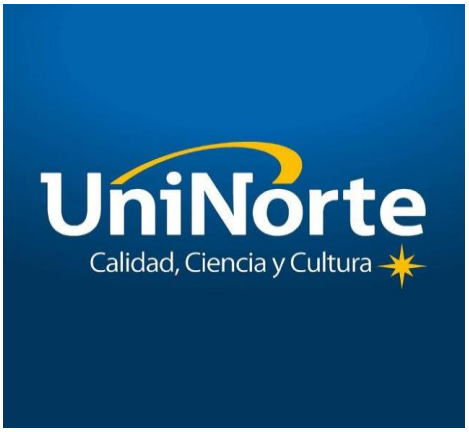

Documentación de API REST

Versión: 1.0

Fecha: 21/09/2025

Autor: José Ignacio Fernández Vera

Estado: Producción

📚 Documentación API Supermercado

Información General

Versión: 1.0
Framework: ASP.NET Core 8.0
Base de Datos: SQL Server
Autenticación: JWT (JSON Web Tokens)
Fecha: Septiembre 2025

📋 Swagger/OpenAPI

La API incluye documentación interactiva automática usando **Swagger/OpenAPI 3.0**:

Acceso a Swagger UI

- **URL de desarrollo:** `https://localhost:7000/swagger`
- **Descripción:** Interfaz web interactiva para probar todos los endpoints
- **Funcionalidades:**
  - Documentación automática de todos los endpoints
  - Esquemas de datos interactivos
  - Capacidad de ejecutar requests directamente desde la interfaz
  - Autenticación JWT integrada

Configuración OpenAPI

```csharp
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();
```

Usando Swagger para Autenticación

1. Haz clic en "**Authorize**" en Swagger UI
2. Ingresa: `Bearer {tu-token-jwt}`
3. Todos los requests posteriores incluirán automáticamente el token

---

🌐 Protocolo

La API utiliza el protocolo **HTTP/HTTPS** para todas las comunicaciones:

- **Protocolo:** HTTP 1.1 / HTTP 2.0
- **Puerto por defecto:** 7000 (HTTPS), 5000 (HTTP)
- **Formato de datos:** JSON
- **Codificación:** UTF-8
- **Content-Type:** `application/json`

URLs Base

```
Desarrollo: https://localhost:7000/api
Producción: https://tu-dominio.com/api
```

🏗️ Arquitectura REST

La API sigue los principios REST (Representational State Transfer):

Características REST Implementadas

Principio	Implementación
Cliente-Servidor	Separación clara entre frontend y backend
Sin Estado	Cada solicitud contiene toda la información necesaria
Cacheable	Respuestas GET pueden ser cacheadas
Interfaz Uniforme	URIs consistentes y métodos HTTP estándar
Sistema en Capas	Middleware, Controladores, Servicios, Datos

Verbos HTTP Utilizados

Verbo	Propósito	Ejemplo
`GET`	Obtener recursos	`GET /api/productos`
`POST`	Crear nuevos recursos	`POST /api/productos`
`PUT`	Actualizar recursos completos	`PUT /api/productos/1`
`DELETE`	Eliminar recursos	`DELETE /api/productos/1`

Estructura de URLs

```
/api/{recurso}          - Colección
/api/{recurso}/{id}     - Recurso específico
/api/{recurso}/{id}/{sub-recurso} - Sub-recursos
```

Ejemplos:

```
GET /api/productos            - Listar todos los productos
GET /api/productos/5          - Obtener producto con ID 5
GET /api/productos/categoria/2  - Productos de categoría 2
```

⚙️ Configuración Inicial

Requisitos del Sistema

- **.NET 8 SDK** o superior
- **SQL Server** (Express/LocalDB mínimo)
- **Visual Studio 2022** o VS Code
- **Postman** o herramienta similar (opcional)

Configuración de la Base de Datos

1. Cadena de Conexión (appsettings.json)

json

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=SupermercadoDB;Trusted_Connection=true;Multipl
  }
}
```

2. Aplicar Migraciones

bash

```
# Crear migración inicial
dotnet ef migrations add InicialCreacion

# Aplicar migraciones a la base de datos
dotnet ef database update
```

Configuración JWT (appsettings.json)

json

```
{
  "Jwt": {
    "Key": "SuperSecretKeyParaNuestroSupermercado2024!EstaClaveTieneMasDe32Caracteres",
    "Issuer": "SupermercadoAPI",
    "Audience": "SupermercadoUsers",
    "ExpireMinutes": 60
  }
}
```

Variables de Entorno

