KANTI
BADEN

Kanti Koala

Die Lern- und Studienhilfsapp für Schüler:innen der Kantonsschule Baden

Maturitätsarbeit, Kantonsschule Baden
Schriftlicher Kommentar

Erstbetreuer: Michael Schneider
Zweitbewerterin: Julia Smits

Geschrieben von: Aryan Anand (G22b), Simon Haddon (G22b)

Datum: 11. November 2025

Abstract

Die vorliegende Maturitätsarbeit befasst sich mit der Entwicklung einer webbasierten Lern- und Studienhilfe für Schüler:innen der Kantonsschule Baden. Ausgangspunkt war die Fragestellung, wie digitale Werkzeuge den Lern- und Organisationsalltag von Schüler:innen erleichtern können. Ziel war es, eine Anwendung zu entwerfen, die schulische Aufgaben, Lernzeiten und Notenverwaltung zentral zusammenführt und durch intelligente Funktionen das Selbstmanagement fördert.

Zur Beantwortung dieser Frage wurde zunächst eine umfangreiche Recherche durchgeführt, bestehend aus einer Literaturstudie zu Lernmethoden, Zeit- und Pausenmanagement sowie Stressbewältigung, ergänzt durch Interviews mit Lehrpersonen und eine Umfrage unter Schüler:innen. Die theoretischen Erkenntnisse flossen in die Konzeption der Applikation ein. Technisch wurde die App mit Python und dem Webframework Flask entwickelt, wobei eine SQL-Datenbank für die Datenspeicherung eingesetzt wurde. Besonderes Augenmerk galt der Benutzerfreundlichkeit, Datensicherheit und der Integration eines algorithmischen Planungssystems.

Das Ergebnis ist die Webanwendung Kanti Koala, die als zentrale Plattform für Schulorganisation dient. Hauptfunktionen sind eine Agenda mit automatischer Lernzeitplanung, ein Notenmanagement-System, ein Pomodoro-basierter Lerntimer sowie tägliche Lern- und Motivationstipps. Der entwickelte Lernzeitalgorithmus plant Lernblöcke dynamisch nach Priorität, Terminen und verfügbaren Zeiten der Nutzer:innen. Die App zeigt, wie digitale Anwendungen das Zeitmanagement und die Lernmotivation von Schüler:innen nachhaltig unterstützen können. Zukünftige Arbeiten sollen Feedback aus der praktischen Nutzung einbeziehen, um die App weiter zu optimieren und

auf andere Schulen auszuweiten.[1]

[1]ChatGPT (Version GPT-5): „Überarbeite den folgenden Text, damit er sprachlich und stilistisch den Standards einer wissenschaftlichen Maturitätsarbeit entspricht. Achte auf korrekte Grammatik, präzisen Ausdruck, logische Argumentation und sachlichen Stil. Behalte den ursprünglichen Sinn und Stil des Textes bei, aber formuliere ihn wissenschaftlicher und grammatikalisch korrekt. [...]. “, 04.11.2025. Antwort ganz übernommen.

Inhaltsverzeichnis

1. Einleitung – unsere Vision

Der Schulalltag an der Kantonsschule Baden ist durch eine hohe Arbeitsdichte geprägt. Neben zahlreichen Prüfungen, auf die gezielt vorzubereiten ist, sind regelmässig Hausaufgaben anzufertigen. In Kombination mit ausserschulischen Verpflichtungen entstehen erhebliche Anforderungen an Planung und Selbstorganisation, die ohne geeignete Hilfsmittel nur schwer zu bewältigen sind. Digitale Werkzeuge können hier ansetzen, indem sie Aufgaben bündeln, Transparenz schaffen und Lernzeiten strukturiert planbar machen.

1.1 Motivation und Relevanz

Aus der Perspektive von Schüler:innen besteht ein wiederkehrendes Problem darin, Lernaufwände, Termine und persönliche Ressourcen in Einklang zu bringen. Versäumnisse führen zu Stress, ungleichmässiger Belastung und ineffizientem Lernen. Eine zentrale, benutzerfreundliche Applikation, welche Lernzeiten, Termine und Noten integriert, ist daher fachlich relevant: Sie stärkt Selbstmanagement und Verbindlichkeit, reduziert kognitive Belastung in der Planung und kann so Lernqualität und Wohlbefinden positiv beeinflussen. Für die Kantonsschule Baden ist der Kontext besonders bedeutsam, da heterogene Anforderungen (Prüfungen, Projekte, Freizeit) koordiniert werden müssen. Zum Zeitpunkt der Fragestellung ist die konkrete Plattformwahl (Web- oder native Applikation) bewusst offen; die Untersuchung erfolgt plattformagnostisch.

1.2 Fragestellung und Zielsetzung

Hauptfragestellung:

> Lässt sich eine digitale Applikation konzipieren und prototypisch umsetzen, die den Lern- und Organisationsalltag von Schüler:innen der Kantonsschule Baden messbar verbessert?

Begründung: Die Frage fokussiert auf Machbarkeit und Wirkung einer digitalen Lösung, ohne die Plattform (Web oder nativ) vorab festzulegen. Sie ist überprüfbar über Indikatoren wie Planbarkeit, Termintreue, wahrgenommene Kontrolle und Aufwandreduktion.

Zur Konkretisierung werden folgende Teilfragen untersucht:

- **Wirksamkeit:** Verbessert die Applikation Planbarkeit, Termintreue und die wahrgenommene Kontrolle über Lernzeiten?
- **Funktionsgruppen:** Welche Funktionen (z. B. Planung/Agenda, Leistungsübersicht, Timer, Hinweise) stiften den grössten Nutzen?
- **Plattform- & Architekturwahl:** Welche Plattform (Web oder nativ) ist unter schulischen Rahmenbedingungen (Verfügbarkeit, Wartung, Deployment) zweckmässig?
- **Datenschutz & Sicherheit:** Sind die vorgesehenen Massnahmen (u. a. Passwortschutz, Transportverschlüsselung) für den Schulkontext angemessen?

1.3 Aufbau und Begründung des schriftlichen Kommentars

Der Bericht ist entlang der Entwicklungslogik strukturiert:

1. **Recherche:** Ausgangslage, Literatur- und Internetrecherche, Interviews und Umfrage als empirische Grundlage; diese sichern Nachvollziehbarkeit der Anforderungen.

2. **Methodik - Programmierung:** Architektur, Datenmodell, Routen, Kernlogik (Lernzeitalgorithmus) und UI; die Reihenfolge folgt der technischen Abhängigkeit vom Systementwurf zur Implementierung.

3. **Features:** Darstellung der Hauptfunktionen (Agenda, Noten, Lerntimer, Tipps) als Umsetzung der Anforderungen.

4. **Sicherheitskonzept:** Begründete Auswahl und Implementierung von Schutzmassnahmen (Passwortsicherheit, HTTPS, CSRF).

5. **Tests:** Sicherung von Funktionsfähigkeit und Qualität.

6. **Schlussfolgerung und Ausblick:** Beantwortung der Fragestellung, Limitationen und Weiterentwicklung.

Diese Gliederung ist gewählt, um von der Problem- und Evidenzbasis über die Umsetzung zur Bewertung zu führen und damit die zentrale Fragestellung stringent zu beantworten.[1]

2. Recherche

2.1 Einführung

2.1.1 Warum brauchen wir eine Recherche?

Da wir eine Web-App erstellen wollen, welche möglichst gut an die Bedürfnisse von Schüler:innen angepasst ist, durften wir uns nicht nur auf unsere eigenen Erfahrungen als Schüler verlassen, sondern mussten auch ein gewisses

[1] ChatGPT (Version GPT-5): „Überarbeite den folgenden Text, damit er sprachlich und stilistisch den Standards einer wissenschaftlichen Maturitätsarbeit entspricht. Achte auf korrekte Grammatik, präzisen Ausdruck, logische Argumentation und sachlichen Stil. Behalte den ursprünglichen Sinn und Stil des Textes bei, aber formuliere ihn wissenschaftlicher und grammatikalisch korrekt. [...]. “, 04.11.2025. Antwort ganz übernommen.

Mass an Recherche erledigen, damit wir wichtige Entscheidungen sinnvoll begründen konnten. Zu diesem Ende haben wir uns entschieden, uns tiefgründig mit unserem Zielpublikum - Schüler:innen der Kantonsschule Baden - auseinanderzusetzen, indem wir Interviews mit PPP-Lehrpersonen führten und eine Umfrage für Schüler:innen gestalteten.

Die Recherche stellt hier nicht das Kernstück unserer Arbeit dar, sondern ist ein unterstützender, aber dennoch sehr wichtiger, Bestandteil für die Entwicklung der Web-Applikation, da sie uns hilft, uns in unsere Zielgrupe zu versetzen und ihre Bedürfnisse Wir setzten einen starken Wert auf begründete Entscheidungen und strebten eine möglichst hohe Qualität an.

2.1.2 Was recherchieren wir?

Nun ging es zuerst einmal darum, herauszufinden was wir überhaupt recherchieren wollten. Da unsere Web-Applikation das Lernen fördern soll, stand der Kernpunkt, nämlich das Lernen, schon von Anfang an klar.

Um genaue Recherche-Themen auszusuchen, stellten wir einige W-Fragen zum Lernen:

- Wie oder warum lernt man gut?
- Wann lernt man gut oder nicht gut?
- Was hindert das Lernen?

Somit entstanden unsere vier Hauptbereiche, welche die obigen drei Fragen beinhalten: Lerntechniken, das Pausen- und Zeitmanagement und das Stressmanagement.

Aufgrund unserer geplanten Features für die Web-Applikation, wie die Agenda oder der Lerntimer, wussten wir auch schon, dass das Zeitmanagement besonders wichtig sein wird.

2.1.3 Wie führen wir die Recherche durch

Uns war klar, dass wir, da wir uns möglichst tief in unser Zielpublikum versetzen, uns hauptsächlich auf die Meinungen von Personen in und im Umfeld

unserer Zielgruppe. Somit standen persönlichere Methoden, wie Interviews und Umfragen, schon früh in Erwägung.

