Eindopdracht Backend
Thomas Oudhoff
2025/02

Medicatie inname App

**Inhoudsopgave**

**Inleiding**

Deze handleiding legt uit hoe je de Medicatie Inname App lokaal draait met MySQL, hoe de seed-data geladen wordt en hoe je de API test met Postman. Ook vind je een overzicht van de belangrijkste endpoints en veelvoorkomende oplossingen.

**Benodigdheden**

- Java 21 (JDK)
- Maven 3.9 of hoger
- MySQL 8.x (lokaal)
- Postman 11.x (optioneel, voor handmatige tests)

**Installatie instructies (stappenplan)**

**Database aanmaken**

Maak lokaal een lege database aan met de naam: **medicijneninname**.
(Indien je een andere naam kiest, pas dan de datasource-URL in de configuratie aan.)

**Applicatie configureren**

Open het bestand: **src/main/resources/application.properties** en controleer de datasource-instellingen:

spring.datasource.url = jdbc:mysql://localhost:3306/medicijneninname?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC

spring.datasource.username = jouw MySQL-gebruikersnaam

spring.datasource.password = jouw MySQL-wachtwoord

spring.jpa.hibernate.ddl-auto = update

spring.sql.init.mode = always

spring.jpa.defer-datasource-initialization = true

Let op: de seed (**data.sql**) gebruikt {bcrypt}-geprefixte hashes. Dit werkt samen met de **DelegatingPasswordEncoder** die in de app is geconfigureerd.

**Build & run**

Optie A (ontwikkelen): draai met Maven: **mvn spring-boot:run**.
Optie B (jar): bouw met **mvn clean package** en start daarna het jar in **target**.
Wanneer de app draait, is de base URL: **http://localhost:8080**.

**Seed-data (automatisch)**

Bij de start voert de applicatie **src/main/resources/data.sql** uit.
Hiermee worden demo-gebruikers, voorbeeld-medicatie, zorgrelaties en een demo-bijsluiter geladen.

**Testgebruikers (uit de seed)**

- **ADMIN** — Email: admin@example.com — Wachtwoord: Admin123!
- **VERZORGER** — Email: verzorger1@example.com — Wachtwoord: admin123!
- **GEBRUIKER** — Email: patient1@example.com — Wachtwoord: (optioneel in seed)
- **GEBRUIKER** — Email: patient2@example.com — Wachtwoord: (optioneel in seed)

Tip: demo-flows gebruiken vooral **admin@example.com** en **verzorger1@example.com**.
Patiënten kun je als ADMIN aanmaken of een wachtwoord geven via de API.

**Postman collecties**

De Postman-bestanden staan in de repo onder **/postman**:

- postman/medicatieinname.postman_collection.json
- postman/medicatieinname.local.postman_environment.json

**Importeren en instellen**

1. Open Postman en importeer beide JSON-bestanden.
2. Selecteer rechtsboven het environment: **medicatieinname.local**.
3. Controleer de variabelen:
    - baseUrl = http://localhost:8080
    - Admin: authUser = admin@example.com, authPass = Admin123!
    - Verzorger: verzorgUser = verzorger1@example.com, verzorgPass = admin123!
    - (Seed) gebruikerId = 2, gebruikerId2 = 3
4. Voer de flow (1–12) in de collectie uit: Admin-stappen, Verzorger-stappen en Self-service Patiënt-stappen.

**Upload tip:** Bij upload van een bijsluiter gebruik je in Postman **Body → form-data** met key **file** van type **File**. Voeg géén eigen Content-Type header toe.

**REST-endpoints (selectie + toelichting)**

**Authenticatie:** HTTP Basic.
**401** = geen/onjuiste login. **403** = ingelogd maar geen rechten.

**Gebruikers & relaties**

- GET /api/gebruiker — alle gebruikers (alleen ADMIN)
- GET /api/gebruiker/{id} — detail (ADMIN of de gebruiker zelf)
- POST /api/relaties — (ADMIN) koppel verzorger ↔ patiënt (body met verzorgerId en gebruikerId)

**Medicatie (ADMIN/owner; verzorger read-only indien gekoppeld)**

- POST /api/medicatie/gebruiker/{gebruikerId} — medicatie aanmaken voor gebruiker
- GET /api/medicatie/{id} — medicatie detail
- PUT /api/medicatie/{id} — medicatie bijwerken
- DELETE /api/medicatie/{id} — medicatie verwijderen

**Bijsluiter**

- POST /api/medicatie/{id}/bijsluiter — upload (multipart file)
- GET /api/medicatie/{id}/bijsluiter — download
- PUT /api/medicatie/{id}/bijsluiter-url — externe URL registreren
- DELETE /api/medicatie/{id}/bijsluiter — verwijderen

**Schema & Toedieningen**

- POST /api/medicatie/{medicatieId}/schema — innameschema aanmaken (bijv. start/einde en dagtijden)
- GET /api/medicatie/{medicatieId}/schema — schema's opvragen
- POST /api/schema/{schemaId}/toedieningen — toediening registreren (tijdstip, hoeveelheid, opmerking)
- GET /api/schema/{schemaId}/toedieningen — toedieningen per schema
- GET /api/medicatie/{id}/toedieningen?from=YYYY-MM-DD&to=YYYY-MM-DD — filter op medicatie
- GET /api/gebruiker/{id}/toedieningen?from=YYYY-MM-DD&to=YYYY-MM-DD — filter op gebruiker

**Notificatie-instellingen (owner/ADMIN)**

- PUT /api/gebruiker/{id}/instellingen — upsert (kanaal, minuten vooraf, snooze, stille tijden, actief)
- GET /api/gebruiker/{id}/instellingen — ophalen
- DELETE /api/gebruiker/{id}/instellingen — verwijderen
  Let op: het kanaal is lowercase: **email** of **push**.

**Overige commando's**

- Tests draaien met Maven: **mvn test**
- Zip maken van repo-inhoud (zonder target/.idea): bijv. met **git archive**
- Seed opnieuw laten draaien: zet **spring.sql.init.mode** op **always** en leeg zo nodig de relevante tabellen of de database

**Veelvoorkomende issues**

- 401 overal → controleer Basic Auth en dat de {bcrypt}-hashes vanuit **data.sql** in de database staan.
- Upload 415/500 → gebruik in Postman **form-data** met key **file** (type **File**), zonder eigen Content-Type.
- Verzorger ziet geen data → leg eerst de zorgrelatie via **POST /api/relaties**.