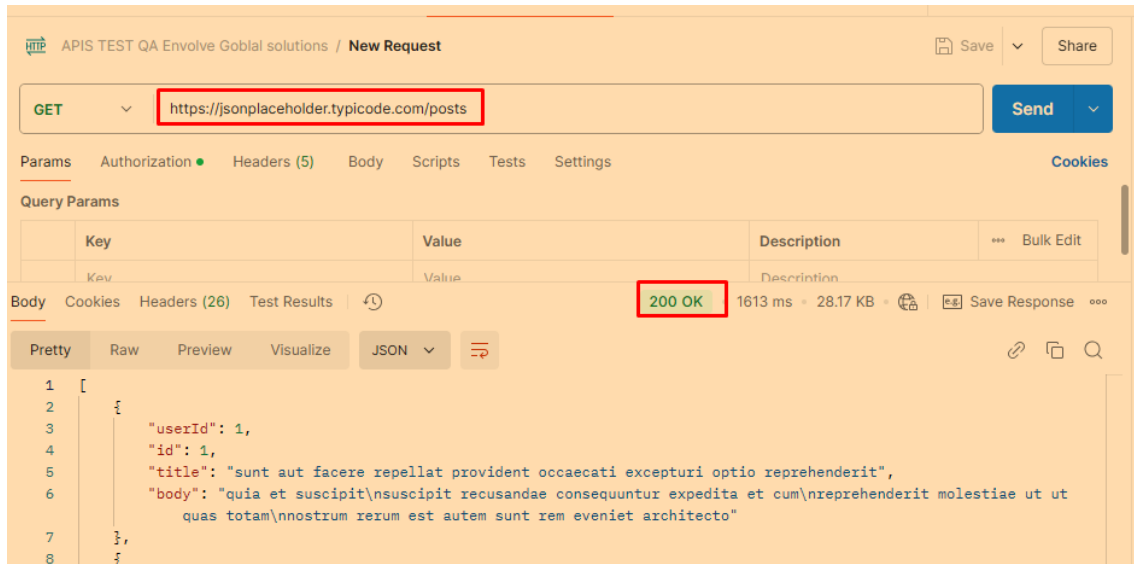


## Método GET

### ✓ Casos positivos

1. Se realiza la petición a la url : <https://jsonplaceholder.typicode.com/posts> dando como resultado exitoso con código de server 200 OK donde se muestran todos los posts.

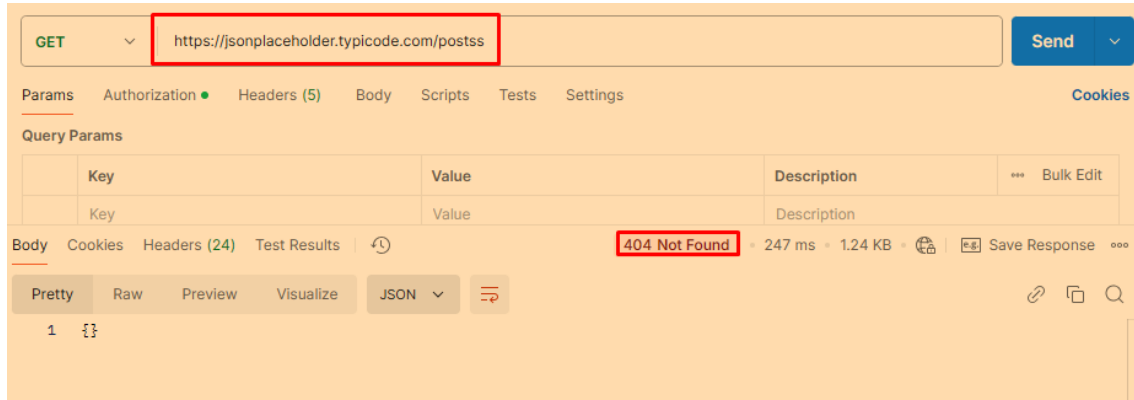


2. Se realiza la petición de un registro a la url : <https://jsonplaceholder.typicode.com/posts/3> dando como resultado exitoso con código de server 200 OK