Wir entschieden uns also nach Rücksprache mit unserem Erstbewerter, die Recherche in drei wesentliche Teile zu teilen:

- Eine begrenzte Literaturstudie, welche uns in das Thema einführen und ein wenig vertrauter mit der Materie machen soll.
- Darauf aufbauend führen wir Interviews mit Expert:innen, um diese Materie konkret zu vertiefen und, wenn möglich, auf das Zielpublikum zu beziehen.
- Durch Feedback aus den vorherigen Schritten bereiten wir eine Umfrage für unser Zielpublikum vor, welches uns Daten aus der Sicht der Schüler:innen liefern soll.

Somit können wir anhand dieser Strategie das Wissen, welche wir in der Literaturstudie auffinden, in den Interviews vertiefen, mit unserem gewünschten Umfeld vergleichen und dann gezielt in der Umfrage mit der Praxis vergleichen. Dies sollte uns erlauben, einiges an nützlichen Informationen für unsere Web-Applikation zu erhalten und daraus wichtige Entscheidungen zu treffen.

2.2 Literaturstudie

2.2.1 Ziel

Da die Literaturstudie als Unterstüztung für die anderen, für uns weitaus wichtigeren Elemente unserer Recherche gedacht war, lag nicht besonders viel Fokus darauf. Ziel war, hier ein kleines Stückchen an grundlegendem Wissen zu erreichen ohne dass wir komplett in der Materie verloren gehen und die Übersicht über was tatsächlich für unsere Arbeit nötig ist verlieren.

2.2.2 Vorgehensweise

Die Literaturstudie ist hauptsächlich aus zwei Teilen aufgebaut: Eine breitere Internet-Recherche um ein gewisses Basiswissen zu erreichen, und einen tiefe-

ren Einblick in zwei Sachtexte zum Lernen. Dieses kombinierte Basiswissen fliesst dann direkt in die Vorbereitung für die Interviews hinein.

2.2.3 Internet-Recherche

Wie erwähnt, ging es hier primär um den Aufbau eines Grundsatzes an Vorwissen, welches wir an unserer bisherigen Erfahrung als Kantischüler anhängen können. Somit fokussierten wir uns nicht darauf, möglichst breite und diverse Quellen einzuholen, sondern darauf, dass wir dieses Vorwissen einigermassen effizient aufbauen können. Daraus merkten wir, dass nicht viel nötig war, um dies zu erlangen.

Lerntechniken und -methoden

Eines der ersten interessanten Einblicke welche wir fanden, war der Unterschied zwischen den Fachbegriffen „Lernmethode“ und „Lerntechnik“.

Nämlich besteht da der wesentliche Unterschied darin, dass Lerntechniken einzelne spezifische Schritte im Lernprozess sind, während eine Lernmethode eine Kollektion von Lerntechniken darstellt und die allgemeine Lernstrategie beschreibt.[1] Dabei wurden wir auf ein paar wenige Lernmethoden und -techniken aufmerksam, welche angeblich das Lernen vereinfachern sollten, wie beispielsweise die Lernmethoden SQ3R[2] oder KWL[3], während eine Lerntechnik beispielsweise das Erschaffen von Verknüpfung zu bestehendem Wissen darstellt. Diese wurden hauptsächlich von Ritschel-Gotal (2023) übernommen.

Zeit- & Pausenmanagement

Dieses Thema war schon von Anfang an wichtig für uns, vor allem wegen unseren Agenda- und Lerntimer-features, also haben wir hauptsächlich im Bereich des Lerntimers, auch bekannt als der „Pomodoro-Timer“, und die Zeiteinplanung recherchiert.

[1] Weber, 2023.
[2] Ritschel-Gotal, 2023.
[3] Ritschel-Gotal, 2023.

Zur erfolgreichen Zeiteinteilung gehört für uns auch die Fähigkeit, sinnvoll Pausen zu machen, weswegen wir auch Informationen zu wann und wie man Pausen machen soll recherchierten. Dies war für uns vor allem wichtig, da wir in unserem Umfeld auch dies immer wieder als Problem beobachtet haben.

Umfragedesign

Da wir später eine Umfrage für die Schüler:innen der Kantonsschule Baden erstellen wollten, war es für uns wichtig, dass wir auch das Umfragedesign berücksichtigen. Der Hauptteil der Informationen dafür kam nicht aus dem Internet, sondern aus einem Lehrmittel[4], welches wir netterweise von den PPP-Lehrpersonen Frau Suter und Herr Schmocker erhalten hatten. Da war hauptsächlich das Kapitel `„Sozialwissenschaftliche Methoden"(S. 191 - 218)"`

2.2.4 Literatur

Für die Literaturstudie liehen wir die folgenden zwei Bücher aus, welche uns einen differenzierten Standpunkt geben sollten:

- `Effektiver Lernen für Dummies (2. Auflage) (Ebbert, 2019) von Dr. Birgit Ebbert`
- `Lernpsychologie (6. Auflage) (Edelmann, 2000) von Walter Edelmann`

Diese zwei Bücher sollten uns einen guten Überblick über das Lernen aus zwei verschiedenen Perspektiven - der direkten Anwendung mit „Effektiver Lernen für Dummies“ und der wissenschaftlichen Perspektive mit „Lernpsychologie“ - geben. Dabei führten wir laufend Notizen und integrierten diese in unser Interviewfragendossier & unsere Umfrage, welche in `Abschnitt 2.3 & 2.4` näher beschrieben werden.

In den Büchern lernten wir viel über was einen guten Lernerfolg voraussetzt. Dazu gehören unter anderem verschiedene Lerntechniken & -strategien, wie man sich erfolgreich auf eine Prüfung vorbereitet und die Bedeutung

[4]Ludwig und Hartmeier, 2019.

von einem guten Lernumfeld, d.h. alle Faktoren um das éernen welche dies unterstützen. Unter anderem beinhaltet dies, dass man einen guten Schlafrhythmus hat, genug sinnvolle Pausen macht, und auch psychologische Faktoren wie ein gutes Mindset. Auch ein paar wenige wissenschaftliche Begriffe waren für uns wichtig, darunter das sogenannte „Assoziationslernen“.[5]. Unter dem Assoziationslernen versteht Edelmann (2000) das Lernen durch der Schaffung von Verknüpfungen zu bisher gelerntem Wissen. Somit kann das Gehirn einfacher neues Wissen aufnehmen und verarbeiten.

Insgesamt konnten wir so unser Vorwissen zu unseren für uns wichtigen Themenbereichen noch um einiges ausbauen, damit wir uns noch besser auf die Interviews vorbereiten konnten.

v– OLD TEXT –v

2.3 Interviews

2.3.1 Vorgehensweise

Wir wussten, das ein wichtiger Aspekt unserer Recherche Interviews mit Expert:innen sein würden, da sie uns vermutlich am Besten weiterhelfen könnten, da sie sich gut mit dem Thema auskennen und viel persönliche Erfahrungen mitbringen. Somit können sie auch direkt auf unsere Fragen eingehen und uns auch für die Umfrage persönlich Feedback geben.

Zu diesem Ende wählten wir zwei PPP-Lehrpersonen der Kantonsschule Baden, Frau Suter und Herr Schmocker, aus. Sie haben beide extensive Erfahrung mit der Lernpsychologie und, dank ihrer Tätigkeit als Lehrpersonen, auch viel Kontakt mit Kantischüler:innen. Deswegen stellten sie die idealen Interviewpartner für uns dar. Wir hatten bereits durch unserem Erstbetreuer, Herr Schneider, zwei Theorie-Dokumente erhalten, welche uns mit der Umfragetheorie und der Durchführung von Interviews helfen sollte.

[5]Edelmann, 2000.

Themenwahl

Wir wollten uns hauptsächlich auf die folgenden vier Themen konzentrieren:

- Lernmethoden & -techniken
- Stressmanagement
- Pausenmanagement
- Zeitmanagement

Diese Themen wählten wir, da wir sie als wichtig für den Schulalltag und das Lernen empfanden und somit auch als besonders relevant für unsere Web-Applikation sahen, da wir dies darin einbauen könnten. Dies geht am einfachsten beim Zeitmanagement, da dies sehr stark in unsere geplanten Agenda- und Pomodoro-Timer-Features einfliesst, aber vor allem unsere geplanten Lern- und Daily-Tipps sind da besonders versatil.

Bei all diesen Themen haben wir auch schon einen persönlichen Bezug dank unserer bisherigen Schulkarriere, und können so uns auch auf unsere eigenen Erfahrungen und Unsicherheiten stützen. Als Letztes haben wir dann auch nach Feedback für unsere Umfrage eingefügt, da wir professionelles Feedback dafür einholen wollten und das Interview dafür die beste Gelegenheit ist, vor allem da die PPP-Lehrpersonen uns schon das Theorie-Dokument zum Umfragedesign zur Verfügung gestellt hatten.

2.3.2 Interviewfragebogen & Interviewfragen

Um sowohl uns selbst als auch die Interviewpartner auf das Interview adequat vorzubereiten, erstellten wir einen Interviewfragebogen, in dem all unsere geplanten Fragen aufgelistet sind. Die Fragen wurden nach den vier Themenblöcken geordnet und nummeriert, um eine klare Struktur zu erstellen, in der das Interview verlaufen soll.

Die jeweiligen Fragen haben wir gesammelt, indem wir uns einerseits überlegten, wo wir Unsicherheiten sahen oder generell mehr wissen wollte, andererseits wo ein genauer Bezug zu den Schüler:innen der Kantonsschule Baden oder die persönlichen Erfahrungen der Interviewpartner wichtig

sein könnten. Ebenfalls benutzten wir ChatGPT um ein paar Vorschläge zu generieren, jedoch sind alle Fragen selbstständig ausgedacht und formuliert worden. (Vergleich *KI Nachweis*)

Interviewfragen: Lernverhalten

Als Erstes überlegten wir uns theoretische Fragen zum Lernverhalten allgemein, aufgeteilt in `Lernmethoden` & `Lerntechniken`. Hier wird nochmals der Unterschied zwischen einer Lernmethode und Lerntechnik wichtig. Wie bereits in `Abschnitt 2.2.2` beschrieben, ist eine Lernmethode eine allgemeine Strategie zum Lernen und kann von mehreren Techniken gebrauch machen, während eine Lerntechnik ein spezifisches Element des Lernens darstellt (Weber, 2023).

