DOCUMENTACIÓN DE API RESTFUL - SERVIDOR DE PRODUCTOS

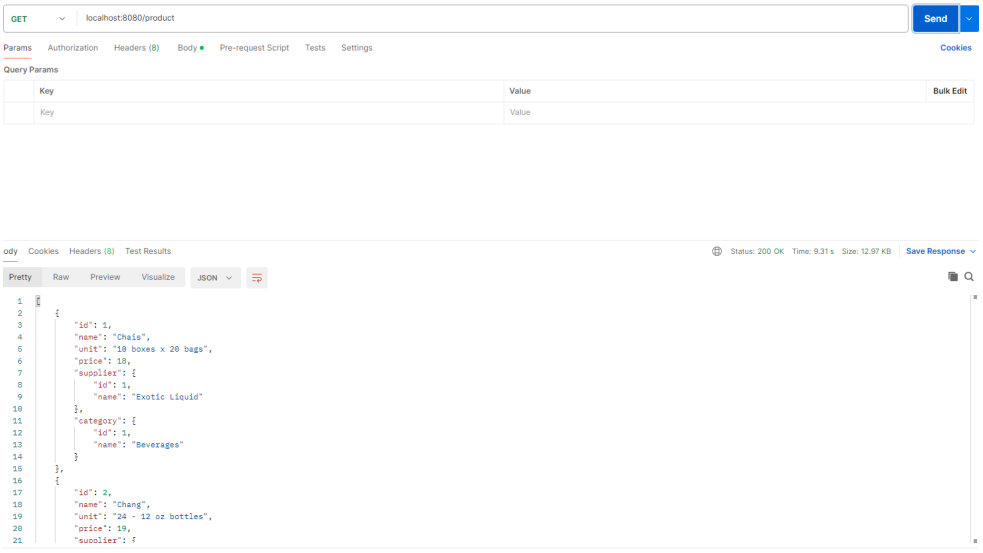
En esta documentación se detalla los endpoints RESTful para el servicio de gestión de productos

1. LISTAR PRODUCTOS

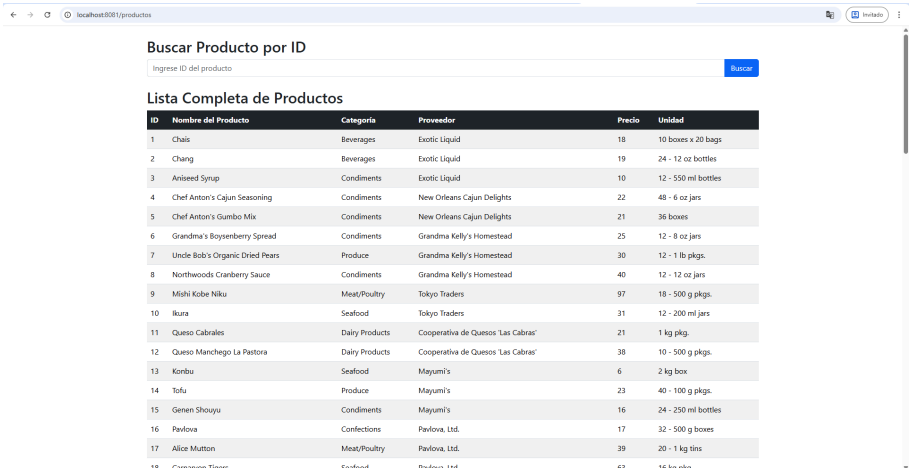
Devuelve una lista de todos los productos que se encuentran en la base de datos.

