Prueba Técnica

Punto 1

Tomando como insumo los servicios web documentados en la pagina https://rickandmortyapi.com/documentation/

Cree una API REST con Spring Boot que cumpla con los siguientes requisitos:

- La petición debe recibir el número del episodio.
- La respuesta deberá ser como esta:


```
{
    "episode": 28,
    "episodeName" : "The Ricklantis Mixup",
    "characters": [
        {
            "name": "Rick Sanchez",
            "species": "Human",
            "gender": "Male",
            "image": "https://rickandmortyapi.com/api/character/avatar/1.jpeg",
            "location": {
                "name": "Earth (Replacement Dimension)",
                "type": "Planet",
                "dimension": "Replacement Dimension"
            }
        },
        {
            "name": "Morty Smith",
            "species": "Human",
            "gender": "Male",
            "image": "https://rickandmortyapi.com/api/character/avatar/2.jpeg",
            "location": {
                "name": "Earth (Replacement Dimension)",
                "type": "Planet",
                "dimension": "Replacement Dimension"
            }
        },
         //...
    ]
}
```

Punto 2

De acuerdo a la información del punto 1, diseñe un modelo entidad-relación que permita almacenar los datos de:

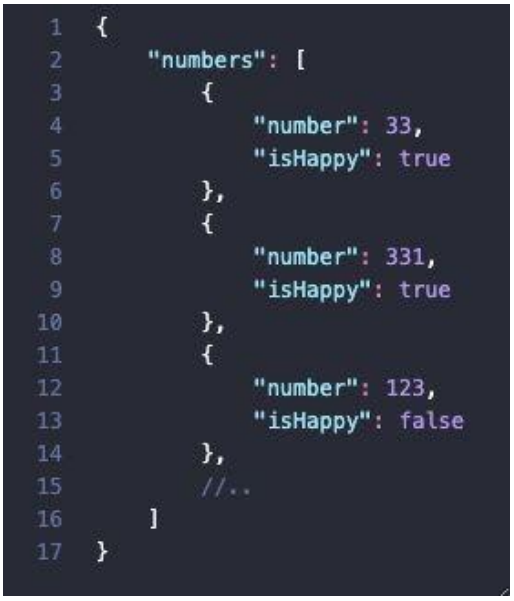
- Episodios.
- Personajes.
- Lugares.

Una vez diseñado el modelo, implemente el modelo en una base de datos de su preferencia y guarde los datos cada vez que se llame al API. Tenga en cuenta que no deberá registrar más de una vez la misma información.

Punto 3

Consultar y entender que es un número feliz; Y en el mismo proyecto Spring Boot donde desarrolló el punto 1, implemente una API que reciba un listado de números y responda para cada número si es feliz o no. Si hay un número que no cumple las validaciones se deberá indicar que “no es posible calcular si es feliz o no” pero el API no deberá arrojar ningún error. Maneje los códigos http correspondientes.

La respuesta se deberá ver similar a esta:



```
{
    "numbers": [
        {
            "number": 33,
            "isHappy": true
        },
        {
            "number": 331,
            "isHappy": true
        },
        {
            "number": 123,
            "isHappy": false
        },
        //..
    ]
}
```

Punto 4

En el mismo proyecto Spring Boot donde desarrolló el punto 1, cree otra API REST que permita calcular el resultado de la suma de números naturales hasta N, donde N es el parámetro hasta el cual debe calcular la sumatoria, y escriba una prueba unitaria que valide el algoritmo para los números N = 5 resultado esperado = 15 y N = 10 resultado esperado = 55

La respuesta se deberá ver similar a esta:

Bonus (Opcional)

Cada una de las siguientes tareas te darán puntos adicionales en tu calificación del reto. Ganarás los bonus si:

- Implementas pruebas unitarias para cada método o función implementada en los puntos anteriores.
- Transformas el modelo entidad-relación del punto 2 a un modelo no relacional.
- Despliegas las APIS anteriores en algún servicio cloud o hosting. (*Heroku, Digital ocean, Google Cloud, AWS, Azure o el que quieras*)

Entregables y condiciones

- Proyecto en un repositorio github público. Agregar como colaboradores a: Cristhian.Rodriguez@farmatodo.com, Jhon.Puentes@farmatodo.com.
- El repo deberá contener un Readme con instrucciones para consumir las APIs y ejecutar el proyecto en local u otras que considere necesario.
- Las imágenes de los modelos (relacional y/o no relacional) deberán estar en una carpeta del proyecto llamada /static/img en el mismo repositorio.
- Scripts SQL necesarios para crear base de datos y tablas. Este script deberá estar en una carpeta del proyecto llamada /static/sql en el mismo repositorio.