Documentación Técnica

Tomás Eduardo Romero Cañas

INTRODUCCIÓN

*la presente documentación técnica describe el desarrollo de una aplicación web y uso de una biblioteca reutilizable en **Java** diseñada para realizar operaciones matemáticas básicas, como suma, resta, multiplicación y división. Este proyecto fue creado con un enfoque modular y reutilizable, utilizando **Maven***

El desarrollo de la pequeña aplicación se hizo utilizando tecnologías Java que permite registrar y listar productos. La solución emplea una arquitectura modular basada en Spring Boot para el backend y React para el frontend. Los datos se gestionan mediante Hibernate y una base de datos MySQL. Adicionalmente, se implementó un sistema de autenticación basado en JWT para asegurar el acceso a las funcionalidades de la aplicación.

Desarrollo de la Aplicación Web

Requisitos Cubiertos

1. ***Registrar y listar productos:***
   - *Los productos tienen los atributos: nombre, descripción, precio y categoría.*
   - *Se incluye un formulario para crear y editar productos.*
2. ***Funcionalidades adicionales:***
   - *Autenticación de usuarios mediante JWT.*
   - *Validaciones en el backend y frontend para campos requeridos y formatos válidos.*
   - *Alertas en el frontend para operaciones exitosas o fallidas.*
3. ***Persistencia de datos:***
   - *Uso de Hibernate/JPA para mapear entidades con MySQL.*
4. ***Frontend:***
   - *React fue utilizado para construir una interfaz dinámica y responsiva.*
   - *Operaciones CRUD están disponibles para la gestión de productos.*

Arquitectura de la Aplicación

*La aplicación sigue el patrón **MVC (Modelo-Vista-Controlador)**:*

1. ***Backend (Spring Boot):***
   - *Controladores RESTful gestionan las peticiones de los clientes.*
   - *Servicios encapsulan la lógica del negocio.*

    - *Repositorios gestionan las interacciones con la base de datos.*
    - *Autenticación con JWT asegura el acceso controlado a las rutas.*
2. ***Base de Datos (MySQL):***
    - *Estructura relacional con tablas para usuarios y productos.*
    - *Scripts de inicialización para insertar datos básicos.*
3. ***Frontend (React):***
    - *Componentes funcionales con hooks para gestionar el estado.*
    - *Axios para consumir los endpoints del backend.*
    - *React Router para navegación.*

Decisiones de Diseño

1. ***Seguridad:***
    - *JWT asegura que solo usuarios autenticados puedan acceder a funcionalidades sensibles.*
2. ***Desacoplamiento:***
    - *Separación estricta entre frontend y backend permite mayor flexibilidad en despliegue y mantenimiento.*
3. ***Escalabilidad:***
    - *Uso de Spring Boot y React permite que la solución escale fácilmente con requerimientos futuros.*

Instrucciones de Instalación
Requisitos Previos

1. ***Software:***
   - *Java JDK 17+*
   - *Node.js 18+*
   - *MySQL 8.1+*
   - *Maven*
2. ***Configuración:***
   - *MySQL configurado y corriendo.*

Pasos para Instalar y Ejecutar

1. ***Clonar el repositorio:***
   Git clone https://github.com/Tomas7855/Prueba-T-cnica-Java.git
   Cd <directorio del proyecto>

2. ***Configurar la base de datos:***
   Crear una base de datos en MySQL
   - CREATE DATABASE productos_db;

   Actualizar las credenciales en application.properties:

   spring.datasource.url=jdbc:mysql://localhost:3306/productos_db
   spring.datasource.username=<usuario>
   spring.datasource.password=<contraseña>

