Terminverwaltung App Projekt

Programmierung Java 2

21. Mai 2019

Ziel

- REST API Server als Backend für eine Web App
- Terminmanagement und Patientenverwaltung

Web App

Angular Projekt für Webaufbau - SS19

Web Server

- Spring Boot
- Spring Framework + Embedded HTTP(s) Servers - XML Configuration oder @Configuration
- Ökosystem von Spring (Spring ORM, Spring Security, ...)

Web Server

- Tomcat als HTTP(s) Server
- Endpoints (Pfade):
  - /api/{model}   (GET/POST)
  - /api/{model}/{id}   (PUT/DELETE)
- Request Type: "application/x-www-form-urlencoded"
- Response Type: "application/json"
- Authorization: Basic Auth

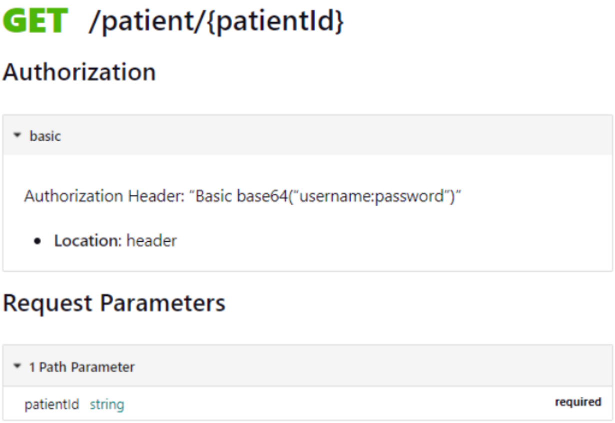
Datenbank

- MySQL als Datenbank
    - MySQL Connector
- Spring Data JPA als ORM
    - Hibernate/JPA
    - Entity, Repository, ...

Modelle

- User
- Patient
- Address
- Appointment
- AppointmentRecord
- Prescription
- Medicine
- Disease

Beispiel



GET /patient/{patientId}

Authorization

▾ basic

Authorization Header: “Basic base64(“username:password”)”

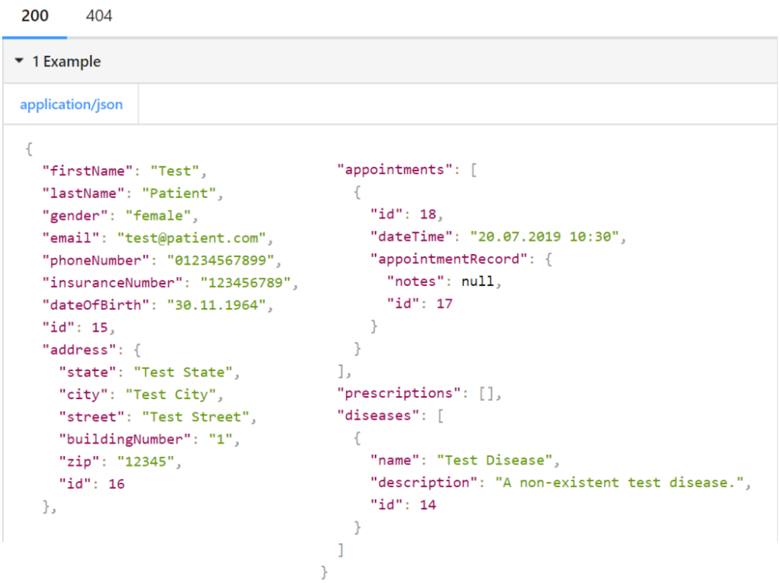
- **Location:** header

Request Parameters

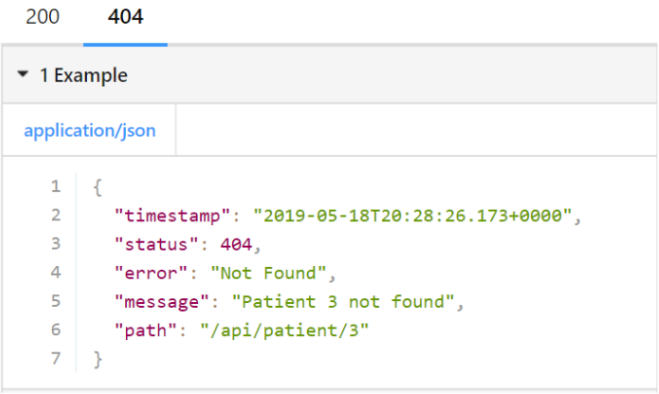
▾ 1 Path Parameter

patientId string **required**

```
curl --request GET \
  --url http://localhost:8080/api/patient/15 \
  --header 'authorization: Basic dGVzdHVzZXI6dGVzdHBhc3N3b3Jk'
```



```
{
  "firstName": "Test",
  "lastName": "Patient",
  "gender": "female",
  "email": "test@patient.com",
  "phoneNumber": "01234567899",
  "insuranceNumber": "123456789",
  "dateOfBirth": "30.11.1964",
  "id": 15,
  "address": {
    "state": "Test State",
    "city": "Test City",
    "street": "Test Street",
    "buildingNumber": "1",
    "zip": "12345",
    "id": 16
  },
  "appointments": [
    {
      "id": 18,
      "dateTime": "20.07.2019 10:30",
      "appointmentRecord": {
        "notes": null,
        "id": 17
      }
    }
  ],
  "prescriptions": [],
  "diseases": [
    {
      "name": "Test Disease",
      "description": "A non-existent test disease.",
      "id": 14
    }
  ]
}
```



```json
{
  "timestamp": "2019-05-18T20:28:26.173+0000",
  "status": 404,
  "error": "Not Found",
  "message": "Patient 3 not found",
  "path": "/api/patient/3"
}
```

TODO

- Effizientere Datenabfragen (Extra Pfade mit Filtern)
- Validierung (Hibernate Validator statt MySQL Fehler)
- Komplette API-Dokumentation (Swagger - OpenAPIv2)