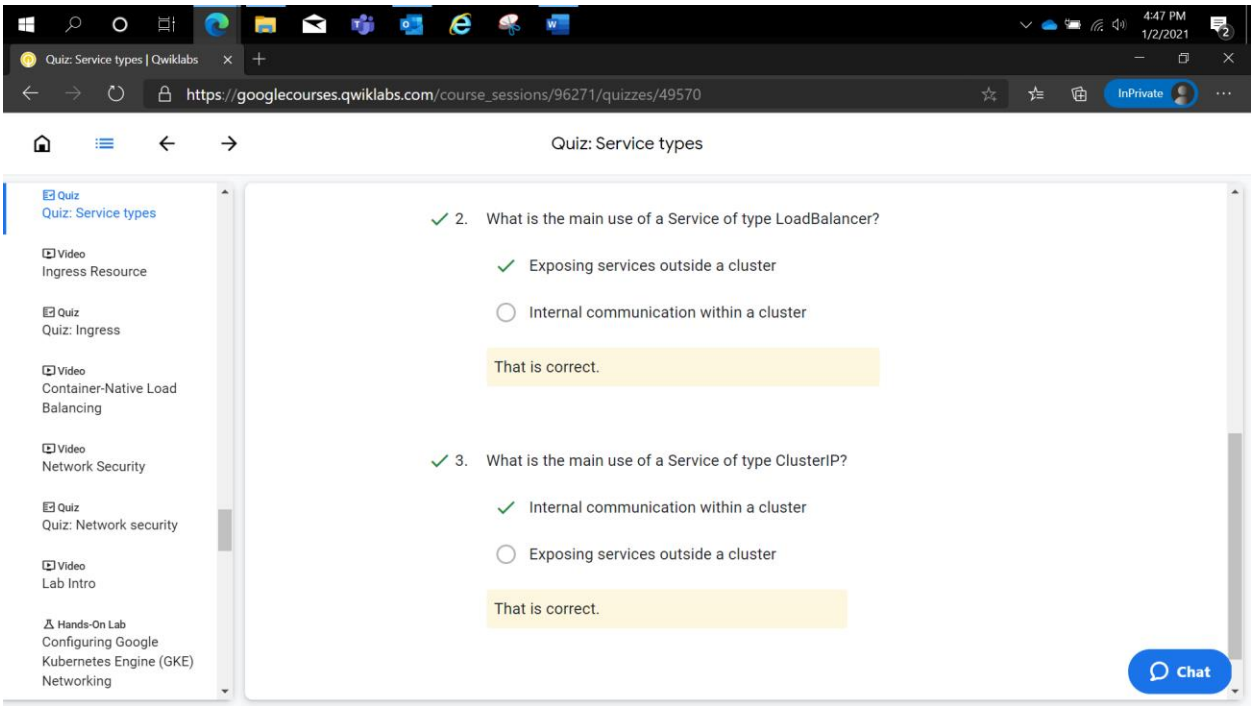

Service Types and Load Bal
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49568
InPrivate
4:37 PM
1/2/2021
Service Types and Load Balancers
Mark as Completed
Video
Service Types and Load Balancers
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Creating a NodePort Service
apiVersion: v1
kind: Service
metadata:
name: my-service
spec:
type: NodePort
selector:
app: Backend
ports:
- protocol: TCP
nodePort: 30100
port: 80
automatically allocated from the range
30 000 to 32700
Google
Chat

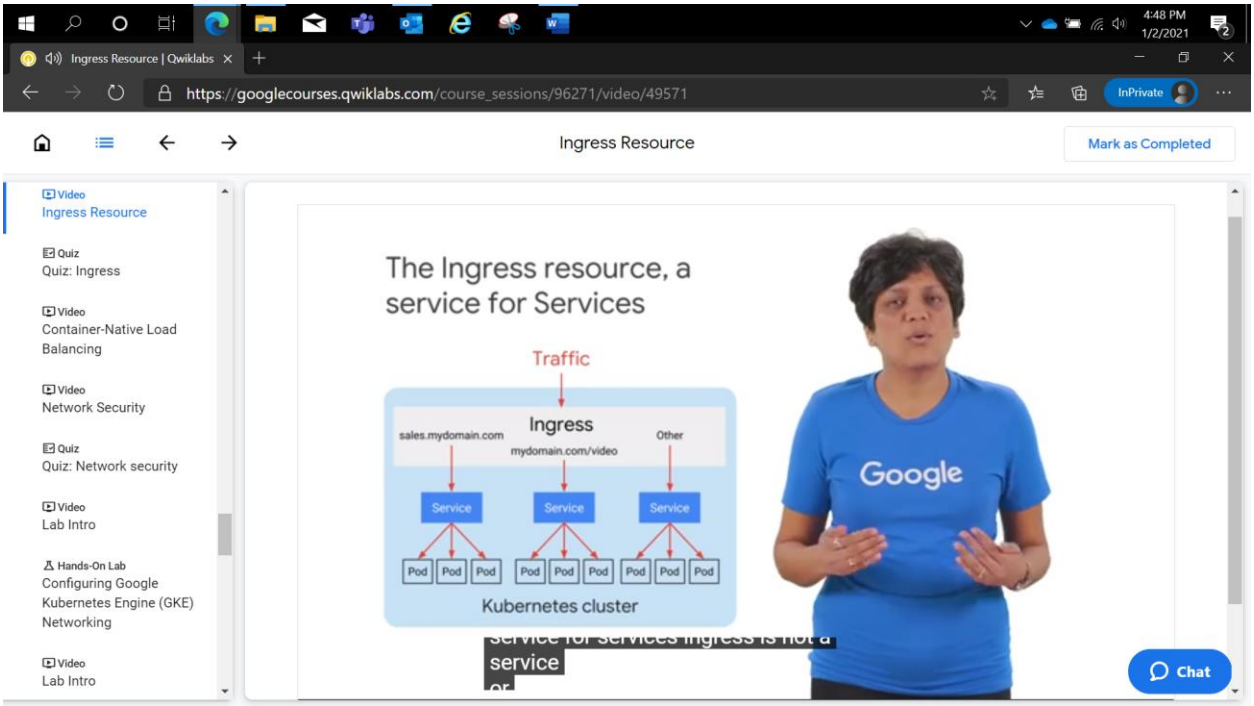

How Load Balancers Work
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49569
InPrivate
4:45 PM
1/2/2021
How Load Balancers Work
Mark as Completed
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
https://www.youtube.com/watch?v=I8QwI2kDI34
LoadBalancer Service
Client
TCP: 80
Node
Node
Node
LoadBalancer Service
TCP: 9376
TCP: 9376
TCP: 9376
External Pod
External Pod
External Pod
Cluster
Google
Chat

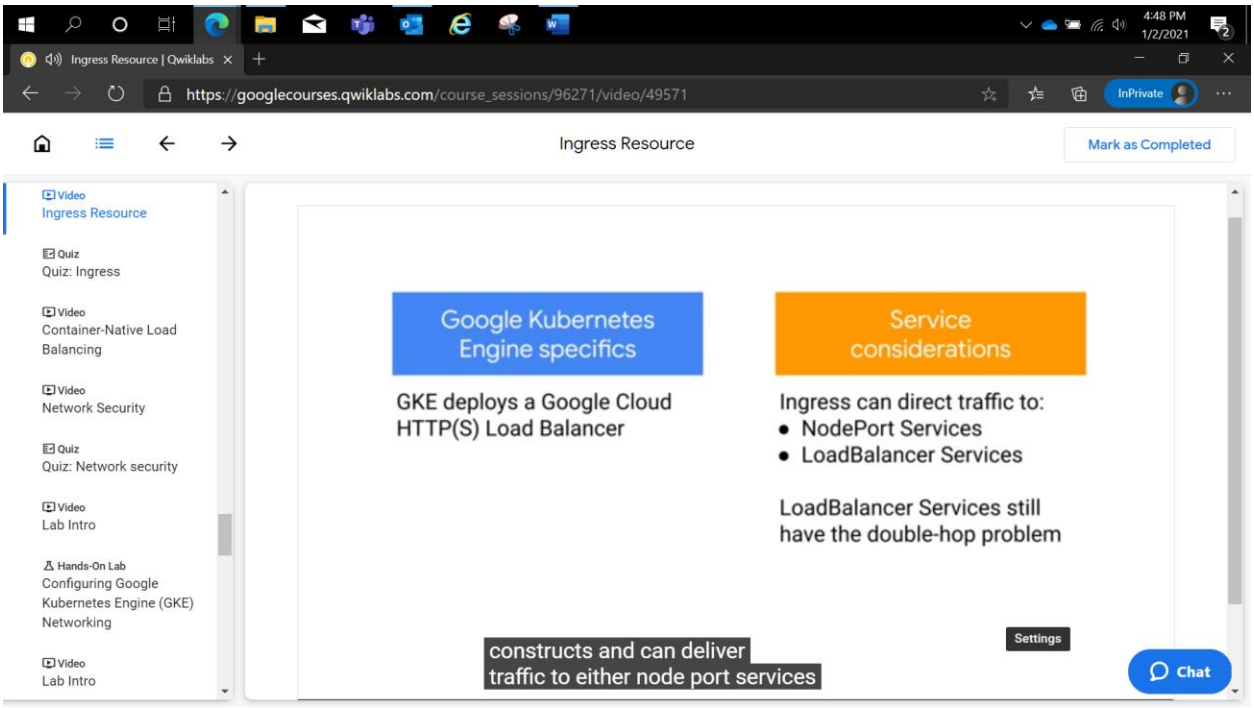

How Load Balancers Work
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49569
InPrivate
4:45 PM
1/2/2021
How Load Balancers Work
Mark as Completed
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Creating a LoadBalancer Service
[...]
spec:
type: LoadBalancer
selector:
app: Backend
ports:
- protocol: TCP
port: 80
targetPort: 9376
Google
Chat

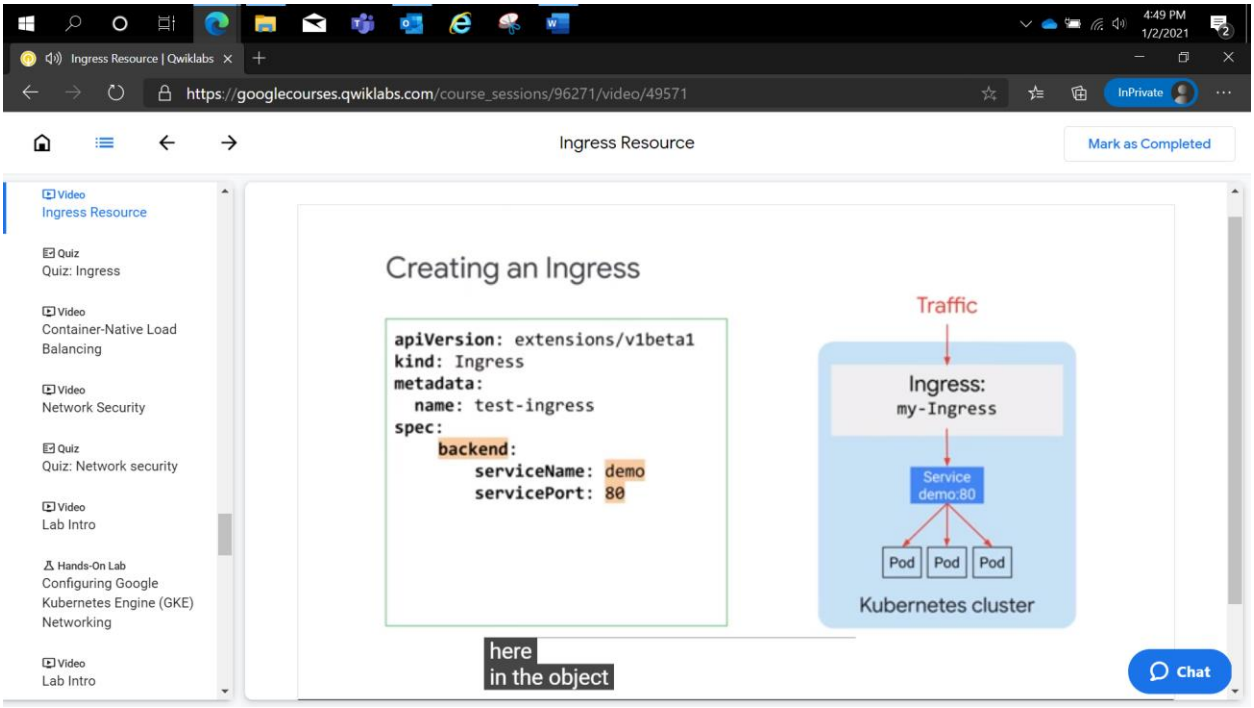

How Load Balancers Work
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49569
How Load Balancers Work
Mark as Completed
Video
How Load Balancers Work
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Service types summary
Cluster
Frontend Pod
Frontend Pod
ClusterIP
Backend Pod
Backend Pod
Backend Pod
ClusterIP
Client
Node
Node
Node
NodePort Service
External Pod
External Pod
External Pod
Cluster
NodePort
Client
Node
Node
Node
LoadBalancer Service
External Pod
External Pod
External Pod
Cluster
LoadBalancer
Chat

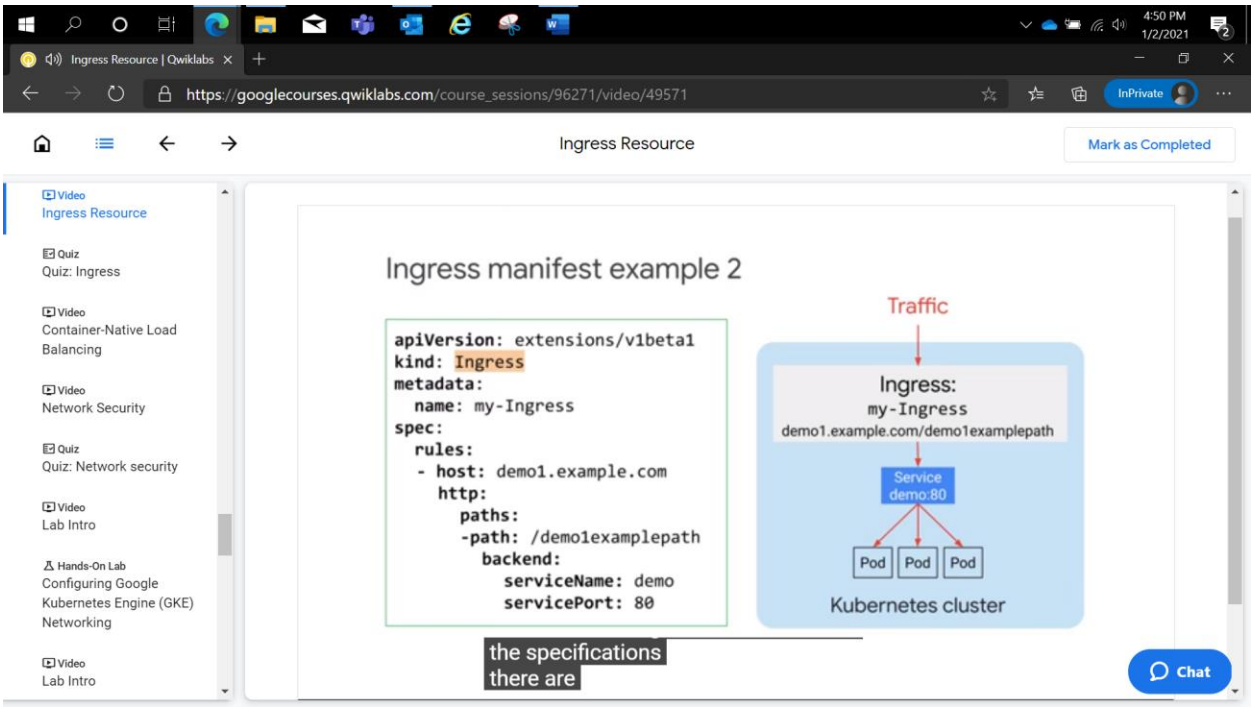

Quiz: Service types | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49570
Quiz: Service types
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Congratulations! You passed this assessment.
1. What is the difference between a Service of type ClusterIP and one of type NodePort?
In addition to exposing the service at a particular cluster IP address, a NodePort Service also exposes the service on every node at a particular port number.
In addition to exposing the service at a particular cluster IP address, a NodePort Service also exposes the service on a Google Cloud Network Load Balancer.
That is correct.
2. What is the main use of a Service of type LoadBalancer?
Exposing services outside a cluster
Chat

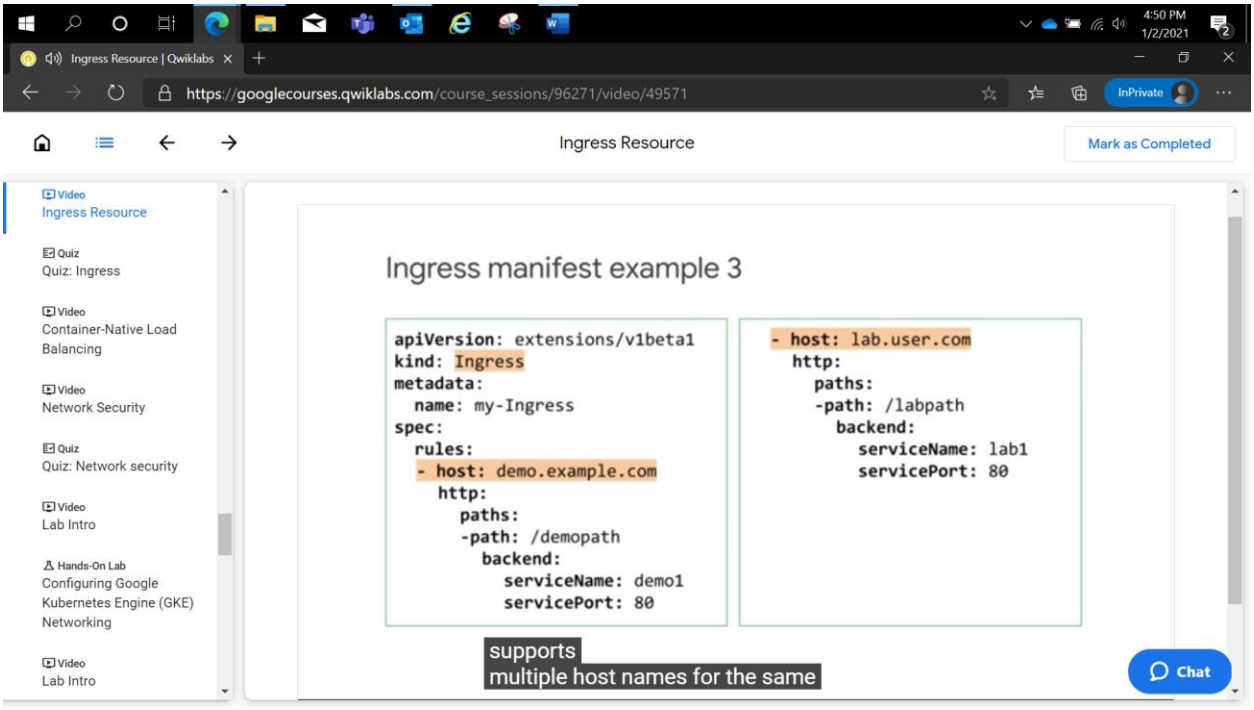

Quiz: Service types | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49570
InPrivate
4:47 PM
1/2/2021
Quiz: Service types
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
2. What is the main use of a Service of type LoadBalancer?
Exposing services outside a cluster
Internal communication within a cluster
That is correct.
3. What is the main use of a Service of type ClusterIP?
Internal communication within a cluster
Exposing services outside a cluster
That is correct.
Chat

