

Especificación de requisitos de software

Proyecto: Lotuz

Revisión:

Fecha: Lunes, 08 Septiembre 2025

Equipo: Gabriela Saldivar, Corina Roa

Docente: Viviana Poblete

Sección: DSY1104-007D

Contenido

1. Introducción	6
1.1. Propósito	6
1.2. Ámbito del Sistema	6
1.3. Definiciones, Acrónimos y Abreviaturas	7
1.4. Referencias	7
1.5. Visión General del Documento	7
2. Descripción General	8
2.1. Perspectiva del Producto	8
2.2. Funciones del Producto	8
2.3. Características de los Usuarios	8
2.4. Restricciones	9
2.5. Suposiciones y Dependencias	9
2.6. Requisitos Futuros	9
3. Requisitos Específicos	9
3.1 Requisitos comunes de los interfaces	9
3.1.1 Interfaces de usuario	9
3.1.2 Interfaces de hardware	10
3.1.3 Interfaces de software	10
3.1.4 Interfaces de comunicación	10
3.2 Requisitos funcionales	10
A. Tienda (público)	10
B. Admin (privado)	11
3.3 Requisitos no funcionales	12
3.3.1 Requisitos de rendimiento	12
3.3.2 Seguridad (FrontEnd)	12
3.3.3 Fiabilidad	13
3.3.4 Disponibilidad	13
3.3.5 Mantenibilidad	13
3.3.6 Portabilidad y accesibilidad	13
3.4 Modelo de datos (localStorage)	13
3.5 Reglas de negocio	14
3.6 Casos de uso (resumen)	14
3.7 Criterios de aceptación (muestra)	15
3.8 Plan de pruebas (breve)	15
3.9 Trazabilidad (ejemplo)	16
4. Entregables y repositorio	16
5. Riesgos y mitigaciones	16
6. Anexos	17
6.1 Estructura de carpetas (actual)	17
6.2 Diagrama de flujo de navegación (guía rápida)	17

Ficha del documento

Fecha	Revisión	Autor	Modificación
08/09/2025	1.0	Gabriela Saldivar, Corina Roa	Primera versión alineada a la implementación actual (cliente + admin)

Documento validado por las partes en fecha:

Por el cliente

Por la empresa suministradora



Sr.

1. Introducción

A nuestro equipo se le ha encomendado el desarrollo de una **tienda online básica para "Lotuz"**, una pyme chilena dedicada a la venta de periféricos gamer de alta calidad. Actualmente, Lotuz opera principalmente por encargo y busca expandir su alcance, llegar a más clientes y optimizar la gestión de sus productos. El sistema actual carece de una plataforma web pública y de una herramienta administrativa eficiente, lo que limita su crecimiento y la visibilidad de sus productos.

Este proyecto tiene como objetivo sentar las bases de un sistema robusto, empezando por una solución tecnológica que permita a Lotuz establecer su presencia online con una tienda funcional y gestionar sus productos de manera eficaz.

1.1. Propósito

Este documento define los requisitos del **Front-End** de la tienda online de **Lotuz**, pyme chilena dedicada a periféricos gamer. La ERS sirve como base para desarrollo, pruebas, evaluación y trazabilidad de la **Evaluación 1**. La solución contempla **módulo de Tienda (público)** y **módulo Administrativo (privado)** con persistencia en **localStorage**.

1.2. Ámbito del Sistema

Incluido en esta fase (E1):

- **Tienda:** catálogo, detalle, **carrito** persistente, **pago simulado** (crea orden), login/registro básico, **perfil con historial**, contacto/nosotros.
- **Admin:** **login simulado**, dashboard (KPI), **CRUD de productos**, listado y **cambio de estado** de órdenes.
- Datos persistidos en **localStorage**. Recursos estáticos `./img/productos/*`.

Fuera de alcance (E1):

- Pasarela de pago real, inventario/stock real, autenticación y autorización seguras en backend.
- **Mantenedor de usuarios** (creación/edición) y gestión de roles reales (queda como futuro).
- Logística y reportes financieros avanzados.

