Prototyping Projektdokumentation

Name: Gioia Finocchi

E-Mail: finocgio@students.zhaw.ch

URL der deployten Anwendung: https://sporteventsapp.netlify.app/

# Einleitung

**Sportevents App – Übersicht und Verwaltung von Sportveranstaltungen**

Die Sportevents App ist eine Webanwendung zur Verwaltung und Darstellung von Sportveranstaltungen. Sie wurde mit SvelteKit 5 entwickelt.

Die zentrale Idee der App ist es, einen einfachen Überblick über Sportevents zu ermöglichen – inklusive Teilnehmer, Rennen und Resultaten. Veranstalter können neue Rennen anlegen, Teilnehmer erfassen und individuelle Resultate verwalten.

Grundfunktionen:

* Startseite mit Karussell-Bildern zur Einstimmung
* Seiten zur Anzeige, Erstellung und Bearbeitung von:
  + Teilnehmern (mit Kategorien und Filtermöglichkeit)
  + Rennen (inkl. Datum, Ort, Distanz, Beschreibung)
  + Resultaten (automatische Gruppierung nach Rennen)
* Intelligente Such- und Filterfunktionen (Textsuche, Kategorieauswahl)
* Benutzerfreundliche Navigation und klare, strukturierte Darstellung
* Interaktive Komponenten wie z. B. ein Button zur schnellen Datenerfassung
* Persistenz via MongoDB und dynamische Seiten per SvelteKit Server-Load

Die App ist sowohl funktional als auch optisch klar strukturiert und mobilfreundlich gestaltet – ideal für kleinere Sportorganisationen oder Eventplaner.

# Datenmodell

*Ein Bild, das Zeichnung, Entwurf, Lineart, Diagramm enthält.

KI-generierte Inhalte können fehlerhaft sein.*

# Beschreibung der Anwendung

## Layout auf allen Seiten

**Datei:** routes/+layout.svelte

Diese Datei definiert das globale Layout der Anwendung. Es beinhaltet die Navigationsleiste oben, den gemeinsamen Seitenrahmen sowie die Fusszeile. Das {@render children()}-Statement dient als Platzhalter für die Inhalte aller Unterseiten und ermöglicht konsistentes Styling und Verhalten über die gesamte App hinweg.

**Funktionen:**

* Navigationsleiste mit Links zu allen Hauptseiten:
  + Home
  + Teilnehmer
  + Rennen
  + Resultate
* Mobile-responsive Menüführung (Burger-Menü bei kleiner Bildschirmbreite)
* Einbindung eines Logos bzw. Icons in der Navbar
* Platzhalter ({@render children()}) für den dynamischen Seiteninhalt
* Gemeinsame Fusszeile mit Kontakt-E-Mail und aktuellem Jahr

**Verwendete Klassen und Elemente:**

* Bootstrap-Klassen (navbar, container, btn, etc.)
* Eigenes CSS via style.css (z. B. .icon, .intro-section)
* Responsive Design durch Bootstrap Grid und Collapse-Mechanismus

**Besonderheit:**

* Die Datei bindet keine Styles direkt ein – CSS und Bootstrap werden zentral in app.html eingebunden

## Startseite – Übersicht und Einstieg

**Route:** /

Auf der Startseite wird der Besucher mit einem Karussell aus stimmungsvollen Bildern empfangen, die verschiedene Veranstaltungsorte repräsentieren (z. B. Lugano, Bern, Lausanne). Darunter folgen drei grosse Buttons, mit denen direkt zu den zentralen Bereichen der App navigiert werden kann: Teilnehmer, Rennen und Resultate.

Die Bilder rotieren automatisch (mittels Bootstrap-Karussell) und geben der Seite ein modernes Look-and-Feel. Die Buttons dienen als Schnellnavigation und sind visuell im Zentrum positioniert.

