VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYHLEDÁVAČ PRO WEB VUT

SEARCH ENGINE FOR THE BUT WEBSITE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **PAVOL VŘBIK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. JAROSLAV DYTRYCH, Ph.D.**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

154237

Ústav: Ústav počítačové grafiky a multimédií (UPGM)
Student: **Vŕbik Pavol**
Program: Informační technologie
Název: **Vyhledávač pro web VUT**
Kategorie: Informační systémy
Akademický rok: 2023/24

Zadání:

1. Seznamte se s přístupy, technologiemi a nástroji pro fulltextové vyhledávání. Zaměřte se na možnosti nástroje Elasticsearch.
2. Prostudujte nejdůležitější datové struktury IS VUT potřebné pro fulltextové vyhledávání s důrazem na rozdílnost dat vyhledávání pro přihlášené a nepřihlášené uživatele.
3. Navrhněte vhodné řešení pro fulltextové vyhledávání na webových stránkách VUT. Zaměřte se při tom na jednoduchost, univerzálnost a efektivitu výsledného řešení.
4. Implementujte navržené řešení.
5. Zhodnoťte dosažené výsledky a vytvořte stručný plakát prezentující výsledky Vaší práce.

Literatura:

- Dle doporučení vedoucího.

Při obhajobě semestrální části projektu je požadováno:
Body 1, 2 a 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Dytrych Jaroslav, Ing., Ph.D.**
Konzultant: Witassek Pavel, Ing.
Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.
Datum zadání: 1.11.2023
Termín pro odevzdání: 9.5.2024
Datum schválení: 9.11.2023

Abstrakt

Cieľom tejto práce je navrhnúť a implementovať nové vyhľadávanie pre IS VUT s použitím nástroja pre fulltextové vyhľadávanie. Pôvodne používané vyhľadávanie spôsobovalo nadmerné zaťaženie databázy, a preto ho bolo potrebné nahradiť. Na základe vykonanej analýzy bol ako vhodný nástroj pre fulltextové vyhľadávanie vybraný Elasticsearch. Pre tento nástroj boli pripravené textové analyzátory, ktoré umožňujú jazykovú analýzu v českom a anglickom jazyku. Pre synchronizáciu dát medzi centrálnou databázou a Elasticsearch bol implementovaný nástroj, ktorý sa spúšťa v pravidelných intervaloch a udržuje tak vyhľadávanie aktuálne. Výsledkom práce je nové vyhľadávanie integrované do vyhľadávačov vo verejnej časti informačného systému VUT.

Abstract

The goal of this thesis is to design and implement a new search for the BUT IS using a full-text search tool. The originally used search was causing excessive load on the database, and therefore, needed to be replaced. Based on the analysis performed, Elasticsearch was selected as a suitable tool for full-text search. For this tool, text parsers were prepared to allow linguistic analysis in Czech and English. To synchronize the data between the central database and Elasticsearch, a tool was implemented that runs at regular intervals to keep the search up-to-date. The result of the work is a new search integrated into the search engines in the public part of the BUT information system.

Klíčová slova

Vyhľadávanie, fulltextové vyhľadávanie, vyhľadávanie informácií, vyhľadávač, Elasticsearch, IS VUT, PHP, SQL, REST, Guzzle

Keywords

Search, fulltext search, information retrieval, search engine, Elasticsearch, IS BUT, PHP, SQL, REST, Guzzle

Citace

VŘBIK, Pavol. *Vyhledávač pro web VUT*. Brno, 2024. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jaroslav Dytrych, Ph.D.

Vyhledávač pro web VUT

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Jaroslava Dytrycha, Ph.D. Ďalšie informácie mi poskytli pán Ing. Pavel Witassek, pán Ing. Marek Strakoš, pán Ing. Roman Bártl a pán Ing. Miroslav Skopal. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

. . . . . .
Pavol Vŕbik
7. května 2024

Poděkování

Rád by som poďakoval vedúcemu práce Ing. Jaroslavovi Dytrychovi, Ph.D. za pomoc a rady pri vypracovávaní tejto bakalárskej práce. Ďalej by som rád poďakoval konzultantovi práce Ing. Pavlovi Witassekovi za pravidelné konzultácie a cenné rady ohľadom štruktúry informačného systému VUT a Ing. Marekovi Strakošovi, Ing. Romanovi Bártlovi za ich odborné konzultácie.

Obsah

Kapitola 1

Úvod

Vyhľadávanie je v rámci verejnej časti informačného systému univerzity Vysoké učení technické v Brně využívané pre získavanie informácií z centrálnej databázy. Umožňuje externým návštevníkom, študentom alebo zamestnancom efektívne pristupovať k veľkému množstvu akademických dát a filtrovať ich podľa rôznych kritérií.

Doteraz vyhľadávanie v informačnom systéme VUT prebiehalo priamo cez databázové operácie, ktoré však spôsobovali nadmerné zaťaženie databázového servera. Z dlhodobého hľadiska, v ktorom sa bude objem digitálnych dát v informačnom systéme neustále zvyšovať, bolo potrebné nájsť spoľahlivé riešenie pre zníženie zaťaženia databázového servera spôsobeného procesmi vyhľadávania.

Cieľom tejto práce je návrh a implementácia nového riešenia pre vyhľadávanie v informačnom systéme VUT s použitím voľne dostupného vyhľadávacieho nástroja. Na základe vykonanej analýzy bol za tento nástroj vybraný Elasticsearch. Toto vyhľadávanie by malo znížiť zaťaženie databázového servera a prinášať používateľom relevantné výsledky. Použitý vyhľadávací nástroj by mal byť spoľahlivo nakonfigurovaný pre prácu najmä s českým a anglickým jazykom. Je rovnako dôležité, aby Elasticsearch pracoval s aktuálnymi dátami a bol pravidelne synchronizovaný s centrálnou databázou. Zároveň musí byť flexibilný pre dynamickú prácu s dátami a z pohľadu dlhodobej udržateľnosti schopný škálovania.

V kapitole 2 je opísané fulltextové vyhľadávanie z pohľadu univerzitných dát. Kapitola analyzuje aj stav vyhľadávania na iných českých univerzitách a v závere opisuje získané poznatky potrebné pre návrh nového vyhľadávania v informačnom systéme VUT. Kapitola 3 opisuje štruktúru informačného systému VUT z pohľadu vyhľadávania. Informuje o prehľadávaných dátach, dostupných vyhľadávačoch a dátových modeloch využívaných pre získavanie dát z databázy. Kapitola 4 sa zaoberá výberom vhodného nástroja pre fulltextové vyhľadávanie. Opisuje niekoľko dostupných nástrojov a v závere uvádza proces výberu nástroja pre implementáciu v informačnom systéme VUT. Informácie o návrhu nového vyhľadávania popisuje kapitola 5. V tejto kapitole je opísaný princíp komunikácie s nástrojom Elasticsearch a návrh komplexnej analýzy textových dát. Okrem toho zachytáva aj návrh synchronizácie dát medzi centrálnou databázou a indexmi vyhľadávacieho nástroja. V závere kapitoly je opísaný návrh implementácie nového vyhľadávania do vyhľadávačov dostupných na webe VUT. Implementácia navrhnutého riešenia je bližšie popísaná v kapitole 6. Testovanie tejto implementácie a procesu analýzy textu spolu s informáciami o nasadení je opísané v kapitole 7. Kapitola 8 zhŕňa dosiahnuté výsledky a uvádza možné rozšírenia funkcionality do budúcnosti.

Kapitola 2

Univerzitné fulltextové vyhľadávanie

Táto kapitola popisuje a analyzuje stav aktuálneho vyhľadávania na webovej časti informačného systému VUT a porovnáva ho s vyhľadávaním dát na ďalších piatich najlepších univerzitách v Českej republike, ktoré boli do porovnania vybrané na základe ich umiestnenia v rebríčku ARWU 2023[1].

2.1 Úvod do fulltextového vyhľadávania

Fulltextové vyhľadávanie predstavuje kľúčovú zložku všetkých moderných informačných systémov a webových stránok, ktorá umožňuje efektívne prehľadávanie veľkého množstva textových dát na základe konkrétnych výrazov alebo fráz. Podľa [2] sa jedná o typ vyhľadávania, ktoré počítač vykonáva, keď porovnáva výrazy v požadovanom dotaze s výrazmi v jednotlivých dokumentoch uložených v jeho databáze a algoritmicky zoraďuje nájdené výsledky. Fulltextové vyhľadávače a ich fungovanie sú podrobnejšie rozobraté v sekcii 4.1.

Princíp fungovania

Fulltextové vyhľadávanie funguje na princípe rozdelenia rozsiahleho textu na jednotlivé slová, ktoré sú potom ukladané do indexu. Počas indexácie dochádza k rôznym úpravám týchto slov s cieľom zlepšiť relevanciu vyhľadávania. Index je katalógom všetkých slov nájdených v dokumentoch, spolu s ich polohou v rámci konkrétneho dokumentu. Pri vyhľadávaní sa dotaz od používateľa znova rozdelí na jednotlivé slová, ktoré sú porovnávané oproti indexu. V prípade určitých zhôd a splnenia určitých podmienok sa vracia zoznam nájdených výsledkov.

Prínosy

Fulltextové vyhľadávanie je dôležitou súčasťou v oblasti vyhľadávania a spracovania informácií. Umožňuje pracovať s veľkým množstvom dát a efektívne sa prispôsobovať rastúcemu počtu uložených dokumentov. Oproti tradičným vyhľadávacím algoritmom ponúka podľa [10] niekoľko kľúčových výhod:

[1] https://www.shanghairanking.com/institution

- **Väčšia presnosť** – fulltextové vyhľadávanie ponúka relevantnejšie výsledky v porovnaní s tradičnými vyhľadávacími algoritmami. Je to preto, že pri indexácii prehľadáva celý text dokumentu a nie iba konkrétne kľúčové slová.
- **Efektivita vyhľadávania** – fulltextové vyhľadávanie je rýchlejšie a efektívnejšie ako tradičné vyhľadávacie algoritmy, pretože vyhľadávanie prebieha cez uložený index slov a nepotrebuje tak pri každom vyhľadávaní algoritmicky prechádzať všetky dokumenty.
- **Lepší používateľský zážitok** – fulltextové vyhľadávanie poskytuje užívateľsky príjemnejšie vyhľadávanie. Používatelia môžu zadávať dopyty v prirodzenom jazyku a získať na základe analýzy textu takmer okamžite relevantné výsledky.

Obmedzenia

Ako uvádza Jeffrey Beall [2], fulltextové vyhľadávanie prináša aj svoje slabiny. Jedným z najväčších problémov môžu byť synonymá, pretože často existuje viac spôsobov, ako pomenovať alebo vyjadriť daný pojem. To bráni správnemu získavaniu informácií a efektívnemu indexovaniu. Rovnaký problémom môžu spôsobovať skratky či akronymy. Zavádzajúce tiež môžu byť homonymá, čiže slová píšuce sa rovnako s iným významom, ktoré môže vyhľadávanie chybne zaradiť a prinášať tak nepresné výsledky. Takmer všetky obmedzenia sa však dajú vyriešiť za použitia adekvátnych slovníkov, algoritmov či rôznych overených postupov.

2.2 Problematika univerzitného vyhľadávania

Vyhľadávanie na webových stránkach je proces, pri ktorom sa používateľ snaží získať relevantné informácie z danej webovej stránky a získať odpovede na svoje otázky. Pri vyhľadávaní na univerzitnej webovej stránke môže mať užívateľ rôzne ciele, ako napríklad:

- nájsť presné informácie o vedeckej a výskumnej činnosti univerzity – vyhľadávanie konkrétnych projektov, článkov a ďalších,
- nájsť užitočné informácie o univerzite, fakultách, histórii alebo spolupráci s inými univerzitami či organizáciami,
- nájsť informácie o študijných programoch, kritériách pre prijatie či prijímacích skúškach,
- nájsť informácie o vyučovaných predmetoch, štipendiách, ubytovaní, poradenstve, alebo
- nájsť kontakt alebo informácie o pedagogickom či nepedagogickom pracovníkovi.

Univerzitné vyhľadávanie by malo používateľovi poskytnúť relevantné výsledky v čo najkratšom čase. Vyhľadávač by mal byť teda rýchly, optimalizovaný pre univerzitné použitie a prinášať spoľahlivé výsledky. Dôležitá je aj schopnosť vyhľadávača rozpoznať rôzne typy dotazov, spracovať ich a eliminovať irelevantné či duplicitné výsledky.

2.3 Analýza aktuálneho vyhľadávania na VUT

Vysoké učení technické v Brně informuje na svojich webových stránkach o aktualitách, svojom výskume, štúdiu či spolupráci. Vyhľadávanie by teda malo pokrývať všetky tieto dáta a poskytovať návštevníkom presné informácie, ktoré na webovej stránke hľadajú. Okrem toho je dôležité, aby boli výsledky vyhľadávania radené na základe skóre, ktoré odráža relevanciu každého dokumentu.

Hodnotenie výsledkov

Aktuálne vyhľadávanie nesprávne ohodnocuje a zoraďuje nájdené výsledky. Môže sa teda stať, že sa najskôr zobrazia menej relevantné výsledky, čo negatívne ovplyvňuje užívateľskú skúsenosť. Problém sa vyskytuje najmä pri vyhľadávaní smerníc, kedy sa častokrát na prvých priečkach zobrazia už neplatné, a teda archívne smernice, aj napriek tomu, že k ním existujú platné ekvivalenty. Podobný problém sa vyskytuje pri vyhľadávaní presného názvu dokumentu, kedy sa presná zhoda zaradí medzi menej relevantné výsledky.

Presnosť výsledkov

Vyhľadávanie ponúka pomerne relevantné výsledky k dopytovaným výrazom. Nájdu sa však prípady, kedy sa v najrelevantnejších výsledkoch vyhľadávania zobrazí položka, ktorá priamo nesúvisí s hľadaným výrazom. Existujú aj prípady, kedy sa vo výsledkoch nezobrazí správne vyhľadávaná stránka, len jej nadradený rodič. Tento problém možno pozorovať napríklad pri hľadaní stránky smerníc, kedy sa vo výsledkoch zobrazí len úradná doska. Problémom sú aj preklepy, ktoré aktuálne vyhľadávanie nedokáže ošetriť, čo spôsobuje výpadok niektorých výsledkov.

Užívateľská skúsenosť

Užívateľské rozhranie každej časti vyhľadávania pôsobí jednotne a každý vyhľadávač má zreteľné využitie, ktorému sú prispôsobené aj jeho filtre. Pri vyhľadávaní chýba aktívne našepkávanie, ktoré by pomohlo užívateľovi nájsť presnejšie výsledky za kratší čas. Užívateľsky menej prívetivejšie sú aj filtre všeobecného, teda hlavného vyhľadávania, ktoré neumožňujú filtrovať výsledky podľa vedy a výskumu.

2.4 Analýza vyhľadávania na UK

Univerzitná webová stránka Karlovej univerzity umožňuje vyhľadávanie informácií pomocou hlavného vyhľadávania alebo vyhľadávania na jednotlivých podstránkach. Najviac informácií je možné nájsť práve vo verejnej časti informačného systému, kde sa dajú vyhľadávať napríklad vyučované predmety alebo osoby. Okrem toho má Karlova univerzita k dispozícii aj vyhľadávanie v repozitári záverečných prác a informačných zdrojoch.

Relevantnosť výsledkov

Hlavné vyhľadávanie na verejnom webe ponúka relatívne výsledky a dokáže pracovať aj s kmeňovou analýzou slov, vďaka čomu odstraňuje z hľadaného dotazu koncovky slov a rozširuje tak oblasť vyhľadávania. Problém však nastáva pri preklepe, kedy vyhľadávanie neponúka žiadne nájdené výsledky. Relevantné výsledky poskytuje aj vyhľadávanie v systéme

DSpace, ktorý je využitý pre repozitár záverečných prác a informačnom systéme Primo, ktorý sa využíva pre informačné zdroje.

Užívateľská skúsenosť

Vyhľadávanie v niektorých častiach webovej stránky, najmä v časti informačného systému pôsobia zastarano a vykonávanie vyhľadávacích operácií trvá častokrát až niekoľko sekúnd. Rýchlosť nájdenia výsledku v hlavnom vyhľadávaní je takmer okamžitá, chýbajú však možnosti presnejšej filtrácie. Užívateľsky prívetivo však pôsobí najmä vyhľadávanie v systémoch DSpace a Primo, ktoré poskytujú rôzne možnosti filtrácie výsledkov a vysokú rýchlosť vyhľadávania.

2.5 Analýza vyhľadávania na MUNI

Vyhľadávanie na webe Masarykovej univerzity sa rozdeľuje do niekoľkých častí: hlavné vyhľadávanie, hľadanie v študentskej časti webu, samostatné vyhľadávanie v informačnom systéme, univerzitnom repozitári a v záverečných prácach. Každá časť vyhľadávania má jasné zameranie a ponúka adekvátne výsledky.

Relevantnosť výsledkov

Vyhľadávanie na MUNI ponúka pomerne presné výsledky, ktoré sú správne ohodnotené a zoradené podľa relevantnosti. Hlavné vyhľadávanie ponúka prekrytie aj na podstránky fakúlt, čo uľahčuje používateľovi vyhľadávanie a orientáciu v dostupných informáciach. Orientáciu vo výsledkoch uľahčuje aj zvýrazňovanie kľúčových slov v popise výsledku.

Užívateľská skúsenosť

V rámci hlavného vyhľadávania sú k dispozícii len dva základné filtre – všetky výsledky a ľudia. To negatívne ovplyvňuje skúsenosť s vyhľadávaním, nakoľko sa v mnohých prípadoch užívateľ musí orientovať vo veľkom množstve výsledkov. Okrem toho hlavné vyhľadávanie nepokrýva záverečné práce študentov. Vyhľadávanie v informačnom systéme pôsobí viac užívateľsky prívetivejšie. Ponúka podrobné filtre, rýchle vyhľadávanie už počas písania dotazu, ktoré je graficky a prehľadne rozdelené do kategórii.

2.6 Analýza vyhľadávania na ČZU

Hľadanie na webe Českej zemědělskej univerzity v Praze sa delí na niekoľko častí. Základom je hlavné vyhľadávanie, ktoré je prístupné počas celého prehľadávania verejného webu a aj na podstránkach jednotlivých fakúlt. Hľadaný dotaz sa tu vyhľadáva samostatne v databáze, ale aj cez vyhľadávač služby Google, ktorý indexuje najmä jednotlivé podstránky. Okrem toho sa nachádzajú na verejnom webe aj vyhľadávače projektov vedy a výskumu alebo záverečných prác.

Relevantnosť výsledkov

Hlavné vyhľadávanie ponúka relevantné výsledky najmä vďaka vyhľadávaniu cez službu Google. Vyhľadávanie v databáze je v skutočnosti určené len pre vyhľadávanie osôb. Dokáže spracovávať čiastočné slová, nedokáže však pracovať s preklepmi. Vyhľadávanie záverečných

prác ponúka rovnako relevantné výsledky a nevyhľadáva sa len na základe názvu, ale aj abstraktu práce, čo rozširuje ponuku dostupných výsledkov.

Užívateľská skúsenosť

V prípade primárneho vyhľadávania pôsobí mätúco, že zadaný výraz sa vyhľadáva súčasne v dvoch vyhľadávačoch. Rovnako nejasné je, že priamo v databáze sa vyhľadávajú len vizitky osôb. Používateľ sa dozvie túto skutočnosť až z hlášky oznamujúcej, že sa nepodarilo v databáze nájsť žiadne záznamy osôb. Vyhľadávanie záverečných prác pôsobí prehľadne a ponúka rôzne možnosti filtrovania. V niektorých prípadoch však vyhľadávanie trvá aj niekoľko sekúnd.

2.7 Analýza vyhľadávania na UPOL

Na verejnom webe Univerzity Palackého v Olomouci je vyhľadávanie rozdelené do viacerých častí, ako sú hlavné vyhľadávanie, vyhľadávanie v databáze univerzitnej knižnice, vyhľadávanie kontaktov alebo informácii o študijných programoch. Hlavné vyhľadávanie je dostupné počas celej doby prehľadávania verejného webu a špecializované vyhľadávania sú k dispozícii na jednotlivých podstránkach.

Relevantnosť výsledkov

Hlavné vyhľadávanie, ktoré je implementované cez služby internetového vyhľadávača Google, ponúka relevantné výsledky spolu so zoradením podľa skóre. Problém mu nerobia ani preklepy alebo určité nepresnosti vo vyhľadávanom dotaze. Vyhľadávanie na podstránkach častokrát chybne radí menej relevantné výsledky na predné pozície. Napriek tomu sú ponúkané výsledky príslušné k používateľskému dotazu.

