Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Webový portál

Katolický David

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Webový portál" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 30.3.2010 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Pavlu Šimkovi Ph.D. za pomoc při psaní bakalářské práce.

Webový portál

Web portal

Souhrn

V bakalářské práci jsou popsány technologie tvorby webových stránek. Práce je zaměřena na skriptovací jazyk PHP a databázový systém MySQL. V praktické části práce je popsán rozvíjející se portál svahy.cz, který je zaměřen na zimní střediska v ČR, body zájmu a možnosti ubytovaní v blízkosti těchto středisek. V závěru práce jsou shrnuty výhody a nevýhody nového portálu a porovnání již existujících projektů s tímto novým.

Summary

The technologies of websites creations are described in this bachelor thesis. The bachelor thesis is focused on a scripting language PHP and a database system MySQL. In a practical section of the thesis the evolving portal svahy.cz is described, which is oriented on winter resorts in Czech Republic, points of view and possibilities of accommondation close to these resorts. At the end of the thesis advantages and disadvantages of the new portal are summarized and existing projects are compared with the above – mentioned portal.

Klíčová slova: Webový portál, PHP, MySQL, Administrace, Zimní střediska, Body zájmu, Ubytování

Keywords: Web portal, PHP, MySQL, Administration, Winter sport center, Point of interest, Accomodation

Obsah

1. Úvod

Internet přešel do komerčního užívání teprve před relativně krátkou dobou. Tudíž se dá říci, že ještě v nedávné době nebyl internet takovým standardem, jako je dnes. První webové prezentace sloužili především pouze k informovanosti lidí a to pomocí statických stránek.

Tyto statické stránky byly návštěvníkům poskytovány v podobě jednoduchých HTML stránek. Žádné skriptovací ani programovací jazyky ve spojení s webovými prezentacemi nebyly používány. Dá se říci, že to nemělo ani smysl, protože využití internetu mezi lidmi nebylo tak rozsáhlé, jako je tomu v současné době.

V průběhu několika let šel vývoj v této oblasti velmi rychle kupředu. Do popředí se dostaly 2 projekty. Prvním z nich je skriptovací jazyk PHP. Skripty jsou prováděny na straně serveru, kde server zpracuje požadavek, vytvoří HTML výstup a následně ho pošle klientskému prohlížeči, který obsah zobrazí. Velikou výhodou je, že PHP je nezávislý na platformě. Používané skripty, až na pár malých výjimek, fungují na mnoha různých operačních systémech. Další velikou výhodou je, že PHP umožňuje využívání mnoha knihoven, které se dají použít pro nejrůznější účely jako je práce se soubory, přímý přístup k většině databázových systémů, zpracování textu, grafiky nebo např. zpracování PDF dokumentů.

Druhým projektem je ASP (Active Server Pages) zastoupený společností Microsoft. Tato skriptovací platforma je též určená pro dynamické zpracování webových stránek na straně serveru. Její, v dnešní době používaný nástupce se označuje ASP.NET, který využívá .NET frameworku. Rozdíl ASP.NET oproti ASP i PHP je především ten, že v ASP.NET se nejdříve napíše prezentace a poté se musí zkompilovat. Toto má jednu výhodu – aplikace běží rychleji, je méně náročná na server a hlavně je většina chyb zachycena a odstraněna již při vývoji. Nevýhodou oproti PHP je ovšem v provozu. ASP.NET lze provozovat taktéž na různých operačních systémech, ale většina operačních systémů musí mít implementovánu .NET platformu. Problém nastává v tom, že ne všechny web hostingové společnosti tuto implementaci nabízejí.

2. Cíl práce a metodika

Cílem bakalářské práce je vytvoření portálu, který má být primárně zaměřený na zimní střediska v ČR se vším, co k nim patří. Má sloužit k získání kompletních informací o jednotlivých lokalitách. Od základních informací jednotlivých středisek, přes aktuální počasí a sněhové zpravodajství včetně krátkodobé předpovědi počasí, až po přehled služeb, které středisko poskytuje svým návštěvníkům.

Další zaměření portálu je pak na body zájmu v blízkosti středisek, přehled ubytování případně na alternativní využití času v blízkosti střediska kromě zimních sportů.

Metodikou práce je analyzování dostupné literatury pro získání dostatečných informací o dostupných technologiích. Tyto zdroje jsou použity v teoretické části práce.

V první kapitole je popsána stručná charakteristika skriptovacího jazyka PHP. Je zde popsáno v čem jsou charakteristické dynamické webové prezentace. Jakým způsobem se pracuje s formuláři na webových stránkách. Dále jsou uvedeny příklady využití jazyka PHP včetně ukázky syntaxí vybraných funkcí.

V další kapitole je popsána problematika databází, také zmínka o existenci komerčních databázových systémů a porovnání s databázovými systémy, které jsou dostupné zdarma. Uvedení SQL příkazů. Srovnání konzolové aplikace MySQL s nástrojem PHPMyAdmin na několika příkladech.

Literární zdroje jsou také použity pro získání potřebných schopností pro praktické vlastní zhotovení portálu. Dále je v praktické části pak popsán vlastní portál, jeho funkce, jednotlivé části, administrace portálu jak pro administrátora, tak pro externí zdroje obsahu.

V závěru práce je srovnání s již existujícími portály zaměřenými na podobnou tématiku.

3. Skriptovací jazyk PHP

3.1 Co je to PHP

„Označení PHP bylo původně zkratkou anglické fráze „Personal Home Page“. Tuto technologii vytvořil v roce 1994 Rasmus Lerdorf kvůli sledování návštěvníků svých stránek. S postupným nárůstem užitečnosti a možností využití této technologie se ujal nový název „PHP: Hypertext Preprocessor“.“[1]

PHP je vloženým skriptovacím jazykem. Když někdo řekne, že PHP je vložen do HTML, znamená to, že jej lze interpretovat přímo v kódu HTML, díky čemuž je vývoj dynamických webových prezentací snáze dostupný.

„PHP není programovací jazyk, je pouze skriptovací. Jazyk PHP je navržen, aby vykonal určitou činnost jako reakci na výskyt určité události – například když uživatel odešle vyplněný formulář nebo přejde na určitou adresu URL.“[1]

3.1.1 Dynamické webové prezentace

„Dynamické webové prezentace jsou flexibilní výtvory s velkou kapacitou. Přesněji je lze popsat jako aplikace, nikoli pouze jako stránky. Dynamické webové prezentace:

- *Reagují na různé vstupní parametry*
- *Mají často vlastní rozhraní, jehož prostřednictvím mohou správci obsah prezentace udržovat*
- *Mají vlastní „paměť“, které umožňuje registraci a přihlašování uživatelů, elektronické obchodování a podobné procesy*
- *Snáze se udržují a aktualizují. Kromě toho jsou lepším základem pro další rozšiřování“*[1]

„K tvorbě dynamických webových prezentací slouží mnoho různých technologií. V současné době dominují ASP (Aktive Server Pages) od firmy Microsoft, dále pak JSP (Java Server Pages), ColdFusion a PHP.“[1]

[1] ULLMAN, Larry. *PHP a MySQL, názorný průvodce tvorbou dynamických WWW stránek, první vydání.* Str. 12

Dynamické webové prezentace jsou v současné době již běžně v kombinaci s nejrůznějšími databázemi. Je to především ze dvou důvodů:

- Ukládaná data v souborech jsou neefektivní, pomalá a pro rozsáhlejší projekty naprosto nevyhovující.
- Na trhu existuje velký výběr nejrůznějšího řešení databází. Lze vybrat profesionální řešení databází jako např. Oracle, ale existují i varianty, které jsou plně zdarma, např. MySQL.

