Universidad Tecnológica de Panamá

Facultad de Sistemas Computacionales

Lic. En Ingeniería de Software

Ingeniería web

Laboratorio – API REST CON POSTMAN

Estudiantes: Jose Bustamante 8-1011-1717

Abigail Koo 8-997-974

Fecha de Ejecución: 13 de noviembre de 2025

Fecha de Entrega: 14 de noviembre de 2025

Facilitador: Irina Fong

Grupo: 1SF131

Segundo Semestre

Año: 2025

Introducción

Este documento presenta el desarrollo de una API RESTful en PHP que implementa los métodos HTTP POST, GET y PUT para la gestión de productos. La API fue desarrollada siguiendo el patrón MVC (Modelo-Vista-Controlador) y probada utilizando Postman como cliente HTTP. Se incluyen evidencias de las pruebas realizadas y el código fuente desarrollado.

Tecnologías Usadas

- **PHP** - Backend
- **MySQL** - Base de datos
- **Apache** - Servidor web
- **Postman** - Pruebas
- **PDO** - Conexión BD
- **JSON** - Formato datos
- **REST** - Arquitectura API

Metodología implementada

- **Configuración de Base de Datos**

  - Se utilizó MySQL con WAMPP para la gestión de la base de datos
  - Creación de la tabla **productos** con campos: id, nombre, precio, stock

- **Desarrollo de Endpoints**

  - **POST:** Crear nuevos productos
  - **GET:** Listar todos los productos
  - **PUT:** Actualizar productos existentes

- **Pruebas con Postman**

- Validación individual de cada método HTTP
- Verificación de códigos de respuesta HTTP
- Prueba de manejo de errores

Documentación de Funciones Principales

database.php

```php
<?php
class Database {
    private $host = "localhost";
    private $db_name = "api_db";
    private $username = "root";
    private $password = "";
    public $conn;

    public function getConnection() {
        $this->conn = null;
        try {
            $this->conn = new PDO("mysql:host=" . $this->host . ";dbname=" . $this->db_name, $this->username, $this->password);
            $this->conn->exec("set names utf8");
            $this->conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        } catch(PDOException $exception) {
            echo "Error de conexión: " . $exception->getMessage();
        }
        return $this->conn;
    }
}
?>
```

index.php

```php
<?php
// Redirigir todas las solicitudes al controlador
include_once './controllers/ProductosController.php';
?>
```

Producto.php

```php
<?php
class Producto {
    private $conn;
    private $table = "productos";

    public $id;
    public $nombre;
    public $precio;
    public $stock;

    public function __construct($db) {
        $this->conn = $db;
    }

    // Crear producto
    public function crear() {
        $query = "INSERT INTO " . $this->table . " SET nombre=:nombre, precio=:precio, stock=:stock";
        $stmt = $this->conn->prepare($query);

        $stmt->bindParam(":nombre", $this->nombre);
        $stmt->bindParam(":precio", $this->precio);
        $stmt->bindParam(":stock", $this->stock);

        if($stmt->execute()) {
            return true;
        }
        return false;
    }
```

```php
    // Obtener todos los productos
    public function listar() {
        $query = "SELECT * FROM " . $this->table;
        $stmt = $this->conn->prepare($query);
        $stmt->execute();
        return $stmt;
    }

    // Actualizar producto
    public function actualizar() {
        $query = "UPDATE " . $this->table . " SET nombre=:nombre, precio=:precio, stock=:stock WHERE
id=:id";
        $stmt = $this->conn->prepare($query);

        $stmt->bindParam(":id", $this->id);
        $stmt->bindParam(":nombre", $this->nombre);
        $stmt->bindParam(":precio", $this->precio);
        $stmt->bindParam(":stock", $this->stock);

        if($stmt->execute()) {
            return true;
        }
        return false;
    }
}
?>
```