Užívateľská skúsenosť

Primárne vyhľadávanie na webe Univerzity Palackého v Olomouci využíva externú službu Google, čo môže viesť k menej konzistentnému používateľskému zážitku oproti vyhľadávaniu implementovanému priamo na podstránkach univerzity. Tieto vyhľadávania sú totiž špecificky prispôsobené potrebám a štruktúre daných sekcií a ponúkajú aj rôzne možnosti filtrovania výsledkov. Okrem toho tu je k dispozícii aj dynamické hľadanie výsledkov už počas písania dotazu. Vyhľadávanie v službe Google bližšie filtrovanie neumožňuje, vyhľadávať je teda možné len na základe konkrétneho používateľského dotazu.

2.8 Analýza vyhľadávania na ČVUT

Hľadanie na webe Českého vysokého učení technického v Praze je rozdelené do niekoľkých častí: vyhľadávanie na hlavnom webe, vo výsledkoch výskumu a na weboch jednotlivých fakúlt. Hlavný web univerzity používa vyhľadávanie pomocou internetového vyhľadávača Google. Ostatné podstránky alebo stránky fakúlt si implementujú vlastné vyhľadávanie špecificky zamerané na účely univerzitného vyhľadávania.

Relevantnosť výsledkov

Vyhľadávanie na podstránkach jednotlivých fakúlt či stránkach výsledkov vedy a výskumu ponúka relevantné výsledky. Vyhľadávanie vie pracovať s rôznymi tvarmi slova, problém mu však robia preklepy vo vyhľadávanom dotaze. Pri preklepoch neponúka žiadne výsledky, ani neponúkne opravený výraz.

Užívateľská skúsenosť

Vyhľadávanie na hlavnom webe ČVUT pôsobí nejednotne, nakoľko vyhľadávaný výraz je vždy presmerovaný na internetový vyhľadávač Google, ktorý nie je prispôsobený pre vyhľadávanie na univerzitnej stránke, a teda neponúka žiadne možnosti filtrácie výsledkov. Vyhľadávanie na ostatných stránkach fakúlt či výsledkov výskumu pôsobí jednotne, ponúka rôzne možnosti filtrovania a k dispozícii je aj automatické dopĺňanie dopytu.

2.9 Záver analýzy

Na základe komplexnej analýzy vyhľadávacích systémov na rôznych univerzitách možno potvrdiť kľúčové body zo sekcie 2.2, ktoré sú podstatné pre optimalizáciu a efektívnosť univerzitného vyhľadávania. Tieto poznatky poskytujú pohľad na to, ako by mohlo byť navrhnuté nové vyhľadávanie pre informačný systém VUT, aby čo najlepšie vyhovovalo potrebám návštevníkom stránky, študentom, akademickým pracovníkom a iným.

Segmentácia, teda rozdelenie vyhľadávania do jednotlivých logických častí, je nevyhnutná pre poskytnutie cielených a relevantných výsledkov. To umožňuje užívateľom lepšie sa orientovať vo veľkom objeme informácií a sústrediť sa na obsah, ktorý je pre nich dôležitý. V kontexte užívateľskej skúsenosti je dôležitá intuitívnosť ovládania a funkcie, ktoré používateľovi zjednodušujú proces hľadia potrebnej informácie. Medzi tieto funkcie patria napríklad podrobné filtračné možnosti a rýchle vyhľadávanie už počas písania dotazu.

Relevancia a presnosť výsledkov sú rovnako kľúčové pre užívateľskú spokojnosť. Vyhľadávania, ktoré efektívne hodnotia a zoraďujú výsledky podľa ich relevancie, značne zlepšujú celkovú užívateľskú skúsenosť. Zvýrazňovanie kľúčových slov v popisoch výsledkov môže tiež značne uľahčiť orientáciu v nájdených informáciách.

Vyhľadávače by mali byť tolerantné voči drobným chybám v dotazoch, ako sú napríklad preklepy, čo pomáha v získavaní relevantných informácií aj v prípadoch, keď je pôvodný dotaz čiastočne nesprávny. Okrem toho je tiež dôležité, aby vyhľadávací systém vedel efektívne pracovať s nekompletnými slovami, ktoré používatelia zadávajú v snahe ušetriť čas pri formulovaní dotazov.

Kapitola 3

Informačný systém VUT

Táto kapitola popisuje súčasnú štruktúru informačného systému VUT z pohľadu vyhľadávania verejných informácií. Bližšie opisuje oblasti prehľadávaných dát a vyhľadávače, ktoré sa nachádzajú vo verejnej časti informačného systému. Súčasťou kapitoly je aj popis modelovej vrstvy Vut2, ktorá pri operáciach s dátami tvorí dôležitú časť systému.

3.1 Prehľadávané dáta v informačnom systéme

Prehľadávané dáta v informačnom systéme sa rozdeľujú do niekoľkých individuálnych oblastí, ktoré sú integrované do príslušných vyhľadávačov na webovom portáli VUT. Táto sekcia opisuje štruktúru týchto oblastí, ich kľúčové prvky a obmedzenia.

Osoby a telefónne čísla

Každá osoba v informačnom systéme VUT má priradené svoje jedinečné identifikačné číslo, adresu elektronickej pošty a roľu na základe štúdia alebo typu profesie, ktorú v rámci univerzity vykonáva. Osoba je okrem toho v systéme zaradená do jednej z hlavných skupín: študent, zamestnanec alebo externý pracovník. Vyhľadávanie osôb prebieha podľa mena, adresy elektronickej pošty alebo v prípade zamestnancov aj podľa telefónneho čísla pevnej linky. Telefónne čísla sú priradené nielen ku konkrétnej osobe, ale aj k areálu a miestnosti, v ktorej sa telefón nachádza.

Z dôvodu ochrany osobných údajov GDPR[1] nie je možné zobrazovať informácie o študentoch pre všetkých návštevníkov webového portálu. Vizitka študenta obsahujúca typ štúdia a študijný program sa teda zobrazuje len užívateľovi prihlásenému do informačného systému VUT. V prípade, že je študent aj zamestnancom univerzity, zobrazí sa verejnému návštevníkovi len informácia o pracovnom pomere osoby.

Záverečné práce

Dátová štruktúra záverečných prác obsahuje informácie o názve práce, roku, autorovi, vedúcom a príslušnej fakulte. Detail práce je doplnený aj o abstrakt, jazyk práce a kľúčové slová. Pre primárne vyhľadávanie v informačnom systéme sa používa názov práce a pre dodatočné filtrovanie výsledkov rok, fakulta, meno autora alebo vedúceho a jazyk práce.

[1] https://www.mvcr.cz/gdpr/clanek/co-je-gdpr.aspx

Pri zobrazovaní autora práce sa uplatňuje rovnaký princíp ochrany osobných údajov ako pri vyhľadávaní osôb. Neprihlásenému návštevníkovi sa z toho dôvodu zobrazí len meno autora, bez možnosti zobrazenia jeho vizitky v informačnom systéme.

Programy a obory

Každý program a obor má v informačnom systéme priradený svoj názov, skratku, fakultu pod ktorú patrí, a rok platnosti. Obor si okrem toho uchováva informáciu, pod ktorý program patrí. Vyhľadávanie prebieha na základe názvu alebo skratky a výsledky sú filtrované podľa aktuálneho akademického roku. Tým sa dosiahne, že sú vo výsledkoch vyhľadávania dostupné len aktuálne programy a ich obory.

Zložky, dokumenty a ich prílohy

Dokument má v systéme svoj názov a telo, ktoré sa môže skladať z rôznych typov obsahu. Okrem toho má pridelený svoj jednotný identifikátor zdroja URI, jazykový kód a kód webu, na ktorom má byť dokument viditeľný (pozri sekcia 3.3). Jednotlivé dokumenty sa radia do zložiek, ktoré určujú ich kategóriu. Dokument môže mať pridané aj prílohy, ktoré majú svoj názov a jazykový kód. Na základe názvu je možné samostatne vyhľadávať dokumenty, zložky a aj prílohy.

V rámci dokumentu a jeho príloh je možné nastaviť dátum vystavenia a platnosti. Dátum vystavenia dokumentu určuje, kedy sa stáva verejne dostupným. Naopak, dátum platnosti definuje, kedy dokument stráca platnosť a prestáva byť prístupný pre prezeranie.

Predmety

V informačnom systéme VUT majú predmety svoj názov, skratku, fakultu a garanta. Ďalej štruktúra predmetu uchováva informácie o semestri, v ktorom sa predmet vyučuje, a akademický rok výučby. Okrem toho obsahuje aj informáciu o jazyku výučby a označenie, či je predmet vhodný pre zahraničných študentov. Primárne vyhľadávanie výsledkov prebieha na základe názvu predmetu alebo jeho skratky a dodatočné upresnenie výsledkov je vykonávané cez filtre semestra, akademického roku, fakulty alebo jazyku výučby.

Stránky

Jednotlivé stránky a podstránky informačného systému VUT majú názov a jednotný identifikátor zdroja URI. Podobne ako dokumenty a prílohy, aj stránky obsahujú informácie o tom, na akom webe v rámci informačného webu VUT majú byť prístupné. Z dôvodu jazykových variácii stránok je k dispozícii aj jazykový kód. Pre vyhľadávanie sa využíva najmä názov, ale k dispozícii sú aj kľúčové slová, ktoré môžu pomôcť k nájdeniu požadovanej stránky.

Veda a výskum

Dáta vedy a výskumu sa rozdeľujú na niekoľko typov. Patria sem publikácie, patenty, produkty a umelecké výstupy. Všetky tieto časti obsahujú položky ako sú názov, mená autorov alebo rok zverejnenia záznamu. K tomu obsahuje informácie o vydavateľovi alebo o vytvorenej citácii práce. Patenty okrem toho uchovávajú informáciu o pridelenom čísle patentu. Vyhľadávanie prebieha na základe názvu, autorov práce alebo kľúčových slov. K bližšiemu filtrovaniu sa využíva rok zverejnenia alebo typ výsledku výskumu.

3.2 Rozdelenie verejných vyhľadávačov

Vyhľadávanie na univerzitnom webe VUT sa delí do niekoľkých častí: všeobecné vyhľadávanie, vyhľadávanie v databáze vedy a výskumu, predmetov a záverečných prác. Tieto vyhľadávače podľa svojho určenia ponúkajú vyhľadávanie nad príslušnými dátami opísanými v sekcii 3.1 a umožňujú presnejšiu filtráciu vyhľadávania alebo prehľadnejší spôsob zobrazovania nájdených výsledkov.

- **Všeobecné vyhľadávanie** – k dispozícii počas celého prehliadania webu VUT. Umožňuje vyhľadávanie v oblastiach záverečných prác, osôb a telefónnych čísel, programov a oborov, zložiek a ich príloh, predmetov a stránok. Ponúka jednoduchú možnosť filtrácie výsledkov podľa vnútorných noriem, ľudí a záverečných prác. Výsledky vyhľadávania obsahujú len základné informácie, ako názov či typ dokumentu a krátky slovný popis nájdeného výsledku.
- **Vyhľadávanie predmetov** – umiestnené v študijnej oblasti verejného webu. Zabezpečuje vyhľadávanie v oblasti predmetov zo všetkých fakúlt VUT. Ponúka rozšírenú možnosť filtrácie podľa fakulty, aktuálneho akademického roku, semestra alebo jazyka výuky. Výsledky vyhľadávania informujú o názve predmetu, jeho skratke, garantovi a fakulte, na ktorej sa vyučuje.
- **Vyhľadávanie záverečných prác** – dostupné v študijnej oblasti webu VUT. Ponúka vyhľadávanie v oblasti všetkých typov záverečných prác. Umožňuje filtrovanie výsledkov podľa fakulty, príslušného ústavu, akademického roku, typu práce a jazyku výuky. Vo výsledkoch vyhľadávania ponúka informácie o názve práce, jej type, autorovi a vedúcom.
- **Vyhľadávanie v databáze vedy a výskumu** – k dispozícii je vo webovej časti vedy a výskumu. Ponúka hľadanie vo výsledkoch a projektoch vedy a výskumu na VUT. Filtrácia výsledkov je možná na základe typu výsledku, roku a v prípade projektov aj podľa fakulty, ústavu alebo stavu projektu. Vo výsledkoch vyhľadávania sa zobrazujú názvy projektu alebo výsledku a krátky popis.

3.3 Štruktúra informačného systému

Webový informačný systém VUT sa skladá zo štyroch hlavných aplikácií. Sú to Portál, StudIS, Teacher a ePrihláška. Aplikácia Portál je rozdelená do verejnej časti, ktorá je určená pre verejný web a internej, ktorá sprostredkuje osobné alebo pracovné informácie. StudIS je primárne zameraná na študijné záležitosti z pohľadu študenta a aplikácia Teacher zase pre študijnú agendu z pohľadu vyučujúceho. Posledná hlavná aplikácia ePrihláška je určená pre záujemcov o štúdium na VUT a prináša možnosť podania prihlášky alebo sledovania výberového procesu. Všetky tieto aplikácie sú napísané v skriptovacom jazyku PHP vo verzii 7.4. Táto verzia jazyka PHP prináša podľa [3] výrazné zvýšenie výkonu (až dvojnásobne vyššia rýchlosť oproti predchádzajúcim verziám) a znížený spotrebu pamäte, čo zaisťuje plynulý chod všetkých aplikácií.

Z pohľadu získavania verejných informácií sa využíva práve aplikácia Portál, konkrétne jej časť pre verejný web. Táto aplikácia je aktuálne vo svojej štvrtej verzii a využíva architektúru `Model-View-Controller`, ktorá oddeľuje aplikačnú logiku od používateľského

rozhrania. Pre prácu s dátami sa využíva systém riadenia relačnej bázy dát Oracle Database vo verzii 19c[2], nad ktorým informačný systém vykonáva operácie cez modely Vut2. Okrem centrálnej databázy sa využíva aj medzipamäť Redis[3] pre databázové dotazy alebo ukladanie používateľských nastavení.

Verejná časť aplikácie portál je okrem hlavného univerzitného webu súbežne nasadená aj na webových stránkach niektorých fakúlt. Dáta určené pre tieto fakulty sú preto označené príslušným identifikačným číslom webovej stránky.

3.4 Modelová vrstva Vut2

Modelová vrstva podľa [15] slúži ako centrálny komponent paradigmy Model-View-Controller. Táto vrstva zabezpečuje spracovanie a dočasné uchovávanie dát, ako aj ich transformáciu potrebnú pre ich ďalšie použitie. Model je oddelený od užívateľského rozhrania, čo umožňuje zmeny v dátach alebo logike bez toho, aby to malo vplyv na zobrazenie alebo interakciu s užívateľom. Tieto modely sú zdieľané naprieč celým informačným systémom VUT, čo je zásadný rozdiel oproti prechádzajúcej verzii modelovej vrstvy, ktorej modely slúžili len pre konkrétnu aplikáciu. Modely sa skladajú z niekoľkých častí:

- **Abstract** – abstraktná trieda modelu, ktorá obsahuje vlastnosti odpovedajúce databázovému objektu a metódy pre získanie hodnoty určitej vlastnosti (`getter`) alebo nastavenie jej hodnoty (`setter`).
- **Builder** – poskytuje vytváranie inštancií objektu či jeho iterátoru a predávanie závislostí.
- **Iterator** – vytvára kolekciu objektov modelu a definuje metódy pre prácu s ňou. Okrem toho umožňuje nastaviť informácie o stránkovaní alebo o vlastnostiach kolekcie.
- **Mapper** – obsahuje a vykonáva SQL operácie nad databázou a získané dáta mapuje na atribúty objektu.
- **Model** – primárna trieda modelu a potomok abstraktnej triedy. Implementuje dodatočné metódy pre rozšírenie funkčnosti modelu.
- **Repository** – načítava a ukladá dáta získané z triedy mapper. Okrem toho môže tieto dáta ukladať v dočasnej pamäti cache pre urýchlenie ďalších prichádzajúcich požiadavok.

Tento prístup sa využíva nielen v rámci aplikačnej logiky, ale aj v dátovej vrstve, kde zrkadlí štruktúru databáze a zabezpečuje prístup k jej dátam. To umožňuje adaptáciu na neustále meniace sa požiadavky v rámci celého informačného systému a prispieva k škálovateľnosti aplikácií.

[2] https://docs.oracle.com/en/database/oracle/oracle-database/19/index.html
[3] https://redis.io

Kapitola 4

Technológie pre fulltextové vyhľadávanie

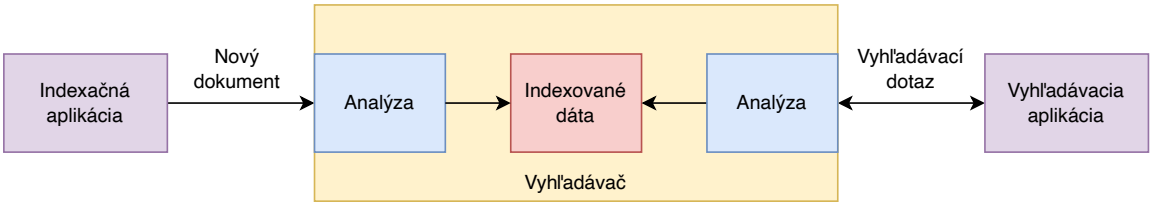
Obsahom tejto kapitoly je popísanie a porovnanie dostupných technológií pre fulltextové vyhľadávanie. Fulltextové vyhľadávanie slúži na vyhľadávanie informácií v rozsiahlych textových dokumentoch na základe zadaných kľúčových slov alebo fráz. Tieto technológie sa využívajú najmä pri webových vyhľadávačoch, kde pomáhajú používateľom nájsť relevantné informácie rýchlo a presne.

4.1 Predstavenie fulltextových vyhľadávačov

Informácie v tejto sekcii boli prebraté z [4]. Vyhľadávač je softvérový systém, ktorý porovnáva dotazy s indexovanými dokumentmi a vytvára z nich zoradené zoznamy dokumentov. Vyhľadávače sú navrhnuté tak, aby zachytávali a spracovávali veľké množstvá údajov, poskytovali rýchlu odozvu na dotaz a začleňovali nové dokumenty či informačné zdroje. Používajú rôzne konfigurácie a algoritmy v závislosti od aplikácie, pre ktorú sú určené.

Fungovanie webových vyhľadávačov

Vyhľadávač zastáva dve hlavné funkcie, ktoré nazývame proces indexovania a proces vyhľadávania. Ako opisuje obrázok 4.1, proces indexácie je iniciovaný indexačnou aplikáciou a proces vyhľadávania sa spúšťa na základe dotazu z vyhľadávacej aplikácie. Pri procese indexovania sa generujú dátové štruktúry, indexy, optimalizované pre rýchle vyhľadávanie. Proces vyhľadávania potom využíva tieto štruktúry a dotaz používateľa na vytvorenie zoradeného zoznamu relevantných dokumentov.



Obrázek 4.1: Zjednodušený diagram popisujúci životný cyklus vyhľadávačov. Obrázok bol inšpirovaný [6].

Pri procese indexácie dochádza k dôkladnej analýze každého nového dokumentu, ktorý má byť zaradený do vyhľadávacieho systému. Priebeh tejto analýzy spočíva v niekoľkých krokoch:

- **Parsing** – je to proces získavania dát z textového dokumentu a ich transformácie na tokeny, teda základné jednotky textu, ktoré budú použité v ďalších krokoch analýzy.
- **Stopping** – proces, ktorý má za úlohu odstrániť z toku prichádzajúcich tokenov všetky najčastejšie používané slová, medzi ktoré patria napríklad predložky či spojky. Ich odstránenie môže značne znížiť veľkosť indexov a urýchliť proces vyhľadávania.
- **Stemming** – zoskupuje a redukuje slová, ktoré sú odvodené od spoločného koreňa. Príkladom sú slová „skočil“, „vyskočila“, „preskočil“, ktoré sa zredukujú na „skočil“. Zvýši tým pravdepodobnosť, že slová použité v dotazoch a dokumentoch sa budú zhodovať.
- **Weighting** – proces váhovania slov, ktorý pomáha určiť dôležité a aj naopak menej dôležité slová v dokumentoch. Posudzovanie dôležitosti slova prebieha na základe jeho frekvencie v dokumente v porovnaní s jeho frekvenciou vo všetkých dokumentoch.

Analýza prebieha aj pri procese vyhľadávania. V tomto prípade vyhľadávač analyzuje dotaz od používateľa, pri ktorom častokrát využíva podobné procesy ako pri indexácii dokumentu s cieľom spresniť hľadaný dotaz. V priebehu analýzy dotazu sa teda využíva tokenizácia, stopping či stemming, a k tomu sa pridáva kontrola pravopisu a navrhovanie dotazov.