3.1.2 Proč právě PHP

V porovnání s ostatními zmíněnými alternativami pro vytváření dynamických webových prezentací, má skriptovací jazyk PHP několik zásadních výhod. „*Mezi jeho přednosti patří zejména jeho výkon, těsná integrace s většinou dostupných databázových systémů, stabilita, přenositelnost a téměř neomezené možnosti rozšiřování. Všechno je navíc zcela zdarma, protože jazyk PHP je dodáván s veřejným zdrojovým kódem. Dalším, velice významným argumentem, je relativně krátká zaučovací doba. Od svého uvedení na trh získává stále větší a větší uplatnění v praxi. V poslední době je více využívaný, než technologie ASP.*“[2]

[2] ULLMAN, Larry. *PHP a MySQL, názorný průvodce tvorbou dynamických WWW stránek, první vydání.* Str. 13

3.2 Úvod do jazyka PHP

Jak již bylo uvedeno, jazyk PHP je určený pro skriptování na straně serveru. To znamená, že kód, který programátor napíše v jazyce PHP, je uložen na serveru. Kdykoli uživatel navštíví webovou prezentaci napsanou v PHP, načte server kód PHP, zpracuje ho, vytvoří z něj HTML výstup a ten odešle webovému prohlížeči na straně klienta. Jeho webový prohlížeč tato data zpracuje stejně, jako standardní statické HTML stránky.[3]

3.3 Formuláře

Formuláře ve webových prezentacích poskytují cestu, jak uživatele požádat o různé informace. Formulář se skládá z mnoha elementů, což umožňuje uživateli zadat text, vybrat různé položky z menu, zadat text jako heslo, aby byly zobrazeny místo textu hvězdičky a podobně. Poté co uživatel vloží informace, může formulář odeslat ke zpracovacímu skriptu, který se postará o příchozí informace. Zaslané informace jsou uchovány na serveru, který musí být schopen tyto informace zpracovat například PHP skriptem. Skript může něco počítat, vyhodnotit, zkontrolovat, uložit do databáze, porovnat s daty v databází apod.

3.3.1 Prvky formulářů

Formuláře jsou záležitostí HTML, nicméně HTML je nedokáže nijak zpracovat. Zpracování je až poté na nějakém ze skriptovacích jazyků. Jak již bylo řečeno, ve formulářích můžeme využívat několika prvků. Každý formulář musí začínat značkou <form> a končit </form>. Lze využít několik parametrů formu. Nejběžnějšími jsou action kde uvádíme, která stránka se má otevřít po odeslání formuláře. Pokud tento parametr neuvedeme, otevře se stejná stránka na které jsme vyplňovali formulář. Další z parametrů je method. Tento parametr slouží k určení metody, jakou se formulář odešle. Je zde na výběr buďto metoda GET, při které se všechny vyplněné prvky formuláře zobrazí v URL adrese, nebo metoda POST, kde odesílání dat probíhá na pozadí a uživatel nic nevidí.

[3] ULLMAN, Larry. *PHP a MySQL, názorný průvodce tvorbou dynamických WWW stránek, první vydání*.str. 11-15

Text

Klasické textové pole. Vkládá se příkazem

```
<input type='text' name='...'>
```

Zdrojový kód č. 1 – Syntaxe input type text

Lze použít opět více parametrů, jako například maxlength kterým určíme maximální povolenou délku řetězce nebo value, kde se vyplní defaultní hodnota textového pole.

Textarea

Jak již anglický název napovídá, jedná se o jakési velké textové pole. Tento prvek používající se ve formulářích, je jako jeden z mála párový.

```
<textarea name="..." cols="10" rows="10">Text</textarea>
```

Zdrojový kód č. 2 – Syntaxe textarea

Jeho parametry cols a rows jsou nepovinné, ale nastavují velikost tohoto textového pole. Velikost se dá nastavit také přes CSS, pak se již dané dva parametry nemusí použít. Text, který je uveden mezi párovými značkami je na webové stránce zobrazen v tomto poli.

Menu

Prvek menu slouží k vytváření listovacích nabídek. I on je také párový. Vypadá takto:

```
<select name="..." size="1">
        <option value="hodnota">prvek v nabídce</option>
</select>.
```

Zdrojový kód č. 3 – Syntaxe select nabídky

Každý prvek menu v menu musí být uveden mezi značkami <option></option>. Parametr value představuje hodnotu proměnné, která bude odeslána při odeslání formuláře.

Radiobutton

Radiobutton je klasické přepínací tlačítko. Používá se například pro zvolení mezi několika variantami, např. muž / žena. Do kódu se zapisuje takto:

```
<input type="radio" name="..." value="m" />
```

Zdrojový kód č. 4 – Syntaxe input type radio

Na uvedeném příkladě, pokud uživatel zvolí tuto možnost, do odesílané proměnné se uloží „m“. Radiobutton je speciální v jedné věci, žádný z radiobuttonků, které mají stejný název (name) nelze zvolit najednou. Chovají se jako jakýsi přepínač.

Checkbox

Checkbox je klasické známé zaškrtávací tlačítko. Jeho iterpretace je:

```
<input name="..." type="checkbox" value="1" />
```

Zdrojový kód č. 5 – Syntaxe input type checkbox

Takto zapsané tlačítko bude nezaškrtnuté. Pokud je žádoucí, aby bylo defaultně zaškrtnuté, musí se přidat ještě parametr checked.

Submit

Submit je nepostradatelné tlačítko každého formuláře. Tímto tlačítkem odešleme všechny vyplněné údaje webovému serveru ke zpracování. Zápis vypadá takto:

```
<input type="submit" value="Odešli" />
```

Zdrojový kód č. 6 – Syntaxe input type submit

Hidden

Prvek hidden, je velice unikátní prvek. Z jeho anglického názvu je jasné, že bude schovaný. Tudíž ho uživatel nikde neuvidí. Dá se využít například pro rozlišení částí zpracovávaného skriptu. Pokud je nějaký formulář rozsáhlý a návštěvník webových stránek ho vyplňuje v několika krocích (např. e-shop), lze rozlišovat kroky tímto prvkem. Další využití je např. pro kontrolu, zdali už byl formulář odeslán aby nedošlo k jeho nechtěnému opětovnému odeslání.

```
<input type="hidden" value="1" name="krok" />
```

Zdrojový kód č. 7 – Příklad použití input type hidden

Samozřejmě využití tohoto prvku formuláře je mnohem rozsáhlejší.

3.3.2 Kontrola formulářů

„Když se při vývoji aplikace používá formulář, programátor očekává že uživatel zadá přesně ta data, na které je formulář připraven. Bohužel tomu není vždy tak – uživatel může v nějakém poli zapomenout nějaké informace nebo zapomenout vyplnit některé údaje. Tyto situace je pochopitelně nutné kontrolovat.“[4]

Kontrola formulářů může probíhat na dvou úrovních. Buďto na úrovni prohlížeče nebo na úrovni serveru.