**Funktionen:**

* Bildkarussell mit 5 Veranstaltungsorten
* Einstiegstext zur Erklärung der App
* Drei Navigationselemente zu:
  + Teilnehmer verwalten
  + Rennen verwalten
  + Resultate anzeigen

**Dateien:**

* routes/+page.svelte
* routes/+page.server.js

**Screenshot:**  
*Ein Bild, das Text, Screenshot, Software, Website enthält.

KI-generierte Inhalte können fehlerhaft sein.*

## Teilnehmer

### Teilnehmerübersicht

**Route:** /participants

Auf dieser Seite werden alle erfassten Teilnehmer übersichtlich in einer Liste angezeigt. Jede Zeile zeigt den Namen und die Kategorie des Teilnehmers. Die Einträge sind klickbar und führen zur Detailansicht eines Teilnehmers.

Zusätzlich bietet die Seite eine intelligente Filterung: Teilnehmer können per Textsuche (z. B. nach Name oder E-Mail) gefiltert werden. Ausserdem stehen Kategorie-Buttons zur Verfügung, mit denen z. B. nur „Junior“- oder „Senior“-Teilnehmer angezeigt werden.

Über den Button **„➕ Neuen Teilnehmer erfassen“** gelangt man zur Seite /participants/create, wo ein neues Formular zur Verfügung steht.

**Funktionen:**

* Darstellung aller Teilnehmer in einer Liste
* **Textsuche** über Name, E-Mail etc.
* **Kategorie-Filter** über Buttons
* Weiterleitung zur Erstellungsseite per Button

**Dateien:**

* routes/participants/+page.svelte
* routes/participants/+page.server.js
* lib/components/createbutton.svelte
* lib/server/db.js

**Screenshot:**  
Ein Bild, das Text, Software, Screenshot, Webseite enthält.

KI-generierte Inhalte können fehlerhaft sein.

### Teilnehmer erfassen

**Route:** /participants/create

Auf dieser Seite kann ein neuer Teilnehmer erfasst werden. Das Formular enthält Felder für Name, E-Mail, Alter, Kategorie sowie optional ein Rennen und eine Zielzeit. Kategorien und Rennen werden dynamisch vom Server geladen und in den Auswahlfeldern angezeigt.

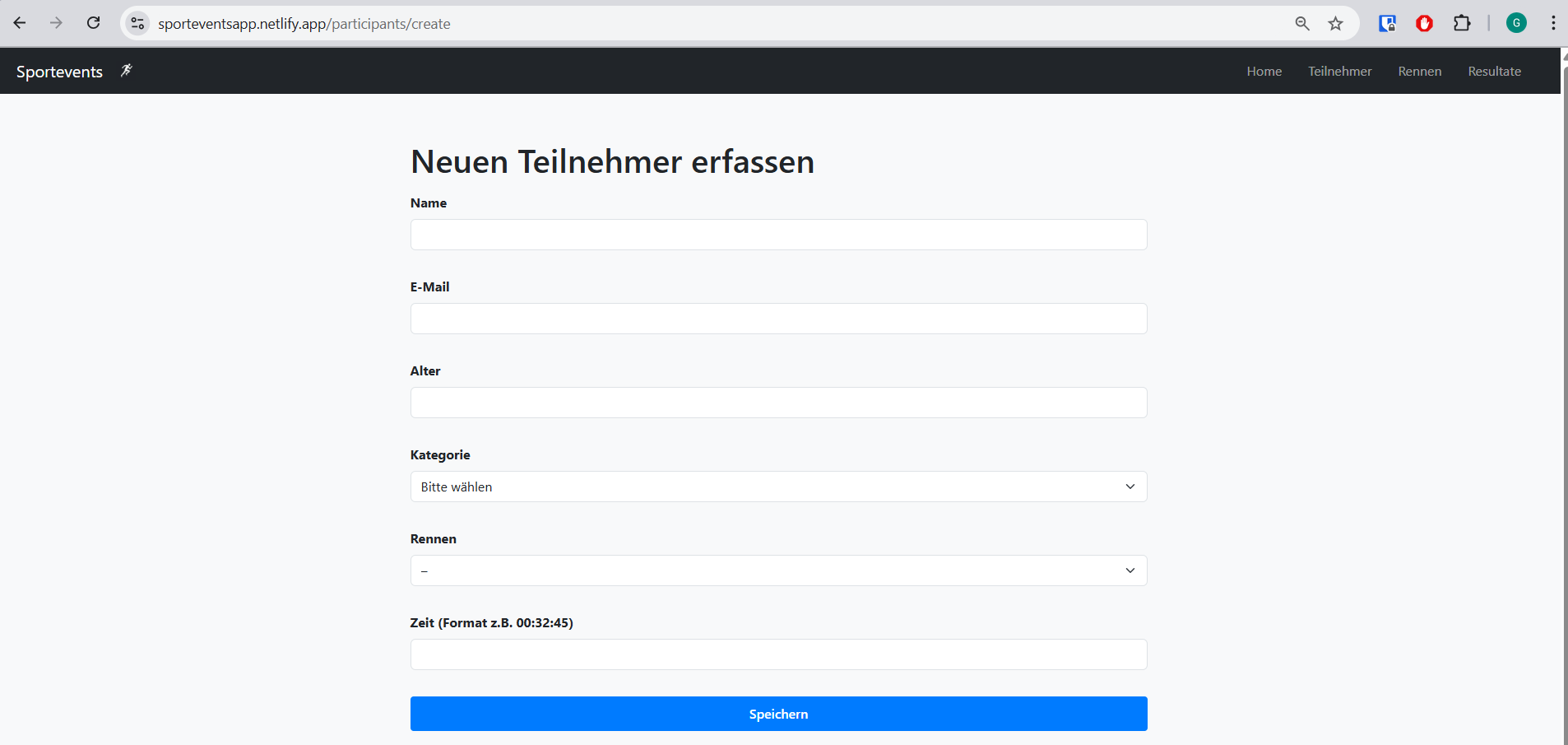
Die Seite ist vollständig validiert: Alle Pflichtfelder müssen ausgefüllt sein. Nach dem Absenden wird der Teilnehmer gespeichert. Wenn ein Rennen und eine Zeit angegeben sind, wird automatisch ein Ergebnis mit dem neuen Teilnehmer erstellt.

