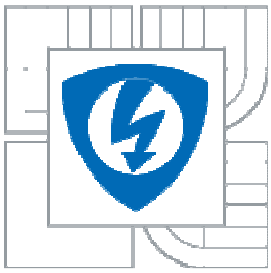
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

WEBOVÁ APLIKACE PRO SPRÁVU UCEBNY

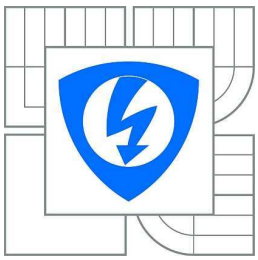
WEB APPLICATION FOR CLASSROOM MANAGEMENT

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — MARTIN ONDREJECH
AUTHOR

VEDOUCÍ PRÁCE — Ing. ONDŘEJ KRAJSA
SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Martin Ondrejech	***ID:***	70243
Ročník:	3	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Webová aplikace pro správu učebny

POKYNY PRO VYPRACOVÁNÍ:

Podrobně se seznamte s možnostmi multiplatformního programování pro Internet. Ve Vámi vybraném vývojovém nástroji navrhněte a realizujte aplikaci pro správu učebny a telefoních přístrojů. Ve svém návrhu upřednostněte její možnou implementaci do webového prostředí a zabezpečte, aby bylo možné tyto přístroje jednoduchým způsobem katalogizovat. Zabezpečte tuto aplikaci proti vniknutí neautorizovanou osobou.

DOPORUČENÁ LITERATURA:

[1] KOSEK, Jiří. PHP :Tvorba interaktivních internetových aplikací. Podrobný průvodce /Praha :Grada Publishing,1998. 1. vyd. 490 s. ISBN 8071693731
[2] NARAMORE, Elizabeth. PHP5, MySQL, Apache: vytváříme webové aplikace / Vyd. 1. Brno : Computer Press, 2006. 813 s. ISBN 80-251-1073-7
[3] KOFLER, Michael; ÖGGL, Berndt .PHP 5 a MySQL 5 : průvodce webového programátora, Vyd. 1. Brno : Computer Press, 2007. 607 s., ISBN: 978-80-251-1813-9

Termín zadání:	29.1.2010	***Termín odevzdání:***	2.6.2010

Vedoucí práce: Ing. Ondřej Krajsa

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Klíčová slova

PHP, MySQL, HTML, CSS, Apache, webový server, konfigurace IP telefonu

Keywords

PHP, MySQL, HTML, CSS, Apache, web server, IP telephone configuration

Abstrakt

Tato práce je zaměřena na možnosti multiplatformního programování pro internet. Podrobněji je tu rozebíráno využití programovacího jazyka PHP a databáze MySQL, jakož i praktická ukázka v podobě webové aplikace pro správu učebny a modulu pro konfiguraci VoIP telefonů Well 3130IF a Well 3195IF.

Abstract

This work is focused on the possibility of multiplatform programming for the Internet. Detailed usage of the programming language PHP and the MySQL database, including a practical example Web application for managing classrooms and module for configuration of VoIP telephones Well 3130IF and Well 3195IF.

Bibliografická citace

ONDREJECH, M. Webová aplikace pro správu učebny. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 39 s. Vedoucí bakalářské práce Ing. Ondřej Krajsa.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma „Webová aplikace pro správu učebny“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení §152 trestního zákona č. 140/1961 Sb.

V Brně dne 2.6.2010 …….

podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Ondřeji Krajsovi za velmi užitečnou metodickou pomoc a cenné rady při zpracování bakalářské práce.

V Brně dne 2.6.2010 .....

podpis autora

OBSAH

ÚVOD

Tento dokument pojednává o možnostech programování pro internet, speciálně o vývoji webové aplikace pro správu učebny. V souladu se zadáním, které znělo podrobně se seznámit s možnostmi multiplatformního programování pro internet a navrhnutí a realizace webové aplikace pro správu učebny, byla vytvořena aplikace rozdělená do několika částí.

Část pro administraci se stará o základní funkce aplikace, jako jsou správa uživatelů, předmětů a zásuvných modulů. Učitelský modul potom obsluhuje rozšíření těchto základních funkcí, přidává správu vlastních vyučování v rozvrhu a možnosti pro hodnocení.

V první části práce jsou uvedeny možnosti nacházející se v současné době na trhu, a podrobněji popsány ty vybrané, se kterými se pracuje v tomto projektu. PHP pro programování na straně serveru, MySQL jako databáze pro uložení dat, HTML jako výstup pro prohlížeč a CSS pro přidání grafického zpracování prostého HTML.

V druhé části je uveden samotný postup při vývoji webové aplikace, popsány důležité části zdrojového kódu, použité techniky a diskutovány problémy vyskytnuvší se v průběhu vývoje.

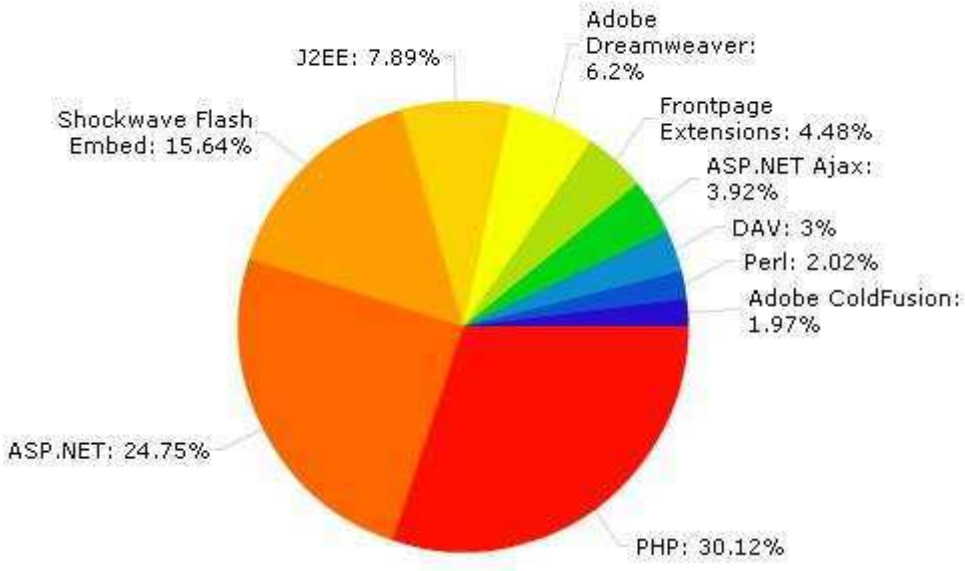
V třetí části je popsán vývoj a použité řešení při programování modulu pro konfiguraci IP telefonů Well 3130IF a Well 3195IF. Jsou zde podrobně popsány problémy vyskytnuvší se při programování modulu a jejich následné řešení, jakožto i programy použité k analýze a řešení těchto překážek.

Na přiloženém DVD jsou umístěny zdrojové kódy webové aplikace pro správu učebny, zásuvného modulu pro konfiguraci IP telefonu, návod na používání webové aplikace a elektronická verze bakalářské práce.

1 ROZBOR POUŽITÝCH TECHNOLOGIÍ V PROJEKTU WEBOVÉ APLIKACE PRO SPRÁVU UČEBNY

1.1 Přehled řešení pro vytvoření a správu webových aplikací

V současné době existuje na trhu mnoho různých řešení pro jednotlivé části projektu. Pro skriptování na straně serveru je na výběr především mezi dvěma konkurenty, PHP a Active Server Page (ASP) společnosti Microsoft. Tyto dva hlavní jazyky zaujímají dohromady okolo 60% řešení webových projektů (obr. 1.1). Zbytek trhu připadá na další méně významné řešení soukromých společností například společnost Adobe se svými ColdFusion a Shockwave Flash.



Obr. 1.1: Srovnání využívání jednotlivých skriptovacích jazyků k 1.12.2009 [10]

V databázích máme na výběr mezi různými typy rozdělujících se do skupin podle způsobu uložení dat. Jedná se o databáze hierarchické, síťové, relační, objektové a objektově-relační. Podrobněji si budeme rozebírat pouze relační databáze, které jsou dnes nejvíce využívány na webech.

Relační databáze používají pro komunikaci standardizovaný dotazovací jazyk SQL, Structured Query Language. Nejnovějším standardem je SQL:2008. Nejznámější a nejpoužívanější relační databáze jsou MySQL, MS SQL Server, PostgreSQL a Oracle.

Každá z těchto databází různých firem implementuje vždy část normy SQL jazyka a naopak ho rozšiřuje o vlastní, proprietární funkce. Pro dokonalou přenositelnost mezi databázemi různých výrobců je tedy potřeba důsledně používat pouze standardizované funkce a datové typy a vyhnout se speciálním rozšířením, které by mohly v budoucnu znamenat problém například při přechodu na jinou databázi.

Pro zobrazení výstupů na monitor se používá nejčastěji značkovací jazyk HTML (HyperText Markup Language) nebo XHTML (eXtensible HyperText Markup Language) jako rozšíření jazyka XML (eXtensible Markup Language) vydaného konsorciem W3.

1.2 PHP jako skriptovací jazyk

PHP je rekurzivní zkratkou názvu PHP: Hypertext Preprocessor [7]. Byl vyvinut v roce 1995 dánsko-kanadským programátorem Rasmusem Lerdorfem. Jedná se o serverový skriptovací jazyk určený pro vytváření dynamických webových stránek. Serverový skriptovací jazyk, protože se PHP kód interpretuje a vyhodnocuje na webovém serveru, kde se vygeneruje nejčastěji HTML kód, který je následně odeslán zpět k uživateli na zobrazení.

K jeho běhu je tedy nutný server, na kterém se budou PHP skripty vykonávat. Zde nastupuje jedna z hlavních výhod PHP – multiplatformnost. PHP můžeme provozovat pod většinou současných systémů, ať už na bázi UNIX-u jako jsou například GNU/Linux, FreeBSD nebo Solaris společnosti Sun Microsystem, tak na všech verzích operačního systému Microsoft Windows. O samotný běh skriptů v operačním systému se pak starají webové servery, kdy nejznámější jsou v prostředí GNU/Linux webový server Apache, v prostředí Microsoft Windows Internet Information Services (IIS). Pro nejčastější propojení operačních systémů, serveru, databáze a skriptovacího jazyka se vžilo označení LAMP (Linux, Apache, MySQL, PHP) a WAMP (Windows, Apache, MySQL a PHP). Tyto kombinace jsou používány na naprosté většině současných serverů.

Další velkou výhodou PHP je jeho příslušnost k Open Source produktům, což znamená, že jeho zdrojové kódy jsou volně šiřitelné a každý do nich může nahlédnout. Tyto zdrojové kódy je také možné libovolně upravovat a v případě potřeby nějaké vlastnosti je ji možné ihned implementovat přímo do PHP a nečekat, až bude vydána v oficiální verzi. PHP je distribuováno pod vlastní licencí, v současné době konkrétně: The PHP License, version 3.01[8].

Samotná syntaxe jazyka PHP vychází z několika starších programovacích jazyků, konkrétně C, Java a Perl [7]. Proto je pro zkušené programátory přechod na jazyk PHP snadnou záležitostí a pro začátečníky velmi jednoduchý na naučení. O jednoduchosti a oblíbenosti programování stránek právě v jazyce PHP svědčí jeho rozšířenost, kdy si stabilně drží nad 29% podílu na trhu (obr. 1.2). Podíl oproti ostatním konkurentům k 1.12.2009 je zobrazen na obrázku 1.1.

Obr. 1.2: Statistika využíváni PHP v roce 2009 [9]

PHP/FI

První verze PHP byla vydána 8. června 1995, tehdy ještě pojmenovaná „Personal Home Page Tools“. Byla vytvořena za účelem zpracování záznamu o přístupech k webu Rasmuse Lerdorfa v jazyce Perl, později, s rozšiřováním funkcí, implementována do jazyka C. Používala Perl-like proměnné a obsahovala některé základní funkce, které přetrvaly do dneška, jako je například automatická interpretace proměnných formuláře nebo přístup k databázím [9].

