

PRACTICA 01 PRACTICA CON REST API

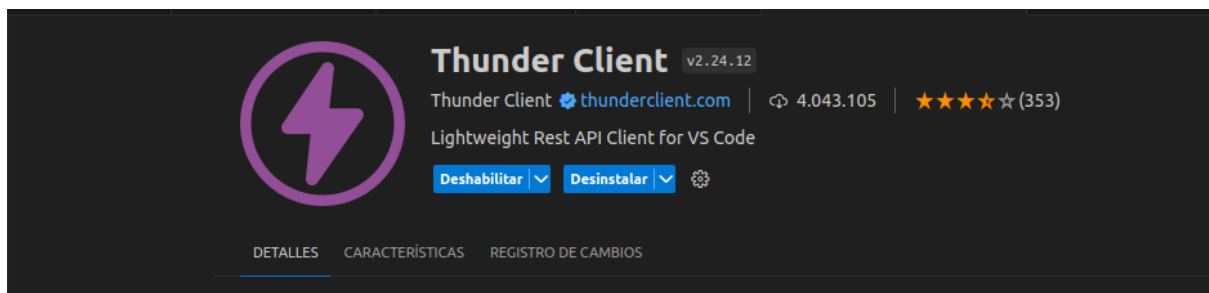
CREAR UN REPOSITORIO PARA LA UNIDAD 3 LLAMADO [INICIALES]-AUTOMATIZA-U3

giri4091-mcr Public

Ejercicios unidad III

Python Updated 3 days ago

ABRIR VISUAL STUDIO CODE E INSTALAR EL MÓDULO THUNDER CLIENT PARA REALIZAR PETICIONES.



VISITAR LA URL GITHUB FAKESTORE PARA CONOCER MÁS SOBRE LA API

Products

- Get all products
- Get a single product
- Limit results
- Sort results
- Get all categories
- Get in category
- Add new product
- Update a product
- Delete a product

Cart

- Get all
- Get a single
- Limit results
- Sort results
- get in date range
- get user cart
- Add new cart

How to use it

fakeStoreApi can be used with any type of shopping project that needs products, carts, and users in JSON format, you can use examples below to check how fakeStoreApi works and feel free to enjoy it in your awesome projects!

Products

Get all products

```
fetch('https://fakestoreapi.com/products')
  .then(res=>res.json())
  .then(json=>console.log(json))
```

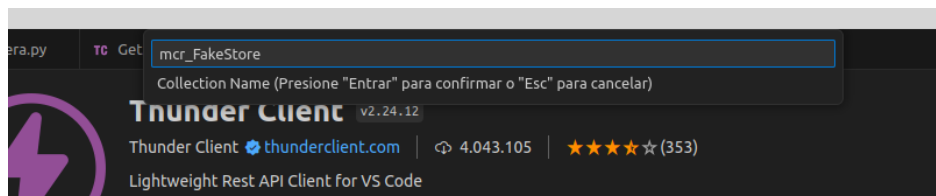
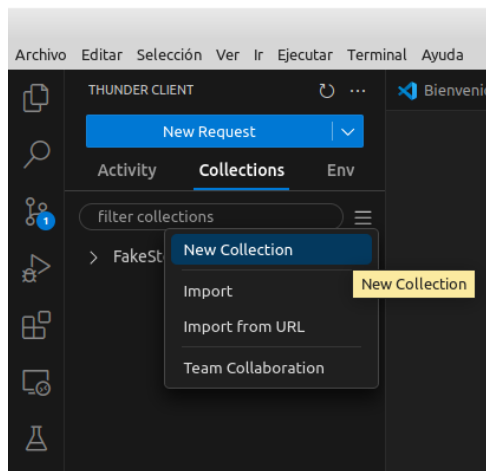
Show output

Get a single product

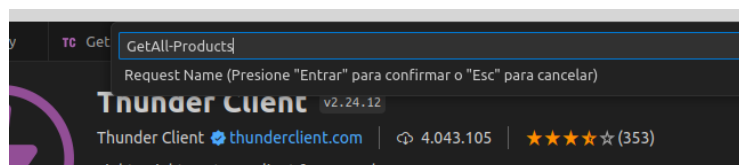
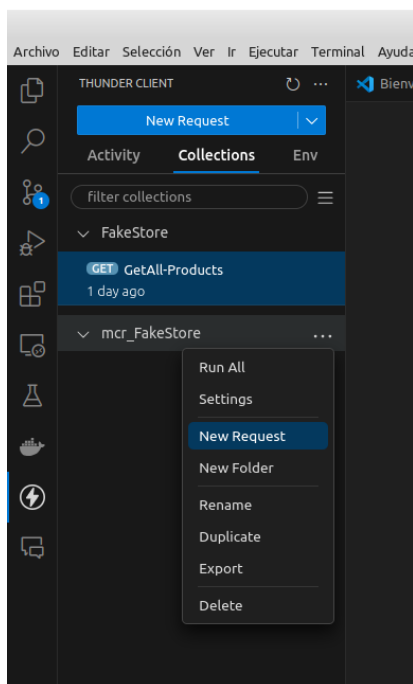
```
fetch('https://fakestoreapi.com/products/1')
  .then(res=>res.json())
  .then(json=>console.log(json))
```

Show output

CON LA EXTENSIÓN THUNDER CLIENT EN VISUAL STUDIO CODE CREAR UNA COLECCIÓN LLAMADA [INICIALES]-FAKESTORE



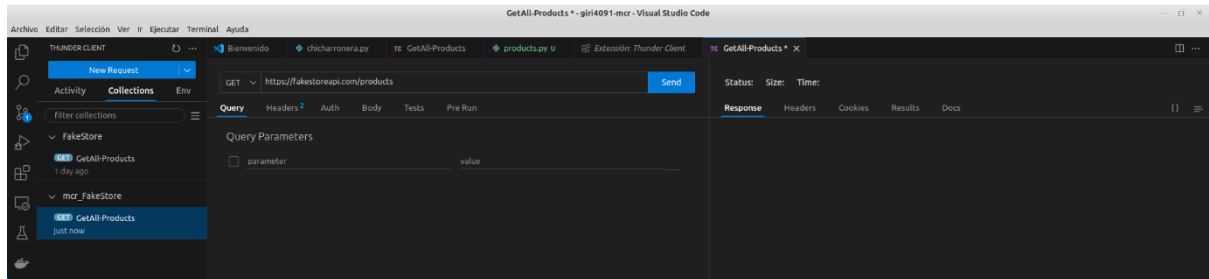
AGREGAR UNA NUEVA PETICIÓN LLAMADA GETALL-PRODUCTS CON EL MÉTODO GET



ENVIAR LA PETICIÓN PARA EXTRAER TODOS LOS PRODUCTOS

Get all products

```
fetch('https://fakestoreapi.com/products')
  .then(res=>res.json())
  .then(json=>console.log(json))
```

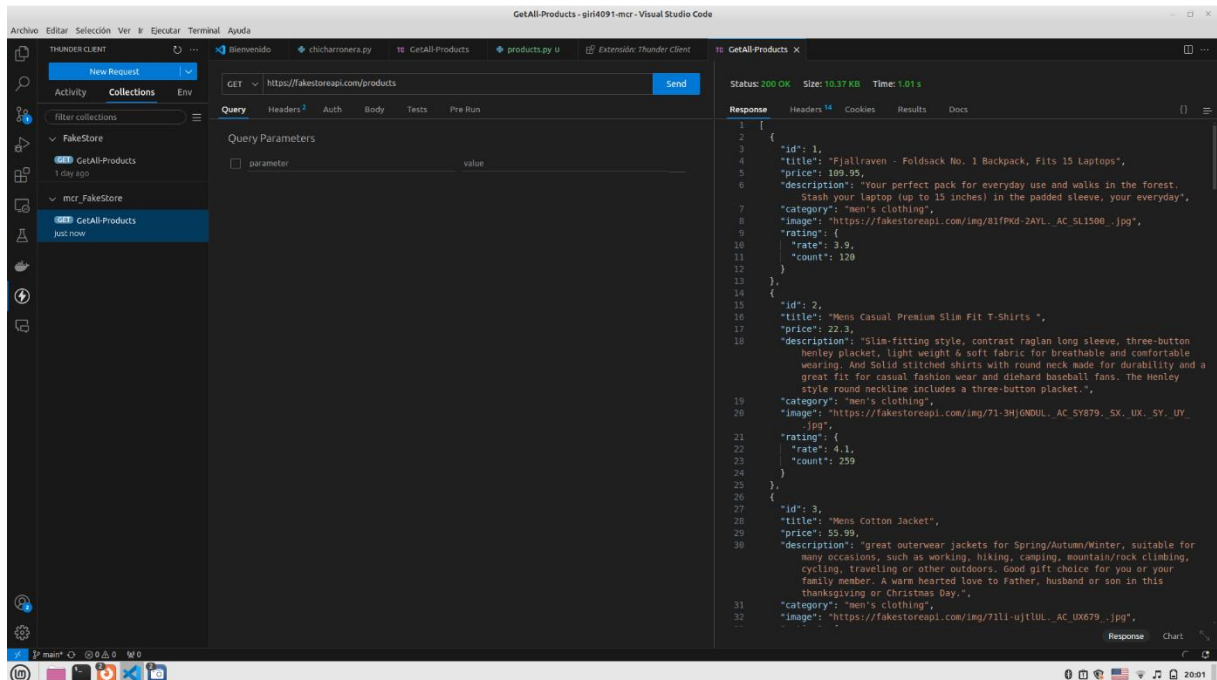


VERIFICACIÓN EN LA RESPUESTA, EL CÓDIGO DE RESPUESTA, Y LOS HEADERS

GET ALL PRODUCTS

Crear una nueva petición en la colección para extraer todos los productos

PETICIÓN EN THUNDER CLIENT



HEADERS

The screenshot shows the Thunder Client interface in Visual Studio Code. A GET request to `https://fakestoreapi.com/products` has been executed successfully, returning a 200 OK status. The response headers are displayed in the right-hand pane.

