

Desarrollo web en entorno servidor

# Plataforma Digital: NetGonVideo

Utilizando Spring Tool y Postman

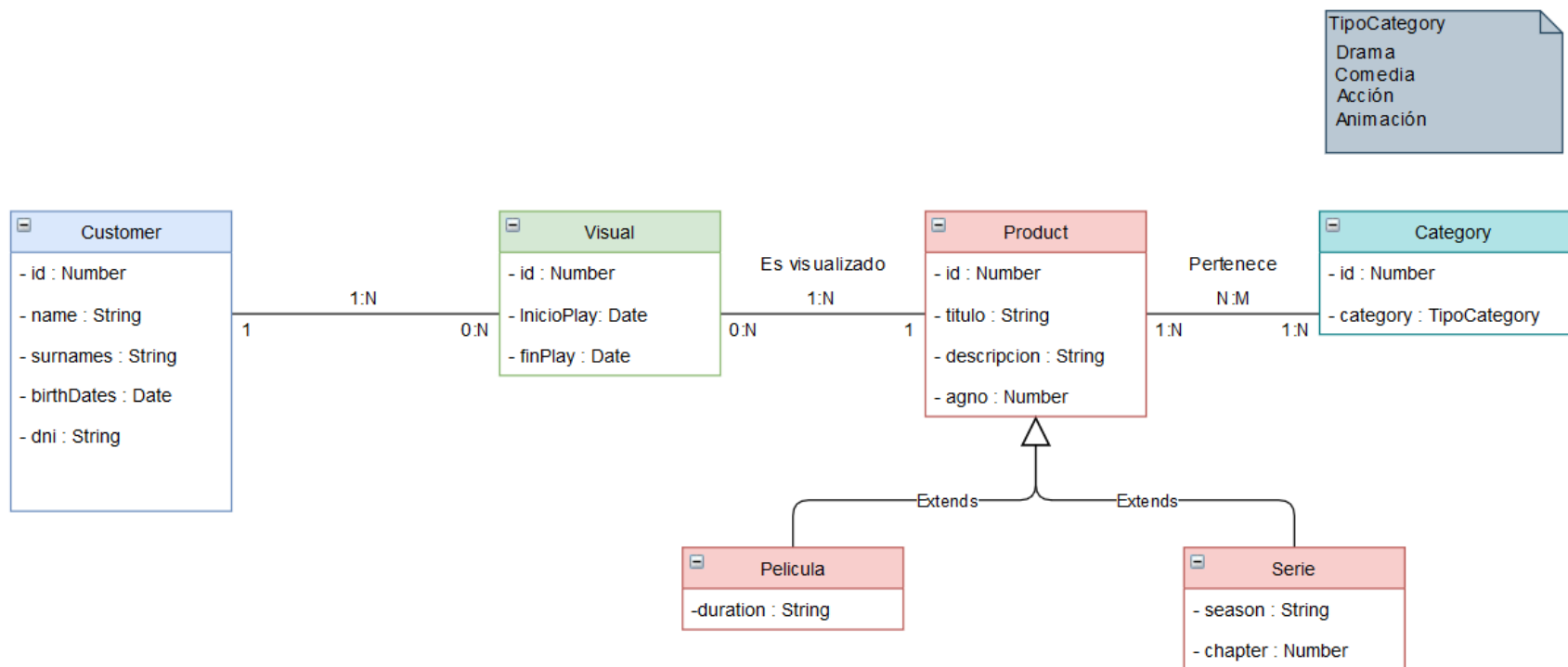
# 1.- Introducción a la propuesta

## NetGonVideo

- Es una plataforma digital que permite a los clientes visualizar series y películas.
- Cada uno de estos productos tiene una o varias categorías.
- He planteado dividirlo en una serie de Entidades: Cliente, Producto, Película, Serie, Categoría y Visual.