```bash
ASPNETCORE_ENVIRONMENT=Development
ASPNETCORE_URLS=https://localhost:7000;http://localhost:5000
```

🔐 Autenticación

Método de Autenticación: JWT (JSON Web Tokens)

Características del Token

- **Algoritmo:** HS256
- **Duración:** 60 minutos
- **Incluye:** ID usuario, nombre, email, rol
- **Header:** `Authorization: Bearer {token}`

Proceso de Autenticación

1. Obtener Token

```http
POST /api/auth/login
Content-Type: application/json

{
  "email": "admin@supermercado.com",
  "password": "Admin123!"
}
```

Respuesta Exitosa (200 OK):

```json
```

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "usuario": {
    "id": 1,
    "nombre": "Administrador",
    "email": "admin@supermercado.com",
    "rol": "Admin",
    "fechaCreacion": "2025-09-01T10:00:00",
    "activo": true
  },
  "expiration": "2025-09-01T11:00:00Z"
}
```

2. Usar Token en Requests

```http
GET /api/productos
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Niveles de Autorización

Nivel	Descripción	Endpoints
Público	Sin autenticación requerida	GET /api/productos, GET /api/categorias
Autenticado	Token JWT requerido	POST, PUT en la mayoría
Admin	Rol "Admin" requerido	DELETE endpoints, gestión usuarios

Registro de Nuevos Usuarios

```http
POST /api/auth/register
Content-Type: application/json

{
  "nombre": "Juan Pérez",
  "email": "juan@ejemplo.com",
  "password": "MiPassword123!",
  "rol": "Empleado"
}
```

📡 Endpoints

🔑 Autenticación (`/api/auth`)

POST /api/auth/login

Descripción: Iniciar sesión y obtener token JWT
Autenticación: No requerida

Request Body:

```
json

{
  "email": "string",
  "password": "string"
}
```

Responses:

- `200 OK` - Login exitoso, devuelve token
- `401 Unauthorized` - Credenciales inválidas
- `400 Bad Request` - Datos de entrada inválidos

POST /api/auth/register

Descripción: Registrar nuevo usuario
Autenticación: No requerida

Request Body:

```
json

{
  "nombre": "string",
  "email": "string",
  "password": "string",
  "rol": "Empleado"
}
```

👥 Usuarios (`/api/usuarios`)

GET /api/usuarios

Descripción: Listar todos los usuarios activos
Autenticación: Admin requerido

Responses: `200 OK`, `401 Unauthorized`, `403 Forbidden`

GET /api/usuarios/{id}

Descripción: Obtener usuario específico
Autenticación: Token requerido (propio perfil o Admin) **Responses:** `200 OK`, `404 Not Found`, `403 Forbidden`

POST /api/usuarios

Descripción: Crear nuevo usuario
Autenticación: Admin requerido
Request Body: UsuarioCreateDto

PUT /api/usuarios/{id}

Descripción: Actualizar usuario
Autenticación: Token requerido (propio perfil o Admin)
Request Body: UsuarioUpdateDto

DELETE /api/usuarios/{id}

Descripción: Eliminar usuario (eliminación lógica)
Autenticación: Admin requerido
Responses: `200 OK`, `400 Bad Request` (último admin)

PUT /api/usuarios/{id}/cambiar-password

Descripción: Cambiar contraseña de usuario
Autenticación: Token requerido (propio perfil o Admin)

📦 Categorías (`/api/categorias`)

GET /api/categorias

Descripción: Listar todas las categorías activas
Autenticación: No requerida
Responses: `200 OK`

Ejemplo de Respuesta:

```
json
```

```
[
  {
    "id": 1,
    "nombre": "Lácteos",
    "descripcion": "Productos lácteos y derivados",
    "fechaCreacion": "2025-09-01T10:00:00",
    "activa": true
  }
]
```


GET /api/categorias/{id}

Descripción: Obtener categoría específica
Autenticación: No requerida
Responses: `200 OK`, `404 Not Found`

POST /api/categorias

Descripción: Crear nueva categoría
Autenticación: Token requerido

Request Body:

```
json

{
  "nombre": "string (requerido, max 100 chars)",
  "descripcion": "string (opcional, max 300 chars)"
}
```

PUT /api/categorias/{id}

Descripción: Actualizar categoría
Autenticación: Token requerido

DELETE /api/categorias/{id}

Descripción: Eliminar categoría
Autenticación: Admin requerido
Validaciones: No debe tener productos asociados

🛒 Productos (`/api/productos`)

GET /api/productos

Descripción: Listar todos los productos activos con información de categoría y proveedor
Autenticación: No requerida

Ejemplo de Respuesta:

```json
[
  {
    "id": 1,
    "nombre": "Leche Descremada",
    "descripcion": "Leche descremada 1L",
    "precio": 2500.00,
    "stock": 50,
    "categoriaId": 1,
    "categoriaNombre": "Lácteos",
    "proveedorId": 1,
    "proveedorNombre": "Lácteos del Valle S.A.",
    "fechaVencimiento": "2025-12-15T00:00:00",
    "fechaCreacion": "2025-09-01T10:00:00",
    "activo": true
  }
]
```

GET /api/productos/{id}

Descripción: Obtener producto específico
Autenticación: No requerida

GET /api/productos/categoria/{categoriaId}

Descripción: Obtener productos por categoría
Autenticación: No requerida

POST /api/productos

Descripción: Crear nuevo producto
Autenticación: Token requerido

Request Body (ProductoCreateDto):

```json
```

```
{
  "nombre": "string (requerido, max 200 chars)",
  "descripcion": "string (opcional, max 500 chars)",
  "precio": "decimal (requerido, > 0)",
  "stock": "int (requerido, >= 0)",
  "categoriaId": "int (requerido)",
  "proveedorId": "int (requerido)",
  "fechaVencimiento": "datetime (opcional)"
}
```

PUT /api/productos/{id}

Descripción: Actualizar producto
Autenticación: Token requerido
Request Body: ProductoUpdateDto

DELETE /api/productos/{id}

Descripción: Eliminar producto (eliminación lógica)
Autenticación: Admin requerido

🏢 Proveedores (`/api/proveedores`)

GET /api/proveedores

Descripción: Listar todos los proveedores activos
Autenticación: No requerida

GET /api/proveedores/{id}

Descripción: Obtener proveedor específico
Autenticación: No requerida

GET /api/proveedores/{id}/productos

Descripción: Obtener productos de un proveedor específico
Autenticación: No requerida

POST /api/proveedores

Descripción: Crear nuevo proveedor
Autenticación: Token requerido

Request Body:

```
json
```

```
{
  "nombre": "string (requerido, max 150 chars)",
  "telefono": "string (opcional, max 20 chars)",
  "email": "string (opcional, formato email)",
  "direccion": "string (opcional, max 250 chars)"
}
```

PUT /api/proveedores/{id}

Descripción: Actualizar proveedor
Autenticación: Token requerido

DELETE /api/proveedores/{id}

Descripción: Eliminar proveedor
Autenticación: Admin requerido
Validaciones: No debe tener productos asociados

📋 Modelos de Datos

Usuario

json

```
{
  "id": "int (PK)",
  "nombre": "string (100 chars, requerido)",
  "email": "string (150 chars, requerido, único)",
  "password": "string (encriptado con BCrypt)",
  "rol": "string (Admin|Empleado, requerido)",
  "fechaCreacion": "datetime",
  "activo": "boolean"
}
```

Categoria

json

```
{
  "id": "int (PK)",
  "nombre": "string (100 chars, requerido, único)",
  "descripcion": "string (300 chars, opcional)",
  "fechaCreacion": "datetime",
  "activa": "boolean"
}
```

Proveedor

json

```
{
  "id": "int (PK)",
  "nombre": "string (150 chars, requerido)",
  "telefono": "string (20 chars, opcional)",
  "email": "string (150 chars, opcional, único)",
  "direccion": "string (250 chars, opcional)",
  "fechaCreacion": "datetime",
  "activo": "boolean"
}
```

Producto

json

```
{
  "id": "int (PK)",
  "nombre": "string (200 chars, requerido)",
  "descripcion": "string (500 chars, opcional)",
  "precio": "decimal(10,2) (requerido, > 0)",
  "stock": "int (requerido, >= 0)",
  "categoriaId": "int (FK a Categoria)",
  "proveedorId": "int (FK a Proveedor)",
  "fechaVencimiento": "datetime (opcional)",
  "fechaCreacion": "datetime",
  "activo": "boolean"
}
```

HistorialStock

json

```
{
  "id": "int (PK)",
  "productoId": "int (FK a Producto)",
  "usuarioId": "int (FK a Usuario)",
  "tipoMovimiento": "string (Entrada|Salida)",
  "cantidad": "int (> 0)",
  "fecha": "datetime",
  "motivo": "string (300 chars, opcional)",
  "precioUnitario": "decimal(10,2) (opcional)"
}
```

DTOs (Data Transfer Objects)

UsuarioCreateDto

json

