Fix: Seed de Módulos PWA con JavaScript para Producción

Fecha: 30 de Octubre de 2025
Tipo: Bug Fix - Producción
Severidad: Alta
Estado: ✅ Resuelto

🐛 Problema

Durante el deploy en EasyPanel, el script de sincronización de módulos PWA fallaba con el siguiente error:

```
Error: Cannot find module 'ts-node/register'
Require stack:
- internal/preload
```

Causa Raíz

- El script `seed-modules.ts` es un archivo TypeScript que requiere `tsx` o `ts-node` para ejecutarse
- Estos paquetes son `devDependencies` y **no están disponibles en producción** (imagen Docker)
- El script `start-improved.sh` intentaba ejecutar el archivo TypeScript directamente
- Esto causaba que la sincronización de módulos PWA fallara en cada deploy

✅ Solución Implementada

1. Creación de Versión JavaScript

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

- Versión JavaScript pura del script TypeScript
- Funcionalidad idéntica usando `require()` y CommonJS
- No requiere dependencias adicionales de desarrollo
- Listo para producción

```
// Versión JavaScript para producción
const { PrismaClient } = require('@prisma/client');
const prisma = new PrismaClient();
// ... resto del código
```

2. Actualización de Start Scripts

Archivo modificado: `start-improved.sh`

Ahora el script:
1. **Prioriza** la versión JavaScript (`seed-modules.js`) para producción
2. **Fallback** a la versión TypeScript (`seed-modules.ts`) para desarrollo local
3. **Manejo robusto** de errores cuando tsx/ts-node no están disponibles

```
# 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)"
    # Intentar con tsx/ts-node...
fi
```

3. Beneficios

✅ **Funciona en producción** - No requiere devDependencies
✅ **Compatible con desarrollo** - Mantiene soporte para TypeScript
✅ **Sin cambios en Dockerfile** - El directorio scripts ya se copia completo
✅ **Idempotente** - Seguro ejecutar múltiples veces
✅ **Mensajes claros** - Indica qué versión está usando

🧪 Verificación

En Producción (Docker/EasyPanel)

El startup log debería mostrar:

```
🔄 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
```

En Desarrollo Local

El startup log puede mostrar cualquiera de las dos versiones:

```
🔄 Sincronizando módulos PWA...
  📂 Script encontrado: scripts/seed-modules.ts (desarrollo)
  🚀 Ejecutando seed de módulos...
  ...
```

📋 Archivos Modificados

```
MODIFICADO:
- start-improved.sh (líneas 62-121)

CREADO:
- app/scripts/seed-modules.js (nuevo archivo para producción)

SIN CAMBIOS:
- app/scripts/seed-modules.ts (versión original de desarrollo)
- Dockerfile (ya copia scripts/)
```

🔄 Deploy Steps

1. ✅ Crear `seed-modules.js`
2. ✅ Actualizar `start-improved.sh`
3. ✅ Commitear y pushear cambios
4. 🔲 Pull en EasyPanel
5. 🔲 Limpiar cache de build
6. 🔲 Rebuild completo
7. 🔲 Verificar logs de startup

🎯 Resultado Esperado

- ✅ El script de seed debe ejecutarse correctamente en producción
- ✅ Los módulos PWA se sincronizan automáticamente en cada deploy
- ✅ No hay errores relacionados con `ts-node` o `tsx`
- ✅ El sistema arranca completamente sin fallos

📝 Notas

- Ambas versiones (.js y .ts) tienen la misma funcionalidad
- Mantener ambas versiones sincronizadas si se hacen cambios
- La versión JavaScript es la oficial para producción
- La versión TypeScript es útil para desarrollo local con hot-reload

---

Commit: `fix: agregar versión JavaScript de seed-modules para producción`
Branch: `main`
Autor: EscalaFin Deploy Bot