VYSOKÁ ŠKOLA POLYTECHNICKÁ JIHLAVA

Aplikovaná informatika

RESTRUKTURALIZACE WEBOVÝCH STRÁNEK ODDÍLU TAEKWONDO LACEK

Bakalářská práce

Autor práce: Petr Svoboda

Vedoucí práce: Mgr. Hana Vojáčková, Ph.D.

Jihlava 2026

Vysoká škola polytechnická Jihlava

Tolstého 16, 586 01 Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce:	**Petr Svoboda**
Studijní program:	Aplikovaná informatika
Obor:	Aplikovaná informatika
Garant studijního programu:	doc. Ing. Lenka Kuklišová Pavelková, Ph.D.

Název práce:	**Restrukturalizace webových stránek oddílu Taekwondo Lacek**
Vedoucí práce:	Mgr. Hana Vojáčková, Ph.D.
Cíl práce:	Cílem bakalářské práce je kompletní rekonstrukce webových stránek oddílu Taekwondo Lacek. Aktuální webové stránky jsou již velmi zastaralé, jak designově, tak i z hlediska funkčnosti. Změny se budou týkat především modernizace designu, zlepšení responzivity a přidání klíčových funkcí pro zajištění všech potřebných funkcionalit jako je přihlašování závodníků na akce či zkoušky, zobrazení kalendáře, ve kterém budou uvedeny termíny jednotlivých akcí. Dále bude přidán registrační formulář pro nové zájemce o členství v oddílu, implementována galerie obrázků, přidány odkazy na sociální sítě a zpřehledněna navigace.

Abstrakt

Bakalářská práce se zaměřuje na restrukturalizaci webových stránek sportovního oddílu Taekwondo Lacek. Hlavním cílem je vytvoření moderního, uživatelsky přívětivého a funkčně optimalizovaného webu, který bude odpovídat současným technologickým standardům. Práce se zabývá jak grafickou stránkou, tak i funkčními prvky webu.

Frontendová část bude realizována pomocí JavaScriptového frameworku ReactJS, který umožňuje efektivní vývoj komponentového uživatelského rozhraní, a frameworku Tailwind CSS, jenž zajišťuje rychlé a flexibilní stylování. Pro backend je využit serverový framework NodeJS, který poskytuje stabilní a škálovatelnou platformu. Údaje budou ukládány do relační databáze MySQL, která zajistí spolehlivou správu dat.

Výsledkem bakalářské práce bude plně funkční webová aplikace splňující požadavky sportovního oddílu, která přinese vyšší komfort uživatelům i administrátorům.

Klíčová slova

Webové stránky; Taekwondo, týmový web; ReactJS; Tailwind CSS; NodeJS; MySQL

Abstract

The bachelor's thesis focuses on the restructuring of the website for the Taekwondo Lacek sports club. The main goal is to create a modern, user-friendly, and functionally optimized website that meets current technological standards. The thesis addresses both the graphical and functional aspects of the website.

The frontend will be developed using the JavaScript framework ReactJS, which facilitates efficient development of component-based user interfaces, and the Tailwind CSS framework, which ensures fast and flexible styling. The backend will utilize the server framework NodeJS, providing a stable and scalable platform. Data will be stored in a relational MySQL database, ensuring reliable data management.

The result of the bachelor's thesis will be a fully functional web application meeting the requirements of the sports club, providing improved convenience for both users and administrators.

Keywords

Websites; Taekwondo, team website; ReactJS; Tailwind CSS; NodeJS; MySQL

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „**AZ**“).

Byl/a jsem seznámen/a s tím, že na mou bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Podle § 47b zákona o vysokých školách souhlasím se zveřejněním své práce podle Směrnice pro vedení, vypracování a zveřejňování závěrečných prací na VŠPJ, a to bez ohledu na výsledek obhajoby.

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mě požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne 14. dubna 2026

…….

Podpis studenta/ky

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce Mgr. Hana Vojáčková, Ph.D. za odborné vedení, cenné rady a konstruktivní připomínky které mi pomohly při vypracovávání mé bakalářské práce. Dále bych chtěl poděkovat mé rodině za podporu a trpělivost.

Obsah

Seznam obrázků

Seznam zkratek

AI	Artificial intelligence (umělá inteligence)
API	Application Programming Interface
CRUD	Create, Read, Update, Delete (vytvořit, číst, aktualizovat, vymazat)
CSS	Cascading Style Sheets (kaskádové styly)
DOM	Document Object Model(objektový model dokumentu)
GDPR	General Data Protection Regulation (obecné nařízení o ochraně osobních údajů)
HTML	HyperText Markup Language
JS	JavaScript
JSON	JavaScript Object Notation
JWT	JSON Web Token
NPM	Node Package Manager (node balíčkový manager)
UI	User Interface (uživatelské rozhraní)
URL	Uniform Resource Locator(jednotný lokátor zdroje)
VŠPJ	Vysoká škola polytechnická Jihlava

Úvod

Bakalářská práce bude zpracovávat problematiku týkající se restrukturalizace webových stránek sportovního oddílu Taekwondo Lacek. Aktuální webové stránky jsou již velmi zastaralé a v rámci oddílu skoro nepoužitelné, a to jak z pohledu grafického, tak i funkčního hlediska. V mé práci se soustředím na modernizaci designu, zajištění responzivity a přidání všech klíčových funkcí jako je například přihlašování závodníků na akce, zkoušky, přidání formulářů a kalendář akcí. Součástí restrukturalizace též bude aktualizace zastaralých informací a přidání odkazů na sociální sítě, což povede ke zlepší propagace oddílu.

Téma jsem si vybral, protože již několik let trénuji Taekwondo pod vedení pana Petra Lacka a zároveň se zajímám o webové stránky a technologie související s jejich vývojem. Kombinace mých zájmů a zastaralé webové stránky oddílu mi sloužily jako dobrý návrh pro mou bakalářskou práci. Restrukturalizace webových stránek povede k zefektivnění práce a k lepšímu dojmu na náš oddíl, dále to pomůže marketingu.

Pracovat budu ve vývojovém prostředí Visual Studio Code a k realizaci využiji JavaScriptový framework React společně s CSS frameworkem Tailwind CSS pro tvorbu frontendu, backend bude realizován skrze databázi MySQL a serverové řešení pomocí NodeJS. Grafický návrh zpracuji v programu Figma.

Mým cílem je pro náš oddíl vytvořit nové, moderní a responzivní webové stránky, které budou disponovat všemi potřebnými funkcionalitami a lákavým, moderním a zároveň jednoduchým designem, který bude obsahovat barvy Taekwonda i samotného oddílu.

1 Komunikace se zákazníkem

Před samotnou tvorbou webových stránek bylo potřeba se spojit s majitelem oddílu Petrem Lackem a ostatními členy, kteří se aktivně podílí na propagaci, trénincích a organizacích sportovních událostí oddílu.

Pro usměrnění komunikace jsme si založili, hromadný chat na komunikační platformě Messenger, kde jsme spolu vše potřebné konzultovali. Hlavním předmětem konzultace bylo stanovení požadovaných funkcí a návrh grafického designu.

Díky společné komunikaci jsem si uvědomil spoustu důležitých aspektů, na které bych se měl zaměřit a které bych měl vynechat. Jedním z aspektů byl můj návrh evidence umístění jednotlivých závodníků na akcích. Majitelem oddílu mi bylo vysvětleno, že se jedná o velmi dobrý návrh, ale že bohužel ne pokaždé lze všechna umístění evidovat a že nemá k dispozici úspěchy všech závodníků z let minulých.

1.1 Stanovení požadovaných funkcí

Komunikace se zákazníkem mi pomohla stanovit všechny potřebné funkcionality, kterými bude nový web disponovat.

Důležitým prvkem je kalendář akcí, který se bude nacházet na hlavní stránce a bude zobrazovat všechny turnaje či zkoušky. Každý člen oddílu bude mít vytvořen vlastní uživatelský profil nebo si o jeho tvorbu bude moci zažádat skrze formulář. Uživatelské profily budou umožňovat závodníkům se přihlásit na turnaje, zkoušky či si upravit vlastní profilové informace jako jsou: aktuální váha, email, telefonní číslo.

Stávající webové stránky nejsou responzivní a velmi špatně se prohlíží například z mobilního telefonu. Další nutnou funkcionalitou je tedy responzivita.

V neposlední řadě se na mě majitel oddílu obrátil s tím, že si přeje mít na úvodní stránce foto galerii se sloganem a rád by měl v patičce odkazy na týmové sociální sítě.

1.2 Návrh grafického designu

Po stanovení požadovaných funkcí bylo potřeba se zákazníkem vykomunikovat návrh grafického designu nových webových stránek. Společně jsme se shodli na použití barevné palety, která reprezentuje náš oddíl a také samotné Taekwondo. Zmíněná barevná paleta se skládá ze zelené, černé a bílé barvy.

Mou vizí bylo vytvořit jednoduchý, ale zároveň přehledný design, který bude mít za cíl zaujmout co nejvíce návštěvníku s čímž majitel organizace souhlasil. Pro estetiku, bude použito několik ikon, které jsou zdarma k použití na internetu a které nabízí sám React.

2 Teoretická východiska

Kapitola popisující teoretická východiska bakalářské práce, do které se řadí, co to je Taekwondo, úvod do problematiky webových stránek, technologie a nástroje pro vývoj webu.

2.1 Co to je Taekwondo

Taekwondo je bojové umění původem z Koreje, které vznikalo a vyvíjelo se po více než dvě tisíciletí. Dnes je známé nejen jako systém sebeobrany, ale také jako mezinárodní sport. Taekwondo využívá celé tělo k provádění obranných i útočných technik, přičemž klade důraz na harmonii těla a ducha.

Hlavní principy zmíněného bojového umění zahrnují sebeovládání, respekt k druhým a rozvoj fyzické i psychické kondice. Taekwondo podporuje nejen fyzickou zdatnost, ale také duševní rovnováhu, přičemž učí cvičence etickým hodnotám a zodpovědnému využití nabytých dovedností. Kodex Taekwonda zdůrazňuje zákaz nečestného útoku a nepřiměřeného použití síly.

2.1.1 Hlavní disciplíny Taekwonda

Taekwondo se dělí na čtyři základní disciplíny, které se zaměřují na různé aspekty cvičení:

- **Kyorugi** (zápas): zahrnuje řízený i volný boj s důrazem na techniku a fyzickou zdatnost
- **Poomsae** (souborná cvičení): představují sestavy pohybů, které kombinují obranné a útočné techniky a zlepšují koordinaci a soustředění
- **Kyokpa** (přerážecí techniky): testují sílu, přesnost a koncentraci cvičence
- **Hosinsul** (sebeobrana): praktické techniky pro obranu proti různým druhům útoků, s důrazem na rychlou reakci a adaptaci

Taekwondo jako celek klade důraz nejen na fyzické dovednosti, ale také na rozvoj morálních hodnot a odpovědného přístupu k naučeným technikám. Je to cesta k seberealizaci, harmonii a osobnímu růstu (WorldTaekwondo, 2024).

2.2 Úvod do problematiky webových stránek

Webové stránky jsou v dnešní době již nedílnou součástí moderního světa. Pomocí webových stránek jsme schopni se na internetu prezentovat, předávat informace, nabízet služby či komunikovat.

Každá webová stránka je složena ze 3 základních stavebních kamenů:

- **HTML:** struktura a sémantika
- **CSS:** vizuální prezentace a responzivita
- **JavaScript:** dynamika a interaktivita

Kapitola se zabývá definicí webových stránek, jejich významem, historií, současnými trendy a bezpečností.

2.3 Definice webových stránek

Webová stránka je dokument nebo soubor dokumentů zpřístupněný na internetu, který je možné zobrazit pomocí webového prohlížeče. Podle definice organizace W3C (World Wide Web Consortium) je webová stránka strukturována pomocí jazyka HTML a obvykle obsahuje text, multimediální obsah a odkazy na jiné stránky. Každá webová stránka má svou jedinečnou adresu (URL), která ji identifikuje na internetu (W3C, 2024).

Webové stránky hrají zásadní roli v mnoha oblastech, jako je podnikání, vzdělávání, zdravotnictví a vládní komunikace. Slouží jako nástroj pro:

- **Prezentaci informací:** organizace mohou sdílet aktuální zprávy, produkty a služby
- **Marketing a reklamu:** podniky využívají web k oslovení širokého publika za nízké náklady
- **Komunikaci:** uživatelé mohou prostřednictvím webu sdílet informace a interagovat
- **E-commerce:** elektronické obchody umožňují snadný nákup a prodej zboží (Chaffey & Ellis-Chadwick, 2022)

2.4 Význam webových stránek

Webová stránka je dokument nebo soubor dokumentů zpřístupněný na internetu, který je možné zobrazit pomocí webového prohlížeče. Podle definice organizace W3C (World Wide Web Consortium) je webová stránka strukturována pomocí jazyka HTML a obvykle obsahuje text, multimediální obsah a odkazy na jiné stránky. Každá webová stránka má svou jedinečnou adresu (URL), která ji identifikuje na internetu (W3C, 2024).

