

# **DESARROLLO DE SOFTWARE**

## **PROYECTO DE SOFTWARE PARA GESTIONAR TRANSACCIONES COMERCIALES DE UNA TIENDA GENÉRICA**

## TABLA DE CONTENIDO

Introducción	5
Descripción del caso	5
Parte 1: Especificación Funcional	5
Parte 2: Organización del Equipo Scrum	8
Parte 3: Organización de los Sprints	8
<b>Sprint 1</b>	8
Historias de Usuario	9
Interfaz Gráfica	9
Conjunto de Pruebas	10
<b>Sprint 2</b>	12
Historias de Usuario	12
Interfaz Gráfica	13
Conjunto de Pruebas	14
<b>Sprint 3</b>	16
Historias de Usuario	17
Interfaz Gráfica	17
Conjunto de Pruebas	18
<b>Sprint 4</b>	18
Historias de Usuario	19
Interfaz Gráfica	19
Conjunto de Pruebas	20
<b>Sprint 5</b>	21
Historias de Usuario	21
Interfaz Gráfica	22
Conjunto de Pruebas	23
Parte 4: Especificaciones Técnicas	24
Lenguajes de programación y stack tecnológico de desarrollo	24
Especificación de diseño de la base de datos	25

Especificación de la API para conexión del Frontend con el Backend de la aplicación	26
Especificaciones de uso y manejo de la plataforma en nube AWS para publicación del proyecto.	38
Especificaciones de Repositorio GitHub para creación de ambientes de desarrollo y producción.	58

## Historial de Versiones

Fecha	Versión	Autor	Organización	Descripción
09/05/2021	1	Instructor	UEB	Descripción inicial de proyecto de software para gestionar las transacciones comerciales de una tienda de propósito general. Se documenta del Sprint 1 al Sprint 4.
22/06/2021	2	Instructor	UEB	Se agrega el modelo entidad relación del proyecto, y la especificación de las APIs, de inicia a documentar el manejo de repositorios GitHub.
27/06/2021	2	Instructor	UEB	Revisión de las estructuras de datos de cada tabla, y ajustes a nombres y tipos de datos. Actualización de las APIs con los nuevos nombres de atributos.
13/07/2021	3	Instructor	UEB	Se agregan las HU del módulo de reportes, las interfaces usuarias y el conjunto de pruebas. Se agregan especificaciones para subir a AWS.
16/07/2021	3	Instructor	UEB	Se agregan especificaciones para la configuración de los repositorios GitHub.
20/07/2021	4	Instructor	UEB	Se modifican los aspectos de creación de instancias EC2 en la configuración de AWS.

## Introducción

El propósito de este documento es dar las especificaciones para el diseño y desarrollo del software escrito en lenguaje Java para Web para gestionar las transacciones comerciales de una tienda que sea funcional para cualquier tipo de comercio que maneje proveedores, clientes, compras, ventas, y productos.

## Descripción del caso

Dentro de los diferentes tipos de negocios que se pueden presentar en la economía de cualquier ciudad o país, existen aquellos que compran a sus proveedores diferentes productos o insumos que, con estos pueden producir los productos que venden a sus clientes a los que se les genera una factura de venta, y estos realizan los pagos de diferentes formas. Este tipo de negocio lo llamaremos en lo sucesivo "tienda genérica", y en ese sentido, EL EQUIPO realizará el desarrollo de tal forma que pueda implementarse en todos los tipos de negocios que operen bajo esta modalidad, cumpliendo los requerimientos que abajo se detallan.

## Parte 1: Especificación Funcional

Se identifica a continuación los requerimientos para la realización del software, consistente en los siguientes módulos y requerimientos por módulo:

### 1. MÓDULO DE LOGIN DEL SISTEMA

EL EQUIPO deberá desarrollar el módulo que permita el ingreso al sistema, una vez se haya realizado la validación por nombre de usuario y contraseña. Deberá tenerse un usuario por defecto con el nombre de **admininicial**, y contraseña **admin123456** para su ingreso la primera vez.

### 2. MÓDULO DE GESTIÓN DE USUARIOS

EL EQUIPO deberá desarrollar el módulo de gestión de los usuarios que trabajen en la tienda para que operen el sistema. Se deberán tener las opciones de crear usuario, actualizar datos de usuario, y borrar usuario. Estos datos se almacenarán en la tabla llamada **usuarios**, y los datos a gestionar son: cedula, nombre completo, correo electrónico, usuario, y contraseña. En este módulo se desactivará el usuario **admininicial**.

### 3. MÓDULO DE GESTIÓN DE CLIENTES

EL EQUIPO deberá desarrollar el módulo de gestión de los clientes de la tienda, para lo cual se deberán tener las opciones de crear cliente, leer clientes, actualizar datos de cliente, y borrar cliente. Estos datos se almacenarán en la tabla llamada **clientes**, y los datos a gestionar son: cedula, nombre completo, dirección, teléfono, y correo electrónico.

#### 4. MÓDULO DE GESTIÓN DE PROVEEDORES.

EL EQUIPO deberá desarrollar el módulo de gestión de los proveedores de la tienda, para lo cual se deberán tener las opciones de crear proveedor, leer proveedor, actualizar datos de proveedor, y borrar proveedor. Estos datos se almacenarán en una tabla llamada **proveedores**, y los datos a gestionar son: NIT, nombre proveedor, dirección, teléfono, y ciudad.

#### 5. MÓDULO DE GESTIÓN DE PRODUCTOS.

EL EQUIPO deberá desarrollar el módulo de gestión de los productos que se venden en la tienda, para lo cual se deberá cargar estos productos de un archivo plano (de texto), con la siguiente estructura de datos:

NOMBRE DEL DATO	TIPO DE DATO	LONGITUD
código_producto	BIGINT	20
nombre_producto	VARCHAR	50
Nitproveedor	BIGINT	20
precio_compra	DOUBLE	
Ivacompra	DOUBLE	
precio_venta	DOUBLE	

El sistema debe validar que el archivo sea un archivo separado por comas (CSV), para que sea cargado correctamente. Se anexa con el presente documento un archivo de muestra de cómo debe conformarse, y una tabla de cómo de ser el formato de cada dato.

```

codigo_producto,nombre_producto,nit_proveedor,precio_compra,iva,precio_venta
1,Melocotones,1,25505,19,30351
2,Manzanas,3,18108,19,21549
3,Plátanos,4,29681,19,35320
4,Lechuga,3,29788,19,35448
5,Tomates,1,12739,19,15159
6,Calabaza,1,21315,19,25365
7,Apio,2,19249,19,22906
8,Pepino,2,10958,19,13040
9,Champiñones,2,11046,19,13145
10,Leche,5,21150,19,25169
11,Queso,5,26571,19,31619
12,Huevos,2,12445,19,14810
13,Requesón,1,14329,19,17052

