VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY
INSTITUTE OF INFORMATICS

NÁVRH APLIKACE PRO STŘEDNÍ PRŮMYSLOVOU ŠKOLU
PROPOSAL OF APPLICATON FOR HIGH TECHNICAL SCHOOL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Ondřej Bret

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Jan Luhan, Ph.D., MSc

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání diplomové práce

Ústav:	Ústav informatiky
Student:	**Bc. Ondřej Bret**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Informační management
Vedoucí práce:	**Ing. Jan Luhan, Ph.D., MSc**
Akademický rok:	2016/17

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává diplomovou práci s názvem:

Návrh aplikace pro střední průmyslovou školu

Charakteristika problematiky úkolu:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

Cíle, kterých má být dosaženo:

Cílem práce je navrhnout dílčí část IS v podobě aplikace pro střední průmyslovou školu ve Frýdku-Místku se zaměřením na studijní agendu a následnou archivaci údajů o studentech této školy.

Základní literární prameny:

BEGG, C., R. HOLOWCZAK a T. CONOLLY. Mistrovství - Databáze: Profesionální průvodce tvorbou efektivních databází. 1. vyd. Praha: Computer Press, 2009. 584 s. ISBN 978-80-251-2328-7.

BRUCKNER, T., J. VOŘÍŠEK, A. BUCHALCEVOVÁ a kol. Tvorba informačních systémů: Principy, metodiky, architektury. 1. vyd. Praha: Grada Publishing a.s., 2012. 360 s. ISBN 978-80-247-4153-6.

FUCHS, J. a A. BARCHFELD. Visual Basic Velká kniha řešení. 1. vyd. Brno: Computer Press, 2010. 744 s. ISBN 978-80-251-2212-9.

HALVORSON, M. Microsoft Visual Basic Krok za krokem. 1. vyd. Brno: Computer Press, 2015. 648 s. ISBN 978-80-251-4412-1.

SCHWALBE, K. Řízení projektů v IT: Kompletní průvodce. 1. vyd. Praha: Computer Press, 2011. 632 s. ISBN 978-80-251-2882-4.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2016/17.

V Brně, dne 28. 2. 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA PODNIKATELSKÁ

L. S.

doc. RNDr. Bedřich Půža, CSc.
ředitel

doc. Ing. et Ing. Stanislav Škapa, Ph.D.
děkan

Abstrakt

Obsahem této diplomové práce je návrh uživatelského rozhraní k SQL databázi pro Střední průmyslovou školu ve Frýdku-Místku. Práce je rozdělena na teoretickou část a praktickou část. V teoretické části jsou uvedeny potřebné informace a teorie nezbytná k úspěšnému splnění cíle práce. V praktické části je analyzována současná situace školy a je navrženo vlastní řešení daného problému.

Abstract

The content of this diploma thesis is to design user interface to SQL database for the Secondary Technical School in Frýdek-Místek. The work is divided into a theoretical part and a practical part. The theoretical part provides the necessary information and theories important for successful fulfilling the goal of the work. The practical part analyses the current situation of the school and offers a solution to the problem mentioned.

Klíčová slova

SQL, Visual basic, Visual Studio, databáze, škola, IS/ICT

Key words

SQL, Visual basic, Visual Studio, database, school, IS/ICT

Bibliografická citace práce

BRET, O. *Návrh aplikace pro střední průmyslovou školu.* Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2017. 71 s. Vedoucí diplomové práce Ing. Jan Luhan, Ph.D., MSc.

Čestné prohlášení

Prohlašuji, že předložená diplomová práce je původní a zpracoval jsem ji samostatně.

Prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb. o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 15. 5. 2017 ……

Podpis studenta

Poděkování

Rád bych tímto poděkoval vedoucímu diplomové práce panu Ing. Janu Luhanovi, Ph.D., MSc. za vstřícný přístup, cenné rady a připomínky, které mi pomohly při řešení této práce.

Dále bych rád poděkoval paní Ing. Jiřině Beňové za poskytnutí podkladů pro mou práci a její vstřícný přístup.

V neposlední řadě patří velké poděkování celé mé rodině za podporu při studiích.

Obsah

Úvod

Informační systémy a informační komunikační technologie (IS/ICT) jsou nepostradatelnou součástí dnešního světa. Značně ovlivňují způsob práce všech lidí v mnoha oborech a na všech úrovních lidské činnosti. Jedná se o odvětví, které se velice dynamicky rozvíjí. Informační technologie, jejich prodej, pořízení i využívání se neustále vyvíjí. Společnosti jsou ustavičně nuceny inovovat své informační systémy, tak aby udržely krok s konkurencí.

V současné době představují základní součást informačních systémů databáze, které od základu mění činnost mnoha společností i způsoby, jak pracují jednotlivci. Vývoj těchto technologií v posledních několika letech vedl ke vzniku výkonnějších databázových systémů, které lze intuitivněji ovládat.

Přechod na databázové systémy usnadňuje vyhledávání podstatných informací, zrychluje a celkově zefektivňuje práci s těmito informacemi. V českém školství je tento přechod na nové informační technologie zpomalován celkovým podfinancováním a absencí tržního prostředí, které tak nevyvíjí tlak na potřebu inovovat a rozvíjet tyto technologie.

1. Vymezení problému a cíle práce

Tato diplomová práce si klade za cíl navrhnout dílčí část informačního systému v podobě uživatelské aplikace pro Střední průmyslovou školu ve Frýdku-Místku se zaměřením na studijní agendu a archivaci údajů o studentech této školy.

Před samotnou analýzou problému a návrhu jeho řešení budou popsány nezbytné teoretické podklady, bez kterých by nebylo možno dosáhnout stanoveného cíle.

Následně v analytické části práce bude zhodnocena současná situace školy, popis současných prostředků IS/ICT školy a proces samotné archivace včetně legislativního prostředí. V této části bude také provedena analýza požadavků školy na výslednou aplikaci a rozbor SQL databáze, nad kterou aplikace poběží. Analýza současného stavu je pro návrh aplikace klíčová, protože by se v průběhu vývoje mohly vyskytnout obtížně řešitelné situace. Těmto situacím lze předejít právě pečlivou analýzou, tak aby byly identifikovány možné problémy a aby v budoucnu nebylo komplikované jejich řešení.

V praktické části bude popsán nejdříve grafický návrh aplikace a slovní popis jednotlivých procesů. Poté dojde k vytvoření vývojových diagramů a na těchto základech bude aplikace vyvíjena. Následně bude popsáno samotné uživatelské rozhraní aplikace. Nakonec bude provedeno ekonomické zhodnocení a přínosů práce.

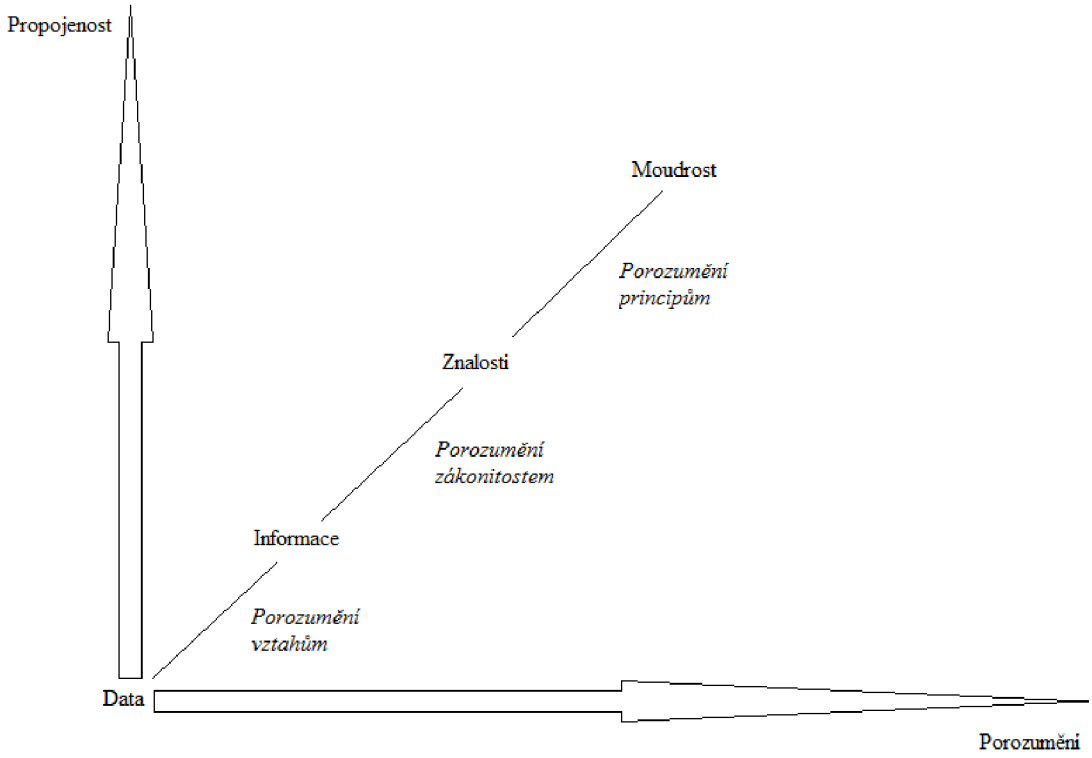
2. Teoretická východiska práce

V této části bude popsána nezbytná základní teorie, bez které by nebylo možné dosáhnout zvoleného cíle. Jako první budou charakterizovány základní pojmy a pojem databáze obecně. Poté budou popsány jednotlivé kroky a postupy tvorby nové databáze a nakonec bude stručně popsán jazyk SQL a VB, ve kterém bude systém vytvořen.

2.1 Data, informace, znalosti

Data, informace a znalosti jsou základním kamenem všech činností a fungování veškerých systémů, organizací, společností i jednotlivců. Když má člověk znalosti, může činit důležitá rozhodnutí. Veškeré jednání je založeno na informacích a znalostech.

Informační a komunikační technologie jsou nástrojem i prostředkem pro uchovávání informací a znalostí [6, 7].



Obrázek č. 1: Data, informace, znalosti, moudrost (Zdroj: převzato z 6, s. 5)

2.1.1 Data

Data jsou potenciálními informacemi. Lidé jsou ustavičně vystavováni působení zpráv. Některé z nich zachytí a porozumí jim a jiné zachytí, ale význam jim uniká. To je pro člověka to, co se nazývá daty. Data se mohou ukládat na pozdější zpracování nebo převést do jiné podoby, například do papírové podoby nebo elektronické podoby ve formě nějakého druhu databáze. Data jsou vyjádřena fyzicky na určitém nosiči, jako tužka a papír, elektrické signály nebo elektromagnetické záření. Data sama o sobě mají vypovídací schopnost [6, 7].

2.1.2 Informace

Informace jsou data, která nabývají významu, který vyplývá z kontextu. Význam byl získán v procesu interpretace a v procesu pochopení vztahů. Informace jsou konkrétní fakta, mohou se vyskytovat ve formě zpráv nebo sdělení. Tyto zprávy nebo sdělení je možné dále interpretovat, třídit nebo sdělovat [6, 7].

2.1.3 Znalosti

Znalosti se vyznačují jako informace o tom, jak využít další informace a data v rozličných situacích. Člověk se rozhoduje na základě znalostí, které získal minulými zkušenostmi. Znalosti jsou získávány v různé formě a v různé intenzitě každý den. Znalost je informace, která prošla uspořádáním a analýzou tak, aby mohla být používána k řešení problémů [6, 7].

2.2 Technická realizace dat

Data se ukládají do paměti počítače v nejmenší možné podobě a to je jeden Byte, který je tvořený osmi bity. Všechny Byty mají své číslo a adresu, podle kterých se lokalizují při ukládání dat. Každý bit může nabývat pouze dvou hodnot, nuly a jedničky. Do jednoho Bytu se vleze 256 možných kombinací jedniček a nul. Do Bytu se pomocí těchto nul a jedniček dají zakódovat různé textové znaky. Provádí se to pomocí takzvané ASCII tabulky, která obsahuje 16 řádků a 16 sloupců, což dává dohromady již zmíněných 256 kombinací. Každému znaku je přiřazena jednoznačná kombinace jedniček a nul [6, 7].

Tabulka č. 1: ASCII tabulka (Zdroj: 7, s. 7)

	0000	0001	0010	0011	0100	0101	0110	0111	...
0000									
0001									
0010		!	"	#	$	%	&	'	
0011	0	1	2	3	4	5	6	7	
0100	@	A	B	C	D	E	F	G	
0101	P	Q	R	S	T	U	V	W	
0110		a	b	c	d	e	f	g	
0111	p	q	r	s	t	u	v	w	
1000									
.....									

Binární hodnota řádku se uvádí do prvních čtyř bitů zleva a hodnota sloupce do druhých čtyř bitu. Například znak jedničky se zakóduje jako 0011 0001. Dále jsou v počítačích definovány takzvané znakové sady, tak aby se pokryly rozdíly mezi jednotlivými státy a jejich variacemi písma a znaků. Například znaková sada pro Českou republiku obsahuje jiné znaky než Ruská abeceda [6, 7].

2.3 Databáze

Databáze jsou neoddělitelnou součástí dnešního života a lidé se s nimi setkávají prakticky denně. Jedná se činnosti, které provádíme zcela automaticky, aniž bychom si uvědomili, že přicházíme do styku s databází. Je to prostředí, kde se uchovávají data o světě, tak aby s nimi šlo jednoduše pracovat, přidávat a mazat tyto data. Databáze se skládají z entit, které se popisují atributy a vzájemnými vazbami dochází k propojení těchto entit [1, 8].

2.3.1 Vývoj databází

Papírové databáze – Nejprimitivnější způsob uspořádání dat

Systémy sálových počítačů, Mainframe – Velmi dobré připojení k těmto systémům a dobrá schopnost ukládat velké objemy dat

Souborově orientované databáze (dBase) – typicky se pro každou tabulku používá jeden samostatný soubor. Předchůdce relačních databází.

Relační databázové systémy – Oproti předcházejícímu systému se vyznačují mnohem lepší datovou integritou. Samotná databáze přebírá zodpovědnost za to, co z databáze odchází a co do ní vstupuje, data jsou proto bezpečnější.

Objektově orientované databáze – Data se neukládají jako tabulky, ale jako objekt s vlastnostmi, které je nutné udržovat. Dochází k použití různých objektově orientovaných mechanismů [8].

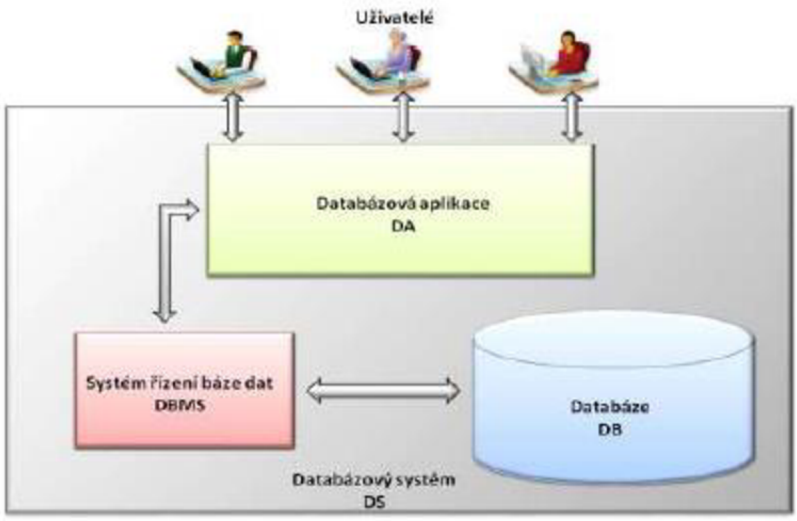
2.3.2 Rozdělení databází z pohledu správy

Transakční (operační databáze) – Typ databáze, ve kterých se data ustavičně mění a vždy odpovídají aktuálnímu stavu z reálného světa. Hlavní prostředek pro sběr a údržbu dat na celém světě. Zkratkou se označují tyto databáze jako OLTP (online transaction processing) v překladu online zpracování transakcí.

Analytické databáze – Typ databáze, ve kterých se ukládají starší data podle časového období. Data se získávají z transakčních databází. Analytické databáze vykonávají úplně odlišný typ úkolů, a proto mají odlišnou strukturu od transakčních databází. Slouží k zobrazování statistických údajů, trendů a podobně. Data v těchto databázích se mění jen velmi zřídka [8].

2.3.3 Co tvoří databázový systém

Databázový systém tvoří samotná databáze, systém řízení báze dat a databázová aplikace. Databázi tvoří také výše zmíněná data a informace. Data se ukládají do databáze a z databáze dostáváme informace. Mimo operační data uložená v databázi, jsou v ní uložena také takzvaná metadata. Pod tímto pojmem označujeme popis uložených dat v databázi. Tyto data se dají popsat také pojmem data o datech [8].



Obrázek č. 2: Databázový systém (Zdroj: 8)

Databázová aplikace neboli uživatelské rozhraní, je počítačový program, který interaguje s databází tak, že vyvolá příkaz jazyka SQL pro systém řízení báze dat [1].

Systém řízení báze dat je softwarový systém, který uživateli umožňuje definovat, vytvářet a udržovat databázi a poskytuje řízený přístup k této databázi [1].

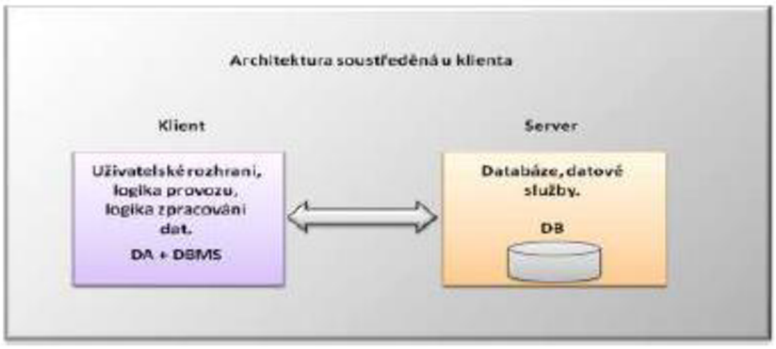
DBMS komunikuje mezi uživateli, databázovými aplikacemi a databází samotnou. Umožňuje uživatelům přidávat nová data, aktualizovat stávající a mazat nepotřebná nebo neaktuální data. DBMS poskytuje možnost dotazování na tato data, tato možnost se nazývá dotazovací jazyk, v tomto případě jazyk SQL. DBMS také spravuje systémový katalog, což jsou informace o datových položkách jako jméno, typ a velikost. Systém také řídí autorizovaný přístup k datům, podporuje transakce, což je posloupnost několika akcí, které pracují s daty v databázi [1, 8].

