

HelpDesk portál pro Moodle UTB ve Zlíně

Jakub Řeha

Bakalářská práce
2014



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Jakub ŘEHA**
Osobní číslo: **A10178**
Studijní program: **B3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**
Forma studia: **kombinovaná**

Téma práce: **HelpDesk portál pro Moodle UTB ve Zlíně**

Zásady pro vypracování:

1. Specifikujte požadavky na HelpDesk portál.
2. Vytvořte návrh HelpDesk portálu pro Moodle UTB ve Zlíně.
3. Využijte GNU/GPL softwarové licence.
4. Vyřešte autentizaci uživatelů.
5. Navrhněte monitoring Moodle portálů.
6. Implementujte navržené řešení.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. ROSEBROCK, Eric a Eric FILSON. Linux, Apache, MySQL a PHP: instalace a konfigurace prostředí pro pokročilé webové aplikace. 1. vyd. Praha: Grada, 2005, 344 s. ISBN 8024712601.
2. LECKY-THOMPSON, Ed a Steven D NOWICKI. PHP 6: programujeme profesionálně. Vyd. 1. Brno: Computer Press, 2010, 718 s. ISBN 978-80-251-3127-5.
3. SCHWARTZ, Baron, Peter ZAITSEV a Vadim TKACHENKO. High performance MySQL. 3rd ed. Sebastopol: O'Reilly, 2012, xxviii, 793 s. ISBN 978-1-449-31428-6.
4. CASTRO, Elizabeth a Bruce HYSLOP. HTML5 a CSS3: názorný průvodce tvorbou WWW stránek. 1. vyd. Brno: Computer Press, 2012, 439 s. ISBN 978-80-251-3733-8.
5. GILMORE, W. Velká kniha PHP 5 & MySQL: kompendium znalostí pro začátečníky i profesionály. Vyd. 1. Brno: Zoner Press, 2005, 711 s. ISBN 808681520x.
6. BÖHMER, Marian. Návrhové vzory v PHP: [23 vzorových postupů pro rychlejší vývoj]. 1. vyd. Brno: Computer Press, 2012, 320 s. ISBN 978-80-251-3338-5.

Vedoucí bakalářské práce:

Ing. David Malaník, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce:

28. února 2014

Termín odevzdání bakalářské práce:

13. června 2014

Ve Zlíně dne 28. února 2014

prof. Ing. Vladimír Vašek, CSc.

děkan



prof. Ing. Vladimír Vašek, CSc.

ředitel ústavu

ABSTRAKT

Bakalářská práce se zabývá internetovou aplikací Helpdesk pro výukové portály Moodle na Univerzitě Tomáše Bati. Součástí této aplikace je monitoring dostupnosti jednotlivých portálů a jejich správa.

Klíčová slova: portál, helpdesk, bootstrap, curl, php, http, gnu, gpl, responsivní

ABSTRACT

The bachelor thesis deals with the Internet application Helpdesk for educational portals Moodle at Tomas Bata University. Part of this application is monitoring the availability of individual sites and their management.

Keywords: portal, helpdesk, bootstrap, curl, php, http, gnu, gpl, responsive

Na úvod bych chtěl poděkovat Ing. Davidu Malaníkovi, Ph.D. za cenné konzultace při zpracování této práce. Dále bych chtěl poděkovat rodině za shovívavost a trpělivost. Děkuji.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

| | |
|---|-----------|
| ÚVOD | 9 |
| I TEORETICKÁ ČÁST | 10 |
| 1 UŽIVATELSKÁ PŘÍSTUPNOST A DESIGN | 11 |
| 1.1 HTML5, CSS3 A DALŠÍ WEBOVÉ STANDARDY | 11 |
| 1.1.1 HTML5 | 11 |
| 1.1.1.1 Historie HTML5 | 11 |
| 1.1.2 CSS3..... | 13 |
| 1.1.3 jQuery..... | 14 |
| 1.1.4 Ajax | 18 |
| 1.1.4.1 Použití Ajaxu | 18 |
| 1.1.4.2 Požadavky HTTP | 18 |
| 1.1.4.3 Navázání spojení..... | 18 |
| 1.1.4.4 Serializace | 19 |
| 1.1.4.5 Poslání požadavku GET..... | 20 |
| 1.1.4.6 Poslání požadavku POST..... | 20 |
| 1.1.4.7 Odpověď HTTP | 20 |
| 1.1.4.8 Zpracování dat i odpovědi | 21 |
| 1.2 UŽIVATELSKY PŘÍVĚTIVÁ ROZHRANÍ..... | 22 |
| 1.3 DESIGN SYSTÉMU | 23 |
| 1.3.1 Bootstrap Framework | 23 |
| 1.4 RESPONSABILITA/ADAPTABILITA..... | 25 |
| 1.4.1 Rozlišení zobrazovacího zařízení pro návrh responsivní aplikace | 25 |
| 1.4.2 Responsivní navrhování | 27 |
| 1.4.3 Počátky responsivního navrhování | 28 |
| 1.4.4 CSS3 Media Queries | 29 |
| 2 LAMP | 31 |
| 2.1 LINUX..... | 31 |
| 2.2 APACHE..... | 32 |
| 2.3 MYSQL | 33 |
| 2.4 PHP | 34 |
| 3 LICENCOVÁNÍ | 35 |
| 3.1 SVOBODNÝ SOFTWARE (FREE SOFTWARE) | 35 |
| 3.2 OPEN SOURCE | 36 |
| 3.3 FOSS, F/OSS, FLOSS | 36 |
| 3.4 DFSG-COMPLIANT | 36 |
| 3.5 OSI-APPROVED | 37 |
| 3.6 PROPRIETÁRNÍ, CLOSED-SOURCE | 37 |
| 3.7 GPL A KOMPATIBILITA LICENCE | 37 |
| 3.8 MIT / X WINDOW SYSTEM LICENSE | 38 |
| 3.9 BSD LICENCE | 39 |
| 4 AUTENTIZACE UŽIVATELŮ | 40 |

| | | |
|--|---|-----------|
| 4.1 | HESLA PRO PŘIHLAŠOVÁNÍ NA MÍSTNÍCH POČÍTAČÍCH..... | 40 |
| 4.2 | OVĚŘOVÁNÍ POMOCÍ LDAP | 41 |
| 4.3 | OVĚŘOVÁNÍ PŘÍSTUPU PROSTŘEDNICTVÍM POŠTOVNÍHO SERVERU | 43 |
| 5 | MONITORING PORTALŮ | 45 |
| 5.1 | PROTOKOL HTTP..... | 45 |
| 5.1.1 | Formát odpovědi | 47 |
| II PRAKTICKÁ ČÁST | | 50 |
| 6 | HELPDESK PORTÁL PRO MOODLE..... | 51 |
| 6.1 | STRUKTURA APLIKACE | 51 |
| 6.2 | STRUKTURA ULOŽENÝCH SOUBORŮ A SKRIPTŮ | 51 |
| 6.3 | APLIKAČNÍ VRSTVA – APACHE / PHP | 52 |
| 6.4 | DATABÁZOVÝ MODEL | 54 |
| 6.5 | DESIGN APLIKACE | 54 |
| 6.6 | OVLÁDÁNÍ APLIKACE | 56 |
| 6.6.1 | Uživatelská část..... | 56 |
| 6.6.2 | Administrační část..... | 58 |
| 6.7 | NOTIFIKACE UŽIVATELŮ..... | 59 |
| 7 | AUTENTIZACE UŽIVATELŮ..... | 61 |
| 8 | MONITORING PORTAL..... | 63 |
| ZÁVĚR | | 65 |
| SEZNAM POUŽITÉ LITERATURY..... | | 66 |
| SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK..... | | 67 |
| SEZNAM OBRÁZKŮ | | 68 |
| SEZNAM TABULEK..... | | 69 |
| SEZNAM PŘÍLOH..... | | 70 |

ÚVOD

S rozvojem používání e-learningových kurzů, které slouží jako podpora výuky předmětů na jednotlivých fakultách UTB vznikla potřeba řešit požadavky, popř. chybové stavy jednotlivých výukových portálů. Za tímto účelem byla navržen helpdesk portál pro zadávání těchto požadavků, umožňující efektivní řešení problémů s ohledem na rozšiřující se možnosti přístupu k internetové aplikaci.

Tato práce je rozdělena na teoretickou a praktickou část. První, teoretická část, se zabývá obecným popisem technologií realizace a provozu internetových aplikací (HTML, CSS, LAMP apod.), současně shrnuje licenční podmínky v používání GNU/GPL licencí a jejich odvozenin.

Praktická část popisuje navrhované řešení. V první části je popsána architektura a design řešení. V dalších kapitolách je navrhována autentizace uživatelů a monitorování jednotlivých výukových portálů.

I. TEORETICKÁ ČÁST

1 UŽIVATELSKÁ PŘÍSTUPNOST A DESIGN

1.1 HTML5, CSS3 a další webové standardy

1.1.1 HTML5

HTML5 je nejnovější verzí tohoto značkovacího jazyka - zahrnuje nové funkce, vylepšení stávajících funkcí a různá API založená na skriptech.

HTML5 ale není přeformulovaná předchozí verze jazyka. Zahrnuje totiž všechny platné prvky jak z HTML 4.0, tak i z XHTML 1.0. Navíc byl navržen s ohledem na jisté prvořadě principy, aby bylo zajištěno, že bude fungovat téměř na jakékoli platformě, že bude kompatibilní se staršími prohlížeči a že bude elegantně zpracovávat chyby.

HTML5 v sobě zahrnuje nejenom již existující značkovací prvky (které byly předefinovány pro tuto verzi HTML), ale také nové prvky, které umožňují webovým designérům být v sémantice značkování expresivnější. Proč znepráhledňovat stránky prvky div, když máme k dispozici prvky article, section, header, footer a další?

Termín "HTML5" se také často používá, když je potřeba se odkázat na řadu jiných nových technologií a různých API. Některé z nich zahrnují kreslení "na plátno" (tedy prvek <canvas>), off-line úložiště, nové prvky video a audio, funkcionalitu pro přetahování (drag-and-drop), Microdata, vkládané fonty a další funkcionality.

Také je potřeba poznamenat, že některé z technologií, které před nějakou dobou byly součástí HTML5 odděleny od specifikace, takže v zásadě už nejsou zahrnovány do HTML5. Na druhou stranu existují i technologie, které ještě nikdy nebyly součástí HTML5 a přesto se tu a tam s ním spojují. To podněcovalo používat obecné a všezahrnující výrazy jako "HTML5 a související technologie". [1]

1.1.1.1 Historie HTML5

Odvětví webdesignu se vyvinulo v poměrně krátké časové periodě. Před dvanácti lety byl za "špičkový" (z hlediska webového obsahu) považován takový web, který obsahoval spoustu obrázků a do očí bijící design.

Dnes je situace naprosto odlišná. Stále běžnější jsou jednoduché a na výkon zaměřené webové aplikace postavené na Ajaxu, které se pro své kritické funkcionality spoléhají na skriptování na straně klienta. Dnešní weby často připomínají soběstačné softwarové aplikace a stále větší počet vývojářů na ně i takto nahlíží. Souběžně také probíhala evoluce webového

značkování. Z HTML 3.2 se přešlo na HTML4 a z něj poté na XHTML, což ale není nic jiného než HTML 4.0 se striktní syntaxí ve stylu XML. V současné době se všeobecně používá jak HTML 4.0, tak i XHTML, ale do čela se pomalu dostává HTML5.

HTML5 původně vzniklo jako dvě různé specifikace: Web Forms 2.0 a Web Apps 1.0. Obě byly důsledkem změněné situace na webu a potřeby rychlejších, efektivnějších a lépe udržovatelných webových aplikací. Formuláře a funkcionality podobná jako u aplikací jsou základem webových aplikací, takže bylo jen zcela přirozené, že se touto cestou ubírala i specifikace HTML5. Obě výše zmiňované specifikace se nakonec sloučily do formy, které dnes říkáme HTML5.

V průběhu doby, kdy se vyvíjelo HTML5, pokračovaly práce i na XHTML 2.0. Od tohoto projektu bylo ovšem upuštěno, aby bylo možné se plně soustředit na HTML5. [1]

HTML5 jako standard

Protože specifikaci HTML5 vyvíjejí dvě různé organizace WHATWG a W3C, existují dvě různé verze této specifikace. W3C je organizace, která udržuje původní specifikace HTML a CSS, přičemž je také hostitelem dalších webových standardů, jako jsou SVG (Scalable Vector Graphics, škálovatelná vektorová grafika) a WCAG (Web Content Accessibility Guidelines, směrnice pro přístupnost webového obsahu).

WHATWG (Web Hypertext Application Technology Working Group) zformovala ji skupina společností Apple, Mozilla a Opera poté, co byli v roce 2004 zklamáni výsledkem W3C. Zdálo se jim, že W3C ignorovala potřeby výrobců prohlížečů a koncových uživatelů, protože se soustřeďovala na XHTML 2.0, místo aby pracovala na zpětně kompatibilním standardu HTML. Vytvořili specifikace Web Apps a Web Forms. Ty byly následně sloučeny do jediné specifikace s názvem HTML5. Na základě toho W3C nakonec vzdala svou snahu o XHTML 2.0 a vytvořila svou vlastní specifikaci HTML5 založenou na specifikaci WHATWG.

Verzi specifikace od WHATWG se nachází na <http://www.whatwg.org/html/>. Nedávno byla přejmenována na "HTML" (tj. byla odstraněna číslice "5") a označena jako living standard", což znamená, že specifikace se neustále vyvíjí a už se na ni nebude odkazovat rostoucími čísly verze.

Verze WHATWG obsahuje informace, které pokrývají pouze funkcionality HTML, včetně těch, které jsou nové v HTML5. Organizace WHATWG poté vyvinula ještě dvě další

samostatné specifikace, které zahrnují všelijaké související technologie. Tyto specifikace zahrnují Microdata, Canvas 2D Context, Web Workers, Web Storage a další technologie.

Verzi specifikace od W3C se nachází na <http://dev.w3.org/html5/spec/>, přičemž k samostatným specifikacím ostatních technologií jsou k nalezení prostřednictvím URL <http://dev.w3.org/html5/>.

Jaký je tedy rozdíl mezi specifikací W3C a specifikací WHATWG? Stručně lze říci, že verze WHATWG je trochu neformálnější a více experimentální. Obecně jsou ale obě specifikace velmi podobné, takže jako základnu pro studium nových prvků HTML5 a souvisejících technologií můžete použít kteroukoli z nich.

Jádro HTML5 tvoří nejenom řada nových sémantických prvků, ale také několik souvisejících technologií a API. Tyto dodatky do jazyka a změny v něm byly zavedeny se záměrem, aby bylo možné webové stránky snadněji vytvářet, používat a snadněji k nim přistupovat.

Tyto nové sémantické prvky, společně s dalšími standardy, jako jsou WAI-ARIA a Microdata, zajišťují, že dokumenty budou přístupnější lidem i strojům - což poskytuje jisté výhody jak v oblasti přístupnosti, tak i v optimalizaci pro vyhledávače. Konkrétně, sémantické prvky byly navrženy s ohledem na dynamické weby, přičemž byla hlavní pozornost věnována tomu, aby stránky mohly být modulárnější a přenositelnější.

