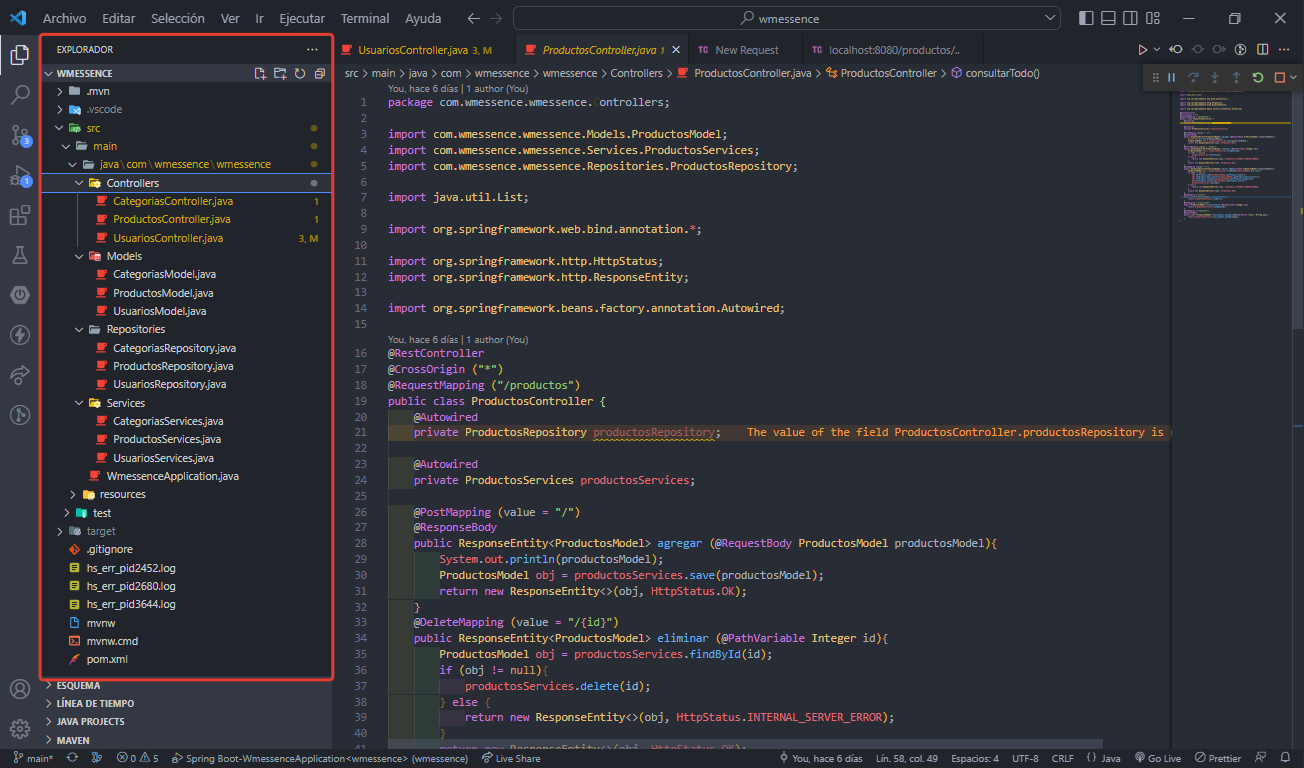
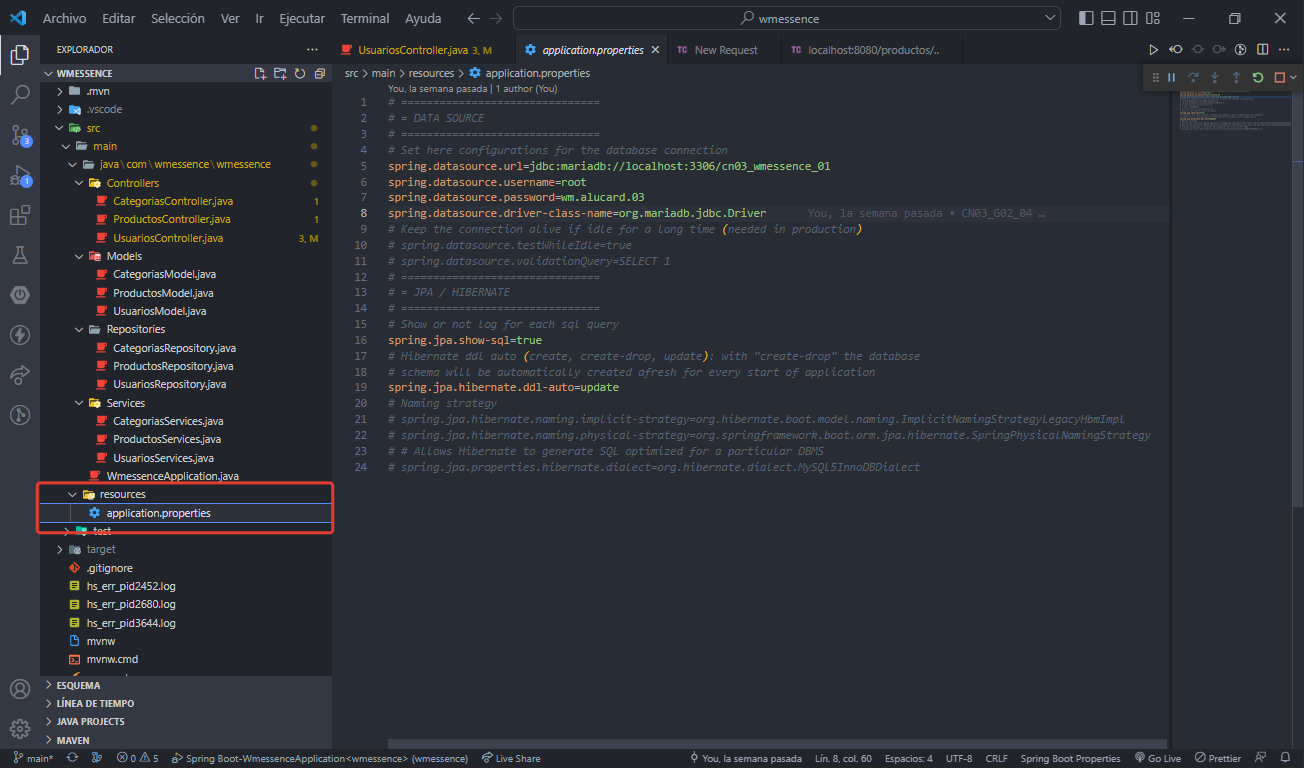
|  |
| --- |
| **SPRINT 3: Desarrollo del Backend** |

