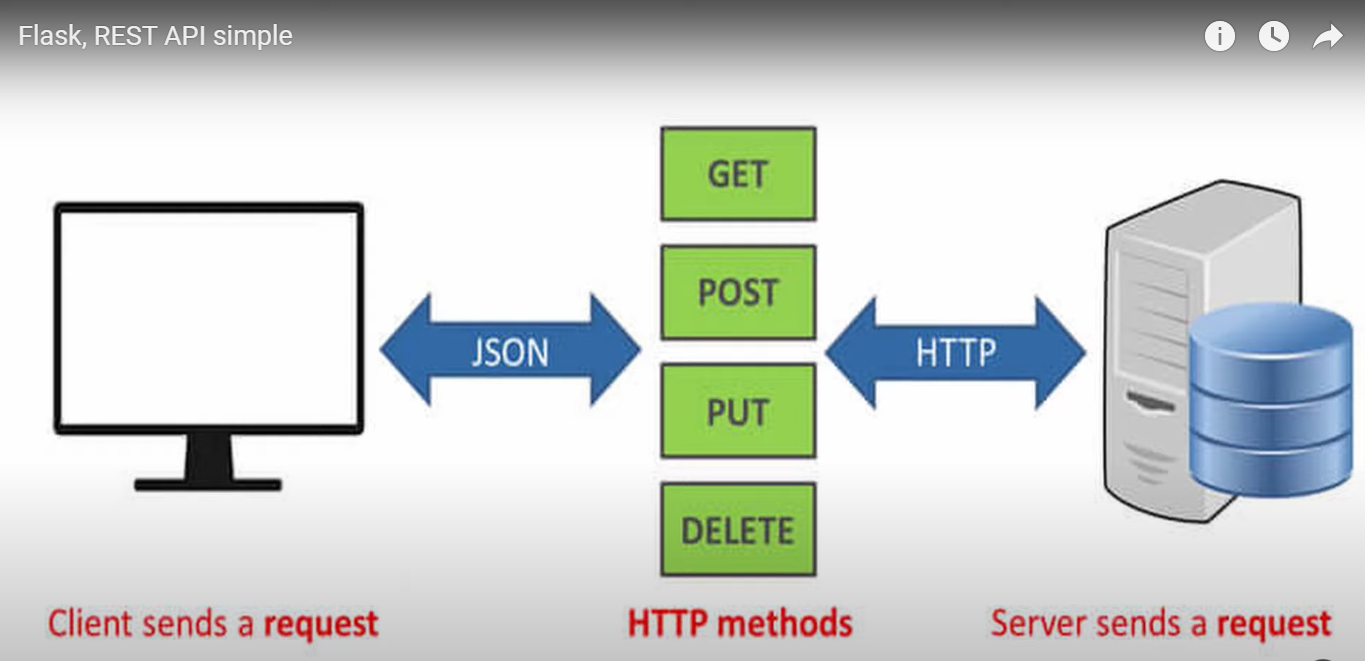
CRUD ( CREATE , READ, UPDATE, DELETE) de python con el framework Flask



Crear un carpeta en el escritorio CrudPython

Abrir la carpeta desde Visual Studio Code

Crear un archivo app.py y otro productos.py y agregarle:

