Cloud Team / Arquitectura

Directrices

Cloud Team / Arquitectura

Development

Development

Directrices generales de desarrollo en ambientes Cloud, en esta sección se provee el framework que será aceptado por arquitectura en instancias como CVT y propuestas de proveedores o desarrollos internos.

Tener en cuenta que en cada evaluación también se tomarán en cuenta aspectos como seguridad, CI/CD, infraestructura y el monitoreo/observabilidad de la solución, referirse a cada directriz según corresponda.

General

- Primera opción siempre, Microservicios !
- La nube principal siempre será GCP para deployar.
- Para poder trabajar en GCP, necesitará un proyecto: la directiva al respecto es que el equipo a cargo de nube creará el proyecto en nuestra organización, y dará los accesos respectivos. Por favor, no crear un proyecto de forma personal o en la misma consola del proveedor.
- Para lenguajes de programación con su Framework, según lo soportado por cada Cloud provider.
    - Lenguajes programación recomendados generales: Java (+ dropwizard framework, o

alguno de esta lista), Node.js (+express framework, o alguno de esta lista), Python (+django framework, +flask framework o alguno de esta lista).
        - **GCP**
        - **AWS**
        - **Azure**
    - Bases de datos: PostgreSQL / MySQL, seguir la documentacion de cada provider para ver versiones aceptadas.
        - **GCP**
        - **AWS**
        - **Azure**
    - Seguir lineamientos de **Seguridad**
    - Seguir lineamientos de **CI/CD**
    - Seguir lineamientos de **DoD**
    - Seguir lineamientos de **Infraestructura** en caso de requerir creación de componentes
    - Seguir lineamientos de **Monitoreo & Observabilidad**
    - NADA pasa a producción sin haber pasado al menos por QA

Cloud Team / Arquitectura

Cloud Team / Arquitectura

Infraestructura

Infraestructura

Mandamientos

- Terraform es la herramienta oficial para **IaC**
- Infraestructura como código, ALWAYS
- Maquinas virtuales deben tener su **agente** de monitoreo instalado
- Utilizar patrones de diseño en la creación de nueva infraestructura
- Seguir directriz de **etiquetado** de recursos

Stack

Tecnológico

Cloud Team / Arquitectura

Seguridad

Seguridad

Mandamientos

- NUNCA se brindan permisos de editor u owner
- Siempre se privilegia el uso de cuentas de servicio, es responsabilidad del proveedor o persona que hace uso de esta, rotar la KEY, dar de baja o dar aviso a Arquitectura/Cloud
- Siempre utilizar **Principle of Least Privilege**
- Los permisos a otorgar serán los siguientes
    - Owner/Editor: Julio Quinteros y/o Ignacio Campos (Excepciones deben ser validadas)
    - Permisos específicos (Compute engine admin, etc): Internos de Redsalud o Proveedores
    - Roles: Idealmente siempre utilizar esta opción al ser un permiso granular
- Utilizar métodos de autenticación controlados (SQL Proxy, IAP, etc)
- TODO en variables, nunca user:password en duro o data sensible.
- La exposición de puertos (en caso de usar instancias/VM/máquinas virtuales) debe ser la mínima posible, es decir, proveer los puertos requeridos para el funcionamiento del servicio, nada más ni nada menos
- Establecer restricciones de orígenes: no dejar habilitada la conectividad/enrutamiento para todos los orígenes (máscaras 0.0.0.0/0, por ejemplo)
- Toda comunicación que ocurre a través de redes públicas

está encriptada

- Todos los servidores de aplicativos web que tienen salida a internet deben estar tras un balanceador
- (Si se requiere) Acceso a grupo de instancias se logra a través de un bastion (expuesto a internet)
- Todo servicio que obtenga/escriba información de RedSalud debe ser a través de Apigee
    - Si el servicio interactúa contra Apigee, posee las credenciales respectivas, conoce quotas de uso, y los endpoints a los cuales conectarse
- Todo servicio debe contar con alguna medida de autenticación
    - Cuando sea posible eliminar la dependencia de una identidad (cuenta de usuario), lo recomendado es usar autenticación por OAuth2.0 o algún token perecible
    - En caso contrario, user/password con protocolo de rotación de clave y auditoría/monitoreo de actividad

