**Environment**

- Java version: 11

- Maven version: 3.8.6

- Spring Boot version: 2.2.1.RELEASE

# **Funcionamiento**

La API REST debe exponer dos endpoints, a saber:

**1) GET request - `/isAlive`**

http response = 200.

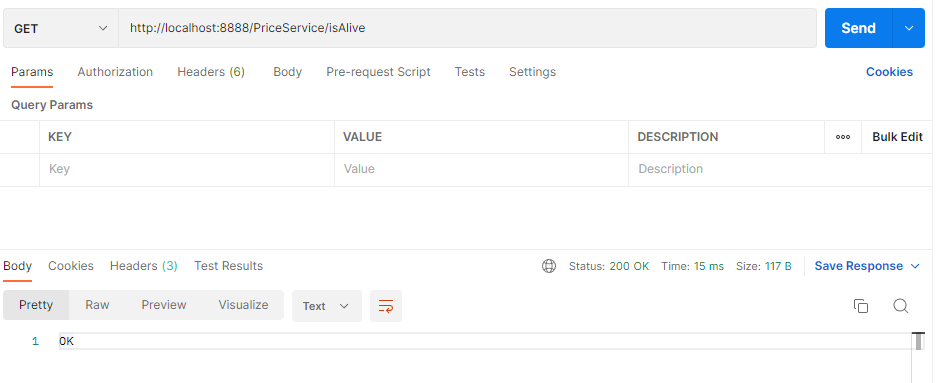
La solicitud fue exitosa. Se devuelve un campo "string" igual al valor "OK" indicando que el servicio 'está vivo'

**Ejemplo**:

Request: <http://localhost:8888/PriceService/isAlive>

Response: OK (Status 200)

Captura Postman:



**2)** **GET request - `/getPrice`:**

http response = 200.

La solicitud fue exitosa. Se devuelve JSON con los siguientes atributos:

* productId: identificador del producto solicitado
* brandId: identificador de la cadena solicitada
* priceList: identificador de la tarifa a aplicar
* startDate: fecha inicial dentro del rango en el cual se aplica la tarifa
* endDate: fecha final dentro del rango en el cual se aplica la tarifa
* finalPrice: precio final de venta

**Ejemplo**:

Request: http://localhost:8888/PriceService/getPrice?dateApply=2020-06-17 21:00:00&brandId=1&productId=35455

Response: Status 200

{

    "productId": 35455,

    "brandId": 1,

    "priceList": 4,

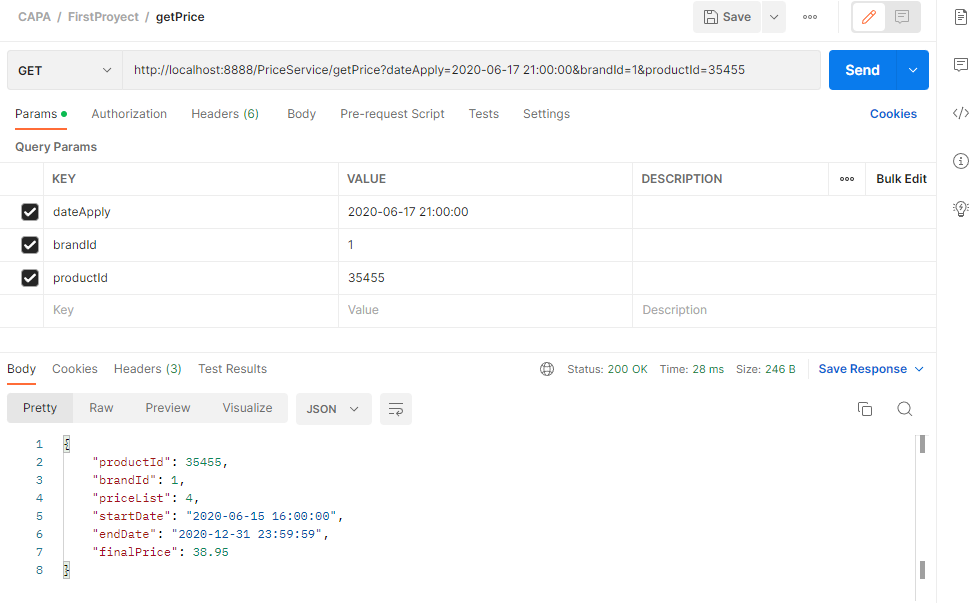
    "startDate": "2020-06-15 16:00:00",

    "endDate": "2020-12-31 23:59:59",

    "finalPrice": 38.95

}

Captura Postman:



# **Datos Iniciales**

La base de datos H2 (en memoria) será inicializada cada vez que se inicie la aplicación, con los siguientes datos (archivo data.sql):

INSERT INTO PRICES (ID, BRAND\_ID, START\_DATE, END\_DATE, PRICE\_LIST, PRODUCT\_ID, PRIORITY, PRICE, CURR)

VALUES (1, 1, parsedatetime('2020-06-14 00:00:00', 'yyyy-MM-dd hh:mm:ss'), parsedatetime('2020-12-31 23:59:59', 'yyyy-MM-dd hh:mm:ss'), 1, 35455, 0, 35.50, 'EUR');

INSERT INTO PRICES (ID, BRAND\_ID, START\_DATE, END\_DATE, PRICE\_LIST, PRODUCT\_ID, PRIORITY, PRICE, CURR)

VALUES (2, 1, parsedatetime('2020-06-14 15:00:00', 'yyyy-MM-dd hh:mm:ss'), parsedatetime('2020-06-14 18:30:00', 'yyyy-MM-dd hh:mm:ss'), 2, 35455, 1, 25.45, 'EUR');

INSERT INTO PRICES (ID, BRAND\_ID, START\_DATE, END\_DATE, PRICE\_LIST, PRODUCT\_ID, PRIORITY, PRICE, CURR)

VALUES (3, 1, parsedatetime('2020-06-15 00:00:00', 'yyyy-MM-dd hh:mm:ss'), parsedatetime('2020-06-15 11:00:00', 'yyyy-MM-dd hh:mm:ss'), 3, 35455, 1, 30.50, 'EUR');

INSERT INTO PRICES (ID, BRAND\_ID, START\_DATE, END\_DATE, PRICE\_LIST, PRODUCT\_ID, PRIORITY, PRICE, CURR)

VALUES (4, 1, parsedatetime('2020-06-15 16:00:00', 'yyyy-MM-dd hh:mm:ss'), parsedatetime('2020-12-31 23:59:59', 'yyyy-MM-dd hh:mm:ss'), 4, 35455, 1, 38.95, 'EUR');

# **Build**

Para compilar el servicio, ejecutar la siguiente línea de comando:

mvn clean install

# **Ejecución Tests**

Para ejecutar los tests unitarios, seguir la siguiente línea de comando:

mvn test

# **Ejecución Servicio**

El servicio Rest corre como StandAlone.

Línea de comando a ejecutar:

java -jar price-service-rest.jar

Se ejecuta sobre el puerto 8888 y en el contextPath "/ PriceService"

# **URLs**

1) **http://localhost:8888/PriceService/isAlive** --> reemplazar localhost por el nombre del server correspondiente.

Parámetro:

- ninguno

2) **http://localhost:8888/PriceService/getPrice?dateApply= ‘fecha\_a\_aplicar'&brandId=’id\_cadena’&productId=’id\_producto’**

Parámetros:

- **dateApply**: fecha a aplicar / Obligatorio / Formato válido yyyy-MM-dd hh:mm:ss

En caso que no se envíe la hora se tomará por defecto la hora 00:00:00

- **productId**: identificador del producto / Obligatorio

- **brandId**: identificador de la cadena /Opcional.