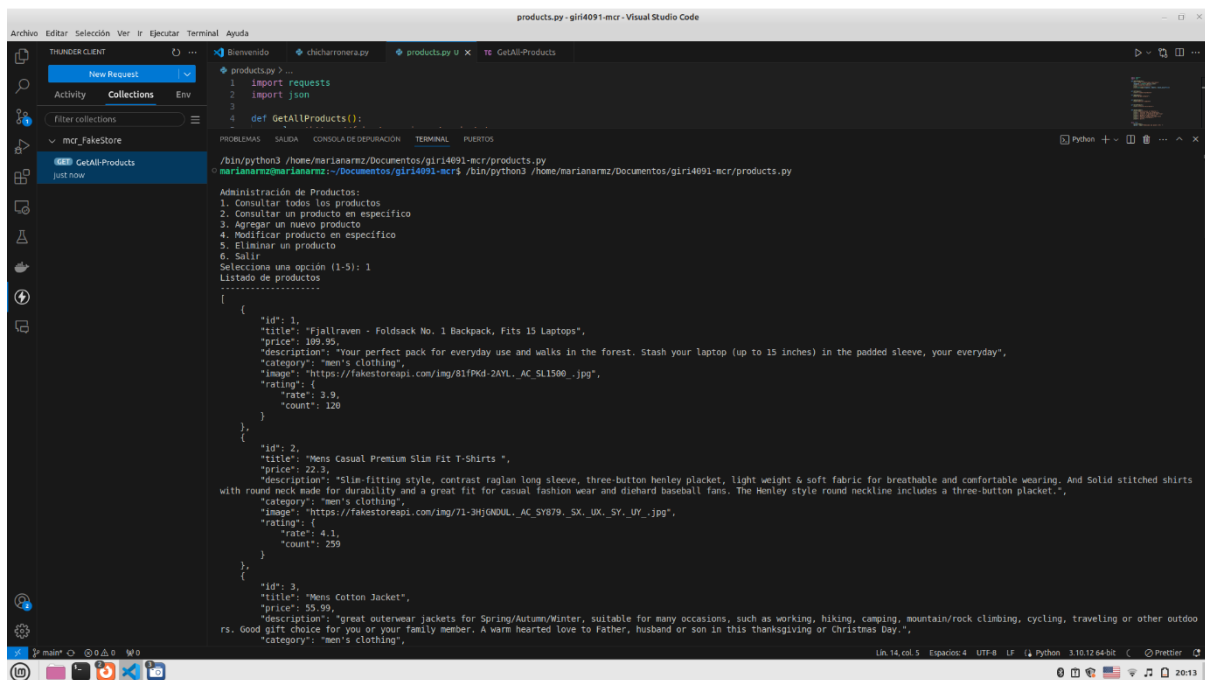
Header	Value
date	Fri, 05 Jul 2024 02:01:30 GMT
content-type	application/json; charset=utf-8
transfer-encoding	chunked
connection	close
access-control-allow-origin	*
etag	W/"297c-h+n6N83M5156528jZo3BluzvU"
x-powered-by	Express
cf-cache-status	DYNAMIC
report-to	{\"endpoints\": [{\"url\": \"https://x.net.cloudflare.com/reportV4?ts=VrrIdw26Lx1DXxTf3u3PwFemajpuh8K5dnCQn80wW5Cp8uM8uM1p307a53yuaM8u85D0d8N5...\"}], \"group\": \"cf-nel\", \"max_age\": 604800}
nel	{\"success_fraction\": 0, \"report_to\": \"cf-nel\", \"max_age\": 604800}
server	cloudflare
cf-ray	89e3c478d576b27-DFW
content-encoding	br
alt-svc	h3=\\\"443\\\"; ma=86400

CÓDIGO EN PYTHON

The screenshot shows the Python code for the `products.py` file in Visual Studio Code. The code defines several functions for interacting with the FakeStore API and a main menu loop.

```
1 import requests
2 import json
3
4 def GetAllProducts():
5     url = 'https://fakestoreapi.com/products'
6     respuesta = requests.get(url).json()
7     print('Estado de productos')
8     print('-----')
9     print(json.dumps(respuesta, indent=4, ensure_ascii=False))
10
11
12 def GetProduct():
13     print('Busqueda de producto')
14
15 def AddProduct():
16     print('Agregar producto')
17
18 def UpdateProduct():
19     print('Modificar producto')
20
21
22 def DeleteProduct():
23     print('Eliminación de producto')
24
25
26
27 def mostrar_menu():
28     print('Administración de Productos:')
29     print('1. Consultar todos los productos')
30     print('2. Consultar un producto en específico')
31     print('3. Agregar un nuevo producto')
32     print('4. Modificar producto en específico')
33     print('5. Eliminar un producto')
34     print('6. Salir')
35
36
37 while True:
38     mostrar_menu()
39     opcion = input('Selecciona una opción (1-5): ')
40
41     if opcion == '1':
42         GetAllProducts()
43     elif opcion == '2':
44         GetProduct()
45     elif opcion == '3':
46         AddProduct()
47     elif opcion == '4':
48         UpdateProduct()
```

SALIDA CON PYTHON



```
products.py - giri4091-mcr - Visual Studio Code

1 import requests
2 import json
3
4 def GetAllProducts():
5     url = "https://fakestoreapi.com/products"
6     response = requests.get(url)
7     data = response.json()
8     print(json.dumps(data, indent=2))
9
10 if __name__ == '__main__':
11     GetAllProducts()
```

Administración de Productos:

1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir

Seleccione una opción (1-5): 1

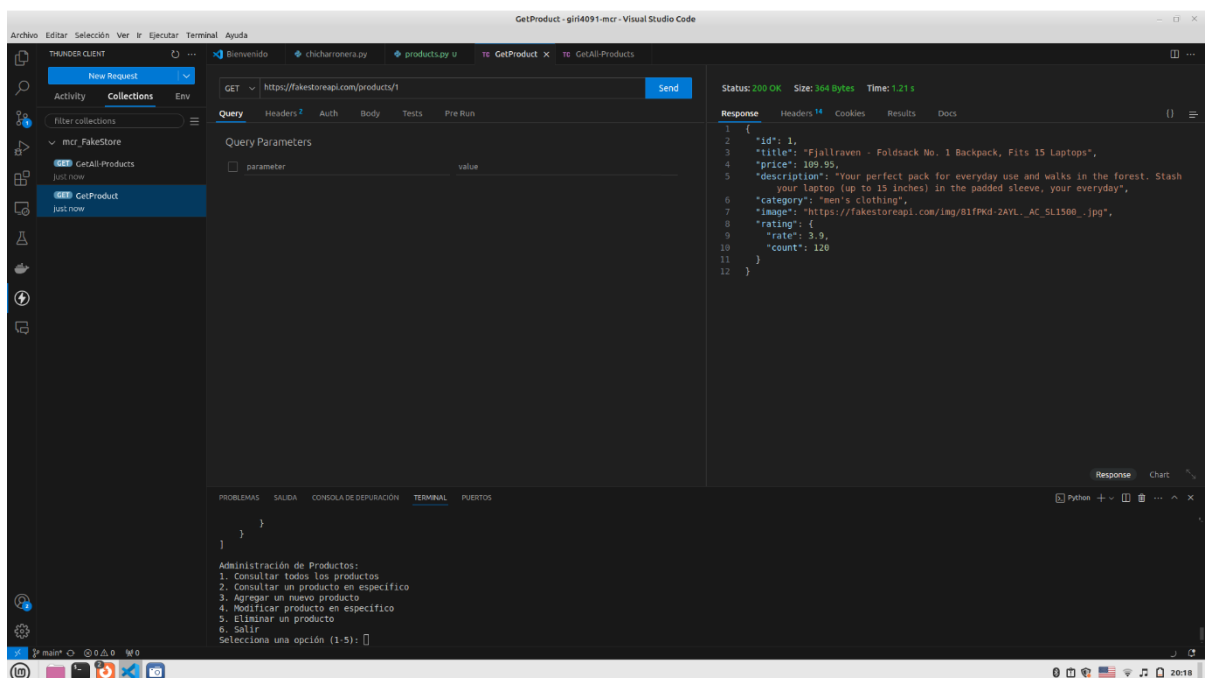
Listado de productos

```
{
  "id": 1,
  "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
  "price": 109.99,
  "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/81fPKd-2AYL_AC_SL1500_.jpg",
  "rating": {
    "rate": 3.9,
    "count": 120
  }
},
{
  "id": 2,
  "title": "Mens Casual Premium Slim Fit T-Shirts ",
  "price": 22.3,
  "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and dishard baseball fans. The Henley style round neckline includes a three-button placket.",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/71-3HjGNDUL_AC_SY879_SX_UX_1V_.jpg",
  "rating": {
    "rate": 4.1,
    "count": 259
  }
},
{
  "id": 3,
  "title": "Mens Cotton Jacket",
  "price": 55.99,
  "description": "great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/71-pjplBu7FU_LBqU8_1.jpg",
  "rating": {
    "rate": 4.5,
    "count": 232
  }
}
```

GET PRODUCT

Crear una nueva petición en la colección buscando un producto en particular.