- **Kontrola pravopisu** – Dôležitá súčasť spracovania dotazov vo vyhľadávačoch. Pomáha opraviť pravopisné chyby v dotazoch používateľov, čo môže zlepšiť presnosť výsledkov vyhľadávania.
- **Navrhovanie dotazov** – Poskytuje používateľovi alternatívne dotazy, ktoré častokrát ponúkajú opravu pravopisných chýb alebo poskytujú spresnený opis jeho dotazu.

4.2 Apache Lucene

Lucene je podľa [11] vysoko výkonná a škálovateľná knižnica napísaná v programovacom jazyku `Java`. Používa sa na indexovanie a vyhľadávanie textových dát v rôznych formátoch a rôznych svetových jazykoch. Lucene umožňuje integrovať vyhľadávacie schopnosti do aplikácií a indexovať dáta z rôznych zdrojov, ako sú webové stránky, dokumenty uložené v lokálnych systémoch alebo textové súbory. Vysoká konfigurovateľnosť umožňuje prispôsobiť vyhľadávanie špecifickým potrebám a jazykom. K dispozícii sú rôzne pokročilé funkcie, ako napríklad vyhľadávanie blízkosti slov (hľadá výskyt slov, ktoré sú blízko seba a môžu medzi sebou indikovať určitý vzťah), zvýrazňovanie vo výsledkoch vyhľadávania, vlastné textové analyzátory a iné. Lucene je základom pre rôzne ďalšie vyhľadávacie technológie, ako je napríklad Elasticsearch alebo Solr.

Kľúčovou oblasťou využitia Lucene je tvorba vlastných vyhľadávacích riešení pre webové stránky. Okrem toho je častokrát využívaný v oblasti rôznych informačných systémoch a programoch spracúvajúcich veľké množstvo textových dát. Využitie teda nájde v rôznych oblastiach, od menších webových implementácií až po rozsiahle informačné systémy.

4.3 Elasticsearch

Elasticsearch je distribuovaný vyhľadávací a analytický nástroj založený na Apache Lucene. Jedná sa o nástroj vyvíjaný v jazyku `Java`, ktorý disponuje aplikačným rozhraním REST a ponúka vysokú dostupnosť, rýchlosť a škálovateľnosť. Ako opisuje [6], používa sa na indexovanie, vyhľadávanie a agregáciu veľkého množstva dát, a to takmer v reálnom čase. Podporuje rôzne typy dát, ako je napríklad text, čísla, dátumy, geografické polohy a mnoho ďalších. Umožňuje okrem toho vytvárať vlastné analyzátory textu, prácu s rôznymi jazykmi a rozsiahle dopyty. K dispozícii sú aj ďalšie pokročilé funkcie, ako je napríklad fuzzy vyhľadávanie (toleruje malé chyby v zadanom dotaze a aj napriek rozdielom prináša relevantné výsledky), geopriestorové vyhľadávanie, automatické dopĺňanie dopytu a ďalšie.

Nástroj Elasticsearch sa vďaka svojej schopnosti efektívne spracovávať a analyzovať veľké objemy dát v reálnom čase využíva v širokom spektre odvetví a aplikácií. Jedným z najbežnejších využití je v oblasti fulltextového vyhľadávania na webových stránkach alebo v podnikových informačných systémoch. Okrem svojho využitia vo fulltextovom vyhľadávaní nachádza uplatnenie aj v analýze, spracovaní a vyhodnocovaní rôznych typov dát.

V príklade výpisu 4.1 sa vyhľadáva v indexe `books`, konkrétne prostredníctvom koncového bodu `_search`. Dotaz využíva `multi_match`, ktorý je efektívny pri hľadaní zadaného výrazu naprieč viacerými poliami dokumentu. V príklade sa teda vyhľadáva výraz *PHP* v poliach `title` a `synopsis`. Zároveň je pole `title` použitím `^3` označené ako pole s väčšou váhou, a teda dôležitosťou.

```
GET books/_search
{
  "query": {
    "multi_match": {
      "query": "PHP",
      "fields": ["title^3", "synopsis"]
    }
  }
}
```

Výpis 4.1: Ukážkový dotaz vyhľadávanie kníh v nástroji Elasticsearch

4.4 Apache Solr

Solr je vyhľadávací nástroj založený na Apache Lucene a napísaný v jazyku `Java`. Jeho schopnosť indexovať a vyhľadávať v obrovských objemoch dát je podľa [7] kľúčová pre rôzne aplikácie od webových vyhľadávačov až po komplexné dátové systémy. Spracovanie dát podporuje v rôznych formátoch, napríklad vo formátoch `XML`, `JSON`, `CSV` a iných, čo umožňuje jeho využitie v rôznych aplikačných prostrediach. Podporuje rôzne dátové typy, ako sú textové dáta, numerické, dátum a čas, geografické a mnohé ďalšie. Ponúka rôzne rozšírené funkcie, medzi ktoré patria facetové vyhľadávanie (umožňuje používateľom zúžiť výsledky vyhľadávania podľa určitých kategórií), zvýrazňovanie vo výsledkoch vyhľadávania, rôzne štatistické funkcie alebo automatické dopĺňanie dopytov.

Solr sa používa v mnohých rôznych oblastiach, od webového vyhľadávania cez akademické výskumné databázy až po veľké podnikové vyhľadávacie systémy. Svojou flexibilitou a výkonom sa teda radí medzi vyhľadávacie nástroje, ktoré je možné využiť v rôznych oblastiach, v rôzne veľkých systémoch a prostrediach.

4.5 Sphinx

Sphinx je vyhľadávací nástroj optimalizovaný pre čo najpresnejšiu relevanciu a vysoký výkon. Je napísaný v programovacom jazyku `C++`, vďaka čomu dokáže podľa Abbas Ali [1] efektívne a rýchlo spracovávať veľké množstvo dát. Jeho architektúra je navrhnutá tak, aby bola škálovateľná a prispôsobiteľná rôznym typom aplikácií, od vyhľadávania v malých lokálnych relačných databázach až po rozsiahle textové indexy. Podporuje rôzne dátové typy, ako sú text, čísla, dátum a čas, ako aj množiny hodnôt. K dispozícii sú aj pokročilé funkcie, medzi ktoré patrí váhovanie slov (umožňuje nastaviť váhu rôznym slovám v dotaze, čo zlepšuje relevanciu výsledkov), podpora booleovských operátorov, zoraďovanie podľa relevancie alebo aj prácu so synonymami slov. Okrem toho Sphinx poskytuje rozhranie pre integráciu s rôznymi programovacími jazykmi, ako napríklad `PHP`, `Python` či `Java`.

Primárne využitie nástroja Sphinx je v oblasti vyhľadávania na webových stránkach, kde umožňuje rýchle a presné vyhľadávanie obsahu. Často je používaný aj v podnikovom prostredí na rýchle vyhľadávanie v rozsiahlych databázach a uplatnenie nachádza aj vo vývoji softvérových aplikácií, kde je častokrát integrovaný pre poskytovanie efektívneho vyhľadávania.

4.6 Xapian

Xapian je nástroj pre vyhľadávanie napísaný v jazyku `C++`. Ako uvádza [12], Xapian je súčasťou všetkých hlavných `GNU` a `Linux` distribúcií a umožňuje rýchlu konfiguráciu a nasadenie. Je určený na použitie v aplikáciách, kde je potrebné vykonávať pokročilé vyhľadávania v textových dokumentoch. Využíva rôzne algoritmy na indexovanie a vyhľadávanie, ktoré sú optimalizované pre veľké objemy dát. K dispozícii sú rozhrania pre viaceré programovacie jazyky, ako sú napríklad `Python`, `Java`, `Ruby` či `PHP`. Okrem toho dokáže indexovať dáta v rôznych formátoch, ako napríklad `HTML`, `XML` a `PDF`. Súčasťou knižnice Xapian je aj dotazovací jazyk, ktorý umožňuje definovať komplexné dotazy pre vyhľadávanie. Podporuje rôzne funkcie, medzi ktoré patria facetové vyhľadávanie, synonymá, booleovské operátory či rýchle a efektívne vyhľadávanie.

Knižnica Xapian je vhodná pre vývoj vyhľadávačov na webových stránkach či informačných systémoch, kde umožňuje nájsť relevantné informácie v širokej databáze obsahu. Jeho výhodou je aj minimálna réžia na dlhodobú údržbu.

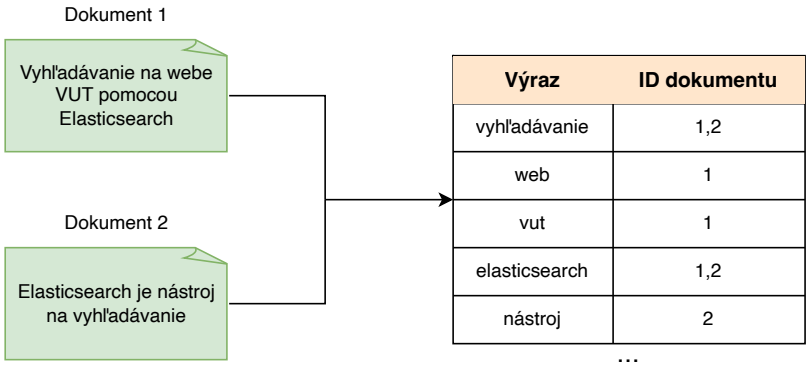
4.7 Výber technológie pre implementáciu hľadania na VUT

Pri výbere technológie pre vyhľadávanie na univerzitnej stránke VUT som zvážil a porovnal všetky technológie uvedené v tejto kapitole. Pre implementáciu komplexného univerzitného vyhľadávania, ktoré bude pracovať s veľkým množstvom dát a požiadavok, som vybral technológiu Elasticsearch. Táto technológia ponúka nielen širokú paletu funkcií, ale aj spoľahlivosť a škálovateľnosť. Lucene, na ktorom je Elasticsearch postavený, dodáva tomuto nástroju vysokú rýchlosť indexovania a vyhľadávania.

Vysoký výkon a rýchlosť

Jedným z kľúčových prvkov, ktorý prispieva k vysokému výkonu, je schopnosť efektívne indexovať dáta. Elasticsearch využíva invertovaný index, ktorý umožňuje rýchle a efektívne vyhľadávanie [6]. Invertovaný index (pozri obrázok 4.2) obsahuje kolekciu slov zo všetkých

indexovaných dokumentov a ku každému slovu uchováva zoznam, v ktorých dokumentoch je dané slovo použité. Tento spôsob ukladania dát umožňuje efektívne vyhľadávať konkrétne dáta vo veľkých súboroch a indexoch.



Obrázek 4.2: Abstraktný príklad popisujúci použitie invertovaného indexu na dvoch dokumentoch

K vysokému výkonu prispieva aj distribuovaný charakter Elasticsearch, ktorý umožňuje rozdeliť uložené dokumenty do shardov. Tieto shardy môžu byť uložené na viacerých serveroch, čo zvyšuje rýchlosť indexovania nových dokumentov a vyhľadávania.

Rýchlosť spracovávania dopytovaných dotazov na univerzitnom webe VUT, ktorý obsahuje veľké množstvo informácii, je dôležitým faktorom pre zlepšenie užívateľskej skúsenosti. Vysoký výkon zase umožní spracovávať veľké množstvo dát, ktoré budú prichádzať z centrálnej databázy VUT.

Škálovateľnosť

Nástroj Elasticsearch je navrhnutý tak, aby dokázal efektívne pracovať so zvyšujúcim sa množstvom dát. Jedným zo základných pilierov jeho škálovateľnosti je možnosť horizontálneho škáľovania, ktore umožňuje systému rásť pridaním viacerých uzlov do klastra. Pridávanie uzlov zvyšuje výkon, a teda aj rýchlosť vyhľadávnia. Klastre môžu byť dynamicky zväčšované alebo zmenšované bez prerušenia prevádzky a bez straty dát. Okrem toho podporuje aj automatické rozdelenie dát do shardov a replikáciu shardov medzi rôznymi uzlami. To zabezpečuje, že dáta sú rovnomerne rozložené a dostupné aj pri výpadku jedného alebo viacerých uzlov. Vďaka tomu je systém schopný poskytovať vysokú dostupnosť a odolnosť voči chybám.

V univerzitnom systéme VUT, do ktorého každným dňom pribúdajú nové dáta, ktoré budú indexované do vyhľadávača, je možnosť pokročilého škálovania veľmi dôležitým aspektom.

Široké možnosti vyhľadávania

Elasticsearch ponúka rôzne vyhľadávacie operácie – od základných až po vyhodnocovanie komplexných a zložitých dopytov. Ponúka rôzne typy dotazov, medzi ktoré patria fulltextové, wildcard, regulárne výrazy či aplikácie rôznych filtrov. K dispozícii sú aj rôzne možnosti zoradenia nájdených výsledkov, čo umožňuje zobraziť užívateľovi tie najrelevantnejšie výsledky. K zlepšeniu užívateľskej skúsenosti napomáha aj možnosť automatického doplňovania dotazu pri vyhľadávaní.

Komunita a dokumentácia

Veľká komunita okolo Elasticsearch poskytuje množstvo zdrojov, podpory a návodov. Komunitný prístup je otvorený a každý, či už začiatočník alebo skúsený vývojár, môže prispieť alebo sa podeliť o riešenie svojho problému. Komunita Elasticsearch sa rozrastá aj v Čechách a na Slovensku, čo je výhodou pri implementácii lokalizovaného vyhľadávania a pri vytváraní lokálnych jazykových analyzátorov. Pri implementácii pomôže aj prehľadná dokumentácia, ktorá podrobne popisuje každý aspekt systému. Dokumentácia je pravidelne aktualizovaná a udržiavaná komunitou a vývojovým tímom, čo zabezpečuje, že informácie v nej sú vždy aktuálne a presné.

Komunita a kvalitná dokumentácia sú neoddeliteľnou súčasťou úspešnej implementácie vyhľadávacieho riešenia. Prehľadná dokumentácia znižuje čas potrebný na vývoj a napomáha k tomu, že vyhľadávanie bude fungovať efektívne a spoľahlivo.

Kapitola 5

Návrh nového vyhľadávača pre VUT

Cieľom tejto kapitoly je popísať návrh implementácie nového vyhľadávača pre VUT. Kapitola najskôr opisuje spôsob komunikácie s Elasticsearch a neskôr sa venuje jeho textovým analyzátorom a indexovým šablónam. Okrem toho sa zaoberá aj návrhom synchronizácie dát a vyhľadávania v konkrétnych oblastiach informačného systému. Záver kapitoly sa zaoberá návrhom implementácie nového vyhľadávania do jednotlivých vyhľadávačov dostupných na webe VUT.

5.1 Komunikácia s Elasticsearch

Nástroj Elasticsearch poskytuje verejné aplikačné rozhranie (API), cez ktoré prebieha odosielanie vyhľadávacieho dotazu, predávanie výsledkov a indexovanie. Architektúra tohto aplikačného rozhrania je definovaná pomocou REST. Pre túto komunikáciu bude systém využívať vlastného klienta, ktorý umožňuje prispôsobenie a optimalizáciu komunikácie podľa špecifických potrieb aplikácie.

Rozhranie REST

REST je rozhranie pre distribuované prostredia orientované na dáta [8]. Rozhranie je používané pre jednotný a ľahký prístup ku zdrojom, teda dátam alebo stavom aplikácie. Všetky zdroje majú svoj vlastný jednotný identifikátor zdroja (URI). Rozhranie REST sa delí do niekoľkých vrstiev, ktoré majú svoje špecifiká a štandardy:

- **Dátová vrstva** – jedná sa o najnižšiu vrstvu, ktorá zaisťuje samotný dátový prenos. Najčastejšie sa pre prenos využíva protokol HTTP[1], ktorý poskytuje štandardizované metódy pre komunikáciu medzi klientom a serverom.
- **Zdrojová vrstva** – zaisťuje, aby neboli všetky dáta posielané na jeden hlavný bod, ale rozdeľuje ho na viaceré zdroje. Každý zdroj má teda jeden koncový bod, ktorý plní určitú úlohu.
- **Vrstva metód HTTP** – táto vrstva sa zameriava na používanie špecifických metód protokolu HTTP, ktoré definujú typ operácie vykonávanej na konkrétnom zdroji.

[1] https://datatracker.ietf.org/doc/html/rfc2616

Najčastejšie sa využívajú: *GET* – získanie dát, *POST* – vytvorenie dát, *PUT* – aktualizácia zdroja, *DELETE* – vymazanie dát.

- **Vrstva hypermediálnych ovládacích prvkov** – udáva, že aplikácia sa má spoliehať na jediný známy začiatočný koncový bod, ktorý s dátami poskytuje aj odkazy vedúce k ďalším zdrojom. Výhodou je, že klient nie je závislý na určitých adresách, ale cez centrálny bod sa postupne dostáva k požadovanému bodu. Tento prístup je známy aj pod skratkou HATEOAS, v súčasnej dobe však nie je vo väčšine prípadoch využívaný z dôvodu zložitosti na implementáciu.

Pri predávaní dát v rozhraní REST sa využívajú rôzne formáty dát, najbežnejším je však v súčasnej dobe formát JSON[2]. REST je bezstavové aplikačné rozhranie, vďaka čomu umožňuje paralelné spracovanie obsahu.

Komunikačný klient

Klient, ktorý bude v informačnom systéme VUT komunikovať s nástrojom Elasticsearch, bude pre komunikáciu protokolom REST využívať knižnicu Guzzle[3]. Táto knižnica umožňuje komunikáciu cez protokol HTTP a je v aplikáciach informačného systému využívaná už dlhodobo. Elasticsearch klient bude obsahovať inicializačnú funkciu, ktorá slúži ako centrálny bod pre inicializáciu a autorizáciu spojenia. Okrem toho bude implementovať funkcie pre jednotlivé operácie – vkladanie, vyhľadávanie, aktualizácia a mazanie dokumentov. Táto architektúra zabezpečuje, že všetky uvedené operácie budú realizované na základe konzistentného a bezpečného modelu.

Funkcie pre dátové operácie budú pracovať primárne s konkrétnym indexom Elasticsearch a vstupom vo formáte JSON, ktorý predstavuje telo danej operácie. Vzhľadom na požadovanú funkciu sa následne vykoná príslušná požiadavka protokolu HTTP. Táto požiadavka môže byť odoslaná metódou typu POST, GET alebo DELETE na koncový bod určený adresou servera, indexom, typom operácie a parametrami upresňujúcimi operáciu. Odpoveď servera je následne návratovou hodnotou predaná ďalej do miesta volania, rovnako vo formáte JSON. V prípade zlyhania požiadavku vygeneruje klient príslušnú výnimku a vytvorí logovaciu správu, čím sa zvyšuje odolnosť systému voči potenciálnym chybám v komunikácii s Elasticsearch serverom.

5.2 Vnútorná analýza textu

Elasticsearch pracuje s dátami, ktoré sú štrukturované alebo neštrukturované. Práca so štrukturovanými dátami je priama a nevyžaduje si žiadnu väčšiu réžiu na ich analýzu alebo spracovanie. Naopak, v prípade neštrukturovaných dát je potrebná hlbšia analýza, na základe ktorej dokáže Elasticsearch určiť, ako relevantný je daný dokument k požadovanému dotazu. Text je analyzovaný nie len v procese indexácie, ale aj pri vyhľadávaní. K tejto analýze sa využívajú analyzátory textu, ktoré plnia dve hlavné úlohy: tokenizáciu a normalizáciu [9].

[2] https://datatracker.ietf.org/doc/html/rfc8259
[3] https://docs.guzzlephp.org/en/stable

Tokenizácia

V procese tokenizácie sa fráza alebo veta rozdeľuje na menšie textové jednotky nazývané tokeny. K tomuto rozdeleniu dochádza na základe určitého rozdeľovača, ktorým môže byť napríklad biely znak, určitý symbol alebo vzor znakov. O tento proces sa stará tokenizér, ktorý rozdeluje frázy a vety na tokeny podľa určených pravidiel.

Pri vyhľadávaní v informačnom systéme VUT sa bude používať zväčša štandardný tokenizér, ktorý sa riadi pravidlami rozdeľovania textu Unicode Text Segmentation[4]. To zaisťuje, že frázy a vety budú korektne a jednotne rozdelené na menšie slovné jednotky, nad ktorými bude prebiehať normalizácia. Rozdielny tokenizér sa bude využívať pre spracovanie adries elektronickej pošty. Pre tento typ dát sa použije tokenizér `uax_url_email`, ktorý rozpoznáva okrem adresy elektronickej pošty aj internetové adresy a ukladá ich ako jeden ucelený token.