- **Método HTTP**: GET
- **URL:** http://localhost:8080/product
- **Argumentos de Entrada:** Ninguno.
- **Argumentos de Salida (Respuesta Exitosa):**
  - **Código HTTP:** 200 (OK)
  - **Ejemplos**:

Ejemplo 1: Extraído desde Postman



Ejemplo 2: Extraído desde un cliente web



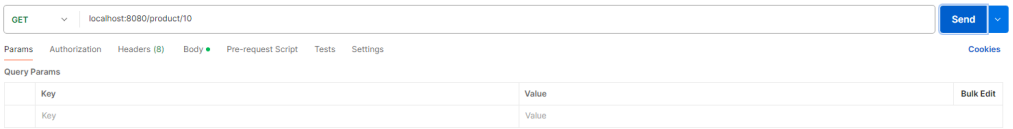
ID	Nombre del Producto	Categoría	Proveedor	Precio	Unidad
1	Chais	Beverages	Exotic Liquid	18	10 boxes x 20 bags
2	Chang	Beverages	Exotic Liquid	19	24 - 12 oz bottles
3	Aniseed Syrup	Condiments	Exotic Liquid	10	12 - 550 ml bottles
4	Chef Anton's Cajun Seasoning	Condiments	New Orleans Cajun Delights	22	48 - 6 oz jars
5	Chef Anton's Gumbo Mix	Condiments	New Orleans Cajun Delights	21	36 boxes
6	Grandma's Boysenberry Spread	Condiments	Grandma Kelly's Homestead	25	12 - 8 oz jars
7	Uncle Bob's Organic Dried Pears	Produce	Grandma Kelly's Homestead	30	12 - 1 lb pkgs.
8	Northwoods Cranberry Sauce	Condiments	Grandma Kelly's Homestead	40	12 - 12 oz jars
9	Mishi Kobe Niku	Meat/Poultry	Tokyo Traders	97	18 - 500 g pkgs.
10	Ikura	Seafood	Tokyo Traders	31	12 - 200 ml jars
11	Queso Cabrales	Dairy Products	Cooperativa de Quesos 'Las Cabras'	21	1 kg pkg.
12	Queso Manchego La Pastora	Dairy Products	Cooperativa de Quesos 'Las Cabras'	38	10 - 500 g pkgs.
13	Konbu	Seafood	Mayumi's	6	2 kg box
14	Tofu	Produce	Mayumi's	23	40 - 100 g pkgs.
15	Genen Shouyu	Condiments	Mayumi's	16	24 - 250 ml bottles
16	Pavlova	Confections	Pavlova, Ltd.	17	32 - 500 g boxes
17	Alice Mutton	Meat/Poultry	Pavlova, Ltd.	39	20 - 1 kg tins

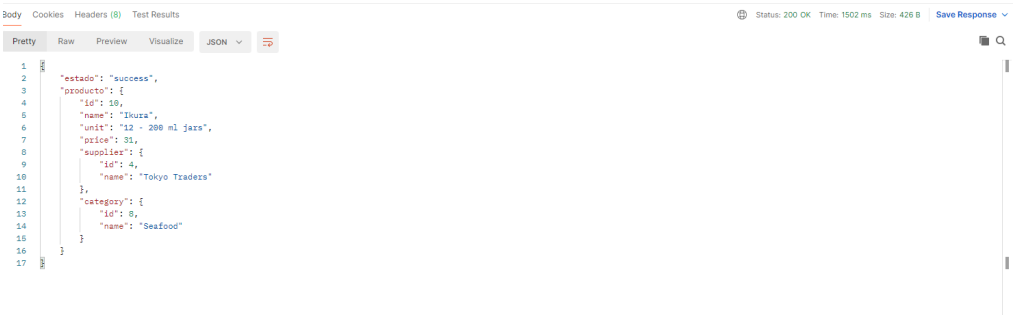
2. OBTENER UN PRODUCTO POR ID

Busca y devuelve un producto específico por su ID.

