

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Plataformas de Desarrollo de Software
Laboratorio: Desarrollo de aplicaciones en Eclipse y NetBeans

Valentina Ceballos Gaviria

Omar Gómez Rojas

Jhon Jairo Giraldo

Especialización en Ingeniería de Software

Fundación Universitaria Internacional Rioja

Presentado a:

Sergio Humberto Rueda Neira

Ing. De Sistemas

Noviembre 2022

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Contenido

Introducción	3
Definición de diagrama	4
Creación de API	5
MySQL	7
Postman -Testing	8

Bibliografía

¡Error! Marcador no definido.

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Introducción

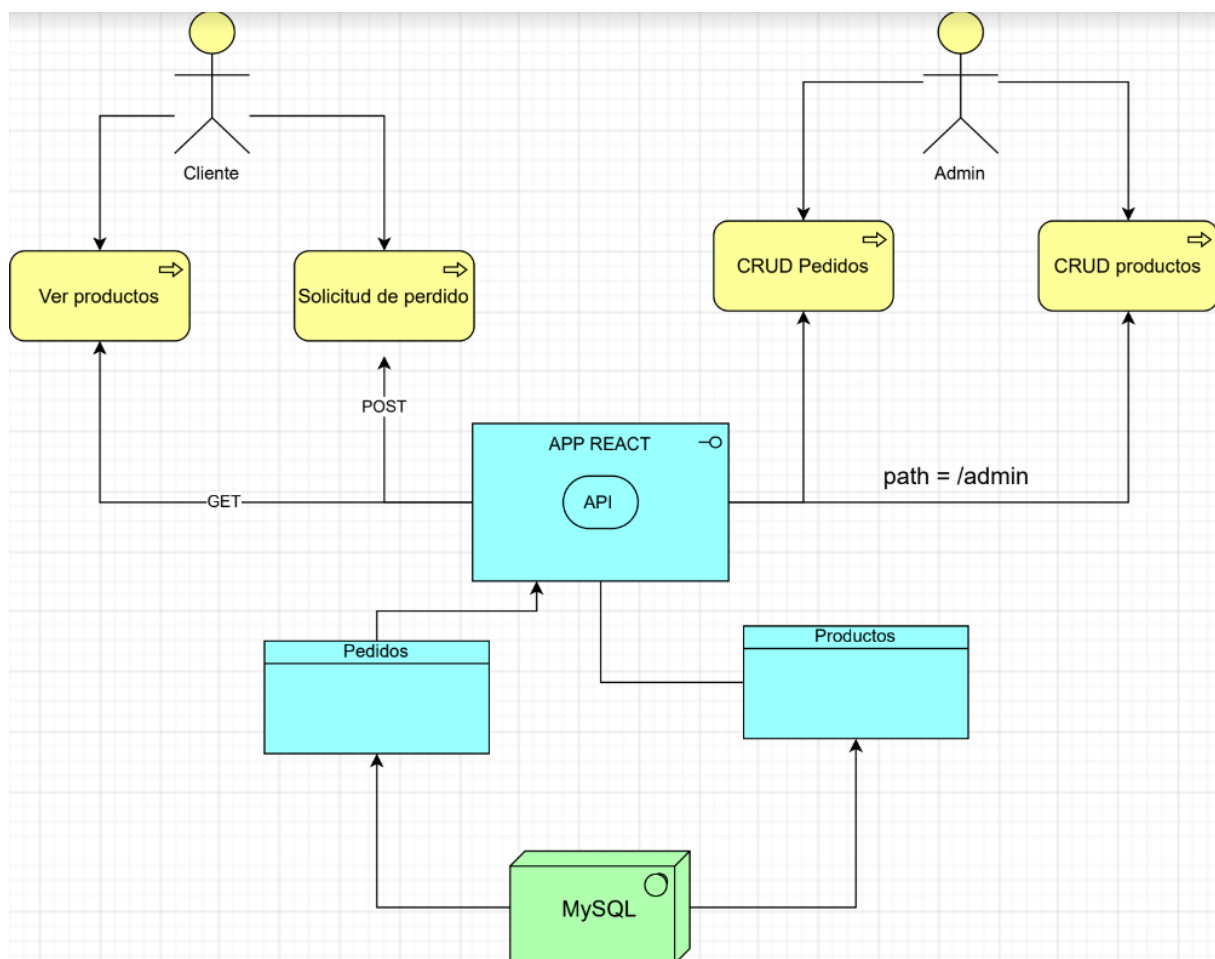
Un software hace referencia a un conjunto de programas, datos y procedimientos que permiten realizar tareas específicas en los sistemas, contando con distintos protocolos, los cuales hacen parte fundamental en el funcionamiento de las aplicaciones y webs; entre ellos se encuentran las API (Application Programming Interfaces) permitiendo la comunicación entre dos componentes.