```

1	Melocotones	1	25505	19	30351
2	Manzanas	3	18108	19	21549
3	Plátanos	4	29681	19	35320
4	Lechuga	3	29788	19	35448
5	Tomates	1	12739	19	15159
6	Calabaza	1	21315	19	25365
7	Apio	2	19249	19	22906
8	Pepino	2	10958	19	13040
9	Champiñones	2	11046	19	13145
10	Leche	5	21150	19	25169
11	Queso	5	26571	19	31619
12	Huevos	2	12445	19	14810
13	Requesón	1	14329	19	17052

14	Crema agria	1	14856	19	17679
15	Yogur	5	14941	19	17780
16	Ternera	5	29335	19	34909
17	Salmón salvaje	5	11878	19	14135
18	Patatas de cangrejo	1	29951	19	35642

Una vez hecha esta validación, el archivo será cargado desde la página Web, y se almacenará en una tabla llamada **productos**, la cual tendrá, por supuesto, los mismos tipos de datos del archivo. El sistema deberá validar que el NIT del proveedor que se indica en el archivo, corresponda a un NIT que exista en la base de datos

## 6. MÓDULO DE GESTIÓN DE VENTAS.

EL EQUIPO deberá desarrollar el módulo de gestión de las ventas que se realicen en la tienda. El sistema, buscará los datos del cliente por cédula. Posteriormente, el sistema permitirá escribir el código del producto, y se visualizará el nombre de este en pantalla. Se digitará la cantidad a vender, y generará el valor total de venta por producto. El sistema permitirá que se realice la misma operación con otros productos hasta que se le dé la opción de totalizar. En ese momento, el sistema calculará el valor del total de IVA para tres (3) productos, según el valor de IVA definido para cada producto, y luego dará el valor total con IVA.

El sistema deberá registrar la venta con los siguientes datos: código de venta – este es un valor consecutivo, cedula del cliente, cedula del usuario, valor total de venta, valor de IVA, y valor total más IVA en una tabla llamada ventas. Deberá, además, guardar en una tabla llamada detalleVentas el detalle de los productos vendidos de esta venta con los siguientes datos: código de producto, cantidad, valor unitario (valor de venta), y valor total, junto con el código de la venta. Finalmente, el sistema deberá mostrar un mensaje de confirmación de la transacción.

## 7. MÓDULO DE CONSULTAS Y REPORTES

EL EQUIPO desarrollará el módulo de consultas del sistema, el cual deberá poder generar las siguientes como mínimo: a) listado de usuarios, b) listado de clientes, y, c) total de ventas por cliente. Las consultas deberán ser por pantalla.

## Parte 2: Organización del Equipo Scrum

EL EQUIPO se conformará por cinco (5) beneficiarios que asistan al Ciclo 3, el cual realizará el desarrollo del software objeto de este documento, mediante la aplicación del marco de trabajo Scrum, para lo cual, los miembros de este tendrán los siguientes roles y responsabilidades:

Nº	ROL	RESPONSABILIDAD
1	Product Owner	Maneja la definición funcional del producto de software completo a desarrollar, y será el punto de contacto con el cliente para validar la aceptación de cada incremento de producto con el cliente.
2	Scrum Master	Maneja, difunde y controla la ejecución correcta del marco de trabajo Scrum en todos los miembros del EQUIPO.
3	Development Team: Arquitecto	Realiza el diseño de arquitectura del software y de la base de datos.
4	Development Team: Desarrollador	Realiza la codificación del software.
5	Development Team: Control de Calidad (QA)	Realiza las pruebas al software establecidas en este documento.

Cabe mencionar que, por la naturaleza del proyecto y el número de participantes del equipo, todos ellos (as) tienen el rol de desarrolladores por defecto, de lo cual, en el momento que se estime necesario el EQUIPO podrá sumar más desarrolladores, conforme a los requerimientos y la ejecución del sprint. El Scrum Master tendrá la responsabilidad de realizar el seguimiento y control del proyecto, para lo cual, convocará las reuniones diarias de seguimiento (*daily standup meeting*), en coordinación con el formador, y se utilizará la herramienta Trello, para realizar el seguimiento del desarrollo de los sprints.

## Parte 3: Organización de los Sprints

### Sprint 1

#### MÓDULO DE LOGIN DEL SISTEMA

EL EQUIPO deberá desarrollar el módulo que permita el ingreso al sistema, una vez se haya realizado la validación por nombre de usuario y contraseña. Deberá tenerse un usuario por defecto con el nombre de **admininicial**, y contraseña **admin123456** para su ingreso la primera vez.

#### MÓDULO DE GESTIÓN DE USUARIOS


EL EQUIPO deberá desarrollar el módulo de gestión de los usuarios que trabajen en la tienda para que operen el sistema. Se deberán tener las opciones de crear usuario, actualizar datos de usuario, y borrar usuario. Estos datos se almacenarán en la tabla llamada **usuarios**, y los datos a gestionar son: cedula, nombre completo, correo electrónico, usuario, y contraseña.



