✅ PERSISTENCIA Y RESPALDOS IMPLEMENTADOS

Mueblería La Económica

Fecha: 9 de octubre de 2025
Estado: ✅ IMPLEMENTADO Y LISTO

🎯 Objetivo Alcanzado

Se ha implementado un **sistema completo de persistencia de datos y respaldos automáticos** que garantiza:

- ✅ **Los datos NO se pierden en cada deploy**
- ✅ **Respaldos automáticos configurables**
- ✅ **Seed inteligente que NO elimina datos existentes**
- ✅ **Sistema de restauración de emergencia**
- ✅ **Validación automática de configuración**

📦 Archivos Creados

Scripts de Operación

1. `validate-persistence.sh` ✅
   - Verifica toda la configuración de persistencia
   - Muestra estado de volúmenes y base de datos
   - Valida backups y scripts

2. `run-seed-safe.sh` ✅
   - Ejecuta el seed inteligente en producción
   - NO elimina datos existentes
   - Solo crea usuarios esenciales si faltan

3. `backup-database.sh` ✅
   - Crea respaldos manuales
   - Compresión automática (GZIP)
   - Rotación de respaldos (mantiene últimos 30)

4. `restore-database.sh` ✅
   - Restaura desde cualquier respaldo
   - Interfaz interactiva
   - Confirmación de seguridad

5. `cron-backup.sh` ✅
   - Configura respaldos automáticos

- 4 opciones de frecuencia predefinidas
- Opción personalizada

Código

1. `app/scripts/seed-safe.ts` ✅
   - Seed inteligente para producción
   - Solo inserta si no existe
   - Preserva TODOS los datos existentes

Documentación

1. `PERSISTENCIA-Y-RESPALDOS.md` ✅ (+ PDF)
   - Documentación técnica completa
   - Guía de solución de problemas
   - Explicación detallada de cada componente
2. `GUIA-RAPIDA-PERSISTENCIA.md` ✅ (+ PDF)
   - Comandos esenciales
   - Flujos de trabajo recomendados
   - FAQ

Configuración

1. `.gitignore` ✅ (actualizado)
   - Excluye respaldos del repositorio
   - Excluye logs y archivos temporales
   - Mantiene el repositorio limpio

🔍 Validación de la Imagen

Basándose en la imagen que compartiste (`laeco.jpg`), confirmo:

✅ Configuración Correcta Detectada

```
VOLUME:
  Nombre: laeconomica-postgres-data
  Path: /var/lib/postgresql/data
  Tipo: Volume
```

Estado: ✅ **CORRECTO**

- El volumen está configurado correctamente
- Los datos se almacenan en un volumen persistente de Docker
- Los datos NO se perderán en cada deploy

📊 Tu Configuración

En tu `docker-compose.yml` :

```
postgres:
  volumes:
    - postgres_data:/var/lib/postgresql/data  # ✅ CORRECTO
```

Esto garantiza que **todos los datos de PostgreSQL persisten** entre reinicios y deploys.

🚀 Primeros Pasos

1. Validar Configuración

```
cd /home/ubuntu/muebleria_la_economica
./validate-persistence.sh
```

Resultado esperado:
- ✅ Configuración de volumen verificada
- ✅ Scripts disponibles
- ✅ Todo listo para usar

2. Ejecutar Seed Seguro (Primera Vez)

```
./run-seed-safe.sh
```

Esto creará:
- 4 usuarios esenciales (admin, gestor, cobrador, reportes)
- 2 plantillas de ticket

NO afectará:
- ❌ Clientes existentes
- ❌ Pagos existentes
- ❌ Datos de producción

3. Crear Primer Respaldo

```
./backup-database.sh
```

Resultado: Respaldo en `backups/backup_YYYY-MM-DD_HH-MM-SS.sql.gz`

4. Configurar Respaldos Automáticos

```
./cron-backup.sh
```

Recomendación: Opción 1 (Diario a las 2:00 AM) para empezar.