Normalizácia

Počas normalizácie sú tokeny z procesu tokenizácie upravované, transformované a obohacované prostredníctvom procesov ako odstraňovanie koncových prípon slov (stemming), používanie synoným a vylučovanie stop slov. Proces normalizácie je v Elasticsearch určený tokenovými filtrami. Pri vyhľadávaní v informačnom systéme budú implementované nasledovné filtre:

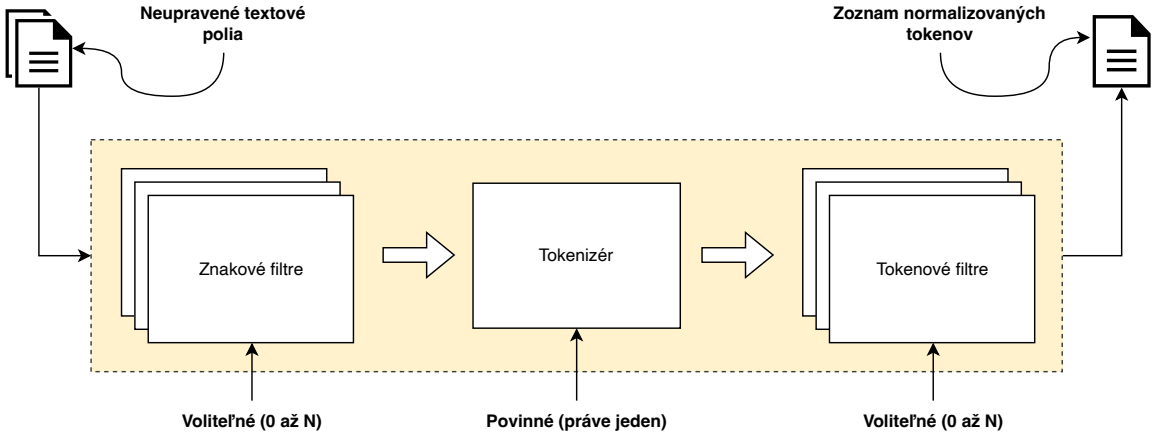
- **Filter minimálnej dĺžky** – filter odstraňuje tokeny, ktoré sú príliš krátke a neboli by prínosné pre vyhľadávanie.
- **Filter duplicít** – odstraňuje duplicitné tokeny, ktoré sa nachádzajú na rovnakej pozícii. Takáto situácia môže nastať napríklad pri normalizácii slov alebo využívaní synoným.
- **Slovníkový filter Hunspell** – tento filter poskytuje kmeňovú analýzu (stemming) tokenu s pomocou slovníka Hunspell[5]. Používa sa na normalizáciu slov podľa morfologických pravidiel daného jazyka, čím pomáha vylepšiť vyhľadávanie zohľadnením rôznych gramatických tvarov slova.
- **Stop filter** – odstraňuje bežné, málo informatívne slová (stop slová), čím pomáha zefektívniť vyhľadávanie zameraním sa na kľúčové tokeny. Okrem toho redukuje veľkosť tokenov, čím pomáha k zrýchleniu vyhľadávania.
- **Filter typu N-gram** – generuje sled N po sebe idúcich symbolov (n-gramy) od začiatku tokenu. Z každého slova teda vygeneruje niekoľko tokenov v rôznych dĺžkach, čo následne umožňuje vyhľadávanie aj na základe nekompletného slova.

Využitie týchto filtrov záleží od konkrétnej povahy dát. Každý z týchto filtrov pridáva pri analýze textu určitú réžiu, ktorá môže viesť k zníženiu výkonu indexácie alebo vyhľadávania. Z toho dôvodu je potrebné zvážiť, ktoré filtre sa v rámci analýzy konkrétnych textových polí budú využívať.

[4] https://unicode.org/reports/tr29/
[5] http://hunspell.github.io

Štruktúra analyzátorov

Procesy tokenizácie a normalizácie sú tvorené tromi hlavnými komponentami – znakovými filtrami, tokenizérom a tokenovými filtrami [9]. Ako ukazuje obrázok 5.1, tieto komponenty nadväzujú na seba a spolu vytvárajú analyzátor textu. Filtre sa v procese analýzy využívajú na vstupný text, kedy sa nazývajú znakovými alebo na už vytvorené tokeny, kedy sa označujú ako tokenové filtre.



Obrázek 5.1: Proces analýzy textu v nástroji Elasticsearch. Obrázok je inšpirovaný [9].

V procese analýzy textového reťazca (pozri obrázok 5.1) je na vstupe neupravený, surový text. Na tento text sa najskôr aplikujú znakové filtre, ktoré z neho odstraňujú nepotrebné znaky alebo reťazce znakov. Najčastejšie to bývajú prvky značkovacích jazykov, napríklad jazyka HTML. Okrem toho môže odstraňovať aj ďalšie redundantné výrazy pomocou regulárnych výrazov (regex). Použitie znakových filtrov je voliteľné a v module analyzátora ich môže byť mnoho.

Tokenizér, ako som opisoval v podsekcii tokenizácie, rozdeluje frázy alebo vety na tokeny a predáva ich do tokenových filtrov. Tokenizér musí byť v module analyzátora práve jeden. Tokenové filtre vykonávajú už spomínanú normalizáciu textu. Úroveň tejto normalizácie závisí od typu použitých filtrov. Tie sú rovnako ako znakové filtre voliteľné, avšak ich použitie zvyšuje relevanciu pri budúcom vyhľadávaní. Z toho dôvodu vznikne pre vyhľadávanie v informačnom systéme VUT viacero typov analyzátorov.

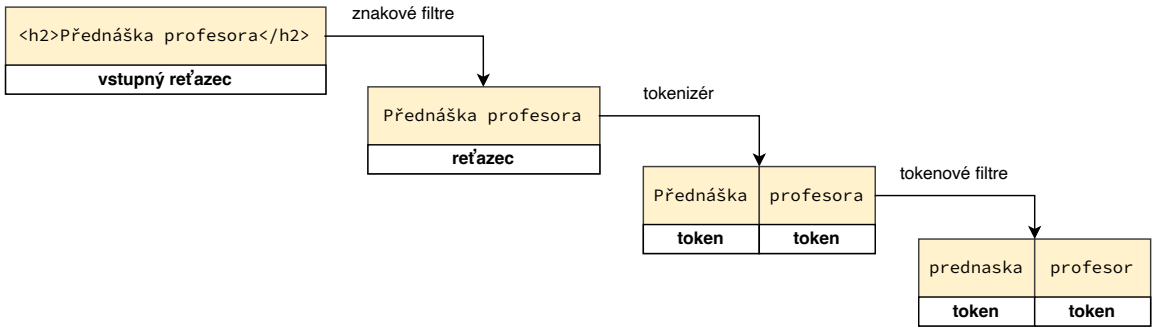
Analyzátory pre dáta informačného systému VUT

Pre potreby indexácie dát prichádzajúcich z informačného systému VUT bude implementovaných niekoľko typov analyzátorov. Tie sa budú líšiť v použitých filtroch alebo tokenizéroch a budú prispôsobené pre všetky druhy textových polí nachádzajúcich sa v databáze informačného systému. Pre české texty sa budú implementovať nasledujúce analyzátory:

- **Jednoduchý analyzátor** – určený pre jednoduchšie a kratšie texty, ktoré nevyžadujú dôslednejšiu analýzu. Používať bude filter pre prevod všetkých písmen na malé, ktorý je vstavaný v nástroji Elasticsearch a filter na odstraňovanie českej diakritiky. V tomto prípade sa nevyužije vstavaný filter, ale filter `icu_folding`, ktorý sa do Elasticsearch pridá ako rozšírenie. Ten dokáže lepšie pracovať s českým jazykom a napríklad vie, že spojenie písmen `c` a `h` za sebou tvorí jedno písmeno `ch` [14].

- **Štandardný analyzátor** – bude implementovaný pre základnú analýzu dlhších textových polí. Používať bude rovnaké tokenové filtre ako jednoduchý analyzátor, pribudne však filter pre odstraňovanie duplicitných tokenov.
- **Analyzátor so slovníkom Hunspell** – určený pre jednoduchšie, ale aj zložitejšie texty, ktoré vyžadujú dôkladnejšiu analýzu. Využíva rovnaké filtre ako jednoduchý analyzátor, ale pribudnú tokenový filter pre odstraňovanie duplicitných tokenov na rovnakej pozícii a filter využívajúci slovník Hunspell pre redukciu slova na jeho kmeňový tvar. Týmto dosahuje väčšej relevancie budúceho vyhľadávania pomocou českých dotazov.
- **Analyzátor so slovníkom Hunspell pre HTML** – určený pre širšie textové polia obsahujúce značky jazyka HTML. Využívať bude rovnaké tokenové filtre ako analyzátor so slovníkom Hunspell, pridaný však bude znakový filter pre odstraňovanie týchto značiek. Výsledkom bude analyzátor schopný správne analyzovať text, ktorý bol do centrálnej databázy uložený vo formáte jazyka HTML.
- **N-gram analyzátory** – určené pre generovanie tokenov typu N-gram, teda sekvenciu N po sebe idúcich písmen. Implementované budú dva takéto analyzátory, jeden rozvíjajúci štandardný a druhý Hunspell analyzátor. Platí, že analyzátory typu N-gram budú používať rovnaké filtre ako ich rodičia, využívať budú navyše len tokenový filter vytvárajúci postupnú sekvenciu znakov.
- **Analyzátor dlhých textov** – využívaný bude pre analýzu dlhších textov, ako sú napríklad abstrakty prác a podobne. Využíva rovnaké tokenové filtre ako analyzátor so slovníkom Hunspell, pridáva navyše ešte filter pre elimináciu zastavovacích (stop) slov. To pomáha odstrániť často používané české slová a zamerať sa tak na významnejší obsah z pohľadu vyhľadávania.

Pre analýzu anglického textu sa budú využívať podobné typy analyzátorov. Najväčším rozdielom bude prístup ku kmeňovej analýze slov (stemming). V prípade českého jazyka sa vo všetkých analyzátoroch využíva k tejto analýze slovník Hunspell, pre anglický jazyk sa však využije algoritmus s názvom Porter stem[6]. Jedná sa o rýchly a spoľahlivý algoritmus, ktorý odstraňuje koncovky v anglických slovách. Filter, ktorý využíva tento algoritmus, je priamo vstavaný v Elasticsearch a prispôsobený pre čo najrýchlejšiu analýzu.



Obrázek 5.2: Príklad analýzy textu pomocou HTML analyzátora so slovníkom Hunspell.

[6] https://snowballstem.org/algorithms/porter/stemmer.html

Príklad na obrázku 5.2 ukazuje použite analyzátora so slovníkom Hunspell pre HTML na textový reťazec. V prvom kroku sa cez znakový filter odstraňujú prebytočné značky jazyka HTML. Následne sa reťazec cez tokenizér rozdelí na tokeny `Přednáška` a `profesor`, na ktoré sa aplikujú tokenové filtre. Týmto je proces analýzy textu ukončený a normalizované tokeny `prednaska` a `profesor` sa pridajú do indexu.

Overenie funkčnosti vytvorených analyzátorov

Nástroj Elasticsearch umožňuje cez svoje aplikačné rozhranie (API) testovať a overovať funkčnosť vytvorených analyzátorov. Toto testovanie je možné cez koncový bod `_analyze`, ktorý zobrazuje výsledok analýzy textu a informuje o vytvorených tokenoch, ich type a pozícii v rámci zadaného textu. Tento koncový bod umožňuje lepšie pochopiť vnútorné spracovanie textu, jeho tokenizáciu a normalizáciu.

Príklad výpisu 5.1 ukazuje využitie koncového bodu `_analyze` pre testovanie správnej funkčnosti analyzátora. Výsledkom tejto operácie by bol zoznam normalizovaných tokenov ako vidieť na obrázku 5.2, na základe čoho by sa dalo overiť, že analyzátor vytvára a normalizuje tokeny v požadovanom tvare.

```
1 GET _analyze
2 {
3   "text": "Přednáška profesora",
4   "analyzer": "czech_hunspell_html"
5 }
```

Výpis 5.1: Príklad využitia koncového bodu `_analyze` pre testovanie analyzátora.

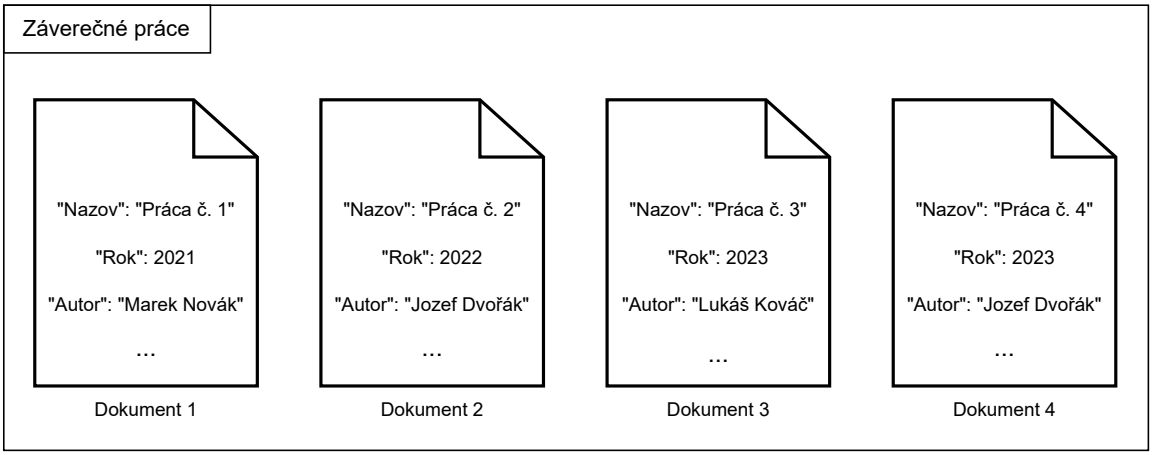
5.3 Indexácia dát

Proces prenosu a indexácie dát z centrálnej databázy do Elasticsearch je kľúčovým krokom pre správne fungovanie vyhľadávača. Cieľom tohto procesu je systematická extrakcia relevantných dát pre vyhľadávanie a ich následné uloženie do indexu.

Index v nástroji Elasticsearch je logická štruktúra s unikátnym názvom, ktorá uchováva kolekciu dokumentov (pozri obrázok 5.3). Každý dokument je tvorený poľami rôznych dátových typov. Tieto polia sú štruktúrované ako dvojice kľúč–hodnota, kde kľúč slúži ako identifikátor a hodnota obsahuje konkrétne dátové entity, ako sú textové reťazce alebo numerické hodnoty. Každý index môže byť podrobne nakonfigurovaný a prispôsobený na základe špecifických potrieb jeho aplikácie, čo zahŕňa napríklad nastavenie analýzy textu, mapovanie polí a definíciu štruktúry dát.

Indexy môžu v Elasticsearch vznikať dvomi spôsobmi – implicitne alebo explicitne. K implicitnému vytvoreniu indexu dochádza v prípade, že sa klient pokúša pridať nový dokument do neexistujúceho indexu. Elasticsearch ho v tom prípade vytvorí s predvolenými nastaveniami. Tento prístup podľa [9] funguje bez problémov, neodporúča sa však využívať v produkčnom prostredí. Môže to mať za následok, že takéto indexy budú neoptimalizované a budú negatívne ovplyvňovať chod celého systému. Pri implicitnom vyhľadávaní dochádza k dynamickému vytváraniu polí na základe vstupných dát. To môže spôsobiť problémy najmä pri určovaní dátového typu, napríklad v prípade dátumov, ktoré môže Elasticsearch chybne vyhodnotiť ako textové polia.

Explicitné, teda manuálne vytváranie umožňuje v procese vytvárania prispôsobiť index pre konkrétne dáta a použitie. Tento prístup umožňuje definovať vlastné nastavenie indexu,



Obrázek 5.3: Abstraktný príklad popisujúci index záverečných prác

jeho filtre, analyzátory a mapovanie jednotlivých polí. To prispieva k optimalizácii indexu a rýchlejšej indexácii alebo vyhľadávaniu.

Indexované dáta a vlastnosti

Dáta z centrálnej databázy budú v Elasticsearch indexované vzhľadom na ich druh a charakteristiky. V dôsledku toho vznikne v systéme viacero dedikovaných indexov, každý špecificky štrukturovaný a optimalizovaný pre jednotlivé oblasti dát opísaných v sekcii 3.1.

Do jednotlivých indexov sa nebudú indexovať všetky vlastnosti, ktoré sa k danej oblasti nachádzajú v centrálnej databáze. Indexované budú len vlastnosti potrebné pre relevantné vyhľadávanie a pre potreby filtrovania, zobrazovania výsledkov či jednoznačnej identifikácie dokumentu. Tento prístup zabezpečí, že každý index bude efektívne spracovávať a uchovávať dáta relevantné pre jeho účel, čo vedie k zníženiu celkového objemu dát v indexoch. To sa pozitívne odrazí na výkone a rýchlosti indexovania alebo vyhľadávania.

Pri každom indexe sa vytvorí mapovanie, ktoré definuje typy polí a ich atribúty, vrátane analyzátorov použitých na textové dáta. Toto mapovanie je zásadné pre správnu interpretáciu a analýzu dát, umožňujúc vyhľadávacím algoritmom Elasticsearch správne spracovať a indexovať informácie.

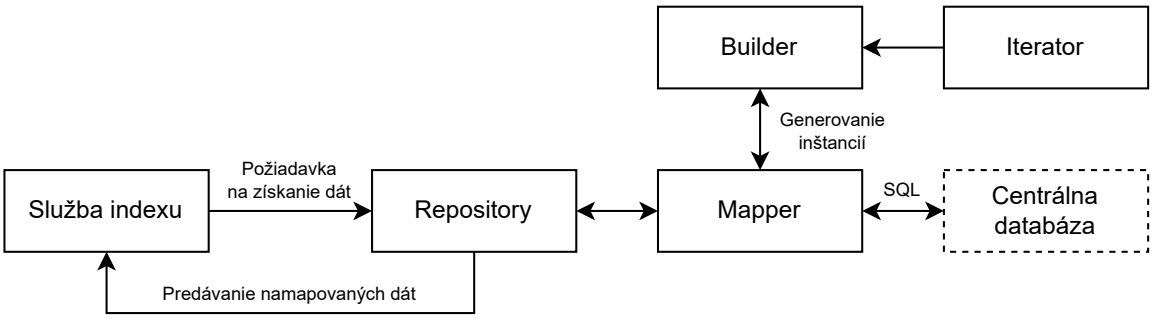
Získavanie a príprava dát

Základom pre indexáciu dát do Elasticsearch je získanie dát z centrálnej databázy. Pre zaistenie tejto úlohy sa budú využívať modely Vut2 opísané v časti 3.4 a servisné triedy, ktoré budú sprostredkovať procesy synchronizácie.

Servisné triedy reprezentujú v systéme Vut2 medzivrstvu medzi repozitárom modelu a konkrétnym kontrolérom. Obsahujú špecifické funkcie, ktoré sú opätovne použiteľné v rôznych častiach aplikácie. Tieto servisné triedy, inak nazývané aj služby, by mali obsahovať všetku aplikačnú logiku zaisťujúcu synchronizáciu dát a vyhľadávanie. Vo výsledku by mal kontrolér tieto servisné triedy inštancovať a delegovať všetko chovanie do tejto inštancie. V službe každého indexu budú funkcie pre indexovanie všetkých položiek z databázy, priebežnú synchronizáciu zmien oproti databáze a vyhľadávanie výsledkov.

Pre každú oblasť vyhľadávania bude potrebné vytvoriť vlastnú službu, ktorá bude prispôsobená povahe dát danej oblasti. Funkcie, ktoré budú zdieľané medzi všetkými služ-

bami, budú implementované v hlavnej rodičovskej servisnej triede, od ktorej budú túto funkcionalitu dediť všetci potomkovia. Ako je znázornené na obrázku 5.4, služba bude pri synchronizácii dát komunikovať s repozitárom modelu príslušnej oblasti. Ten deleguje synchronizačnú požiadavku a jej parametre na mapper, ktorý pripraví potrebný dotaz pre získanie dát z databáze v jazyku `SQL`. Zároveň požiada builder o vytvorenie novej inštancie iterátora a vykoná pripravený dotaz. Získané dáta sa potom mapujú na objektové entity, ktoré sú pridávané do kolekcie iterátoru. Po ukončení mapovacieho procesu je objekt iterátoru predaný späť do repozitára, odkiaľ sa vracia do služby.



Obrázek 5.4: Princíp získavania dát pri indexácii

Pri indexovaní všetkých dát z centrálnej databázy je potrebné postupne prechádzať dátami tak, aby nedošlo k nadmernému zaťaženiu databázového servera a nástroja Elasticsearch. Každá služba bude preto dáta indexovať po jednotlivých mesiacoch pridania daného záznamu do systému. Prestávky medzi indexáciou budú po zaindexovaných N mesiacoch, kde N bude určené a vyrátané podľa povahy daného indexu a celkového objemu dát.

