UNIVERSIDAD POLITECNICA INTERDISCIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS.

PRACTICA 2

Materia :Aplicaciones distribuidas

Profesor : Noe Sierra Romero

TITULO :

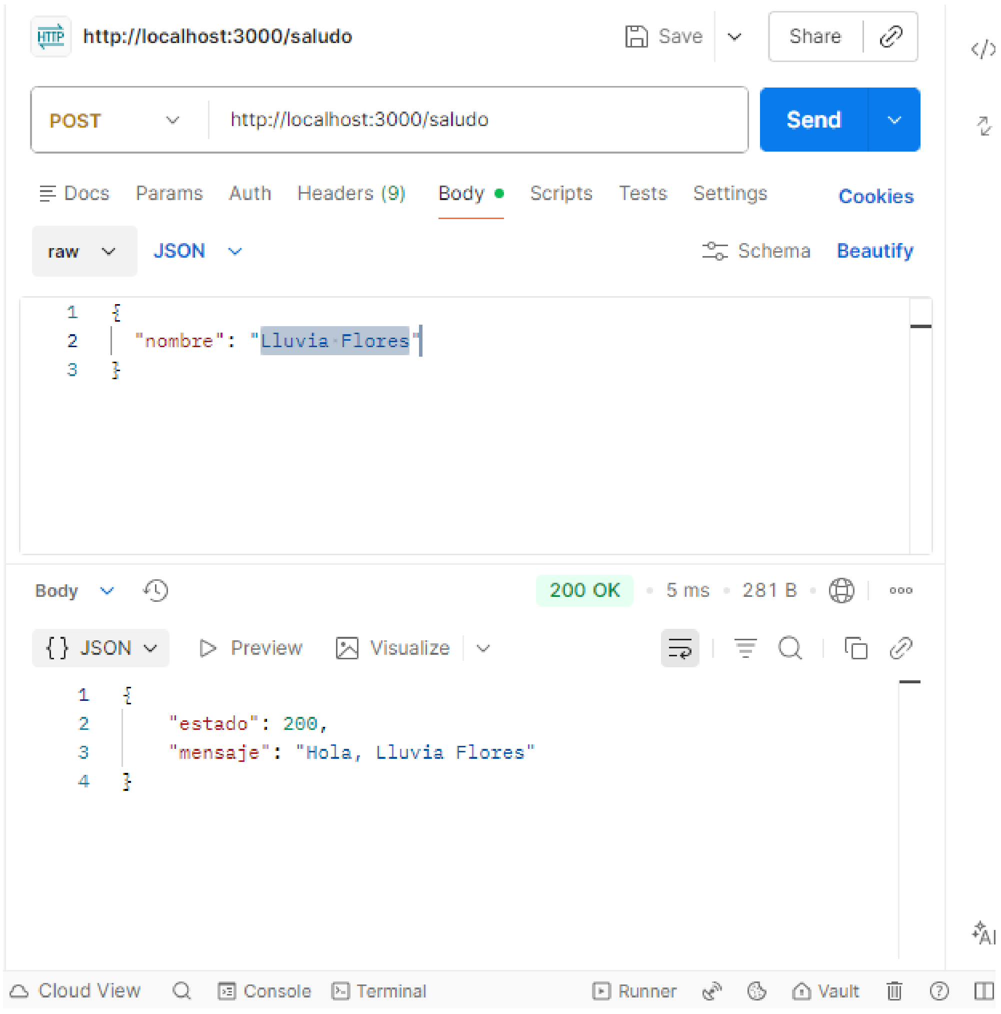
Práctica de repaso para API Restful

FECHA : 21 de Febrero del 2026. Hecho por :

Flores Hernandez Lluvia Nahivy
Boleta : 2022640090

Desarrollo de practica 2

Ejercicio 1 : Servicio de Saludo Básico



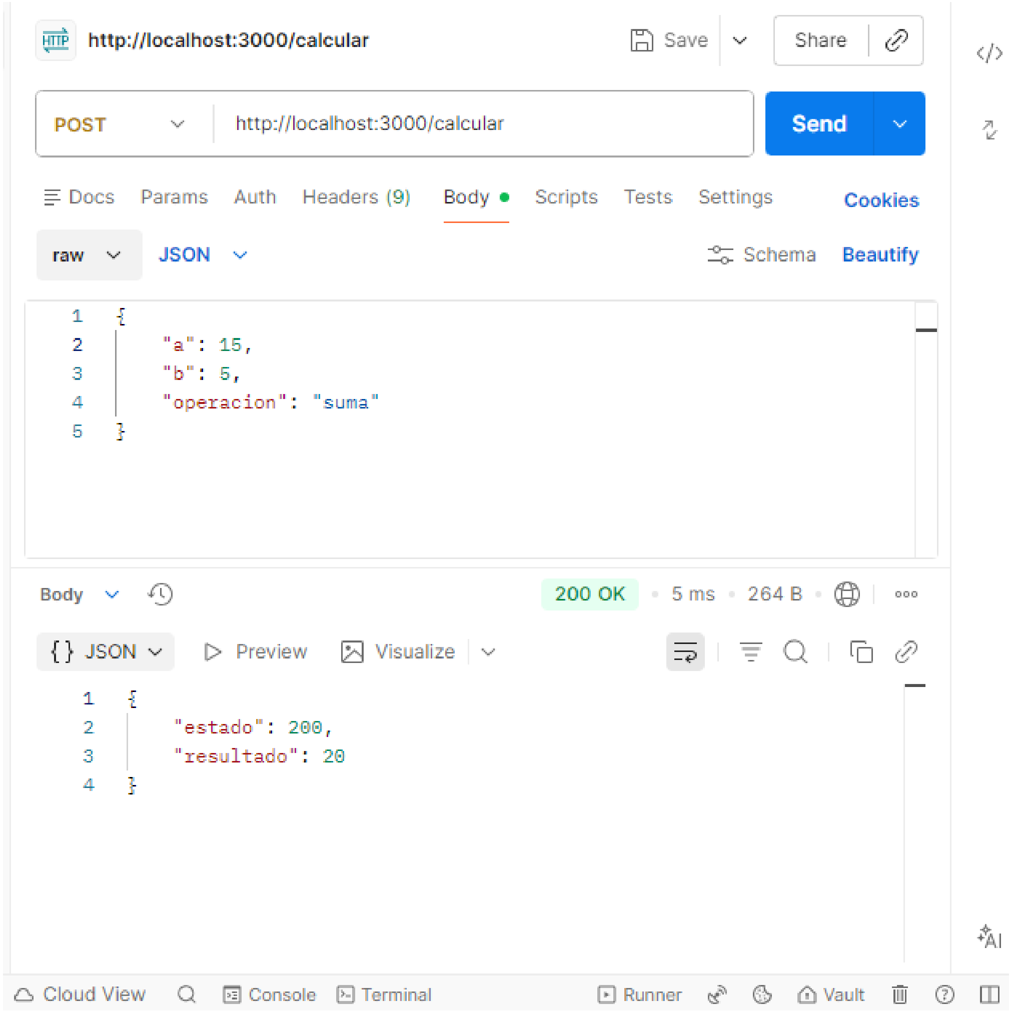
POST http://localhost:3000/saludo

```
{
  "nombre": "Lluvia Flores"
}
```

```
{
    "estado": 200,
    "mensaje": "Hola, Lluvia Flores"
}
```

Recibimos de regreso un saludo con el nombre de la persona, cualquier tipo de nombre que proporcione con una logitud menor o igual a 200 caracteres.

