Documentación APIs e Integración

"E-commerce TechStore"

Título: Documentación de API para Gestión de E-commerce

Versión del Documento: 1.0
Fecha de Creación: 04/12/2024

Descripción General

Este documento proporciona una descripción detallada de los endpoints de la API REST diseñados para gestionar el sistema de e-commerce de artesanías con temática de Capibaras. Está dirigido a desarrolladores y administradores de sistemas que interactúan con el backend Spring Boot y el frontend React del proyecto.

Información General

- **URL Base del Backend:** http://localhost:8080/api
- **Base de Datos:** H2 (en memoria)
- **Consola H2:** http://localhost:8080/h2-console
- **Frontend:** http://localhost:5173
- **Autenticación:** JWT (JSON Web Token)
- **Framework Backend:** Spring Boot 3.2.5

Documentación de Endpoints

1. Endpoints de Autenticación

POST /api/auth/register

- **Descripción:** Registra un nuevo usuario en el sistema.
- **Método:** POST
- **Datos de Entrada:** {"firstName": "Juan", "lastName": "Pérez", "email": "juan@mail.com", "password": "password123"}
- **Respuesta:** 201 (Creado exitosamente), 400 (Datos inválidos), 409 (Email ya existe)
- **Autenticación:** No requerida
- **Observaciones:** Asigna automáticamente el rol USER. Retorna token JWT para autenticación inmediata.

POST /api/auth/login

- **Descripción:** Autentica un usuario existente.
- **Método:** POST
- **Datos de Entrada:** {"email": "admin@capibara.cl", "password": "admin123"}
- **Respuesta:** 200 (Login exitoso con token JWT), 401 (Credenciales inválidas)
- **Autenticación:** No requerida
- **Observaciones:** Retorna token JWT válido por 24 horas con roles del usuario.

2. Endpoints de Productos

GET /api/products

- **Descripción:** Lista todos los productos disponibles en el catálogo.
- **Método:** GET
- **Respuesta:** 200 (Lista de productos), 500 (Error del servidor)
- **Autenticación:** No requerida (acceso público)
- **Observaciones:** Retorna nombre, descripción, precio, stock, imageUrl y categoría.

GET /api/products/{id}

- **Descripción:** Obtiene los detalles completos de un producto específico.
- **Método:** GET
- **Parámetro:** id (Long) - ID del producto
- **Respuesta:** 200 (Producto encontrado), 404 (Producto no existe)
- **Autenticación:** No requerida

POST /api/products

- **Descripción:** Crea un nuevo producto en el catálogo.
- **Método:** POST
- **Datos de Entrada:** {"name": "Macetero Capibara", "description": "Macetero artesanal", "price": 15990.00, "stock": 20, "imageUrl": "...", "category": "DECORACION"}
- **Respuesta:** 201 (Producto creado), 400 (Datos inválidos), 403 (No autorizado)
- **Autenticación:** Sí (Solo ADMIN)
- **Observaciones:** Requiere token JWT con rol ROLE_ADMIN en header Authorization.

PUT /api/products/{id}

- **Descripción:** Actualiza la información de un producto existente.
- **Método:** PUT
- **Parámetro:** id (Long) - ID del producto a actualizar
- **Datos de Entrada:** {"name": "Macetero Capibara Grande", "price": 18990.00, "stock": 15}
- **Respuesta:** 200 (Actualizado), 400 (Datos inválidos), 404 (No encontrado), 403 (No autorizado)
- **Autenticación:** Sí (Solo ADMIN)

DELETE /api/products/{id}

- **Descripción:** Elimina un producto del catálogo.
- **Método:** DELETE
- **Parámetro:** id (Long) - ID del producto a eliminar
- **Respuesta:** 204 (Eliminado exitosamente), 404 (No encontrado), 403 (No autorizado)
- **Autenticación:** Sí (Solo ADMIN)

3. Endpoints de Órdenes

POST /api/orders

- **Descripción:** Crea una nueva orden de compra con los productos del carrito.
- **Método:** POST
- **Datos de Entrada:** {"items": [{"productId": 1, "quantity": 2}], "shippingAddress": "Av. Providencia 123", "notes": "Sin cebolla"}
- **Respuesta:** 201 (Orden creada), 400 (Stock insuficiente o datos inválidos), 401 (No autenticado)
- **Autenticación:** Sí (USER o ADMIN)
- **Observaciones:** Calcula el total automáticamente, verifica stock y actualiza inventario. Estado inicial: PENDING.

