Jonnathan De La Barra

1. ¿Cuál es la diferencia entre nube pública, privada e hibrida?

La nube pública, privada e híbrida son tres modelos de computación en la nube que ofrecen diferentes niveles de control, seguridad y escalabilidad.

Nube Pública:

- **Infraestructura compartida:** Los recursos informáticos (servidores, almacenamiento, redes) se comparten entre múltiples usuarios a través de internet.
- **Proveedores**: Grandes empresas como Amazon Web Services (AWS), Microsoft Azure y Google Cloud Platform ofrecen estos servicios.
- **Pago por uso**: Los usuarios pagan solo por los recursos que utilizan, lo que la hace muy flexible y escalable.
- **Menor control:** Los usuarios tienen menos control sobre la infraestructura subyacente.
- **Ideal para:** Pequeñas y medianas empresas, aplicaciones con demanda variable y proyectos que no requieren un alto nivel de seguridad.

Nube Privada:

- **Infraestructura exclusiva:** Los recursos informáticos están dedicados a una sola organización y suelen estar ubicados en el centro de datos de la empresa o en un centro de datos gestionado por un proveedor de servicios.
- **Mayor control**: La empresa tiene un control total sobre la infraestructura y los datos.
- **Mayor seguridad:** Ideal para manejar datos sensibles y cumplir con regulaciones de cumplimiento.
- **Costos más altos:** Requiere una inversión inicial más alta y costos operativos continuos.
- **Ideal para:** Grandes empresas con altas demandas de seguridad, aplicaciones críticas y datos confidenciales.

Nube Híbrida:

- **Combinación de ambos**: Es una combinación de nube pública y privada, donde la empresa utiliza ambos entornos para aprovechar las ventajas de cada uno.
- **Flexibilidad**: Permite a las empresas mover cargas de trabajo entre la nube pública y privada según sea necesario.
- **Escalabilidad**: Permite escalar rápidamente los recursos en la nube pública durante picos de demanda.
- **Seguridad**: Los datos sensibles pueden almacenarse en la nube privada, mientras que las aplicaciones menos críticas pueden ejecutarse en la nube pública.
- **Ideal para**: Empresas que necesitan una solución flexible y escalable, pero también requieren un alto nivel de seguridad para ciertos datos.

2. Describa tres prácticas de seguridad en la nube

Cifrado de Datos:

- **En tránsito:** Protege la información mientras se mueve entre diferentes puntos de la nube, utilizando protocolos como HTTPS para cifrar las comunicaciones.
- **En reposo:** Cifra los datos almacenados en la nube para garantizar que, incluso si alguien accediera a ellos de forma no autorizada, no puedan leerlos.

- **Claves de cifrado**: Gestiona de forma segura las claves de cifrado para evitar que se vean comprometidas.

Gestión de Identidades y Accesos (IAM):

- **Autenticación fuerte**: Implementa mecanismos de autenticación de múltiples factores (MFA) para verificar la identidad de los usuarios.
- **Autorización granular**: Otorga a cada usuario los permisos mínimos necesarios para realizar sus tareas, siguiendo el principio de "privilegio mínimo".
- **Auditoría de acceso:** Monitorea y registra todas las acciones realizadas por los usuarios en la nube para detectar actividades sospechosas.

Continuidad del Negocio y Recuperación ante Desastres:

- **Respaldos regulares**: Realiza copias de seguridad frecuentes de los datos y las aplicaciones para poder restaurarlas en caso de pérdida o corrupción.
- **Pruebas de recuperación**: Ejecuta pruebas periódicas para verificar que los procedimientos de recuperación ante desastres funcionan correctamente.
- **Replicación de datos**: Replica los datos en diferentes regiones o centros de datos para aumentar la disponibilidad y reducir el riesgo de pérdida de datos.

3. ¿Qué es la IaC, y cuáles son sus principales beneficios?, mencione 2 herramientas de IaC y sus principales características.