Una API establece un modelo para la comunicación o interacción de un software con otro, para el cumplimiento de una o muchas funciones programadas por el desarrollador; asimismo son utilizadas para la integración de nuevas aplicaciones con software existente, siendo un componente esencial en la optimización de tiempo y dinero en el desarrollo.

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Definición de diagrama

En el diagrama propuesto se evidencian dos actores el cliente y el administrador, los cuales tendrán una interacción directa con la app; por una parte el cliente puede ver los productos y solicitar un pedido, mediante los métodos GET y POST los cuales están debidamente conectados a la base de datos, la cual tiene la información que ha sido registrada por el administrador ya que este tiene la capacidad de actualizar, crear y eliminar un producto de la base de datos.



Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Creación de API

Inicialmente se crea el archivo .zip por medio de Spring Initializr que contiene toda la estructura del proyecto; a través del lenguaje JAVA se hace el desarrollo de la API que se quiere utilizar, por medio de MySql se crea la base de datos y los metos son comprobados por medio de Postman. Se configura de manera previa la conexion a la base de datos y sus variables; posteriormente se crean las entidades o modelos, donde se encontrarán los atributos con sus respectivas restricciones según se estipule.

```
package com.unir.api_sql.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "productos", catalog = "tienda", schema = "")
public class Producto {
    @Id
    @Column
    private Integer idproducto;
    @Column
    private String codigo;
    @Column
    private String nombre;
    @Column
    private float precio;
    @Column
    private int cantidad;
    public Integer getIdproductos() {
        return idproducto;
    }
    public void setIdproductos(Integer idproductos) {
        this.idproducto = idproductos;
    }
    public String getCodigo() {
        return codigo;
    }
    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }
    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public float getPrecio() {
        return precio;
    }
    public void setPrecio(float precio) {
        this.precio = precio;
    }
}
```

```
package com.unir.api_sql.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "pedidos", catalog = "tienda", schema = "")
public class Pedido {
    @Id
    @Column
    private Integer idpedido;
    @Column
    private String nombres;
    @Column
    private String correo;
    @Column
    private String direccion;
    @Column
    private String ciudad;
    public Integer getIdpedido() {
        return idpedido;
    }
    public void setIdpedido(Integer idpedido) {
        this.idpedido = idpedido;
    }
    public String getNombres() {
        return nombres;
    }
    public void setNombres(String nombres) {
        this.nombres = nombres;
    }
    public String getCorreo() {
        return correo;
    }
    public void setCorreo(String correo) {
        this.correo = correo;
    }
    public String getDireccion() {
        return direccion;
    }
    public void setDireccion(String direccion) {
        this.direccion = direccion;
    }
}
```