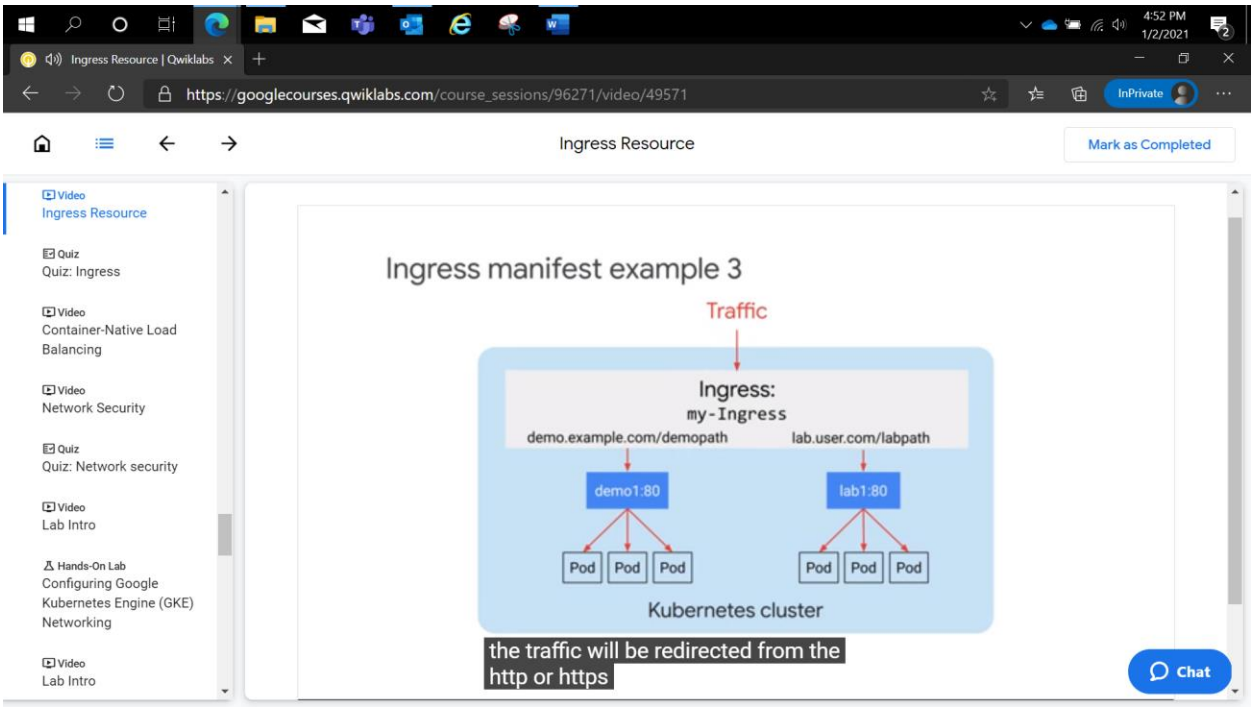

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:48 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
The Ingress resource, a service for Services
Traffic
Ingress
sales.mydomain.com
mydomain.com/video
Other
Service
Service
Service
Pod
Kubernetes cluster
Google
service
Chat

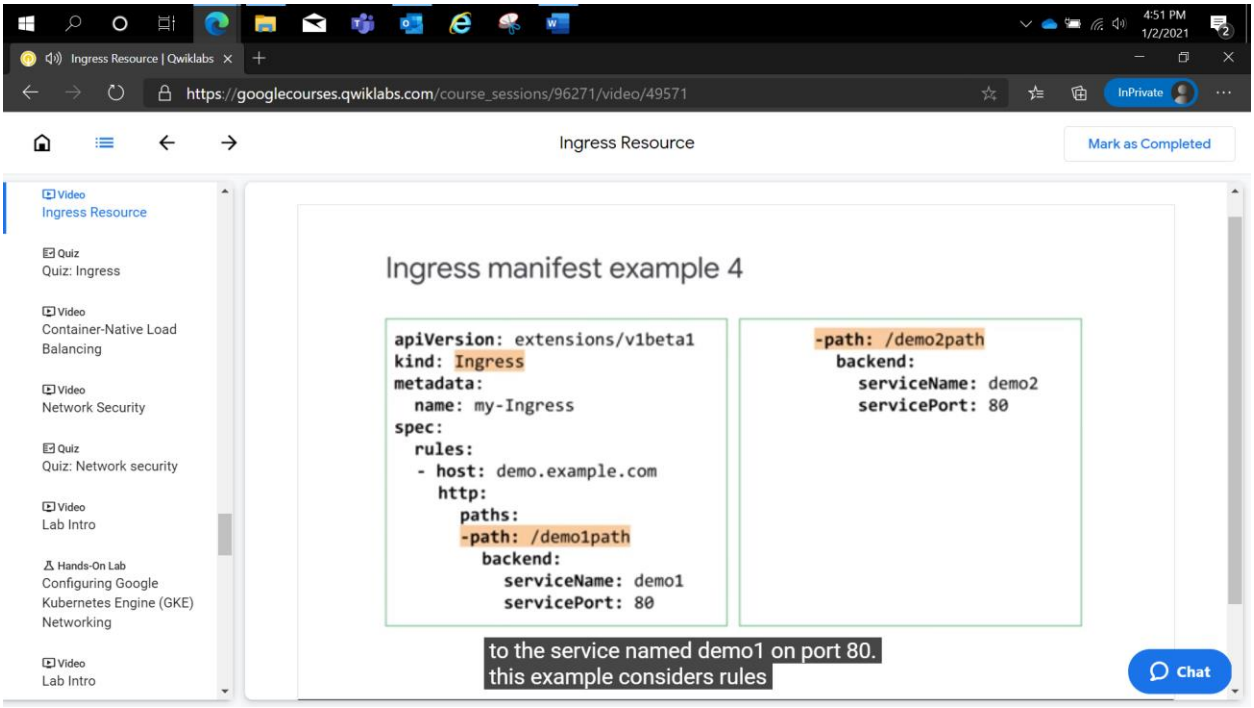

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:48 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Google Kubernetes Engine specifics
GKE deploys a Google Cloud HTTP(S) Load Balancer
Service considerations
Ingress can direct traffic to:
NodePort Services
LoadBalancer Services
LoadBalancer Services still have the double-hop problem
constructs and can deliver traffic to either node port services
Settings
Chat

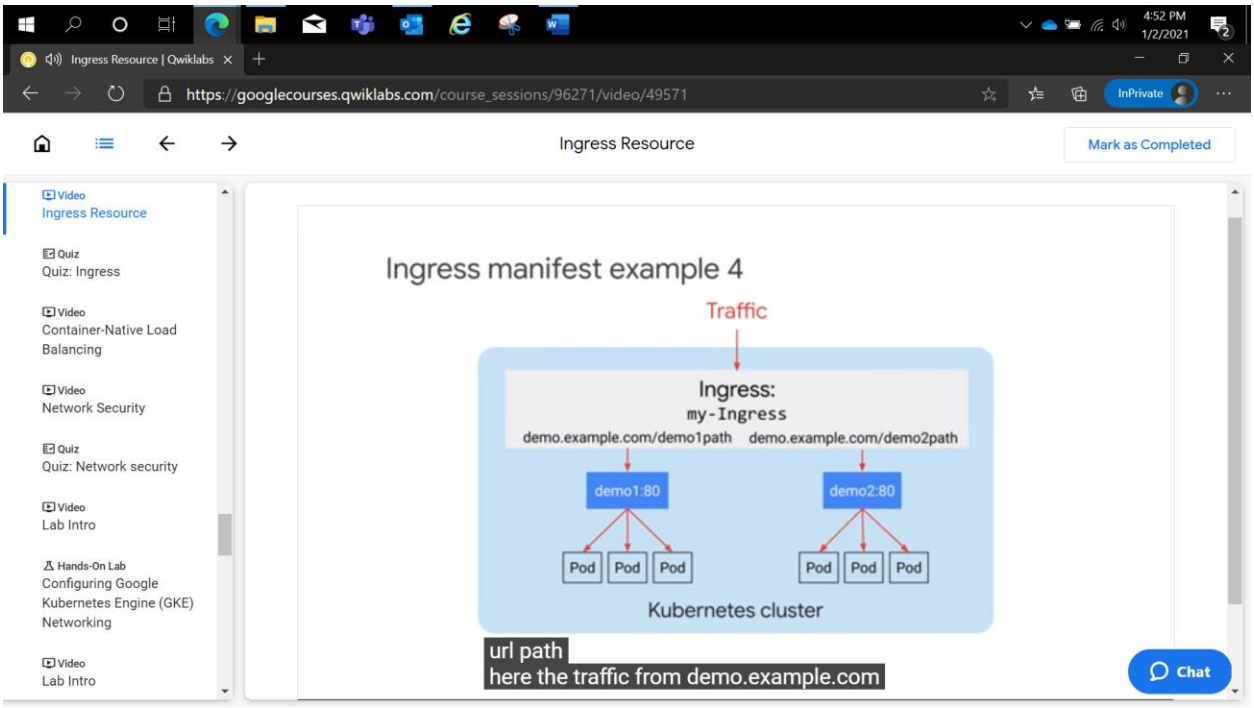

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:49 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Creating an Ingress
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
name: test-ingress
spec:
backend:
serviceName: demo
servicePort: 80
Traffic
Ingress:
my-Ingress
Service
demo:80
Pod
Pod
Pod
Kubernetes cluster
here in the object
Chat

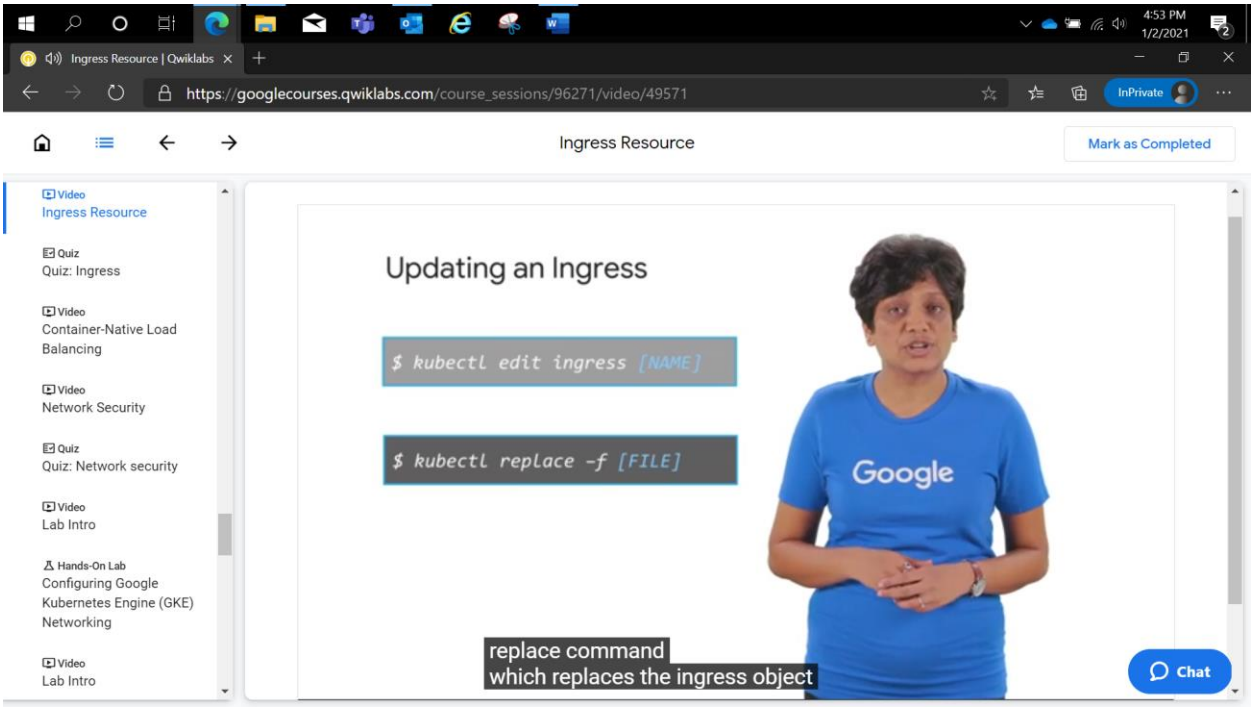

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:50 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Ingress manifest example 2
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
name: my-Ingress
spec:
rules:
- host: demo1.example.com
http:
paths:
-path: /demo1examplepath
backend:
serviceName: demo
servicePort: 80
Traffic
Ingress:
my-Ingress
demo1.example.com/demo1examplepath
Service
demo:80
Pod
Pod
Pod
Kubernetes cluster
the specifications
there are
Chat

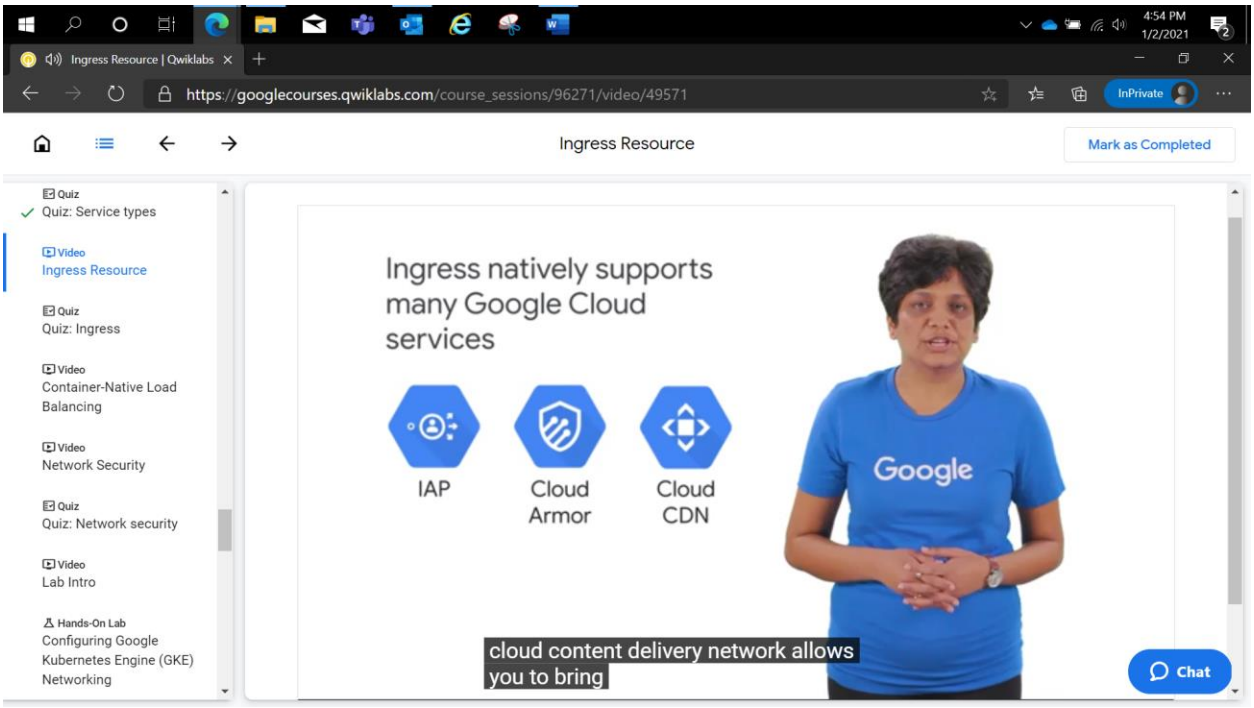

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:50 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Ingress manifest example 3
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
name: my-Ingress
spec:
rules:
- host: demo.example.com
http:
paths:
-path: /demopath
backend:
serviceName: demo1
servicePort: 80
- host: lab.user.com
http:
paths:
-path: /labpath
backend:
serviceName: lab1
servicePort: 80
supports
multiple host names for the same
Chat

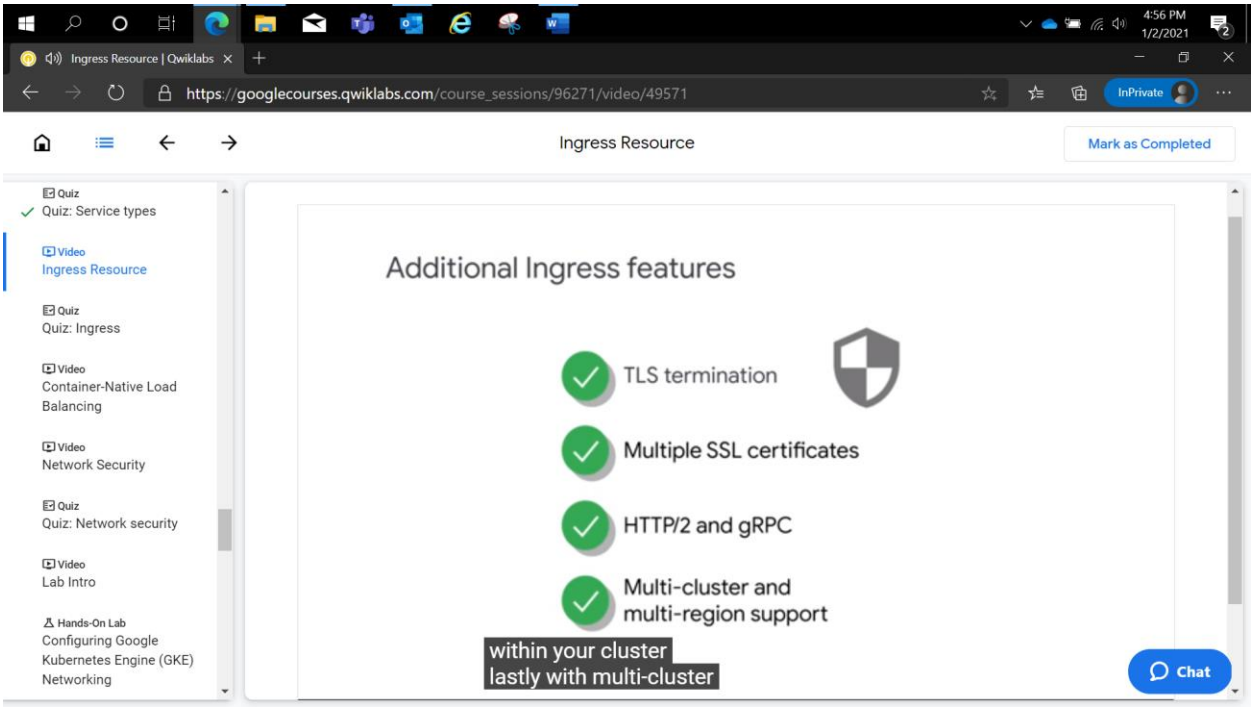

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:52 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Ingress manifest example 3
Traffic
Ingress:
my-Ingress
demo.example.com/demopath
lab.user.com/labpath
demo1:80
lab1:80
Pod
Pod
Pod
Pod
Pod
Pod
Kubernetes cluster
the traffic will be redirected from the http or https
Chat

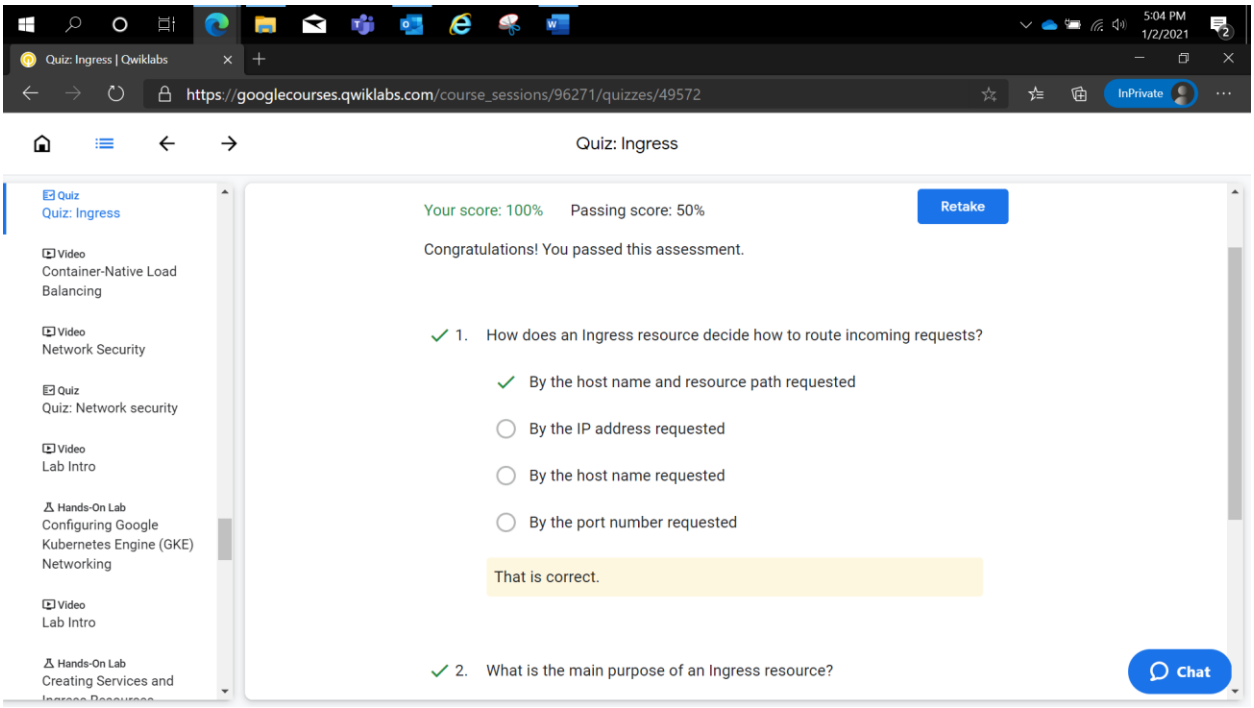

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:51 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Ingress manifest example 4
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
name: my-Ingress
spec:
rules:
- host: demo.example.com
http:
paths:
-path: /demo1path
backend:
serviceName: demo1
servicePort: 80
-path: /demo2path
backend:
serviceName: demo2
servicePort: 80
to the service named demo1 on port 80.
this example considers rules
Chat

