

# Registro Histórico de Fixes Aplicados

## Propósito de este Documento

Este documento mantiene un registro detallado de todos los problemas encontrados y corregidos durante el desarrollo y despliegue de EscalaFin, junto con sus soluciones. Es una referencia rápida para evitar repetir los mismos errores.

## Octubre 28-29, 2025 - Sincronización con CitaPlanner

### FIX #1: Ruta Absoluta en schema.prisma

#### Problema Detectado:

```
Error: @prisma/client could not find Prisma Client
```

#### Causa Raíz:

```
// schema.prisma - INCORRECTO
generator client {
  provider = "prisma-client-js"
  output   = "/app/node_modules/.prisma/client"
}
```

La ruta absoluta `/app/` solo existe en el contenedor Docker, no en desarrollo local.

#### Solución Aplicada:

```
// schema.prisma - CORRECTO
generator client {
  provider = "prisma-client-js"
  output   = "../node_modules/.prisma/client"
}
```

**Commit:** `ddfba6`

#### Prevención:

- Script de revisión verifica rutas absolutas en schema.prisma
- Usar siempre rutas relativas en configuraciones

### FIX #2: Referencias a yarn.lock (Proyecto usa NPM)

#### Problema Detectado:

```
Error: Cannot find module 'yarn.lock'
Next.js buscando yarn.lock en /app/..
```

### Causa Raíz:

1. next.config.js tenía outputFileTracingRoot: '/app/..'
2. Dockerfile copiaba yarn.lock innecesariamente
3. Next.js asumía que el proyecto usa Yarn

### Solución Aplicada:

```
# Dockerfile - ANTES (INCORRECTO)
COPY package*.json yarn.lock ./
```

```
# Dockerfile - DESPUÉS (CORRECTO)
COPY package*.json ./
RUN touch ../yarn.lock
```

```
// next.config.js - CORRECCIÓN
// Se mantiene outputFileTracingRoot pero se crea yarn.lock dummy
```

**Commit:** ddfbaf6

### Prevención:

- Script verifica que no haya referencias a yarn en Dockerfile
- Mantener consistencia: proyecto usa NPM exclusivamente

## FIX #3: Scripts Excluidos del Docker Build

### Problema Detectado:

```
! scripts/setup-users-production.js no encontrado
! start-improved.sh no encontrado en contenedor
```

### Causa Raíz:

.dockerignore excluía archivos necesarios:

```
*.sh
scripts/
```

### Solución Aplicada:

```
# .dockerignore - CORRECTO
# Excluir scripts de testing y desarrollo
test-*.sh
build-*.sh
diagnostico-*.sh

# NO excluir scripts de producción:
# start-improved.sh ✓
# emergency-start.sh ✓
# healthcheck.sh ✓
# scripts/ ✓
```

```
# Dockerfile - Añadido
COPY scripts/ /app/scripts/
COPY *.sh /app/
```

**Commit:** ddfbaf6

#### Prevención:

- Script verifica existencia de scripts críticos
- Revisar .dockerignore antes de cada deploy



## FIX #4: bcryptjs Module Not Found

#### Problema Detectado:

```
Error: Cannot find module 'bcryptjs'
```

#### Causa Raíz:

En modo standalone de Next.js, las dependencias no se copian automáticamente a `node_modules/` del runtime.

#### Solución Aplicada:

```
# Dockerfile - Builder stage
RUN npm ci
RUN npm run build

# Runtime stage
COPY --from=builder /app/node_modules ./node_modules
COPY --from=builder /app/node_modules/.prisma ./node_modules/.prisma
```

```
# start-improved.sh
export NODE_PATH=/app/node_modules
```

**Commit:** ddfbaf6

#### Prevención:

- Script verifica que bcryptjs esté en dependencies
- Configurar NODE\_PATH en scripts de inicio

---

## **FIX #5: NODE\_PATH No Configurado**

### **Problema Detectado:**

```
Error: Cannot find module 'bcryptjs'  
Error: Cannot find module 'jsonwebtoken'
```

Incluso con node\_modules copiado, Node no los encontraba.

### **Causa Raíz:**

Next.js standalone mode no configura NODE\_PATH automáticamente.

### **Solución Aplicada:**

```
# start-improved.sh  
export NODE_PATH=/app/node_modules  
export LD_LIBRARY_PATH=/app/node_modules/.prisma/client:$LD_LIBRARY_PATH  
  
echo "NODE_PATH configurado: $NODE_PATH"  
node --version  
npm --version
```

**Commit:** ddfbaf6

### **Prevención:**

- Script verifica presencia de NODE\_PATH en start-improved.sh
- Documentar en README.md



## **Octubre 27-28, 2025 - Alineación de Versiones**

## **FIX #6: Versiones de Dependencias Desalineadas**

### **Problema Detectado:**

Build fallaba con errores de compatibilidad entre:

- Node.js 20 vs 18
- Prisma 6.9.0 vs 6.7.0
- Next.js diferentes versiones

### **Solución Aplicada:**

```
// package.json - Alineado con CitaPlanner  
{  
  "dependencies": {  
    "@prisma/client": "6.7.0"  
  },  
  "devDependencies": {  
    "prisma": "6.7.0"  
  }  
}
```

```
# Dockerfile
FROM node:18-alpine AS builder
```

**Commit:** ddfbaf6

#### Prevención:

- Mantener parity con CitaPlanner
- Script `verify-versions.sh` para verificar alineación



## FIX #7: Prisma Client Output Path

#### Problema Detectado:

```
Error: @prisma/client generated incorrectly
```

#### Causa Raíz:

Prisma generaba el client en ubicación incorrecta para standalone mode.