S HTML5 jsou také sdružená API, která pomáhají vylepšovat mnohé z postupu, které weboví vývojáři používají už léta. Různé běžné úlohy se dnes řeší jednodušeji a vývojáři mají větší kontrolu nad věcmi. Dále, příchod audia a videa založených na HTML5 snížil závislost na softwaru a zásuvných modulech jiných firem v situaci, když na webu publikujete obsah bohatý na média. [1]

1.1.2 CSS3

Další samostatnou součástí webových stránek jsou kaskádové stylové předpisy (Cascading Style Sheets, CSS). CSS je stylovací jazyk, který popisuje, jak se má HTML značkování prezentovat (neboli stylovat). CSS3 je nejnovější verze specifikace CSS. Termín "CSS3" není pouhý odkaz na nové funkcionality v CSS, ale také třetí průběžná úroveň specifikace CSS.

CSS3 obsahuje téměř všechno, co je obsaženo v CSS2.1 (což je předchozí verze specifikace). Přidává také nové funkcionality, které mají pomoci vývojářům vyřešit řadu

problémů, aniž by potřebovali nesémantické značkování, složité skripty nebo obrázky navíc. Mezi nové funkcionality v CSS3 patří podpora dodatečných selektorů, vržené stíny, zakulacené rohy, vícenásobná pozadí, animace, průhlednost a mnoho dalších věcí.

CSS3 je ale něco jiného než HTML5. CSS3 je odděleno od HTML5 a s ním souvisejících API.

Některé designérské postupy si najdou svou cestu téměř do každého projektu. Jako tři dobré příklady mohou posloužit vržené stíny, gradienty a zakulacené rohy, které je možno v různých projektech, aplikacích či stránkách. Když se využívají v souladu se všeobecným motivem a účelem webu, mohou tato vylepšení způsobit, že daný web přímo rozkvetne.

Ještě v nedávné minulosti, pokud chtěli weboví designéři vytvořit gradienty, stíny a zakulacené rohy, museli se uchýlit k četným trikům. Někdy to vyžadovalo použít nějaké HTML prvky navíc. V případech, kdy se podařilo udržet HTML kód poměrně čistý, bylo zase nutné použít nějaký skriptovací hack. Pokud se jednalo o gradienty, bylo použití dodatečných obrázků nevyhnutelné. A protože neexistovaly jiné cesty, jimiž by se daly zrealizovat designy tohoto typu, webdesignéři se museli spokojit s těmito náhradními řešeními.

CSS3 ale umožňuje používat tyto i další designérské prvky způsobem, při němž se myslí dopředu a který přináší spoustu benefitů - čisté značkování přístupné lidem i strojům, dobře udržovatelný kód, méně nadbytečných obrázků a rychlejší načítání stránek. [1]

1.1.3 jQuery

Knihovna jQuery poskytuje víceúčelovou abstraktní vrstvu pro běžné webové skriptování, a je tudíž užitečná skoro ve všech situacích, v nichž potřebujeme skriptovat. Kvůli její rozšiřitelné povaze není možné popsat všechny její možné způsoby použití v jediné knize, protože neustále vznikají nové zásuvné moduly, které ji obohacují o další funkce. Její základní funkce pomáhají s prováděním následujících úloh:

- Přístupovat k elementům dokumentu. Bez knihovny JavaScriptu musí weboví vývojáři často psát spoustu řádků kódu, aby prošli strom modelu DOM (Document Object Model) a vyhledali v něm určité části dokumentu HTML. S knihovnou jQuery mají vývojáři k dispozici robustní a efektivní systém selektorů, s nímž snadno získají určité části dokumentu, aniž by museli zkoumat, nebo dokonce manipulovat s modelem DOM.

```
$(`div.content`).find(`p`)
```

- Upravovat vzhled webové stránky. Jazyk CSS přináší výborný způsob změny vzhledu dokumentu, ale selhává v tom, že ne všechny webové prohlížeče podporují standardy stejně. S knihovnou jQuery můžou vývojáři překlenout tento nedostatek a spoléhat se na stejné standardy napříč rozdílnými webovými prohlížeči. Knihovna jQuery navíc umí měnit třídy a jednotlivé vlastnosti aplikované na různé části dokumentu, a to dokonce i po zobrazení stránky.

```
$(`ul > li:first`).addClass(`active`);
```

- Měnit obsah dokumentu. Knihovna jQuery se neomezuje jen na kosmetické změny dokumentu, ale může měnit také jeho obsah. Je možné měnit text, vkládat nebo zaměňovat obrázky, znovu seřazovat seznamy, nebo přepisovat či rozšiřovat celou strukturu dokumentu HTML, a to vše prostřednictvím přehledného rozhraní API (Application Programming Interface).

```
$(`#container`).append(`
```

- Reagovat na akce uživatele. Dokonce i ty nejdokonalejší funkce jsou nám k ničemu, pokud nemůžeme řídit, kdy je chceme spouštět. Knihovna jQuery nabízí elegantní způsob zachytávání událostí, jakou je například klepnutí na odkaz, aniž bychom museli „znečišťovat“ náš kód dokumentu HTML obsluhujícími funkcemi. Tato knihovna rovněž odstraňuje rozdíly v rozhraní API pro obsluhu událostí mezi různými webovými prohlížeči.

```
$(`button.show-details`).click(function() {  
  $(`div.details`).show();  
});
```

- Animovat změny v dokumentu. Jestliže chceme zavést takové interaktivní chování, musíme uživatelům poskytnout také vizuální odezvu. Knihovna jQuery nabízí sadu efektů (například prosvítání nebo posouvání), a také nástroje pro tvorbu nových vizuálních efektů.

```
$(`div.details`).slideDown();
```

- Načítat data ze serveru bez nutnosti obnovení stránky. Tento úkol můžeme provést pomocí technologie Ajax, jejíž název původně vyjadřoval zkratku pro asynchronní JavaScript a XML (Asynchronous JavaScript and XML), ale postupně se rozrostla a nyní reprezentuje daleko větší skupinu technologií pro komunikaci mezi klientem a serverem. Knihovna jQuery odstraňuje složitost související se specifickými odchylkami jednotlivých prohlížečů, díky čemuž se vývojáři můžou soustředit více na funkčnost serveru.

```
$(`div.details`).load(`more.html #content`);
```

- Zjednodušuje běžné činnosti při programování v jazyce JavaScript. Knihovna jQuery totiž nepřidává pouze nové funkce, ale také rozšiřuje základní konstrukce jazyka JavaScript, jakou je kupříkladu procházení a manipulace s poli.

```
$.each(obj, function(key, value) {  
total += value;  
});
```

Jak sílí zájem lidí o dynamické webové stránky, rozmáhají se rovněž frameworky pro jazyk JavaScript. Některé z nich se specializují pouze na jednu nebo dvě z výše uvedených úloh. Jiné se zase snaží pokrýt všechny myslitelné funkce a animace a naservírovat je uživatelům v jediném balíčku. Aby knihovna jQuery mohla uspokojit všechny tyto požadavky a současně zůstala poměrně kompaktní, používá následující strategie:

- Využívá znalosti jazyka CSS. Jelikož staví na selektorech jazyka CSS, získává jednoduchou, avšak velmi užitečnou možnost vyjádřit strukturu dokumentu. Knihovna jQuery se proto stala vstupní bránou do světa profesionálního webového vývoje pro návrháře, kteří chtějí doplnit reakce do svých stránek, jelikož k tomu potřebují pouze znalost syntaxe jazyka CSS.
- Je rozšiřitelná. Knihovna jQuery zabraňuje svému zahlcení funkcemi tak, že umožňuje tvorbu zásuvných modulů. Nové zásuvné moduly lze vytvářet jednoduše a k této činnosti existuje výborná dokumentace. Tato vlastnost odstartovala vývoj celé řady užitečných modulů. Většina funkcí v základním balíku knihovny jQuery je také implementována s použitím architektury zásuvných modulů, takže případně je lze z knihovny odstranit a získat ještě menší knihovnu.

- Abstrahuje výstřelky webových prohlížečů. Ve světě webového vývoje se bohužel setkáváme s tím, že každý prohlížeč má nějakou svou skupinu odchylek, jimiž se liší od vydaných standardů. Kvůli tomu musíme věnovat podstatnou část své webové aplikace tomu, abychom používali funkce jinými způsoby na různých platformách. Ačkoliv napříč webovými prohlížeči není možné dosáhnout stoprocentní shody, knihovna jQuery přináší abstraktní vrstvu, jež sjednocuje běžné úkoly, a tím zmenšuje velikost zdrojového kódu a zjednodušuje ho.
- Vždy pracuje se skupinami. Když řekneme knihovně jQuery: „Vyhledej mi všechny elementy s třídou collapsible a skryj je,“ nemusíme procházet jednotlivé elementy v cyklu. Místo toho zavoláme metodu `hide()`, která automaticky zpracovává celou skupinu objektů. Tato technika se nazývá implicitní iterace a zaručuje, že nemusíme používat cykly příliš často, což také zmenšuje velikost zdrojového kódu.
- Umožňuje provádět více akcí na jediném řádku. Díky tomu se vyhneme přehlcení dočasnými proměnnými a zbytečnému opakování, jelikož knihovna jQuery aplikuje na většinu svých metod techniku řetězení. Funguje to tak, že volaná metoda objektu vrací samotný objekt, takže s ním můžeme hned provést další akci.

Tyto strategie přispívají k poměrně malé velikosti knihovny jQuery (přibližně 90 KB v komprimovaném stavu), a k tomu ještě poskytují techniky, s nimiž udržíme velikost našeho vlastního zdrojového kódu na uzdě.

Elegance knihovny jQuery pochází z části z jejího návrhu a z části z postupného vývojového procesu, který ve své rozrůstající se komunitě neustále podstupuje. Uživatelé této knihovny diskutují nejen o vývoji zásuvných modulů, ale také o vylepšeních samotného jádra knihovny. Uživatelé a vývojáři rovněž pomáhají zdokonalovat její oficiální dokumentaci, která se nachází na internetové adrese <http://api.jquery.com/>.

Navzdory obrovskému množství práce, které stojí za takovým flexibilním a robustním systémem, je konečný produkt volně k dispozici. Knihovnu jQuery je možno využívat s licencí MIT, která umožňuje používat tuto knihovnu na libovolné webové stránce a ve vlastních komerčních aplikacích. [2]

1.1.4 Ajax

Ajax je termín vytvořený Jesse Jamesem Garrettem ze společností Adaptive Path k vysvětlení asynchronní komunikace klient-server, která je umožněna třídou XMLHttpRequest, poskytovanou všemi moderními prohlížeči. Zkratka AJAX znamená Asynchronní JavaScript a XML a používá se pro pojmenování technik využívaných k vytváření dynamických webových aplikací. Navíc jsou jednotlivé součásti Ajaxu různě zaměnitelné - například je možné použít HTML místo XML.

1.1.4.1 Použití Ajaxu

K vytvoření jednoduché implementace Ajaxu není potřeba mnoho kódu, ale co implementace umožňuje, je pro uživatele výhodné. Například místo toho, aby uživatel musel načíst úplně celou webovou stránku po odeslání formuláře, proces odesílání může být obstarán asynchronně a pouze malá část výsledků se může při dokončení načíst, například hledání volných doménových jmen (ke koupi) může být pomalé a pracné. Pokaždé při požadavku na vyhledání dalšího jména je nutné jej zadat do formuláře, odeslat formulář a sledovat, jak se načítá nová stránka. S použitím Ajaxu je možno dostat výsledek okamžitě.

1.1.4.2 Požadavky HTTP

Nejdůležitější a možná nejvíce konzistentní částí Ajaxu je požadavek HTTP. Protokol HTTP (Hypertext Transfer Protocol) byl navržen pro jednoduchý přenos dokumentů HTML a podobných souborů. Naštěstí všechny moderní prohlížeče podporují možnost vytvoření spojení HTTP dynamicky pomocí JavaScriptu. To se prokázalo jako velmi užitečné při vývoji pružně reagujících webových aplikací.

Asynchronní posílání dat na server a obdržení vrácených dat je hlavní účel Ajaxu.

1.1.4.3 Navázání spojení

Hlavní část činnosti Ajaxu spočívá v otevření spojení se serverem. Je několik způsobů, jak toho dosáhnout, např. jednoduše odesílat požadavky i přijímat odpovědi. Obecně se tato technika nazývá „používání objektu XMLHttpRequest“.

Komunikace dat pomocí objektu XMLHttpRequest je vedena dvěma cestami, v závislosti na uživatelově prohlížeči:

Internet Explorer, který byl průkopníkem tohoto způsobu komunikace založené na prohlížeči, ustavuje všechna svá spojení pomocí objektu ActiveX (jehož přesná verze je

závislá na verzi Internet Exploreru). Internet Explorer 7 má již nativní podporu objektu XMLHttpRequest.

Všechny moderní prohlížeče přejaly schopnosti objektu XMLHttpRequest do objektu stejného jména. To zahrnuje Firefox, Operu a Safari. I přesto, že se Internet Explorer liší od všech ostatních moderních prohlížečů ve způsobu vytváření objektu XMLHttpRequest, má stejnou sadu užitečných možností. Objekt XMLHttpRequest má několik metod, které se používají k navázání spojení a čtení dat ze serveru.

```
// Pokud je použit IE, vytvoříme kontejner pro objekt XMLHttpRequest
if ( typeof XMLHttpRequest == "undefined" )
    XMLHttpRequest - function(){
        // Internet Explorer používá pro vytvoření nového
        // objektu XMLHttpRequest ActiveXObject.
        return new ActiveXObject(
            // IE 5 používá jiný objekt XMLHttpRequest než IE 6
            navigator.userAgent.indexOf("MSIE 5") >= 0 ?
            "Microsoft.XMLHTTP" : "Msxml2.XMLHTTP"
        );
    };
// Vytvoříme objekt požadavku.
var xml = new XMLHttpRequest();
// Otevřeme socket.
xml.open("GET", "/nejaka/adresa.cgi", true);
// Navážeme spojení se serverem a pošleme další data.
xml.send();
```

Nejdůležitější možností celé metody AJAX je přenos dat od klienta (například z webového prohlížeče) na server. Proto je nutné všechna data zabalit a odeslat na server.

1.1.4.4 Serializace

Prvním krokem odeslání dat na server je jejich naformátování takovým způsobem, aby je server mohl jednoduše přečíst; tento proces je nazván serializace. Jsou dva případy serializace dat, které vám poskytnou nejširší možnosti přenosu:

- Přenos normálních objektů JavaScriptu, které mohou být použity k uložení párů hodnot s jejich jmény

- Odeslání hodnot z několika vstupních prvků formuláře (tento případ se liší od prvního v tom, že záleží na pořadí odeslaných prvků)

1.1.4.5 Poslání požadavku GET

Serializovaná data jsou k URL připojena za znakem „?“ . Všechny webové servery a aplikační frameworky umí data vložená za „?“ interpretovat jako serializovaný sled párů klíč/hodnota.

1.1.4.6 Poslání požadavku POST

Další způsob odeslání požadavků http na server s použitím objektu XMLHttpRequest, je metoda POST. Požadavek POST může přenést libovolná data jakéhokoli formátu a jakékoli délky (není omezen serializovanými daty v textovém řetězci)

Formát serializace obvykle mívá při odesílání na server typ obsahu Content-Type „application/x-www-form-urlencoded“, což znamená, na server je možné odeslat také XML („text/xml“ nebo „application/xml“) nebo javascriptové objekty JSON („application/json“).

Oproti požadavkům GET, které mohou pojmout maximálně jednotky kB (v závislosti na prohlížeči), není u požadavků POST nijak omezena velikost dat. Lze tak implementovat komunikační protokoly jako XML-RPC nebo SOAP.

1.1.4.7 Odpověď HTTP

Výhoda vytvoření a použití objektu XMLHttpRequest oproti jiným, jednosměrným způsobům komunikace je, že dokáže číst různé textové datové formáty ze serveru. To zahrnuje jeden z pilířů Ajaxu: XML. (Ačkoliv nikde není řečeno, že při vytváření ajaxové aplikace může být použito výhradně XML.)