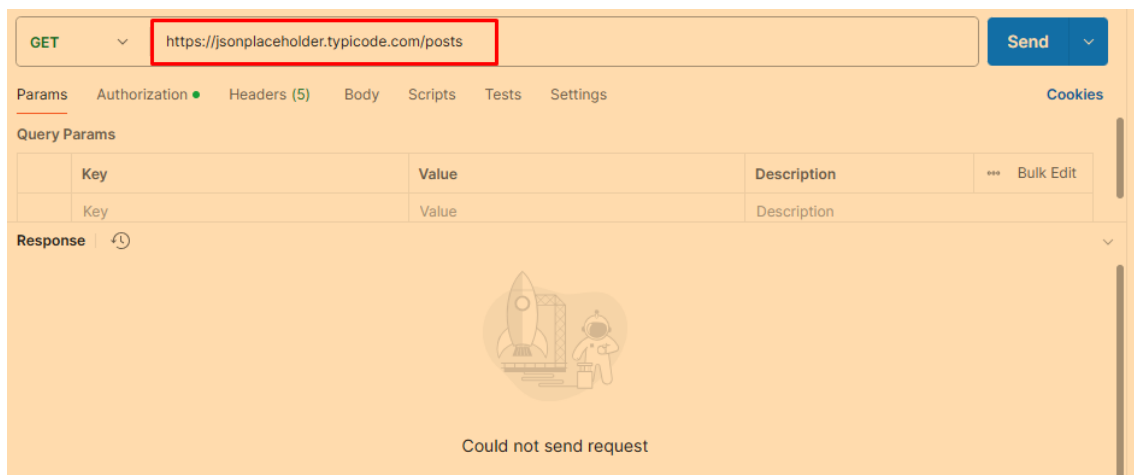


## ✓ Casos Negativos

1. Petición a una url incorrecta <https://jsonplaceholder.typicode.com/postss> que se debe de mostrar código de error 404.



2. Petición a url correcta <https://jsonplaceholder.typicode.com/posts>, pero con server inaccesible.



## Método POST

### ✓ Casos Positivos

1. Se realiza la petición POST desde el body mediante registro JSON, dando como resultado de creación exitoso.

The first screenshot shows a POST request to `https://jsonplaceholder.typicode.com/posts` with a JSON body: `{ "title": "title field expects a string", "body": "body field expects a string", "userId": 1 }`. The response is `201 Created` with a status of 753 ms and 1.47 KB.

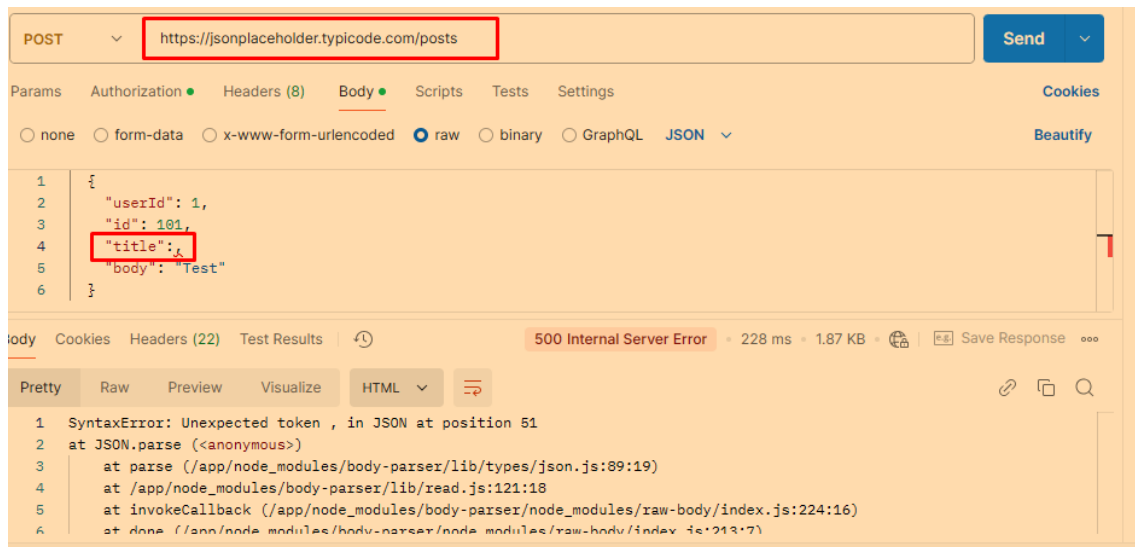
The second screenshot shows a POST request to the same URL with a JSON body: `{ "userId": 1, "id": 101, "title": "Este es el siguiente post de prueba", "body": "Test" }`. The response is `201 Created` with a status of 303 ms and 1.46 KB.