2.3.4 Architektury DBMS

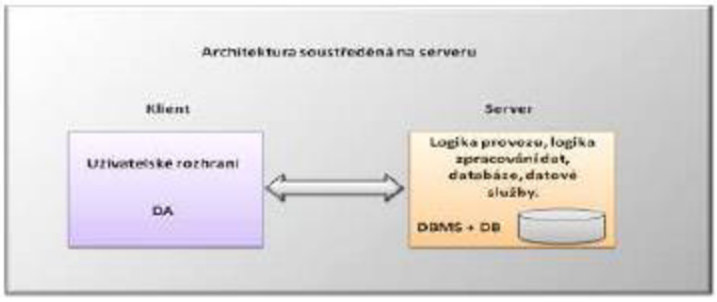
Architektury DBMS se dělí do třech skupin a liší se tím, kde probíhá logika provozu a kde vlastní zpracování dat.

Jednovrstvá architektura – Nejstarší architektura, nazývaná také jako centrální. V této architektuře je logika provozu, zpracování dat, databáze a datové služby na jednom centrálním počítači [1, 8].

Dvouvrstvá architektura – Tato architektura označována také jako klient server se dělí na dva typy. Architektura soustředěná u klienta a architektura soustředěná na serveru.

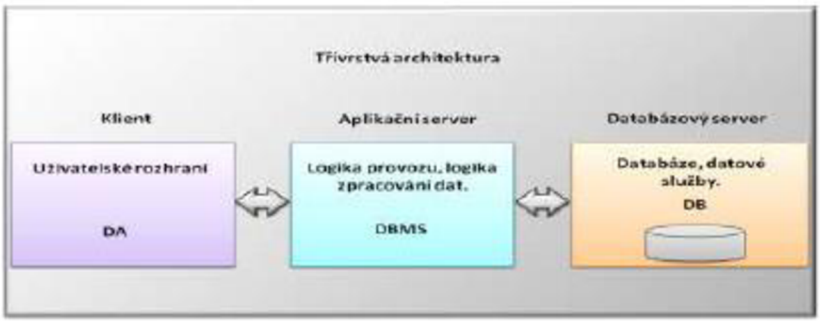


Obrázek č. 3: Klient/server (Zdroj: 8)



Obrázek č. 4: Server/klient (Zdroj: 8)

Třívrstvá architektura – Tato architektura rozděluje služby na vrstvu uživatelského rozhraní, která běží na klientovi, vrstvu provozu a zpracování dat, která běží na aplikačním serveru, databázi a datové služby, která běží na databázovém serveru.

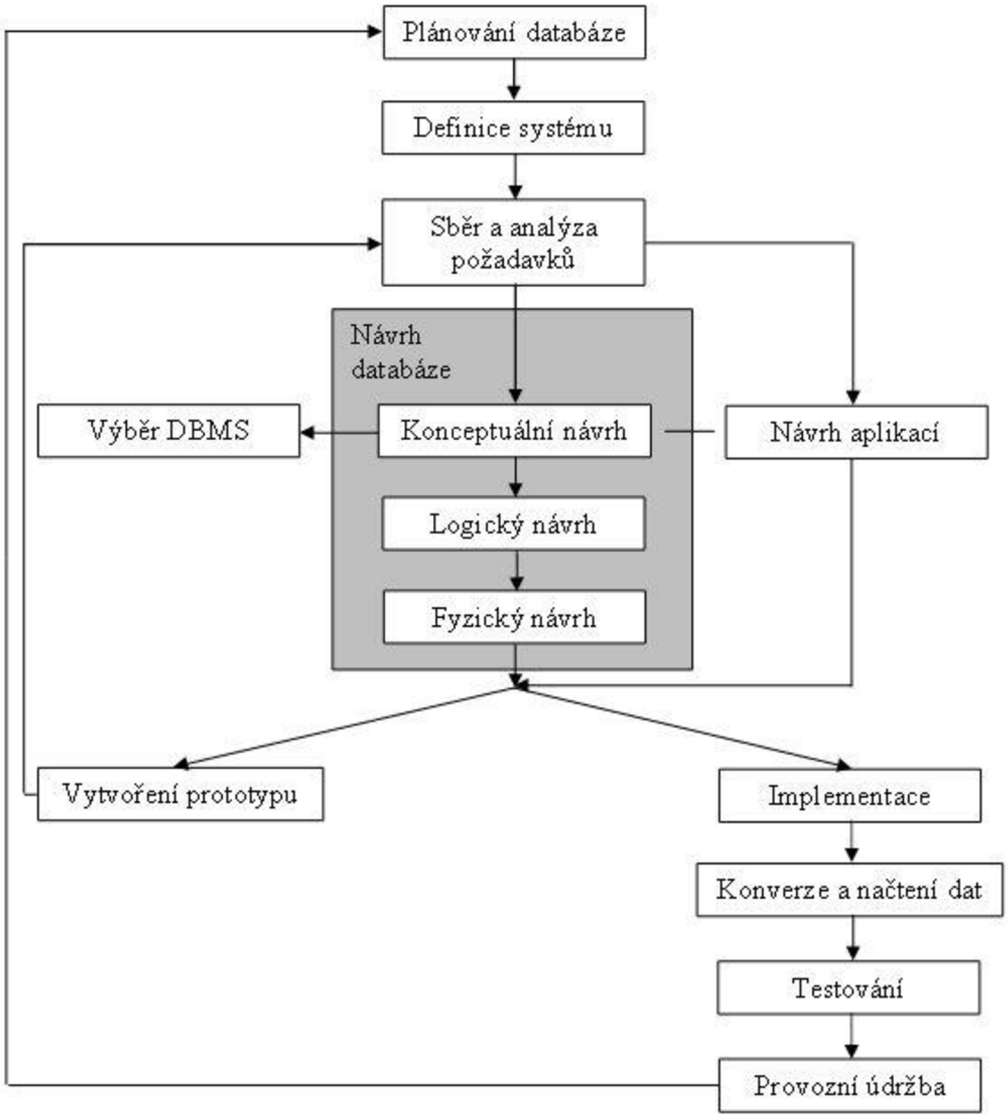


Obrázek č. 5: 3vrstvá architektura (Zdroj: 8)

2.4 Životní cyklus informačního systému

Informační systém je určen nejenom ke shromažďování, správě a kontrole dat, ale také slouží k transformaci dat na informace. Informační systém také slouží pro usnadnění šíření informací k těm, kteří činí rozhodnutí. Důležitou částí informačního systému je databáze, která tento informační systém podporuje [1, 2].

Životní cyklus informačního systému tvoří fáze plánování, sběr a analýza požadavků, návrh databáze, vytvoření zkušební verze, návrh uživatelského rozhraní, implementace, testování, spuštění ostrého provozu a nakonec provozní údržba [1, 2].



Obrázek č. 6: Životní cyklus databázového systému (Zdroj: 1, s. 110)

2.4.1 Plánování databáze

Tvorba databázového projektu začíná stanovením celkového poslání a dílčích cílů pro databázový systém. Celkové poslání stanovuje hlavní cíl databázového systému a dílčí cíle určují dílčí úkoly, které bude databáze vykonávat. Měl by zde být zahrnut odhad rozsahu práce, potřebných zdrojů a celkových finančních prostředků [1, 2].

2.4.2 Definice systému

Před samotným návrhem databázového systému je potřebné nejdříve identifikovat rozsah a hranice systému, jeho rozhraní s ostatními informačními systémy organizace. Je důležité promyslet, jaké informace databáze bude obsahovat a jaké služby bude poskytovat [1].

2.4.3 Sběr a analýza požadavků

V této fázi životního cyklu informačního systému probíhá sběr a analýza informací o organizaci nebo jenom její části, které bude vytvářená databáze sloužit. Sběr informací se provádí pro každý důležitý uživatelský pohled včetně:

- Popisu používaných nebo vytvářených dat
- Podrobností o tom, jak jsou data používaná a vytvářená
- Všech dalších požadavků na databázový systém

Poté se analyzují informace, aby se určily požadavky, které bude potřeba obsáhnout do vytvářeného databázového systému [1].

2.4.4 Návrh databáze

Je to proces návrhu, který bude podporovat celkové poslání a dílčí cíle pro vytvářený databázový systém. Návrh databáze tvoří tři hlavní fáze nazývané konceptuální, logický a fyzický návrh. V konceptuálním návrhu se identifikují objekty, které budou v databázi a relační vazby mezi nimi. V logickém návrhu se provádí reprezentace těchto objektů a jejich relací množinou tabulek. Ve fyzickém návrhu databáze se rozhoduje o tom, jak je třeba tyto tabulky fyzicky implementovat v konkrétním DBMS [1].

2.4.5 Návrh aplikací

Návrh databáze a návrh aplikací jsou souběžné činnosti v životním cyklu vývoje nového databázového systému. Nelze dokončit vývoj aplikací, dokud nedojde k vytvoření databáze samotné. Naopak databáze existují kvůli podpoře aplikací, a proto musí probíhat tok informací mezi návrhem aplikací a návrhem databáze. Kromě návrhu, jak docílit

požadované funkčnosti, musí se navrhnout patřičné uživatelské rozhraní databázového systému. Toto rozhraní by mělo poskytovat požadované informace uživatelsky přívětivým způsobem. Uživatelské rozhraní je jednou z nejdůležitějších složek informačního systému. Pokud je uživatelské rozhraní přehledné a srozumitelné a lze ho jednoduše, intuitivně a přímočaře používat, uživatel bude schopen využívat poskytované informace naplno. Pokud uživatelské rozhraní tyto charakteristiky nemá, bude systém uživateli způsobovat potíže a celkový efekt práce s novým informačním systémem vyzní do ztracena. Mezi základní doporučení pro návrh přívětivého uživatelského rozhraní patří:

- Smysluplné titulky v záhlaví formulářů
- Logické seskupení a uspořádání polí
- Graficky působivé rozvržení formuláře
- Konzistentní terminologie a zkratky
- Konzistentní používání barev
- Přehledné pole pro zadávání dat
- Kontrola špatně zadaných vstupních dat
- Chybová hlášení
- Označení nepovinných polí [1]

2.4.6 Vytvoření prototypu databáze

Prototyp je fungující model databáze, který nemá všechny požadované vlastnosti nebo nepodporuje veškerou funkčnost vyvíjeného systému. Jeho účelem je umožnit uživatelům identifikovat, které vlastnosti fungují dobře a které ne a navrhnout zlepšení systému [1].

2.4.7 Implementace

Po dokončení fáze návrhu se může přejít k samotné implementaci databáze a databázové aplikace. Implementace databáze se provádí pomocí jazyka pro definici dat (DDL).

Příkazy DDL se využívají k vytvoření databázových struktur a prázdných databázových souborů. Jazyk SQL a jeho kategorie bude popsán níže v další kapitole. K implementaci databázových aplikací se v současné době používají jazyky třetí nebo čtvrté generace jako jsou: Visual Basic, Python, C++, Java. V této práci bude využito jazyka Visual Basic [1].

2.4.8 Testování

Před ostrým spuštěním provozu by se měla databáze a databázová aplikace důkladně otestovat. Testování se provádí s realistickými daty, aby se celý proces testování uskutečnil seriózně. Testování se provádí za účelem odhalení chyb v databázových aplikacích a možné jsou i chyby v samotné databázové struktuře. Další účel fáze testování je ukázka, že databáze a její fungování se jeví v souladu se specifikací a že požadavky na databázi a její aplikaci byly splněny. Podobně jako při návrhu a vytváření prototypu, by se měli testování zúčastnit i uživatelé nově vzniklé aplikace. Testování by se mělo zaměřit hlavně na tyto oblasti:

- Osvojitelnost
    - Jak dlouho trvá novému uživateli, než je se systémem schopen produktivně pracovat?
- Provedení
    - Do jaké míry odpověď systému odpovídá pracovním postupům uživatele?
- Robustnost
    - Jak odolný je systém vůči chybám uživatele?
- Zotavení
    - Jak dobře se systém zotaví po chybě uživatele?

Po dokončení testování a vyhodnocení kritérií je databázový systém připraven k předání uživatelům a nasazení do ostrého provozu [1].

2.4.9 Provozní údržba

Poslední stádium životního cyklu informačního systému, zahrnuje tyto činnosti:

- Monitorování provozu databázového systému
- Udržování a aktualizace databázového systému [1]

2.5 Zabezpečení databáze

Zabezpečení databáze můžeme definovat jako mechanismy, které zabezpečují databáze proti záměrným nebo náhodným ohrožením. Zajištění zabezpečení se netýká jen dat uchovávaných v databázi. Narušení bezpečnosti ovlivňuje také jiné části systému, což mívá zpětně dopady na databázi samotnou. Zabezpečení databáze tedy zahrnuje také hardware, software, pracovníky a data. Funkční zavedení zabezpečení vyžaduje odpovídající kontroly, které jsou upřesněny v dílčích cílech systému. V minulosti vlastnící databází zabezpečení zanedbávali nebo přehlíželi, ale v současnosti si již uvědomují jeho význam [1].

2.5.1 Následky nezabezpečení

Databáze představuje velmi důležitý zdroj organizace a musí být řádně zabezpečena. Následující faktory patří k nejdůležitějším z hlediska bezpečnosti databází:

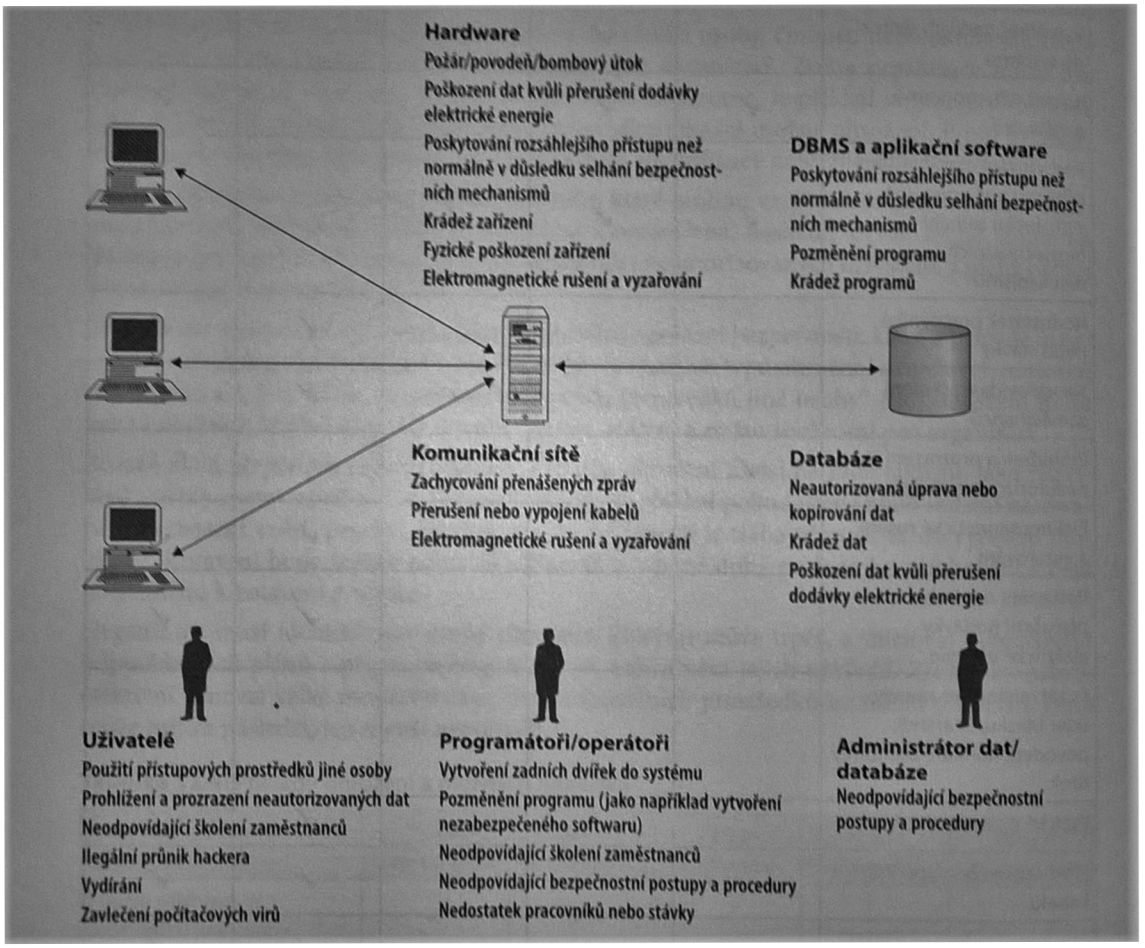
- Krádež a zpronevěra
    - Neovlivňují jen databázové prostředí, ale také celou organizaci. Z důvodu toho, že se těchto činů dopouštějí lidé, je třeba se zaměřit na redukci příležitostí. Krádež a zpronevěra nemusí změnit data.
- Ztráta utajení
    - Mechanismus zabezpečující data a informace tak, aby byly přístupné jen osobám, které jsou k této činnosti autorizovány. Týká se potřeby udržet data v tajnosti, obvykle pouze těch kriticky důležitých pro chod organizace.

