ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Krejčí** Jméno: **Martin** Osobní číslo: **507626**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

Specializace: **Business informatics**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Průvodce výběrem studia informatiky na vysokých školách

Název bakalářské práce anglicky:

A guide to choosing a computer science degree programme at a university

Pokyny pro vypracování:

Navrhněte a vytvořte aplikaci, která zájemcům o studium informatických oborů pomůže s výběrem vhodné vysoké školy. Postupujte následovně:
1) Definujte základní pojmy – vysoká škola, univerzita, technický obor, informatický obor atd.
2) Analyzujte existující nabídku studijních oborů, zaměřených na studium informatiky v České republice.
3) Proveďte průzkum existujících podpůrných materiálů, sloužících jako pomůcka s výběrem studia a vyhodnoťte je.
4) Navrhněte vlastního průvodce výběrem vhodného studijního oboru a školy, inspirujte se například volebními kalkulačkami atd.
5) Navrženého průvodce implementujte. Očekávaným výstupem průvodce bude nejen doporučená vysoká škola, ale také argumenty, proč byla tato škola vybrána a jaká kritéria výběru byla zásadní.
6) Vytvořenou aplikaci uživatelsky otestujte a na vybrané sílové skupině vyhodnoťte její nejen použitelnost, ale především využitelnost.

Seznam doporučené literatury:

ECKEL, Bruce. Thinking in Java. Fourth edition. Upper Saddle River, NJ: Prentice Hall, c[2006]. ISBN 0131872486.
ŽÁRA, Ondřej. JavaScript: programátorské techniky a webové technologie. Brno: Computer Press, 2015. ISBN 978-80-251-4573-9.
BASSOT, Barbara. University Choice Journal: Decision making guide to help students choose which university and degree. January 9, 2018. Trotman, 2018. ISBN 1911067648.
GAMMA, Erich. Design patterns: elements of reusable object-oriented software. Boston: Addison-Wesley, 1995. ISBN 0201633612.
DOLEŽAL, Jan; MÁCHAL, Pavel a LACKO, Branislav. Projektový management podle IPMA. 2., aktualiz. a dopl. vyd. Expert (Grada). Praha: Grada, 2012. ISBN 9788024742755.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Pavel Náplava, Ph.D. Centrum znalostního managementu FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **15.02.2024** Termín odevzdání bakalářské práce: **24.05.2024**

Platnost zadání bakalářské práce: **21.09.2025**

Ing. Pavel Náplava, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ _____

Datum převzetí zadání Podpis studenta

ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

FAKULTA ELEKTROTECHNICKÁ

BAKALÁŘSKÁ PRÁCE

2024

MARTIN
KREJČÍ

Bakalářská práce Martin Krejčí 2024

České vysoké učení technické v Praze

Fakulta elektrotechnická

Centrum znalostního managementu

Bakalářská práce

Podpora výběru studijního programu

Autor: Martin Krejčí

Vedoucí práce: Ing. Pavel Náplava, Ph.D

Studijní program: Softwarové inženýrství a technologie – Business Informatics

Praha 2024

Deklarace

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 24.5.2024

……

Martin Krejčí

Poděkování

Rád bych využil této příležitosti a poděkoval Ing. Pavlu Náplavovi, Ph.D., za pravidelné a velice přínosné konzultace plné vědomostního obohacení a plodného dialogu. Bez pravidelných konzultací pod jeho vedením by zhotovit tuto bakalářskou práci bylo obtížnější.

Abstrakt

Tato bakalářská práce se zaměřuje na vytvoření aplikace, která pomáhá středoškolákům při výběru vhodného informatického oboru na vysoké škole. Cílem je usnadnit rozhodovací proces a poskytnout přehledné a užitečné informace o studijních programech dostupných v České republice.

Metodologicky se práce skládá z několika klíčových kroků. Nejprve byly definovány základní pojmy, jako jsou vysoká škola, univerzita a informatický obor. Následně byla provedena analýza existujících studijních oborů a podpůrných nástrojů. Na základě této analýzy byl navržen a implementován průvodce, který využívá uživatelský dotazník k poskytování personalizovaných doporučení studijních oborů.

Výsledkem je webová aplikace s backendem v Java SpringBoot a frontendem v Angular, která umožňuje uživatelům vyplnit dotazník a obdržet doporučení vhodných studijních oborů a vysokých škol. Aplikace byla testována na cílové skupině uživatelů, kteří poskytli zpětnou vazbu ohledně její použitelnosti a užitečnosti.

V závěru práce bylo potvrzeno, že vytvořená aplikace efektivně podporuje středoškoláky při výběru studijního oboru v oblasti informatiky. Aplikace zjednodušuje přístup k relevantním informacím a poskytuje konkrétní doporučení, což může výrazně ovlivnit jejich rozhodování o dalším studiu.

Klíčová slova informatika, vysoká škola, průvodce studijními programy, rozhodovací proces

Abstract

This bachelor's thesis focuses on creating an application that helps high school students choose a suitable computer science degree program at a university. The aim is to simplify the decision-making process and provide clear and useful information about the available study programs in the Czech Republic.

Methodologically, the work consists of several key steps. Firstly definition of the basic terms such as university, study program, and IT study program were defined. Subsequently, an analysis of existing study programs and supporting tools was conducted. Based on this analysis, a guide was designed and implemented using a user questionnaire to provide personalized recommendations for study programs.

The result is a web application with a backend in Java SpringBoot and a frontend in Angular, allowing users to fill out a questionnaire and receive recommendations for suitable study programs and universities. The application was tested on a target group of users, who provided feedback on its usability and usefulness.

In conclusion, the thesis confirmed that the created application effectively supports high school students in choosing a computer science degree program. The application simplifies access to relevant information and provides specific recommendations, which can significantly influence their decision-making regarding further studies.

Keywords: computer science, university, study program guide

Seznam použitých zkratek

IT	Informační technologie
MŠMT	Ministerstvo školství a tělovýchovy
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
GUI	Graphical User Interface
ACID	Atomicity, Consistency, Isolation, Durability
NFR	Non-Functional Requirement
UML	Unified Modeling Language
CRUD	Create, Read, Update, Delete

Seznam obrázků

Obsah

1. Úvod

1.1. Úvod do bakalářské práce

Tato bakalářská práce se zaměřuje na výběr IT oborů bakalářských studijních programů pro studenty středních škol a jiné zájemce o studium IT na vysokoškolské úrovni v České republice. V současné době se na trhu vysokých škol pohybuje velké množství institucí, státních i soukromých, které nabízejí vysokoškolské vzdělání v oboru IT. S příbytkem možností pro potenciální studenty informačních technologií je složité se zorientovat na trhu vysokých škol a najít optimální volbu pro studium.

Studijních IT oborů je na území České republiky velké množství. Rozdíly mezi jednotlivými obory je těžké identifikovat bez předchozí zkušenosti studia, nebo doporučení od bývalých studentů daných institucí. Výběr správné studijní instituce je složitý proces, který se skládá z mnoha proměnných. Informační technologie jako disciplína jsou velice široký pojem, a proto se mezi vysokými školami, ba i fakultami v rámci jedné vysoké školy, přístup k výuce IT výrazně liší. Tato skutečnost rozhodovací proces velmi komplikuje a potenciální studenti jsou postaveni před složitý problém, ke kterému v současné době není zveřejněno dostatek podpůrných materiálů.

Z vlastních zkušeností a zkušeností mých vrstevníků vím, že volba správného oboru a školy může být velmi stresující a náročná. Na oficiálních stránkách univerzit a stáncích na veletrzích je zájemce o studium obklopen superlativy popisující studium na dané instituci. Proto je pro potenciálního zájemce velice složité učinit správnou volbu pro jeho konkrétní situaci. Mnoho mých spolužáků, kteří již tuto volbu podstoupili, se potýkalo s nedostatkem jasných a konkrétních informací, což vedlo k nejistotě a obavám ohledně budoucnosti. V rámce této bakalářské práce se budeme zabývat celým kontextem této rozhodovací situace a pokusíme se budoucím studentům jejich rozhodovací proces ulehčit.

1.2. Cíle práce

Cílem této bakalářské práce je zpracovat důkladnou analýzu již existujících podpůrných nástrojů, které mají zjednodušit rozhodování potenciálních studentů při výběru studijního oboru. Práce se zaměřuje na dialog s vrstevníky a budoucími studenty vysokých škol, kteří tuto rozhodovací situaci již podstoupili, nebo je teprve čeká. Na základě tohoto dialogu bude proveden

sběr kritérií, jež jsou klíčová pro cílové uživatele. Tato kritéria budou obsahovat otázky, na kterých staví dotazníková část naší aplikace.

Podrobně získaná kritéria budou definována a na jejich základě bude provedena filtraci studijních oborů. V neposlední řadě bude vyvinut podpůrný nástroj, který bude tuto rozhodovací situaci uživatelům usnadňovat. Tento projekt se zaměřuje výhradně na bakalářské studijní IT programy, které jsou nabízeny na veřejných vysokých školách.

Hlavním výstupem této bakalářské práce je vyvinout průvodce výběrem studijního oboru. Motivací k tomuto projektu je skutečnost, že ačkoli existuje mnoho průvodců a volebních kalkulaček pro výběr studijních oborů, žádné z nich nepokrývají všechny aspekty, které jsou potřeba. Tento průvodce se zaměří na poskytování komplexních informací, které budou brát v úvahu širokou škálu kritérií důležitých pro cílové uživatele. Zároveň se budeme inspirovat pozitivními vlastnostmi již existujících nástrojů, a naopak se budeme snažit vyvarovat chybám a nedostatkům, které svazují již existující podpůrné nástroje.

1.3. Struktura a rozdělení práce

Tato bakalářská práce je členěna do několika hlavních částí, které na sebe logicky navazují a tvoří ucelený přehled o celém tématu a rozsáhlosti problematiky.

První část, rešeršní část, se zaměřuje na vytvoření teoretického základu pro celou práci. Nejprve jsou definovány klíčové pojmy a koncepty, které jsou v práci používány. To zahrnuje vysvětlení pojmů jako veřejná vysoká škola, fakulta, studijní program, bakalářský studijní program a IT studijní obor. Následuje představení rozhodovací situace, ve které se nacházejí středoškoláci při výběru svého budoucího studijního oboru, a popis konkrétních scénářů, které mohou nastat. Dále je provedena analýza existujících podpůrných nástrojů, které mají usnadnit výběr studijního oboru, včetně jejich silných a slabých stránek. Rešeršní část je zakončena popisem metodiky výběru studijních oborů pro tento projekt, která zahrnuje konkrétní výčet studijních oborů a vysokých škol, které budeme v rámci práce hlouběji analyzovat a hodnotit.

Druhá část, analytická část, se zaměřuje na praktické kroky vedoucí ke sběru a analýze dat. Začíná úvodním kolem rozhovorů s budoucími, současnými a úspěšnými studenty vysokoškolských oborů, které poskytují cenné informace o jejich zkušenostech a kritériích pro výběr studijního oboru. Následuje sběr dat, která jsou následně transformována na konkrétní kritéria, jež tvoří základ pro otázky v dotazníkové části aplikace. Na základě těchto kritérií jsou

hodnoceny jednotlivé studijní obory. Tato část je završena prvotním návrhem aplikace, který popisuje vývoj a funkcionalitu navržené aplikace.

Třetí část, implementační část, se zabývá praktickou realizací návrhu aplikace. Nejprve je popsán výběr vhodných technologií pro implementaci aplikace. Následně je detailně popsána implementace backendu, frontendu a databáze. Následuje testování výsledné aplikace, které zahrnuje jak vývojářské, tak uživatelské testování. Tato část je zakončena druhým kolem rozhovorů, kde se diskutuje správnost zhotovení aplikace a její přínos pro uživatele.

Poslední částí je závěr práce, kde jsou diskutovány hlavní výsledky a přínosy aplikace. Tato část shrnuje klíčové výstupy práce a nabízí návrhy na potenciální vylepšení a rozšíření aplikace v budoucnu.

2. Definice klíčových pojmů

2.1. Úvod

Než začneme s metodikou výběrem konkrétních studijních institucí a analýzou již existujících podpůrných nástrojů pro volbu studijního programu, je potřeba nejprve objasnit a definovat klíčové pojmy, které jsou často používané a referované ve stati zbytku bakalářské práce. Při definici většiny pojmů v této bakalářské práci se budeme řídit ustanoveními zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů, vydávané Ministerstvem školství, mládeže a tělovýchovy [3].

2.1.1. Veřejná vysoká škola

Prvním pojmem, který je nutné definovat, je pojem veřejná vysoká škola. V rámci této bakalářské práce jsou veřejné vysoké školy primárním subjektem, který budeme brát v úvahu. Dle MŠMT je pro naše účely veřejná vysoká škola definována následujícím způsobem:

Zákon č. 111/1998 Sb., o vysokých školách:

§ 2

(1) Vysoká škola uskutečňuje akreditované studijní programy a programy celoživotního vzdělávání. Typ vysokoškolské vzdělávací činnosti je určen typem uskutečňovaných akreditovaných studijních programů. Typy studijních programů jsou bakalářský, magisterský a doktorský.

(3) Vysoká škola je univerzitní nebo neuniverzitní.

(4) Vysoká škola univerzitní uskutečňuje magisterské nebo doktorské studijní programy a v souvislosti s tím vědeckou a výzkumnou, vývojovou, uměleckou nebo další tvůrčí činnost. Může uskutečňovat též bakalářské studijní programy.

(5) Vysoká škola neuniverzitní uskutečňuje převážně bakalářské studijní programy a v souvislosti s tím výzkumnou, vývojovou, uměleckou nebo další tvůrčí činnost. Vysoká škola neuniverzitní se nečlení na fakulty.

(7) Vysoká škola je veřejná, soukromá nebo státní. Státní vysoká škola je vojenská nebo policejní.

2.1.2. Členění veřejné vysoké školy

Jelikož jsme již obeznámeni se sémantickým významem vysoké školy je také potřeba zmínit, jakým způsobem se veřejná vysoká škola člení na dílčí části. Opět budeme vycházet z definice MŠMT, které členění veřejné vysoké školy definuje následovně.

Zákon č. 111/1998 Sb., o vysokých školách:

§ 22

(1) Veřejná vysoká škola se může členit na tyto součásti:

a) fakulty,

b) vysokoškolské ústavy,

c) jiná pracoviště pro vzdělávací a vědeckou, výzkumnou, vývojovou, uměleckou nebo další tvůrčí činnost nebo pro poskytování informačních služeb,

d) účelová zařízení pro kulturní a sportovní činnost, pro ubytování a stravování zejména členů akademické obce nebo k zajišťování provozu školy.

(2) Vnitřní předpisy součástí musí být v souladu s vnitřními předpisy veřejné vysoké školy.

2.1.3. Fakulta

V této podkapitole budeme definovat jednu z nejdůležitějších dílčích částí veřejných vysokých škol, kterou je fakulta.

Zákon č. 111/1998 Sb., o vysokých školách:

§ 23

(1) Fakulta uskutečňuje nejméně jeden akreditovaný studijní program a vykonává vědeckou, výzkumnou, vývojovou, uměleckou nebo další tvůrčí činnost.

(2) Na fakultě se ustavuje samosprávný zastupitelský akademický orgán. Fakulta má právo používat vlastní akademické insignie a konat akademické obřady.

(3) O zřízení, sloučení, splynutí, rozdělení nebo zrušení fakulty rozhoduje na návrh rektora akademický senát veřejné vysoké školy. Toto rozhodnutí je podmíněno souhlasným stanoviskem Akreditační komise.

2.1.4. Studijní obor

Důležitým pojmem pro tuto bakalářskou práci je pojem studijní obor. Jedná se totiž o pojem, který je v této bakalářské práci nejčastěji skloňován a je nejdůležitějším subjektem pro pochopení zbytku bakalářské práce.

Zákon č. 111/1998 Sb., o vysokých školách:

§ 44

(1) Vysokoškolské vzdělání se získává studiem v rámci akreditovaného studijního programu podle studijního plánu stanovenou formou studia.

(2) Součástmi studijního programu jsou

a) název studijního programu, jeho typ, forma a cíle studia,

b) členění studijního programu na studijní obory, jejich charakteristika a jejich kombinace, jakož i stanovení profilu absolventa příslušných studijních oborů,

c) charakteristika studijních předmětů,

d) pravidla a podmínky pro vytváření studijních plánů, popřípadě délka praxe,

e) standardní doba studia při průměrné studijní zátěži vyjádřená v akademických rocích,

f) podmínky, které musí student splnit v průběhu studia ve studijním programu a při jeho řádném ukončení podle § 45 odst. 3, § 46 odst. 3 a § 47 odst. 4 včetně obsahu státních zkoušek,

g) udělovaný akademický titul,

h) návaznost na další typy studijních programů v téže nebo příbuzné oblasti studia.

(3) Studijní plán stanoví časovou a obsahovou posloupnost studijních předmětů, formu jejich studia a způsob ověření studijních výsledků.

(4) Forma studia vyjadřuje, zda jde o studium prezenční, distanční nebo o jejich kombinaci.

2.1.5. Bakalářský studijní program

Pojem bakalářský studijní program je přidružený k pojmu studijní program. Avšak v těchto pojmech jsou klíčové rozdíly. MŠMT bakalářský studijní program definuje následovně:

Zákon č. 111/1998 Sb., o vysokých školách:

§ 45

(1) Bakalářský studijní program je zaměřen zejména na přípravu k výkonu povolání, při nichž se bezprostředně využívají soudobé poznatky a metody; obsahuje též vybrané teoretické poznatky.

(2) Standardní doba studia včetně praxe je nejméně tři a nejvýše čtyři roky.

(3) Studium se řádně ukončuje státní závěrečnou zkouškou, jejíž součástí je zpravidla obhajoba bakalářské práce.

(4) Absolventům studia v bakalářských studijních programech se uděluje akademický titul „bakalář“ (ve zkratce „Bc.“ uváděné před jménem)

2.1.6. IT Studijní obor

V České republice existuje mnoho bakalářských studijních programů v oblasti IT napříč různými vysokými školami. Jejich definice se může lišit podle univerzit a konkrétních programů. Pro účel této bakalářské práce jsem, na základě mých zkušeností v oboru a konzultaci s vedoucím bakalářské práce doktorem Náplavou, zformuloval následující definici studijního IT oboru:

Studijní obor v oblasti Informačních technologií (IT) v České republice je akademický program zaměřený na vývoj a zdokonalování znalostí a dovedností studentů v oblasti informačních systémů, softwarového inženýrství, správy sítí, databázových technologií a dalších aspektů IT. Cílem tohoto programu je připravit studenty na komplexní a efektivní práci v oblasti informačních technologií a jejich aplikací v různých odvětvích.

Tento obor obvykle zahrnuje kurzy týkající se programování, algoritmizace, počítačových sítí, databázových systémů, bezpečnosti informací, softwarového inženýrství a dalších relevantních témat. Studenti mohou mít také možnost se specializovat v konkrétních odvětvích IT podle svých zájmů a kariérních cílů.

2.2. Shrnutí

V této kapitole jsme se seznámili a definovali všechny klíčové pojmy, které jsou nutné pro pokračování v této bakalářské práci. Získali jsme základní porozumění pojmům jako veřejná vysoká škola, fakulta, studijní program, bakalářský studijní program a IT studijní obor. Tyto definice jsou klíčové pro správné pochopení dalšího textu a pro aplikaci metod a přístupů, které budou v této práci použity.

V následující kapitole se budeme zabývat prezentací problematiky, se kterou se setkávají středoškoláci při výběru správného IT programu. Pro lepší ilustraci problému představíme dva scénáře – online vyhledávání studijních oborů a návštěvu veletrhu vysokých škol (Gaudeamus). Tyto scénáře nám pomohou identifikovat konkrétní výzvy a problémy, kterým čelí potenciální studenti.

