

Testovací prostředí pro výuku kurzů Microsoft IT Academy na UTB ve Zlíně

Software Environment for Teaching Courses
on the Microsoft IT Academy by TBU in Zlín

Vladimír Štusák



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2010/2011

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Vladimír ŠTUSÁK**

Osobní číslo: **A10778**

Studijní program: **B 3902 Inženýrská informatika**

Studijní obor: **Informační a řídicí technologie**

Téma práce: **Testovací prostředí pro výuku kurzů Microsoft IT Academy na UTB ve Zlíně**

Zásady pro vypracování:

1. Vypracujte literární rešerši na téma testovacích prostředí pro klasifikaci studentů.
2. Provedte rozbor možností jazyka C pro zabezpečení aplikací a demonstруйте na příkladech.
3. Vytvořte softwarové testovací prostředí pro klasifikaci studentů kurzů MS IT Academy na UTB ve Zlíně tak, aby výsledný produkt umožňoval klasifikaci studentů kurzů formou elektronických testů.
4. Navrhněte a realizujte zabezpečení testovacího prostředí před jinou činností studentů, než je vypracovávání otázek.
5. Vytvořte rozhraní testovacího prostředí pro správu a editaci testů.
6. Založte nový test se vzorovými otázkami / odpověďmi.
7. Vypracujte uživatelský manuál k prostředí.
8. Podrobně popište jednotlivé kroky použité k zabezpečení testovací aplikace.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. SHARP, John. Microsoft Visual C 2010 – Krok za krokem. Computer Press, 2010. ISBN 978-80-251-3147-3.
2. NASH, Trey. C 2010 – Rychlý průvodce novinkami a nejlepšími postupy. Computer Press, 2010. ISBN 978-80-251-3147-3.
3. MARTIN, C., Robert. Čistý kód – Návrhové vzory, refaktorování, testování a další techniky agilního programování. Computer Press, 2009. ISBN 978-80-251-2285-3.
4. BISHOPOVÁ, Judith. C – návrhové vzory. Přeložil Jiří Koutný. Zoner Press, 2010. ISBN 978-80-7413-076-2.
5. HANÁK, Ján. C 3.0 – Programování na platformě .NET 3.5. Zoner Press, 2009. ISBN 978-80-7413-046-5.

Vedoucí bakalářské práce:

Ing. Lukáš Kouřil

Ústav informatiky a umělé inteligence

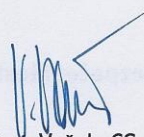
Datum zadání bakalářské práce:

25. února 2011

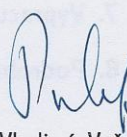
Termín odevzdání bakalářské práce:

7. června 2011

Ve Zlíně dne 25. února 2011


prof. Ing. Vladimír Vašek, CSc.
děkan




prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Tato práce se zabývá možnostmi zkoušení znalostí u studentů pomocí testů na počítačích. Práce okrajově porovnává dostupné prostředky a je především zaměřena na vývoj nového jednoduchého testovacího prostředí, ve kterém bude účinně zabráněno studentům použít při testu jiné zdroje informací, než jsou jejich znalosti.

Klíčová slova:

Microsoft Visual Studio, závěrečná zkouška, testovací prostředí, WPF, C#, Entity Framework, SQL server

ABSTRACT

This thesis deals with the possibilities of testing students' knowledge with the help of computer-based tests. The thesis briefly compares available testing means. Its main aim is to develop a new and uncomplicated testing environment in which the tested students will be prevented from using other sources of information than their knowledge.

Keywords:

Microsoft Visual Studio, final exam, testing environment, WPF, C#, Entity Framework, SQL server

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce Lukáši Kouřilovi, Ing za jeho čas, trpělivost, připomínky a pomoc při řešení mé bakalářské práce.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST.....	10
1 TESTOVACÍ PROSTŘEDÍ	11
1.1 VÝHODY A NEVÝHODY ELEKTRONICKÉHO TESTOVÁNÍ.....	11
1.1.1 Výhody	11
1.1.2 Nevýhody	11
1.2 PŘÍKLADY TESTOVACÍCH APLIKACÍ	12
1.2.1 Moodle.....	12
1.2.2 Prometric	12
1.2.3 LMS eDoceo.....	13
2 POUŽITÉ TECHNOLOGIE.....	14
2.1 MICROSOFT VISUAL STUDIO 2010	14
2.2 WINDOWS PRESENTATION FOUNDATION.....	15
2.2.1 Jazyk XAML	15
2.3 PROGRAMOVACÍ JAZYK C#.....	15
2.4 ADO.NET ENTITY FRAMEWORK	15
2.5 MICROSOFT SQL SERVER 2008 R2	16
2.5.1 SQL jazyk.....	16
II PRAKTICKÁ ČÁST	17
3 HLAVNÍ MYŠLENKA A PŘEDSTAVA ŘEŠENÍ.....	18
3.1 POŽADAVKY NA APLIKACI	18
3.1.1 Testy z více předmětů.....	18
3.1.2 Znemožnění podvádění během vypracování testu	18
4 NÁVRH ŘEŠENÍ.....	19
4.1 DESIGN A WINDOWS PRESENTATION FOUNDATION.....	19
4.1.1 Studentské rozhraní aplikace.....	19
4.1.1.1 Zabezpečení proti podvádění při testu	20
4.1.2 Lektorské rozhraní aplikace	20
4.2 DATABÁZE SQL A PRÁCE S DATY (TESTY A UŽIVATELÉ)	21
4.3 PROPOJENÍ DATABÁZE S APLIKACÍ.....	21
5 OVLÁDÁNÍ APLIKACE – UŽIVATELSKÝ MANUÁL	22
5.1 INSTALACE A NASTAVENÍ SQL SERVERU	22
5.1.1 Instalace aplikace a požadavky	22
5.1.2 Vytvoření databáze na SQL serveru.....	22
5.1.3 Potřebná nastavení SQL serveru	23
5.1.4 Doporučení pro nasazení aplikace na testovací počítače	23
5.2 SPUŠTĚNÍ APLIKACE	23
5.2.1 Název aplikace	23
5.3 NASTAVENÍ PŘIPOJENÍ K DATABÁZI S TESTY	24
5.3.1 Za běhu aplikace.....	24
5.3.2 Konfiguračním souborem.....	25

5.4	STUDENTSKÉ ROZHRANÍ	25
5.4.1	Popis přihlášení a registrace	25
5.4.2	Výběr testu	26
5.4.3	Vypracování testu	27
5.4.4	Odeslání vypracovaného testu	27
5.4.5	Ukončení práce se studentskou verzí aplikace	28
5.5	LEKTORSKÉ ROZHRANÍ	28
5.5.1	Popis přihlášení a registrace	28
5.5.1.1	První uživatel v lektorské roli	28
5.5.1.2	Změna role registrovaného uživatele	29
5.5.2	Vytvoření skupiny testů	30
5.5.3	Detail skupiny a okruhy otázek	31
5.5.4	Detail okruhu a otázky	35
5.5.5	Ukončení práce s lektorskou verzí aplikace	37
6	IMPLEMENTACE APLIKACE	38
6.1	PŘIPOJENÍ K DATABÁZI	38
6.2	REGISTRACE A PŘIHLÁŠENÍ UŽIVATELŮ	39
6.2.1	Kontrola zadaných údajů při registraci a registrace	39
6.2.2	Autentizace uživatele při přihlášení	40
6.2.3	Bezpečnost zadávaných hesel před odcizením	40
6.3	VYTVÁŘENÍ SKUPIN TESTŮ, OKRUHŮ OTÁZEK A OTÁZEK	40
6.4	VYTVÁŘENÍ ŠABLON TESTŮ	40
6.5	GENEROVÁNÍ TESTŮ A KOPIE OTÁZEK	40
6.6	VYHODNOCOVÁNÍ TESTŮ	42
6.7	UKLÁDÁNÍ TESTŮ	42
6.7.1	Průběžné ukládání – ochrana proti výpadku sítě	42
6.7.1.1	Zneužití ochrany proti výpadku sítě	43
6.8	ZABEZPEČENÍ PROTI PODVÁDĚNÍ	44
6.8.1	Odchycení klávesových zkratk pro přepínání mezi aplikacemi	44
6.8.2	Využití vlastností WPF oken	44
6.9	KOMPLIKACE PŘI IMPLEMENTACI	45
7	TESTOVÁNÍ ROZHRANÍ	46
8	DALŠÍ MOŽNÝ VÝVOJ	47
	ZÁVĚR	48
	ZÁVĚR V ANGLIČTINĚ	49
	SEZNAM POUŽITÉ LITERATURY	50
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	51
	SEZNAM OBRÁZKŮ	52
	SEZNAM PŘÍLOH	53

ÚVOD

Cílem bakalářské práce je porovnat dostupné prostředky pro testování studentů a určit jejich výhody a nevýhody.

Po tomto zhodnocení si vzít ponaučení z dostupných řešení a pokusit se implementovat vlastní řešení testovacího rozhraní, ve kterém bude kladen důraz na předcházení jiné činnosti na počítači, než je vypracování zadaného testu. Usoudil jsem, že by bylo vhodné se pokusit i o variabilně generované testy, které by zabezpečili, že studenti nebudou moci kontrolovat s jinými studenty tak, že jim nenápadně podívají na jejich počítač, kde by za běžných okolností běžel shodný test.

Součástí celé práce je také rozhraní pro lektory, ve kterém by bylo možno spravovat jednotlivé otázky, vytvářet testy a zobrazovat výsledky testů a podrobné výsledky testu, který vypracoval určitý student (například sloužící k nahlédnutí do vypracovaného testu studentem, při jeho ústním dozkoušení).

Z osobní zkušenosti, kterou v této oblasti mám díky svému zaměstnání, vím, že je vhodné, aby studenti viděli pouze aktuálně dostupné testy a ne všechny vytvořené testy, aby se minimalizovala chyba výběru špatného testu studentem. Z tohoto chci u jednotlivých testů implementovat také zámky. Odemčené testy budou vidět ve studentském rozhraní, kdežto o existenci zamknutých testů budou vědět pouze lektori v lektorském rozhraní aplikace.

Od této práce především očekávám rozšíření svých znalostí a zkušeností v oblasti programování uživatelských aplikací pro Microsoft Windows v jazyce C#, v němž budu celé rozhraní implementovat za využití ovládacích prvků Windows Presentation Foundation.

I. TEORETICKÁ ČÁST

1 TESTOVACÍ PROSTŘEDÍ

Před tím, než jsem začal s vlastní implementací testovacího rozhraní pro studenty, jsem si ověřil, jaké jsou aktuální možnosti testování studentů. Na tomto poli dominují aplikace Moodle, popřípadě Prometric.

1.1 Výhody a nevýhody elektronického testování

Elektronické testování studentů má různé výhody a nevýhody v závislosti na použití konkrétní aplikace. Rozdíly mezi aplikacemi budou vypsány níže, nejdříve shrnutí obecných vlastností.

1.1.1 Výhody

- Ulehčené opravování testu – testy automaticky opraví testovací prostředí bezprostředně po dokončení testu a počet testů nehraje žádnou roli.
- Vyloučení opisování – při elektronických testech může aplikace z databáze generovat otázky v testu náhodně, což má za následek velké množství variant zadání. Pokud máme například pro vygenerování testu k dispozici 80 otázek a test bude obsahovat pouze 20 náhodně vybraných otázek ještě s odpověďmi v náhodném pořadí, je velmi malá pravděpodobnost, že dva studenti vedle sebe dostanou alespoň podobný test.
- Velké množství různých testů – viz bod „Vyloučení opisování“
- Objektivita při hodnocení testů – při vyhodnocování testu aplikací jsou zaručeny stejné podmínky pro všechny studenty.
- Statistiky úspěšnosti – pokud aplikace umožňuje.

1.1.2 Nevýhody

- Nároky na vybavení učebny počítačovou technikou – pro vypracování elektronického testu je vždy potřeba počítač s běžící aplikací a podle implementace aplikace případně server, který obsahuje datové podklady k testům.
- Zajištění ověření studenta – jestli test vypracovává student, který má být zkoušený.
- Neexistuje ekvivalence některých typů otázek – například „Nakresli graf průběhu VA charakteristiky diody“.
- Absence individuálního přístupu zkoušejícího lektora ke zkoušenému studentovi.

1.2 Příklady testovacích aplikací

1.2.1 Moodle

Moodle je online vzdělávací prostředí, které podporuje prezenční i kombinovanou formu výuky prostřednictvím online kurzů dostupných přes internet. Umožňuje snadnou publikaci studijních materiálů, zakládání diskuzních fór, sběr a hodnocení elektronicky odevzdávaných úkolů, tvorbu online testů a řadu dalších činností pro podporu výuky (čerpáno z [1]).

Tento systém umožňuje vytváření testů, které budou dostupné odkudkoliv přes internet nebo pouze z dané učebny (na základě skupiny IP adres počítačů). Bohužel však už neumožňuje studentům zabránit v tom, aby si zobrazený test uložili a poté s ním dále disponovali nebo aby studenti během testu měli otevřené na počítači studijní materiály a používali je během testu. Tyto problémy se však mohou řešit dalšími nástroji, například v operačním systému Windows nastavením uživatelských účtů přes Group Policy.

Další nevýhodou tohoto systému je nutnost být připojen k internetu, protože celý systém běží na webovém serveru a přistupuje se k němu přes internetové prohlížeče.

Naopak výhodou Moodlu je, že je dostupný nezávisle na použitém operačním systému.

Moodle umožňuje všechny možné druhy testů. Například otázky s jednou správnou odpovědí, otázky s různým počtem správných odpovědí, otázky, ve kterých student odpovídá celou větou, i otázky ve kterých student doplňuje slovíčka přímo do textu pomocí nabídky možných odpovědí.

Snad největší výhodou systému Moodle je to, že se jedná o volně šiřitelný software s otevřeným kódem a běží na kterémkoliv systému podporujícím PHP. Data jsou ukládána v databázi MySQL, Microsoft SQL nebo Oracle.

