

SPRINT 3: Desarrollo del Backend

Identificación Proyecto	
Nombre Proyecto:	Vaper Shop App
Número Equipo:	
Integrantes del equipo	
Rol	Nombre
(Líder-Desarrollador – Cliente)	
(Líder-Desarrollador – Cliente)	Weimar Moisés Álvarez Cardona
(Líder-Desarrollador – Cliente)	Diego Alejandro Vera Jiménez

Evidencia construcción del Backend

Construcción de Backend con Spring-boot y Visual Studio

```
Ф
         ✓ WMESSENCE
         > la .vscode

v la src
                                                                                  import com.wmessence.wmessence.Models.ProductosModel;
import com.wmessence.wmessence.Services.ProductosServices;
import com.wmessence.wmessence.Repositories.ProductosRepository;
          → 🖮 main
           import org.springframework.web.bind.annotation.*;

✓ ■ Models

                                                                                  import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
                  You hace 8 diss | 1 author (You)

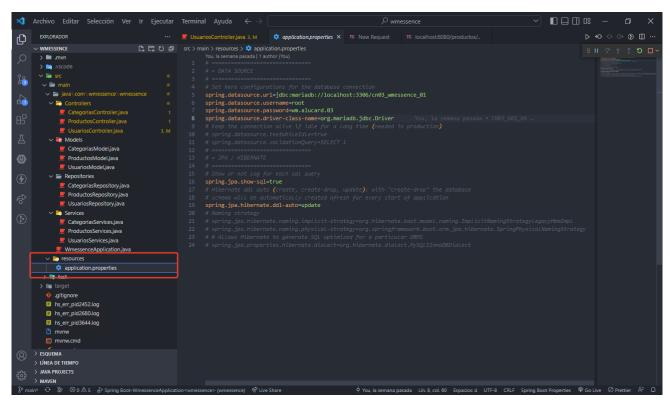
@RestController

@CrossOrigin (""")

@RequestMapping ("/productos")

public class ProductosController {
                Repositories
                   CategoriasRepository.java
                   UsuariosRepository.iava
                   WmessenceApplication.java
                                                                                        @PostMapping (value = "/")
@ResponseBody
public ResponseEntity<ProductosModel> agregar (@RequestBody ProductosModel productosModel){
    System.out.println(productosModel);
    ProductosModel obj = productosServices.save(productosModel);
    return new ResponseEntity<>(obj, HttpStatus.OK);
             hs_err_pid2680.log
                                                                                        mvnw 🖺
           / pom.xml
       > LÍNEA DE TIEMPO
                                                                                                     return new ResponseEntity<>(obj, HttpStatus.INTERNAL_SERVER_ERROR);
       > JAVA PROJECTS
```













Código JAVA Categorías Controller

```
package com.wmessence.wmessence.Controllers;
import com.wmessence.wmessence.Models.CategoriasModel;
import com.wmessence.wmessence.Repositories.CategoriasRepository;
import com.wmessence.wmessence.Services.CategoriasServices;
import java.util.List;
import org.springframework.web.bind.annotation.*;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.beans.factory.annotation.Autowired;
@RestController
@CrossOrigin ("*")
@RequestMapping ("/categorias")
public class CategoriasController {
   @Autowired
   private CategoriasRepository categoriasRepository;
   @Autowired
   private CategoriasServices categoriasServices;
   @PostMapping (value = "/")
   @ResponseBody
    public ResponseEntity <CategoriasModel> agregar (@RequestBody CategoriasModel categoriasModel){
        System.out.println(categoriasModel);
        CategoriasModel obj = categoriasServices.save(categoriasModel);
        return new ResponseEntity<>(obj, HttpStatus.OK);
    @DeleteMapping (value = "/{id}")
    public ResponseEntity <CategoriasModel> eliminar (@PathVariable Integer id){
        CategoriasModel obj = categoriasServices.findById(id);
        if (obj != null){
           categoriasServices.delete(id);
            return new ResponseEntity<>(obi, HttpStatus.INTERNAL SERVER ERROR);
```



```
    return new ResponseEntity<>(obj, HttpStatus.OK);
}

@PutMapping (value = "/")
public ResponseEntity <CategoriasModel> editar (@RequestBody CategoriasModel categoriasModel){
    CategoriasModel obj = categoriasServices.findById(categoriasModel.get_id());
    if(obj != null){
        obj.setCategory_name(categoriasModel.getCategory_name());
        categoriasServices.save(obj);
    } else {
        return new ResponseEntity<>(obj, HttpStatus.INTERNAL_SERVER_ERROR);
}
    return new ResponseEntity<>(obj, HttpStatus.OK);
}

@GetMapping ("/list")
public List <CategoriasModel> consultarTodo(){
        return categoriasServices.findAll();
}

@GetMapping ("/list/{id}")
public CategoriasModel consultarPorId(@PathVariable Integer id){
        return categoriasServices.findById(id);
}
```

