



# IBMS\{A}: StudyCal

Alexa van der Meulen, Claas Wenk, Tobias Gehrke, Vera Schwarz, Sebastian Fuchs,  
Raphael Tack

## Table of Contents

<b>Kurzbeschreibung der Projektidee .....</b>	<b>4</b>
<b>Wer ist die Zielgruppe? .....</b>	<b>4</b>
<b>Festgelegter Funktionsumfang für „StudyCal“ .....</b>	<b>5</b>
<b>Geplante Technologie für Front- und Backend .....</b>	<b>7</b>
<b>Zuständigkeiten im Team .....</b>	<b>7</b>
<b>Product Backlog – Kalender-App „StudyCal“ .....</b>	<b>8</b>
<b>Dokumentation .....</b>	<b>9</b>
<b>SQL (Backend Datenbank) .....</b>	<b>9</b>
<b>Front-End .....</b>	<b>11</b>
Beispielhafte Fehlermeldung .....	18
<b>Java-Script.....</b>	<b>19</b>
InactivityTimer.js .....	19
Main.js .....	19
Popup.js .....	19
<b>Back End .....</b>	<b>20</b>
<b>Verzeichnis .....</b>	<b>20</b>
<b>Config .....</b>	<b>21</b>
<b>Database .....</b>	<b>21</b>
<b>Klassenstruktur Back End.....</b>	<b>21</b>
<b>Calendar Classes.....</b>	<b>21</b>
<b>Datenbankklasse .....</b>	<b>22</b>
<b>Handler .....</b>	<b>22</b>
Calendar-Handler .....	22
Entry-Popup-Handler .....	23
Export-Handler: .....	23
To-Do-Handler .....	23
<b>Login-Klassen .....</b>	<b>23</b>
Password.php .....	23
Registration.php .....	24
<b>Session-Klassen .....</b>	<b>24</b>
Session-Manager .....	24
User-Session .....	24
<b>User-Klassen .....</b>	<b>24</b>
Account-Manager: .....	24
User.php:.....	25
<b>Hauptkomponenten .....</b>	<b>25</b>
Admin.php.....	25

22.04.2025

Datenschutz.html .....	25
Index.html .....	25
Login.php .....	25
Logout.php: .....	26
Password.php .....	26
Register.php .....	26
Start.php .....	26

## Kurzbeschreibung der Projektidee

StudyCal ist eine intelligente Kalender-App, welche speziell für Lernende entwickelt wird. Dabei soll die App die Terminplanung übernehmen mit dem Ziel, effiziente Lehreinheiten zu gestalten. Auf diese Weise können die Lerneinheiten, welche durch ein individuelle Lernideal angepasst werden können, strukturiert werden. Somit können Termine und „To-Do’s“ übersichtlich dargestellt werden.

So kann strukturierter, stressfreier und produktiver gelernt werden.

## Wer ist die Zielgruppe?

Die Zielgruppe wird durch alle Personen dargestellt, die sich mehr Struktur und Unterstützung in ihrer Lernphase wünschen. Demnach handelt es sich dabei um einzelne Nutzer (in diesem Sinne ebenfalls Käufer) der App, wobei diese sowohl für Schüler sowie Studenten geeignet ist als auch für Personen mit Abgabeterminen oder Prüfungen. Auch können Unternehmen die Struktur verwenden als auch von Bildungsinstituten genutzt werden, um gesamten Schüler- und. Studentenschaft Unterstützung anzubieten.

Weiterhin sind die Beeinflusser festzustellen: Diese setzen sich aus den Käufern, welche die Inhalte und Optimierung der App beeinflussen können, sowie anderen Lern-Apps zusammen. Dabei können andere existierende Lern-Apps aus zwei verschiedenen Sichtweisen erfasst werden, einerseits für eine mögliche Kooperation andererseits als Konkurrenz.

# Festgelegter Funktionsumfang für „StudyCal“

## 1. Nutzerregistrierung und Login

- *Funktion:*
  - Nutzer\*innen können sich registrieren und anmelden
  - Sichere Speicherung der Zugangsdaten (Hinterlegung in der Datenbank, das Passwort wird gehasht abgelegt, sowie die Sicherheitsfrage)
  - Zurücksetzen des Passwortes durch eine Sicherheitsfrage, die hinterlegt wird
- *Zuständigkeit:*
  - Tobias Gehrke (Front-End), Claas Wenk(Back-End)
- *Ziel:*
  - Persönliche Lernumgebung schaffen

## 2. Persönlicher Kalender

- *Funktion:*
  - Nutzer\*innen können eigene Termine (z. B. Klausuren) erstellen
  - Kalenderansicht in Wochenansicht (geplant: Werktage)
- *Zuständigkeit:*
  - Tobias Gehrke und Alexa van der Meulen (Front-End), Claas Wenk und Raphael Tack (Back-End), Sebastian Fuchs und Vera Schwarz (Datenbank)
- *Ziel:*
  - Besser Übersicht über Lerneinheiten und Klausuren, sowie tägliche To-Dos

## 3. Lernzeit-Tracking

- *Funktion:*
  - Optionale manuelle Eingabe von Lernzeiten (Auswahl der Zeiten z.B. 30 min, 60 min, etc.)
- *Zuständigkeit:*
  - Raphael Tack (Back-End), Alexa van der Meulen (Front-End)
- *Ziel:*
  - Transparente Erfassung des tatsächlichen Lernverhaltens

## 4. Lernziele setzen

- *Funktion:*
  - Nutzer definieren tägliche, wöchentliche oder themenbasierte Lernziele
    - Ein einfaches Pop-Up-Fenster, welches einen Eintrag ermöglicht, später auch verwendet für den „To-Do“-

Eintrag. Gespeichert wird es in der Datenbank und durch das eingestellte Lernideal wird automatisch eine Lerneinheit anhand des Lernideals erstellt

- *Ziel:*
  - Zielgerichtetes und nachhaltiges Lernen durch Zielverfolgung

## 5. Benutzeroberfläche mit Dark Mode

- *Funktion:*
  - Umschaltbarer Dark Mode (manuell) an den User gekoppelt
    - Durchgeführt wird es durch einen Auswählbutton im Einstellungsbereich
    - Die Einstellung wird in der Datenbank hinterlegt und anschließend wieder abgerufen, sobald sich der User wieder anmeldet
  - Benutzerfreundliches, anpassbares Design (Accessibility beachten)
- *Zuständigkeit:*
  - Tobias Gehrke und Claas Wenk und Alexa van der Meulen (Front-End), Raphael Tack (Back-End)
- *Ziel:*
  - Angenehmes Nutzererlebnis auch bei langen Lernsessions oder in dunkler Umgebung

### **Zusammenfassung:**

Kategorie	Enthaltene Funktionen
Nutzerkonto	Registrierung, Login
Lernplanung	Kalender, Ziele setzen
Lernverhalten	Tracking
Nutzererlebnis	Dark Mode

## Geplante Technologie für Front- und Backend

Für die technische Umsetzung unseres Projekts verfolgen wir einen klassischen Ansatz mit klarer Trennung von Front- und Back-End. Die serverseitige Entwicklung erfolgt mithilfe von PHP, das lokal über den in XAMPP integrierten Apache-Webserver betrieben wird. Die Datenspeicherung erfolgt über eine MySQL-Datenbank, die ebenfalls über XAMPP lokal gehostet wird.