1.2.2 Prometric

Prometric je americká firma zabývající se provozem sítě zkušebních středisek. Prometric nabízí širokou škálu služeb v oblasti testování a certifikace pro velké firmy jako je například Microsoft.

Prometric vlastně zajišťuje testování a certifikaci klientů (studentů/lektorů), poskytuje v této oblasti outsourcované služby.

Systém aplikace Prometric je založený na operačních systémech Windows s připojením k internetu. Přes několik drobných nevýhod (uvedeny níže) má velmi dobře vyřešenou právě oblast zabezpečení zneužití nepovolených zdrojů informací během testu. Při přihlašování uživatele v OS Windows se spustí na pozadí proces, který potlačí spuštění aplikace Explorer, která uživateli umožňuje pracovat s počítačem a následně spustí systém Prometric. Tímto je uživateli znemožněno používat jakékoli elektronické dokumenty. Organizace Prometric také myslela na znemožnění používání jakýchkoliv jiných neelektronických dokumentů, případně mobilních telefonů apod., tím, že samotný průběh zkoušky může být nahráván na video či jinak kontrolován. S touto situací musí uživatel předem souhlasit podepsáním příslušného formuláře, jinak není připuštěn ke zkoušce.

Nevýhodou tohoto systému je náročná instalace a zprovoznění aplikace na počítači, potřeba mít vlastní server, kde bude spuštěna serverová část systému Prometric a datová náročnost na spojení s internetem a vzdáleným serverem Prometricu. Prometric načítá celý test ze vzdáleného serveru na lokální server a potom do klientské aplikace. Při vypracovávání otázek se průběžně odesílají zodpovězené otázky jak na lokální server tak i na vzdálený, což při větším výpadku může způsobit problém.

Tento systém však umožňuje spuštění i různých simulací během testu (jako příklad lze uvést Microsoft certifikaci, během níž je uživateli spuštěna simulace operačního systému Windows Server 2008 R2, ve které má za úkol nastavit Active Directory a spustit DHCP server).

1.2.3 LMS eDoceo

Learning Management System eDoceo je určen pro správu prezenčních a elektronických vzdělávacích programů. Lze jej provozovat v rámci intranetové sítě nebo internetu. Umožňuje testování, vyhodnocování, sledování výsledků studia, certifikování absolventů a schvalování procesů. Systém je možné propojit přímo s personálními databázemi nebo ERP systémy.

EDoceo je vyvíjen na technologiích Java, J2EE a XML firmou Trask solution s.r.o. Celý systém doplňují desktopové aplikace Autor a Off-line Student, které umožňují práci či studium bez nutnosti být připojen v daný moment k internetu (čerpáno z [2]).

2 POUŽITÉ TECHNOLOGIE

2.1 Microsoft Visual Studio 2010

Microsoft Visual Studio je vývojové prostředí pro aplikace od firmy Microsoft. Má aplikace byla vytvářena právě v tomto vývojovém prostředí, které mi svými vlastnostmi ulehčilo spoustu práce.

Ve Visual Studiu je možné vytvářet konzolové aplikace i aplikace s grafickým uživatelským prostředím, například s Windows Forms nebo pomocí Windows Presentation Foundation. Visual Studio má velké možnosti rozšiřitelnosti pomocí zásuvných modulů, tzv. pluginů, a případně pomocí šablon vytvářených projektů.

Hlavní prvky, které Microsoft Visual Studio podporuje, jsou editor kódu, debugger (pro ladění vyvíjené aplikace) a designér (pro návrh formulářů nebo jiných grafických objektů). Editor kódu podporuje tzv. „IntelliSense“. Jedná se o automatické doplňování psaného zdrojového kódu. Tato funkce Visual Studia programátorovi usnadňuje spoustu práce. IntelliSense doplňuje názvy proměnných, funkcí a metod jednotlivých tříd, které jsou z daného místa programově dostupné (například zobrazuje pouze veřejné proměnné a funkce jiných tříd, privátní proměnné a funkce programátorovi vůbec nenabídne, pokud nejsou součástí stejné třídy, kde je právě psán zdrojový kód). Další velmi užitečná vlastnost editoru kódu je podpora refaktorování. Jedná se o proces provádění změn ve zdrojovém kódu tak, že výsledné změny nemají vliv na vnější chování kódu. Používá se pro zpřehlednění zdrojového kódu. V praxi to znamená především to, že pokud je třeba změnit název nějaké proměnné, která je použita na více místech zdrojového kódu, tak není nutné ručně procházet celý kód a hledat všechna místa kde byla proměnná použita, ale refaktORIZACE tuto činnost udělá za programátora. Tímto se eliminuje případné vnášení chyb do zdrojového kódu a následně jejich složité hledání.

Visual Studio podporuje velké množství programovacích jazyků. Mezi hlavní vestavěné jazyky patří C/C++, VB.NET a C#. Podpora dalších jazyků může být přidána pomocí jazykových služeb, které musí být doinstalovány.

Microsoft Visual Studio 2010 je v rámci programu Microsoft Academy pro studenty zdarma, což je velkou výhodou tohoto vývojového prostředí. Toto je přínosem zejména při vyvíjení aplikací pro nekomerční využití.

2.2 Windows Presentation Foundation

WPF (Windows Presentation Foundation) je jedno z rozšíření rozhraní .NET Framework. Jedná se o knihovnu pro tvorbu grafického uživatelského rozhraní aplikací. Dříve bylo využíváno ovládacích prvků Windows Forms, které byly nativními ovládacími prvky systému Windows. Používali popisovače okna, které byly založené na pixelech.

WPF je založeno na rozhraní DirectX. Tato aplikace uživatelského rozhraní popisovače okna nepoživá a velmi snadno se s ním může měnit velikost uživatelského rozhraní (přizpůsobuje se velikosti aplikačního okna) a podporuje zvuk a video.

Jednou z velkých výhod rozhraní WPF je, že umožňuje snadno rozdělit práci na projektu mezi více lidí a mezi návrháře a vývojáře. Vývojář může pracovat na kódu, aniž by potřeboval hotové uživatelské rozhraní a návrhář naopak nepotřebuje při vývoji funkční a datovou část aplikace (čerpáno z [3]).

2.2.1 Jazyk XAML

XAML (Extensible Application Markup Language) je jazyk podobný XML, který je využit při návrhu aplikace ve WPF. Používá se pro definování rozvržení ovládacích prvků ve formuláři. Je to silný nástroj, který umožňuje připravit celé rozhraní aplikace i bez použití designové návrháře Visual Studio. Tato vlastnost může být užitečná ve chvílích, kdy nemáte v Visual Studiu přístup (čerpáno z [4]).

2.3 Programovací jazyk C#

Jedná se o objektově orientovaný programovací jazyk. Tento jazyk vyvinula firma Microsoft společně s platformou .NET Framework. Jazyk C# je založen na jazyce C++.

Programovací jazyk C# lze použít pro tvorbu konzolových aplikací, formulářových aplikací ve Windows (Windows Forms, Windows Presentation Foundation), webových aplikací a stránek, databázových programu a software pro mobilní zařízení jakou jsou PDA či mobilní telefony (adaptováno z [5]).

Tento programovací jazyk jsem použil pro vytvoření aplikace na testování studentů

2.4 ADO.NET Entity Framework

„ADO.NET Entity Framework je objektově relační mapovací prostředí, které vychází z platformy .NET 3.5. [...]

ADO.NET Entity Framework nabízí zobrazení ze schématu relační databáze na objekty. Relační databáze a objektově orientované jazyky definují vzájemné vztahy odlišným způsobem. Například ukázková databáze od Microsoftu s názvem Northwind obsahuje tabulky Customers a Orders. Chcete-li získat řádky objednávek pro určitého zákazníka, musíte provést v SQL vnitřní spojení tabulek. V objektově orientovaných jazycích je běžnější definovat třídy Customer a Order a přistupovat k objednávkám od zákazníka pomocí vlastnosti Orders třídy Customer.[...]

ADO.NET Entity Framework používá entity SQL k vytváření dotazů do uložiště založených na entitách. [...] Prostředí objektů uchovává stav měněných entit, a proto jsou k dispozici potřebné informace ve chvíli, kdy je potřeba zapsat entity zpět do uložiště. [...]

ADO.NET Entity Framework nabízí několik vrstev pro zobrazování databázových tabulek na objekty [...]:

- *Logická – tato vrstva definuje relační data,*
- *Konceptuální – tato vrstva definuje třídy .NET,*
- *Mapovací – tato vrstva definuje vzájemné zobrazení mezi třídami .NET a relačními tabulkami a vztahy*

(převzato z [4]).

2.5 Microsoft SQL Server 2008 R2

Microsoft SQL Server 2008 R2 je relační databázový systém vyvinutý společností Microsoft. Zajišťuje „rozhraní“ mezi datovým uložištěm databáze a aplikacemi. Zpracovává dotazy v jazyce SQL, které přesně definují, jaká data chce aplikace, která vyvolala SQL dotaz, získat.

Pro vytváření, editaci a spravování databáze je možné využít grafického uživatelského rozhraní SQL Server 2008 Management Studio.

2.5.1 SQL jazyk

Jedná se o strukturovaný dotazovací jazyk používaný pro práci s daty v relačních databázích.

II. PRAKTICKÁ ČÁST

3 HLAVNÍ MYŠLENKA A PŘEDSTAVA ŘEŠENÍ

Cílem bakalářské práce bylo vytvořit aplikaci pro testování znalostí studentů, která by zjednodušila lektorům a zkoušejícím práci jak při přípravě testu, tak i při samotném zkoušení. Navíc díky tomu, že zkoušení studentů bude probíhat elektronicky, nebude potřeba testy pro studenty složitě tisknout a poté archivovat.

3.1 Požadavky na aplikaci

Finální verze aplikace by měla být rozdělena do dvou projektů. Prvním projektem je lektorské rozhraní pro vyučující a zkoušející, ve kterém by mohli připravovat otázky do testů, vypisovat nové termíny testů a kontrolovat výsledky studentů.

Druhým projektem je studentské rozhraní, které bude obsahovat pouze zobrazení dostupných testů a rozhraní pro vypracování vybraného testu.

V celé aplikaci také musí být zajištěna autentizace uživatelů, aby bylo možné přidělit výsledky konkrétním studentům nebo odlišit lektory od studentů (pokud by lektorská verze ověřovala, zda je přihlašovaný uživatel opravdu lektor).

Součástí práce musí být velmi podrobný uživatelský manuál, pokud by byla aplikace nasazena do prostředí nezkušených uživatelů v oblasti IT.

3.1.1 Testy z více předmětů

Požadavkem na aplikaci bylo také umožnění testování studentů z více různých předmětů. To znamená, aby aplikace, respektive databáze na pozadí, nebyla zaměřena pouze na jeden vyučovaný předmět, ale aby bylo možné testy z jednotlivých předmětů v aplikaci snadno rozlišit. Tím se zajistí vyšší využitelnost celého testovacího systému.

3.1.2 Znemožnění podvádění během vypracování testu

Hlavním požadavkem na celý systém bylo znemožnění studentům při testu jakkoliv podvádět, alespoň, co se elektronických materiálů týká. Aby si tedy studenti na pozadí aplikace s testem nemohli otevřít dokument s výpisky z předmětu nebo dokonce dokumenty obsahující celé přednášky. Zabránit použití tištěných materiálů by měli zajistit přítomní zkoušející přímo během testu vizuální kontrolou studentů. V lektorské verzi aplikace toto samozřejmě není vyžadováno.

4 NÁVRH ŘEŠENÍ

4.1 Design a Windows Presentation Foundation

Při tvorbě grafického uživatelského prostředí bude využito WPF. WPF má oproti staré metodě, která využívá Windows Forms, hned několik výhod. Asi nejpodstatnější je, že vykreslování aplikace přechází oproti WF z procesoru přímo na čip grafické karty. Tím se značně sníží zátěž procesoru a zbyde více strojového času na ostatní rutiny.

WPF také umožňuje snadnou lokalizaci aplikace a skinovatelnost (změnu vzhledu, vytváření šablon vzhledu).

Během implementace WPF umožňuje tzv. bindování, což znamená provázání ovládacích prvků z grafického uživatelského rozhraní přímo na proměnné.

4.1.1 Studentské rozhraní aplikace

Při návrhu aplikace se musel brát ohled na co nejintuitivnější ovládání. Cílem bylo docílit toho, aby se student mohl soustředit především na test.

Aby bylo možné studenty při zobrazování výsledků rozpoznat, musí se student nejdříve registrovat. Pokud registrace proběhne úspěšně, tedy pokud student zadá všechny povinné údaje v registračním formuláři korektně a pokud nebude v databázi registrován uživatel se stejným uživatelským jménem, bude student neprodleně přihlášen a zobrazí se mu nabídka dostupných testů.

Ze seznamu by si student měl vybrat test a poté aplikace zobrazí rozhraní na vypracování testu. Součástí tohoto rozhraní bude seznam otázek v testu, tlačítka pro přechod mezi jednotlivými otázkami, text otázky a samozřejmě možné odpovědi k dané otázce.

Součástí celého studentského rozhraní také musí být informační panel, ve kterém bude vidět uživatelské jméno přihlášeného studenta a v průběhu testu zbývající čas do vynuceného ukončení testu. V tomto informačním panelu bude zobrazena adresa databázového serveru a konkrétní databáze, se kterou bude aplikace komunikovat. Tyto dvě položky bude možné před přihlášením studenta možné editovat. Informace o serveru a databázi budou defaultně uloženy přímo v aplikaci, pokud tato nastavení nebudou vyhovovat, bude možné aplikaci distribuovat s XML souborem obsahujícím alternativní nastavení. Pokud aplikace tento XML soubor najde, načte si právě tato nastavení.

Nejefektivnějším způsobem jak celou aplikaci navrhnout bylo naimplementování tzv. „Application Wizard“ pod čímž si lze představit například průvodce, který se používá při instalaci pokročilejších aplikací. Průvodce bude obsahovat právě tři jednoduché kroky: přihlášení/registraci, výběr testu a test. Ze strany studenta nebudou další kroky k ovládání aplikace nutné.

