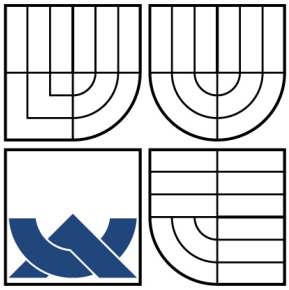
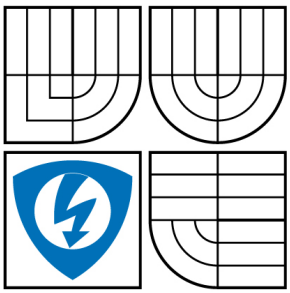


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

VÝUKOVÁ APLIKACE PRO INTERNETOVÉ PROSTŘEDÍ

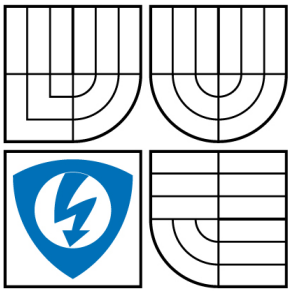
INSTRUCTION APPLICATION FOR INTERNET ENVIRONMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. RASTISLAV BARTÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. DAVID KUBÁNEK, Ph.D.
SUPERVISOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bartík Rastislav Bc.	***ID:***	50516
Ročník:	2	***Akademický rok:***	2007/2008

NÁZEV TÉMATU:

Výuková aplikace pro internetové prostředí

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte pomocí PHP a AJAX technologie univerzální aplikaci pro výukový portál univerzální sady skriptů, metod a nástrojů pro tvorbu výukových lekcí a testů znalostí. Projekt řešte ve spolupráci s pracovníky vývojového centra Honeywell v Brně.

DOPORUČENÁ LITERATURA:

[1] Lacko, Ľ.: PHP a MySQL - hotová řešení, CP Books, 2005.
[2] Kosek, J.: PHP podrobný průvodce, GRADA 1999.

Termín zadání:	11.2.2008	***Termín odevzdání:***	28.5.2008

Vedoucí práce: Ing. David Kubánek, Ph.D.

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

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Bc. Rastislav Bartík
Bytem: č.d. 369, 97221, Nitrianske Sučany
Narozen/a (datum a místo): 12.5.1984, Bojnice

(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 244/53, 60200 Brno 2
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Ing. Kamil Vrba, CSc.

(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - ☐ disertační práce
   - ☑ diplomová práce
   - ☐ bakalářská práce

   jiná práce, jejíž druh je specifikován jako .....

(dále jen VŠKP nebo dílo)

Název VŠKP: Výuková aplikace pro internetové prostředí
Vedoucí/školitel VŠKP: Ing. David Kubánek, Ph.D.
Ústav: Ústav telekomunikací
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

- ☑ tištěné formě - počet exemplářů 1
- ☑ elektronické formě - počet exemplářů 1

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - ☑ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

ABSTRAKT

Ajax je pomerne nová technológia vo svete tvorby internetových stránok, aj keď v skutočnosti ide o prepojenie viacerých programovacích jazykov, a to JavaScript, PHP a XML. Hlavnou výhodou je kontaktovanie strany servera v pozadí. Táto práca je zameraná na vytvorenie výukovej aplikácie pre internetové prostredie. V práci je riešenie užívatežského rozhrania a rozhrania pre administráciu.

KLÍČOVÁ SLOVA

AJAX, Javascript, PHP, XML, MySQL

ABSTRACT

Ajax is quite new technology in world of web pages creation. In fact, it consits of more programming techniques like Javascript, PHP and XML. The main advantage of this technique is contacting server side on the background. This paper is focused on creating educational application for internet environment. There is written about solution for user’s interface and interface for administration.

KEYWORDS

AJAX, Javascript, PHP, XML, MySQL

BARTÍK, R. *Výuková aplikace pro internetové prostředí.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. XY s. Vedoucí diplomové práce Ing. David Kubánek, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Výuková aplikace pro internetové prostředí jsem vypracoval samostatně pod vedením vedoucího semestrálního projektu a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedeného semestrálního projektu dále prohlašuji, že v souvislosti s vytvořením tohoto projektu 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 ..... ……..

podpis autora

PODĚKOVÁNÍ

Ďakujem vedúcemu diplomovej práce Ing. Davidovi Kubánkovi za užitočnú pomoc a rady pri spracovaní diplomovej práce. Tak isto ďakujem pánovi Ing. Ondřejovi Pavelkovi z firmy Honeywell za pomoc pri realizácii projektu.

ZOZNAM SKRATIEK

AJAX	*Asynchronous JavaScript and XML*
DOM	*Document Object Model*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protokol*
MDB	*Access Database (Microsoft)*
SQL	*Structured Query Language*
PHP	*PHP: Hypertext Preprocessor*
TCP	*Transmission Control Protocol*
URL	*Uniform Ressource Locator*
WWW	*World Wide Web*
XML	*eXtensible Markup Language*

OBSAH

ZOZNAM OBRÁZKOV

ÚVOD

Ajax je pomerne nová technológia vo svete tvorby internetových stránok, aj keď v skutočnosti ide len o prepojenie viacerých programovacích jazykov, a to JavaScript, PHP a XML. Hlavná výhoda použitia AJAXu je, že užívateľ je schopný pracovať na stránke a tá môže komunikovať asynchrónne so serverom bez toho, aby sa stránka musela znovu načítať. Čiže hlavná výhoda je v tom, že dátový prenos nie je až tak zaťažený ako keby sa mala stiahnuť celá stránka znovu.

Táto práca je zameraná na vytvorenie výukovej aplikácie pre internetové prostredie. Databázu slov, lekcií, výukových textov a testov je možné dopĺňať a mazať pomocou aplikácie. Databáza beží v prostredí MySQL. S touto databázou komunikuje Javascript s PHP skriptom cez prvok XMLHttpRequest. Ten slúži ako medziprvok medzi týmito dvomi technológiami.

Realizácia práce sa zameriava na úlohu so spracovaním textu a jeho zobrazením, úlohu s výberom možností správneho slovného tvaru, úlohu na preklad viet a úlohu s otáčacími kartičkami vo forme pexesa. Priebeh celého procesu odoslania požiadavku, načítania a aktualizácie stránky je prakticky popísaný názorne na zadaných úlohách. Pri prvej úlohe je podrobnejšie popísaná klientská aj serverová časť. Pri ďalších len zásadné zmeny.

Administratívna časť je zameraná na zadávanie a mazanie lekcií, správu emailu lektora a prístupových údajov. Aplikácia je zabezpečená heslom a nie je možné pre užívateľov len študentskej časti pristupovať aj do časti administrátorskej.

Kompatibilita bola testovaná s rôznymi prehliadačmi a časť zdrojových kódov je prispôsobená paralelne ku viacerím prehliadačom.

1. HTTP PROTOKOL A AJAX

Hypertext Transfer Protokol (HTTP) je protokol aplikačnej vrstvy, vlastný prenos dát zaisťuje protokol nižšej vrstvy napr. TCP. Používa sa pre distribuované hypermediálne informačné systémy. Pre službu WWW sa používa od roku 1990. Slúži na výmenu HTML dokumentov medzi klientom a serverom. WWW server sa často nazýva aj HTTP serverom.

Klient je kombinácia hardwaru a softwaru, na ktorú sa dotazuje server. Na internete sa klienti chovajú ako koncoví užívatelia, ktorí prijímajú dáta pomocou programov internetových prehliadačov. Počítače na internete sú najprv v úlohe klienta, prijímajú dáta, neskôr môžu v prípade, že ďalej odovzdávajú informácie, plniť úlohu servera. Stranou klienta sa myslí taká akcia, ktorú môžeme previesť na lokálnom počítači bez podpory servera.

Server je kombinácia hardwaru a softwaru, ktorá dáva k dispozícii dáta. Na internete sú servery, uzly a počítače, ktoré ďalej posielajú a riadia informácie (text, grafiky). Za každou WWW adresou, ktorej obsah si prečítame v prehliadači, sa skrýva server, ktorý rozosiela dáta.

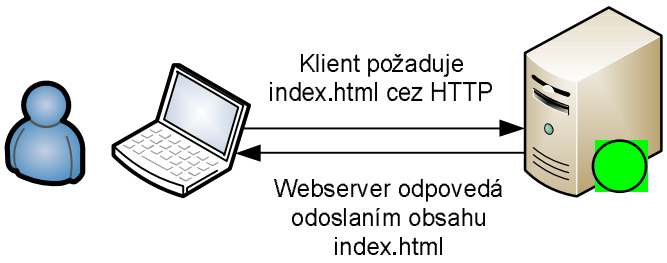
URL (Uniform Ressource Locator) je internetová adresa, ktorá jednoznačne identifikuje dátovú stránku. URL sa skladá z protokolu, domény, cesty a súboru.
Najnovšia verzia HTTP protokolu je 1.1. Trvalé spojenie v predchádzajúcich verziách protokolu bolo pre každé URL nadviazované zvláštne TCP spojenie. V prípade obrázkov vložených v HTML stránke nadviazoval klient v krátkom časovom intervale niekoľko spojení na ten istý originálny server. To spôsobovalo záťaž serveru a preťaženie internetových liniek.

Verzia HTTP 1.1 zavádza možnosť trvalého spojenia. V rámci tohoto spojenia klient posiela všetky svoje požiadavky na server a server mu potom posiela svoje odpovede. Je definovaný mechanizmus ukončenia spojenia. Vo verzii HTTP 1.1 je trvalé spojenie chápané ako implicitné pre všetky HTTP spojenia. Klient teda môže predpokladať, že server udržuje trvalé spojenie a naopak. Spojenie ostáva dovtedy, kým klient alebo server spojenie neukončia. Spojenie sa ukončuje vyslaním hlavičky Connection : close, požiadavka obsahujúca túto hlavičku je v danom spojení posledná. Ak je signalizované ukončenie spojenia zo strany servera, nesmie už klient posielať ďalšie požiadavky. Všetky správy definované v trvalom spojení musia mať definovanú dĺžku v hlavičke Message-length.

1.1 HTTP a HTML

Protokol HTTP je podporovaný všetkými prehliadačmi a veľmi dobre si plní svoju funkciu prijímania webového obsahu. Protokol HTTP sa používa pri každom požiadavku vo webovom prehliadači. Štandardným typom súboru, ktorý sa používa na zobrazenie stránok v prehliadači je HTML (HyperText Markup Language). HTML nebolo postavené na tvorbu komplexných webovýh aplikácií s interaktívnym obsahom, alebo užívateľsky prijateľným rozhraním. Ak je potreba otvoriť ďalšiu HTML stránku cez HTTP, musí sa inicializovať obnovenie celej stránky, ktorá je statická a musí na danej adrese existovať. S týmito vlastnosťami sa HTML nedá použiť na vyššie požiadavky užívateľov.

Na obrázku 1.1 je znázornená komunikácia prenosu, keď užívateľ požaduje webovú stránku z Internetu cez protokol HTTP.



Obr. 1.1: Jednoduchá HTTP požiadavka

Keďže je HTTP-HTML komunikácia veľmi limitovaná v tom, že dokáže prijímať len statický obsah stránok, boli vyvinuté ďalšie technológie.

Technológie na strane klienta – umožňujú užívateľovi zobraziť viac zaujímavého obsahu ako len statické dokumenty. Väčšinou tieto dokumenty majú stále príponu HTML a ich obsah je z časti statický.

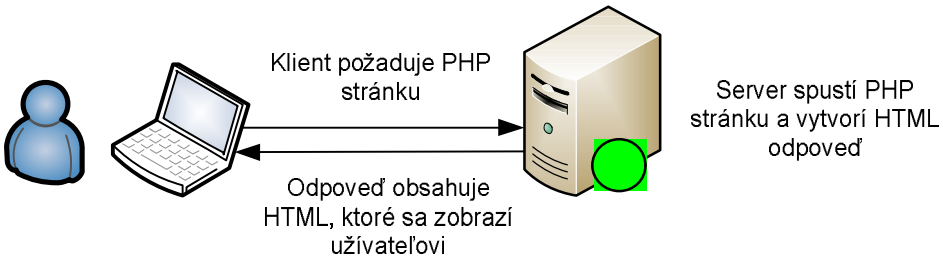
Technológie na strane servera – sú tie, ktoré dokážu používať určitú logiku a tým vytvárať web stránky počas chodu.

PHP a iné technológie na strane serveru

Technológie na strane servera umožňujú používať na webservery komplexné kalkulácie, objektovo orientované programovanie, prácu s databázami, čo je oveľa viac, ako odosielanie len statického HTML.

PHP je jedna z technológií, ktorá sa dá použiť na implementáciu logických operácií na strane servera. Existujú však aj iné technológie ako ASP.NET, Java Server Pages, Perl, ColdFusion, Ruby on Rails a iné. Každá z nich sprístupňuje programátorom funkčnosť na strane servera.

PHP však nie je len technológia ale aj programovací jazyk, ktorým sa dá vytvoriť PHP skript. Obrázok 1.2 zobrazuje požiadavku na súbor *index.php*. V tomto prípade namiesto odoslania obsahu *index.php*, server spustí *index.php* a pošle výsledky tohoto procesu. Tieto výsledky musia byť v HTML alebo v inom jazyku, ktorým bude klient rozumieť.



Obr. 1.2: Klient požaduje PHP stránku

S PHP je možné vytvoriť stránky, ktoré sú schopné komunikovať s databázou, ale internetový prehliadač stále zobrazuje statický dokument – ktorý nie je moc užívateľsky efektívny. Preto boli vytvorené technológie, ktoré bežia na strane klienta a robia tak zo statického obsahu niečo oveľa zaujímavejšie.

Javascript a iné technológie na strane klienta

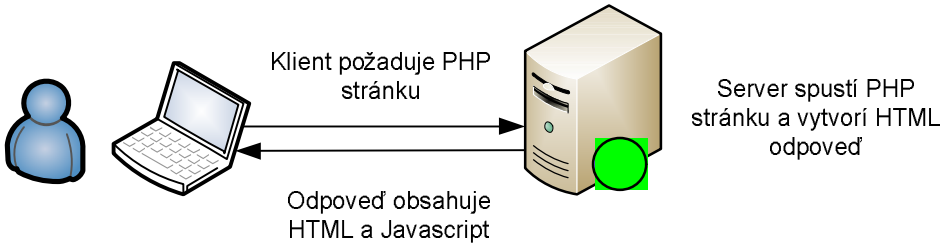
Javascript je skriptovací jazyk, ktorého kód je zapísaný ako prostý text a vložený do stránok HTML za účelom ich vylepšenia. Tento jazyk je podporovaný všetkými modernými prehliadačmi, pričom nie je nutné do systému inštalovať nové komponenty. Je objektovo orientovaný a nie je nijak kompilovaný, takže sa nehodí na náročné operácie na strane klienta. Hodí sa najmä na operácie typu kontrolovanie zadaných údajov pred odoslaním na server, správny tvar emailu a pod.

Z ďalších populárnych technológií pre tvorbu funkcionality na strane klienta sú Java aplety a Macromedia Flash. Java aplety sa píšu v jazyku Java, ktorý sa spúšťa pomocou software Java Virtual Machine a ten je potrebné do počítača nainštalovať. Java aplety sú nepochybne istou cestou k vývoju výkonnejších aplikácií, ale pretože využívajú mnoho systémových prostriedkov, stratili svoju popularitu.

