

# **Especificación de Requerimientos**

## **Asociación de agricultores**

### **Plaza Campesina**

#### **La Plaza**

## Table de Contenido

1.	Tabla 1. Nombres, correo electrónico y rol del equipo de trabajo SCRUM. ....	3
2.	Tabla 2. Especificaciones del proyecto SCRUM. ....	4
3.	Tabla 3. Especificaciones de historias de usuario. ....	5
4.	Proceso de comercialización: .....	7
5.	Definición caso de uso:.....	7
6.	Diagrama de secuencia-ejemplo-acceso y petición administrador: .....	8
7.	Diagrama de entidad relación: .....	9
8.	Mockups.....	10
9.	Tablas de Bases de Datos .....	11
10.	Conexión con base de datos.....	12
11.	Creación de Base de Datos. ....	13
12.	Conexión con la base de datos .....	13
13.	Crear Producto .....	14
14.	Actualizar Producto .....	19
15.	Borrar Producto .....	22
16.	Listar Producto .....	25
17.	DESPLIEGUE.....	26
18.	FUNCIONAMIENTO APIS.....	31
19.	INFORME DE RETROSPECTIVA:.....	36

**1. Tabla 1. Nombres, correo electrónico y rol del equipo de trabajo SCRUM.**

Apellidos, Nombres	Correo electrónico	Rol
Cagua Carrillo Sayda Yamile	saydayamilecaguac@gmail.com	Gestor de Proyecto
José Daniel Ramírez Saldaña	Jose206@utp.edu.co	Gestor Base de Datos
Julián Andrés Segura González	julian888s@gmail.com	Desarrollador Frontend
Dora Paola Pacheco	paolapacheco.moreno@gmail.com	Desarrollador Backend
Isis Nirvana Segura Valero	isiissegura@gmail.com	Tester

## 2. Tabla 2. Especificaciones del proyecto SCRUM.

<b>CATEGORÍA</b>	Agro
<b>NOMBRE</b>	Plaza campesina (Asociación de Agricultores)
<b>DESCRIPCIÓN</b>	Las asociaciones agrícolas desempeñan un papel importante para apoyar a los pequeños productores; hombres y mujeres, y grupos marginados. Ofreciendo oportunidades de mercado a servicios como una mejor gestión. La asociatividad es un mecanismo de cooperación entre campesinos que trabajan por un bien común, y permite disminuir costos, acceder a tecnología de punta, acrecentar el poder de negociación y dar estabilidad a los precios de ventas, lo que se ve reflejado en un aumento de la rentabilidad.
<b>OBJETIVO ESTRATÉGICO</b>	Crear un software como apoyo a pequeños productores, con el fin de establecer los precios unitarios de sus productos agrícolas, mejorar su margen de venta y rentabilidad.
<b>PÚBLICO OBJETIVO</b>	Productores y/o campesinos
<b>IMPACTO ESPERADO</b>	Mayores ingresos de los campesinos.

A continuación, se presentan las cartas de información que complementan las historias de usuarios del proyecto: Conexión agro con empresas:

### 3. Tabla 3. Especificaciones de historias de usuario.

Historias de usuario	
Número: 1	Nombre: CREAR PRODUCTO(Crear)
Puntos Estimados:	
Descripción: El líder de los productores, necesita crear un producto, con la finalidad de comercializar su producto en el sistema.	
Criterios de aceptación:	
El producto debe quedar registrado con un nombre real y único, ya que podrá ser más fácil su ubicación por otros usuarios.	
Para la creación de un producto se debe tener en cuenta el, Nombre del producto, Descripción del producto (Tipo de producto, Calidad/Estado, Tamaño, Cantidad, Precio) y el Productor que tenga la disposición del producto.	

Historias de usuario	
Número: 2	Nombre: LISTAR PRODUCTOS
Puntos Estimados:	
Descripción: El líder de los productores necesita ver la lista de productos creados en el sistema.	
Criterios de aceptación:	
El Productor puede validar la lista de productos añadidos, así mismo validar la información agregada, a su vez permite la actualización y eliminación de cada producto.	

Historias de usuario	
Número: 3	Nombre: ACTUALIZAR PRODUCTO (Editar)
Puntos Estimados:	
Descripción: El líder de los productores, necesita modificar su producto creado, con la finalidad de poder realizar algún cambio o actualización en el sistema.	
Criterios de aceptación:	
El líder de los productores debe actualizar el producto con los requerimientos solicitados por el sistema y de forma clara y concisa, de lo contrario el registro no será válido y/o almacenado en el sistema.	
Para la actualización de un producto se debe tener en cuenta el, Nombre del producto, Descripción del producto (Tipo de producto, Calidad/Estado, Tamaño, Cantidad, Precio) y el Productor que tenga la disposición del producto.	

Historias de usuario	
Número: 4	Nombre: BORRAR PRODUCTO (Eliminar)
Puntos Estimados:	
Descripción: El líder de los productores, necesita eliminar un producto, con la finalidad de poderlo borrar del sistema.	
Criterios de aceptación: El líder de los productores debe tener en claro cuál es el producto que desea eliminar, debe eliminar el producto que no desea seguir ofreciendo y no cuenta con disponibilidad.	



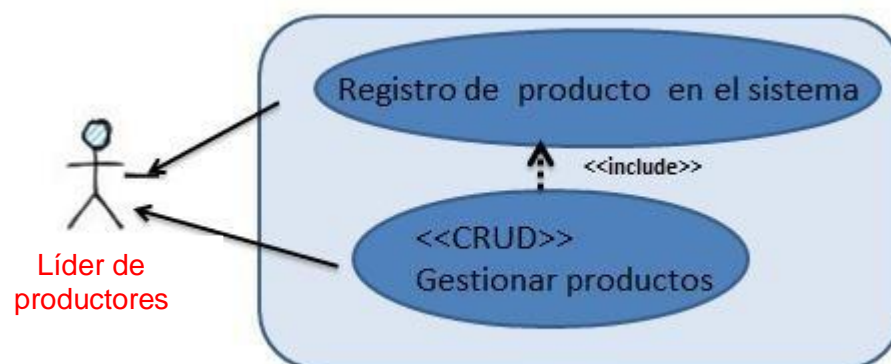
Este software tendrá el siguiente modulo;

- Módulo de productos

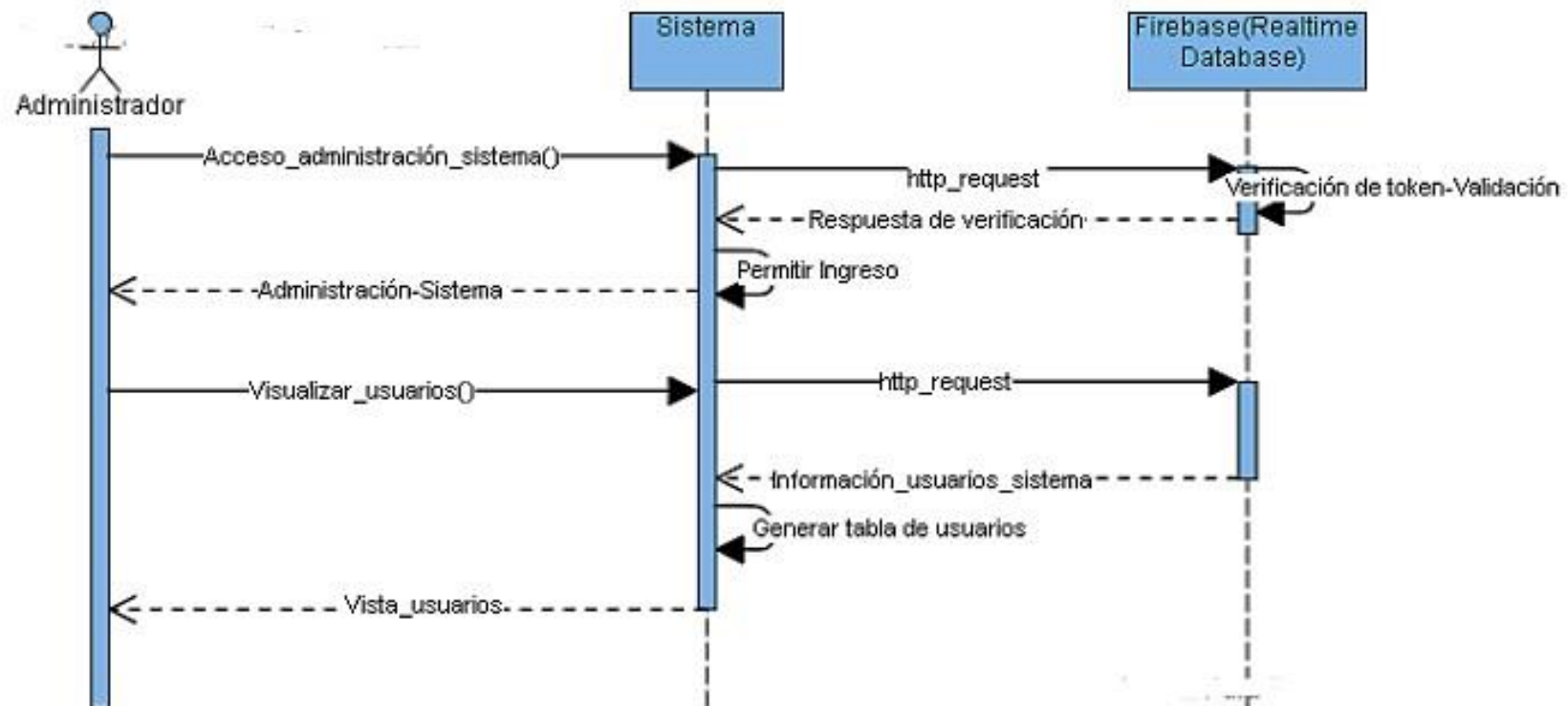
#### 4. Proceso de comercialización:

PRODUTOR —————> PRODUCTO

#### 5. Definición caso de uso:

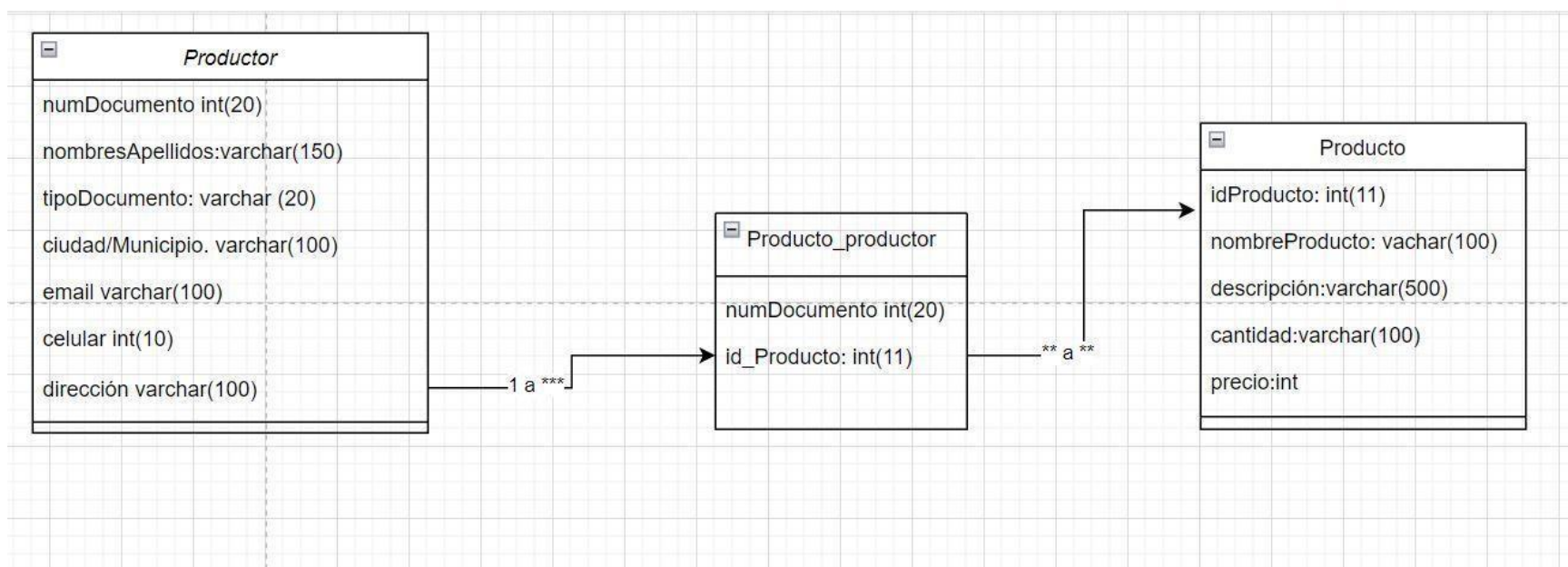


## 6. Diagrama de secuencia-ejemplo-acceso y petición administrador:

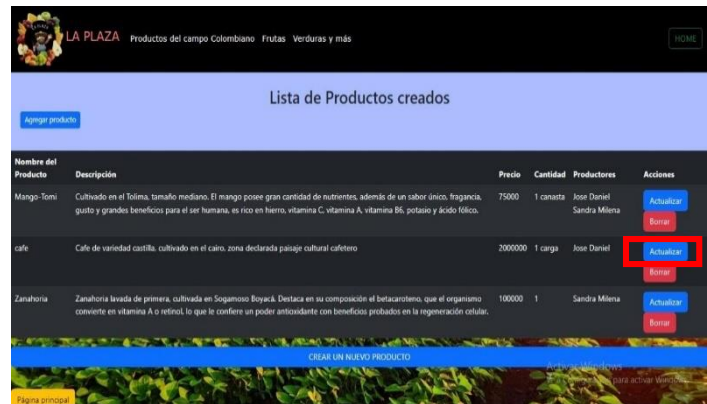
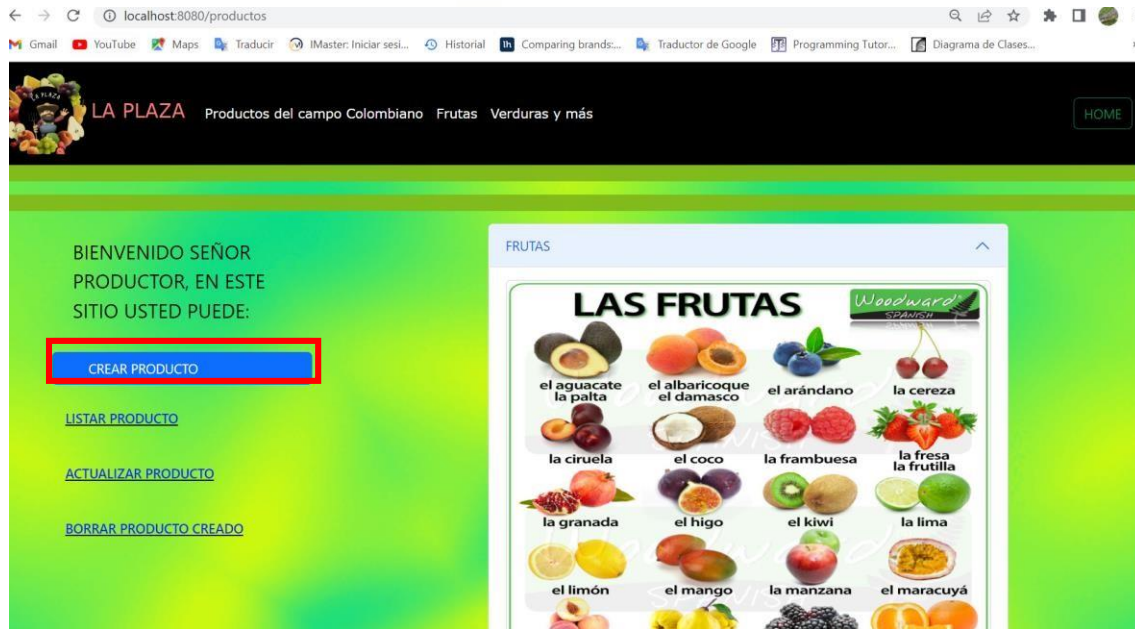




## 7. Diagrama de entidad relación:



## 8. Mockups



## 9. Tablas de Bases de Datos

phpMyAdmin

Reciente Favoritas

- Nueva
- information\_schema
- la\_plaza
  - Nueva
  - producto
  - tipo\_usuario
  - usuario
  - usuario\_productor
- mysql
- performance\_schema
- phpmyadmin
- test
- tipo\_usuario

Base de datos: la\_plaza » Tabla: producto

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	idProducto	int(11)			No	Ninguna		AUTO_INCREMENT	<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
2	nomProducto	varchar(100)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
3	descripcion	varchar(500)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
4	cantidad	varchar(100)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
5	precio	int(100)			No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
6	insertarFoto	varchar(1)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>

☐ Seleccionar todo Para los elementos que están marcados: [Examinar](#) [Cambiar](#) [Eliminar](#) [Primaria](#) [Único](#) [Índice](#)  
[Espacial](#) [Texto completo](#) [Agregar a columnas centrales](#) [Eliminar de las columnas centrales](#)

[Imprimir](#) [Planteamiento de la estructura de tabla](#) [Hacer seguimiento a la tabla](#) [Mover columnas](#) [Normalizar](#)  
 Agregar 1 columna(s) después de insertarFoto [Continuar](#)

Índices

Ilustración 1. Tabla de Producto.

phpMyAdmin

Reciente Favoritas

- Nueva
- information\_schema
- la\_plaza
  - Nueva
  - producto
  - tipo\_usuario
  - usuario
  - usuario\_productor
- mysql
- performance\_schema
- phpmyadmin
- test
- tipo\_usuario

Base de datos: la\_plaza » Tabla: usuario

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id_tipo	int(11)			No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
2	nombres_apellidos	varchar(150)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
3	tipo_documento	varchar(20)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
4	numDocumento	int(20)			No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
5	ciudad_municipio	varchar(100)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
6	direccion	varchar(100)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
7	email	varchar(100)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
8	celular	varchar(11)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>
9	contraseña	varchar(30)	utf8mb4_general_ci		No	Ninguna			<a href="#">Cambiar</a> <a href="#">Eliminar</a> <a href="#">Más</a>

☐ Seleccionar todo Para los elementos que están marcados: [Examinar](#) [Cambiar](#) [Eliminar](#) [Primaria](#) [Único](#) [Índice](#)  
[Espacial](#) [Texto completo](#) [Agregar a columnas centrales](#) [Eliminar de las columnas centrales](#)

Ilustración 2. Tabla Usuario



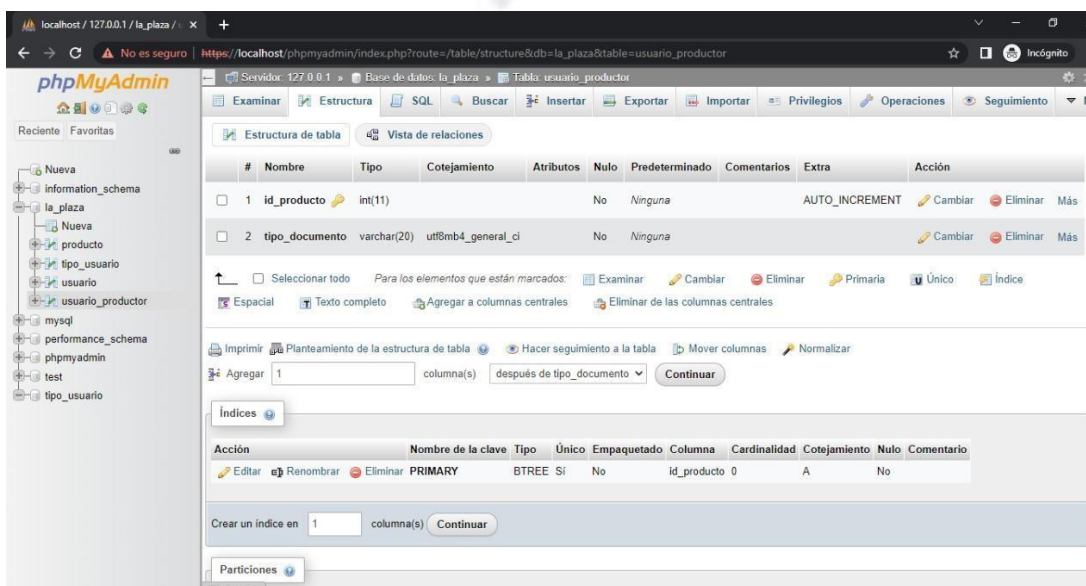
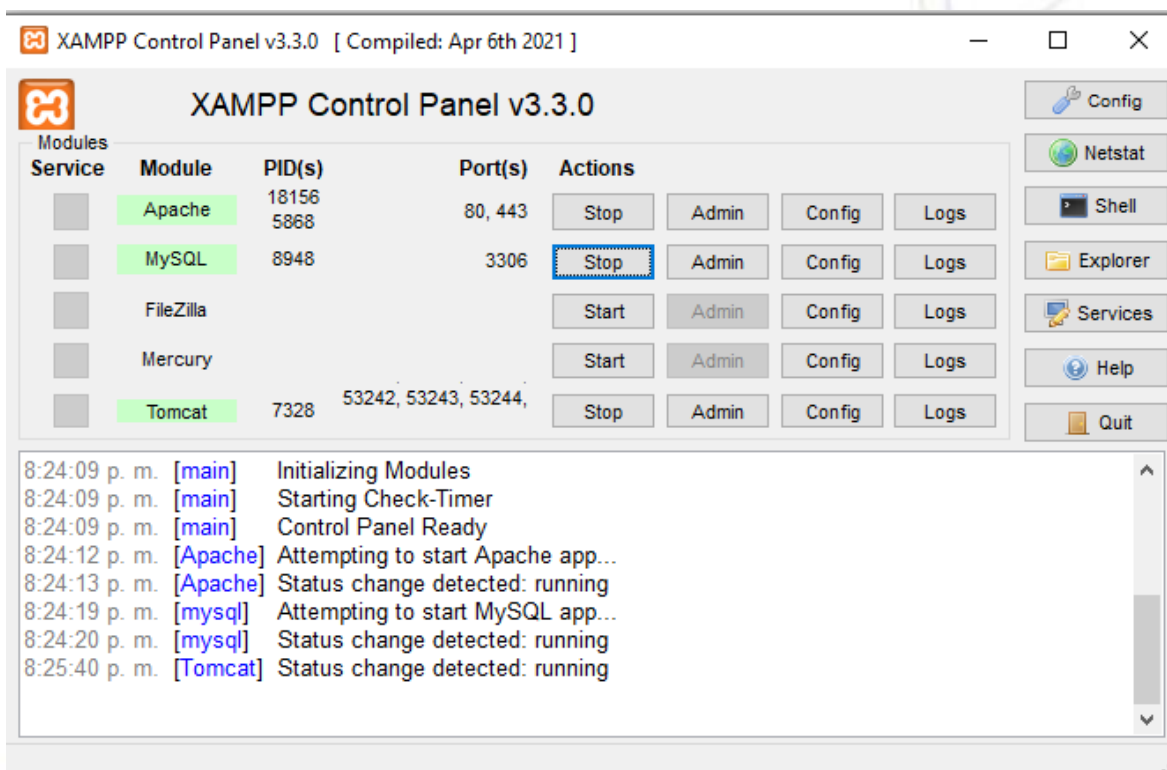


Ilustración 3. Tabla Usuario Productor

## 10. Conexión con base de datos.

Es necesario establecer una conexión con la base de datos donde se almacena la información. Para el desarrollo del proyecto La Plaza, se utiliza un servidor local para emular una base de datos. Esto se realiza a través de la aplicación XAMP.



