SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE
dokumentace

Ultimate Search Engine

David Stoček

Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2022/2023

Poděkování

Rád bych chtěl poděkovat především Filipovi Peterkovi za nápad tohoto projektu a za pomoc, kterou nám při jeho plnění poskytoval. Dále děkuji panu Ing. Petru Grussmannovi za zprovoznění a hostování našeho programu na jeho serveru.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2016

podpis autora práce

ANOTACE

Práce se zaměřuje na proces a poté i konečný výsledek vývoje moderní *front-end* webové aplikace, vytvořené za pomocí *Next.js frameworku* a stylované s použitím *SCSS* technologie, sloužící jako vizualizace a uživatelské rozhraní pro studentský projekt *USE*. Práce čtenáři podrobně vysvětluje tvorbu a funkčnost jednotlivých *Next.js* stránek a komponentů s využitím moderních postupů na vývoj webových aplikací.

Klíčová slova

Vyhledávač, Front-end, React, web, programování

OBSAH

ÚVOD

Společně se dvěma mými kolegy jsme se rozhodli udělat plnohodnotný vyhledávač jako náš projekt. Chtěli jsme všichni vědět jaké to je pracovat v týmu na něčem takového obrovského rozměru.

Mým cílem jako člověk, co si vybral pracovat na *front-endu* aplikace, bylo hlavně vytvořit uživatelskou aplikaci od našeho *search enginu*, tak aby byl srozumitelný pro obyčejného člověka, ale zároveň bohatý na funkcích. Vytvořen za pomocí různých *javascriptových* knihoven, na lehčí vývoj komplexních aplikací, a stylových preprocesorů pro větší přehlednost.

Výsledný produkt by tedy měl správně spouštět a zobrazovat webovou aplikaci dostupnou a responzivní na všech operačních systémech a prohlížečích. Stránky by měly obsahovat vstupní pole na vyhledávání uživatelských dotazů, které budou zpracovány ostatními moduly *USE* v *back-endu* aplikace, poté navrátit relevantní výsledky na zobrazení uživateli zpět na stránku.

V následujících kapitolách popíšu podrobné informace o mém procesu vytváření takové stránky, technologie a způsoby co jsem při tom použil a ukázku výsledného produktu.

1 VYUŽITÉ TECHNOLOGIE

1.1 React

React je open-source *front-end* webový *framework* psaný v jazyce *JavaScript* pro vývoj dynamického uživatelského rozhraní. Jeho nejlepší použití je při budování webů s často se měnícími daty, jelikož je založen na stavbě z menších *UI* elementů, zvaných komponentů, které se v reálném čase dokážou aktualizovat na nejnovější data bez potřeby obnovení celé stránky. Což pro můj projekt s neustále se měnící mi komplexními výsledky byla ta nejlepší volba.

1.1.1 Next.js

Framework postavený na knihovně *React,* obohacený o funkce, kterými *React* samotný nedisponuje, například nástroji pro ovládání *back-endu* stránek, jako je hlavně *routování*. React je také omezený na vykreslování na straně klienta, kde Next.js má možnost tzv. *server side rendering* a generování statických stránek, ty v určitých případech dokážou snížit rychlost načítání.

1.2 TypeScript

TypeScript je syntaktický superset jazyka *JavaScript*, který přidává statické typy. Celý *front-end* je v něm napsaný, zvolil jsem si ho především kvůli definovaní typů, ty mi pomohly třeba při debuggování a pro větší přehlednost mého kódu. *TypeScript*, avšak potřebuje kompilaci zdrojového kódu na *JavaScript*, což pro některé může být problém.

1.3 Node.js

Pro běh aplikace na straně serveru jsem využil právě *Node.js,* ten je asynchronní a stavěn pro budování síťových aplikací. Stará se o komunikace mezi klientem a serverem.

1.4 SCCS

SCSS neboli *Sass* je stylovací jazyk, postavený na jazyku a plně kompatibilní s jazykem *CSS*. Pro mě to byla jasná volba díky jeho vylepšeným vlastnostem, kterými ostatní jazyky nedisponovali, jako je například vnořování, proměnné a především dědičnost.

1.5 Docker

Docker je *open-source* platforma, umožňuje mi oddělit aplikaci do izolovaných spustitelných kontejnerů. Výhodou je multiplatformní podpora a možnost spuštění mnoho takových kontejnerů současně.

