TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2612 – Elektrotechnika a informatika

Studijní obor: Informatika a logistika

Databázový informační systém pro základní školy

Information System for Primary School

Bakalářská práce

Autor: Matěj Liederhaus

Vedoucí práce: RNDr. Klára Císařová, Ph.D.

Konzultant:

V Liberci 20.5.2011

PODĚKOVÁNÍ

Děkuji RNDr. Klára Císařová, Ph.D. za vedení mé práce za odborné konzultace a obecné rady.

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou 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 bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své bakalářské 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).

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

Datum

Vlastnoruční podpis

Anotace

Tato práce popisuje vývoj informačního systému pro účely základní školy. Stručně je zde uvedena historie jednotlivých použitých technologií. V práci je uveden postup, jak je tvořena datová a funkční analýza pro tento informační systém. Dále je zde popsán vývoj, jakým byla webová aplikace vytvářena. Jsou uvedeny bezpečnostní rizika a opatření, která byla provedena. Práce obsahuje i postup implementace do reálného prostředí základní školy a problémy s tím spojené.

Klíčová slova:PHP, HTML, SQL, informační systém, datový model, funkční analýza

This paper describes the development of an information system for primary schools. Briefly stated, there is the history of technology. The paper describes the procedure as it is formed data and functional analysis for this information system. Beyond describes the development, which was created web application. They are given the security risks and measures which have been implemented. The work includes the implementation process in real elementary school and associated problems.

Key words: PHP, HTML, SQL, information system, data model, functional analysis

Obsah

1. Seznam obrázků

2. Úvod

Informačních systémů je v dnešní době veliké množství. Co se týče informačních systémů pro účely škol, tak nejrozšířenější je IS Bakalář a Relax-Keš. Mnoho škol si na rozdíl od možnosti těchto dvou IS zadají zakázku na svůj vlastní IS, který je vyroben přímo na potřeby konkrétní školy. Důvody mohou být nedostatek finančních zdrojů, nedostatečné služby dosavadních IS nebo naopak zbytečně naddimenzované služby IS.

Zadání mé práce bylo vytvořit jednu takovou aplikaci, která by realizovala informační systém. Jako modelovou školu jsem si zvolil Základní a Mateřskou školu ve Višňové. Pro informační systém jsem zvolil název Elektronická žákovská knížka.

2.1 Cíle práce

Celá práce není zaměřena pouze na vytvoření webové aplikace, ale také na to abych se seznámil s postupy tvorby webových aplikací obecně. Jde o podrobné rozebrání prostředí, kam bude aplikace implementována. Vytvoření datové struktury tohoto prostředí a zmapování funkčních analýz. To znamená nalezení vztahů mezi jednotlivými zkoumanými objekty. Následně mám z této datové a funkční analýzy vytvořit odpovídající databázi realizovanou na vhodném enginu.

Nad touto databází je následně mým cílem vytvořit webovou aplikaci, která bude sloužit jakožto informační systém pro rodiče žáků, kteří chodí na základní školu. Zde se jedná o pohledy do databáze ze strany rodičů a žáků a ze strany učitelů jde o zápisy a vkládání údajů do této databáze.

Zároveň by celá webová aplikace měla být dostatečně bezpečná, aby se nedala zneužívat. Příklad zneužití je zcela jasný: Žák neoprávněně se dostane k možnosti zapsat nebo pozměnit si sám známku.

Celý takto vytvořený informační systém jsem zároveň dostal za úkol ověřit. Jedna věc je testování ve zkušebních podmínkách a druhá je ostrý provoz. Ostrý provoz budu realizovat na reálné základní škole.

2.2 Jaké volím prostředky pro vývoj aplikace

Elektronickou žákovskou knížku jsem se rozhodl vyvíjet jako webovou aplikaci. Důvodem byl snadný přístup k aplikaci přes internet, její používání a také můj zápal pro skriptovací jazyk PHP a tvorbu dynamických webových stránek. Při vývoji aplikace jsem si ovšem nevystačil pouze s HTML a PHP. Ukládání dat jsem realizoval pomocí databáze MySql, která se mi zdála jako nejvýhodnější pro její snadnou implementaci na server, velikou oblibu na většině domén a snadnou správu pomocí PhpMyAdmin. S databází se spojuje databázový jazyk SQL. Pro dynamické a uživatelsky příjemné ovládání aplikace jsem použil funkcí JAVA Scriptu, který také využívám k animacím. Aby celá aplikace byla i hezká na pohled, použil jsem CSS styly.

3. Historie použitých technologií

Protože jsem za celou dobu vývoje informačního systému pracoval s různými technologiemi tak bych je také rád představil. V následujících kapitolách jsou stručně popsány historické vývoje jednotlivých technologií.

3.1 PHP

Zkratka PHP se poprvé objevila v roce 1994, kdy byla napsána binární část Common Gateway Interface (CGI) v programovacím jazyku C. Původní označení PHP znamenalo Personál Home Page (osobní domácí stránky) a prvotním tvůrcem byl dánsko-grónský programátor Rasmus Lerdorf. Lerdorf zpočátku vytvořil tyto nástroje pro osobní domácí stránky (Personal Home Page) za účelem možné záměny s malou skupinou skriptů v Perlu, které chtěl používat pro údržbu osobní domovské stránky. Původní skripty měly zajistit běh úloh jako například zobrazení obsahu nebo zaznamenávání návštěvnosti jeho stránek. Tento binární kód ještě tentýž rok spojil s jiným programem, který sám napsal. Spojením s Form Interpreter tak vznikla kombinace PHP/FI, která představovala větší možnosti a zajišťovala mnohem větší funkčnost. PHP/FI obsahovala širokou implementaci pro programovací jazyk C a navíc tato verze byla schopná komunikovat s databázemi, což umožnilo tvorbu prvních jednoduchých dynamických webových aplikací. Lerdorf 8. června 1995 veřejně vydal PHP, aby mohl najít co největší množství neodhalených chyb a jejich opravami tak

zdokonalit zdrojový kód PHP. Tato verze byla pojmenována jako PHP verze 2 a již poskytovala základní funkčnost srovnatelnou s dnešní PHP. To zahrnuje například proměnné ve stylu Perlu, zpracování formulářů a možnost vložit HTML kód. Syntaxe byla podobná Perlu, ale byla především jednodušší, do jisté míry omezenější a méně konzistentní.

Zeev Suraski a Andi Gutmans, dva Izraelští vývojáři na Technion IIT, přepsali parser v roce 1997, vytvořili tak základ PHP 3 a změnili název jazyka na rekurzivní zkratku PHP = PHP: Hypertext Preprocesor. Tým vývojářů oficiálně vydal PHP/FI 2 v Listopadu 1997 po měsíčním testování beta verze. Poté začalo veřejné testování PHP 3, a její oficiální uvolnění přišlo v červnu 1998. Zeev Suraski a Andi Gutmans poté začali opětovné přepisování jádra PHP a vydali Zend Engine v roce 1999. Založili firmu Zend Technologies v Ramat Gan, Izrael. [3]

Dne 22. května 2000 byla vydána verze PHP 4 postavená na Zend Engine 1.0. Dne 13. června 2004 byla představena verze PHP 5, která již stojí na novém Zend Engine II. PHP 5 obsahuje nové rysy jako je vylepšená podpora pro objektově orientované programování, PHP Data Objects extension (ta definuje lehké a konzistentní rozhraní pro napojení k databázím) a nesčetné množství výkonových vylepšení. PHP 4 se již dále nevyvíjí a pro tuto verzi se nebudou vydávat ani žádné bezpečnostní aktualizace. [3]

V roce 2008 se stává PHP 5 jedinou stabilní verzí, která se vyvíjí. Později se zjistilo, že zde chybí static binding a bude přidáno v PHP 5.3. PHP 6 se bude zároveň vyvíjet s verzí PHP 5. Mezi hlavní změny patří odebrání register_globals, magické uvozovky a safe mode. [3]

3.2 HTML

HTML se poprvé objevilo v roce 1991 po ukončení dvouletého projektu WWW, který vedl Tim Berners-Lee a Robert Caillau. Celý projekt vznikl v CERNu a cílem bylo vyřešení sdílení vědeckých výsledků a informací mezi vědci ve světě. Po skončení projektu byla uveřejněna neformální verze HTML, která umožňovala rozčleňovat text do několika logických úrovní, nabízela používání několika druhů zvýraznění textu a podporovala řazení odkazů a obrázků do textů.

S tímto projektem byl vyvinut software WWW ,jenž obsahoval jak prohlížeč tak i editor, ve kterém uživatel nemusel jazyk HTML znát. Implementování editoru se zdálo pro autora tehdejšího prohlížeče Mosaic Marc Anderssen z NCSA (National Center for Supercomputing Applications) příliš obtížné a tak dnes autoři profesionálních stránek jazyk HTML musejí znát.

Postupem času jak se navyšovali požadavky uživatelů WWW, producenti rozličných webových prohlížečů obohacovali HTML o nové prvky. Důvodem je komerce a nastává chaos. Chaos trvá až do vypracování první standardizované verze HTML 2.0 roku 1994 vydaná pod záštitou komunity IETF (Internet Engineering Task Force) a pracující na podpoře SGML. Tato verze zachycovala všechny v té době používané prvky HTML. Zvláště pak rozšiřovala původní specifikace o interaktivní práci s formuláři a začátek podpory grafického rozhraní.

Následně se vyvíjeli verze HTML+ a HTML 3.0, které ovšem v pozdějších verzích buď nejsou nebo z nich zbyl pouze ořezaný zbytek díky své náročnosti(HTML 3.0).