## 11. Creación de Base de Datos.

Para crear la base de datos, se utiliza lenguaje SQL, a través del siguiente lenguaje:

CREATE DATABASE

Crea la tabla para producto en lenguaje SQL:

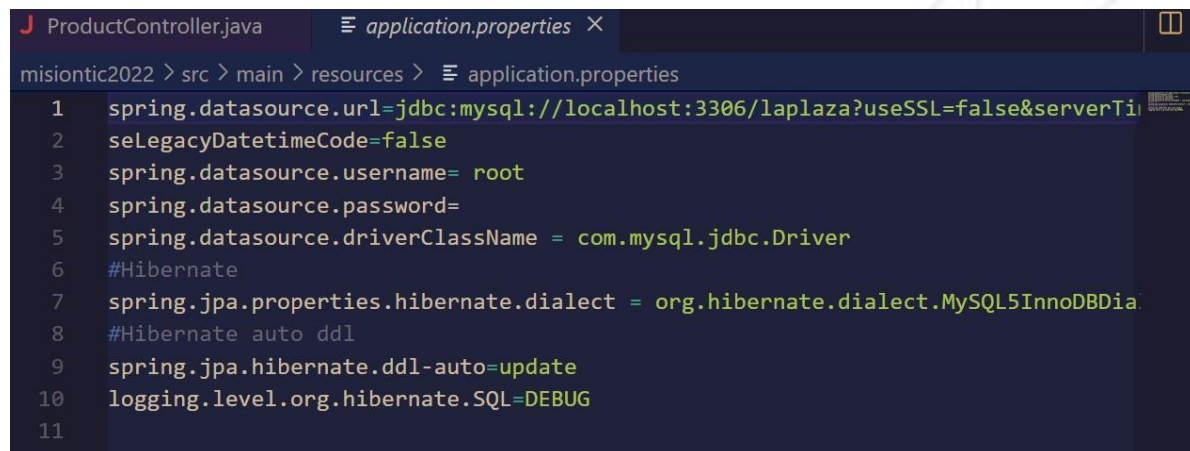
```
CREATE TABLE producto (
  id_producto INT PRIMARY KEY,
  cantidad VARCHAR(255) NOT NULL,
  descripcion VARCHAR(255) NOT NULL,
  nombre_producto VARCHAR(255) NOT NULL,
  precio INT NOT NULL
)
```

Crear Tabla Productor

```
CREATE TABLE productor (
  cedula INT PRIMARY KEY,
  id_productor INT,
  nombre_productor VARCHAR(100) NOT NULL,
  municipio VARCHAR(100) NOT NULL,
  correo VARCHAR(100) NOT NULL,
  celular VARCHAR(100) NOT NULL,
)
```

## 12. Conexión con la base de datos

La configuración se realiza a través de fichero application.properties, el cual se crea al ejecutar el proyecto Spring Bot.



```
ProductController.java  application.properties X
misiontic2022 > src > main > resources > application.properties
1  spring.datasource.url=jdbc:mysql://localhost:3306/laplaza?useSSL=false&serverTi
2  seLegacyDatetimeCode=false
3  spring.datasource.username= root
4  spring.datasource.password=
5  spring.datasource.driverClassName = com.mysql.jdbc.Driver
6  #Hibernate
7  spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDia
8  #Hibernate auto ddl
9  spring.jpa.hibernate.ddl-auto=update
10 logging.level.org.hibernate.SQL=DEBUG
11
```

Se crea html index donde se reciben las peticiones del usuario posterior los html de cada uno de las historias de usuarios o acciones que se requieren.

### 13. Crear Producto

```

<> CrearProducto.html ×
misiontic2022 > src > main > resources > templates > <> CrearProducto.html > html > body > body > table > tr >
268 <div class="container">
269 <div class="row">
270 <div class="col-sm">
271 <form th:action="@{/products}" th:object="${product}" method="POST">
272

```

Para poder mostrar el resultado de la implementación del método, se necesita crear archivo HTML.

En este archivo tenemos que agregar el thymeleaf, el cual realiza una query para implementar la consulta a la base de datos.

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.com">
<head>

```

Es importante llamar a la dependencia thymeleaf a la hora de crear la tabla en HTML, para poder incorporar elementos dinámicos en el fichero HTML sin tener que crear un archivo javascript.

```

J ProductController.java ×
htic2022 > src > main > java > misiontic2022 > com > laplaza > controller > J ProductC
49
50 @GetMapping("/products/new")
51 public String createProductForm(Model model) {
52
53     Product product = new Product();
54
55     model.addAttribute(attributeName: "product", product);
56     model.addAttribute(attributeName: "producersList", producersList);
57
58     return "CrearProducto";
59 }
60

```

El método (createProductForm), debe retornar un String que va a ser el nombre de la vista (la página "CrearProducto") la cual nos va a permitir crear nuevos productos.



Se genera la llamada de producto por Thymeleaf desde html al controlador, esta petición la recibe el controller, el cual va implementar el método a través de la interface del servicio, a su vez implementa el servicio.

```
J IProductService.java X
misiontic2022 > src > main > java > misiontic2022 > com > laplaza > Service > J IPr
1 package misiontic2022.com.laplaza.Service;
2
3 import java.util.List;
4
5 import misiontic2022.com.laplaza.Entity.Product;
6
7 public interface IProductService {
8
9     List<Product> getAllProduct();
10
11     List<Product> getProductByName(String product_name);
12
13     Product saveProduct(Product product);
14
15     Product getProductById(Long id);
16
17     Product updateProduct(Product product);
18     Product actualizar(Product product);
19
20     void deleteProductById(Long id);
21
22 }
23
```

Se genera la interfaz para conexión con base de datos y se crea en la carpeta entidad los JAP de la tabla de producto.

```
J Product.java X
misiontic2022 > src > main > java > misiontic2022 > com > laplaza > Entity > J Prod
17 @Entity
18 @Table(name = "products")
19 public class Product {
20
21     @Id
22     @GeneratedValue
23     private Long id;
24
25     @Column(name = "product_name", nullable = false )
26     private String product_name;
27
28     @Column(name = "description")
29     private String description;
30
31     @Column(name = "price")
32     private int price;
33     @Column(name = "cantidad")
34     private String cantidad;
35
36
37     @ManyToMany(fetch = FetchType.LAZY, cascade = CascadeType.PERSIST)
38     @JoinTable(name = "products_producers", joinColumns = {
39         @JoinColumn(name = "product_id", referencedColumnName = "id", nullable = false,
40     }, inverseJoinColumns = {
41         @JoinColumn(name = "producer_id", referencedColumnName = "id", nullable = false,
42     })
43 }
```

Para poder obtener la información de un registro de la base de datos, se necesita implementar un método que se desarrolla en el backend.

```

J ProductService.java X
misiontic2022 > src > main > java > misiontic2022 > com > laplaza > Service > J Pro
1 package misiontic2022.com.laplaza.Service;
2
3 import java.util.List;
4
5 import org.springframework.stereotype.Service;
6
7 import misiontic2022.com.laplaza.Entity.Product;
8 import misiontic2022.com.laplaza.Repository.IProductRepository;
9
10 @Service
11 public class ProductService implements IProductService{
12
13     private IProductRepository productRepository;
14
15     public ProductService(IProductRepository productRepository){
16         this.productRepository=productRepository;
17     }
18
19     @Override
20     public List<Product> getAllProduct() {
21         return productRepository.findAll();
22     }
23
24     @Override
25     public List<Product> getProductByName(String product_name) {
26         return productRepository.findByNameContaining(product_name);
27     }

```

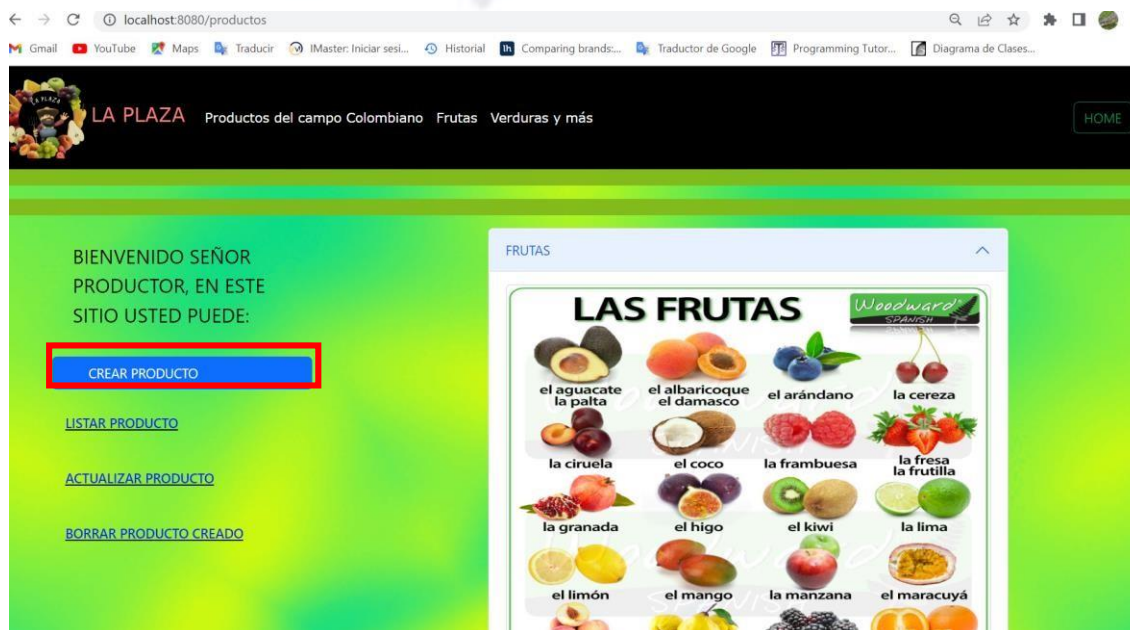
Se crea el repositorio de los valores de la tabla producto