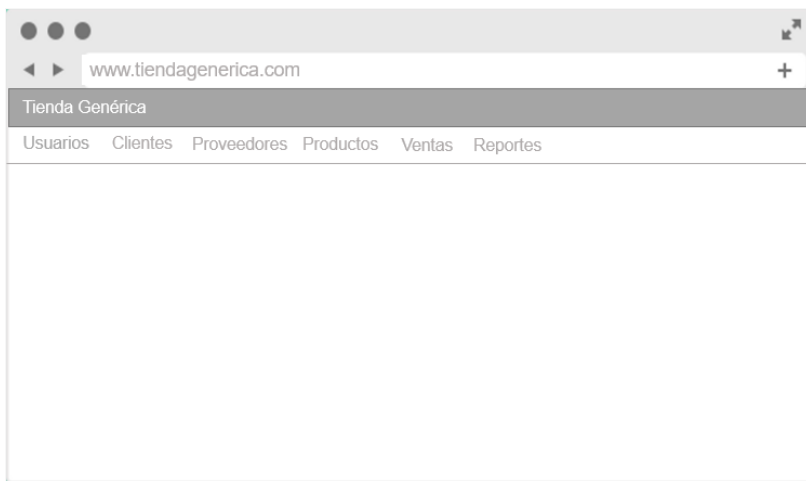
## Historias de Usuario

Identificador (ID) de la historia	Rol	Característica / Funcionalidad	Razón / Resultado
HU-001	Como administrador	Necesito ingresar al sistema con el usuario inicial <b>admininicial</b> y la contraseña <b>admin123456</b> .	Con la finalidad de validar el funcionamiento del login del sistema, la interfaz gráfica y que el usuario inicial opere correctamente.
HU-002	Como administrador	Necesito crear nuevos usuarios que entrarán al sistema	Con la finalidad de cargar los usuarios que harán las labores de ingreso de datos para los demás módulos del sistema.
HU-003	Como administrador	Necesito consultar los datos del usuario, por medio de la cédula	Con la finalidad de poder recuperar los datos de un usuario creado.
HU-004	Como administrador	Necesito actualizar los datos de los usuarios que estén en el sistema	Con la finalidad de poder actualizar los datos de nombre completo, correo electrónico, usuario y contraseña, previa consulta por cédula.
HU-005	Como administrador	Necesito borrar los datos de usuarios que estén en el sistema	Con la finalidad de borrar los datos de los usuarios que ya no deben tener ingreso al sistema, previa consulta por cédula.

## Interfaz Gráfica



The image shows a web browser window with the address bar displaying "www.tiendagenerica.com". The main content area has a heading "Bienvenidos a la Tienda Genérica". Below the heading are two input fields: "Usuario" and "Contraseña". At the bottom of the form are two buttons: "Aceptar" and "Cancelar".



## Conjunto de Pruebas

ID Prueba	Caso de Prueba	Descripción	Acción de Entrada	Resultado Esperado
SP1-QA-1	Ingreso correcto	El usuario debe poder registrarse correctamente, si los campos de usuario y contraseña son correctos.	Escribir el nombre de usuario inicial <b>admininicial</b> y/o usuario creado, y la contraseña correspondiente.	Si es el usuario <b>admininicial</b> , o un usuario ya creado, el sistema dará ingreso al menú general.
SP1-QA-2	Ingreso incorrecto por error en el usuario y/o contraseña.	El usuario debe recibir un mensaje de error si, al consultar en la base de datos no se recupera la información, o si los datos	Escribir el usuario y/o contraseña con errores de tipeo, o no escribir alguno de los datos solicitados.	Debe generar un mensaje de error con el error de "usuario o contraseña errados, intente de nuevo".

		de <b>admininicial</b> son errados.		
SP1-QA-3	Creación de un nuevo usuario correcto	El usuario debe poder crear un nuevo usuario en el sistema, si todos los datos del nuevo usuario están completos.	Escribir los datos de cédula, nombre completo, correo electrónico, usuario y contraseña del nuevo usuario, posteriormente, se deberá presionar el botón "Crear".	Debe genera un mensaje de información "Usuario Creado". Luego, debe insertar los datos en la base de datos y, limpiar todos los datos ya creados.
SP1-QA-4	Creación de un nuevo usuario con errores, por falta de completitud de algunos de los campos.	El usuario debe recibir un mensaje de error, si no se diligencian todos los datos del nuevo usuario.	Escribir los datos del nuevo usuario, con algún faltante.	Debe genera un mensaje de error "Faltan datos del usuario", posteriormente, debe limpiar todos los datos ya creados.
SP1-QA-5	Consulta de usuario existente.	El usuario escribe la cédula de otro usuario en el campo "Cédula", y éste se encuentra en la base de datos.	Escribir la cédula del usuario a consultar.	El sistema retorna todos los demás datos de nombre completo, correo electrónico, usuario y contraseña.
SP1-QA-6	Consulta de usuario inexistente o con errores.	El usuario debe recibir un mensaje de error.	Escribir la cédula del usuario a consultar.	Debe genera un mensaje de error "Usuario Inexistente", posteriormente, debe limpiar todos los datos ya creados.
SP1-QA-7	Actualización Correcta de datos de usuario	El usuario, actualizará los datos de nombre completo y/o correo electrónico, y/o usuario, y/o contraseña.	Escribir la cédula del usuario a consultar para ser actualizado.	Debe genera un mensaje de información "Datos del Usuario Actualizados". Luego, debe actualizar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.
SP1-QA-8	Actualización de datos de usuario con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de cédula, nombre completo, correo electrónico, usuario y/o contraseña.	Escribir la cédula del usuario a consultar, y borrar u omitir algún tipo de dato de los recuperados del usuario.	Debe genera un mensaje de error "Datos faltantes", posteriormente, debe limpiar todos los datos ya creados.
SP1-QA-9	Borrado correcto de datos de usuario	El usuario, podrá borrar los datos de cédula, nombre completo y/o correo electrónico, y/o usuario, y/o contraseña.	Escribir la cédula del usuario a consultar para ser borrado.	Debe consultar la base de datos por la existencia del usuario con la cédula indicada. Debe genera un mensaje de información "Datos del Usuario Borrados". Luego, debe borrar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.
SP1-QA-10	Borrado de datos de usuario con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de cédula, nombre completo, correo	Escribir la cédula del usuario a consultar, y alterar la cédula y/o borrarla.	Debe consultar la base de datos por la existencia del usuario con la cédula indicada, y debe generar un mensaje de error "Cédula Errada",

		electrónico, usuario y/o contraseña.		posteriormente, debe limpiar todos los datos ya creados.
--	--	--------------------------------------	--	--

## Sprint 2

### MÓDULO DE GESTIÓN DE CLIENTES

EL EQUIPO deberá desarrollar el módulo de gestión de los clientes de la tienda, para lo cual se deberán tener las opciones de crear cliente, leer clientes, actualizar datos de cliente, y borrar cliente. Estos datos se almacenarán en la tabla llamada **clientes**, y los datos a gestionar son: cedula, nombre completo, dirección, teléfono, y correo electrónico.

### MÓDULO DE GESTIÓN DE PROVEEDORES.

EL EQUIPO deberá desarrollar el módulo de gestión de los proveedores de la tienda, para lo cual se deberán tener las opciones de crear proveedor, leer proveedor, actualizar datos de proveedor, y borrar proveedor. Estos datos se almacenarán en una tabla llamada **proveedores**, y los datos a gestionar son: NIT, nombre proveedor, dirección, teléfono, y ciudad.