2 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

2.1 Začátky projektu

Jako první věc, o kterou jsem se musel postarat bylo správné nainstalování *Node.js* na pracovní počítač. Jakmile jsem měl všechno připravené našel jsem si na oficiálních stránkách *Next.js* jak takový nový projekt založit. Podle jejich postupů a příkazů se základ aplikace postaví zcela sám, a tak se mi aplikace podařila zprovoznit okamžitě, bez jakýchkoliv problémů.

Chtěl jsem, avšak využít vylepšených funkcí, které *TypeScript* a *SCSS* nabízejí, jejich podpora ale do *Next.js* není přímo zabudovaná, dá se, zato snadno s pomocí jejich dokumentace nastavit.

A jakmile bylo všechno připraveno mohl jsem se pustit do designování a následného vytváření mého projektu.

2.2 Návrh projektu

Mým cílem bylo navrhnout stránku v poměrně minimalistickém stylu, připadalo mi to nejvhodnější, jelikož účel každého vyhledávače je, aby byl přístupný pro všechny, nehodí se tedy ho zdobit všelijakými zbytečnými barevnými efekty.

Jako hlavní barevný režim jsem použil režim světlý, pokud je váš operační systém nastaven na světlý a dynamicky i tmavý režim, pokud je systém nastavený na tmavý. Toto můžeme udělat kouskem kódu v *JavaScriptu* (v mém případě *TypeScriptu*), zjišťující systémové nastavení.

```
if(window.matchMedia("(prefers-color-scheme: dark)")){
      // Systémový barevný režim je tmavý
} else {
      // Systémový barevný režim je světlý
}
```

Příklad kódu 1 – Barevný režim

2.2.1 Domovská stránka

Domovská stránka není nic speciálního, perfektní příklad minimalistického designu, neobsahuje nic víc než jedno vstupní pole, určené pro zadávání uživatelských dotazů pro tento vyhledavač a naše logo postaveno nad ním.

Pole i obrázek je přesně vycentrované doprostřed obrazovky. Toho efektu jsem jednoduše dosáhl použitím *CSS* parametru *display: flex;*. Ten, avšak elementy ještě nevycentruje, k vycentrování všech dceřiných prvku horizontálně musíme ještě dodat *justify-content: center;* a vertikálně *align-items: center;*

2.2.2 Stránka s výsledky

Po zadání libovolného dotazu do vyhledávacího pole se vám načte právě stránka s relevantními výsledky. Od domovské stránky se pole i logo zmenšilo a přesunulo nahoru obrazovky do záhlaví, to zůstává pořád viditelné, i když se posouváme na stránce. Přidal jsem také klikatelný obrázek ozubeného kola, ten nás přesune na stránku nastavení.

V hlavní sekci rozložení stránky jsem udělal místo pro vykreslení všech výsledků, ty jsou rozděleny do 2 řad (pomocí *display: grid;*), postavené vedle sebe, pro ušetření místa vertikálně. Na menších monitorech se řady postaví nad sebe pro lepší viditelnost. Levá část řady obsahuje všechny obyčejné výsledky, na té pravé všechny výsledky, které jsou o něco více specializované na určité typy výsledků, jako například výsledky z *wikipedia.org* nebo *goodreads.com.* Tyto výsledky často disponují přídavnými daty relevantní pouze pro tyto stránky samotné.

Na zápatí stánky se vyskytují očíslované odkazy na další stránky, mezi kterými můžeme proklikávat.

2.2.3 Stránka nastavení

Stránka nastavení je rozdělena na dvě sekce, sekci nastavení a sekci „o nás“. Do sekce nastavení se mi zatím, ale jenom podařilo vymyslet mřížku s kupou různých měnitelných barevných motivů. Pod nimi se nachází „o nás“ sekce kde je krátký text vysvětlující vzniknutí tohoto projektu, zakončený malými avatary všech vývojářů, co na tomto projektu pracovali.

2.3 Funkčnost modulů

K realizaci jakýchkoliv nápadů, co jsem měl se budu prvně potřebovat seznámit se strukturou souborů.

V *Next.js* je již vytvořen *routovací* systém, který zajišťuje provoz dostupných cest mojí webové stránky. Pro přidání nové automaticky *routované* stránky je zapotřebí pouze vytvořit v kořenovém adresáři složku *pages*, kde poté musíme vytvořit indexovou stránku *index.tsx*, bez ní by se nedalo na server dovolat. Nyní jsem již mohl vytvářet libovolné stránky, ale nutno je mít pod stejnou příponou *.tsx* a na webovém prohlížeči je mohu nahlédnout pod stejným jménem.

