📋 Guía Completa de Deployment - Sistema de Inventario Rossi

Para Principiantes - Paso a Paso

Esta guía te llevará desde tu proyecto local hasta tenerlo funcionando en internet, explicando cada paso de manera sencilla.

🚀 FASE 1: PREPARACIÓN DEL PROYECTO

1.1 Limpiar el Proyecto

Primero, vamos a limpiar archivos que no necesitamos en producción:

```
cd /home/ubuntu/rossi-inventory

# Eliminar archivos temporales y caché
rm -rf .next/
rm -rf node_modules/
rm -rf .vercel/
rm -rf dist/
rm -rf build/

# Limpiar logs si existen
rm -f *.log
```

¿Qué hace esto? Eliminamos carpetas que se generan automáticamente y no deben subirse a GitHub.

1.2 Verificar .gitignore

Crea o verifica que tengas este archivo `.gitignore` en la raíz de tu proyecto:

```
# Dependencies
node_modules/
/.pnp
.pnp.js

# Production builds
/.next/
/out/
/build
/dist

# Environment variables
.env
.env*.local
.env.production

# Database
*.db
*.db-journal

# Vercel
.vercel

# Logs
npm-debug.log*
yarn-debug.log*
yarn-error.log*

# IDE
.vscode/
.idea/

# OS
.DS_Store
Thumbs.db

# Prisma
prisma/migrations/
```

¿Para qué sirve? Le dice a Git qué archivos NO debe subir a GitHub (como contraseñas y archivos temporales).

1.3 Preparar Variables de Entorno para Producción

Crea un archivo `.env.example` con las variables que necesitarás:

```
# Crea el archivo de ejemplo
cat > .env.example << 'EOF'
# Database
DATABASE_URL="postgresql://usuario:password@host:5432/database_name"

# NextAuth
NEXTAUTH_SECRET="tu-secreto-super-seguro-aqui"
NEXTAUTH_URL="https://tu-app.vercel.app"

# App Configuration
APP_NAME="Sistema de Inventario Rossi"
APP_VERSION="1.0.0"
EOF
```

¿Qué es esto? Un archivo de ejemplo que muestra qué variables de entorno necesitas, sin incluir los valores reales (por seguridad).

1.4 Crear README.md Completo

```
cat > README.md << 'EOF'
# 🏢 Sistema de Inventario Rossi

Sistema completo de gestión de inventario desarrollado con NextJS 14 y PostgreSQL.

## ✨ Características

- 👤 **Autenticación Multi-Rol**: Admin, Warehouse, Viewer
- 📦 **Gestión de Productos**: CRUD completo con categorías
- 📊 **Dashboard Interactivo**: Estadísticas y gráficos en tiempo real
- 🔄 **Movimientos de Inventario**: Entradas, salidas y transferencias
- 📱 **Responsive Design**: Funciona en desktop y móvil
- 🌙 **Modo Oscuro**: Interfaz adaptable

## 🛠️ Stack Tecnológico

- **Frontend**: NextJS 14, TypeScript, Tailwind CSS
- **Backend**: NextJS API Routes, Prisma ORM
- **Base de Datos**: PostgreSQL
- **Autenticación**: NextAuth.js
- **Gráficos**: Recharts
- **UI**: shadcn/ui components

## 🌐 Demo en Vivo

- **URL**: [https://tu-app.vercel.app](https://tu-app.vercel.app)

### 👥 Usuarios de Prueba

| Rol | Email | Password |
|-----|-------|----------|
| Super Admin | admin@rossi.com | admin123 |
| Warehouse | almacen@rossi.com | admin123 |

## 🚀 Instalación Local

1. Clona el repositorio
```bash
git clone https://github.com/tu-usuario/rossi-inventory.git
cd rossi-inventory
```

1. Instala dependencias

```
npm install
```

1. Configura variables de entorno

```
cp .env.example .env
# Edita .env con tus valores
```

1. Configura la base de datos
```

```
npx prisma generate
npx prisma db push
npx prisma db seed
```

1. Ejecuta el proyecto

```
npm run dev
```

📝 Licencia

Este proyecto es privado y pertenece a Rossi Company.