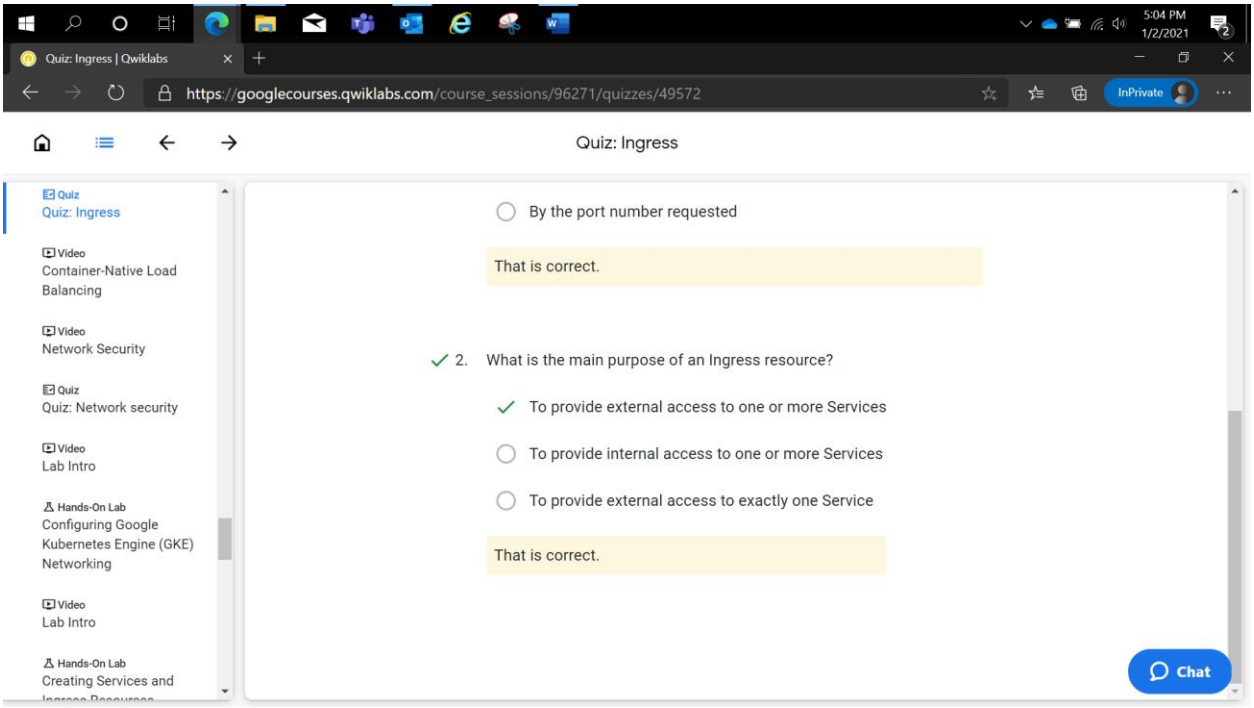

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:52 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Ingress manifest example 4
Traffic
Ingress:
my-Ingress
demo.example.com/demo1path
demo.example.com/demo2path
demo1:80
demo2:80
Pod
Pod
Pod
Pod
Pod
Pod
Kubernetes cluster
url path
here the traffic from demo.example.com
Chat

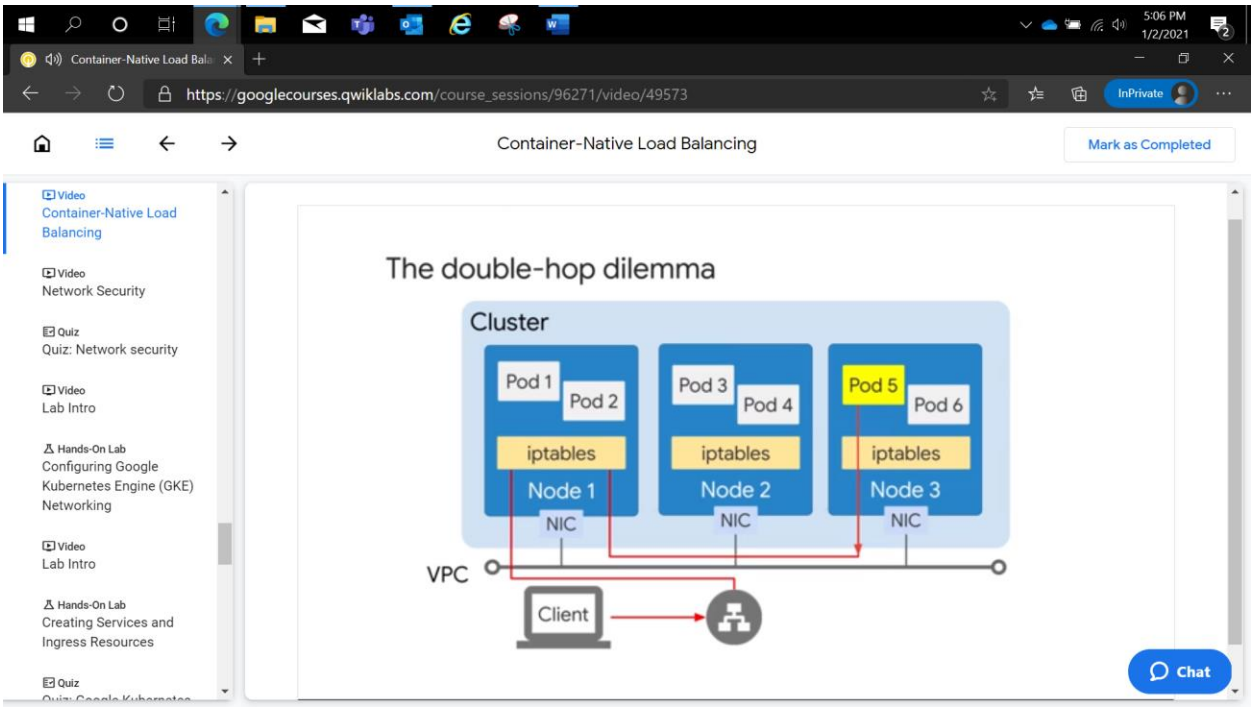

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:53 PM
1/2/2021
Ingress Resource
Mark as Completed
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Updating an Ingress
$ kubectl edit ingress [NAME]
$ kubectl replace -f [FILE]
Google
replace command
which replaces the ingress object
Chat

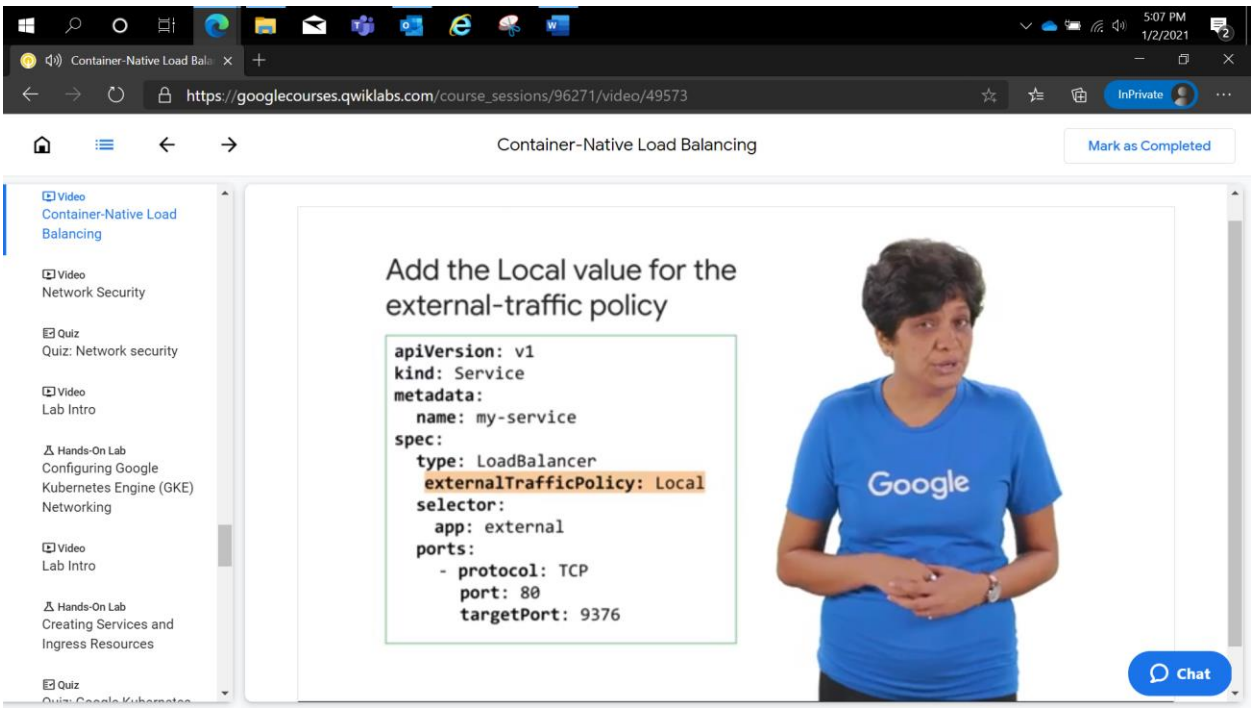

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:54 PM
1/2/2021
Ingress Resource
Mark as Completed
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Ingress natively supports many Google Cloud services
IAP
Cloud Armor
Cloud CDN
Google
cloud content delivery network allows
you to bring
Chat

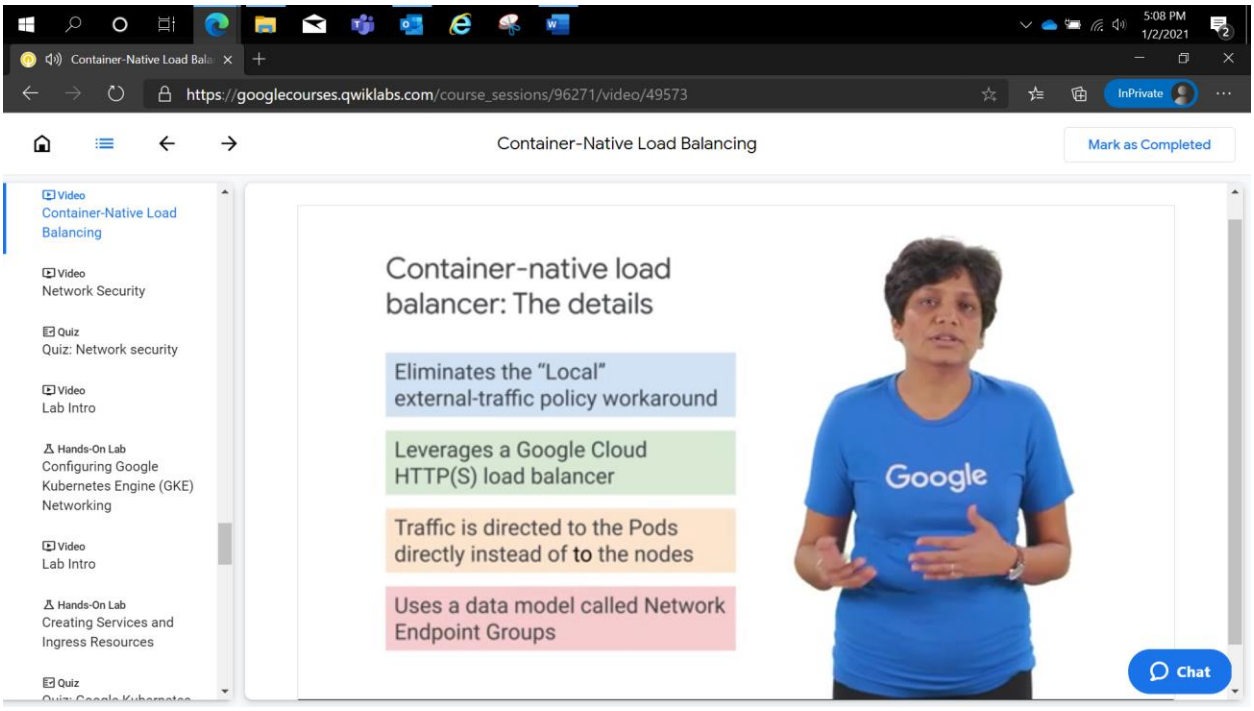

Ingress Resource | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49571
InPrivate
4:56 PM
1/2/2021
Ingress Resource
Mark as Completed
Quiz
Quiz: Service types
Video
Ingress Resource
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Additional Ingress features
TLS termination
Multiple SSL certificates
HTTP/2 and gRPC
Multi-cluster and multi-region support
within your cluster
lastly with multi-cluster
Chat

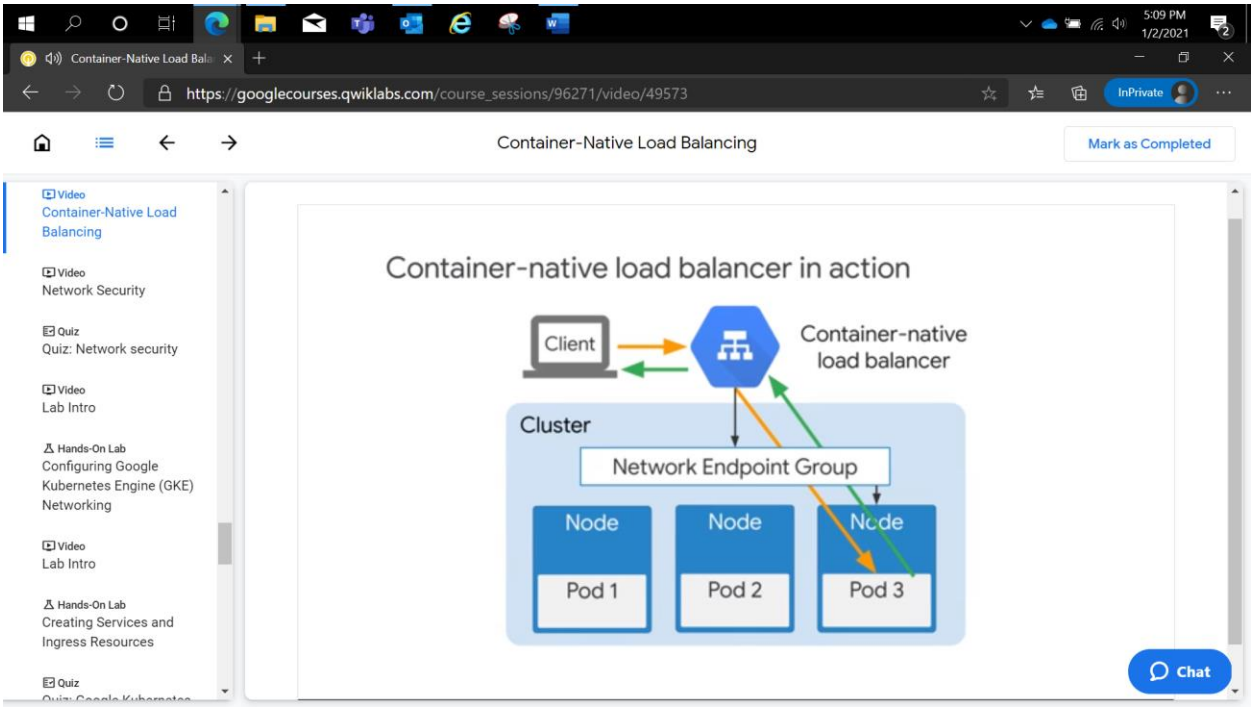

5:04 PM
1/2/2021
Quiz: Ingress | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49572
InPrivate
Quiz: Ingress
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and
Your score: 100%
Passing score: 50%
Retake
Congratulations! You passed this assessment.
1. How does an Ingress resource decide how to route incoming requests?
By the host name and resource path requested
By the IP address requested
By the host name requested
By the port number requested
That is correct.
2. What is the main purpose of an Ingress resource?
Chat

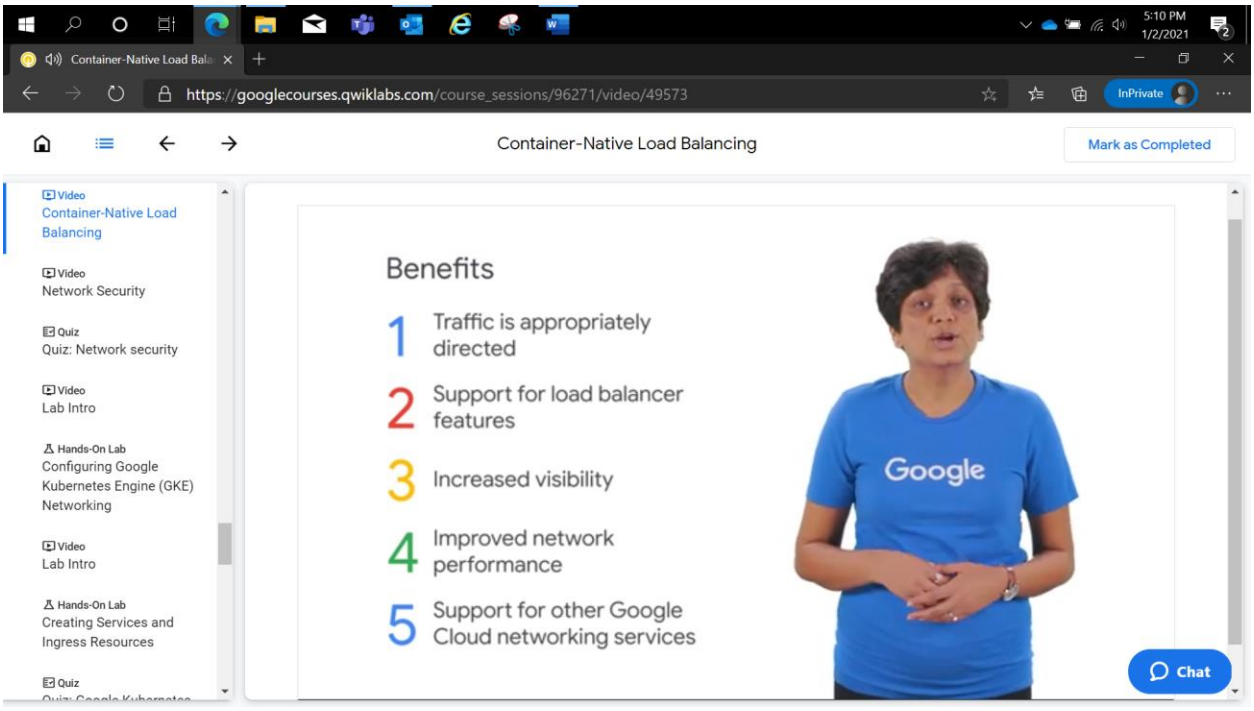

5:04 PM
1/2/2021
Quiz: Ingress | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49572
InPrivate
Quiz: Ingress
Quiz
Quiz: Ingress
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and
By the port number requested
That is correct.
2. What is the main purpose of an Ingress resource?
To provide external access to one or more Services
To provide internal access to one or more Services
To provide external access to exactly one Service
That is correct.
Chat

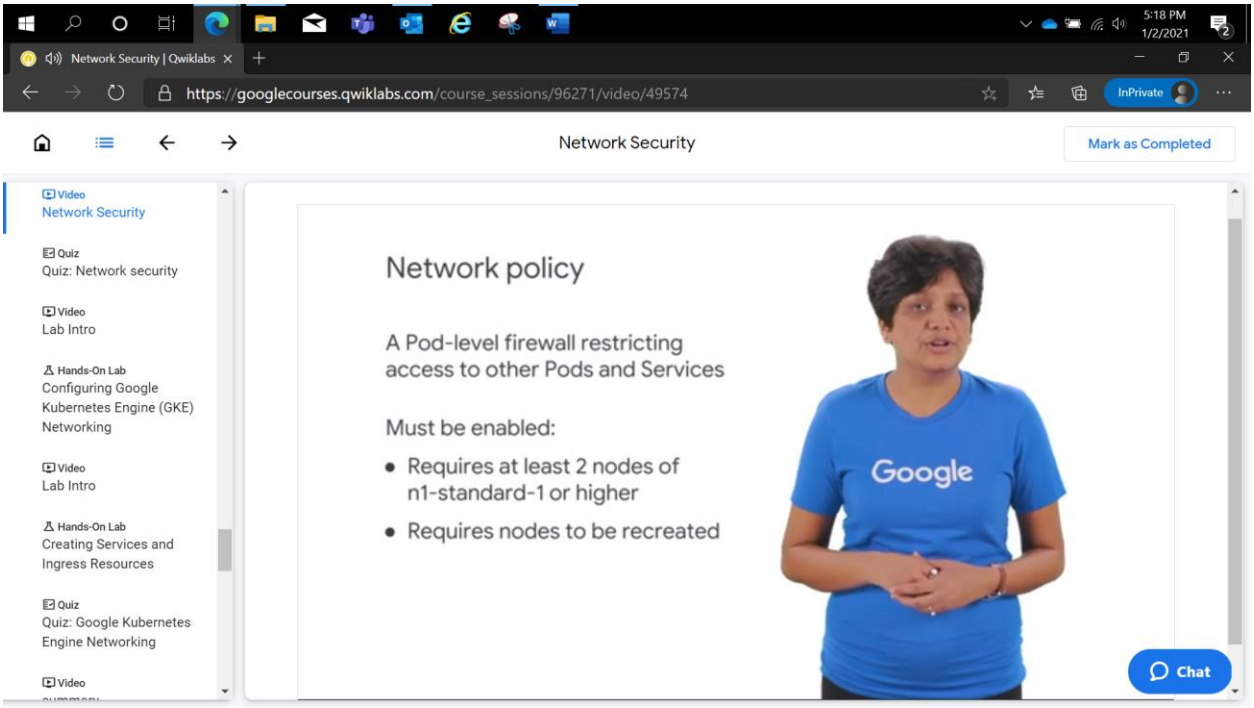

5:06 PM
1/2/2021
Container-Native Load Bala
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49573
InPrivate
Container-Native Load Balancing
Mark as Completed
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
The double-hop dilemma
Cluster
Pod 1
Pod 2
Pod 3
Pod 4
Pod 5
Pod 6
iptables
iptables
iptables
Node 1
Node 2
Node 3
NIC
NIC
NIC
VPC
Client
Chat