PETICIÓN EN THUNDER CLIENT



Thunder Client interface showing a GET request to `https://fakestoreapi.com/products/1`. The response is a JSON object representing a product.

Query: `https://fakestoreapi.com/products/1`

Query Parameters:

parameter	value
-----------	-------

Response: Status: 200 OK, Size: 364 Bytes, Time: 1.21 s

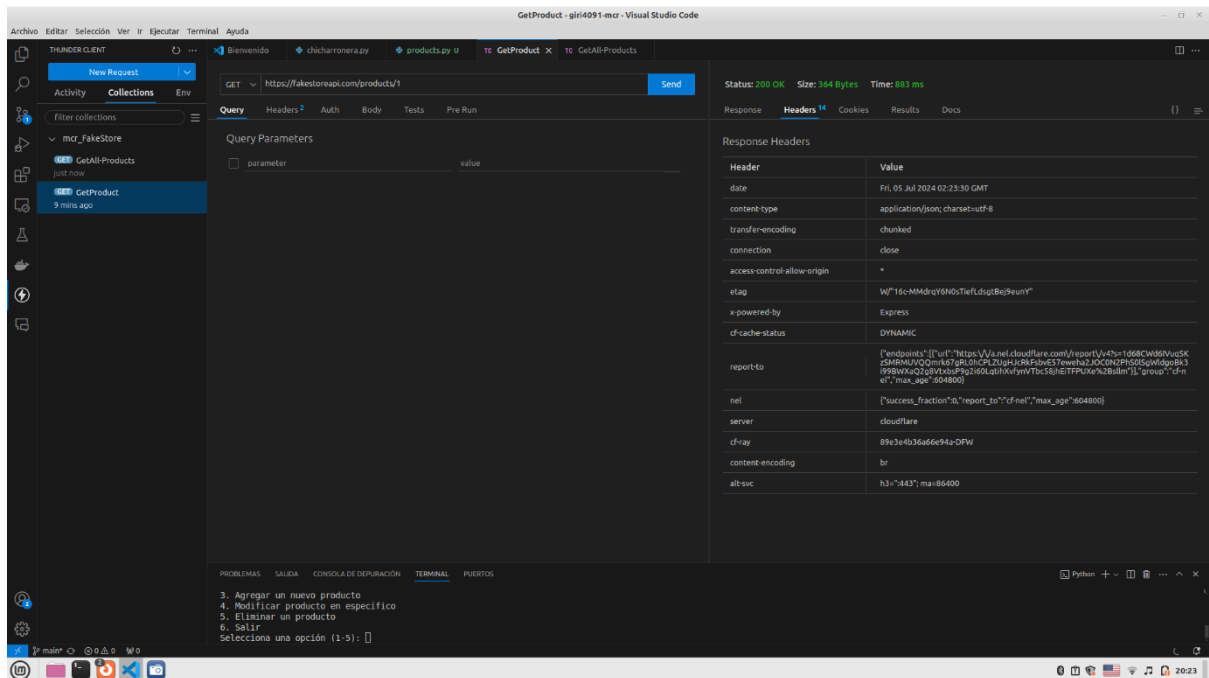
```
{
  "id": 1,
  "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
  "price": 109.99,
  "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/81fPKd-2AYL_AC_SL1500_.jpg",
  "rating": {
    "rate": 3.9,
    "count": 120
  }
}
```

Administración de Productos:

1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir

Seleccione una opción (1-5):

HEADERS



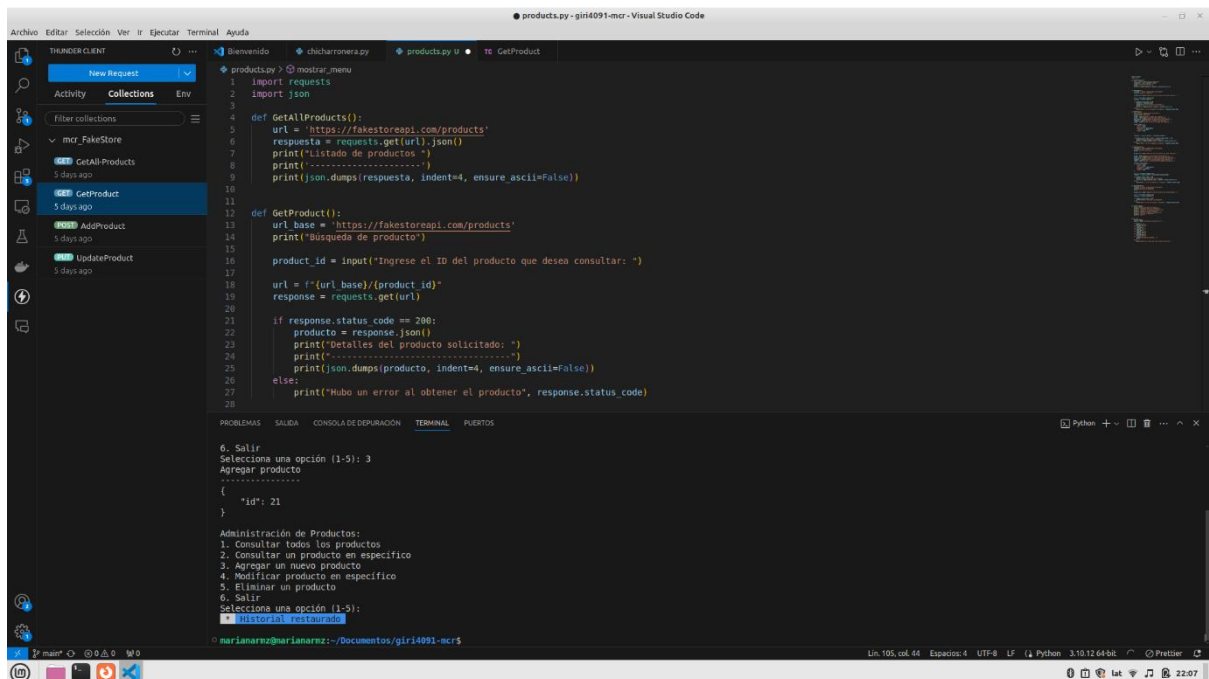
Thunder Client - Visual Studio Code

GET <https://fakestoreapi.com/products/1> Send

Status: 200 OK Size: 354 Bytes Time: 883 ms

Header	Value
date	Fri, 05 Jul 2024 02:23:30 GMT
content-type	application/json; charset=utf-8
transfer-encoding	chunked
connection	close
access-control-allow-origin	*
etag	W/"16cMMa8qV8NduTiefLdgtBe9unY"
x-powered-by	Express
cf-cache-status	DYNAMIC
report-to	[{"endpoints": [{"url": "https://va.net.cloudflare.com/reportV4?n=1d68Cw6dVuGSKC2d8H4uJCQmrv67gRdLhCPLZu3HJkR6PdvE5Teweh2JOC0N2PH50SGWldge8K3598WVwK238Vxv8Pp326dLqUWvYwTtc58HEITP2AwNw3a3m31_group", "en et", "max_age": 604800}], "max_age": 604800}]
nel	[{"success_fraction": 0, "report_to": "cf-nel", "max_age": 604800}]
server	cloudflare
cf-ray	893e4b36a6ec4a-DFW
content-encoding	br
alt-svc	h3="443"; ma=86400

