Universidad Técnica Federico Santa María

Tarea 2: Flujo Pokémon

Alonso Díaz Candia

1 de junio de 2025

Para levantar los contenedores use *docker compose up –build*. Puede agregar el flag *-d* si no quiere tener la terminal ligada al proceso. Usted debe, sí o sí, ejecutar esta línea para comenzar esta tarea.

Antes de comenzar esta tarea, déjeme decirle algo: en esta actividad usted no solo deberá *codear*, sino que además deberá preocuparse de entender el flujo que intento explicar (se aceptan sugerencias post-preguntas) en la imagen que adjunto a continuación.

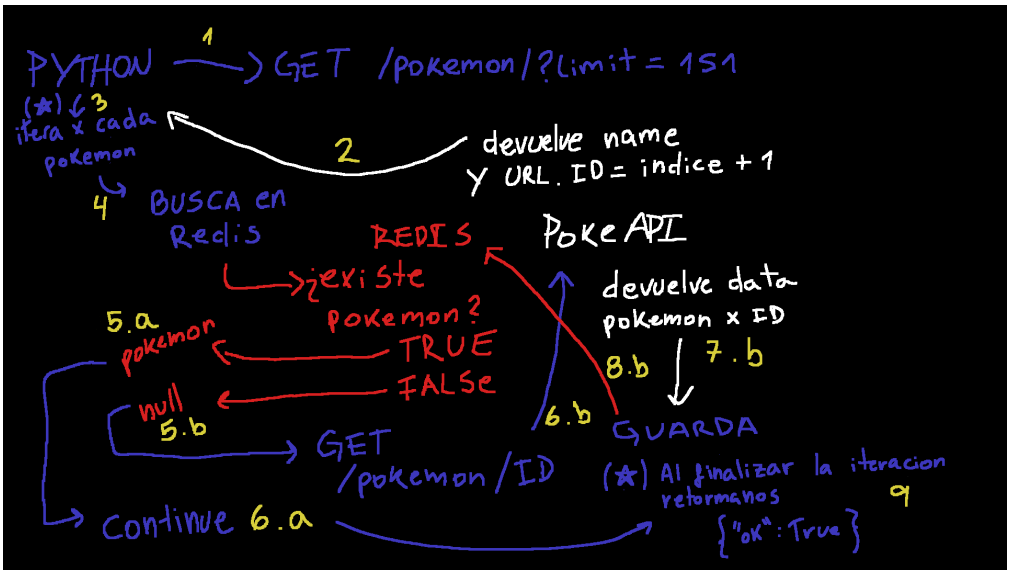

Figure 1: Intento de explicación del flujo nº1.

En la imagen anterior, el color azul representa lo que hace Python, el color rojo lo que hace Redis, y el color blanco representa la PokeAPI. Dentro de los archivos que se le adjuntan junto a este PDF, usted puede encontrar un archivo .py con el flujo mostrado en la imagen anterior y con comentarios en los puntos que se ven en la imagen. Voy a volver a recalcar: USTED, ANTES DE ESCRIBIR CUALQUIER LÍNEA DE CÓDIGO, DEBE ENTENDER MUY BIEN EL FLUJO ENTREGADO. PUEDE USAR LA CANTIDAD DE PRINTS

QUE ESTIME CONVENIENTE PARA ENTENDERLO, PERO NO PUEDE AGREGAR NI MODIFICAR EL ENDPOINT ENTREGADO, O INCURRIRÁ EN UN DESCUENTO DE 70 PUNTOS.

Una vez que haya entendido el flujo anterior, se le pide crear los siguientes endpoints utilizando FastAPI:

1. Un endpoint del tipo GET, cuya URL sea `/pokemon/ID`, que entregue como JSON la información de cada Pokémon dado su ID.
2. Un endpoint del tipo GET, cuya URL sea `/pokemon/filter/type/TYPE`, que entregue una lista en JSON con todos los Pokémon cuyo tipo primario corresponda al tipo entregado en la variable `TYPE`.
3. Un endpoint del tipo GET, cuya URL sea `/pokemon/filter/type/TYPE/2`, similar al anterior, pero que entregue todos los Pokémon cuyo segundo tipo sea el tipo entregado en la variable `TYPE`.
4. Un endpoint del tipo GET, cuya URL sea `/pokemon/filter/power/`, que entregue los Pokémon más fuertes según su stat de ataque. Por defecto, debe entregar un top 5, pero puede recibir un parámetro opcional llamado `results` (de tipo entero), que haga referencia a la cantidad de resultados que debe entregar el endpoint.
5. Realice lo mismo con la defensa, la velocidad y el peso. Los endpoints deben llamarse `defense`, `speed` y `weight`, respectivamente.
6. Cree un endpoint muy similar a `/pokemon/extract_pokemon` del tipo GET, cuya URL sea `/pokemon/extract_power_move`, que sea capaz de guardar en Redis (de la forma que usted estime conveniente) el poder del ataque consultado.
7. Cree un endpoint que devuelva una batalla Pokémon por turnos, cuya URL sea `/pokemon/fight/ID1v` donde `ID1` e `ID2` hacen referencia a los IDs de dos Pokémon. En cada turno, cada Pokémon lanza un ataque; el que comienza se elige de forma aleatoria. Todos los ataques tienen un porcentaje de efectividad del 100%. La batalla debe iterar hasta que un Pokémon quede fuera de combate. El historial de la batalla debe ser guardado en algún tipo de estructura que usted estime conveniente, con tal de que el endpoint devuelva un JSON.