

Guía de Despliegue - Plataforma de Transparencia Municipal

Esta guía te ayudará a desplegar la Plataforma de Transparencia Municipal en **Vercel** (frontend) y **Railway** (backend).





Tabla de Contenidos

1. [Prerequisitos](#)
2. [Arquitectura de Despliegue](#)
3. [Paso 1: Desplegar Backend en Railway](#)
4. [Paso 2: Desplegar Frontend en Vercel](#)
5. [Paso 3: Configurar Dominio Personalizado](#)
6. [Paso 4: Verificación y Testing](#)
7. [Variables de Entorno](#)
8. [Troubleshooting](#)
9. [Mantenimiento](#)




Prerequisitos

Antes de comenzar, asegúrate de tener:

Cuentas Necesarias

-  Cuenta de [Railway](https://railway.app) (<https://railway.app>) (con método de pago configurado)
-  Cuenta de [Vercel](https://vercel.com) (<https://vercel.com>) (plan gratuito es suficiente)
-  Cuenta de GitHub/GitLab/Bitbucket con el código del proyecto
-  Base de datos PostgreSQL (ya configurada en Abacus.AI)

Información Requerida

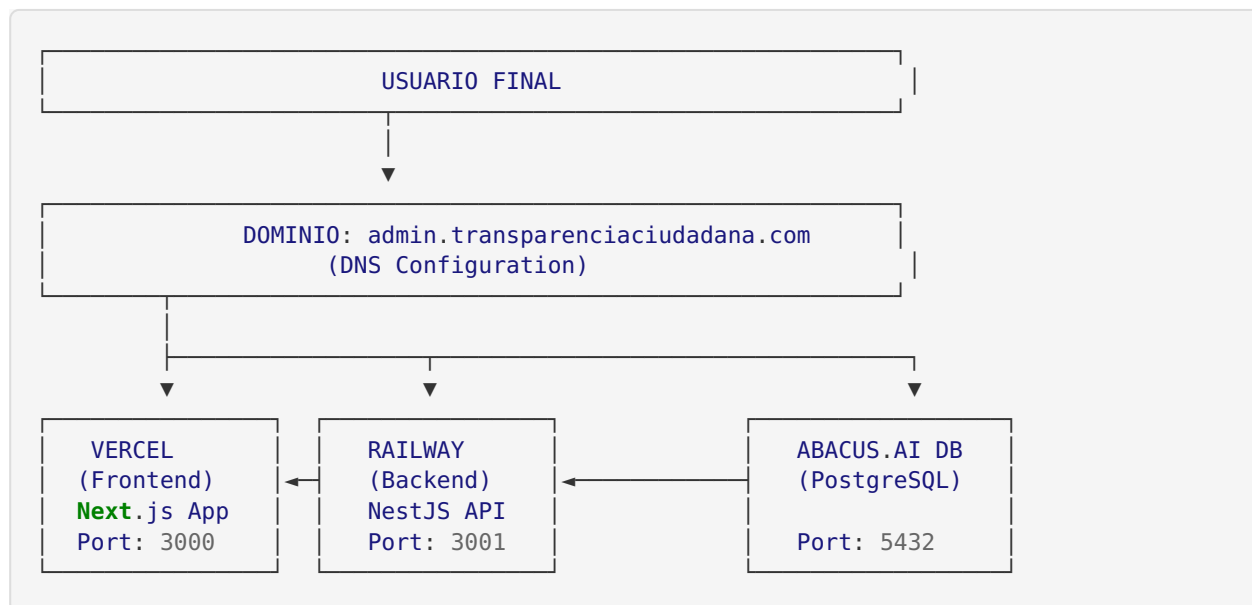
-  **DATABASE_URL**: Tu conexión a PostgreSQL de Abacus.AI
`postgresql://role_f804661bc:PK43sEVjwsuP1X1ArUp09gUYk4L4Ej48@db-f804661bc.db002.hosteddb.reai.io:5432/f804661bc?connect_timeout=15`
-  **JWT Secrets**: Guarda tus secrets existentes
-  **Dominio**: `admin.transparenciaciudadana.com` (o tu dominio personalizado)

Herramientas de CLI (Opcional)

```
# Instalar Railway CLI
npm install -g @railway/cli

# Instalar Vercel CLI
npm install -g vercel
```

Arquitectura de Despliegue



Flujo de Datos:

1. Usuario accede a `admin.transparenciaciudadana.com`
2. Vercel sirve la aplicación Next.js (frontend)
3. Frontend hace peticiones a `/api/*` que se proxean a Railway
4. Railway ejecuta el backend NestJS
5. Backend consulta la base de datos PostgreSQL en Abacus.AI

Paso 1: Desplegar Backend en Railway

1.1 Crear Nuevo Proyecto

1. Ve a [Railway.app](https://railway.app) (<https://railway.app>) e inicia sesión
2. Click en **"New Project"**
3. Selecciona **"Deploy from GitHub repo"**
4. Conecta tu cuenta de GitHub y selecciona el repositorio
5. Railway detectará automáticamente el proyecto

1.2 Configurar el Servicio Backend

1. En el proyecto, click en **"New"** → **"GitHub Repo"**
2. Selecciona tu repositorio y la rama `main` o `master`
3. Railway creará el servicio automáticamente

1.3 Configurar Variables de Entorno

En el dashboard de Railway, ve a tu servicio → **Variables** y agrega:

```
# Database
DATABASE_URL=postgresql://role_f804661bc:PK43sEVjwsuP1X1ArUp09gUYk4l4Ej48@db-f804661bc.db002.hosteddb.reai.io:5432/f804661bc?connect_timeout=15

# JWT Configuration
JWT_SECRET=prod-secret-c8a9f3e2d1b4a5c7f9e8d2b1a4c6f8e9d2b5a7c9f1e3d5b7a9c1f3e5d7b9a1c3
JWT_REFRESH_SECRET=prod-refresh-secret-a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0u1v2w3x4y5z6
JWT_EXPIRATION=15m
JWT_REFRESH_EXPIRATION=7d

# NextAuth Secret
NEXTAUTH_SECRET=qr6zNaFiLtKpLB6QjrPVPx4nEMbZFKeB

# Encryption
ENCRYPTION_KEY=KN6X9ososWFJv8WvNy2r7rXr3+B2XKc0oQTW0tV6qc=

# Environment
NODE_ENV=production
PORT=3001

# CORS (actualizar después de desplegar el frontend)
CORS_ORIGIN=https://admin.transparenciaciudadana.com

# Throttling
THROTTLE_TTL=60
THROTTLE_LIMIT=100

# Logging
LOG_LEVEL=info
LOG_FILE_ENABLED=false

# Swagger (deshabilitado en producción)
SWAGGER_ENABLED=false
```

1.4 Configurar el Directorio de Build

1. Ve a **Settings** → **Build**
2. Configura:
 - **Root Directory:** `apps/api`
 - **Build Command:** `npm install && npm run build`
 - **Start Command:** `npm run start:prod`

1.5 Generar Dominio Público

1. Ve a **Settings** → **Networking**
2. Click en **“Generate Domain”**
3. Copia el URL generado (ej: `your-app.railway.app`)
4. **Guarda este URL** - lo necesitarás para configurar Vercel

1.6 Verificar Despliegue

1. Espera a que el despliegue termine (2-5 minutos)
2. Verifica el health endpoint:


```
bash
curl https://your-app.railway.app/api/health
```

3. Deberías recibir:

```
json
{
  "status": "ok",
  "timestamp": "2025-11-26T...",
  "uptime": 123.456,
  "environment": "production"
}
```

Paso 2: Desplegar Frontend en Vercel

2.1 Importar Proyecto

1. Ve a [Vercel.com](https://vercel.com) (<https://vercel.com>) e inicia sesión
2. Click en **“Add New...”** → **“Project”**
3. Click en **“Import Git Repository”**
4. Selecciona tu repositorio
5. Vercel detectará automáticamente que es un proyecto Next.js

2.2 Configurar el Proyecto

En la configuración del proyecto:

1. **Framework Preset:** Next.js (detectado automáticamente)
2. **Root Directory:** `frontend/nextjs_space`
3. **Build Command:** `npm run build`
4. **Output Directory:** `.next` (default)
5. **Install Command:** `npm install`

