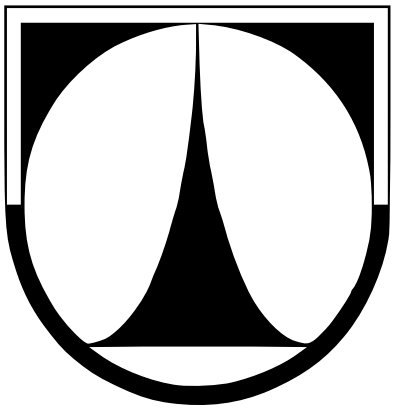
Technická univerzita v Liberci

Fakulta Mechatroniky



DIPLOMOVÁ PRÁCE

ELEKTRONICKÁ ŽÁKOVSKÁ KNÍŽKA

Information system "On-line grade book"

Autor: **Bc. Martin Žaloudek**

Vedoucí práce: **doc. RNDr. Pavel Satrapa, Ph. D.**

LIBEREC, 2009

Tato strana je v originále nahrazena zadáním diplomové práce

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé DP a prohlašuji, že **souhlasím** s případným užitím mé diplomové práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

V Liberci dne 21.5.2009

Martin Žaloudek

Poděkování

Rád bych touto cestou poděkoval docentu Pavlovi Satrapovi za vedení této diplomové práce. Dále pak vedení 1. Základní školy v Klášterci nad Ohří, zejména paní ředitelce magistře Janě Dimunové, za spolupráci při návrhu tohoto informačního systému.

Abstrakt

Práce řeší aplikaci elektronické žákovské knížky pro základní školy (dále jen EŽK), která má být alternativou ke klasické žákovské knížce v papírově podobě.

V tomto dokumentu je zprvu proveden rozbor aktuálně existujících řešení obdobných informačních systémů. Další část se zabývá návrhem aplikace a jejich možných funkcí a vlastností. Poslední část je pak věnována implementaci vybraných funkcí popsaných v druhé části.

Klíčová slova: Žákovská knížka, informační systém, HTML, PHP, SQL

Abstract

This diploma thesis solves the application of an electronic grade book for primary schools. The electronic grade book should be an option to a paper grade book.

The study consists of three parts: in the first part, there is an analysis of existing solutions of similar information systems. The second part deals with the application project and its possible functions. The last part is devoted to implementation of selected functions described in the second part.

Keywords: Grade book, information system, HTML, PHP, SQL

Obsah

1 Úvod....7
2 Analýza....8
2.1 Existující řešení....8
2.2 SWOT analýza vlastního IS EŽK z pohledu zadavatele....15
3 Návrh aplikace....17
3.1 Specifikování požadavků....17
3.2 Návrh zabezpečení....20
3.3 Entity EŽK....24
3.4 Oprávnění uživatelů....26
3.5 Školní rok a pololetí....28
3.6 Spolupráce s externím off-line klientem....29
3.7 Zálohování dat....30
4 Implementace....31
4.1 Databáze....31
4.2 Autentizační jádro....36
4.3 Ověření oprávnění....39
4.4 Zobrazovací engine....41
4.5 Zpracování akcí....43
5 Testování aplikace....46
5.1 Testování při vývoji....46
5.2 Testování zadavatelem....47
5.3 Zkušební provoz....47
6 Závěr....48
7 Literatura....49
8 Slovník pojmů a zkratek....50
9 Přílohy....52

1 Úvod

Tato diplomová práce se zabývá tématem návrhu a tvorby informačního systému „elektronické žákovské knížky[1]“.

První část objasní čtenáři aktuální situaci na trhu obdobných aplikací určených základním a středním školám. Seznámení bude probíhat formou rozboru jednotlivých dostupných aplikací. SWOT analýza pak ukáže, jaké mají existující projekty silné a slabé stránky, příležitosti a hlavně rizika.

Největší část práce bude věnována samotné aplikaci, jejíž nejdůležitější části budou v rámci této práce vytvořeny. Prvotní analýza potřeb a požadavků ukáže směr, jakým se bude ubírat návrh celého projektu. V něm pak budou popsány jednotlivé části budoucí aplikace, jejich struktura a funkce. Dále bude třeba se detailněji zabývat zabezpečením aplikace, a s tím souvisejícími otázkami přidělování oprávnění jednotlivých uživatelů. Další navazující část pak popíše vybrané důležité části implementace navrženého sytému.

Závěrečná pasáž se pak bude věnovat fázi testování, zhodnocení celého projektu a případně vniklým komplikacím.

Zadavatelem EŽK je 1. Základní škola, Krátká 676, Klášterec nad Ohří. Celý projekt je koncipován jako aplikace na klíč. Zároveň však bude navržen tak, aby bylo možné ho v budoucnu zprovoznit i v dalších školách.

1 Dále může být používána zkratka EŽK

2 Analýza

2.1 Existující řešení

2.1.1 iŠkola.cz

Projekt iŠkola je komplexní produkt určený základním a středním školám. Celý program běží na serverech provozovatele, možnost přesunout aplikaci na vlastní server neexistuje. Program samotný se skládá ze zhruba dvacítky modulů, z nichž každý se stará o jinou funkci.

Kromě základních funkcí hodnocení a poznámek jsou dostupné i tvorby rozvrhů, suplování a docházka. Z nadstandardních modulů bych rád jmenoval možnost vytváření a absolvování on-line testů a písemek. Testy mohou být koncipovány s textovými odpověďmi, které pak ohodnotí učitel, či s možnostmi typu A,B,C,D, které jsou vyhodnocovány automaticky. Testy mohou být nastaveny jako cvičné, při kterých se student zdokonaluje, nebo jako zkušební, které lze absolvovat jen jednou a výsledek je okamžitě uložen a případně i opraven a ohodnocen.

Systém založí každému uživateli (studentovi, učiteli i rodičům) vlastní emailovou schránku, kam budou následně chodit nastavená upozornění a důležité události. Tato emailová upozornění je možné samozřejmě přesměrovat i do svého běžně používaného emailu. Další způsob komunikace je formou SMS, ve které zvolené informace okamžitě informují například o narychlo svolané poradě či třídních schůzkách.

Velmi rozmanitě umí tento IS generovat tiskové sestavy. Jejich formát je v mnohých případech totožný s materiály, které byly zvyklí používat učitelé před přechodem na IS EŽK. Jako příklad uveďme tisk třídní knihy, která je pak po svázání velmi podobná té, která se běžně na školách dodnes využívá.

Společnost se chválí velmi vysokým zabezpečením, a to jak proti hackerským útokům, tak proti krádeží hardwaru, výpadku napájení, ohni či živelné katastrofě. Data jsou údajně zálohována několikrát denně a to na záložní server umístěný několik desítek kilometrů od hlavního. Celý proces zálohování i následné uchování záložních dat má být chráněno šifrováním. Uživatel sám pak možnost vytvoření nějaké vlastní zálohy nemá.

Demo

Systém umožňuje za účelem seznámení se s produktem založit novou školu, třídy, rozvrh, apod. Omezeni jsme pouze na maximum 50 studentů a na dobu 30 dnů, po jejichž uplynutí přestane být do zaplacení příslušné částky nástroj dostupný. Zbytek modulů a funkcí pracuje zcela bez omezení.

Cena

Roční částka za placenou verzi se odvíjí od počtu žáků, které chcete v EŽK evidovat. Vezmeme-li v úvahu školu s devíti ročníky po třech třídách, vychází počet studentů na cca 600. Pro takto velkou školu je cena ročního užívání služby 6800 Kč.

2.1.2 ŠkolaOnline.cz

Nad tímto projektem převzalo záštitu Ministerstvo školství, mládeže a tělovýchovy.[2] Jde podobně jako v předchozím případě o hostovaný produkt určený základním, středním a vyšším odborným školám k vedení matriky a elektronické agendy spojené s provozem školy. Jednotlivé funkce jsou velmi podobné předchozí aplikaci.

Až po podrobnějším pročtení webu projektu však zjistíte, že aplikace „Katedra“, jak se EŽK jmenuje, poskytuje přístup a správu pouze školskému zařízení. Společnost v tomto ohledu zvolila zajímavou marketingovou strategii. Pokud totiž škola „Katedru“ využívá a rodič má zájem sledovat elektronickou cestou snažení svého potomka, musí si

2 Dle prezentace webu skolaonline.cz

sám u ŠkolaOnline.cz zakoupit druhý produkt s názvem „Žákovská“, který mu umožní přístup k požadovaným informacím.

Cena

Aplikace se „pronajímá“ stejně jako předchozí produkt iŠkola. Roční platba se počítá dle počtu žáků a výběru modulů, které bude škola využívat.

Budeme-li uvažovat stejně velkou školu jako v předchozím případě, tedy 600 žáků, vyjdou nám po sečtení položek následující částky: Cena základních modulů aplikace „Katedra“ je cca 12000 Kč/rok. Cena všech modulů je pak 17000 Kč/rok.

Jak již bylo řečeno, další náklady vznikají rodičům, kteří chtějí mít k prospěchu a souvisejícím informacím přístup. Cena za měsíc přístupu je stanovena na 45 Kč.

2.1.3 WebNotes.cz

WebNotes je projekt velmi podobným těm výše zmíněným. Veškerá data jsou opět ukládána na serveru provozovatele. Na rozdíl od předchozích jsou známky ukládány vždy jen do konce školního roku. Pak je kompletní databáze vypálena na CD médium a předána škole. O formátu těchto dat není na webu projektu bohužel žádná zmínka. Známky jsou po vypálení na CD ze serveru smazány (seznamy učitelů a žáků samozřejmě zůstanou) a další školní rok začíná zcela „nově“ s prázdnou databází. Prohlížet data zpětně několik let tedy možné není.

Aplikace jako taková je dle mého názoru graficky až příliš strohá a na první pohled se nejeví jako úplně přehledná. Autoři sami v průvodních textech jasně píší, že grafika je úmyslně co nejjednodušší, aby se stránky rychle načítaly i na pomalejších strojích a pomalejších připojeních k internetu. Nicméně i bez použití obrázků by pouhým vhodnějším zvolením fontů a barev bylo možné dosáhnout mnohem lepšího dojmu.

Funkce této aplikace jsou mnohem chudší, než funkce dříve popisovaných projektů. To však zcela jistě nemusí být překážkou. WebNotes poskytuje ty nejpotřebnější funkce, jako jsou známkování, evidence nepřítomnosti a komunikace s rodiči a učiteli.

Cena

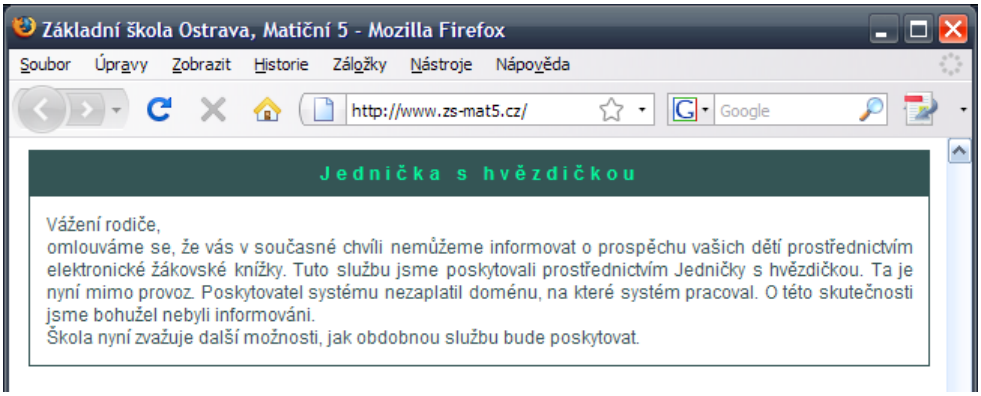
Má-li škola o tento projekt zájem, může si ho dva měsíce zcela zdarma bez omezení testovat. Po této době budou v případě neobjednání placené verze veškerá data smazána.

Cena placené verze je 800 Kč/měsíc, a to bez ohledu na velikost školy a počty žáků nebo učitelů.

2.1.4 Uzavřené projekty

Kromě výše zmíněných funkčních[3] EŽK jsem na internetu narazil i na odkazy a zmínky o dalších podobných informačních systémech, které ale již zjevně nejevily známky života. Buď byly stránky nedostupné, nebo byla již dokonce zrušena celá internetová doména, na které projekt původně fungoval.