2.5 Historický vývoj

První webová stránka byla vytvořena v roce 1991 Timem Berners-Leem, který je považován za otce World Wide Webu (Berners-Lee, 1991). V prvních letech byly webové stránky jednoduché a zaměřené na text. Výrazný pokrok nastal s příchodem CSS (Cascading Style Sheets) v roce 1996, které umožnily lepší design a stylizaci. Nástup Webu 2.0 v roce 2004 přinesl interaktivitu a uživatelsky generovaný obsah. V současnosti jsou webové stránky sofistikované, zaměřené na uživatelský zážitek a přizpůsobené mobilním zařízením (O'Reilly, 2005).

2.6 Současné trendy

Mezi klíčové trendy ve vývoji webových stránek patří:

- **Responzivní design:** optimalizace webů pro různá zařízení, včetně mobilů a tabletů (Marcotte, 2010)
- **Použití AI:** implementace umělé inteligence pro personalizaci obsahu a zlepšení uživatelského zážitku (Smith et al., 2023)
- **Progressive Web Apps (PWA):** aplikace, které kombinují funkce webu a mobilních aplikací
- **Minimalismus a rychlost:** důraz na jednoduchý design a rychlé načítání stránek

2.7 Bezpečnost

Bezpečnost webových stránek je klíčová pro ochranu uživatelských dat a prevenci kybernetických útoků. Mezi nejčastější hrozby patří:

- **Phishing:** podvodné stránky, které se snaží získat citlivé informace
- **SQL injection:** neoprávněný přístup k databázím pomocí manipulace s dotazy
- **Cross-Site Scripting (XSS):** vložení škodlivého kódu do webové aplikace
- **Použití SSL/TLS:** šifrování komunikace mezi serverem a klientem jako standard ochrany dat (RFC 5246)

Prevence zahrnuje pravidelnou aktualizaci softwaru, používání bezpečnostních certifikátů a monitorování aktivit na webových serverech (OWASP, 2024).

Pokud webové stránky pracují s citlivými údaji je vhodné se zaměřit na GDPR – obecné nařízení o ochraně osobních údajů je nařízení Evropského parlamentu a Rady (EU) 2016/679 ze dne 27. dubna 2016 o ochraně fyzických osob v souvislosti se zpracováním osobních údajů a o volném pohybu osobních údajů. Nařízení představuje nový právní rámec ochrany osobních údajů v evropském prostoru, které od 25. května 2018 přímo stanovuje pravidla pro zpracování osobních údajů. Včetně práv subjektu. Cílem GDPR je ochrana osobních údajů v dnešní době (Ministerstvo vnitra České republiky, 2024).

Souhlas se zpracováním osobních údajů musí být svobodný, konkrétní, informovaný a jednoznačný projev vůle, kterým subjekt údajů dává prohlášením či jiným zjevným potvrzením své svolení ke zpracování osobních údajů. Jde o aktivní a dobrovolný projev vůle subjektu údajů, ke kterému nesmí být nucen. Může se například jednat o zaškrtnutí políčka při návštěvě internetové stránky. Mlčení, předem zaškrtnutá políčka nebo nečinnost by tudíž neměly být považovány za souhlas (EUR-Lex, 2024).

Subjekt údajů má právo na to být informován o zpracování jeho osobních údajů. Přístupem k osobním údajům se rozumí právo subjektu získat od správce informaci (potvrzení), zda jsou či nejsou jeho osobní údaje zpracovávány a pokud jsou zpracovávány, má subjekt údajů právo osobní údaje získat a zároveň má právo získat následující informace:

- Účely zpracování
- Kategorie dotčených osobních údajů
- Příjemci nebo kategorie příjemců, kterým osobní údaje byly nebo budou zpřístupněny
- Plánovaná doba, po kterou budou osobní údaje uloženy
- Existence práva požadovat od správce opravu nebo výmaz osobních údajů, právo vznést námitku
- Právo podat stížnost u dozorového úřadu
- Veškeré dostupné informace o zdroji osobních údajů, pokud nejsou získány od subjektu údajů
- Skutečnosti, že dochází k automatizovanému rozhodování, včetně profilování (Ministerstvo vnitra České republiky, 2024)

2.8 Technologie a nástroje pro vývoj webu

Dříve se pro vývoj webových stránek používalo pouze HTML, CSS, JavaScript či jiné technologie, ale komplexnost mých webových stránek si vyžaduje použít frameworky a knihovny, které mi celý proces zjednoduší a přispějí k lepší optimalizovatelnosti.

K realizaci webových stránek použiji moderní technologie, které jsou díky své efektivnosti, množství dokumentace a škálovatelnosti řazeny mezi nejoblíbenější a nejspolehlivější. Tvorba frontendu bude realizována pomocí frameworků Tailwind CSS, React. Backend bude postaven na serverovém frameworku NodeJS a databázi MySQL. Grafický návrh bude zpracován v programu Figma.

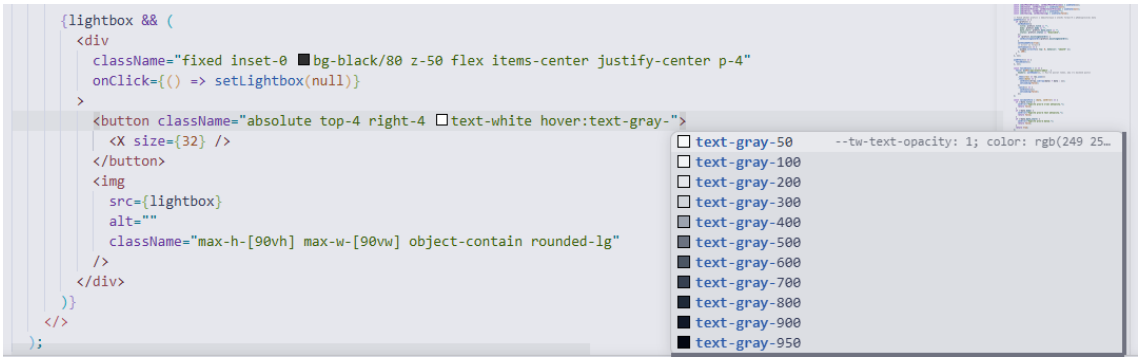
2.8.1 Tailwind CSS

Vzhled webových stránek bude zajištěn pomocí moderního CSS frameworku Tailwind CSS. Pomocí zmíněného frameworku budu schopen vytvořit atraktivní, responzivní web rychlým a efektivním způsobem. Framework je utility-first a zaměřuje se na psaní malých, specifických tříd. Třídy se aplikují přímo do HTML, čímž se uživatel může vyvarovat nadbytečnému psaní vlastního CSS kódu.

Předdefinované třídy pro styling zrychlují proces psaní kódu, a tím se zvyšuje i efektivita vývojáře. Samotný přístup zároveň zjednodušuje kód.

Jedním z hlavních důvodů výběru frameworku byla podpora responzivního designu. Framework je navržen tak, aby byl mobile-first, což znamená, že je primárně optimalizován pro mobilní zařízení. Vzhledem k tomu, že na web, který tvořím, budou uživatelé přistupovat primárně z mobilních zařízení, je pro mě Tailwind CSS ideální.

Dále Tailwind CSS automaticky odstraňuje veškerý nepoužívaný CSS při vytváření produkční verze, což znamená, že výsledný CSS soubor je co nejmenší. Ve skutečnosti většina projektů postavených na Tailwindu doručí klientovi CSS soubor menší než 10 kB (Tailwind CSS, 2024).



Obr. 1: Tailwind CSS syntaxe

Zdroj: Vlastní

2.8.2 React

React je JavaScriptová knihovna sloužící k vytváření uživatelských rozhraní, zejména pro aplikace, kde je potřeba efektivně aktualizovat a renderovat obsah v uživatelském rozhraní. React byl původně vyvinut firmou Facebook (nyní Meta). V dnešní době se jedná o jeden

z nejpopulárnějších frameworků pro vývoj webových aplikací, a to díky své flexibilitě, rychlosti a jednoduchosti použití. Disponuje širokou komunitou vývojářů a velkým množstvím projektů ze kterých lze čerpat inspiraci (React, 2024).

Výhody Reactu

1. **Hooky** – React nabízí širokou škálu vestavěných hooků, které usnadňují správu stavů, kontextů a výkonu v komponentách. Hooky jsou funkce, které umožnují používat různé vlastnosti Reactu přímo v komponentách a přispívají k efektivnější a čitelnější struktuře aplikací. Můžeme si také vytvořit vlastní hooky, které mohou být definovány jako funkce, které využívají jiné hooky k vytvoření specifických logik a vzorců pro opakované použití (React, 2024).
   Typy hooků:
   - **State:** umožňují komponentám uchovávat informace, například uživatelský vstup nebo stav vybrané položky
   - **Context:** umožňují sdílet data mezi vzdálenými komponentami, aniž by bylo nutné předávat je prostřednictvím props
   - **Ref:** umožňují ukládat informace, které neovlivňují vykreslování, například DOM uzly nebo ID časovačů
   - **Effect:** umožňují komponentám synchronizovat se s externími systémy
   - **Performance:** slouží k optimalizaci výkonu komponent
   - **Other:** hooky, které se využívají při vývoji knihoven

```jsx
import React, { useState, useEffect } from 'react';

const App = () => {
    const [name, setName] = useState('World');

    useEffect(() => {
      document.title = `Hello, ${name}`;
    });

    return (
      <div className="App">
        <h1>Hello, {name}!</h1>
        <button onClick={() => setName('James')}>
          Click me to change the name
        </button>
      </div>
    );
  }
```

Obr. 2: Ukázka hooku
Zdroj: Vlastní

2. **Komponenty** – Komponenta je základní stavební kámen Reactu. Jedná se o izolované a opakovatelně použitelné části UI. Komponenty mohou být složeny z jiných komponent, a to umožňuje vytvářet složitější UI z menších a snadno spravovatelných

částí. Pokud je potřeba, aby komponenta reagovala na změny, tak se používají stavy, které jsou spravovány uvnitř komponenty (React, 2024).

```
You, 18 seconds ago | 1 author (You)
function Showcase() {

  return (
    <>

      <div class="w-full max-h-96 object-cover h-96 overflow-hidden rounded-2xl bg-customBlack">

       <img src="../src/assets/Showcase/main.jpg" alt="" class="w-full h-fit brightness-50 opacity-75 -z-20" />
       <figcaption class="absolute px-4 text-lg text-white top-32">
    <div>
      <p class="text-3xl">Sportovní klub</p>
       <p class="text-5xl font-mono">Taekwondo <span class="text-customGreen text-5xl font-mono">Lacek</span></p>
       <p>Taekwondo - rychleji, výš, silněji, ...</p>
       </div>
  </figcaption>
      </div>
    </>
  );
}

export default Showcase;
```

Obr. 3: Příklad komponenty

Zdroj: Vlastní

Komponenty mohou přijímat vstupní data, kterým se odborně říká props. Používají se v případech, když je potřeba dynamicky měnit hodnoty uvnitř komponenty.

```
function Showcase({ title, subtitle, description, imgSrc }) {
    return (
      <>
        <div className="w-full max-h-96 object-cover h-96 overflow-hidden rounded-2xl bg-customBlack">
          <img src={imgSrc} alt="Showcase" className="w-full h-fit brightness-50 opacity-75 -z-20" />
          <figcaption className="absolute px-4 text-lg text-white top-32">
            <div>
              <p className="text-3xl">{title}</p>
              <p className="text-5xl font-mono">
                {subtitle} <span className="text-customGreen text-5xl font-mono">Lacek</span>
              </p>
              <p>{description}</p>
            </div>
          </figcaption>
        </div>
      </>
    );
  }

  export default Showcase;
```

Obr. 4: Komponenta se vstupy

Zdroj: Vlastní

```jsx
import React from 'react';
import Showcase from './Showcase';

function App({ clubData }) {
  if (!clubData) {
    return <div>Načítání...</div>;
  }

  return (
    <div>
      <Showcase
        title={clubData.title}
        subtitle={clubData.subtitle}
        description={clubData.description}
      />
    </div>
  );
}

export default App;
```