2.3 Configurar Variables de Entorno

En **Environment Variables**, agrega:

```
# API Backend (URL de Railway del paso anterior)
NEXT_PUBLIC_API_URL=https://your-app.railway.app

# Database (mismo que el backend)
DATABASE_URL=postgresql://role_f804661bc:PK43sEVjwsuP1X1ArUp09gUYk4l4Ej48@db-f804661bc.db002.hosteddb.reai.io:5432/f804661bc?connect_timeout=15

# NextAuth
NEXTAUTH_SECRET=qr6zNaFiLtKpLB6QjrPVPx4nEMbZFKeB

# AWS Storage (opcional si usas Abacus.AI storage)
AWS_PROFILE=hosted_storage
AWS_REGION=us-west-2
AWS_BUCKET_NAME=abacusai-apps-19658dee6a168bd21a949e89-us-west-2
AWS_FOLDER_PREFIX=5986/

# Environment
NODE_ENV=production
```

2.4 Actualizar vercel.json

Antes de desplegar, edita `frontend/nextjs_space/vercel.json` y reemplaza:

```
{
  "rewrites": [
    {
      "source": "/api/:path*",
      "destination": "https://your-app.railway.app/api/:path*"
    }
  ]
}
```

Reemplaza `your-app.railway.app` con el dominio de Railway del Paso 1.5

2.5 Desplegar

1. Click en **“Deploy”**
2. Espera a que termine el build (3-7 minutos)
3. Vercel te dará un URL temporal (ej: `your-app.vercel.app`)

2.6 Actualizar CORS en Railway

1. Regresa al dashboard de Railway
2. Edita la variable `CORS_ORIGIN` :

`env`
`CORS_ORIGIN=https://your-app.vercel.app,https://admin.transparenciaciudadana.com`
3. Guarda los cambios (Railway redesplegará automáticamente)

Paso 3: Configurar Dominio Personalizado

3.1 Configurar Dominio en Vercel

1. En Vercel, ve a tu proyecto → **Settings** → **Domains**
2. Click en **“Add”**
3. Ingresa: `admin.transparenciaciudadana.com`
4. Click en **“Add”**

3.2 Configurar DNS

Vercel te mostrará los registros DNS que necesitas agregar:

Opción A: Registro A (recomendado)


```
Type: A
Name: admin
Value: 76.76.21.21
TTL: Auto
```

Opción B: Registro CNAME


```
Type: CNAME
Name: admin
Value: cname.vercel-dns.com
TTL: Auto
```

3.3 Actualizar Proveedor de DNS

1. Ve a tu proveedor de dominio (ej: GoDaddy, Namecheap, Cloudflare)
2. Busca la sección de **DNS Management** o **Zone File**
3. Agrega el registro proporcionado por Vercel
4. Guarda los cambios

 **Nota:** Los cambios DNS pueden tardar hasta 48 horas, pero usualmente se propagan en 5-30 minutos.

3.4 Verificar Dominio

1. Espera a que el DNS se propague
2. Vercel verificará automáticamente el dominio
3. Verás un  cuando esté listo

3.5 Actualizar CORS Final

Actualiza la variable `CORS_ORIGIN` en Railway una vez más:

```
CORS_ORIGIN=https://admin.transparenciaciudadana.com
```

Paso 4: Verificación y Testing

4.1 Checklist de Verificación

- ☐ Backend responde en Railway:


```
bash
curl https://your-app.railway.app/api/health
```
- ☐ Frontend carga en Vercel:


```
bash
curl https://admin.transparenciaciudadana.com
```
- ☐ Login funciona correctamente:
- Email: `superadmin@transparencia.cl`
- Password: `demo12345`
- ☐ API endpoints responden:


```
bash
curl -H "Authorization: Bearer YOUR_TOKEN" \
  https://your-app.railway.app/api/municipalities
```
- ☐ Database connections funcionan

- [] Uploads de archivos funcionan (si aplica)
- [] Notificaciones se crean correctamente

4.2 Testing Manual

1. Login Superadmin

- Ve a `/superadmin/login`
- Ingresa credenciales del superadmin
- Verifica que accedas al dashboard

2. Crear Municipalidad de Prueba

- Ve a **Crear Tenant**
- Crea una municipalidad de prueba
- Verifica que se cree correctamente

3. Login Administrador Municipal

- Ve a `/demo/admin/login` (o tu slug de prueba)
- Ingresa credenciales del admin municipal
- Verifica acceso al panel de administración

4. CRUD Operations

- Crea un presupuesto de prueba
- Crea un gasto de prueba
- Verifica que las operaciones funcionen

5. Portal Ciudadano

- Ve a `/demo` (o tu slug de prueba)
- Verifica que los datos públicos se muestren
- Verifica gráficos y KPIs