PHP/FI2

Vydáno 16. dubna 1996, pouze mezistupeň mezi PHP 1 a PHP 3. Většinu času dostupné pouze v betaverzích [9].

PHP v3.0

Vydáno 6. června 1998, na vývoji se podíleli Andi Gutmans a Zeev Suraski. Největším přínosem v téhle verzi byla podpora mnoha různých databází, protokolů, API. Dále pak přibyla podpora objektově orientovaného programování a vylepšila se původní syntaxe.

Spolu s těmito vylepšeními došlo také ke změně názvu jazyka na dnešní PHP: Hypertext Preprocessing jako signalizace, že se už nejedná pouze o jazyk sloužící pouze pro osobní stránky [9].

PHP v4.0

Vydáno 22. května 2000, v této verzi bylo kompletně přepsáno jádro jazyka PHP. Nově bylo postaveno na Zend enginu (spojení křestních jmen Zeev a Andi), který vylepšoval efektivitu a zlepšoval výkon ve složitých aplikacích. Dále byla zlepšena modularita PHP a vylepšena podpora databází a API třetích stran. Dále byly přidány superglobální proměnné jako je $ _SESSION, $_POST a $_GET, také byla přidána podpora pro více web serverů [9].

PHP v5.0

Vydáno 13. července 2004. V jádře použit Zend Engine II, vylepšena podpora pro objektové modelování, přidány nové funkce přímo do php, od verze 5.3 přidána podpora jmenných prostorů [9].

Version 5.3.0

V roce 2009 byla 30.června vydána nová aktualizace PHP v5, a to konkrétně na verzi 5.3.0. Ta kromě spousty vylepšení a oprav nahlášených chyb přináší podle informací:

Improved crypt() function: (Pierre)

Added Blowfish and extended DES support. (Using Blowfish implementation from Solar Designer).

Made crypt features portable by providing our own implementations for crypt_r and the algorithms which are used when OS does not provide them. PHP implementations are always used for Windows builds [17].

Tohle vylepšení má za následek nepřenositelnost zašifrovaných hesel uložených v databázi mezi staršími verzemi a verzí 5.3.0. Je nutné všechna hesla vymazat a uložit pomocí nového šifrování, nebo si napsat vlastní šifrovací funkci, která bude zaručovat stejný výsledek ve všech současných i budoucích verzích PHP.

Version 5.3.2

Zatím poslední verze PHP vydaná 4.března 2010 přináší kromě dalších oprav nahlášených chyb také podporu kvalitnějších šifrování SHA-256 a SHA-512, vylepšena ochrana superglobálního pole $_SESSION [19].

PHP v6

V současné době je ve vývoji šestá verze PHP mezi jejíž hlavní novinky, co se týče vztahu k mnou řešenému problému, patří úplné odstranění register_globals (v PHP5 je stále přítomno, ale jako výchozí hodnota je off a musí se dodatečně zapnout v .htaccess). Tato změna způsobí, že k prvkům superglobálních polí $_POST, $_GET ani $_SESSION již nejde přistupovat přímo zkráceným zápisem, ale musí se na začátku každého skriptu přepsat do zkrácených proměnných nebo

k nim přistupovat jako k prvkům pole. Odstraněním register_globals dojde ke zvýšení bezpečnosti aplikací, protože již nebude možné pro útočníka jednoduše podvrhnout proměnné používané ve skriptech [18].

1.2.1 Syntaxe jazyka PHP

Syntaxe jazyka vychází původně z jazyka Perl. Začátek kódu musí být označen symbolem „<?php“, případně zkrácenou verzí „<?“, ukončení skriptu pak dojde použitím symbolu „?>“. Názvy proměnných musí vždy začínat klíčovým znakem „$“ a nemusí být, na rozdíl od jiných programovacích jazyků, dopředu deklarovány a nemusí být při prvním zápisu uvedeno jakého typu proměnná bude. Datový typ proměnné si PHP hlídá samo podle vložené hodnoty při definici proměnné. S těmito proměnnými můžeme ve skriptu libovolně pracovat, přiřazovat k nim různé hodnoty nebo je měnit. Pokud chceme definovat konstantu, která bude mít vždy stejnou hodnotu a chceme zamezit jakékoliv, byť i nechtěné změně této hodnoty, použijeme pro její zápis klíčové slovo DEFINE(‘název_konstanty’, ’hodnota’). K takto definované konstantě potom přistupujeme přímým zápisem názvu bez znaku $ a můžeme z ní pouze získávat data, ne do ní už zapisovat.

Podmínky a cykly mají v PHP dvě různé varianty zápisu. První je obdobná zápisu známého například z jazyka C, kdy po názvu podmínky nebo cyklu se kód vykonávaný v těle podmínky nebo cyklu uzavře do složených závorek, například if(nějaká_podmínka) { blok příkazů vykonávaný při splnění podmínky}. Druhou variantou je zápis podobný jazyku Visual Basic, kdy se pro uzavření kódu vykonávaných v těle podmínky nebo cyklu nepoužije složených závorek, ale ukončí se příkazem end s názvem příkazu, který byl na startu bloku. Příklad: for(počet_opakování) blok příkazů vykonávaný po dobu definovanou v příkazu for endfor;. Protože jsem již dříve programoval v jazyku C, volím raději způsob zápisu stejný jako v něm, tedy variantu se složenýma závorkami. Usnadňuje to migraci mezi těmito jazyky a snižuje riziko vzniku chyby špatným zápisem.

Poslední důležitou věcí je zápis funkcí v jazyku PHP. Jazyk PHP obsahuje celou řadu vnitřních funkcí pro nejčastěji používané příkazy, takže je nemusíme již sami znovu definovat v každém projektu zvlášť. Jejich zápis je ve tvaru název_funkce(argumenty_funkce) a jedná se například o sadu funkcí pro práci s databází MySQL, funkce pro práci se soubory, časem atd.

I když je paleta vnitřních funkcí poměrně rozsáhlá, vždy se nám hodí si definovat nějaké vlastní funkce, například pro kontrolu platných e-mailových adres. Pro definici vlastních funkcí se používá zápis začínající klíčovým slovem function název_funkce(argumenty_funkce) { blok příkazů, který funkce vykonává, return návratová_hodnota;}. Na takto vytvořenou funkci se pak již odkazujeme v samotném skriptu jako na jakoukoliv jinou funkci obsazenou přímo v PHP.

1.3 Použití a vývoj databáze MySQL

MySQL je víceuživatelský a vícevláknový databázový server vyvinutý švédskou firmou MySQL AB, nyní vlastněný firmou Sun Microsystem. Počátky vývoje sahají až do roku 1979, nicméně první veřejnosti přístupná verze se objevila až v roce 1996 [3]. Jedná se o multiplatformní databázi použitelnou jak na serverech s operačním systémem GNU/Linux, tak na operačních systémech Microsoft Windows. MySQL je typický příklad softwaru s dvojí licencí, je možno využít jak Community Edition licenci GPL, která je bezplatná, tak komerční Enterprise licenci [6]. Komerční licence zahrnuje kompletní servis a podporu pro server MySQL, verze šířena pod licencí GPL se musí spolehnout pouze na podporu komunity složené z technologických nadšenců a open source vývojářů.

MySQL používá pro komunikaci standard SQL, který rozšiřuje o své vlastní příkazy nenacházející se v příslušném standardu ISO/IEC. Podporuje širokou škálu různých typů tabulek, jako jsou InnoDB, MyISAM, IBMDB2I, BerkeleyDB. Nejvíce používané jsou tabulky typu MyISAM, které jsou nastaveny jako základní datové úložiště MySQL. Jejich předností je rychlost, nevýhodou, že nepodporují transakce a cizí klíče.

Tyto nedostatky řeší typ tabulek InnoDB, který transakce i cizí klíče podporuje, avšak je pomalejší než MyISAM. InnoDB dále nabízí podporu bodů obnovení což minimalizuje hrozbu ztráty dat. Při návrhu databáze se tedy musíme rozhodnout, zda je pro nás důležitější rychlost nebo potřeba transakcí a lepší zálohy dat.

MySQL v3.23

Oficiálně vydána v lednu 2001. Obsahuje podporu cizích klíčů a transakcí pro tabulky typu InnoDB.

MySQL v4

Oficiální vydání v březnu 2003. Přidává podporu SQL příkazu UNION, který slouží pro sjednocování dotazů ze dvou tabulek se stejnými daty a automaticky filtruje duplicitní záznamy. Od verze 4.1 přidává podporu pro různé znakové sady a podporu pro poddotazy.

MySQL v5

Oficiálně uvolněno v říjnu 2005. Implementuje podporu triggerů a distribuovaných XA transakcí pro tabulky InnoDB. Od verze 5.1 pak přidává podporu datového úložiště IBMDB2I, replikaci na úrovni řádků a logování na straně serveru.

1.4 HTML a CSS pro zobrazení dokumentů v prohlížeči

1.4.1 HTML

Jazyk HTML slouží pro výpis informací na výstup internetového prohlížeče a je v současnosti spravován konsorciem W3C které vydává specifikace a standardy pro nové verze jazyka. Jazyk HTML má popisovat pouze obsah a strukturu dokumentu, ne jeho grafické zobrazení. O grafické zpracování dat se mají starat až přiložené styly k jednotlivým HTML značkám.

HTML v0.9

Vydána v roce 1991, pouze textový režim [11].

HTML v2.0

Vydána v roce 1994 komunitou IETF, vychází ze standardu jazyka SGML a poprvé přidává podporu grafiky a formulářů [12].

HTML v3.2

Vydána v lednu 1997 konsorciem W3C. Přidává podporu pro tabulky a styly [13].

HTML v4.0

Vydána v prosinci 1997 s rozšířením prvků pro tvorbu tabulek a formulářů. Zároveň byly jako standardy přijaty rámy. Větší důraz na používání stylů pro tvorbu grafiky a vydáno doporučení samotné HTML nechat pouze pro vyznačení významu prvků [15].

HTML v4.1

Vydána v prosinci 1999, tato verze nepřidává žádné důležité věci do standardu, pouze opravuje některé chyby. Tato verze HTML měla původně být poslední a HTML nemělo být nadále vyvíjeno. Počítalo se s přechodem na XHTML, ale od tohoto plánu bylo nakonec upuštěno a v roce 2007 byla konsorciem W3C ustanovena pracovní skupina s úkolem vyvinout další verzi HTML pojmenovanou jako HTML 5 [14].

1.4.2 CSS

Cascading Style Sheet, neboli kaskádové styly, je jazyk vytvořený a spravovaný konsorciem W3C pro zobrazení stránek napsaných v jazycích HTML a XHTML. V současné době jsou k dispozici dvě verze CSS a pracuje se na dokončení standardu CSS3.

Hlavní výhodou používání CSS jsou rozsáhlé možnosti formátování textu, oddělení struktury od stylu dokumentu, jednoduchá úprava stávajících stylů za nové změnou jediného souboru a podpora ve všech prohlížečích, které přijaly tento standard.

Nevýhodou pak zůstává neúplná podpora celého standardu ve velkých prohlížečích, speciálně v prohlížečích Internet Explorer Microsoft Windows, v důsledku čehož je často nutné psát různé styly pro různé prohlížeče. Tato podpora se však lepší s každou další vydanou verzí prohlížeče a kaskádových stylů.

Verze CSS1

Vstoupilo v platnost jako doporučení konsorcia W3C 12.prosince 1996 a definuje základy kaskádových stylů, které potom vylepšují další verze [20].

Verze CSS2, revize 1

V současné době kandidát doporučení konsorcia W3C od 8.září 2009. Opravuje chyby z verze 2.0 vydané v roce 1998, obsahuje všechny funkce z verze CSS1 a některá vylepšení.