Jako názorný příklad uvedu webovou službu a aplikaci s názvem „Jednička s hvězdičkou“, která zcela nečekaně a bez ohlášení ukončila svůj provoz.



Základní škola Ostrava, Matiční 5 - stav k 19.1.2009

3 Stav k únoru 2009

2.1.5 Zhodnocení současného stavu a rizik existujících řešení

V současné době se společnosti ve spojení s EŽK zaměřují na internetu hlavně na poskytování služeb na vlastních serverech, nikoliv na prodej samotných aplikací. Pro školy má tento přístup mnoho výhod i nevýhod.

Hlavně výše ukázaná nemožnost kontrolovat a ovlivňovat dostupnost služeb může být v jistých situacích kritická. Příklad služby „Jedničky s hvězdičkou“ z předchozí kapitoly jasně ukázal, že riziko nečekané a definitivní nedostupnosti aplikace a hlavně jejích dat je více než reálné. Škola se tím pak vystavuje riziku, že se dostane do prakticky neřešitelné situace – doména neexistuje a data jsou možná navždy ztracena.

2.1.6 SWOT analýza existujících řešení

SWOT analýza *je metoda, pomoci které je možno identifikovat silné (ang:* ***S****trengths) a slabé (ang:* ***W****eaknesses) stránky, příležitosti (ang:* ***O****pportunities) a hrozby (ang:* ***T****hreats), spojené s určitým projektem, typem podnikání, opatřením, politikou apod. Jedná se o metodu analýzy užívanou především v marketingu, ale také např. při analýze a tvorbě politik (policy analysis).* [zdroj 1]

Silné stránky	Slabé stránky
• Aplikace je již hotova, lze ji okamžitě nasadit a spustit • Aplikace je pravděpodobně dobře odladěna • Není třeba žádné správy software • Není třeba vlastní server • Nízké počáteční náklady • Časově rozložené financování projektu • Rozmanitost funkcí • Zajištěna bezpečnost zálohováním	• Do jádra aplikace „není vidět“, jde o uzavřený systém • Nelze zkontrolovat zabezpečení systému • Nelze zjistit, kdo má k databázím přístup • Nutnost stálého investování finančních prostředků (software je pronajímán) • Je třeba kompromisů – jen málokdy řešení přesně odpovídá požadavkům • Nutnost proškolení učitelů

Příležitosti	Ohrožení
• Dlouhodobá spolupráce se silným IT partnerem	• Systém lze po delší době používání jen obtížně vyměnit za jiný • Systém může být pro potřeby školy příliš složitým • Poskytovatel může nečekaně ukončit provoz nebo změnit cenovou politiku • Data mohou být zneužita • Data mohou být poškozena, ztracena • Napadení systému hackery • Výpadek systému, který nelze ovlivnit

Vzhledem ke komplikovanosti celého problému je nutné některé body SWOT analýzy dále rozvést, a proniknout tak do nich hlouběji.

Rizika spojená se shromažďováním informací

Jak již bylo řečeno, všechny existující projekty běží na serverech společností, které tyto aplikace nabízejí. To s sebou přináší výhody, nevýhody i omezení.

Jednou z nesporných výhod je, že za správu dat ručí provozovatel. Ten také provádí zálohy a ostatní kroky, kterými zajišťuje bezpečnost dat v případě nečekaných událostí. Jako další výhodu bych označil nemožnost přistoupit přímo k vlastním databázím. Takto nízkoúrovňové zasahování není vhodné, protože špatným zásahem může jednoduše dojít k narušení konzistence dat. Přímý přístup do databáze bývá ve většině případů nutný jen pro potřeby záloh (kterou za nás v tomto případě vytváří provozovatel).

Velice rizikovou část však vidím v nutnosti plně a slepě se spoléhat na provozovatele. Praktická zkušenost totiž často ukazuje, že spousta věcí, kterými se společnosti chtějí zviditelnit, jsou jen reklamní slogany – realita je však zcela jiná.

V případě, že se tedy škola rozhodne svěřit někomu svá data, měla by si důkladně prověřit, zda ten, s kým jedná je opravdu tak spolehlivý, jak se prezentuje.

Ať už dojde k napadení a odcizení dat hackery, úmyslnému „prodeji“ informací, či zanedbání bezpečnosti, výsledek bude vždy podobný. Je důležité si uvědomit, že škola jakožto zákazník nemá sebemenší možnost tyto případné problémy předem ovlivnit a pokusit se jim vyvarovat. Protože data vždy leží kdesi na vzdáleném serveru, nemůže škola v žádném případě transparentně zjistit, jakým způsobem jsou data zabezpečena a kdo k nim přistupuje!

Rozmanitost funkcí

Ta může být v jistých případech výhodou i nevýhodou. Na jedné straně umožňuje škole využívat přidaných funkcí, na které by u aplikace na zakázku nikdy finančně nedosáhla. Z druhého úhlu pohledu však tyto funkce mohou aplikaci natolik zkomplikovat, že bude pro danou školu v důsledku naprosto nepoužitelná.

Vezmeme-li jako příklad školu, která chce pouze rodičům poskytovat informace o prospěchu jejich dětí. V takovémto základu celkem jednoduché ovládání: „Učitel se přihlásí, podle jména najde studenta, napíše textem název předmětu a hodnotu známky.“ Pokud ale do aplikace přidáme kupříkladu funkci rozvrhu, vzniká spousta komplikací. Tou první je samotné zadání studenta. U něj již není třeba jen zadat jméno, příjmení, případně nějakou další identifikaci. Teď už je nutné pro něj vytvořit každý rok třídu a do té zmíněného studenta přiřadit. Následně se podobně vytvoří jednotlivé předměty a spojí se s učitelem, který je vyučuje. Je pravděpodobné, že v takovémto rozšířeném systému již nebude možné, aby známky jednoduše zadal kterýkoliv z učitelů. Toto oprávnění bude mít nejspíše jen učitel, který v dané třídě daného žáka daný předmět vyučuje. Na to se pak ještě mohou nabalit komplikace typu „suplování“, či jiné podobné ne zcela běžné události.

Tento příklad jasně ukazuje, že v zadání projektu na klíč i jeho následné analýze je třeba důkladně zvážit, jak složitý výsledný informační systém má být a jaké má mít funkce, aby byl dobře použitelný a byl využíván z přesně 100 procent (tj. nebyl ani zbytečně rozsáhlý, ani příliš okleštěný).

Výběr bývá definitivní

Je dobré si uvědomit, že pokud škola přistoupí k tomu, že začne využívat a platit jedné ze společností za služby EŽK, lze toto rozhodnutí po delším čase jen těžko vzít zpět a přejít například na systém konkurence. Hlavní důvody jsou tři:

1. V systému je již zadáno příliš mnoho údajů, které většinou nelze hromadně vyexportovat. Ruční přepsání je již nemyslitelné a nerealizovatelné.
2. Jakýkoliv jiný systém bude z větší či menší části nekompatibilní. I v případě, že bychom byli schopni vyřešit export ze starého systému a konverzi i import do systému nového, bude třeba některé informace doplnit. A nebo může nastat situace opačná – a to taková, že o některá data přijdeme, protože je nový systém nebude podporovat.
3. Učitelé, studenti i rodiče si na stávající systém dlouho zvykali, než ho dokázali plně využít a ovládat. Změna by pro ně byla více než náročná.

2.2 SWOT analýza vlastního IS EŽK z pohledu zadavatele

Silné stránky	Slabé stránky
• Licence nemá omezení počtu registrovaných uživatelů v systému • Aplikace je vytvořena přesně dle potřeb a požadavků školy • Žádné budoucí nutné náklady na provoz aplikace • Lepší komunikace při řešení problémů • Možnost navázání na stávající IS školy	• Vyšší počáteční náklady • Lze očekávat počáteční problémy a potřebu aplikaci odladit v provozu • Je třeba vlastní server, na kterém bude aplikace umístěna • Nutnost proškolení učitelů • Aplikace se bude nějakou dobu vyvíjet, než ji bude možné nasadit

Příležitosti	Ohrožení
• Další rozvíjení aplikace podle potřeb školy • Možnost uplatnění na dalších školách a z toho vyplývající rozšíření využití aplikace a tedy i její intenzivnější vývoj.	• Skryté závažné chyby, které se projeví až po delším čase provozu • Zabezpečení vlastních serverů • Ukončení spolupráce ze strany vývojářů systému

Jako nejvážnější položku celé SWOT analýzy lze označit ohrožení v podobě možných skrytých závažných chyb, které se v systému časem mohou projevit. Tento problém se u mnohých aplikací eliminuje jejich masovým nasazením. Pokud totiž aplikace již delší dobu běží někde jinde (na mnoha místech zároveň), je velmi nepravděpodobné, že by se objevil ještě nějaký nový závažný problém. Naopak v aplikaci na klíč existuje i přes intenzivní testování reálná šance, že nějaký problém v budoucnu nastane.

Jedním z ohrožení je také možnost ukončení spolupráce s vývojáři. Jde o riziko porovnatelné s možností ukončení projektu, který je provozován na cizích serverech a poskytován pouze jako služba. Pokud by však tato situace nastala zde, neměla by takové kritické důsledky jako v případě hostované aplikace. Škola může provozovat vše dále a případně si najít jiného vývojáře, který bude na projektu pokračovat (pokud to dovolí licenční podmínky). Aplikace může ale plně fungovat ve svém současném stavu dále i bez původní podpory.

3 Návrh aplikace

3.1 Specifikování požadavků

Prvním krokem, který musí předcházet samotný návrh aplikace je podrobné shrnutí veškerých požadavků se zadavatelem. Pak následuje zhodnocení a konzultace získaných informací s programátorem. Třetí a poslední fází je „vyjednávání“ zadavatele s programátory o možném kompromisu. Jen málokdy lze totiž všechny požadavky na 100% splnit.

Informační systém EŽK bude od počátku tvořen hlavně jako webová aplikace použitelná v běžném internetovém prohlížeči bez potřeby cokoliv na klientských počítačích instalovat.

3.1.1 Požadavky zadavatele

Zadavatel (konkrétní škola) má zájem EŽK nasadit do zkušebního provozu, přičemž rozsah uživatelů bude pravděpodobně jedna vybraná třída žáků, jejich rodiče a učitelé.

Systém je však nutné navrhnout tak, aby byl co nejsnáze ovladatelný. Primárně je třeba klást důraz na rychlost zadávání nejčastěji vkládaných dat – tedy známek.

Základní požadované funkce

- zabezpečená komunikace
- evidence známek
- správa uživatelů a jejich profilů a oprávnění
- profil uživatele „rodič“
  - možnost prohlížení známek potomků
- profil uživatele „student“
  - možnost prohlížení svých známek, případně dalších informací

Požadavky na server

Škola provozuje několik vlastních serverů, na kterých je možné EŽK zprovoznit. Servery jsou povětšinou linuxové. V případě potřeby je k dispozici i server s operačním systémem Windows a službami IIS.

Tvořená aplikace EŽK bude využívat webový server Apache s programovacím jazykem PHP. Jako úložiště dat bude použitao výhradně SQL databáze, konkrétně půjde o MySQL verze 5.

Důvodem použití právě jazyku PHP a databáze MySQL je jejich současná dostupnost na většině veřejných i privátních webových serverů.. Díky využití právě těchto technologií má projekt informačního systému EŽK mnohem větší šancí na rozšíření i do dalších škol.

Možnost rozšíření o Win32 aplikaci

Učitelé nemají ve většině tříd během hodiny možnost se připojit on-line k EŽK a zadávat nové známky. Důvodem je většinou nepřítomnost počítače připojeného do intranetové sítě. Mnozí vyučující si však s sebou do tříd nosívají zapůjčené notebooky.

