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

AKTUALIZACE PORTÁLU EVROPSKÝCH PROJEKTŮ A JEHO ROZŠÍŘENÍ O IDENTIFIKACI VÝSLEDKŮ, SOUVISEJÍCÍCH S TÉMATY NOVĚ VYPISOVANÝCH VÝZEV

AN UPDATE OF EUROPEAN PROJECTS PORTAL AND ITS ENHANCEMENT BY IDENTIFICATION OF RESULTS RELATED TO NEW CALL TOPICS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **JIŘÍ FURDA**
AUTHOR

VEDOUCÍ PRÁCE — **doc. RNDr. PAVEL SMRŽ, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Furda Jiří**
Program: Informační technologie
Název: **Aktualizace portálu evropských projektů a jeho rozšíření o identifikaci výsledků, souvisejících s tématy nově vypisovaných výzev**
An Update of European Projects Portal and Its Enhancement by Identification of Results Related to New Call Topics
Kategorie: Web

Zadání:

1. Seznamte se s metodami automatického vyhledávání, stahování, zpracování a indexování webových dat a dostupným řešením portálu, zaměřeného na výsledky evropských projektů.
2. Navrhněte a realizujte úpravu stávajícího systému tak, aby bylo možné s použitím nejnovějších verzí systému ElasticSearch indexovat a intuitivním způsobem vyhledávat projekty k tématům, souvisejícím s nově vypisovanými výzvami.
3. Připravte vhodná data, která dovolí vyhodnotit úspěšnost automatických metod a uživatelskou stránku prezentační části systému.
4. Vytvořte stručný plakát prezentující práci, její cíle a výsledky.

Literatura:

- dle doporučení vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Funkční prototyp řešení

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Smrž Pavel, doc. RNDr., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 7. května 2019

Abstrakt

Tato bakalářská práce se věnuje úpravě stávajícího webového portálu, zaměřeného na výsledky evropských projektů. Výstupem této práce je portál s novým uživatelským rozhraním, pracující s novější verzí systému Elasticsearch. Oproti předchozímu řešení umožňuje vyhledávat v tématech vypsaných výzev. V teoretické části práce přibližuje praktiky využívané při tvorbě webů a popisuje využité systémy a jejich principy. V praktické části je blíže specifikováno fungování celého portálu a je popsána konkrétní implementace. Výsledek práce je vyhodnocen v závěru experimenty.

Abstract

This bachelor thesis deals with modification of an existing web portal focused on results of European projects. The output of this thesis is a portal working with newer version of the engine Elasticsearch. Compared to the previous solution, this portal allows to search topics of new calls. The theoretical part of the thesis describes practises used in web developement and frameworks used in this thesis and their principles. The practical part specifies the functioning of the whole portal and describes a specific implementation. The result is evaluated at the end of the thesis based on experiments.

Klíčová slova

webový portál, Python, Elasticsearch, Vue, Flask, fasetové vyhledávání

Keywords

web portal, Python, Elasticsearch, Vue, Flask, faceted search

Citace

FURDA, Jiří. *Aktualizace portálu evropských projektů a jeho rozšíření o identifikaci výsledků, souvisejících s tématy nově vypisovaných výzev*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Pavel Smrž, Ph.D.

Aktualizace portálu evropských projektů a jeho rozšíření o identifikaci výsledků, souvisejících s tématy nově vypisovaných výzev

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Furda
15. května 2019

Poděkování

Děkuji panu doc. RNDr. Pavlu Smržovi, Ph.D. za odborné vedení a věnovaný čas v průběhu tvorby této práce. Dále bych rád za pomoc poděkoval panu Mgr. Adamu Kövárimu.

Obsah

Kapitola 1

Úvod

Webové portály jsou v současné době nedílnou součástí internetu. Umožňují jejich návštěvníkům získat na jednom místě relevantní informace shromážděné z vícero různých zdrojů [25].

Cílem této bakalářské práce je rozšíření stávajícího portálu, zaměřeného na výsledky evropských projektů. Od vytvoření tohoto portálu již uplynulo několik let, a proto bylo třeba jej upravit pro použití novějších verzí knihoven. Při této činnosti bylo kompletně přepracováno uživatelské rozhraní a obsah tohoto portálu byl rozšířen o témata nově vypsaných výzev.

Pro toto téma jsem se rozhodl, protože mám k webovým technologiím blízko již od dětství a chci v tomto směru nadále rozvíjet své znalosti.

Tato práce vychází z bakalářské práce *Automaticky aktualizovaný webový portál* Petra Staňka [36]. Webový portál byl od základu implementován znovu s použitím modernějších technologií a čistějšího zdrojového kódu. Extrakce dat zůstala zachována, pouze byla rozšířena o extrakci témat výzev k předkládání návrhů.

V druhé kapitole práce, nazvané „Rozbor řešené problematiky“, nejprve stručně shrnu základní principy vývoje webových stránek a poté přiblížím několik technologií, které jsem při tvorbě této bakalářské práce využil. Ve třetí kapitole s názvem „Návrh a implementace systému“ se již zaměřím na konkrétní řešení této práce, uvedu několik snímků z portálu, jež je výsledkem této práce, a popíši několik zajímavých částí samotného kódu. Ve čtvrté kapitole pojmenované „Experimenty a vyhodnocení“ provedu s výsledným portálem několik experimentů, následně vyhodnotím jejich výstupy a vyvodím z nich možná rozšíření portálu. V samém závěru práce pak stručně zhodnotím výsledky této práce.

Výsledek práce je k nahlédnutí online[1].

[1] European Research Projects Portal: http://athena17.fit.vutbr.cz:2021/

Kapitola 2

Rozbor řešené problematiky

Tato kapitola v úvodu přibližuje základní koncepci vývoje webu a praktiky, které jsou při tomto vývoji často využívány. Následně jsou v této kapitole popsány jazyky a knihovny, využívané při implementaci této bakalářské práce. V samém závěru kapitoly je přiblížena koncepce podobnostního hledání.

2.1 Vývoj webu

Tato kapitola čerpá z [9].

Z pohledu vývoje nejen webových stránek je zvykem projekt rozlišovat do dvou vrstev. Tou první je klientská část (anglicky *frontend*) a druhou je serverová část (anglicky *backend*).

Klientská část webu je to, co uživatel vidí, když stránku navštíví. Sestavení této části se v žádném případě neobejde bez značkovacího jazyka HTML (*Hypertext Markup Language*), který udává strukturu a obsah webové stránky. Další nedílnou součástí jsou kaskádové styly neboli CSS (*Cascading Style Sheets*). Ty ovlivňují, jak bude obsah stránky esteticky působit. Další součástí, se kterou se setkáte téměř na každém webu, je skriptovací jazyk JavaScript, který umožňuje webové stránky obohatit o interaktivitu. Zpracování této části webu probíhá na straně klienta - tedy internetovému prohlížeči uživatele. Na základě tohoto faktu lze každé stránce na internetu zobrazit zdrojové kódy klientské části. Ve většině případů je možné i rovnou v internetovém prohlížeči tyto kódy upravit, což má samozřejmě vliv jen pro současného uživatele a ostatním návštěvníkům webu se tyto změny nijak neprojeví.

Oproti tomu serverová část je od uživatele izolována a zdrojové kódy pro něj nejsou přístupné, pokud je sám autor někde nezveřejní. Jejich interpretace totiž probíhá přímo na serveru a ten poté klientovi odešle již zpracovanou odpověď, obsahující pouze zdrojové kódy klientské části. Výběr technologie implementace serverové části je podstatně rozmanitější, patří mezi ně například Python, PHP, Java, C#, C++, .NET, Ruby a další. Díky prostředí Node.js[1] lze ale využít i dříve zmiňovaný JavaScript, který pro tento účel nebyl primárně určen.

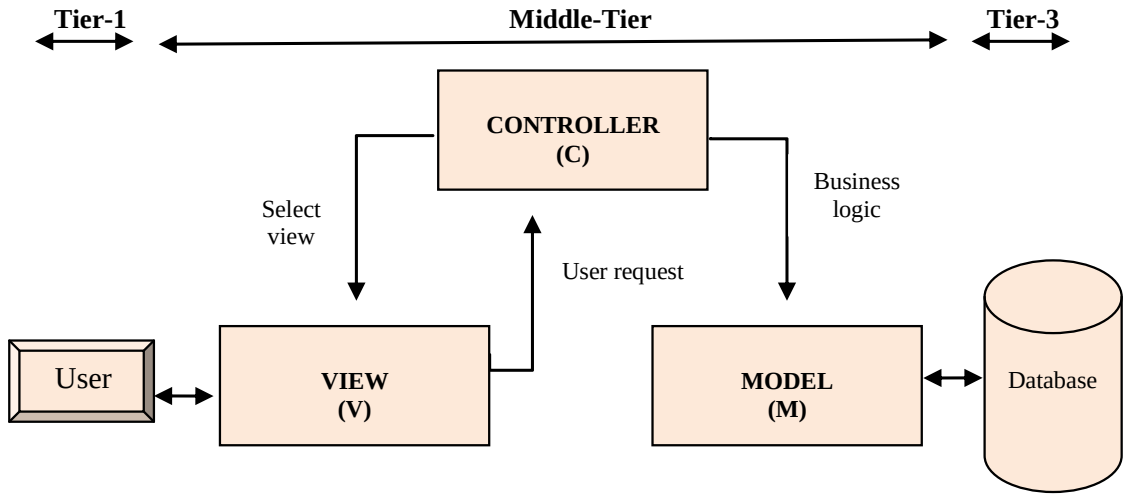
2.1.1 Architektura MVC

Tato kapitola čerpá z [34].

Většinu aplikací lze obecně rozdělit na tři hlavní celky - data, rozhraní a logiku. V anglickém jazyce se tyto celky označují pojmy *model*, *view* a *controller* (zkratka MVC). Historie

[1] Node.js: https://nodejs.org/en/

této architektury sahá až do sedmdesátých let, nicméně i v dnešní době je stále naprostým standardem vývoje. Schéma zmíněné architektury je znázorněno v grafu 2.1.



Obrázek 2.1: Diagram toku akcí při využití architektury MVC. *Převzato z [34].*

- Pojmem pohled (*view*) se označuje uživatelské rozhraní (například stránka zobrazená v prohlížeči), tato vrstva prezentuje data a je to také jediná vrstva, s kterou uživatel přímo komunikuje. Akce provedené v pohledu se předají kontroleru ke zpracovaní.
- Kontroler (*controller*) se chová jako prostředník mezi daty a rozhraním, zpracovává požadavky uživatele a provede s modelem potřebné akce. Ve většině případů po provedené akci kontroler znovu obnoví pohled.
- Model je obvykle objekt obsahující patřičná data z databáze. Zprostředkovává jak jejich získání, tak i ukládání. Mimo to může být objekt obohacen i o funkce, zpracovávající tyto data (například může převádět Unixový čas do formátu lépe čitelného pro člověka).

Oddělením těchto celků je zajištěna lepší organizovanost zdrojových kódů, a tím je usnadněn rychlejší vývoj. Jednotlivé celky jsou také snadněji znovupoužitelné a práce programátorů na nich může probíhat souběžně. Díky rozdělení je také možné pro jeden model mít pohledů hned několik. Další pohledy mohou přibývat nebo naopak ty stávající mohou být smazány a na databázi aplikace to nebude mít žádný vliv.

2.1.2 Responzivita

Tato kapitola čerpá z [20].

Responzivní design, jak název vypovídá, je design, který je schopný reagovat. Stránka, která má takový design, se přizpůsobuje zařízení, na kterém je právě zobrazena. Díky tomu lze pohodlně prohlížet tu stejnou webovou stránku jak na stolním počítači, tak i mobilním zařízení.

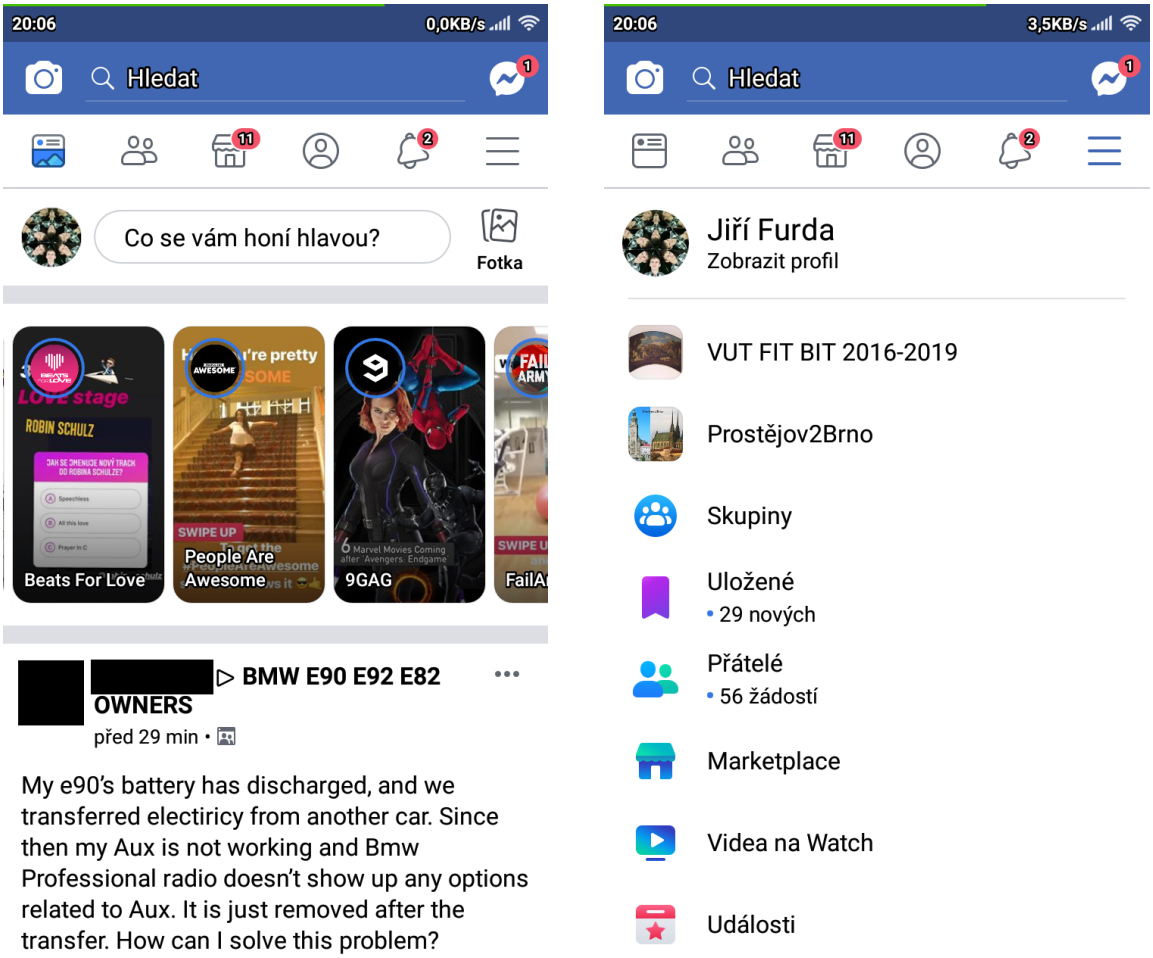
Před několika lety bylo zvykem tvořit webové stránky s pevnou šířkou obsahu. Tato šířka byla volena tak, aby bylo možné stránku zobrazit jak na starších monitorech s poměrem stran 4:3, tak i na modernějších širokoúhlých monitorech.