Endpoints (WIP)

- IAP, Ingress, LB

Cloud Team / Arquitectura

CI/CD

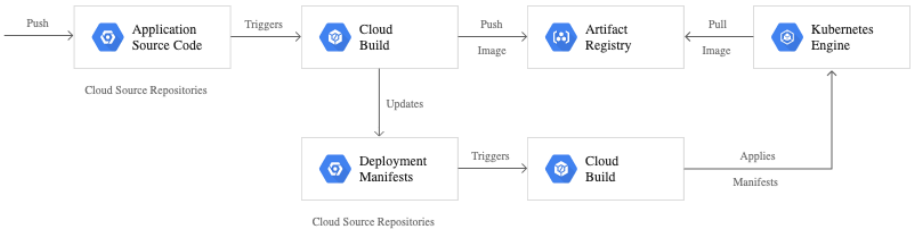
Continuous deployment &

Continuous integration

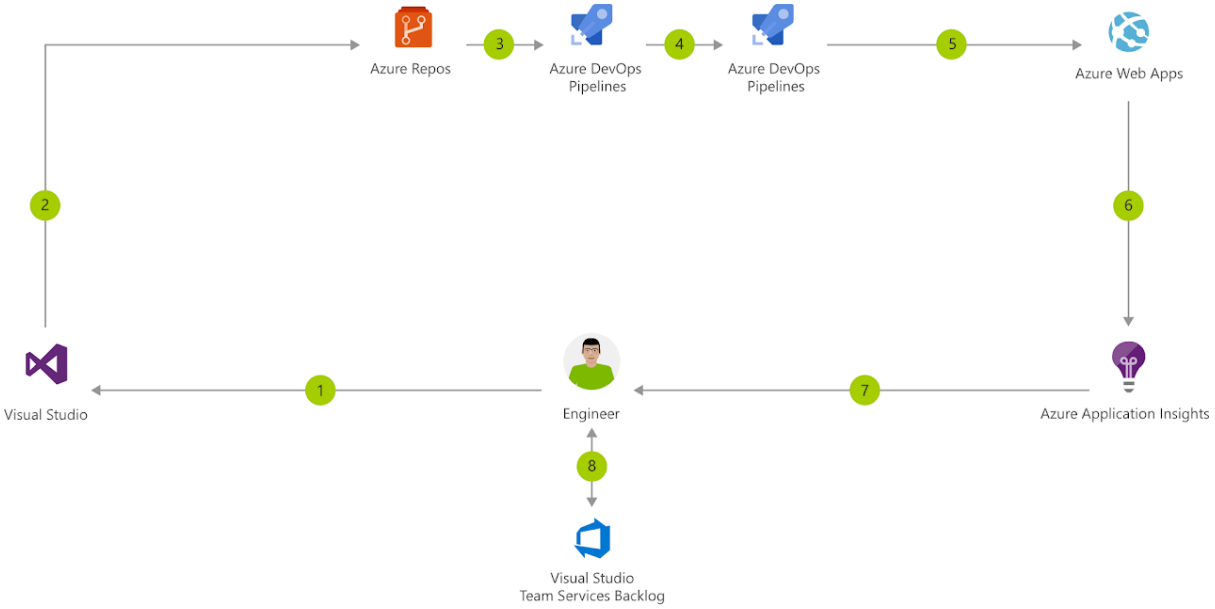
Mandamientos

- Utilizar herramientas nativas de cada Cloud Provider como se muestran en los diagramas correpondientes
- TODO artefacto, deploy o solución debe tener un pipeline declarativo y con comentarios en cada step