Obr. 5: Práce se vstupy

Zdroj: Vlastní

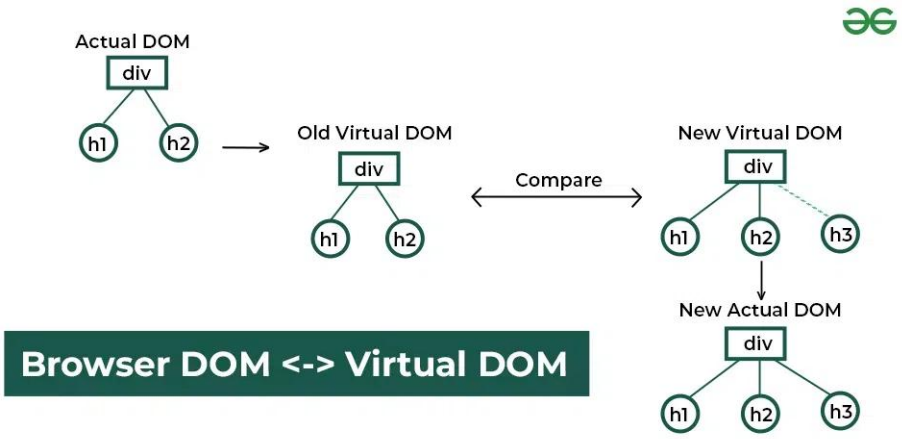
Pro použití komponent je musíme naimportovat a poté je použít jako JSX elementy.

```jsx
import { StrictMode } from 'react';
import { createRoot } from 'react-dom/client';
import './index.css';
import App from './App.jsx';
import Header from './Components/Header.jsx';
import Footer from './Components/Footer.jsx';
import Body from './Components/Body.jsx';
import Fighter from './Components/Fighter.jsx';
import Showcase from './Components/Showcase.jsx';

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <>
      <Header />

      <main className="pt-20">

        <Body>
        <Showcase />
          <App />
          <Fighter />
          <App />
        </Body>
      </main>
      <Footer />
    </>
  </StrictMode>,
);
```

Obr. 6: Použití komponenty

Zdroj: Vlastní

3. **Virtuální DOM** – Další významnou výhodou je virtuální DOM, což je paměťová reprezentace skutečného DOM, která optimalizuje vykreslování UI. Při změnách stavu React vytváří novou stromovou strukturu Virtuální DOM a pomocí algoritmu určuje minimální změny potřebné pro aktualizaci skutečného DOM. To zrychluje výkon díky efektivnímu vykreslování, snížení přímých interakcí se skutečným DOM a dávkování aktualizací. Virtuální DOM také standardizuje vykreslování napříč prohlížeči. Nevýhodami jsou zvýšená paměťová náročnost a složitost pro jednoduché aplikace (GeeksforGeeks, 2024).



Obr. 7: Porovnání DOMu

Zdroj: https://www.geeksforgeeks.org/reactjs-virtual-dom/ (2024)

4. **JSX** – JavaScript XML je syntaxe, která je rozšířením jazyka JavaScript. Používá se v Reactu a umožňuje psát HTML strukturu přímo v JavaScriptovém kódu. JSX vypadá jako HTML, ale má plnou sílu JavaScriptu. Když JSX kód projde kompilací, stává se z něj běžný JavaScript, který vytváří objekty, které React používá k renderování. Díky automatickému escapování hodnot před jejich zobrazením jsou vstupy bezpečně zpracovány, což pomáhá zabránit XSS útokům (React, 2024).

```jsx
import React from 'react';

function App() {
  return (
    <div style={{ textAlign: 'center', marginTop: '20px' }}>
      <h1>Welcome to React!</h1>
      <p>This is a simple JSX example.</p>
      <button onClick={() => alert('Button clicked!')}>Click Me</button>
    </div>
  );
}

export default App;
```

Obr. 8: Ukázka JSX

Zdroj: Vlastní

2.8.3 NodeJS

NodeJS je open-source, multiplatformní runtime prostředí pro JavaScript, které umožňuje vývojářům vytvářet servery, webové aplikace a nástroje pro příkazový řádek a skripty.

Díky své asynchronní a událostmi řízené architektuře je NodeJS ideální pro aplikace, které musí zvládnout vysoký počet požadavků v jeden čas. Mezi ně se například řadí real-time aplikace či API.

NodeJS podporuje širokou škálu knihoven či modulů, které uživateli usnadňují vývoj. Celkově se jedná o velmi flexibilní prostředí, které je hojně podporováno rozsáhlou komunitou vývojářů (NodeJS, 2024).

```
1
2   // server_tst.js
3   const http = require('http');
4
5   const host = '127.0.0.1';
6   const port = 2000;
7
8   const server = http.createServer((req, res) => {
9     res.statusCode = 200;
10    res.setHeader('Content-Type', 'text/plain');
11    res.end('This is a test of Node.js on a local computer.');
12  });
13
14  const str = 'Node.js server running at http://' + host + ':' + port + '/.'
15  server.listen(port, host, () => {
16    console.log(str);
17  });

L: 1 C: 1   JavaScript   Unicode (UTF-8)   Unix (LF)   Saved: 2:42:42 PM   413 / 48 / 17
```

Obr. 9: Ukázka kódu NodeJS

Zdroj: https://www.techtarget.com/whatis/definition/Nodejs (2024)

Další nedílnou součástí NodeJS je NPM – správce balíčků pro platformu NodeJS, který umožňuje vývojářům snadno a znovu využívat kód, stejně jako spravovat závislosti v jejich projektech. NPM poskytuje online úložiště pro publikování open-source NodeJS projektů a také příkazový řádek pro možnou interakci s online úložištěm. Jedná se o nedílnou součást ekosystému NodeJS a je hraje významnou roli při vývoji moderních JavaScriptových aplikací (NPM, 2024).

NPM umožňuje

- **Instalaci balíčků:** vývojáři mohou snadno a rychle přidávat knihovny a nástroje do svých projektů pomocí jednoduchých příkazů v terminálu
- **Správu závislostí:** NPM automaticky sleduje a spravuje závislosti mezi balíčky, což umožňuje a usnadňuje udržování konzistentního vývojového prostředí
- **Publikování balíčků:** vývojáři mezi sebou mohou sdílet své vlastní balíčky a veřejně je publikovat s komunitou pomocí veřejného NPM registru
- **Verzování:** NPM podporu správy verzí balíčků, což umožňuje vývojářům specifikovat a instalovat konkrétní verze balíčků, které potřebují využít do svých projektů (NPM, 2024)

Obr. 10: Logo NPM

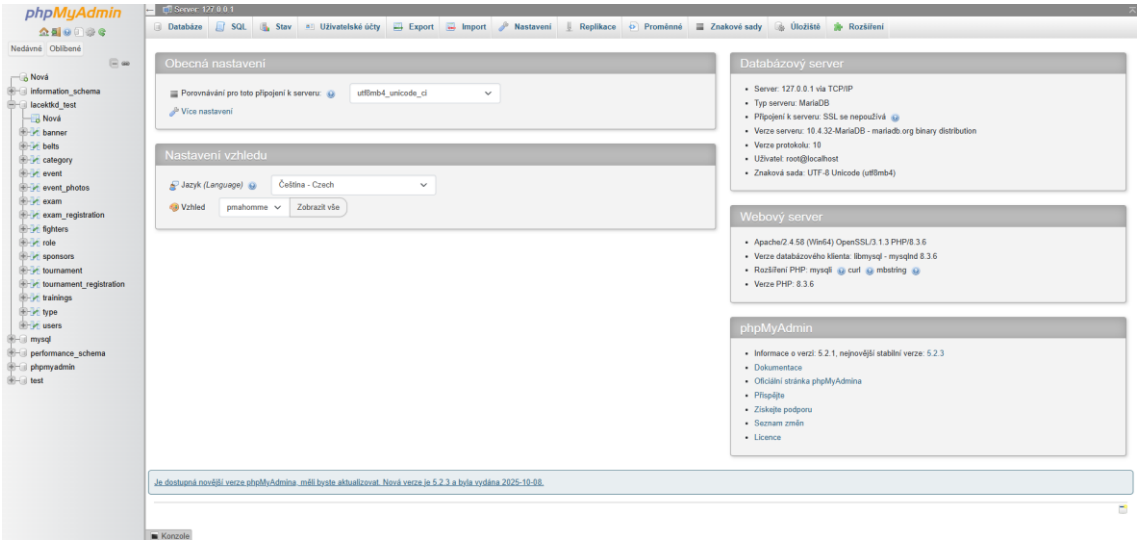
Zdroj: https://en.m.wikipedia.org/wiki/File:Npm-logo.svg (2024)

2.8.4 MySQL

MySQL je open-source relační databázový systém, který se stal nejpopulárnějším na světě díky své spolehlivosti a výkonu. Nabízí různé edice, včetně bezplatného Community Server a placené MySQL Enterprise, která je určena pro komerční aplikace a zahrnuje pokročilou bezpečnost a podporu.

Mezi jeho klíčové vlastnosti patří podpora pro vysokou dostupnost, optimalizace výkonu prostřednictvím HeatWave, a možnosti integrace s cloudovými platformami. MySQL je ideální pro webové, cloudové a embedded aplikace, které vyžadují efektivní správu databází.

Dále je velmi známý pro svou jednoduchost, flexibilitu a širokou komunitu vývojářů (MySQL, 2024).



Obr. 11: Prostředí MySQL

Zdroj: Vlastní

2.8.5 Figma

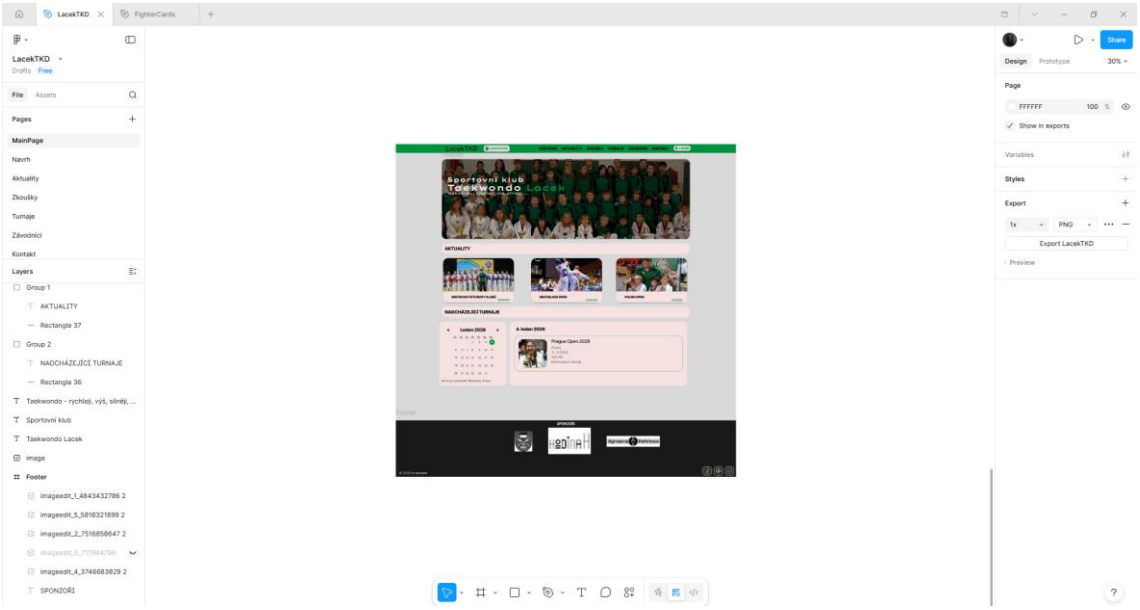
Figma je cloudový nástroj pro návrh uživatelského rozhraní, který umožňuje týmům či jednotlivcům pracovat v reálném čase na tvorbě webových stránek, mobilních aplikací a dalších digitálních produktů.

Figma je velmi oblíbená právě pro možnost spolupráce v reálném čase, možnosti tvořit komponenty a knihovny, které lze opakovaně využívat, což zajišťuje konzistenci v designu napříč projekty.

Díky cloudovému řešení není potřeba instalovat žádný software, přestože Figma disponuje vlastní aplikací a lze na projektech pracovat téměř odkudkoliv.

Nástroj obsahuje funkcionalitu prototypování, což je proces, při kterém se vytváří interaktivní verze návrhu, která přímo imituje chování skutečného webu či aplikace.

V prémiové verzi nástroje lze samotný návrh i přímo naprogramovat (Figma, 2024).