Macromedia Flash ponúka veľmi užitočné nástroje pre tvorbu animácií a grafických efektov a v tejto oblasti je na webe určitým štandardom – takisto ako Java aplety – vyžaduje inštaláciu plug-inu do prehliadača. Keď sa HTML skombinuje so serverovými a klientskými technológiami, je možné vytvárať veľmi výkonné riešenia.

Spojenie technológií

Časť trhu chce mať vo webových klientoch výkonnejšiu funkcionalitu, ale bez použitia Flasha, Java apletov alebo iných technológií, ktoré sa pre niektoré účely nehodia pre ich požiadavky na systém alebo nekompatibilitu s užívateľským prostredím. V týchto prípadoch vývojári vytvárajú weby a webové aplikácie v HTML, Javascripte a PHP (alebo inej serverovej technológii). V takom prípade je proces s požiadavkom od klienta zobrazený na obrázku 1.3 – požiadavok HTTP a odpoveď zostavená z HTML a JavaScriptu za použitia PHP.

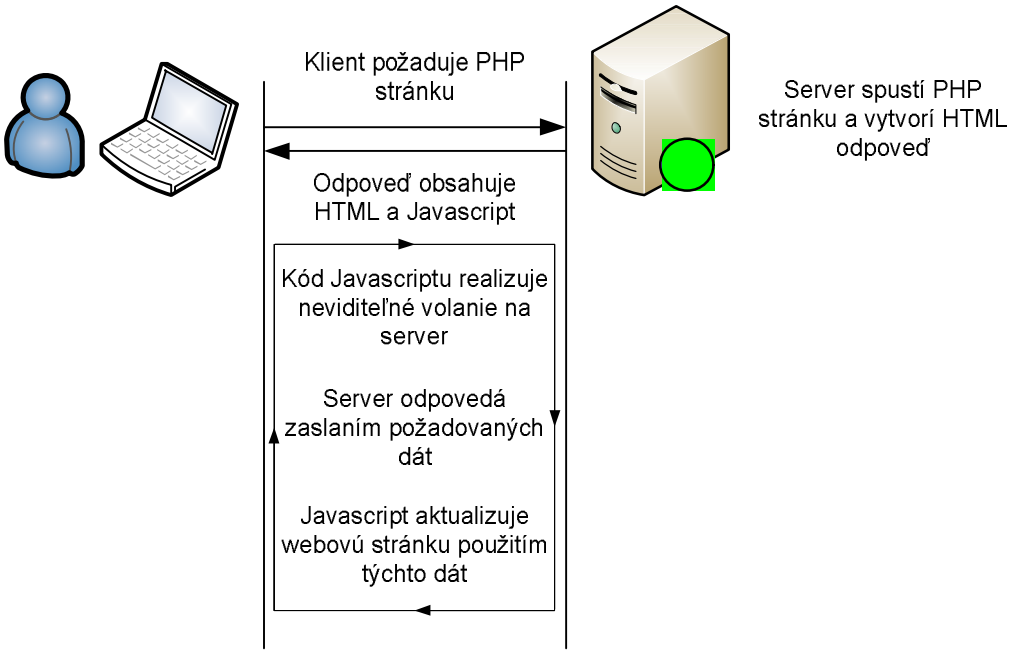


Obr. 1.3: HTTP, HTML, PHP a Javascript

Skrytým problémom je v tomto prípade tá skutočnosť, že zakaždým, keď klient potrebuje ďalšie dáta zo serveru, musí byť kvôli znovunačítaniu stránky vytvorený nový požiadavok HTTP, čo samozrejme brzdí aktivitu užívateľa. Znovunačítanie stránok je súčastným problémom, kôli ktorému tu je AJAX.

1.5 AJAX a jeho štruktúra

Ajax je skratkou Asynchronous Javascript and XML - čo sa dá pochopiť ako vylepšený Javascript o funkciu volania servera v pozadí a podľa potreby tak získavať požadované dáta. Týmto spôsobom je možné aktualizovať niektoré časti stránky bez nutnosti opätovného načítania celého obsahu. Na obrázku 1.4 je znázornené, čo sa deje, keď je typická webová stránka s technológiou AJAX vyžiadaná užívateľom.



Obr. 1.4: Typické volanie AJAXu.

Čiže AJAX ako taký by sa dal popísať tromi časťami:

Javascript - je základnou zložkou AJAXu ktorá umožňuje budovať funkcionalitu na strane klienta. Vo svojich funkciách manipuluje s časťami HTML stránky a používa sa Document Object Model (DOM), ktorý má schopnosť manipulovať (vytvárať, modifikovať, parsovať, vyhľadávať) s dokumentmi, ktoré sú založené na XML (vrátane HTML)

Objekt XMLHttpRequest - umožňuje Javascriptu asynchrónne komunikovať so serverom tak, že zatiaľ čo prebieha komunikácia na pozadí, môže užívateľ pokračovať v práci. Táto komunikácia znamená vytvorenie HTTP požiadavku na súbor alebo skript umiestnený na servery. Tieto požiadavky sa tvoria jednoducho a nespôsobujú žiadne problémy s firewallom.

Strana serveru - je tu potrebná technológia na spracovanie požiadavkov, ktoré prichádzajú od klienta. V projekte je použité PHP.

Pre komunikáciu medzi klientom a serverom potrebujú obe strany spôsob, ako vkladať dáta a ako im porozumieť. Klientský skript, ktorý pristupuje k serveru, môže prostredníctvom GET a POST poslať dvojice meno-hodnota. Tieto hodnoty sa dajú jednoducho prečítať na strane servera. Tento serverový skript potom zašle odpoveď späť pomocou HTTP, ale na rozdiel od normálnej webovej stránky bude odpoveď vo formáte, ktorý bude parsovaný Javascriptom v klientovi. Doporučeným formátom je

XML, ktorého výhodou je veľké rozšírenie a veľa knižníc, pomocou ktorých je možné s XML manipulovať. Samozrejme je možné poslať zo servreru hocijakú inú odpoveď, ktorú je Javascript svojim syntaxom schopný spracovať. Preto aj v tomto projekte je použitý výstup zo strany servera v rôznych podobách podľa použitej databázy - čiže aj neformátovaný text, ktorý je oddelený buď stranou servera alebo klienta.

1.6 Výhody a nevýhody AJAXu

Výhody

+ Umožňuje vytvárať lepšie a prístupnejšie weby a webové aplikácie
+ Využíva súčastné technológie
+ Využíva súčastné znalosti vývojárov
+ Pri požiadavkoch zo servera prenáša menej dát

Nevýhody

- Pretože adresy stránok sa pri práci s nimi nemenia, nie je možné stránky AJAX jednoducho bookmarkovať
- Vyhľadávače nemusia byť schopné indexovať všetky časti stránky s aplikáciou
- Stlačenie tlačítka Späť nemá rovnaký efekt ako pri klasických webových aplikáciách, pretože všetky akcie sa odohrávajú na jednej stránke. Je potrebné použiť túto funkciu iným spôsobom
- Javascript môže byť na strane klienta vypnutý, čo má za následok nefunkčnosť Ajaxovej aplikácie. Je mať potrebné pripravené aj náhradné riešenie

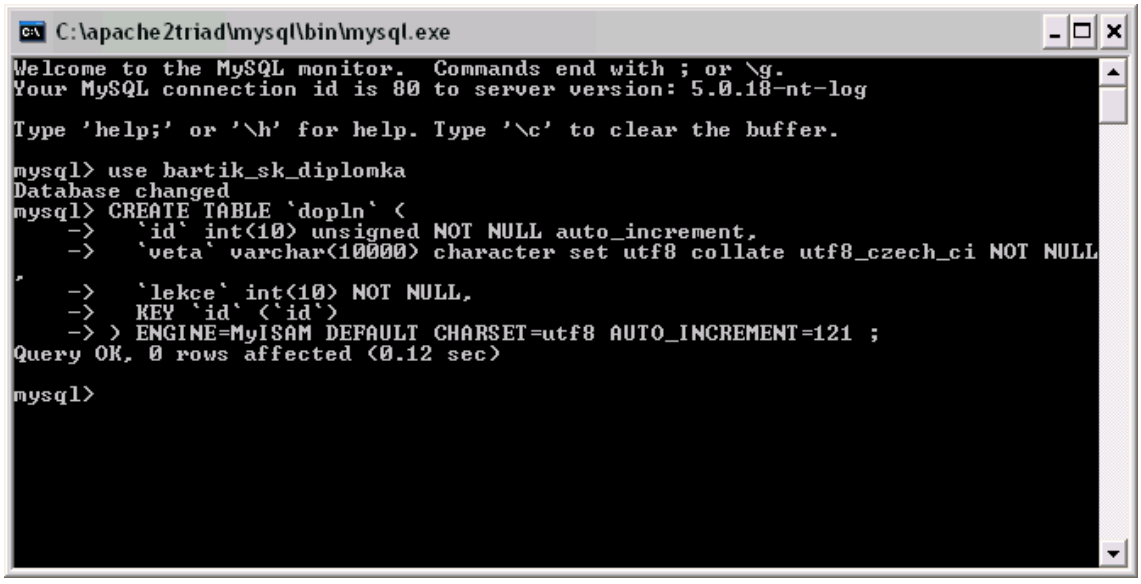
2 VYTVORENIE PROSTREDIA APLIKÁCIE

2.1 HTTP a MySQL server

Ako HTTP server bol použitý voľne šíriteľný software Apache (www.apache.org), ktorý má implementovanú priamu podporu PHP a MySQL databáz. Po nainštalovaní bolo potrebné nastaviť root účet na MySQL. Tvorenie databázy je potom na užívateľovi, či zvolí príkazový riadok alebo grafické rozhranie.

2.2 MySQL databáza

V databáze je potrebné vytvoriť 8 zložiek, kde budú uložené slovíčka, texty, názvy lekcií a testov. Je potrebné mať vytvorené aj zložky na ukladanie výsledkov a hlavných prístupových informácií. Pre rýchlu inštaláciu tabuľky je vhodné spustiť pôvodný terminál mysql a vložiť cez clipboard vopred pripravný SQL súbor dp.sql



Obr.2.1: Rozhranie terminálu MySql

Príklad vloženia TABLE heslo, ktorý bude použitý v úlohe na dopĺňanie správneho slovného tvaru

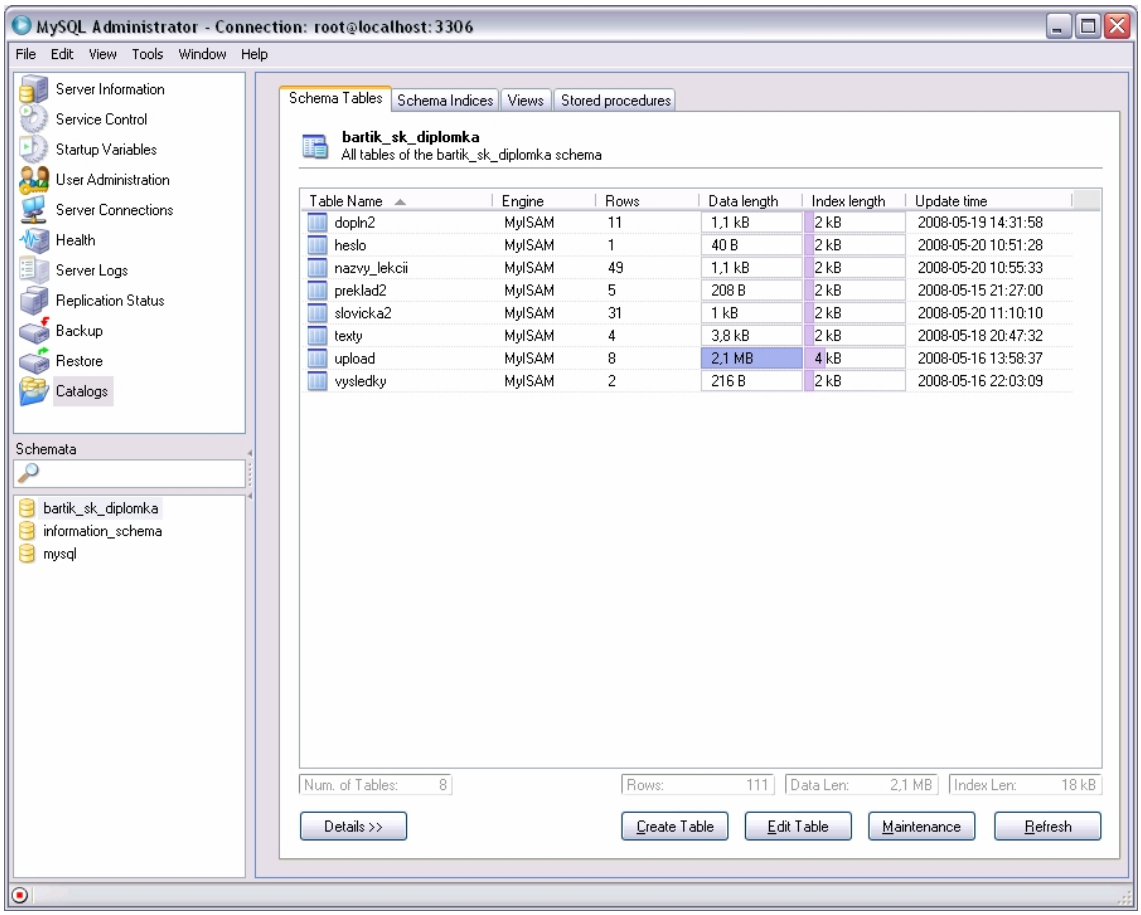
```
CREATE TABLE `heslo` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `heslo` varchar(45) character set utf8 collate
utf8_czech_ci NOT NULL,
```

```
  `folder` varchar(45) character set utf8 collate
utf8_czech_ci NOT NULL,
  `email` varchar(45) NOT NULL,
  KEY `id` (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=4 ;
```

Príklad zadávania hodnôt do vytvoreného TABLE heslo.

```
INSERT INTO `heslo` (`id`, `heslo`, `folder`, `email`)
VALUES (1, 'heslo', 'folder', 'lektor@email.com');
```

Keď sú vygenerované dáta spustené v príkazovom riadku, je potom vhodnejšie použiť grafické rozhranie pre lepšiu prehľadnosť. Na toto poslúži MySQL Administrator. Je potrebné sa do neho prihlásiť tak isto ako do príkazového riadku. Výhodou tohoto klienta je, že do databázy je možné vkladať rýchlejšie, prehľadnejšie a je tu veľa iných funkcií vrátanie zálohovania dát, vzdialeného pripojenia a uploadu databázy. Obr. 2.2 zobrazuje výstup použitej databázy pre výukovú internetovú aplikáciu.



