

Prototyping Projektdokumentation

Name: Fabian Lucas Liechty

E-Mail: liechfa1@students.zhaw.ch

URL der deployten Anwendung: <https://jazzy-frangollo-7dc7f6.netlify.app/>

Github Repository: <https://github.com/macbaileys/TrailAdventurer>

Einleitung

Die Idee zu Trail Adventurer entstand aus meiner Leidenschaft für die Natur, Outdoor-Dinge und das Wandern. Ich wollte eine App machen, die dem Nutzer erlauben, Wanderwege der Schweiz auf spielerische Weise zu entdecken. Zudem mag ich es, Aktivitäten zu tracken und daraus ein motivierendes Spiel zu machen.

Hauptfunktionen

Die WebApp ermöglicht es Nutzern, insgesamt 52 verschiedene Wanderungen in der Schweiz zu entdecken und abzuschliessen. Der Nutzer hat die Möglichkeit, neue Wanderungen einzulesen und den Fortschritt bei den abgeschlossenen Herausforderungen zu verfolgen.

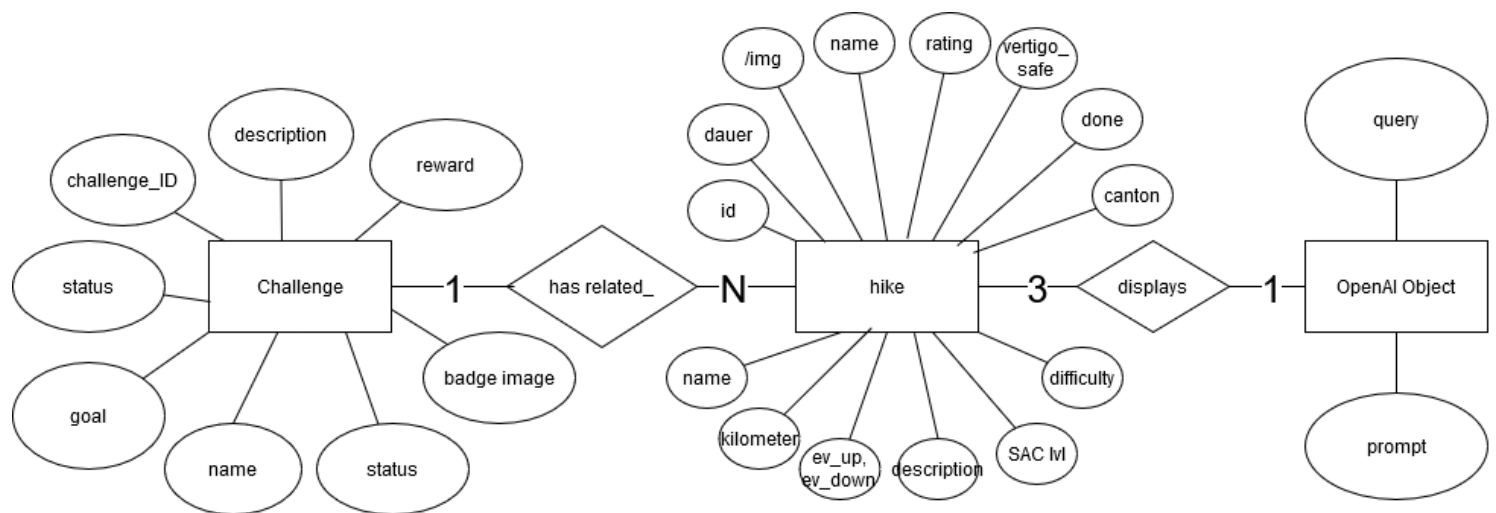
Gamifizierung und Herausforderungen

Die Anwendung enthält 10 verschiedene Badges, die durch das Abschliessen bestimmter Aufgaben verdient werden können. Diese reichen vom einfachsten Badge, den man erhält, wenn man nur eine Wanderung abgeschlossen hat, bis hin zum "Mutigen Wanderer", für den man eine Wanderung mit SAC-Level 6 meistern muss, und dem "Wanderhelden", der 20 der in der Datenbank vorhandenen Wanderungen abgeschlossen haben muss. Diese Herausforderungen motivieren die Nutzer, die verschiedenen Wanderwege zu erkunden und ihren Fortschritt sichtbar zu machen.

Um die Wanderungen visuell ansprechend zu gestalten, habe ich DALL-E's Bildgenerationsfunktion benutzt. Mit individuellen Prompts wurden schöne einheitliche Bilder zu den jeweiligen Wanderungen generiert.

KI-Feature mit GPT-4

Im Modul "Data Management" bei Alexandre de Spindler hatten wir in der letzten Lektion einen Exkurs, wie man Chat GPT in Anwendungen einbindet. Ich wollte mich mit dem genauer auseinandersetzen und habe die Gelegenheit genutzt dies in ein eigenes Projekt einbinden und habe einen OpenAI API Key gekauft und in meine Web-App integriert. Dies ermöglicht es den Nutzern, mithilfe von AI-Prompts nach passenden Wanderungen zu suchen. Die KI versteht die Eingaben der Nutzer und schlägt entsprechend der Datenbank passende Wanderungen vor.