4.1.1.1 Zabezpečení proti podvádění při testu

Při návrhu zabezpečení studentské verze aplikace jsem se mohl vydat dvěma směry. První možností je pomocí jazyka C# se snažit zachytit všechny události operačního systému vyvolané klávesovými zkratkami nebo klikáním myši mimo spuštěnou aplikaci s běžícím testem. Druhou možností je využít vlastností hlavního okna aplikace, které máme přístupné díky WPF.

4.1.2 Lektorské rozhraní aplikace

Lektorské rozhraní aplikace bude řešeno podobně jako studentské rozhraní a to průvodcem. Vzhledem k tomu, že lektor bude muset mít k dispozici mnohem větší množství funkcí, bude i průvodce aplikací mnohem rozsáhlejší a bude nabízet více voleb. Lektorské rozhraní bude muset nabídnout přihlášení a registraci uživatele, vytváření skupin testů (například podle vyučovaného předmětu), v jednotlivých skupinách okruhy otázek, vytvořené testy a prostředí pro vytvoření testu. Nedílnou součástí musí být v jednotlivých okruzích vytváření otázek a odpovědí k nim. Případně by bylo možné implementovat editaci registrovaných uživatelů.

Stejně jako u studentské verze aplikace bude implementován informační panel, který bude mít stejné vlastnosti (možnost přenastavit server a databázi, ke které se bude aplikace připojovat a zobrazení uživatelského jména), s výjimkou odpočítávání času při testu.

Průvodce aplikací bude reprezentován pomocí záložek. Pracovní nástroje budou na záložky rozděleny podle tématu. Například na jedné záložce budou vypsány vyučované předměty, ze kterých je možné vytvořit test, na další budou testované okruhy ve skupině. Okruhy ve skupině znamenají, že bude možné rozdělit otázky ve vyučovaném předmětu například podle přednášek nebo podle vyučujících a jejich skupin studentů (aby si jednotliví lektori mohli udělat vlastní okruhy otázek, podle toho jaké informace studentům na přednáškách předali). Nedílnou součástí bude záložka pro vytváření a editaci otázek.

4.2 Databáze SQL a práce s daty (testy a uživatelé)

Aplikace (lektorská i studentská verze) bude zajišťovat pouze zobrazování dat v takové podobě, aby s nimi bylo možné pracovat – tedy zobrazování formulářů pro vytvoření otázek a testů nebo registrace nového uživatele a podobně. Veškerá práce s daty bude obstarána na straně databáze pomocí uložených SQL procedur – vytvoření otázek, vygenerování nového testu i vyhodnocení dokončených testů. Návrh celého modelu databáze je v příloze P1.

4.3 Propojení databáze s aplikací

Propojení databáze s aplikací bude zajištěno pomocí ADO.NET Entity Framework. Během implementace aplikace tato technologie umožní programátorovi pracovat s celou databází jako by se jednalo o objekty přímo v aplikaci. Celá datová vrstva na pozadí aplikace se tímto stává pro programátora zcela transparentní při implementaci uživatelského rozhraní.

EF také umožňuje mapování uložených procedur v databázi tak, že je bude možno využít tak, jako by se jednalo o metody jiné třídy.

5 OVLÁDÁNÍ APLIKACE – UŽIVATELSKÝ MANUÁL

5.1 Instalace a nastavení SQL serveru

5.1.1 Instalace aplikace a požadavky

Aplikace pro svůj bezchybný chod vyžaduje v počítači nainstalovaný Microsoft .NET Framework 4 Client Profile (x86 and x64), který novější operační systémy Microsoft Windows obsahují v základní instalaci (MS Windows Vista, MS Windows 7, MS Windows Server 2008). U starších operačních systémů Windows (XP) je potřeba tento balíček doinstalovat.

Aplikaci je možné na počítač nahrát dvěma způsoby:

- Zkopírováním spustitelného souboru – pokud je na počítači, kde aplikace poběží, nainstalováno vše potřebné, je možné zkopírovat a používat spustitelný soubor aplikace. Před kopírováním souborů si přečtěte kapitolu „Doporučení pro nasazení aplikace na testovací počítač“
- Nainstalováním z instalačních souborů – pokud si nejste jisti, zda na počítači máte nainstalováno vše potřebné, je možné využít instalačních souborů aplikace. Instalační průvodce automaticky zkontroluje a případně stáhne z internetu vše, co bude nainstalovaná aplikace potřebovat ke svému chodu. Po instalaci se aplikace automaticky spustí, případně ji bude možné spustit z hlavní nabídky „Start“ ve složce „Všechny programy/FEX/FEX_lektor“. Takto nainstalovaná aplikace si nastavení nutná k připojení k databázi pamatuje, ale nelze využít konfiguračního souboru. Před instalací aplikace si přečtěte kapitolu „Doporučení pro nasazení aplikace na testovací počítač“

Všechny potřebné soubory je možné nalézt na přiloženém CD.

5.1.2 Vytvoření databáze na SQL serveru

Pro vytvoření databáze na SQL serveru je možné využít SQL skript uložený na přiloženém CD ve složce „Database“, který vytvoří potřebné tabulky, procedury a uživatele pro správný chod aplikace.

SQL skript vytvoří nový přihlašovací účet k databázi („Login“), povolí vzdálené připojení k SQL serveru a SQL autentizaci, vytvoří novou databázi „FEX-database“ a potřebné tabulky, pohledy, procedury a funkce.

5.1.3 Potřebná nastavení SQL serveru

Po dokončení SQL skriptu je NUTNÉ SQL server restartovat, kvůli uložení změn a zkontrolovat jestli je zapnutý pro vzdálené připojení TCP/IP protokol. Postup:

- 1) Spustit „SQL Server Configuration Manager“
- 2) V levém stromu objektů zvolit „Protocols for (název serveru)“
- 3) Nastavit „TCP/IP“ na „Enabled“
- 4) Restartovat SQL server
- 5) Ve firewallu povolit port 1433 (standardní pro komunikaci s SQL serverem)

5.1.4 Doporučení pro nasazení aplikace na testovací počítače

- Pokud si nejste jisti, zda máte nainstalované v operačním systému všechny potřebné balíčky a aplikace, využijte instalačních souborů, které vše potřebné stáhnou z internetu a nainstalují.
- Poté používejte klasických spustitelných souborů – výhoda přenosného konfiguračního souboru s aplikací.
- Pokud využíváte spustitelných souborů, je vhodné mít aplikaci s konfiguračním souborem skrytou někde na disku (například na disku C: ve složce „Program Files“) a na ploše mít pouze zástupce spustitelného souboru.
- Lektorskou verzi aplikace nedávat k dispozici studentům a ani ji neinstalovat/nekopírovat na počítače, na kterých studenti budou pracovat.

5.2 Spuštění aplikace

Aplikaci pustíme standardním způsobem, jako jakoukoliv jinou aplikaci v operačním systému Windows. Celé testovací rozhraní se skládá ze dvou aplikací – verze pro lektora a verze pro studenty, kteří mají být testováni. Aplikace jsou rozlišené podle jejich názvu.

5.2.1 Název aplikace

Aplikace určená pro studenty se jmenuje „FEX_student“ a pomocí ní je možné se do systému pouze přihlásit, vybrat si jeden z dostupných (odemknutých) testů a vypracovat jej. Do této aplikace se může přihlásit i uživatel s lektorskými právy, například aby

zkontroloval, jestli je nově vytvořený a odemknutý test opravdu vidět mezi dostupnými testy nebo aby mohl test přímo vyzkoušet.

Aplikace s lektorským rozhraním je pojmenována jako „FEX_lector“ a obsahuje všechny nástroje, které jsou potřeba pro správu testů a testových otázek.

„FEX“ je zkratka z anglického slovního spojení „Final Exam“, což v překladu znamená závěrečná zkouška, avšak aplikaci lze



spolehlivě využít i při půlsemestrálních testech nebo při testech na praktických cvičeních z daného předmětu.

Obrázek 1 –

Spustitelné soubory

Ikonou aplikace je červený terč, symbolizuje zaměření se na znalosti (Obrázek 1 – Spustitelné soubory).

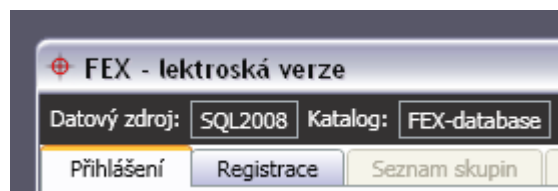
5.3 Nastavení připojení k databázi s testy

Lektorská i studentská aplikace se při přihlašování připojují k databázi, kde jsou uloženy všechny informace o testech, otázkách a uživateli. Adresa serveru a volba databáze uložené na něm není v aplikaci zakomponována na pevně, tudíž je možné tyto hodnoty nastavovat libovolně. To umožňuje aplikaci využívat rozsáhleji, například pro každý ústav na fakultě jiná databáze.

5.3.1 Za běhu aplikace

Po spuštění aplikace je možné nastavení pro připojení k databázi měnit v horním informačním panelu v levé části (Obrázek 2 – Nastavení připojení). Do datového zdroje se vepisuje název serveru, na kterém běží databáze a do katalogu se píše přímo název počítače.

Pokud informace nejsou správně zadané, aplikaci se nepodaří připojit a upozorní na chybu v zadání těchto nastavení. Aplikace také sama rozpozná, jestli je zvolen správný katalog a obsahuje opravdu data, která s aplikacemi FEX souvisí.



Obrázek 2 – Nastavení připojení

Po přihlášení už tyto informace není možné měnit a aplikace si po úspěšném přihlášení/registraci tyto údaje uloží do aplikačních nastavení do konfiguračního souboru (viz níže).

V základním nastavení aplikace po instalaci je katalog předvyplněný základní hodnotou (aby nebylo potřeba tuto hodnotu měnit, když se dodrží doporučená pravidla pro vytvoření databáze).

5.3.2 Konfiguračním souborem

Informace o připojení k databázi lze uložit do konfiguračního souboru aplikace. Soubor je možné najít na místním disku, ve složce s uživatelskými daty aplikací, standardně:

```
1 C:\Documents and Settings\Uživatel\Local Settings\Data aplikací
```

Data v konfiguračním souboru jsou v XML formátu. Pasáže, které jsou důležité pro připojování k databázovému serveru, jsou v druhé části v „userSettings“ a to konkrétně:

```
1 <setting name="Source" serializeAs="String">
2 <value>vbox-vs2010\sqlxpress</value>
3 </setting>
4 <setting name="Catalog" serializeAs="String">
5 <value>bctest</value>
6 </setting>
```

kde položka hodnota položky „Source“ je název, pod kterým se můžeme připojit k databázi a „Catalog“ je konkrétní databáze na serveru.

Pro vytvoření konfiguračního souboru doporučuji aplikaci jednou spustit a provést přihlášení – soubor se po úspěšném přihlášení vygeneruje automaticky (pokud neexistuje, pokud existuje, tak jej aplikace přepíše), případně použít základní konfigurační soubor, který naleznete ve stejné složce jako spustitelný soubor.

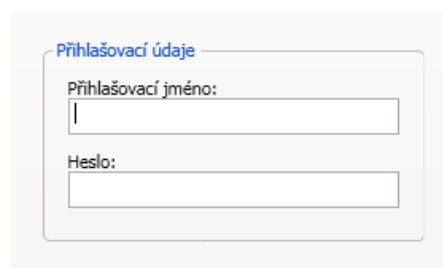
5.4 Studentské rozhraní

Studentská verze aplikace slouží pro samotné testování studentů. V této verzi jsou implementována opatření proti používání počítače pro jakýkoliv jiný účel, než je vypracování testu. Aplikace se spouští v režimu přes celou obrazovku.

5.4.1 Popis přihlášení a registrace

Po spuštění aplikace se zobrazí pole pro přihlašovací údaje (Obrázek 3 – Přihlašovací údaje). Uživatel, který je již registrovaný, vyplní uživatelské jméno a

heslo a přihlásí se do systému stisknutím tlačítka



Obrázek 3 – Přihlašovací údaje

„Přihlásit“. Pokud registrovaný ještě není, přejde na záložku „Registrace“ (Obrázek 4 – Registrační údaje) a vyplní registrační údaje podle pokynů. Po registraci je uživatel automaticky přihlášen a jeho uživatelské jméno se zobrazí v informačním panelu v pravém horním rohu.

Registrovanému uživateli je automaticky přidělena role „student“ a může se přihlásit pouze do studentské verze aplikace. Lektori se do této verze aplikace mohou přihlásit také, to je vhodné například pro kontrolu vygenerování testů.

Obrázek 4 – Registrační údaje

5.4.2 Výběr testu

Po přihlášení se uživateli zobrazí dva seznamy (Obrázek 5 – Výběr testu). V prvním (levém) je seznam předmětů, ve kterých se může vyskytovat test. Uživatel si vybere předmět, ze kterého má psát test a po kliknutí na něj se zobrazí v druhém seznamu (pravém) se zobrazí dostupné šablony testů. Po kliknutí na tlačítko „Spustit test“ se uživateli v databázi vygeneruje unikátní test podle pravidel, která zadal lektor při vytváření šablony testu.

Pokud má ale uživatel

v databázi

rozpracovaný

jakýkoliv test, tak tento krok po přihlášení aplikace automaticky přeskočí a zobrazí uživateli jeho rozpracovaný test i s odpověďmi, které již měl označené. Toto je ochrana proti neplánovanému ukončení aplikace ze strany operačního systému nebo chybou v aplikaci.

Obrázek 5 – Výběr testu

5.4.3 Vypracování testu

V dalším kroku se uživateli zobrazí vygenerovaný test. Seznam otázek je v levé části okna a uživatel v tomto seznamu vidí vždy celou otázku. Po výběru otázky (kliknutí na ní) se zobrazí celá otázka i s možnými odpověďmi v pravé části okna (Obrázek 6 – Vypracování testu).

Pokud se uživatel vrátí k otázce, kterou již vypracoval, tak se mu po načtení dané otázky označí odpověď, kterou dříve sám označil a případně může svou odpověď změnit.