productos=[

{"nombre":"laptop", "precio":150000, "stock":15},

{"nombre":"monitor", "precio":12000, "stock":10},

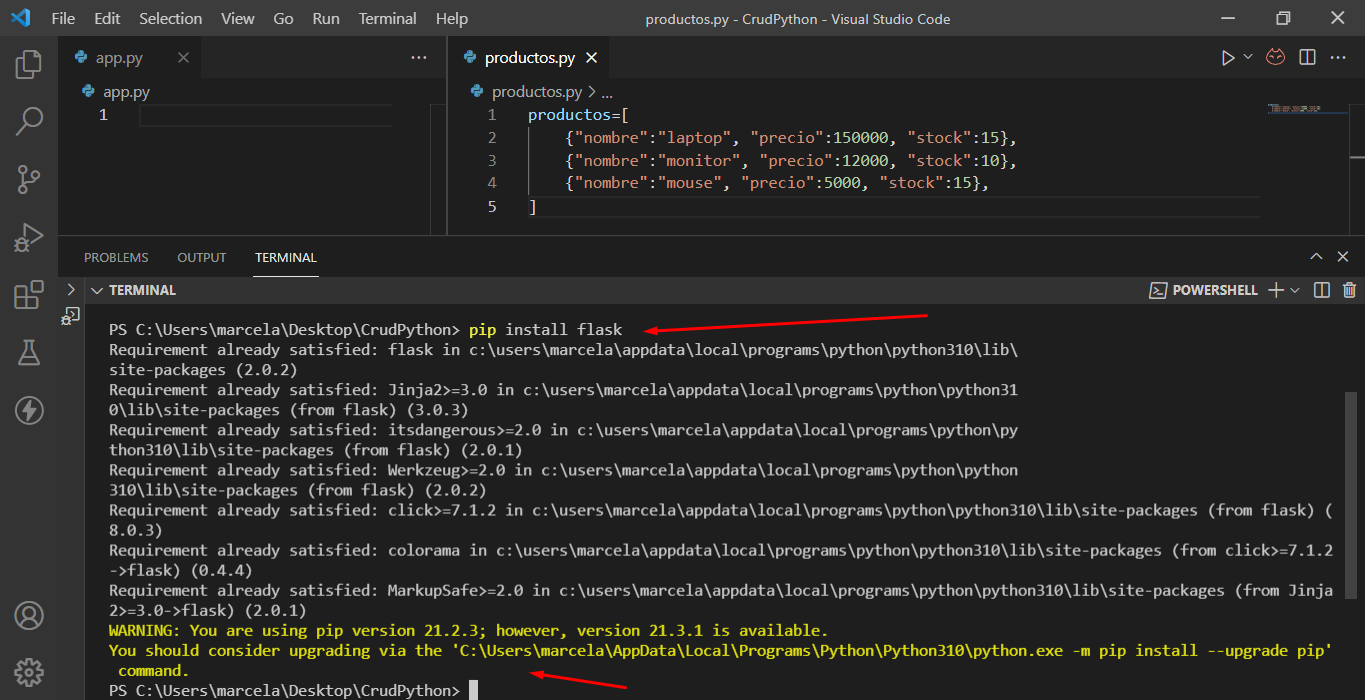
{"nombre":"mouse", "precio":5000, "stock":5},

]

Vamos instalar Flask

Ir al menú de VSC y crear una terminal y escribir:

* para windows: pip install flask
* para mac: pip3 install flask
* para linux sudo apt install python3-pip



En app.py agregar:

from flask import Flask #importo el modulo Flask del archivo flask

app=Flask(\_\_name\_\_) # inicio Flask y creo un objeto 'app' de Flask

# le paso como parametro la propiedad \_\_name\_\_

# app es mi aplicacion de servidor, va a ser mi servidor

# programa principal

if \_\_name\_\_=='\_\_main\_\_': # si el nombre del archivo es esta ejecutando como archivo principal

app.run(debug=True, port=5000)

# inicio el servidor app como parametro debug,

# para que si hay cambios

# en el fuente se reinicie la aplicacion

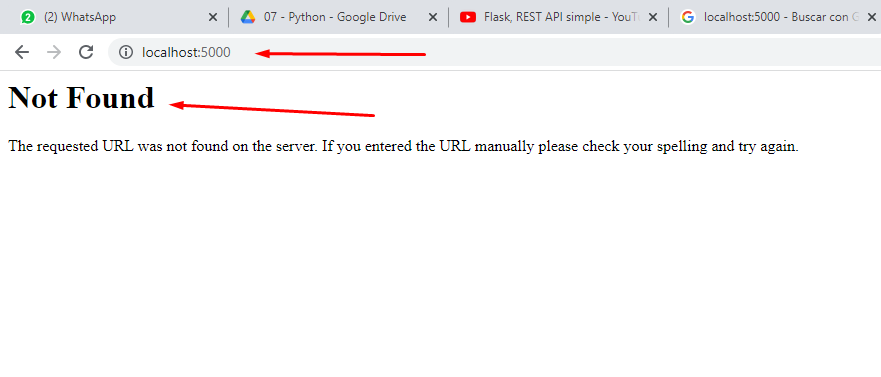
# y le paso el puerto donde se va a ejecutar nuestro servidor

# ( pueden utilizar otro puerto)

Luego limpiar la consola y ejecutar en la terminal en windows usar python app.py, en mac o linux : python3 app.py

