Manual de Programador – Sistema de Inventario Multibodega

1. Estructura del Proyecto

```
inventario-app/
├── src/
│   ├── components/
│   │   ├── atoms/
│   │   ├── organisms/
│   │   └── pages/
│   ├── services/
│   ├── store/
│   ├── styles/
│   ├── types/
│   ├── App.tsx
│   └── main.tsx
├── public/
├── package.json
├── .env
```

2. Tecnologías Utilizadas

- React + TypeScript (UI)
- Vite (Build & Dev Server)
- Zustand (Estado global)
- TanStack React Query (Manejo de datos)
- Supabase (Backend + Autenticación + DB)
- Styled Components (Estilos CSS-in-JS)
- jsPDF + autoTable (PDFs)
- Atomic Design (Arquitectura de componentes)

3. Arquitectura y Patrones

- Atomic Design: Átomos, Organismos, Páginas.
- Zustand: Gestión de estados globales.
- React Query: Fetching y sincronización con Supabase.

- Supabase: CRUD de todos los recursos y autenticación.

4. Módulos Principales y Lógica

- Bodegas: CRUD, tabla bodegas.
- Productos: CRUD con soporte a fraccionables, tabla products.
- Inventario por Bodega: Consulta de stock, tabla inventario_bodega.
- Kardex: Entradas y salidas, tabla inventory_movements.
- Transferencias: Validación y actualización de inventarios, tabla transferencias_bodega.
- Otros: Personal, Categorías, Empresa (CRUD estándar).

5. Lógica de Inventario y Movimientos

- Cada movimiento afecta directamente inventario_bodega.
- Validación automática del stock disponible.
- Soporte a fracciones si el producto lo permite.

6. Exportación a PDF

- Uso de jsPDF y autoTable.
- Botón de exportación extrae datos de la tabla visible.
- Encabezados y pie de página automáticos.

7. Agregar Nuevas Funcionalidades

- Crear componente en src/components/pages/.
- Definir endpoint y lógica con React Query.
- Reutilizar formularios y tablas existentes.
- Agregar estilos con Styled Components.
- Validar inputs y probar funcionalidad.

8. Buenas Prácticas de Desarrollo

- Código limpio y tipado con TypeScript.
- Documentar funciones complejas.
- Uso de hooks para lógica reutilizable.
- Evitar código duplicado.

- Validación de datos frontend/backend.

- Versionar la base de datos cuando sea necesario.

9. Despliegue

- Recomendado: Vercel o Netlify.

- Variables en archivo .env: VITE_SUPABASE_URL, VITE_SUPABASE_KEY.

- Comando de compilación: npm run build.