- Ztráta soukromí
    - Týká se nutnosti chránit samotná data o jednotlivých osobách. Narušení bezpečnosti, které má za následek ztrátu soukromí, mohou znamenat právní akci pro organizaci.
- Ztráta integrity
    - Má za následek to, že data jsou neplatná nebo znehodnocená, což může ovlivnit chod organizace.
- Ztráta dostupnosti
    - Znamená to, že systém, data nebo obojí jsou nedostupné a to může mít opět vliv na rutinní chod organizace.

V těchto oblastech by organizace měla minimalizovat riziko neboli možnost ztráty nebo škody. V některých situacích jsou tyto faktory tak propojeny, že ztráta v jedné oblasti může vést ke ztrátě i v jiných oblastech. Platí také, že události jako krádež nebo ztráta soukromí vznikají buď záměrnými nebo nezáměrnými činy a organizace nemusí pozorovat žádné změny informačního systému nebo databáze [1].

2.5.2 Druhy ohrožení

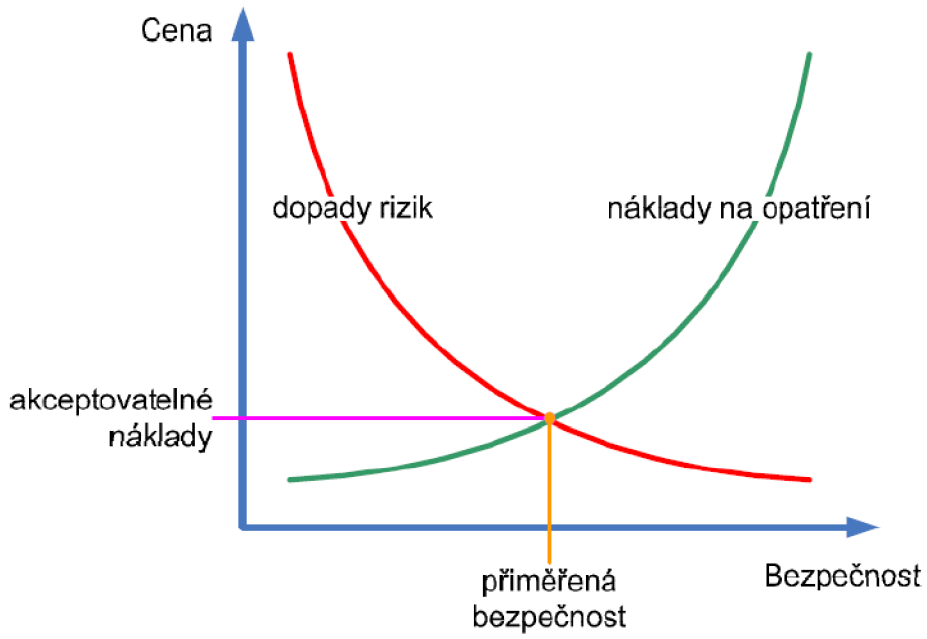
Ohrožení se může definovat jako situace nebo událost, která může být záměrná i nezáměrná a která může nepříznivě ovlivnit systém a následně i organizaci. Může být způsobeno situací nebo událostí zahrnující osoby, činnosti nebo okolnosti, která bude mít neblahé důsledky pro organizaci. Poškození může být hmotné: hardware, software, data nebo nehmotné: ztráta věrohodnosti nebo důvěry. Každá organizace by měla identifikovat potenciální ohrožení, a proto by se mělo investovat do času a úsilí, které povedou k identifikaci nejvážnějších ohrožení [1].



Obrázek č. 7: Shrnutí potenciálních ohrožení databázových systémů (Zdroj: 1, s. 296)

2.5.3 Přiměřená bezpečnost

Cílem zajištění bezpečnosti informací je zajistit tzv. přiměřenou bezpečnost. Přehnané nároky na zabezpečení databáze a uživatelské aplikace zvyšují náklady na vývoj a provoz těchto systémů. Naopak nedostatečná kvalita zabezpečovacích mechanismů může znehodnotit celý proces návrhu a vývoje informačního systému. Obecně lze říci, že informační systém je kvalitní, když splňuje požadavky uživatelů a funguje bezpečně a spolehlivě [2].



Obrázek č. 8: Přiměřená bezpečnost (Zdroj: 10)

Velikost úsilí a investic do bezpečnosti informačního systému musí odpovídat hodnotě aktiv a míře možných rizik. To stanovuje především bezpečnostní politiky organizace [10].

2.6 Základy jazyka SQL

SQL označuje Structured Query Language a tedy strukturovaný dotazovací jazyk. Jedná se o jazyk deklarativní. U těchto typů jazyků se říká počítači pouze, co má být výsledkem a již nikoho nezajímá, jak toho výsledku počítač dosáhne. Databázové dotazy jsou zjednodušeny na příkazy typu, vrať hodnotu. Databáze takový dotaz pochopí a zpracuje a poté vrátí výsledek. Jazyk vznikl v laboratořích společnosti IBM za účelem vytvořit jazyk, kterým by se dalo komunikovat s databází pomocí jednoduché angličtiny. SQL databáze měly úspěch a ujaly se v běžném životě. Dnes se prakticky nic jiného nepoužívá [9, 16].

Nejprve budou popsány datové typy, které budou použity v praktické části a poté základní příkazy jazyka, tak ať vznikne teoretický podklad pro vypracování vlastního návrhu řešení.

2.6.1 Datové typy jazyka SQL

V této podkapitole budou popsány datové typy, které budou použity v praktické části práce.

Celočíselné datové typy

- BIT – vyjadřuje dvě hodnoty, 0 nebo 1 anebo pravda/nepravda
- INT – celé číslo v rozsahu od -2^{31} do 2^{31} -1, zabírá 4 bajty
- SMALLINT – celé číslo v rozsahu -2^{15} do 2^{15} -1, zabírá 2 bajty
- TINYINT – celé číslo v rozsahu od 0 do 255, zabírá 1 bajt [15, 16]

Znakové datové typy

- CHAR() – ukládá textový řetězec pevné délky, podle parametru
- VARCHAR() – ukládá řetězec variabilní délky opět podle parametru [15, 16]

Časové datové typy

- DATETIME – ukládá data jako jedinou hodnotu ve formátu 01/01/1753 až 12/31/9999
- SMALLDATETIME – rozsah 01/01/1900 až 12/31/2079
- GETDATE – vloží aktuální datum a čas [15, 16]

2.6.2 Základní příkazy jazyka SQL

DDL

Data definition language. Příkazy, které umožňují vytvářet, mazat a upravovat databázové objekty. Tyto příkazy neumožňují vytvářet, mazat nebo upravovat jednotlivé záznamy v tabulkách.

- CREATE DATABASE – vytvoření databáze
- CREATE TABLE – vytvoření tabulky
- DROP TABLE – smazání tabulky
- ALTER TABLE – úprava tabulky [15, 16]

DQL

Pomocí těchto příkazů se načítají data z databáze. Jsou založeny pouze na jediném příkazu.

- SELECT – výběr dat [15, 16]

DML

Tato skupina příkazů umožňuje přidávat data do databáze a manipulovat s nimi.

- INSERT – vložení dat
- UPDATE – úprava dat
- DELETE – smazání dat [15, 16]

2.7 Základy programování a jazyka Visual Basic

První předchůdce VB.NET se nazýval jednoduše BASIC, což byla zkratka pro Beginners All Purpose Symbolic Instruction Code. Byl navrhnut roku 1963. U jeho zrodu stáli dva pánové: John Kemeny a Thomas Kurz. Zaměstnanci Dartmouth College. Jako jeden z potomků BASICU vyvinul Microsoft Visual Basic. Jeho první verze pro Windows vznikla v roce 1991. Visual Basic posléze prošel mnoha vývojovými verzemi, z nichž poslední byl Visual Basic 6.0, který vyšel v roce 1998. Poté se vývoj Visual Basicu zastavil a jeho roli převzal Visual Basic.NET. Koncovka NET znamená, že tato verze je stavěna pro tvorbu programů v prostředí .NET Frameworku. Tento Framework je již součástí všech operačních systémů od Microsoftu. VB.NET se používá v rámci vývojového prostředí neboli IDE. Konkrétně se jedná o vývojové prostředí Visual Studio opět z dílny společnosti Microsoft [17, 18].

Programování ve Visual Basicu se řadí mezi objektově orientované. To znamená, že programátor může využívat množství předdefinovaných objektů, jako jsou formuláře, textová pole pro zadávání a zobrazování dat, různé příkazové tlačítka a zaškrtávací boxy, rolovací menu, popisky a další objekty. Souhrnně se tyto objekty nazývají ovládací prvky. Každý ovládací prvek má své vlastnosti, metody a události. Vlastnosti udávají vzhled a chování ovládacího prvku v aplikaci. Vlastnosti lze dělit do několika kategorií, jako jsou vzhled, chování, vazba na databázová data atd. Metody těchto objektů představují činnosti, které daný objekt může vykonávat nebo které činnosti mohou být vykonány na něm. U každého objektu existuje seznam událostí, které mohou při běhu programu vzniknout ve vztahu k tomuto objektu. Klasickými událostmi jsou: kliknutí, dvojité kliknutí, přejetí kurzorem na objekt a další. Procedura, která bude přiřazena události klik, se spustí vždy, když uživatel na daný objekt klikne [17, 18].

2.7.1 Proměnné

Proměnná je rezervované místo v paměti počítače, do kterého se přiřazuje hodnota. Velikost tohoto místa se určuje dle typu proměnné. Mezi základní typy proměnných, které budou využity v praktické části práce, patří:

- Integer – celé číslo, které má velikost 32 bitů a rozsah -2^{31} až 2^{31}
- Boolean – Logická hodnota (pravda/nepravda), má velikost 8 bitů a rozsah true nebo false
- String – Textová hodnota, která má velikost 16 bitů pro každý znak a rozsah není dán.
- Date – proměnná pro ukládání data

K vytvoření proměnné slouží deklarace:

Dim (název proměnné) As (Typ proměnné) [17, 19]

2.7.2 Metody a funkce

Metoda je oddělený kus logiky programu, při zavolání metody se provede všechen kód v ní obsažený a poté se metoda ukončí. Funkce má oproti metodě jednu důležitou funkci navíc. Tou funkcí je návratová hodnota. Když se volá funkce, proběhne všechen kód v ní až po klíčové slovo return, které navrátí hodnotu.

K vytvoření metody a funkce slouží deklarace:

Sub (Název metody) (parametry metody), tělo metody a End

Function (Název funkce) (parametry funkce), tělo funkce, return a End [17, 18]

2.7.3 Podmínky

Podmínky slouží k vyhodnocování údajů při běhu programu a zvolení správné cesty při větvení programu. Syntaxe pro psaní podmínek je následující:

If (podmínka) Then

(kód při splnění podmínky)

Else

(kód při nesplnění podmínky)

EndIf

Když program dojde k podmínce, vyhodnotí kód podmínky, a když je podmínka vyhodnocena jako true provede kód za Then, v opačném případě provede kód za Else [17, 18].

2.7.4 Cykly

Cyklus je část programu, která vykonává dokola nějakou činnost.

Cyklus while je jedním z nejpoužívanějších cyklů vůbec. Dalším cyklem je Do, který je téměř shodný s while. Cyklus while nejdříve testuje podmínku a až poté provádí tělo cyklu. Pokud je podmínka nepravdivá, neprovede se tělo cyklu ani jednou. Do provede

tělo cyklu a až potom testuje podmínku a tedy tělo cyklu se provede vždy alespoň jednou [17, 18].

Syntaxe cyklu While je následující:

While (podmínka)

tělo cyklu

End While

Syntaxe cyklu Do má dvě varianty, s podmínkou na konci, kdy se tělo cyklu provede vždy alespoň jednou a s podmínkou na začátku.

Do

tělo cyklu

Loop while (podmínka)

Do until (podmínka)

tělo cyklu

Loop

Dalším velmi důležitým cyklem je cyklus For. Ten je charakteristický tím, že už na začátku se ví, kolikrát cyklus proběhne. Syntaxe vypadá následovně:

For (proměnná) to (proměnná)

tělo cyklu

Next [17, 18]

2.7.5 Zachytávání chyb

Běhové chyby a další typy chyb jsou běžnou součástí vývoje programu. Tyto chyby se též označují, jako výjimky. Tyto chyby neboli výjimky vznikají v důsledku normálního

běhu programu. Procedury, které zpracovávají výjimky, se nazývají strukturovanými zpracovateli výjimek a mohou být využity k detekci běhových chyb, potlačení chybových hlášení nebo upravení podmínek, tak aby aplikace mohla pokračovat běžným způsobem. K tomuto účelu se ve Visual Basicu využívá výkonný blok Try…Catch. Havárie nebo pád programu je způsoben nečekaným problémem, který nebyl schopen program nijak vyřešit. Problém je v tom, že programu nikdo neřekl, jak se má chovat v případě vyskytnutí nečekané situace. Strukturovaný zpracovatel chyb řekne programu, jak se má v případě takové nečekané situace zachovat. Syntaxe vypadá následovně:

Try

(tělo programu, které může vyvolat běhovou chybu)

Catch

(příkazy, které mají být provedeny, pokud se běhová chyba vyskytne)

End Try [4]

3. Analýza současného stavu

V této části práce bude provedena analýza současné situace školy a daného problému. Poté bude popsáno hardwarové a softwarové vybavení školy a stručný popis informačních systémů, které škola používá.

3.1 Základní údaje o škole

- **Název**: Střední průmyslová škola, Obchodní akademie a Jazyková škola s právem státní jazykové zkoušky, Frýdek-Místek, příspěvková organizace
- **Sídlo**: 28. října 1598, 738 01 Frýdek-Místek
- **IČO**: 00601381

3.2 Stručná historie školy a její vývoj v průběhu let do dnešní podoby

O tom, že bude založena Střední průmyslová škola, bylo rozhodnuto v roce 1955 v okamžiku, kdy město dospělo k závěru, že v místecké části města je potřeba zřídit novou střední školu, které bude zaměřena na výchovu absolventů s technickými znalostmi a která by vychovávala pracovní sílu pro průmyslové podniky v moravskoslezském kraji. Jako první se otevřely obory hutnictví a strojírenství v září roku 1958. O čtyři roky později měla škola otevřeny již všechny čtyři ročníky těchto dvou oborů. Později, v sedmdesátých letech přibyl obor slévárenství. Tento stav tří oborů zůstal nezměněn až do roku 1988, kdy se škola rozrostla o nový obor textilní technologie. V roce 1990 byl počet studentů 902, což je doposud nejvyšší počet studentů, jaký na této střední škole působil. Škola po celou dobu své existence spolupracuje s průmyslovými podniky v okolí, jako jsou Vítkovické strojírny a železárny, textilka Slezan Frýdek-Místek nebo Ferrum. Po Sametové revoluci stoupl zájem o administrativní obory, a proto vznikly další nové studijní obory jako strojírenská a hutnické administrativa, kde se vychovávali zaměstnanci do kanceláří. Po roce 2000 vznikly další studijní programy a to stavební a technické zařízení budov a také bylo založeno technické lyceum, které připravuje žáky na studium na technických vysokých školách. V současné době škola spolupracuje se společnostmi jako Arcelor Mittal Ostrava a.s. a Třinecké železárny a ocelárny a.s., které také poskytují finanční podporu perspektivním žákům a i těm, kteří musí dojíždět z okolí.

Škola je držitelem několika prestižních certifikátů, které umožnily vytvořit na škole akreditované středisko výpočetní techniky. Žáci této školy se dlouhodobě umísťují na předních příčkách ve státních a mezinárodních studentských soutěžích a olympiádách [11].

3.3 Způsob archivace údajů na škole

Každodenní agenda, která se stará o rutinní chod školy, běží na školském informačním systému Bakaláři. Když žák ukončí své studium na škole, ať už úspěšným složením maturitní zkoušky nebo ne, tak jsou jeho údaje převedeny z elektronické podoby ze systému Bakaláři do podoby papírové a jsou uloženy do školního archivu. Tento archiv se nachází ve sklepních místnostech budovy. V tomto školním archivu se nacházejí maturitní protokoly a žákovské práce, které během svého studia žáci vypracovali a které se musejí ze zákona určitou dobu uchovávat. V maturitních protokolech jsou obsaženy údaje o jednotlivých studentech, o jejich hodnocení z maturitní zkoušky a také údaje o tom, kdo toto hodnocení udělil. Žákovské práce jsou vypracované dlouhodobé projekty, které jsou u některých studentů součástí maturitní zkoušky. Tyto žákovské práce jsou tříděny do svazků podle předmětu a školního roku přičemž, každý svazek obsahuje předepsané písemné práce z předmětu za třídu [20].

3.4 Legislativní požadavky na archivaci

Škola si uchovává předepsané dokumenty dle zákona o archivnictví č. 499/2004 Sb. o archivnictví a spisové službě a jeho prováděcí vyhlášky č. 646/2004 Sb. o podrobnostech výkonu spisové služby, dále ze zákona o účetnictví č. 563/91 Sb. a jeho pozdějších novel a směrnice MŠ ČSR ze dne 13.6.1986 č.j. 18200/86-49, který byl vydán jako skartační řád pro školy a školská zařízení [12].

3.5 Proces archivace

Požadované dokumenty se archivují podle typu dokumentu různou dobu. Veškeré práce a protokoly jsou archivovány dle Skartačního a spisového řádu a jsou zařazeny do skupin, podle skartačních znaků A a S. Typ S – stoupa nebo skart, se po uplynutí ochranné skartační lhůty skartují a dále se neuchovávají. Typ A – po uplynutí skartační lhůty se

předávají do hlavního archivu, kde jsou trvale uloženy. Dokumenty jsou rozděleny na písemné práce žáků, které náleží do skartační skupiny S5, a maturitní zkoušky, které jsou zařazeny ve skupině A45. Maturitní zkouška se dále dělí na písemné práce a protokoly o maturitní zkoušce. Písemné žákovské práce ze skupiny S5 se archivují ve školním archivu po dobu pěti let a poté jsou skartovány. Maturitní zkoušky a protokoly ze skupiny A45 se ve školním archivu uchovávají 45 let a poté jsou předány místní pobočce matriky. V současné době se počty studentů na škole pohybují kolem čísla 650 a těchto informací o všech studentech, kteří školou prošli za posledních 45 je ohromné množství. Tyto informace v papírové podobě zabírají místo v místnostech školy, které by mohly být využity lépe a také hledání v tomto papírovém archivu zabírá zaměstnancům školy množství času, který by mohl být využit efektivněji [20].