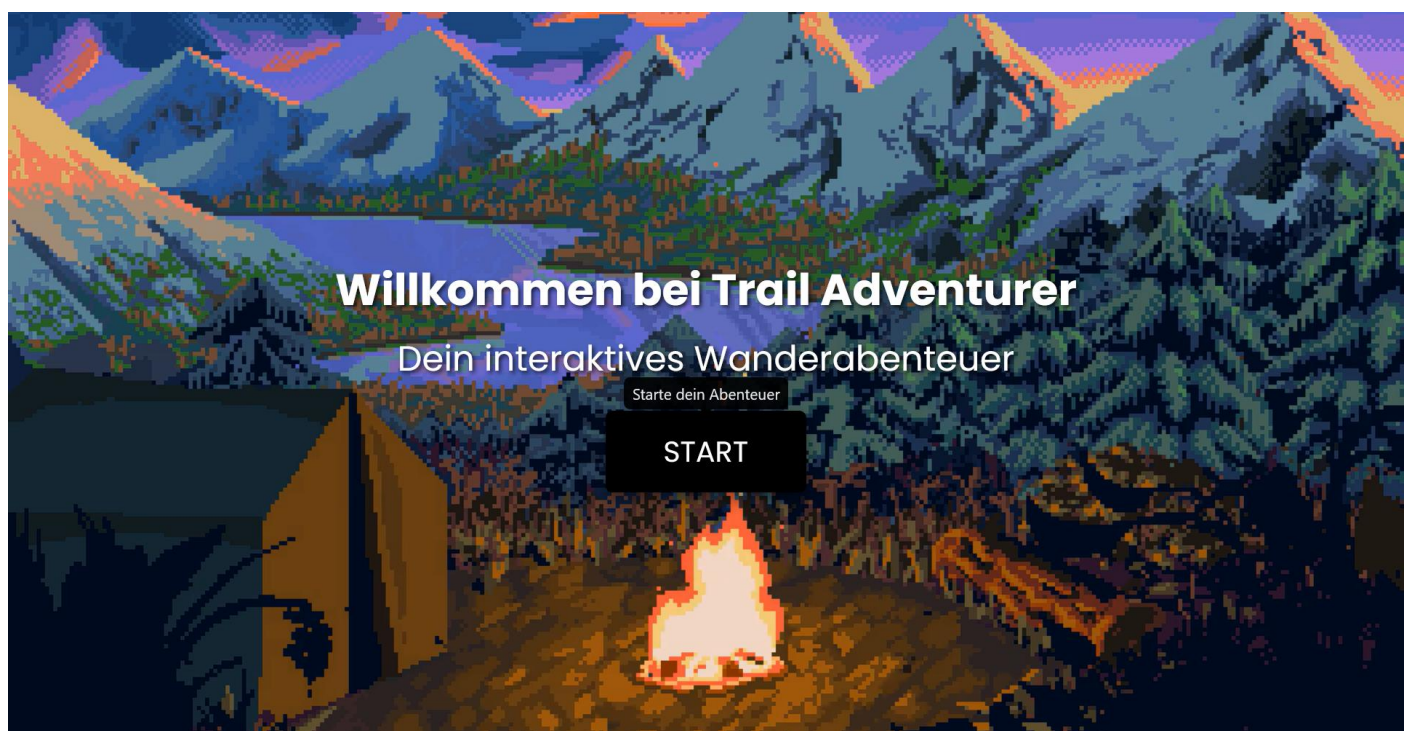


Das Objekt Challenge hat theoretisch N related Hikes die Benötigt werden um eine challenge/Herausforderung zu bestehen und den Bool "status" zu switchen. Ein Hike Objekt hat eine Vielzahl an Attributen, die fast alle konfiguriert werden können beim Erstellen eines neuen Hikes bzw einer neuen Wanderung. Das Open AI Objekt erhält eine query von dem Benutzer, die es zu einem prompt umändert, um die 3 akkuratesten Wanderungen anzuzeigen. Das Model hat eine Auswahl aus 52 verschiedenen Wanderungen/hikes in der HikeDB.

Beschreibung der Anwendung und das Grundgerüst

Auf der Startseite von Trail Adventurer wird der Nutzer von einer atmosphärischen Pixelgrafik empfangen, die eine malerische Berglandschaft zeigt. Der Schriftzug „Willkommen bei Trail Adventurer, dein interaktives Wanderabenteuer“ begrüsst den User herzlich und vermittelt sofort die spielerische Atmosphäre. Im Hintergrund lodert ein animiertes Lagerfeuer, was die Abenteuerstimmung verstärkt.

Ein grosser „Start“-Button lädt dazu ein, die Reise zu beginnen, und sobald man mit der Maus darüberfährt, erscheint eine Bootstrap-Animation mit dem Text „Starte dein Abenteuer“, die den User motiviert, das Abenteuer zu beginnen. src/routes/+page.svelte und src/routes/styles.css



Dieser einladende und interaktive Einstieg sorgt dafür, dass sich der Nutzer sofort in das Abenteuer hineinversetzt fühlt.

Dateien: `src\routes\+page.svelte` und `src\routes\styles.css`

Nachdem du auf den „Starte dein Abenteuer“-Button geklickt hast, erscheint die Navigationsleiste am oberen Bildschirmrand. Links sieht man ein animiertes GIF mit einem Lagerfeuer, das zum Thema passt, und daneben den Schriftzug „Trail Adventurer“.