Se stále rostoucí oblibou a dostupností chytrých telefonu se ale pro vývojáře a designéry objevila nová výzva. Bylo potřeba webové stránky přizpůsobit i pro tyto zařízení s různorodou velikostí obrazovky. Webovou stránku, která nenabízí responzivní design, sice je možné zobrazit i na mobilním zařízením, ale prohlížení takové stránky zpravidla není pro uživatele nikterak přívětivé. Uživatel může být nucen stránku neustále přibližovat a oddalovat, aby se dostal k části, s kterou zrovna potřebuje pracovat, nebo aby se mu podařilo stisknout konkretní tlačítko či odkaz. V některých případech může být dokonce část obsahu stránky překrytá jiným prvkem a uživateli bude tento obsah skryt. Kromě toho faktu může mít nevyužití responzivního designu také negativní vliv na pořadí stránky ve výsledcích internetových vyhledávačů.

Díky nástupu nových verzí jazyků HTML5 a CSS3 je možné tuto situaci řešit bez nutnosti použití některého ze serverových řešení. Nejpřínosnější novinkou v tomto směru jsou tzv. *media queries* v kaskádových stylech, díky kterým je možné aplikovat odlišné styly na základě specifikací zobrazovacího zařízení. Jak již bylo v úvodu zmíněno, design stránky se nejčastěji odvíjí od šířky obrazovky zařízení. V souboru s kaskádovými styly se tedy lze potkat například s následující specifikací stylů.

```
@media(max-width: 640px)
{
    /* Specifické styly */
}
```

Takto obalené styly budou aplikovány jen v případě, že zobrazovací zařízení splňuje podmínku v kulatých závorkách. V tomto případě se jedná o zařízení s šířkou obrazovky menší nebo rovnou 640 pixelům. Taková obrazovka může být ku příkladu moc malá na zobrazení postranního panelu s navigací. Řešením takové situace dnes v drtivé většině případů bývá skrytí navigace do značně menšího tlačítka označujícího se jako „hamburgerové menu“.



Obrázek 2.2: Hamburgerové menu použité v mobilní aplikaci sociální sítě Facebook[3]. Jeho zobrazení je ovládáno tlačítkem v pravé horní části obrázků, znázorněným třemi vodorovnými čárami. Na levém snímku je možné vidět aplikaci se skrytým menu a v pravém snímku zase se zobrazeným.

2.1.3 Princip nejdříve mobil

Ustálenou technikou, jak vyvíjet responzivní design webových stránek, je princip „nejdříve mobil“ (anglicky *mobile-first*). Jak název vypovídá, základní myšlenkou tohoto principu je nejdříve myslet na mobilní zařízení, a až poté na stolní počítač. Tento přístup je možné aplikovat jak pro samotný návrh designu, tak i pro jeho následnou implementaci.

Protože mobilní zařízení nabízejí o mnoho menší zobrazovací plochu než stolní počítače, je zde každý kus prostoru drahocenný. Designér si proto musí nejprve ujasnit, které části webové stránky jsou pro uživatele prioritní a na základě těchto priorit sestavit návrh.

V případě využití tohoto principu při samotné implementaci designu se postupuje obdobně. Pokud jsou všechny kaskádové styly jak pro mobilní, tak pro stolní počítače zapsána v jednom souboru, bude tento soubor na začátku nejprve obsahovat styly společné pro obě zařízení a styly určené pouze pro mobilní zařízení. Až poté se budou v souboru objevovat specifické styly, nejpravděpodobněji podmíněné hodnotou vlastnosti `min-width`. Tyto styly budou aplikovány od určité šířky displeje a budou zaměřené na zobrazení například na stol-

ních počítačích. Díky tomuto přístupu bude samotný kód stylů zjednodušen. U mobilních zařízení bývá zpravidla potřeba méně stylů a často lze využít některé z výchozích vlastností daných prvků [38].

2.2 Boostrap

Tato kapitola čerpá z [5].

Bootstrap[4] je systém usnadňující vývoj klientské části webových stránek. Byl vyvinut společností Twitter[5], která jej v roce 2010 vydala jako otevřený software. Pouhé dva roky poté se Bootstrap stal nejoblíbenějším projektem na stránce GitHub[6]. Bootstrap se skládá převážně z kaskádových stylů. Interaktivní prvky systému využívají skriptovací jazyk JavaScript s knihovnou jQuery[7]. Existují ovšem i další alternativy, které místo jQuery využívají například Vue.js[8], React[9] nebo Angular[10].

Bootstrap se drží moderních zásad jako je princip „nejdříve mobil“ (uvedený v kapitole 2.1.3) a nabízí snadnou možnost tvorby responzivního designu (popsaného v kapitole 2.1.2). Mimo to také nabízí možnost využít moderně působící vzhled základních HTML prvků jako jsou například vstupní pole formulářů nebo tlačítka. Bootstrap dále poskytuje i celou řadu nových předem připravených komponent, a to ku příkladu karty, modální okna, stránkování, informační hlášky a spoustu dalších.

Vzhledem ke své rozsáhlé komunitě lze nalézt mnoho dalších uživateli tvořených komponent nebo dokonce celých hotových šablon webů postavených na tomto systému. Jedna ze stránek, určená pro sdílení takového obsahu, se nazývá Bootsnipp[11].

2.3 Python

Python[12] je vysokoúrovňový interpretovaný objektově orientovaný programovací jazyk. Byl vytvořen roku 1990 a jeho autorem je *Guido van Rossum*. Samotné jádro tohoto jazyka je velmi malé, ale nabízí využití spousty nástrojů pro nejrůznější účely. Tato rozšíření jsou psaná jak v samotném jazyce Python, tak i v jazyce C nebo C++ [33]. Specifikem tohoto jazyka je nepoužívání středníku jako oddělovače instrukcí a také striktní pravidla pro odsazení kódu. Dle odsazení se totiž rozlišují bloky kódu, na rozdíl od většiny ostatních jazyků, kde k tomuto účelu slouží symboly složených závorek.

2.3.1 Flask

Flask[13] je systém s otevřeným zdrojovým kódem určený pro tvorbu systémové části webových stránek. Je napsaný v programovacím jazyce Python a jedná se o systém typu *microframework*. Klíčový znak tohoto typu systémů je co nejmenší nebo i nulová závislost na

[4] Bootstrap: https://getbootstrap.com/
[5] Twitter: https://twitter.com/
[6] GitHub: https://github.com/
[7] jQuery: https://jquery.com/
[8] BootstrapVue: https://bootstrap-vue.js.org/
[9] React Bootstrap: https://react-bootstrap.github.io/
[10] NG Bootstrap: https://ng-bootstrap.github.io/#/home/
[11] Bootsnipp: https://bootsnipp.com/
[12] Python: https://www.python.org/
[13] Flask: http://flask.pocoo.org/

externích knihovnách [29]. Flask si klade za cíl udržet si co nejjednodušší jádro, ale zároveň se soustředí na možnost tento základ jednoduše rozšířit [24].

Smyslem systému Flask je být základním kamenem pro jakýkoliv typ webové aplikace. Z tohoto důvodu na rozdíl od ostatních systémů neobsahuje žádnou abstrakci databázové vrstvy nebo knihovnu pro formuláře. V případě potřeby využití takovýchto funkcí, lze použít některé z dostupných rozšíření [28].

Ve svém základu systém využívá šablonovací jazyk Jinja2[14] zjednodušující udržení konzistentní struktury stránek. Poskytuje například dědičnost šablon nebo znovupoužitelné bloky a dokáže se vypořádat s bezpečnostními útoky typu XSS (*Cross Site Scripting*) [3]. Druhou knihovnou, kterou systém Flask využívá, je Werkzeug[15] - jedna z nejpokročilejších knihoven pro rozhraní brány webového serveru (WSGI neboli *Web Server Gateway Interface*), které umožňuje webovou aplikaci provozovat nezávisle na technologii použité na serveru [30].

Tento systém je využíván například sociální sítí Pinterest[16] [37] nebo síťovou dopravní společností Lyft[17] [32].

2.4 Vue.js

V dnešní době jsou webové prohlížeče stále schopnější a výkonnější. Díky tomu se stává trendem přenášet stále větší části webové aplikace ze serveru na stranu klienta. Toho je docíleno pomocí skriptovacího jazyka JavaScript. V současné době se nejvíce využívá knihovna jQuery, ale začaly se objevovat pokročilejší systémy jako například React[18], Angular[19] a v neposlední řadě Vue.js[20].

Vue.js je progresivní systém - přizpůsobuje se složitosti projektu. Podobně jako dříve zmiňovaný systém Flask se nejedná o rozsáhlý systém, jehož části by se vypínaly, ale jedná se o základní systém, který lze dále rozšiřovat [17]. Díky tomuto přístupu učební křivka není tak strmá, jako u konkurenčních systémů. Jediné, co člověku pro začátek práce se systémem Vue.js stačí, je být obeznámen s jazyky HTML a JavaScript [18].

Jedna z předností toto systému je jednoduchost obousměrné vazby dat. Tato vazba znamená, že hodnota proměnné v jazyce JavaScript je synchronizována s hodnotou v objektovém modelu dokumentu (DOM neboli *Document Object Model*). To stejné platí i v opačném směru [1]. V praxi může být tato funkce využita například v internetovém obchodě, kdy uživatel klikne na tlačítko „Přidat do košíku“, které pouze rozšíří pole košíku o další produkt. Bez jakéhokoliv dalšího úsilí komponenty, závislé na této proměnné, zaznamenají změnu a automaticky provedou odpovídající akce. Košík přepočítá celkový počet vybraného zboží, celkovou cenu nákupu, případně se zlevní cena daného produktu, při koupi dalších kusů a podobně.

2.4.1 Vuex

Jedním z oficiálních rozšíření je Vuex, knihovna pro správu stavu. Pokud se v aplikaci používá více komponent, které využívají stejnou proměnnou, brzy se zdrojový kód začne

[14] Jinja2: http://jinja.pocoo.org/
[15] Werkzeug: https://pypi.org/project/Werkzeug/
[16] Pinterest: https://cz.pinterest.com/
[17] Lyft: https://www.lyft.com/
[18] React: https://reactjs.org/
[19] Angular: https://angular.io/
[20] Vue.js: https://vuejs.org/

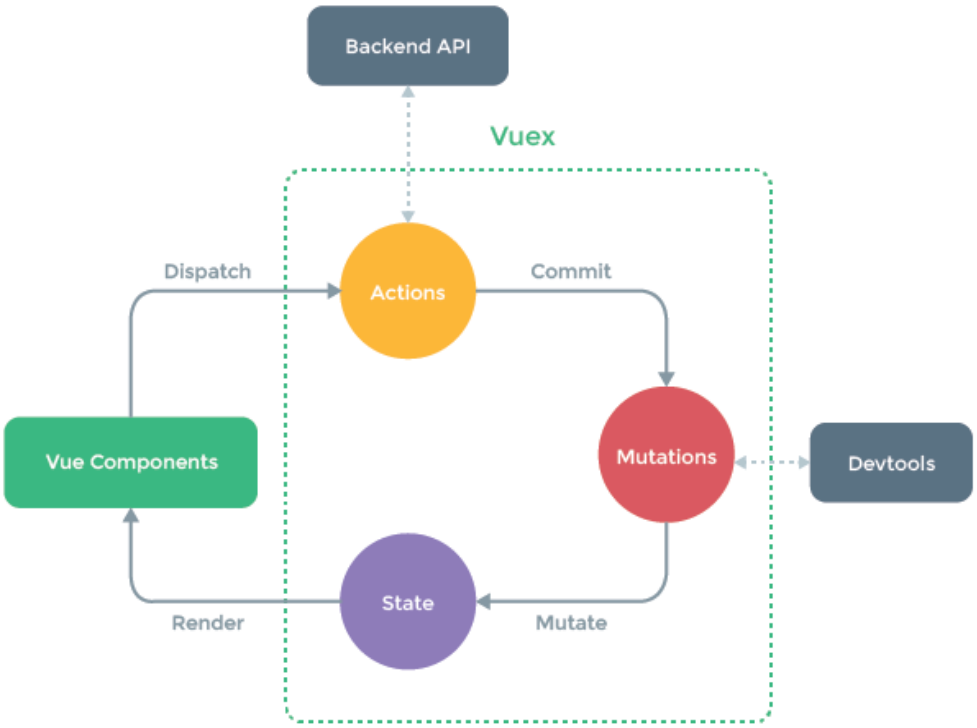
stávat velmi komplexním. V tuto chvíli je vhodné využit knihovnu Vuex. Jejím účelem je udržovat centrální stav proměnných, které jsou sdíleny napříč různými komponentami v aplikaci. Tohoto je docíleno díky dodržování návrhového vzoru jménem Flux [19].