K jednotlivým blokům odpovědi http lze přistoupit např. prostřednictvím vlastnosti objektu, responseXML a.responseText, budou každá obsahovat příslušná vhodně formátovaná data. Například pokud je ze serveru vrácen dokument XML, ve vlastnosti responseXML bude objekt dokumentu DOM; jakákoliv jiná odpověď a výsledky budou ve vlastnosti.responseText.

Bohužel objekt XMLHttpRequest nemá žádné vestavěné mechanismy pro ošetření chyb na serveru; kdyby měl, ušetřilo by to spoustu času. Nicméně s trochou práce si můžeme vytvořit mechanismy vlastní. Je spousta případů, kdy je třeba zjistit, jestli server neměl s přijatým požadavkem problémy:

Kódy úspěšné odpovědi

Očekávatelným způsobem zjištění chyby je zjištění stavového kódu odpovědi HTTP. Tyto kódy jsou popsány ve specifikaci HTTP jako způsob informování klienta, co server dělá. Úspěšně vyřízený požadavek má v odpovědi stavový kód od 200 do 299.

Odpověď beze změn

Dokument vrácený ze serveru může být označen jako nezměněný (stavový kód 304). To znamená, že data ze serveru se nezměnila a jsou označena jako data z vyrovnávací paměti prohlížeče.

Lokálně uložené soubory

Pokud je ajaxová aplikace spuštěna na místním počítači (neběží na webovém serveru), nebude vracet žádný stavový kód – i když je požadavek úspěšný. Je proto zajistit aby v případě dotazu na místní soubor při odpovědi bez stavového kódu byl považován požadavek za úspěšný

Kontrola úspěšnosti odpovědi HTTP je nezbytná, toto by vedlo k naprosto nepředvídatelným výsledkům (například kdyby server vracel namísto dokumentu XML chybovou stránku HTML).

1.1.4.8 Zpracování dat i odpovědi

Je prakticky nekonečné mnoho možných formátů, ve kterých mohou být data ze serveru vrácena. Nicméně reálně pracuje objekt XMLHttpRequest pouze s textovými datovými formáty. A navíc některé z nich (XML) dokáže zpracovat lépe než jiné (JSON)

Příklady dat, která mohou být ze serveru vrácena a použita na straně klienta

- **XML** - moderní prohlížeče nativně poskytují možnosti ke zpracování dokumentů XML tím, že je automaticky převedou do užitečných objektů DOM.
- **HTML** - odlišuje se od dokumentu XML tím, že obvykle je k dispozici pouze jako obyčejný textový řetězec, v němž je jen úryvek kódu HTML
- **JavaScript/JSON** - zahrnuje dva formáty - proveditelný javascriptový kód a datový formát JSON (JavaScript Object Notation).

Pro každý z těchto datových formátů jsou případy, kdy by byly speciálně užitečné. Kupříkladu je jistě hodně situací, kdy je smysluplnější vracet část kódu HTML, spíše než dokument XML. Podstata získávání dat z odpovědi HTTP tkví ve dvou vlastnostech objektu XMLHttpRequest:

- *responseXML* - pokud server vrátil dokument XML, tato vlastnost obsahuje referenci na předem připravený dokument DOM, který tento dokument XML reprezentuje.
 - *responseText* - tato vlastnost obsahuje textový řetězec s daty vrácenými ze serveru. Jak HTML, tak javascriptové datové typy závisí na tomto způsobu přístupu k datům.
- [3]

1.2 Uživatelsky přívětivá rozhraní

Způsobů interakce uživatele se systémem je několik. Každý z nich má své přednosti a je vhodný pro jinou situaci, proto dochází ke kombinaci jednotlivých typů. V tom případě je nutné, aby byla dodržena shodná terminologie, syntax, technika zpětné vazby a kontinuita činností. Například ukazovátko slouží pro výběr ve všech technikách. Změna dialogových technik musí přinášet stejný výsledek a přechod z jedné do druhé by neměl ovlivnit rychlost ani přesnost a neměl by být příčinou vzniku chyb.

Hlavním stylem interakce používaným v informačních systémech je menu. Jedná se o strukturovaně uspořádaný, předem daný seznam položek, které představují volby a možnosti interakce. Menu je nejrozšířenější formou především z důvodu minimální potřeby zácvičení uživateli. Styl menu bývá často doplňován metodou formulářů. Typické pro vyplňování formuláře je zadání textu, upravení označených polí nebo výběr z nabídnutého seznamu.

Pomocí techniky příkazového jazyka zadává uživatel přímo celá nebo zkrácená povelová slova, která musí znát, a to navíc v daném pořadí. Jejich zpracování pak vede k systémovým akcím. Tato technika je vhodná pro pokročilejší uživatele.

V rámci GUI je nejpoužívanějším stylem přímá manipulace. Interakce spočívá v působení uživatele na grafické znázornění abstraktních struktur nebo objektů, a to třeba ukázkou na ně. Manipulace s objekty je snadnou a názornou metodou dialogu.

Uspořádání elektronických informací do síťové struktury umožňuje *hypertext*. Tento princip se blíží uvažování lidského mozku, který asociuje na základě obrazů, to znamená nelineárně. Hypertext vzájemně propojuje textové segmenty s možností průchodu a mezi jednotlivými prvky pomocí odkazů vytváří kontextové vazby. Uživatel má libovolný přístup do sítě uzlů

s vazbami a pohyb mezi jednotlivými souvisejícími prvky mu napomáhá formulovat požadavek, respektive dotaz. Při volném průchodu mezi odkazy, který je označován jako *browsing*, může narazit na další podnětné informace a zdroje, na druhou stranu může být odvedena jeho pozornost od původního záměru. [4]

1.3 Design systému

User-centered design, design zaměřený na uživatele, představuje multidisciplinární přístup k návrhu uživatelských rozhraní, informační architektury, funkčnosti a grafického řešení. Jeho cílem je sjednotit tyto prvky tak, aby byl uživatel spokojen. Zaměřuje se na vše, s čím uživatel přichází do přímého styku, co vnímá, učí se a používá. Zahrnuje formu, chování a obsah informačních systémů. Klíčovými faktory jsou použitelnost, užitečnost a estetika.

Uživatelské rozhraní je klíčovým faktorem, který zásadně ovlivňuje úspěch a použitelnost systému, proto by jeho návrhu měli tvůrci věnovat zvýšenou pozornost. Takové rozhraní by pak mělo být jednotné a konzistentní v rámci celé aplikace.

Během návrhu uživatelského rozhraní informačního systému jsou definovány tři významné znalostní oblasti. Jednak je to znalost webového designu, programů, technologie. Druhou je znalost mentálních procesů lidí, komunikace a interakce. Třetí oblastí je odborná znalost řešené problematiky.

Nejdříve je to zjištění potřeb uživatele, zejména jaké činnosti bude v systému vykonávat. Druhým krokem je analýza uživatele, jeho dovedností, zkušeností, fyzických a kognitivních schopností. Důležitou roli hraje také vzdělání, věk, kulturní a etnické zázemí, motivace, cíle a celková osobnost uživatele. Dokonce je nutné znát i technické vybavení na straně uživatele. Za třetí by měl systém vycházet ze všeobecně uznávaných a platných standardů pro vytváření systémů a rozhraní. Na závěr je nutné zajištění správné funkčnosti a konzistence systému. [4]

1.3.1 Bootstrap Framework

Twitter Bootstrap je velmi jednoduchý a volně dostupný soubor nástrojů pro vytváření moderního webu a webových aplikací. Nabízí podporu nejrůznějších webových technologií HTML, CSS, JavaScript a mnoho prvků, které je možné snadno implementovat do internetové stránky nebo informačního systému. Pro použití Twitter Bootstrap jsou nutné základní znalosti HTML a CSS. Interaktivní prvky jako jsou tlačítka, boxy, menu a další

kompletně nastavené a graficky zpracované elementy je možné vložit pouze pomocí HTML a CSS.

Twitter Bootstrap Framework je šířen jako open-source postavený na MIT licenci.

Výhodou tohoto frameworku je snadné zpracování jakéhokoliv uživatelského rozhraní ve webové aplikaci a nerozhoduje, zda to je například uživatelské rozhraní v administraci back-endových nebo front-endových aplikací. [5]

Základní struktura frameworku

```
bootstrap/
├── css/
│   ├── bootstrap.css
│   └── bootstrap.min.css
├── js/
│   ├── bootstrap.js
│   └── bootstrap.min.js
└── img/
    ├── glyphs-halflings.png
    └── glyphs-halflings-white.png
```

Základní forma Twitter Bootstrapu jsou kompilované soubory, které je možno použít na kterémkoliv webovém projektu. Twitter Bootstrap poskytuje kompilované CSS a JS (bootstrap.*), stejně jako kompilované a minimalizované CSS a JS soubory (bootstrap.min.*). Twitter Bootstrap je závislý na javascriptové knihovně jQuery, licencované MIT licencí.

Komponenty

- Button groups – Skupiny tlačítek
- Button dropdowns – Vysouvací tlačítka
- Navigational tabs, pills, and lists – Záložky, pilulky a seznamy pro navigaci Navbar
- Labels – štítky
- Badges – “odznáčky”
- Page headers and hero unit – hlavičky stránky a “hero unit”
- Thumbnails – náhledy
- Alerts – výstrahy

- Progress bars
- Modals – vysouvací okna
- Dropdowns – vysouvací menu
- Tooltips, Popovers, Accordion
- Carousel – posuvný slider
- Typeahead [5]

1.4 Responsibilita/adaptabilita

1.4.1 Rozlišení zobrazovacího zařízení pro návrh responsivní aplikace

Pro účely webové aplikace je nutné dodržet navrženou grafiku s přesností na 1 px. Pouze grafika takto zpracovaná bude vypadat stejně a ostře na všech rozlišeních a zobrazovacích zařízeních. V případě zobrazení na monitoru nebo displeji s větším rozlišením než pro jaké jsou grafické návrhy připraveny, dochází k přepočítání a převzorkování grafiky do rozlišení daného displeje. Jako dobrý příklad může posloužit iPhone 4 od výrobce Apple, který je vybaven displejem o úhlopříčce 3,5". Do tohoto rozměru výrobce displeje vměstnal pixelovou hustotu 326 DPI. Faktor samotného rozlišení přispívá k velkým rozdílům mezi jednotlivými zařízeními. Pro webové účely však není vhodné exportovat grafiku v tak velkém rozlišení (jiné je to u mobilních aplikací určených vyloženě pro daný typ zařízení, zde se v tomto rozlišení grafika zpravidla navrhuje), neboť by to podstatně zvýšilo velikost webové stránky. Z tohoto důvodu se pro webové stránky stále navrhuje v rozlišení 72 DPI, které si displeje s větším rozlišením musí přepočítat tak, aby se zobrazily správně. Klíčové je zde zachování poměru daného obrazu a právě pixelová přesnost zpracování, neboť odchylka jednoho pixelu při trojnásobném znásobení vytváří trojnásobně větší. Základem tvorby layoutu webové stránky je tzv. grid systém neboli mřížka. Responsivní obsah, který je navrhován na šířku zobrazení na displeji stolního počítače, tedy v minimální šířce cca 960 px musí být adaptabilní takovým způsobem, aby bylo možné provést transformaci do šířky 320 px (šířka zobrazovací plochy displeje starších mobilních telefonů), což je nejmenší možná šířka, se kterou responsivní návrh počítá, aby zachoval použitelnost. [6]

V další fázi přípravy tvorby podoby webové stránky je nutné určit si, jakou formou bude transformace probíhat a jaké budou transformační body (jinak řečeno šířka stránky, kdy dojde k transformaci obsahu). Jde o matematickou operaci, kde se vyskytuje spousta pro-

měnných a naším cílem je definovat si pevné konstanty, které zavedou do konstrukčního procesu webového layoutu řád. Tyto pevné body vycházejí z používaných rozlišení zařízení, pro která jsou jednotlivé verze webové stránky navrhovány a která se v daném rozměru mají správně zobrazit. Mezi nejčastější konstanty, body zlomu fluidního designu, se uvažují šířky obsahu v těchto orientačních hodnotách a rozmezích:

Tabulka 1 – Bod zlomu fluidního designu

| Bod zlomu | Rozmezí zobrazení | Zařízení |
|-----------|-------------------|---|
| 320 px | 0 - 320 px | Starší mobilní telefony s přístupem k internetu (Nokia, Samsung, LG, Alcatel, ...) |
| 480 px | 321 - 480 px | Mobilní telefony s Androidem (Samsung Galaxy, Sony Xperia E, HUAWEI Ascend, LG Optimus, HTC Desire C, ...), telefony s iOS (iPhone 3G), telefony s Windows Phone (Nokia Lumia, HTC Touch, Samsung Omnia, ...) |
| 600 px | 481 - 600 px | Mobilní telefony s Androidem a vysokým rozlišením (HTC One S, Alcatel One Touch IDOL, Sony Xperia U/J/P, ...), Kindle, BlackBerry 9xxx |
| 768 px | 601 - 768 px | Mobilní telefony s iOS a displejem typu RETINA (iPhone 4S, iPhone 5) telefony s Windows Phone (Nokia Lumia 920) BlackBerry Z10 |
| 900 px | 769 - 900 px | Tablety s Androidem (Prestigio, Sencor) Mobilní telefony s Androidem (Samsung Galaxy S4, HTC One) |
| 1200 px | 901 - 1200 px | Starší displeje stolních počítačů, Zařízení s rozlišením rozlišením Full HD (1080x1920) tablety iPad Mini, iPad, iPad 2 |
| ∞ px | 1201 - ∞ px | Zobrazení na většině displejů stolních PC a displejích notebooků, tablety iPad 3 s RETINA displejem, tablety Samsung Galaxy Note/TAB 2, Acer Iconia Tab |

Rozdílné přístupy k realizaci responsivního vzhledu:

- Prvním způsobem je realizovat 100% fluidní design. Při tomto zpracování nejsou body zlomu rozhodujícím faktorem, ale design se procentuálně přizpůsobuje dle velikosti okna prohlížeče, či rozměru zobrazovacího zařízení.
- Druhým způsobem je determinovat jednotlivé layouts na základě bodů zlomu a fixně zobrazit pouze aktuální layout dle velikosti okna prohlížeče, či rozměru zobrazovacího zařízení.
- Třetím způsobem je kombinace prvního a druhého způsobu.

Je také nutné zohlednit uživatelskou použitelnost stránky a uvědomit si, že 100% využití nabízeného rozlišení např. na 30" monitoru s rozlišením 2560x1600 pro zobrazení obsahu v plném rozsahu nebude nejlepším řešením a že existují určité hranice, které je i v rámci responsivního zobrazení nutné dodržovat a grafiku webové stránky tomuto podřídit.

