


Resumen Completo: Fix de Seed Módulos PWA - Producción


Fecha: 30 de Octubre de 2025

Commits: 45c61e7 , 9d088aa , c24aa0f





Estado:  Completado y Pusheado a GitHub

Problema Original

El usuario reportó que durante el deploy en EasyPanel persistía el siguiente error:

```
 Sincronizando módulos PWA...  
 Script encontrado: scripts/seed-modules.ts  
 Ejecutando seed de módulos...  
 tsx no encontrado, intentando con node + ts-node  
node:internal/modules/cjs/loader:1143  
  throw err;  
  ^  
  
Error: Cannot find module 'ts-node/register'  
Require stack:  
- internal/preload
```






Causa del Problema

1.  El script `seed-modules.ts` es TypeScript y requiere `tsx` o `ts-node`
 2.  Estos paquetes son **devDependencies** (no en producción)
 3.  El contenedor Docker de producción no tiene estas herramientas
 4.  El sistema fallaba al sincronizar módulos PWA en cada deploy
-

Solución Implementada

Crear Versión JavaScript para Producción

Archivo creado: `app/scripts/seed-modules.js`

-  Versión JavaScript pura (CommonJS)
-  Usa `require()` y `module.exports`
-  No requiere dependencias de desarrollo
-  Funcionalidad idéntica a la versión TypeScript
-  16 KB - Misma funcionalidad que `.ts`

```
const { PrismaClient } = require('@prisma/client');
const prisma = new PrismaClient();

async function seedModules() {
  console.log('🌱 Seeding PWA modules...');
  // ... resto del código
}

module.exports = { seedModules };
```

2 Actualizar Script de Inicio

Archivo modificado: `start-improved.sh`

Cambios implementados:

1. **Priorizar JS sobre TS:** Busca primero `seed-modules.js`
2. **Fallback inteligente:** Si no hay `.js`, intenta `.ts` (desarrollo)
3. **Mensajes claros:** Indica qué versión está usando
4. **Manejo de errores:** Mensaje claro si faltan `tsx/ts-node`

```
# Preferir versión JavaScript (producción) sobre TypeScript (desarrollo)
if [ -f "scripts/seed-modules.js" ]; then
  echo " 📁 Script encontrado: scripts/seed-modules.js (producción)"
  node scripts/seed-modules.js
elif [ -f "scripts/seed-modules.ts" ]; then
  echo " 📁 Script encontrado: scripts/seed-modules.ts (desarrollo)"
  # Intenta con tsx/ts-node si están disponibles
fi
```

3 Fixes Adicionales del Pre-Push Check

Durante el push, el sistema de verificación automática detectó y corrigió:

Fix 1: yarn.lock Symlink

```
❌ ERROR: yarn.lock era un symlink
✅ SOLUCIÓN: Convertido a archivo regular (495KB)
📝 Commit: 9d088aa
```

Fix 2: Schema.prisma Ruta Absoluta

```
❌ ERROR: output = "/home/ubuntu/escalafin_mvp/app/node_modules/.prisma/client"
✅ SOLUCIÓN: output = "../node_modules/.prisma/client"
📝 Commit: c24aa0f
```




Commits Realizados

1. Commit Principal: 45c61e7

fix: agregar versión JavaScript de seed-modules para producción

- Crear seed-modules.js (versión JavaScript para producción)
- Actualizar start-improved.sh para priorizar versión JS
- Fallback a versión TS solo en desarrollo
- Soluciona error de ts-node/register en producción
- Documentación completa del fix

Archivos modificados:

-  app/scripts/seed-modules.js (NUEVO)
-  start-improved.sh (MODIFICADO)
-  FIX_SEED_MODULES_JS_30_OCT_2025.md (NUEVO)
-  FIX_SEED_MODULES_JS_30_OCT_2025.pdf (NUEVO)

2. Commit Auto-Fix: 9d088aa

fix: convertir yarn.lock a archivo regular (pre-push check)

- Detectado y corregido automáticamente por pre-push hook
- Convierte symlink a archivo real
- Previene errores de Docker COPY










3. Commit Auto-Fix: c24aa0f

fix: cambiar output de schema.prisma a ruta relativa (pre-push check)

- Detectado y corregido automáticamente por pre-push hook
- Cambia ruta absoluta a relativa
- Previene errores de Prisma generate en Docker

Verificación

Pre-Push Checks Pasados

-  Proyecto usa Yarn (yarn.lock detectado)
-  yarn.lock es un archivo regular (495KB)
-  Sin rutas absolutas problemáticas
-  Dockerfile tiene verificación de node_modules
-  Dockerfile copia .yarn/ correctamente
-  schema.prisma tiene **output** path correcto (relativo)
-  start-improved.sh tiene shebang **correcto**: #!/bin/bash
-  Dockerfile configura HOME correctamente
-  Archivos críticos verificados

Estructura de Archivos

```
app/scripts/
├── seed-modules.js      ← NUEVO - Para producción (16KB)
├── seed-modules.ts      ← Existente - Para desarrollo (16KB)
├── setup-users-production.js
├── seed-whatsapp.ts
└── seed.ts
```

Deploy en EasyPanel

Pasos para el Usuario

1. Pull del último commit en EasyPanel:

```
bash
```

```
Commit: c24aa0f
```

```
Mensaje: "fix: cambiar output de schema.prisma a ruta relativa"
```

