|  |
| --- |
| DUOC UC - Escuela de informatica y telecomunicaciones |
| Especificación de Requisitos de Software |
| *Proyecto: Beyblade Store* |
|  |
| **Revisión*: [1.0]*** |
| **10/09/2025** |

Integrantes:

* Tomás del Fierro
* Cristóbal Martínez

Profesor: Helton Bustos

|  |
| --- |
| Especificación de Requisitos según estándar de IEEE 830. |

Contenido

[Ficha del documento 4](#_Toc208410078)

[1. Introducción 5](#_Toc208410079)

[1.1. Propósito 5](#_Toc208410080)

[Este documento describe los requisitos para el sistema Beyblade Store, una tienda online especializada en productos Beyblade. Está dirigido al equipo de desarrollo, stakeholders. 5](#_Toc208410081)

[1.2. Ámbito del Sistema 5](#_Toc208410082)

[1.3. Definiciones, Acrónimos y Abreviaturas 6](#_Toc208410083)

[ ERS: Especificación de Requisitos de Software 6](#_Toc208410084)

[ HTML: HyperText Markup Language 6](#_Toc208410085)

[ CSS: Cascading Style Sheets 6](#_Toc208410086)

[ JS: JavaScript 6](#_Toc208410087)

[ RF: Requisito Funcional 6](#_Toc208410088)

[ RNF: Requisito No Funcional 6](#_Toc208410089)

[ UI: User Interface 6](#_Toc208410090)

[ UX: User Experience 6](#_Toc208410091)

[1.4. Referencias 6](#_Toc208410092)

[1.5. Visión General del Documento 6](#_Toc208410093)

[2. Descripción General 7](#_Toc208410094)

[2.1. Perspectiva del Producto 7](#_Toc208410095)

[2.2. Funciones del Producto 7](#_Toc208410096)

[2.3. Características de los Usuarios 8](#_Toc208410097)

[2.4. Restricciones 9](#_Toc208410098)

[2.5. Suposiciones y Dependencias 9](#_Toc208410099)

[2.6. Requisitos Futuros 10](#_Toc208410100)

[3. Requisitos Específicos 11](#_Toc208410101)

[3.1 Requisitos comunes de las interfaces 12](#_Toc208410102)

[3.1.1 Interfaces de usuario 12](#_Toc208410103)

[3.1.2 Interfaces de hardware 12](#_Toc208410104)

[3.1.3 Interfaces de software 12](#_Toc208410105)

[3.1.4 Interfaces de comunicación 12](#_Toc208410106)

[No hay interfaces de comunicación con otros sistemas en esta versión. Toda la comunicación es interna mediante localStorage API. 12](#_Toc208410107)

[3.2 Requisitos funcionales 12](#_Toc208410108)

[3.3 Requisitos no funcionales 14](#_Toc208410109)

[3.3.1 Requisitos de rendimiento 14](#_Toc208410110)

[3.3.2 Seguridad 14](#_Toc208410111)

[3.3.3 Fiabilidad 14](#_Toc208410112)

[3.3.4 Disponibilidad 14](#_Toc208410113)

[3.3.5 Mantenibilidad 14](#_Toc208410114)

[3.3.6 Portabilidad 14](#_Toc208410115)

[3.4 Otros Requisitos 14](#_Toc208410116)

# Ficha del documento

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Modificación** |
| *10/09/2025* | *1.0* | *Cristobal Martinez, Tomas del Fierro* | *Creación del documento ERS* |
|  |  |  |  |

# 1. Introducción

## 1.1. Propósito

## Este documento describe los requisitos para el sistema Beyblade Store, una tienda online especializada en productos Beyblade. Está dirigido al equipo de desarrollo, stakeholders.

## 1.2. Ámbito del Sistema

**Nombre del sistema:** Beyblade Store

**Lo que el sistema hará:**

* Permitirá la visualización y gestión de productos Beyblade
* Implementará un sistema de autenticación y autorización de usuarios
* Gestionará carritos de compra con persistencia en localStorage
* Incluirá un panel administrativo para gestión de contenido
* Mostrará blogs y contenido informativo sobre Beyblade

**Lo que el sistema no hará:**

* No procesará pagos reales
* No se integrará con sistemas externos o APIs
* No enviará notificaciones por email
* No gestionará inventario en tiempo real

**Beneficios y objetivos:**

* Crear una experiencia de compra online intuitiva para fanáticos de Beyblade
* Establecer las bases para un sistema escalable de e-commerce
* Cumplir con los requisitos académicos del curso DSY1104