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Además, se crean los CRUD para trabajar las interfaces de la base de datos extendiendo de esta manera el repositorio, así mismo se elabora el controlador, donde se definen todos los estereotipos o estándares que se quieren implementar.

```
package com.unir.api_sql.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PatchMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.unir.api_sql.model.Pedido;
import com.unir.api_sql.model.Producto;
import com.unir.api_sql.repository.InterfacePedido;

@RestController
@RequestMapping("/pedidos")
public class PedidoController {

    @Autowired
    private InterfacePedido interfacePedido;

    @GetMapping
    public List<Pedido> pedidos(){
        return (List<Pedido>) interfacePedido.findAll();
    }

    @PostMapping
    public void insertar(@RequestBody Pedido pe) {
        interfacePedido.save(pe);
    }

    @PatchMapping (value="/{id}")
    public void modificar(@RequestBody Pedido pe, @PathVariable("id") Integer id) {
        interfacePedido.save(pe);
    }

    @DeleteMapping(value="/{id}")
    public void eliminar(@PathVariable("id") Integer id){
        interfacePedido.deleteById(id);
    }
}
```

```
package com.unir.api_sql.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PatchMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.unir.api_sql.model.Producto;
import com.unir.api_sql.repository.InterfaceProducto;

@RestController
@RequestMapping("/productos")
public class ProductoController {

    @Autowired
    private InterfaceProducto interfaceProducto;

    @GetMapping
    public List<Producto> productos(){
        return (List<Producto>) interfaceProducto.findAll();
    }

    @PostMapping
    public void insertar(@RequestBody Producto pe) {
        interfaceProducto.save(pe);
    }

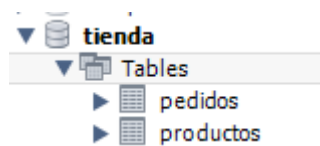
    @PatchMapping (value="/{id}")
    public void modificar(@RequestBody Producto pe, @PathVariable("id") Integer id) {
        interfaceProducto.save(pe);
    }

    @DeleteMapping(value="/{id}")
    public void eliminar(@PathVariable("id") Integer id){
        interfaceProducto.deleteById(id);
    }
}
```

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

MySQL

Se utiliza como gestor de base de datos MySQL-Workbench , así mismo se crea una base de datos llamada tienda y en ella dos tablas una llamada productos y otra pedidos.



- **Tabla productos**

	idproducto	codigo	nombre	precio	cantidad
	1	001	Computadores	800	8
	2	002	Celulares	900	25
	3	003	playStation	1600000	60
	4	004	xbox One-Series S	1800000	120
	5	005	Televisores	1256897	145
	7	007	Impresora	5687545	1500
	NULL	NULL	NULL	NULL	NULL

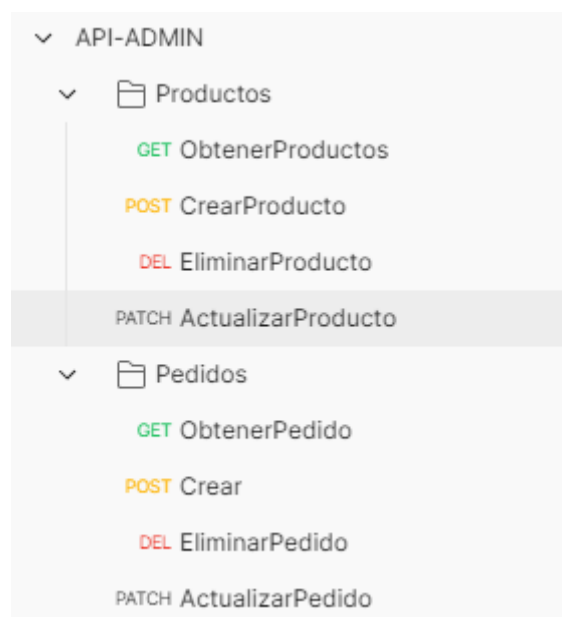
- **Tabla pedidos**

	idpedido	nombres	correo	direccion	ciudad
	1	Omar Gomez	omar@gmail.com	cra 64 a # 65 s	Bogota
	2	Carolina Vega Amaya	carolina@gmail.com	cra 98 a # 88 s	Bogota
	NULL	NULL	NULL	NULL	NULL