Ejercicio 2 : Calculadora de Operaciones Básicas

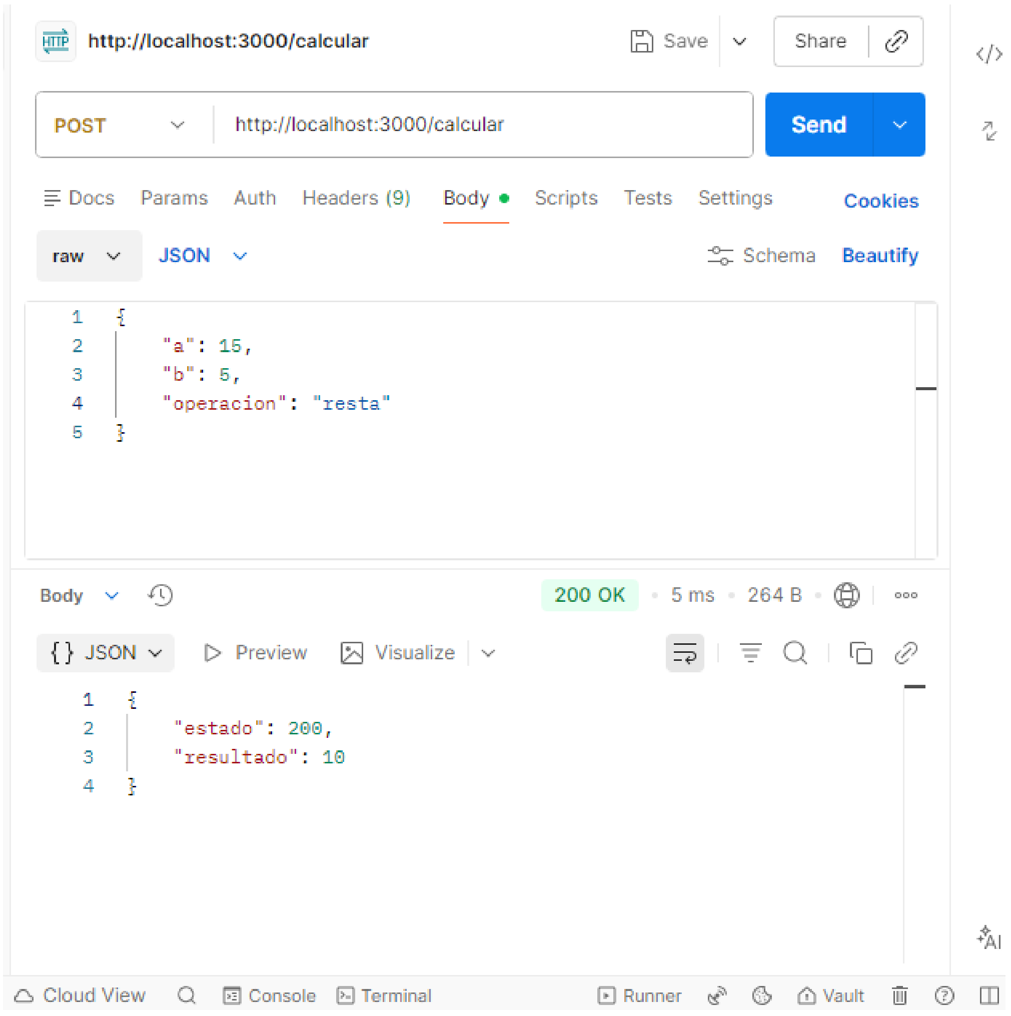

http://localhost:3000/calcular
Save
Share
POST
Send
Docs Params Auth Headers (9) Body Scripts Tests Settings
Cookies
raw JSON
Schema Beautify
Body
200 OK 5 ms 264 B
{ } JSON Preview Visualize
Cloud View Console Terminal Runner Vault

POST http://localhost:3000/calcular

```
{
    "a": 15,
    "b": 5,
    "operacion": "suma"
}
```

```
{
    "estado": 200,
    "resultado": 20
}
```

Resuelve sumas, si llegaran a ingresar letras, marcaria un error.



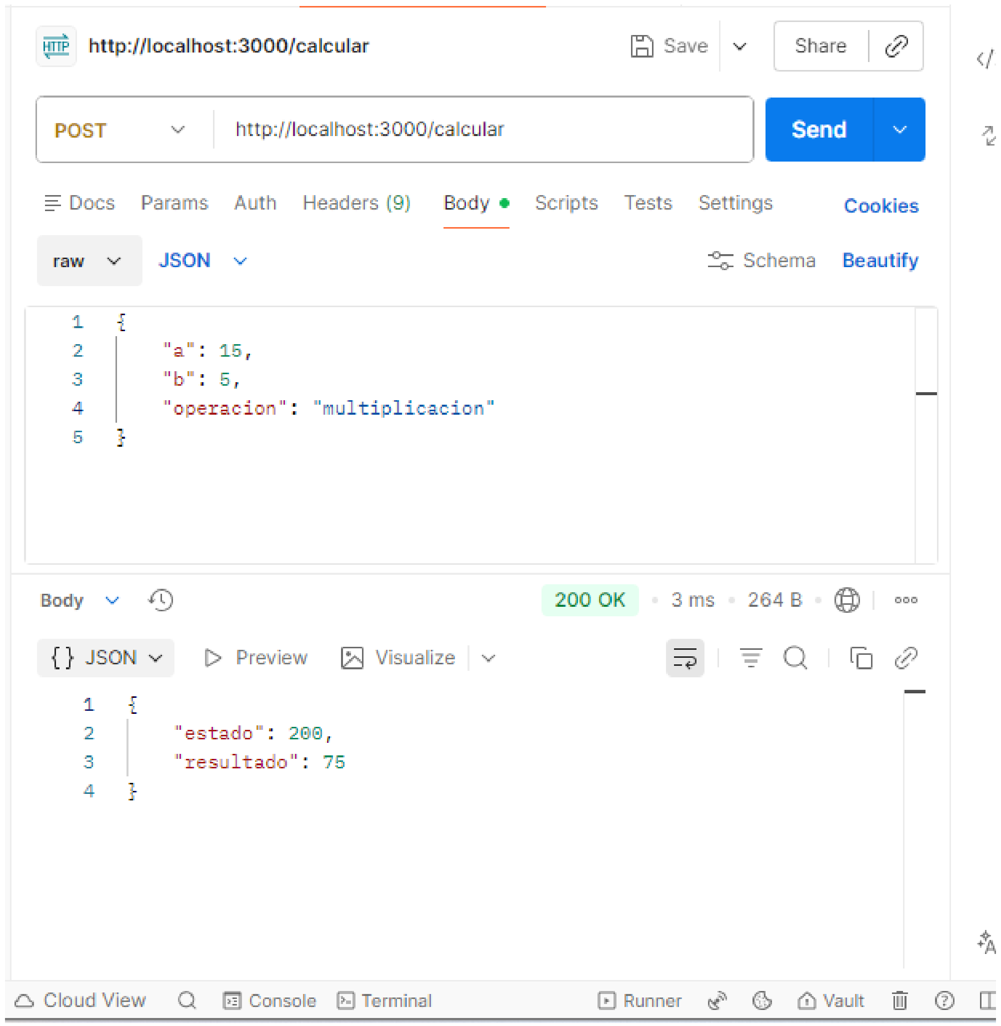
POST http://localhost:3000/calcular

```
{
    "a": 15,
    "b": 5,
    "operacion": "resta"
}
```

Body 200 OK • 5 ms • 264 B

```
{
    "estado": 200,
    "resultado": 10
}
```

Devuelve el resultado de una resta.