```

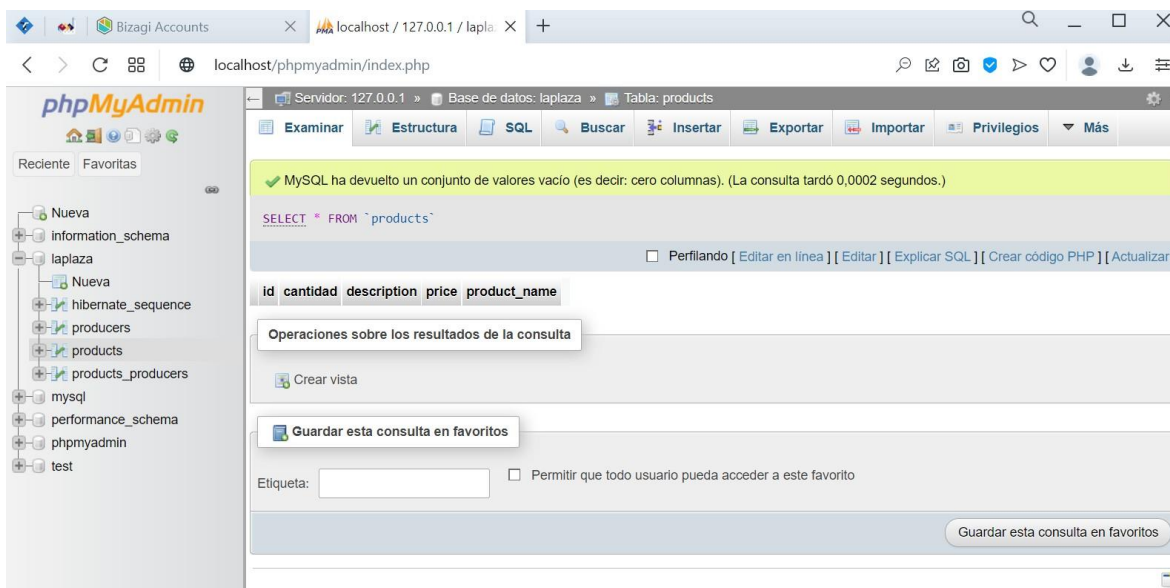
J IProductRepository.java X
misiontic2022 > src > main > java > misiontic2022 > com > laplaza > Repository > J
1 package misiontic2022.com.laplaza.Repository;
2
3 import java.util.List;
4
5 import org.springframework.data.jpa.repository.JpaRepository;
6 import org.springframework.data.jpa.repository.Query;
7 import org.springframework.data.repository.query.Param;
8
9 import misiontic2022.com.laplaza.Entity.Product;
10
11 public interface IProductRepository extends JpaRepository<Product, Long> {
12
13     @Query("FROM Product p WHERE p.product_name LIKE :name")
14     public List<Product> findByNameContaining(@Param("name") String name);
15
16 }

```

## Visualización fronted selección “CREAR PRODUCTO”



## Visualización base Xampp tabla products sin registros





Se Ingresan datos creación de producto por fronted

Se registra a la base de datos

localhost/phpmyadmin/index.php

Servidor: 127.0.0.1 » Base de datos: laplaza » Tabla: products

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Más

Mostrando filas 0 - 0 (total de 1, La consulta tardó 0,0002 segundos.)

SELECT \* FROM `products`

Perfilando [ Editar en línea ] [ Editar ] [ Explicar SQL ] [ Crear código PHP ] [ Ac

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

Opciones extra

	id	cantidad	description	price	product_name
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	11	1 bulto	Su bajo contenido en sodio e hidratos de carbono, ...	20000	Naranja Valenciana

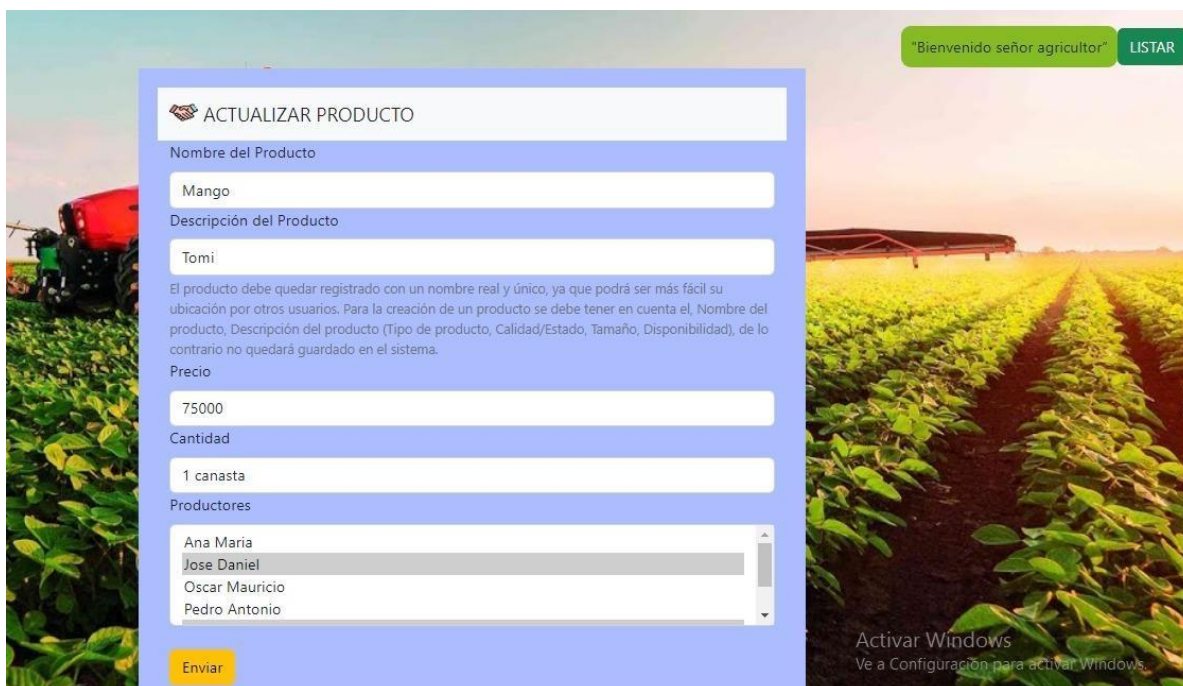
Seleccionar todo Para los elementos que están marcados: Editar Copiar Borrar Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

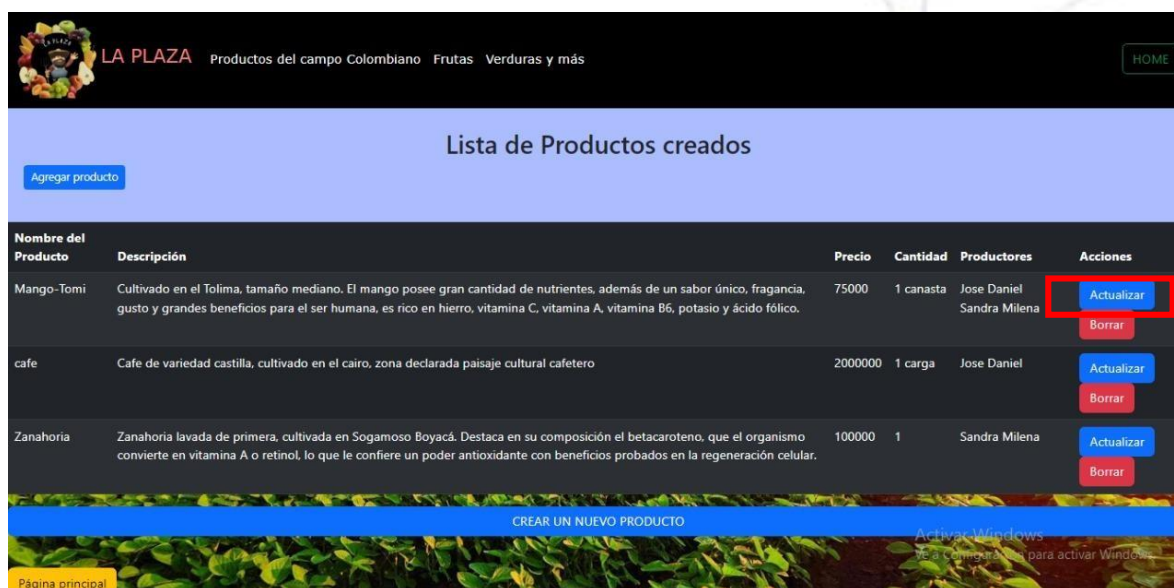
Operaciones sobre los resultados de la consulta

## 14. Actualizar Producto

Visualización fronted selección “ACTUALIZAR PRODUCTO”



Despliega html de “Lista de productos creados” allí seleccionamos el producto a

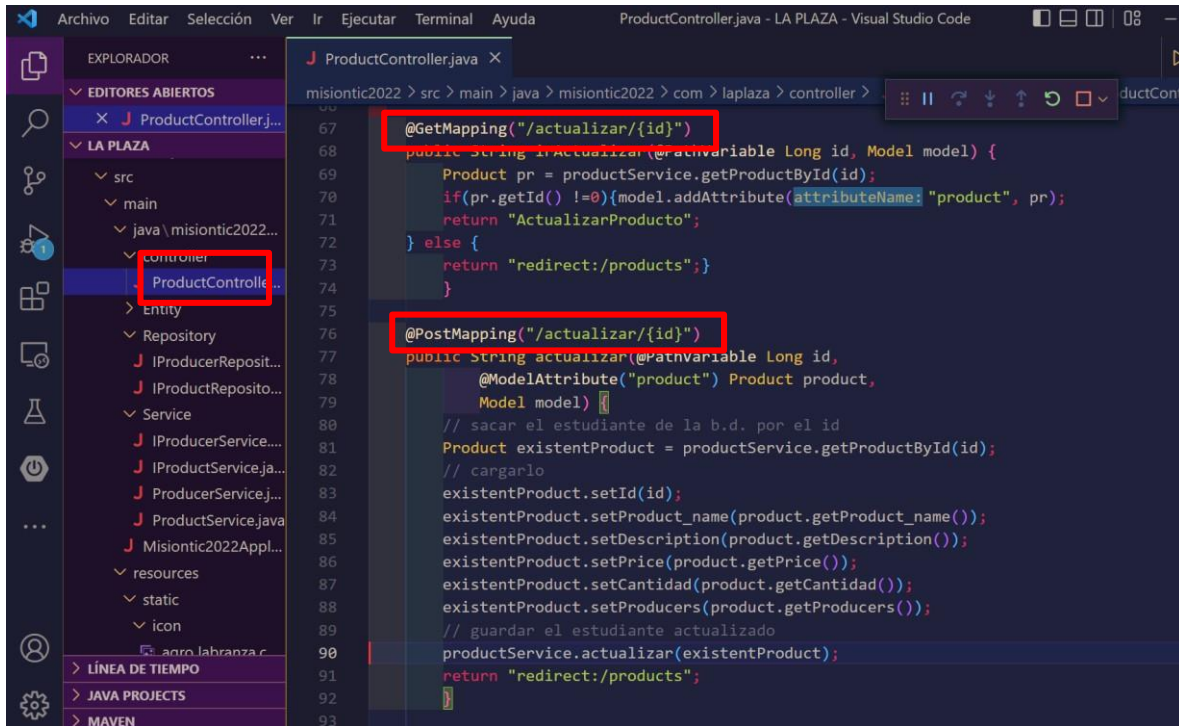


Nombre del Producto	Descripción	Precio	Cantidad	Productores	Acciones
Mango-Tomi	Cultivado en el Tolima, tamaño mediano. El mango posee gran cantidad de nutrientes, además de un sabor único, fragancia, gusto y grandes beneficios para el ser humano, es rico en hierro, vitamina C, vitamina A, vitamina B6, potasio y ácido fólico.	75000	1 canasta	Jose Daniel Sandra Milena	<a href="#">Actualizar</a> <a href="#">Borrar</a>
cafe	Cafe de variedad castilla, cultivado en el cairo, zona declarada paisaje cultural cafetero	2000000	1 carga	Jose Daniel	<a href="#">Actualizar</a> <a href="#">Borrar</a>
Zanahoria	Zanahoria lavada de primera, cultivada en Sogamoso Boyacá. Destaca en su composición el betacaroteno, que el organismo convierte en vitamina A o retinol, lo que le confiere un poder antioxidante con beneficios probados en la regeneración celular.	100000	1	Sandra Milena	<a href="#">Actualizar</a> <a href="#">Borrar</a>

modificar dando clic botón “Actualizar”

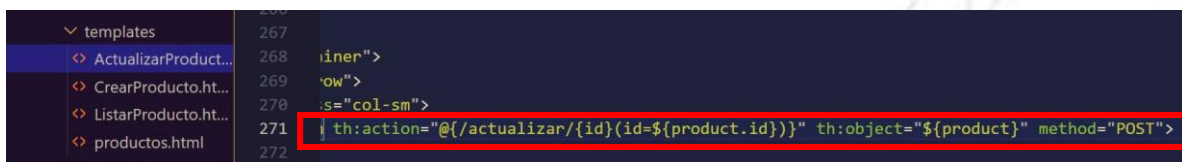


Para poder apreciar el diseño de nuestra siguiente página (ActualizarProducto.html), es necesario integrar el siguiente llamado a nuestra página. Para ello, nos vamos a nuestro único controlador "ProductController.java" y agregamos la siguiente anotación



