Google Cloud Platform (GCP) DevOps Cheatsheet

A comprehensive cheat sheet for DevOps engineers working with Google Cloud Platform (GCP). This guide provides a quick reference to essential GCP services, commands, and best practices for implementing DevOps principles in the cloud.

Core Services & Concepts

Compute Engine

Description:	Virtual machines in the cloud. Provides customizable instances with various OS options.
Key Features:	Scalable, Customizable, Global infrastructure.
Use Cases:	Web hosting, application servers, batch processing.
gcloud command to create instance:	`gcloud compute instances create [INSTANCE_NAME] --zone=[ZONE]`

Kubernetes Engine (GKE)

Description:	Managed Kubernetes service for container orchestration.
Key Features:	Automated deployment, scaling, and management of containerized applications.
Use Cases:	Microservices architecture, containerized workloads.
gcloud command to create cluster:	`gcloud container clusters create [CLUSTER_NAME] --zone=[ZONE]`

Cloud Storage

Description:	Scalable and durable object storage.
Key Features:	Object versioning, lifecycle management, multiple storage classes (Standard, Nearline, Coldline, Archive).
Use Cases:	Storing backups, media files, and data archives.
gsutil command to create bucket:	`gsutil mb -l [LOCATION] gs://[BUCKET_NAME]`

Infrastructure as Code (IaC)

Cloud Deployment Manager

<table>
<tr><td><b>Description:</b></td><td>GCP’s native IaC service.</td></tr>
<tr><td><b>Key Features:</b></td><td>Uses YAML or Python to define infrastructure, supports templates and reusable configurations.</td></tr>
<tr><td><b>Use Cases:</b></td><td>Automating infrastructure provisioning and management.</td></tr>
<tr><td><b>Example Deployment Manager Configuration (YAML):</b></td><td><pre>
resources:
- name: my-instance
  type: compute.v1.instance
  properties:
    zone: us-central1-a
    machineType: zones/us-central1-a/machineTypes/f1-micro
    disks:
    - deviceName: boot
      type: PERSISTENT
      boot: true
      autoDelete: true
      initializeParams:
        sourceImage: projects/debian-cloud/global/images/family/debian-9
    networkInterfaces:
    - network: global/networks/default
</pre></td></tr>
</table>

Terraform on GCP

<table>
<tr><td><b>Description:</b></td><td>A popular open-source IaC tool that supports GCP.</td></tr>
<tr><td><b>Key Features:</b></td><td>Declarative configuration, state management, multi-cloud support.</td></tr>
<tr><td><b>Use Cases:</b></td><td>Managing infrastructure across multiple cloud providers and on-premises environments.</td></tr>
<tr><td><b>Example Terraform Configuration:</b></td><td><pre>
resource "google_compute_instance" "default" {
  name   = "terraform-instance"
  machine_type = "f1-micro"
  zone   = "us-central1-a"

  boot_disk {
    initialize_params {
      image = "debian-cloud/debian-9"
    }
  }

  network_interface {
    network = "default"
  }
}
</pre></td></tr>
</table>

CI/CD Pipelines

Cloud Build

Description:	GCP's managed CI/CD service.
Key Features:	Automated build, test, and deployment of applications, integrates with Cloud Source Repositories, GitHub, and Bitbucket.
Use Cases:	Continuous integration and continuous delivery pipelines.

Cloud Build Configuration (cloudbuild.yaml):

```yaml
steps:
- name: 'gcr.io/cloud-builders/docker'
  args: ['build', '-t', 'gcr.io/$PROJECT_ID/my-app:$SHORT_SHA', '.']
- name: 'gcr.io/cloud-builders/docker'
  args: ['push', 'gcr.io/$PROJECT_ID/my-app:$SHORT_SHA']
- name: 'gcr.io/cloud-builders/kubectl'
  args: ['set', 'image', 'deployment/my-app', 'my-app=gcr.io/$PROJECT_ID/my-app:$SHORT_SHA', '-n', 'default']
  env: ['CLOUDSDK_COMPUTE_ZONE=us-central1-a', 'CLOUDSDK_CONTAINER_CLUSTER=my-cluster']
```

Cloud Deploy

Description:	GCP's managed continuous delivery service that automates and orchestrates deployments to a variety of environments.
Key Features:	Progressive deployments (canary, blue/green), integrations with Cloud Build, approvals, rollback capabilities.
Use Cases:	Automated and safe deployments of applications to GKE, Cloud Run, and Compute Engine.

Monitoring and Logging

Cloud Monitoring

Description:	Provides visibility into the performance, uptime, and overall health of cloud-powered applications.
Key Features:	Dashboards, alerting, uptime checks, service monitoring.
Use Cases:	Monitoring application performance, infrastructure health, and user experience.

Example Metric Query (PromQL):

```
sum(rate(container_cpu_usage_seconds_total{namespace="production"}[5m])) by (pod)
```

Cloud Logging

Description:	Centralized log management for GCP services and applications.
Key Features:	Log aggregation, filtering, searching, and exporting.
Use Cases:	Troubleshooting application issues, auditing security events, and analyzing usage patterns.

Example Log Filter:

```
resource.type="gce_instance" AND severity>=ERROR
```