**Funktionen:**

* Eingabemaske für neue Teilnehmer
* Pflichtfelder: Name, E-Mail, Alter, Kategorie
* Optional: Rennen und Zielzeit (z. B. „00:32:45“)
* Dynamisches Dropdown für Kategorie und Rennen
* Weiterleitung zur Teilnehmerübersicht nach dem Speichern

**Dateien:**

* routes/participants/create/+page.svelte
* routes/participants/create/+page.server.js
* lib/server/db.js

**Screenshot:**  
**

### Teilnehmer Detailansicht

**Route:** /participants/[id]

Diese Seite zeigt die Detailinformationen eines bestimmten Teilnehmers. Angezeigt werden Name, E-Mail, Alter und Kategorie. Falls der Teilnehmer bereits an einem Rennen teilgenommen hat, wird zusätzlich das Rennen (Name, Ort, Datum) und die erreichte Zeit sowie die Platzierung angezeigt.

Am unteren Rand befinden sich zwei Buttons:

* **„Bearbeiten“**: führt zur Bearbeitungsseite /participants/[id]/edit.
* **„Löschen“**: löscht den Teilnehmer nach einer Bestätigung und leitet zurück zur Übersicht.

**Funktionen:**

* Anzeige aller relevanten Teilnehmerdaten
* Darstellung eines verknüpften Rennens inklusive Zeit & Platz (falls vorhanden)
* Möglichkeit zur Bearbeitung oder Löschung des Teilnehmers

**Dateien:**

* routes/participants/[id]/+page.svelte
* routes/participants/[id]/+page.server.js
* lib/server/db.js

*Ein Bild, das Text, Software, Webseite, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.***Screenshot:**

### Teilnehmer bearbeiten

**Route:** /participants/[id]/edit

Diese Seite ermöglicht das Bearbeiten eines bestehenden Teilnehmers. Die Eingabefelder sind bereits mit den aktuellen Daten des Teilnehmers vorausgefüllt. Änderungen können direkt vorgenommen und über den Button **„Änderungen speichern“** bestätigt werden.

Zusätzlich gibt es einen **„Abbrechen“**-Button, der ohne Änderungen zurück zur Detailansicht führt.

