VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2020 Tomáš Chalupa

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Tomáš Chalupa**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: AVE Soft s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Ing. Radim Bača, Ph.D.**

Konzultant bakalářské práce: Ing. Martin Prokeš

Datum zadání: 01.09.2018

Datum odevzdání: 30.04.2020

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * 1D16 OSTRAVA *

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 15. května 2020 .....

Rád bych na tomto místě poděkoval všem, kteří mi s prací pomohli, protože bez nich by tato práce nevznikla.

Abstrakt

Tato práce popisuje průběh mé bakalářské práce formou individuální odborné praxe ve firmě AVE Soft s.r.o.. Ve firmě jsem působil jako C# .NET DEVELOPER. Mým úkolem bylo vytvořit pomocnou aplikaci k již existujícímu informačnímu systému Evolio, která by zvládla pracovat s velkým množstvím dat.

Klíčová slova: Bakalářská praxe; Informační systém; C#; WPF

Abstract

This paper discribes my bachelor's thesis by form of individual apprenticeship in company AVE Soft s.r.o..

Keywords: Bachelor thesis; Information system; C#; WPF

Obsah

Seznam použitých zkratek a symbolů

WPF	–	Windows Presentation Foundation
HTML	–	Hyper Text Markup Language
VB.NET	–	Visual Basic .NET
ASP	–	Active Server Pages
API	–	Application Programming Interface
REST	–	Representational State Transfer
SQL	–	Structured Query Language
JSON	–	JavaScript Object Notation
MVVM	–	Model–view–viewmodel
XML	–	Extensible Markup Language
XAML	–	Extensible Application Markup Language
DI	–	dependency injection
CDS	–	Custom Data Source

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

TODO!!

2 O firmě

AVE Soft s.r.o.[1] je softwarová společnost založena roku 1997 vyvíjející informační systémy pro exekutory a právní kanceláře, kromě toho také vytváří software na míru. Mezi jejich zákazníky patří české pobočky prodejců automobilů Opel a životní pojišťovna Wustenrot, D.A.S. - Pojišťovna pro právní výdaje. AVE Soft s.r.o.[2] získala řadu ocenění, včetně druhého místa v soutěži české mobilní aplikace 2012 nebo finalisty Microsoft Industry Awards 2007 a 2012.

2.1 Startupy

Vedle klasický produktů u nás vznikly projekty, které svým přesahem a zaměřením vyžadovaly odlišný přístup. Vzdali jsme se proto pohodlí běžné IT společnosti a projekty osamostatnili jako plnohodnotné startupy. Rozhodně nelitujeme.

2.1.1 EXDRAZBY.CZ

V roce 2010 byl změněn zákon, který umožnil provádět dražby nemovitostí online. Byla to skvělá příležitost pro rozjezd nového projektu exdrazby.cz.

Po osmi měsících vývoje byla na trh uvedena ostrá verze dražebního portálu, který se během prvního roku stal největším na trhu, svou pozici v dalších letech ještě posílil. Klíčovým faktorem úspěchu byla sada nadstandardních služeb rozšiřující základní službu provedení aukce.

Za prvních pět let provozu byly prostřednictvím exdrazby.cz vydraženy nemovitosti v hodnotě více než 8 miliard korun.

2.1.2 PRESENTIGO

Hlavní orientací tohoto startupu pod křídly AVE Soft s.r.o. je trh USA, z tohoto důvodu v Silicon Valey strávil téměř rok. Dnes se zákazníci nacházejí převážně v USA a ČR. Společnost získala investici ve výši 500 tisíc dolarů a našla zákazníky jako např. E.ON, O2, UPC, Škoda Auto apod.

Presentigo slouží pro zvýšení efektivity prodeje obchodním týmům. Firmám přináší dva základní benefity:

- Digitalizace obchodního procesu. Presentigo sbírá data z terénu, které pak využívají zaměstnanci marketingového oddělení i obchodní manažeři. Kromě toho dochází ke zjednodušení práce obchodníků.
- Na klíč jsou vytvářeny prezentace, které zákazníky zaujmou a zároveň pomohou dobře vysvětlit přínosy nabízeného produktu nebo služby.

