

# Prototyping Projektdokumentation

Name: Noemi Frey

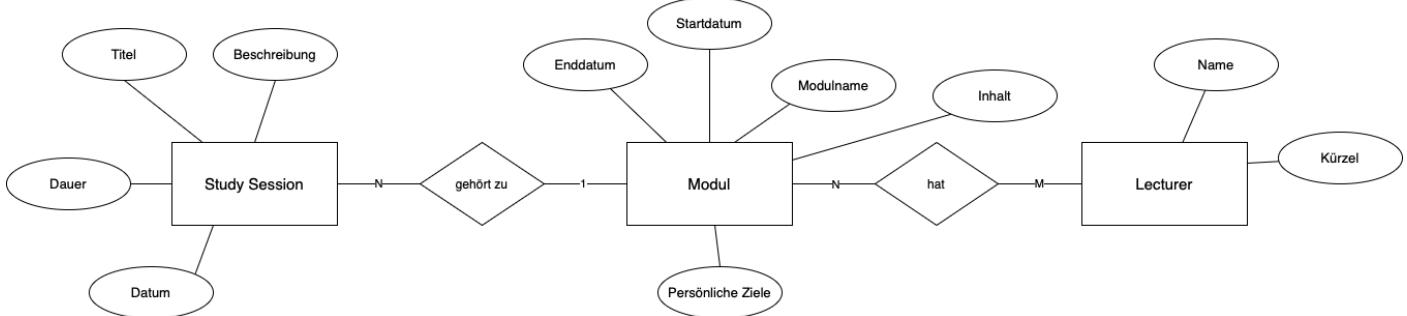
E-Mail: [freynoe1@students.zhaw.ch](mailto:freynoe1@students.zhaw.ch)

URL der deployten Anwendung: <https://gostudynow.netlify.app>

## Einleitung

Die App **GoStudyNow** soll Studierende dabei helfen, ihren Lernprozess festzuhalten und nachzuverfolgen. Mit einem Timer können Lernphasen aufgezeichnet, pausiert oder angepasst werden. Zusätzlich ermöglicht die App eine einfache Verwaltung von Modulen und Dozierenden. **GoStudyNow** macht das Lernen strukturierter, motiviert zum Erreichen persönlicher Ziele und hilft den Überblick über den eigenen Fortschritt zu behalten.

