Prueba Técnica Desafio Devops

Bienvenido a este desafío para ingenieros de DevOps. La intención de esta tarea de prueba es brindarle una tarea diaria del mundo real que el equipo de ingeniería de Finaktiva podría enfrentar en un día de trabajo o en un proyecto. Dicho esto, es bastante probable que el proyecto que estás a punto de afrontar valga más que un día de trabajo. Sin embargo, la intención no es que te apresures y completes todos los objetivos a continuación, sino que te concentres en realizar al menos un pequeño trabajo, completamente funcional y bien documentado, en lugar de múltiples piezas sueltas que no funcionen. Recuerde que la idea principal es presentar una idea coherente sobre lo que estás tratando de hacer para dar solución al requerimiento planteado.

Cómo abordarlo

Como se indicó anteriormente, la idea es que asumas pequeñas porciones de trabajo teniendo en cuenta que esto es algo que es probable (y será para fines de revisión) que se entregará a otro ingeniero para que lo ponga a funcionar. Teniendo esto en cuenta, el enfoque sugerido debería ser el siguiente:

- Documenta lo que está a punto de hacer, idealmente en un archivo README. Hacerlo te ayudará a establecer tus ideas y te dará una idea de qué tiene sentido (y qué no) abordar primero.
- Escribe algún tipo de prueba para tu trabajo. Haz que falle tal como lo harías mientras haces TDD (Test driven development). Luego escribe la pieza que hará que pases el “examen”. No es necesario que sea una prueba automatizada (incluso una lista de cómo hacerlo es suficiente), sino una forma de garantizar una forma repetible de hacer que tu trabajo funcione.
- Has que tu trabajo funcione en partes aisladas y componibles. Hacerlo te ayudará a llegar al final del día con algo funcional, que no necesita ser la pieza completa.

Qué esperamos de ti

Hay más cosas que puedes hacer a continuación de las que un día te puede permitir hacer. Somos **MUY** conscientes de ello. No esperamos que entregue todo en un día. En cambio, queremos que apuestes por tu área más fuerte y centres tu trabajo allí. Concéntrate en lo que sabes que puedes hacer, hazlo, teniendo SIEMPRE en mente que es un trabajo de entrega a otra persona que realmente intentará ponerlo a funcionar. Eso significa que no se trata sólo de si funciona o no, sino también de qué tan viable es que otra persona lo haga funcionar siguiendo tus instrucciones.

Después de un día de trabajo, puedes enviarnos el resultado de tu trabajo a través de un repositorio de GitHub, compartido al usuario gustavo.jimenez@finaktiva.com para que lo revise. Quizás te preguntes si puedes tomar más de un día. Para responder a eso, tenga en cuenta que, si tardas más de un día en entregar tu tarea, esperaremos que entregues un trabajo más completo y funcional, por lo que el límite de tiempo lo defines al final eres tú.

Proyecto de prueba

Este es el proyecto de prueba en el que trabajarás. Está dividido en varias secciones para que puedas elegir dónde tiene más sentido comenzar.

IaaC

Para esta sección, deberás crear un clúster de Fargate alojado en AWS mediante el servicio ECS y ECR. Para lograr esto, deberá proporcionar la siguiente lista de requisitos:

- El aprovisionamiento **DEBE** gestionarse mediante la técnica IaaC. Nuestra herramienta preferida para realizar el trabajo es AWS CDK, pero puedes sugerir cualquier otra herramienta que consiga el resultado final. Si opta por utilizar otra herramienta, nos gustaría que nos proporcione los motivos (beneficios, desventajas) de la herramienta de su elección.
- El clúster **DEBE** vivir y utilizar VPC y subredes personalizadas.
- Las subredes del clúster y la VPC **DEBEN** de igual construirse con IaaC. Los valores de estas (CDIR, cantidad de sus redes públicas, privadas y aislada) deben proporcionarse como valores parametrizables (archivo de variables) o calcularse mediante cualquier otro método.
- El clúster **DEBE** tener al menos dos aplicaciones corriendo (es de tu libre elección elegir estas aplicaciones), debes definir la mejor arquitectura que consideres para desplegar estas aplicaciones (un servicio con dos tasks, dos servicios cada una con un task, u otra de tu preferencia) y los motivos (beneficios, desventajas) de tu elección.
- La comunicación con las aplicaciones **DEBE** realizarse mediante una URL pública (Load Balancer, API Gateway u otro). El acceso a través de subredes privada NO DEBE estar habilitado.
- El grupo de nodos o tasks **NO DEBE** tener IP públicas habilitadas de forma predeterminada.
- Se **DEBE** poder acceder desde internet a las aplicaciones del clúster solo a través de HTTPS (puerto 443).
- Desde IaaC se **DEBE** establecer al menos una capa de seguridad para garantizar en caso de requerirse, poder limitar el acceso a un listado de IPs públicas registradas en un archivo de parametrización.
- Se **DEBE** garantizar alta disponibilidad de las aplicaciones y disponer de escenarios de fallas sobre la misma y/o de autoescalado ante escenario como crecimiento de la CPU.

Nota: Recuerde que todos estos puntos deben ser configurados con IaaC.

CI/CD

Para esta sección, necesitaremos que crees los pipelines para la implementación de la aplicación descrita anteriormente. Puedes elegir la herramienta de tu preferencia, Jenkins File, MakeFile, Code Pipeline, GitHub Actions u otra de tu preferencia, eres libre de elegir la herramienta que te resulte más cómoda. No obstante, debes proporcionar algunos aportes como beneficios o desventajas de la herramienta que elijas.

Para el pipeline en sí, queremos que:

- Crees un pipeline que permita la implementación de sus aplicaciones en tres ambientes distintos (dev, stg, prod) sobre tres regiones diferentes (virginia, Ohio y oregon), es decir la aplicación sobre un ambiente específico debe ser desplegada sobre una región determinada (ejemplo dev sobre Ohio). Esto debe ser parametrizable (ambientes y regiones) desde el archivo de propiedades.
- Describa y configure 2 estrategias de despliegues y actualización de aplicaciones sobre contenedores en AWS (fargate), detalle las ventajas de cada una de estas estrategias, y en que tipo de aplicaciones es beneficioso o no utilizar una u otra. Recuerde documentar y evidencias las pruebas realizadas y deje como variables parametrizables el poder utilizar una estrategia u otra en su código IaaC.

Prueba teórica o de conocimientos técnicos

El conjunto de escenarios presentados a continuación pretender entender un poco su capacidad de resolución de problemas o incidentes, no hay respuesta mala o incorrecta. No obstante, debe explicar detalladamente y justificar las soluciones ofrecidas a cada uno de los casos expuestos de manera escrita.

1. Actualmente dispone de un servidor web alojado sobre su LAN (e.g 192.168.0.10), en el cual tiene instalado un apache-tomcat sobre la cual ha alojado la página web de la empresa por el puerto 8080, y Web Services desplegados sobre el puerto 8443. Se solicita hacer uso del dominio mipruebaFinaktiva.com.co para publicar a internet tanto la página web, como los web services. No obstante, se requiere que ambos sean accesibles por la misma URL y puerto (HTTPS -443). Describa al menos 2 soluciones para resolver este requerimiento, es libre de disponer el uso de cualquier dispositivo y/o aplicación conocida para dar solución a lo anterior.

2. Un cliente indica que está intentando conectarse los servicios https://mipruebaFinaktiva.com.co/servicio pero que cuando carga la página web en su navegador no carga. sin embargo, al realizar usted la prueba internamente si logra ver disponible los servicios. Describa y enuncie 5 razones por la cual puede estar pasando el problema y como ayudar al cliente a resolverla con su ayuda.

3. Un balanceador de carga dispone de diferentes algoritmos de balanceo como: round robin, round robin weight, least connection, source, URI para tráfico TCP y HTTP. Describa cada uno de estos y en qué casos ustedes utilizarían cada uno de estos, su comportamiento para tráfico TCP y HTTP, además de las ventajas y desventajas de estos.