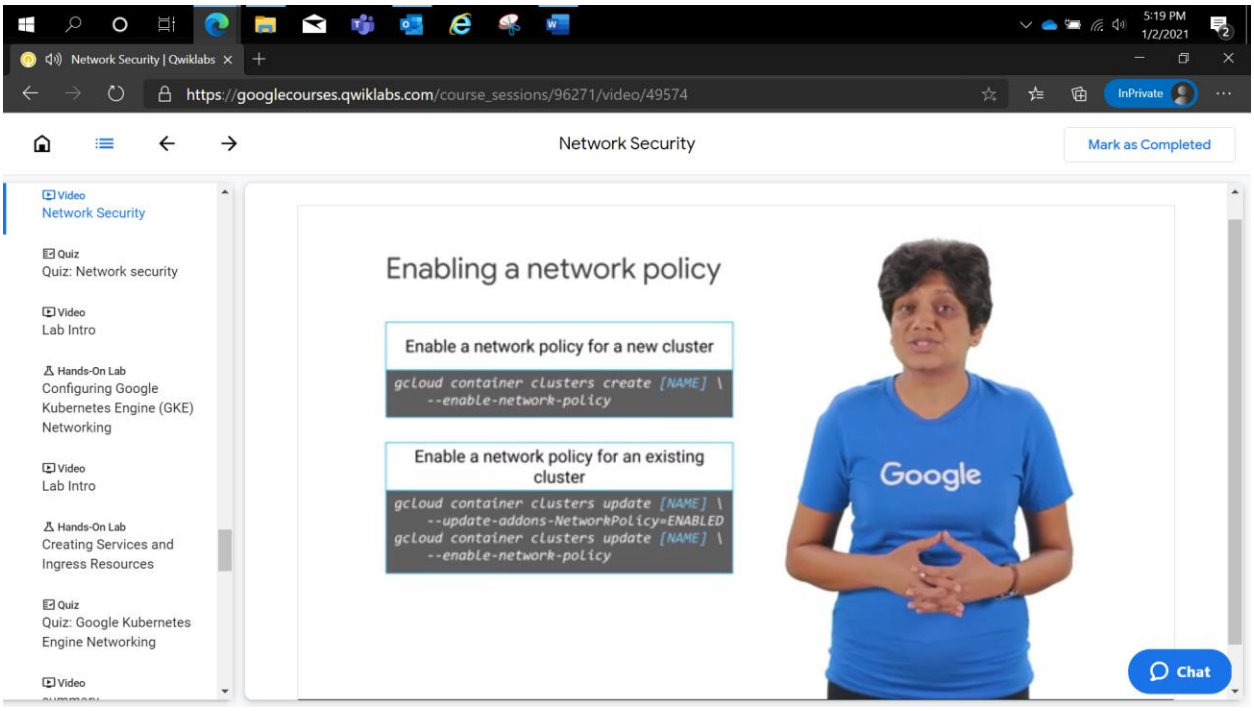

5:07 PM
1/2/2021
Container-Native Load Bala
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49573
InPrivate
Container-Native Load Balancing
Mark as Completed
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Add the Local value for the external-traffic policy
apiVersion: v1
kind: Service
metadata:
name: my-service
spec:
type: LoadBalancer
externalTrafficPolicy: Local
selector:
app: external
ports:
- protocol: TCP
port: 80
targetPort: 9376
Google
Chat

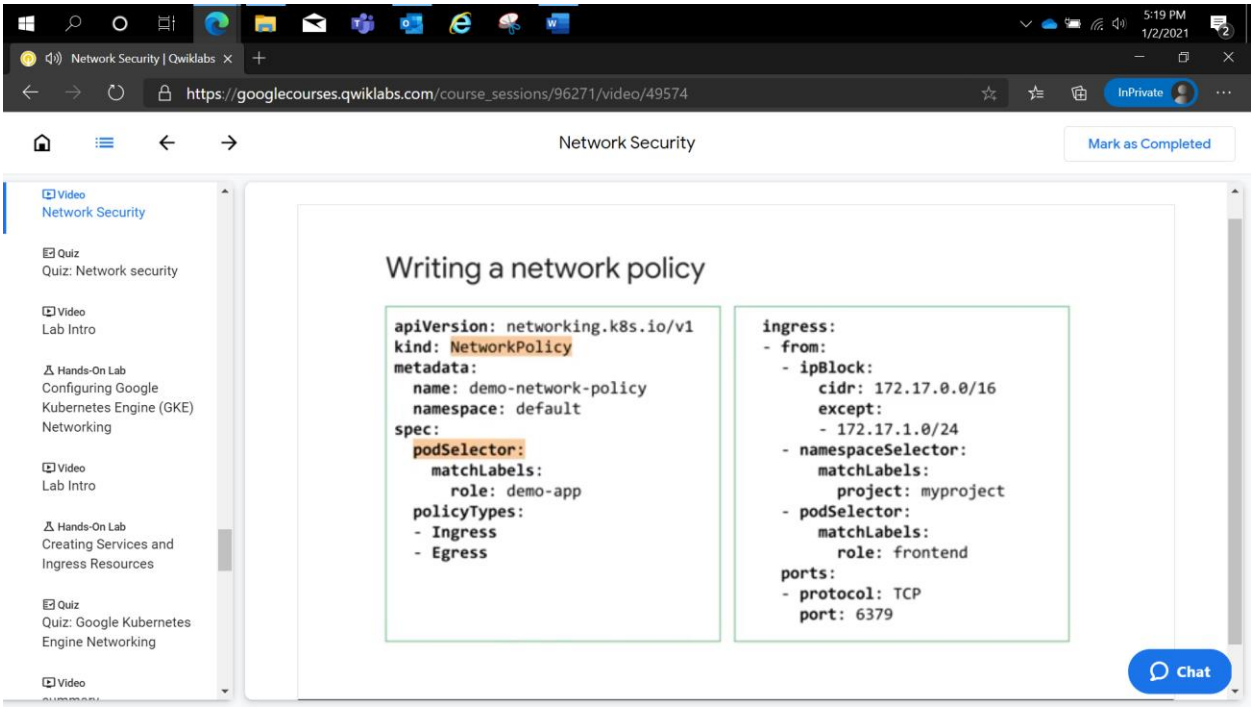

Container-Native Load Bala
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49573
InPrivate
5:08 PM
1/2/2021
Container-Native Load Balancing
Mark as Completed
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Container-native load balancer: The details
Eliminates the "Local" external-traffic policy workaround
Leverages a Google Cloud HTTP(S) load balancer
Traffic is directed to the Pods directly instead of to the nodes
Uses a data model called Network Endpoint Groups
Google
Chat

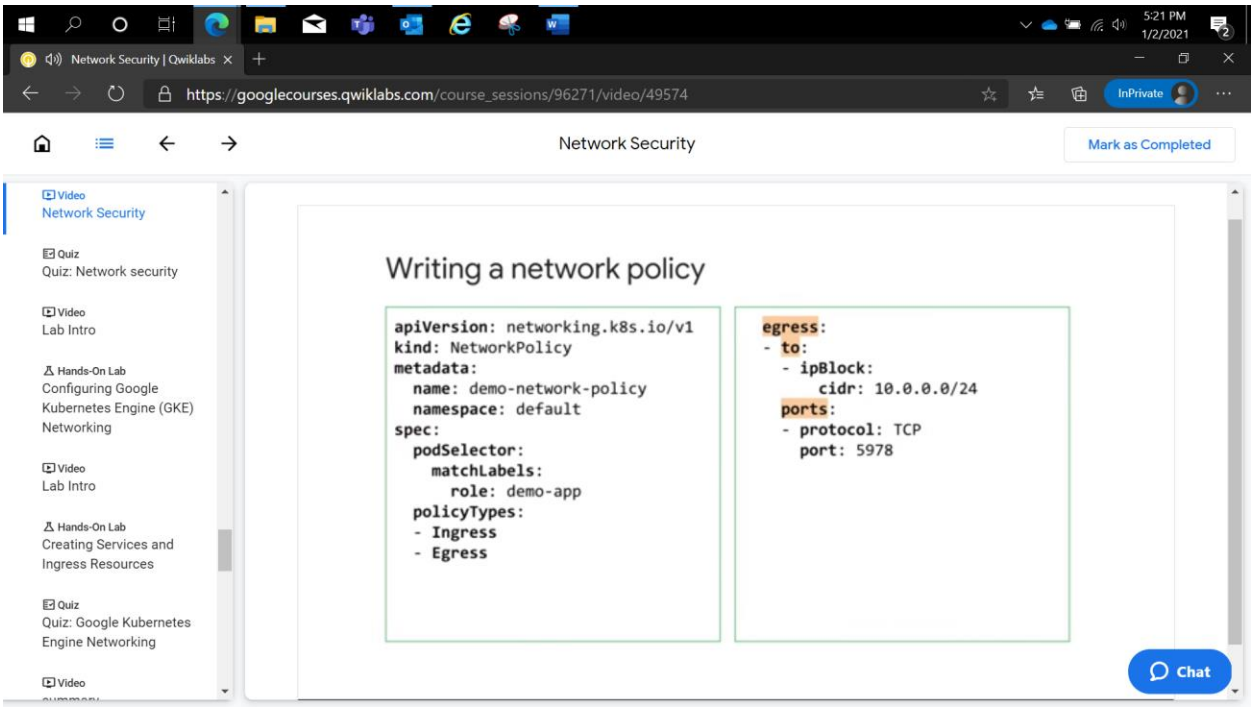

Container-Native Load Bala
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49573
InPrivate
5:09 PM
1/2/2021
Container-Native Load Balancing
Mark as Completed
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Container-native load balancer in action
Client
Container-native load balancer
Cluster
Network Endpoint Group
Node
Pod 1
Node
Pod 2
Node
Pod 3
Chat

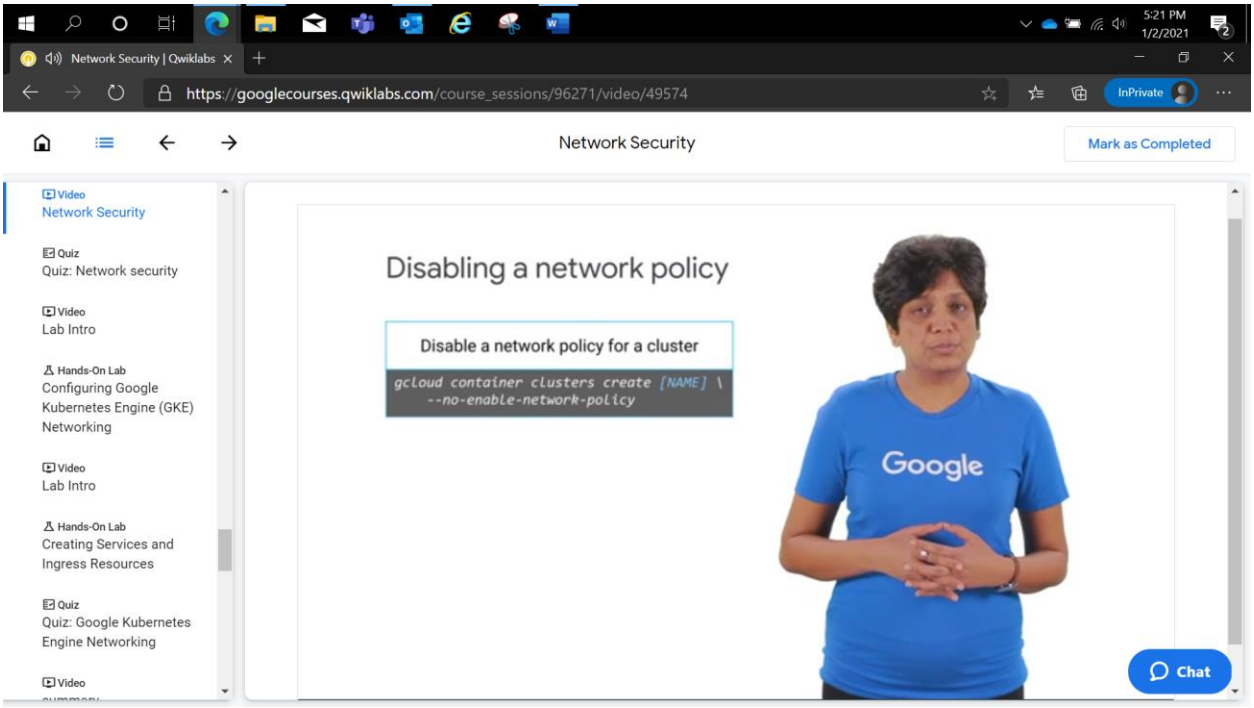

Container-Native Load Bala
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49573
InPrivate
5:10 PM
1/2/2021
Container-Native Load Balancing
Mark as Completed
Video
Container-Native Load Balancing
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Benefits
1 Traffic is appropriately directed
2 Support for load balancer features
3 Increased visibility
4 Improved network performance
5 Support for other Google Cloud networking services
Google
Chat

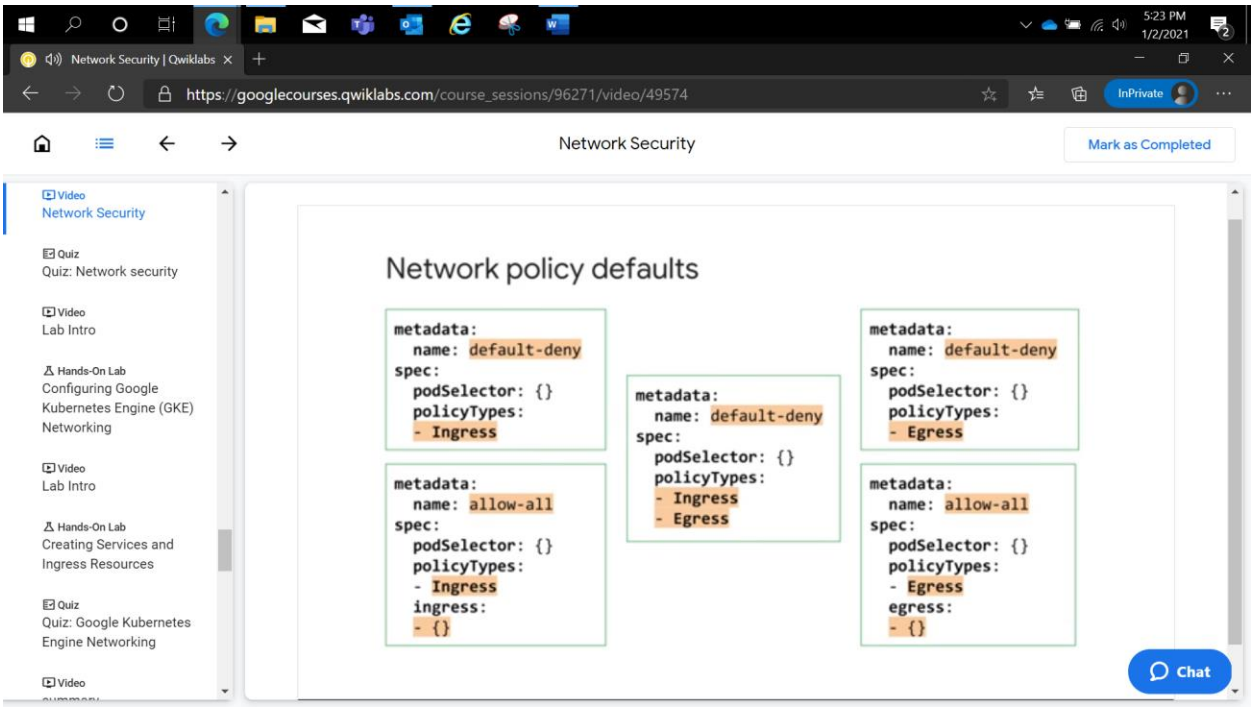

Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:18 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Network policy
A Pod-level firewall restricting access to other Pods and Services
Must be enabled:
• Requires at least 2 nodes of n1-standard-1 or higher
• Requires nodes to be recreated
Google
Chat

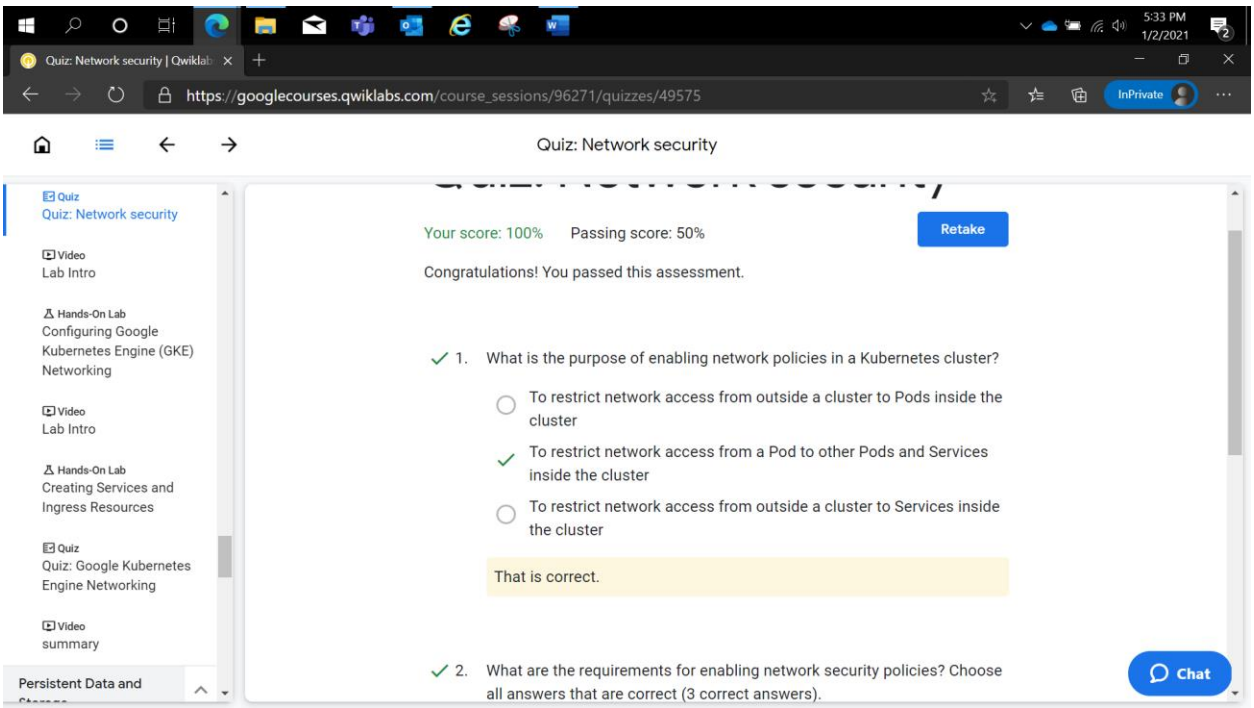

Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:19 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Enabling a network policy
Enable a network policy for a new cluster
gcloud container clusters create [NAME] \
--enable-network-policy
Enable a network policy for an existing cluster
gcloud container clusters update [NAME] \
--update-addons-NetworkPolicy=ENABLED
gcloud container clusters update [NAME] \
--enable-network-policy
Google
Chat