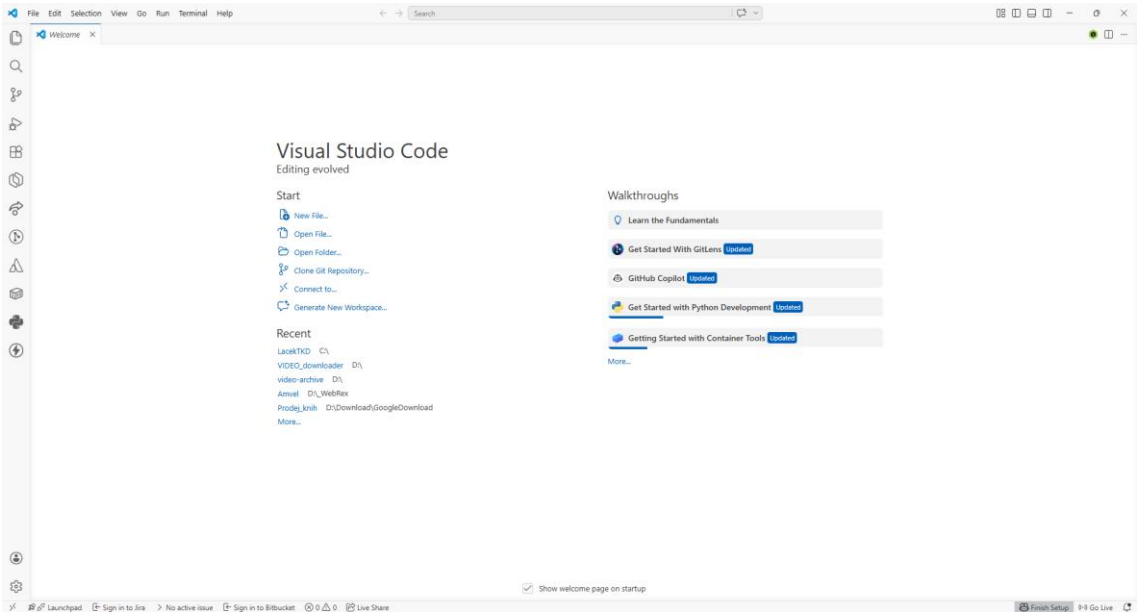


Obr. 12: Prostřední Figmy
Zdroj: Vlastní

2.8.6 Visual Studio Code

Visual Studio Code je open-source editor zdrojových kódu od společnosti Microsoft. Jedná se o univerzální program využívaný pro vývoj webových, mobilních a desktopových aplikací. Program je dostupný pro platformy Windows, Linux, macOS což z něj činí velice rozšířený software, který je oblíbený převážně mezi vývojáři.

Nabízí integrovanou podporu pro Git, terminál, ladění a také obsahuje možnost přidat si do prostředí širokou řadu rozšíření. Podporuje širokou škálu programovacích jazyků jako jsou například JavaScript, Python, C++, a mnoho dalších. Díky rozšiřitelnosti lze pomocí rozšíření přidat podporu i pro další jazyky (Visual Studio Code, 2024).



Obr. 13: Prostředí Visual Studio Code

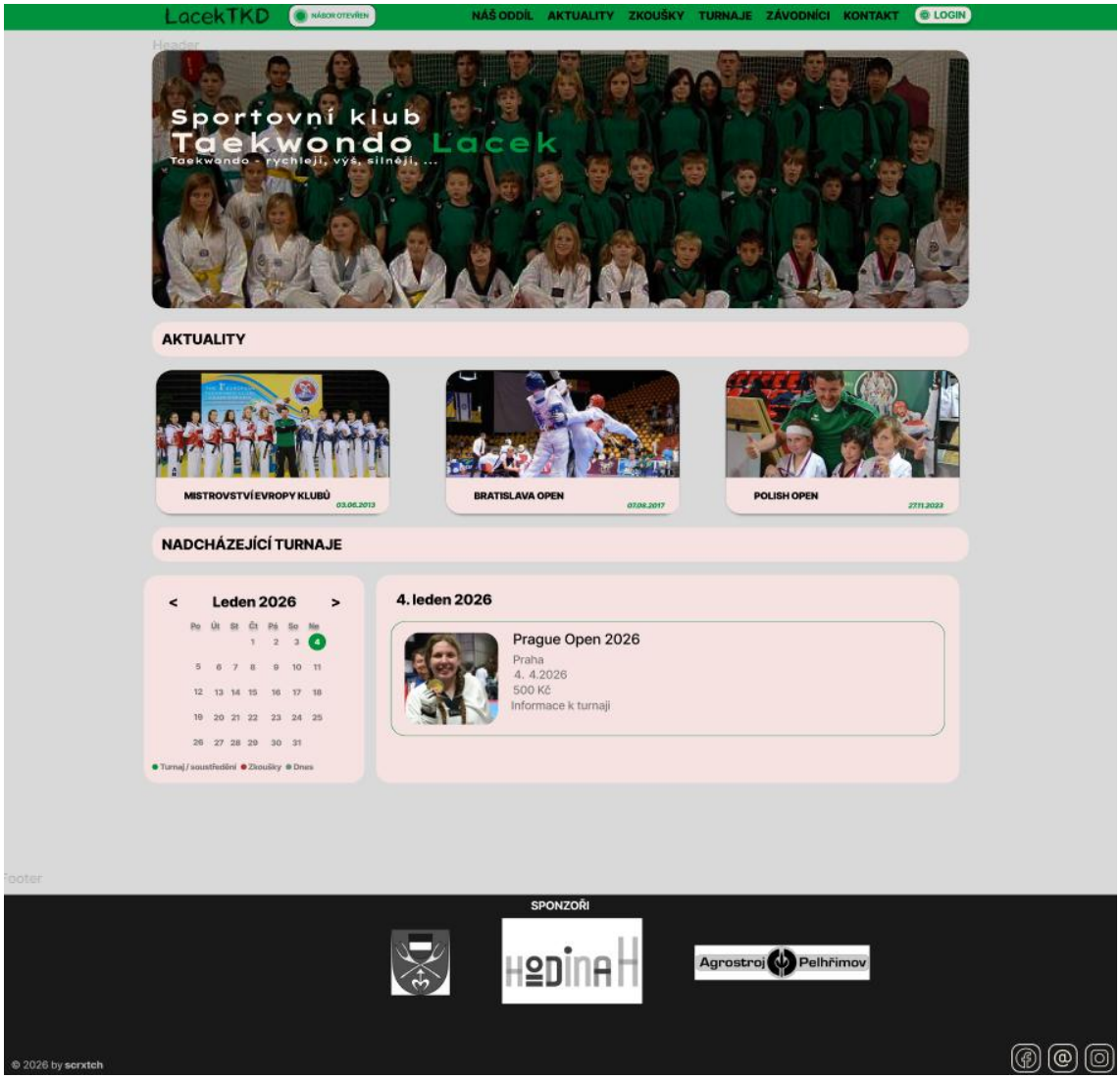
Zdroj: Vlastní

3 Návrh

Kapitola pojednává o návrhu uživatelského prostředí, popisu uživatelských rolí, které budou se samotnou webovou stránkou interagovat a v neposlední řadě kapitola obsahuje návrh databáze, která bude zajišťovat práci s daty. Na základě poznatků uvedených v kapitole bude následně vyhotoveno finální řešení mé bakalářské práce.

3.1 Uživatelské prostředí

Uživatelské prostředí bude obsahovat 3 základní barvy, a to krémově bílou, zelenou v týmovém odstínu a černou. Celé uživatelské prostředí bude vytvořeno v minimalistickém stylu a bude kombinovat prvky moderního designu, čímž bude zajištěna větší přehlednost webových stránek. Webová stránka bude mít fixní mezery mezi jednotlivými prvky a bude zde použito zakulacení ostrých hran. Důraz dále bude kladen na responzivitu webu, aby bylo možné web plnohodnotně zobrazit na jakémkoliv zařízení. Uživatelské prostředí bylo navrženo v grafickém programu Figma.



Obr. 14: Návrh úvodní stránky

Zdroj: vlastní

3.2 Uživatelské role

Nové řešení bude obsahovat 3 typy uživatelských účtů. Každý typ bude mít své pravomoce na základě kterých bude moci s nově navrženým systémem interagovat. Mezi role patří administrátor, trenér, závodník. S webovou stránkou bude moci interagovat i běžný návštěvník, avšak nebude mít přístup do administrace a nebude mít možnost přihlašovat se na různé turnaje nebo zkoušky. Definování uživatelských rolí je velmi důležitým aspektem, který slouží k jasnému stanovení pravomocí a možností jednotlivých uživatelů. Zároveň slouží jako pojistka proti neautorizovanému přístupu do administrace.

Administrátor – role disponující nejvyššími pravomocemi, administrátor má plnou kontrolu nad webem, může přidávat, mazat a upravovat turnaje, zkoušky, aktuality. Pracovat s bannery, odkazy, sponzory. V neposlední řadě má přístup ke všem uživatelským účtům a může spravovat přihlášky uživatelů včetně jejich profilových informací.

Trenér – může přidávat, mazat a upravovat turnaje, zkoušky, aktuality, upravovat závodníkům profilové informace jako jsou například technické stupně, aktuální váhová kategorie a jejich úspěchy. Trenéři mají pravomoce přihlašovat a odhlašovat uživatele z turnajů a zkoušek.

Závodník – po registraci a schválení uživatelského účtu administrátorem má závodník možnost přihlásit se na turnaje a zkoušky. Může si upravit informace týkající se jeho účtu jako je například emailová adresa, telefonní číslo, heslo nebo aktuální váhová kategorie, která je velmi důležitým aspektem při přihlašování se na zkoušky. Uživatelé s přiděleným oprávněním samozřejmě budou mít také přístup k webovým stránkám jako běžný návštěvník.

Návštěvník – návštěvník webu má přístup ke všem veřejným částem webu. To zahrnuje turnaje, zkoušky, aktuality, informace týkající se oddílu, závodníky případně sestavy potřebné k úspěšnému zvládnutí zkoušek na vyšší technický stupeň pásu. V případě zájmu o členství v řadách oddílu se zde nachází sekce kontaktů i registrační formulář.

3.3 Databáze

Pro správnou funkcionalitu webových stránek je potřeba navrhnout kvalitní a dobře strukturovanou databázi, která bude disponovat všemi potřebnými entitami a relačními vztahy mezi nimi. Pro samotný návrh jsem využíval webovou stránku dbdiagram, ve které jsem si specifikoval všechny potřebné entity a vazby mezi nimi. Databáze je implementována v relačním databázovém systému MySQL a obsahuje celkem 15 tabulek, které jsou vzájemně provázány pomocí cizích klíčů.

3.3.1 Popis jednotlivých tabulek

Fighters – závodníci

Tabulka fighters představuje centrální entitu celého systému a uchovává informace o všech závodnících oddílu. Každý záznam obsahuje základní osobní údaje jako jméno (name), příjmení (surname) a datum narození (birth). Dále jsou zde uloženy záznamy týkající se úspěchů závodníků – cizí klíč belts_id odkazující na aktuální technický stupeň závodníka, category_id určující věkovou kategorii a actual_weight_category reprezentující aktuální váhovou kategorii závodníka v kilogramech.

Tabulka také obsahuje booleovské hodnoty best a legend, které slouží k označení výjimečných závodníků zobrazovaných na veřejné části webu, a stav active určující, zda je závodník aktuálně aktivním členem oddílu či nikoliv. Sloupec img_path uchovává cestu k profilové fotografii závodníka.

Belts – pásky

Tabulka belts obsahuje všechny dostupné technické stupně v Taekwodnu, od žlutého pásku přes barevné pásky až po jednotlivé dany černého pásku. Každý stupeň má svůj název (belt_name), označení cupu (cup, např. „8. CUP" nebo „1. DAN"), cenu zkoušky (price), cestu k obrázku pásku (img_path), doplňující informace (info) a cestu k výukovému videu (video_path). Tabulka je ve vztahu 1:N s tabulkou fighters – jeden technický stupeň může mít více závodníků, ale každý závodník má právě jeden aktuální stupeň.

Category – kategorie

Tabulka category definuje věkové kategorie závodníků používané v soutěžích Taekwondo. Každá kategorie je charakterizována názvem (name, např. Youth, Cadet, Junior, Senior, Ultra) a věkovým rozsahem vyjádřeným sloupci min a max. Vztah s tabulkou fighters je opět 1:N – v jedné kategorii může závodit více závodníků, ale každý závodník patří do právě jedné kategorie.

Users – uživatelé

Tabulka users spravuje uživatelské účty. Každý uživatel má přihlašovací jméno (login), zahashované heslo (password) pomocí byCriptu, e-mail (email) a telefonní číslo (phone). Sloupec role_id určuje uživatelskou roli v systému a fighter_id umožňuje provázání uživatelského účtu s konkrétním závodníkem. Propojení je ve vztahu 1:1 – jeden závodník může mít právě jeden uživatelský účet a jeden uživatelský účet může být přiřazen právě jednomu závodníkovi. Provázání umožňuje závodníkům přihlásit se do systému a spravovat vlastní profil nebo své registrace na turnaje či zkoušky. Tabulka obsahuje i sloupec status (status), který určuje, v jakém stavu se nachází uživatelský účet po registraci, na výběr jsou 3 možnosti pending pro podané žádosti, approved pro potvrzené účty a rejected pro zamítnuté účty.