Obr. 2.2: Prostredie programu MySQL administrator

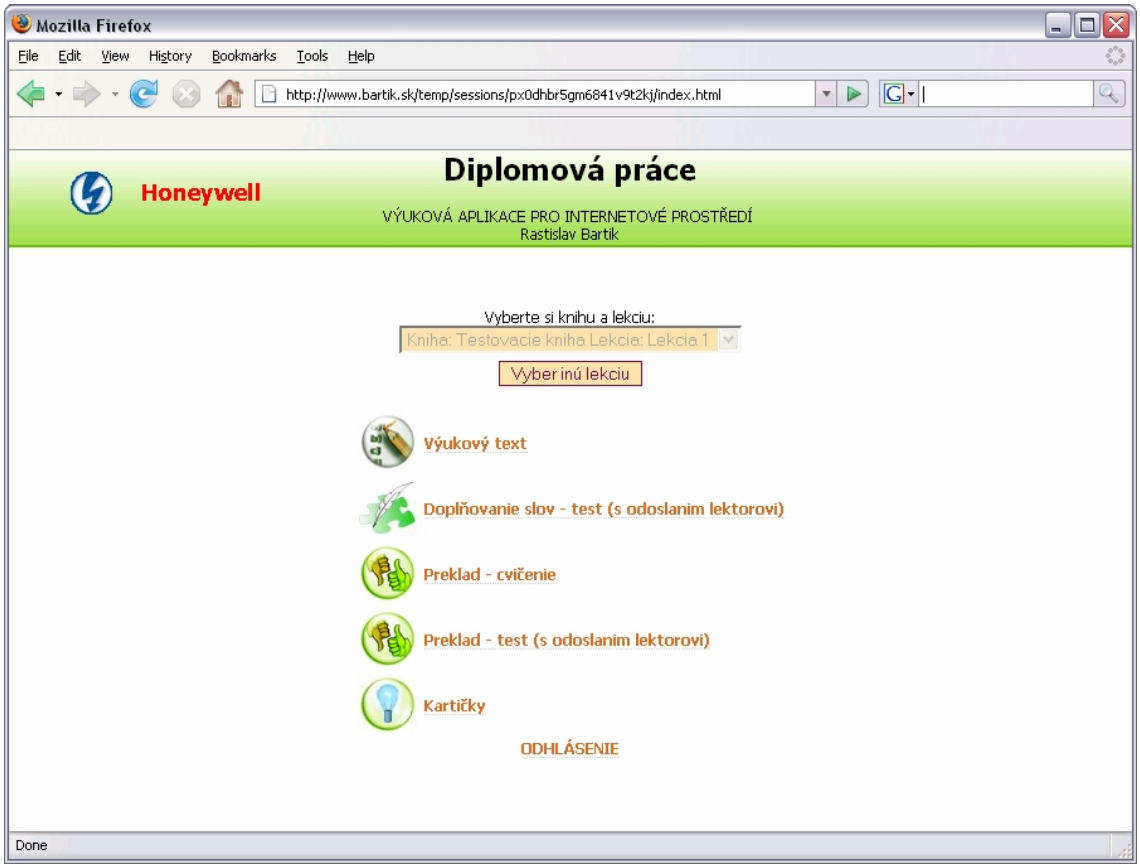
K už existujúcej databáze bolo potrebné vytvoriť ešte jeden TABLE s výsledkami, do ktorých sa budú zapisovať hodnoty odoslané užívateľmi pri jednotlivých úlohách. TABLE má mať 4 stĺpce - heslo, meno, text a lekce. Vytvorí sa nasledujúcim príkazom:

```
CREATE TABLE vysledky
     (
     id  INT NOT NULL,
     heslo  CHAR(45),
     meno  CHAR(45),
     text  CHAR(1000),
     lekce  CHAR(10),
     );
```

3. VÝUKOVÁ APLIKÁCIA V INTERNETOVOM PROSTREDÍ AJAX

3.1. Rozhranie študenta

V rozhraní pre študenta sa po prihlásení zobrazí výber kníh a lekcií. Zobrazia sa len tie lekcie, ktoré lektor sprístupnil. Po zvolení lekcie sa odošle požiadavok na server a sú prijaté len tie cvičenia, ktoré patria do zvolenej lekcie. Po zvolení cvičenia sa aplikácie po stlačení tlačítka úvod vráti vždy na túto úvodnú stránku na obr.3.1.



Obr. 3.1: Úvodná stránka študentského rozhrania

3.1.1 Práca s článkom - klientská strana

Pri riešení zadaných úloh je na strane klienta jednoduchá stránka HTML, ktorá obsahuje pevne stanovené prvky DOMu a odkazuje sa na Javascript, ktorý riadi všetko ostatné. Na začiatku každej HTML stránky je funkcia, ktorá sa spustí hneď po načítaní stránky celý proces Javascriptu. Ukážka HTML stránky je na obrázku 3.2.

```
 1
 2
 3 <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
 4
 5    <head>
 6      <meta http-equiv="content-type" content="text/html; charset=utf-8" />
 7      <link rel="stylesheet" type="text/css" href="style.css" />
 8    <script type="text/javascript" src="soundmanager2.js"></script>
 9 <script type="text/javascript">
10
11 soundManager.url = 'soundmanager2.swf'; // override default SWF url
12 soundManager.debugMode = false;
13 soundManager.consoleOnly = false;
14
15 soundManager.onload = function() {
16   soundManager.createSound('mySound0','sample/bass.mp3');
17
18 }
19
20 </script>
21
22
23
24    </head>
25 <body onload="lekce()">
26 <a href="index.html" style="position: absolute; right: 40px; top: 10px"">
27 <img src="img/uvod.gif"></a><script src="time.js"></script>
28 <script src="wz_tooltip.js"></script>
29 Vyberte si výukový text: <div name="divlekcie" id="divlekcie"></div>
30 <div name="outputText" id="outputText"></div>
31
32 </body>
33 </html>
```

Obr. 3.2: Ukážka HTML stránky

HTML stránka má dva DOM prvky, ktoré nemajú na začiatku žiadny obsah. Na 25. riadku ukážky sa spúšťa hneď po načítaní funkcia lekce() ktorá je súčasťou skriptu time.js ktorý je na 27 riadku. Ďalej sa načíta skript wz_tooltip ktorý bude potrebný na zobrazenie poznámky ku slovu. Skriptová časť stránky načíta externý súbor soundmanager2.js. Tento skript je potrebný na prehrávanie mp3 súborov, ktoré obsahujú výslovnosť alebo preklad daného slova. Keďže javascript nie je schopný prahrať mp3 súbory, je použitá táto funkcia z Flash. Soundmanager2 používa externý súbor soundmanager2.swf. Do funkcie prehrávania sa odosiela názov súboru a adresa mp3.

Úlohou bolo načítať text v cudzom jazyku, kde budú určité slová opatrené poznámkami a tieto informácie sa zobrazia pri nabehnutí myšou na tento text. Po klinutí na slovo sa prehrá mp3 s výslovnosťou. V prípade tlače sú tieto slová zoradené pod textom a vytlačené.

Čiže hneď na začiatku Javascriptu je potrebné zabezpečiť prenos dát medzi Javascriptom a PHP, keďže texty v cudzom jazyku a názvy lekcií sú uložené v databáze. Bude sa tak diať cez prvok XMLHttpRequest ktorý je schopný túto komunikáciu

zabezpečiť. Je použitý pri všetkých dotazoch na serverovú časť skriptov. V prvej časti Javascriptu time.js sa o to postará funkcia getHTTPObject na obrázku 3.3

```
 1 function getHTTPObject(){
 2    if (window.ActiveXObject) return new ActiveXObject("Microsoft.XMLHTTP");
 3    else if (window.XMLHttpRequest) return new XMLHttpRequest();
 4    else {
 5       alert("Your browser does not support AJAX.");
 6       return null;
 7    }
 8 }
```

Obr. 3.3: Funkcia getHTTPObject()

Funkcia zisťuje, o aký prehliadač sa jedná. V prípade IE6 a starších sa použije ActiveXObject, v prípade že je prehliadač IE7, Opera alebo Mozilla, vyberie sa druhá možnosť. Ak ani jeden z prvkov neuspeje, zobrazí sa hláška o nekompatibilite prehliadača a stránka nebude funkčná. Na túto funkciu sa obracajú iné funkcie vždy, keď potrebujú dáta zo strany servera. Prípadom toho je aj funkcia lekce(), ktorá sa spustí hneď po načítaní stránky. Je potrebné zistiť pre texty v databáze, ku ktorým lekciám patria a aký je ich názov. Funkcia je zobrazená na obr.3.4

```
78 function lekce(){
79     httpObject = getHTTPObject();
80     var RandVal = Math.random(); //refresh zobrazenie pre Internet Explorer
81     if (httpObject != null) {
82         httpObject.open('POST', 'time.php?rand='+RandVal, true);
83         httpObject.send(null);
84         httpObject.onreadystatechange = spracujlekcie;
85     }
86 }
```

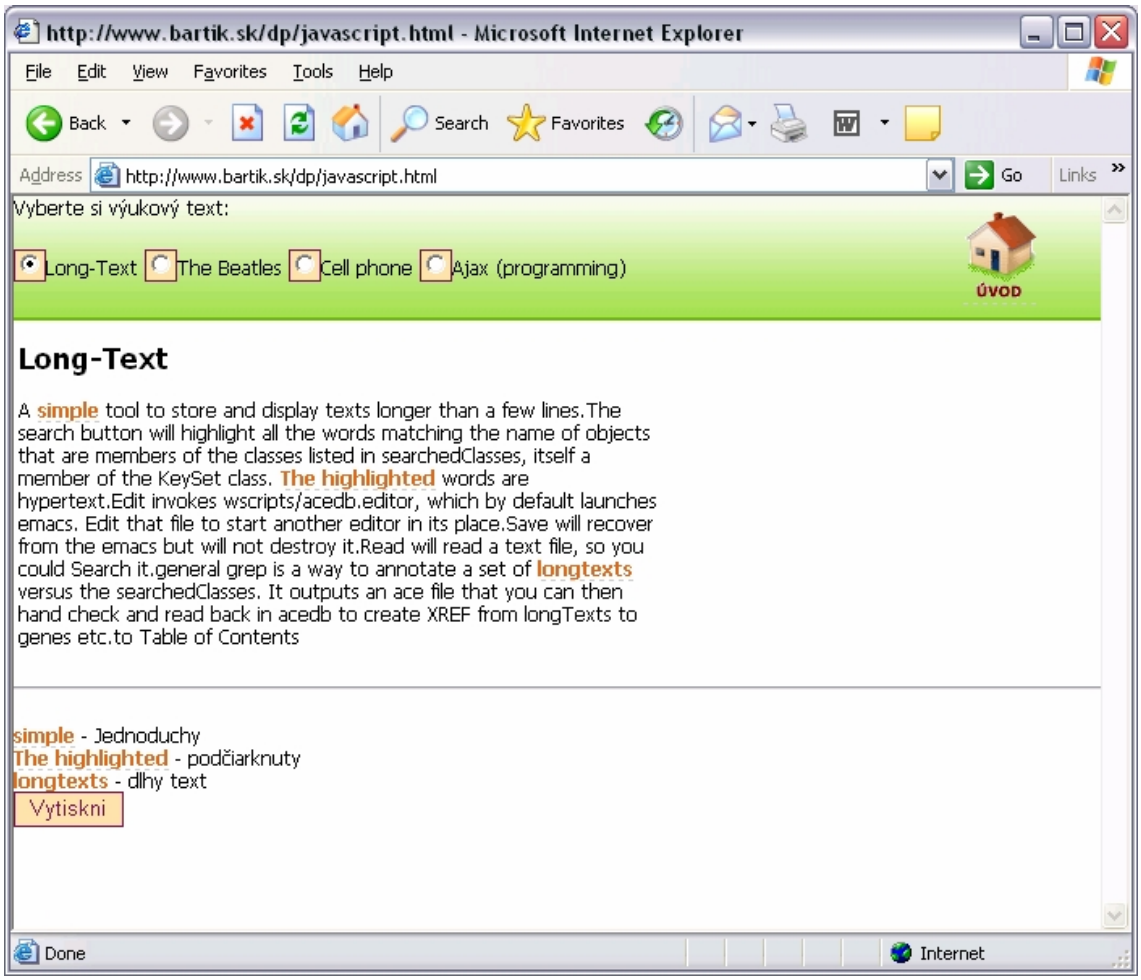
Obr.3.4: Funkcia lekce() na načítanie zoznamu z PHP skriptu

Hneď na začiatku sa volá funkcia getHTTPObject, ak je všetko na strane prehliadača v poriadku, pristupuje sa k otvoreniu súboru time.php. Výstup z neho putuje do funkcie spracujlekcie. Spôsobom spracovania PHP sa bude zaoberať serverová časť. Funkcia spracujlekcie() čaká na na stav "4" httpObject.readyState, čo je stav kedy je pripravený výstup z PHP na XMLHTTPRequest. Stavy readyState sú zobrazené na obr. 3.5

readyState	vracia prvky
	0 = uninitialized (neinicializovaný)
	1 = loading (načítava)
	2 = loaded (načítané)
	3 = interactive (spracované)
	4 = complete (kompletné)

Obr. 3.5: Stavy readyState na ktoré môže Javascript reagovať

Pri stave 4, keď je výstup z PHP pripravený sa načíta obsah do premennej *text*, a s tým sa ďalej pracuje. Text je rozdelený bodkočiarkou na číslolekcie;názovlekcie takže syntax rozdelí tento reťazec do poľa a vytvorí radiobutton s názvami. To sa udeje po aktualizácii DOMu HTML stránky - sekcie "divlekcie" (obr. 3.6). Keďže každý radiobutton je opatrený funkciou onclick="doWork()", po kliknutí naň sa spustí ďalšia funkcia, ktorá načíta stlačenú hodnotu bez odoslania formulára. Čiže Javascript v pozadí odošle údaj cez XMLHttpRequest na PHP skript s už požadovanou hodnotou výberu lekcie.



Obr.3.6: Zobrazenie "radio" tlačítok po aktualizácii časti HTML stránky a stlačení lekcie.

Celý proces sa opakuje a PHP zase príjme text. Syntax funkcie setOutput() musí spracovať text z databázy, ktorý má tvar **text**xaxa**slovo**@@@**poznamka_ku_slovu** @@@adresa_mp3@@@xaxa. To učiní načítaním do 2 polí a následným vypísaním textu a poznámky pomocou knižnice wz_tooltip.js. Tá má za úlohu zobraziť text po nabehnutí myšou ponad text so slovom, ktoré obsahuje poznámku. O prehranie súboru

mp3 sa postará soundmanager2.js, ktorého parameter je prijatý v druhej časti poľa. Tlačítko, ktoré po stlačení vytlačí text je použité na konci funkcie. Ak si užívateľ zvolí inú lekciu, celý proces sa opakuje znovu a načíta sa iný text so slovíčkami. Dole pod textom sú znovu vytlačené a zoradené všetky slová ktoré sú opatrené poznámkou.