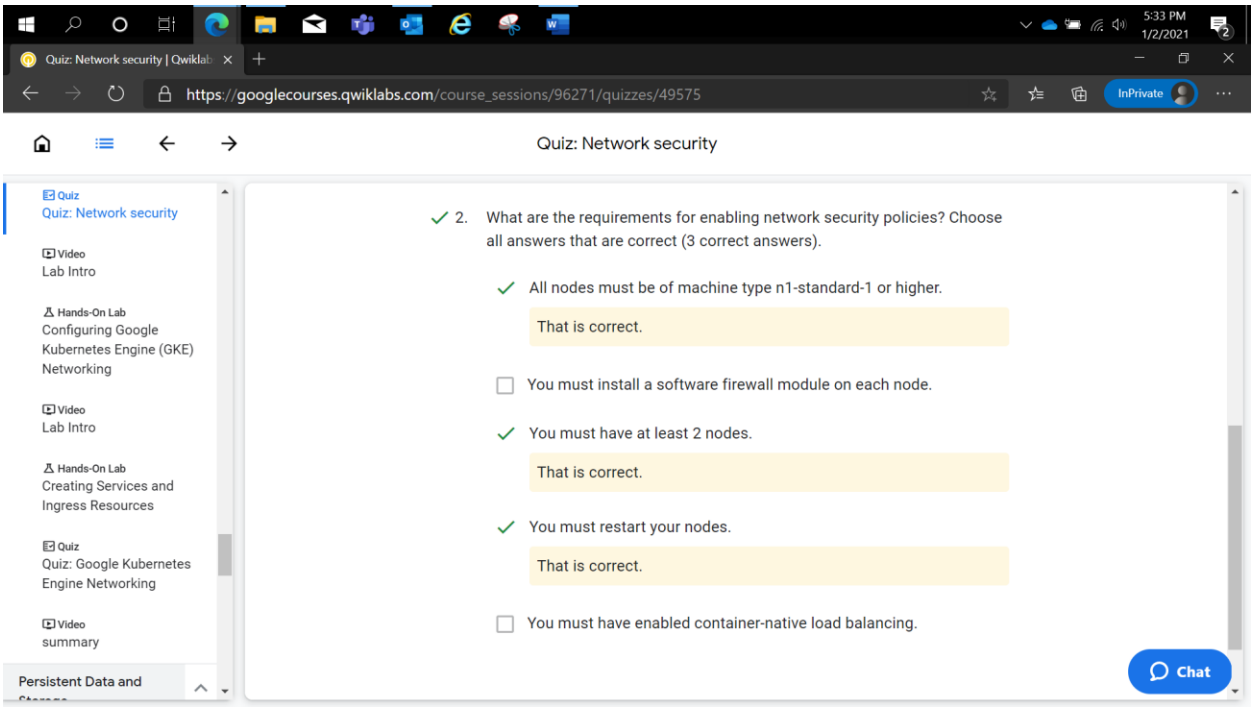

Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:19 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Writing a network policy
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
name: demo-network-policy
namespace: default
spec:
podSelector:
matchLabels:
role: demo-app
policyTypes:
- Ingress
- Egress
ingress:
- from:
- ipBlock:
cidr: 172.17.0.0/16
except:
- 172.17.1.0/24
- namespaceSelector:
matchLabels:
project: myproject
- podSelector:
matchLabels:
role: frontend
ports:
- protocol: TCP
port: 6379
Chat

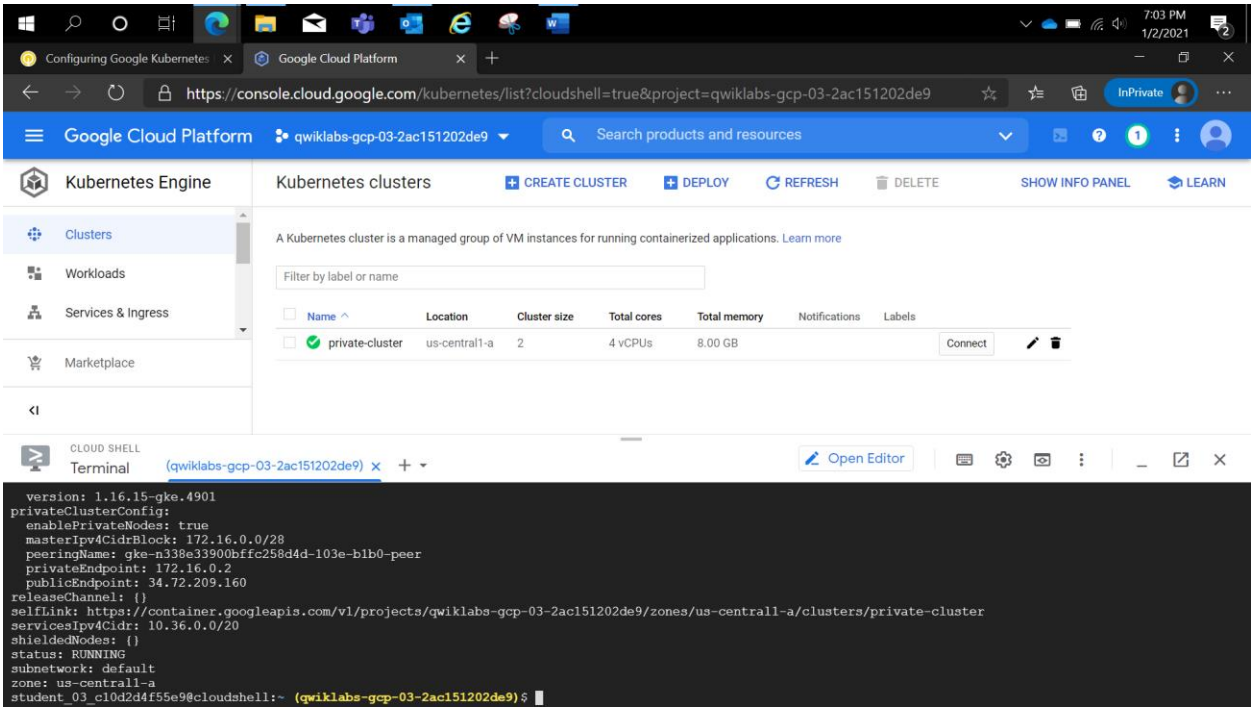

Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:21 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Writing a network policy
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
name: demo-network-policy
namespace: default
spec:
podSelector:
matchLabels:
role: demo-app
policyTypes:
- Ingress
- Egress
egress:
- to:
- ipBlock:
cidr: 10.0.0.0/24
ports:
- protocol: TCP
port: 5978
Chat


Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:21 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Disabling a network policy
Disable a network policy for a cluster
gcloud container clusters create [NAME] \
--no-enable-network-policy
Google
Chat


Network Security | Qwiklabs
https://googlecourses.qwiklabs.com/course_sessions/96271/video/49574
InPrivate
5:23 PM
1/2/2021
Network Security
Mark as Completed
Video
Network Security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
Network policy defaults
metadata:
name: default-deny
spec:
podSelector: {}
policyTypes:
- Ingress
metadata:
name: default-deny
spec:
podSelector: {}
policyTypes:
- Egress
metadata:
name: default-deny
spec:
podSelector: {}
policyTypes:
- Ingress
- Egress
metadata:
name: allow-all
spec:
podSelector: {}
policyTypes:
- Ingress
ingress:
- {}
metadata:
name: allow-all
spec:
podSelector: {}
policyTypes:
- Egress
egress:
- {}
Chat


Quiz: Network security | Qwiklab
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49575
InPrivate
5:33 PM
1/2/2021
Quiz: Network security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and
Your score: 100%
Passing score: 50%
Retake
Congratulations! You passed this assessment.
1. What is the purpose of enabling network policies in a Kubernetes cluster?
To restrict network access from outside a cluster to Pods inside the cluster
To restrict network access from a Pod to other Pods and Services inside the cluster
To restrict network access from outside a cluster to Services inside the cluster
That is correct.
2. What are the requirements for enabling network security policies? Choose all answers that are correct (3 correct answers).
Chat


Quiz: Network security | Qwiklab
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49575
InPrivate
Quiz: Network security
Quiz
Quiz: Network security
Video
Lab Intro
Hands-On Lab
Configuring Google Kubernetes Engine (GKE) Networking
Video
Lab Intro
Hands-On Lab
Creating Services and Ingress Resources
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and
2. What are the requirements for enabling network security policies? Choose all answers that are correct (3 correct answers).
All nodes must be of machine type n1-standard-1 or higher.
That is correct.
You must install a software firewall module on each node.
You must have at least 2 nodes.
That is correct.
You must restart your nodes.
That is correct.
You must have enabled container-native load balancing.
Chat


Configuring Google Kubernetes
Google Cloud Platform
https://console.cloud.google.com/kubernetes/list?cloudshell=true&project=qwiklabs-gcp-03-2ac151202de9
InPrivate
Google Cloud Platform
qwiklabs-gcp-03-2ac151202de9
Search products and resources
Kubernetes Engine
Kubernetes clusters
CREATE CLUSTER
DEPLOY
REFRESH
DELETE
SHOW INFO PANEL
LEARN
Clusters
Workloads
Services & Ingress
Marketplace
A Kubernetes cluster is a managed group of VM instances for running containerized applications. Learn more
Filter by label or name
Name
Location
Cluster size
Total cores
Total memory
Notifications
Labels
private-cluster
us-central1-a
2
4 vCPUs
8.00 GB
Connect
CLOUD SHELL
Terminal
(qwiklabs-gcp-03-2ac151202de9)
Open Editor
version: 1.16.15-gke.4901
privateClusterConfig:
enablePrivateNodes: true
masterIpv4CidrBlock: 172.16.0.0/28
peeringName: gke-n338e33900bffc258d4d-103e-b1b0-peer
privateEndpoint: 172.16.0.2
publicEndpoint: 34.72.209.160
releaseChannel: {}
selfLink: https://container.googleapis.com/v1/projects/qwiklabs-gcp-03-2ac151202de9/zones/us-central1-a/clusters/private-cluster
servicesIpv4Cidr: 10.36.0.0/20
shieldedNodes: {}
status: RUNNING
subnetwork: default
zone: us-central1-a
student_03_c10d2d4f55e9@cloudshell:~ (qwiklabs-gcp-03-2ac151202de9)$

Configuring Google Kubernetes Engine (GKE) Networking

1 hourFree

Rate Lab

Overview

In this lab, you will create a private cluster, add an authorized network for API access to it, and then configure a network policy for Pod security.

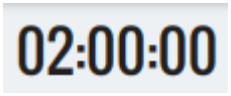
Objectives

In this lab, you learn how to perform the following tasks:

- Create and test a private cluster
- Configure a cluster for authorized network master access
- Configure a Cluster network policy

Access Qwiklabs

For each lab, you get a new GCP project and set of resources for a fixed time at no cost.

1. Make sure you signed into Qwiklabs using an **incognito window**.
2. Note the lab's access time (for example, 

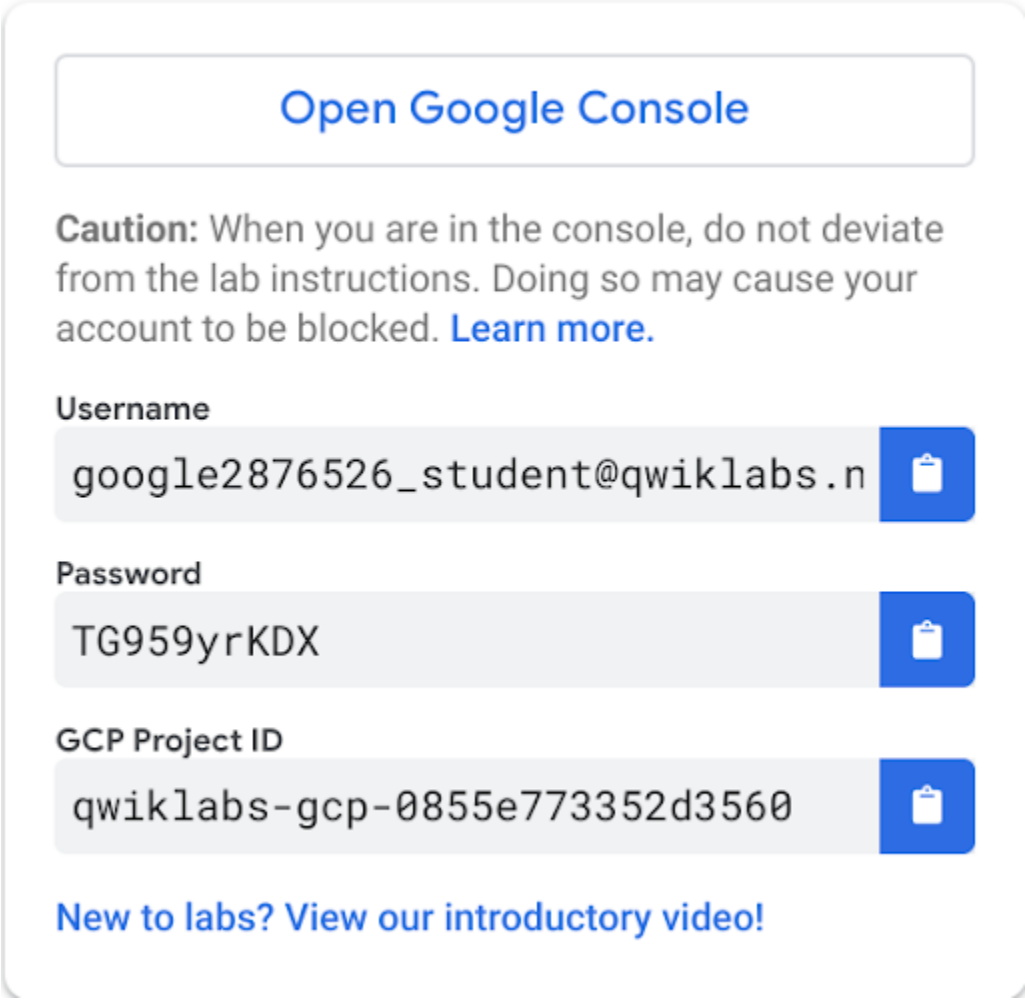
and make sure you can finish in that time block.

There is no pause feature. You can restart if needed, but you have to start at the beginning.

3. When ready, click 

.

4. Note your lab credentials. You will use them to sign in to Cloud Platform Console.



5. Click **Open Google Console**.

6. Click **Use another account** and copy/paste credentials for **this** lab into the prompts.

If you use other credentials, you'll get errors or **incur charges**.

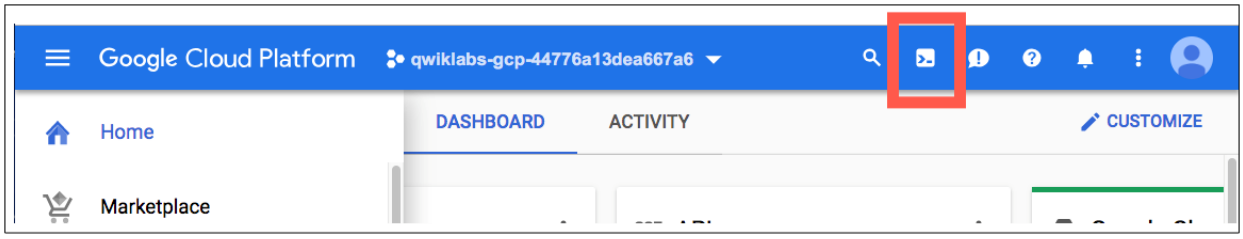
7. Accept the terms and skip the recovery resource page.

Do not click **End Lab** unless you are finished with the lab or want to restart it. This clears your work and removes the project.
After you complete the initial sign-in steps, the project dashboard appears.

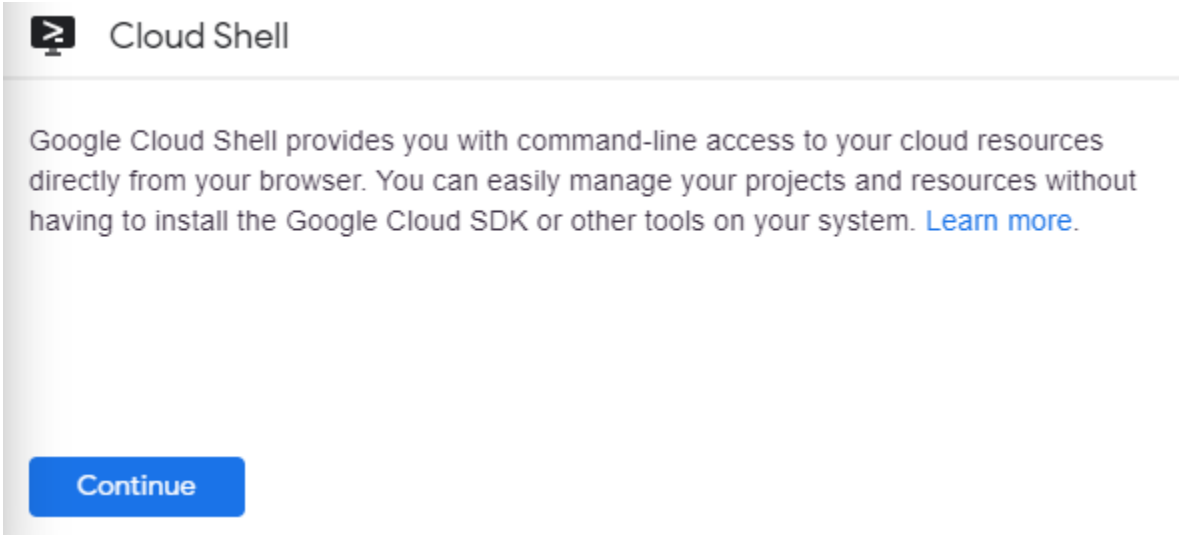
Activate Google Cloud Shell

Google Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Google Cloud Shell provides command-line access to your GCP resources.

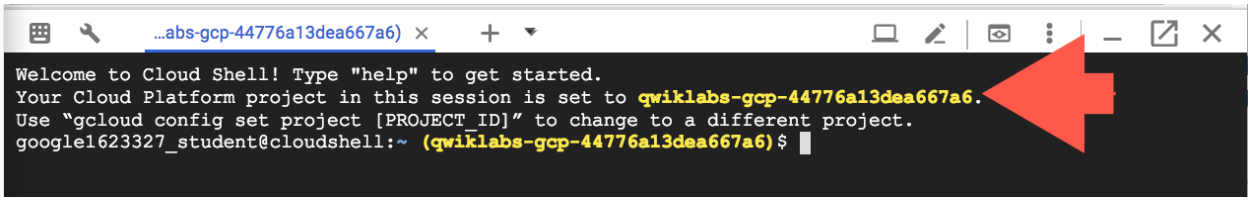
1. In GCP console, on the top right toolbar, click the Open Cloud Shell button.



2. Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT_ID*. For example:



gcloud is the command-line tool for Google Cloud Platform. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

```
gcloud auth list
content_copy
```

Output:

```
Credentialed accounts:
 - <myaccount>@<mydomain>.com (active)content_copy
```

Example output:

```
Credentialed accounts:
 - google1623327_student@qwiklabs.netcontent_copy
```

You can list the project ID with this command:

```
gcloud config list project
content_copy
```

Output:

```
[core]
project = <project_ID>content_copy
```

Example output:

```
[core]
project = qwiklabs-gcp-44776a13dea667a6content_copy
```

Full documentation of **gcloud** is available on Google Cloud gcloud Overview .

Task 1. Create a private cluster

In this task, you create a private cluster, consider the options for how private to make it, and then compare your private cluster to your original cluster.

In a private cluster, the nodes have internal RFC 1918 IP addresses only, which ensures that their workloads are isolated from the public Internet. The nodes in a non-private cluster have external IP addresses, potentially allowing traffic to and from the internet.

Set up a private cluster

1. On the **Navigation menu** (☰), click **Kubernetes Engine > Clusters**.
2. Click **Create cluster**.
3. Name the cluster `private-cluster`.
4. Select `us-central1-a` as the zone.
5. Click on **default-pool** under **NODE POOLS** section and then enter `2` in **Number of nodes** section.
6. Click on **Networking** section, select **Enable VPC-native traffic routing (uses alias IP)**.
7. In the **Networking** section, select **Private cluster** and select **Access master using its external IP address**.
8. For **Master IP Range**, enter **172.16.0.0/28**.

Note: This step assumes that you don't have this range assigned to a subnet. Check VPC Networks in the Google Cloud Console, and select a different range if necessary. Behind the scenes, Google Cloud uses VPC peering to connect the VPC of the cluster master with your default VPC network.

9. Deselect **Enable master authorized networks**.

This setting allows you to define the range of addresses that can access the cluster master externally. When this checkbox is not selected, you can access `kubectl` only from within the Google Cloud network. In this lab, you will only access `kubectl` through the Google Cloud network but you will modify this setting later.

10. Click **Create**.

Note: You need to wait a few minutes for the cluster deployment to complete.

Inspect your cluster

1. In the Cloud Shell, enter the following command to review the details of your new cluster:

```
gcloud container clusters describe private-cluster --region us-central1-a
```
content_copy

The following values appear only under the private cluster:

- privateEndpoint, an internal IP address. Nodes use this internal IP address to communicate with the cluster master.

- publicEndpoint, an external IP address. External services and administrators can use the external IP address to communicate with the cluster master.
  You have several options to lock down your cluster to varying degrees:

- The whole cluster can have external access.
- The whole cluster can be private.
- The nodes can be private while the cluster master is public, and you can limit which external networks are authorized to access the cluster master.
  Without public IP addresses, code running on the nodes can't access the public internet unless you configure a NAT gateway such as Cloud NAT.