La IaC es una metodología que permite gestionar y provisionar la infraestructura de TI de forma automatizada, utilizando código en lugar de procesos manuales. En esencia, se trata de definir la infraestructura (servidores, redes, almacenamiento, etc.) mediante archivos de configuración, lo que facilita su creación, modificación y escalado.

Beneficios de la IaC:

- **Automatización**: Elimina la necesidad de realizar tareas manuales repetitivas, reduciendo el riesgo de errores humanos y agilizando los procesos.
- **Consistencia**: Garantiza que los entornos sean consistentes y reproducibles, facilitando la implementación de cambios y la detección de problemas.
- **Escalabilidad**: Permite escalar la infraestructura de forma rápida y eficiente para adaptarse a las demandas cambiantes del negocio.
- **Versionamiento**: Al utilizar código, se pueden versionar los cambios en la infraestructura, lo que facilita la colaboración, la auditoría y la reversión de cambios.
- **Mayor agilidad**: Permite desplegar nuevos entornos y aplicaciones de manera más rápida, acelerando el time-to-market.

Menciono aquí 2 Herramientas de IaC y sus características

1. **Terraform**:

- **Multi-cloud**: Permite gestionar infraestructuras en múltiples proveedores de nube (AWS, Azure, GCP, etc.).
- **Declarativo**: Describe el estado deseado de la infraestructura, y Terraform se encarga de crear y actualizar los recursos necesarios.
- **Planificación**: Genera un plan de ejecución antes de realizar cambios, lo que permite revisar las modificaciones antes de aplicarlas.

- **Módulos**: Facilita la reutilización de código mediante módulos, lo que agiliza el desarrollo.
- **Estado**: Mantiene un estado de la infraestructura, lo que permite realizar cambios incrementales y rastrear la historia de la configuración.

2. **Ansible**:

- **Agenteless**: No requiere la instalación de agentes en los servidores, lo que simplifica la configuración.
- **Declarativo y imperativo**: Combina la sintaxis declarativa para definir el estado deseado con la sintaxis imperativa para tareas más complejas.
- **Playbooks**: Utiliza playbooks para definir las tareas a realizar en los servidores.
- **Módulos**: Ofrece una amplia variedad de módulos para realizar tareas comunes, como la instalación de software, la configuración de servicios, etc.
- **Idempotencia**: Garantiza que la ejecución de un playbook múltiples veces produzca el mismo resultado, evitando configuraciones inconsistentes.

4. ¿Qué métricas considera esenciales para el monitoreo de soluciones en la nube?

Métricas de Rendimiento:

- **Latencia**: Tiempo de respuesta de las aplicaciones y servicios.
- **Throughput**: Volumen de datos procesados por unidad de tiempo.
- **CPU**: Uso de la unidad central de procesamiento.
- **Memoria**: Uso de la memoria RAM.
- **Disco**: Uso del espacio de almacenamiento y velocidad de E/S.
- **Red**: Tráfico de red entrante y saliente, latencia de red.

Métricas de Disponibilidad:

- **Tiempo de actividad:** Porcentaje de tiempo que un servicio está disponible.
- **Tiempo de inactividad**: Duración de las interrupciones del servicio.
- **Tasa de errores**: Número de errores por unidad de tiempo.
- **Número de solicitudes**: Cantidad de solicitudes recibidas por el servicio.

Métricas de Uso y Costos:

- **Uso de recursos:** Consumo de recursos de computación, almacenamiento y red.
- **Costo por servicio**: Costo asociado a cada servicio en la nube.
- **Costo total de propiedad (TCO):** Costo total de operar la infraestructura en la nube.

Métricas de Seguridad:

- **Intentos de acceso no autorizados**: Número de intentos fallidos de acceso a sistemas o datos.
- **Vulnerabilidades identificadas**: Número de vulnerabilidades detectadas en la infraestructura.
- **Incidentes de seguridad**: Número de incidentes de seguridad reportados.