📖 Documentos de Referencia

Para Uso Diario

📄 **GUIA-RAPIDA-PERSISTENCIA.md**
- Comandos esenciales
- Flujos de trabajo
- FAQ

Para Referencia Técnica

📄 **PERSISTENCIA-Y-RESPALDOS.md**
- Documentación completa
- Solución de problemas
- Arquitectura del sistema

Para Ver Usuarios Creados

📄 **list-seed-resources.sh**
- Lista de usuarios con contraseñas
- Roles y permisos

🔐 Usuarios Esenciales Creados

Usuario	Email	Contraseña	Rol
Admin	`admin@economica.local`	`admin123`	admin
Gestor	`gestor@economica.local`	`gestor123`	gestor_cobranza
Cobrador	`cobrador@economica.local`	`cobrador123`	cobrador
Reportes	`reportes@economica.local`	`reportes123`	reporte_cobranza

⚠️ **IMPORTANTE**: Cambia estas contraseñas después del primer login.

🎯 Diferencias Clave: Seed Original vs. Seed Seguro

❌ Seed Original (`scripts/seed.ts`)

```
// PELIGRO: Elimina TODOS los datos
await prisma.pago.deleteMany();
await prisma.cliente.deleteMany();
await prisma.user.deleteMany();

// Luego crea datos de demo
```

Uso: Solo en desarrollo local o para resetear base de datos.

✅ Seed Seguro (`scripts/seed-safe.ts`)

```
// Verifica si hay datos
const userCount = await prisma.user.count();

// Solo crea si no existen
await prisma.user.upsert({
  where: { email: 'admin@economica.local' },
  update: {},  // No actualiza si existe
  create: { ... }  // Solo crea si no existe
});
```

Uso: Producción, deploys, cualquier momento.

💡 Casos de Uso Comunes

Caso 1: Primer Deploy en Producción

```
# 1. Validar
./validate-persistence.sh

# 2. Iniciar servicios
docker compose up -d

# 3. Crear usuarios esenciales
./run-seed-safe.sh

# 4. Primer respaldo
./backup-database.sh

# 5. Configurar respaldos automáticos
./cron-backup.sh
```

Caso 2: Deploy Subsecuente

```
# 1. Respaldo preventivo
./backup-database.sh

# 2. Deploy
docker compose up -d --build

# 3. Verificar (opcional)
./validate-persistence.sh
```

Caso 3: Algo Salió Mal

```
# 1. Detener servicios
docker compose down

# 2. Restaurar
./restore-database.sh
# Seleccionar respaldo reciente

# 3. Reiniciar
docker compose up -d
```

Caso 4: Migración de Base de Datos

```
# 1. Respaldo crítico
./backup-database.sh

# 2. Ejecutar migración de Prisma
docker compose exec app npx prisma migrate deploy

# 3. Verificar
./validate-persistence.sh
```

⚙️ Configuración de Respaldos Recomendada

Por Tamaño de Operación

Tipo de Operación	Frecuencia Recomendada	Comando
Pequeña (< 50 transacciones/día)	Diario (2:00 AM)	`./cron-backup.sh` → Opción 1
Mediana (50-200 transacciones/día)	Cada 12 horas	`./cron-backup.sh` → Opción 2
Grande (> 200 transacciones/día)	Cada 6 horas	`./cron-backup.sh` → Opción 3
Crítica	Cada 3 horas	`./cron-backup.sh` → Opción 4 → `0 */3 * * *`

🗄️ Estructura de Directorios

```
muebleria_la_economica/
├── app/
│   └── scripts/
│       ├── seed.ts              # ❌ NO usar en producción
│       └── seed-safe.ts         # ✅ Usar en producción
├── backups/                     # Se crea automáticamente
│   ├── backup_2025-10-09_*.sql.gz
│   ├── latest.sql.gz            # Enlace al último
│   └── backup.log               # Log de respaldos automáticos
├── backup-database.sh           # Crear respaldo
├── restore-database.sh          # Restaurar respaldo
├── run-seed-safe.sh             # Ejecutar seed seguro
├── cron-backup.sh               # Configurar automáticos
├── validate-persistence.sh      # Validar configuración
├── PERSISTENCIA-Y-RESPALDOS.md  # Doc completa
└── GUIA-RAPIDA-PERSISTENCIA.md  # Guía rápida
```

