

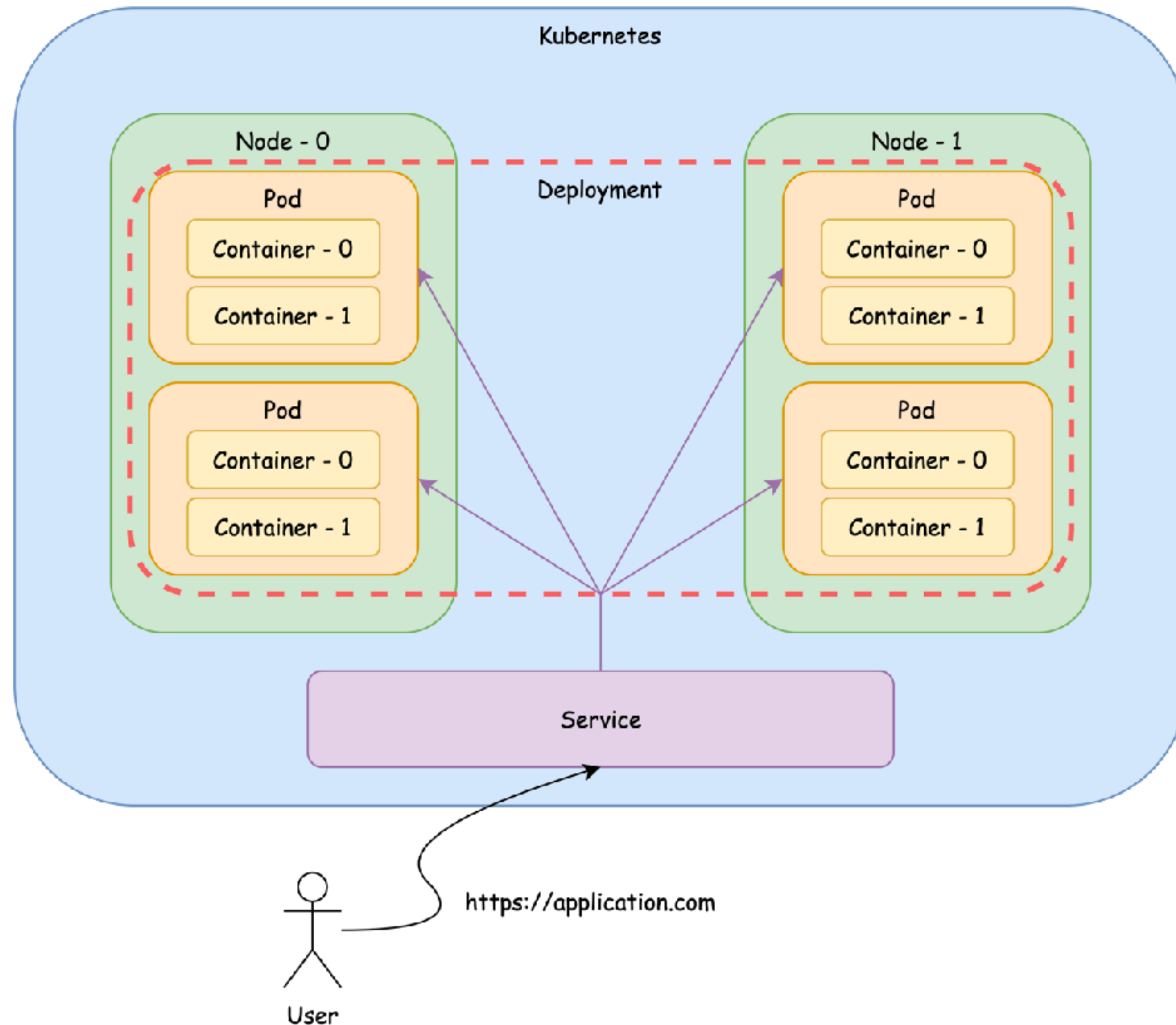
GKE Use Case 01

投資程式設計科 DevOps 組

Outline

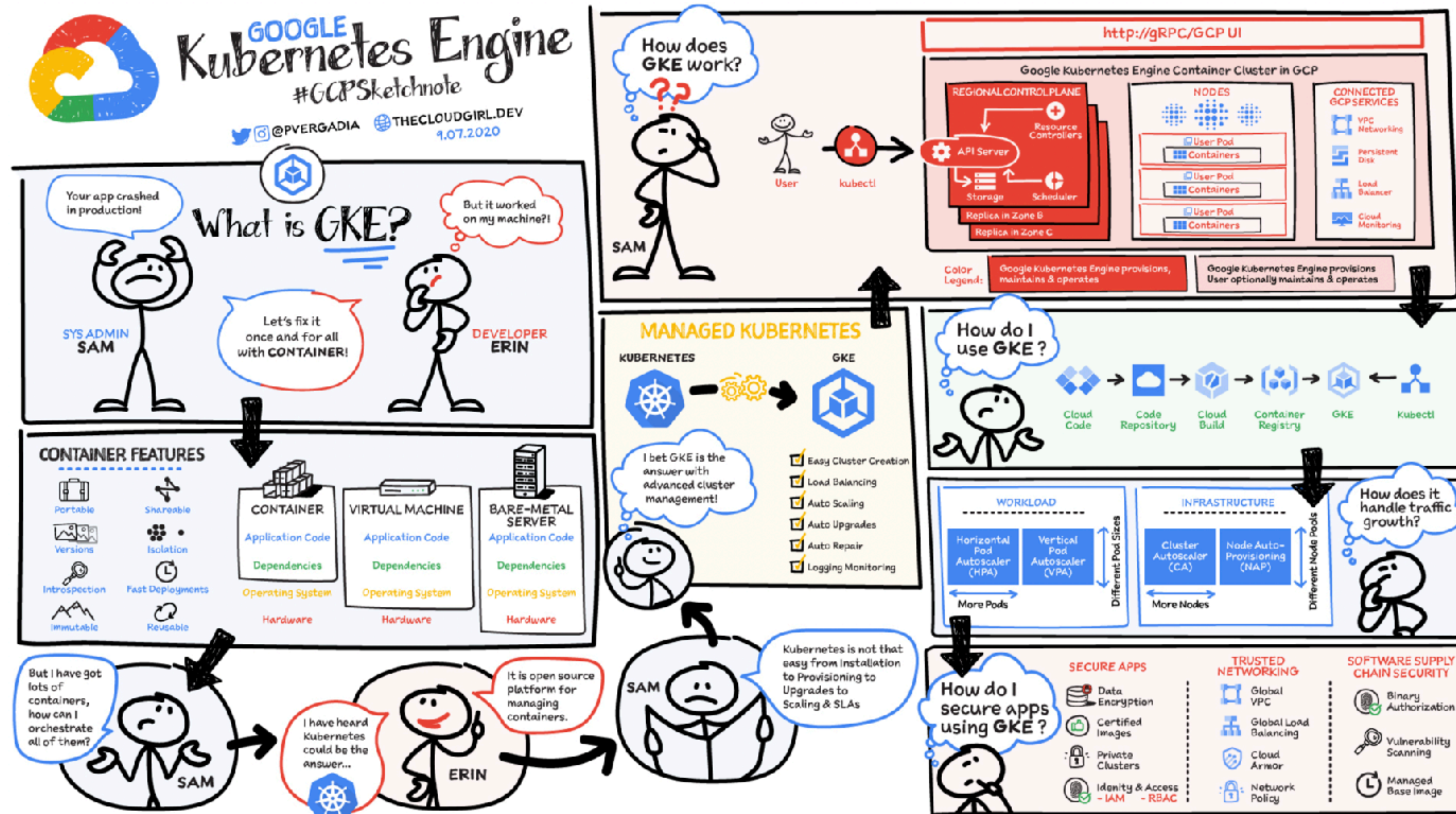
- Kubernetes Recap
- Google Kubernetes Engine (GKE)
- Lab: Set Up and Configure a Cloud Environment in Google Cloud
 - Task 1: Docker Image and Dockerfile
 - Task 2: Test Docker Image
 - Task 3: Push Image
 - Task 4: Create and Expose Deployment
 - Task 5: Update Deployment
 - Task 6: Jenkins Pipeline
- Recap

Kubernetes Recap




- Kubernetes (K8s)：容器管理平台
- Container：執行程式的獨立環境
 - Image：Container 的模版
- Pod：一個 Pod 可以容納多個 Container
- Node：實際運行 Container 的機器
- Deployment：定義 Pod 的內容
- Service：接收 Request 並轉發至 Pod 上






Google Kubernetes Engine (GKE)



Source: <https://thecloudgirl.dev/GKE.html>

Lab

 Deploy to Kubernetes in Google Cloud: Challenge Lab

Start Lab

01:30:00


Deploy to Kubernetes in Google Cloud: Challenge Lab

1 hour 30 minutes

7 Credits

★★★★☆

GSP318

 Google Cloud Self-Paced Labs

GSP318

Overview

Challenge scenario

Task 1: Create a Docker image and store the Dockerfile

Task 2: Test the created Docker image

Task 3: Push the Docker image in the Container Repository

Task 4: Create and expose a deployment in Kubernetes

Task 5: Update the deployment with a new version of valkyrie-app

Task 6: Create a pipeline in Jenkins to deploy your app

Congratulations!

