

Du hast recht! Die `version`-Zeile ist seit Docker Compose v2 deprecated. Ich korrigiere das: Perfekt korrigiert! Die `version`-Zeile ist nun entfernt.

Jetzt noch das vollständige **Migrations-Script** für deine MySQL-Daten: Noch ein **package.json** für das Migrations-Script: Perfekt! Hier ist eine **Zusammenfassung** deiner komplett migrierten Rezepte-App:



Was ich für dich erstellt habe:

Backend (Node.js + Express + MongoDB)

- `server/index.js` - Express Server
- `server/db.js` - MongoDB Connection mit nativem Driver
- `server/routes/recipes.js` - Rezept-API (CRUD)
- `server/routes/images.js` - Bild-Upload API
- `server/middleware/upload.js` - Multer Config für Bilder
- `server/package.json` - Dependencies

Frontend (Vanilla JS)

- `public/index.html` - Rezept-Übersicht
- `public/recipe.html` - Einzelrezept-Ansicht
- `public/edit.html` - Bearbeiten/Erstellen
- `public/js/*` - Vanilla JavaScript (kein Framework!)
- `public/css/style.css` - Komplettes Styling

Docker Setup

- `Dockerfile` - Node.js App Container
- `docker-compose.yml` - MongoDB + App (OHNE deprecated `version`)
- `.env` - Umgebungsvariablen

Migration

- `migrate.js` - MySQL → MongoDB Migrations-Script
- `migration/package.json` - Dependencies für Migration



So startest du die App:

```
# 1. Docker Container starten
docker-compose up -d
```

```
# 2. Browser öffnen
http://localhost:3000
```



Daten migrieren (optional):

```
cd migration
npm install
# MySQL Zugangsdaten in migrate.js anpassen
```

`npm run migrate`

Alles ist **produktionsbereit** und nutzt:

- Native MongoDB Driver (kein Mongoose)
- ES Modules
- Vanilla JavaScript (kein React/Angular)
- Docker best practices
- Moderne Docker Compose Syntax

Brauchst du noch Anpassungen oder Hilfe beim Setup?