



**FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Aplikace pro přehlednou správu inzerátů nabídek pracovních pozic
Student:	Hynek Bulíř
Vedoucí:	Ing. Pavel Štěpán
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce zimního semestru 2019/20

Pokyny pro vypracování

Cílem práce je aplikace pro přehlednou správu inzerátů nabídek pracovních pozic, kde bude možno přehledně pracovat s nabídkami a exportovat výsledky s možností filtrace. Aplikace bude vyvinuta pro operační systém Windows, na platformě .NET. Součástí práce bude grafický design aplikace a uživatelský návod k použití.

Postupujte dle následujících kroků:

1. Analyzujte požadavky cílové skupiny uživatelů.
2. Navrhněte aplikaci vyhovující požadavkům.
3. Implementujte aplikaci a řádně ji otestujte .
4. Zhodnoťte výsledek a diskutujte další možná vylepšení.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 28. února 2018



**FAKULTA
INFORMAČNÍCH
TECHNOLÓGIÍ
ČVUT V PRAZE**

Bakalářská práce

Aplikace pro přehlednou správu inzerátů nabídek pracovních pozic

Hynek Bulíř

Katedra softwarového inženýrství
Vedoucí práce: Ing. Pavel Štěpán

8. ledna 2020

Poděkování

Chtěl bych poděkovat svému vedoucímu bakalářské práce Ing. Pavlu Štěpánovi za velice vstřícný a trpělivý přístup. Také bych rád poděkoval své mamince, že mě podporovala i v nelehkých chvílích při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 8. ledna 2020

.....

České vysoké učení technické v Praze
Fakulta informačních technologií

© 2020 Hynek Bulíř. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Bulíř, Hynek. *Aplikace pro přehlednou správu inzerátů nabídek pracovních pozic*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

Abstrakt

Tato bakalářská práce spočívá v analyzování potřeb, návrhu a vytvoření webové aplikace pro zjednodušení náborového řízení. Aplikace je určena pro personální oddělení firmy, která si sama spravuje nábor kandidátů, stará se o inzerování nabídek volných pracovních pozic a nevyužívá služeb personální agentury.

Cílem aplikace je pomoci udržet pořádek ve vlastních nabídkách práce firmy na inzertních webových stránkách a v kandidátech, kteří se na tyto nabídky hlásí.

Klíčová slova webová aplikace, C#, ASP.NET MVC, ATS, lidské zdroje (HR)

Abstract

This bachelor thesis deals with analysis of needs, design and development of a web application which should simplify process of recruiting new employees. The application is designed for Human Resources department of company that recruits new employees independently, i. e. without cooperation with recruitment agency.

Purpose of the application is to keep transparency and order in their own job offers which are being advertised at different websites; and also evidence of candidates applying for these offers.

Keywords web application, C#, ASP.NET MVC, ATS, human resources (HR)

Obsah

Úvod	1
1 Cíl práce	3
2 Analýza	5
2.1 Popis problematiky	5
2.2 Cílová skupina	5
2.3 Rešerše existujících řešení	6
2.4 Analýza požadavků	8
2.5 Případy užití	10
2.6 Doménový model	12
3 Návrh aplikace a zvolené technologie	15
3.1 Zvolené technologie	15
3.2 Architektura	19
3.3 Souhrn	20
4 Implementace aplikace	23
4.1 Vývoj	23
4.2 Souborová struktura	24
4.3 Ukázky kódu	26
4.4 Uživatelské rozhraní	27
5 Testování	29
5.1 Průběžné testování během vývoje	29
5.2 Uživatelské testování	29
5.3 Automatizované UI testování	30
6 Výsledek a další možná rozšíření	33
6.1 Výsledek	33

6.2 Možná rozšíření	33
Závěr	35
Literatura	37
A Uživatelský návod k použití	39
B Seznam použitých zkratk	45
C Obsah příloženého CD	47

Seznam obrázků

2.1	Diagram aktivity úpravy pracovní nabídky	11
2.2	Diagram případů užití související s pracovními nabídkami	11
2.3	Diagram případů užití související s kandidáty	12
2.4	Doménový model	13
3.1	MVVM Diagram - převzato z [1]	18
3.2	EF - Code First přístup - převzato z [2]	18
3.3	MVC Diagram	20
3.4	MVVM Diagram + jazyky jednotlivých vrstev	21
4.1	Adresářová struktura projektu	25
4.2	Notifikace úspěšného uložení	27
4.3	Notifikace neúspěšného načtení nabídek	27
4.4	Vyskakovací modal	28
A.1	Seznam pracovních nabídek	40
A.2	Přidání nové nabídky	41
A.3	Notifikace uložení záznamu	41
A.4	Notifikace smazání záznamu	41
A.5	Seznam kandidátů	42
A.6	Přidání nového kandidáta	42
A.7	Přidání nového komentáře kandidátovi	43

Úvod

Téměř každá firma musí nabírat nové zaměstnance, ať už kvůli odchodu aktuálních zaměstnanců, nebo z důvodu rozšiřování firmy. Pokud firma nevyužívá služeb personální agentury, o nábor nových zaměstnanců se většinou stará personální oddělení firmy. Toto oddělení má na starosti evidenci aktuálně otevřených pozic a kandidátů, kteří se na tyto pozice hlásí. Udržovat evidenci na papírových formulářích je dost nepřehledné a náročné na údržbu. Alternativou by mohly být oblíbené kancelářské programy jako Microsoft Excel¹, Google Sheets². Tyto nástroje však nejsou přímo na míru pro toto využití, a tak v nich může velice rychle vzniknout zmatek.

Moje práce se bude touto problematikou zabývat a výsledná aplikace by měla ušetřit spoustu zbytečné práce personálnímu oddělení. Téma „*Aplikace pro přehlednou správu inzerátů pracovních nabídek*“ jsem si zvolil na základě vyzorování nedostatků ve firmě, kde pracuji, a díky tomu jsem tam mohl nabrat dostatek informací potřebných k vytvoření této aplikace.

¹Dostupné na: <https://products.office.com/en/excel>

²Dostupné na: <https://www.google.com/intl/cs/sheets/about/>

Cíl práce

Cílem práce je webová aplikace na platformě .NET, která pomůže udržovat přehlednou správu inzerátů nabídek pracovních pozic. Aplikace bude evidovat nabízené pracovní pozice a k nim příslušné kandidáty.

Samotné implementaci aplikace předchází analýza, která spočívá ve srovnání současných řešení a zdokumentování požadavků personálního oddělení. Podle analýzy bude vytvořen návrh aplikace, následuje výběr a popis technologií, které budou k vývoji použity. Na závěr pak otestování a shrnutí výsledku.

Analýza

2.1 Popis problematiky

Personální oddělení menší až střední firmy potřebuje udělat nábor nových kandidátů na otevřené pracovní pozice a zároveň si nějakým způsobem udržovat přehled o již nabídnutých pozicích a kandidátech, kteří se na tyto pozice hlásí. Existuje mnoho způsobů, kam nabídku inzerovat, ať už papírovou formou, například na nástěnce ve škole, nebo elektronickou formou, (kterou se zde budeme zabývat), na inzertních portálech, jako jsou jobs.cz³, prace.cz⁴, LinkedIn⁵ a mnoho dalších.

K jednotlivým nabídkám je dobré si udržovat i drobnou statistiku úspěšnosti, aby se tak firma vyhnula inzerování na místa, kde je úspěšnost příliš malá. To ušetří firmě peníze a personálnímu oddělení práci a čas. Při výběru kandidátů je vhodné si od prvního pohovoru, někdy jednoho z mnoha, udržovat jejich hodnocení. A zaměstnancům, kteří s nimi přišli do styku u tohoto pohovoru, nebo v předchozím zaměstnání, dát možnost je slovně okomentovat, což se později projeví u finálního výběrů z těchto kandidátů.

2.2 Cílová skupina

Aplikace má být zatím zaměřena pouze pro personální oddělení firmy, která spravuje v řádech desítky pracovních nabídek a kandidátů. Aplikace je cílena tak, aby k jejímu užívání nebylo potřeba proškolení zaměstnanců a aby k ní stačily základní dovednosti na práci s PC.

³Dostupné na: <https://www.jobs.cz>

⁴Dostupné na: <https://www.prace.cz>

⁵Dostupné na: <https://www.linkedin.com>

2.3 Rešerše existujících řešení

2.3.1 Papírová forma

Asi nejjednodušší možné řešení je psaní evidence nabízených inzerátů na papírový formulář. Ve firmě, která je opravdu malá, by toto řešení pravděpodobně obstálo, ale u firmy nabízející současně několik desítek pracovních pozic by to bylo opravdu složité na údržbu a zároveň velice nepřehledné.

2.3.2 Kancelářský software

V kancelářských balících (Microsoft Office, LibreOffice a další) se již dá práce značně usnadnit a zajistit lepší přehlednost. Sdílení těchto záznamů je také podstatně snazší jejich ukládáním na místa přístupná v síti, nebo prostřednictvím elektronické pošty.

Avšak stále u těchto nástrojů chybí možnost přistupovat do záznamů více uživateli najednou. Tuto možnost nám dovolí například online nástroj Google Sheets⁶, který obsahuje více užitečných funkcí pro tento účel, jako chat mezi uživateli, komentáře, atd. Bohužel ale nedokáže zajistit nastavené formátování, a tak i v takovém nástroji může vzniknout zmatek.