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

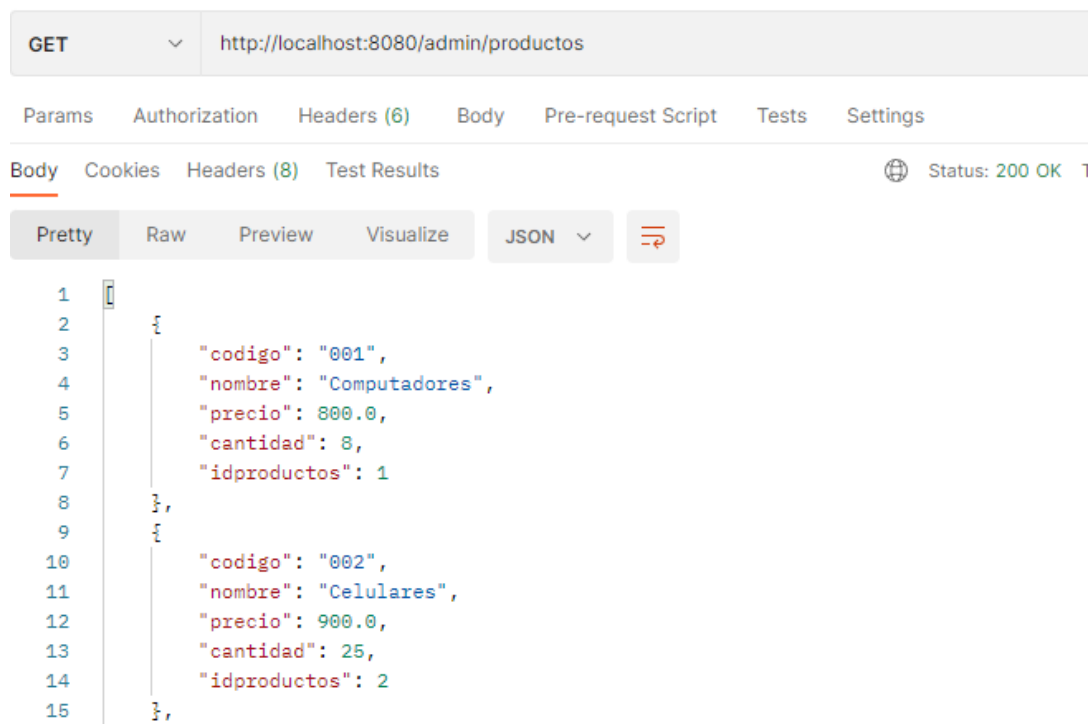
Postman -Testing

Se utilizó la herramienta Postman para verificar el funcionamiento de la API con sus respectivos métodos (GET,POST,PATCH,DEL)



Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

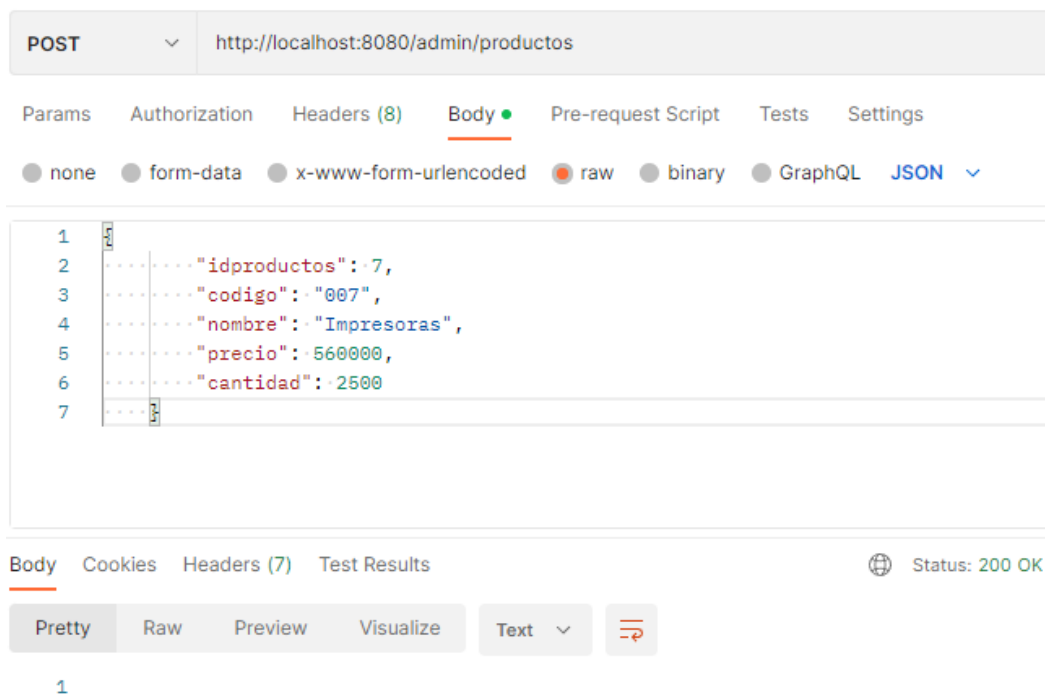
- Método **GET** -Obteniendo todas las filas de la tabla productos.