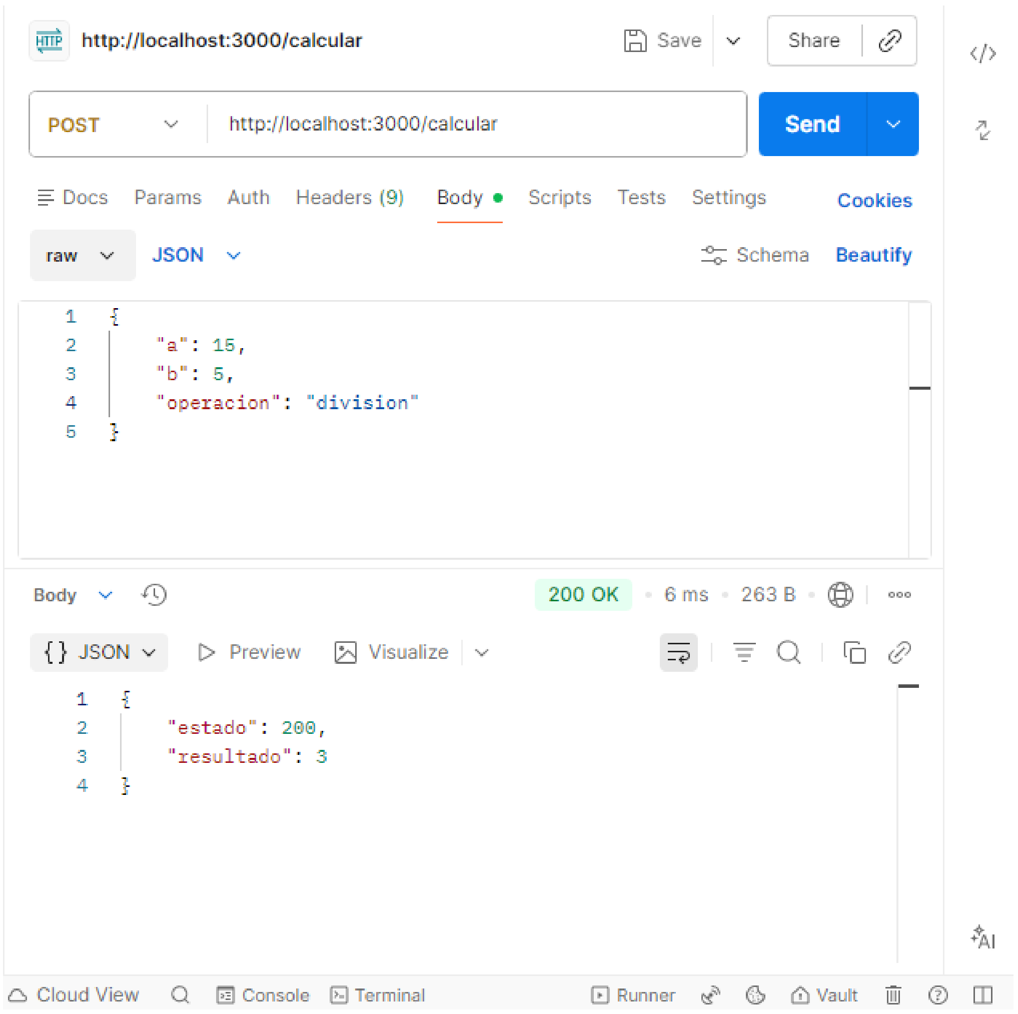


POST http://localhost:3000/calcular

```
{
    "a": 15,
    "b": 5,
    "operacion": "multiplicacion"
}
```

```
{
    "estado": 200,
    "resultado": 75
}
```

Devuelve el resultado de una multiplicación.



POST http://localhost:3000/calcular

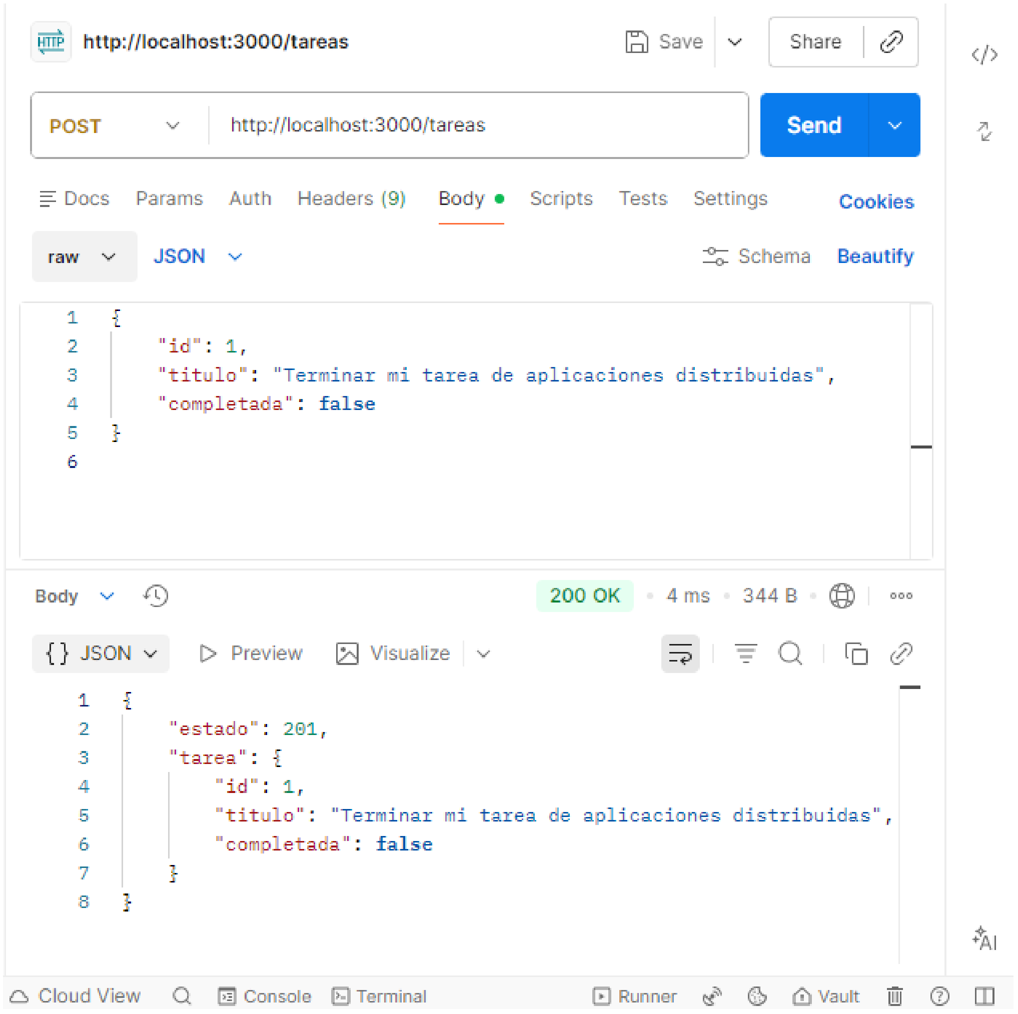
```
{
    "a": 15,
    "b": 5,
    "operacion": "division"
}
```

200 OK • 6 ms • 263 B

```
{
    "estado": 200,
    "resultado": 3
}
```

Devuelve el resultado de una división.

