Resumen de Merge - PR #96

Estado del Merge

- **PR Number:** #96
- **Título:** Fix: Validación completa y manejo de categoryId en servicios
- **Branch:** `fix/service-create-edit-error` → `main`
- **Commit SHA:** `c9ff9c718a3939f74a87ae253e0c12c56d3a4c1c`
- **Método de Merge:** Squash Merge
- **Fecha:** 2025-10-13 17:30:51 UTC
- **Estado:** Mergeado exitosamente
- **Rama feature:** Eliminada automáticamente

Problema Identificado

Después de mergear los PRs #93, #94 y #95, se detectaron errores críticos al **crear y editar servicios** en `dashboard/services`:

Causas Raíz:

1. **Falta de validación en PUT**: El endpoint `PUT /api/services/[id]` no tenía validación de datos, a diferencia del POST
2. **Manejo incorrecto de categoryId**: Cuando `categoryId` era una cadena vacía `''`, causaba errores en Prisma
3. **Logging insuficiente**: Difícil debugging en operaciones de actualización
4. **Inconsistencia**: Validación diferente entre POST y PUT

Archivos Modificados

1. app/app/api/services/[id]/route.ts (Endpoint PUT)

- **Cambios:** +50 líneas, -3 líneas
- **Modificaciones:**
  - Agregada validación completa de datos (name, price, duration)
  - Validación condicional: solo valida campos presentes en el body
  - Limpieza de `categoryId`: convierte `''` a `null` antes de enviar a Prisma
  - Logging detallado mejorado con información de debug
  - Mensajes de error descriptivos en español
  - Stack trace completo en logs para debugging

Ejemplo de validación implementada:

```
// Validación de datos si están presentes
if (body.name !== undefined && (!body.name || body.name.trim() === '')) {
  console.log('[Services API] Validation error: name cannot be empty');
  return NextResponse.json({
    success: false,
    error: 'El nombre del servicio no puede estar vacío'
  }, { status: 400 });
}

// Limpiar categoryId si es cadena vacía
const cleanedData = {
```

```
  ...body,
  categoryId: body.categoryId === '' ? null : body.categoryId,
};
```

2. app/app/api/services/route.ts (Endpoint POST)

- **Cambios:** +7 líneas, -1 línea
- **Modificaciones:**
  - Agregada limpieza de `categoryId` antes de crear servicio
  - Consistencia con el endpoint PUT
  - Prevención de errores de Prisma con cadenas vacías

Código implementado:

```
// Limpiar categoryId si es cadena vacía
const cleanedData = {
  ...body,
  categoryId: body.categoryId === '' ? null : body.categoryId,
};

const newService = await serviceManager.createService({
  ...cleanedData,
  tenantId,
});
```

3. app/components/modals/service-modal.tsx

- **Cambios:** +3 líneas, -3 líneas
- **Modificaciones:**
  - Mejorado manejo de `categoryId` en `onSubmit`
  - Trim de `name` y `description` para evitar espacios en blanco
  - Conversión explícita de `categoryId` vacío a `null`
  - Validación más robusta antes de enviar datos

Código mejorado:

```
const serviceData = {
  name: data.name.trim(),
  description: data.description?.trim() || null,
  categoryId: data.categoryId && data.categoryId !== '' ? data.categoryId : null,
  // ...
}
```

Beneficios

- **Validación consistente** entre POST y PUT
- **Prevención de errores** de Prisma con datos inválidos
- **Mejor UX** con mensajes de error claros en español
- **Debugging mejorado** con logs detallados
- **Manejo robusto** de campos opcionales (categoryId)
- **Sin breaking changes**
- **Sin migraciones** de base de datos requeridas

Testing Recomendado

Crear Servicio:

1. Crear servicio sin categoría (categoryId vacío)
2. Crear servicio con categoría válida
3. Validar campos requeridos (name, price, duration)
4. Verificar mensajes de error descriptivos

Editar Servicio:

1. Editar servicio y cambiar categoría
2. Editar servicio y remover categoría (dejar vacío)
3. Editar solo algunos campos (validación parcial)
4. Verificar que los cambios se persisten correctamente

Casos Edge:

1. Nombre con espacios en blanco
2. Precio negativo o inválido
3. Duración menor a 5 minutos
4. categoryId como cadena vacía

Estadísticas del Merge

- **Total de archivos modificados:** 3
- **Líneas agregadas:** +60
- **Líneas eliminadas:** -7
- **Cambios netos:** +53 líneas
- **Breaking changes:** No
- **Migraciones DB:** No requeridas
- **Prioridad:** Alta (Bug crítico en producción)

Relacionado

- **PRs relacionados:** #93, #94, #95
- **Fixes:** Error al crear/editar servicios en dashboard/services
- **Mejora:** Validación y manejo de datos en endpoints de servicios

---

Próximos Pasos para Deployment

1. Deployment Automático en Easypanel

El deployment se activará automáticamente al detectar el nuevo commit en `main`.

2. Verificación Post-Deployment

Después del deployment, verificar:

a) Creación de Servicios:

```
# Verificar que se pueden crear servicios sin categoría
POST /api/services
{
  "name": "Servicio Test",
  "price": 100,
  "duration": 30,
  "categoryId": ""  // Debe convertirse a null
}
```

b) Edición de Servicios:

```
# Verificar que se pueden editar servicios
PUT /api/services/[id]
{
  "name": "Servicio Actualizado",
  "categoryId": ""  // Debe convertirse a null
}
```

c) Validaciones:

- Intentar crear servicio sin nombre → Debe mostrar error en español
- Intentar crear servicio con precio negativo → Debe mostrar error
- Intentar crear servicio con duración < 5 minutos → Debe mostrar error

3. Monitoreo de Logs

Revisar los logs de Easypanel para verificar: - Logging detallado de operaciones de servicios - Stack traces completos en caso de errores - Mensajes de validación claros

4. Testing en Producción

1. Acceder a `dashboard/services`
2. Crear un nuevo servicio sin categoría
3. Editar un servicio existente
4. Verificar que no hay errores en la consola del navegador
5. Confirmar que los datos se persisten correctamente

Notas Importantes

Cambios Seguros

- No hay breaking changes
- No se requieren migraciones de base de datos
- Compatibilidad total con código existente
- Mejoras de validación no afectan funcionalidad actual

Debugging Mejorado

El nuevo logging incluye: - Información detallada de cada operación - Stack traces completos en errores - Validación de datos antes de enviar a Prisma - Mensajes descriptivos para troubleshooting

Mensajes en Español

Todos los mensajes de error están en español para mejor UX: - “El nombre del servicio no puede estar vacío” - “El precio debe ser un valor válido mayor o igual a 0” - “La duración debe ser al menos 5 minutos” - “Error interno del servidor al actualizar el servicio”

Checklist de Verificación

- ☒ PR mergeado exitosamente
- ☒ Rama feature eliminada
- ☒ Commit verificado en main
- ☒ Repositorio local actualizado
- ☒ Documentación generada
- ☐ Deployment automático completado
- ☐ Testing post-deployment realizado
- ☐ Logs verificados en producción
- ☐ Funcionalidad de servicios confirmada

Conclusión

El PR #96 ha sido mergeado exitosamente, resolviendo los errores críticos en la creación y edición de servicios. Los cambios incluyen:

1. **Validación robusta** en ambos endpoints (POST y PUT)
2. **Manejo correcto** de categoryId vacío
3. **Logging detallado** para debugging
4. **Mensajes claros** en español para usuarios

El sistema está listo para deployment automático en Easypanel. Se recomienda verificar la funcionalidad de servicios después del deployment y monitorear los logs para confirmar el correcto funcionamiento.

Versión sugerida para próximo release: v1.3.2
Prioridad: Alta

Generado: 2025-10-13
Commit SHA: c9ff9c718a3939f74a87ae253e0c12c56d3a4c1c
Autor: qhosting admin@qhosting.net