### Historias de Usuario

Identificador (ID) de la historia	Rol	Característica / Funcionalidad	Razón / Resultado
HU-006	Como administrador	Necesito crear nuevos clientes	Con la finalidad de registrar a los clientes que realicen compras en la tienda
HU-007	Como administrador	Necesito consultar los datos del cliente, por medio de la cédula	Con la finalidad de poder recuperar los datos de un cliente creado.
HU-008	Como administrador	Necesito actualizar los datos de los clientes que estén en el sistema	Con la finalidad de poder actualizar los datos de nombre completo, dirección, teléfono, y correo electrónico, previa consulta por cédula.
HU-009	Como administrador	Necesito borrar los datos de clientes que estén en el sistema	Con la finalidad de borrar los datos de los clientes que ya no deben tener ingreso al sistema, previa consulta por cédula.
HU-010	Como administrador	Necesito crear nuevos proveedores que entrarán al sistema	Con la finalidad de cargar los proveedores de la tienda.
HU-011	Como administrador	Necesito consultar los datos del proveedor, por medio de del NIT	Con la finalidad de poder recuperar los datos de un usuario creado.
HU-012	Como administrador	Necesito actualizar los datos de los proveedores que estén en el sistema	Con la finalidad de poder actualizar los datos de nombre de

			proveedor, Dirección, teléfono y ciudad, previa consulta por NIT.
HU-013	Como administrador	Necesito borrar los datos de un proveedor que estén en el sistema	Con la finalidad de borrar los datos de los proveedores que ya no deben tener ingreso al sistema, previa consulta por NIT.

## Interfaz Gráfica

www.tiendagenerica.com

Tienda Genérica

Usuarios **Clientes** Proveedores Productos Ventas Reportes

Cédula  Teléfono

Nombre Completo  Correo Electrónico

Dirección

Consultar Crear Actualizar Borrar

www.tiendagenerica.com

Tienda Genérica

Usuarios Clientes **Proveedores** Productos Ventas Reportes

NIT  Teléfono

Nombre Proveedor  Ciudad

Dirección

Consultar Crear Actualizar Borrar

## Conjunto de Pruebas

ID Prueba	Caso de Prueba	Descripción	Acción de Entrada	Resultado Esperado
SP2-QA-1	Creación de un nuevo cliente correcto	El usuario debe poder crear un nuevo cliente en el sistema, si todos los datos del nuevo cliente están completos.	Escribir los datos de cédula, nombre completo, dirección, teléfono, y correo electrónico del nuevo cliente, posteriormente, se deberá presionar el botón "Crear".	Debe genera un mensaje de información "Cliente Creado". Luego, debe insertar los datos en la base de datos y, limpiar todos los datos ya creados.
SP2-QA-2	Creación de un nuevo cliente con errores, por falta de completitud de algunos de los campos.	El usuario debe recibir un mensaje de error, si no se diligencian todos los datos del nuevo cliente.	Escribir los datos del nuevo cliente con algún faltante.	Debe genera un mensaje de error "Faltan datos del cliente", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-3	Consulta de cliente existente.	El usuario escribe la cédula del cliente en el campo "Cédula", y éste se encuentra en la base de datos.	Escribir la cédula del cliente a consultar.	El sistema retorna todos los demás datos de cédula, nombre completo, dirección, teléfono, y correo electrónico del cliente.
SP2-QA-4	Consulta de cliente inexistente o con errores.	El usuario debe recibir un mensaje de error.	Escribir la cédula del cliente a consultar.	Debe genera un mensaje de error "Cliente Inexistente", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-5	Actualización Correcta de datos de cliente	El usuario, actualizará los datos de nombre completo, dirección, teléfono, y correo electrónico del cliente.	Escribir la cédula del usuario a consultar para ser actualizado.	Debe genera un mensaje de información "Datos del Cliente Actualizados". Luego, debe actualizar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.
SP2-QA-6	Actualización de datos de cliente con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de cédula, nombre completo, dirección, teléfono, y correo electrónico del cliente.	Escribir la cédula del cliente a consultar, y borrar u omitir algún tipo de dato de los recuperados del cliente.	Debe genera un mensaje de error "Datos faltantes", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-7	Borrado correcto de datos de cliente	El usuario, podrá borrar los datos de nombre completo, dirección, teléfono, y correo electrónico del cliente.	Escribir la cédula del cliente a consultar para ser borrado.	Debe consultar la base de datos por la existencia del cliente con la cédula indicada. Debe genera un mensaje de información "Datos del Cliente Borrados". Luego, debe borrar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.

SP2-QA-8	Borrado de datos de cliente con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de cédula, nombre completo, dirección, teléfono, y correo electrónico del cliente.	Escribir la cédula del cliente a consultar, y alterar la cédula y/o borrarla.	Debe consultar la base de datos por la existencia del cliente con la cédula indicada, y debe generar un mensaje de error "Cédula Errada", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-9	Creación de un nuevo proveedor correcto	El usuario debe poder crear un nuevo proveedor en el sistema, si todos los datos del nuevo proveedor están completos.	Escribir los datos de NIT, Nombre Proveedor, Dirección, Teléfono, y Ciudad, posteriormente, se deberá presionar el botón "Crear".	Debe genera un mensaje de información "Proveedor Creado". Luego, debe insertar los datos en la base de datos y, limpiar todos los datos ya creados.
SP2-QA-10	Creación de un nuevo proveedor con errores, por falta de completitud de algunos de los campos.	El usuario debe recibir un mensaje de error, si no se diligencian todos los datos del nuevo proveedor.	Escribir los datos del nuevo proveedor con algún faltante.	Debe genera un mensaje de error "Faltan datos del proveedor", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-11	Consulta de proveedor existente.	El usuario escribe el NIT del proveedor en el campo "NIT", y éste se encuentra en la base de datos.	Escribir el NIT del proveedor a consultar.	El sistema retorna todos los demás datos de NIT, Nombre Proveedor, Dirección, Teléfono, y Ciudad del proveedor.
SP2-QA-12	Consulta de proveedor inexistente o con errores.	El usuario debe recibir un mensaje de error.	Escribir el NIT del Proveedor a consultar.	Debe genera un mensaje de error "Proveedor Inexistente", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-13	Actualización Correcta de datos del proveedor	El usuario, actualizará los datos de Nombre Proveedor, Dirección, Teléfono, y Ciudad del proveedor.	Escribir el NIT del proveedor a consultar para ser actualizado.	Debe genera un mensaje de información "Datos del Proveedor Actualizados". Luego, debe actualizar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.
SP2-QA-14	Actualización de datos de proveedor con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de Nombre Proveedor, Dirección, Teléfono, y Ciudad del proveedor.	Escribir el NIT del proveedor a consultar, y borrar u omitir algún tipo de dato de los recuperados del proveedor.	Debe genera un mensaje de error "Datos faltantes", posteriormente, debe limpiar todos los datos ya creados.
SP2-QA-15	Borrado correcto de datos de proveedor	El usuario, podrá borrar los datos de Nombre Proveedor, Dirección, Teléfono, y Ciudad del proveedor.	Escribir el NIT del proveedor a consultar para ser borrado.	Debe consultar la base de datos por la existencia del proveedor con la cédula indicada. Debe genera un mensaje de información "Datos del Proveedor Borrados". Luego, debe borrar los mismos los datos en la base de datos y, limpiar todos los datos ya creados en el formulario.