## Datenmodell



## Beschreibung der Anwendung

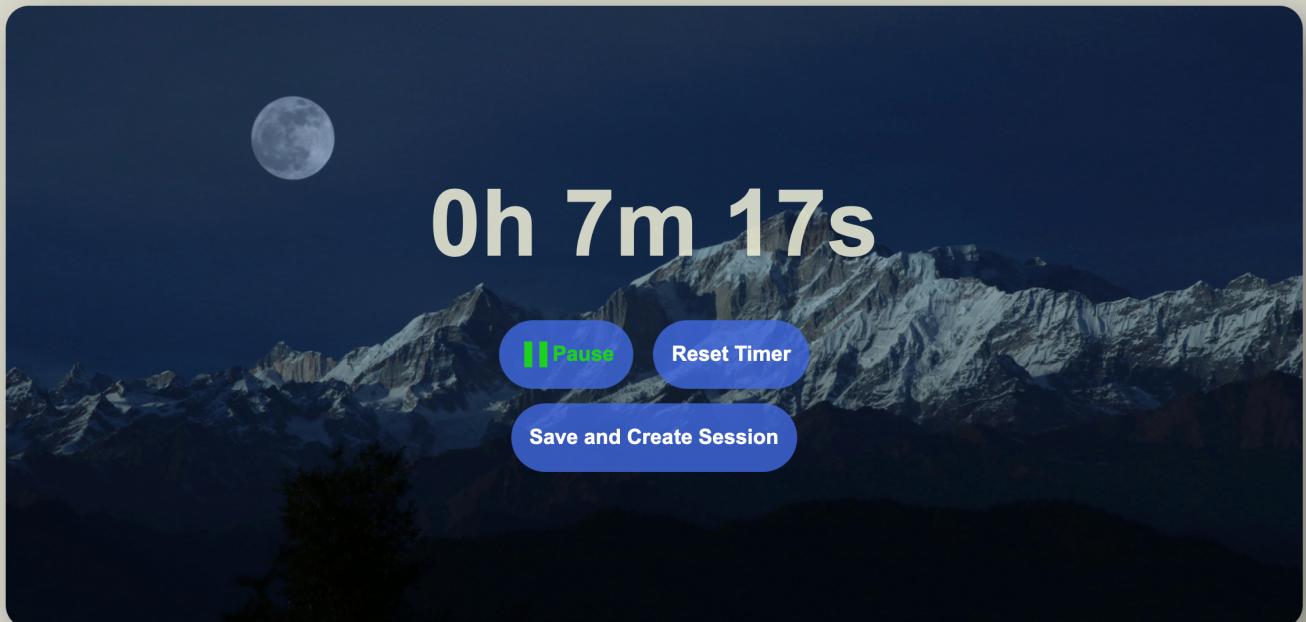
### Startseite – Timer

Route : /

Auf der Startseite wird der Timer angezeigt, der gestartet, pausiert oder zurückgesetzt werden kann. Mit «Speichern» gelangt man zu einem Formular, um eine neue Session zu erstellen. Unten rechts kann ein Thema aus einer kleinen Bildauswahl für den Hintergrund gewählt werden. Dies wurde alles auf +page.svelte gestaltet und umgesetzt. Für save and Create Session wird auf db.js zugegriffen, um eine neue Session zu erstellen.

Dateien:

- lib/db.js
- routes/+page.svelte



## Sessions

### ***Neue Session erstellen***

Route : /sessions/create

Wenn man seine Session stoppen und speichern möchte, kann man auf den Button «Save and Create Session» klicken. Dadurch gelangt man zu einem Formular, um eine neue Session zu erstellen. Das Formular wurde mit der SessionForm.svelte-Komponente umgesetzt und global in der Datei styles.css gestaltet. Innerhalb dieses Formulars besteht zudem die Möglichkeit, ein neues Modul zu erstellen.

Dateien:

- lib/db.js
- routes/sessions/create/+page.svelte
- routes/sessions/create/+page.server.js

## Neue Session erstellen

**Titel****Beschreibung****Datum****Dauer**Dauer: 0h 8m 10s**Modul****Speichern**[\*\*Alle Sessions anzeigen\*\*](#)**Route:** /sessions

Wenn man auf «Sessions» klickt, werden die einzelnen Sessions nach Datum sortiert in einer Timeline angezeigt. Die Timeline selbst wird direkt in der Datei `+page.svelte` umgesetzt und gestaltet. Für die Darstellung der einzelnen Sessions innerhalb der Timeline wird die Komponente `SessionCard.svelte` verwendet.

Dateien:

- `routes/sessions/+page.svelte`
- `routes/sessions/+page.server.js`
- `lib/components/SessionCard.svelte`
- `routes/styles.css`

## Alle Sessions

Mittwoch, 08. Januar 2025

### Gestalten

Beschreibung: html, css

Laufzeit: 2h 17m 53s

Modul: Prototyping

Dienstag, 07. Januar 2025

### Mathe

Beschreibung: Rechnen, Lachen, Gleichungen lösen

Laufzeit: 0h 15m 6s

Modul: Mathe

### Einzelne Session anzeigen

Route: /sessions/[session\_id]

Eine Session kann durch einen Klick auf den Titel im Detail angesehen werden. Die Gestaltung der Detailansicht erfolgt in der Datei styles.css. In dieser Ansicht ist es möglich, die Session zu löschen oder zu bearbeiten. Außerdem gelangt man mit Klick auf das Modul direkt zur Detailansicht des verknüpften Moduls. Mit dem «Back»-Button gelangt man zurück zur Übersicht aller Sessions.