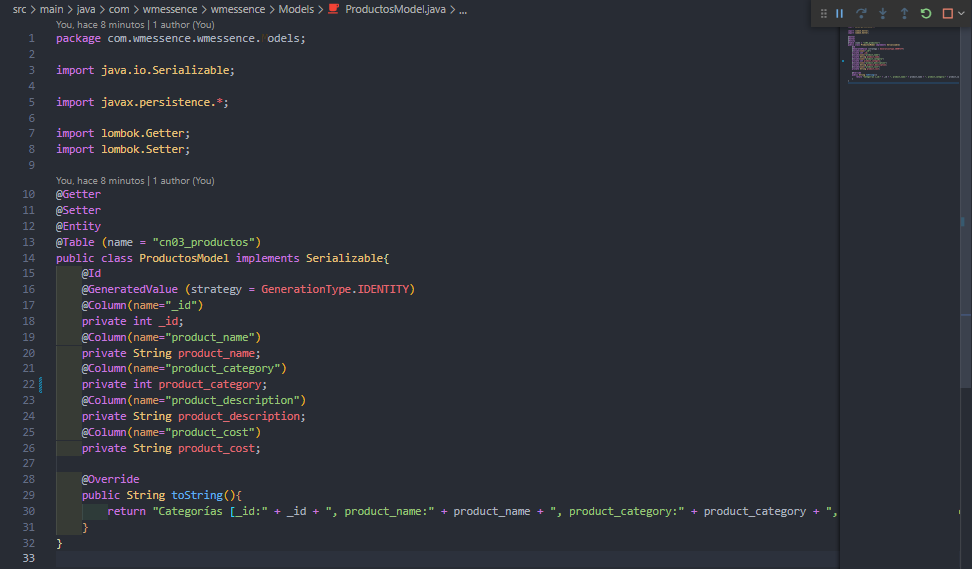
|  |  |
| --- | --- |
| **Identificación Proyecto** | |
| Nombre Proyecto: | Vaper Shop App |
| Número Equipo: |  |
| **Integrantes del equipo** | |
| Rol  (Líder-Desarrollador – Cliente) | Nombre |
| (Líder-Desarrollador – Cliente) | Weimar Moisés Álvarez Cardona |
| (Líder-Desarrollador – Cliente)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