  You might use private clusters to provide services such as internal APIs that are meant only to be accessed by resources inside your network. For example, the resources might be private tools that only your company uses. Or they might be backend services accessed by your frontend services, and perhaps only those frontend services are accessed directly by external customers or users. In such cases, private clusters are a good way to reduce the surface area of attack for your application.

  Click *Check my progress* to verify the objective.

Create a private cluster

Check my progress

Task 2. Add an authorized network for cluster master access

After cluster creation, you might want to issue commands to your cluster from outside Google Cloud. For example, you might decide that only your corporate network should issue commands to your cluster master. Unfortunately, you didn't specify the authorized network on cluster creation.

In this task, you add an authorized network for cluster master access.

In this task, you make the Kubernetes master API accessible to a specific range of network addresses. In a real-world use of GKE, this connection would be used by IT staff and automated processes, not end-users.

1. In the Google Cloud Console **Navigation menu** (☰), click **Kubernetes Engine > Clusters**.
2. Click **private-cluster** to open the Clusters details page.
3. Click on **Edit** button.
4. In the **Master authorized networks** drop-down list, select **Enabled**.
5. Click **Add authorized network**.
6. For **Name**, type the name for the network, use `Corporate`.
7. For **Network**, type a CIDR range that you want to grant whitelisted access to your cluster master. As an example, you can use `192.168.1.0/24`.
8. Click **Done**.

Multiple networks can be added here if necessary, but no more than 50 CIDR ranges.

Outside this lab environment, a practical example might be to whitelist only the public, outside address of your corporate firewall. For example, if your corporate firewall's IP address were 8.8.8.14, you could whilelist access to 8.8.8.14/32.

9. Click **Save** at the bottom of the menu.

Click *Check my progress* to verify the objective.

Add an authorized network for cluster master access

Check my progress

Task 3. Create a cluster network policy

In this task, you create a cluster network policy to restrict communication between the Pods. A zero-trust zone is important to prevent lateral attacks within the cluster when an intruder compromises one of the Pods.

Create a GKE cluster

1. In Cloud Shell, type the following command to set the environment variable for the zone and cluster name.

```
export my_zone=us-central1-a
export my_cluster=standard-cluster-1
content_copy
```

2. Configure kubectl tab completion in Cloud Shell.

```
source <(kubectl completion bash)
content_copy
```

3. In Cloud Shell, type the following command to create a Kubernetes cluster. Note that this command adds the additional flag `--enable-network-policy` to the parameters you have used in previous labs. This flag allows this cluster to use cluster network policies.

```
gcloud container clusters create $my_cluster --num-nodes 3 --enable-ip-alias
--zone $my_zone --enable-network-policy
content_copy
```

4. In Cloud Shell, configure access to your cluster for the kubectl command-line tool, using the following command:

```
gcloud container clusters get-credentials $my_cluster --zone $my_zone
content_copy
```

5. Run a simple web server application with the label `app=hello`, and expose the web application internally in the cluster.

```
kubectl run hello-web --labels app=hello \
  --image=gcr.io/google-samples/hello-app:1.0 --port 8080 --expose
content_copy
```

6. In Cloud Shell enter the following command to clone the repository to the lab Cloud Shell.

```
git clone https://github.com/GoogleCloudPlatform/training-data-analyst
content_copy
```

7. Create a soft link as a shortcut to the working directory.

```
ln -s ~/training-data-analyst/courses/ak8s/v1.1 ~/ak8s
content_copy
```

8. Change to the directory that contains the sample files for this lab.

```
cd ~/ak8s/GKE_Networks/
content_copy
```

Restrict incoming traffic to Pods

A sample NetworkPolicy manifest file called `hello-allow-from-foo.yaml` has been provided for you. This manifest file defines an ingress policy that allows access to Pods labeled `app: hello` from Pods labeled `app: foo`.

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: hello-allow-from-foo
spec:
  policyTypes:
  - Ingress
  podSelector:
    matchLabels:
      app: hello
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: foo
content_copy
```

1. Create an ingress policy.

```
kubectl apply -f hello-allow-from-foo.yaml
content_copy
```

2. Verify that the policy was created.

```
kubectl get networkpolicy
content_copy
```

Output (Do not copy)

```
NAME                   POD-SELECTOR   AGE
hello-allow-from-foo   app=hello      7s
content_copy
```

Validate the ingress policy

1. Run a temporary Pod called `test-1` with the label `app=foo` and get a shell in the Pod.

```
kubectl run test-1 --labels app=foo --image=alpine --restart=Never --rm --
stdin --tty
content_copy
```

Note: The kubectl switches used here in conjunction with the run command are important to note.

`--stdin` (alternatively `-i`) creates an interactive session attached to STDIN on the container.

`--tty` (alternatively `-t`) allocates a TTY for each container in the pod.

`--rm` instructs Kubernetes to treat this as a temporary Pod that will be removed as soon as it completes its startup task. As this is an interactive session it will be removed as soon as the user exits the session.

`--label` (alternatively `-l`) adds a set of labels to the pod.

`--restart` defines the restart policy for the Pod

2. Make a request to the hello-web:8080 endpoint to verify that the incoming traffic is allowed.

```
wget -qO- --timeout=2 http://hello-web:8080
content_copy
```

Output (Do not copy)

```
If you don't see a command prompt, try pressing enter.
/ # wget -qO- --timeout=2 http://hello-web:8080
Hello, world!
Version: 1.0.0
Hostname: hello-web-8b44b849-k96lh
/ #
content_copy
```

3. Type **exit** and press **ENTER** to leave the shell.

4. Now you will run a different Pod using the same Pod name but using a label, `app=other`, that does not match the podSelector in the active network policy. This Pod should not have the ability to access the hello-web application.

```
kubectl run test-1 --labels app=other --image=alpine --restart=Never --rm --
stdin --tty
content_copy
```

5. Make a request to the hello-web:8080 endpoint to verify that the incoming traffic is not allowed.

```
wget -qO- --timeout=2 http://hello-web:8080
content_copy
```

The request times out.

Output (Do not copy)

```
If you don't see a command prompt, try pressing enter.
/ # wget -qO- --timeout=2 http://hello-web:8080
wget: download timed out
/ #
content_copy
```

6. Type **exit** and press ENTER to leave the shell.

Restrict outgoing traffic from the Pods

You can restrict outgoing (egress) traffic as you do incoming traffic. However, in order to query internal hostnames (such as hello-web) or external hostnames (such as www.example.com), you must allow DNS resolution in your egress network policies. DNS traffic occurs on port 53, using TCP and UDP protocols.

The NetworkPolicy manifest file `foo-allow-to-hello.yaml` file that has been provided for you defines a policy that permits Pods with the label `app: foo` to communicate with Pods labeled `app: hello` on any port number, and allows the Pods labeled `app: foo` to communicate to any computer on UDP port 53, which is used for DNS resolution. Without the DNS port open, you will not be able to resolve the hostnames.

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: foo-allow-to-hello
spec:
  policyTypes:
  - Egress
  podSelector:
    matchLabels:
      app: foo
  egress:
  - to:
    - podSelector:
        matchLabels:
          app: hello
  - to:
    ports:
    - protocol: UDP
      port: 53
content_copy
```

1. Create an egress policy.

```
kubectl apply -f foo-allow-to-hello.yaml
content_copy
```

2. Verify that the policy was created.

```
kubectl get networkpolicy
content_copy
```

Output (Do not copy)

```
NAME                   POD-SELECTOR   AGE
foo-allow-to-hello     app=foo        7s
hello-allow-from-foo   app=hello      5m
content_copy
```

Validate the egress policy

1. Deploy a new web application called hello-web-2 and expose it internally in the cluster.

```
kubectl run hello-web-2 --labels app=hello-2 \
  --image=gcr.io/google-samples/hello-app:1.0 --port 8080 --expose
content_copy
```

2. Run a temporary Pod with the `app=foo` label and get a shell prompt inside the container.

```
kubectl run test-3 --labels app=foo --image=alpine --restart=Never --rm --
stdin --tty
content_copy
```

3. Verify that the Pod can establish connections to hello-web:8080.

```
wget -qO- --timeout=2 http://hello-web:8080
content_copy
```

Output (Do not copy)

```
If you don't see a command prompt, try pressing enter.
/ # wget -qO- --timeout=2 http://hello-web:8080
Hello, world!
Version: 1.0.0
Hostname: hello-web
/ #
content_copy
```

4. Verify that the Pod cannot establish connections to hello-web-2:8080.

```
wget -qO- --timeout=2 http://hello-web-2:8080
```
content_copy

This fails because none of the Network policies you have defined allow traffic to Pods labelled `app: hello-2`.

5. Verify that the Pod cannot establish connections to external websites, such as www.example.com.

```
wget -qO- --timeout=2 http://www.example.com
```
content_copy

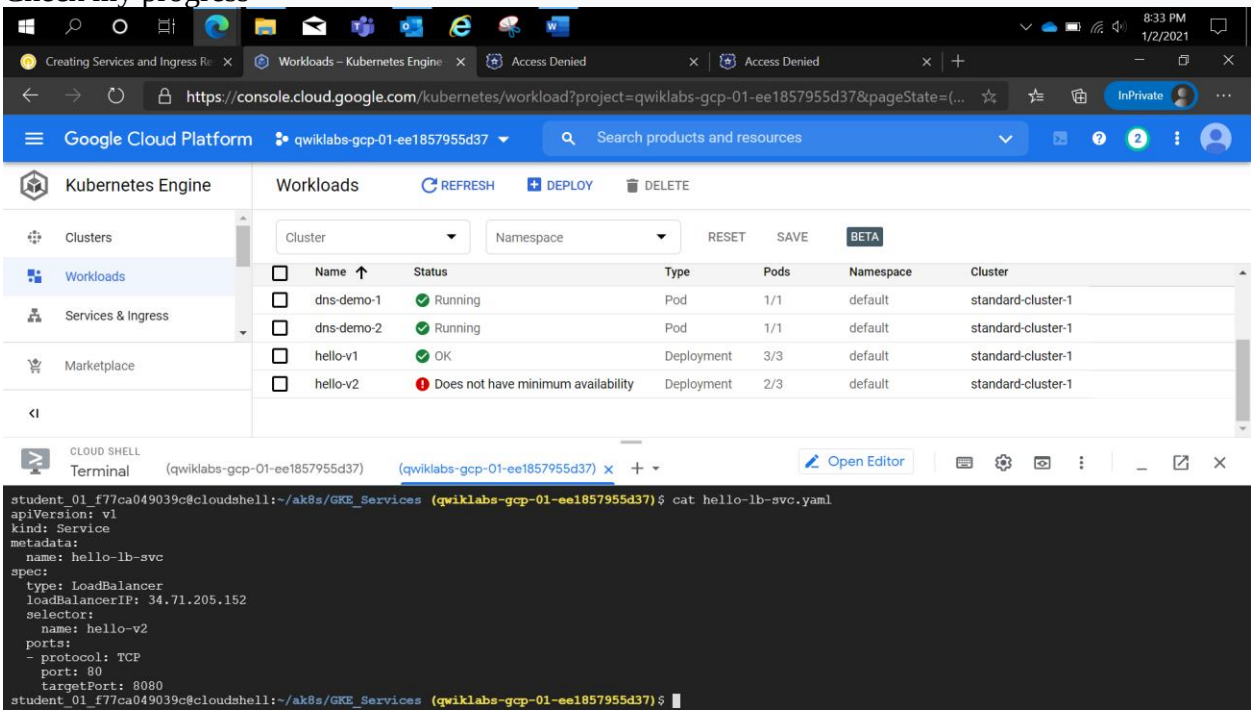
This fails because the network policies do not allow external http traffic (tcp port 80).

6. Type **exit** and press **ENTER** to leave the shell.

Click *Check my progress* to verify the objective.

Run web server applications

Check my progress



Creating Services and Ingress Resources

1 hourFree

Rate Lab

Overview

In this lab, you will create two deployments of Pods and work with three different types of services, including the Ingress resource, and explore how Kubernetes services in GKE are associated with Google Cloud Network Load Balancers.

Objectives

In this lab, you learn how to perform the following tasks:

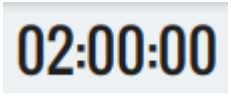
- Observe Kubernetes DNS in action.

- Define various service types (ClusterIP, NodePort, LoadBalancer) in manifests along with label selectors to connect to existing labeled Pods and deployments, deploy those to a cluster, and test connectivity.
- Deploy an Ingress resource that connects clients to two different services based on the URL path entered.
- Verify Google Cloud network load balancer creation for type=LoadBalancer services.

Task 0. Lab Setup

Access Qwiklabs

For each lab, you get a new GCP project and set of resources for a fixed time at no cost.

1. Make sure you signed into Qwiklabs using an **incognito window**.
2. Note the lab's access time (for example, 

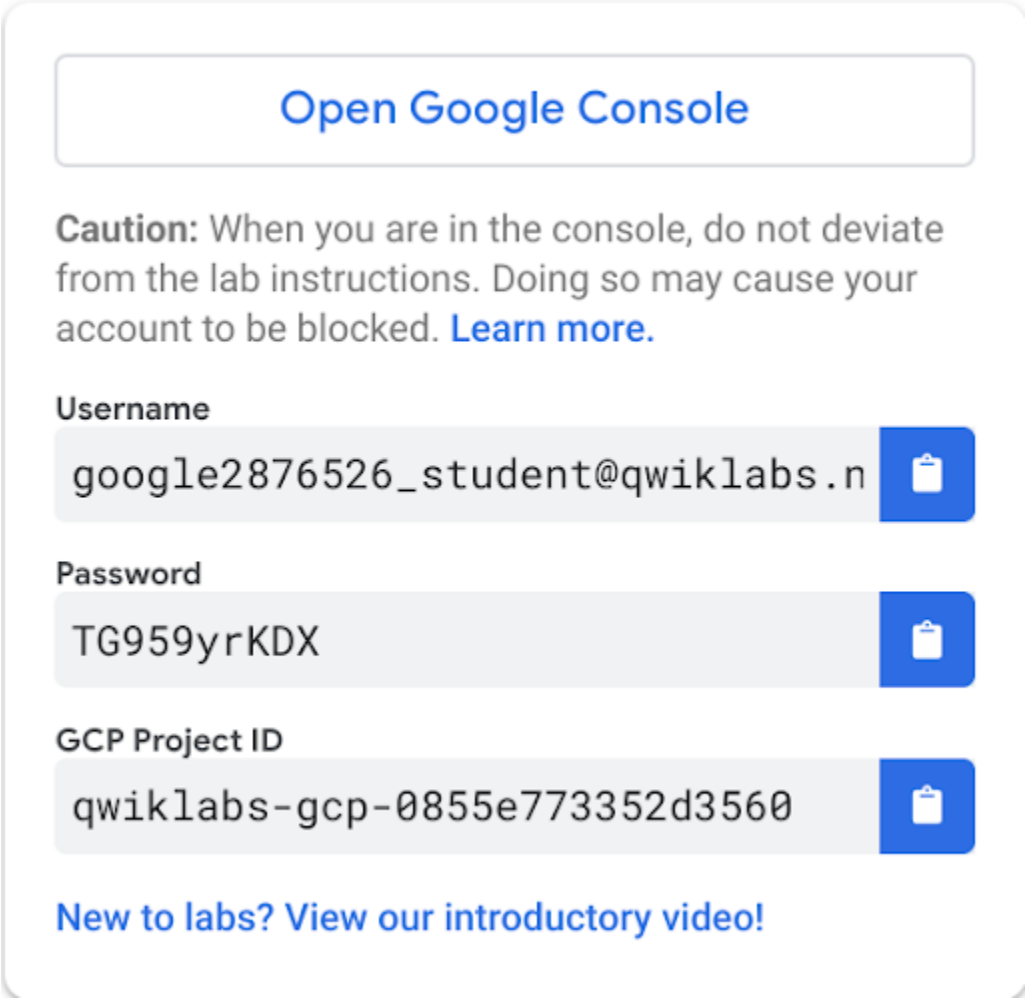
and make sure you can finish in that time block.

There is no pause feature. You can restart if needed, but you have to start at the beginning.

3. When ready, click 

.

4. Note your lab credentials. You will use them to sign in to Cloud Platform Console.



5. Click **Open Google Console**.

6. Click **Use another account** and copy/paste credentials for **this** lab into the prompts.

If you use other credentials, you'll get errors or **incur charges**.

7. Accept the terms and skip the recovery resource page.

Do not click **End Lab** unless you are finished with the lab or want to restart it. This clears your work and removes the project.
After you complete the initial sign-in steps, the project dashboard appears.

Open Cloud Shell

You will do most of the work for this lab in Cloud Shell.

1. On the Google Cloud Console title bar, click **Activate Cloud Shell (**  **)**.
2. Click **Start Cloud Shell**.

After a moment of provisioning, the Cloud Shell prompt appears:

Task 1. Connect to the lab GKE cluster and test DNS

In this task, you connect to the lab GKE cluster and create a deployment manifest for a set of Pods within the cluster that you will then use to test DNS resolution of Pod and service names.

Connect to the lab GKE cluster

1. In Cloud Shell, type the following command to create environment variables for the Google Cloud zone and cluster name that will be used to create the cluster for this lab.

```
export my_zone=us-central1-a
export my_cluster=standard-cluster-1
```
content_copy

2. Configure tab completion for the kubectl command-line tool.

```
source <(kubectl completion bash)
```
content_copy

3. Configure access to your cluster for kubectl:

```
gcloud container clusters get-credentials $my_cluster --zone $my_zone
```
content_copy

Task 2. Create Pods and services to test DNS resolution

You will create a service called `dns-demo` with two sample application Pods called `dns-demo-1` and `dns-demo-2`.

Clone the source file repository and deploy the sample application pods

The pods that you will use to test network connectivity via Kubernetes services are deployed using the `dns-demo.yaml` manifest file that has been provided for you in the source repository. You will use these pods later to test connectivity via various services from systems inside your Kubernetes cluster. You will also test connectivity from external addresses using the Cloud Shell. You can use the Cloud Shell for this because it is on a network that is completely separate to your Kubernetes cluster networks.

```
apiVersion: v1
kind: Service
metadata:
  name: dns-demo
spec:
  selector:
    name: dns-demo
  clusterIP: None
  ports:
  - name: dns-demo
    port: 1234
    targetPort: 1234
---
apiVersion: v1
kind: Pod
metadata:
  name: dns-demo-1
  labels:
    name: dns-demo
spec:
  hostname: dns-demo-1
  subdomain: dns-demo
  containers:
  - name: nginx
    image: nginx
---
apiVersion: v1
```

```
kind: Pod
metadata:
  name: dns-demo-2
  labels:
    name: dns-demo
spec:
  hostname: dns-demo-2
  subdomain: dns-demo
  containers:
  - name: nginx
    image: nginx
```
content_copy

1. In Cloud Shell enter the following command to clone the repository to the lab Cloud Shell.

```
git clone https://github.com/GoogleCloudPlatform/training-data-analyst
```
content_copy

2. Create a soft link as a shortcut to the working directory.

```
ln -s ~/training-data-analyst/courses/ak8s/v1.1 ~/ak8s
```
content_copy

3. Change to the directory that contains the sample files for this lab.

```
cd ~/ak8s/GKE_Services/
```
content_copy

4. Create the service and Pods.

```
kubectl apply -f dns-demo.yaml
```
content_copy

5. Verify that your Pods are running.

```
kubectl get pods
```
content_copy

The output should look like this example.

Output (do not copy)

```
NAME                          READY     STATUS    RESTARTS   AGE
dns-demo-1                    1/1       Running   0          8s
dns-demo-2                    1/1       Running   0          8s
```
content_copy

Click *Check my progress* to verify the objective.

Create Pods and services to test DNS resolution

Check my progress

6. To get inside the cluster, open an interactive session to `bash` running from dns-demo-1.

```
kubectl exec -it dns-demo-1 -- /bin/bash
content_copy
```

Now that you are inside a container in the cluster, subsequent commands will run from that context and you will use this to test network connectivity by pinging various targets in later tasks. However, you don't have a tool to ping in this container, so you need to install the `ping` command first.

7. Update `apt-get` and install a `ping` tool.

```
apt-get update
apt-get install -y iputils-ping
content_copy
```

8. Ping dns-demo-2:

```
ping dns-demo-2.dns-demo.default.svc.cluster.local
content_copy
```

This ping should succeed and report that the target has the ip-address you found earlier for the `dns-demo-2` Pod.

9. Press Ctrl+C to abort the ping command.

10. Ping the `dns-demo` service's FQDN, instead of a specific Pod inside the service:

```
ping dns-demo.default.svc.cluster.local
content_copy
```

This ping should also succeed but it will return a response from the FQDN of one of the two demo-dns Pods. This Pod might be either `demo-dns-1` or `demo-dns-2`.

11. Press Ctrl+C to abort the ping command.

When you deploy applications, your application code runs inside a container in the cluster, and thus your code can access other services by using the FQDNs of those services from inside that cluster. This approach is simpler than using IP addresses or even Pod names, because those are more likely to change.

