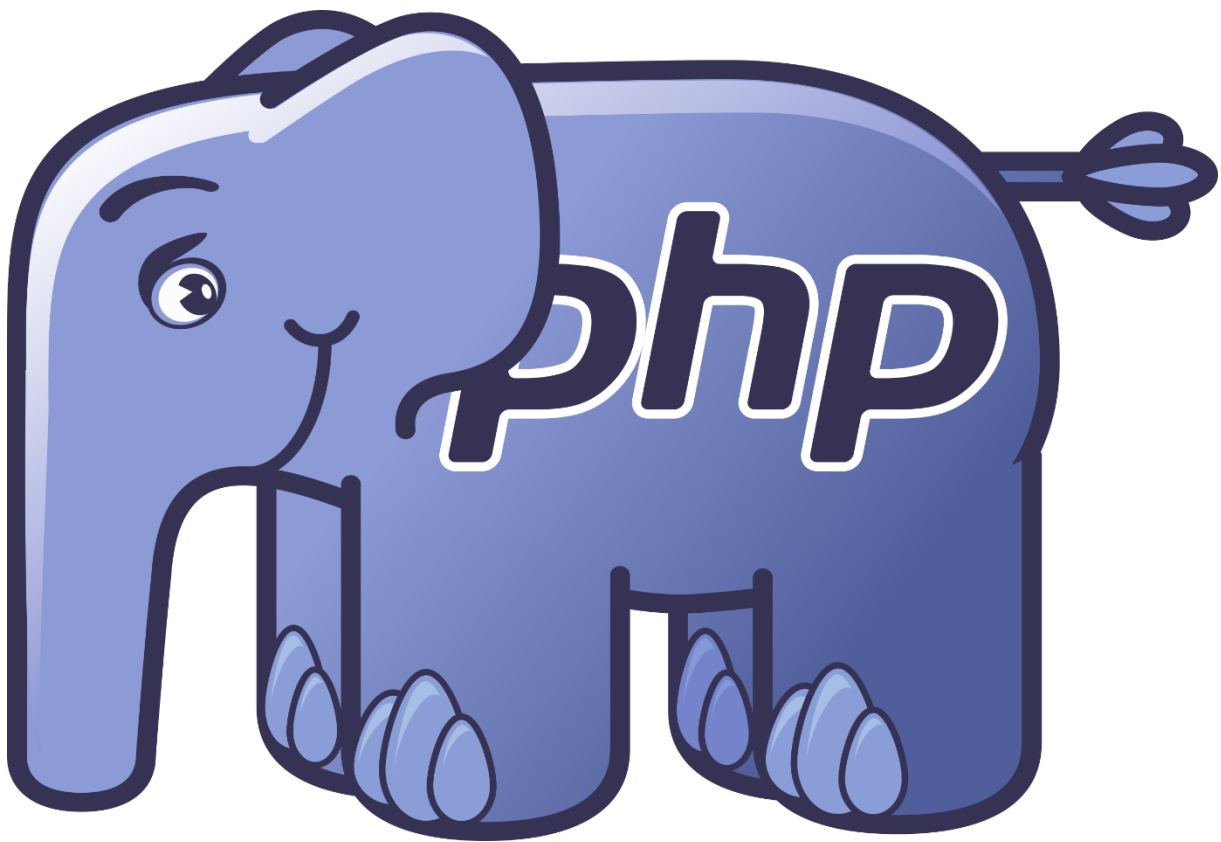


# M295 – Backend für Applikationen realisieren

Raul Faria – 2. Lehrjahr

1



---

<sup>1</sup> Quelle : [Wikipedia](#)

# Inhalt

---

|                                  |    |
|----------------------------------|----|
| Überblick über das Projekt ..... | 3  |
| Ziel des ÜK .....                | 3  |
| Schritte zum Ziel.....           | 3  |
| Vorwort.....                     | 3  |
| Schnittstellendokumentation..... | 4  |
| Datenbankschema .....            | 5  |
| HTTP- Client Vergleich.....      | 6  |
| Git- Version control .....       | 7  |
| Commit- Vorgehen.....            | 7  |
| Commit- History.....             | 8  |
| Tests .....                      | 9  |
| Konzept .....                    | 9  |
| Berichte.....                    | 10 |
| Schlusswort.....                 | 11 |