Die Client-Seite wird mit HTML und CSS realisiert. Zur Umsetzung dynamischer und interaktiver Funktionen setzen wir zusätzlich JavaScript ein. Diese Technologien ermöglichen eine benutzerfreundliche Oberfläche und eine reibungslose Kommunikation mit dem Back-End.

Zur Versionskontrolle und teaminternen Zusammenarbeit wird Git in Kombination mit einem zentralen GitHub-Repository ([GitHub](#)) verwendet. Dadurch wird eine strukturierte und nachvollziehbare Entwicklung gewährleistet.




## Zuständigkeiten im Team

- Backend: Raphael Tack, Claas Wenk
- Frontend: Tobias Gehrke, Alexa van der Meulen
- Datenbank: Sebastian Fuchs, Vera Schwarz




## Product Backlog – Kalender-App „StudyCal“

**Zweck:** Eine App, die Nutzer\*innen hilft, ihre Lernzeiten zu planen, zu tracken und mithilfe von Analysen Empfehlungen zur optimalen Lernzeit gibt.

### Hoch priorisierte Einträge:

ID	Titel	Typ	Priorität	Machbar?
1	Nutzerregistrierung und Login	Nutzer können sich mit einem Passwort anmelden	Funktional	
2	Persönlicher Kalender	Nutzer können Termine und Lernzeiten eintragen und bearbeiten	Funktional	
3	Lernzeit-Tracking	Erfassung des Lernens	Funktional	X
4	Empfehlung idealer Lernzeiten	Algorithmus gibt Empfehlungen auf Basis ein- & vorgegeben Daten	Funktional	
5	Erinnerungen	Automatische Erinnerungen an Lernzeiten	Funktional	X

### Mittlere Priorität

ID	Titel	Typ	Priorität	Machbar?
7	Lernziele setzen	Nutzer können sich Lernziele setzen	Funktional	
8	Dark-Modus	Nutzer können entscheiden zwischen Dark-/ Light-Modus	Nicht- Funktional	
9	Exportfunktion	Exportieren der Nutzer	Funktional	

### Niedrige Priorität

ID	Titel	Typ	Priorität	Machbar?
10	Erfolge?	Nutzer erhalten Badges für Erfolge	Nicht-Funktional	X



# Dokumentation

## SQL (Backend Datenbank)

Dieser Teil der Dokumentation beschreibt die Struktur und Funktionsweise der relationalen Datenbank, welche zur Umsetzung des Projektes verwendet wird. Diese Art der Datenbank kann mithilfe der *Structured Query Language* (SQL) sowie unter Verwendung der Open-Source Software *XAMPP* die Erstellung, Verwaltung und Abfrage von Daten in der Datenbank ermöglichen. Zudem bietet sie Entwicklern und Administratoren eine klare Übersicht über die zugrundeliegenden Datenmodelle sowie die Abfragemöglichkeiten.

Zu Beginn wurde eine allgemeine Übersicht erarbeitet und auf Basis von analogen Medien ein Konzept zum grundlegenden Aufbau der Datenbank entwickelt. Anschließend erfolgte eine Einarbeitung in *XAMPP* und insbesondere in *PHPMyAdmin*, um eine erste Datenbank zu erstellen. Das analog erarbeitete Konzept wurde im Anschluss in SQL umgesetzt und optimiert, wobei der Fokus auf einer effizienten Datenabfrage und der verbesserten Übersichtlichkeit der Datenbank lag.

Im weiteren Verlauf wurde sich intensiv mit den benötigten Daten und den überflüssigen Informationen auseinandergesetzt und überlegt, wie diese am besten strukturiert und untergliedert werden können. Durch verschiedene Überlegungen und mit Absprache mit dem Back-End Team wurde die Struktur der Datenbank überarbeitet.

Schließlich haben wir ein Konzept mit mehreren Tabellen entwickelt. Dieses Konzept ist ebenfalls in Form eines Entity Relationship Model (ERM) sowie eines Relationalen Datenmodells (RDM) dargestellt (siehe unten). Die Erklärung der Datenbankstruktur und der dahinterliegenden Überlegungen zu Darstellbarkeit der Daten sowie der Beziehungen, in denen die Daten zueinanderstehen, wird hierbei an den genannten Modellen erläutert.

In dem ERM existieren vier Entitäten. Diese sind *user*, *todo*, *event* und *entry*. *User* enthält dabei alle Attribute, welche relevant sind, um sämtliche den Nutzer betreffende Daten zu erfassen. Dabei ist zu beachten, dass der *Username* einzigartig sein muss, da dieser zur Erstellung des Nutzers und anschließend zur Ermittlung der entsprechenden *UserID* verwendet wird. Dies wird jedoch sowohl in der Datenbank durch die entsprechende Bezeichnung *UNIQUE* als auch durch eine Abfrage bei Erstellung eines neuen Nutzers der Kalender-App sichergestellt. Der Primärschlüssel wird durch die *UserID* realisiert, welche automatisch von der Datenbank bei neuer Registrierung eines Nutzers erstellt wird. Auf diese Weise kann über die *UserID* andere Entitäten mit dieser Entität verbunden werden.

Die Entität *todo* stellt die umzusetzende Speicherung eines To-Dos dar, wobei an dieser Stelle die *UserID* des Users als Fremdschlüssel in der Tabelle eingetragen wird, um die Beziehung *sets* umzusetzen.

Weiterhin stellt *event* die jeweiligen durch den Nutzer zu erstellende Kalendereinträge dar. Damit wird demnach beschrieben, welche Klausuren oder Abgaben anstehen, für welche der Nutzer lernen möchte. Wie bei der Umsetzung der Beziehung zwischen *user* und *todo* wird bei *user* und *event* ebenfalls die *UserID* bei *event* als Fremdschlüssel eingetragen, um die Beziehung *creates* zu realisieren. Der Primärschlüssel dieser Entität ist dabei als *EventID* zu identifizieren.

Als letzte Entität besteht *entry*, welche die einzelnen im Kalender eingetragenen Termine beinhaltet. Dabei ist *entry* als schwache Entität modelliert, da das *event* aus einzelnen Einträgen zusammengesetzt wird und somit *entry* nur in Verbindung mit *event* eindeutig ist. Durch diese Modellierung ist es nicht notwendig, den Namen des *events* in jedem einzelnen Kalendereintrag zu speichern, wodurch eine Mehrfachspeicherung in *event* und *entry* vermieden wird. Aufgrund dessen besitzt *entry* die beiden Primärschlüssel *EntryID* sowie *EventID*, um diese schwache Entität in der Datenbank umzusetzen.

Um den Zugriff über PHP zu ermöglichen, wurde ein spezieller Administrator-Account eingerichtet. Dieser Account dient ausschließlich dem Zugriff via PHP und stellt sicher, dass nur berechtigte Personen Änderungen an der Datenbank vornehmen können. Somit kann die App während der Ausführung auf die Datenbank zugreifen, wobei die Daten für den Login dieses Nutzers zur Datenbank in einer csv-Datei gespeichert, welche mit *configuration.csv* bezeichnet wurde.