Každá část mého webu je díky *reaktu* postavena z malých stavebních bloků neboli komponentů, které se renderují nezávisle na ostatních. Pro vytvoření takového modulu jsem si v kořenovém adresáři vytvořil složku *components*, do které všechny budoucí komponenty budu ukládat. Všechny komponenty také mohou mít svoje vlastní styly, které nebudou nijak zasahovat do ostatních stylů komponentů jiných pod stejným názvem.

2.3.1 Vstupní pole

Jako první komponent, se kterým jsem se musel poprat bylo právně vstupní pole, v adresáři *components* jsem jej vytvořil pod názvem *form.tsx*. Začátek byl snadný, stačilo v *HTML* napsat značku formuláře a do něj vložit značku vstupu, do kterého bude dotaz zadáván, a tlačítka, ve formě obrázku lupy, který bude sloužit jako příkaz na odeslání dotazu.

Obrázek 1 – Ukázka vyhledávacího pole

Avšak *HTML* samo o sobě nic neudělá, musel jsem vymyslet, jak vzít uživatelský dotaz a pomocí něj se přesunout na stránku s výsledky. V *Next.js* funkci *useRouter* je možnost se na danou adresu s daným řetězem dotazu, které ve své podstatě jsou informace pro *search manager*, přesunout po spuštění funkce. To stačilo implementovat ať se spustí pokaždé co uživatel odešle dotaz (ve formuláři parametrem *onSubmit*), a předá přitom hodnotu ve vstupním políčku.

Problémem na stránce vyhledávání poté bylo že vyhledávaný dotaz se díky obnovení stránky nepřenesl. Takže hodnotu ve vstupním poli jsem začal neustále ukládat do proměnné

pomocí zabudované *React* funkce *useState,* jejíž hodnota je ponechána i po takzvaném *„měkkém obnovení“*.

2.3.2 Výsledky

Po odeslání dotazu z formuláře, jsme ihned přesunuti na novou stránku. Ta je ze začátku prázdná, ale ihned co se načte, spustí se kus kódu, který asynchronně vyžádá od search manageru, pomocí funkce *fetch* relevantní výsledky pro uživatelský dotaz.

```
{
        type: "default",
        url: "https://www.sspu-opava.cz/cs/",
        title: "SŠPU Opava - úvodní stránka",
        description: "Domovská stránka Střední školy průmyslové a umělecké v Opavě",
        status: "Crawled"
},
```

Příklad kódu 2 – Data pro výsledek

Search manager poté odpovídá odesláním *JSON* dat, ty obsahují data pro všechny výsledky, které dokázal najít. Data poté musím dále zpracovat.

Zpracovávám například příliš dlouhé názvy z kolonky *title*, které zkrátím na maximální povolenou délku a zakončím třemi po sobě jdoucími tečkami. Zpracovávám data odkazů, aby byly rozporcované na jednotlivé pod-odkazy oddělené znaménkem „větší než“. Nebo také zpracovávám různé typy výsledků, ty dělím na běžné výsledky a na speciální, které se potom ještě větví na další. Tím vlastně poznám, na které straně výsledků budou zobrazeny.

Běžné výsledky

Běžné výsledky jsou všechny výsledky bez jakýchkoliv speciálních dat navíc. Vždy se vyobrazují na levé straně obrazovky, v případě mobilního rozložení, pod speciálními výsledky. Skládají se z parametrů: nadpis, popisek, odkaz a typ nastaven na hodnotu *„default“*.

Speciální výsledky

Mezi speciální výsledky řadíme výsledky s parametrem typ s hodnotou jinou než *„default“*, může se jednat například o výsledek z *Wikipedie*, jehož parametry jsou obohaceny ještě o pole obsahu článku. Nebo výsledek z webu *Goodreads*, ten má navíc data o hodnocení knihy a žánry knihy. Speciální výsledky se čistě kvůli vizuálnímu rozlišování vyobrazují na pravé straně webové stránky.

2.3.3 Našeptávač

Komponenta návrhů slouží, jak už jméno napovídá, nabízet uživatelům různé návrhy nejrelevantnějších otázek nebo pouze dokončit rozepsané slovo. Komponenta je vložena do stejného souboru s vyhledávacím políčkem, jelikož spolu blízky souvisejí a neustále na sebe reagují. V podstatě pokaždé co uživatel zadá jakékoliv znaménko je pomocí funkce *fetch* odeslán dotaz na *suggester*, ten na zpátky vrací pole výsledků, které následně zobrazím ve vyskakovacím okýnku pod polem s vyhledáváním.