ProductosController.php

```php
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: POST, GET, PUT");
header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

include_once '../config/database.php';
include_once '../model/Producto.php';

$database = new Database();
$db = $database->getConnection();
$producto = new Producto($db);

$method = $_SERVER['REQUEST_METHOD'];

switch($method) {
    case 'POST':
        // Crear producto
        $data = json_decode(file_get_contents("php://input"));
        $producto->nombre = $data->nombre;
        $producto->precio = $data->precio;
        $producto->stock = $data->stock;

        if($producto->crear()) {
            http_response_code(201);
            echo json_encode(array("message" => "Producto creado."));
        } else {
            http_response_code(503);
            echo json_encode(array("message" => "Error al crear producto."));
        }
        break;
```

```php
    case 'GET':
        // Listar productos
        $stmt = $producto->listar();
        $num = $stmt->rowCount();

        if($num > 0) {
            $productos_arr = array();
            while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
                extract($row);
                $producto_item = array(
                    "id" => $id,
                    "nombre" => $nombre,
                    "precio" => $precio,
                    "stock" => $stock
                );
                array_push($productos_arr, $producto_item);
            }
            http_response_code(200);
            echo json_encode($productos_arr);
        } else {
            http_response_code(404);
            echo json_encode(array("message" => "No se encontraron productos."));
        }
        break;

    case 'PUT':
        // Actualizar producto
        $data = json_decode(file_get_contents("php://input"));
        $producto->id = $data->id;
        $producto->nombre = $data->nombre;
        $producto->precio = $data->precio;
        $producto->stock = $data->stock;
```

```php
        if($producto->actualizar()) {
            http_response_code(200);
            echo json_encode(array("message" => "Producto actualizado."));
        } else {
            http_response_code(503);
            echo json_encode(array("message" => "Error al actualizar producto."));
        }
        break;

    default:
        http_response_code(405);
        echo json_encode(array("message" => "Método no permitido."));
        break;
}
?>
```

