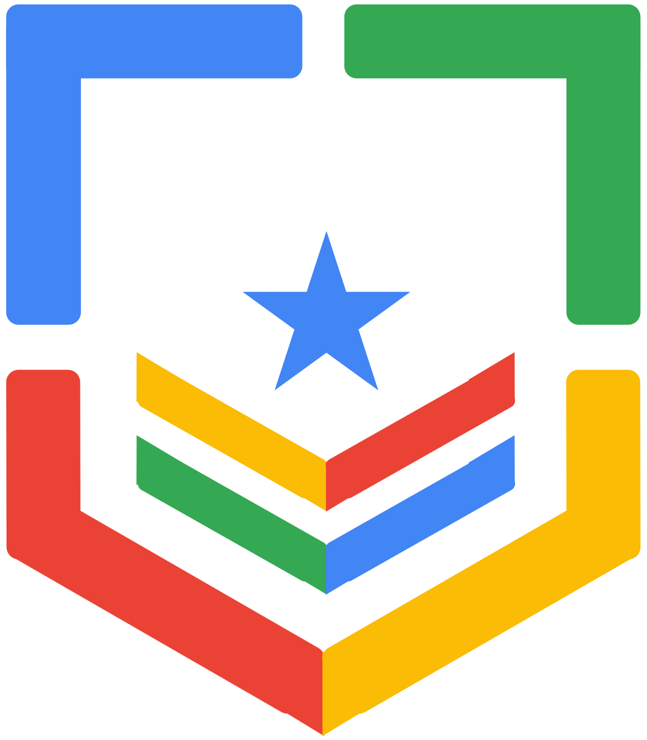
Google

Modernizing Monoliths

Participant Lectures

Challenge 1 - *Containerize the Web Application*



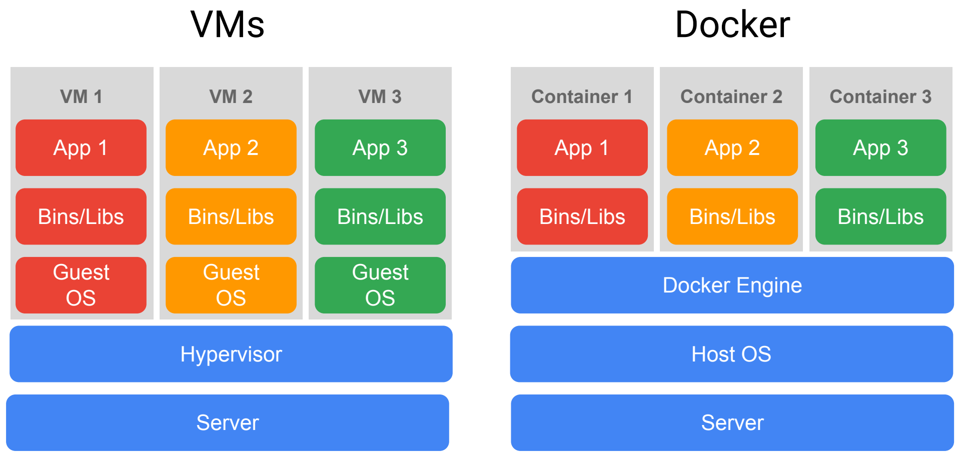
Study recap – Docker fundamentals

What Docker is

A container framework that allows you to standardize your application deployments across different machines as long as the host system has Docker installed

What Docker Isn't

A Virtual Machine



Study recap – Docker process, simplified

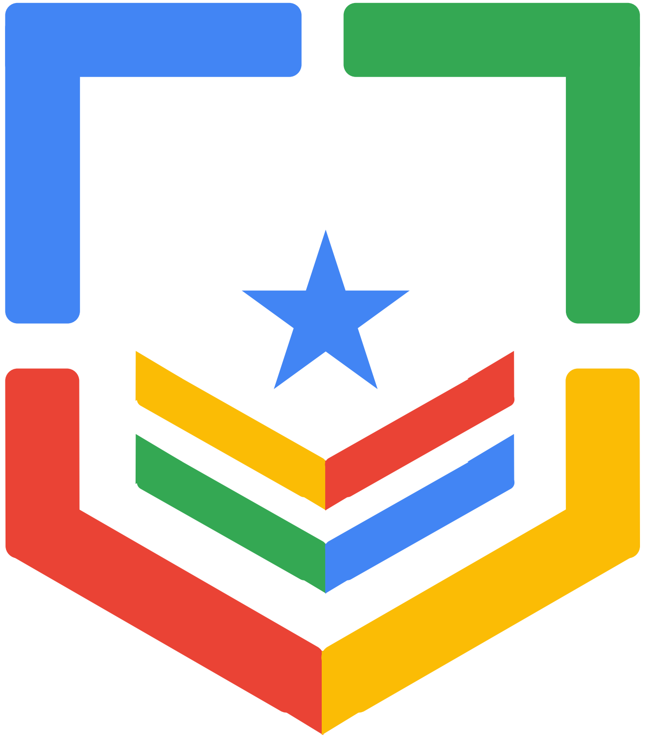
```
> 1. Download the application code
> 2. Create a Dockerfile and populate it with:
>     a. FROM commands to set the base image for your different stages
>     b. RUN commands to install needed packages
>     c. COPY commands to package the application files into the image
>     d. An ENTRYPOINT command to start the application
>     e. See https://docs.docker.com/engine/reference/builder/ for the full list of
>        commands
> 3. Build the image from the console using `docker build .` command
> 4. Test your newly created image from the console with the `docker run` command
> 5. Push your image to Dockerhub or Artifact Registry from the console using the
>     `docker push` command
```