1.3. Definiciones, Acrónimos y Abreviaturas

- ERS, UI/UX, CRUD, KPI, localStorage, Cliente (comprador), Admin (gestor).
- HTML5, CSS3, JS (ES6+). PEL: Plataforma E-commerce Lotuz.

1.4. Referencias

- Instrucciones y **Rúbrica** de la Experiencia 1.
- Plantilla ERS (IEEE-830).
- Repositorio del proyecto (README, capturas).
-

1.5. Visión General del Documento

Se detallan contexto, alcance, restricciones y **requisitos** (funcionales/no funcionales, reglas de negocio, datos), además de **casos de uso**, **criterios de aceptación**, **plan de pruebas** y **trazabilidad**.

Diagrama de flujo de las paginas de Usuario (cliente) y Administrador

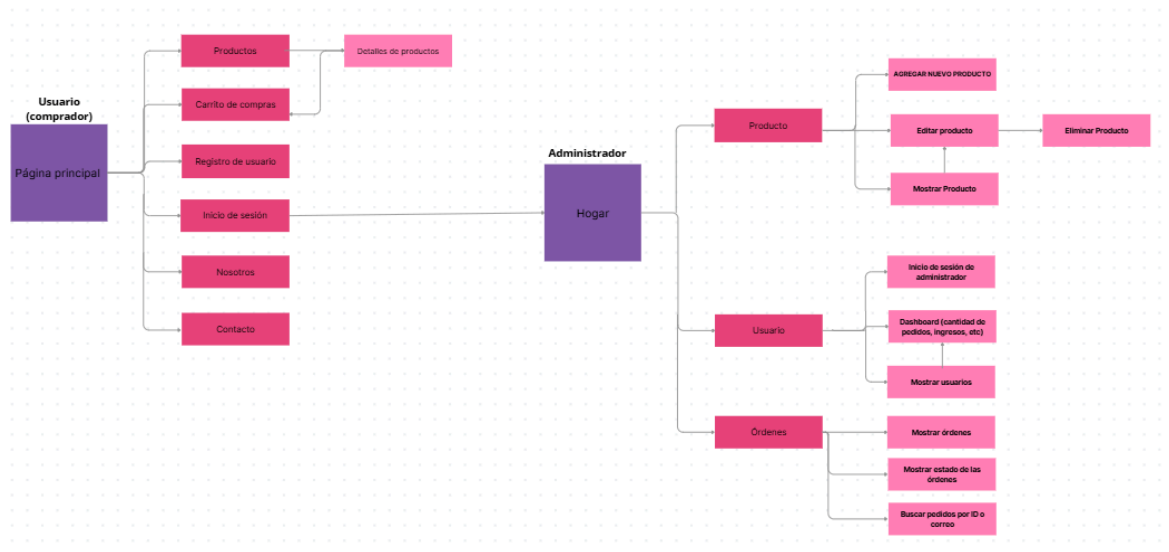
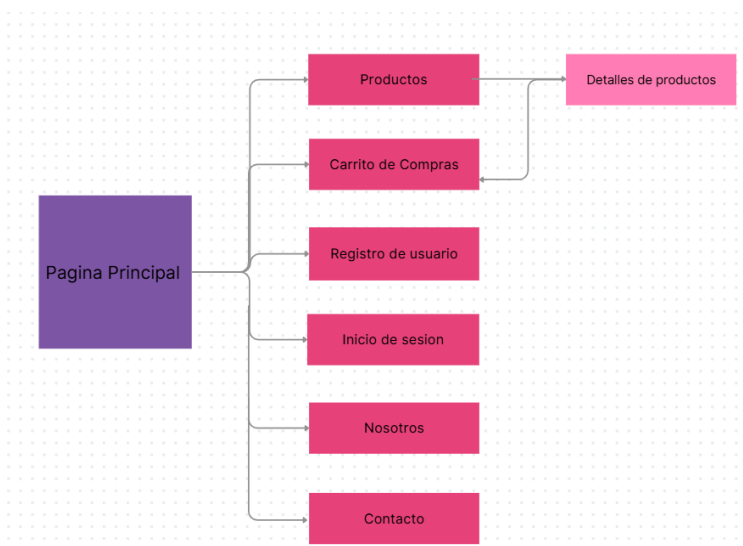


Diagrama de flujo solo de la tienda (vista de cliente)



2. Descripción General

2.1. Perspectiva del Producto

Aplicación **Front-End estática** que corre en navegador y persiste estado en **localStorage**. Dos módulos:

- **Tienda (cliente):** exploración/compra simulada.
- **Admin (privado):** mantenimiento de catálogo y monitoreo de órdenes.