2.3.3 Aplikace

Jednoznačně nejlepším současným řešením je specializovaná aplikace, kde se nestane, že by se špatnou manipulací se záznamy nějaké ztratily, jak tomu může být u obyčejného kancelářského softwaru. Ve většině případů aplikace umožňuje i analýzu dat, a může tak usnadnit a zefektivnit celý proces nabírání pracovníků. Zároveň se ale jedná o jednoznačně nejnákladnější variantu.

Tyto aplikace se nazývají ATS, neboli Applicant tracking system, volně přeloženo Systém pro sledování žadatelů. Těchto aplikací existuje mnoho, jen se většinou liší funkcionalitou a hlavně cenou.

2.3.3.1 Greenhouse

Aplikaci Greenhouse⁷ používají velké firmy, například Booking.com⁸, Airbnb⁹. Umí vyhodnotit výkon procesu najímání, umožňuje analyzovat nabídky a jejich úspěšnost, porovnávat kandidáty, organizovat pohovory s kandidáty. Aplikace Greenhouse je uzpůsobená použití jak na počítači, tak na mobilním zařízení, kterým lze například vyfotit životopis a a rovnou ho zdigitalizovat. Aplikace zjednoduší a urychlí práci v různých ohledech a umožní vyhnout se

⁶Dostupné na: <https://www.google.com/intl/cs/sheets/about/>

⁷Dostupné na: <https://www.greenhouse.io/>

⁸Dostupné na: <https://www.booking.com>

⁹Dostupné na: <https://www.airbnb.cz/>

papírování. Aplikace je ovšem zpoplatněná, platí se měsíční paušální poplatek, jehož výše závisí na velikosti firmy.

2.3.3.2 Recooty

Aplikace Recooty¹⁰ taktéž zaznamenává a analyzuje nábor od samého začátku a to od vložení inzerátu na web až po přijetí nového zaměstnance. Umožňuje zpracování statistik za účelem optimalizace celého procesu nábora zaměstnanců. Oproti předešlé aplikaci se ale Recooty liší značným rozdílem ceny, ta totiž začíná zhruba na desetině ceny aplikace Greenhouse.

2.3.3.3 BreezyHR

Aplikace BreezyHR¹¹ je na první pohled jasná a přehledná. Chlubí se navíc funkcí drag & drop při třídění kandidátů, neboli že záznam kandidáta lze jednoduše přesunout posunem myši do příslušné kategorie nebo jím změnit fázi nábora. Zaujala mě zde možnost přidávání komentářů k jednotlivým kandidátům. U ostatních aplikací se většinou jen vyplňuje tabulkové hodnocení. Cena této aplikace je podobná jako u aplikace Greenhouse, tedy jedna z těch dražších.

2.3.3.4 Vlastní webová aplikace

Webová aplikace vytvořena na míru potřebám personálního oddělení přináší největší výhodu, protože je nastavená přesně podle potřeb dané firmy. Hlavní nevýhodou je však její vysoká pořizovací cena. Hlavně menší firmy, kde ani není tak velká fluktuace zaměstnanců, si takové náklady navíc většinou rychle rozmyslí a zůstanou u svého zaběhlého systému, případně využijí pomoc personální agentury. Pokud firma disponuje svým vlastním vývojářským oddělením, může mu vytvoření takové aplikace zadat. Velkou výhodou pak bude znalost procesů firmy a požadavků na výběr kandidátů. Samotní zaměstnanci personálního oddělení pak mohou být využiti jako živí testeři. Vzhledem k tomu, že firma se na vývoj takové aplikace nespécializuje, je pravděpodobné, že za jejího prvního provozu bude nacházet ještě další provozní chyby a budou potřeba nějaké úpravy. Ty mohou souviset i s tím, že se budou postupně objevovat další náměty, které by taková aplikace mohla umět zpracovat. Samozřejmě další možností je objednat si vlastní aplikaci u dodavatele softwaru, což jak už bylo řečeno, je značně nákladné. Mimo to se musí počítat se složitější komunikací a delší odezvou v případě řešení pozdějších úprav a chyb. Na druhou stranu, specializovaná firma by měla dodat hotový nástroj, kde bude chybivost zanedbatelná. Kromě prvotních nákladů se však musí počítat i s dalšími

¹⁰Dostupné na: <https://recooty.com/>

¹¹Dostupné na: <https://breezy.hr/>

poplatky za spravování systému podle SLA ¹², tedy na základě smlouvy mezi uživatelem služby a jejím poskytovatelem.

2.4 Analýza požadavků

Vycházel jsem zde ze zkušeností ze svého zaměstnání ve firmě Commerz Systems GmbH, kde jsem působil v týmu programátorů, který měl za úkol vytvořit podobnou aplikaci pro personální oddělení, ale bohužel k vývoji této aplikace nakonec nikdy nedošlo.

Požadavky byly velice jednoznačné a to ukládání záznamů nabídek s popisem, kde jednotlivé záznamy obsahují datum od kdy a do kdy jsou nabídky inzerovány. Dále na jakém portálu jsou umístěny s možností uchovávání informací, kolik lidí na nabídku zareagovalo, kolik jich bylo pozváno k pohovoru a kolik jich ve výsledku bylo zaměstnáno. Aplikace by též uchovávala záznamy o jednotlivých kandidátech, a umožňovala tak jednodušší selekci potencionálních zaměstnanců. Ke kandidátům by se daly přidávat komentáře s hodnocením, aby se usnadnil jejich výběr. Díky informacím o úspěšnosti jednotlivých nabídek se personální oddělení bude moci soustředit spíše na internetové stránky, na kterých mají větší úspěšnost s náborem pracovníků. V neposlední řadě, ač to může znít legračně, je důležitý vzhled aplikace. V dnešní době si každý oblíbí raději aplikaci s efektními animacemi a barevným designem, než obyčejnou formulářovou aplikaci se strohým vzhledem. Měla by mít poutavý vzhled, ale neztratit tím na přehlednosti.

Aplikace by měla být lokalizovaná pro konkrétní zemi, aby bylo zamezeno pozdějšímu složitějšímu refaktorování. Zobrazující se výrazy v daném jazyce budou soustředěny do jednoho lokalizačního souboru, který stačí pouze vyměnit za jiný, a celá aplikace bude v jiném jazyce. Toto je z důvodu zachování přehlednějšího kódu kvůli případnému budoucímu rozšiřování. Díky soustředění výrazů do jednoho souboru[3] znamená překlad kompletní aplikace jen několik desítek minut práce.

A v poslední řadě k aplikaci se bude přistupovat přes tzv. tenkého klienta, kterého bude představovat webový prohlížeč. Tím se zjednoduší přenositelnost a nebude potřebná jakákoliv instalace na straně klienta, jelikož aplikace poběží na jednom serveru.

2.4.1 Požadavky

Tato sekce se věnuje funkčním požadavkům a nefunkčním požadavkům a popisuje, jak by se měl výsledný produkt chovat a jaké procesy užití má splňovat. Funkční požadavky určují, co musí aplikace splňovat a co umožní svým uživatelům. Naopak nefunkční požadavky jsou omezující vlastnosti, které musí systém, na kterém aplikace poběží, splňovat.

¹²Service-level agreement

2.4.1.1 Funkční požadavky

Vytvoření pracovní nabídky Uživatel vytvoří nabídku v aplikaci, ta se uloží na server, a bude tak i pro všechny ostatní uživatele přístupná. Pracovní nabídka musí obsahovat informace jako název, popis, portál, na kterém je vystavena, datum platnosti nabídky, počet kandidátů, kteří firmu kontaktovali, počet kandidátů pozvaných na pohovor a počet přijatých, tedy statistické údaje. Záznam pracovní nabídky by měl zobrazovat informace o datu vytvoření a datu poslední úpravy.

Správa uložených nabídek Uživatelé budou moci upravovat a mazat uložené nabídky, opět se změna promítne na server, a bude tak viditelná pro všechny uživatele.

Filtrování nabídek Aplikace uživateli umožní filtrovat uložené pracovní nabídky podle názvu a popisu.

Export nabídek Uživatel si bude moci všechny nabídky, nebo pouze ty vyfiltrované, exportovat do souboru.

Vytvoření kandidáta Uživatel vytvoří kandidáta, kterému bude přiřazena pracovní nabídka místa, na kterou se hlásí. Tento kandidát bude aplikací uložen na server a bude přístupný i pro ostatní uživatele.

Správa kandidátů Jednotlivé kandidáty bude možno uživateli upravovat a mazat. Změny se opět musí projevit u všech uživatelů.

Komentování a hodnocení kandidátů Uživatelé budou moci jednotlivé kandidáty komentovat a hodnotit, na základě čehož bude vypočítáno průměrné hodnocení kandidáta ke srovnání s ostatními.

2.4.1.2 Nefunkční požadavky

Platforma .NET Aplikace bude na platformě ASP.NET MVC 5

Aplikace na vnitřní síti Aplikace bude přístupna pouze z vnitřní sítě firmy a pomocí restrikce adres v síti budou mít přístup pouze oprávnění uživatelé.

Tenký klient Aplikace bude tzv. Tenký klient, tedy nebude kromě prohlížeče vyžadovat žádné další nástroje na straně uživatele.

Verze prohlížeče na straně klienta U klienta bude aplikace správně pracovat na prohlížeči Microsoft Edge verze minimálně 44.18362.267.0 a prohlížeči Google Chrome verze minimálně 77.0.3865.90.