Lernmethoden

Zum Thema Lernmethoden überlegten wir uns zwei sehr spezifische Fragen:

- Was sind, Ihrer Meinung nach, die besten Lernmethoden für Schulstoff?
- Was halten Sie von Lernmethoden wie „SQ3R“ („Survey, Question, Read, Recite, Review“) oder „KWL“ („Know, Want, Learn“)? Sind solche Methoden Ihrer Meinung nach für den Schulalltag sinnvoll?

Mit diesen Fragen wollten wir nach spezifischen Lernmethoden nachforschen, und die Meinung der Interviewpartner dazu herausfinden. Dies da, wenn sich solche als sinnvoll herausstellen würden, diese eventuell in die Web-Applikation integriert oder, beispielsweise, speziell erklärt werden könnten. Somit konnten wir auch die erwähnten Lernmethoden aus der Internet-Recherche hineinarbeiten.

Lerntechniken

Die Lerntechniken stellten mit fünf Fragen das umfangreichste Segment des Interviewfragebogens dar:

- Wie sehr variiert, welche Lerntechniken am besten funktionieren, von Person zu Person?

- Welche Lerntechniken sind generell am nützlichsten, um das Gelernte so gut wie möglich zu verinnerlichen?
- Was sind Ihre „Geheimtipps“ für das Lernen von Schulstoff?
- Inwiefern hat das „Mindset“ etwas mit dem Lernen zu tun, und wie kann man das „Mindset“ verbessern, beziehungsweise was macht ein gutes „Mindset“ aus.
- Macht das Fach, für welches man lernt, einen Unterschied in welche Lerntechnik man verwenden sollte, oder gibt es eine „universelle“ Methode welche für alles anwendbar ist.

Hier ging es uns darum, anstatt spezifische Techniken zu erforschen, was für eine individuelle Person am Besten funktionieren würde. Das grundlegende Ziel war natürlich immernoch, herauszufinden, wie man dies in die Web-Applikation integrieren kann. Beispielsweise wäre es schlau, wenn nun ganz klar eine spezifische Technik empfohlen wird, diese Technik, ähnlich wie beispielsweise die Pomodoro-Technik für das Zeitmanagement, einzubauen.

Besonders interessiert waren wir am sogenannten „Mindset“ und den persönlichen Tipps der Interviewpartner:innen, da diese uns womöglich einen guten Einblick in die Materie aufgrund ihrer eigenen Erfahrung als Lehrperson geben könnten. Natürlich sind alle Fragen auch besonders relevant für unsere Lerntipps.

Interviewfragen: Pausenmanagement

Dieser Abschnitt ist vor allem für unseres geplantes „Pomodoro“-Feature wichtig, hat aber auch eine Relevanz für unsere Lerntipps. Somit stellten wir die folgenden drei Fragen:

- Was sind gute Anzeichen, dass man beim Lernen eine Pause braucht?
- Wie lange sollte eine Pause während dem Lernen sein und was für Faktoren könnten die „ideale“ Pausenlänge beeinflussen?

- Ist es besser, Pausen fix einzuplanen oder sie „nach Gefühl“ durchzuführen?

Somit lag der Hauptfokus darauf, festzulegen wann und wie Lange eine Pause sein soll, anstatt auf wie man sie verbringen sollte.

Interviewfragen: Stressmanagement

Das Stressmanagement war ein für uns durch persönliche Erfahrungen bereits sehr vertrautes Problem und eines, welches wir so gut wie möglich in unsere Lerntipps einbauen wollten. Ob und wie andere Schüler:innen oft Stress empfinden, wollten wir mit der Umfrage herausfinden, weswegen es auch hier hauptsächlich um die Erfahrungen der Lehrpersonen und um ihre Empfehlungen, wie man Stress abbauen kann, geht. Daraus entstanden diese zwei Fragen:

- Was sind die häufigsten Gründe für Prüfungsstress in der Schule und vor Prüfungen, welche Sie hier an der Kantonsschule Baden beobachtet haben?
- Was sind Ihre empfohlenen Methoden, um Stress abzubauen.

Interviewfragen: Zeitmanagement

Als letztes Segment mit konkreten Fragen kam das Zeitmanagement. Auch dies stellte sich für uns als ein vertrautes Problemfeld dar, vor allem wegen der Prokrastination. Hier ging es uns aber im Interview hauptsächlich darum, herauszufinden wie man die Zeit ausserhalb des Stundenplans einteilen sollte, was auch für unseren Agenda-Algorithmus von grosser Bedeutung ist. Auch fragten wir hier zur Pomodoro-Methode nach, da wir auch die persönlichen Meinungen der Interviewpartner:innen in Bezug nehmen wollten, da sie als Lehrpersonen das Umfeld an der Kanti Baden sehr gut kennen. Somit haben wir also folgende Fragen ausgewählt:

- Haben Sie schon einmal vom sog. „Pomodoro-Timer“ gehört, und wenn ja - was halten Sie davon?

- Was haben Sie für Tipps, um mit der Zeitplanung eine Balance zwischen Freizeit und Schule/Lernen zu gestalten?
- Was für ausserschulische Leistungserwartungen sind für Schüler Ihrer Meinung nach realistisch?
- Soll das Lernen am Wochenende Ihrer Meinung nach vermieden oder gefördert werden?

Umfragedesign

Ganz am Schluss wollten wir noch das Umfragedesign besprechen. Hierzu gibt es keine konkreten Fragen, sondern es ging uns darum, genaues Feedback von den Interviewpartner:innen zu unserer bisher erstellten Umfrage einzuholen, damit wir diese so verbessern können. Dabei achteten wir uns vor allem auch auf das Umfrage-Layout und die Art der Fragen, welche wir in der Umfrage einbauten.

2.3.3 Durchführung

Wir wussten schon früh, wer wir als unsere Interviewpartner:innen haben wollten. Wir haben uns zwei PPP-Lehrpersonen der Kantonsschule Baden ausgesucht, nämlich Herr Schmocker und Frau Suter, von welchen wir auch durch unseren Erstbewerter Theoriedokumente zum Umfrage- und Interviewdesign erhalten haben.

Somit luden wir diese zwei Lehrpersonen per E-Mail zum Interview ein, vereinbarten ein Datum und schickten ihnen jeweils etwa eine Woche vor dem Interview den fertiggestellten Interviewfragebogen und einen Link zu unserer Umfrage, damit sie sich gut vorbereiten konnten. Die vereinbarten Daten waren der `24. April 2025` mit Frau Suter und der `8. Mai 2025` mit Herr Schmocker.

Die Interviews führten wir in einem reservierten Klassenzimmer an der Kantonsschule Baden, jeweils am Mittag um `12:15` durch, mit einer Dauer von je etwa einer Stunde. Diese wurden mit der Erlaubnis der Interviewpartner aufgenommen, damit sie später besser transkribiert und analysiert

werden können und schrieben nebenbei reichliche Notizen. Ein Fehler, welcher uns hier unterlief, war, dass wir die Interviews auf Mundart führten, welches diese spätere Analyse um einiges erschwerte.

Nach dem Interview mit Frau Suter tauschten wir uns noch per E-Mail aus, um weiteres Feedback für die Umfrage, welche nach dem Feedback aus den Interviews ergänzt wurde, zu gewinnen.

2.3.4 Transkription

Als erster Schritt der formellen Analyse der Interviews transkribierten wir die jeweiligen Audio-Aufnahmen der Interviews auf Papier als Word-Dateien, um die spätere detailierte Analyse zu erleichtern. Dies war, wie in Durchführung erwähnt, aufgrund der Verwendung von Mundart nicht sehr einfach, da dies aufgrund des Mangels an Mundart-Übersetzern nun komplett manuell vorlaufen musste.

Somit wurde das Gesprochene auf Hochdeutsch übersetzt und allfällige sprachliche Füller wie beispielsweise „ähm“ wurden entfernt. Wir achteten uns immer genau darauf, dass klar ist wer wann spricht. Zu diesem Ende wurden Textabschnitte geformt, indem alles, was eine Person zu einem Zeitpunkt ununterbrochen sagt, zusammengefügt wurde. Dies sah dann beispielsweise folgendermassen aus:

> [**3:27**] **Frau Suter:** Von den effektiven Zeiten bin ich ein wenig überfragt. Ich kann mir vorstellen, dass es auch wieder draufankommt, um was es nun genau geht.

Die Zeitangabe signifiert, wann der gesprochene Abschnitt anfängt, damit man ihn zur Überprüfung leicht wiederfinden kann und die Person wer die Aussage getroffen hat. Ebenso haben wir uns als Hilfe notiert, wann in etwa welche Frage / welches Thema diskutiert wird.

2.3.5 Analyse

In der Analyse ging es uns um dreierlei: Direkte Antworten und Aussagen zu unseren Fragen, allgemeine nützliche Informationen zum Lernen und all-

gemeine nützliche Informationen zur Web-Applikation. Diese sammelten wir durch genaue Analyse der schriftlichen Transkription, dabei wurde immer für spätere Referenz auch auf die genaue Textquelle vermerkt.

2.4 Umfrage

3. Methodik – Programmieren der Web-Applikation

3.1 Erste Entscheidungen[1]

Zu Beginn stand die grundsätzliche Plattformwahl im Zentrum: native Applikation (z. B. für Smartphones) oder webbasierte Lösung. Unter Berücksichtigung von Geräteunabhängigkeit, Verteil- und Updateaufwand, Entwicklungsressourcen sowie des verfügbaren Zeitrahmens erwies sich eine Web-Applikation als zweckmässig. Sie ist plattformagnostisch im Browser nutzbar, benötigt keine Installation und lässt sich zentral aktualisieren. Zudem reduziert eine einheitliche Codebasis den Implementierungs- und Wartungsaufwand gegenüber mehreren nativen Anwendungen für unterschiedliche Betriebssysteme.

