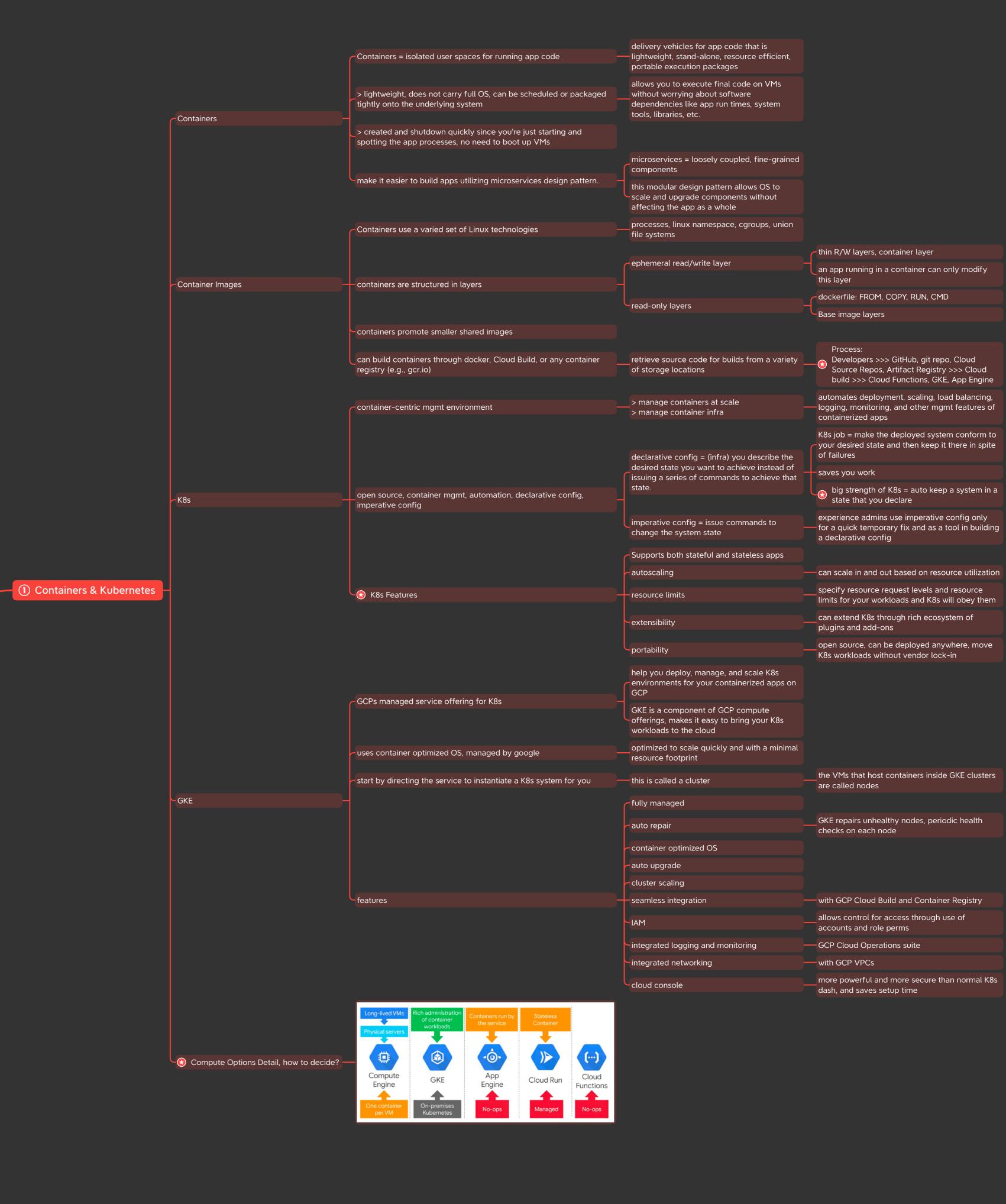
many command uses: create K8s objects, view objects, delete objects, view and export configs	use kubectl to see a list of pods in a cluster	transforms CLI entries into API calls		
	relies on config file \$HOME/.kube/config	- kubectl must be configured first	kubectl command	
what do you want to do on what type of object	— [command] — [TYPE]	— syntax has several parts		
	— [NAME] — [flags]	Syritax has several parts		
	roll out updates to the pods, roll back pods to previous revision, scale or autoscale pods,	— declare the state of pods))	
	well-suited for stateless apps .yaml file >>> deployment object >>>	— deployment = 2 step process		
	deployment controller >>> node progressing , complete, failed	— 3 different lifecycles states		
	can output deployment config in a YAML format	— use kubectl to inspect your deployment —	deployments	
	1, create deployment declaratively using a manifest file such as the YAML file you've			
	just seen and kubectl apply command 2. (imperatively) use kubectl run command	— 3 ways to create deployment		
	that specifies the parameters inline 3. use GKE workloads menu in GCP console	5 ways to create deployment		
	can edit denloyment manifest in GCP console		— Services & Scaling— Updating deployments	
	can edit deployment mannest in GCP console –	Service is a load-balancing front end for Pods		
	creates new Deployment with newer version of the app (VI, V2)	blue/green deployment strategy = useful strategy when you want to deploy a new version of an app and also ensure that app services remain available while the Deployment is updated.	blue-green deployments	3 Kubernetes Workloads
		update strategy based on blue/green method		
		a/b testing is used to make business decisions based on the results derived from data	— Canary deployments	
	console shows you revision list with	shadow testing allows you to run a new, hidden version — can roll back using kubectl		
	summaries and creation dates	can pause rollouts with kubectl pause command, same with resume	— Managing deployments	
comms through pod-to-pod communication on	— your workload doesn't run in a single pod	and status — Pod = group of containers with shared storage and networking	rianaging deproyments	
the same node	your workload doesn't for in a single pour	nodes get Pod IP addresses from address ranges assigned to your	Pod networking	
	some volumes are ephemeral	VPC		
> manage durable storage in a cluster > independent of the pods lifecycle		— are a directory which is accessible to all of the containers in a pod		
>provisioned dynamically through PersistentVolumeClaims or explicitly created by a cluster admin	— some volumes are persistent			
emptyDir > ephemeral: shares pod's lifecycle				
ConfigMap > object can be referenced in a volume			— Volumes)
Secret > stores sensitive info, such as passwords	— ephemeral types explained			
donwardAPI > makes data about pods data available to				
containers PersistentVolume (PV)		— volume types	J	
independent of a pods lifecyclemanaged by K8smanually or dynamically provisioned				
> persistent disks are used by GKE as PVs *PVs must be claimed	— PersistentVolumes			
PersistentVolumeClaim (PVC) desired state described by us	— Object Spec	Objects: each thing K8s manages is represented by an object, you		
	— Object Status	can view and change these objects, attributes, and state >> persistent entities representing the sate of the cluster		