Priebeh indexácie dát

Dáta získané z databázy a pripravené v iterátori bude služba jednotlivo odosielať do Elasticsearch cez pripraveného klienta popísaného v sekcii 5.1. Cez protokol HTTP budú pripravené dáta vo formáte JSON odoslané v tele požiadavku na Elasticsearch server. Po prijatí dát začne server proces indexácie. Počas indexácie sa dáta tokenizujú, analyzujú a uložia v invertovanom indexe, ktorého princíp je popísaný na obrázku 4.2. V prípade, že indexácia prebehne v poriadku, informuje server klienta o úspešnom vytvorení stavovým kódom protokolu HTTP typu `201 Created` spolu s identifikátorom vytvoreného dokumentu. Ak by pri indexovaní nastala chyba, server bude odpovedať príslušným stavovým kódom. Na základe toho bude klient adekvátne reagovať:

- `400 Bad Request` – požiadavka na indexáciu je neplatná, telo požiadavku pravdepodobne obsahuje chybné údaje alebo zle štrukturovaný formát JSON. V tomto prípade je chyba nahlásená do logov a indexácia sa preruší.
- `404 Not Found` – Požadovaný index nebol nájdený, a preto nemohol byť dokument zaindexovaný. V tomto prípade je chyba klientom nahlásená do logov a celý proces indexácie sa preruší.
- `408 Request Timeout` – odpoveď Elasticsearch servera trvala až príliš dlho. V tomto prípade sa proces indexácie pozastaví, prepne sa do stavu spánku a o niekoľko minút

sa pokúsi indexáciu obnoviť. V prípade opätovnej chyby sa proces preruší a chybová hláška sa zapíše do logov.

- `500 Internal Server Error` – na serveri došlo ku chybe. Klient sa pokúsi po určitom čase dokument znovu zaindexovať, v prípade opätovného neúspechu dochádza k prerušeniu celého procesu indexácie a chyba sa zaloguje.

Priebežná synchronizácia dát

Po počiatočnej indexácii všetkých dát je potrebné udržovať jednotlivé indexy aktuálne vzhľadom na centrálnu databázu. O túto synchronizáciu sa bude starať služba konkrétneho indexu, v ktorej bude v pravidelnom časovom intervale spúšťaná funkcia na aktualizáciu Elasticsearch indexu.

Funkcia najskôr získa z databázy položky, ktoré boli naposledy modifikované. Tento výber sa realizuje na základe času poslednej úspešnej aktualizácie. V prípade nájdenia modifikovaných položiek, systém prostredníctvom Elasticsearch klienta overuje existenciu záznamov s identickými identifikačnými číslami v špecifikovanom indexe. V prípade, že sa v indexe nájdu položky s rovnakými identifikačnými číslami, vyráta sa hash z databázových dát danej položky a z dát prichádzajúcich z indexu Elasticsearch. Tieto hashe sa porovnajú a ak sa zistí rozdiel, dochádza buď k aktualizácii dokumentu alebo jeho odstráneniu z indexu. Akcia, ktorá sa má v prípade rozdielnych hashov vykonať, sa určuje na základe nastaveného stavu položky v databáze. Pokiaľ zostáva položka v databáze platná, naznačuje to potrebu aktualizácie príslušného dokumentu v Elasticsearch a naopak, ak je položka v databáze označená za neplatnú, musí byť daný dokument odstránený z indexu Elasticsearch.

Volanie aktualizačných funkcií bude prebiehať automaticky po určitom čase za pomoci nástroja Cron[7], ktorý slúži na plánovanie spúšťania úloh v rôznych časových intervaloch. V prípade potreby bude možné kedykoľvek spustiť aktualizáciu manuálne.

Relevancia vyhľadávania

Pri budovaní vyhľadávania na stránke je dôležité vopred určiť, čo je pre koncového používateľa relevantná informácia. Ako uvádzajú Doug Turnbull a John Berryman [13], aj keď princíp väčšiny vyhľadávacích aplikácií vyzerá na prvý pohľad rovnako, každá má svoje unikátne preferencie, na základe ktorých prináša používateľovi svoje relevantné výsledky. Podľa nich je vhodné chápať Elasticsearch nie ako kompletné vyhľadávacie riešenie, ale skôr ako flexibilné vyhľadávacie rozhranie, ktoré poskytuje možnosť vyhľadávania informácií na základe kritérií, považovaných za relevantné v našom konkrétnom použití. Pri riešení otázky relevancie v kontexte vyhľadávania informácií je potrebné zvoliť tento postup:

- Identifikácia kľúčových charakteristík, ktoré opisujú obsah, užívateľa alebo vyhľadávaciu požiadavku. Tento krok sa zameriava na rozpoznanie faktorov, ktoré ovplyvňujú, čo je pre konkrétnu aplikáciu relevantné.
- Vytvorenie zoznamu kľúčových polí, ktoré sú pre daný index informačne dôležité. Tým sa zabezpečí, že vyhľadávač má prístup k potrebným informáciám pre správne hodnotenie relevancie.
- Meranie relevancie počas vyhľadávania tak, že sa výsledky ohodnocujú na základe kritérií relevancie.

[7] https://man7.org/linux/man-pages/man8/cron.8.html

- Vyváženie vplyvu rôznych kritérií na poradie výsledkov, čím sa zabezpečí, že konečné výsledky vyhľadávania budú relevantné pre konkrétneho užívateľa a jeho vyhľadávaciu požiadavku.

Pri relevancii výsledkov je potrebné zvážiť aj stav používateľa. Pokiaľ je používateľ prihlásený, systém dokáže presnejšie personalizovať výsledky vyhľadávania na základe jeho údajov. Ovplyvniť to môže napríklad radenie výsledkov podľa ústavu, v ktorom používateľ pôsobí, alebo odlíšiť informácie určené pre študentov a zamestnancov.

5.4 Indexové šablóny

Ako je opísané v sekcii 5.3, pri implementácii vyhľadávania bude potrebné pre každú oblasť vyhľadávania vytvoriť v Elasticsearch vlastný index. Ručné kopírovanie rovnakých nastavení pri vytváraní viacerých indexov by bolo neefektívne a z dlhodobého hľadiska údržby či škálovania neudržateľné. Z tohto dôvodu budú v Elasticsearch implementované indexové šablóny, ktoré uľahčia, zjednotia a čiastočne zautomatizujú proces vytvárania nových indexov. Indexové šablóny umožňujú podľa [9] vytvárať indexy podľa preddefinovaných vzorov a konfigurácií. V Elasticsearch sa indexové šablóny delia na dve kategórie:

- **Šablóny komponentov** – ako napovedá názov, tento typ šablón umožňuje vytvárať moduly, ktoré plnia určitú úlohu. Komponenty môžu definovať mapovania, konkrétne nastavenia indexu, analyzátory alebo aliasy. Sú znovupoužiteľné a možno ich aplikovať na viacero indexov. Platí, že šablóny komponentov môžu existovať samé o sebe, ale nie sú príliš užitočné, ak nie sú priradené ku kompozitným šablónam indexov.
- **Kompozitné šablóny indexov** – poskytujú celkový vzor pre vytvorenie určitého typu indexu. Skladajú sa z jednotlivých komponentov, čo umožňuje modulovať každú šablónu indexu podľa potreby, ale zachovať modularitu a dlhodobú udržateľnosť. Tento typ šablóny sa aplikuje automaticky na základe určitého vzoru znakov v názve indexu. Stačí teda vytvoriť index a ak sa názov tohto indexu zhoduje so vzorom definovanom v niektorej šablóne, táto šablóna sa na index automaticky aplikuje.

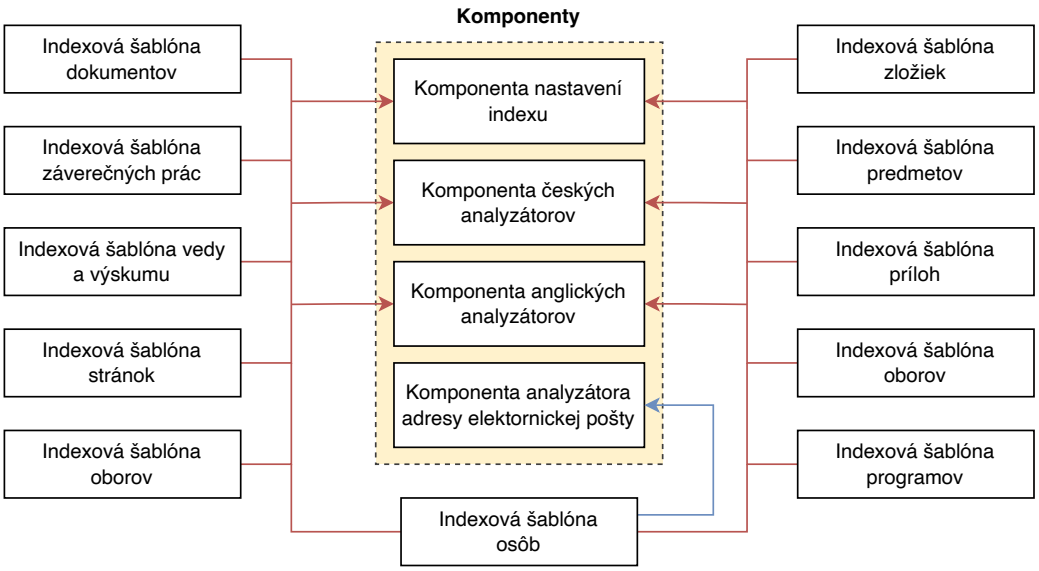
Pre zhrnutie, šablóna komponentu je znovupoužiteľný blok konfigurácie, ktorý môže byť využitý pre vytváranie celkových šablón indexov. Pre potreby vyhľadávania v informačnom systéme VUT preto bude potrebné implementovať určité komponenty a kompozitné šablóny indexov pre všetky oblasti vyhľadávania.

Návrh indexových šablón pre VUT

Pre implementáciu indexov uchovávajúcich dáta z informačného systému VUT bude potrebné vytvoriť dva typy komponentov: nastavenie indexu a definovanie analyzátorov. Rovnako ako nastavenia indexu, aj analyzátory navrhnuté v sekcii 5.2 sa budú zdieľať medzi všetkými vytvorenými indexmi. Z toho dôvodu vznikne jedna komponenta pre nastavenia indexu, medzi ktoré patria napríklad počet shardov a replík. Pre analyzátory vzniknú tri oddelené komponenty. Prvá komponenta bude obsahovať analyzátory pre český jazyk, druhá analyzátory pre anglický jazyk a posledná komponenta bude obsahovať samostatný analyzátor pre adresy elektronickej pošty.

Indexové (kompozitné) šablóny budú vytvorené pre každú oblasť vyhľadávania. Je to z dôvodu, že okrem využitia pripravených komponentov budú definovať aj mapovanie dát

a typy dátových polí. Ako zobrazuje obrázok 5.5, väčšina indexových šablón bude napojená na komponenty českých a anglických analyzátorov, a zároveň nastavenia indexu. V prípade potrebnej zmeny vo všeobecných nastaveniach alebo analyzátoroch sa vďaka tomu zmena automaticky zosynchronizuje so šablónami všetkých indexov.



Obrázek 5.5: Vizualizácia návrhu komponent a indexových šablón.

5.5 Vyhľadávanie dát

Vyhľadávanie je kľúčovou funkcionalitou nástroja Elasticsearch, ku ktorej je možné pristúpiť po úspešnom zaindexovaní dokumentov. Ako opisuje [9], Elasticsearch pracuje s dvoma variantami vyhľadávaní:

- **Štrukturované vyhľadávanie** – funguje na princípe vyhľadávania pomocou termínov a nájdeným výsledkom nepriraďuje žiadne skóre relevancie. Zameriava sa len na dáta, ktoré majú pevnú štruktúru, kde hľadá striktne presné zhody. Žiadny výsledok teda nemôže spadať do kategórie „možno“. Tento typ vyhľadávania sa využíva napríklad pri hľadaní dokumentov podľa určitého roku, fakulty alebo jazyka.
- **Neštrukturované vyhľadávanie** – zameriava na prácu s textovými dátami, ktoré nemajú jasnú a pevnú štruktúru. Výsledky vyhľadávania majú svoje skóre relevancie, ktoré určuje, ako dokument vyhovuje zadaným podmienkam. Čím bližšie sa dokument zhoduje s podmienkami a zadaným dotazom, tým vyššie skóre dostáva, a tým vyššie sa umiestňuje v rebríčku výsledkov.

V nástroji Elasticsearch sa pre vyhľadávanie dát využíva koncový bod `_search`. Pre získanie výsledkov vyhľadávania sa používa metóda GET protokolu HTTP, kde sa dotaz na vyhľadávanie predáva cez telo požiadavku. Pre formulovanie požiadavok na vyhľadávanie používa Elasticsearch doménovo špecifický jazyk (DSL). Ten zaobaľuje vyhľadávacie kritéria do formátu JSON a umožňuje tak vyskladať aj komplexné vyhľadávanie dotazy.

Doménovo špecifický jazyk nástroja Elasticsearch podporuje dva typy vyhľadávacích dotazov – listové dotazy (Leaf query) a zložené dotazy (Compound query). Listové dotazy sú základné vyhľadávacie dotazy, ktoré priamo pracujú s konkrétnymi poliami v dokumentoch. Zložené dotazy potom kombinujú listové dotazy a umožňujú tak vytvárať zložitejšie logické podmienky. Pre potreby vyhľadávania v informačnom systéme sa budú využívať oba typy dotazov.

Návrh štruktúry vyhľadávacích dotazov

Pre vyhľadávanie nad indexmi informačného systému VUT sa bude využívať tri základné typy listových dotazov: `match`, `multiple_match` a `term`. Dotaz `match` je štandardný dotaz na vykonávanie fulltextového vyhľadávania v rámci určeného poľa a vracia dokumenty, ktoré sa zhodujú so zadaným textom, číslom, dátumom alebo boolovskou hodnotou [5]. Dotaz `multiple_match` funguje na rovnakom princípe, umožňuje však vyhľadávať vo viacerých poliach dokumentu a priraďovať im rôznu váhu. `Term` vyhľadáva dokumenty na základe presnej hodnoty, používa sa teda na štrukturované vyhľadávanie.

Pre vytváranie komplexnejších dotazov sa budú využívať zložené booleovské dotazy, ktoré umožňuje kombinovať viaceré listové dotazy pomocou booleovských logických operátorov. Tieto operátory sú v Elasticsearch reprezentované nasledovne:

- **Must** – funguje ako logický operátor AND. Listové dotazy zaradené pod týmto operátorom musia vrátiť pravdivý výsledok, aby bol dokument považovaný za zhodný.
- **Should** – používa sa ako logický operátor OR s možnosťou špecifikácie minimálneho počtu zhôd. Platí, že aspoň jeden listový dotaz musí platiť, aby bol dokument považovaný za zhodný.
- **Must Not** – funguje ako logický operátor NOT. Listové dotazy pod týmto operátorom nesmú vrátiť pravdivý výsledok. Slúži na vylúčenie dokumentov, ktoré spĺňajú určité kritériá.
- **Filter** – funguje na podobnom princípe ako `must`, neovplyvňuje však skórovanie dokumentu. Vďaka tomu je efektívnejší z hľadiska výkonu, pretože nevyžaduje ďalšie výpočty relevancie. Používa sa na filtrovanie výsledkov na základe nejakej vlastnosti.

Kombináciou týchto typov dotazov bude možné implementovať všetky typy vyhľadávaní, ktoré sa nachádzajú v informačnom systéme VUT, a ich ladením dosiahnuť vysokú relevanciu nájdených výsledkov.

5.6 Všeobecný vyhľadávač

Ako opisuje sekcia 3.2, všeobecný vyhľadávač využíva na vyhľadávanie viacero dátových oblastí. Pri implementácii nového všeobecného vyhľadávania sa preto budú využívať tieto indexy: dokumenty, obory, osoby, predmety, prílohy, programy, stránky, telefóny, záverečné práce a zložky. Vyhľadávač bude pri každom dotaze súčasne prehľadávať tieto indexy a nájdené výsledky zoraďovať podľa skóre.

Práca so službami indexov

Všeobecný vyhľadávač bude osobitne využívať služby všetkých použitých indexov. Cez tieto služby prebehne príprava dotazu, vyhľadávanie v danom indexe a výsledky vyhľadávania

sa aj s prideleným skóre uložia do spoločnej kolekcie výsledkov. Po získaní výsledkov zo všetkých indexov sa v kontroléri spustí proces, ktorý zostupne zoradí kolekciu všetkých nájdených výsledkov podľa prideleného skóre. Tým sa dosiahne, že najrelevantnejšie dokumenty budú umiestnené na prvých priečkach nájdených výsledkov.

Relevancia vyhľadávania

Z dôvodu, že všeobecné vyhľadávanie bude prehľadávať až desať indexov, je dôležité určiť, ktoré polia v rámci týchto indexov budú využívané pre vyhľadávanie. Cieľom pri výbere týchto polí je zabezpečiť vysokú relevanciu výsledkov vyhľadávania, pričom je dôležité udržať ich počet na úrovni, ktorá zaručuje prehľadnosť pri ich prezeraní. Pre jednotlivé indexy sa budú využívať tieto polia:

- **Dokumenty** – pre vyhľadávanie výsledkov dokumentov sa bude využívať ich názov a upútavka. Pre dosiahnutie vyššej relevancie sa budú okrem toho tieto výsledky filtrovať podľa identifikačného čísla webu, pre ktorý je dokument určený.
- **Programy a obory** – vyhľadávanie bude prebiehať na základe polí názvu programu alebo oboru a zároveň ich skratky.
- **Osoby** – vyhľadávanie osôb bude realizované na základe mena, e-mailov osoby alebo životopisu uvedeného v informačnom systéme VUT.
- **Predmety** – pre vyhľadávanie predmetov sa bude využívať názov predmetu a jeho skratka.
- **Prílohy, zložky a stránky** – vyhľadávanie príloh dokumentov, zložiek a stránok bude možné cez ich názov. Okrem toho sa budú výsledky rovnako ako pri dokumentoch automaticky filtrovať podľa identifikačného čísla webu.
- **Telefóny** – vyhľadávanie telefónnych čísel bude možné podľa konkrétnej osoby, telefónneho čísla alebo miestnosti, ku ktorej je číslo priradené.
- **Záverečné práce** – hľadanie záverečných prác bude možné na základe ich názvu, kľúčových slov alebo abstraktu.

Používateľské rozhranie

Všeobecné vyhľadávanie umožňuje filtrovať výsledky na základe noriem, ľudí a záverečných prác. Ponúkané výsledky obsahujú základné informácie o nájdenom dokumente. V rámci implantácie vyhľadávania pomocou nástroja Elasticsearch dôjde k zmenám v zobrazovaní popisov výsledkov. Tie budú presnejšie vystihovať nájdený dokument. Okrem toho pribudne vizuálne oddelenie výsledkov podľa fakulty, z ktorej dokument pochádza. Oproti aktuálnemu vyhľadávaniu pribudne aj zoraďovanie na základe skóre, ktoré by malo používateľovi prinášať čo najrelevantnejšie výsledky na popredných pozíciach.

5.7 Vyhľadávač záverečných prác

Pri vyhľadávaní záverečných prác je potrebné vybrať kľúčové dáta, určiť relevanciu a hodnotenie výsledkov. Tento vyhľadávač bude pracovať len s indexom záverečných prác. To znamená, že príprava dotazu a vyhľadávanie bude prebiehať cez jednu službu. Služba bude okrem používateľského dotazu spracovávať aj jednotlivé filtre dostupné vo vyhľadávači.

Relevancia vyhľadávania

Pre účely vyhľadávania konkrétnej práce bude využívaný najmä jej názov. Názov práce poskytuje primárny identifikátor obsahu a umožňuje tak užívateľom rýchlo a efektívne nájsť špecifickú prácu na základe jej hlavných črtov. Abstrakt práce potom poskytuje hlbšie pochopenie obsahu práce. To umožňuje používateľom vyhľadávať práce na základe širšieho spektra kritérií, ako sú napríklad určité postupy, témy alebo technológie. Ďalším dôležitým faktorom, ktorý môže upresniť relevanciu výsledkov, sú kľúčové slová záverečnej práce. Umožňujú užívateľom presne a rýchlo vyhľadávať podľa špecifickej témy alebo konceptov.

Z pohľadu váhového hodnotenia jednotlivých vlastností je najzásadnejší názov práce. Obvykle je to hlavná položka, podľa ktorej používateľ vyhľadáva. Rovnako dôležitými sú aj kľúčové slová, ktoré vystihujú hlavné koncepty práce. Abstrakt dopĺňa informácie o práci, ktoré môžu pomôcť vyhľadávaču upresniť relevanciu výsledkov.