4.3 Pruebas de Rendimiento

```
# Prueba de carga con Apache Bench (opcional)
ab -n 100 -c 10 https://admin.transparenciaciudadana.com/

# Prueba de health endpoint
while true; do
  curl https://your-app.railway.app/api/health
  sleep 5
done
```

Variables de Entorno

Backend (Railway)

Variable	Descripción	Valor de Ejemplo	Requerida
DATABASE_URL	URL de conexión PostgreSQL	postgresql:// user:pass@host:5432/ db	✓
JWT_SECRET	Secret para JWT access tokens	your-secret-here	✓
JWT_REFRESH_SECRET	Secret para JWT refresh tokens	your-refresh-secret	✓
JWT_EXPIRATION	Expiración de access token	15m	✓
JWT_REFRESH_EXPIRATION	Expiración de refresh token	7d	✓
NEXTAUTH_SECRET	Secret para NextAuth	your-nextauth-secret	✓
ENCRYPTION_KEY	Clave de encriptación (Base64)	KN6X9os...	✓
NODE_ENV	Entorno de ejecución	production	✓
PORT	Puerto del servidor	3001	✓
CORS_ORIGIN	Orígenes permitidos (CORS)	https://yourdomain.com	✓
THROTTLE_TTL	Tiempo para rate limiting (ms)	60000	✗
THROTTLE_LIMIT	Límite de requests	100	✗
LOG_LEVEL	Nivel de logging	info	✗
SWAGGER_ENABLED	Habilitar Swagger docs	false	✗

Frontend (Vercel)

Variable	Descripción	Valor de Ejemplo	Requerida
<code>NEXT_PUBLIC_API_URL</code>	URL del backend en Railway	<code>https://your-app.railway.app</code>	✓
<code>DATABASE_URL</code>	URL de PostgreSQL (para API routes)	<code>postgresql://...</code>	✓
<code>NEXTAUTH_SECRET</code>	Secret para NextAuth	<code>your-nextauth-secret</code>	✓
<code>NODE_ENV</code>	Entorno de ejecución	<code>production</code>	✓
<code>AWS_PROFILE</code>	Perfil AWS (si usas S3)	<code>hosted_storage</code>	✗
<code>AWS_REGION</code>	Región AWS	<code>us-west-2</code>	✗
<code>AWS_BUCKET_NAME</code>	Nombre del bucket S3	<code>your-bucket</code>	✗

Ver archivo `ENV_VARIABLES_REFERENCE.md` para copiar/pegar valores rápidamente.

Troubleshooting

Problema: Backend no responde en Railway

Síntomas: Error 502 o timeout al acceder al backend

Soluciones:

1. Verifica los logs en Railway:

Railway Dashboard → Your Service → Deployments → View Logs

1. Verifica que el puerto esté correctamente configurado:

```
env
```

```
PORT=3001
```

2. Verifica la variable `DATABASE_URL` esté correcta

3. Reinicia el servicio:

Railway Dashboard → Your Service → Settings → Restart

Problema: Frontend no puede conectarse al Backend

Síntomas: Errores CORS o "Network Error" en el frontend

Soluciones:

1. Verifica `NEXT_PUBLIC_API_URL` en Vercel:

```
env
```

```
NEXT_PUBLIC_API_URL=https://your-app.railway.app
```

1. Verifica `CORS_ORIGIN` en Railway incluya el dominio de Vercel:

```
env
```

```
CORS_ORIGIN=https://your-app.vercel.app,https://admin.transparenciaciudadana.com
```

2. Verifica que `vercel.json` tenga el rewrite correcto

Problema: Login no funciona (Credenciales incorrectas)

Síntomas: "Credenciales incorrectas" al intentar login

Soluciones:

1. Verifica que el usuario existe en la base de datos:

```
bash
```

```
node verify-users.js
```

1. Resetea la contraseña del superadmin:

```
bash
```

```
node ensure-superadmin.js
```

2. Verifica que `JWT_SECRET` y `NEXTAUTH_SECRET` sean iguales en backend y frontend

3. Revisa los logs del backend para ver el error específico

Problema: Database connection failed

Síntomas: Error "Cannot connect to database"

Soluciones:

1. Verifica la `DATABASE_URL` :

- Usuario y contraseña correctos
- Host y puerto correctos
- Nombre de base de datos correcto