Dateien:

- routes/sessions/[session\_id]/+page.svelte
- routes/sessions/[session\_id]/+page.server.js
- lib/db.js

[Back](#)

## Lesen

Beschreibung: S. 35 - 56

Laufzeit: 0h 8m 10s

Datum: 2025-01-08

Modul: [IT-Projekt Management](#)

[Edit](#)[Delete](#)

### Session bearbeiten

Route: /sessions/edit

Beim Bearbeiten einer Session wird, wird wie auch bei der Erstellung einer neuen Session, das Styling für ein Formular übernommen aus styles.css. Der Unterschied liegt jedoch in den Funktionen: Statt eine neue Session zu erstellen (create), wird eine bestehende aktualisiert (update). Auch in der Bearbeitungsansicht ist es möglich, ein neues Modul zu erstellen oder ein bereits vorhandenes Modul auszuwählen. Für das Formular selbst wird der Komponent SessionForm.svelte verwendet.

Dateien:

- lib/db.js
- lib/SessionForm.svelte
- routes/sessions/edit/[session\_id]/+page.svelte
- routes/sessions/edit/[session\_id]/+page.server.js
- routes/styles.css

[Zurück](#)

## Session bearbeiten

**Titel**[Gestalten](#)**Beschreibung**[html, css](#)**Datum**[08.01.2025](#)**Dauer**[8273](#)

Dauer: 2h 17m 53s

**Modul**[Prototyping](#)[Oder ein neues Modul erstellen](#)

## Module

[\*\*Einzelnes Modul anzeigen\*\*](#)**Route:** /modules/[module\_id]

Entweder durch einen Klick auf den Titel in der Gesamtübersicht aller Module oder durch einen Klick auf das Modul in der SessionCard gelangt man zur Detailansicht eines Moduls. In dieser Ansicht kann das Modul bearbeitet oder gelöscht werden. Das Styling wurde in der Datei styles.css umgesetzt. Ausserdem werden in dieser Ansicht auch die verknüpften Dozenten angezeigt.

**Dateien:**

- lib/db.js
- routes/sessions/[session\_id]/+page.server.js

- routes/sessions/[session\_id]/+page.svelte
- routes/styles.css

## GoStudyNow Sessions Module Dozente

Back

# Prototyping

Inhalt: Curabitur in libero ut massa volutpat convallis. Morbi odio odio, elementum eu, interdum eu, tincidunt in, leo. Maecenas pulvinar lobortis est. Phasellus sit amet erat. Nulla tempus. Vivamus in felis eu sapien cursus vestibulum.

Persönliche Ziele: zufrieden sein

Laufzeit: 2024-09-16 bis 2025-01-09

Dozent: Max Meisterhans, Benjamin Kühnis

Edit

Delete

## Alle Module anzeigen

Route: /modules

Wenn man auf «Module» klickt, werden alle Module dargestellt. Über einen Haken «Nur laufende Module anzeigen» können ausschliesslich die laufenden Module gefiltert werden. Andere Module werden als archiviert betrachtet. Die Gestaltung wird von der Komponente ModuleCard.svelte übernommen. Die Karten-Styling erfolgt in der Datei styles.css.

Dateien:

- lib/db.js
- lib/components/ModuleCard.svelte
- routes/modules
- routes/modules
- routes/styles.css

Nur laufende Module anzeigen

## Alle Module

helfen

lachen

Laufend

Mathe

Kein Ziel definiert

Laufend

Handsgi

Kein Ziel definiert

Archiv

Prototyping

zufrieden sein

Archiv

join

Kein Ziel definiert

Archiv

### Modul bearbeiten

Route: /modules/edit

Für das Styling wird erneut die Datei styles.css verwendet. Es ist möglich, ein Modul zu bearbeiten sowie einen passenden Dozenten auszuwählen, abzuwählen oder hinzuzufügen. Für das Formular wird die Komponente ModuleForm.svelte verwendet.