Auf Basis dieser Entscheidung fiel die Technologiewahl auf Python mit dem Microframework Flask. Ausschlaggebend waren vorhandene Vorerfah-

[1] ChatGPT (Version GPT-5): „Überarbeite den folgenden Text, damit er sprachlich und stilistisch den Standards einer wissenschaftlichen Maturitätsarbeit entspricht. Achte auf korrekte Grammatik, präzisen Ausdruck, logische Argumentation und sachlichen Stil. Behalte den ursprünglichen Sinn und Stil des Textes bei, aber formuliere ihn wissenschaftlicher und grammatikalisch korrekt. [...]. “, 04.11.2025. Antwort ganz übernommen.

rungen und eine schlanke Lernkurve. Als Entwicklungsumgebung wurde Visual Studio Code eingesetzt. Die integrierten Funktionen zur Produktivitätssteigerung (u. a. Sprachunterstützung, Linting und Code-Assistenz) unterstützen einen effizienten, nachvollziehbaren Entwicklungsprozess.

Für die kollaborative Arbeit kamen Git als Versionsverwaltung und GitHub als zentrales Remote-Repository zum Einsatz. Der Quellcode wurde dort gemeinsam versioniert und ausgetauscht; regelmässige Synchronisationen (Push/Pull) stellten einen konsistenten, aktuellen Projektstand sicher.

3.2 Anforderungen[2]

Bevor wir mit dem Programmieren der Web-Applikation beginnen konnten, mussten wir uns zuerst über die Anforderungen an die Applikation klar werden. Da es sich um eine Web-Applikation handelt, welche den Schüler:innen der Kantonsschule Baden helfen soll, mussten wir uns überlegen, welche Funktionen die Applikation beinhalten sollte und wie diese umgesetzt werden könnten.

Die Anforderungen an die Kanti Koala Web-Applikation sind wie folgt:

- **Home-Screen**: Von dem Home-Screen sollte man auf seinen Account und die Agenda zugreifen können. Zusätzlich sollte hier jeden Tag ein allgemeiner Tipp für die Kantonsschule angezeigt werden.

- **Account Management**: Die Nutzer:innen sollten sich registrieren, einloggen, ihr Passwort zurücksetzen und ihre Account-Einstellungen ändern können. Sie sollten die Möglichkeit haben, ihr Passwort zu ändern und allfälligerweise ihr Account zu löschen.

- **Agenda**: Die Nutzer:innen sollten ihren Stundenplan eintragen können, sowoul manuell wie auch durch den Import einer .ics-Datei. Ebenso

[2]ChatGPT (Version GPT-5): „Überarbeite den folgenden Text, damit er sprachlich und stilistisch den Standards einer wissenschaftlichen Maturitätsarbeit entspricht. Achte auf korrekte Grammatik, präzisen Ausdruck, logische Argumentation und sachlichen Stil. Behalte den ursprünglichen Sinn und Stil des Textes bei, aber formuliere ihn wissenschaftlicher und grammatikalisch korrekt. [...]. “, 04.11.2025. Antwort ganz übernommen.

sollte man neue Erreignisse eintragen können. Die Erreignisse sollten veränderbar sein. Die Farbe der Erreignisse sollten auch frei bestimmbar sein. Die Agenda sollte auch einen Lernzeitalgorithmus beinhalten, welcher automatisch Lernzeiten basierend auf den eingetragenen Erreignissen und den Prioritätseinstellungen der Nutzer:innen plant.

- **Notenverwaltung**: Die Nutzer:innen sollten ihre Noten für jedes Fach eintragen können. Die Noten sollten veränderbar und löschbar sein. Die Nutzer:innen sollten auch ihre Semester verwalten können, indem sie neue Semester hinzufügen, bestehende Semester bearbeiten und löschen können.

- **Lerntimer**: Die Nutzer:innen sollten einen Pomodoro-Timer verwenden können, um ihre Lernzeiten zu strukturieren. Der Timer sollte anpassbar sein, sodass die Nutzer:innen die Länge der Lern- und Pausenintervalle einstellen können.

- **UI**: Die Web-Applikation sollte ein benutzerfreundliches und ansprechendes UI haben, welches einfach zu navigieren ist.

3.3 Technische Dokumentation (System- und Datenstruktur)

Dieser Abschnitt beschreibt die technische Grundlage der Kanti Koala Web-Applikation, einschliesslich der Systemarchitektur, der Datenstruktur und der Code-Struktur, wie es auch implementiert wurde.

3.3.1 Darstellung der Systemarchitektur

Die Kanti Koala App ist als monolithische Webanwendung konzipiert, die auf einem zentralen Backend-Server läuft.

- **Frameworks:** Das Kernstück der Anwendung ist das Python-Microframework Flask. Es steuert das Routing (die Zuordnung von URLs zu Funktio-

nen), verarbeitet HTTP-Anfragen (GET, POST, usw.) und rendert die HTML-Templates für den Benutzer.

- **Komponentenübersicht (Wichtige Pakete):**
  - **Flask-SQLAlchemy**: Dient als Object-Relational Mapper (ORM) für die Datenbank. Es ermöglicht die Definition von Datenbanktabellen als Python-Klassen (Models) und vereinfacht Datenbankabfragen.
  - **Flask-Bcrypt**: Wird für die Sicherheit der Benutzerpasswörter eingesetzt. Es hasht und verifiziert Passwörter mithilfe des bcrypt-Algorithmus.
  - **Flask-Migrate**: Erleichtert Schema-Migrationen der Datenbank, wenn sich die Modelle (Tabellenstruktur) ändern.
  - **Resend**: Dient als E-Mail-API für den Versand von systemgenerierten E-Mails, insbesondere für die „Passwort vergessen"-Funktion.
  - **icalendar**: Eine Python-Bibliothek, die zum Parsen und Importieren von `.ics`-Kalenderdateien verwendet wird, um den Schulnetz-Stundenplan zu importieren.
  - **itsdangerous**: Wird verwendet, um sichere, zeitlich begrenzte Tokens zu generieren, die für die „Passwort zurücksetzen"-Links benötigt werden.
- **Server-Setup:**
  - Die Anwendung ist für den Betrieb auf DigitalOcean, einer Cloud-Platform, ausgelegt.
  - Die Datenbankkonfiguration und verschiedene Schlüssel/API Keys werden dynamisch über Umgebungsvariablen geladen.
  - Der Code unterstützt sowohl PostgreSQL (für die Produktion auf Cloud-Diensten) als auch SQLite (für wenn wir lokal entwickeln).

3.3.2 Beschreibung der Datenbank(en) und Datenstruktur

Die Datenstruktur ist in der Datei `models.py` durch SQLAlchemy-Modelle definiert. Eine detaillierte Beschreibung der einzelnen Tabellen (`User`, `Settings`, `PrioritySetting`, `Event`, `Semester`, `Subject`, `Grade`) und ihrer Attribute ist im Abschnitt „Datenbankmodelle und Schema“ ausführlicher dokumentiert.

- **Tabellen:** Die Datenbank besteht aus sieben Haupttabellen: `User`, `Settings`, `PrioritySetting`, `Event`, `Semester`, `Subject` und `Grade`, die direkt den SQLAlchemy-Klassen in `models.py` entsprechen.
- **Beziehungen (Relationships):** Die Beziehungen werden durch `db.relationship` und `db.ForeignKey` in den Modellen verwaltet.
  - **User-zentrierte Struktur:** Der `User` ist das zentrale Modell. Alle anderen Hauptdaten sind direkt oder indirekt mit ihm verknüpft:
    - `User` (1) → `Settings` (1)
    - `User` (1) → `Event` (N)
    - `User` (1) → `Semester` (N)
  - **Hierarchische Beziehungen:**
    - `Settings` (1) → `PrioritySetting` (N): Jede Einstellung hat mehrere Prioritätsregeln.
    - `Semester` (1) → `Subject` (N): Jedes Semester hat mehrere Fächer.
    - `Subject` (1) → `Grade` (N): Jedes Fach hat mehrere Noten.
  - **Kaskadierendes Löschen:** Die Beziehungen sind mit `cascade="all, delete-orphan"` konfiguriert. Das bedeutet, wenn ein übergeordnetes Objekt (z.B. ein `User` oder ein `Semester`) gelöscht wird, werden alle zugehörigen untergeordneten Objekte (z.B. alle `Events` und `Settings` des Users) automatisch mitgelöscht. Dies stellt die Datenintegrität sicher.

3.3.3 Code-Struktur

Um die Wartbarkeit und Skalierbarkeit der Anwendung zu verbessern, wurde die ursprüngliche Code-Struktur von einer einzigen `app.py`-Datei in ein modulares Python-Paket namens `kkoala` umstrukturiert. Dieser Ansatz folgt dem „Application Factory"-Pattern, einer bewährten Methode für Flask-Anwendungen. Zudem ist diese Anwendung auch die Best Practice für Flask-Anwendungen. (Muneeb, 2025)

1. **Das „Application Factory"-Pattern (`kkoala/__init__.py`)**: Das Herzstück des Pakets ist die `create_app`-Funktion. Anstatt einer globalen App-Instanz wird die Anwendung durch diesen „Factory"-Aufruf erzeugt. Dies ermöglicht es, verschiedene Konfigurationen (z.B. für Entwicklung, Test oder Produktion) dynamisch zu laden und macht die Anwendung robuster. In dieser Datei werden auch die Flask-Erweiterungen initialisiert und die Blueprints registriert.

2. **Konfiguration (`kkoala/config.py`)**: Diese Datei enthält Konfigurationsklassen (z.B. `DevConfig`, `ProdConfig`). Sie verwalten wichtige Einstellungen wie den `SECRET_KEY`, die Datenbank-URL und API-Schlüssel. Die Konfiguration wird je nach Umgebungsvariable beim Start der App ausgewählt.

3. **Blueprints für Routen (`kkoala/routes/`)**: Die Routen der Anwendung sind in „Blueprints" aufgeteilt, die eine Gruppierung von zusammengehörigen Endpunkten ermöglichen. Dies sorgt für eine saubere Trennung der Anwendungslogik:

    - **`auth.py`**: Enthält alle Routen für die Benutzerauthentifizierung (Login, Registrierung, Passwort zurücksetzen).
    - **`events.py`**: Verwaltet die API-Endpunkte für die Agenda, einschliesslich des Erstellens, Bearbeitens und Löschens von Kalendereinträgen sowie den Start des Lernalgorithmus.
    - **`grades.py`**: Beinhaltet die API für das Notenmanagement.