3.6 Informační systémy školy

V současné době jsou informační systémy nepostradatelnou součástí českého školství a státní správy obecně. Tyto informační systémy umožňují efektivnější komunikaci a zpracovávání informací. Starají se o ukládání informací nezbytných pro chod školy a její komunikaci uvnitř školy i s veřejností. Moderní informační systémy se starají o rutinní činnosti a vytváří podmínky pro plynulý chod školy.

Školské informační systémy se neustále vyvíjejí, za poslední roky prošly tyto systémy podstatným vývojem od databází v papírové podobě až po komplexní informační systémy, přičemž zaměření na administrativní práci spojenou s pedagogickými procesy zůstává. Zcela rutinními se staly funkce jako tisk vysvědčení, tvorba rozvrhů, elektronická třídní kniha a tak dále. Technický pokrok nezastavuje, a aby dodavatelé nezaostali za konkurencí, musí svou nabídku neustále rozšiřovat a zkvalitňovat, jsou důležité inovace. Ve výsledku dochází k neustálému vyvíjení nových modulů, které doplňují a rozšiřují stávající funkčnost.

3.6.1 Moodle

Modular Object-Oriented Dynamic Learning Environment – Programový balíček pro vytváření výukových systémů a online kurzů na internetu. Je distribuován zdarma jako Open Source pod veřejnou licencí GNU. Lze používat na počítačích s funkčním php [13].

3.6.2 Bakaláři

Jedná se o školský informační systém od společnosti Bakaláři software s.r.o. Řeší rutinní agendu jako evidenci žáků a zaměstnanců školy, klasifikaci a úvazky, sestavování rozvrhů hodin a suplování. Je to modulární systém a každá škola si může zvolit s širokého výběru modulů a tím co nejvíce sladit své potřeby s možnostmi systému. Mezi další moduly patří půjčování knih a učebnic, tvorba tematických plánů, správa majetku a tvorba rozpočtu školy. Systém je vytvořen speciálně pro český a slovenský trh a je kompatibilní s Microsoft SQL Server [14].

3.7 Důvody realizace elektronického archivu

Ani jeden z výše uvedených informačních systémů, které jsou školou využívány, neumožňuje elektronické uchovávání informací o absolventech i jejich maturitních protokolech a žákovských pracích. Když žák ukončí na škole své studium, jsou data z IS Bakaláři transformována do papírové podoby a umístěna do školního archivu. Správu tohoto archivu má na starosti správce školního archivu. Udržovat tento archiv, který uchovává informace v papírové podobě je poměrně náročné na čas. Je třeba všechny operace dělat ručně a udržovat archiv v provozuschopném stavu, tak aby se v něm nacházely pouze relevantní informace. Musejí se ručně zakládat nové složky s novými daty, hlídat expiraci stávajících dat. Tento systém archivace je v dnešní době již zastaralý nemluvně o časové náročnosti a spotřebě papíru. Hledání konkrétních informací v tomto archivu zabírá množství času a je neefektivní. Proto se tato práce tomuto problému věnuje a řeší ho.

Navrhovaná databáze a k ní vytvářené uživatelské rozhraní bude fungovat zcela samostatně. Využije se zde kompatibility IS Bakaláři s MS SQL serverem. Základní data o studentech a zaměstnancích budou exportovány z IS Bakaláři do nově vzniklé databáze, tak ať se tato opakující se činnost nemusí vykonávat ručně. V okamžiku, kdy se z žáka stane absolvent, oprávněná osoba vloží hodnocení do databáze. Databáze také hlídá, kdy vyprší ochranná skartační lhůta.

3.8 Analýza požadavků z pohledu životního cyklu IS

V této části bude provedena analýza požadavků na nový databázový systém, tak jak byly tyto činnosti popsány v teoretické části práce.

3.8.1 Plánování databáze

Plánovaný databázový systém bude sloužit k archivování údajů o absolventech školy. Za cíl si klade především uchovávat údaje z maturitních protokolů. Uchovávání hodnocení studentů za dlouhodobé maturitní práce, praktickou maturitní zkoušku a především samotnou maturitní zkoušku. Studenti budou rozdělení do tříd dle jejich zaměření a to buď na standardní studium nebo na technické lyceum, které studenty připravuje na studium na vysokých školách. Práce by měla být zhotovena v co nejkratším termínu, tak ať může být od příštího akademického roku implementována. Celkové finanční prostředky na tuto aplikaci by se měly vejít do 50 000 Kč.

3.8.2 Definice systému

Rozsah systému bude pouze o uchovávání údajů o maturitním hodnocení studentů. Běžnou školní agendu spojenou s každodenním provozem školy budou obstarávat stávající informační systémy, které byly popsány výše. Po úspěšném složení maturitní zkoušky budou údaje exportovány z těchto systémů do nově vznikající databáze, tak ať se tyto údaje nemusejí vkládat ručně.

K databázi budou přistupovat především zaměstnanci školy. Nejvíce bude používat aplikaci správce školního archivu a třídní učitelé, kteří budou vkládat nové údaje o žácích, kteří odmaturovali.

3.8.3 Sběr a analýza požadavků

Škola si přeje minimální náklady na návrh řešení, proškolení zaměstnanců, kteří přijdou s novou aplikací do styku a také na údržbu softwaru.

Výsledná aplikace by měla splňovat tyto funkce:

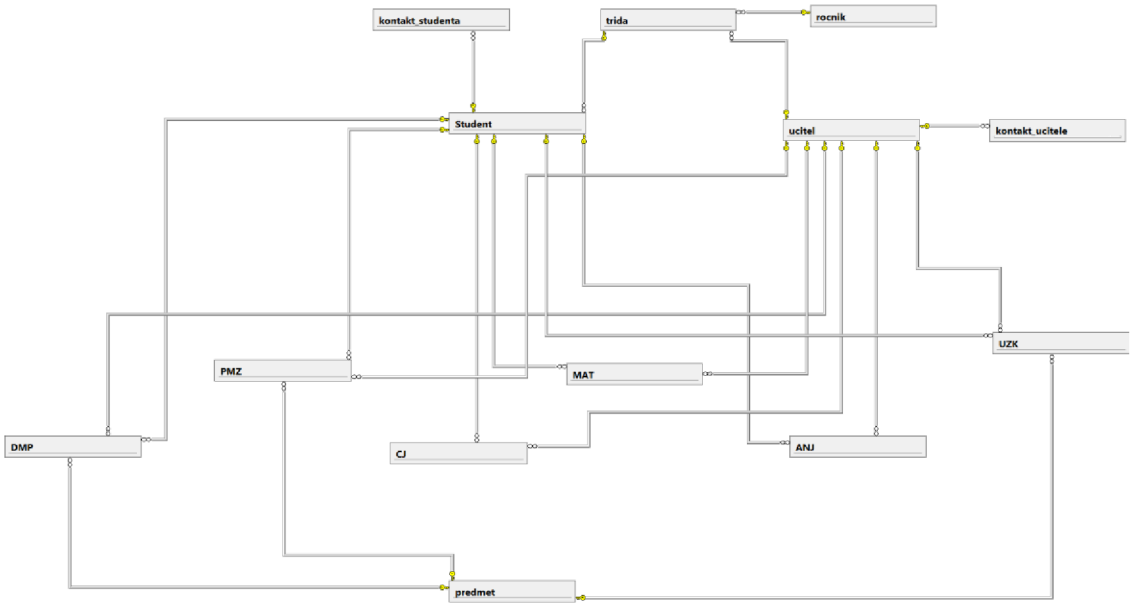
- Vkládání nových dat do databáze exportem ze stávajících aplikací
- Ruční vkládání dat
- Úprava a mazání nevyhovujících dat
- Různé možnosti zobrazení výsledných údajů, dle konkrétních požadavků

Tyto požadavky souvisejí s požadavky na náklady. Dalším z požadavků je, aby výsledná databáze a její uživatelské rozhraní bylo uživatelsky přívětivé a bez velkých nároků na jeho ovládání.

3.8.4 Návrh databáze

Konceptuální návrh

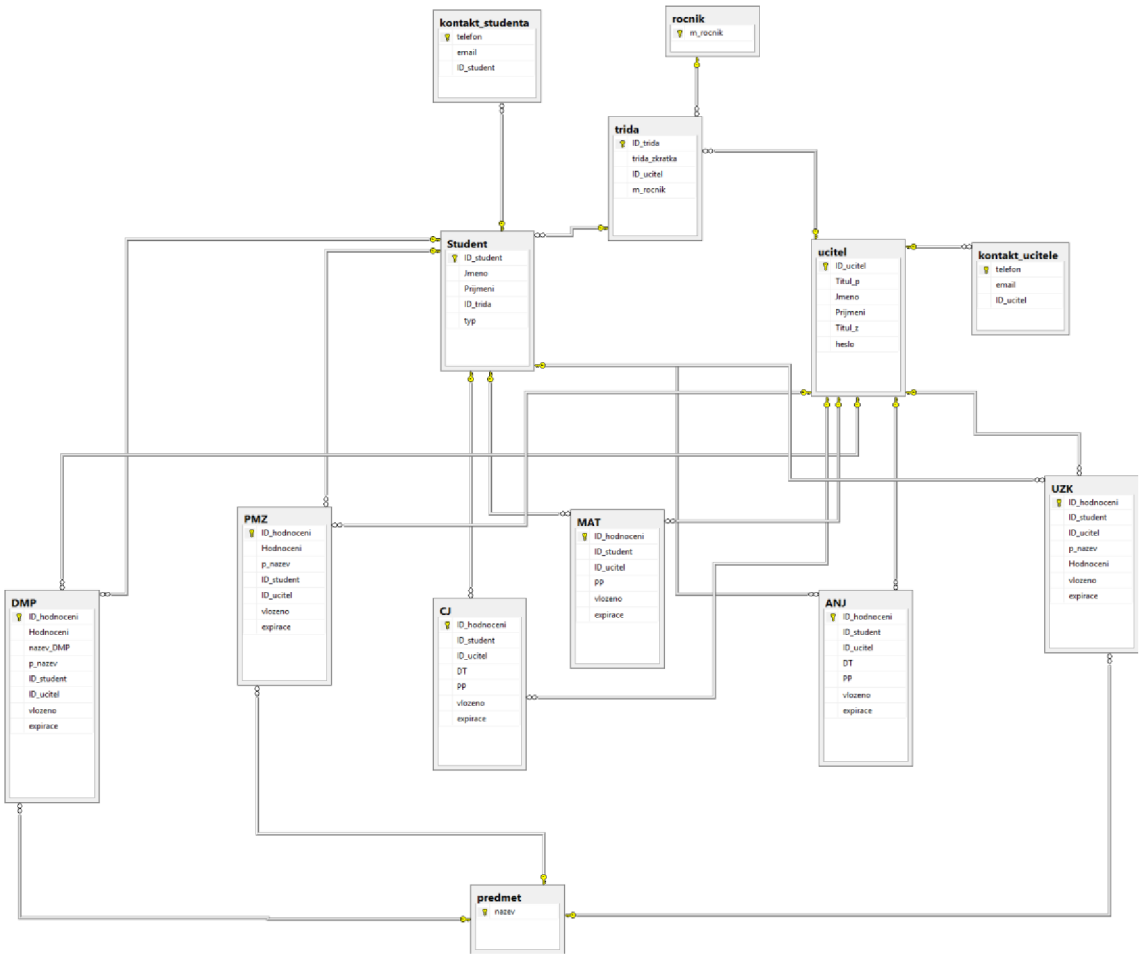
Cílem konceptuálního návrhu je vytvořit ER diagram, který splňuje požadavky na databázi. Výsledný ER Diagram, který popisuje pouze entity a vazby mezi nimi vypadá následovně:



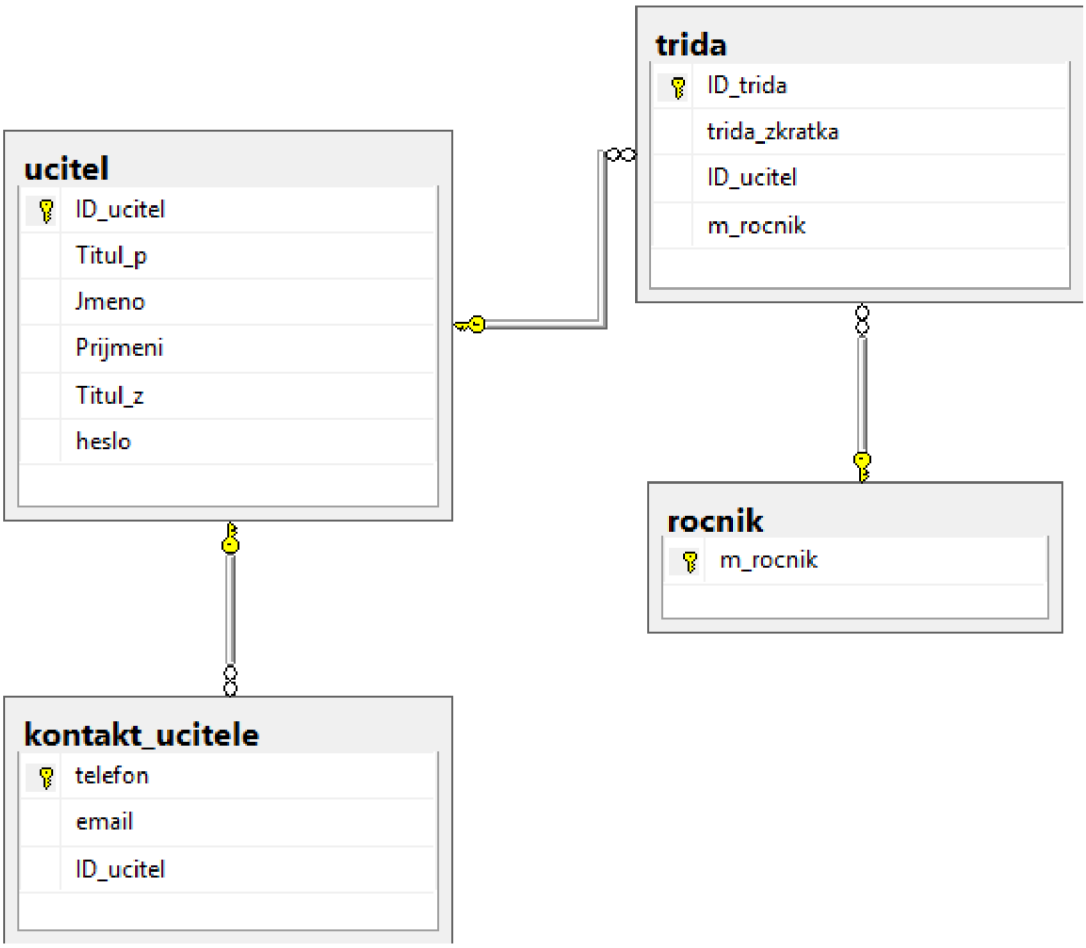
Obrázek č. 9: ER Diagram - Entity (Zdroj: Vlastní zpracování)

Logický návrh

V logickém návrhu se již tvoří samotné tabulky databáze a dochází zde ke kontrole tabulek s využitím normalizačních pravidel a kontrole návrhu budoucími uživateli systému, které zastupuje školní správce archivu. Na základě datového slovníku je poté vytvořen finální ER Diagram, který bude rozdělen do několika částí, kde budou detailně popsány entity databáze.

