

# Anwendung

Das Programm ist eine API (Application Programming Interface) REST (Representational State Transfer Application Program Interface) entwickelt mit der Technologie : Spring Boot, H2, JPA, java SE 11

Die Datei **MPD.pdf** stellt das Physikalische Modell der Daten (MPD) von allen Entitäten dar, die in Programmierung implementiert wurden.

Die Datei **tp\_apotheke.jar** ist das Programm. Dies enthält alle Quellexecoden.

Die Beschreibung der Klassen und Methoden in diesem Rahmen des Projektes befindet sich in **JavaDoc**.

Die Datei **API\_ROUTES.txt** beschreibt alle verwendeten Wege abhängig von Methoden http.

## Ausführen und testen

- 1) führen die Applikation Spring Boot mit dem Befehl : **mvn spring-boot:run** aus oder führen die Datei **tp\_apotheke.jar** dar
- 2) Die Tabelle der DB werden automatisch in der Speicherung generieren
- 3) Öffnen die Konsole H2 mit url : **http://localhost:8080/h2-ui**
- 4) Öffnen **POSTMAN** (zum API testen). Die Wege für alle Entitäten sind :

<http://localhost:8080/api/fabricant>

<http://localhost:8080/api/entrepot>

<http://localhost:8080/api/pharmacieSuccursale>

<http://localhost:8080/api/medicament>

<http://localhost:8080/api/commander>

<http://localhost:8080/api/stoker>

Beispiel einer Seite mit : **http://localhost:8080/h2-ui** werden aufgerufen. Zugangsparameter auf der DB :

JDBC URL: **jdbc:h2:mem:testdb**

User Name: **sa**

Password:

Einfach klicken auf **CONNECT**, um zu der DB zu gelangen.

← → ↻ localhost:8080/h2-ui/login.jsp?sessionId=6321c6ed9dbd01853faa5dcd68d18df8

English ▾ Preferences Tools Help

### Login

Saved Settings: Generic H2 (Embedded) ▾

Setting Name: Generic H2 (Embedded) Save Remove

---

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:testdb

User Name: sa

Password:

Connect Test Connection

Auto commit: ☒ Max rows: 1000 Auto complete: Off Auto select: On

jdbc:h2:mem:testdb

- APOTHEKE
- FABRIKANT
- LAGER
- MEDIKAMENT
- STOCKAPOTHEKE
- STOCKLAGER
- INFORMATION\_SCHEMA
- Sequences
- Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement

```
SELECT * FROM STOCKAPOTHEKE
```

```
SELECT * FROM STOCKAPOTHEKE
```

ID_APOTHEKE	ID_MEDIKAMENT	QUANTITAET	VERFUEGBARE_APOTHEKE
1	2	35	

(1 row, 2 ms)

Edit

Beispiel einer Einfügung mit POSTMAN



POST http://localhost:8080/api/stockapotheken

Params Authorization Headers (0) Body Pre-request Script Tests Settings Cookies

none form-data www-form-urlencoded raw binary GraphQL JSON

1 {  
2 "id": {}  
3 "id\_apotheka": 1,  
4 "id\_medikament": 1  
5 },  
6 "quantitaet\_vorhanden\_apotheka": 10  
7 }  
8 }

Body Cookies Headers (5) Test Results Status: 201 Created Time: 13 ms Size: 262 B Save Response

Pretty Raw Preview Visualize JSON

1 {  
2 "id": {}  
3 "id\_apotheka": 1,  
4 "id\_medikament": 1  
5 },  
6 "quantitaet\_vorhanden\_apotheka": 10  
7 }  
8 }

PUT http://localhost:8080/api/stockapotheken/1

Params Authorization Headers (0) Body Pre-request Script Tests Settings Cookies

none form-data www-form-urlencoded raw binary GraphQL JSON

1 {  
2 "id": {}  
3 "id\_apotheka": 1,  
4 "id\_medikament": 1  
5 },  
6 "quantitaet\_vorhanden\_apotheka": 10  
7 }  
8 }

Body Cookies Headers (5) Test Results Status: 204 OK Time: 40 ms Size: 318.07 KB Save Response

Pretty Raw Preview Visualize JSON

1 {  
2 "id": {}  
3 "id\_apotheka": 1,  
4 "id\_medikament": 1  
5 },  
6 "medikament": {  
7 "stockapotheken": {  
8 {  
9 "id": {}  
10 "id\_apotheka": 1,  
11 "id\_medikament": 1  
12 },  
13 "medikament": {  
14 "stockapotheken": {  
15 {  
16 "id": {}  
17 }  
18 }  
19 }  
20 }  
21 }  
22 }