Al enviar la solicitud al servidor podemos observar que nos retorna un código 200 OK y no muestra todas las filas que están creadas en la tabla productos de MySQL como se aprecia en la imagen.

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

- Método **POST** -Creando un nuevo producto.

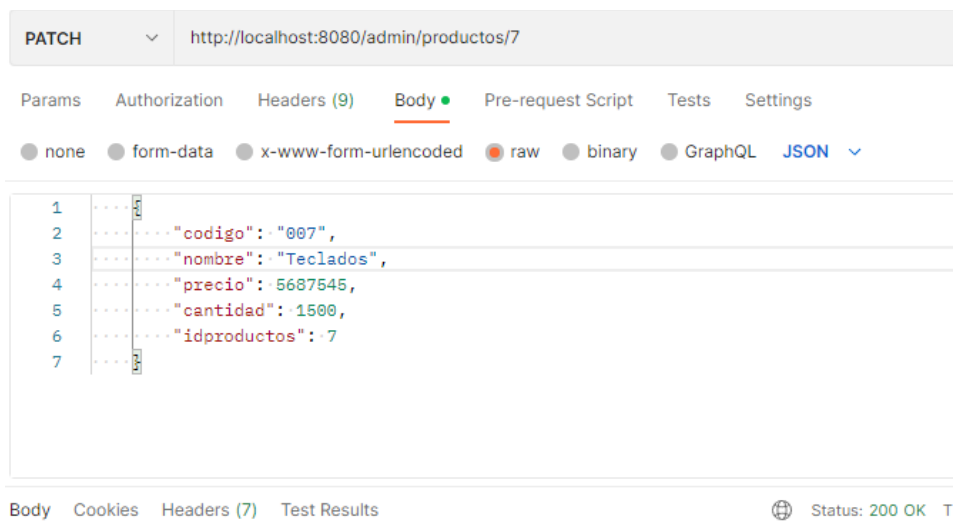


Al enviar la solicitud al servidor podemos observar que nos retorna un código 200 OK y nos crea una nueva fila con los datos enviados al servidor, como se aprecia en la imagen

	idproducto	codigo	nombre	precio	cantidad
	1	001	Computadores	800	8
▶	2	002	Celulares	900	25
	3	003	playStation	1600000	60
	4	004	xbox One-Series S	1800000	120
	5	005	Televisores	1256897	145
	6	006	Monitores	5687545	1500
	7	007	Impresoras	560000	2500
*	NULL	NULL	NULL	NULL	NULL

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

- Método **PATCH**-Actualizando un elemento de la tabla producto.



Al enviar la solicitud al servidor podemos observar que nos retorna un código 200 OK y nos actualiza el nombre del producto con el id : 7 , pasa de ser Impresoras a ser teclados.

	idproducto	codigo	nombre	precio	cantidad
	1	001	Computadores	800	8
▶	2	002	Celulares	900	25
	3	003	playStation	1600000	60
	4	004	xbox One-Series S	1800000	120
	5	005	Televisores	1256897	145
	6	006	Monitores	5687545	1500
	7	007	Teclados	5687545	1500
✱	NULL	NULL	NULL	NULL	NULL

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

- Método **DELETE**-Eliminar un elemento de la tabla producto.


DELETE
⌵
http://localhost:8080/admin/productos/6

Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings

Query Params

	KEY	VALUE	DE
	Key	Value	D

Body
Cookies
Headers (7)
Test Results

 Status: 200 OK

Al enviar la solicitud al servidor podemos observar que nos retorna un código 200 OK y nos elimina el producto con el id : 6

Asignatura	Datos del alumno	Fecha
Plataformas de Desarrollo de Software	Apellidos: Ceballos, Gómez, Giraldo	21-11-22
	Nombre: Valentina, Omar, Jhon	

Bibliografía

Ed-douibi, H. I. (2018). Una propuesta para componer APIs orientadas a datos.

Hernández, J. (2014). Análisis y desarrollo web. *Jesús Hernández*.

Mendoza González, G. (2015). Herramienta de Desarrollo Netbeans. pág, 1.