Prueba técnica para DevOps

1. Creación de un repositorio con CI/CD

- Crea un repositorio en GitHub con un código básico (puede ser un servicio web minimalista en cualquier lenguaje, por ejemplo, Node.js, Python, Go o cualquier otro de tu preferencia).
- Configura un pipeline de CI/CD usando **GitHub Actions**. El pipeline debe incluir:
    1. **Build**: Compilar el código.
    2. **Test**: Ejecutar pruebas unitarias y enviar los resultados a Sonarqube.
    3. **Deploy**: Desplegar la aplicación en un clúster **EKS** en AWS.

Requisitos:

Estas Credenciales tiene duración de una semana

- **Aws Access Key AKIAT4Q3YABN7S2PQ4UW**
- **Aws Secret Access Key** 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
- Cluster Name rappi-web-eks-cluster
- Sonar URL sonarqube.imaginamos.com
- Sonar Token sqp_f61ea613abc37cafa41a10c0a1b82855295984cb

Si poseen un error en la comunicación o conexión deben describir el error y proponer cuál es la configuración faltante para llevar a cabo el proceso

---

2. Generación de infraestructura con Terraform

Crea un archivo de Terraform que cumpla con los siguientes requisitos:

1. **Infraestructura base:**
    - Una red VPC configurada con subnets públicas y privadas.
    - Reglas de seguridad (Security Groups) necesarias.
    - Un Internet Gateway y NAT Gateway.
2. **Recursos principales:**
    - Un clúster **EKS** con nodos configurados.
    - Una base de datos **RDS** (PostgreSQL o MySQL).
    - Opcional: DocumentDB o buckets S3. La elección debe depender de la variable project_type ("database" o "storage").
3. **Configuraciones adicionales:**

- Configurar IAM Roles y Policies necesarias para los servicios.
- Proveer opciones escalables para los nodos del clúster.

Entrega:

- Proporciona los archivos .tf necesarios para levantar esta infraestructura en AWS.
- Asegúrate de incluir variables reutilizables para facilitar la personalización.

---

3. Monitoreo centralizado de microservicios

Diseña un proceso para monitorear los logs de múltiples microservicios distribuidos en varios clústeres **EKS**. Describe:

1. Las herramientas seleccionadas y su configuración:
   - **Logs**: Propón una solución centralizada para recolectar y visualizar logs, por ejemplo, **Loki** con **Grafana** o **Elasticsearch** con **Kibana**.
   - **Costos y eficiencia**: Argumenta por qué estas herramientas son eficientes en términos de costo y escalabilidad.
2. Proceso:
   - Cómo configuraste la recolección de logs desde los microservicios.
   - Cómo los desarrolladores pueden consultar fallos específicos en tiempo real.
3. Opcional: Proponer métricas clave para medir la salud de los microservicios (uso de recursos, tiempos de respuesta, errores).

Entrega:

- Documentación del proceso con diagramas si es necesario.
- Configuración y ejemplos de código para implementar esta solución.

Criterios de evaluación

1. **Funcionalidad**: El pipeline debe funcionar y desplegar correctamente en EKS.
2. **Calidad del código**: Limpieza, modularidad y uso adecuado de herramientas.
3. **Infraestructura**: Diseño óptimo y reusable.
4. **Documentación**: Claridad y detalle en los pasos seguidos.
5. **Innovación**: Soluciones innovadoras para optimizar costos y mejorar el monitoreo.