Servicios:

**Obtener todas las categorías (/categories/all)**

* **Método**: POST
* **URL**: /categories/all
* No es necesario enviar un JSON para este caso, ya que no se reciben parámetros en el cuerpo de la petición.
* **Error al recuperar categorías**:
  + **Código**: 500
  + **Cuerpo de respuesta**:

{

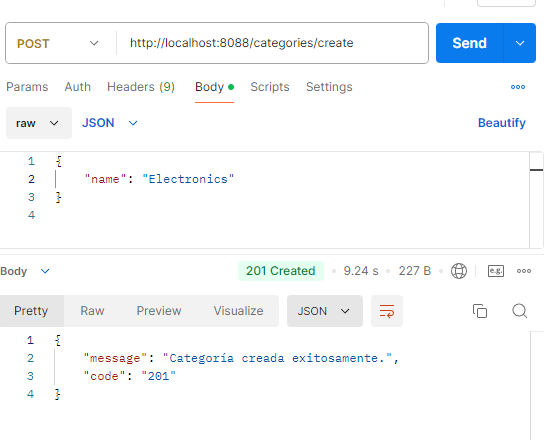
"message": "Error al recuperar categorías: [Mensaje de error]",

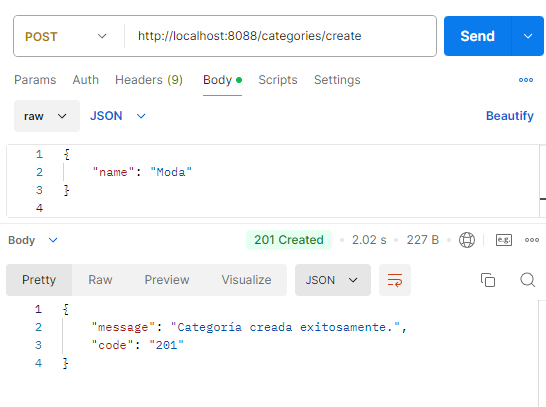
"code": "500"

}

**2. Crear una categoría (/categories/create)**

* **Método**: POST
* **URL**: /categories/create
* **Categoría válida**:
  + **Código**: 201





* **Campo name nulo o vacío**:
  + **Código**: 400
  + **Cuerpo de respuesta**:

{

"message": "El campo 'name' no puede ser nulo o vacío.",

"code": "400"

}

* **Error al crear la categoría**:
  + **Código**: 500
  + **Cuerpo de respuesta**:

{

"message": "Error al crear la categoría",

"code": "500"

}

**3. Actualizar una categoría (/categories/update)**

* **Método**: POST
* **URL**: /categories/update
* **Categoría actualizada exitosamente**:
  + **Código**: 201
  + **Cuerpo de respuesta**:

{

"message": "Categoría actualizada exitosamente.",

"code": "201"

}

* **Campo name nulo o vacío**:
  + **Código**: 400

{

"name": null

}

{

"name": ""

}

* + **Cuerpo de respuesta**:

"message": "El campo 'name' no puede ser nulo o vacío.",

"code": "400"

}

* **Error al actualizar la categoría**:
  + **Código**: 500
  + **Cuerpo de respuesta**:

{

"message": "Error al actualizar la categoría",

"code": "500"

}

**4. Buscar una categoría por ID (/categories/find)**

* **Método**: POST
* **URL**: /categories/find

{

"id": 1

}

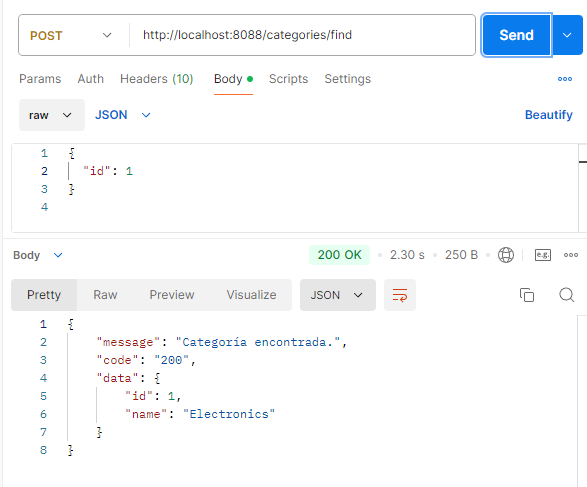
* **Categoría encontrada**:
  + **Código**: 200
  + **Cuerpo de respuesta**:

{

"id": 1,

"name": "Electronics"

}



* **Categoría no encontrada**:

{

"id": null

}

{

"id": -1

}

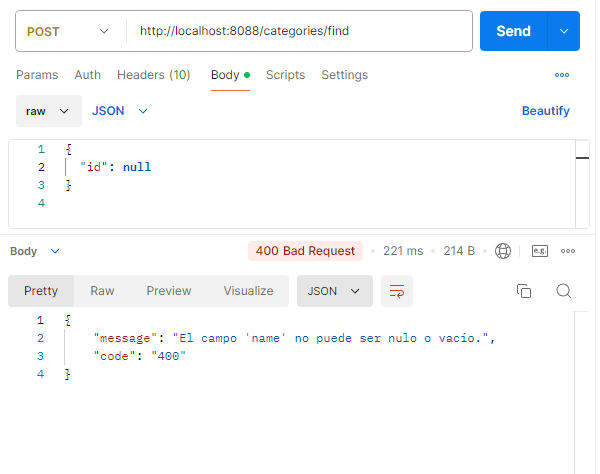
* + **Código**: 404
  + **Cuerpo de respuesta**:

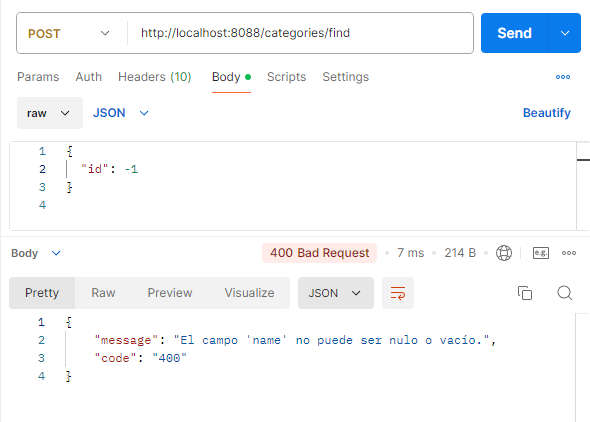
{

"message": "Categoría no encontrada con el ID: 123",

"code": "404"

}





**Obtener todos los productos (/products/all)**

**Método**: POST

No es necesario enviar un JSON para este caso, ya que no se reciben parámetros en el cuerpo de la petición.