Jako novinka se ve verzi 2.1 objevuje například přímá podpora pro oranžovou barvu pomocí hodnoty orange atributu color a zobrazení textu display: inline-block, kdy se obsah prvku formátuje jako blok, ale prvek sám se k okolí tváří jako řádkový (inline) prvek [21].

Vylepšení oproti verzi 2.0 se dočkalo pozicování prvků, například nové definování šířky/výšky absolutně pozicovaných prvků v dokumentu, podpora 3D stylů rámečků prvků [22].

Verze CSS3

Nadcházející verze CSS, která je v současné době ve stádiu vývoje a jejíž část už je hotová a implementovaná v nejnovějších verzích prohlížečů. Jedná se například o vylepšení práce s rámečky prvků v dokumentu, konkrétně přidání atributu border-radius, který způsobí zakulacení rohů rámečku o zadaném radiusu [23]. V dřívějších verzích bylo nutné tenhle moderní způsob ohraničení prvků tvořit pomocí vkládání obrázků s nakresleným zakulaceným rohem.

Dalšího vylepšení se dočkal atribut color pro práci s barvami. V CSS3 bude možno kromě slovního, hexadecimálního a RGB zápisu barvy použít i HSL model zápisu barvy [24].

2 PROGRAMOVÁNÍ WEBOVÉ APLIKCE PRO SPRÁVU UČEBNY

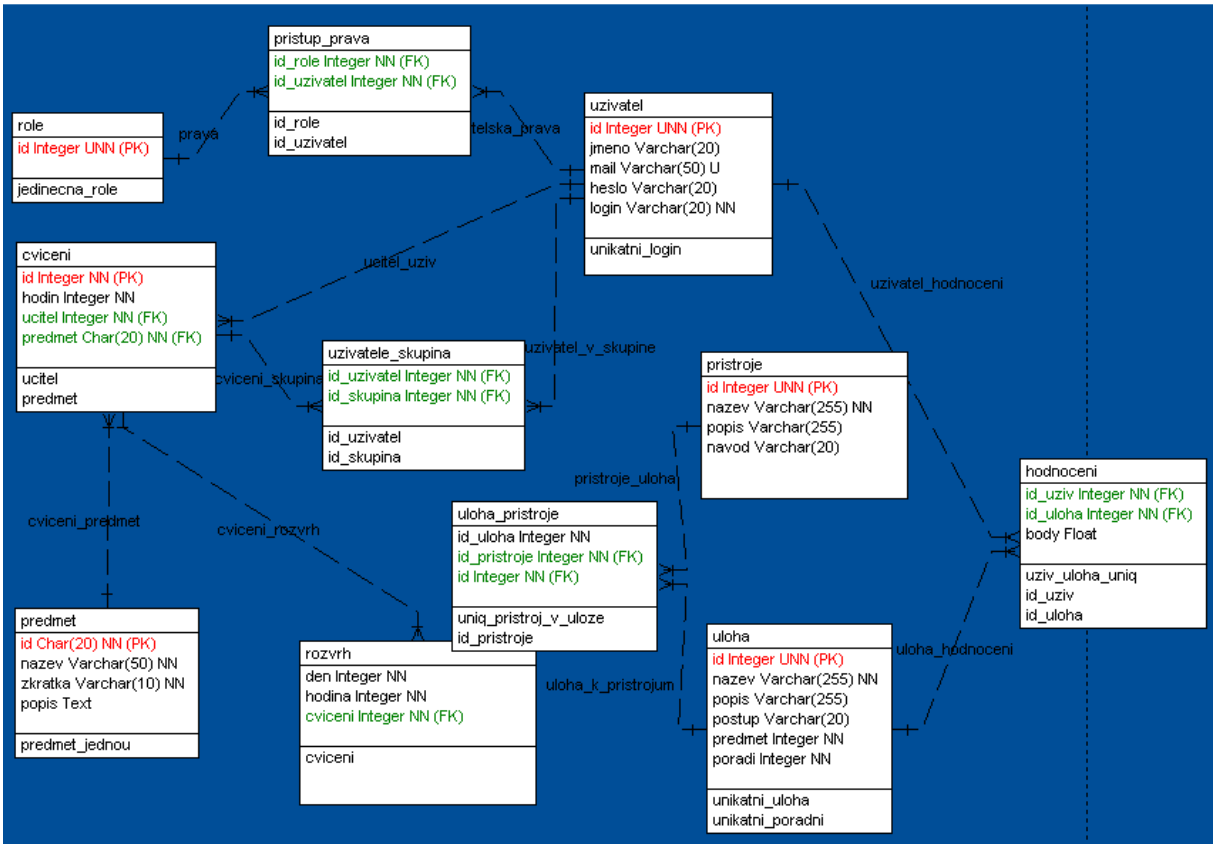
2.1 Vývojové prostředí

Na začátek je velmi důležité zvolit správné vývojové prostředí, ve kterém se bude aplikace vyvíjet. Správně zvolené programy totiž dokáží podstatně snížit nároky kladené na vývojáře tím, že přeberou některé starosti pod svou správu a programátor se může víc soustředit na samotný vývoj a výslednou funkčnost aplikace.

Pro webovou aplikaci pro správu učebny jsem se rozhodl použít známou zavedenou kombinaci WAMP, kde je již zaručena bezproblémová funkčnost a spolupráce všech potřebných složek.

2.1.1 Návrh databáze

Pro návrh databází jsem použil program CASE Studio společnosti CHARONWARE, který umožňuje rychle a přehledně navrhovat Entitně relační diagramy pro rozsáhlé projekty. V jednoduchém grafickém prostředí si navolíme jednotlivé entity, které reprezentují konečné reálné tabulky v databázi. Jednotlivé entity se navzájem propojí relacemi, znázorňujícími vazby mezi tabulkami. Odtud název Entitně relační diagramy. Jednotlivé relace mohou být dvojího typu. Jednodušší relace 1:N, kdy jednomu záznamu v jedné tabulce může odpovídat N záznamů v druhé tabulce, nebo složitější relace M:N, kdy pro M záznamů v jedné tabulce existuje N záznamů v tabulce druhé. Tento vztah nelze udělat přímo a musí se řešit pomocí takzvané vazební tabulky, která má s každou tabulkou, první i druhou, relaci 1:N. Jako typický příklad vazby M:N je vztah mezi přístroji v učebně a úlohami, které jsou vyučovány. Jeden přístroj může být využíván u více úloh, a zároveň u jedné úlohy může být využíváno více přístrojů (obr. 2.1).

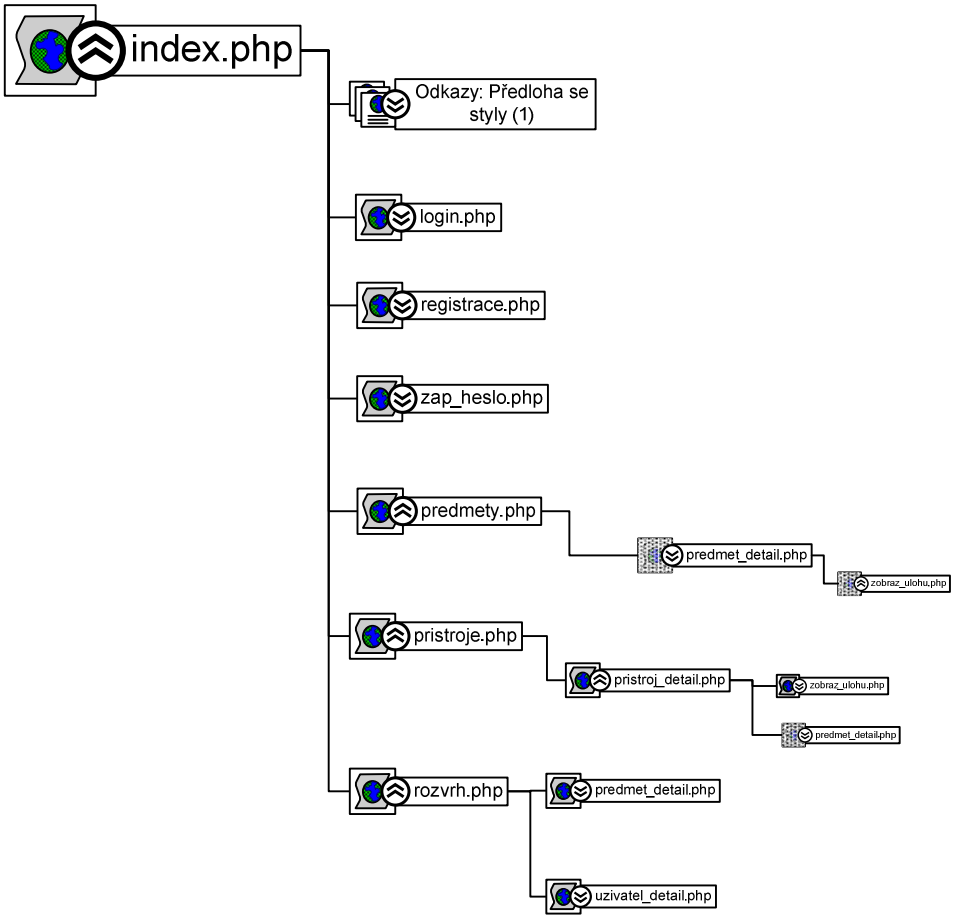


Obr. 2.1: Ukázka návrhu Entitně relačního diagramu v programu CASE Studio

Kromě návrhu ER diagramů umožňuje CASE Studio také vytváření Data Flow diagramů. Spolu s ER diagramy jsou součásti strukturované analýzy návrhu systému a sestávají z procesů, datových toků, skladišť (Data Store) a terminátorů. Proces v DF diagramu znázorňuje tu část systému, která mění vstupní informace na informace výstupní, datový tok znázorňuje přesun informací mezi jednotlivými částmi systému, Data Store ukazuje datové úložiště informací pro pozdější využití a terminátor představuje entity, které jsou vně našeho systému, ale nějakým způsobem s ním komunikují. Pomocí DF diagramů lze během chvíle popsat celý navrhovaný systém a na základě fyzického pohledu datových toků ho optimalizovat pro naše požadavky.

Po skončení návrhu systému si můžeme výsledný projekt nechat jednoduše vygenerovat jako SQL skript pro databázi MySQL, kdy nám program sám vygeneruje příkazy, které po nahrání do databáze vytvoří námi navrhnutý systém. Tento postup nejenže šetří čas, který bychom museli strávit ručním přepisováním SQL příkazů, ale také snižuje riziko zanesení chyby, která by ručním psaním mohla vzniknout.

Základní schéma struktury webových stránek se nachází na obr. 2.2.



Obr. 2.2: Struktura webových stránek aplikace pro správu učebny

2.1.2 Zabezpečení proti nežádoucímu přístupu

Pro tvoření samotných skriptů PHP jsem zvolil textový editor PSPad. Jedná se o volně šiřitelný textový editor pro operační systémy MS Windows, který nabízí širokou možnost formátování textu a vestavěnou podporu pro mnoho programovacích jazyků. Jde o zvýrazňování syntaxe, což dělá kód

přehlednější pro úpravu a orientaci v něm, doplňování známých příkazů, hlídání správnosti zápisu, zobrazování náhledu stránek, spravování TO-DO listů atd.

Nejdříve jsem se rozhodoval, jaké zvolím zabezpečení aplikace proti vniknutí nežádoucí osoby. V úvahu připadají v podstatě 3 možnosti.

Soubor .htaccess a .htpasswd

Soubory .htaccess a .htpasswd jsou speciální konfigurační soubory sloužící pro nastavení serveru pro jednotlivé adresáře. Jednou z jejich funkcí je možnost nastavit omezení přístupu k vybranému adresáři, v našem případě tedy ke kořenovému adresáři serveru. Nastavit omezení v těchto souborech můžeme buďto zakázáním/povolením přístupu k adresáři z vybraných adres nebo uvedením přístupových jmen a hesel uživatelů.

Velká nevýhoda tohoto řešení spočívá v tom, že fungují pouze na serverech Apache a ne vždycky může být na serveru povoleno používání těchto dodatečných konfiguračních souborů.