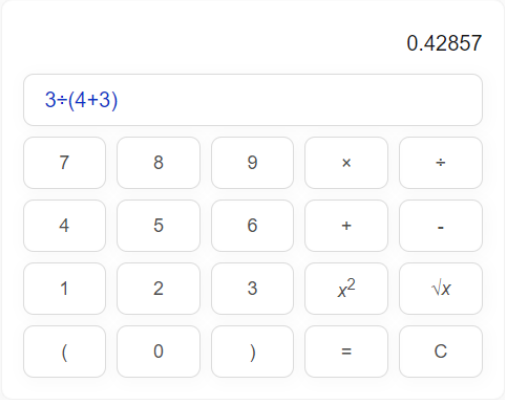
2.4 Hinty

Pod tím si můžeme představit samostatné moduly, které na specifický uživatelský dotaz zhodnotí, zda uživatel například potřebuje převést měnu, vypočítat matematický příklad, nebo rovnou zobrazit funkční kalkulačku a pošle vhodný hint zpátky webové aplikaci, která jej vyobrazí.

Současně se, ale ještě nepodařilo *hint server* zprovoznit, tudíž se jedná jednom o koncept do budoucna.

2.4.1 Kalkulačka

Jediný *hint*, na kterém jsem již pracoval je kalkulačka, ta by se měla objevit na pravé straně webu pokaždé co uživatel napíše do vyhledavače dotaz, který *hint server* vyhodnotí jako matematický příklad, nebo si kalkulačku přímo vyhledá.

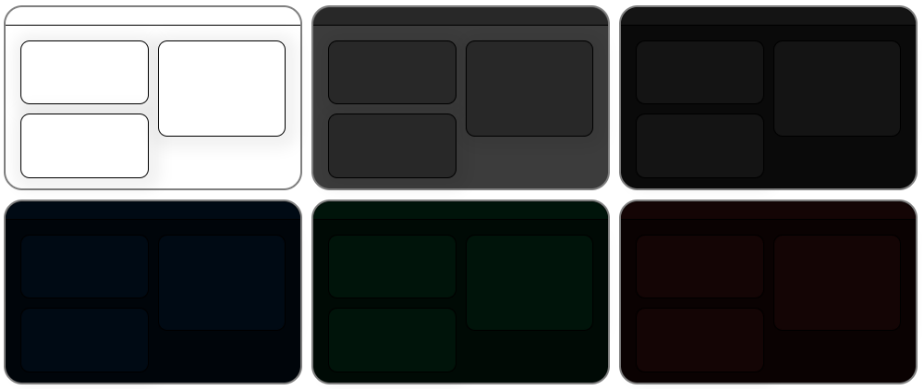


Obrázek 2 – Ukázka kalkulačky

2.5 Barevné témata

Mezi moje dlouhodobé cíle bylo uživateli umožnit si webový vyhledávač přizpůsobit podle sebe, tak jak vám a vašim očím nejvíce vyhovuje. Proto jsem na tom začal pracovat. Nápad byl přidat, na stránku "o nás", sekci s nastavením, kde všechny tyto přizpůsobení budou.

Volitelné možnost jsou uspořádány do responzivní mřížky po jedné až třech řadách, kde se všechny předefinované možnosti na barevné režimy zobrazí. Po kliknutí na jakýkoliv z nich se barvy stránky ihned změní.



Obrázek 3 – Ukázka výběru stylů

Tohoto efektu jsem dosáhl s využitím *SCSS* proměnných, které jsem uložil do nového souboru pod jménem *design_tokens.scss*, ty se při stisknutí možnosti změní na dané hodnoty.

Obrázek 4 – Ukázka různých stylů

Uživatelský výběr se také ukládá do místního úložiště webové stránky, takže se uživatel nemusí starat o nastavování znovu pokaždé co si stránku resetuje.