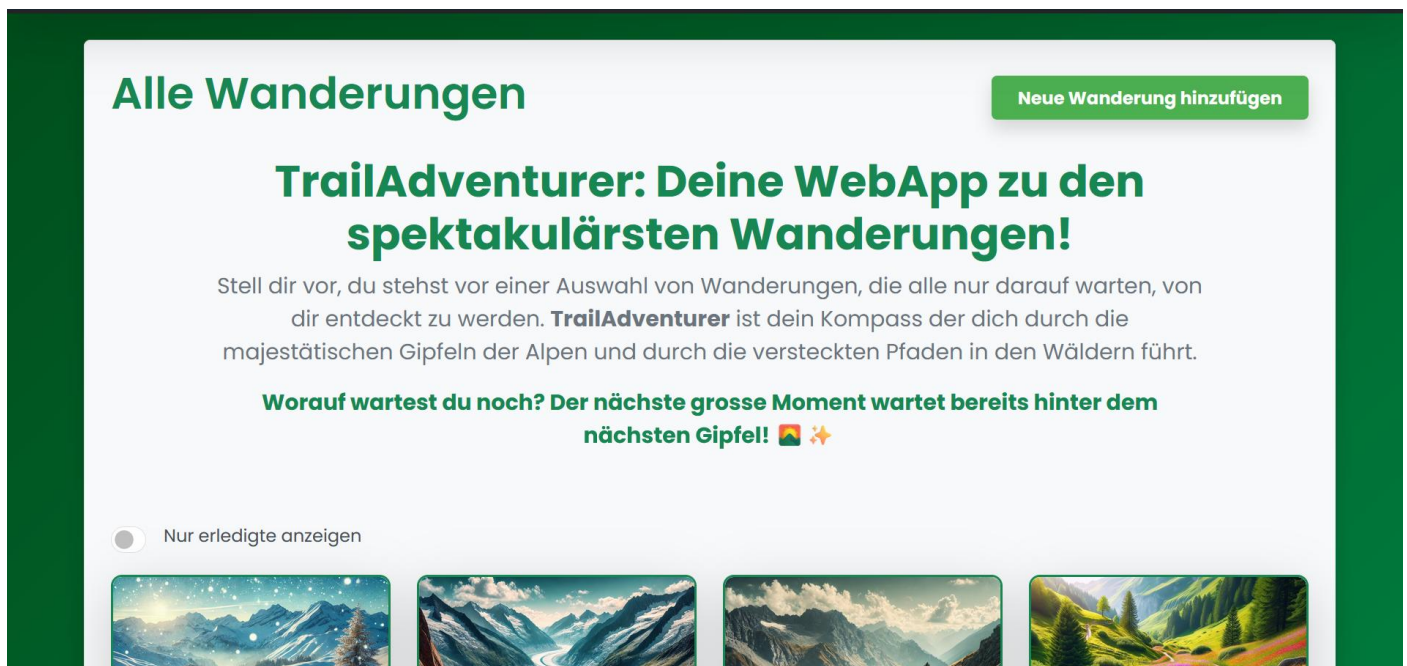
Dateien: `src\lib\components\Navbar.svelte` und `src\routes\styles.css`

In der Navigationsleiste gibt es drei Hauptbuttons: „Alle Wanderungen“, die eine Übersicht aller verfügbaren Wanderrouen bietet, „Herausforderungen“, die zu einem späteren Zeitpunkt erklärt werden, und einen speziell hervorgehobenen grünen Button „Wanderung mit AI finden“, der ein besonderes Feature andeutet.



Diese Leiste strukturiert die Navigation klar und lenkt die Aufmerksamkeit auf die wichtigsten Funktionen der Website.

Auf der Seite „Alle Wanderungen“ erscheint das Layout, das an ein Blatt Papier auf einem grünen Hintergrund erinnert. Ein einladender Text begrüsst den Nutzer und erklärt die Idee hinter Trail Adventure: eine Web-App, die zu spektakulären Wanderungen führt. Daneben befindet sich ein Button „Neue Wanderung hinzufügen“, den wir später besprechen werden.



Beim Scrollen erscheinen die ersten vier „Hike Cards“. Diese Karten zeigen jeweils ein Thumbnail-Bild, das mit spezifischen DALL-E-Prompts generiert wurde. Beim Überfahren mit der Maus erscheint der Titel als Overlay, und die Karte zoomt leicht heraus, was einen modernen, fast greifbaren Effekt erzeugt. Jede „Hike Card“ enthält den Namen der Wanderung und Attribute wie Kanton, Kilometeranzahl und Dauer.

Dateien: `src\routes\hikes\+page.svelte`, `src\routes\hikes\+page.server.js`, `src\lib\db.js`, `src\lib\components\HikeCard.svelte`, `src\routes\styles.css`

☐ Nur erledigte anzeigen



Schneekristallpfad

Kanton: Schwyz

Kilometer: 15 km

Aufstieg: 700 m,
Abstieg: 500 m

Dauer: 120 Minuten

Schwierigkeit: 5



Gletscherblickroute

Kanton: Wallis

Kilometer: 14 km

Aufstieg: 800 m,
Abstieg: 700 m

Dauer: 135 Minuten

Schwierigkeit: 6

Nutzerbewertung:



Felsenpfad

Kanton:

Graubünden

Kilometer: 12 km

Aufstieg: 600 m,
Abstieg: 600 m

Dauer: 110 Minuten

Schwierigkeit: 5



Wildblumenwanderung

Kanton: Bern

Kilometer: 13 km

Aufstieg: 900 m,
Abstieg: 850 m

Dauer: 150 Minuten

Schwierigkeit: 6

Es gibt die Möglichkeit, eine Wanderung als „erledigt“ zu markieren, wodurch der Status in der Datenbank geändert wird. Standardmässig werden nur die ersten vier Einträge angezeigt, um den Nutzer nicht zu überwältigen. Mit einem Klick auf „Alle anzeigen“ werden alle 52 verfügbaren Wanderungen sichtbar. Jede Wanderung kann individuell als „erledigt“ oder „zu tun“ markiert werden, wobei die Buttons entsprechend ihre Farbe ändern.