Pro můj projekt webové aplikace pro správu učebny je také nevhodné, že omezení nastavené těmito soubory se vztahuje na celý adresář včetně jeho podadresářů a přidávání nových uživatelů je složité, pouze přes editaci konfiguračních souborů.

Použití souborů cookies

Další z možností zabezpečení je použití souborů cookies. Jedná se o soubory uložené na počítači uživatele, ve kterých si můžeme uchovávat jakékoliv informace důležité pro naši aplikaci. Může jít například o jednoznačný identifikátor konkrétní osoby, který uchováváme v souboru cookies na uživatelově počítači a prostým porovnáním se záznamy o uživatelích uložených na serveru, nejčastěji v nějaké databázi nebo speciálním souboru poznáme, zda má uživatel oprávnění přistupovat ke konkrétním chráněným informacím.

Výhodou tohoto řešení oproti použití souborů .htacces a .htpasswd je možnost nastavovat tímhle způsobem zabezpečení jak jednotlivých skriptů, tak i pouze jejich konkrétních částí.

Nevýhodou je uložení souboru na počítači uživatele, tudíž nemáme kontrolu nad tím, zda si cookies nějak neupravil nebo nám dokonce nepodvrhl falešnou. Další nevýhodou je možnost zakázání přijímání cookies ze strany uživatele, čímž se znemožní základní funkčnost skriptů.

Řízení sezení v PHP

Třetí možností je využití superglobálních proměnných $_SESSION, do kterých si můžeme uložit jakékoliv hodnoty, nejčastěji ty, které jsou důležité pro jednoznačnou identifikaci uživatele, nebo hodnoty, které často využíváme a potřebujeme je přenášet mezi jednotlivými skripty. Hodnota proměnné $_SESSION je po nastavení udržována ve speciální paměti serveru po celou dobu trvání návštěvy a je přístupná z kteréhokoliv skriptu patřícího k danému serveru.

Data uložená v poli $_SESSION jsou přístupná do uzavření spojení mezi serverem a počítačem uživatele, nebo do manuálního zrušení na serveru. Velkou výhodou oproti cookies je, že uživatel nemá žádný přístup k těmto proměnným a nemůže je tudíž nijak upravit, nevýhodou je, že vyžaduje vyšší výkon serveru oproti cookies, které jsou uloženy na počítači uživatele.

V našem případě nastavujeme tři proměnné $_SESSION při přihlášení do aplikace a to ID uživatele, který se přihlásil, čas posledního kliku v aplikaci a úroveň přístupových práv. ID uživatele potřebujeme pro jednoznačnou identifikaci uživatele na serveru, čas posledního kliku pro zajištění bezpečnosti v případě, že se uživatel zapomene po skončení práce odhlásit. Po určité době pak bude

session neplatná a potencionální útočník se tak bez znalosti přístupových údajů, přihlašovacího jména a hesla k důležitým částem aplikace nedostane.

V aplikaci na správu učebny velmi často zjišťujeme, zda má uživatel oprávnění provést nějakou operaci, například přidání předmětu nebo smazání jiného uživatele. Tato oprávnění jsou uložena v databázi na serveru a při každém spuštění skriptu by se nejdřív dotazoval databázového serveru, jaké má uživatel oprávnění. Jelikož je tahle hodnota po většinu času pořád stejná, pomineme-li speciální případ, kdy bude administrátor zrovna měnit někomu oprávnění, můžeme si tuhle hodnotu také uložit do jedné proměnné $_SESSION, ve které ji budeme mít po celou dobu k dispozici. Tímto způsobem můžeme ulevit od zátěže databázovému serveru.

2.1.3 Tvorba administračního skriptu

Administrační skript je nejdůležitější součásti aplikace. Umožňuje širokou možnost úprav stránek a dat v nich obsažených.

Hned na začátek jsem se musel potýkat s velkým problémem, kterým byla rozsáhlost tohoto skriptu. Skript obsahuje přes 700 řádků kódu, což velmi snižuje jeho čitelnost a každá úprava, byť i minimální, se stává složitou pro hledání patřičného místa, kde se potřebný kus kódu nachází. Proto jsem se rozhodl zvýšit přehlednost skriptu rozdělením jednotlivých částí na tři celky uložené ve třech souborech, rozdělené podle funkce, kterou plní. Jeden soubor obsahuje zdrojový kód pro veškerou práci s uživateli, druhý pro práci s předměty a třetí pro práci s externími moduly. Bloky kódů v těchto souborech jsou navíc definovány pomocí vlastních funkcí, které jsou pak volány ze základního administračního skriptu, do kterého jsou soubory vloženy přes interní php funkci include(„adresa_souboru“).

Taková konstrukce nejenže zvýší přehlednost výsledného kódu, kdy je hned poznat, která funkce si vyžádala jaký kus kódu, ale také zlepšuje možnosti vyhledání potřebného místa, kde se má udělat úprava a umožňuje opětovné využití již jednou napsaného kódu voláním dané funkce z více míst.

Administrace uživatelů

Část administračního skriptu administrace.php sloužící k práci s uživateli využívá pomocný soubor administrace_uzivatel.inc uložený v adresáři /include, obsahující funkce volané ze skriptu administrace.php.

Základní funkce pro výpis všech uživatelů uz_zobraz_prehled_vsech($_GET, $radku) přejímá jako parametry superglobální pole $_GET a proměnnou $radku, ve které je uloženo, kolik řádků na stránku se má vypsat. Defaultně je v administrace.php nastavená konstanta RADKU na hodnotu 10, tzn. vypíše se vždy 10 řádků na stránku a zobrazí se navigační menu pro posun o dalších 10 řádků vpřed nebo vzad. Konstanta se v php zapisuje pomocí funkce define, která má tvar

bool define (string $name , mixed $value [, bool $case_insensitive = false]).

Parametr $name značí název konstanty, parametr $value hodnotu konstanty a nepovinný parametr $case_insensitive, který zadává zda bude v názvu konstanty brán ohled na velikost písmen nebo nikoliv. Základní hodnota je false, tedy case sensitive. Pokud se použije hodnota true, pak jsou všechny konstanty brány jako zapsané malými písmeny [7].

V poli $_GET pak může funkce přejímat hodnoty $_GET['vyhl_jm'] pro nastavení filtru vyhledávání podle části jména, $_GET['vyhl_mail'] pro nastavení filtru vyhledávání podle části

e-mailové adresy, $_GET['rad'] pro nastavení podle jakého kritéria mají být výsledky seřazené na stránce a hodnoty $_GET['od'] a $_GET['celkem'] sloužící pro nastavení správného stránkování.

Výstupem funkce je zobrazení všech uživatelů, vyhovujících parametrům filtru zadaných na vstupu funkce na monitor.

Přidat nového uživatele lze i manuálně zásahem administrátora. Pro zobrazení formuláře pro takovéto přidání slouží funkce uz_zobraz_pridani_noveho(), která nepřebírá žádné parametry a jejím výstupem je zobrazení formuláře na monitor. Zpracování dat z tohoto formuláře pak probíhá ve funkci uz_pridej_noveho($_POST), přijímající jako parametry hodnoty ze superglobální proměnné $_POST, konkrétně $_POST['jmeno'], $_POST['mail'] a $_POST['login']. Jakékoliv jiné hodnoty uložené v poli jsou ignorovány. Všechny vstupní hodnoty jsou nejdříve zpracovány funkcí trim

string trim (string $str [, string $charlist]),

která odstraní z řetězce $str všechny prázdné znaky nacházející se na začátku a na konci řetězce. Volitelný parametr $charlist umožňuje definovat další, neprázdné znaky, které má funkce taktéž odstranit [7].

Funkce uz_pridej_noveho() sama volá 3 funkce definované v souboru funkce.inc v adresáři /include. Jedná se o generuj_heslo(), zkontroluj_registraci($jmeno, $mail, $login, $pass, $pass2) a vloz_uzivatele($jmeno, $mail, $login, $pass).

Funkce generuj_heslo() nepřebírá žádný parametr a jejím výstupem je náhodně vygenerované heslo. Tato funkce sama využívá ještě podpůrnou funkci nahodne_slovo($min_delka, $max_delka), která vyhledá v elektronickém slovníku náhodné slovo o délce v rozmezí zadané hodnotou $min_delka až $max_delka. Jako zdroj dat zde posloužil anglický slovník ispell verze 3.1.20. Funkce využívá generátor náhodný čísel

void srand ([int $seed]),

která inicializuje generátor náhodných čísel s nepovinnou vstupní hodnotou $seed nebo s náhodnou hodnotou pokud není zadána [3].

int rand ([int $min , int $max])

Poté vrátí náhodné číslo z intervalu zadaným hodnotami $min a $max [7].

K náhodnému slovu ze slovníku ispell, vrácenému funkcí nahodne_slovo(), se ještě přiřadí náhodné trojmístné číslo a tento řetězec je poslán na výstup funkce generuj_heslo().

Funkce zkontroluj_registraci($jmeno, $mail, $login, $pass, $pass2) přejímá jako paramaetr kompletní údaje potřebné pro vložení nového uživatele a provádí kontrolu správného vyplnění. Využívá interní funkci

int strlen (string $string),

která vrací délku zadaného řetězce $string [7].

Na kontrolu validní e-mailové adresy používá funkci spravny_email($mail)[3] přebírající parametr $mail, což je e-mailová adresa vložená uživatelem do formuláře.

int ereg (string $pattern , string $string [, array &$regs])

Prohledá řetězec $string na přítomnost regulérního výrazu $pattern, seznam možných znaků je v tab. 2.1 [3]. Prohledávání probíhá v case sensitive módu, takže záleží na velikosti zadaných písmen. Do volitelného parametru pole $regs jsou postupně ukládany všechny nalezené výskyty shody $string s $pattern, a to postupně od $reg[1]. V $reg[0] je uložen celý původní řetězec. Funkce ereg() je podporována pouze do verze PHP 5.2, od verze PHP 5.3 již nadále není poskytována [7].

Tab. 2.1: Seznam možných znaků ve stylu POSIX

Znak	Význam
\	řídící znak
^	shoda na začátku řetězce
$	shoda na konci řetězce
.	shoda se vším kromě znaku \n
|	začátek nové větve
(	začátek podvýrazu
)	konec podvýrazu
*	0 nebo více opakování
+	jedno nebo více opakování
{	začátek kvantifikátoru
}	konec kvantifikátoru

Následující funkcí po projití předchozích je vloz_uzivatele($jmeno, $mail, $login, $pass), kdy jako parametry přebírá údaje o uživateli a vloží je do databáze. Heslo $pass není ukládano v čistém formátu, ale je na něj nejdříve uplatněna funkce

string crypt (string $str [, string $salt]),

která vrátí zahashovaný řetězec $str podle základu $salt a ten je teprve uložený do databáze.

Různé operační systémy a různé verze PHP mohou používat různé hashovací algoritmy pro funkci crypt(). Z tohoto důvodu nemusí být možné přenést celou databázi včetně hesel na jiný server, nebo aktualizace serveru na vyšší verzi se zachováním stoprocentní funkčnosti uložených hesel [7].

Zbytek funkcí v souboru administrace_uzivatel.inc se zabývá úpravou, případně mazáním již existujících uživatelů. Jde konkrétně o funkce uz_zobraz_detail($detail) pro vypsání detailních informací o uživateli na monitor, uz_zobraz_upravu($detail) pro zobrazení formuláře na editaci údajů o vybraném uživateli, uz_smaz($detail) pro smazání vybraného uživatele a uz_zobraz_prid_skup($detail) pro zobrazení formuláře na přiřazení uživatele do skupiny. Ve všech případech funkce přejímají jako parametr proměnnou $detail, ve které je uloženo unikátní ID uživatele, pro kterého dané funkce voláme.