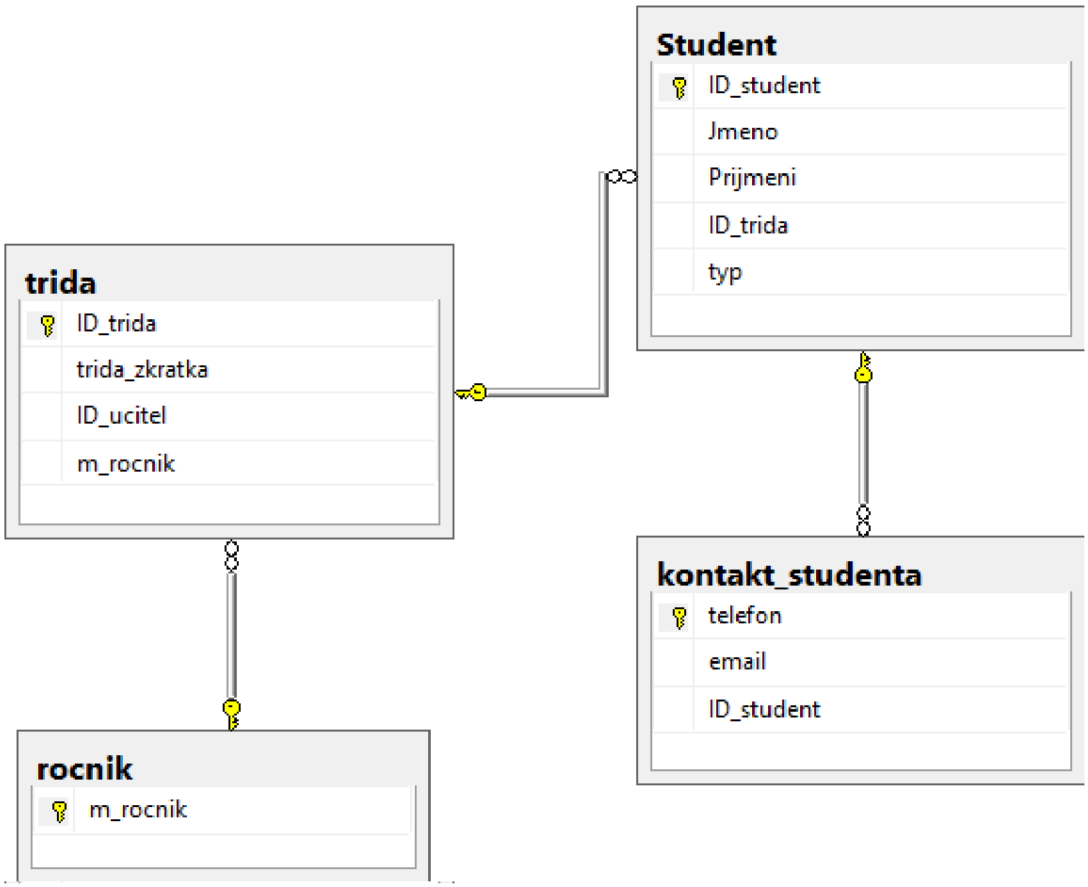


Obrázek č. 10: Finální ER Diagram (Zdroj: Vlastní zpracování)



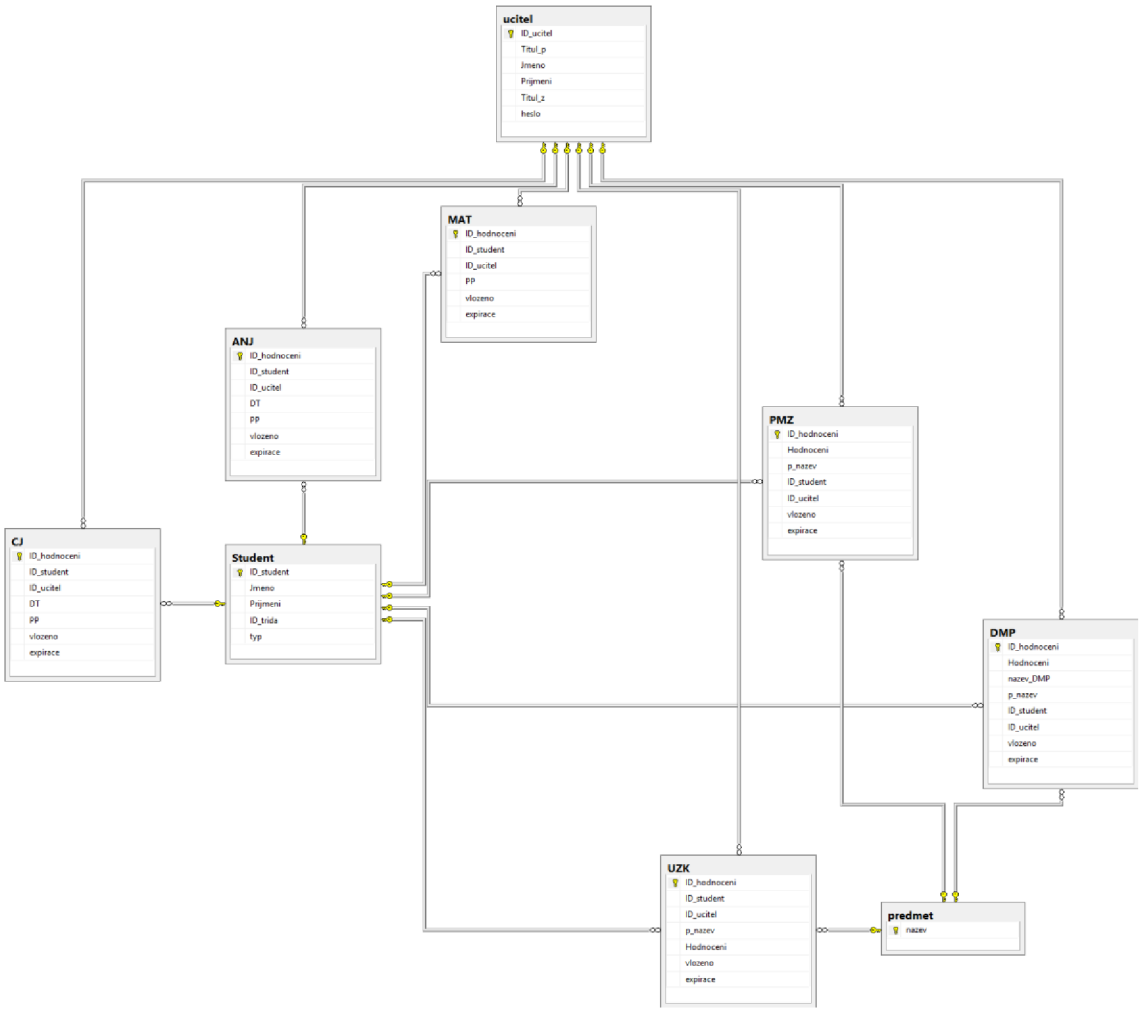
Obrázek č. 11: Evidence učitelů (Zdroj: Vlastní zpracování)

Primárním klíčem tabulky učitel bylo zvoleno ID učitele. Spolu s tímto ID a heslem se budou učitelé přihlašovat do databáze, tak aby se do ní nedostal nikdo nepovolaný. Poté, co se vytvoří nový učitel s jeho kontaktem, tak se může vytvořit záznamy o ročnících a jednotlivých třídách v nich. Tabulku třída, lze naplnit až po vytvoření ročníků a učitelů, protože obsahuje cizí klíče z těchto tabulek. Tabulka ročníků je samostatná, protože v každém ročníku může být více tříd a názvy těchto tříd se v čase opakují.



Obrázek č. 12: Evidence studentů (Zdroj: Vlastní zpracování)

Primárním klíčem tabulky student je opět ID, obdobně jako u tabulky učitelů. Po vytvoření nového studenta a jeho kontaktů, lze studenta přiřadit do již vytvořených tříd, které jsou přiřazeny k jednotlivým ročníkům. Při vytváření studenta se bude volit, jestli je to student technického lycea anebo standardních učebních oborů. Kontakty studenta jsou uloženy do samostatné tabulky proto, že každý student může uvést více těchto kontaktů.



Obrázek č. 13: Hodnocení studentů (Zdroj: Vlastní zpracování)

Poté, co jsou studenti vytvořeni a přiřazeni do tříd, se může přejít k zadávání údajů o jejich výsledcích z maturitního procesu. Nejdříve je nutné vložit jednotlivé předměty do tabulky předmětů. Poté se vkládají údaje do tabulky DMP, což je zkratka pro dlouhodobé maturitní práce, které vypracovávají studenti technických lyceí. Do tabulky PMZ, což je zkratka pro praktickou maturitní zkoušku, se vkládají údaje studentů ostatních studijních oborů. Poté se vkládají údaje všech studentů podle toho, jaké předměty mají zvoleny, do tabulek CJ, ANJ a MAT, což představuje předměty státní maturity z českého jazyka, anglického jazyka a matematiky. Český jazyk je povinný a mezi anglickým jazykem a matematikou si student vybírá. Jako poslední se vkládají údaje do tabulky UZK, což představuje školní část maturitní zkoušky. Všechny tabulky s hodnocením obsahují položku vloženo a expirace, což představuje data, kdy bylo hodnocení vloženo a kdy

může být z archivu odstraněno. O tyto data se budou starat databázové spouště, které se aktivují při vložení nového záznamu do tabulky.

Fyzický návrh

V této části návrhu databáze se tvoří již samotný kód v jazyce SQL v konkrétním DBMS. Kompletní zdrojový kód bude k nalezení v příloze č. 1.

4. Vlastní návrh řešení

V této části bude navrženo řešení analyzovaného problému. Nejprve bude předvedeno navržení uživatelského rozhraní včetně slovního popisu jednotlivých procesů. Poté budou jednotlivé procesy popsány pomocí vývojových diagramů. Následně bude podrobně rozebrán návrh uživatelského rozhraní k této databázi. Na závěr této kapitoly bude provedeno ekonomické zhodnocení a budou shrnuty přínosy řešení.

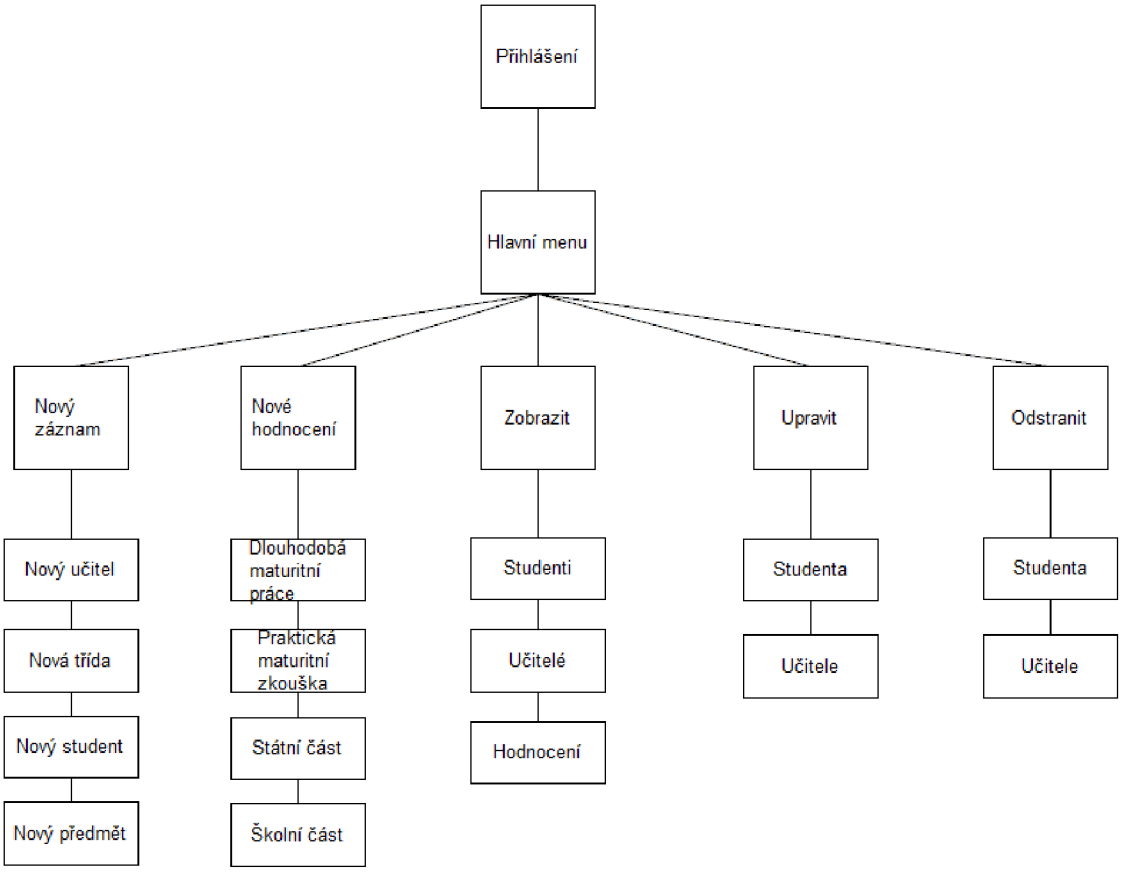
4.1 Shrnutí požadavků vyplývajících z analýzy

Během analýzy současného stavu byly požadavky na databázi i aplikační rozhraní konzultovány se správcem školního archivu. Z analýzy vyplynulo, že stěžejním požadavkem je navrhnout databázi a k ní uživatelské rozhraní, které zpřehlední a v budoucnu i nahradí správu školního archivu, který je v současné době v papírové formě v podobě kartotéky. Dalším z požadavků vyplývající z analýzy je navrhnout k této databázi uživatelsky přívětivé prostředí, tak aby se dalo s databází jednoduše a přehledně pracovat. Aplikace bude pracovat s údaji o žácích školy, kteří své studium úspěšně dokončili a složili všechny náležitosti maturitní zkoušky. Nejde opomenout ani požadavek na minimální náklady na provoz aplikace (školení, platforma apod.)

Jako platforma pro vytvoření databáze bylo vybráno prostředí MS SQL Server 2014 z důvodu kompatibility se stávajícím školním softwarem. Pro návrh uživatelského rozhraní byl vybrán Microsoft Visual Studio 2013 rovněž z důvodu kompatibility se stávajícím školním softwarem i s MS SQL Server 2014. Dané platformy jsou vhodné, aby škole nevznikly další dodatečné náklady. Oba tyto softwary jsou volně dostupné studentům fakulty podnikatelské oboru informační management.

4.2 Grafické znázornění úloh

Na následujícím obrázku bude graficky znázorněna dekompozice úloh základního členění programu. Jednotlivé procesy jsou řazeny do hlavního menu podle podobnosti a svého zaměření.



Obrázek č. 14: Grafické zobrazení struktury programu (Zdroj: Vlastní zpracování)

4.3 Slovní popis procesů

Slovní popis jednotlivých procesů popisuje, co proces vykonává a s jakými daty pracuje.

4.3.1 Přihlášení

Formulář přihlášení je vstupní branou do aplikace. Uživatel musí zadat své ID a heslo jinak ho aplikace nepustí dále. Jedná se o důležitý zabezpečovací prvek, jelikož aplikace obsahuje osobní údaje. Formulář je také odolný vůči technice SQL Injection, což je podvodné podsunutí sql kódu, dle vůle hackera.

4.3.2 Nový záznam/nový učitel

Pro vložení nového záznamu o učiteli, musí uživatel aplikace vyplnit potřebné údaje. Jedná se o základní informace o učiteli, jako jsou jméno, příjmení, dosažené vzdělání a kontaktní informace učitele. Je rovněž potřeba vyplnit heslo, kterým se poté bude učitel do aplikace přihlašovat. Před samotným vložením údajů proběhne kontrola, jestli jsou všechna povinná pole vyplněna a zda se již učitel v databázi nevyskytuje. Pokud vše proběhne v pořádku, záznam se uloží do databáze. Uživatel je informován o úspěšném či neúspěšném pokusu o vložení.

4.3.3 Nový záznam/nová třída

Pomocí tohoto formuláře se vkládá do databáze nová třída. Uživatel opět musí vyplnit veškeré potřebné informace, jinak ho program nepustí dále. Jedná se o název třídy, maturitního ročníku, do kterého náleží a přiřazení třídního učitele. Jakmile jsou vyplněny veškeré potřebné informace a kontrola údajů proběhne v pořádku, záznam se uloží do databáze. Uživatel je opět informován o výsledku vložení.

4.3.4 Nový záznam/Nový student

V tomto formuláři se vkládají data o nových studentech. Vyplní se údaje jako jméno a příjmení, kontaktní informace. Poté se student přidělí do příslušné třídy a maturitního ročníku. Také se zde volí typ studia, jestli běžné studium nebo technické lyceum. Po zadání údajů opět proběhne kontrola vstupních dat, a jestli vše proběhne korektně, data se uloží a uživatel je informován o výsledku.

4.3.5 Nový záznam/nový předmět

V tomto formuláři se vyplňují vyučované předměty na škole. Tyto vyplněné předměty se poté v dalších formulářích přiřazují k maturitnímu hodnocení. Než se údaje uloží do databáze, proběhne kontrola, jestli už se daný předmět v databázi nenachází. Uživatel je informován o výsledku vložení.

4.3.6 Nové hodnocení/Dlouhodobá maturitní práce

Pro vložení nového hodnocení je třeba, aby uživatel vyplnil všechny požadované informace v sekci Nový záznam. Poté je možno přejít ke vložení nového hodnocení dlouhodobé maturitní práce, kterou vypracovávají žáci technického lycea. Vyplňuje se zde ID studenta, název práce, z jakého předmětu je práce psána, známka a osoba, která hodnocení provedla. Po kontrole všech údajů se data zapíší do databáze a uživatel je informován o výsledku procesu.

4.3.7 Nové hodnocení/Praktická maturitní zkouška

Tento formulář je svou funkčností velice podobný předcházejícímu. Vyplňují se zde podobné údaje a opět je zde provedena kontrola a výsledek procesu.

4.3.8 Nové hodnocení/Státní část

Tento formulář obstarává ukládání údajů státní části maturitní zkoušky. Při kliknutí na položku v menu se formulář větví na tři pod formuláře. Jedná se o státní maturitu z českého jazyka, anglického jazyka nebo matematiky, podle zaměření studenta. Formuláře pro český a anglický jazyk jsou podobné, skládají se z údajů o studentovi, hodnocení písemné práce a didaktického testu a o hodnotící osobě. Formulář věnující se matematice na rozdíl od dvou výše zmíněných neobsahuje položku didaktický test. Data jsou opět zkontrolována, a když vše proběhne v pořádku, data se vloží do databáze a je vypsána informační zpráva o zapsání.

4.3.9 Nové hodnocení/Školní část

Při vkládání nového hodnocení školní části maturitní zkoušky se vyplňují data o studentovi a čtyři předměty ze seznamu předmětů vyučovaných na škole. Z každého předmětu je udělena známka a hodnotící osoba. Nakonec je automaticky přidáno datum vložení do databáze a pomocí databázových spouští je k tomuto datu přičteno určitý počet let, po které se tyto údaje musejí zachovat. Stejný princip se vkládáním dat a hlídáním expirační doby je aplikován v celé sekci Nové hodnocení.

4.3.10 Zobrazit/Studenti

Tento formulář má na starosti zobrazování vložených dat a výsledků o studentech. Formulář se po rozkliknutí větví na tři podformuláře, každý s jinou funkčností. První zobrazuje všechny studenty, dle vybraných kritérií pod sebe do řádků. Na výběr jsou možnosti: všichni studenti školy, studenti technického lycea anebo studenti ostatních oborů. V druhém podformuláři lze vyhledat konkrétního studenta, dle zadaného id, a detailnější informace o požadovaném studentovi. Třetí podformulář zobrazuje studenty rozdělené podle jednotlivých tříd, do kterých náleží. V těchto třech formulářích také probíhá určitá kontrola dat a to taková, jestli hledaný student nebo třída vůbec existují. Jestli ano je vypsán výsledek dotazu a jestli ne, je vypsáno upozornění, že zadaný student nebo třída neexistují.

4.3.11 Zobrazit/Učitelé

Zmiňovaný formulář zobrazuje konkrétního učitele a veškeré dostupné informace o něm, v závislosti na vloženém id. Princip je podobný jako u zobrazování údajů o studentech. Kontrola taktéž probíhá na základě zjištění existence učitele.

