Cloud & DevOps Fundamentals – Automation Tools
Practical Task

Tu actual cliente quiere migrar a la nube, tu equipo no posee mucha experiencia con la nube así que acuden a ti para lograr la migración. El cliente tiene varios requerimientos para que los tengas en cuenta en el momento de la ejecución de tu plan:

- Quieren que su infra cumpla con los mejores estándares y prácticas, para ellos encuentras esta información que crees te será de utilidad para realizar la migración (Azure: https://learn.microsoft.com/en-us/azure/well-architected/ y AWS: https://docs.aws.amazon.com/wellarchitected/latest/framework/the-pillars-of-the-framework.html)
- Actualmente el cliente está pasando por una crisis económica y busca lograr la migración, pero reduciendo los costos, están abiertos a acceder a usar recursos que requieran de algún budget siempre y cuando haya una razón de peso detrás de esa decisión.
- El cliente sabe que tienes un buen conocimiento de Terraform, así que confían en ti para desarrollar la nueva infraestructura usando esta herramienta, así que recae en ti cualquier decisión de planeación sobre el uso de la herramienta. El cliente quiere saber cuáles fueron las decisiones que tomaste y el porqué de ellas.
    - El cliente quiere desplegar dos ambientes: QA (ambiente de pruebas) y Producción (ambiente que los usuarios finales usarán) – crees que el uso de Workspaces te servirán para cumplir este requerimiento y encuentras esta información: https://developer.hashicorp.com/terraform/cli/workspaces
    - Como tu equipo te apoyará en esta tarea es clave que puedas poder trabajar remotamente tú y tu equipo sin tener problemas durante la creación de la nueva infraestructura así que compartes esta información con tu equipo: Azure - https://learn.microsoft.com/en-us/azure/developer/terraform/store-state-in-azure-storage?tabs=azure-cli y AWS - https://aws.amazon.com/blogs/devops/best-practices-for-managing-terraform-state-files-in-aws-ci-cd-pipeline/ (tip: puedes crear de forma manual los recursos tales como buckets y Storage Accounts para almacenar los states, no es necesario incluirlos en Terraform).
    - El cliente quiere poder reutilizar su código de Terraform en otras de la empresa y que este pueda ser mantenido por tú equipo por lo que se sugiere el uso de módulos públicos o privados para crear la infraestructura.
- El cliente ha escuchado hablar de Ansible, así que asume que es algo importante que quieren tener implementado en su migración. Decides que la mejor forma de implementar esto es para la configuración de tu bastion host/jump box y de las instancias (si llegases a implementar alguna de estas) y sus dependencias.

- Por último, el cliente quiere poder monitorear los recursos creados y sus comportamientos y rendimientos, no hay presupuesto para herramientas de terceros así que optan por una solución in-house, es decir, una solución dentro del mismo proveedor de nube.
    - Para este requerimiento esperan también que puedas mostrar un diagrama de arquitectura que sirva para documentación al cliente.

Con toda esta información el cliente espera que puedas llevar a cabo la migración sin ningún problema, recuerda que toda decisión tomada sobre los servicios usados y procesos y demás deben tener una razón detrás - al final todo será evaluado.

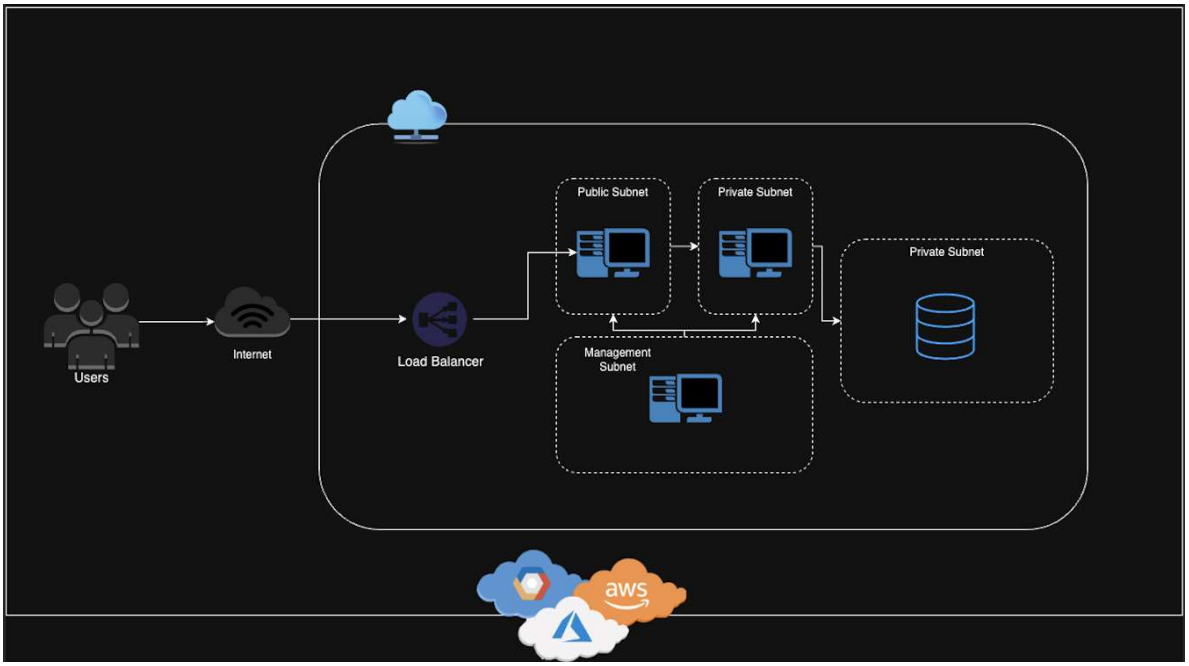
Descripción de la tarea:

El código de la aplicación a migrar se encuentra en el siguiente repositorio: https://github.com/aljoveza/devops-rampup/tree/master la aplicación está compuesta por los siguientes componentes:

- Front-end
- Back-end
- Base de datos MySQL

Se debe desplegar cada uno de los componentes haciendo uso de la nube pública asignada, incluyendo todos los componentes de networking tales como load-balancer, nat-gateway, subnets, route tables y demás necesarios para colocar la aplicación en funcionamiento.

Este es el diagrama de una arquitectura básica a grandes rasgos (habrá cosas que tendrás que cambiar de acuerdo con los requisitos del cliente y de la tarea):



Aspectos obligatorios:

- No utilizar servicios serverless, todos los componentes se deben desplegar sobre instancias.
- La base de datos se debe desplegar usando el servicio adecuado (AWS RDS y Azure Database for MySQL)
- La base de datos y la instancia qué contiene el back-end deben estar en subnets privadas y deben contar con acceso a internet saliente para la instalación de dependencias.
- El tráfico hacía el backend debe ser a través de un load-balancer.
- Se deben utilizar solo servicios incluidos en el free-tier.
- Se debe configurar un bastion host qué se encargue de correr todos los procesos de Ansible para configurar las instancias.