```

67  @GetMapping("/actualizar/{id}")
68  public String actualizar(@PathVariable Long id, Model model) {
69      Product pr = productService.getProductById(id);
70      if(pr.getId() != 0){model.addAttribute("product", pr);
71      return "ActualizarProducto";
72      } else {
73      return "redirect:/products";
74      }
75
76  @PostMapping("/actualizar/{id}")
77  public String actualizar(@PathVariable Long id,
78      @ModelAttribute("product") Product product,
79      Model model) {
80      // sacar el estudiante de la b.d. por el id
81      Product existentProduct = productService.getProductById(id);
82      // cargarlo
83      existentProduct.setId(id);
84      existentProduct.setProduct_name(product.getProduct_name());
85      existentProduct.setDescription(product.getDescription());
86      existentProduct.setPrice(product.getPrice());
87      existentProduct.setCantidad(product.getCantidad());
88      existentProduct.setProducers(product.getProducers());
89      // guardar el estudiante actualizado
90      productService.actualizar(existentProduct);
91      return "redirect:/products";
92  }
93  
```



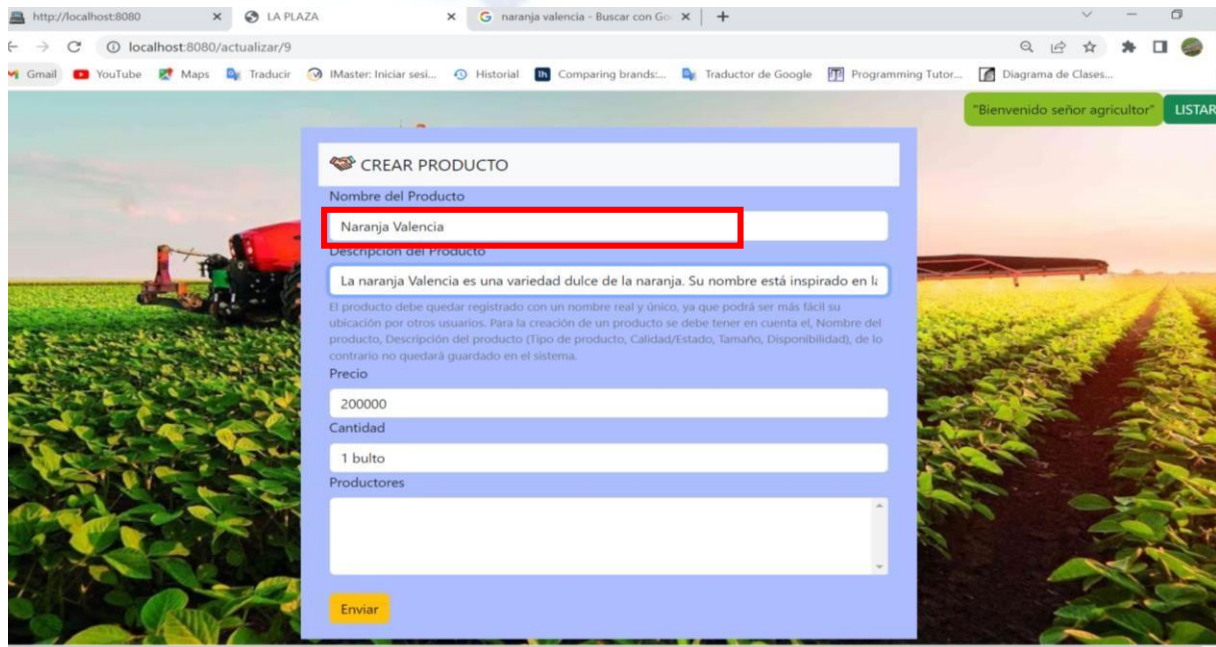
```

267  <div>
268      <div>
269          <div>
270              <div>
271                  <th:action="@{/actualizar/{id}}(id=${product.id})" th:object="${product}" method="POST">
272

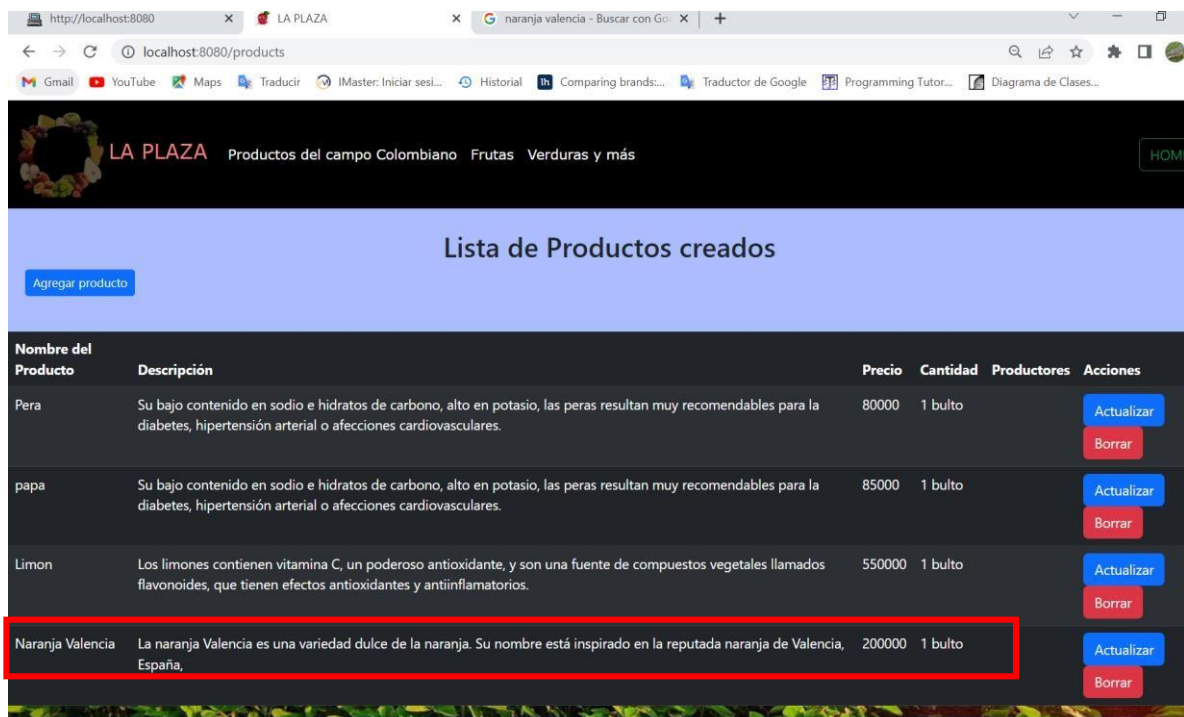
```

Extrayendo los datos de la base previamente ya registrados para poder modificarlos directamente en el front, al cambiarlos y dar clic botón "enviar"





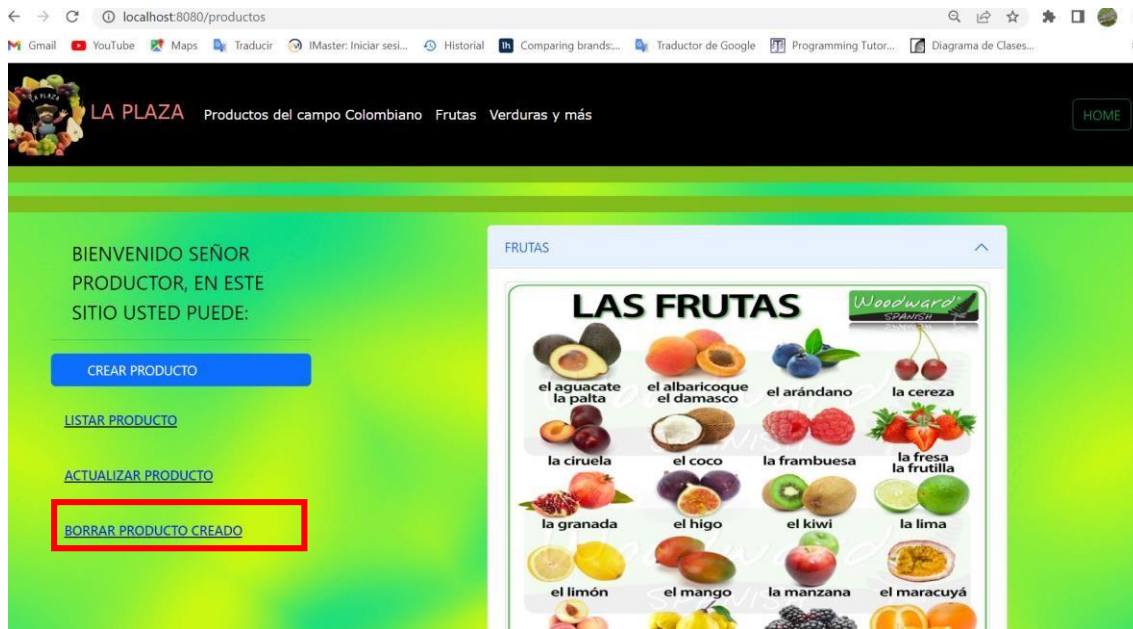
## Modifica el producto



Nombre del Producto	Descripción	Precio	Cantidad	Productores	Acciones
Pera	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	80000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
papa	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	85000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Limon	Los limones contienen vitamina C, un poderoso antioxidante, y son una fuente de compuestos vegetales llamados flavonoides, que tienen efectos antioxidantes y antiinflamatorios.	550000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Naranja Valencia	La naranja Valencia es una variedad dulce de la naranja. Su nombre está inspirado en la reputada naranja de Valencia, España,	200000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>

## 15. Borrar Producto

Visualización fronted selección “BORRAR PRODUCTO”



El método (deleteProduct), debe eliminar registro completo de la clase products a la cual va mostrar el cambio automático.