Ejercicio 3 : Gestor de Tareas (CRUD Básico)



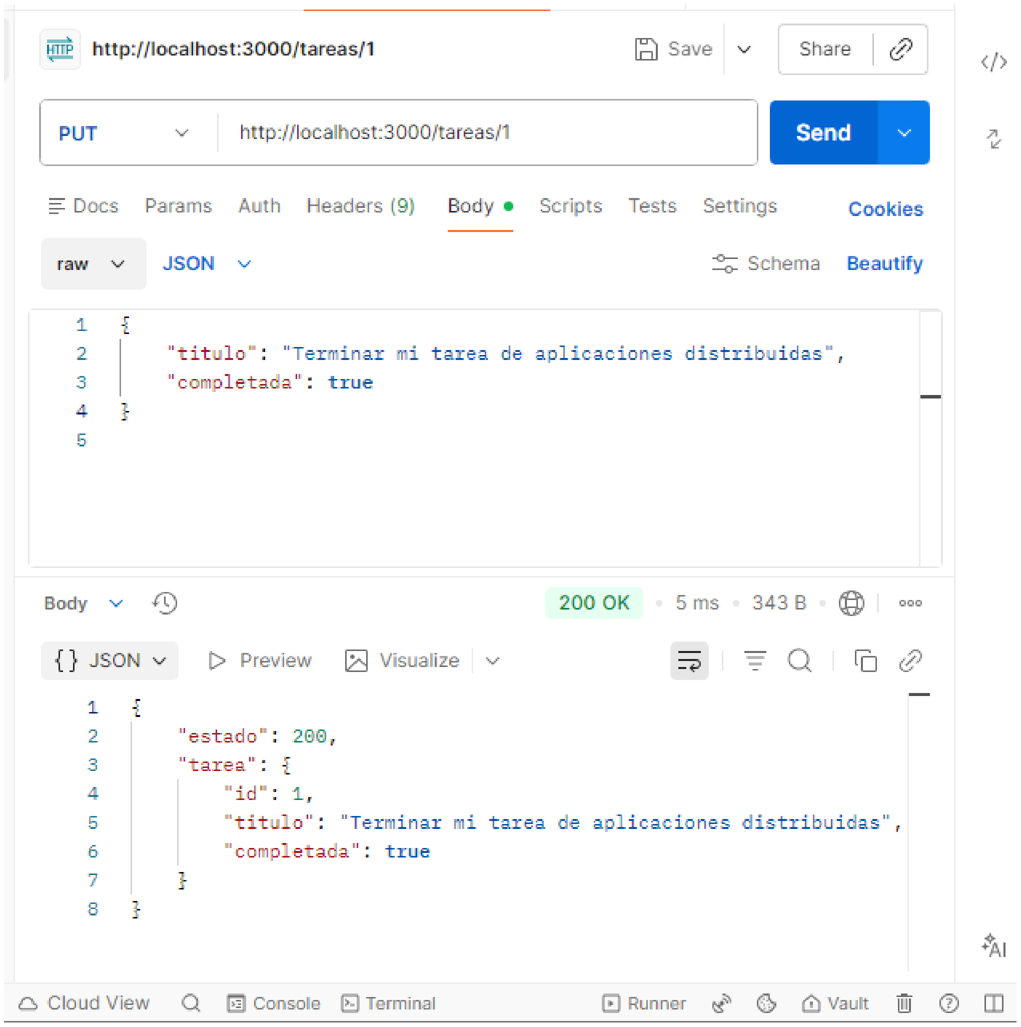
POST http://localhost:3000/tareas

```
{
    "id": 1,
    "titulo": "Terminar mi tarea de aplicaciones distribuidas",
    "completada": false
}
```

200 OK • 4 ms • 344 B

```
{
    "estado": 201,
    "tarea": {
        "id": 1,
        "titulo": "Terminar mi tarea de aplicaciones distribuidas",
        "completada": false
    }
}
```

En este caso utiliza el método POST para guardar la tarea con el Id que le quieras asignar después para poder localizar la tarea vuelves a usar ese id.



PUT http://localhost:3000/tareas/1

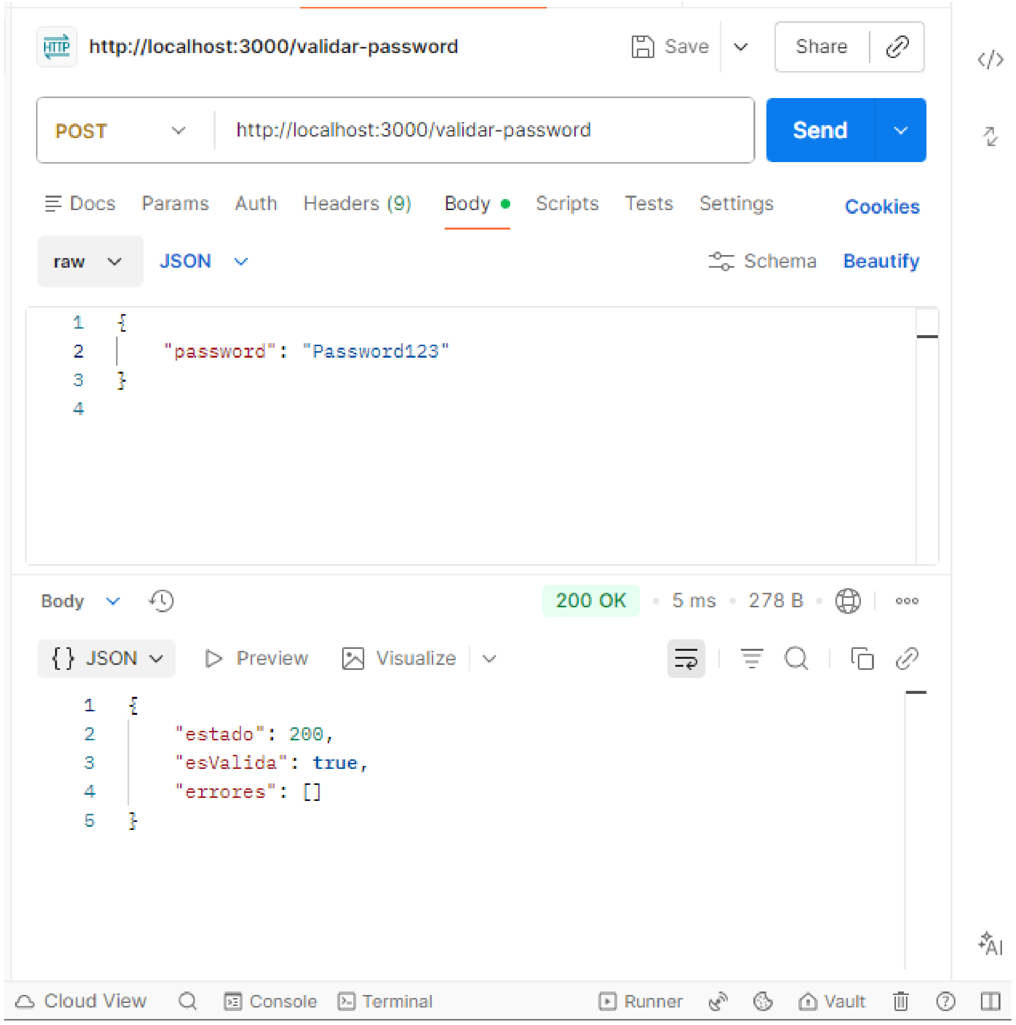
```
{
    "titulo": "Terminar mi tarea de aplicaciones distribuidas",
    "completada": true
}
```

200 OK · 5 ms · 343 B

```
{
    "estado": 200,
    "tarea": {
        "id": 1,
        "titulo": "Terminar mi tarea de aplicaciones distribuidas",
        "completada": true
    }
}
```

En este ejemplo se recupera la tarea que se ha guardado previamente. En este caso de una un método PUT.