Role

Tabulka role definuje role a oprávnění uživatelů v systému. Aktuálně jsou definovány role jako administrátor (admin), trenér (trainer), závodník (user). Vztah s tabulkou users je 1:N – jedna role může být přiřazena více uživatelům, ale každý uživatel má právě jednu roli. Tabulka je klíčová pro řízení přístupu k administraci systému a k přihlašování uživatelů na zkoušky či turnaje.

Tournament – turnaje a soustředění

Tabulka tournament uchovává informace o plánovaných i proběhlých turnajích a soustředěních oddílu. Každý záznam obsahuje název akce (name), místo konání (location), startovné (price), datum začátku a konce (start_date, end_date), datum uzávěrky přihlášek (registrable_date), doplňující informace (info) a cestu k fotografii nebo plakátu akce (img_path). Sloupec type_id odkazuje na typ akce a users_id eviduje, který uživatel akci vytvořil. Sloupec google_event_id slouží pro interakci a zobrazování turnajů a soustředění pomocí integrace Google Calendar API. Nachází se zde i sloupec status (status), který obsahuje záznam, zda turnaj či zkouška již proběhly.

Tournament_registration – registrace na turnaje

Tabulka tournament_registration realizuje vazební tabulku vztahu M:N mezi závodníky a turnaji. Závodník se může registrovat na více turnajů a jeden turnaj může mít více registrovaných závodníků. Každá registrace obsahuje cizí klíč fighter_id odkazující na závodníka, tournament_id určující turnaj na který je závodník registrován a dosažené umístění (place), které lze vyplnit zpětně po skončení turnaje.

Type – typy akcí

Tabulka type je pomocná tabulka definující typy akcí, například Turnaj nebo Soustředění. Je ve vztahu 1:N s tabulkou tournament – jeden typ může zahrnovat více turnajů.

Event – aktuality

Tabulka event uchovává aktuality a příspěvky zveřejňované na webu oddílu. Každá aktualita má název (title), textový obsah (body), stav (status), datum zveřejnění (date_start), časové razítko poslední úpravy (created_at) a cestu k titulní fotografii (photo). Cizí klíč user_id odkazuje na autora příspěvku. Vztah s tabulkou users je 1:N – jeden uživatel může vytvořit více aktualit.

Event_photos – fotogalerie aktualit

Tabulka event_photos rozšiřuje systém aktualit o podporu více fotografií u jednoho příspěvku. Každý záznam obsahuje cestu k souboru (img_path), pořadí fotografie (sort_order) pro správné zobrazení galerie a cizí klíč event_id odkazující na příslušnou aktualitu. Vazba na tabulku event je definována s kaskádovým mazáním (ON DELETE CASCADE), což zajišťuje, že při smazání aktuality se automaticky odstraní i všechny její fotografie. Vztah je 1:N – jedna aktualita může mít více fotografií.

Sponsors – sponzoři

Tabulka sponsors eviduje sponzory oddílu zobrazované na webu. Každý sponzor má název (sponsor_name), volitelný odkaz na web (url) a cestu k logu (img_path).

Banner

Tabulka banner spravuje reklamní bannery zobrazované na úvodní stránce webu. Každý banner má název (banner_name), cestu k obrázku (img_path) a příznak active určující, zda je banner aktuálně zobrazen.

Trainings – tréninky

Tabulka trainings uchovává pravidelný tréninkový rozvrh oddílu. Každý záznam definuje den začátku a konce tréninku (day_start, day_end), typ tréninku (type, např. začátečníci nebo pokročilí) a čas začátku i konce (time_start, time_end).

Exam – zkoušky

Tabulka zkoušky obsahuje informace o zkouškách, mezi tyto informace patří název zkoušky (title), potřebné informace (description), datum konání (date), datum do kterého je možné se na zkoušky přihlásit (registrable_date), místo konání (location), cenu zkoušky (price), status (status) určující, zda je zkouška viditelná či nikoliv, id uživatele, který zkoušku vytvořil (created_by), časové razítko vypsání zkoušky (created_at) a Google Calendar API (google_calendar_api) pro evidenci zkoušky v Google kalendáři.

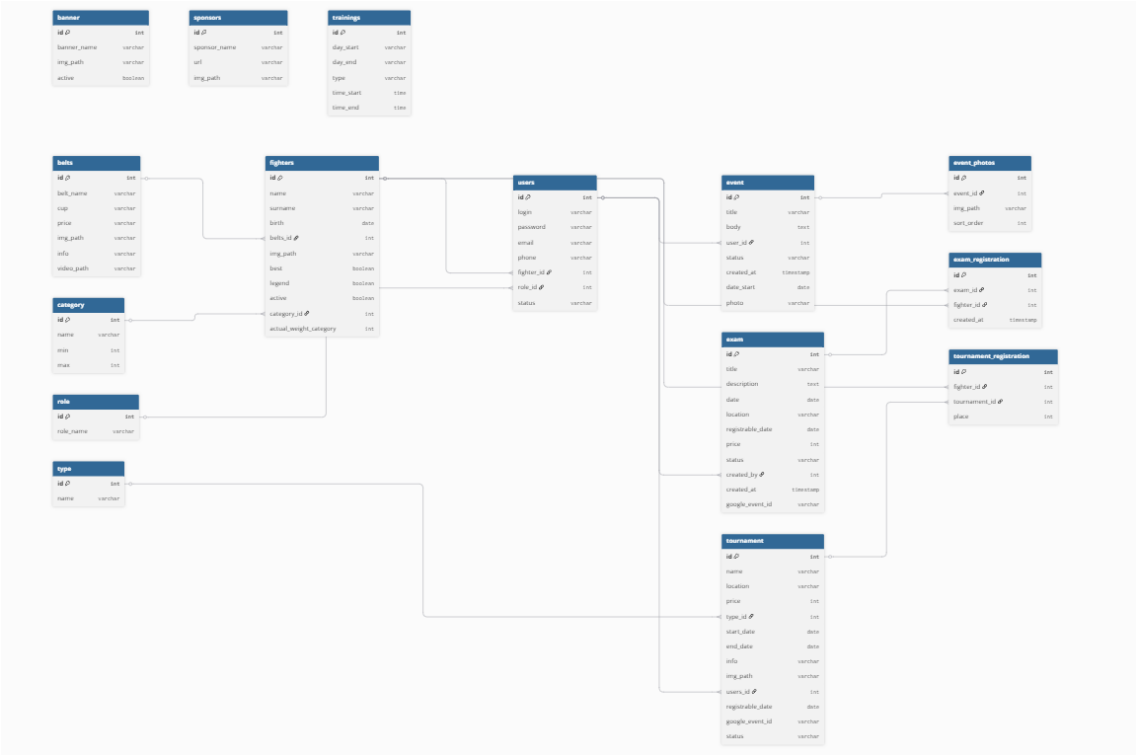
Exam_registration – registrace na zkoušky

Tabulka exam_registration reprezentuje spojovací tabulku vztahu M:N mezi závodníky a zkouškami. Závodník se může přihlásit na více zkoušek a jedna zkouška může mít více zaregistrovaných závodníků. Registrace obsahuje cizí klíč exam_id odkazující na zkoušku které registrace podléhá a fighter_id určující registrovaného závodníka, nachází se zde i časové razítko (created_at) určující, kdy se závodník na zkoušku přihlásil.

Relace mezi tabulkami

Databáze využívá standardní typy relací relačního modelu. Vztahy 1:N jsou nejčastější – například jeden technický stupeň může mít více závodníků, jedna role více uživatelů nebo jedna aktualita více fotografií. Vztah 1:1 je realizován mezi tabulkami users a fighters, kde každý závodník může mít nejvýše jeden uživatelský účet. Vztah M:N je implementován prostřednictvím vazebních tabulek tournament_registration a exam_registration, které propojují závodníky s turnaji a zkouškami.

Referenční integrita je vynucována pomocí cizích klíčů s různými strategiemi při mazání záznamů. Tabulky event_photos, tournament_registration a exam_registraion používají kaskádové mazání, zatímco ostatní vazby používají výchozí omezení zabraňující smazání rodičovského záznamu, pokud na něj existují závislé záznamy.



Obr. 15 Diagram

Zdroj: vlastní

4 Praktická část

Praktická část práce se zabývá realizací webové aplikace pro oddíl TKD Lacek. Cílem bylo vytvořit plnohodnotný webový systém skládající se z veřejně přístupného webu pro návštěvníky a administračního rozhraní pro správu obsahu.

Vývoj aplikace byl rozdělen na dvě vzájemně propojené části. Frontend představuje uživatelské rozhraní běžící v prohlížeči návštěvníka, které zobrazuje informace o oddílu, aktuality, turnaje, závodníky a další obsah. Backend tvoří serverová část aplikace zajišťující zpracování požadavků, komunikaci s databází a autentizaci uživatelů.

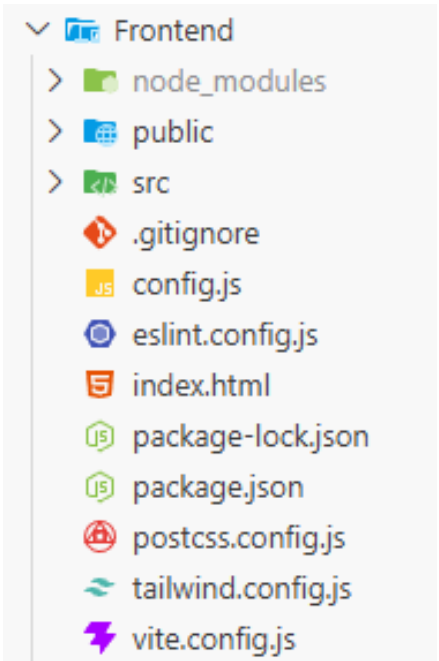
Obě části spolu komunikují prostřednictvím REST API – frontend odesílá HTTP požadavky na backendové endpointy a zpracovává přijatá data ve formátu JSON. Pro vývoj byl zvolen následující technologický stack: na straně frontendu framework ReactJS s knihovnou React Router pro správu navigace a Tailwind CSS pro stylování, na straně backendu Node.js s frameworkem Express.js a relační databáze MySQL pro persistentní ukládání dat.

4.1 Frontend

Část frontendu byla realizována pomocí JavaScriptového frameworku React a stylována CSS frameworkem Tailwind CSS. Frontend má jasnou strukturu a představuje tu část, kterou samotný uživatel při interakci s webem vidí.

4.1.1 Struktura frontendu

Frontend obsahuje několik hlavních složek, ve kterých jsou uloženy jednotlivé části. Struktura odpovídá typickému Reactovému projektu.



Obr. 16: Struktura frontendu
Zdroj: Vlastní

Kořenová složka obsahuje několik velmi důležitých podsložek a souborů, které jsou důležité pro správnou a přehlednou strukturu projektu.

Adresáře:

node_modules/ – adresář, který obsahuje veškeré externí knihovny a závislosti projektu stažené pomocí správce balíčku NPM

public/ – obsahuje statické soubory, které se při práci pouze zkopírují a webpack /vite je nezpracovává, ukládají se zde většinou favikony, loga a soubor index.html který slouží vstupní bod aplikace

src/ – složka obsahující přímo kód psaný programátorem ve které si sám definuje logiku i samotný vzhled aplikace obsahuje například komponenty a jednotlivé stránky

Soubory:

.gitignore – jsou zde definovány složky a soubory, které nemají být zpracovávány verzovacími systémy jako je například Git, typickým příkladem bývá soubor .env ve kterém se nachází citlivé údaje jako jsou API klíče či konfigurace databáze ke které je projekt napojený

index.html – obsahuje HTML šablonu do které je vkládána React aplikace

package-lock.json – generuje se automaticky při každé instalaci a uživatel by do něj neměl vůbec zasahovat

package.json – jsou zde základní informace o projektu a seznam hlavních závislostí

Konfigurační soubory:

eslint.config.js – konfigurační soubor nástroje ESLint, který slouží k statické analýze zdrojového kódu, definuje sadu syntaktických a stylových pravidel, zajišťuje konzistenci kódu

postcss.config.js – konfigurační soubor pro PostCSS, které optimalizuje CSS a umožňuje práci s Tailwind CSS

Tailwind.config.js – konfigurační soubor pro Tailwind CSS

vite.config.js – nachází se zde definice, jak se má projekt spouštět, sestavovat a jaké technologie má podporovat

4.1.2 Layout

V projektu se nachází složka layout, ve které jsou definovány všechny klíčové komponenty, které definují to, jak bude stránka vypadat. Nachází se zde header, footer, sidebar a také container ve kterém se vykreslují všechny komponenty specifické pro určité stránky. Komponenty z layout složky jsou poté importovány do souboru App.jsx a jsou zde použity.