Probar en el navegador localhost:5000

Va a dar error de server 'Not Found' pero está bien porque no creamos ninguna ruta todavía



Empezamos a crear las rutas

from flask import Flask

app=Flask(\_\_name\_\_)

@app.route('/ping')

def ping():

return 'pong'

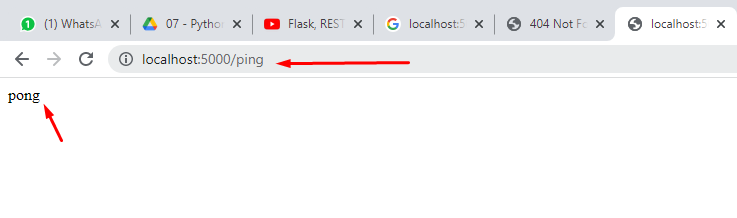
#programa principal

if \_\_name\_\_=="\_\_main\_\_":

app.run(debug=True, port=5000)

ahora pruebo en el navegador

localhost:5000/ping



me tiene que devolver ‘pong’

si agrego la ruta /hola , después de creado el objeto app y antes del programa principal

@app.route('/hola')

def hola():

return '<h1>Bienvenidos a Flask </h1>'

y pruebo en el navegador localhost:5000/hola

Si da error, parar el proceso con Ctrl+C, o borrar la terminal (si tienen mas de 1 borrar todas) y volver a correr : python app.py

* instalar en la terminal

pip install -U flask-cors

Para crear la ruta de productos importo jsonify, productos y CORS

Y después de crear el objeto app llamo a Cors(app)

from flask import Flask,jsonify # agrego import jsoninffy

from productos import productos # importo la lista de productos

from flask\_cors import CORS # agregar esta linea

app=Flask(\_\_name\_\_)

CORS(app) # agregar esta linea

# Si sigue dando error de CORS, salir del VSC y volver a entrar

agrego la ruta /productos ( en cualquier lado después de creado el objeto app y antes del programa principal

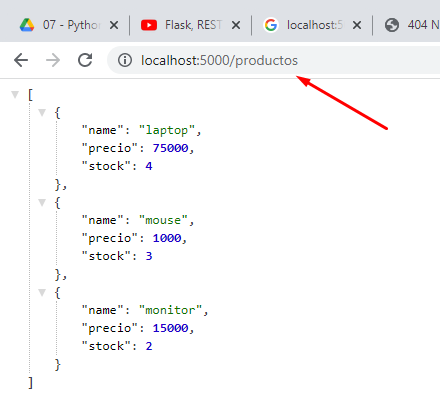
@app.route('/productos')

def getproductos():

return jsonify(productos) #tengo que importar jsonify

# tambien productos

pruebo en el navegador localhost:5000/productos



Tengo que ver los productos cargados en la ruta de productos

Ya puedo crear el frontend y armar una tabla que muestre los productos leyendo los datos del json, con bootstrap, js, fetch() y Vue

Si sigue dando error de CORS, salir del VSC y volver a entrar

OJO : Todavía no grabamos nada en la base de datos

Crear una carpeta js y un archivo main.js que contenga el objeto de Vue , que lea los productos que están en la url 'http://localhost:5000/productos'

const app=new Vue({

el:"#app",

data:{

productos:[]

},

created() {

var url='http://localhost:5000/productos'

this.fetchData(url)

},

methods:{

fetchData(url){

fetch(url)

.then(response => response.json())

.then(data => {

this.productos = data;

// this.loading = false;

})

.catch(err => {

// this.errored = true

})

}

}

})

Crear un archivo index.html que contenga los script de bootstrap 5 y Vue, la conexión al archivo main,js y muestre en una tabla los productos con Vue con algo de estilos de bootstrap

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<script src="<https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js>" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"

crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

</head>

<body>

<main>

<div class="container" id="app">

<table class="table table-striped">

<tr>

<th>Nombre</th>

<th>Precio</th>

<th>Stock</th>

<th>Acciones</th>

</tr>

<tr v-for="producto in productos">

<td>{{producto.nombre}}</td>

<td>{{producto.precio}}</td>

<td>{{producto.stock}}</td>

<td>

</td>

</tr>

</table>

</div>

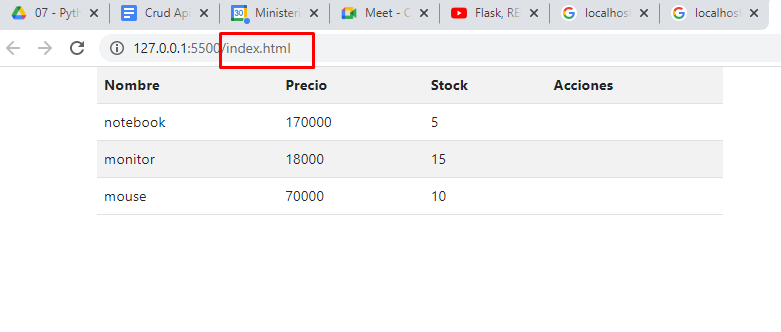
</main>

<script src="./js/main.js"></script>

</body>

</html>

ver en el navegador la página index.html



Clase 2 del CRUD

Despues agrego la ruta : /productos/nombreProducto

@app.route('/productos/<string:product\_name>')

def getproducto( product\_name):

productoEncontrado=[ producto for producto in productos if producto["nombre"]==product\_name]

if len(productoEncontrado)>0:

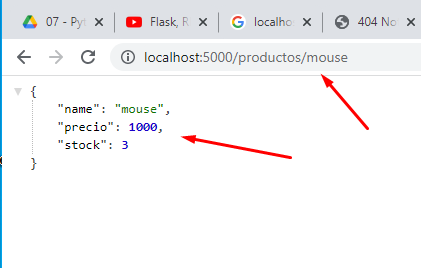
return jsonify(productoEncontrado[0])

else:

return "no encontrado"

y pruebo en el navegador

localhost:5000/productos/mouse



agregar a las importaciones de la 1ra línea a request

from flask import Flask,jsonify,request

y agregar la ruta para agregar un producto

@app.route('/productos', methods=["POST"])

def agregarProducto():

new\_producto={

"nombre":request.json['nombre'],

"precio":request.json['precio'],

"stock":request.json['stock'],

}

productos.append(new\_producto)

return jsonify({"productos": productos })

También agregar la ruta para el delete

@app.route('/productos/<string:product\_name>',methods=['DELETE'])

def deleteProducto( product\_name):

productoEncontrado=[ producto for producto in productos if producto["nombre"]==product\_name]

if len(productoEncontrado)>0:

productos.remove(productoEncontrado[0])

return jsonify({

"message":"prducto eliminado",

"productos":productos

})

return jsonify({

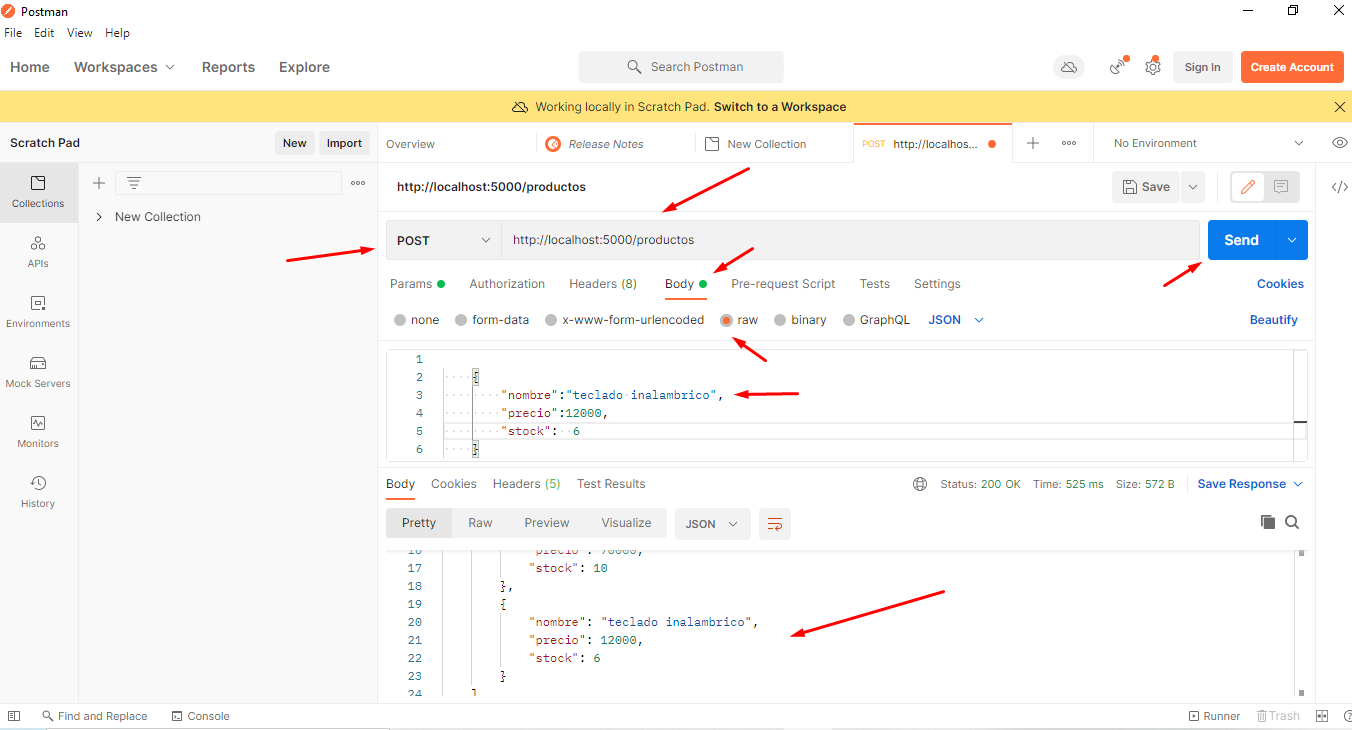
"message":"producto no encontrado " })