2.2. Funciones del Producto

- **Catálogo:** buscar, filtrar, ordenar; ver detalle; agregar al carrito.
- **Carrito/Checkout simulado:** resumen, total, creación de **orden** y vaciado del carrito.
- **Cuenta y perfil:** login/registro básico; **historial** por correo.
- **Admin:** login, **CRUD productos, órdenes** (listar, ver detalle, cambiar estado), **dashboard** (KPI).

2.3. Características de los Usuarios

- **Cliente:** navega, añade al carrito, compra simulada, consulta historial.
- **Administrador:** mantiene catálogo, revisa órdenes, actualiza estados y visualiza KPI.
(Rol “Vendedor” queda como futuro; ver 2.6).

2.4. Restricciones

- Sin backend/BD; **sin** APIs externas.
- Rutas de imágenes normalizadas a ./img/productos/....
- Navegadores modernos (Chrome/Edge/Safari/Firefox).
- **HTML semántico, CSS externo, JS modular.**

2.5. Suposiciones y Dependencias

- Estructura de carpetas: /cliente, /admin, /css, /js, /img.
- Servidor estático simple o “Live Server”.
- JavaScript habilitado en el navegador.

2.6. Requisitos Futuros

- Backend y BD real; autenticación y autorización seguras; **mantenedor de usuarios y roles**; pasarela de pago; logística; reportes; recomendaciones; app móvil.

3. Requisitos Específicos

3.1 Requisitos comunes de los interfaces

3.1.1 Interfaces de usuario

- **Tienda (/cliente):**
inicio.html, productos.html, detalleProducto.html, carrito.html, pago.html, login.html, perfil.html, nosotros.html.
- **Admin (/admin):**
admin-login.html, admin.html (dashboard), admin-productos.html (CRUD), admin-ordenes.html.
- Estándares: <header>/<nav>/<main>/<footer>, tipografía consistente, foco visible, mensajes de error claros, estados vacíos, **responsive**.

3.1.2 Interfaces de hardware

PCs y dispositivos móviles con navegador moderno y JS habilitado.

3.1.3 Interfaces de software

HTML5, CSS3, JS ES6+. Uso de **localStorage**. Carga de assets estáticos.

3.1.4 Interfaces de comunicación

HTTP/HTTPS para servir archivos estáticos. Sin APIs de terceros.

3.2 Requisitos funcionales

A. Tienda (público)

RF.TIENDA.001 – Home

- Nav con logo y enlaces; hero/bienvenida; destacados; footer.

RF.TIENDA.002 – Productos

- Listado dinámico desde lotuz:catalog.
- **Buscar, filtrar por categoría y ordenar** (relevancia, precio asc/desc).

- Cada tarjeta: imagen, nombre, precio, botón **Añadir**.
- **Agrupación por categoría** cuando no hay filtros/búsqueda.

RF.TIENDA.003 – Detalle de producto

- Imagen, nombre, descripción, precio; **Añadir al carrito**; navegación de retorno.

RF.TIENDA.004 – Carrito (JS)

- Añadir/editar cantidades/eliminar; subtotal/total; **contador** en header.
- **Persistencia en localStorage** (cart).
- Estado vacío con mensaje claro.

RF.TIENDA.005 – Pago simulado

- Resumen y total; botón “Pagar ahora”.
- Al pagar: validar carrito no vacío y correo; **crear orden** en lotuz:orders con id, items, total, email, createdAt, status="Pagado"; **vaciar** carrito.

RF.TIENDA.006 – Login/Registro

- Formulario básico (correo y contraseña) con validaciones mínimas de formato y longitud.