Nutzerbewertung:

★★★★★ (4.2)

Höhenangst: erlaubt

Als erledigt
markieren

Nutzerbewertung:

★★★★★ (4.5)

Höhenangst: nicht
erlaubt

Als zu Tun
markieren

Nutzerbewertung:

★★★★★ (4.1)

Höhenangst: erlaubt

Als zu Tun
markieren

Nutzerbewertung:

★★★★★ (4.3)

Höhenangst: nicht
erlaubt

Als erledigt
markieren

Alle anzeigen



Tessiner Schluchtenweg

Kanton: Tessin

Kilometer: 15 km

Aufstieg: 700 m,
Abstieg: 600 m



Obwaldner Höhenweg

Kanton: Obwalden

Kilometer: 18 km

Aufstieg: 850 m,
Abstieg: 750 m



Walliser Hängebrückentour

Kanton: Wallis

Kilometer: 10 km

Aufstieg: 400 m,
Abstieg: 350 m



Uri Klippenpfad

Kanton: Uri

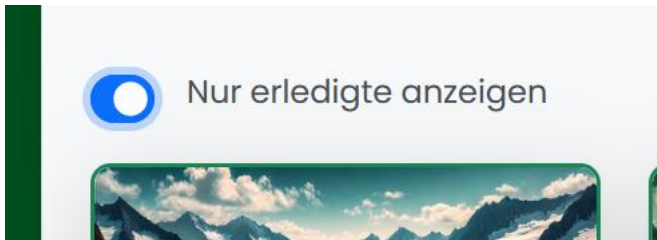
Kilometer: 13 km

Aufstieg: 650 m,
Abstieg: 600 m

Diese Seite bildet den zentralen Ausgangspunkt für die Erkundung aller Wanderungen in der App.


Dateien: `src\routes\hikes\+page.svelte`, `src\routes\hikes\+page.server.js`, `src\lib\db.js`,
`src\lib\components\HikeCard.svelte`, `src\routes\styles.css`

Es gibt ausserdem einen Schalter „Nur erledigte anzeigen“, mit dem man gezielt alle bereits abgeschlossenen Wanderungen filtern kann.



Wenn du eine „Hike Card“ auswählst, wie zum Beispiel den „Schneekristallpfad“, öffnet sich eine detaillierte Seite mit einer ansprechenden, animierten Darstellung. Hier siehst du den grossen Titel der Wanderung, ein beeindruckendes Bild und viele weitere Attribute wie das SAC-Level, das die Schwierigkeit auf einer Skala von T1 bis T6 angibt.

Schneekristallpfad



Details

Distanz: 15 km
Aufstieg: 700 m
Abstieg: 500 m
SAC-Level: T3
Unser Schwierigkeitsrating: 5/10
Durchschnittliches Rating: ★★★★★
Dauer: 120 Minuten
Höhenangst: erlaubt
Kanton: Schwyz
Bereits gewandert?
Als erledigt markieren

Zusätzlich findest du eine ausführliche Beschreibung der Wanderung. Du kannst den Status der Wanderung hier ebenfalls ändern – von „bereits gewandert“ zu „noch zu tun“ und umgekehrt. Eine besondere Funktion ist die Möglichkeit, die Wanderung direkt aus der Datenbank zu entfernen, was den Eintrag vollständig löscht.



Wandernangst: erlaubt

Kanton: Schwyz

Bereits gewandert?

Als erledigt markieren

Beschreibung

Ein leichter Wanderweg durch eine glitzernde Winterlandschaft im Kanton Schwyz. Ideal für Familien und Einsteiger.

Aus Datenbank entfernen

Diese Detailseite bietet eine umfassende, interaktive Übersicht über die jeweilige Wanderung.

Dateien: `src\routes\hikes\[hike_id]\+page.svelte`, `src\routes\hikes\[hike_id]\+page.server.js`, `src\lib\db.js`, `src\routes\styles.css`

Wenn du auf den Button „Neue Wanderung hinzufügen“ klickst, öffnet sich ein Formular, in dem du eine komplett neue Wanderung erstellen kannst. Hier kannst du alle wichtigen Details wie den Namen, die Beschreibung und den Kanton eingeben.

Neue Wanderung hinzufügen

← Zurück

Wanderung hinzufügen

Name

Name der Wanderung

Beschreibung

Kurze Beschreibung der Wanderung

Kanton

z.B. Zürich

Kilometer

z.B. 12.5

Höhenmeter Aufstieg

z.B. 500

Höhenmeter Abstieg

z.B. 450

Exponiertheitslevel

Ein besonderes Feature ist die Möglichkeit, ein Titelbild auszuwählen und hochzuladen, das dann in die MongoDB gespeichert wird. Wenn kein Bild ausgewählt wird, verwendet die App ein Platzhalterbild aus der Datenbank.

