

API REST

PRACTICA DE MICROSERVICIOS Y CONTENEDORES

Lucero Estefania Moreno Ortiz
Lucero.e.moreno.ortiz@gmail.com

Contenido

INTRODUCCIÓN	2
Información general de la API	2
Pruebas realizadas con cliente Rest	3
Listado.	3
Producto por ID	4
Crear un producto	4
Modificar un producto	5
Eliminar un producto.....	5

INTRODUCCIÓN

Este documento contiene toda la información documentada sobre la API REST solicitada por parte de SPS Solutions. En la primera parte se da a conocer datos generales de la API, además de mostrar las pruebas documentadas desde un cliente REST.

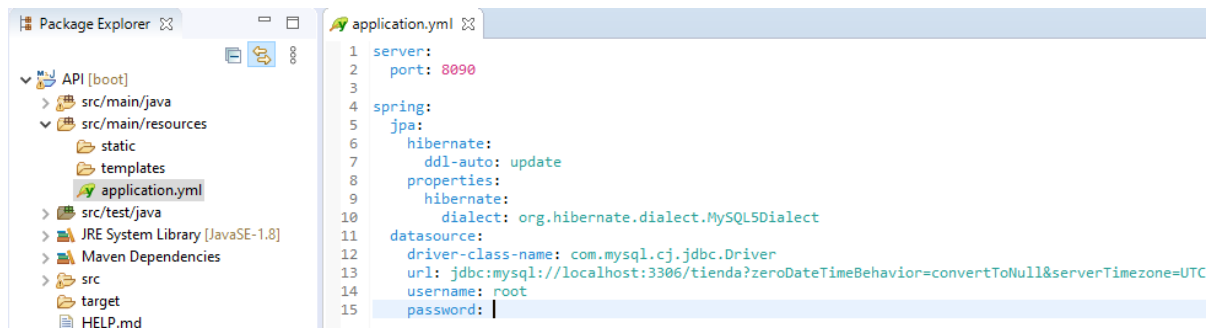
Información general de la API.

La API se realizó mediante el lenguaje Java, utilizando el framework Spring Boot y el IDE Spring Tool versión 4, en cuanto la base de datos se utilizó MySQL, y el script se encuentra en el repositorio.

El formato de los mensajes es tipo JSON y la estructura es la siguiente:

```
{
  "id": 9,
  "nombre": "Producto modificado",
  "precio": 17,
  "tipo": "A"
}
```

Se tendrá que modificar los datos de conexión en el archivo “application.yml”



Pruebas realizadas con cliente Rest

El cliente Rest utilizado para realizar estas pruebas es Talend API Tester en su versión gratuita y como extensión de Chrome. Las pruebas se mostrarán por módulos, desplegando la URL utilizada y la respuesta obtenida.

Listado.

A continuación, se muestra la prueba realizada para obtener el listado de productos de una tienda.

The screenshot displays the Talend API Tester interface. At the top, the request configuration is shown with the method 'GET' and the URL 'http://localhost:8090/api/sps/helloworld/v1/listado'. Below this, the 'QUERY PARAMETERS' section is visible. The response section shows a status code of '200'. The 'HEADERS' tab is active, displaying the following headers: 'Content-Type: application/json', 'Transfer-Encoding: chunked', 'Date: Fri, 17 Jan 2020 20:09:12 GMT', 'Keep-Alive: timeout=60', and 'Connection: keep-alive'. The 'BODY' tab is also active, showing a JSON array of five product objects. The JSON is formatted in a 'pretty' view. The response length is indicated as 271 bytes.

```
[{"id": 1, "nombre": "Producto 1", "tipo": "C", "precio": 20}, {"id": 2, "nombre": "Producto 2", "tipo": "B", "precio": 13}, {"id": 3, "nombre": "Producto 3", "tipo": "C", "precio": 20}, {"id": 4, "nombre": "Producto 4", "tipo": "A", "precio": 12}, {"id": 5, "nombre": "Producto 5", "tipo": "C", "precio": 16}]
```

Producto por ID

A continuación, se muestra la prueba realizada para obtener el producto buscado por ID de una tienda.

The screenshot shows a REST client interface. The METHOD is set to GET. The URL is `http://localhost:8090/api/sps/helloword/v1/producto/1`. The Send button is visible. Below the URL bar, there are tabs for QUERY PARAMETERS, HEADERS, and BODY. The HEADERS tab is active, showing a form to add headers. The BODY tab is also visible, showing a message: "XHR does not allow payloads for GET request." Below the request details, the Response section is displayed. It shows a status code of 200. The Response headers are listed: Content-Type: application/json, Transfer-Encoding: chunked, Date: Fri, 17 Jan 2020 20:13:28 GMT, Keep-Alive: timeout=60, Connection: keep-alive. The Response body is shown in a pretty-printed JSON format:

```
{  id: 1,  nombre: "Producto 1",  tipo: "C",  precio: 20}
```

Crear un producto

A continuación, se muestra la prueba realizada para crear un producto.

The screenshot shows a REST client interface. The METHOD is set to POST. The URL is `http://localhost:8090/api/sps/helloword/v1/crear/`. The Send button is visible. Below the URL bar, there are tabs for QUERY PARAMETERS, HEADERS, and BODY. The HEADERS tab is active, showing a form to add headers. The BODY tab is also visible, showing a message: "XHR does not allow payloads for GET request." Below the request details, the Response section is displayed. It shows a status code of 200. The Response headers are listed: Content-Type: application/json, Transfer-Encoding: chunked, Date: Fri, 17 Jan 2020 20:19:41 GMT -1s, Keep-Alive: timeout=60, Connection: keep-alive. The Response body is shown in a pretty-printed JSON format:

```
{  id: 9,  nombre: "Producto nuevo",  tipo: "B",  precio: 12}
```

Modificar un producto

A continuación, se muestra la prueba realizada para modificar un producto.

The screenshot shows a REST client interface with the following details:

- METHOD:** PUT
- URL:** http://localhost:8090/api/sps/helloworld/v1/modificar/
- QUERY PARAMETERS:** (empty)
- HEADERS:** Content-Type: application/json
- BODY:**

```
{
  "id": 9,
  "nombre": "Producto modificado",
  "precio": 17,
  "tipo": "A"
}
```

Response: 200 OK. Cache Detected - Elapsed Time: 75ms.

Response Headers: Content-Type: application/json, Transfer-Encoding: chunked, Date: Fri, 17 Jan 2020 20:22:58 GMT -1s, Keep-Alive: timeout=60, Connection: keep-alive.

Response Body:

```
{
  id: 9,
  nombre: "Producto modificado",
  tipo: "A",
  precio: 17
}
```

Eliminar un producto

A continuación, se muestra la prueba realizada para eliminar un producto.

The screenshot shows a REST client interface with the following details:

- METHOD:** DELETE
- URL:** http://localhost:8090/api/sps/helloworld/v1/eliminar/9
- QUERY PARAMETERS:** (empty)
- HEADERS:** (empty)
- BODY:** XHR does not allow payloads for DELETE request.

Response: 200 OK. Cache Detected - Elapsed Time: 63ms.

Response Headers: Content-Length: 0 byte, Date: Fri, 17 Jan 2020 20:25:01 GMT, Keep-Alive: timeout=60, Connection: keep-alive.

Response Body: No Content