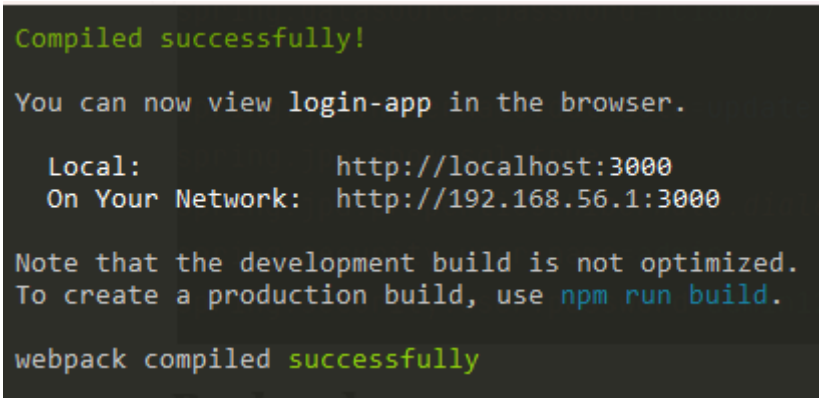
```
spring.datasource.url=jdbc:mysql://localhost:3306/productos_db
spring.datasource.username=root
spring.datasource.password=rc18087

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.security.user.name=admin
spring.security.user.password=admin123
```

3. ***Backend:***

   Navegar al directorio del backend: cd <directorio> (se puede abrir con Intellij)

   mvn clean install
   mvn spring-boot:run

4. ***Frontend:***
   Navegar al directorio del frontend: cd <directorio> (se puede abrir con Vscode)
   - Instalar dependencias:
     npm install
   - Ejecutar la aplicación:
     npm start



5. ***Probar la aplicación:***
   Acceder a http://localhost:3000 para interactuar con la interfaz.

Ejemplo de Operaciones CRUD
Los ejemplos de operaciones del crud se realizaron en Postman

1. Registrar un producto
Endpoint: POST /api/productos/registrar

```
{
"nombre": "Producto 1",
"descripcion": "Descripción del producto",
"precio": 100.0,
"categoria": "Categoría A"
}
```

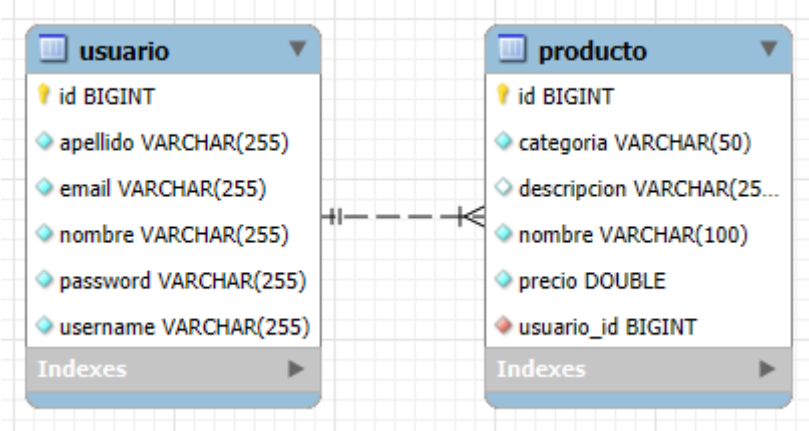
2. Listar productos
Endpoint: GET /api/productos/listar

3. Editar un producto
Endpoint: PUT /api/productos/editar/{id}

```
{
 "nombre": "Producto actualizado",
 "descripcion": "Descripción actualizada",
 "precio": 120.0,
 "categoria": "Categoría B"
}
```

4. Eliminar un producto
Endpoint: DELETE /api/productos/eliminar/{id}

Esquema de base de datos



Biblioteca Reutilizable

Para importar la biblioteca es de agregarla al pom.xml y agregar

```
<groupId>com.Rc18087</groupId>
<artifactId>CalcBib</artifactId>
<version>1.0</version>
```

```xml
<dependency>
    <groupId>com.Rc18087</groupId>
    <artifactId>CalcBib</artifactId>
    <version>1.0</version>
</dependency>
```

Luego de esp ya la podemos utilizar:
Importamos la biblioteca: import org.Rc18087.CalcBib;
Y por ultimo la utilizamos: System.out.println("Suma" + CalcBib.suma(5,9));