```
@GetMapping("/products/{id}")
public String deleteProduct(@PathVariable Long id) {
    productService.deleteProductById(id);
    return "redirect:/products";
}
```

Mostrando filas 0 - 2 (total de 3, La consulta tardó 0,0003 segundos.)

```
SELECT * FROM `products`
```

Perfilando [ Editar en línea ] [ Editar ] [ Explicar SQL ] [ Crear código PHP ] [ Actualizar ]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

	id	cantidad	description	price	product_name
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	1 bulto	mango tomy	100000	Mango
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	1 bulto	Su bajo contenido en sodio e hidratos de carbono, ...	80000	Pera
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	1 bulto	Los limones contienen vitamina C, un poderoso anti...	70000	Limon

Seleccionar todo | Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

Se verifican productos

LA PLAZA Productos del campo Colombiano Frutas Verduras y más

HOME

Agregar producto

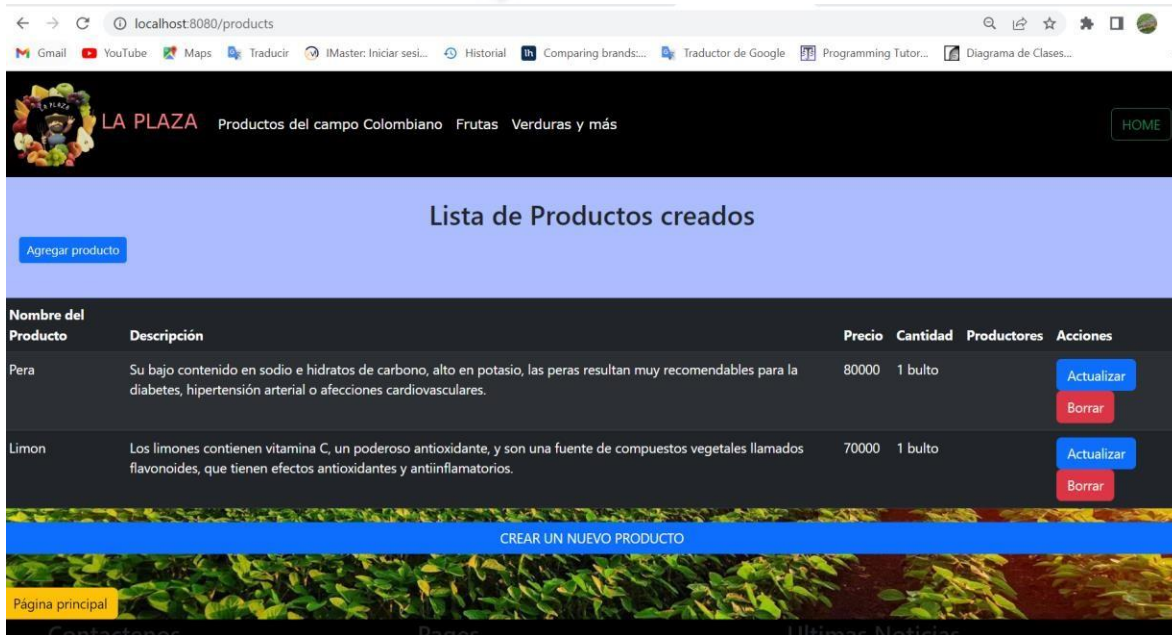
### Lista de Productos creados

Nombre del Producto	Descripción	Precio	Cantidad	Productores	Acciones
Mango	mango tomy	100000	1 bulto		<input type="button" value="Actualizar"/> <input type="button" value="Borrar"/>
Pera	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	80000	1 bulto		<input type="button" value="Actualizar"/> <input type="button" value="Borrar"/>
Limon	Los limones contienen vitamina C, un poderoso antioxidante, y son una fuente de compuestos vegetales llamados flavonoides, que tienen efectos antioxidantes y antiinflamatorios.	70000	1 bulto		<input type="button" value="Actualizar"/> <input type="button" value="Borrar"/>

CREAR UN NUEVO PRODUCTO



Al dar clic botón borrar se elimina el producto



LA PLAZA Productos del campo Colombiano Frutas Verduras y más

HOME

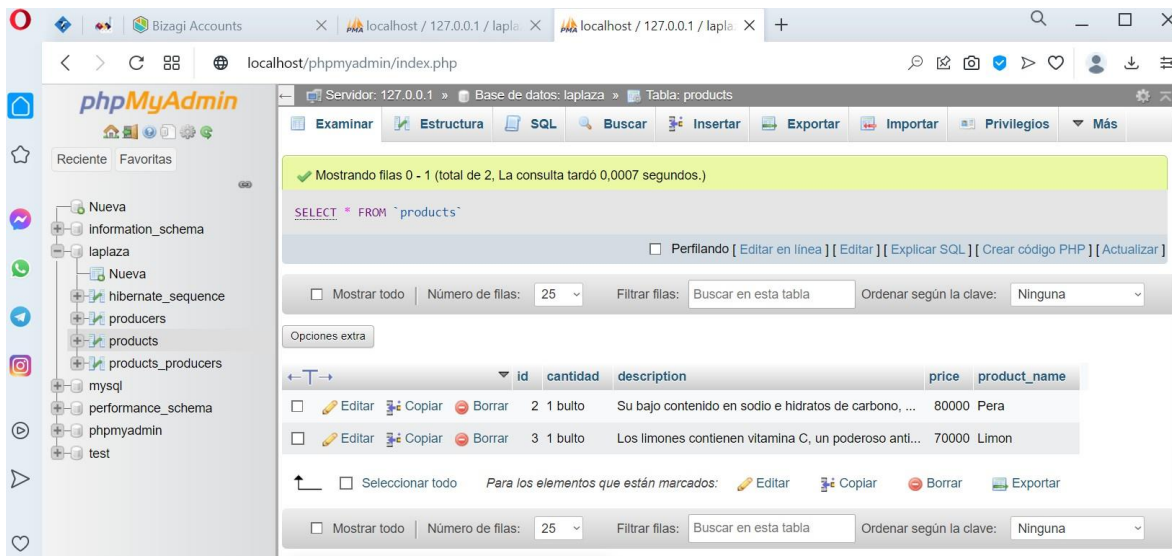
### Lista de Productos creados

Agregar producto

Nombre del Producto	Descripción	Precio	Cantidad	Productores	Acciones
Pera	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	80000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Limon	Los limones contienen vitamina C, un poderoso antioxidante, y son una fuente de compuestos vegetales llamados flavonoides, que tienen efectos antioxidantes y antiinflamatorios.	70000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>

CREAR UN NUEVO PRODUCTO

Página principal



phpMyAdmin

Reciente Favoritas

- Nueva
- Information\_schema
- laplaza
  - Nueva
  - hibernate\_sequence
  - producers
  - products
  - products\_producers
- mysql
- performance\_schema
- phpmyadmin
- test

Servidor: 127.0.0.1 » Base de datos: laplaza » Tabla: products

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Más

Mostrando filas 0 - 1 (total de 2, La consulta tardó 0,0007 segundos.)

SELECT \* FROM `products`

Perfilando [ Editar en línea ] [ Editar ] [ Explicar SQL ] [ Crear código PHP ] [ Actualizar ]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

Opciones extra

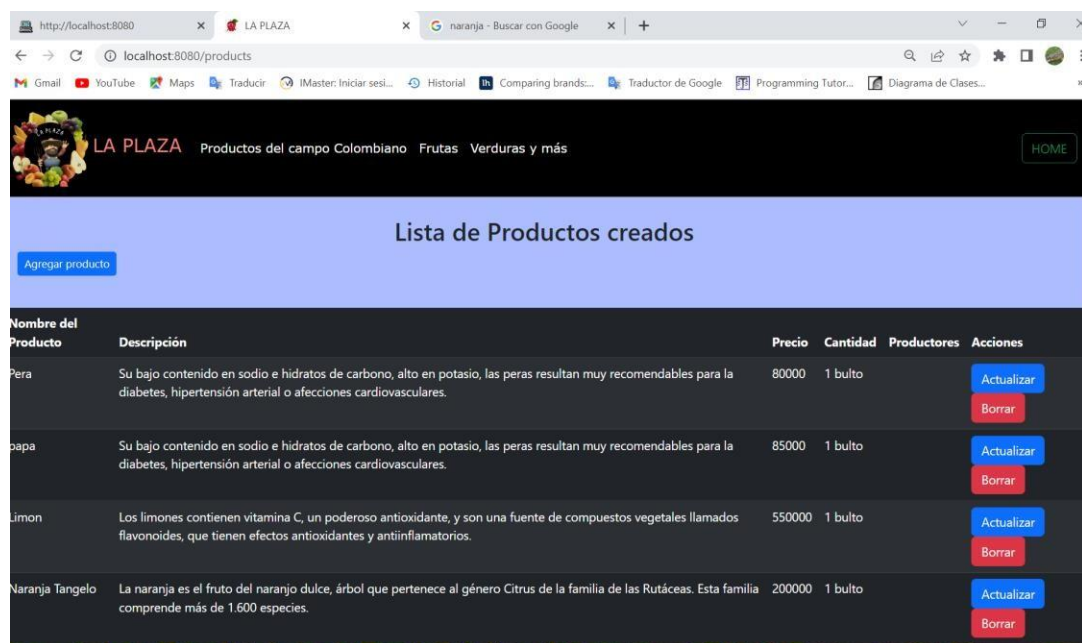
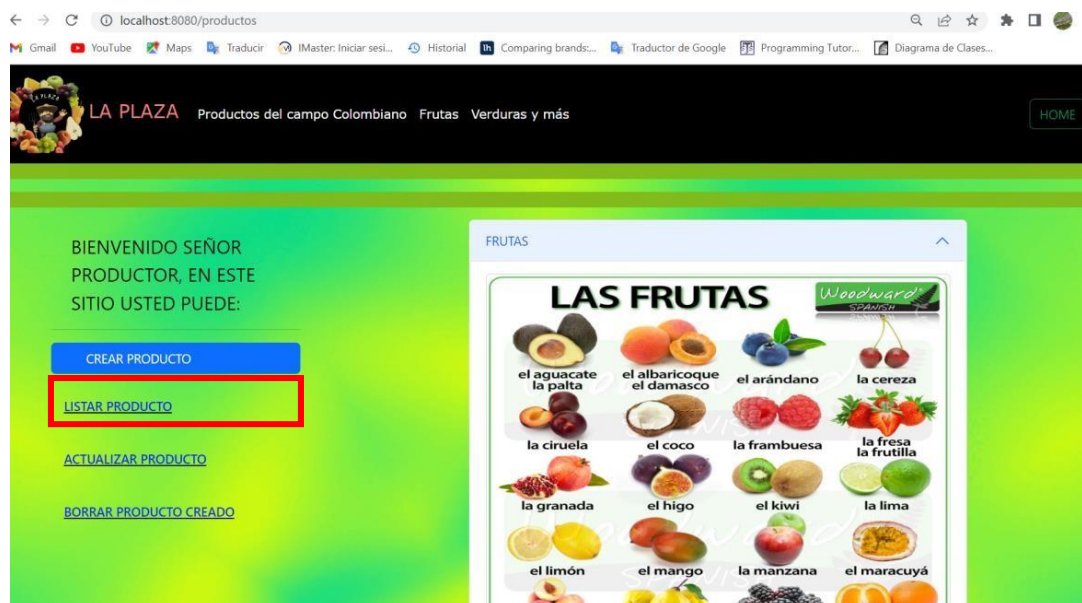
	id	cantidad	description	price	product_name
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	1 bulto	Su bajo contenido en sodio e hidratos de carbono, ...	80000	Pera
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	1 bulto	Los limones contienen vitamina C, un poderoso anti...	70000	Limon

Seleccionar todo Para los elementos que están marcados: Editar Copiar Borrar Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

## 16. Listar Producto

Visualización fronted selección “LISTAR PRODUCTO”