CÓDIGO EN PYTHON



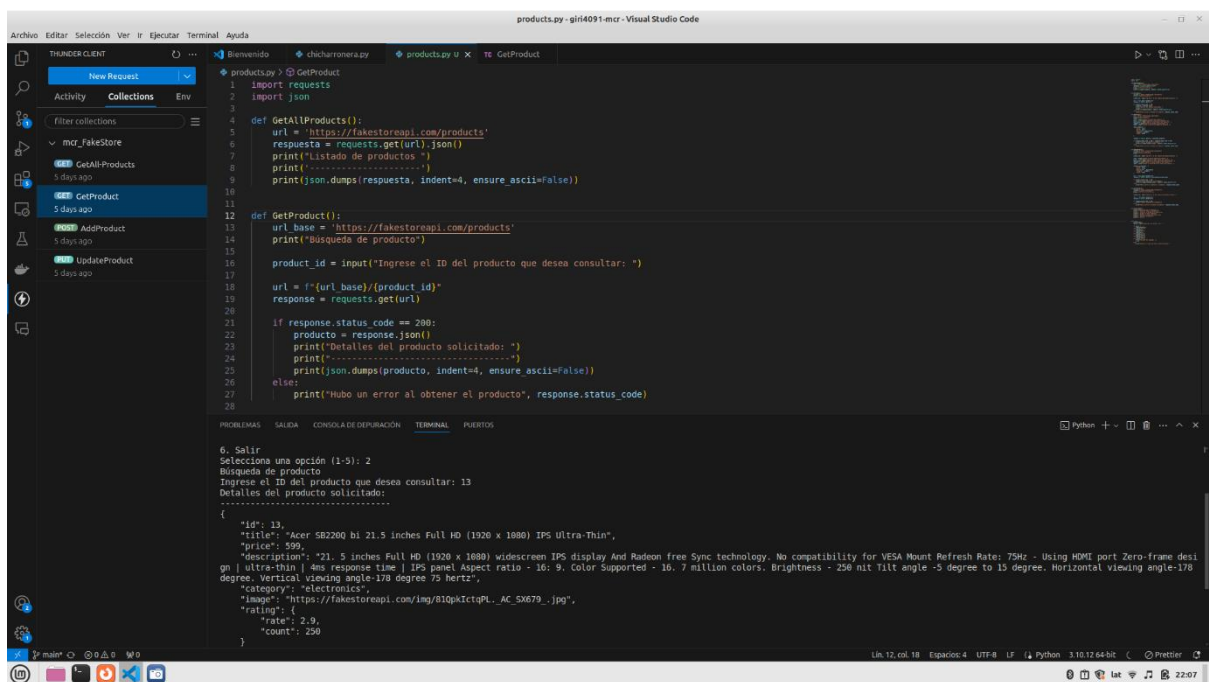
products.py - gir4091-mcr - Visual Studio Code

```
1 import requests
2 import json
3
4 def GetAllProducts():
5     url = 'https://fakestoreapi.com/products'
6     respuesta = requests.get(url).json()
7     print("Listado de productos")
8     print(json.dumps(respuesta, indent=4, ensure_ascii=False))
9
10
11
12
13 def GetProduct():
14     url_base = 'https://fakestoreapi.com/products'
15     print("Búsqueda de producto")
16
17     product_id = input("Ingrese el ID del producto que desea consultar: ")
18     url = f'{url_base}/{product_id}'
19     response = requests.get(url)
20
21     if response.status_code == 200:
22         producto = response.json()
23         print("Detalles del producto solicitado:")
24         print(json.dumps(producto, indent=4, ensure_ascii=False))
25     else:
26         print("Hubo un error al obtener el producto", response.status_code)
27
28
```

6. Salir
Selecciona una opción (1-5): 3
Agregar producto
{
 "id": 21
}

Administración de Productos:
1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir
Selecciona una opción (1-5):
* historial restaurado

SALIDA CON PYTHON



```
products.py - gir14091-mcr - Visual Studio Code

1 import requests
2 import json
3
4 def GetAllProducts():
5     url = 'https://fakestoreapi.com/products'
6     respuesta = requests.get(url).json()
7     print('Listado de productos:')
8     print(json.dumps(respuesta, indent=4, ensure_ascii=False))
9
10
11
12 def GetProduct():
13     url_base = 'https://fakestoreapi.com/products'
14     print('Busqueda de producto')
15
16     product_id = input('Ingrese el ID del producto que desea consultar: ')
17
18     url = f'{url_base}/{product_id}'
19     response = requests.get(url)
20
21     if response.status_code == 200:
22         producto = response.json()
23         print('Detalles del producto solicitado:')
24         print(json.dumps(producto, indent=4, ensure_ascii=False))
25     else:
26         print('Hubo un error al obtener el producto', response.status_code)
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

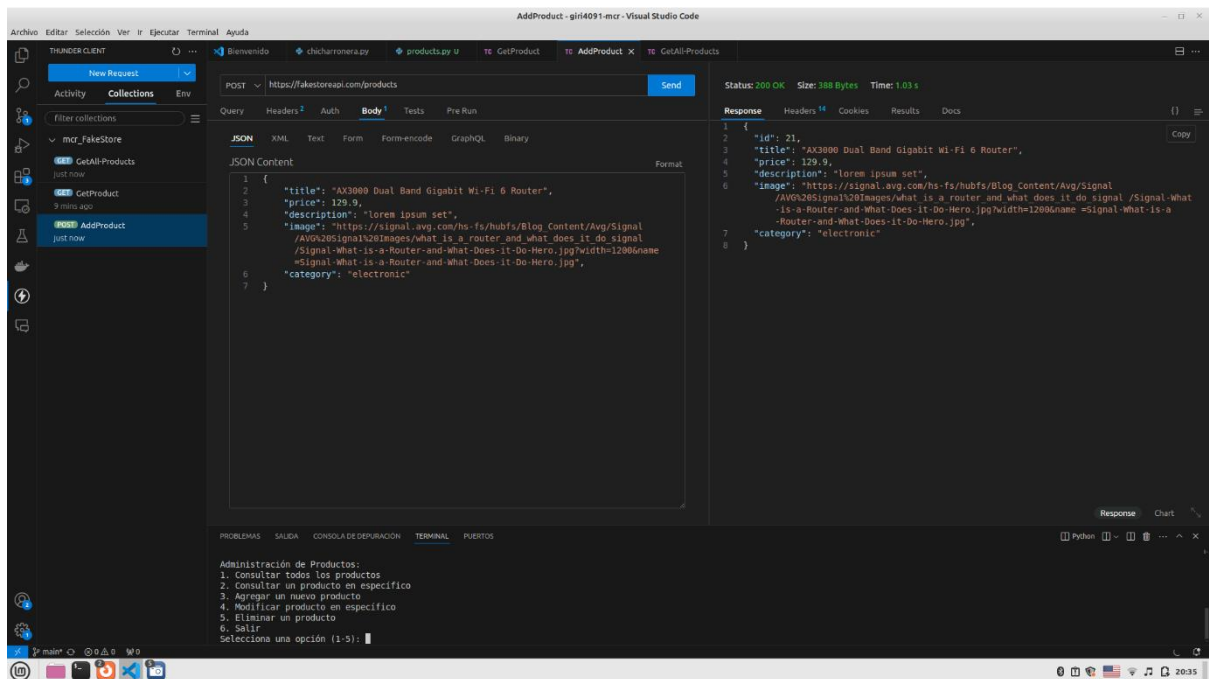
6. Salir
Selecciona una opción (1-5): 2
Busqueda de producto
Ingrese el ID del producto que desea consultar: 13
Detalles del producto solicitado:

{
 "id": 13,
 "title": "Acer SB2200 bi 21.5 inches Full HD (1920 x 1080) IPS Ultra-Thin",
 "price": 599,
 "description": "21.5 inches Full HD (1920 x 1080) widescreen IPS display And Radeon free Sync technology. No compatibility for VESA Mount Refresh Rate: 75Hz - Using HDMI port Zero-frame design | ultra-thin | 4ms response time | IPS panel Aspect ratio - 16: 9. Color Supported - 16. 7 million colors. Brightness - 250 nit Tilt angle - 5 degree to 15 degree. Horizontal viewing angle-178 degree. Vertical viewing angle-178 degree 75 hertz",
 "image": "https://fakestoreapi.com/img/81opkItqPL_AC_SxM79_.jpg",
 "rating": {
 "rate": 2.9,
 "count": 250
 }
}

ADD PRODUCT

Crear una nueva petición para agregar un nuevo producto.

PETICIÓN EN THUNDER CLIENT



```
POST https://fakestoreapi.com/products

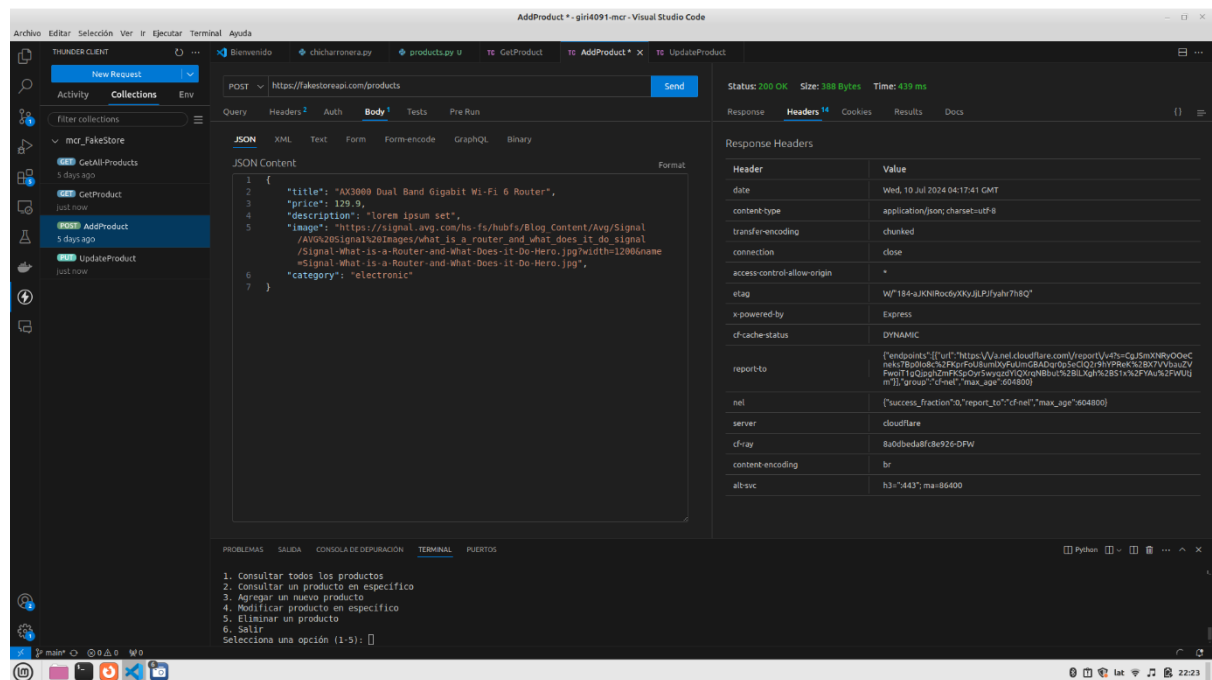
{
  "title": "AX3000 Dual Band Gigabit Wi-Fi 6 Router",
  "price": 129.9,
  "description": "Lorem ipsum set",
  "image": "https://signal.avg.com/hs-fs/hubs/Blog Content/Avg/Signal/AVG26Signal126Images/What is a router and what does it do signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1280&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
  "category": "electronic"
}
```

Status: 200 OK Size: 388 Bytes Time: 1.03s

