

Railway Deployment Root Directory Fix

Investigation Summary

I've investigated the Railway deployment failure for the workigom backend service. Here's what I found and fixed.

What I Discovered

1. Project Structure Analysis

✓ TypeScript Configuration (`tsconfig.json`):

- `outDir: "./dist"` - Compiled files go to `dist/` directory
- `rootDir: "./src"` - Source files are in `src/` directory
- Configuration is **CORRECT** ✓

✓ Package.json:

- `"start": "node dist/server.js"` - Expects `dist/server.js`
- `"build": "tsc"` - Compiles TypeScript to JavaScript
- Configuration is **CORRECT** ✓

✓ Entry Point File:

- Main entry file: `src/server.ts`
- Compiles to: `dist/server.js`
- Entry point is **CORRECT** ✓

✓ Dockerfile:

- Uses multi-stage build with Node.js 20 Alpine
- Runs `npm run build` to compile TypeScript
- Start script executes `node dist/server.js`
- Dockerfile is **CORRECT** ✓

2. Build Verification

I ran the build process locally:

```
npm run build
```

Result: ✓ Build completed successfully!

Generated files in `dist/` directory:

```

dist/
├── server.js      ☒ Main entry point ☒
├── server.d.ts
├── server.js.map
├── app.js
├── app.d.ts
├── app.js.map
├── config/
├── controllers/
├── middleware/
├── routes/
├── types/
└── utils/

```

! ROOT CAUSE IDENTIFIED

Based on the Railway dashboard screenshots you provided, I identified the **critical issue**:

Railway Root Directory Misconfiguration

Your Railway service settings show:

Root Directory: backend/

BUT your backend files have been moved to the project root directory!

This mismatch causes Railway to:

1. Look for files in backend/ subdirectory
2. Not find the Dockerfile in the expected location
3. Fail to build or use the wrong Dockerfile

🔧 Changes Made

1. Updated railway.toml

Added explicit root directory configuration:

```

[build]
builder = "dockerfile"
dockerfilePath = "Dockerfile"
root = "." ☒ NEW ☒ Explicitly set root to project root

```

Commit: fix: Add root directory configuration to railway.toml to fix deployment path issues

Status: ☒ Committed and pushed to GitHub

2. Cleaned Up Source Directory

- Removed stray src/main.tsx file (frontend React file that was causing build errors)
- This file was not in the repository, just a local leftover
- Build now completes cleanly



REQUIRED ACTIONS ON RAILWAY DASHBOARD

CRITICAL: You need to update the Railway service settings manually:

Option 1: Update Root Directory in Railway Settings

1. Go to Railway Dashboard
2. Select your **workigom** service
3. Go to **Settings** tab
4. Find **Root Directory** setting
5. **Change from:** `backend/` or `backend`
6. **Change to:** `(empty)` or `.`
7. Click **Save**
8. Trigger a new deployment

Option 2: Let railway.toml Override

Railway should now read the `root = "."` setting from `railway.toml` and use it. Try triggering a new deployment first to see if this works automatically.



Verification Checklist

After updating Railway settings, verify:

- ☐ ☒ Railway build starts and finds the Dockerfile
- ☐ ☒ Build completes without "nginx.conf not found" errors
- ☐ ☒ TypeScript compilation succeeds
- ☐ ☒ Prisma migrations run successfully
- ☐ ☒ Application starts on the PORT environment variable
- ☐ ☒ Health check endpoint responds at `/api/health`
- ☐ ☒ Service status shows as "Active"



Expected Deployment Flow

Once the root directory is fixed, Railway should:

1. Build Stage:

- ✓ Use Dockerfile from project root
- ✓ Install dependencies
- ✓ Generate Prisma client
- ✓ Compile TypeScript (`src/` → `dist/`)

2. Production Stage:

- ✓ Copy `dist/` directory
- ✓ Install production dependencies
- ✓ Generate Prisma client
- ✓ Create uploads directories

3. Deployment:

- ✓ Run `start.sh` script
- ✓ Execute Prisma migrations
- ✓ Start server with: `node dist/server.js`
- ✓ Listen on `$PORT` (provided by Railway)

Current Project Structure

workigom/	
 Dockerfile	 Used by Railway 
 railway.toml	 Updated with root = "." 
 package.json	 Backend dependencies 
 tsconfig.json	 TypeScript config 
 prisma/	 Database schema 
 schema.prisma	
 migrations/	
 src/	 Backend source code 
 server.ts	 Entry point 
 app.ts	
 config/	
 controllers/	
 middleware/	
 routes/	
 services/	
 types/	
 utils/	
 dist/	 Generated by build (gitignored)
 server.js	 Executed by Docker 
 src-frontend/	 Frontend code (separate)
 backend/	 Old location (deprecated)
 Dockerfile	 Reference only
 node_modules/	





Why This Issue Occurred



1. **Previous Setup:** Backend files were in `backend/` subdirectory
2. **Migration:** Files were moved to root directory to fix Railway deployment
3. **Stale Config:** Railway service still had `Root Directory: backend/` set
4. **Result:** Railway looked for files in wrong location

Additional Notes

Dockerfile Verification

The current Dockerfile is correct and includes:

-  Prisma client generation
-  TypeScript compilation
-  Multi-stage build for smaller image
-  Health check configuration

-  Startup script with migration
-  Proper environment variable handling

Environment Variables

Ensure these are set in Railway:

- `DATABASE_URL` - PostgreSQL connection string
- `JWT_SECRET` - Secret key for JWT tokens
- `NODE_ENV` - Set to "production"
- `PORT` - Automatically set by Railway

Railway.toml Configuration

The updated railway.toml now explicitly sets:

- Builder: dockerfile
- Dockerfile path: Dockerfile (in root)
- Root directory: . (project root)
- Start command: /app/start.sh
- Health check: /api/health



Summary



What's Working

- TypeScript compilation
- Dockerfile configuration
- Start script and entry point
- Build process
- Project structure



What Needs Action

- **Railway Root Directory setting** must be changed from `backend/` to empty or `.`
- This can be done in Railway Dashboard → Service Settings → Root Directory
- Or Railway should automatically use the `root = "."` from railway.toml



Next Steps

1. Update Railway service Root Directory setting (or verify railway.toml overrides it)
2. Trigger a new deployment
3. Monitor the build logs to ensure it uses the root Dockerfile
4. Verify the application starts successfully
5. Test the health check endpoint
6. Verify database connectivity



Need Help?

If deployment still fails after updating the root directory:

1. Share the new build logs from Railway
2. Check if railway.toml settings are being respected

3. Verify environment variables are set correctly
 4. Ensure PostgreSQL database is connected
-

Generated: October 24, 2025

Status:  Fix committed and pushed to GitHub

Commit: 545f41b - fix: Add root directory configuration to railway.toml