- **`main.py`**: Definiert die Hauptrouten der Webseite, wie die Startseite.
- **`settings.py`**: Steuert die Einstellungsseite und die zugehörige Speicherlogik.

4. **Datenbankmodelle (`kkoala/models.py`)**: Alle SQLAlchemy-Datenbankmodelle (z.B. `User`, `Event`, `Semester`) sind zentral in dieser Datei definiert. Dies erleichtert die Verwaltung der Datenstruktur und Beziehungen.

5. **Kernlogik und Hilfsfunktionen**:
   - **`kkoala/algorithms.py`**: Eine dedizierte Datei, die ausschliesslich die komplexe Logik des Lernzeitalgorithmus (LZA) enthält.
   - **`kkoala/utils.py`**: Beinhaltet wiederverwendbare Hilfsfunktionen und sogenannte „Decorators“. Die wichtigsten sind:
     - **`@login_required`**: Dieser Decorator wird über Routen platziert, die nur von angemeldeten Benutzern aufgerufen werden dürfen. Er prüft automatisch, ob ein Benutzer in der aktuellen Sitzung (Session) angemeldet ist. Falls nicht, wird der Benutzer zur Login-Seite umgeleitet.
     - **`@csrf_protect`**: Schützt Formulare und API-Endpunkte vor Cross-Site Request Forgery Angriffen, wie im Abschnitt „CSRF-Schutz“ beschrieben.
   - **`kkoala/extensions.py`**: Hier werden die Flask-Erweiterungen (SQLAlchemy, Bcrypt, usw.) initialisiert, um zirkuläre Importfehler zu vermeiden.

6. **Startpunkt (`wsgi.py`) und die WSGI-Schnittstelle**: Die Datei `wsgi.py` im Hauptverzeichnis ist der standardisierte Einstiegspunkt für den Webserver. Ihre einzige Aufgabe ist es, die `create_app`-Factory zu importieren und das daraus resultierende Flask-`application`-Objekt zu erstellen.

   Dieses Objekt ist entscheidend, da es der WSGI (Web Server Gateway Interface) Spezifikation entspricht. WSGI ist ein Python-Standard,

der als universelle Schnittstelle oder „Brücke“ zwischen dem Webserver (der Anfragen aus dem Internet empfängt) und der Webanwendung (unserem Flask-Code) dient. (Srivastav, 2022)

Für den produktiven Einsatz unserer App verwenden wir Gunicorn („Green Unicorn“), einen robusten und weit verbreiteten WSGI-HTTP-Server. Während der eingebaute Entwicklungsserver von Flask für das Testen ausreicht, ist er nicht dafür ausgelegt, eine hohe Anzahl von Anfragen zu bewältigen. Gunicorn agiert hier als leistungsfähiger „Middleman“ zwischen dem Internet und unserer Flask-Anwendung. Er kann mehrere Anfragen gleichzeitig bearbeiten, indem er mehrere „Worker“-Prozesse verwaltet, was die Leistung und Stabilität der Anwendung unter Last sicherstellt. („Launching a Flask Application with Gunicorn“, o. D.) Wenn wir Gunicorn starten, geben wir ihm den Befehl `gunicorn wsgi:application`. Er weiss dann, dass er in der Datei `wsgi.py` nach dem `application`-Objekt suchen und dieses als Startpunkt für die Anwendung verwenden muss.

3.3.4 Frontend-Struktur (Templates und Statische Dateien)

Die Benutzeroberfläche der Kanti Koala App wird dynamisch auf dem Server generiert und als fertige HTML-Seiten an den Browser des Nutzers gesendet.

- **Templates (`kkoala/templates/`)**: In diesem Verzeichnis befinden sich alle HTML-Dateien der Anwendung. Flask verwendet die Template-Engine Jinja, um diese Dateien zu verarbeiten. Jinja ermöglicht es, Python-Code direkt in HTML einzubetten, zum Beispiel um Schleifen zu erstellen (`{% for item in items %}`), bedingte Blöcke anzuzeigen (`{% if user.is_authenticated %}`) oder Variablen aus dem Backend auszugeben (`{{ user.username }}`). Dies macht die Seiten dynamisch und personalisiert. („Template Designer Documentation — Jinja Documentation“, o. D.; „Templates — Flask Documentation“, o. D.)

- **Statische Dateien (`kkoala/static/`)**: Dieses Verzeichnis enthält alle statischen Assets, die vom Browser direkt geladen werden, ohne dass der Server sie verarbeiten muss. Dazu gehören:
  - **CSS-Dateien**: Für das Styling und das visuelle Design der Anwendung.
  - **Bilder**: Logos und andere grafische Elemente.

3.4 Features

Zunächst werden alle Entscheidungen über die Features erklärt. Natürlich werden die Features auch erklärt.

3.4.1 Datenbank

Da es sich um eine Webanwendung handelt, können nicht alle Informationen des Benutzers lokal gespeichert werden. Das bedeutet erstens, dass alle Informationen extern auf einem Server gespeichert werden müssen. Zweitens müssen jetzt natürlich die Informationen jedes Nutzers gespeichert werden, und nur die Informationen eines Nutzers zu speichern, wie man es lokal tun würde, funktioniert nicht. Das bedeutete für uns zwei Dinge. Wir müssen einen Weg finden, die Informationen zu speichern, und wir müssen herausfinden, wo diese Informationen gespeichert werden sollen.

Um die Informationen zu speichern, haben wir uns für SQL-Datenbanken entschieden, da diese am einfachsten mit Flask zu verwenden sind. („Define and Access the Database“, o. D.) Wenn wir lokal arbeiten, können wir SQLite verwenden, und wenn es sich um die Produktionsumgebung handelt, können wir PostgreSQL verwenden, welches viel besser ist für eine solche Umgebung („PostgreSQL vs SQLite: The Ultimate Database Showdown“, 2025).

Die Struktur dieser Tabellen ist im Folgenden dargestellt:

Datenbankmodelle und Schema

Die Datenbank besteht aus sieben Hauptmodellen, die die Nutzerdaten und die Planungslogik abbilden.

User

Dieses Modell speichert die Authentifizierungsdetails und dient als zentraler Ankerpunkt für alle anderen Daten des Nutzers.

id Eindeutige ID des Nutzers (Primary Key).

username Der gewählte Benutzername (eindeutig, notwendig).

password Das gehashte Passwort (notwendig).

email Die E-Mail-Adresse des Nutzers (eindeutig, notwendig).

Settings

Speichert globale Einstellungen für den Lernalgorithmus, die dem `User` zugeordnet sind.

id Eindeutige ID (Primary Key).

user_id Fremdschlüssel zur `User`-Tabelle (notwendig).

learn_on_saturday Boolesche Variable, ob am Samstag gelernt werden soll (Standard: False).

learn_on_sunday Boolesche Variable, ob am Sonntag gelernt werden soll (Standard: False).

preferred_learning_time Bevorzugte Startzeit für Lernblöcke (Standard: 18:00).

study_block_color Hex-Code für die Farbe der Lernblöcke (Standard: #0000FF).

PrioritySetting
Definiert die spezifischen Parameter für jede Prioritätsstufe des Lernalgorithmus.

id Eindeutige ID (Primary Key).

settings_id Fremdschlüssel zur `Settings`-Tabelle (notwendig).

priority_level Die Prioritätsstufe (Integer, notwendig).

color Die dem Prioritätslevel zugeordnete Farbe (Hex-Code, notwendig).

max_hours_per_day Maximale Lernstunden pro Tag für diese Priorität.

total_hours_to_learn Die gesamte zu lernende Stundenanzahl für diese Priorität.

Event
Speichert Kalendereinträge des Nutzers sowie Metadaten für den Planungsalgorithmus.

id Eindeutige ID (Primary Key).

user_id Fremdschlüssel zur `User`-Tabelle (notwendig).

title Titel des Ereignisses.

start Startzeitpunkt im ISO-Format (notwendig).

end Endzeitpunkt im ISO-Format (optional).

color Farbe des Ereignisses.

priority Prioritätsstufe (Integer).

recurrence Wiederholungsregel des Ereignisses.

recurrence_id Eindeutige ID zur Gruppierung wiederkehrender Ereignisse.

all_day Boolesche Variable, ob das Ereignis ganztägig ist (Standard: False).

locked Boolesche Variable für den Algorithmus; `True` bedeutet, das Ereignis ist fixiert (Standard: True).

exam_id ID des zugehörigen Examens, falls zutreffend.

Semester, Subject und Grade
Diese Modelle bilden die akademische Hierarchie ab.

Semester Speichert akademische Abschnitte. Enthält **user_id** (Fremdschlüssel) und **name**.

Subject Speichert Fächer innerhalb eines Semesters. Enthält **semester_id** (Fremdschlüssel) und **name**.

Grade Speichert Bewertungen für Fächer. Enthält **subject_id** (Fremdschlüssel), **name**, **value**, **weight** und **counts**.

Die Daten in der Datenbank werden unten erklärt. Was wichtig bei den Datenbanken ist, ist dass die beiden Datenbanken zusammen verbunden sind, mithilfe eines Foreign Keys. Dieser Foreign Key befindet sich in der Eventdatenbank, unter dem Name „user_id“. Dieser sagt uns, welcher Event zu welchem User gehört.

Beziehungsstruktur Die Abhängigkeiten und Kaskadenlöschungen (z.B. ein gelöschter `User` löscht alle seine `Events`, `Semesters` und `Settings`) sind über Fremdschlüsselverweise in allen untergeordneten Tabellen implementiert. Die zentralen Verbindungen sind:

- `User` → `Settings` (1:1), `Events` (1:n), `Semester` (1:n)
- `Settings` → `PrioritySetting` (1:n)
- `Semester` → `Subject` (1:n)
- `Subject` → `Grade` (1:n)

3.4.2 Serververbindung

Da es um eine Web-App handelt, müssen die SQL-Datenbanken irgendwo extern gespeichert werden, wo man sie jederzeit abrufen kann. Das heisst, die SQL-Datei muss auf ein Server gespeichert werden. Wir haben uns schliesslich für die Cloud-Applikation „DigitalOcean“ entschieden. Der Grund dafür war, weil wir erhalten gratis Credits mit einem Student Developer Pack. Gleichzeitig konnten wir auch die Webseite auf derselben Plattform hosten, welches den Prozess vereinfacht hat, da wir alles intern verbinden konnten.