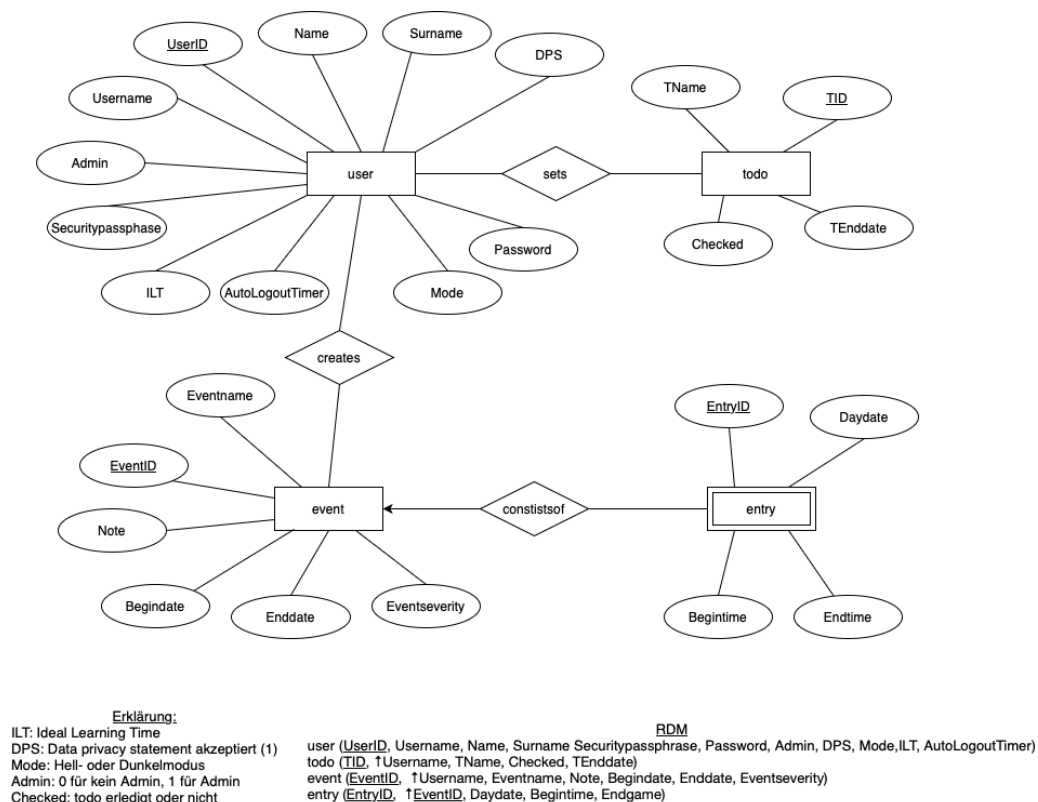


Abbildung 1 - ERM und RDM

## Front-End

Das Front-End, welches sich vor allem mit dem Aufbau (HTML), der Gestaltung der Webseite (CSS) und der Interaktion der einzelnen Elemente (JavaScript) beschäftigt, wurde vor allem in der Festlegung des Designs sichtbar. Dabei wurden ein Violett-Ton und ein Dunkelblau ausgewählt, um die primären Farben darzustellen.

Beginnend mit der Landeseite, kann zunächst ausgewählt werden, ob sich neu registriert werden möchte oder eingeloggt werden möchte.



Abbildung 2 - Landeseite

Sofern nun die Registrierung ausgewählt wird, so erscheint eine neue Seite:

**Registriere dich bei StudyCal**

Vorname

Vorname eingeben

Nachname

Nachname eingeben

Benutzername

Benutzername eingeben

Passwort

Passwort eingeben

Passwort wiederholen

Passwort wiederholen

Sicherheitsfrage

Wie lautet der Name deines ersten Haustiers?

Antwort eingeben

Registrieren

[← Zurück](#)

© 2025 StudyCal. Alle Rechte vorbehalten.

Abbildung 3 - Registrierungsseite

Hier müssen verschiedene relevante Felder ausgefüllt werden, wobei die Daten anschließend in der Datenbank gespeichert werden. Hierbei wurde vor allem auf eine Struktur geachtet, welche eine Rangfolge in der Beantwortung darstellt. Nach der Eingabe der relevanten Felder, kann sich angemeldet werden:

**Willkommen zurück bei StudyCal**

  
**STUDYCAL**

Bitte melden Sie sich an

Benutzername

Benutzername eingeben

Passwort

Passwort eingeben

[Passwort vergessen?](#)

Einloggen

[← Zurück](#)

© 2025 StudyCal. Alle Rechte vorbehalten.

Abbildung 4 – Seite zum Login

22.04.2025

Nachdem sich angemeldet wurde, mit dem erstellten Benutzernamen und Passwort, wird man zu dem Kalender weitergeleitet.

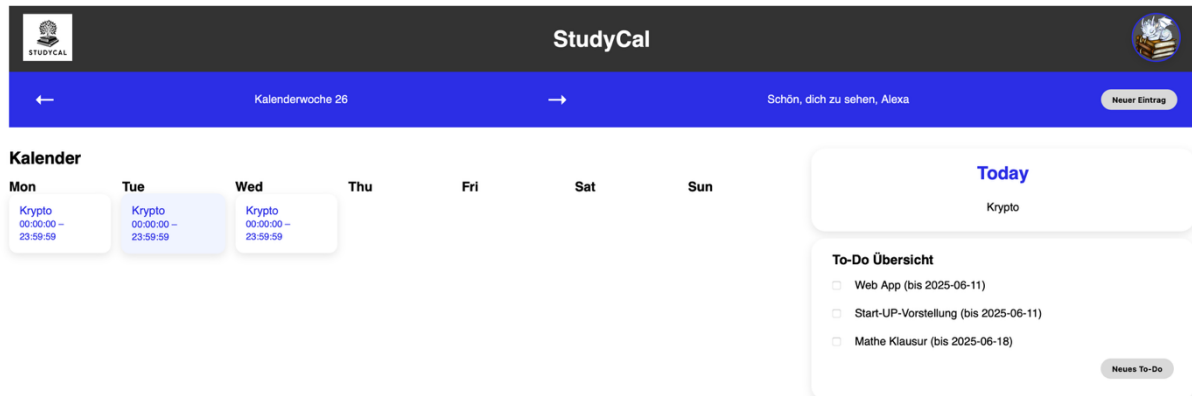


Abbildung 5 - Kalender

Hier werden übersichtlich Termine in der kommenden Woche sowie heutige Aktivitäten (auf der rechten Seite unter *Today*) dargestellt. Darunter ist die To-Do-Liste abgebildet.

Sofern eigene Einstellungen erwünscht werden, kann auf das Profilbild geklickt werden, wodurch ein Drop-Down-Menü aufgezeigt wird.

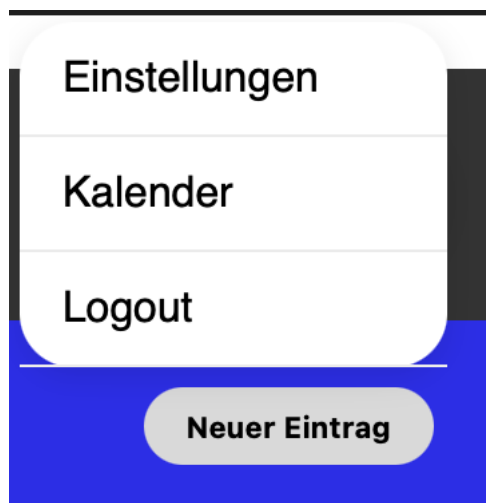


Abbildung 6 - Dropdown

22.04.2025

Hier kann nun ausgewählt werden, welche Aktion ausgeführt werden soll. Beispielsweise können die Einstellungen ausgewählt werden:

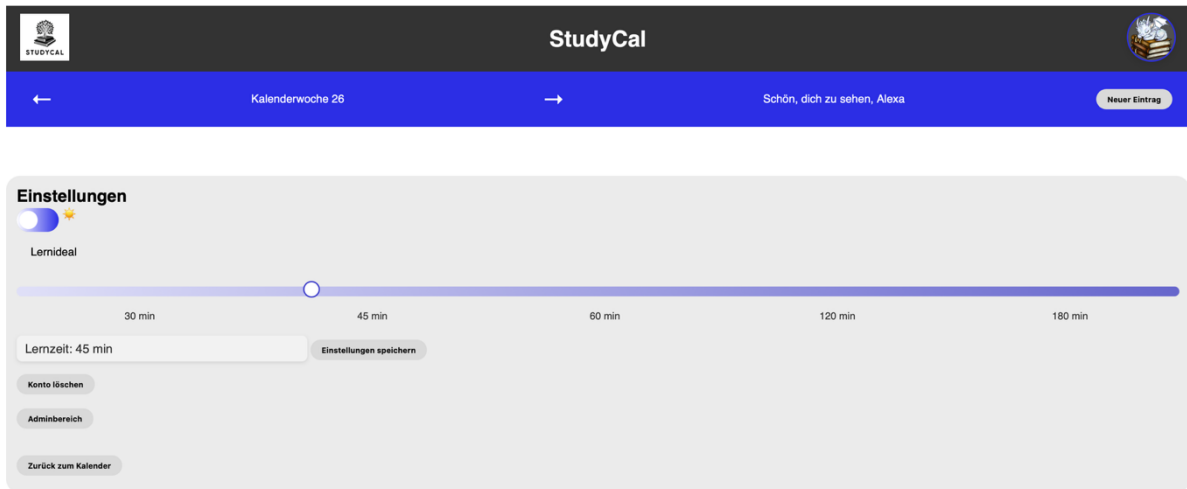
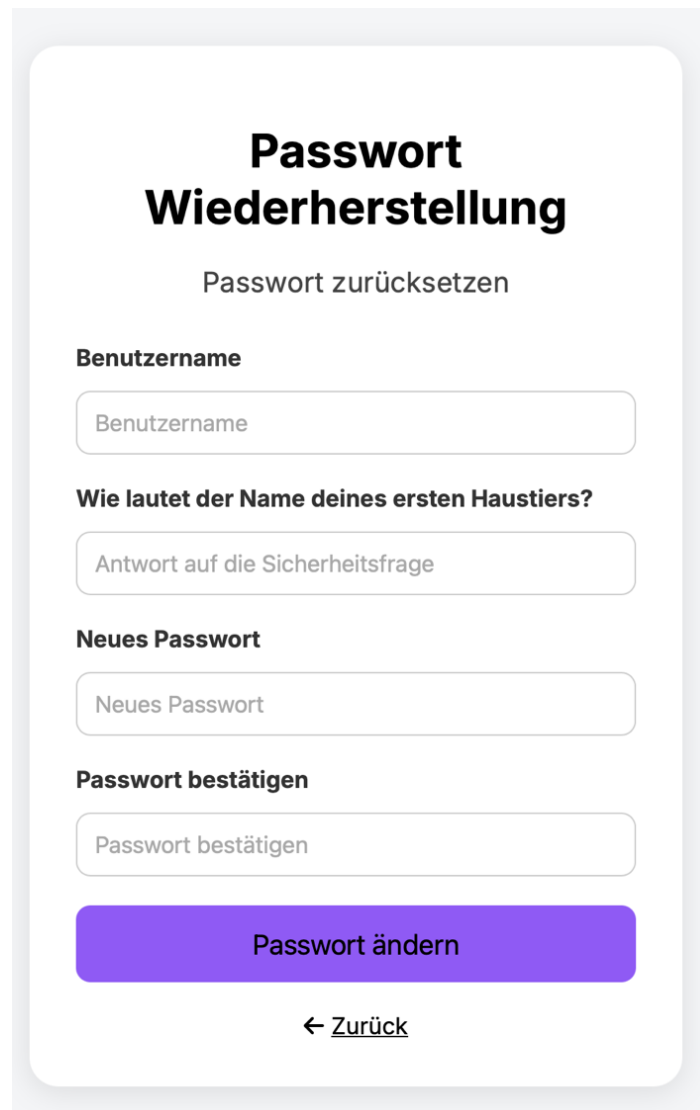


Abbildung 7 - Einstellung

In dieser Ansicht können verschiedene Einstellungen ausgewählt werden, einerseits das Lernideal (die gewünschte Zeit für eine Lerneinheit), aber auch eine Passwortänderung.

22.04.2025

Bei dieser Auswahl wird der Nutzer auf eine andere Seite geleitet, sodass er durch die Sicherheitsfrage das Passwort zurücksetzen kann.



The image shows a mobile application interface for password recovery. It features a white card with rounded corners on a light gray background. The card has a title 'Passwort Wiederherstellung' in bold black text, followed by the subtitle 'Passwort zurücksetzen'. Below this, there are four input fields, each preceded by a label: 'Benutzername', 'Wie lautet der Name deines ersten Haustiers?', 'Neues Passwort', and 'Passwort bestätigen'. The first three labels are in bold, while the fourth is not. Each input field contains a placeholder text matching its label. At the bottom of the card is a large purple button labeled 'Passwort ändern' and a link with a left arrow and the text '← Zurück'.

**Passwort Wiederherstellung**

Passwort zurücksetzen

**Benutzername**

Benutzername

**Wie lautet der Name deines ersten Haustiers?**

Antwort auf die Sicherheitsfrage

**Neues Passwort**

Neues Passwort

**Passwort bestätigen**

Passwort bestätigen

**Passwort ändern**

← Zurück

Abbildung 8 - Passwort zurücksetzen

Hinzukommend kann auch der Dark-Modus in dem Einstellungsbereich ausgewählt werden, sodass die Benutzeroberfläche dunklere Kontraste verwendet, um dem Benutzer die Bedienbarkeit zu vereinfachen.