#### Solución Aplicada:

```
generator client {
  provider = "prisma-client-js"
  output   = "../node_modules/.prisma/client"
}
```

```
COPY --from=builder /app/node_modules/.prisma ./node_modules/.prisma
```

**Commit:** ddfbaf6



## Octubre 26-27, 2025 - Corrección de Dashboards



## FIX #8: Header Duplicado en Dashboards

#### Problema Detectado:

Asesor y Cliente dashboards mostraban dos headers (MainLayout + header interno).

#### Solución Aplicada:

```
// Eliminado de asesor-dashboard.tsx y cliente-dashboard.tsx
// ✗ REMOVIDO
<header className="bg-escalafin-navy-600/90 backdrop-blur-lg border-b border-
escalafin-turquoise-500/20 p-4 sticky top-0 z-10">
  { /* ... */ }
</header>
```

Ya existe en `MainLayout` el navbar global.

**Commit:** Incluido en actualizaciones de dashboards

**Prevención:**

- Usar solo MainLayout para navegación global
  - Componentes de dashboard solo contenido específico
- 

**FIX #9: Módulos Faltantes en Admin Dashboard****Problema Detectado:**

Admin dashboard no mostraba módulos críticos:

- Gestión de Archivos
- Centro de Notificaciones
- Configuración del Sistema
- Gestión de Módulos
- Almacenamiento/Storage
- Recarga de Mensajes WhatsApp

**Solución Aplicada:**

Se añadieron todos los módulos al dashboard con sus respectivas rutas:

```
// admin-dashboard.tsx
{
  title: 'Gestión de Archivos',
  icon: Folder,
  href: '/app/admin/files',
  // ...
}
```

**Commit:** Actualización de admin-dashboard

---

**Octubre 25-26, 2025 - Branding y Diseño****FIX #10: Actualización de Colores del Logo****Problema:**

El sistema usaba colores genéricos, no los del logo de EscalaFin.

**Solución Aplicada:**

```
// tailwind.config.ts
colors: {
  'escalafin-navy': {
    50: '#f0f5ff',
    // ...
    950: '#0a1628',
    DEFAULT: '#1a3a5c',
  },
  'escalafin-turquoise': {
    50: '#ecfeff',
    // ...
    950: '#042f2e',
    DEFAULT: '#4dd0e1',
  },
  // ...
}
```

**Commit:** Actualización de branding



## Resumen de Fixes por Categoría



### Docker & Build (7 fixes)

1. Ruta absoluta schema.prisma
2. Referencias a yarn.lock
3. Scripts excluidos
4. bcryptjs module missing
5. NODE\_PATH no configurado
6. Versiones desalineadas
7. Prisma output path



### UI/UX (3 fixes)

1. Header duplicado
2. Módulos faltantes admin
3. Colores de branding



### Seguridad & Auth (Pendiente revisión)

- Login redirect loops
- Token expiration handling
- Session management



### Base de Datos (Pendiente revisión)

- Migrations en producción
- Seed data consistency
- Connection pooling

## Patrón de Problemas Recurrentes

---

### Problema Tipo A: Rutas Absolutas

**Identificador:** Uso de `/app/` , `/home/` , etc. en configuraciones

**Solución:** Siempre usar rutas relativas

**Detección:** Script busca rutas absolutas en archivos de config

### Problema Tipo B: Package Manager Mixup

**Identificador:** Referencias a yarn en proyecto NPM

**Solución:** Usar exclusivamente NPM

**Detección:** Script busca referencias a yarn

### Problema Tipo C: Files No Copiados

**Identificador:** "File not found" en runtime pero existe en repo

**Solución:** Revisar `.dockerignore` y Dockerfile COPY

**Detección:** Script verifica archivos críticos

### Problema Tipo D: Module Not Found

**Identificador:** "Cannot find module" en runtime

**Solución:** Configurar `NODE_PATH` y copiar `node_modules`

**Detección:** Script verifica dependencies y `NODE_PATH`

---

## Checklist Pre-Deploy

---

Usar `scripts/revision-fix.sh` que verifica automáticamente:

- ☐ No rutas absolutas en configs
  - ☐ No referencias a yarn
  - ☐ Scripts necesarios presentes
  - ☐ `.dockerignore` correcto
  - ☐ Dependencias críticas instaladas
  - ☐ `NODE_PATH` configurado
  - ☐ Dockerfile estructura correcta
  - ☐ Prisma config correcto
  - ☐ Variables env documentadas
  - ☐ Package manager consistente
- 

## Contacto y Mantenimiento

---

**Última actualización:** 29 de Octubre, 2025

**Mantenedor:** Equipo de Desarrollo EscalaFin

**Script de revisión:** `scripts/revision-fix.sh`

**Documentación:** Este archivo + `GUIA_USO_SCRIPT_REVISION.md`

---




## Proceso de Actualización

---

Cuando se encuentre y corrija un nuevo problema:

1. **Documentar aquí** con el formato:
    - Problema detectado
    - Causa raíz
    - Solución aplicada
    - Commit hash
    - Prevención
  2. **Actualizar script** `revision-fix.sh` si es automatizable
  3. **Actualizar guía** de uso del script
  4. **Commit y push** con mensaje descriptivo
- 

 **Objetivo:** Zero regresiones. Cada problema se resuelve una sola vez.