Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Redakční systém pro kompletní dynamickou správu webu

Jiří ZECHMEISTER

Bakalářská práce
2009

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Katedra informačních technologií
Akademický rok: **2008/2009**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Jiří ZECHMEISTER**

Studijní program: **B2646 Informační technologie**

Studijní obor: **Informační technologie**

Název tématu: **Redakční systém pro komplexní dynamickou správu webu**

Zásady pro vypracování:

Cílem práce bude vytvoření aplikace pro komplexní a dynamickou správu webových stránek
Teoretická část:
V teoretické části bakalářské práce budou představeny a zhodnoceny používané technologie webových aplikací jako jsou zejména PHP, MySQL a moderní technologie AJAX. Bude proveden úvod do problematiky redakčních systémů. Bude provedena analýza problému a navrženo vhodné optimální řešení.
Implementační část:
Aplikace bude obsahovat správu uživatelských souborů, textového obsahu stránek, dynamickou správu menu, správu galirií a správu uživatelských účtů. Systém bude implementován v jazyce PHP za použití technologie Ajax (Asynchronní JavaScript). Datová část bude realizována relační databází typu MySQL nebo Oracle.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

Jesus Castagnetto, Haris Rawat, Sascha Schumann, Chris Scollo, Deepak Veliath: PHP – Programujeme profesionálně, Computer press Brno 2004.

Christian Darie, Bogdan Brinzarea, Filip Chereches-Tosa, Mihai Bucica: AJAX a PHP – tvoříme interaktivní webové aplikace profesionálně, Zoner press 2006.

Vedoucí bakalářské práce: **Ing. Zdeněk Šilar**
Katedra informačních technologií

Datum zadání bakalářské práce: **15. ledna 2009**

Termín odevzdání bakalářské práce: **15. května 2009**

Univerzita Pardubice Česká republika

L.S.

doc. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan
vedoucí katedry

V Pardubicích dne 31. března 2009

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 15. 05. 2009

Jiří Zechmeister

Poděkování

Rád bych touto cestou poděkoval panu Ing. Zdeňku Šilarovi za ochotnou spolupráci při tvorbě této práce.

ANOTACE

Tato práce je zaměřena na vývoj webového redakčního systému pro správu obsahu webových stránek. Hovoří obecně o problematice redakčních systémů. Zaměřuje se na technologii PHP5, konkrétně na objektově orientované programování v tomto jazyku, v kombinaci s JavaScriptem a technologií AJAX. Dále se zaměřuje na databázové technologie MySQL a Oracle. Součástí práce je vlastní modulární redakční systém pro kompletní dynamickou správu webových stránek. V aplikaci je kladen důraz na snadné použití uživatelem.

KLÍČOVÁ SLOVA

php, javascript, ajax, crm, modulární, systém, mysql, oracle, redakční

TITLE

Content management systém for comprehensive management of dynamic web pages

ANNOTATION

This bachelor thesis focuses on developing web-oriented publishing system for managing websites content. It consists of basic theoretical publishing system study. Focuses on PHP5 technology, object oriented programming in combination with Javascript and AJAX technology to be more precise. It also consists of database technlogies, MySQL and Oracle.This thesis also includes practical part - modular content management system for complete dynamical website maintenance. This application insists on user friendliness

KEYWORDS

php, javascript, ajax, crm, modular, system, mysql, oracle

OBSAH

SEZNAM OBRÁZKŮ

1. Úvod

Cílem této bakalářské práce je vytvořit redakční systém pro kompletní dynamickou správu webových stránek. Celý redakční systém má být navržen jako přehledné, jednoduše spravovatelné rozhraní mezi databází, soubory uloženými na serveru a webovými stránkami. Toto bude umožňovat jednoduchou editaci veškerého obsahu internetových stránek a to hlavně pro uživatele, kteří nemají znalosti značkovacího jazyka HTML nebo jiného programovacího jazyka. Aplikace je určena jak pro zkušené administrátory, tak pro úplně začátečníky.

V kapitole vývoj internetových aplikací dojde k představení nepoužívanějších technologií a jejich rozdělení. Dále je kapitola zaměřena na jazyk PHP a jeho návaznosti na databázi MySQL a dále je zde dojde k seznámení s moderní technologií asynchronního JavaScriptu (AJAX). Kapitola redakční systémy, bude svým obsahem zaměřena zejména na typy redakčních systémů. Jejich představení a možná implementační řešení.

Kapitola analýza problému poodhalí přípravné práce na projektu. Představí se zde analýza celého systému. V kapitole implementace redakčního systému se představí nejdůležitější části implementace systému. Konkrétně se zaměří na jádro systému, vybudování rozhraní aplikace a předávání dat v rámci aplikace. Kapitola databáze představí způsoby připojení k databázovým technologiím. Kapitola implementace do internetových stránek je zaměřena na principy připojení redakčního systému do internetových stránek. V kapitole popis práce se systémem je zaměřena na popis jednotlivých modulů systému a jejich funkčnímu vybavení.

2. VÝVOJ INTERNETOVÝCH APLIKACÍ

2.1. STATICKÉ PUBLIKOVÁNÍ A AKTIVNÍ WEBOVÉ STRÁNKY

První generací tvorby webových aplikací bylo statické publikování. Takové aplikace se spoléhaly pouze na jazyk HTML. Ten umožňoval pouze zobrazení statických textů a obrázků, které ne šli umístit na konkrétní souřadnice. Každá úprava takových stránek vyžadovala buď zásah přím do HTML, nebo užití nějakého editoru. Statické stránky také nepodporovali připojení k databázi [1].

Aktivní webové stránky již umožňovaly uživateli odesílání přizpůsobených stránek a umožňovali dynamické prohlížení. Existují jako kombinace jazyků a technologií. Technologie můžeme rozdělit do dvou kategorií: technologie, které se aplikují na straně klienta a technologie, které se aplikují na straně serveru[1].

2.1.1. TECHNOLOGIE NA STRANĚ KLIENTA

- Ovládací prvky ActiveX – vytvořené v jazyce Visual C++ nebo Visual Basic
- JavaApplety
- Skriptování na straně klienta a dynamické HTML

2.1.2. TECHNOLOGIE NA STRANĚ SERVERU

- CGI
- Vnitřní rozhraní API webových serverů, jako ISAPI a NSAPI
- ASP (Active Server Peges)
- Java Server Pages a Servlety
- Server-Side Javascript
- PHP

2.2. TECHNOLOGIE POUŽITÉ PRO VÝVOJ PROJEKTU

V následujícím textu se zaměříme pouze na technologie, které byly využity při implementaci projektu. Zejména se zaměříme na programovací jazyk PHP, databáze MySQL a technologii AJAX.

2.3. PHP

PHP (Hypertextový preprocesor) je skriptovací jazyk využívající se pro vývoj dynamických webových stránek. Většinou se začleňuje přímo do HTML kódu. PHP lze také využít pro programování konzolových a desktopových aplikací. My se ovšem zaměříme na tvorbu dynamických webových stránek.

Jazyk PHP je prováděn na serveru to znamená, že se jedná o skriptování na straně serveru. V dělení programovacích jazyků se řadí mezi jazyky interpretované. Aplikace je až do jejího spuštění uchovávána v textové formě. Teprve v okamžiku, kdy si uživatel vyžádá zobrazení stránky, je kód PHP předám interpretu jazyka PHP, který kód zkompiluje a provede. Uživateli se vrací pouze vygenerovaný HTML kód. Uživatel tedy může zasáhnout do provádění kódu pouze pomocí parametrů předaných skriptu. PHP je nezávislý na platformě to znamená, že skripty fungují bez větších úprav na většině operačních systémů. Jeho syntaxe vychází z jazyků Perl, C, Pascal a Java.

Jazyk PHP je dynamicky typový to znamená, že datový typ proměnné je určen až v okamžiku, kdy je do ní přiřazena hodnota. Z tohoto důvodu je v jazyku implementován jak operátor ==, který provádí konverzi operandů na jeden datový typ a porovnává pouze hodnoty, tak operátor ===, který je vyhodnocen jako pravdivý teprve tehdy, pokud jsou oba operandy také stejného datového typu[2].