```
ProductController.java
misiontic2022 > src > main > java > misiontic2022 > com > laplaza > controller > ProductController.java
44 @GetMapping("/products")
45 public String listProducts(Model model) {
46     model.addAttribute(attributeName: "products", productService.getAllProduct());
47     return "ListarProducto";
48 }
49
```

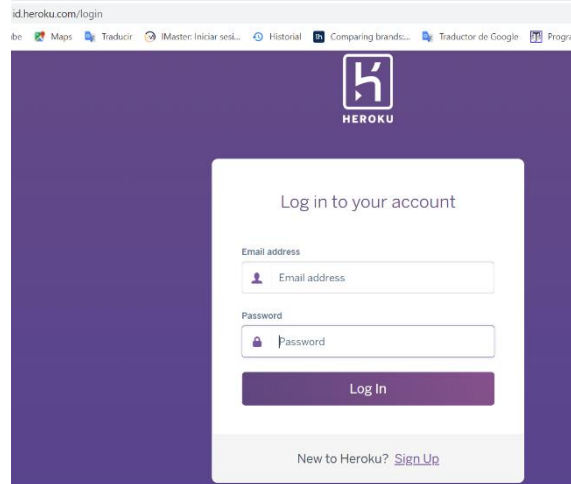
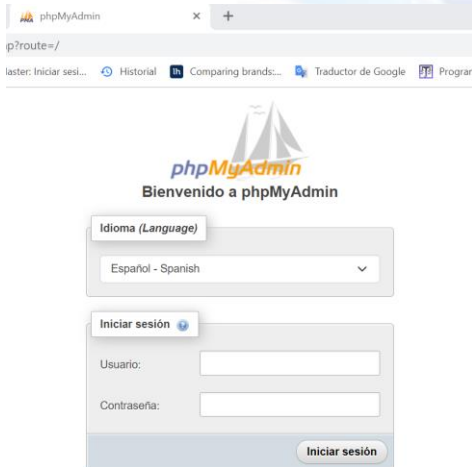
El método (listProducts), debe retornar un String que va a ser el nombre de la vista ListarProducto cuya base es tomada de la clase products

```
ListarProducto.html
misiontic2022 > src > main > resources > templates > ListarProducto.html
148 <tr th:each="product: ${products}" scope="row">
149     <td th:text="${product.product_name}"></td>
150     <td th:text="${product.description}"></td>
151     <td th:text="${product.price}"></td>
152     <td th:text="${product.cantidad}"></td>
153     <td>
154         <th:block th:each="producer, iter: ${product.producers}">
155             <label th:text="${producer.name}">
156                 <th:block th:if="${!iter.last}">, </th:block>
157         </th:block>
158     </td>
159 </tr>
160
161 <td>
162
163     <a th:href="@{/actualizar/{id}(id=${product.id})}" class="btn btn-primary">Actualizar</a>
164     <a th:href="@{/products/{id}(id=${product.id})}" class="btn btn-danger">Borrar</a>
165
166 </td>
167
168 </tr>
169 </tbody>
```

## 17. DESPLIEGUE

Para el despliegue se realiza la instalación de Maven posterior, se crea el repositorio en el Github, se crea cuenta en base de datos en la nube <https://db4free.net/> y se crea cuenta en heroku <https://signup.heroku.com/login>





Después de la instalación correspondiente al sistema operativo, se comprueba desde la consola del sistema (CMD desde Windows) la ejecución de Heroku CLI, de la forma:

```
heroku --version
```

Debería obtener una salida como la siguiente:

```

C:\> Símbolo del sistema

Microsoft Windows [Versión 10.0.19044.2006]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\YAMILE>heroku --version
» Warning: heroku update available from 7.53.0 to 7.63.4.
heroku/7.53.0 win32-x64 node-v12.21.0

C:\Users\YAMILE>
```

Finalmente, es necesario realizar el login en nuestro computador, para realizar el login puede ejecutar uno de los dos comandos y seguidamente, introducir sus datos

de heroku login **heroku login**

ó heroku login -i **heroku login -i**

Con la autenticación exitosa, se actualiza los siguientes archivos y se crear el proyecto en Heroku:

```
</dependencies>
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
    <plugin>
      <groupId>com.heroku.sdk</groupId>
      <artifactId>heroku-maven-plugin</artifactId>
      <version>3.0.4</version>
      <configuration>
        <appName>fierce-earth-92364</appName>
        <includeTarget>false</includeTarget>
        <includes>
          <include>${project.build.directory}/${project.build.finalName}.jar</include>
        </includes>
        <jdkVersion>${java.version}</jdkVersion>
        <processTypes>
          <web>java $JAVA_OPTS -jar target/${project.build.finalName}.jar</web>
        </processTypes>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Figure 1.pom.xml

Para lograr subir con éxito el proyecto a Heroku, es necesario hacer algunas adiciones en primer lugar (configuraciones del plugin de Maven) y actualizamos: El archivo "pom.xml" Agreguemos el siguiente código en el Plug-in que se encuentra dentro de la etiqueta (ubicada al final del archivo "pom.xml"), por el siguiente: com.heroku.sdk heroku-maven-plugin 3.0.4, Se actualiza el archivo "application.properties".

```
You, hace 5 días | 1 author (You)
# port

server.port=${PORT:8090}

# Configurar la coneccion a la base de datos
spring.datasource.url=jdbc:mysql://db4free.net:3306/laplaza2022?useSSL=false&serverTimezone=UTC&use
spring.datasource.username=laplaza2022
spring.datasource.password=laplaza2022
spring.datasource.driverClassName = com.mysql.jdbc.Driver

#Hibernate
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

#Hibernate auto ddl
spring.jpa.hibernate.ddl-auto=update
logging.level.org.hibernate.SQL=DEBUG
```

Figure 2.application.properties

Se configura el puerto 8090 en el archivo "application.properties". El archivo se ubica en la ruta /src/main/resources/, al inicio del archivo application.properties debe agregar la siguiente línea # port server.port=\${PORT:8090}

Se abre la consola en vscode y se ejecutan los siguientes comandos: git add. Seguidamente: git commit -m "initial commit" posterior, ejecutamos el siguiente comando: heroku create. El sistema automáticamente crea el nombre aleatorio de la página la cual se configura (archivo "pom.xml") appName: de tal forma que debe coincidir con el nombre de la aplicación creada en Heroku.

```
<version>3.0.4</version>
<configuration>
  <appName>tranquil-anchorage-91313</appName>
  <includeTarget>false</includeTarget>
  <includes>
```

Antes de ejecutarlos verificamos que existen cambios no confirmados o pendientes por cargar al repositorio de GIT, la secuencia sería la siguiente:

```
git status
```

-> para verificar el estado

```
git add .
```

-> para agregar los cambios a la pila

```
git commit -m "commit message"
```

-> para confirmar los cambios

Ahora ejecutamos el comando que realizará el push de nuestro proyecto a la aplicación creada en heroku, el comando puede tardar unos minutos dependiendo de nuestro aplicativo:

```
mvn clean heroku:deploy
```

Una vez finalizado el proceso, ejecutamos:

```
heroku open
```

para que abra la aplicación en el navegador.

En caso de errores se ejecuta:

```
heroku logs ó heroku logs -tail
```

Nos ayuda con la depuración. A continuación, se muestra el despliegue

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Personal

New

Welcome to Heroku

Now that your account has been set up, here's how to get started.

Show next steps

Dismiss

Starting November 28th, 2022, free Heroku Dynos, free Heroku Postgres, and free Heroku Data for Redis\* will no longer be available. If you have apps using any of these resources, you must upgrade to paid plans by this date to ensure your apps continue to run and to retain your data. For students, we will announce a new program by the end of September. [Learn more](#)

fierce-earth-92364

Sep 29 at 1:36 PM

heroku-maven-plugin · heroku-22 · United States

## Se muestra el estado actúa de la rama de Github del proyecto

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#)

Choose a branch to deploy

main

Deploy Branch

Receive code from GitHub

Build main 719f3e5c

Release phase

Deploy to Heroku

Your app was successfully deployed.

View

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#)

Choose a branch to deploy

main

Deploy Branch

Receive code from GitHub

Build main 719f3e5c

Release phase

Deploy to Heroku

Your app was successfully deployed.

View





<https://fierce-earth-92364.herokuapp.com/>

El usuario genera peticiones por medio de la vista Controlador estas son enviadas a service (interfaz) cumple la funcion de definir los métodos posterior se envia al repositorio y allí se generan las consultas y envían los datos a mysql que su vez le devuelve respuesta al repositorio al service entidad y controler.

## 18. FUNCIONAMIENTO APIS

Para este proyecto se implementó spring MVC para el desarrollo del controler, adicional a esto se importan paquetes específicos para su funcionamiento.

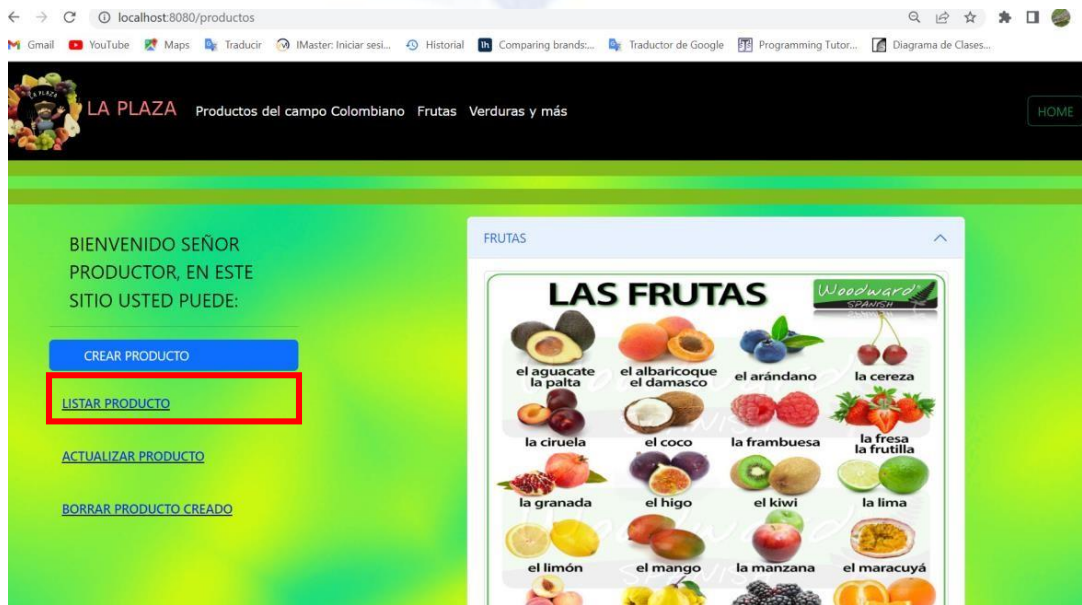
En este controler se tuvieron en cuenta las siguientes APIS.