2.1.3 MIXIEW

Myšlenka vytvořit Mixiew vzešla z následující úvahy: Každý amatérský závodník chce mít ze závodu nějaké fotky a video. Co takhle posbírat záznamy od profesionálních kameramanů i diváků na tratit a sestříhat originální video pro každého závodníka?

Aby celá myšlenka dávala smysl, bylo nutné najít funkční business model. Ukázalo se, že do sportu teče obrovské množství peněz od sponzorů. Ti stále hledají nové způsoby, jak tyto peníze vynaložit efektivně a měřitelně.

Závodníci svá videa z Mixiew sdílí na sociálních sítích a tak sponzor získá možnost propagace své značky nejen vůči závodníkům, ale i v okruhu jejich kontaktů.

2.2 Firemní software

2.2.1 Evolio 8

Evolio 8 je desktopová aplikace napsaná v C# WPF s jádrem z předchozí verze napsané v VB.NET. Jedná se o informační systém navržený pro použití v advokacii a příbuzných oborech.

2.2.2 Evolio

Tento přímý nástupce Evolio 8 je webová aplikace napsaná technologií TypeScript, backend pak v C# ASP.NET Core MVC.

2.2.3 EData

EData je webové REST API napsané v jazyce C# a postavené na technologii ASP.NET Core MVC. Slouží především k přístupu do databázi pro Evolio a Evolio Power Reporting.

3 Moje úkoly

Moje hlavní náplní praxe byl vývoj aplikace Evolio Power Reporting, sekundárně pak také oprava chyb aplikace Evolio 8.

3.1 Evolio Power Reporting

Evolio Power Reporting je desktopová aplikace, kterou jsem měl za úkol vytvořit. Tato část aplikace Evolio 8 (v ní pod názvem „Filtry“) by jako webová aplikace fungovala hůře. Filtry jsou uživatelem napsané SQL dotazy. Pro optimalizaci exekučního plánu jsou uloženy jako procedury, nad kterými lze provádět hromadné operace, ať už přes SQL nebo jiné interní systémy.

Vzhledem k tomu, že filtry mohou vrátit veliké množství dat – až v řádech milionů záznamů, není webová aplikace pro jejich implementaci vhodná.

3.1.1 Demo aplikace

Prvním krokem při vývoji této aplikace bylo zjistit, zda vůbec lze vytvořit dostatečně rychlou aplikaci, která by zvládla přibližně milion záznamů a začala je zobrazovat ihned poté, co je obdržela ze serveru – bez potřeby stáhnout je před zobrazením všechny, jako tomu bylo u předchozí aplikace Evolio 8. Taktéž tato demo aplikace mi sloužila jako seznámení s WPF, se kterým jsem do té doby nepracoval.

K databázi jsem měl přístup jen přes EData takže můj první úkol byl napsat controller který by poskytoval data pro daný filter. Dostat data z databáze do controlleru bylo celkem jednoduché, v ostatních controllerech se požíval „Dapper“ tak jsem ho požil také, má už zabudouvanou serilizaci takže to jsem řešit nemusel, bohužel nemam žadný model do kterého bych mohl data serilazovat tak sem si musel vystačit se serializací do pole objektů, což je skoro jako dynamická třída. Teď jsem ale musel dostat tyto data z EDat do mé aplikace.

3.1.2 Serializace

Jeden z prvích rozhodnutí které jsem musel udělat bylo v jakém formátu data přenášet. Plain text, nejednoduší možnost, měla nevýhodu že se ztratil datový typ. Je sice možné parsovat string zpátky do jednotlivých typů ale není to nejrychlejší ba naopak je to pomalé natolik že jsem to použít nemohl.

Další můj pokus by byl buď JSON nebo nějak data serilizovat do binaru, při zkoumání těchto dvou možností jsem narazil na MessagePack.