Poslední dvě funkce slouží na zpracování dat odeslaných z formulářů pro editaci uživatele a přejímají jako vstupní parametry hodnoty superglobálního pole $_POST. Jsou to uz_pridej_skupinu($_POST) zpracovávající data týkající se přiřazení uživatelů do skupin a pracující pouze s poli $_POST['uzivatel_id'] a $_POST['id_skup'], a uz_uprav_pristup_prava($_POST) sloužící k úpravě přístupových oprávnění do jednotlivých částí aplikace.

Ukázka administrace uživatelů viz Přílohy, obr. A.2.

Administrace předmětů

Pro práci s administrací předmětů je napsán pomocný soubor administrace_predmet.inc v adresáři /include, viz Přílohy obr. A.4. V něm jsou obsaženy funkce volané z hlavního admistračního skriptu, kdy se jedná o zobraz_prehled_predmetu($rad) přebírající jako parametr proměnnou, podle které se seřazují výsledky vypsané z databáze a tyto jsou zobrazeny na výstup monitoru, zobraz_predmet($detail) přebírající jako parametr unikátní ID předmětu, o kterém chceme zobrazit podrobnější informace, zobraz_pridani_editaci_predmetu($akce, $id) pro přidání nového nebo editaci stávajícího předmětu. Má povinné parametry $akce, který může nabývat hodnot „predmet_edit“ nebo „predmet_pridej“, a $id, což je unikátní ID předmětu, který chceme upravit. V případě přidávání nového předmětu se musí předat hodnota ID null. Funkce zobraz_pridani_editaci_cvik($detail, $akce, $id_cvik) pak podle parametru $akce zobrazí buď přidání nového nebo editaci stávajícího cvičení. Parametr $detail obsahuje unikátní ID předmětu, kterého se úprava cvičení týká a v proměnné $id_cvik je buď unikátní ID upravovaného cvičení nebo hodnota null pro přidání nového.

Další funkce slouží pro zpracování dat odeslaných z formulářů zobrazených pomocí funkcí zmíněných výše a vždy přejímají jako parametr hodnoty ze superglobální proměnné $_POST. Jedná se o pridej_edituj_predmet($_POST) pro přidání nového nebo úpravu stávajícího předmětu a o pridej_edituj_cvika($_POST) pro práci s cvičeními.

Na všechny textové uživatelské vstupy je před uložením do databáze aplikováno přidání zpětných lomítek \

string addslashes (string $str)

před všechny znaky, které by mohly být interpretovány jako řídící a mohly by způsobit prolomení zabezpečení aplikace. Funkce přebírá jako vstup řetězec $str zadaný uživatelem do pole formuláře a vrací tentýž řetězec doplněný o zpětná lomítka před všechny řídící znaky používané v databázových dotazech [7].

Při výpisu z databáze je potom potřeba všechna tato přidaná zpětná lomítka z textu odstranit a vypsat na výstup původně vložený text.

string stripslashes (string $str) [7].

Poslední dvě funkce z pomocného souboru administrce_predmet.inc, smaz_predmet($detail) a smaz_cviceni($detail, $id_predmet) mají za úkol pouze vymazat z databáze vybraný předmět nebo cvičení přiřazené k předmětu.

Administrace modulů

Blok kódu určený pro obsluhování práce s moduly se nachází v pomocném souboru administrace_modules.inc v adresáři /include. Hlavní částí je blok pro výpis všech již nainstalovaných dodatečných modulů do aplikace zobraz_vsechny(). Funkce nepřebírá žádný parametr a pomocí interních PHP funkcí pro práci s databází MySQL

resource mysql_query (string $query [, resource $link_identifier])

int mysql_num_rows (resource $result)

object mysql_fetch_object (resource $result [, string $class_name [, array $params]]),

vypíše na monitor všechny nainstalované zásuvné moduly [7].

Funkce zobrazující formulář pro nahrání nového zásuvného modulu pridat_modul() nepřebírá žádný parametr a jejím výstupem je seznam všech modulů připravených a umístěných v adresáři /modules. Ke zjištění, jaké moduly jsou přítomny, využívá funkci

resource opendir (string $path [, resource $context]),

která otevře adresář zadaný v cestě $path. Při úspěšném volání funkce vrací odkaz na zdrojový adresář předaný funkci nebo FALSE, pokud volání úspěšné není [7].

Pro listování v adresáři otevřeném funkcí opendir() slouží funkce

string readdir ([resource $dir_handle]),

která vypíše seznam všech souborů umístěných v adresáři, na který se odkazuje $dir_handle [7]. Z tohoto seznamu je ještě nutné odfiltrovat pro nás nepotřebné . a .. , což jsou součásti relativní cesty k adresáři a značí aktuální pracovní a nadřazený adresář, a také všechny adresáře, které už jsou nahrané v aplikaci.

Samotné nahrání zásuvného modulu do aplikace pak obstarává funkce pridame_modul($_POST), která jako parametry přebírá data odeslané funkcí pridat_modul(). K úspěšnému nahrání funkce kontroluje, zda se v modulu nachází 2 klíčové soubory, a to tabulky.inc a tabulky_delete.inc v adresáři /modules/nazev_modulu/include, sloužící pro nahrání a vymazání databázové struktury potřebné pro běh daného modulu do databáze MySQL.

bool file_exists (string $filename)

Přebírá jako vstupní parametr cestu k souboru, u kterého potřebujeme zjistit zda fyzicky existuje. Vrací hodnotu TRUE, pokud se soubor nachází na disku, nebo FALSE pokud nikoliv [7].

Poslední funkcí v souboru administrace_modules.inc je smaz_modul($id_mod) pro vymazání nahraného mudulu z aplikace. Funkce přebírá jako parametr unikátní ID modulu a voláním souboru tabulky_delete.inc vymaže z databáze strukturu dodanou při nahrávání modulu.

Ukázka administrace modulů viz Přílohy, obr. A.5.

Zabezpečení administračního skriptu

Vzhledem k povaze skriptu a k možnostem přístupu a editace prakticky kteréhokoliv místa aplikace je obzvláště důležité zabezpečení proti přístupu neautorizované osoby. Takové proniknutí by mělo fatální důsledky pro celou aplikaci, jelikož by útočník dostal přístup prakticky ke všem funkcím, které jsou k dispozici. Proto při každém volání kódu z administračního skriptu je potřeba nejdříve spustit kontrolu, zda má uživatel volající danou funkci vůbec pravomoci tento kód vyžadovat.

Pro tento účel máme právě v proměnné $_SESSION['prava'] uloženou úroveň oprávnění přihlášeného uživatele, která je kdykoliv k dispozici v průběhu trvání komunikace mezi uživatelem a serverem. Jednoduchou vlastní funkcí, kterou jsem si napsal, pak můžeme otestovat. zda má nebo nemá přístup ke konkrétnímu kódu, který si žádá.

2.1.4 Výpis dat a učitelský modul

Poté, co již máme hotovou kompletní databázi a obsluhující administrační skript, který nám umožní do ní vkládat data, můžeme přistoupit k napsání skriptů, které tyto data vypíší z databáze a zobrazí uživateli. Jedná se o jednoduché příkazy typu select „data“ from „tabulka“ where „podmínka

pro filtr“, kdy každý výpis má svůj vlastní skript pojmenovaný, pro lepší přehlednost, podle funkce, kterou vykonává.

Poslední věcí, programovanou pro webovou aplikaci pro správu učebny, je učitelský modul obsahující funkce pro správu vyučovaných předmětů. Tyto funkce jsou přidány ke každému skriptu pro výpis dat a umožňují přidávat, upravovat a odstraňovat doplňující informace jako jsou například úlohy k jednotlivým předmětům, návody k nim apod, viz Přílohy, obr. A.6 a A.7.

Jelikož lze pomocí nástrojů pro učitele přímo editovat data na serveru a přistupovat k databázi, je nutné tyto funkce zabezpečit obdobným způsobem jako administrační skript proti vniknutí neautorizované osoby. Veškerá data vkládaná pomocí webových formulářů, odesílaná na server a ukládaná do databáze je vždycky potřeba zkontrolovat, zda obsahují skutečně hodnoty, které mají a preventivně je nechat projít funkcí addslashes(). Před výpisem dat z databáze do prohlížeče je zase potřeba použít funkci stripslashes() pro správné zobrazení dat na monitor v původnm formátu.

Dodržováním těchto bezpečnostních postupů se sníží riziko prolomení aplikace na minimum.

3 KONFIGURACE IP TELEFONU

3.1 Propojení konfiguračního rozhraní

Konfigurační rozhraní pro telefony Well 3130IF a Well 3195IF je řešeno jako zásuvný modul do webové aplikace pro správu učebny. Oba dva typy telefonu se liší pouze hardwarem, software pro konfiguraci telefonu používají stejný, proto je možné pro ně použít stejný modul.

Telefony lze slučovat do skupin podle úloh, ke kterým jsou přiřazeny a ty pak dále spravovat buď jednotlivě nebo hromadně, pokud k tomu máme oprávnění. Přístup ke konfiguraci telefonu má každý, kdo je přihlášen do webové aplikace pro správu učebny, a to buď přes seznam všech přístrojů uložených v databázi aplikace, přes detail konkrétního telefonu nebo přes skupinu všech přístrojů přiřazených ke konkrétní úloze. Přístup bez uživatelského účtu a bez přihlášení v aplikaci je hlídán a zamezen. Hromadnou konfiguraci celé skupiny telefonů má k dispozici každý s vyšším stupněm oprávnění přes seznam přístrojů přiřazených k jedné úloze.

3.2 Přepis formulářů pro nastavení telefonu

3.2.1 Firebug

Jedná se o rozšíření prohlížeče Mozilla Firefox speciálně určený pro vývojáře webových stránek. Umožňuje rychle a přehledně monitorovat, editovat a debugovat HTML kód, CSS a JavaScript přímo online v prohlížeči, bez nutnosti úpravy zdrojových kódů [25].

3.2.2 Implementované části

Síťová nastavení

V části věnované nastavení sítě byly z originální aplikace implementovány nastavení pro LAN, QoS, Porty služeb a nastavení času.

VoIP nastavení

Implementovány všechny části nacházející se v originální aplikaci. V sekci pro nastavení SIP je základní nastavení pro 2 podporované SIP linky, dále nastavení pro IAX2, STUN a Dial plán.

Nastavení telefonu

V sekci pro nastavení telefonu byly zachovány všechny části z originální aplikace telefonu. Tedy nastavení zvuku, volacích funkcí, klávesnice, telefonního seznamu a funkčních kláves.

Pro snadnější orientaci a lepší komunikaci s telefonem byly všechny formuláře přepsány z originální aplikace tak jak byly, včetně názvu vstupů jednotlivých prvků formulářů. Byly pouze opraveny některé typografické chyby, špatný překlad originální aplikace do češtiny a v sekci pro nastavení SIP pak byly opraveny dvě skupiny vstupů odkazující obě v originální aplikaci na nastavení proxy serveru, ale ve skutečnosti patřící jedna k nastavení proxy serveru a druhá k nastavení registračního serveru.

3.3 Komunikace s telefonem

3.3.1 Požadavky protokolu HTTP

HTTP je internetový protokol definovaný jako standard IETF [26]. Původně byl navržený pro výměnu hypertextových dokumentů ve formátu HTML, ale od verze 1.0 je díky MIME možno přenášet jakýkoliv soubor. V současně aktuální verzi 1.1 se všechna spojení stala trvalými, dokud klient nebo server neukončí komunikaci. Díky trvalým spojením se komunikace mezi klientem a serverem podstatně zrychlila, protože již není třeba otevírat pro každý požadavek spojení nové [27].

Formát požadavku protokolu HTTP
Metoda URL verze_HTTP

odesílané hlavičky

vložený prázdný řádek

Odesílaných hlaviček může být více v jednom požadavku, ale každá z nich musí být oddělena novým řádkem [27].

