INVENTARIOX

MANUAL TÉCNICO

Sistema de Gestión de Inventarios

Versión 1.0.0

Diciembre 2025

Arquitectura: React + Firebase + Tailwind CSS

Desarrollado con tecnologías modernas para empresas del futuro

ÍNDICE

1. STACK TECNOLÓGICO

Tecnología	Versión	Función
JavaScript (ES6+)	ECMAScript 2020+	Lenguaje base del proyecto
React	^18.2.0	Framework de UI declarativo
Vite	^7.3.0	Build tool y dev server
Node.js	16.x o superior	Entorno de ejecución
npm	8.x o superior	Gestor de paquetes

Lenguaje Base: JavaScript moderno (ES6+) con módulos ESM

Framework: React 18 con Hooks (useState, useEffect, custom hooks)

Build Tool: Vite - Hot Module Replacement (HMR) ultra rápido

Package Manager: npm (Node Package Manager)

2. LIBRERÍAS CORE

Librería	Versión	Función Principal
firebase	^12.7.0	Base de datos NoSQL en tiempo real + Auth
tailwindcss	^3.3.0	Framework CSS utility-first
jspdf	^2.5.2	Generación de reportes PDF
jspdf-autotable	^3.8.4	Tablas automáticas en PDF
xlsx	^0.18.5	Exportación de hojas de cálculo Excel
react-hot-toast	^2.6.0	Notificaciones tipo toast
lucide-react	^0.263.1	Iconos SVG optimizados
recharts	^3.6.0	Gráficos y visualizaciones
sweetalert2	^11.10.0	Modales y alertas elegantes

Detalles de Implementación:

- Firebase: Firestore para persistencia, Auth para autenticación
- Tailwind: Diseño responsive mobile-first
- jsPDF: Reportes con encabezados corporativos personalizados
- XLSX: Formato de moneda y auto-width de columnas
- Toast: Feedback visual instantáneo al usuario

3. INFRAESTRUCTURA DE DESPLIEGUE

Control de Versiones:

- Plataforma: GitHub
- Rama principal: main
- Estrategia: Git Flow simplificado
- Commits: Mensajes descriptivos en español

Hosting y Deployment:

- Plataforma: Vercel (recomendado) o Netlify
- Despliegue: Automático desde rama main
- Build Command: npm run build
- Output Directory: dist/
- Entorno: Node.js 18.x

Variables de Entorno (.env):

```
VITE_FIREBASE_API_KEY=<tu-api-key>
VITE_FIREBASE_AUTH_DOMAIN=<tu-dominio>
VITE_FIREBASE_PROJECT_ID=<tu-proyecto>
VITE_FIREBASE_STORAGE_BUCKET=<tu-bucket>
VITE_FIREBASE_MESSAGING_SENDER_ID=<tu-sender>
VITE_FIREBASE_APP_ID=<tu-app-id>
```

& I M P O R T A N T E :

Las variables DEBEN tener el prefijo VITE_ para ser accesibles en el cliente.

4. GUÍA DE ESTILO UI/UX

4.1 PALETA DE COLORES

Nombre	Hex	RGB	Uso
Primary	#206DDA	rgb(32, 109, 218)	Botones principales, links activos
Secondary	#4CAF50	rgb(76, 175, 80)	Estados de éxito, confirmaciones
Dark BG	#111827	rgb(17, 24, 39)	Fondo principal modo oscuro
Dark Card	#1f2937	rgb(31, 41, 55)	Tarjetas y contenedores
Dark Border	#374151	rgb(55, 65, 81)	Bordes y separadores
Success	#10B981	rgb(16, 185, 129)	Mensajes de éxito
Warning	#F59E0B	rgb(245, 158, 11)	Advertencias
Error	#EF4444	rgb(239, 68, 68)	Errores y alertas
Danger	#DC3545	rgb(220, 53, 69)	Acciones destructivas

4.2 TIPOGRAFÍA

- Fuente Principal: Sistema (system-ui, -apple-system, sans-serif)
- Títulos H1: 24px - 32px, font-bold
- Títulos H2: 20px - 24px, font-semibold
- Texto Cuerpo: 14px - 16px, font-normal
- Texto Pequeño: 12px - 14px, font-normal
- Tablas: 13px - 15px, font-medium (encabezados)

4.3 COMPONENTES UI

Botones:

- Primario: bg-[#206DDA] hover:bg-[#1557b0] text-white
- Secundario: bg-gray-600 hover:bg-gray-700 text-white
- Peligro: bg-red-500 hover:bg-red-600 text-white
- Padding: px-4 py-2 (estándar), px-6 py-3 (grande)
- Border Radius: rounded-lg (8px)

Inputs y Formularios:

- Background: bg-[#374151] (dark) / bg-white (light)
- Border: border border-gray-600
- Focus: focus:ring-2 focus:ring-[#206DDA]
- Padding: px-4 py-2
- Border Radius: rounded-lg

4.3 COMPONENTES UI (continuación)

Tarjetas (Cards):

- Background: bg-[#1f2937] (dark) / bg-white (light)
- Border: border border-gray-700
- Shadow: shadow-lg
- Padding: p-6
- Border Radius: rounded-lg

Tablas:

- Encabezado: bg-[#374151] text-white font-medium
- Filas: hover:bg-[#374151] transition
- Bordes: border-collapse con border-gray-700
- Padding celdas: px-4 py-3

Modales:

- Overlay: bg-black bg-opacity-50 backdrop-blur-sm
- Contenedor: bg-[#1f2937] rounded-xl shadow-2xl
- Max Width: max-w-2xl
- Padding: p-6 o p-8

Sidebar:

- Background: bg-[#1f2937]
- Width: w-64 (abierto) / w-20 (colapsado)
- Item activo: bg-[#206DDA] text-white
- Item hover: bg-gray-700
- Transición: transition-all duration-300

4.4 ESPACIADO Y LAYOUT

- Contenedor Principal: p-4 sm:p-6 lg:p-8
- Grid Responsive: grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3
- Gap: gap-4 o gap-6
- Margen entre secciones: mb-6 o mb-8
- Breakpoints: sm(640px), md(768px), lg(1024px), xl(1280px)

5. ARQUITECTURA DE COMPONENTES

Estructura de carpetas:

```
src/
%%%   components/                    #  Componentes reut
%       %%%   Sidebar.jsx            #  Navegación lateral
%       %%%   Navbar.jsx             #  Barra superior
%       %%%   Logo.jsx               #  Logo corporativo
%       %%%   Toast.jsx              #  Notificaciones
%       %%%   Modal.jsx              #  Modales genéricos
%%%   pages/                         #  Páginas principale
%       %%%   Dashboard.jsx          #  Panel principal
%       %%%   Stock.jsx              #  Gestión de product
%       %%%   Inventory.jsx          #  Inventario físico
%       %%%   Orders.jsx             #  Pedidos
%       %%%   Analytics.jsx          #  Análisis visual
%       %%%   Settings.jsx           #  Configuración
%%%   services/                      #  Servicios externos
%       %%%   firebaseService.js     #  CRUD Firebase
%%%   hooks/                         #  Custom Hooks
%       %%%   useSettings.js         #  Hook de configura
%%%   utils/                         #  Utilidades
%       %%%   helpers.js             #  Funciones auxiliar
%       %%%   translations.js        #  i18n
%%%   config/                        #  Configuración
%       %%%   firebase.js            #  Config Firebase
%%%   App.jsx                        #  Componente raíz
%%%   main.jsx                       #  Entry point
```

Patrones de Diseño Utilizados:

- Container/Presenter: Separación de lógica y UI
- Custom Hooks: Lógica reutilizable (useSettings)
- Service Layer: Abstracción de Firebase
- Prop Drilling vs Context: Props para componentes cercanos

6. FLUJO DE TRABAJO

6.1 DESARROLLO LOCAL

1. **Clonar Repositorio:**

```
git clone <url-repositorio>
cd inventariox
```

2. **Instalar Dependencias:**

```
npm install
```

3. **Configurar Variables de Entorno:**

```
cp .env.example .env
```

Editar .env con las credenciales de Firebase

4. **Iniciar Servidor de Desarrollo:**

```
npm run dev
```

Aplicación disponible en http://localhost:3000

6.2 PROCESO DE DEPLOYMENT

1. **Commit de Cambios:**

```
git add .
git commit -m "Descripción del cambio"
```

2. **Push a GitHub:**

```
git push origin main
```

3. **Despliegue Automático:**

- Vercel detecta el push en main
- Ejecuta: npm run build
- Despliega carpeta dist/ a producción
- URL de producción actualizada automáticamente

6.3 COMANDOS NPM

Comando	Descripción
npm run dev	Inicia servidor de desarrollo (puerto 3000)
npm run build	Genera build de producción en dist/
npm run preview	Preview del build de producción
npm install	Instala todas las dependencias
npm update	Actualiza paquetes a últimas versiones

7. CONFIGURACIÓN DEL ENTORNO

7.1 FIREBASE SETUP

1. Crear proyecto en Firebase Console
2. Habilitar Authentication (Email/Password)
3. Crear Firestore Database (modo producción)
4. Habilitar Storage para logos de empresa
5. Copiar credenciales al archivo .env

7.2 VITE CONFIG

```
// vite.config.js
export default defineConfig({
   plugins: [react()],
   server: {
      port: 3000,
      open: true
   }
})
```

7.3 TAILWIND CONFIG

```
// tailwind.config.js
theme: {
   extend: {
      colors: {
         primary: '#206DDA',
         secondary: '#4CAF50',
         'dark-bg': '#111827'
      }
   }
}
```

8. ESTRUCTURA DE DATOS FIREBASE

Colección	Campos Principales	Descripción
products	nombre, stock, costo, precio, category, proveedor, userId	Catálogo de productos
providers	nombre, contacto, email, telefono, userId	Proveedores activos
orders	productos[], fecha, proveedor, total, estado, userId	Pedidos a proveedores
inventory_history	productos[], responsable, fecha, userId	Historial de inventarios
movements	tipo, producto, cantidad, motivo, fecha, userId	Movimientos de stock
company_settings	nombreEmpresa, logo, direccion, userId	Configuración empresa
mermas	producto, cantidad, motivo, fecha, userId	Registro de mermas

REGLAS DE SEGURIDAD FIRESTORE:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{collection}/{document} {
      allow read, write: if request.auth != null
                         && resource.data.userId == request.auth.uid;
    }
  }
}
```

MEJORES PRÁCTICAS

' CÓDIGO:

- Componentes funcionales con Hooks
- Nombres descriptivos en español
- Comentarios para lógica compleja
- Validación de datos antes de guardar
- Manejo de errores con try/catch

' UI/UX:

- Mobile-first responsive design
- Feedback visual inmediato (toast)
- Estados de loading claros
- Confirmación en acciones destructivas
- Accesibilidad (aria-labels)

' RENDIMIENTO:

- Lazy loading de componentes pesados
- Debounce en búsquedas
- Paginación en tablas grandes
- Optimización de imágenes

' SEGURIDAD:

- Variables de entorno para credenciales
- Validación en cliente Y servidor
- Reglas de Firestore estrictas
- HTTPS obligatorio en producción