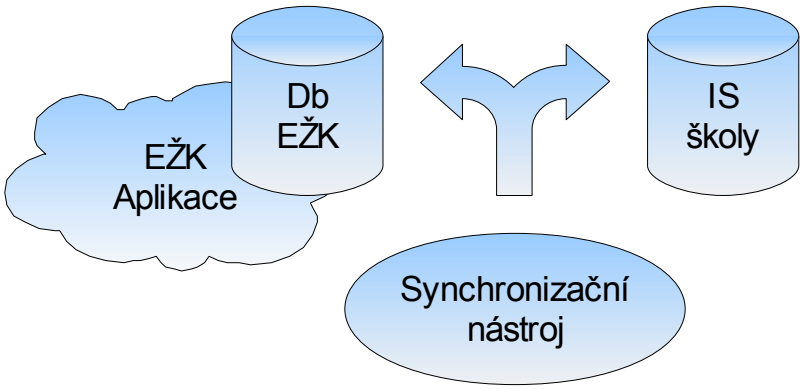
Tohoto stavu by chtěla škola využít a v budoucnu rozšířit sytém EŽK o off-line klienta, do kterého by během dne byly známky zadávány. Po vyučování by se pak klient po připojení k síti spojil se serverem a provedl synchronizaci všech nově vložených nebo upravených známek.

Součástí diplomové práce však není návrh této aplikace. Je ale nutné již při návrhu zajistit, aby bylo v budoucnu toto rozšíření možno realizovat. Vhodné je též rovnou navrhnout způsob, jakým bude off-line klient se serverem komunikovat.

3.1.2 Požadavek na návaznost na jiné IS

Tento informační systém je primárně realizován jako aplikace na klíč pro konkrétní organizaci. Aby však bylo možné v budoucnu využít aplikaci i v dalších školách, nebude nikde v systému realizována žádná návaznost na stávající informační systém (respektive databáze) školy obsahující například seznamy žáků nebo učitelů. Veškerá tato data bude systém EŽK ukládat a spravovat ve vlastním nesdíleném úložišti.

V současnosti nevznikl od zadavatele žádný požadavek na spojení EŽK s jiným IS. Bude-li třeba nespravovat v budoucnu uživatele EŽK a jejich oprávnění ručně, ale navázat aplikaci na jiný systém, který tyto informace bude obsahovat, bude tento úkol řešen individuálně vytvořením externího nástroje (mimo EŽK), který bude synchronizovat veškerá data mezi databází EŽK a úložištěm dalšího IS, který škola pro účely evidence využívá.



3.2 Návrh zabezpečení

Zabezpečení aplikace jako takové lze rozdělit do několika samostatných částí.

3.2.1 Autentizace uživatelů

Každý uživatel (student, rodič i učitel) dostane přidělené uživatelské jméno a výchozí heslo, kterým se bude autentifikovat do celého systému. Vyšší zabezpečení, například osobními certifikáty, nebude využito. Důvodem je složitost autentizace – a to hlavně na straně klienta, jehož mobilita, tedy možnost přihlášení odkudkoliv by byla značně omezena. Uživatel by totiž musel mít osobní certifikát vždy u sebe, což v praxi není dost dobře realizovatelné.

Je dobré zamyslet se, kde bude aplikace využívána. Prostředí základní školy, konkrétně kabinetů učitelů se vyznačuje tím, že se v něm pohybuje občas velké množství lidí (často i studentů). U jednoho počítače se pak místy střídá denně i několik vyučujících. Této situaci je nutné přizpůsobit úroveň zabezpečení a hlavně způsob odhlašování uživatelů ze systému.

Uživatelská jména

Zvlášť je třeba se zamyslet nad tím, jaká uživatelům rozdávat jména. Je totiž nutné zajistit jejich unikátnost a zároveň pokud možno snadnou zapamatovatelnost. Z tohoto důvodu zavrhuji možnost přidělovat jako přihlašovací jména nějaké číselné identifikátory. I přesto je vhodné říci, že číselné ID mají tu výhodu, že není třeba řešit obtíže, které mohou v ostatních případech s uživatelskými jmény jiného druhu nastat.

Jako snadno zapamatovatelná bych označil jména ve formátu „*jméno.příjmení*“. Nicméně tato volba s sebou přináší úskalí se zajištěním jednoznačnosti. Problémy se projeví často až při větším počtu uživatelů a to ve formě různých prefixů či postfixů, které je nutno zavést v případě duplicity jména i příjmení.

Další možností formy uživatelského jména je emailová adresa. Ta má tu výhodu, že je zcela unikátní. Vzhledem k tomu, že freemailových poskytovatelů je

v České republice i celosvětově[4] několik desítek, stává se jen opravdu zřídka, že jednu emailovou adresu využívá více lidí současně (například v rodině). Také pravděpodobnost, že cílová skupina, která bude využívat služeb a možností EŽK, nemá žádnou emailovou schránku, je zcela zanedbatelná. Kdyby však tato situace opravdu nastala, může jí uživatel vyřešit jednoduchým založením jakékoliv freemailové schránky.[5]

Jako zcela nejlepší volbu považuji nechat administrátorovi možnost uživatelské jméno nového uživatele zvolit sám. Pakliže správce toto uživatelské jméno nezvolí, měla by aplikace nějaké vhodné vygenerovat sama. O ručním přidělování uživatelských jmen lze spekulovat pravděpodobně jen v případě uživatelů z řad zaměstnanců školy. Pokud půjde o žáky, jejich množství ruční výběr jména prakticky zcela vylučuje.

Žákům bude systém vytvářet přihlašovací jméno automaticky, a to nejlépe ve formátu „jmeno.prijmeni“. Případná vznikající duplicita bude řešena přidáním číselného postfixu, tedy například „jmeno.prijmeni2“. Formát generovaných jmen bude v případě potřeby možné kdykoliv změnit bez ovlivnění již dříve vytvořených účtů.

Hesla

V souvislosti s přihlašovacími hesly je nutné zaměřit se na dva hlavní body:

Výchozí heslo

Z hlediska bezpečnosti lze jako zcela nevhodné označit přidělování stejného výchozího hesla všem uživatelům - navíc, jak je na mnohých menších serverech zvykem, bývají to hesla typu „12345“. Je tedy jednoznačně nutné umět generovat náhodná hesla tak, aby každý nový uživatel dostal heslo zcela jedinečné.

Jak je známo, jistá skupina nezkušených uživatelů si často heslo vůbec nezmění – prostě ho nechají při prvním přihlášení uložit do prohlížeče a již ho nezadávají. Je tedy nutné, aby vygenerované heslo bylo dostatečně silné a odolné vůči útokům. Zároveň je ale vhodné neznechutit uživatele hned na počátku tím, že mu bude na papíru předáno heslo 20 znaků dlouhé a plné symbolů, které začátečník jen stěží na klávesnici vůbec najde.

4 celosvětově využívané služby, jako je například Gmail, Hotmail, apod.
5 Vzhledem k tomu, že email je žádán u prakticky všech služeb a registrací na internetu, tomuto problému by uživatel čelil tak jako tak i na jiných serverech.

Jako zajímavá alternativa k zcela nesmyslnému textu se jeví možnost vygenerovat heslo pomocí velkého slovníku. Aby bylo znemožněno jakémukoliv slovníkovému útoku, je samozřejmě nutné do hesla dát několik slov za sebou (například i v kombinaci s čísly). Výsledné heslo může vypadat tedy například takto: „martin35Letadlo981“.

Minimální nutná složitost při změně hesla

Protože nejslabší částí aplikace většinou bývá uživatel, je nutné alespoň částečně zamezit uživateli používat zcela triviální hesla. Omezení minimální délky hesla se ukazuje jako postačující řešení. Pokud totiž přidáme i mnohá složitější pravidla (jako například pamatování si historie použitých hesel, povinnost pravidelné změny hesla, apod), přílišného zvýšení bezpečnosti se nedočkáme. Uživatelé si totiž často vytvářejí systémy, podle kterých hesla mění za velice podobná – čímž obcházejí jinak nutnou námahu pamatovat si stále nová hesla.

3.2.2 Přenos dat po síti

Vzhledem k citlivosti dat je vhodné, aby veškerá data mezi serverem a jednotlivými klienty byla chráněna proti napadení. V současné době je systém navržen pouze pro komunikaci přes web. Jako nejednodušší a nejúčinnější se jeví použití šifrovaného protokolu HTTPS.

Pokud si škola zařídí i vlastní certifikát vydaný důvěryhodnou a pokud možno celosvětově uznávanou certifikační autoritou, bude touto cestou zároveň vždy zaručena i identita serveru. Jednotliví uživatelé si tedy budou jisti, že komunikují opravdu se správným a nepodvrženým systémem EŽK.

Jako vhodný vlastní, celosvětově uznávaný certifikát lze doporučit certifikát od CA GeoTrust, který lze v České republice získat od společnosti Ignum. Jde o SSL certifikát o délce 256 bitů. V nabídce lze zvolit, jestli bude certifikát určený po konkrétní doménu včetně názvu poddomény, nebo půjde o certifikát použitelný pro všechny domény třetí úrovně (například *.1zsKlasterec.cz).

Vystavení certifikátu	Cena
certifikát pro 1 doménu na 1 rok od důvěryhodné CA CA GeoTrust, doména ve tvaru např. www.domena.cz	1490 Kč / rok
certifikát wildcard SSL na 1 rok od důvěryhodné CA CA GeoTrust, domény ve tvaru *.domena.cz	5190 Kč / rok

3.2.3 Databáze

Veškerá data budou uložena v SQL databázi. Její kvalitní zabezpečení je tedy velmi důležité. K serveru s daty bude vždy přistupovat pouze webová aplikace, nikdo jiný. Přímý přístup do databáze jinými nástroji bude v běžném provozu zcela zakázán.

Z hlediska bezpečnosti je nejvhodnějším řešením oddělit databázový server od vlastního webserveru. Pokud by se útočníkovi podařilo proniknout přes některá zabezpečení webserveru, zůstane databáze stále zabezpečena.

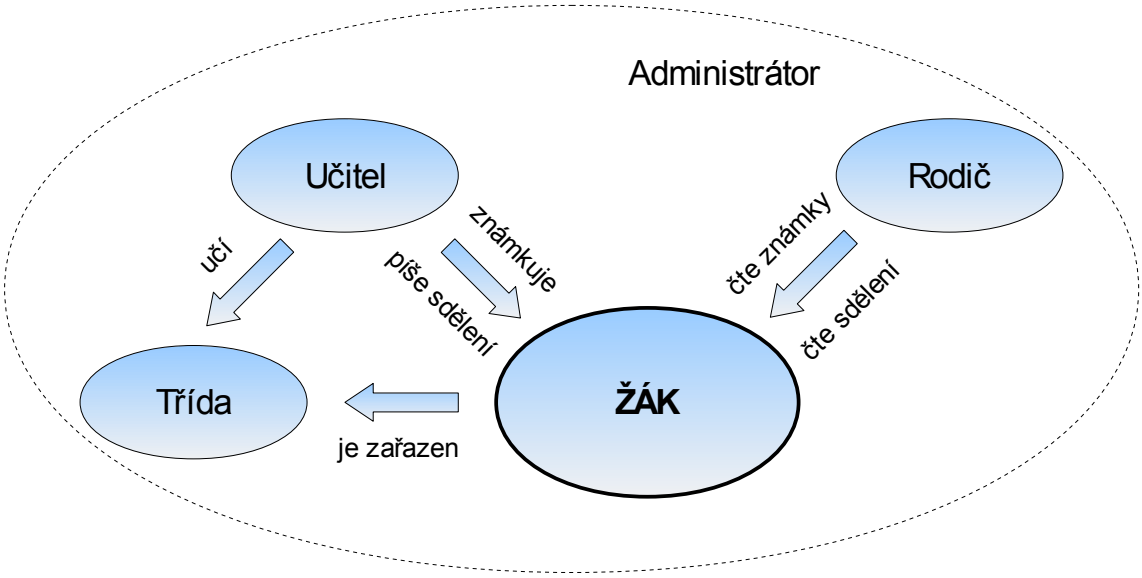
Při průniku do systému může nastat problém, že útočník získá přístup ke zdrojovým kódům aplikace, z nichž lze získat přístupové údaje do databáze. Je tedy vhodné nastavit omezení, aby databázový server přijímal požadavky pouze z webserveru, od nikoho jiného. Pak by útočník musel všechny své útoky vést právě přes server s webovou aplikací (a to se mu již podařit nemusí).

Nevýhoda řešení dvou serverů spočívá v potřebě přenášet veškeré SQL dotazy a odpovědi přes síť, což může snížit rychlost odezvy aplikace a zvýšit zátěž aktivních prvků v intranetu. Tento nedostatek lze odstranit virtualizací serverů, kdy oba servery běží na zcela oddělených systémech, z hlediska hardwaru půjde však o tentýž stroj. Toto řešení je však mnohem náročnější a je třeba zvážit, jestli jsou vzniklé výhody dostatečné v poměru s vynaloženou prací.

