Dockerfile Comprehensive Fix - Cannot find module '/bin/bash'

Problem Analysis

Root Cause

The error "Cannot find module '/bin/bash'" occurred because the Dockerfile was using an incorrect `CMD` instruction:

```
CMD ["/bin/bash", "./docker-entrypoint.sh"]
```

This caused Docker to try to execute `/bin/bash` through Node.js's module system, which is fundamentally wrong.

Why This Happened

1. **Incorrect CMD syntax**: Using `CMD ["/bin/bash", "./docker-entrypoint.sh"]` tells Docker to run bash as the main process
2. **Process chain issue**: The entrypoint script ends with `exec node server.js`, which should re-place the shell with Node.js
3. **Signal handling**: Using CMD instead of ENTRYPOINT prevents proper signal handling and process replacement

Solution

The Fix

Changed from:

```
CMD ["/bin/bash", "./docker-entrypoint.sh"]
```

To:

```
ENTRYPOINT ["./docker-entrypoint.sh"]
```

Why This Works

1. **Proper shebang handling**: The script has `#!/bin/bash` at the top, so Docker will automatically use bash to execute it
2. **Process replacement**: The script ends with `exec node server.js`, which replaces the bash process with Node.js
3. **Signal handling**: Using ENTRYPOINT ensures proper signal forwarding (SIGTERM, SIGINT) to the Node.js process
4. **Executable permissions**: The Dockerfile already sets `chmod +x docker-entrypoint.sh` on line 79

Verification

Dockerfile Structure (Correct)

```dockerfile
# Multi-stage build
FROM node:18-alpine AS base
WORKDIR /app

FROM base AS deps
# Install dependencies

FROM base AS builder
# Build Next.js with standalone output

FROM base AS runner
WORKDIR /app
# Copy standalone output from /app/.next/standalone/app to /app
COPY --from=builder --chown=nextjs:nodejs /app/.next/standalone/app ./
# Copy entrypoint script
COPY --chown=nextjs:nodejs docker-entrypoint.sh ./
RUN chmod +x docker-entrypoint.sh
# Use ENTRYPOINT for proper process handling
ENTRYPOINT ["./docker-entrypoint.sh"]
```

Entrypoint Script Flow (Correct)

```bash
#!/bin/bash
set -euo pipefail

# 1. Validate DATABASE_URL
# 2. Check PostgreSQL connectivity
# 3. Run migrations
# 4. Generate Prisma client
# 5. Run seed (if needed)
# 6. Configure master password
# 7. Start Next.js server

# Final line - replaces shell with Node.js
exec node server.js
```

Key Differences: CMD vs ENTRYPOINT

CMD ["/bin/bash", "./script.sh"]

- ❌ Runs bash as the main process
- ❌ Script runs as a child process
- ❌ Signals don't propagate correctly
- ❌ Process doesn't get replaced by `exec`

ENTRYPOINT ["./script.sh"]

- ✅ Script runs as the main process (PID 1)
- ✅ Shebang (`#!/bin/bash`) is respected
- ✅ Signals propagate correctly
- ✅ `exec node server.js` replaces the shell with Node.js

Testing Checklist

After deploying this fix, verify:

1. ✅ Container starts without "Cannot find module '/bin/bash'" error
2. ✅ Entrypoint script runs and shows initialization logs
3. ✅ PostgreSQL connection is established
4. ✅ Migrations run successfully
5. ✅ Prisma client is generated
6. ✅ Next.js server starts on port 3000
7. ✅ Application is accessible via browser
8. ✅ Container stops gracefully with SIGTERM

Related Files

- `Dockerfile` - Fixed CMD → ENTRYPOINT
- `docker-entrypoint.sh` - Already correct with `exec node server.js`
- `app/package.json` - No changes needed

Additional Notes

Why Previous PRs Didn't Work

- PR #32-39: Made various changes but didn't address the fundamental CMD/ENTRYPOINT issue
- PR #40: Restored working Dockerfile from PR #30, but still had the CMD issue
- PR #41: Added documentation but didn't fix the actual problem

This Fix is Minimal and Surgical

- Only changes the CMD → ENTRYPOINT instruction
- No other modifications to Dockerfile structure
- No changes to entrypoint script (already correct)
- No changes to Next.js configuration

Deployment Instructions

1. Merge this PR
2. Easypanel will automatically rebuild the image
3. The new container will start correctly
4. Monitor logs to verify initialization completes
5. Update DATABASE_URL if connectivity issues persist (see PR #41 documentation)

Success Criteria

✅ Container starts without module errors
✅ Entrypoint script executes completely
✅ Next.js server runs on port 3000
✅ Application is accessible and functional