Die Domaine, die wir kauften, wurde wegen dem gleichen Grund gekauft, nämlich, dass wir mit demselben Student Developer Pack die Top-Level-Domaine .app gratis erhalten haben. Die Domaine heisst kantikoala.app.

3.4.3 Authentifizierung

Um eine App mit verschiedenen Nutzern zu haben, brauchen wir ein gutes Authentifizierungssystem. Das bedeutet, dass es eine Anmelde- und Registrierungsfunktion sowie eine Option zum Vergessen des Passworts, eine Option zum Ändern des Passworts und schliesslich auch eine Option zum Löschen des Kontos geben muss.

Natürlich können wir ein Passwort nicht im Klartext speichern, denn das wäre ein Sicherheitsrisiko und ein ethisches Risiko, weil wir als Entwickler dann die Passwörter der einzelnen Konten einsehen können. Die einfachste Lösung für dieses Problem besteht darin, das Kennwort zu hashen. Unter Passwort-Hashing versteht man die algorithmische Umwandlung eines Passworts in Chiffretext oder eine unumkehrbar verschleierte Version seiner selbst („What is Password Hashing?“, 2022). Ein anderes wichtiges Konzept ist das Salting. Salting ist die Praxis, zufällige Daten (ein „Salt“) zu einem Passwort hinzuzufügen, bevor es gehasht wird. Dies verhindert Angriffe mit vorgefertigten Tabellen (Rainbow Tables), da das gleiche Passwort mit unterschiedlichen Salts zu unterschiedlichen Hashes führt („Rainbow Table Attacks: How They Work and How to Defend Against Them“, o. D.). Glücklicherweise kümmert sich das Flask-Bcrypt-Modul um all diese Dinge für uns, sodass wir uns keine Sorgen machen müssen, wie wir das Passwort

hashen und salten. („Flask-Bcrypt“, o. D.)

Sobald man sich anmeldet oder registriert, wird man auf die Startseite der Website weitergeleitet.

Sehr wichtig bei jedem Anmeldesystem ist natürlich eine Option das Passwort zurückzusetzen wenn man es vergisst. Um so ein System zu haben, ist es wichtig, dass man den Link, um das Passwort zurückzusetzen, nur einmal verwenden kann. Um dies zu erreichen, haben wir den Token, welches gebraucht wird, um zu überprüfen, dass das Passwort für den richtigen User zurückgesetzt wird, mit dem Hash vom alten Passworts generiert. Wir sind dank Custic (2024) auf diese Idee gekommen. Das Prinzip funktioniert, da das Passwort sich ändern wird, und somit kann man den gleichen Token nicht wieder verwenden. Zudem brauchen wir, um die E-Mail zu verschicken, eine API, damit wir sie mit unserer eigenen Domaine verschicken können. Wir haben uns für die API von Resend entschieden, da sie eine kostenlose Stufe hat, die für unser Projekt ausreicht.

Im Falle, dass ein Nutzer zum Beispiel mit einer E-Mail, die schon gebraucht wird, sich registrieren will, oder mit einem falschen Passwort sich anmelden will, bekommt der Nutzer eine Fehlermeldung angezeigt, die ihm sagt, was falsch gelaufen ist, mithilfe von Flash-Nachrichten, welche eine Funktion von Flask ist (Acsany, o. D.).

3.4.4 Agenda

Das Hauptmerkmal unserer App ist die Agenda. Dieser Terminkalender muss leicht verständlich sein, und man muss die folgenden drei Dinge tun können: einen Termin erstellen, einen Termin bearbeiten und einen Termin löschen.

Jedes Ereignis sollte die folgenden Informationen enthalten: einen Titel, die Start- und Endzeit, ob es sich um ein wiederkehrendes Ereignis handelt, eine Farbe und seine Priorität, die für den Algorithmus wichtig sein wird. Man kann auch einstellen, ob der Event sich um ein All-Day Event handelt, oder ein normales Event ist, mit Start- und Endzeit. Die Benutzer-ID wird automatisch übermittelt, wenn ein Ereignis erstellt wird, je nachdem, wer gerade angemeldet ist.

Zur Anzeige des Ereignisses haben wir ein Modul namens FullCalendar verwendet, eine JavaScript-Bibliothek zur Anzeige eines Kalenders. Wir haben uns für dieses Modul entschieden, weil es das erste Ergebnis war, das bei der Suche nach einer Bibliothek mit seinen Fähigkeiten auftauchte.

Was man mit dieser Agenda auch machen kann, ist, seinen Stundenplan aus dem `schulnetz`, zu importieren. Das ist die Plattform, die von der Kantonsschule verwendet wird, um den Schüler:innen ihren Stundenplan anzuzeigen. Da jeder Kanti-Schüler seinen Stundenplan auf `schulnetz` hat, dachten wir, es wäre nützlich, wenn man ihn von dort importieren könnte, anstatt jede Stunde manuell eingeben zu müssen. Unsere Agenda kann also Agenden im .ics-Format importieren, das ist das Format, in dem `schulnetz` seine Kalender exportiert. Es gibt keine direkte Funktion, um den Stundenplan zu importieren, also müssen die Nutzer:innen einfach zuerst ihren Stundenplan auf `schulnetz` exportieren, und dann die .ics-Datei in unsere App importieren.

3.4.5 Der Lernzeitalgorithmus

Der Lernzeitalgorithmus (LZA) ist der Kern unserer Web-App. Er automatisiert die Planung der notwendigen Lernzeiten für die anstehenden Prüfungen eines Nutzers. Wir bezeichnen unseren Mechanismus als Algorithmus, da er die formalen Kriterien eines Algorithmus erfüllt: Jeder Planungsschritt ist ausführbar (existierende Funktionen), deterministisch und determiniert (gleiche Eingabedaten führen stets zur gleichen Priorisierung und Planung). Zudem ist die Anzahl der zur Erstellung des Lernplans notwendigen Schritte endlich (Finitheit), wodurch der Mechanismus garantiert terminiert und eine strukturierte Ausgabe (den Lernplan) basierend auf den Eingabedaten (Benutzereinstellungen und Prüfungstermine) liefert. („Algorithmus“, o. D.)

Eingabeparameter und Planungsziel

Der Lernzeitalgorithmus (LZA) verwendet globale Benutzereinstellungen sowie prüfungsspezifische Prioritätseinstellungen als Eingabeparameter, um eine optimale Lernplanung zu gewährleisten.

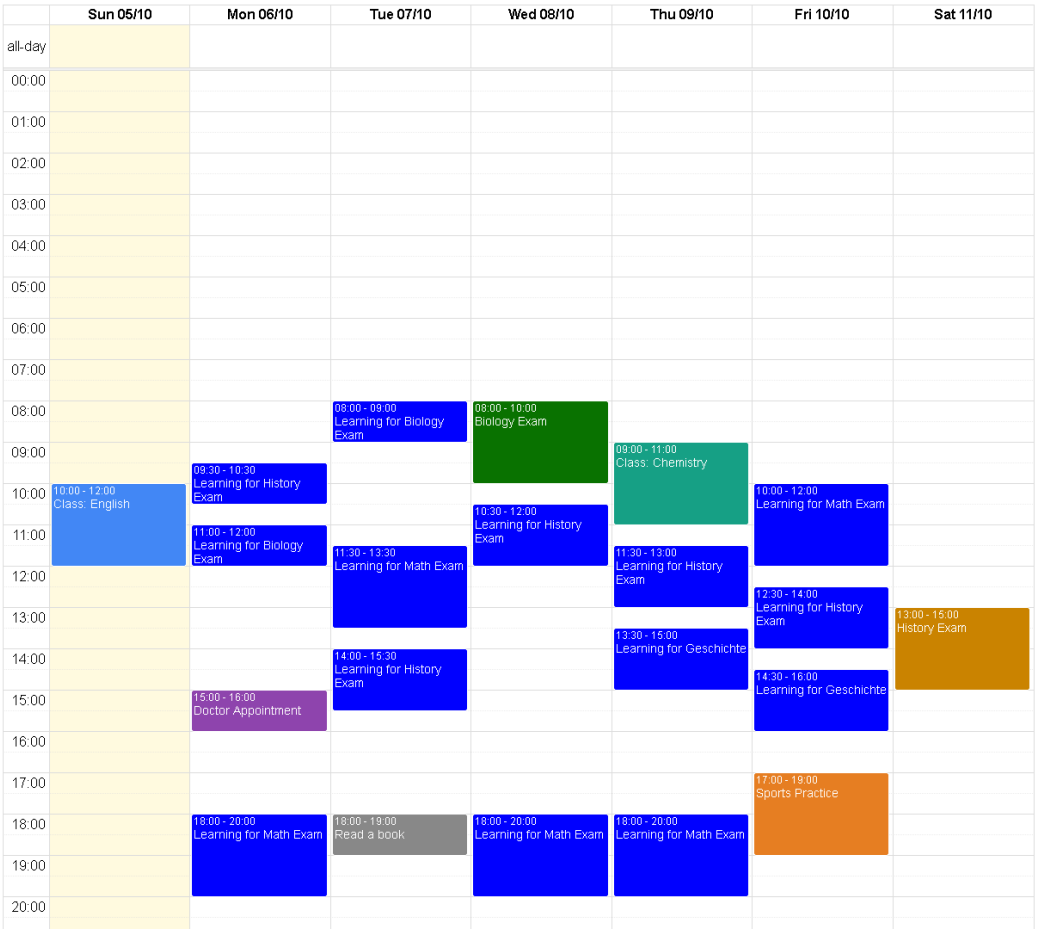

Abbildung 3.1: Beispiel einer Agenda, mit Lernblöcke von der LZA

Das zentrale Planungsziel des LZA ist es, die definierten totale Lernstunden für jede Prüfung innerhalb des gültigen Lernfensters zu erreichen, während das tägliche Maximum und alle bestehenden Kalenderkonflikte des Nutzers strikt eingehalten werden.