2. Se realiza la petición POST desde el body mediante formulario dando como resultado de creación exitoso.

The screenshot shows a POST request to `https://jsonplaceholder.typicode.com/posts` with a form-data body. The fields are: `userId` (1), `Id` (2), `title` (Test), and `body` (Test2). The response is `201 Created` with a status of 437 ms and 1.39 KB.

## ✓ Casos Negativos

1. Se realiza la petición POST desde el body mediante registro JSON con campos vacíos. NO SE ESPERAN CAMPOS VACIOS



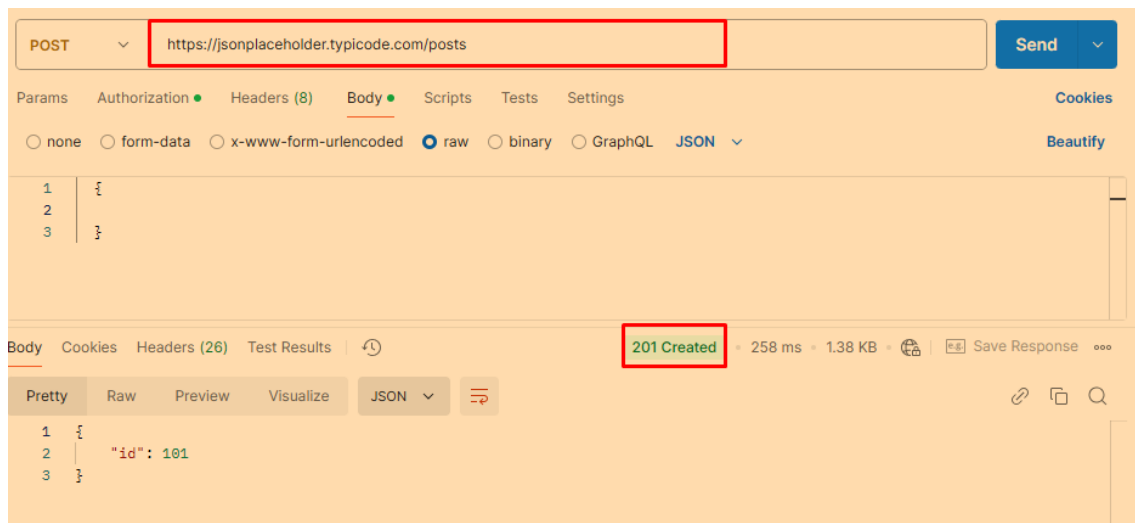
The screenshot shows a REST client interface with a POST request to `https://jsonplaceholder.typicode.com/posts`. The body is set to raw JSON. The JSON body is:

```
{
  "userId": 1,
  "id": 101,
  "title": ,
  "body": "Test"
}
```

The response status is **500 Internal Server Error**. The error message in the response body is:

```
SyntaxError: Unexpected token , in JSON at position 51
at JSON.parse (<anonymous>)
    at parse (/app/node_modules/body-parser/lib/types/json.js:89:19)
    at /app/node_modules/body-parser/lib/read.js:121:18
    at invokeCallback (/app/node_modules/body-parser/node_modules/raw-body/index.js:224:16)
    at done (/app/node_modules/body-parser/node_modules/raw-body/index.js:213:7)
```

2. Se realiza la petición POST desde el body mediante registro JSON vacío completamente, había que ver las validaciones de la API para hacer algún otro campo obligatorio.



