

Solución al Problema de Autenticación de Commits en GitHub

Diagnóstico del Problema

Descripción

Vercel está fallando al desplegar el proyecto **quantpaychain-mvpro** con el siguiente error:






 **“No se encontró ninguna cuenta de GitHub que coincida con la dirección de correo electrónico del autor de la confirmación”**

Este error indica que los commits en el repositorio fueron realizados con direcciones de correo electrónico que **NO están asociadas** con ninguna cuenta de GitHub, lo que impide que Vercel verifique la autoría de los commits y, por lo tanto, bloquea el despliegue automático.

Análisis de Commits Recientes

He analizado los últimos 50 commits del repositorio y encontré **5 direcciones de correo electrónico diferentes**:

Resumen de Correos Encontrados:

| Correo Electrónico | Número de Com- mits | Estado | Nombre de Autor |
|-----------------------------|------------------------|---|---------------------------------|
| quantpay-chain@example.com | 6 commits |  PROBLEMÁTICO | QuantPay Chain Bot |
| fmengarelli@gmail.com | 6 commits |  VÁLIDO | francoMengarelli |
| ai@quantpaychain.com | 4 commits |  PROBLEMÁTICO | QuantPay AI |
| agent@abacus.ai | 4 commits |  PROBLEMÁTICO | Abacus AI Agent |
| quant-pay@quantpaychain.org | 2 commits |  PROBLEMÁTICO | QuantPay Chain Development Team |

Problema Crítico Identificado

Los **6 commits más recientes** (realizados el 24 de octubre de 2025) fueron hechos con el correo:

quantpaychain@example.com

Este es un **correo de ejemplo/placeholder** que NO está asociado con ninguna cuenta real de GitHub. **Esto es lo que está causando que Vercel falle.**

Commits Problemáticos Recientes:

```
e992925 | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | docs: Add comprehensive backend documentation
0344550 | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | docs: Add comprehensive implementation summary
f7c2f9c | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | chore(deps): Update dependencies and configuration
df23376 | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | feat(api): Implement complete REST API with Next.js 14 routes
0734728 | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | feat(backend): Implement core backend services
faafd72 | 2025-10-24 | QuantPay Chain Bot <quantpaychain@example.com> | feat(database): Complete Prisma schema for real estate tokenization
```



Soluciones Disponibles

Tienes **3 opciones** para resolver este problema. La **Solución 1** es la más rápida y recomendada si no necesitas mantener el historial exacto de commits.



Solución 1: Agregar Correos a tu Cuenta de GitHub (RECOMENDADA)

Esta es la solución más rápida y no requiere reescribir el historial de Git.

Paso 1: Agregar los Correos Problemáticos a tu Cuenta de GitHub

1. Ve a tu configuración de GitHub:

- Visita: <https://github.com/settings/emails>

2. Agrega cada correo problemático:

- Click en **"Add email address"**
- Agrega los siguientes correos uno por uno:

- `quantpaychain@example.com`
- `ai@quantpaychain.com`
- `quantpay@quantpaychain.org`
- `agent@abacus.ai`

3. Verifica cada correo:

- GitHub enviará un correo de verificación a cada dirección
- Haz click en el enlace de verificación en cada correo
- Si no tienes acceso a estos correos (como `quantpaychain@example.com`), pasa a la **Solución 2**

Paso 2: Verificar en GitHub

```
# Después de agregar los correos, verifica que estén asociados
# Visita: https://github.com/settings/emails
```

⚠ Limitación de esta Solución

- Solo funciona si tienes acceso a las direcciones de correo para verificarlas
- Para `quantpaychain@example.com`, necesitarías configurar primero el dominio `example.com` (no recomendado)
- **Si no tienes acceso a estos correos, usa la Solución 2**

✅ Solución 2: Reconfigurar Git Localmente para Futuros Commits (RECOMENDADA SI NO TIENES ACCESO A LOS CORREOS)

Esta solución asegura que **todos los commits futuros** usen el correo correcto asociado a tu cuenta de GitHub.