Datový zdroj: vbox-vs2010\sqlexpress Katalog: FEX-database Přihlášený uživatel: superadmin

Přihlášení Registrace Výběr testu Test

Seznam otázek

Vytvořený seznam může být seřazen:

Co je to seznam?

Jakým způsobem Word použije záhlaví a zápatí?

Jakým způsobem lze použít Odrážky a číslování?

Jakým způsobem Word používá Automatické opravy?

K čemu slouží styly v programu MS Word

Jaký postup použijete, pokud chcete vyhledat část textu v rozsáhlém dokumentu?

Komentáře lze:

Které tři základní věci musíme mít nebo nastavit při vytváření hromadné korespondence

K čemu slouží volba "Konce" z karty "Rozložení stránky"?

V dialogu "Odstavec" je na kartě "Tok textu" zaškrtnutá políčko "Svázat s následujícím". K čemu slouží?

Jaké pole můžeme vložit do popisku obrázku, pokud chceme automaticky očíslovat obrázky?

Otázka s odpověďmi

V dialogu "Odstavec" je na kartě "Tok textu" zaškrtnutá políčko "Svázat s následujícím". K čemu slouží?

- ☒ Zařídí, že neskončí stránka mezi daným odstavcem a následujícím. Vhodné je to zejména pro nadpisy
- ☐ Spojí každý řádek odstavce s následujícím, takže bude celý odstavec na jedné stránce
- ☐ Zakazuje vložit za daný odstavec nadpis
- ☐ V popsaném dialogu žádná taková volba není

Ukonči Odeslat

Obrázek 6 – Vypracování testu

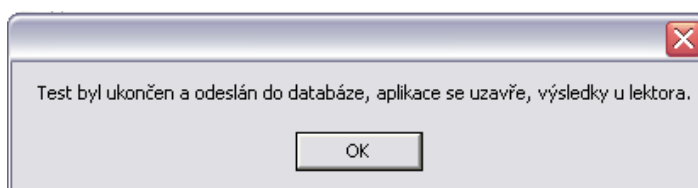
5.4.4 Odeslání vypracovaného testu

Po zodpovězení všech otázek uživatel klikne na tlačítko „Odeslat“ (Obrázek 6 – Vypracování testu). Tím je test označen za dokončený a uloží se v databázi. K ukončenému testu se již nelze vrátit. Po ukončení testu se zobrazí výsledek testu uživateli lektorovi v jeho aplikaci.

5.4.5 Ukončení práce se studentskou verzí aplikace

Jakmile uživatel vypracuje a odešle test, aplikace se automaticky ukončí (Obrázek 7 – Zpráva o odeslání testu).

Uživatel může práci kdykoliv ukončit stisknutím tlačítka „Ukonči“ v levém dolním rohu



(Obrázek 6 – Vypracování testu).

Obrázek 7 – Zpráva o odeslání testu

Při ukončování aplikace bude uživatel odhlášen. Toto tlačítko uživatel nesmí používat pro odeslání testu na server.

5.5 Lektorské rozhraní

Lektorská verze aplikace slouží pro kompletní správu otázek a šablon testů. Umožňuje třídit otázky podle skupin a podle okruhů. V tomto rozhraní lze prohlížet výsledky vypracovaných testů i s podrobným náhledem na jednotlivé testy. Je zde také dostupný nástroj pro přidělování a odebírání lektorské role registrovaným uživatelům.

5.5.1 Popis přihlášení a registrace

Přihlášení a registrace uživatelů jsou shodné se studentskou verzí, s tím rozdílem, že po registraci není uživatel automaticky přihlášen a uživatel potřebuje k přihlášení lektorská oprávnění. Nově registrovanému uživateli je vždy nastavena role „student“ bez ohledu na to, v které verzi aplikace se registroval. Oprávnění k přihlášení do lektorské verze (přirazení do role „lektor“) může udělit pouze jiný uživatel, který už v roli „lektor“ je a může se do lektorské verze aplikace přihlásit.

5.5.1.1 První uživatel v lektorské roli

Když se registruje první uživatel, má také automaticky nastavenou roli „student“, po registraci je nutné se přihlásit následujícím uživatelským jménem a heslem:

Uživatelské jméno: superadmin

Heslo: adminadmin

a nastavit prvního registrovaného uživatele do lektorské role (viz níže).

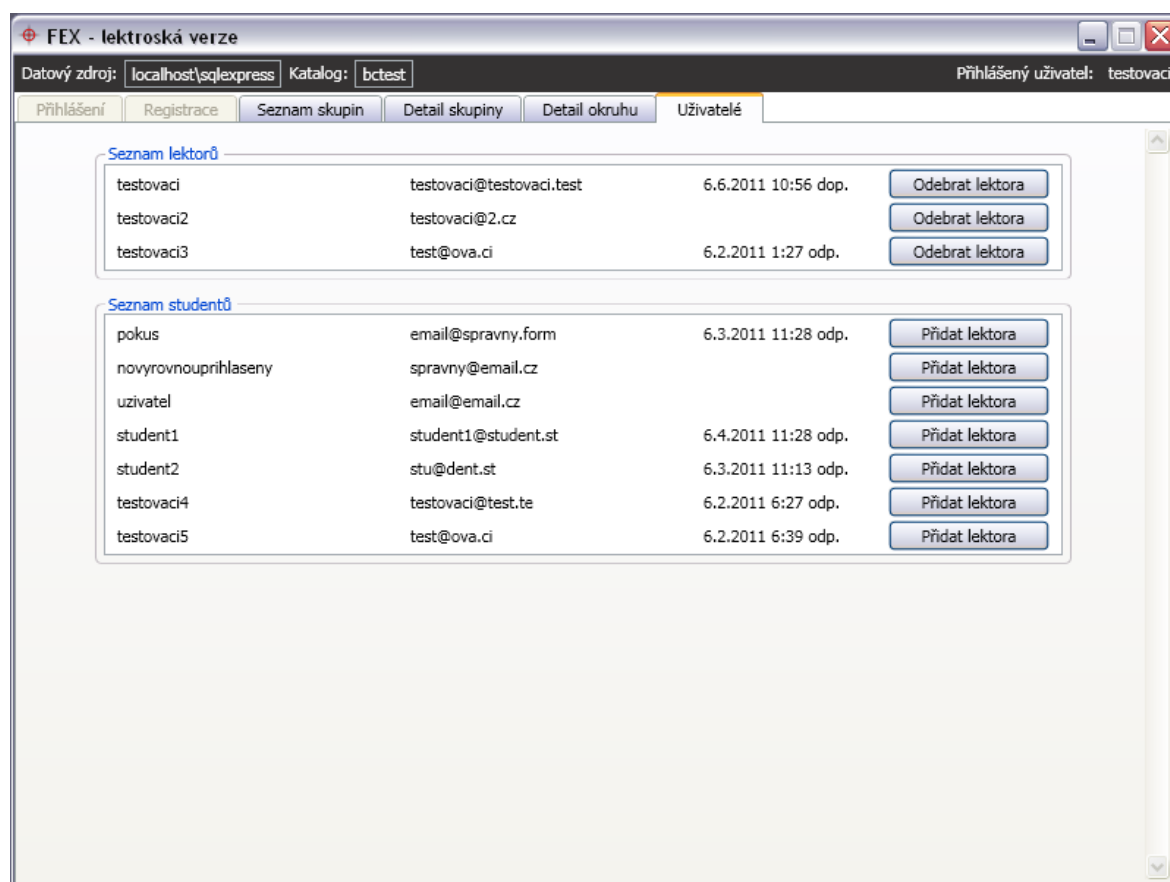
Z bezpečnostních důvodů je doporučeno uživateli „superadmin“ alespoň odebrat lektorská práva nebo úplně vymazat z databáze (pomocí aplikace „Microsoft SQL Server Management Studio“)

Pokud by si poslední z uživatelů v lektorské roli odebral lektorské práva, je možné přímo v databázi si tyto práva zpět přidělit v tabulce „UsersInRoles“ a do kolonky „RoleName“ vložit „lector“.

Nedávejte lektorská práva uživatelům, kteří je nepotřebují, aby nedošlo ke zbytečnému zneužití (například studenty).

5.5.1.2 Změna role registrovaného uživatele

Změnu role uživatele je možné provést na záložce „Uživatelé“. Na této kartě naleznete seznam uživatelů oddělených podle role. V seznamech se zobrazují uživatelské jména, emailové adresy uživatelů a čas jejich posledního přihlášení do některé z FEX aplikací. Pomocí tlačítek „Odebrat lektora“ a „Přidat lektora“ můžete měnit role uživatelů (Obrázek 8 – Správa uživatelů).



Obrázek 8 – Správa uživatelů

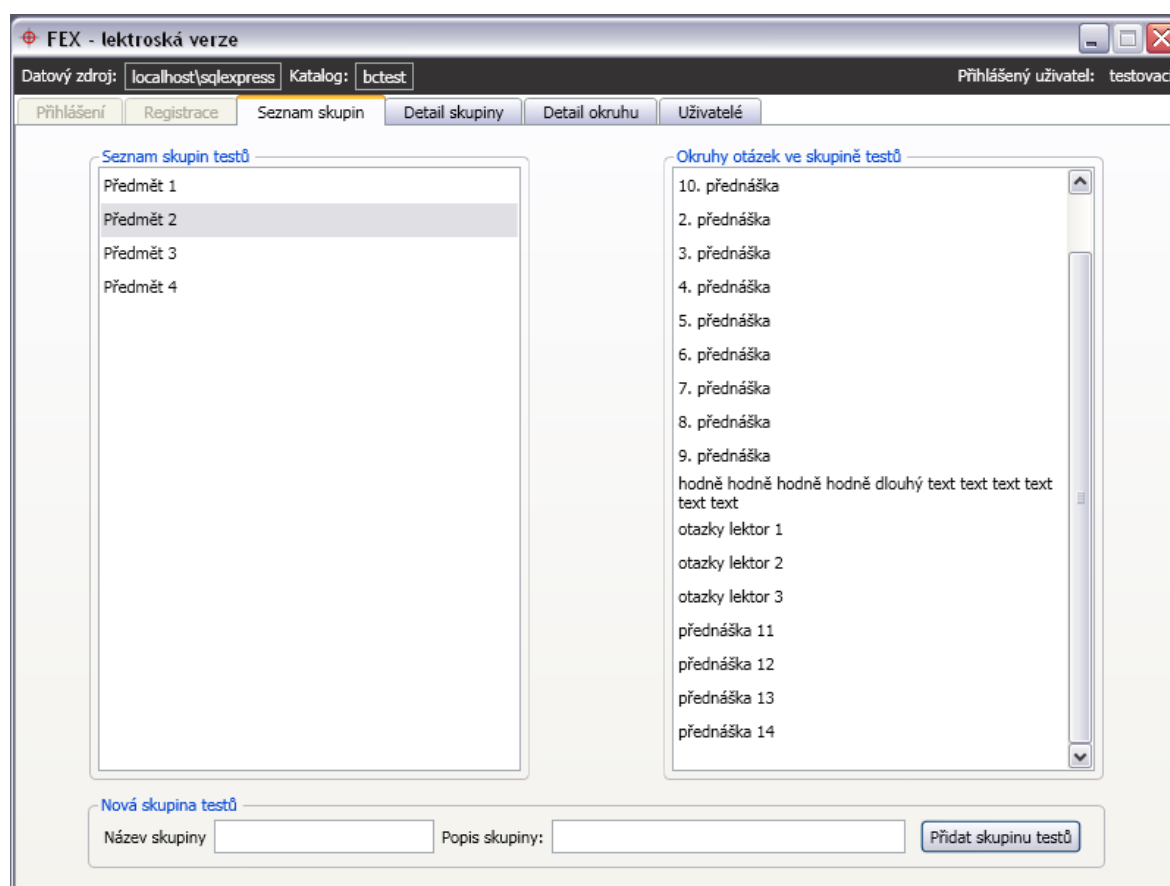
5.5.2 Vytvoření skupiny testů

Skupinou testů jsou zamýšleny testy stejného druhu (například rozdělení podle předmětů).

Na záložce „Seznam skupin“ (Obrázek 9 – Seznam skupin) jsou zobrazeny vytvořené skupiny a po zvolení konkrétní skupiny se v pravé části zobrazí okruhy otázek dané skupiny (viz níže).

Novou skupinu je možné přidat v dolní části karty vyplněním údajů o skupině a potvrzením stiskem tlačítka „Přidat skupinu testů“. Seznam skupin se automaticky aktualizuje a zobrazí novou skupinu.

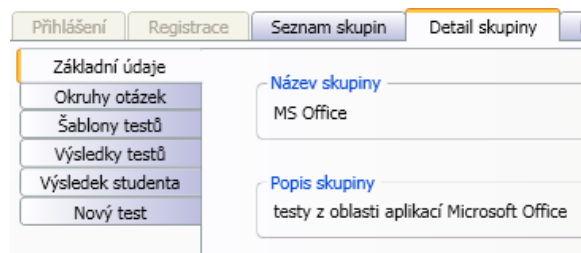
Po vybrání skupiny se zároveň do karty „Detail skupiny“ načítají informace o skupině a je možné na tuto kartu přejít. Stejnou funkci zastane také dvojité kliknutí myši na vybranou skupinu.



