**Instructivo**

**Manual Técnico,**

**versión 1.0**

# INTRODUCCIÓN

# El funcionamiento del servicio es que el cliente realiza una compra en línea y la envía a una dirección de Estados Unidos provista por ¡Te lo traigo!, en la cual su paquete será recibido y posteriormente será enviado a El Salvador, donde se gestionará la importación y la entrega en una sucursal de su preferencia.

# Herramientas Back End

**1. Lenguaje de Programación: Java**

Java es un lenguaje de programación orientado a objetos, conocido por su portabilidad y robustez. Es ampliamente utilizado en el desarrollo de aplicaciones empresariales, y su ecosistema rico en bibliotecas y frameworks lo hace ideal para construir aplicaciones backend.

**2. Framework: Spring Boot**

Spring Boot es un framework que simplifica el proceso de desarrollo de aplicaciones basadas en Spring. Ofrece una configuración automática y una serie de convenciones que permiten crear aplicaciones de manera rápida y eficiente. Entre sus características destacan:

* **Configuración simplificada:** Al eliminar la necesidad de archivos de configuración extensos.
* **Soporte para microservicios:** Facilita la creación de aplicaciones escalables y mantenibles.
* **Integración con herramientas:** Proporciona soporte para herramientas como Spring Data, Spring Security, y más.

**3. Seguridad: Implementación de JWT (JSON Web Tokens)**

Para manejar la autenticación y autorización, se implementa JWT, un estándar abierto (RFC 7519) que permite transmitir información de forma segura entre partes como un objeto JSON. En este contexto:

* **Generación de Token:** Al autenticarse, el usuario recibe un token que contiene su identidad y posiblemente roles o permisos.
* **Validación del Token:** Cada solicitud posterior incluye el token en el encabezado. El backend lo valida para determinar si el usuario tiene acceso a los recursos solicitados.
* **Estado sin sesión:** JWT permite que la aplicación sea stateless, ya que no se almacenan sesiones en el servidor, lo que facilita la escalabilidad.

**4. Versiones**

* **Java 11:** Esta versión incluye mejoras de rendimiento y nuevas características del lenguaje, como las API de colección mejoradas y la introducción de var para inferencia de tipos.
* **Spring 2.6.13:** Incluye mejoras en la gestión de configuraciones, un mejor manejo de la inyección de dependencias y optimizaciones para la ejecución en entornos de producción.

**5. Uso de JPA (Java Persistence API)**

JPA es una especificación que simplifica el manejo de la persistencia de datos en aplicaciones Java. En este desarrollo:

* **Hibernate:** Como proveedor de JPA, permite la mapeo objeto-relacional (ORM), facilitando la interacción con bases de datos.
* **Entidades:** Las clases de entidad se anotan con @Entity, representando tablas en la base de datos.
* **Repositorios:** Se utilizan interfaces que extienden JpaRepository para realizar operaciones CRUD sin necesidad de implementar lógica de acceso a datos.

**6. Integración de Spring Security**

Spring Security es un potente framework para manejar la seguridad en aplicaciones. En este desarrollo:

* **Autenticación y Autorización:** Se configuran filtros de seguridad para manejar la autenticación basada en JWT, donde el AuthenticationManager verifica las credenciales del usuario.
* **Configuración de seguridad:** Se definen reglas para proteger rutas específicas, permitiendo o denegando el acceso según el rol del usuario.
* **Contexto de seguridad:** Spring Security proporciona un contexto de seguridad que permite acceder a la información del usuario autenticado en cualquier parte de la aplicación.

# Herramientas Front End

**1. Lenguaje de Programación: TypeScript**

TypeScript es un superconjunto de JavaScript que agrega tipos estáticos y características de programación orientada a objetos. Esto permite detectar errores en tiempo de compilación y mejorar la mantenibilidad del código. TypeScript se compila a JavaScript, lo que asegura compatibilidad con cualquier entorno que soporte JS.