Dauer

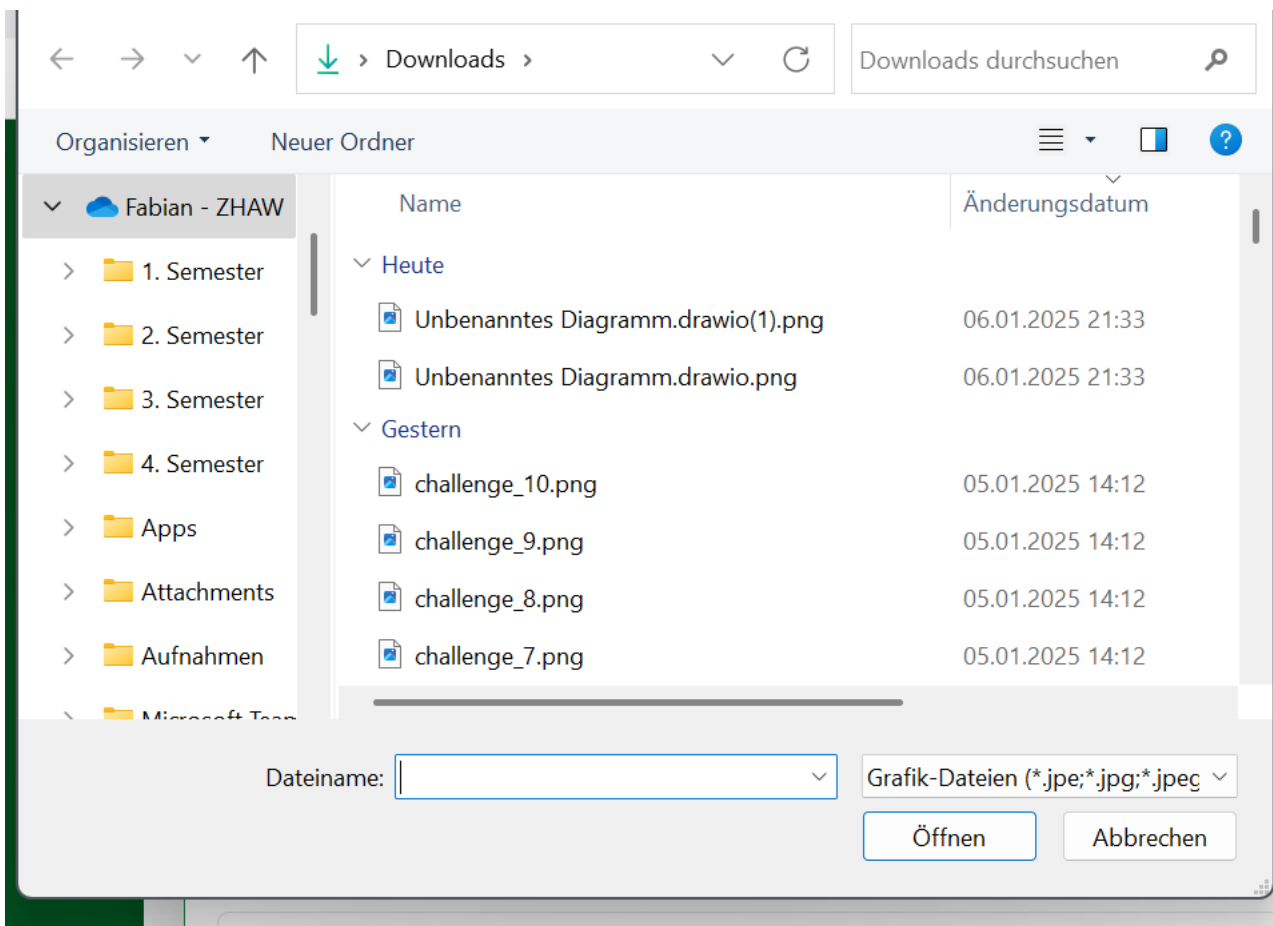
z.B. 300 Minuten

Titelbild

Durchsuchen...

Keine Datei ausgewählt.

Wanderung hinz



Sobald du auf „Wanderung hinzufügen“ klickst, wird die neue Wanderung in die Datenbank eingefügt und erscheint auf der Übersichtsseite.

Wanderung hinzufügen

Diese Funktion ermöglicht es, die App individuell zu erweitern und persönliche Wanderungen hinzuzufügen.

Dateien: `src\routes\hikes\create\+page.svelte`, `src\routes\hikes\create\+page.server.js`, `src\lib\db.js`, `src\routes\styles.css`

Erweiterungen: Challenges/Herausforderungen

Navigieren wir auf der Navbar auf Herausforderungen:



Auf der Seite „Herausforderungen“ kannst du auf den Button „Challenges aktualisieren“ klicken, um zu prüfen, ob du neue Erfolge oder Herausforderungen abgeschlossen hast. Dieser Schritt überprüft in der Datenbank, ob du die notwendigen Wanderungen für bestimmte Abzeichen bereits absolviert hast.