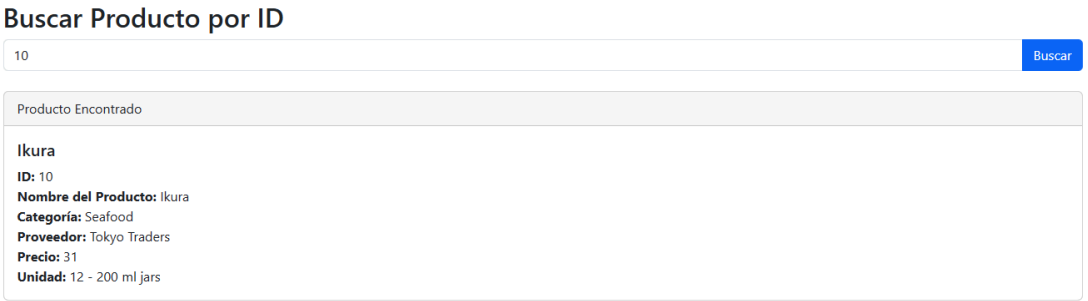
- **Método HTTP**: GET
- **URL:** http://localhost:8080/product/{id}
- **Argumentos de Entrada:**
    - **id (parámetro de ruta):** El ID del producto a buscar.
- **Argumentos de Salida (Respuesta Exitosa):**
    - **Código HTTP:** 200 (OK)
    - **Ejemplos**

Ejemplo 1: Extraído desde Postman



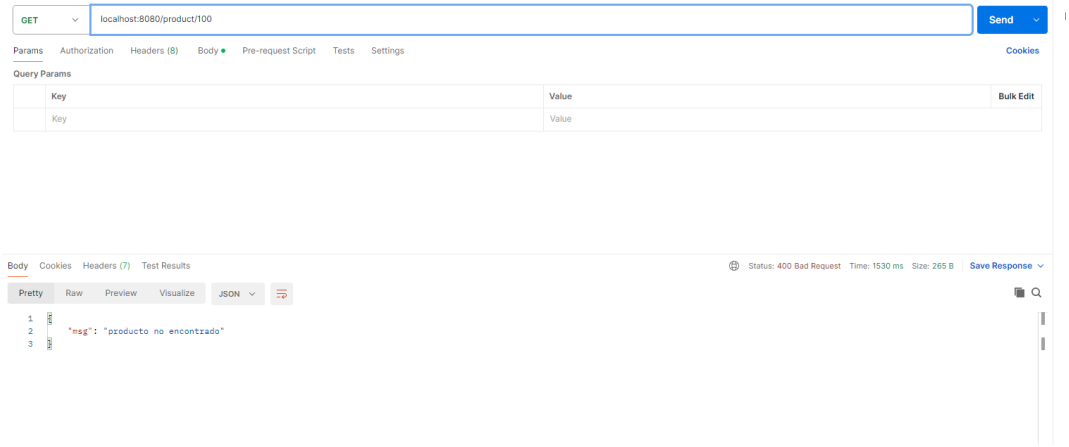


Ejemplo 2: Extraído desde un cliente web

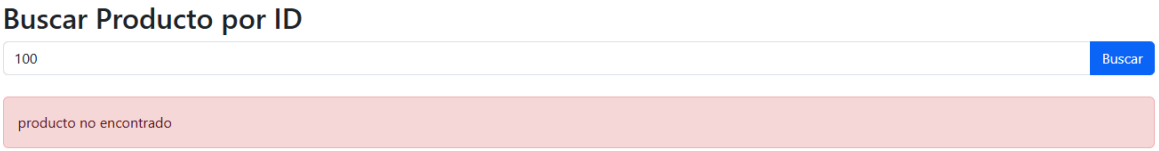


- **Argumentos de Salida (Respuesta con error):**
    - **Código HTTP:** 400 (Bad request)
    - **Descripción:** Cuando el producto no existe
    - **Ejemplos:**