**2. Framework: Angular**

Angular es un framework para el desarrollo de aplicaciones web que utiliza TypeScript. Proporciona una arquitectura basada en componentes que facilita la creación de aplicaciones escalables y mantenibles. Entre sus características más destacadas se incluyen:

* **Componentes:** Angular se basa en componentes que encapsulan la lógica y la presentación.
* **Inyección de Dependencias:** Facilita la gestión de dependencias y la creación de aplicaciones modulares.
* **Enrutamiento:** Proporciona un enrutador robusto para gestionar la navegación entre vistas.

**3. Seguridad: Implementación y Uso de JWT (JSON Web Tokens)**

JWT se utiliza para manejar la autenticación y autorización en la aplicación Angular. El flujo general es el siguiente:

* **Autenticación:** Cuando un usuario inicia sesión, las credenciales se envían al backend. Si son válidas, el backend devuelve un token JWT.
* **Almacenamiento del Token:** El token se almacena en el almacenamiento local (localStorage) o en cookies.
* **Acceso a Recursos Protegidos:** En cada solicitud a un recurso protegido, el token se envía en el encabezado de autorización.
* **Validación del Token:** El backend valida el token y responde con los datos solicitados si es válido.

**4. Versiones**

* **Angular CLI 14.1.3:** Herramienta de línea de comandos que facilita la creación y gestión de proyectos Angular, permitiendo generar componentes, servicios, y más, con comandos simples.
* **Node (npm) 8.15.1:** Node.js proporciona el entorno de ejecución para la aplicación y npm gestiona las dependencias.
* **TypeScript 4.1.3:** Esta versión introduce nuevas características que mejoran la escritura de código y el manejo de tipos.

**5. Uso de Bootstrap para Diseño**

Bootstrap es un framework CSS que facilita el diseño responsivo y moderno. En este desarrollo:

* **Componentes de UI:** Se utilizan clases de Bootstrap para estilizar componentes de la aplicación, como botones, formularios y modales.
* **Grid System:** Bootstrap proporciona un sistema de rejilla (grid) que permite crear layouts responsivos de manera sencilla.
* **Customizable:** Se pueden personalizar los estilos de Bootstrap mediante variables y temas.

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

# Herramientas Base de datos

**1. Descripción de PostgreSQL**

PostgreSQL es un sistema de gestión de bases de datos relacional y objeto, conocido por su robustez, escalabilidad y cumplimiento de estándares SQL. Ofrece características avanzadas como:

* **Transacciones ACID:** Asegura la integridad de los datos con soporte para transacciones atómicas, consistentes, aisladas y duraderas.
* **Soporte para JSON:** Permite almacenar datos en formato JSON, lo que facilita la gestión de datos semiestructurados.
* **Extensibilidad:** Permite la creación de tipos de datos personalizados, funciones y operadores.

**2. Gestor de Base de Datos: PgAdmin 4**

PgAdmin 4 es una herramienta de administración y desarrollo para PostgreSQL. Proporciona una interfaz gráfica intuitiva para gestionar bases de datos, ejecutar consultas SQL y visualizar datos. Entre sus características destacan:

* **Interfaz de Usuario:** Una GUI que facilita la navegación por esquemas, tablas y datos.
* **Editor de Consultas:** Permite ejecutar consultas SQL y ver resultados en tiempo real.
* **Gestión de Roles y Permisos:** Facilita la administración de usuarios y sus privilegios en la base de datos.

**3. Versiones**

* **PostgreSQL 17.0-1:** La última versión incluye mejoras en el rendimiento y nuevas características, como optimizaciones de consulta, soporte para el manejo de grandes volúmenes de datos y mejoras en la concurrencia.

**4. Integración con Spring Boot por JDBC**

La integración de PostgreSQL en un proyecto Spring Boot se puede realizar a través de JDBC (Java Database Connectivity), que permite a la aplicación Java interactuar con la base de datos. Los pasos clave para la integración son:

* **Dependencias en Maven:** Asegúrate de incluir las dependencias necesarias en el archivo pom.xml:

Texto

Descripción generada automáticamente

# Uso de proyecto Spring Boot

**1. Prerequisitos**

**Antes de comenzar, asegúrate de que tienes lo siguiente:**

* **Java JDK 11 instalado en tu máquina. Puedes verificarlo ejecutando java -version en la terminal.**
* **Maven instalado para la gestión de dependencias y construcción del proyecto. Verifica con mvn -version.**
* **PostgreSQL ejecutándose localmente y la base de datos ya creada.**