## 1.3. Definiciones, Acrónimos y Abreviaturas

## ERS: Especificación de Requisitos de Software

## HTML: HyperText Markup Language

## CSS: Cascading Style Sheets

## JS: JavaScript

## RF: Requisito Funcional

## RNF: Requisito No Funcional

## UI: User Interface

## UX: User Experience

## 1.4. Referencias

* IEEE Std 830-1998: Recommended Practice for Software Requirements Specifications
* Documento de instrucciones: DSY1104 Evaluación Parcial 1
* Mockups y diagramas de flujo del proyecto Beyblade Store

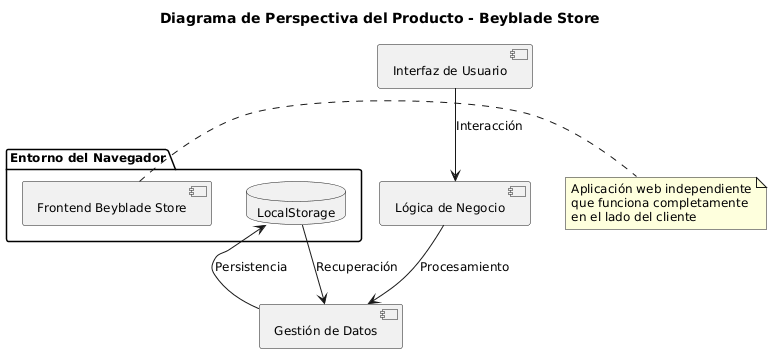
## 1.5. Visión General del Documento

Este documento sigue el estándar IEEE 830 y se organiza en tres secciones principales: Introducción (contexto del proyecto), Descripción General (factores que afectan al producto) y Requisitos Específicos (definición detallada de requisitos). Incluye anexos con casos de uso y diagramas adicionales.

# 2. Descripción General

## 2.1. Perspectiva del Producto

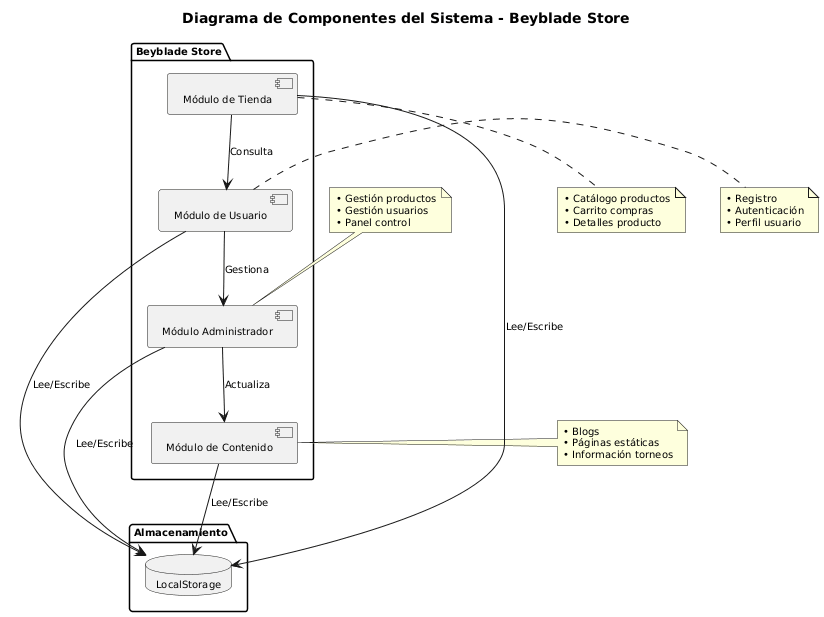
Beyblade Store es una aplicación web independiente que funciona completamente en el lado del cliente. No se integra con otros sistemas o productos en su versión inicial. El siguiente diagrama de bloques ilustra su arquitectura básica:



## 2.2. Funciones del Producto

El sistema Beyblade Store proporciona las siguientes funciones organizadas por módulos:

**Diagrama de Componentes del Sistema:**



## 2.3. Características de los Usuarios

**Administrador:**

* Nivel educacional: Educación superior
* Experiencia técnica: Conocimientos medios de PC y Excel
* Responsabilidades: Gestión completa del sistema

**Vendedor:**

* Nivel educacional: Educación media completa
* Experiencia técnica: Conocimientos básicos de navegación web
* Responsabilidades: Visualización de productos y órdenes

**Cliente:**

* Nivel educacional: Variado (desde educación básica)
* Experiencia técnica: Uso básico de navegadores web
* Responsabilidades: Navegación y compras en la tienda