,

3.1.2 Práca s článkom - strana servera

Na strane servera sa o úlohu práca s článkom starajú 3 PHP skripty. Prvý z nich, *time.php*, je volaný Javascriptom, keď je potrebné načítať z databázy zoznam lekcií a ich názvy. Deje sa tak príkazom "SELECT text, lekce FROM texty ORDER by lekce", ktorým sú načítané čísla lekcií z ukážok textov obr.3.7

```
 6 $query = 'SELECT text, lekce  FROM texty ORDER by lekce';
 7 $select=MySQL_DB_Query($dbname,$query,$conn);
 8 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
 9 $p[]= $vysledok['lekce'];
10 }
11 $p=array_keys(array_flip($p)); // vytriedenie unikatnych hodnot
12 $pomocna=0;
13
14 $query = 'SELECT nazev, id  FROM nazvy_lekcii ORDER by id';
15 $select=MySQL_DB_Query($dbname,$query,$conn);
16 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
17 if ($vysledok['id'] == $p[$pomocna]){
18 $nazov[$pomocna]= $vysledok['nazev']; //nacitanie nazvov k lekciam
19 if ($nazov[$pomocna]==''){
20 $nazov[$pomocna]='Lekcia nema nazov';
```

Obr. 3.7: Ukážka načítania prvkov z databáze

PHP skript štandardným spôsobom načítava dáta databázy a súčastne ich zapisuje do poľa *$p[ ]*. Na riadku 11 sú vytriedené prvky, keby sa náhodou nachádzala lekcia viackrát. V druhej časti skriptu sa podľa zistených lekcií načítajú názvy. Čiže skript otvorí inú časť databázy s názvami lekcií a načíta prvky *nazev* a *id*. Tie potom porovnáva s číslami lekcií a ak sa id a číslo lekcie zhodujú, zapíše ich do poľa. Kombinácia čísla lekcie a názvu je potom na výstupe skriptu oddelená bodkočiarkou.

Druhý skript, *text.php*, očakáva na vstupe hodnotu inputText, ktorá je odoslaná z Javascriptu. Táto hodnota je číslo lekcie, ktoré sa odoslalo zo stránky po stlačení radio buttonu. Číslo sa uloží do premennej $a. Skript pokračuje načítaním databázy s ukážkami. Ak pri prechádzaní databázou skript dojde na lekciu zhodnú s prijatým číslom, vypíše sa text, ktorý je už ďalej spracovaný Javascriptom.

```
 3 $a=$_GET['inputText'];
 4 include 'opendb.php';
 5
 6 $query = 'SELECT  text, lekce  FROM texty ORDER by lekce';
 7 $select=MySQL_DB_Query($dbname,$query,$conn);
 8 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
 9 if ($vysledok['lekce']==$a){
10 $text=$vysledok['text'];
11 echo $text;
```

Obr. 3.8: Ukážka text.php

Tretí PHP skript, ***download.php,*** je schopný stiahnuť z databázy súbor mp3 a priradiť k nemu hlavičku. Využíva na to MySQL príkaz „SELECT id, name, type, size, content FROM upload“, kde „upload“ je zložka v databáze určená pre ukladanie mp3. Časť content, čiže obsah, je typu mediublob a v nej sú uložené dáta. Ostatné prvky popisujú vlastnosti súboru – meno, typ, veľkosť. Tie sú zobrazené v header súboru na riadkoch 17 až 19 obr. 3.9.

```
 7 $query = "SELECT id, name, type, size, content FROM upload";
 8 $select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not select database (' .
 9 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)) {
10 if ($vysledok['id'] == $id) {
11 $content=$vysledok['content'];
12 $size=$vysledok['size'];
13 $type=$vysledok['type'];
14 $name=$vysledok['name'];
15 }
16 }
17 header("Content-length: $size");
18 header("Content-type: $type");
19 header("Content-Disposition: attachment; filename=$name");
20 echo $content;
```

Obr. 3.9: Načítanie mp3 z databáze a priradenie hlavičky k súboru

3.2.1 Doplňovanie slov - klientská strana

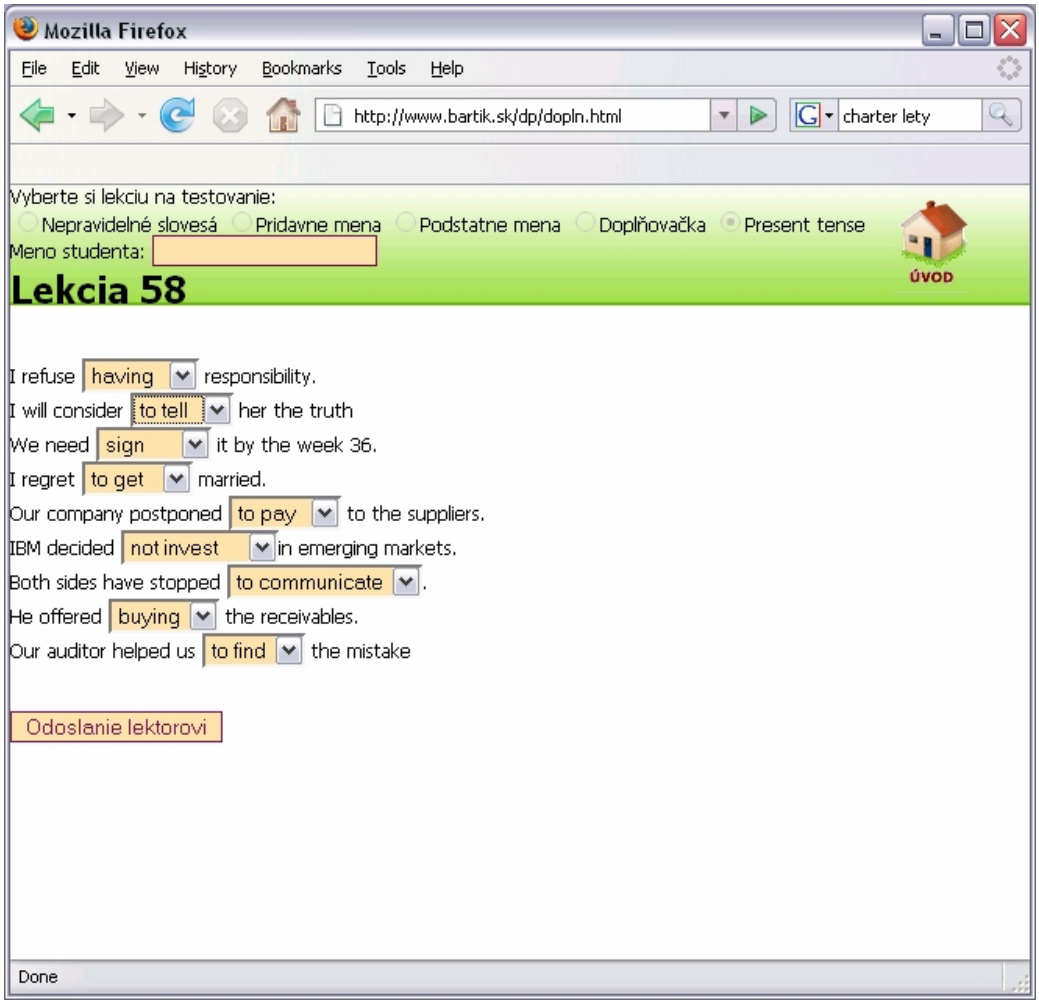
Princíp výberu lekcie je pri tejto úlohe taký istý ako pri predošlej, takže HTML stránka sa skladá z dvoch prázdnych DOM prvkov divlekcie a outputText. Po načítaní stránky sa spustí výber lekcie, kde je odoslaná na stranu serveru požiadavka. Vrátené sú čísla lekcií a ich názvy. Po stlačení určitého radiobuttonu sa odošle na stranu serveru číslo lekcie a prichádza text z databázy v tvare ***veta****xaxa****možnosť1****@@@****možnosť2*** *@@@možnosť3@@@xaxa****veta***. Podobne ako pri predošlej úlohe je tento výstup rozdelený do poľa a možnosti výberu správneho slovného tvaru sú vložené do prvkov <SELECT>. Aby sa zabránilo zobrazeniu prázdnych možností, ktoré neboli v priebehu zadávania vyplnené, sú do výberu možností pridané len prvky, ktoré neobsahujú

“1.moznost”, “2.moznost”, “null” riadok 27 na obr 3.10. Čiže tie, ktoré sú skutočne zadané ako možnosti.

```
25 for(var a=0;a<array2.length -1 ;a++){
26
27 if (array2[a]=='1.moznost' || array2[a]=='2.moznost' || array2[a]=='null')
28 {
29
30 }
31 else {
32 zlepenec=zlepenec+'<option value="' + array2[a] +'">'+ array2[a];
```

Obr.3.10: Zabránenie zobrazenia nevyplnených možností

Keď je radiobutton stlačený a text je už načítaný, je užívateľovi zamedzený prístup zvoliť inú lekciu (Obr.: 3.11). Po stlačení tlačítka Odoslať lektorovi sa spustí funkcia *odosli()*. V nej sa načítajú do poľa označené odpovede a tie sú spojené s pôvodnými vetami. Spojený výstup sa odošle na stranu servera. Ten, ak je všetko v poriadku, zobrazí na výstupe reťazec “ok“ a na obrazovku sa vypíše text o odoslaní emailu.



Obr.:3.11: Zobrazené vety s možnosťami výberu slov

3.2.2 Doplňovanie slov - strana servera

Na strane servera sú po prijatí čísla lekcie, ktorá bola zvolená užívateľom, načítané vety, ktoré súhlasia s číslom lekcie. V databáze je uložená veta v tvare ***veta*xaxa*možnosť'1*@@@*možnosť'2*@@@*možnosť3*@@@xaxa*veta.*** Prvá možnosť je zadaná ako správna, takže predtým ako bude veta odoslaná do Javascriptu, na strane servera je potrebné tieto možnosti premiešať. Deje sa tak rozdelením reťazca do polí a aplikovaním príkazu shuffle(nazov_pola), príklad na obr. 3.12. Možnosti sú potom pridané do vety v náhodnom poradí, v tom istom tvare ako pred tým.

```
 6 $query = 'SELECT id, veta, lekce  FROM dopln2 ORDER by id';
 7 $select=MySQL_DB_Query($dbname,$query,$conn);
 8 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
 9 if ($vysledok['lekce']==$a){
10 //zamiesanie  moznosti
11 $veta=$vysledok['veta'];
12 $array = split(';', $veta);
13 $array2 = split('@@@', $array[1]);
14 srand ((double)microtime()*1000000);
15 unset($array2[3]); // vymazanie prazdneho prvku
16 shuffle ($array2); // zamiesanie moznosti
17
18 $text=$text . $array[0] .';'.  $array2[0] .'@@@'. $array2[1].'@@@'
19 . $array2[2].'@@@;'. $array[2].'<br>';
```

Obr. 3.12: Zamiešanie možností

Keď je na serverovú časť odoslaný test, odošle sa lektorovi na uložený mail správa príkazom mail(email_príjemcu, predmet, odkaz, hlavička). Email príjemcu je načítaný z databázy, zo zložky „heslo“, tak isto aj celé meno a email študenta. Predmet je pevne stanovený, text je prijatý z formulára. Lektorovi bude zaslaný na jeho email odkaz o obdržaní testu študenta. Je potrebné text formulára zapísať do databázy a lektorovi poslať unikátny prvok, pod ktorým ho môže nájsť. Je preto tiež potrebné vygenerovať heslo. V *email.php* je použité vygenerovanie 20 miestneho hesla, ktoré je taktiež uložené do databázy s výsledným textom (Obr.3.9). Keď lektorovi príde email, bude ako meno odosielateľa zobrazené meno študenta a link sa bude odkazovať na súbor vysledky.php. V linku sa nachádza aj unikátny parameter heslo, pod ktorým bude PHP súbor v databáze vyhľadávať. Čiže ak načítaný prvok z databázy bude obsahovať heslo zhodné zo zadaným v odkaze, zobrazia sa aj ostatné údaje a to meno študenta a jeho výsledný formulár. To je potom na výstupe.

```
 8 $query = 'SELECT email FROM heslo';
 9 $select=MySQL_DB_Query($dbname,$query,$conn);
10 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
11
12 $sendTo = $vysledok['email']; //email prijemcu nacitany z databaze
```

Obr.3.13: Načítanie emailu lektora z databázy

Vo výsledkoch sa načíta z databázy text, ktorý bol uložený ako odpoveď. Pod textom sa načítajú z databázy správne odpovede. V poli s odpoveďami je prvý prvok ako správna odpoveď, ten sa spojí s pôvodnou vetou.

3.3.1 Preklad viet

Po načítaní lekcií a zvolení užívateľa, ktorá lekcia sa má zobraziť, je prijatý na vstupe Javascriptu text, ktorý obsahuje vetu a aj jej preklad. Reťazec sa uloží do poľa, kde prvý prvok je česká veta a nasledujúci veta anglická. Najprv sa zobrazí formulár s českou vetou a vstupným boxom, kde užívateľ napíše jeho preklad. Po stlačení tlačítka sa spustí funkcia odosli(), ktorá má za úlohu načítať hodnoty inputboxov a zobraziť českú vetu, preklad užívateľa a správny preklad. Deje sa tak už bez kontaktovania serverovej časti. Dáta správneho prekladu už Javascript v sebe obsahuje, a preto by tento postup nebol vhodný na testovanie, lebo pri menšej analýze zdrojového kódu je možné zistiť preklad, ktorý posiela server Javascriptu.

3.4.1 Test prekladu viet - klientská časť

Tak isto ako v predošlej úlohe sú načítané lekcie do radiobuttonov a po ich stlačení sa odošle číslo lekcie. Zo serverovej časti je prijatý reťazec, ktorý však neobsahuje anglický preklad, ale len česky zadanú vetu. Tento reťazec je rozdelený do poľa a tak isto sú použité inputy, do ktorých užívateľ píše jeho preklad. Po stlačení tlačítka sa tieto hodnoty načítajú a spolu s menom a číslom lekcie sú odoslané na serverovú časť. Ak je všetko v poriadku, serverová časť odpovie "ok" a Javascript zobrazí, že bol lektorovi odoslaný email. Deje sa tak bez akéhokoľvek znovunačítania stránky, mení sa len obsah DOM prvku.