Ejemplos:

http://localhost:8888/PriceService/getPrice?dateApply=2020-06-17 21:00:00&brandId=1&productId=35455

http://localhost:8888/PriceService/getPrice?dateApply=2020-06-14 10:00:00&brandId=1&productId=35455

# **Actuator**

El servicio cuenta con dos funcionalidades adicionales:

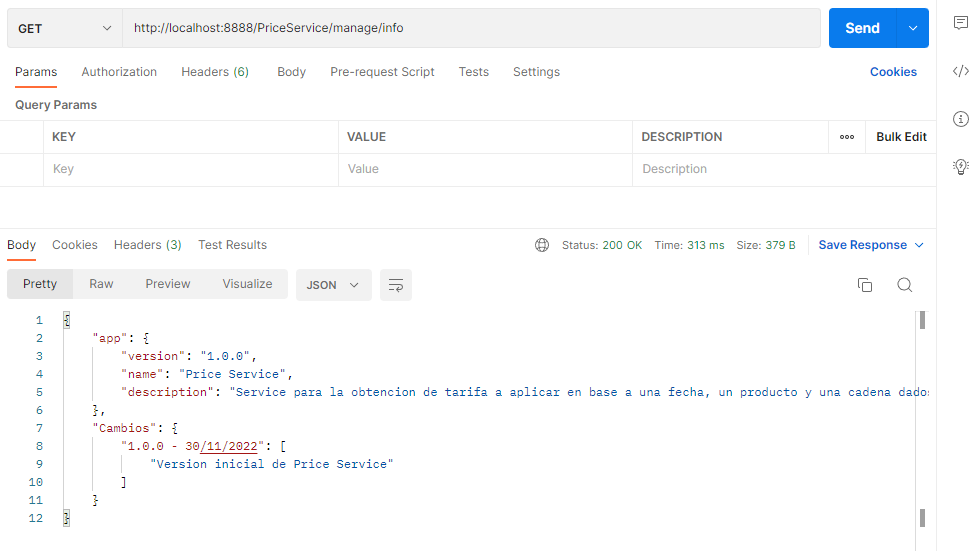
1. INFO

Método: GET

Url: <http://localhost:8888/PriceService/manage/info>

El cual provee información del servicio en sí y su versionado

Captura Postman



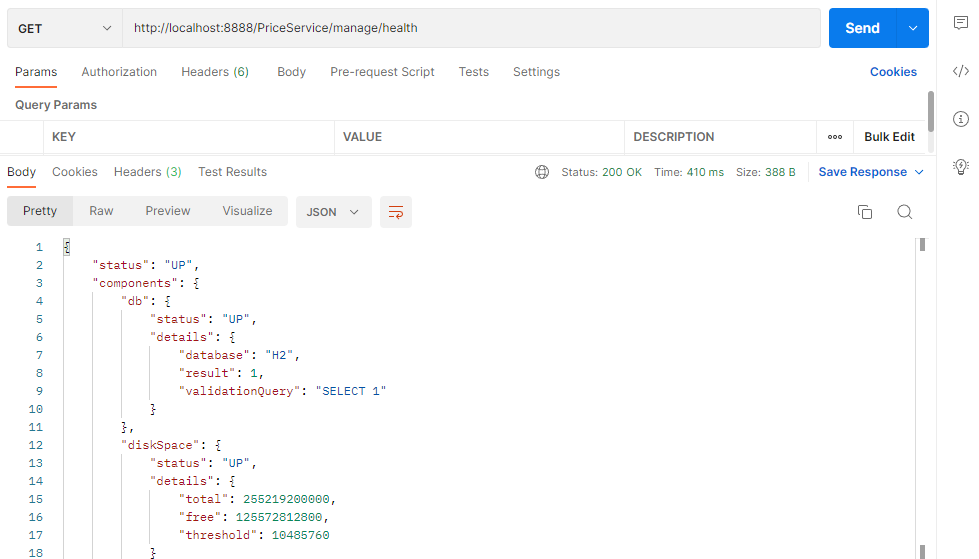
1. HEALTH

Método: GET

Url: <http://localhost:8888/PriceService/manage/health>

El cual provee información en cuanto al estado general del servicio e infraestructura

Captura Postman



# **Swagger**

Url: <http://localhost:8888/PriceService/swagger-ui.html>