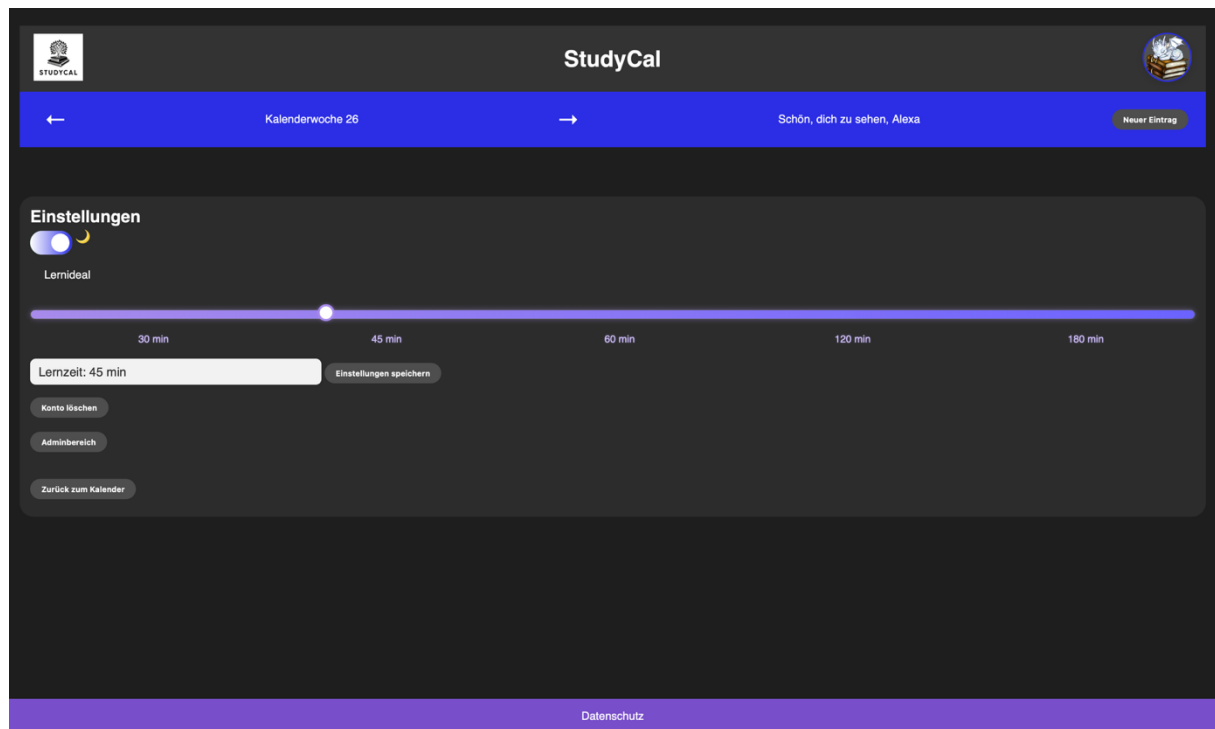


Abbildung 9 - Einstellung (Dark-Mode)

Sofern nun der Rückgang zum Kalender erwünscht ist, kann erneut auf das Profilbild geklickt werden.

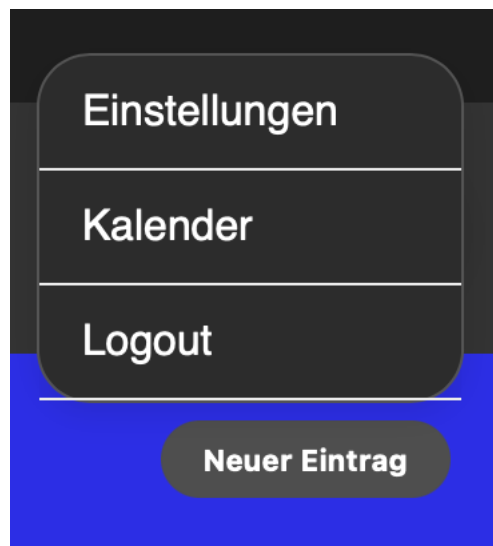


Abbildung 10 - Dropdown (Dark-Mode)



22.04.2025

Auch dieses Menü ist nun für die Bedienbarkeit angepasst, sodass die Benutzerfläche einheitlich ist.

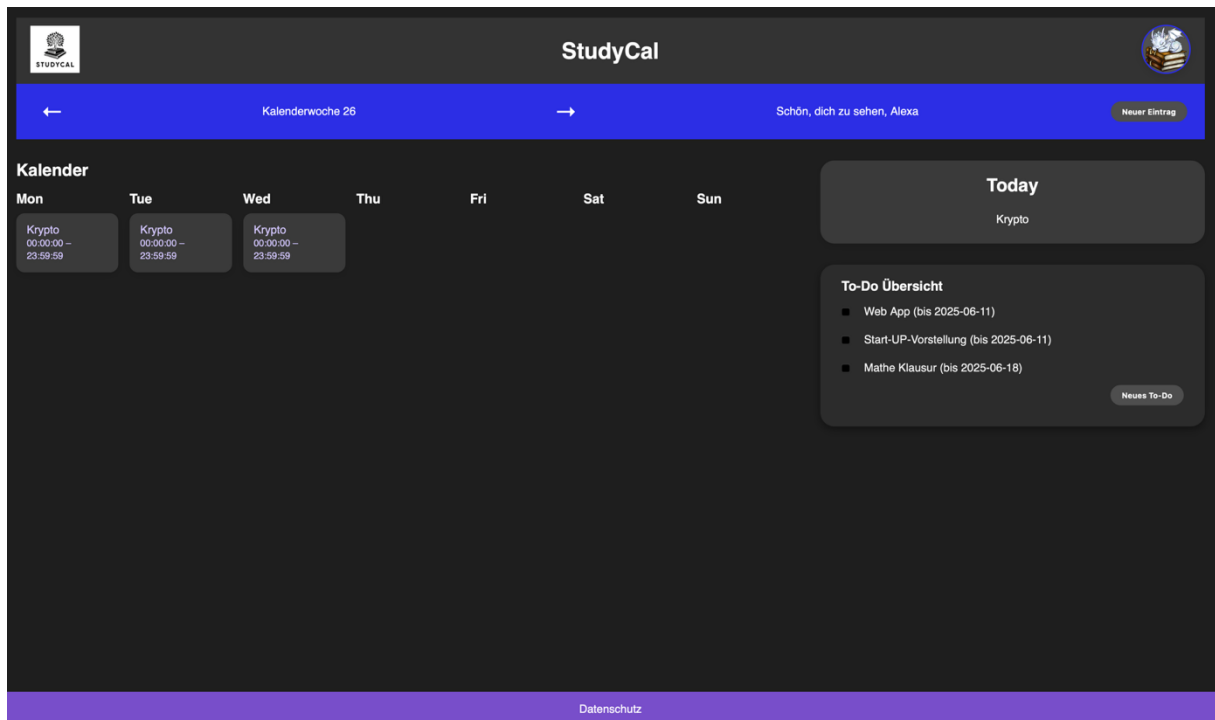
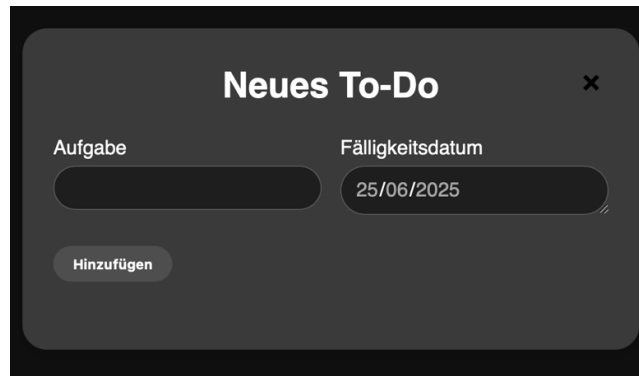


Abbildung 11 - Kalender (Dark-Mode)

Nun kann durch den Button Neuer Eintrag ein neuer Eintrag für den Kalender eingestellt werden, aber auch ein neues To-Do kann erstellt werden, welche durch JQuery umgesetzt wurde.

Abbildung 12 - Neuer Eintrag (Pop-Up)



Neues To-Do

Aufgabe

Fälligkeitsdatum

25/06/2025

Hinzufügen

Abbildung 13 - To-Do-Eintragserstellung (Dark-Mode)

## Beispielhafte Fehlermeldung

### Eingabefehler

5–50 Zeichen, nur Buchstaben, Zahlen, \_  
und - erlaubt.

Schließen

### Eingabefehler

Bitte gleiche Passwörter eingeben.

Schließen

### Login fehlgeschlagen

Benutzername oder Passwort ist falsch.

Schließen

### Eingabefehler

Bitte alle Felder ausfüllen.

Schließen

Abbildung 14 - Fehlermeldungen

## Java-Script

Das Java-Script, welches im System Unterordner zu finden ist, beschreibt die Interaktion der Webseite mit dem Nutzer. Aufgrund dessen wird im Folgenden die grundlegende Funktion jeder Datei erklärt.