Study recap – Multi-stage builds

- Remove layers that are needed in building the image but are not needed to run the final containerized application
- No need for multiple Dockerfiles or shell scripts to keep images lean

"Images are like parfaits; they have layers."
– Damian Lance

Challenge 2 - *Deploy with GKE*



Google

Challenge 1 recap

Docker is a container framework that allows you to standardize your application deployments across different machines as long as the host system has Docker installed. Docker containers are not VMs

What you accomplished:

- Created an image of the application for a game server
- Ran a container based on the image that you built
- SSH’d into your container and verified that all files are present
- Pushed your image to Artifact Registry

GKE Fundamentals

What GKE is

Google Kubernetes Engine (GKE) is a managed solution for automatically deploying, scaling, and managing containerized applications

What GKE Isn’t

- GCE Managed Instance Groups
- Easy to operate via the GUI

Important GKE Terms

Kubernetes - an open-source system for automating deployment, scaling, and management of containerized applications

Node - worker machines that run your containerized applications and other workloads

Cluster - a group of nodes that run containerized applications. Every cluster has at least one worker node

Node pool - a group of nodes within a cluster that all have the same configuration

Pod - the most basic deployable unit within a Kubernetes cluster, capable of running one or more containers

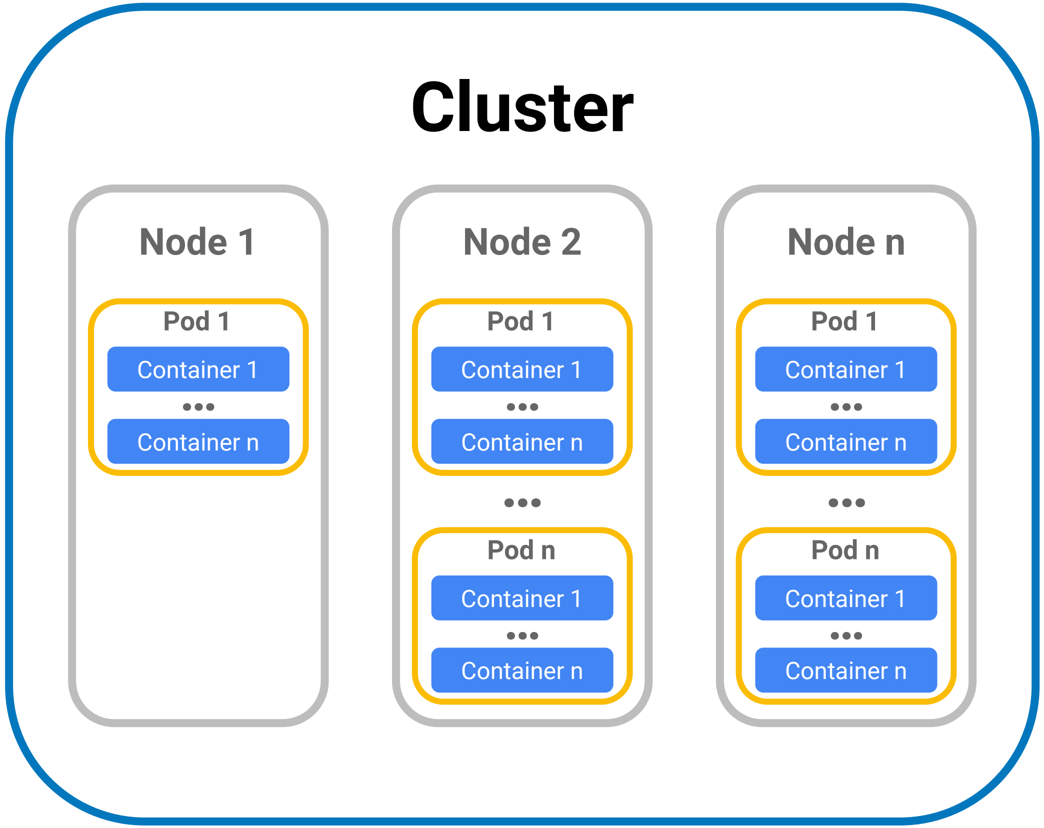
Namespace - an abstraction used by Kubernetes to organize objects in a cluster and provide a way to divide cluster resources