Nachdem die Überprüfung abgeschlossen ist, werden die „Challenge Cards“ angezeigt. Es gibt zehn verschiedene Herausforderungen, die jeweils als Karte dargestellt werden. Jede „Challenge Card“ zeigt deinen Fortschritt mit einer animierten Fortschrittsanzeige. Ist die Fortschrittsleiste vollständig, erhältst du ein digitales Abzeichen mit einem ansprechenden Bild. Auch hier ist wieder mit Bootstrap ein zeitgemässes UI eingerichtet.



Wenn eine Herausforderung noch nicht abgeschlossen ist, wird dies durch eine gelbe, animierte «ladende» Fortschrittsanzeige dargestellt. Du kannst jederzeit auf „Challenges aktualisieren“ klicken, um zu sehen, ob du für weitere Abzeichen berechtigt bist.

Jede Herausforderung ist mit der bestimmten Anzahl an Wanderungen verbunden, und die App überprüft, ob diese Wanderungen abgeschlossen wurden, um den Status der Challenges zu aktualisieren.

```
_id: "challenge_4"
name: "Flachland-Entdecker"
description: "Schliesse 3 Wanderungen in einem Nicht-Berg Kanton ab."
goal: 3
related_hikes: Array (12)
  0: "6630e72c95e12055f661ff12"
  1: "6630e72c95e12055f661ff31"
  2: "6630e72c95e12055f661ff32"
  3: "6630e72c95e12055f661ff33"
  4: "6630e72c95e12055f661ff34"
  5: "6630e72c95e12055f661ff35"
  6: "6630e72c95e12055f661ff36"
  7: "6630e72c95e12055f661ff37"
  8: "6630e72c95e12055f661ff38"
  9: "6630e72c95e12055f661ff39"
  10: "6630e72c95e12055f661ff40"
  11: "6630e72c95e12055f661ff52"
badge_image: "/images/challenge_4.png"
reward: "Badge 'Flachland-Entdecker'"
progress: 2
status: "in_progress"
```

Dateien: *src/routes/challenges/+page.svelte*, *src/routes/challenges/+page.server.js*, *src/routes/challenges/+server.js*, *src/lib/db.js*, *src/routes/styles.css*, *src/routes/api/updateChallenges/+server.js*

Der Code implementiert ein System zur Verwaltung und Aktualisierung von Herausforderungen. Das Zusammenspiel zwischen den 4 Wichtigen Dateien funktioniert wie folgt:

1. Frontend: Hauptseite (Svelte)

- Die Seite zeigt Challenges als Karten (ChallengeCard) an, die aus der Datenbank geladen werden.
- Ein „Aktualisieren“-Button (updateChallenges) sendet eine POST-Anfrage an die API, um den Fortschritt der Challenges zu aktualisieren.
- Nach der Aktualisierung wird die Seite neu geladen, um die aktualisierten Daten anzuzeigen.

2. +server.js

- **GET-Route:** Ruft alle Challenges aus der MongoDB-Datenbank ab und liefert diese an das Frontend.
- **PATCH-Route:** Aktualisiert den Fortschritt einer spezifischen Challenge, wenn eine dazugehörige Wanderung abgeschlossen wird.

3. +page.server.js

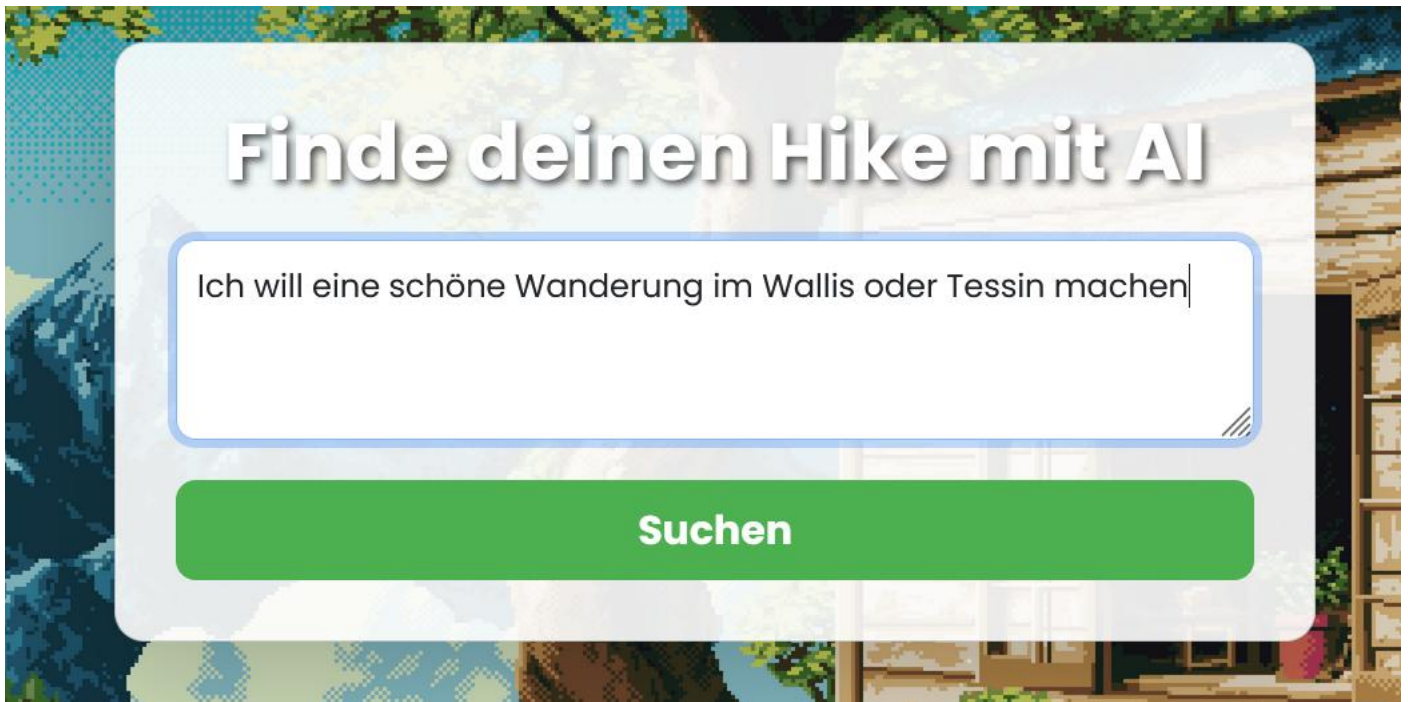
- Lädt die Challenges beim ersten Seitenaufruf aus der Datenbank und stellt sie dem Frontend zur Verfügung.