RF.TIENDA.007 – Perfil

- Mostrar **historial** de órdenes filtrado por correo de sesión.
- Actualización básica de datos visibles (p.ej., correo para asociar órdenes futuras).

RF.TIENDA.008 – Contacto/Nosotros

- Formulario de contacto con validaciones; página informativa.

B. Admin (privado)

RF.ADMIN.001 – Login

- Formulario de acceso; persistir sesión admin en lotuz:admin; **guard** que redirige a admin-login.html si no hay sesión.

RF.ADMIN.002 – Dashboard

- KPI: **#Órdenes, Ingresos (CLP), Ticket promedio**; actividad reciente (últimas 5 órdenes).

RF.ADMIN.003 – Productos (CRUD)

- **Listar** productos de lotuz:catalog; **buscar y filtrar por categoría**.
- **Crear/Editar/Eliminar** con validaciones:
 - id (único, ≥ 3), nombre (≤ 100), categoría \in {mousepad, mouse, audifonos, teclado}, precio (> 0 , entero CLP), foto (URL relativa normalizada), descripcion (≤ 500).
- Diálogo/modal para alta/edición; feedback de éxito/errores.

RF.ADMIN.004 – Órdenes

- **Listar** órdenes de lotuz:orders con fecha, cliente, ítems, total y **estado** editable: {Pagado, Preparando, Despachado, Entregado, Anulado}.
- **Detalle** de orden y **eliminación** (confirmada).
- Cambios de estado **persisten**.

Nota: El **Mantenedor de Usuarios** y roles **Vendedor/Admin** a nivel de permisos finos quedan **fuera de E1**. Se mantienen referencias para futuras versiones (2.6).

3.3 Requisitos no funcionales

Estos requisitos se enfocan en las cualidades que el sistema debe poseer.

3.3.1 Requisitos de rendimiento

- Carga inicial por vista ≤ 2 s en entorno de laboratorio con assets locales.
- Interacciones y validaciones con **feedback inmediato**.

3.3.2 Seguridad (FrontEnd)

- Validación de entradas en cliente.
- Ocultar opciones admin a no autenticados; **guard** en rutas /admin.
- No se almacenan datos sensibles; sin cifrado en E1

3.3.3 Fiabilidad

- Validaciones robustas; manejo de entradas inesperadas.
- Consistencia del carrito/órdenes en localStorage.

3.3.4 Disponibilidad

- Operativa 24/7 como sitio estático; **autónoma sin backend**.

3.3.5 Mantenibilidad

- Código **modular**: catalog.js, productos.js, admin_productos.js, admin_ordenes.js, admin_guard.js, session.js, etc.
- **CSS externo** (estilos.css) y HTML semántico

3.3.6 Portabilidad y accesibilidad

- Compatibilidad Chrome/Edge/Safari/Firefox.
- **Responsive** móvil/escritorio.
- Imágenes con alt, labels asociados, contraste y foco visibles.

3.4 Modelo de datos (localStorage)

- **lotuz:catalog**

- { "id": "slug", "nombre": "str", "categoria": "mousepad|mouse|audifonos|teclado", "precio": 12345, "foto": "./img/productos/..png", "descripcion": "str" }}
- **cart**
- { "id": "slug", "nombre": "str", "precio": 12345, "foto": "./img/productos/..png", "cantidad": 1 }
- **lotuz:orders**
- { "id": "ORD-YYYYMMDD-HHMMSS-XXXX", "email": "cliente@dominio.com", "items": [{ "id": "slug", "nombre": "str", "precio": 12345, "cantidad": 1 }, "total": 12345, "status": "Pagado|Preparando|Despachado|Entregado|Anulado", "createdAt": "ISO-8601" }
- **lotuz:admin**
- { "email": "admin@lotuz.cl", "logged": true, "name": "Admin" }

3.5 Reglas de negocio

- **RB-01** id de producto **único** y ≥ 3 caracteres.
- **RB-02** categoria válida (mousepad/mouse/audifonos/teclado).
- **RB-03** precio > 0 , entero en CLP.
- **RB-04** Imágenes **normalizadas** a ./img/productos/....
- **RB-05** Para **pagar**, carrito no vacío y correo válido.
- **RB-06** Estados de orden válidos y persistentes.
- **RB-07** Guard de admin: si no hay lotuz:admin.logged, redirigir a admin-login.html.