The screenshot shows the same REST client interface with a POST request to `https://jsonplaceholder.typicode.com/posts`. The body is set to raw JSON and is empty:

```
{
}
```

The response status is **201 Created**. The response body is:

```
{
  "id": 101
}
```

## Método PUT

### ✓ Casos Positivos

1. Se envía petición PUT desde el body mediante un JSON para actualizar post con id = 1 dando resultado exitoso.

The screenshot shows a REST client interface with the URL `https://jsonplaceholder.typicode.com/posts/1` and the method `PUT`. The request body is a JSON object: `{ "userId": 1, "id": 1, "title": "Updated post title", "body": "Updated content of the post." }`. The response status is `200 OK`, with a response time of 327 ms and a size of 1.38 KB. The response body is also shown in JSON format.

```
1 {
2   "userId": 1,
3   "id": 1,
4   "title": "Updated post title",
5   "body": "Updated content of the post."
6 }
```

2. Se envía petición PUT desde el body mediante formulario para actualizar post con id = 100 dando resultado exitoso.

The screenshot shows a REST client interface with the URL `https://jsonplaceholder.typicode.com/posts/100` and the method `PUT`. The request body is set to `form-data`. The form contains three fields: `userId` (value: 1), `title` (value: Nuevo Titulo), and `body` (value: Nuevo body). The response status is `200 OK`, with a response time of 251 ms and a size of 1.29 KB. The response body is shown in JSON format: `{ "id": 100 }`.

Key	Value	Description
<input checked="" type="checkbox"/> userId	1	
<input checked="" type="checkbox"/> title	Nuevo Titulo	
<input checked="" type="checkbox"/> body	Nuevo body	

```
1 {
2   "id": 100
3 }
```

## ✓ Casos Negativos

1. Se envía petición PUT desde el body mediante un JSON para actualizar post con id = 101 dando resultado erróneo ya que ese registro no existe.

PUT <https://jsonplaceholder.typicode.com/posts/101> Send

Params Authorization Headers (7) **Body** Scripts Tests Settings Cookies Beautify

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☐ JSON

```
1 {
2   "title": "Updated post title",
3   "body": "Updated content of the post.",
4   "userId": 1
5 }
```

Body Cookies Headers (22) Test Results **500 Internal Server Error** 265 ms • 1.99 KB Save Response

Pretty Raw Preview Visualize **HTML**

```
1 TypeError: Cannot read properties of undefined (reading 'id')
2 at update (/app/node_modules/json-server/lib/server/router/plural.js:262:24)
3 at Layer.handle [as handle_request] (/app/node_modules/express/lib/router/layer.js:95:5)
4 at next (/app/node_modules/express/lib/router/route.js:137:13)
5 at next (/app/node_modules/express/lib/router/route.js:131:14)
6 at Route.dispatch (/app/node_modules/express/lib/router/route.js:112:3)
7 at Layer.handle [as handle_request] (/app/node_modules/express/lib/router/layer.js:95:5)
```

2. Se realiza la petición PUT desde el body mediante registro JSON a actualizar vacío completamente, no actualiza nada.

PUT <https://jsonplaceholder.typicode.com/posts/3> Send

Params Authorization Headers (7) **Body** Scripts Tests Settings Cookies Beautify