3.4.2 Test prekladu viet - serverová časť

Prijatie formulára je podobné ako pri úlohe s doplňovaním slov. V tomto prípade je prijaté meno užívateľa, číslo lekcie a hodnoty vstupov s vlastným prekladom. Tieto údaje sú zapísané do databázy a tak isto je generované 20 miestne heslo, ktoré je ako odkaz preposlané lektorovi. Ten po kliknutí na odkaz zobrazí užívateľove výsledky. Obsah PHP skriptu s výsledkami je trochu iný ako v predošlom prípade, pretože okrem zobrazenia zadaných výsledkov, ktoré sú uložené v databáze, je potrebné zobraziť aj pôvodné zadanie a správny anglický preklad. Čiže zo zadaného hesla sa načíta celý

riadok, ktorého heslo sa zhoduje. Načítané hodnoty za zapíšu do premenných a hodnota text je rozdelená do poľa s odpoveďami užívateľa. Obr. 3.14

```
14 $heslo = $_GET["heslo"];
15 $query = "SELECT heslo,meno,text, lekce FROM vysledky";
16 $select=MySQL_DB_Query($dbname,$query,$conn);
17 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)) {
18 if ($vysledok['heslo'] == $heslo) {
19 $meno=$vysledok['meno'];
20 $text=$vysledok['text'];
21 $lekce=$vysledok['lekce'];
22
23 $rozsekane = array();
24 $rozsekane = split("[;]+", $text);
25 $size=sizeof($rozsekane);
```

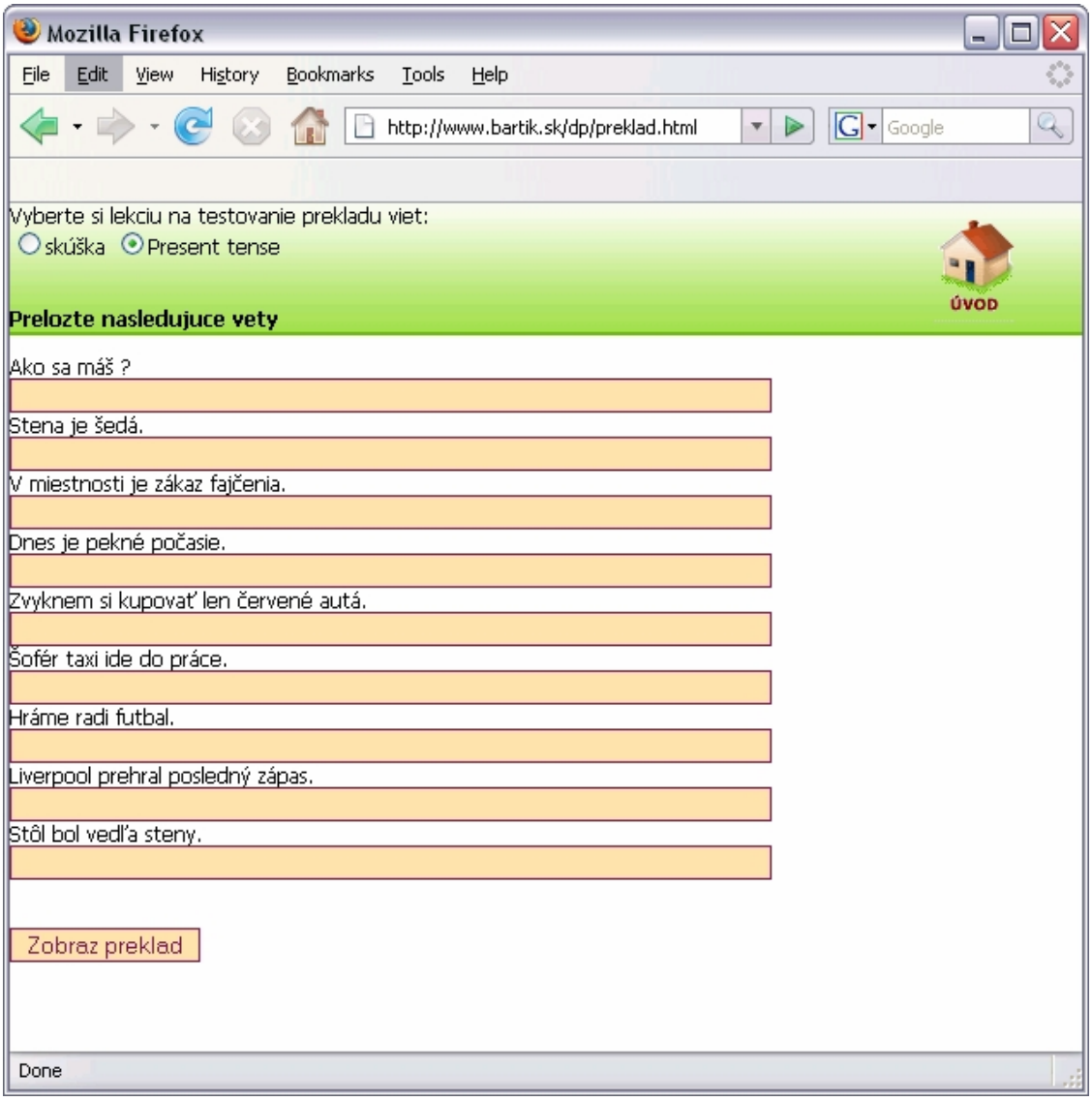
Obr.3.14: Načítanie údajov z databázy a rozdelenie reťazca do poľa

V ďalšom priebehu skriptu sa podľa hodnoty čísla lekcie z databázy načíta názov lekcie z tabuľky "lekce", tak ako aj v predošlých skriptoch. Potom je potrebné podľa čísla lekcie zistiť pôvodné zadanie, ktoré užívateľ mal Obr. 3.15.

```
41 $query = 'SELECT id, cesky, anglicky, lekce  FROM preklad2 ORDER by id';
42 $select=MySQL_DB_Query($dbname,$query,$conn);
43 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
44 if ($lekce==$vysledok['lekce']){
45 $cesky[]= $vysledok['cesky'];
46 $anglicky[]=$vysledok['anglicky'];
```

Obr. 3.15: Načítanie pôvodného zadania z databázy a priradenie do poľa

Keď sa všetky úkony prevedú, je už len potrebné vytvoriť výstup hodnôt meno študenta, lekcia, česká veta, odpoveď študenta, anglická veta. Lektor tak po kliknutí na odkaz v jeho emaily vidí priamo, čo bolo študentovi zadané a aké boli jeho odpovede.



Obr.3.16: Formulár pre testovanie prekladu

3.5.1 Otáčacie kartičky - strana klienta

Po načítaní lekcií zo strany servera sa odošle na stranu servera označená lekcia. Prijatý je reťazec, ktorý má tvar "česke_slovo;anglické_slovo;". Reťazec sa rozdelí do dvoch polí. Nachádza sa tu premenná *pocet*, ktorá určuje koľko bude mať jedna strana políčok. Je pevne stanovená na 4, ale pre ďalší vývoj programu je možné na vstupe zadať, koľko hracích kariet by sa malo zobraziť. Cyklom je vytlačený <table> a v ňom sú na začiatku zobrazené české slovíčka z poľa *cesky[]*. Každé slovo má svoj DOM názov a pri kliknutí na slovo sa spustí funkcia *zmen(id_slova)*. Táto funkcia má za úlohu načítať podľa hodnoty id, či sa rovná českému alebo anglickému slovu. Potom zmení jeho hodnotu na opačný jazyk Obr.3.17.

```
103 if (document.getElementById(b).firstChild.nodeValue==anglicky[b]){
104 document.getElementById(b).firstChild.nodeValue=cesky[b];
105 }
106 else {
107 document.getElementById(b).firstChild.nodeValue=anglicky[b];
108 }
```

Obr.3.17: Načítanie hodnoty slova a zmena hodnoty na opačnú

Funkcia *zmen* sa spustí vždy po kliknutí na slovo a nijak neovplyvňuje zvyšok stránky, keďže sa mení len obsah id slova. Zvyšná časť stránky je nezmenená. Pod tabuľkou so slovami sa nachádza tlačítko na vytlačenie slovíčok vo forme kartičiek. Po stlačení sa zobrazí verzia pre tlač. Sú to kartičky, ktoré obsahujú české slovo a anglické slovo a sú usporiadané na vytlačenie a nastrihanie. Po kliknutí na tlačítko sa otvorí okno tlačiarne. Po kliknutí na spátky sa spustí funkcia lekce(), takže hra je spustená od začiatku.

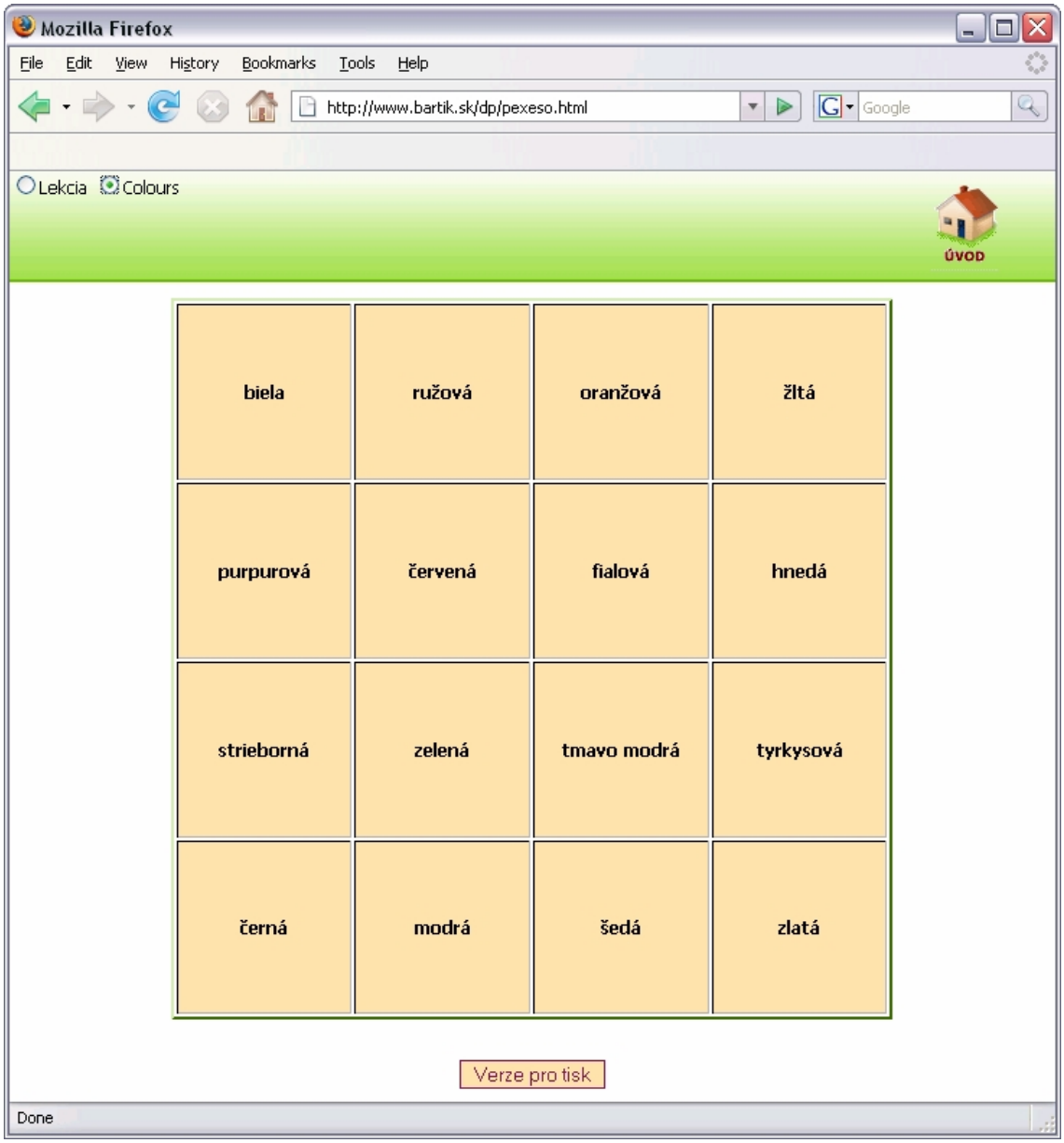
3.5.2 Otáčacie kartičky - strana servera

Prijatie lekcie na strane PHP sa deje cez súbor *nacitaj_pexeso.php*. Na vstupe sa čaká na prijatie čísla lekcie, ktorá bola v Javascripte vybraná. Je otvorená databáza, ktorá vyberie všetky dvojice slov, ktoré sú pod touto lekciou. Slová si uloží do dvoch polí *cesky[]* a *anglicky[]*. Nasleduje zamiešanie týchto dvojíc syntaxom na obr. 3.18 a ich následné vypísanie

```
 18 srand((float) microtime() * 10000000); // generovanie nahodnych cisel
 19 $rand_keys = array_rand($cesky, count($cesky));
 20 for ($i=0;$i<count($cesky);$i++){
 21 echo $cesky[$rand_keys[$i]] . ';'.$anglicky[$rand_keys[$i]]. ';'; //vypisanie zamies
 22 }
 23
```

Obr.: 3.18: Premiešanie dvojíc s českym a anglickým slovom

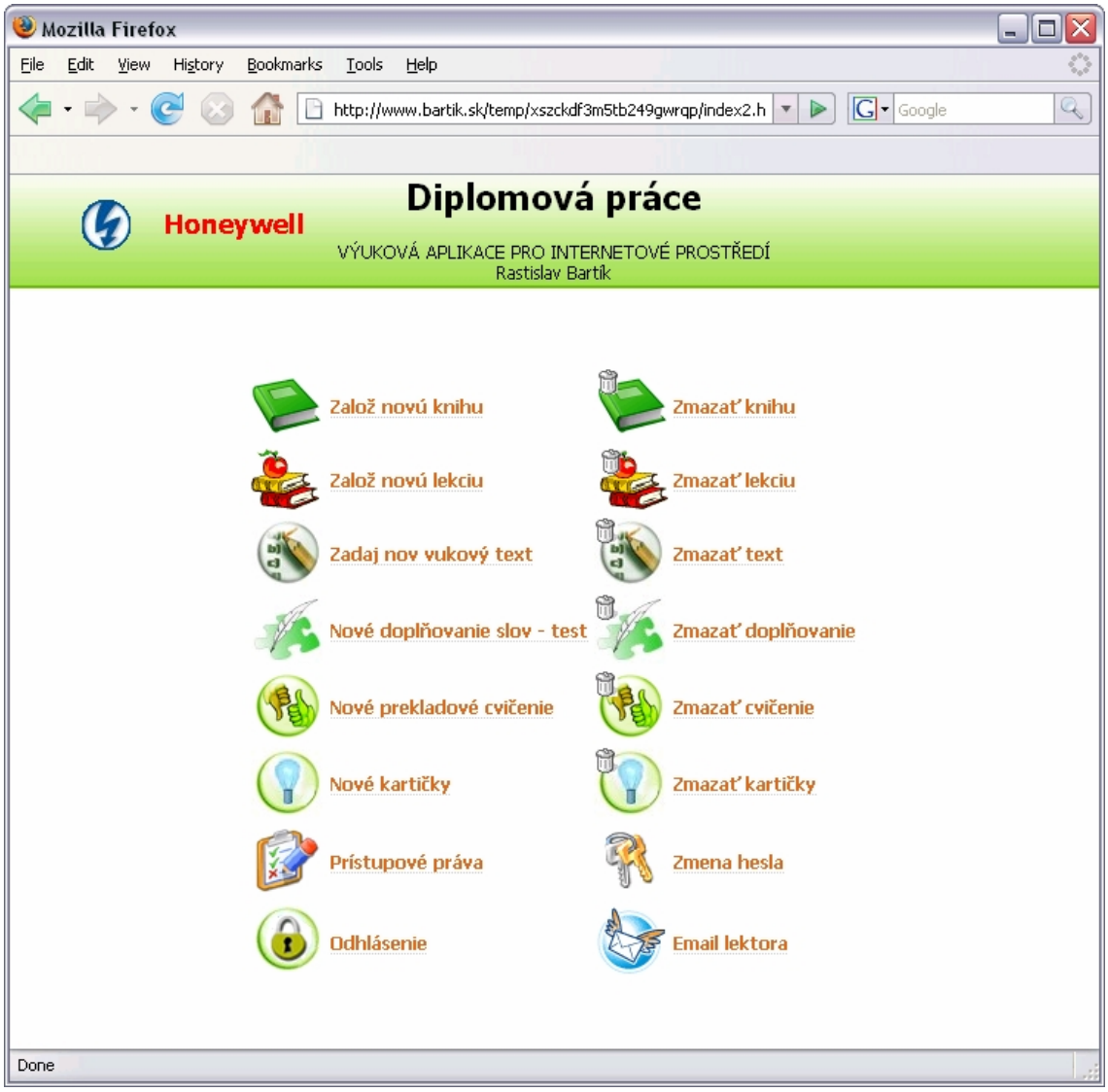
Výsledok celej úlohy sa nachádza na obr. 3.19.