Código JAVA Productos Controller

```
package com.wmessence.wmessence. ontrollers;
import com.wmessence.wmessence.Models.ProductosModel;
import com.wmessence.wmessence.Services.ProductosServices;
import com.wmessence.wmessence.Repositories.ProductosRepository;
import java.util.List;
import org.springframework.web.bind.annotation.*;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.beans.factory.annotation.Autowired;
```







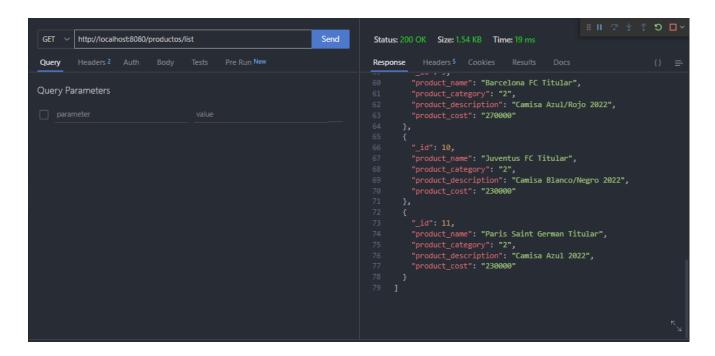
```
@RestController
@CrossOrigin ("*")
@RequestMapping ("/productos")
public class ProductosController {
   @Autowired
   private ProductosRepository productosRepository;
   @Autowired
   private ProductosServices productosServices;
   @PostMapping (value = "/")
    @ResponseBody
   public ResponseEntity<ProductosModel> agregar (@RequestBody ProductosModel productosModel){
        System.out.println(productosModel);
        ProductosModel obj = productosServices.save(productosModel);
        return new ResponseEntity<>(obj, HttpStatus.OK);
    @DeleteMapping (value = "/{id}")
    public ResponseEntity<ProductosModel> eliminar (@PathVariable Integer id){
        ProductosModel obj = productosServices.findById(id);
        if (obj != null){
           productosServices.delete(id);
            return new ResponseEntity<>(obj, HttpStatus.INTERNAL_SERVER_ERROR);
        return new ResponseEntity<>(obj, HttpStatus.OK);
   @PutMapping (value = "/")
    public ResponseEntity<ProductosModel> editar (@RequestBody ProductosModel productosModel){
        ProductosModel obj = productosServices.findById(productosModel.get_id());
        if (obj != null){
           obj.setProduct name(productosModel.getProduct name());
           obj.setProduct_description(productosModel.getProduct_description());
            obj.setProduct category(productosModel.getProduct category());
            obj.setProduct_cost(productosModel.getProduct_cost());
           productosServices.save(obj);
            return new ResponseEntity<>(obj, HttpStatus.INTERNAL_SERVER_ERROR);
        return new ResponseEntity<>(obj, HttpStatus.OK);
```



```
@GetMapping ("/list")
public List<ProductosModel> consultarTodo(){
    return productosServices.findAll();
}
@GetMapping ("/list/{id}")
public ProductosModel consultarPorId (@PathVariable Integer id){
    return productosServices.findById(id);
}
@GetMapping ("/consulta")
@ResponseBody
public List <ProductosModel> qry_product_category(@RequestParam ("qpc") Integer qpc){
    return productosServices.qry_product_category(qpc);
}
```

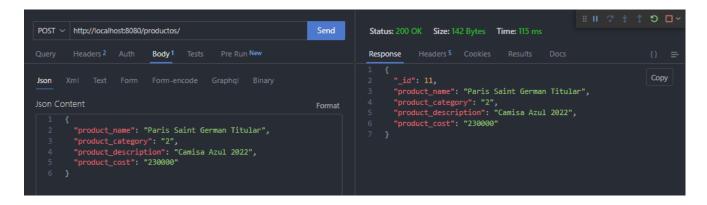
Evidencias de los "endpoint" con el consumo de recursos del API REST

Método GET en la función LIST de la tabla Productos.