Capturas de Pantalla

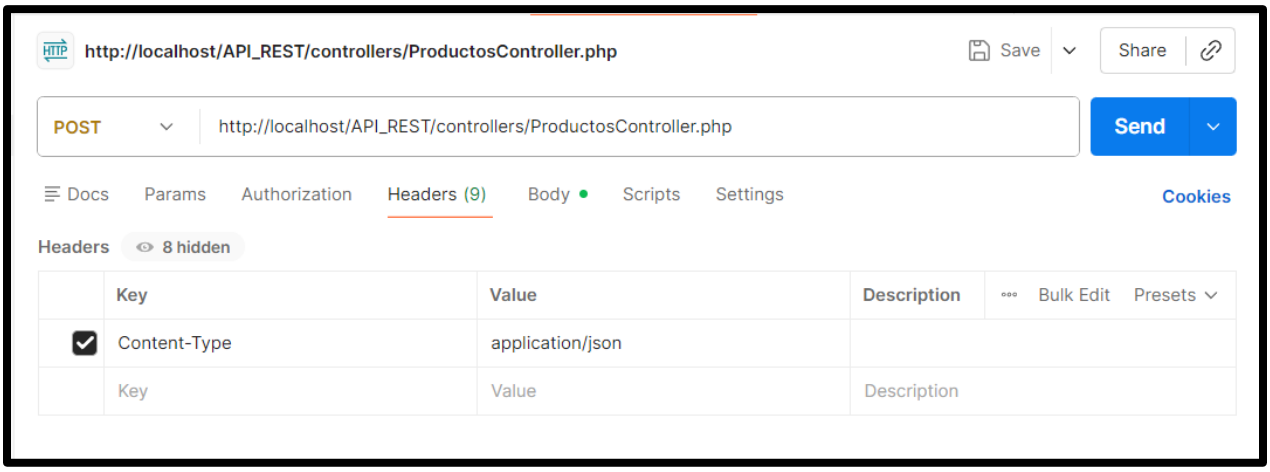

Ilustración 1 – Headers método POST en POSTMAN

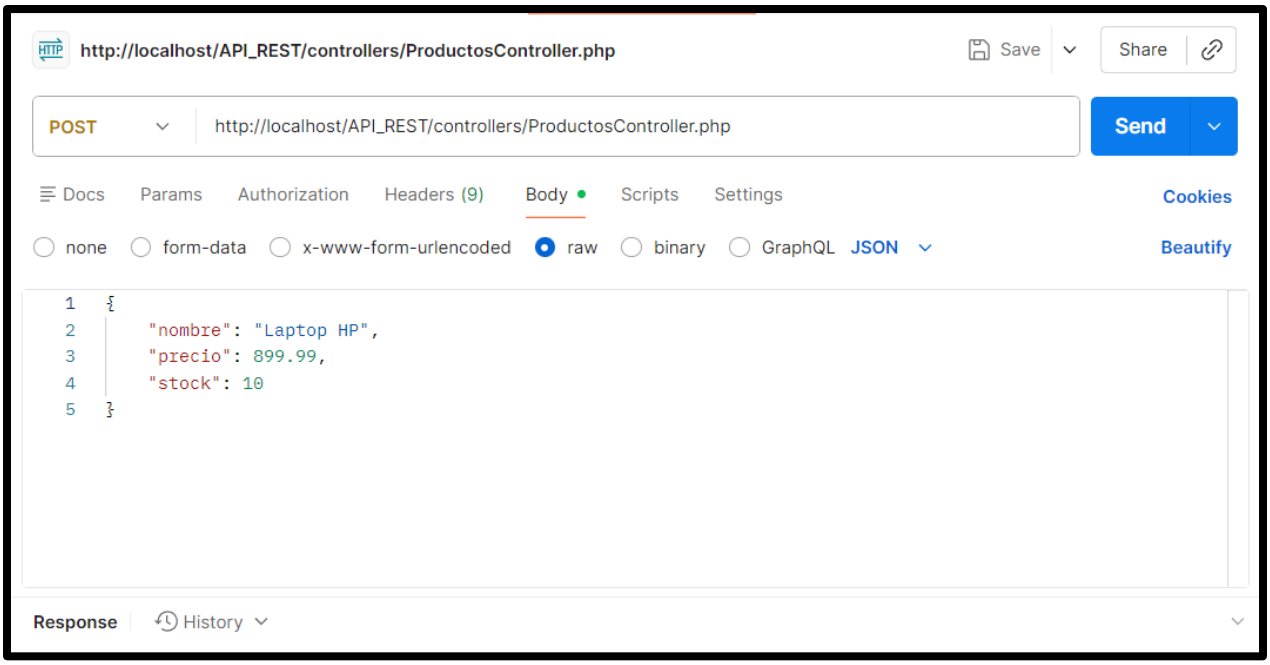

Ilustración 2 – Body método POST en POSTMAN