12. Leave the interactive shell on dns-demo-1 running.

Task 3. Deploy a sample workload and a ClusterIP service

In this task, you create a deployment manifest for a set of Pods within the cluster and then expose them using a ClusterIP service.

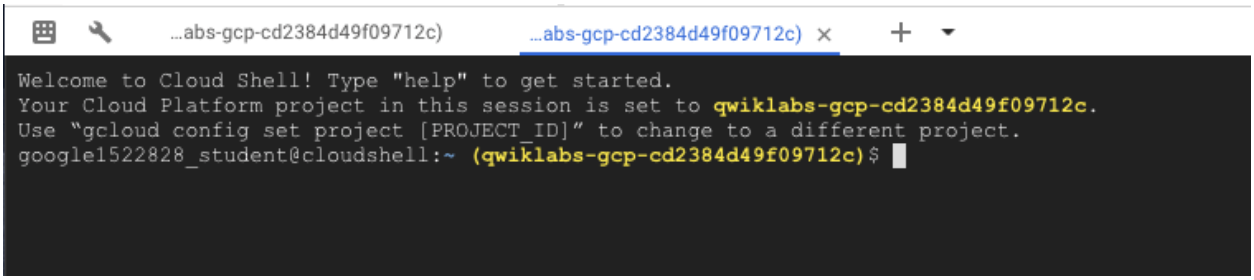
Deploy a sample web application to your Kubernetes Engine cluster

In this task you will deploy a sample web application container image that listens on an HTTP server on port 8080 using the following manifest that has already been created for you in the `hello-v1.yaml` manifest file.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-v1
spec:
  replicas: 3
  selector:
    matchLabels:
      run: hello-v1
  template:
    metadata:
      labels:
        run: hello-v1
        name: hello-v1
    spec:
      containers:
      - image: gcr.io/google-samples/hello-app:1.0
        name: hello-v1
        ports:
        - containerPort: 8080
          protocol: TCP
```

content_copy

1. In the Cloud Shell menu bar, click the plus sign (+) icon to start a new Cloud Shell session.



A second Cloud Shell session appears in your Cloud Shell window. You can switch sessions by clicking them in the menu bar.

2. In the second Cloud Shell tab, change to the directory that contains the sample files for this lab.

```
cd ~/ak8s/GKE_Services/
```
content_copy

3. To create a deployment from the `hello-v1.yaml` file, execute the following command:

```
kubectl create -f hello-v1.yaml
```
content_copy

4. To view a list of deployments, execute the following command:

```
kubectl get deployments
```
content_copy

The output should look like this example.

Output (do not copy)

```
NAME        READY    UP-TO-DATE   AVAILABLE   AGE
hello-v1    3/3      3            3           14s
```
content_copy

Define service types in the manifest

In this task you will deploy a Service using a ClusterIP using the `hello-svc.yaml` sample manifest that has already been created for you.

```
apiVersion: v1
kind: Service
metadata:
```

```
  name: hello-svc
spec:
  type: ClusterIP
  selector:
    name: hello-v1
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8080
```
content_copy

1. To deploy the ClusterIP service, execute the following command:

```
kubectl apply -f ./hello-svc.yaml
```
content_copy

This manifest defines a ClusterIP service and applies it to Pods that correspond to the selector. In this case, the manifest is applied to the hello-v1 Pods that you deployed. This service will automatically be applied to any other deployments with the `name: hello-v1` label.

2. Verify that the service was created and that a Cluster-IP was allocated:

```
kubectl get service hello-svc
```
content_copy

Your IP address might be different from the example output.

Output (do not copy)

```
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
hello-svc    ClusterIP   10.11.253.203   <none>        80/TCP    20s
```
content_copy

No external IP is allocated for this service. Because the Kubernetes Cluster IP addresses are not externally accessible by default, creating this service does not make your application accessible outside of the cluster.

Click *Check my progress* to verify the objective.

Deploy a sample workload and a ClusterIP service

Check my progress

Test your application

1. In the Cloud Shell, attempt to open an HTTP session to the new service using the following command:

```
curl hello-svc.default.svc.cluster.local
content_copy
```

The connection should fail because that service is not exposed outside of the cluster.

Output (do not copy)

```
curl: (6) Could not resolve host: hello-svc.default.svc.cluster.local
content_copy
```

Now you will test the service from inside the cluster using the interactive shell you have running on the `dns-demo-1` Pod.

2. Return to your first Cloud Shell window, which is currently redirecting the stdin and stdout of the `dns-demo-1` Pod.

3. Install `curl` so you can make calls to web services from the command line.

```
apt-get install -y curl
content_copy
```

4. Use the following command to test the HTTP connection between the Pods.

```
curl hello-svc.default.svc.cluster.local
content_copy
```

This connection should succeed and provide a response similar to the output below. Your hostname might be different from the example output.

Output (do not copy)

```
root@dns-demo-1:/# curl hello-svc.default.svc.cluster.local
Hello, world!
Version: 1.0.0
Hostname: hello-v1-85b99869f-mp4vd
root@dns-demo-1:/#
content_copy
```

This connection works because the clusterIP can be resolved using the internal DNS within the Kubernetes Engine cluster.

Task 4. Convert the service to use NodePort

In this task, you will convert your existing ClusterIP service to a NodePort service and then retest access to the service from inside and outside the cluster.

A modified version of the hello-svc.yaml file called `hello-nodeport-svc.yaml` that changes the service type to NodePort has already been created for you.

```
apiVersion: v1
kind: Service
metadata:
  name: hello-svc
spec:
  type: NodePort
  selector:
    name: hello-v1
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8080
    nodePort: 30100
```
content_copy

1. Return to your second Cloud Shell window. This is the window NOT connected to the stdin and stdout of the dns-test Pod.

2. To deploy the manifest that changes the service type for the `hello-svc` to NodePort , execute the following command:

```
kubectl apply -f ./hello-nodeport-svc.yaml
```
content_copy

This manifest redefines `hello-svc` as a NodePort service and assigns the service port 30100 on each node of the cluster for that service.

3. Enter the following command to verify that the service type has changed to NodePort:

```
kubectl get service hello-svc
```
content_copy

Your IP address might be different from the example output.

Output (do not copy)

```
NAME         TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)         AGE
hello-svc    NodePort   10.11.253.203   <none>        80:30100/TCP    59m
content_copy
```

Note that there is still no external IP allocated for this service.

Click *Check my progress* to verify the objective.

Convert the service to use NodePort

Check my progress

Test your application

1. In the Cloud Shell, attempt to open an HTTP session to the new service using the following command:

```
curl hello-svc.default.svc.cluster.local
content_copy
```

The connection should fail because that service is not exposed outside of the cluster.

Output (do not copy)

```
curl: (6) Could not resolve host: hello-svc.default.svc.cluster.local
content_copy
```

Now you will test the service from another Pod.

2. Return to your first Cloud Shell window, which is currently redirecting the stdin and stdout of the dns-test Pod.

3. Use the following command to test the HTTP connection between the Pods.

```
curl hello-svc.default.svc.cluster.local
content_copy
```

This connection should succeed and provide a response similar to the output below. Your hostname might be different from the example output.

Output (do not copy)

```
root@dns-demo-1:/# curl hello-svc.default.svc.cluster.local
```

```
Hello, world!
Version: 1.0.0
Hostname: hello-v1-85b99869f-mp4vd
root@dns-demo-1:/#
content_copy
```

This connection works because the clusterIP can be resolved using the internal DNS within the GKE cluster.

Task 5. Create static public IP addresses using Google Cloud Networking

Reserve Google Cloud Networking Static IP Addreesses

1. In the Google Cloud Console navigation menu, navigate to **Networking** > **VPC Network** > **External IP Addresses**.

2. Click **+ Reserve Static Address**.

3. Enter `regional-loadbalancer` for the **Name**.

4. Explore the options but leave the remaining settings at their defaults. Note that the default type is **Regional**.

Note that the default region must match the region where you have deployed your GKE cluster. If this is not the case then change the region here to match the region you used to create your cluster. If you accepted the lab defaults then that should be the us-central1 (Iowa) region.

5. Click **Reserve**.

6. Make a note of the external ip-address called `regional-loadbalancer`. You will use this in a later task.

7. Click **+ Reserve Static Address**.

8. Enter `global-ingress` for the **Name**.

9. Change the Type to **Global**.

10. Leave the remaining settings at their defaults.

11. Click **Reserve**.

12. Make a note of the external ip-address called `global-ingress`. You will use this in a later task.

Click *Check my progress* to verify the objective.

Create static public IP addresses using Google Cloud Networking

Check my progress

Task 6. Deploy a new set of Pods and a LoadBalancer service

You will now deploy a new set of Pods running a different version of the application so that you can easily differentiate the two services. You will then expose the new Pods as a LoadBalancer service and access the service from outside the cluster.

A second Deployment manifest called `hello-v2.yaml` has been created for you that creates a new deployment that runs version 2 of the sample hello application on port 8080.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-v2
spec:
  replicas: 3
  selector:
    matchLabels:
      run: hello-v2
  template:
    metadata:
      labels:
        run: hello-v2
        name: hello-v2
    spec:
      containers:
      - image: gcr.io/google-samples/hello-app:2.0
        name: hello-v2
        ports:
        - containerPort: 8080
          protocol: TCP
```
content_copy

1. Return to your second Cloud Shell window. This is the window NOT connected to the stdin and stdout of the dns-test Pod.

2. To deploy the manifest that creates the `hello-v2` deployment execute the following command:

```
kubectl create -f hello-v2.yaml
```
content_copy

3. To view a list of deployments, execute the following command:

```
kubectl get deployments
```
content_copy

The output should look like this example.

Output (do not copy)

```
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
hello-v1   3/3     3            3           18m
hello-v2   3/3     3            3           7s
```
content_copy

Define service types in the manifest

In this task you will deploy a LoadBalancer Service using the `hello-lb-svc.yaml` sample manifest that has already been created for you.

You will use the sed command to replace the 10.10.10.10 placeholder address in the load balancer yaml file with the static address you reserved earlier for the load balancer.

```
apiVersion: v1
kind: Service
metadata:
  name: hello-lb-svc
spec:
  type: LoadBalancer
  loadBalancerIP: 10.10.10.10
  selector:
    name: hello-v2
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8080
```
content_copy

1. Make sure you are still in the second Cloud Shell window. This is the window NOT connected to the stdin and stdout of the dns-test Pod.

2. Enter the following command in the Cloud Shell to save the regional static IP-address you created earlier into an environment variable.

```
export STATIC_LB=$(gcloud compute addresses describe regional-loadbalancer --region us-central1 --format json | jq -r '.address')
```
content_copy

3. Enter the following command in the Cloud Shell to replace the placeholder address with the regional static IP-address.

```
sed -i "s/10\.10\.10\.10/$STATIC_LB/g" hello-lb-svc.yaml
```
content_copy

4. To check that the external ip-address has been replaced correctly enter the following command in the Cloud shell.

```
cat hello-lb-svc.yaml
```
content_copy

The `loadBalancerIP` should match the address you recorded earlier for the `regional-loadbalancer` reserved static ip-address.

5. To deploy your LoadBalancer service manifest, execute the following command:

```
kubectl apply -f ./hello-lb-svc.yaml
content_copy
```

This manifest defines a LoadBalancer service, which deploys a Google Cloud Network Load Balancer to provide external access to the service. This service is only applied to the Pods with the `name: hello-v2` selector.

6. Verify that the service was created and that a node port was allocated:

```
kubectl get services
content_copy
```

Your IP address might be different from the example output.

Output (do not copy)

```
NAME          TYPE           CLUSTER-IP     EXTERNAL-IP    PORT(S)         AGE
dns-demo      ClusterIP      None           <none>         1234/TCP        1h
hello-lb-svc LoadBalancer  10.11.253.48   35.184.45.240  80:30103/TCP    2m
hello-svc     NodePort       10.11.253.203  <none>         80:30100/TCP    1h
kubernetes    ClusterIP      10.11.240.1    <none>         443/TCP         2h
content_copy
```

Notice that the new LoadBalancer service lists the `regional-loadbalancer` reserved address as its external IP address. In GKE environments the external load balancer for the LoadBalancer service type is implemented using a Google Cloud load balancer and will take a few minutes to create. This external IP address makes the service accessible from outside the cluster.

Confirm that a Regional Google Cloud Load Balancer has been created

1. In the Google Cloud Console Navigate to **Networking** > **Network Services** > **Load Balancing**

2. You should see a TCP Load Balancer listed with 1 target pool backend and 2 instances. Click the name to open the details page.

3. The details should show that it is a regional TCP load balancer using the static IP address you called `regional-loadbalancer` when you reserved it earlier.

Click *Check my progress* to verify the objective.

Deploy a new set of Pods and a LoadBalancer service

Check my progress

Test your application

1. In the Cloud Shell, attempt to open an HTTP session to the new service using the following command:

```
curl hello-lb-svc.default.svc.cluster.local
```
content_copy

The connection should fail because that service name is not exposed outside of the cluster.

Output (do not copy)

```
curl: (6) Could not resolve host: hello-lb-svc.default.svc.cluster.local
```
content_copy

This occurs because the external IP address is not registered with this hostname..

2. Try the connection again using the External IP address associated with the REgional Load Balancer service. Type the following command and substitute `[external_IP]` with your service's external IP address.

```
curl [external_IP]
```
content_copy

Example (do not copy)

```
curl 35.184.45.240
```
content_copy

This time the connection does not fail because the LoadBalancer's external ip-address can be reached from outside Google Cloud.

Note: This may take up to 5 minutes to become available.

Output (do not copy)

```
Hello, world!
Version: 2.0.0
Hostname: hello-v2-59c99d8b65-jrx2d
content_copy
```

3. Return to your first Cloud Shell window, which is currently redirecting the stdin and stdout of the `dns-demo-1` Pod.

4. Use the following command to test the HTTP connection between the Pods.

```
curl hello-lb-svc.default.svc.cluster.local
content_copy
```

This connection should succeed and provide a response similar to the output below. Your hostname will be different from the example output.

Output (do not copy)

```
root@dns-demo-1:/# curl hello-lb-svc.default.svc.cluster.local
Hello, world!
Version: 2.0.0
Hostname: hello-v2-59c99d8b65-78ggz
root@dns-demo-1:/#
content_copy
```

The internal DNS name works within the Pod, and you can see that you are accessing the same v2 version of the application as you were from outside of the cluster using the external ip address.

5. Try the connection again within the Pod using the External IP address associated with the service. Type the following command and substitute `[external_IP]` with your service's external IP address.

```
curl [external_IP]
content_copy
```

Example (do not copy)

```
curl 35.184.45.240
content_copy
```

Output (do not copy)

```
root@dns-demo-1:/# curl 35.184.45.240
Hello, world!
Version: 2.0.0
Hostname: hello-v2-59c99d8b65-jrx2d
root@dns-demo-1:/#
```

```
content_copy
```

The external IP also works from inside Pods running in the cluster, and returns a result from the same v2 version of the applications.

4. Exit the console redirection session to the Pod by typing exit.

```
exit
content_copy
```

This will return you to the Cloud Shell command prompt.

Task 7. Deploy an Ingress resource

You have two services in your cluster for the hello application. One service is hosting version 1.0 via a NodePort service, while the other service is hosting version 2.0 via a LoadBalancer service. You will now deploy an Ingress resource that will direct traffic to both services based on the URL entered by the user.

Create an ingress resource

Ingress is a Kubernetes resource that encapsulates a collection of rules and configuration for routing external HTTP(S) traffic to internal services.

On GKE, Ingress is implemented using Cloud Load Balancing. When you create an ingress resource in your cluster, GKE creates an HTTP(S) load balancer and configures it to route traffic to your application.

A sample manifest called `hello-ingress.yaml` that will configure an ingress resource has been created for you.

```
apiVersion: app/v1
kind: Ingress
metadata:
```

```
  name: hello-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
    kubernetes.io/ingress.global-static-ip-name: "global-ingress"

spec:
  rules:
  - http:
      Paths:
     - path: /v1
        backend:
          serviceName: hello-svc
          servicePort: 80
      - path: /v2
        backend:
          serviceName: hello-lb-svc
          servicePort: 80
content_copy
```

This configuration file defines an ingress resource that will attach to the `global-ingress` static ip-address you created in a previous step. This ingress resource will create a global HTTP(S) load balancer that directs traffic to your web services based on the path entered.

The `kubernetes.io/ingress.global-static-ip-name` annotation allows you to specify a named resxerved ip-address that the Ingress resource will use when creating the Google Cloud Global HTTP(S) load balancer for the Ingress resource.

1. To deploy this ingress resource, execute the following command:

```
kubectl apply -f hello-ingress.yaml
content_copy
```

When you deploy this manifest, Kubernetes creates an ingress resource on your cluster. The ingress controller running in your cluster is responsible for creating an HTTP(S) load balancer to route all external HTTP traffic (on port 80) to the web NodePort service and the LoadBalancer service that you exposed.

Confirm that a Global HTTP(S) Load Balancer has been created

1. In the Google Cloud Console Navigate to **Networking** > **Network Services** > **Load Balancing**

2. You should now also see a HTTP(S) Load Balancer listed. Click the name to open the details page.

3. The details should show that it is a global HTTP(S) load balancer using the static IP address you called `global-ingress` when you reserved it earlier.

Click *Check my progress* to verify the objective.

Deploy an Ingress resource

Check my progress

Test your application

1. To get the external IP address of the load balancer serving your application, execute the following command:

```
kubectl describe ingress hello-ingress
```
content_copy

Partial output (do not copy)

```
Name:             hello-ingress
Namespace:        default
Address:          35.244.173.44
Default backend:  default-http-backend:80 (10.8.2.5:8080)
Rules:
  Host  Path  Backends
  ----  ----  --------
 *
       /v1   hello-svc:80 (<none>)
       /v2   hello-lb-svc:80 (<none>)
[...]
  ingress.kubernetes.io/backends:{"k8s-[...]"HEALTHY"[...]}
[...]

Events:
 Type   Reason Age From                    Message
 ----   ------ --- ----                    -------
 Normal ADD    5m  loadbalancer-controller default/hello-ingress
 Normal CREATE 5m  loadbalancer-controller ip: 35.244.173.44
```
content_copy

Note: You need to wait for a few minutes for the load balancer to become active, and for the health checks to succeed, before the external address will be displayed. Repeat the command every few minutes to check if the Ingress resource has finished initializing. When the Google Cloud global HTTP(S) load balancer has

been created and initialized the command will report the external ip-address which will match the global static ip-address you reserved earlier that you called `global-ingress`.

2. Use the External IP address associated with the Ingress resource, and type the following command, substituting `[external_IP]` with the Ingress resource's external IP address. Be sure to include the /v1 in the URL path.

```
curl http://[external_IP]/v1
content_copy
```

Example (do not copy)

```
curl http://35.201.92.69/v1
content_copy
```

Output (do not copy)

```
Hello, world!
Version: 1.0.0
Hostname: hello-v1-85b99869f-4rmqw
content_copy
```

The v1 URL is configured in `hello-ingress.yaml` to point to the `hello-svc` NodePort service that directs traffic to the v1 application Pods. .

Note: GKE might take a few minutes to set up forwarding rules until the Global load balancer used for the Ingress resource is ready to serve your application. In the meantime, you might get errors such as HTTP 404 or HTTP 500 until the load balancer configuration is propagated across the globe.

3. Now test the v2 URL path from Cloud Shell. Use the External IP address associated with the Ingress resource, and type the following command, substituting `[external_IP]` with the Ingress resource's external IP address. Be sure to include the /v2 in the URL path.

```
curl http://[external_IP]/v2
content_copy
```

The v2 URL is configured in `hello-ingress.yaml` to point to the `hello-lb-svc` LoadBalancer service that directs traffic to the v2 application Pods.

Example (do not copy)

```
curl http://35.201.92.69/v2
content_copy
```

Output (do not copy)

```
Hello, world!
```

```
Version: 2.0.0
Hostname: hello-v2-59c99d8b65-jrx2d
content_copy
```

Inspect the changes to your networking resources in the Google Cloud Console

1. In Google Cloud Console, on the **Navigation menu** , click **Networking** > **Network services**> **Load balancing**.