Obrázek 9 – Seznam skupin

5.5.3 Detail skupiny a okruhy otázek

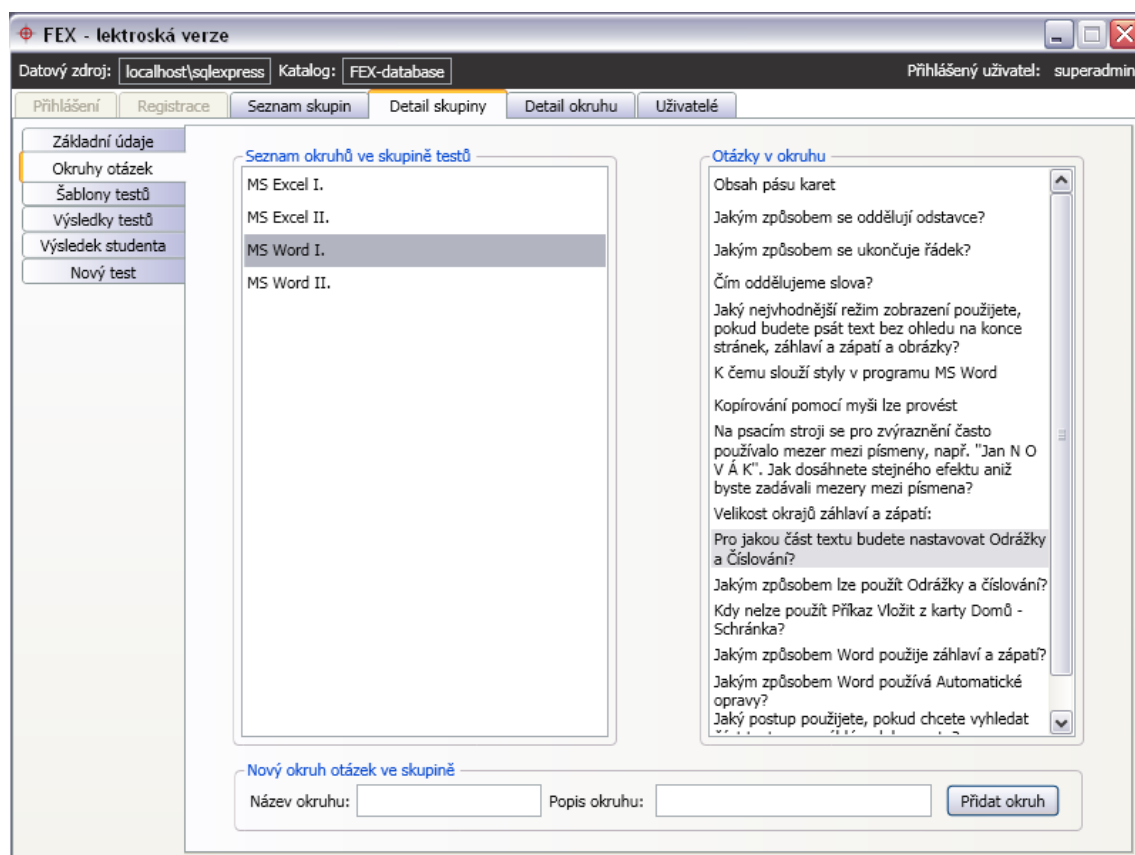
V jednotlivých skupinách lze vytvořené otázky rozdělit do tzv. okruhů (například podle přednášek, cvičení nebo každý vyučující v daném předmětu může mít svou sadu otázek pro své studenty).



Karta „Detail skupiny“ obsahuje menu s několika volbami:

Obrázek 10 – Základní údaje

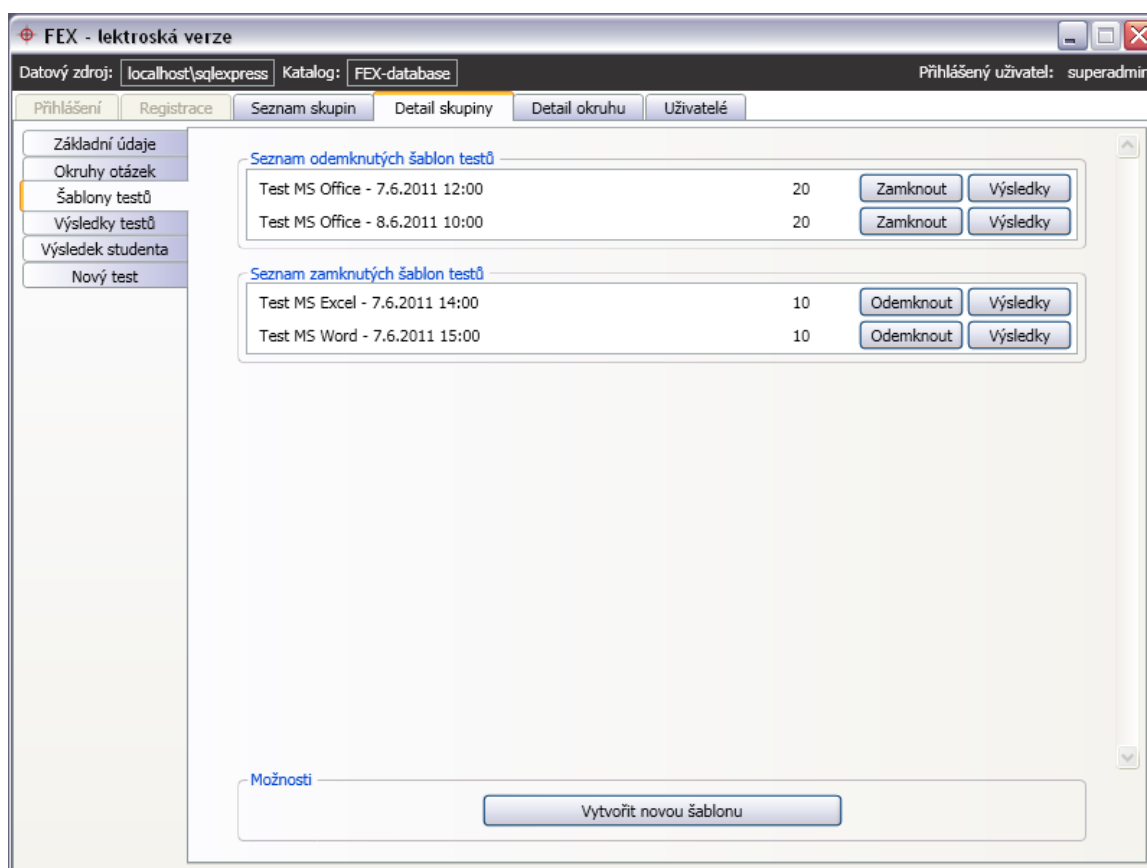
- Základní údaje – informace o vybrané skupině testů (Obrázek 10 – Základní údaje)
- Okruhy otázek – zobrazení okruhů otázek ve vybrané skupině. Po výběru okruhu se otázky související s tímto okruhem zobrazí v pravé části karty (Obrázek 11 – Okruhy otázek ve skupině). Nový okruh otázek je možné přidat stejným způsobem



Obrázek 11 – Okruhy otázek ve skupině

jako vytvoření nové skupiny testů ve spodní části karty. Po vybrání okruhu je možné přejít na kartu „Detail okruhu“ nebo po dvojitém kliknutí myši na okruh se automaticky zobrazí „Detail okruhu“, kde je možné přidávat, editovat a mazat otázky v okruhu (viz níže).

- Šablony testů – seznamy vytvořených šablon testů. Testy jsou rozděleny na odemknuté a zamknuté (Obrázek 12 – Šablony testů). Odemknuté jsou viditelné ve

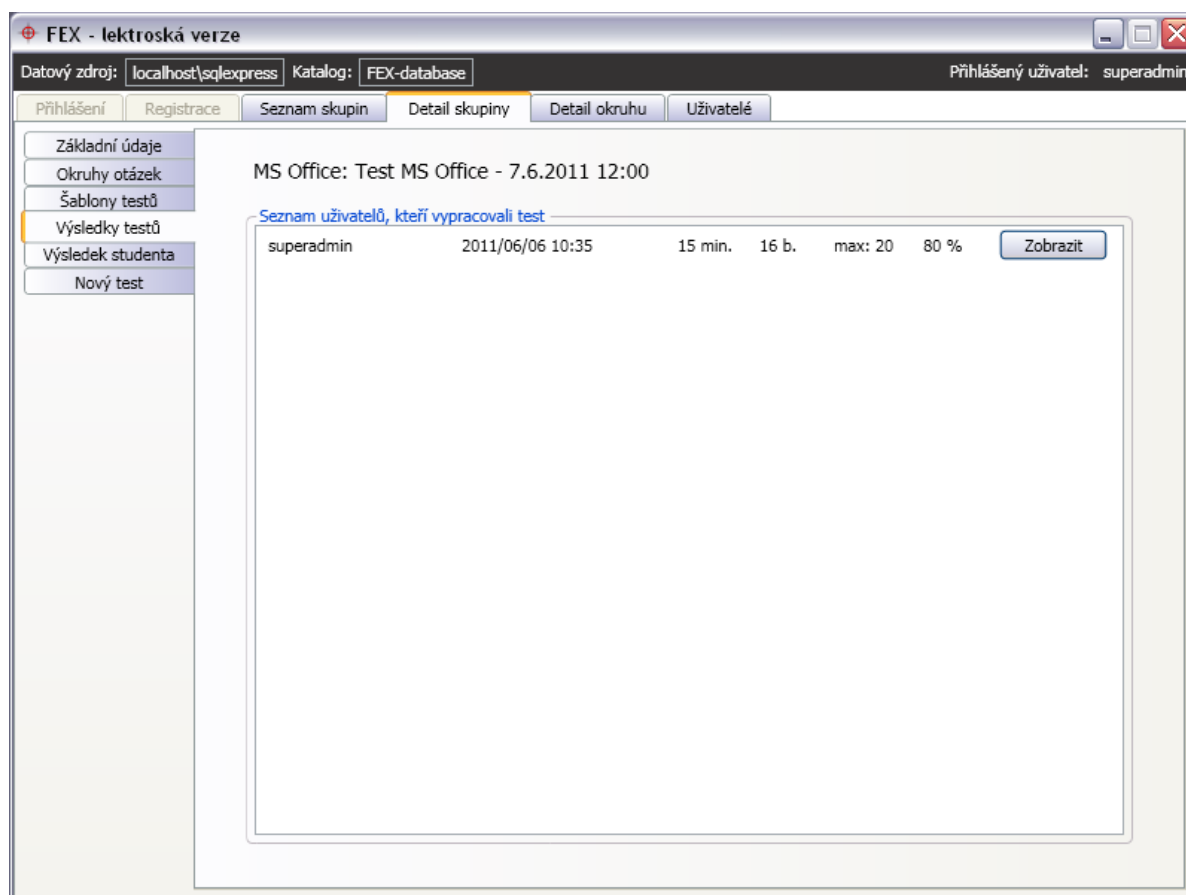


Obrázek 12 – Šablony testů

studentské verzi aplikace FEX. U každé šablony je zobrazen počet otázek, které se vygenerují při vytvoření testu z dané šablony. Šablony testů lze libovolně odemykat a zamykat pomocí tlačítek u každého testu. Po kliknutí na tlačítko „Výsledky“ se zobrazí souhrnné výsledky v daném testu (viz níže). Přidat novou šablonu testu je možné pomocí tlačítka „Vytvořit novou šablonu“ nebo po zvolení „Nový test“ v levém menu.

- Výsledky testů – ve výsledcích testů je zobrazen seznam všech uživatelů, kteří test vyplnili (Obrázek 13 – Výsledky testů). Seznam obsahuje základní informace o vypracovaných testech vygenerovaných z vybrané šablony (název skupiny a šablony je zobrazen nad výsledky) jako jsou uživatelská jména, čas dokončení

testu, doba trvání testu, počet získaných bodů, maximální možný počet bodů a procentuální úspěšnost v testu. Podrobné výsledky uživatele je možné zobrazit



Obrázek 13 – Výsledky testů

stisknutím tlačítka „Zobrazit“ a přepnutím se v menu aplikace na „Výsledek studenta“.

- Výsledek studenta – zobrazení velmi podrobného výsledku studenta, včetně všech otázek a odpovědí, které byly uživateli do testu vygenerovány (Obrázek 15 – Výsledek studenta). Nad otázkami je zobrazena souhrnná informace o výsledku v testu (Skupina: šablona testu – uživatel – Výsledek: získané body/maximální počet bodů). U každé otázky je zobrazeno její pořadí v testu, počet bodů za otázku a odpovědi jsou barevně odlišeny. V zobrazení odpovědí je zaškrtnuta ta odpověď, kterou zvolil uživatel při vypracování testu. Zelená barva označuje správně zodpovězenou otázku, červená naopak špatně zvolenou odpověď nebo odpověď, která byla správná a nebyla zvolena uživatelem. Ke každé otázce je zobrazeno také vysvětlení, pokud bylo nějaké vepsáno při vytváření otázky.

