



# Sistema de Migraciones Automáticas

---

## Descripción General

---

CitaPlanner está configurado para **ejecutar migraciones de Prisma automáticamente** en cada despliegue. Esto significa que no necesitas ejecutar comandos manualmente después de hacer deploy - el sistema se encarga de todo.

## ¿Cómo Funciona?

---

### Proceso de Inicialización Automática

Cada vez que se despliega la aplicación en Easypanel, el script `docker-entrypoint.sh` ejecuta automáticamente:

- 1. Validación de Conexión a Base de Datos**
  - Verifica que `DATABASE_URL` esté configurada correctamente
  - Prueba la conectividad con PostgreSQL usando múltiples métodos
  - Valida que todos los componentes de la URL sean correctos
- 2. Aplicación de Migraciones** ( `prisma migrate deploy` )
  - Verifica el estado actual de las migraciones
  - Aplica todas las migraciones pendientes en orden
  - Registra el estado final de las migraciones
  - **Si falla:** El contenedor no inicia y muestra logs detallados del error
- 3. Generación del Cliente Prisma** ( `prisma generate` )
  - Genera el cliente TypeScript de Prisma
  - Verifica que el cliente se haya creado correctamente
  - **Si falla:** El contenedor no inicia
- 4. Seed de Datos** (solo si la BD está vacía)
  - Verifica si la tabla `users` tiene datos
  - Si está vacía, ejecuta el seed automáticamente
  - Crea datos de ejemplo para desarrollo/testing
- 5. Configuración de Master Admin** (idempotente)
  - Configura el password del Master Admin si no existe
  - Es seguro ejecutarlo múltiples veces
- 6. Inicio de la Aplicación**
  - Solo si todos los pasos anteriores fueron exitosos
  - Inicia el servidor Next.js en modo standalone

## Ventajas del Sistema Automático

---

- ✓ **Sin Intervención Manual:** No necesitas SSH al servidor ni ejecutar comandos
- ✓ **Despliegues Más Rápidos:** Todo sucede automáticamente al hacer deploy
- ✓ **Menos Errores:** Elimina el riesgo de olvidar ejecutar migraciones
- ✓ **Logs Detallados:** Si algo falla, obtienes información clara del problema

- ✓ **Seguridad:** Las migraciones se aplican antes de que la app esté disponible
- ✓ **Idempotencia:** Puedes redespargar sin problemas - solo aplica lo necesario

## Flujo de Trabajo para Desarrolladores

### Agregar Nuevas Migraciones

#### 1. Desarrolla tu cambio en el schema:

```
bash
# Edita app/prisma/schema.prisma
```

#### 2. Crea la migración localmente:

```
bash
cd app
npx prisma migrate dev --name descripcion_del_cambio
```

#### 3. Commit y push:

```
bash
git add app/prisma/migrations/
git add app/prisma/schema.prisma
git commit -m "feat: agregar nueva funcionalidad X"
git push origin main
```

#### 4. Deploy en Easypanel:

- Ve a tu proyecto en Easypanel
- Click en "Deploy" o espera el auto-deploy
- **Las migraciones se aplicarán automáticamente** ✨

### Verificar el Estado de las Migraciones

Después del deploy, puedes ver los logs en Easypanel:

🔄 APLICANDO MIGRACIONES DE BASE DE DATOS

Verificando migraciones pendientes...  
[Salida de prisma migrate status]

Aplicando migraciones con `prisma migrate deploy`...  
[Salida de prisma migrate deploy]

✓ Migraciones aplicadas correctamente

Estado final de migraciones:  
[Estado actualizado]

## Manejo de Errores

### Si las Migraciones Fallan

El contenedor **NO iniciará** y verás logs detallados:

## ✖ ERROR AL APLICAR MIGRACIONES

Las migraciones de Prisma fallaron. Esto puede deberse a:

1. Conflictos en el esquema de la base de datos
2. Migraciones incompatibles con el estado actual de la BD
3. Errores de sintaxis SQL en los archivos de migración



Logs disponibles en:

- /tmp/migrate-status.log - Estado de migraciones
- /tmp/migrate-deploy.log - Salida de migrate deploy



Soluciones posibles:

1. Revisa los logs arriba para identificar el error específico
2. Verifica que todas las migraciones estén en el repositorio
3. Si es necesario, ejecuta 'prisma migrate resolve' manualmente
4. Considera hacer un backup de la BD antes de resolver

