**SISTEMA DE GESTIÓN VENTAS Y CONTROL DE INVENTARIO (Tienda Virtual)**

**Versión 1.0**

**Fecha: [09/01/2025]**

**CONTROL DE VERSIONES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Fecha** | **Responsable** | **Descripción** |
| 1.0 | 09/01/2025 | Equipo Dev | Versión inicial del documento |

**TABLA DE CONTENIDOS**

1. VISIÓN DEL PROYECTO
2. ALCANCE Y OBJETIVOS
3. ARQUITECTURA DEL SISTEMA
4. HISTORIAS DE USUARIO
5. ANEXOS
6. **VISIÓN DEL PROYECTO**
   1. **DESCRIPCIÓN GENERAL**

El Sistema de Ventas y Control de Inventario es una solución integral diseñada para facilitar la gestión de ventas y el control de inventarios en una tienda virtual. Este sistema permite a los usuarios administrar productos, realizar transacciones, generar reportes de ventas, y mantener un control preciso del stock, asegurando eficiencia operativa y satisfacción del cliente.

* 1. **OBJETIVOS PRINCIPALES**
* Optimizar la gestión de ventas y productos.
* Facilitar el seguimiento en tiempo real del inventario.
* Proporcionar reportes detallados de ventas y stock.
* Mejorar la experiencia de compra en línea.
* Garantizar la seguridad de la información de los clientes y transacciones.
  1. **STAKEHOLDERS**
* Propietarios de la tienda.
* Administradores del sistema.
* Clientes finales.
* Proveedores.
* Equipo de desarrollo.

1. **ALCANCE Y OBJETIVOS**
   1. **MÓDULOS DEL SISTEMA**
      1. **Gestión de Productos**

* Registro y actualización de productos.
* Gestión de categorías y descripciones.
* Subida de imágenes de productos.
* Control de precios y descuentos.
  + 1. **Gestión de Ventas**
* Carrito de compras en tiempo real.
* Registro y procesamiento de pedidos.
* Métodos de pago (tarjetas, transferencias).
* Generación de facturas electrónicas.
  + 1. **Gestión de Inventario**
* Control de stock en tiempo real.
* Alertas por baja disponibilidad de productos.
* Reportes de movimientos de inventario.
* Configuración de inventario mínimo.
  + 1. **Gestión de Usuarios**
* Registro y autenticación de clientes.
* Gestión de roles (administradores, clientes).
* Historial de compras y preferencias.
  + 1. **Reportes y Estadísticas**
* Reportes de ventas (diarios, semanales, mensuales).
* Estadísticas de productos más vendidos.
* Gráficos interactivos para análisis de tendencias.

1. **ARQUITECTURA DEL SISTEMA**
   1. **STACK TECNOLÓGICO**

**Frontend:**

* Angular con TypeScript.
* Bootstrap para diseño responsivo.
* Servicios HTTP para consumo de APIs.

**Backend:**

* Node.js con Express para APIs REST.
* Autenticación con JSON Web Tokens (JWT).
* Swagger para documentación de API.

**Base de Datos**:

* MySQL para almacenamiento relacional.
* Configuración de índices para consultas optimizadas.
* Backups automatizados

**Infraestructura:**

* Docker para contenedores.
* CI/CD con GitHub Actions.
* Deploy en un proveedor cloud (AWS, GCP o similar).
  1. **REQUISITOS NO FUNCIONALES**

**Disponibilidad:**

* Uptime: 99.9%
* Tiempo máximo de inactividad: 8.76 horas/año

**Rendimiento:**

* Tiempo de respuesta < 2 segundos
* Throughput: 1000 requests/segundo
* Latencia máxima: 200ms

**Seguridad:**

* Autenticación JWT
* Encriptación de datos sensibles
* Auditoría de acciones
* Backups cifrados

**Escalabilidad:**

* Horizontal para componentes stateless
* Vertical para base de datos
* Auto-scaling basado en carga

1. **HISTORIAS DE USUARIO**
   1. **EPIC 1: Gestión de Productos**
      1. **HU-001: Registro de Productos**

**Como** administrador**.  
Quiero** registrar productos con detalles completos. **Para** mantener el inventario actualizado**.**

**Criterios de Aceptación:**

1. El formulario debe incluir:
   * Nombre del producto.
   * Categoría.
   * Precio.
   * Imagen.
   * Stock inicial.
2. Validar que el nombre del producto no esté duplicado.
3. Subida de imagen con formatos JPG, PNG.