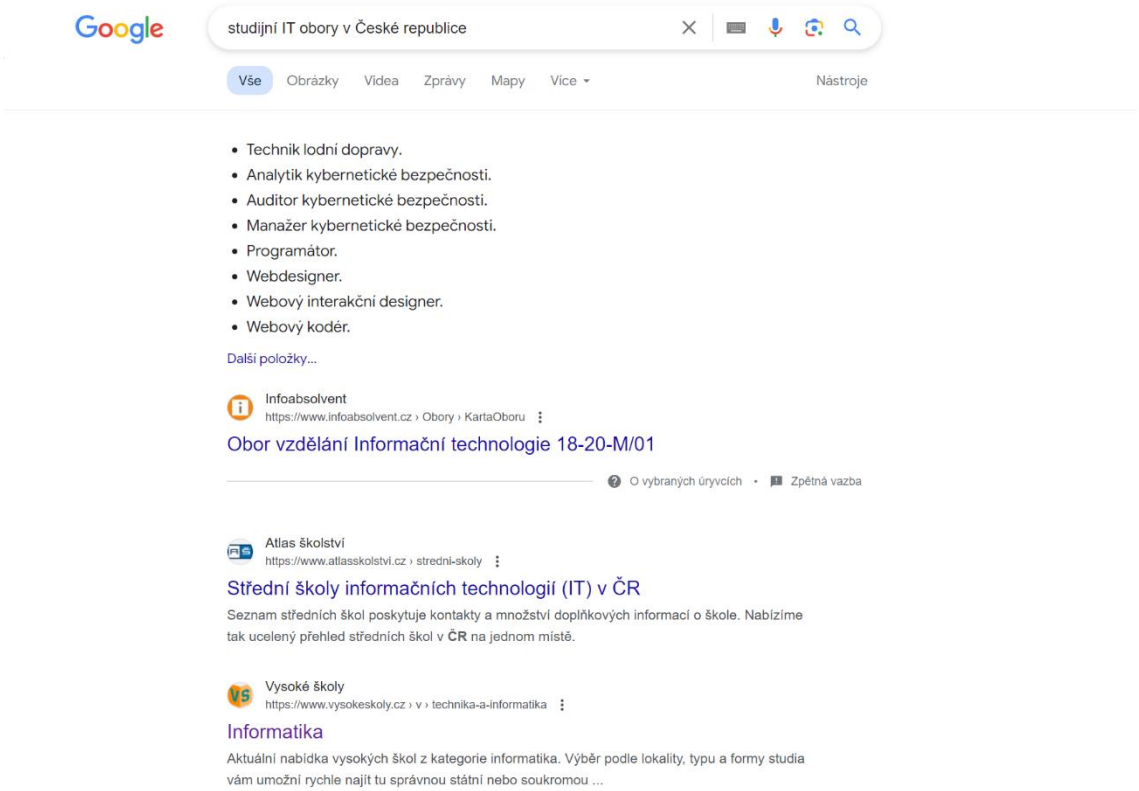
3. Prezentace rozhodovací situace

Tato kapitola slouží jako úvod do problematiky výběru správného IT programu pro středoškoláky, což je klíčovou součástí této práce. Pro lepší ilustraci rozhodovací situace je dobré si všechny subjekty konkrétně definovat, aby byly všechny aspekty rozhodovací situace jasné. Abychom lépe porozuměli těmto výzvám, představíme dva scénáře, ve kterých se středoškoláci mohou ocitnout. Těmito scénáři je vyhledávání studijních oborů online a osobní návštěva veletrhu vysokých škol - Gaudeamu.

3.1. Scénář online

3.1.1. Popis scénáře online

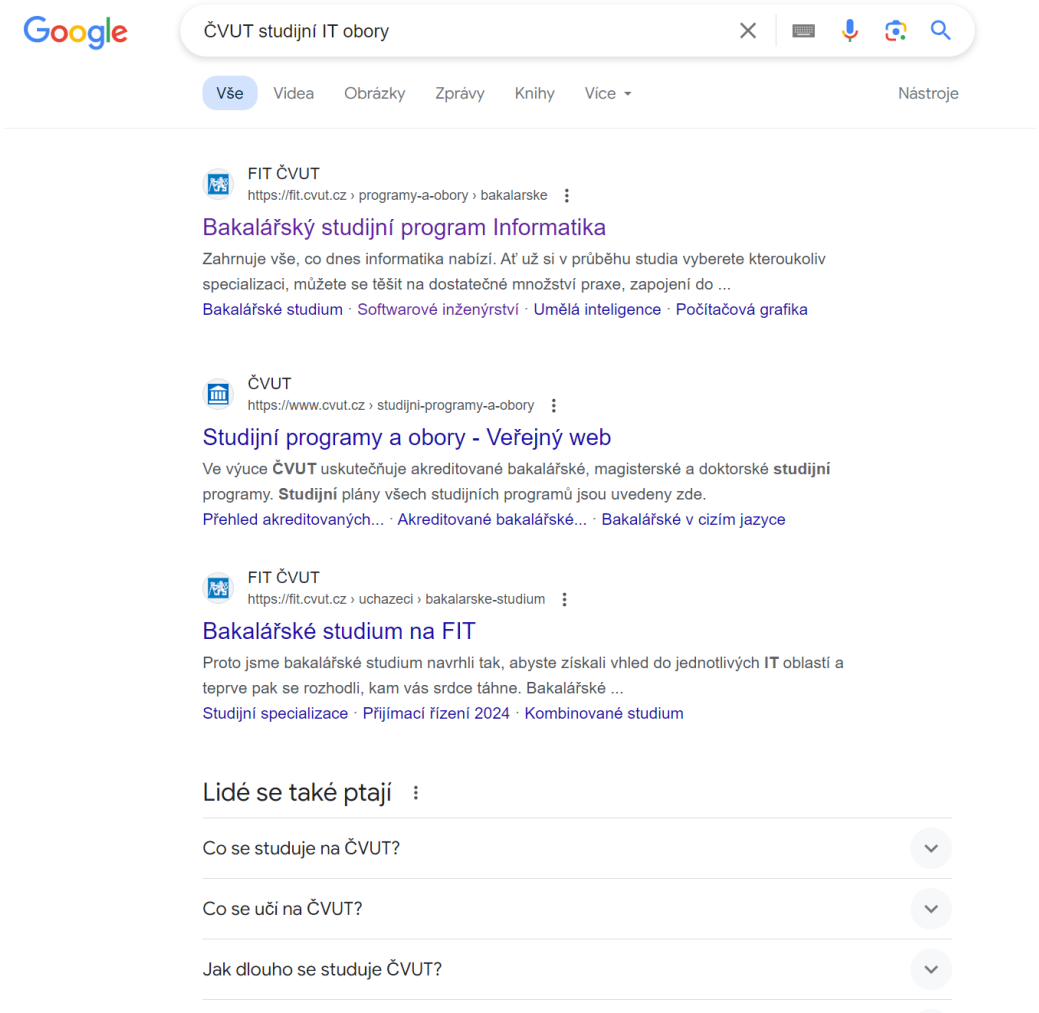
Pojďme se podívat na první ze dvou vzorových scénářů a pokusit se identifikovat problémy s výběrem správného IT programu. Představme si následující situaci: Jsem středoškolák, který chce najít studijní IT obor, který mi bude vyhovovat. Nejprve vyhledám na Google studijní IT obory. Poté se objeví velké množství výsledků a potenciálních voleb (viz. Obrázek X).



Obrázek 1: Výsledek vyhledávání

Tento počet vysokých škol, oborů a návrhů je zahlcující. Z této velké množiny možností je velice složité vybrat právě ten obor, který by mi vyhovoval. Výsledky nejsou nijak seřazeny, jednotlivé návrhy se od sebe diametrálně liší, kde některé vůbec nesouvisí s IT (technik lodní dopravy viz. Obrázek X). Mimo jiné zde nejsou uvedeny žádné konkrétní fakulty, studijní obory ani vysoké školy a volba je tedy stále velice složitá.

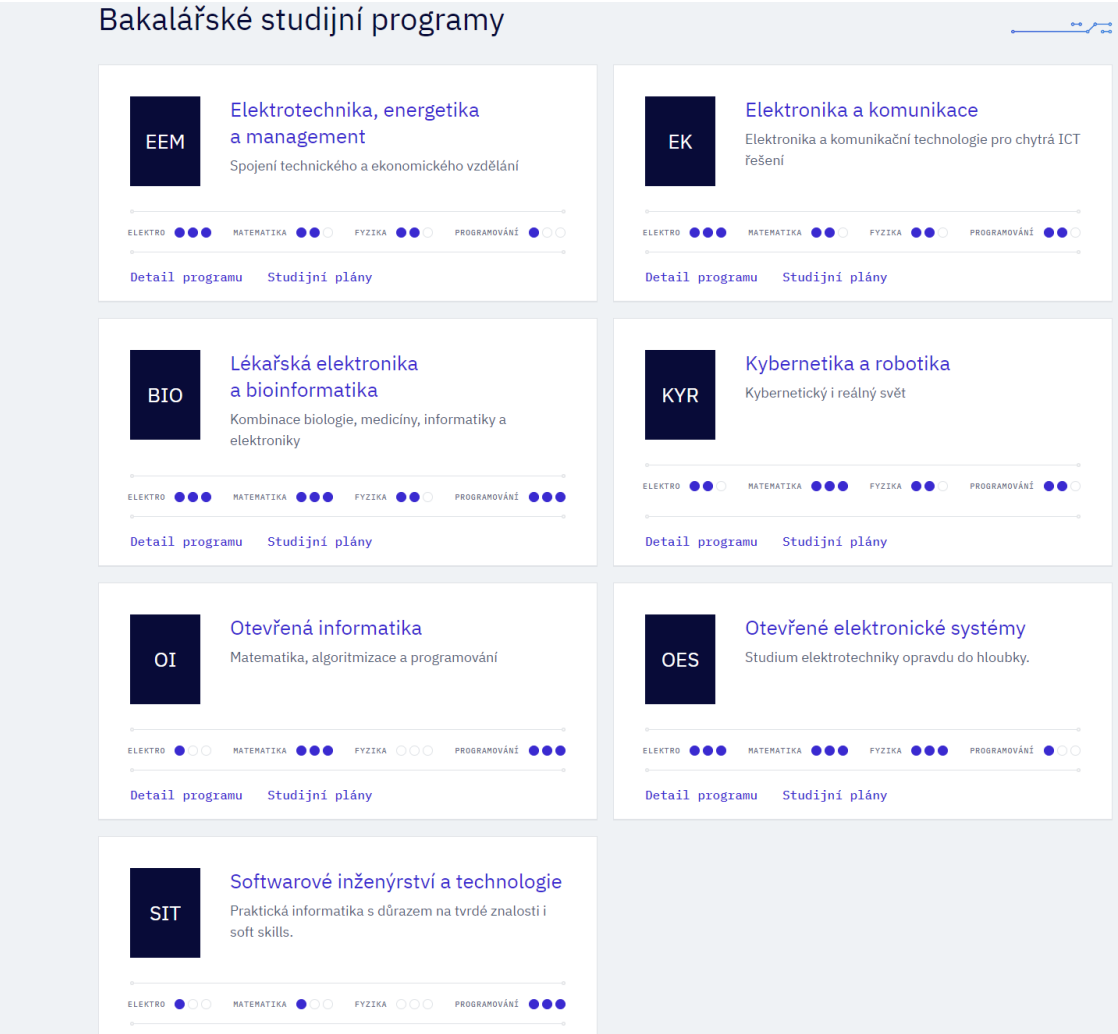
Zkusíme tedy další vyhledávací příkaz. Tentokrát zmíníme konkrétní vysokou školu a opět se budeme tázat na studijní IT obory.



Obrázek 2: Výsledek vyhledávání 2

Množství doporučených institucí i v rámci jedné vysoké školy je velice velké (viz. Obrázek 2). I když se nám povedlo se dostat ke konkrétním institucím, tak je rozhodovací proces stále daleko od úspěšného konce. Prokliknu se tedy na první stránky univerzit a začnu prohlížet studijní

obory v rámci dané fakulty. Zjistím, že jich je mnoho a jsem opět obklopen velkým množstvím možností (viz. Obrázek 3).



Obrázek 3: Příklad oborů na fakultě

Výše můžete vidět příklad jedné z mnoha fakult, která vyučuje IT studijní obory. I v rámci jedné fakulty je studijních oborů velké množství a nezbývá nic jiného, než každý obor pracně prostudovat a pokusit se vybrat ten správný. Avšak takovýchto vysokých škol a fakult je na území České republiky velké množství. Každá univerzita nabízí různé programy a informace jsou často velmi obecné. Jednotlivé vysoké školy a fakulty se dále liší nejen programy, ale i lokalitami, což ještě více komplikuje mé rozhodování.

3.1.2. Shrnutí získaných informací ze scénáře online

V této podkapitole jsme se zaměřili na problémy, kterým čelí středoškoláci při online vyhledávání studijních IT oborů. Zjistili jsme, že množství dostupných informací je ohromující a často nepřehledné. Výsledky vyhledávání nejsou seřazeny ani filtrovatelné podle relevantních

kritérií, což ztěžuje výběr vhodného studijního programu. Navíc, informace na webových stránkách univerzit jsou často velmi obecné a neposkytují dostatečné detaily, které by usnadnily rozhodování. Stránky univerzit jsou plné superlativ o studiu a studijním životě na instituci, což nemusí odrážet reálný stav. Tyto problémy vedou k frustraci a nejistotě, což podtrhuje potřebu efektivnějších nástrojů pro výběr studijního oboru.

V následující podkapitole se podíváme na další scénář, kde středoškolák osobně navštěvuje veletrh Gaudeamus a snaží se najít vhodný IT studijní obor přímo na místě. Tento přístup nám poskytne další pohled na výzvy a překážky spojené s výběrem správného studijního programu.

3.2. Scénář Gaudeamus

3.2.1. Popis scénáře Gaudeamus

Pojďme se podívat na druhý vzorový scénář a identifikovat problémy s výběrem správného IT programu při návštěvě veletrhu vysokých škol Gaudeamus. Představme si následující situaci: Jsem středoškolák hledající ideální obor. Rozhodl jsem se navštívit veletrh Gaudeamus, nebo akci mu podobnou, která se koná v rozsáhlé výstavní hale plné stánků různých univerzit.

Jakmile vstoupím do haly, jsem obklopen velkým množstvím vysokých škol a studijních oborů. Všude kolem mě jsou stánky univerzit, které lákají potenciální studenty svými nabídkami a sliby. Každá univerzita tvrdí, že jejich IT programy jsou nejlepší, že mají nejmodernější laboratoře, vysoké platy po úspěšném dokončení studia a vysokou šanci najít práci.

Přítomní studenti vysokých škol mluví velice obecně. Nemám připravené žádné specifické otázky, což činí rozhovory s prezentátory méně efektivními. Každý prezentátor se snaží přesvědčit, že právě jejich univerzita je ta nejlepší volba. Tato obecná tvrzení ale nepomáhají v rozhodovacím procesu, protože informace jsou často povrchní a zaměřené spíše na propagaci než na poskytnutí skutečných detailů.

Procházím jednotlivými stánky a zjišťuji, že studijní obory v rámci jednotlivých škol splývají. Rozdíly mezi nimi se zdají minimální, což komplikuje výběr. Prezentátoři se mě snaží přesvědčit k výběru jejich školy, kterou reprezentují, a to často bez ohledu na mé individuální potřeby a preference.

Některé stánky mají připravené letáky a brožury, ale i ty jsou plné obecných informací. Chybí konkrétní detaily o studijních plánech, specializacích a reálných zkušenostech studentů. Tento nedostatek konkrétních informací a jasných rozdílů mezi obory celou situaci nezlehčuje.

3.2.2. Shrnutí získaných informací ze scénáře

V této podkapitole jsme se zaměřili na problémy, kterým čelí středoškoláci při návštěvě veletrhu vysokých škol Gaudeamus. Zjistili jsme, že množství dostupných informací je ohromující a často povrchní. Prezentátoři univerzit poskytují obecné informace a snaží se propagovat své školy bez ohledu na individuální potřeby studentů. Studijní obory v rámci různých škol splývají a rozdíly mezi nimi nejsou dostatečně jasné, což ztěžuje výběr správného oboru.

Tento scénář ukazuje, že i když je veletrh Gaudeamus užitečným nástrojem pro získání přehledu o možnostech studia, nedostatek konkrétních informací a jasných rozdílů mezi obory představuje významnou překážku pro potenciální studenty. Tyto problémy podtrhují potřebu efektivnějších a konkrétnějších nástrojů, které by usnadnily výběr studijního oboru.

3.3. Scénář vlastní akce vysoké školy

3.3.1. Popis scénáře vlastní akce vysoké školy

Pojďme se podívat na třetí a poslední vzorový scénář a identifikovat problémy s výběrem správného IT programu při návštěvě Dne otevřených dveří na vysoké škole, nebo jiné vlastní akce jednotlivých vysokých škol a fakult. Představme si následující situaci: Jsem středoškolák hledající ideální obor. Rozhodl jsem se navštívit Den otevřených dveří na vybrané vysoké škole, abych se podrobněji seznámil s jejím prostředím a konkrétním IT programem.

Jakmile dorazím na místo, mám příležitost se setkat se všemi důležitými lidmi fakulty nebo školy, jako jsou děkan, jednotliví profesoři a další akademičtí pracovníci. Tato přítomnost klíčových osobností fakulty mi umožňuje získat hlubší vhled do struktury a kultury školy. Procházíme fakultu a ukazují nám možnosti a dispozice jejich laboratoří a učeben. Mám příležitost vidět, jaké vybavení je k dispozici a jaké technologie se používají ve výuce. Dostávám detailní informace o studijních programech, které fakulta nabízí, a mám možnost se zeptat na vše,

co mě zajímá. Program je uzpůsobený právě pro studenty ve stejné situaci jako jsem já, což znamená, že mnoho z mých otázek je předem zodpovězeno v průběhu prezentací a prohlídek.

Silnou stránkou této akce je možnost získat podrobné informace o tom, co mě čeká při studiu, a mít prostor pro osobní otázky. Často se jedná o specifické dotazy, které mi pomohou udělat informované rozhodnutí. Akce je navržena tak, aby zodpověděla nejčastěji kladené otázky studentů z minulých let, což mi pomáhá lépe pochopit, co mohu očekávat.

Na druhou stranu, tato akce vyžaduje, abych byl osobně na místě. To může být problematické, pokud se škola nachází v jiném kraji republiky. Cestování je časově náročné a může být komplikované, pokud se akce různých škol křižují. Pokud mám mnoho vysokých škol, které chci navštívit, může být tento proces velmi časově náročný a vyčerpávající.

3.3.2. Shrnutí získaných informací ze scénáře vlastní akce školy

V této podkapitole jsme se zaměřili na problémy a výhody spojené s návštěvou Dne otevřených dveří na vysoké škole. Zjistili jsme, že silnou stránkou této akce je možnost podrobně se seznámit s fakultou a studijním programem, získat detailní informace a osobně se setkat s klíčovými osobnostmi školy. Tato akce poskytuje prostor pro specifické otázky a je navržena tak, aby odpověděla na nejčastěji kladené otázky studentů ve stejné rozhodovací situaci.

Na druhou stranu, nutnost být osobně na místě může být problematická kvůli časové náročnosti a logistickým výzvám spojeným s cestováním. Akce různých škol se mohou křižovat, což může ztížit účast na všech akcích. Tento scénář ukazuje, že i když Den otevřených dveří nabízí mnoho výhod, existují také slabé stránky a překážky s ním spojené. Ve zbytku této práce se budeme zabývat sběrem dat, metodikou a návrhem řešení, které by tento proces zjednodušilo a zpřehlednilo. V následující kapitole se budeme zabývat metodikou výběru relevantních studijních IT oborů.

3.4. Shrnutí prezentace rozhodovací situace

V této kapitole jsme se podrobně zaměřili na různé scénáře, kterým čelí středoškoláci při výběru správného IT studijního programu. Prozkoumali jsme tři hlavní situace: online vyhledávání studijních oborů, návštěvu veletrhu vysokých škol Gaudeamus a účast na Dni otevřených dveří konkrétní vysoké školy.

3.4.1. Shrnutí on-line scénáře

Zjistili jsme, že při online vyhledávání je středoškolák zahlcen množstvím informací a možností. Výsledky jsou často nesystematické, neorganizované a zahrnují i irelevantní návrhy. To vede k frustraci a ztěžuje výběr správného oboru.

3.4.2. Shrnutí scénáře Gaudeamus

Návštěva veletrhu poskytuje přehled o různých školách a oborech, ale informace bývají často povrchní a zaměřené na propagaci. Prezentátoři neberou v úvahu individuální potřeby a preference studentů, což komplikuje rozhodování.

3.4.3. Scénář vlastní akce školy

Den otevřených dveří, nebo akce mu podobné, poskytuje detailní vhled do specifických programů a umožňuje osobní interakci s akademickými pracovníky. Nicméně nutnost fyzické přítomnosti a časová náročnost představují významné překážky.

3.4.4. Závěr

Tyto scénáře ilustrují komplexitu a obtížnost procesu výběru správného studijního programu. Identifikovali jsme, že klíčovými problémy jsou nepřehledné informace, povrchní prezentace a logistické výzvy spojené s osobními návštěvami. Na základě analýzy těchto scénářů jsme zjistili, že existuje naléhavá potřeba efektivnějších nástrojů, které by studentům usnadnily výběr studijního oboru. Naším cílem je vyvinout podpůrný online nástroj, který poskytne studentům více informací, díky kterým se budou moci zaměřit na užší množinu vysokých škol a oborů na akcích jako Gaudeamus a Dny otevřených dveří. Nebo v nejlepším případě jim přímo poskytne dostatek informací, aby se mohli sebevědomě rozhodnout.

V následující kapitole se budeme zabývat metodikou výběru relevantních studijních IT oborů. Zaměříme se na kritéria, která jsou důležitá pro hodnocení a výběr oborů, a na to, jak tato kritéria aplikovat při vývoji našeho online nástroje. Tato metodika bude základem pro vytvoření objektivního, spolehlivého a uživatelsky přívětivého systému, který pomůže studentům nalézt nejvhodnější studijní program odpovídající jejich individuálním potřebám a zájmům.

4. Metodika výběru relevantních oborů

4.1. Úvod

V této kapitole se budeme zabývat metodikou analýzy a výběru vysokých škol a konkrétních studijních oborů, které vyhovují naší definici z kapitoly s klíčovými pojmy. V dalších podkapitolách se tedy pokusíme nejprve definovat konkrétní seznam veřejných vysokých škol. Následně z těchto vysokých škol vybereme studijní obory, na nich akreditované, které se zabývají informačními technologiemi. V rámci této bakalářské práce, jak již bylo zmíněno, budeme brát v potaz pouze veřejné vysoké školy a studijní obory na nich akreditované. Proto z množiny vzdělávacích institucí vyřadíme vysoké školy, které jsou soukromé dle definice MŠMT a také vysoké školy státní. Státní vysoké školy eliminujeme z finální množiny, jelikož se jedná o policejní, či vojenské vysoké školy, jejichž studijní programy nejsou vyhovující pro kritéria, která jsme si vytyčili na začátku práce. Následně vyřadíme také veškeré komerční zaučovací kurzy, které mohou poskytnout zájemci o studium adekvátní vzdělání v sektoru informačních technologií, avšak nesplňují definici bakalářského studijního oboru.

