









Next Steps for Drizzle Migration

What's Been Completed




Infrastructure (100% Complete)

-  Drizzle schema with all 33 tables (`src/db/schema.ts`)
-  Comprehensive logging system (`src/lib/logger.ts`)
-  Database helper functions (`src/lib/db-helpers.ts`)
-  Migration guide (`DRIZZLE_MIGRATION_GUIDE.md`)
-  Migration tracking document (`MIGRATION_PROGRESS.md`)
-  Migration helper scripts (`migrate-all-to-drizzle.sh` , `auto-migrate.py`)

Critical Routes (2 files migrated)

-  `src/app/api/audio-processor/test-connection/route.ts`
- Fixed SQLite binding errors
- Added comprehensive logging
- Proper type checking for processorId
-  `src/app/api/audio-processor/route.ts`
- Full CRUD operations (GET, POST, PUT, DELETE)
- Complete API logging
- Error handling with detailed messages

Build Status

-  **Application compiles successfully**
-  All syntax errors fixed
-  No Drizzle-related errors

What Needs to Be Done Next

Immediate Priority - Audio Processor Routes (8 files)

These are the most critical routes that need migration next:

1. `src/app/api/audio-processor/[id]/zones-status/route.ts` - **HIGH PRIORITY**
 - Currently has runtime errors shown in logs
 - Critical for zone status monitoring
2. `src/app/api/audio-processor/zones/route.ts`
 - Zone management
3. `src/app/api/audio-processor/inputs/route.ts`
 - Input management

4. `src/app/api/audio-processor/outputs/route.ts`
- Output management
5. `src/app/api/audio-processor/control/route.ts`
- Control commands
6. `src/app/api/audio-processor/[id]/input-gain/route.ts`
- Gain control
7. `src/app/api/audio-processor/[id]/ai-gain-control/route.ts`
- AI-based gain control
8. `src/app/api/audio-processor/input-levels/route.ts`
- Level monitoring
9. `src/app/api/audio-processor/matrix-routing/route.ts`
- Matrix routing
10. `src/app/api/audio-processor/meter-status/route.ts`
 - Meter monitoring

Step-by-Step Migration Process

For Each File:

1. **Read the file** to understand its operations

2. **Update imports:**

```
``typescript
// Remove
import { prisma } from '@lib/db'

// Add
import { schema } from '@db'
import { logger } from '@lib/logger'
import { findMany, create, update, deleteRecord, eq, and, desc, asc } from '@lib/db-helpers'
...

```

1. **Add API logging** at start of each HTTP method:

```
typescript
export async function GET(request: NextRequest) {
  logger.api.request('GET', '/api/your-route')
  // ... rest of code
}
```

2. **Convert Prisma operations:**

- `prisma.table.findMany()` → `findMany('tableName', {})`
- `prisma.table.create()` → `create('tableName', {})`
- `prisma.table.update()` → `update('tableName', eq(schema.tableName.id, id), {})`
- `prisma.table.delete()` → `deleteRecord('tableName', eq(schema.tableName.id, id))`

3. **Fix Date objects:**

- Change `new Date()` to `new Date().toISOString()` when assigning to database fields

4. Add response logging:

```
typescript
logger.api.response('GET', '/api/your-route', 200, { count: results.length })
return NextResponse.json({ results })
```

5. Add error logging:

```
typescript
catch (error: any) {
  logger.api.error('GET', '/api/your-route', error)
  return NextResponse.json({ error: 'Failed', details: error.message }, { status: 500 })
}
```

6. **Test** by running `npm run build` to ensure no compilation errors

7. **Commit** with descriptive message

Example Migration

See these files as reference:

- `src/app/api/audio-processor/test-connection/route.ts` - Complex route with multiple updates
 - `src/app/api/audio-processor/route.ts` - Full CRUD operations
 - `src/app/api/schedules/route.ts` - Simple CRUD with calculated fields
 - `src/app/api/home-teams/route.ts` - Multiple orderBy example
-

Table Name Reference (Quick)

Prisma	Drizzle
audioProcessor	'audioProcessors'
audioZone	'audioZones'
channelPreset	'channelPresets'
schedule	'schedules'
matrixConfiguration	'matrixConfigurations'
matrixInput	'matrixInputs'
matrixOutput	'matrixOutputs'
irDevice	'irDevices'
globalCacheDevice	'globalCacheDevices'
todo	'todos'

See `MIGRATION_PROGRESS.md` for complete table mapping.

Testing Commands

```
# Check for remaining Prisma usage
./migrate-all-to-drizzle.sh

# Build and test
npm run build

# Open database GUI
npm run db:studio

# Commit changes
git add -A
git commit -m "feat: Migrate [route-name] to Drizzle ORM"
git push origin main
```

Progress Tracking

Current: **2 of 108 files (1.9%)**

Update `MIGRATION_PROGRESS.md` as you complete each file:

- Change `[]` to `[x]` for completed files
 - Update the completion percentage
 - Add any notes about issues encountered
-

Getting Help

If you encounter issues:

1. **Check the migration guide:** `DRIZZLE_MIGRATION_GUIDE.md`
 2. **Reference completed examples** in the same directory
 3. **Check schema:** `src/db/schema.ts` for table structures
 4. **Review db-helpers:** `src/lib/db-helpers.ts` for available functions
-

Common Pitfalls to Avoid

1. ❌ Don't forget to convert table names (e.g., `audioProcessor` → `'audioProcessors'`)
 2. ❌ Don't forget to convert `new Date()` to `new Date().toISOString()`
 3. ❌ Don't forget to add type checking for IDs: `if (id && typeof id === 'string')`
 4. ❌ Don't forget to add API logging (request, response, error)
 5. ❌ Don't forget to change `console.error` to `logger.api.error`
-

Final Steps (After All Files Migrated)

1. Remove Prisma completely:

```
bash
rm src/db/prisma-adapter.ts
rm src/lib/prisma.ts
rm prisma/schema.prisma.deprecated
npm uninstall @prisma/client prisma
```

2. Final build test:

```
bash
npm run build
npm start
```

3. Verify no Prisma references:

```
bash
grep -r "prisma\" src/ --include="*.ts" --include="*.tsx"
grep -r "@prisma/client" src/ --include="*.ts" --include="*.tsx"
```

4. Final commit:

```
bash
git add -A
git commit -m "feat: Complete Prisma to Drizzle ORM migration - Remove all Prisma dependencies"
git push origin main
```

Time Estimate

- **Immediate Priority (10 audio-processor files):** 2-3 hours
- **High Priority (channel-presets, matrix, todos - 16 files):** 3-4 hours
- **Medium Priority (remaining API routes - 57 files):** 10-12 hours
- **Service Files (20 files):** 4-5 hours
- **Testing & Cleanup:** 1-2 hours

Total Estimated Time: 20-26 hours

With the semi-automated script and established patterns, this can be reduced significantly.

Questions?

Refer to:

- **DRIZZLE_MIGRATION_GUIDE.md** - Complete guide with examples
- **MIGRATION_PROGRESS.md** - Current status and file lists
- **Completed examples** in `src/app/api/` directories