Ejercicio 4 : Validador de Contraseñas



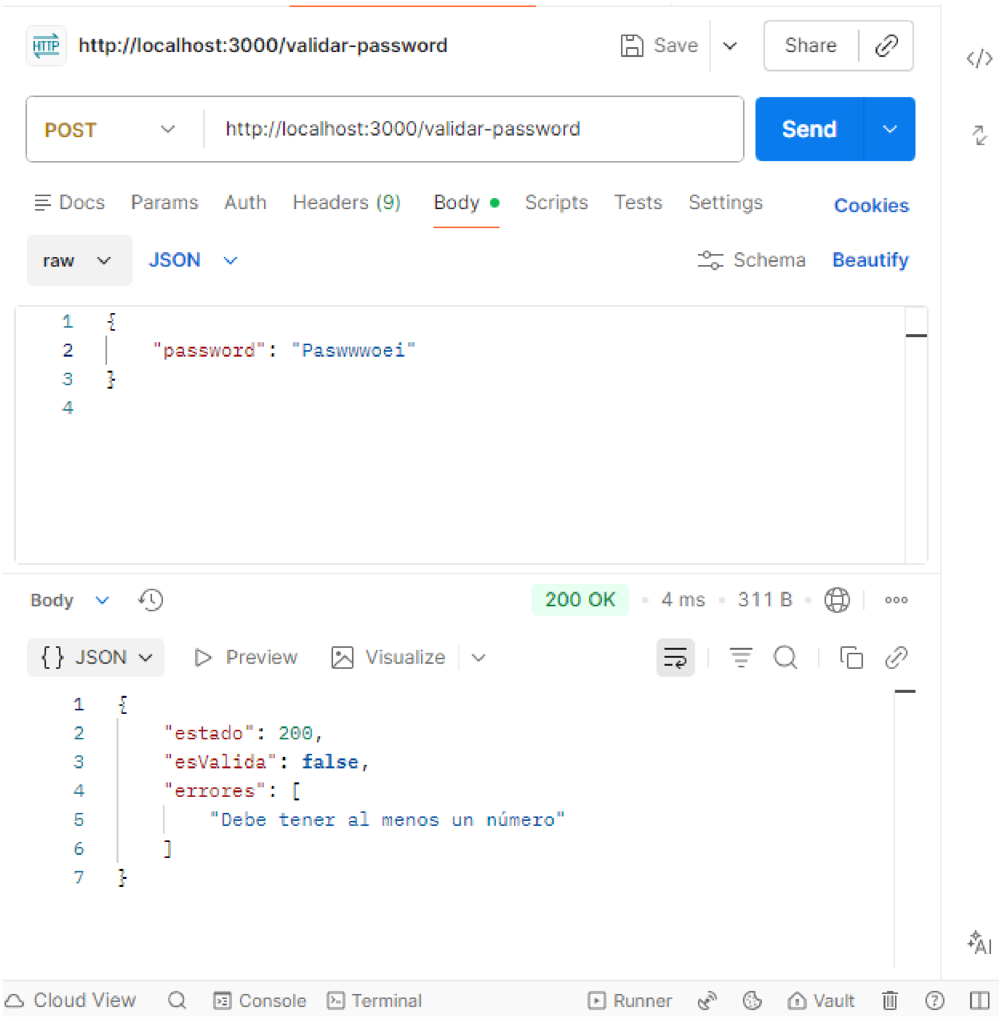
POST http://localhost:3000/validar-password

```
{
    "password": "Password123"
}
```

200 OK • 5 ms • 278 B

```
{
    "estado": 200,
    "esValida": true,
    "errores": []
}
```

Se hicieron diferentes validaciones para que la contraseña sera valida una vez que la ingresen.



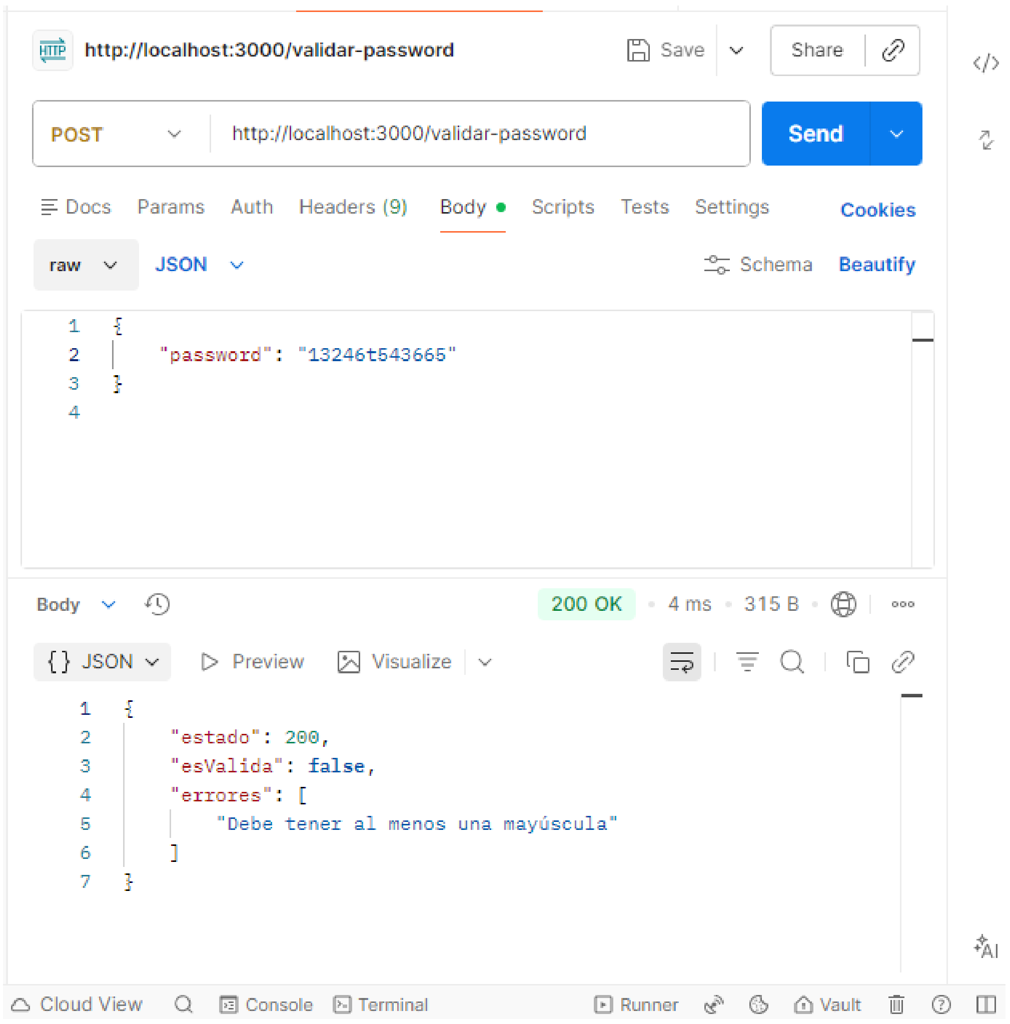
POST http://localhost:3000/validar-password

```
{
    "password": "Paswwwoei"
}

```

```
{
    "estado": 200,
    "esValida": false,
    "errores": [
        "Debe tener al menos un número"
    ]
}
```

Esta opción para no hacer valida cuando lo ingresan una contraseña que le hacen falta caracteres en mayúsculas.