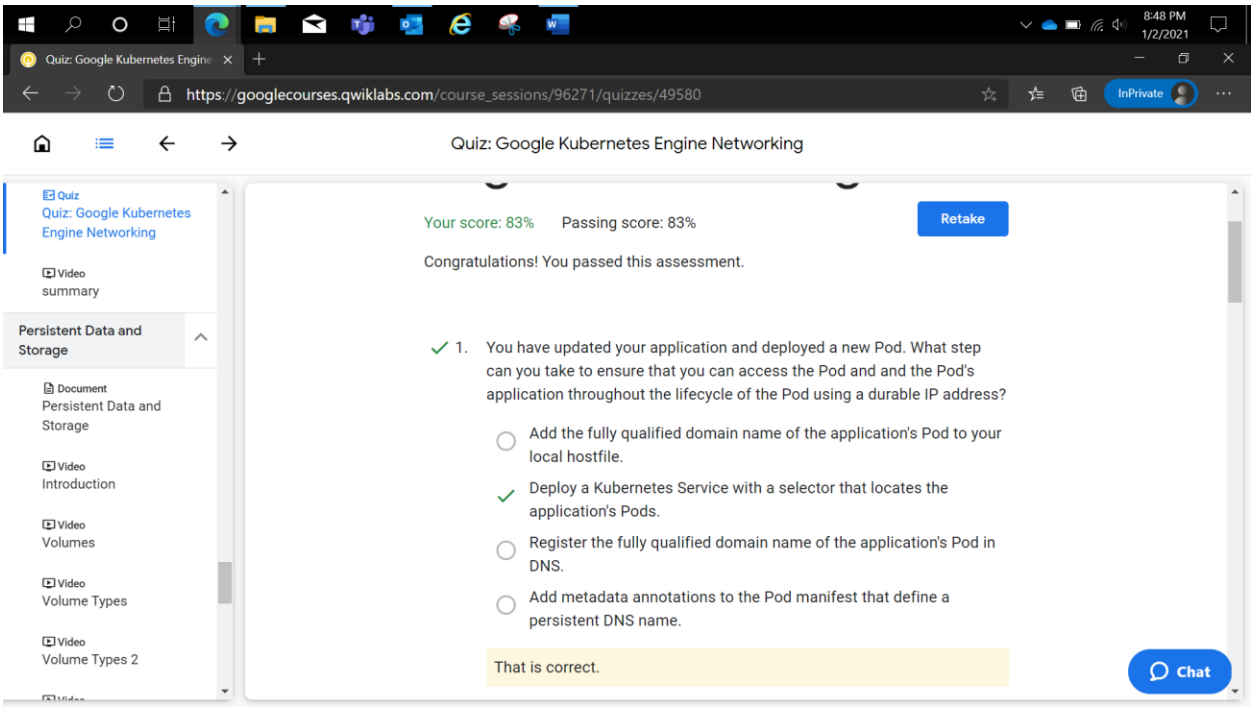
There are now two load balancers listed:

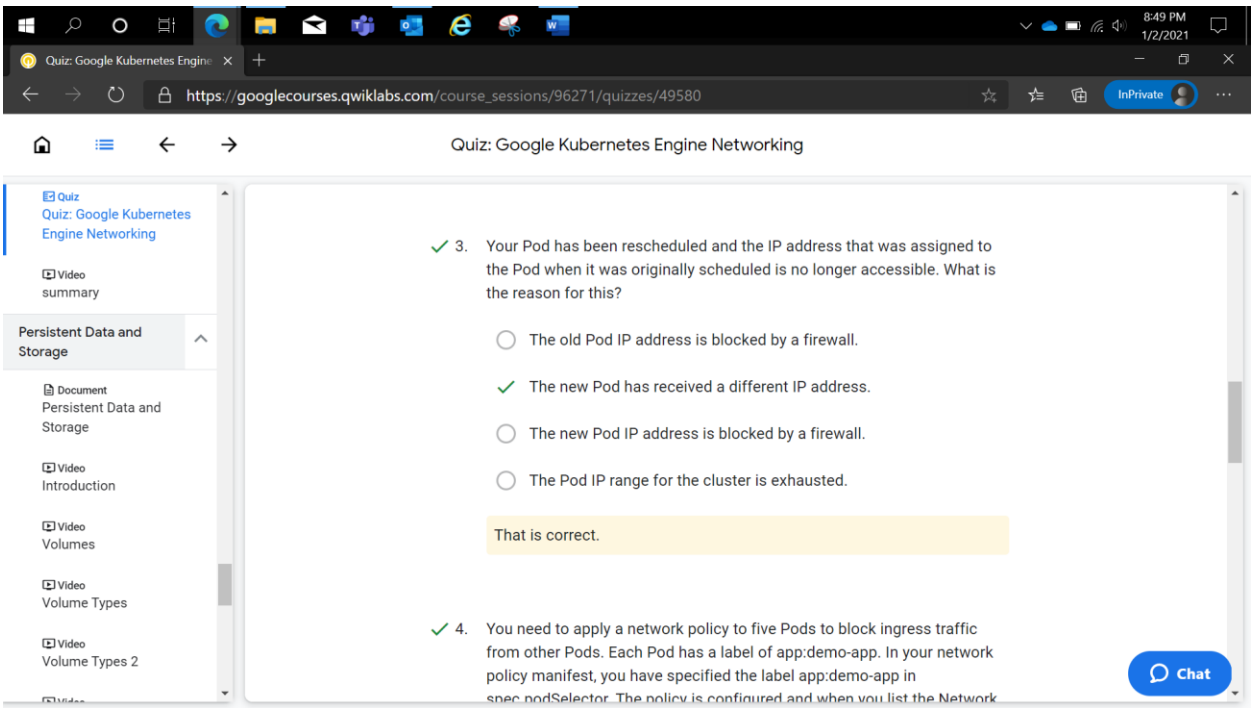
There is the initial regional load balancer for the `hello-lb-svc` service. This has a UID style name and is configured to load balance TCP port 80 traffic to the cluster nodes.

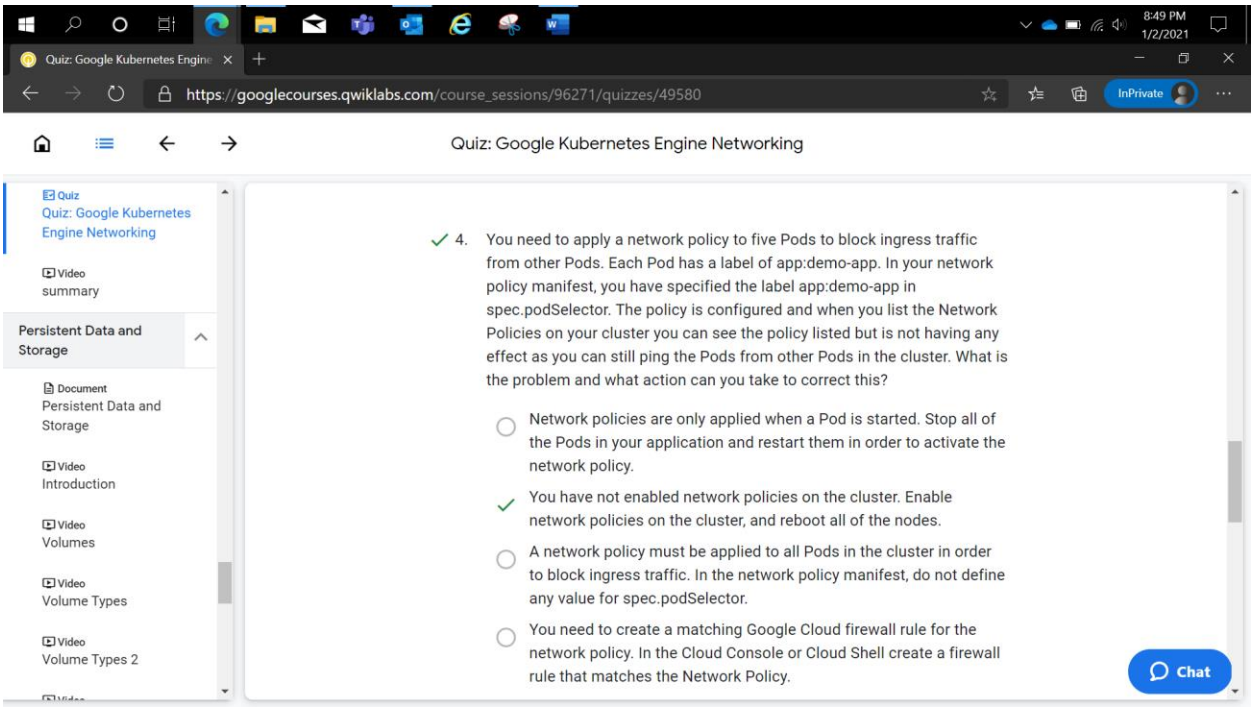
The second was created for the Ingress object and is a full HTTP(S) load balancer that includes host and path rules that match the Ingress configuration. This will have `hello-ingress` in its name.

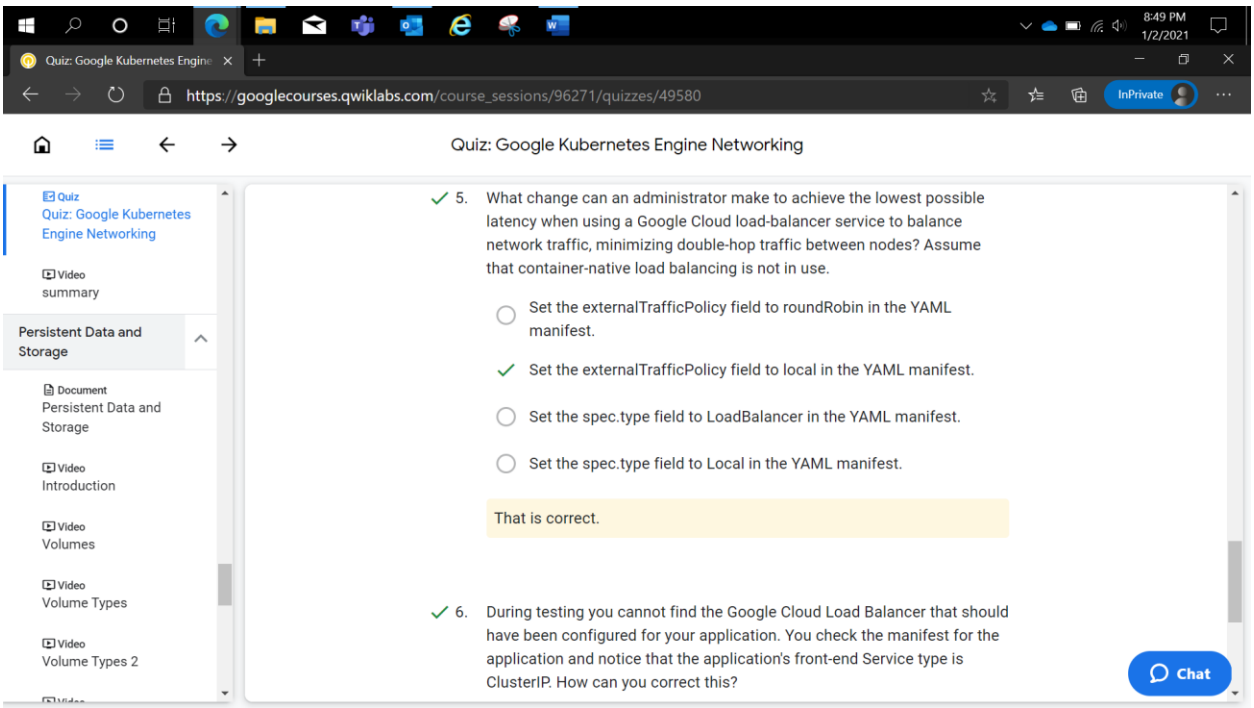
2. Click the load balancer with `hello-ingress` in the name.

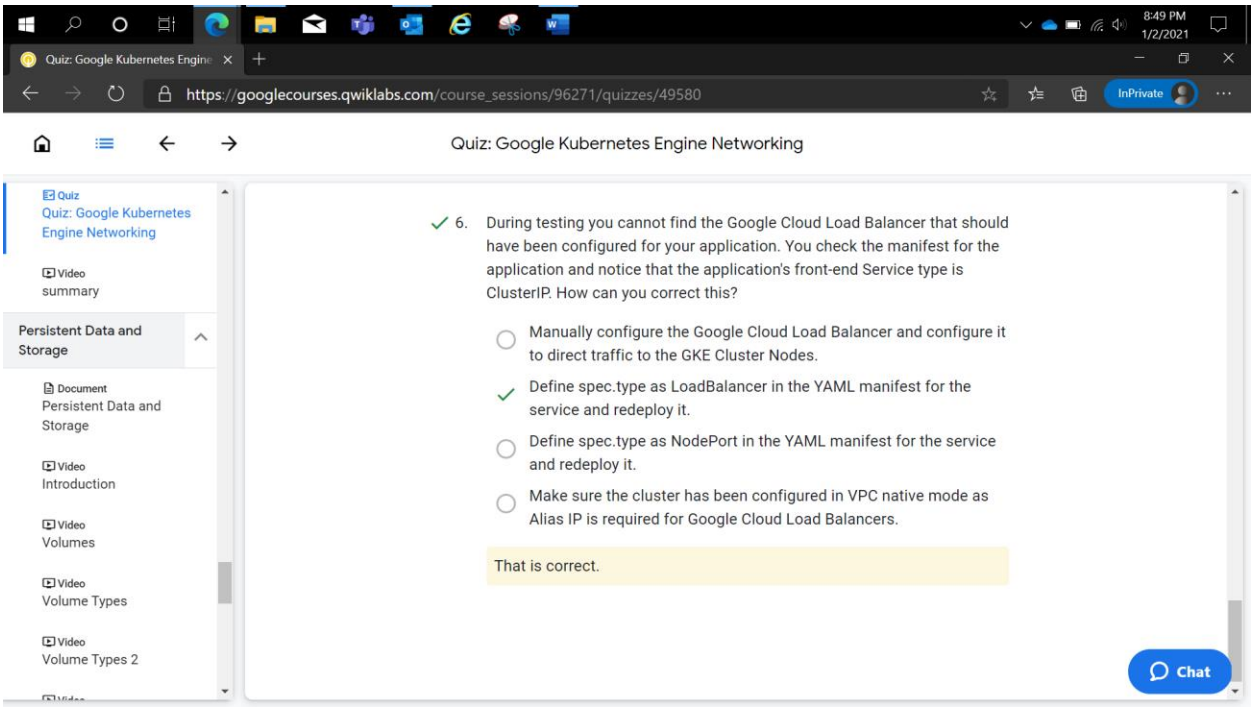
This will display the summary information about the protocols, ports, paths and backend services of the Google Cloud Global HTTP(S) load balancer created for your Ingress resource.

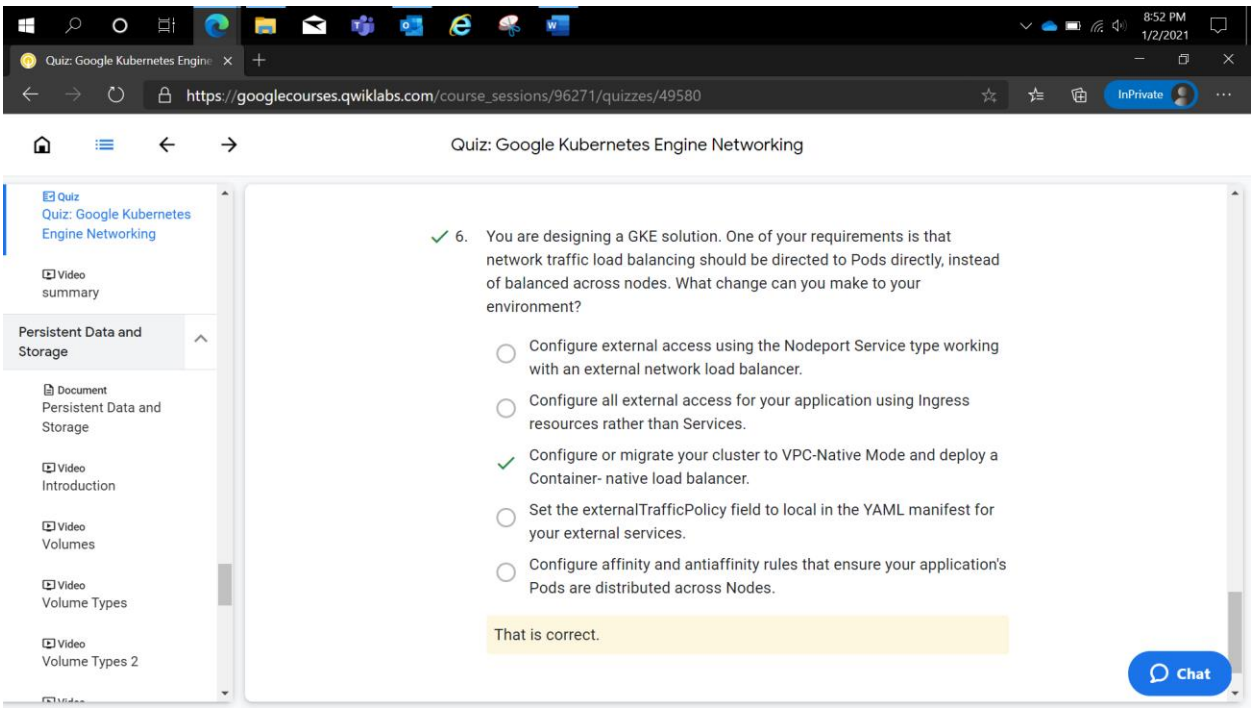

8:48 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
Your score: 83%
Passing score: 83%
Retake
Congratulations! You passed this assessment.
1. You have updated your application and deployed a new Pod. What step can you take to ensure that you can access the Pod and and the Pod's application throughout the lifecycle of the Pod using a durable IP address?
Add the fully qualified domain name of the application's Pod to your local hostfile.
Deploy a Kubernetes Service with a selector that locates the application's Pods.
Register the fully qualified domain name of the application's Pod in DNS.
Add metadata annotations to the Pod manifest that define a persistent DNS name.
That is correct.
Chat


8:49 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
3. Your Pod has been rescheduled and the IP address that was assigned to the Pod when it was originally scheduled is no longer accessible. What is the reason for this?
The old Pod IP address is blocked by a firewall.
The new Pod has received a different IP address.
The new Pod IP address is blocked by a firewall.
The Pod IP range for the cluster is exhausted.
That is correct.
4. You need to apply a network policy to five Pods to block ingress traffic from other Pods. Each Pod has a label of app:demo-app. In your network policy manifest, you have specified the label app:demo-app in
Chat


8:49 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
✓ 4. You need to apply a network policy to five Pods to block ingress traffic from other Pods. Each Pod has a label of app:demo-app. In your network policy manifest, you have specified the label app:demo-app in spec.podSelector. The policy is configured and when you list the Network Policies on your cluster you can see the policy listed but is not having any effect as you can still ping the Pods from other Pods in the cluster. What is the problem and what action can you take to correct this?
Network policies are only applied when a Pod is started. Stop all of the Pods in your application and restart them in order to activate the network policy.
✓ You have not enabled network policies on the cluster. Enable network policies on the cluster, and reboot all of the nodes.
A network policy must be applied to all Pods in the cluster in order to block ingress traffic. In the network policy manifest, do not define any value for spec.podSelector.
You need to create a matching Google Cloud firewall rule for the network policy. In the Cloud Console or Cloud Shell create a firewall rule that matches the Network Policy.
Chat


8:49 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
✓ 5. What change can an administrator make to achieve the lowest possible latency when using a Google Cloud load-balancer service to balance network traffic, minimizing double-hop traffic between nodes? Assume that container-native load balancing is not in use.
Set the externalTrafficPolicy field to roundRobin in the YAML manifest.
✓ Set the externalTrafficPolicy field to local in the YAML manifest.
Set the spec.type field to LoadBalancer in the YAML manifest.
Set the spec.type field to Local in the YAML manifest.
That is correct.
✓ 6. During testing you cannot find the Google Cloud Load Balancer that should have been configured for your application. You check the manifest for the application and notice that the application's front-end Service type is ClusterIP. How can you correct this?
Chat


8:49 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
✓ 6. During testing you cannot find the Google Cloud Load Balancer that should have been configured for your application. You check the manifest for the application and notice that the application's front-end Service type is ClusterIP. How can you correct this?
Manually configure the Google Cloud Load Balancer and configure it to direct traffic to the GKE Cluster Nodes.
Define spec.type as LoadBalancer in the YAML manifest for the service and redeploy it.
Define spec.type as NodePort in the YAML manifest for the service and redeploy it.
Make sure the cluster has been configured in VPC native mode as Alias IP is required for Google Cloud Load Balancers.
That is correct.
Chat


8:52 PM
1/2/2021
Quiz: Google Kubernetes Engine
https://googlecourses.qwiklabs.com/course_sessions/96271/quizzes/49580
InPrivate
Quiz: Google Kubernetes Engine Networking
Quiz
Quiz: Google Kubernetes Engine Networking
Video
summary
Persistent Data and Storage
Document
Persistent Data and Storage
Video
Introduction
Video
Volumes
Video
Volume Types
Video
Volume Types 2
✓ 6. You are designing a GKE solution. One of your requirements is that network traffic load balancing should be directed to Pods directly, instead of balanced across nodes. What change can you make to your environment?
Configure external access using the Nodeport Service type working with an external network load balancer.
Configure all external access for your application using Ingress resources rather than Services.
Configure or migrate your cluster to VPC-Native Mode and deploy a Container- native load balancer.
Set the externalTrafficPolicy field to local in the YAML manifest for your external services.
Configure affinity and antiaffinity rules that ensure your application's Pods are distributed across Nodes.
That is correct.
Chat