Obr. 3.19: Úloha s otáčacími kartičkami

4. Rozhranie lektora

Rozhranie lektora je oddelená časť aplikácie, prístupná po zadaní hesla. Je v nej možné zadávať a mazať nové výukové texty, doplňovanie slov a prekladov. Nastavenia umožňujú zmeniť prístupové heslo a nastaviť email lektora, na ktorý sú odosielané výsledky testov. Ďalej je možné vytvárať knihy a lekcie, ktorých súčasťou sú cvičenia. Čiže ako prvé je nutné vytvoriť knihu, potom lekciu a do lekcie pridávať cvičenia. Na každú lekciu je možné nastaviť práva pre každého užívateľa zvlášť. Knihy, lekcie a cvičenia je možné mazať s tým, že ak je mazaná zložka v hierarchii vyššie, zmažú sa všetky jej podzložky. Ukážka hlavnej stránky pre lektora je na obr. 4.1.



Obr. 4.1: Rozhranie pre spŕavu aplikácie

4.1.1 Nový výukový text

Stránka slúži na zadanie výukového textu, ktorý je možné obohatiť o poznámky ku slovám alebo frázam. Tie sú zobrazené, keď sa kurzor myši nachádza nad slovom. Ďalej je možné pridať ku slovám zvukový záznam. Akceptované sú súbory mp3, ktoré sú nahrané priamo do databázy.

Po zadaní nadpisu a textu sa text uloží do globálnej premennej a je s ním možné ďalej pracovať. Ak užívateľ nezadá nadpis alebo text, je mu to oznámené. Inak nie je možné pokračovať. Po uložení textu sa zadávanie a zmena textových polí zablokuje a nie je ich možné meniť. Slová ktoré je potrebné opatriť poznámkou sa musia označiť myšou a potom stlačiť tlačítko „Pridaj poznámku“. Po stlačení tlačítka sa spustí funkcia *display( )* zobrazená na obr. 4.2.

```
46 function display() {
47 if (document.getElementById("textarea1")){
48  if (document.selection && document.selection.createRange) {
49     var range = document.selection.createRange();
50     var str = range.text;
51   } else {
52    var $obj = document.getElementById("textarea1");
53    str=($obj.value.substring($obj.selectionStart, $obj.selectionEnd));
54   }
55   if (str!=0){ //ak oznacenie mysou nie je prazdne
56     poz = prompt("Zadajte poznamku ku slovu: "+ str," ");
57     if (poz!=null){ // zadanie do pola len vtedy ak akcia nie je zrusena
58       poznamka[poznamka.length]=poz;
59       slova[slova.length]=str;
60       document.getElementById('rekapitulacia').innerHTML+= slova[slova.length-1]
61       +' - '+ poznamka[poznamka.length-1] +'<br>';
62       }
```

Obr. 4.2: Funkcia display( )

Funkcia sa spustí len vtedy, pokiaľ existuje prvok textarea1. To zabráni spúštanie funkcie počas iných operácií. V podmienke je implementovaná podpora pre rôzne prehliadače. Ak prehliadač podporuje document.selection, prevedie sa prvá časť podmienky. Túto funkciu podporuje hlavne Internet Explorer. Opera a Mozilla Firefox podporujú možnosť $obj.selectionStart a $obj.selectionEnd. Ďalej je vo funkcii ošetrené, že sa pokračuje len v prípade, ak označený text nie je prázdny. Príkazom prompt sa pridá ku slovu poznámka. Tá tiež nemôže byť prázdna, inak je akcia neplatná. Poznámka a tak isto samotné slovo sú uložené do globálnych polí poznamka[] a slova[]. Pod formulárom sa tak zapíše slovo a poznámka, ktoré boli úspešne zapísané. Po ukončení pridávania sa stránka prepíše na formulár s pridávaním mp3. Ku každému slovu je možné pridať zvuk, ktorý sa bude prehrávať pomocou soundmanager2.

Formulár je ošetrený pre prípad, že užívateľ stornuje výber mp3 súboru. Ak je súbor vybraný, je potrebné stlačiť tlačítko *Upload*. To spustí funkciu startUpload, ktorá má za úlohu zobraziť namiesto formulára obrázok počas nahrávania súboru na server.

Serverová strana spúšta funkciu stopUpload, keď je súbor nahraný. Tak isto vracia číslo id, pod ktorým je súbol uložený v databázy. Toto číslo je vložené do poľa mp3url[] v tvare id=XX.

Keď sú mp3 uložené po stlačení tlačítka sa prechádza na ďaľšiu funkciu *preview( )*. Úlohou tejto funcie je previesť text do tvaru **text**xaxa**slovo**@@@**poznamka** @@@**mp3url**xaxa**text**. Xaxa a @@@ sú oddelovače ktoré pri dešifrovaní rozkladajú text zase do polí. Na prevod textu poslúži príkaz replace, ktorý nájde vo výukovom texte všetky slová z poľa slova[] a nahradí ich požadovaným tvarom. Tento tvar je konečný a je odoslaný na stranu servera pre zápis do databázy. Príklad sa nachádza na obr. 4.3.

```
115 for(var i=0;i<slova.length;i++)
116                     {
117 var rexp=new RegExp(slova[i]);
118 
119 vyukovy_text=vyukovy_text.replace(rexp, 'xaxa'+slova[i]+'@@@'+poznamka[i]+'@@@'
120               + mp3url[i] + 'xaxa');
121     }
```

Obr. 4.3: Nahradenie slova v texte požadovaným tvarom

Funkcia preview ďalej dekóduje zadaný tvar takým istým spôsobom, ako sa to bude diať na strane študenta. To znamená, že z tvaru, ktorý bol vytvorený z výukového textu sa prvky oddelené oddeľovačmi uložia do polí a sú k nim priradené hepertextové odkazy. Tak bude lektor vidieť, ako bude jeho text zobrazený. Bude funkčné zobrazenie poznámky aj prehranie mp3 súboru, ktorý je už nahraný v databáze. V ďaľšom kroku funkcie je nahradený apostrof iným znakom, keďže tento znak robí problémy na strane servera pri zadávaní do databázy. V časti pre študenta sa tento znak znovu zmení tým istým nahradením na apostrof.

```
140 var rexp=new RegExp("'");
141 for(var i=0;i<100;i++){
142 vyukovy_text=vyukovy_text.replace(rexp, '`');
143 }
```

Obr.4.4: Nahradenie apostrofu iným znakom.

Lekcia je po stlačení tlačítka uložiť odoslaná na stranu servera a po prijatí odpovede je pridávanie zavŕšené oznámením.

4.1.2 Nový výukový text – strana servera

Strana servera pri tejto úlohe vykoná zápis do databázy s názvami lekcií, zároveň načíta id číslo, pod ktorým bol názov lekcie do zložky pridaný. Toto číslo je zapísané spolu s textom do zložky texty.

4.2.1 Nové dopĺňanie slov

Po načítaní stránky sa ako prvá spustí funkcia *start*. Má za úlohu odoslať na stranu servra požiadavok, ktorý vráti číslo, pod ktorým bude uložená súčastná lekcia. Prijaté číslo sa vo funkcii *afterstart* uloží do globálnej premennej *lekcia* a vytlačí sa spolu s formulárom. Formulár obsahuje 10 textových vstupov na vety v angličtine. Je potrebné vyplniť aspoň jednu vetu a nadpis, inak nie je možné po stlačení tlačítka pokračovať na ďalšiu funkciu. Ak je táto podmienka splnená postupuje sa na výber vstupov, ktoré boli vyplnené. Ak nebola vyplnená veta, nie je zapísaná do poľa s vetami. Vety sa zobrazia v takom istom formulári, ale už ich nie je možné meniť, tak isto ani nadpis. Aby bolo možné pridať nesprávne možnosti ku slovu, je potrebné dané slovo označiť a stlačiť tlačítko „Pridaj pozn.“

Po stlačení prechádza skript na funkciu display, ktorá, podobne ako v predošlej úlohe, načíta do premennej začiatok až koniec označenia myšou v danom riadku. Do funkcie display je odoslaný aj parameter p, ktorý nesie informáciu, o ktorý riadok vo formulári sa jedná. Na základe toho je ošetrené pridanie možností iba k danému riadku. To znamená, že nie je možné si spliesť tlačítka pri označení iného riadku. Ošetrné je aj označenie myšou, výber myši nemôže byť prázdny. Ak sú tieto všetky podmienky splnené, prechádza skript k vytvoreniu tzv. dropdown menu. Funkcia vyhľadá vo vete slovo, ktoré bolo označené a prepíše ho pomocou príkazu replace. Namiesto slova je do vety pridaný html kód s prvkom select a 3 možnosťami. Prvá možnosť je samotné slovo, ktoré bolo nahradené. Ostatné možnosti nie sú definované, a preto sú nahradené textom 1.možnosť a 2.možnosť.

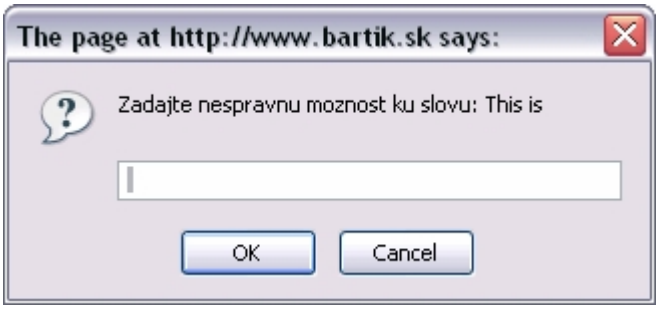
HTML kód obsahuje aj akciu, ktorá má byť vykonaná pri výbere z možností. Keďže Internet Explorer nepodporuje funkciu onclick pre option prvky, je potrebné použiť na prvku select príkaz *onchange*. Tak bude zaručená kompatibilita s viacerými prehliadačmi, keďže Opera a Mozilla podporujú oba spôsoby. Onchange odošle hodnoty jednotlivých možností, ktoré sa skladajú z dvoch čísel. Riadok, v ktorom sa dané slovo nachádza a možnosť, ktorá bola stlačená.

Príkaz onchange="ukaz(this.value);" odošle tieto dve hodnoty, ktoré sú oddelené čiarkami do funkcie *ukaz*.

```
106 function ukaz(temp) {
107
108 var array = temp.split(",");
109 var veta=array[0];
110 veta=(veta*1);
111 var moznost=array[1];
112 if (moznost){
113 if (moznost == 1) {
114 moznost1[veta] = prompt("Zadajte nespravnu moznost ku slovu: "+ slova[veta]," ");
115
116 }
117
118 if (moznost == 2) {
119 moznost2[veta] = prompt("Zadajte nespravnu moznost ku slovu: "+ slova[veta]," ");
120
121 }
```

Obr.4.5: Prijatie možnotí a ich zapisovanie do poľa

Funkcia *ukaz* príjme na vstupe reťazec s dvomi číslami. Ako prvé je nutné tento reťazec oddeliť do poľa. Na riadku 108 je ako oddeľovač použitá čiarka. Získané pole má 2 prvky, vetu a možnosť, ktorá bola stlačená. Na to, aby bolo možné s číslom vety pracovať ako integer, je potrebné ho vynásobiť 1, pretože príkaz split generuje reťazec typu string. Premenná veta bude ďalej použitá pri spočítavaní, takže tento krok je nutný pre správne fungovanie neskoršieho cyklu. Nasleduje podmienka, či druhá časť poľa obsahuje 1 alebo 2. V oboch prípadoch sa postupuje rovnako a to tak, že možnostiam do poľa je pridané slovo, ktoré užívateľ zadá do prompt boxu na obr.4.6.



Obr. 4.6: Prompt box pre zadanie nesprávnej možnosti ku slovu.

Po zadaní sa veta s prvkom select obnoví už o zadanú hodnotu nesprávnej možnosti. Takýmto spôsobom je možné vyplniť zvyšné možnosti. Ak ale užívateľ nezvolí pridanie možnosti, alebo stlačí storno pri pridávaní, tákáto možnosť nebude zobrazená pri výbere v študentskej časti. Zobrazia sa len zadané slová do prompt boxu. Po doplnení možností a stlačení tlačítka hotovo prechádza skript na funkciu preview. Úlohou funkcie je dané slová vo vete nahradiť tvarom ;slovo@@@moznost1@@@moznost2@@@;xaxa . Tento tvar bude potom schopná servrová časť rozdeliť do riadkov, keďže vety nebudú uložené v jednom reťazci, ale každá veta v jednom riadku. Ďalej sa funkcia preview na riadku 140 obr. 4.7 stará

o nahradenie apostrofu „ ‘ “ za znak „ ` ” ktorý je prijateľnejší pre MySQL databázu. Apostrof bude prepísaný skriptom v študentskej časti znovu na správny tvar.

```
136 var rexp=new RegExp(slova[i]);
137 temp = temp + vety[i].replace(rexp, ';'+slova[i]+'@@@'+moznost1[i]+'@@@'
138        +moznost2[i]+'@@@;');
139 temp = temp + 'xaxa';
140 var rexp=new RegExp("'");
141 temp=temp.replace(rexp, '`');
142
143 }
```

Obr. 4.7: Nahradenie slova vo vete tvarom pre databázu

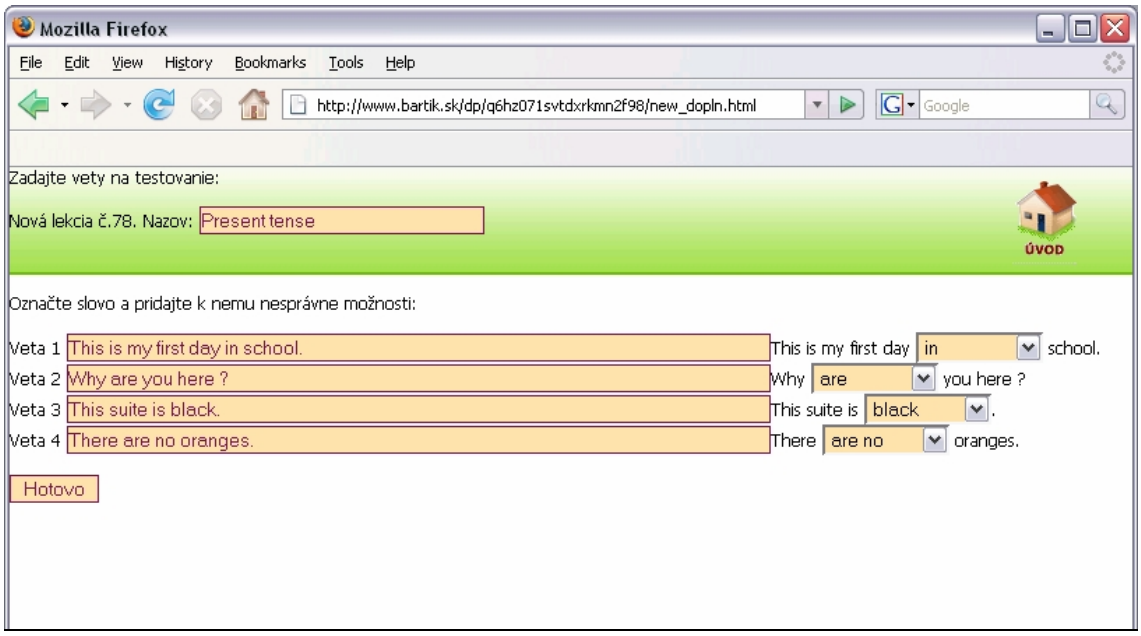
V ďaľšej časti funkcie sa získané a upravené dáta odošlú na stranu servra cez XmlHttpRequest. Zobrazená je hláška o nahrávaní na server. Po prijatí je užívateľovi oznámené, že lekcia bola úspešne uložená.