2. Limpiar cache de build:


- En EasyPanel: Settings → Build Cache → Clear

3. Rebuild completo:


- Trigger nuevo build desde GitHub

4. Verificar logs de startup:

Deberías ver:


```
 Sincronizando módulos PWA...
```

```
 Script encontrado: scripts/seed-modules.js (producción)
```

```
 Ejecutando seed de módulos...
```

```
Processing module: Vista General del Dashboard (dashboard_overview)
```

```
...
```

```
 Módulos PWA sincronizados exitosamente
```

1. Verificar health check:

- URL pública debe responder
- No debe haber errores de módulos faltantes

Resultado Esperado

En Producción (EasyPanel/Docker)

Sin errores de ts-node:

-  El script seed-modules.js se ejecuta correctamente
-  Usa Node.js directamente (sin dependencias extra)

Módulos PWA sincronizados:

- ✓ 35+ módulos creados/actualizados automáticamente
- ✓ Permisos de roles configurados correctamente

✓ Sistema arranca completamente:

- ✓ **Next.js** inicia en puerto 3000
- ✓ Health check responde OK
- ✓ Aplicación accesible públicamente

En Desarrollo Local

✓ Ambas versiones funcionan:

- ✓ Puede usar `seed-modules.js` (producción-like)
- ✓ Puede usar `seed-modules.ts` (con `tsx/ts-node`)
- ✓ Hot reload sigue funcionando con TypeScript



Sistema de Validación Mejorado

El `pre-push-check.sh` ahora valida automáticamente:

1. ✓ **Yarn Lock:** No debe ser symlink
2. ✓ **Schema Prisma:** Sin rutas absolutas
3. ✓ **Shebangs:** Scripts usan `/bin/bash` no `/bin/sh`
4. ✓ **Dockerfile HOME:** Variable HOME configurada
5. ✓ **Node Modules:** Dockerfile verifica existencia



Lecciones Aprendidas

✓ Buenas Prácticas

1. **Separar Dev y Prod:**
 - Scripts TypeScript para desarrollo
 - Scripts JavaScript para producción
 - Priorizar JS en runtime
2. **Pre-Push Validation:**
 - Detecta problemas antes de GitHub
 - Auto-fix cuando es posible
 - Previene errores de build
3. **Documentación:**
 - Cada fix documentado en .md
 - Commits descriptivos
 - PDFs para referencia rápida

⚠ Evitar en el Futuro

1. ❌ No ejecutar TypeScript directo en producción
2. ❌ No usar devDependencies en runtime
3. ❌ No usar symlinks para archivos críticos
4. ❌ No usar rutas absolutas en configs

Estado del Proyecto

Commits en GitHub

- ✓ c24aa0f - fix: cambiar **output** de schema.prisma a ruta relativa
- ✓ 9d088aa - fix: convertir yarn.lock a archivo regular
- ✓ 45c61e7 - fix: agregar versión JavaScript de seed-modules
- ✓ 37c1e0d - EasyPanel config verified **and** working

Archivos Críticos Verificados

- | | |
|-------------------------------|---------------------------|
| ✓ app/scripts/seed-modules.js | - 16KB (NUEVO) |
| ✓ app/scripts/seed-modules.ts | - 16KB (original) |
| ✓ app/yarn.lock | - 495KB (archivo regular) |
| ✓ app/prisma/schema.prisma | - Ruta relativa |
| ✓ start-improved.sh | - Usa seed-modules.js |
| ✓ Dockerfile | - Configura HOME correcto |

Sistema de Validación

- | | |
|--------------------------------------|-------------------------|
| ✓ scripts/pre-push-check.sh | - Validación completa |
| ✓ scripts/pre-build-check.sh | - Verificación de build |
| ✓ scripts/validate-absolute-paths.sh | - Detector de rutas |

Próximos Pasos

Inmediatos (Usuario)

1. ☐ Pull del commit c24aa0f en EasyPanel
2. ☐ Limpiar cache de build
3. ☐ Rebuild completo
4. ☐ Verificar logs: debe mostrar "seed-modules.js (producción)"
5. ☐ Confirmar que no hay errores de ts-node
6. ☐ Verificar módulos PWA en admin/modules

Mantenimiento Futuro

1. ✓ Mantener ambas versiones sincronizadas (.js y .ts)
2. ✓ Versión JS es la oficial para producción
3. ✓ Versión TS útil para desarrollo local
4. ✓ Pre-push checks previenen regresiones



Referencias

- `FIX_SEED_MODULES_JS_30_OCT_2025.md` - Documentación detallada
 - `FIX_SHELL_BASH_HOME_30_OCT_2025.md` - Fix previo relacionado
 - `FIX_PRISMA_RUTA_RELATIVA_30_OCT_2025.md` - Fix de Prisma paths
 - `AUTO_SEED_MODULOS_30_OCT_2025.md` - Documentación original de seed
-



Checklist Final

- ☒ Crear `seed-modules.js` para producción
 - ☒ Actualizar `start-improved.sh` para usar `.js`
 - ☒ Verificar que Dockerfile copia scripts/
 - ☒ Documentar el fix completo
 - ☒ Pasar pre-push checks
 - ☒ Corregir yarn.lock symlink
 - ☒ Corregir schema.prisma ruta absoluta
 - ☒ Push exitoso a GitHub
 - ☒ Crear resumen completo
 - ☐ Usuario pull en EasyPanel
 - ☐ Usuario rebuild completo
 - ☐ Verificación en producción
-

Estado: ☒ Código pusheado, listo para deploy

Última actualización: 30 de Octubre de 2025, 18:05 UTC

Commit actual: `c24aa0f`

Branch: `main`

END OF DOCUMENT