Ejemplo 1: Extraído desde Postman

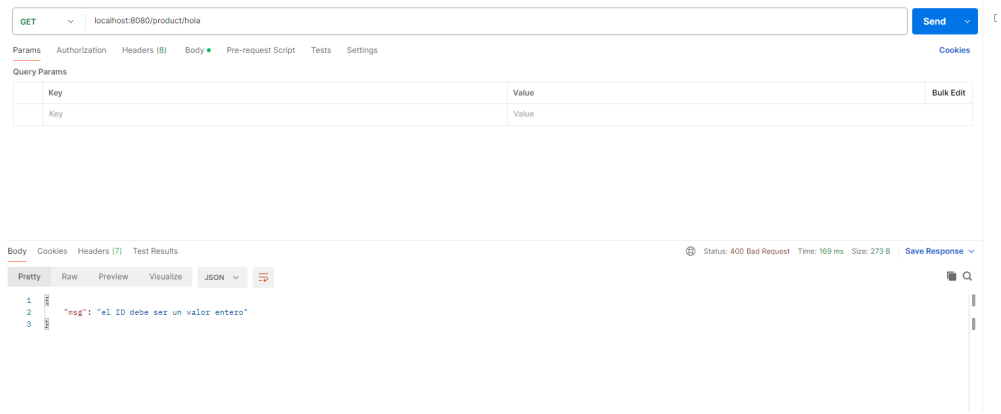


Ejemplo 2: Extraído desde un cliente web



- **Argumentos de Salida (Respuesta con error):**
  - **Código HTTP:** 400 (Bad request)
  - **Descripción:** Cuando el ID no es un valor entero
  - **Ejemplos:**

Ejemplo 1: Extraído desde Postman



Ejemplo 2: Extraído desde un cliente web

Buscar Producto por ID

hola

Buscar

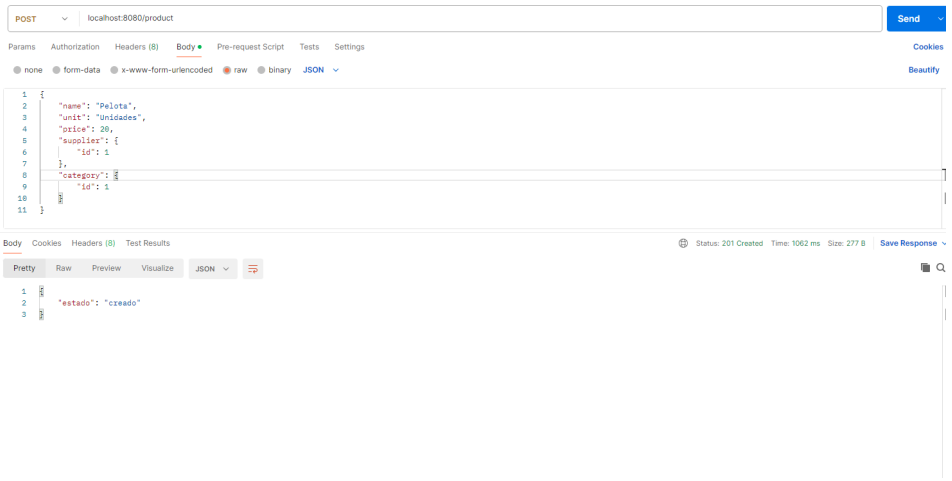
El ID debe ser un número entero

3. CREAR UN PRODUCTO

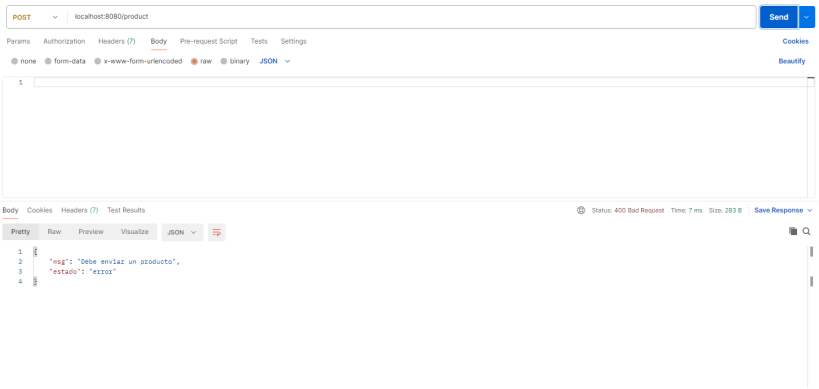
Guarda un nuevo producto en la base de datos.