**Estimación:** 5 puntos **Prioridad:** Alta

* + 1. **HU-002: Actualización de Productos**

**Como** administrador  
**Quiero** editar los datos de productos existentes  
**Para** reflejar cambios en precios o stock.

**Criterios de Aceptación**:

1. Permitir edición de todos los campos excepto ID.
2. Registrar historial de cambios.

**Estimación:** 3 puntos **Prioridad:** Media

* 1. **EPIC 2: Gestión de Ventas**
     1. **HU-003: Carrito de Compras**

**Como** cliente  
**Quiero** agregar productos al carrito  
**Para** realizar una compra fácilmente.

**Criterios de Aceptación:**

1. Permitir agregar y eliminar productos.
2. Calcular el total dinámicamente.

Estimación: 8 puntos  
Prioridad: Alta

* 1. **EPIC 3: Reportes y Estadísticas**
     1. **HU-0064: Generación de Reportes de Ventas**

**Como** administrador  
**Quiero** obtener reportes de ventas por período  
**Para** analizar el desempeño comercial.

Criterios de Aceptación:

1. Filtrar por fecha, categoría o producto.
2. Exportar a CSV o PDF.

Estimación: 5 puntos  
Prioridad: Media

1. **ANEXOS**
   1. **DIAGRAMAS**
      1. **Arquitectura del sistema**

**Arquitectura General:** Se utiliza una arquitectura de 3 capas (Three-Tier Architecture):

* **Capa de Presentación** (Frontend): Interfaz de usuario.
* **Capa de Lógica de Negocios** (Backend): Procesa la lógica del sistema.
* **Capa de Datos** (Base de Datos): Almacena y administra la información.

**Componentes Detallados**

1. **Capa de Presentación (Frontend):**

**Descripción:** Es la interfaz de usuario accesible desde navegadores web, Diseñada para ser responsiva y amigable.

**Tecnologías a usar:** Web: HTML5, CSS3, JavaScript, Frameworks como Angular.

**Funciones principales:**

* Gestión de productos (búsqueda, visualización, categorías).
* Carrito de compras.
* Procesos de compra.
* Panel administrativo para gestionar productos, categorías, usuarios, etc.

1. **Capa de Lógica de Negocios (Backend):**

**Descripción:** Se encarga de procesar la lógica del sistema y gestionar la comunicación entre la capa de presentación y la base de datos.

**Tecnologías a usar:** Lenguaje de programación: Java (Spring Boot), Node.js. Servidores basados en contenedores como Docker.

**Funciones principales:**

* Autenticación y autorización.
* Gestión de usuarios, productos, órdenes y pagos.
* Implementación de lógica de negocios (calcular precios, verificar stock, aplicar descuentos).
* Generación de reportes administrativos.

**Ejemplo de API REST:**

* POST /api/login: Autenticación de usuario.
* GET /api/products: Obtener la lista de productos.
* POST /api/orders: Registrar una nueva orden.

