Universidad Técnica Federico Santa María
Departamento de Informática
1º semestre 2023

Estudiantes: Fernanda Araya Zarate 202173026-5
Rodrigo Pizarro Alegría 201910012-2
Fecha: 11/06/2023

Tarea 2 INF239: Bases de datos

1. **Link a repositorio git:** `https://github.com/Silent-math/Tareas-de-bases-de-datos.git`

2. **Link a documentación postman:** `https://documenter.getpostman.com/view/27654629/2s93sc5YFS`

3. **Manejo de excepciones:** Nuestra API maneja errores del tipo cliente (en particular error 404 «Error, objeto no encontrado»), errores del servidor (en particular error 500 «Error interno del servidor») y además incluimos (como se explicará mas adelante en los supuestos) un error de prisma asociado al `manejo de relaciones` En particular, en los delete, se impide la eliminación de elementos que estén relacionados con elementos de otras tablas. En particular, detectamos el error de prisma P2014 que se expresa como «The change you are trying to make would violate the required relation '{relation_name}' between the {model_a_name} and {model_b_name} models.» Este error nos parecía un poco largo así que lo redujimos a «Cannot delete {nombre_tabla} with associated relations»

4. **Supuestos:** En primer lugar, como se puede apreciar en la documentación de Postman, todo lo asociado a CRUD para tablas con dos PK,FK, solicitamos dos id a la hora de trabajar las operaciones asociadas a get one, update y delete. Esto se puede evidenciar en la documentación de Postman. El segundo supuesto es que asumiremos que las personas que utilicen la API estarán entendidas en el manejo que nosotros le damos (como utilizar el CRUD y los endpoints) y el manejo de excepciones que podrán haber son casos particulares que a la vez son muy comunes, y que le pueden ocurrir a cualquiera al interactuar con una base de datos (no encontrado, error interno o error al borrar elementos relacionados). Es bastante difícil intentar cubrir todos los casos factibles así que hicimos este supuesto. El supuesto final es que para el update, documentamos en Postman que solo actualizamos un atributo de la tabla, porque en general solo uno es el que realmente se podría cambiar, ya que cambiar los otros carece un poco de sentido. Por ejemplificar, no parece razonable cambiar dentro de la tabla personajes su fecha de nacimiento, pero si se podría cambiar el objeto o la fuerza. Para finalizar, para el último supuesto se puede considerar que decidimos descomponer la relación muchos a muchos mediante una tabla llamada reino_tiene_defensa que cumple la misma estructura y función que las tablas compuestas que ya tiene el modelo de por si.

5. **Evidencias de participación:** Fernanda Araya se encargó de la documentación en Postman y la confección del informe. Rodrigo Pizarro se encargó de la creación de controllers y endpoints de la API. El modelado en la base de datos y el rellenado se hizo en conjunto de los dos estudiantes ya que bajo nuestro criterio esta parte fue la que nos trajo mayor cantidad de problemas. En general, crear los controllers (una vez el modelo de datos estaba completamente correcto) era repetir lo mismo para cada tabla modificando pequeños detalles y documentar en Postman seguía más o menos la misma idea, también repetir para cada tabla y endpoint. En el repositorio también se puede ver quien subió cada cosa, aunque realmente subimos todo al mismo tiempo prácticamente cuando ya teniamos todo listo.