- **Método HTTP**: POST
- **URL:** http://localhost:8080/product
- **Argumentos de Entrada:**
    - En formato JSON:

- **Argumentos de Salida (Respuesta Exitosa):**
    - **Código HTTP:** 201 (Created)
    - **Ejemplo: (extraído desde Postman)**



- **Argumentos de Salida (Respuesta con error):**
    - **Código HTTP:** 400 (Bad request)
    - **Descripción**: No se ha enviado ningún parámetro para crear el producto.
    - **Ejemplo: (extraído desde Postman)**

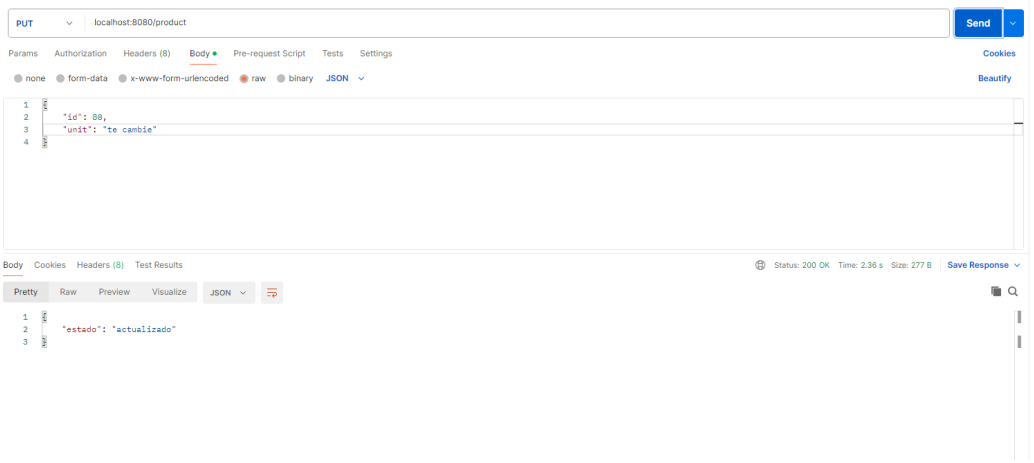

POST
localhost:8080/product
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
1
Body
Cookies
Headers (7)
Test Results
Status: 400 Bad Request
Time: 7 ms
Size: 283 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "msg": "Debe enviar un producto",
3 "estado": "error"
4 }

4. ACTUALIZAR PRODUCTO

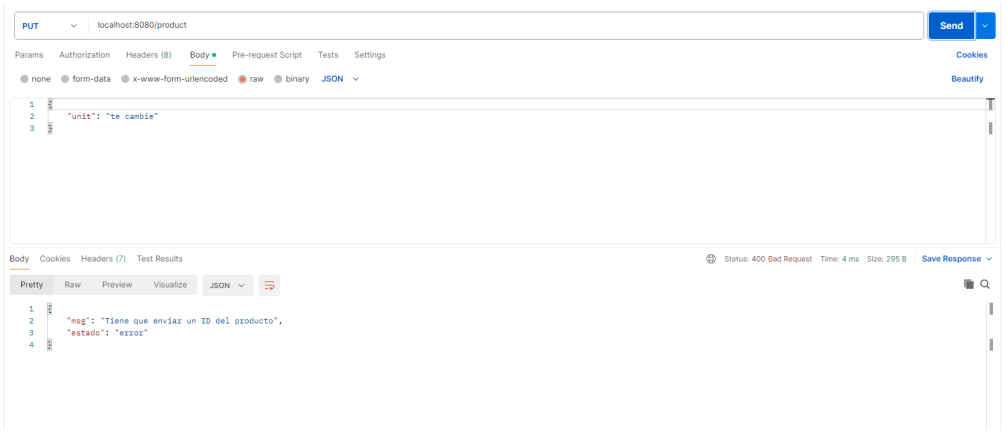
Actualiza un producto existente. Esta implementación realiza una actualización parcial, pues solo actualiza los campos donde no son nulos.