# Überblick über das Projekt

---

## Ziel des ÜK

- ❑ API für eine Datenbank mit allen benötigten Komponenten wie Eingabevalidierung, Fehlerbehandlung usw. erstellen.
- ❑ Konzepte wie APIs und HTTP Requests näherbringen.  
Die Requests:
  - GET → Daten holen
  - POST → Daten einfügen
  - DELETE → Daten löschen
  - PUT/UPDATE → Daten aktualisieren
- ❑ Übung an Implementierung von Applikationen.

## Schritte zum Ziel

1. Datenbank mit korrekter Struktur erstellen
  - a. Constraints mit 'ON DELETE CASCADE' erstellen
2. Elementare Bestandteile der Applikation bereitstellen:
  - a. Routing
  - b. Datenbankverbindung
3. Rest der Applikation implementieren:
  - a. Eingabevalidierung
  - b. Fehlerbehandlung
  - c. Request- Aktionen
  - d. Home- Route

## Vorwort

Ich war zuversichtlich, dass ich eine funktionierende Lösung implementieren kann, da ich schon andere APIs implementiert habe, zwar alle in TypeScript, doch das Konzept ist etwa das gleiche, und weil ich schon mehrere Web- Applikationen mit PHP entwickelt habe.

Es kamen doch schnell Zweifel auf, da es nicht möglich war, libraries zu verwenden, was das Routing um einiges komplizierter macht. Trotzdem war ich motiviert, diese API zu implementieren.

## Schnittstellendokumentation

Diese OpenAPI 3.0.0 Collection zeigt ein paar Beispiele zur Verwendung der API.

[m295.yaml](#)

Alle verfügbaren Routen sind:

- ☐ /
- ☐ /home
- ☐ /doc/m295.yaml
- ☐ /lehrbetriebe/[ id | spalte ]/[ Bedingung ]
- ☐ /lernende/[ id | spalte ]/[ Bedingung ]
- ☐ /lehrbetrieb\_lernende/[ id | spalte ]/[ Bedingung ]
- ☐ /laender/[ id | spalte ]/[ Bedingung ]
- ☐ /dozenten/[ id | spalte ]/[ Bedingung ]
- ☐ /kurse/[ id | spalte ]/[ Bedingung ]
- ☐ /kurse\_lernende/[ id | spalte ]/[ Bedingung ]

Man hat die Option, alles auszuwählen, ein Element anhand der id auszuwählen, oder eine eigene Kondition mittels einer Spalte und einem Wert.

Beispiele:

Alle Kurse:

/kurse

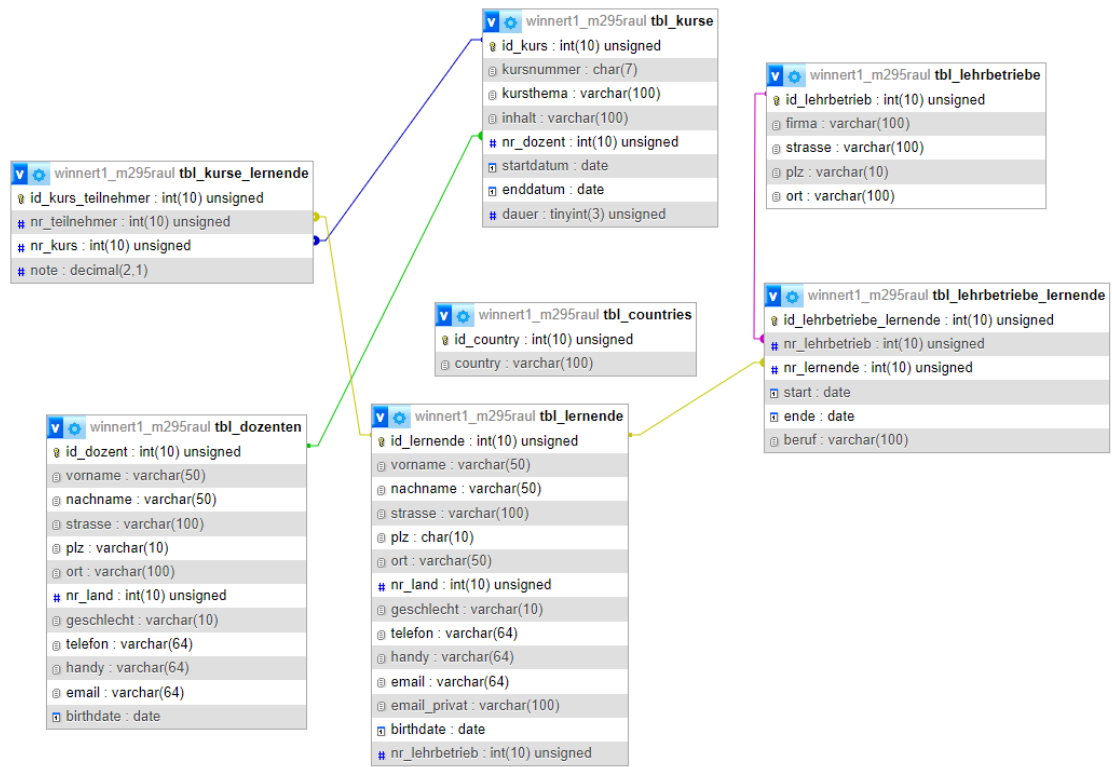
Kurs mit id 38:

/kurse/38

Alle Kurse mit der Kursnummer 999:

/kurse/kursnummer/999

Die Verbindungen sind Constraints mit 'ON CASCADE DELETE'.



## HTTP- Client Vergleich

| Nr.   | Kriterium                | Gewichtung | postman.com |        | curl      |        | PHPStorm built-in |        |
|-------|--------------------------|------------|-------------|--------|-----------|--------|-------------------|--------|
|       |                          |            | Bewertung   | Punkte | Bewertung | Punkte | Bewertung         | Punkte |
| 1     | Bedienungsfreundlichkeit | 3          | 5           | 15     | 0         | 0      | 0                 | 0      |
| 2     | Fokus                    | 2          | 2           | 4      | 5         | 10     | 3                 | 6      |
| 3     | Benötigte Vorkenntnisse  | 2          | 4           | 8      | 0         | 0      | 2                 | 0      |
| 4     | Response- Anzeige        | 4          | 5           | 20     | 0         | 0      | 4                 | 16     |
| Total |                          |            |             | 47     | 10        |        | 24                |        |

Ich habe Postman, Curl und den eingebauten HTTP- Client von PHPStorm verglichen und diese Bewertung erstellt.

Postman hat mit Abstand am besten abgeschlossen und ist für dieses Projekt meiner Meinung nach das Beste, da es wenig Vorkenntnisse benötigt, simpel aufgebaut ist und die Daten intuitiv darstellt.

Curl habe ich integriert, damit ich auch ein Command- Line Tool dabei ist. Curl gibt es schon sehr lange und es ist ein Standard Tool für Linux. Im Grunde ist es ein simpler HTTP- Client. Man kann alle Aktionen, die man mit Postman oder anderen Tools kann, auch. Man muss aber die Optionen dafür kennen, deshalb benötigt man mehr Vorwissen.

PHPStorm erkennt selbstständig HTTP- Routen und kann diese mit einem integrierten Client anfragen. Dabei sieht die Response schöner aus als bei curl, doch das interface ist nicht sehr intuitiv, da es im Grunde nur der Text- Editor von PHPStorm ist.

Das wichtigste Feature hat aber nur Postman, die Möglichkeit automatisch eine Schnittstellendokumentation zu erstellen, deshalb ist in meinem Fall Postman definitiv die beste Wahl.

# Git- Version Control

---

## Commit- Vorgehen

Ich bevorzuge es, mehrere Commits zu machen, anstatt erst zu committen, wenn ein Feature fertig implementiert ist. So hat man mehrere Anhaltspunkte, um den Code auf eine bestimmte Version zurückzusetzen.