3.3 Entity EŽK

Informační systém EŽK předpokládá několik druhů entit ve smyslu objektů, které v informačním systému existují.

Jednoznačně nejdůležitější entitou, kolem které se vše děje, je žák. Tuto entitu tedy označíme jako střed celého informačního systému.



Poznámka: Učitelé píší sdělení rodičům. Nicméně toto sdělení se týká konkrétního žáka, ne konkrétního rodiče. Sdělení je tedy uloženo u žáka a přečíst si ho může kterýkoliv z definovaných rodičů.

Objekt „třída“ má v systému dvě hlavní funkce. První funkcí je vnesení jisté hierarchie a přehlednosti do jinak nerozlišitelného seznamu všech žáků školy. Díky tomu, že každý žák je zařazen (ale nemusí) do některé z definovaných tříd, kterou navštěvuje, má učitel možnost tohoto žáka (nebo skupinu žáků) vyhledat mnohem rychleji, než kdyby se orientoval pouze pomocí jmen a příjmení žáků. Díky řazení do tříd lze ale také definovat učitelům podrobnější oprávnění popisující, které třídy učitelé učí – a komu tedy mají v důsledku oprávnění zapisovat známky.

Rozšíření o detailnější oprávnění

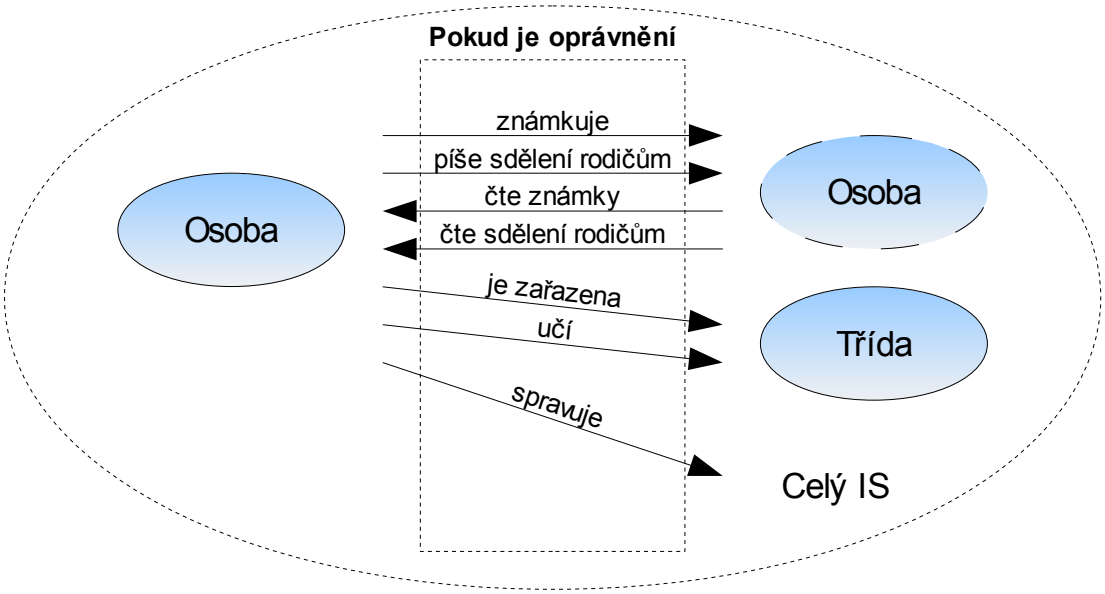
Jeden z návrhů počítal s oprávněním až na úroveň jednotlivých předmětů vyučovaných ve třídách. Podle zadavatele byla tato část však nadbytečná a nepřinesla by užitek, proto nebyla po společné konzultaci implementována. Systém bude však ukládat informace o zadavatelích (učitelích) jednotlivých položek (například známek), což společně se zpětnou vazbou kontroly ze strany samotných studentů a rodičů poskytne dostatečnou spolehlivost a úroveň kontroly zadávaných údajů.

Pokud by v budoucnu bylo přeci jen třeba nastavovat oprávnění až na úroveň jednotlivých předmětů, nebude dle současného návrhu nutné příliš zasahovat do implementace aplikace. Z pohledu databáze půjde pouze o přidání jedné tabulky popisující oprávnění učitelů k zápisu jednotlivých předmětů ve třídách. V samotném programu pak bude kromě krátké části ověřování tohoto oprávnění nutné naprogramovat především administrační část, ve které se budou oprávnění přidělovat a odebírat.

Z pohledu implementace

Celý návrh, tak jak byl naznačen výše, je velmi abstraktní. Je nutné si totiž uvědomit, že skupiny učitelů, rodičů a administrátorů se mohou vzájemně prolínat. Prolínat by se mohly samozřejmě i se skupinou žáků, ale tato situace není na základní škole příliš reálná. Vezmeme-li tedy v úvahu myšlenku, že rodič může být zároveň učitelem, některý učitel může být i administrátorem IS, vniká nám ve stávajícím návrhu jedna možná nepříjemná vlastnost. A to ta, že pokud zůstanou entity striktně oddělené, musel by mít člověk, který je rodičem, učitelem i administrátorem tři různá přístupová jména.

Upravme tedy původní myšlenku tak, aby byla jednak příjemnější pro jednotlivé uživatele, ale také, aby byla blíže výsledné implementaci.



Tento návrh již dává přesnější představu o tom, jak bude výsledná aplikace vypadat. Původní 4 entity (žák, učitel, rodič, administrátor) se nám redukují na jednu zobecněnou entitu Osoba. Veškerá rozlišení původních entit jsou v tomto případě přenesena na oprávnění, což je soubor vlastností, které budou u každé osoby zadány.

3.4 Oprávnění uživatelů

Systém EŽK bude primárně rozlišovat čtyři hlavní oprávnění, které odpovídají rozlišení entity osoby do čtyř různých skupin, tak jak to ukazoval první abstraktnější návrh entit. Každé oprávnění může být buď povoleno (hodnota 1) nebo zakázáno (hodnota 0). Pokud bude třeba v budoucnu nastavit oprávnění jemněji, bude to možné přidáním dalších hodnot (2,3,4, …), které budou mít specifický význam. Díky možnosti nastavení každého oprávnění zvlášť bude možné oprávnění libovolně kombinovat. Jde o dříve zmíněné případy, kdy například učitel je zároveň rodičem nějakého žáka vedeného v EŽK.

Samotný běžný uživatel má oprávnění změnit svou emailovou adresu a heslo. Administrátoři mají navíc oprávnění změnit i jméno a příjmení kterékoliv osobě. Pokud by tuto možnost měli i běžní uživatelé, mohla by vzniknout komplikace při jejich následném dohledávání v systému.

Změnu jména lze předpokládat jen v případě vdávání, což nebude příliš častý jev. Pokud bude tedy učitel požadovat změnu jména, bude muset požádat administrátora. Rodič pak může stejný požadavek vyřešit zprostředkovaně přes třídního učitele svého dítěte.

Oprávnění ŽÁK

Uživatel s oprávněním „žák“ má možnost prohlížet informace zadané o něm (známky, sdělení). Dále může zobrazit profilové informace všech učitelů (na nich může najít například jejich emailovou adresu).

Oprávnění UČITEL

Může procházet všechny třídy, vyhledávat a zobrazovat v nich žáky. Žákům vybraných tříd (které učí) může zadávat známky. Známky jím zadané má oprávnění smazat.

Sdělení rodičům může uživatel psát všem žákům (bez ohledu na to, jestli danou třídu, kde žák je, učí). Není-li sdělení zatím rodičem podepsáno, může ho upravit nebo smazat. Pokud podepsáno již je, upravit jeho znění již možné není, je možné ho pouze smazat (tato vlastnost bude ještě konzultována při dalším testování).

Oprávnění RODIČ

Může prohlížet známky a sdělení pouze svých dětí. Kterýkoliv z definovaných rodičů může nové známky a sdělení podepsat a stvrdit tak, že je četl. Dále pak může prohlížet profily s informacemi o učitelích.

Oprávnění ADMINISTRÁTOR

Má přístup k celému systému. Spravuje seznam uživatelů (vytváření, změny, mazání), jejich profily (jména, emaily, atd.) a oprávnění. Jakémukoliv uživateli může změnit přístupové heslo, nemůže však zjistit současné. Administrátor má dále na starosti vytváření nových tříd, přiřazování žáků do tříd, třídní učitele, nastavení oprávnění učitelům k zápisu známek. Definuje také rodičovské vazby mezi uživateli (vazba rodiče-žáci). Každý rok se pak stará o vykonání přechodu do nového roku (převod tříd,

smazání známek, atd.). Kromě toho může komukoliv ze systému zadat či smazat známku, či sdělení.

3.5 Školní rok a pololetí

Původní návrh umožňoval ukládat data o známkách a následně se k nim vracet v rámci několika let. Tato vlastnost však zvyšovala náročnost celého systému a nepřinášela škole žádný užitek. Na návrh zadavatele byla tato vlastnost ze systému odstraněna a dále se s ní nikterak nepočítá.

Současný konečný návrh tedy nezná pojem „školní rok“. Zná však pojem pololetí. Veškeré přehledy známek jsou vztahovány k aktuálnímu půlroku s možností přepnutí mezi prvním a druhým pololetím.

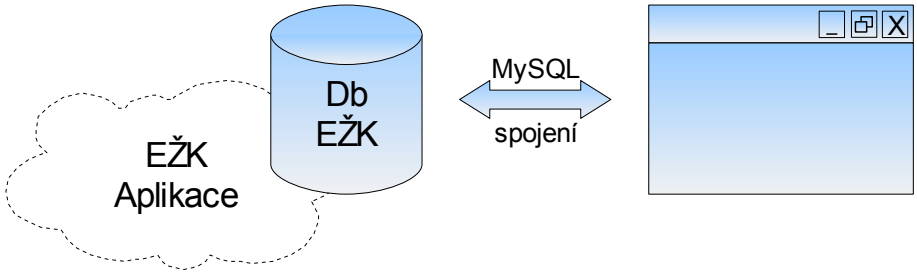
O letních prázdninách provede administrátor úkon spojený s přechodem do dalšího školního roku. Ten spočívá v následujících krocích:

1. Zazálohování dat.
2. Vyřazení žáků devátých ročníků ze tříd.
3. Odstranění vyřazených žáků ze systému.
4. Odstranění rodičů vyřazených žáků (pokud v systému již není žádný jiný jejich potomek).
5. Převod všech ostatních žáků o ročník výše.
6. Smazání všech zapsaných známek.
7. Smazání všech sdělení rodičům.
8. Odstranění oprávnění k zápisu známek všem učitelům.
9. Vytvoření uživatelů vztahujících se k prvnímu ročníku.
10. Nové nastavení oprávnění učitelům k zadávání známek.
11. Ruční přeřazení žáků, kteří budou opakovat ročník.

Návrh počítá s tím, že v rámci zjednodušení bude systém schopen asistovat – s možností úprav a nutným potvrzením provede úkony 2 až 8 automaticky. Úkon 1, zálohování databáze může být v budoucnu na žádost zadavatele součástí systému, nebo lze použít jiných již existujících prostředků k zálohování databází.

3.6 Spolupráce s externím off-line klientem

Původní návrh programátora, který bude externí aplikaci v budoucnu psát, bylo napojit klienta přímo na databázi a veškeré akce provádět přímo, nezávisle na aplikaci EŽK.



Navrhovaný systém měl obsahovat jakýsi token, který uděloval možnost do systému zapisovat. Standardně by byl token na serveru. V případě připojení off-line klienta by byl token předán klientovi, přičemž v tu chvíli by se stal jediným, kdo by měl oprávnění do dat zasahovat. Až by pak provedl potřebnou synchronizaci (většinou jen nahrání nových známek a sdělení rodičům), vrátil by token zpět na server.