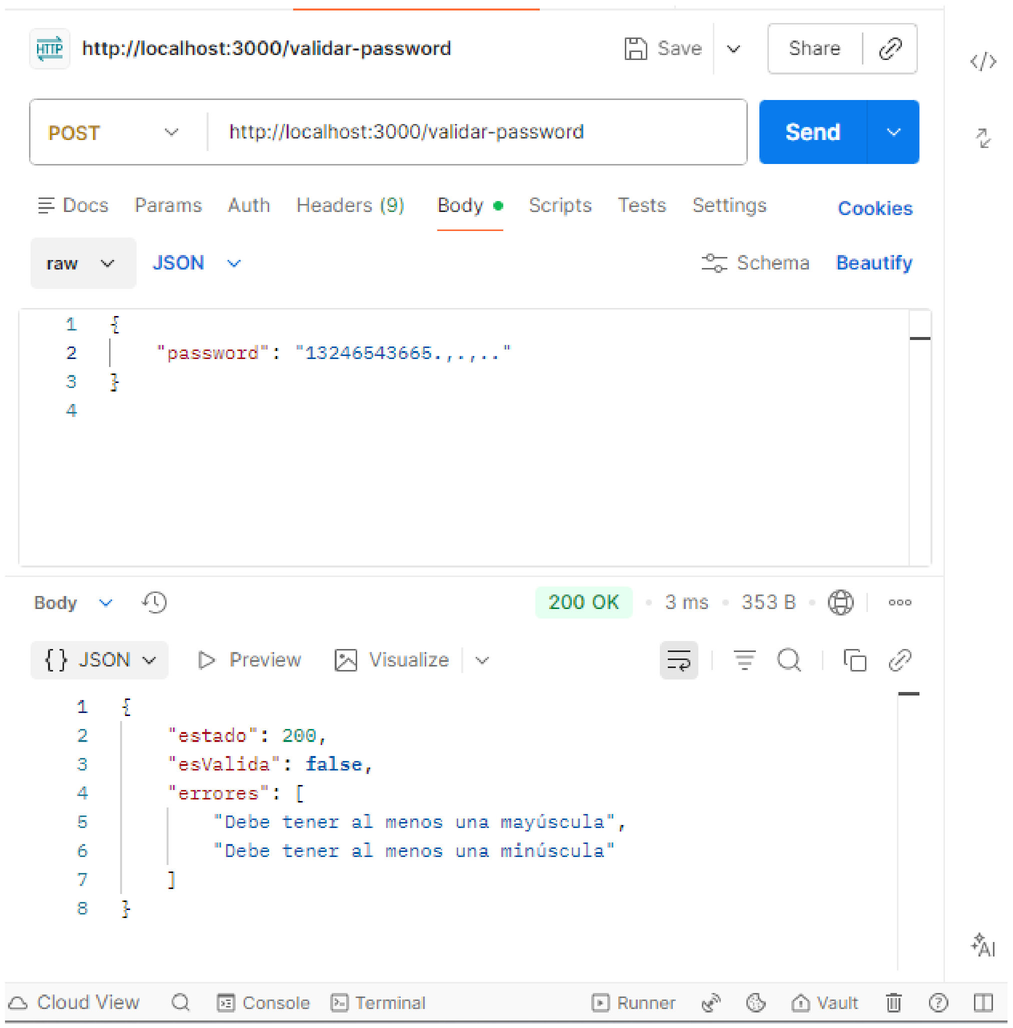


http://localhost:3000/validar-password

```
{
    "password": "13246t543665"
}
```

```
{
    "estado": 200,
    "esValida": false,
    "errores": [
        "Debe tener al menos una mayúscula"
    ]
}
```

Esta otra opción donde se argumenta que no ingresaron números en el password.



http://localhost:3000/validar-password

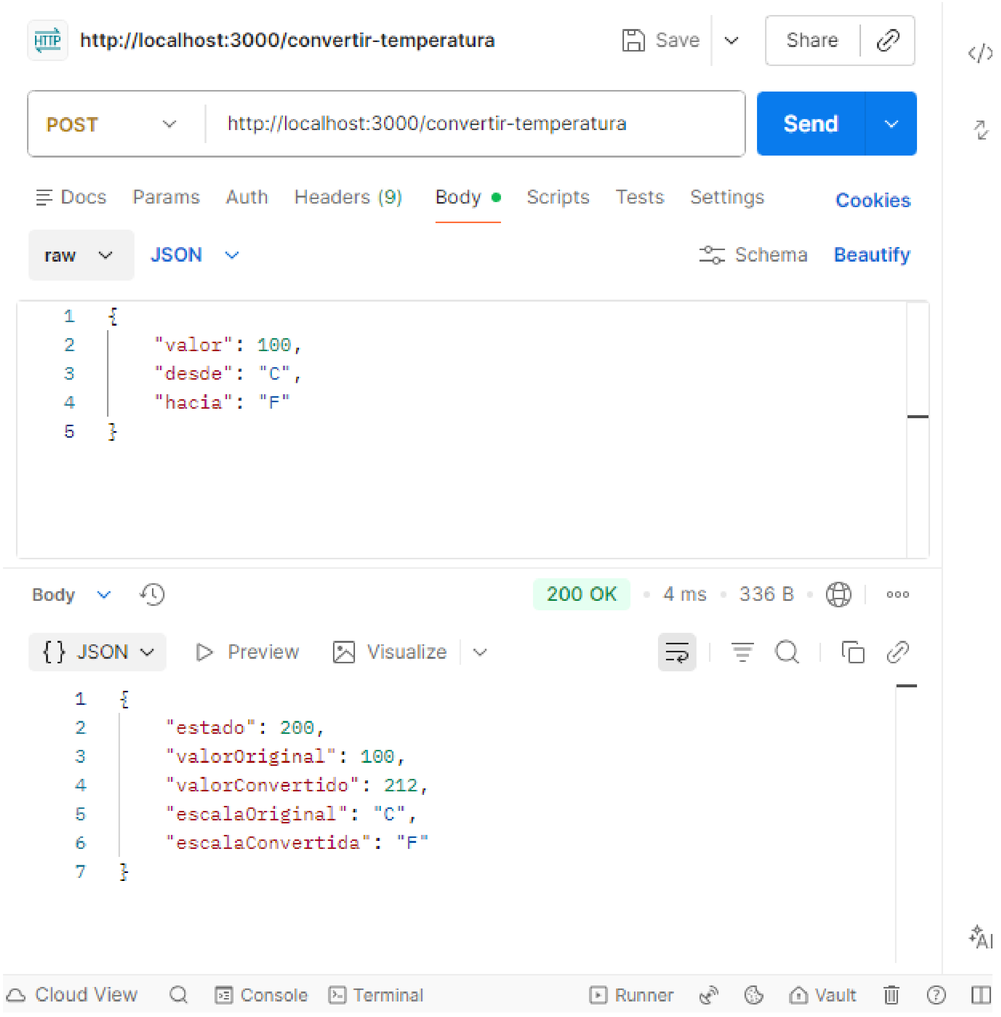
```
{
    "password": "13246543665.,.,.."
}

```

```
{
    "estado": 200,
    "esValida": false,
    "errores": [
        "Debe tener al menos una mayúscula",
        "Debe tener al menos una minúscula"
    ]
}
```

Al igual que en los anteriores falto algún carácter por completar.