1. **Capa de Datos (Base de Datos):**

**Descripción:** Gestiona el almacenamiento persistente y seguro de la información.

**Tecnología a usar:** Base de datos relacional MySQL Workbench.

**Funciones principales:**

* Almacenamiento de información estructurada: usuarios, productos, órdenes, pagos.
* Optimización de consultas y seguridad de datos.
* Backup y recuperación ante fallos.
  + 1. **Modelo de datos**
       1. **Esquema de Base de Datos**

**1. Usuarios**

Tabla para gestionar los usuarios del sistema.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_usuario | INT | PRIMARY KEY, AUTO\_INCREMENT |
| nombre\_usuario | VARCHAR(50) | NOT NULL, UNIQUE |
| contraseña | VARCHAR(255) | NOT NULL |
| rol | ENUM('admin', 'cliente') | NOT NULL |
| correo | VARCHAR(100) | NOT NULL, UNIQUE |
| fecha\_registro | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP |

**2. Productos**

Tabla para gestionar los productos disponibles en la tienda.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_producto | INT | PRIMARY KEY, AUTO\_INCREMENT |
| nombre\_producto | VARCHAR(100) | NOT NULL |
| descripcion | TEXT |  |
| precio | DECIMAL(10, 2) | NOT NULL |
| stock | INT | NOT NULL |
| imagen | BLOB |  |
| categoria\_id | INT | FOREIGN KEY REFERENCES Categorias(id\_categoria) ON DELETE CASCADE |

**3. Categorías**

Tabla para clasificar los productos en categorías.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_categoria | INT | PRIMARY KEY, AUTO\_INCREMENT |
| nombre\_categoria | VARCHAR(50) | NOT NULL, UNIQUE |
| descripcion | TEXT |  |

**4. Clientes**

Tabla para registrar la información de los clientes.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_cliente | INT | PRIMARY KEY, AUTO\_INCREMENT |
| nombre | VARCHAR(100) | NOT NULL |
| apellido | VARCHAR(100) | NOT NULL |
| correo | VARCHAR(100) | NOT NULL, UNIQUE |
| telefono | VARCHAR(9) |  |
| direccion | VARCHAR(255) |  |
| fecha\_registro | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP |

**5. Órdenes**

Tabla para registrar las órdenes de compra realizadas por los clientes.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_orden | INT | PRIMARY KEY, AUTO\_INCREMENT |
| cliente\_id | INT | FOREIGN KEY REFERENCES Clientes(id\_cliente) ON DELETE CASCADE |
| fecha\_orden | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP |
| estado | ENUM('pendiente', 'completada', 'cancelada') | DEFAULT 'pendiente' |
| total | DECIMAL(10, 2) | |  | | --- | |  |  |  | | --- | | NOT NULL | |

**6. Detalles de Órdenes**

Tabla para registrar los productos incluidos en cada orden.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_detalle | INT | PRIMARY KEY, AUTO\_INCREMENT |
| orden\_id | INT | FOREIGN KEY REFERENCES Órdenes(id\_orden) ON DELETE CASCADE |
| producto\_id | INT | FOREIGN KEY REFERENCES Productos(id\_producto) ON DELETE CASCADE |
| cantidad | INT | NOT NULL |
| precio\_unitario | DECIMAL(10, 2) | NOT NULL |
| subtotal | DECIMAL(10, 2) | NOT NULL |

**7. Carrito de Compras**

Tabla para gestionar los productos que los clientes tienen en su carrito de compras.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_carrito | INT | PRIMARY KEY, AUTO\_INCREMENT |
| cliente\_id | INT | FOREIGN KEY REFERENCES Clientes(id\_cliente) ON DELETE CASCADE |
| producto\_id | INT | FOREIGN KEY REFERENCES Productos(id\_producto) ON DELETE CASCADE |
| cantidad | INT | NOT NULL |

**8. Historial de Pagos**

Tabla para registrar los pagos realizados por los clientes.

|  |  |  |
| --- | --- | --- |
| Campo | Tipo | Restricciones |
| id\_pago | INT | PRIMARY KEY, AUTO\_INCREMENT |
| orden\_id | INT | FOREIGN KEY REFERENCES Órdenes(id\_orden) ON DELETE CASCADE |
| fecha\_pago | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP |
| monto | DECIMAL(10, 2) | NOT NULL |
| metodo\_pago | ENUM('tarjeta', 'paypal', 'transferencia') | NOT NULL |

