Pruebas técnicas Desarrolladores de Software Senior y Semi Senior

¡Hola!

Gracias por querer pertenecer a Double V Partners. Antes de conocernos en una llamada queremos determinar tu seniority. Si bien la prueba no lo es todo, no ayudará a saber tus habilidades y, una vez hagas parte de la compañía, saber en qué enfocar tu plan de carrera para que nunca pares de crecer.

A continuación hay 2 pruebas. Deberás escoger 1 de ellas y seguir las instrucciones. Hay una prueba de front y una prueba de back. Escoge la que más te guste.

El repositorio lo tendrás que enviar a welcome@doublevpartners.com con el asunto “Prueba Técnica”.

Nos vemos pronto.

¿Qué se busca evaluar?

Principalmente los siguientes aspectos:

- Creatividad para resolver los requerimientos,
- Calidad del código entregado (estructura y buenas prácticas),
- Eficiencia de los algoritmos entregados,
- Familiaridad con Frameworks y plataformas de desarrollo Web.
- Se darán puntos de seniority por el uso de principios SOLID y patrones de diseño.
- Tendrá especial calificación pruebas con cobertura de Tests(Unit Testing). Recomendables Jest o Testing Library:
    - https://jestjs.io/
    - https://testing-library.com/
- Compartir Repositorio a través de GitHub: https://github.com/

Prueba Front End # 1

1. Asegúrate de tener Node.js y npm instalados.

2. Se solicita crear la aplicación utilizando la tecnología Web de su elección, se privilegiarán a los candidatos que utilicen Angular(8 o superior) + Bootstrap.
3. Recomendamos emplear un máximo de 3 (tres) horas y enviar todo lo que puedas.

Objetivo: Crear una aplicación que ayude a obtener una lista de usuarios y muestre la información de sus perfiles, explotando el API Rest pública de GitHub https://api.github.com/search/users?q=YOUR_NAME.

Requerimientos generales

1. La aplicación debe cumplir con los siguientes requisitos funcionales:
    - Crear una aplicación que incluya un campo de entrada texto y un botón, para que se pueda capturar el usuario y recuperar la información utilizando el API anteriormente indicada.
    - Mostrar los primeros 10 usuarios del resultado de la búsqueda, incluyendo su nombre de usuario ('user.login') y el id ('user.id') de cada registro.
    - Convertir cada Perfil de usuario en un enlace, para que al hacer clic en cada registro, navegue a una ruta que incluya la propiedad 'user.login' como parámetro.
    - Crear un componente independiente en el que se lea el parámetro de la URL, y a continuación, obtenga los datos de dicho usuario mediante la siguiente API: https://api.github.com/users/YOUR_NAME
    - Incluir la imagen del usuario ('avatar_url') y alguna otra información (de su elección) en el componente.
    - Incluir un validador que verifique que el texto de búsqueda de usuarios sea de un mínimo de 4 caracteres, y otro que NO permita realizar la búsqueda de la palabra “doublevpartners”.
    - Integrar cualquier librería de gráficos que pueda encontrar y crear un gráfico de barras simple para mostrar el número de seguidores de los 10 primeros usuarios.
    - Incluir un componente para mostrar mensajes de Error en toda la aplicación.

- *Si estás utilizando Angular,* agregar a la declaración del servicio que obtiene los datos un método que utilice Observables y otro Promises.
- *Si estás utilizando Angular,* agregar un Guard que no permita consultar el perfil de usuarios con un 'score' menor a 30.0.

2. CSS: Utilizar algún framework (a elección) para escribir los archivos CSS, tomando en cuenta la compatibilidad con distintos navegadores.
3. Iconos: Utilizar una librería para el manejo de iconos donde lo considere necesario (*se recomienda el uso de Font Awesome o Glyphicons.*)

Prueba Back End # 2

Queremos un API que nos permita crear, eliminar, editar y recuperar tickets con paginación. Que se pueda recuperar todos o filtrar por uno específico.

Los ticket tienen un id, un usuario, una fecha de creación, una fecha de actualización y un estatus (abierto/cerrado).

Exponer el servicio mediante http RESTFUL, valorable GRPC y GraphQL.

Puedes realizar la prueba en Go o Netcore. Ningún otro lenguaje será revisado.

Dejar en el readme la manera de poder probar en local. Utiliza el lenguaje de programación Go o NetCore. Tener en cuenta, para la ejecución en local, la base de datos (puedes utilizar un contenedor como Docker y eso te dará algunos puntos extra).

¡¡¡ Mucha suerte y gracias !!!