Vlastnosti dokumentu

Vyhľadávač zameraný priamo na záverečné práce vyžaduje pre správne fungovanie viacero zaindexovaných vlastností. Na základe týchto vlastností prebieha buď štrukturované filtrovanie výsledkov alebo vyhľadávanie na základe slovného dotazu vloženého používateľom. Pre správne fungovanie tohto vyhľadávača budú v indexe záverečných prác potrebné tieto polia:

- **Názov práce (česky a anglicky)** – vlasnosť typu `text`, ktorá bude použitá na vyhľadávanie práce podľa názvu. Pre dosiahnutie relevantných výsledkov bude využívať český analyzátor so slovníkom Hunspell. Pre anglickú variantu sa bude využívať anglický analyzátor s kmeňovou analýzou slov.
- **Abstrakt práce (česky a anglicky)** – vlastnosť typu `text` používaná na spresnenie a rozšírenie možností vyhľadávania. Použitý bude český analyzátor dlhých textov, ktorý umožní z abstraktu získanie kľúčových informácií. Pre anglickú variáciu poľa sa bude používať obdobný analyzátor v jeho anglickej verzii.
- **Kľúčové slová (česky a anglicky)** – vlastnosť typu `text` určená pre ukladanie kľúčových slov záverečnej práce. V oboch jazykových variáciách sa bude využívať štandardný analyzátor pre príslušný jazyk.
- **Meno autora, vedúceho** – Vlastnosť typu `text`, ktorá slúži na vyhľadávanie práce podľa konkrétneho autora alebo vedúceho. Využívať bude základný český analyzátor, ktorý umožní prácu aj s českými alebo slovenskými menami.
- **Identifikačné číslo práce** – neindexovaná vlastnosť typu `integer`, ktorá slúži na uchovanie jedinečného identifikátora záverečnej práce. Bude uložená v rámci dokumentu, nebude však indexovaná pre vyhľadávanie. Jej hodnota sa bude využívať len pre interné procesy porovnávania dokumentu oproti centrálnej databáze.
- **Ostatné systémové vlastnosti** – tieto vlastnosti budú zaindexované primárne pre účely filtrácie a zobrazovanie podrobnejších informácií pri výsledkoch vyhľadávania. Medzi tieto atribúty sa radí rok publikácie, jazyk dokumentu, typ akademickej práce, skratka súčasti VUT, ako aj identifikačné čísla študenta, vedúceho práce, fakulty a príslušného ústavu.

Používateľské rozhranie

Používateľské rozhranie vyhľadávania záverečných prác bude pri výsledkoch vyhľadávania zobrazovať názov práce, akademický rok, typ práce a jej autora a vedúceho. K dispozícii je aj rozsiahly filter, ktorý bude ponúkať filtrovanie podľa roku, typu práce, jazyku, ústavu alebo fakulty.

V prípade vyhľadávania záverečných prác sa používateľské rozhranie a zobrazovanie výsledkov líši pre návštevníkov stránky a prihlásených používateľov. Z dôvodu ochrany osobných údajov nemá návštevník prístup k niektorým častiam stránky, obzvlášť k vizitke študentov. Preto sa vo výsledkoch vyhľadávania môže zobrať meno autora práce, presmerovanie na vizitku však musí byť prístupné len prihláseným používateľom.

5.8 Vyhľadávač vedy a výskumu

Vyhľadávanie výsledkov vedy a výskumu je pre univerzitné prostredie dôležitým zdrojom informácií. Pre uľahčenie procesu získania konkrétnych vedeckých publikácií bude implantovaný vyhľadávač, ktorý bude navrhnutý s cieľom uľahčiť prístup k vedeckým poznatkom. Pri vyhľadávaní dát vedy a výskumu je dôležité zaistiť aj to, aby návštevník dostal požadované informácie v čo najkratšom čase.

Relevancia vyhľadávania

Pre dosiahnutie relevantného vyhľadávania výsledkov vedy a výskumu bude kľúčovým prvkom názov dosiahnutého výsledku, ktorý slúži ako hlavný užívateľský identifikátor dokumentu. Ďalším dôležitým faktorom je abstrakt výsledku, na základe ktorého vie vyhľadávač zanalyzovať širší kontext dokumentu a priniesť tak relevantnejšie informácie na dopyt. K nájdeniu požadovaného výsledku vedy a výskumu ďalej môžu pomôcť kľúčové slová, autor práce, rok vydania alebo možnosť filtrácie podľa typu výsledku.

Pri váhovaní výsledkov bude mať najväčšiu váhu názov výstupu, ktorý dokáže najlepšie vystihnúť podstatu danej vedeckej činnosti. K doplnení názvu sa budú využívať kľúčové slová, ktoré pomôžu upresniť vyhľadávanie a priniesť tak čo najrelevantnejšie výsledky. K širšiemu doplneniu výsledkov sa bude využívať abstrakt, ktorý pomôže rozšíriť možnosti vyhľadávania pri menej presnom alebo nejednoznačnom zadaní dotazu, čím umožní objaviť aj tie vedecké výstupy, ktoré by inak mohli zostať prehliadnuté.

Vlastnosti dokumentu

K správnemu fungovaniu vyhľadávania vo výsledkoch vedy a výskumu je potrebné definovať kľúčové polia a zaistiť ich správnu textovú analýzu. Okrem toho je dôležité indexovať aj dáta, ktoré by mohli byť využité k bližšej filtrácii nájdených výsledkov:

- **Názov výstupu (česky a anglicky)** – vlastnosť typu `text`, ktorá bude obsahovať názov daného vedeckého výstupu. Analyzovaný bude s ohľadom na kratší rozsah pomocou českého analyzátora so slovníkom Hunspell. Pre anglickú variantu bude použitý obdobný anglický analyzátor s kmeňovou analýzou slov.
- **Abstrakt (česky a anglicky)** – rozsiahlejšie pole typu `text` obsahujúce abstrakt výsledku. Vzhľadom na zložitosť textu bude použitý český analyzátor dlhých textov, ktorý bude z textu extrahovať dôležité informácie pre vyhľadávanie. Anglický verzia bude používať rovnaký analyzátor v anglickej podobe.

- **Kľúčové slová (česky a anglicky)** – zoznam kľúčových slov typu `text`, ktoré umožňujú spresniť výsledky vyhľadávania. Pre analýzu kľúčových slov sa bude využívať štandardný analyzátor pre český aj anglický jazyk.
- **Autor** – autor výstupu dátového typu `text`, analyzovaný českým štandardným analyzátorom.
- **Ostatné vlastnosti** – tieto vlastnosti sú indexované pre ďalšie možnosti filtrácie alebo pomáhajú systému so synchronizáciou. Patria tam identifikačné číslo výsledku, patentové označenie, rok a typ publikácie a vydavateľ.

Používateľské rozhranie

Vyhľadávanie vedy a výskumu má vlastný oddelený vyhľadávač, ktorý poskytuje rôzne možnosti filtrácie a ponúka prehľadný zoznam výsledkov. Pri každom výsledku sa zobrazí celý názov, typ dosiahnutého výsledku, rok publikácie a vygenerovaná citácia.

Vyhľadávať bude možné cez názov výsledku, autora, vydavateľa alebo aj číslo patentu. Presnejšie filtrovanie výsledkov je možné na základe typu dosiahnutého vedeckého výsledku a roku publikácie.

5.9 Vyhľadávač predmetov

Vyhľadávanie predmetov umožňuje študentom VUT alebo záujemcom o štúdium prehľadávať vyučované predmety. Vyhľadávač by mal poskytovať možnosť nájsť relevantné informácie o jednotlivých predmetoch a umožniť ich filtrovať podľa rôznych požiadaviek študenta alebo návštevníka stránky. Vyhľadávač bude pracovať primárne so službou predmetov, cez ktorú sa budú spracovávať nielen textové dotazy na vyhľadávania, ale aj požadované filtre.

Relevancia vyhľadávania

Pri procese vyhľadávania predmetov je dôležitým aspektom presné identifikovanie predmetov na základe ich názvu. Pre zvýšenie presnosti a relevancie výsledkov je rovnako dôležité zahrnúť do vyhľadávacieho procesu aj dodatočné informácie, ako sú napríklad skratka predmetu, ktorá sa medzi študentami aj vyučujúcimi často používa namiesto celého názvu predmetu. Tieto informácie poskytnú vyhľadávaču presnejší kontext pre vyhľadávanie daného predmetu.

Hodnotenie výsledkov bude klásť dôraz na zhodu s názvom predmetu, avšak vyhľadávač bude zisťovať relevantnosť dotazu k skratkám predmetov. Spresniť vyhľadávanie pomôže napríklad aj výber semestra, v ktorom je daný predmet vyučovaný, alebo fakulta, pod ktorou je predmet vyučovaný.

Vlastnosti dokumentu

Pre zabezpečenie spoľahlivého a presného vyhľadávania predmetov je nevyhnutné indexovať rôzne atribúty, s ktorými bude môcť vyhľadávač pracovať a ktoré umožnia ďalšie filtrovanie nájdených výsledkov. Pri vyhľadávaní predmetov na VUT bude dôležité v Elasticsearch ukladať tieto špecifické vlastnosti:

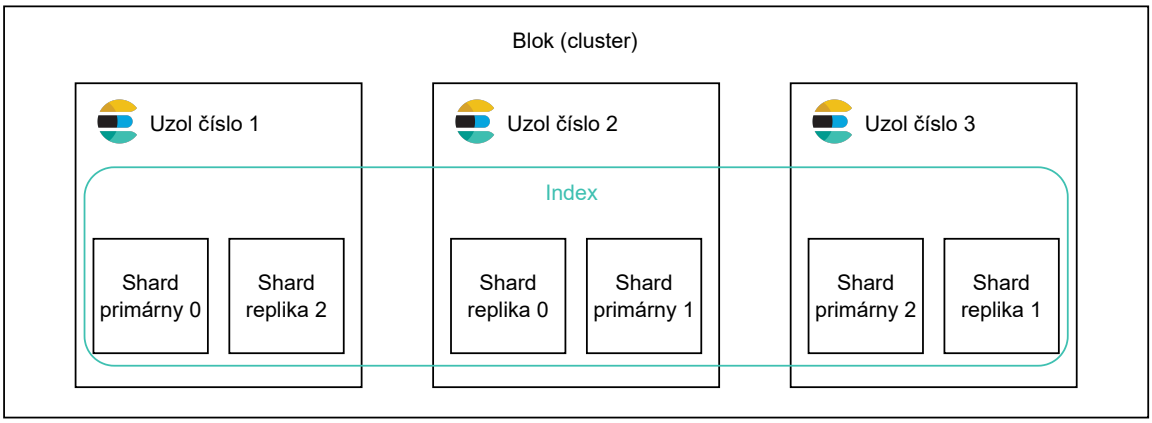
- **Názov predmetu (česky a anglicky)** – vlastnosť typu `text`, ktorá bude obsahovať názov daného predmetu. Vzhľadom na kratšiu dĺžku textového reťazca bude pre

analýzu využitý český analyzátor so slovníkom Hunspell. Pre anglickú verziu bude anglický analyzátor s kmeňovou analýzou slov.

- **Skratka predmetu (česky a anglicky)** – pole typu `text`. Použitý bude štandardný český a anglický analyzátor pre správnu normalizáciu skratky.
- **Garant predmetu** – pole pre vyhľadávanie, filtrovanie a zobrazovanie vo výsledkoch vyhľadávania typu `text`. Pre textovú analýzu sa bude využívať český analyzátor so slovníkom Hunspell.
- **Ostatné filtrovacie vlastnosti** – pre potreby filtrovania výsledkov budú v dokumentoch indexované aj identifikačné číslo fakulty, semester, akademický rok či jazyk výuky. Okrem toho je potrebné uchovávať príznak pre predmety, ktoré sú určené pre zahraničných študentov.
- **Ostatné systémové vlastnosti** – Pre potreby správnej identifikácie a synchronizácie dát s centrálnou databázou je potrebné uchovávať v dokumente aj identifikačné číslo predmetu.

5.10 Architektúra a distribúcia

Informácie v tejto sekcii sú čerpané z [6]. Architektúra Elasticsearch je navrhnutá s dôrazom na škálovateľnosť a odolnosť voči rôznym zlyhaniam, čo je dosiahnuté prostredníctvom implementácie fragmentov (shards), replík a uzlov. Fragmenty sú softvérové komponenty, ktoré udržiavajú dáta, vytvárajú pomocné dátové štruktúry (ako napríklad invertovaný index) a spravujú prichádzajúce dotazy. Pri vytváraní nového dokumentu dochádza k jeho umiestneniu do niektorého z dostupných fragmentov daného indexu. Ako znázorňuje obrázok 5.6, každý vytvorený index môže byť rozdelený do N rôznych fragmentov. Tieto fragmenty sú distribuované po celom bloku (cluster) k dosiahnutiu dostupnosti a prevencie voči zlyhaniam.



Obrázek 5.6: Princíp delenia indexu do fragmentov (shards), replík a uzlov (nodes).

Repliky sú kópiami primárnych fragmentov (shards) a slúžia ako mechanizmus pre zabezpečenie vysokej dostupnosti a odolnosti. V prípade výpadku primárneho fragmentu môže byť replika okamžite povýšená na primárny fragment a Elasticsearch ju automaticky redistribuuje medzi dostupné uzly. Repliky sú využívané aj pri vysokej záťaži hlavného fragmetu,

kedy dokážu obsluhovať požiadavky pre čítanie. To umožňuje nástroju Elasticsearch rozložiť vyhľadávacie operácie na viacero uzlov a tým zvýšiť rýchlosť vyhľadávania. Platí, že repliky sa musia nachádzať len na iných uzloch ako je umiestnený hlavný fragment, aby sa zachovala odolnosť systému proti zlyhaniu a zabezpečila sa distribúcia záťaže.

Uzly sú jednotlivé inštancie nástroja Elasticsearch, ktoré obsahujú fragmenty a repliky. Viacero uzlov vytvára jeden blok (cluster). To umožňuje distribúciu dát a záťaže, čo je v prípade veľkého objemu dokumentov dôležitý faktor. Vzhľadom na veľký objem dát, ktorými informačný systém VUT disponuje, by rozdelenie indexov na menšie časti mohlo zvýšiť rýchlosť vyhľadávania a pridať odolnosť voči výpadkom. Počet fragmentov a replík, ktoré majú byť vytvorené, závisí od vlastnosti konkrétneho indexu a počtu indexovaných dokumentov.

5.11 Nástroj Kibana

Kibana[8] je viacúčelová webová konzola, ktorá sa využíva pre vykonávanie dotazov nad Elasticsearch, vizualizáciu dát a optimalizáciu vyhľadávania. Tento nástroj je súčasťou platformy Elastic Stack, ktorá sa zameriava na vyhľadávanie, analyzovanie a vizualizáciu dát v reálnom čase. Kibana umožňuje priamu komunikáciu s Elasticsearch cez REST a prináša tak možnosť ladiť alebo bližšie analyzovať dotazy v doménovo špecifickom jazyku.

Pre účely implementácie nového vyhľadávania pre informačný systém VUT sa bude tento nástroj využívať najmä pre vývoj jednotlivých indexov, analýzu vyhľadávacích dotazov a monitorovanie stavu nástroja Elasticsearch a jeho súčastí. Dôležitú úlohu bude teda zohrávať najmä pri:

- **Vytváranie indexových šablón** – Kibana v rámci svojich vývojárskych nástrojov uľahčuje proces vytvárania jednotlivých indexových komponentov opísaných v sekcii 5.4 a následne aj použitie týchto komponent v šablónach indexov. Vytvorené komponenty sú v Kibane prehľadne vypísané a pri vytváraní indexových šablón je možné tieto komponenty cez používateľské rozhranie priradiť k danej šablóne. K dispozícii je aj rozhranie pre vytváranie mapovania polí. Okrem toho Kibana poskytuje možnosti pre testovanie šablón, čo umožňuje uistiť sa, že sú správne nastavené, čo minimalizuje riziko chýb pri tvorbe nových indexov.
- **Testovanie a ladenie vyhľadávacích dotazov** – vývojové prostredie nástroja Kibana sa bude využívať pre testovanie relevancie vyhľadávacích dotazov, ktorého cieľom je doladiť tieto dotazy k dosiahnutiu najlepších výsledkov. V toto ladenie sa bude využívať vývojárska konzola, ktorá umožňuje priamo v nej písať dotazy v doménovo špecifickom jazyku nástroja Elasticsearch a okamžite vidieť ich výsledok.
- **Ladenie analyzátorov** – ako opisujem v sekcii 5.2, Elasticsearch umožňuje overiť funkčnosť jednotlivých analyzátorov. Toto overovanie a prípadne dolaďovanie bude prebiehať cez vývojársku konzolu nástroja Kibana, vďaka čomu bude možné okamžite analyzovať výsledok jednotlivých zmien.

Nástroj Kibana bude tvoriť dôležitú súčasť implementácie návrhu nového vyhľadávania v informačnom systéme VUT a značne uľahčí prácu pri konfigurácii nástroja Elasticsearch a jeho indexov. Okrem zjednodušenia procesu konfigurácie poskytne Kibana aj širšie možnosti správy indexov a analýzy uložených dát.

[8] https://www.elastic.co/kibana

Kapitola 6

Implementácia riešenia

Táto kapitola popisuje implementáciu navrhnutého riešenia pre vyhľadávanie na univerzitnom webe VUT. V úvode sa venuje práci s dátami a vytváraniu štruktúr potrebných pre synchronizáciu dát a vyhľadávanie. Ďalej sa venuje integrácii nového vyhľadávania cez Elasticsearch s jednotlivými vyhľadávačmi dostupnými vo verejnej časti informačného systému. Kapitola sa zameriava najmä na postupy a princípy, ktoré boli využité pri implementácii návrhu.

6.1 Operácie s dátami

Pre operácie nad dátovou vrstvou sa využívajú dátové a servisné modely Vut2. Pre potreby vyhľadávania vznikli v hlavných oblastiach Base, Studium, Provoz a Systém nové modely „Vyhledávání“ alebo boli doplnené už existujúce modely o potrebnú logiku alebo vlastnosti. Jedná sa konkrétne o tieto podoblasti:

- **Dokument, osoba** – podoblasti spadajúce pod hlavnú oblasť `Base`. Podoblasť dokument implementuje nielen dokumentové modely, ale aj modely zložky a prílohy. V rámci nej boli aktualizované už existujúce modely a novo implementované boli len servisné triedy. V podoblasti osoby bol implementovaný zvlášť model pre vyhľadávanie a príslušná servisná trieda.
- **Telefónna ústredňa** – podoblasť patriaca pod hlavnú oblasť `Provoz`. Táto podoblasť novo vznikla spolu s príslušným modelom a servisnou triedou.
- **Záverečné práce, aktuálny predmet, obor, program** – podoblasti patriace pod hlavnú oblasť `Studium`. V podoblasti aktuálenho predmetu, oboru a programu došlo k prispôsobeniu existujúcich modelov, ku ktorým boli vytvorené servisné triedy. V záverečných prácach došlo k implementácii vlastného modelu pre vyhľadávanie a k nemu príslušnej servisnej triedy.
- **Systém pre správu obsahu** – pooblasť patriaca pod hlavnú oblasť `System`. V rámci tejto podoblasti sa nachádza model pre stránky, ktorý bol prispôsobený pre účely vyhľadávania. K nemu bola implementovaná nová servisná trieda.
- **Výsledok** – podoblasť nachádzajúca sa pod hlavnou oblasťou vedy a výskumu. Existujúci model bol doplnený o vlastnosti potrebné k vyhľadávaniu a bola k nemu vytvorená príslušná servisná trieda.

Každý model má v systéme pevne určený menný priestor, ktorý umožňuje využívať mechanizmus nazývaný autoloading. Ten automaticky načíta všetky potrebné súbory modelu, bez potreby manuálneho prepájania jednotlivých častí. Pre správne načítanie musí byť model umiestnený v nasledujúcom mennom priestore: `Vut2/{hlavna_oblast}/{podoblast}/[Command|Enum|Model|Service]`. `Command` je určený pre definíciu konzolových príkazov nad daným modelom, `Enum` obsahuje modely entít pre číselníky a `Model` obsahuje konkrétne dátové modely. Menný priestor `Service` je určený pre definíciu business logiky nad dátovými modelmi.

6.2 Implementácia vnútornej štruktúry Elasticsearch

V rámci nástroja Elasticsearch boli implementované indexové šablóny podľa návrhu v sekcii 5.4. Pre implementáciu šablón komponentov sa v rámci Elasticsearch využíval koncový bod `_component_template`. Vytvorené komponenty implementovali najmä jednotlivé filtre, umiestnené v rámci parametru `filter`, a analyzátory pre spracovávanie textu, umiestnené v rámci parametru `analyzer`. Okrem toho vznikla aj komponenta pre nastavenia indexov, ktoré sú zadané pod parametrom `index`.