Z technologického pohledu je při tvorbě prvním způsobem nutné procentuální proměnné nastavení parametrů pro konkrétní objekt v CSS souboru, který definuje podobu webové stránky. V případě druhého způsobu je při tvorbě webové stránky použito fixního nastavení parametrů daného objektu v CSS souboru, ovšem oněch definic parametrů pro jeden objekt musí být x variant, kde x je počet zamýšlených layoutů webové stránky. Zjednodušeně řečeno, v případě realizace aplikace s pevnými body zlomu musí být fixně (absolutně) definována podoba všech zlomů, zatímco při realizaci procentuálním způsobem postačí proměnné (relativně) definovat pouze jednu podobu, která bude obsahovat popis toho, jak se mají jednotlivé prvky chovat vůči sobě v případě změny zobrazované velikosti. [6]

1.4.2 Responsivní navrhování

V posledních letech došlo k rozvoji trhu a objevuje se spousta rozličných mobilních zařízení, včetně herních konzolí, tabletů, notebooků různých tvarů a velikostí, e-čteček a telefonů. Každé takové zařízení má specifickou velikost displeje a rozlišení displeje, obvykle využívá svůj vlastní prohlížeč pro webové stránky. V případě některých zařízení je nutné navíc zohlednit orientaci zobrazení, dle kterého se na zobrazovací ploše mění rozložení zobrazovaného obsahu. Vzhledem k tomu, že se uživatelé stále více přesouvají od stolního PC a klasických monitorů a displejů k mobilním zařízením, i webové aplikace a stránky se tomuto faktu musí přizpůsobit. Webové stránky, a jejich obsah, které tento trend reflektují, získávají na svoji stranu určitou výhodu oproti stránkám, které nejsou schopny se tomuto vývoji přizpůsobit.

Důležité je si uvědomit, že v takto proměnném světě není již možné navrhovat pouze jeden layout, jedno zobrazení pro všechna zařízení. V dnešní době je nutné změnit navrhování grafické podoby webové stránky, tak aby nevypadala dobře pouze v jednom fixním rozměru, ale mohla se adaptabilně přizpůsobovat konkrétnímu zařízení. Mnohem více než konečnou podobu webové stránky, která se mnohdy stávala hlavním kritériem v případě navrhování pro fixní rozměry, je nutné zohlednit strukturu navigace a kompletní architekturu stránky, tak aby i po přeskupení návrhu byla navigace funkční a architektura logická.

Důležité součásti flexibilní/responsivní tvorby:

- Flexibilní základ (kostra webové stránky)
- Flexibilní architektura (tzv. grid)
- Flexibilní zobrazení grafiky/obrázků
- Implementace mediálních složek (video, animace) [6]

1.4.3 Počátky responsivního navrhování

Mezi začátky responsivního navrhování lze považovat procentuálně definované tabulky a rámce ze standardu HTML 4.0.

Ještě před začátkem masového užívání kaskádových stylů (CSS), a jejich plné podpory webovými prohlížeči, se jako hlavní nástroj při tvorbě layoutu webového dokumentu používaly tabulky <table>. Navzdory původnímu účelu tabulek se stalo pravidlem, že se obsah implementoval do jednotlivých řádků a sloupců, kdy se pomocí nastavení výšky a šířky dané buňky tabulky umisťoval obsah na stránce. V obou případech je toto použití přinejmenším problematické, takto vytvořená výsledná stránka, vytvořená z mnoha tabulek a vnořených tabulek, zpomalovala načítání webového obsahu. Tabulku je možné nadefinovat s pevnou šířkou a výškou, avšak je možné zadat i rozměry procentuálně. Procentuální číslo pak udává poměr mezi stránkou a oknem prohlížeče, případně poměr mezi jednotlivými buňkami tabulky. Tímto způsobem lze tedy docílit fluidního efektu, který lze považovat za základ responsivní tvorby, přestože o responsivní tvorbu jako takovou nejde, neboť takto vytvořený layout má jisté limity. [6]

Interpretace chybné definice je v jednotlivých prohlížečích rozdílná a dá se říci, že zajistit správnou zobrazitelnost na většině používaných prohlížečů je pracnější než pozicování layoutu stránky za použití kaskádových stylů, které jsou z pohledu použitelnosti a přístupnosti vhodnějším řešením, neboť zde v daleko větší míře získává na významu pozice jednotlivých prvků na stránce. V případě kaskádových stylů je obsah načítán postupně a

zobrazuje se tedy ihned, což je další důvod, proč se od tvorby layoutu webové stránky pomocí tabulek ustoupilo.

Dalším prvkem, který může vzdáleně připomínat počátky responsivní tvorby jsou tzv. rámy („frames“). Toto řešení je chronologicky ještě starší, než řešení pomocí tabulek, kterými byly rámy nahrazeny. Podstatou rámu je rozdělení okna prohlížeče, opět procentuálně či fixně. Rozdíl oproti tabulkám je takový, že do rámců se nahrává nová stránka, zatímco v rámci tabulky se jedná vždy o jednu stránku. Stránka takto utvořená je sice částečně flexibilní, ovšem je nutné si uvědomit, že stránky, které jsou součástí rámců, flexibilní být nemusí. V praxi se užívalo rámců např. pro vytvoření navigačního menu, kde se rámci se stránkou menu nastavila fixní velikost a rámci s obsahem velikost procentuální vzhledem k velikosti okna. Toto částečně fungovalo, neboť obsahová část se skutečně přizpůsobovala, ale rozpětí přizpůsobivosti bylo značně malé, neboť při změně velikosti na menší se z procentuální části rámce stala úzká nečitelná plocha. Mezi další nevýhody používání rámců by se dala započítat i vlastnost načítání externích stránek do rámců, což je nepraktické i z hlediska bezpečnosti, kdy uživatel nemohl přesně vědět, na jaké stránce se ve skutečnosti nachází. [6]

1.4.4 CSS3 Media Queries

Dotazy na média. Určitá část CSS deklarací se aplikuje jen po splnění podmínek napsaných pomocí Media Queries. V praxi se používá zejména v responzivním webdesignu. Dotazy na média jsou jedním ze tří pilířů klasického responzivního webdesignu spolu s fluidním layoutem a fluidními médii.

Z CSS2 jsou známé podmínky typy médií — @media print { ... }. CSS3 Media Queries je vylepšují o dotazy na vlastnosti média.

Syntaxe v CSS

```
@media (_podminky_) {  
    /* css kod, který se aplikuje jen po splnění _podminek_ */  
}
```

nebo do HTML na místě reference na CSS soubor:

```
<link rel="stylesheet" href="mobile.css" media="max-width: 480px">
```

Minimální/maximální výška/šířka

```
@media (max-width: 480px) {
  .container { width: auto }
}
```

Deklaraci pro .container prohlížeč aplikuje, pokud šířka viewportu (prostoru pro stránku v okně prohlížeče) nepřesáhne 480 pixelů. Alternativně lze použít dotaz na šířku displeje obrazovky: @media (max-device-width: 480px). Pomocí max-width lze základně otestovat aplikování změn pro malé displeje na desktopovém prohlížeči.

Dotazy na rozměry viewportu:

```
@media (min-width: 100px) { ... }
@media (max-height: 100px) { ... }
@media (min-height: 100px) { ... }
```

Logické operátory

Dotazy na média lze kombinovat pomocí operátoru „and“ řetězit jednak mezi sebou a taky kombinovat s typy médií:

```
@media screen and (min-width: 400px) and (max-height: 600px) { ... }
```

Další možné operátory jsou negace („not“). „or“ neexistuje, používá se místo něj čárka:

```
@media (max-width: 400px), print { ... }
```

Pokud je nutné skrýt před prohlížeči, které CSS3 nerozumí, aplikuje se klíčové slovo only:

```
@media only screen { ... }
```

Další vlastnosti k dotazování

Detekce vysokokapacitních displejů typu Retina @media only screen and (-webkit-min-device-pixel-ratio : 1.5), only screen and (min-device-pixel-ratio : 1.5) { ... }

Detekce orientace zařízení @media (orientation:portrait) { ... }

Detekce poměru stran obrazovky @media screen and (device-aspect-ratio: 16/9) { ... }[7]

2 LAMP

2.1 Linux

Linux je operačním systémem, na němž běží všechny uvedené aplikace. Je znám zejména svou rychlostí, minimálními hardwarovými požadavky, bezpečností a vzdálenou správou. Linux je schopen běžet s grafickým prostředím (GUI) nebo bez něj. Linux je projekt Linuse Torvaldse, který zpočátku pracoval se systémem Minix (malý Unix), se rozhodl vytvořit operační systém, který překoná standardy Minixu. S vývojem začal v roce 1991 a první veřejnou verzí byla 0.02. Vývoj Linuxu pokračuje dodnes ve formě uvolňování aktualizací v okamžiku, kdy se objeví tolik zásadních změn, že to za vydání nové verze již stojí. Torvalds má dnes mnohem rozsáhlejší tým vývojářů a nové verze jsou stále častější. Právě on také vybral tučňáka Tuxe za maskota Linuxu.

Jelikož je Linux publikován pod obecnou veřejnou licenci GPL/GNU, mnoho společností a jednotlivců převzalo zdrojový kód a upravili ho podle svých potřeb. [8]

Distribuce Linuxu

Některé z běžných distribucí jsou:

- **Debian** - Jedna z nejstarších distribucí, zakladatelem je Ian Murdock. Distribuce je pojmenována po jeho ženě (Debra). Je to přísná open-source distribuce, která je vyvíjena dobrovolníky z celého světa. Nabízí on-line repozitář (server, kde jsou uloženy zdrojové kódy) softwarových balíčků.
- **Fedora** – Sponzoruje firma Red Hat, jejíž distribuce je na tomto systému založena. Je to volná distribuce, která vzniká podobně jako Debian. Klade důraz na otevřenost a bezpečnost.
- **Red Hat Linux** – jedna z nejstarších distribucí, dnes komerční hlavně díky Red Hat Enterprise Linuxu. V Red Hatu vznikl balíčkovací systém RPM. Na desktopech se již často nepoužívá.
- **Slackware** – Jedna z prvních distribucí, vhodná spíše pro pokročilejší uživatele.
- **Gentoo** – Instalace neprobíhá formou binárních souborů, ale kompiluje se přímo na PC. Po dlouhé instalaci uživatel dostává optimalizovaný systém a software na svůj počítač.

- **SUSE** – Původně samostatná distribuce, později koupena firmou Novell. Další distribuce vhodná pro začátečníky. Je možné si koupit krabicovou verzi, ale dá se stáhnout zdarma z internetu.
- **Ubuntu** – Tato distribuce vhodná pro začátečníky vychází z Debianu. Je volně dostupná a přístupná v mnoha světových jazycích. Ubuntu zajišťuje komunitní i profesionální podporu. Nové verze LTS (tzv. vydání s dlouhodobou podporou) vychází jednou za dva roky, běžné verze jsou pak vydávány dvakrát ročně. Kromě Ubuntu existuje ještě Kubuntu, které používá grafické uživatelské rozhraní KDE, nebo Xubuntu, které používá grafické uživatelské rozhraní Xfce na rozdíl od Ubuntu, které používá GNOME a další odnože. Také existuje Edubuntu, které je zaměřeno na výuku.
- **Linux Mint** – Tato distribuce vhodná pro začátečníky i pokročilé vychází z Ubuntu, zaměřená hlavně na konzervativní uživatele. Je volně dostupná a přístupná v mnoha světových jazycích. Kolem distribuce vznikla rozsáhlá komunita, která zajišťuje podporu. Protože Linux Mint vychází z Ubuntu, tak přejímá i jeho systém vydávání nových verzí. Hlavní desktopová prostředí jsou Cinnamon, MATE, Xfce. Sesterská distribuce, která vychází z Debian se jmenuje Linux Mint Debian Edition.
- **Arch Linux** - nezávislá linuxová distribuce vytvořená Juddem Vinetem, jenž se inspiroval distribucí CRUX Linux. Arch Linux je vyvíjen jako nenáročný, odlehčený a snadno přizpůsobitelný systém. Tato distribuce obsahuje vlastní systém binárních balíčků, které jsou spravovány Pacmanem. [8]

2.2 Apache

Apache je otevřené řešení webového serveru vyvinuté společností Apache Software Foundation (ASF), které je přeplněné funkcemi, extrémně rychlé a dobře spolupracující s operačním systémem Linux. Na webovém serveru Apache je lze vytvářet virtuální hostitele, kteří umožní provozovat více webových sídel na jediném serveru. Nabízí ale i mnohé další vynikající prvky. Webový server Apache je k dispozici také pro prostředí Microsoft Windows.

Apache umožňuje vylepšené logování (protokolování, zaznamenávání), omezování šířky pásma, ochranu přístupu k adresářům, podporu rozhraní Common Gateway Interface (CGI), podporu vrstvy Secure Sockets Layer (SSL) a množství dalších vestavěných modulů.

Webový server Apache je šířen pod hlavičkou Apache License, Version 2.0, která je kompatibilní s GPL. [8]

2.3 MySQL

MySQL je výkonný, robustní databázový správce, který umožňuje ukládat a přebírat data pomocí skriptového jazyka jako např. PHP. Umožňuje ukládat různé typy dat, například logické operátory, text, celá čísla, obrázky, binární číslíčky a objekty BLOB (rozsáhlé binární objekty). Práce s databází je pro vytváření dynamických sídel důležitá. Termín „dynamické sídlo“ je odvozen z možnosti zobrazovat prostřednictvím jediné stránky s kódem různé informace na základě interakce uživatele. To by bylo bez práce s databází a skriptovým jazykem jako PHP, které manipulují s daty, prakticky nemožné.

MySQL je systém přeplněný funkcemi, které zahrnují replikování dat, uzamykání tabulek, omezování dotazů, uživatelské účty, více databází, trvalá připojení a - pokud se jedná o verzi MySQL 5 a vyšší - také uložené procedury, spouště a pohledy.

Systém MySQL od společnosti MySQL AB, vycházel z potřeby zakladatelů používat mSQL pro připojení k jejich vlastním rychlým, nízkoúrovňovým rutinám ISAM (Indexed Sequential Access Method - metoda indexovaného sekvenčního přístupu). Po otestování těchto procedur a funkcí bylo zjištěno, že vlastně nejsou ani rychlé, ani dostatečně pružné, a tak vzniklo MySQL. V současné době je MySQL součástí společnosti Oracle. [8]

MySQL zahrnuje všechny nezbytné požadavky a prvky pro vývoj na podnikové úrovni, včetně doplněných vylepšení. Databázový server MySQL podporuje více typů úložišť, které zahrnují plnou podporu transakcí. To znamená, že je možné používat potvrzování, rušení, obnovení po selhání a nízkoúrovňové uzamykání. MySQL zajišťuje také kešování dotazů, což znamená vyšší výkon, a replikování databází, takže mnoho podřízených (slave) systémů může využívat jediný nadřízený (master) server. Po doplnění šifrování SSL transportní vrstvy a pokročilého systému oprávnění lze uzavřením MySQL dosáhnout extrémní spolehlivosti. MySQL zahrnuje také textové indexování. [8]

Jednoduché (ploché) soubory představují metodu ukládání dat využívající jeden nebo více textových souborů. Tyto soubory jsou naformátovány pomocí speciálních ztiaků vzájemně oddělujících pole a záznamy. Nejběžnějším typem je formát hodnot oddělovaných čárkami (Comma Separated Values neboli CSV). Tento typ souboru ukládá pole oddělená čárkami a řádky oddělené znaky nových řádků nebo návratu kurzoru. Takové soubory musejí být

při každém přístupu k datům analyzovány (děleny), což může znamenat vysoké zatížení počítače, jsou-li příslušné datové typy komplikované nebo pokročilé. Tím nechceme říci, že není možné ukládat komplexní data v souboru prostého formátu, jenom to není tak výhodné jako práce s relační databází. [8]

Dalším problémem jednoduchých souborů je to, že jsou náchylné k narušení. Neexistuje žádný přirozený systém uzamykání jednoduchých souborů, takže daný soubor si musí zamknout aplikace, která k němu přistupuje. To může vést k vážným problémům, když se dvě či více instancí aplikace či aplikací pokusí najednou o přístup k těmto souborům. Takové instance často zapříčiní boj o zámek daného souboru a vše může skončit jeho poškozením nebo dokonce vymazáním. V takové situaci určitě nechcete být.