Trotzdem befolge ich es, nach jedem fertig implementierten Feature einen Commit zu machen. Normalerweise würde ich dazu ein Issue mit zugehörigem Branch erstellen, und diesen mergen, sobald das Issue aufgelöst ist. So sieht alles viel geordneter aus und man pusht nichts direkt auf den main- Branch. Dies ist bei einem Projekt mit einem so kleinen Umfang wie hier aber nicht sehr sinnvoll.

Ich habe ebenfalls verschiedene Geräte benutzt, was zusätzliche Commits benötigt, damit das Repository auf allen Geräten auf dem neusten Stand ist.

Ich benutze die Standardsprache Englisch, in den inline- Kommentaren sowie bei der Dokumentation. Nur dieses Dokument ist in Deutsch geschrieben.


Ich benutze fast nie die «Amend»- Funktion. Diese bewirkt, dass die Änderung dem letzten Commit hinzugefügt wird, anstatt dass ein neuer Commit erstellt wird. Ich versuche das so wenig wie möglich einzusetzen, dass man jede Änderung genau sieht.

Ich habe es nur ein paarmal eingesetzt, als ich aus Versehen noch eine Linie auskommentiert habe oder in vergleichbaren Fällen.

Ich habe mich dazu entschlossen, bei meiner Applikation eine Versionierung zu implementieren, der Vollständigkeit halber.

Dazu kommt der Modus der Applikation, der entweder DEV oder PROD ist.

## Commit- History

|   |                                  |   |
|---|----------------------------------|---|
| Commits on Dec 11, 2023   |                                  |   |
| updated injection detector to be more robust  | Rfaria06 committed 2 minutes ago | 5fa9365 <a href="#">📄</a> <a href="#">↗</a> |
| Commits on Dec 4, 2023  |                                  |   |
| openapi collection as yaml  | Rfaria06 committed last week     | bf77c33 <a href="#">📄</a> <a href="#">↗</a> |
| Postman collection JSON   | Rfaria06 committed last week     | 5fbf675 <a href="#">📄</a> <a href="#">↗</a> |
| Database schema PDF   | Rfaria06 committed last week     | f5d39a6 <a href="#">📄</a> <a href="#">↗</a> |
| commented out error reporting -> Code now in Production version 1.0.0   | Rfaria06 committed last week     | fb14575 <a href="#">📄</a> <a href="#">↗</a> |
| fixed type error  | Rfaria06 committed last week     | b7f3e2e <a href="#">📄</a> <a href="#">↗</a> |
| implement injection detector for each string sanitizing iteration  | Rfaria06 committed last week     | f4ad800 <a href="#">📄</a> <a href="#">↗</a> |
| Commits on Nov 27, 2023   |                                  |   |
| finished sanitizing   | Rfaria06 committed 2 weeks ago   | 5545503 <a href="#">📄</a> <a href="#">↗</a> |
| finished sanitizing   | Rfaria06 committed 2 weeks ago   | 33fd051 <a href="#">📄</a> <a href="#">↗</a> |
| sanitizer injection detector and date sanitizer   | Rfaria06 committed 2 weeks ago   | 055fba1 <a href="#">📄</a> <a href="#">↗</a> |
| error handling  | Rfaria06 committed 2 weeks ago   | 1823d9c <a href="#">📄</a> <a href="#">↗</a> |
| DELETE allow delete all   | Rfaria06 committed 2 weeks ago   | 30fb4d1 <a href="#">📄</a> <a href="#">↗</a> |
| finished backend  | Rfaria06 committed 2 weeks ago   | 785549b <a href="#">📄</a> <a href="#">↗</a> |
| Commits on Nov 20, 2023   |                                  |   |
| commit for device switch  | Rfaria06 committed 3 weeks ago   | 6a653a8 <a href="#">📄</a> <a href="#">↗</a> |
| test file removed   | Rfaria06 committed 3 weeks ago   | 47a4d68 <a href="#">📄</a> <a href="#">↗</a> |
| Commits on Nov 13, 2023   |                                  |   |
| testing on own database   | Rfaria06 committed last month    | 59af26f <a href="#">📄</a> <a href="#">↗</a> |



# Tests

---

## Konzept

Ich führe mehrere Blackbox- Tests aus.