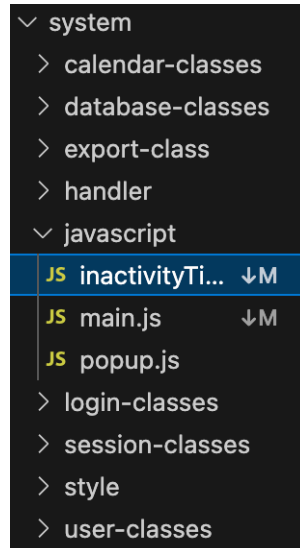


Abbildung 15 - Java-Script Ordnerstruktur

### InactivityTimer.js

Diese Datei ist zuständig für den *Log-Out-Timer*. Dies bedeutet konkret, dass der Timer sich bei einer Aktion des Benutzers zurücksetzt. Wenn der Nutzer sich neu anmeldet, wird der Timer wieder gestartet. Weiterhin dient sie zur Umleitung auf die Landeseite des Nutzers, sofern der Timer abgelaufen ist.

### Main.js

In der Main.js-Datei sind grundlegende Funktionen implementiert. Dazu gehören das Drop-Down-Menü (welches für die Navigation durch die Webseite genutzt wird) sowie der Dark-Mode, welcher ausgewählt werden kann. Weiterhin sind hier die Einträge und To-Do-Pop-Ups integriert. Bei dem To-Do-Pop-Up wurde ebenfalls AJAX verwendet, um die To-Do's bei Beendigung der Aktivität durchzustreichen. Desweiteren ist in der *start.php*-Seite ebenfalls der HTML-Code für die Einstellungen beschrieben. Allerdings ist auffällig, dass diese jedoch nicht zu sehen sind. Daher wird eine Anforderung gesendet, die anschließend bearbeitet wird, sodass in der *main.js* die Anfrage in einer Funktion bearbeitet wird. Darüber hinaus wird auch der Wert des Sliders und die Adminrechte überprüft.

### Popup.js

In dieser Datei werden die benutzerdefinierten Pop-Ups beschrieben und individualisiert. Hinzukommend wird hier die PHP-Fehlermeldung exportiert und anschließend als Pop-Up angezeigt.

## Back End

Dieser Teil der Dokumentation beschreibt den grundsätzlichen Aufbau des Back Ends mit dem Schwerpunkt der Strukturierung des Dateiverzeichnisses. Für dieses wurde hauptsächlich PHP verwendet, wobei die grundsätzlichen Funktionen durch den Import verschiedener Bibliotheken erweitert werden (PDO, Hash, etc.).

## Verzeichnis

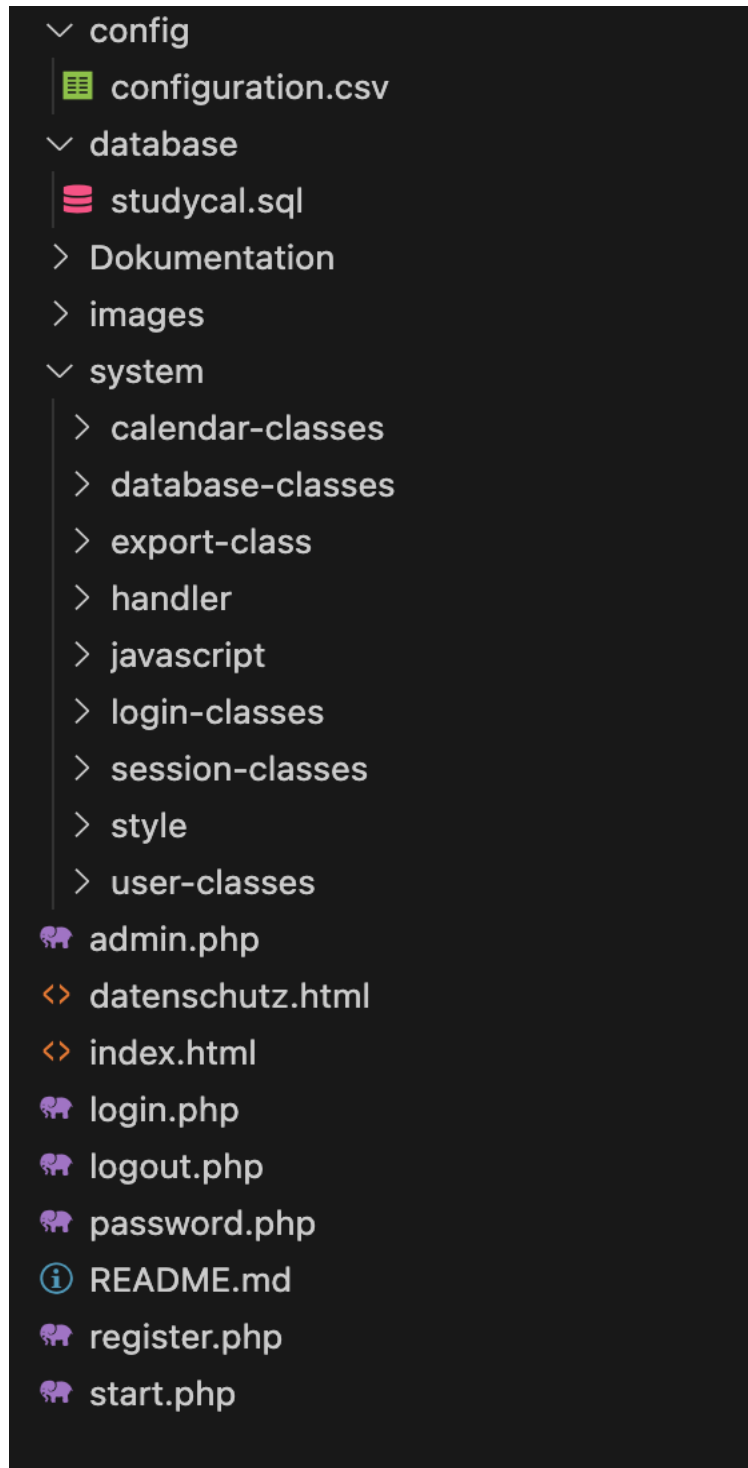


Abbildung 16 - Back-End Verzeichnis

Die grundlegende Struktur des Back Ends wird in der o.g. Abbildung dargestellt. Auffällig ist dabei, dass Dateien in verschiedenen Unterordnern entsprechend ihrer Funktion abgelegt werden.

## Config

Unter *config* befindet sich die Konfigurationsdatei, welche mit *configuration.csv* benannt wurde. Diese Datei enthält die Konfigurationsdaten zu Erstellung einer Verbindung von der Web-App zu der Datenbank.

## Database

In diesem Unterordner befindet sich die Datenbankdatei, welche mit PHPMyAdmin importiert werden kann. Als Name der Datenbank ist es notwendig, *studycal* anzugeben, falls dieser bei dem Import nicht übernommen wurde. Weiter ist zu beachten, dass der entsprechende Nutzer für die Anbindung der Datenbank angelegt wird. Dafür werden folgende Informationen benötigt: **localhost** (als Host), **studycal** (als Datenbankname), **Admin** (als Nutzernamen), **rH!>|r'h6.XXlN.=2}A\_#u[gxvhU3q;** (als Passwort des Nutzers).