Pole v jazyce PHP jsou heterogenní, což znamená, že je možné do pole uložit různé datové typy. Indexování polí je možné jak pomocí číselných indexů tak mohou fungovat jako hash-mapa, je tedy možné jako index využít řetězec. Oba způsoby indexování je možné kombinovat. Řetězce je možné uzavírat jak do uvozovek, tak do apostrofů. Při použití uvozovek se provede nahrazení proměnných uvnitř řetězce,

kdežto při použití apostrofů, jsou nahrazeny pouze escape sekvence. Jazyk PHP je v kombinaci s databází MySQL a jazykem HTML velice oblíbený pro tvorbu webových stránek. V této kombinaci vytvořena spousta internetových projektů. Aktuální verze jazyka PHP v době psaní práce je 5.2.8

2.4.MySQL

MySQL je databáze nezávislá na platformě. Ke komunikaci s databázovým strojem je užit jazyk SQL. Pro svoji snadnou implementaci, výkon a volnou šiřitelnost je velice oblíbený. Nejčastější využití tohoto databázového systému je v kombinaci s jazykem PHP.

2.5.AJAX

Název AJAX je zkratka slov Asynchronous JavaScript a XML. Jedná se o obecné označené technologie pro vývoj interaktivních webových aplikací. Umožňují vytváření uživatelsky přívětivějších internetových aplikací[6].

AJAX dále využívá další technologie

- HTML (DHTML), CSS pro prezentaci výstupu
- DOM a JavaScript pro zobrazování a dynamické změny v prezentaci
- xmlHttpRequest pro asynchronní výměnu dat s webovým serverem

Princip práce s technologií AJAX spočívá v odesílání požadavků na server bez nutnosti nového načítání stránky. Požadavek je odeslán pomocí rozhraní *xmlHttpRequest*. Toto rozhraní umožňuje komunikaci klienta se serverem na základě protokolu http. Navrácené informace mohou být k dispozici jako XML dokument. Je ovšem také možnost, požadovat jako formát navrácených dat například čistý text nebo jiné formáty. Navrácená data mohou být zpracována například pomocí DOM což je objektový model pro zpracování XML a HTML dokumentů a JavaScriptem zavedena na výstup k uživateli.

3. WEBOVÉ REDAKČNÍ SYSTÉMY

Redakční systém nebo také systém pro správu obsahu CMS (z anglického content management system) je aplikace zajišťující správu obsahu nejčastěji internetových stránek. Většinou se jedná o internetovou aplikaci, ale může být rozšířena i o desktopové aplikace umístěné u klienta[3].

Pro vývoj redakčních systémů se využívá různých technologií. Nejjednodušší redakční systémy mohou být vytvořeny pouze v JavaScriptu. Většina je však programována v jazyce PHP s podporou databází například MySQL.

Mezi základní funkce CMS patří:

- Správa obsahu webové prezentace. Většinou řešené vestavěným WYSIWYG editorem a umožňuje snadnou editaci obsahu bez znalosti HTML.
- Správa přístupu k dokumentům spojený se správou uživatelů a uživatelských skupin
- Správa diskusí a komentářů
- Správa souborů
- Správa obrázků a galerií
- Kalendářní funkce
- Statistiky

3.1. TYPY WEBOVÝCH REDAKČNÍCH SYSTÉMŮ

Existují tři hlavní skupiny webových redakčních systémů

3.1.1. OFFLINE REDAKČNÍ SYSTÉMY

Offline redakční systémy připraví kompletně všechen obsah stránek před samotnou publikací stránek na internetu. Tento typ redakčního systému nevyžaduje server pro aplikování změn. Někdy se jedná pouze o nástroj pro návrh stránek.

3.1.2. ONLINE REDAKČNÍ SYSTÉMY

Online redakční systém upravuje obsah webových stránek v reálném čase. Samotný výstup stránek je generován až při požadavku uživatele na načtení stránky.

3.1.3. HYBRIDNÍ REDAKČNÍ SYSTÉMY

Hybridní systémy kombinují online a offline způsoby. Většinou nejsou umístěny přímo na serveru, ale slouží k nahrávání dokumentů, které slouží k dynamickému zobrazování, jako jsou dokumenty PHP, ASP a další.

4. ANALÝZA NAVRHOVANÉHO SYSTÉMU

Stále více zájemců o vytvoření internetových stránek, ať již jednoduché prezentace nebo složitějších aplikací, chce spravovat obsah samostatně. Považují za zbytečné, kvůli náhlé změně telefonního čísla kontaktovat vývojáře, který ještě zrovna nemusí být k dispozici. Na druhou stranu, i když člověk vyloženě nechce své stránky spravovat sám, je pro vývojáře jednodušší implementovat redakční systém a pozdější změny podle přání zákazníků provádět právě přes rozhraní, než zasahovat do kódu. Zásah do kódu aplikace může totiž v některých případech zanést nechtěné chyby. Je tedy nutné zhotovit natolik jednoduchý a přehledný systém, aby s ním byly schopni pracovat i běžní uživatelé s obvyklou praxí z desktopových kancelářských aplikací. Dále systém musí být dost robustní na to, aby pokryl veškeré nároky uživatele.

Redakční systém by tedy měl mít příjemné uživatelské rozhraní. Pokud možno podobné běžným desktopovým aplikacím, aby se uživatel nemusel stále a stále učit nové věci. Dále by měl být jednoduchý na ovládání a jednoznačný. Určitě také modulární, což pokryje nároky a robustnost a flexibilitu.

Systém bude tvořen jako modulární systém. Toto znamená, že veškeré součásti je možné do systému postupně přidávat a také je odebírat podle požadavků. Toto umožní maximální možnou flexibilitu. Vzhled aplikace bude uzpůsoben tak, aby co nejvíce připomínal klasickou okenní aplikaci typu win32 a stal se tak uživatelsky co nejpřívětivější. Základem aplikace bude jádro naprogramované v jazyce PHP. Rozšíření interakce s uživatelem bude řešeno pomocí JavaScriptu. Pro další rozšíření možností uživatelského rozhraní je možnost použití knihovny script.aculo.us a JavaScriptového frameworku Prototype.

5. IMPLEMENTACE REDAKČNÍHO SYSTÉMU

5.1. JÁDRO SYSTÉMU

Jádro systému se skládá ze dvou hlavních souborů. První hlavní soubor jádra je soubor *Core.php*. Je programován v jazyku PHP a obsahuje hlavní třídy, z jejichž instancí je tvořené celé prostředí redakčního systému. Druhým hlavním souborem je soubor *Core.js*. V tomto souboru jsou implementovány funkce v jazyku JavaScript pro zlepšení vlastností prostředí. Jsou zde také implementovány hlavní funkce technologie AJAX. Dalším důležitým souborem jádra systému je soubor *function.php*, ve kterém jsou umístěny funkce a procedury, které nejsou součástí tříd. Soubory *Core.php* a *function.php* je umístěn v adresáři *included*. Soubor *Core.js* je umístěn v kořenovém adresáři systému

5.2. DATABÁZOVÉ PŘIPOJENÍ

Redakční systém umožňuje práci jak nad databází MySQL tak Oracle. Implementace připojení k jednotlivým databázím je tvořena pomocí tříd ovládajících požadovanou databázi.

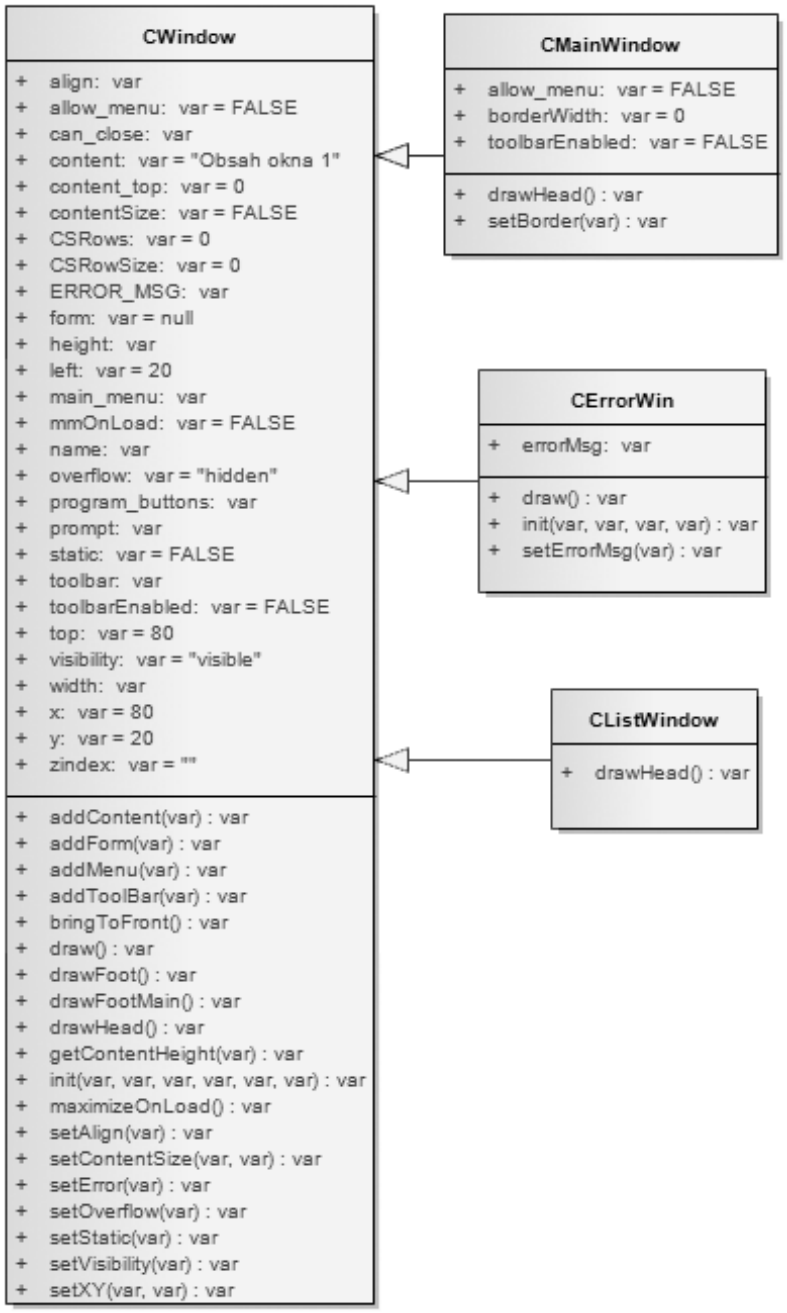
CMySQLDriver (viz obrázek 6) je třída řídící přístup do databáze MySQL. Ve třídě je implementována metoda *execSQL*, která slouží k provedení daného SQL kódu. Parametry do dotazů jsou předávány pomocí instancí třídy *CParamList* a *CParametr*. Ve třídě *CParametr* také zabezpečuje ochranu proti útokům typu sql injection.