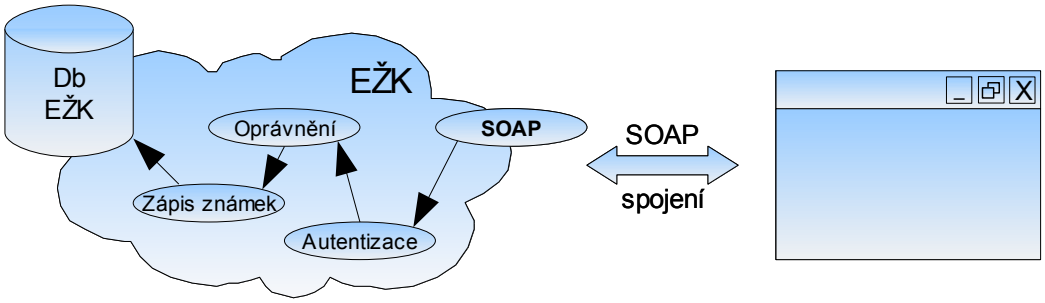
Tento přístup s sebou nese jistá rizika. Tím hlavním je nutnost aplikace znát přístupové heslo do databázového serveru. Pokud by někdo neoprávněný z aplikace toto heslo získal, mohl by číst i zapisovat naprosto cokoliv. Další problém se objeví ve chvíli, kdy bude aplikace EŽK poupravena, případně následně lehce změněna struktura databází. V té chvíli by byla externí aplikace buď nefunkční, nebo by mohla dokonce narušit integritu již uložených dat. Myšlenka tokenu pak nese riziko jeho ztráty. Pokud by při synchronizaci dat došlo k přerušení spojení, zůstal by do obnovení token pravděpodobně u klienta, čímž by ostatním bylo na delší dobu znemožněno do systému zapisovat.

Je tedy třeba navrhnout jinou metodu přístupu off-line klientem, která by splňovala následující podmínky:

- Šifrovaný přenos
- Aplikace nemusí znát nějaké globální přístupové heslo
- Ověření oprávnění provádí EŽK (nikoliv off-line klient)
- Oba systémy (EŽK i off-line klient) musí být použitelné zároveň

XML / SOAP

Jako vhodné řešení se jeví přenos dat pomocí SOAP či XML protokolu. Díky implementaci v PHP by XML komunikace umožnila přenos dat mezi off-line klientem a samotnou EŽK (namísto komunikace s databází). Šlo by tedy o nepřímý přístup, kdy by samotný fyzický zápis neřešila off-line aplikace, ale webová EŽK. Ta by samozřejmě zápis povolila stejně jako při přihlášení uživatele přes webové prostředí, tedy až po ověření hesla a všech potřebných oprávnění.



Výhoda tohoto řešení je i v tom, že velké množství potřebných komponent (např. autentizace, ověření oprávnění, či fyzický zápis známek) by bylo již implementováno. Chybělo by tedy realizovat jen samotné přijímání a odpovídání XML protokolem nebo konfigurace služeb SOAP protokolu.

3.7 Zálohování dat

Současný návrh informačního systému EŽK nepředpokládá implementaci žádného vlastního nástroje pro zálohování dat. Tvorbu všech záloh bude mít na starost externí zálohovací aplikace, která bude ve stanovených intervalech brát celou databázi a bude ji kopírovat po síti na druhý server. Takto vytvářené zálohy pak bude administrátor příležitostně ukládat na externí média (například na CD/DVD).

Zálohování databáze může probíhat fyzickým zkopírováním souborů, do kterých si databázový server ukládá veškeré obsahy tabulek a indexů. Druhou cestou je použití specifického zálohovacího nástroje, který dokáže do databáze přistoupit a data vyexportovat do jiného formátu.

4 Implementace

K implementaci byly v plné míře, kromě operačního systému, využity open-source a freeware aplikace a vývojové nástroje.

Hlavním programovacím jazykem aplikace je PHP 5. Právě díky verzi 5, která má již plně rozvinutou podporu práce s objekty, bylo možné některé části aplikace zapouzdřit do vlastních tříd.

Na straně klienta aplikace je pak pro větší interaktivitu formulářů využíváno JavaScriptu, který je odladěn pro bezchybný běh v IE8, Mozilla Firefox, Opeře i Google Chrome. Jako výstupní značkovací jazyk je použit XHTML 1.0 Transitional dále doplněný o CSS 2. Při samotném programování je kladen důraz na validitu celého kódu, který je klientovi odesílán. Stálou kontrolu validity při vývoji pak v prohlížeči Mozilla Firefox 3 zajišťuje zásuvný modul „Tidy HTML validátor“.

4.1 Databáze

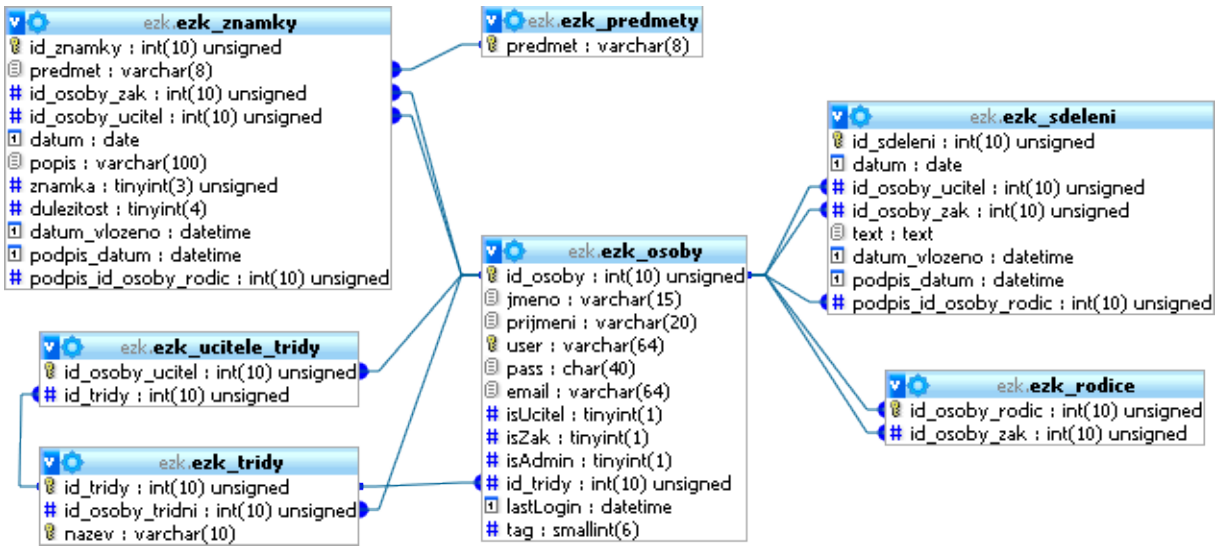
Prvním krokem před psaním vlastní webové aplikace je návrh databázové struktury. Pro návrh byl použit webový nástroj PhpMyAdmin, který umožňuje i následnou vizualizaci vytvořené databáze a relací mezi tabulkami.

U názvů všech tabulek je použit prefix „*EZK_*“, který je možné libovolně změnit a nahradit jiným. V samotné aplikaci je v tom případě třeba pozměnit hodnotu *DBPREFIX* definovanou v souboru `config\mysql.php`. Zde se též nastavují údaje pro přístup do databáze. Tato možnost slouží k případnému vyloučení kolize názvů tabulek na serverech, kde je k dispozici pouze jedna sdílená databáze pro všechna data. Je však

velmi doporučeno uložit tabulky do vlastní databáze, a to s vlastním přístupovým heslem.

Kromě spojovacích tabulek je všude využit umělý celočíselný primární klíč s nastaveným „autoincrement“.

4.1.1 Tabulky a vzájemné vazby mezi nimi



EZK_OSOBY

Hlavní tabulka. Obsahuje informace o všech uživatelích v informačním systému. Atributy pak určují typ uživatele a jeho oprávnění.

Id_osoby: Umělý primární klíč.

Jmeno: Křestní jméno uživatele.

Prijmeni: Příjmení uživatele.

User: Uživatelské jméno, které uživatel využívá pro přihlášení do systému. Atribut má nastaven unikátní index.

Pass: Otisk hesla uživatele. V tomto atributu je uložen pouze SHA1 hash, nikoliv samotné heslo. Původní řetězec hesla tedy není možné ze systému již nikdy získat.

Email: Emailová adresa uživatele. Je využita hlavně pro zaslání přístupového hesla (při založení uživatele, případně při ztrátě hesla).

IsUcitel: Atribut nabývá hodnot 0 nebo 1 a definuje oprávnění učitele a úkonů s ním spojených, tedy zadávání známek a sdělením rodičů. Tento uživatel pak dále může být třídním učitelem.

IsZak: Atribut nabývá hodnot 0 nebo 1 a definuje statut žáka. Žák pak může být zařazen do nějaké studijní třídy.

IsAdmin: Atribut nabývá hodnot 0 nebo 1 a nastavuje uživateli oprávnění spojené s administrací celého systému.

Id_tridy: Hodnota má význam pouze pro uživatele „žáka“. Přiřazuje, v jaké třídě se tento uživatel právě nachází. Vyřazení uživatelé mají tuto hodnotu NULL.

LastLogin: Datum posledního přihlášení. Pokud se uživatel ještě nepřihlásil, je hodnota NULL.

Tag: Pomocný atribut, který je využíván pro hromadné přeřazování uživatelů mezi jednotlivými třídami. Při složitějších operacích si tak systém označí, kterého uživatele již přesunul a který na přesun teprve čeká. V normálním stavu by měl být NULL.

EZK_RODICE

Tabulka obsahuje seznam vazeb, které popisují rodinný vztah Rodič – Potomek. Tím se definují oprávnění, který rodič (uživatel) může prohlížet a podepisovat známky kterého dítěte.

Id_osoby_rodic: Udává, která osoba je v daném vztahu rodičem.

Id_osoby_zak: Udává, která osoba je v daném vztahu potomkem (dítětem, žákem).

EZK_TRIDY

Záznamy obsahují seznam všech aktuálních tříd, které jsou v systému definované.

Id_tridy: Umělý primární klíč.

Id_osoby_tridni: Třídní učitel dané třídy. Pokud není zadán, je hodnota NULL.

Nazev: Název třídy. Např. „2.A“.

EZK_PREDMETY

Seznam předmětů, které jsou na škole vyučovány. Tabulka je definována z důvodu vyloučení překlepů a dvojího různého zápisu názvu jednoho předmětu.

Predmet: Zkratka předmětu. Např: „ČJ“ pro český jazyk. Název musí být unikátní, tedy nelze mít dva předměty se shodným názvem.

EZK_UCITELE_TRIDY

Tabulka popisuje oprávnění, žákům kterých tříd může daný učitel zapisovat známky. Je-li učiteli odebráno oprávnění, nemůže zapisovat nové známky, může však jím již zadanou známku smazat, je-li třeba,

Id_osoby_ucitel: Učitel vlastnící oprávnění.

Id_tridy: Třída, do které může *Id_osoby_ucitel* známky zapisovat.

EZK_ZNAMKY

Seznam všech známek, které jsou v systému zadány.

Id_znamky: Umělý primární klíč.

Predmet: Předmět, ke kterému známka patří

Id_osoby_zak: Žák, kterému známka patří.

Id_osoby_ucitel: Učitel, který známku zadal.

Datum: Datum, ke kterému byla známka daná, například datum testu, písemky či zkoušení. Hodnotu tohoto atributu může učitel při zadání ovlivnit a změnit.

Popis: Informace, čeho se známka týkala. Například „Písemný test“.

Znamka: Hodnota známky (1 až 5). Známka nemusí být zadána (hodnota NULL). Možnost NULL lze použít, pokud je třeba zdůraznit, že žák při psaní nějakého hromadného testu chyběl, nebo z jiného důvodu z písemky známku nedostal.

Dulezitost: Označuje váhu této známky při hodnocení celkového prospěchu žáka. Atribut může nabývat hodnot:

1 => 'velmi nízká' 2 => 'nízká'
3 => 'normální' 4 => 'vysoká'
5 => 'VELMI VYSOKÁ'

Datum_vlozeno: Obsahuje přesný časový údaj, kdy byla známka fyzicky uložena do databáze. Uživatel nemůže tuto hodnotu ovlivnit, vkládá se automaticky.

Podpis_datum: Časový údaj, kdy rodiče danou známku podepsali. Není-li známka zatím podepsána, je hodnota NULL.

Podpis_id_osoby_rodic: Informace, který rodič známku podepsal. Není-li známka zatím podepsána, je hodnota NULL. Pozor! Nastane-li situace, že rodič známku podepíše, a následně je později ze systému odstraněn, je databází atribut nastavena na NULL; atribut „Podpis_datum” však zůstane nastaven. Kontrolu, jestli je známka podepsána je tedy nutné provádět právě atributem *Podpis_datum*.