**Crear un producto (/products/create/json)**

Aquí se debe enviar un JSON con los detalles completos del producto.

* **Método**: POST
* **URL**: /products/create/json

**JSON Correcto (Producto válido)**:

{

"product": {

"name": "Laptop",

"price": 1200.0,

"category": {

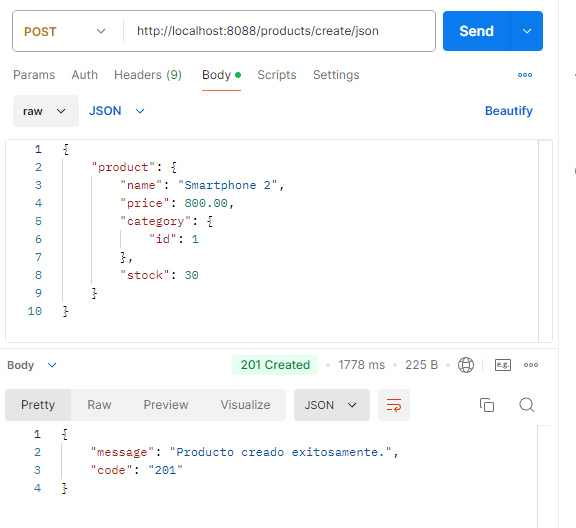
"id": 1

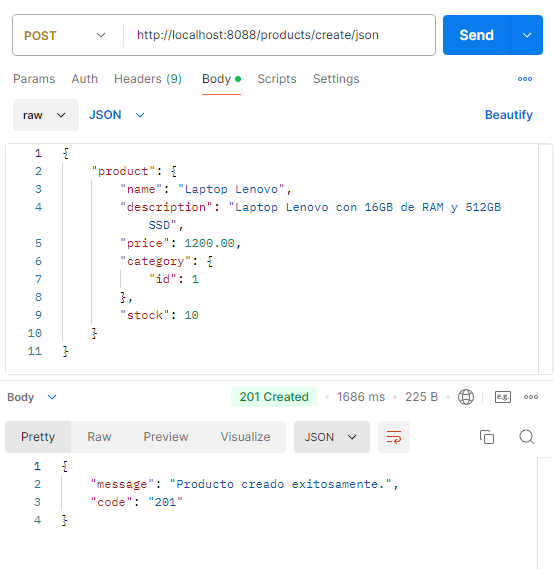
},

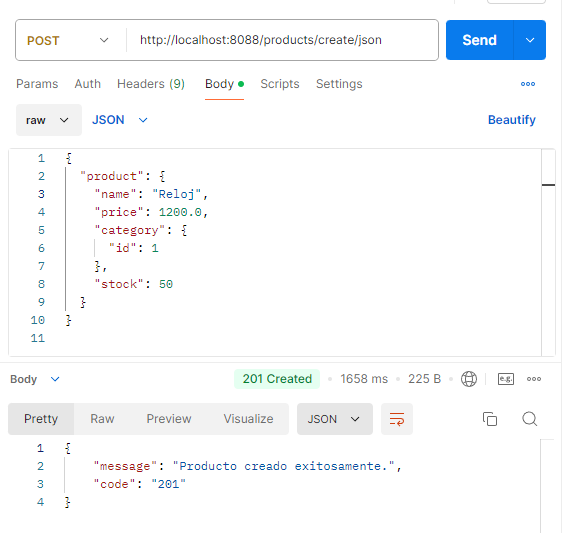
"stock": 50

}

}







**JSON Incorrecto (Producto con errores de validación)**:

1. Producto con nombre vacío:

{

"product": {

"name": "",

"price": 1200.0,

"category": {

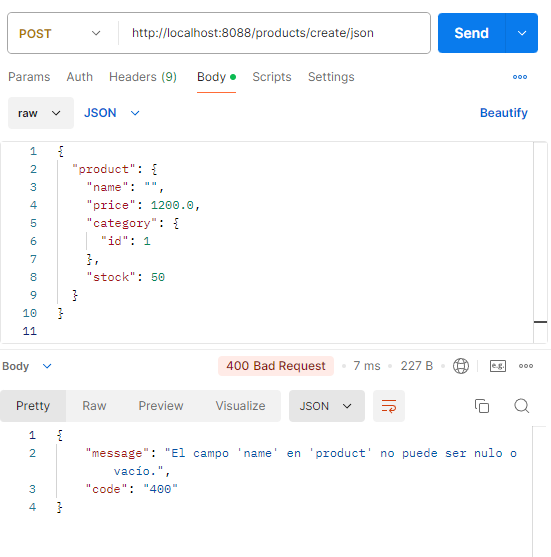
"id": 1

},

"stock": 50

}

}



1. Producto con precio incorrecto:

{

"product": {

"name": "Laptop",

"price": -10.0,

"category": {

"id": 1

},

"stock": 50

}

}



1. Producto sin categoría:

{

"product": {

"name": "Laptop",

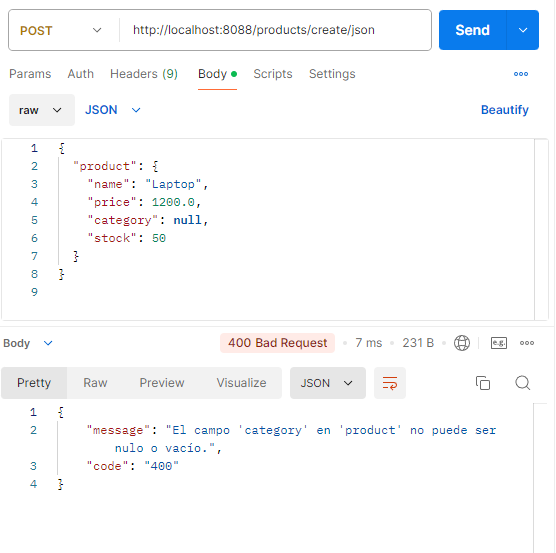
"price": 1200.0,

"category": null,

"stock": 50

}

}



#### Posibles escenarios:

* **Producto válido**: El producto será creado con código 201.
* **Campo product nulo**: Respuesta con código 400 (El campo 'product' no puede ser nulo).
* **Nombre vacío**: Respuesta con código 400 (El campo 'name' en 'product' no puede ser nulo o vacío).