- Na úrovni prohlížeče tuto kontrolu zajišťují JavaScripty
- Na úrovni serveru kontrolu zajišťují skriptovací jazyky

Touto kontrolou, jak již bylo uvedeno, se myslí například kontrola, jestli jsou vyplněna všechna pole, zdali vyplněné údaje odpovídají určité masce (např. telefon, email atd.), zdali jsou vyplněna dvě pole stejně (heslo + ověření hesla). Dále může být testováno, zdali zadaný údaj je číslo nebo ne, případně také délka zadaného řetězce.

3.3.3 Zapisování do souborů nebo databáze

Již dříve bylo uvedeno, že v dnešní době se využívají pro ukládání dat především databáze. Nicméně může nastat situace, kdy je lepší ukládaní určitých věcí do souboru. *„Tento soubor může být pouze dočasný, připravený k pozdějšímu zpracování, nebo trvalého charakteru pro uchování informací.*“[4] Pro práci existuje v PHP několik speciálních funkcí. Pro zapisování, respektive čtení, do souboru je potřeba nejdříve soubor otevřít. K tomu slouží funkce fopen. Pro vkládání dat do souboru slouží např. funkce fputs, kdy vkládáme po jednotlivých řádcích. Obdobně existuje funkce fgets pro získání jednotlivých řádek ze souboru. Po ukončení práce se souborem je ho potřeba uzavřít a uvolnit z paměti. To se provede funkcí fclose. Funkcí pro práci se soubory existuje více. Toto jsou pouze základní funkce pro práci se soubory.

[4] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky*. Str. 158

3.4 Posílání e-mailů pomocí PHP

Skriptovací jazyk PHP obsahuje mnoho nejrůznějších funkcí pro práci s kde čím např. s čísly, textem, obrázky atd. Jednou z nich je i funkce mail(). Tato funkce je v současné době velice hojně využívána. Nejčastěji se používá pro odesílání automatických reportů, případně jako upozornění na novinky na webových stránkách (především diskusní fóra).

Základní syntaxí této funkce je

```
mail(komu, předmět, zpráva);
```

Zdrojový kód č. 8 – Syntaxe funkce mail()

Ovšem v nyní by takto zaslanou zprávu většina provozovatelů e-mailových služeb zařadila jako spam. Proto se musí do funkce zahrnout i určité informace, které jsou obsaženy v hlavičce e-mailu, aby zaslané e-maily dorazily v pořádku.[5]

3.4.1 Hlavičky v e-mailu

Při odeslání e-mailu formou, jak byla uvedena v předchozí kapitole, dorazí e-mail, ve kterém bude uveden nějaký defaultně nastavený odesílatel. Adresa defaultního odesílatele je nastavená v souboru php.ini. Tento soubor ale ve většině případů není možno modifikovat. Proto nezbývá než upravit funkci mail(). „*Pro tento případ použijeme první nepovinný parametr funkce a tím je zaslání speciálních hlaviček. Skript pro odeslání emailu bude vypadat takto:*“[6]

```
<?PHP
$to = “test@seznam.cz“;
$from = “cilova_adresa@spolecnost.cz“;
$subject = “Předmět emailu“;
$message = “Text emailové zprávy“;
$headers = “From: David Katolický <dkatolicky@gmail.com> \n“;
mail($to,$subject,$message,$headers);
?>
```

Zdrojový kód č. 9 – Syntaxe funkce mail() s hlavičkou from

[5] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky*. Str. 189
[6] MACH, Jakub. *PHP pro úplné začátečníky, Druhé vydání.* Str 138

Položka From rozhodně není jediná hlavička která putuje v každém e-mailu. Lze nastavit i priorita.

U e-mailových zpráv existují 4 stupně priority:

- *1 = Highest Priority*
- *2 = High Priority*
- *3 = Normal Priority*
- *4 = Low Priority* [7]

```
$headers = "X-Priority: 1\n";
```

Zdrojový kód č. 10 – Syntaxe hlavičky pro nastavení priority

Lze určit i další parametry e-mailu:

```
//Nastavení adresy na kterou přijde chybová zpráva v případě nedoručení e-mailu
$headers = "Return-Path: <error@spolecnost.cz> \n";
//Nastavení pro odesílání e-mailu jako HTML + nastavení znakové sady těla zprávy
$headers = "Content-Type: text/html; charset=iso-8859-1 \n";
//Přidání e-mailové adresy na kterou má být zaslána kopie emailu
$headers = "cc: admin@seznam.cz \n";
// Přidání e-mailové adresy na kterou má být zaslána skrytá kopie emailu
$headers = "bcc: admin@seznam.cz \n";
```

Zdrojový kód č. 11 – Příklad dalších hlaviček které lze použít

3.5 Proměnné

V dynamických stránkách se bez proměnných rozhodně nelze obejít. „*Jedním z rysů PHP je, že po vás nepožaduje, abyste proměnné deklarovali předtím, než je použijete. Proměnné se vytvoří až v okamžiku, kdy do nich poprvé přiřadíte nějakou hodnotu. Hodnotu přiřadíte proměnné přiřazovacím operátorem „=“.*“[8]

[7] MACH, Jakub. *PHP pro úplné začátečníky, Druhé vydání.* Str 138
[8] WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Str. 58

3.5.1 Typy proměnných

„Typ proměnné se odkazuje na druh dat, která jsou v ní uložena.

PHP zná následující typy dat:

- ***Integer** – celá čísla*
- ***Double** – reálná čísla*
- ***String** – řetězce znaků*
- ***Boolean** – logické hodnoty true nebo false*
- ***Array** – pole používaná k uložení více položek stejného typu*
- ***Object** – objekty používáme k ukládání instancí tříd při objektovém programování.“*[9]

„Ve většině programovacích jazyků se v proměnné mohou uchovávat hodnoty pouze jednoho typu a tento typ musí být deklarován předem – například v jazyce C. V PHP je typ proměnné určen automaticky podle hodnoty, která v ní je uložena.“[9]

```
$promenna = 0; - Integer
$promenna = 0.00; - Double
$promenna = „text“; - String
```

Zdrojový kód č. 12 – Automatické určení typu proměnné

3.5.2 Proměnná chameleón

„PHP poskytuje ještě jeden typ proměnné – proměnou chameleón (v angličtině termín variable variables). Proměnné chameleón umožňují dynamicky měnit název proměnné. Funguje to tak, že použijete hodnotu jedné proměnné jako název té druhé.“[9]

```
$varname = ‘pocet‘;
$$varname = 5;
//v proměnné $pocet bude hodnota 5
```

Zdrojový kód č. 13 – Příklad použití proměnné chameleón

[9] WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Str. 58-60

3.5.3 Konstanty

V jazyce PHP lze jako ve většině programovacích jazyků vytvářet konstantní proměnné. *„Konstanty ukládají data podobně jako proměnné, ale s tím rozdílem, že je nikde ve skriptu nelze změnit. Tyto konstanty definujeme s použití funkce define.“*[10]

```
define('DPH', 20);
define('MAX_NOSNOST', 1220);
//při použití konstanty se nepoužívá $
echo DPH;
```

Zdrojový kód č. 14 – Příklad použití konstant

3.5.4 Obor proměnných

„Obor je termín, který se používá k určení oblastí ve skriptu, kde lze nějakou konkrétní proměnou používat (je tam viditelná). V PHP jsou čtyři druhy oboru:

- *Zabudované super globální proměnné, které jsou viditelné všude ve skriptu*
- *Globální proměnné deklarované ve skriptu jsou viditelné v celém našem skriptu, ale ne uvnitř funkcí*
- *Proměnné definované uvnitř funkce jsou lokální vzhledem k této funkci*
- *Proměnné používané uvnitř funkcí, ale deklarované jako globální, se odkazují na globální proměnné stejného názvu“* [10]

„Od PHP 4.2 mají pole $_GET, $_POST a některé další speciální proměnné svá vlastní pravidla týkající se oboru. Říká se o nich, že jsou super globální a jsou viditelné všude, uvnitř funkcí i mimo ně.