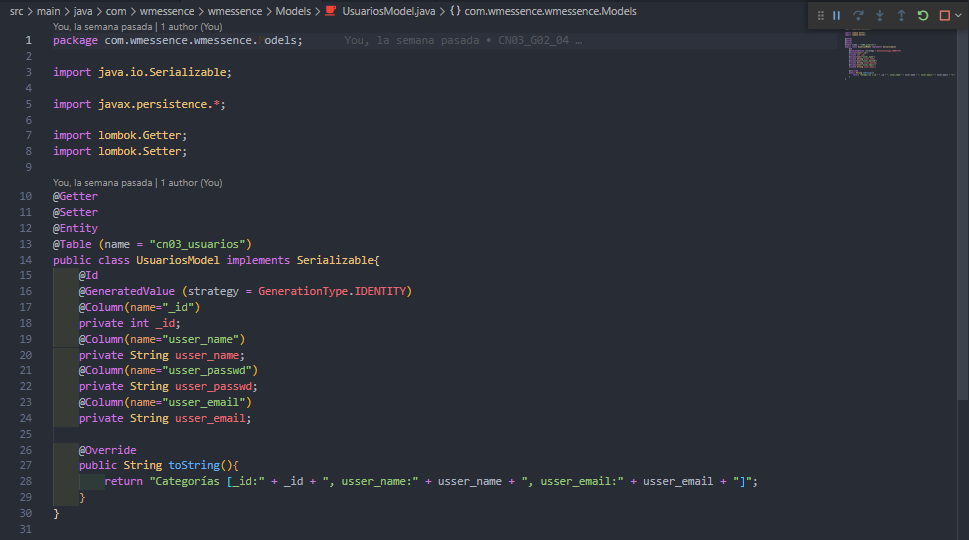
|  |
| --- |
| **Evidencia construcción del Backend** |