Návrhový vzor Flux

Tato kapitola čerpá z [2].

Mezi základní pilíře návrhového vzoru Flux patří několik následujících pravidel, kterých se knihovna Vuex drží. Principy fungování tohoto návrhového vzoru jsou vyobrazeny na diagramu 2.3.

- **Jediný zdroj pravdy.** Data, která jsou sdílena více komponentami, jsou uložena na jednom místě - sklad (anglicky *store*) a jsou oddělena od komponent, které je využívají. Komponenty mohou stále mít svá lokální data, ale nesmějí udržovat svou vlastní kopii sdílených dat, ty se vždy musí číst přímo ze skladu.
- **Data pouze ke čtení.** Komponenty nesmějí přímo upravovat data ve skladu. V případě potřeby změny těchto dat sklad pouze informují a ten se sám o postará o provedení změny pomocí tzv. mutací (reprezentovány uzlem *Mutations* v grafu 2.3). Díky tomuto přístupu se minimalizuje šance nepředpokládaných změn sdílených dat a funkce, starající se o tyto změny, jsou přehledněji umístěné.
- **Změny dat jsou synchronní.** Asynchronní provádění operací mnohdy přináší spoustu výhod, v tomto případě je ale vyžadováno mutace provádět synchronně kvůli odstranění závislosti na pořadí a načasování různých událostí.



Obrázek 2.3: Diagram znázorňující životní cyklus dat v knihovně Vuex. *Převzato z [19].* Komponenty vysílají znamení o provedení akce (*Dispatch*). Tyto akce (*Actions*) jsou provedeny a následuje zápis změn (*Commit*). Tento zápis vede k mutacím (*Mutations* a *Mutate*), kde proběhne samotná úprava dat uložených na skladě. Tím je upraven stav aplikace (*State*) a pokud jsou na změněných datech závislé některé z komponent, proběhne jejich překreslení (*Render*).

BootstrapVue

Tato kapitola čerpá z [31].

BoostrapVue je systém klientské části webových stránek založený na systému Bootstrap, ten však pro svou plnou funkčnost vyžaduje použít také knihovnu jQuery. Pokud ale webová aplikace již pracuje se systémem Vue.js, mnohdy odpadá potřeba jQuery v aplikaci využívat. Vzhledem k faktu, že se jedná o poměrně rozsáhlou knihovnu, není žádoucí ji používat jen z důvodu, že je aplikace postavená na systému Bootstrap.

V tuto chvíli je vhodné využít alternativu jménem BoostrapVue, která závislost na knihovně jQuery nahrazuje závislostí na systému Vue.js. Systém BootstrapVue nabízí všechny komponenty původního systému, část z nich ještě vylepšuje a také nabízí komponenty úplně nové.

Mimo to, využití tohoto systému přináší větší pohodlí pro vývojáře. Dlouhý zápis některých komponent, tvořených pomocí množství vnořených prvků `<div>` lze díky Vue.js značně zjednodušit. Je využito funkce systému Vue.js nahradit prvek struktury HTML za

rozsáhlejší strukturu na základě šablony dané komponenty. Konvencí systému je takové prvky pojmenovávat s předponou `b-`, například tedy `b-button`. Přímo učebnicovým příkladem takového zjednodušení je zápis modálního okna, které se klasickým způsobem tvoří následovně:

```
<div class="modal" tabindex="-1" role="dialog">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modal title</h5>
        <button type="button" class="close" data-dismiss="modal"
        aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Modal body text goes here.</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary"
        data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Kód byl převzat z [6].

Pomocí BoostrapVue lze identickou komponentu zapsat mnohem kratším a elegantnějším způsobem a to konkrétně:

```
<b-modal title="Modal title">
  <p>Modal body text goes here.</p>
</b-modal>
```

2.5 Elasticsearch

Tato kapitola čerpá z [21].

Elasticsearch[21] je vyhledávač s otevřeným zdrojovým kódem umožňující distribuované vyhledávání a analýzu v reálném čase. Umožňuje velmi rychle pracovat s rozsáhlými daty (anglicky *big data*) a provádět nad nimi plnotextové vyhledávání, strukturované vyhledávání a nebo zmiňovanou analýzu.

Tento vyhledávač je hojně využíván jak menšími projekty, tak i světoznámými organizacemi. Za zmínku stojí například internetová encyklopedie Wikipedia[22], britský deník

[21] Elasticsearch: https://www.elastic.co/products/elasticsearch
[22] Wikipedia: https://www.wikipedia.org/

The Guardian[23], komunitní stránka pro výměnu rad programátorů StackOverflow[24] nebo populární webová služba pro verzovací nástroj Git[25] - GitHub[26].

Elasticsearch je postavený na knihovně Apache Lucene[27], využívá především její funkce plnotextového vyhledávání. Oproti této knihovně však systém Elasticsearch nabízí o mnoho přívětivější způsob používání, především pro začínající uživatele.

Systém pracuje s bezschémovým typem databázi. Oproti relačním databázím nenabízí propojovací dotazy, a proto je nutné ukládaná data denormalizovat.

2.5.1 Základní struktura

Tato kapitola čerpá z [10].

Základní struktura databáze Elasticsearch se na první pohled od tradičních relačních databázi velmi neliší, ke spoustě pojmů užívaných v systému Elasticsearch lze nalézt ekvivalent právě z tohoto typu databází.

Index

Skupina dokumentů s podobnou strukturou se nazývá index. Ekvivalentem v relačních databázích by v tomto případě byla tabulka.

Dokument

Dokument je základní jednotka dat, která může být indexována. Obsah dokumentu je zapsán ve formátu JSON (anglicky *JavaScript Object Notation*). V relační databázi lze dokument přirovnat k řádku tabulky.

2.5.2 Dotazy

Pro komunikaci s touto databází se využívá především Query DSL - dotazy aplikačního rozhraní REST (anglicky *Representational State Transfer*) ve formátu JSON.

Obsah dotazu může vypadat například následovně:

```
{
  "query": {
    "bool": {
      "should": [
        { "match": { "title": "Virtual Reality" }},
        { "match": { "title": "Augmented Reality" }}
      ],
      "filter": [
        { "term":  { "participant.country.keyword": "Czech Republic" }}
      ]
    }
  }
}
```

[23]The Guardian: https://www.theguardian.com/international
[24]StackOverflow: https://stackoverflow.com/
[25]Git: https://git-scm.com/
[26]GitHub: https://github.com/
[27]Apache Lucene: https://lucene.apache.org/

Uvedený dotaz požaduje od databáze výsledky obsahující ve svém nadpisu buď „Virtual Reality“ nebo „Augmented Reality“, jejichž partnerem byla některá z organizací sídlících v České Republice.

Na tento dotaz může server odpověď například zprávou začínající obsahem:

```
{
  "took" : 38,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 1515,
    "max_score" : 15.635861,
    "hits" : [
      {
        "_index" : "xstane34_projects",
        "_type" : "data",
        "_id" : "101785",
        "_score" : 15.635861,
        "_source" : {
          "startDate" : "2012-01-01",
          "endDate" : "2015-12-31",
          "reference" : "278169",
    ...
```

V úvodu odpovědi jsou uvedeny stručné statistiky vyhodnocení dotazu a pod klíčem `hits` se nachází informace o nalezených dokumenty. Ty jsou obohaceny kromě svého původního obsahu i o metadata označená podtržítkem v prvním znaku. Za zmínku stojí například položka `_score`, udávající skóre relevance.

Kontext

Tato kapitola čerpá z [13].

Dotazy v Elasticsearch se rozlišují na dva základní typy - kontext dotazu (*query*) a kontext filtru.

První z nich se dá lidsky přeložit do otázky „Jak moc dokument splňuje dotaz?“. V tomto případě se u každého výsledku počítá skóre relevance, podle kterého jsou ve většině případu ve výsledné odpovědi dokumenty sestupně řazeny.

Druhý typ dotazů, jak název vypovídá, je určen především k filtrování. V lidské řeči by se tento typ dotazů mohl zformulovat do věty „Splňuje dokument dotaz?“. Vyhodnocení tohoto typu dotazů je mnohem rychlejší, díky absence nutnosti počítat skóre relevance. Mimo to jsou takovéto dotazy automaticky ukládány do mezipaměti, což výrazně urychluje jejích opětovné využití.

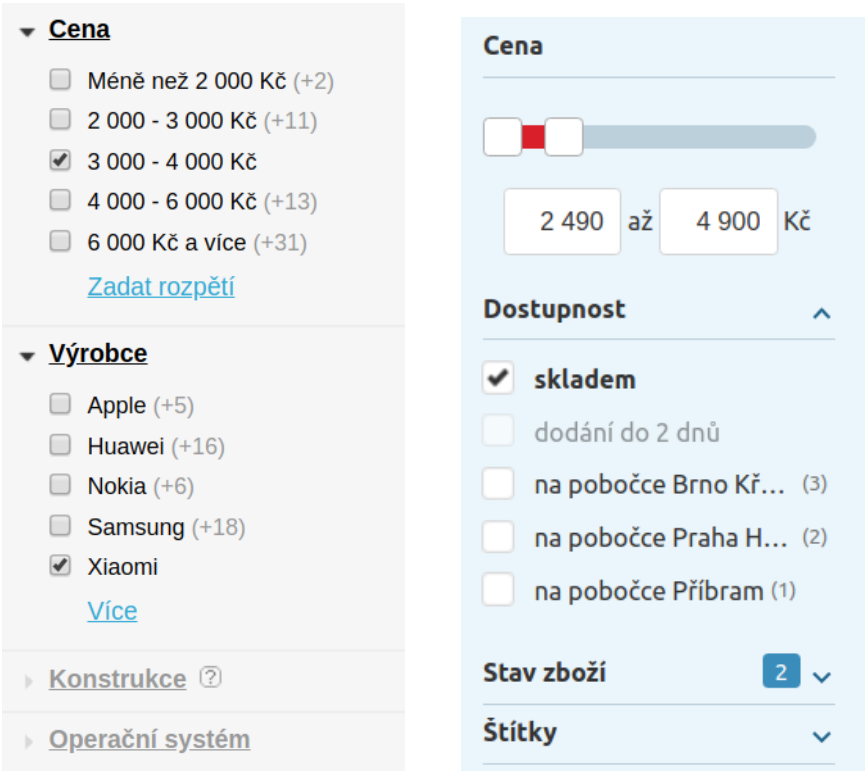
V jednom složeném dotazu je možné oba tyto typy kombinovat.

2.5.3 Fasetové vyhledávání

S velkým množstvím dat přichází potřeba obsah dle určitých kriterií zúžit. K tomuto účelu slouží filtry a fasetové vyhledávání (někdy také označováno jako fastová navigace - anglicky *faceted search* nebo *faceted navigation*). Využití faset je pokročilejší než použití filtrů, jelikož fasetová navigace umožňuje využít hned několik filtrů najednou [23].

V systému Elasticsearch je možné tuto navigaci vytvořit pomocí tzv. kyblíkových agreagací (anglicky *bucket aggregations*). Jednoduše řečeno se pro dokumenty vytvoří kyblíky, kde každý jeden odpovídá určitému kriteriu. Pokud dokument toto kriterium splňuje, je do kyblíku vložen [11].

S tímto typem vyhledávání se lze setkat velmi často v seznamech produktů například v internetových obchodech nebo cenových srovnávačích.



Obrázek 2.4: Příklad využití fasetového vyhledávání na cenovém srovnávači Heureka[30] (vlevo) a internetovém obchodě CZC.cz[31] (vpravo).

2.5.4 Plnotextové vyhledávání

Tato kapitola čerpá z [21].

Uložená textová data v dokumentech se dělí na dva typy - přesné hodnoty a plnotextové hodnoty (anglicky *full-text*). Přesné hodnoty jsou určeny pro data, jako například identifikátor uživatele. Pokud totiž vyhledáváme uživatele s identifikátorem „jiri furda“, jako výsledek neočekáváme uživatele, který má identifikátor „martin furda“, ale očekáváme výsledek, který bude přesně splňovat hledanou položku. Pokud ovšem člověk pracuje s textovými položkami, nemusí nutně hledat stoprocentní shodu. Lidský jazyk je velice rozmanitý a umožňuje slova časovat, skloňovat, či vyjádřit jednu věc vícero různými slovy. V těchto případech hledání přesných hodnot selhává, i když má text z pohledu člověka

stejný nebo podobný význam. Pro tento typ hledání slouží právě plnotextové hledání. Můžeme se s ním setkat například při hledání článku v internetovém magazínu. Elasticsearch pro takové hledání musí text nejprve analyzovat a vytvořit pro něj obrácené indexy.

Obrácený index

Obrácený index (anglicky *inverted index*) je datová struktura obsahující seznam všech unikátních slov vyskytujících se v uložených dokumentech. Pro každé z těchto slov uchovává informaci, ve kterých dokumentech bylo obsaženo. Díky tomu je možné v textových položkách rychle provádět plnotextové hledání.

Příkladem mohou být dva dokumenty, kde každý obsahuje právě jednu z následujících vět:

- Na moskevském letišti se stala havárie letadla.
- Moskvu silně zasáhla včerejší havárie na letišti.

Zpracováním těchto dvou dokumentů tedy může vzniknout následující obrácený index:

Token	Dokument č. 1	Dokument č. 2
Moskvu	Ne	Ano
Na	Ano	Ne
havárie	Ano	Ano
letadla	Ano	Ne
letišti	Ano	Ano
moskevském	Ano	Ne
na	Ne	Ano
se	Ano	Ne
silně	Ne	Ano
stala	Ano	Ne
včerejší	Ne	Ano
zasáhla	Ano	Ne

Tabulka 2.1: Ilustrační příklad obraceného indexu pro dva různé dokumenty.

Takovýto index by byl ale v praxi nepoužitelný. Hledání slova „havárie“ by sice za výsledek označilo oba dva dokumenty, ale ku příkladu hledání fráze „letiště Moskva“ by v tomto případě nenalezlo žádnou shodu, i přesto, že z lidského pohledu se oba dva dokumenty hledané fráze týkají. Z tohoto důvodu text nestačí pouze strojově rozdělí, ale musí proběhnout kompletní analýza textu.

Analýza

Analýza představuje proces, kdy se kus textu rozdělí do jednotlivých tokenů a následně se tyto tokeny upraví do podoby vhodné pro použití plnotextového vyhledávání. Tento proces obstarává analyzér, který se skládá ze tří části:

- **Filtry znaků** upraví zpracovávaný řetězec ještě před samotným rozdělením do tokenů. Jeho úkolem může být například odstranění HTML značek z textu.
- **Tokenizér** z textu sestaví jednotlivé tokeny. Tokeny ve většině případů představují jednotlivá slova, ale nemusí tomu tak vždy být.

- **Filtry tokenů** na konec projde všechny vytvořené tokeny. Některé může odstranit, přidat a nebo pouze upravit.

Filtrů může být v jednom analyzéru použitá celá řada, a to jak filtrů znaků, tak i filtrů tokenů. Tokenizér je ale v rámci analyzéru vždy použit jen jeden jediný.

Elasticsearch již ve svém základu nabízí škálu různých analyzéru. Tyto analyzéry jsou různé složitosti, mezi ty nejjednodušší patří analyzér bílých znaků, který text rozdělí striktně podle mezer, tudíž je i tečka na konci věty vnímána jako část slova. Nejčastěji se ale člověk může setkat s jazykovým analyzérem, který je uzpůsoben přímo dané lidské řeči. Elasticsearch takový analyzér nabízí pro mnoho světových jazyků mezi které patří i jazyk český.

Filtry tokenů

Tato kapitola čerpá z [26].

Velmi častou filtrací bývá změna velkých písmen na malé, díky tomu nejsou slova na začátku věty vnímána jako jiná slova, než ty v jiné části věty. Pro český jazyk je také typické odstranění diakritiky.

Součástí analyzátorů bývá často odstranění tokenů, které obsahují nepodstatná slova. Jedná se především o spojky a předložky. Taková slova se v plnotextovém vyhledávání označují jako stop slova (anglicky *stop words*). Tento seznam je vždy specifický pro konkrétní jazyk textu, použít výchozí anglická stop slova nad česky psaným textem by ztrácelo význam.

Další nedílnou součástí pokročilejších analyzátorů je *stematizace*, aneb získání kmene slova. Pro tuto operaci lze využit dva způsoby - algoritmus nebo slovník. Každý z těchto způsobů má své plusy i mínusy.

Při využití algoritmické stematizace sice analyzátor nemusí znát všechna slova v daném jazyce, protože slova převádí do požadovaného tvaru pouze na základě sady pravidel, na druhou stranu ale dochází k větší nepřesnosti. Elasticsearch v případě využití české algoritmické stematizace ze slov odstraňuje všechny přípony.

Druhým způsobem, který lze využít je stematizace na základě slovníku. Díky tomu jsou převedené tvary přesnější než ty získané algoritmem. Hlavním předpokladem v této metodě je ovšem využití vhodného slovníku s dostatečnou zásobou slov.

2.5.5 Knihovny

Společnost Elastic[32] také vyvíjí několik oficiálně podporovaných nízkoúrovňových klientů systému Elasticsearch, určených pro mnoho populárních programovacích jazyků. Patří mezi ně například Java, JavaScript, Go, .NET, PHP, Perl nebo Python [12]. Klientů vyvíjených komunitou lze samozřejmě nalézt o mnoho více[33].

Práce s nízkoúrovňovými klienty pro programátory nebývá příliš pohodlná, z tohoto důvodu vznikají komunitou vyvíjení klienti, kteří nabízí vysokoúrovňový přístup. Konkrétně pro programovací jazyk Python byl do roku 2015 společností Mozilla[34] vyvíjen klient jménem ElasticUtils[35]. Za nástupce toho klienta je považována knihovna Elasticsearch DSL[36] [27].

[32] Elastic: https://www.elastic.co/

[33] Komunitní klienti Elasticsearch: https://www.elastic.co/guide/en/elasticsearch/client/community/current/index.html

[34] Mozzila: https://www.mozilla.org/

[35] ElasticUtils: https://elasticutils.readthedocs.io/en/latest/

[36] Elasticsearch DSL: https://elasticsearch-dsl.readthedocs.io/en/latest/

Elasticsearch DSL

Tato kapitola čerpá z [22].

Elasticsearch DSL je knihovna pro programovací jazyk Python. Jedná se o vysokoúrovňovou knihovnu postavenou na oficiální nízkoúrovňové knihovně Python Elasticsearch Client[37]. Díky tomu lze s dokumenty v databázi Elasticsearch pracovat jako s objekty, ale zároveň je stále možné použít původní přístup oficiálního klienta ku příkladu ke zjištění zdraví shluku.

Jako příklad mějme následující dotaz, napsaný s využitím oficiální knihovny Python Elasticsearch Client:

```
from elasticsearch import Elasticsearch
client = Elasticsearch()

response = client.search(
{
  index="xstane34_projects",
  body={
    "query": {
      "bool": { "must": { "title": "Virtual Reality" }},
        "filter": [
          { "term":  { "participant.country.keyword": "Czech Republic" }}
        ]
      }
    },
    "aggs": {
        "coordinator_coutry": {
          "terms": {"field": "coordinator.country.keyword"},
        }
      }
  }
```

Tento poměrně zdlouhavý dotaz lze se stejným výsledkem přepsat za použití knihovny Elasticsearch DSL do následující, mnohem elegantnější formy:

```
from elasticsearch import Elasticsearch
from elasticsearch_dsl import Search

client = Elasticsearch()

s = Search(using=client, index="xstane34_projects")  \
    .query("match", title="Virtual Reality")  \
    .filter("term", coordinator__country__keyword="Czech Republic")

s.aggs.bucket("coordinator_coutry", "terms",  \
    field="coordinator.country.keyword")