—/100

<https://www.cloudskillsboost.google/focuses/10457?parent=catalog>

Lab - Task 1: Docker Image and Dockerfile

1. Clone valkyrie-app from Source Repository(GCP service)
 1. 右上 Clone Icon
 2. 下拉選單選擇 Google Cloud SDK
2. 在 valkyrie-app 目錄下建立 Dockerfile
3. 使用 Dockerfile 建立 Image valkyrie-dev:[tag]
 1. `docker build -t valkyrie-dev:[tag] .`
4. 執行 step1_v2.sh 確認進度

Lab - Task 2: Test Docker Image

1. 使用 valkyrie-dev:v0.0.2 啟動一個 Container，並將 Host 的 8080 port 與 Container 的 8080 port mapping
 1. `docker run --rm -it -p 8080:8080 valkyrie-dev:[tag]`
 1. `--rm` Container 完成後自動刪除
 2. `-it` 將 input 與 output 都接到當前的 terminal 上
 3. `-p 8080:8080` 將 host 的 8080(前面那組) 與 container 的 8080(後面那組) mapping
2. 開啟另一個 Terminal 視窗，執行 `step2_v2.sh` 確認進度
3. 也可以利用 Cloud Shell 的 Web Preview 功能檢視

Lab - Task 3: Push Image

1. 將 valkyrie-dev:[tag] 加上 tag gcr.io/qwiklabs-gcp-00-6808bfaf519e/valkyrie-dev:[tag]
 1. `docker tag valkyrie-dev:[tag] gcr.io/[GCP Project ID]/valkyrie-dev:[tag]`
 2. 將 Image push 至 Container Registry^{*}(GCP Service)
 1. `docker push gcr.io/[GCP Project ID]/valkyrie-dev:[tag]`
- ^{*} GCP 建議轉移使用 Artifact Registry，Container Registry 將不再新增功能，只進行安全性修復

Lab - Task 4: Create and Expose Deployment

1. 編輯與檢視 valkyrie-app/k8s 目錄下的 deployment.yaml 與 service.yaml
 1. 更新 deployment.yaml 中的 IMAGE_NAME
2. 取得 GKE 的權限
 1. 進入預先建立的 valkyrie-dev Cluster
 2. 點選 CONNECT，複製指令 Command-line access 至 Cloud Shell 中執行
3. 佈署 deployment
 1. `kubectl apply -f deployment.yaml`
 2. 執行後可以至 Workload 檢視佈署狀況
4. 佈署 service
 1. `kubectl apply -f service.yaml`
 2. 執行後可以至 Services & Ingress 檢視佈署狀況，佈署完成後點選 Endpoint IP 確認可以正常顯示服務

Lab - Task 5: Update Deployment

1. 將 Replica 從 1 增加至 5，有以下兩種方法
 1. 更新 deployment.yaml 的 replica
 2. 在 Cloud Console 的 Workloads Deployment 頁面中利用 ACTIONS => Scale => Edit replicas 直接修改 replica
2. Merge branch kurt-dev 的程式至 master
 1. git merge origin/kurt-dev
3. 建立一版新的 Image 版本為 [new version tag]，並 push 至 Container Registry
 1. docker build -t gcr.io/[GCP Project ID]/valkyrie-dev:[new version tag] .
 2. docker push gcr.io/[GCP Project ID]/valkyrie-dev:[new version tag]
4. 將 GKE 上的 deployment 更新為新版 Image
 1. 更新 deployment.yaml 中 image 為新版 Image Tag
 2. kubectl apply -f deployment.yaml

Lab - Task 6: Jenkins Pipeline

1. 登入在 GKE 上的 Jenkins

1. 取得密碼

1. `printf $(kubectl get secret cd-jenkins -o jsonpath="{.data.jenkins-admin-password}" | base64 --decode);echo`

2. 使用 port-forward 進入 Jenkins WebUI，帳號為 admin，指令參考畫面提示

2. 建立與 GCP 連線的認證

1. 管理 Jenkins => Manage Credentials => global => Add Credentials

2. Kind 選擇 Google Service Account from metadata

Lab - Task 6: Jenkins Pipeline

3. 建立一個 Pipeline 類型的 Job

1. Pipeline 選擇 Pipeline script from SCM

1. 類型: Git
2. Repository URL: `https://source.developers.google.com/p/[GCP Project ID]/r/valkyrie-app` (可以在 Source Repository 的 Clone 的 Manually generated credentials 找到 URL)
3. Credentials: 選擇剛剛建立名稱為 [GCP Project ID] 的 Service Account

4. 更改 Repo 內的內容並 commit 和 push

1. 更改 Jenkinsfile

1. YOUR_PROJECT 為實際的 GCP Project ID
2. 68 行的 `sh("sed ... deployment.yaml")` 更改為 `sh("sed -i.bak 's#IMAGE_HERE#${IMAGE_TAG}#' ./k8s/deployment.yaml")`

2. 更改 source/html.go 中的 green 為 orange

3. `git add Jenkinsfile Dockerfile source/html.go`

4. Commit and Push

5. 手動觸發 Jenkins 上的 Job

Recap

- GKE 是 GCP 提供的**全託管 K8s 服務**，無須自行建立與設定叢集等底層，可以專注於**功能使用與服務發布**
- Cloud Console 的 GKE 頁面可以**檢視與編輯**各種資源
 - Workloads : **Deployment**, Stateful Set, Job
 - Services & Ingress : **Service**, Ingress
- Image 可以放置於 Container Registry 或 Artifact Registry 中
- Code 可以放置於 Source Repository 中