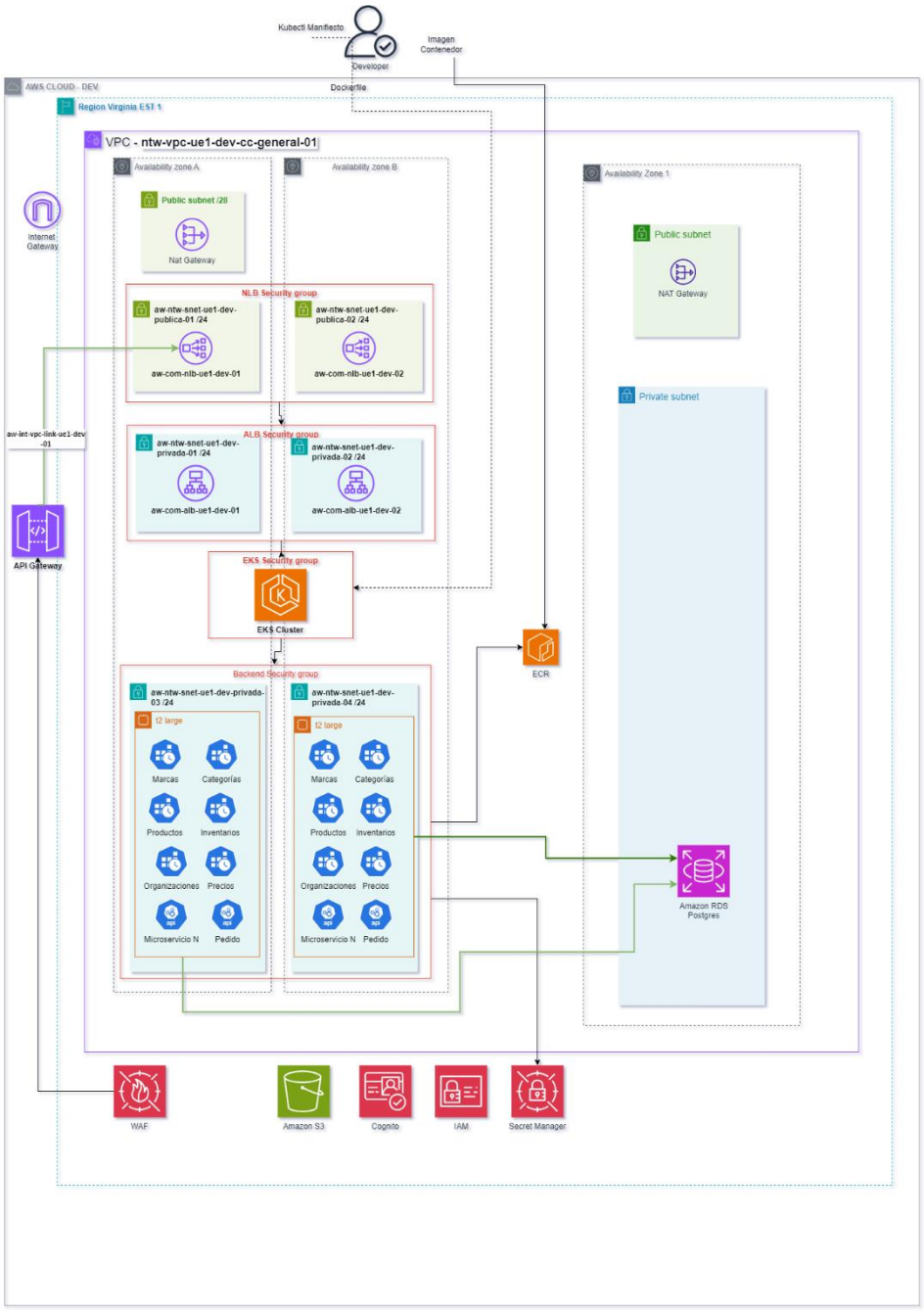
5. ¿Qué es Docker y cuáles son sus componentes principales?