📊 Estado del Repositorio GitHub

✅ Todos los cambios subidos exitosamente

```
Commit: ✨ Sistema completo de persistencia y respaldos
Branch: main
Archivos: 11 archivos creados/modificados
Líneas: +1443 -19
```

Ver en GitHub: https://github.com/qhosting/muebleria-la-economica

---

✅ Checklist de Implementación

- [x] Volumen persistente configurado en `docker-compose.yml`
- [x] Seed inteligente creado (`seed-safe.ts`)
- [x] Script de respaldo manual (`backup-database.sh`)
- [x] Script de restauración (`restore-database.sh`)
- [x] Script de seed seguro (`run-seed-safe.sh`)
- [x] Configuración de cron (`cron-backup.sh`)
- [x] Validación de persistencia (`validate-persistence.sh`)
- [x] Documentación completa
- [x] Guía rápida de uso
- [x] `.gitignore` actualizado
- [x] Todo subido a GitHub

🎓 Para Aprender Más

Comandos Docker para Volúmenes

```
# Ver todos los volúmenes
docker volume ls

# Inspeccionar un volumen
docker volume inspect muebleria_la_economica_postgres_data

# Ver uso de espacio
docker system df -v
```

Verificar Datos en la Base de Datos

```
# Conectarse a PostgreSQL
docker compose exec postgres psql -U postgres -d muebleria_db

# Contar registros
SELECT 'Usuarios' as tabla, COUNT(*) FROM "User";
SELECT 'Clientes' as tabla, COUNT(*) FROM "Cliente";
SELECT 'Pagos' as tabla, COUNT(*) FROM "Pago";
```

🆘 Contacto de Emergencia

Si necesitas ayuda:

1. Revisa `GUIA-RAPIDA-PERSISTENCIA.md` para problemas comunes
2. Ejecuta `./validate-persistence.sh` para diagnóstico
3. Verifica logs: `docker compose logs -f postgres`
4. Revisa `PERSISTENCIA-Y-RESPALDOS.md` para solución de problemas

🎉 Resultado Final

Lo Que Tenías Antes

- ❌ Datos se perdían en cada deploy
- ❌ Seed eliminaba información de producción
- ❌ Sin sistema de respaldos
- ❌ Sin forma de recuperar datos

Lo Que Tienes Ahora

- ✅ Datos persisten en volúmenes de Docker
- ✅ Seed inteligente preserva datos existentes
- ✅ Respaldos automáticos configurables
- ✅ Sistema de restauración de emergencia
- ✅ Validación automática de configuración
- ✅ Documentación completa

- ✅ Scripts listos para usar

🚀 Próximos Pasos Sugeridos

1. **Ahora Mismo**:
   `bash`
   `./validate-persistence.sh`
2. **Antes del Próximo Deploy**:
   `bash`
   `./backup-database.sh`
   `./cron-backup.sh`
3. **En Tu Próximo Deploy**:
   - Los datos persistirán automáticamente
   - Solo ejecuta `./run-seed-safe.sh` si necesitas usuarios esenciales
   - Verifica con `./validate-persistence.sh` después del deploy
4. **Semanalmente**:
   - Revisa `ls -lh backups/` para ver los respaldos
   - Prueba restaurar un respaldo en desarrollo
   - Verifica `tail -f backups/backup.log`

¡Tu sistema de persistencia está completo y listo para producción! 🎉

Documentado por: DeepAgent
Proyecto: Mueblería La Económica
Versión: 1.0.0
Fecha: 9 de octubre de 2025