SP2-QA-16	Borrado de datos de proveedor con errores por datos en blanco	El usuario, recibirá un mensaje de error de datos al poner en blanco y/u omitir datos de NIT, Nombre Proveedor, Dirección, Teléfono, y Ciudad del proveedor.	Escribir la cédula del proveedor a consultar, y alterar el NIT y/o borrarlo.	Debe consultar la base de datos por la existencia del proveedor con la cédula indicada, y debe generar un mensaje de error "NIT Errado", posteriormente, debe limpiar todos los datos ya creados.
-----------	---	--	--	---

## Sprint 3

### MÓDULO DE GESTIÓN DE PRODUCTOS.

EL EQUIPO deberá desarrollar el módulo de gestión de los productos que se venden en la tienda, para lo cual se deberá cargar estos productos de un archivo plano (de texto), con la siguiente estructura de datos:

NOMBRE DEL DATO	TIPO DE DATO	LONGITUD
código_producto	BIGINT	20
nombre_producto	VARCHAR	50
nitproveedor	BIGINT	20
precio_compra	DOUBLE	
ivacompra	DOUBLE	
precio_venta	DOUBLE	

El sistema debe validar que el archivo sea un archivo separado por comas (CSV), para que sea cargado correctamente. Se anexa con el presente documento un archivo de muestra de cómo debe conformarse, y una tabla de cómo de ser el formato de cada dato.

código_producto	nombre_producto	nitproveedor	precio_compra	ivacompra	precio_venta
1	Melocotones	1	25505	19	30351
2	Manzanas	3	18108	19	21549
3	Plátanos	4	29681	19	35320
4	Lechuga	3	29788	19	35448
5	Tomates	1	12739	19	15159
6	Calabaza	1	21315	19	25365
7	Apio	2	19249	19	22906
8	Pepino	2	10958	19	13040
9	Champiñones	2	11046	19	13145
10	Leche	5	21150	19	25169

1	Melocotones	1	25505	19	30351
2	Manzanas	3	18108	19	21549
3	Plátanos	4	29681	19	35320
4	Lechuga	3	29788	19	35448
5	Tomates	1	12739	19	15159
6	Calabaza	1	21315	19	25365
7	Apio	2	19249	19	22906
8	Pepino	2	10958	19	13040
9	Champiñones	2	11046	19	13145
10	Leche	5	21150	19	25169



11	Queso	5	26571	19	31619
12	Huevos	2	12445	19	14810
13	Requesón	1	14329	19	17052
14	Crema agria	1	14856	19	17679
15	Yogur	5	14941	19	17780
16	Ternera	5	29335	19	34909
17	Salmón salvaje	5	11878	19	14135
18	Patas de cangrejo	1	29951	19	35642

Una vez hecha esta validación, el archivo será cargado desde la página Web, y se almacenará en una tabla llamada **productos**, la cual tendrá, por supuesto, los mismos tipos de datos del archivo. El sistema deberá validar que el NIT del proveedor que se indica en el archivo, corresponda a un NIT que exista en la base de datos.

## Historias de Usuario

Identificador (ID) de la historia	Rol	Característica / Funcionalidad	Razón / Resultado
HU-014	Como administrador	Necesito subir al sistema un archivo texto separado por comas (csv) con los datos de productos, según formato definido (1).	Con la finalidad de cargar a la tabla de productos del sistema, reemplazando los productos que hubiere previamente, y poderlos utilizar para realizar ventas.

(1) Indicado previamente en la descripción funcional

## Interfaz Gráfica

## Conjunto de Pruebas

ID Prueba	Caso de Prueba	Descripción	Acción de Entrada	Resultado Esperado
SP3-QA-1	Carga exitosa del archivo	El usuario realiza una carga de archivo correcta.	El usuario seleccionará un archivo al presionar el botón "Examinar", el archivo corresponde al formato CSV, y tiene la estructura de datos correcta.	Debe generar un mensaje de información "Archivo Cargado Exitosamente". Luego, debe borrar los anteriores datos que tenga la tabla Productos, e insertar los datos leídos, previa validación.
SP3-QA-2	Carga Fallida del archivo por falta de nombre de archivo	El usuario intenta realizar una carga sin archivo.	El usuario oprime el botón "Cargar" sin haber seleccionado previamente un archivo.	Debe generar un mensaje de error "Error: no se seleccionó archivo para cargar".
SP3-QA-3	Carga Fallida del archivo por errores de formato de archivo	El usuario carga un archivo inválido para ser reconocido por el sistema.	El usuario intenta cargar un archivo que tiene errores en el formato de creación de este.	Debe generar un mensaje de error "Error: formato de archivo inválido".
SP3-QA-4	Carga Fallida del archivo por errores en la validación de los datos cargados	El usuario carga un archivo con datos inválidos.	El usuario intenta cargar un archivo con errores en los tipos de datos.	Debe generar un mensaje de error "Error: datos leídos inválidos".

## Sprint 4

### MÓDULO DE GESTIÓN DE VENTAS.

EL EQUIPO deberá desarrollar el módulo de gestión de las ventas que se realicen en la tienda. El sistema, buscará los datos del cliente por cédula. Posteriormente, el sistema permitirá escribir el código del producto, y se visualizará el nombre de este en pantalla. Se digitará la cantidad a vender, y generará el valor total de venta por producto. El sistema permitirá que se realice la misma operación con otros productos hasta que se le dé la opción de totalizar. En ese momento, el sistema calculará el valor del

total de IVA para tres (3) productos, según el valor de IVA definido para cada producto, y luego dará el valor total con IVA.