#### 2. **Configuración del Archivo** pom.xml

Asegúrate de que tu archivo pom.xml contenga las dependencias necesarias para Spring Boot y PostgreSQL:

Texto

Descripción generada automáticamente

**6. Ejecutar la Aplicación**

**Para ejecutar la aplicación, sigue estos pasos:**

* + **Ingresar al editor de código de su elección, ir a la carpeta del proyecto y dar clic derecho sobre el aplicativo, luego seleccionar Run As Spring boot app**

Captura de pantalla de computadora

Descripción generada automáticamente

# Uso de proyecto Angular

**1. Prerequisitos**

**Antes de comenzar, asegúrate de tener lo siguiente:**

* **Node.js y npm instalados. Puedes verificarlo ejecutando node -v y npm -v en la terminal.**
* **Angular CLI instalado globalmente. Verifica la instalación con ng version. Si no lo tienes, instálalo con el siguiente comando:**

**npm install -g @angular/cli**

* **Proyecto Spring Boot ya en ejecución. Asegúrate de que el backend esté corriendo y accesible, normalmente en http://localhost:8083.**

**2. Clonar o Descargar el Proyecto Angular**

**Si no has clonado o descargado el proyecto, hazlo desde el repositorio correspondiente. Asegúrate de navegar a la carpeta del proyecto:**

**git clone <URL-del-repositorio>**

**cd nombre-del-proyecto-angular**

**3. Instalación de Dependencias**

**Si es la primera vez que ejecutas el proyecto Angular, usa el siguiente comando para instalar las dependencias, teniendo en cuenta la advertencia sobre las dependencias:**

**npm install --legacy-peer-deps=true**

**Esto instalará las dependencias necesarias y evitará problemas con versiones de paquetes que pueden no ser compatibles.**

**4. Configuración del Proyecto Angular**

**Verifica si el proyecto tiene un archivo de configuración que se conecta al backend. Por lo general, esto se configura en un archivo de entorno, como src/environments/environment.ts. Asegúrate de que la URL del backend esté correctamente configurada:**

**export const environment = {**

**production: false,**

**apiUrl: 'http://localhost:8083/api' // Asegúrate de que esta URL sea correcta**

**};**

**5. Ejecutar la Aplicación Angular**

**Una vez que las dependencias estén instaladas y la configuración esté correcta, puedes ejecutar el proyecto Angular con el siguiente comando:**

**npm start**

**Esto compilará el proyecto y levantará un servidor de desarrollo. Por defecto, la aplicación se ejecutará en** [**http://localhost:4200**](http://localhost:4200)**.**

**6. Acceder a la Aplicación**

**Abre tu navegador y navega a http://localhost:4200 para acceder a la aplicación Angular. Aquí podrás interactuar con la interfaz de usuario y realizar autenticación o cualquier otro flujo que dependa del backend.**

**7. Realizar Autenticación**

**Si tu aplicación Angular incluye autenticación, asegúrate de que estés enviando las credenciales adecuadas al backend. Generalmente, esto se hará a través de un formulario en el componente correspondiente. El flujo de autenticación típicamente es:**

1. **Formulario de Inicio de Sesión: El usuario ingresa sus credenciales.**
2. **Envio de Credenciales: Al enviar el formulario, se realiza una solicitud POST al endpoint de autenticación del backend (ej. http://localhost:8083/devintech/authenticate/).**
3. **Recepción de JWT: Si las credenciales son válidas, el backend devuelve un JWT que se almacena en localStorage o en cookies para su uso posterior.**

**8. Consideraciones sobre Dependencias**

**Si encuentras problemas relacionados con dependencias durante la instalación o ejecución, considera:**

* **Contactar al Desarrollador: Si no puedes resolver el problema, es recomendable contactar al desarrollador del proyecto para obtener asistencia sobre las dependencias específicas.**
* **Verificar Versiones de Dependencias: Revisa el archivo package.json para identificar versiones específicas que podrían ser problemáticas.**