GET /api/orders/my-orders

- **Descripción:** Obtiene todas las órdenes del usuario autenticado.
- **Método:** GET
- **Respuesta:** 200 (Lista de órdenes del usuario), 401 (No autenticado)
- **Autenticación:** Sí (USER o ADMIN)
- **Observaciones:** Retorna órdenes con items, total, estado y fechas de creación/actualización.

GET /api/orders/{id}

- **Descripción:** Obtiene los detalles completos de una orden específica.
- **Método:** GET
- **Parámetro:** id (Long) - ID de la orden
- **Respuesta:** 200 (Orden encontrada), 404 (Orden no existe), 403 (No autorizado)
- **Autenticación:** Sí (USER o ADMIN)
- **Observaciones:** Usuarios solo pueden ver sus propias órdenes. ADMIN puede ver todas.

GET /api/orders

- **Descripción:** Lista todas las órdenes del sistema.
- **Método:** GET
- **Respuesta:** 200 (Lista completa de órdenes), 403 (No autorizado)
- **Autenticación:** Sí (Solo ADMIN)
- **Observaciones:** Panel administrativo para gestión de órdenes.

PATCH /api/orders/{id}/status

- **Descripción:** Actualiza el estado de una orden.
- **Método:** PATCH
- **Parámetro:** status (String) - Valores: "PENDING", "CONFIRMED", "SHIPPED", "DELIVERED", "CANCELLED"
- **Respuesta:** 200 (Estado actualizado), 400 (Estado inválido), 404 (Orden no existe)
- **Autenticación:** Sí (Solo ADMIN)

PATCH /api/orders/{id}/cancel

- **Descripción:** Cancela una orden existente.
- **Método:** PATCH
- **Parámetro:** id (Long) - ID de la orden a cancelar
- **Respuesta:** 200 (Orden cancelada), 400 (No se puede cancelar), 404 (No existe)
- **Autenticación:** Sí (USER o ADMIN)
- **Observaciones:** Devuelve stock al inventario. Solo se pueden cancelar órdenes en estado PENDING.

Integración con el Frontend

Verificación de Entrega

El frontend React interactúa con la API REST mediante servicios dedicados que encapsulan las llamadas HTTP utilizando Axios. A continuación se documenta el flujo de integración y las capturas de pantalla que verifican las respuestas exitosas de cada endpoint.

Servicios del Frontend

- **authService.ts:** Gestiona login, registro y almacenamiento de tokens JWT en localStorage.
- **productService.ts:** Obtiene listado y detalles de productos desde /api/products.
- **orderService.ts:** Crea órdenes y obtiene historial de compras desde /api/orders.

Flujo de Autenticación

1. Usuario ingresa email y contraseña en el formulario de login.

2. Frontend envía POST /api/auth/login con las credenciales.

3. Backend valida credenciales y retorna token JWT con roles del usuario.

4. Frontend almacena token en localStorage y agrega header Authorization: Bearer <token> a todas las peticiones subsecuentes.

5. Usuario es redirigido al catálogo de productos con sesión activa.

Flujo de Creación de Orden

1. Usuario agrega productos al carrito desde el catálogo.

2. Procede al checkout y completa datos de envío y pago.

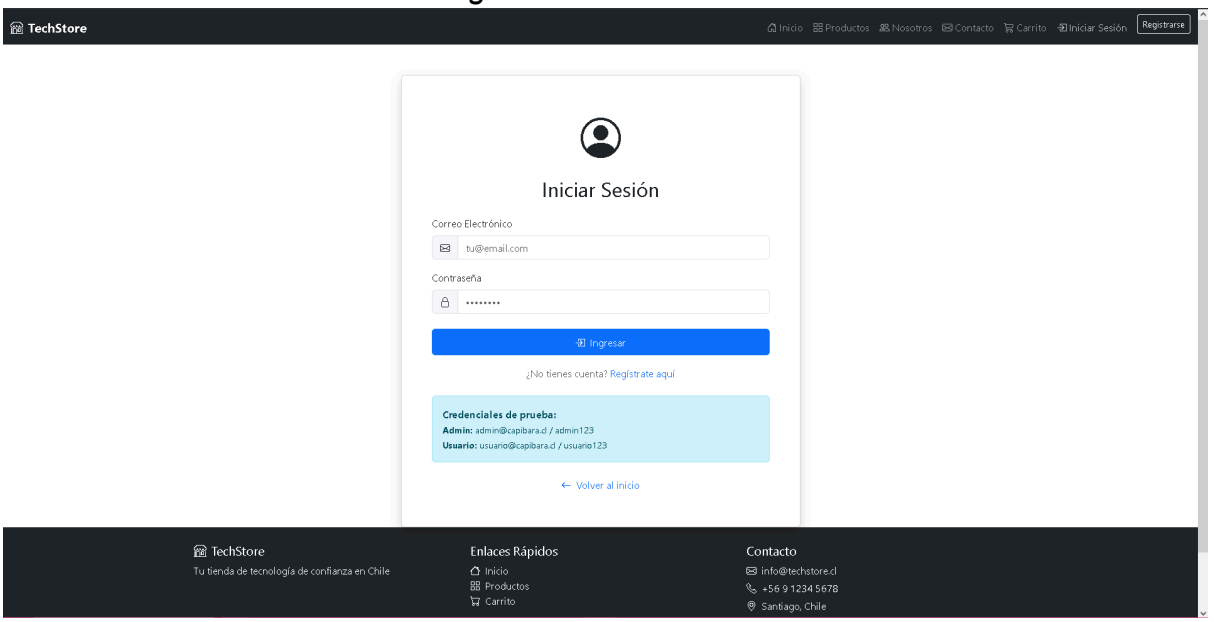
3. Frontend envía POST /api/orders con items, dirección y notas.