Jazyková lokalizace Aplikace bude snadno lokalizovatelná pomocí jednoho souboru s příslušným jazykem. Výměna tohoto souboru nebude vyžadovat znovu sestavování celé aplikace.

2.5 Případy užití

2.5.1 UC1 - Vytvoření nabídky

Uživatel chce přidat novou pracovní nabídku. Jde do panelu pracovních nabídek, vybere možnost přidání nové nabídky, následně zadá potřebné údaje ohledně ní a uloží ji. Aplikace nabídku uloží na server, a ta bude tak přístupná pro všechny uživatele.

2.5.2 UC2 - Správa nabídek

Uživatel chce upravit, nebo smazat pracovní nabídku. Jde do panelu pracovních nabídek, do textového vyhledávacího pole může zadat výraz, kterým budou následně nabídky filtrovány. Zadaný výraz bude vyhledáván jak v názvu, tak v popisu nabídky. Uživatel jednu vybranou nabídku buď otevře, upraví a uloží, nebo ji smaže. Pokud nabídku smaže, smažou se s ní i všichni kandidáti z **UC4** přiřazení k této nabídce a všechny komentáře a hodnocení z **UC6**, přiřazené k jednotlivým kandidátům. Aplikace akci zaznamená, uloží na server a změna se promítne i u ostatních uživatelů.

Aktivitu úpravy pracovní nabídky znázorňuje diagram 2.1

Nutná podmínka Je vytvořena alespoň jedna pracovní nabídka z **UC1**.

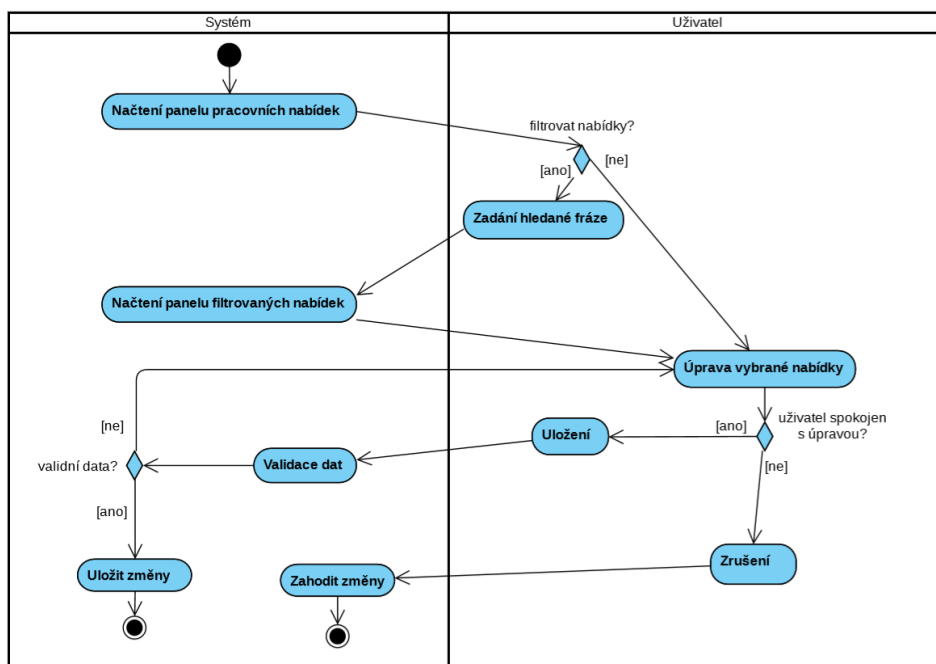
2.5.3 UC3 - Export nabídek

Uživatel chce exportovat všechny, nebo jen filtrované nabídky. Jde do panelu pracovních nabídek, kde exportuje všechny nabídky, nebo zadá hledaný výraz do vyhledávacího pole a exportuje pouze nabídky, které mu aplikace vyhledá.

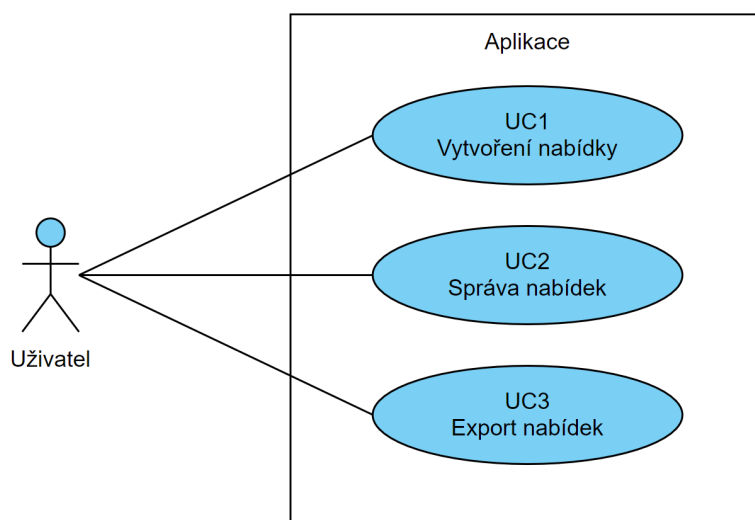
2.5.4 UC4 - Vytvoření kandidáta

Uživatel chce vytvořit nového kandidáta. Jde do panelu kandidátů, vybere možnost přidání kandidáta, zadá informace o kandidátovi, vybere pracovní nabídku, na kterou se kandidát hlásí, a uloží ho. Aplikace uloží záznam na server, přiřadí kandidáta k vybrané nabídce a poskytne informaci i ostatním uživatelům.

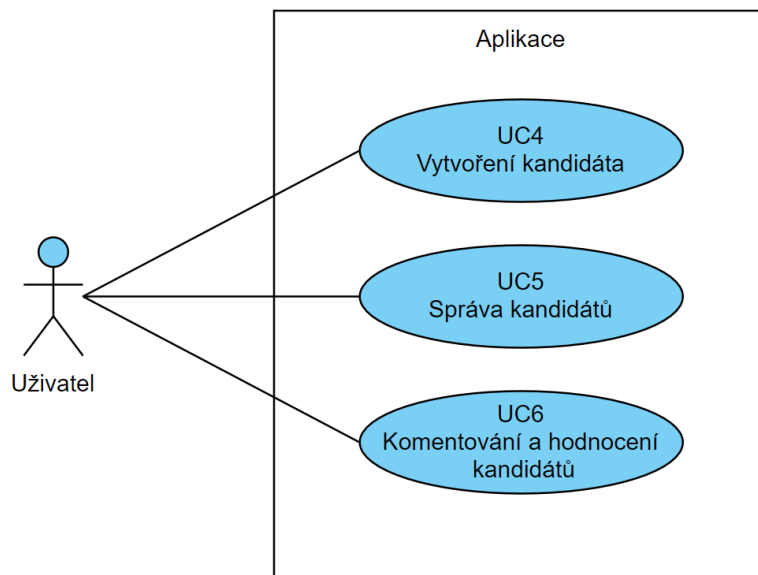
Nutná podmínka Je vytvořena alespoň jedna pracovní nabídka z **UC1**, ke které je kandidát přiřazen.



Obrázek 2.1: Diagram aktivity úpravy pracovní nabídky



Obrázek 2.2: Diagram případů užití související s pracovními nabídkami



Obrázek 2.3: Diagram případů užití související s kandidáty

2.5.5 UC5 - Správa kandidátů

Uživatel chce upravit nebo smazat kandidáta. Jde do panelu kandidátů, vybere příslušného kandidáta a buď ho otevře, upraví a uloží, nebo ho smaže. Smazáním kandidáta se smažou i k němu přiřazené komentáře a hodnocení z UC6.

Nutná podmínka Je vytvořen alespoň jeden kandidát z UC4.

2.5.6 UC6 - Komentování a hodnocení kandidátů

Uživatel chce přidat komentář a hodnocení ke kandidátovi. Jde do panelu kandidátů, vybere příslušného kandidáta, otevře ho a přidá komentář s hodnocením kandidáta. Aplikace přepočítá kandidátovo průměrné hodnocení, uloží komentář i hodnocení na server, aby bylo dostupné všem uživatelům.

Nutná podmínka Je vytvořen alespoň jeden kandidát z UC4, ke kterému se komentář a hodnocení přidává.

2.6 Doménový model

Důležitou částí analýzy je doménový model na obrázku 2.4. Tento platformě nezávislý model nám určuje závislosti jednotlivých entit z reálného světa a

pomůže je přenést do objektového návrhu tříd i databázového modelu.

2.6.1 Pracovní nabídka

Klíčová entita celé aplikace. Záznam obsahující název pracovní pozice, její popis, datum, kdy byla nabídka přidána do aplikace, a datum její poslední úpravy.

2.6.2 Platnost

Entita uchovávající datum začátku a datum konce platnosti pracovní nabídky.