* **Precio negativo o cero**: Respuesta con código 400 (El campo 'price' en 'product' debe ser mayor que 0).
* **Categoría nula**: Respuesta con código 400 (El campo 'category' en 'product' no puede ser nulo o vacío).
* **Stock negativo**: Respuesta con código 400 (El campo 'stock' en 'product' no puede ser negativo).

**Obtener un producto por ID (/products/find/json)**

En este caso, debes enviar un JSON con el id del producto.

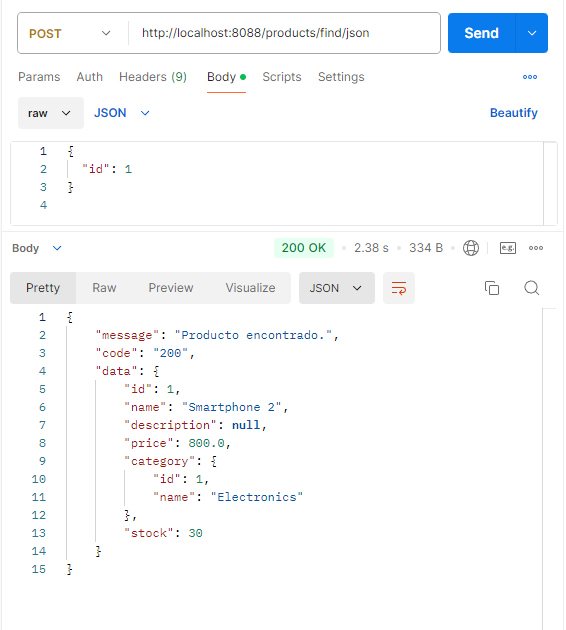
* **Método**: POST
* **URL**: /products/find/json

**JSON Correcto (Producto existente)**:

{

"id": 1

}

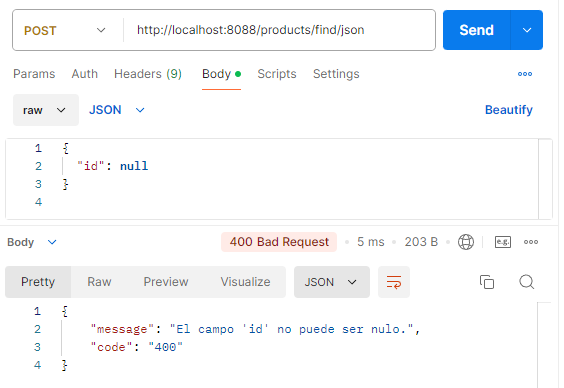


**JSON Incorrecto (ID nulo)**:

{

"id": null

}

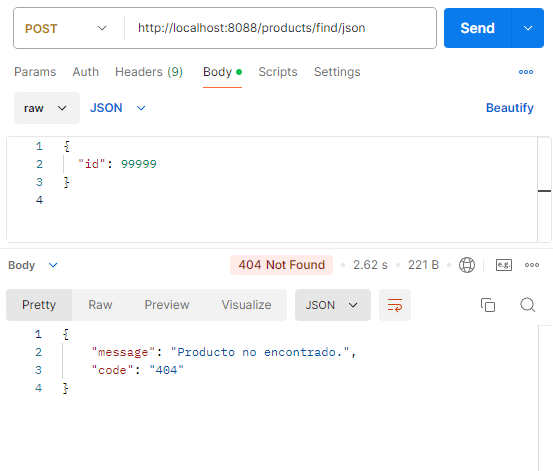


Producto no encontrado:

{

  "id": 99999

}



#### Posibles escenarios:

* **ID válido**: El producto será devuelto con código 200.
* **ID inválido o inexistente**: Respuesta con código 404 (Producto no encontrado).
* **ID nulo**: Respuesta con código 400 (El campo 'id' no puede ser nulo).

**Actualizar un producto (/products/update/json)**

Para actualizar, el JSON incluye el id del producto que se va a actualizar y solo los campos que se deseen modificar.

