DUOC UC - ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Especificación de Requisitos de Software

Proyecto: Amasanderia Masamagna

Revisión*: [99.99]*

[26/11/2025]

Integrantes:

-Camila Pino

-Gonzalo Navarrete

Especificación de Requisitos según estándar de IEEE 830.

Contenido

1. Introducción

Este documento constituye la Especificación de Requisitos Software (ERS) del sitio web “Amasanderia Masamagna”. En este documento, también se detalla el propósito del proyecto, el ámbito del sistema, definiciones y acrónimos, las referencias, y los requisitos funcionales, no funcionales y pruebas unitarias implementadas.

1.1. Propósito

El propósito de este documento es especificar los requisitos de software para el desarrollo del sitio web full-stack de la pastelería “Amasandería Masamagna”, además de estar dirigido a los propietarios de la amasandería como medio de validación de los requisitos y los desarrolladores sirviendo como guía técnica del proyecto.

1.2. Ámbito del Sistema

El sistema implementado es una aplicación web full-stack completa que incluye:

Tecnologías Implementadas:

- Frontend: React con componentes para Productos, Contacto, Login, Carrito.
- Backend: Spring Boot con API REST (Producto, Usuario, Carrito, ItemCarrito).
- Base de Datos: MySQL gestionada con XAMPP y MySQL Workbench.
- Comunicación: REST API entre React y Spring Boot.
- Testing: Vitest + React Testing Library para pruebas unitarias.

Funcionalidades implementadas:

- Catálogo de productos con imágenes, descripciones y precios reales.
- Sistema de autenticación y registro funcional.
- Carrito de compras persistente en base de datos.
- CRUD completo de Producto y Usuario (Crear, Listar, Actualizar, Eliminar, Desactivar)
- Gestión de usuarios con roles diferenciados.
- API REST documentada con Swagger.
- Encriptación de contraseñas .
- Manejo de sesión con token JWT
- 6 pruebas unitarias que cubren los componentes más importantes de React.
- Interfaz React moderna y responsive.

Lo que hace por ahora:

-Simula compras mediante un botón “PROCESAR PAGO”
-Gestion de usuarios
-Gestión de productos
-Generación de boletas

-Gestión de Boletas
-Encriptación de contraseñas
-Carrito funcional con backend
-Control de sesion mediante token JWT

Lo que no hace por ahora:

- No procesa pagos en línea.

Los beneficios del sistema son:

- Ampliar el alcance comercial de la pastelería.
- Proporciona base tecnológica para futuras integraciones.
- Facilitar la visibilidad de productos.

1.3. Definiciones, Acrónimos y Abreviaturas

- ERS: Especificación de Requisitos de Software.
- Frontend: Interfaz visual y navegable por el usuario.
- Backend: Lógica de negocio y base de datos que se integrará en futuras versiones.
- Carrito: Sección del sistema donde se agrupan productos seleccionados por el cliente antes de confirmar la compra.
- Login: iniciar sesión en el sitio web.
- Register: Registrarse en el sitio web.
- SPA: Single Page Application (Aplicación de una sola página).
- API REST: Interfaz de programación de aplicaciones RESTful.
- JPA: Java Persistence API (Persistencia de datos en Java).
- CRUD: Create, Read, Update, Delete (Operaciones básicas de datos).
- XAMPP: Entorno de desarrollo web con Apache, MySQL, TomCat.
- Vitest: Framework de testing para Vite.
- Jest DOM: Librería de matchers extendidos para testing.

1.4. Referencias

En esta subsección se mostrará una lista completa de todos los documentos referenciados en la ERS.

1.5. Visión General del Documento

El documento se organiza de la siguiente manera:

1. Introducción.
2. Descripción general del sistema full-stack.
3. Especificación de los requisitos (Funcionales, no Funcionales y Técnicos)
4. Especificación de pruebas unitarias implementadas.

2. Descripción General

El sistema implementado es una aplicación web full-stack operativa desarrollada con React, Spring Boot y MySQL, que superó el alcance de simulación inicial al digitalizar completamente las operaciones de la pastelería.

