Visual Studio Code Thunder Client es uno de los clientes que poco a poco se ha ido haciendo hueco a la hora de realizar peticiones REST . Recordemos que cada día usamos más Visual Studio Code . Vamos a ver un ejemplo sencillo de como realizar peticiones con este plugin de Visual Studio Code. Para ello empezaremos con un pequeño servicio REST que delega en Spring Data.

```
package com.arquitecturajava.web1.restcontrollers;

import java.util.List;
import java.util.Optional;

import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.arquitecturajava.web1.models.Producto;
import com.arquitecturajava.web1.services.ProductoService;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;


@RestController
@RequestMapping("webapi/productos")
public class ProductoRestController {

    private ProductoService productoService;

    public ProductoRestController(ProductoService productoService) {
```

```
        this.productoService = productoService;
    }

    @DeleteMapping("/{id}")
    public void deleteProducto(@PathVariable int id) {
        productoService.deleteProducto(new Producto(id));
    }

    @GetMapping
    public List<Producto> findAllProductos() {
        return productoService.findAllProductos();
    }
    @GetMapping("/{id}")
    public Optional<Producto> findByIdProducto(@PathVariable int id) {
        return productoService.findByIdProducto(id);
    }
   @PostMapping
    public Producto saveProducto(@RequestBody Producto entity) {
        return productoService.saveProducto(entity);
    }

    @PutMapping ("/{id}")
    public Producto actualizarProducto(@RequestBody Producto nuevo
,@PathVariable int id) {
       Optional<Producto> oProducto=
productoService.findByIdProducto(id);
        if(oProducto.isPresent()) {

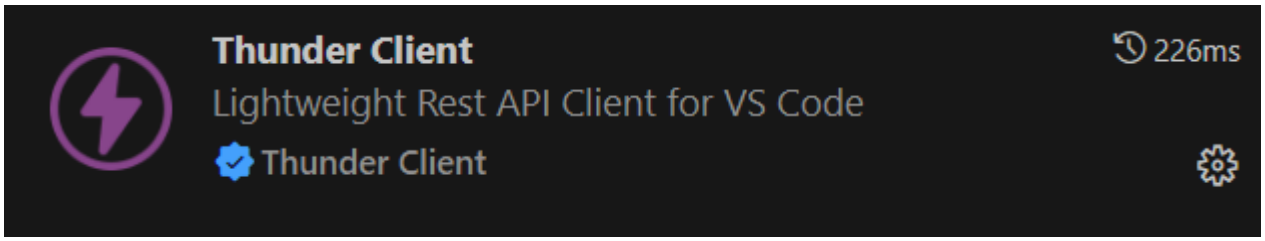
            Producto productoActualizar= oProducto.get();
            productoActualizar.setNombre(nuevo.getNombre());
            productoActualizar.setPrecio(nuevo.getPrecio());
```

```
            productoActualizar.setCantidad(nuevo.getCantidad());
            return productoService.saveProducto(productoActualizar);
        }
        return null;
    }

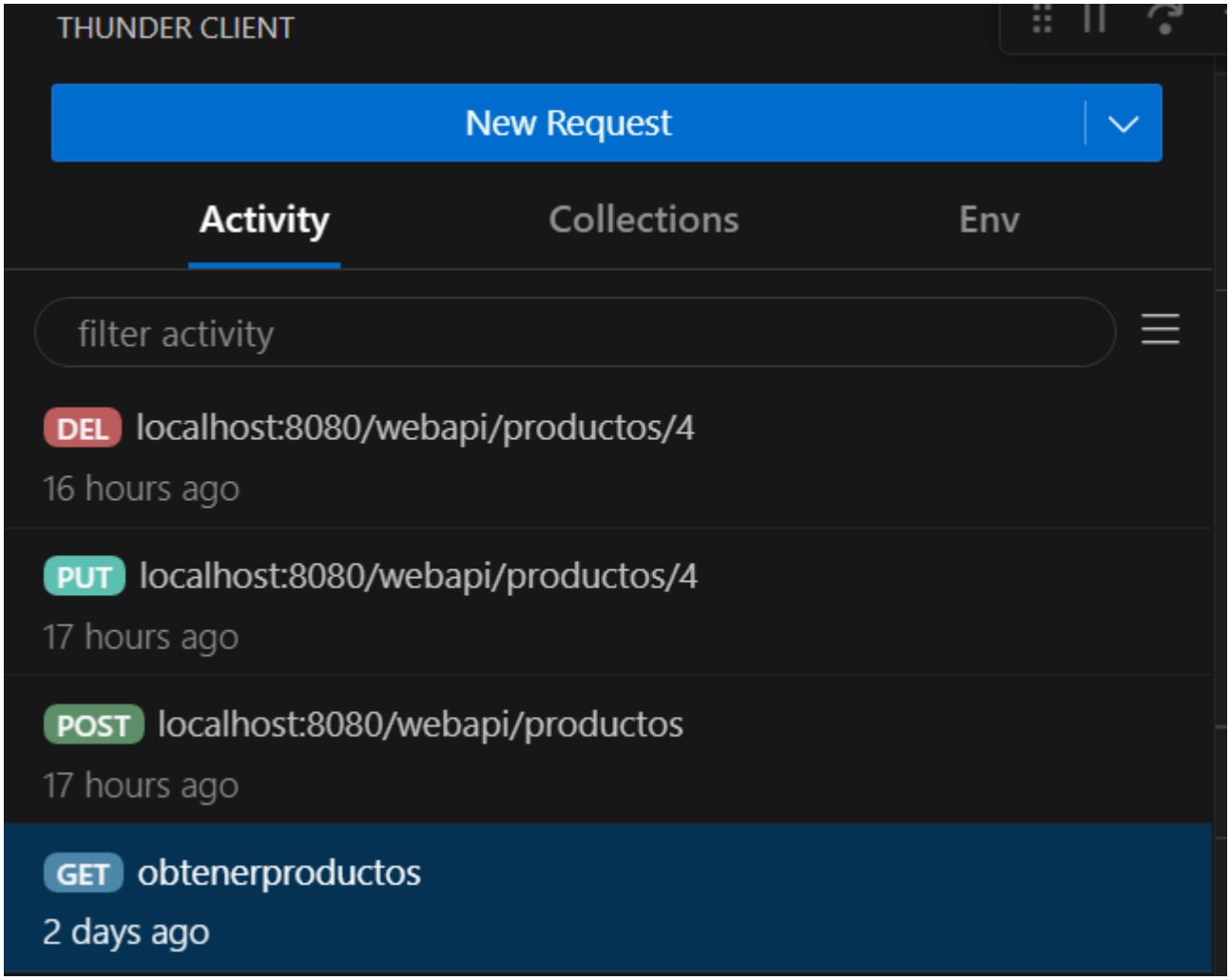
}
```