El sistema deberá registrar la venta con los siguientes datos: código de venta – este es un valor consecutivo, cedula del cliente, cedula del usuario, valor total de venta, valor de IVA, y valor total más IVA en una tabla llamada **ventas**. Deberá, además, guardar en una tabla llamada **detalleVentas** el detalle de los productos vendidos de esta venta con los siguientes datos: código de producto, cantidad, valor unitario (valor de venta), y valor total, junto con el código de la venta. Finalmente, el sistema deberá mostrar un mensaje de confirmación de la transacción.

## Historias de Usuario

Identificador (ID) de la historia	Rol	Característica / Funcionalidad	Razón / Resultado
HU-015	Como usuario	Necesito registrar las ventas de la tienda para un cliente.	Con la finalidad de ingresar los datos de una nueva venta de la tienda.
HU-016	Como usuario	Necesito obtener los datos del cliente mediante consulta por cédula.	Con la finalidad de cargarlos al formulario de ventas, al momento que se desee registrar una venta para este cliente.
HU-017	Como usuario	Necesito obtener los datos de los productos mediante consulta por código de producto.	Con la finalidad de cargarlos al formulario de ventas, al momento que se desee registrar una venta para este cliente.
HU-018	Como usuario	Necesito calcular el valor de venta por cada producto vendido, mediante la cantidad vendida, hasta un total de tres (3) productos.	Con la finalidad de acumular el total de venta para los valores totales de los productos.
HU-019	Como usuario	Necesito generar el valor de la venta total para el cliente, incluyendo IVA y total con IVA.	Con la finalidad de presentar al cliente los valores totales de venta e IVA al cliente.
HU-020	Como usuario	Necesito confirmar el valor de venta, dada la aceptación del cliente.	Con la finalidad registrar en la base de datos de ventas de la tienda.

## Interfaz Gráfica

## Conjunto de Pruebas

ID Prueba	Caso de Prueba	Descripción	Acción de Entrada	Resultado Esperado
SP4-QA-1	Consulta exitosa de la cédula del cliente	El usuario realiza una consulta de cédula del cliente que recupera el nombre de este de forma correcta.	El usuario digita la cédula del cliente y oprime el botón "Consultar".	Debe recuperar el nombre del cliente de la base de datos de forma correcta, desplegándolo en el espacio respectivo.
SP4-QA-2	Consulta fallida de la cédula del cliente	El usuario realiza una consulta de cédula del cliente que no se encuentra en la base de datos.	El usuario digita una cédula de cliente y oprime el botón "Consultar".	Debe generar un mensaje por pantalla que indique que la cédula no se encuentra registrada en la base de datos.
SP4-QA-3	Consulta exitosa de producto	El usuario realiza una consulta del código de producto y se recupera de forma correcta.	El usuario digita el código del producto para la venta y oprime el botón "Consultar".	Debe recuperar el nombre del producto de la base de datos de forma correcta, desplegándolo en el espacio respectivo.
SP4-QA-4	Consulta fallida de producto	El usuario realiza una consulta de código de producto que no se encuentra en la base de datos.	El usuario digita un código de producto y oprime el botón "Consultar".	Debe generar un mensaje por pantalla que indique que el código de producto no se encuentra registrada en la base de datos.
SP4-QA-5	Validación del campo cantidad de productos	El sistema realiza una validación exitosa de la cantidad de productos, donde el dato sea inferior a cero, o nulo	El usuario digita una cantidad de producto inferior a cero, o valor nulo, cuando el usuario presiona el botón "Confirmar"	Debe generar un mensaje por pantalla que indique que el valor de cantidad es incorrecto.
SP4-QA-6	Validación del campo valor total por producto	El sistema realiza una validación exitosa del valor total por cada producto de los tres permitidos.	El sistema calcula de forma correcta el valor total de cada producto (cantidad x precio de venta), cuando el usuario	Debe muestra el valor correcto en el campo correspondiente de valor total por producto.

			presiona el botón "Confirmar"	
SP4-QA-7	Validación del campo Total Venta	El sistema realiza una validación exitosa del total de la venta de todos los tres (3) productos	El sistema calcula de forma correcta el valor total todos los productos, cuando el usuario presiona el botón "Confirmar"	Debe muestra el valor correcto en el campo correspondiente de valor total por producto.
SP4-QA-8	Validación del campo Total IVA	El sistema realiza una validación exitosa del total del IVA de todos los tres (3) productos	El sistema calcula de forma correcta el valor del IVA de los tres (3) productos, cuando el usuario presiona el botón "Confirmar"	Debe muestra el valor correcto en el campo correspondiente de valor total del IVA.
SP4-QA-9	Validación del campo Total con IVA	El sistema realiza una validación exitosa del total de la venta, incluyendo el valor del IVA de todos los tres (3) productos	El sistema calcula de forma correcta el valor total de la venta, incluyendo el IVA de los tres (3) productos, cuando el usuario presiona el botón "Confirmar"	Debe muestra el valor correcto en el campo correspondiente de valor total de la venta incluyendo el IVA.
SP4-QA-10	El consecutivo de la venta se genera exitosamente	Al momento de confirmar la venta, se genera un número consecutivo de esta.	El usuario presiona el botón "Confirmar"	Se debe imprimir un número consecutivo en el campo destinado para tal fin.

## Sprint 5

### MODULO DE CONSULTAS Y REPORTES

EL EQUIPO desarrollará el módulo de consultas del sistema, el cual deberá poder generar las siguientes como mínimo: a) listado de usuarios, b) listado de clientes, y, c) total de ventas por cliente. Las consultas deberán ser por pantalla.

### Historias de Usuario

Identificador (ID) de la historia	Rol	Característica / Funcionalidad	Razón / Resultado
HU-021	Como usuario	Necesito consultar el listado de usuarios del sistema.	Con la finalidad de verificar quienes son los usuarios registrados.
HU-022	Como usuario	Necesito consultar el listado de clientes	Con la finalidad de verificar cuales clientes han comprado en la tienda.
HU-023	Como usuario	Necesito consultar el total de ventas por cliente	Con la finalidad de contabilizar las ventas por cliente.