👨‍💻 Desarrollado por

Tu nombre - [tu-email@email.com]
EOF

```
### 1.5 Verificar que Todo Funciona

```bash
# Instala dependencias limpias
npm install

# Verifica que compile sin errores
npm run build

# Si no hay errores, limpia el build
rm -rf .next/
```

✅ **Checkpoint:** Si el comando `npm run build` termina sin errores, tu proyecto está listo.

📦 FASE 2: SUBIR CÓDIGO A GITHUB

2.1 Inicializar Git (si no está inicializado)

```
# Ve a tu directorio del proyecto
cd /home/ubuntu/rossi-inventory

# Inicializa Git
git init
```

¿Qué hace `git init`? Convierte tu carpeta en un repositorio Git que puede trackear cambios.
```

2.2 Configurar Git (si es la primera vez)

```
# Configura tu nombre (reemplaza con el tuyo)
git config --global user.name "Tu Nombre"

# Configura tu email (usa el mismo de GitHub)
git config --global user.email "tu-email@gmail.com"
```

2.3 Agregar Todos los Archivos

```
# Agregar todos los archivos al "staging area"
git add .

# Verificar qué archivos se agregaron
git status
```

¿Qué hace `git add .`? Le dice a Git “prepara todos estos archivos para subirlos”.

![Conceptual: Terminal mostrando lista de archivos en verde listos para commit]

2.4 Hacer el Primer Commit

```
# Crear el commit con un mensaje descriptivo
git commit -m "🎉 Initial commit: Sistema de Inventario Rossi completo

- ✅ Autenticación multi-rol implementada
- ✅ CRUD de productos funcional
- ✅ Dashboard con estadísticas
- ✅ Movimientos de inventario
- ✅ UI responsive con modo oscuro
- ✅ Base de datos PostgreSQL configurada"
```

¿Qué hace `git commit`? Crea una “fotografía” de tu código en este momento.

2.5 Conectar con GitHub

IMPORTANTE: Antes de continuar, ve a GitHub.com y crea un repositorio nuevo llamado `rossi-inventory`.

![Conceptual: Pantalla de GitHub mostrando botón “New repository”]

```
# Conectar con tu repositorio de GitHub (reemplaza TU-USUARIO)
git remote add origin https://github.com/TU-USUARIO/rossi-inventory.git

# Verificar que se conectó correctamente
git remote -v
```

¿Qué hace `git remote add origin`? Conecta tu proyecto local con el repositorio de GitHub.

2.6 Subir el Código

```
# Subir tu código a GitHub por primera vez
git push -u origin main
```

¿Qué hace `git push`? Sube todo tu código a GitHub para que esté disponible online.

Si te pide usuario y contraseña, usa tu token de GitHub (no tu contraseña).

✅ **Checkpoint:** Ve a GitHub.com y deberías ver todos tus archivos allí.

🗄️ FASE 3: CONFIGURAR BASE DE DATOS EXTERNA

3.1 Elegir Servicio de Base de Datos

Recomendamos **Neon** por ser gratuito y fácil de usar:

1. Ve a neon.tech (https://neon.tech)
2. Regístrate con GitHub
3. Click en “Create a project”

![Conceptual: Dashboard de Neon con botón “Create a project”]

3.2 Crear la Base de Datos

1. **Nombre del proyecto**: `rossi-inventory`
2. **PostgreSQL Version**: 15 (por defecto)
3. **Region**: Elige la más cercana
4. Click “Create project”

![Conceptual: Formulario de creación de proyecto en Neon]

3.3 Obtener la URL de Conexión

1. En tu dashboard de Neon, busca “Connection string”
2. Copia la URL que empieza con `postgresql://`
3. Se verá así: `postgresql://username:password@host.neon.tech/database_name`

![Conceptual: Panel de Neon mostrando la connection string]

