Prueba técnica para desarrolladores backend

Introducción

Habi desea tener una herramienta en la que sus usuarios puedan consultar los inmuebles disponibles para la venta. En esta herramienta los usuarios deben ser capaces de ver tanto los inmuebles vendidos como los disponibles. Con el objetivo de hacer más fácil la búsqueda, se espera que los usuarios puedan aplicar diferentes filtros a la búsqueda.

Adicionalmente, se espera que los usuarios puedan darle “me gusta” a los inmuebles con el fin de tener un ranking interno de los inmuebles más llamativos.

Requerimientos

Habi desea tener dos microservicios. El primero para que los usuarios externos puedan consultar los inmuebles disponibles almacenados en la base de datos. El segundo para que los usuarios puedan darle “Me gusta” a un inmueble en específico.

No funcionales

- Se espera que entregues código fácil de mantener, fácil de leer, y autodocumentado.
- Se espera que el código siga una guía de estilos definida (ej. PEP8 para Python).
- Los microservicios deben construirse para ser consumidos en una arquitectura REST.

Funcionales (historias de usuario)

Servicio de consulta

- Los usuarios pueden consultar los inmuebles con los estados: “pre_venta”, “en_venta” y “vendido” (los inmuebles con estados distintos nunca deben ser visibles por el usuario).
- Los usuarios pueden filtrar estos inmuebles por: Año de construcción, Ciudad, Estado.
- Los usuarios pueden aplicar varios filtros en la misma consulta.
- Los usuarios pueden ver la siguiente información del inmueble: Dirección, Ciudad, Estado, Precio de venta y Descripción.

Servicio de “Me gusta”

- Los usuarios pueden darle me gusta a un inmueble en específico y esto debe quedar registrado en la base de datos.
- Los “Me gusta” son de usuarios registrados, y debe quedar registrado en la base de datos el histórico de “me gusta” de cada usuario y a cuáles inmuebles.

Instrucciones para la prueba técnica

1. El código debe quedar almacenado en un repositorio de git.
2. Como tu primer commit, incluye un README detallando la tecnologías que vas a utilizar y cómo vas a abordar el desarrollo.
3. Si tienes dudas escríbelas en el REAME y resuelvelas tú mismo, junto con la razón de porque las resolviste de esa manera.
4. En el correo que recibiste la prueba deben estar las credenciales de acceso para conectarse a la base de datos.
5. El primer requerimiento (Servicio de consulta) es práctico, por lo tanto se espera el código funcional.
6. En el primer requerimiento, crear un archivo JSON con los datos que esperas que lleguen del front con los filtros solicitados por el usuario.
7. En el primer requerimiento, el estado actual de un inmueble es el último estado insertado en la tabla “status_history” para dicho inmueble.
8. Se espera que no modifiques ningún registro en la base de datos, pero si necesitas una mayor cantidad de registros, puedes agregar nuevos.
9. La información de otros registros puede que tenga inconsistencias, recuerda manejar esas excepciones.
10. El segundo requerimiento (Servicio de “Me gusta”) es conceptual. No existe el modelo en la base de datos para soportar esta información.
11. En el segundo requerimiento se espera que tu extiendas este modelo con un diagrama de Entidad-Relación para soportar esta información. Por lo tanto no se espera que escribas código del microservicio, ni modifiques la base de datos. Únicamente el diagrama y el código SQL para extender el modelo, junto con la explicación de porque lo modificaste de esa forma (incluyelo en el README).
12. Tu código debe tener pruebas unitarias.
13. Recuerda divertirte haciendo este reto, si tienes bloqueos, continúa con otra parte.
14. Al terminar la prueba, responde al mismo correo desde el cual se te envió la prueba con el enlace al repositorio.

Puntos extra

Esta parte es completamente opcional. Se considerarán estos retos como puntos extra y harán destacar tu prueba.

1. Hacer las pruebas unitarias de tu código con TDD (Test-Driven Development).
2. Proponer un mejor modelo de la estructura actual de base de datos, con el objetivo de mejorar la velocidad de las consultas, se espera un diagrama y la explicación de porque lo modelaste de esa forma.