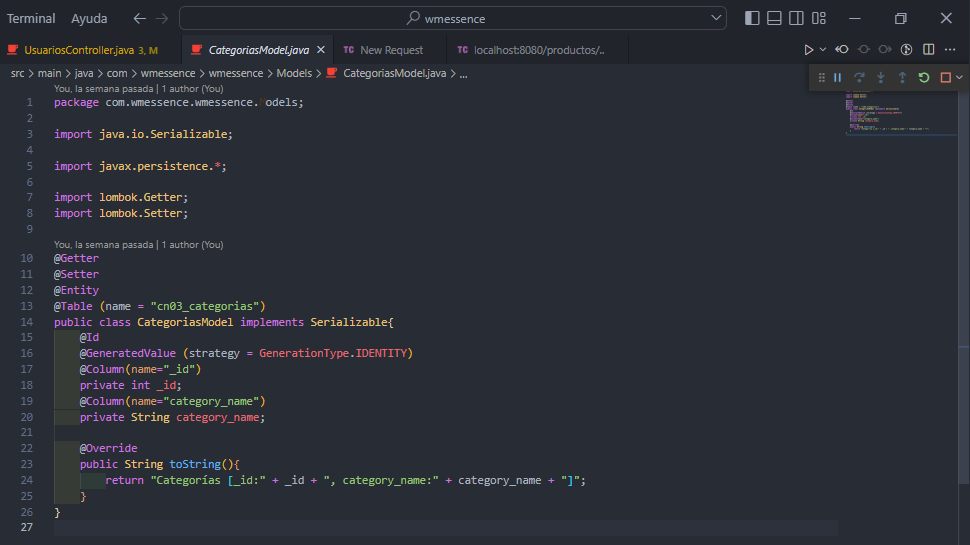
Construcción de Backend con Spring-boot y Visual Studio











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);

        } else {

            return new ResponseEntity<>(obj, 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.Controllers;

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);

        } else {

            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);

        } else {

            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){

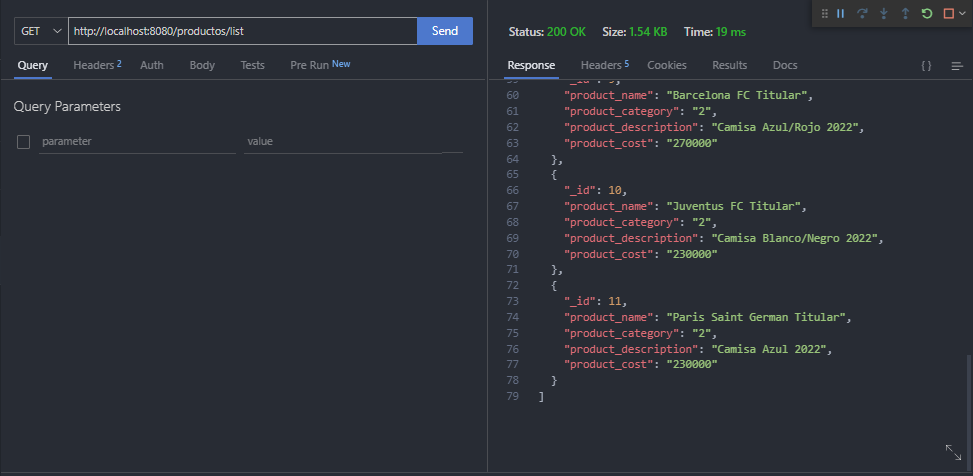
        return productosServices.qry\_product\_category(qpc);

    }

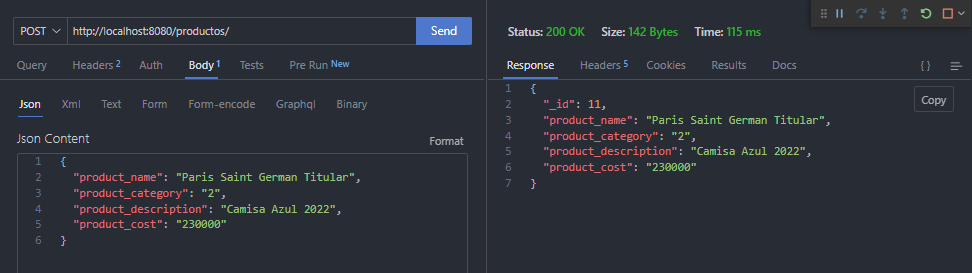
}

|  |
| --- |
| **Evidencias de los “endpoint” con el consumo de recursos del API REST** |