## Klassenstruktur Back End

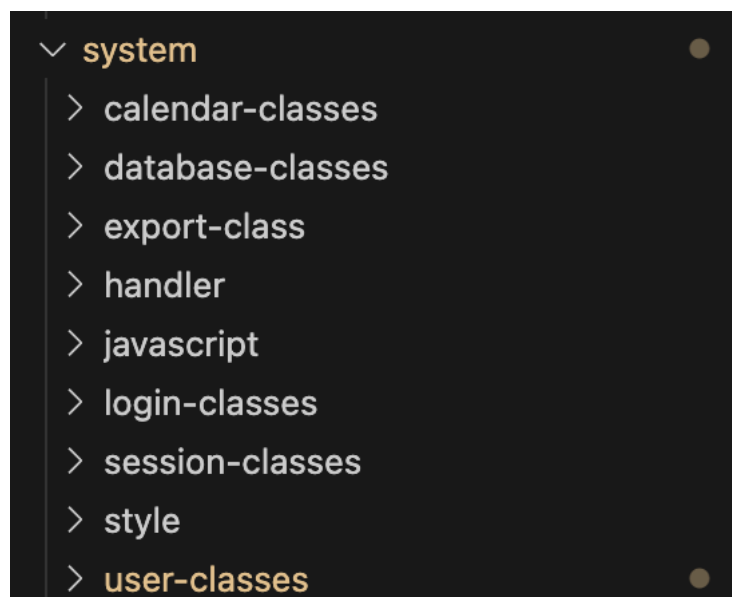
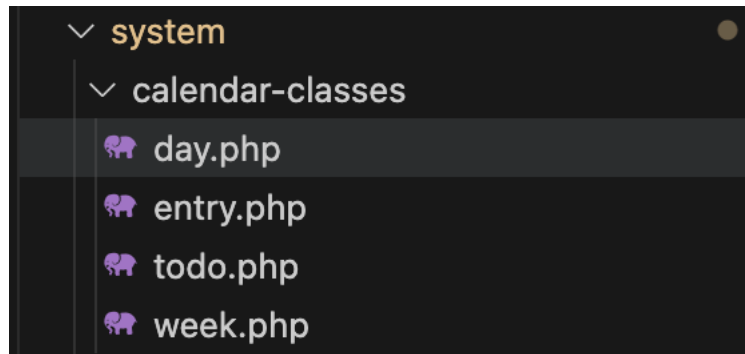


Abbildung 17 - Klassenstruktur im Back-End

Im Systemverzeichnis liegen die grundlegenden Klassen für die Funktionsweise des Back-Ends. Hier sind Datenstrukturen definiert, die Datenbankanbindung, sämtliche Handler, sowie logische Funktionsweisen.

## Calendar Classes

Um den Kalenderalgorithmus zu implementieren, müssen grundlegende Datenstrukturen gegeben sein, um die Daten aus der Datenbank in das Back-End zu übernehmen.

*Abbildung 18 - Kalender Klassen*

Jeder Kalender wird in Wochen organisiert, welche die Kalenderwoche und die entsprechenden Tage enthalten. Die Tage beinhalten das Datum und sämtliche Einträge, die über die Eintrag-Klasse erstellt werden und in dem korrespondierenden Tag dargestellt werden. Weiterhin ist ein Eintrag organisiert mit einem Titel, einer Beschreibung sowie einer Start- und Endzeit.

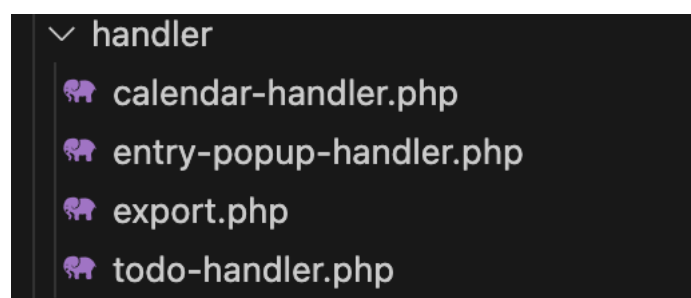
Unabhängig vom Kalender sind die To-Do's, welche ausschließlich aus einem Titel bestehen.

Dieses Vorgehen ermöglicht eine Trennung von Daten und Logik, wodurch eine bessere Lesbarkeit entsteht.

## Datenbankklasse

In dieser Klasse erfolgt die Verknüpfung der Web-App mit der Datenbank durch die Konfigurationsdatei. Dabei wurde die Datenbankanbindung zur besseren Lesbarkeit sowie Verringerung des Codes in anderen Bereichen ausgelagert.

## Handler

*Abbildung 19 - Handler Dateien*

Die Handlerklassen sind essenziell und enthalten grundlegende logische Funktionen des Kalenders, der Popups und To-Do's sowie der JSON-Exporte. Dabei existieren verschiedene Handler, welche im Folgenden betrachtet werden.

### Calendar-Handler

Zunächst existiert der Calendar-Handler zur Abrufung sämtlicher Einträge. Außerdem dient er dem Hinzufügen von Elementen sowie der Anbindung der Datenbank und

organisiert die Wochen, Tage und Einträge in der Datenbank. Dabei erfolgt eine logische Gruppierung der Tage mit den zugehörigen Einträgen in einer Woche in einem Array.

## Entry-Popup-Handler

Weiterhin gibt es den Entry-Popup-Handler. Dieser enthält die Logik für den Eintrag-Button sowie die Fehlerabfrage bei leeren Feldern. Außerdem dient er für das Hinzufügen von Einträgen zu den jeweiligen Tagen.

## Export-Handler:

Der Exporthandler ist zuständig für den Export der User-Daten, um diese in einer JSON-Datei zu speichern.

## To-Do-Handler

Der To-Do-Handler ruft sämtliche To-Do's von der Datenbank ab und ordnet diese der jeweiligen User-ID-Session zu. Weiterhin können To-Do's als erledigt markiert und aus der Datenbank gelöscht oder neue To-Do's hinzugefügt werden.

## Login-Klassen

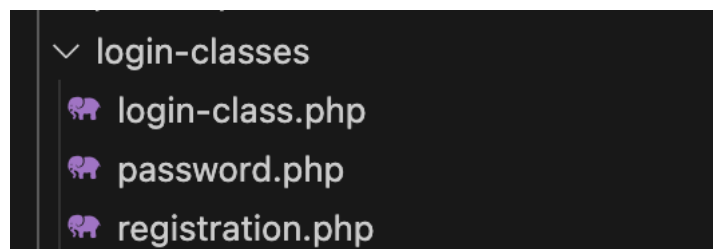


Abbildung 20 - Login Klassen

In der Login-Klasse befindet sich die gesamte Logik für das Login. Darunter auch das Regex-Template, welches für das Abfangen falscher Eingaben genutzt wird. Hinzukommend werden Passwörter mit ihrem korrespondierenden Hashwert verglichen, sodass der User sich anmelden kann. Zusätzlich wird hier die Session erstellt und alle Userdaten werden in einer Session-ID gespeichert. Hierbei werden vorab eingestellte Angaben (bspw. Dark-Mode) übernommen. Zusätzlich liegt hier auch die Anbindung für das Login, sodass durch sämtliche genannte Aspekte die Anforderung zur Manipulation erfüllt sind.

## Password.php

In dieser Datei wird die grundlegende Logik für das Zurücksetzen der Passwörter beschrieben. Außerdem wird die Sicherheitsfrage, nachdem sie gehasht wurde, mit dem in der Datenbank gespeicherten Hash verglichen. Ebenfalls erfolgt ein Vergleich der verbleibenden Felder mit Regex. Außerdem enthält diese Datei die Anbindung für das Passwort.

