

Prueba Técnica Desarrollador Junior para Sectorial S.A

Por: Diego Medina Velásquez

The screenshot displays a web application interface for managing a hierarchical structure. At the top, under the heading "Categorías", there is a form with a "Nueva Categoría" input field and an "Agregar Categoría" button. Below this, a section titled "Categoría 1" contains "Borrar Categoría" and "Desactivar categoría" buttons. Under "Categoría 1", the "Subcategorías" section includes a "Nueva Subcategoría" input field and an "Agregar Subcategoría" button. Two subcategory blocks are shown: "Subcategoría 1.1" and "Subcategoría 1.2". Each block has "Borrar Subcategoría" and "Desactivar subcategoría" buttons. Inside each subcategory block, there is a "Temas" section with a "Nuevo tema" input field and an "Agregar tema" button. Subcategory 1.1 contains two topic entries: "Tema 1.1.1" and "Tema 1.1.2", each with "Borrar" and "Desactivar" buttons. Subcategory 1.2 contains one topic entry: "Tema 1.2.1", also with "Borrar" and "Desactivar" buttons. The interface uses a light blue background with white borders for the main sections and colored borders (purple for subcategories, green for topics) to distinguish the nested levels.

Frontend

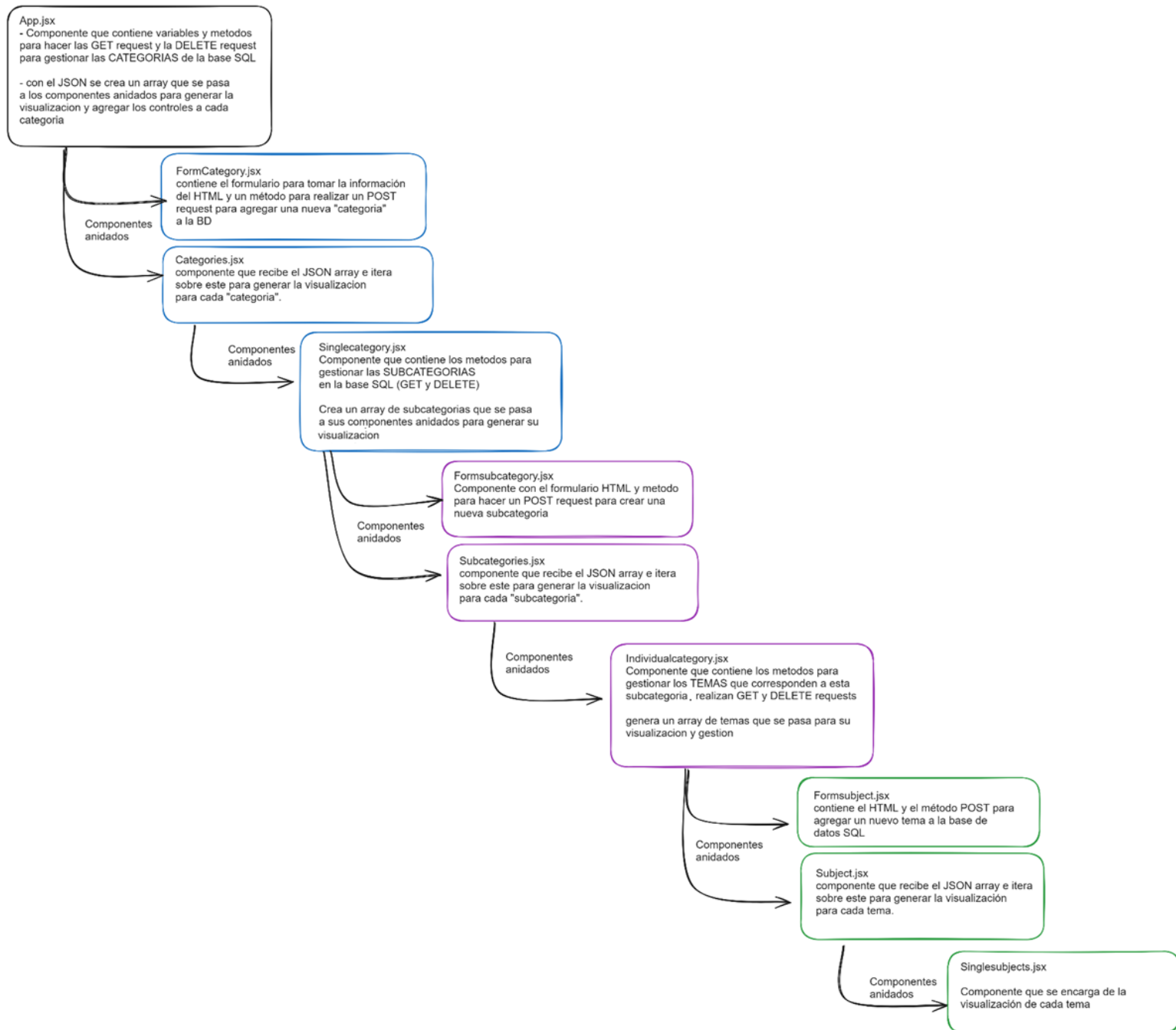
Para la realización del frontEnd se utilizaron las siguientes herramientas:

- React 18.2.0
- Tailwind 3.3.5 para la aplicación de estilos CSS
- Axios 1.6.2 para realizar HTTP request a la API que se encarga de gestionar la base de datos
- Nanoid para la numeración automática de componentes

El concepto de la app es crear componentes anidados que se encargan de gestionar las tablas que contienen la información de la base de datos, hacer las operaciones CRUD para obtener y publicar información y pasar objetos JSON a componentes de JSX para generar el HTML a visualizar.

Estructura de App y componentes

La siguiente figura muestra un esquema de la estructura de los componentes



Funciones

1. Categorías

Se puede agregar categorías para generar una nueva estructura jerárquica. Cada categoría se gestionará independientemente y contiene sus propios controles para “borrar categoría” y de “desactivar categoría”

Categorías

Categoría 1

Subcategorías

Categoría 2

Subcategorías

2. Subcategorías

En cada categoría deseada puede agregar una subcategoría que quedará relacionada a la categoría correspondiente. La base de datos está construida usando el modelo Snowflake, de modo que cada subcategoría está relacionada a la categoría a la cual pertenece mediante “foreigns keys” que mantienen su asociación. También puede borrar y desactivar subcategorías usando los controles propios

Categorías

Categoría 1

Subcategorías

Subcategoría 1.1

Temas

Subcategoría 1.2

Temas

Categoría 2

Subcategorías

3. Temas

De manera similar puede agregar temas a cada subcategoría, manteniendo estructuras jerárquicas independientes. También cuenta con controles para “inactivar tema” y “borrar tema”

