Informe FullStack

Fecha: 18-05-2025

Anderson Diaz
Gerardo Bustos
Fernanda Orellana

Introducción

En este informe se analizará el enfoque del proyecto “Neurotecno” y su código, resaltando la implementación de los microservicios y como estos se comunicaran con las API’s a través de diferentes métodos como Crear, Leer, Actualizar y Borrar (C.R.U.D.), además de la creación de una base de datos creada en MySQL con diferentes tablas que estén relacionadas entre sí, de igual forma, se explicara algunas buenas prácticas como también explicar las funciones para verificar que los datos sean correctos y que se almacenen apropiadamente en la base de datos.

Implementación de Microservicios

A.- Se utilizarán diferentes microservicios, tales como:

1. Microservicio de Atención: Este microservicio contará con diferentes atributos:: id, fecha atención, hora atención, costo, comentario, básicamente, este microservicio es el encargado de gestionar las atenciones médicas a través de C.R.U.D.

2. Microservicio de Médico: Sus atributos son: id, run, nombre completo, especialidad, jefe turno, ciertos atributos se usarán para identificar(id run y nombre completo) y los otros (especialidad y jefe turno) se utilizan para categorizar la atención y así, definir un psicólogo especialista en cada área.

3. Microservicio de paciente: Contará con los siguientes atributos: id, run, nombres, apellidos, fecha de nacimiento, correo, se utilizará para llevar un registro de los pacientes al igual que una vía de comunicación y contacto.

4. Microservicio de tipo de usuario: Sus únicos atributos serán id y nombre, esta se utilizará para diferenciar a los usuarios que accedan y sus roles.

B.- Para comunicar los microservicios a través de API's se utilizarán diferentes métodos para esto, ocuparemos la aplicación "Postman" para utilizar los métodos, los cuales son:

1. GET: Este método será utilizado para solicitar datos desde nuestro servidor, por ejemplo el id de un usuario o su nombre. Lo usamos solamente para leer o consultar información.

2. POST: A diferencia del método anterior, este sí modifica los datos del servidor. Será utilizado para enviar datos al servidor con el objetivo de crear un nuevo recurso (por ejemplo, agregar un nuevo usuario).

3. PUT: La función de este método es de actualizar por completo un recurso en particular, por ejemplo un usuario desea cambiar su información personal (correo, teléfono etc), esta solicitud se realiza y se reemplazará el recurso existente con los nuevos datos enviados.

4. PATCH: El método es muy similar al anterior, pero este se utiliza para actualizar parcialmente un recurso existente. A diferencia del PUT, no reemplaza todo el recurso, sino que modifica solo los campos que envías.

5. DELETE: Se usará para eliminar un recurso existente del servidor.

Base de datos

A.- Para la realización de este proyecto, se utilizará un sistema de gestión de bases de datos en MySQL. Se trabajará con una única base de datos dominada “db_neurotecno”, dentro de la cual se crearán múltiples tablas, una para cada microservicio que compone la arquitectura del sistema. Estas tablas estarán diseñadas de forma modular para mantener la independencia lógica de cada microservicio, pero se establecerán relaciones entre éstas mediante claves foráneas, permitiendo la correcta unión de los datos cuando sea necesario. Asimismo, se aplicarán buenas prácticas de normalización para optimizar el rendimiento de las consultas y evitar la redundancia de datos.

B.- Para verificar que los datos sean correctos y se almacenen apropiadamente, se definirán el tipo de datos apropiado para cada atributo (por ejemplo, Integer para id), se utilizará “nullable = false” para evitar enviar información incompleta en ciertos atributos, así como ciertos manejos de errores y de validaciones como en el siguiente ejemplo:

```
( if (atencionActualizada == null) {
                return ResponseEntity.notFound().build();)
```

También, se realizó una solicitud GET en ‘Postman’(esta aplicación ayuda a poder validar toda la api, incluyendo la base de datos), por su ID a los recursos creados para confirmar que los datos se guardaron correctamente en el servidor y verificar que los datos devueltos coincidan con los que se envían. Además, se comprobará que la respuesta del servidor tenga un código de estado exitoso como el código 200 (OK) o el 201 (Created).

Tabla de Tipo de Usuario:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	nombre	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

Tabla de Paciente:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	apellidos	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	correo	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	fecha_nacimiento	datetime(6)			Yes	NULL			Change Drop More
5	nombres	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
6	run	varchar(13)	utf8mb4_general_ci		No	None			Change Drop More
7	tipo_usuario	int(11)			No	None			Change Drop More

Tabla de Médico:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	comentario	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	costo	int(11)			No	None			Change Drop More
4	fecha_atencion	date			No	None			Change Drop More
5	hora_atencion	time(6)			No	None			Change Drop More
6	medico_id	int(11)			No	None			Change Drop More
7	paciente_id	int(11)			No	None			Change Drop More
8	tipousuario_id	int(11)			No	None			Change Drop More

Tabla de Atención:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	especialidad	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	jefe_turno	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	nombre_completo	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	run	varchar(13)	utf8mb4_general_ci		No	None			Change Drop More

Conclusión

El informe presenta una implementación de código de un proyecto basada en los microservicios y como este se puede relacionar a una API, así como también el uso correcto de datos y su verificación, el agregado de métodos para realizar el C.R.U.D. (Crear, Leer, Actualizar, Eliminar) y seguir las buenas prácticas.