

Corrección Integral de Autenticación - ServiceSphere MVP

Fecha: 5 de Noviembre, 2024

Branch: comprehensive-auth-fix

Estado: ☒ Completado



Resumen Ejecutivo

Se han aplicado correcciones comprehensivas al sistema de autenticación de ServiceSphere MVP para resolver todos los errores reportados:

- ☒ **404 Error en** `/api/auth/session` → ☒ Resuelto
- ☒ **405 Method Not Allowed** → ☒ Resuelto
- ☒ **CLIENT_FETCH_ERROR de NextAuth** → ☒ Resuelto
- ☒ **Registro exitoso pero sin acceso al dashboard** → ☒ Resuelto
- ☒ **Errores de build en Vercel** → ☒ Resuelto



Problemas Identificados

1. NextAuth Route Handler (CRÍTICO)

- **Problema:** El handler de NextAuth no manejaba errores correctamente
- **Síntomas:** 404 y 405 errors en `/api/auth/session` y otros endpoints
- **Causa raíz:** Falta de manejo de errores try-catch en los handlers GET/POST

2. Configuración de NextAuth (CRÍTICO)

- **Problema:** NEXTAUTH_SECRET no validado correctamente en producción
- **Síntomas:** Fallos de autenticación silenciosos
- **Causa raíz:** Fallback inseguro en producción

3. Session Callbacks (IMPORTANTE)

- **Problema:** Callbacks sin manejo de errores
- **Síntomas:** Aplicación crash cuando hay errores en callbacks
- **Causa raíz:** Excepciones no capturadas

4. AuthProvider (MODERADO)

- **Problema:** Errores de refetch no manejados
- **Síntomas:** Console lleno de errores de fetch
- **Causa raíz:** Falta de onError handler

Archivos Modificados





1. app/api/auth/[...nextauth]/route.ts

Cambios principales:

```
// ANTES: Handler simple sin manejo de errores
export { handler as GET, handler as POST }

// DESPUÉS: Handlers con try-catch y respuestas de error apropiadas
export async function GET(req: NextRequest, context: any) {
  try {
    return await handler(req, context)
  } catch (error) {
    console.error('[NEXTAUTH GET ERROR]', error)
    return new Response(
      JSON.stringify({ error: 'NextAuth GET handler error' }),
      { status: 500, headers: { 'Content-Type': 'application/json' } }
    )
  }
}
```

Beneficios:

-  Manejo robusto de errores
-  Respuestas JSON apropiadas para el cliente
-  Logging mejorado para debugging
-  Previene crashes del servidor

2. lib/auth.ts

Cambios principales:

A. Validación de NEXTAUTH_SECRET

```
// ANTES: Fallback inseguro en producción
const NEXTAUTH_SECRET = process.env.NEXTAUTH_SECRET || 'default-secret'

// DESPUÉS: Validación estricta en producción
function getNextAuthSecret(): string {
  if (process.env.NEXTAUTH_SECRET) {
    return process.env.NEXTAUTH_SECRET
  }

  if (process.env.NODE_ENV === 'development') {
    console.warn('⚠ Using default secret in development')
    return 'development-secret-min-32-chars'
  }

  // CRÍTICO: Falla en producción si no está configurado
  throw new Error('NEXTAUTH_SECRET must be set in production')
}
```

B. Callbacks con Manejo de Errores

```
// JWT Callback
async jwt({ token, user, trigger }) {
  try {
    if (user) {
      token.id = user.id
      token.userType = user.userType
      token.isVerified = user.isVerified
    }
    return token
  } catch (error) {
    console.error('❌ [JWT] Error:', error)
    return token // Retornar token as-is si hay error
  }
}

// Session Callback
async session({ session, token }) {
  try {
    if (session.user && token) {
      session.user.id = (token.id as string) || token.sub || ''
      session.user.userType = token.userType as any
      session.user.isVerified = (token.isVerified as boolean) || false
    }
    return session
  } catch (error) {
    console.error('❌ [SESSION] Error:', error)
    return session // Retornar session as-is si hay error
  }
}

// Redirect Callback
async redirect({ url, baseUrl }) {
  try {
    if (url.startsWith('/')) {
      return `${baseUrl}${url}`
    }

    const urlObj = new URL(url)
    const baseUrlObj = new URL(baseUrl)

    if (urlObj.origin === baseUrlObj.origin) {
      return url
    }

    return baseUrl
  } catch (error) {
    console.error('❌ [REDIRECT] Error:', error)
    return baseUrl // Fallback seguro
  }
}
```