3.6 Casos de uso (resumen)

- **UC-01** Buscar/filtrar/ordenar productos.
- **UC-02** Ver detalle y **agregar al carrito**.
- **UC-03 Pagar** (simulado) y generar orden.
- **UC-04** Ver **historial** en perfil (por correo).
- **UC-05 CRUD** de producto (admin).
- **UC-06** Cambiar **estado** de orden (admin).

3.7 Criterios de aceptación (muestra)

- **CA-CAT-01**: “logi” filtra productos por nombre/descr/ID conteniendo “logi”.
- **CA-CARR-01**: Contador del carrito = suma de cantidad.
- **CA-CHECK-01**: “Pagar ahora” crea orden con id, items, total, email, status="Pagado" y vacía el carrito.
- **CA-ADMIN-CRUD-01**: Crear producto válido lo muestra en admin y en tienda.
- **CA-ORD-01**: Cambiar estado de orden persiste tras recargar.
- **CA-IMG-01**: Ninguna imagen de catálogo devuelve 404 (rutas normalizadas).

3.8 Plan de pruebas (breve)

Smoke – Tienda

1. Abrir productos.html → grilla visible.
2. Buscar “razer” → aparecen productos esperados.
3. Agregar 2 ítems → contador y total correctos.
4. pago.html → **Pagar** → orden creada y carrito vacío.

5. perfil.html (logueado) → historial muestra la orden.

Smoke – Admin

1. Login admin → admin.html muestra KPI.
2. admin-productos.html → crear producto válido → aparece en tabla y tienda.
3. admin-ordenes.html → cambiar estado → persiste.
4. Guard: acceder a /admin sin sesión → redirige a admin-login.html.

3.9 Trazabilidad (ejemplo)

Requisito	Vista/Script
RF.TIENDA.002/003	productos.html, detalleProducto.html, catalog.js, productos.js
RF.TIENDA.004/005	carrito.html, pago.html, carrito.js, pago.js
RF.TIENDA.006/007	login.html, perfil.html, session.js
RF.ADMIN.001/002	admin-login.html, admin.html, admin_login.js, admin_guard.js
RF.ADMIN.003	admin-productos.html, admin_productos.js, catalog.js
RF.ADMIN.004	admin-ordenes.html, admin_ordenes.js

4. Entregables y repositorio

- Código fuente organizado (/cliente, /admin, /css, /js, /img).
- **README.md** con ejecución, estructura, capturas.
- **ERS (este documento) en PDF** en /docs.
- **≥6 plantillas HTML** (cliente + admin) exigidas por la rúbrica.
- Presentación ≤ 15 minutos (equipo, cliente, requisitos y herramientas).

5. Riesgos y mitigaciones

- **Borrado de localStorage** → pérdida de datos → *seed automático del catálogo si vacío.*
- **Rutas relativas** (cliente/admin) → 404 de imágenes → *normalización en catalog.js.*
- **Sin backend** → límites de realismo → *declarado en alcance; plan de evolución.*
- **Accesibilidad parcial** → *mejoras en labels/aria y contraste en siguientes iteraciones.*

6. Anexos

6.1 Estructura de carpetas (actual)

/cliente

inicio.html, productos.html, detalleProducto.html, carrito.html,
pago.html, login.html, perfil.html, nosotros.html

/admin

admin-login.html, admin.html, admin-productos.html, admin-ordenes.html

/css

estilos.css

/js

catalog.js, productos.js, carrito.js, pago.js, login.js, session.js,

admin_login.js, admin_guard.js, admin_nav.js, admin_productos.js, admin_ordenes.js

/img/productos/*.png (rutas normalizadas: ./img/productos/...)

6.2 Diagrama de flujo de navegación (guía rápida)

- **TIENDA:** inicio → productos → (detalle) → carrito → pago; acceso a login y perfil; contacto/nosotros.
- **ADMIN:** admin-login → admin (dashboard) → admin-productos (CRUD) y admin-ordenes (estados/detalle).