4.3.12 Zobrazit/Hodnocení

Stěžejní formulář aplikace, které má stejnou strukturu jako sekce nové hodnocení. Dělí se na dlouhodobou maturitní práci, praktickou maturitní zkoušku, státní část zkoušky a jednotlivé předměty a také školní část maturitní zkoušky. Veškerá zadaná data, která se týkají hodnocení, jsou k nalezení v tomto formuláři. Jako v každém formuláři, je zde zavedena kontrola správných vstupů a v případě problému dá aplikace uživateli vědět, co je špatně. Tento formulář a jeho podformuláře hlídají i datum, kdy můžou být tyto informace skartovány.

4.3.13 Upravit/Studenta

Jedná se formulář, pomocí kterého lze upravit data jednotlivých studentů. Pokud uživatel zjistí, že zadaná data jsou nepřesná nebo s chybou, může využít možnosti upravení těchto údajů. Uživatel zadá ID záznamu, záznam je načten a je možné ho změnit podle potřeby. Po úpravě jej může uživatel opět uložit do databáze. Při novém ukládání probíhají stejné

kontrolní procesy jako při prvotním vložení. Formulář Upravit/Učitele je stejný jako tento a proto nebude podrobněji popisován.

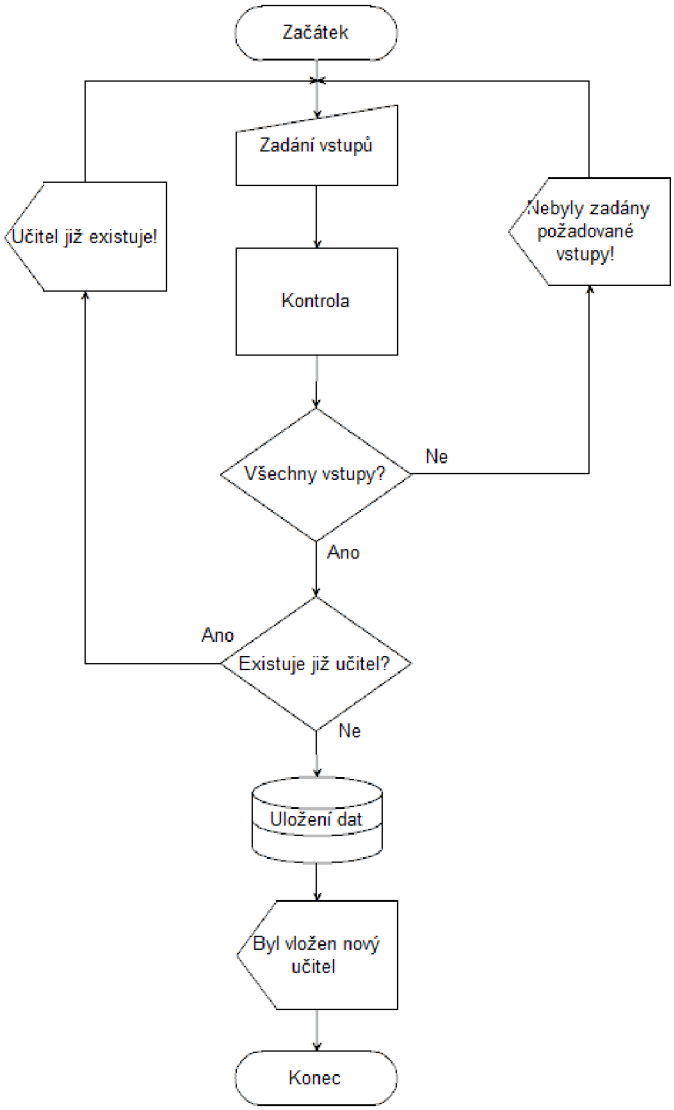
4.3.14 Odstranit/Studenta

Jestliže je uživatel s některým záznamem úplně nespokojen nebo záznam obsahuje více chyb nebo vůbec nedopovídá realitě, bude možné jej vymazat. Zadává se zde opět ID záznamu, jako jeho jednoznačný identifikátor. Nejprve se odstraní záznamy v relačně svázaných tabulkách a až poté doje ke smazání samotného záznamu. V případě úspěchu či neúspěchu je uživatel informován. Formulář Odstranit/Učitele je stejný a tudíž zde nebude detailněji vysvětlován.

4.4 Vývojové diagramy stěžejních procesů

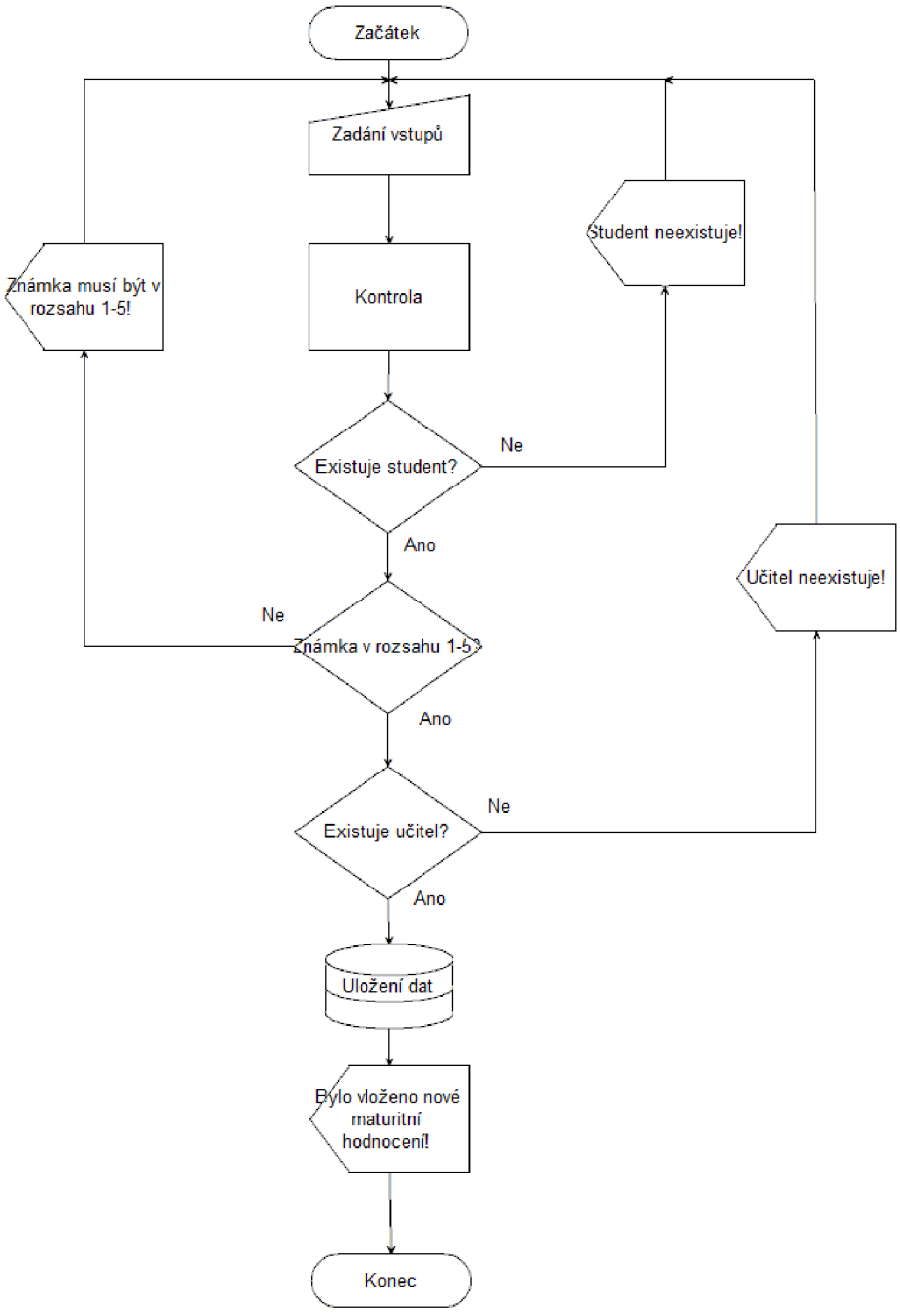
Za účelem kvalitního návrhu samotné uživatelské aplikace bude sestrojeno několik vývojových diagramů zaměřující se na stěžejní činnosti programu. Z každé sekce aplikace zmíněné výše, bude vybrán jeden nejdůležitější vývojový diagram.

4.4.1 Vložení nového učitele



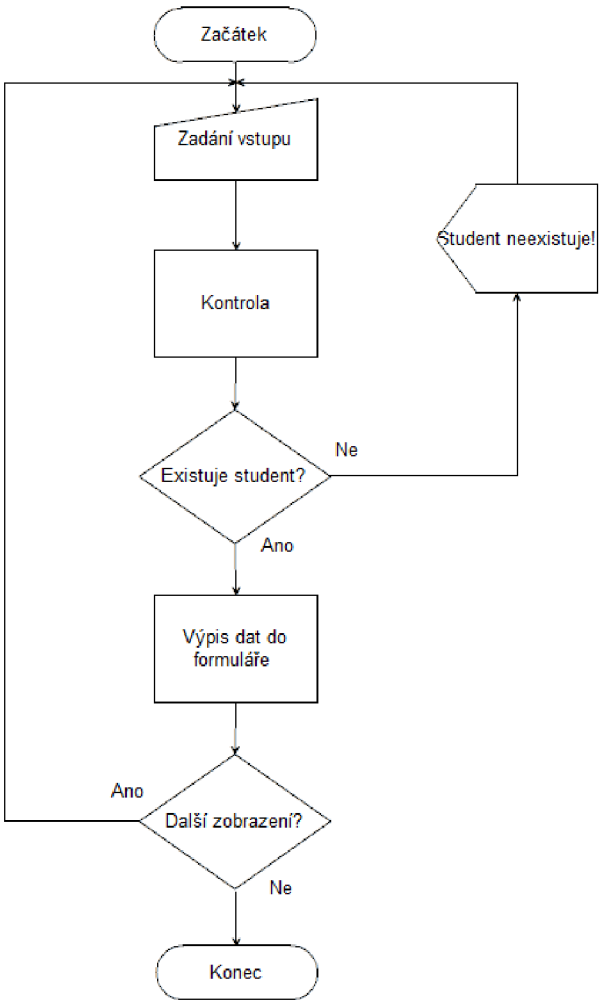
Obrázek č. 15: Vývojový diagram procesu vložení nového učitele (Zdroj: Vlastní zpracování)

4.4.2 Vložení nového maturitního hodnocení



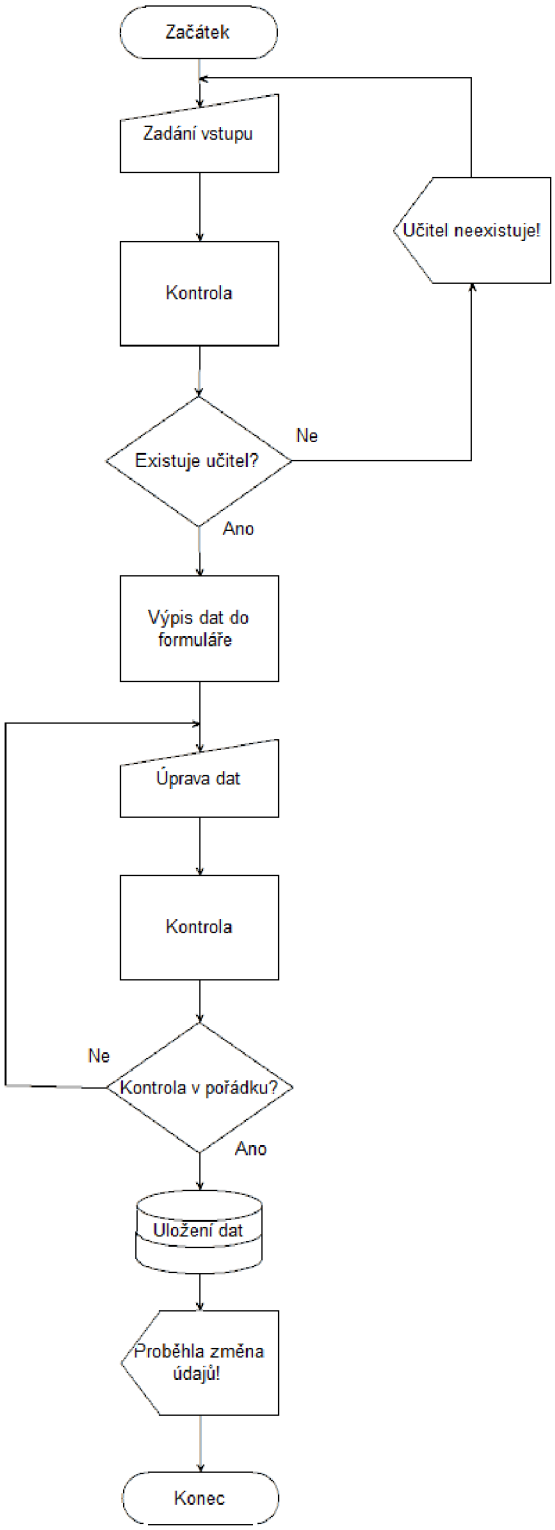
Obrázek č. 16: Vývojový diagram procesu vložení nového maturitního hodnocení (Zdroj: Vlastní zpracování)

4.4.3 Zobrazení maturitního hodnocení



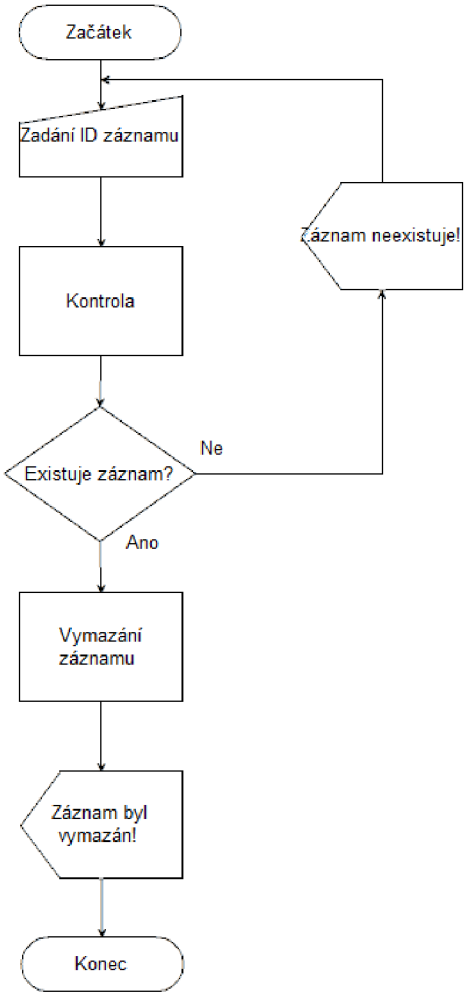
Obrázek č. 17: Vývojový diagram procesu zobrazení maturitního hodnocení (Zdroj: Vlastní zpracování)

4.4.4 Úprava učitelských dat



Obrázek č. 18: Vývojový diagram procesu úpravy učitelských dat (Zdroj: Vlastní zpracování)

4.4.5 Odstranění záznamu

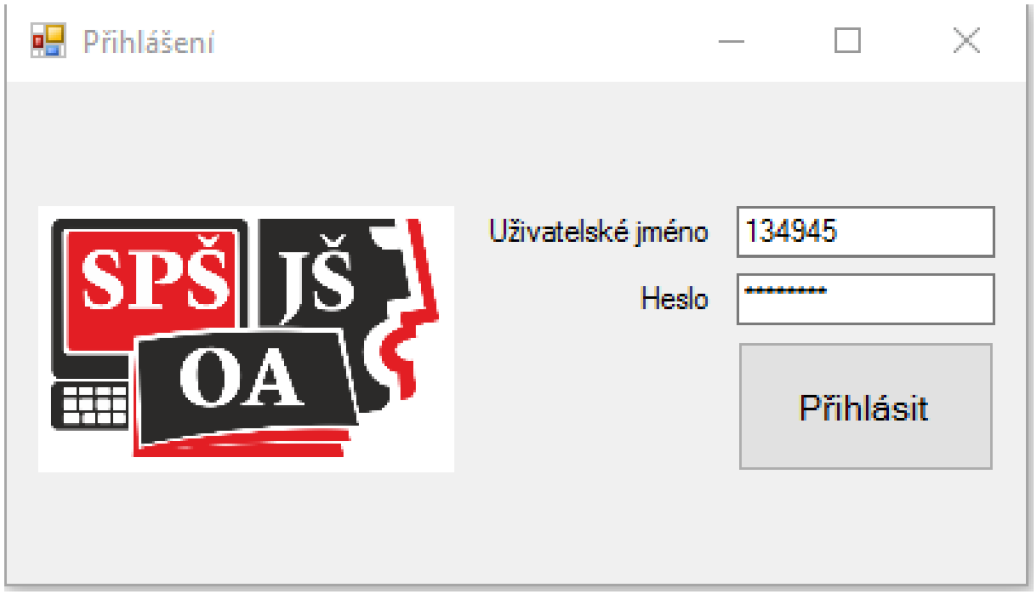


Obrázek č. 19: Vývojový diagram procesu odstranění záznamu (Zdroj: Vlastní zpracování)

4.5 Představení aplikace

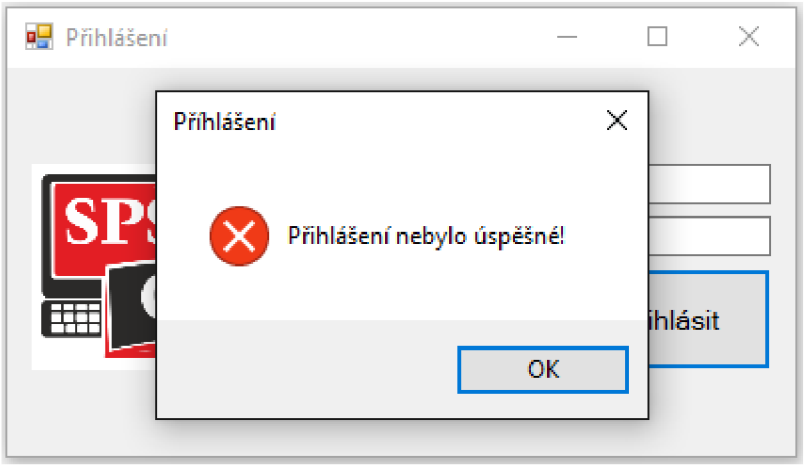
Z požadavků na aplikaci vyplývá, že škola požaduje, aby byla finální verze aplikace přehledná a jednoduchá na obsluhu. Vzhledem k těmto požadavkům bude výsledná verze formulářovou aplikací. Aplikace bude mít v záhlaví přehledné rozbalovací menu naprogramované podle grafické struktury aplikace. Aplikace bude spustitelná jako exe soubor. Po spuštění se zobrazí přihlašovací formulář, tak aby do databáze měli přístup pouze odpovědné osoby. Po přihlášení se uživatel dostane do hlavního menu, kde si

z rozbalovacího menu vybere typ procesu, který program vykoná. Samotné rozhraní má vlastnosti aplikací pro desktopové Windows 10.