COraDriver (viz obrázek 6) je třída, která má na starosti ovládání komunikace se serverem Oracle. Ve třídě je stejně jako ve třídě *CMySQLDriver* implementována metoda *execSQL* a předávání parametrů je řešeno pomocí tříd *CParamList* a *CParametr*.

Rozhodnutí jaká třída ovladače bude použita, závisí na funkci *initDtbDriver*, která na základě konfiguračních souborů vrací instanci patřičné třídy ovladače.

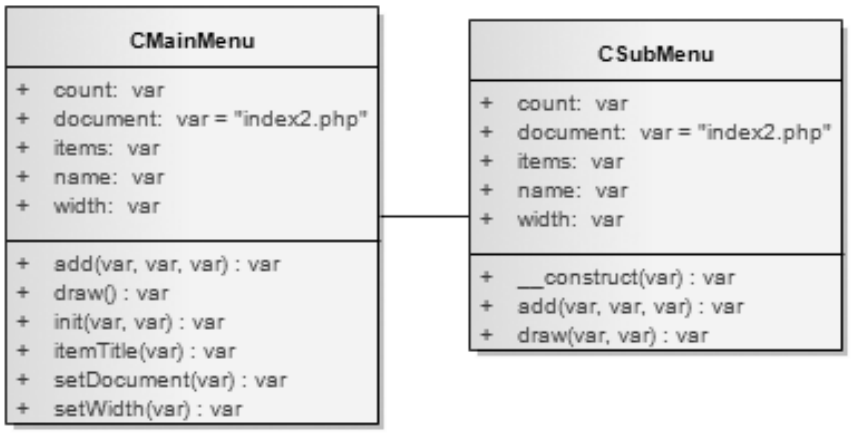
5.3. PROSTŘEDÍ

Základní prostředí je tvořeno instancemi třídy *CWindow* (viz obrázek 1) a potomky této třídy. Uvnitř třídy jsou definovány metody pro zobrazení hlaviček oken a vytvořeny handlery JavaScriptových funkcí pro manipulaci s okny. Třída *CWindow* je nejdůležitější součástí celého systému.

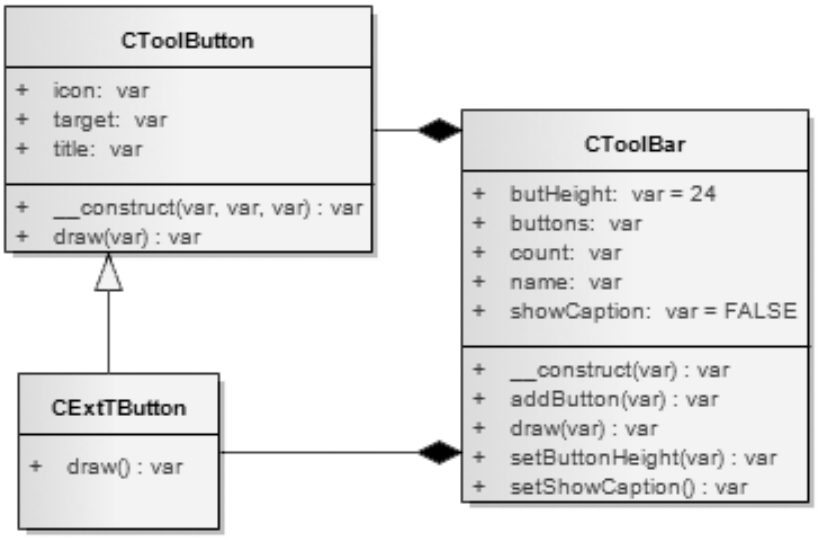


Obrázek 1 - UML Class Diagram - Interface aplikace

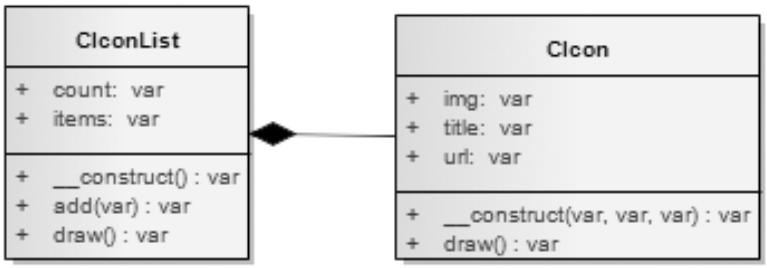
O další části prostředí se starají třídy *CMainMenu*, *CSubMenu* (viz obrázek 2) pro vytvoření hlavního menu aplikace. Třídy *CToolBar*, *CToolButton* a *CExtTButton* (viz obrázek3) které mají na starost vytvoření a zobrazení hlavní nástrojové lišty aplikace. Třídy *CIconList* a *CIcon* (viz obrázek 4) zajišťující vytvoření a zobrazení plochy se seznamem ikon modulů.



Obrázek 2 - UML Class Diagram - Hlavní menu



Obrázek 3 - UML Class Diagram – Panel nástrojů



Obrázek 4 - UML Class Diagram - Plocha ikon

5.3.1. POSTUP BUDOVÁNÍ PROSTŘEDÍ APLIKACE

Po úspěšné validaci přihlašovacího formuláře se uloží informace o uživateli a o prostředí, ve kterém pracuje. Uchovávány jsou informace o IP adrese počítače, ze kterého je do systému přistupováno, operačním systému a webovém prohlížeči. Dále také o maximální velikosti zobrazitelného obsahu v okně webového prohlížeče. Tyto hodnoty jsou předány pomocí SESSION proměnných souboru *index2.php*, který má na starost vykreslení interface aplikace a řízení zobrazování modulů.

Do soboru *index2.php* je nejprve přilinkován soubor *head.php*, který v sobě obsahuje hlavičky dokumentu HTML a přilinkování dalších souborů jako *Core.php*, *Core.js*, *function.php*, *session.php* a konfiguračních souborů. Dále je zde také implementován mechanizmus na odhlášení uživatele a validaci URL požadavku.

Jako další krok jsou vytvořeny instance třídy, ze kterých bude interface tvořen. Jako hlavní je vytvořena instance třídy *CMainWindow*, která jako potomek třídy *CWindow* zobrazuje hlavní okno aplikace. Dále jsou vytvořeny instance pro vytvoření hlavního menu aplikace, panelu nástrojů a seznamu ikon. Dále je volána metoda třídy *CMainWindow* nad její instancí, která nastaví parametry okna. Jako parametr velikost, je předán obsah proměnných s maximální možnou velikostí obsahu webového prohlížeče. Toto zajistí, že na různých rozlišeních je velikost aplikace vždy přizpůsobena velikosti okna.

Následující důležitá funkcionalita je připojení modulů do aplikace. Více v kapitole 5.6. Tato funkcionalita naplní instanci menu, nástrojové lišty a seznamu ikon daty.