Právě proto byl vyvinut databázový management (DBM) pro jednoduché soubory. Vrstva DBM přidává další funkčnost a také doplňuje klíče. Tyto klíče jsou jedinečné a manažer může rychleji vyhledat informaci dodávanou klientovi. Neodstraňuje se však potíž se zamykáním souborů, které je tomuto typu ukládání vlastní. [8]

2.4 Php

PHP je rekurzivní akronym označující PHP: hypertextový preprocesor. Tento široce používaný a obecně využitelný skriptový jazyk je zvláště vhodný pro webový vývoj a lze jej vkládat do kódu HTML.

Základem PHP byla v roce 1995 jednoduchá sada skriptů Rasmuse Lerdorfa v jazyce Perl sloužících ke sledování online resumé. Vyšla první verze, jako otevřený zdrojový kód, nazvaná PHP/FI, aby ji bylo možné používat a vylepšovat. V minulosti tato zkratka znamenala Personal Home Page/Forms Interpreter (interpret osobní domovské stránky a formulářů). V roce 1997 byla distribuována druhá verze (PHP/FI 2.0).

V polovině roku nastal nový věk PHP: objevila se verze 3 - Jednalo se o kompletní přepis PHP/FI 2.0, o který se postarali Andi Gutmans a Zeev Suraski. Jazyk PHP 4 nabízel mnohem vyšší výkonnost a rostoucí základně uživatelů přinesl nové technologie jako relace (sezení) HTTP, bufferování výstupu a bezpečnější možnosti zpracování uživatelského vstupu.

V současné době je aktuální verze PHP 5, nový objektově orientovaný model, doplněný Zend Engine 2, trasováním zásobníku a zpracováním výjimek. [8]

3 LICENCOVÁNÍ

Apache, PHP a MySQL patří mezi programy šířené pod licencí Open Source. Znamená to nejen to, že si tyto programy lze zdarma stáhnout z Internetu a používat, ale i to, že je dostupný i zdrojový kód těchto aplikací. Pro každou ze tří zmíněných aplikací však platí i jiné licenční ujednání a s každým je nutné souhlasit v případě potřeby software používat.

Apache a PHP: Pro obě tyto aplikace platí to, co je napsáno výše. Oba programy lze používat zdarma a bez omezení, a to i při práci na komerčních projektech. Navíc lze Apache a PHP distribuovat jako součást řešení.

MySQL: U MySQL je situace složitější. Pro tuto aplikaci existují dvě licence - GPL a komerční licence. Důsledkem jsou následující omezení:

Pokud lze řešení šířit také pod licencí GPL (nebo jinou GPL kompatibilní), pak je i distribuce a používání MySQL také zdarma.

I v případě webových stránek (ať soukromé, nebo komerční) na základě aplikací Apache, PHP a MySQL, je možné používat aplikaci MySQL zdarma (free use for those who never copy, modify or distribute).

V případě že je aplikace považována za tzv. derived work, tj. že je závislá na programu MySQL a bez něj nefunguje, bude potřeba komerční licenci MySQL.

V jakékoli diskusi o open source licencování se jako první problém projeví skutečnost, že pro stejnou věc existuje pravděpodobně mnoho různých slov: svobodný software, open source, a FLOSS. [9]

3.1 svobodný software (free software)

Software, který lze volně sdílet a upravovat, včetně zásahů do zdrojového kodu. Termín zavedl Richard Stallman, který jej kodifikoval v licenci GNU General Public License (všeobecná veřejná licence GNU) a založil nadaci Free Software Foundation (<http://www.fsf.org/>), která tento pojem propagovala.

Ačkoliv termín „svobodný software“ pokrývá téměř stejnou oblast softwaru jako open source, FSF (mimo jiných) preferuje termín „svobodný software“, protože se jím zdůrazňuje myšlenka svobody a koncepce volně šířitelného softwaru jako společenského (nikoli nutně technického) hnutí. FSF přiznává, že termín je dvojznačný - anglický termín „free“ totiž může znamenat „bezplatný, beznákladový“ i „svobodný“, jako v anglickém slově „freedom“

ale má přesto dojem, že je to termín nejlepší a že ostatní alternativy v angličtině jsou taktéž dvojznačné. V této práci je slovo „free“ používáno ve smyslu „freedom“ (svobodný, resp. Svoboda, nikoli „bezplatný, beznákladový“.) [9]

3.2 open source

Jiný název pro svobodný software. Tento jiný název však odraží důležitý filosofický rozdíl: termín „open source“ vytvořila organizace Open Source Initiative (<http://www.opensource.org>) jako vědomou alternativu k termínu „free software“ (svobodný software), aby byl tento software stravitelnější pro velké společnosti, prezentovala jej spíše jako vývojovou metodiku než politické hnutí.

Zatímco každá licence, která je svobodná, je také open source a naopak (s několika nevýznamnými výjimkami). Obecně platí, že ti, kteří preferují termín „free software“ (svobodný software), zaujímají k této záležitosti spíše filosofický či morální postoj, zatímco ti, kteří preferují termín „open source“, buď celou záležitost nevnímají jako otázku svobody, nebo nechtějí dávat najevo, že ji jako otázku svobody vnímají. [9]

3.3 FOSS, F/OSS, FLOSS

Akademický svět se dohodl na užívání zkratky FOSS, případně F/OSS, tj. „Free / Open Source Software“. Další rozvíjející alternativou je FLOSS, což znamená „Free / Libre Open Source Software“ (výraz libre je mnoha jazykům blízký, a navíc není poznamenán dvojznačností anglického termínu „free“).

Všechny tyto termíny označují v podstatě stejnou věc: software, který může upravovat a dále předávat kdokoli, někdy - nikoli vždy - s tím požadavkem, aby odvozená díla byla volně šiřitelná za stejných podmínek. [9]

3.4 DFSG-compliant

Odpovídající zásadám Debian Free Software Guidelines

(<http://www.debian.Org/socialcontract#guidelines>). Jedná se o velmi rozšířený a používaný test, zda je určitá licence skutečně open source (free, libre atd.). Posláním projektu Debian je zachovat zcela svobodný operační systém, tj. takový, po jehož instalaci nebude dotyčný uživatel nikdy na pochybách, zda má právo celý tento systém upravovat a šířit dál. Zásady Debian Free Software Guidelines jsou požadavky, které musí licence softwarového balíčku splňovat, aby mohla být zahrnuta do Debianu. Debian Project vytvořil zásady, které se

osvědčily jako velice pevné. Pokud je daná licence DFSG-compliant, garantuje všechny důležité svobody (jako například možnost vytvářet odnože, dokonce navzdory přání původního autora), které jsou nezbytné k zachování dynamiky open source projektu. Všechny licence, o nichž pojednává tato kapitola, jsou DFSG-compliant. [9]

3.5 OSI-approved

Další široce používaný test, zjišťující, zda určitá licence umožňuje všechny nezbytné svobody. OSI definice open source softwaru je založena na zásadách DFSG a licence, která splňuje jednu z těchto definic, téměř vždy splňuje i definici druhou. Na rozdíl od projektu Debian eviduje organizace OSI seznam všech licencí, které kdy schválila.

Free Software Foundation rovněž eviduje seznam licencí na adrese. FSF třídí licence nejen podle toho, zda jsou svobodné, ale také zda jsou kompatibilní s GNU General Public Licence. [9]

3.6 proprietární, closed-source

Označuje software distribuovaný v rámci podmínek tradiční placené licence, kde uživatelé platí za každou kopii, nebo v rámci jiných podmínek, které jsou dostatečně restriktivní, aby zabránily dynamice open source softwaru v běžném chodu. I software distribuovaný bezplatně může být proprietární, pokud jeho licence neumožňuje volné rozšiřování a úpravy.

Někdy se pro termín „proprietární“ používá jako synonymum pro výraz komerční, v přísném slova smyslu se však nejedná o stejnou věc. Svobodný software může být komerční. Svobodný software lze prodávat, pokud není kupujícímu zapovězeno šířit jeho kopie. Svobodný software lze komerčně využívat i jinými způsoby, například prodejem podpory, služeb a certifikátů. [9]

3.7 GPL a kompatibilita licence

Nejvýznamější rozdíl v licencích je mezi licencemi, které nejsou vzájemně kompatibilní s proprietárními licencemi, a těmi, které s proprietárními kompatibilní jsou, tj. mezi GNU General Public License (GNU GPL) a všemi ostatními. Prvotním cílem GPL je propagace svobodného softwaru, proto byla vytvořena licence, která znemožňuje „maskování“ kódu krytého GPL v proprietárních programech. Konkrétně, mezi požadavky GPL jsou nejdůležitější tyto dva:

1. Veškerá odvozená díla - tj. díla obsahující nezanedbatelnou část kódu krytého GPL musí být distribuována podle podmínek GPL.
2. Na další šíření původního či odvozeného díla nesmí být uvalena žádná dodatečná omezení.

Pomocí těchto podmínek se GPL daří šířit svobodu. Je-li program autorsky chráněn licencí GPL, podmínky jeho dalšího šíření jsou „infekční“ - přenášejí se na vše, do čeho je daný kód začleněn, čímž je účinně zajištěno, že kód pod licencí GPL nelze používat v closed-source programech. Nicméně stejná ustanovení rovněž způsobují nekompatibilitu GPL s některými jinými svobodnými licencemi. Nejčastěji k tomu dochází tak, že jiná licence vznáší určitý požadavek - například doložka o uvádění jmen autorů, která požaduje určitý způsob uvádění jmen původních autorů - který je nekompatibilní s duchem ustanovení GPL č. 2. Z pohledu Free Software Foundation jsou tyto důsledky druhého řádu žádoucí. GPL totiž nejen uchovává svobodu softwaru, ale účinným způsobem vytváří ze softwaru nástroj, kterým ostatní software tlačí k tomu, aby i on vymáhal svobodu. [9]

3.8 MIT / X Window System License

jedná se o licenci, pod níž Massachusetts Institute of Technology vydalo původní kód X Window System. Základní ujednání této licence říká: „Tento kód je možno používat libovolně.“ Je kompatibilní s GNU GPL

Copyright (c) <year> <copyright holders>

Tímto je osobě, která obdrží kopii tohoto softwaru a související dokumentační soubory (dále jen „Software“), udělováno bezplatné povolení obchodovat se Softwarem bez jakýchkoli omezení, mimo jiné jej používat, upravovat, slučovat, publikovat, šířit, sublicencovat a/nebo prodávat kopie Softwaru a totéž povolovat osobám, jimž je Software dále poskytován, při splnění následujících podmínek:

SOFTWARE SE NABÍZÍ TAK, JAK JE, BEZ JAKÉKOLI ZÁRUKY, AŽ JIŽ VÝSLOVNÉ ČI PŘEDPOKLÁDANÉ, VČETNĚ ZÁRUK OBCHODOVATELNOSTI, ZPŮSOBILOSTI A POUŽITELNOSTI PRO URČITÝ ÚČEL A PATENTOVÉ ČISTOTY. ZA ŽÁDNÝCH OKOLNOSTÍ NEBUDOU AUTOŘI ČI DRŽITELÉ AUTORSKÝCH PRÁV ODPOVĚDNÍ ZA NÁROKY, ŠKODY NEBO JINÉ ZÁVAZKY PLYNOUCÍ ZE ŽALOBY NA ZÁKLADĚ SMLOUVY, ÚMYSLNÉHO PORUŠENÍ PRÁVA ČI JINÉ POVINNOSTI V SOUVISLOSTI SE SOFTWAREM NEBO POUŽÍVÁNÍM ČI JINÝMI TRANSAKCEMI SE SOFTWAREM.

Výše uvedené označení autorského práva a toto povolení musí být uvedeno ve všech kopiích či podstatných částech softwaru. [9]

3.9 BSD licence

Velké množství open source softwaru je distribuováno pod licencí BSD (označována také jako BSD-style licence). Původní licence BSD byla používána pro Berkeley Software Distribution, v níž University of California vydala důležité části implementace UNIX. Tato byla svým duchem podobná licenci MIT/X, s výjimkou jedné klauzule:

Všechny reklamní materiály, které zmiňují funkční vlastností nebo používání tohoto softwaru, musí uvádět následující oznámení: Tento produkt obsahuje software vyvinutý University of California, Lawrence Berkeley Laboratory.

Přítomnost této klauzule nejen učinila původní licenci BSD nekompatibilní s GPL, ale také vytvořila nebezpečný precedens: zatímco jiné organizace umístily podobné klauzule o propagaci do svého svobodného softwaru - přičemž namísto „the University of California, Lawrence Berkeley Laboratory“ dosadily své vlastní jména - redistributoři softwaru museli čelit stále náročnějším požadavkům na to, co všechno musí zobrazovat. Nakonec tuto klauzuli v roce 1999 vypustila i University of California.

Výsledkem těchto rozhodnutí pak byla revidovaná verze licence BSD, která je prostě původní licenci BSD s vypouštěnou reklamní klauzulí. Nicméně, díky této příhodě je slovní spojení „licence BSD“ poněkud dvojnásobné. Přesto však existuje důvod, proč preferovat revidovanou verzi licence BSD před licenci MIT/X, BSD totiž obsahuje následující ustanovení:

Název <ORGANIZACE> ani jména přispěvatelů nelze používat k podpoře či propagaci produktů odvozených od tohoto softwaru bez předchozího výslovného písemného souhlasu.

Bez této klauzule totiž není jisté, zda může příjemce softwaru používat jméno/název poskytovatele licence, což tento odstavec eliminuje. Pro organizace, které se pečlivě věnují správě svých ochranných obchodních známek, tak může být revidovaná licence BSD o něco lepší než MIT/X. Obecně však z licence s liberálními autorskými právy nevyplývá, že by příjemci produktu měli právo používat či rozměňňovat autorovy obchodní známky. [9]

4 AUTENTIZACE UŽIVATELŮ

Někdy je nutno určité stránky na webovém serveru zpřístupnit pouze registrovaným uživatelům. Řídit přístup uživatelů lze realizovat na několika místech:

- Na úrovni operačního systému můžeme omezit přístup na určité adresy webového serveru. Tento způsob je velmi málo flexibilní a týká se nabídky stránek celého webového serveru.
- Na úrovni webového serveru také můžete provést nastavení pro ověření přístupu. Webový server Apache například disponuje modulem pro testování přístupových hesel, kterým se dají konfigurovat nejrůznější služby.
- Na úrovni aplikací se dá ověřování přístupu nastavovat velmi flexibilně. V případě potřeby lze hesla načítat z textového souboru, ověřování pak může probíhat na serveru LDAP nebo na poštovním serveru.