Paso 1: Verificar tu Correo de GitHub

Primero, confirma cuál es el correo asociado a tu cuenta de GitHub:

1. Ve a: <https://github.com/settings/emails>
2. Busca tu correo principal o el correo “noreply” de GitHub

Paso 2: Configurar Git Globalmente

Configura Git para usar tu correo correcto en **todos** los repositorios:

```
# Reemplaza con tu correo real de GitHub
git config --global user.email "fmengarelli@gmail.com"
git config --global user.name "francoMengarelli"
```

Paso 3: Configurar Git Solo para este Repositorio (Opcional)

Si prefieres usar un correo diferente solo para este proyecto:

```
# Navega al repositorio
cd /ruta/a/quantpaychain-mvpro

# Configura el correo solo para este repo
git config user.email "fmengarelli@gmail.com"
git config user.name "francoMengarelli"
```

Paso 4: Verificar la Configuración

```
# Verificar configuración global
git config --global user.email
git config --global user.name

# Verificar configuración del repositorio actual
git config user.email
git config user.name
```

Paso 5: Hacer un Commit de Prueba

Haz un pequeño cambio y crea un commit de prueba:

```
# Crear un archivo de prueba
echo "# Test" > TEST.md

# Agregar y hacer commit
git add TEST.md
git commit -m "test: verify correct email configuration"

# Verificar que el commit use el correo correcto
git log --pretty=format:"%h | %an <%ae> | %s" -1

# Si todo está bien, puedes hacer push
git push origin main

# Eliminar el archivo de prueba (opcional)
git rm TEST.md
git commit -m "chore: remove test file"
git push origin main
```

✓ Solución 3: Reescribir el Historial de Commits (AVANZADA - USA CON PRECAUCIÓN)

⚠ **ADVERTENCIA:** Esta solución **reescribe el historial de Git**, lo que puede causar problemas si otras personas están trabajando en el repositorio. Solo usa esta opción si:

- Eres el único desarrollador del proyecto, O
- Todo el equipo está de acuerdo con reescribir el historial, O
- Los commits problemáticos son muy recientes y no hay trabajo derivado

Opción 3.1: Reescribir TODOS los Commits con Correos Problemáticos

Este script cambiará **todos** los commits que usen correos problemáticos al correo correcto:

```
#!/bin/bash

# IMPORTANTE: Haz un backup primero
git branch backup-antes-de-reescribir

# Reescribir el historial
git filter-branch --env-filter '

# Tu correo correcto de GitHub
CORRECT_EMAIL="fmengarelli@gmail.com"
CORRECT_NAME="francoMengarelli"

# Cambiar todos los commits con correos problemáticos
if [ "$GIT_COMMITTER_EMAIL" = "quantpaychain@example.com" ] ||
  [ "$GIT_COMMITTER_EMAIL" = "ai@quantpaychain.com" ] ||
  [ "$GIT_COMMITTER_EMAIL" = "agent@abacus.ai" ] ||
  [ "$GIT_COMMITTER_EMAIL" = "quantpay@quantpaychain.org" ]; then
  export GIT_COMMITTER_NAME="$CORRECT_NAME"
  export GIT_COMMITTER_EMAIL="$CORRECT_EMAIL"
fi

if [ "$GIT_AUTHOR_EMAIL" = "quantpaychain@example.com" ] ||
  [ "$GIT_AUTHOR_EMAIL" = "ai@quantpaychain.com" ] ||
  [ "$GIT_AUTHOR_EMAIL" = "agent@abacus.ai" ] ||
  [ "$GIT_AUTHOR_EMAIL" = "quantpay@quantpaychain.org" ]; then
  export GIT_AUTHOR_NAME="$CORRECT_NAME"
  export GIT_AUTHOR_EMAIL="$CORRECT_EMAIL"
fi

' --tag-name-filter cat -- --branches --tags
```

Opción 3.2: Usar `git-filter-repo` (Método Moderno y Recomendado)

`git-filter-repo` es más rápido y seguro que `git filter-branch`:

```
# Instalar git-filter-repo (si no lo tienes)
pip3 install git-filter-repo

# Crear un archivo de mapeo de correos
cat > mailmap.txt << 'EOF'
francoMengarelli <fmengarelli@gmail.com> QuantPay Chain Bot <quantpay-chain@example.com>
francoMengarelli <fmengarelli@gmail.com> QuantPay AI <ai@quantpaychain.com>
francoMengarelli <fmengarelli@gmail.com> Abacus AI Agent <agent@abacus.ai>
francoMengarelli <fmengarelli@gmail.com> QuantPay Chain Development Team <quant-pay@quantpaychain.org>
EOF

# Hacer un backup
git branch backup-antes-de-reescribir

# Aplicar el mapeo
git filter-repo --mailmap mailmap.txt --force
```

Paso Final: Force Push (SOLO DESPUÉS DE CONFIRMAR)