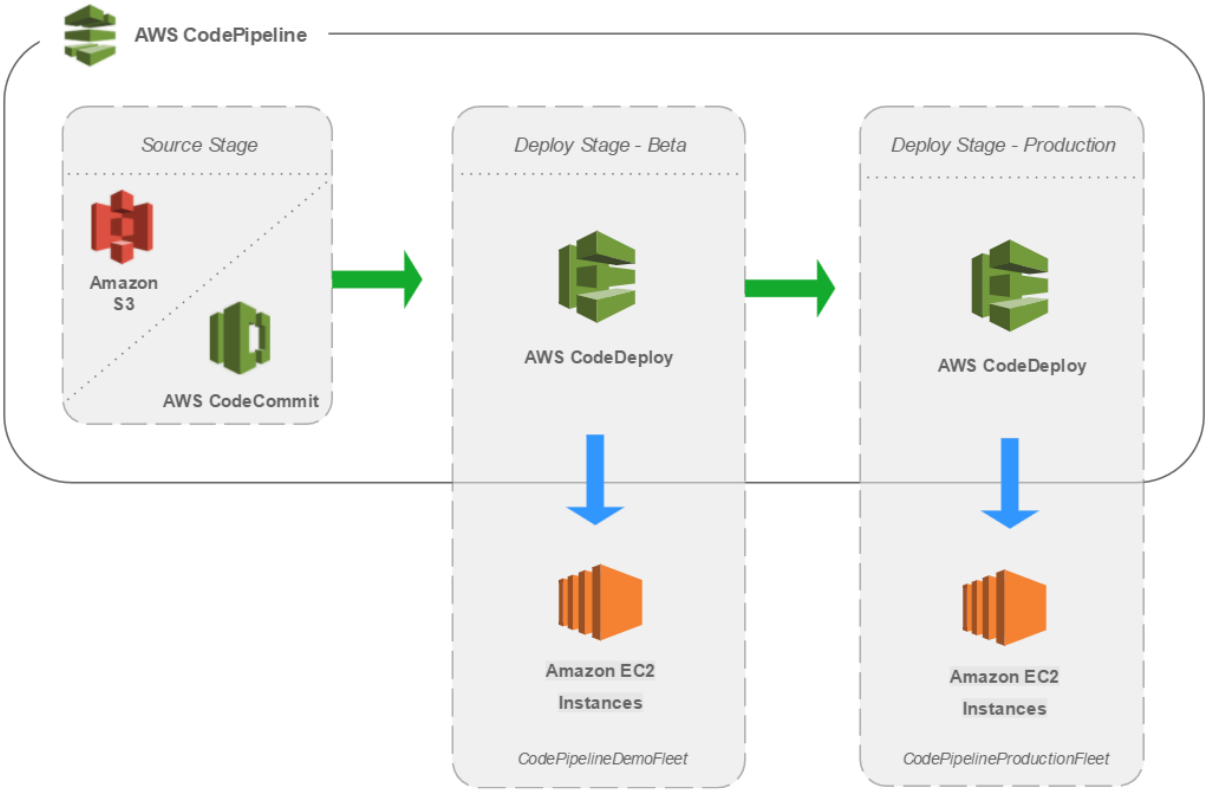
GCP



Azure



AWS



Cloud Team / Arquitectura

Monitoreo y Observabilidad

Cloud Team / Arquitectura

Diagramas

Diagramas & Docs

Directriz de como realizar diagramas y documentacion para soluciones implementadas en ambientes Cloud

Mandamientos

- Herramientas oficiales de diagramas:
  - **Cloud diagramming tool**
  - **Draw.io**

Cloud Team / Arquitectura

Definition of Done

Definition of done

Definition of Done es un conjunto de reglas que determinan cuándo un elemento está terminado. Terminado significa listo para poner en producción a disposición del usuario.

Mandamientos

- Monday actualizado
    - Cards finalizadas
    - Comentarios diarios del status en la card
    - Vinculos a documentacion de referencia o creada para la tarea
- Documentación asociada en Gitlab o Repo de Cloud Provider según corresponda
    - Readme en el repo
    - Flujos si es que existen con integraciones
- Diagramas de arquitectura asociados, ver **directriz** para diagramas
- **Planilla** de product owners completada

Cloud Team / Arquitectura

Networking