- *$_GLOBALS – pole všech globálních proměnných*
- *$_SERVER – pole proměnných prostředí serveru*
- *$_GET – pole proměnných předaných do skriptu metodou GET*
- *$_POST – pole proměnných předaných do skriptu metodou POST*
- *$_COOKIE – pole proměnných COOKIE*
- *$_FILES – pole proměnných vztahujících se k nahrávání souborů*
- *$_ENV – pole proměnných prostředí*
- *$_REQUEST – pole všech uživatelských vstupních proměnných*
- *$_SESSION – pole proměnných sezení“* [10]

[10] WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Str. 60,61

3.5.5 SESSION

„Session proměnné vznikly až v PHP 4. Stručně řečeno, sessions jsou tak trochu podobné cookie, až na to, že všechny záležitosti s nimi spjaté jsou v tomto případě vestavěny přímo v PHP. Pokaždé, když uživatel vstoupí na stránku s aktivovanými sessions, je mu okamžitě přiděleno unikátní identifikační číslo, které nám například umožní zkontrolovat, zdali onen uživatel nenavštívil naše stránky i v předchozí době. Práce se session je stále ještě považována za záležitost relativně novou a mnozí lidé neustále používají cookie a funkci setcookie. Pro inicializaci session je nutné použít funkce session_start() – ihned poté pro vás PHP samo vytvoří cookie. Tato cookie bude unikátní a bude uložena jak na straně serveru, tak na straně klienta. Platnost session končí v okamžiku, když zavřete svůj prohlížeč. Soubor na straně serveru je vygenerován v momentě, kdy do vaší session začnete vkládat proměnné a ve stejný moment PHP vytvoří několik webových proměnných. Jednou z proměnných vygenerovaných funkcí session_start je unikátní identifikační číslo, které přesně označuje volající prohlížeč – tato informace je potom uložena v globální proměnné $PHPSESSID nebo může být dostupná pomocí funkce sess_id(). Podobně jako v případě cookie, i session_start() musíme volat ještě před začátkem jakéhokoli HTML nebo PHP kódu.“[11]

3.5.5.1 Nastavení session

Abychom mohli vytvořit v session proměnné, musíme použít funkci session_register. Základní syntaxe je následující:

```
session_register („name“);
```

Zdrojový kód č. 15 – Syntaxe session_register

V této funkci „name“ představuje pojmenování proměnné. Přiřazení hodnoty je velmi jednoduché:

```
$name = hodnota;
```

Zdrojový kód č. 16 – Přiřazení hodnoty session proměnné

Tyto informace jsou uloženy v souboru, většinou v adresáři /tmp. Jméno souboru bude ono unikátní identifikační číslo, které můžeme zjistit z proměnné $PHPSESSID. [12]

[11] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 253

[12] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 240, 253 - 256

3.5.5.2 Vymazání session

V případě, že již session nejsou potřeba, lze je odstranit a zrušit pomocí funkce session_destroy(). Tento postup však zlikviduje i všechny proměnné, které jsou k dané session přiřazeny. V ideálním případě bychom tuto funkci měli volat až na konci všech našich skriptů – danou funkci voláme bez parametrů stejně jako session_start().

Pokud je potřeba zrušit pouze nějaké session proměnné, ale ne všechny, lze použít funkci session_unregister a jako parametr se zadá jméno session proměnné, která je potřeba zrušit.

3.5.6 Cookies

„Jednou z nejpoužívanějších metod dlouhodobějšího uchování hodnot na světové informační síti je použití cookies – malých textových souborů. Nejznámějším způsobem využití cookies je jejich uložení na straně klienta. Je toho docíleno tak, že cookie je po svém vytvoření uložena do paměti prohlížeče, ze které se posléze přesouvá na pevný disk klienta. Je důležité vzít ovšem na vědomí, že pokud klient na svém počítači nemá povolené cookies, neexistuje způsob, jakým bychom mu je mohli poslat. Vlastní textový soubor je většinou ukládán do domovského adresáře klientského prohlížeče.“[13]

„Cookies používají metody párů klíčových hodnot (klíč – hodnota). Poté, co je cookie ze serveru odeslána, nemůžeme ji číst až do té doby, než se klient znovu připojí. Cookie také nemůžeme odeslat v případě, že jsme předtím odeslali jiný HTML nebo PHP kód. Soubor cookie nesmí mít více, než 4kb. Cookies se skvěle hodí pro uchování malého množství informací o klientech. Mohou obsahovat šest komponent, které ovšem většinou všechny nepotřebujeme. U cookies potřebujeme minimálně tyto informace: jméno, hodnota a datum, kdy cookie vyprší.“ [13]

[13] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 240, 253 - 256

Komponenty cookie

- Hodnota – vlastní obsah cookie
- Exspirace – doba, po kterou je cookie validní
- Cesta – pro jaký adresář je cookie ze strany serveru validní – zpětné lomítko „\“ znamená, že je validní pro všechny adresáře
- Doména – Cookies jsou implicitně validní pouze pro doménu, do které patří webový server – kupříkladu kdyby server patřil do domény mydomain.com, cookies by byly validní třeba pouze pro domény mydomain.net nebo mydomain.co.uk
- Bezpečnost – je-li tato hodnota nastavena na 1, můžeme použít pomocí HTTPS zabezpečené spojení (SSL) [14]

3.5.6.1 Nastavení cookies

Pro nastavení cookie na klientské straně nabízí PHP funkci setcookie – obecná syntaxe této funkce je následující:

```
setcookie (jméno cookie, hodnota, exspirace, cesta, doména, míra bezpečnosti);
```

Zdrojový kód č. 17 – Syntaxe setcookie

Takže typické volání této funkce může vypadat takto:

```
setcookie („Cookie_test“, „nic zvláštního“, time()+43200, „/“);
```

Zdrojový kód č. 18 – Příklad setcookie

„*Vzorový příklad nemá nastavenou ani doménu, ani zabezpečení. Funkce setcookie() má implicitně nastaveno spojení normální a vaší doménu. Výše uvedená cookie se jmenuje „cookie_test“, jakže pro její volání použijeme proměnné $cookie_test, jejíž hodnota bude „nic zvláštního“. Tato cookie je validní 12 hodin (43 200 sec), takže k ní můžeme přistoupit od zaslání pouze 12 hodin do té doby, než nám jaksi vyčichne.*“ [14]

[14] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 241, 242

3.5.6.2 Vymazání cookie

Pro vymazání cookie existují 3 způsoby.