Pre každú oblasť vyhľadávania vznikli aj kompozitné šablóny indexov, ktoré využívajú pripravené komponenty a okrem toho definujú mapovanie jednotlivých polí. Pre vytváranie bol využitý koncový bod `_index_template`. Pre schopnosť automatickej aplikácie šablóny pri vytvorení indexu sa musel v každej šablóne nastaviť vzor názvu. Na základe tohto vzoru šablóna aplikuje na rovnomenný index všetky vytvorené nastavenia.

Všetky vytvorené štruktúry majú predponu (prefix) nastavenú na `vut_`, čím sa odlišujú od systémových štruktúr nástroja Elasticsearch.

6.3 Elasticsearch klient

Pre účely vkladania dokumentov do indexov, následnú pravidelnú synchronizáciu a vyhľadávanie bol implementovaný klient, ktorý zastrešuje komunikáciu cez aplikačné rozhranie Elasticsearch. Pre vytvorenie spojenia protokolom `HTTP` sa využíva knižnica `Guzzle`[1], kde klient najskôr inicializuje spojenie a nastaví autentizačné či ostatné hlavičky. Pre dosiahnutie dôveryhodnej komunikácie a integrity dát sa využíva komunikácia cez protokol `HTTPS` a autentifikácia na základe API kľúča, ktorý je bezpečne uložený v konfiguračných súboroch informačného systému VUT. Telo požiadavku sa nastaví na základe požadovanej operácie a parametrov predaných od konkrétnej volajúcej strany. V prípade úspešnej operácie je výsledok predaný späť do miesta volania alebo v prípade chyby je vyvolaná výnimka informujúca o podstate chyby.

Pri implementácii bol využitý návrhový vzor fasáda, kde klient abstrahuje zložitosť komunikácie s Elasticsearch a poskytuje jednoduché rozhranie pre služby indexov alebo mapper konkrétnych modelov (pozri sekcia 6.1). Klient je umiestnený v systémovom priečinku, ktorý je súčasťou rodičovského adresára `_base`, aby bol prístupný vo všetkých častiach a aplikáciach informačného systému.

[1] https://docs.guzzlephp.org

6.4 Indexové služby

Každý index má svoju službu, ktorá je zodpovedná za synchronizáciu jeho obsahu medzi databázou a Elasticsearch. Okrem toho služba na mieru zostavuje dotazy `Elasticsearch Query DSL`[2] vo formáte `JSON`. Každá služba je špecificky prispôsobená povahe dát obsiahnutých v indexe, všetky však ponúkajú rovnaké rozhranie. Z tohto dôvodu sa využíva rodičovská abstraktná trieda, ktorá pevne definuje aplikačné rozhranie a implementuje zdieľané metódy. Tým sa zabezpečí, že všetky implementácie služieb budú mať rovnaké vonkajšie rozhranie, hoci ich vnútorná logika a spôsob práce s dátami môžu byť odlišné.

Synchronizácia obsahu

Služba implementuje dva spôsoby synchronizácie dokumentov medzi databázou a Elasicsearch – úplná synchronizácia a čiastočná synchronizácia (pozri sekcia 5.3). Úplná synchronizácia najskôr získa z databáze informáciu o najstaršom zázname a následne si pripraví iterácie po jednotlivých mesiacoch až do aktuálneho dátumu. To sa realizuje cez vstavanú triedu `DatePeriod`, ktorá na základe zvolenej konfigurácie vracia iterátor mesiacov v určenom rozsahu. Tento iterátor sa iteruje cyklom `foreach` a postupne cez dátový model získava jednotlivé dokumenty a indexuje ich do Elasticsearch. Dokumenty sú teda týmto cyklom z databázy vyberané na základe systémového stĺpca `UPD_TS`, ktorý značí dátum ich vloženia alebo poslednej aktualizácie.

Čiastočná synchronizácia prebieha na podobnom princípe ako úplná, iterácie sa však pripravujú od poslednej úspešnej synchronizácie. Tento typ synchronizácie okrem indexácie nových dokumentov pokrýva aj aktualizáciu už existujúcich. K tomu som implementoval aplikačnú logiku, ktorá najskôr skontroluje, či sa dokument s určitým identifikačným číslom už nachádza v indexe Elasticsearch. V prípade úspešného nájdenia sa pomocou hashovacej funkcie `MD5` vytvorí hash dokumentu, ktorý sa porovná oproti hashu pochádzajúcemu z databázových dát. Hash sa vyrátava zo serializovaného poľa dát, ktorého štruktúra je definovaná v abstrakte dátového modelu. Tým sa zabezpečuje konzistentné usporiadanie položiek poľa, aby bolo možné výsledné hashe spoľahlivo porovnať. Ak sa tieto hashe líšia, je potrebné zistiť, či nedošlo k zmene statusu záznamu. Status záznamu v centrálnej databáze značí, či je záznam stále platný alebo bol zmazaný. Pri zistení nezmenenej platnosti dôjde k aktualizácii údajov alebo v opačnom prípadne odstráneniu dokumentu z indexu Elasticsearch.

Vzhľadom na podobnosť procesov úplnej a čiastočnej synchronizácie dokumentov som v implementácii vytvoril spoločnú metódu, ktorá zastrešuje vkladanie, aktualizáciu a vymazávanie dokumentov pre oba typy synchronizácií. Tento prístup umožňuje jednotné spracovanie dát a zabezpečuje znovupoužiteľnosť implementácie. Metóda pracuje s rozhraním Elasticsearch klienta, ktorého implementáciu opisujem v sekcii 6.3.

Príprava vyhľadávacieho dotazu

Služba, ako medzivrstva medzi užívateľským rozhraním a dátovým úložiskom, získava užívateľskú požiadavku na vyhľadávanie priamo z formulára. Požiadavka je predávaná kolekciou hodnôt, ktorá okrem vyhľadávaného výrazu obsahuje aj jednotlivé filtre. Na vytvorenie dotazu pre Elasticsearch sa v implementácii využívajú booleovské dotazy (pozri 5.5), ktoré

[2] https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html

umožňujú kombináciu viacerých podmienok v jednom dotaze na základe logických operátorov.

Pre hlavný vyhľadávaný výraz sa používa dotaz typu `multi_match`, ktorý umožňuje vyhľadávanie v niekoľkých poliach dokumentu naraz. Tento dotaz je súčasťou klauzule `must`, čím sa zabezpečuje, že výsledky musia obsahovať zadaný vyhľadávací výraz v jednom alebo viacerých špecifikovaných poliach, aby boli považované za relevantné zhody. Pre dosiahnutie čo najrelevantnejších výsledkov sa v dotaze `multi_match` používajú nasledujúce parametre:

- `query` – predstavuje vyhľadávací výraz, ktorý sa má nájsť v dokumentoch, špecifikuje slovo alebo frázu, na základe ktorej sa vyhľadávanie vykonáva.
- `fields` – určuje polia v dokumente, v ktorých sa vyhľadávací dotaz aplikuje a každému nastavuje špecifickú váhu pre najpresnejších výsledkov.
- `operator` – určuje logický operátor, ktorý sa použije medzi jednotlivými slovami zadanej frázy. Pre dosiahnutie väčšej relevancie sa v mojej implementácii využíva výhradne operátor `AND`.
- `fuzziness` – umožňuje implementáciu aproximálneho vyhľadávania založeného na editačnej vzdialenosti zadaného výrazu. V implementácii je konštantne nastavená na hodnotu `AUTO`, na základe ktorej sa editačná vzdialenosť dorátava podľa dĺžky hľadaného slova.
- `analyzer` – definuje, aký analyzátor sa má použiť na spracovanie hľadaného výrazu pred spustením samotného vyhľadávania.

Filtre, ktoré upresňujú výsledky vyhľadávania, sú pridávané do klauzuly `filter`. V tejto časti sa využívajú dva typy dotazov – `term` a `match`. `Term` je základný typ dotazu, ktorý sa používa na vyhľadávanie dokumentov obsahujúcich presné a konkrétne termíny. To sa využíva najmä pri filtrovaní podľa určitej fakulty, akademického roku alebo jazyka. `Match` je flexibilnejší pri filtrovaní na základe textu, nakoľko využíva aj textové analyzátory. Tento typ filtrácie sa využíva najmä pri filtrovaní podľa určitých osôb.

Priebeh získania výsledkov

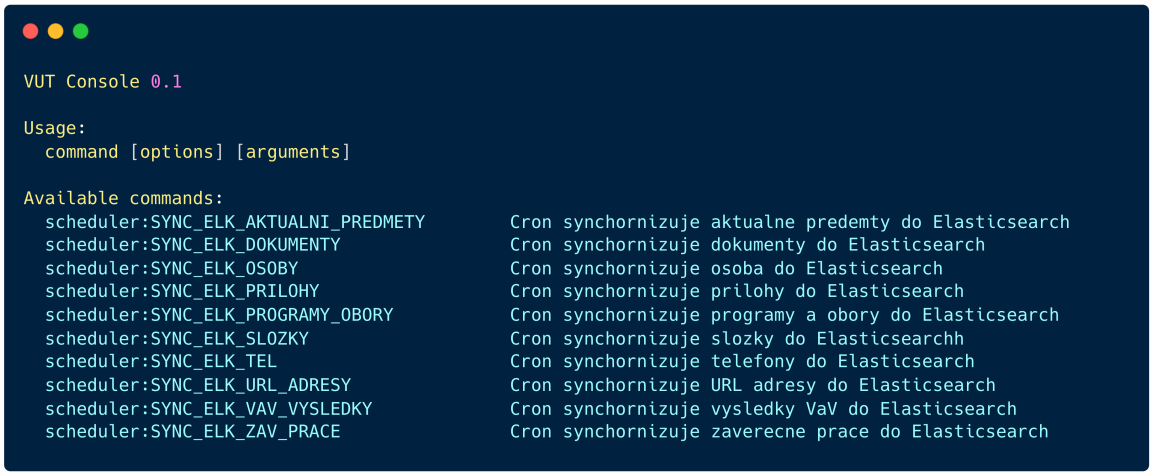
Pripravený dotaz je ako argument metódy predávaný do triedy `Repository` daného dátového modelu a odtiaľ do triedy `Mapper`. Trieda `Mapper` cez Elasticsearch klient vytvorí spojenie so serverom a získava odpoveď na požiadavok. Nájdené výsledky sa nachádzajú v odpovedi v časti `hits`. Táto časť obsahuje zoznam nájdených dokumentov, ich identifikačné čísla, získané skóre každého dokumentu a ich vlastnosti. Vlastnosti výsledkov sa mapujú na konkrétne modely. Do modelu pridá `Mapper` aj informáciu o pridelenom skóre, ktoré slúži ku konečnému zoradeniu výsledkov na používateľskom rozhraní. Všetky výsledné modely sú pridané do iterátora a v tejto forme sa predávajú na vykreslenie.

6.5 Implementácia príkazov

Príkazy pre príkazový riadok sú implementované návrhovým vzorom `Command`. Príkazy obsluhujúce jednotlivé indexy sú umiestnené spolu so servisnými triedami v konkrétnych podoblastiach systému v mennom priestore `Command`, čo umožňuje ich automatické načítanie. Abstraktná trieda príkazu je implementovaná cez knižnicu Symfony Console[3], ktorá

[3] https://symfony.com/components/Console

ponúka rozhranie pre vykonanie príkazov v prostredí príkazového riadku (CLI). Nad týmto rozhraním je vytvorená abstraktná trieda `ScheduledCommand`, ktorá pridáva metódy pre monitorovanie procesu a zasielanie logovacích správ. Konkrétne koncové príkazy potom implementujú túto abstraktnú triedu.



Obrázek 6.1: Ukážka konzolového rozhrania poskytovaného knižnicou Symfony Console s vytvorenými synchronizačnými príkazmi

V príkaze sa prepísaním vlastnosti nastavuje meno, ktoré má podľa konvencií zaužívaných v informačnom systéme VUT predponu `scheduler`, a krátky popis vytvoreného príkazu. Každý synchronizačný príkaz vyžaduje konzolový argument, ktorý určuje zvolený typ synchronizácie. Príjemca príkazu je servisná trieda konkrétneho modelu, ktorá na základe zvoleného typu synchronizácie vykoná požadovanú operáciu. Príjemca po vykonaní príkazu vracia odosielateľovi, teda príkazovému riadku, štatistiky o vytvorených, upravených a odstránených dokumentoch.

Automatické spúšťanie prebieha cez nástroj `Cron`, ktorý je na webovom serveri používaný na plánovanie úloh. Pridávanie automatizovaných procesov prebieha cez administráciu v systéme Apollo, ktorý na základe názvu príkazu a intervalu opakovania zaregistruje do tabuľky úloh prispôsobený skript v jazyku `PHP`. Synchronizácia každého indexu je v implementácii nastavená na spúšťanie každých 5 minút.

Okrem automatického spúšťania v určených časových intervaloch je možné synchronizáciu spustiť aj manuálne cez rozhranie konzolového riadku (pozri obrázok 6.1). To je možné dosiahnuť príkazom `php bin/console scheduler:{nazov_indexu} [all|sinceLastSync]`.

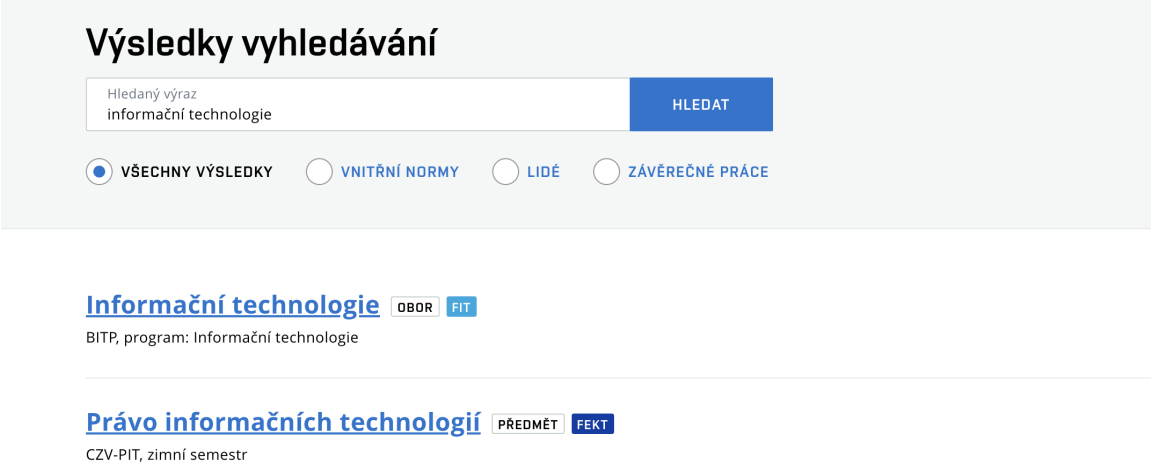
6.6 Integrácia vyhľadávania do používateľského rozhrania

Pri implementácii vyhľadávania cez Elasticsearch bolo potrebné nahradiť pôvodný systém vyhľadávania novým. Okrem zdroja dát bolo potrebné zmeniť aj spôsob vykresľovania či získavania filtrovacích možností a integrovať do pohľadu prácu s objektovou reprezentáciou výsledkov vyhľadávania. Implementácia nového vyhľadávania pracuje vo všetkých častiach výhradne so servisnými triedami jednotlivých indexov, do ktorých sa zasiela vyhľadávaný výraz, filtrovacie dáta a informácie o stránkovaní alebo počte výsledkov na jednu stránku používateľského rozhrania.

Hlavné vyhľadávanie

Hlavné vyhľadávanie sa vykonáva postupne po jednotlivých oblastiach vyhľadávania. V pôvodnej implementácii bol mechanizmus vyhľadávania priamo integrovaný do používateľského rozhrania, `SQL` operácie sa teda vykonávali priamo z pohľadu. V novej implementácii som všetky databázové operácie realizované priamo v pohľade nahradil použitím servisných tried, čím som dosiahol oddelenie dátovej vrstvy od používateľského rozhrania. Pri požiadavku na vyhľadávanie sa cez vyhľadávací cyklus prejdú všetky oblasti a do ich servisných tried sú odosielané informácie o vyhľadávanom výraze, jazykový kód a identifikačné číslo webu, z ktorého vyhľadávanie prebieha. Servisné triedy následne vracajú nájdené výsledky, ktoré sú pridávané do spoločného poľa dát. Po skončení cyklu vyhľadávania sa všetky výsledky zostupne zoradia na základe skóre cez vstavanú funkciu `usort`, kde sa v anonymnej funkcii porovnávajú pomocou operátora typu `spaceship`[4] a vykresľujú sa na používateľské rozhranie.

Pre zlepšenie prehľadnosti používateľského rozhrania som upravil aj formát výpisu výsledkov. Popisy výsledkov sú detailnejšie a poskytujú presnejší opis každého nájdeného dokumentu, čím sa zvyšuje používateľská prívetivosť. Pre väčšiu prehľadnosť medzi výsledkami som implementoval nové označovanie na základe fakulty, z ktorej dokument pochádza. Označenie sa nachádza vedľa typu dokumentu a je označené príslušnou farbnou schémou univerzity (pozri obrázok 6.2).



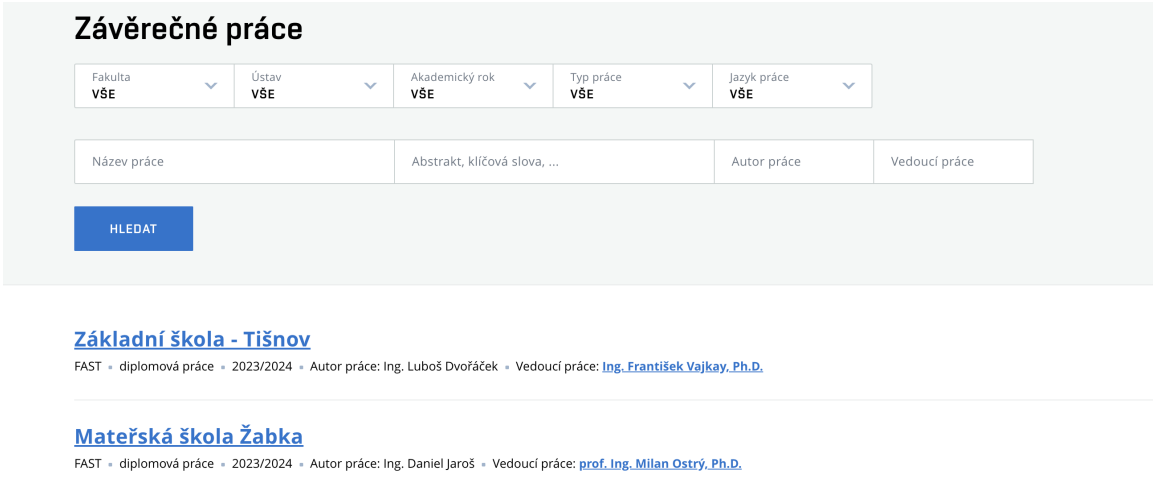
Obrázek 6.2: Upravené používateľské rozhranie hlavného vyhľadávania na webe VUT

Vyhľadávanie záverečných prác

V oblasti vyhľadávania záverečných prác sa už v pôvodnej implementácii aktívne využívala modelová vrstva pre hľadanie prác v centrálnej databáze. Pre prácu s Elasticsearch však bolo potrebné pridať prepojenie so servisnou vrstvou. S tou bolo potrebné prepojiť aj jednotlivé filtre dostupné na používateľskom rozhraní (pozri obrázok 6.3) a zaistiť, aby sa zabezpečilo ich vzájomné koordinované fungovanie. K zmene došlo aj v šablóne stránky, aby sa zaistila kompatibilita s novou modelovou vrstvou vyhľadávania.

Na používateľskom rozhraní sa odstránilo overenie pomocou `reCaptcha`, ktoré obmedzovalo externých webových robotov. Tí boli pôvodne na podstránke záverečných prác ob-

[4] https://www.php.net/manual/en/language.operators.comparison.php

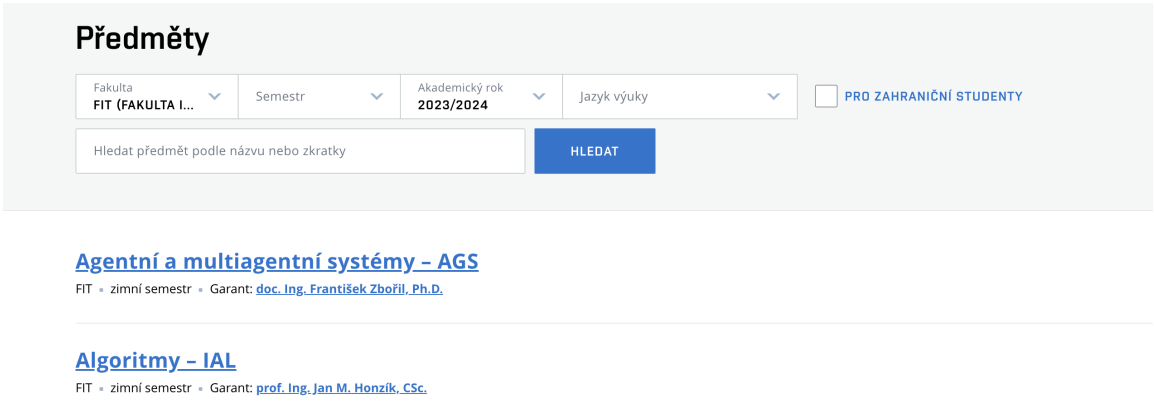


Obrázek 6.3: Používateľské rozhranie vyhľadávania záverečných prác.

medzení z dôvodu nadmerného zaťažovania centrálnej databázy. Výsledky vyhľadávania sú na používateľskom rozhraní zoradené podľa skóre dokumentov, ktoré sa v Elasticsearch vyrátava aj na základe použitých filtrov.