4.2.2 Nové dopĺňanie slov – strana servera

Tak isto ako pri predošlej úlohe sa najprv zapíše do databázy a zložky nazvy_lekcii nadpis. Potom sa načíta číslo id, pod ktorým bol názov uložený a pridá sa do premennej $lekce. Reťazec, ktorý bol prijatý z JavaScriptu je potrebné rozdeliť na vety, pretože je potrebné mať uloženú každú vetu vo zvlášť riadku. Ako oddelovač bol použitý tvar „xaxa“. Ukážka rozhrania pre lektora sa nachádza na obr. 4.9.

```
136 $array = split('xaxa', $text);
137
 20 for ( $i=0; $i<sizeof($array); $i++) {
 21 if ($array[$i]!=''){
 22 $query = "INSERT INTO dopln2 (veta,lekce) VALUES ('$array[$i]', '$lekce')";
 23 $select=MySQL_DB_Query($dbname,$query,$conn);
 24 }
```

Obr. 4.8: Rozdelenie reťazca do poľa oddelovačom a zápis do databázy



Obr.4.9: Ukážka rozhrania pre zadávanie dopĺňania

4.3.1 Nové prekladové cvičenie

Na začiatku skriptu sa odošle na serverovú časť požiadavok, ktorý vráti číslo lekcie, ktorá má nasledovať. Toto číslo sa spolu s formulárom zobrazí na úvodnej stránke. Po zadaní viet je potrebné stlačiť tlačítko na uloženie. Odošle sa len toľko viet, koľko ich bude vyplnených. Formulár je ošetrený proti odoslaniu bez nadpisu lekcie. Funkcia spracuj má za úlohu priradiť do poľa cesky[] a anglicky[] vety, ktoré boli vyplnené. Súčasne pridá do premennej *zlepenec* vety odelené oddelovačom @@@. Ak je splnená podmienka, že dĺžka poľa je večšia ako nula, skript pokračuje na časť s odosielaním. Ešte pred tým je aplikované nahrádzanie apostrofu za iný znak, kvôli zápisu do databáze. Potom sa odošle vytvorený reťazec spolu s nadpisom na serverovú časť do súboru new_preklad.php. Po prijatí odpovede zo servera sa objaví informácia o úspešnom vložení do databázy.

4.3.2 Nové prekladové cvičenie – strana servera

Serverová časť je riešená v súbore new_preklad.php. Na vstupe sa priradia do premenných $nadpis a $text hodnoty prijaté z JavaScriptu. Nadpis sa zapíše do zložky nazvy_lekcii . Po zápise načíta skript z databázy číslo, pod ktorým bola lekcia uložená. Hodnota sa priradí do premennej $lekce. Reťazec prijatý ako $text sa rozdelí príkazom split na pole. V tomto poli sú uložené vety oddelené oddeľovačom @@@. Do databázy sa vložia prvky do každého riadku zvlášť spolu s číslom lekcie. Na obr. 4.10 sa cyklus zvyšuje o dva prvky poľa a do databázy sú tak zapísané české a anglické vety zvlášť.

```
22 for ( $i=0; $i<sizeof($array); $i+=2) {
23 if ($array[$i]!=''){
24 $a=$i+1;
25 $query = "INSERT INTO preklad2 (cesky,anglicky,lekce)
26          VALUES ('$array[$i]', '$array[$a]', '$lekce')";
27 $select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not select
28         database (' . mysql_errno() . '): ' . mysql_error()); ;
29 }
30 }
```

Obr. 4.10: Vloženie českej a anglickej vety do MySQL

4.4.1 Nové kartičky

Po prijatí čísla lekcie zo strany servera sa zobrazí formulár, ktorý obsahuje 32 vstupných koloniek. Zapíše sa do nich české slovo a jeho anglický preklad. Formulár je ošetrený proti nezadaniu nadpisu lekcie. Je potrebné vyplniť všetky kolonky, aby bolo možné odoslať formulár. Pexeso, ktoré bude vytvorené, je vždy 4x4. Po odoslaní sa

slová vložia do polí a priradia sa do reťazca, ktorý je spojený s oddelovačom @@@. Reťazec je spolu s názvom odoslaný na stranu servera. Ten keď odpovie, JavasScript zmení DOM prvok outputText na oznámenie o úspešnom vložení.

4.4.2 Nové kartičky – strana servera

Strana servera je riešená v súbore new_pexeso.php. Tak isto ako v predošlej úlohe je na vstupe prijatý text a nadpis, tie sú zapísané do databazy v nazvy_lekcii a slovicka2. Poradie je miešané v študentskej časti, takže nie je potrebné prehadzovať poradie slov.

4.5 Mazanie vytvorených cvičení

Mazanie cvičení je vytvorené pre každý typ úlohy, čiže zmazanie výukového textu, zmazanie lekcie s dopľňovaním, prekladom a zmazanie kartičiek. Vo všetkých prípadoch je riešenie spracované cez AJAX. Čiže cez html stránku ktorá spustí Javascript a ten požaduje od serverovej časti zoznam lekcií pre danú úlohu. Lekcie sú vypísané do select boxu. Po zvolení lekcie sa odošle znovu na stranu servera číslo lekcie, ktorá bola vybraná. Po prijatí sa zobrazia cvičenia, ktoré do danej lekcie patria.

V PHP skripte sa číslo cvičenia príjme a priradí sa do premennej $na_zmazanie. Na obr. 4.11. je príklad na zmazanie cvičenia z databázy preklad a súčastne vyradenie cvičenia z lekcie.

```
 3 $na_zmazanie=$_GET['inputText'];
 4 include 'opendb.php';
 5 $query = "DELETE FROM dopln2 WHERE lekce='$na_zmazanie'";
 6 $select=MySQL_DB_Query($dbname,$query,$conn);
 7
 8 $query = "DELETE FROM nazvy_lekcii WHERE id='$na_zmazanie'";
 9 $select=MySQL_DB_Query($dbname,$query,$conn);
10
11 $query = 'SELECT id, lekcie FROM kniha';
12 $select=MySQL_DB_Query($dbname,$query,$conn);
13 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
14 $nove_pristupy='';
15 $vysledok_id=$vysledok['id'];
16 $pristupy = split(';', $vysledok['lekcie']);
17       for ( $a=0; $a<sizeof($pristupy); $a++) {
18           if ($pristupy[$a]!==$na_zmazanie and $pristupy[$a]!==''){
19           $nove_pristupy=$nove_pristupy.$pristupy[$a].';';
20           }
21       }
22 $query2 = "UPDATE kniha SET lekcie='$nove_pristupy' WHERE id='$vysledok_id'";
23 $select2=MySQL_DB_Query($dbname,$query2,$conn);
24 }
```

Obr. 4.11: Zmazanie cvičenia z lekcie.

Takto je riešené mazanie vo všetkých zložkách.Výukové texty zo zložky texty, doplňovacie cvičenia zo zložky dopln2 a kartičky zo zložky slovicka2. Po vybraní sa

ešte zobrazí potvrdzovacie okno, ktoré je na obr. 4.12. Po jeho potvrdení je cvičenie lekcie nezvratne zmazaná v databáze.

Obr. 4.12: Potvrdenie zmazania lekcie.

5. Bezpečnosť a kompatibilita

Študentská časť musí byť oddelená od lektorskej, aby študenti nemohli vôbec zasahovať do zadávania textov alebo mazania databázy.

5.1 Prihlasovanie lektora

Prihlasovanie je riešné spôsobom cez časť PHP. Administrácia je prístupná po zadaní hesla, meno nie je vyžadované. Skript príjme heslo a priradí ho k premennej $heslo. Tú potom porovná s heslom, ktoré je uložené v databáze. Ak je heslo zadané správne, z databázy sa priradí do premennej $folder zložka, v ktorej sa nachádza celá administratívna časť.

```
 6 if(isset($_POST['heslo'])){
 7 $heslo=$_GET['heslo'];
 8 include 'opendb.php';
 9
10 $query = 'SELECT heslo, folder FROM heslo';
11 $select=MySQL_DB_Query($dbname,$query,$conn);
12 while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
13 if ($heslo == $vysledok['heslo']){
14 $folder = $vysledok['folder'];
15 header('Location: '.$folder.'/index2.html');
16 }
17 }
18 include 'closedb.php';
19 }
```

Obr. 5.1: Načítanie a porovnanie hesla z databázy.

Zložka $folder sa mení po každom prihlásení, takže znovu prihlásenie do tej istej zložky nie je možné. Príkazom header na riadku 15 obr.5.1 sa stránka presmeruje na index2.html, ktorý je hlavnou stránoku rozhrania pre lektora.

5.2 Zmena hesla

Zmena hesla je už riešená cez AJAX, a to stránkami password.html, password.js a readpass.php. Na začiatku Javascriptu je spustená funkcia start. Nachádza sa v nej vypísanie formulára pre pôvodné heslo a nové heslo. Po stlačení tlačítka sa spustí funkcia spracuj, ktorá odošle v pozadí na serverovú časť staré a nové heslo. Po prijatí odpovede sa zobrazí celý text zo serverovej časti. Čiže informácia o úspechu alebo neúspechu zmeny hesla.

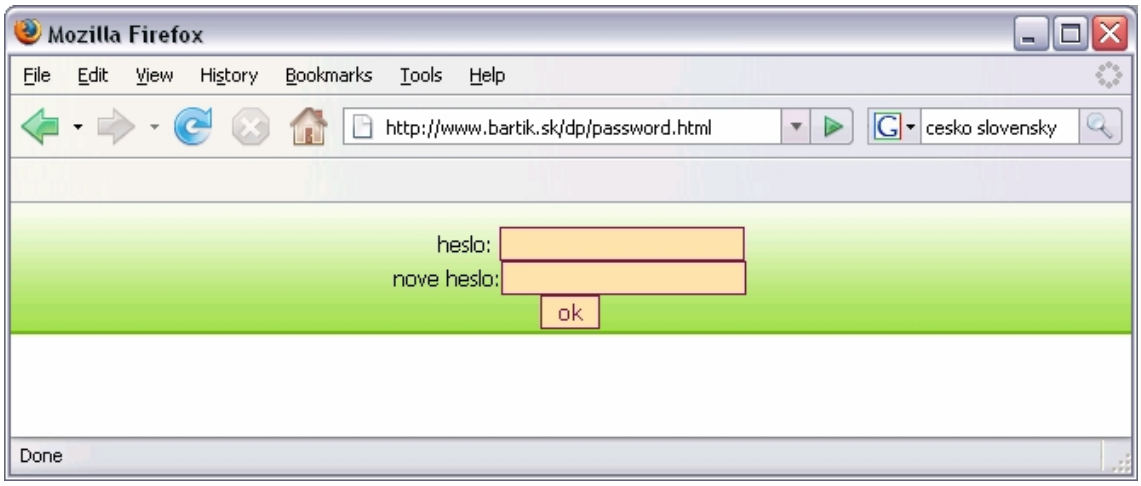
Na strane servera operuje skript readpass.php. Ten po prijatí starého a nového hesla načíta z databázy, zo zložky *heslo* súčastné heslo. Ak heslo nie je správne, je do Javascriptu odoslaná hláška „nesprávne heslo“. Ak je heslo správne, skript pokračuje

aktualizovaním pôvodného hesla. Príkaz má tvar: "UPDATE heslo SET heslo='$heslo_new' WHERE id='1'". Aktualizuje sa ním zložka heslo, stĺpec heslo, v prvom riadku zložky. Po zmene hesla sa aktualizuje aj zložka folder, pod ktorou je uložená celá administrátorská časť.

```
38 echo 'Heslo zmenene!';
39 rename($folder_old,$folder);
40 $query = "UPDATE heslo SET folder='$folder' WHERE id='1'";
41 $select=MySQL_DB_Query($dbname,$query,$conn);
```

Obr. 5.2. Premenovanie zložky a zapísanie názvu do databázy

Starý adresár sa načíta z databázy a nový folder sa vytvorí generovaním náhodného 20 miestneho reťazca. Tým je potom adresár premenovaný príkazom. Oznámenie sa vypíše echom „Heslo zmenené“ a zložka sa premenuje príkazom rename na 39 riadku obr.5.2. Potom sa zapíše nová zložka do databázy.



Obr. 5.3: Zmena hesla lektora

5.3 Email lektora

Zmena emailu lektora je riešená na strane servera cez skript lektor_email.php. Tento skript je spúšťaný vo vnútri adresára prístupného len keď je užívateľ prihlásený. Na začiatku sa načíta súčastný email lektora a je zapísaný do formulára na zmenu hesla. Po prepísaní tohoto emailu sa kolonka email v záložke heslo v MySQL nahradí novým heslom. To iba v prípade, že kolonka nie je odoslaná prázdna. Ak užívateľ nezadá žiadny email, do databázy sa nič nezapíše. Po úspešnom zadaní emailu sa databáza s emailom aktualizuje a užívateľ je presmerovaný pomocou príkazu header na hlavnú stránku. Na obr.5.4 je zobrazená hlavná časť aktualizácie emailu.

```
26 $email_new=$_GET['email'];
27 if ($email_new!=''){
28 $query = "UPDATE heslo SET email='$email_new' WHERE id='1'";
29 $select=MySQL_DB_Query($dbname,$query,$conn);
30 header('Location: index2.html');
31 }
```

Obr. 5.4: Aktualizácia emailu lektora

Email je z tejto časti databázy načítavaný ostatnými skriptami pri odosielaní výsledkov študenta. Po jeho aktualizácii sa email automaticky odošle na adresu z databázy.

5.4 Odhlásenie

Pri odhlásení je potrebné prepísať meno zložky, v ktorej sa nachádza celá administrátorská časť. Na začiatku skriptu sa príjme z databázy súčastné meno zložky. Vygeneruje sa nový názov zložky s 20 znakmi. Tento názov je zapísaný do databázy a zložka je súčastne prepísaná na nový názov. Tak sa už nikto do administrátorskej časti nedostane bez znalosti hesla, pretože stránky sa budú nachádzať na novej adrese.