FEX - lektorská verze

Datový zdroj: localhost\\sqlexpress Katalog: FEX-database Přihlášený uživatel: superadmin

Přihlášení Registrace Seznam skupin Detail skupiny Detail okruhu Uživatelé

Základní údaje
Okruhy otázek
Šablony testů
Výsledky testů
Výsledek studenta
Nový test

MS Office: Test MS Office - 7.6.2011 12:00 - superadmin - Výsledek: 16/20 bodů

Otázky a odpovědi studenta

4 Je možné změnit počet listů v sešitu?

1/1b. ☒ Ano, kdykoliv během práce
☐ Ano, ale pouze s použitím průvodce
☐ Ano, ale pouze při zakládání nového sešitu
☐ Ne

Vysvětlení:

5 Máte tabulku se dvěma tisíci řádky. Chcete najít konkrétní hodnotu. Jak budete postupovat?

0/1b. ☐ Na kartě "Domů - Úpravy" zvolím příkaz Najít
☐ V nabídce "Tlačítka Office" zvolím příkaz Přejít na...
☐ Požadovanou hodnotu zapisuji do řádku vzorců, Excel se pokusí najít buňku s požadovanou hodnotou
☒ Budu listovat sešitem až najdu požadovanou hodnotu

Vysvětlení:

6 Pro vytvoření scénáře potřebuji:

1/1b. ☒ Existující výpočet v libovolné buňce a hodnotu, která bude dosazena do tohoto vzorce
☐ Pouze tabulku, hodnoty budou do scénáře dosazeny automaticky dle tabulky
☐ Tabulku s výpočty a hodnoty, které se budou dosazovat do těchto vzorců
☐ Kontingenční tabulku

Vysvětlení:

Obrázek 14 – Výsledek studenta

FEX - lektorská verze

Datový zdroj: localhost\\sqlexpress Katalog: FEX-database Přihlášený uživatel: superadmin

Přihlášení Registrace Seznam skupin Detail skupiny Detail okruhu Uživatelé

Základní údaje
Okruhy otázek
Šablony testů
Výsledky testů
Výsledek studenta
Nový test

Informace o testu

Název testu: MS Office pro mírně pokročilé- 24.6.2011 10:00

Popis testu: Závěrný test po skončení kurzu Microsoft Office pro začátečníky

Počet otázek z jednotlivých okruhů (maximální počet otázek v tématu za editačním polem)

MS Excel I.	10	(14)
MS Excel II.	0	(14)
MS Word I.	10	(15)
MS Word II.	0	(8)

Možnosti

Vytvořit test Vyprázdnit všechna pole

Obrázek 15 – Nový test

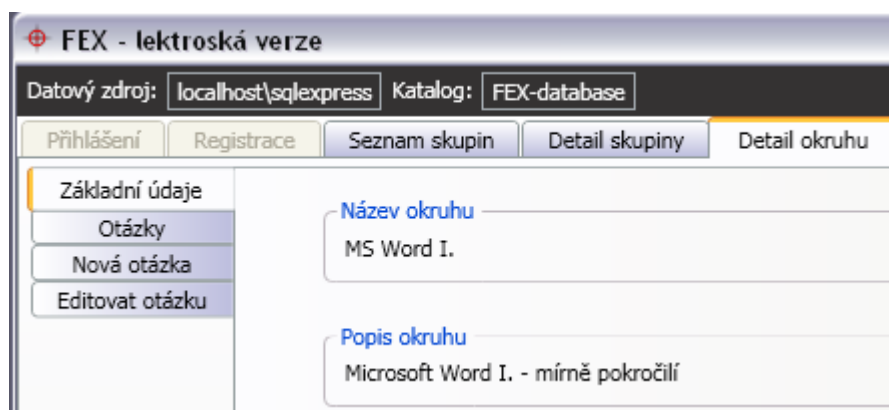
- Nový test – Vytvoří novou šablonu testu, ze které budou generovány testy. Je nutné zadat název testu, popis testu je volitelný (Obrázek 14 – Nový test). V seznamu

okruhů může lektor vybrat, kolik otázek bude ze kterého okruhu do cílového testu vygenerováno (maximální počet otázek je v závorce). To zajistí, že při dostatečném počtu otázek v okruhu nebudou vygenerovány dva stejné testy. Tlačítkem „Vytvořit test“ lektor vytvoří novou šablonu. Pokud by chtěl hodnoty ve formuláři vynulovat, může využít tlačítka „Vyprázdnit všechna pole“

5.5.4 Detail okruhu a otázky

V detailu okruhu skupiny je možné spravovat přímo otázky. V levém menu jsou dostupné tyto volby:

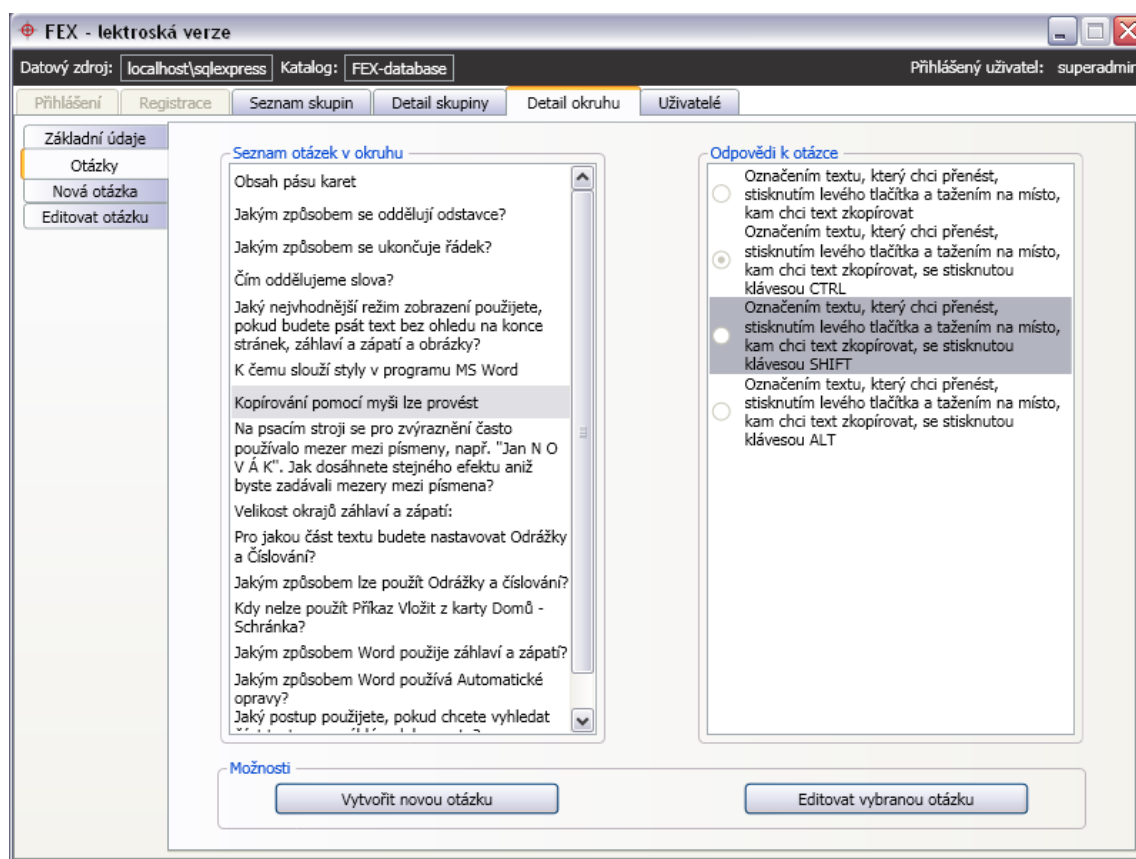
- Základní údaje – obsahují základní informace o okruhu



The screenshot shows a web application titled "FEX - lektorská verze". At the top, there are fields for "Datový zdroj:" (localhost\\sqlexpress) and "Katalog:" (FEX-database). Below these are several tabs: "Přihlášení", "Registrace", "Seznam skupin", "Detail skupiny", and "Detail okruhu". The "Detail okruhu" tab is active. On the left side of this tab, there is a vertical menu with four buttons: "Základní údaje" (highlighted), "Otázky", "Nová otázka", and "Editovat otázku". The main content area of the "Detail okruhu" tab contains two text input fields. The first is labeled "Název okruhu" and contains the text "MS Word I.". The second is labeled "Popis okruhu" and contains the text "Microsoft Word I. - mírně pokročilí".

Obrázek 16 – Základní údaje o okruhu otázek

- Otázky – seznam otázek v okruhu, po výběru jedné z otázek jsou v pravé části karty zobrazeny možné odpovědi na otázku a označená správná odpověď (Obrázek 17 – Seznam otázek). Po dvojitém kliknutí na otázku se zobrazí formulář pro editaci otázky (viz níže). Otázky je také možné mazat (například z důvodu, že se stanou neaktuálními) vybráním a stisknutím klávesy „Delete“.



Obrázek 17 – Seznam otázek

- Nová otázka – Formulář pro vytvoření nové otázky v okruhu. Lektor vyplní znění otázky a možných odpovědí, zvolí správnou odpověď a zvolí počet bodů, které student dostane za správné zodpovězení otázky. Volitelně může dopsat vysvětlení správné odpovědi, které se zobrazí ve výsledcích testu studenta. Po vyplnění uloží novou otázku stisknutím tlačítka „Uložit novou otázku“ nebo může všechna pole vymazat tlačítkem „Vymaž obsah všech položek“.
- Editovat otázku – formulář sloužící pro úpravu již existujících otázek. Pokud byla vybrána některá z existujících otázek nebo na ní bylo dvakrát kliknuto myší, načte se do tohoto formuláře (Obrázek 18 – Editace otázky). Změny v otázce se uloží po stisknutí tlačítka „Uložit změny otázky“. Pokud změny ukládat nechceme, zvolíme „Zrušit změny otázky“, čímž se zobrazí zpět seznam otázek. Případné změny otázky nijak neovlivní již vypracované testy, kde byla otázka použita. Z tohoto důvodu se do testu vždy vytváří kopie otázky.

The screenshot shows a web application window titled "FEX - lektorská verze". At the top, there are fields for "Datový zdroj: localhost\\sqlexpress" and "Katalog: FEX-database", and a user status "Přihlášený uživatel: superadmin". Below this is a navigation bar with tabs: "Přihlášení", "Registrace", "Seznam skupin", "Detail skupiny", "Detail okruhu", and "Uživatelé". On the left side, there is a sidebar with buttons: "Základní údaje", "Otázky", "Nová otázka", and "Editovat otázku". The main content area is titled "Parametry nové otázky" and contains the following fields:

- Otázka:** A text input field containing "K čemu slouží styly v programu MS Word".
- 1. odpověď:** A radio button followed by the text "Uchovávají celkový vzhled dokumentu, jako jsou rozměry stránky, okraje, atp."
- 2. odpověď:** A radio button followed by the text "Např. při psaní dopisu si můžeme vybrat, jaký styl dopisu budeme psát - osobní, formální, neform"
- 3. odpověď:** A radio button (which is selected) followed by the text "Uchovávají nastavení vlastností textu, jako jsou velikosti písmen, odsazení odstavců, atd."
- 4. odpověď:** A radio button followed by the text "Styly umožňují snadné vkládání často používaných textů (např. datum, čas, ...)"
- Vysvětlení:** A large text input field.
- Body:** A small input field containing the number "1".

At the bottom of the form, there is a section labeled "Možnosti" with two buttons: "Uložit změny otázky" and "Zrušit změny otázky".

Obrázek 18 – Editace otázky

5.5.5 Ukončení práce s lektorskou verzí aplikace

Po skončení činnosti v lektorské verzi aplikace je možné program ukončit ikonou pro zavření v pravém horním rohu nebo stiskem klávesové zkratky „Alt + F4“. Během ukončování je uživatel ze systému automaticky odhlášen.

Z bezpečnostních důvodů po dokončení své práce aplikaci vždy ukončujte a nenechávejte ji spuštěnou, aby k testovým otázkám a odpovědím neměly přístup nepovolané osoby.

6 IMPLEMENTACE APLIKACE

V kapitole číslo 6 se převážně věnuji stěžejním částem bakalářské práce, neboť popsání do detailu všech implementačních záležitostí by znamenalo z části se několikrát opakovat (protože jsou dvě aplikace založené na stejném principu) a z části by bylo zacházeno do záležitostí, které nejsou až tak podstatné z hlediska zadání práce.

6.1 Připojení k databázi

Připojení k databázi je zajištěno pomocí ADO.NET Entity Framework, jak už bylo zmíněno v návrhu aplikace.

Připojení k databázi je určeno „connection stringem“, což je řetězec, ve kterém jsou uloženy všechny důležité informace zajišťující spojení s databází (název serveru, databáze, ověření). Vytvoření tohoto řetězce zajišťuje metoda

```
7 private string MyConnectionStringBuilder(string source, string catalog)
```

její vstupní parametry jsou řetězce s adresou serveru ve formátu „jméno počítače\služba“ (například „VBOX-VS2010\SQLEXPRESS“ nebo „SQL2008SERVER“) a jméno databáze na serveru (příklad: „testyFAI“, v základním nastavení: „FEX-database“). Obě tyto hodnoty jsou získávány z TextBoxů¹ do kterých je možno vepisovat. Ostatní části connection stringu jsou již neměnné a metoda je doplní automaticky.

Do těchto textboxů jsou hodnoty při spuštění aplikace načítány z „Application settings“ kde mají nastavenou „Scope“ na hodnotu „User“. Když se aplikace spouští, zkontroluje nejdříve, jestli nemá dostupný XML soubor s údaji, který byl aplikací uložen dříve nebo byl k aplikaci nakopírován (kontroluje ve stejné složce, ve které se nachází spustitelný soubor aplikace). Pokud aplikace tento soubor nenalezne, použije základní hodnoty těchto proměnných.

Po úspěšném připojení aplikace nastavené hodnoty pro připojení k databázi ukládá do XML souboru, aby je bylo možné při příštím spuštění načíst a TextBoxy jsou označeny pouze pro čtení (po připojení k databázi už není potřeba tyto informace měnit). Uložení informací do XML souboru obstarává metoda

¹ TextBox Ovládací prvek WPF pro vepisování uživatelských dat

```
1 private void SaveApplicationSettings()
```

6.2 Registrace a přihlášení uživatelů

V aplikaci je implementována třída „User“, jejíž objekty jsou využity pro uchování informací o přihlášeném uživateli nebo pro přípravu registrace nového uživatele.

Při registraci nového uživatele je objekt typu „User“ nabídnutý do registračního formuláře. To zjednoduší a hlavně zpřehlední předávání údajů o registrujícím se uživateli databázi.

Uchovávat informace o přihlášeném uživateli je nutné kvůli rozpoznání uživatelů při činnosti v databázi. Tato informace může být užitečná pro rozlišení toho, který lektor vytvořil daný okruh otázek a nezbytná pro identifikaci který student, vypracoval který test.

6.2.1 Kontrola zadaných údajů při registraci a registrace

Kontrola registračních údajů je zajištěna validačními pravidly přímo spojených s TextBoxy. Validační pravidlo pomocí podmínek testuje zadanou hodnotu v TextBoxu. Pokud hodnota všemi podmínkami projde, znamená to, že je validní a validační pravidlo bude na dotaz metody třídy „Validation“:

```
1 Validation.GetHasError(registrationControl.txtUserName)
```

vracet informaci, že zadaná hodnota je v pořádku. Pokud hodnota v pořádku nebude, vrátí tuto skutečnost případně doplněnou o informaci, kterou částí kontroly hodnota neprošla.

Validační pravidla mohou ověřovat položky jako je neprázdný TextBox nebo emailová adresa zadaná ve správném formátu.

