# LU07.A02 - Bookshelf testen mit Bruno

Teste die API der Bookshelf-Applikation mit Postman.

## Bookshelf-Applikation

Die Webapplikation „Bookshelf“ stellt eine API zur Verwaltung eines Bücherregals bereit. Über diese API kann ein Programm die Bücher lesen, erstellen, ändern und löschen.

### Dokumentation der API

Die API ist unter `https://it.bzz.ch/book/FUNKTION` erreichbar. Der Pfad der einzelnen Funktionen wird jeweils hinten angehängt. Also z.B.: `https://it.bzz.ch/book/list`.

#### Buch lesen

Liest ein einzelnes Buch anhand der UUID.

- Pfad: `read/<book_uuid>`
- HTTP-Methode: GET
- Response: Buch als JSON-Struktur
- HTTP-Status:
  - 200 = Buch gefunden
  - 404 = Kein Buch gefunden

#### Buch löschen

Löscht ein einzelnes Buch anhand der UUID.

- Pfad: `delete/<book_uuid>`
- HTTP-Methode: DELETE
- Response: leer
- HTTP-Status:
  - 200 = Buch gelöscht
  - 404 = Kein Buch gefunden

#### Buch speichern

Speichert ein Buch mit den angegebenen Formulardaten. Falls eine bekannte Buch-UUID mitgegeben wird, wird das Buch geändert. Andernfalls wird ein neues Buch erstellt.

- Pfad: `save`

- HTTP-Methode: POST
- Daten (x-www-form-urlencoded):
    - book_uuid (optional)
    - title
    - author
    - price
    - format
    - isbn
- Reponse: leer
- HTTP-Status
    - 200 = Bestehendes Buch wurde aktualisiert.
    - 201 = Neues Buch wurde eingefügt.
    - 400 = Daten fehlen oder sind formal ungültig
    - 422 = Daten entsprechen nicht den Validierungsregeln

## Validierungsregeln

| Datenfeld | Datentyp | Regeln |
|---|---|---|
| book_uuid | UUIDv4 | Die UUID eines Buchs. |
| title | String | 2-50 Zeichen, alle Zeichen der ASCII-Codetabelle gemäss ISO-8859-1. |
| author | String | 4-100 Zeichen, alle Zeichen der ASCII-Codetabelle gemäss ISO-8859-1. |
| price | Decimal | 0.00 - 299.95 mit maximal 2 Nachkommastellen |
| isbn | String | ISBN-13 Nummer |
| format | Auswahl | Hardcover, Softcover, eBook oder Audio |

## Testdaten

Diese Testdaten stehen in der API zur Verfügung.

```
[
  {
    "book_uuid": "c746a291-0ef9-4b2a-8268-392b12d636bd",
    "title": "The Winds of Winter",
    "author": "George R R Martin",
    "price": 52.75,
    "format": "Hardcover",
    "isbn": "978-0-1256-0432-1"
  },
  {
    "book_uuid": "d82bf28e-eb73-45c9-9ce3-1f94a599a0b5",
    "title": "The Shadow of What Was Lost",
    "author": "James Islington",
    "price": 15.3,
    "format": "eBook",
    "isbn": "978-0-3162-7407-4"
  },
  {
    "book_uuid": "3247c340-c712-402e-b400-0a23a9368c97",
```

```
    "title": "An Echo Of Things To Come",
    "author": "James Islington",
    "price": 32.5,
    "format": "",
    "isbn": "978-0-3162-7409-8"
  }
]
```

# Aufträge

## Testfälle (Requests) erstellen

Erstelle die Testfälle mit **Bruno** zu jeder API-Funktion.

- Für jeden möglichen HTTP-Statuscode muss ein eigener Request angelegt werden (z. B. `read_200.bru`, `read_404.bru`).
- Zum Testen der Validierung der Daten, müssen Testfälle gemäss Äquivalenzklassen-Analyse erstellt werden.

## Beispiel: read

Falls die Funktion `read` ein Buch findet, liefert sie die Daten des Buchs und den HTTP-Status 200. Andernfalls antwortet die Funktion mit dem HTTP-Status 404. Du brauchst also 2 Requests für diese Funktion.

## Tests durchführen

Führe die Tests manuell aus (Request anklicken → „Send“). Betrachte die Resultate (Daten, HTTP-Statuscode).

Verwende danach den **Collection Runner** in Bruno, um alle Requests automatisch auszuführen. Studiere den Output des Collection Runners.

# Abgabe

Exportiere dein gesamtes Bruno-Projekt (Ordner mit allen `.bru`-Dateien und Environments). Packe den Ordner als ZIP-Datei und lade ihn hier in Moodle hoch.

---

M450-LU07

Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
**https://wiki.bzz.ch/modul/m450/learningunits/lu07/aufgaben/bookshelfbruno?rev=1758179180**

Last update: **2025/09/18 09:06**