Ilustración 3 – Mensaje de éxito al ejecutar método POST en POSTMAN

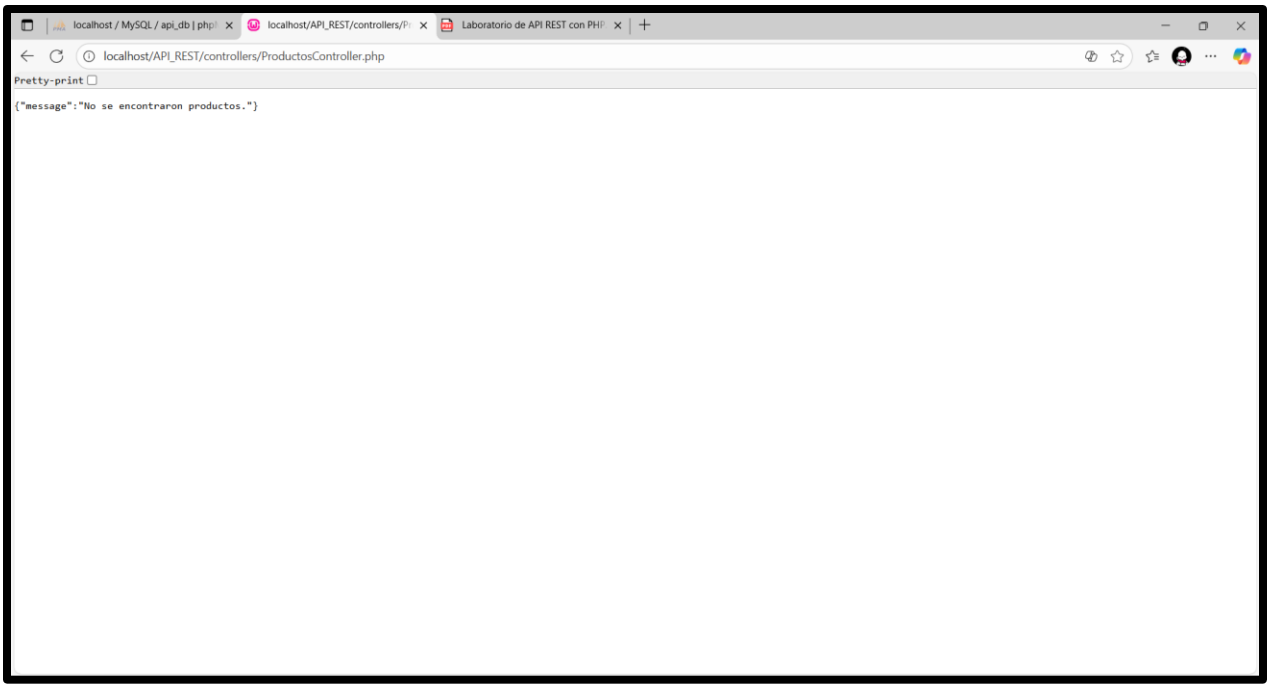

Ilustración 4 – Método GET en navegador (Error tabla vacía)