descargar postman de <https://www.postman.com/downloads/> y ejecutarlo

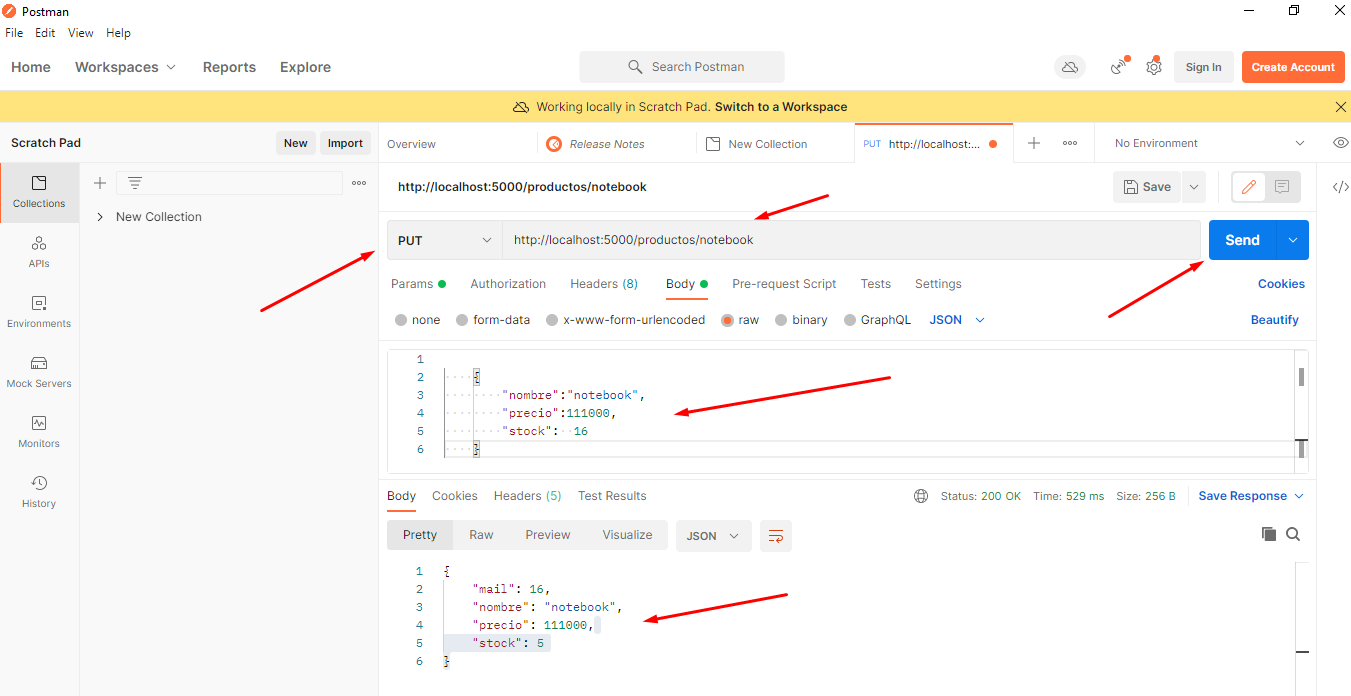
Con postman vamos a simular el cliente ( el frontend ),