Do instance hlavního okna aplikace jsou připojeny instance menu a nástrojové lišty a následně je vyvolána nad instancí hlavního okna metoda *drawHead*, která vykreslí záhlavní aplikačního okna a zobrazí hlavní menu a nástrojové lišty.

Následující prostor je obsah okna aplikace. V této části budou později zobrazeny zvolené moduly a veškerý obsah. Je zde tedy implementována funkcionalita pro zobrazení aktuální zvoleného modulu. Dále je zde také vykreslen seznam ikon, které se zobrazí na ploše aplikace a také logo aplikace. Logo i seznam ikon jsou překrývány později jednotlivými moduly.

Poslední částí souboru *index2.php* je volání metody *drawFoot* nad instancí hlavního okna aplikace. Tato metoda zajistí uzavření vytvořených HTML kontejnerů a provede případné zarovnání okna aplikace pomocí funkcí JavaScriptu. Metoda také obstarává případné zobrazení chybových hlášení z celého systému. Úplně na konci je připojen soubor *foot.php*, který zakončuje otevřené entity HTML jako je <*body*> a <*html*>.

5.4. PŘEDÁVÁNÍ INFORMACÍ UVNITŘ SYSTÉMU

K předávání informací uvnitř systému jsou použity požadavky protokolu http. Tento protokol podporuje sedm metod předávání požadavků. Tyto metody jsou: GET, POST, HEAD, OPTIONS, PUT, DELETE, TRACE. Aplikace redakčního systému využívá pro přenos informací mezi skripty metody POST a GET. Dále jsou také použity cookies a session proměnné.

5.4.1. METODA GET

Jedná se o nejjednodušší metodu a patří mezi základní metody. Téměř pokaždé, pokud nebyl odeslán formulář metodou POST, je webová stránka odeslána metodou GET. Výsledkem je stránka a její hlavičky, na kterou se pomocí metody ptáme. Zjednodušeně řečeno, metoda GET odesílá veškeré informace v URL požadované stránky[4]. Uvnitř systému se metoda GET používána pro směrování jednotlivých skriptů. Jelikož je ale metoda GET velice snadno napadnutelná, v systému probíhá šifrování GET požadavku. Více v kapitole 5.7.1 Šifrování GET požadavků.

Struktura hlavičky metody GET

GET URL-STRÁNKY VERZE-PROTOKOLU
HLAVIČKY
prázdný řádek

Příklad:
GET /index.asp HTTP/1.1
Host: www.interval.cz
prázdný řádek

5.4.2. METODA POST

Metoda POST funguje podobně jako metoda GET, navíc však umožňuje za hlavičkami odesílat skriptu data. Tato data jsou tak na první pohled neviditelná. Data však lze získat, pokud si prohlédneme obsah hlavičky protokolu http. Uvnitř systému se metoda POST využívá pro odesílání informací z formulářů. Většinou jsou obě metody kombinovány. Data v hlavičce odeslané metodou POST, jsou směrované pomocí GET požadavku.

Struktura hlavičky metody POST

POST URL-STRÁNKY VERZE-PROTOKOLU
HLAVIČKY
prázdný řádek
DATA Z FORMULÁŘE

Příklad:
POST /zpracujdata.php HTTP/1.1
Host: www.formulare.cz
Content-Length: 29
Content-Type: application/x-www-form-urlencoded
prázdný řádek
pole1=hodnota1&pole2=hodnota2

5.4.3. COOKIE

Termínem cookie se v protokolu http označuje malé množství dat, které je možno uložit na počítači uživatel. Při každé další návštěvě, jsou data odeslána zpět serveru[5].

V jazyce PHP jsou cookie k dispozici přes asociativní poke *$_COOKIE[]*. Jako index do pole slouží název požadované cookie. Pro použití cookie v JavaScriptu jsou implementovány funkce *setCookie* a *getCookie*. V systému jsou cookie využívány k uchovávání dat při použití technologie AJAX.

5.4.4. SESSION PROMĚNNÉ

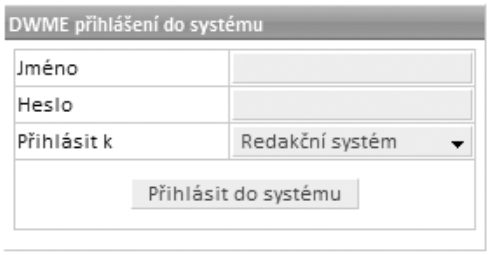
Session proměnné jsou speciální proměnné, které umí udržet svoji hodnotu mezi jednotlivými http požadavky. Hodnota proměnných se po dokončení provádění skriptu ukládá na pevný disk serveru. Při opětovném požadavku od stejného uživatele, je hodnota opět k dispozici. Identifikace uživatele je realizována pomocí speciální cookie s názvem *PHPSESSID*, která obsahuje id aktuálního uživatele. Jejich využití je podobné jako u cookie.

Před použitím session proměnné je nutné ji zaregistrovat pomocí funkce *session_register()*. Registrovaná proměnná je potom přístupná v asociativním poli *$_SESSION[]*, kde jako index je použit název proměnné. V systému jsou session proměnné využité pro nastavení prostředí a také je v session proměnné uložena instance hlavního okna aplikace, aby bylo možné k ní přistupovat uvnitř všech funkcí.

5.5. Řízení chodu aplikace

5.5.1. Přihlášení

Přihlášení do aplikace je realizováno přihlašovacím dialogem v souboru *index.php*. Po odeslání přihlašovacího formuláře jsou v systému udržovány informace o identifikaci uživatel, IP adrese, ze které uživatel přistupuje, uživatelské jméno a heslo v podobě md5 otisku pro zajištění bezpečnosti.



Obrázek 5 - Okno pro zadání přihlašovacích informací

5.5.2. Ověření uživatelů

Ověření uživatele je prováděno na základě uživatelského jména, hesla a IP adresy počítače, ze kterého uživatel přistupuje. Ověřování správnosti údajů následně probíhá při každém načtení obsahu stránky. Nejprve je kontrolován otisk hesla s otiskem uvedeným v databázi pro daného uživatele, poté je kontrolována IP adresa počítače. Pokud nesouhlasí heslo uživatele, nebo IP adresa nastala možnost, že se jedná o nepovolený přístup s falešnými údaji. V takovém případě je uživatel odhlášen a přesměrována na formulář s přihlášením.

5.5.3. Přihlášení do módu nastavení

Mód nastavení je speciální část aplikace implementovaná v souboru *index3.php*, která slouží k nastavení přístupů do databáze včetně volby, na které databázi bude aplikace fungovat. Při přihlašování do módu nastavení je ověření realizováno přes datový soubor, ve kterém je obsaženo přednastavené heslo administrátora.

Pro přístup do módu nastavení je nutné pouze heslo administrátora. Heslo v datovém souboru je uchováno v podobě md5 otisku. Při každé změně hesla uživatele *admin*, je heslo zapsáno také do datového souboru.

5.5.4. PŘIHLÁŠENÍ DO PROSTŘEDÍ REDAKČNÍHO SYSTÉMU

Samotné prostředí je implementováno v souboru *index2.php*. Ověřování přístupu do části redakčního systému je realizováno ve spojení s databází a vyžaduje správnou kombinaci uživatelského jména a hesla.

5.6. MODULY

Moduly jsou umístěny v adresáři *modules*. Každý modul je umístěn ve vlastním adresáři, kde jsou dva soubory povinné. Rozsah modulu, počet podadresářů a souborů je závislý pouze na tvůrci modulu.

Každý modul se skládá z několika částí. Hlavní součást je zavádějící soubor modulu. Dalším velice důležitým souborem, bez kterého by nemohlo dojít k jeho zavedení do systému, je soubor *config.inc*. Přesný název souboru je nutný pro všechny moduly. V tomto souboru jsou udány informace o názvu modulu, způsobu jeho zavedení a souboru, který bude zaveden jako spouštěcí. Dále obsahuje informace o podmodulech a přednastaveném oprávnění pro modul.

5.6.1. ZAVEDENÍ MODULŮ

Pro zavedení modulu je načten konfigurační soubor modulu, *config.inc*. tento soubor zavede do systému informace o modulu a o způsobu jeho zavádění.

Zavádění probíhá cyklickým průchodem adresáře *modules* a postupném načítání konfiguračních souborů jednotlivých modulů. Každý konfigurační soubor přidá záznam do globálního pole *$TARGETS*, pomocí kterého se provádí směrování v rámci aplikace. Také jsou z každého konfiguračního souboru načteny informace o jeho názvu a o ikoně. Tyto informace jsou použity pro zobrazení modulů ve hlavním menu a na panelu nástrojů.