**Funktionen:**

* Vorbelegung der Felder mit bestehenden Teilnehmerdaten
* Validierung aller Pflichtfelder wie Name, E-Mail, Alter und Kategorie
* Speichern der Änderungen und Weiterleitung zur Detailansicht

**Dateien:**

* routes/participants/[id]/edit/+page.svelte
* routes/participants/[id]/edit/+page.server.js
* lib/server/db.js

Ein Bild, das Text, Software, Computersymbol, Webseite enthält.

KI-generierte Inhalte können fehlerhaft sein.**Screenshot:**

## Rennen

### Rennen Übersicht

**Route:** /races

Auf dieser Seite werden alle erfassten Rennen übersichtlich gelistet. Jedes Rennen zeigt Name, Ort, Datum und Distanz. Die Seite enthält zusätzliche Funktionen zur Filterung und Verwaltung:

* Oben befindet sich ein Button **„➕ Neues Rennen erfassen“**, der zur Renn-Erstellungsseite weiterleitet.
* Über ein Suchfeld können Rennen nach Name, Ort, Datum oder Distanz gefiltert werden.
* Die Filterung funktioniert sowohl für Texteingaben als auch für konkrete Distanzen (z. B. 10 findet 10 km-Rennen).
* Jeder Rennname ist verlinkt und führt zur Detailansicht des entsprechenden Rennens.

**Funktionen:**

* Übersicht aller Rennen mit Kerninformationen
* Dynamische Filterung nach mehreren Kriterien
* Navigation zur Detail- und Bearbeitungsansicht

**Dateien:**

* routes/races/+page.svelte
* routes/races/+page.server.js
* lib/components/createbutton.svelte (für den Erstellungs-Button)
* lib/server/db.js

**Screenshot:**  
Ein Bild, das Text, Software, Webseite, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.

### Rennen erfassen

**Route:** /races/create

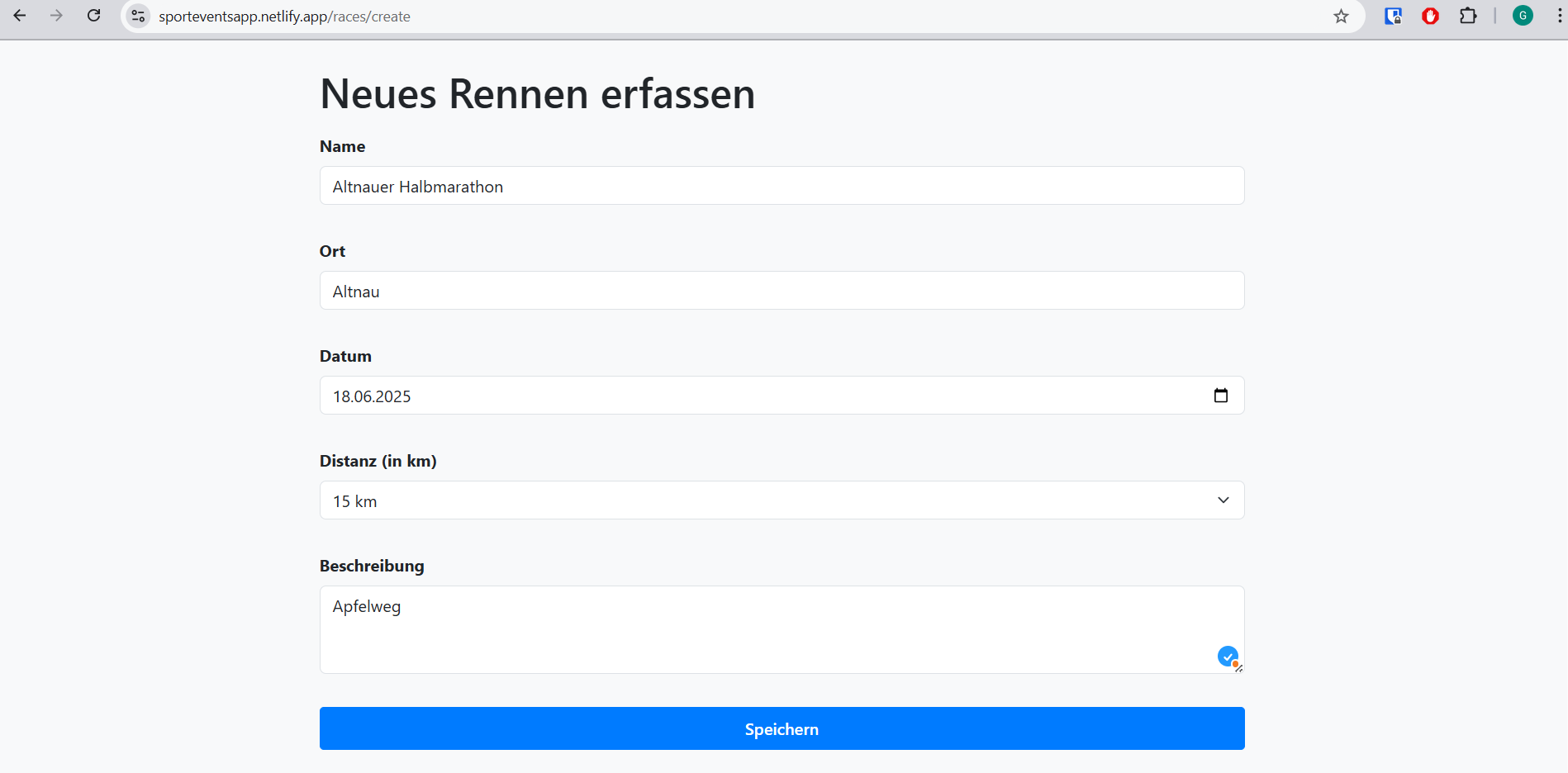
Auf dieser Seite können neue Rennen über ein Formular erfasst werden. Es stehen alle relevanten Felder zur Verfügung, darunter Name, Ort, Datum, Distanz und eine optionale Beschreibung.