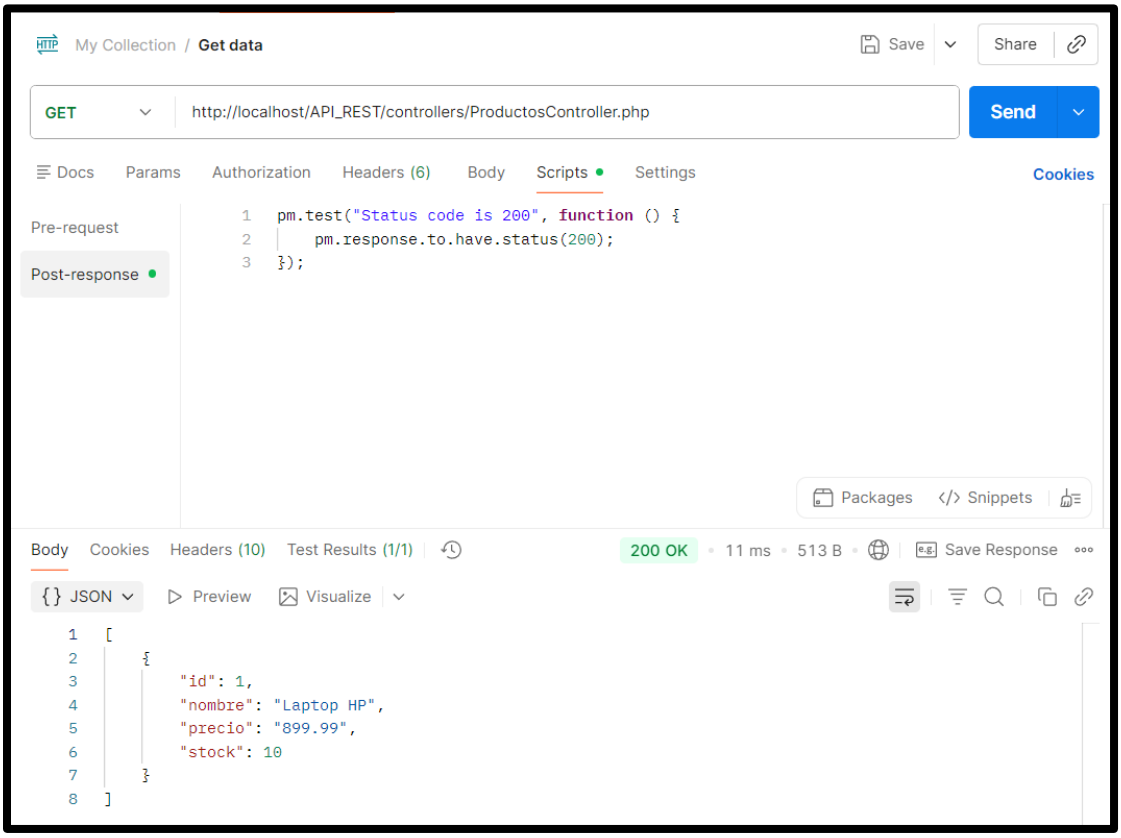

Ilustración 5 - Método GET implementado en POSTMAN

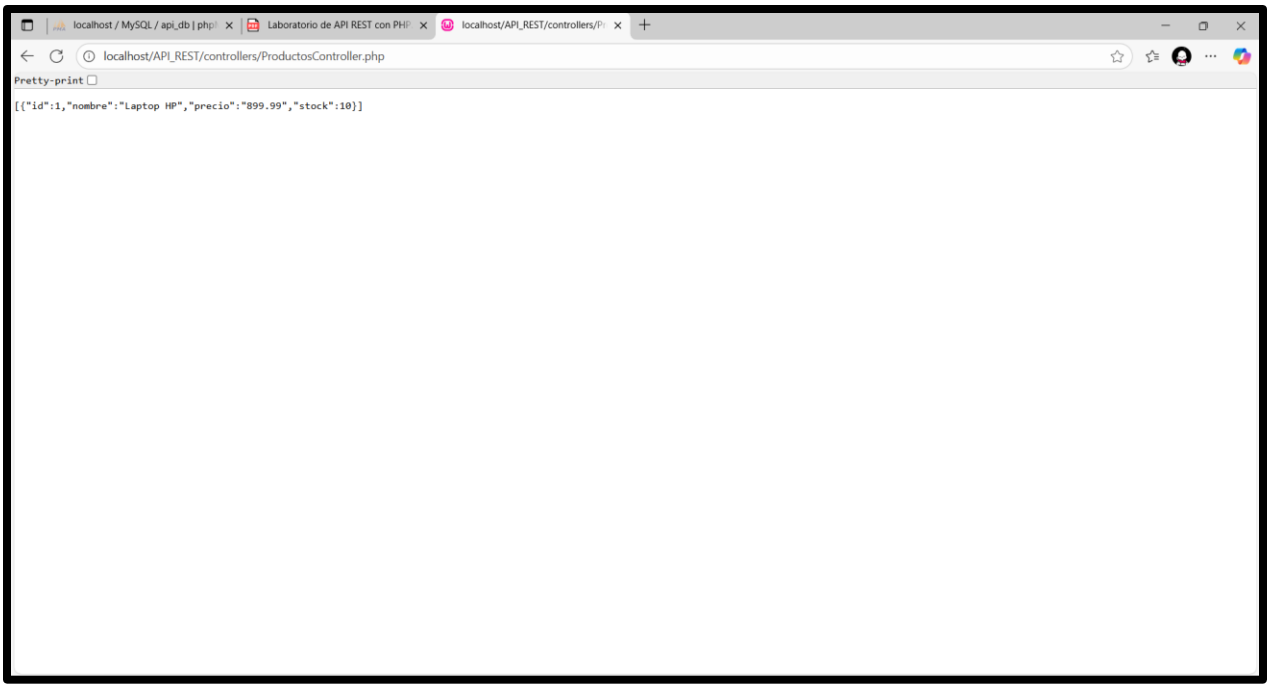

Ilustración 6 - Éxito de método GET (web)