Roku 1994 je založena organizace World Wide Web Consortium *(*W3C), která se ujala koordinace vývoje standardů HTML a tak na počátku roku 1996 vznikla oficiální specifikace HTML 3.2. Tuto verzi v tehdejší době podporovalo největší množství webových prohlížečů a skládala se z rozšířené verze HTML 2.0 a ořezaného HTML 3.2 (ve své podstatě zůstali jen neúplné prvky pro tabulky). Jako novinka byla přidána podpora CSS stylů.

Vývoj se ovšem nezastavil a již na jaře roku 1997 W3C zveřejnilo plány na další rozšíření HTML. To spočívalo v možnostech použití rámů, podpoře skriptovacích jazyků jako jsou JavaScript a obecného vkládání objektů.

W3C na novou verzi HTML nenechalo dlouho čekat a již v červenci roku 1997 zveřejnilo návrh na HTML 4.0. Tato verze byla v prosinci téhož roku přijata jako standard a na relativně dlouhou dobu zastavil další vývoj jazyka.

V roce 1999 W3C vydalo novou verzi HTML 4.01, která ovšem pouze opravuje některé chyby z předchozí specifikace.

Zároveň se zveřejněním standardu HTML 4.0 W3C vydalo ještě standard XML *(*eXtensible Markup Language*)*. XML se po vydání v roce 1998 stal všudypřítomným formátem umožňujícím výměnu a ukládání dat. Tímto standardem byl ovlivněn i následný vývoj HTML.

XML se na začátku roku 2000 dočkalo své oficiální standardizované verze v podobě XHTML 1.0. Jde o upravenou verzi jazyka HTML, který není závislý na podpoře SGML. Ve své podstatě tyto verze přinesly jen kosmetické změny v podobě rozličných syntaxí. Funkční stránka ovšem byla zachována. XHTML přineslo oproti HTML 4.01 pouze jistou módnost do psaní stránek a proto nebyl pragmatický důvod pro přecházení z HTML. Navíc tou dobou nejpoužívanější prohlížeč Internet Explorer nedokázal XHTML zpracovávat zcela správně.

Vývojáři W3C začali pracovat na nové verzi jazyka XHTML 2.0, ve které se objevili mnohé zajímavé vlastnosti. Bohužel za cenu porušení stávající kompatibility s dřívějšími verzemi jazyků HTML a XHTML. To nakonec vedlo k názoru, že XHTML 2.0 je slepá cesta. Pokračování vývoje webových jazyků se začalo směřovat více na HTML.

V roce 2007 se spojily síly pracovních skupin W3C a WHATWG (The Web Hypertext Application Technology Working Group) a společně zahájili vývoj verzi jazyka HTML 5, který by měl přinést spoustu užitečných funkcí. Tato verze by měla být natolik výhodná, že bude výhodné ji používat hned od jejího vydání. To se bohužel podle optimistů neplánuje dříve než kolem roku 2020.

3.3 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich z tehdejší společnosti Netscape. [3]

V dnešní době se zpravidla používá jako interpretovaný skriptovací jazyk pro WWW stránky. Ve velké míře je přímo vkládaný do HTML kódu stránky. Zpravidla jím jsou ovládány všelijaké interaktivní prvky stránky GUI (textová pole, odesílání formulářů, reakce stránky na aktivitu myši a jiné). Jako další možnost použití JavaScriptu je tvoření různých animací nebo efektů obrázků.

U tohoto jazyka je oproti ostatním skriptovacím jazykům (PHP a ASP) rozdíl v tom, že jeho běh se provádí na straně klienta až po stažení stránky ze serveru. Jeho možnosti jsou proto do jisté míry omezeny s ohledem na bezpečnost dat.

JavaScript sice patří do rodiny jazyků C/C++/Java ale klíčové slovo Java v jeho názvu je pouze marketingový tah. S programovacím jazykem Java jej pojí pouze podobná syntaxe.

Tento skriptovací jazyk bal původně vyvíjen firmou Netscape pod názvem Mocha. Později pak pod názvem LiveScript. Společnost Microsoft jej vyvíjela pod názvem JScript, tato verze byla podporována platformou .NET.

Společně s oznámením o založení společnosti Sun Microsystems v prosinci 1995 byl oznámen jako doplněk k jazykům HTML a Java.

JavaScript se v červenci 1997 dočkal svého standardizování společností ECMA (Europen Computer Manufacturers Association) pod názvem ECMAScript a v srpnu roku 1998 byl vydán standard institucí ISO (International Organization for Standardization). Z tohoto standardu se odvozovali další implementace (př.: ActionScript).

3.4 CSS styly

Myšlenka CSS začala vznikat na začátku 90. Let 20. století, kdy Tim Berners-Lee vyslovil návrh na oddělení formátování od struktury a obsahu stránek.

Až do první standardizace v roce 1996 institutem W3C se CSS styly vyvíjeli zcela neorganizovaně a chaoticky. Tento chaotický vývoj byl zapříčiněn také rozličným vývojem standardů HTML a vývojem internetových prohlížečů. Celá historie CSS v podstatě spočívá na jejich podpoře respektive nepodpoře u prohlížečů. Za otce CSS je označován Hakon Wium Lie, vývojář Opery.

Jako první prohlížeč, který částečně podporoval CSS styly, byl Internet Explorer 3. Ten byl vydán těsně před první standardizací CSS. Prohlížeče Nestscape v původních verzích CSS nepodporovali díky jisté skepsi a navíc protože měli svoji velikou zbraň JavaScript. Nakonec kvůli marketingovému nátlaku se CSS začali podporovat i v Netscape.

Již poměrně slušná podpora CSS byla zabudována do prohlížečů Internet Explorer 4 a Netscape Navigator. To bylo v roce 1998. V tomtéž roce byla přijata nová specifikace CSS 2. I přes novou verzi CSS 2 je v prohlížečích spíše podporována první verze CSS 1.

V roce 1999 se začalo pracovat na další verzi CSS 3. Ta byla dokončována roku 2006 a v nynější době se jen čeká na její podporu v prohlížečích.

3.5 SQL

SQL je standardizovaný jazyk pro práci s daty v relačních databázích. Jeho vznik podmínil výzkum relačních databází v 70. letech ve firmě IBM. Zde bylo nutné vytvořit sadu ovládajících příkazů pro relační databáze. Zároveň byla snaha o vytvoření jazyka, jehož příkazy by se tvořili syntakticky a co nejblíže angličtině. Tak vznikl jazyk SEQUEL (Structured English Query Language).

Na vývoji jazyka se začaly podílet i další firmy. V roce 1979 byla uvedena na trh relační databázová platforma Oracle Database firmou Relational Software, Inc. (dnešní Oracle Corporation). IMB na svou platformu nenechala dlouho čekat a tak v roce 1981vydalo nový systém SQL/DS. V roce 1983 pak systém DB2. V těchto letech byly na trh uvedeny i jiné systémy jako třeba Progres, Informix a SyBase. Až do této doby se ve všech systémech používaly obměněné varianty jazyka SEQUEL, ten byl přejmenován na SQL (Structured Query Language).

Relační databáze se stávali stále významnější a proto bylo nutné provést standardizaci jejich jazyka. Americký institut ANSI chtěl původně jako standard ustanovit zcela nový jazyk RDL. Ten se však neprosadil a byl zamítnut. Institut ANSI uznalo standard založený na jazyku SQL, který se stal *de facto* standardem již předtím. Tento standard bývá označován jako SQL-86 podle roku, kdy byl ustanoven.

Následující léta ukázaly, že SQL-86 obsahuje některé nedostatky. Také v něm nebyly obsaženy některé důležité funkční prvky. Nedostatky se hlavně týkaly integrity databáze. Díky tomu byl v roce 1992 přijat nový standard SQL-92 (v některých zdrojích se uvádí jako SQL2).

V dnešní době je zatím nejnovější verzí SQL-99, která reaguje na potřeby nejmodernějších databází.

3.6 Databáze

Databázi si lze nejjednodušeji představit jako papírovou kartotéku, např. seznam knih a čtenářů v knihovně. Data zde jsou uspořádány podle různých kritérií a lze přidávat nové položky. Veškeré operace s nimi provádí přímo člověk. Přesně tak vypadal předchůdce databází jako takových.

Dnes obsluhu nad daty (informacemi) neprovádí člověk ale tzv. databázový stroj. Jako první velké strojové zpracování dat (informací) je možno považovat sčítání

lidu ve Spojených státech v roce 1890. Paměťovým médiem byl děrný štítek a zpracování sebraných informací probíhalo pomocí elektromechanických strojů. Elektromechanické stroje se využívaly pro účely zpracování dat další půlstoletí.

Dalším velikým impulsem pro vývoj databází byl další vývoj výpočetní techniky v padesátých letech 20. století. V této době se ukázalo, že použití strojového kódu pro zpracování dat je neefektivní a tudíž se začali vyvíjet nové, vyšší jazyky pro zpracování dat. Z popudu amerického ministerstva obrany a zástupců firem o univerzální databázový jazyk byl představen roku 1960 jazyk COBOL. Současně s představením tohoto nového jazyka bylo jasné, že se bude i nadále vyvíjet. I přesto se COBOL stal na dlouhá léta nejrozšířenějším jazykem pro zpracování dat.

