Prototyping Projektdokumentation

Name: Andjelina Tosic

E-Mail: tosicand@students.zhaw.ch

URL der deployten Anwendung: [https://greatestofnba.netlify.app

# Einleitung

NBA Legends ist eine Webanwendung zur übersichtlichen Darstellung von Informationen rund um die US-amerikanische Basketballliga NBA. Ziel der Anwendung ist es, Fans und Interessierten eine Plattform zu bieten, auf der sie sich schnell über Teams, Spieler und Spielpläne informieren können, mit Live-Daten und eigener Datenbankanbindung.

Die App kombiniert Daten aus einer externen API (balldontlie.io) mit einer eigenen MongoDB-Datenbank. Die Benutzeroberfläche ist mit SvelteKit 5 Runes-Modus umgesetzt und responsive gestaltet.

Funktionen im Überblick:

* Spielplanübersicht: Zeigt die kommenden Spiele der Saison mit Datum, Uhrzeit und Ort. Die Daten werden live aus der API geladen.
* Teamübersicht & Vergleich: Alle NBA-Teams werden mit Details wie Gründungsjahr, Arena, CEO, Ort und Titelgewinnen dargestellt. Zwei Teams lassen sich interaktiv miteinander vergleichen.
* Spielerverwaltung: Nutzer können eigene Spieler hinzufügen, bearbeiten und löschen. Die Daten werden in einer MongoDB-Datenbank gespeichert. Zusätzlich gibt es Filterfunktionen nach Team, Position oder Lebensstatus.
* Quiz: Eine kleine Quiz-Funktion stellt Fragen zu Teams aus der Datenbank, um das Wissen über die NBA zu testen.
* Videoeinbindung & Navigation: Auf der Spielplan-Seite wird ein YouTube-Video eingebettet, zudem gibt es eine klare Navigation mit Home- und Zurück-Buttons.

Die App wurde als Einzelprojekt entwickelt und verbindet Datenvisualisierung, API-Integration und Datenbankoperationen zu einer modernen, interaktiven Weblösung

*[Beschreibung der einzelnen Pages und Funktionen inklusive Screenshots. Die Funktionalität und die Workflows müssen anhand der Screenshots und Textbeschreibungen nachvollziehbar sein. Der Code muss nicht beschrieben werden.]*

.

# Datenmodell

Ein Bild, das Screenshot, Kreis, Schrift, Schwarzweiß enthält.

KI-generierte Inhalte können fehlerhaft sein.

# Beschreibung der Anwendung

## Startseite

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

KI-generierte Inhalte können fehlerhaft sein.*

Auf der Startseite wird die gesamte Anwendung kurz vorgestellt. Sie dient als zentraler Einstiegspunkt für die Nutzer. Die Seite besteht aus einem hervorgehobenen Titelbereich mit dem Namen der App („NBA Legends“), einem erklärenden Untertitel und farblich hervorgehobenen Buttons zu den vier Hauptbereichen:

* Spieler entdecken: Zeigt Biografien und Bilder berühmter NBA-Spieler
* Teams anzeigen: Präsentiert alle Teams mit Gründung, Arena, CEO und mehr
* Spielplan ansehen: Zeigt kommende NBA-Spiele über eine Live-API
* Quiz: Testet das Wissen über NBA-Teams anhand von Fragen

Darunter befindet sich ein Kachelbereich, der die vier Funktionen nochmals mit Icons und Beschreibung zusammenfasst. Die Seite nutzt die Navigationskomponente Navbar.svelte, die in der gesamten App verwendet wird.

Dateien:

* → lib/components/Navbar.svelte
* → routes/+page.svelte
* → styles.css
* → routes/+layout.svelte

## Spielerübersicht

Route: /player Ein Bild, das Text, Screenshot, Software, Website enthält.

KI-generierte Inhalte können fehlerhaft sein.

Auf der Seite Spielerübersicht (/players) werden alle verfügbaren NBA-Spieler in Kartenform angezeigt. Jede Karte zeigt grundlegende Informationen wie Name, Team, Position, Alter und Lebensstatus. Nutzer:innen können die Liste mithilfe eines Suchfelds sowie Dropdown-Menüs nach bestimmten Kriterien wie Name, Position, Team oder Lebensstatus filtern. Zusätzlich steht ein Button zur Verfügung, um neue Spieler direkt zur Datenbank hinzuzufügen. Für die Darstellung der Spieler wird die Komponente PlayerCard.svelte verwendet. Die Daten werden serverseitig über +page.server.js geladen.

Dateien:

* lib/components/PlayerCard.svelte
* routes/players/+page.svelte
* routes/players/+page.server.js

## Detailansicht Spieler

***Route:*** */players/[player\_id]*

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

KI-generierte Inhalte können fehlerhaft sein.*

*Auf dieser Seite werden alle verfügbaren Informationen zu einem einzelnen Spieler angezeigt. Dazu gehören Team, Position, Alter und Lebensstatus, die prominent neben dem Spielerbild dargestellt werden. Unterhalb folgt eine ausführliche Biografie.  
Diese Biografie wird dynamisch aus einer separaten MongoDB-Collection (BioPlayers) geladen – anhand des Spielernamens. Die Suche erfolgt per Regex-Match auf den Namen des Spielers.*

*Die Daten werden automatisch serverseitig beim Laden der Seite bereitgestellt. Zusätzlich besteht die Möglichkeit, den Spieler aus der Datenbank zu löschen.*

***Dateien:***

* *lib/components/PlayerDetailCard.svelte*
* *routes/players/[player\_id]/+page.svelte*
* *routes/players/[player\_id]/+page.server.js*
* *lib/database.js (Funktion: getBioByName)*

