

# LB03 - Projektauftrag

## Einleitung

In den ersten zwei Dritteln des Moduls **M290** haben Sie sich intensiv mit **relationalen Datenbanken, SQL** (DDL, DML, DCL), **ERM/ERD**, Tabellenbeziehungen, JOINS und Aggregatfunktionen beschäftigt.

In der dritten und letzten Leistungsbeurteilung (LB3) verknüpfen wir dieses Wissen mit einem **Backend-Server**: Sie implementieren einen **Node.js/Express-Server**, der über eine **REST-API** auf Ihre MySQL-Datenbank zugreift. Das Frontend wird **nicht** programmiert – stattdessen simulieren wir es mit **Postman**.

## Auftrag

Sie wählen im 2er-Team einen der beschriebenen **Use Cases** aus (pro Klasse: jedes Thema nur einmal). Zu diesem Use Case entwickeln Sie:

- ein **sauberes Datenmodell** (ERM/ERD),
- eine passende **MySQL-Datenbank** mit Startdaten und einem AppUser,
- einen **Express-Server**, der die wichtigsten Funktionen als **CRUD-REST-API** bereitstellt,
- ein **Video-Tutorial** (ca. 15 Minuten), in dem Sie Ihre Lösung erklären und demonstrieren.

Zielpublikum des Tutorials sind Ihre «Mit-Auszubildenden» aus dem 2. Lehrjahr, die die Themen

- Daten & Datenbanken,
- Zugriff auf Daten in einer 3-Schichten-Architektur (Client - Server - Datenbank),
- CRUD-Operationen in SQL und über eine REST-API

besser verstehen sollen.

## Inhalt des Videotutorials

Das Video soll strukturiert und nachvollziehbar sein und mindestens folgende Teile enthalten:

### 1. Einleitung

1. Kurzvorstellung des Use Cases (Problem / Idee)
2. Was Ihre App grob können soll

### 2. Analyse & Datenmodell

1. Erklärung des **ERM** (Entitäten, Beziehungen, Kardinalitäten)
2. Erklärung des **ERD** in Crow's-Foot-Notation (Tabellen, PK/FK, Datentypen)

### 3. Datenbank

1. Anlegen der Datenbank & Tabellen per SQL-Skript/Befehlen (DDL)
2. Import der Startdaten per SQL-Skript/Befehlen (DML)
3. Anlegen und Berechtigen eines **AppUsers** (DCL)

#### 4. Backend / Server

1. Aufbau des Node.js/Express-Projekts (wichtigste Dateien kurz erklären)
2. Erklärung der wichtigsten Routen (z.B. GET /api/..., POST /api/...)

#### 5. Tests mit Postman

1. Vorführen der wichtigsten **REST-Endpunkte** mit Postman (Create, Read, Update, Delete) inkl. zeigen der Änderungen in der Datenbank (in Webstorm mit Datenbank-Plugin)
2. Anzeigen eines Beispiels mit **JOIN** (Daten aus 2 Tabellen)
3. Anzeigen eines Beispiels mit **Aggregatfunktion** (z.B. COUNT, AVG, MAX)
4. Sinnvolle Verwendung von **HTTP-Statuscodes** (z.B. 200, 201, 400, 404, 500 ...)

#### 6. Reflexion

1. Jede Person im Team nennt **mindestens 2 Learnings** (positiv / herausfordernd)
2. Was würden Sie beim nächsten Mal anders machen?

#### 7. Schluss

1. Kurze Zusammenfassung Ihrer Lösung
2. Ausblick / mögliche Erweiterungen

## Wichtige Hinweise

**Frontend:** Die Programmierung eines eigenen Frontends ist **kein Bestandteil des Auftrags**. Alle Funktionen werden mit **Postman** getestet und demonstriert.

- Fokus liegt auf:
  - **sauberem Datenmodell** (ERM/ERD, richtige Datentypen, PK/FK),
  - korrekten **SQL-Skripten** (DDL, DML, DCL),
  - funktionierendem **Express-Server** mit CRUD-Endpunkten,
  - sinnvollen **Fehlermeldungen** und HTTP-Statuscodes,
  - verständlicher **Erklärung** im Video.
- Es reicht, wenn Ihr Server Daten **erstellen, lesen, ändern und löschen** kann (CRUD) und mindestens **einen JOIN** und **eine Aggregat-Abfrage** bereitstellt.
- Der Backend-Server darf nicht über den Root-User in der Datenbank Änderungen machen, sondern „nur“ via AppUser.

## Abgabe - Zu liefernde Lernprodukte

Das gesamte Projekt ist als **ein ZIP-File in Moodle** abzugeben.

Im ZIP-File müssen enthalten sein:

1. **Video-Tutorial** (ca. 15 Minuten, gängiges Format: MP4, H.264 Codec, max. FullHD Auflösung (1920x1080px), min. Auflösung: 1280x720px, max. Video-Grösse: 500 MB, Bitrate: ca. 3000-5000kbps)
2. **ERM** und **ERD** als PDF-Datei

3. **SQL-Skript (DDL)**: Anlegen der Tabellenstruktur
4. **SQL-Skript (DML)**: Import / Insert der Startdaten
5. **SQL-Skript (DCL)**: Anlegen und Berechtigen eines AppUsers zur DB (inkl. Kommentar zu den vergebenen Rechten)
6. **Datenbank-Dump**: Erstellen Sie mit mysqldump ein SQL-File, woraus die Datenbank inkl. Daten und Tabellen-Struktur wiederhergestellt werden kann.
7. **Node.js-Projektordner**:
  1. package.json
  2. Server-Datei(en) (z.B. app.js / index.js)
  3. ggf. .env-Beispiel (ohne echte Passwörter)
  4. README.txt mit kurzen Start-Hinweisen (z.B. 'npm install', 'npm start')

Stellen Sie sicher, dass Ihr Projekt auf einem anderen Rechner mit den bereitgestellten Skripten **reproduzierbar** eingerichtet werden kann.

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

[https://wiki.bzz.ch/modul/m290\\_guko/leistungsbeurteilungen/03\\_lb/b\\_projektbeschreibung?rev=1764532398](https://wiki.bzz.ch/modul/m290_guko/leistungsbeurteilungen/03_lb/b_projektbeschreibung?rev=1764532398)

Last update: **2025/11/30 20:53**