2.6.3 Statistika

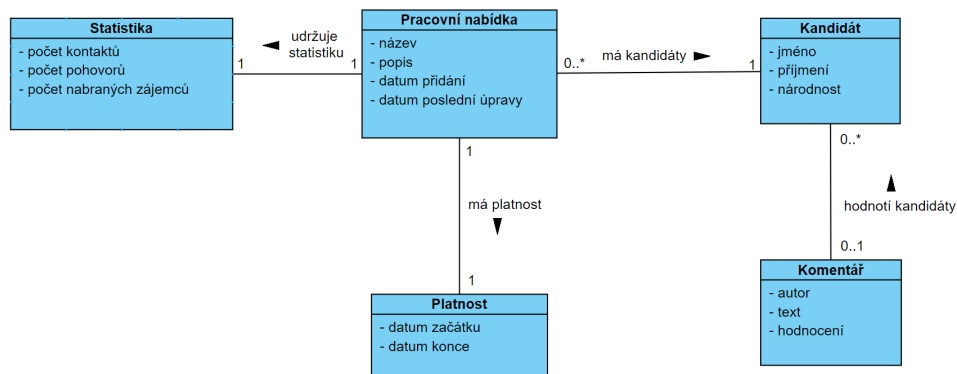
Entita Statistika zachycuje počet zájemců, kteří se na pracovní nabídku ozvali, počet těchto zájemců pozvaných na pohovor a počet úspěšně přijatých.

2.6.4 Kandidát

Tato entita představuje zájemce o jednotlivé pracovní nabídky a uchovává informace o něm.

2.6.5 Komentář

Entita, která uchovává hodnocení a komentáře ke kandidátům, na základě kterých je následně vypočítáno hodnocení Kandidáta.



Obrázek 2.4: Doménový model

Návrh aplikace a zvolené technologie

Kapitola popisuje technologie, které byly zvoleny k vypracování této práce a návrh samotné aplikace.

3.1 Zvolené technologie

3.1.1 ASP.NET MVC

ASP.NET MVC[4] je nástroj pro tvorbu webových aplikací. Je součástí platformy .NET, který běží na straně serveru a je určen pro tvorbu dynamických webových aplikací a služeb. Jedná se o alternativu k ASP.NET WebForms s tím rozdílem, že odděluje logiku a prezentaci. Je to ale také sada knihoven, díky kterým se webová aplikace vyvíjí v jazyce C#. ASP.NET MVC 5 je poslední verzí, tento nástroj nahradil ASP.NET Core.

3.1.2 C#

C# [5] je objektově orientovaný programovací jazyk vyvinutý firmou Microsoft¹³, umožňující vývoj aplikací běžících na platformě .NET. Je založen na jazycích C, C++ a Java a může být využit k tvorbě webových aplikací, servisních služeb, formulářových aplikací a mnoha dalších. Jazyk C# spadá do kategorie programovacích jazyků s virtuálním strojem, jako například Java. Znamená to, že je napsaný kód nejdříve přeložen do tzv. *mezikódu*, neboli CIL, a tento kód je dále interpretovatelný virtuálním strojem, u .NET je to CLR a výsledkem je již strojový kód. Tento proces vypadá na první pohled složitě, ale ve skutečnosti přináší mnoho výhod. Umožňuje například použití více jazyků, jako C++, Visual Basic, C#, F# a tyto jazyky jsou následně přeloženy do CIL. Dále má výhodu zabezpečení. Aplikace se dále může šířit jen jako kód v

¹³Dostupné na: <https://www.microsoft.com/>

CIL, který není snadno čitelný. Virtuální stroj CLR, na kterém běží tyto CIL kódy, pak dokáže sám spravovat paměť a rozpozná nebezpečný kód před jeho vykonáním a neprovede ho.

3.1.3 HTML

HTML[6] je základní stavební kámen téměř každé webové stránky. Jedná se o značkovací jazyk, protože používá značky k vytváření a formátování obsahu webových stránek. Tento kód je následně zobrazen webovým prohlížečem. Dnes se jazyk HTML používá spíše pouze k definování struktury webové stránky. Dříve se používal i k formátování vzhledu, ale k tomu se dnes používají spíše kaskádové styly, viz dále.

3.1.4 Razor

Razor[7] je nástroj, který je v ASP.NET MVC od verze 3, umožňuje použít jazyk C# přímo v HTML kódu webové stránky. Kód se vždy vyhodnotí ještě na straně serveru, ke klientovi do prohlížeče již jde pouze HTML, případně dodatečné skripty. Samotnou syntaxi Razoru ale prohlížeč nedokáže zpracovat.

3.1.5 CSS

Kaskádové styly[8] popisují, jak mají jednotlivé elementy HTML vypadat. Buď mohou být styly popsány v odděleném souboru, typicky *style.css*, který se pak ale musí nechat v hlavičce HTML stránky naimportovat, což jde udělat dvěma způsoby:

```
<head>
<link rel="stylesheet" type="text/css"
      href="style.css">
</head>
```

Což lze v ASP.NET MVC 3+ nahradit za volání s využitím Razor:

```
<head>
@Styles.Render("style.css")
</head>
```

Nebo rovnou zadávat parametry stylu k HTML elementu:

```
<h2 style="margin-top: 0px;">...</h2>
```

3.1.6 JavaScript

JavaScript[9] je, jak už jméno napovídá, skriptovací jazyk pro tvorbu a ovládání dynamického webového obsahu. Tím se myslí pohyblivé věci, animace obrázků,

aktualizace, aniž by se musela aktualizovat celá webová stránka. K webové HTML stránce se přidá podobně jako kaskádové styly:

```
<script type="text/javascript"
    src="/path/to/javascript"></script>
```

Lze odkazovat přímo na online verzi:

```
<script type="text/javascript"
    src="https://code.jquery.com/jquery-3.2.1.min.js">
</script>
```

Nebo lze skript připsat rovnou do HTML kódu, což je vhodné využít pouze pokud je skript opravdu stručný:

```
<script type="text/javascript">
alert("Page is loaded");
</script>
```

3.1.7 AJAX

Neboli asynchronní JavaScript a XML je využíván při tvorbě interaktivních webových aplikací, kde mění obsah View bez nutnosti celkové aktualizace stránky. Klient pomocí AJAX[10] pošle na server dotaz, většinou ve formátu JSON (JavaScript Object Notation), což je formát právě pro tuto výměnu dat. Server mu odešle zpět odpověď, která se zaznamená, a klient ji zpracuje. Mezitím je ale zobrazení webové stránky u klienta funkční a nedochází k žádnému načítání či zamrznutí, proto má tato technologie v názvu slovo asynchronní. Jelikož se jedná o knihovnu v jazyce JavaScript, většina novějších prohlížečů jej podporuje.

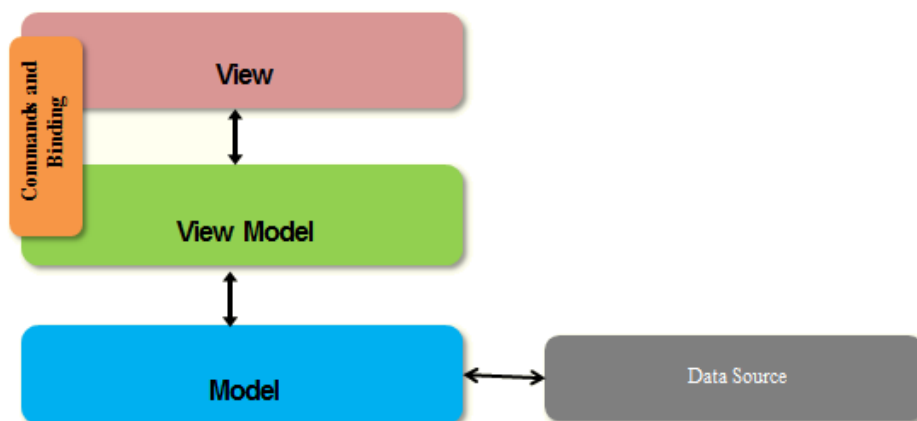
3.1.8 Knockoutjs

Tato technologie je v mé aplikaci zásadní, proto je uvedena až nyní, kdy již byly definované ostatní technologie, které jsou pro ni nutné. Knockout[11] je čistě JavaScriptová knihovna, která pomáhá vytvořit uživatelské rozhraní. Konkrétně se stará o svázání dat uživatelského rozhraní s datovým modelem a dokáže uživatelské rozhraní aktualizovat na základě změny datového modelu. Oproti například Angularu [12] není knockoutjs tolik obsáhlý a pro aplikace, které jsou méně složité na práci s daty, bohatě dostačuje. Podporuje všechny nejrozšířenější internetové prohlížeče.

Využívá se zde architektura MVVM, kde hlavní rozdíl mezi ní a MVC architekturou je, že MVVM má *binder* (svázání dat), které umožňuje automatickou komunikaci mezi View a ViewModelem. Jakákoliv uživatelem vyvolaná událost se tak projeví ve ViewModelu a obráceně, pokud se změní nějaká data ve ViewModelu, zobrazí se i uživateli ve View. Na obrázku MVVM Diagram

3. NÁVRH APLIKACE A ZVOLENÉ TECHNOLOGIE

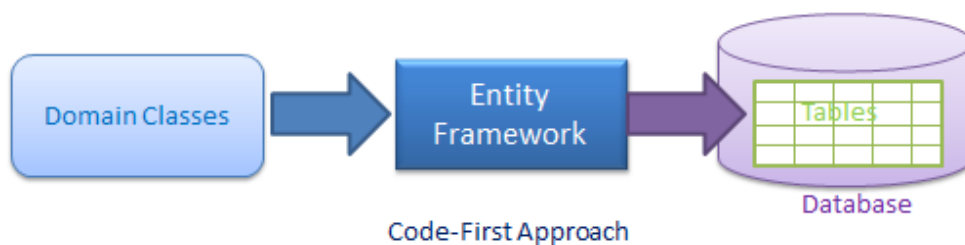
3.1 je zobrazeno, jak implementace MVVM (tedy Model-View-ViewModel) vypadá.