response = s.execute()
```

[37] Python Elasticsearch Client: https://elasticsearch-py.readthedocs.io/en/master/

Práce s dotazy je tedy mnohem snadnější a vývojářům alespoň částečně ubývá rutinní činnost. Obrovskou výhodou je možnost sestrojení dotazu pomocí zřetězených funkcí. Díky této funkci knihovny lze dotaz v průběhu programu jednoduše rozšiřovat a kód se tak nemusí stát těžko čitelným.

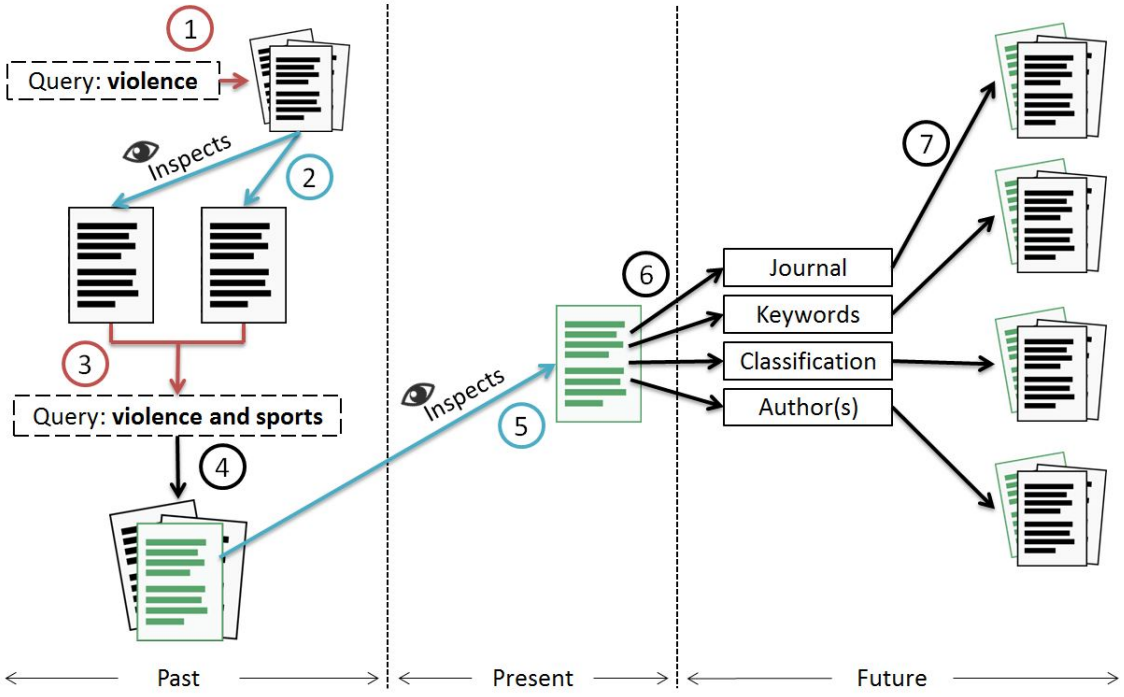
Objekt dotazu je v knihovně reprezentován třídou `Search`. Tu lze ale kdykoliv převést do slovníku odpovídajícímu klasickému dotazu ve formátu JSON pomocí metody `to_dict()`. Stejně tak je možné dotaz napsaný ve slovníku převést do objektu knihovny Elasticsearch DSL použitím metody `from_dict()`. Tato funkcionalita umožňuje bez nutnosti zásahu do stávajícího kódu zpětnou kompatibilitu s aplikacemi využívajícími například jen nízkoúrovňového klienta Python Elasticsearch Client.

2.6 Kontextové procházení

Tato kapitola čerpá z [7].

Pro zobrazení relevantnějších výsledků vyhledávání lze využít kontext hledání uživatele. V takovém případě se při počítání skóre relevantnosti nepracuje pouze s aktuálním hledáním, ale i s historií uživatele. Díky tomu je docíleno přizpůsobení výsledků přímo pro daného uživatele. Pro tyto účely je možné brát v potaz předchozí hledání nebo podobnost s dříve otevřenými dokumenty.

Výhodou kontextového hledání je zpřesnění výsledků hledání bez nucení uživatele manuálně toto hledání blíže specifikovat. O takovou možnost ovšem uživatel nepřijde, tento typ hledání se s použitím fasetové navigace nijak nevylučuje.



Obrázek 2.5: Schéma kontextuálního hledání. Převzato z [7].

Na obrázku 2.5 uživatel v prvních dvou krocích hledal pojem „violence“ a prohlédl si dva výsledky. Poté v kroku 3 hledání rozšířil a vyhledal pojem „violence and sports“.

Z výsledků v kroku 4 zvolil dokument, který si zobrazí podrobněji. Tento proces probíhá již v současnosti a je reprezentován krokem 5. Krok 6 tvoří kontext pro budoucí hledání. V kroku 7 jsou znázorněna další hledání, ve kterých budou při počítání skóre relevance brána v potaz metadata z právě zobrazeného dokumentu z kroku 5.

Kapitola 3

Návrh a implementace systému

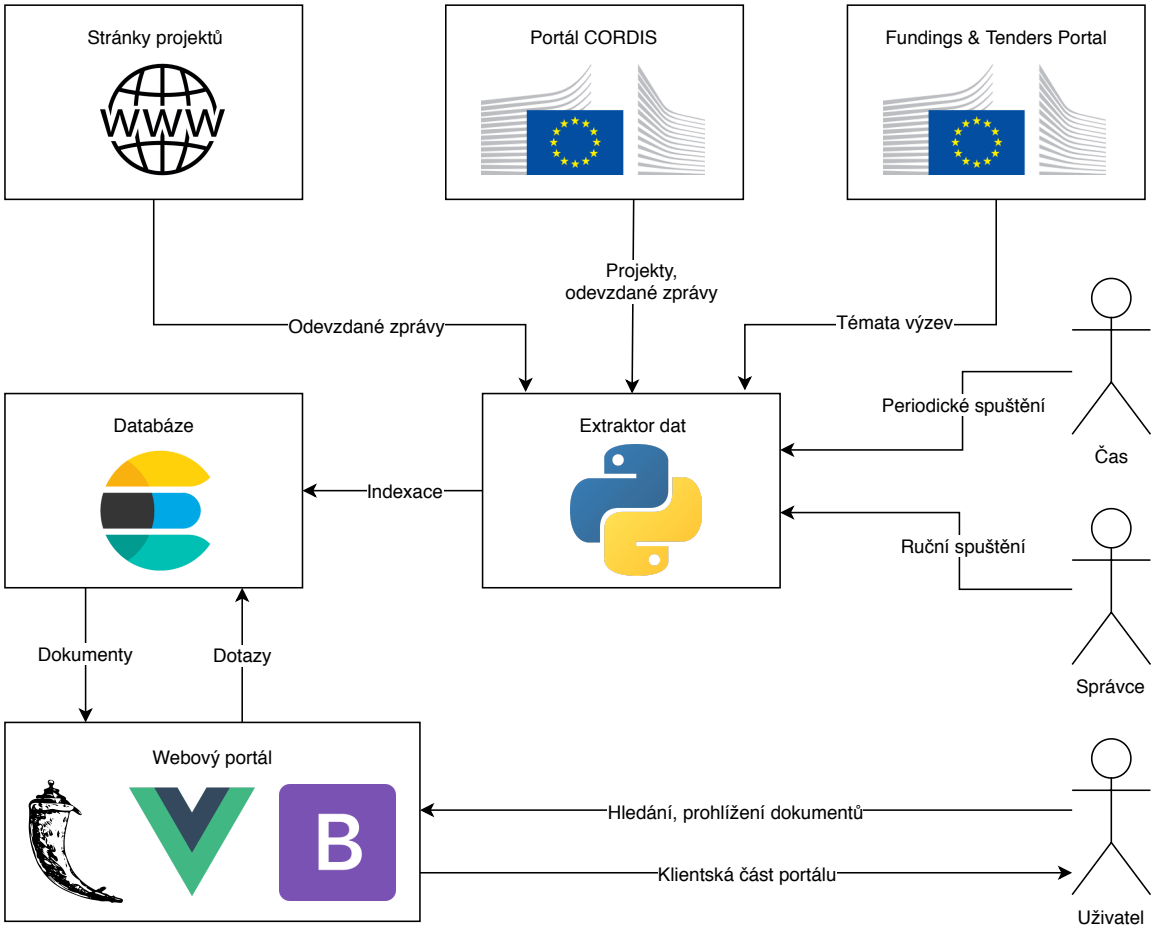
Tato kapitola popisuje můj přístup k řešení daného systému, popisuje principy fungování systému a konkrétně rozebírá jeho implementaci.

Předchozí řešení mých kolegů umožňovalo vyhledávat pouze projekty a vědecké zprávy. Nabízelo se tento obsah rozšířit o výzvy k předkládání návrhů, na základě kterých Evropská komise vybírá projekty, které budou uskutečněny se spolufinancováním Evropské unie [14].

Samotné výzvy jsou ale hodně obecné a zaštiťují mnoho témat napříč různými oblastmi. Dalším úskalím byl nedostatek informací v popisu výzev, rozsáhlý slovní popis se vztahoval až ke konkrétním tématům. Proto jsem se rozhodl portál nerozšiřovat o výzvy, ale právě o zmiňovaná témata.

3.1 Schéma systému

Schéma jednotlivých částí systému a jejich návaznost je možné vidět na obrázku 3.1. Ve vrchní části obrázku jsou znázorněny zdroje dat, které jsou zpracovávány extraktorem dat. Extrakce probíhá bez zásahu správce periodicky za pomoci plánovače úloh (anglicky *cron*). Správce má však možnost extrakci spustit i manuálně. Zpracovaná data jsou ukládána do databáze, odkud je čerpá webový portál a následně je prezentuje uživateli. V blocích modulů systému („databáze“, „extraktor dat“ a „webový portál“) jsou znázorněna loga klíčových technologií využitích pro fungování daného modulu.



Obrázek 3.1: Schéma jednotlivých částí vyvíjeného systému.

3.2 Extraktor dat

Extrakce dat získává tři typy informací. Shromažďuje témata výzev, projekty a odevzdávané zprávy. Většina informací je čerpána přímo z portálu Evropské komise, část dat se ale získává i přímo ze stránek jednotlivých účástníků.

3.2.1 Témata výzev

Témata výzev jsou čerpána z portálu Fundings and Tenders. Ten pro podobné účely nabízí aplikační rozhraní, kde je možné získat potřebná data ve formátu JSON, čímž bylo značně zjednodušeno jejich dolování.

Pro indexaci dat z toho portálu je nejprve použit dotaz pro získání souhrnného seznamu témat. Ten je následně extraktorem postupně celý zpracován. Z toho seznamu lze získat základní informace o tématech. Pro každé jedno téma je odeslán nový dotaz, který na základě identifikátoru vrátí podrobnější informace vztahující se k tématu, jako například jeho slovní popis.

The Funding & Tenders Portal

The Funding & Tenders Portal[1] je webový portál spravován převážně Evropskou komisí. Portál zprostředkovává informace určené hlavně expertům a účastníkům v programech financovaných EU [14]. V případě zájmu o zažádání financování výzkumu je nutné nejprve prostřednictvím toho portálu najít výzvu k předkládání návrhů v odpovídajícím odvětví a dodržet konkrétní postup pro podání žádosti [16].

3.2.2 Projekty a odevzdávané zprávy

Informace o projektech jsou čerpány z portálu CORDIS, stejně jako většina odevzdávaných zpráv. Některé další zprávy jsou získávány přímo ze stránek jednotlivých projektů. Pro získání a zpracování těchto dat je využit extraktor dat z bakalářské práce, jejíž autorem je Petr Staněk [36].

CORDIS

CORDIS[2] (*The Community Research and Development Information Service*) je portál provozovaný Úřadem pro úřední tisky Evropské Unie sloužící jako hlavní zdroj výsledků projektů, sponzorovaných v rámci programů EU pro výzkum a inovaci. Na jednom místě veřejně poskytuje informace jak o těchto projektech, tak i o jejích účastnících, vědeckých zprávách a publikacích [15].

3.3 Webový portál

Výsledný webový portál umožňuje uživateli prohlížet a vyhledávat data získaná extraktorem dat. V této kapitole je výsledný portál podrobně rozebrán.

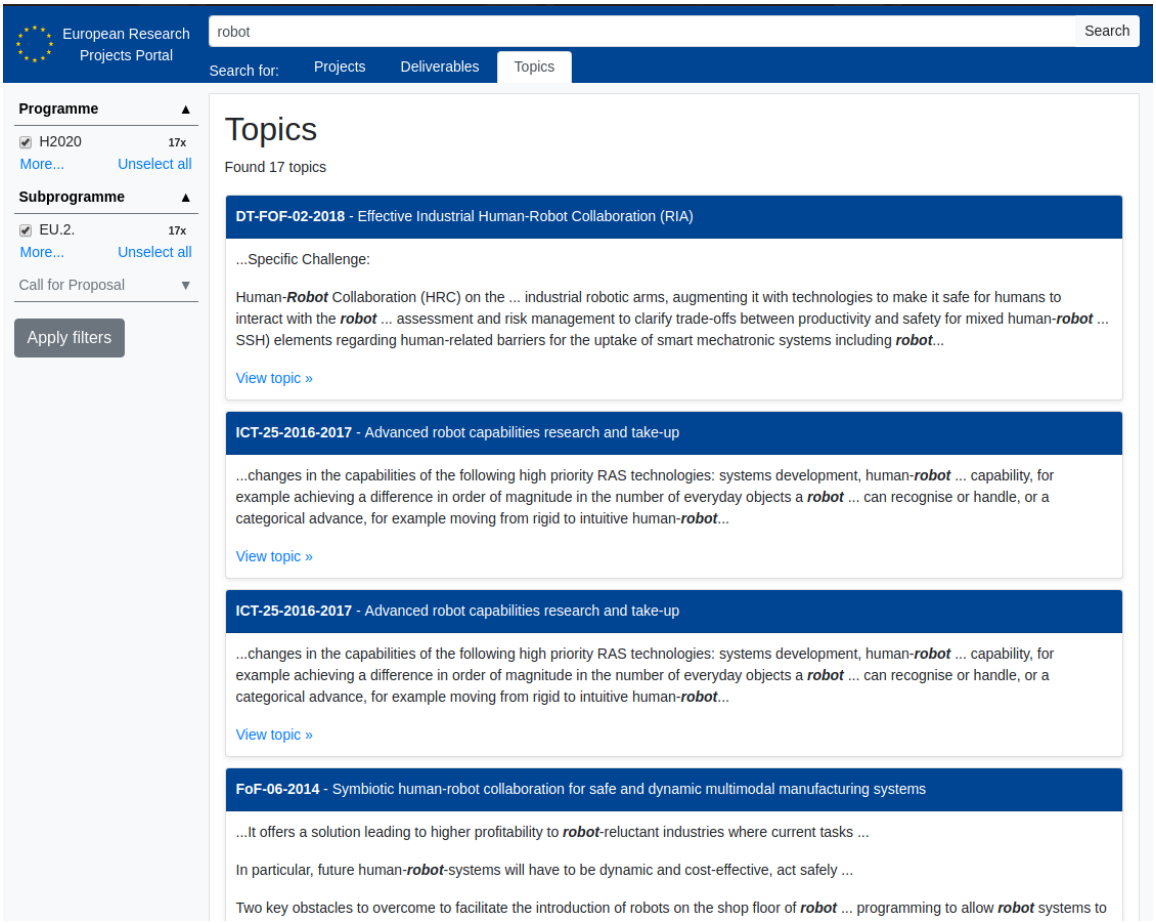
3.3.1 Vizuální podoba

Vizuální struktura vytvořeného webového portálu lze rozdělit do tří hlavních celků - hlavičky, postranního panelu a hlavního obsahu stránky. První dva z těchto celků jsou určeny pro specifikaci vyhledávání. Aby bylo vyhledávání při procházení portálu vždy dostupné, pozice těchto celků ve verzi pro stolní počítače zůstává neměnná i při rolování zobrazené stránky.

Portál obsahuje obecně dva typy stránek. Prvním typem jsou stránky obsahující výsledky vyhledávání, které lze vidět na snímku 3.2. V případě úspěšného hledání lze z těchto výsledků pomocí odkazů „View topic“ nebo „View project“ přejít na detailní stránku konkrétního výsledku. Tento typ stránek ovšem existuje pouze pro témata a projekty. U odevzdaných zpráv se místo stránky detailu zobrazí přímo daný dokument. Příklad stránky detailu výsledku je vyobrazen na snímku 3.3.

[1] The Funding & Tenders Portal: https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/home