* **Método**: POST
* **URL**: /products/update/json

**JSON Correcto (Producto existente para actualizar)**:

{

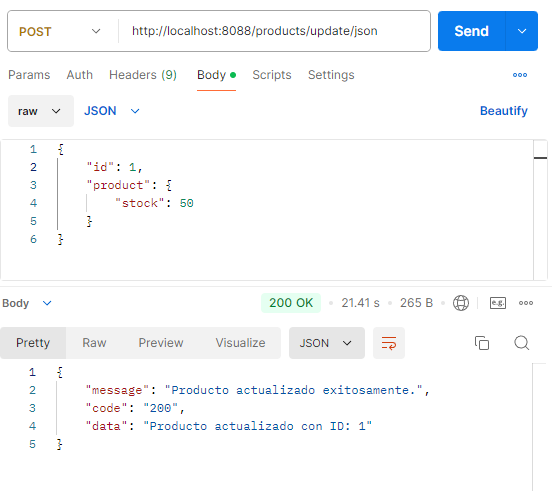
    "id": 6,

    "product": {

        "stock": 50

    }

}



Caso con producto no encontrado

{

"id": 999,

"product": {

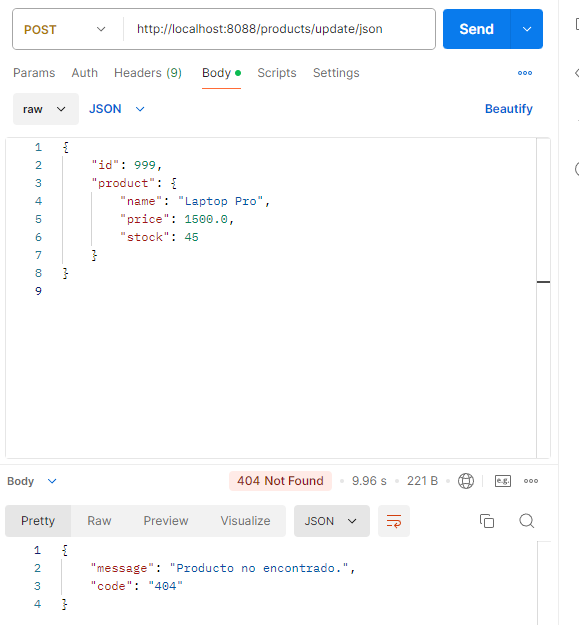
"name": "Laptop Pro",

"price": 1500.0,

"stock": 45

}

}



#### Caso con ID nulo (validación fallida)

{

    "id": **null**,

    "product": {

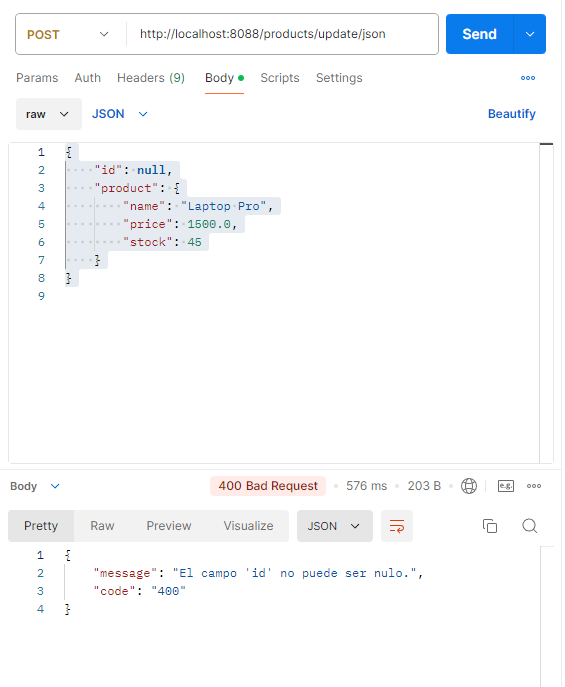
        "name": "Laptop Pro",

        "price": 1500.0,

        "stock": 45

    }

}



#### Caso con nombre vacío en la actualización (validación fallida)

* **Cuerpo de entrada**:

{

"id": 1,

"product": {

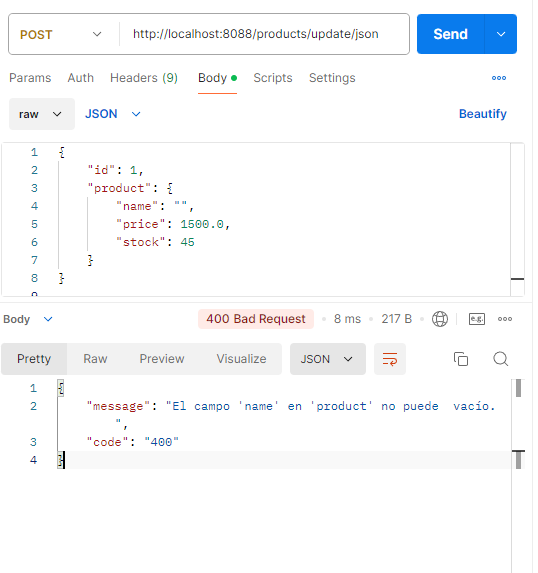
"name": "",

"price": 1500.0,

"stock": 45

}

}



#### e. Caso con precio negativo en la actualización (validación fallida)

* **Cuerpo de entrada**:

{

"id": 1,

"product": {

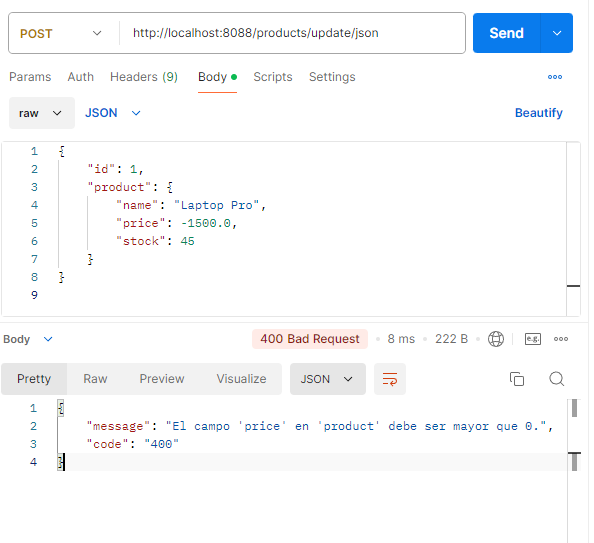
"name": "Laptop Pro",

"price": -1500.0,

"stock": 45

}

}



#### Caso con categoría sin ID (validación fallida)

* **Cuerpo de entrada**:

{

"id": 1,

"product": {

"name": "Laptop Pro",

"price": 1500.0,

"category": {

"id": null

},

"stock": 45

}

}

#### Posibles escenarios:

* **Actualización exitosa**: El producto será actualizado con código 200.
* **Producto no encontrado**: Respuesta con código 404 (Producto no encontrado).
* **ID nulo**: Respuesta con código 400 (El campo 'id' no puede ser nulo).
* **Validación de campos fallida**: Respuesta con código 400 con un mensaje de error detallado.

**Eliminar un producto (/products/delete/json)**

Debes enviar el id del producto que quieres eliminar.

