



REGLAS CRÍTICAS - Prisma + Docker



NUNCA HACER

1. NO usar rutas absolutas en schema.prisma

✗ INCORRECTO:

```
generator client {
  provider = "prisma-client-js"
  output = "/home/ubuntu/muebleria_la_economica/app/node_modules/.prisma/client"
}
```

✓ CORRECTO:

```
generator client {
  provider = "prisma-client-js"
  binaryTargets = ["native", "linux-musl-arm64-openssl-3.0.x"]
  // NO incluir línea "output" - usa la ubicación predeterminada
}
```

¿Por qué? Las rutas absolutas locales NO existen en el contenedor Docker, causando que el Prisma Client no se genere y el build falle.

2. NO modificar schema.prisma sin validar

Antes de cada commit que modifique `prisma/schema.prisma`:

```
# Ejecutar validación
npm run validate

# O manualmente:
node scripts/validate-prisma-schema.js
```

3. NO hacer push sin verificar el build local

Antes de hacer push a GitHub:

```
cd app
npm run build

# Si el build falla localmente, NO hacer push
# Si pasa, entonces push es seguro
```

✓ SIEMPRE HACER

1. Usar ubicaciones predeterminadas de Prisma

```
generator client {  
  provider = "prisma-client-js"  
  binaryTargets = ["native", "linux-musl-arm64-openssl-3.0.x"]  
}
```

2. Regenerar Prisma Client después de cambios al schema

```
npx prisma generate  
npx prisma db push # o migrate deploy en producción
```

3. Verificar tipos de TypeScript

```
npx tsc --noEmit
```

4. Usar el script de validación

```
npm run validate
```

🔧 Scripts Disponibles

Validación Completa

```
npm run validate
```

Ejecuta:

- ✓ Validación de schema.prisma
- ✓ Verificación de TypeScript
- ✓ Checks de Dockerfile
- ✓ Verificación de variables de entorno

Validación Solo Prisma

```
node scripts/validate-prisma-schema.js
```

Build de Prueba

```
npm run build
```



Protección Automática con Git Hooks

Se ha configurado un **pre-commit hook** que automáticamente ejecuta validaciones antes de cada commit.

Si las validaciones fallan, el commit se cancela y debes corregir los errores.

Instalar Husky (si no está instalado)

```
cd app
npx husky-init && npm install
npm run prepare
```

Verificar que el hook está activo

```
ls -la .husky/pre-commit
```



Flujo de Trabajo Seguro

Antes de Modificar schema.prisma

1. Haz backup del schema actual:

```
bash
cp prisma/schema.prisma prisma/schema.prisma.backup
```

Después de Modificar schema.prisma

1. **Validar schema:**

```
bash
npm run validate
```

2. **Regenerar Prisma Client:**

```
bash
npx prisma generate
```

3. **Verificar TypeScript:**

```
bash
npx tsc --noEmit
```

4. **Build de prueba:**

```
bash
npm run build
```

5. **Si todo pasa, hacer commit:**

```
bash
git add -A
git commit -m "Update: Descripción del cambio"
git push origin main
```

Checklist Pre-Commit

Antes de cada commit que afecte Prisma:

- [] ☒ Schema.prisma NO tiene rutas absolutas
- [] ☒ Schema.prisma NO tiene línea `output`
- [] ☒ Prisma Client regenerado (`npx prisma generate`)
- [] ☒ TypeScript sin errores (`npx tsc --noEmit`)
- [] ☒ Build exitoso (`npm run build`)
- [] ☒ Script de validación pasa (`npm run validate`)

Configuración Correcta del Generador

```
generator client {
  provider = "prisma-client-js"
  binaryTargets = ["native", "linux-musl-arm64-openssl-3.0.x"]
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}
```

Elementos clave:

- ☒ `provider = "prisma-client-js"`
- ☒ `binaryTargets` incluye target para Docker Alpine
- ☒ NO incluye `output`
- ☒ `url` usa `env("DATABASE_URL")`

Si el Build Falla en Coolify

Diagnóstico Rápido

1. **Revisar logs del build** en Coolify

2. **Buscar errores de Prisma:**

```
error TS2305: Module '"@prisma/client"' has no exported member
```

3. **Verificar schema.prisma localmente:**

```
bash
```

```
node scripts/validate-prisma-schema.js
```

4. **Si encuentra errores:**

- Corregir schema.prisma
- Regenerar cliente: `npx prisma generate`
- Build local: `npm run build`
- Push: `git push origin main`
- Redeploy en Coolify

Referencias

- [Prisma Client Generation](https://www.prisma.io/docs/concepts/components/prisma-client/working-with-prismaclient/generating-prisma-client) (https://www.prisma.io/docs/concepts/components/prisma-client/working-with-prismaclient/generating-prisma-client)
 - [Docker Best Practices for Prisma](https://www.prisma.io/docs/guides/deployment/deployment-guides/deploying-to-docker) (https://www.prisma.io/docs/guides/deployment/deployment-guides/deploying-to-docker)
 - [Next.js + Prisma](https://www.prisma.io/docs/guides/other/troubleshooting-orm/help-articles/nextjs-prisma-client-monorepo) (https://www.prisma.io/docs/guides/other/troubleshooting-orm/help-articles/nextjs-prisma-client-monorepo)
-

Fecha de Creación: 13 de octubre, 2025

Última Actualización: 13 de octubre, 2025

Versión: 1.0