3.4 Configurar Variables en Desarrollo

Crea un archivo `.env.production` para las variables de producción:

```
cat > .env.production << 'EOF'
# Database
DATABASE_URL="postgresql://tu-usuario:tu-password@host.neon.tech/rossi_inventory"

# NextAuth
NEXTAUTH_SECRET="genera-un-secreto-super-largo-y-seguro-aqui-2024"
NEXTAUTH_URL="https://tu-app.vercel.app"

# App Configuration
APP_NAME="Sistema de Inventario Rossi"
APP_VERSION="1.0.0"
EOF
```

🔐 **IMPORTANTE:**
- Reemplaza la DATABASE_URL con la tuya de Neon
- Genera un NEXTAUTH_SECRET único (puedes usar: `openssl rand -base64 32`)

3.5 Migrar la Base de Datos

```
# Usar la nueva URL para migrar
DATABASE_URL="tu-url-de-neon-aqui" npx prisma db push

# Sembrar datos iniciales
DATABASE_URL="tu-url-de-neon-aqui" npx prisma db seed
```

✅ **Checkpoint:** Tu base de datos en Neon debería tener todas las tablas y datos iniciales.

⚡ FASE 4: CONFIGURAR VERCEL

4.1 Conectar GitHub con Vercel

1. Ve a vercel.com (https://vercel.com)
2. Click “Sign up with GitHub”
3. Autoriza Vercel para acceder a tus repositorios

![Conceptual: Página de login de Vercel con botón GitHub]

4.2 Importar tu Proyecto

1. En el dashboard de Vercel, click “New Project”
2. Busca “rossi-inventory” en la lista
3. Click “Import”

![Conceptual: Lista de repositorios de GitHub en Vercel]

4.3 Configurar el Proyecto

Configure Project:

- **Framework Preset**: Next.js (se detecta automáticamente)
- **Root Directory**: `./` (dejar por defecto)
- **Build Command**: `npm run build` (por defecto)
- **Output Directory**: `.next` (por defecto)
- **Install Command**: `npm install` (por defecto)

![Conceptual: Formulario de configuración del proyecto en Vercel]

4.4 Configurar Variables de Entorno

Antes de hacer deploy, configura las variables de entorno:

1. En la sección “Environment Variables”
2. Agrega cada variable una por una:

Name	Value
`DATABASE_URL`	`tu-url-completa-de-neon`
`NEXTAUTH_SECRET`	`tu-secreto-generado`
`NEXTAUTH_URL`	`https://rossi-inventory.vercel.app`
`APP_NAME`	`Sistema de Inventario Rossi`
`APP_VERSION`	`1.0.0`

🔴 **IMPORTANTE:** El `NEXTAUTH_URL` debe coincidir exactamente con la URL que te asigne Vercel.

![Conceptual: Panel de variables de entorno en Vercel]

4.5 Hacer el Primer Deploy

1. Click “Deploy”
2. Espera 2-3 minutos mientras Vercel construye tu app

![Conceptual: Pantalla de progreso del deploy]

✅ **Checkpoint:** Vercel te dará una URL como `https://rossi-inventory.vercel.app`

✅ FASE 5: VERIFICACIÓN Y TROUBLESHOOTING

5.1 Verificar que Todo Funciona

1. **Abre tu URL de Vercel** en una nueva pestaña
2. **Verifica la página de login** aparece correctamente
3. **Prueba login** con: admin@rossi.com / admin123
4. **Navega por las secciones** principales
5. **Verifica datos** en el dashboard

5.2 Problemas Comunes y Soluciones

🔴 Error: “Database connection failed”

Solución:

```
# Verifica tu DATABASE_URL en Vercel
# Ve a: Tu Proyecto → Settings → Environment Variables
# Asegúrate que la URL esté completa y correcta
```

🔴 Error: “NEXTAUTH_URL is not defined”

Solución:
1. Ve a Vercel → Settings → Environment Variables
2. Agrega `NEXTAUTH_URL` con tu URL exacta de Vercel