Dateien:

- lib/db.js
- lib/components/ModuleForm.svelte
- routes/modules/edit/[module\_id]/+page.svelte
- routes/modules/edit/[module\_id]/+page.server.js
- routes/styles.css

09.01.2025

#### Lehrperson

Max Meisterhans

Benjamin Kühnis

Alexandre de Spindler

Oder neuen Dozenten erstellen

Kürzel

Add

Speichern

## Dozenten

[Alle Dozenten anzeigen](#)

Route: /lecturers

Wenn man auf «Dozenten» klickt, werden alle Dozenten angezeigt. Die Karten-Komponenten werden in LecturerCard.svelte definiert, und das Styling erfolgt ebenfalls in styles.css.

**GoStudyNow** [Sessions](#) [Module](#) [Dozente](#)

# Alle Dozente

Max  
Meisterhans  
Kürzel:meix

Benjamin  
Kühnis  
Kürzel:kuhs

Alexandre de  
Spindler  
Kürzel:desa

## Einzelner Dozent

Route: /lecturers/{lecturer\_id}

In dieser Ansicht wird erneut das Styling aus styles.css verwendet. Der Name sowie das Kürzel eines Dozenten werden angezeigt, ebenso wie dessen Verknüpfung mit dem entsprechenden Modul. Auch hier besteht die Möglichkeit, einen Dozenten zu löschen oder zu bearbeiten. Die Funktionen zur Verknüpfung von Dozenten und Modulen werden in den Dateien +page.server.js und db.js verwaltet.

Dateien:

- lib/db.js
- routes/lecturers/{lecturer\_id}/+page.server.js
- routes/lecturers/{lecturer\_id}/+page.svelte

**GoStudyNow** [Sessions](#) [Module](#) [Dozente](#)

[Back](#)

# Max Meisterhans

Kürzel: meix

Module: Mathe,Prototyping,join

[Edit](#)

[Delete](#)

## **Dozent bearbeiten**

**Route:** /lecturers/edit

Beim Bearbeiten eines Dozenten können der Name und das Kürzel angepasst und gespeichert werden. Die zusätzliche Auswahl eines Moduls wurde bewusst weggelassen, da sie redundant gewesen wäre. Für das Formular wird die Komponente LecturerForm.svelte verwendet, und das Styling erfolgt erneut über die Datei styles.css.

Dateien:

- lib/db.js
- routes/lecturers/edit/[lecturer\_id]/+page.server.js
- routes/lecturers/edit/[lecturer\_id]/+page.svelte
- styles.css