Metoda GET protokolu HTTP

Metoda GET se používá prakticky pokaždé, když žádáme načtení nějaké stránky a neodeslali jsme přitom formulář s využitím metody POST. Formát požadavku GET je poté následující [27]:

GET URL verze_HTTP

odesílané hlavičky

vložený prázdný řádek

Metoda POST protokolu HTTP

Pomocí metody POST můžeme za odeslané hlavičky a prázdný řádek vložit data, která chceme odeslat pomocí protokolu HTTP. Toho se využívá například při odesílání dat z formuláře webové stránky. Formát požadavku pak vypadá [27]:

POST URL verze_HTTP

odesílané hlavičky

vložený prázdný řádek

data odesílaná serveru

vložený prázdný řádek

Ostatní metody protokolu HTTP

Kromě nejpoužívanějších metod GET a POST jsou v protokolu HTTP zahrnuty ještě méně používané metody. HEAD, která nevrací celou stránku uvedenou v požadavku, ale pouze její hlavičky. PUT uchovává tělo požadavku na zadaném URL. DELETE smaže dokument zadaný v URL ze serveru. TRACE trasuje požadavek přes firewall a proxy, přes které prochází. OPTIONS zadává dotazy na možnosti serveru [27].

Stavové kódy v odpovědi protokolu HTTP

Na každý požadavek klienta odešle server zpět ke klientovi odpověď v jakém stavu se nachází po přijetí jeho žádosti. Tato odpověď má formát:

verze_HTTP stavový_kód stavové_hlášení

poslané hlavičky

vložený prázdný řádek

vrácená data

Stavový kód zde reprezentuje číslo, ke kterému je přiřazeno stavové hlášení, textový popis stavového kódu. Stavové kódy jsou rozděleny do několika kategorií, kdy každá popisuje skupinu se společnými rysy. Jejich seznam je uveden v tabulce 3.1 [28].

Tab. 3.1: Kategorie stavových kódů protokolu HTTP

Kategorie	**Rozsah**	**Popis**
Informační	100-199	pouze informují o průběhu
Úspěch	200-299	potvrzení úspěšného zpracování
Přesměrování	300-399	přesměrování za účelem dokončení požadované operace, uživatel o přesměrování nemusí být informován
Chyba na straně klienta	400-499	problém na straně klienta
Chyba na straně serveru	500-599	problém na straně serveru

V tabulce 3.2 je potom uveden seznam konkrétních stavových kódů s jejich stavovými hlášeními, které se mohou vyskytovat v rozhraní pro konfiguraci IP telefonu [28].

Tab. 3.2: Vybrané stavové kódy a jejich popis

Stavový kód	**Popis**
200 OK	vše proběhlo vpořádku
400 Bad Request	klient odeslal špatný požadavek, kterému server nerozumí
403 Forbidden	požadavek je serverem zamítnut pro špatnou autorizaci klienta
404 Not Found	požadovaná adresa URL není na serveru k dispozici
406 Not Acceptable	server nepodporuje formát požadovaný klientem
408 Request Timeout	vypršel časový limit pro zaslání požadavku klientem
500 Internal Server Error	neočekávaná chyba serveru
503 Service Unavailable	server nemůže zpracovat požadavek, například kvůli přetížení

3.3.2 Programy na odchytávání odeslaných hlaviček

Live HTTP Headers

Live HTTP Headers je rozšíření prohlížeče Mozilla Firefox, umožňující během prohlížení zachytávat a zobrazovat hlavičky, které si mezi sebou vyměňují klient a server. Pomocí tohoto addonu jsem odchytával základní komunikaci mezi prohlížečem a IP telefonem a získával jaké informace si mezi sebou vlastně přesně vyměňují, abych je mohl napodobit. Ukázka originální komunikace z programu Live HTTP Headers:

http://147.229.147.114:9999/

POST / HTTP/1.1

Host: 147.229.147.114:9999

User-Agent: Mozilla/5.0 (X11; U; Linux x86_64; cs-CZ; rv:1.9.1.9) Gecko/20100402 Ubuntu/9.10 (karmic) Firefox/3.5.9

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/*;q=0.8*

Accept-Language: cs,en-us;q=0.7,en;q=0.3

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-2,utf-8;q=0.7,;q=0.7*

Keep-Alive: 300

Connection: keep-alive

Referer: http://147.229.147.114:9999/

Cookie: auth=59661b75000001e0

Content-Type: application/x-www-form-urlencoded

Content-Length: 91

encoded=root%3Affa9ecf16bc288abba8ef3d3d231eed8&nonce=59661b75000001e0&goto=Odeslat&URL=%2F

HTTP/1.1 200 OK

Server: Rapid Logic/1.1

MIME-version: 1.0

Date: Thu Jan 1 00:09:48 1970 GMT

Content-Type: text/html

Transfer-Encoding: chunked

---

http://147.229.147.114:9999/menu.htm

GET /menu.htm HTTP/1.1

Host: 147.229.147.114:9999

User-Agent: Mozilla/5.0 (X11; U; Linux x86_64; cs-CZ; rv:1.9.1.9) Gecko/20100402 Ubuntu/9.10 (karmic) Firefox/3.5.9

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/*;q=0.8*

Accept-Language: cs,en-us;q=0.7,en;q=0.3

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-2,utf-8;q=0.7,;q=0.7*

Keep-Alive: 300

Connection: keep-alive

Referer: http://147.229.147.114:9999/

Cookie: auth=59661b75000001e0

HTTP/1.1 200 OK

Server: Rapid Logic/1.1

MIME-version: 1.0

Date: Thu Jan 1 00:09:48 1970 GMT

Content-Type: text/html

Transfer-Encoding: chunked

Velkou výhodou tohoto programu je, že rychle a přehledně zobrazí probíhající komunikaci mezi klientem a telefonem, z které je například hned vidět, jaká data musíme odeslat telefonu pro úspěšné přihlášení k němu. Naopak dost zásadní nevýhodou je, že program tuto komunikaci již formátuje do lidsky přijatelné formy, takže nelze přesně poznat, jak mají být hlavičky naformátované při odesílání z modulu pro konfiguraci telefonu, aby je telefon správně zpracoval.

Wireshark

Jelikož zkoumání hlaviček pomocí nástroje Live HTTP Headers nevedlo ke kýženému výsledku, musel jsem využít sofistikovanějších nástrojů k vývoji aplikace. Tím se stal protokolový analyzér Wireshark.

Wireshark je multiplatformní open source program šířený pod licencí GNU GPL, sloužící mimo jiné k zachytávání komunikace probíhající na počítačové síti. Mezi jeho výhody patří široká možnost vyfiltrování a analyzování zachycených výsledků [29]. Pro moje potřeby jsem využil filtr pro zachycení veškeré komunikace na protokolu TCP a následnou analýzou zachycených paketů jsem získal jako výstup data:

POST / HTTP/1.1
User-Agent: Opera/9.80 (Windows NT 6.1; U; cs) Presto/2.5.24 Version/10.53
Host: 147.229.147.114:9999
*Accept: text/html, application/xml;q=0.9, application/xhtml+xml, image/png, image/jpeg, image/gif, image/x-xbitmap, */*;q=0.1*
Accept-Language: cs-CZ,cs;q=0.9,en;q=0.8
*Accept-Charset: iso-8859-1, utf-8, utf-16, *;q=0.1*
*Accept-Encoding: deflate, gzip, x-gzip, identity, *;q=0*
Referer: http://147.229.147.114:9999/
Cookie: auth=59661b7500068079
Cookie2: $Version=1
Connection: Keep-Alive, TE
TE: deflate, gzip, chunked, identity, trailers
Content-Length: 91
Content-Type: application/x-www-form-urlencoded

encoded=root%3Ac48d58c5884c2fe312eec548eb9eca93&nonce=59661b7500068079&goto=Odeslat&URL=%2FHTTP/1.1 200 OK
Server: Rapid Logic/1.1
MIME-version: 1.0
Date: Mon Jan 5 22:25:02 1970 GMT
Content-Type: text/html
Transfer-Encoding: chunked

Hned na první pohled je patrný rozdíl mezi již naformátovanými daty získanými z programu Live HTTP Headers a komunikací, tak jak šla, zachycenou programem Wireshark. Tou jsou chybějící volné řádky odesílané v hlavičce POST, které musí být vždy přítomny. Tyto získané informace již byly dostačující pro úspěšné napodobení komunikace v aplikaci.

3.3.3 Funkce pro komunikaci s telefonem

Komunikaci s telefonem obstáravají funkce uložené v pomocném souboru function_iptel.inc v adresáři /include, který se vkládá na začátku všech skriptů v aplikaci. Níže uvedu přehled nejdůležitějších funkcí obstarávajících chod aplikace.

Funkce otevri_socket($id_telefonu) přebírá jako parametr unikátní ID telefonu uloženého v databázi. Jako návratová hodnota funkce je odkaz na otevřený socket k telefonu, po kterém může probíhat další komunikace mezi aplikací a IP telefonem.

resource fsockopen (string $hostname [, int $port = -1 [, int &$errno [, string &$errstr [, float $timeout = ini_get("default_socket_timeout")]]]])

Otevře socket na adresu $hostname a port $port. Do ukazatelů &$errno a &$errstr se ukládají číslo a popis případné chyby vzniklé při volání funkce fsockopen(). Parametr $timeout udává v sekundách timeout pro spojení [7].

Funkce ziskani_nonce($ipadresa, $port, $id_telefonu) získává autentizační kód zasílaný telefonem klientovi zpět při prvním obdrženém požadavku ve formě cookies. Tento autentizační kód je poté nadále vyžadován při jakékoliv komunikaci s telefonem.

Pro úspěšné přihlášení se k telefonu je potřeba sestavit přesnou adresu získanou odchytnutím komunikace přes Wireshark. O to se stará funkce sestav_prihlaseni($nonce, $id_telefonu), která ze zadaného ID telefonu najde v databázi uživatelské jméno a heslo, a ze zadaného autentizačního kódu sestaví přesnou kopii přihlašovací hlavičky podle schématu odchyceného dříve. Uživatelské jméno a heslo se nepřenáší k telefonu v čistém formátu, ale jako md5 hash složený z kombinace uživatelského jména, hesla a autentizačního kódu.

Univerzálními funkcemi pro sestavení kompletní hlavičky pak jsou sestav_post($data, $ipadresa, $port, $nonce, $stranka), pro sestavení hlavičky metody POST protokolu HTTP, a sestav_get($ipadresa, $port, $nonce, $stranka). Obě dvě funkce přebírají jako parametr IP adresu a port telefonu, s kterým komunikujeme, autentizační kód získaný při přihlášení k telefonu a jméno stránky na serveru, se kterou chceme komunikovat. Funkce sestav_post() navíc ještě přebírá data, která chceme metodou POST odeslat na server.

Jako návratová hodnota je z obou dvou funkcí kompletně sestavená GET nebo POST hlavička připravená k odeslání na server.

Samotné odeslání požadavku na server pak zajišťuje funkce odesli_telefonu($hlavicka, $id_telefonu), která přebírá jako parametr sestavenou hlavičku, jedno jaké metody, a unikátní ID telefonu, s kterým zrovna komunikujeme.

Odeslání na server pak probíhá pomocí interní funkce PHP fputs(), což je alias pro

int fwrite (resource $handle , string $string [, int $length])

funkci na zapisování do souborů. Ta přebírá ukazatel na otevřený soubor určený k zápisu, v našem případě na socket otevřený funkcí fsockopen(), a řetězec $string, který se pokusí do něj zapsat. Nepovinný parametr $length udává po kolika bytech se zapisování automaticky zastaví [7].

Důležitou částí po odeslání požadavku telefonu je nastavení timeoutu před čtením návratové hodnoty.

bool stream_set_timeout (resource $stream , int $seconds [, int $microseconds = 0]) [7]

Bez tohoto nastavení totiž skript čeká až vyprší timeout nastavený v základu na serveru a komunikace s telefonem pak trvá dlouhou dobu. V našem přpadě funkce očekává jako parametr $stream socket otevřený funkcí fsockopen() a čas vypršení se nastaví jako součet hodnot $seconds a $microseconds.