EZK_SDELENI

V této tabulce se uschovávají všechna sdělení rodičům, která jsou učiteli do systému zadána. Všechna sdělení se vztahují k žákům, nikoliv přímo k rodičům. Dané sdělení pak mohou prohlížet všichni rodiče daného žáka, kteří jsou definování v dříve popsané tabulce *EZK_RODICE*.

Id_sdeleni: Umělý primární klíč.

Text: Samotné sdělení.

Id_osoby_ucitel, **Id_osoby_zak**, **Datum, Datum_vlozeno**, **Podpis_datum**, **Podpis_id_osoby_rodic**: Shodný účel a chování atributů jako v tabulce *EZK_ZNAMKY*.

*Poznámka: U tabulek EZK_*ZNAMKY, EZK_SDELENI *existuje měnitelný atribut datum z důvodu předpokládaného zpoždění při zápisu údajů do informačního systému. V případě, že by systém zobrazoval pouze datum fyzického zápisu, vnikaly by rozdíly mezi záznamy v elektronické žákovské knížce a knížce papírové.*

4.2 Autentizační jádro

Nejdůležitější část celého systému je jádro, které se stará o kontrolu a následné udržování informací a oprávnění uživatelů přistupujících do aplikace.

Jádro je fyzicky zcela samostatný kód zapouzdřený ve vlastní třídě *Autentication*. Všechny skripty aplikace hned na svém počátku volají statickou metodu *Autentication::AutoStart*, kterou lze označit za jakýsi konstruktor. Je nutné hned na počátku říci, že celý objekt instance třídy se uchovává v session. Pseudokonstruktor *AutoStart* má pak za úkol zjistit, zda již v session autentizace existuje. Pokud ano, vrací tento uložený objekt. Naopak, pokud je session prázdná, vytváří konstruktor instanci zcela novou. Tu pak do session uloží a zároveň vytvořený objekt předá návratovou hodnotou funkce (stejně jako v případě, kdy v session uložená data najde).

Ukládání do session má své opodstatnění. Jsou-li informace o stavu přihlášení ukládány, nemusí se autentizační jádro při spouštění každého skriptu znovu a znovu dotazovat databáze, zda je uživatel platný a pak o tomto uživateli získávat (opět z databáze) všechny další potřebné informace pro běh celého informačního systému. Informace z databáze se čtou pouze při přihlášení, dále jsou pak již čteny pouze ze session (nikoliv z databáze) – což je rychlejší a k databázi šetrnější. Tento algoritmus lze také označit za jakousi vyrovnávací paměť. Přepnutím hodnoty jednoho z mnoha možných nastaveních lze tuto funkci samozřejmě vypnout a vynutit si stálé a opakované ověřování přímo z databáze uživatelů.

Pokud necháme zmíněnou vyrovnávací paměť zapnutou, musíme si uvědomit, že získané zrychlení a snížení zátěže databázového serveru je vykoupeno rizikem spojeným s nemožností okamžité změny oprávnění. Pokud totiž komukoliv změníme oprávnění či nastavení, všechny změny se projeví teprve ve chvíli, kdy se daný uživatel znovu přihlásí (za přihlášení je považováno i zavření a následné nové otevření prohlížeče v případě, že uživatel povolil funkci „stálého přihlášení“).

Pokud je uživatel ze systému vymazán, s okamžitou platností již nemůže do systému zapsat žádná nová data, a to bez ohledu na cachování oprávnění. Zadávání dat

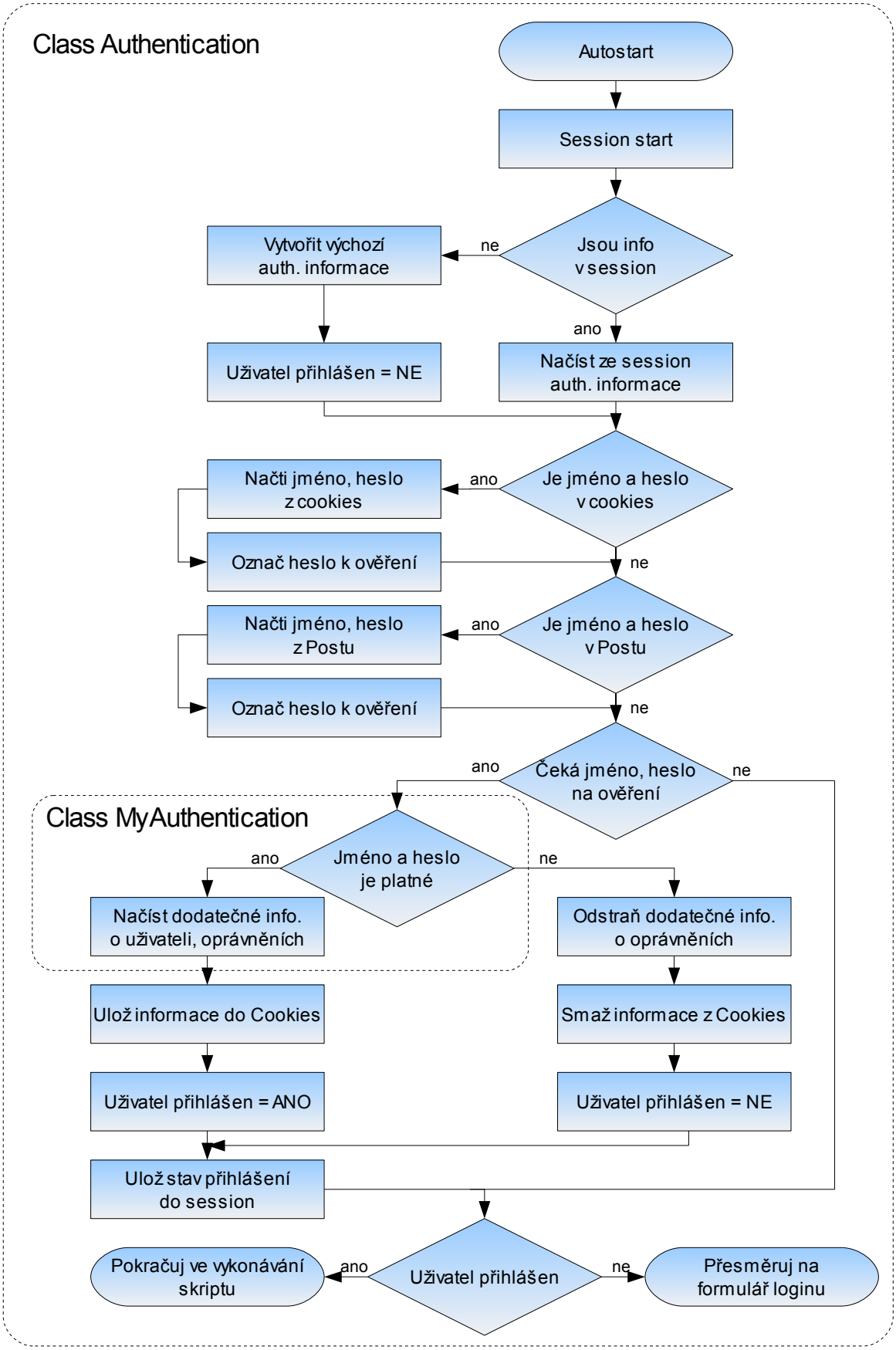
mu totiž neumožní integritní omezení vzájemných relací mezi tabulkami v databázovém systému. Známky i sdělení rodičům jsou opatřeny informací o uživateli, který informace zadal – a ten, jak je zřejmé, již v databázi neexistuje. DB server tedy SQL dotaz odmítne s odůvodněním, že položka popisující, kdo záznam zadal, je neplatná.

Objekt autentizačního mechanizmu má vnitřní stav, který obsahuje informaci o tom, zda je v současné chvíli řádně přihlášen nějaký uživatel. Tento stav (boolean hodnota) lze získat zavoláním metody *getCanLogin()*. Pokud je uživatel přihlášen, lze přistoupením k proměnné *userInfo* získat další podrobnější informace o autentizované osobě. Proměnná *userInfo* je polem, ve kterém kromě jména, emailu a dalších kontaktních informací o uživateli lze najít i jeho oprávnění a data o rodičích (v případě žáka) či potomcích (v případě rodiče).

Třída *Autentication* je napsána jako obecný autentizační mechanizmus, který lze úspěšně použít v jakémkoliv projektu, bez ohledu na strukturu databází či způsob ukládání seznamu uživatelů. Třída samotná totiž obsahuje abstraktní metodu *getUserInfo()*, kterou je třeba doimplementovat ve zděděné třídě *MyAutentication*. Metoda má jeden povinný a jeden volitelný úkol. Povinně musí na základě informací o jménu a heslu (*$this->user* a *$this->passSHA1*) rozhodnout, zda může být uživatel přihlášen. Výsledek svého rozhodnutí metoda předá do boolean proměnné *$this->canLogin*. Zároveň může metoda do *$this->userInfo* zapsat libovolné doplňující informace o přihlašovaném uživateli. Jakým způsobem o přihlášení metoda *getUserInfo()* rozhodne, je samozřejmě zcela na ní.

Autentizace umí automaticky přijímat přihlašovací informace i z POST dat, cookies, ručního vložení pomocí funkce a samozřejmě session. Pokud uživatel kterékoliv stránce informačního systému pošle z webového formuláře jméno a heslo, skripty POST data přijmou a pokusí se podle nich uživatele přihlásit. Položky formuláře pak pro funkčnost musí obsahovat pole *auth_user* a *auth_pass* nebo *auth_passSHA1*. Po síti se tedy heslo nemusí vůbec předávat. Postačí, když klient zašle SHA1 otisk. Stejně tak v cookies se ukládá pouze otisk.

Pokud uživatel při přihlášení zvolí možnost trvalého loginu, je jméno a SHA1 otisk hesla uložen v cookies (v opačném případě se neukládá). V případě, že autentizační mechanizmus později nenajde jméno a heslo v session, zkouší informace hledat právě v cookies, čímž zajistí automatické obnovení přihlášeného uživatele. Možnost ukládání do cookies je možné z bezpečnostních důvodů v nastavení zcela vypnout a znepřístupnit.


Class Authentication
Autostart
Session start
Jsou info v session
ne
Vytvořit výchozí auth. informace
ano
Uživatel přihlášen = NE
Načíst ze session auth. informace
Je jméno a heslo v cookies
ano
Načti jméno, heslo z cookies
Označ heslo k ověření
ne
Je jméno a heslo v Postu
ano
Načti jméno, heslo z Postu
Označ heslo k ověření
ne
Čeká jméno, heslo na ověření
ano
ne
Class MyAuthentication
Jméno a heslo je platné
ano
ne
Načíst dodatečné info. o uživateli, oprávněních
Odstraň dodatečné info. o oprávněních
Ulož informace do Cookies
Smaž informace z Cookies
Uživatel přihlášen = ANO
Uživatel přihlášen = NE
Ulož stav přihlášení do session
Uživatel přihlášen
ano
ne
Pokračuj ve vykonávání skriptu
Přesměruj na formulář loginu

4.3 Ověření oprávnění

Aplikace ověřuje oprávnění z informací uvedených v poli *userInfo* autentizačního jádra. V této proměnné lze najít všechny hodnoty položek přihlášeného uživatele získané z databázové tabulky EZK_OSOBY a dále pak informace o rodičích a potomcích. Pokud je uživatel učitel, lze se dozvědět zároveň informace o třídách, kde je osoba třídním učitelem a o oprávnění zadávat známky jednotlivým třídám.