## 2.- UML para el diagrama de clases

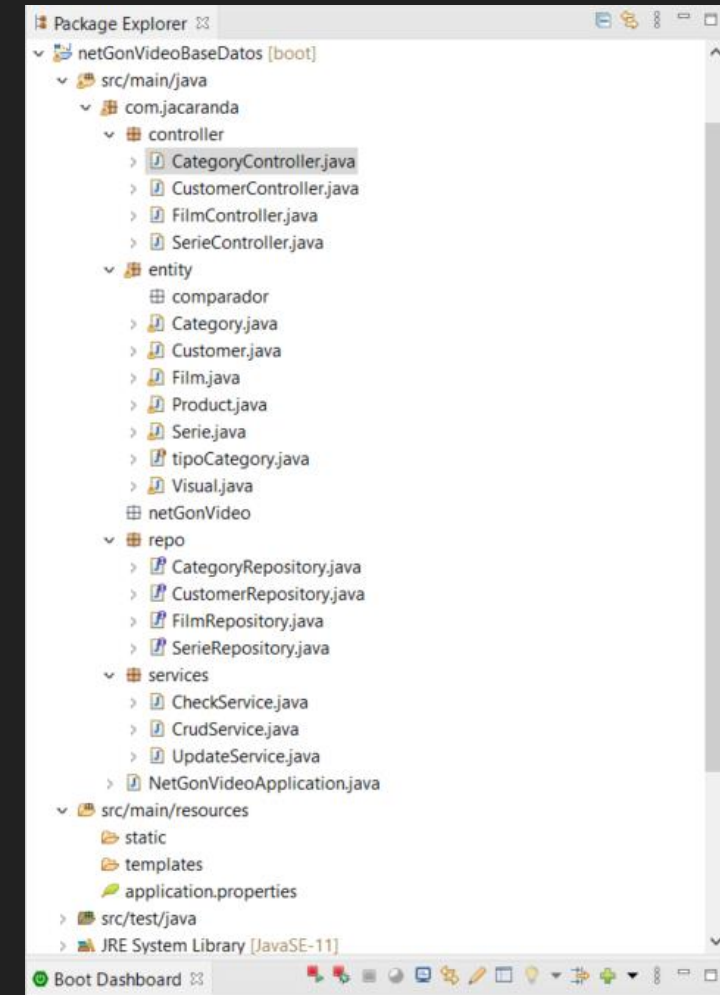
### NetGonVideo UML



## 3.- Estructura de NetGonVideo

### Paquetes:

- Controller → Todos los controladores.
- Entity → Todas las entidades.
- netGonVideo → ScanBasePackages (controller y servicios).
- Repo → Todas interfaces CrudRepository
- Services → Todos los servicios.



# 3.- Estructura de NetGonVideo

Entity:  
Customer

```
1 package com.jacaranda.entity;
2
3
4 *import java.io.Serializable;[]
5
6 @Entity
7 public class Customer implements Comparable<Customer>,Serializable {
8     //Variables
9
10    @Id
11    @GeneratedValue(strategy = GenerationType.AUTO)
12    private long id;
13
14    private String name;
15    private String surnames;
16
17    @NonNull
18    @DateTimeFormat(pattern = "ddMMyyyy")
19    private LocalDate birthDate;
20
21    private String dni;
22
23    // Relaciones con la entidad Visual
24    @OneToMany(mappedBy="customer")
25    private List<Visual> visualizaciones;
26
27    //Constructores
28    public Customer() {
29        super();
30        visualizaciones = new ArrayList<Visual>();
31    }
32
33    public Customer(String name, String surnames, LocalDate birthDate, String dni) {
34        super();
35        this.name = name;
36        this.surnames = surnames;
37        this.birthDate = birthDate;
38        this.dni = dni;
39        visualizaciones = new ArrayList<Visual>();
40    }
41
42    //GET y SET
43
44    public long getId() {
45        return id;
46    }
47
48    public String getName() {
49        return name;
50    }
```

```
65
66
67 public void setBirthDate(LocalDate birthDate) {
68     this.birthDate = birthDate;
69 }
70
71
72 public String getDni() {
73     return dni;
74 }
75
76
77
78 public List<Visual> getVisualizaciones() {
79     return visualizaciones;
80 }
81
82 //ComparaTo
83 @Override
84 public int compareTo(Customer other) {
85
86     return Integer.valueOf(getId()).compareTo(getId());
87 }
88
89 }
```