5.7. ZABEZPEČENÍ

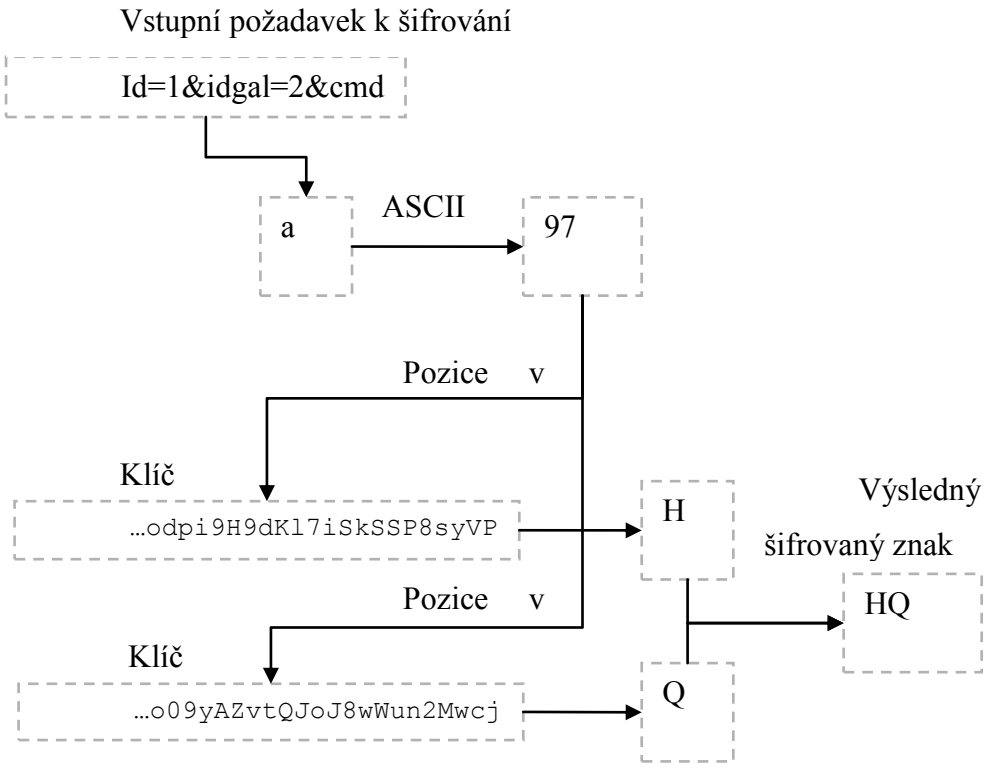
Zabezpečení aplikace probíhá na více úrovních. Na první úrovni je volitelné použití šifrovaného přenosu. Jedná se tedy o použití vrstvy SSL nad protokolem HTTP. V této verzi aplikace ještě není využití šifrovaného spojení vynucené. Další vrstvou zabezpečení je přihlášení uživatele a jeho ověření. O této problematice je psáno v kapitole 5.5.1. Poslední vrstvou zabezpečení je šifrování GET požadavku pro zamezení odpozorování směrování aplikace.

5.7.1. ŠIFROVÁNÍ GET POŽADAVKŮ

Základ pro algoritmus šifrování je vygenerování páru klíčů. Klíče jsou generovány při každém přihlášení uživatele a jsou uchovávány po celou dobu relace. Oba klíče mají délku 500B.

Z každého požadavku je speciální funkcí *getEncrypt()* s použitím párů klíčů vytvořen šifrovaný požadavek ve tvaru *index2.php?getid=*. Tato metoda zamezuje útočníkovi odpozorování směrování a teda možnost přímého přechodu na požadovaný segment systému.

5.7.2. SCHÉMA ŠIFROVACÍHO ALGORITMU GET POŽADAVKU



Obrázek 6 - Schéma šifrovacího algoritmu GET požadavku

6. DATABÁZE REDAKČNÍHO SYSTÉMU

Redakční systém podporuje práci jak pod MySQL tak pod Oracle databázovým serverem.

6.1. POPIS DATABÁZOVÝCH OBJEKTŮ

6.1.1. DATABÁZE MYSQL

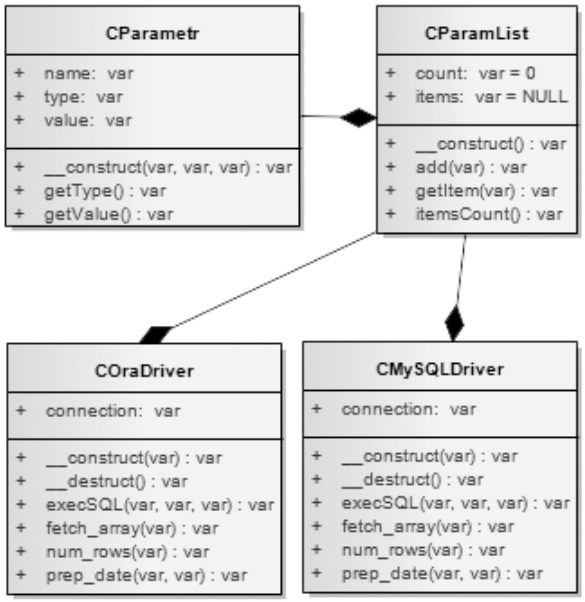
Pro použití systému nad databází MySQL, jsou vytvořeny patřičné tabulky ke každému modulu a nad každou tabulkou je vytvořen primární klíč. Žádné další databázové objekty pro chod aplikace nad touto databází nejsou třeba.

6.1.2. DATABÁZE ORACLE

Pro použití systému nad databází Oracle, jsou vytvořeny také patřičné tabulky pro každý modul podobně jako v případě pro databázi MySQL. Pro chod aplikace nad touto databází je však nutné použít ještě další databázové objekty. Aby byla zajištěna unikátnost primárních klíčů, je nad každou tabulkou vytvořena sekvence, která generuje řadu identifikátorů. Aby bylo možné sekvence použít, jsou také implementovány nad tabulkami triggery, které před každým vložením záznamu, přidají další unikátní identifikátor ze sekvence.

6.2. ZPŮSOB PŘÍSTUPU K DATABÁZI

Přístup k databázi je realizován pomocí tříd. Redakční systém umožňuje připojení k databázovému serveru MySQL a serveru Oracle. Pro každý z těchto databázových technologií je vytvořena jedna obslužná třída. Obsluhu databáze MySQL obstarává třída *CMySQLDriver*. Obsluhu databáze Oracle zajišťuje třída *COraDriver*. UML Class diagram třídy pro práci s databází, je k dispozici na obrázku 7.



Obrázek 7 - UML Class Diagram - Přístup do databáze

Konfigurace pro server MySQL je uložena v souboru *mysql_config.php*. Pro server Oracle je konfigurace místěna v souboru *ora_config.php*. Oba datové soubory jsou uloženy v adresáři *configs*.

6.2.1. NAVÁZÁNÍ A ZRUŠENÍ PŘIPOJENÍ MYSQL

Připojení vzniká vytvořením instance třídy *CMySQLDriver*. Připojení setrvává po celou dobu trvání generování výstupu stránky. Připojení je uzavřeno v destruktoru třídy. Jelikož je destruktor zavolán vždy po dokončení načítání stránky automaticky, není možné, že by připojení zůstalo neuzavřené. Připojení k databázi je realizováno následující PHP funkcí. Jednotlivé proměnné, které vstupují jako parametry, jsou naplněny z konfiguračního souboru podle nastavení uživatele.

```
$this->connection =
@mysql_connect($mysql_server,$mysql_user,$mysql_password,$mysql_database_name);
```

6.2.2. NAVÁZÁNÍ A ZRUŠENÍ PŘIPOJENÍ ORACLE

Připojení vzniká při vytvoření instance třídy *COraDriver*. Sestavené připojení je potom udržováno po celou dobu generování stránky. Připojení je zrušeno v destruktoru třídy *COraDriver*. Následuje PHP funkce, které provádí připojení k serveru Oracle. Proměnné, které do funkce vstupují jako parametry, jsou naplněny hodnoty podle konfiguračního souboru.

```
$this->connection = @oci_connect
($ora_user,$ora_password,$ora_server,$ora_charset);
```

6.2.3. SESTAVENÍ A PROVEDENÍ DOTAZU SQL

Sestavování a provádění příkazů SQL má na starost metoda *execSQL*. Tato metoda je obsažena v obou třídách. Parametry zvenčí vstupují do příkazu jako instance třídy *CParamList*. Tato třída slouží jako seznam parametrů. Každý parametr je tvořen samostatnou instancí třídy *CParametr*. Tato třída zajišťuje kontrolu vstupních parametrů. U parametrů je kontrolován datový typ, konkrétně pokud se jedná o číslo, datum nebo text. Pokud se jedná o datum, je kontrolován formát zápisu data. U řetězce je kontrolován možný pokus o útok typu sql injection.