Visual Studio Code Thunder Client

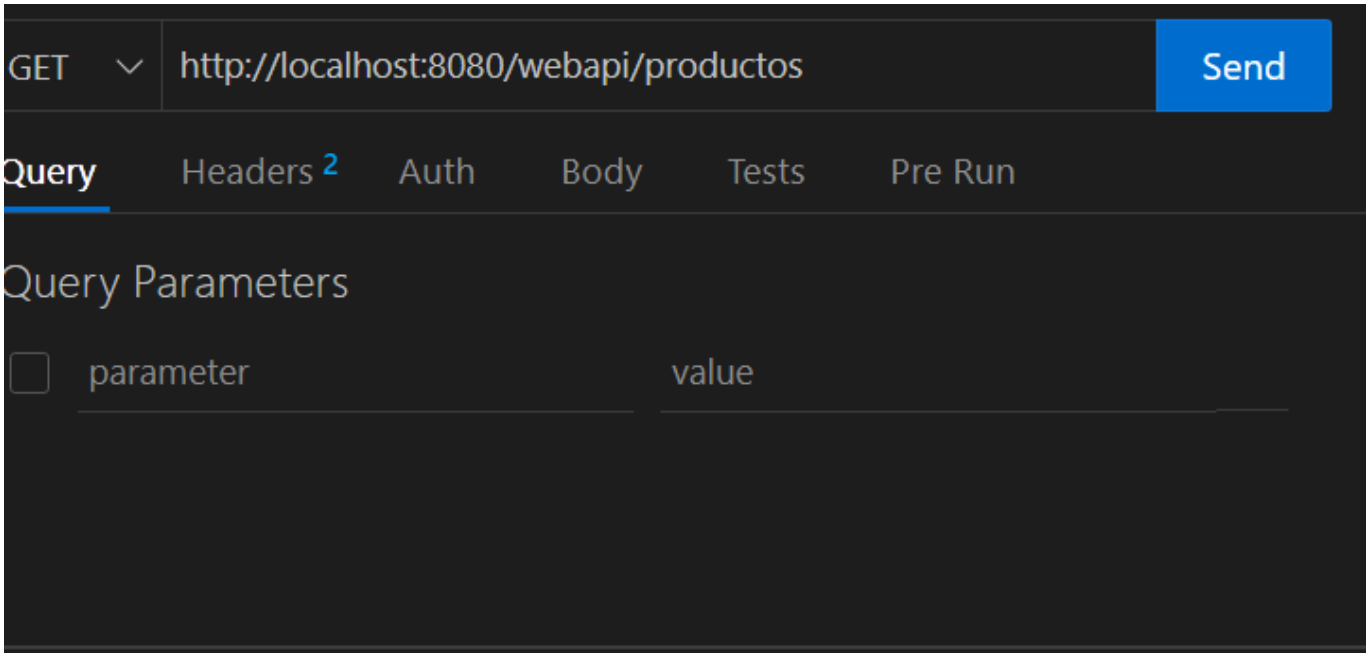
Este es un servicio que contiene las operaciones más habituales de GET , POST, PUT y DELETE. Podemos instalar Thunder Client para gestionarlas.