Categorías

Nueva Categoría

Agregar Categoría

Categoría 1

Borrar Categoría

Desactivar categoría

Subcategorías

Nueva Subcategoría

Agregar Subcategoría

Subcategoría 1.1

Borrar Subcategoría

Desactivar subcategoría

Temas

Nuevo tema

Agregar tema

Tema 1.1.1

Borrar

Desactivar

Tema 1.1.2

Borrar

Desactivar

Subcategoría 1.2

Borrar Subcategoría

Desactivar subcategoría

Temas

Nuevo tema

Agregar tema

Tema 1.2.1

Borrar

Desactivar

Categoría 2

Borrar Categoría

Desactivar categoría

Subcategorías

Nueva Subcategoría

Agregar Subcategoría

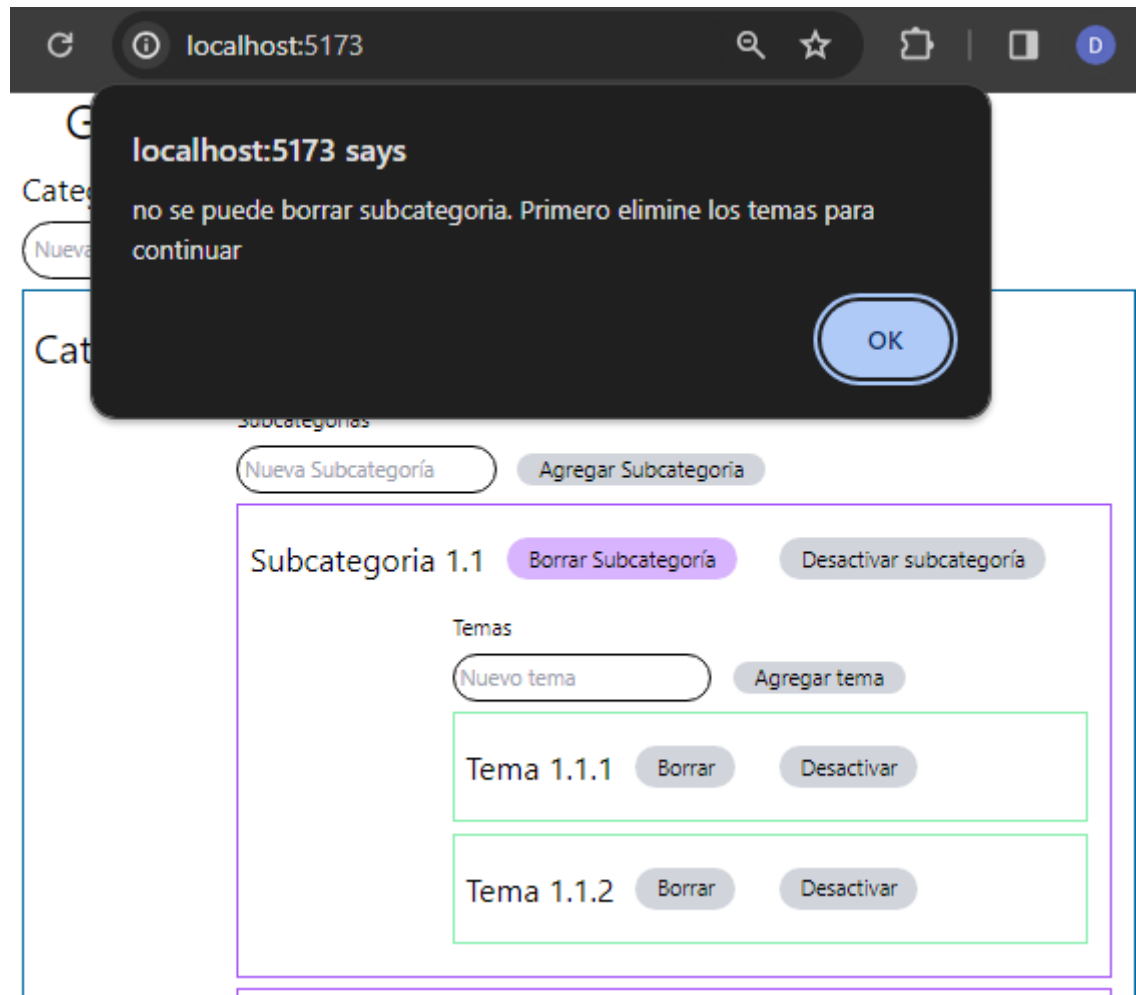
4. Control de “desactivar”

Cuando presiona, desaparece el formulario de ingreso de nuevos campos y también la posibilidad de borrar las subestructuras anidadas.

Campos desactivados	Campos activados
<div><div>Categoría 1</div><div><div>Activar categoría</div></div><div>Subcategorías</div><div><div>Subcategoría 1.1</div><div><div>Activar subcategoría</div></div><div>Temas</div><div><div>Tema 1.1.1</div><div>Activar</div></div><div><div>Tema 1.1.2</div><div>Activar</div></div></div><div><div>Subcategoría 1.2</div><div><div>Activar subcategoría</div></div><div>Temas</div><div><div>Tema 1.2.1</div><div>Activar</div></div></div></div>	

5. Control de “borrar”

Permite eliminar cualquier nivel, siempre y cuando no tenga entradas anidadas en ella.
Ej.: la subcategoría 1.1 no se puede eliminar ya que contiene el “tema 1.1.1” y el “tema 1.1.2”



Backend

Para la realización de la API que controla la base de datos se utilizaron las siguientes herramientas:

1. Express 4.18.2 para la creación de un servidor usando Node.js
2. Mysql2 3.6.5 para gestionar una base de datos SQL
3. Sequelize 6.35.1 para generar SQL queries, que faciliten la creación de controladores que realicen las funciones CRUD para gestionar la base de datos
4. Dotenv para almacenar la información sensible de la base de datos, como nombre y contraseña
5. Cors para permitir el acceso del Frontend operando en otro puerto

Base de Datos SQL

Como la información tiene una estructura jerárquica, se usa una base de datos configurada de manera snowflake, donde las relaciones de las tablas se definen con “foreign keys” como se muestra en la figura.

Tabla de categorías

id_key	CategoryName
17	Categoría 1
18	Categoría 2
NULL	NULL

la id_key de una tabla es foreign key en su subcategoría para mantener la relación jerárquica

Tabla de Subcategorías

id_key	SubcategoryName	CategoryIdKey
11	Subcategoría 1.1	17
12	Subcategoría 1.2	17
NULL	NULL	NULL

Tabla de Temas

id_key	SubjectName	SubcategoryIdKey
16	Tema 1.1.1	11
17	Tema 1.1.2	11
18	Tema 1.2.1	12
NULL	NULL	NULL

Cada entrada nueva de “Subcategoría” o de “Tema”, lleva el Id_key del nivel superior a sí misma, de modo que se pueda seguir la relación. Ej. el tema 16 pertenece a la subcategoría 11 que a su vez pertenece a la categoría 17

Servidor Express y rutas

El siguiente diagrama muestra el diseño de la API que se encargará de la gestión de la base de datos