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☐ JSON

```
1 { }
```

Body Cookies Headers (24) Test Results **200 OK** 261 ms • 1.28 KB Save Response

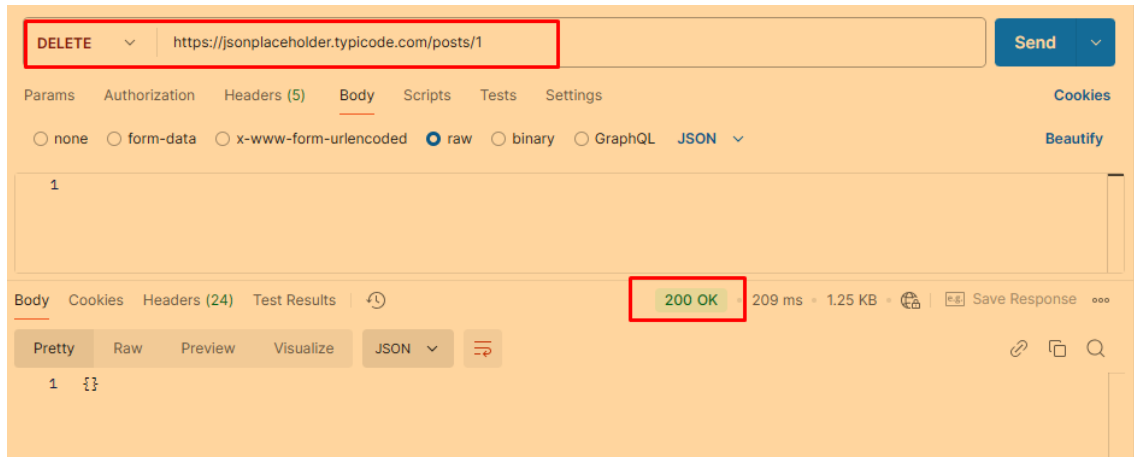
Pretty Raw Preview Visualize **JSON**

```
1 {
2   "id": 3
3 }
```

## Método DELETE

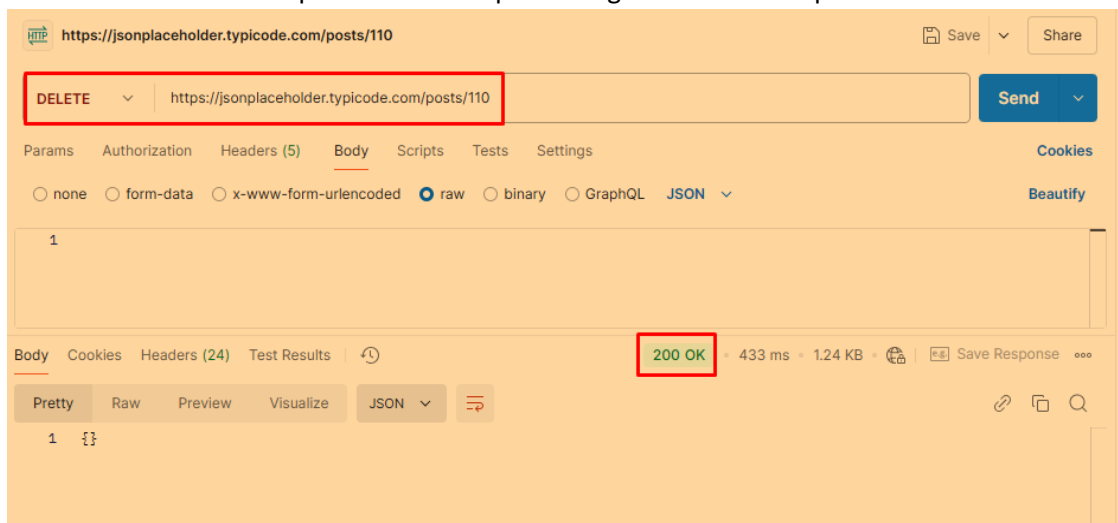
### ✓ Caso Positivo

1. Se realiza la petición DELETE para el registro de id=1 de manera exitosa.



### ✓ Caso Negativo

1. Se realiza la petición DELETE para el registro de id=110 que no existe.



**\*NO debería mostrar el código de éxito 200 ya que el registro no existe.**