[2] CORDIS: https://cordis.europa.eu/



Obrázek 3.2: Stránka vytvořeného portálu zobrazující výsledky hledání výzev obsahující klíčové slovo „robot“, financované v rámci programu „H2020“ a podprogramu „E.2.“

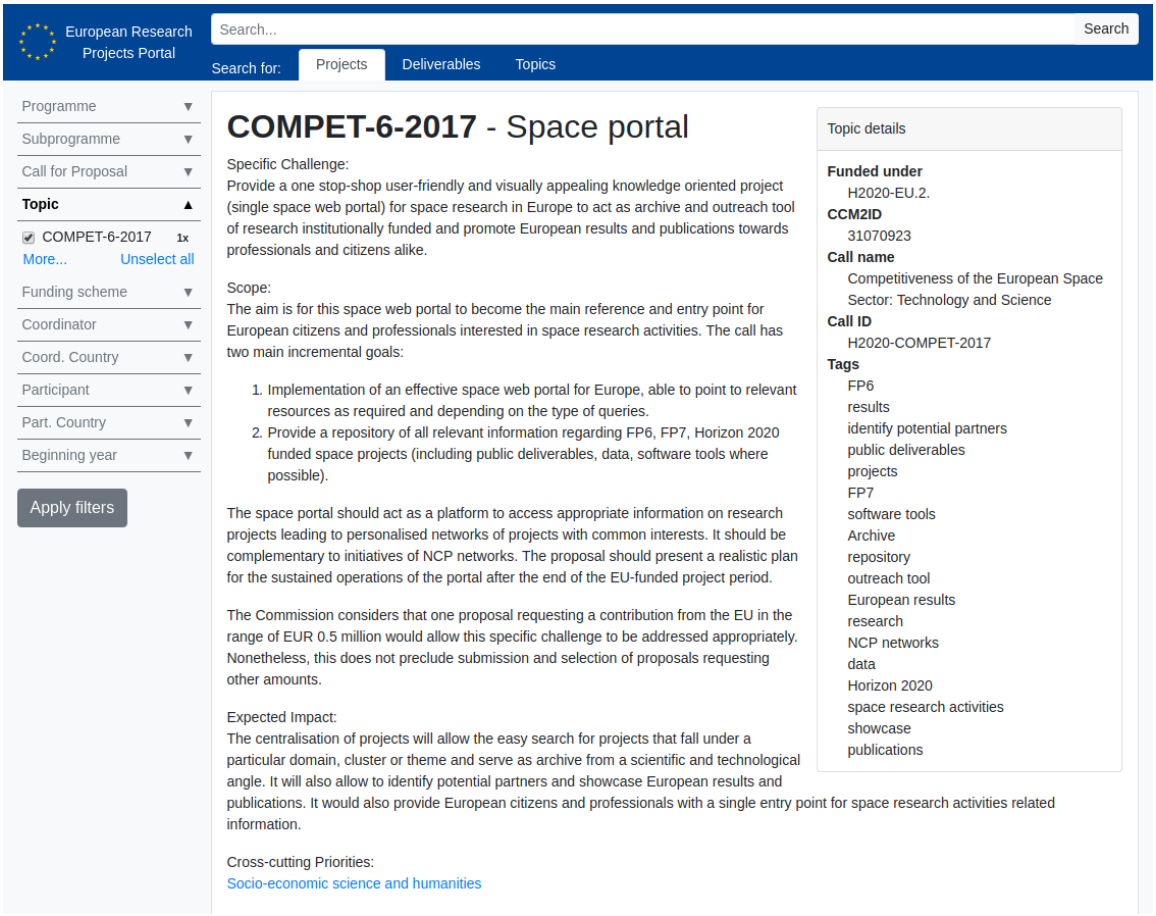
.

1. Logo a název portálu se dle konvence nachází v levém horním rohu. Zároveň funguje jako odkaz na hlavní stránku.

2. Zbylá část hlavičky obsahuje vyhledávací pole. Do něj může být vložen i velmi komplexní dotaz, proto byl při návrhu kladen důraz na co největší šířku tohoto pole. Pod polem se nachází odkazy určující jaký typ výsledků je od vyhledávání očekáván.

3. V levé části portálu se nachází postranní panel s fasetovou navigací. Slouží především pro méně pokročilé uživatele portálu, kterým postačí tvořit jednoduché dotazy. Zároveň lze informace v tomto panelu využít jako přehled informací typu „Které země se nejčastěji podílí na projektech splňujících dané kriteria?“. Postranní panel je podrobněji popsán v kapitole 3.3.3.

4. Největší část stránky je věnována právě výsledkům daného hledání. Pokud žádné hledání prozatím neproběhlo, stránka místo výsledků vyhledávání prezentuje data s nejpozdějším datem obnovy.



Obrázek 3.3: Snímek vytvořeného portálu zobrazující detailní popis výzvy k předkládání návrhů na téma „Space portal“.

Základní struktura tohoto typu stránek je sdílena se snímkem 3.2. Místo výsledků vyhledávání však prezentuje informace o konkretním tématu výzvy. Tyto informace se skládají z následujících částí:

1. Plné znění obsahu výzvy a instrukce pro předkládání návrhů na toto téma.
2. Klíčové specifikace dané výzvy stručně zobrazené v postranní kartě (vpravo).

Mimo to má postranní panel s fasetovým vyhledáváním ve výchozím stavu pro fasetu „Topic“ zvolenou hodnotu právě zobrazeného tématu. Díky tomu lze snadno provést hledání projektů nebo odevzdaných zpráv právě s tímto tématem.

3.3.2 Struktura adresáře

Veškeré zdrojové kódy portálu se nachází v adresáři `portal`. Uvnitř této složky jsou obsaženy adresáře odpovídající architektuře MVC uvedené v kapitole 2.1.1.

- Složka `models` obsahuje modely.
- Složka `controllers` obsahuje kontrolery.
- Složka `templates` obsahuje pohledy.

- Složka `static` obsahuje kaskádové styly a zdrojové kódy Vue komponent.
- Soubor `app.py` slouží ke spuštění portálu.

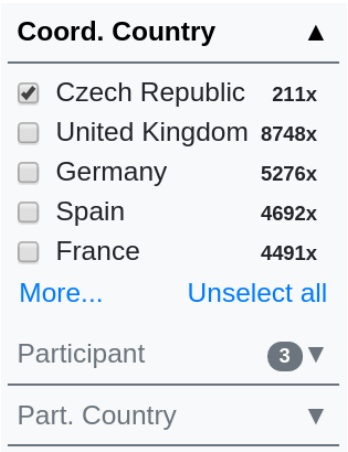
3.3.3 Komponenty systému Vue.js

Některé z využívaných faset obsahují i stovky položek a jejich hodnoty mají často dlouhé názvy. Jejich zobrazení pouze v postranním panelu jsem považoval za nedostatečné, proto jsem se rozhodl toto řešení oproti předchozímu řešení portálu přepracovat s využitím technologie Vue.js (představenou v kapitole 2.4) a několika prvků ze systému BootsrapVue. Pro každou z faset jsem vytvořil dvě komponenty - položku v postranním panelu a modální okno.

Postranní panel

Tento panel je tvořen Vue komponentou `sidebar-facet-list`, která pro každou fasetu vytvoří instanci komponenty `sidebar-facet`. Několik takových komponent lze vidět na snímku 3.4.

Každá komponenta `sidebar-facet` ve výchozím stavu nabízí nejvýše pět nejčastěji vyskytujících se hodnot dané fasety. Ty jsou postupně nahrazovány zvolenými hodnotami dané fasety. Další hodnoty s menším počtem výskytů jsou dostupné v modálním okně, které se zobrazí po kliknutí na tlačítko „*More...*“. Obsahem této komponenty je mimo jiné dílčí komponenta systému BoostrapVue `b-collapse`, která v případě potřeby umožňuje danou fasetu minimalizovat. Pokud se takto stane a faseta má zvolené některé ze svých možností, objeví se vedle jejího názvu počet vybraných možností, aby měl uživatel i při minimalizovaných fasetách přehled o tom, které možnosti filtrování právě zvolil.



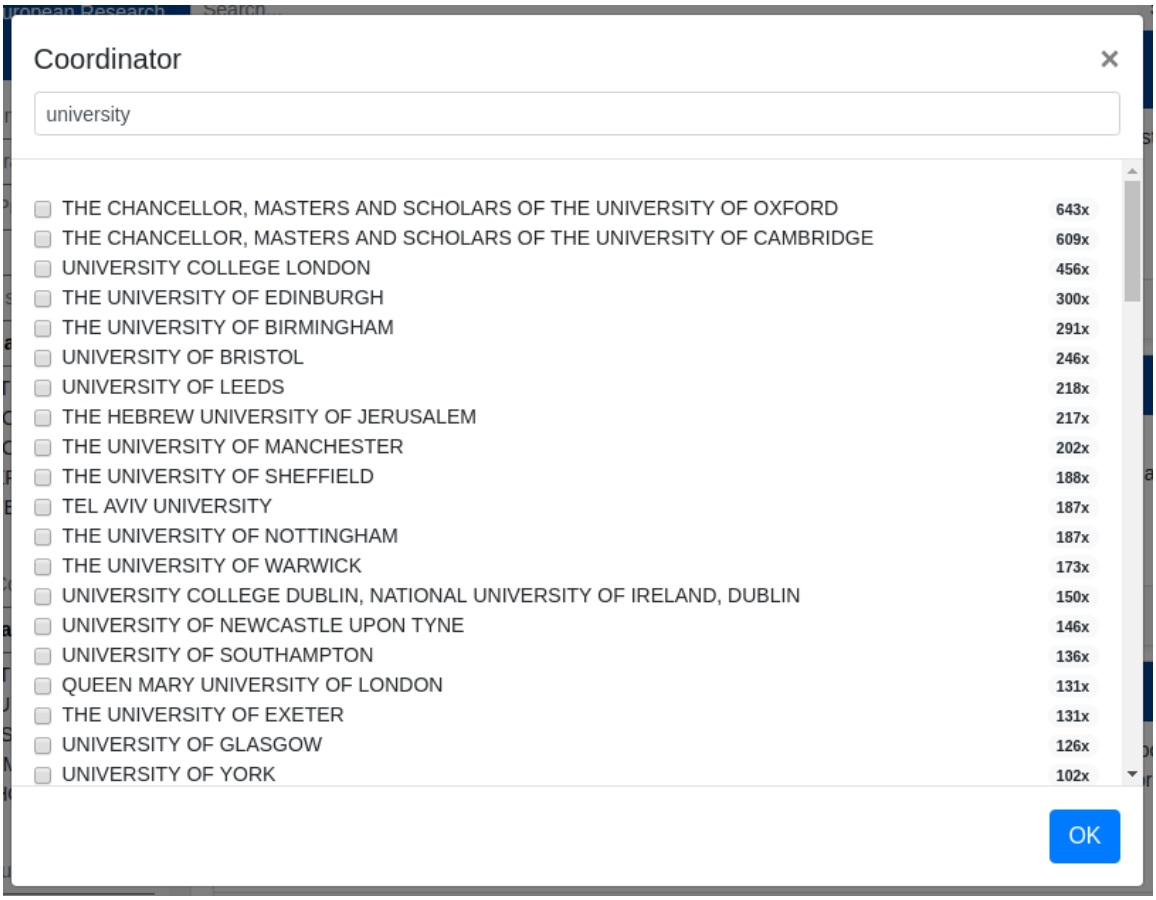
Obrázek 3.4: Část postranního panelu implementovaného portálu

1. Otevřená faseta „Coord. Country“ s jednou zvolenou možností a dalšími čtyřmi nejčastěji vyskytujícími se hodnoty.
2. Minimalizovaná faseta „Participant“ se třemi vybranými možnostmi indikovanými číslem v pravé části fasety.
3. Minimalizovaná faseta „Part. Country“ bez zvolených možností.

Modální okno

Modální okna jednotlivých faset jsou tvořena instancemi komponenty `modal-facet`. O jejich vytvoření se stará komponenta `modal-facet-list`. Příklad konkretního modálního okna je možné vidět na snímku 3.5.

Rozsáhlejší výpis možností dané fasety se nyní oproti původnímu řešení portálu zobrazuje v modálním okně. Zde je možnost současně vidět podstatně více možností a také je možné vypsat jejich hodnoty v plném znění. V tomto okně je také obsaženo vstupní pole, které asynchronními dotazy na databázi Elasticsearch vyhledává mezi možnostmi daného fasetu. Samotné modální okno je komponenta systému BootstrapVue, rozšířená o prvky nutné pro tento portál (například zmiňované vyhledávací pole).



Obrázek 3.5: Modální okno s dalšími hodnoty fasety s názvem „Coordinator“

.

1. Nadpis značící, ke které fasetě se modální okno vztahuje.
2. Vstupní pole pro vyhledávání v hodnotách fasety.
3. Fasetové hodnoty odpovídající hledané frázi z bodu 2. Výsledky se mění v reálném čase v závislosti na obsahu tohoto vstupního pole.

Fasetová možnost

Jednotlivé fasetové možnosti jsou reprezentovány komponentou `option-facet`, která zobrazuje zaškrtávací pole, název dané možnosti a počet výsledků aktuálního vyhledávání obsahující tuto možnost. Při zaškrtnutí možnosti se její data zkopírují do proměnné `checkedOptions` příslušné fasety. Tato komponenta se vyskytuje jak v modálním okně, tak i v postranním panelu.

3.3.4 Implementace vyhledávání

Samotné vyhledávání obstarává třída `IndexSearch`, při svém vytvoření uloží vstupní parametry a sestaví dotazy pro Elasticsearch. Mezi tyto parametry patří

- index, ve kterém se bude hledat.
- pole dokumentu, ve kterém budou v případě shody zvýrazněna klíčová slova.
- pole dokumentu, ve kterých se budou klíčová slova hledat.

buildSearch()

První z dotazů je vytvořen tuto metodou, která zpracuje všechny parametry typu GET protokolu HTML a v případě shody jejich názvů s některým z faset, přidá tuto hodnotu do dotazu. Takový dotaz se využívá pro hledání v modálním okně fasetu.

buildAggregationsSearch()

Druhý dotaz vzniká rozšířením dotazu z metody `buildSearch()`. Je obohacen o agregaci fasetových hodnot, které jsou poté zobrazeny v postranním panelu.

prepareLayoutData()

Poté, co byly výše zmíněné dotazy vykonány, je třeba výsledky zpracovat a připravit pro předání do odpovídajících komponent. Výsledkem této funkce jsou tři struktury formátu JSON zabalené ve slovníku, obsahující data využívané pro inicializaci komponent v šabloně.

Jedna ze struktur, která je poté uložena do skladu knihovny Vuex může vypadat například následovně:

```
{
    "facets":[
    {
        "mostFrequentOptions":[
            {
               "count":8748,
               "text":"United Kingdom",
               "value":"United Kingdom"
            },
            {
               "count":5276,
               "text":"Germany",
               "value":"Germany"
```

```
            }
         ],
         "field":"coordinator.country.keyword",
         "name":"coordcountry",
         "checkedOptions":[
            {
               "count":4692,
               "text":"Spain",
               "value":"Spain"
            },
         ],
         "title":"Coord. Country"
      }
      ]
}
```


Každá z faset má svůj objekt obsažený v poli s klíčem `facets`.

- `mostFrequentOptions` obsahuje pole s nejvýše pěti nejčastějšími hodnotami fasety.
- `name` značí unikátní interní název, díky kterému je portál schopný fasetu identifikovat.
- `checkedOptions` je pole, kde se udržují zvolené hodnoty fasety.
- `title` nese název fasety, který je zobrazován uživateli.

Každá z hodnot faset je zase reprezentována objektem o třech atributech:

- `count` udává množství výsledků hlavního hledání s výskytem této hodnoty fasety.
- `text` je název hodnoty, která se zobrazuje uživateli.
- `value` obsahuje reálnou hodnotu, která se aplikuje do dotazu hledání. Od atributu `text` se mnohdy neliší, ale je třeba mít tyto dvě informace rozdělené například pro použití možností s názvem sloučených ze dvou polí databáze (příkladem může být název a zkratka instituce).

createForIndex()

Protože se v portálu vytváří hledání na vícero místech, vytvořil jsem statickou metodu třídy `IndexSearch`, která vrací předem připravenou instanci této třídy, určenou pro hledání v jednom z typů obsahu portálu. Tato funkce očekává ve svém jediném parametru řetězec, určující typ hledání (`projects`, `deliverables` nebo `topics`)

Hromadné akce

Pro případy, kdy je potřeba pracovat s několika hledáními naráz, si lze práci ušetřit využitím dalších statických metod třídy `IndexSearch` a to konkrétně metoda `createForEveryIndex()`, která vrací slovník se všemi třemi možnými typy hledání, a poté také `executeMany()`, která vykoná hledání všech instancí `IndexSearch` ve slovníku předaném této funkci v parametru.

3.3.5 Vyhledávání hodnot faset

Protože ve fasetách lze asynchronně vyhledávat, bylo nutné napsat aplikační rozhraní propojující databázi Elasticsearch s komponentou systému Vue.js reprezentující toto vyhledávání. Tuto činnost obstarává funkce `showApi()` v kontroleru `facet_controller`.

Jako parametr, který je předán cestou URL adresy očekává platný název fasety, pro kterou existuje model třídy `Facet`. Další parametry jsou předávány metodou GET protokolu HTTP. Prvním povinným parametrem je znění původního dotazu na Elasticsearch. Díky tomuto parametru výsledky funkce korespondují s kontextem aktuálního vyhledávání v portálu. Dalším parametrem je pak klíčové slovo nebo jeho část, které se hledá v dostupných hodnotách fasety.

Po úspěšném dotazu funkce vrací strukturu formátu JSON ve stejném tvaru jako například v `checkedOptions` zmiňovaném v kapitole 3.3.4.

3.3.6 Dotazovací jazyk

Pokročilí uživatelé portálu mají možnost při vyhledávání sepsat složitější dotazy. Toho je docíleno díky využití dotazu typu `query_string`, který nabízí systém Elasticsearch. V takovém dotaze je možné využít například logické operátory `AND` a `OR`, žolíkový znak `?`, který může zastupovat libovolný řetězec nebo žolíkový znak `*`, který se chová obdobně, ale připouští i variantu prázdného řetězce. Dále jsou k dispozici kulaté závorky, díky kterým lze části dotazu seskupit či dokonce vnořit. Další speciální funkci má znaménko `-`, díky kterému lze například označit, které slovo se ve výsledcích nemá objevit, oproti tomu znaménko `+` daný výraz ve výsledku vyžaduje. Text v uvozovkách zase označuje hledání přesné shody namísto plnotextového hledání.

Při použití pokročilých dotazů lze také specifikovat, kterých polí se daná část dotazu týká. Díky vyhledávání fráze `title:Oil` se dané slovo bude hledat výhradně v názvu výzvy a ne například v jejím obsahu.

Dotaz s obsahem `title:Innovation AND tags:("smart cities" OR "bio*")` tedy nalezne všechny výzvy, které mají v názvu slovo `Innovation` a zároveň mají ve svých značkách frázi `smart cities` nebo jakékoliv slovo začínající na `bio`.

3.3.7 Model fasety

Jednotlivé fasety jsou v portálu implementovány modelovou třídou `Facet` i přes to, že se jedná o poměrně jednoduché entity, Tento přístup jsem zvolil zejména kvůli větší modulárnosti zdrojových kódů. Metody této třídy jsou většinou statické a usnadňují práci s fasetami napříč portálem.

K vytvoření instance této třídy stačí zadat tři parametry obsahující interní jméno fasety, jméno fasety zobrazované uživateli a také odpovídající pole v dokumentech uložených v databázi Elasticsearch.

Navázání na pole v databázi je záměrně uváděno bez odpovídajícího indexu. Je předpokladem, že pole použitá pro fasetové vyhledávání jsou ve všech indexech stejně pojmenována. Tento předpoklad umožňuje zachovat zvolené hodnoty fasetového vyhledávání i při přepnutí kontextu hledání.

Při vytvoření instance se automaticky připraví název toho pole vhodný pro použití jako parametr funkce při sestrojování dotazu (tečky ve vnořených polích je nutné nahradit symboly `__`).

all()

Tato statická metoda slouží jako jednotný seznam všech faset dostupných v portálu. Volání metody způsobí vytvoření seznamu obsahující instanci třídy `Facet` pro všechny dostupné fasety.

get()

Tato třídní metoda se pokusí nalézt odpovídající instanci třídy `Facet` v seznamu z metody `all()`. Metoda je využita hlavně při vyhledávání hodnot faset.

toDict()

Jednoduchá metoda převádějící data uvedená v konstruktoru objektu do slovníku. Je určená především k předání informací ze serverové části do komponent v uživatelské části.

Kapitola 4

Experimenty a vyhodnocení

Tato kapitola popisuje experimenty prováděné za účelem zhodnocení výsledku práce. Část testování byla směřována přímo na uživatele a skládala se jak z jejich implicitní, tak z i explicitní zpětné vazby. V závěru kapitoly jsou diskutována možná budoucí vylepšení portálu.

4.1 Testování s uživateli

Do testování portálu se zapojilo 27 uživatelů. Během procházení webového portálu bylo sledováno chování všech těchto uživatelů a následně tato data sloužila pro vytvoření statistik oblastí zájmů. Z těchto uživatelů navíc 23 vyplnilo dotazník, a tím poskytli přímé sdělení svých dojmů z portálu.

4.1.1 Dotazník uživatelského prožitku

Návrh dotazníku čerpá z [35].

Dotazník, zaměřený na uživatelský prožitek (UX neboli *user experience*), se skládal z 26 otázek zaměřených na šest různých aspektů portálu.

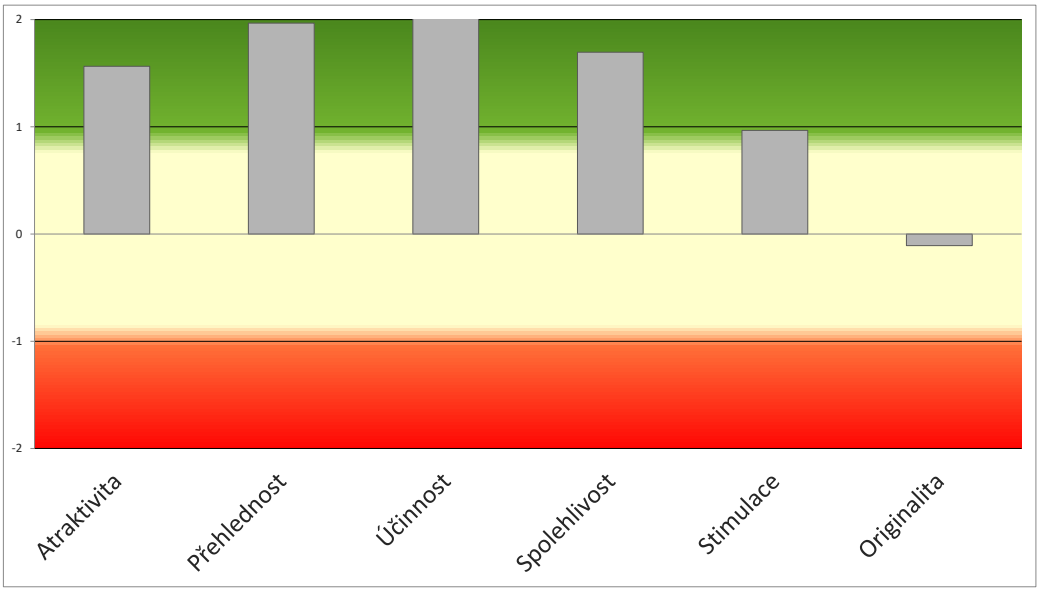
- **Atraktivita** - Celkový dojem.
- **Přesvědčivost** - Je snadné se s ním naučit pracovat?
- **Efektivita** - Lze jej používat bez zbytečného úsilí?
- **Spolehlivost** - Má uživatel pocit, že má vše pod kontrolou?
- **Stimulace** - Je jeho používání vzrušující a zábavné?
- **Novota** - Je inovativní? Dokáže zaujmout?

Celkový dojem lze rozdělit do dvou různých kvalit. Přesvědčivost, efektivita a spolehlivost jsou aspekty spadající do skupiny pragmatické kvality, která je zaměřena na technickou efektivnost a splnění cílů. Stimulace a novota jsou oproti tomu aspekty označované jako hedonická kvalita, která je spojovaná s požitkem a potěšením [4].

Na všechny otázky bylo možné odpovědět pomocí škály od 1 do 7, přičemž na každé straně škály byly postaveny vzájemně protikladné fráze zaměřené právě na jeden z uvedených aspektů. Respondent pomocí číselné škály volil, ke které frázi mají jeho pocity z portálu blíže.

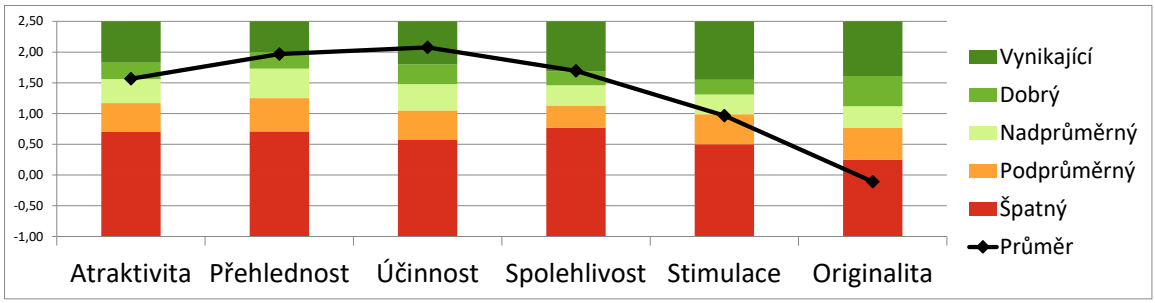
Obsah dotazníku vycházel z otázek UEQ[1] (*User Experience Questionnaire*) a respondentům byl prezentován pomocí nástroje Google Formuláře[2].

Dotazník vyplnilo 23 respondentů, nejčastěji se jednalo o muže v období rané dospělosti. Vzorek účastníku byl složen z pokročilých uživatelů internetu a pouze několik jedinců mělo předchozí zkušenost s portály podobného zaměření. Výsledky dotazníku jsou zobrazeny v grafu 4.1.



Obrázek 4.1: Graf prezentující výsledky dotazníku uživatelského prožitku portálu. Hodnoty větší než 0.8 (znázorněny zeleným pozadím) jsou považovaný za pozitivní, hodnoty mezi 0.8 a -0.8 jsou označovaný za neutrální (žluté pozadí) a hodnoty pod -0.8 (červené pozadí) značí negativní výsledek.

Pro relevantnější vyhodnocení odpovědí je k dispozici oficiální srovnání dotazníku s výsledky dalších 246 produktů s celkovým počtem 9905 respondentů.



Obrázek 4.2: Graf prezentující výsledky dotazníku uživatelského prožitku portálu ve srovnání s hodnotami dalších produktů.

[1] UEQ: https://www.ueq-online.org/

[2] Google Formuláře: https://www.google.com/intl/cs_CZ/forms/about/

Aspekt	Průměr	Výsledek	Interpretace
Atraktivita	1.57	Dobrý	10% výsledků je lepších, 75% je horších
Přehlednost	1.97	Vynikající	Patří mezi 10% nejlepších výsledků
Účinnost	2.08	Vynikající	Patří mezi 10% nejlepších výsledků
Spolehlivost	1.70	Vynikající	Patří mezi 10% nejlepších výsledků
Stimulace	0.97	Podprůměrný	50% výsledků je lepších, 25% je horších
Originalita	-0.11	Špatný	Patří mezi 25% nejhorších výsledků

Tabulka 4.1: Interpretace výsledků dotazníků uživatelského prožitku portálu ve srovnání s průměrnými hodnotami dalších produktů.

Z výsledků zobrazených v tabulce 4.1 a grafu 4.2 lze usoudit, že většina aspektů portálu je vnímána pozitivně. Za povšimnutí stojí především výborný výsledek v oblasti přehlednosti, což považuji za velký úspěch. Na druhou stranu dotazník odhalil lehce podprůměrné hodnoty v oblasti stimulace a poukázal především na slabou stránku portálu a to jeho originalitu.

V souhrnu lze tedy portál označit jako pragmaticky velmi kvalitní, s podprůměrnou hedonickou kvalitou. Vzhledem k zaměření portálu ovšem nepovažuji hedonickou kvalitu za tolik podstatnou, proto výsledek uživatelského testování vnímám pozitivně.

4.1.2 Sledování chování uživatelů

Pro získání analýzy chování uživatelů byl využit nástroj Smartlook[3]. K analýze bylo díky tomu dostupných hned několik typů tepelných map sledujících klikání uživatele, pohyb kurzoru myši nebo rolování po stránce. Navíc bylo možné sledovat nahrávky chování uživatelů a přímo tak pozorovat jejich interakci s portálem.

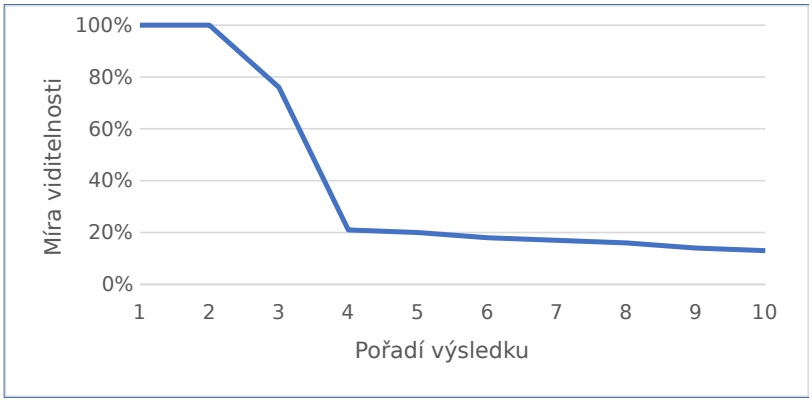
[3]Smatlook: https://www.smartlook.com/cs/

Obrázek 4.3: Tepelná mapa klikání uživatelů na hlavní stránce portálu. Některé části obsahu stránky byly nástrojem pro sledování nahrazeny symbolem „*“ z důvodu možného výskytu citlivých údajů.

Výsledky tepelné mapy z obrázku 4.3, založené na statistice klikání uživatelů na hlavní stránce, jsou ovlivněny zaznamenáním interakce nejen s vyobrazenou stránkou, ale částečně interakcí i s modálním oknem, překrývajícím tento obsah. Z toho důvodu se v tepelné mapě vyskytují i zdánlivě neopodstatněné oblasti zájmu uživatelů.

Z mapy je patrné, že fasetová navigace je mezi uživateli velmi oblíbená. Uživatelé dále jeví zájem především o první zobrazený výsledek vyhledávání, část z nich si ale podrobněji prohlédne i níže umístěné výsledky.