Vyhľadávanie predmetov

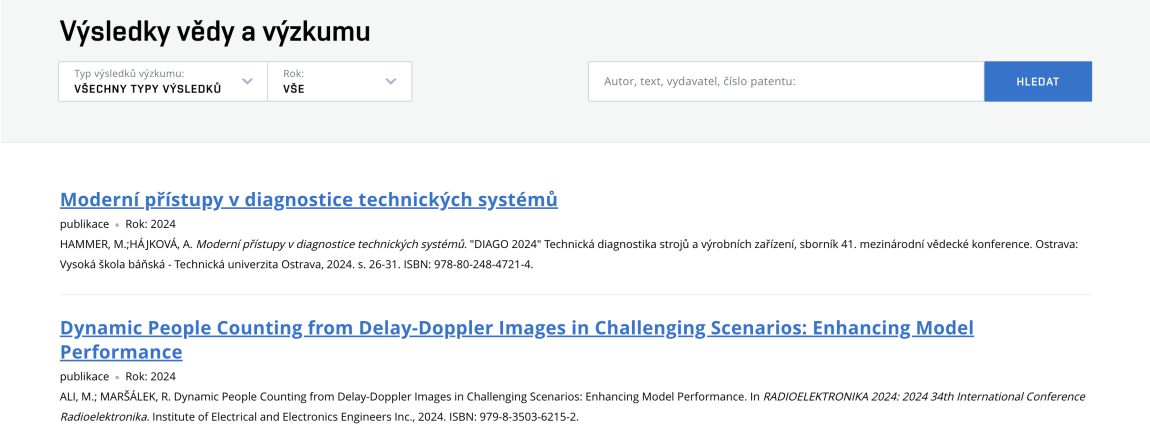
Oblasť vyhľadávania predmetov v pôvodnej implementácii načítavala výsledky vyhľadávania v kontroléri, ktoré boli predávané do šablóny na vykreslenie. V integrácii nového vyhľadávania cez Elasticsearch boli z kontroléra odstránené všetky databázové dotazy a operácie, ktoré nahradila inštancia servisnej triedy. Trieda je inštanciovaná pomocou návrhového vzoru `factory`. Do servisnej triedy sú predávané dáta z filtrovacieho formuláru (pozri obrázok 6.4). V šablóne predmetov som upravil spôsob vypisovania výsledkov tak, aby šablóna dokázala pracovať s iterátorom a objektami získanými zo servisnej triedy. Výsledky vyhľadávania sú na používateľskom rozhraní zoradené štandardne podľa abecedy a v prípade použitia filtrovacích možností sa pridáva zoradenie zostupne podľa skóre.



Obrázek 6.4: Používateľské rozhranie vyhľadávania predmetov.

Vyhľadávanie výsledkov vedy a výskumu

Rovnako ako v prípade záverečných prác, aj pri výsledkoch vedy a výskumu sa už pre vyhľadávanie v centrálnej databáze aktívne využívala modelová vrstva. Pre implementáciu vyhľadávania cez Elasticsearch však podľa návrhu vznikla nová služba obstarávajúca aj toto vyhľadávanie. Táto služba pracuje s už zaužívaným modelom výsledkov vedy a výskumu, ktorý bol pri implementácii nového vyhľadávania doplnený o potrebné vlastnosti a funkcie.



Obrázek 6.5: Používateľské rozhranie vyhľadávania vedy a výskumu.

Ako je vidieť na obrázku 6.5, používateľské rozhranie ponúka filtráciu výsledkov podľa typu výsledku a roku publikácie. Tieto filtre sú v kontroléri spracované a odosielané do služby, ktorá ich pridá do dotazu pre Elasticsearch. Spolu s implementáciou hľadania cez Elasticsearch bola aj v tomto prípade odstránená `reCaptcha`, ktorá obmedzovala webových robotov.

Kapitola 7

Testovanie a nasadenie vyhľadávania

V tejto kapitole je opísané testovanie a nasadenie implementovaného vyhľadávania. Testované sú jednotlivé analyzátory textu, ktoré boli implementované v nástroji Elasticsearch. Okrem toho kapitola opisuje testovanie na základe vytvoreného scenára a vyhodnocuje konečné výsledky.

7.1 Testovanie správnosti textovej analýzy

Textové analyzátory implementované v nástroji Elasticsearch tvoria základ pre relevantné vyhľadávanie. Z pohľadu dlhodobého vývoja a udržateľnosti bolo potrebné zaistiť, že akékoľvek budúce zmeny v nastaveniach indexov neovplyvnia navrhnutú analýzu textu. Z tohto dôvodu vznikli jednotkové testy, ktoré overia funkčnosť implementovaných analyzátorov.

Pre tento typ testovania sa bude využívať knižnica pre jednotkové testy `PHPUnit`[1], ktorá je aktuálne v informačnom systéme VUT využívaná vo verzii 8.5. Testovací adresár je umiestnený v rámci menného priestoru `Elasticsearch` v zložke `Tests`. Táto zložka obsahuje konfiguračný súbor testov `phpunit.xml` a súbor s pripravenými testami. Pred začiatkom testovania sa pripravuje prostredie potrebné pre testovanie, v rámci čoho sa vytvára aj inštancia implementovaného Elasticsearch klienta. Cez klienta budú odosielané jednotlivé testovacie texty na koncový bod `_analyze`, ktorý bližšie opisujem v sekcii 5.2. Priebeh každého jednotkového testu je nasledovný:

1. V teste sa pripraví telo dotazu. To obsahuje zvolený analyzátor a text, ktorý má byť analyzovaný.
2. Pripravia sa aj vzorové tokeny, ktoré okrem slova obsahujú aj informáciu o dátovom type a ich umiestnení v rámci textu.
3. Test odošle dotaz cez Elasticsearch klient na požadovaný index a čaká na odpoveď.
4. Získaná odpoveď s analyzovaným textom sa porovná oproti vzorovým tokenom. V prípade, že sú tokeny, ich pozície a dátove typy zhodné, test je označený za úspešný.

V testoch sú obsiahnuté všetky implementované analyzátory, a to pre každý používaný index. Testovanie teda overí nielen funkčnosť samotných analyzátorov, ale aj ich konzisten-

[1] https://phpunit.de

ciu medzi jednotlivými indexmi. Testované sú aj analyzátory typu N-gram, aby sa v prípade zmien v nastavení indexov overilo správne vytváranie postupnosti znakov. Pre spustenie automatického testovania je potrebné prejsť do testovacej zložky a spustiť príkaz pre spustenie testovacieho procesu `/lib/vendor/bin/phpunit`.

7.2 Testovanie relevancie vyhľadávania

Testovanie relevancie vyhľadávania prebiehalo na testovacej verzii informačného systému VUT už v priebehu integrácie nového vyhľadávania do dostupných webových vyhľadávačov. Cieľom tohto testovania bolo zhodnotiť relevancie ponúkaných výsledkov a ich zoradenie podľa skóre vzhľadom na zadaný vyhľadávací dotaz.

Toto testovanie prebiehalo na základe vopred vytvorených testovacích scenárov. Tie boli vytvorené pre všetky implementované vyhľadávače a indexy, aby sa overilo, že sú výsledky konzistentné naprieč celým informačným systémom. Do testovania boli zo systému zámerne vybrané také dokumenty, ktoré boli považované za najviac reprezentatívne vzhľadom na rôznorodosť a komplexnosť dát, čo umožnilo detailne posúdiť presnosť a relevanciu vyhľadávaných výsledkov. Celkovo bolo pre tento druh testovania pripravených jedenásť scenárov, ktoré sú pripojené ako elektronické prílohy k práci.

Pre príklad som do tejto práce vybral jeden z pripravených scenárov. Jedná sa o testovanie vyhľadávača záverečných prác a jeho jednotlivých filtrovacích možností. Cieľom tohto testu je overiť nasledujúce tvrdenia:

- Vyhľadávač dokáže správne analyzovať používateľský dotaz.
- Dostupné filtre fungujú podľa očakávaní a je možné ich navzájom kombinovať.
- Výsledky vyhľadávania sú relevantné a správne zoradené podľa dotazu od používateľa.

Pre každý scenár je pripravených niekoľko úloh, ktoré sa majú vykonávať v zadanom poradí. Na základe týchto úloh sa následne vyhodnotí výsledok testu. Príkladový scenár pre vyhľadávač záverečných prác pozostáva z ôsmych úloh:

1. Otvorenie verejnej časti informačného systému VUT a prejdenie do časti „Studium na VUT“ a jej súčasti „Závěrečné práce“.
2. Skontrolovať, či sa pri prvotnom otvorení vyhľadávača vypísali záverečné práce za aktuálny akademický rok. Okrem toho skontrolovať, či správne funguje stránkovanie v spodnej časti webovej stránky.
3. Akademický rok nastaviť na „vše“ a do poľa „Název práce“ vyplniť „informačné systémy“.
4. Skontrolovať, že aj napriek preklepu (vyhľadávaniu so slovenským dotazom) nájde zoznam prác.
5. K vyhľadávaniu pridať nasledujúce filtre: „Fakulta“ nastaviť na „FIT“, akademický rok na „2022/2023“, typ práce na „bakalářská práce“ a jazyk na „slovenština“.
6. Skontrolovať nájdené výsledky, k dispozícii by mali byť práve tri nájdené výsledky.

7. K aktuálnemu vyhľadávaniu pridať ďalšie filtre: pole „Abstrakt, klíčová slova“ nastaviť na „výskumná skupina znalostných technológií“ a „Vedoucí práce“ na „Jaroslav Dytrych“ (meno vedúceho uviesť bez titulov, aby sa overila správna analýza textu).

8. V prípade, že bola nájdená práve jedna práca s názvom „Integrace a optimalizace modulů informačního systému KNOTIS“, testovanie bolo úspešne ukončené.

7.3 Nasadenie vyhľadávania

Pri nasadení nového vyhľadávania bolo potrebné zriadiť novú serverovú inštanciu pre nástroj Elasticsearch. Pre potreby tohto nástroja je potrebné, aby bolo na serveri nainštalované prostredie pre programovací jazyk `Java`. Na server bola nainštalovaná najnovšia stabilná verzia Elasticsearch. Pri inštalácii bolo potrebné nástroj Elasticsearch nakonfigurovať cez súbor `elasticsearch.yml` tak, aby sa zaistilo optimálne využitie zdrojov, maximálna dostupnosť služieb a efektívne indexovanie dát. Spolu s Elasticsearch bol inštalovaný aj nástroj Kibana, ktorý je bližšie opísaný v sekcii 5.11.

Pre potreby analýzy textu v českom jazyku bolo potrebné nainštalovať do Elasticsearch dve rozšírenia. Prvým rozšírením je `ICU analysis`[2], ktoré je možné nainštalovať priamo cez príkazový riadok nástroja Elasticsearch. Druhým rozšírením je český slovník Hunspell. Tento slovník sa inštaluje manuále cez domovskú zložku nástroja Elasticsearch. V adresári `config` bolo potrebné vytvoriť zložku s názvom `hunspell` a vrámci nej podzložku `cs_CZ`, do ktorej sa presunuli slovníkové súbory: `cs_CZ.aff`, `cs_CZ.dic` a `settings.yml`. Týmto bola inštalácia servera pre nástroj Elasticsearch dokončená a pripravená na použitie.

Implementované riešenie tejto práce je aktuálne nasadené na vývojovom serveri a je pripravené pre nasadenie do produkčnej verzie informačného systému VUT. Do produkcie malo byť nové vyhľadávanie nasadené už v priebehu apríla 2024, kvôli odhalenej chybe v databázovej vrstve informačného systému však došlo k odkladu. Chyba bola nájdená pri synchronizácii záverečných prác, kedy v databázovej vrstve dochádzalo k únikom pamäte, čo spôsobovalo, že pamäť nebola postupne uvoľňovaná. Pri synchronizácii väčšieho objemu dát záverečných prác to viedlo k prekročeniu alokačného limitu pamäti. Chyba bola nahlásená pracovníkom z oddelenia CVIS, ktorí aktuálne pracujú na jej odstránení. Nový dátum nasadenia do produkčného prostredia bol po porade s vedúcimi pracovníkmi oddelenia CVIS určený na letné prázdniny 2024.

[2] https://www.elastic.co/guide/en/elasticsearch/plugins/current/analysis-icu.html

Kapitola 8

Záver

Cieľom tejto práce bolo preštudovať možnosti fulltextového vyhľadávania a jednotlivých dátových štruktúr informačného systému VUT a získané znalosti využiť pri návrhu a implementácii nového vyhľadávania.

Po preštudovaní dostupných nástrojov pre fulltextové vyhľadávanie bol pre implementáciu vybraný nástroj Elasticsearch. V rámci návrhu vzniklo niekoľko typov textových analyzátorov a indexových šablón, ktoré boli neskôr implementované do nástroja Elasticsearch. V informačnom systéme následne došlo k implementácii služieb pre synchronizáciu dát, ktoré sú spúšťané v pravidelnom intervale. Počas implementácie služieb došlo aj k ladeniu dotazov používaných pre vyhľadávanie, aby sa zaistila vysoká relevancia a konzistencia získavaných výsledkov. V závere bolo vyhľadávanie cez Elasticsearch implementované do jednotlivých vyhľadávačov dostupných v IS VUT. Na základe testovania podľa pripravených scenárov sa podarilo overiť správnu funkčnosť vyhľadávania a nájdené nedostatky odstrániť.

Všetky navrhnuté časti boli úspešne implementované, otestované a sú pripravené pre nasadenie do produkčného prostredia. V priebehu návrhu a implementácie som jednotlivé kroky konzultoval s pracovníkmi CVIS. Výsledkom mojej práce je vyhľadávanie, ktoré jednotlivými požiadavkami nezaťažuje centrálnu databázu a ponúka spracovávanie rôznych typov vyhľadávacích dotazov a relevantné výsledky. Vyhľadávací nástroj a jeho indexy boli nakonfigurované pre schopnosť automatického škálovania, čo umožňuje jeho používanie aj z dlhodobého hľadiska.

V rámci nadväzujúcej práce by som chcel rozšíriť možnosti užívateľského rozhrania vyhľadávačov a implementovať aktívne vyhľadávanie počas písania vyhľadávacieho dotazu. Okrem toho by bolo možné vyhľadávanie cez nástroj Elasticsearch postupne rozšíriť aj na internú časť informačného systému VUT, napríklad pre aplikácie `Portál`, `StudIS` alebo `Teacher`.

Literatura

[1] Ali, A. *Sphinx Search Beginner's Guide*. 1. vyd. Packt Publishing, 2011. ISBN 978-1849512541.

[2] Beall, J. The Weaknesses of Full-Text Searching. *The Journal of Academic Librarianship* online. 1. vyd., 2008, sv. 34, č. 5, s. 438–444. ISSN 0099-1333. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0099133308001067.

[3] Bierer, D.; Hussain, A. a Ajzele, B. *PHP 7: Real World Application Development*. 1. vyd. Packt Publishing, 2016. ISBN 978-1787129009.

[4] Croft, W. B.; Metzler, D. a Strohman, T. *Search Engines: Information Retrieval in Practice*. 1. vyd. Pearson, 2009. ISBN 978-0136072249.

[5] Elastic. *Full Text Queries - Elasticsearch Documentation* online. 2021. Dostupné z: https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html. [cit. 2024-28-04].

[6] Gheorghe, R.; Hinman, M. L. a Russo, R. *Elasticsearch in Action*. 1. vyd. Manning Publications, 2015. ISBN 978-1617291623.

[7] Grainger, T. a Potter, T. *Solr in Action*. 1. vyd. Manning, 2014. ISBN 978-1617291029.

[8] Hanák, D. *Stopárov sprievodca REST API* online. 2021. Dostupné z: https://www.itnetwork.sk/programovani/nezaradene/stoparov-sprievodca-rest-api. [cit. 2024-24-04].

[9] Konda, M. *Elasticsearch in Action*. 2. vyd. Manning Publications, 2023. ISBN 978-1617299858.

[10] Macrometa. *What is Full-Text Search?* online. 2024. Dostupné z: https://www.macrometa.com/articles/what-is-full-text-search. [cit. 2024-26-04].

[11] McCandless, M.; Hatcher, E. a Gospodnetić, O. *Lucene in Action*. 2. vyd. Manning, 2010. ISBN 978-1933988177.

[12] Pessotto, M. *Full-text search on a budget: Xapian* online. 2021. Dostupné z: https://www.endpointdev.com/blog/2021/08/full-text-search-xapian. [cit. 2024-24-04].

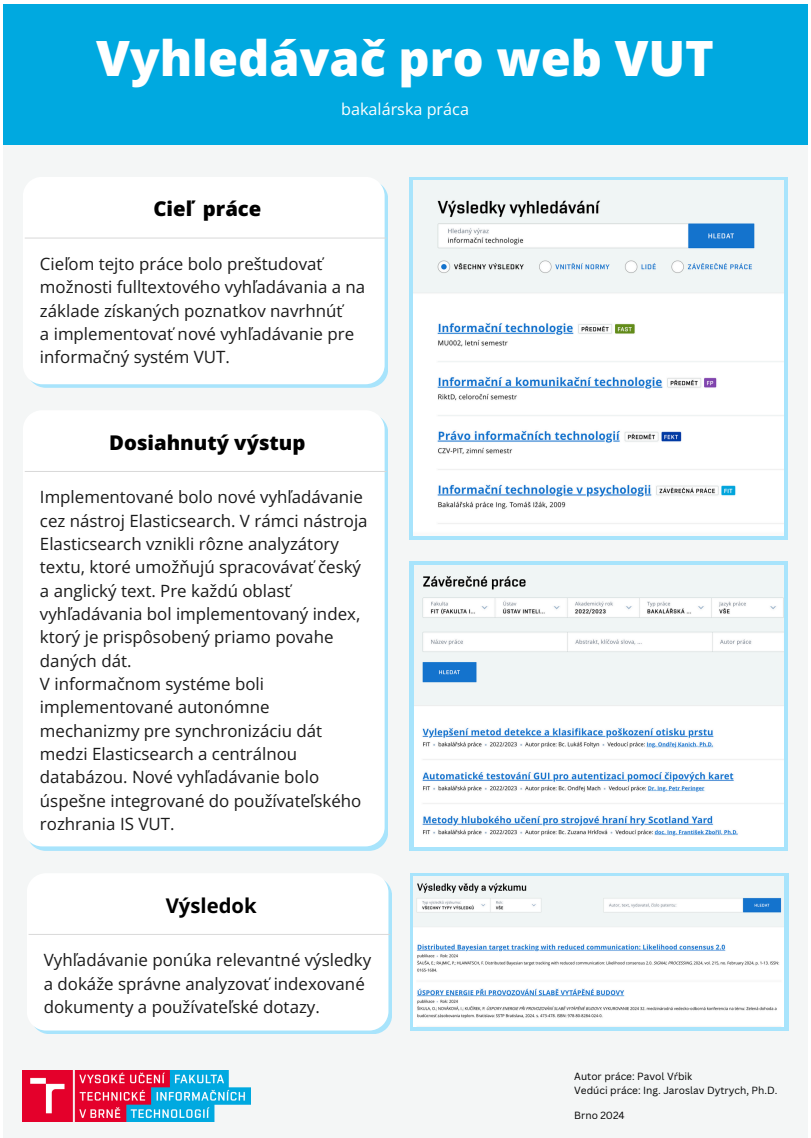
[13] Turnbull, D. a Berryman, J. *Relevant Search*. 1. vyd. Manning, 2016. ISBN 978-1617292774.

[14] Veselý, L. *Seriál Elasticsearch: 4. Fulltextové vyhledávání v češtině* online. 2017. Dostupné z: https://www.ludekvesely.cz/serial-elasticsearch-4-fulltextove-vyhledavani-v-cestine. [cit. 2024-26-04].

[15] Weisfeld, M. *The Object-Oriented Thought Process*. 3. vyd. Pearson Education, 2009. ISBN 978-0672330162.

Příloha A

Plagát



Obrázek A.1: Plagát prezentujúci výsledky práce.