

Railway Deployment - Status y Monitoreo

Commit Desplegado: 9c7ccfc9

Fecha: 13 Diciembre 2024, 13:13 UTC

Push Completado: 13:13 UTC

DEPLOYMENT AUTOMÁTICO EN MARCHA

Railway detecta automáticamente los nuevos commits vía webhook de GitHub. El deployment comenzó automáticamente cuando se detectó el push del commit 9c7ccfc9.

NO necesitas hacer nada manualmente. Railway está construyendo ahora mismo.

CÓMO MONITOREAR EL PROGRESO

1. Acceder a Railway Dashboard

URL: <https://railway.app/dashboard>

2. Localizar el Proyecto

Buscar: - **Proyecto:** loving-creation - **Servicio:** inmov-a-app - **URL:** inmov-a-app

3. Ver Build Logs en Tiempo Real

1. Haz clic en el servicio
2. Ve a la pestaña “Deployments”
3. Busca el deployment con commit 9c7ccfc9
4. Haz clic para ver “Build Logs”

4. Identificadores del Deployment

- **Commit Hash:** 9c7ccfc9
 - **Commit Message:** “ merge: Sincronizar repo principal con cambios de Docker”
 - **Branch:** main
-

TIMELINE ESTIMADO DEL BUILD

Fase 1: Detection (1-2 min) - COMPLETADO

13:13 UTC - Push detectado

13:14 UTC - Webhook activado

13:14 UTC - Railway comienza build

Fase 2: Docker Build Iniciado (0-1 min) - EN PROGRESO

Railway lee railway.toml

builder = "DOCKERFILE" detectado

Carga contexto de build

Fase 3: Stage 1 - deps (3-5 min) - SIGUIENTE

FROM node:20-alpine AS deps
→ Descarga imagen base (40 MB)
→ Instala libc6-compat
→ COPY package.json yarn.lock
→ RUN yarn install --frozen-lockfile --ignore-engines
 Instala ~200 dependencias

Tiempo estimado: 13:15 - 13:20 UTC

Fase 4: Stage 2 - builder (10-15 min) - PENDIENTE

FROM node:20-alpine AS builder
→ COPY node_modules desde deps
→ COPY código fuente
→ RUN npx prisma generate
 Genera @prisma/client
→ RUN yarn build
 Compila 234 páginas estáticas
 Genera .next/standalone/

Tiempo estimado: 13:20 - 13:35 UTC

Fase 5: Stage 3 - runner (2-3 min) - PENDIENTE

FROM node:20-alpine AS runner
→ Imagen limpia (40 MB)
→ Crea usuario nextjs:nodejs
→ COPY archivos necesarios
→ USER nextjs
→ CMD ["node", "server.js"]

Tiempo estimado: 13:35 - 13:38 UTC

Fase 6: Container Start (1-2 min) - PENDIENTE

→ Railway inicia contenedor
→ Inyecta variables de entorno
→ Ejecuta: node server.js
→ Servidor escucha en 0.0.0.0:3000
→ Health check: GET /api/health

Tiempo estimado: 13:38 - 13:40 UTC

Fase 7: Deployment Complete - PENDIENTE

→ Health check passed (200 OK)
→ Railway actualiza DNS
→ Traffic redirigido al nuevo contenedor
→ Deployment succeeded

Tiempo estimado: 13:40 UTC

PROGRESO ACTUAL

Tiempo transcurrido: ~4 minutos desde push

Fase actual: Detection/Build Iniciado

Progreso estimado: 10%

Próximo hito: Stage 1 (deps) comenzará en ~2 minutos

LOGS ESPERADOS EN RAILWAY

Detection

```
"New commit detected: 9c7ccfc9"
"Branch: main"
"Detected Dockerfile in nextjs_space/"
```

Docker Build

```
"Building with Docker (builder=DOCKERFILE)"
"Step 1/XX : FROM node:20-alpine AS base"
"Step X/XX : FROM base AS deps"
"Step X/XX : RUN apk add --no-cache libc6-compat"
"Step X/XX : RUN yarn install --frozen-lockfile --ignore-engines"
"Step X/XX : FROM base AS builder"
"Step X/XX : RUN npx prisma generate"
"Prisma schema loaded from prisma/schema.prisma"
"Generated Prisma Client"
"Step X/XX : RUN yarn build"
"Route (app)                                Size      First Load JS"
" /                                           XXX B      XXX kB"
"... [234 páginas compiladas]"
"Step X/XX : FROM base AS runner"
"Step X/XX : CMD [\"node\", \"server.js\"]"
"Successfully built"
"Successfully tagged"
```

Container Start

```
"Starting container..."
"Server listening on 0.0.0.0:3000"
"ready - started server on 0.0.0.0:3000, url: http://localhost:3000"
```

Health Check

```
"Health check: GET /api/health"
"Response: 200 OK"
"Container healthy"
```