- Zapsání funkce setcookie stejně jako když jsme jí vytvářeli. Tento způsob ne vždy funguje.
- Použití funkce setcookie s tím, že ji nastavíme realistický časový úsek, což znamená, že cookie se po určité době stane nepoužitelná
- Použití setcookie s takovým datem, které již pominulo – tuto metodu můžeme použít pro vymazání cookie na straně klienta. Při použití tohoto způsobu však musí být všechny parametry stejné. Lze tedy použít:[15]

```
setcookie („Cookie_test“, „nic zvláštního“, time()-43200, „/“);
```

Zdrojový kód č. 19 – Příklad vymazání cookie

3.5.6.3 Využití cookies

„*Cookies jsou skutečně ve svém živlu v případě, kdy se snažíme nějakým způsobem personalizovat části našich stránek. Například Yahoo umožňuje svým návštěvníkům ihned při první návštěvě nastavit úvodní stránku tak, aby vyhovovala jejich zájmům. Aby tohoto Yahoo dosáhl, používá cookie s databází osobních nastavení.*“[15]

Cookies se dají také např. použít pro zapamatování uživatelského jména a hesla pro přihlášení do webových aplikací atp.

3.5.7 Správa proměnných

Ke správě proměnných se používají 3 základní funkce.

- *Isset() – ověřuje, zda určitá proměnná existuje*
- *Unset() – odstraní proměnné*
- *Empty() – ověřuje pravdivost hodnoty proměnné* [16]

[15] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 244
[16] GUTMANS, Andi a spol. *Mistrovství v PHP 5, První vydání.* Str. 48

3.5.7.1 Funkce isset()

„Tato funkce ověřuje, zda určitá proměnná byla v prostředí PHP deklarována. Byla-li proměnná nastavena, vrací funkce logickou hodnotu TRUE. *V opačném případě vrací hodnotu* FALSE, *což znamená, že proměnná obsahuje hodnotu* NULL *(prázdná nebo chybějící data)*“[17]

3.5.7.2 Funkce unset()

„Funkce unset() *ruší deklaraci dříve nastavené proměnné a v případě, že se na proměnnou zrovna neodkazuje žádná jiná proměnná, uvolňuje veškerou paměť, která by mohla být proměnou použita. Volání této funkce pro proměnnou, která již byla zrušena nebo neexistuje, vrací hodnotu* FALSE.“[17]

3.5.7.3 Funkce empty()

„Funkci empty() lze použít k ověření, zda proměnná byla deklarována nebo zda je její hodnotou hodnota FALSE. Tato funkce se obvykle používá při ověření formulářových proměnných. Umožňuje ověřit, zda uživatel vyplnil příslušné formulářové pole a zda formulářová proměnná obsahuje data. Při ověření zda proměnná obsahuje hodnotu, vrací funkce hodnotu TRUE nebo FALSE.“ [17]

3.6 Propojení PHP a MySQL

Pro práci s MySQL nabízí PHP mnoho různých funkcí.

„To že se webové stránky připojí k databázi a dokáží s ní nějakým způsobem pracovat, dodává webovým stránkám život. Zobrazení dynamických informací po stisku tlačítka dává stránkám teprve tu správnou šťávu.“[18]

3.6.1 Spojení s MySQL

Příkaz **mysql_connect** inicializuje spojení s MySQL serverem.

```
mysql_connect (“ip_adresa_serveru“,“uživatel“,“heslo“)
```

Zdrojový kód č. 20 – Syntaxe mysql_connect

Příkaz **mysql_pconnect** otevře do MySQL persistentní (trvalé) spojení.

```
mysql_pconnect (“ip_adresa_serveru“,“uživatel“,“heslo“)
```

Zdrojový kód č. 21 – Syntaxe mysql_pconnect

[17] GUTMANS, Andi a spol. *Mistrovství v PHP 5, První vydání.* Str. 48
[18] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 301, 302

„Uvedený příkaz funguje úplně stejně, jako mysql_connect, až na to, že když MySQL opustíme, zůstane spojení stále otevřené. Proto také po návratu do MySQL budeme moci použít toho samého spojení.“ [18]

Příkaz **mysql_select_db** přepíná do aktuální pracovní databáze.

```
Mysql_select_db(“jmeno_databaze“,“spojeni“)
```

Zdrojový kód č. 22 – Syntaxe mysql_select_db

3.7 Autorizace v PHP

Občas je potřeba určitou část webových stránek mít přístupnou pouze některým lidem. Přístup do těchto míst se dá omezit několika způsoby. Jedním ze způsobů je použití souboru .htaccess se statickým listem uživatelů a hesel, ale to je spíš přežitek doby minulé. V dnešní době existují v podstatě 2 způsoby, jak lze vyřešit autorizaci na určité části stránek.

Jednou z možností je využití předdefinovaných systémových proměnných PHP. *„Tyto proměnné umožňují PHP spolupracovat s HTTP. Pokud si uživatel vyžádá chráněný soubor, PHP vyvolá autorizační proces stejný jako v případě použití souboru .htaccess tím, že pošle prohlížeči příslušné hlavičky.“*[19]

```
If (!isset($PHP_AUTH_USER)){
//není zadán uživatel
header (“WWW-Authenticate: Basic realm=\“Poznamka napsana v autorizačním okne\““);
header  (“HTTP/1.0 401 Unauthorized“);
echo “Authorization Required“;
exit;
} else {
//ověřeno
echo “Tvoje přihlašovací jméno je: $PHP_AUTH_USER<br />“;
echo “Tvoje heslo je: $PHP_AUTH_PW<br />“;
}
```

Zdrojový kód č. 23 – Vyvolání autorizačního dialogu zasláním hlavičky

[18] GUTMANS, Andi a spol. *Mistrovství v PHP 5, První vydání.* Str. 48
[19] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 460

Tímto kódem se spustí dialogové okno vybízející uživatele zadat uživatelské jméno a heslo. Pokud uživatel zadá uživatelské jméno a heslo, vypíše se v následujícím skriptu, pokud nezadá a zmáčkne tlačítko storno, vypíše se „Authorization Required“ a skript skončí.[19] Pak už záleží jen na programátorovi stránek jakým způsobem bude ve skriptu zajištěno ověřování pomocí PHP skriptu, zdali zadané uživatelské jméno a heslo je správné nebo nikoli, případně jak webová stránka zareaguje.

Druhou z možností, která je v dnešní době velmi využívaná, je přihlášení pomocí formuláře přímo na stránkách. Tento způsob má širší uplatnění. Využívá se jednak ke zjištění uživatelského oprávnění prohlížet tu část stránek, která je kontrolována, nebo také třeba k identifikaci návštěvníků. Například u elektronických obchodů, emailových schránek kde by jiná alternativa byla téměř nerealizovatelná. „*Implementovat jednoduchou kontrolu přístupu není těžké. Následující příklad má tři možné výstupy jak se zobrazí. Pokud se stránka načte bez parametrů, zobrazí se HTML formulář, který od uživatele žádá, aby zadal své uživatelské jméno a heslo. Pokud jsou parametry zadány, ale nejsou správné zobrazí se chybová hláška. Pokud jsou parametry zadány správně, zobrazí se tajný obsah.*“[20]

```
<?PHP
$name = $HTTP_POST_VARS[‘name‘];
$psw = $HTTP_POST_VARS[‘psw‘];

If (empty($name) || empty($psw))
{
?>
<h1>Přihlašte se prosím</h1>
<form Metod=“post“ acrion=““>
Uživatelské jméno: <input type=“text“ name=“name“><br/>
Heslo: <input type=“password“ name=“psw“><br/>
<input type=“submit“ value=“Přihlaš“>
</form>
<?PHP
}
else if($name==‘user‘ && $psw==‘pass‘)
{
Echo “Přihlášení proběhlo úspěšně“;
}
else
{ Echo “Bylo zadáno špatné uživatelské jméno a heslo.“; }
```