4. api/updateChallenges

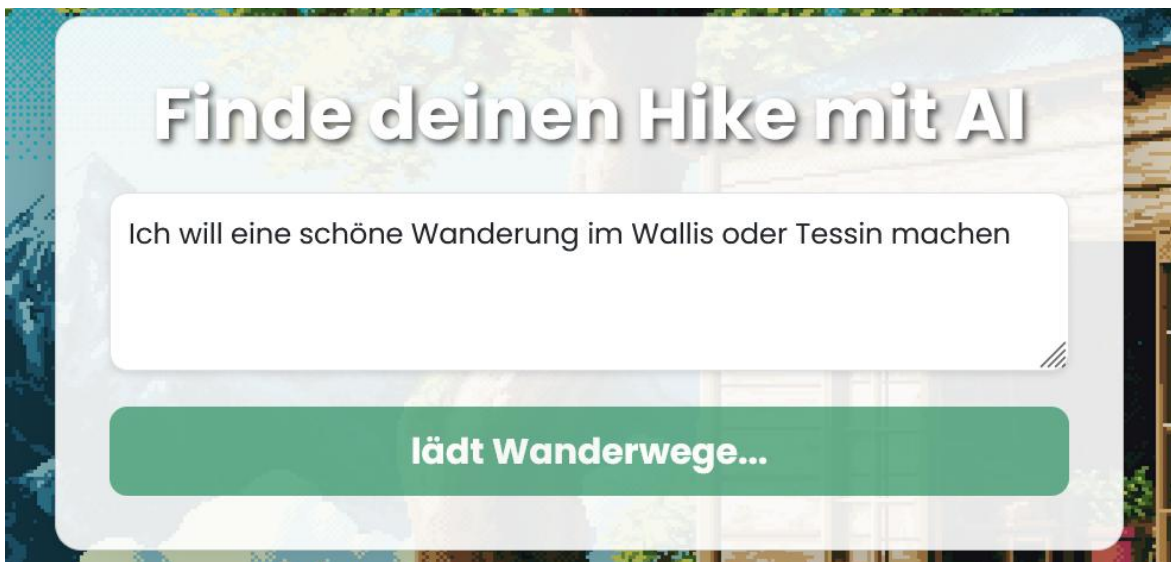
- Diese API prüft, welche „favorisierten Wanderungen“ (aus einer anderen Sammlung) mit den Challenges verknüpft sind.
- Es zählt die Übereinstimmungen und aktualisiert den Fortschritt sowie den Status der Challenges (in_progress oder achieved).

Erweiterungen: «Find your Hike» mit OpenAI API Backend.

Ein weiteres besonderes Feature der App ist das „Finde deinen Hike mit AI“-Menu. Basierend auf einem API-Key von OpenAI, den man in eigene Backends integrieren kann, inspiriert von einem Exkurs in Data Management bei Alexandre De Spindler, ermöglicht dieses Menü die Suche nach Wanderungen durch einfache Eingabe eines Prompts.



In einem Eingabefeld kannst du einen Wunsch, wie „Ich möchte eine Wanderung im Wallis machen“, eingeben. Nach einem Klick auf „Suchen“ verwandelt die App deinen Prompt in eine Suchanfrage...



und schlägt dir bis zu drei passende Wanderungen vor. Diese können wie gewohnt als „erledigt“ markiert oder im Detail betrachtet werden.



Dieses Feature erweitert die App um eine interaktive, personalisierte Komponente, die auf künstlicher Intelligenz basiert.

Dateien: `src\routes\openai_integration\+server.js`, `src\routes\openai_integration\+page.server.js`,
`src\routes\openai_integration\+page.svelte`, `src\routes\styles.css`

Funktionsweise des Codes:

1. Benutzerinteraktion auf der Oberfläche

Der Nutzer öffnet die Website und sieht eine zentrale Eingabemaske mit dem Titel „Finde deinen Hike mit AI“. In ein Textfeld kann der Nutzer eine Beschreibung der gewünschten Wanderung eingeben, z. B. „eine leichte Wanderung im Wallis für Familien“. Sobald der Nutzer auf den Button „Suchen“ klickt, wird die Funktion `getHike` aufgerufen.