Globale Parameter
Diese Einstellungen gelten für den gesamten Planungszeitraum:

- **Lernen am Samstag**
  Definiert, ob der Algorithmus Lernblöcke an Samstagen planen darf.
- **Lernen am Sonntag**
  Definiert, ob der Algorithmus Lernblöcke an Sonntagen planen darf.
- **Bevorzugte Lernzeit**
  Die bevorzugte Uhrzeit am Tag, zu der die Platzierung von Lernblöcken primär angestrebt wird.

Prüfungsspezifische Parameter (Pro Prioritätstufe)
Diese Werte werden basierend auf der Priorität jeder Prüfung zugewiesen:

- **Tägliches Maximum**
  Die maximale Stundenzahl, die pro Tag für Prüfungen dieser Priorität geplant werden darf.
- **Total Lernstunden**
  Die gesamte Anzahl an Lernstunden, die für Prüfungen dieser Priorität absolviert werden muss.

Ablauf und Planungsstrategie

Der Algorithmus arbeitet iterativ und bearbeitet alle als Prüfung markierten Ereignisse in aufsteigender Reihenfolge ihrer Priorität. Eine niedrigere Prioritätsnummer kennzeichnet dabei eine höhere Wichtigkeit, um eine optimale Ressourcenverteilung zu gewährleisten.

1. **Zyklische Neuberechnung der Anforderungen (Recycling)**:

- **Flexibilitätsbereinigung**: Alle vom System selbst geplanten, aber nicht gesperrten (gesperrt sind die Blöcke, die vom Nutzer bearbeitet worden sind) Lernblöcke für die aktuelle Prüfung werden gelöscht. Dies ermöglicht eine dynamische, optimale Neuplanung, falls sich die Rahmenbedingungen (z.B. neue Events, geänderte Prioritätseinstellungen) geändert haben.
- **Soll-Stunden-Ermittlung**: Die noch zu erbringende Lernzeit wird neu berechnet. Hierbei werden alle bereits absolvierten Stunden sowie Stunden, die durch manuelle oder vom Nutzer gesperrte Blöcke abgedeckt sind, von den Gesamtsollstunden abgezogen.

2. **Rückwärts-Iterative Planung**:

   - Die Planungsstrategie ist eine Rückwärtsiteration: Sie beginnt beim Prüfungstermin und arbeitet sich tageweise, jedoch maximal drei Wochen, bis zum aktuellen Datum vor. Diese Zeitspanne wurde auf Basis unserer Recherche als optimaler Zeitraum für eine effektive Prüfungsvorbereitung festgelegt. Dieses Vorgehen stellt sicher, dass die Lernblöcke mit höchster Dringlichkeit (die Tage, die am nächsten zur Prüfung liegen) zuerst belegt werden.
   - An jedem Tag wird die maximale Lernzeit für diese spezifische Prüfung ermittelt. Diese ergibt sich aus der Differenz zwischen dem täglichen Maximum und den Stunden, die bereits an diesem Tag für die Prüfung geplant wurden. Dadurch wird das tägliche Zeitlimit zuverlässig eingehalten und eine Überlastung vermieden.

3. **Platzierung und strikte Konfliktvermeidung**:

   - **Bevorzugter Slot**: Es wird primär versucht, einen Lernblock in der vom Nutzer festgelegten bevorzugten Lernzeit zu platzieren.
   - **Konfliktprüfung**: Die Verfügbarkeit des Slots wird gegen alle Kalendereinträge des Nutzers für den aktuellen Tag geprüft. Dabei wird ein 30-minütiger Puffer vor und nach jedem bestehenden Ereignis (wie Sport oder Arzttermin) beachtet, um knappe

Überlappungen und unnötigen Stress zu vermeiden.

- **Alternative Slots**: Falls die bevorzugte Zeit belegt ist, sucht ein dediziertes Modul den grössten verfügbaren, konfliktfreien Zeitabschnitt des Tages, um die Platzierung zu maximieren.
- **Echtzeit-Aktualisierung**: Nach der erfolgreichen Generierung und Speicherung eines Lernblocks wird dieser sofort zur Liste der aktuellen Kalendereereignisse hinzugefügt. Dieser Mechanismus ist entscheidend, um sicherzustellen, dass alle unmittelbar nachfolgenden Planungsversuche am selben Tag diesen neu erstellten Block als bereits belegt berücksichtigen und somit Überlappungen ausgeschlossen sind.

4. **Ergebnisrückgabe und Zusammenfassung**:

   - Nachdem alle Prüfungen bearbeitet wurden, gibt der Algorithmus eine detaillierte Zusammenfassung des Planungsvorgangs zurück.
   - Diese Zusammenfassung informiert den Nutzer in einem Popup über die Gesamtzahl der hinzugefügten Lernblöcke und die Gesamtstunden, die erfolgreich geplant wurden.
   - Zusätzlich wird für jede einzelne Prüfung der Planungsstatus (erfolgreich / nicht erfolgreich) und die geplante Stundenanzahl angezeigt.

3.4.6 Daily Tipps

Ein relativ wichtiges Feature unserer App sind die Daily Tipps. Es sollte jeden Tag ein neuer Tipp an den Nutzer gezeigt werden auf der Startseite, entweder über die Kantonsschule oder allgemeine Lerntipps. Die einfachste Möglichkeit, dies zu implementieren, ist eine simple Modulo-Rechnung.

$$\text{Tipp des Tages} = (\text{Tag des Jahres}) \bmod (\text{Anzahl der Tipps})$$

Somit wird an einem bestimmten Tag nur einen Tipp gezeigt und über das ganze Jahr sollten alle Tipps gezeigt werden (da wir sowieso weniger als

365 Tipps haben).

3.4.7 Notenorganisation

In der Kantonsschule muss man immer wieder Prüfungen schreiben. Die Noten, die man erhält, sind dann wichtig für die Promotion in die nächste Stufe. Deswegen haben wir ein Feature, in dem man seine Noten pro Semester speichern kann. Jedes Semester hat schon die jeweiligen Fächer, die man dann in diesem Semester haben würde, geladen. Man kann natürlich immer noch Fächer löschen und hinzufügen. Man kann mit diesem Feature dann seine Durchschnitte pro Fach und Semester sehen. Ebenfalls kann man mit dem Notenrechner sehen, welche Note man in einem Fach brauchen würde, um einen bestimmten Schnitt in diesem Fach zu haben.

3.4.8 Lerntimer

Ein weiteres wichtiges Feature unserer App ist der Lerntimer. Dieser Timer basiert auf der Pomodoro-Technik, welche im Recherche Teil dieses Bericht erklärt wurde. Mit der Recherche, die wir führten, fanden wir es wichtig, ein solcher Lerntimer zu implementieren, da es eine sehr effektive Lerntechnik ist. Der Timer hat die Standard-Einstellungen von 25 Minuten Lernen und 5 Minuten Pause, welche man aber auch ändern kann, falls man das möchte.

3.4.9 Lerntipps

Ein weiteres Feature unserer App sind die Lerntipps. Diese sind in verschiedene Kategorien aufgeteilt, wie zum Beispiel Zeitmanagement, Pausenmanagement, Stressmanagement und Lernmethoden. In jeder Kategorie gibt es verschiedene wissenschaftlich fundierte Tipps, welche wir aus unserer Recherche und den Interviews gesammelt haben. Diese Tipps sollen den Nutzer:innen helfen, ihr Lernverhalten zu verbessern und effektiver zu lernen.

3.5 Sicherheitskonzept

Neben der reinen Authentifizierung wurden weitere grundlegende Sicherheitsmassnahmen implementiert, um die Daten der Nutzer und die Integrität der Anwendung zu schützen.

3.5.1 Datenspeicherung und Passwort-Sicherheit

Die Sicherheit der Benutzerdaten hat höchste Priorität. Wie im Abschnitt zur Authentifizierung beschrieben, werden Passwörter niemals im Klartext gespeichert. Stattdessen wird die Flask-Bcrypt-Bibliothek verwendet, um von jedem Passwort einen kryptografischen Hash zu erzeugen. Beim Login-Vorgang wird das eingegebene Passwort ebenfalls gehasht und dieser Hash wird mit dem in der Datenbank gespeicherten Hash verglichen. Da dieser Prozess unumkehrbar ist, kann selbst bei einem direkten Zugriff auf die Datenbank das ursprüngliche Passwort nicht wiederhergestellt werden.

3.5.2 Transportverschlüsselung (HTTPS)

Die gesamte Kommunikation zwischen dem Browser des Nutzers und unserem Server wird durch das HTTPS-Protokoll verschlüsselt. Dies wird durch ein SSL/TLS-Zertifikat realisiert, das auf unserem Server bei DigitalOcean installiert ist. Die Verschlüsselung stellt sicher, dass alle übertragenen Daten, von Login-Informationen über Kalendereinträge bis hin zu Noten, vor dem Abhören durch Dritte geschützt sind. Ein Angreifer in einem öffentlichen WLAN könnte beispielsweise die Daten nicht mitlesen. Der Browser zeigt dies durch ein Schlosssymbol in der Adressleiste an und garantiert so eine sichere Verbindung zur Domain `kantikoala.app`. („What is HTTPS?“, o. D.)

3.5.3 CSRF-Schutz

Neben der reinen Authentifizierung ist es entscheidend, die Aktionen eines angemeldeten Benutzers abzusichern. Eine der häufigsten Schwachstellen in Webanwendungen ist die Cross-Site Request Forgery (CSRF). Bei einem

CSRF-Angriff bringt ein Angreifer den Browser eines authentifizierten Benutzers dazu, eine unerwünschte Aktion in einer Webanwendung auszuführen, bei der der Benutzer gerade angemeldet ist. Dies geschieht, ohne dass der Benutzer es merkt. So könnte ein Angreifer beispielsweise einen Benutzer dazu verleiten, auf einen bösartigen Link zu klicken, der im Hintergrund unbemerkt das Passwort des Benutzers ändert oder sein Konto löscht. (Shaji, 2022)

Um dies zu verhindern, haben wir das „Synchronizer Token Pattern“ implementiert, eine von „Cross-Site Request Forgery Prevention Cheat Sheet“ (o. D.) empfohlene Methode. Das Prinzip ist einfach, aber sehr wirksam:

1. Für jede Benutzersitzung wird ein einzigartiges, geheimes und unvorhersehbares Token generiert und auf dem Server gespeichert.