Obrázek 3.1: MVVM Diagram - převzato z [1]

3.1.9 Entity framework

Entity Framework [13] je určen pro interakci mezi .NET aplikací a relační databází, je to takzvaný ORM nástroj, který má usnadnit mapování mezi prvky relační databáze (tabulky, sloupce) a objektů v aplikaci. Je více způsobů použití tohoto nástroje, první je tzv. *Database first*, kdy se nejdříve vytvoří databázový model. Pro tuto práci bude využit tzv. *Code first*, kdy se nejdříve naimplementuje model v jazyce C# a z něj se následně vygeneruje a namapuje databáze, viz obrázek 3.2. Nespornou výhodou tohoto způsobu vytvoření databáze je čitelnost kódu.



Obrázek 3.2: EF - Code First přístup - převzato z [2]

3.1.10 Bootstrap

Bootstrap je dnes nejrozšířenější [14] volně dostupný nástroj pro tvorbu responzivních webů. Tento framework vznikl v roce 2010 ve firmě Twitter jako *Twitter Blueprint*. Je to framework obsahující šablony a styly, které může vývojář při tvorbě webu použít, a jednoduše tak vytvořit responzivní vzhled. Má rozsáhlou, propracovanou dokumentaci, která je dostupná na oficiálních stránkách [15]. Bootstrap je aktuálně nejnovější ve verzi 4.2.1, já jsem se rozhodl použít verzi 3.4.0, jelikož jsem s touto verzí již v minulosti pracoval a mám s ní vlastní zkušenosti.

3.2 Architektura

3.2.1 MVC na straně serveru

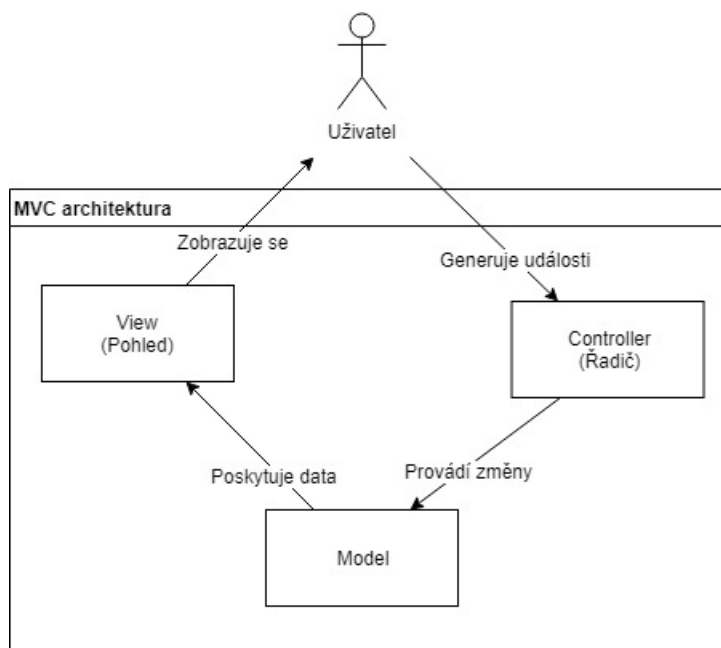
Byla zvolena architektura MVC, která je dnes běžná pro tvorbu uživatelského rozhraní. Rozděluje aplikaci do tří částí, a odděluje tak interní data od toho, co uživatel ve výsledku vidí. Jednotlivé komponenty jsou:

- Model - který reprezentuje vnitřní data, přímo s nimi nakládá a stará se o jejich uložení do databáze. Dále obsahuje business logiku a dotazy směřované na databázi. Běžně obsahuje i validace dat, v našem případě se data validují už na straně klienta.
- View(Pohled) - zajišťuje pouze prezentaci dat uživateli. Tato data přijdou z Modelu přes Controller.
- Controller(Řadič) - „naslouchá“ událostem od uživatele a zpracovává je dál pro Model, naopak od Modelu posílá data zpátky do View. Controller je takovým prostředníkem mezi uživatelem, Modelem a View.

3.2.2 MVVM na straně klienta

Tato architektura má výhody jako obousměrný binding dat mezi View, tedy tím, co se uživateli zobrazuje, a Modelem, tedy dat na pozadí. Bohužel má ale i stinnou stránku, a to že je velice špatně testovatelná.

- Model - zde opět reprezentuje vnitřní data, se kterými aplikace pracuje
- View(Pohled) - také zajišťuje reprezentaci dat uživateli, ale zde nastává rozdíl, kde data ve View jsou pevně svázána pomocí ViewModelu s Modelem, změní-li se tak něco na jedné straně, změní se to i na druhé.
- ViewModel - spojuje Model a View. Data jsou zde mezi nimi nabíndovaná a jak se změní v Modelu, ihned se změna promítne ve View.



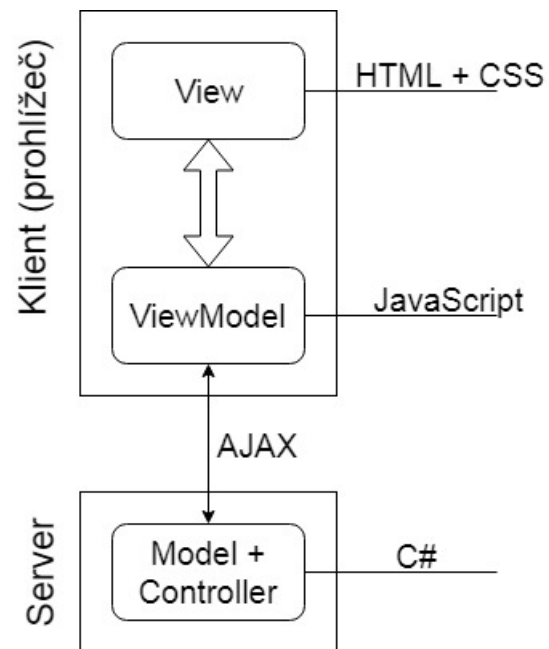
Obrázek 3.3: MVC Diagram

3.3 Souhrn

Tento souhrn popisuje, jak spolu jednotlivé technologie pracují a jak jsou spolu svázané.

Na straně serveru je architektura MVC. Je zde použit ASP.NET MVC, v jazyce C#, kterým je zastoupen Model a Controller. Zároveň ale server renderuje Views, které jsou v ASP.NET MVC zastoupeny ve formátu cshtml. Formát cshtml je HTML s přidaným kódem (Razor) určeným pouze pro server, dále přidanými CSS styly a skripty v JavaScriptu určenými pro klienta. Na stranu klienta přijde tedy vždy čisté HTML + CSS + JavaScript, které se zobrazí v prohlížeči a tvoří tenkého klienta.

Klient obdrží HTML + CSS + JavaScript. View zde představuje HTML + CSS, ViewModel zastupuje JavaScript a Model jsou data, která ViewModel obdrží od serveru. Tato data se získávají pomocí AJAX volání a pro jejich výměnu mezi klientem a serverem slouží přenos JSON objektů.



Obrázek 3.4: MVVM Diagram + jazyky jednotlivých vrstev

Implementace aplikace

V této kapitole popisuji, jak jsem samotnou aplikaci realizoval. K implemencaci jsem používal program Visual Studio 2019 Community Edition[16], který velice usnadňuje vývoj. Obsahuje například lokální databázi, nebo NuGet¹⁴ package manager, který se stará o správu a import potřebných knihoven. K testování HTTP API aplikace jsem použil nástroj Postman¹⁵, který je schopný přehledně zobrazit obdržená data a uložit různé volané požadavky, které pak jdou znovu použít.

4.1 Vývoj

Nejprve jsem zprovoznil jednoduchý projekt s architekturou MVC[17], nevyužívající žádná další rozšíření, a na něj jsem postupně přidával jednotlivou funkcionalitu. Začal jsem pracovat na serverové části, tedy vytvoření tříd potřebných k ukládání dat. Tady jsem použil Entity framework, kde se nastaví kontext databáze a EF se postará o databázi samotnou na základě zadaných modelů.

Následně jsem se přesunul k části prezentační, tedy na View. Tady jsem se snažil vytvořit co nejpřehlednější, jednoduchý a zároveň hezky vypadající vzhled aplikace, jelikož to je to jediné, s čím samotný uživatel přijde do kontaktu. Po nespočetném množství přečtených článků a návodů o uživatelských rozhraních, po přečtení spousty rad a tipů ohledně vzhledů webových stránek a nástroje Bootstrap jsem byl s vzhledem spokojen.

Měl jsem pocit, že už je má práce téměř hotová. Začal jsem tedy s poslední věcí a to propojení samotného View s Modelem. To byl pro mě kámen úrazu a několik probdřených nocí. Nasazení komponenty ViewModel považuji za nejnáročnější část celé práce. S JavaScriptem jsem si moc nerozuměl, tak jsem se mu chtěl vyhnout a nasadit ViewModel pomocí knihovny Automapper,

¹⁴Dostupný na: <https://www.nuget.org/>

¹⁵Dostupný na: <https://www.getpostman.com/>

kteřou se namapují Modely a jim příslušné prvky View. Bohužel jsem čerpal z literatury, která byla zhruba tři roky stará, a má verze Automapperu byla, jak jsem o dost později zjistil, novější. Po prohraném boji s Automapperem, jelikož jsem aktuální verzi nebyl schopný zprovoznit, jsem se rozhodl dát na zkušenosti programátorů na fórech.