La plataforma permite a clientes explorar productos reales y gestionar pedidos mediante carrito persistente, mientras los administradores disponen de un panel integral para gestionar inventario, usuarios y órdenes a través de APIs REST documentadas, constituyendo una solución empresarial completa.

2.1. Perspectiva del Producto

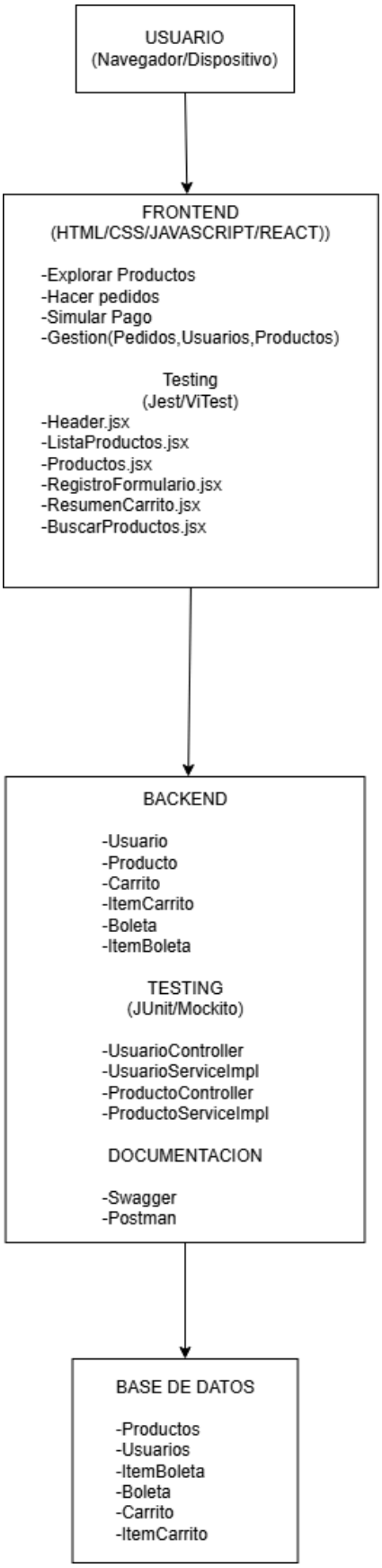
El sistema implementado es una aplicación web full-stack que incluye:

Arquitectura implementada:

- Frontend React: Aplicación moderna con componentes para Productos, Contacto, Login y Carrito.
- Backend Spring Boot: API REST completa con controladores, servicios, repositorios y entidades (Producto, Usuario, Carrito, ItemCarrito)
- Base de datos MySQL: Persistencia real mediante XAMPP y MySQL Workbench.
- Comunicación: Frontend (puerto 5173) consume API del backend mediante HTTP/REST.

Funcionalidades:

- Autenticación real con usuarios en base de datos.
- Carrito persistente que mantiene los ítems entre sesiones.
- CRUD completo de productos con operaciones en base de datos.
- Gestión administrativa real de usuarios y pedidos.
- Suite completa de pruebas unitarias (6 test exitosos).
- API documentada con Swagger/OpenAPI.


USUARIO
(Navegador/Dispositivo)
FRONTEND
(HTML/CSS/JAVASCRIPT/REACT))
-Explorar Productos
-Hacer pedidos
-Simular Pago
-Gestion(Pedidos,Usuarios,Productos)
Testing
(Jest/ViTest)
-Header.jsx
-ListaProductos.jsx
-Productos.jsx
-RegistroFormulario.jsx
-ResumenCarrito.jsx
-BuscarProductos.jsx
BACKEND
-Usuario
-Producto
-Carrito
-ItemCarrito
-Boleta
-ItemBoleta
TESTING
(JUnit/Mockito)
-UsuarioController
-UsuarioServiceImpl
-ProductoController
-ProductoServiceImpl
DOCUMENTACION
-Swagger
-Postman
BASE DE DATOS
-Productos
-Usuarios
-ItemBoleta
-Boleta
-Carrito
-ItemCarrito

2.2. Funciones del Producto

Frontend (React):

- Catálogo visual de productos con filtros y búsqueda.
- Carrito de compras con gestión de cantidades y persistencia.
- Sistema de autenticación y registro de usuarios.
- Panel administrativo con gestión completa.
- Formularios de contacto y configuración.