🔴 Error: “Build failed”

Solución:

```
# Prueba el build localmente primero
npm run build

# Si falla localmente, revisa los errores y corrígelos
# Luego haz push a GitHub para trigger nuevo deploy
```

🔴 Página se ve rota o sin estilos

Solución:

1. Ve a Vercel → Functions → Logs
2. Busca errores de CSS o JavaScript
3. Usualmente es un problema de rutas absolutas

5.3 Cómo Ver Logs de Errores

1. **En Vercel**:
   - Ve a tu proyecto → Functions → Logs
   - Aquí verás errores en tiempo real

2. **En tu navegador**:
   - Press F12 → Console
   - Busca errores en rojo

5.4 Cómo Hacer Updates Posteriores

```
# 1. Hacer cambios en tu código local
# 2. Probar que funcionen con:
npm run dev

# 3. Commit y push a GitHub:
git add .
git commit -m "✨ Descripción de los cambios"
git push origin main

# 4. Vercel automáticamente detectará los cambios y hará redeploy
```

🎯 **Pro Tip:** Cada push a la rama `main` trigger un nuevo deploy automáticamente.

📋 FASE 6: DOCUMENTACIÓN FINAL

🌐 URLs y Accesos del Proyecto

📍 **URLs Principales:**

- **Aplicación**: `https://tu-app-real.vercel.app`
- **GitHub Repo**: `https://github.com/tu-usuario/rossi-inventory`
- **Base de Datos**: Neon Dashboard

👥 **Credenciales de Acceso:**

Rol	Email	Password	Permisos
Super Admin	admin@rossi.com	admin123	Acceso completo
Warehouse	almacen@rossi.com	admin123	Inventario y movimientos

🔄 Workflow para Cambios Futuros

1. **Desarrollo Local:**
   ```
   bash
   git pull origin main  # Obtener últimos cambios
   # Hacer cambios...
   npm run dev  # Probar localmente
   ```

2. **Deploy:**
   ```
   bash
   git add .
   git commit -m "✨ Descripción del cambio"
   git push origin main  # Deploy automático
   ```

3. **Verificar:** Revisa tu URL de Vercel en 1-2 minutos

📚 Recursos Adicionales

Documentación Oficial:
- NextJS Docs (https://nextjs.org/docs)
- Vercel Docs (https://vercel.com/docs)
- Neon Docs (https://neon.tech/docs)
- Prisma Docs (https://www.prisma.io/docs)

Tutoriales Útiles:
- Git Básico (https://git-scm.com/docs)
- Environment Variables (https://vercel.com/docs/concepts/projects/environment-variables)
- Custom Domains (https://vercel.com/docs/concepts/projects/custom-domains)

🎯 Próximos Pasos Recomendados

1. **Dominio Personalizado**: Configura `inventario.rossi.com`
2. **Backup Automático**: Configura backups de la base de datos
3. **Monitoreo**: Configura alertas para errores
4. **Analytics**: Agrega Google Analytics
5. **Performance**: Optimiza imágenes y carga

📞 Soporte

Si tienes problemas:

1. **Revisa esta guía** paso a paso
2. **Consulta los logs** en Vercel
3. **Busca el error específico** en Google
4. **Revisa la documentación** oficial

✨ ¡FELICITACIONES!

Has deployado exitosamente el Sistema de Inventario Rossi. Tu aplicación ahora está disponible en internet 24/7 y puedes compartir la URL con quien necesites.

🎉 **Lo que has logrado:**
- ✅ Subiste tu código a GitHub profesionalmente
- ✅ Configuraste una base de datos en la nube
- ✅ Deployaste en Vercel con variables de entorno seguras
- ✅ Tienes un pipeline CI/CD automático
- ✅ Tu app es accesible desde cualquier lugar del mundo

🚀 **Tu app está LIVE en:** `https://tu-url-final.vercel.app`

Guía creada específicamente para el Sistema de Inventario Rossi
Última actualización: Agosto 2025