- **Método HTTP**: PUT
- **URL:** http://localhost:8080/product
- **Argumentos de Entrada:**
  - En formato JSON: Es necesario colocar el id del producto a actualizar.

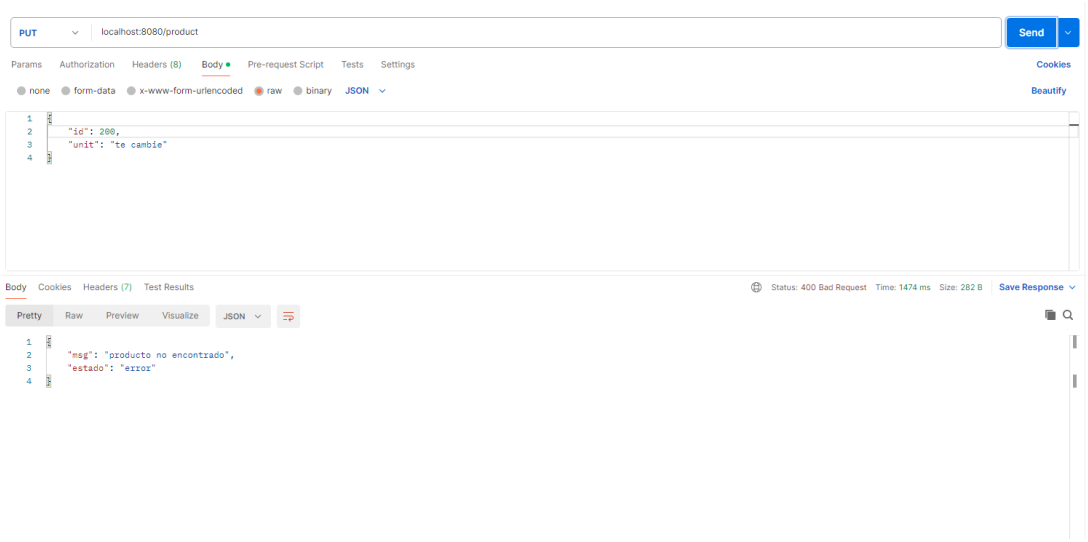
- **Argumentos de Salida (Respuesta Exitosa):**
  - **Código HTTP:** 200 (OK)
  - **Ejemplo: (extraído desde Postman)**



- **Argumentos de Salida (Respuesta con error):**
  - **Código HTTP:** 400 (Bad request)
  - **Descripción**: No se ha enviado ningún ID del producto a actualizar.
  - **Ejemplo: (extraído desde Postman)**



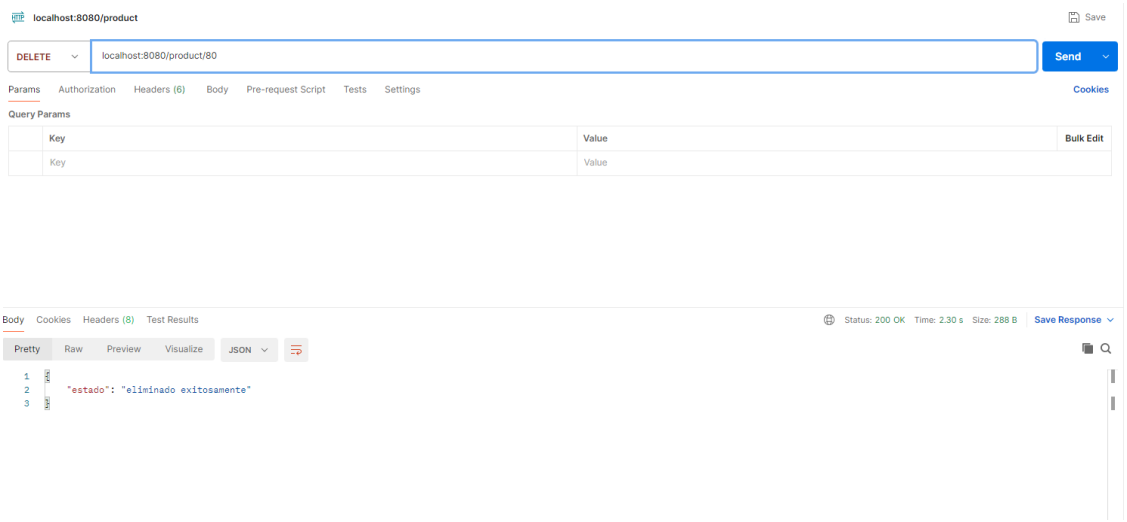
- **Argumentos de Salida (Respuesta con error):**
    - **Código HTTP:** 400 (Bad request)
    - **Descripción**: No existe el ID del producto a actualizar.
    - **Ejemplo: (extraído desde Postman)**