V souladu se Zákonem č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů, jsou veřejnými vysokými školami v České republice jak je uvedeno v Příloze č. 1 k tomuto zákonu:

- Univerzita Karlova v Praze
- Univerzita Palackého v Olomouci
- České vysoké učení technické v Praze
- Vysoká škola báňská - Technická univerzita Ostrava
- Akademie výtvarných umění v Praze
- Vysoké učení technické v Brně
- Veterinární a farmaceutická univerzita Brno
- Masarykova univerzita v Brně
- Mendelova zemědělská a lesnická univerzita v Brně
- Akademie múzických umění v Praze
- Vysoká škola umělecko-průmyslová v Praze
- Janáčkova akademie múzických umění v Brně
- Univerzita Pardubice
- Vysoká škola chemicko-technologická v Praze
- Česká zemědělská univerzita v Praze
- Technická univerzita v Liberci
- Vysoká škola ekonomická v Praze
- Vysoká škola pedagogická v Hradci Králové
- Jihočeská univerzita v Českých Budějovicích
- Ostravská univerzita v Ostravě

- Slezská univerzita v Opavě
- Univerzita Jana Evangelisty Purkyně v Ústí nad Labem
- Západočeská univerzita v Plzni

4.2. Metodika výběru vysokých škol

Nejprve začneme vybírat vysoké školy. Ze seznamu vysokých škol vyřadíme veškeré instituce, které nemají akreditovaný ani jeden studijní obor, který by vyhovoval definici studijního IT oboru z kapitoly s klíčovými pojmy. Po aplikování naší definice jako filtru pro výběr vypadá zredukovaný seznam vysokých škol následovně.

- Univerzita Karlova v Praze
- České vysoké učení technické v Praze
- Vysoká škola báňská - Technická univerzita Ostrava
- Vysoké učení technické v Brně
- Masarykova univerzita v Brně
- Technická univerzita v Liberci
- Vysoká škola ekonomická v Praze
- Ostravská univerzita v Ostravě
- Slezská univerzita v Opavě
- Západočeská univerzita v Plzni

Posledním krokem celého procesu je pak výpis veškerých studijních oborů, které vyhovují námi stanoveným kritériím. Seznam těchto studijních oborů je následující.

Univerzita Karlova v Praze Matematicko-fyzikální fakulta

- Informatika
- Matematika pro informační technologie

České vysoké učení technické v Praze Fakulta informačních technologií

- Informační bezpečnost
- Manažerská informatika
- Počítačová grafika
- Počítačové inženýrství
- Počítačové sítě a Internet
- Počítačové systémy a virtualizace
- Softwarové inženýrství
- Teoretická informatika
- Umělá inteligence
- Webové inženýrství

České vysoké učení technické v Praze Fakulta elektrotechnická

- Elektrotechnika, energetika a management
- Kybernetika a robotika

- Otevřená informatika
- Otevřené elektronické systémy
- Softwarové inženýrství a technologie

Vysoká škola báňská - Technická univerzita Ostrava Fakulta elektrotechniky a informatiky

- Komunikační a informační technologie
- Počítačové systémy pro průmysl 21. století
- Informatika

Vysoké učení technické v Brně Fakulta elektrotechniky a komunikačních technologií

- Telekomunikační a informační systémy
- Informační bezpečnost
- Elektronika a komunikační technologie

Vysoké učení technické v Brně Fakulta informačních technologií

- Informační technologie

Masarykova univerzita v Brně Fakulta informatiky

- Programování a vývoj aplikací
- Kyberbezpečnost
- Informatika

Technická univerzita v Liberci

- Informační management
- Informační technologie
- Systémové inženýrství a informatika
- Technická kybernetika

Vysoká škola ekonomická v Praze Fakulta informatiky a statistiky

- Aplikovaná informatika
- Data Analytics

Ostravská univerzita v Ostravě Přírodovědecká fakulta

- Aplikovaná informatika
- Softwarové systémy
- Informatika

Slezská univerzita v Opavě Filozoficko-přírodovědecká fakulta

- Informatika
- Informační a komunikační technologie

Západočeská univerzita v Plzni Fakulta aplikovaných věd

- Softwarové inženýrství

- Počítačové vědy
- Informační systémy

4.3. Shrnutí

V této kapitole jsme se zabývali metodikou analýzy a výběru vysokých škol a studijních oborů. Nejprve jsme dle definice MŠMT zahrnuli všechny veřejné vysoké školy a následně jsme se zaměřili výhradně na veřejné vysoké školy, které poskytují studijní IT obory. Z těchto vysokých škol jsme vybrali pouze studijní obory, které se zabývají informačními technologiemi, a dospěli jsme k výslednému seznamu uvedeném v předešlé podkapitole.

Jak je vidět tak škol a oborů je mnoho a jsou různorodého typu a zaměření. Tato různorodost znamená, že každý obor nabízí specifickou kombinaci znalostí a dovedností, které mohou být pro studenty atraktivní v závislosti na jejich individuálních zájmech a kariérních cílech. Výběr vhodného oboru proto může být složitý a časově náročný proces. Aby se tento proces usnadnil, existují různé nástroje, jako jsou průvodci a volební kalkulačky, které pomáhají zájemcům o studium nalézt nejvhodnější obor na základě jejich preferencí a potřeb. Analýza těchto nástrojů a jejich efektivity při usnadňování výběru studijního IT oboru je podrobně rozebrána v další kapitole.

5. Analýza podpůrných nástrojů pro výběr studijních IT oborů

5.1. Úvod

Než jsem se pustil do vývoje aplikace, která by pomohla budoucím studentům s výběrem studijního oboru v oblasti IT, rozhodl jsem se nejprve prozkoumat již existující podpůrné nástroje. Moje cesta začala vyhledáváním na internetu a rozhovory s vrstevníky, současnými studenty a absolventy vysokých škol. Tito lidé mi poskytli cenné poznatky a nasměrovali mě k nástrojům, které sami používali nebo o nich slyšeli.

Při analýze jsem se zaměřil na tři hlavní aspekty: jak jednotlivé nástroje pracují s daty, jakým způsobem uživatelům poskytují informace a jak jsou tyto informace relevantní a užitečné. Cílem bylo pochopit silné a slabé stránky každého nástroje, abych mohl vyvinout aplikaci, která by co nejlépe naplnila potřeby budoucích studentů.

5.1.1. Vlastnosti optimálního nástroje

Než začneme analyzovat konkrétní podpůrné nástroje, je důležité zmínit, jak by měl takový optimální online nástroj vypadat, jaké by měl mít vlasnosti a co by měl uživateli poskytovat.

Optimální nástroj pro výběr studijního oboru by měl splňovat několik klíčových kritérií:

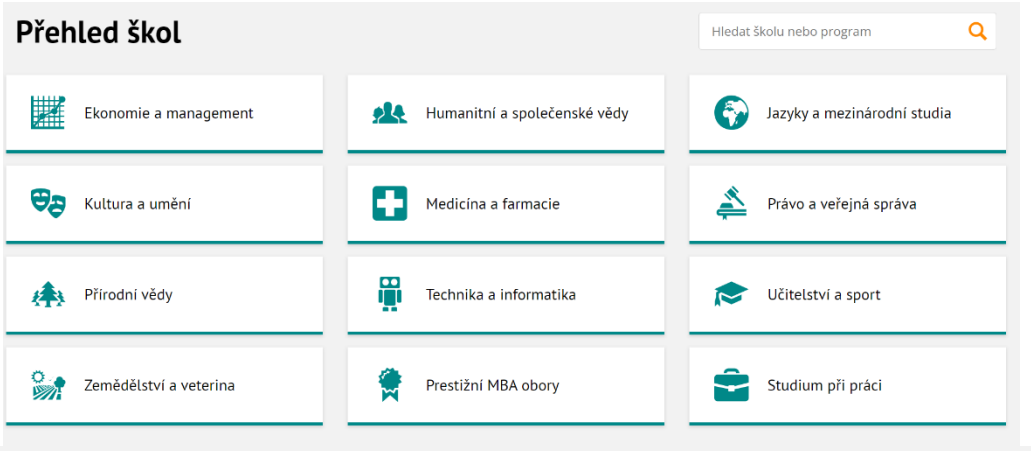
- **Přesnost a aktuálnost**: Poskytovat aktuální a přesné informace o dostupných studijních programech a jejich akreditacích.
- **Uživatelská přívětivost**: Rozhraní nástroje by mělo být intuitivní a snadno použitelné, aby i méně zkušení uživatelé mohli snadno najít potřebné informace.
- **Detailní filtrování**: Umožňovat detailní filtrování podle různých kritérií, jako jsou studijní programy, zaměření, lokalita a další obecnější relevantní aspekty.
- **Personalizovaná doporučení**: Na základě uživatelských preferencí poskytovat personalizovaná doporučení studijních oborů.
- **Komplexní informace**: Poskytovat hluboké a komplexní informace o studijních oborech, včetně možností kariérního uplatnění, předmětové náplně a dalších důležitých aspektů.

5.2. Vysokéškoly.cz

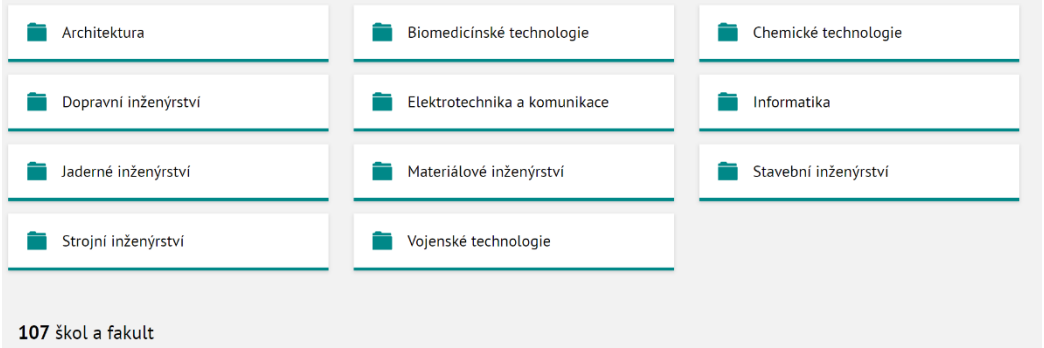
Webová stránka VysokéŠkoly.cz [2] je centrum informací, které pomáhá zájemcům, o studium na vysokých školách, seznámit se s co největším množstvím možností, které jsou jim k dispozici. Na stránce lze ke každé vysoké škole najít důležité informace jako jsou studijní programy dané školy, informace o přijímacím řízení, aktuality a kontakty. Elementy webové stránky, které jsou relevantní pro naše účely, jsou primárně záložka „Přehled škol“ a záložka s on-line testem. Tyto podpůrné nástroje si v následujících podkapitolách podrobně představíme a analyzujeme jejich užitečnost.

5.2.1. Přehled škol

Stránka Vysokéškoly.cz pokrývá 204 vysokých škol, které jsou organizovány dle atributů do filtrového vyhledávání. Každá vysoká škola má přiřazena atributy, které definují její zaměření. Uživatel při rozkliknutí přehledu škol je prezentován s primárním filtrem, kde má na výběr dvanáct kategorií vysokých škol. Po výběru primárního filtru je prezentován se sekundárním filtrem specifickým pro jeho primární výběr.



Obrázek 4: Ukázka primárních filtrů VysokéŠkoly.cz



Obrázek 5: Ukázka sekundárních filtrů na VysokéŠkoly.cz

V rámci této bakalářské práce se zabýváme vysokými školami se zaměřením na IT. Uvažujme tedy primární výběr kategorie „Technika a informatika“ pro zájemce o studium IT. Uživateli je následně zobrazen seznam sekundárních filtrů, kde jediné relevantní, pro našeho hypotetického

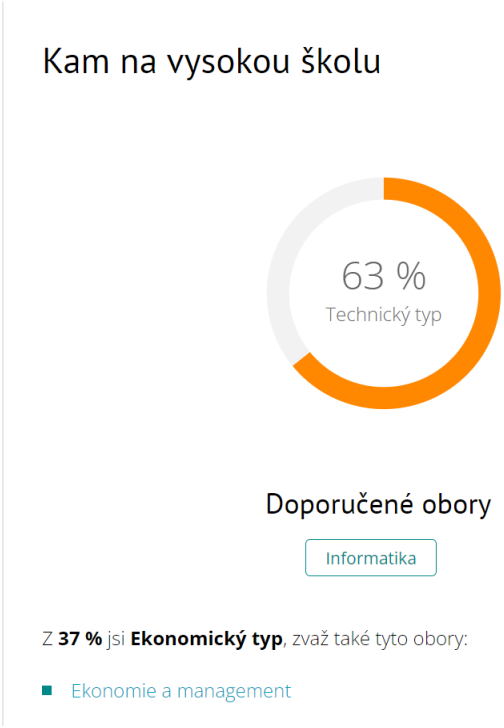
uchazeče o studium IT oboru, jsou záložky „Elektrotechnika a komunikace“ a „Informatika“. Tyto filtry zúží výčet vysokých škol z původních 204 na 22 (pro filtr „Elektrotechnika a komunikace“ a 62 (pro filtr „Informatika“). Uživatel následně může vyfiltrované školy procházet a dozvědět se o nich něco více, případně je přesměrován přímo na stránky vysoké školy, kde nalezne zbytek informací.

Silnou stránkou tohoto nástroje je intuitivnost ovládání a šířka vysokých škol, kterou nástroj bere v úvahu při filtrování. Slabou stránkou je však přesnost těchto filtrů, které jsou velice obecné a uživatel je zahlcen vysokými školami, které nevyhovují definici IT oboru, jak jsme si definovali v klíčových pojmech této bakalářské práce. Uživatel tedy musí na úkor vlastního času dělat hlubší průzkum stránek vysokých škol a nedostane konkrétní výsledek. Nástroj funguje jako takový leták vysokých škol, kde se dozví pouze povrchní informace.

5.2.2. Online test

On-line test na stránce vysokéškoly.cz ulehčuje výběr uchazeči pomocí interaktivního testu. Uživatel vždy vybírá mez dvěma možnostmi v testu, který je třicet otázek dlouhý. Na konci testu jsou odpovědi vyhodnoceny a uživateli je přiřazen primární a sekundární studijní typ dle odpovědí, které uživatel vybral. Studijní typ uživateli navrhuje jaký typ vzdělávací instituce by měl zvolit. Typy, které jsou přiřazovány jsou velice obecné např. „Technický typ“, „Ekonomický typ“, „Umělecký typ“, „Humanitární typ“, abych uvedl některé z potenciálních výsledků . Po přiřazení typů je uživateli přiřazen i obecný studijní obor. Uživatel následně může uživatel

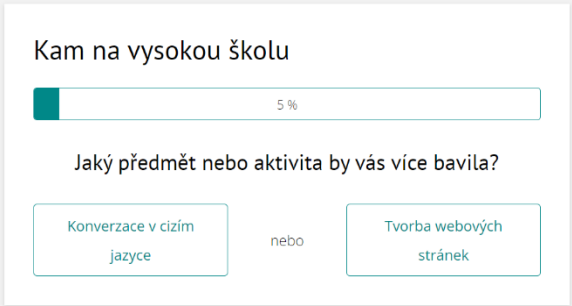
procházet přiřazené vysoké školy, které jsou vyfiltrovány pomocí filtrů zmíněných v předešlé podkapitole.



Obrázek 6: Ukázka výsledku on-line testu

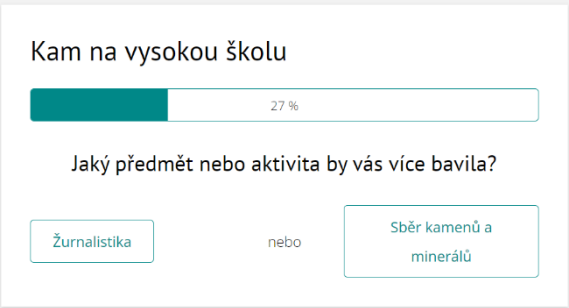
Silnou stránkou tohoto nástroje je eliminování vysokých škol, které nevyhovují uživatelově preferencím. Například uživatel, který se nevyvinul žádné úsilí průzkumu trhu vysokých škol, může velice rychle zjistit jaký studijní typ je a o jakých vysokých školách by měl nadále uvažovat. Ovšem slabou stránkou tohoto nástroje je obecnost výsledků jako takových. Odpověď „Ekonomický typ“ a „Technický typ“ je nicneříkající a uživatel stále nemá jasno jakou vysokou školu by měl zvolit. Dalším nedostatkem je struktura testu. Uživatel může být prezentován, aby si vybral jednu z možností, ač obě mohou být pro něj atraktivními volbami. Test proto může být nepřesný ve svých výsledcích a nemusí uživateli poskytovat požadovaný konkrétní výsledek.

Nemluvě o tom, že uživatel by se pomocí záložky „Přehled škol“, která je popsána v minulé podkapitole, dostal k totožným seznamům vysokých škol.



Obrázek 7: Ukázka dilema dvou přívětivých možností

Obrázek 8: Ukázka dilema dvou přívětivých možností



Obrázek 9: Ukázka dilema dvou irelevantních možností

5.3. ChatGPT

Jako poslední podpůrný nástroj pro analýzu jsem zvolil produkt od OpenAI a to jejich ChatGPT [1]. S pomocí několika příkazů jsem definoval své požadavky pro studium a popsal svou rozhodovací situaci a následně se tázal ChatGPT. Kvalita výstupů nebyla uspokojivá. A to hlavně z důvodů, že ChatGPT nemá ve své databázi aktuální IT studijní programy, ani všechny vysoké školy v České republice, které poskytují vzdělání v IT sféře, nebo i mimo obor informačních technologií. Těmito vlivy byly výstupy zatíženy chybovostí, kde ChatGPT nabízel programy na vysokých školách, které neexistují, nebo nevyhovují preferencím specifikovaných v příkazu. Proto jako podpůrný nástroj pro výběr studijního oboru ChatGPT není vhodným řešením.

5.4. Výsledek analýzy a shrnutí

V této kapitole jsme provedli analýzu existujících podpůrných nástrojů pro výběr studijního oboru. Zjistili jsme, že každý z těchto nástrojů má své silné a slabé stránky. Některé nástroje byly uživatelsky přívětivé a intuitivní, jiné zase nabízely široké spektrum informací o mnoha vysokých školách. Přesto žádný z analyzovaných nástrojů nedokázal plně uspokojit potřeby budoucích studentů optimálně.

Respondenti, se kterými jsem konzultoval výsledky analýzy, se shodli na tom, že existuje touha po nástroji, který by poskytoval konkrétní a přesné informace o studijních programech a nabízel personalizovaná doporučení na základě individuálních preferencí. Takový nástroj by jim usnadnil rozhodovací proces a poskytl větší jistotu při výběru studijního oboru. Většina tázaných by ocenila nástroj, který by eliminoval nutnost zdlouhavého průzkumu a poskytoval jasné a konkrétní výsledky. Například by uvítali možnost srovnání různých studijních programů na základě míry zaměření na praktické dovednosti, možností mezinárodních výměn, spolupráce s firmami v oboru, a také zpětnou vazbu od současných studentů a absolventů.

Závěrem této analýzy je, že na trhu podpůrných nástrojů pro výběr studijního oboru v oblasti IT neexistuje žádný nástroj, který by dokázal zprostředkovat uspokojivé, konzistentní a konkrétní výsledky či návrhy. Tento nedostatek na trhu potvrzuje smysl a potřebu vyvíjené aplikace, která je jedním z hlavních cílů této bakalářské práce. Vyvíjená aplikace má tedy potenciál významně přispět k usnadnění rozhodovacího procesu pro budoucí studenty a má své místo na trhu.

V následující kapitole se budeme zabývat sběrem dat. Tato data budou stěžejní odrazový můstek pro hlubší chápání uživatelských preferencí a relevantních kritérií, které studenti hodnotí jako důležité při výběru své vysněné studijní instituce a studijního oboru na ní se nacházejícím.

6. Sběr dat k definici kritérií

V této kapitole se budeme zabývat jednou z nejdůležitějších částí této bakalářské práce a to sběrem dat. V rámci implementace dotazníkové části vyvíjené aplikace, která má fungovat jako podpůrný nástroj k výběru studijního IT oboru, je nutné zjistit jaká kritéria jsou pro uživatele důležitá v jejich rozhodovací situaci. V následujících podkapitolách budeme popisovat definici a transformaci souboru kritérií, které jsou stěžejní částí dotazníkové části našeho podpůrného nástroje.

6.1. Úvodní formulace kritérií

Prvním krokem k identifikaci nejdůležitějších kritérií pro výběr studijního IT oboru byla formulace vlastních předpokladů a zkušeností. Tato předběžná kritéria byla stanovena na základě mých osobních zkušeností a přesvědčení o tom, co je důležité pro studium v oblasti IT. Tato kritéria byla formulována ve formě otázek a výčet vypadal následovně:

- Kolik matematických předmětů je ve studijním plánu?
- Jaká je rozmanitost cizích jazyků v rámci studijního programu?
- Je součástí výuky fyzika?
- Je součástí výuky robotika?
- Má studijní obor navazující magisterské studium?
- Jak prestižní je univerzita, na které je studijní obor?
- Kde se univerzita nachází?