Ejercicio 5 : Conversor de Temperatura



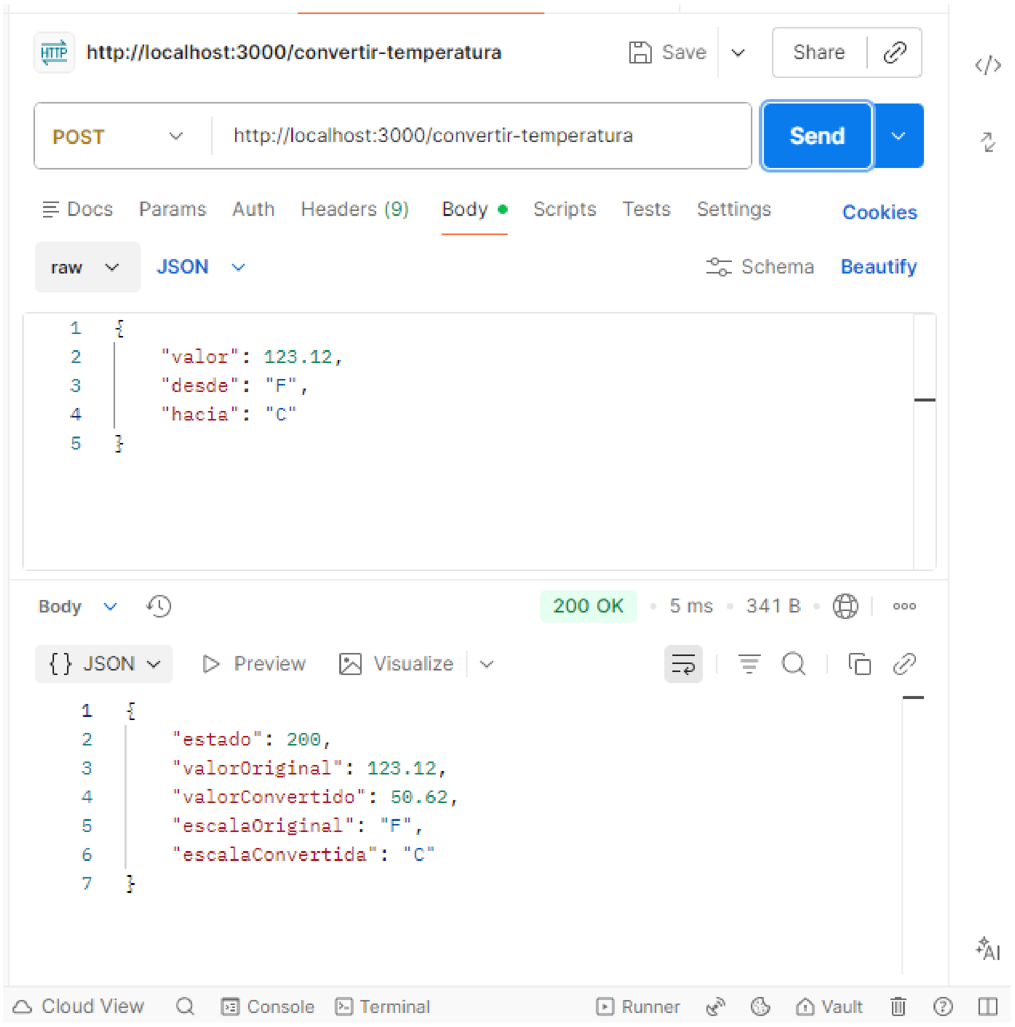
POST http://localhost:3000/convertir-temperatura

```
{
    "valor": 100,
    "desde": "C",
    "hacia": "F"
}
```

200 OK

```
{
    "estado": 200,
    "valorOriginal": 100,
    "valorConvertido": 212,
    "escalaOriginal": "C",
    "escalaConvertida": "F"
}
```

El ejercicio pedía una conversión en los valores en que se mide la temperatura y se muestra la validación para la conversión en la temperatura, en este caso de C a F.



POST http://localhost:3000/convertir-temperatura

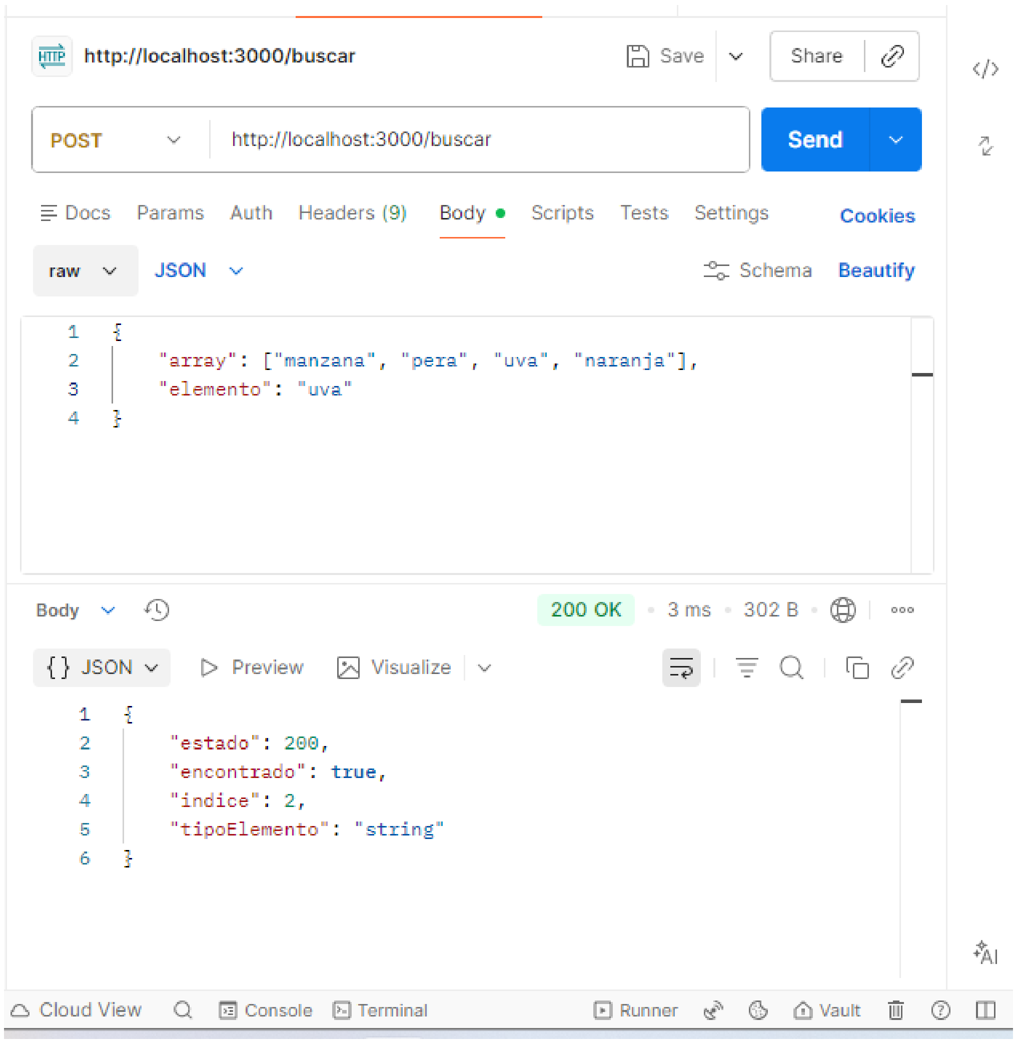
```
{
    "valor": 123.12,
    "desde": "F",
    "hacia": "C"
}
```

Response: 200 OK • 5 ms • 341 B

```
{
    "estado": 200,
    "valorOriginal": 123.12,
    "valorConvertido": 50.62,
    "escalaOriginal": "F",
    "escalaConvertida": "C"
}
```

En este caso de ver el caso al contrario se convierte de F a C.