```
{
  "nombre": "string",
  "email": "string",
  "password": "string (min 6 chars)",
  "rol": "string"
}
```

ProductoCreateDto

json

```
{
  "nombre": "string",
  "descripcion": "string",
  "precio": "decimal",
  "stock": "int",
  "categoriaId": "int",
  "proveedorId": "int",
  "fechaVencimiento": "datetime"
}
```

LoginDto

json

```
{
  "email": "string",
  "password": "string"
}
```

⚠ Códigos de Error

Códigos HTTP Estándar

Código	Significado	Cuándo se usa
200	OK	Operación exitosa (GET, PUT)
201	Created	Recurso creado exitosamente (POST)
400	Bad Request	Datos de entrada inválidos
401	Unauthorized	Token ausente, inválido o expirado
403	Forbidden	Sin permisos para la operación
404	Not Found	Recurso no encontrado
500	Internal Server Error	Error interno del servidor

Estructura de Respuestas de Error

Error Estándar

json

```
{
  "statusCode": 400,
  "message": "Descripción del error",
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Error de Validación

json

```
{
  "statusCode": 400,
  "message": "Datos de entrada inválidos",
  "errors": {
    "email": ["El email es obligatorio"],
    "password": ["La contraseña debe tener al menos 6 caracteres"]
  },
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Error de Autenticación

json

```
{
  "statusCode": 401,
  "message": "Email o contraseña incorrectos",
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Error de Autorización

json

```
{
  "statusCode": 403,
  "message": "No tienes permisos para realizar esta operación",
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Error de Recurso No Encontrado

json

```
{
  "statusCode": 404,
  "message": "Producto no encontrado",
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Errores Específicos del Dominio

Errores de Negocio

json

```
{
  "statusCode": 400,
  "message": "No se puede eliminar la categoría porque tiene productos asociados",
  "timestamp": "2025-09-01T10:00:00Z"
}
```

Errores de Integridad

```json
{
  "statusCode": 400,
  "message": "Ya existe un usuario con ese email",
  "timestamp": "2024-12-01T10:00:00Z"
}
```

🧪 Ejemplos de Uso

Flujo Completo: Crear un Producto

1. Login

```http
POST /api/auth/login
Content-Type: application/json

{
  "email": "admin@supermercado.com",
  "password": "Admin123!"
}
```

2. Crear Categoría

```http
POST /api/categorias
Authorization: Bearer {token}
Content-Type: application/json

{
  "nombre": "Bebidas",
  "descripcion": "Bebidas frías y calientes"
}
```

3. Crear Proveedor

```http
POST /api/proveedores
Authorization: Bearer {token}
Content-Type: application/json

{
  "nombre": "Distribuidora Refrescos S.A.",
  "telefono": "0981-123456",
  "email": "ventas@refrescos.com",
  "direccion": "Av. Principal 123"
}
```

4. Crear Producto

```http
POST /api/productos
Authorization: Bearer {token}
Content-Type: application/json

{
  "nombre": "Coca Cola 500ml",
  "descripcion": "Bebida gaseosa Coca Cola 500ml",
  "precio": 3500.00,
  "stock": 100,
  "categoriaId": 6,
  "proveedorId": 4,
  "fechaVencimiento": "2026-06-01T00:00:00"
}
```

Consultas Útiles

Listar productos por categoría:

```http
GET /api/productos/categoria/1
```

Buscar proveedor y sus productos:

```http
```

```
GET /api/proveedores/1
GET /api/proveedores/1/productos
```

Cambiar contraseña:

```http
PUT /api/usuarios/1/cambiar-password
Authorization: Bearer {token}
Content-Type: application/json

{
  "passwordActual": "Admin123!",
  "passwordNueva": "NuevaPassword123!"
}
```

🔧 Configuración Avanzada

CORS (Cross-Origin Resource Sharing)

La API está configurada para permitir todas las origenes en desarrollo:

```csharp
services.AddCors(options =>
{
    options.AddPolicy("AllowAll", policy =>
    {
        policy.AllowAnyOrigin()
              .AllowAnyMethod()
              .AllowAnyHeader();
    });
});
```

Logging

Los logs se configuran en `appsettings.json`:

```json
```

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning",
      "SupermercadoAPI": "Debug"
    }
  }
}
```

PRUEBAS CON POSTMAN