2.6 Docker

Dockerizace Next.js aplikace mi umožní spustit aplikaci na jakémkoliv stroji s nainstalovaným *dockerem*, aniž bych musel instalovat všechny požadované závislosti a nastavení

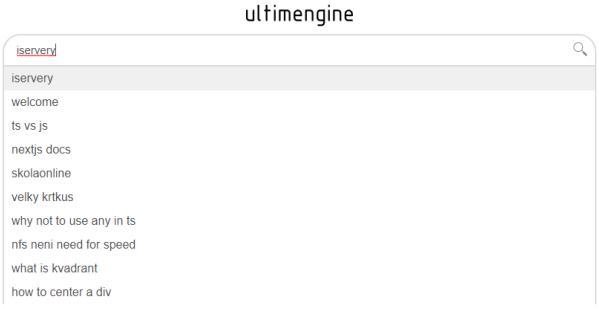
vývojového prostředí. A zároveň umožní celý projekt uzavřít do jednoho kompletního *Docker compose*

Proces *dokerizace* neprobíhal nijak složitě, stačilo mi vytvořit *Dockerfile* v kořenovém adresáři a v něm dopsat požadovanou sekvenci kódu, kterou jsem snadno vyhledal na oficiálních stránkách *Next.js* a ten z velké části fungoval.

3 VÝSLEDKY ŘEŠENÍ

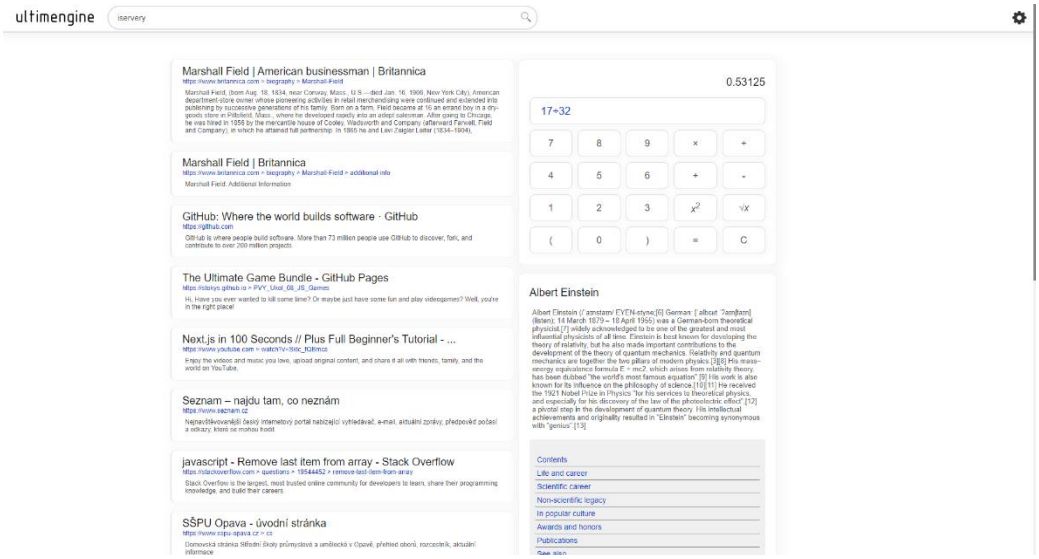
Výsledkem práce je plně funkční internetový vyhledávač s velmi vyvinutým *back-endem* a moderním, responsivním uživatelským webovým rozhraním na *front-endu*.

Náš web umí přijímat uživatelské dotazy skrze jednoduché vyhledávací pole, které při každém stisknutí klávesy komunikuje s modulem našeptávače, ten zpět vrací vždy nejvhodnější našepty, podle kterých má uživatel snadnější práci s psaním.



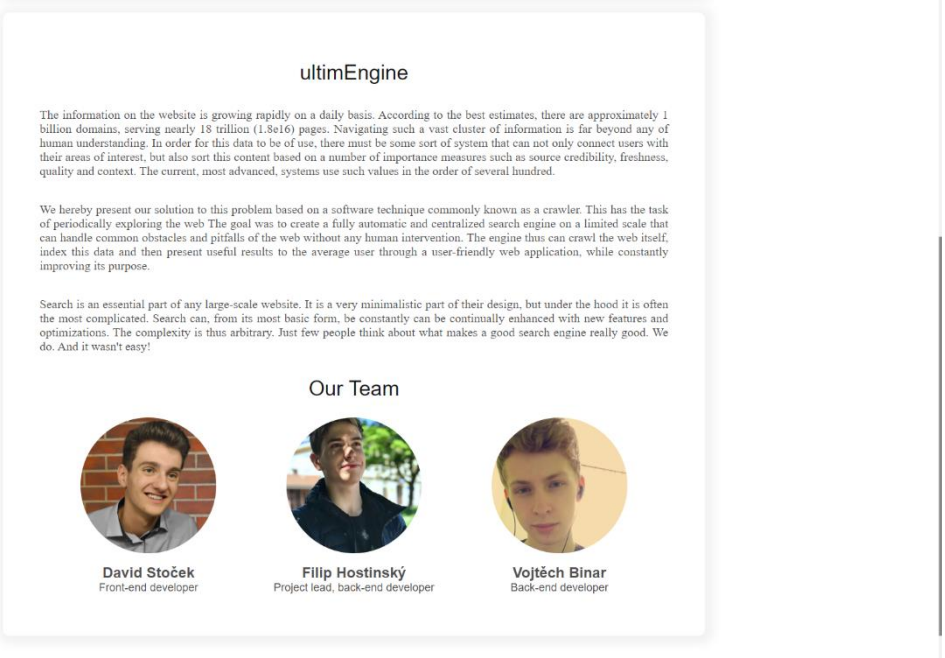
Obrázek 5 – Ukázka úvodní obrazovky s rozkliknutími našepty

