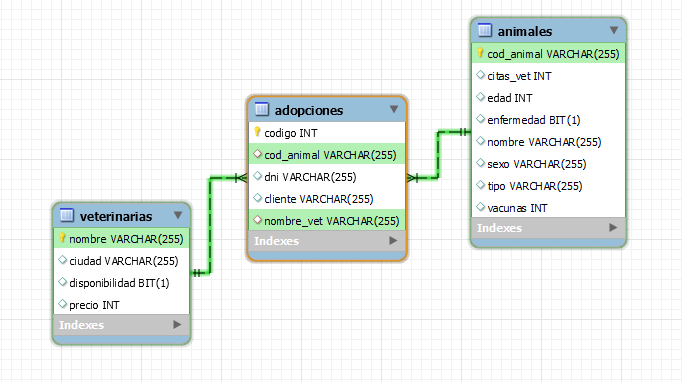
**Creación de microservicios protectora animal**

Se pretende crear una aplicación para una veterinaria, basada en microservicios y con acceso a una base de datos “protectora”:



**Servicio de animal :8080**

A partir de la tabla de animales que incluye los campos: codAnimal (String), tipo, nombre, sexo, edad, enfermedad (booleana), vacunas y citasVeterinario, se crean los siguientes recursos:

* **GET** \_ Devolver una lista con todos los animales

<http://localhost:8080/animales>

* **GET** \_ Con URL “animal/{código}” Devolver datos de un animal a partir de su código

<http://localhost:8080/animal/LOLA02>

* **GET** \_ Con URL “enfermo/true” devuelve una lista de los animales enfermos

http://localhost:8080/enfermo/false

* **DELETE \_** Con URL “liberar/{código}” No devuelve nada, libera a un animal de la protectora.

(DELETE) http://localhost:8080/liberar/LOLA01

* **POST** \_ Mediante body JSON se inscribe un animal nuevo a la protectora animal

(POST) http://localhost:8080/inscribir

* **PUT/PATCH**\_ Con URL “curar/{codAnimal}” si el animal está enfermo, se cura y devuelve un mensaje (String); si no lo está devolverá un mensaje indicándolo.

(PATCH) http://localhost:8080/curar/LOLA02

**Servicio de veterinaria :9090**

Actúa sobre una tabla de veterinaria con los siguientes campos: nombreVet (String), ciudad, precio y disponibilidad (booleana). Los recursos son:

* **GET**\_ Devuelve una lista de veterinarias disponibles

http://localhost:9090/veterinarias/false

* **GET**\_ Con URL “precio/{precioLimite}” Devuelve una lista de veterinarias que no sobrepasen el precio indicado.

http://localhost:9090/precio/50

* **PUT/PATCH**\_ Con URL “disponibilidad /{nombreVet}/{disponibilidad}” Actualiza los datos de la veterinaria.

(PATCH) http://localhost:9090/disponibilidad/PRADO/true

**Servicio de adopción :8989**

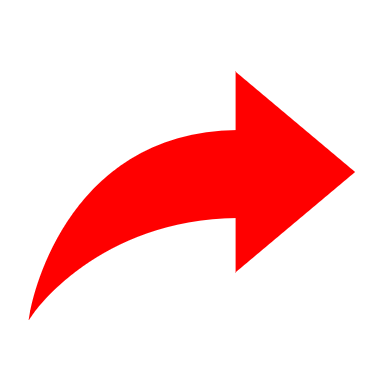
Utiliza una tabla de adopción con los siguientes campos: idAdopcion (autoincremental), nombreCliente, dni, codAnimal y nombreVet. Tendrá los siguientes recursos:

* **POST**\_ Mediante body JSON se crea el informe de adopción y se registra en la BD. Interacciona con el servicio de animales para actualizar una vez se haga la adopción (desaparece su registro del array animales)

(POST) http://localhost:8989/adoptar

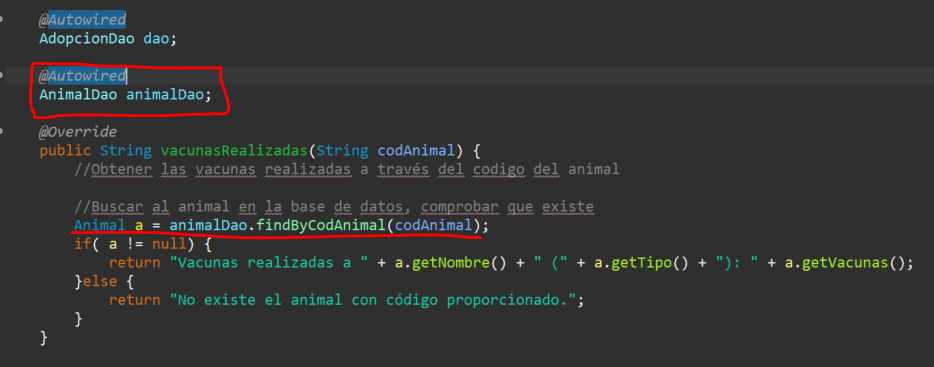
* **GET**\_ Con URL “adopcion/{ codAnimal }” devolverá las vacunas realizadas al animal recibido como parámetro (codAnimal). Interacciona con el servicio animal para conocer las vacunas a partir del código del animal.

<http://localhost:8989/vacunas/LOLA01>

**Correción**

**ResponseEntity para poder conseguir al animal (**se llama al método findByCodAnimal de AnimalDao en el AdopcionServiceImpl**).**

**Se buscará al animal cuyo código coincida en la base de datos.**



* **GET**\_ Devuelve una lista de las adopciones realizadas

http://localhost:8989/adopciones