```
{
  "id": 21,
  "title": "AX3000 Dual Band Gigabit Wi-Fi 6 Router",
  "price": 129.9,
  "description": "Lorem ipsum set",
  "image": "https://signal.avg.com/hs-fs/hubs/Blog Content/Avg/Signal/AVG26Signal126Images/What is a router and what does it do signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1280&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
  "category": "electronic"
}
```

Administración de Productos:
1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir
Selecciona una opción (1-5):

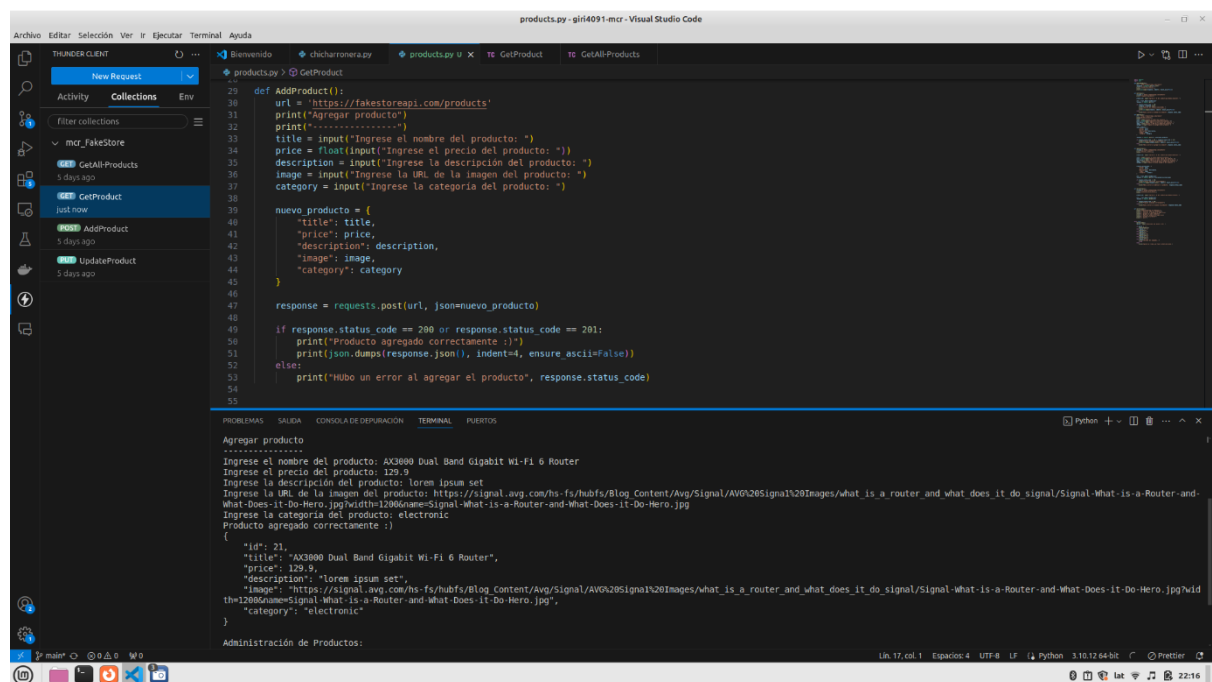
HEADERS



Thunder Client interface showing a REST client request to `https://fakestoreapi.com/products`. The response status is **200 OK**, size is **388 Bytes**, and time is **439 ms**. The response headers are displayed in a table.

Header	Value
date	Wed, 10 Jul 2024 04:17:41 GMT
content-type	application/json; charset=utf-8
transfer-encoding	chunked
connection	close
access-control-allow-origin	*
etag	W/"184-aJXNfRocky9KyJLPJ/yah7h8Q"
x-powered-by	Express
cf-cache-status	DYNAMIC
report-to	[{"endpoints": [{"url": "https://va.net.cloudflare.com/report/v4?r=CgJSmXNfRocky9KyJLPJ/yah7h8Q"}], "group": "cf-met", "max_age": 604800}]
nel	[{"success_fraction": 0, "report_to": "cf-nel", "max_age": 604800}]
server	cloudflare
cf-ray	8u0bde8f8de92c-dfw
content-encoding	br
alt-svc	h3="443"; ma=86400

CÓDIGO EN PYTHON



products.py - gir4091-mcr - Visual Studio Code

```
def AddProduct():
    url = "https://fakestoreapi.com/products"
    print("Agregar producto")
    print("-----")
    title = input("Ingrese el nombre del producto: ")
    price = float(input("Ingrese el precio del producto: "))
    description = input("Ingrese la descripción del producto: ")
    image = input("Ingrese la URL de la imagen del producto: ")
    category = input("Ingrese la categoría del producto: ")

    nuevo_producto = {
        "title": title,
        "price": price,
        "description": description,
        "image": image,
        "category": category
    }

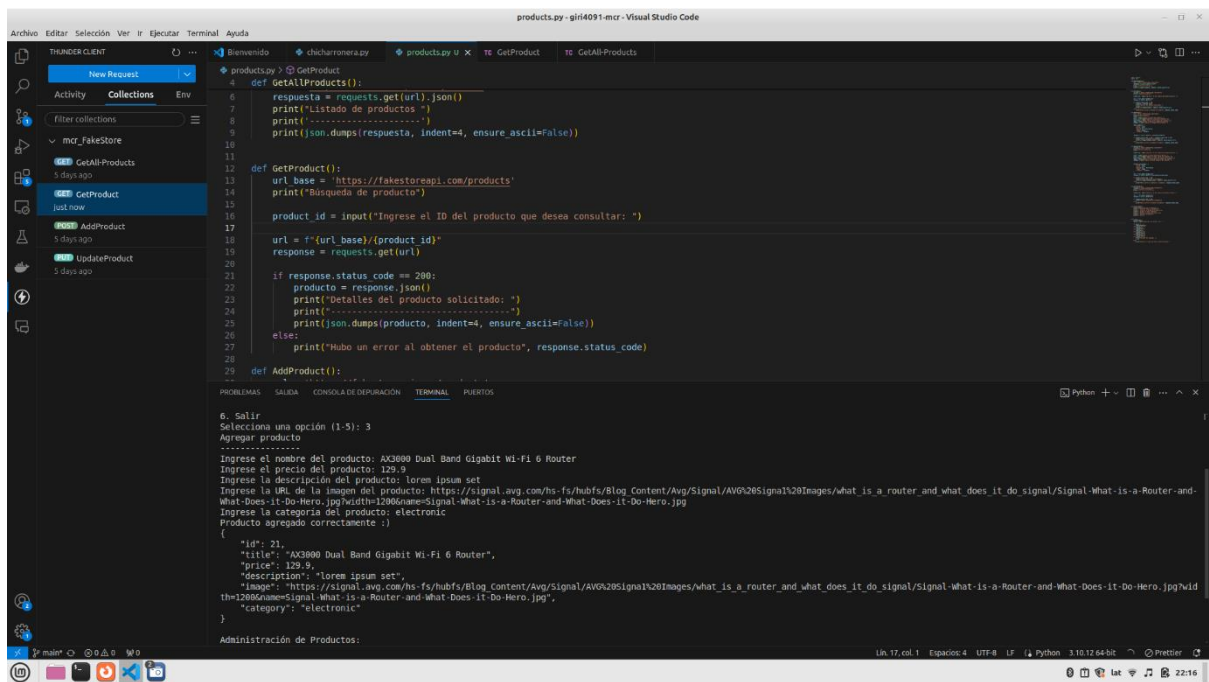
    response = requests.post(url, json=nuevo_producto)

    if response.status_code == 200 or response.status_code == 201:
        print("Producto agregado correctamente :)")
        print(json.dumps(response.json(), indent=4, ensure_ascii=False))
    else:
        print("Hubo un error al agregar el producto", response.status_code)
```

Terminal output:

```
Agregar producto
-----
Ingrese el nombre del producto: AC090 Dual Band Gigabit Wi-Fi 6 Router
Ingrese el precio del producto: 129.9
Ingrese la descripción del producto: https://signal.avg.com/hs-fs/hubfs/Blog Content/Avg/Signal/AVG26Signal20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg
Ingrese la categoría del producto: electronic
Producto agregado correctamente :)
{
  "id": 21,
  "title": "AC090 Dual Band Gigabit Wi-Fi 6 Router",
  "price": 129.9,
  "description": "lorem ipsum set",
  "image": "https://signal.avg.com/hs-fs/hubfs/Blog Content/Avg/Signal/AVG26Signal20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
  "category": "electronic"
}
```


SALIDA CON PYTHON



The screenshot shows a Visual Studio Code window with a Python script named `products.py` and its terminal output. The script defines three functions: `getProducts()`, `GetProduct()`, and `AddProduct()`. The terminal output shows the execution of the `AddProduct()` function, which prompts the user to enter product details and returns a JSON response.