2. Dieses Token wird in alle Formulare, die eine Zustandsänderung bewirken (z.B. das Ändern von Einstellungen oder das Löschen eines Kontos), als verstecktes Feld eingebettet.

3. Wenn der Benutzer das Formular abschickt, wird das Token zusammen mit den anderen Formulardaten an den Server gesendet.

4. Der Server vergleicht das vom Client gesendete Token mit dem in der Sitzung gespeicherten Token. Stimmen die beiden nicht überein, wird die Anfrage abgelehnt.

Da ein Angreifer auf einer fremden Website dieses geheime Token nicht kennen kann, schlägt der Fälschungsversuch fehl.

In unserer Flask-Anwendung haben wir diese Logik mithilfe eines eigenen Decorators (`@csrf_protect`) umgesetzt. Dieser Decorator wird auf alle Routen angewendet, die Daten durch `POST`-, `PUT`- oder `DELETE`-Anfragen ändern. Bei Standard-HTML-Formularen wird das Token als verstecktes `<input>`-Feld übergeben. Für unsere dynamischen Agenda-Funktionen, die auf AJAX basieren, wird das Token aus einem Meta-Tag ausgelesen und in einem benutzerdefinierten HTTP-Header (`X-CSRF-Token`) mit jeder Anfrage gesendet.

Dies stellt sicher, dass jede datenverändernde Aktion, die in unserer Applikation ausgeführt wird, legitim vom Benutzer und von unserer eigenen Webseite stammt.

Eine wichtige Anmerkung ist, dass es schon Module wie `Flask-WTF` gibt, die CSRF-Schutz bieten. Wir haben uns jedoch entschieden, unseren eigenen Decorator zu schreiben, um ein tieferes Verständnis für die Funktionsweise von CSRF-Schutzmechanismen zu erlangen und die Kontrolle über die Implementierung zu behalten.

3.6 Tests

Um die Qualität und Zuverlässigkeit der Kanti Koala Web-App sicherzustellen, wurden verschiedene Testfälle erstellt. Diese Tests decken sowohl die Backend-Logik als auch die Frontend-Funktionalität ab. Die genauen Testfälle und deren Ergebnisse sind im Anhang dokumentiert.

4. Schlussfolgerung und Ausblick

Das Ziel dieser Maturitätsarbeit war es, eine Web-App zu entwickeln, die Kantonsschüler:innen dabei unterstützt, ihr Lernen effizienter zu organisieren und zu planen. Mit der Kanti Koala App haben wir eine Lösung geschaffen, die nicht nur eine benutzerfreundliche Agenda bietet, sondern auch einen intelligenten Lernzeitalgorithmus integriert, der auf wissenschaftlichen Erkenntnissen basiert. Die App ermöglicht es den Nutzer:innen, ihre Prüfungen und Lernzeiten effektiv zu verwalten, was zu einer besseren Vorbereitung und letztlich zu besseren schulischen Leistungen führen kann.

Dabei haben wir, als die Entwickler, wertvolle Erfahrungen in der Webentwicklung gesammelt, insbesondere im Umgang mit Flask, Datenbanken und Sicherheitsaspekten. Die Implementierung des Lernzeitalgorithmus stellte eine besondere Herausforderung dar, da er sowohl flexibel als auch robust sein musste, um den unterschiedlichen Bedürfnissen der Nutzer gerecht zu werden.

Die App ist aber lang noch nicht fertig, denn es gibt noch viele Möglichkeiten zur Erweiterung. Als erstes würden wir sie gerne noch von Schüler:innen testen lassen, damit wir genau wissen, was unsere Nutzer:innen wollen und brauchen. Basierend auf diesem Feedback könnten wir dann weitere Features implementieren.

Literaturverzeichnis

Acsany, P. (o. D.). *Add Logging and Notification Messages to Flask Web Projects*. Real Python. Zugriff 2. November 2025 unter https://realpython.com/flask-logging-messages/

Algorithmus. (o. D.). Studyflix. Zugriff 4. November 2025 unter https://studyflix.de/informatik/algorithmus-4244

Blakley, J. (2024, 15. März). *Burnout on the Rise: Global Survey Exposes Alarming Trends in Workplace Stress*. The Alert Program. Zugriff 10. Oktober 2025 unter https://alertprogram.com/burnout-on-the-rise/#:~:text=Burnout%20is%20defined%20by%20the,worsening%20trend%20in%20mental%20exhaustion.

Contributors, W. (2019, 20. September). *Pomodoro Technique*. Wikimedia Foundation. Zugriff 10. Oktober 2025 unter https://en.wikipedia.org/wiki/Pomodoro_Technique

Cross-Site Request Forgery Prevention Cheat Sheet. (o. D.). Zugriff 21. Oktober 2025 unter https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

Custic, P. (2024, 16. Januar). *Yet Another Password Reset Tutorial in Flask*. Zugriff 7. Mai 2025 unter https://freelancefootprints.substack.com/p/yet-another-password-reset-tutorial

Define and Access the Database. (o. D.). Flask Pallets Projects. Zugriff 18. Mai 2025 unter https://flask.palletsprojects.com/en/stable/tutorial/database/

Documentation. (o. D.). Zugriff 26. März 2025 unter https://fullcalendar.io/docs

Ebbert, D. B. (2019). *Effektiver Lernen für Dummies (2. Auflage)*. WILEY-VCH Verlag GmbH & Co. KGaA.

Edelmann, W. (2000). *Lernpsychologie (6. Auflage)*. Beltz.

Flask-Bcrypt. (o. D.). Read the Docs. Zugriff 4. November 2025 unter https://flask-bcrypt.readthedocs.io/en/1.0.1/

How to Deploy a Flask App and Postgres Database to Render. (2023, 13. Januar). Pretty Printed. Zugriff 18. Mai 2025 unter https://www.youtube.com/watch?v=IBfj_0Zf2Mo

icalendar. (o. D.). Zugriff 2. April 2025 unter https://pypi.org/project/icalendar/

Launching a Flask Application with Gunicorn. (o. D.). CodeSignal. Zugriff 31. Oktober 2025 unter https://codesignal.com/learn/courses/introduction-to-flask-basics/lessons/launching-a-flask-application-with-gunicorn

Lerntechniken: Die 10 Erfolgreichsten Methoden! (2017, 7. Juli). mystipendium. Zugriff 10. Oktober 2025 unter https://www.mystipendium.de/studium/lerntechniken

Lopez, M. (2025, 24. Februar). *12 Tips To Avoid Academic Burnout*. Rochester Institute of Technology. Zugriff 10. Oktober 2025 unter https://www.rit.edu/admissions/blog/12-tips-avoid-academic-burnout

Ludwig, M., & Hartmeier, G. (2019). *Forschen, aber wie? (1. Auflage)*. hep Verlag.

Muneeb. (2025, 12. Juni). *Flask Project Structure: Best Practices with Blueprints & Application Factory Pattern*. Zugriff 31. Oktober 2025 unter https://muneebdev.com/flask-project-structure-best-practices/

PostgreSQL vs SQLite: The Ultimate Database Showdown. (2025, 17. Februar). Astera. Zugriff 31. Oktober 2025 unter https://www.astera.com/knowledge-center/postgresql-vs-sqlite/

Rainbow Table Attacks: How They Work and How to Defend Against Them. (o. D.). Netwrix. Zugriff 4. November 2025 unter https://netwrix.com/en/cybersecurity-glossary/cyber-security-attacks/rainbow-table-attack/

Ritschel-Gotal, A. D. (2023, 24. Mai). *Die Top 5 Lernmethoden & Top 15 Lerntechniken für berufliche Weiterbildungen*. evrlearn. Zugriff 10. Oktober 2025 unter https://www.evrlearn.ch/blog/2023/05/24/die-top-5-lernmethoden-top-15-lerntechniken-fur-berufliche-weiterbildungen/

Schmitz, C. (2017, 25. August). *The psychology of colours*. Limesurvey. Zugriff 10. Oktober 2025 unter https://www.limesurvey.org/blog/knowledge/colour-psychology-in-survey-design

Shaji, A. (2022, 14. Oktober). *CSRF Protection in Flask*. Zugriff 21. Oktober 2025 unter https://testdriven.io/blog/csrf-flask/

Srivastav, C. (2022, 5. August). *Flask Framework: WSGI Explained*. Medium. Zugriff 31. Oktober 2025 unter https://chaitanya-srivastav.medium.com/flask-framework-wsgi-explained-669753ca2b72

Template Designer Documentation — Jinja Documentation. (o. D.). The Pallets Projects. Zugriff 31. Oktober 2025 unter https://jinja.palletsprojects.com/en/stable/templates/

Templates — Flask Documentation. (o. D.). The Pallets Projects. Zugriff 31. Oktober 2025 unter https://flask.palletsprojects.com/en/stable/templating/

Tomazic, N. (2022, 8. November). *Heroku Alternatives for Python-based Applications*. Zugriff 18. Mai 2025 unter https://testdriven.io/blog/heroku-alternatives/

Weber, B.

bibinitperiod R. (2023, 17. September). *Lernmethoden und Lerntechniken*. Bernd Weber Team. Zugriff 10. Oktober 2025 unter https://www.diebegabungsspezialisten.de/2023/09/17/lernmethoden-und-lerntechniken/

What is HTTPS? (o. D.). Cloudflare. Zugriff 31. Oktober 2025 unter https://www.cloudflare.com/learning/ssl/what-is-https/

What is Password Hashing? (2022, 27. Juli). Stytch. Zugriff 10. April 2025 unter https://stytch.com/blog/what-is-password-hashing/

Abbildungsverzeichnis

Anhang

- **Code:** Der vollständige Code der Kanti Koala Web-App ist auf GitHub verfügbar unter: https://github.com/CoderAryanAnand/lernapp.
- **KI-Nachweis**
- **Tests**
- **Umfragen und Interviews**