TRES 3 DOS

Sistema de Gestión Integral de Eventos para Organizaciones Culturales

Documento de Especificación de Requerimientos de Software (SRS v2)

---

1. Introducción

Propósito:
El presente documento describe los requerimientos funcionales y no funcionales del sistema **Tres 3 Dos**, una aplicación web destinada a facilitar la gestión integral de eventos culturales o sociales de pequeña y mediana escala.
El documento servirá como guía técnica para el desarrollo, mantenimiento y validación del sistema, garantizando coherencia entre las necesidades de los usuarios y la implementación.

Alcance:
El sistema permite a organizaciones culturales autogestionadas o cooperativas:

- Gestionar inventario y consumo de productos durante eventos.
- Administrar presupuestos, ingresos, egresos y deudas.
- Organizar y controlar la venta de entradas.
  Gestionar usuarios, roles y accesos.
- Registrar operaciones con trazabilidad y control de permisos.

Público destinatario:

- **Usuarios finales:** integrantes de organizaciones culturales (tesorería, organización, abastecimiento, boletería).
- **Equipo técnico:** programadores, diseñadores, testers y responsables de soporte.

---

2. Descripción general

Perspectiva del producto:
Tres 3 Dos es una aplicación web modular basada en arquitectura de tres capas:

- **Presentación:**
  - Interfaz desarrollada en **React.js**, responsiva y compatible con navegadores modernos.

- **Lógica de negocio:**
    - API REST desarrollada en **Node.js (Express)** con control de acceso por **JWT**.
- **Acceso a datos:**
    - Base de datos **SQLite** local o en servidor, gestionada mediante consultas SQL parametrizadas.

Funcionalidad general del producto:

- Registro y control de inventario en tiempo real.
- Actualización automática del stock al asignar productos a eventos.
- Creación y gestión de eventos con control de insumos y entradas.
- Administración de ingresos, egresos y deudas (creadas manual o automáticamente).
- Control de usuarios, roles y permisos.
- Reportes financieros y operativos básicos.
- Persistencia de sesión mediante token JWT.

Características de los usuarios:

- Los usuarios poseen conocimientos informáticos básicos.
- Se prioriza una interfaz simple, con validaciones claras y acciones guiadas.
- Roles principales:

    - **Administrador general:** control total
    - **Administrador:** gestión global del sistema y usuarios.
    - **Tesorero:** control de presupuesto, ingresos, egresos y deudas.
    - **Abastecimiento:** administración de productos y stock.
    - **Organizador de eventos:** gestión de eventos y ventas de entradas.

---

3. Restricciones

- El sistema opera bajo **conectividad estándar a Internet**.
- Requiere **navegadores modernos (Chrome, Brave, Edge)**.
- Se utilizan **componentes de software libre** (Node.js, React, SQLite).
- La base de datos debe ser respaldable y portable.
- El sistema usa **autenticación JWT** con espiración controlada.

---

4. Suposiciones y dependencias

- Los usuarios disponen de conexión estable a Internet.
- El backend corre en un entorno con soporte Node.js y SQLite.
- Los datos persisten localmente o en servidor, con respaldo periódico.
- El sistema puede ampliarse para soportar otras bases de datos relacionales.

---

5. Requerimientos funcionales

5.1 Módulo de gestión de inventario

- Registrar productos con atributos: nombre, categoría, costo, proveedor y stock.
- Consultar stock en tiempo real.
- Actualizar automáticamente el stock al asignar o eliminar productos de eventos.
- Crear deudas automáticas cuando se asigne una cantidad superior al stock disponible.
- Permitir reposición manual o automática de productos.
- Mostrar alertas de stock bajo.

5.2 Módulo de presupuesto y tesorería

- Registrar ingresos, egresos y deudas.
- Crear pagos asociados a deudas.
- Cambiar estado de deuda a “saldada” al completar el pago.
- Generar reportes de flujo de caja.
- Integración automática con movimientos de inventario y eventos.