## Spieler erstellen

**Route**:/players/create

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

KI-generierte Inhalte können fehlerhaft sein.*

*Auf dieser Seite können Benutzer neue NBA-Spieler manuell zur Datenbank hinzufügen. Das Formular umfasst Felder für den Namen, das Alter, das Team, die Position, den Lebensstatus (lebend oder verstorben) sowie optional eine Bild-URL. Die Eingaben werden nach dem Absenden serverseitig verarbeitet und über eine definierte POST-Action in die MongoDB-Datenbank gespeichert. Die Altersangabe wird dabei automatisch als Zahl interpretiert. Nach erfolgreicher Erstellung wird der Benutzer zurück zur Übersichtsseite weitergeleitet, auf der der neue Spieler dann erscheint. Die Seite nutzt use:enhance, um das Formular nahtlos in den Runes-Modus von SvelteKit zu integrieren.*

***Dateien:***

* *src/routes/players/create/+page.svelte*
* *src/routes/players/create/+page.server.js*
* *src/lib/database.js*

## Teamübersicht

**Route:** /teams

*Ein Bild, das Text, Screenshot, Logo, Software enthält.

KI-generierte Inhalte können fehlerhaft sein.*

*Auf dieser Seite werden alle NBA-Teams aus der MongoDB-Datenbank visuell in Kartenform dargestellt. Jede Karte enthält wichtige Teaminformationen wie CEO, Arena, Ort und die Anzahl gewonnener Championships. Zusätzlich ist ein Button integriert, über den ein themenspezifisches Quiz zum jeweiligen Team gestartet werden kann. Die Teams werden durch die Server-Load-Funktion geladen und anschließend mit der TeamCard-Komponente dargestellt. Ein Zurück-Button in der oberen rechten Ecke ermöglicht die einfache Navigation zur vorherigen Seite.*

***Dateien:***

* *src/lib/components/TeamCard.svelte*
* *src/routes/teams/+page.svelte*
* *src/routes/teams/+page.server.js*

## Quiz

***Route:*** */quiz/[teamname]*

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

KI-generierte Inhalte können fehlerhaft sein.*

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

KI-generierte Inhalte können fehlerhaft sein.*

*Auf dieser Seite wird ein interaktives Quiz für das jeweils ausgewählte NBA-Team angezeigt. Die Route basiert dynamisch auf dem Teamnamen in der URL, z. B. /quiz/los-angeles-lakers.*

*Die Seite lädt beim Aufruf zunächst alle Teams aus der MongoDB und vergleicht den URL-Teamnamen mit den vorhandenen Teams. Anschließend werden passende Quizfragen zu diesem Team aus der Collection quizfragen geladen. Die Fragen stammen ebenfalls aus der MongoDB und beinhalten jeweils eine richtige sowie mehrere falsche Antwortmöglichkeiten.*

*Der Benutzer kann die Fragen nacheinander beantworten. Nach dem Klick auf „Antwort prüfen“ wird visuell angezeigt, ob die Antwort richtig oder falsch war. Am Ende des Quiz wird die Gesamtpunktzahl angezeigt.*

***Dateien:***

* *→ routes/quiz/[teamname]/+page.svelte*
* *→ routes/quiz/[teamname]/+page.server.js*
* *→ lib/database.js (getQuizByTeamname)*

## Spielplanübersicht

***Route:*** */schedule*

*Ein Bild, das Text, Menschliches Gesicht, Mann, Screenshot enthält.

KI-generierte Inhalte können fehlerhaft sein.*

*Auf dieser Seite werden automatisch alle anstehenden NBA-Spiele der nächsten Tage angezeigt. Die Daten stammen von der* ***externen Balldontlie API*** *und werden beim Laden der Seite über eine API-Route (/api/schedule) abgerufen. Es werden jeweils Heim- und Auswärtsteam, das Datum, die Uhrzeit sowie der Austragungsort der Partie dargestellt.*

*Zusätzlich wird oberhalb der Liste ein eingebettetes YouTube-Video angezeigt. Dies dient zur Einstimmung und verleiht der Seite zusätzliche Dynamik.*

*Im Hintergrund nutzt die API-Funktion getUpcomingGamesNBA() aus der zentralen Datei database.js, welche eine HTTP-Anfrage an die Balldontlie API sendet. Die abgerufenen Daten werden clientseitig verarbeitet und formatiert angezeigt.*

***Dateien:***

* *→ routes/schedule/+page.svelte*
* *→ routes/api/schedule/+server.js*
* *→ lib/database.js (getUpcomingGamesNBA)*

# Erweiterungen

*[Liste der Erweiterungen, die über die Grundanforderungen hinausgehen, siehe Punkt «Erweiterungen» in der Aufgabenstellung. Achten Sie auf Vollständigkeit.* ***Es werden nur die hier beschriebenen Erweiterungen bewertet****. Der Code muss grundsätzlich nicht beschrieben werden, aber es sollten Hinweise gegeben werden, wo die Erweiterung implementiert wurde (Dateiname, Funktionsname, etc.). Screenshots sind nur nötig, falls die Erweiterung nicht unter «Beschreibung der Anwendung» beschrieben wurde.]*

*Beispiel:*

## Watchlist-Filter

Auf der Seite /movies können via Checkbox Filme ausgeblendet werden, die nicht auf der Watchlist stehen. Dies wurde mit einem Formular und einer MongoDB-Query in der Funktion getMoviesOnWatchlist() in der Datei lib/db.js realisiert.

Datei(en):

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