**Funktionen:**

* Eingabe aller Renndaten über ein strukturiertes Formular
* Auswahl der Distanz aus einer Dropdownliste (inkl. Halbmarathon/Marathon)
* Validierung aller Pflichtfelder vor dem Speichern
* Nach dem Speichern wird direkt zur Detailansicht des neuen Rennens weitergeleitet

**Dateien:**

* routes/races/create/+page.svelte
* routes/races/create/+page.server.js
* lib/server/db.js

**Screenshot:**  


### Rennen Detailansicht

**Route:** /races/[id]

Diese Seite zeigt die vollständigen Informationen eines bestimmten Rennens:

* Ort, Datum, Distanz und Beschreibung werden in einer strukturierten Liste dargestellt.
* Über zwei Buttons kann das Rennen **bearbeitet** oder **gelöscht** werden:
  + *„Bearbeiten“* führt zur Bearbeitungsmaske
  + *„Löschen“* entfernt das Rennen nach Bestätigung
* Ein Link unten führt zurück zur Übersicht aller Rennen.

**Funktionen:**

* Anzeige aller Renndetails
* Navigation zur Bearbeitung
* Löschen eines Rennens per POST-Action

**Dateien:**

* routes/races/[id]/+page.svelte
* routes/races/[id]/+page.server.js
* lib/server/db.js

*Ein Bild, das Text, Software, Schrift, Webseite enthält.