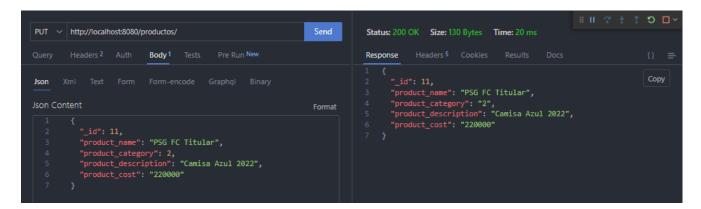




Método POST para AGREGAR ítems a la tabla Productos.



Método PUT para EDITAR ítems de la tabla Productos.

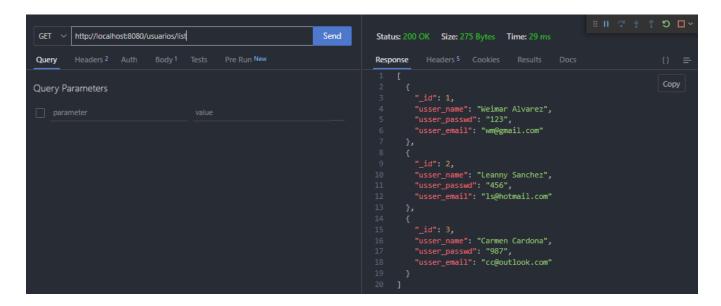


Método GET en la función LIST de la tabla Categorías.

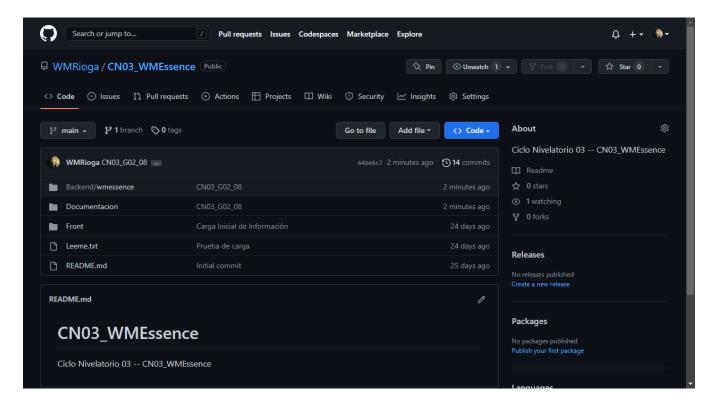




Método GET en la función LIST de la tabla Usuarios.



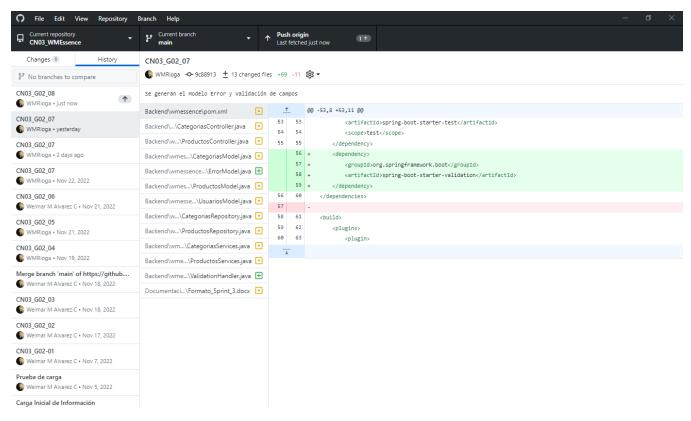
Evidencia GitLab o GitHub











Evidencia JIRA (Seguimiento del proyecto)