Cílem úvodní formulace kritérií bylo vytvořit první iteraci souboru kritérií, která budou dále rozšířena a upravena na základě dodatečného sběru dat a zpětné vazby od dalších respondentů. Po formulaci prvního souboru kritérií čistě z vlastních zkušeností bez jakéhokoliv výzkumu jsem započal dlouhý sběr dat, který se skládal z několika dílčích částí. Tento sběr měl za úkol ověřit relevanci a úplnost stanovených kritérií. Případně kritéria doplnit a upravit na základě již zmíněné zpětné vazby. Jednotlivé dílčí části si teď představíme v následujících podkapitolách.

6.2. Osobní rozhovory

Po formulaci první iterace kritérií jsem započal individuální pohovory s potenciálními uživateli této aplikace. Skupina respondentů byla pestrá, skládala se ze studentů středních škol, které teprve čeká volba bakalářského studia, současných studentů bakalářských a magisterských programů a na konec i absolventů bakalářských a magisterských programů. Rozhovory probíhali jako nejvýše deseti minutové dialogy, kde jsem se respondentů ptal na otázku „Která kritéria jsou/byla pro vás nejdůležitější pro výběr bakalářského studijního oboru? “. Na tuto otázku jsem

dostal mnoho odpovědí. Některé odpovědi přesně zapadaly do mého původního souboru kritérií, avšak některé odpovědi byly unikátní. Z unikátních odpovědí jsem vytvořil nová kritéria a svůj původní soubor kritérií obohatil o následující:

- Jak praktický/teoretický je studijní obor?
- Má studijní obor/univerzita partnerské firmy?
- Jak časově náročný je studijní obor?
- Poskytuje studijní obor Erasmus?
- Jak široké jsou možnosti výběru tělocviku?

Jednu z nejčastějších odpovědí však nebylo možné transformovat v kritérium. Tímto kritériem bylo „osobní doporučení“. Osobní doporučení bohužel nelze transformovat na kritérium do dotazníku, protože každý z respondentů a uživatelů dostává od svého okolí jiné osobní doporučení. Nutné je však podotknout, že kritéria mohou být ovlivněna a zkreslena, kvůli poměrně malému množství respondentů i přes maximální snahu o objektivitu. Z výsledků rozhovorů plyne, že rozhodovací situace je pro každého studenta individuální. Musíme tedy složit co nejvíce komplexní soubor kritérií, který bude zahrnovat nejčastěji relevantní kritéria, abychom byli schopni zohlednit všechny aspekty rozhodovací situace. Proto musíme náš sběr dat ještě dále rozšířit, což bude středobodem následující podkapitoly.

6.3. Oficiální stránky univerzit

V této podkapitole se budeme zabývat oficiálními stránkami univerzit a jejich studijními obory. Prvním krokem bylo shromáždění veškerých informací o studijních oborech a univerzitách, které bereme v potaz v rámci tohoto projektu. Nejprve jsem tedy sepsal všechny důležité informace o jednotlivých univerzitách a studijních oborech. Zaměřoval jsem se primárně na obsahy jejich doporučených průchodů studijních programů. Tím je myšlen soupis všech povinných a povinně volitelných předmětů při standartním úspěšném průchodu studijním programem. Pokud se jedná o volitelné předměty, tak si student může zvolit libovolný předmět z libovolné fakulty, kterou dovoluje studijní program. Tato vysoká míra volitelnosti vedla k zúžení zorného pole projektu na povinné a povinně volitelné předměty programu, bez kterých nelze studijní program úspěšně dokončit.

V této fázi projektu jsem se zaměřoval primárně na rozdíly v předmětech mezi jednotlivými studijními programy. Jelikož se zabýváme IT studijními programy jsou jejich studijní plány z velké části postaveny podobně, proto bylo klíčové mezi jednotlivými studijními plány najít rozdíly. Tyto rozdíly pak definují relativní profil jednotlivého programu v porovnání se zbytkem studijních programů. V rámci výběru všech relevantních studijních oborů jsem narazil na studijní obory, které obsahovali vysoké množství diametrálně odlišných specializací. Příkladem takového

studijního programu je například ČVUT FIT Informatika, kde si student na začátku studia vybírá mezi desíti různými specializacemi, které se mezi sebou liší vice, či méně. Jako příklad diametrálně odlišných specializací je například “Informační bezpečnost”, “Manažerská informatika” a “Počítačová grafika”. Toto zjištění vedlo k upravení celkového počtu studijních oborů. Rozhodl jsem se v rámci této bakalářské práce s takovýmito studijními obory s vysokou mírou specializace operovat jako s jedním oborem. Tudíž došlo ke sjednocení několika studijních oborů do jednoho a změně celkového počtu studijních oborů v rámci tohoto projektu. Tyto programy a jejich kritéria jsou hodnoceny jiným způsobem, avšak o hodnocení kritérií budeme hovořit až v nadcházejících kapitolách.

V rámci průzkumu oficiálních stránek se soubor kritérií rozrostl o kritéria, která souvisí s předměty vyučovaných na daných oborech. Soubor kritérií byl tedy rozšířen o následující kritéria:

- Vyučuje obor předměty Internetu věcí?
- Vyučuje obor předměty webového vývoje?
- Vyučuje obor předměty UNIXových systémů?
- Zabývá se obor Low-code a No-code platformami?
- Vyučuje obor Low-lvl programovací jazyky?
- Vyučuje obor High-lvl programoací jazyky?
- Vyučuje obor předměty datové analýzy?
- Zabývá se obor multimédii a počitačovou grafikou?
- Vyučuje obor předměty zabývající se podnikovou informatikou a business stránkou IT?
- Má obor brzkou specializaci, nebo obecnější výuku?

Abychom shrnuli tuto podkapitolu, tak došlo k největšímu rozšíření souboru kritérií za současný životní cyklus projektu. Narazili jsme a vyřešili problém se studijními obory s mnohými různými specializacemi. Studijní obory, které mají velké množství diametrálně odlišných specializací jsme sloučili do jednoho oboru, na který bude v implementované aplikaci uživatel upozorněn, že se jedná o obor s vysokou mírou uzpůsobitelnosti. Rozpoznali jsme klíčové rozdíly mezi jednotlivými studijními obory a finalizovali soubor studijních oborů, které figurují jako potenciální kandidáti uživateli.

6.4. Shrnutí sběru dat

V této kapitole jsme podstoupili první důležitý krok k návrhu a implementaci našeho podpůrného nástroje. Začali jsme formulováním úvodní hypotézy a společně s ní první iterací souboru kritérií. Od prvního návrhu souboru kritérií jsme se pomocí osobních rozhovorů

s budoucími, současnými i vystudovanými studenty přiblížili k definici souboru kritérií, které jsou důležité pro uživatele naší budoucí aplikace. Z těchto rozhovorů se potvrdila všechna kritéria z úvodní hypotézy, dále jsme přidali další kritéria, která v souboru kritérií do té doby nefigurovala. Tento nově obohacený soubor kritérií jsme dále rozšířili o kritéria z oficiálních stránek univerzit a studijních oborů. Po hlubším pohledu na studijní obory jsme předefinovali vstupní množinu studijních oborů na konečných 34 oborů. Nakonec jsme zformulovali finální podobu nejdůležitějších kritérií.

7. Hodnocení kritérií a studijních oborů

V této kapitole se budeme zabývat hodnocením jednotlivých studijních oborů v kontextu kritérií, které jsme definovali v minulé kapitole. Hodnocení provádíme, abychom popsali silné a slabé stránky jednotlivých studijních oborů, abychom získali unikátní „otisk prstu“ každého studijního oboru.

7.1. Škála hodnocení

Každý studijní obor je hodnocen na škále od 1 do 5 pro každé kritérium. Škála byla zvolena pro co nejpřesnější hodnocení jednotlivých kritérií v kontextu ostatních oborů. Jednoduché binární hodnoty (true/false) nejsou dostatečné, protože některé obory se věnují předmětům ve větší hloubce. Hodnota 1 reprezentuje minimální naplnění kritéria, což znamená, že daný obor buď vůbec dané kritérium nesplňuje, nebo jej splňuje jen v minimální míře vzhledem k ostatním oborům. Oproti tomu hodnota 5 indikuje maximální naplnění kritéria ve srovnání s ostatními obory. Tento způsob hodnocení nám umožňuje rozlišovat mezi obory, kde určité předměty tvoří jádro výuky, a těmi, kde jsou tyto předměty pouze doplňkem..

7.2. Proces hodnocení studijních oborů

Proces hodnocení studijních oborů zahrnoval několik kroků, které byly nutné k dosažení objektivního a spravedlivého hodnocení. Nejprve jsme shromáždili veškeré dostupné informace o jednotlivých studijních oborech, přičemž jsme se zaměřili na povinné a povinně volitelné předměty, jejich obsah a strukturu studijních programů. Poté jsme analyzovali získaná data v kontextu jednotlivých kritérií. Každé kritérium bylo posuzováno samostatně pro každý studijní obor. Hodnocení bylo provedeno na základě dostupných informací z oficiálních stránek univerzit, studijních materiálů a konzultací s bývalými studenty daného studijního oboru, nebo univerzity. Výsledné hodnocení pro každé kritérium bylo diskutováno a schváleno respondenty individuálních rozhovorů, aby byla zajištěna co nejvyšší míra objektivity. Hodnocení proběhlo v druhém kole osobních rozhovorů s respondenty, toto kolo proběhlo po stanovení kritérií a k ním přidruženým otázkám. Konkrétní kroky hodnocení studijních oborů byly jmenovitě následující:

1. Sběr dat: Získání detailních informací o studijních oborech z oficiálních zdrojů, včetně studijních plánů, sylabů a dalších relevantních dokumentů. Dále byly informace ověřeny a konzultovány se současnými studenty i absolventy daných studijních oborů
2. Analýza kritérií: Každé kritérium bylo analyzováno v kontextu dostupných informací. Byly posouzeny aspekty jako hloubka pokrytí tématu, nebo jeho

úplná absence. Například bylo zkoumáno, jak důležitý je daný předmět pro úspěšný průchod daným studijním programem (povinný/povinně volitelný).

3. Hodnocení: Na základě analýzy bylo každému studijnímu oboru přiřazeno hodnocení na škále od 1 do 5 pro každé kritérium. Hodnocení bylo prováděno nezávisle a konzultováno do doby, než bylo dosaženo konsensu. Tento krok zahrnoval pečlivé srovnání mezi obory, aby byla zajištěna konzistence a integrita hodnocení.
4. Zápis hodnot: Po dosažení konsensu byly hodnoty daného kritéria pro jednotlivé studijní obory zaznamenány a přidány do hodnotící tabulky v Excelu. Tento strukturovaný zápis umožňuje snadnou aktualizaci a srovnání výsledků v kompaktním provedení.

Níže můžete vidět příklad ohodnoceného studijního oboru. Konkrétně se jedná o ČVUT FEL Softwarové inženýrství a technologie (viz. Obrázek 10).

Jazyky četnost	Matematika	Elektrotec	Robotika	Tělocvik	Teoretičnc	Specializace	Grafika	Návaznost	Prestiž/Ná	UNIX	IoT	WebDevel	Low/NoCode	High-lvl	Low-lvl	Zahraničí	Data analyt	Podniky
5	3	2	1	4	3	4	3	2	4	2	3	4	5	5	2	4	3	4

Obrázek 10: Příklad hodnoceného oboru

7.3. Algoritmus doporučení studijních oborů

V této kapitole se budeme zabývat algoritmem, který je základem pro doporučování studijních oborů na základě uživatelských odpovědí v testu. Algoritmus funguje na principu vážené odchylky, která bere v úvahu jak hodnocení jednotlivých kritérií, tak i jejich prioritu zadanou uživatelem.

Princip algoritmu

Algoritmus začíná tím, že uživatel prochází test a odpovídá na otázky týkající se jednotlivých kritérií, jako je například úroveň matematiky, rozmanitost cizích jazyků, a další. Každá odpověď uživatele je poté porovnána s hodnocením příslušného kritéria pro každý studijní obor. Výsledkem tohoto porovnání je vážená odchylka.

- Příklad výpočtu odchylky:
- Pokud má studijní obor hodnocení 3 v kritériu matematika a uživatel zadá hodnotu 5, odchylka je 2 (|5 - 3|).

- Pro jiný obor s hodnocením 5 v matematice je odchylka 0 (|5 - 5|).

Váha kritéria

Uživatel také nastavuje prioritu pro každé kritérium na škále od 1 do 10. Tato priorita určuje, jak důležité je dané kritérium pro uživatele. Základní nastavení je priorita 5, což představuje střední důležitost. Čím vyšší je priorita, tím větší váhu má dané kritérium v celkovém hodnocení.

Výpočet vážené odchylky

Odchylka pro každé kritérium je poté vynásobena prioritou daného kritéria. Tento výpočet zajišťuje, že kritéria s vyšší prioritou mají větší vliv na celkovou odchylku.

Výpočet:

- Vážená odchylka = Odchylka × Priorita
- Například, pokud je odchylka 2 a priorita 7, vážená odchylka bude 14 (2 × 7).

Převod vážené odchylky

Celková odchylka je součet všech vážených odchylek veškerých kritérií ke každému z oborů v databázi. Tato hodnota je poté přepočítána na procentuální hodnotu, která reprezentuje míru shody studijního oboru s uživatelskými preferencemi dle jejich odpovědí

Seřazení a doporučení

Studijní obory jsou následně seřazeny podle celkové vážené odchylky od nejnižší po nejvyšší. Uživateli je poté zobrazeno TOP10 oborů s nejnižší váženou odchylkou, respektive s nejvyšší procentuální hodnotou shody s jeho odpověďmi.

Shrnutí algoritmu

V této kapitole jsme se podrobně zabývali algoritmem doporučení studijních oborů, který je založen na vážené odchylce. Tento algoritmus bere v úvahu jak uživatelské odpovědi na otázky týkající se jednotlivých kritérií, tak i jejich prioritu. Výsledkem je seznam studijních oborů seřazených podle shodnosti s uživatelskými preferencemi, což uživateli umožňuje lépe se rozhodnout při výběru vhodného studijního programu.

7.4. Shrnutí

V této kapitole jsme se zaměřili na kvantifikaci kritérií a hodnocení studijních oborů. Definovali jsme škálu hodnocení od 1 do 5, kde 1 znamená minimální a 5 maximální naplnění

kritéria. Tento systém umožňuje srozumitelně hodnotit jednotlivé studijní obory. Dále jsme popsali proces hodnocení, který zahrnoval sběr dat z oficiálních zdrojů, analýzu kritérií, hodnocení každého oboru a zaznamenání výsledků do excel tabulky. Tento strukturovaný přístup zajišťuje objektivitu a přesnost hodnocení, což je nezbytné pro poskytování užitečných doporučení uživatelům aplikace.

Dalším důležitým bodem byl algoritmus doporučení studijních oborů, který využívá váženou odchylku. Uživatel odpovídá na otázky a nastavuje prioritu kritérií, přičemž algoritmus počítá odchylku pro každý obor, která je vynásobena prioritou. Výsledná vážená odchylka je přepočítána na procentuální hodnotu shodnosti a studijní obory jsou seřazeny podle nejnižší vážené odchylky, aby uživateli poskytly nejlepší možné doporučení.

Tato kapitola nám poskytla jasný a strukturovaný přehled o hodnocení studijních oborů, čímž položila základy pro vytváření spolehlivých a informovaných doporučení pro budoucí studenty. V následujících kapitolách nás bude čekat výběr technologií a návrh aplikace.V této kapitole jsme si popsali principy algoritmu hodnocen a rámcové funkcionality jsou již také známe, tak můžeme přejít k výběru technologií pro implementaci.

8. Analýza technologií pro implementaci

Tato kapitola přináší přehled technologií vhodných pro realizaci našeho projektu. Postupně se budeme zabývat dílčími částmi systému. Nejprve budeme analyzovat frontend, backend a nakonec databázi. Analýzu technologií zakončíme volbou nejvhodnějších technologií pro implementaci našeho projektu.

8.1. Frontend

Frontend je část webové aplikace, se kterou uživatel přímo interaguje. Tato část, označovaná také jako "klientská strana aplikace", zahrnuje všechny vizuální prvky, jako jsou barvy, styly, obrázky, grafy, tabulky a navigační nabídky. Hlavním cílem frontendu je zajistit rychlý, efektivní a spolehlivý uživatelský zážitek, přičemž musí být zajištěna responzivnost na různých zařízeních a obrazovkách [4] . Frontendové technologie jsou založeny na HTML (Hyper Text Markup Language), CSS (Cascading Style Sheets) a JavaScriptu.

8.1.1. Angular

Angular je robustní framework pro vývoj frontendu, vyvinutý společností Google, založený na TypeScriptu. Angular umožňuje vývojářům vytvářet dobře strukturované a udržovatelné aplikace díky své modulární architektuře a bohaté sadě nástrojů [5].

Výhody:

- Silná podpora pro Dependency Injection, což zjednodušuje správu závislostí
- Vysoká modularita usnadňuje údržbu a rozšiřování aplikace
- Široká komunita a množství dostupných knihoven a nástrojů

Nevýhody:

- Strmá křivka učení pro začátečníky
- Velká komplexnost může vest k vyšší náročnosti na výkon u velkých aplikací

8.1.2. React

React je populární JavaScriptová knihovna pro vytváření uživatelských rozhraní, vyvinutá společností Facebook. React je založen na komponentním přístupu, což umožňuje vytvářet opakovaně použitelné UI komponenty.[6]

Výhody:

- Jednoduchý a intuitivní pro rychlé použití
- Znovupoužitelné komponenty
- Velice jednoduché na testování

Nevýhody:

- React pokrývá pouze UI část vývoje, proto je potřeba využít kombinaci dalších technologií, abychom zajistili např. routing
- Nedostatečná dokumentace
- Časté změny. Pro nezkušeného vývojáře může být složité udržet krok s konstatními novinkami
- Některé funkce, jako například routing, vyžadují externí knihovny

8.1.3. Vue.js

Vue.js je progresivní framework pro JavaScript, který se zaměřuje na jednoduchost a snadnou integraci s dalšími projekty. Vue.js je vhodný pro vytváření single-page aplikací a je velmi přívětivý pro začínající vývojáře[7][7]

Výhody:

- Snadná křivka učení
- Felxibilita a škálovatelnost
- Čitelnost

Nevýhody:

- Menší komunitní podpora a ekosystém v porovnání s Angular a React
- Menší podpora pro rozsáhlé aplikace ve srovnání s Angular

8.1.4. Volba technologie pro implementaci frontendu

Pro vývoj frontendu byla zvolena technologie Angular. Angular je jediný framework zvažovaných možností, který využívá TypeScript. TypeScript je nadmnožina JavaScriptu, která přidává statické typování, což umožňuje psát přesnější a méně chybový kód. Díky tomu je TypeScript velice konkrétní a deskriptivní jazyk, který usnadňuje vývoj a údržbu komplexních aplikací. Angular nabízí mnoho funkcí "out of the box", což znamená, že poskytuje vestavěné

nástroje a knihovny, které jsou nezbytné pro náš projekt. Tato výbava zahrnuje například nástroje pro správu formulářů, HTTP klienty pro komunikaci s backendem, a vestavěnou podporu pro Dependency Injection, která zlepšuje modularitu a testovatelnost aplikace. Dalším důvodem pro výběr Angularu je má předchozí zkušenost s touto technologií. Mám již zkušenosti s vývojem aplikací v Angularu, což zkrátí čas potřebný na seznámení se s nástroji a metodikami a umožní rychlejší zahájení vývoje. Tento faktor je zvláště důležitý pro dodržení časového harmonogramu projektu a zajištění vysoké kvality výsledné aplikace.

8.2. Backend

Backend, někdy taky nazývaný „serverová strana“, je část aplikace, která se stará o interní logiku, databáze, serverové operace a reprezentaci informací v systému, které jsou následně frontendem zobrazovány na klientské straně aplikace. Slouží k administraci a zpracování dat, ať už jde o správu obsahu, uživatelských účtů nebo jiných požadavků z frontendu[8].

8.2.1. Java

Java je všestranný programovací jazyk používaný k vývoji širokého spektra aplikací, včetně webových backendů. Java je známá svou přenositelností mezi platformami a robustními bezpečnostními funkcemi[9].

Výhody:

- Přímočarý proces učení
- Objektově orientovaný jazyk
- Nezávislost platformy

Nevýhody:

- Limitace ve výkonu oproti např. C++
- Výzvy spojené s GUI vývojem

8.2.2. Node.js

Node.js je runtime pro JavaScript na straně serveru, který umožňuje vytvářet škálovatelné a výkonné aplikace. Díky asynchronní povaze je Node.js ideální pro aplikace s vysokou mírou vstupů a výstupů[10].

Výhody:

- Rychlý a efektivní díky asynchronnímu zpracování
- Jednotné použití JavaScriptu pro frontend i backend
- Nabízí jednoduchou škálovatelnost

Nevýhody:

- Postrádá knihovnovou podporu
- Méně vhodný pro CPU-intenzivní operace.

8.2.3. PHP

PHP je jedním z nejrozšířenějších programovacích jazyků používaných pro vývoj webových aplikací. Je to serverový skriptovací jazyk, který je navržený speciálně pro webový vývoj a může být vložen do HTML[11].