```
products.py - giri4091-mcr - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

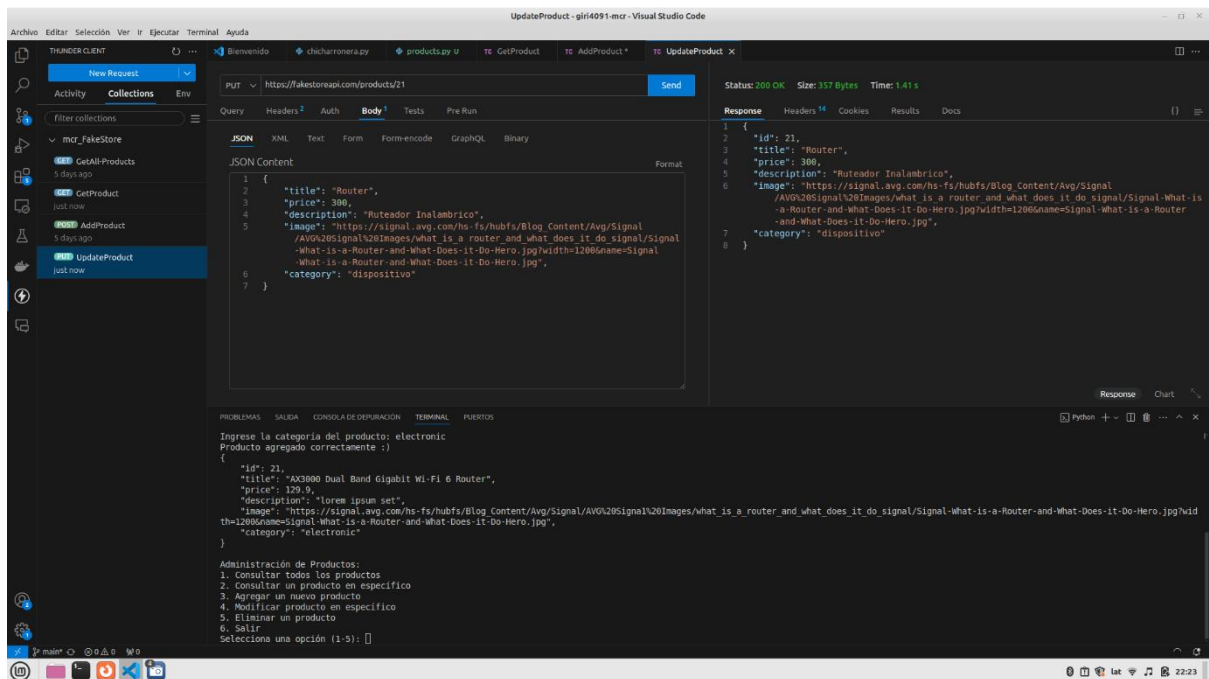
THUNDER CLIENT
New Request
Activity Collections Env
filter collections
mcr_fakeStore
GetAll-Products 5 days ago
GetProduct just now
AddProduct 5 days ago
UpdateProduct 5 days ago

products.py > def GetProduct():
4 def getProducts():
6     respuesta = requests.get(url).json()
7     print("Listado de productos ")
8     print(json.dumps(respuesta, indent=4, ensure_ascii=False))
9
10
11
12
13 def GetProduct():
14     url_base = 'https://fakestoreapi.com/products'
15     print("Búsqueda de producto")
16
17     product_id = input("Ingrese el ID del producto que desea consultar: ")
18
19     url = f'{url_base}/{product_id}'
20     response = requests.get(url)
21
22     if response.status_code == 200:
23         producto = response.json()
24         print("Detalles del producto solicitado: ")
25         print(json.dumps(producto, indent=4, ensure_ascii=False))
26     else:
27         print("Hubo un error al obtener el producto", response.status_code)
28
29
30 def AddProduct():
31
32     6. Salir
33     Seleccione una opción (1-5): 3
34     Agregar producto
35     -----
36     Ingrese el nombre del producto: AX3000 Dual Band Gigabit Wi-Fi 6 Router
37     Ingrese el precio del producto: 129.9
38     Ingrese la descripción del producto: lorem ipsum set
39     Ingrese la URL de la imagen del producto: https://signal.avg.com/hs-fs/hubfs/Blog_Content/Avg/Signal/AVG20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg
40     Ingrese la categoría del producto: electronic
41     Producto agregado correctamente :)
42     {
43       "id": 21,
44       "title": "AX3000 Dual Band Gigabit Wi-Fi 6 Router",
45       "price": 129.9,
46       "description": "lorem ipsum set",
47       "image": "https://signal.avg.com/hs-fs/hubfs/Blog_Content/Avg/Signal/AVG20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
48       "category": "electronic"
49     }
50
51 Administración de Productos:
52 1. Consultar todos los productos
53 2. Consultar un producto en específico
54 3. Agregar un nuevo producto
55 4. Modificar producto en específico
56 5. Eliminar un producto
57 6. Salir
58 Seleccione una opción (1-5):
```

UPDATE PRODUCT

Modificando un nuevo producto.

PETICIÓN CON THUNDER CLIENT



The screenshot shows a Visual Studio Code window with a Thunder Client request and response for updating a product. The request is a PUT request to `https://fakestoreapi.com/products/21` with a JSON body. The response is a 200 OK status with a JSON body containing the updated product details.

```
UpdateProduct - giri4091-mcr - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

THUNDER CLIENT
New Request
Activity Collections Env
filter collections
mcr_fakeStore
GetAll-Products 5 days ago
GetProduct just now
AddProduct 5 days ago
UpdateProduct just now

PUT https://fakestoreapi.com/products/21 Send
Query Headers Auth Body Tests Pre Run
JSON Content
JSON Content
1 {
2   "title": "Router",
3   "price": 300,
4   "description": "Ruteador inalámbrico",
5   "image": "https://signal.avg.com/hs-fs/hubfs/Blog_Content/Avg/Signal/AVG20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
6   "category": "dispositivo"
7 }
8

Status: 200 OK Size: 357 Bytes Time: 1.41 s
Response Headers Cookies Results Docs
1 {
2   "id": 21,
3   "title": "Router",
4   "price": 300,
5   "description": "Ruteador inalámbrico",
6   "image": "https://signal.avg.com/hs-fs/hubfs/Blog_Content/Avg/Signal/AVG20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
7   "category": "dispositivo"
8 }

Ingrese la categoría del producto: electronic
Producto agregado correctamente :)
{
  "id": 21,
  "title": "AX3000 Dual Band Gigabit Wi-Fi 6 Router",
  "price": 129.9,
  "description": "lorem ipsum set",
  "image": "https://signal.avg.com/hs-fs/hubfs/Blog_Content/Avg/Signal/AVG20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-It-Do-Hero.jpg",
  "category": "electronic"
}

Administración de Productos:
1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir
Seleccione una opción (1-5):
```

HEADERS

The screenshot shows the Thunder Client interface within Visual Studio Code. A PUT request is being sent to the URL `https://fakestoreapi.com/products/21`. The request body is a JSON object representing a product:

```
1 {
2   "title": "Router",
3   "price": 380,
4   "description": "Ruteador inalámbrico",
5   "image": "https://signal.avg.com/hs-fs/hubfs/Blog Content/Avq/Signal/AVG205signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-what-is-a-Router-and-what-Does-it-Do-Hero.jpg?width=1200&name=Signal-what-is-a-Router-and-what-Does-it-Do-Hero.jpg",
6   "category": "dispositivo"
7 }
```

The response status is 200 OK, with a size of 357 Bytes and a time of 1.41 s. The response headers are listed on the right:

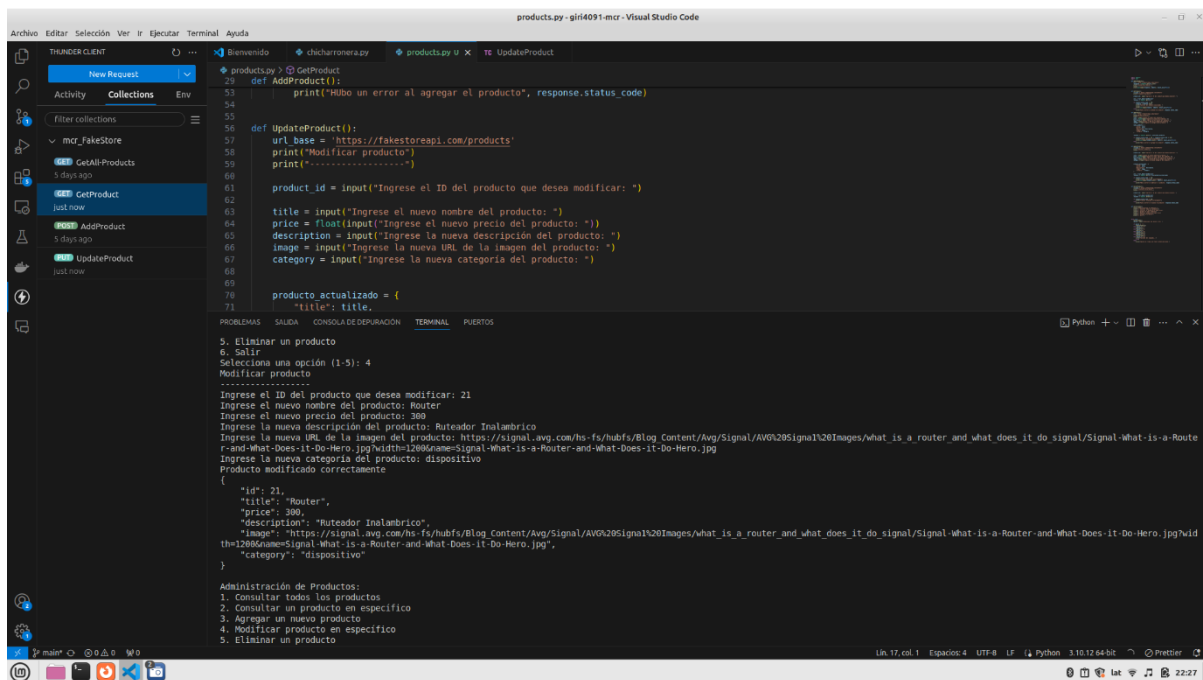
Header	Value
date	Wed, 10 Jul 2024 04:22:47 GMT
content-type	application/json; charset=utf-8
transfer-encoding	chunked
connection	close
access-control-allow-origin	*
etag	W/"165-a2ZKJSM8NQjiva3Tsl1TDQU"
x-powered-by	Express
cf-cache-status	DYNAMIC
report-to	[{"endpoints":[{"url":"https://a.net.cloudflare.com/r/eport/Vv47n6KO9wdhTcNkuw0Dy9r2f8k-CwQoR6GCHC867r0bUCT7EqZ8V3cm30Uf8-499WwS7FahM4bK0C2P4852qphE-o4Ism52ZpCtLV49T04Ww4sk1ERh0wae575b075ahE"}],"group":"cf-net","max_age":604800}]
nel	[{"success_fraction":0,"report_to":"cf-nel","max_age":604800}]
server	cloudflare
cf-ray	8a0d64e7ae52d39-DFW
content-encoding	br
alt-svc	h3="443"; ma=86400

CÓDIGO EN PYTHON

The screenshot shows the Python code for the REST client in the `products.py` file. The code includes functions for adding, updating, and deleting products:

```
1 def AddProduct():
2     print("Hubo un error al agregar el producto", response.status_code)
3
4 def UpdateProduct():
5     url_base = "https://fakestoreapi.com/products"
6     print("Modificar producto")
7     print("-----")
8     product_id = input("Ingrese el ID del producto que desea modificar: ")
9
10    title = input("Ingrese el nuevo nombre del producto: ")
11    price = float(input("Ingrese el nuevo precio del producto: "))
12    description = input("Ingrese la nueva descripción del producto: ")
13    image = input("Ingrese la nueva URL de la imagen del producto: ")
14    category = input("Ingrese la nueva categoría del producto: ")
15
16    producto_actualizado = {
17        "title": title,
18        "price": price,
19        "description": description,
20        "image": image,
21        "category": category
22    }
23
24    url = f"{url_base}/{product_id}"
25    response = requests.put(url, json=producto_actualizado)
26
27    if response.status_code == 200:
28        print("Producto modificado correctamente")
29        print(json.dumps(response.json(), indent=4, ensure_ascii=False))
30    else:
31        print("Hubo un error al modificar el producto", response.status_code)
32
33 def DeleteProduct():
34     url_base = "https://fakestoreapi.com/products"
35     print("Eliminación de producto")
```

SALIDA CON PYTHON



The screenshot shows a Visual Studio Code window with a Python script named `products.py` and its terminal output. The script defines functions for adding, updating, and deleting products. The terminal output shows the execution of the script, including prompts for product details and the resulting JSON response for a product update.

```
products.py - giri4091-mcr - Visual Studio Code

def AddProduct():
    print("Hubo un error al agregar el producto", response.status_code)

def UpdateProduct():
    url_base = 'https://fakestoreapi.com/products'
    print("-----")
    product_id = input("Ingrese el ID del producto que desea modificar: ")
    title = input("Ingrese el nuevo nombre del producto: ")
    price = float(input("Ingrese el nuevo precio del producto: "))
    description = input("Ingrese la nueva descripción del producto: ")
    image = input("Ingrese la nueva URL de la imagen del producto: ")
    category = input("Ingrese la nueva categoría del producto: ")

    producto_actualizado = {
        "title": title,
        "price": price,
        "description": description,
        "image": image,
        "category": category
    }

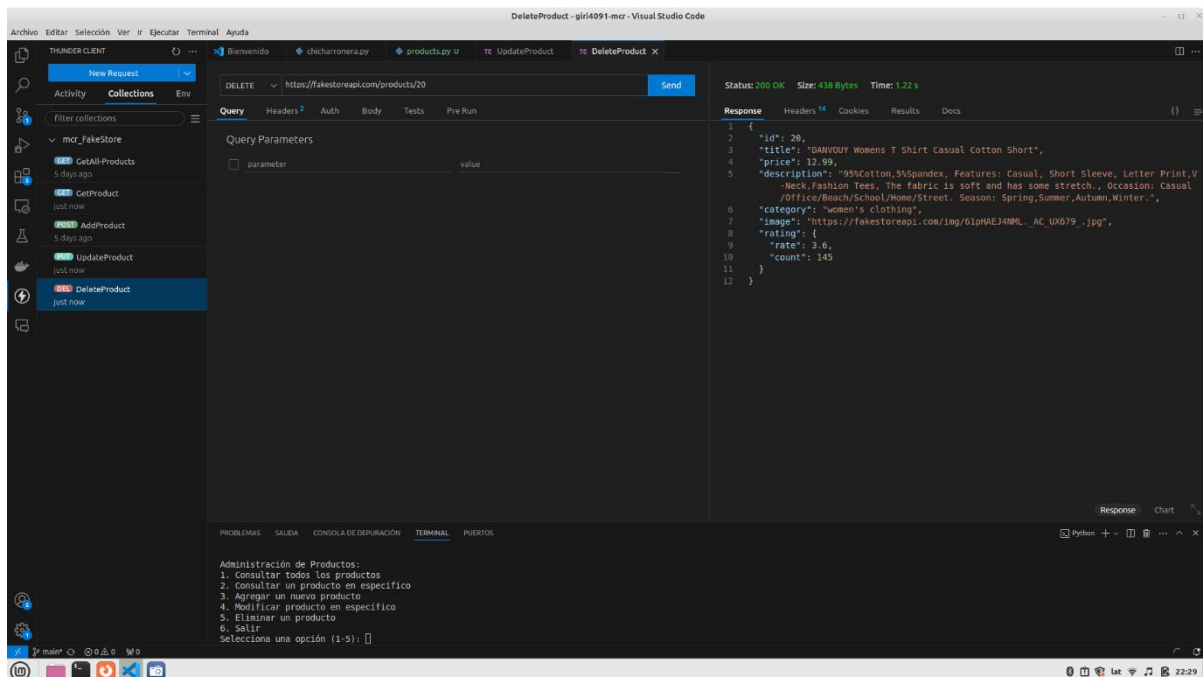
    # Eliminar un producto
    # Salir
    # Selecciona una opción (1-5): 4
    # Modificar producto
    # Ingrese el ID del producto que desea modificar: 21
    # Ingrese el nuevo nombre del producto: Router
    # Ingrese el nuevo precio del producto: 300
    # Ingrese la nueva descripción del producto: Router Inalambrico
    # Ingrese la nueva URL de la imagen del producto: https://signal.avg.com/hs-fs/hubfs/Blog Content/Avg/Signal/AVG%20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-it-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-it-Do-Hero.jpg
    # Ingrese la nueva categoría del producto: dispositivo
    # Producto modificado correctamente
    {
      "id": 21,
      "title": "Router",
      "price": 300,
      "description": "Router Inalambrico",
      "image": "https://signal.avg.com/hs-fs/hubfs/Blog Content/Avg/Signal/AVG%20Signal%20Images/what_is_a_router_and_what_does_it_do_signal/Signal-What-is-a-Router-and-What-Does-it-Do-Hero.jpg?width=1200&name=Signal-What-is-a-Router-and-What-Does-it-Do-Hero.jpg",
      "category": "dispositivo"
    }