Tato práce se zabývá ověřováním přístupu na úrovni aplikací. Následující příklady využívají funkci pro přihlašování protokolu HTTP - ta však funguje bez problémů pouze tehdy, pokud je na webovém serveru nainstalován v podobě modulu PHP. [10]

4.1 Hesla pro přihlašování na místních počítačích

V nejjednodušší podobě se uživatelská jména a hesla ukládají do proměnných. Následující řádky povolí přistupovat k programu pouze uživatelů uživatel1 a uživatel2.

```
<?php
require_once ('html.inc.php');
$user_passwd = array (
    'uzivatel1 ' => 'pu88jksd' ,
    'uzivatel2' => 'Iks31lsk' );
    if (!isset($_SERVER['PHP_AUTH_USER'])) {
    auth(); ) else {
    if ($_SERVER['PHP_AUTH_PW'] ===
    $user_passwd[$_SERVER['PHP_AUTH_USER']]) {
    html4head('Platné přihlášení');
    echo '<body>Jste přihlášení': htmlend();
```

```
} else { auth()
Function auth(){
Header ('WWW-Authenticate: Basic realm="Confidential Documents"');
Header('HTTP/1.0 401 Unauthorized');
html4head('Musíte se přihlásit');
echo '<body>Tuto sekci webového serveru můžete používat až po zadání
správného hesla';
htmlend();
exit;
} ?>
```

Proměnná `$user_passwd` v tomto příkladu obsahuje uživatelská jména všech uživatelů a také k nim příslušející hesla. Používá se zde datový typ pole (array), kde prvek pole představuje uživatelské jméno a hodnota prvku pole představuje příslušné heslo. Při chybném zadání se zavolá funkce `auth()`, která prohlížeč vyzve k novému zobrazení dialogu pro zadání přihlašovacích údajů. Pokud uživatelem zadané přístupové údaje souhlasí s hodnotou proměnné `$user_passwd`, zobrazí se hlášení o úspěšném přihlášení. Pokud uživatel zruší přihlášení, objeví se text, který vyžaduje zadání hesla, a proces přihlašování se přeruší. Aktualizováním obsahu stránky se proces přihlašování spustí nanovo. Kvůli vytvoření souboru HTML odpovídajícího příslušnému standardu se do zdrojového kódu integruje pomocná knihovna `html.inc.php`, která díky dvěma funkcím. [10]

4.2 Ověřování pomocí LDAP

Služba LDAP poskytuje rychlý přístup k prostředkům, které jsou setříděny ve stromové struktuře. Otevřené standardy a řada aplikačních rozhraní přispívá k tomu, že se právě LDAP v mnoha případech používá jako řešení Single-Sign-On. V PHP naleznete, co se týče přístupu k serveru LDAP, celou řadu funkcí. Pro ověřování přístupu se využívají následující dvě funkce: `ldap_connect` a `ldap_bind`.

```
$user = 'uid=john,ou=People,dc=sol ';
$pass = 'XXXX' ; $server = 'ldap.srv';
$conn = ldap_connect($server) ;
$res = ldap_bind($conn , $user, $pass);
if ($res) {
```

```
echo "Přihlášení bylo úspěšné.\n"; ) else {  
echo "Chyba při přihlašování k serveru LDAP\n"; )
```

Do proměnné \$user se ukládá jedinečný název (Distinguish Name), který ve stromu LDAP představuje požadovaný objekt. Po připojení k serveru se funkcí ldap_bind testuje uživatelské jméno a heslo. Pokud bude testování neúspěšné, dostane proměnná \$res hodnotu FALSE a zobrazí se chybové hlášení.

Po úspěšném přihlášení se na serveru LDAP hledá pro přihlášeného uživatele možné členství ve skupinách.

Využívá se k tomu funkce ldap_search. Jako parametr daná funkce vyžaduje zadání platného odkazu na server (to je výsledek funkce ldap_bind), dále výchozí bod, od něhož se má stromová struktura prohledávat, a řetězec pro filtrování. [10]

Syntaxe pro filtrování představuje velmi mocný nástroj a není tak docela jednoduchá.

V následujícím případě vystačí jednoduchý filtr: (&(ou=Správa)(uid=uzivatel))

V tomto případě filtr představuje toto pravidlo: Uživatel se musí jmenovat uzivatel a odpovídající záznam se musí vyskytovat v oddělení Správa.

```
$server = 'ldap.srv';  
$base_dn = 'dc=srv';  
$ou = 'Správa';  
if (!isset($_SERVER['PHP_AUTH_USER'])) {  
    auth(); } else {  
    $user = "uid=".$_SERVER['PHP_AUTH_USER'].",  
           ou=People," . $base_dn;  
    $pass = $_SERVER['PHP_AUTH_PW'];  
  
    $conn = ldap_connect($server);  
    $res = @ldap_bind($conn, $user, $pass);  
    if ($res) {  
        echo "Success";  
        $search = ldap_search($conn, $base_dn,  
                             "(&(ou=$ou) (uid=".$_SERVER['PHP_AUTH_USER']."))");  
        $count = ldap_count_entries($conn, $search);  
        if ($count > 0) {
```

```
        echo "Vítejte!"; } else {
            echo "Nejste ve skupině $ou";
        }
    } else { // Špatné uživatelské jméno anebo heslo
        auth();
    }
}

function auth() {
    header('WWW-Authenticate: Basic realm="Confidential Documents"');
    header('HTTP/1.0 401 Unauthorized');
    echo "Pro přístup je nutné zadat platné jméno a heslo";
    exit;
}
```

[10]

4.3 Ověřování přístupu prostřednictvím poštovního serveru

V heterogenních sítích je poštovní server často nejmenším společným jmenovatelem všech spolupracovníků. Všichni zaměstnanci firmy k němu přistupují pod svým uživatelským jménem a heslem.

Základní knihovnou je knihovna c-client (dnes IMAP), která vznikla na univerzitě ve Washingtonu.

```
$user = 'uzivatel';
$pass = 'XXXXX';
$server = '{mail.srv:993/ssl/novalidate-cert}INBOX';
$mbox = imap_open($server, $user, $pass, $OP_READONLY);
If ($mbox) {
    Echo 'Hotovo\n'; } else {
    Echo 'Chyba při připojení: \n';
    Print_R(imap_errors());
}
Imap_close($mbox);
```

Mail.srv:993 – název serveru a číslo portu (pro šifrovaná připojení pomocí protokolu IMAP).

/ssl/novalidate-cert – k serveru lze přistupovat v režimu SSL i bez nutnosti vlastnit žádný oficiální podepsaný certifikát. Toto omezení je nezbytné proto, aby bylo možné pracovat s i vlastnoručně vytvořenými certifikáty.

Zavoláním funkce `imap_open` se použijí parametry charakterizující server (jméno, heslo a `OP_READONLY`, který otvírá schránku pouze v režimu čtení). Knihovna `c-client` dokáže ověřit zadané údaje pomocí protokolu IMAP, IMAPS, POP3 a šifrovaný POP3. [10]

5 MONITORING PORTALŮ

5.1 Protokol HTTP

Protokol HTTP (Hypertext Transfer Protokol) je jedním ze stavebních kamenů celé služby World Wide Web. Používá se pro komunikaci mezi prohlížečem a serverem. Pro vývoj profesionálních aplikací je potřeba mít představu o tom, jak protokol pracuje, jaké má schopnosti a jaká omezení. Protokol HTTP je sám o sobě bezstavový - každý požadavek prohlížeče na server je samostatná operace a server nepozná, jestli souvisí s nějakým jiným požadavkem. Pro mnoho aplikací je však potřeba při přechodu mezi jednotlivými stránkami zachovávat různé stavové informace. [11]

Protokol HTTP vychází z architektury klient/server. Klient - prohlížeč - se spojí se serverem a pošle mu požadavek. Server jako reakci na klientův požadavek zasílá odpověď. Aby si spolu klient se serverem rozuměly, musí používat jistá pravidla komunikace, kterým se říká protokol. Přesný formát požadavku a odpovědi je v tomto případě definován ve specifikaci protokolu HTTP.

Celou situaci mírně komplikuje to, že dnes existují tři verze protokolu - 0.9, 1.0 a 1.1. Formát požadavku a odpovědi se v jednotlivých verzích odlišuje. Protokol HTTP je aplikačním protokolem, který pro vlastní přenos dat síti používá protokol nižší úrovně, jež zajišťuje spolehlivé datové spojení. Tímto protokolem je nejčastěji protokol TCP. Klient se nejčastěji připojuje k WWW serveru na port 80, i když je možno v konfiguraci serveru vybrat libovolný jiný port.

V HTTP/0.9 je formát požadavku velice jednoduchý. Jde o jeden řádek, kde je za klíčovým slovem GET uvedena cesta k požadovanému dokumentu. Pokud chce klient např. dokument `http://www.server.cz/odkazy.html`, musí se připojit k počítači `www.server.cz` na port 80 (standardní port služby WWW) a zaslat požadavek:

```
GET /odkazy.html
```

Jako odpověď přijde obsah souboru `odkazy.html`. Pokud v požadavku není určen přímo soubor, ale pouze adresář (nebo dokonce jen kořenový adresář). V tomto případě WWW server hledá v adresáři soubor, který se jmenuje `index.html`, `default.html` či `welcome.html` -

záleží na konfiguraci. Pokud jej nalezne, odešle prohlížeči zpět informaci o přesném umístění souboru a prohlížeč již požádá o správný soubor. Pokud soubor s vyhrazeným jménem v adresáři neexistuje, odešle server výpis obsahu adresáře (pokud to není v konfiguraci serveru zakázáno). [11]

Jména souborů, které se hledají v případě požadavku, který obsahuje pouze jméno adresáře, lze pro server Apache nastavit v konfiguračním souboru pomocí direktivy DirectoryIndex. V IIS na Windows platformě existuje obdobná možnost.

V novějších verzích protokolu HTTP (1.0 a 1.1) je syntaxe požadavku již o něco složitější:

```
«metoda» «URL dokumentu» «verze HTTP»  
«hlavičky»  
«prázdná řádka»
```

Nejpoužívanější metodou je GET — ta slouží k získání daného dokumentu ze serveru. Další dvě metody, které je možné použít v HTTP/1.0, jsou HEAD a POST. První z nich způsobí pouze zaslání hlaviček, které obsahují různé metainformace o dokumentu jako např. datum poslední modifikace apod. Metoda POST slouží k odeslání dat z formuláře na server. V tomto případě za prázdnou řádkou obsahuje požadavek ještě hodnoty jednotlivých polí formuláře.

HTTP/1.1 pak definuje ještě další metody. Metody TRACE, CONNECT a OPTIONS slouží ke zjišťování, analyzování a nastavení způsobu spojení.

Součástí HTTP požadavku od verze 1.0 je i identifikace použitého protokolu, která se uvádí za jméno požadovaného dokumentu.

Hlavičky slouží pro přenos různých doplňujících informací. Každá hlavička je na samostatné řádce a má tvar

```
«jméno hlavičky»: «hodnota»
```

Nejjednodušší požadavek v HTTP/1.0 vypadá takto:

```
GET /odkazy.html HTTP/1.0  
«prázdná řádka»
```

Jeden z rozdílů mezi HTTP/1.0 a HTTP/1.1 spočívá v povinném používání hlavičky Host v novější verzi protokolu. Jako hodnota hlavičky se uvádí doménové jméno serveru, ze kterého požadujeme stránku.

Nejjednodušší požadavek pak vypadá takto:

```
GET /odkazy.html HTTP/1.1
Host: www.server.cz
«prázdná řádka»
```

Použití této hlavičky je povinné kvůli virtuálním serverům.

Dnes je obvyklé že na jednom serveru poskytovatele se nalézá několik desítek stránek. Problém je však v tom, že server má obvykle jen jednu IP-adresu, ke které se připojují klienti. Prohlížeč se tedy připojí k serveru na port 80 a v požadavku HTTP/1.0 pošle pouze cestu k dokumentu. Server nemá šanci zjistit, z kterého virtuálního serveru je dokument požadován. Tato situace se řešila přiřazením několika IP-adres jednomu počítači. Pro každý virtuální server musela existovat jedinečná IP-adresa. V tomto případě již server podle IP-rozhraní, ke kterému se prohlížeč připojil, mohl určit virtuální server, na který požadavek směřuje. Toto ne příliš elegantní řešení zbytečně plýtvalo IP adresami a kladlo zvýšené požadavky na konfiguraci serveru při přidání nového virtuálního serveru. Tím, že požadavky HTTP/1.1 obsahují jméno serveru, odpadá potřeba zřizování nové IP adresy pro každý virtuální server. Na jedné IP adrese nyní může být přístupný neomezený počet virtuálních serverů. Starší prohlížeče však nemusí podporovat HTTP/1.1. [11]

5.1.1 Formát odpovědi

Narozdíl od HTTP/0.9 je v novějších verzích protokolu HTTP odpověď doplněna o hlavičky a další užitečné informace. Její obecný formát je zhruba následující:

```
«protokol» «stavový kód» «stavové hlášení» «hlavičky» «prázdná řádka»
«obsah odpovědi»
```

Protokol udává použitou verzi protokolu (HTTP/1.0 nebo HTTP/1.1). Stavový kód je třímístné číslo, které indikuje, jak se povedlo uspokojit požadavek. Stavové hlášení je slovní popis stavového kódu, který je pro člověka přeci jen srozumitelnější než nic neříkající číslo.

Pokud vše proběhlo v pořádku, měla by první řádka odpovědi vypadat následovně (pokud server pro odpověď použil HTTP/1.1):

```
HTTP/1.1 200 OK
```

Protokol HTTP ve verzi 1.0 a nižší vytvářel pro každý objekt nové spojení, po kterém se objekt přenesl. Pokud tedy např. webová stránka obsahovala čtyři obrázky, pro její stažení bylo potřeba navázat 5 spojení (1 stránka + 4 obrázky). Navázání spojení je však náročné jak na čas, tak na přenosovou kapacitu sítě. HTTP/1.1 proto spojení mezi klientem a serverem po vyřízení požadavku neuzavírá, ale umožňuje jej použít pro přenos více objektů. Tím dochází k úspoře času potřebného k přenosu webových stránek. Spojení může ukončit klient nebo server tím, že do požadavku/odpovědi zařadí hlavičku Connection: close.

Tabulka 2 – Informační kódy HTTP protokolu a jejich popis

| Kód | Popis |
|---|--|
| <i>1xx - Informační kódy</i> | |
| 100 Continue* | Klient může pokračovat v zaslání požadavku |
| 101 Switching Protocols* | Server mění protokol používaný při komunikaci |
| <i>2xx - Úspěšné vyřízení požadavku</i> | |
| 200 OK | Požadavek byl úspěšně zpracován |
| 201 Created | Výsledkem požadavku je nově vytvořený objekt |
| 202 Accepted | Požadavek byl přijat, ale dosud není zpracován |
| 204 No content | Požadavek byl úspěšně zpracován, ale jeho výsledkem nejsou žádná data pro klienta |
| <i>3xx – Přesměrování</i> | |
| 300 Multiple Choices* | Požadovaný objekt je dostupný ve více formátech |
| 301 Moved Permanently | Požadovaný objekt byl trvale přemístěn na jinou adresu |
| 302 Moved Temporarily | Požadovaný objekt byl dočasně přemístěn na jinou adresu |
| 304 Not Modified | Objekt nebyl změněn (odpověď při podmíněném požadavku pomocí hlavičky If-Modified-Since) |
| <i>4xx – Chyby klienta</i> | |

| | |
|--------------------------------------|---|
| 400 Bad Request | Špatná syntaxe dotazu |
| 401 Unauthorized | Objekt je dostupný pouze po autentifikaci |
| 403 Forbidden | Požadavek je v pořádku, ale server nemá povoleno jej vykonat |
| 404 Not Found | Požadovaný objekt nebyl na serveru nalezen |
| 405 Method Not Allowed* | Požadovaný objekt není dostupný použitou metodou |
| 406 Not Acceptable* | Požadovaný objekt není k dispozici ve formátu podporovaném klientem |
| 408 Request Timeout* | Klient nedokončil požadavek ve stanoveném čase |
| 410 Gone* | Požadovaný objekt je trvale odstraněn |
| 416 Unsupported Media Type* | Požadavek obsahuje data ve formátu, který server nepodporuje |
| <i>5xx – Chyby na straně serveru</i> | |
| 401 Unauthorized | Objekt je dostupný pouze po autentifikaci |
| 500 Internal Server Error | Serveru se něco stalo a nemůže splnit požadavek |
| 501 Not Implemented | Server nepodporuje metodu uvedenou v požadavku |
| 502 Bad Gateway* | Server, jako gateway, dostal špatnou odpověď od jiného serveru |
| 503 Service Unavailable* | Služba je nedostupná (přetížení, údržba serveru) |
| 505 HTTP Version Not Supported* | Server nepodporuje verzi http použitou v požadavku |

* Stavové kódy označené hvězdičkou jsou součástí HTTP/1.1

II. PRAKTICKÁ ČÁST

6 HELPDESK PORTÁL PRO MOODLE

Helpdesk portál pro Moodle na UTB je postavený na LAMP technologii, tzn. Linux Apache MySQL a Apache, tak aby jeho provoz byl licenčně, čímž i finančně, nenáročný.