3.1.3 MessagePack

MessagePack[3] je velmi rychlí bynární serilizační formát. Naštěstí už někdo ho naimplementoval pro C# takže jsem to nemusel implantovat sám. Bohužel jsem měl pořád ten samý problém že nemam model takže jsem je musel je mít v pamětí jako pole polí objektů, kde každé pole objektů

byl jeden řádek. Teoreticky bych si mohl za běhu vytvořit třídu a serializovat data na ní, ale pak bych k datům musel přistupovat přes reflexi a nemyslím si že to by bylo dostatečně rychlé. Ta implementace MessagePacku[4] kterou jsem si vybral dokonce podporovala kompresní pomocí LZ4. LZ4[5] je velmi rychlí bezztrátový kompresní algoritmus. Který ještě nadále zrychlil přenos, zmenšením přenášených dat.

3.1.4 WPF

WPF bylo vybráno hlavě kvůli tomu že firma vlastnila už nějaké komponenty které byli použity v předchozí aplikaci.

WPF podporuje návrhový vzor MVVM který, stejně jako další MVX vzory odděluje data od vizuálu. To je docíleno pomocí bindingu. Což je mechanizmus který synchronizuje data co jsou ve View a ViewModel. To znamená že pokud uživatel zadá neco do text boxu tak se to oběví i v property která je na ten text box na bindovaná. Pokud ViewModel implementuje rozhraní „INotifyPropertyChanged“ tak tento proces bude fungovat i opačným směrem.

Pro to aby binding fungoval tak se nad každým View musí vyplněné property DataSource což je náš ViewModel, a pro zachování separace mezi daty a vizuálem je toto děláno přes DI. Kde na začátku programu si specifikuji k jakému View vytvořit jaký ViewModel a při vytvoření je spáruji. A pokud chce jakákoliv část programu komunikovat s tím View tak to musí udělat přes ten ViewModel.

View jsou napsána v XAMLu což je značkovací jazyk založený na XML. Kde jednotlivé tagy jsou reprezentace tříd v C# (např. „<sys:Int32>“ kde „sys“ je namespace „System“, je třída „Int32“). XAML umožňuje editaci za běhu programu, což bylo velmi užitečné při stylování.

Nejsložitější částí byli styly, hlavně kůly tomu že jsem požíval komponenty od třetích stan. Ty sice většinou podporovaly nějakou customizaci ale ne vždy to stačilo. To s mou neznalostí WPF/XAML způsobilo že sem většinou nad tímto strávil dosti času.

3.1.5 Lokalizace

Poněvadž aplikace Evolio 8 byla používaná nejen v České republice lokalizace byla velmi důležitá. Překlady jsou uloženy v databázi takže po přihlášení uživatele si je můžu stáhnout, do té doby překlady budou muset být hardcodovaný v aplikaci. Také je důležité mít přístup jak z C# tak z XAMLu. Pro C# stačí statická metoda ale pro XAML je po třeba napsat speciální třídu, v mém případě třída „Localization“, která bude dědit z „MarkupExtension“. Což mi dovolí použít tu třídu takhle.

```
<Label Content="{Shared:Localization Key='PŘEKLADOVÝ.KLÍČ', Default='Defaultní
    text'}"/>
```

Výpis 1: MarkupExtension příkad

Kde „Shared“ je namespace kde třída „Localization“ je, „Localization“ je MarkupExtension třída, „Key“ je property na třídě „Localization“ která obsahuje překladový klíč a property „Default“ obsahuje defaultní text který se zobrazí pokud překlad není dostupný.

3.1.6 Filtry

Filter je rozdělen do několika tabulek, tabulka „Filters“ obsahuje jméno, SQL dotaz a jméno procedury plus nejáké další nepoviné data.

Dále tabulka „FilterProperties“ obsahuje data pro jednotlivé property filtru jako je jméno, datová typ, přesnost, velikost, jestli je parametr povinný a také data pro CDS.