Výhody:

- Open source a zdarma
- Nezávislý na platformě
- Kratší doba učení, kvůli jedoduchosti a přímočarosti

Nevýhody:

- Bezpečnostní rizika, pokud není kód správně napsán a zabezpečen
- Používání vice funkcí PHP framework a nástrojů způsobuje špatný výkon online aplikací.
- Některé starší verze PHP mohou mít problémy s výkonem a stabilitou

8.2.4. Volba technologie pro implementaci backendu

Pro vývoj backendu byla zvolena technologie Java. Java je velmi populární programovací jazyk pro vývoj backendových částí webových aplikací. Nabízí širokou škálu knihoven a frameworků, které výrazně usnadňují vývoj. Jedním z nejpoužívanějších frameworků je Spring Boot, který umožňuje rychlé vytvoření funkční aplikace s minimální konfigurací. Spring Boot poskytuje přednastavené konfigurace, které zrychlují vývoj a zároveň umožňují přizpůsobení specifickým požadavkům projektu. Díky tomu je možné vytvořit aplikaci, která je připravena k nasazení v různých prostředích bez nutnosti rozsáhlých úprav. Vývoj v jazyce Java také poskytuje možnost oddělit vývojové prostředí od produkčního prostředí. To znamená, že aplikaci lze vyvíjet a testovat lokálně a následně nasadit do produkčního prostředí bez větších problémů. Tento

přístup zajišťuje vysokou stabilitu a minimalizuje riziko chyb při nasazení aplikace. Výběr Java jako backendové technologie je podpořen její robustností, výkonem a širokou podporou komunity, což z ní činí ideální volbu pro tento projekt.

8.3. Databáze

Databáze jsou organizované kolekce dat, které umožňují efektivní správu, přístup a aktualizaci uložených dat. Databáze jsou spravovány prostřednictvím systémů pro správu databází (DBMS).

Jsou dva hlavní typy databází a to relační a nerelační. Relační databáze je strukturovaná, což znamená, že data jsou organizována v tabulkách. Mnohokrát data v těchto tabulkách mají vztahy s jednou nebo více závislostí[12].

8.3.1. Relační database

Relační databáze ukládají data ve strukturovaných tabulkách, které jsou propojeny předem definovanými vztahy. Každá tabulka obsahuje sloupce (atributy) a řádky (záznamy). Relační databáze umožňují snadné propojení a manipulaci s daty pomocí SQL.

Výhody[13].

- Jednoduchý a intuitivní model dat
- Vysoká úroveň integrity da konzistence dat
- Jednoduché k použití

Nevýhody[14].

- Nedostatek škálovatelnosti
- Problémy s údržbou
- Vyžadují dobrou strukturu

8.3.2. Nerelační database

Nerelační databáze je typ databáze, který neukládá data v tabulkách, ale místo toho v jakémkoliv formátu, který je nejlepší pro daná data. Z tohoto důvodu jsou nerelační databáze uzpůsobeny pojmout nestrukturovaná data[15]

Výhody.

- Vysoká flexibilita a dynamické změny datových modelů
- Horizontální škálovatelnost
- Možnost ukládání nestrukturovaných nebo semi-strukturovaných dat.

Nevýhody:

- Nižší úroveň integrity a konzistence dat ve srovnání s relačními databázemi.
- Složitější dotazování a manipulace s daty

8.3.3. Volba technologie pro databázi

Pro tento projekt byla vybrána relační databáze PostgreSQL. PostgreSQL je open-source databázový systém známý svou robustností, výkonem a flexibilitou, ideální pro komplexní dotazy a velké objemy dat. PostgreSQL podporuje pokročilé datové typy a umožňuje definovat vlastní funkce a operátory, což zvyšuje flexibilitu při manipulaci s daty. Nabízí plnou podporu ACID(Atomicity, Consistency, Isolation, Durability) transakcí, což zajišťuje bezpečnost a spolehlivost databázových operací, což je klíčové pro práci s citlivými daty. Další výhodou PostgreSQL je její škálovatelnost, což umožňuje efektivní správu rostoucího objemu dat a vyššího zatížení systému. Její integrace s Java a Spring Boot je plynulá, což usnadňuje vývoj a správu databázových operací. PostgreSQL byla vybrána pro svou spolehlivost, výkon, flexibilitu a silnou podporu komunity, což z ní činí ideální volbu pro databázové potřeby tohoto projektu.

9. Návrh aplikace

Tato kapitola popisuje návrh vyvíjené aplikace. V následujících podkapitolách se budeme zabývat návrhem aplikace, nejprve si představíme aktéry systému, poté popíšeme požadavky na systém, následně vytvoříme diagram tříd a diagramy použití. V neposlední řadě se budeme zabývat diagramem nasazení celého systému a grafickým návrhem aplikace.

9.1. Úvod do návrhu aplikace

Podpůrné nástroje ze sekce analýzy podpůrných nástrojů poukazují, že výsledná aplikace by měla mít jednoduché a intuitivní uživatelské rozhraní. Rozhraní by mělo být obohaceno o prvky, které pomohou lépe popsat uživatelské preference.

Jedním z těchto prvků je pole pro určení priority daného kritéria, na které je uživatel dotazován v naší navrhované aplikaci. Dále je velice důležitá samotná formulace otázek, které budou uživateli prezentovány v dotazníku. Tyto otázky musí být jednoduché na pochopení a co nejpřesněji popisovat kritéria, jež jsou pro uživatele aplikace důležité v rozhodovacím procesu. Po celou dobu vyplňování dotazníku by měl uživatel vědět, na které kritérium je daná otázka zaměřena a co jednotlivé odpovědi znamenají.

Další stěžejní částí aplikace je prezentace výsledků po vyplnění dotazníku. Výsledky by měly obsahovat pouze nejrelevantnější studijní obory, které nejlépe naplňují požadavky uživatele aplikace. Finální pořadí studijních oborů by mělo být jednoznačné, srozumitelné a poskytovat odkaz na oficiální stránky daného oboru, pro minimalizaci jakéhokoliv nedorozumění.

. Další důležitou dílčí částí aplikace je prezentace všech univerzit a k nim přidruženým studijním oborům, které bereme v potaz. Tato sekce by měla obsahovat základní informace o univerzitě, názvy studijních oborů a odkaz na oficiální stránky pro zjištění dodatečných informací. Poslední částí je pak sekce zabývající se metodologií, kde uživatel může najít veškeré informace o algoritmu, metodice filtrace univerzit a studijních oborů a jakým způsobem byly studijní obory hodnoceny dle jednotlivých kritérií.

9.2. Aktéři a systémové požadavky

Aktéři

- Aktér je role, která komunikuje s jednotlivými případy užití. V této roli může být obsazen uživatel nebo externí systém. Aktérem tedy může být např. Uživatel, Administrátor, SMS server nebo dokonce Čas. Aktér inicializuje nějaký případ užití [16].

Náš systém je jednoduchý a obsahuje pouze jednoho aktéra, kterým je uživatel. Všichni uživatelé aplikace mají stejná práva a přístupy a mohou se dostat ke stejným informacím.

Funkční požadavky

Funkční požadavky jsou požadavky, které vyžaduje koncový uživatel specificky, jako základní výbavu, kterou by měl systém nabízet. Všechny tyto funkcionality musí být nutně zahrnuty do systému[17]. Mimo jiné podrobně popisují, jak by systém měl reagovat na jednotlivé vstupy a na základě těchto vstupů provádět dané operace. Specifikují, co má systém dělat, jaké operace je uživatel v systému schopen provádět a jaké jsou výstupy jednotlivých operací. Mezi hlavní funkční požadavky patří:

- Uživatel může vyplnit dotazník s otázkami zaměřenými na jednotlivá kritéria.
- Systém musí umožnit uživateli nastavit prioritu pro každé kritérium.
- Po vyplnění dotazníku systém vyhodnotí výsledky a zobrazí uživateli seznam doporučených studijních oborů.
- Uživatel musí mít přístup k detailním informacím o doporučených oborech včetně odkazu na oficiální stránky.

Nefunkční požadavky

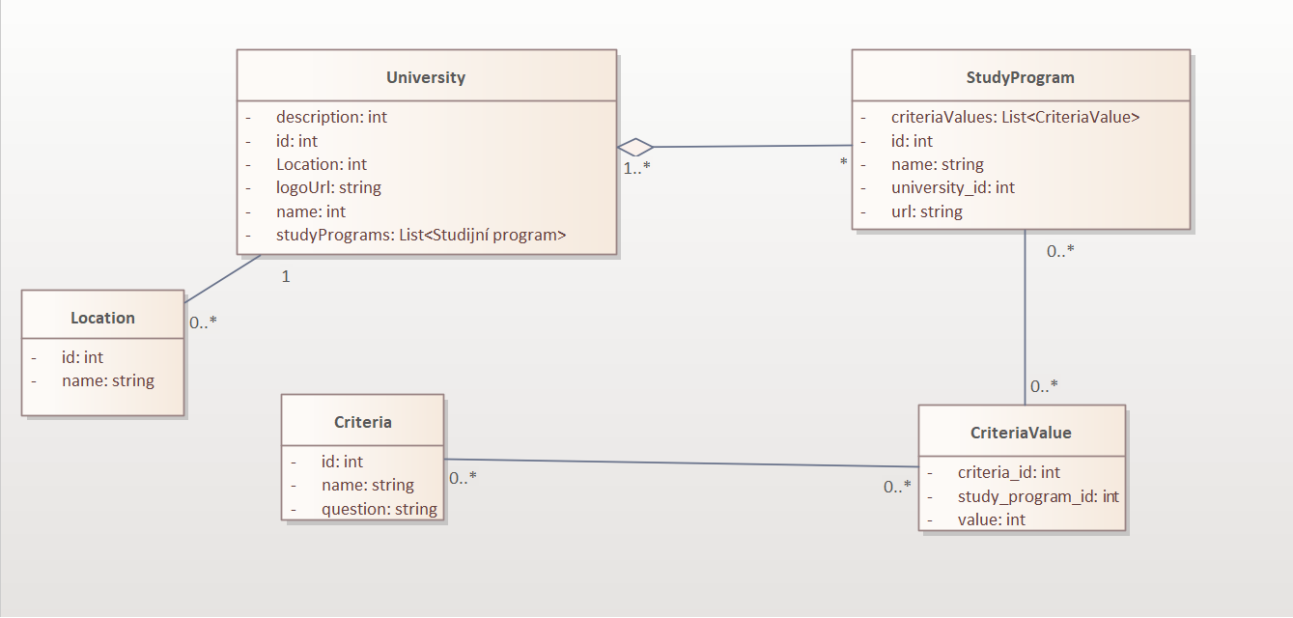
Nefunkční požadavky, neboli NFR, jsou množinou specifikací, které popisují operační možnosti a omezení. Jedná se vlastně o požadavky, které naznačují jak dobře systém operuje, obvykle mezi ně patří rychlost, bezpečnost, spolehlivost datová integrita a podobné. Naše nefunkční požadavky pak definujeme následovně [18]:

- Systém musí být dostupný a stabilní
- Systém musí být škálovatelný

- Aplikace musí být intuitivní a snadno použitelná i pro technicky méně zdatné uživatele.
- Systém musí být testovatelný
- Systém musí být rozšiřitelný
- Systém musí být rychlý a responzivní

9.3. Diagram tříd

V softwarovém inženýrství ,v jazyce UML, je diagram tříd statickým strukturovým diagramem, který popisuje strukturu systému, tím že ukazuje třídy systému, jejich atributy, operace a vztahy mezi objekty.[20] Diagram tříd je vytvořen nezávisle na implementaci a slouží jako základní stavební kámen pro následný vývoj. Na jeho základě se navrhují databázové tabulky tak, aby odpovídaly definovaným třídám a jejich atributům.



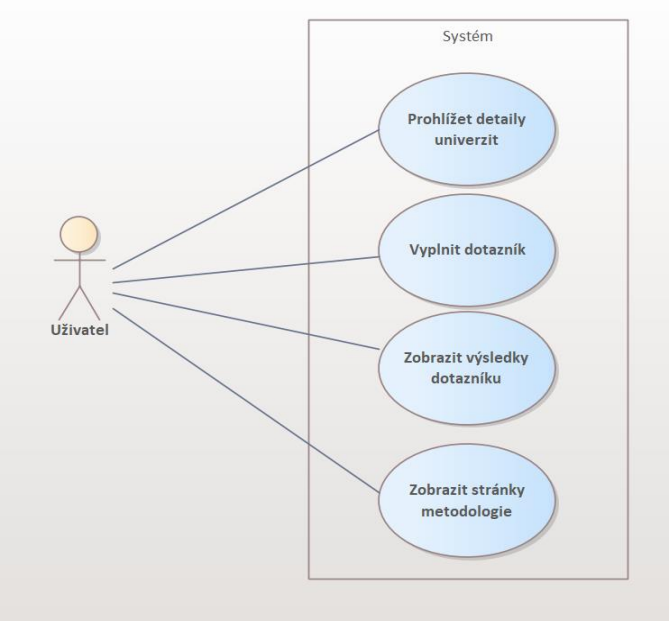
Obrázek 11: Diagram tříd

Na diagramu výše lze vidět všech pět entit, které mezi sebou mají vztahy. Hlavními entitami jsou entity univerzita, studijní program a kritéria. Entita univerzita obsahuje seznam studijních

programů, které lze na univerzitě najít, a dále atributy jako je název a lokace univerzity. Entita studijní program má důležitý atribut criteriaValues, který obsahuje seznam CriteriaValues. Tento seznam pomáhá udržovat hodnoty pro kritéria hodnocení studijního oboru. Kritéria mají se studijními programy vazební tabulku CriteriaValue, která umožňuje systému provázat hodnoty kritérií se studijními obory pro funkci hodnocení studijních programů v kontextu jednotlivých kritérií. Tabulka Kritéria pak drží v atributech název a otázku, která je pokládána při průchodu dotazníkovou částí aplikace.

9.4. Diagram použití

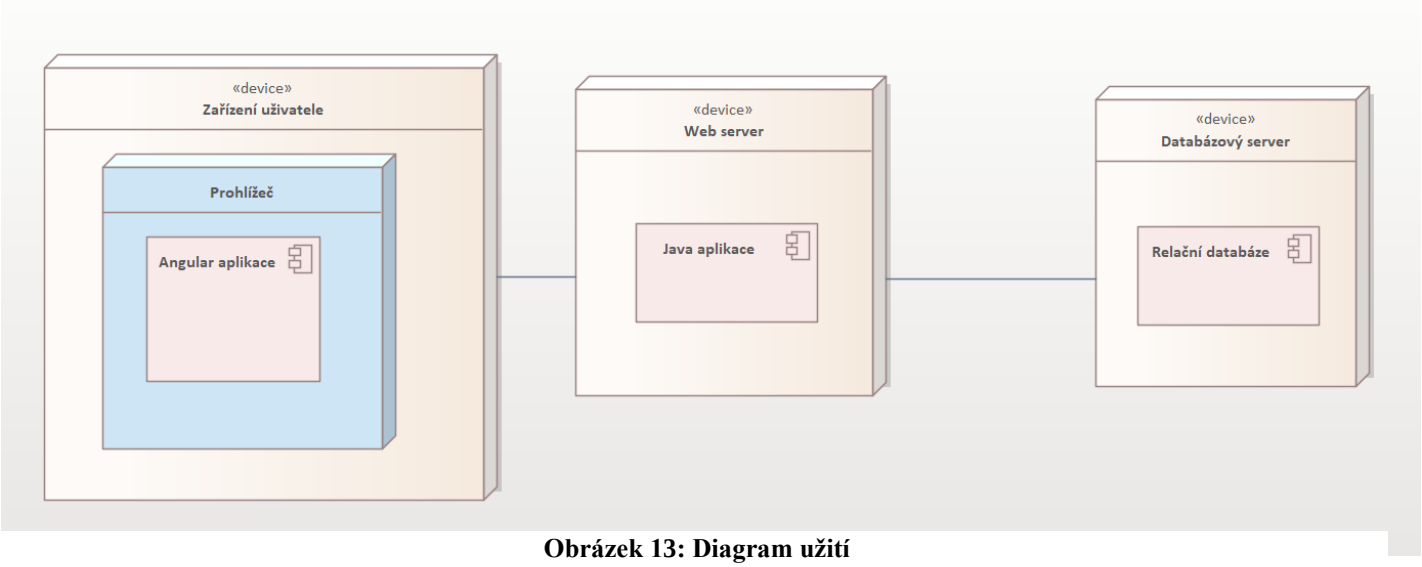
Diagram použití je klíčovým nástrojem pro systémový návrh. Poskytuje vizuální reprezentaci interakce uživatele se systémem. Reprezentuje interakci mezi aktéry a systémem a bere na zřetel plnění specifických cílů[20].Diagramy použití ilustrují různé scénáře interakcí mezi uživatelem a systémem. Tyto diagramy zobrazují základní aktivity, jako je registrace, přihlášení, vyplňování dotazníků a zobrazení výsledků. Diagramy také ukazují, jak jednotlivé funkce systému spolupracují a jak uživatelé s nimi interagují.



Obrázek 12: Diagram užití

9.5. Diagram nasazení

Diagram nasazení zobrazuje způsob, jakým bude aplikace rozmístěna v produkčním prostředí. Tento diagram znázorňuje jednotlivé komponenty systému, jejich umístění na serverech a vzájemnou komunikaci. Pomáhá zajistit, aby byla aplikace správně integrována a optimalizována pro provoz. Na diagramu výše je znázorněno nasazení, kde koncové uživatelské zařízení má zapnutý browser, na kterém běží Angular aplikace, která zajišťuje frontend. Na webovém serveru běží Java aplikace zprostředkující backend operace a komunikaci s databází. Poslední komponentou je databázový server, na kterém běží databázový systém. Tento diagram ukazuje, jak jednotlivé části aplikace spolupracují a jak jsou distribuovány mezi různými servery, což je klíčové pro zajištění efektivní a bezpečné komunikace v rámci celého systému.

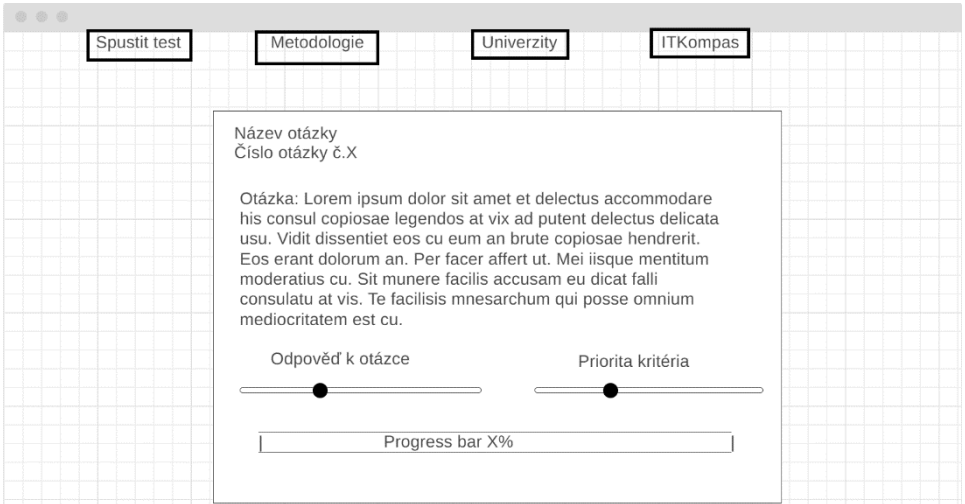


Obrázek 13: Diagram užití

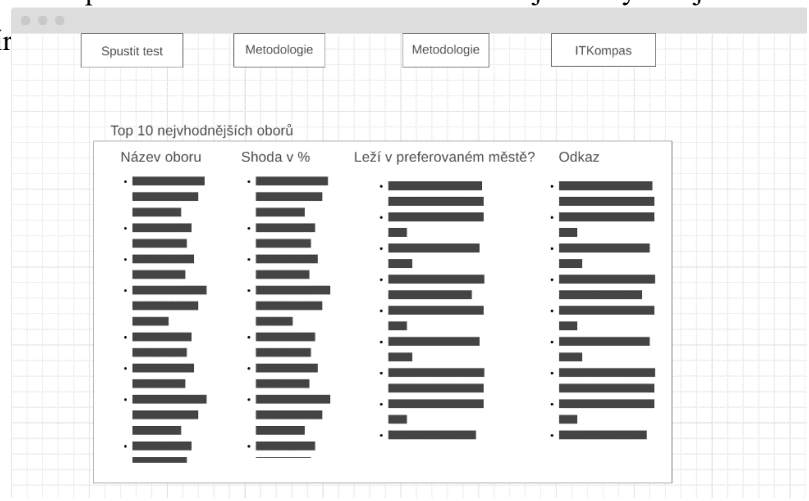
9.6. Wireframes

Wireframe je vizuální průvodce, který představuje kostru uživatelského rozhraní aplikace nebo webové stránky. Slouží k znázornění rozložení jednotlivých prvků na stránce a jejich vzájemných vztahů, bez detailního zaměření na vizuální styl nebo grafické detaily. Wireframe pomáhá vývojářům, designérům a ostatním zainteresovaným stranám porozumět, jak bude aplikace nebo webová stránka fungovat, a slouží jako základ pro další fáze návrhu a vývoje.

Níže můžete vidět příklady tří wireframů, které nám budou sloužit jako šablona pro následující implementaci. Na prvním wireframu níže můžete vidět návrh obrazovky, kde uživatel prochází dotazníkovou částí aplikace a je mu zobrazena otázka. Uživatel následně odpoví pomocí jednoho z posuvníkem a druhým posuvníkem určí prioritu kritéria (viz. Obrázek 14).