* **Método**: POST
* **URL**: /products/delete/json

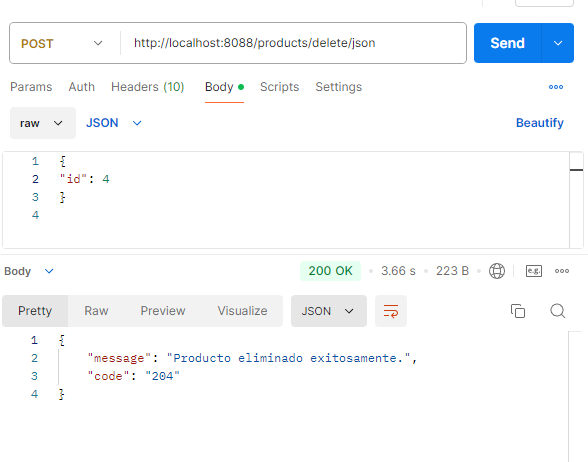
#### Caso exitoso (Producto eliminado)

* **Cuerpo de entrada**:

{

"id": 1

}

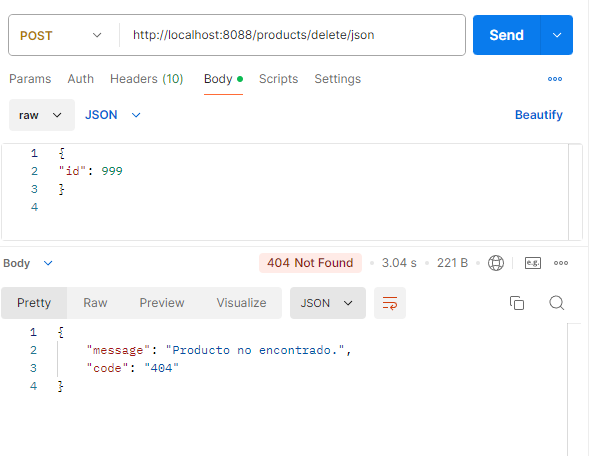


#### Caso con producto no encontrado

* **Cuerpo de entrada**:

{

"id": 999



}

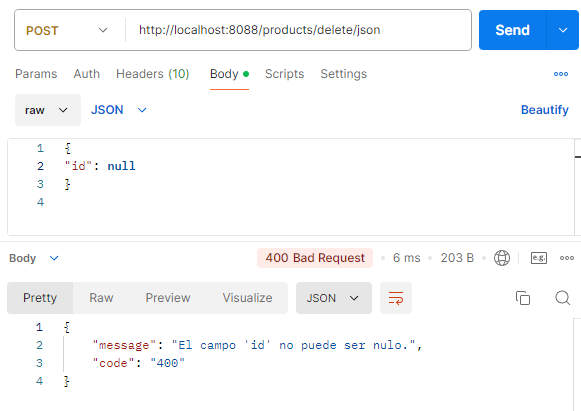
#### Caso con ID nulo (validación fallida)

* **Cuerpo de entrada**:

{

"id": null

}



#### Posibles escenarios:

* **Producto eliminado exitosamente**: Respuesta con código 204.
* **Producto no encontrado**: Respuesta con código 404 (Producto no encontrado).
* **ID nulo**: Respuesta con código 400 (El campo 'id' no puede ser nulo).

Con estos JSON, puedes cubrir todos los casos de uso para cada uno de los

**Obtener todos los productos (/products/all)**

Este endpoint no requiere un cuerpo en la solicitud. Simplemente se hace una solicitud POST vacía:

* **Método**: POST
* **URL**: /products/all
* **Cuerpo**: No aplica.

### . ****Crear un pedido (/orders/create/json)****

* **Método**: POST
* **URL**: /orders/create/json
* **Pedido válido**:

{

"userId": 1,

"orderDate": "2024-09-13T10:00:00",

"status": "PENDING",

"items": [

{

"productId": 1,

"quantity": 2,

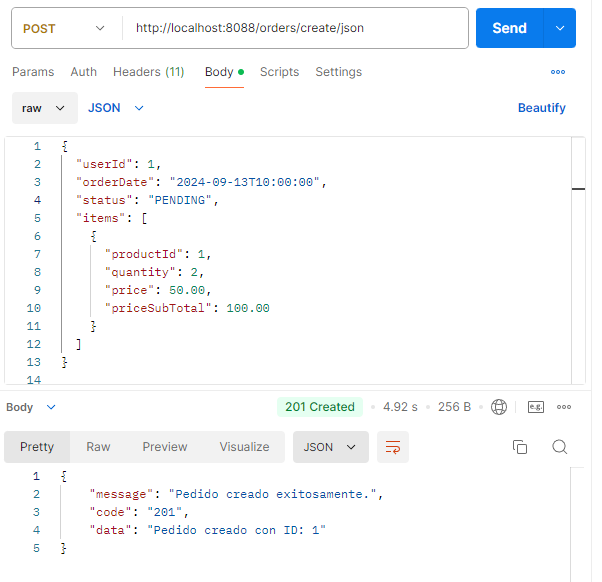
"price": 50.00,

"priceSubTotal": 100.00

}

]

}



* **Campo orderDate nulo**:

{

"userId": 1,

"orderDate": null,

"status": "PENDING",

"total": 100.00,

"items": [

{

"productId": 1,

"quantity": 2,

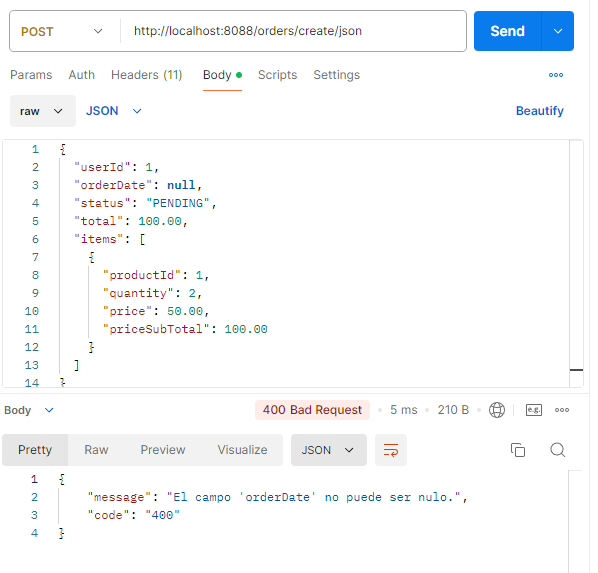
"price": 50.00,

"priceSubTotal": 100.00

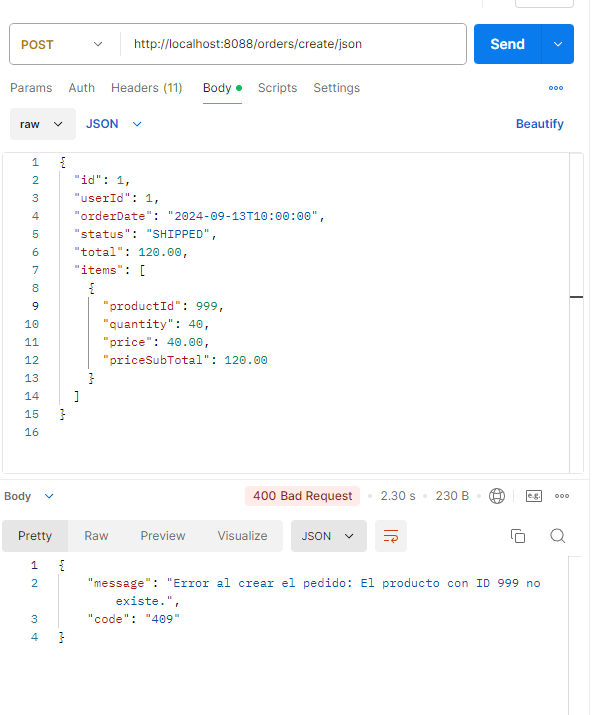
}

]

}



El producto no existe:



* **Campo status nulo o vacío**:

{

"userId": 1,

"orderDate": "2024-09-13T10:00:00",

"status": "",

"total": 100.00,

"items": [

{

"productId": 1,

"quantity": 2,

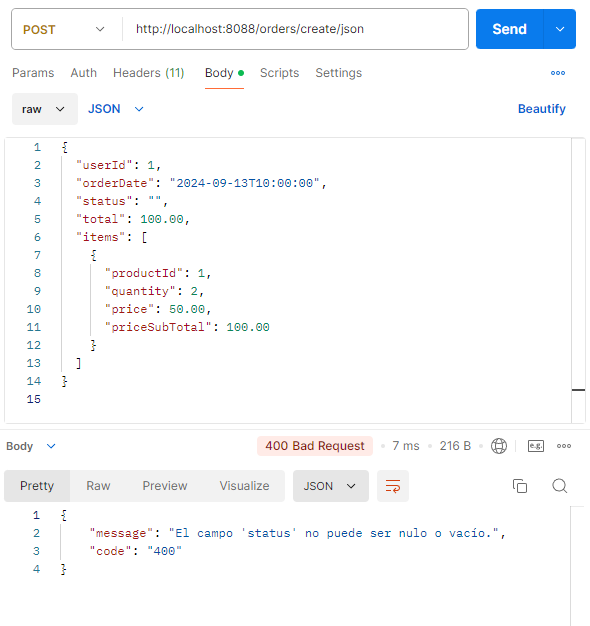
"price": 50.00,

"priceSubTotal": 100.00

}

]

}



* **Campo items nulo o vacío**:

{

"userId": 1,

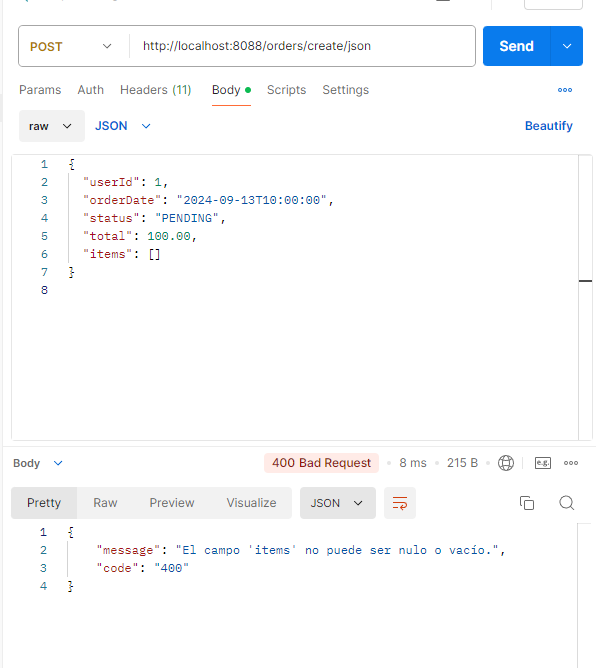
"orderDate": "2024-09-13T10:00:00",

"status": "PENDING",

"total": 100.00,

"items": []

}

****

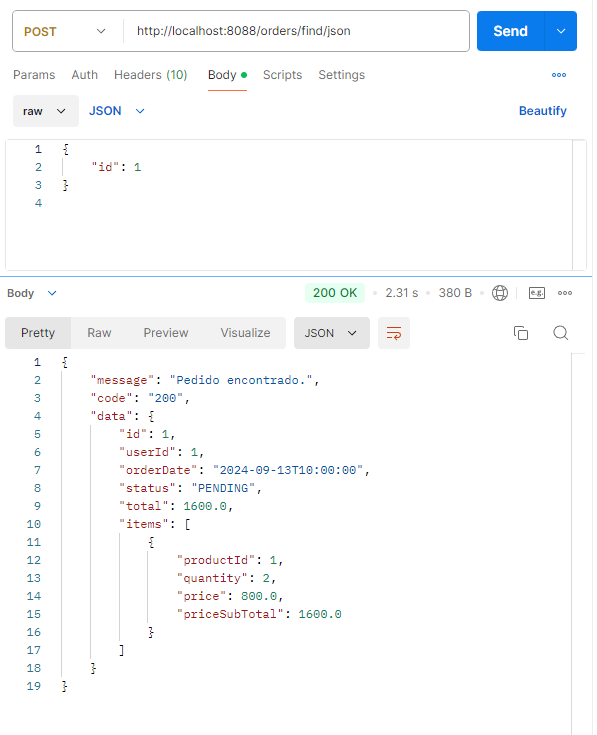
### ****Buscar un pedido por ID (/orders/find/json)****

* **Método**: POST
* **URL**: /orders/find/json
* Pedido válido:

{

"id": 1

}

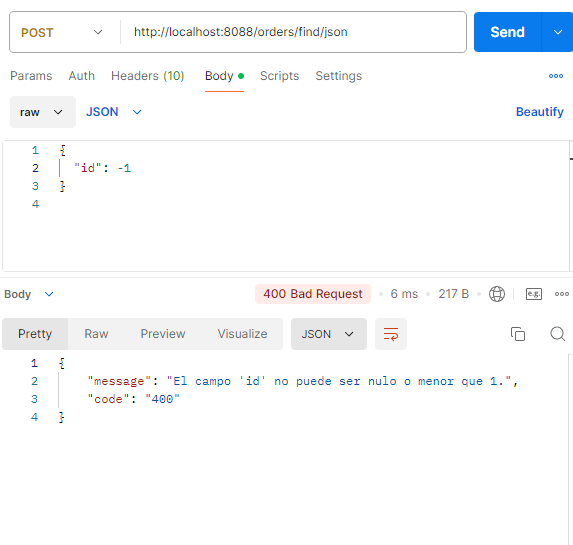


* **Campo id nulo**:

{

"id": null

}



### ****Actualizar un pedido (/orders/update/json)****

**Método para actualizar los parámetros de** orderDate, status y ítems. Los demás no se toman.

El price, priceSubTotal y total se buscan y calculan

* **Método**: POST
* **URL**: /orders/update/json
* **Pedido válido**:

{

"id": 1,

"userId": 1,

"orderDate": "2024-09-13T10:00:00",

"status": "SHIPPED",

"total": 120.00,

"items": [

{

"productId": 1,

"quantity": 3,

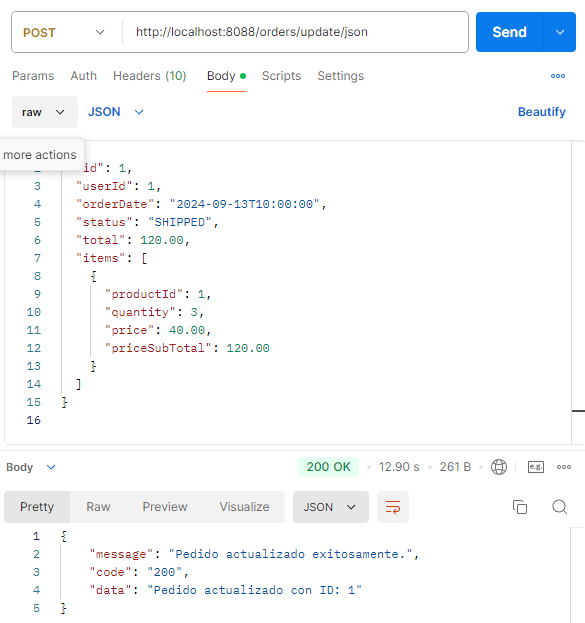
"price": 40.00,

"priceSubTotal": 120.00

}

]

}



* **Campo id nulo**:

{

"id": null,

"userId": 1,

"orderDate": "2024-09-13T10:00:00",

"status": "SHIPPED",

"total": 120.00,

"items": [

{

"productId": 1,

"quantity": 3,

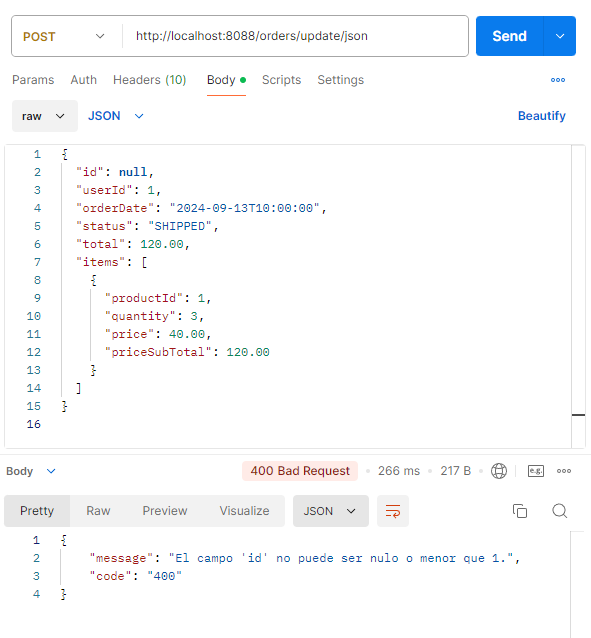
"price": 40.00,

"priceSubTotal": 120.00

}

]

}



* **Campo orderDate nulo**:

{

"id": 1,

"userId": 1,

"orderDate": null,

"status": "SHIPPED",

"total": 120.00,

"items": [

{

"productId": 1,

"quantity": 3,

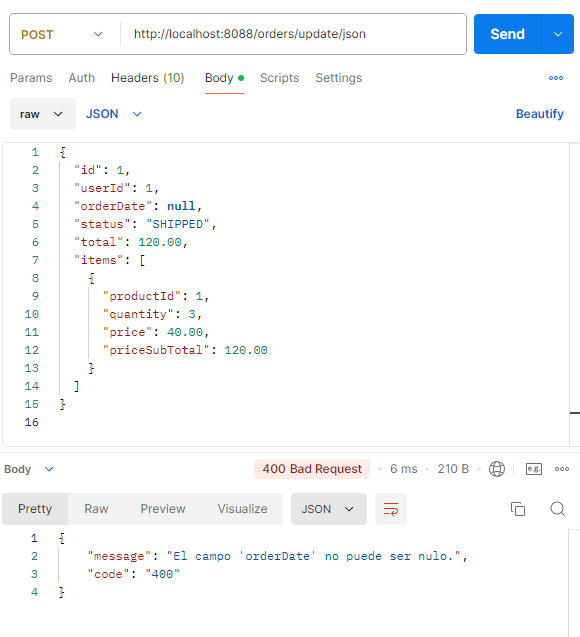
"price": 40.00,

"priceSubTotal": 120.00

}

]

}



* **Campo items nulo o vacío**:

{

"id": 1,

"userId": 1,

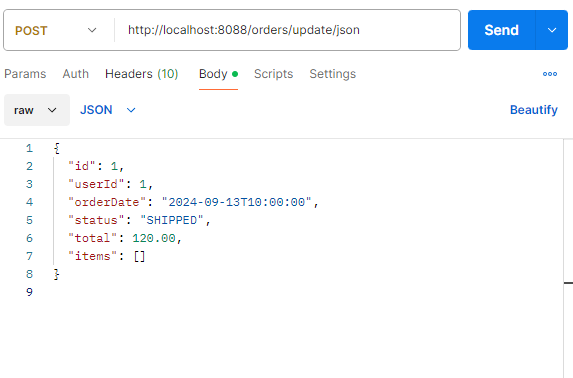
"orderDate": "2024-09-13T10:00:00",

"status": "SHIPPED",

"total": 120.00,

"items": []

}



Productos no existen:



### ****Eliminar un pedido (/orders/delete/json)****

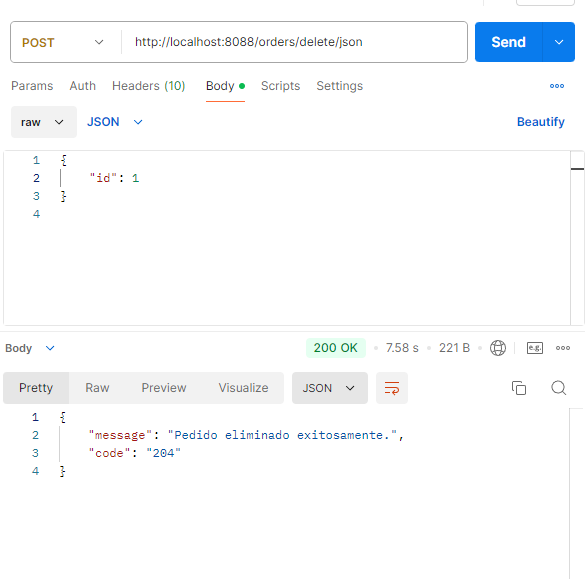
### ****Elimina las ordenes que este canceladas y repone el stock****

* **Método**: POST
* **URL**: /orders/delete/json
* **Pedido válido**:

{

    "id": 1

}

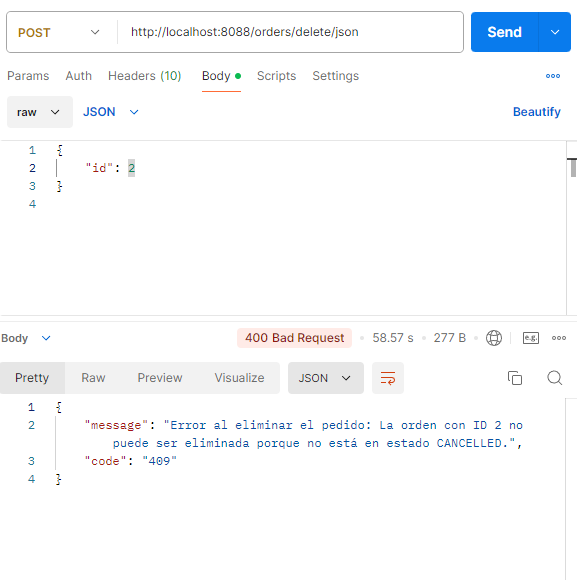


* **Pedido válido pero debe cumplir la condicion**:

{

"id": 2

}



* **Campo id nulo**:

{

"id": null

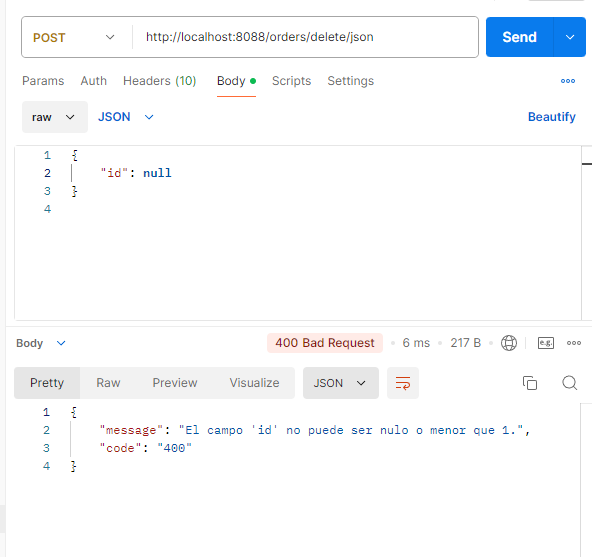
}

* **Campo id no válido** (por ejemplo, un valor negativo):

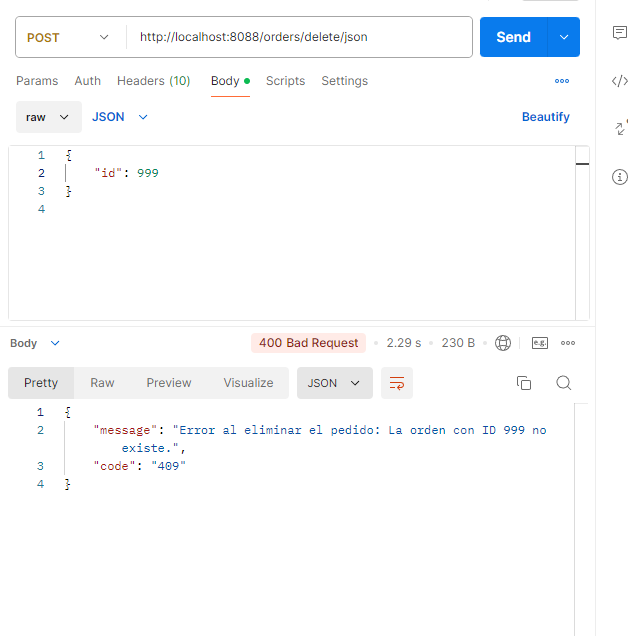
{

"id": -1

}



**Orden no existe**

****

**Obtener todos los pedidos (/orders/all)**

* **Método**: POST
* **URL**: /orders/all
* **Cuerpo de ingreso**: No se requiere cuerpo de solicitud. Se envía una solicitud POST vacía.