Database Setup Guide - SQLite3

1. Installation

Install SQLite3 for Node.js:

```bash
npm install sqlite3
npm install sqlite  # Promise wrapper (recommended)
```

Verify Installation:

```bash
# Check if sqlite3 command is available
sqlite3 --version

# If not installed system-wide (optional but useful):
# macOS:
brew install sqlite3

# Ubuntu/Debian:
sudo apt-get install sqlite3

# Windows:
# Download from https://www.sqlite.org/download.html
```

2. Directory Structure

```
project/
├── db/
│   ├── schema.sql          # Database schema
│   ├── diet.db            # SQLite database file (created by init)
│   └── backups/           # Backup files directory
├── lib/
│   └── db/
│       ├── Database.js      # KEEP - Database connection manager
│       ├── init.js         # KEEP - Initialize/reset database
│       ├── Populate.js      # THROWAWAY - Migrate JSON to DB
│       ├── Backup.js       # KEEP - Backup utilities
│       └── test.js         # THROWAWAY - Test database connection
│
```

3. Usage

Initialize Database (First Time):

```
bash

node lib/db/init.js
```

Test Connection:

```
bash

node lib/db/test.js
```

Populate from JSON (One Time):

```
bash

node lib/db/Populate.js
```

Backup Database:

```
bash

# Manual backup
node lib/db/Backup.js

# Or via code
npm run db:backup
```

Reset Database (Development):

```
bash

node lib/db/init.js --reset
```

4. Backup Strategy

Automatic Backups:

- **Option A:** Cron job (Linux/Mac)

```
bash
```

bash

```bash
# Edit crontab
crontab -e

# Add daily backup at 2 AM
0 2 * * * cd /path/to/project && node lib/db/Backup.js
```

- **Option B:** Node-cron in application

javascript

```javascript
// In app.js
import cron from 'node-cron';
import { createBackup } from './lib/db/Backup.js';

// Daily backup at 2 AM
cron.schedule('0 2 * * *', () => {
  createBackup();
});
```

- **Option C:** Git-based

bash

```bash
# Add to .gitignore
db/diet.db

# But commit backups
db/backups/*.db
```

Manual Backups:

bash

```bash
# Simple copy
cp db/diet.db db/backups/diet_$(date +%Y%m%d_%H%M%S).db

# Or use the utility
node lib/db/Backup.js
```

5. NPM Scripts (Add to package.json)

json

```
{
  "scripts": {
    "db:init": "node lib/db/init.js",
    "db:reset": "node lib/db/init.js --reset",
    "db:test": "node lib/db/test.js",
    "db:populate": "node lib/db/Populate.js",
    "db:backup": "node lib/db/Backup.js",
    "db:restore": "node lib/db/Backup.js --restore"
  }
}
```

Then use:

```bash
npm run db:init
npm run db:test
npm run db:populate
npm run db:backup
npm run db:reset
```

6. Development Workflow

First Time Setup:

```bash
# 1. Initialize database
npm run db:init

# 2. Test connection
npm run db:test

# 3. Populate with JSON data
npm run db:populate

# 4. Verify data
sqlite3 db/diet.db "SELECT COUNT(*) FROM brands;"
```

Daily Development:

```bash
```

```
# Reset and repopulate (if needed)
npm run db:reset
npm run db:populate

# Backup before major changes
npm run db:backup
```

Before Deployment:

```bash
# Create final backup
npm run db:backup

# Test everything
npm run db:test
npm test
```

7. Backup Retention Policy

Suggested:

- Keep daily backups for 7 days
- Keep weekly backups for 1 month
- Keep monthly backups for 1 year

Implement in Backup.js with cleanup logic