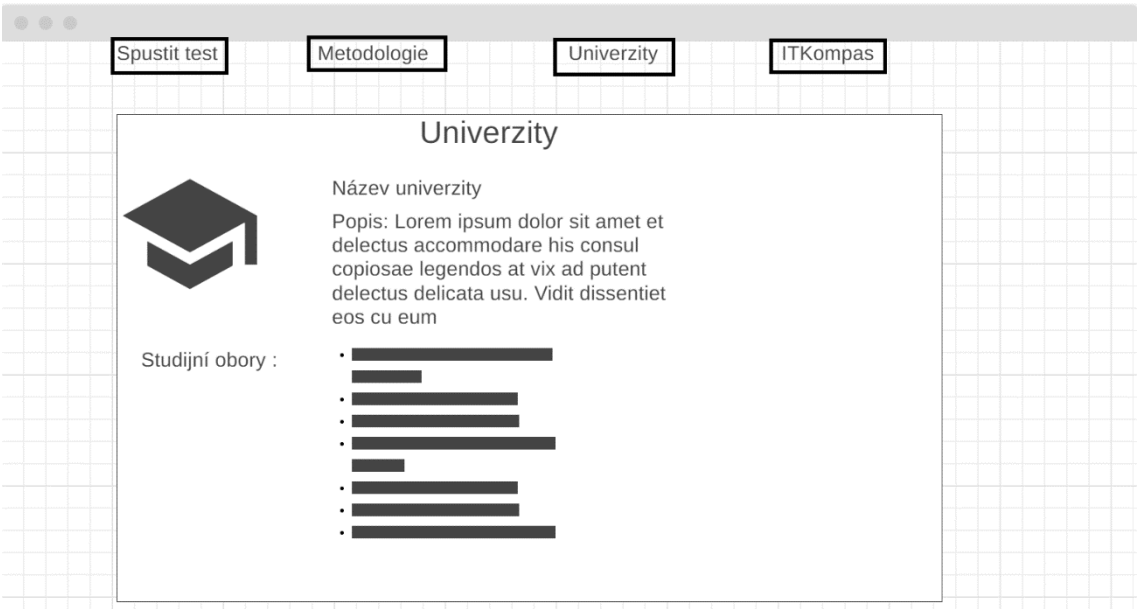


V druhém wireframe níže můžete vidět návrh obrazovky pro zobrazení výsledků zmíněné dotazníkové části aplikace. Uživateli se zobrazí 10 nejvíce vyhovujících studijních oborů společně s mír



Obrázek 15: Wireframe výsledků

Poslední wireframe níže (viz. Obrázek 16) znázorňuje návrh obrazovky pro zobrazení všech univerzit a k nim přidruženým studijním oborům. V rámci univerzity se zobrazuje popis a logo, kdežto u studijního oboru se zobrazuje jméno a odkaz na stránky oboru.



9.7. Shrnutí návrhu

V této kapitole jsme popsali návrh vyvíjené aplikace. Začali jsme úvodem do návrhu aplikace, kde jsme zdůraznili důležitost jednoduchého a intuitivního uživatelského rozhraní. Dále jsme představili aktéry systému, což je v našem případě pouze jeden uživatel a definovali systémové požadavky, které jsou klíčové pro vývoj a testování aplikace.

Následně jsme vytvořili diagram tříd, který poskytuje vizuální reprezentaci struktury dat v aplikaci a popisuje vztahy mezi jednotlivými entitami, jako jsou univerzita, studijní program a kritéria. Diagram použití ilustruje různé scénáře interakcí mezi uživatelem a systémem, zatímco diagram nasazení znázorňuje, jak bude aplikace rozmístěna v produkčním prostředí.

V neposlední řadě jsme se zabývali wireframy, které definují rozložení jednotlivých komponent ve vyvíjeném systému a slouží jako šablony pro implementaci uživatelského rozhraní.

Nyní přejdeme k další kapitole, která se bude zabývat implementací aplikace, kde se zaměříme na technické detaily vývoje a nasazení systému.

10. Implementace aplikace

V této kapitole se budeme zabývat hlavním postupem při implementaci aplikace. Začneme popisem backendové části, následně popíšeme část frontendovou a zakončíme databází.

10.1. Backend

V této podkapitole je popsáno všechno co se týče backendového vývoje vyvíjené aplikace. Primárním subjektem této podkapitoly je popsání základních principů pro tvorbu funkčního backend prostředí, které je schopné přijímat požadavky z klientské strany (frontendové části). V kapitole také najdete snímky obrazovek ze samotného vývoje.

10.1.1. Vývojové prostředí

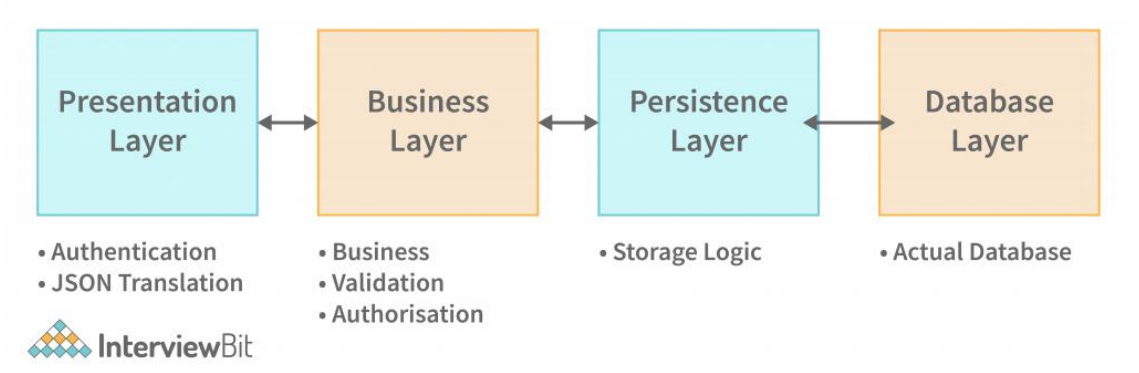
Jakžto vývojové prostředí pro vývoj backendu bylo zvolena IntelliJ IDEA, protože nabízí plnou podporu vývoje v Javě a mám s ní v porovnání s alternativami největší zkušenosti.

10.1.2. Založení projektu

Framework, který jsem zvolil pro implementaci, byl Spring Boot ze stejného důvodu jako bylo zvoleno vývojové prostředí. Pro inicializaci projektu jsem použil webové stránky Spring Initializr, což je online nástroj, který vygeneruje základní projektovou strukturu do konfiguračního souobru. Tuto strukturu následně může vývojář obohatit o další závislosti důležité pro vývoj. Po vygenerování základní projektové struktury je možné aplikaci ve vývojovém prostředí spustit a následně monitorovat její aktivitu standartně na lokálním portu počítače 8080.

10.1.3. Struktura projektu

Implementaci backendu a serverové části je nutné rozdělit do vrstev. Těmito vrstvami jsou vrstva prezentační, business, perzistentní a databázová. Tyto vrstvy si podrobněji popíšeme.



Obrázek 17: Popis struktury ve vrstvách

Prezentační vrstva

Tato vrstva se stará o zpracování a příjem dotazů od uživatele skrz klientskou stranu aplikace. Třídy, které tuto vrstvu reprezentují v Spring Boot označujeme anotací @RestController. Jednotlivé metody pak anotujeme a specifikujeme typ HTTP požadavku, který zpracovávají.

```
Martin Krejčí *
@CrossOrigin(origins = "http://localhost:4200")
@RestController
@RequestMapping("/calculation")
public class CalculationController {

    2 usages
    private final CalculationService calculationService;
    2 usages
    private final StudyProgramService universityService;

    Martin Krejčí
    public CalculationController(CalculationService calculationService, StudyProgramService universityService) {
        this.calculationService = calculationService;
        this.universityService = universityService;
    }

    Martin Krejčí *
    @PostMapping
    public List<SchoolScore> calculate(@RequestBody UserInput userInput) {
        List<SchoolScore> calculate = calculationService.calculate(userInput, universityService.getSchoolsForCalculation());
        return calculate;
    }
}
```

Obrázek 18: Příklad prezentační vrstvy

Business vrstva

Tato vrstva řeší problematiku business logiky aplikace. Jedná se o klíčovouo součást backendové architektury. Business vrstva se zabývá reakcí aplikace na různé požadavky a události, které se odehrávají v systému. Je zodpovědná za validaci dat, zpracování požadavků, komunikaci s perzistentní vrstvou a správu transakcí.

```java
@Component
public class CalculationService {
    1 usage
    private final CriteriaRepository criteriaRepository;
    2 usages
    private final int questionCount;

    Martin Krejčí
    public CalculationService(CriteriaRepository criteriaRepository) {
        this.criteriaRepository = criteriaRepository;
        this.questionCount = (int) criteriaRepository.count();
    }


    1 usage  Martin Krejčí
    public List<SchoolScore> calculate(UserInput userInput, List<School> schools) {
        List<Answer> answers = userInput.answers();
        List<Integer> cities = userInput.cities();
        return schools.stream()  Stream<School>
                .map(school -> calculateSchoolScore(answers, school, cities))  Stream<SchoolScore>
                .sorted(SchoolScore::compareTo)
                .limit( maxSize: 10)
                .toList();

    }

    1 usage  Martin Krejčí
    public SchoolScore calculateSchoolScore(List<Answer> answers, School school, List<Integer> cities) {
        int schoolDeviations = 0;
        var answerIterator  : Iterator<Answer>  = answers.iterator();
        var criteriaIterator  : Iterator<Integer>  = school.criteria().iterator();
        while (answerIterator.hasNext() && criteriaIterator.hasNext()) {
            var answer  : Answer  = answerIterator.next();
            var criteria  : Integer  = criteriaIterator.next();
            var deviation  : int  = Math.abs(answer.value() - criteria);
            var prioritizedDeviation = deviation * answer.priority();
            schoolDeviations += prioritizedDeviation;
```

Obrázek 19: Příklad business vrstvy kód 1

```java
        return new SchoolScore(school.id(), convertDeviationToScore(schoolDeviations, questionCount), matchedCities);
    }

    1 usage
    private static final int MAX_PRIORITY = 10;
    1 usage
    private static final int MAX_VALUE_DEVIATION = 4;

    1 usage  Martin Krejčí
    private BigDecimal convertDeviationToScore(int deviation, int questionCount) {
        var maxDeviation = new BigDecimal( val: questionCount * MAX_PRIORITY * MAX_VALUE_DEVIATION);
        var absoluteScore  : BigDecimal  = maxDeviation.subtract(new BigDecimal(deviation));
        BigDecimal onePercent = maxDeviation.divide(new BigDecimal( val: 100),  scale: 3, HALF_UP);
        return absoluteScore.divide(onePercent,  scale: 0, HALF_UP);
    }


}
```

Obrázek 20: Příklad business vrstvy kód 2

Perzistentní vrstva

Poslední vrstvou, kterou si budeme představovat je vrstva perzistentní. Tato vrstva je zodpovědná za správu a ukládání dat v databázi. Poskytuje však take abstrakci and databázovými operacemi, což umožňuje aplikaci komunikovat s databází bez nutnosti past konkrétní SQL dotazy. Je proto klíčovou komponentou pro zajištění konzistence a integrity dat. Mezi zodpovědnosti perzistentní vrstvy patří CRUD (Create, Read, Update, Delete) operace, transakční management, dotazování a filtrování dat.

```
10  @Repository
11
12  public interface UniversityRepository extends JpaRepository<University, Integer>{
13
        1 usage  Martin Krejčí
14      @Query("SELECT DISTINCT u.location FROM University u")
15      List<Location> findDistinctLocations();
16
17  }
```

Obrázek 21: Příklad perzistentní vrstvy 1

Business objekt

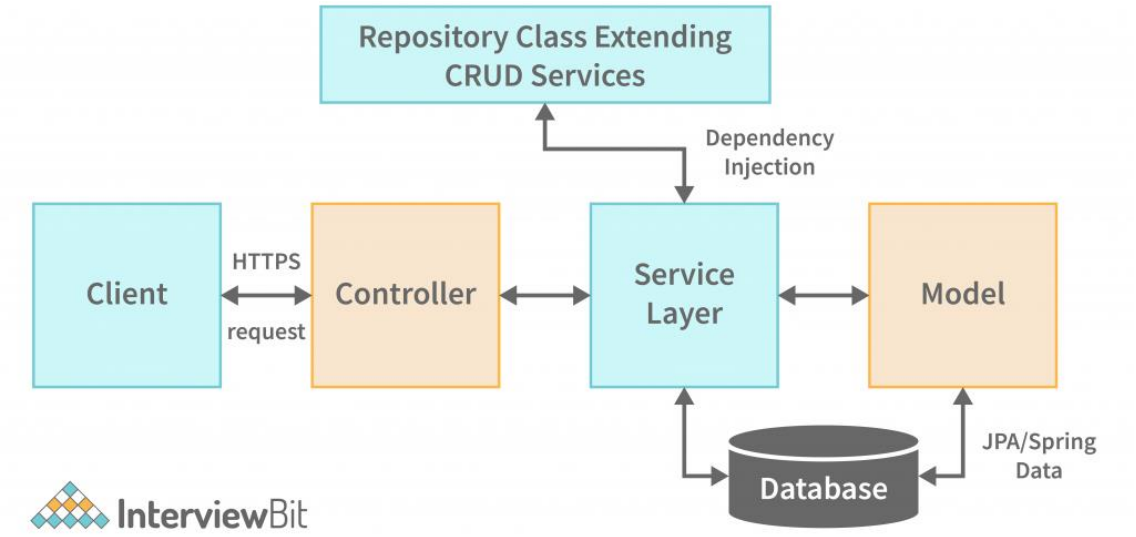
Business objekty jsou komponenty, které v architecture aplikacereprezentují základní entity a jejich chování v rámci doménové logiky aplikace. Tyto objekty zapouzdřují data a metody, které definují business logiku. Mimo jiné business objekty poskytují čisté a srozumitelné rozhraní pro manipulaci s daty a provádění operací. Příklad můžete vidět v kódu na další stránce (viz. Obrázek 22)

```java
@Table(name = "studyprogram", schema = "itcompass")
public class StudyProgram {

    2 usages
    @Id
    @GeneratedValue(strategy = jakarta.persistence.GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

    3 usages
    @JsonIgnore
    @ManyToOne
    @JoinColumn(name = "university_id")
    private University university;

    2 usages
    @Column(name = "name")
    private String name;

    2 usages
    @Column(name = "url")
    private String url;

    2 usages
    @OneToMany(mappedBy = "studyProgram")
    private List<CriteriaValue> criteriaValues;

    1 usage  Martin Krejčí
    public int getId() { return id; }

    Martin Krejčí
    public void setId(int id) { this.id = id; }

    1 usage  Martin Krejčí
    public University getUniversity() { return university; }
```

Obrázek 22: Příklad business vrstvy v kódu

10.1.4. Závěr

V předešlých podkapitolách jsme si představili vývoj bakcendové části aplikace, která slouží ke zpracování požadavků od uživatele, manipulaci s daty a jejich následnému uložení do databáze. Zpracovaná data figurují v kontrolerech, které volají metody v servisních třídách. Servisní třídy, neboli business vrsta, provádějí veškeré operace potřebné pro splnění business logiky a následně volají patřičné repozitáře, pokud je nutné komunikovat s databází. Tento proces je znázorněn v následujícím obrázku (viz.



23: Popis finální architektury

10.2. Frontend

V této části je popsáno jak nastavit projekt pro vývoj frotnendu pomocí Angularu, jak pracovat s touto technologií a jak celou aplikaci správně implementovat. Nakonec jsouo v této části popsány základní principy a postupy spojené s vývojem frontendové části aplikace.

10.2.1. Vývojové prostředí

Pro vývoj fronendové části aplikace bylo zvoleno vývojové prostředí Visual Studio Code. Toto vývojové prostředí poskytuje široký výběr rozšíření, které usnadňují čitelnost a implementaci výsledného kódu. Visual Studio Code také nabízí skvělou podporu pro vývoj TypeScriptu a Angularu, take je zde možnost vytvořit si vlastní server pro testování frontendu, pokud vývojář ještě nemá implementovaný backend.

10.2.2. Založení projektu

K založení projektu byl použit příkaz “ng new Frontend”, který vytvoří plně nakonfigurovanou aplikaci Angular, kterou stačí dale jen přizpůsobovat. Pro spuštění tohoto příkazu stačí mít na počítači nainstalovaný Node.js a Angular CLI. Po vytvoření projektu můžete aplikaci spustit příkazem “ng serve”, čímž dojde ke kompilaci a spuštění aplikace, která je standartně dostupná na lokálním počitači na portu 4200.

10.2.3. Základní práce s Angularem

Při vývoji v Angularu je nutné znát základy HTML, CSS a TypeScriptu. Angular používá komponentově orientovaný přístup, kde každá komponenta představuje izolovanou a

opakovatelně použitelnou část aplikace, která vrací HTML a logiku specifickou pro danou část aplikace.

Komponenty

Komponenty jsou základním stavebním blokem Angular aplikace. Každá komponentná čítá tři implementační a jednu testovací část: HTML šablony, TypeScript třídy, CSS styly a TypeScript testovací třídy. Komponenty jsou definovány pomocí anotace @Component, jež specifikuje šablonu, styl a selektor komponenty.

```
<div class ="app">
  <header>
            <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top" id="mainNav">
           <div class="container px-4">
              <a class="navbar-brand" href="/question">Spustit pomocníka</a>
              <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false"
              <div class="collapse navbar-collapse" id="navbarResponsive">
                 <ul class="navbar-nav ms-auto">
                    <li class="nav-item"><a class="nav-link" href="/university">Univerzity</a></li>
                    <li class="nav-item"><a class="nav-link" href="/about">Metodologie</a></li>
                    <li class="nav-item"><a class="nav-link" href="">ITKompas</a></li>
                 </ul>
              </div>
           </div>
        </nav>
  </header>
  <br>
  <br>
  <br>
  <main>
    <router-outlet></router-outlet>
  </main>
        <!-- Footer-->
        <footer class="py-5 bg-dark">
           <div class="container px-4">
              <p class="m-0 text-center text-white">Vytvořeno v rámci bakalářské práce Martina Krejčího</p>
           </div>
          <div class="container px-4"><p class="m-0 text-center text-white">Copyright &copy; Bakalářská práce Martina Krejčího 2024</p></div>
      </footer>
</div>
```

Obrázek 24: Příklad komponenty v kódu

Angular App-routing

Angular router usnadňuje vývojářům přiřadit každé URL adrese specifickou komponentu, která se při návštěvě dané URL adresy vykreslí. Toto umožňuje velice snadnou navigaci mezi dílčími částmi aplikace bez nutnosti načítat celou stránku znovu.

```
import { Routes } from '@angular/router';
import{ HomeComponent } from './pages/home/home.component';
import { QuestionsComponent } from './pages/questions/questions.component';
import { UniversitiesComponent } from './pages/universities/universities.component';
import { AboutComponent } from './pages/about/about.component';
import { CalculationComponent } from './pages/calculation/calculation.component';
export const routes: Routes = [
    { path: '', component: HomeComponent },
    { path: 'question', component: QuestionsComponent },
    { path: 'university', component: UniversitiesComponent },
    { path: 'about', component: AboutComponent },
    {path: 'calculation', component: CalculationComponent}
];
```

Obrázek 25: Ukázka routingu v angular

10.2.4. Komunikace se serverem

Pro komunikaci se serverem v Angularu je možné použít základní HTTP klient, který je součástí Angular framework. Angular HTTP klient poskytuje snadný a efektivní způsob, jak provádět HTTP požadavky a zpracovávat odpovědi

```
1  import { Injectable } from '@angular/core';
2  import { HttpClient } from '@angular/common/http';
3  import { Observable } from 'rxjs';
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class QuestionService {
9    private apiURL = 'http://localhost:8081/question/all';
10   //private state -> ukladat stav otazek -> priorita + odpoved na otazku
11
12   constructor(private http:HttpClient) { }
13   getQuestions(): Observable<string[]> {
14     return this.http.get<string[]>(this.apiURL);
15   }
16   sendResults(data: any): Observable<any> {
17     console.log(data);
18     return this.http.post('http://localhost:8081/calculation', data);
19   }
20 }
```

Obrázek 26:Ukázka komunikace se serverem

10.2.5. Závěr

Angular využívá moderní webové technologie a přidává k nim vlastní strukturované přístupy, díky nimž je vývoj webových aplikací, které komunikují s backendem, rychlý, jednoduchý a efektivní. Tento fakt byl jedním z hlavních důvodů volby Angularu pro implementaci frontendu. Díky komponentovému přístupu a silné podpoře TypeScriptu nabízí Angular robustní a škálovatelné řešení pro frontendový vývoj.

10.3. Databáze

Pro ukládání důležitých dat je nutné mít databázi, jelikož vyvíjíme aplikaci lokálně, tak si pro vývoj můžeme dovolit použít lokální databázi. Nasazení aplikace pro uživatelské testery je řešeno přes Cloudflare tunel, který nastaví bezpečné, ale dočasné spojení mezi lokálním portem v počítači a internetem.