Während der Button gedrückt wird, ändert sich der Zustand: Ein Ladeindikator signalisiert, dass die Suche läuft. Gleichzeitig wird das Textfeld geleert und der Zustand zurückgesetzt, um neue Daten einzufügen, sobald sie verfügbar sind.

2. Verarbeitung der Anfrage im Backend

Die Anfrage wird über die Funktion `getHike` an einen Endpunkt `/openai_integration` gesendet. Dort übernimmt der Server die Eingabe und nutzt die OpenAI GPT-4 API, um relevante Schlüsselwörter aus der Beschreibung des Nutzers zu extrahieren. Zum Beispiel könnten aus der Eingabe „eine leichte Wanderung im Wallis für Familien“ die Schlüsselwörter „leicht, Wallis, Familie“ gewonnen werden. Diese Schlüsselwörter werden ohne Präfix oder zusätzliche Beschreibungen als Liste zurückgegeben.

Eine eingebaute Retry-Logik stellt sicher, dass die Anfrage auch bei Überlastung der API (Rate-Limiting) bis zu drei Mal wiederholt wird, bevor ein Fehler ausgegeben wird.

3. Suche in der MongoDB-Datenbank

Mit den gewonnenen Schlüsselwörtern wird eine Anfrage an die MongoDB-Datenbank gestellt. Hier wird versucht, Wanderungen zu finden, die den Schlüsselwörtern entsprechen. Die Datenbank durchsucht dabei verschiedene Felder wie `wanderung`, `description` oder `canton`, um Übereinstimmungen zu finden. Wenn keine direkten Treffer erzielt werden, wird eine breitere Suche durchgeführt, die mithilfe der Volltextsuche (`$text`) nach relevanten Wanderungen sucht. Sollte dies ebenfalls keinen Erfolg bringen, werden als Fallback die drei höchstbewerteten Wanderungen aus der gesamten Datenbank angezeigt.

Die gefundenen Ergebnisse werden so vorbereitet, dass die spezifischen Datenbank-IDs (ObjectId) in eine stringbasierte Form umgewandelt werden, die im Frontend genutzt werden kann.

4. Darstellung der Ergebnisse im Frontend

Die Suchergebnisse werden zurück an das Frontend geschickt, wo sie als HikeCard-Komponenten dargestellt werden. Jede Karte zeigt relevante Informationen zur Wanderung wie Name, Ort und Schwierigkeitsgrad. Die Ergebnisse erscheinen in einem Rasterformat, und der Nutzer kann die vorgeschlagenen Wanderungen durchsehen.

Wenn keine Wanderungen gefunden werden, zeigt die Seite eine Meldung wie „Keine Wanderungen gefunden. Bitte versuche es erneut!“.

Erweiterungen: Challenge Card

Die ChallengeCard-Komponente stellt eine Herausforderung strukturiert dar und bietet einige spezielle Features, um Fortschritte und Status übersichtlich zu visualisieren. Eine dynamische Fortschrittsanzeige zeigt den Fortschritt in Prozent an und passt sich mit einer Progress-Bar visuell an. Der Fortschrittsbalken ist farblich codiert: Grün für abgeschlossene Challenges mit einer zusätzlichen sparkle-effect-Klasse und gelb mit animierten Streifen für laufende. Status-Badges heben den aktuellen Stand hervor und enthalten über Tooltips zusätzliche Informationen, etwa eine Gratulation bei abgeschlossener Herausforderung. Die Karte zeigt außerdem eine Beschreibung der Herausforderung, die Belohnung und, falls verfügbar, ein rundes Badge-Bild im Footer, das bei Abschluss prominent dargestellt wird. Insgesamt kombiniert die ChallengeCard klare Visualisierungen, interaktive Elemente und Belohnungsdetails, um den Fortschritt einer Herausforderung präzise darzustellen.

Erster Schritt

Schliesse eine Wanderung ab.

Fortschritt:

100%

Status: Abgeschlossen

Belohnung: Badge 'Erster Schritt'



Verdienter Erfolg!

Walliser Entdecker

Schliesse 3 Wanderungen im Kanton Wallis ab.

Fortschritt:

33%

Status: In Bearbeitung

Belohnung: Badge 'Walliser Entdecker'

Erweiterungen: Filter nach Erledigten



Nur erledigte anzeigen

Der Code definiert eine abgeleitete (derived) reaktive Variable namens `hikes`, die basierend auf bestimmten Bedingungen dynamisch eine gefilterte Liste von Wanderungen (`hikes`) zurückgibt. Hier ist eine Erklärung der Logik:

1. Filteroption für Favoriten (`filterByFavorites`):

- Wenn die Variable `filterByFavorites` wahr ist, werden nur die Wanderungen aus der ursprünglichen Datenquelle (`data.hikes`) zurückgegeben, die als Favoriten markiert sind (`hike.favorite`).

2. Option zum Anzeigen aller Wanderungen (`showAllHikes`):

- Wenn `filterByFavorites` nicht aktiv ist und `showAllHikes` wahr ist, wird die gesamte Liste der Wanderungen (`data.hikes`) zurückgegeben.

3. Standardverhalten:

- Falls weder `filterByFavorites` noch `showAllHikes` aktiv sind, wird nur ein Teil der Wanderungen (die ersten vier Einträge aus der Liste) zurückgegeben, um die Anzeige einzuschränken.