Aplikace nepracuje s daty získanými z telefonu v online režimu, ale stahuje si při přihlášení a potom při každé změně odeslané telefonu konfigurační soubor IP telefonu a dále s ním pracuje v offline režimu. Výhoda tohoto řešení je omezení komunikace mezi aplikací a telefonem na minimum a tím zrychlení běhu celé aplikace. O stažení aktuálního konfiguračního souboru se stará funkce uloz_conf($hlavicka, $id_telefonu). Parametry jsou hlavička se žádostí o poslání konfiguračního souboru telefonu uvedeného v proměnné $id_telefonu.

Zpracování a zobrazení dat o telefonu v offline režimu zajišťuje funkce najdi_conf($stranka, $hledany, $novy_radek, $oddelovac), která ve stáhnutém konfiguračním souboru telefonu $stranka vyhledá data $hledany a vyextrahuje z nich hodnotu, která začíná vždy parametrem $oddelovac, v našem případě se jedná o znak „:“, a končí parametrem $novy_radek, „\r\n“.

Ukázka stránky z modulu pro konfiguraci IP telefonu viz Přílohy, obr. A.8

ZÁVĚR

V rámci bakalářské práce jsem provedl srovnání různých produktů pro navrhování a vyvíjení aplikací pro internet. Podrobněji jsem rozebral produkty, které jsem si vybral pro tvorbu svého projektu webové aplikace pro správu učebny. Zjistil jsem výhody a nevýhody jednotlivých programů a podle toho navrhl postup a řešení svého projektu.

Webová aplikace byla naprogramována a úspěšně vyzkoušena za použití operačního systému Windows XP, webového serveru Apache 2, MySQL verze 4 a PHP verze 5. Pro výstup pro prohlížeče bylo použito standardu HTML 4.01 Transitional a dodržení validního kódu otestováno pomocí online validátoru konsorcia W3C, kdy všechny stránky prošly testem bez přítomnosti varování o neplnění standardu.

Vývoj a programování zásuvného modulu pro konfiguraci IP telefonů Well 3130IF a Well 3195IF pak pokračoval za použití operačního systému Ubuntu 9.10 (Karmic Koala) jádro 2.6.31-21-generic, webového serveru Apache 2, MySQL verze 5.1.37-1ubuntu5.1 a PHP verze 5.2.10-2ubuntu6.4. Na stejné konfiguraci byla také odzkoušena multiplatformnost vyvíjené aplikace a úspěšně otestována bez jakýchkoliv chyb, či změn v chování programu, webová aplikace pro správu učebny vyvíjená původně pod operačním systémem MS Windows.

V průběhu programování aplikace však vycházely nové verze jazyka PHP, které přinášely řadu změn, a to i do funkcí používaných ve vyvíjené aplikaci. Přestože jsem se snažil tento vývoj reflektovat a upravovat i podle nejnověji vydaných specifikací, je možné, že v důsledku těchto změn, na jiné verzi než na mnou odzkoušené, nebude možno využívat všechny funkce webové aplikace tak, jak byly původně navrženy a naprogramovány.

Výsledkem této bakalářské práce je tedy kompletní webová aplikace pro správu učebny, obsahující základní možnosti práce s předměty, cvičeními a žáky nacházející se v aplikaci. Tato aplikace je dále rozšiřitelná o zásuvné moduly naprogramované podle předem daného schématu. Součástí aplikace je naprogramovaný zásuvný modul pro konfiguraci IP telefonů 3130IF a 3195IF firmy Well. Pomocí tohoto modulu je možno konfigurovat vybrané vlastnosti každého telefonu, jakožto i posílat hromadně jednotnou konfiguraci na více telefonů naráz. Veškeré vlastnosti jsou úspěšně otestovány na konfiguracích zmíněných výše a je možno je volně libovolně používat, upravovat či rozšiřovat.

Seznam použitých zdrojů

[1] DUBOIS, Paul. *MySQL profesionálně*. Vydání první. Praha : Mobil Media, 2003. 1071 s. ISBN 80-86593-41-X.

[2] STANÍČEK, Petr. *CSS Kaskádové styly : Kompletní průvodce*. Vydání druhé. Brno : Computer Press, 2003. 178 s. ISBN 80-7226-872-4.

[3] WELLING, Luke; THOMSONOVÁ, Laura. *PHP a MySQL : rozvoj webových aplikací*. Druhé vydání. Praha : SoftPress, 2004. 910 s. ISBN 80-86497-60-7.

[4] ZAJÍC, Petr. *Linux Software* [online]. 27.5.2004, poslední změna 4.2.2005 [cit. 2009-12-12]. PHP seriál. Dostupné z WWW: <http://www.linuxsoft.cz/php/>.

[5] LERDORF, Rasmus. *LERDORF.COM* [online]. 2000 [cit. 2009-12-12]. Bio. Dostupné z WWW: <http://lerdorf.com/bio.php>.

[6] *MySQL* [online]. c2009 [cit. 2009-12-12]. The world's most popular open source database. Dostupné z WWW: <http://www.mysql.com>.

[7] *PHP* [online]. c2001 [cit. 2009-12-12]. PHP: Hypertext Preprocessor. Dostupné z WWW: <http://www.php.net/>.

[8] *PHP : PHP: Hypertext Preprocessor* [online]. c1999, c2009 [cit. 2009-12-12]. The PHP License, version 3.01. Dostupné z WWW: <http://www.php.net/license/3_01.txt>.

[9] *PHP : PHP: Hypertext Preprocessor* [online]. c2001, last updated: Fri, 11 Dec 2009 [cit. 2009-12-12]. History of PHP. Dostupné z WWW: <http://www.php.net/manual/en/history.php.php>.

[10] *BuiltWith Technology Usage Statistics* [online]. 15.12.2009 [cit. 2009-12-17]. PHP Usage Statistics. Dostupné z WWW: <http://trends.builtwith.com/framework/PHP>.

[11] BERNERS-LEE, Tim; CONNOLLY, Daniel. *World Wide Web Consorcium (W3C)* [online]. June 1993 [cit. 2009-12-12]. Hypertext Markup Language (HTML). Dostupné z WWW: <http://www.w3.org/MarkUp/draft-ietf-iiir-html-01.txt>.

[12] BERNERS-LEE, T. *Internet Engineering Task Force* [online]. November 1995 [cit. 2009-12-12]. Hypertext Markup Language - 2.0. Dostupné z WWW: <http://www.ietf.org/rfc/rfc1866.txt>.

[13] RAGGETT, Dave. *World Wide Web Consorcium (W3C)* [online]. REC-html32. 1996, 14-Jan-1997 [cit. 2009-12-12]. HTML 3.2 Reference Specification. Dostupné z WWW: <http://www.w3.org/TR/REC-html32>.

[14] RAGGETT, Dave; LE HORS, Arnaud; JACOBS, Ian. *World Wide Web Consorcium (W3C)* [online]. REC-html401-19991224. 24 December 1999 [cit. 2009-12-12]. HTML 4.01 Specification. Dostupné z WWW: <http://www.w3.org/TR/1999/REC-html401-19991224/>.

[15] RAGGETT, Dave; LE HORS, Arnaud; JACOBS, Ian. *World Wide Web Consorcium (W3C)* [online]. REC-html40-19980424. 18 December 1997, revised on 24-Apr-1998 [cit. 2009-12-12]. HTML 4.0 Specification. Dostupné z WWW: <http://www.w3.org/TR/1998/REC-html40-19980424/>.

[16] *W3Schools Online Web Tutorials* [online]. c2010 [cit. 2009-12-12]. PHP File Upload. Dostupné z WWW: <http://www.w3schools.com/PHP/php_file_upload.asp>.

[17] *PHP : PHP: Hypertext Preprocessor* [online]. 2009 [cit. 2010-05-16]. PHP 5 ChangeLog. Dostupné z WWW: <http://www.php.net/ChangeLog-5.php#5.3.0>.

[18] VRÁNA, Jakub. *Root.cz : informace nejen ze světa Linuxu* [online]. 2005 [cit. 2010-05-16]. Vývoj PHP 6. Dostupné z WWW: <http://www.root.cz/clanky/vyvoj-php-6/>.

[19] *PHP : PHP: Hypertext Preprocessor* [online]. 2010 [cit. 2010-05-16]. PHP 5 ChangeLog. Dostupné z WWW: <http://www.php.net/ChangeLog-5.php#5.3.2>.

[20] LIE, Håkon Wium; BOS, Bert. *World Wide Web Consorcium (W3C)* [online]. REC-CSS1-20080411. 1996, revised 11 Apr 2008 [cit. 2010-05-16]. Cascading Style Sheets, level 1. Dostupné z WWW: <http://www.w3.org/TR/CSS1/>.

[21] *World Wide Web Consorcium (W3C) : Cascading Style Sheets Level 2 Revision 1 (CSS 2.1)* [online]. 2009 [cit. 2010-05-16]. Visual formatting model. Dostupné z WWW: <http://www.w3.org/TR/CSS21/visuren.html#propdef-display>.

[22] *World Wide Web Consorcium (W3C) : Cascading Style Sheets Level 2 Revision 1 (CSS 2.1*) [online]. 2009 [cit. 2010-05-16]. Appendix C. Changes. Dostupné z WWW: <http://www.w3.org/TR/CSS21/changes.html#changes>.

[23] BOS, Bert; ETEMAD, Elika J.; KEMPER, Brad. *World Wide Web Consorcium (W3C)* [online]. 2009 [cit. 2010-05-16]. CSS Backgrounds and Borders Module Level 3. Dostupné z WWW: <http://www.w3.org/TR/css3-background/>.

[24] *World Wide Web Consorcium (W3C)* [online]. 2008 [cit. 2010-05-16]. CSS Color Module Level 3 . Dostupné z WWW: <http://www.w3.org/TR/css3-color/>.

[25] *Firebug : Web Development Evolved* [online]. c2005 [cit. 2010-05-26]. What is Firebug?. Dostupné z WWW: <http://getfirebug.com/whatisfirebug>.

[26] http://tools.ietf.org/html/rfc2616

[27] BISOM, Robert. *Interval.cz* [online]. 2002-11-10 [cit. 2010-05-26]. Požadavky protokolu HTTP a jejich zpracování v PHP. Dostupné z WWW: <http://interval.cz/clanky/pozadavky-protokolu-http-a-jejich-zpracovani-v-php/>.

[28] JAKEL, Milan. *Interval.cz* [online]. 2002-5-29 [cit. 2010-05-26]. Stavové kódy a hlášení v odpovědi protokolu HTTP. Dostupné z WWW: <http://interval.cz/clanky/stavove-kody-a-hlaseni-v-odpovedi-protokolu-http/>.

[29] *Wireshark.org* [online]. 2004-09-05 [cit. 2010-05-26]. The Wireshark Wiki. Dostupné z WWW: <http://wiki.wireshark.org/>.