## 3.- Estructura de NetGonVideo

Entity:  
Product

```
1 package com.jacaranda.entity;
2
3 import java.io.Serializable;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22 @MappedSuperclass
23 public class Product implements Serializable{
24     @Id
25     @GeneratedValue(strategy = GenerationType.AUTO)
26     private long id;
27     private String title;
28     private String description;
29     private int agno;
30
31     // Relación con la entidad Categoría
32     @ManyToOne(cascade=CascadeType.ALL)
33     @JoinColumn(name="categorias_id", foreignkey = @ForeignKey(name="categorias_id_fk"), nullable = false)
34
35     //Objeto categoría para la relación
36     private Category categorias;
37
38
39     //Constructores
40     public Product() {
41         super();
42     }
43
44
45     public Product(String title, String description, int agno, Category categorias) {
46         super();
47         this.title = title;
48         this.description = description;
49         this.agno = agno;
50         this.categorias = categorias;
51     }
52
53
54     //GET Y SET
55
56     public String getTitle() {
57         return title;
58     }
59
60     public String getDescription() {
61         return description;
62     }
63
64     public void setDescription(String description) {
65         this.description = description;
66     }
67
68     public int getAgno() {
```

```
        public void setAgno(int agno) {
            this.agno = agno;
        }
        public void setId(int id) {
            this.id = id;
        }
        public long getId() {
            return id;
        }
        public Category getCategorias() {
            return categorias;
        }
    }
}
```



## 4.- Controladores de NetGonVideo

Realizamos el Servicio para el crud.

Enrutamos los Métodos:

Get

Post

Put

Delete

```
package com.jacaranda.controller;

import org.springframework.beans.factory.annotation.Autowired;

@RestController
@RequestMapping(path = "/api")

public class CustomerController {

    // Creamos el servicio
    @Autowired
    private CrudService crudService;

    // Método GET
    @GetMapping("/customers")
    public ResponseEntity<> getCustomer() {
        return crudService.getCustomer();
    }

    // Método Get por parámetro id
    @GetMapping("/customers/{id}")
    public ResponseEntity<> getCustomerId(@PathVariable long id) {
        return crudService.getCustomerId(id);
    }

    // Método POST
    @PostMapping("/customers")
    public ResponseEntity<> createCustomer(@RequestBody Customer sent) {
        return crudService.createCustomer(sent);
    }

    // Método PUT
    @PutMapping("/customers")
    public ResponseEntity<> updateCustomer(@RequestBody Customer sent) {
        return crudService.updateCustomer(sent);
    }

    // Método DELETE
    @DeleteMapping("/customers/{id}")
    public ResponseEntity<> deleteCustomer(@PathVariable long id) {
        return crudService.deleteCustomer(id);
    }
}
```

# 5.- Servicios de NetGonVideo

CrudService:

Realizamos los crud de las entidades.

UpdateService

Servicio para actualizar las entidades en el crud

```
package com.jacaranda.services;

import java.util.Collection;

@Service
public class CrudService {

    // Añadiendo los repositorios
    @Autowired
    private CrudRepository<Customer, Long> crudRepositoryCustomer;

    // @Autowired
    // private CrudRepository<Product, Long> crudRepositoryProduct;

    @Autowired
    private CrudRepository<Category, Long> crudRepositoryCategory;

    @Autowired
    private CrudRepository<Film, Long> crudRepositoryFilm;

    @Autowired
    private CrudRepository<Serie, Long> crudRepositorySerie;

    @Autowired
    private CustomerRepository customerRepository;

    // @Autowired
    // private ProductRepository productRepository;

    @Autowired
    private CategoryRepository categoryRepository;

    @Autowired
    private FilmRepository filmRepository;

    @Autowired
    private SerieRepository serieRepository;

    // Añadiendo Servicios
    @Autowired
    private UpdateService updateService;

    // ENTIDAD CUSTOMER

    // Método Get Customer
    public ResponseEntity<?>getCustomer(){
        Iterable<Customer> resultado = crudRepositoryCustomer.findAll();
        ResponseEntity<?> response;
        if(((Collection<?>)resultado).size() == 0) {
            response= ResponseEntity
                .status(HttpStatus.NOT_FOUND)
                .body("La base de datos está vacía");
        }else {
            response= ResponseEntity
                .status(HttpStatus.OK)
                .body(crudRepositoryCustomer.findAll());
        }
        return response;
    }

    // Método Get con parámetro id
    public ResponseEntity<?> getCustomerId(@PathVariable long id){
        Customer customer = customerRepository.findCustomerId(id);
        ResponseEntity<?> response;
        if(customer==null) {
            response = ResponseEntity
                .status(HttpStatus.NOT_FOUND)
                .body("El cliente no existe");
        }else {
            response = ResponseEntity
                .status(HttpStatus.OK)
                .body(crudRepositoryCustomer.findById(id));
        }
        return response;
    }
}
```

```
// ENTIDAD CUSTOMER

// Método Get Customer
public ResponseEntity<?>getCustomer(){
    Iterable<Customer> resultado = crudRepositoryCustomer.findAll();
    ResponseEntity<?> response;
    if(((Collection<?>)resultado).size() == 0) {
        response= ResponseEntity
            .status(HttpStatus.NOT_FOUND)
            .body("La base de datos está vacía");
    }else {
        response= ResponseEntity
            .status(HttpStatus.OK)
            .body(crudRepositoryCustomer.findAll());
    }
    return response;
}

// Método Get con parámetro id
public ResponseEntity<?> getCustomerId(@PathVariable long id){
    Customer customer = customerRepository.findCustomerId(id);
    ResponseEntity<?> response;
    if(customer==null) {
        response = ResponseEntity
            .status(HttpStatus.NOT_FOUND)
            .body("El cliente no existe");
    }else {
        response = ResponseEntity
            .status(HttpStatus.OK)
            .body(crudRepositoryCustomer.findById(id));
    }
    return response;
}
```