Ochrana proti útoku typu SQL Injection

Instance třídy *CParametr* dostanete v konstruktoru požadovanou hodnotu spolu s očekávaným datovým typem hodnoty. Pokud požadovaná číselná hodnota není číslo, je vyvolán chybový dialog a uživatel je okamžitě odhlášen z aplikace. U řetězcových parametrů, není nutná speciální kontrola, protože v celé aplikaci řetězce okamžitě převáděny funkcí *urlEncode* do tvaru, ve kterém nejsou žádné nebezpečné znaky. Možnost podvržení dalšího příkazu SQL je ošetřeno již na úrovni funkcí jazyka PHP. Příkazy *mysql_query* a *ociexecute* nepovolují provedení více jak jednoho příkazu.

Provedení dotazu

Metodě *execSQL* jsou předány dva parametry, kde první představuje samotný SQL příkaz k provedení a druhý instanci třídy *CParamList*, která v sobě obsahuje jednotlivé parametry. Tento parametr však není povinný, pokud nepotřebujeme vkládat parametry do SQL dotazu. Uvnitř metody se dosadí parametry do řetězce příkazu a příkaz se provede. Návratovou hodnotou této metody je FALSE v případě chyby, nebo výsledek dotazu v případě úspěchu.

7. IMPLEMENTACE REDAKČNÍHO SYSTÉMU DO WWW STRÁNEK

7.1. UMÍSTĚNÍ REDAKČNÍHO SYSTÉMU

Pro správnou funkčnost redakčního systému je nutné, aby soubory systému byly nahrány v adresáři *DWME* v kořenovém adresáři internetových stránek. Do adresáře *DWME* je nutné povolit na serveru zápis skriptů. Pokud by se tak nestalo, nebylo by možné aplikaci nastavit.

Dále je nutné vytvořit adresář *data* v kořenovém adresáři aplikace. Tento adresář slouží k uchovávání uživatelských souborů. Z důvodu ochrany dat, je redakčnímu systému povoleno pracovat pouze s tímto adresářem. Také pro tento adresář je nutné na serveru povolit skriptu zápis.

7.2. IMPLEMENTACE

Implementace je realizována tak, aby byla co nejjednodušší i pro méně znalé vývojáře internetových stránek. Základem je soubor *Output.php*, který je nutné připojit do stránek, například funkcí *include*. Tento soubor obsahuje definice tříd, které se starají o zobrazení obsahů z jednotlivých modulů systému. Všechny výstupní třídy jsou potomky třídy *COutput*, ve které jsou definovány základní abstraktní metody a atributy.

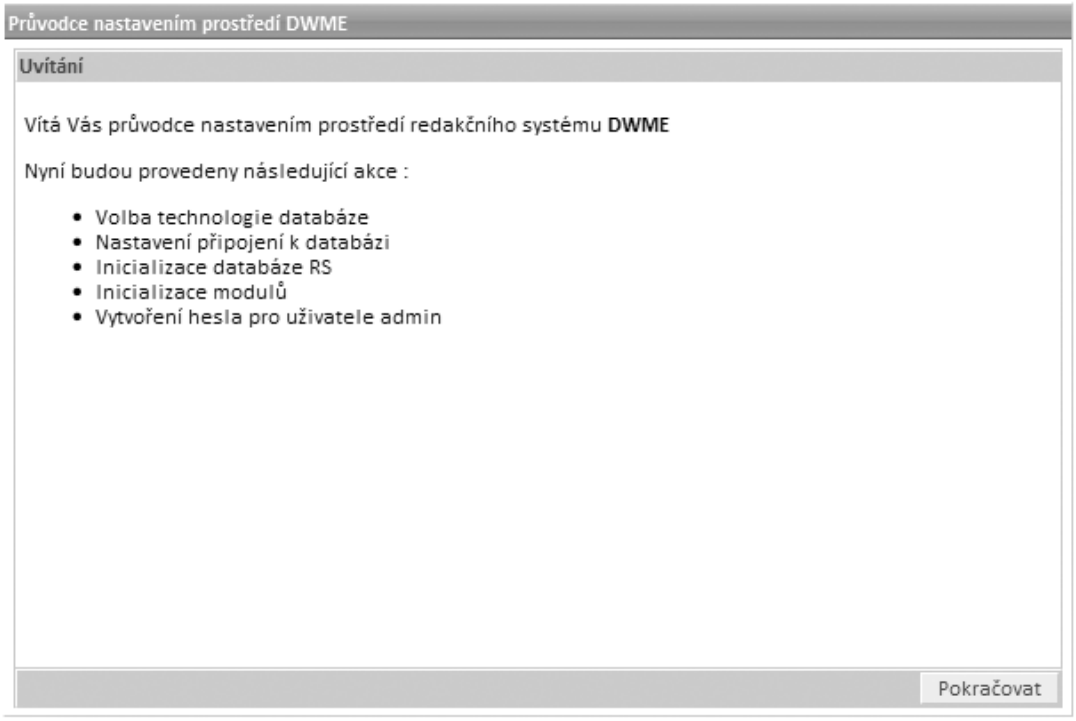
Pro zobrazení konkrétního obsahu již stačí pouze vytvořit instanci obslužné třídy a zavolat metodu *draw* s požadovanými parametry, která se postará o samotné vykreslení.

7.3. PRVNÍ SPUŠTĚNÍ

Pokud byl systém nakopírován do správného adresáře, a tomu byly přiděleny patřičná oprávnění, je možné provést nastavení systému. K nastavení systému přejdeme z hlavního přihlašovacího formuláře tak, že v položce „přihlásit k“ zvolíme

hodnotu „Nastavení“. Pro vstup do této sekce je vyžadováno pouze heslo administrátora. Toto heslo je pro první spuštění přednastaveno na hodnotu „1234554321“.

V nastavení se nejprve provede výběr databáze, nad kterou systém bude pracovat a nastaví se přístupové informace k databázovému serveru. V dalším kroku se systém pokusí připojit k databázi. Pokud ji nenajde, pokusí se ji vytvořit. Pokud se nezdaří ani vytvoření, systém vypíše chybové hlášení. Bez správného nastavení databáze systém nebude pracovat. Pokud se systému podařilo přihlásit k databázovému serveru a podařilo se také vytvořit nebo přihlásit k databázi, spustí se vytváření jednotlivých tabulek pro moduly instalované v systému. Pokud proběhne vše bez problémů, je ještě uživatel vyzván k vytvoření nového hesla pro uživatele *admin*. Nyní je systém připraven k používání a je možné přihlásit se do prostředí redakčního systému novým heslem.



Obrázek 8 - Průvodce nastavením redakčního systému

8. POPIS PRÁCE SE SYSTÉMEM

Obrázek 9 - Pracovní plocha redakčního systému

8.1. UŽIVATELSKÉ SKUPINY

Architektura systému umožňuje dělení uživatelů do tří uživatelských skupin. Přiřazení uživatele do skupiny, oprávnění se provádí při vytváření uživatelského účtu pomocí modulu Správa uživatelů.

Každý uživatel má podle uživatelské role přednastavené oprávnění pro každý z modulů integrovaných do systému. Rozšíření oprávnění pro uživatele je možné přes editaci uživatele, kde administrátor může přiřadit každému uživateli rozsah jeho oprávnění.

Hlavním uživatelským účtem je účet *admin*. Jedná se o správce systému, který je přednastaven již od fáze nastavování systému. Tento administrační účet jako jediný není možné ze systému odstranit. Učet disponuje maximálním stupněm oprávnění.

8.1.1. POPIS UŽIVATELSKÝCH SKUPIN

Administrator

Skupina správců. Členové této skupiny mají přednastavené oprávnění ke všem položkám oprávnění jednotlivých modulů.

PowerUser

Skupina uživatelů s vyšším oprávněním. Členové této skupiny mají přístup ke všem hlavním položkám oprávnění jednotlivých modulů s výjimkou Správy uživatelských účtů a Správy souborů.

User

Základní skupina oprávnění vhodná pro většinu uživatelů. Uživatelé v této skupině mají přístup pouze k základním položkám oprávnění jednotlivých modulů. Pro většinu běžných uživatelů se doporučuje využívat právě tohoto typu účtu.

8.2. POPIS MODULŮ SYSTÉMU

Jak již bylo řečeno, DWME je modulární systém. Je tedy možné do něho připojit libovolné množství různých modulů. V této kapitole si projdeme základní moduly systému.