⚠ ADVERTENCIA CRÍTICA: El force push **sobrescribirá el historial remoto**. Asegúrate de:

1. Haber hecho un backup

2. Estar completamente seguro de los cambios
3. Notificar a tu equipo (si aplica)

```
# Verificar los cambios antes de hacer push
git log --pretty=format: "%h | %an <%ae> | %s" -20

# Si todo se ve bien, hacer force push
git push origin main --force

# Eliminar el backup local (solo si todo salió bien)
git branch -D backup-antes-de-reescribir
```

Restaurar desde Backup (si algo sale mal)

```
# Si necesitas revertir los cambios
git reset --hard backup-antes-de-reescribir
git push origin main --force
```

Recomendación Final

Para resolver el problema inmediatamente:

1. **Usa la Solución 2** para configurar Git correctamente y prevenir futuros problemas
2. **Haz un nuevo commit dummy** con el correo correcto para “empujar” el commit problemático más atrás en el historial
3. **Vercel debería poder desplegar** el nuevo commit sin problemas

Comandos rápidos:

```
# 1. Configurar el correo correcto
cd /ruta/a/quantpaychain-mvpro
git config user.email "fmengarelli@gmail.com"
git config user.name "francoMengarelli"

# 2. Hacer un commit vacío con el correo correcto
git commit --allow-empty -m "fix: update git email configuration for Vercel deployment"

# 3. Hacer push
git push origin main

# 4. Verificar en Vercel
# Ve a tu proyecto en Vercel y verifica que el despliegue se active automáticamente
```

¿Por qué funciona esto?

Vercel verifica el **commit más reciente** (HEAD) cuando hace el despliegue. Al crear un nuevo commit con un correo válido, Vercel podrá verificar la autoría y proceder con el despliegue. Los commits antiguos con correos problemáticos permanecen en el historial, pero no bloquean el despliegue.

Recursos Adicionales

- [GitHub - Managing commit signature verification](https://docs.github.com/en/authentication/managing-commit-signature-verification) (https://docs.github.com/en/authentication/managing-commit-signature-verification)
 - [GitHub - Setting your commit email address](https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-email-preferences/setting-your-commit-email-address) (https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-email-preferences/setting-your-commit-email-address)
 - [Vercel - GitHub Integration](https://vercel.com/docs/deployments/git/vercel-for-github) (https://vercel.com/docs/deployments/git/vercel-for-github)
 - [Git - Rewriting History](https://git-scm.com/book/en/v2/Git-Tools-Rewriting-History) (https://git-scm.com/book/en/v2/Git-Tools-Rewriting-History)
-

Preguntas Frecuentes

¿Por qué mis commits tienen correos incorrectos?

Los commits con correos como `quantpaychain@example.com`, `agent@abacus.ai`, etc., probablemente fueron creados por:

- Scripts automatizados o bots
- Agentes de IA (como Abacus AI)
- Configuración incorrecta de Git en tu máquina local
- Variables de entorno mal configuradas en CI/CD

¿Cuál correo debo usar?

Usa el correo que está **verificado en tu cuenta de GitHub**. Puedes verificar tus correos en: <https://github.com/settings/emails>

¿Esto afectará a otros colaboradores?

- **Solución 1 y 2:** NO afectan a otros (son cambios locales o de cuenta)
- **Solución 3:** Sí afecta (reescribe historial), requiere coordinación con el equipo

¿Qué pasa si no puedo verificar `quantpaychain@example.com`?

No puedes verificar `example.com` porque es un dominio reservado para ejemplos. Debes:

1. Usar la **Solución 2** para futuros commits
 2. Hacer un commit nuevo con correo válido (recomendación rápida)
 3. O usar la **Solución 3** para reescribir el historial (avanzado)
-

Soporte

Si necesitas ayuda adicional:





1. Revisa la documentación de GitHub sobre correos de commit
 2. Contacta al soporte de Vercel si el problema persiste después de aplicar las soluciones
 3. Verifica los logs de despliegue en Vercel para mensajes de error específicos
-

Fecha de creación: 24 de octubre de 2025

Última actualización: 24 de octubre de 2025

Versión: 1.0

✨ Estado Actual

-  Problema identificado
 -  Correos problemáticos documentados
 -  Soluciones proporcionadas
 -  Pendiente: Aplicar solución y verificar despliegue en Vercel
-

¡Buena suerte con el despliegue! 🚀