## Interfaz Gráfica





## Conjunto de Pruebas

ID Prueba	Caso de Prueba	Descripción	Acción de Entrada	Resultado Esperado
SP5-QA-1	Generación de listado de usuarios exitosa	El usuario, al presionar el botón "listado de usuarios", puede ver el	El usuario presiona el botón "Listado de Clientes"	Debe llevar a una página nueva donde se listen los datos de cedula, nombre, email, usuario y Password.

		listado de usuarios existente.		Si no existen usuarios, se deberá generar un mensaje.
SP5-QA-2	Generación de listado de clientes exitosa	El usuario, al presionar el botón "listado de clientes", puede ver el listado de clientes existente.	El usuario presiona el botón "Listado de Usuarios"	Debe llevar a una página nueva donde se listen los datos de cedula, nombre, email, dirección y teléfono. Si no existen clientes, se deberá generar un mensaje.
SP5-QA-3	Generación de Total de Ventas por cliente exitosa	El usuario, al presionar el botón "Total ventas por Cliente", puede ver el listado de clientes con el total de ventas y el total consolidado.	El usuario presiona el botón "Total ventas por Cliente"	Debe llevar a una página nueva donde se listen los datos de cedula, y del cliente, y el valor total de ventas de cada cliente. Al final deberá totalizar el valor de todas las ventas desplegadas, que corresponden al total indicado por cliente Si no existen clientes, se deberá generar un mensaje.

## Parte 4: Especificaciones Técnicas

### Lenguajes de programación y stack tecnológico de desarrollo

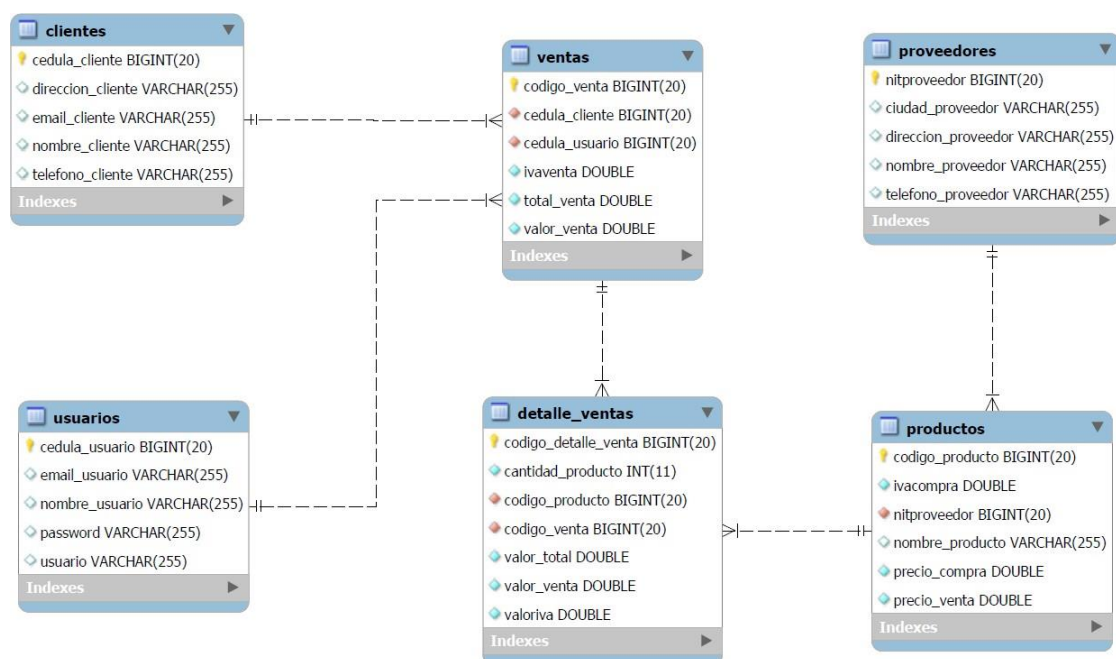
Los lenguajes de programación, frameworks, y herramientas tecnológicas que se emplearán para el desarrollo de este proyecto son los siguientes:

- 1) Java EE para desarrollo Web. JDK: JavaSE 11.0.2
- 2) Ambiente Integrado de Desarrollo (IDE) Eclipse for Enterprise Java and Web Developers Versión 2021-06
- 3) Servidor de Aplicaciones Web para Java Apache Tomcat® - Apache Tomcat 9
- 4) Gestor de Base de datos MySQL 8.0.24
- 5) Cliente gestor de bases de datos MySQL Workbench Community Edition 8.0.25.
- 6) Repositorio Github, e integración con Eclipse.



## Especificación de diseño de la base de datos

Se presenta a continuación el diseño de la base de datos de la tienda genérica:



## Especificación de la API para conexión del Frontend con el Backend de la aplicación

### 1. APIs para gestión de Clientes.

#### Actualizar Clientes

**PUT** /clientes/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
cliente	(required)	cliente	body	Model

Parameter content type: **application/json**

**Example Value**

```
{  "cedula_cliente": 0,  "direccion_cliente": "string",  "email_cliente": "string",  "nombre_cliente": "string",  "telefono_cliente": "string"}
```

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

#### Eliminar Clientes

**DELETE** /clientes/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
id	(required)	id	path	integer

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!

## Agregar Clientes

**POST** /clientes/guardar guardar

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
cliente	(required) <div></div> <div>Parameter content type: <span>application/json</span></div>	cliente	body	<div>Model   Example Value</div> <pre>{  "cedula_cliente": 0,  "direccion_cliente": "string",  "email_cliente": "string",  "nombre_cliente": "string",  "telefono_cliente": "string"}</pre>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Clientes

**GET** /clientes/listar listar

**Response Class (Status 200)**  
OK

Model | Example Value

```
[  {    "cedula_cliente": 0,    "direccion_cliente": "string",    "email_cliente": "string",    "nombre_cliente": "string",    "telefono_cliente": "string"  }]
```

**Response Content Type** \*/\*

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## 2. APIs para Detalle de Ventas

### Actualizar Detalle de Ventas

**PUT** /detalleventas/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>detalleVenta</b>	(required) <div></div> <div>Parameter content type: <span>application/json</span></div>	<b>detalleVenta</b>	<b>body</b>	<div>Model   <b>Example Value</b></div> <pre>{  "cantidad_producto": 0,  "codigo_detalle_venta": 0,  "codigo_producto": 0,  "codigo_venta": 0,  "valor_total": 0,  "valor_venta": 0,  "valor_iva": 0}</pre>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

### Eliminar Detalle de Ventas

**DELETE** /detalleventas/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>id</b>	(required) <div></div>	<b>id</b>	<b>path</b>	<b>integer</b>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!