S narůstajícími výpočetními kapacitami se rozvíjeli i nové databázové architektury a nové požadavky na databázové systémy. V roce 1965 byl vytvořen výbor pro vytvoření koncepce databázových systémů Database Task Group (DBTG) a zároveň se začínají objevovat první Databáze Management Systém (DBMS) český používaný ekvivalent je Systém Řízení Báze Dat (SŘBD) na sálových počítačích. Database Task Group v roce 1971 vydal správu *The DBTG April 1971 Report* ve, které se poprvé objevily pojmy schéma databáze, databázový model, jazyk pro definici schématu, subschéma a podobně. Zároveň zde byla popsána kompletní architektura síťového databázového systému. Ve stejné době se vyvíjeli různé databázové modely. Tyto modely určují uspořádání dat v databázích a případné vazby mezi nimi. Vzniká hierarchický model, který uspořádává data do stromového modelu, jež umožňuje vyjádřit ve směru shora dolů jednosměrné vztahy typu 1:N. Zdokonalený hierarchický model se nazývá síťový model, který přímo navazuje na hierarchický model. Nejznámější produkt byl IDMS od firmy Culliname Corp. Dále se rozvíjel i relační databázový model, který na data pohlíží jako na tabulky. S těmi posléze pracuje za pomoci základních operací (sjednocení, kartézský součin, rozdíl, selekce, projekce a spojení) a pomocí těchto operací je schopen provést všechny potřebné operace s daty.

Přibližně v roce 1974 se poprvé objevuje databázový jazyk SQL. Původní název byl SEQUEL (Structured English Query Language). Firma IBM při zkoumání relačních databází potřebovala jazyk, který by obsahoval sadu instrukcí pro ovládání těchto databází. Cílem bylo vytvoření příkazů jenž by se tvořili syntakticky a co nejvíce podobné angličtině. K vývoji jazyka se přidaly i další firmy. Tento jazyk se vyvíjel až do dnešní podoby. Jako první byl představen firmou Relational Software, Inc.

(dnešní Oracle Corporation) na platformě Oracle Databáze roku 1979. IBM uvedla v roce 1981 nový systém SQL/DS a v roce 1983 systém DB2. Relační databáze se stávali stále významnější, a proto bylo nutné standardizovat jazyk pro jejich obsluhu. Americký institut ANSI se proto rozhodl pro vydání zcela nového standardu a to jazyk RDL. Místo toho se však SQL ustanovil v podstatě jako standard a ANSI založil nový standard s podporou tohoto jazyka. Tento standard se také označuje jako SQL-86 podle roku, kdy byl přijat.

V nadcházejících letech se ovšem ukázalo, že tento standart je nevyhovující. Objevili se v něm nedostatky a zjistilo se, že v něm nejsou obsaženy některé důležité prvky týkající se integrity databáze. Díky tomu byl v roku 1992 přijat nový standart SQL-92 (někdy se uvádí SQL2). Zatím nejmladším standardem je SQL3 z roku 1999, který reaguje na potřeby moderních databází s objektovými prvky.

Standardy podporuje prakticky každá relační databáze, ale obvykle nejsou implementovány vždy všechny požadavky normy. A naopak, každá z nich obsahuje prvky a konstrukce, které nejsou ve standardech obsaženy. Přenositelnost SQL dotazů mezi jednotlivými databázemi je proto omezená.[3]

V 90. letech 20. století se začínaly objevovat první objektově orientované databáze, jejichž filozofie byla přebírána z objektově orientovaných jazyků. Tyto databáze měly podle předpokladů vytlačit relační systémy. Původní předpoklady se však nenaplnily a vznikla kompromisní objektově-relační technologie.[3]

4. Bezpečnost

Bezpečnost je jednou z kapitol, které je třeba věnovat zvláštní pozornost a nebrat ji na lehkou váhu. Různé stupně zabezpečení se liší podle velikosti projektu a citlivosti dat, se kterými se manipuluje. Osobní stránky pro sdílení fotografií proto nebude mít zabezpečení jako internetové bankovnictví a i opačně je to nemyslitelné.

4.1 Bezpečnostní rizika a způsoby zabezpečení

Vzhledem k tomu, že se z této práci věnuji webové aplikaci, budu se zabývat bezpečností webových aplikací, zabezpečením serveru případně pak komunikace mezi klientem a serverem.

Jaká vlastně jsou bezpečnostní rizika, se kterými se můžeme v nynějších webových aplikacích setkat? Ve světě existuje organizace The Open Web Application Security Project (OWASP), která se již několik let zabývá monitoringem hrozeb pro webové aplikace. Nejznámější výstup této organizace je seznam těch nejvíce častých chyb zabezpečení webů společně s případným řešením daného problému. Ty nejčastější bezpečnostní hrozby pro webové aplikace pro rok 2010 jsou:

- Injection

  Napadení aplikace vsunutím kódu přes neošetřený vstup. Např. SQL, LDAP, OS injection.

- Cross Site Scripting (XSS)

  Narušení webových aplikací vkládáním skriptů či jejich částí do webového prohlížeče.

- Broken Authentication and Session Management

  Napadení funkcí aplikací, které souvisí s autentizací a správou sezení s cílem převzít identitu jiného uživatele.

- Insecure Direct Object References

  Přístupy k neoprávněným datům nezabezpečeným přístupem k vnitřnímu objektu aplikace (souboru, adresáři, databázovému klíči).

- Cross Site Request Forgery (CSRF)

  Útok podvržením požadavku webové aplikace.

- Security Misconfiguration

  Útoky využívající nedostatků v zabezpečení konfigurací aplikačních serverů, webových serverů, databázových serverů, softwarových platforem atd. (např. ponecháním výchozích hodnot, neaktualizová-ní aj.)

- Insecure Cryptographic Storage

  Rizika vyplývající z nedostatečně chráněných citlivých dat (čísla kreditních karet, rodná čísla atd.).

- Failure to Restrict URL Access

  Nedostatečně zabezpečené řízení přístupu ke zdrojům informací (dokument, služba) přes URL.

- Insufficient Transport Layer Protection

Útoky odposlechem sítí.

- Unvalidated Redirects and Forwards

  Útoky přesměrováním na jiné stránky

  [8]

4.2 Použité metody zabezpečení

- Přihlašování do systému a uložení přihlašovacích údajů do globálních proměnných $_SESSION, které jsou vhodně navrženy. Ukázka kontroly přihlášení je na Obrázek 4.1 Ukázka kontroly přihlášení
- Při každém načtení stránek opětovná kontrola přihlášení
- Nastavení práv na serveru, které zamezují přístup a náhled do složek z vně serveru
- Kontrola vstupních dat od uživatelů, snaha používat komponentu *select*, která znemožní uživateli zadat jiná než předdefinovaná data. Zamezení injection
- Hesla jsou uložená v podobě hashe

```
<?php
//kontrola, zda je člověk přihlášen
if (!isset($_SESSION["jmeno"]) or $_SESSION["status"]!="ucitel")
die (include "../neprihlasen.php");
?>
```

Obrázek 4.1 Ukázka kontroly přihlášení

4.3 Možná bezpečnostní rizika

I seberobustnější ochrany proti prolomení bezpečnosti jsou stejně jen ztížení přístupu k datům, které útočník hledá. V případě mého informačního systému si jsem vědom hned několika bezpečnostních rizik. Některé rizika nebylo možno odstranit z závislosti na následné funkčnosti některých funkcí (převážně editace). Jiná rizika jsem při vytváření aplikace do jisté míry podceňoval a proto jim nevěnovat takovou pozornost jakou by si zasloužila.

Možná rizika, která se buď objevila nebo jsem schopen posoudit jako relevantní:

- Jako uživatelská id jsou použita rodná čísla uživatelů. Tyto id jsou viditelná pouze přihlášenému uživateli nicméně jedná se o jisté bezpečnostní riziko
- Po přihlášení se uživatel odhlásí buď zavřením sezení nebo odhlášením. V případě, že neprovede ani jednu operaci, zůstává stále přihlášen
- Při přidávání nových funkcí by se mohla opomenout kontrola autentizace a tím narušit bezpečnost aplikace
- Prozrazení žákovi heslo rodiče. Výsledná zpětná kontrola rodičovského zájmu je zkreslená

5. Postup tvorby

Informační systém jsem vytvářel takzvaně na „zelené louce“. Veškeré zdrojové kódy jsem psal od prvopočátku bez použití frameworku nebo jiných již předpřipravených knihoven. Vycházel jsem tedy z funkcí PHP a ostatních použitých skriptovacích jazyků.

Ještě před samotným zahájením programování bylo však třeba vytvořit jasný obraz toho jak by výsledná aplikace měla vypadat, jaké bude obsahovat funkce a podobně. První kroky vedli do základní školy a konzultace s pověřenou učitelkou o podobě informačního systému. Následně bylo velice důležité zjistit technické zázemí školy, zda má či nemá server, na kterém by byl spuštěn následný provoz systému. Ten škola má, takže jsem se již nemusel dále zabývat outsoursovým zajišťováním hostingu.

Po ujasnění si prvních potřebných informacích jsem se mohl pustit do vytváření datové a funkční struktury.

5.1 Návrh datové struktury

Při vytváření datové struktury jsem vycházel z klasické papírové formy evidence žáků v katalogových listech ve škole a pro učitele jsem zvolil model, kde se vyskytli krom základních i všechny relevantně důležité informace (e-mail, číslo kabinetu, webové stránky, a jiné).

Dále jsem do datového modelu musel zohlednit tvorbu a správu rozvrhů. Ty jsou nakonec realizovány samostatným vytvářením jednotlivých předmětů v tabulce *predmety* se vztahy k učitelům a následně vazbou M:N k žákům, kteří mají tímto způsobem přiděleny rozvrhy. Tato vazba je realizována tabulkou *mezi_predmet*, ve které jsou uloženy údaje o všech rozvrzích.