Následuje názorná ukázka struktury proměnné *userInfo*:

```
Array
(
    [id_osoby] => 31
    [jmeno] => Jana
    [prijmeni] => Dimunová
    [user] => jana.dimunova
    [pass] => 356a192b7913b04c54574d18c28d46e6395428ab
    [email] => jana.dimunova@nothing.cz
    [isUcitel] => 1
    [isZak] => 0
    [isAdmin] => 1
    [id_tridy] =>
    [lastLogin] =>
    [tag] =>
    [potomci] => Array
        (
            [0] => Array
                (
                    [id_osoby] => 30
                    [jmeno] => Martin
                    [prijmeni] => Žaloudek
                    [user] => martin.zaloudek
                    [pass] => 8cb2237d0679ca88db6464eac60da96345513964
                    [email] =>
                    [isUcitel] => 1
                    [isZak] => 1
                    [isAdmin] => 1
                    [id_tridy] =>
                    [lastLogin] =>
                    [tag] =>
                )

            [1] => Array
                (
                    [id_osoby] => 39
                    [jmeno] => Ana
                    [prijmeni] => Malá
                    [user] => ana.mala
                    [pass] => 15c0f00fa9cdd437b2161edbaf051cba9a96abf5
                    [email] =>
                    [isUcitel] => 0
                    [isZak] => 1
                    [isAdmin] => 0
                    [id_tridy] => 32
                    [lastLogin] =>
                    [tag] =>
                )

        )
```

```
    [id_osoby_potomci] => Array
        (
            [0] => 30
            [1] => 39
        )

    [tridy_tridni] => Array
        (
            [0] => Array
                (
                    [id_tridy] => 36
                    [id_osoby_tridni] => 31
                    [nazev] => 4.A
                )

        )

    [opravneni_id_tridy] => Array
        (
            [0] => 30
            [1] => 32
            [2] => 36
        )

)
```

Výše uvedené informace jsou dostupné kdykoliv, v kterémkoliv skriptu přes globální proměnnou *$GLOBALS[CONFIG]['auth']->userInfo*.

Samotné ověřování oprávnění k jednotlivým akcím se provádí ve dvou fázích. V prvé řádě se podle oprávnění skrývají nebo zobrazují formuláře a různé navigační prvky (odkazy, tlačítka, menu, atd.). V druhé fázi je oprávnění kontrolováno na začátku skriptů a před každou požadovanou akcí, například přidáním známky. Situace, kdy se zjistí, že uživatel oprávnění nemá, by nastat neměla. Uživateli se skryje příslušný odkaz do nepovolené sekce, takže by se do ní nikdy neměl „doklikat“. Pokud se tak tedy ale přeci jen stane, značí to buď nějaký problém nebo pokus o neoprávněné proniknutí do informačního systému. Při jakémkoliv odhaleném narušení bezpečnosti je volán příkaz *die*, který zajistí, okamžité ukončení skriptu a zamezení případných dalších problémů či napadení tohoto skriptu útočníkem.

Příklad

Následující ukázka je z počáteční kontroly skriptu zobrazujícího známky konkrétního žáka s ID předeným v *$_GET['id_osoby_zak']*.

```
if (
         !$GLOBALS[CONFIG]['auth']->userInfo['isUcitel']
      && !$GLOBALS[CONFIG]['auth']->userInfo['isAdmin']
      &&  $GLOBALS[CONFIG]['auth']->userInfo['id_osoby']
            != (int)$_GET['id_osoby_zak']
      && !in_array($_GET['id_osoby_zak'],
            $GLOBALS[CONFIG]['auth']->userInfo['id_osoby_potomci'])
)
{
      die ('Nemate oprávnění prohlížet tento obsah.');
}
```

Z předchozího zdrojového kódu jsou zřejmá následující potřebná oprávnění. Přihlášený uživatel musí splňovat jednu z podmínek (pokud nesplňuje ani jednu, dojde k ukončení skriptu):

1. Jde o učitele
2. Jde o administrátora
3. Jde o žáka, jehož známky se mají zobrazit
4. Jde o rodiče žáka, jehož známky se mají zobrazit

4.4 Zobrazovací engine

Aplikace EŽK využívá zobrazovacího enginu, který umožňuje oddělit okolní HTML kód od vlastního hlavního obsahu generovaného skripty systému. Okolním kódem jsou myšleny záhlaví, zápatí, hlavní menu a další vedlejší části webu.

Ve výsledku se do URL píše pouze vlastní název volaného skriptu bez jakýchkoliv parametrů. Parametry je samozřejmě možné předat zcela libovolné, dle potřeb a požadavků operace, kterou má skript vykonat. Jde o opak řešení, ve kterém se vždy volá jeden stejný skript a daný předávaný parametr udává, kterou stránku má skript zobrazit.

Systém je navržen tak, že v každém skriptu je nutné vložit pouze tři příkazy, které zařídí výpis veškerého okolního kódu. Pak už je možné psát pouze algoritmy

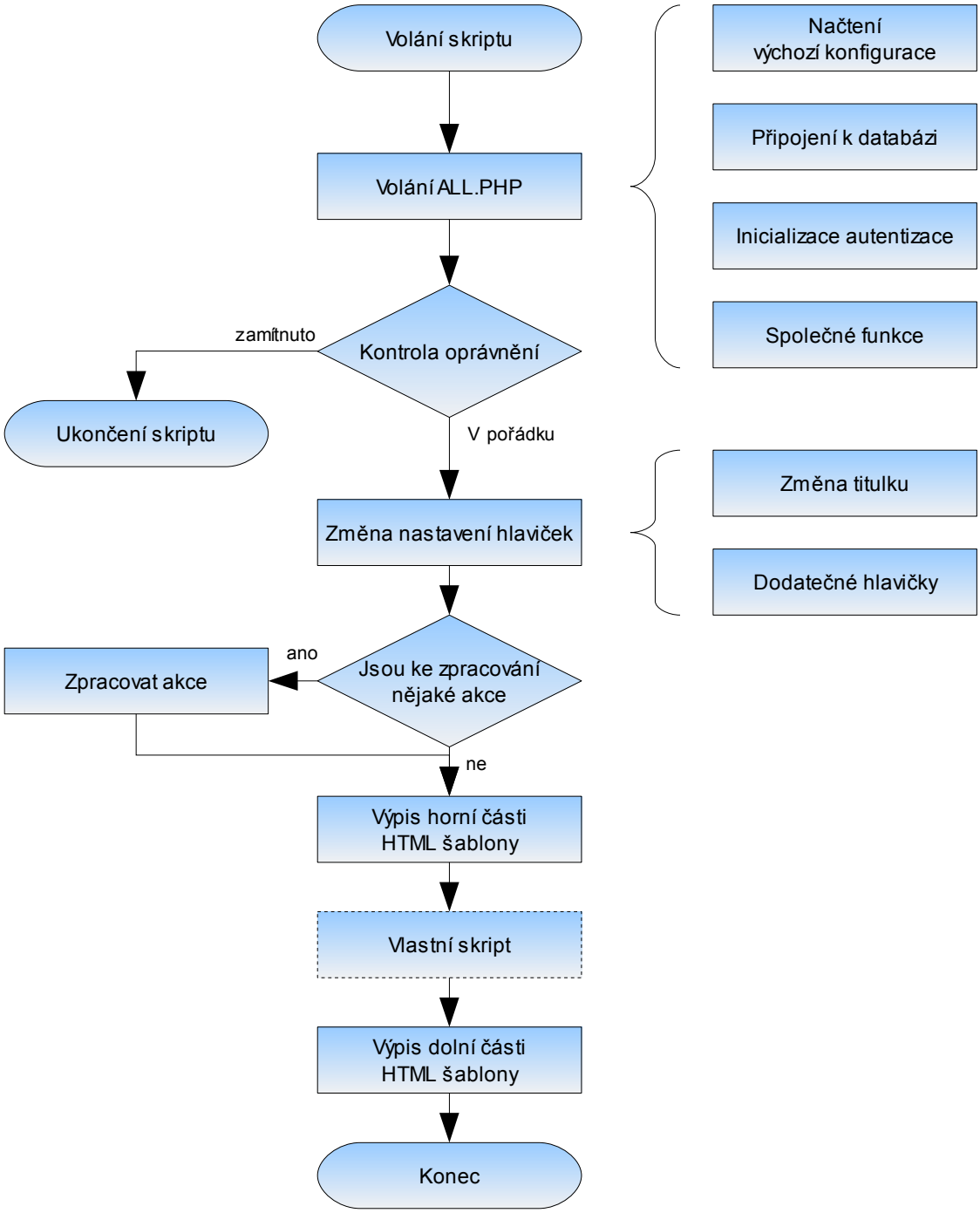
a příkazy týkající se vlastní funkce skriptů. V reálné situaci se většinou využívají ještě další příkazy, které mění například titulek webové stránky, případně přidává doprovodné volání javascriptu, či CSS stylu.

Příklad

```
<?php
      require_once 'engine/all.php';
      $GLOBALS[CONFIG]['auth']->redirectOnLoginFailed();


      $GLOBALS[CONFIG]['header']['title'] .= ' - zkušební stránka';
      $GLOBALS[CONFIG]['header']['others'] .= '<link rel="stylesheet"
            href="css/doplnkovy.css" type="text/css" />';

      require_once 'engine/html_top.php';

      echo '<p>Zde již vkládáte obsah vlastního obsahu...</p>';

      require_once 'engine/html_bottom.php';
?>
```

Volání *all.php* zařídí importování společných funkcí, připojení k databázi a nastavení různých výchozích hodnot. Zároveň je v tomto skriptu inicializováno autentizační jádro. Druhý řádek pak zajistí přesměrování na stránku požadující přihlášení v případě, že není autentizován žádný uživatel. Pokud přesměrování neproběhne (tj. přihlášení je v pořádku), je nastaven titulek a doplňující CSS styl. Pak se již nechá HTML obsah vypsat. Obsah je rozdělen na části „top“ a „bottom“, mezi kterými skript vypisuje všechen požadovaný obsah, který má uživatel vidět.

Zobrazovací engine zároveň umožňuje zobrazit chybová a informační hlášení. Stačí kdekoliv před vypsaním obsahu přidat do globální proměnné *$chyby* nebo *$informace* jakýkoliv text reprezentující chybové nebo informační hlášení. Této funkce se často využívá při zpracování formulářových dat nebo při chybách zápisu do databáze.



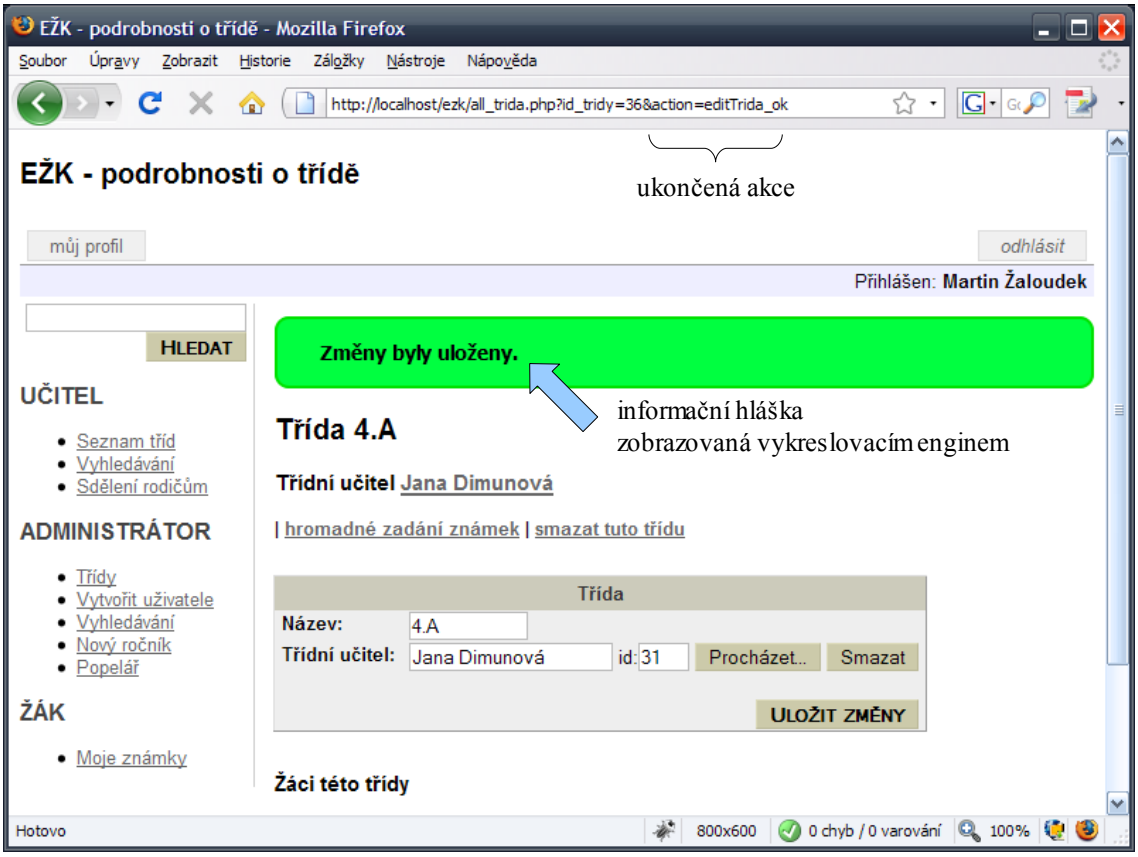
4.5 Zpracování akcí

Akce jsou většinou situace, kdy je třeba přidat, pozměnit či smazat nějakou položku v databázi. V našem případě jde například o zadání známky, vytvoření

uživatele, změnu oprávnění, apod. Obecně sem patří také všechna zpracování formulářových dat.