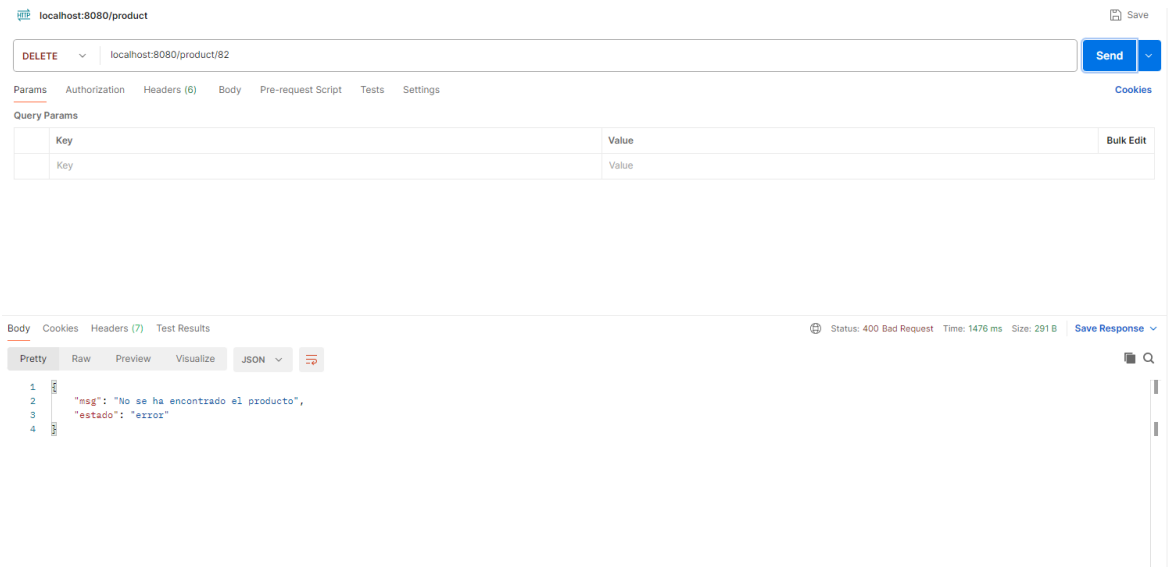
5. BORRAR UN PRODUCTO

Elimina un producto en base a su ID.

- **Método HTTP**: DELETE
- **URL:** http://localhost:8080/product/{id}
- **Argumentos de Entrada:**
  - **id (parámetro de ruta):** El ID del producto a eliminar.
- **Argumentos de Salida (Respuesta Exitosa):**
  - **Código HTTP:** 200 (OK)
  - **Ejemplo: (extraído desde Postman)**



- **Argumentos de Salida (Respuesta con error):**
  - **Código HTTP:** 400 (Bad request)
  - **Descripción:** Cuando el producto no existe
  - **Ejemplo: (extraído desde Postman)**



- **Argumentos de Salida (Respuesta con error):**
    - **Código HTTP:** 400 (Bad request)
    - **Descripción:** Cuando el ID del producto no es un valor entero.
    - **Ejemplo: (extraído desde Postman)**