10.3.1. Připojení database

Připojení databáze není nic složitého, jelikož na backendu využíváme Springboot. Tento framework za nás řeší nespočet věcí a jediné nutné kroky pro úspěšné připojení databáze jsou následující. Do konfiguračního souboru “application.properties” zapíšeme pár řádků, které specifikují adresu databáze, uživatelské jméno, heslo a konkrétní ovladač. Po této konfiguraci se připojení automaticky vytvoří po spuštění aplikace.

```
spring.application.name=ITKompas
spring.datasource.url=jdbc:postgresql://localhost:5432/ITCompass
spring.datasource.username=postgres
spring.datasource.password=1234
#spring.jpa.hibernate.ddl-auto=update
#spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true
spring.jpa.show-sql=true
server.port=8081
```

Obrázek 27: Ukázka připojení k databázi

10.3.1. Tvorba tabulek

Stejně jako připojení databáze, tak i tvorba tabulek je díky používání framework Springboot snadné. Pro vytvoření tabulky stačí použít anotaci @Entity nad deklarací třídy a přidat libovolný název tabulky pomocí anotace @Table

```
import jakarta.persistence.Entity;
import jakarta.persistence.Table;

@Entity
@Table(name = "user", schema = "itcompass")

public class User {
    //definice polí třídy
}
```

Obrázek 28: popis vzniku tabulky

11. Testování

Tato kapitola popisuje testování aplikace včetně testovací strategie a konkrétních testů. Testování je rozděleno do dvou částí: první část probíhá současně s implementací a je prováděna vývojářem, zatímco druhá část je předána potenciálním koncovým uživatelům, kteří budou aplikaci používat.

11.1. Vývojářské testování

V této podkapitole se budeme zabývat testy, které byly prováděny paralelně s vývojem aplikace vývojářem. Vývojářské testování si ještě dále rozdělíme na dvě dílčí rozdílné skupiny testů. První částí je testování backendové části a druhou částí je testování frontendu.

11.1.1. Backendová část

Testy na backendové straně ověřují, zda je serverová část připravena na požadavky, které jsou zadávány pomocí frontendu. V rámci vývoje se nejprve provádí takzvané Unit testy. Pokud všechny dílčí Unit testy procházejí s očekávanými hodnotami, lze následně přejít na testy integrační. Tyto testy ověřují, zda různé části systému spolu správně komunikují. Zahrnují testování interakcí mezi jednotlivými moduly a také komunikaci s externími závislostmi, jako jsou databáze nebo externí služby. Cílem integračních testů je zajistit, aby všechny komponenty systému fungovaly správně dohromady a aby data plynule procházela celým systémem bez chyb. Díky těmto testům se zajišťuje, že backendová část aplikace je robustní, spolehlivá a připravená na nasazení do produkčního prostředí. Všechny testovací fáze - od unit testů přes integrační testy až po výkonnostní testy - jsou nezbytné pro zajištění vysoké kvality a stability aplikace, což je klíčové pro její úspěšné použití v reálném světě.

Unit testy

Unit testy se zaměřují na nejmenší testovatelné části aplikace, kterými jsou jednotlivé metody. Každá metoda je testována na různé vstupy, aby bylo zajištěno, že vrací správná data a provádí správné operace. Jednotkové testování vyžaduje pečlivou přípravu všech možných vstupů a následnou kontrolu výstupů. Pro tento účel se často používá framework JUnit, který poskytuje strukturované prostředí pro psaní a spouštění unit testů v aplikacích napsaných v jazyce Java. Unit testy by měli splňovat jmenné konvence, název by měl obsahovat název metody, která je testována, očekávaný vstup a očekávaný výstup testu. Struktura testu se zpravidla skládá ze tří částí, kterým se zkráceně říká AAA(Arrange, Act, Assert)

11.1.2. Frontendová část

Frontendové testování ověřuje, zda uživatelské rozhraní aplikace funguje správně a poskytuje očekávaný uživatelský zážitek. Testy jsou prováděny vývojářem během vývoje aplikace a zahrnují různé typy testů, které se zaměřují na různé aspekty frontendového vývoje.

Testování kompatibility prohlížečů

Vzhledem k tomu, že různé webové prohlížeče mohou vykreslovat komponenty odlišně, je nezbytné otestovat chování jednotlivých komponent napříč různými prohlížeči. Tento krok zajišťuje, že aplikace bude správně zobrazena bez ohledu na použitý prohlížeč. Testování bylo provedeno v nejrozšířenějších prohlížečích, jako jsou Safari, Chrome a Firefox. Zvláštní pozornost byla věnována posuvníkům a dalším interaktivním prvkům, které během implementace způsobovaly problémy.

Testování přístupnosti

Testy přístupnosti byly provedeny za účelem ověření, že uživatelé se nikdy nezaseknou na jedné obrazovce a že všechny tlačítka a odkazy fungují podle očekávání. Tento test zahrnoval kontrolu správného přesměrování uživatele po přihlášení a odhlášení a zajištění, že všechny interaktivní prvky jsou přístupné a použitelné. Testování přístupnosti je klíčové pro zajištění, že aplikace je snadno použitelná pro všechny uživatele, včetně těch s různými druhy postižení.

Testy uživatelského rozhraní

Testy uživatelského rozhraní (UI) se zaměřují na vizuální a funkční správnost jednotlivých komponent. Tyto testy zahrnují ověření, že všechny komponenty jsou správně vykresleny, mají správné rozvržení a reagují správně na uživatelské akce. Testování UI bylo prováděno pomocí nástrojů jako Selenium, které umožňují simulovat uživatelské interakce a ověřovat, že aplikace reaguje správně.

11.2. Uživatelské testování

V této podkapitole budeme popisovat průběh uživatelského testování aplikace. Uživatelského testování se účastnili stejní jedinci, kteří byli mými respondenty při formulaci kritérií studijních oborů. Pro účely testování byl definován testovací scénář, který je popsán níže. Mimo testy samotné funkcionality vyvíjeného nástroje bylo součástí uživatelského testování i poskytnutí zpětné vazby k výsledkům dotazníkové části podpůrného nástroje.

11.2.1. Testovací scénář

Dotazník

- Uživatelé vyplnili dotazník, kde odpověděli na otázky týkající se jejich preference a kritérií pro výběr studijního oboru.
- Po vyplnění dotazníku si uživatelé zobrazili výsledky, které jim byly prezentovány na základě jejich odpovědí

Zpětná vazba

- Uživatelé poskytli zpětnou vazbu na výsledky dotazníkové části a celkovou použitelnost a smysluplnost nástroje.
- Diskutovali jsme o jejich dojmech z aplikace, co se jim líbilo, co by zlepšili, zda našli nějaké chyby nebo nedostatky, zda je aplikace užitečná a zda výsledky odpovídají jejich představám.

11.2.2. Průběh testování

Uživatelé obdrželi dvě sady instrukcí pro testování aplikace. První sada instrukcí vyzývala uživatele k samostatnému vyplnění dotazníku bez jakýchkoliv pokynů od vývojáře. Tato část testování měla za cíl zjistit, jak intuitivní a uživatelsky přívětivé je rozhraní aplikace. Uživatelé vyplňovali dotazník na základě vlastního porozumění a bez vnějších intervencí.

Ve druhé iteraci testování byly uživatelům poskytnuty dodatečné instrukce, které měly napomoci přesnějšímu a konzistentnímu odpovídání na otázky. Tyto instrukce obsahovaly specifické pokyny, jak odpovídat na jednotlivé otázky, aby výsledky byly co nejpřesnější a nejrelevantnější. Tento krok měl ověřit, zda aplikace dokáže poskytnout užitečné výsledky i bez rozsáhlého vysvětlení a jaký vliv mají dodatečné pokyny na kvalitu výsledků.

11.2.3. Zpracování zpětné vazby

Zpětná vazba od uživatelů byla velmi pozitivní. Nástroj fungoval podle očekávání všech zúčastněných, a žádné chyby ve funkcionalitě nebyly identifikovány. Několik uživatelů mělo drobné připomínky k vizuálnímu zpracování aplikace, což jsme vzali jako podnět pro budoucí vylepšení uživatelského zážitku. Kritéria, na která se dotazník ptal, byla vnímána jako velmi rozsáhlá a žádný z respondentů nenaznačil, že by chybělo některé z klíčových kritérií pro rozhodování o studijním oboru.

Většina uživatelských testerů hodnotila aplikaci jako „užitečný podpůrný nástroj pro získání více informací pro jejich rozhodovací situaci“. Zpětná vazba ukázala, že aplikace je efektivní v poskytování relevantních informací a usnadňuje rozhodovací proces uživatelů.

Co se týče rozdílů mezi vyplňováním dotazníku bez dodatečných instrukcí a s nimi, rozdíly byly minimální. Tento fakt potvrzuje, že uživatelské rozhraní aplikace je intuitivní a snadno použitelné i bez podrobných instrukcí. To je důležitým indikátorem, že aplikace je dobře navržena a uživatelé se v ní snadno orientují.

Celkově lze říci, že uživatelské testování poskytlo cenné informace o použitelnosti a efektivitě aplikace, které budou využity pro další vylepšování a optimalizaci nástroje.

\

11.3. Shrnutí testování

V této kapitole jsme podrobně popsali proces testování vyvíjené aplikace, která slouží jako podpůrný nástroj pro výběr studijního IT oboru. Testování bylo rozděleno na dvě hlavní části: vývojářské testování a uživatelské testování.

V rámci vývojářského testování byly prováděny unit testy, integrační testy a výkonnostní testy. Unit testy se zaměřily na jednotlivé metody a ověřovaly jejich správnou funkčnost na základě různých vstupů. Integrační testy pak kontrolovaly správnou komunikaci mezi jednotlivými částmi systému a zajišťovaly, že data procházejí systémem bez chyb.

Uživatelské testování bylo klíčové pro ověření celkové použitelnosti a smysluplnosti aplikace. Respondenti, kteří se podíleli na formulaci kritérií studijních oborů, se účastnili také uživatelského testování. Testování probíhalo podle předem definovaného scénáře, který zahrnoval vyplnění dotazníku a poskytnutí zpětné vazby. Uživatelé hodnotili aplikaci pozitivně, přičemž žádné zásadní chyby ve funkcionalitě nebyly nalezeny. Některé drobné výtky k vizuálnímu zpracování byly zaznamenány a budou sloužit jako podnět pro budoucí zlepšení.

Zpětná vazba od uživatelů potvrdila, že aplikace je užitečným podpůrným nástrojem, který intuitivně a srozumitelně vede uživatele procesem výběru studijního oboru. Rozdíly mezi vyplněním dotazníku bez dodatečných instrukcí a s instrukcemi byly minimální, což svědčí o kvalitním a uživatelsky přívětivém designu aplikace. Celkově lze říci, že testování aplikace prokázalo její stabilitu, spolehlivost a užitečnost. Aplikace byla schopna splnit očekávání

uživatelů a poskytla jim cenné informace pro jejich rozhodovací proces. Výsledky testování potvrzují, že aplikace je připravena k nasazení a použití v reálném světě.

12. Aktuální stav aplikace

Tato kapitola poskytuje přehled o současném stavu aplikace. Popisuje, jak aplikaci nainstalovat a spustit od stažení až po úspěšné spuštění. Dále se budeme zabývat problematikou udržitelnosti objektivního hodnocení studijních oborů a v neposlední řadě budeme vést diskuzi nad budoucím rozšířením aplikace.

12.1. Provoz a instalace

Tato část kapitoly se detailně zaměřuje na proces instalace a spuštění aplikace. Aplikace je vytvořena jako projekt v IntelliJ IDEA, napsaný v Java SpringBoot pro backend a Angular pro frontend, jak je známo z kapitoly 9 této bakalářské práce.

1) Klonování repozitáře: Nejprve si naklonujte projetk z GIT repozitáře, který je přiložen v textovém souvoru odkazy.txt v příloze bakalářské práce.

   Použijte příkaz git clone <URL-repozitáře>
2) Nastavení backendu:
   a. Otevřete IntelliJ IDEA a načtěte klonovaný projekt. Ujistěte se, že máte nainstalovanou Javu (verze 11 a novější) a Maven.
   b. V IntelliJ IDEA otevřete soubor „application.properties“, který se nachází v src/main/resources. Upravte konfiguraci připojení k databázi dle vašich údajů
   c. Pro inicializaci databáze použijte přiložený skript s názvem „ITCompass.sql“ z git repozitáře a spusťte ho v PostgreSQL
   d. Po konfigurace spusťte backend pomocí IntelliJ IDEA, to lze provést buď tlačítkem „run“ v horním pravém rohu IntelliJ, nebo použitím příkazu v terminálu „mvn spring-boot:run“
3) Nastavení frontendu
   a. Ujistěte se, že máte nainstalovaný Node.js a Angular CLI.
   b. Přejděte do adresáře projektu s frontendem, který se jmenuje „Frontend“ příkazem „cd frontend“
   c. Nainstalujte potřebné závislosti pomocí příkazu „npm run“
   d. Spusťte aplikaci pomocí příkazu „ng serve“
   e. Aplikace bude dostupná na adrese http://localhost:4200

12.2. Udržitelnost hodnocení studijních oborů

Jedním z problémů projektu je zajištění dlouhodobé udržitelnosti aplikace. Hodnocení jednotlivých studijních oborů na základě stanovených kritérií bylo provedeno manuálně a je založeno na aktuálních akreditacích těchto programů. Vzhledem k tomu, že akreditace studijních oborů se mohou v průběhu let měnit, může dojít k nesouladu mezi aktuálními studijními plány a těmi, které byly hodnoceny v rámci této práce. Tento faktor může vést k nepřesnostem v doporučeních aplikace, pokud nebudou data pravidelně aktualizována.Možná právě faktor dlouhodobé udržitelnosti hodnocení oborů je důvodem pro jednoduchost již existujících podpůrných nástrojů pro volbu studijního IT oboru.

12.3. Možnosti dalšího rozvoje aplikace

Tato podkapitola podrobněji rozebírá směry, kterými se potenciální rozvoj aplikace může dále rozrůstat.

1. Rozšíření o jiné obory

   Aplikace by mohla být v budoucnu rozšířena o další typy studijních oborů, než pouze IT studijní obory. Toto rozšíření by umožnilo širší využití aplikace mezi studenty s rozdílnými zájmy a akademickým zaměřením.

2. Rozšíření o zahraniční univerzity

   Současná aplikace je vyhrazena pouze pro univerzity na území České republiky. Potenciální rozšíření by mohlo rozšířit základnu uživatelů o cizojazyčné uživatele a poskytnout celkově širší portfolio studijních oborů na výběr.

3. Rozšíření o magisterské obory

Kromě bakalářských oborů by aplikace mohla v budoucnu zahrnovat také magisterské studijní programy. Toto rozšíření by umožnilo uživatelům pokračovat v používání aplikace i po nástupu k bakalářskému studiu. Tímto bychom usnadnili rozhodovací situaci i úspěšným absolventům bakalářského studia při plánování dalšího kroku svého vysokoškolského vzdělání.

4. Spolupráce s univerzitami

Aplikace by mohla navázat přímou spolupráci s univerzitami. To by mohlo zahrnovat získávání aktuálních informací o nově akreditovaných studijních programech a doplnění informací o jakýchkoliv změnách na již evidovaných programech. Tento krok by zásadně pomohl přesnosti a objektivitě prezentovaných výsledků této aplikace.

5. Rozšíření sady otázek

Další možností rozvoje aplikace je rozšíření sady otázek, které jsou kladeny uživatelům během dotazníkové části. Tím bychom mohli lépe pochopit preference a potřeby uživatelů a poskytnout jim přesnější a personalizovanější doporučení studijních oborů.

6. Implementace AI chat modelu

Dalším inovativním krokem by mohlo být zvážení nasazení AI Chat modelu, který by byl trénovaný na specifických datech relevantních pro volbu studijních oborů. Tento model by mohl nahradit tradiční dotazníkovou část aplikace, čímž by se snížila složitost údržby a aktualizace otázek. AI model by umožnil interaktivní komunikaci s uživateli, což by mohlo zlepšit jejich zkušenost a poskytovat dynamická a aktuální doporučení.

12.4. Shrnutí

V této kapitole jsme se zaměřili na aktuální stav aplikace, její instalaci, a spuštění. Popsali jsme proces, jak klonovat repozitář, nastavit backend a frontend, a inicializovat databázi. Dále jsme diskutovali o problémech dlouhodobé udržitelnosti hodnocení studijních oborů a navrhli několik možností dalšího rozvoje aplikace, včetně rozšíření o další obory, zahraniční univerzity, magisterské obory, spolupráce s univerzitami, rozšíření sady otázek a implementace AI chat modelu.

Tato kapitola uzavírá technickou část naší práce a plynule nás přivádí k závěrečné kapitole o vyhodnocení výstupů práce, kde shrneme dosažené výsledky a přínosy tohoto projektu.

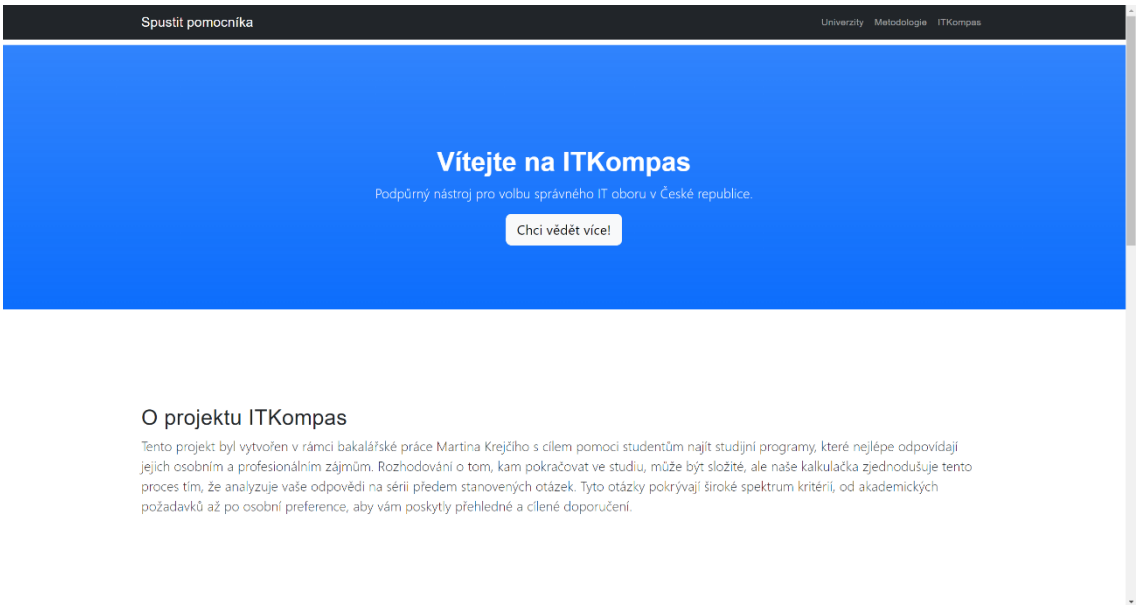
13. Vyhodnocení výstupů práce

13.1. Vytvořená aplikace

Jedním z výstupů této bakalářské práce je aplikace, sloužící jako podpůrný nástroj pro výběr studijního IT oboru v České republice

13.1.1. Domovská stránka

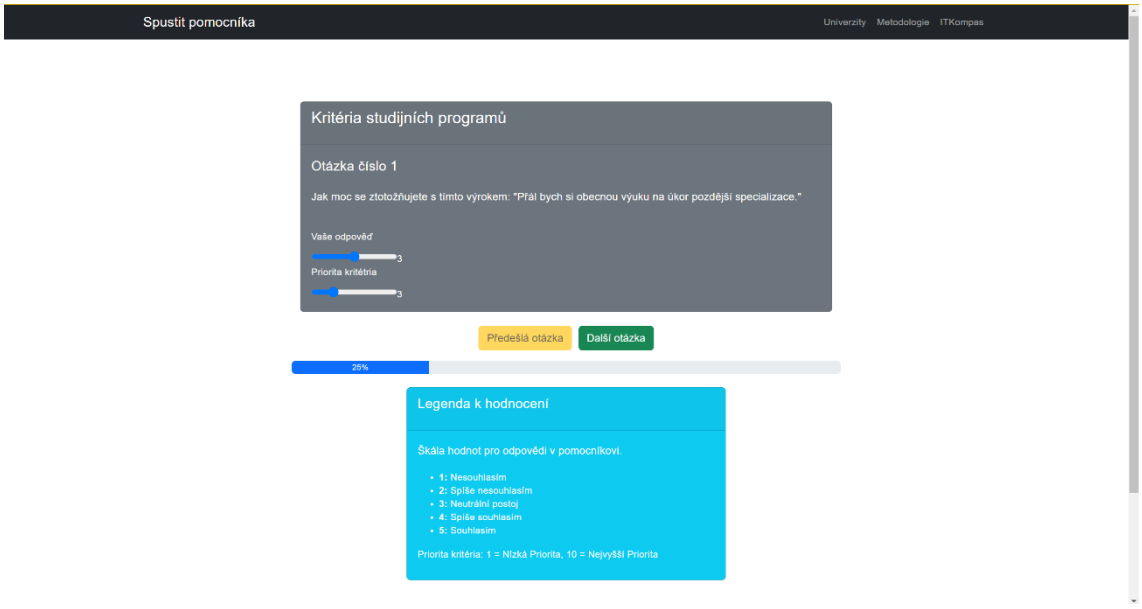
Domovská stránka aplikace poskytuje uživateli základní informace o aplikaci a jejím účelu. Obsahuje úvodní text, který vysvětluje, jak aplikace funguje, a jak může pomoci při výběru vhodného IT oboru. Na domovské stránce jsou take popsány všechny zbylé sekce aplikace a k čemu slouží. Na domovské stránce se take nachází navigační menu, které umožňuje uživateli přístup k zmíněným sekcím aplikace, jako je dotazník, výsledky, university a metodologie.