4.1.3 App.jsx

Soubor, který slouží jako hlavní komponenta aplikace a kde se skládá celá struktura. Stará se o to, co uživatel uvidí a jak se aplikace bude chovat jako celek. Jsou zde naimportovány všechny stránky a pomocí logiky *ProtectedRoute* jsou rozděleny na veřejnou a privátní část, která se skrývá za loginem a je dostupná pouze uživatelům, kteří disponují uživatelským účtem.

```
import {
  BrowserRouter as Router,
  Routes,
  Route,
  useLocation,
} from "react-router-dom";
import "./App.css";
import Header from "./layout/Header";
import Footer from "./layout/Footer";
import Container from "./layout/Container";
import Zavodnici from "./Components/Zavodnici";
import Contact from "./Components/Contact";
import Aktuality from "./Components/Aktuality";
import Zkousky from "./Components/Zkousky";
import Home from "./Components/Home";
import Turnaje from "./Components/Turnaje";
import AboutUs from "./Components/AboutUS";
import Detail from "./pages/Turnaje/Detail";
import SideBar from "./layout/Sidebar";
import AddFighter from "./Components/AddFighter";
import FightersAdmin from "./Components/FightersAdmin";
import Login from "./Components/Login";
import Register from "./Components/Register";
import ProtectedRoute from "./pages/Login/ProtectedRoute";
import AdminBanner from "./pages/_Admin/AdminBanner";
import AdminAktuality from "./pages/_Admin/AdminAktuality";
import AdminUsers from "./pages/_Admin/AdminUsers";
import AdminSponsors from "./pages/_Admin/AdminSponsors";
import AdminZavodnici from "./pages/_Admin/AdminZavodnici";
import AdminTurnaje from "./pages/_Admin/AdminTurnaje";
import AdminMe from "./pages/_Admin/AdminMe";
import AktualitaDetail from "./pages/Home/AktualitaDetail";
import TurnajDetail from "./pages/Turnaje/TurnajDetail";
import AdminRequests from "./pages/_Admin/AdminRequests";
import FighterDetail from "./pages/Zavodnici/FighterDetail";
import AdminZkousky from "./pages/_Admin/AdminZkousky";
import Gdpr from "./pages/Login/Gdpr";
import NotFound from "./pages/Login/NotFound";
import ScrollToTop from "./Components/ScrollToTop";
```

Na začátku souboru se nachází import routeru, který je zajištěn pomocí knihovny react-router-dom a zajišťuje navigaci mezi stránkami, dále se zde nachází import globálního CSS souboru, ve kterém jsou definovány styly a použité třídy na stylizaci komponent. Soubor obsahuje také import layout komponent a jednotlivých stránek, které jsou poté obaleny v routách.

Nachází se zde také logika určující, zda se jedná o komponentu, která je skrytá za autentizací či jde o veřejnou část webu dostupnou pro běžného návštěvníka. Jednotlivé komponenty mají stanovené uživatelské role, které k nim mají oprávněný přístup, čímž je zajištěno, že běžní

uživatelé nemají přístup k pokročilé administraci jako je například správa závodníků a správa žádostí.

```
function Layout() {
  const location = useLocation();
  const isAdmin = location.pathname.startsWith("/admin");

  if (isAdmin) {
    return (
      // Chráněné admin routy
      <div className="flex min-h-screen">
        <SideBar />

        <main className="pt-14 lg:pt-0 lg:ml-64 min-h-screen w-full">
          <div className="p-4 sm:p-6 lg:p-8">
            <Routes>
              <Route path="*" element={<NotFound />} />

              <Route
                path="/admin/me"
                element={
                  <ProtectedRoute allowedRoles={["admin", "trainer",
"user"]}>
                    <AdminMe />
                  </ProtectedRoute>
                }
              />
              <Route
                path="/admin/users"
                element={
                  <ProtectedRoute allowedRoles={["admin"]}>
                    <AdminUsers />
                  </ProtectedRoute>
                }
              />
              <Route
                path="/admin/zadosti"
                element={
                  <ProtectedRoute allowedRoles={["admin"]}>
                    <AdminRequests />
                  </ProtectedRoute>
                }
              />
          </div>
        </main>
      </div>
    );
  }
…
```

```
// Veřejné routy
  return (
    <>
      <Header />
      <Container>
        <ScrollToTop />
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/turnaje" element={<Turnaje />} />
          <Route path="/kontakt" element={<Contact />} />
          <Route path="/zavodnici" element={<Zavodnici />} />
          <Route path="/aktuality" element={<Aktuality />} />
          <Route path="/zkousky" element={<Zkousky />} />
          <Route path="/nas-oddil" element={<AboutUs />} />
          <Route path="/detail" element={<Detail />} />
          <Route path="/login" element={<Login />} />
          <Route path="/register" element={<Register />} />
          <Route path="/aktualita/:id" element={<AktualitaDetail />} />
          <Route path="/turnaj/:id" element={<TurnajDetail />} />
          <Route path="/zavodnik/:id" element={<FighterDetail />} />
          <Route path="/gdpr" element={<Gdpr />} />
          <Route path="*" element={<NotFound />} />
        </Routes>
      </Container>
      <Footer />
    </>
  );
```

Veřejné routy nemají přidělené žádné uživatelské role tudíž jsou přístupné všem návštěvníkům webu. Dalším rozdílem oproti chráněným routám je celková struktura. Veřejná část obsahuje header a footer, chráněná část obsahuje pouze sidebar a nedisponuje žádným footerem

4.1.4 Ukázka komponenty Zkousky.jsx

Komponenta zkoušky slouží jako hlavní kontejner, ve kterém jsou naimportovány jednotlivé menší komponenty, které jsou společně propojené a které dohromady tvoří funkční celek pro zobrazení zkoušek uživateli.

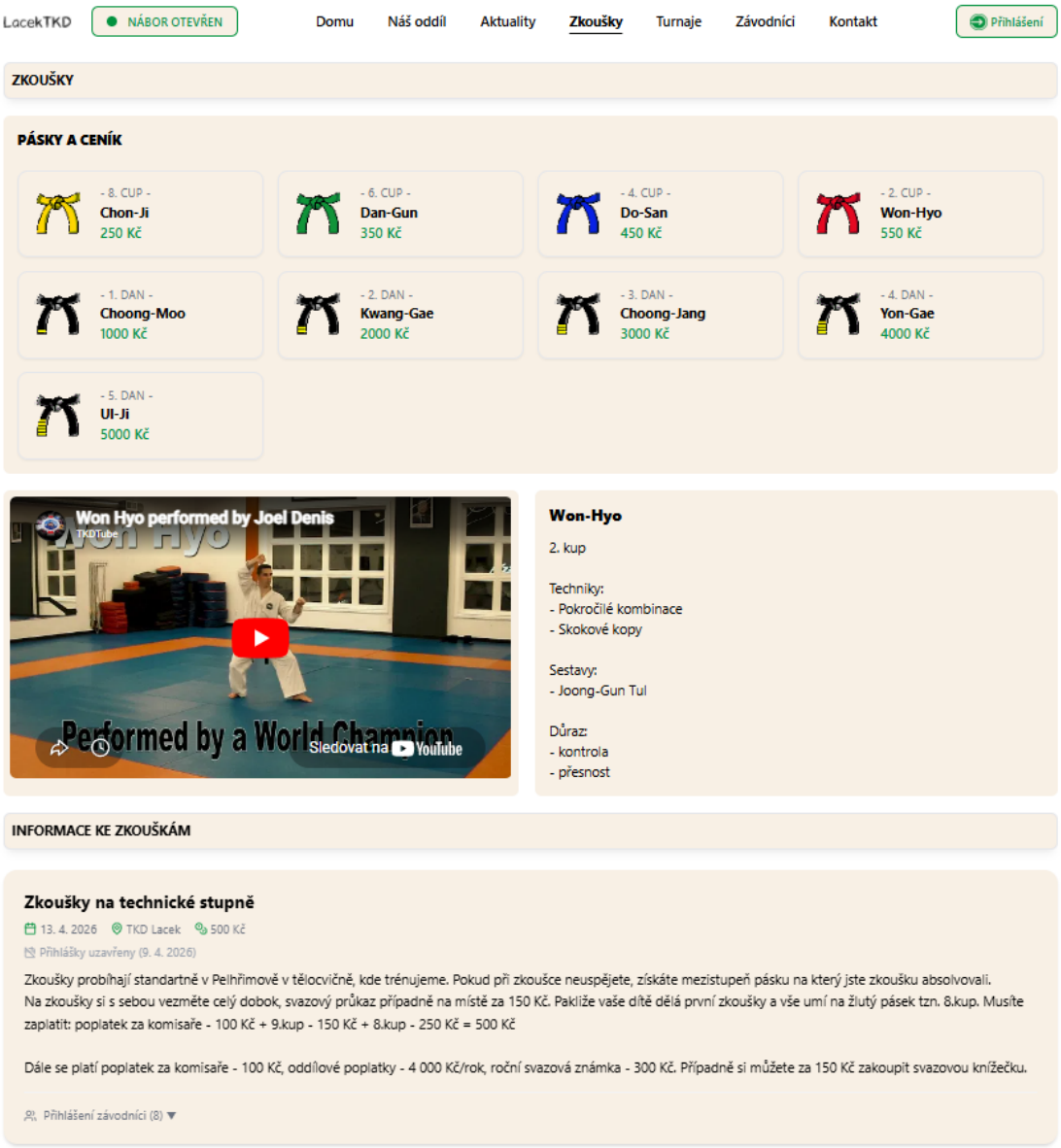
```
import React, { useState } from "react";
import Cenik from "../pages/Zkousky/Cenik";
import Video from "../pages/Zkousky/Video";
import SestavyInfo from "../pages/Zkousky/SestavyInfo";
import Prihlasky from "../pages/Zkousky/Prihlasky";

function Zkousky() {
  const [selectedBelt, setSelectedBelt] = useState(null);

  return (
    <>
      <div className="devider">ZKOUŠKY</div>
```

```
      <Cenik onSelectBelt={setSelectedBelt} />

      {selectedBelt && (
        <div className="flex flex-col lg:flex-row gap-5">
          <Video belt={selectedBelt} />
          <SestavyInfo belt={selectedBelt} />
        </div>
      )}

      <Prihlasky />
    </>
  );
}

export default Zkousky;
```

Stránku tvoří tabulka s přehledem pásků obsahující informace o názvu pásku, třídě danu či cupu, ceně za zkoušku na jednotlivý technický stupeň a obrázek pásku. Pod tabulkou se nachází sekce, která se dynamicky mění na základě vybraného technického stupně. Sekci tvoří komponenty *Video* a *SestavyInfo*. Komponenta *Video* se vykresluje pouze v případě, pokud má daný technický stupeň v databázi uložený odkaz na video. Klíčové pojmy k jednotlivým sestavám vykresluje komponenta *SestavyInfo*.

Dynamické vykreslování je realizováno pomocí useState hooku, který pracuje s informacemi o aktuálně zvoleném technickém stupni. Výchozí hodnota je nastavena na null, což zajišťuje, že při prvotním načtení stránky nebude vybrán žádný technický stupeň a tím pádem nebudou vykreslovány komponenty *Video* a *SestavyInfo*. Pokud však uživatel vybere určitý technický stupeň pásu, tak dojde k dynamickému předání hodnot do jednotlivých komponent a vykreslí se komponenty se specifickými informacemi.

Spodní část stránky tvoří komponenta *Prihlasky*, která zobrazuje informace týkající se vypsaných zkoušek. Obsahuje informace o zkoušce, datum konání a konce přihlášek, místo konání, cenu za účast a seznam přihlášených závodníků. Závodníkům, kteří jsou přihlášeni do systému je zde dostupné i tlačítko na přihlášení a odhlášení ze zkoušky. Zobrazení zkoušek podléhá administrátorům a trenérům, kteří mají v administraci možnost zkoušku vytvořit, upravit, smazat a rozhodovat, zda zkouška bude veřejná či skrytá.



Obr. 17: Zkoušky

Zdroj: vlastní

4.1.5 Responzivita

Responzivita webových stránek je zajištěna pomocí CSS frameworku Tailwind CSS. Hlavním úkolem bylo zajistit, aby se uživatelské rozhraní zobrazovalo správně na všech koncových zařízeních bez ohledu na jejich rozlišení.