A tak jsem napsal ViewModel v JavaScriptu, kterému jsem se tolik vyhýbal, a chtěl jsem v něm psát jen funkce pro výměnu dat s Controllerem. Jelikož jsem v JavaScriptu doposud pracoval jen okrajově, znamenalo to pro mne mnoho četby dokumentace a návodů, ale nakonec lituji, že jsem v něm nepracoval hned od začátku.

4.2 Souborová struktura

Zde je popsána adresářová struktura projektu. Pro jednoduchost jsou některé soubory jako skripty jQuery, styly Bootstrapu vynechány.

4.2.1 App_start

Tato složka obsahuje konfigurační soubory, které se registrují při spuštění aplikace a to v metodě *Application.Start()* umístěné v souboru **Global.asax**.

4.2.2 Content

Tato složka obsahuje převážně css soubory stylů a použité obrázky.

4.2.3 Controllers

Zde jsou veškeré controllery, které se starají o renderování Views a ApiControllery které se starají pouze o přenos serializovaných dat.

4.2.4 DAL

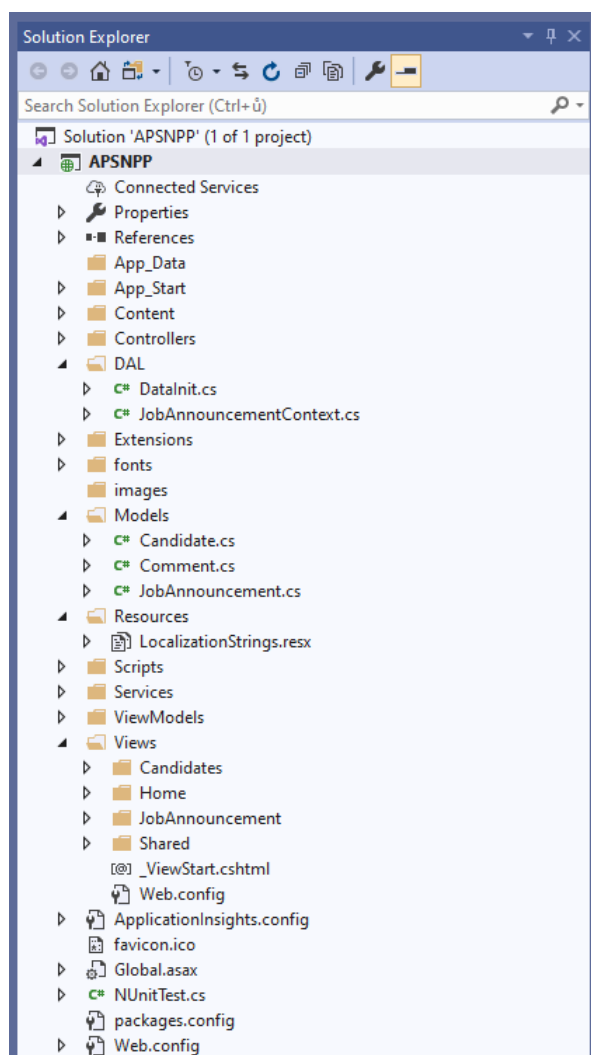
Data access layer, obsahuje třídu, která zajišťuje komunikaci s databází a inicializační třídu s testovacími daty do databáze.

4.2.5 Models

Obsahuje modely entit využívané v business logice aplikace a ukládající se do databáze.

4.2.6 Scripts

Zde jsou nadefinované JavaScriptové funkce pro zpracování dat na straně klienta, ViewModely pro KnockoutJS a zároveň podpůrné skripty jako jQuery, moment, které napomáhají správné funkci aplikace na straně klienta.



Obrázek 4.1: Adresářová struktura projektu

4.2.7 Resources

Zde je soubor LocalizationStrings.resx, který obsahuje jazykovou lokalizaci.

4.2.8 Views

Obsahuje soubory cshhtml, ze kterých se renderuje HTML na straně serveru a odesílá se klientovi do prohlížeče.

4.2.9 NUnitTest.cs

Soubor obsahující NUnit testy aplikace

4.3 Ukázky kódu

Následující kód ve ViewModelu se stará o načtení uživatelem zadaných dat z View, převede je do formátu JSON a posílá je Controlleru

```
var submitData = {
    ID : jobAnnouncementID ,
    Title : jobAnnouncementTitle ,
    Descr ... //zkraceni zapisu
};

$.ajax({
    type: 'PUT',
    url: '/api/JobAnnouncementsAPI/'+jobAnnouncementID ,
    contentType: 'application/json',
    data: ko.toJSON(submitData),
    success: function (data) {
        localStorage.setItem("Saved", true);
        window.location.href = urlPath + '/';
    }
})
```

Zde je metoda Controlleru, která přijme data a zpracuje si je. Pokud jsou v pořádku, uloží je do kontextu databáze.

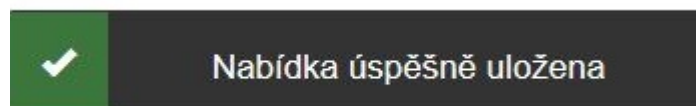
```
// PUT: api/JobAnnouncementsAPI/5
[ResponseType(typeof(void))]
public IActionResult PutJobAnnouncement(int id ,
    JobAnnouncement jobAnnouncement)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    if (id != jobAnnouncement.ID)
    {
        return BadRequest();
    }
    db.Entry(jobAnnouncement).State =
        EntityState.Modified;
    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateConcurrencyException)
```



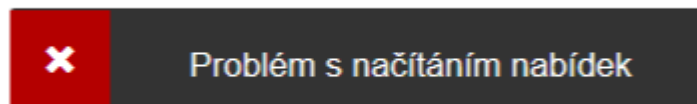
```
{
    if (!JobAnnouncementExists(id))
    {
        return NotFound();
    }
    else
    {
        throw;
    }
}
return StatusCode(HttpStatusCode.NoContent);
}
```

4.4 Uživatelské rozhraní

Snažil jsem se vytvořit přehledné a dobře vypadající prostředí. Tomu značně napomohl nástroj Bootstrap se spoustou šablon. Aby se uživatel vyhnul přepínání a odkazování na jiné stránky, rozhodl jsem se vytvořit formulář na přidání a editaci záznamu přímo na stránce pomocí vyskakovacího modalu. Díky tomu se uživatel nesetká s žádnou bílou obrazovkou a čekáním na znovunačtení celé webové stránky. Při ukládání a načítání je použito asynchronní volání, po kterém se aktualizuje pouze určitá část stránky a uživatele to tak nijak nerozptyluje. Dále jsou v aplikaci použity elegantní animované notifikace, informující například o úspěchu na obrázku 4.2 uživatelem provedené akce, či naopak o neúspěchu na obrázku 4.3 načtení nabídek.



Obrázek 4.2: Notifikace úspěšného uložení



Obrázek 4.3: Notifikace neúspěšného načtení nabídek

4. IMPLEMENTACE APLIKACE

Uživatelské rozhraní má texty a popisky v českém jazyce, využívající lokalizační soubor s těmito texty, umožňující jednoduchý překlad celé aplikace změnou jednoho souboru.

Detail pracovní nabídky

ID: 2

Název pozice: .NET Developer - e-commerce

Portál: www.jobs.cz

Popis: Do našeho e-commerce týmu hledáme aktuálně .NET vývojáře, který se s námi bude podílet na tvorbě a rozvoji e-shopů a webových řešení (některé z našich projektů a řešení je možné najít na: <http://webomat.sprinx.com/Reference>).

Od data: 2019-11-01

Do data: 2019-12-31

Kontaktováno: - 4 +

Po pohovoru: - 3 +

Zaměstnáno: - 1 +

Vytvořeno: 2019-11-01

Poslední úprava: 2019-11-15

Zrušit Uložit

Obrázek 4.4: Vyskakovací modal

Testování

Tato část popisuje, jak byla aplikace během vývoje i po něm testována. Zahrnuje jak uživatelské testování, automatizované testování uživatelského prostředí, tak testování během vývoje pomocí ladících nástrojů.

5.1 Průběžné testování během vývoje

Aplikace byla průběžně spouštěna a testována na lokálním IIS Express, které je součástí operačního systému MS Windows. Vývojářský nástroj MS Visual Studio podporuje ladění přímo v prohlížeči Microsoft Edge od stejné firmy. Tento ladící nástroj umožňuje krokování a sledování hodnot proměnných, což poměrně usnadní hledání chyb. Ladící nástroj nabízí sice i ladění skriptů, které se ale bohužel moc nedařilo, a tak muselo stačit vypisování logů do konzole prohlížeče, což podstatně zneprůjemňovalo práci.

Jako databáze bylo využito MSSQLLocalDB[18], která je také součástí MS Visual Studia a díky tomu šlo využívat nástroj SQL Server Object Explorer. Ten umožňuje sledovat aktuální data v databázi, případně je dle potřeby upravovat. Čehož bylo využito zejména při testování celkové „propustnosti“ všemi vrstvami aplikace, jestli se uživatelem zadaná informace dostane až do datového úložiště.