Při potvrzení dotazu budete přesunuti na stránku s výsledky, které již při načtení budou dostupné díky komunikaci se search managerem, který posílá nejrelevantnější výsledky zpátky na *front-end*.



Obrázek 6 – Ukázka obrazovky s výsledky

Máte taky možnost kliknout na tlačítko s ikonku ozubeného kolečka a budete přesunuti na stánku s nastavením barevného režimu a sekci o “o nás”, kde je náš *search engine* stručně popsán návštěvníkům.



Obrázek 7 – Ukázka stránky s nastavením a „o nás“ sekci

ZÁVĚR

Cílem projektu bylo sestavit internetový vyhledávač se vším všudy. Mojí částí práce bylo vytvoření celého *front-endu* aplikace. Ten se skládal ze tří hlavních stránek a spousty důležitých komponentů, jako například pole pro zadávání dotazů, které úspěšně komunikuje s našeptávačem nebo komponent samotných výsledků, který za pomocí další vyřešené komunikace se *search managerem* dokáže zobrazit správná data o které si uživatel žádá.

Práce ovšem ale má i své limitace, navrácené výsledky nemusí vždy být ty nejlepší z důvodu malého počtu uložených dat o stránkách, ty zabírají hodně místa a jejich procesování muže trvat hodiny.

Projekt je možné vylepšit, některé by mohl odradit nedostatek všelijakých možných nástrojů pro usnadnění života, které *hinty* představují a jsou nyní mým hlavním cílem k dokončení. Do budoucna bych se chtěl více poprat s tímto *hint serverem* a jeho jednotlivými *hinty*, napadá mě něco ve smyslu převodu hodnot, jako délky, objemu, měny a podobně.

Výsledek práce můžete veřejně vidět na stránce *https://search.sspu-opava.cz/*, nebo jako zdrojový kód na našem *GitHub* repositáři *https://github.com/ultimate-search-engine/UltimateSearchEngine*.

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] *Getting Started | Next.js* [online]. 2016, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://nextjs.org/docs>

[2] *Sass: Documentation* [online]. 2006, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://sass-lang.com/documentation/>

[3] *TypeScript: The starting point for learning TypeScript* [online]. 2012, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://www.typescriptlang.org/docs/>

[4] *Getting Started – React* [online]. 2013, poslední revize 30.9.2022 [cit. 2023-01-06]. <https://reactjs.org/docs/getting-started.html>

[5] *next.js/example/with-docker-multi-env at canary · vercel/next.js* [online]. 2022, poslední revize 30.9.2022 [cit. 2023-01-06]. <https://github.com/vercel/next.js/commits/canary/examples/with-docker-multi-env>

[6] *Docker Documentation | Docker Documentation* [online]. 2013, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://docs.docker.com/>

[7] *CSS Tutorial* [online]. 1998, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://www.w3schools.com/css/default.asp>

[8] *JavaScript Tutorial* [online]. 1998, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://www.w3schools.com/js/default.asp>

[9] *React Tutorial* [online]. 2013, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://www.w3schools.com/react/default.asp>

[10] *Search engine – Wikipedia* [online]. 2006-2023, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://en.wikipedia.org/wiki/Search_engine>

[11] *Documentation | Node.js* [online]. 2009, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://nodejs.org/en/docs/>

[12] *JavaScript | MDN* [online]. 2005, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

[13] *React documentation – DevDocs* [online]. 2022, poslední revize 6.1.2023 [cit. 2023-01-06]. <https://devdocs.io/react/>