Řešení je realizováno metodou mobile-first, což znamená, že rozhraní bylo primárně optimalizováno pro mobilní zařízení a následně upravováno pro větší obrazovky, jako jsou tablety, notebooky a desktopové počítače. Zmíněná metoda zajišťuje dobré zobrazení na všech zařízeních.

Správné rozvržení obsahu je především realizováno pomocí flexboxů a gridů, které umožňují dynamicky přizpůsobovat rozmístění jednotlivých prvků na základě aktuálního rozlišení a v případě potřeby také některé prvky pro menší rozlišení skrýt.

Flexbox je využíván zejména pro zarovnání prvků, jako je například navigační menu, nebo u některých komponent, jako jsou informace týkající se závodníků. Grid je použit pro složitější rozvrhování obsahu, jako jsou například seznamy závodníků nebo aktualit, kdy je potřeba měnit počet zobrazovaných prvků na základě aktuálního rozlišení. V případě většího rozlišení se prvky zobrazují vedle sebe, zatímco se zmenšováním rozlišení se prvky vykreslují pod sebe do sloupce.

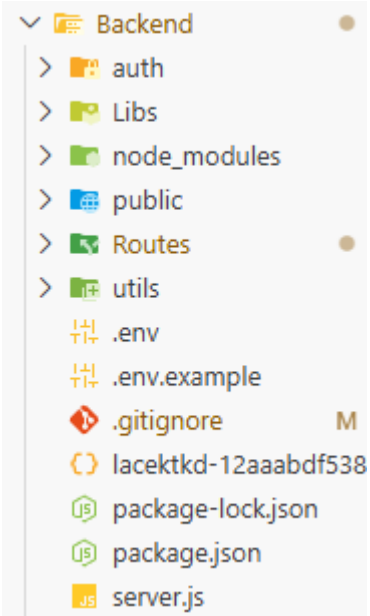
Obr. 18: Ukázka mobilní responzivity
Zdroj: vlastní

4.2 Backend

Backendová část je postavena na JavaScriptovém enginu NodeJS a frameworku Express. Ve složkách se nachází důležité soubory, které se starají o zabezpečení webu, navigaci a o výpis informací z databáze.

4.2.1 Struktura backendu

Backend složka obsahuje několik souborů a složek ve kterých se nachází všechny potřebné routy a konfigurace, které zajišťují správné dodávání dat pro frontend část aplikace.



Obr. 19: Struktura backendu
Zdroj: vlastní

Adresáře:

auth/ – adresář obsahující middleware a JWT logiku, starající se o autorizaci uživatelů

Libs/ – obsahem adresáře je soubor zprostředkovávající propojení s databází

node_modules/ – závislosti týkající se projektu

Routes/ – nachází se zde centrální Router.js soubor a řada podsložek vytvořených podle stránek, ve kterých se nachází jednotlivé routery pro dané stránky

public/ – obsahuje fotografie, které uživatel nahrál, a poté jsou roztřízeny do podsložek podle entity, ke které patří

utils/ – adresář obsahující pomocné funkce, jako je například práce s datem nebo nahrávání souborů

Soubory:

.env – konfigurační soubor, ve kterém jsou uloženy citlivé informace, jako jsou API klíče, JWT SECRET_KEY a údaje týkající se databáze, se kterou aplikace interaguje

.env.example – reprezentuje, jak má vypadat .env soubor

.gitignore – obsahuje složky, soubory nebo koncovky souborů, které nemá verzovací nástroj spravovat

lacektkd-12aaabdf5383.json – konfigurační soubor pro Google kalendář

package-lock.json – generuje se automaticky při každé instalaci a uživatel by do něj neměl vůbec zasahovat

package.json – jsou zde základní informace o projektu a seznam hlavních závislostí

server.js – vstupní bod aplikace

4.2.2 Routy

Důležitou částí týkající se backendu je bezpochyby routování. Jedná se o mechanismus, který definuje, jaká komponenta nebo jaká data se mají aktuálně vykreslit. Vše začíná http požadavkem, který je zpracován v souboru server.js a dále předán centrálnímu souboru Router.js. Zde se podle prefixu URL určí příslušný sub-router, který dále zpracuje samotný požadavek.

Mé řešení obsahuje jeden centrální soubor Router.js ve kterém mám definované routy a cesty k nim.

```
const eventRouter = require("./Events/RouterEvents");
const tournamentRouter = require("./Tournaments/RouterTournaments");
const beltRouter = require("./Belts/RouterBelts");
const fightersRouter = require("./Fighters/RouterFighters");
const newsRouter = require("./News/RouterNews");
const usersRouter = require("./Users/RouterUsers");
const sponsorsRouter = require("./Sponsors/RouterSponsors");
const bannerRouter = require("./Banner/RouterBanner");
const roleRouter = require("./Roles/RouterRoles");
const categoryRouter = require("./Category/RouterCategory");
const trainingRouter = require("./Trainings/RouterTrainings");
const tournamentRegistrationRouter =
require("./TournamentRegistration/RouterTournamentRegistration");
const examRouter = require("./Exams/RouterExams");
```

Zvolil jsem modulární řešení, které je připraveno na další škálovatelnost aplikace a jednotlivé routy jsem umístil do příslušných složek.

Kromě definic routerů je potřeba jednotlivé routery i registrovat, což jsem provedl následovně:

```
router.use("/events", eventRouter);
router.use("/tournaments", tournamentRouter);
router.use("/belts", beltRouter);
router.use("/fighters", fightersRouter);
router.use("/users", usersRouter);
router.use("/sponsors", sponsorsRouter);
router.use("/news", newsRouter);
router.use("/banner", bannerRouter);
router.use("/roles", roleRouter);
router.use("/category", categoryRouter);
router.use("/trainings", trainingRouter);
router.use("/tournamentRegistration", tournamentRegistrationRouter);
router.use("/exams", examRouter);
```

4.2.3 API

Komunikace mezi frontendovou a backendovou částí aplikace je realizována prostřednictvím REST API. Backend je postaven na Node.js s frameworkem Express.js, který obsahuje sadu endpointů organizovaných podle jednotlivých zdrojů. Každý zdroj má vlastní router umístěný v samostatném souboru, přičemž HTTP metody GET, POST, PUT a DELETE odpovídají operacím

CRUD – vytváření, čtení, úpravy a mazání dat. Přenos dat probíhá ve formátu JSON. Přičemž je zde i autentizace, která je řešena pomocí JWT tokenu předávaného v hlavičce každého požadavku, aby k operacím neměl přístup i neoprávněný uživatel.

4.2.4 Ukázka API endpointů

Pro demonstraci volání API endpointů jsem si připravil příklad z RouterEvents.js, kde se nachází různé endpointy pro aktuality. Získávání dat je realizováno pomocí metody active record někdy známé jako raw SQL. Řešení jsem zvolil z důvodu jednoduchosti a plné kontroly nad dotazy.

Pro veřejný výpis všech aktualit se využívá metoda *GET*, která vrací všechny potřebné informace, jako je například název turnaje, náhledová fotografie, popis aktuality, status či datum, kdy byla aktualita vydána. Každá aktualita obsahuje náhledovou fotografii a poté sadu fotografií související s danou aktualitou. Sada fotografií je uložena v separátní tabulce event_photos, která je s tabulkou event propojena na základě identifikátoru event_id. Nachází se zde i chybové hlášky upozorňující na chybu při načítání.

```
router.get("/", (req, res) => {
  db.query(
    `SELECT
        e.id,
        e.title,
        e.body,
        e.status,
        e.photo AS cover_photo,
        e.user_id,
        u.login AS author,
        DATE_FORMAT(e.date_start, '%d.%m.%Y') AS date_start,
        e.date_start AS date_start_raw,
        e.created_at
     FROM event e
     LEFT JOIN users u ON e.user_id = u.id
     WHERE e.status = 'Availible'
     ORDER BY e.id DESC`,
    (err, events) => {
      if (err)
        return res.status(500).json({ error: "Chyba při načítání
aktualit" });
      if (!events.length) return res.json([]);

      db.query(
        "SELECT * FROM event_photos ORDER BY event_id, sort_order ASC",
        (err2, photos) => {
          if (err2)
            return res.status(500).json({ error: "Chyba při načítání
fotek" });

          const result = events.map((event) => ({
            ...event,
```

```
            photos: photos.filter((p) => p.event_id === event.id),
          }));

          res.json(result);
        },
      );
    },
  );
});
```

V souboru mám definovanou i *GET* metodu pro získání 3 nejnovějších aktualit na úvodní stránku.

Endpoint je /api/events/latest a od předchozí ukázky se liší pouze SQL dotazem, který je upraven tak, aby zobrazovaly pouze aktuality, které mají stav nastavený na „Dostupné“, jsou nejnovější a limitovány na 3.

```
 `SELECT
        e.id,
        e.title,
        e.body,
        e.status,
        e.photo AS cover_photo,
        DATE_FORMAT(e.date_start, '%d.%m.%Y') AS date_start
     FROM event e
     WHERE e.status = 'Availible'
     ORDER BY e.id DESC
     LIMIT 3`
```

Zobrazení specifické aktuality je realizováno pomocí id v endpointu - /api/events/id. Pokud neexistuje žádný záznam s uvedeným id dojde k vypsání chybové hlášky informující uživatele o faktu, že aktualita nebyla nalezena.

```
router.get("/:id", (req, res) => {
  const { id } = req.params;

  db.query(
    `SELECT e.*, u.login AS author,
       DATE_FORMAT(e.date_start, '%d.%m.%Y') AS date_start_formatted,
       DATE_FORMAT(e.created_at, '%d.%m.%Y') AS created_at_formatted,
       e.date_start AS date_start_raw
     FROM event e
     LEFT JOIN users u ON e.user_id = u.id
     WHERE e.id = ?`,
    [id],
    (err, results) => {
      if (err)
        return res.status(500).json({ error: "Chyba při načítání
aktuality" });
      if (!results.length)
        return res.status(404).json({ error: "Aktualita nenalezena" });
```

```
      const event = results[0];

      db.query(
        "SELECT * FROM event_photos WHERE event_id = ? ORDER BY
sort_order ASC",
        [id],
        (err2, photos) => {
          if (err2)
            return res.status(500).json({ error: "Chyba při načítání
fotek" });
          res.json({ ...event, photos });
        },
      );
    },
  );
});
```

Vytváření aktualit je realizováno metodou *POST* na endpointu /api/events a mají k němu přístup pouze trenéři a administrátoři. O autentizaci se zde stará JWT token, který je přidělen každému přihlášenému uživateli. Autentizace zde byla přidána z důvodu bezpečnosti, aby nevyžádané osoby nemohly přidávat příspěvky.

```
router.post("/", verifyToken, isAdminOrTrainer, uploadFields, (req, res)
=> {
  const { title, body, status, date_start, user_id } = req.body;

  if (!title) return res.status(400).json({ error: "Chybí název
aktuality" });

  const coverFile = req.files?.cover?.[0];
  const coverPath = coverFile
    ? "/uploads/events/" + coverFile.filename
    : req.body.cover_url || null;

  const photoFiles = req.files?.photos || [];

  db.query(
    `INSERT INTO event (title, body, status, date_start, user_id, photo)
VALUES (?, ?, ?, ?, ?, ?)`,
    [
      title,
      body || null,
      status || "Availible",
      date_start || null,
      req.user.id,
      coverPath,
    ],
    (err, result) => {
      if (err)
```

```
        return res.status(500).json({ error: "Chyba při ukládání
aktuality" });

      const eventId = result.insertId;

      if (photoFiles.length > 0) {
        const photoValues = photoFiles.map((file, index) => [
          eventId,
          "/uploads/events/" + file.filename,
          index,
        ]);

        db.query(
          "INSERT INTO event_photos (event_id, img_path, sort_order)
VALUES ?",
          [photoValues],
          (err2) => {
            if (err2) console.error("Chyba při ukládání fotek galerie:",
err2);
          },
        );
      }

      res.status(201).json({
        success: true,
        message: "Aktualita byla úspěšně přidána",
        id: eventId,
      });
    },
  );
});
```