Obrázek 29: Ukázka domovské stránky

13.1.2. Dotazník

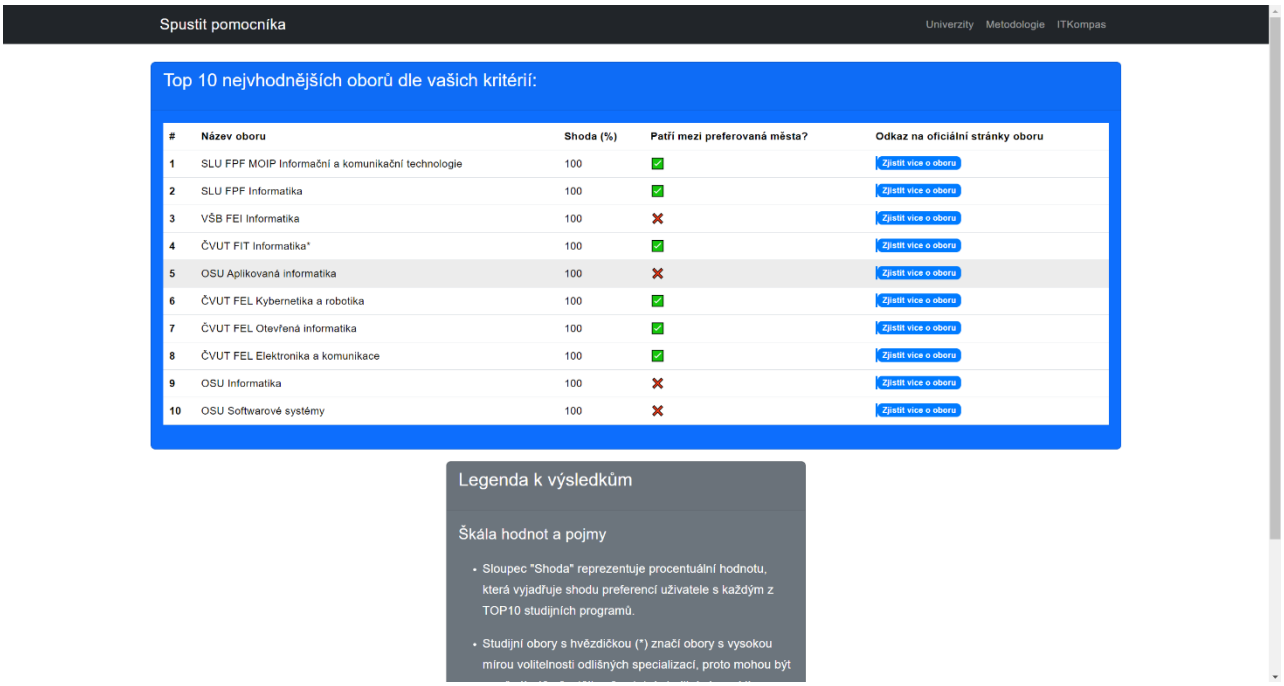
Dotazníková část aplikace je klíčovou komponentou, která shromažďuje informace o preferencích uživatele. Dotazník se skládá ze sady otázek, z nichž každá odpovídá různým kritériím, která jsou důležitá pro rozhodovací proces při výběru studijního IT oboru. Uživatelé odpovídají na otázky a určují prioritu jednotlivých kritérií dle osobních preferencí. Tyto preference a priority jsou následně použity pro výpočet nejvhodnějších studijních oborů.



Obrázek 30: Ukázka kladené otázky

13.1.3. Výsledky

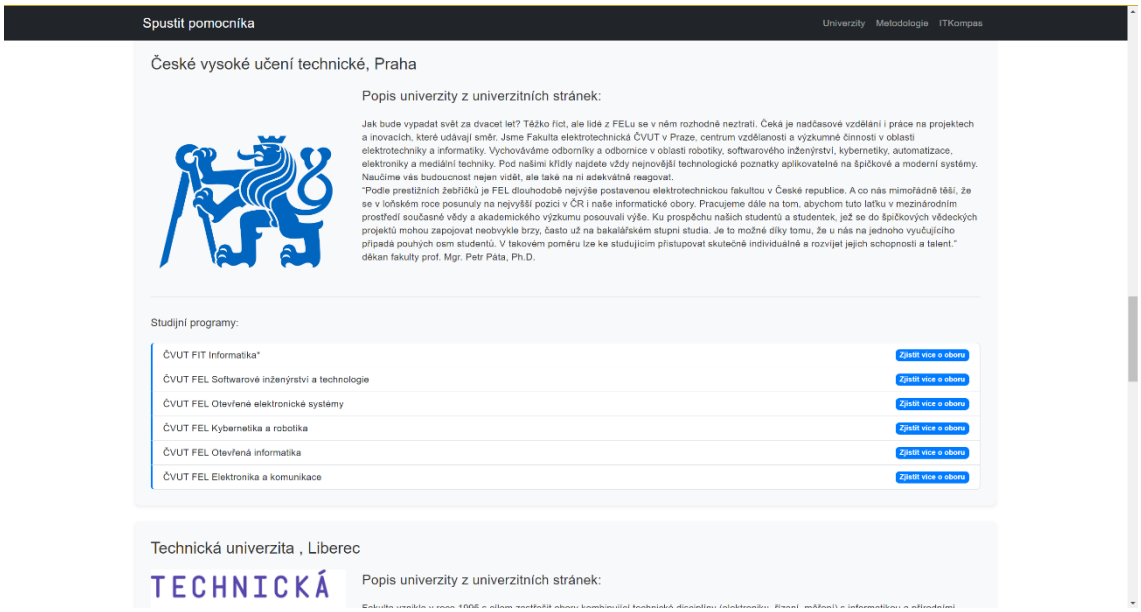
Po vyplnění dotazníku jsou uživatelům prezentovány výsledky, které zahrnují seznam nejvhodnějších studijních oborů podle jejich preferencí. Výsledky jsou zobrazeny ve formě TOP10 nejlepších kandidátů, kde každý obor je hodnocen na základě zadaných kritérií a jejich váhy. Kromě samotných studijních oborů jsou ve výsledcích zohledněny i preference města, kde se nachází univerzita, na které studijní obor figuruje.



Obrázek 31: Demonstrace výsledků

13.1.4. Univerzity

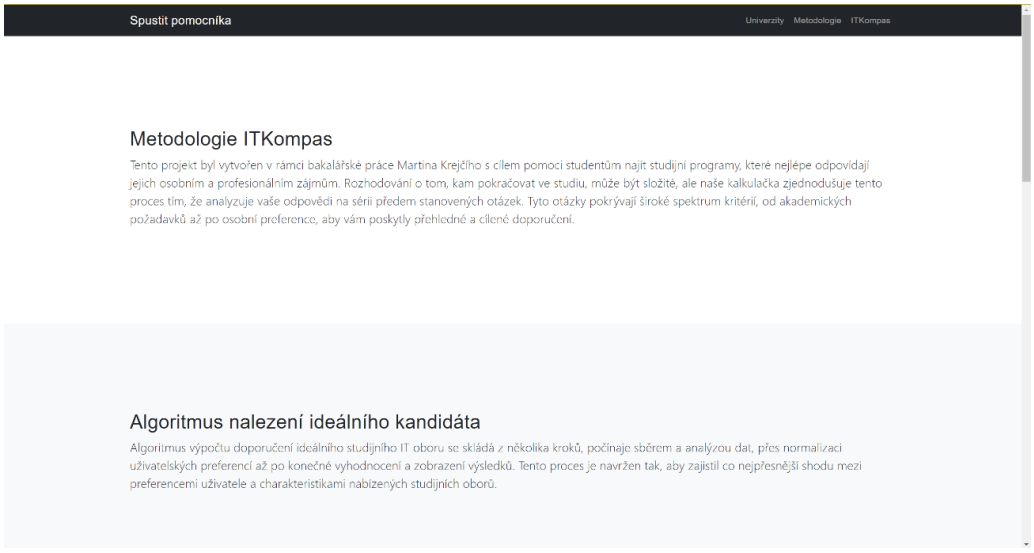
V sekci 'Univerzity' najdou uživatelé seznam všech univerzit nabízejících studijní programy v oblasti IT. Tato sekce poskytuje základní informace o jednotlivých univerzitách a odkazy na oficiální stránky jednotlivých oborů.



Obrázek 32: Ukázka prohlížení univerzit

13.1.5. Metodologie

Sekce metodologie aplikace popisuje celý algoritmus výpočtu a process sběru dat pro objektivní a přesné výsledky dotazníku. Jedná se o statickou stránku poskytující podrobný vhled do systému.



Obrázek 33: Ukázka metodologie

13.2. Vyhodnocení cílů práce

V této kapitole se zaměříme na vyhodnocení cílů stanovených na začátku této bakalářské práce. Každý cíl bude rozebrán a bude uvedeno, jakým způsobem byl splněn.

13.2.1. Definice základních pojmů

Prvním cílem bylo definovat základní pojmy, jako jsou vysoká škola, univerzita, technický obor, informatický obor a další. Tohoto cíle bylo dosaženo v kapitole 2, kde byly podrobně vysvětleny všechny klíčové pojmy nezbytné pro pochopení následujícího textu. Tato část poskytla čtenáři jasný a srozumitelný základ, na kterém mohl stavět při dalším studiu práce.

13.2.2. Analýza existující nabídky studijních oborů

Druhým cílem bylo analyzovat existující nabídku studijních oborů zaměřených na studium informatiky v České republice. Tento cíl byl splněn v kapitole 3, kde byla provedena podrobná analýza dostupných studijních programů na různých veřejných vysokých školách v České republice. Analýza zahrnovala nejen přehled jednotlivých programů, ale také jejich porovnání na základě různých kritérií, jako jsou délka studia, zaměření a kvalita výuky.

13.2.3. Průzkum existujících podpůrných materiálů

Třetím cílem bylo provést průzkum existujících podpůrných materiálů, které slouží jako pomůcka s výběrem studia, a vyhodnotit je. Tento cíl byl dosažen v kapitole 5, kde byly analyzovány různé nástroje a materiály, jako jsou webové stránky vysokých škol, volební kalkulačky a jiné online zdroje. Byly identifikovány jejich silné a slabé stránky a na základě této analýzy byly navrženy doporučení pro zlepšení.

13.2.4. Návrh vlastního průvodce výběrem studijního oboru a školy

Čtvrtým cílem bylo navrhnout vlastního průvodce výběrem vhodného studijního oboru a školy, inspirovaného například volebními kalkulačkami. Tento cíl byl splněn v kapitole 7, kde byl podrobně popsán návrh aplikace, která zájemcům o studium informatických oborů pomůže s výběrem vhodné vysoké školy. Návrh zahrnoval strukturu aplikace, uživatelské rozhraní a klíčové funkce, které zajišťují efektivní a intuitivní používání aplikace.

13.2.5. Implementace průvodce

Pátým cílem bylo implementovat navrženého průvodce. Tento cíl byl splněn v kapitole 11, kde byla detailně popsána implementace aplikace. Byly použity moderní technologie pro vývoj frontendu i backendu, a také byla vytvořena databáze pro uchovávání potřebných dat.

Implementace zahrnovala všechny klíčové funkce navržené v předchozích kapitolách, včetně dotazníkové části, vyhodnocení odpovědí a generování doporučení.

13.2.6. Uživatelské testování a vyhodnocení

Šestým a posledním cílem bylo uživatelsky otestovat vytvořenou aplikaci a na vybrané cílové skupině vyhodnotit její použitelnost a především využitelnost. Tento cíl byl splněn v kapitole 12, kde bylo provedeno testování s reálnými uživateli. Testování zahrnovalo sběr zpětné vazby od budoucích a současných studentů vysokých škol, analýzu jejich zkušeností s aplikací a identifikaci oblastí pro zlepšení. Výsledky testování ukázaly, že aplikace je uživatelsky přívětivá a poskytuje relevantní a užitečné informace, které pomáhají při výběru studijního oboru a vysoké školy.

13.3. Závěr

Všechny stanovené cíle byly v rámci této bakalářské práce úspěšně splněny. Byly definovány klíčové pojmy, analyzována existující nabídka studijních oborů, proveden průzkum podpůrných materiálů, navržen a implementován průvodce výběrem studijního oboru a vysoké školy a aplikace byla uživatelsky testována. Výsledky této práce přinášejí cenný nástroj, který může výrazně usnadnit rozhodovací proces potenciálním studentům informatických oborů v České republice.

14. Závěr

Předmětem této bakalářské práce nebyla pouze implementace podpůrného nástroje pro volbu studijního IT programu. Implementace této aplikace vyplynula z analýzy současně dostupných podpůrných nástrojů, které neposkytují dostatečnou podporu pro volbu budoucích studentů bakalářského studia.V úvodní části této práce byly stanoveny hlavní cíle, mezi které patřilo vytvoření aplikace, která by sloužila jako efektivní nástroj pro výběr studijního IT oboru. Analýza klíčových pojmů a existujících podpůrných nástrojů poskytla důkladný přehled o stávajících možnostech a identifikovala mezery na trhu, které mohla naše aplikace vyplnit.

Metodika práce zahrnovala detailní přístup k filtraci vysokých škol na základě stanovených kritérií. Na základě tohoto přístupu byly vyvinuty algoritmy pro hodnocení univerzit a studijních programů podle uživatelských preferencí získaných prostřednictvím dotazníků.

V části týkající se návrhu aplikace jsme popsali aktéry, systémové požadavky a vytvořili diagramy tříd, použití a nasazení. Wireframes poskytly vizuální představu o uživatelském rozhraní aplikace. Pro sběr dat byly provedeny osobní rozhovory s potenciálními uživateli a analýza oficiálních stránek univerzit. Tato data byla kvantifikována a použita pro hodnocení jednotlivých studijních programů.nJedním z úskalí projektu je udržitelnost aplikace, jelikož hodnocení jednotlivých studijních oborů v kontextu kritérií bylo prováděno „ručně“ a pouze pro současné akreditace jednotlivých programů. Je možné že v následujících letech se akreditace studijních oborů změní. Tento fakt může vést k rozdílu mezi aktuálním studijním plánem a studijním plánem, který byl hodnocen v rámci této práce

Analýza technologií vedla k výběru Angularu pro frontend, Java pro backend a PostgreSQL pro databázi. Tato kombinace byla zvolena pro jejich robustnost, širokou komunitní podporu a vhodnost pro vývoj komplexních webových aplikací. Implementace aplikace zahrnovala vývoj backendové a frontendové části, připojení k databázi a nasazení aplikace. Testování bylo prováděno jak na vývojářské úrovni (unit a integrační testy), tak i za účasti uživatelů, kteří poskytli cennou zpětnou vazbu.

Výstupy této práce zahrnují plně funkční webovou aplikaci, která umožňuje uživatelům získat doporučení pro studijní IT programy na základě jejich osobních preferencí. Testování prokázalo, že aplikace je intuitivní, uživatelsky přívětivá a poskytuje přesné výsledky. Možnosti dalšího rozvoje aplikace zahrnují rozšíření na jiné obory, zahraniční univerzity, magisterské programy a spolupráci s univerzitami. Tyto rozšíření by mohly být předmětem budoucích diplomových prací a dále zvýšit užitečnost aplikace.

V závěru lze konstatovat, že cíle stanovené na začátku této práce byly úspěšně splněny. Vytvořená aplikace představuje významný krok směrem k usnadnění výběru studijního IT programu pro budoucí studenty a poskytuje základ pro další rozvoj a vylepšení.

15. Použitá literatura

[1] OPENAI. ChatGPT. Online. 2022, 2024. Dostupné z: https://chat.openai.com/. [cit. 2024-01-21].

[2] EDUROUTE S.R.O. VysokéŠkoly.cz. Online. EDUROUTE S.R.O. VysokéŠkoly.cz. 2024. Dostupné z: https://www.vysokeskoly.cz/. [cit. 2024-01-21].

[3] ČESKÁ REPUBLIKA. Zákon č. 111/1998 Sb.: o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách). In: Sbírka zákonů. 22. 4. 1998. 1998. ISSN 1211-1244.

[4] GEEKSFORGEEKS. *Frontend vs Backend* [online]. 2023, 2023-04-18, 2023-04-18 [cit. 2024-05-24]. Dostupné z: https://www.geeksforgeeks.org/frontend-vs-backend/

[5] EDUREKA!. Advantages and disadvantages of angular. *Advantages and disadvantages of angular* [online]. 2023, 2023-04-18, 2024-04-26 [cit. 2024-05-24]. Dostupné z: https://www.edureka.co/blog/advantages-and-disadvantages-of-angular/

[6] JAVATPOINT. Pros and cons of react. *Pros and cons of react* [online]. 2023, 2023-04-18 [cit. 2024-05-24]. Dostupné z: https://www.javatpoint.com/pros-and-cons-of-react

[7] NATURAILY. Pros and cons of Vue.js. *Pros and cons of Vue.js* [online]. 2023 [cit. 2024-05-24]. Dostupné z: https://naturaily.com/blog/pros-cons-vue-js

[8] AMAZON. The difference between frontend and backend. *The difference between frontend and backend* [online]. 2023 [cit. 2024-05-24]. Dostupné z: https://aws.amazon.com/compare/the-difference-between-frontend-and-backend/

[9] NETGURU. Pros and cons of Java. *Pros and cons of Java development in 2023* [online]. 2024, 2024-04-17 [cit. 2024-05-24]. Dostupné z: https://www.netguru.com/blog/java-pros-and-consEPAM. Node.js pros and cons. *Node.js pros and cons* [online]. 2024, 2024-04-17 [cit. 2024-05-24]. Dostupné z: https://anywhere.epam.com/en/blog/node-js-pros-and-cons

[10] GEEKSFORGEEKS. Advantages and Disadvantages of PHP. *Advantages and Disadvantages of PHP* [online]. 2022, 2022-06-20 [cit. 2024-05-24]. Dostupné z: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-php/

[11] ALOA. Relational vs. Non-Relational Database: Pros & Cons. ALOA. *Relational vs. Non-Relational Database: Pros & Cons* [online]. 2023 [cit. 2024-05-24]. Dostupné z: https://aloa.co/blog/relational-vs-non-relational-database-pros-cons

[12] DATABASETOWN. Relational Database Benefits and Limitations (Advantages & Disadvantages). DATABASETOWN. *Relational Database Benefits and Limitations (Advantages & Disadvantages)* [online]. 2023 [cit. 2024-05-24]. Dostupné z: https://databasetown.com/relational-database-benefits-and-limitations/

[13] TECHTARGET. DEFINITION relational database. TECHTARGET. *DEFINITION relational database* [online]. 2023 [cit. 2024-05-24]. Dostupné z: https://www.techtarget.com/searchdatamanagement/definition/relational-database

[14] COURSERA. Relational vs. Non-relational Database: The Difference Explained. [4] COURSERA. *Relational vs. Non-relational Database: The Difference Explained* [online]. 2023, 2023-12-6 [cit. 2024-05-24]. Dostupné z: https://www.coursera.org/articles/relational-vs-non-relational-database?utm_medium=sem&utm_source=gg&utm_campaign=B2C_EMEA__coursera_FTCOF_career-academy_pmax-multiple-audiences-country-multi-set2&campaignid=20882109092&adgroupid=&device=c&keyword=&matchtype=&network=x&devicemodel=&adposition=&creativeid=&hide_mobile_promo&gad_source=1&gclid=CjwKCAjw9cCyBhBzEiwAJTUWNUq6EVyEjjMF1z5jJBNpYnLsTW-pOEs4GW2Gy-tXm-zuEeGNk-HwsBoCe7sQAvD_BwE

[15] MŮČKA, Jan. MASTER. Relační databáze vs. nerelační databáze: jaké jsou mezi nimi rozdíly? MASTER. *Relační databáze vs. nerelační databáze: jaké jsou mezi nimi rozdíly?* [online]. 2023, 2021-05-27 [cit. 2024-05-24]. Dostupné z: https://www.master.cz/blog/relacni-databaze-nerelacni-databaze-jake-jsou-rozdily/

[16] ITNETWORK. Lekce 2 - UML - usecase diagram. ITNETWORK. *Lekce 2 - UML - usecase diagram* [online]. 2023, 2021-05-27 [cit. 2024-05-24]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-use-case-diagram

[17] GEEKSFORGEEKS. Functional vs Non Functional Requirements. GEEKSFORGEEKS. *Functional vs Non Functional Requirements* [online]. 2024, 2024-05-22 [cit. 2024-05-24]. Dostupné z: https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/

[18] ALEXSOFT. Onfunctional Requirements in Software Engineering: Examples, Types, Best Practices. ALEXSOFT. *Onfunctional Requirements in Software Engineering: Examples, Types, Best Practices* [online]. 2023, 2023-12-30 [cit. 2024-05-24]. Dostupné z: https://www.altexsoft.com/blog/non-functional-requirements/

[19] VISUAL PARADIGM. What is Class Diagram? VISUAL PARADIGM. *What is Class Diagram?* [online]. [cit. 2024-05-24]. Dostupné z: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/

[20] VISUAL PARADIGM. Use Case Diagrams | Unified Modeling Language (UML). VISUAL PARADIGM. *Use Case Diagrams | Unified Modeling Language (UML)* [online]. 2024, 2024-02-09 [cit. 2024-05-24]. Dostupné z: https://www.geeksforgeeks.org/use-case-diagram/

16. Seznam příloh

V zazipovaném souboru se nacházejí následující přílohy:

1)BakalářskáPráceKrejčí.pdf

2) Složka IT Kompas obsahující další složky:

Frontend implementované aplikace

Backend implementované aplikace

ITCompass.sql -> Skript nutný pro zprovoznění databáze

Odkaz.md -> textový soubor obsahující odkaz na GitHub projektu,