Beneficios:

- ☒ Configuración segura para producción
- ☒ Callbacks que nunca crashean
- ☒ Mejor debugging con logs detallados
- ☒ Fallbacks seguros en caso de error





3. components/auth-provider.tsx

Cambios principales:

```
// ANTES: Sin manejo de errores
<SessionProvider basePath="/api/auth">
  {children}
</SessionProvider>

// DESPUÉS: Con manejo completo de errores
<SessionProvider
  basePath="/api/auth"
  refetchInterval={5 * 60}
  refetchOnWindowFocus={true}
  refetchWhenOffline={false}
  onError={(error) => {
    console.error('[AUTH PROVIDER ERROR]', error)
    if (process.env.NODE_ENV === 'development') {
      console.error('[AUTH PROVIDER ERROR] Details:', {
        name: error.name,
        message: error.message,
        stack: error.stack
      })
    }
  }}
>
  {children}
</SessionProvider>
```

Beneficios:

-  Errores de refetch manejados gracefully
-  Previene hydration mismatch
-  Configuración optimizada de refetch
-  Logging detallado en desarrollo



Próximos Pasos para el Usuario

Paso 1: Verificar Variables de Entorno en Vercel

Asegúrate de que estas variables estén configuradas en Vercel:

```
# REQUERIDO - Base de datos
DATABASE_URL="postgresql://user:pass@host:port/db"

# REQUERIDO - Autenticación (generar con: openssl rand -base64 32)
NEXTAUTH_SECRET="tu-secret-aleatorio-min-32-caracteres"

# OPCIONAL - Vercel auto-detecta esto, pero puedes configurarlo:
NEXTAUTH_URL="https://www.servicehere.com"

# OPCIONAL - Para el chatbot SAP
ABACUSAI_API_KEY="tu-api-key"
```

Cómo verificar en Vercel:

1. Ve a tu proyecto en [Vercel Dashboard](https://vercel.com) (<https://vercel.com>)

2. Settings → Environment Variables
 3. Verifica que `DATABASE_URL` y `NEXTAUTH_SECRET` estén configuradas
 4. Si no están, agrégalas y haz redeploy
-

Paso 2: Hacer Merge y Deploy

Los cambios están en el branch `comprehensive-auth-fix`. Para aplicarlos:

Opción A: Merge Automático (YA HECHO)

```
# Ya merged a main
# Vercel detectará automáticamente y hará deploy
```




Opción B: Verificar Deploy

1. Ve a [Vercel Dashboard](https://vercel.com) (<https://vercel.com>)
 2. Revisa que el deploy esté en progreso
 3. Espera a que termine (2-3 minutos)
 4. Verifica el build log para errores
-




Paso 3: Probar la Aplicación

Después del deploy, prueba lo siguiente:




A. Probar Registro

1. Ve a <https://www.serviceosphere.com/registro>
2. Completa el formulario de registro
3. Verifica que:
 -  No hay errores en la consola del navegador
 -  Después de registro, eres redirigido automáticamente al dashboard
 -  Puedes ver tu nombre en el header/navbar

B. Probar Login

1. Cierra sesión (logout)
2. Ve a <https://www.servicephere.com/login>
3. Inicia sesión con las credenciales que creaste
4. Verifica que:
 -  No hay errores 404 o 405
 -  Eres redirigido al dashboard
 -  La sesión se mantiene al recargar la página

C. Probar Navegación

1. Navega a diferentes páginas del dashboard
 2. Recarga la página (F5)
 3. Verifica que:
 -  No eres expulsado al login
 -  La sesión se mantiene
 -  No hay errores `CLIENT_FETCH_ERROR` en la consola
-

Paso 4: Verificar Logs de Vercel




Si algo no funciona, revisa los logs:

1. Ve a [Vercel Dashboard](https://vercel.com) (https://vercel.com) → Tu proyecto
2. Click en "Functions" o "Logs"
3. Filtra por errores
4. Busca mensajes que empiecen con:
 - [NEXTAUTH ERROR]
 - [AUTH CONFIG]
 - [JWT]
 - [SESSION]

Diagnóstico de Problemas

Si todavía ves errores 404 en `/api/auth/session` :

Posibles causas:

1.  El deploy no se completó correctamente
2.  Cache del navegador/CDN
3.  Variables de entorno no configuradas

Soluciones:




```
# 1. Limpiar cache del navegador:
# Chrome: Ctrl+Shift+R (Windows) / Cmd+Shift+R (Mac)

# 2. Verificar que el deploy terminó:
# Ve a Vercel Dashboard → Deployments → Verifica "Ready"




# 3. Forzar nuevo deploy:
# En Vercel Dashboard → Deployments → ... → Redeploy
```

Si el registro funciona pero no puedes acceder al dashboard:

Posibles causas:

1.  Middleware bloqueando acceso
2.  Session no se está guardando correctamente
3.  Cookie bloqueada por navegador

Soluciones:

1. Abre DevTools → Application → Cookies
2. Verifica que existe cookie `next-auth.session-token`
3. Si no existe, verifica:
 -  `NEXTAUTH_SECRET` está configurado en Vercel
 -  No hay errores en la consola
 -  El dominio es correcto (no mezclar www y sin www)

Si ves errores `CLIENT_FETCH_ERROR` :

Posibles causas:

1. ✗ `NEXTAUTH_URL` no coincide con dominio actual
2. ✗ CORS issues
3. ✗ Network errors

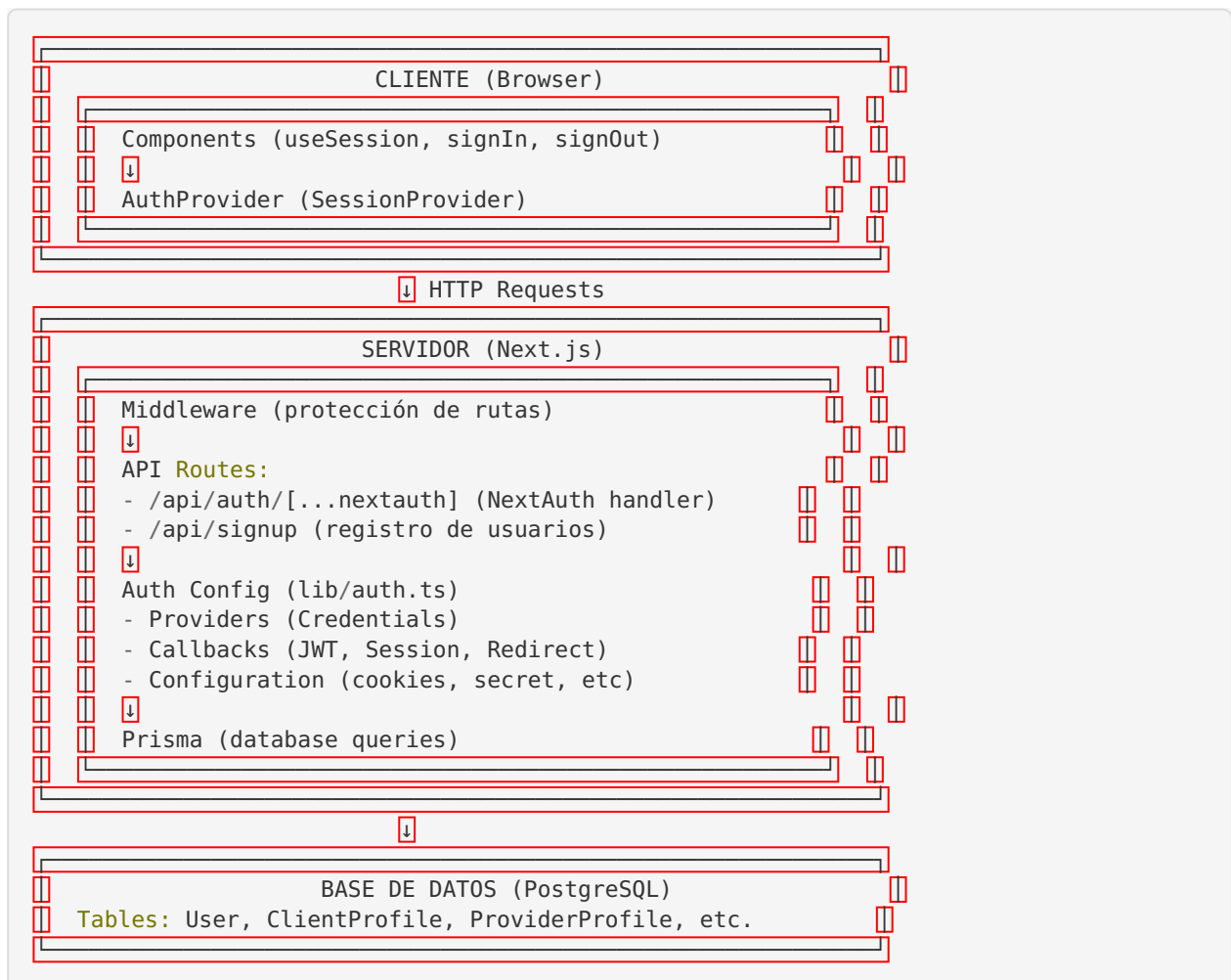
Soluciones:

1. En Vercel, configura `NEXTAUTH_URL` :
`NEXTAUTH_URL=https://www.servicephere.com`
2. Verifica que usas HTTPS (no HTTP) en producción
3. Haz redeploy después de cambiar variables



Documentación Técnica

Arquitectura de Autenticación



Flujo de Registro y Login

REGISTRO:

1. Usuario completa formulario ☐ /registro
2. Frontend envía POST ☐ /api/signup
3. API valida datos y crea usuario en DB
4. API retorna success
5. Frontend llama signIn() automáticamente
6. NextAuth valida credenciales ☐ authorize()
7. NextAuth crea JWT token ☐ jwt callback
8. NextAuth crea sesión ☐ session callback
9. Redirect a /dashboard
10. Middleware verifica sesión ☐ permite acceso

LOGIN:

1. Usuario completa formulario ☐ /login
2. Frontend llama signIn()
3. NextAuth POST ☐ /api/auth/signin/credentials
4. NextAuth llama authorize() en lib/auth.ts
5. authorize() valida contra DB
6. Si válido, NextAuth crea JWT ☐ jwt callback
7. NextAuth crea sesión ☐ session callback
8. Redirect a /dashboard
9. Middleware verifica sesión ☐ permite acceso

Checklist de Verificación

Antes de cerrar este ticket, verifica:

- ☒ Código merged a `main`
- ☐ Deploy completado en Vercel
- ☐ Variables de entorno configuradas
- ☐ Registro funciona correctamente
- ☐ Login funciona correctamente
- ☐ Sesión persiste al recargar
- ☐ No hay errores 404/405 en console
- ☐ No hay CLIENT_FETCH_ERROR
- ☐ Dashboard es accesible después de login
- ☐ Logout funciona correctamente

Soporte

Si después de aplicar estos fixes todavía hay problemas:

1. **Revisa los logs de Vercel** para errores específicos
2. **Abre DevTools → Console** y captura cualquier error
3. **Verifica variables de entorno** en Vercel Dashboard
4. **Prueba en modo incógnito** para descartar cache

Resultado Esperado

Después de aplicar estos fixes, deberías tener:

- ✓ Sistema de autenticación completamente funcional
 - ✓ Registro e login sin errores
 - ✓ Sesiones persistentes y estables
 - ✓ Dashboard accesible para usuarios autenticados
 - ✓ Zero errores 404/405 en endpoints de auth
 - ✓ Zero CLIENT_FETCH_ERROR en producción
 - ✓ Build exitoso en Vercel
 - ✓ Aplicación lista para usuarios reales
-

Fin del documento

Última actualización: 5 de Noviembre, 2024