KI-generierte Inhalte können fehlerhaft sein.***Screenshot:**

### Rennen bearbeiten

**Route:** /races/[id]/edit

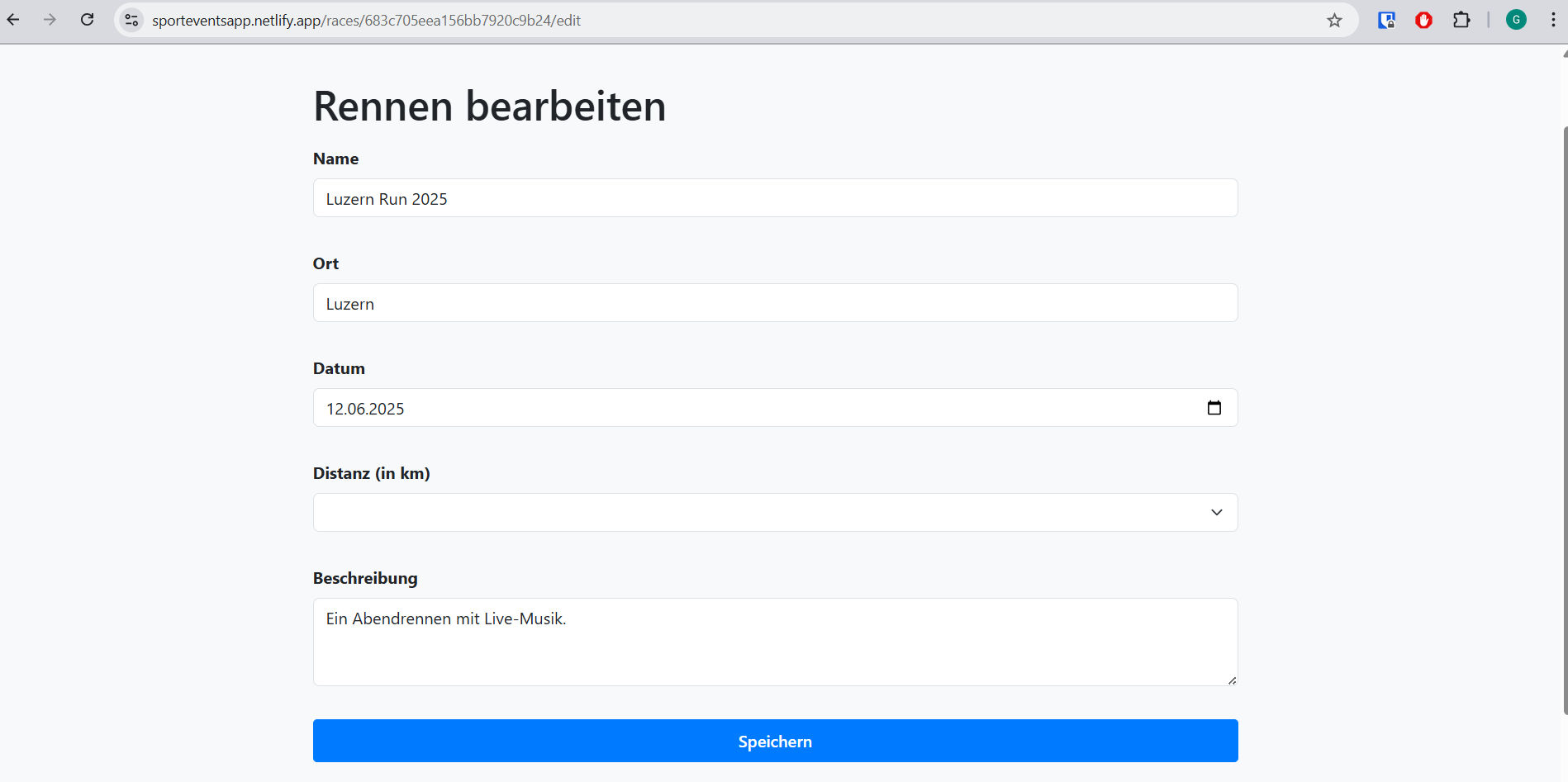
Auf dieser Seite kann ein bestehendes Rennen bearbeitet werden. Die Eingabefelder sind mit den aktuellen Daten des Rennens vorausgefüllt.

**Funktionen:**

* Bearbeiten aller Felder: Name, Ort, Datum, Distanz, Beschreibung
* Dropdown-Auswahl für die Distanz (z. B. 5 km, 10 km, Marathon)
* Speicherung der Änderungen per POST-Action
* Nach dem Speichern wird zur aktualisierten Detailansicht weitergeleitet
* Ein *«Abbrechen»*-Button führt zurück ohne Änderungen

**Dateien:**

* routes/races/[id]/edit/+page.svelte
* routes/races/[id]/edit/+page.server.js
* lib/server/db.js