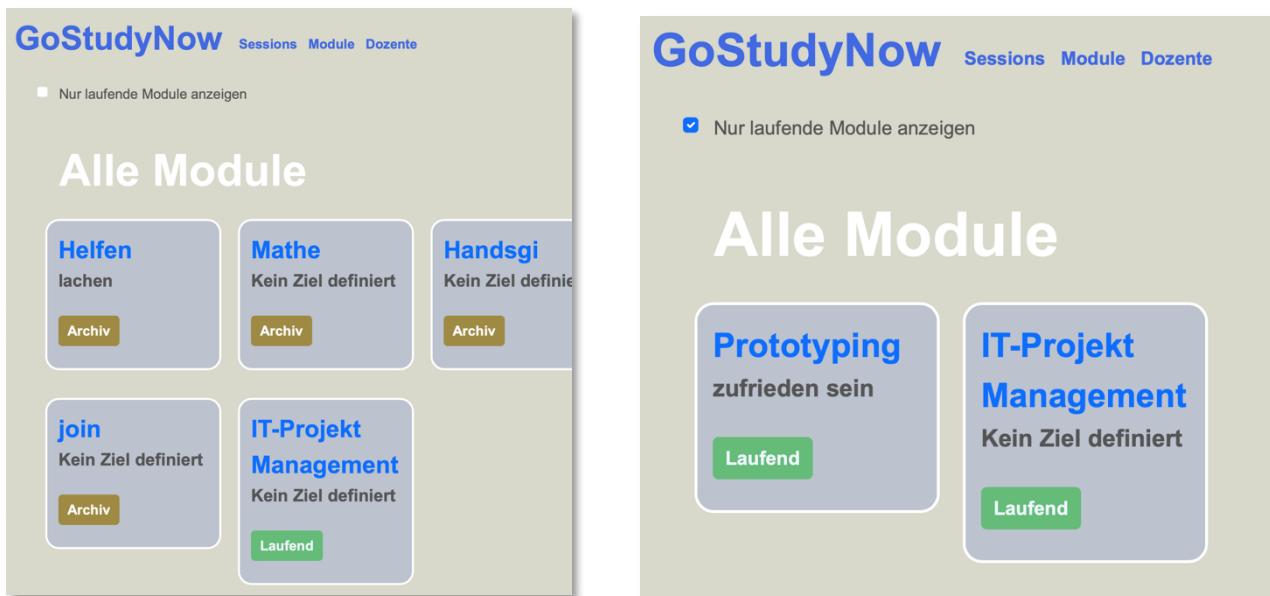


## Erweiterungen

### Laufende-Module-Filter

Bei /modules können Module als laufend markiert oder archiviert werden.

1. **Datenfilterung**
  - Die Module werden anhand ihres current-Status gefiltert.
  - Wenn die Checkbox "**Nur laufende Module anzeigen**" aktiviert ist, wird die Liste der Module dynamisch aktualisiert und zeigt nur die Module mit current: true.
2. **Interaktive Steuerung**
  - Der filterByCurrent-Status wird über die Checkbox gesteuert. Änderungen am Status führen automatisch zu einer Neuberechnung der anzuzeigenden Module.
3. **Komponenten-Integration**
  - Die gefilterten Module werden mithilfe der ModuleCard-Komponente dargestellt, wodurch alle relevanten Informationen visuell konsistent angezeigt werden.
4. **Effiziente Datenbindung**
  - Die Nutzung von \$derived sorgt dafür, dass die Filterung effizient und reaktiv bleibt, ohne die ursprünglichen Moduldaten zu verändern.



### Dateien:

- lib/db.js
- routes/modules/+page.svelte
- routes/modules/+page.server.js

### N zu M Beziehung zwischen Modulen und Dozenten

Die Verknüpfung zwischen Modulen und Dozenten wird in der Datenbank mithilfe von ObjectIds realisiert. Jedes Modul speichert ein Array von ObjectIds im Feld lecturers, das auf die entsprechenden Dozenten verweist. Dieses Konzept ermöglicht eine flexible N-zu-M-Beziehung zwischen Modulen und Dozenten.

### 1. Abrufen und Anzeigen der Verknüpfung

- Beim Laden eines Moduls (Route: /modules/[module\_id]) wird das Array der lecturers-IDs aus der Datenbank gelesen.

- Die Funktion db.getLecturersByIds wird verwendet, um die vollständigen Dozentendaten (Name, Kürzel) anhand der IDs abzurufen.
- Die Namen der Dozenten werden in der Detailansicht eines Moduls angezeigt. Falls ein Dozent nicht gefunden wird, bleibt das Feld leer.

## 2. Bearbeiten der Verknüpfung

- Beim Bearbeiten eines Moduls (Route: /modules/edit/[module\_id]) kann die Liste der Dozenten angepasst werden. Hierbei werden die Dozentendaten im Formular ModuleForm.svelte angezeigt, wo bestehende Dozenten ausgewählt oder neue hinzugefügt werden können.
- Neue Dozenten werden durch db.createLecturer in die Datenbank eingefügt, falls sie noch nicht existieren.
- Die IDs aller ausgewählten Dozenten werden gesammelt und in der Funktion db.updateModule zusammen mit den anderen Moduldaten gespeichert.

## 3. Speicherung der Verknüpfung

- Die Verknüpfung wird durch die Speicherung der Dozenten-IDs im lecturers-Array des Moduls realisiert. Beim Speichern werden alle IDs in ObjectIds umgewandelt, um sie mit der Datenbankstruktur kompatibel zu machen.

