

4. Api Rest

A) Para este proyecto cree un total de 13 end points, 4 para cada uno de los cruds de las 4 tablas, adicionalmente, un end point para consultar los **Devices** para un **Bus** especifico. Estos end points se pueden consumir de la siguiente forma:

Nota: Acostumbro a retornar un Map con **“exito”** un boolean true o false si se logro hacer el proceso satisfactoriamente o no. **“mensaje”** un String vacio si fue exitoso, o con un mensaje de que error ocurri , si es un error controlado. Y el tipo de dato espero, con el valor del objeto esperado, adem s si ocurre un error inesperado un **“error”** con el mensaje del error capturado.

ConcessionaireController:

METODO	URL BASE	END POINT	DESCRIPCION
Get	/concessionaire	“”	Devuelve una lista con todos los registros en la tabla Concessionaire.
Get		“/{id}”	Devuelve (si existe) un objeto registrado en la tabla Concessionaire con ese ID.
Post		“”	Recibe en el body un objeto de tipo Concessionaire, si el ID = 0 lo registra, si es diferente hace un edit.
Delete		“/{id}”	Elimina (Si existe) el registro del Concessionaire con ese ID y los registros de Bus que tengan la FK de dicho Concessionaire.

BusController:

METODO	URL BASE	END POINT	DESCRIPCION
Get	/bus	“”	Devuelve una lista con todos los registros en la tabla Bus.
Get		“/{id}”	Devuelve (si existe) un objeto registrado en la tabla Bus con ese ID.
Post		“”	Recibe en el body un objeto de tipo DeviceType, si el ID = 0 lo registra, si es diferente hace un edit.
Delete		“/{id}”	Elimina (Si existe) el registro del Bus con ese ID y los registros de Device que tengan la FK de dicho Bus.

DeviceTypeController:

METODO	URL BASE	END POINT	DESCRIPCION
Get	/deviceType	""	Devuelve una lista con todos los registros en la tabla DeviceType.
Get		"/{id}"	Devuelve (si existe) un objeto registrado en la tabla DeviceType con ese ID.
Post		""	Recibe en el body un objeto de tipo DeviceType, si el ID = 0 lo registra, si es diferente hace un edit.
Delete		"/{id}"	Elimina (Si existe) el registro del DeviceType con ese ID y los registros de Device que tengan la FK de dicho DeviceType.



DeviceController:

METODO	URL BASE	END POINT	DESCRIPCION
Get	/device	""	Devuelve una lista con todos los registros en la tabla Device.
Get		"/{id}"	Devuelve (si existe) un objeto registrado en la tabla Device con ese ID.
Post		""	Recibe en el body un objeto de tipo Device, si el ID = 0 lo registra, si es diferente hace un edit.
Post		"deviceForBus"	Recibe en el Body un objeto del tipo Bus, y retorna una lista con todos los Device que esten asociados a dicho Bus.
Delete		"/{id}"	Elimina (Si existe) el registro del Device con el ese ID.

B) Para que el front-end pueda consumir el query que retorna la lista de **Devices** para un **Bus** especifico deberá hacerlo a través de un método POST, al url base de lo que busca como respuesta **Device** y al end point construido para dicha solicitud **"deviceForBus"**, quedando de la siguiente manera: (Este end point espera un objeto del tipo **Bus** en el requeste body, no se debe olvidar)

POST ▼	localhost:8080/device/deviceForBus
--------	------------------------------------

Pruebas Unitarias:

busesApi (30/05/2020 7:49:09 p. m.)				
Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
>  busesApi	 41,1 %	818	1.171	1.989

Realice las pruebas Unitarias con Junit 4, en estas pruebas realice los Mock de varios servicios. Y solo logre un 41% de cobertura del código por temas de tiempo, pero con las pruebas realizadas se logra evidenciar el funcionamiento de las pruebas.