## Registration.php

Bei der Registrierung eines neuen Nutzers werden für jedes Feld die jeweilige Eingabe serverseitig geprüft sowie das Passwort und die eingegebene Antwort auf die Sicherheitsfrage gehasht in der Datenbank gespeichert. Dabei existiert ein Regex-Template für die Eingabe und für bestimmte Einträge in der Datenbank werden voreingestellte Werte gespeichert. Diese sind: White-Mode, Lernideal auf den Wert 2 sowie Admin mit dem Wert 0 (kein Admin). Somit stellt diese Datei die Anbindung für die Registrierung dar. Außerdem wird die Richtigkeit der Angaben überprüft und bei fehlerhaften Angaben eine entsprechende Meldung ausgegeben.

## Session-Klassen

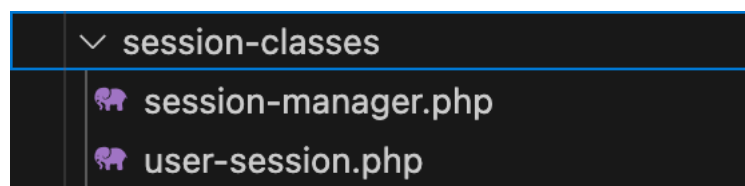


Abbildung 21 - Session Klassen

## Session-Manager

In dem Session-Manager liegt die Logik der Session. Dies bedeutet das Starten einer Session, die Zuweisung der Variablen und auch die Überprüfung von bereits laufenden Sessions. Sofern die Session abgelaufen ist, so wird der User direkt zur Loginseite geleitet. Desweiteren kann die Session durch den Session-Manager beendet und diese mit dem Befehl `unset` deklariert werden. Abschließend können die User-Daten innerhalb einer Methode aufgerufen werden.

## User-Session

Die User-Session ist eine separate Klasse für die Einstellung von Dark-Mode und Lernideal. Sofern diese Daten (innerhalb einer Session) verändert werden, werden diese gespeichert und können anschließend wieder aufgerufen werden.

## User-Klassen

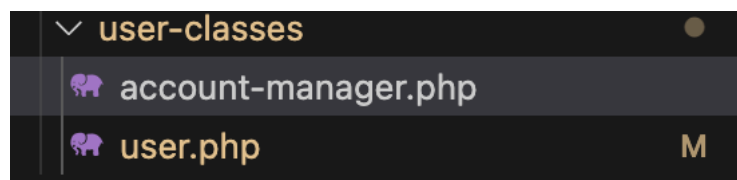


Abbildung 22 - User Klassen

## Account-Manager:

Der Account-Manager ist für die Logik des Accounts zuständig. Konkret bedeutet dies das Löschen des Accounts und das Updaten von Account-Daten (bspw. Lernideal und Dark-Mode).



## User.php:

Diese Klasse definiert einen Nutzer, welcher bei der Registrierung erstellt wird. Auf diese Weise können Daten in Objekten gespeichert und in die Session geschrieben werden. Dabei werden sämtliche Operationen in Methoden zusammengefasst. Dazu gehören: Löschen von Daten, Speichern der Einstellungen sowie Laden des Nutzers aus der Datenbank, Auto-Logout Timer sowie die Lernideal-Logik. Mithilfe von Prepared-Statements sollen dabei *SQL-Injections* verhindert werden, da die Manipulation der Daten in der Datenbank mithilfe von SQL-Abfragen umgesetzt wurde.

## Hauptkomponenten

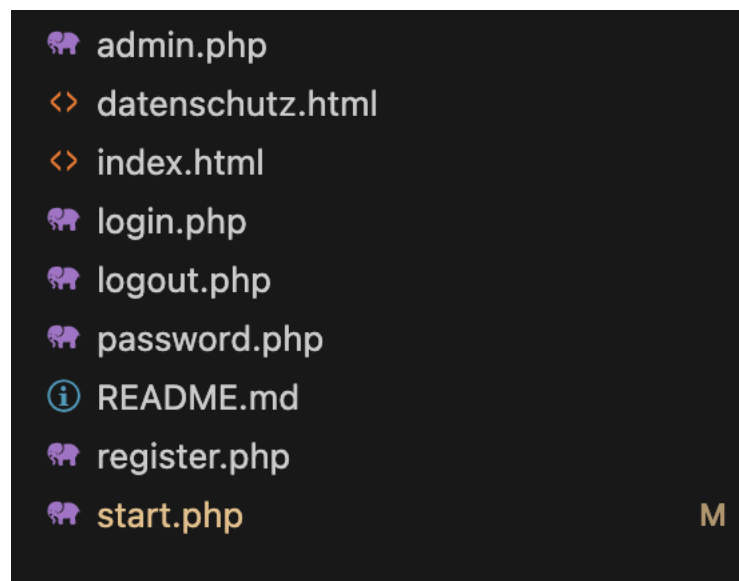


Abbildung 23 - Hauptkomponenten im Back-End

## Admin.php

Diese Datei stellt einen separaten Adminbereich dar, welcher nur von Admins aufgerufen werden kann. Dafür wird überprüft, ob der Nutzer ein Admin ist, wobei eine 1 in dem Feld Admin eingetragen sein muss. Unter diesem Bereich ist es möglich, Nutzerdaten zu ändern. Dabei besteht ebenfalls die Möglichkeit, einen Nutzer als Admin ändern zu können oder ihm die Adminrechte zu entziehen. Außerdem ist ein Admin ebenfalls in der Lage, die Nutzerdaten als JSON-Datei zu exportieren.

## Datenschutz.html

Die HTML-Datei, welche den Datenschutz beinhaltet.

## Index.html

Die Loginseite (Abbildung x im Front-End), welche als Landeseite angesteuert wird.

## Login.php

Die Login.php.-Datei beinhaltet die Überprüfung der Vollständigkeit sämtlicher Felder, wie auch den HTML-Code für den Syntax und den Inhalt der Seite. Sofern eine fehlerhafte

oder unvollständige Eingabe erfolgt, so wird eine Mitteilung dieser ausgegeben. Relevant ist dabei, dass Klassen aus dem System Unterordner verwendet werden.

### Logout.php:

Im Gegensatz zur Login.php gibt es auch eine Logout.php, wodurch der Nutzer sich aus der App ausloggen kann, sodass er zur Landeseite geleitet wird. Hinzukommend wird die Session beendet, indem die Session-Manager-Klasse angewendet wird.

### Password.php

Password.php beinhaltet das Zurücksetzen des Passworts. Auch hier wird die Überprüfung der Vollständigkeit sämtlicher Felder durchgeführt, sowie eine Fehlermeldung bei fehlerhafter oder unvollständiger Eingabe. Dabei wird erneut auch den *system*-Unterordner zurückgegriffen.

### Register.php

Diese Datei beinhaltet die Eingabefelder für die Registrierung eines Nutzers. Dabei wird die Richtigkeit der Angaben überprüft und bei fehlender Eingabe erscheinen entsprechende Meldungen. Für die Logik wird auf die Datei *registration.php* unter *system* zugegriffen.

### Start.php

Die Start.php Seite besteht aus dem HTML-Code, sowie die grundsätzlichen Funktionen, die für die Verwendung der App benötigt werden. Desweiteren auch der Algorithmus zur Erstellung von Lerneinheiten auf Grundlage des Lernideals. (Weitere Funktionen werden im Code beschrieben und durch Kommentare getrennt.)