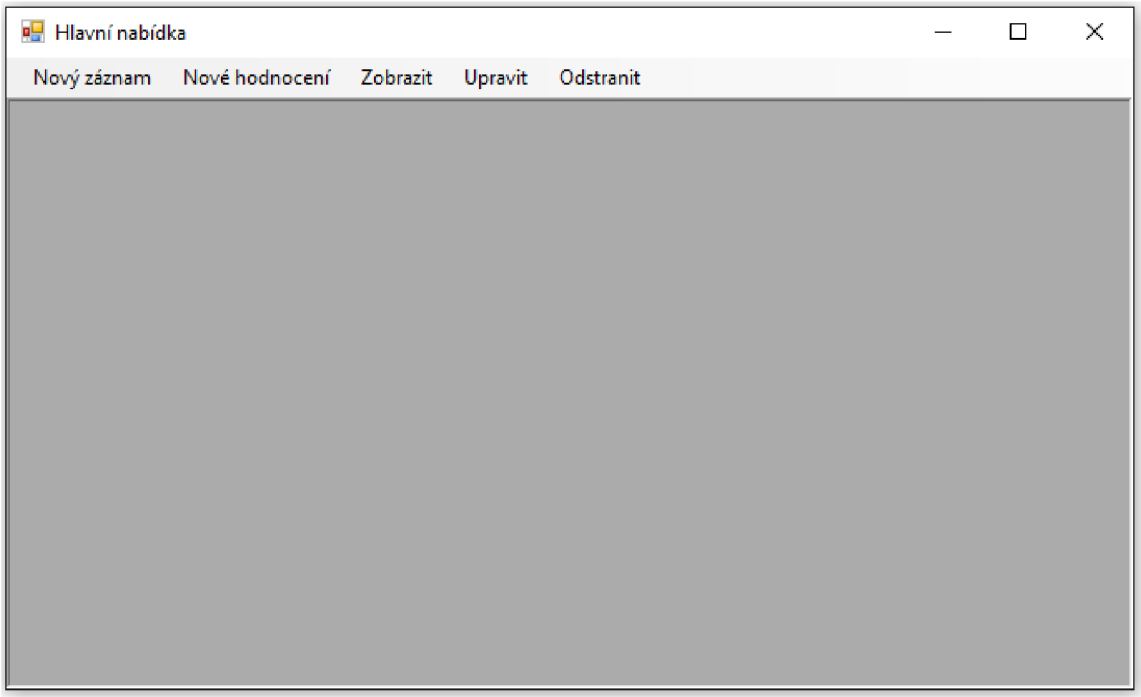
Obrázek č. 20: Přihlášení do aplikace (Zdroj: Vlastní zpracování)

Na základě přihlášení vyskočí uživateli buď zpráva o úspěšném přihlášení nebo o tom, že se přehlášení nezdařilo. Přihlašovací formulář rozeznává v heslu velká a malá písmena a je tedy tzv. Case sensitive.



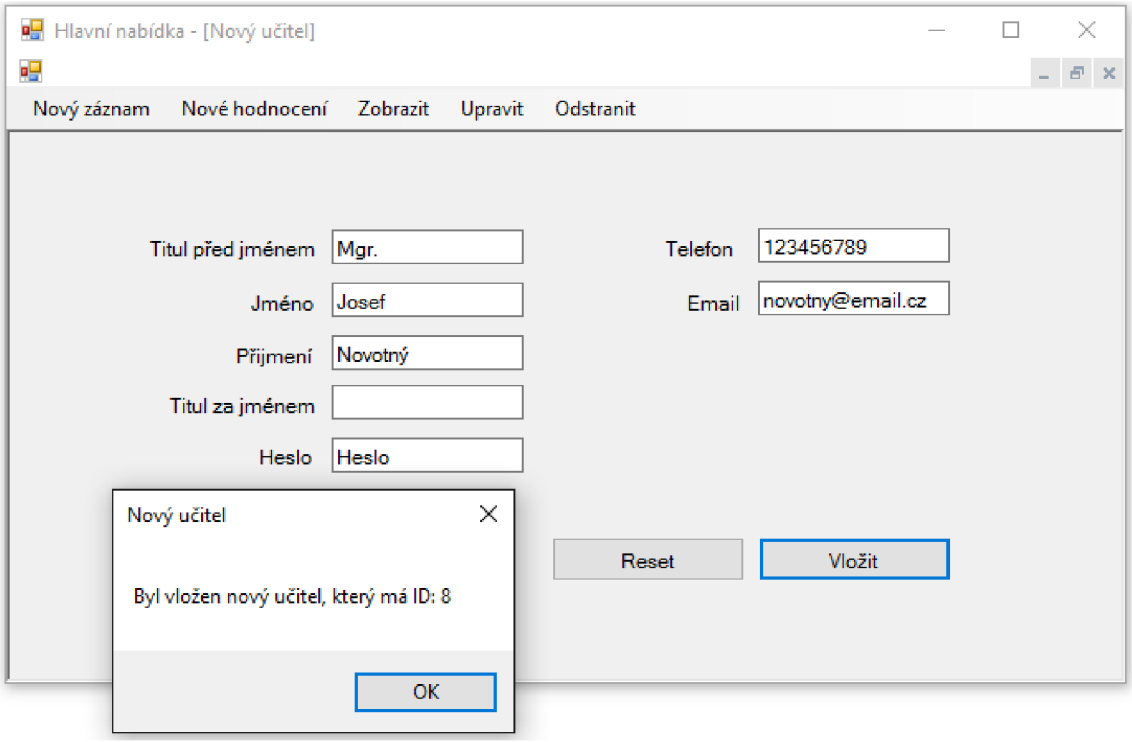
Obrázek č. 21: Neúspěšné přihlášení (Zdroj: Vlastní zpracování)

Aby se předešlo možnosti útoku na databázi pomocí SQL Injection, nepřistupuje databáze k údajům přímo, ale pomocí parametrů, které načítá do proměnných. Proto když se hacker pokusí do textboxů podsunout podvodný příkaz, aplikace ho dále nepustí.



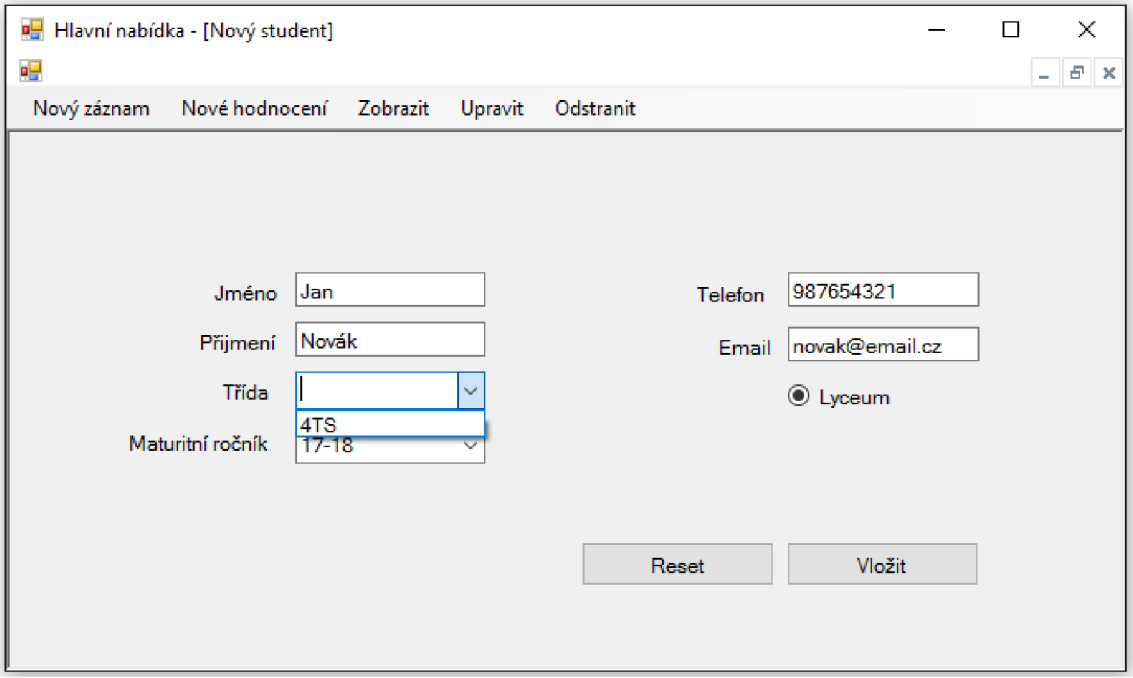
Obrázek č. 22 Hlavní menu (Zdroj: Vlastní zpracování)

V hlavním menu, které je jednoduché a přehledné na ovládání, si uživatel vybere v záhlaví, pomocí rozbalovacího menu, typ úlohy a po kliknutí se nový formulář objeví v těle hlavního formuláře. Toto je umožněno tím, že hlavní nabídka, je nastavena jako rodičovský formulář. Formulář je naprogramován přesně podle grafického návrhu. Neobsahuje tlačítko ukončit program, stačí v kterémkoliv okamžiku stisknout křížek a program se řádně ukončí. Záhlaví hlavní nabídky obsahuje možnost volby z celkem 25 podformulářů, ať už se jedná o vkládání nových údajů, zobrazení stávajících nebo změnu či odstranění nevyhovujících záznamů.



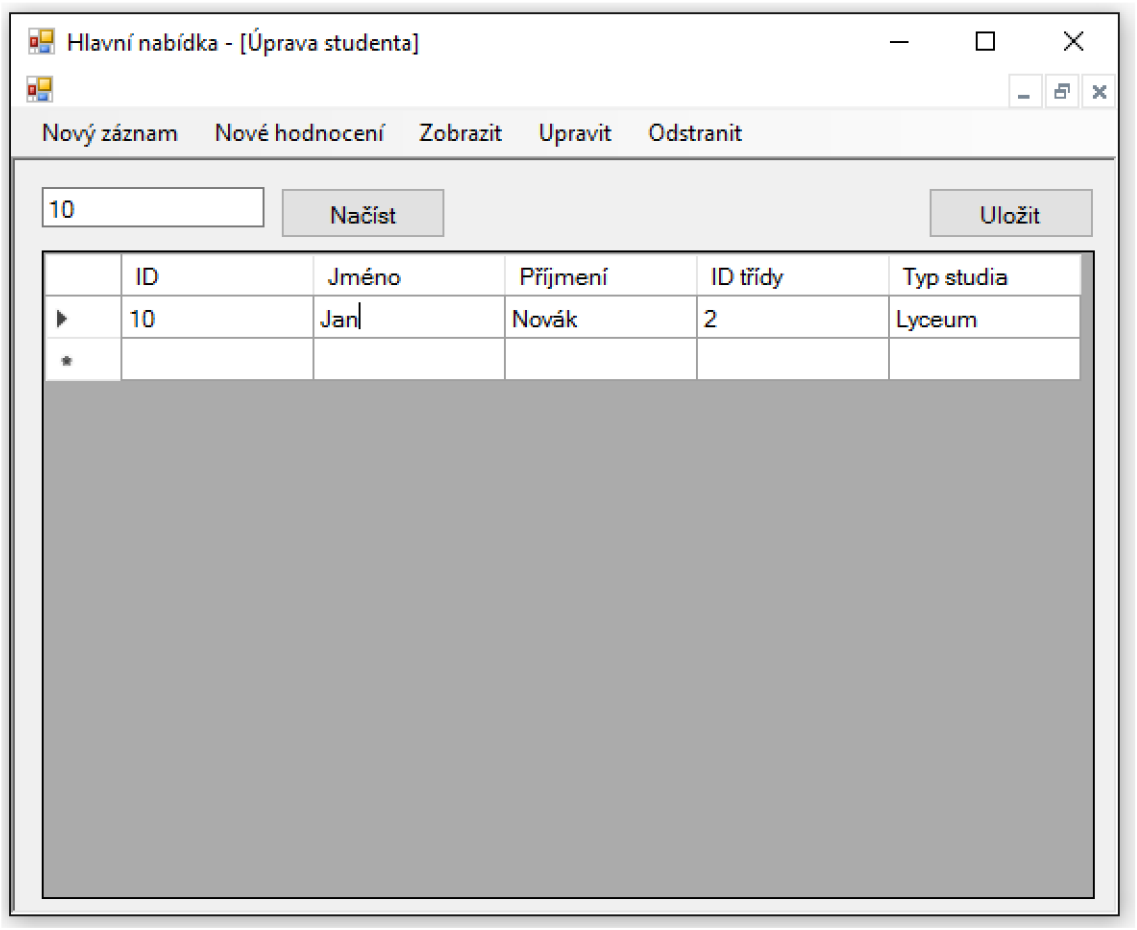
Obrázek č. 23 Vložení nového učitele (Zdroj: Vlastní zpracování)

Po kliknutí na nový záznam a v něm na nového učitele se otevře formulář pro vkládání nových dat. Tento formulář obsahuje textboxy, číselníky a textboxy s maskou, do kterých lze zadat jen určité znaky anebo si lze vybrat z již předdefinovaných údajů. Po vyplnění všech potřebných dat a kliknutí na tlačítko vložit, program data zkontroluje a poté vypíše hlášku, že byl nový záznam vložen anebo došlo k chybě a program na to uživatele upozorní. Každý formulář obsahuje tlačítko reset, kdy se po kliknutí vymažou všechna zadávací pole, tak ať je formulář připraven pro případné zadání nového záznamu. S jednotlivými formuláři lze pracovat jako se standardními okny ve windows 10. Tedy minimalizovat, schovat na lištu a vypnout formulář. V záhlaví hlavní nabídky se vždy zobrazí cesta, tak ať uživatel ví, v které části programu se zrovna nachází.



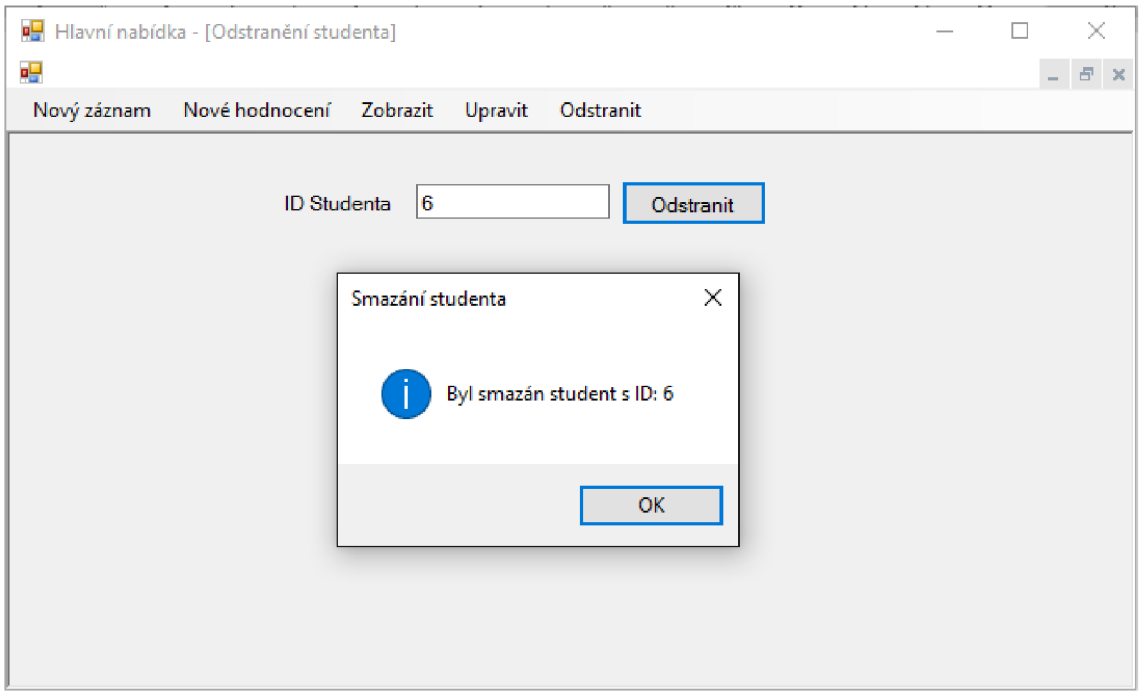
Obrázek č. 24: Vložení nového studenta (Zdroj: Vlastní zpracování)

V tomto formuláři, který slouží pro vkládání nového studenta, se nejdříve vyplní jméno a příjmení, poté se z rozbalovacího seznamu vybere požadovaný maturitní ročník a ten je provázán s rozbalovacím seznamem tříd, kde se zobrazí pouze třídy, které se v daném maturitním ročníku vyskytují, tak ať vkládající osoba má ušetřeno, co nejvíce práce a navíc je tento způsob řešení přehledný. Formulář také obsahuje radiobutton, kde se volí, jestli student studuje technické lyceum nebo standardní obory. Po kliknutí na tlačítko vložit proběhne kontrola dat a poté se tyto data uloží do databáze.