    # Administración de Productos:
    # 1. Consultar todos los productos
    # 2. Consultar un producto en específico
    # 3. Agregar un nuevo producto
    # 4. Modificar producto en específico
    # 5. Eliminar un producto
    # 6. Salir
    # Selecciona una opción (1-5):
```

DELETE PRODUCT

Eliminando un producto.

PETICIÓN CON THUNDER CLIENT



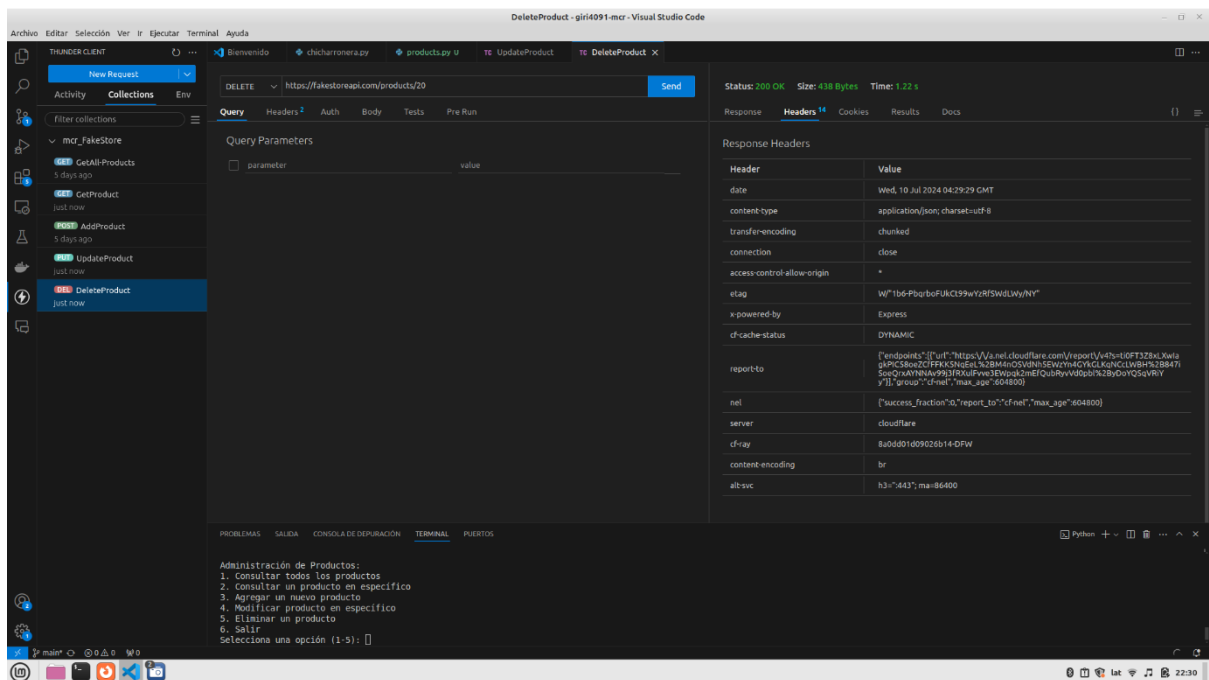
The screenshot shows the Thunder Client interface with a DELETE request to `https://fakestoreapi.com/products/20`. The request is successful, returning a 200 OK status. The response body is a JSON object representing the deleted product.

```
DELETE https://fakestoreapi.com/products/20

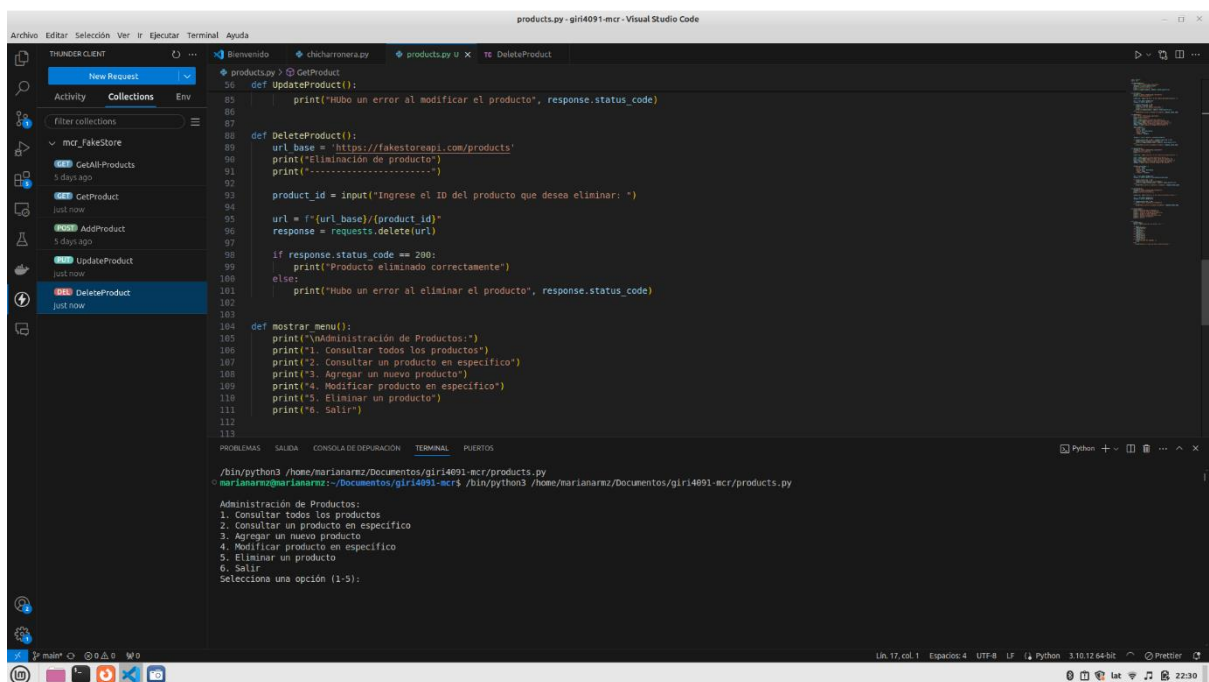
Query Parameters
parameter value

Response
1 {
2   "id": 20,
3   "title": "DANVOUY Womens T Shirt Casual Cotton Short",
4   "price": 12.99,
5   "description": "95%Cotton,5%Spandex, Features: Casual, Short Sleeve, Letter Print,V-Neck,Fashion Tees, The fabric is soft and has some stretch., Occasion: Casual
6   /Office/Beach/School/Home/Street. Season: Spring,Summer,Autumn,Winter.",
7   "category": "women's clothing",
8   "image": "https://fakestoreapi.com/img/61pAEJ4NML_AC_UX679_.jpg",
9   "rating": {
10    "rate": 3.6,
11    "count": 145
12  }
```

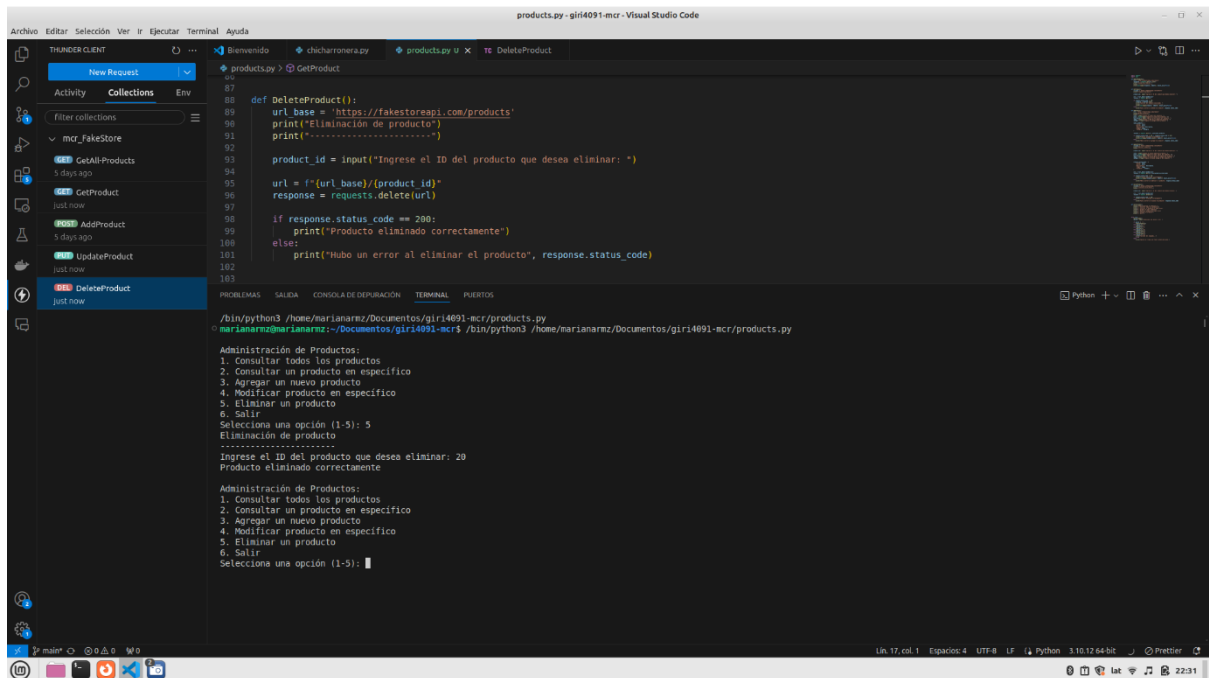
HEADERS



CÓDIGO EN PYTHON



SALIDA CON PYTHON



The screenshot displays the Visual Studio Code interface with a Python script named `products.py` and its execution output in the terminal.

Python Script (`products.py`):

```
def DeleteProduct():
    url_base = 'https://fakestoreapi.com/products'
    print("Eliminación de producto")
    print("-----")
    product_id = input("Ingrese el ID del producto que desea eliminar: ")
    url = f"{url_base}/{product_id}"
    response = requests.delete(url)
    if response.status_code == 200:
        print("Producto eliminado correctamente")
    else:
        print("Hubo un error al eliminar el producto", response.status_code)
```

Terminal Output:

```
~/bin/python3 /home/marianamz/Documentos/giri4091-mcr/products.py
marianamz@marianamz:~/Documentos/giri4091-mcr$ ./bin/python3 /home/marianamz/Documentos/giri4091-mcr/products.py

Administración de Productos:
1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir
Seleccione una opción (1-5): 5
Eliminación de producto
-----
Ingrese el ID del producto que desea eliminar: 20
Producto eliminado correctamente

Administración de Productos:
1. Consultar todos los productos
2. Consultar un producto en específico
3. Agregar un nuevo producto
4. Modificar producto en específico
5. Eliminar un producto
6. Salir
Seleccione una opción (1-5):
```

The interface also shows a sidebar with a collection named `mcr_fakeStore` containing several API requests, including `DeleteProduct` which is currently selected.