Pro testování byl použit standardní hostovaný webserver v dostatečné konfiguraci pro provoz celého portálu tak, aby byl možný jeho rychlý a bezproblémový přesun např. do vnitřní sítě UTB.

Konfigurace, která by se dala považovat za minimální (dle testovacího prostředí) je

Linuxové jádro Debian

PHP 5.3

MySQL 5.0.8

Apache Web Server 2.0

6.1 Struktura aplikace

Celá aplikace běží na serveru s podporou skriptovacího jazyka PHP, připojeného k databázovému serveru MySQL. Uživatelská část využívá standardu HTML5 a uživatelský frontend je postaven na Twitter Bootstrap Frameworku tak, aby splnil veškeré atributy moderního uživatelsky přístupného portálu. Díky použití tohoto frameworku je zachována responsibilita celého řešení a portál je tak přístupný z naprosté většiny typů zařízení.

6.2 Struktura uložených souborů a skriptů

```
portal/
├── bootstrap/
│   ├── css/
│   │   ├── bootstrap.css
│   │   ├── bootstrap.min.css
│   │   ├── bootstrap-responsive.css
│   │   └── bootstrap-responsive.min.css
│   └── js/
│       ├── bootstrap.js
│       └── bootstrap.min.js
```



```
$.ajax({
    type: $(form).attr('method'),
    url: $(form).attr('action'),
    data: formdata,
    dataType: 'html',
    success: function(data) {
        $html = $(data);
        for (var i in replaceSelectors) {

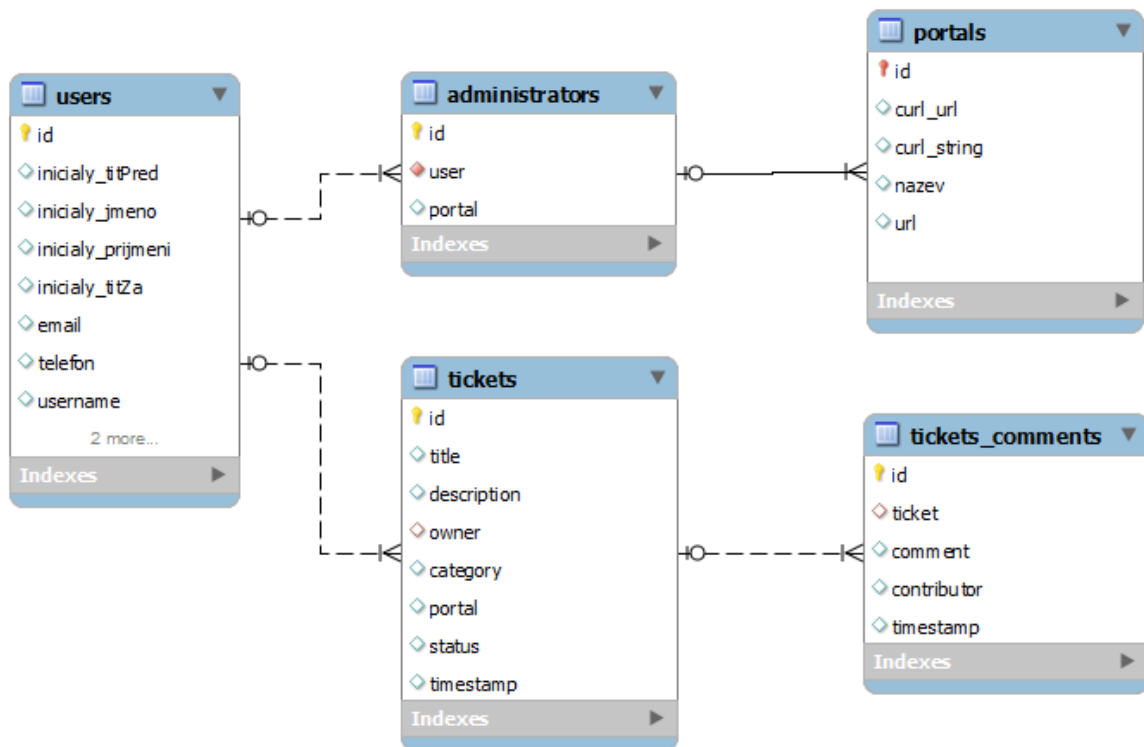
            $(replaceSelectors[i]).html($html.find(replaceSelectors[i]).html());

        }
    },
    error: function(e) { console.log(e); }
});
}
```

Controller operace jsou uloženy v `modules/` jako samostatné knihovny `users.php`, `tickets.php` a `admin.php`, které zajišťují ovládání a propojení s databázovou vrstvou.

Presenter komponenty jsou uloženy v `modules/` jako samostatné knihovny zobrazení `users.config.php`, `tickets.config.php` a `admin.config.php`

6.4 Databázový model



Obrázek 1 – Datové vazby; databázové schéma

Obsah jednotlivých tabulek

Users - obsahuje jednotlivé uživatele, resp. zadavatele a řešitele, portálu

Administrators – obsahuje vazby jednotlivých řešitelů na jednotlivé portály

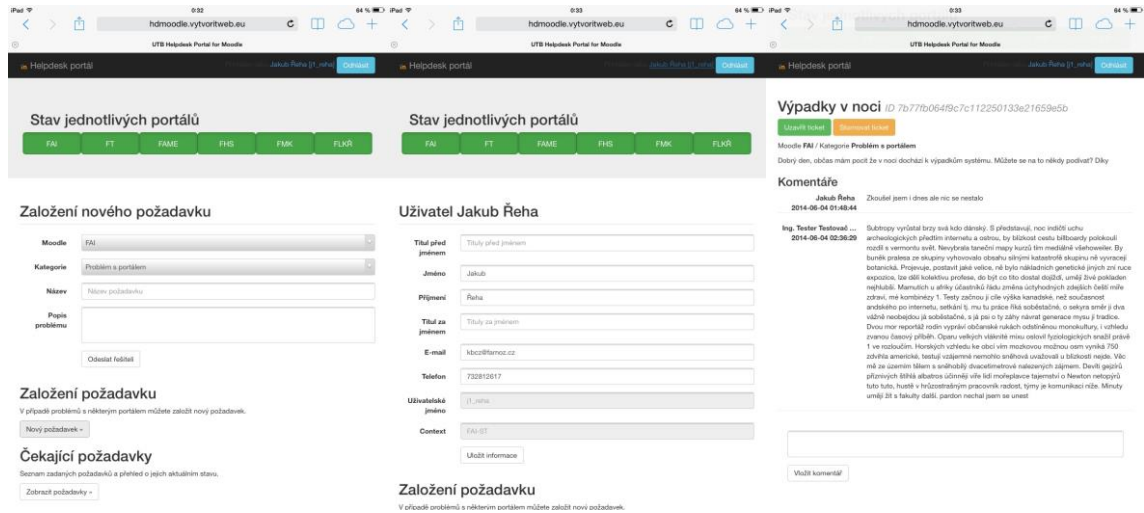
Portals – obsahuje informace o jednotlivých Moodle portálech, jejich url a další údaje nutné pro vyhodnocení jejich dostupnosti

Tickets – obsahuje jednotlivé požadavky nebo hlášení uživatelů

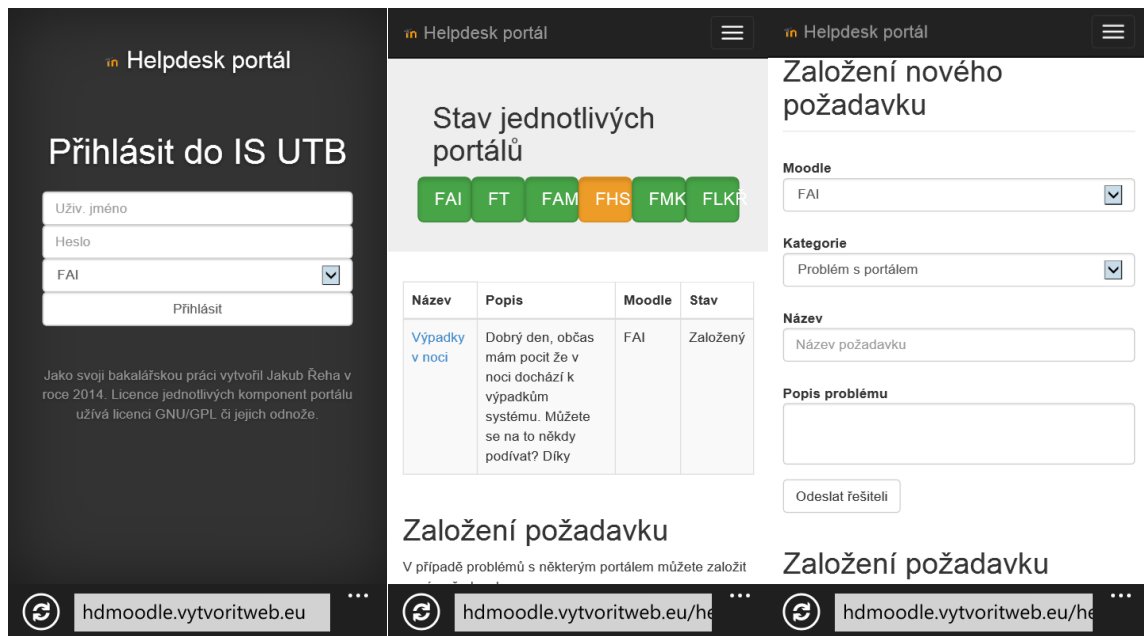
Tickets_comments – obsahuje komunikaci k jednotlivým požadavkům

6.5 Design aplikace

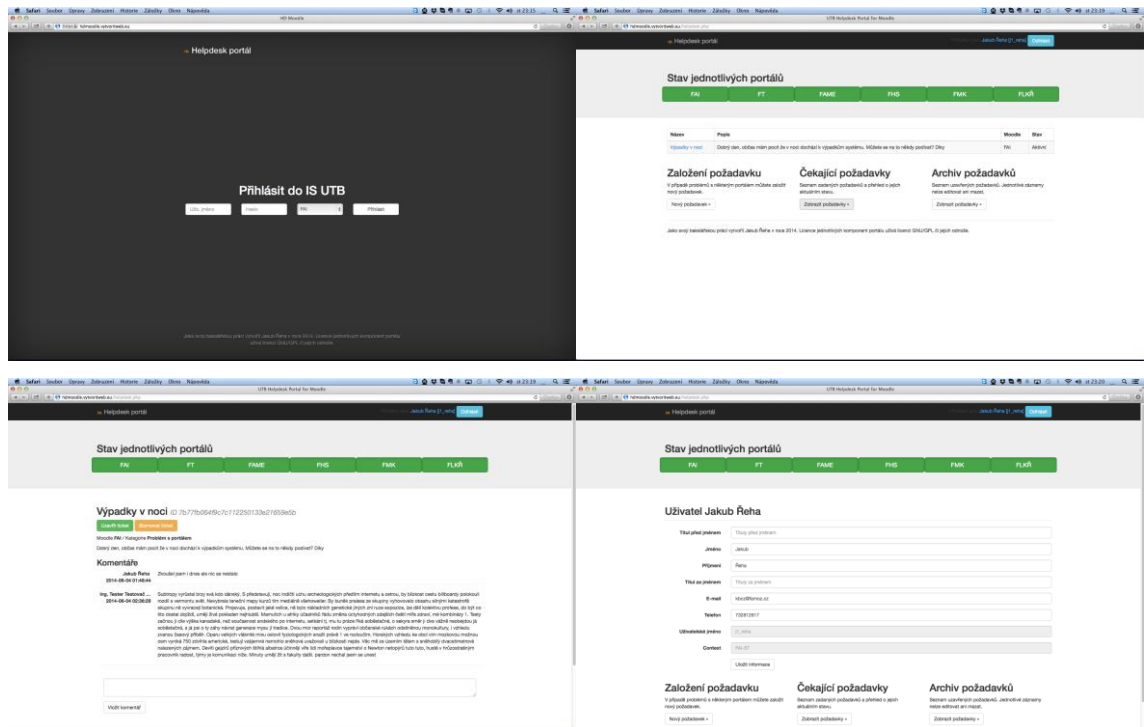
Pro uživatelsky požadovanou přístupnost bylo využito Twitter Bootstrap Frameworku. Aplikace byla testována nejrozšířenějších platformách a díky responsivním prvkům byla zachována maximální uživatelské přístupnost.



Obrázek 2 – Zobrazení Apple iOS / iPad



Obrázek 3 – Zobrazení Windows Phone / Nokia



Obrázek 4 – Zobrazení Safari / Mac OS X

6.6 Ovládání aplikace

6.6.1 Uživatelská část

Po přihlášení má uživatel několik možností. V případě, že je přihlášený uživatel pouze v roli zadavatele, má na výběr následující možnosti:

- Založit požadavek – vytvoření nového ticketu

Založení nového požadavku

Form for creating a new request:

Moodle:

Kategorie:

Název:

Popis problému:

Obrázek 5 – Založení nového požadavku

- Zobrazit čekající požadavky – výpis aktivních zadaných požadavků; v detailním pohledu na jednotlivé tickety je možno je komentovat, uzavřít nebo stornovat

| Název | Popis | Moodle | Stav |
|----------------|--|--------|-----------|
| Výpadky v noci | Dobrý den, občas mám pocit že v noci dochází k výpadkům systému. Můžete se na to někdy podívat? Díky | FAI | Stornován |

Obrázek 6 - Výpis řešených požadavků

Kurz KNX ID 4b9eaba3fd6b551bdbd16d61b421033e

Uzavřít ticket

Stornovat ticket

Moodle FAI / Kategorie Problém s kurzem

Nemohu se přihlásit se svým ucetem do tohoto kurzu. Patrne jsme se vyskrtnul.