La primera API hace el llamado a nuestra página, la cual se integra por medio de la notación @GetMapping instancia a una solicitud hacia el endpoint (raíz de la API)

```
@GetMapping("/")
public String index(Model model) {
    model.addAttribute(attributeName: "index", productService.getAllProduct());
    return "index";
}
```

Seguidamente creamos la notación @Getmapping que realiza solicitud hacia el endpoint ("/productos"), esta muestra información de productos y menciona las diferentes opciones que el usuario puede hacer en esta página.

```
@GetMapping("/productos")
public String crudp(Model model) {
    model.addAttribute(attributeName: "products", productService.getAllProduct());
    return "productos";
}
```



La siguiente API se utiliza la notación @GetMapping que instancia la solicitud al endpoint "/products/new" (página de productos) se crea un objeto "product" que almacena los valores teniendo en cuenta que a través del Model crea un nuevo producto y se listan los productores, por medio de la clase createProductForm muestra la pantalla del con el formulario de creación de producto.

```
@GetMapping("/products/new")
public String createProductForm(Model model) {

    Product product = new Product();

    model.addAttribute(attributeName: "product", product);
    model.addAttribute(attributeName: "producersList", producersList);

    return "CrearProducto";
}
```

The screenshot shows the 'CREAR PRODUCTO' form. It has a title 'CREAR PRODUCTO' and a subtitle 'Nombre del Producto'. The form contains several input fields: 'Nombre del Producto' (with the value 'Naranja Valencia'), 'Descripción del Producto' (with a text area containing 'La naranja Valencia es una variedad dulce de la naranja. Su nombre está inspirado en la...'), 'Precio' (with the value '200000'), 'Cantidad' (with the value '1 bulto'), and 'Productores' (with a dropdown menu). There is an 'Enviar' button at the bottom.



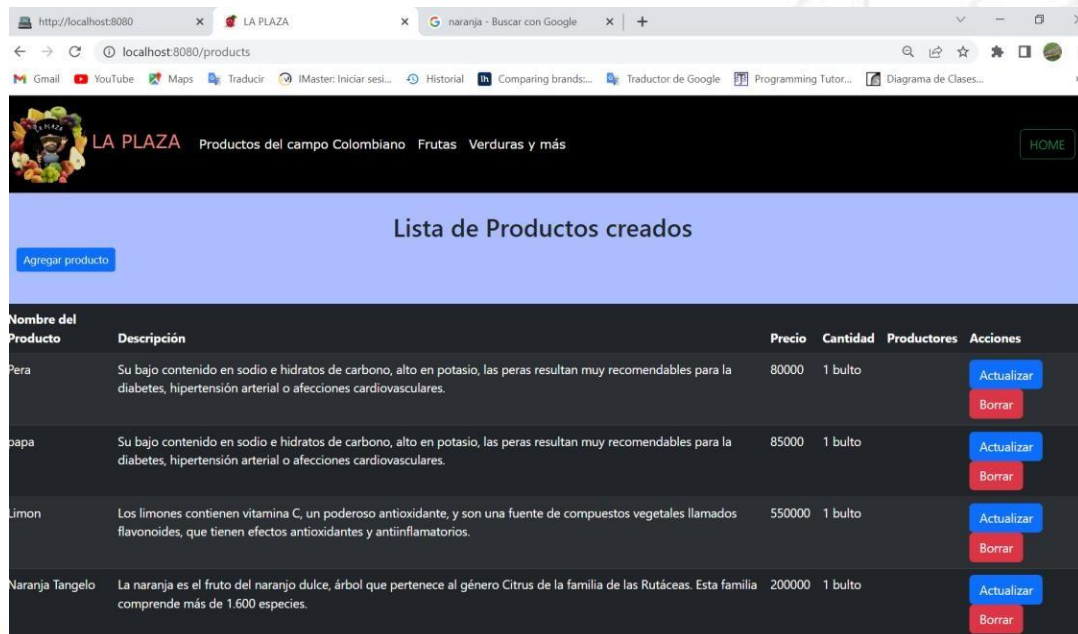
creamos una API (hacia el endpoint “/products”) que guardara en la base de datos un nuevo producto creado, de la siguiente forma:

```
@PostMapping("/products")
public String saveProduct(@ModelAttribute("product") Product product) {
    productService.saveProduct(product);
    return "redirect:/products";
}
```

Haciendo el llamado a nuestra Entidad “Product” luego de esto se va al productService dentro de este se sobrescriben los métodos definidos en la clase principal IProductService (interfaz) que contiene los métodos, en este caso saveProduct que guarda el producto. Posterior se lo almacena en el repositorio enviándolo al mysql.

Seguidamente realizamos la tercera API utilizando la notación @GetMapping que realiza solicitud para listar productos a través del Model quien trae todos los objetos de la base de datos. Para este se envía la información a la base de datos que es consultada a través del @Query en el repositorio. A su vez este lo envía al service allí se capturan los productos según su nombre y a su vez lo muestra como lista en la Vista controlador (listProducts).

```
@GetMapping("/products")
public String listProducts(Model model) {
    model.addAttribute("products", productService.getAllProduct());
    return "ListarProducto";
}
```



Nombre del Producto	Descripción	Precio	Cantidad	Productores	Acciones
Pera	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	80000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Papa	Su bajo contenido en sodio e hidratos de carbono, alto en potasio, las peras resultan muy recomendables para la diabetes, hipertensión arterial o afecciones cardiovasculares.	85000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Limon	Los limones contienen vitamina C, un poderoso antioxidante, y son una fuente de compuestos vegetales llamados flavonoides, que tienen efectos antioxidantes y antiinflamatorios.	550000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>
Naranja Tangelo	La naranja es el fruto del naranjo dulce, árbol que pertenece al género Citrus de la familia de las Rutáceas. Esta familia comprende más de 1.600 especies.	200000	1 bulto		<a href="#">Actualizar</a> <a href="#">Borrar</a>

Seguidamente creamos una API que me devuelve un producto desde la base de datos para ser editado, esta API utiliza el id para alcanzar el modelo(objeto).

```
@GetMapping("/actualizar/{id}")
public String irActualizar(@PathVariable Long id, Model model) {
    Product pr = productService.getProductById(id);
    model.addAttribute(attributeName: "producersList", producersList);
    if(pr.getId() !=0){model.addAttribute(attributeName: "product", pr);
    return "ActualizarProducto";
} else {
    return "redirect:/products";
}
}
```

El usuario devuelve la actualización desde la vista controlador según los atributos definidos en la función Entity(product) a su vez está la envía al service, a través del método actualizar (updateProduct) se actualiza y se guarda a través del método saveProduct lo envía al repositorio y lo almacena en Mysql.

```
@PostMapping("/actualizar/{id}")
public String actualizar(@PathVariable Long id,
    @ModelAttribute("product") Product product,
    Model model) {
    // sacar el producto de la b.d. por el id
    Product existentProduct = productService.getProductById(id);
    // cargarlo
    existentProduct.setId(id);
    existentProduct.setProduct_name(product.getProduct_name());
    existentProduct.setDescription(product.getDescription());
    existentProduct.setPrice(product.getPrice());
    existentProduct.setCantidad(product.getCantidad());

    existentProduct.setProducers(product.getProducers());

    // guardar el producto actualizado
    productService.actualizar(existentProduct);

    return "redirect:/products";
}
```

Finalmente creamos un API para eliminar un producto

```
@GetMapping("/products/{id}")  
public String deleteProduct(@PathVariable Long id) {  
    productService.deleteProductById(id);  
    return "redirect:/products";  
}
```

El usuario envía la petición en la vista controlador posterior esta se va al service y a través del método deleteProduct lo envía al repositorio por medio de la notación @Query realiza la consulta por nombre y a lo elimina de la base de datos.



## 19. INFORME DE RETROSPECTIVA:

Con los conocimientos adquiridos de los Spring anteriores se logra concluir el desarrollo del proyecto en el Spring 4. Logrando ejecutar el despliegue en la aplicación y realización de las correspondientes pruebas de funcionamiento.

Agradecemos el acompañamiento del tutor Scrum Juan Pablo Solarte en cada uno de los Spring desarrollados y la oportuna enseñanza del instructor Luis Molero, gracias a ello pudimos concluir de manera significativa nuestro proyecto.

Como enseñanza de aprendizaje se adquieren habilidades blandas en trabajo de equipo y sincronización de horarios e ideas. Ha sido una experiencia grata con resultados satisfactorios. Se logró una constante comunicación asertiva la cual nos permitió crecer tanto profesional como personal.

Se utilizaron las siguientes aplicaciones para llevar la trazabilidad del proyecto:

**Trello:** <https://trello.com/b/HFOhNBkj/proyecto-software>

**GitHub:**

## GESTIÓN DE CONFIGURACIÓN

**Gestor Proyecto** Sayda Yamile Cagua Carrillo

**Gestor Base Datos** José Daniel Ramírez Saldaña

**Desarrollador Frontend** Julián Andrés Segura González

**Desarrollador Backend** Dora Paola Pacheco Moreno

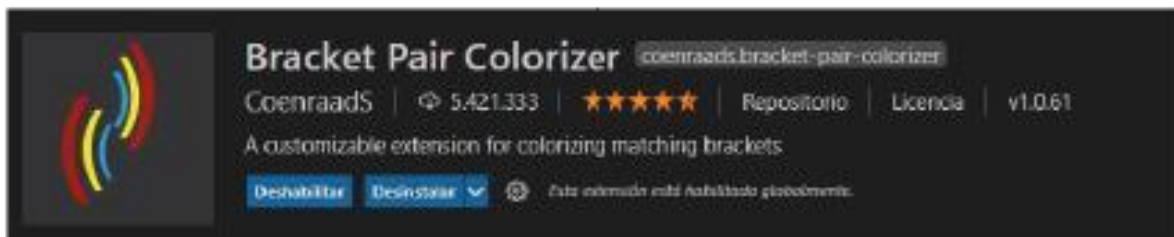
**Tester** Isis Nirvana Segura Valero

### Instalación del editor de código

Selección de editor de código: **Visual Studio Code**

Instalación de extensiones:

Bracket pair colorizer



**Bracket Pair Colorizer** coenraads.bracket-pair-colorizer  
CoenraadS | 5.421.333 | ★★★★★ | Repositorio | Licencia | v1.0.61  
A customizable extension for coloring matching brackets  
Deshabilitar Desinstalar Esta extensión está habilitada globalmente.

ESLint: corregir errores de sintaxis en java script



**ESLint** dbaeumer.vscode-eslint  
Dirk Baeumer | 14.869.381 | ★★★★★ | Repositorio | Licencia | v2.1.22  
Integrates ESLint JavaScript into VS Code.  
Deshabilitar Desinstalar Esta extensión está habilitada globalmente.

HTML Snippets: completa código HTML



**HTML Snippets** abusaidm.html-snippets  
Mohamed Abusaid | 5.359.440 | ★★★★★ | Licencia | v0.2.1  
Full HTML tags including HTML5 Snippets  
Deshabilitar Desinstalar Esta extensión está habilitada globalmente.

Intellisense for CSS class names in HTML: autocompletar o auto rellenar las clases CSS