Struktura známek a výchovných opatření byla zcela zřejmá. Samostatné tabulky s vazbami k učitelům a žákům. V tabulkách jsou uloženy charakterizující údaje o záznamech společně s údaji o rodičovské kontrole.

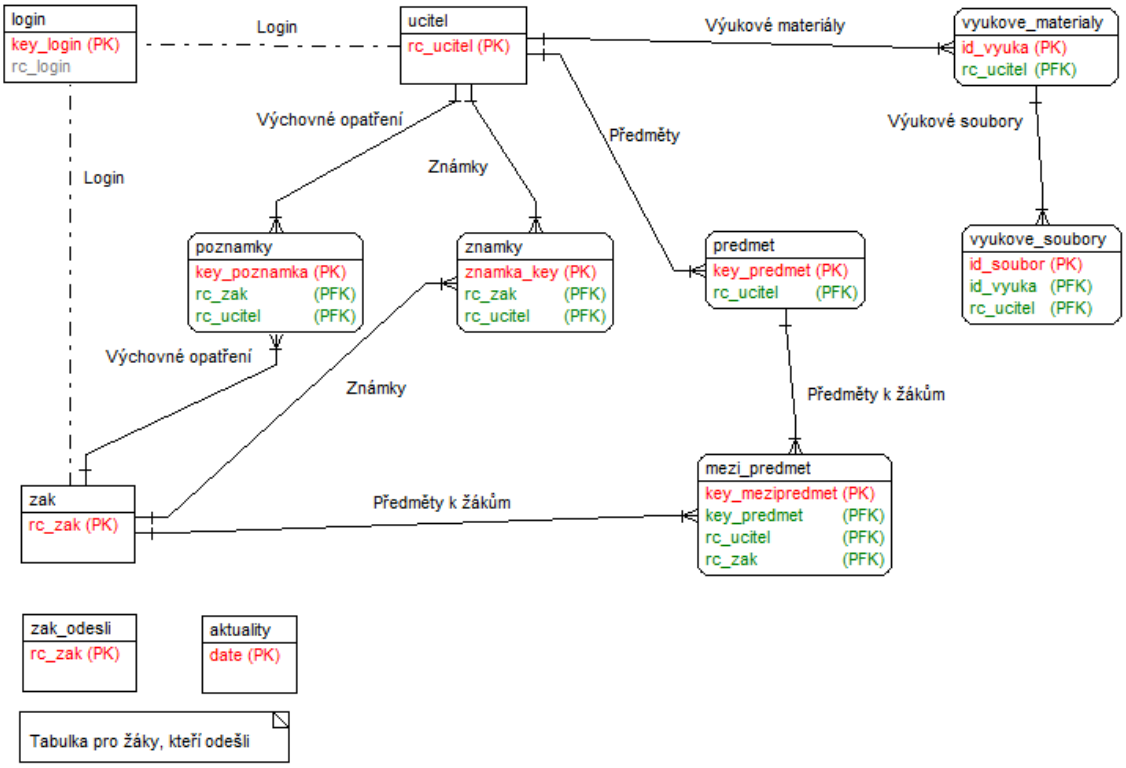
Na žádost školy je do datového modelu přidáno několik tabulek, které se neuvažovali v původním návrhu, ale které rozšiřují aplikaci o další funkce. Jsou to tabulka *login*, která realizuje ukládání přístupů do systému a ze které lze vyhodnocovat

zájem rodičů o studijní výsledky svých dětí, zájem žáků o informace a aktivitu jednotlivých učitelů. Další rozšíření datového modelu je v podobě výukových materiálů, kde se jedná o ukládání tzv. výukových šablon sloužících jako podklady pro výuku. Ukládání a editace je řešeno přes webovou aplikaci žákovské knížky, která zajišťuje autentizaci pověřených osob (učitelů), a publikování souborů je na veřejných webových stránkách školy.

Při přechodu na nový školní rok vyvstala otázka jak naložit s žáky, kteří buď odešli a nebo ukončili studium, aby nepřekáželi v tabulce *zak*. Pro tyto žáky jsem vytvořil tabulku *zak_odesli* shodnou s tabulkou *zak*. Do této nové tabulky se ukládají všichni žáci, kteří opustili školu. Zde jsou informace archivovány.

Celé schéma databázové struktury je vyobrazeno na obrázku Obrázek 5.1 Datový model.

Pro vytváření tohoto datového modelu mi posloužilo vývojové prostředí CASE Studio 2, které je přímo určené k vytváření datových modelů. Zároveň slouží k vytvoření zdrojového kódu, který na zvoleném databázovém enginu vytvoří databázi z připraveného modelu.



Obrázek 5.1 Datový model

Valná většina objektů v tomto databázovém návrhu byla vytvořena ještě před zahájením programování. To ovšem neplatí zcela globálně. Některé části byly přidávány do tohoto návrhu později v závislosti na jejich potřebě při psaní zdrojových kódů.

5.2 Funkční analýza

Před samotným vývojem aplikace bylo třeba vytvořit funkční analýzu, podle které se vytvářeli jednotlivé části aplikace.

Vzhledem k tomu, že se jedná o aplikaci, do které je třeba se přihlásit, je nutné, aby obsahovala formulář pro přihlášení. S přihlášením je spojeno i ověřování zda jsou v pořádku přihlašovací údaje. Zároveň je třeba při přihlášení rozpoznat o jakou osobu se jedná, jaké jí poskytnout prostředí, ve kterém se může pohybovat a jaká práva pro případné používání nastavených funkcí.

Jednotlivé funkční celky by se daly rozdělit do čtyř skupin podle osob a oprávnění, které osoby mají, pro přístup do aplikace. Skupiny jsem rozdělil takto:

- Administrátor
- Učitel
- Rodič
- Žák

Jako nejvyšší možná autorizace je administrátor, který má na starost správu celé aplikace, především vkládání dat a jejich případnou úpravou. Pro elektronickou žákovskou knížku je administrátor schopen vykonávat tyto funkce:

- Vkládání, mazání a editace žáků a učitelů
- Tvoření rozvrhů jednotlivým třídám
- Tvoření individuálních rozvrhů
- Prohlížení si výsledných rozvrhů v zjednodušeném zobrazení
- Vkládání a editaci vyučujících předmětů
- Správu hesel, tj. možnost
- Přechod na nový školní rok

Další pověřenou osobou je učitel, který má rovněž možnost vkládat data (známky a výchovná opatření). Aby elektronická žákovská knížka plnila potřebnou činnost je potřeba aby učiteli poskytovala následné funkce:

- Udělování známek žákům
- Udělování výchovných opatření žákům

Pro třídní učitele pak:

- Počítání zameškaných,omluvených a neomluvených hodin
- Kontrola zájmu rodičů (zda rodiče kontrolují výsledky svých dětí pomocí této aplikace)

A jako nadstavba pro potřeby školy:

- Vkládání výukových materiálů

Po přihlášení jako rodič nebo žák aplikace funguje pouze jako informační kanál. Žádný z těchto uživatelů nemá možnost zadávat ani odesílat žádná data. Pouze rodič má možnost využít tlačítka „Ano viděl jsem“ u známek a výchovných opatření. Pro rodiče a žáky jsou potřeba následující pohledy:

- Zobrazení známek žáka
- Zobrazení výchovných opatření žáka
- Pro rodiče možnost přidat příznak shlédnutí k známkám a výchovným opatřením

Funkce čistě informativního rázu jsou potom:

- Zobrazení rozvrhu žáka, který je přihlášen
- Zobrazení jednotlivých předmětů
- Zobrazení učitelů vyučujících žáka

5.3 Vývoj aplikace

Zde bych chtěl podotknout, že při seznámení se školou a podobou učitelského sboru mi bylo jasné, že aplikace bude muset být značně robustní proti technicky neznalým uživatelům. Učitelský sbor totiž obsahuje veliké množství starších nebo výpočetní technikou nezasažených kantorů. Veškeré funkce jsou proto konstruovány, aby byly co nejjednodušší na užívání. O to více bylo v některých chvílích těžké vytváření funkční logiky stránek a logických postupů pro práci s informačním systémem.

Po dokončení první verze datové analýzy jsem vytvořil na lokálním serveru databázi, která tomuto datovému modelu odpovídala. Databázi jsem si zvolil MySql verze 4. A pro správu této databáze mi přišlo jako nejjednodušší řešení PhpMyAdmin.

Při tvorbě informačního systému, do kterého je nutno se přihlašování mi přišlo jako logické nejprve vytvořit systém přihlašování a přidělování oprávnění. Přihlašování je realizováno pomocí zadání rodného čísla uživatele a hesla. Tyto údaje se posléze odešlou jako dotaz do databáze a výsledkem je buď úspěšné, nebo neúspěšné přihlášení. Úspěšné přihlášení uživatele odkáže na jemu určenou webovou stránku, ze které se

může pohybovat dále. Při neúspěšném přihlášení je uživatel opět odkázán na přihlašovací formulář a vše se opakuje.

Pro pozdější tvorbu jsem toto přihlašování prozatím měl vypnuté. Důvodem bylo rychlejší testování napsaných zdrojových kódů, než abych se musel stále přihlašovat, navíc jsem v té době neměl (nepotřeboval jsem) testovací osoby v databázi.

Jako další krok bylo napsání si zatím nefunkčních menu pro jednotlivé uživatele (administrátor, učitel, žák/rodič). Tyto menu jsem vkládal do zatím prázdných stránek, kam byli odkázáni uživatelé po přihlášení.

Tyto menu obsahovali seznam všech funkcí, které se budou tvořit.

Postupně jsem začal se zprovozňováním jednotlivých funkčních celků. Někdy se práce na jedné funkci neobešla bez dobré funkčnosti funkce jiné. Jindy jsem zase vyvíjel více funkcí najednou.