Tabulka „FilterResultColumns“ obsahuje data o sloupcích které filter vrací. Ty jsou generována při ukládání, nebo editaci, pomocí SQL funkce „sys.sp_describe_first_result_set“ která bez toho aby proceduru spustila vrátí sloupce které by procedura vrátila. Taktéž se tam ukládají data k zobrazení sloupců.

A nakonec tabulka „FilterActions“ obsahuje data k nepárovaným akcím.

3.1.7 Editor filtrů

Další součástí této aplikace byl editor filtrů, který dovolil uživatelům přímo v aplikaci vytvářet a editovat filtry.

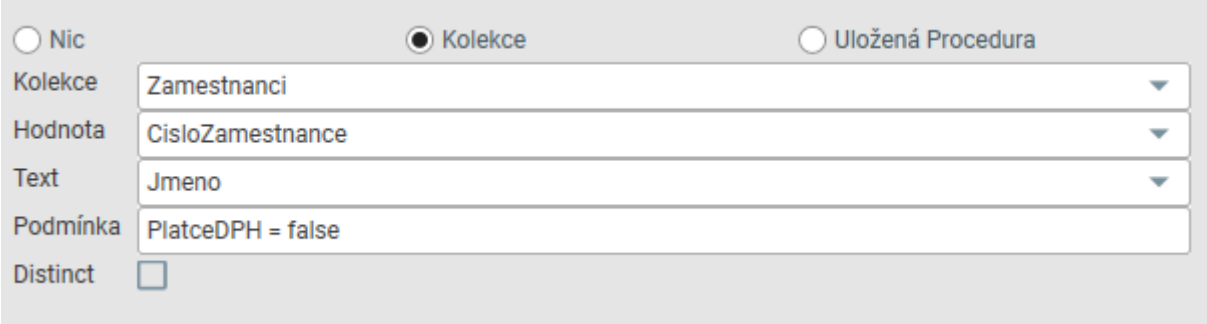
ukládání,

validace,

sql

3.1.8 CDS

CDS je velice podobné Enumu v C#. Kde je hodnota, u Enumu číslo, a text který vidí uživatel. CSD má dva typy, ten první je z tabuky a to znamená že si uživatel vybere dva sloupce z jedné tabulky a jeden bude text a druhý bude hodnota. Taktéž na to jde přidat podmínku pro výběr jen nějakých hodnot.



Obrázek 1: Tohle je ukázka nastavení CSD 1. typu.

Pro uživatele by CSD z obrázku 1 vypadal jako combobox jmen zaměstnanců co mají PlatceDPH nastaveno na false, proceduře by se ale poslalo CisloZamnestnance místo jména.

Druhý typ CSD je předem připravená procedura, pro složitější dotazy (např. když je nutno použít join) je potřeba vytvořit proceduru která bude vracet páry textu a hodnot.

3.1.9 Akce

TODO!!

4 Využité a scházející znalosti a dovednosti

4.1 Využité znalosti

C#

asp.net

4.2 Scházející znalosti

WPF

async

5 Závěr

TODO!!

Literatura

1. AVE SOFT S.R.O. *AVE Soft s.r.o.: Přehled | LinkedIn* [online]. 2001 [cit. 2018-09-02]. Dostupné z: `https://www.linkedin.com/company/ave-soft-s-r-o-/`.
2. AVE SOFT S.R.O. *O nás - AVE Soft.* [online]. 2001 [cit. 2018-09-02]. Dostupné z: `https://avesoft.cz/o-nas/`.
3. FURUHASHI, Sadayuki. *MessagePack: It's like JSON. but fast and small.* [online]. 2008 [cit. 2018-09-30]. Dostupné z: `https://msgpack.org/`.
4. KAWAI, Yoshifumi. *MessagePack: Extremely Fast MessagePack Serializer for Csharp* [online]. 2017 [cit. 2018-09-30]. Dostupné z: `https://github.com/neuecc/MessagePack-CSharp/`.
5. KRAJEWSKI, Milosz. *l4z: Extremely Fast Compression algorithm.* [online]. 2017 [cit. 2018-09-30]. Dostupné z: `https://github.com/lz4/lz4`.