Backend (Spring Boot):

- API REST para gestión de productos, usuarios, carritos y pedidos.
- Validación de datos y manejo de errores HTTP.
- Persistencia con Spring Data JPA y MySQL.
- Documentación automática con Swagger.

Base de Datos (MySQL):

- Almacenamiento persistente de todos los datos transaccionales.
- Relaciones entre entidades (Producto, Usuario, Carrito, ItemCarrito).

2.3. Características de los Usuarios

- Cliente: Navega, agrega productos al carrito, realiza pedidos.
- Administrador: Gestiona productos, usuarios, pedidos desde panel admin.
- Vendedor: Visualiza y gestiona órdenes de pedidos.

2.4. Restricciones

Las restricciones del sistema son:

- Desarrollado en HTML, CSS, JavaScript y React.
- Backend: Spring Boot, Java 11.
- Base de datos: MySQL con XAMPP.
- Entorno: LocalHost (React: 5173. Spring Boot: 8080)
- Persistencia: Base de datos real MySQL.
- Lenguaje: Español.

2.5. Suposiciones y Dependencias

Las suposiciones y dependencias son:

- Los usuarios tendrán acceso a internet y a navegadores web.
- Java y Node.js instalados para desarrollo.
- XAMPP ejecutándose para base de datos MySQL.
- Conexión a internet para consumo de APIs y dependencias.

2.6. Requisitos Futuros

Dentro de los requisitos futuros, tenemos:

- Integración con pasarelas de pago (Webpay, MercadoPago).
- Sistema de notificaciones por email.
- Panel de reportes y analytics avanzados.
- Aplicación móvil complementaria.

3. Requisitos Específicos

En esta sección se describen los requisitos técnicos detallados para el desarrollo de la aplicación web full-stack, incluyendo especificaciones para el frontend React, backend Spring Boot, base de datos MySQL y pruebas unitarias, asegurando una implementación coherente entre todos los componentes del sistema.

3.1 Requisitos comunes de las interfaces

Descripción detallada de todas las entradas y salidas del sistema de software, incluyendo la comunicación entre el frontend React, backend Spring Boot, base de datos MySQL y el sistema de testing.

3.1.1 Interfaces de usuario

- Tecnología: Aplicación React con componentes.
- Header/Navegación Superior: Menú con acceso a Productos, Contacto, Login y Carrito.
- Panel de Administración: Sidebar con opciones (Panel Principal, Productos, Usuarios, Pedidos, Configuración, Cerrar Sesión)
- Diseño: Interfaz responsive con temática de pastelería.
- Navegación: React Router para transiciones suaves entre secciones sin recarga de página.
- Área de Contenido Dinámico: Sección central que cambia según la ruta:
    - Catálogo de Productos: Grid de tarjetas con imágenes, nombres, precios.
    - Detalle de Producto: Vista individual con descripción completa y stock.
    - Carrito de Compras: Lista de productos seleccionados con cantidades, total y checkout.
    - Formulario de Contacto: Campos para nombre, email y mensaje.
    - Panel Administrativo: Vistas de gestión para productos, usuarios, pedidos y configuración.

3.1.2 Interfaces de hardware

- Servidor: XAMPP con Apache, MySQL y entorno Java.
- Cliente: Compatible con desktop con pantalla táctil.
- Requisitos mínimos: 4GB RAM, conexión a internet, navegador moderno.

3.1.3 Interfaces de software

Indicar si hay que integrar el producto con otros productos de software.