Práce nad databází je bez dat velice komplikovaná, až bych řekl neproveditelná. Proto další krok po vytvoření přihlášení a jednoduché kostry aplikace bylo vytvoření vkládání uživatelů (učitelů a žáků) pro administrátora.

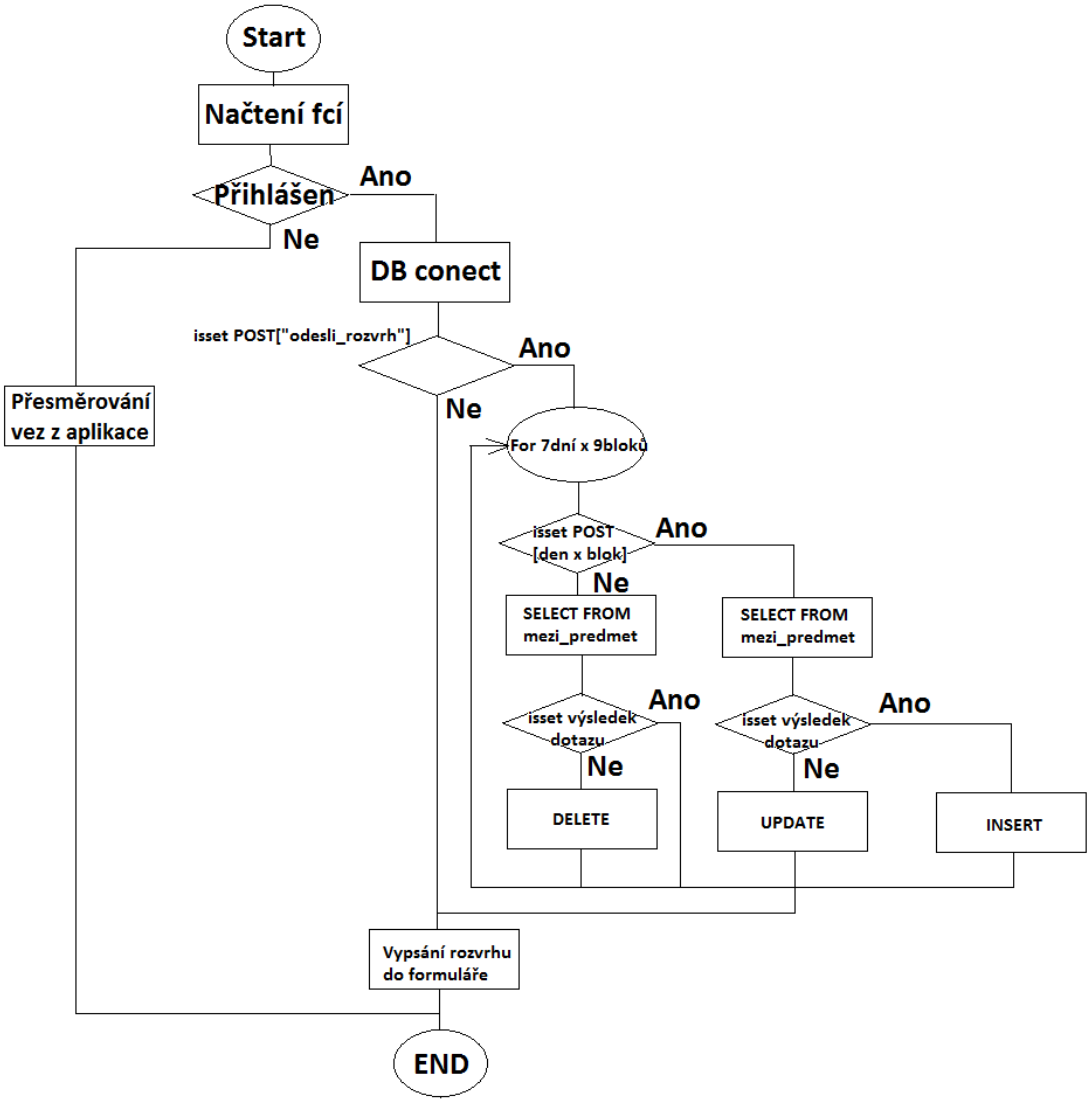
Zde stačilo využití dvou formulářů, které obsahují všechny potřebné položky pro vyplnění. Po odeslání formuláře určeného učiteli nebo žákovi se provede uložení do databáze. Při vkládání nových uživatelů jim je automaticky vygenerováno náhodné heslo. Při neúspěšném zápisu do databáze je vypsáno chybové hlášení s popisem chyby.

Když už jsem měl možnost vložit nějakou osobu, tak bylo zároveň zapotřebí, abych byl schopen těmto osobám měnit zadaná data nebo je byl schopen vymazat celou osobu. Tyto operace může provádět opět jen administrátor. Editace a mazání se prování na samostatných stránkách a to jak pro učitele, tak pro žáka. Stránka je napsána tak, že se nejprve zobrazí rozbalovací nabídka, kde jsou podle abecedy seřazeni buď učitelé nebo žáci. Nabídka je součástí formuláře. Po výběru jména se přes JavaScript formulář odešle a PHP script posléze provede dotaz do databáze a vytáhne informace o daném uživateli. Tyto data jsou následně vypsána do formuláře obsahujícím všechny měnitelné položky.

Po odeslání formuláře potvrzovacím tlačítkem *vložit změny* se ty zapíšou a stránka zaujme svou výchozí pozici. Při stisku tlačítka *smazat žáka* je žák smazán a stránka se taktéž vrací do zobrazení rolovacího menu jako na začátku.

Tvorbu rozvrhů jsem vytvářel na několik etap. Nejprve jsem si musel řádně vyjasnit jak celý rozvrhový systém vytvořit a potom také jak bude pracovat výsledná stránka. Chod a vyhodnocování stránky v jednoduché podobě je na Obrázek 5.2 Logický strom tvorby rozvrhů.

Tvoření rozvrhů probíhá přiřazením jednotlivých předmětů jednotlivým žákům. To znamená, že každému žákovi se tvoří jeho vlastní rozvrh. Ovšem při tvorbě skupinových rozvrhů jsou tyto rozvrhy nahrávány všem žákům v dané skupině nebo třídě. Proto je důležité před vytvářením rozvrhu pro třídy vybrat danou třídu, pro kterou se bude rozvrh tvořit.



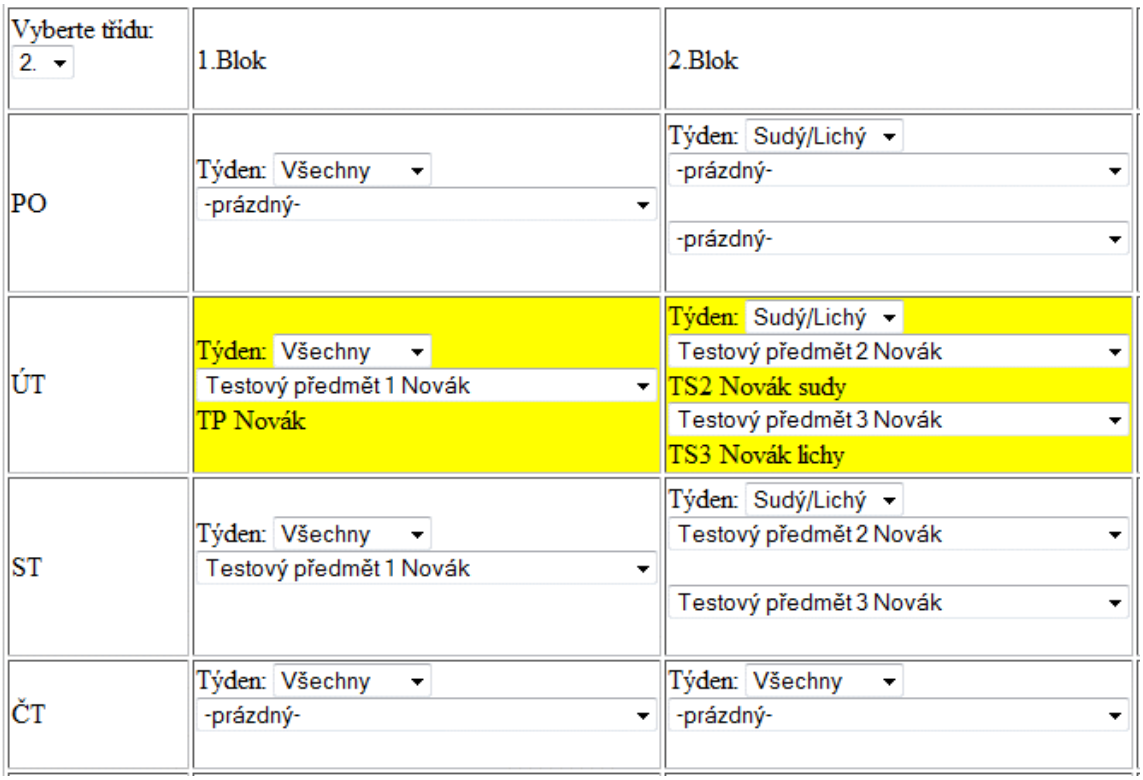
Obrázek 5.2 Logický strom tvorby rozvrhů

Tato stránka je dle mého mínění jednou z nejsložitějších v celém informačním systému. Jak jsem již zmínil, její vývoj byl veden v několika etapách. Po vytvoření si jasné představy, jak by stránka měla fungovat, jsem vytvořil jednoduchý formulář. Ten obsahoval rozbalovací menu se seznamem předmětů v každém bloku rozvrhu (5 dní x 9 bloků). Tento formulář odesílal předměty do databáze do tabulky *mezi_predmet*. Další komponentou, kterou tato stránka obsahovala již od začátku, bylo rozbalovací menu s možností výběru třídy, pro kterou se rozvrh tvoří.

