Construcción de Compiladores - Laboratorio Guiado No. 3 — Uso de ANTLR para Parsear Terraform y Administrar Droplets de DigitalOcean

1 Introducción

En este laboratorio, deberán usar ANTLR para parsear un archivo de Terraform y mapear el resultado a llamadas de API REST con Python para administrar Droplets de DigitalOcean. Además, utilizarán Docker para configurar y ejecutar su entorno para hacerlo más fácil.

Pero antes de eso, deberán conocer primero las diferencias entre usar Terraform para crear un Droplet en DigitalOcean y usar las API Rest de DigitalOcean para crear el droplet. Para ello realice lo siguiente:

- Lea el archivo titulado **"Construcción de Compiladores - Laboratorio Guiado No. 4 - Terraform"** y realice las actividades dentro de esta guía.
- Lea el archivo titulado **"Construcción de Compiladores - Laboratorio Guiado No. 4 - Bash"** y realice las actividades dentro de esta guía.
- Para ambas guías, deberá de tomar capturas de pantalla de las operaciones y resultados y adjuntarlos al reporte de este laboratorio.
- Luego, realice las actividades dentro de esta guía. Se proporciona código base que usted puede utilizar, además puede utilizar herramientas de GenAI o cualquier material de apoyo para realizar las actividades. La idea principal es que usted experimente con sus propias manos el proceso de crear un parser "pseudo-compilador" para un lenguaje de programación de DSL que pueda analizar y luego transformar a otro conjunto de operaciones resultantes, con el fin que usted vea un caso de uso real que podemos dar a los conceptos de compiladores en la vida real.
- Consulte todos los archivos proporcionados en Canvas.
- No realice operaciones indebidas con el API TOKEN de Digital Ocean que usaremos para estas actividades, de lo contrario se penalizará sobre la entrega por no seguir instrucciones.
- **Deberá realizar este laboratorio en grupos de 5 personas**.

2 Instrucciones y Requerimientos

- Este Lab se encuentra en el repo oficial. Ahi encontrará un playground para ejecutar la parte de ANTLR también.
- Compile el archivo de Terraform en la carpeta de antlr usando el driver de Python que se proporciona y vea cómo se crea el droplet (el código imprime la IP del droplet y el ID del mismo).
- Luego de hacerlo, haga un ping a la IP y vea que si está up el recurso.
- Luego, haga un delete manual de este droplet usando el script de bash que usó ya anteriormente cuando creó instancias con bash y la REST API. DEBE HACER DELETE MANUAL.
- Luego, deberá de completar el ecosistema de su propio Terraform, este archivo como está hace mimic de un ¨terraform apply –auto-approve¨. Entonces lo que ahora necesito que haga, es que cree el código que soporte hacer un ¨terraform destroy –auto-approve".
- Para lograr esto, deberá soportar parámetros en el archivo de Python.
- Dependiendo del flag proporcionado al cli de Python entonces ya hará un apply o destroy.
- Después, necesita crear el código para hacer un destroy, que básicamente es llamar a la REST API también.
- Pero para poder hacer un destroy, necesita saber el ID del droplet, entonces, deberá agregar código para crear un terraform statefile (.tfstate).
- Este statefile se debe crear cuando haga un apply. Se guardará un archivo y este tendrá un json con la información correspondiente, es decir, para el droplet con este nombre proporcionado, se asocia este ID y esta IP.
- Ya con este archivo, cuando haga un destroy entonces ya puede mandar el ID para destruir el recurso.

3 Puntos Extra

- Investigue cómo se configuran las llaves de SSH para un Droplet en DigitalOcean.
- Añada el soporte en su implementación con ANTLR para permitir leer una llave de SSH y attachearla al droplet cuando se cree.
- Luego, ingrese al droplet cuando ya se haya creado usando la llave de SSH y muestre su evidencia al respecto.
- Ponderación: 1 punto neto extra.

4 Entregables

- Repositorio de Github con todo su código, PDF y/o README con documentación y evidencias.
- Un video no listado de YouTube con la ejecución de su lab.