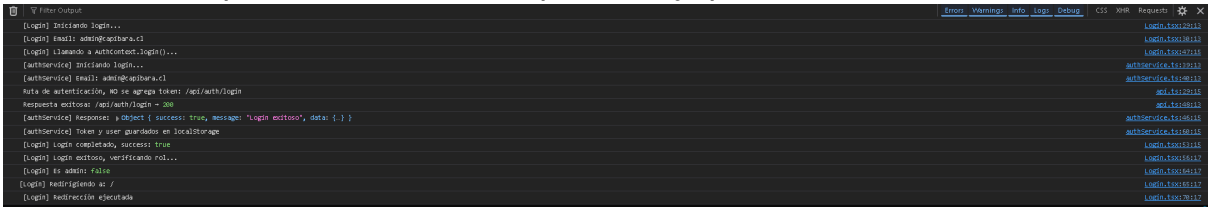
4. Backend verifica stock, calcula total, actualiza inventario y crea la orden.

5. Frontend muestra confirmación con número de orden y limpia el carrito.

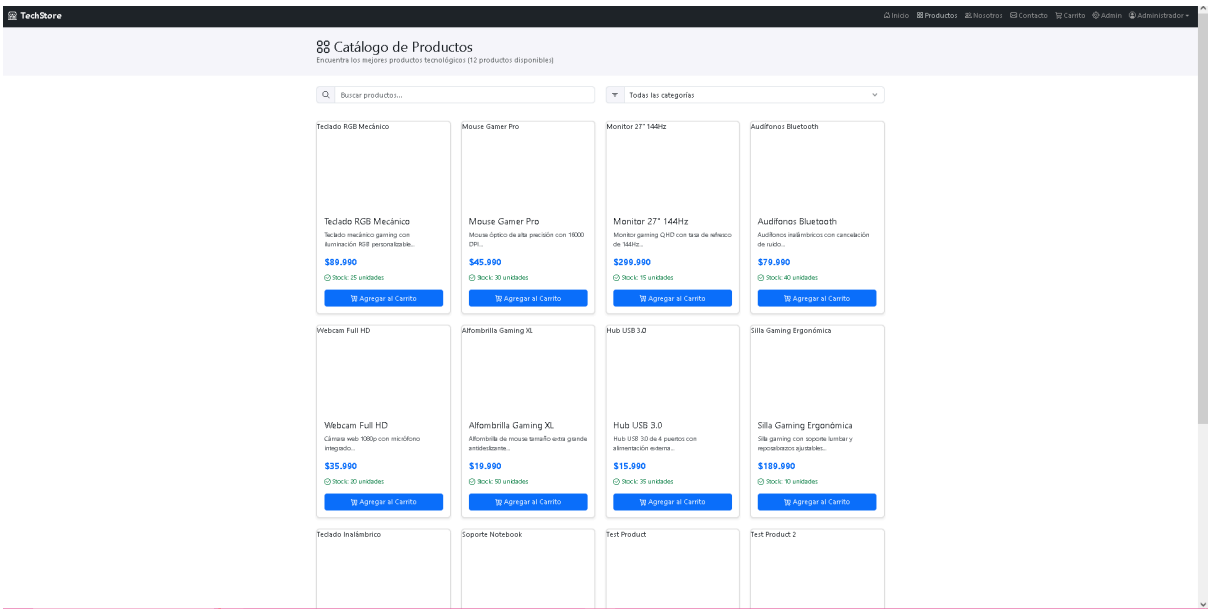
6. Orden persiste en base de datos H2 con estado PENDING.