To bylo v tuto chvíli vše. Odladil jsem kód a přešel k dalšímu kroku a tím bylo zadávání, zda je předmět vyučován v lichý, sudý nebo v každém týdnu. Tento krok již byl složitější díky složitosti kontroly, kterou jsem musel provádět před dotazy do databáze.

Jako třetí etapa zbylo vtvoření rozvrhů tak, aby bylo možné do jednoho bloku zadat dva různé předměty. Jeden pro sudý vyučovací týden a druhý předmět pro lichý vyučovací týden. Tato úprava mi zabrala asi nejvíce času, protože jsem musel zohlednit dva zápisy do jednoho bloku, při mazání jsem musel vytvořit vyhodnocování, o jaký týden se jedná (sudý/lichý) a také bylo nutno zohlednit přechod ze zadávání sudý/lichý na oba týdny zároveň. Na stránce je vytvořeno několik operací pomocí JavaScriptu, které se starají o dynamičtější chod stránky. Jedním z příkladů je aktualizace stránky po změně rozbalovacího menu, kde jsou jako možnosti právě sudý/lichý týden nebo oba týdny zároveň.

Na Obrázek 5.3 Ukázka tvorby rozvrhů je zobrazen výřez jakým způsobem uživatel (administrátor) vidí výsledný formulář. V prvním řádku jsou vidět dva bloky, kde v jednom je předmět s výukou ve všechny týdny a v druhém bloku jsou vidět dvě rozbalovací lišty pro výběr předmětu jak pro sudý tak lichý týden zvlášť. Žlutě podbarvené bloky již obsahují předmět/y. Ty se dají samozřejmě měnit. Je zde i možnost změna volby z sudý/lichý na všechny a obráceně. Vyhodnocovací algoritmy si samy poradí se všemi kombinacemi situací, které mohou nastat.



Obrázek 5.3 Ukázka tvorby rozvrhů

Po dokončení stránky pro tvorbu kolektivních rozvrhů bylo ještě třeba vytvořit jednu této stránce podobnou. Ta by měla obsahovat do jisté míry stejné funkce, až na rozdíl, že se rozvrh ukládá pouze individuálně vybranému žáku. To nebyl takový problém, stačilo pouze vedle výběru tříd udělat ještě jedno rozbalovací menu, které obsahovalo žáky z vybrané třídy. Po vybrání konkrétního žáka je možné tvořit rozvrh přímo tomuto žákovi. V kódu se provede dotaz pouze na tohoto žáka a pro ostatní je jeho rozvrh nezobrazitelný.

V pozdějším vývoji aplikace se ukázalo, že zadávání některých individuálních rozvrhů po jednom je zdlouhavé a uživatelsky nepohodlné. Proto jsem vytvořil další nástavbu při tvoření individuálních rozvrhů v podobě seznamu žáků vybrané třídy s check boxy. Při zaškrtnutí těchto check boxů se rozvrhy uloží všem žákům, kteří byli označeni.

Po dokončení většiny funkcí v administrátorském rozhraní jsem se vrhnul na kompletování rozhraní pro učitele.

První krok nebyl tolik komplikovaný, protože jako první funkci jsem vytvořil výpis rozvrhu učitele. To po vyladění tvorby rozvrhů bylo již snadné. Jedná se o pohled do databáze, který je specifikován identifikačními údaji přihlášeného učitele.

Hlavní funkce tohoto informačního systému je informovat o studijním prospěchu žáků. K tomu ovšem učitelé musejí mít možnost tyto známky zadat. To je tedy jedna z funkcí, které jsem se v tuto chvíli začal věnovat.

Z datové analýzy mi bilo jasné, že při zadávání známek jsou k uložení potřebné tyto informace:

- hodnota známky
- kdo známku zadal
- z jakého předmětu je známka
- kdy byla známka udělena
- popis za co byla známka udělena

Pro tyto účely jsem vytvořil formulář, který obsahuje kolonky pro všechny tyto atributy až na zadání učitele a data. Tyto dva údaje se berou automaticky. Jméno učitele je do databáze uloženo v podobě id učitele a je bráno z jeho přihlašovacích údajů. Datum je generováno automaticky pomocí příkazu NOW() při předávání SQL dotazu databázi. Zadávání hodnoty známky je realizováno pomocí rolovacího menu. Důvodem užití tohoto způsobu vybírání známky je kontrola vstupu dat. Hodnota známky musí být realizována pouze číslem.

Vybírání žáků, kterým se budou zadávat známky, je realizováno pomocí rozbalovací nabídky se seznamem tříd. Po výběru požadované třídy se zobrazí seznam žáků. U každého jména jsou komponenty formuláře pro zadání hodnoty známky a popis známky. Z jakého předmětu je známka udělena je umístěno nad seznamem žáků. Není třeba ke každému žákovi zadávat zvlášť předmět.

Další ulehčení učiteli je vytvoření jakéhosi centrálního popisu známek. Ten jsem vytvořil také nad seznamem žáků a za jeho pomoci lze udělat jeden popis ke všem žákům a známkám pro jednotlivé zadání. Učitel nemusí vyplňovat např.: „Opakovací písemná práce“ do každého okénka nýbrž jen do tohoto jednoho. Stránka je nastavena tak, že při vyplnění centrálního popisku a zároveň některého konkrétního pole u žáka, tak je žákovi zapsán konkrétní popisek. Všichni ostatní ohodnocení žáci mají popisek centrální.

Jelikož uživatel je jen člověk a lidé jsou omylní, bylo potřeba vytvořit možnost zadané známky pozměnit nebo smazat.

K tomuto účelu jsem na spodu stránky vytvořil další rozbalovací seznam žáků. Zde je možno vybrat žáka z právě vybrané třídy a kliknutím na tlačítko *Opravit známky* se zobrazí seznam známek. Vše je realizováno odesláním formuláře, kde je id vybraného žáka. Zobrazené známky pocházejí pouze od učitele, který je zrovna přihlášen. Učitel má možnost změnit hodnotu známky, popisek a nebo známku vymazat.

Podobně jako u zadávání známky funguje i zadávání výchovných opatření jen tím rozdílem, že výchovná opatření se zadávají jednotlivcům. Tudíž učitel má k dispozici dvě rozbalovací menu. V jednom navolí třídu, toto menu se odešle pomocí JavaScriptu a do druhého vysouvacího políčka se rovnou zapíší všichni žáci z dané třídy. K dispozici je textové pole, do kterého je možné zapsat výchovné opatření.

Ostatní údaje jako jsou datum a kdo opatření udělil jsou k odeslání dotazu do databáze vyplněny automaticky. Stejně jako u známek.

Pro třídní učitele jsem připravil další dvě funkce. Jedná se o počítání zameškaných, omluvených a neomluvených hodin a kontrolu zájmu rodičů.

V případě zameškaných hodin se jedná o ukládání celého čísla do řádku v tabulce, kde je uložen zápis žáka. Na stránce jsou k dispozici tři textová pole a jeden *select* , ve kterém jsou vypsány všechny žáci třídy, kde je učitel třídní. Vše je uloženo do jednoho formuláře.

Zpracování tohoto formuláře se provádí odesláním SQL požadavku do tabulky *zak* na položky zameškaných, omluvených a neomluvených hodin.Tyto hodnoty jsou číselné a proto se dají sečíst s hodnotami odeslanými přes formulář. Následně se aktualizuje záznam v databázi.

Realizace rodičovské kontroly je prováděna dvěma dotazy do databáze a výpisem tabulky přes while cyklus. Dotazy jsou směřovány nejprve na tabulku *zak*, kde je výsledek seznam žáků třídy, kde je učitel třídním. Druhý dotaz se potom provádí přímo při vypisování tabulky. Ten je veden na tabulku *login*. Odtud je vytažen poslední zápis z přihlášení rodiče žáka, u kterého je zrovna while cyklus. Do tabulky se vypíše čas a datum posledního přihlášení rodiče.

Až do této doby jsem neměl udělány téměř žádné funkce pro žáka. Ten je ve své podstatě nejníže na žebříčku práv. Jako jediný nemá žádné právo zápisu nebo jiných editačních možností.

Jednoduché komponenty jsem vytvořil jako první a posléze se prací přesouval k těm složitějším. Jako jednoduché jsem považoval zobrazování rozvrhu žáka. To je ve své podstatě stejné jako u učitele, pouze s drobnými změnami v SQL dotazu, kde jako klíčové byla informace o žákovi. V pozdějších grafických úpravách byl každému učiteli přidán atribut *barva_ucitel*, kde je uložena nějaká barva. Ta potom vyplňuje pozadí buněk rozvrhu.

Další ne příliš složitá funkce bylo zobrazení všech předmětů a učitelů, které žák má. Funkce pracují opět na dotazu do rozvrhové tabulky. V těchto funkcích již bylo zapotřebí kontrolovat aby nebyl jeden předmět nebo učitel vypsán dvakrát. To jsem řešil

přes ukládání vypisovaných řádků do pole. Toto pole jsem vždy před výpisem zkontroloval a na základě porovnání, zda v něm zápis je nebo není, řádek do tabulky vypsal.

Jako hlavní funkcí je ovšem zobrazování známek. Zde jsou známky vypisovány od nejnovějších po ty nejstarší. Vše je realizováno dotazem do tabulky *znamky* a výpisem pomocí cyklu while.