Deployment - an API object that manages a replicated application

Service - a method for exposing a network application that is running as one or more Pods in your cluster

K8s - an abbreviation for Kubernetes

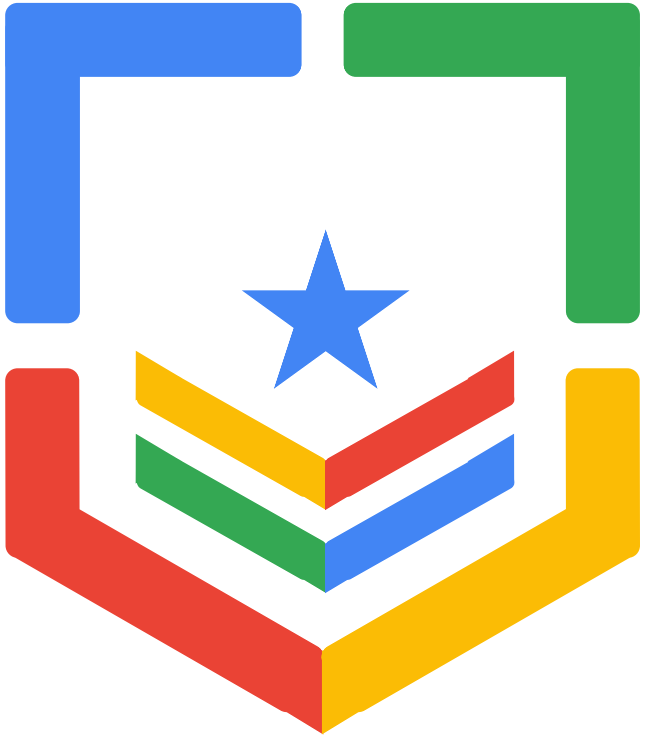
GKE Visual


Cluster
Node 1
Pod 1
Container 1
•••
Container n
Node 2
Pod 1
Container 1
•••
Container n
•••
Pod n
Container 1
•••
Container n
Node n
Pod 1
Container 1
•••
Container n
•••
Pod n
Container 1
•••
Container n

Deploying on GKE - Simplified Process

```
> 1. Create a cluster
> 2. Create a node pool for your workload
       a.   Delete the default node pool
> 3. Create a namespace
> 3. Create a deployment for your application
       a.   Add liveness and readiness probes
> 4. Create a service to load balance traffic to your pods
```

Challenge 3 - *Load Testing*



Google

Activity 2 – Recap

Google Kubernetes Engine is a managed solution for automatically deploying, scaling, and managing containerized applications

What you accomplished:

- Created a cluster and node pool
- Deployed your game servers to your node pool
  - Used liveness and readiness probes to ensure your pods are operable before sending traffic to them
- Used a service to load balance traffic to your pods

Appendix

Helpful Links - Day 1

- https://docs.docker.com/engine/reference/builder/
- https://pmac.io/2019/02/multi-stage-dockerfile-and-python-virtualenv/
- https://docs.docker.com/engine/install/
- https://github.com/git-guides/install-git
- https://github.com/crawl/crawl
- https://github.com/crawl/crawl/tree/master/crawl-ref/source/webserver#prerequisites
- https://github.com/crawl/crawl/blob/master/crawl-ref/INSTALL.md

Helpful Links - Day 2

- https://cloud.google.com/kubernetes-engine/docs/how-to/creating-a-zonal-cluster
- https://cloud.google.com/kubernetes-engine/docs/how-to/flexible-pod-cidr#setting_the_maximum_number_of_pods_in_a_new_node_pool_for_an_existing_cluster
- https://cloud.google.com/kubernetes-engine/docs/concepts/deployment
- https://cloud.google.com/kubernetes-engine/docs/concepts/deployment
- https://cloud.google.com/kubernetes-engine/docs/concepts/service
- https://cloud.google.com/kubernetes-engine/docs/concepts/service-load-balancer
- https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer
- https://cloud.google.com/kubernetes-engine/docs/how-to/exposing-apps - if stuck on creating a service
- https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/#define-a-liveness-http-request
- https://cloud.google.com/blog/products/containers-kubernetes/kubernetes-best-practices-setting-up-health-checks-with-readiness-and-liveness-probes