Docker es una plataforma de software que permite crear, desplegar y gestionar aplicaciones dentro de contenedores. Estos contenedores son como pequeños paquetes autónomos que incluyen todo lo necesario para ejecutar una aplicación: código, bibliotecas, herramientas y configuraciones. Gracias a Docker, es posible empaquetar una aplicación y sus dependencias de manera consistente, lo que facilita su distribución y ejecución en diferentes entornos, ya sean locales o en la nube.

Componentes principales de Docker:

- **Docker Daemon:** Es el servicio que se ejecuta en segundo plano y gestiona los objetos de Docker, como imágenes, contenedores, redes y volúmenes. Recibe solicitudes de la interfaz de línea de comandos (CLI) o de la API REST.
- **Docker Imágenes:** Son plantillas inmutables que contienen todo lo necesario para ejecutar una aplicación. Se crean a partir de un Dockerfile, un archivo de texto que contiene las instrucciones para construir la imagen.
- **Docker Contenedores:** Son instancias en ejecución de una imagen. Cada contenedor es un proceso aislado que comparte el kernel del sistema operativo host con otros contenedores.
- **Docker Hub:** Es un registro público en la nube donde se almacenan las imágenes de Docker. Permite a los desarrolladores compartir y descargar imágenes de Docker de forma gratuita.
- **Docker CLI**: Es la interfaz de línea de comandos que se utiliza para interactuar con el Docker daemon. Permite crear, iniciar, detener, eliminar y gestionar contenedores e imágenes.

6. Caso práctico



6.- ARQUITECTURA

6.1 .- PATRONES DE ARQUITECTURA

El sistema Incluirá los siguientes patrones:

6.1.1 .- FrontEnd

A). Web

Se utilizara el patrón **Contenedor-Componente** con la estructura de carpetas definidas por el framework Next.js.

- ***Organización clara:*** La estructura de carpetas definida por Next.js separa los componentes en contenedores y componentes de presentación, lo que facilita la búsqueda y modificación de componentes específicos.
- ***Reutilización y escalabilidad:*** El patrón contenedor-componente permite la reutilización de componentes de presentación en diferentes contenedores, evitando la duplicación de código y mejorando la escalabilidad del proyecto.
- ***Facilidad de pruebas:*** La separación entre la lógica en los contenedores y la presentación en los componentes de presentación simplifica las pruebas unitarias, ya que puedes probar la lógica de manera aislada.

B). Mobile

Se utilizara el Patrón **Provider** con estructura de carpetas **orientadas a funcionalidades**, el cual permitirá:

- Puedes comunicarse con varios componentes que se encuentran en diferentes niveles.
- Aislamiento de componentes para mayor flexibilidad en la escalabilidad y eficiencia en el desarrollo.
- Código mas legible por ende nos ayuda al mantenimiento y soporte del desarrollo.

6.1.2 .- BackEnd

A). El patrón que se utilizara a nivel de arquitectura esta basado en microservicios, el cual permitirá:

- Mayor velocidad y agilidad en el despliegue del sistema, lo que mejora el time to market.
- Alta escalabilidad ante una alta concurrencia de usuarios
- Reusabilidad, pues se basa en la creación de pequeños componentes que realice una única tarea.
- Interoperabilidad, ya que cualquier aplicación o componente pueda comunicarse con ellos, sin importar en que tecnología está desarrollado.

B). El patrón que se utilizará en el Desarrollo está basado en marco Hexagonal, el cual permitirá:

- Separación de responsabilidades, con ello nos ayuda a mantener la lógica de negocio independiente de la Tecnología a usar.
- Mayor facilidad en la automatización de pruebas.
- Permite tener mayor escalabilidad en diferentes niveles (Lógico o Infraestructura).