Ejercicio 6 : Buscador en Array

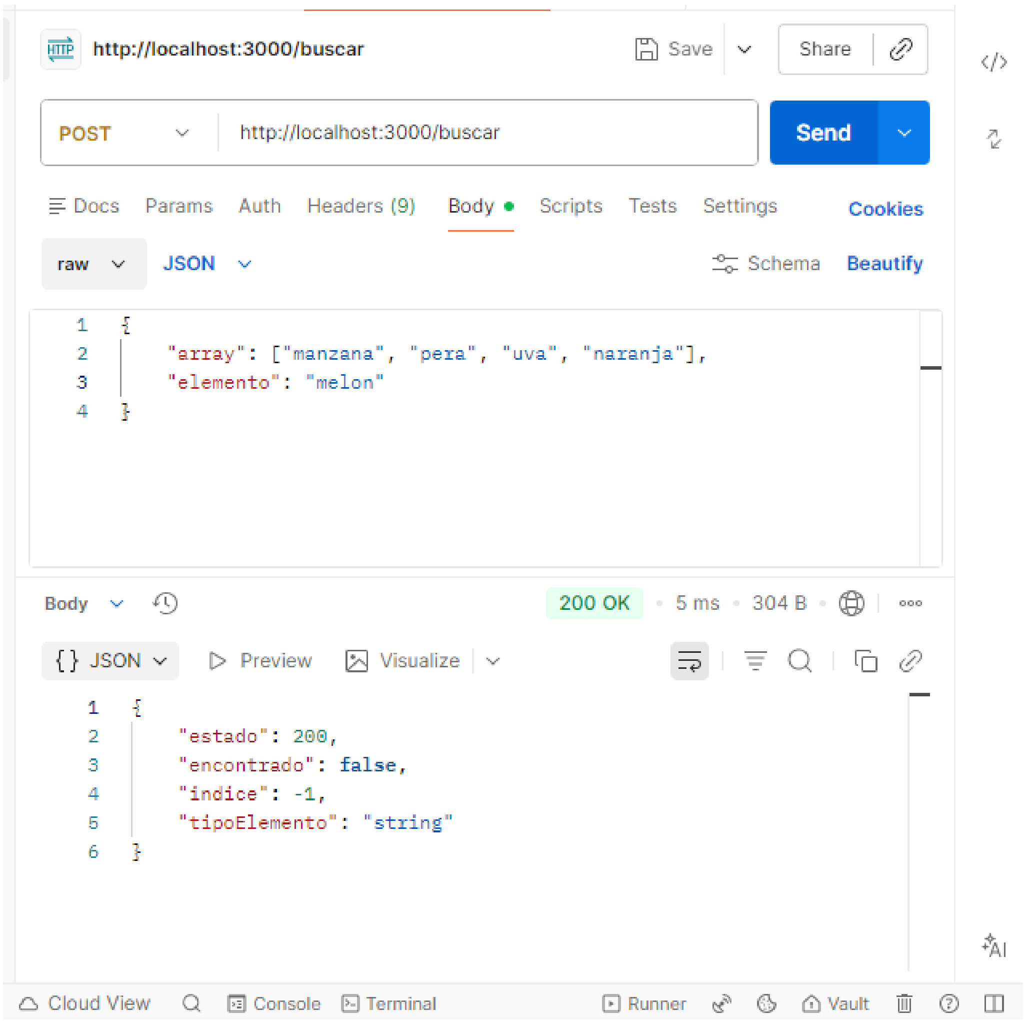


POST http://localhost:3000/buscar

```
{
    "array": ["manzana", "pera", "uva", "naranja"],
    "elemento": "uva"
}
```

```
{
    "estado": 200,
    "encontrado": true,
    "indice": 2,
    "tipoElemento": "string"
}
```

Dentro de una lista de objetos se pide ingresar uno. El programa te devolverá el objeto que seleccionaste y si realmente existe dentro de la lista.



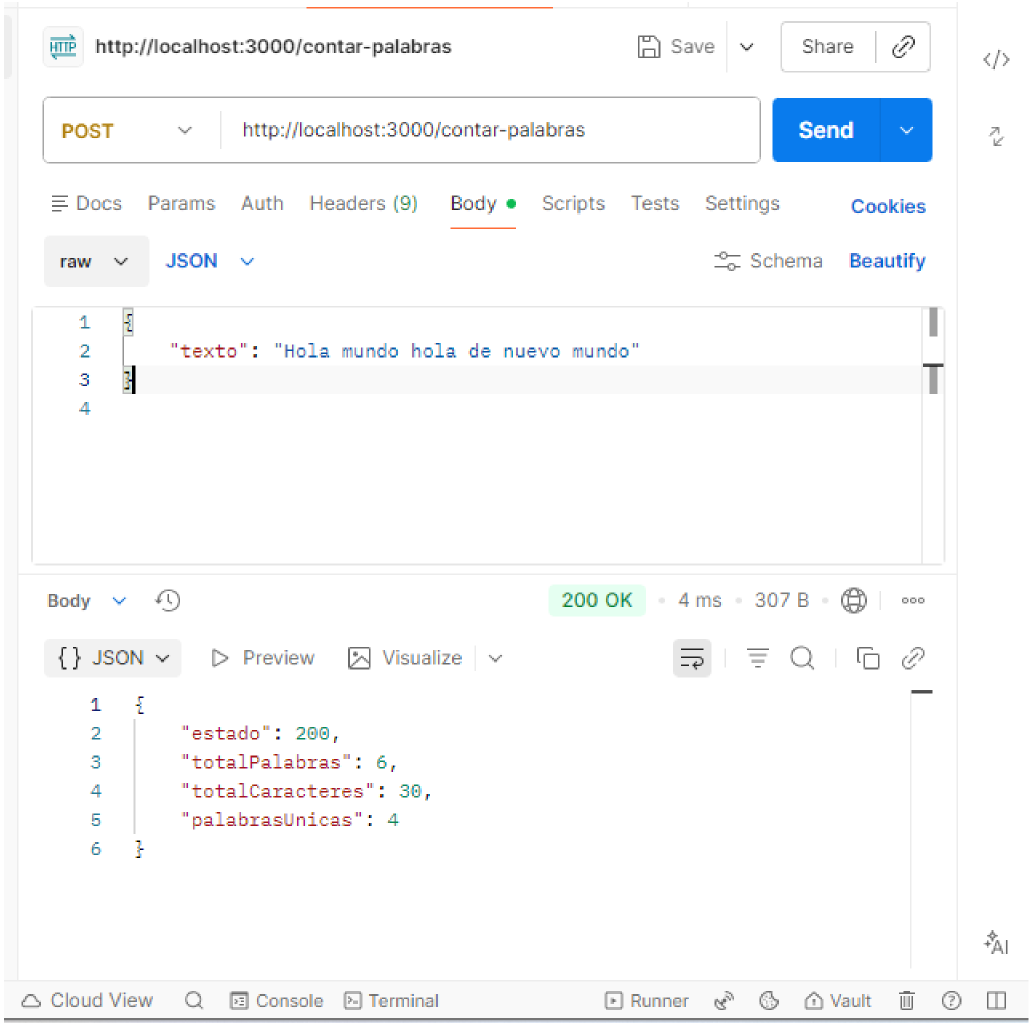
POST http://localhost:3000/buscar

```
{
    "array": ["manzana", "pera", "uva", "naranja"],
    "elemento": "melon"
}
```

```
{
    "estado": 200,
    "encontrado": false,
    "indice": -1,
    "tipoElemento": "string"
}
```

En este ejemplo se buscó un objeto que no está dentro de la lista.

Ejercicio 7 : Contador de Palabras



POST http://localhost:3000/contar-palabras

```
{
    "texto": "Hola mundo hola de nuevo mundo"
}
```

```
{
    "estado": 200,
    "totalPalabras": 6,
    "totalCaracteres": 30,
    "palabrasUnicas": 4
}
```

En este ejercicio se cuanta el total de palabras, los caracteres (se cuenta el carácter de espacio para hacer ese conteo) y nos muestra cuantas palabras no se repiten.

Conclusiones

Las API son conocidas por poder estar en cual quier lugar y en cualquier dispositivo y una vez que se combina con Restful que trabaja directamente con JSON se vuelve en una transmisión mucho más rápida de la información, esto también debido a que utilizan el protocolo TCP y aunque también podrían utilizan algún otro protocolo como UDP este por ejemplo es muy más lento, también más confiable pero TCP tiene una velocidad que no se le comprara y en nuestra comunicación actual siempre buscamos la velocidad.