8.2.1. MODULY JÁDRA

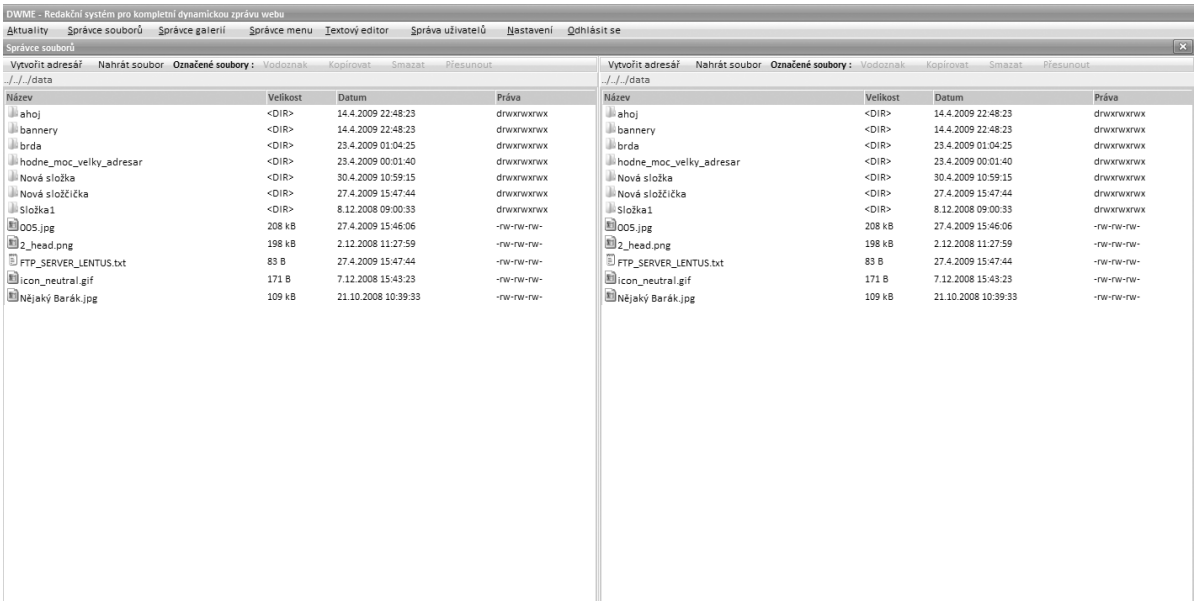
Jsou moduly přímo integrované do systému. Tyto moduly není možné ze systému odstranit, aniž by nebyla ovlivněna funkčnost. Umístění modulů je v adresáři *global* a jedná se o moduly, které slouží k nastavování uživatelských účtů a jednoduchou správu systému. Konkrétně se jedná se o moduly Správa uživatelů a Nastavení.

8.2.2. SPRÁVA SOUBORŮ

Modul slouží k nahrávání souborů na server a jejich správu. Pracovní plocha modulu je rozdělena na dva rámy. V každém rámu jsou zobrazeny soubory a adresáře z aktuálního adresáře nezávisle na sobě. Toto rozvržení umožňuje snadné kopírování a přesouvání mezi adresáři. Celé rozhraní je postaveno na technologii AJAX. To znamená, že požadavky na jednotlivé operace se soubory jsou prováděny bez nutnosti načítání stránky. Práce je tedy rychlá a efektivní. Uživatel také může pracovat s více soubory najednou. Jedná se hlavně o operace kopírovat, odstranit a přesunout. Dále modul umožňuje jednoduchou úpravu velikosti obrázků.

Přednastavené oprávnění pro tento modul je nastaveno tak, že pracovat se soubory mohou pouze uživatelé ve skupině administrator.

Soubory modulu jsou umístěny v adresáři *modules/FileCommander*.



Obrázek 10 - Prostředí modulu správce souborů

8.2.3. TEXTOVÝ EDITOR

Textový editor je navržený pro snadné vytváření a úpravu textových obsahů internetových stránek. Pro editaci textů je zde implementován WYSIWYG editor openWYSIWYG verze 1.47. Jedná se o rychlý a výkonný editor, který umožňuje uživateli vytvořit textový obsah dle vlastních představ. Editor umožňuje běžné funkce jako formát textu, vkládání tabulek a odkazů, zarovnání, odrážkování, vkládání číslovaných i nečíslovaných seznamů a obrázků. Vkládání obrázků a fotografií bylo explicitně doprogramováno z důvodu nutnosti napojení na stávající systémovou adresářovou strukturu.

Přednastavené oprávnění pro tento modul je nastaveno tak, že editaci a vytváření mohou provádět členové skupiny Administrátor a PowerUser.

Soubory modulu jsou umístěny v adresáři *modules/TextEditor*.

8.2.4. AKTUALITY

Pomocí integrovaného WYSIWYG editoru openWYSIWYG, umožňuje vytváření a editace krátkých článků a aktualit na internetové stránky. Text článků je možné libovolně formátovat stejně jako samotný text stránek. Lze vkládat obrázky, tabulky a odkazy.

Uživatelé skupiny User a PowerUser mohou vytvářet aktuality a editovat pouze aktuality, které uživatel vlastní. Členové skupiny Administrator mohou vytvářet a editovat veškeré aktuality.

8.2.5. SPRÁVCE GALERIÍ

Modul spravující jednotlivé galerie webových stránek. Ke galeriím jsou přidružovány fotografie nahrané na server pomocí modulu *Správa souborů*. Počet galerií vytvořených v systému není pevně omezen. Modul umožňuje také vytváření titulků pro jednotlivé fotografie a nastavení pořadí zobrazení fotografií.

Výstup na webových stránkách je realizován několika způsoby. Nejpoužívanější způsob je implementace pomocí komponenty LightBox JS. Tato komponenta umožňuje přehledné procházení jednotlivých fotografií v galerii. Další možností je galerie vytvoření pomocí technologie Flash. Tato galerie je řízena speciálním XML souborem, který je nutné vygenerovat po každé změně v galerii. Způsob zobrazení galerie se volí při implementaci do internetových stránek.

Funkce modulu *Správce galerií* jsou přístupné pro uživatele ze skupiny PowerUser a Administrator. Uživatelé patřící do skupiny User, nemají přednastavené oprávnění pro žádné operace.

8.2.6. SPRÁVCE MENU

Modul umožňuje dynamické vytváření navigace. Vytvořit je možno různé typy, které se specifikují až při použití na stránkách. Maximální hloubka vnoření navigace jsou tři stupně. Základní oprávnění pro tento modul mají přiděleno pouze uživatelé ze skupiny Administrator. Ostatní uživatelé nemohou nijak ovlivnit žádnou část výsledné navigace. Modul Správce menu tvoří později základ celé webové prezentace.