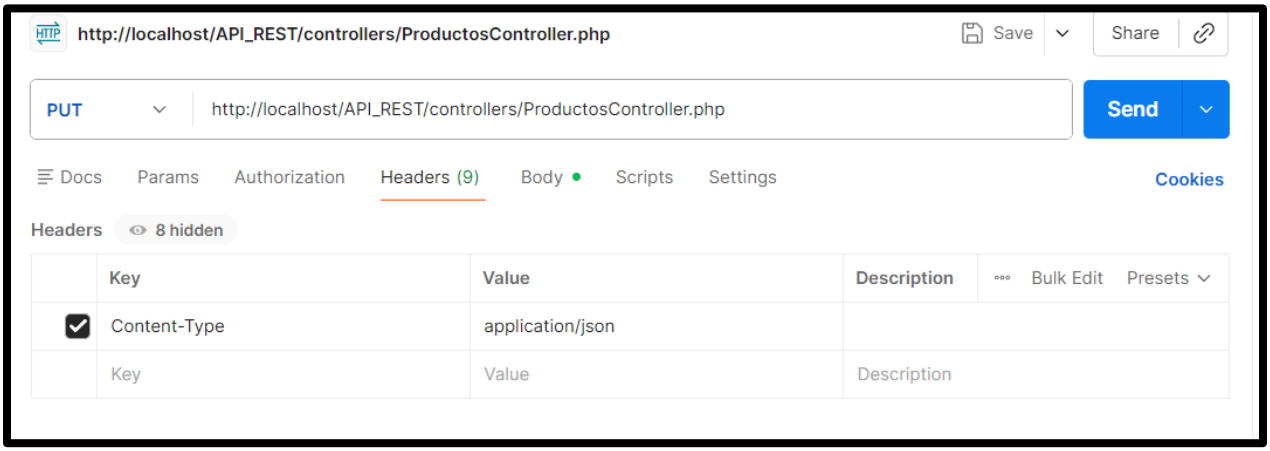

Ilustración 7 – Headers método PUT en POSTMAN

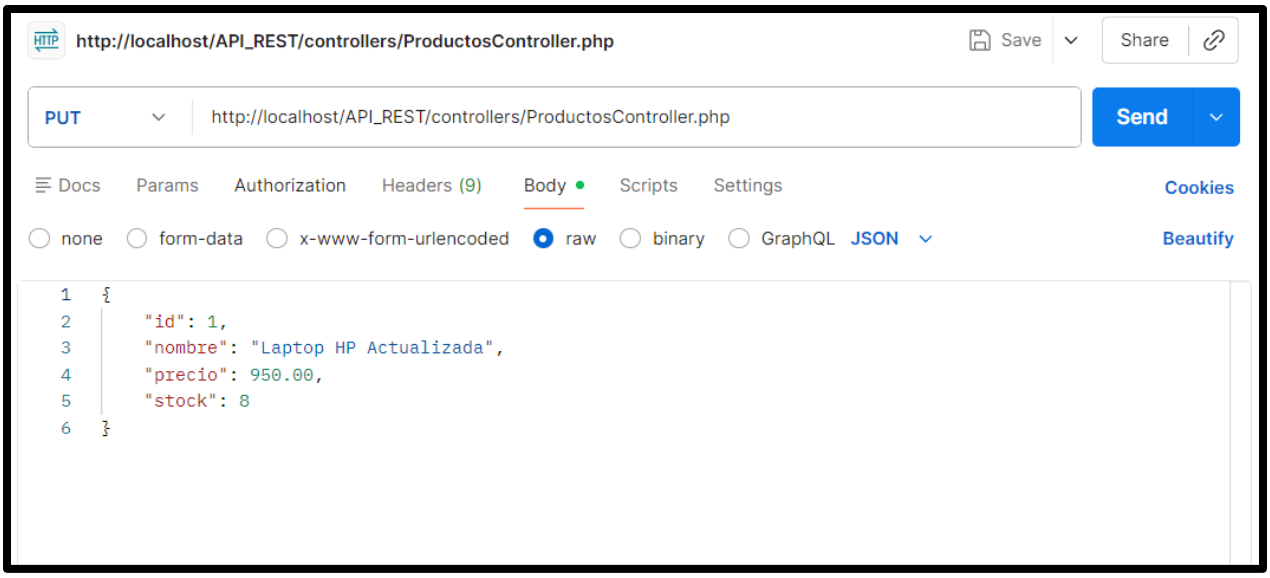

Ilustración 8 - Body método PUT en POSTMAN

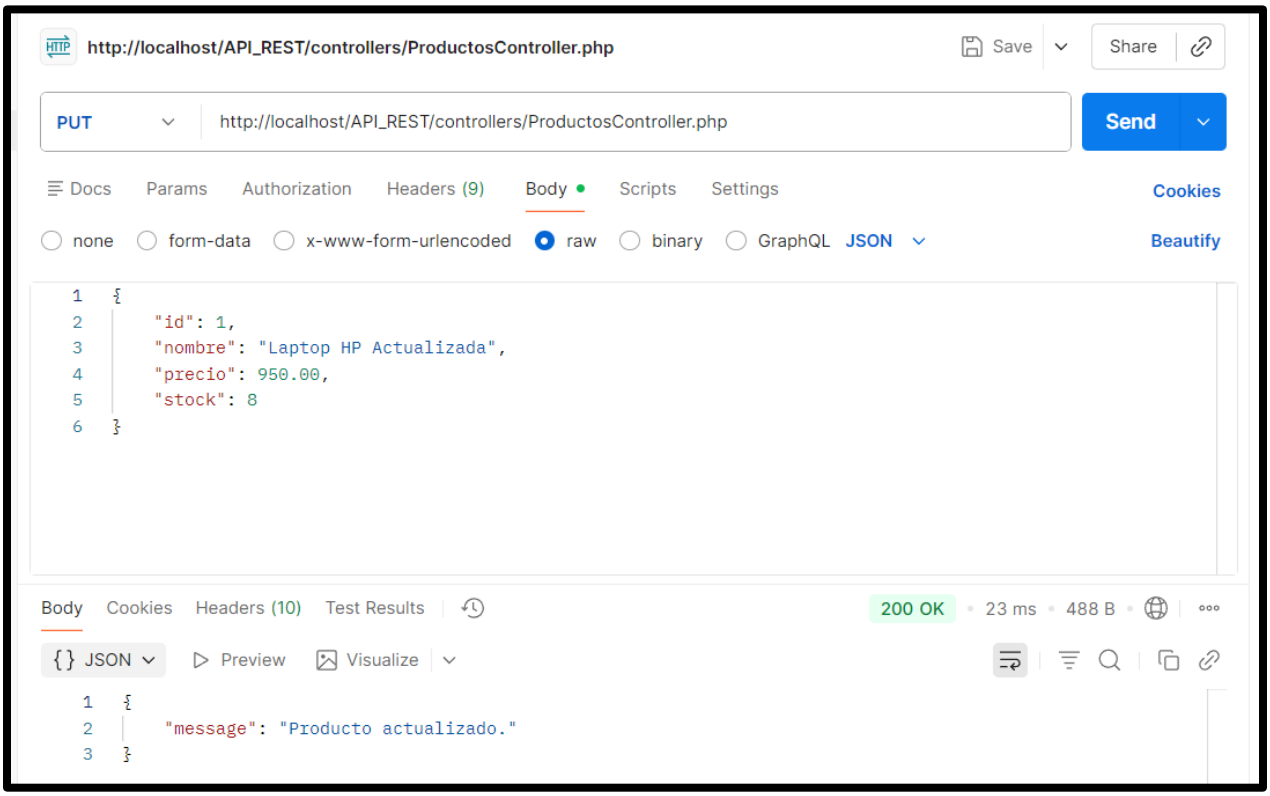

Ilustración 9 - Éxito de método PUT en POSTMAN

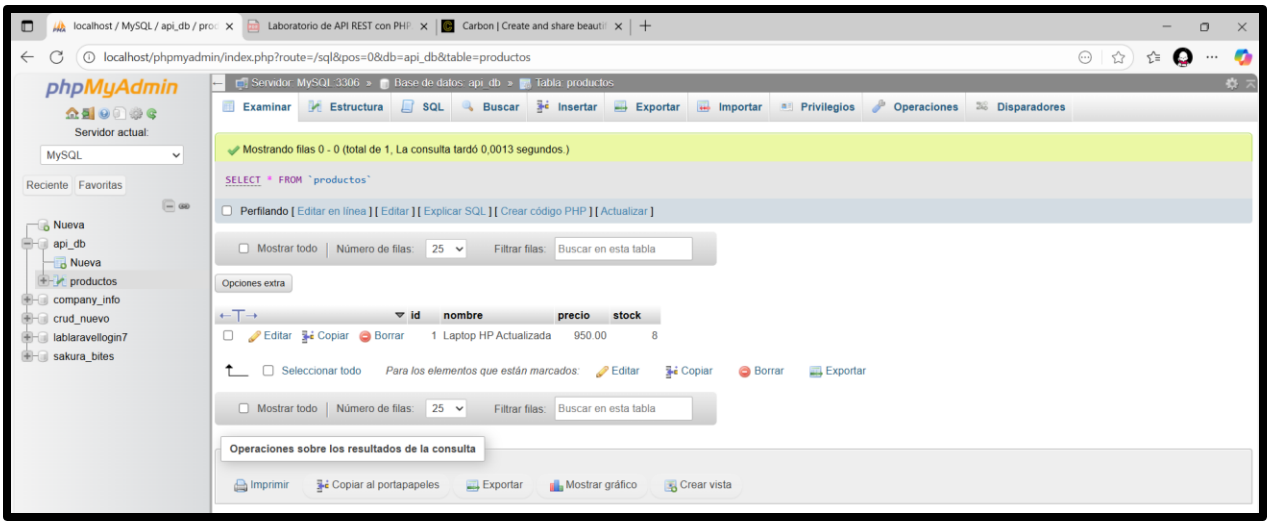

Ilustración 10 - Base de Datos con tabla

Resultados obtenidos

Funcionalidades Implementadas

- **POST:** Creación exitosa de productos con respuesta 201
- **GET:** Listado correcto de productos existentes
- **PUT:** Actualización funcional de productos por ID
- **Manejo de errores:** Respuestas HTTP apropiadas para diferentes escenarios

Códigos HTTP Utilizados

- **201 Created:** Producto creado exitosamente
- **200 OK:** Operación completada correctamente
- **404 Not Found:** No se encontraron productos
- **503 Service Unavailable:** Error en operación de base de datos
- **405 Method Not Allowed:** Método HTTP no soportado
- Sanitización de entradas.

Conclusiones

El desarrollo de esta API REST con PHP demostró el éxito en la implementación de los métodos HTTP POST, GET y PUT para la gestión de productos. Se logró comprender la estructura fundamental de una API RESTful, utilizando Postman para validar cada endpoint mediante pruebas exhaustivas. El código, organizado bajo el patrón MVC, incorporó un control centralizado de peticiones mediante switch y un manejo adecuado de errores con respuestas HTTP apropiadas. Todos los objetivos del laboratorio fueron alcanzados, confirmando la funcionalidad completa de la API y el correcto uso de las tecnologías involucradas.

Referencias

- Repositorio base: https://github.com/Salomon2514/Ejemplo_API_REST

Link del repositorio:

https://github.com/flopero29/Laboratorio---API-REST