Pro případ úpravy aktuality se zde nachází metoda *PUT*, která má povinný parametr a tím je id, které identifikuje, která aktualita se má upravit. Endpoint se nachází na adrese /api/events/id a taktéž je chráněný pomocí JWT. K endpointu mají přístup pouze administrátoři a trenéři.

```
router.put("/:id", verifyToken, isAdminOrTrainer, uploadFields, (req,
res) => {
  const { id } = req.params;
  const { title, body, status, date_start } = req.body;

  if (!title) return res.status(400).json({ error: "Chybí název
aktuality" });

  const coverFile = req.files?.cover?.[0];
  const photoFiles = req.files?.photos || [];

  const updateQuery = coverFile
    ? `UPDATE event SET title=?, body=?, status=?, date_start=?, photo=?
WHERE id=?`
```

```
    : `UPDATE event SET title=?, body=?, status=?, date_start=? WHERE
id=?`;

  const updateParams = coverFile
    ? [
        title,
        body || null,
        status || "Availible",
        date_start || null,
        "/uploads/events/" + coverFile.filename,
        id,
      ]
    : [title, body || null, status || "Availible", date_start || null,
id];

  db.query(updateQuery, updateParams, (err, result) => {
    if (err)
      return res.status(500).json({ error: "Chyba při aktualizaci
aktuality" });
    if (result.affectedRows === 0)
      return res.status(404).json({ error: "Aktualita nenalezena" });

    if (photoFiles.length > 0) {
      db.query(
        "SELECT COALESCE(MAX(sort_order), -1) AS maxOrder FROM
event_photos WHERE event_id = ?",
        [id],
        (err2, rows) => {
          if (err2) return;
          const startOrder = rows[0].maxOrder + 1;
          const photoValues = photoFiles.map((file, index) => [
            id,
            "/uploads/events/" + file.filename,
            startOrder + index,
          ]);
          db.query(
            "INSERT INTO event_photos (event_id, img_path, sort_order)
VALUES ?",
            [photoValues],
          );
        },
      );
    }

    res.json({ success: true, message: "Aktualita byla úspěšně upravena"
});
  });
});
```

Mazání aktualit zajišťuje metoda *DELETE* s endpointem /api/events/id, která je skrytá za verifyToken – zajišťuje, že požadavek odeslal pouze přihlášený uživatel s platným JWT a oprávněnou rolí. Při mazaní aktuality se automaticky z tabulky event_photos vymažou veškeré fotografie, které s aktualitou souvisely, a to díky nastavenému CASCASDE DELETE.

```
router.delete("/:id", verifyToken, isAdminOrTrainer, (req, res) => {
  const { id } = req.params;

  db.query("SELECT photo FROM event WHERE id = ?", [id], (err, eventRows)
=> {
    db.query(
      "SELECT img_path FROM event_photos WHERE event_id = ?",
      [id],
      (err2, photos) => {
        db.query("DELETE FROM event WHERE id = ?", [id], (err3, result)
=> {
          if (err3)
            return res
              .status(500)
              .json({ error: "Chyba při mazání aktuality" });
          if (result.affectedRows === 0)
            return res.status(404).json({ error: "Aktualita nenalezena"
});

          if (eventRows?.[0]?.photo) {
            const coverPath = getFilePath(eventRows[0].photo);
            if (fs.existsSync(coverPath)) fs.unlinkSync(coverPath);
          }

          if (photos) {
            photos.forEach((photo) => {
              const fullPath = getFilePath(photo.img_path);
              if (fs.existsSync(fullPath)) fs.unlinkSync(fullPath);
            });
          }

          res.json({ success: true, message: "Aktualita byla smazána" });
        });
      },
    );
  });
});
```

4.2.5 Zabezpečení

Bezpečnost webové aplikace je klíčovou součástí každého moderního systému. V rámci mé práce bylo implementováno několik vrstev zabezpečení pokrývajících autentizaci uživatelů, ochranu API endpointů a bezpečné ukládání přístupových údajů.

4.2.6 Autentizace pomocí JWT

Pro autentizaci byl zvolen standard JSON Web Token (JWT), který představuje bez stavový mechanismus ověřování identity uživatele. Na rozdíl od tradičních session-based přístupů JWT nevyžaduje ukládání stavu na straně serveru – veškeré potřebné informace jsou zakódovány přímo v tokenu.

Token se skládá ze tří částí oddělených tečkou: hlavičky (header) obsahující typ tokenu a použitý algoritmus, těla (payload) s uživatelskými daty a podpisu (signature) zajišťujícího integritu tokenu. Po úspěšném přihlášení server vygeneruje token s payloadem obsahujícím identifikátor uživatele, jeho přihlašovací jméno a roli. Token má nastavenou platnost 7 dní, po jejímž uplynutí je uživatel automaticky odhlášen.

```
const token = jwt.sign(
        { id: user.id, login: user.login, role: user.role_name },
        SECRET_KEY,
        { expiresIn: "7d" },
      );
```

SECRET_KEY – tajný klíč použitý k podpisu tokenu je uložen v konfiguračním souboru .env a není součástí zdrojového kódu, čímž je zajištěna jeho bezpečnost. Samotný soubor .env je taktéž zahrnut v souboru .gitignore, čímž nedochází k jeho zpracování pomocí verzovacích nástrojů.

4.2.7 Hashování hesel

Uživatelská hesla se do databáze neukládají v čitelné podobně, ale při registraci se heslo do databáze odesílá zaheshované pomocí ByCript knihovny. Faktor hashovaní je nastaven na náročnost 10, faktor určuje výpočetní složitost procesu a znesnadňuje případné útoky.

```
const hashedPassword = await bcrypt.hash(password, 10);
```

Při přihlášení je zadané heslo porovnáno s uloženým hashem pomocí funkce bcrypt.compare, která vrátí true pouze pokud hesla odpovídají. Díky porovnávání není nikdy nutné hash dešifrovat, což je záměrná vlastnost jednosměrného hashování.

Při registraci je navíc provedena kontrola duplicity emailové adresy – pokud již účet s daným emailem existuje, server vrátí chybu s HTTP status kódem 409 (Conflict) a registrace není dokončena.

4.2.8 Frontendové zpracování autentizace

Na straně frontendu je autentizace řešena pomocí centrálního modulu utils/auth.js obsahujícího službu authService. Po úspěšném přihlášení je JWT token uložen do localStorage prohlížeče a přikládán ke každému chráněnému požadavku v authorization hlavičce. Pro dynamické překreslení uživatelského rozhraní po přihlášení a odhlášení je využit mechanismus vlastních DOM událostí (authChange), který zajišťuje aktualizaci stavu bez nutnosti obnovení stránky.

4.2.9 CORS

Server má nakonfigurovanou politiku CORS (Cross-Origin Resource Sharing) povolenou pouze pro doménu aplikace, na které běží frontendová aplikace. Čímž je zajištěno že API nepřijímá požadavky z neznámých nebo neoprávněných domén.

5 Implementace řešení

Výsledné řešení je rozdělené na frontendovou a backendovou část, které jsou provozovány odděleně, avšak spolu vzájemně komunikují.

Frontendová část aplikace je nejdříve sestavena, přičemž dojde k optimalizaci zdrojových kódu a jejich převedení do statických souborů (HTML, CSS, JavaScript). Soubory jsou poté odesílány pomocí reverzní proxy, která zajišťuje jejich dostupnost pro koncové uživatele.

Backendová část je implementována v prostředí NodeJS a je spuštěna pomocí příkazu npm start. Backend je provozován jako služba, což zajišťuje nepřetržitý běh a dochází k automatickému spuštění při startu serveru. Zmíněné řešení zajišťuje vyšší stabilitu aplikace.

Reverzní proxy slouží jako prostředník mezi klientem a serverem. Slouží k přesměrování požadavků na frontend část aplikace a zároveň směruje API požadavky na backend, který běží na jiném portu. Zmíněné řešení zajišťuje oddělení jednotlivých částí aplikace a zároveň zvyšuje bezpečnost a efektivitu komunikace mezi nimi.

Výše popsaná architektura umožňuje lepší škálovatelnost, jednodušší správu a vyšší bezpečnost celého řešení.

Závěr

Cílem bakalářské práce bylo kompletně restrukturalizovat webové stránky sportovního oddílu Taekwondo Lacek. Původní webové stránky měly zastaralý design, nebyly responzivní a chybělo jim několik klíčových funkcionalit, které jsem při tvorbě nového webu zohlednil a implementoval. Jednou z klíčových funkcionalit je přihlašování závodníků na turnaje či zkoušky což majiteli oddílu šetří spoustu času a zvyšuje efektivitu, protože nyní již nemusí složitě vyhledávat a sepisovat závodníky, kteří mají zájem o účast na oddílových akcích.

Nové řešení splňuje veškeré stanovené požadavky. Výsledkem jsou webové stránky, které odpovídají současným požadavkům na design a technické standardy jako je například bezpečnost dat či responzivita, která je pro většinu uživatelů klíčová.

Tvorba bakalářské práce mi přinesla cenné zkušenosti nejen v oblasti programování, ale i v oblasti komunikace se zákazníkem a vyjednávání. Během vývoje jsem se seznámil s principy moderního designu a bezpečností webových stránek, získané zkušenosti považuji za velmi cenné a jsem si jistý, že mi pomohou v mém profesním růstu. Vytvořil jsem funkční webovou stránku, která splňuje veškeré požadavky, má praktickou hodnotu a přispívá k efektivnější organizaci a propagaci oddílu Taekwondo Lacek.

Do budoucna přemýšlím, že bych web rozšířil o možnost objednat si týmové oblečení, zažádat o svazovou známku či svazovou knížečku. Dále bych zde i rád přidal možnost posílání zpráv mezi závodníky webu.

Seznam použité literatury

About npm. Online. 2024. Dostupné z: https://docs.npmjs.com/about-npm. [cit. 2024-12-29].

Berners-Lee, T. The World Wide Web. Online. 1991. Dostupné z: https://www.w3.org/History.html. [cit. 2024-12-22].

Built-in React Hooks. Online. 2024. Dostupné z: https://react.dev/reference/react/hooks. [cit. 2024-12-29].

Co je GDPR. Online. 2016. Dostupné z: https://mv.gov.cz/gdpr/clanek/co-je-gdpr.aspx. [cit. 2024-12-29].

Co je Taekwondo. Online. HTTPS://WWW.WORLDTAEKWONDO.CZ/CZ/O-TAEKWONDO-CO-JE-TO-TAEKWONDO.HTML. Český svaz Taekwondo. 2024. Dostupné z: https://www.worldtaekwondo.cz/cz/o-taekwondo-co-je-to-taekwondo.html. [cit. 2024-12-19].

Figma. Online. 2016. Dostupné z: https://www.figma.com/about/. [cit. 2024-12-22].

Figma: co to je a jaké jsou jeho základní funkce a principy? Online. *DrawPlanet*. 2023, roč. 2023, č. 12, s. 1. Dostupné z: https://www.drawplanet.cz/co-je-tedy-figma/. [cit. 2024-12-22].

Chaffey, D., & Ellis-Chadwick, F. Digital Marketing. Online. 2022. Dostupné z: https://www.digitalmarketing.com/. [cit. 2024-12-22].

Marcotte, E. Responsive Web Design. Online. 2010. Dostupné z: https://alistapart.com/article/responsive-web-design/. [cit. 2024-12-22].

MySQL. Online. 2024. Dostupné z: https://www.mysql.com/. [cit. 2024-12-22].

Nařízení - 2016/679 - EN – GDPR – EUR-Lex. Online. 2016. Dostupné z: https://eur-lex.europa.eu/legal-content/CS/TXT/?uri=CELEX:32016R0679. [cit. 2024-12-29].

NodeJS. Online. 2009. Dostupné z: https://nodejs.org/en/about. [cit. 2024-12-22].

O'Reilly, T. What Is Web 2.0. Online. 2005. Dostupné z: https://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html. [cit. 2024-12-22].

OWASP. OWASP Top Ten. Online. 2024. Dostupné z: https://owasp.org/www-project-top-ten/. [cit. 2024-12-22].

Práva subjektu údajů. Online. 2016. Dostupné z: https://mv.gov.cz/gdpr/clanek/prava-subjektu-udaju.aspx. [cit. 2024-12-29].

React. Online. 2013. Dostupné z: https://react.dev/reference/react. [cit. 2024-12-22].

ReactJS Virtual DOM. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/reactjs-virtual-dom/. [cit. 2024-12-29].

RFC 5246. The Transport Layer Security (TLS) Protocol. Online. 2008. Dostupné z: https://datatracker.ietf.org/doc/html/rfc5246. [cit. 2024-12-22].

Smith, J., Doe, A., & Brown, B. Artificial Intelligence in web Development. Online. 2023. Dostupné z: https://www.aidevelopment.com/. [cit. 2024-12-22].

TailwindCSS. Online. 2024. Dostupné z: https://Tailwindcss.com/. [cit. 2024-11-24].

VisualStudioCode. Online. 2024. Dostupné z: https://code.visualstudio.com/. [cit. 2024-12-29].

W3C. HTML and Web Standards. Online. 2024. Dostupné z: https://www.w3.org/standards/webdesign/. [cit. 2024-12-22].