- Spring Boot - MySQL: Conexión JDBC con entidades JPA y Spring Data.
- React - Spring Boot: Comunicación HTTP/REST con JSON.
- Swagger/OpenAPI: Documentación automática de endpoints.
- Vitest - React: Framework de testing para componentes React.
- Navegador - React: Ejecución de SPA con componentes.

3.1.4 Interfaces de comunicación

- Protocolo: HTTP/HTTPS para APIs REST.
- Formato: JSON para intercambio de datos.
- Origen: Configurado entre React (5173) y Spring Boot (8080).
- Testing: Comunicación aislada con mocks para APIs.
- Autenticación: Headers HTTP para gestión de sesiones.

3.2 Requisitos funcionales

Cliente:

- Requisito Funcional-01: Visualizar catálogo de productos con imágenes, precios y filtros por categoría.
- Requisito Funcional-02: Ver detalle completo de productos (descripción, ingredientes, stock, categoría).
- Requisito Funcional-03: Gestionar carrito de compras (agregar, modificar cantidades, desactivar, persistencia).
- Requisito Funcional-04: Proceso de checkout y generación de pedidos con validación de stock.
- Requisito Funcional-05: Formulario de contacto funcional con validaciones de longitud y formato.

Administrador:

- Requisito Funcional-06: Panel de administración con navegación completa y dashboard de estadísticas.
- Requisito Funcional-07: CRUD de productos (crear, listar, editar, eliminar/desactivar/activar).
- Requisito Funcional-08: Gestión de usuarios del sistema (registro, edición, cambio de estado, asignación de roles).
- Requisito Funcional-09: Administración de pedidos y cambios de estado.
- Requisito Funcional-10: Configuración general del sistema.
- Requisito Funcional-11: Control de sesiones (login/logout seguro).

Sistema:

- Requisito Funcional-12: Autenticación y autorización por roles (Cliente, Vendedor, Administrador).
- Requisito Funcional-13: Comunicación frontend-backend mediante APIs REST documentadas.
- Requisito Funcional-14: Persistencia de datos en base de datos MySQL con relaciones JPA.
- Requisito Funcional-15: Sistema de pruebas unitarias para validación de componentes React..

3.3 Requisitos no funcionales

- Requisito No Funcional-01: Carga de página principal en < 3 segundos (conexión 10Mbps)
- Requisito No Funcional-02: Compatibilidad con navegadores modernos
- Requisito No Funcional-03: Código modular y mantenible (React components + Spring services
- Requisito No Funcional-04: APIs RESTful documentadas con Swagger

- Requisito No Funcional-05: Interfaz responsive en dispositivos móviles y desktop
- Requisito No Funcional-06: Persistencia confiable en base de datos MySQL

3.3.1 Requisitos de rendimiento

- El sitio debe cargar en menos de 3 segundos en conexión de 10 Mbps
- Las respuestas de API deben ser menores a 500ms
- La aplicación debe manejar múltiples usuarios concurrentes

3.3.2 Seguridad

- Validación de datos en frontend y backend
- Autenticación con roles y protección de rutas
- Sanitización de inputs en formularios
- No se manejan datos de pago en esta versión

3.3.3 Fiabilidad

- El sistema debe estar disponible un 95% del tiempo.
- Manejo de errores y estados de carga.

3.3.4 Disponibilidad

- Accesible desde navegadores modernos (Chrome, Firefox, Edge, Safari).
- Interfaz responsive en dispositivos desktop.

3.3.5 Mantenibilidad

- Código modular en React (componentes reutilizables).
- Servicios separados en Spring Boot.
- Estructura clara de proyectos frontend y backend.
- 6 pruebas unitarias para facilitar el mantenimiento.

3.3.6 Portabilidad

- Compatible con navegadores: Chrome, Firefox, Edge, Safari.

3.4 Otros Requisitos

- Diseño visual coherente con identidad de pastelería.
- Experiencia de usuario intuitiva (máximo 3 clics para acciones principales).
- Documentación técnica completa para desarrolladores.
- Cobertura de pruebas unitarias para componentes.