9. ZÁVĚR

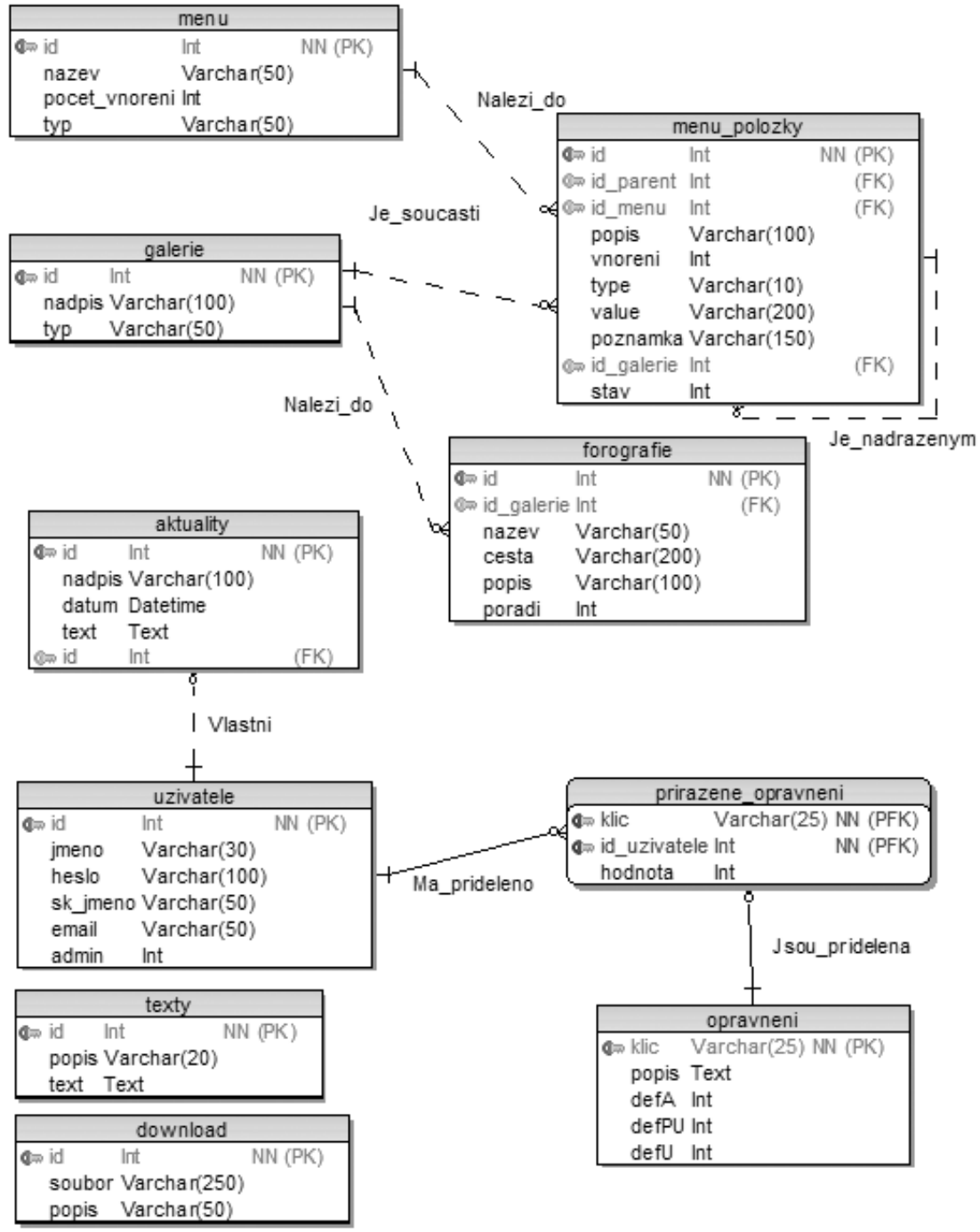
Dle mého názoru má aplikace dostatečný potenciál na to, aby se z ní stal opravdu komplexní systém, který by bylo možné implementovat do stránek bez jakýchkoliv znalostí jazyka PHP nebo HTML. Rád bych tedy, aby bylo do budoucna možné přes aplikaci přímo navrhovat grafický návrh stránek. Pro vytvoření takových stránek by potom bylo třeba pouze kopírovat soubory systému na webový server.

Další možností rozšíření by byla ještě větší integrace technologie AJAX, která je zatím použita pouze na dotvoření interface aplikace. Doufám, že tento redakční systém usnadní uživatelů práci při aktualizaci a budování obsahu internetových stránek. Dále také doufám, že jim systém umožní uspořit nemalé finanční prostředky, které by jinak byli nuceni vynaložit na údržbu a aktualizace stránek od vývojáře.

Cílem práce bylo vytvořit intuitivní, uživatelsky přívětivý redakční systém. Posouzení, zda se podařilo cíle naplnit, teď leží na budoucích uživatelích systému.

PŘÍLOHA A

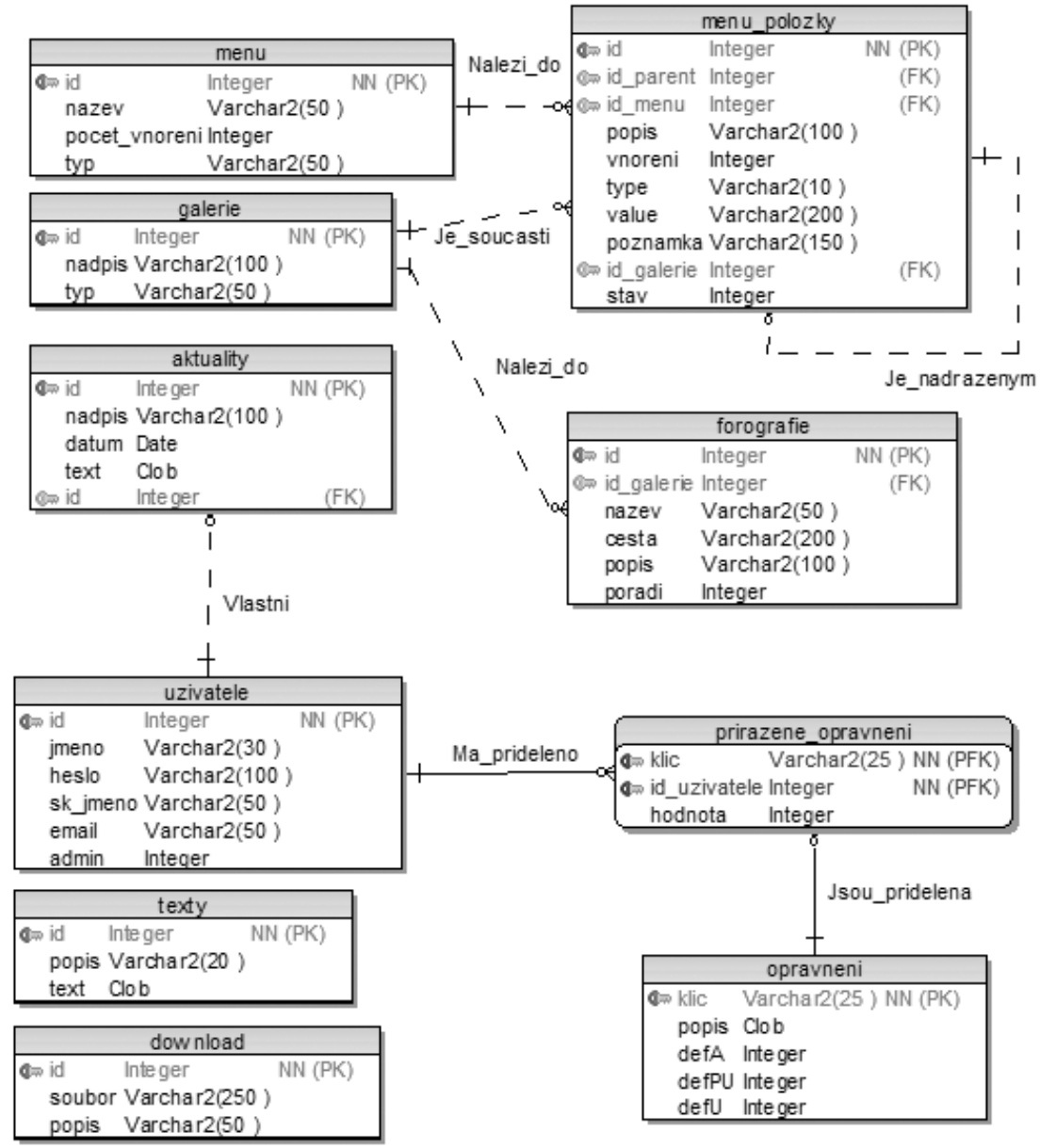
ER diagram databáze MySQL



Obrázek 11 - ER Diagram návrhu databáze pro MySQL. Vypracováno v programu Toad Data Modeler 3

PŘÍLOHA B

ER diagram databáze Oracle



Obrázek 12 - ER Diagram návrhu databáze pro Oracle. Vypracováno v programu Toad Data Modeler 3

POUŽITÉ ZDROJE

[1] CASTAGNETO, J. – RAWAT, H. – SCHUMANN, S. – SCOLLO, S. – VELIATH, D. *PHP Programujeme profesionálně*. 2. Vyd. Brno: Computer Press, 2004. 656 s. ISBN 80-7226-310-2. Kapitola 3, Programování v prostředí webu, s. 53-60

[2] Wikipedia. *PHP*[online]. 30 Březen 2009 [cit. 2009-04-11]. Dostupný z: http://cs.wikipedia.org/wiki/PHP

[3] Wikipedia. *Web kontent management systém*[online]. 11 April 2009 [cit. 2009-04-11]. Dostupný z: http://en.wikipedia.org/wiki/Web_content_management_system

[4] BISOM Robert. *Požadavky protokolu http a jejich zpracování v PHP*[online]. 10 Listopad 2002 [cit. 2009-04-24]. Dostupný z: http://interval.cz/clanky/pozadavky-protokolu-http-a-jejich-zpracovani-v-php/

[5] Wikipedia. *HTTP Cookie*[online]. 12 Duben 2009 [cit. 2009-04-24]. Dostupný z: http://cs.wikipedia.org/wiki/HTTP_cookie

[6] DARIE, C. – BRINZAREA, B. – CHERECHES-TOSA, F. – BUCICA, M. *AJAX a PHP tvoříme interaktivní webové aplikace profesionálně*. 1. Vyd. Brno: Zoner Press, 2006. 316s. ISBN80-86815-47-1. Kapitola 1, AJAX a budoucnost webových aplikací