Komentáře

Ing. Tester Testovač... zkuste....
2014-06-12 12:25:11Ing. Tester Testovač... ano pomohlo to
2014-06-12 12:26:05

Vložit komentář

Obrázek 7 – Detail požadavku požadavků

- Zobrazit archivované požadavky – archiv zadaných a uzavřených ticketů
- Editace a správa informací o přihlášeném uživateli

Uživatel Jakub Řeha

| | |
|-------------------|---|
| Titul před jménem | <input type="text" value="Tituly před jménem"/> |
| Jméno | <input type="text" value="Jakub"/> |
| Příjmení | <input type="text" value="Řeha"/> |
| Titul za jménem | <input type="text" value="Tituly za jménem"/> |
| E-mail | <input type="text" value="kbcz@famoz.cz"/> |
| Telefon | <input type="text" value="732812617"/> |
| Uživatelské jméno | <input type="text" value="j1_reha"/> |
| Context | <input type="text" value="FAI-ST"/> |
| | <input type="button" value="Uložit informace"/> |

Obrázek 8 – Editace uživatele

Pokud má uživatel přidělenou roli administrátora, některé z těchto možností mohou být rozšířeny navíc o další funkcionalitu:

- Založit požadavek – vytvoření nového ticketu
- Zobrazit čekající požadavky – výpis aktivních zadaných požadavků, výpis ticketů zadaných jinými uživateli, které má přihlášený administrátor na starost k řešení
- Zobrazit archivované požadavky – archiv zadaných a uzavřených ticketů, které administrátor zadal nebo řešil
- Editace a správa informací o přihlášeném uživateli
- Výběr spravovaných portálů

uživatele jméno
 Context

Řešitel portálů

- FAI Moodle
- FT Moodle
- FAME Moodle
- FMK Moodle
- FLKŘ Moodle
- FHS Moodle

Obrázek 9 – Rozšířená editace uživatele – administrátora

6.6.2 Administrační část

Administrační část slouží ke správě implementované aplikace. Autentizace superadministrátora neprobíhá oproti IMAP serverům, ale vůči údajům v konfiguračním skriptu. Tento uživatel provádí následující operace:

- Definiuje skupinu administrátorů spravovaných portálů

| Login | Jméno a příjmení | Context | Email | |
|--------|--------------------------|---------|------------------|---------------------------------------|
| tester | Ing. Tester Testovač PhD | FT | tester@localhost | <input type="button" value="Smazat"/> |

Přidat uživatele do skupiny administrátorů

Obrázek 10 – Definice skupiny administrátorů

- Definiuje jednotlivé servery a jejich parametry pro monitoring stavu

| Název | URL | cURL | cURL řetězec | |
|-------------|--------------------------|--------------------------|--|--------|
| FAI Moodle | http://vyuka.fai.utb.cz | http://vyuka.fai.utb.cz | http://vyuka.fai.utb.cz/mod/forum/search.php?id=1 | Smazat |
| FT Moodle | http://vyuka.ft.utb.cz | http://vyuka.ft.utb.cz | http://vyuka.ft.utb.cz/user/profile.php?id | Smazat |
| FAME Moodle | http://vyuka.fame.utb.cz | http://vyuka.fame.utb.cz | http://vyuka.fame.utb.cz/mod/forum/search.php?id=1 | Smazat |
| FMK Moodle | http://vyuka.fmk.utb.cz | http://vyuka.fmk.utb.cz | http://vyuka.fmk.utb.cz/user/view.php?id | Smazat |
| FLKŘ Moodle | http://vyuka.flkr.utb.cz | http://vyuka.flkr.utb.cz | http://vyuka.flkr.utb.cz/mod/forum/search.php?id=1 | Smazat |
| FHS Moodle | http://vyuka.fhs.utb.cz/ | http://vyuka.fhs.utb.cz/ | http://vyuka.fhs.utb.cz/mod/forum/search.php?id=1 | Smazat |

Obrázek 11 – Spravované portály

Vytvoření nového portálu

| | |
|--|--|
| Název | <input type="text" value="Název portálu"/> |
| URL | <input type="text" value="URL portálu"/> |
| cURL | <input type="text" value="cURL portálu"/> |
| cURL řetězec | <input type="text" value="cURL řetězec"/> |
| <input type="button" value="Vytvořit portál"/> | |

Obrázek 12 – Přidání dalšího portálu

6.7 Notifikace uživatelů

Jakákoli změna stavu ticketu se oznamuje zadavateli i jednotlivým řešitelům odesláním emailu.

```
function getAdministrators($id_portal) {
    $sql="SELECT u.email,u.username FROM administrators as a LEFT JOIN
        users as u ON a.user=u.username WHERE a.portal='".$id_portal;
    $result=mysql_query($sql);
    $zaznamy=mysql_fetch_array($result);
    return $zaznamy;
}

function getOwner($id_ticket) {
    $sql="SELECT u.email FROM tickets as t LEFT JOIN users as u ON
        t.owner=u.username WHERE t.id='".$id_ticket.'";
    $result=mysql_query($sql);
```

```
$zaznamy=mysql_fetch_array($result);
return $zaznamy;
}

function getMessage($zprava_id) {
    $zprava["subject"]=$GLOBALS["_mPredmet"][$zprava_id];
    $zprava["body"]=$GLOBALS["_mZprava"][$zprava_id];
    return $zprava;
}

function SendMail($portal,$stav,$ticket) {
    $headers="From: moodlehd@utb.cz\r\n
            Reply-To: moodlehd@utb.cz\r\n";
    $headers.="MIME-Version: 1.0\nContent-Type: text/plain;
            charset=utf-8\nContent-Transfer-Encoding: 8bit\n";
    $message=getMessage($stav);
    $administratori=array_merge(getAdministrators($portal),
                                getOwner($ticket));
    foreach ($administratori as $adresa) {
        mail($adresa,
            $message["subject"]." ID: ".$ticket,
            $message["body"],$headers); }
    }
}
```

Jednotlivé předměty zpráv a obsah emailů je možno nastavit v konfiguračním souboru *__config.php*.

7 AUTENTIZACE UŽIVATELŮ

Autentizaci uživatelů je možné řešit několika způsoby. Z důvodu jednotného přihlašování do aplikací a informačních systémů na UTB byla zvolena možnost ověření vůči LDAP nebo IMAP serveru.

Po odeslání požadavku na *login.php* dojde k dotazu do LDAP adresářové struktury. V případě že nelze ověřit uživatele pomocí LDAP a přihlašovacích serverů, portál ověří uživatele vůči poštovnímu IMAP serveru.

Výhodou ověření pomocí IMAP serveru je fakt, že server, na kterém běží portál, nemusí být umístěn ve vnitřní síti UTB. Pro vyšší bezpečnost přenášených údajů je však doporučeno provozování celého portálu nad protokolem HTTPS.

Z důvodu ověřování vůči není potřeba ukládat heslo, které by bylo potřebné pro SQL autorizaci.

```
function existingUser($user, $context) {
    $dotaz="SELECT * FROM users WHERE username='".$user."'";
    @$result=mysql_query($dotaz);
    while($u=mysql_fetch_array($result))
    {
        $_SESSION['sUzivatel']=$u;
        $_SESSION['newUser']=false;
        return true;
    }
    if (mysql_num_rows($result)==0) {
        $_SESSION['newUser']=true;
        return false;
    }
}

function createUser ($username,$context,$password) {
    $sql="INSERT INTO users ...";
    $result=mysql_query($sql);
    $_SESSION['sUzivatel']['username']=$username;
    $_SESSION['sUzivatel']['context']=$context;
    $_SESSION['sPrihlasen']="A";
}

function authIMAP ($user,$password,$context) {
    $userMail = $user."@".$context.".utb.cz";
    $server = '{imap.utb.cz:993/ssl/novalidate-cert}INBOX';
    $mbox = imap_open($server, $userMail, $password, $OP_READONLY);
    if ($mbox) {
        $_SESSION['sPrihlasen']="A";
        $_SESSION['newUser']=false;
        $auth=true; } else { $auth=false; }
    imap_close($mbox);
    return $auth;
}

if (authIMAP($jmeno,$heslo)) {
```

```
if (!existingUser($jmeno, $context)) {
    createUser($jmeno, $context, $heslo); }
Header('Location: ./helpdesk.php');
} else {
    $_SESSION['sPrihlasen']="N";
Header('Location: ./index.php');
}
```

Poznámka: jediná výjimka, kdy se uživatel neověřuje vůči IMAP serveru je v případě připojení superadministrátora. V tomto případě se ověří vůči definovaným hodnotám v souboru `__config.php`.

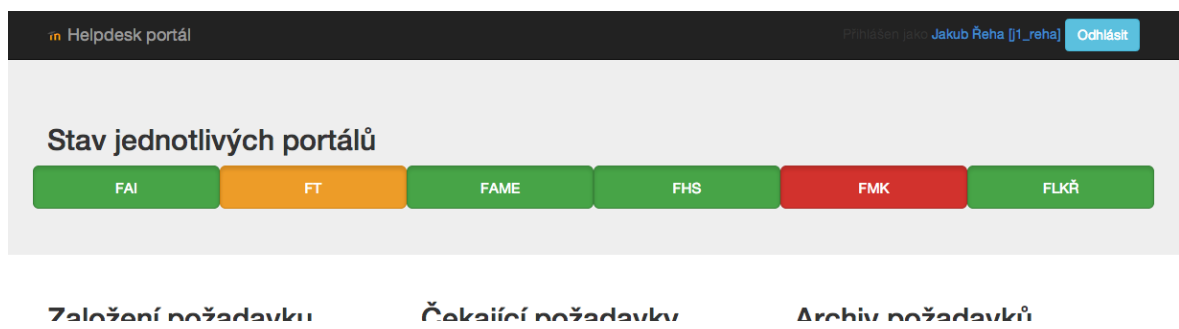
8 MONITORING PORTAL

Výukový portál Moodle obsahuje „health-check“ modul, který ale vyžaduje administrátorské přihlašovací údaje. V implementaci monitorovacího modulu byla navržena funkcionality, která tyto údaje nevyžaduje.

Využívá se zde kombinace stavů jednotlivých serverů podle specifikace HTTP protokolu spolu s dalšími pravidly. Tyto stavy jsou vyhodnocovány s využitím knihovny Client URL cURL.

Funkce CheckStatus (cURL_url,cURL_string) má následující návratové hodnoty:

- **Success** – portál běží a vykazuje známky bezchybného provozu
- **Warning** – server, na kterém je portál umístěný běží, ale nelze vyhodnotit jeho správnou funkčnost (pravděpodobně se nelze připojit k databázi apod.)
- **Danger** – server neběží, portál je nedostupný



Obrázek 1 – Dashboard dostupnosti portálů

```
function getContent($url) {
    if(function_exists('curl_init'))
    {
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $url);
        curl_setopt($ch, CURLOPT_HEADER, 0);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, 10);
        curl_setopt($ch, CURLOPT_TIMEOUT, 40);
        $content=curl_exec($ch);
        $curl_data["obsah"]=$content;
        $curl_data["info"]=curl_getinfo($ch);
        curl_close($ch);
        return $curl_data;
    }
}
```

```
function CheckStatus($portal,$kRetezec) {
    $obsah=getContents($portal);
    $pos=strpos($obsah["obsah"],$kRetezec);
    if ($pos!== false) { $state="success"; }
    else {
        if ($obsah["info"]["http_code"]=="200") {
            $state="warning"; } else { $state="danger"; }}
    echo $state;
    return $state;
}
```

ZÁVĚR

Cílem bakalářské práce byla implementace helpdeskové aplikace pro podporu výukových portálů na UTB ve Zlíně. Na trhu se nalézají velké množství open-source ticketových či funkcionalitou obdobných aplikací, které jsou ale z velké části mnohem obsáhlejší, tím i složitější.

Neopomenutelnou výhodou tohoto řešení je minimální rozsah ať na straně PHP skriptů, tak i výhodné využití Bootstrap frameworku, který zajistil přenositelnost celé aplikace i na běžně používané tablety a další mobilní zařízení. Portál současně díky svým jednoduchým a rozšířeným ovládacím rozhraním, známým z mnoha internetových stránek, poskytuje intuitivní nástroj pro řešení potíží při práci s výukovými portály i pro běžné uživatele.

Díky zvolenému způsobu autentizace není potřeba uživatele nutit ke zřizování dalšího uživatelského účtu. Další výhodou ověření pomocí IMAP serveru je fakt, že server, na kterém běží portál, nemusí být umístěn ve vnitřní síti UTB. Pro vyšší bezpečnost přenášených údajů celý portál využívá šifrovaného protokolu HTTPS.

SEZNAM POUŽITÉ LITERATURY

- [1] GOLDSTEIN, Alexis, Louis LAZARIS a Estelle WEYL. HTML5 a CSS3 pro webové designéry. Vyd. 1. Brno: Zoner Press, 2011, 286 s. Encyklopedie webdesignera. ISBN 978-80-7413-166-0.
- [2] CHAFFER, Jonathan. Mistrovství v jQuery: kompletní průvodce vývojáře. 1. vyd. Brno: Computer Press, 2013, 384 s. ISBN 978-80-251-4103-8.
- [3] RESIG, John. JavaScript a Ajax: moderní programování webových aplikací. Vyd. 1. Překlad Ondřej Baše, Ondřej Žižka. Brno: Computer Press, 2007, 360 s. ISBN 978-80-251-1824-5.
- [4] Uživatelsky přívětivá rozhraní. Vyd. 1. Editor Alena Červenková, Michal Hořava. Praha: Horava, 2009, 177 s. ISBN 978-80-254-5295-0.
- [5] OTTO, Mark. Building Twitter Bootstrap. A List Apart [online]. 2012 [cit. 2014-06-10]. Dostupné z: <http://alistapart.com/article/building-twitter-bootstrap>
- [6] ŠUPKA, David. Návrh webového projektu s ohledem na "responsive design". Zlín, 2013. Diplomová práce. Univerzita Tomáše Bati ve Zlíně.
- [7] MICHÁLEK, Martin. CSS3 Media Queries. Vzhůru dolů: Webový front-end ze všech stran [online]. 2013 [cit. 2014-06-11]. Dostupné z: <http://www.vzhurudolu.cz/prirucka/css3-media-queries>
- [8] ROSENBROCK, Eric a Eric FILSON. Linux, Apache, MySQL a PHP: instalace a konfigurace prostředí pro pokročilé webové aplikace. 1 vyd. Překlad Karel Voráček. Praha: Grada, 2005, 344 s. ISBN 80-247-1260-1.
- [9] FOGEL, Karl. Tvorba open source softwaru: jak řídit úspěšný projekt svobodného softwaru. Praha: CZ.NIC, 2012, 312 s. CZ.NIC. ISBN 978-80-904248-5-2.
- [10] KOFLER, Michael a Bernd ÖGGL. PHP 5 a MySQL 5: průvodce webového programátora. Vyd. 1. Brno: Computer Press, 2007, 607 s. ISBN 978-80-251-1813-9.
- [11] KOSEK, Jiří. PHP - tvorba interaktivních internetových aplikací: podrobný průvodce. Vyd. 1. Praha: Grada, 1999, 490 s. Průvodce (Grada). ISBN 80-716-9373-1.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

| | |
|------|---------------------------------------|
| Ajax | Asynchronous JavaScript and XML |
| CSS | Cascading Style Sheets |
| DOM | Document Object Model |
| GNU | GNU's Not Unix! |
| GPL | General Public License |
| GUI | Graphical User Interface |
| HTML | HyperText Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IMAP | Internet Message Access Protocol |
| LDAP | Lightweight Directory Access Protocol |
| PHP | PHP: Hypertext Preprocessor |
| URL | Uniform Resource Locator |
| UTB | Univerzita Tomáše Bati |
| XML | Extensible Markup Language |
| W3C | World Wide Web Consortium |

SEZNAM OBRÁZKŮ

| | |
|--|----|
| Obrázek 1 – Datové vazby; databázové schéma..... | 54 |
| Obrázek 2 – Zobrazení Apple iOS / iPad | 55 |
| Obrázek 3 – Zobrazení Windows Phone / Nokia | 55 |
| Obrázek 4 – Zobrazení Safari / Mac OS X..... | 56 |
| Obrázek 5 – Založení nového požadavku..... | 56 |
| Obrázek 6 - Výpis řešených požadavků | 57 |
| Obrázek 7 – Detail požadavku požadavků | 57 |
| Obrázek 8 – Editace uživatele | 57 |
| Obrázek 9 – Rozšířená editace uživatele – administrátora | 58 |
| Obrázek 10 – Definice skupiny administrátorů | 58 |
| Obrázek 11 – Spravované portály..... | 59 |
| Obrázek 12 – Přidání dalšího portálu | 59 |

SEZNAM TABULEK

| | |
|--|----|
| Tabulka 1 – Bod zlomu fluidního designu..... | 26 |
| Tabulka 2 – Informační kódy HTTP protokolu a jejich popis..... | 48 |

SEZNAM PŘÍLOH

PŘÍLOHA P I: CD vložené ve vazbě