5.5 Prihlasovanie žiakov

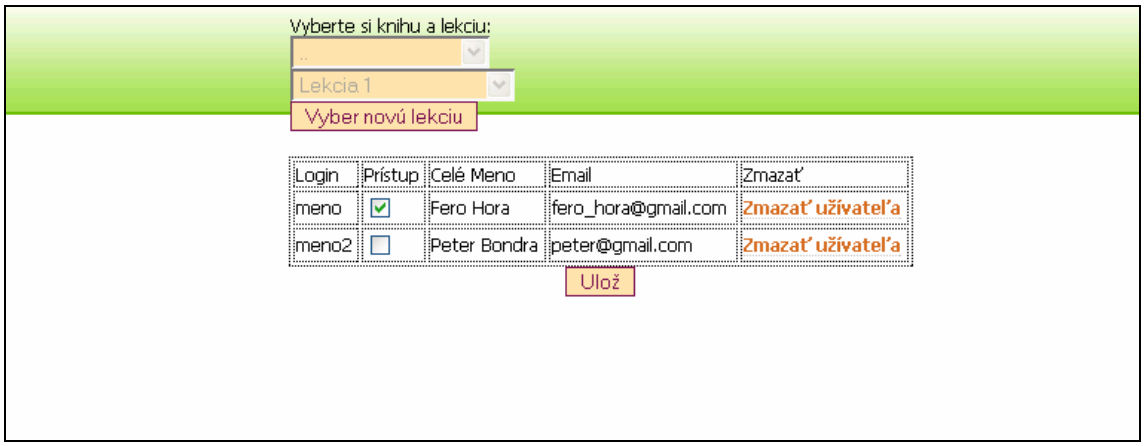
Pri prvom vstupe na stránku sa musí žiak zaregistrovať. Slúži na to skript registracia.php, ktorý je prístupný z hlavnej prihlasovacej stránky. Po zadaní všetkých informácií sú údaje zapísané do databázy s kolonkou žiaci. Skript je ošetrený pre prípady keď aspoň jedna kolonka ostane nevyplnená a pre prípad, že prihlasovacie meno už existuje. Ak sú všetky podmienky splnené, pristúpi skript na zadanie do databázy. Deje sa tak príkazom "INSERT INTO ziaci (meno,heslo,cele_meno,email) VALUES ('$meno', '$heslo','$cele_meno','$email')". Premenné sú pridané priamo z formulára.

Po prihlásení žiaka sa vygeneruje náhodný reťazec o dĺžke 20 znakov a vytvorí sa adresár s menom vygenerovaného reťazca. Do neho sú vložené potrebné súbory na beh študentského rozhrania. Pri odhlásení sa zložka zmaže a nie je možné na danú adresu viac pristupovať. Pri znovu-prihlásení má zložka iný vygenerovaný názov.

5.6 Správa prístupov k lekciám a knihám

Po spustení súboru ziaci.html sa zobrazí zoznam kníh a v ňom zoznam lekcií, ktoré existujú. Po vybraní lekcie sa zobrazia všetci užívatelia, ktorých systém pozná. Zaškrtávacími tlačítkami je možné užívateľov zapísať do lekcie alebo ich z lekcie

odstrániť. Sú zobrazené aj iné údaje o užívateľovi pre ich lepšiu identifikáciu. Možnosť zmazať užívateľa, vymaže celý záznam vrátane prístupov k lekciám. Zobrazenie je na obr. 5.5.



Obr.5.5: Rozhranie pre správu užívateľov a prístupu k lekciám.

5.7 Kompatibilita a kódovanie

Úlohy boli testované v prehliadači Mozilla Firefox verzia 2.0.0.11, Opera 9.24 a Internet Explorer 6 a 7. Vo všetkých prehladačoch pracuje aplikácia bez problémov. Problémom Internet Exploreru je tzv. „cash effect“, keď Javascript odosielal ten istý požiadavok viackrát na stranu servera. Odpoveď má vopred uloženú a vôbec nekontaktuje server. V tom prípade je potrebné vygenerovať náhodné číslo a priradiť ho k reťazcu odoslaného na stranu servera spolu s ostatnými premennými. Tým sa docieli, že Internet Explorer adresu nepozná a bude ju znovu požadovať. Príklad je na obr.5.6.

```
150 function save(){
151 httpObject = getHTTPObject();
152     if (httpObject != null) {
153 
154 var RandVal = Math.random(); //refresh zobrazenie pre Internet Explorer
155 httpObject.open('GET', 'selection.php?nadpis='+nadpis+'&rand='+RandVal, true);
156 httpObject.send(null);
157 httpObject.onreadystatechange = setOutput;
158 }
```

Obr.5.6: Generovanie náhodnej hodnoty pri casche efekte.

Aplikácia obsahuje aj akciu, ktorá má byť vykonaná pri výbere z možností <select><option>. Internet Explorer nepodporuje funkciu onclick pre option prvky, je potrebné použiť na prvku select príkaz *onchange*. Tak bude zaručená kompatibilita s viacerými prehliadačmi, keďže Opera a Mozilla podporujú oba spôsoby.

Aplikácia je odladená v Error konzoly programu Mozilla Firefox, kde sa nezobrazuje ani jedno upozornenie alebo chyba počas práce.

Tým, že je prihlasovanie na stránku riešené premenovaním administrátorskej zložky, nie sú potrebné ani cookies v internetovom prehliadači. Tým sa zvyšuje kompatibilita na viacerých prehliadačoch, keďže táto funkcia býva častokrát vypnutá.

Kódovanie stránok je riešené cez UTF-8. Toto kódovanie musí byť nastavené na html stránke, php skripte a tak isto aj na každom poli v MySQL databázy. Zobrazovanie bolo testované v prehladačoch Internet Explorer, Mozilla Firefox a Opera.

6. ZÁVER

AJAX pri kombinácii Javascriptu a PHP je významný nástroj na tvorbu internetových aplikácií. V tejto diplomovej práci je použitie tejto metódy vhodné pre väčšinu úloh.

Pri študentskom rozhraní boli spracované cvičenia pre zobrazovanie výukového textu, test doplňovania správneho tvaru, prekladové cvičenie, test prekladového cvičenia a výukové kartičky. Vo všetkých úlohách je prednostne použitá technológia zasielania požiadavkov na server v pozadí a tým sú demonštrované vlastnosti a výhody tohoto postupu pri webových aplikáciách.

V časti pre lektora je technológia AJAX taktiež využitá pri zadávaní cvičení. Pri zadávaní výukového textu sa väčšia časť práce deje na strane užívateľa v podobe Javascriptu. Na server sú už odoslané hotové dáta pre zápis do databázy. Pri dopĺňaní slov je taktiež dosť využitá strana klienta a tým funkcie Javacriptu, ktorý je vhodný na menšie úpravy. Pri zadávaní prekladu a kartičiek s prekladom je odoslaný len formulár a strana servera po zapísaní len odpovie o úspešnom zadaní.

V tejto časti sa nachádza aj vytváranie a mazanie kníh a lekcií. Táto na pohľad jednoduchá funkcia je riešená v pozadí na strane servera medzi PHP skriptom a MySQL databázou. Je potrebné prejsť celú databázu pri mazaní určitého prvku, či už cvičenia alebo lekcie. Všetky časti sú na seba naviazané a nesmie dôjsť k zbytočnému zapĺňaniu databázy. Po zmazaní knihy alebo lekcie,sú zmazané aj všetky ich prvky. Po zmazaní lekcie sú zmazané aj prístupy žiakov na už neexistujúcu lekciu.

Realizácia tohto projektu je kompatibilná s prehliadačmi Mozilla a Opera, po opravení „cache efektu“ pri prehliadači Internet Explorer, je stránka funkčná aj v tomto veľmi rozšírenom prehliadači. Dotvorená bola aj vizuálna stránka aplikácie použitím kaskádových štýlov. Tým, že pri používaní aplikácie nie je potrebné mať zapnuté cookies ani pri prihlásení na stránku, sa rozširuje jeho použiteľnosť v širokom spektre rôznorodých nastavení prehliadačov.

Aplikácia bola tvorená pre spoločnosť Honeywell, kde bude využívaná na výuku cudzích jazykov.

ZOZNAM POUŽITEJ LITERATÚRY

[1] Tutoriály: PHP MySQL Tutorial [online]. c2004-2007 [cit. 2007-11-11]. Dostupný z WWW: < http://www.php-mysql-tutorial.com/>.

[2] Tutoriály: AJAX Tutorial [online]. c1999-2007 [cit. 2007-10-11]. Dostupný z WWW: < http://www.w3schools.com/ajax/default.asp>.

[3] DOSTÁLEK, L., KABELOVÁ A. Velký průvodce porotkoly TCP/IP a DNS. 3 vydání. : Computer Press, a.s., c2002. 535 s. ISBN 80-7226-675-6.

[4] Darie C., Bogdan B., Cherecheş-Toşa F., Bucica M. AJAX and PHP - Building Responsive Web Applications: Packt Publishing Ltd., c2006. 273s. ISBN 1-904811-82-

[5] Tutoriály: Internet Explorer cache effect [online]. c2004-2007 [cit. 2008-12-05]. Dostupný z WWW: < http://heshanmw.blogspot.com/2008/02/avoid-cache-effect-in-internet.html>.

ZOZNAM PRÍLOH:

1. ziaci_pristupy.php – editácia prístupov k lekciám
2. delete_lekcia.php – zmazanie lekcie a všetkých jej súčastí
3. založenie nového výukového textu
4. zmazanie výukového textu

1. ziaci_pristupy.php – editácia prístupov k lekciám

```
<?php

$text=$_GET['text'];
$db_lekcia=$_GET['db_lekcia'];
$mena = split(';', $text);
include 'opendb.php';
$pomocna=0;
$query = 'SELECT meno,pristupne FROM ziaci ORDER by meno';
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
$vysledok_meno=$vysledok['meno'];
$nove_pristupy='';
$pristupy = split(';', $vysledok['pristupne']);
      for ( $a=0; $a<sizeof($pristupy); $a++) {
          if ($pristupy[$a]!==$db_lekcia and $pristupy[$a]!==''){
          $nove_pristupy=$nove_pristupy.$pristupy[$a].';';
          }
      }
      $query2 = "UPDATE ziaci SET pristupne='$nove_pristupy' WHERE
meno='$vysledok_meno'";
      $select2=MySQL_DB_Query($dbname,$query2,$conn) or exit('Could
not select database (' . mysql_errno() . '): ' . mysql_error()); ;
$nove=$nove.$nove_pristupy.'@@@';
$fero=$fero.$vysledok['meno'].';';
$pomocna++;
}
$mena_pov=split(';', $fero);
$nove = split('@@@', $nove);
for ( $i=0; $i<sizeof($mena_pov)-1; $i++) {

  for ( $a=0; $a<sizeof($mena); $a++) {

  if ($mena_pov[$i]==$mena[$a]){

  $zapis=$nove[$i].$db_lekcia.';';

    $query = "UPDATE ziaci SET pristupne='$zapis' WHERE
meno='$mena[$a]'";
    $select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not
select database (' . mysql_errno() . '): ' . mysql_error()); ;
}
}
}



echo 'ok';


include 'closedb.php';
?>
```

2. delete_lekcia.php – zmazanie lekcie a všetkých jej súčastí

```
<?php
$time= time();
$na_zmazanie=$_GET['lekcia'];
include 'opendb.php';

$query = "SELECT id, lekcie FROM kniha WHERE id='$na_zmazanie'";
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
$lekcie = split(';', $vysledok['lekcie']);
}
echo $na_zmazanie;
for ( $i=0; $i<sizeof($lekcie); $i++) {
$query = "DELETE FROM nazvy_lekcii WHERE id='$lekcie[$i]'";
$select=MySQL_DB_Query($dbname,$query,$conn);
    }

for ( $i=0; $i<sizeof($lekcie); $i++) {
     $query = "DELETE FROM dopln2 WHERE lekce='$lekcie[$i]'";
     $select=MySQL_DB_Query($dbname,$query,$conn);
    }

for ( $i=0; $i<sizeof($lekcie); $i++) {
$query = "DELETE FROM preklad2 WHERE lekce='$lekcie[$i]'";
$select=MySQL_DB_Query($dbname,$query,$conn);
    }

for ( $i=0; $i<sizeof($lekcie); $i++) {
    $query = "DELETE FROM texty WHERE lekce='$lekcie[$i]'";
$select=MySQL_DB_Query($dbname,$query,$conn);
    }

for ( $i=0; $i<sizeof($lekcie); $i++) {
$query = "DELETE FROM slovicka2 WHERE lekce='$lekcie[$i]'";
$select=MySQL_DB_Query($dbname,$query,$conn);
    }

$query = "DELETE FROM kniha WHERE id='$na_zmazanie'";
$select=MySQL_DB_Query($dbname,$query,$conn);

$query = 'SELECT meno,pristupne FROM ziaci ORDER by meno';
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
$vysledok_meno=$vysledok['meno'];
$nove_pristupy='';
$pristupy = split(';', $vysledok['pristupne']);
      for ( $a=0; $a<sizeof($pristupy); $a++) {
          if ($pristupy[$a]!==$na_zmazanie and $pristupy[$a]!==''){
          $nove_pristupy=$nove_pristupy.$pristupy[$a].';';
          }
      }
      $query2 = "UPDATE ziaci SET pristupne='$nove_pristupy' WHERE
meno='$vysledok_meno'";
      $select2=MySQL_DB_Query($dbname,$query2,$conn) or exit('Could
not select database (' . mysql_errno() . '): ' . mysql_error()); ;
}
echo $na_zmazanie . ' zmazana';
include 'closedb.php';

?>
```

3. založenie nového výukového textu

```
<?php

$nadpis=$_GET['nadpis'];
$text=$_GET['text'];
$db_kniha=$_GET['db_kniha'];
$db_lekcia=$_GET['db_lekcia'];

include 'opendb.php';


$query = 'SELECT id,lekcie  FROM kniha ORDER by id';
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
if ($vysledok['id']==$db_lekcia){
$lekcie_retazec=$vysledok['lekcie'];
}
}


$query = "INSERT INTO nazvy_lekcii (nazev,typ) VALUES ('$nadpis',
'text')";
$select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not select
database (' . mysql_errno() . '): ' . mysql_error()); ;



$query = 'SELECT nazev, id  FROM nazvy_lekcii ORDER by id';
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
if ($vysledok['nazev']==$nadpis){
$lekce=$vysledok['id'];
}
}

$lekcie_retazec=$lekcie_retazec.$lekce.';';
$query = "UPDATE kniha SET lekcie='$lekcie_retazec' WHERE
id='$db_lekcia'";
$select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not select
database (' . mysql_errno() . '): ' . mysql_error()); ;
$query = "INSERT INTO texty (text,lekce) VALUES ('$text', '$lekce')";
$select=MySQL_DB_Query($dbname,$query,$conn) or exit('Could not select
database (' . mysql_errno() . '): ' . mysql_error()); ;

echo $nadpis;
include 'closedb.php';

?>
```

4. zmazanie výukového textu

```
<?php
$time= time();
$na_zmazanie=$_GET['inputText'];
include 'opendb.php';
$query = "DELETE FROM texty WHERE lekce='$na_zmazanie'";
$select=MySQL_DB_Query($dbname,$query,$conn);

$query = "DELETE FROM nazvy_lekcii WHERE id='$na_zmazanie'";
$select=MySQL_DB_Query($dbname,$query,$conn);

$query = 'SELECT id, lekcie FROM kniha';
$select=MySQL_DB_Query($dbname,$query,$conn);
while ($vysledok = MySQL_FETCH_ARRAY($select,MYSQL_ASSOC)){
$nove_pristupy='';
$vysledok_id=$vysledok['id'];
$pristupy = split(';', $vysledok['lekcie']);
      for ( $a=0; $a<sizeof($pristupy); $a++) {
          if ($pristupy[$a]!==$na_zmazanie and $pristupy[$a]!==''){
          $nove_pristupy=$nove_pristupy.$pristupy[$a].';';
          }
      }
      $query2 = "UPDATE kniha SET lekcie='$nove_pristupy' WHERE
id='$vysledok_id'";
      $select2=MySQL_DB_Query($dbname,$query2,$conn) or exit('Could
not select database (' . mysql_errno() . '): ' . mysql_error()); ;
}

echo $na_zmazanie . 'zmazana';



include 'closedb.php';


?>
```