Seznam použitých zkratek

ASP Active Server Page

CSS Cascading Style Sheets

CSS1 Cascading Style Sheets Level 1

CSS2 Cascading Style Sheets Level 2

CSS2.1 Cascading Style Sheets Level 2 Revision 1

CSS3 Cascading Style Sheets Level 3

DF diagram Data Flow diagram

ER diagram Entitně-relační diagram

GNU GNU's Not Unix

GPL General Public License

HSL Hue, Saturation, Lightness

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

IAX2 Inter-Asterisk eXchange 2

ID IDentification

IEC International Electrotechnical Commission

IETF Internet Engineering Task Force

IIS Internet Information Services

IP Internet Protocol

ISO International Organization for Standardization

LAMP Linux Apache MySQL PHP

LAN Local Area Network

NAT Network Address Translation

PHP PHP: Hypertex Preprocessor

PHP/FI PHP/Forms Interpreter

QoS Quality of Service

RGB Red, Green, Blue

SGML Standard Generalized Markup Language

SIP Session Initiation Protocol

SQL Structured Query Language

STUN Session Traversal Utilities for NAT

TCP Transmission Control Protocol

VoIP Voice over IP

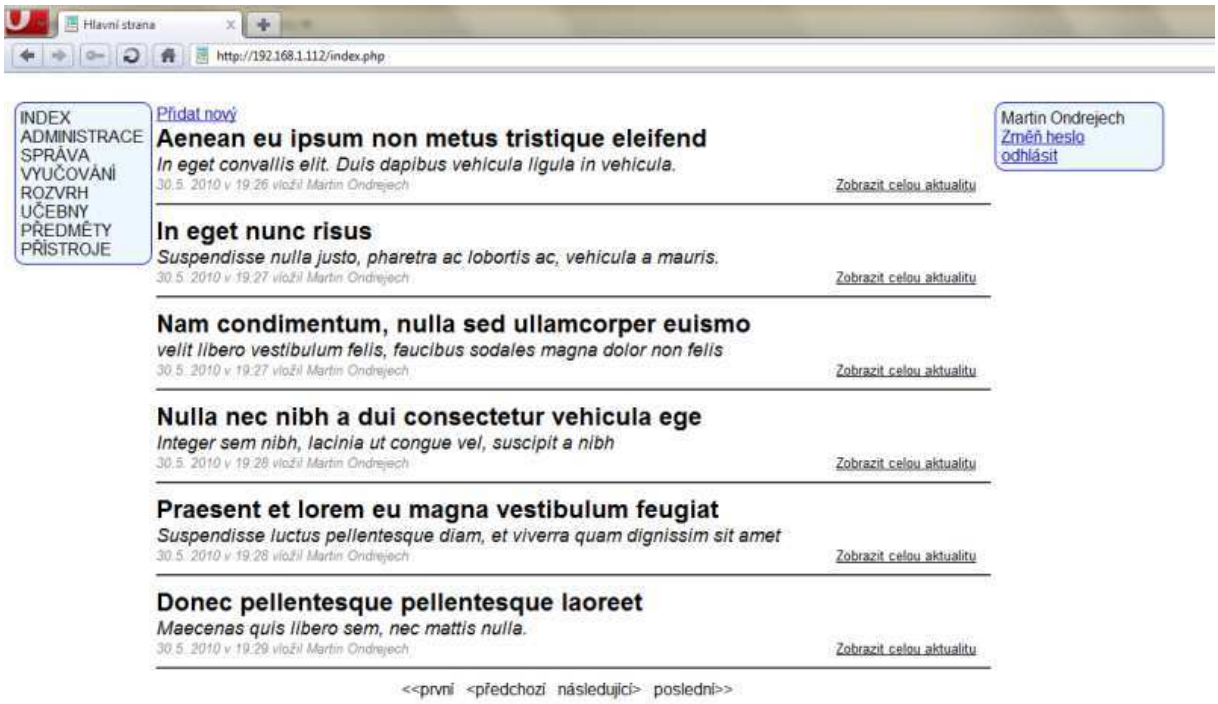
W3C World Wide Web Consorcium

WAMP Windows Apache MySQL PHP

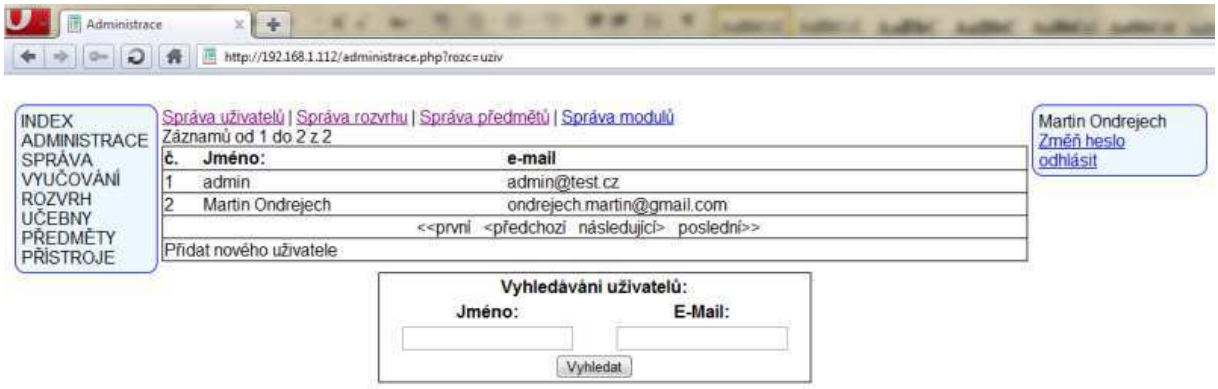
XHTML eXtensible HyperText Markup Language

XML eXtensible Markup Language

A UKÁZKY WEBOVÉ APLIKACE PRO SPRÁVU UČEBNY

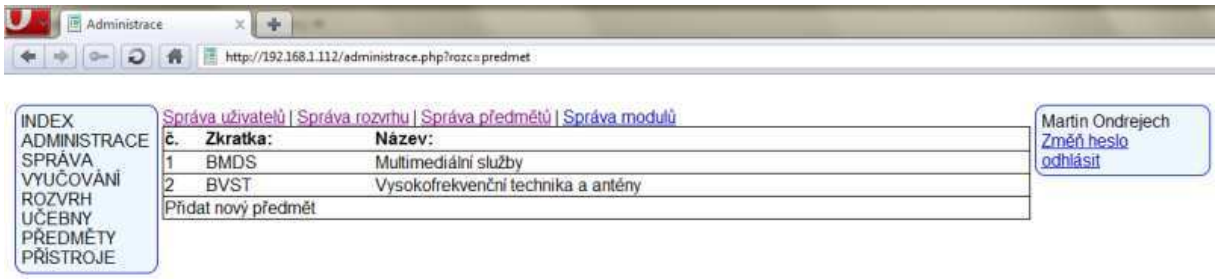


Obr. A.1: Hlavní strana

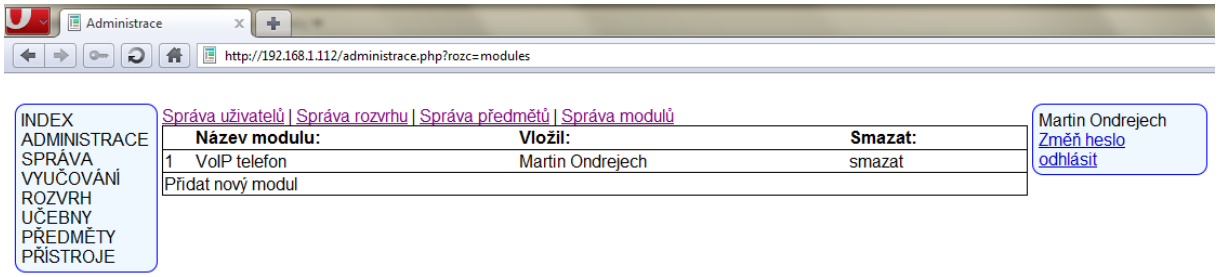


Obr. A.2: Administrace uživatelů

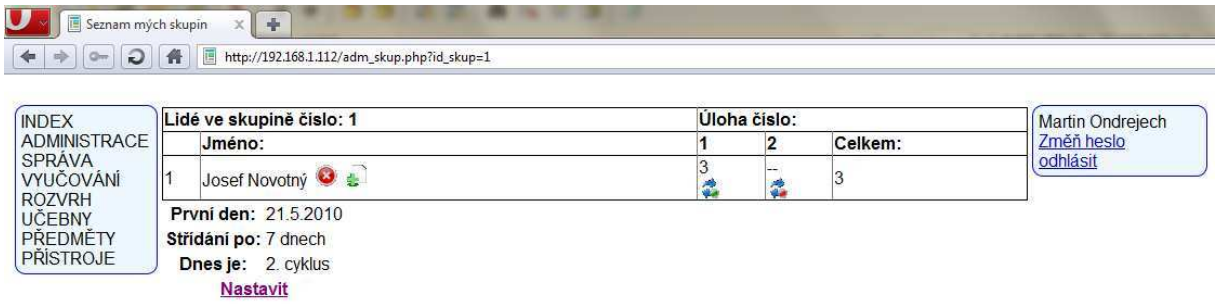
Obr. A.3: Administrace rozvrhu



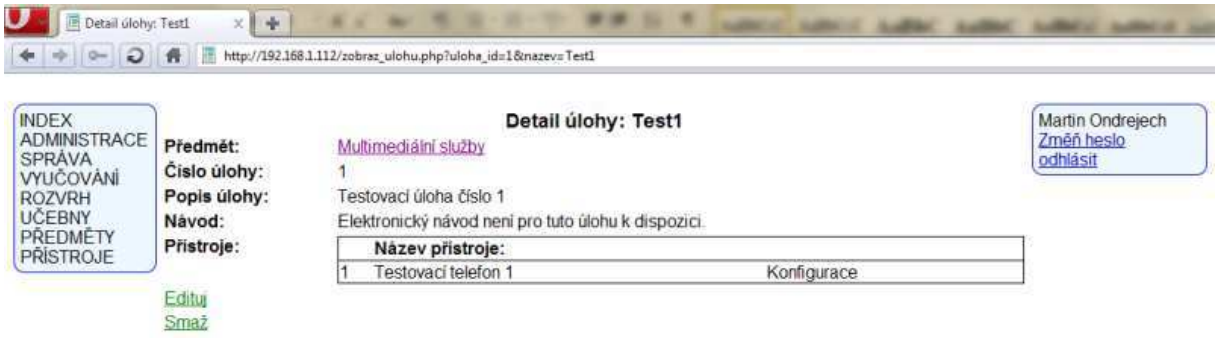
Obr. A.4: Administrace předmětů



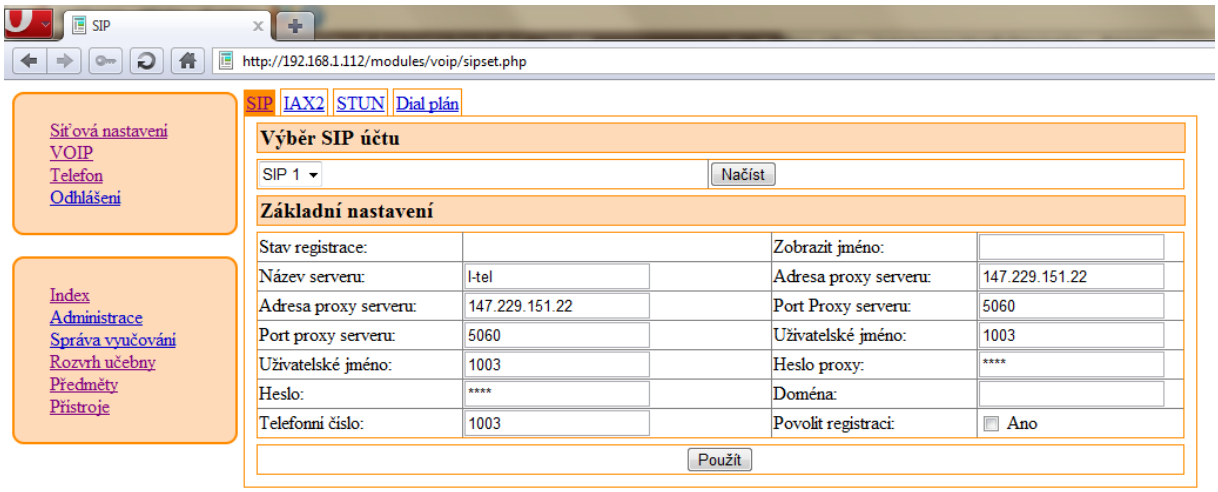
Obr. A.5: Administrace modulů



Obr. A.6: Správa vyučování



Obr. A.7: Detail laboratorní úlohy



Obr. A.8: Základní stránka modulu pro konfiguraci IP telefonu

B DVD

/aplikace – Webová aplikace pro správu učebny

/dokumenty – Text práce ve formátu PDF

/navod – Návod k webové aplikaci pro správu učebny.