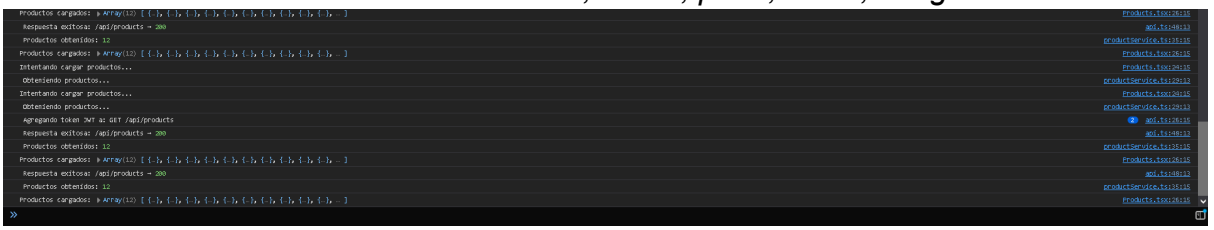
Capturas de Pantalla

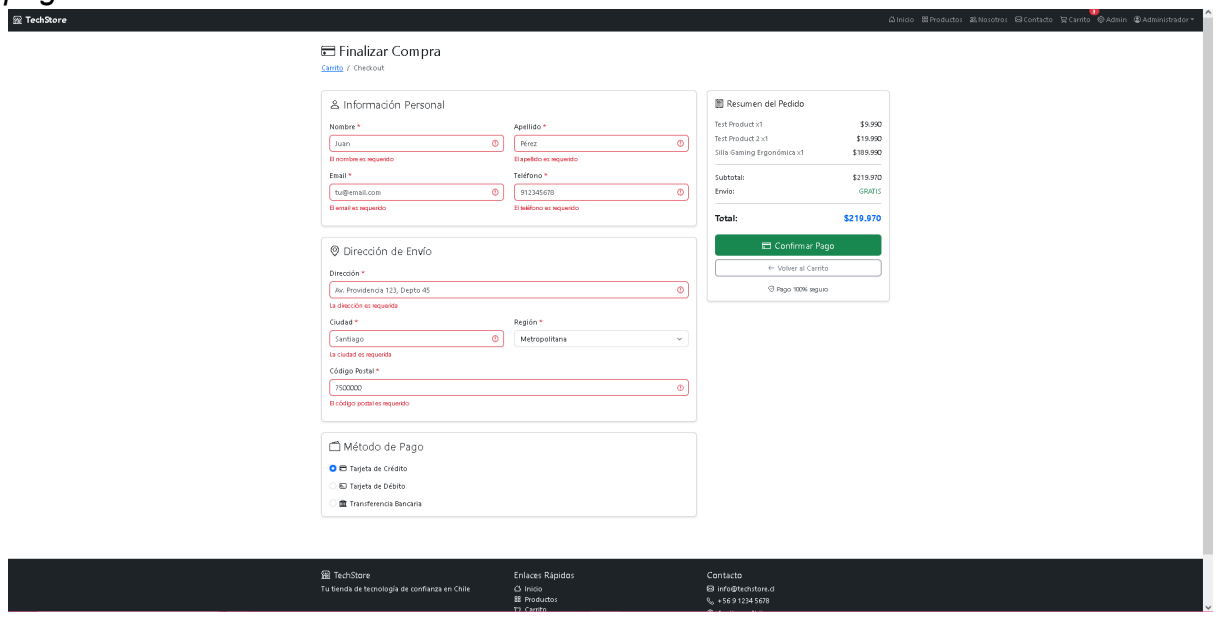
- **Figura 1:** Pantalla de Login - Formulario de autenticación con campos email y password.
- ***Nota:*** *Muestra el formulario de login antes de autenticarse.*
- 

- **Figura 2:** Console del navegador mostrando respuesta exitosa POST /api/auth/login con token JWT.
- ***Nota:*** *Verificar que el token JWT está presente y que el status es 200.*
- 

- **Figura 3:** Catálogo de productos - Grid con 12 productos mostrando imágenes, nombres, precios y botón Agregar al Carrito.
- ***Nota:*** *Muestra la respuesta exitosa de GET /api/products en la interfaz.*