Deployment Success

```
"Deployment succeeded"
"URL: https://inmova.app"
```

SI HAY ERRORES

Error Común: “Cannot find module ‘@prisma/client’”

Causa: Prisma Client no se generó correctamente

Buscar en logs:

```
Error: Cannot find module '@prisma/client'
```

Verificar: - Stage builder debe mostrar: "Generated Prisma Client" - Si no aparece, hay un problema con `npx prisma generate`

Solución: Ya está implementado en línea 23 del Dockerfile

Error Común: “ENOENT: no such file or directory, open ‘package.json’”

Causa: Railway no encuentra package.json

Buscar en logs:

```
COPY package.json yarn.lock* ./
ENOENT: no such file or directory
```

Verificar: - Railway Root Directory debe ser: `nextjs_space/` - Dockerfile debe estar en: `nextjs_space/Dockerfile`

Solución: Ya está configurado correctamente

Error Común: “yarn install failed”

Causa: Conflictos de versiones de dependencias

Buscar en logs:

```
error <package>@<version>: The engine "node" is incompatible
```

Verificar: - Dockerfile debe usar: `--ignore-engines` - Línea 12: `RUN yarn install --frozen-lockfile --ignore-engines`

Solución: Ya está implementado

Error Común: “Build timed out”

Causa: Build toma más de 30 minutos

Buscar en logs:

```
Build exceeded maximum time limit
```

Posibles causas: - Contexto de build muy grande - Problemas de red con npm/yarn registry - Recursos insuficientes

Solución: - `.dockerignore` ya optimizado - Usar caché de Docker (Railway lo hace automáticamente) - Si persiste, contactar soporte de Railway

NOTIFICACIONES

Railway Email Notifications

Railway enviará emails automáticamente: - **Build Started:** “Deployment started for inmovia-app” - **Build Success:** “Deployment successful for inmovia-app” - **Build Failed:** “Deployment failed for inmovia-app”

Revisar tu bandeja de entrada (el email asociado a la cuenta de Railway).

VERIFICACIÓN POST-DEPLOYMENT

Una vez que el deployment esté completo:

1. Verificar URL Principal

```
curl https://inmovia.app
```

Respuesta esperada: HTML de la página principal

2. Verificar Health Check

```
curl https://inmovia.app/api/health
```

Respuesta esperada:

```
{
  "status": "ok",
  "timestamp": "2024-12-13T13:40:00.000Z",
  "uptime": 123.45,
  "environment": "production"
}
```

3. Verificar Login

1. Ir a: <https://inmovia.app/login>
2. Intentar login con credenciales de prueba
3. Verificar que redirige a dashboard

4. Verificar Módulos Principales

- Dashboard: <https://inmovia.app/home>
 - Propiedades: <https://inmovia.app/propiedades>
 - Inquilinos: <https://inmovia.app/tenants>
 - Room Rental: <https://inmovia.app/room-rental>
-

MÉTRICAS ESPERADAS

Build Time

- Total: 20-25 minutos
- Stage 1 (deps): 3-5 min
- Stage 2 (builder): 10-15 min
- Stage 3 (runner): 2-3 min
- Container start: 1-2 min

Image Size

- **Final image:** ~150 MB
- **Base (Alpine):** 40 MB
- **Dependencies:** 60 MB
- **Build artifacts:** 30 MB
- **Prisma:** 10 MB
- **Assets:** 10 MB

Runtime Resources

- **RAM:** 256 MB mínimo, 512 MB recomendado
- **CPU:** 0.5 vCPU mínimo
- **Startup time:** 5-10 segundos
- **Cold start:** 10-15 segundos

TROUBLESHOOTING RÁPIDO

Síntoma	Causa Probable	Solución
Build no inicia	Webhook no configurado	Verificar en Railway → Settings → Deployments
“package.json not found”	Root Directory incorrecto	Debe ser <code>nextjs_space/</code>
“Cannot find Prisma”	Prisma generate falló	Ver logs de Stage 2 (builder)
Build muy lento	Contexto muy grande	<code>.dockerignore</code> ya optimizado
Container crashea	Error en runtime	Ver logs de Container Start
Health check falla	<code>/api/health</code> no existe	Ya está implementado

SOPORTE

Railway Support

- Dashboard: <https://railway.app/dashboard>
- Docs: <https://docs.railway.app>
- Community: <https://discord.gg/railway>

INMOVA Team

- Email: dvillagrab@hotmail.com
- GitHub Repo: <https://github.com/dvillagrablanco/innova-app>

CUANDO ESTÉ COMPLETO

Verás en Railway Dashboard: - **Status:** Active - **Health:** Healthy - **Uptime:** XX minutes - **URL:** `innova.app` (clickeable)

Y podrás acceder a:

`https://innova.app`

¡Felicidades! El deployment habrá sido exitoso.

Última actualización: 13 Diciembre 2024, 13:17 UTC
Status: Build en progreso
ETA: ~13:40 UTC (23 minutos restantes)