## 6.- Repo de NetGonVideo

Interfaces donde va la query y métodos de algunas entidades.

CategoryRepository

CustomerRepository

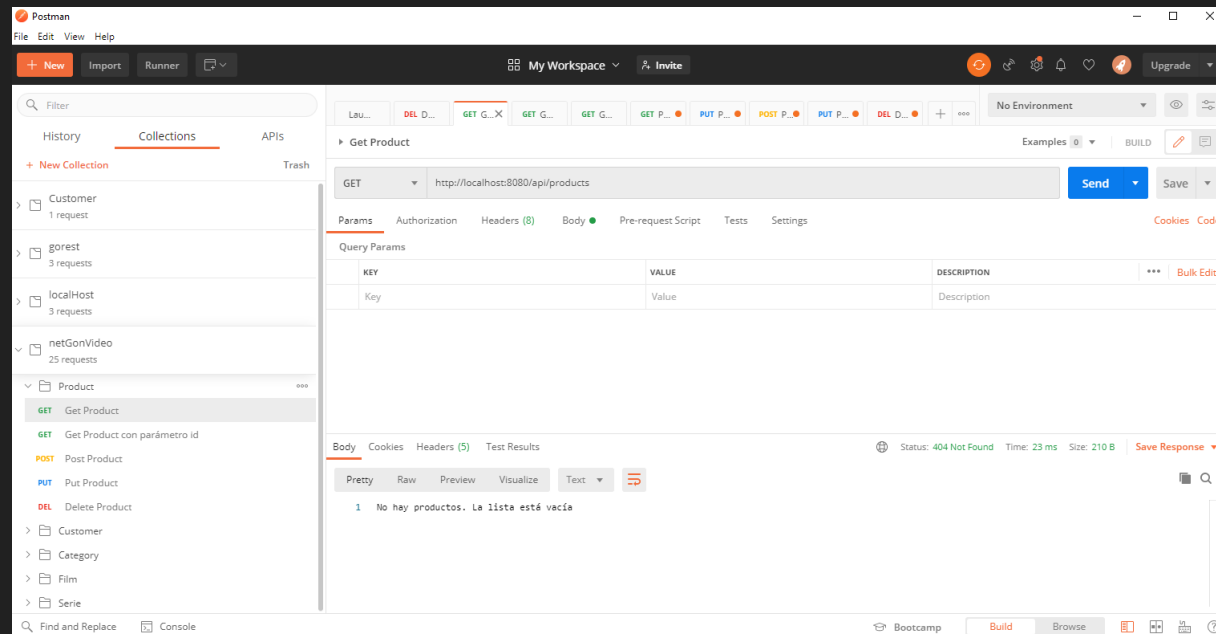
FilmRepository

SerieRepository

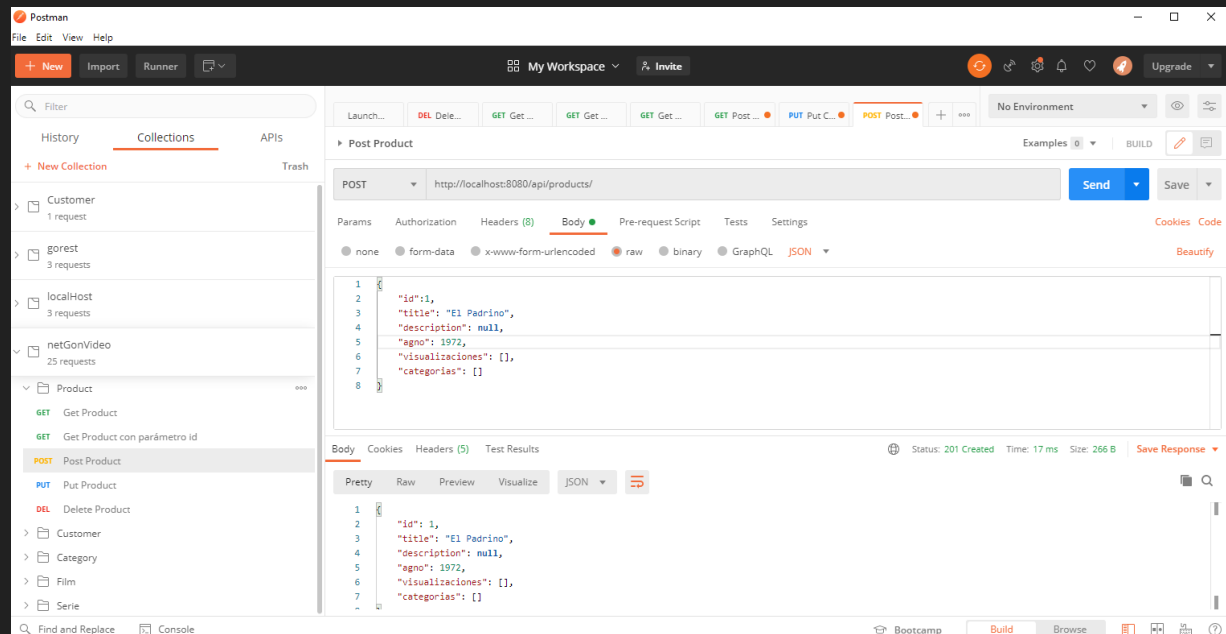
```
CustomerRepository.java
1 package com.jacaranda.repo;
2
3 import java.util.List;
4
10
11 @Repository
12 public interface CustomerRepository extends CrudRepository<Customer, Long>{
13
14     @Query(value="select * from Customer order by name", nativeQuery = true)
15     public List<Customer> findAllOrderedByName();
16
17     public Customer findCustomerByDni(String dni);
18
19     public Customer findCustomerById(Long id);
20
21 }
22
```