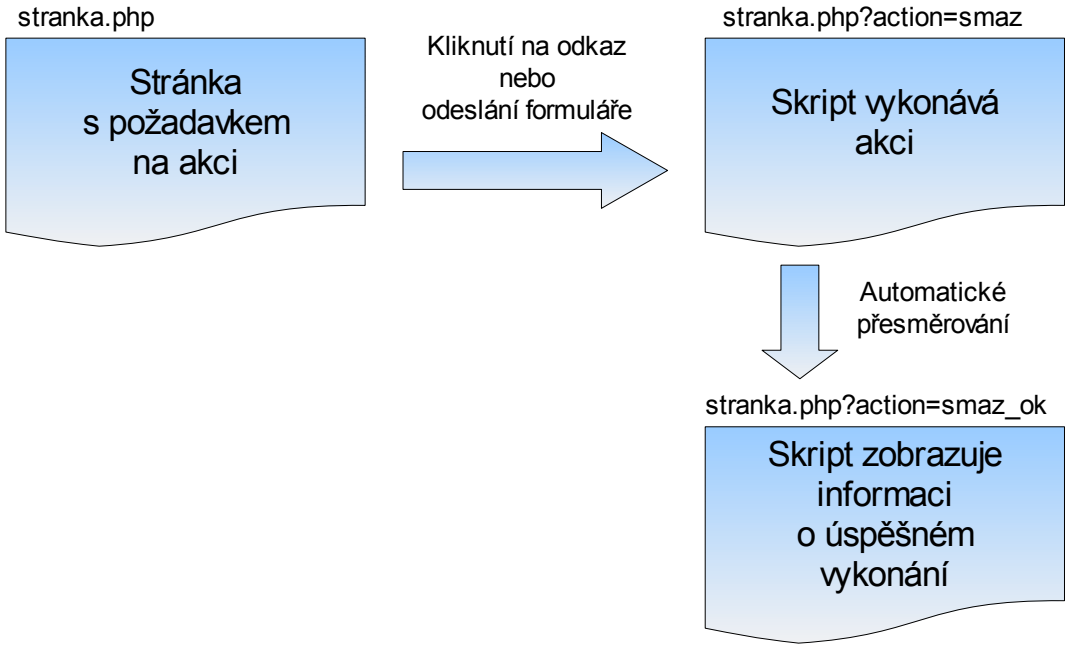
Aplikace EŽK řeší požadavky na akce a jejich další zpracování následovně: Každá akce má svůj název a skript, který ji vykoná. Pokud se nepředpokládá, že by byla akce volána z několika různých míst, je vykonávající skript stejný jako skript, který akci volá. Volání probíhá pomocí GET parametru *action*, jehož hodnota odpovídá názvu akce. Jsou-li akcí požadovány nějaké další upřesňující parametry, předají se buď v GETu (v případě volání pomocí odkazu), nebo v POSTu (většinou v případě zpracování formuláře).

Požadavek na vykonání pak skript pozná jednoduše – přítomností parametru *action*. V případě, že je akce rozpoznána, přejde program na vykonání potřebných změn (například dojde k fyzickému zápisu do databáze). Pokud vše proběhne v pořádku, předá se HTTP hlavička *Location*, ve které je žádost na nové spuštění skriptu, tentokrát bez volání jakékoliv události. Často se také v předávaných parametrech *Location* udává žádost o zavolání další události. Ta však už ale jen zajistí vypsání nějaké informace o tom, že vše proběhlo v pořádku.



Přesměrování pomocí *Location* se provádí z důvodu, aby v případě, že uživatel později stiskne v prohlížeči navigační tlačítko zpět, nedošlo k znovuprovedení této akce. To by totiž mohlo mít za následek například nechtěné opětovné vložení známky žákovi, či zápis jiných již duplicitních dat do databáze.

Je důležité, aby se všechny akce vykonávaly ještě před provedením jakéhokoliv výpisu, tedy před voláním *html_top.php*. Jedině tak je totiž možné odeslat HTTP hlavičku s žádostí o přesměrování.



5 Testování aplikace

Testování programu se provádí ve všech částech životního cyklu projektu. Již fáze návrhu aplikace obsahuje některé prvky testování. Jedná se však pouze o testovaní teoretické, tzv. „na papíře“, kdy se po vytvoření návrhu databází a soustavy modulů aplikace předpokládají různé scénáře chování uživatele a následně se modeluje, jak by na tato chování aplikace aplikace zareagovala. Skutečné testování produktu přichází až ve fázi implementace.

5.1 Testování při vývoji

Tato fáze testování se skládá zejména z vyzkoušení všech modulů, skriptů aplikace a všech akcí a možností v nich.

Protože se do projektu EŽK budou ukládány citlivé informace, byl kladen důraz zejména na úroveň zabezpečení, konkrétně pak odolnost proti:

1. získání informací bez autentizace
2. přístupu do sekcí a funkcí, do nichž uživatel nemá oprávněním
3. podvržení autora vkládaných údajů
4. vykonání vlastního SQL dotazu (SQL injection)
5. vstupu neplatných a nesmyslných údajů

Při programování a testování je stále zapnuto vypisování chybových zpráv PHP, a to včetně varování a oznámení (tj. warning a notify). Tím lze velmi efektivně zajistit

odhalování i drobných chyb vzniklých při psaní samotného kódu. Současně se automaticky validuje veškerý HTML výstup, který aplikace generuje. To se provádí pomocí zásuvného modulu „Tidy HTML validator“, který je volně poskytován pro prohlížeč Mozilla Firefox.

Další krok testování se týkal zobrazovaného obsahu a použitého JavaScriptového kódu. Zde byl kladen důraz na plnou funkčnost v majoritních prohlížečích, za které byl označen Internet Explorer 8, Internet Explorer 7, Mozilla Firefox 3, Google Chrome, Opera 9 a Safari. Zastaralý, avšak stále používaný Internet Explorer 6, podporován není, funkčnost aplikace je v něm však předpokládána – vzhledem k velké kompatibilitě s IE7, IE8 (hlavně v oblasti JavaScriptu).

5.2 Testování zadavatelem

Zadavatel sám při předvádění aplikaci testoval. Tato část testování se týkala převážně ovládání a orientace v programu. První seznámení ukázalo, že navržená struktura nabídek a formulářů přesně odpovídá původním představám zadavatele.

K důkladnému seznámení s aplikací je však třeba mnohem více času a testujících osob. Lze tedy předpokládat, že požadavky na drobné úpravy se objeví až během zkušebního provozu.

5.3 Zkušební provoz

Od příštího školního roku, tedy září 2009, je naplánován začátek zkušebního provozu ve skutečném prostředí. Ten bude spočívat ve zprovoznění aplikace na serveru školy a následném zavedení zkušebního vzorku žáků a jejich rodičů do informačního systému EŽK. Vzorek se bude dle současného plánu skládat z jedné až dvou vybraných tříd.

Během tohoto provozu by měla být všechna chybová hlášení přesměrována do logovacího souboru, případně přesměrována i na email administrátora, aby bylo možné okamžitě odhalit a následně odstranit vzniklý problém.

Provoz se zkušebním vzorkem uživatelů bude probíhat několik měsíců. Pokud bude vše v pořádku, bude následně systém rozšířen na celou základní školu, čímž dojde k přechodu na ostrý provoz.

6 Závěr

Hlavním cílem této diplomové práce bylo navržení a implementace aplikace, která bude existovat jako alternativa k v současnosti již existujícím řešením a která tak bude doplňovat vniklou mezeru na trhu informačních systémů s podobným zaměřením. Vytvořená aplikace splnila všechny představy zadavatele a je připravena ke zkušebnímu provozu v reálném prostředí. Tento provoz by měl začít počátkem příštího školního roku, tedy v září roku 2009.

Během tvorby informačního systému došlo po projednání se zadavatelem k jedné změně ve funkčnosti programu. V původním návrhu podporovala aplikace ukládání všech dat i několik let nazpět. To umožňovalo prohlédnout si studijní prospěchy studentů za celou dobu, po kterou byli v informačním systému vedeni. Vedení školy však usoudilo, že tato funkce je nadbytečná a nepřináší dostatečný užitek. Bylo tedy přistoupeno k zjednodušení. K navržení této změny však bohužel došlo až ve fázi implementace, kdy byla aplikace již částečně funkční. Důsledkem toho byl nutný celkem velký zásah do již vytvořené části aplikace. Reálně šlo o smazání a přepsání několika set řádků zdrojového kódu rozmístěného v různých částech programu.

Všeobecně užitečný přínos vidím ve vytvoření aplikace, která není běžně k dispozici a není možné si ji na softwarovém trhu zakoupit. Osobní přínos pak je v prohloubení znalostí a zkušeností týkající se tvorby aplikací s víceuživatelským přístupem řízeným na základě oprávnění. V oblasti programování je pak velmi zajímavé autentizační jádro, které lze během velmi krátké chvíle použít v libovolném budoucím projektu.

Při návrhu i tvorbě aplikace bylo použito výlučně Open-source a freeware aplikací. Výsledný zdrojový kód je tvořen cca 4000 řádky a zabírá 150 KiB paměti.

7 Literatura

[1] Wikipedia, *SWOT* [online].
Dostupný z WWW: <http://cs.wikipedia.org/wiki/SWOT>.

[2] IGNUM – ještě lepší hosting [online].
Dostupný z WWW: <http://ignum.cz/web/df/cz/?typ=wbh_ssl>.

[3] GeoTrust – SSL Certificates from a Leading SSL Certificate Authority [online].
Dostupný z WWW: <http://www.geotrust.com/>.

[4] LACKO, Luboslav, *SQL: Hotová řešení*.
Tomáš Edér. Brno : Computer press, 2003.
298 s. ISBN 80-7226-975-5.

[5] ŽÁK Karel, *Modelování databází* [online].
Internet Info s.r.o., 1998-2008 [cit. 2008-05-13].
Dostupný z WWW:
<http://www.root.cz/clanky/modelovani-databazi/>.

8 Slovník pojmů a zkratek

Autentizace Je proces ověření proklamované identity subjektu.

EŽK Zkratka pro výraz „Elektronická žákovská knížka“.

GET Je jednou ze základních dvou metod přenosu dat serveru ve skriptovacím jazyce PHP. Jde o předávání parametrů v URL adrese.

IE Internet Explorer

IIS Internetové informační služby

IS Informační systém

MySQL Je databázový systém, vytvořený švédskou firmou MySQL AB.

PHP Je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek.

POST Je jednou ze základních dvou metod přenosu dat serveru ve skriptovacím jazyce PHP. Jde o metodu, při které se data přenášejí přímo serveru beze změny URL.

Prefix Předpona. V databázovém systému je často používána k označení různých tabulek, které patří jedné aplikaci.

Session Relace. V informatice označuje trvající síťové spojení mezi klientem a serverem, zahrnující výměnu většího množství paketů.

SHA1 (Secure Hash Algorithm) je rozšířená hashovací funkce, která vytváří ze vstupních dat výstup (otisk) fixní délky.

SQL Structured Query Language je standardizovaný dotazovací jazyk používaný pro práci s daty v relačních databázích.

SWOT analýza Je metoda, pomoci které je možno identifikovat silné (ang: Strengths) a slabé (ang: Weaknesses) stránky, příležitosti (ang: Opportunities) a hrozby (ang: Threats), spojené s určitým projektem, typem podnikání, opatřením, politikou apod.

Ve slovnících jsou použity texty a definice z projektu *Wikipedia*.

9 Přílohy

[1] CD disk

- elektronická podoba diplomové práce ve formátu PDF
- zdrojové kódy aplikace EŽK