## Guardar Detalle de Ventas

**POST** /detalleventas/guardar guardar

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
detalleDetalleVenta	(required)	detalleDetalleVenta	body	Model

Parameter content type: application/json

**Example Value**

```
{  "cantidad_producto": 0,  "codigo_detalle_venta": 0,  "codigo_producto": 0,  "codigo_venta": 0,  "valor_total": 0,  "valor_venta": 0,  "valoriva": 0}
```

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Detalle de Ventas

**GET** /detalleventas/listar listar

Response Class (Status 200)  
OK

**Model** | **Example Value**

```
[  {    "cantidad_producto": 0,    "codigo_detalle_venta": 0,    "codigo_producto": 0,    "codigo_venta": 0,    "valor_total": 0,    "valor_venta": 0,    "valoriva": 0  }]
```

Response Content Type \*/\*

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

### 3. APIs para Productos

#### Actualizar Productos

**PUT** /productos/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
producto	(required)	producto	body	Model

Parameter content type: **application/json**

**Example Value**

```
{  "codigo_producto": 0,  "ivacompra": 0,  "nitproveedor": 0,  "nombre_producto": "string",  "precio_compra": 0,  "precio_venta": 0}
```

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

#### Eliminar Productos

**DELETE** /productos/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
id	(required)	id	path	integer

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!

## Guardar Productos

**POST** /productos/guardar **guardar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
producto	(required) <div></div> <div>Parameter content type: <b>application/json</b></div>	producto	body	<div>Model</div> <div>Example Value</div> <pre>{  "codigo_producto": 0,  "ivacompra": 0,  "nitproveedor": 0,  "nombre_producto": "string",  "precio_compra": 0,  "precio_venta": 0}</pre>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Productos

**GET** /productos/listar **listar**

**Response Class (Status 200)**  
OK

Model

Example Value

```
[  {    "codigo_producto": 0,    "ivacompra": 0,    "nitproveedor": 0,    "nombre_producto": "string",    "precio_compra": 0,    "precio_venta": 0  }]
```

**Response Content Type** \*/\* **▼**

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## 4. APIs para Proveedores

### Actualizar Proveedores

**PUT** /proveedores/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>proveedor</b>	(required) <div></div>	<b>proveedor</b>	<b>body</b>	Model Example Value <pre>{   "ciudad_proveedor": "string",   "direccion_proveedor": "string",   "nitproveedor": 0,   "nombre_proveedor": "string",   "telefono_proveedor": "string" }</pre>

Parameter content type: **application/json**

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

### Eliminar Proveedores

**DELETE** /proveedores/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>id</b>	(required) <div></div>	<b>id</b>	<b>path</b>	<b>integer</b>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!



## Agregar Proveedores

**POST** **/proveedores/guardar** **guardar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>proveedor</b>	(required) <div></div> <div>Parameter content type: <b>application/json</b></div>	<b>proveedor</b>	<b>body</b>	<div>Model   Example Value</div> <div><pre>{   "ciudad_proveedor": "string",   "direccion_proveedor": "string",   "nitproveedor": 0,   "nombre_proveedor": "string",   "telefono_proveedor": "string" }</pre></div>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Proveedores

**GET** **/proveedores/listar** **listar**

**Response Class (Status 200)**  
OK

Model | Example Value

```
[
  {
    "ciudad_proveedor": "string",
    "direccion_proveedor": "string",
    "nitproveedor": 0,
    "nombre_proveedor": "string",
    "telefono_proveedor": "string"
  }
]
```

**Response Content Type** **\*/\***

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## 5. APIs de Usuarios

### Actualizar Usuarios

**PUT** /usuarios/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>usuario</b>	(required) <div></div>	<b>usuario</b>	<b>body</b>	<div>Model   Example Value</div> <pre>{   "cedula_usuario": 0,   "email_usuario": "string",   "nombre_usuario": "string",   "password": "string",   "usuario": "string" }</pre>

Parameter content type: **application/json**

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

### Eliminar Usuarios

**DELETE** /usuarios/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>id</b>	(required) <div></div>	<b>id</b>	<b>path</b>	<b>integer</b>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!

## Agregar Usuarios

**POST** /usuarios/guardar **guardar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
usuario	(required) <div></div> <div>Parameter content type: application/json ▼</div>	usuario	body	<div>Model   Example Value</div> <div><pre>{   "cedula_usuario": 0,   "email_usuario": "string",   "nombre_usuario": "string",   "password": "string",   "usuario": "string" }</pre></div>

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Usuarios

**GET** /usuarios/listar **listar**

**Response Class (Status 200)**  
OK

Model | Example Value

```
{
  {
    "cedula_usuario": 0,
    "email_usuario": "string",
    "nombre_usuario": "string",
    "password": "string",
    "usuario": "string"
  }
}
```

Response Content Type \*/.\* ▼

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## 6. APIs de Ventas

### Actualizar Ventas

**PUT** /Ventas/actualizar **actualizar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
venta	(required)	venta	body	Model

Parameter content type: **application/json**

**Example Value**

```
{  "cedula_cliente": 0,  "cedula_usuario": 0,  "codigo_venta": 0,  "ivaventa": 0,  "total_venta": 0,  "valor_venta": 0}
```

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

### Eliminar Ventas

**DELETE** /Ventas/eliminar/{id} **eliminar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
id	(required)	id	path	integer

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
204	No Content		
401	Unauthorized		
403	Forbidden		

Try it out!

## Agregar Ventas

**POST** /Ventas/guardar **guardar**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
venta	(required)	venta	body	Model

Parameter content type: **application/json**

**Example Value**

```
{  "cedula_cliente": 0,  "cedula_usuario": 0,  "codigo_venta": 0,  "ivaventa": 0,  "total_venta": 0,  "valor_venta": 0}
```

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200	OK		
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

## Listar Ventas

**GET** /Ventas/listar **listar**

**Response Class (Status 200)**  
OK

**Model** | **Example Value**

```
[  {    "cedula_cliente": 0,    "cedula_usuario": 0,    "codigo_venta": 0,    "ivaventa": 0,    "total_venta": 0,    "valor_venta": 0  }]
```

**Response Content Type** **\*/\***

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