## Soluciones Comunes

### Error: “Migration X has already been applied”

```
# Marca la migración como resuelta
npx prisma migrate resolve --applied "nombre_de_la_migracion"
```

### Error: “Migration X failed to apply”

```
# Si la migración falló parcialmente
npx prisma migrate resolve --rolled-back "nombre_de_la_migracion"
# Luego vuelve a desplegar
```

### Error: “Database schema is not in sync”

```
# Verifica el estado
npx prisma migrate status

# Si es necesario, resetea (⚠ CUIDADO: borra datos)
npx prisma migrate reset
```

## Archivos Importantes

### docker-entrypoint.sh

Script principal que orquesta todo el proceso de inicialización:

- Ubicación: /docker-entrypoint.sh
- Funciones clave:
  - run\_migrations() : Aplica migraciones
  - generate\_prisma\_client() : Genera el cliente
  - run\_seed() : Ejecuta seed si es necesario
  - configure\_master\_password() : Configura master admin

## Migraciones

- Ubicación: app/prisma/migrations/

- Formato: `YYYYMMDDHHMMSS_nombre_descriptivo/migration.sql`
- Cada carpeta contiene:
- `migration.sql` : El SQL que se ejecutará
- Metadatos de Prisma

## Schema

- Ubicación: `app/prisma/schema.prisma`
- Define la estructura de la base de datos
- Fuente de verdad para las migraciones

## Mejores Prácticas

---



### DO (Hacer)

- **Siempre crea migraciones localmente** con `prisma migrate dev`
- **Commit las migraciones** junto con los cambios de código
- **Usa nombres descriptivos** para las migraciones
- **Prueba localmente** antes de hacer push
- **Revisa los logs** después de cada deploy
- **Haz backups** antes de migraciones complejas



### DON'T (No Hacer)

- **No edites migraciones existentes** que ya están en producción
- **No uses** `db push` en producción (solo desarrollo)
- **No hagas** `migrate reset` en producción sin backup
- **No ignores errores** de migración - investiga la causa
- **No hagas cambios manuales** en la BD de producción

## Comandos Útiles

---

### Ver Estado de Migraciones

```
cd app
npx prisma migrate status
```

### Aplicar Migraciones Manualmente (si es necesario)

```
cd app
npx prisma migrate deploy
```

### Generar Cliente Prisma

```
cd app
npx prisma generate
```

## Ver Schema de la Base de Datos



```
cd app
npx prisma db pull # Trae el schema actual de la BD
```

## Crear Nueva Migración

```
cd app
npx prisma migrate dev --name nombre_descriptivo
```

## Monitoreo y Debugging

### Ver Logs en Easypanel

1. Ve a tu proyecto en Easypanel
2. Click en “Logs” o “Console”
3. Busca las secciones:
  -  APLICANDO MIGRACIONES DE BASE DE DATOS
  -  GENERANDO CLIENTE PRISMA

### Logs Disponibles en el Contenedor

Si necesitas acceder al contenedor:

```
# Ver logs de migraciones
cat /tmp/migrate-status.log
cat /tmp/migrate-deploy.log

# Ver logs de generación de cliente
cat /tmp/prisma-generate.log
```

## Preguntas Frecuentes

### ¿Qué pasa si redespliego sin cambios?

- Las migraciones ya aplicadas se omiten automáticamente
- Solo se aplican las nuevas migraciones pendientes
- Es completamente seguro redesplegar

### ¿Puedo desactivar las migraciones automáticas?

- No es recomendado, pero podrías modificar `docker-entrypoint.sh`
- Mejor solución: asegúrate de que tus migraciones sean correctas antes de hacer push

### ¿Qué pasa con los datos existentes?

- Las migraciones respetan los datos existentes
- Si una migración requiere cambios destructivos, Prisma te lo advertirá
- Siempre haz backups antes de migraciones complejas

### ¿Cómo revierto una migración?

- Crea una nueva migración que revierta los cambios
- No intentes eliminar migraciones ya aplicadas

- Usa `prisma migrate diff` para ver diferencias

## Soporte

---

Si encuentras problemas con las migraciones automáticas:

1. **Revisa los logs** en Easypanel
  2. **Verifica el estado** con `prisma migrate status`
  3. **Consulta la documentación** de Prisma: <https://www.prisma.io/docs/concepts/components/prisma-migrate>
  4. **Revisa este documento** para soluciones comunes
- 

**Última actualización:** Octubre 2025

**Versión del sistema:** CitaPlanner v1.0+