## 2.4. Restricciones

* **Políticas de la empresa:** Debe seguir lineamientos de diseño de Duoc UC
* **Limitaciones del hardware:** Funciona en dispositivos con mínimo 2GB RAM
* **Interfaces con otras aplicaciones:** No hay integración con sistemas externos
* **Operaciones paralelas:** Soporta hasta 50 usuarios concurrentes
* **Funciones de auditoría:** Registro básico de actividades en consola
* **Funciones de control:** Validación de datos en frontend
* **Lenguaje(s) de programación:** HTML5, CSS3, JavaScript vanilla
* **Protocolos de comunicación:** HTTP/HTTPS para carga de recursos
* **Requisitos de habilidad:** Conocimientos en desarrollo web frontend
* **Criticidad de la aplicación:** Media (proyecto académico)
* **Consideraciones acerca de la seguridad:** Validación frontend, protección básica XSS

## 2.5. Suposiciones y Dependencias

**Suposiciones:**

1. Los usuarios tienen acceso a navegadores web modernos
2. JavaScript está habilitado en los navegadores de los usuarios
3. La resolución mínima soportada es 320px (móviles)
4. No se requiere conexión permanente a internet (funcionalidad offline básica)

**Dependencias:**

1. Navegadores web con soporte para HTML5 y CSS3
2. Compatibilidad con localStorage
3. No hay dependencias de servidores externos o APIs

## 2.6. Requisitos Futuros

* Integración con pasarelas de pago reales
* Sistema de notificaciones por email
* Aplicación móvil nativa
* Panel de analytics y reportes
* Integración con redes sociales
* Sistema de reseñas y valoraciones
* Gestión avanzada de inventario

# 3. Requisitos Específicos

Esta sección contiene los requisitos a un nivel de detalle suficiente como para permitir a los diseñadores diseñar un sistema que satisfaga estos requisitos, y que permita al equipo de pruebas planificar y realizar las pruebas que demuestren si el sistema satisface, o no, los requisitos. Todo requisito aquí especificado describirá comportamientos externos del sistema, perceptibles por parte de los usuarios, operadores y otros sistemas. Esta es la sección más larga e importante de la ERS. Deberán aplicarse los siguientes principios:

• El documento debería ser perfectamente legible por personas de muy distintas formaciones e intereses.

• Deberán referenciarse aquellos documentos relevantes que poseen alguna influencia sobre los requisitos.

• Todo requisito deberá ser unívocamente identificable mediante algún código o sistema de numeración adecuado.

• Lo ideal, aunque en la práctica no siempre realizable, es que los requisitos posean las siguientes características:

* **Corrección**: La ERS es correcta si y sólo si todo requisito que figura aquí (y que será implementado en el sistema) refleja alguna necesidad real. La corrección de la ERS implica que el sistema implementado será el sistema deseado.
* **No ambiguos**: Cada requisito tiene una sola interpretación. Para eliminar la ambigüedad inherente a los requisitos expresados en lenguaje natural, se deberán utilizar gráficos o notaciones formales. En el caso de utilizar términos que, habitualmente, poseen más de una interpretación, se definirán con precisión en el glosario.
* **Completos**: Todos los requisitos relevantes han sido incluidos en la ERS. Conviene incluir todas las posibles respuestas del sistema a los datos de entrada, tanto válidos como no válidos.
* **Consistentes**: Los requisitos no pueden ser contradictorios. Un conjunto de requisitos contradictorio no es implementable.
* **Clasificados**: Normalmente, no todos los requisitos son igual de importantes. Los requisitos pueden clasificarse por importancia (esenciales, condicionales u opcionales) o por estabilidad (cambios que se espera que afecten al requisito). Esto sirve, ante todo, para no emplear excesivos recursos en implementar requisitos no esenciales.
* **Verificables**: La ERS es verificable si y sólo si todos sus requisitos son verificables. Un requisito es verificable (testeable) si existe un proceso finito y no costoso para demostrar que el sistema cumple con el requisito. Un requisito ambiguo no es, en general, verificable. Expresiones como a veces, bien, adecuado, etc. Introducen ambigüedad en los requisitos. Requisitos como “en caso de accidente la nube tóxica no se extenderá más allá de 25Km" no es verificable por el alto costo que conlleva.
* **Modificables**: La ERS es modificable si y sólo si se encuentra estructurada de forma que los cambios a los requisitos pueden realizarse de forma fácil, completa y consistente. La utilización de herramientas automáticas de gestión de requisitos facilitan enormemente esta tarea.
* **Trazables**: La ERS es trazable si se conoce el origen de cada requisito y se facilita la referencia de cada requisito a los componentes del diseño y de la implementación. La trazabilidad hacia atrás indica el origen (documento, persona, etc.) de cada requisito. La trazabilidad hacia delante de un requisito R indica que componentes del sistema son los que realizan el requisito R.