5.2 Uživatelské testování

Testování uživateli bylo prováděno již při vývoji aplikace. Několika osobám byl zadán úkol v aplikaci něco zaznamenat podle scénáře a nasimulovat tak práci personálního oddělení. Instrukce pro testery byly pouze takové, že se jedná o aplikaci pro personální oddělení a že zaznamenává pracovní nabídky a jejich kandidáty. Zbytek už byl na nich. Následně byli dotázáni, zda je pro ně vše dostatečně intuitivní, na první pohled jednoduché, a co by případně změnili. Toto testování proběhlo celkem úspěšně. Na jeho základě bylo jen upraveno

pár popisků ve formuláři pro zadávání nové nabídky, aby bylo na první pohled jasné, co je očekáváno za vstup do konkrétního pole a bylo přesunuto několik prvků ve formuláři pro úpravu kandidáta.

Toto testování bylo provedeno jen na lokálním stroji, jelikož nemám k dispozici server s platformou Windows, na který by mohla být aplikace nasazena.

5.3 Automatizované UI testování

Na testování aplikace byl použit nástroj Selenium [19], který dokáže simulovat interakci uživatele s webovou aplikací. Pomocí identifikátorů jednotlivých ovládacích prvků ve webové aplikaci na nich dokáže tento nástroj vyvolat události, jako je kliknutí, vyplnění textového vstupu, nebo hledání určitého výrazu na webové stránce. Byl použit spolu s nástrojem NUnit 3[20], který slouží pro tvorbu unit testů. Tento nástroj NUnit je určen pro .NET a je portován z JUnit, který je určen pro jazyk Java.

Zvolil jsem celkem 10 scénářů UI testů a to následovně:

1. Přidání nové nabídky
2. Kontrola data vytvoření nabídky
3. Editace nabídky
4. Kontrola data poslední úpravy nabídky
5. Smazání nabídky
6. Funkce filtrování nabídek
7. Přidání nového kandidáta
8. Přidání komentáře ke kandidátovi
9. Editace kandidáta
10. Smazání kandidáta

Následující kód představuje jeden z testů a to přidání nového záznamu pracovní nabídky. Každý NUnit test obsahuje návěští [SetUp], kde se vše nainicializuje a připraví k testu. Zde se nastaví ovladač pro webový prohlížeč Google Chrome¹⁶. Další návěští [Test] metody značí, že se jedná o samotný test, kde se provedou operace simulující chování uživatele. Nakonec se spustí metoda s návěštím [TearDown], kde probíhá případné vyčištění po testu a uzavření spojení s ovladačem.

¹⁶Dostupný na: <https://www.google.com/chrome/>

```
public class NUnitTest1
{
    IWebDriver driver;
    string testTitle = "Testovaci_nazev";
    [SetUp]
    public void Initialize ()
    {
        driver = new ChromeDriver ();
    }
    [Test]
    public void AddNewJATest ()
    {
        driver.Url = ".../JobAnnouncement";
        driver.FindElement(By.Id("addJA")).Click();
        Thread.Sleep(1500); //nez se modal zobrazi
        //prepnuti na modal, který se zobrazil
        driver.SwitchTo().ActiveElement();
        //nasleduje vyplneny udaju
        driver.FindElement(By.Id("TitleInput"))
            .SendKeys(testTitle);
        driver.FindElement(By.Id("PortalInput"))
            .SendKeys("www.test.com");
        //... zkraceni zapisu ...
        driver.FindElement(By.Id("saveBtn")).Click();
        //vyckani na knockout aktualizaci dat
        Thread.Sleep(5000);
        //kontrola jestli se zaznam nachazi v seznamu
        IReadOnlyCollection<IWebElement> list = driver
            .FindElements(By.XPath
                ("//*[contains(text(),'"+
                    testTitle + "')])");
        Assert.True(list.Count() > 0, "Job_announcement_"
            + testTitle + "_not_found!");
    }
    [TearDown]
    public void EndTest ()
    {
        driver.Close ();
    }
}
```

Výsledek a další možná rozšíření

6.1 Výsledek

Výsledkem této práce je funkční webová aplikace, která splňuje všechny požadavky, které byly definovány v analýze. Aplikace byla otestována jak uživateli, tak automatizovanými testy.

Architektura aplikace dodržuje programovací konvence, což by mělo usnadnit její případné rozšiřování.

6.2 Možná rozšíření

Aplikace by mohla být nasazena i mimo intranet firmy, aby byla dostupná veřejně a ne jen z vnitřní sítě. Pro tento účel by bylo potřeba vytvořit rozhraní pro přihlášení a autorizaci uživatelů. Uživatelům by pak mohly být přiděleny role s různým oprávněním k úpravám a k prohlížení záznamů.

Dále by aplikaci bylo možné rozšířit v modulu kandidátů o podporu správy jejich životopisů a mezd.

Závěr

Cílem mé práce bylo vytvoření aplikace pro přehlednou správu inzerátů nabídek pracovních pozic, která usnadní práci personálnímu oddělení při náboru do nových pracovních pozic. Nejprve byl nutný sběr požadavků, pak vytvoření návrhu aplikace a nakonec samotná implementace. Na závěr byla aplikace otestována a navrhnuti možnosti dalšího rozšíření.

Tato webová aplikace je první webová .NET aplikace, kterou jsem vytvářel od úplného začátku. Doposud jsem na webových aplikacích pracoval, ale pouze jsem je rozšiřoval, či opravoval existující řešení. Nebyl to tedy pro mě snadný úkol. Začátek implementace byl nakonec těžší, než jsem předpokládal. Před samotným programováním jsem si musel rozšířit znalosti ohledně vytváření nové aplikace v ASP.NET MVC a zvolit si vhodné technologie, které použiji. Možností je nespočetné množství a každá má své výhody i nevýhody. Několikrát jsem se tak dostal do slepé uličky, kdy až po několikahodinové práci jsem zjistil, že zvolená technologie není pro mou aplikaci vhodná a bude lepší použít jinou. Tak jsem se musel vrátit zpět na začátek.

Zpočátku mi nejvíce pomohla kniha[21] o ASP.NET MVC 5, kterou jsem měl možnost si zapůjčit v zaměstnání. Je sice z roku 2015, takže některé součásti již nebyly úplně aktuální, ale alespoň pro základy, na kterých bych mohl stavět, mi dobře posloužila.

Výsledkem je funkční webová aplikace zahrnující všechny požadavky, které byly v analýze stanoveny. Aplikace má českou lokalizaci, pro jednoduché a srozumitelné ovládání, která může být kdykoliv a jednoduše přeložena bez nutnosti zásahu do zdrojového kódu. Aplikace má pro uživatele atraktivní vzhled, zmínil bych například animované notifikace o úspěchu či neúspěchu uživatelem provedených operací.

Literatura

- [1] Gupta, D.: *Knockout Tutorial - Part 1 [online]*. 2013 [cit. 2019-10-30]. Dostupné z: <https://www.codeproject.com/Articles/636789/Knockout-Tutorial-Part-1>
- [2] *What is Code-First? [online]*. 2019 [cit. 2019-12-22]. Dostupné z: <https://www.entityframeworktutorial.net/code-first/what-is-code-first.aspx>
- [3] Microsoft: *Localize strings in your UI and app package manifest [online]*. [cit. 2019-12-19]. Dostupné z: <https://docs.microsoft.com/en-us/windows/uwp/app-resources/localize-strings-ui-manifest>
- [4] Microsoft: *ASP.NET MVC [online]*. © 2019 [cit. 2019-11-20]. Dostupné z: <https://dotnet.microsoft.com/apps/aspnet/mvc>
- [5] Microsoft: *Introduction to the C# Language and the .NET Framework [online]*. 2015 [cit. 2019-12-30]. Dostupné z: <https://docs.microsoft.com/en-us/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework>
- [6] Shannon, R.: *What is HTML? [online]*. 2012 [cit. 2019-12-30]. Dostupné z: <https://www.yourhtmlsource.com/starthere/whatishtml.html>
- [7] Sheridan, M.: *Using the Model Keyword in Razor - ASP.NET MVC [online]*. 2010 [cit. 2019-11-30]. Dostupné z: <https://www.dotnetcurry.com/ShowArticle.aspx?ID=601>
- [8] Morris, S.: *Tech 101: The Ultimate Guide to CSS [online]*. 2012 [cit. 2019-12-30]. Dostupné z: <https://skillcrush.com/2012/04/03/css/>
- [9] Morris, S.: *Tech 101: What is JavaScript? [online]*. 2012 [cit. 2019-12-30]. Dostupné z: <https://skillcrush.com/2012/04/05/javascript/>

- [10] Ingawale, S.: *What Is AJAX [online]*. 2019 [cit. 2019-11-12]. Dostupné z: <https://www.c-sharpcorner.com/blogs/what-is-ajax>
- [11] Google: *Introduction - Knockoutjs [online]*. © 1996 [cit. 2019-11-20]. Dostupné z: <https://knockoutjs.com/documentation/introduction.html>
- [12] Tomas Kirda: *Angular vs. Knockout: Similarities and Fundamental Differences [online]*. 2014 [cit. 2019-11-20]. Dostupné z: <https://www.devbridge.com/articles/angular-vs-knockout-similarities-and-fundamental-differences/>
- [13] *Entity Framework - Overview [online]*. 2019 [cit. 2019-12-30]. Dostupné z: https://www.tutorialspoint.com/entity_framework/entity_framework_overview.htm
- [14] Arsenault, C. *Top 10 Front-End Frameworks of 2018 [online]*, září 2018, [cit. 2019-11-20]. Dostupné z: <https://www.keycdn.com/blog/front-end-frameworks>
- [15] *Bootstrap - The most popular HTML, CSS, and JS library in the world.. [online]*. © 1996 [cit. 2019-11-20]. Dostupné z: <https://getbootstrap.com/docs/4.2/getting-started/introduction/>
- [16] Microsoft: *Visual Studio 2019 version 16.0.9 [online]*. 2019 [cit. 2019-11-15]. Dostupné z: <https://docs.microsoft.com/en-us/visualstudio/releases/2019/release-notes-v16.0>
- [17] Freeman, A.: *Pro ASP.NET MVC 5*. New York: Apress, 2013, ISBN 978-1430265290.
- [18] Microsoft: *Microsoft SQL Server 2016 Express LocalDB [online]*. 2016 [cit. 2019-12-30]. Dostupné z: <https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/sql-server-2016-express-localdb?view=sql-server-2017>
- [19] *Selenium - Web Browser Automation [online]*. 2010 [cit. 2019-12-30]. Dostupné z: <https://www.seleniumhq.org/>
- [20] *NUnit - unit-testing framework for all .Net languages [online]*. 2017 [cit. 2019-12-30]. Dostupné z: <https://nunit.org/>
- [21] Munro, J.: *ASP.NET MVC 5 with Bootstrap and Knockout.js*. O'Reilly Media, Inc., 2015, ISBN 9781491914373.