Validační pravidla aplikace jsou ve zdrojových kódech ve složce „ValidationRules“ a jsou to implementované třídy zděděné z třídy „ValidationRule“ a přepisují její metodu „Validate“.

Mimo validační pravidla se také ověřuje, jestli uživatelské jméno, které chce uživatel registrovat, už v databázi neexistuje.

Pokud je vše ve formuláři v pořádku, provede se zápis nového uživatele do databáze, což je díky použitému ADO.NET Entity Frameworku jednoduchá záležitost. Zápis uživatele do databáze zajišťuje tato metoda (nepodstatné části pro tuto ukázkou byly umazány):

```
1 private void DoRegistration(Users newUser, UsersInRoles  
    newUserInRole)  
2 {
```

```
3 db.Users.AddObject(newUser);  
4 db.UsersInRoles.AddObject(newUserInRole);  
5 db.SaveChanges();  
6 }
```

6.2.2 Autentizace uživatele při přihlášení

Přihlášení uživatele probíhá kontrolou existence uživatelského jména v databázi a kontrolou hesla.

6.2.3 Bezpečnost zadávaných hesel před odcizením

Uživatelská hesla je nutné přenášet přes internet a ukládat v databázi v šifrované podobě, aby se předešlo jejich zneužití. Zašifrování hesla (příloha P II) probíhá vypočítáním otisku (hashe) hesla algoritmem SHA512, který v dnešní době není prolomitelný.

Aby bylo zabráněno slovníkovému útoku (zkoušení náhodných slov ze slovníku hesel) je za heslo v nezašifrované podobě připojena tzv. sůl. Sůl je náhodně vygenerovaná posloupnost pěti znaků. Výsledné spojení hesla a soli je zašifrováno a uloženo v databázi. Ověření probíhá načtením soli z databáze, připojením za heslo, které uživatel zadá při přihlašování a porovnají se výsledné SHA512 otisky (hashe) hesel.

6.3 Vytváření skupin testů, okruhů otázek a otázek

Všechny výše uvedené činnosti jsou pouze přidáváním nových záznamů do databáze, podobně jako byla registrace uživatele.

6.4 Vytváření šablon testů

Při vytváření nové šablony testů je kontrolováno, aby v některém z okruhů nebylo zadáno více otázek než je skutečně dostupných nebo jestli uživatel nezadal do pole místo hodnoty nějaký text místo číslic. Jedná se pouze o šablonu, což znamená, že v této chvíli stále není vygenerovaný konkrétní test, ale pouze jakási pravidla určující, ze kterých okruhu budou otázky losovány a jejich počet.

6.5 Generování testů a kopie otázek

Samotný test, který má uživatel vyplňovat, je náhodně generován pomocí SQL uložené procedury až ve chvíli, kdy uživatel spustí test. Studentská verze aplikace tato data načte až ve chvíli, kdy je vygenerován do databáze nový záznam o testu. SQL procedura pro

vygenerování nového testu nevytváří odkazy na otázky v databázi, ale vytváří jejich kopie. Kdyby došlo později ke změnám v otázkách, musí být zajištěno, že to nijak neovlivní již vypracované testy, což vytvoření kopie otázek řeší spolehlivě. Databáze byla navržena důkladně, aby vyhovovala i případným budoucím rozšířením aplikace jako je variabilní počet otázek nebo otázky s 0 – n možných odpovědí.

Aplikace pro zavolání databázové SQL procedury využívá Microsoft Entity Frameworku, který umožní přístup k proceduře, jako by se jednalo o běžnou funkci v rámci třídy EF. Tato funkce má namapované vstupní parametry a vrací data v komplexním typu, který je výsledkem SQL dotazu v proceduře.

Příklad funkce, která obsahuje volání procedury pro vygenerování testu, viz příloha P III. Pokud se podaří v databázi vytvořit nový test, funkce vrací o této skutečnosti informaci a aplikace zobrazí okno s načteným testem. Test se načítá při zobrazení záložky, na které jsou umístěny prvky pro test. Prvky mají nastavený tzv. binding², takže pro načtení stačí do datacontextu³ přiřadit zdroj hodnot, v tomto případě tabulku s otázkami:

```
1 /// <summary>
2 /// Nacte testove otazky
3 /// </summary>
4 private void ReadTestQuestions()
5 {
6     listQuestions.DataContext =
        db.viewGetTestAttemptQuestionsView.Where
        (taID => taID.TestAttemptID == testAttemptID_session).ToList();
7 }
```

Jak je z příkladu vidět, pro získání tabulky z SQL databáze pomocí EF je možné využít metody „Where“ která je ekvivalentem pro stejný příkaz v jazyce SQL.

Do datacontextu přiřazují pouze pohled z databáze, protože se nejedná přímo o tabulku. Pohled je výsledkem SQL dotazu nad databází. Při přiřazování zdroje dat v tomto případě nebylo možné využít uložené SQL procedury na straně databázového serveru (viz kapitola 6.9 Komplikace při implementaci).

² Binding – provázání vlastnosti ovládacího prvku s proměnnou

³ Datacontextu – datový zdroj objektu, ve většině případů se jedná o tabulku

6.6 Vyhodnocování testů

Výsledek testu jako takový v databázi není uložen. Výsledek se počítá během každého zobrazení pro každou otázku zvlášť pomocí funkce v databázi a nakonec se udělá v proceduře pro výpočet výsledku suma všech bodů za otázky.

Funkce je předpřipravena na implementaci otázek typu více možných odpovědí.

Získání výsledku do aplikace je opět řešeno pomocí pohledu v databázi z důvodů popsaných v kapitole 6.9 Komplikace při implementaci.

6.7 Ukládání testů

Ukládání testů a hlavně jejich odpovědí je v databázi řešeno kopií původních testových otázek. Důvod k tomuto kroku byl popsán výše.

6.7.1 Průběžné ukládání – ochrana proti výpadku sítě

Aplikace při každé změně zobrazené otázky (událost „SelectedChanged“ seznamu otázek) odesílá pomocí namapované funkce:

```
1 db.uspUpdateTestAttemptAnswer(answerID, isChecked);
```

na SQL proceduru, která obsahuje:

```
1 PROCEDURE [dbo].[uspUpdateTestAttemptAnswer]
2   @TestAttemptQuestionAnswerID int, @UserAnswer bit
3 AS
4 BEGIN
5   SET NOCOUNT ON;
6   UPDATE TestAttemptQuestionAnswers SET UserAnswer = @UserAnswer
7     WHERE TestAttemptQuestionAnswerID = @TestAttemptQuestionAnswerID
8 END
```

aktualizaci všech odpovědí a jejich stavu (označené nebo neoznačené) předchozí otázky, která byla zobrazena. Až po tomto kroku je zobrazena nově vybraná otázka s odpověďmi.

Tato implementace řeší v aplikaci hned dva problémy:

- 1) Když se uživatel vrátí k již vypracované otázce, aby překontroloval svou volbu odpovědi, je možné mu načíst odpovědi i s volbou, kterou zvolil dříve a aplikace si nemusí držet v paměti celé pole otázek a odpovědí.
- 2) Pokud dojde k výpadku připojení k síti, uživatel nepřijde o svůj vypracovaný test a nemusí se obávat, že jsou jeho odpovědi uloženy pouze v lokálním počítači a

nebude je mít jak odeslat ke kontrole, protože minimálně ta část testu, která byla vypracována před výpadkem připojení je bezpečně uložena v databázi na serveru.

Tento způsob implementace odesílání a ukládání odpovědí na databázový server také umožňuje uživateli vrátit se k rozpracovanému testu. Pokud například dojde k pádu⁴ aplikace, uživatel může spustit aplikaci znovu a po přihlášení se mu zobrazí test ve fázi, ve které skončil (nebude zobrazena přesně otázka, u které byl naposledy, ale uživatelské odpovědi budou načteny).

6.7.1.1 Zneužití ochrany proti výpadku sítě

Pokud bude toto chování aplikace prozrazeno studentům, nebo na něj studenti přijdou při nějakém výpadku, může některého z nich napadnout tuto vlastnost aplikace zneužít (aplikaci ukončit, otevřít například skripta v elektronické podobě nebo vyhledat informace na internetu a aplikaci opět spustit). K jeho neštěstí je v události celého okna:

```
1 private void Window_Closing(object sender,  
    System.ComponentModel.CancelEventArgs e)  
2 {  
3     if (testAttemptID_session != 0)  
4     {  
5         db.uspCloseTest(testAttemptID_session);  
6     }  
7 }
```

volána přes EF SQL procedura, která označí test za dokončený. To znamená, že se uživatel už zpět k testu nedostane (aplikace načte pouze ten test, který není označen za dokončený). V tu chvíli zůstanou uživateli pouze dvě možnosti, skončit test úplně nebo si vygenerovat nový test s úplně novými otázkami. Pokud uživatel zvolí druhou variantu, zkoušející lektor poté uvidí dva různé vypracované testy se stejným uživatelským jménem. Důsledky, které z toho pro studenta budou plynout, už jsou pouze na lektorovi.

⁴ Pád aplikace – ukončení běhu aplikace, které může způsobit chyba ve zdrojovém kódu aplikace nebo chybovou událostí vyvolanou operačním systémem počítače.

6.8 Zabezpečení proti podvádění

Při snaze zabránit studentům v použití nepovolených elektronických materiálů bylo potřeba zajistit, aby testovací aplikace byla jediné místo, kde student může pracovat a nebylo možné se nikam jinam přepnout.

6.8.1 Odchycení klávesových zkratk pro přepínání mezi aplikacemi

První možností bylo odchytávat všechny události z operačního systému vyvolané stisknutím klávesových zkratk, případně kliknutí myši mimo testovací rozhraní a tyto události zpracovat testovací aplikací. Ovšem tato implementace má několik nevýhod. První je zvýšení náročnosti aplikace na výpočetní výkon procesoru a výkon operačního systému a druhou nevýhodou je skutečnost, že v operačním systému Windows bude vždy existovat klávesová zkratka, jejíž obsluhu nemůže vyřešit jakákoliv běžící aplikace. Touto klávesovou zkratkou je „Ctrl + Alt + Delete“ pro vyvolání správce úloh operačního systému, ve kterém už není obtížné se přepnout do jiného dokumentu nebo aplikace

6.8.2 Využití vlastností WPF oken

Druhou možností bylo využít vlastností oken vytvořených ve WPF. Správně využité kombinace několika vlastností okna zajistilo, že student sice může používat klávesové zkratky, ale z pohledu podvádění při vypracování testu se stejně nebude moci přepnout do jiné aplikace nebo dokumentu. Bylo využito následujících vlastností:

- `WindowState` – určuje, v jaké podobě bude aplikace spuštěná. Na výběr jsou možnosti normálně, minimalizovaná a maximalizovaná. Hodnota byla zvolena maximalizovaná.
- `TopMost` – nabývá pouze dvou hodnot, pravda a nepravda (true and false). Tato vlastnost byla nastavena na hodnotu „pravda“. To znamená, že aplikace bude stále nadvrchu a nepůjde ji ničím překrýt.
- `WindowStyle` – `WindowStyle` říká, jaký styl okna bude pro aplikaci použit. Při nastavené hodnotě na „None“ se nepoužije žádný styl = žádné okraje, žádné úchyty na změnu velikosti okna aplikace.
- `ResizeMode` – tato vlastnost určuje, jestli se oknu aplikace může měnit velikost a zároveň ovlivňuje zobrazení tlačítek pro minimalizaci a maximalizaci/obnovení z maximalizace v pravém horním rohu aplikace. Pokud je hodnota nastavena na

„NoResize“ je zajištěno, že okno aplikace nepůjde ani obnovit z maximalizace a ani minimalizovat či jinak změnit velikost.

6.9 Komplikace při implementaci

Během implementace jsem narazil na jednu komplikaci s využíváním Microsoft Entity Frameworku. Jednalo se o přiřazování výsledku SQL procedury, respektive jejího komplexního typu.

V aplikaci jsou využívány pro zobrazování seznamů převážně ovládací prvky „ListView“, kterým pomocí datatemplatu⁵ definuji, jak budou vypadat jednotlivé položky vytvořené na základě bindingu. U většiny těchto seznamů je odchytávána událost „SelectedChanged“⁶, která zaznamenává, že byl vybrán jiný řádek v seznamu. Při této události bylo načteno identifikační číslo prvku, který představoval prvek v databázi. Na základě tohoto identifikačního čísla byly do dalšího seznamu zobrazeny informace o vybraném prvku nebo související prvky z databáze.

Pokud byla zdrojem dat prvního seznamu prvků uložená SQL procedura, tak se z nepochopitelných důvodů přesouval „Focus“⁷ postupně na všechny nadřazené ovládací prvky, což vzhledem k implementaci mohlo mít za následek znovunačtení prvního seznamu a tím zrušení označení vybraného prvku. To v důsledku způsobilo, že nebyly zobrazeny žádné podrobné informace o vybraném prvku.

Tuto komplikaci vyřešilo namísto uložených SQL procedur použití pohledů v databázi v kombinaci s příkazem „Where“, který jednoznačně identifikoval prvek, který byl vybrán.

⁵ DataTemplate – vzorový formát ovládacích prvků, kterým se načítají hodnoty z bindingu

⁶ SelectedChanged – událost ovládacího prvku, který obsahuje objekty, které je možno označit a dojde k označení jiného prvku.

⁷ Focus – zaměření na ovládací prvek. Metoda, pomocí které lze aktivovat jiné ovládací prvky.

7 TESTOVÁNÍ ROZHRANÍ

Testování obou aplikací, jejich funkčností i jejich spojení s databází, nejdříve probíhalo pouze v rámci jednoho počítače s Windows autentizací na SQL serveru. To umožnilo snadnější testování při vývoji aplikace.

Druhá část testování, která měla za úkol ověřit volby připojení aplikace k různým serverům pomocí SQL autentizace, také probíhala stále na lokálním počítači. Došlo pouze ke změně autentizace na straně serveru a na straně aplikace se nevyužíval standartní „ConnectionString“⁸ ale vygenerovaný aplikací na základě vložených údajů uživatelem. Testování tímto způsobem probíhalo z důvodů hardwarových omezení.