Aby stránka nebyla po zadání většího počtu stránek zbytečně dlouhá, použil

```
$obj_znamky=mysqli_query($db_spojeni, "SELECT   rc_zak,
                                                jmeno_predmet,
                                                jmeno_ucitel,
                                                znamka,
                                                datum_znamka,
                                                poznamka_znamka,
                                                precteno_znamka
                                                FROM znamky
                                                WHERE rc_zak='".$_SESSION["user"]."'
                                                ORDER BY datum_znamka DESC LIMIT ".$od.",".$radku."");
```

jsem stránkování seznamu známek. K tomuto stránkování jsem se nechal inspirovat na webových stránkách http://www.linuxsoft.cz/. Jedná se o zaslání dotazu, ze kterého zjistím, kolik záznamů daný žák má a podle toho je vytvořen seznam odkazů. Po kliknutí na odkaz se přes metodu GET odešle číslo listu seznamu, na který se chceme podívat. Konkrétní list se vypíše z dotazu SQL, jenž obsahuje limity. SQL dotaz je ukázán na Obrázek 5.4 SQL dotaz na stránkový výpis známek.

Obrázek 5.4 SQL dotaz na stránkový výpis známek

Jak je vidět tak stránkování je zajištěno SQL příkazem LIMIT, kde první hodnota určuje, od kolikátého záznamu dotaz vrátí výsledek a druhá hodnota říká kolik řádků bude do výsledku posláno.

Při přihlášení jako rodič je do tabulky s výpisem známek přidán navíc jeden sloupec, do kterého se vypisuje, zda známka byla nebo nevyla již shlédnuta. Tento příznak rodič může odebrat pomocí tlačítka *Potvrdit přečtení*.

Stejný postup je zvolen u vypisování výchovných opatření. Jediný rozdíl je v nepoužití stránkování. Na této stránce jsem to považoval za zbytečné. Je zde tedy

dotaz do tabulky *poznamky* a následný výpis všech potřebných atributů. Pro rodiče je zde opět připraven sloupec navíc, kde je zapsán příznak shlédnutí a tlačítko pro odebrání tohoto příznaku.

V tuto chvíli jsem měl hotové všechny potřebné funkce pro chod informačního systému. Jak jsem se již zmínil, měl jsem vypnuté přihlašování. To jsem v této etapě vývoje opět zapnul a ověřil si jeho funkčnost.

S přihlašováním je spojeno i užívání hesel. Ty občas uživatel zapomene. Proto jsem pro administrátora vytvořil funkci, která dokáže řešit problém zapomenutého hesla.

Jedná se o stránku na které je seznam všech žáků, kteří jsou v databázi. Vedle každého jména je tlačítko s funkcí smazání stávajícího hesla a jeho nahrazením náhodně vygenerovaného řetězce písmen a číslic.

Žák má posléze možnost si své heslo změnit. A to na úvodní straně svého účtu. Tuto možnost mají i učitelé. Jedná se o přesměrování na jednoduchou stránku, kde jsou tři textová pole k vyplnění. Do prvního je třeba zadat staré nebo vygenerované heslo a do zbylých dvou zadat heslo nové. Script vyhodnotí shodu starého zadaného hesla s heslem uloženým v databázi a shodu dvakrát zadaného nového hesla. Pokud je vše v pořádku, tak bude proveden příkaz UPDATE v databázi a staré heslo se nahradí novým.

Jednou za rok je potřeba funkce která převede celou databázi, respektive záznamy, na nový školní rok. Jedná se především o postup žáků do vyšších tříd. Zároveň aby žáci, kteří již byli ve třídě nejvyšší, byli z databáze přesunuti do archivačního úložiště.

K těmto účelům jsem vytvořil funkci nového školního roku. Stránka s tlačítkem převádějící databázi na nový rok je přístupná pouze administrátorovi.

Převod je proveden v následujících krocích:

- je proveden dotaz na všechny žáky, které databáze obsahuje
- provede se zvětšení třídy o jedna
- v případě, že třída již nelze zvednout, nastaví se příznak „pryc“

- všichni žáci s příznakem „pryc“ se vymaží z tabulky *zak* a přesunou se do tabulky *zak_odesli*

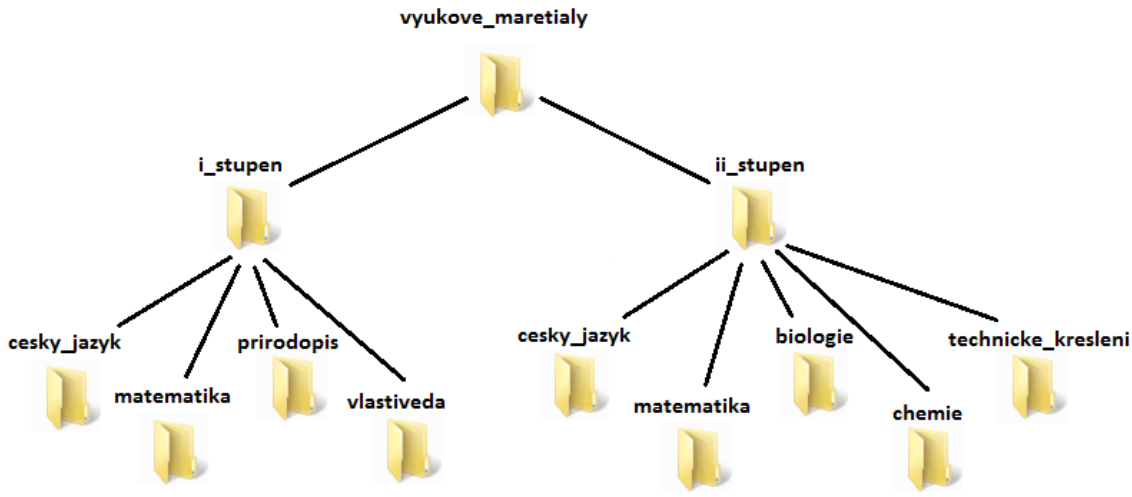
Další úpravy informačního systému jako jsou mazání známek, výchovných opatření, rozvrhů a předmětů je prováděno ručně přes databázi. Tyto funkce jsem prozatím nevytvářel, protože zatím není zcela jasné jaké záměry s těmito daty škola bude mít.

Jako poslední funkci jsem vytvářel ukládání výukových materiálů na server základní školy. Tato funkce vznikla na základě žádosti základní školy a je umístěna pro užití učitelům. Zároveň s nahráním souborů je uložen záznam do databáze do tabulek *vyukove_materialy* a *vyukove_soubory*. Záznam ukládající se do první tabulky obsahuje název tématu, anotace k materiálům,stupeň a předmět, pro který jsou materiály určeny, id nahrávajícího učitele, čas a datum nahrání souboru. Do druhé tabulky je uložen seznam vložených souborů s referencí na předmět, ke kterému patří.

Celé výukové materiály jsou rozděleny na dvě sekce. V jedné je umístěn formulář pro vkládání materiálů a v druhé je seznam vložených materiálů s referencemi na jejich editaci.

Vzhledem k tomu,že jsem tuto funkci vytvářel až na závěr, tak je zde využito několik nových postupů. První je oddělení funkční části skriptu od zobrazovací části stránky. Druhý nový postup je využití vícera dynamických možností JavaScriptu a další novinka je vytvoření funkční části jako soubor knihoven funkcí, které se posléze volají.

Pro ukládání souborů je na serveru vytvořena složka určená pro výukové materiály obsahující další větvení viz Obrázek 5.5 Schéma složky vyukove_materialy. Zde se jednotlivé soubory ukládají podle toho, pro jaký předmět jsou určeny.



Obrázek 5.5 Schéma složky vyukove_materialy

Vkládání souborů do systému je tedy realizováno jednou funkcí, která je spuštěna při odeslání formuláře tlačítkem *Odešli*. Ve formuláři jsou až na datum, čas a identifikační údaje učitele textová pole k vyplnění všech potřebných atributů. Těmi jsou popis tématu, anotace k souborům, stupeň a předmět.

Pro vkládání souborů je použita HTML formulářová komponenta <input> typu file. Počet nahrávaných souborů je řešen dynamicky. Jakmile je vyplněn jeden soubor pro vložení, automaticky se pomocí JavaScriptu vloží do tabulky řádek, ve kterém je nový <input> pro vložení dalšího souboru. Těmto novým vstupním komponentám se přiřazují názvy ve formátu „file“+číslo, které následuje v řadě před předchozím. Takto pojmenované <input> komponenty odešlou soubory na server, zde je možné k nim přistupovat pomocí globálních proměnných POST.

Vyhodnocovací algoritmus funkce pro vkládání nejprve provede kontrolu, zda v dané složce již není soubor shodného jména. V případě, že složka soubor stejného jména již obsahuje, je nahrávání souboru přerušeno. Jestliže se jednalo o jediný soubor k nahrání, neprovede se ani zapsání do databáze. Pokud soubor stejného jména neexistuje potom se nejprve provede jeho nahrání na server. Toto nahrání representuje funkce PHP move_uploaded_file(). Až po úspěšném nahrání souboru je proveden zápis do databáze.

5.4 Testování na lokální úrovni

Při testování na lokální úrovni jsem do databáze vložil malé množství fiktivních dat. Jednalo se o několik žáků a přibližně dva učitele. Bohužel mě v té době nenapadlo použití jakéhokoliv beta testera a tak jsem celé testování prováděl výhradně sám.

Vkládání žáků a učitelů již proběhlo přes administrátorské rozhraní, čímž zároveň proběhlo testování zadávání učitelů a žáků. Následně jsem zkontroloval i správnou funkčnost editace a mazání uživatelů. Pro některé úkony jsem ovšem musel používat přímo zápisy do databáze.