## 3.1 Requisitos comunes de las interfaces

Descripción detallada de todas las entradas y salidas del sistema de software.

### 3.1.1 Interfaces de usuario

Las interfaces de usuario serán páginas web con una distribución de menú superior y un área de contenido principal. El diseño seguirá una paleta de colores azul, rojo y negro (temática Beyblade) con tipografía Arial o sans-serif. Todas las páginas mantendrán una estructura consistente con menú de navegación superior y footer informativo.

### 3.1.2 Interfaces de hardware

El sistema debe poder conectarse a dispositivos touch móviles y funcionar correctamente en resoluciones desde 320px (móviles) hasta 4K. No requiere hardware específico adicional.

### 3.1.3 Interfaces de software

* **Navegadores compatibles:** Chrome 90+, Firefox 88+, Edge 90+, Safari 14+
* **Sistemas operativos:** Windows 10+, macOS 10.14+, Linux Ubuntu 18.04+
* **No hay integración** con otros productos software.

### 3.1.4 Interfaces de comunicación

## No hay interfaces de comunicación con otros sistemas en esta versión. Toda la comunicación es interna mediante localStorage API.

## 3.2 Requisitos funcionales

**3.2.1 RF-001: Navegación por catálogo de productos**

**Actores:** Cliente, Vendedor, Administrador  
**Descripción:** El sistema debe permitir navegar por el catálogo de productos Beyblade  
**Secuencia de operaciones:**

1. Usuario accede a la página de productos
2. Sistema carga productos desde localStorage
3. Sistema muestra productos en formato grid o listado
4. Usuario puede filtrar por categorías disponibles

**3.2.2 RF-006: Registro de nuevo usuario**

**Actores:** Cliente  
**Descripción:** Formulario de registro con validación de datos  
**Comprobación de validez de entradas:**

* RUN chileno válido (formato correcto, dígito verificador)
* Email con dominios permitidos (@[duoc.cl](https://duoc.cl/), @profesor.duoc.cl, @gmail.com)
* Contraseña entre 4-10 caracteres
* Campos obligatorios completos

**3.2.3 RF-007: Inicio de sesión**

**Actores:** Cliente, Vendedor, Administrador  
**Descripción:** Autenticación de usuarios con credenciales válidas  
**Respuesta a situaciones anormales:**

* Credenciales incorrectas: mostrar mensaje de error
* Campos vacíos: destacar campos obligatorios
* Usuario no registrado: sugerir registro

**3.2.4 RF-032: Carrito de compras persistente**

**Actores:** Cliente  
**Descripción:** Gestión de productos en carrito con persistencia  
**Relaciones entre entradas y salidas:**

* Añadir producto → Incrementar contador carrito
* Eliminar producto → Decrementar contador carrito
* Modificar cantidad → Recalcular total inmediatamente

## 3.3 Requisitos no funcionales

### 3.3.1 Requisitos de rendimiento

* Tiempo de carga inicial: < 3 segundos en conexión de 10Mbps
* Renderizado de páginas: < 1 segundo después de carga inicial
* Operaciones de localStorage: < 100ms por operación
* Soporte para hasta 100 productos simultáneos

### 3.3.2 Seguridad

* Validación de todos los inputs en frontend
* Sanitización de datos para prevenir XSS básico
* Almacenamiento seguro en localStorage
* Restricción de acceso según roles de usuario

### 3.3.3 Fiabilidad

* Tasa de error: < 1% en operaciones críticas
* Recuperación de errores con mensajes al usuario
* Persistencia de datos ante cierre inesperado
* Funcionamiento estable por 8 horas continuas

### 3.3.4 Disponibilidad

* 100% de disponibilidad en entorno de desarrollo
* Funcionamiento en horario extendido (7:00 - 23:00)
* Compatibilidad con navegadores principales

### 3.3.5 Mantenibilidad

* Código modular y comentado
* Estructura de archivos organizada
* Documentación clara de funciones
* Uso de control de versiones Git

### 3.3.6 Portabilidad

* Compatibilidad con Windows, macOS, Linux
* Funcionamiento en iOS y Android
* Sin dependencias de plugins adicionales
* Responsive design para múltiples dispositivos

## 3.4 Otros Requisitos

* **Colaboración:** Uso de GitHub con commits descriptivos
* **Documentación:** Comentarios en código
* **Diseño:** Temática coherente con Beyblade
* **Accesibilidad:** Nivel básico AA de WCAG 2.1
* **Navegación:** Menú consistente en todas las páginas