****Screenshot:**

## Resultate

**Route:** /results

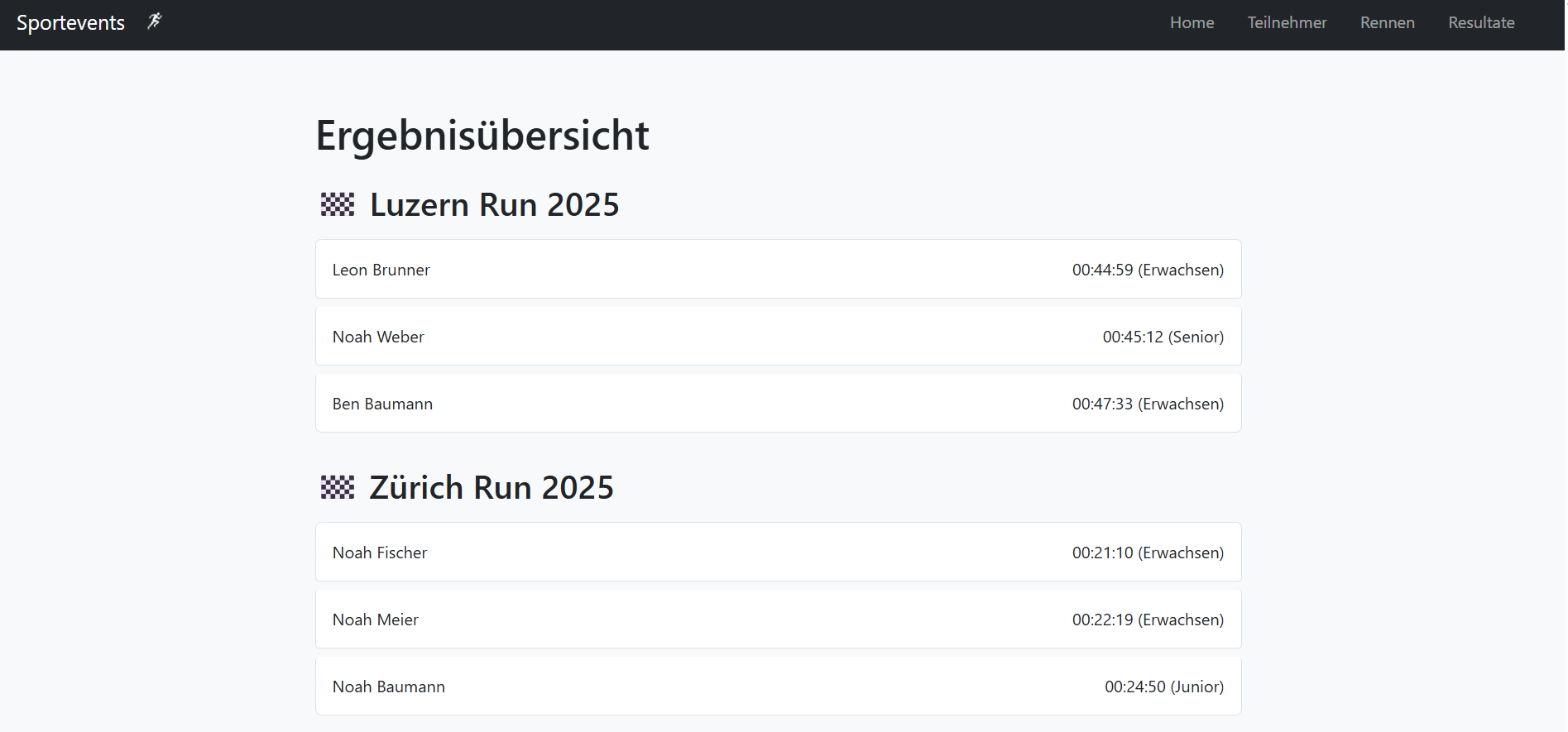
Diese Seite zeigt eine nach Rennen gruppierte Übersicht aller erfassten Resultate. Zu jedem Resultat werden Teilnehmername, Zeit und ggf. Kategorie angezeigt.

