Serverless es un modelo economico, que nos permite pagar por lo que uso, y que podamos escalar como lo necesito

opciones:

* Serverless
  + Funciones -> Cloud Funtions -> unidades atómicas de código
* No Serverless
  + Iaas ( Infraestructura como servicio ) -> Maquinas virtuales (mas cercano a la administración y configuración del servidor)

En medio

* Kubernet es Engine (administras el escalamiento)
* Cloud Run (escalamiento automático)
* App Egine (PaaS: Plataforma como servicio) Solo escribo el código y el google quien se encarga del escalamiento, seguridad, actualizaciones

¿Por que Serverless?

* Fácil
* Ágil
* Barato: no pago si no lo uso
* Escalabilidad: si tengo mayor demanda el servidor me va a ayudar

El proveedor

me descarga de la responsabilidad de administrar la carga operativa (manejo de servidores, seguridad, actualización de software)

Function as a Service… una tarea puntual para ejecutar

* Esto para hacer pruebas de concepto es barbaro.

**LA EXPRESIOMN MAS POPULAR DE SERVERLESS SON LAS FUNCIONES**

**-HTTP**

**-CLOUD STRORAGE**

**CLOUD PUB/SUB**

**-CLOUD FIRESTORE**

**-FIREBASE :** realtime database, storage, analytics, auth

**-STACKDRIVER :** logging via PUB/SUB

SEGÚN EL RUNTIME, LAS FUNCIONES HTTP RECIBEN;

RUNTIME\_\_\_\_\_argumentos

PYTHON\_\_\_\_\_\_(request) de flask

NOde.js(8/10)\_\_\_\_ (request,response) de express.JS

Go\_\_\_\_\_\_\_ ( thtp, ResponseWriter, Http.Request)

IAM ( control de acceso y comportamiento)=

Es un mecanismo para determinar quien hace qué en los recursos de la plataforma.

* + Gestionar roles de cuentas de Google

EXISTEN DIFERENTES ROLES

-PRIMITIVOS

-PREDEFINIDOS

-PERSONALIZADOS

**Service accounts**

-Cuenta para autenticación

-Roles gestionados via IAM

-Generación de claves para consumos de servicios

**Escalabilidad Horizontal:**

* Replicar las instancias

**Escalabilidad Vertical:**

-Aumentar los recursos

SUPERVISION

Métricas, valores que obtenmos de lo que ocurre en la plataforma

SITE RELIABILITY ENGINEERING(**una manera de trabajar)**

.PILARES

-Orientada la disponibilidad y confiabilidad

- sutentada en cuantificaciones

.A) service level indicator

b)service level objective

c) server level agreement

CONECTORE VPC

Un servicio para conectar sus funciones a tu vppc

- Accede via IP privadas a los recursos

-La comunicacon es uniderccional

-Un ocnector puede ser compartido por varias funciones

Notas de Contenedores

**Process**  
Recursos manejados por el sistema operativo que tiene su propia memoria virtual.

**Namespaces**  
Alcances: Qué elementos son visibles, qué IPs puedo ver, qué árbol de directorios a los cuales tengo acceso.

**CGroups**  
Cuánto CPU, Cuánta Memoria voy a ocupar

**Union File System**  
Mecanismo que permite que se construya nuestra aplicación a través de capas en las que se definen las relaciones con las bibliotecas.

Containers: Codigo y dependencia empaquetados para que la aplicación se ejecute de forma rápida y confiable de un entorno a otro.  
Kubernetes: Es una plataforma open source que automatiza las operaciones de los contenedores.

KNATIVE

Es una herramienta para desplegar y trabajar cargas de trabjo serverless en kubernetes

Permite construir bloques ( fácil en cualquier proveedor de nube)

Mecanismo que atiende y distribuye las peticiones entre múltiples servicios , y sus revisiones.

Usa Istio

Permite que sea funcional aunque hay escalada a cero.

EVENTOS

Mecanismo que permite consumir y producir eventos según las especificaciones de Cloud Events

Sigue los principios básicos de PUB/SUB

Cloud run: knative en Google cloud

Cloud run = app engine + cloud function + kubernetes engine

Service administrada

. gestionado por Google cloud

. listo ´para usarse

. ellos despliegan y mantienen saludable y activa las instancias.

. manejan escalabilidad.

. actualizaciones provista por el proveedor.

CLOUD RUN FOR ANTHOS

. gestionado manualmente.

. despliegue y gestión del cluster de kubernetes.

. actualización son nuestra responsabilidad.

. la esclabilidad recae parcialmente en nosotros

. recomendable cuando ya tienees inversiones hechas en kubernetes

CLOUD RUN- SUPERVISION

. Google captura la salida estándar y la refleja en los logs

. Podemos usar la Logging API de Google Cloud para hacer nuestros registros personalizados.

**CI/CD**

**Continuous integration**

**Continuous delivery**

**Continuous deployement**

GOOGLE CLOUD BUILD

Construye artefactos

. capacidades de despliegue

-GKE

- APP ENGINE

- COMPUTE ENGINE

- CLOUD FUNCTIONS

. Posibilidades de testing

.Tareas de consistencias

* + EJECUTA PASOS REPETIBLES

INTEGRACIONES Y TRIGGERS

. cloud source repositories

. github

. bitbucket

. spinnaker via PUB/SUB

OPTIMIZACIONES

. cache

.local container images

. Google network

SERVERLESS

PERSONALIZACIONES E INTEGRACIONES

CLOUD SCHEDULER

Ejecutar con una periocidad

CLOUD STACK

Ejecuta tareas asíncronas

FIREBASE/CLOUD FUNCTION

Que no limite JS

CLOUD SQL

Persistencia de datos administrada