1. Verifica que la base de datos en Abacus.AI esté activa

2. Prueba la conexión manualmente:

```
bash
```

```
psql "postgresql://role_f804661bc:PK43sEVjwsuP1X1ArUp09gUYk4l4Ej48@db-f804661bc.db002.hosteddb.reai.io:5432/f804661bc"
```

Problema: Build fails en Railway o Vercel

Síntomas: El deployment falla durante el build

Soluciones (Railway):

1. Verifica el **Root Directory**: `apps/api`
2. Verifica el **Build Command**: `npm install && npm run build`
3. Revisa los logs de build para errores específicos
4. Asegúrate que todas las dependencias estén en `package.json`

Soluciones (Vercel):

1. Verifica el **Root Directory**: `frontend/nextjs_space`
2. Verifica el **Build Command**: `npm run build`

3. Asegúrate que todas las variables de entorno estén configuradas
4. Revisa los logs de build en Vercel

Problema: 404 en rutas del frontend

Síntomas: Algunas rutas devuelven 404

Soluciones:

1. Verifica que la estructura de carpetas en `app/` sea correcta
2. Asegúrate que `middleware.ts` esté configurado correctamente
3. Limpia el caché de Vercel:

Vercel Dashboard → Your Project → Settings → Clear Cache

Problema: Variables de entorno no se aplican

Síntomas: Los cambios en variables no tienen efecto

Soluciones (Railway):

1. Después de cambiar variables, Railway redespiega automáticamente
2. Si no, fuerza un redespigue manualmente
3. Verifica que las variables estén en el entorno correcto (Production)

Soluciones (Vercel):

1. Las variables nuevas requieren un nuevo deployment
2. Ve a **Deployments** → último deployment → **"Redeploy"**
3. O haz un nuevo commit al repositorio

Mantenimiento

Actualizar el Código

Railway (Backend)

1. Haz push a tu rama principal en GitHub:

```
bash
git add .
git commit -m "Update backend"
git push origin main
```

2. Railway detectará el cambio y redesplegará automáticamente

Vercel (Frontend)

1. Haz push a tu rama principal:

```
bash
git add .
git commit -m "Update frontend"
git push origin main
```

2. Vercel redesplegará automáticamente

Rollback

En Railway

1. Ve a **Deployments**
2. Encuentra el deployment anterior

3. Click en "..."/> → "Redeploy"

En Vercel

1. Ve a **Deployments**
2. Encuentra el deployment anterior
3. Click en "..."/> → "Promote to Production"

Monitoreo

Railway

- **Logs:** Railway Dashboard → Your Service → View Logs
- **Metrics:** Railway Dashboard → Your Service → Metrics
- **Health:** <https://your-app.railway.app/api/health>

Vercel

- **Analytics:** Vercel Dashboard → Analytics
- **Logs:** Vercel Dashboard → Deployments → View Function Logs
- **Speed Insights:** Vercel Dashboard → Speed Insights

Backups de Base de Datos

⚠ **Importante:** Configura backups automáticos en Abacus.AI:

1. Ve a tu base de datos en Abacus.AI
2. Configura backups automáticos diarios
3. Guarda backups locales periódicamente:

```
bash
```

```
pg_dump "postgresql://..." > backup-$(date +%Y%m%d).sql
```

Escalabilidad

Railway

- Railway escala automáticamente según demanda
- Puedes ajustar recursos en **Settings** → **Resources**

Vercel

- Vercel escala automáticamente (plan Pro tiene más capacidad)
- Edge Functions para mejor rendimiento global






¡Despliegue Completo!

Si llegaste hasta aquí, tu plataforma debería estar funcionando en:

- 🌐 **Frontend:** <https://admin.transparenciaciudadana.com>
- 🔧 **Backend:** <https://your-app.railway.app>
- 🗄 **Database:** Abacus.AI PostgreSQL




Próximos Pasos

1. ✅ Configura monitoreo y alertas
2. ✅ Configura backups automáticos de la base de datos
3. ✅ Documenta tu flujo de CI/CD

4.  Configura SSL/TLS (Vercel y Railway lo hacen automáticamente)
5.  Revisa logs regularmente
6.  Planifica mantenimiento y actualizaciones

Soporte

Si necesitas ayuda:

-  **Email:** soporte@transparenciaciudadana.com
-  **Documentación:** Ver carpeta `/docs`
-  **Issues:** GitHub Issues en el repositorio

Creado por: Municipal Transparency Platform Team

Última actualización: 26 de noviembre de 2025

Versión: 1.0.0