# 7.- Peticiones de NetGonVideo con Postman

GET

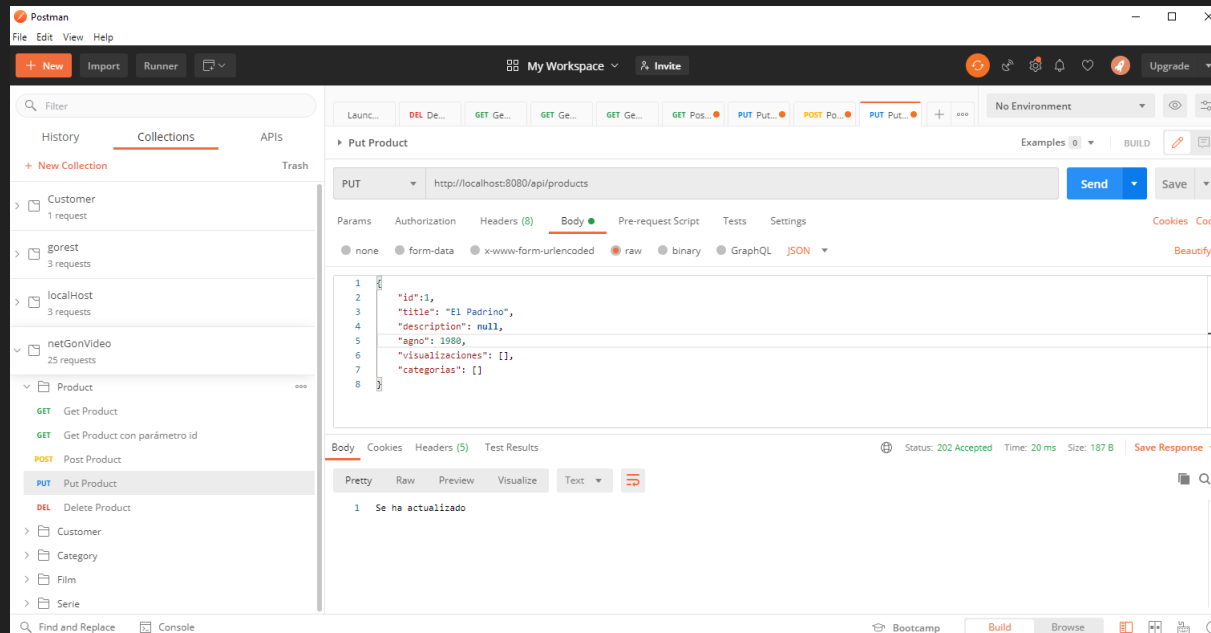


POST

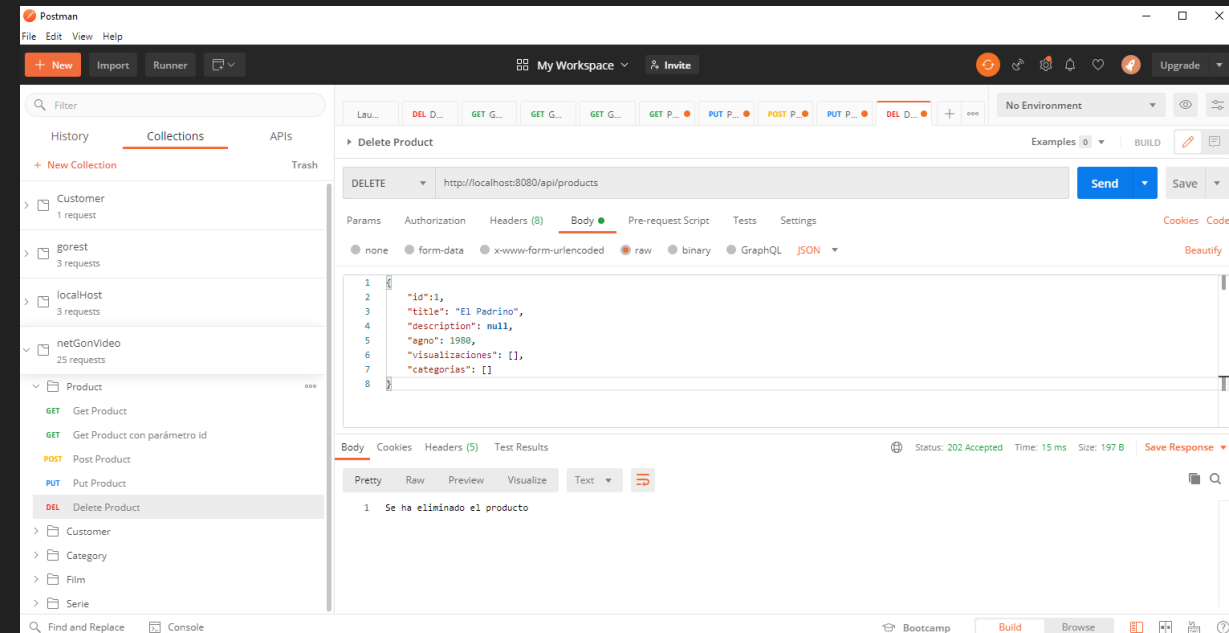


# 8.- Peticiones de NetGonVideo con Postman

## PUT



## DELETE



## 6.- Dificultades de NetGonVideo encontradas

- Problema de escaneado con los controladores.

**Solución:** En la clase NetGonVideoApplication añadir a la etiqueta `@SpringBootApplication(scanBasePackages = {"com.jacaranda.controller", "com.jacaranda.services"})` los paquetes correspondientes.

- Problema con la etiqueta `@Autowired`.

**Solución:** Colocando la etiqueta `@Autowired` por cada servicio en la correspondiente clase del controlador.

- Problema con la interface `Serializable`.

**Solución:** Toda clase que se le pasa un objeto en el body de la petición debe llevar siempre el `Serializable`.

- Error de `NullPointerException`

**Solución:** Al corregir el error de la etiqueta `Autowired` y el de escaneo de paquetes el `NullPointerException` desapareció.

## 9.- Propuestas de mejoras de NetGonVideo

- Ordenar la lista de productos.

Desarrollo web en entorno servidor

# GRACIAS

<https://github.com/Kurtgon/netGonVideo>