Zdrojový kód č. 24 – Přihlášení pomocí formuláře

[19] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 460
[20] WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Str. 343, 345

Tento skript má ještě minimálně dva veliké nedostatky. Má pouze jedno uživatelské jméno a heslo se přenáší jako čistý text. „*Existuje mnoho lepších míst, kam uložit uživatelská jména a hesla, než je vnitřek skriptu. Uvnitř skriptu je obtížné tato data změnit.*“ [21] Naprosto jednoznačně nejlepší a nejčistší způsob je mít uživatelská jména a hesla uložená v databázi. V případě velkého množství uživatelů by bylo používání databáze uživatelů v textovém souboru příliš pomalé. Je mnohem jednodušší přidávat nové uživatele do databáze, případně mazat nežádoucí uživatele z databáze, nežli s nimi pracovat v souborech.

3.7.1 Šifrování hesel

„*Bez ohledu na to, zda ukládáme naše data do databáze, nebo do souboru, je zbytečné riskovat ukládání hesla jako čistý text. Jednosměrný hashovaní algoritmus nám poskytne jen s minimálním úsilím navíc o něco lepší zabezpečení.*“[21]

Existují 2 velmi používané hashovací funkce. Jednou z nich je funkce crypt, druhá je funkce, která vytvoří MD5 hash.

```
$zakodovane_heslo = crypt(“string který chceme zabezpečit“,“řetězec kterým se zabezpečuje“)
```

Zdrojový kód č. 25 – Použití hashování pomocí crypt

```
$zakodovane_heslo = md5(“string který chceme zabezpečit“)
```

Zdrojový kód č. 26 – Použití hashování pomocí MD5

Jakmile použijeme nějaké šifrování, už nezískáme původní hodnotu. Ale u hesel to ničemu nevadí. Není potřeba původní heslo. Stačí zadané heslo zašifrovat stejným způsobem a porovnat výsledky šifer, zdali se shodují.[21]

[21] WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Str. 343,345,348, 349

3.8 Funkce v PHP

Funkce v jakémkoli programovacím jazyce se používají ke srozumitelnému rozdělení kódu na menší funkční bloky. *„Každý takový blok vykonává na základě zadaných parametrů určitou činnost, případně vypočítá nějakou hodnotu. Může však také uložit např. údaje do souboru nebo databáze.“*[22] *„Dá se shrnout, že funkce je vlastně blok kódu, který lze jednou nadefinovat a pak jej vyvolávat z ostatních částí programu.“*[23]

3.8.1 Tvorba a užití funkcí

„K tvorbě funkce se používá klíčové slovo function, *za nímž následuje název funkce a v závorkách pak seznam argumentů. Volba názvu funkce přispívá ke srozumitelnosti kódu. Je proto zapotřebí volit takové názvy funkcí, které jsou krátké, ale výstižně charakterizují její určení. Název funkce musí začínat písmenem nebo podtržítkem. Potom mohou následovat písmena a číslice. V názvech funkcí se rozlišují malá a velká písmena. Argumenty funkcí jsou její lokální proměnné, proto pro ně platí pravidla proměnných. Tyto argumenty začínají znakem $ stejně jako proměnné.“*[22]

```
Function název ($argument1, $argument2, ... , $argument_x)
{
        //blok kódu funkce
        echo “Funkce provedla svůj kód“;
        return $promenna;
}
```

Zdrojový kód č. 27 – Syntaxe funkce

3.8.2 Platnost proměnných

„Každá proměnná má určitou oblast, ve které je její hodnota přístupná pod daným jménem. Je zřejmé, že proměnné deklarované uvnitř funkce jsou přístupné pouze v dané funkci, aby se náhodou nepřepsala proměnná, která je dostupná vně funkce. To platí prakticky ve všech programovacích jazycích. I když v PHP není třeba deklarovat proměnné a je možnost je použít rovnou, nelze použít ve funkci proměnnou deklarovanou vně funkce.“[24]

[22] LACKO, Luboslav. *PHP a MySQL Hotová řešení.* Str. 51
[23] CASTAGNETTO, Jesus a spol. *PHP Programujeme profesionálně, První vydání.* Str. 117
[24] BRÁZA, Jiří. *PHP 4 – učebnice základů jazyka, První vydání.* Str. 58

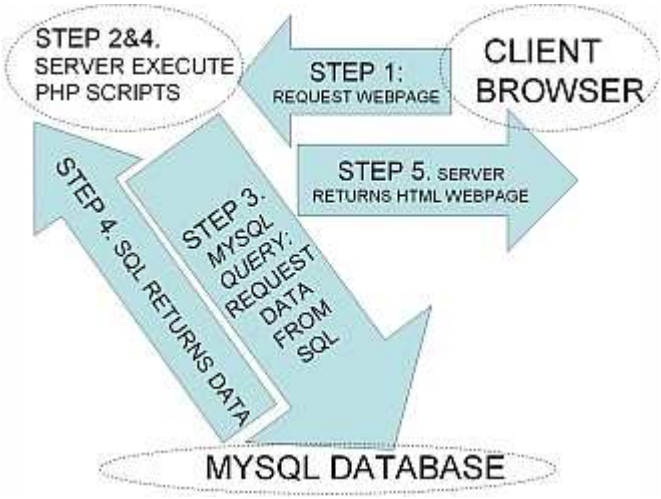
4. Databázový systém MySQL

V dnešní době naprostá většina webových stránek má dynamický obsah. Jak ale tento dynamický obsah zajistit? Jsou dvě možnosti uchovávání informací ze kterých se dynamický obsah vytváří.

- Data uložená v souborech
- Data uložená v databázi

Ukládání dat do souboru již není moc používané. Více méně se nyní používají pro uchovávání informací pouze databázové systémy a pomocí SQL dotazů se tyto informace z databáze získávají, případně do databáze vkládají.

Vztah popisující práci webového serveru a databáze vystihuje následující obrázek:



Obrázek č. 1 – Princip fungování databázového systému v kombinaci s PHP serverem (zdroj: http://images.devshed.com/ds/stories/Client_Browser_Request_JPG.JPG)

4.1 Co jsou to databáze

„Databáze ve své nejjednodušší podobě je určité úložiště obsahující neuspořádaná nebo spolu nějak související data. Aby byly webové stránky skutečně dynamické, je nevyhnutelné použití na pozadí aplikace běžící databázi. Databáze může sloužit jednoduché aplikaci, která bude například obsahovat statické informace o cenách produktů aktualizovaných denně nebo třebas personalizovaná nastavení

uživatele vašich stránek nebo aplikace nákupního košíku – seznam může být více méně nekonečný.“[25]

Nejčastěji používaný databázový software je MySQL, protože je aplikace pokrývající kompletně veškeré potřeby. Její největší výhodou je, že je zdarma.