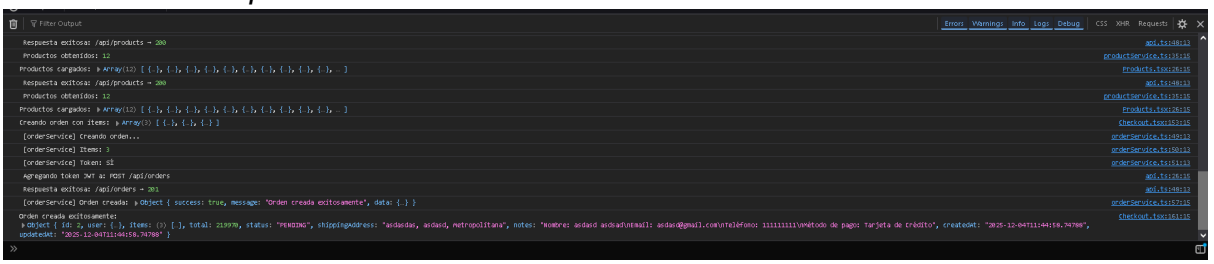
- 

- **Figura 4:** Console del navegador con log 'Productos obtenidos: 12' y array de productos.
- ***Nota:*** *Verificar estructura de datos con id, name, price, stock, imageUrl.*
- 

- **Figura 5:** Página de Checkout con formulario completo de envío y resumen de orden.
- ***Nota:*** *Muestra los campos nombre, email, teléfono, dirección y método de pago.*
- 

- **Figura 6:** Pantalla de confirmación de compra exitosa con número de orden ORD-1-XXXX.

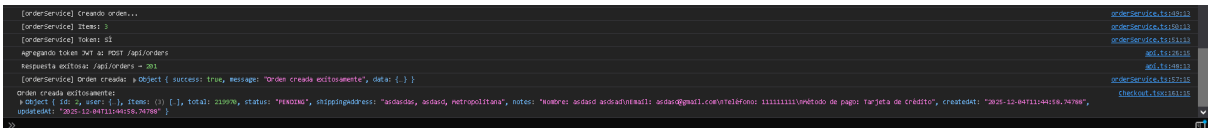
- ***Nota:*** *Verificar que muestra total pagado, email y dirección de envío.*
-

- **Figura 7:** Console del navegador mostrando POST /api/orders 201 Created con objeto de orden.
- ***Nota:*** *Confirmar que la orden fue creada exitosamente en el backend.*
- 

- **Figura 8:** H2 Console mostrando query SELECT * FROM ORDERS con la orden recién creada.
- ***Nota:*** *Verificar que la orden persiste con ID, USER_ID, TOTAL, STATUS=PENDING, fechas.*
- select * from orders;

ID	CREATED_AT	NOTES	SHIPPING_ADDRESS	STATUS	TOTAL	UPDATED_AT	USER_ID
1	2025-12-04 11:18:52.057002	Nombre: asdasdasd asdasd Email: asdasd@gmail.com Teléfono: 111111111 Método de pago: Tarjeta de Crédito	asdasd, ASDASD, Metropolitana	PENDING	29980.00	2025-12-04 11:18:52.057002	1
2	2025-12-04 11:44:58.74788	Nombre: asdasd asdsad Email: asdasd@gmail.com Teléfono: 111111111 Método de pago: Tarjeta de Crédito	asdasdas, asdasd, Metropolitana	PENDING	219970.00	2025-12-04 11:44:58.74788	1

(2 rows, 0 ms)

- **Figura 9:** Network tab del navegador mostrando header Authorization: Bearer <token> en request POST /api/orders.
- ***Nota:*** *Confirmar que el token JWT se envía correctamente en cada petición autenticada.*
- 

Consideraciones de Seguridad

Autenticación JWT

- Los tokens JWT tienen una validez de 24 horas desde su emisión.
- Los tokens incluyen roles del usuario (ROLE_USER, ROLE_ADMIN) en los claims.
- Se utiliza firma HMAC con clave secreta configurada en application.properties.

Autorización por Roles

- Spring Security valida roles mediante anotaciones @PreAuthorize en los controladores.
- Endpoints administrativos (POST/PUT/DELETE productos, GET todas órdenes) requieren ROLE_ADMIN.
- Los usuarios solo pueden acceder a sus propias órdenes mediante validación en el servicio.

Configuración CORS

- Se permite acceso desde http://localhost:5173 (frontend React) mediante SecurityConfig.
- Se permiten métodos HTTP: GET, POST, PUT, PATCH, DELETE, OPTIONS.
- Se exponen headers Authorization y Content-Type para el frontend.

Usuarios de Prueba

Usuario Administrador:

- **Email:** admin@capibara.cl
- **Password:** admin123
- **Roles:** ROLE_ADMIN, ROLE_USER

Usuario Regular:

- **Email:** usuario@capibara.cl
- **Password:** usuario123
- **Roles:** ROLE_USER

--- Fin del Documento ---