Další testování administrátorského rozhraní byla kontrola zadávání předmětů a tvorby rozvrhů. To proběhlo v pořádku. Všem uživatelům, kteří byly v databázi se rozvrhy zapisovali správně.

Hlavní testování bylo, zda učitelé mohou zadávat známky a výchovná opatření. Následně bylo potřeba otestovat, zda tyto známky a výchovná opatření jsou zobrazovány ve správném formátu a se všemi potřebnými údaji. Zobrazování známek a výchovných opatření bylo také provedeno na úrovni přihlášení rodiče.

5.5 Oprava nalezených chyb

Vyskytlé chyby z lokálního testování se hlavně týkaly sladění kódování databáze a znakových sad použitých ve webových stránkách. Tyto chyby jsem nakonec odstranil důsledným překontrolováním kompatibility jednotlivých kódování.

6. Implementace do reálného prostředí

V této fázi jsem již měl na lokálním serveru ozkoušenou celou aplikaci, která jen čekala na nahrání na server ZŠ a MŠ Višňová. Bylo mi jasné, že na lokálním serveru jsem nemohl testováním v jednom člověku nalézt všechny chyby. Zároveň budou zcela nevyhnutelné kosmetické úpravy prostředí, které je pro mě při vývoji vyhovující ale pro užívání učiteli a žáky nikoliv.

6.1 Samotná implementace a zkušební provoz

Pro nahrání na ostrý server školy byl zapotřebí mít na serveru školy nainstalovaný webový server s podporou PHP a nainstalovanou databázi (MySql 4). Dále bylo nutné mít přístup k webovému prostoru, na který odkazuje webová adresa školy (www.zsvisnova.cz). Na tento prostor mi byl přidělen přístup pomocí webového správce OpenWebMail. Po obdržení všech přístupových práv a po nahrání pluginu podporující PHP jsem nahrál databázi a soubory informačního systému na web školy.

Na stávajících stránkách školy jsem vytvořil odkaz na informační systém a začal s testováním funkcí online.

Jako první byly testy provedeny na fiktivních osobách. Posléze jsem začal nahrávat do databáze všechny žáky z katalogových listů a učitele ze seznamu vyučujících.

Po zadání všech uživatelů byl systém připraven na ozkoušení učiteli a žáky. Prvnímu připojení učitele předcházela prezentace o informačním systému vedená přímo učitelům. Zde se sešel celý učitelský sbor, aby se seznámilo s novinkou, kterou budou používat. Následovalo praktické vyzkoušení si systému. Při tomto zkušebním odpoledni jsem odhalil nebo mi bylo doporučeno několik věcí k vylepšení například: seznamy žáků řadit abecedně podle příjmení, zpřehlednění zadávání známek, odstranění prokliků mezi úkony a jiné.

Již náročnější byl první nápor žáků. Hned v prvních dnech jsem byl nemile překvapen narušením bezpečnosti jedním z žáků. Ten využil opomenutého nastavení práv na složkách webu. Tento nedostatek jsem okamžitě opravil.

Další závady nebo připomínky se týkaly řazení známek, problémů se zapomenutými nebo nevyplněnými hesly. Ze začátku vyvstal i problém se zobrazováním rozvrhů žákům. Ten byl způsoben chybným přiřazováním hodnot pomocných proměnných o 1.

6.2 Oprava nalezených chyb a přizpůsobení uživatelům

Jak jsem uved výše, opravoval jsem chybu zabezpečení nastavením správných přístupových práv pro složky a soubory systému nahraného na server. Tyto práva jsou uložena jako reprezentace čtyř číslic v rozsahu od 0 až po 7 (tj. tři bity). Na stávajícím informačním systému jsou nastavena práva na hodnotu 0755.

Pro přizpůsobení uživatelům jsem se snažil eliminovat zbytečné klikání, které jsem spíše realizoval automatickým přesměrováním nebo jsem využil JavaScriptu a reaguji akcemi na změny komponent a podobně.

Po delší době se vyskytnul další problém, který byl spíše problémem návrhu než provedení. Na chybu se přišlo při vkládání nových žáků do systému. Těmto žákům nebyl přidělen žádný rozvrh.

Tento problém jsem nakonec vyřešil přidáním kořenových žáků se jmény root do každé třídy a při tvorbě rozvrhů jim taktéž připisuji rozvrh. Následně při vložení nových žáků se jim nahraje rozvrh právě těchto root žáků podle třídy. Tyto root žáci se také používají při tvorbě individuálních rozvrhů k porovnání, jaké předměty jsou individuální a jaké jsou pro všechny žáky stejné.

S nasazením těchto žáků byly spjaty i další úpravy zdrojových kódů. Na všech místech, kde se pracuje s výpisem žáků ze tříd, byl nutný dodat vyhodnocovací řádek, který neprovádí vypisování, ukládání a jiné operace s žáky root.

S těmito kořenovými žáky se vyskytl ještě jeden problém v podobě jejich „rodného“ čísla. To mělo na začátku tvar jedna nula+číslo třídy. Při zápisu žáků narozených v roce 2000 ti měli na začátku rodného čísla 00. Ve většině algoritmů se počítalo s tím, že root žáci budou jako první řádek výsledku z databáze. Žáci narození v roce 2000 díky většímu počtu nul root žáky předběhli a tak způsobovali chyby.

Chybu jsem odstranil přidáním čtyřmi dalšími nulami na začátek „rodného“ čísla root žákům.

6.3 Průběžná údržba

I po nahrání aplikace na server školy a po odladění provozu se stále o tento systém starám. Provádím pravidelné zálohy databáze. Starám se o řešení případných problémů. Celkově zůstávám v kontaktu se základní školou.

Průběžně se snažím dojíždět do Višňové a konzultovat s učiteli jejich případná přání nebo návrhy na vylepšení aplikace. Jako poslední přání bylo vytvoření zobrazování známek u žáků po předmětech namísto stávajícího zobrazování chronologicky, jak byly zadány. Další plánovanou změnou bude přidělání nové funkce pro učitele, pomocí které budou schopni měnit si své údaje (telefonní číslo, e-mail, ...).

Stále se také vyskytují chyby spojené především s užíváním hesel. Tyto drobné chyby opravuji drobnými úpravami v databázi. Většinou je jedná o problém s přihlášením, kde uživateli nejde přihlásit se. Problém je buď v zapomenutí hesla nebo ve špatně zadaném hesle. Tyto problémy by pro žáky mohl pohodlně řešit administrátor. K učitelským heslům jsem administrátorovi nedal přístup a proto je správa hesel učitelů zatím na mě.

7. Závěr

Při vytváření tohoto informačního systému jsem měl za úkol se seznámit s postupy tvorby webových aplikací a zároveň s technologiemi, které se k tomu používají. Začátek mé tvorby byl značně žákovský. S většinou programovacích jazyků jsem se musel nejprve seznámit, než jsem je mohl použít k vytváření aplikace. První zdrojové kódy tedy vypadaly amatérsky, ale postupem času jsem pronikal hlouběji do možností jednotlivých jazyků a finální podobě zdrojových kódů již nechybí snaha o elegantní řešení.

Prošel jsem si celým vývojem aplikace od vytváření prvního datového modelu, přes vytváření kostry webové aplikace až po dolaďování uživatelsky příjemných funkcí. Při tvorbě jsem se setkal i problémy na straně mé neznalosti nebo i technické neproveditelnosti vymyšlených funkcí. Oba dva jsem nakonec vyřešil konzultací se zkušenějšími kolegy nebo zkoušením jiných variant návrhu.

Výsledkem tedy je fungující informační systém, který je testován ne jen v laboratorních podmínkách, ale především v reálném prostředí, kam je určen. Právě toto umístění do reálného prostředí mi ukázalo, jak je v některých případech komplikované vytvářet aplikaci a muset jí „vidět“ očima koncového uživatele. Pro mě jakožto tvůrce systému je fungování jednotlivých komponent zcela automatická. U lidí, kteří systém používají, tomu tak být nemusí.

Informační systém má, krom teoretického přínosu, i přínos do prostředí modelové školy. Ne málo žáků a rodičů se naučilo tento systém používat a čerpat z něj informace o prospěchu. Jedním z přínosů takovéhoto informačního systému je přímé informování rodičů, žák nemá jakým způsobem ovlivnit informace jdoucí k rodiči, jak to je u klasických papírových žákovských knížek.

Na aplikaci ještě zůstává prostor pro vylepšování stávajícího a dalšímu vývoji nového. Jsem tedy přesvědčen, že se bude dále vyvíjet a bude sloužit k účelům, ke kterým byla vytvořena, i do budoucna.

8. Seznam použité literatury a zdrojů

[1] Molnár Z., Moderní metody řízení informačních systémů, Grada a.s., 1992, ISBN 80-85623-07-2

[2] Ponkrác M., PHP a MySQL bez předchozích znalostí, Computer Press a.s., 2007, první vydání, ISBN 978-80-251-1758-3

[3] http://cs.wikipedia.org

[4] Cinkly T., Begg C., Holowczak R., Mistrovství – Databáze, Profesionální průvodce tvorbou efektivních databází, Computer Press a.s., 2009, první vydání, ISBN 978-80-251-2328-7

[5] Williams H. E., Lane D., Programujeme webové aplikace pomocí PHP a MySQL, Computer Press a.s., 2002, ISBN 80-7226-760-4

[6] http://htmlguru.cz/

[7] http://www.jakpsatweb.cz/

[8] http://www.root.cz/

[9] http://www.linuxsoft.cz/

[10] http://www.php.net/

[11] http://www.w3schools.com/