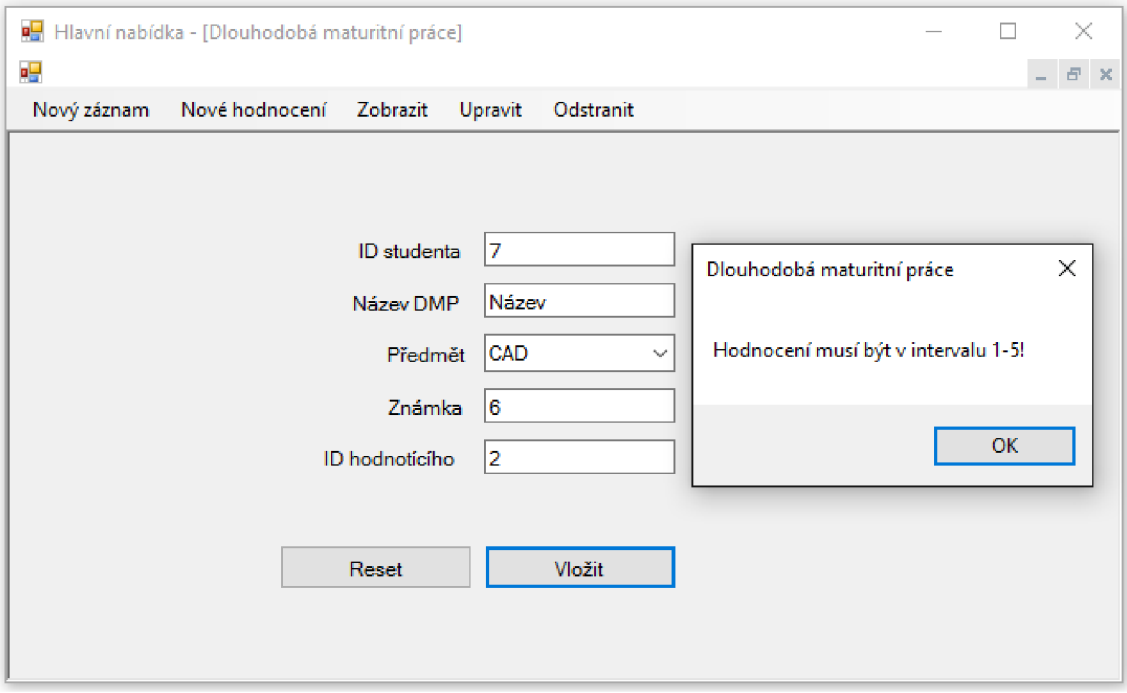
Obrázek č. 25: Úprava studenta (Zdroj: Vlastní zpracování)

V tomto jednoduchém formuláři se po zadání ID zobrazí požadovaný záznam. V tomto záznamu je možno měnit vybrané informace tím, že se klikne do požadované buňky a záznam se přepíše. Poté, co je záznam přepsán a po stisku tlačítka uložit, se provede kontrola vstupů a záznam se uloží do databáze.



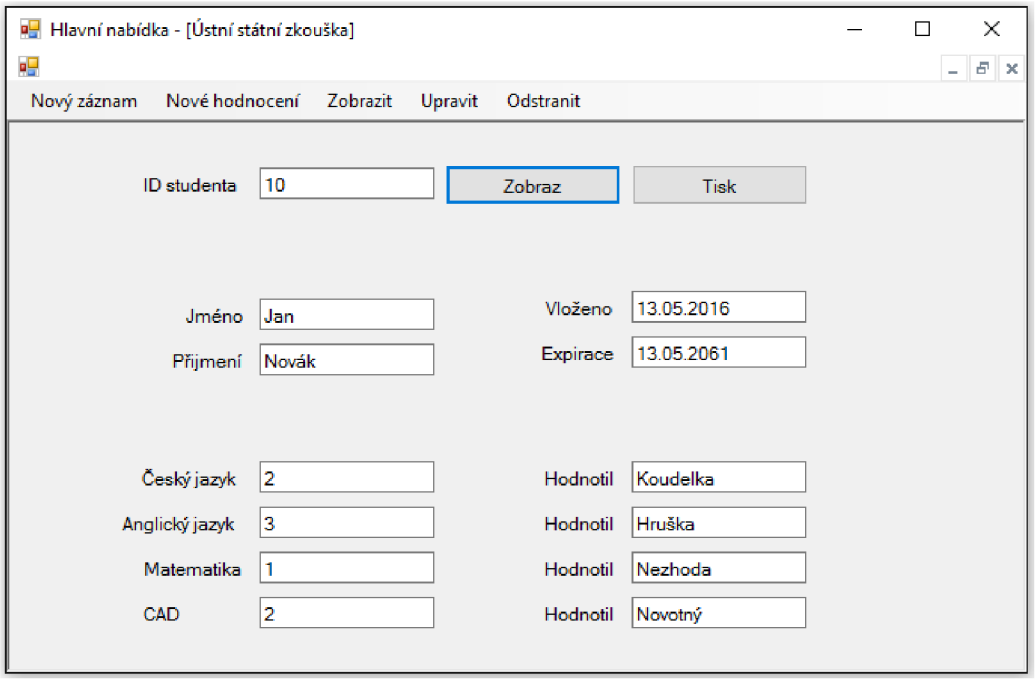
Obrázek č. 26: Odstranění studenta (Zdroj: Vlastní zpracování)

Odstranění záznamu probíhá tak, že se do textboxu vloží ID záznamu, který chceme smazat, proběhne kontrola, zda se záznam vůbec v databázi vyskytuje. Nejprve se odstraní záznamy z relačně propojených tabulek a až poté z tabulky samotné. Nakonec se uživateli vypíše zpráva s výsledkem smazání.



Obrázek č. 27 Vložení Dlouhodobé maturitní práce (Zdroj: Vlastní zpracování)

Při vkládání nového hodnocení se vyplní požadované údaje, vybere se předmět z rozbalovacího seznamu. Po stisku tlačítka vložit se provede kontrola studenta, známky a hodnotícího a poté se záznam uloží v databázi.



Obrázek č. 28: Zobrazení hodnocení (Zdroj: Vlastní zpracování)

Zobrazení hodnocení ústní státní zkoušky je stěžejní část programu. Po zadání ID studenta se přepíšou názvy popisků textboxů na názvy předmětů, ze kterých student provedl školní část maturitní zkoušky. V příslušných textboxech se zobrazí jejich hodnocení. Ve druhém sloupci se zobrazí také informace o tom, kdo hodnocení provedl. Program také zobrazuje, kdy bylo hodnocení provedeno a kdy bude možné záznamy skartovat. Formulář nabízí i možnost exportu do formátu pdf anebo rovnou tisk na papír.

4.6 Ekonomické zhodnocení a přínos řešení

Přínos návrhu nové aplikace pro správu školního archivu Střední průmyslové školy ve Frýdku-Místku spočívá ve zrychlení, zjednodušení a zkvalitnění správy archivu. Tyto přínosy jsou realizovány díky uživatelskému rozhraní, které pracuje s databází.

Naprogramované funkce a procesy v celkem 26 formulářích umožňují zaměstnancům školy lehce a přehledně vkládat, upravovat a mazat data v archivu a především rychle zobrazovat potřebné informace. Díky tomu je dosáhnuto úspory času zaměstnanců, kteří se školním archivem pracují a tudíž se budou moci věnovat i jiným činnostem.

Jelikož se jedná o podpůrnou aplikaci v oboru školství, není aplikace zaměřena na tvorbu zisku. Není to ani potřeba, protože se jedná o státní školu, která jako taková nepotřebuje generovat žádný zisk. Je ale nutné, aby aplikace plnila svou funkci.

Co se týče nákladů na aplikaci, tak ty budou posuzovány z pohledu TCO, což je zkratka pro celkové náklady na vlastnictví aplikace. TCO vyjadřuje kompletní náklady na investici a její provoz, zohledňující nejen pořizovací cenu, ale také výdaje vznikající s provozem aplikace.

Vývoj samotné aplikace je nejnákladnější částí práce. Hodinová sazba za práci programátora specializované společnosti se pohybuje v rozmezí od 1000 Kč po 2500 Kč dle zkušeností. V rámci dobrých vztahů se školou byla dohodnuta hodinová sazba 750 Kč. Na návrh aplikace bylo zapotřebí 15 hodin práce. Na vývoj aplikace bylo zapotřebí 30 hodin práce. K efektivnímu používání aplikace bude zapotřebí čtyřhodinového školení příslušných zaměstnanců. Celkové náklady na vlastnictví aplikace po dobu tří let jsou

tedy 45 750 Kč. Celková částka se tedy vešla do požadovaného rozpočtu stanoveného školou.

Tabulka č. 2: TCO na 3 roky v Kč

Činnost	**Sazba Kč/h**	**Počet hodin**	**Celková částka**
Návrh aplikace	750	15	11250
Vývoj aplikace	750	30	22500
Školení	750	4	3000
Údržba a servis	750	12	9000
Celkem			**45750**

5. Závěr

Cílem této diplomové práce bylo navrhnout uživatelské rozhraní k SQL databázi, pomocí kterého se bude přistupovat ke školnímu archivu. Důraz byl kladen na uchovávání údajů, které se používají pro výpisy maturitních protokolů. Aplikace také uchovává údaje o dlouhodobých či praktických maturitních pracích.

Vlastní návrh řešení vychází z analýzy současného stavu, kde byl popsán současný stav ICT školy a aplikací, které již škola využívá. Dále byl v této části proveden rozbor problému, popis procesů archivace i popis legislativního prostředí. Na závěr byly analyzovány požadavky na výslednou aplikaci a proveden rozbor SQL databáze, nad kterou aplikace bude fungovat.

Na základě této analýzy bylo navrhnuto vlastní řešení uvedeného problému. Nejdříve byl proveden grafický návrh řešení a poté slovně popsány jednotlivé procesy v aplikaci. Na základě těchto popisů, byly vytvořeny vývojové diagramy a podle nich a požadavků školy byla vytvořena uživatelská aplikace, která zjednodušuje, zrychluje a zkvalitňuje práci se školním archivem. Aplikace byla navržena tak, ať se s ní pracuje jednoduše a intuitivně, bez potřeby dlouhého a složitého školení zaměstnanců. Tímto byly naplněny cíle této práce.

Seznam použitých zdrojů

[1] BEGG, C., R. HOLOWCZAK a T. CONOLLY. *Mistrovství - Databáze: Profesionální průvodce tvorbou efektivních databází*. Praha: Computer Press, 2009. 584 s. ISBN 978-80-251-2328-7

[2] BRUCKNER, T., J. VOŘÍŠEK, A. BUCHALCEVOVÁ a kol. *Tvorba informačních systémů: Principy, metodiky, architektury*. Praha: Grada Publishing a.s., 2012. 360 s. ISBN 978-80-247-4153-6.

[3] FUCHS, J. a A. BARCHFELD. *Visual Basic Velká kniha řešení*. Brno: Computer Press, 2010. 744 s. ISBN 978-80-251-2212-9.

[4] HALVORSON, M. *Microsoft Visual Basic Krok za krokem. 1. vyd.* Brno: Computer Press, 2015. 648 s. ISBN 978-80-251-4412-1.

[5] SCHWALBE, K. *Řízení projektů v IT: Kompletní průvodce.* Praha: Computer Press, 2011. 632 s. ISBN 978-80-251-2882-4.

[6] INFORMAČNÍ VĚDA. *Modul č. 4. Informační a znalostní management.* Informacniveda.cz [online]. 2009 [cit. 2017-04-16] Dostupné z: www.informacniveda.cz/dwn/1003/1167_IZM_Havlickova_final.pdf

[7] KOCH, M. *Datové a funkční modelování.* Brno: CERM, 2004.
ISBN 80-214-2724-8.

[8] LUHAN, J. *Databázové systémy a metodologie návrhu.* Vut.cz [online]. Vysoké učení technické v Brně, Fakulta podnikatelská: 2011 [cit. 2017-04-17]. Dostupné z: http://luhan.comlu.com/DBS/doc/P/02/02.pdf

[9] LACKO, L. *1001 tipů a triků pro SQL.* Brno: Computer Press, 2011.
ISBN 978-80-251-3010-0

[10] SEDLÁK, P. *Přiměřená bezpečnost*. [2. prezentace z předmětu Management informační bezpečnosti]. Vysoké učení technické v Brně, Fakulta podnikatelská: 2013 [cit 2017-04-26].

[11] SPSOAFM. Historie. Spsoafm.cz [online]. [cit. 2017-05-01]. Dostupné z: http://www.oafm.cz/web/stredni-prumyslova-skola-2/historie-sps2

[12] ARCHIV HLAVNÍHO MĚSTA PRAHY (AHMP). Vzorový spisový a skartační řád pro základní a střední školy. Ahmp.cz [online]. 2004 [cit. 2017-05-01]. Dostupné z: www.ahmp.cz/page/docs/skoly.pdf

[13] MOODLE. Moodle. Moodle.org [online]. [cit. 2017-05-01]. Dostupné z: http://docs.moodle.org/archive/cs/Co_je_Moodle

[14] BAKALÁŘI. Bakalari. Bakalari.cz [online]. [cit. 2017-05-01]. Dostupné z: http://www.bakalari.cz/programy.aspx

[15] OPPEL, A. *SQL bez předchozích znalostí.* Brno: Computer Press, 2012. ISBN 978-80-251-1707-1.

[16] STEPHENS, R.,R. PLEW a A. D.JONES. *Naučte se SQL za 28 dní.* Brno: Computer Press, 2010. ISBN 978-80-251-2700-1.

[17] PROGRAMUJTE. Programujte. Programujte.com [online]. [cit. 2017-05-01]. Dostupné z: http://programujte.com/clanky/59-serial-visual-basic/

[18] MORKES, D. *Visual Basic 6.0*. Praha: Computer Press, 2000. ISBN 80-7226-312-9

[19] GURTLER, M. P. KOCICH. *1001 Tipů a triků pro Visual Basic.* Brno: Computer Press, 2010. 326 s. ISBN 80-7226-368-4

[20] BEŇOVÁ, J. Ústní sdělení. Střední průmyslová škola, Obchodní akademie a Jazyková škola s právem státní jazykové zkoušky, Frýdek-Místek, příspěvková organizace, 28. října 1598, 73801 Frýdek-Místek. 29.3.2017.

Seznam obrázků

Seznam tabulek

Seznam příloh

Příloha č. 1 Zdrojový kód SQL

```
use Diplomka
go
Create table ucitel
(
ID_ucitel int identity (1,1) primary key not null,
Titul_p varchar(5) default 'Neuvedeno',
Jmeno varchar(25) not null,
Prijmeni varchar(30) not null,
Titul_z varchar(5) default 'Neuvedeno',
heslo varchar(25) not null
)

Create table kontakt_ucitele
(
telefon varchar(9) primary key not null,
email varchar(25),
ID_ucitel int,
foreign key (ID_ucitel) references ucitel(ID_ucitel)
)

create table rocnik
(
m_rocnik varchar(5) primary key
)

Create table trida
(
ID_trida int identity(1,1) primary key not null,
trida_zkratka varchar(6) not null,
ID_ucitel int,
m_rocnik varchar(5),
foreign key (m_rocnik) references rocnik(m_rocnik),
foreign key (ID_ucitel) references ucitel(ID_ucitel)
)

Create table Student
(
ID_student int identity (1,1) primary key,
Jmeno varchar(25),
Prijmeni varchar(30),
ID_trida int,
typ varchar(10),
foreign key (ID_trida) references trida(ID_trida)
)

Create table kontakt_studenta
(
telefon varchar(9) primary key not null,
email varchar(25),
ID_student int,
foreign key (ID_student) references student(ID_student)
)

Create table predmet
(
nazev varchar(25) primary key not null
```

```
)

Create table DMP
(
ID_hodnoceni int identity(1,1) primary key not null,
Hodnoceni int not null,
nazev_DMP varchar(25) not null,
p_nazev varchar(25),
ID_student int,
ID_ucitel int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel),
foreign key (p_nazev) references predmet(nazev)
)

Create table PMZ
(
ID_hodnoceni int identity(1,1) primary key not null,
Hodnoceni int not null,
p_nazev varchar(25),
ID_student int,
ID_ucitel int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel),
foreign key (p_nazev) references predmet(nazev)
)

Create table CJ
(
ID_hodnoceni int identity(1,1) primary key not null,
ID_student int,
ID_ucitel int,
DT int,
PP int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel)
)

Create table ANJ
(
ID_hodnoceni int identity(1,1) primary key not null,
ID_student int,
ID_ucitel int,
DT int,
PP int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel)
)

Create table MAT
(
ID_hodnoceni int identity(1,1) primary key not null,
```

```
ID_student int,
ID_ucitel int,
PP int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel)
)

Create table UZK
(
ID_hodnoceni int identity (1,1) primary key not null,
ID_student int,
ID_ucitel int,
p_nazev varchar(25),
Hodnoceni int,
vlozeno date,
expirace date,
foreign key (ID_student) references student(ID_student),
foreign key (ID_ucitel) references ucitel(ID_ucitel),
foreign key (p_nazev) references predmet(nazev)
)
go

create trigger expirace_DMP
on DMP
for insert
as
update DMP
set expirace = DATEADD(year,5,vlozeno)
go

create trigger expirace_PMZ
on PMZ
for insert
as
update PMZ
set expirace = DATEADD(year,5,vlozeno)
go

create trigger expirace_CJ
on CJ
for insert
as
update CJ
set expirace = DATEADD(year,45,vlozeno)
go


create trigger expirace_ANJ
on ANJ
for insert
as
update ANJ
set expirace = DATEADD(year,45,vlozeno)
go


create trigger expirace_MAT
on MAT
```

```
for insert
as
update MAT
set expirace = DATEADD(year,45,vlozeno)
go


create trigger expirace_UZK
on UZK
for insert
as
update UZK
set expirace = DATEADD(year,45,vlozeno)
```