V třetí a poslední fázi byla celá aplikace nasazena do reálného prostředí přímo na fyzickou síť a fyzický server. Po menších komplikacích s nastavením SQL serveru bylo vše důkladně otestováno. Tento test mi hlavně ukázal potřebu důkladně popsat instalaci databáze pro aplikaci při nasazení celého systému do ostrého provozu.

⁸ ConnectionString – řetězec znaků obsahující potřebné údaje k připojení ke vzdálenému SQL serveru.

8 DALŠÍ MOŽNÝ VÝVOJ

Další možné doplňky, které by bylo možné do celého návrhu rozhraní implementovat je například přesný odpočet času, po který by bylo možné test vypracovat. Tato funkce nebyla v první verzi implementována z důvodu vlastních zkušeností sledování vypracovávání testů studentů z pozice lektora.

Zajímavé rozšíření, pro které je už i databáze připravena, by byla možnost použít otázky, u kterých bude možno označit více správných odpovědí. Bylo by také možné použít otázky, ke kterým by student musel napsat odpověď sám, ale opravu této verze otázek by už nebylo možné automatizovat ze strany aplikace ani databáze.

Pro vývoj grafického uživatelského rozhraní bylo použito technologie WPF, což do budoucna zajišťuje možnost použití různých stylů pro vzhled celé aplikace. Lektor by si po té mohl volit vzhled aplikace podle aktuální situace (ve dne na přímém slunci použití vysokého jasu a kontrastních barev a naopak v přitnutí použití tmavých barev).

Dalším možným rozšířením je udělat do studentské i lektorské verze rozhraní, které by umožňovalo editaci údajů o sobě.

Z hlediska studentské verze a konkrétně při vypracovávání testu by mohlo být implementováno označení otázek, u kterých by si student nebyl jistý a chtěl by se k nim vrátit později. Podobně jako je v testovacím rozhraní firmy Prometric.

Posledním možným rozšířením, které by mohlo být při nasazení do praxe užitečné, je import a export otázek do XML souborů. Umožňovalo by to jednoduše přenášet testové otázky mezi systémy nebo otázky zálohovat.

ZÁVĚR

Hlavním cílem práce bylo vytvořit aplikaci pro využití v kurzech Microsoft IT Academy na UTB ve Zlíně na testování znalostí studentů. Obě vytvořené aplikace splňují požadavky, které na ně byly kladeny a jsou připraveny k nasazení do praxe.

Při implementaci studentské verze jsem se zaměřil na zabezpečení testovacího prostředí, aby studentům byla znemožněna jakákoliv jiná práce, než vypracování testu. Z hlediska elektronických dokumentů je tato část splněna, ale je důležité, aby zkoušející na studenty dohlížel – žádný program nikdy nedokáže ovlivnit, jestli student nemá u sebe vytištěná skripta, knihu nebo jiné poznámky ve fyzické formě.

Tato práce mi dala spoustu cenných zkušeností v oblasti vytváření uživatelských aplikací v jazyce C#, který je součástí Microsoft .NET Framework, vytváření návrhu databáze a její aplikace na Microsoft SQL serveru a vytváření uživatelských prostředí ve Windows Presentation Foundation. Věřím, že zkušenosti nasbírané během vývoje aplikace mi budou velkou a nezbytnou oporou pro budoucí práci v oblasti informačních technologií a programování aplikací.

ZÁVĚR V ANGLIČTINĚ

The main aim of the thesis was to create an application which could be used for testing of students' knowledge in the courses of Microsoft IT Academy at Tomas Bata University in Zlín (TBU Zlín). Both created application comply with all the requirements of the assignment and they both are ready to be put into practice.

During the implementation, the emphasis was laid on the security of the testing environment and on preventing test-takers from working on other applications than the test itself. From the point of view of the electronic documents, this part of task is fulfilled but the importance of the supervision from the examiner remains as it is impossible for any kind of program to eliminate the possibility of students having non-electronic based materials such as lesson notes or textbooks.

The thesis provided me with much valuable experience in the field of creating of user applications in the C# language, which is a part of Microsoft .NET Framework, creating database design and its application with Microsoft SQL server, and creating user interfaces in Windows Presentation Foundation. I believe that this knowledge gathered in the course of the development of the application will be of a great and indispensable help for my future work in the field of information technology and application programming.

SEZNAM POUŽITÉ LITERATURY

- [1] MOODLE.CZ [online]. 2003 [cit. 2011-05-28]. Dostupné z WWW: <<http://moodle.cz>>.
- [2] LMS eDoceo [online]. 2009 [cit. 2011-05-28]. Dostupné z WWW: <<http://www.edoceo.cz>>.
- [3] NAGEL, Christian, et al. C# 2008 : Programujeme profesionálně. Brno : Computer Press, 2009. 772 s. ISBN 978-80-251-2401-7.
- [4] SHARP, John. Microsoft Visual C# 2010 : krok za krokem. Brno : Computer Press, 2010. 696 s. ISBN 978-80-251-3147-3
- [5] GUNNERSON, Eric. Začínáme programovat v C#. Praha : Computer Press, 2001. 316 s. ISBN 80-7226-525-3.
- [6] NATHAN, Adam. Windows Presentation Foundation : Unleashed. Indiana, USA : Sams Publishing, 2007. 638 s. ISBN 0-672-32891-7.
- [7] Msdn [online]. © 2011 Microsoft Corporation. All rights reserved., 2011 [cit. 2011-05-28]. Msdn. Dostupné z WWW: <http://msdn.microsoft.com/en-us/default.aspx>
- [8] Stack Overflow [online]. 2011 [cit. 2011-06-06]. Dostupné z WWW: <<http://stackoverflow.com/>>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

WPF Windows Presentation Foundation

WF Windows Forms

EF Entity Framework

SQL Structured Query Language (strukturovaný dotazovací jazyk)

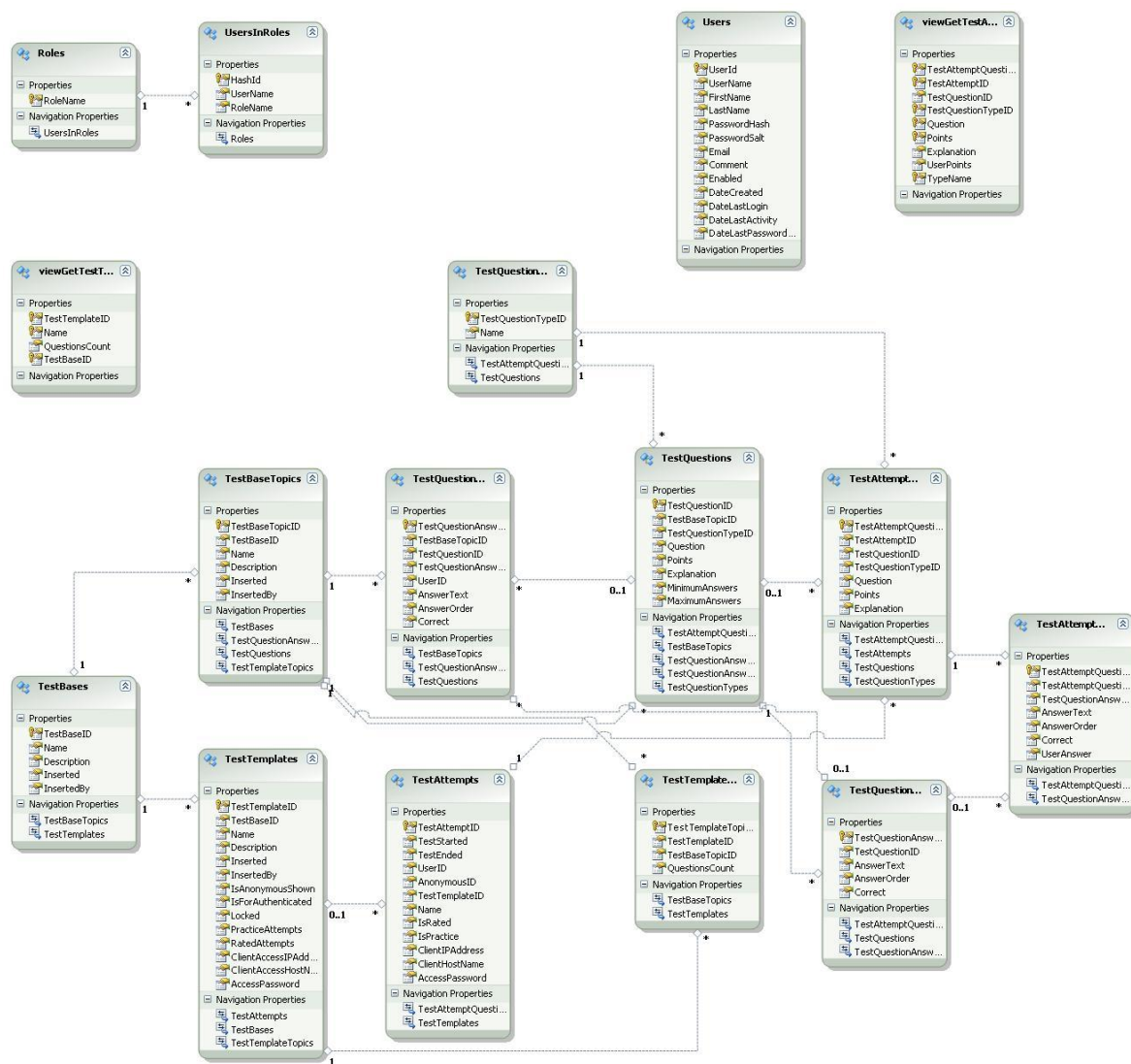
SEZNAM OBRÁZKŮ

Obrázek 1 – Spustitelné soubory	24
Obrázek 2 – Nastavení připojení.....	24
Obrázek 3 – Přihlašovací údaje.....	25
Obrázek 4 – Registrační údaje	26
Obrázek 5 – Výběr testu.....	26
Obrázek 6 – Vypracování testu.....	27
Obrázek 7 – Zpráva o odeslání testu.....	28
Obrázek 8 – Správa uživatelů	29
Obrázek 9 – Seznam skupin.....	30
Obrázek 10 – Základní údaje	31
Obrázek 11 – Okruhy otázek ve skupině	31
Obrázek 12 – Šablony testů	32
Obrázek 13 – Výsledky testů	33
Obrázek 14 – Nový test.....	34
Obrázek 15 – Výsledek studenta.....	34
Obrázek 16 – Základní údaje o okruhu otázek	35
Obrázek 17 – Seznam otázek.....	36
Obrázek 18 – Editace otázky	37

SEZNAM PŘÍLOH

- P I Model implementované databáze
- P II Šifrování hesla algoritmem SHA512
- P III Vygenerování nového testu

PŘÍLOHA P I: MODEL IMPLEMENTOVANÉ DATABÁZE



Poznámka: k podrobnějšímu nahlédnutí je digram přiložen na CD ve složce přílohy

PŘÍLOHA P II: ŠIFROVÁNÍ HESLA ALGORITMEM SHA512

```
private void FillNewUser(Users newUser, UsersInRoles newUserInRole)
{
    newUser.UserName = registrationControl.RegistrationUser.UserName;
    newUser.Email = registrationControl.RegistrationUser.Email;
    newUser.FirstName = registrationControl.RegistrationUser.FirstName;
    newUser.LastName = registrationControl.RegistrationUser.LastName;

    newUser.PasswordSalt = GenerateSalt();
    string pwdSalt = registrationControl.txtPassword.Password + newUser.PasswordSalt;
    newUser.PasswordHash = ComputeSHA512(pwdSalt);

    newUser.Enabled = true;
    newUser.DateCreated = DateTime.Now;
    newUser.DateLastPasswordChange = DateTime.Now;

    //pouze v lektorske verzi! jinak automaticky student
    newUserInRole.UserName = newUser.UserName;
    if (registrationControl.checkLector.IsChecked == true)
        newUserInRole.RoleName = "lector";
    else
        newUserInRole.RoleName = "student";
}

/// <summary> ...
private string ComputeSHA512(string pwdSalt)
{
    if (string.IsNullOrEmpty(pwdSalt)) throw new ArgumentNullException();
    byte[] buffer = System.Text.Encoding.UTF8.GetBytes(pwdSalt);
    buffer = System.Security.Cryptography.SHA512Managed.Create().ComputeHash(buffer);
    return System.Convert.ToBase64String(buffer).Substring(0, 86);
}

/// <summary> ...
private string GenerateSalt()
{
    StringBuilder salt = new StringBuilder();
    Random random = new Random();
    for (int i = 0; i < 5; i++)
    {
        var choose = random.Next(0, 2);
        switch (choose)
        {
            case 0:
                salt.Append(Convert.ToChar(
                    Convert.ToInt32(Math.Floor(26 * random.NextDouble() + 65)))
                );
                break;
            case 1:
                salt.Append(random.Next(1, 10));
                break;
        }
    }
    return salt.ToString();
}
```

PŘÍLOHA P III: VYGENEROVÁNÍ NOVÉHO TESTU

```
/// <summary>
/// Vygeneruje pro uzivatele v db unikatni test ze sablony
/// </summary>
private bool GenerateTest()
{
    int tTemplateID = ((uspGetAvailableTests_Result)listTests.SelectedItem).TestTemplateID;
    int tUserID = loggedUser.UserID;
    string tAnonymID = "";
    string clientHostName = System.Environment.MachineName;
    string clientIP = GetLocalIP();

    ObjectParameter tAttemptID = new ObjectParameter("TestAttemptID", typeof(int));
    ObjectParameter tStatus = new ObjectParameter("Status", typeof(int));

    db.uspGenerateTest(tTemplateID, tUserID, tAnonymID, clientHostName, clientIP, null, true, false, tAttemptID, tStatus);

    if (Convert.ToInt32(tStatus.Value) == 1)
    {
        testAttemptID_session = Convert.ToInt32(tAttemptID.Value);
        return true;
    }
    return false;
}
```