5.3 Módulo de eventos y entradas

- Crear, editar y eliminar eventos (nombre, fecha, precio de entrada).
- Asignar productos desde el inventario al evento (con actualización de stock).
- Editar o eliminar productos asignados con validación de stock.
- Registrar ventas de entradas y reflejar ingresos en tesorería.
  Consultar ventas totales y presupuesto generado por evento.

5.4 Módulo de usuarios y roles

- Registrar y autenticar usuarios mediante JWT.
- Mantener sesión activa si el token es válido al recargar la página.
  Redirigir automáticamente al login ante respuestas “401 Unauthorized”.
- Asignar roles y permisos a cada usuario.
- Listar usuarios, organizaciones y roles disponibles.
- Registrar auditoría de acciones críticas (opcional).

5.5 Usabilidad y transparencia

- Interfaz unificada y minimalista.
- Menús principales: **Eventos**, **Inventario**, **Tesorería**, **Usuarios**.
- Formularios validados en frontend y backend.
- Mensajes de confirmación claros (por ejemplo:
  “Stock actualizado y deuda registrada correctamente”).

---

6. Requerimientos no funcionales

Rendimiento:

- Tiempo de respuesta: ≤ 2 segundos en operaciones normales.
- Carga inicial de interfaz: ≤ 5 segundos en conexión de 10 Mbps.

Seguridad:

- Autenticación JWT con expiración de sesión automática.
- Contraseñas almacenadas mediante hash seguro (bcrypt).
- Validación de tokens en cada endpoint protegido.
- Redirección automática al login en caso de error 401.
- Canales de comunicación bajo HTTPS (en despliegue productivo).

Disponibilidad y fiabilidad:

- Copias de seguridad periódicas de la base de datos SQLite.
- Manejo de errores y respuestas coherentes del backend.

Mantenibilidad y escalabilidad:

- Código modular, organizado por controladores y servicios.
- Versionado con Git.
- Posibilidad de migrar la base de datos a PostgreSQL o MySQL sin cambios mayores.

Portabilidad:

- Compatible con navegadores modernos.
- Diseño responsive adaptable a dispositivos móviles.

---

7. Casos de uso

CU-1: Registro de venta de producto
Actor: Usuario con rol de cajero u organizador.
Flujo: Selecciona producto → Verifica stock → Registra movimiento → Actualiza inventario y presupuesto.

CU-2: Registro de compra o reposición
Actor: Usuario de abastecimiento o tesorería.
Flujo: Carga datos de producto → Actualiza stock → Registra deuda automáticamente.

CU-3: Pago de deuda
Actor: Tesorería.
Flujo: Selecciona deuda → Registra pago → Actualiza presupuesto y estado de deuda.

CU-4: Gestión de evento
Actor: Organización o boletería.
Flujo: Crea evento → Asigna productos → Vende entradas → Actualiza tesorería y stock.

CU-5: Autenticación y sesión persistente
Actor: Usuario del sistema.
Flujo: Inicia sesión → Recibe token JWT → Accede al sistema → En caso de "Unauthorized" es redirigido al login → Si recarga con token válido, la sesión persiste.

---

8. Requisitos de interfaz de usuario

- Menú principal accesible con navegación por secciones.
- Formularios validados y mensajes de error descriptivos.
- Modal para editar productos asignados con stock visible.
- Confirmaciones y alertas claras en operaciones críticas.
- Redirección automática al login cuando el token no es válido.

---

9. Dependencias e infraestructura

- **Backend:** Node.js (Express)
- **Frontend:** React.js
- **Base de datos:** SQLite
- **Autenticación:** JWT
- **Control de versión:** Git
- **Servidor opcional:** Nginx o Node.js standalone

---

10. Pruebas

- **Unitarias:** controladores y servicios backend.
- **Integración:** flujo completo evento–inventario–tesorería.
- **Seguridad:** validación de tokens, acceso restringido.
- **Usabilidad:** pruebas manuales de interfaz y flujos críticos.

---

11. Requerimientos de dominio

- Orientado a redes LAN o entornos conectados.
  Permite operar sin estructura empresarial compleja.
- Pensado para organizaciones culturales autogestionadas.