Uživatelský návod k použití

Aplikace pro správu pracovních nabídek

Aplikace obsahuje dva moduly. Modul pro pracovní nabídky a modul pro kandidáty na tyto nabídky.

Modul Pracovní nabídky

Po vstoupení na adresu, na které je webová aplikace hostována, přistoupíme na panel „Pracovní nabídky“ a zobrazí se nám seznam veškerých pracovních nabídek, viz obrázek A.1.

Chceme-li filtrovat uložené pracovní nabídky, napíšeme hledaný výraz do pole „Filtrovat“. Námi zadaný výraz se bude vyhledávat v polích „Název pozice“ a „Popis“.

Tlačítko „Export filtrovaných dat do excelu“ nám umožní uložení pracovních nabídek, bude na ně aplikován výše zmíněný filtr, a tak uložíme pouze ty nabídky, které chceme. Export je do souboru s příponou *.xlsx*, podporovanou programy Kancelářských balíčků, jako jsou Microsoft Excel, LibreOffice a další.

Pro vložení nové pracovní nabídky stiskneme tlačítko „Přidat nový záznam“. Zobrazí se modální okno s textovými poli pro nový záznam pracovní nabídky, viz obrázek A.2. V tomto okně vyplníme požadovaná pole (pole pro název pozice je povinné) a stiskneme tlačítko „Uložit“ pro zaznamenání nabídky. V opačném případě pak tlačítko „Zrušit“, nebo „X“ v pravém horním rohu pro zahoezení změn.

Pro smazání nabídky klikneme v seznamu pracovních nabídek na tlačítko „Smazat“. Dojde ke smazání záznamu a s ním se smažou i kandidáti, kteří byli přiděleni na tuto nabídku.

Po uložení záznamu se v spodní části obrazovky vysune notifikace o úspěšnosti této operace. Stejně tak při smazání záznamu nabídky nás aplikace upozorní, zda byla nabídka úspěšně smazána.

A. UŽIVATELSKÝ NÁVOD K POUŽITÍ

Aplikace pro správu pracovních nabídek Pracovní nabídky Kontakty

Pracovní nabídky 5 Počet záznamů + Přidat nový záznam

Filtrovat Pole pro filtrace záznamů

#	Název pozice	Popis	Do data	Možnosti
1	.NET Developer - e-commerce	Do našeho e-commerce týmu hledáme aktuálně .NET vývojáře, který se s námi bude podílet na tvorbě a rozvoji e-shopů a webových řešení (některé z našich projektů a řešení je možné najít na: http://webomat.sprinx.com/Reference).	2019-04-01	Upravit Smazat
2	Front-end Developer - ReactJS (možnost HO režimu)	* HTML, CSS, sass, less * JavaScript React, NodeJS, Flow, Neutrino, react scripts	2019-04-01	Upravit Smazat
3	Tester: SoapUI + SQL + pozitivní přístup k vývojářům	Chybí nám zkušený tester, který obsáhne SoapUI, SQL, i nějaký kus automatizace testů (skriptování postačí). Velký důraz klademe zejména na pokročilou znalost SoapUI.	2019-03-01	Upravit Smazat
4	C / C++ Programátor	Vývoj nové verze informačního systému v C++ Vývoj probíhá v prostředí OS Windows nebo Linux (možnost specializace) Spolupráce při analýze a testování funkčnosti vyvíjených aplikací	2019-01-10	Upravit Smazat
5	.NET ASP.NET Programátor / .NET Senior Developer	Do nově otevřeného pražského vývojového centra zahraniční SW společnosti hledáme juniorské i zkušené ASP.NET programátory.	2019-10-10	Upravit Smazat

ID záznamu Export filtrovaných dat do excelu

Obrázek A.1: Seznam pracovních nabídek

Modul Kandidáti

Po vstoupení na adresu, na které je webová aplikace hostována, přistoupíme na panel „Kandidáti“ a zobrazí se nám seznam veškerých kandidátů, viz obrázek A.5.

Chceme-li přidat nového kandidáta, klikneme na tlačítko „Přidat nový záznam“ a zobrazí se modální okno. V tomto okně se vyplní informace o kandidátovi. Pole „Jméno“, „Příjmení“ a „Pracovní pozice“ jsou povinné. Bez těchto údajů nám aplikace nedovolí kandidáta uložit. Záznam uložíme pomocí tlačítka „Uložit“ nebo zahodíme tlačítka „Zrušit“ a „X“ v pravém horním rohu modálního okna.

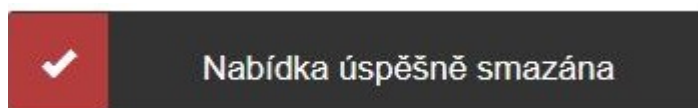
Pro úpravu záznamu kandidáta přistoupíme na panel „Kandidáti“ a zde ze seznamu tlačítkem „Upravit“ otevřeme modální okno s detaily vybraného kandidáta. Stejně jako u zadávání nového záznamu kandidáta musí být vyplněná povinná pole „Jméno“, „Příjmení“ a „Pracovní pozice“, bez kterých záznam nepůjde uložit.

Pro vložení komentáře ke kandidátovi postupujeme stejně jako u úpravy. Otevřeme detail kandidáta a v zápatí modálního okna vyplníme pole „Autor“, „Příjmení“ a na pravé straně zvolíme hodnocení, které chceme kandidátovi udělit. Uložíme ho tlačítkem „Přidat komentář“. Pro smazání záznamu kan-

Obrázek A.2: Přidání nové nabídky



Obrázek A.3: Notifikace uložení záznamu



Obrázek A.4: Notifikace smazání záznamu

didáta klikneme v seznamu kandidátů na tlačítko „Smazat“. Dojde ke smazání i všech komentářů přidělených kandidátovi.

A. UŽIVATELSKÝ NÁVOD K POUŽITÍ

Aplikace pro správu pracovních nabídek Pracovní nabídky Kandidáti Kontakty

Kandidáti 4 + Přidat nový záznam

ID	Jméno	Příjmení	Hodnocení	Hodnocení
1	Josef	Technický	4.0	<input type="button" value="Upravit"/> <input type="button" value="Smazat"/>
2	František	Prokop	5.0	<input type="button" value="Upravit"/> <input type="button" value="Smazat"/>
3	Adam	Milner	4.5	<input type="button" value="Upravit"/> <input type="button" value="Smazat"/>
4	Dana	Dozrak	5.0	<input type="button" value="Upravit"/> <input type="button" value="Smazat"/>

Obrázek A.5: Seznam kandidátů

Detail kandidáta

ID:

Jméno: Příjmení: Národnost:

Průměrné hodnocení: Pracovní pozice:

Komentáře 0 Hodnocení

Autor: Komentář:

Obrázek A.6: Přidání nového kandidáta

Detail kandidáta

ID: 5

Jméno: Tomáš Příjmení: Novák Národnost: CZE

Průměrné hodnocení: 0 Pracovní pozice: C / C++ Programátor

Zrušit Uložit

Komentáře 0 Hodnocení

Igor Dubec Určitě bude přínos pro náš tým, mnohaleté zkušenosti s C++, navíc má příjemné vystupování. Doporučuji. 5

Přidat komentář

Obrázek A.7: Přidání nového komentáře kandidátovi

Seznam použitých zkratek

AJAX Asynchronous JavaScript and XML

CIL Common Intermediate Language

CIL Common Language Runtime

CSS Cascading Style Sheets

EF Entity Framework

HR Human Resources

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

IIS Internet Information Services

JS JavaScript

JSON JavaScript Object Notation

MVC Model-View-Controller

MVVM Model-View-ViewModel

ORM Object-Relational Mapping

SQL Structured Query Language

UI User Interface

URL Uniform Resource Locator

XML Extensible Markup Language

Obsah přiloženého CD

	readme.txt	stručný popis obsahu CD
	src	
	impl	Visual Studio řešení, zdrojové kódy
	thesis	zdrojová forma práce ve formátu $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$
	text	text práce
	thesis.pdf	text práce ve formátu PDF