the K8s control plane will continuously monitor the state of the cluster, endlessly comparing reality to what has been declared and remedying	K8s expects you to tell it what you want the state of the objects under its mgmt to be, it will work to bring that state into being and	— principle of declarative mgmt	K8's concepts)
state as needed containers within a pod are tightly coupled with	keep it there a Pod embodies the environment where			
one another and can communicate using localhost IP address		Containers in a Pod share resources)	
only component where clients interact directly	interact using kubectleach node runs a kubelet > reports back to	kube APIserver)	
kubelet = K8s agent on each node	kube APIserver store the state of the cluster	etcd = cluster's db	K8s control plane	
		— kube scheduler		
use to manage workloads —	continuously monitors state of the cluster through kubeAPIserver GKE provisions control planes as part of	— kube controller manager)	
	abstract parts of the GKE service that are not exposed to GCP customers	— GKE manages entire K8s control plane		
use node pools to manage different kinds of nodes	GKE manages this by deploying and registering Compute Engine instances as	K8s doesn't create nodes, cluster admins create nodes and add them to K8s	K8s Engine Concepts	
a regional or zonal GKE cluster can also be set	regional cluster ensure that availability of app			
up as a private cluster	 is maintained across multiple zones in single region - 3 zones is default 	— zonal vs regional clusters		
apiVersion kind	— YAML file			
metadata spec	TATIL TILE			
cannot have 2 or the same chiest types with	— all objects are assigned unique ID	— objects are defined in a YAML file, USE version control on YAML files		
labels can be matched by label selectors	— labels = key value pairs, tag objects with			
ReplicaSets Deployments				2 Kubernetes Architecture
Replication Controllers StatefulSets	— controller object: manage state of the pods			
DaemonSets		— pods have a life cycle, they are ephemeral —	K8s object mgmt	
Jobs	deployment object: ensure that a defined set			
default	of pods is running at any given time			
kube-system -	provide scope for naming resources such as pods, deployments, and contollers			
kube-public implement resource quotas across cluster	K8s allows to abstract a single physical	— namespaces		
allow to use object names that would otherwise be duplicates of one another	— cluster into multiple clusters known as namespaces			
the default, exposes service on an IP address that is only accessible from within this cluster	— ClusterIP			
exposes service on the IP address of each node in the cluster, at a specific port number	— NodePort —	services provide load-balanced access to specific pods	— services	
exposes the service externally —	 LoadBalancer move and convert workloads into containers 			
	workloads can start as physical servers or VMs			
	Moves workload compute to container immediately (<10 min)	Migrate for Anthos moves VMs to containers		
	data can be migrated all at once or " streamed" to the cloud until the app is live in			
Migrate for Anthos is then installed on a GKE	the cloud			
processing cluster and is composed of many K8s resources >> container goes into Cloud Storage >> images stores in Container Registry	allow Migrate for Compute Engine to create a — pipeline for streaming or migrating the data from on prem or another CSP into GCP	— a migration requires an architecture to be built —	Migrate for Anthos	
storage >> images stores in Container Registry	configure processing cluster >> add	— Migration path		
	plan >> generate artifacts			
	migetl	migrate for Anthos installation		



Getting Started with

GKE

2023 Ivan Vlad S.