Dále lze z mapy vyvodit i nedostatky uživatelského prostředí. Konkrétně ve výběru typu výsledků („Projects“, „Deliverables“, „Topics“) se uživatelé často snažili kliknout i na informativní popis „Search for:“. Dalším poznatkem je nezájem uživatelů o vyhledávací pole. Napravení těchto nedostatků bude diskutováno v kapitole 4.3.



Obrázek 4.4: Graf znázorňující míru zobrazení výsledků v závislosti na jejich pořadí při zobrazení. Založeno na hodnotách z tepelné mapy B.1.

Z grafu 4.4 lze vypozorovat, že k zobrazení pátého výsledku vyhledávání se dostane průměrně pouze každý pátý uživatel. Možné řešení bude taktéž probráno v kapitole 4.3.

4.2 Rychlost obnovy dat

Prázdnou databázi je možné naplnit 3483 dostupnými tématy výzev za 17 minut. Periodická obnova dat je prováděna jednou za sedm dní.

4.3 Možnost úprav

Současné řešení práce nabízí prostor pro další vylepšení. Za zmínku stojí například implementace našeptávače vyhledávacího vstupní pole pro jednodušší sepisování pokročilejších dotazů. Našeptávač by mohl uživateli nabízet jak názvy jednotlivých polí dokumentů, tak případně i hodnoty těchto polí. Přínosné by mohlo také být zvýraznění dvojice závorek nebo ověřování správnosti vstupu v reálném čase.

Vzhledem k automatickému plnění obsahu portálu může docházet k ukládání nepřesných informací. Pokud uživatel takovou informaci nalezne, bylo by vhodné dát mu možnost tuto situaci nahlásit. Tyto hlášení by mohly být sepsány v přehledu určeném pouze pro správce portálu, který by manuálně provedl nápravu.

Dále by mohlo být užitečné přidat funkcionalitu „hlídacího psa“, kdy by uživatel měl možnost zadat obory jeho zájmu a v případě otevření nové výzvy v daném oboru, by byl uživatel o této příležitosti bezprostředně informován.

V reakci na poznatky uvedené v kapitole 4.1.2 se nabízí několik úprav uživatelského rozhraní. Především je třeba v hlavičce webu graficky odlišit nabízené odkazy pro výběr typu hledání („Projects“, „Deliverables“, „Topics“) od textového popisu „Search for:“. Dále by bylo vhodné vyvolat v uživatelích větší zájem o vyhledávací pole. Toho se lze pokusit docílit například nabízením náhodně vybraných frází, které lze hledat. Tyto fráze by sloužily pouze jako zástupný text (atribut `placeholder`) a po kliknutí do pole by byl skryt.

Další možné zlepšení by se mohlo týkat zvýšení viditelnosti níže umístěných výsledků vyhledávání. Pomoci by mohlo kompaktnější prezentování jednotlivých výsledků, alespoň co se týče výšky těchto prvků. Za pokus by stálo prezentovat na jedné stránce více výsledků. Dle průzkumu by tato úprava mohla vést ke značnému zvýšení viditelnosti výsledků, pro jejichž zobrazení je nutné stránku rolovat [8].

Kapitola 5

Závěr

Hlavním cílem této bakalářské práce bylo navrhnout a realizovat úpravu stávajícího portálu zaměřeného na výsledky evropských projektů. Portál byl úspěšně inovován s použitím novějších verzí technologií použitých v původním řešení. Mimo to byly využity i technologie úplně nové jako například Vue.js. Do obsahu portálu byly úspěšně zakomponovány témata výzev pro podávání návrhů nových projektů. Nad obsahem portálu lze provádět jak plnotextové tak i fasetové hledání. V případě potřeby je možné pomocí dotazovacího jazyka sestrojit pokročilejší dotazy. Úspěšné řešení práce bylo potvrzeno několika experimenty. Pro další rozvoj práce bylo předloženo několik návrhů.

Řešení této práce mi bylo přínosné především rozšířením obzorů v oblasti nerelačních databází, hlubším pohledem do principů plnotextového vyhledávání a také seznámením se s používáním systému Vue.js.

Výsledný portál je k dispozici online[1].

[1] European Research Projects Portal: http://athena17.fit.vutbr.cz:2021/

Literatura

[1] Anthony Gore: *Exploring Vue.js: Reactive Two-Way Data Binding*. Medium - a place to read and write big ideas and important stories, 2016-09-19, [online; navštíveno 24.04.2019].
URL https://medium.com/js-dojo/exploring-vue-js-reactive-two-way-data-binding-da533d0c4554

[2] Anthony Gore: *WTF is Vuex? A Beginner's Guide To Vue's Application Data Store*. Medium - a place to read and write big ideas and important stories, 2016-11-30, [online; navštíveno 25.04.2019].
URL https://medium.com/js-dojo/vuex-for-the-clueless-the-missing-primer-on-vues-application-data-store-33fa51ffc3af

[3] Armin Ronacher: *Welcome to Jinja2*. Jinja 2 Documentation (2.10), 2008, [online; navštíveno 24.04.2019].
URL http://jinja.pocoo.org/docs/2.10/

[4] Bernardo, M.; Marimon, F.; del Mar Alonso-Almeida, M.: Functional quality and hedonic quality: A study of the dimensions of e-service quality in online travel agencies. *Information & Management*, ročník 49, č. 7, 2012: s. 342 – 347, ISSN 0378-7206, doi:https://doi.org/10.1016/j.im.2012.06.005.
URL http://www.sciencedirect.com/science/article/pii/S0378720612000511

[5] Bhaumik, S.: *Bootstrap Essentials*. Packt Publishing Ltd., 2015, ISBN 978-1-78439-517-9.

[6] Bootstrap team: *Modal*. Bootstrap - The most popular HTML, CSS and JS library in the world., 2019, [online; navštíveno 04.05.2019].
URL https://getbootstrap.com/docs/4.3/components/modal/

[7] Carevic, Z.; Schüller, S.; Mayr, P.; aj.: Contextualised Browsing in a Digital Library's Living Lab. *CoRR*, ročník abs/1804.06426, 2018, 1804.06426.
URL http://arxiv.org/abs/1804.06426

[8] Clicktale: *Clicktale scrolling research report V2.0 – Part 1: visibility and scroll reach*. Clicktale blog, 2007-10-05, [online; navštíveno 12.05.2019].
URL https://www.clicktale.com/resources/blog/clicktale-scrolling-research-report-v20-part-1-visibility-and-scroll-reach/

[9] Code Conquest: *Introduction to Web Development*. 2019, [online; navštíveno 27.04.2019].
URL https://www.codeconquest.com/what-is-coding/web-programming/

[10] Elasticsearch B.V.: *Basic Concepts*. Elasticsearch Reference [7.0], 2019, [online; navštíveno 29.04.2019].
URL https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started-concepts.html

[11] Elasticsearch B.V.: *Bucket Aggreagtions*. Elasticsearch Reference [7.0], 2019, [online; navštíveno 21.04.2019].
URL https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket.html

[12] Elasticsearch B.V.: *Elasticsearch Clients*. Elasticsearch - Open Source Search & Analytics, 2019, [online; navštíveno 02.05.2019].
URL https://www.elastic.co/guide/en/elasticsearch/client/index.html

[13] Elasticsearch B.V.: *Query and filter context*. Elasticsearch - Open Source Search & Analytics, 2019, [online; navštíveno 07.05.2019].
URL https://www.elastic.co/guide/en/elasticsearch/reference/current/query-filter-context.html

[14] European Union: *About*. European Commission, 2019, [online; navštíveno 27.05.2019].
URL https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/home

[15] European Union: *About CORDIS*. 2019, [online; navštíveno 27.04.2019].
URL https://cordis.europa.eu/about/

[16] European Union: *Find a funding opportunity*. European Commission, 2019, [online; navštíveno 05.05.2019].
URL https://ec.europa.eu/info/funding-tenders/how-eu-funding-works/how-get-funding/find-funding-opportunity_en

[17] Evan You: *The Progressive Framework*. Slides - Create and share presenatations online, 2016, [online; navštíveno 23.04.2019].
URL http://slides.com/evanyou/progressive-javascript#/19

[18] Evan You: *Comparison with Other Frameworks*. Vue.js, 2019, [online; navštíveno 24.04.2019].
URL https://vuejs.org/v2/guide/comparison.html#Learning-Curve

[19] Evan You: *What is Vuex?* Vuex, 2019, [online; navštíveno 24.04.2019].
URL https://vuex.vuejs.org

[20] Frain, B.: *Responsive Web Design with HTML5 and CSS3*. Packt Publishing Ltd., 2012, ISBN 978-1-84969-318-9.

[21] Gormley, C.; Tong, Z.: *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine*. O'Reilly Media, Inc., 2015, ISBN 978-1-449-35854-9.

[22] Honza Král: *Elasticsearch DSL*. Elasticsearch DSL 7.0.0 documentation, 2019, [online; navštíveno 03.05.2019].
URL https://elasticsearch-dsl.readthedocs.io/en/latest/

[23] Kathryn Whitenton: *Filters vs. Facets: Definitions*. Nielsen Norman Group: UX Training, Consulting, & Research, 2014-03-16, [online; navštíveno 21.04.2019].
URL https://www.nngroup.com/articles/filters-vs-facets/

[24] Kushal Das: *Introduction to Flask*. Python for you and me, 2017, [online; navštíveno 23.04.2019].
URL https://pymbook.readthedocs.io/en/latest/flask.html

[25] Liferay Inc.: *What is a Web Portal?* 2019, [online; navštíveno 20.04.2019].
URL https://www.liferay.com/resources/l/web-portal

[26] Luděk Veselý: *Seriál Elasticsearch: 4. Fulltextové vyhledávání v češtině*. 2017-09-28, [online; navštíveno 02.05.2019].
URL https://www.ludekvesely.cz/serial-elasticsearch-4-fulltextove-vyhledavani-v-cestine/

[27] Mozilla Foundation: *ElasticUtils*. ElasticUtils dev documentation, 2015, [online; navštíveno 03.05.2019].
URL https://elasticutils.readthedocs.io/en/latest/

[28] Pallets Team: *Design Decisions in Flask*. Flask 1.0.2 Documentation, 2010, [online; navštíveno 23.04.2019].
URL http://flask.pocoo.org/docs/1.0/design/

[29] Pallets Team: *Foreword*. Flask 1.0.2 Documentation, 2010, [online; navštíveno 24.04.2019].
URL http://flask.pocoo.org/docs/1.0/foreword/

[30] Phillip J. Eby: *PEP 333 – Python Web Server Gateway Interface v1.0*. Python.org, 2003-12-7, [online; navštíveno 24.04.2019].
URL https://www.python.org/dev/peps/pep-0333/#rationale-and-goals

[31] Ratcliffe, S.: *ASP.NET Core 2 and Vue.js: Full Stack Web Development with Vue, Vuex, and ASP.NET Core 2.0*. Packt Publishing Ltd., 2018, ISBN 978-1-78883-946-4.

[32] Roy Williams: *Python 3 at Lyft*. Youtube, 2018-01-10, [online; navštíveno 23.04.2019].
URL https://www.youtube.com/watch?v=aetoXWRt24k&t=164

[33] Sanner, M.: Python: A Programming Language for Software Integration and Development. 11 1998, doi:10.1016/S1093-3263(99)99999-0.

[34] Sarker, I.; Apu, K.: MVC Architecture Driven Design and Implementation of Java Framework for Developing Desktop Application. *International Journal of Information Technology*, ročník 7, 09 2014: s. 317–322, doi:10.14257/ijhit.2014.7.5.29.

[35] Schrepp, M.: User Experience Questionnaire Handbook. 08 2019.

[36] Staněk, P.: *Automaticky aktualizovaný webový portál*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta informačních technologií, 2016.
URL http://www.fit.vutbr.cz/study/DP/BP.php?id=18568

[37] Steve Cohen: *What challenges has Pinterest encountered with Flask?* Quora - A place to share knowledge and better understand the world, 2015-01-08, [příspěvek v diskuzním fóru; navštíveno 23.04.2019].
URL https://www.quora.com/What-challenges-has-Pinterest-encountered-with-Flask/answer/Steve-Cohen?share=1&srid=hXZd

[38] Zell Liew: *How To Write Mobile-first CSS*. 2014-12-17, [online; navštíveno 03.05.2019].
URL https://zellwk.com/blog/how-to-write-mobile-first-css/

Příloha A

Obsah přiloženého CD

Struktura přiloženého CD

- **README.txt** - Textový soubor popisující obsah CD
- **src/** - Adresář obsahující zdrojové kódy
- **thesis/** - Adresář obsahující písemnou zprávu a její zdrojové kódy
- **misc/** - Adresář obsahující plakát a výsledky uživatelského testování

Příloha B

Přiložené obrázky

Obrázek B.1: Tepelná mapa rolování uživatelů na stránce prezentující výsledky vyhledávání.