Es wird klar strukturiert, was getestet wird.

Es werden Angaben zu allgemeinen Daten gemacht, die einen Einfluss auf das Ergebnis des Tests haben könnten.

Diese sind:

- ☐ Version der Applikation
- ☐ Modus der Applikation
- ☐ Sprache
- ☐ Version der Sprache

Der Modus der Applikation beeinflusst das Verhalten, wenn ein Fehler oder eine Ausnahme während der Laufzeit auftritt.

Im DEV- Modus werden Fehler direkt als Stack trace unverändert ausgegeben.

Im PROD- Modus werden Fehler auf eine kleinere Meldung reduziert, die keine Technischen Infos zur Applikation angibt, sondern nur eine JSON- Response mit dem dazugehörigen Fehlercode und einer Meldung.

Bei einem Blackbox- Test wird anders als bei einem Unit- Test nichts automatisiert.

Alle Aktionen werden von Hand ausgeführt. Dabei muss der Tester eng mit der Software vertraut sein, um den Test korrekt auszuführen.

Ich werde ein Excel- Sheet bereitstellen, in dem die Daten eingetragen werden.

Ich werde die im Excel definierten Routen mit Postman abfragen und dieses Ergebnis mit dem Erwarteten vergleichen.

Wenn das erwartete Ergebnis erfüllt wird, fällt der Test positiv aus.

## Berichte

|                    |  |
|--------------------|--|
| <u>Test</u>        |  |
| Titel              | GET - Test   |
| Beschreibung       | Test einer GET - Anfrage   |
| Datum              | 04.12.2023   |
| Vorgehen           | Ein HTTPS- Request wird an den nachfolgenden Pfad geschickt und die Antwort wird mit dem erwarteten Ergebnis verglichen  |
| Pfad               | <a href="https://raul.undefiniert.ch/lernende">https://raul.undefiniert.ch/lernende</a>  |
| <u>Applikation</u> |  |
| Version            | 1.0.0  |
| Modus              | PROD   |
| Sprache            | PHP 8.1  |
| <u>Ergebnis</u>    |  |
| Erwartet           | JSON- Objekt mit einem Array, das 1 oder mehrere Objekte enthält, mit Angaben zu Lehrlingen. Nach der Datenbank sollte nur der Name ausgefüllt sein, alle anderen Daten (NULL) oder 0. |
| Tatsächlich        | Das erwartete Ergebnis wurde erfüllt   |
| Fazit              | Der Test ist positiv ausgefallen   |

|                    |   |
|--------------------|---|
| <u>Test</u>        |   |
| Titel              | 404 Test  |
| Beschreibung       | Test bei Anfrage auf eine Ressource, die nicht vorhanden ist  |
| Datum              | 04.12.2023  |
| Vorgehen           | Ein HTTPS- Request wird an den nachfolgenden Pfad geschickt und die Antwort wird mit dem erwarteten Ergebnis verglichen |
| Pfad               | <a href="https://raul.undefiniert.ch/dieseRouteGibtEsNicht">https://raul.undefiniert.ch/dieseRouteGibtEsNicht</a>       |
| <u>Applikation</u> |   |
| Version            | 1.0.0   |
| Modus              | PROD  |
| Sprache            | PHP 8.1   |
| <u>Ergebnis</u>    |   |
| Erwartet           | Antwort, die den Fehler verständlich macht  |
| Tatsächlich        | JSON- Objekt mit HTTP- Status und Message "Resource not found"  |
| Fazit              | Der Test ist positiv ausgefallen  |

## Schlusswort

---

Ich fand das Projekt sehr interessant.

Man benutzt bei so ziemlich jeder API Libraries, die einem das ganze erleichtern, hier konnten wir selbst sehen, wie solch eine Library auf einem technischen Niveau funktioniert.

Das Programmieren mit PHP war angenehm, da es eine sehr simple Sprache ist, und viele eingebaute Funktionen hat, die bei einem Webserver von riesigem Nutzen sind.

Alles in allem fand ich das Projekt sehr lehrreich und ich war motiviert, die Applikation zu implementieren.