* + - 1. **script SQL**

**-- Crear la base de datos**

CREATE DATABASE TiendaVirtual;

**-- Usar la base de datos**

USE TiendaVirtual;

**-- Tabla Usuarios**

CREATE TABLE Usuarios (

id\_usuario INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_usuario VARCHAR(50) NOT NULL UNIQUE,

contraseña VARCHAR(255) NOT NULL,

rol ENUM('admin', 'cliente') NOT NULL,

correo VARCHAR(100) NOT NULL UNIQUE,

fecha\_registro TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**-- Tabla Categorías**

CREATE TABLE Categorias (

id\_categoria INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_categoria VARCHAR(50) NOT NULL UNIQUE,

descripcion TEXT

);

**-- Tabla Productos**

CREATE TABLE Productos (

id\_producto INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_producto VARCHAR(100) NOT NULL,

descripcion TEXT,

precio DECIMAL(10, 2) NOT NULL,

stock INT NOT NULL,

imagen BLOB,

categoria\_id INT NOT NULL,

FOREIGN KEY (categoria\_id) REFERENCES Categorias(id\_categoria) ON DELETE CASCADE

);

**-- Tabla Clientes**

CREATE TABLE Clientes (

id\_cliente INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

apellido VARCHAR(100) NOT NULL,

correo VARCHAR(100) NOT NULL UNIQUE,

telefono VARCHAR(15),

direccion VARCHAR(255),

fecha\_registro TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**-- Tabla Órdenes**

CREATE TABLE Ordenes (

id\_orden INT AUTO\_INCREMENT PRIMARY KEY,

cliente\_id INT NOT NULL,

fecha\_orden TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

estado ENUM('pendiente', 'completada', 'cancelada') DEFAULT 'pendiente',

total DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (cliente\_id) REFERENCES Clientes(id\_cliente) ON DELETE CASCADE

);

**-- Tabla Detalles de Órdenes**

CREATE TABLE DetallesOrden (

id\_detalle INT AUTO\_INCREMENT PRIMARY KEY,

orden\_id INT NOT NULL,

producto\_id INT NOT NULL,

cantidad INT NOT NULL,

precio\_unitario DECIMAL(10, 2) NOT NULL,

subtotal DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (orden\_id) REFERENCES Ordenes(id\_orden) ON DELETE CASCADE,

FOREIGN KEY (producto\_id) REFERENCES Productos(id\_producto) ON DELETE CASCADE

);

**-- Tabla Carrito de Compras**

CREATE TABLE CarritoCompras (

id\_carrito INT AUTO\_INCREMENT PRIMARY KEY,

cliente\_id INT NOT NULL,

producto\_id INT NOT NULL,

cantidad INT NOT NULL,

FOREIGN KEY (cliente\_id) REFERENCES Clientes(id\_cliente) ON DELETE CASCADE,

FOREIGN KEY (producto\_id) REFERENCES Productos(id\_producto) ON DELETE CASCADE

);

**-- Tabla Historial de Pagos**

CREATE TABLE HistorialPagos (

id\_pago INT AUTO\_INCREMENT PRIMARY KEY,

orden\_id INT NOT NULL,

fecha\_pago TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

monto DECIMAL(10, 2) NOT NULL,

metodo\_pago ENUM('tarjeta', 'paypal', 'transferencia') NOT NULL,

FOREIGN KEY (orden\_id) REFERENCES Ordenes(id\_orden) ON DELETE CASCADE

);

* 1. **MOCKUPS**
* Interfaces principales













**GITHUB**

* FRONT
* BACK
* DOCUMENTACION
* INFORME
* BD (SCRIPT)
* MANUAL DE INSTALACIÓN Y CONFIGURACIÓN

**APROBACIONES**

Product Owner Scrum Master

Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tech Lead Stakeholder Principal

Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_