Databáze se dá popsat jako určitá kolekce souborů, která obsahují setříděná data nebo informace. Tato data jsou v databázích uložena ve zvláštních strukturách, kterým se říká tabulky, přičemž každý řádek tabulky nazýváme jednotlivým záznamem. V každé tabulce je nejlepší, aby byl určen jeden sloupec jako primární klíč. Hodnota v tomto sloupci musí být unikátní pro každý řádek, proto je nejlepší ponechat tuto práci na samotném databázovém serveru a nechat generovat číselnou řadu. [25]

4.2 Co je to RDBMS

RDBMS je zkratka pro Relational Databáze Management Systém neboli Relační databáze. Rozdíl mezi normálním databázovým serverem a relačním databázovým serverem je ten, že v relačním je možné mít data uložena ve více tabulkách. Tyto tabulky se dají při dotazech do databáze spojovat a pokud je databáze správně navržena, neměly by vznikat zbytečné duplicitní záznamy. V dnešní době se dá říci, že všechny používané databázové systémy jsou právě relační. [25]

4.3 Komunikace s MySQL

„Všechny dobře známé databázové systémy umožňují práci s SQL. Hlavním důvodem pro vytvoření SQL bylo zřídit jednoduchý a mocný nástroj pro práci s databázemi, tabulkami – vkládat, číst a pojmenovávat určitá data.“ [25]

Jazyk SQL obsahuje nejrůznější příkazy a funkce. Veškeré funkce jsou zpracovány a do detailu popsány v dokumentaci např. již zmíněné MySQL, který lze stáhnout z www.mysql.com.[25]

S MySQL lze pracovat více způsoby. Můžeme psát příkazy přímo do konzole na serveru, kde je MySQL nainstalována. Ovšem tento způsob není příliš pohodlný, proto existuje několik správců databází. Jedním z nich je velmi oblíbený PHPMyAdmin, který má velice přívětivé uživatelské prostředí a téměř veškerá práce s databázovým systémem probíhá pomocí jednoduchých formulářů. Tento opensource projekt lze

[25] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 267-271

zdarma stáhnout z internetu a bezplatně využívat. Slouží především k vytváření nejrůznějších oprávnění uživatelů, vytvoření struktury databáze a tabulek. Po vytvoření základního konceptu databáze už veškerá práce s daty v tabulkách probíhá za pomoci SQL dotazů z nějaké aplikace, ať už webové nebo nějaké desktopové.

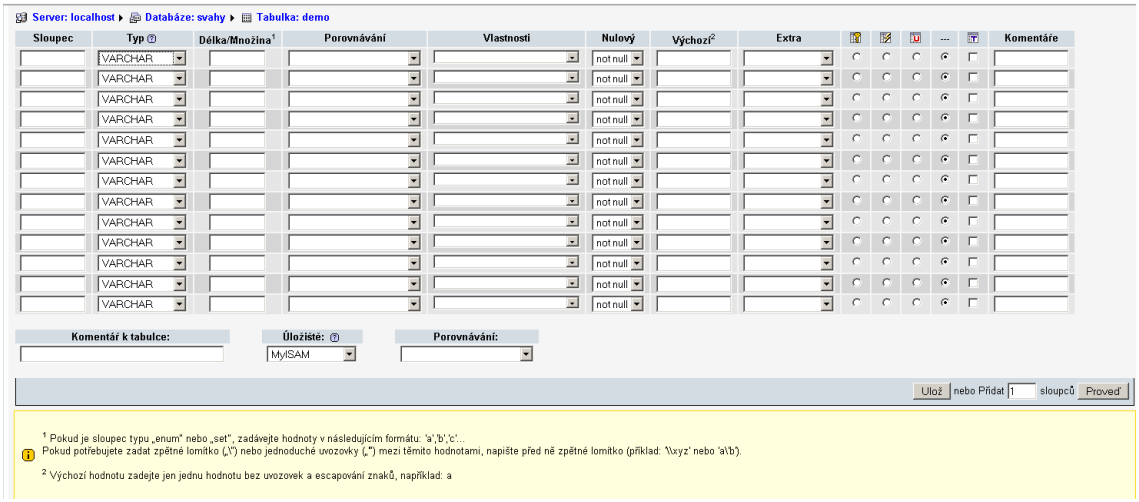
4.4 Databáze, tabulky

Jak již bylo uvedeno, každá databáze se skládá z tabulek.

„Při vytváření tabulek se zcela jistě vyplatí rozmyslet si, jaký druh informací bude v databázi skladován.“[26]

4.4.1 Vytvoření tabulky

Jednotlivé tabulky v databázi lze vytvořit přímým SQL dotazem nebo v některém již zmíněném zjednodušeném uživatelském rozhraní, kde probíhá vytvoření tabulky vyplněním jednoduchého formuláře. Porovnání obtížností dvou zmíněných způsobů lze vidět na následujícím obrázku.



Obrázek č. 2 – Příklad vytvoření tabulky v PHPMyAdmin

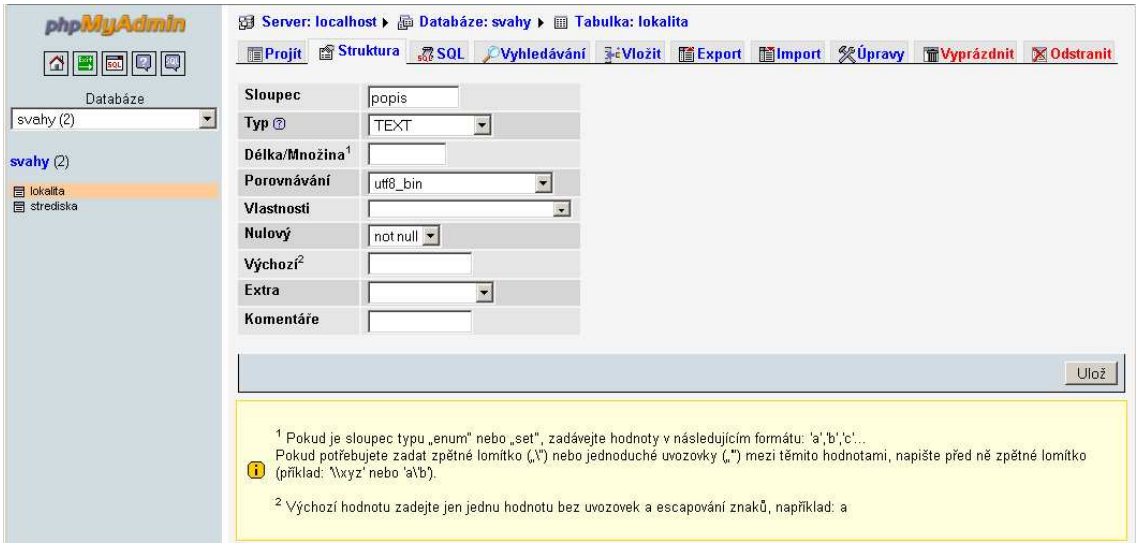
```
CREATE TABLE IF NOT EXISTS `strediska` (`id_strediska` int(11) NOT NULL auto_increment, `snazev` varchar(80) collate utf8_bin NOT NULL, `spopis` text collate utf8_bin NOT NULL, `lokalita_id` int(11) NOT NULL, `str-seo` varchar(60) collate utf8_bin NOT NULL, PRIMARY KEY (`id_strediska`))
```

Zdrojový kód č. 28 – Příklad vytvoření tabulky SQL dotazem

[26] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 272

4.4.2 Úprava struktury tabulky

Pokud jsou tabulky vytvořeny a úplně nevyhovují požadovaným potřebám, není problém je změnit. Opět jsou 2 možnosti: PHPMyAdmin, nebo SQL dotaz. Lze je porovnat na následujících ukázkách.