## 4. Löschen von Verknüpfungen

- Wird ein Modul gelöscht, bleiben die Dozentendaten in der Datenbank bestehen, da die Verknüpfung nur auf Modulseite gespeichert wird. Dadurch bleibt die Integrität der Datenbank erhalten.

## Dateien

- lib/db.js – Datenbankoperationen für Module und Dozenten
- routes/modules/edit/[module\_id]/+page.server.js – Serverlogik für das Bearbeiten von Modulen
- lib/components/ModuleForm.svelte – Formular zur Bearbeitung von Modulen
- routes/lecturers/[lecturer\_id]/+page.svelte – Anzeige einzelner Dozenten

## Timer

Methode	Funktion	Details
toggleTimer	Startet oder pausiert den Timer.	- Wenn der Timer läuft ( <code>isRunning</code> ), wird der aktuelle <code>setInterval</code> -Timer gestoppt. - Falls pausiert, wird ein neuer <code>setInterval</code> gestartet, der die Zeit ( <code>time</code> ) jede Sekunde erhöht und den Zustand speichert.
resetTimer	Setzt den Timer zurück.	- Stoppt den laufenden Timer mit <code>clearInterval</code> . - Setzt die Zeit ( <code>time</code> ) auf 0 und den Status ( <code>isRunning</code> ) auf <code>false</code> .
saveState	Speichert den aktuellen Zustand des Timers im <code>sessionStorage</code> .	- Speichert die aktuelle Zeit ( <code>time</code> ) und den Status ( <code>isRunning</code> ) als Strings, um sie später wiederherstellen zu können.
loadState	Lädt den gespeicherten Zustand des Timers aus dem <code>sessionStorage</code> .	- Liest die gespeicherte Zeit und den Status. - Falls der Timer aktiv war, wird er automatisch wieder gestartet.
startTimer	Startet den Timer manuell.	- Setzt ein neues <code>setInterval</code> , das die Zeit jede Sekunde erhöht und den Zustand speichert. - Wird in <code>loadState</code> verwendet, um den Timer automatisch fortzusetzen.
handleVisibilityChange	Pausiert den Timer automatisch, wenn die Seite nicht mehr sichtbar ist.	- Bei einem Wechsel in den Hintergrund ( <code>document.hidden</code> ) wird der laufende Timer gestoppt, und der Zustand wird gespeichert.
handleSaveAndCreate	Speichert die Zeit und leitet den Nutzer zur Seite für das Erstellen einer neuen Session weiter.	- Stoppt den Timer und erstellt Query-Parameter mit der Zeit und dem Datum, die an die nächste Seite übergeben werden.
changeBackground	Ändert das Hintergrundbild des Timers.	- Aktualisiert das Hintergrundbild basierend auf dem ausgewählten Bildpfad.

## Dateien

- `/routes/+page.svelte`

## Timeline

- **Sortieren der Sessions:** Die Sessions werden nach Datum absteigend sortiert, sodass die neueste Session zuerst angezeigt wird.
- **Gruppieren nach Datum:** Sessions werden pro Tag gruppiert und mit einer Überschrift in einem benutzerfreundlichen Format dargestellt (z. B. "Montag, 7. Januar 2025").
- **Darstellung der Gruppen:** Jede Gruppe enthält eine Überschrift (Tag) und die zugehörigen Sessions als Timeline-Elemente.
- **Visuelle Hervorhebung:** Sessions werden mit einem Marker (✓) und in ansprechenden Boxen mit Schatten und farbigem Hintergrund angezeigt.
- **Session-Komponente:** Die Details jeder Session werden mithilfe der SessionCard-Komponente visualisiert.
- **Dynamische Datenbindung:** Änderungen in den Daten (z. B. Hinzufügen oder Löschen von Sessions) werden automatisch in der Timeline reflektiert.

## Dateien:

- `Routes/sessions/+page.svelte`