**Funktionen:**

* Gruppierung nach Rennenname (jeweils mit Überschrift)
* Sortierung der Resultate **innerhalb jedes Rennens nach Zeit**
* Anzeige von Teilnehmernamen, Zeiten und Kategorien
* Saubere Darstellung in nummerierten Listen (⟶ Rangfolge)

**Dateien:**

* routes/results/+page.svelte
* routes/results/+page.server.js
* lib/server/db.js

**Screenshot:**  
**

# Erweiterungen

Neben den im Aufgabenblatt definierten Grundfunktionen wurden in der Anwendung verschiedene funktionale und gestalterische Erweiterungen umgesetzt, die den Funktionsumfang sinnvoll ergänzen und die Benutzerfreundlichkeit erhöhen. Die folgenden Punkte dokumentieren diese zusätzlichen Features.

## Create Button Komponente

Ein wiederverwendbarer Button zur Navigation wurde als eigenständige Komponente ausgelagert. Dadurch kann z. B. der „➕ Neuen Teilnehmer erfassen“-Button flexibel in verschiedenen Seiten eingesetzt werden, ohne Redundanz im Code.

**Dateien:**

* lib/components/createbutton.svelte
* Verwendet in:
  + routes/participants/+page.svelte
  + routes/races/+page.svelte

4.2. Bildkarussell auf der Startseite

Ein automatisches Bootstrap-Karussell mit fünf wechselnden Bildern wurde in die Startseite integriert. Die Bilder stellen verschiedene Austragungsorte dar und sorgen für visuelles Interesse.

**Dateien:**

* routes/+page.svelte
* Bilder liegen im Verzeichnis /static/

4.3. Textbasierte Suche bei Teilnehmern

Teilnehmer können über ein Texteingabefeld gefiltert werden. Die Suche prüft alle Teilnehmerfelder (z. B. Name, E-Mail) dynamisch auf Übereinstimmungen.

**Dateien:**

* routes/participants/+page.svelte
* Funktion: filterParticipants(array, query, category)

4.4. Kategorie-Filter für Teilnehmer

Über Buttons können Teilnehmer nach Kategorie (Junior, Erwachsen, Senior) gefiltert werden. Der aktive Button ist visuell hervorgehoben, um die aktuelle Auswahl anzuzeigen.

**Dateien:**

* routes/participants/+page.svelte
* Variablen: selectedCategory, kategorien

4.5. Kombinierte Filterung bei Rennen

Rennen können sowohl über ein Textfeld (Name, Ort, Datum, Beschreibung) als auch über eine konkrete Kilometerzahl (z. B. „10“) gefiltert werden. Die Eingabe wird dynamisch analysiert, um Text- oder Zahlenfilter anzuwenden.

**Dateien:**

* routes/races/+page.svelte
* Funktion: filterRaces(array, query)

4.6. Automatische Ergebnis-Zuweisung beim Erstellen eines Teilnehmers

Wenn beim Erfassen eines neuen Teilnehmers gleichzeitig ein Rennen und eine Zielzeit angegeben werden, wird automatisch ein Resultat für diesen Teilnehmer erstellt.

**Dateien:**

* routes/participants/create/+page.server.js
* Logik im actions.default-Handler

4.7. Ergebnisanzeige in Teilnehmer-Detailseite

In der Teilnehmer-Detailansicht wird das zugehörige Rennen samt Zielzeit und Rang angezeigt, sofern der Teilnehmer ein Resultat hat.

**Dateien:**

* routes/participants/[id]/+page.svelte
* Daten geladen via: routes/participants/[id]/+page.server.js

4.8. Gruppierung und Sortierung von Resultaten

Die Resultate-Seite gruppiert alle Resultate nach Rennen. Innerhalb jedes Rennens werden die Teilnehmer nach ihrer Zeit aufsteigend sortiert. Die Darstellung erfolgt in nummerierten Listen (Platzierung).

**Dateien:**

* routes/results/+page.svelte
* Gruppierung mit reduce(), Sortierung mit localeCompare()

4.9 Kleine UX-Details

* Dropdown-Auswahl mit verständlichen Labels: z. B. „Halbmarathon (21 km)“
* Automatisches Füllen von Formularfeldern bei „Bearbeiten“
* Favicon auf eigenes Logo angepasst
* Visuelle Hervorhebung (Farben, Buttons, Icons) über eigenes CSS