Vamos a enviar peticiones al servidor( que traiga datos, que agregue, que modifique y que elimine de una ruta dada

por ejemplo: POST a la ruta:localhost:5000/usuarios , click en Body, click en RAW, crear un json con el producto que quiero agregar, presiono Send y me muestra el json con el producto agregado

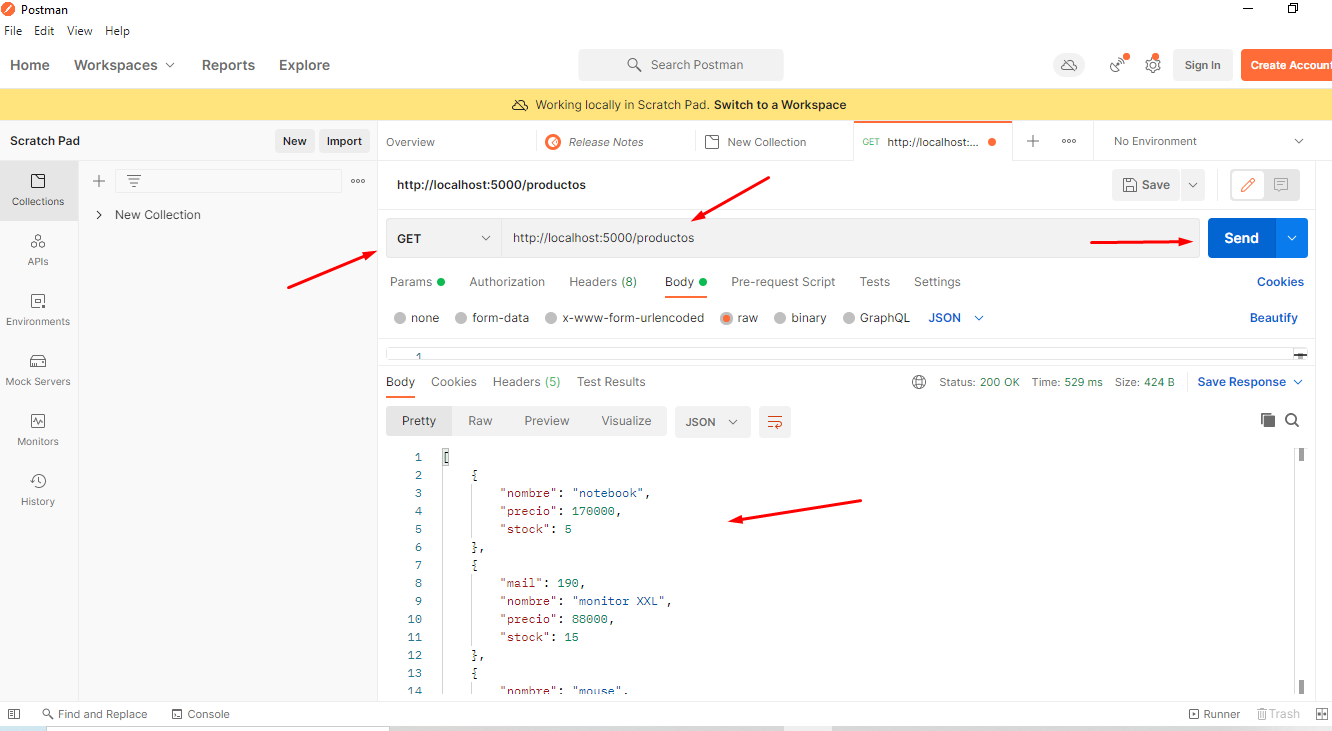


Y una modificación, PUT a la ruta localhost/productos/notebook , tilde body y Raw , que lo modifique con el json y Send

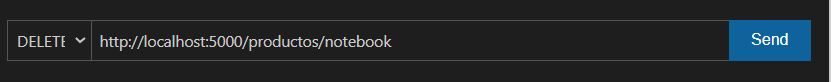


Por ejemplo:

que haga un GET a la ruta ‘localhost:5000/productos’ y presione send



Y pruebo eliminar el producto: notebook



da error no veo porque