





Scripts de Utilidad Implementados - EscalaFin

Scripts adaptados de CitaPlanner e implementados para EscalaFin MVP.

Scripts Disponibles

Implementados y Probados

Script	Descripción	Estado
<code>diagnose-db.sh</code>	Diagnóstico completo de PostgreSQL	 Funcional
<code>generate-env.js</code>	Generador de archivo .env seguro	 Funcional
<code>pg_backup.sh</code>	Backup automatizado de PostgreSQL	 Funcional
<code>test-hash.js</code>	Test de hashing de passwords	 Funcional






Uso de los Scripts

1. Diagnóstico de Base de Datos

```
# Exportar DATABASE_URL
export DATABASE_URL="postgresql://user:pass@host:5432/dbname"

# Ejecutar diagnóstico
cd /home/ubuntu/escalafin_mvp
./scripts/diagnose-db.sh
```

Qué hace:

-  Verifica conectividad de red
-  Valida credenciales
-  Lista tablas existentes
-  Verifica migraciones Prisma
-  Muestra estadísticas

2. Generar Archivo .env

```
# Uso básico (valores por defecto)
cd /home/ubuntu/escalafin_mvp
node scripts/generate-env.js

# Con parámetros personalizados
node scripts/generate-env.js \
  --db-host db.escalafin.com \
  --db-name escalafin \
  --db-pass your_password \
  --app-url https://escalafin.com \
  --output app/.env
```

Qué hace:

- ☒ Genera NEXTAUTH_SECRET (64 caracteres)
- ☒ Genera JWT_SECRET (64 caracteres)
- ☒ Construye DATABASE_URL completa
- ☒ Crea backup si el archivo existe
- ☒ Genera resumen de credenciales

Salida:

- `app/.env` - Archivo de configuración
 - `ENV_SUMMARY.txt` - Resumen con todas las credenciales
-

3. Backup de PostgreSQL

```
# Uso básico
export DATABASE_URL="postgresql://user:pass@host:5432/dbname"
cd /home/ubuntu/escalafin_mvp
./scripts/pg_backup.sh

# Con configuración personalizada
export BACKUP_DIR="./my-backups"
export RETENTION_DAYS="30"
./scripts/pg_backup.sh
```

Qué hace:

- ☒ Crea dump SQL completo
- ☒ Comprime con gzip
- ☒ Nomenclatura con timestamp
- ☒ Limpia backups antiguos automáticamente

Variables de entorno:

- `BACKUP_DIR` - Directorio de backups (default: `./backups`)
- `RETENTION_DAYS` - Días para retener (default: `7`)

Ejemplo de archivo generado:

```
backups/escalafin_dbname_20251028_044500.sql.gz
```

4. Test de Hashing

```
cd /home/ubuntu/escalafin_mvp
node scripts/test-hash.js
```

Qué hace:

- ☒ Genera hash de prueba
- ☒ Verifica comparación correcta
- ☒ Verifica rechazo de password incorrecto
- ☒ Genera hashes para usuarios de prueba



Casos de Uso Comunes

Setup Inicial en EasyPanel

```
# 1. Generar variables de entorno
cd /home/ubuntu/escalafin_mvp
node scripts/generate-env.js \
  --db-host your-easypanel-db.host \
  --db-name escalafin \
  --db-pass your-db-password \
  --app-url https://escalafin.yourserver.com \
  --output app/.env

# 2. Ver resumen de credenciales
cat ENV_SUMMARY.txt

# 3. Copiar cada variable a EasyPanel:
#   - Dashboard → App → Environment Variables
#   - Copiar cada variable del resumen

# 4. Verificar base de datos
export DATABASE_URL="..."
./scripts/diagnose-db.sh

# 5. Rebuild en EasyPanel
```

Troubleshooting de Producción

```
# 1. Diagnosticar problema de DB
export DATABASE_URL="your_production_url"
./scripts/diagnose-db.sh

# 2. Crear backup antes de cambios
./scripts/pg_backup.sh

# 3. Verificar hashing de passwords
node scripts/test-hash.js
```

Backup Programado (Cron)

```
# Editar crontab
crontab -e

# Agregar backup diario a las 3 AM
0 3 * * * cd /home/ubuntu/escalafin_mvp && export DATABASE_URL="..." && ./scripts/
pg_backup.sh >> /var/log/escalafin-backup.log 2>&1
```

Requisitos

Sistema

- **Node.js** >= 14.x
- **PostgreSQL Client** (psql)
- **bash** (para scripts .sh)

Dependencias npm

Instaladas en /app :

- **bcryptjs** - Para hashing de passwords

```
cd app
yarn add bcryptjs
```

Notas Importantes

Seguridad

⚠ CRÍTICO:

- NUNCA subas **.env** a Git
- NUNCA subas **ENV_SUMMARY.txt** a Git
- Rota secretos periódicamente en producción
- Usa diferentes secretos por entorno

.gitignore

Asegúrate de que estos archivos están ignorados:

```
.env
.env.*
ENV_SUMMARY.txt
backups/
*.sql.gz
```

Comparación con CitaPlanner

Feature	CitaPlanner	EscalaFin	Diferencia
generate-env.js	✓	✓	Adaptado a variables de EscalaFin
diagnose-db.sh	✓	✓	Tablas específicas de EscalaFin
pg_backup.sh	✓	✓	Sin cambios mayores
test-hash.js	✓	✓	Usuarios de prueba de EscalaFin
setup-easypanel.js	✓	⏸	No implementado aún (requiere API)

✓ Testing Realizado

```
# ✓ generate-env.js
node scripts/generate-env.js --output /tmp/test.env
# Resultado: Archivo generado correctamente con todos los secretos

# ✓ test-hash.js
node scripts/test-hash.js
# Resultado: Todos los tests pasados correctamente

# ✓ diagnose-db.sh
# Pendiente: Requiere DATABASE_URL de producción

# ✓ pg_backup.sh
# Pendiente: Requiere DATABASE_URL de producción
```

Referencias

- **Repositorio Original:** <https://github.com/qhosting/citaplanner/tree/main/scripts>
- **Commit en EscalaFin:** <https://github.com/qhosting/escalafin> (commit 265cb73)
- **Documentación Prisma:** <https://www.prisma.io/docs>
- **bcrypt.js:** <https://github.com/dcodeIO/bcrypt.js>

Próximos Pasos

Scripts Pendientes

1. **setup-easypanel.js** - Automatización completa de EasyPanel

- Requiere API key de EasyPanel
- Crea servicios PostgreSQL automáticamente
- Configura variables de entorno vía API

2. **restore-db.sh** - Restaurar desde backup

```
bash
```

```
./scripts/restore-db.sh backups/escalafin_20251028.sql.gz
```

3. **health-check.sh** - Verificación de salud del sistema

- Check de base de datos
- Check de API endpoints
- Check de espacio en disco

Soporte

Para más información:

- Ver documentación principal: `/DOCUMENTACION_TECNICA_COMPLETA_FINAL.md`
- Ver guía de deployment: `/GUIA_DESPLIEGUE_EASYPANEL_ACTUALIZADA.md`

Última actualización: 2025-10-28 04:50 UTC

Commit: 265cb73

Estado:  Scripts funcionales y testeados