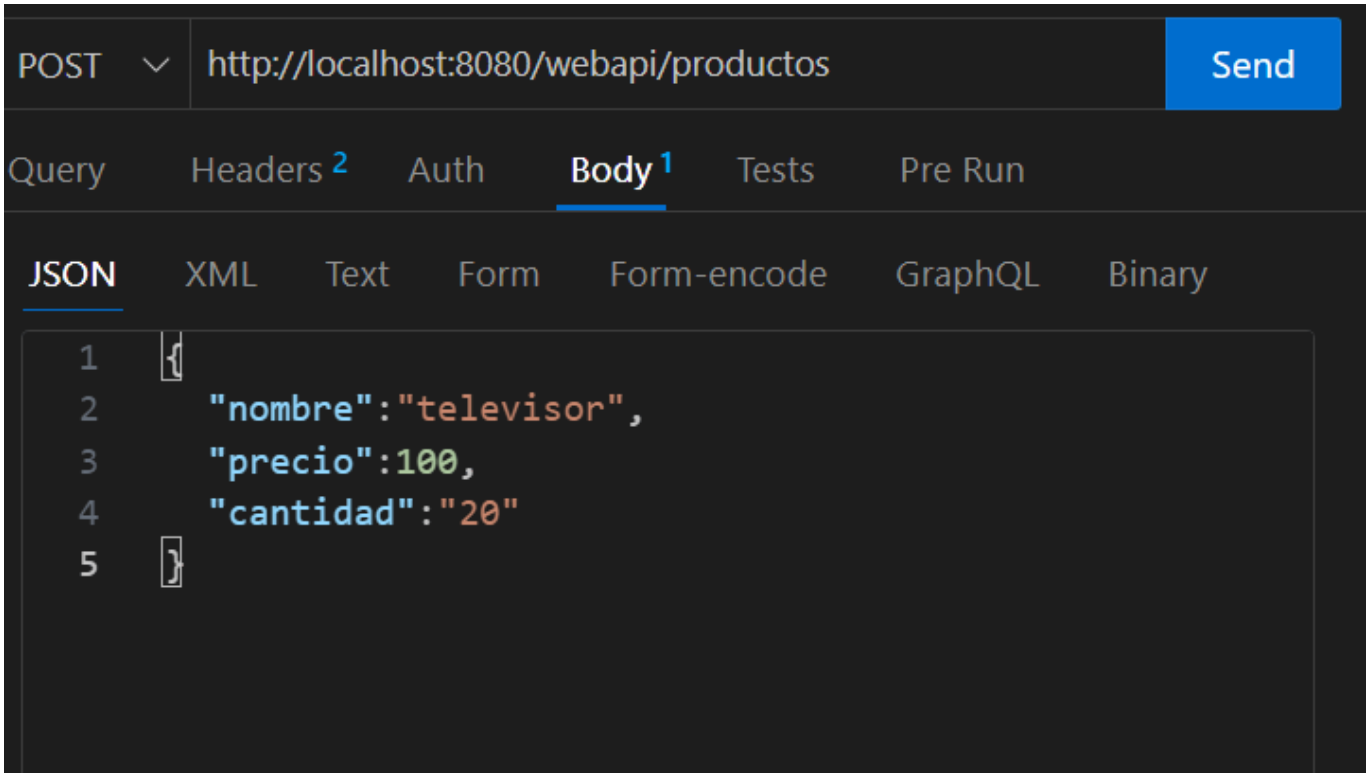
Este cliente moderno de peticiones REST tiene un aire a PostMan y es gratuito para un conjunto importante de operaciones aunque tiene una versión de pago. Podemos construir operaciones REST de una forma sencilla.



Para ello basta con solicitar new request y ver como podemos definir los diferentes tipos de petición en la siguiente imagen se muestra una petición de tipo GET:



Aquí tenemos una petición de tipo POST:



Poco a poco se esta convirtiendo en el cliente de referencia a nivel de Visual Studio code cuando tenemos que gestionar diferentes peticiones REST y substituyendo a REST Client

que era otra de las grandes referencias por su simplicidad. Si te gusta PostMan Thunder Client es probablemente una alternativa a valorar:

- Visual Studio Code REST Client
- Spring Boot Visual Studio Code y extensiones
- Visual Studio Code Java y Extensiones
- Visual Studio Code Tomcat y su configuración
- REST HTTP return codes y sus curiosidades