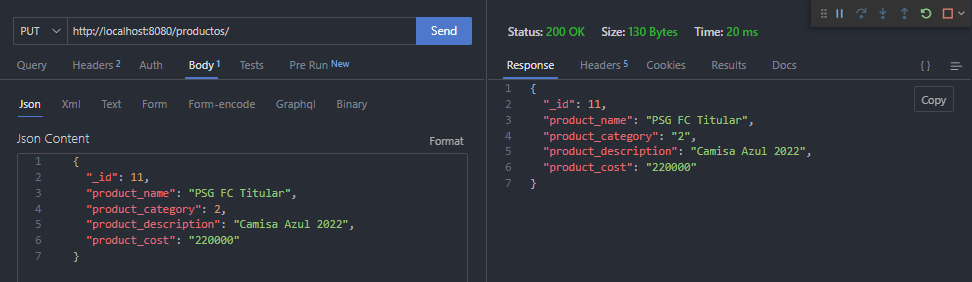
Método GET en la función LIST de la tabla Productos.



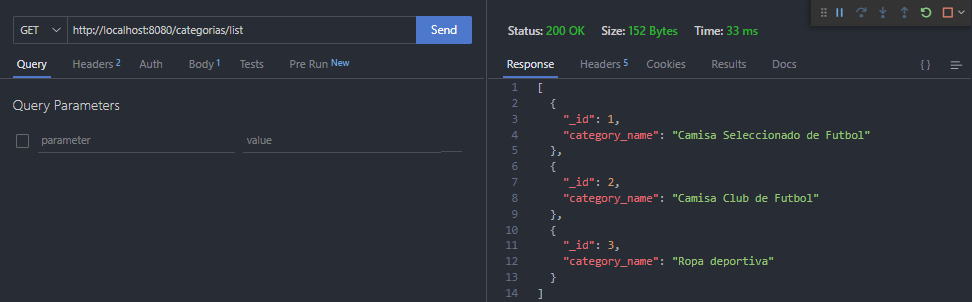
Método POST para AGREGAR ítems a la tabla Productos.



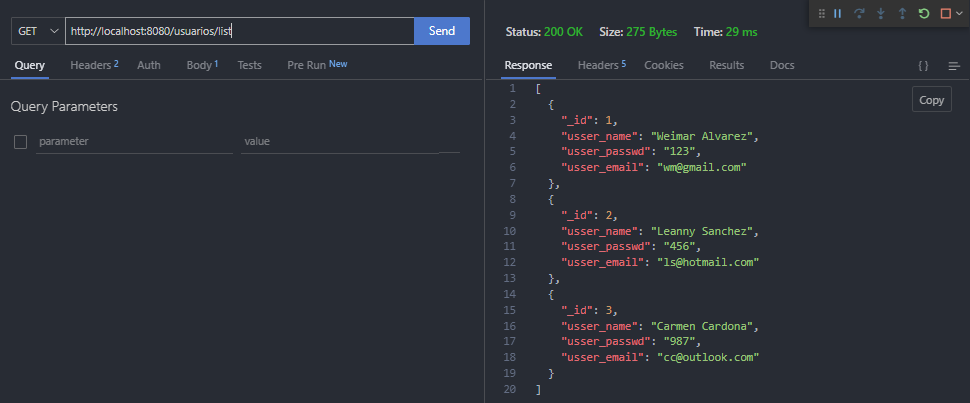
Método PUT para EDITAR ítems de la tabla Productos.



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 de la realización de alguna actualización (commit), donde se visualice la actualización y el historial de actualizaciones (Versión)

Enlace a github (achivos html e imagenes)

|  |
| --- |
| Evidencia JIRA (Seguimiento del proyecto) |

Como evidencia del seguimiento del proyecto con la metodología ágil SCRUM, utilizando el software JIRA, se debe presentar capturas de pantalla donde se visualice la ejecución de los Sprint con las historias de usuario relacionadas con el desarrollo del Backend.

|  |
| --- |
| Evidencias de las Reuniones de Equipo |

Como evidencia de las reuniones que efectúa el equipo del proyecto, presentar capturas de pantalla de las reuniones efectuadas y si lo consideran pertinente algunas actas de las reuniones.