Obrázek č. 3 – Příklad úpravy sloupce v tabulce v PHPMyAdmin

```
ALTER TABLE `lokalita_demo` CHANGE `popis` `popis` VARCHAR( 50 ) CHARACTER SET utf8 COLLATE utf8_bin NOT NULL
```

Zdrojový kód č. 29 – Upravení sloupce v existující tabulce SQL dotazem

4.4.3 Vymazání tabulky a databáze

Stejně jako vytvářet a editovat, je také možnost vymazat cokoli. Lze smazat celou databázi, jednotlivou tabulku, celý obsah tabulky, ale také třeba jen některé záznamy.

4.5 Práce s daty v databázi

S daty jako takovými se pracuje především vždy prostřednictvím nějaké aplikace.

4.5.1 Vkládání do tabulek

Při vkládání nových dat je zapotřebí do dotazu uvést do které tabulky se mají vložit a čím mají být vyplněny jednotlivé sloupce tabulky. Vkládání dat do tabulky se provede následujícím SQL dotazem.

```
INSERT INTO `lokalita` (`id_lokalita`, `nazev`, `popis`, `lok-seo`) VALUES
(1, 'Jizerské hory', '', 'jizerske-hory');
```

Zdrojový kód č. 30 – Vložení záznamu do tabulky SQL dotazem

4.5.2 Zobrazení dat

Zobrazování dat probíhá obdobným způsobem jako vkládání. U vkládání určujeme pouze kam a co se má vložit. U dotazu pro získání dat z databáze se používají „filtry“.

„*Výraz SELECT je jedním z nejobvyklejších příkazů SQL a slouží právě pro vyvolání informací z databáze.*“[27]

Nejzákladnější syntaxe vypadá takto:

```
SELECT sloupec1, sloupec2,…. FROM jmeno_tabulky;
SELECT * FROM jmeno_tabulky;
```

Zdrojový kód č. 31 – Základní syntaxe SQL příkazu SELECT

Pokud je zapotřebí získat data na základě nějakého filtru – data, která mají něco společného, použije se do příkazu ještě část WHERE. Příkaz pak může vypadat např. takto:

```
SELECT * FROM `strediska` WHERE `lokalita_id` = 1
```

Zdrojový kód č. 32 – Příklad příkazu SELECT s částí WHERE

[27] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 280

4.5.3 Úprava dat

Pokud návštěvník webových stránek vloží nějaká data do databáze, např. prostřednictvím registrace, je dost možné, že tyto informace bude někdy v budoucnu potřeba změnit, aby byly aktuální. Pro nejen tento případ existuje příkaz UPDATE.

Update odstraní vše, co bylo v aktualizovaném poli předtím, takže je dobré se nejdříve přesvědčit, zdali výsledek podmínky WHERE je opravdu přesně takový, jaký je potřeba a že se neupraví a tudíž neztratí něco, co není žádoucí.[28]

```
UPDATE `svahy`.`lokalita_demo` SET `popis` = 'Beskydy jsou velice zajímavé místo' WHERE
`lokalita_demo`.`id_lokalita` =6;
```

Zdrojový kód č. 33 – Příklad příkazu UPDATE

4.5.4 Mazání dat

„*Občas nastane situace, kdy je zapotřebí nějaké informace vymazat – buďto celý záznam, nebo se třebas v tabulce objeví nesprávné informace.*“ [28]

Základní syntaxe příkazu DELETE vypadá takto:

```
DELETE FROM jmeno_tabulky WHERE jmeno_sloupce=“hodnota ve sloupci“;
```

Zdrojový kód č. 34 – Základní syntaxe příkazu DELETE

„*Při mazání záznamů by se k identifikaci měl vždy používat sloupec s ID, aby byl záznam jednoznačně určený.*“[28] Pokud je zapotřebí odstranit z tabulky více záznamů na základě stejných kritérií (např. zaměstnanci zrušeného podniku), lze použít právě tento identifikátor namísto ID.

[28] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 283 - 285

4.5.5 Řazení

Při velkém množství dat v databázi je vhodné, když jsou při výpisu nějakým způsobem seřazeny. K tomu slouží část příkazu ORDER BY, která se připisuje na konec příkazu SELECT. Lze také použít dodatek DESC, který zajistí, že výsledek dotazu bude vypsán v sestupném pořadí.

```
SELECT nazev FROM lokalita ORDER BY nazev
SELECT nazev FROM lokalita ORDER BY nazev DESC
```

Zdrojový kód č. 35 – Příklad řazení vzestupně a sestupně

4.5.6 Omezení počtu výsledků dotazu

„Při práci s velkým množstvím záznamů je občas dobré nějakým způsobem omezit počet vypisovaných záznamů. K tomuto účelu slouží funkce LIMIT – její základní syntaxe vypadá takto:“[29]

```
SELECT * FROM tabulka LIMIT n,m
```

Zdrojový kód č. 36 – Základní syntaxe funkce LIMIT

Kde n je množství zobrazovaných záznamů a m je pozice, od které má být určený počet záznamů vypsaný. Parametr m může být vynechán.

4.5.7 Vyloučení duplicitních záznamů

„Při některých příležitostech je potřeba vypsat obsah některého sloupce, ovšem by bylo dobré z výstupu vyloučit duplikované záznamy. Funkce DISTINC tuto činnost vykoná za nás a to v každém sloupci, který na který je použita.“ [29]

Základní syntaxe vypadá takto:

```
SELECT DISTINC sloupec FROM tabulka;
```

Zdrojový kód č. 37 – Základní syntaxe příkazu DISTINC

4.5.8 Počet výsledků dotazu

Při práci s rozsáhlými tabulkami obsahujícími velké množství záznamů není jejich součet zcela triviální. Naštěstí existuje funkce COUNT, která nám bez problémů spočítá všechny vybrané záznamy.[30]

[29] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 287

Základní zápis pak vypadá následovně:

```
SELECT COUNT(*) AS pocetstredisek FROM strediska;
SELECT COUNT(*) AS pocetstredisek FROM strediska WHERE lokalita_id=1;
```

Zdrojový kód č. 38 – Základní syntaxe funkce COUNT

4.5.9 Práce s více tabulkami

V praxi není výjimkou pro redukci duplicitních dat používání dvou a více tabulek navzájem propojených.

„Práce s více, než jednou tabulkou a výrazy INSERT a SELECT přidává do hry tzv. spojovací operace – tyto operace se mohou vztahovat buď k určitému výběru dat, nebo vráceným výsledkům. Spojování (Join) je jednou z klíčových operací, které odlišují klasický databázový systém od RDBMS. Základní přidanou hodnotou (pokud ne tou největší) spojování je absolutní flexibilita v přidávání nových tabulek do databáze. Lze tedy přidat různé nové tabulky obsahující pestré množství informací a poté použít spojení k tomu, aby tyto tabulky utvořili jakýsi celek.“ [30]

Zde je složitější příklad SELECTu se spojením několika tabulek dohromady:

```
SELECT * FROM t_pisemky,t_predmety,t_znamky,t_zaci WHERE
t_znamky.pisemka_id=t_pisemky.id_pisemky AND t_znamky.zak_id=t_zaci.id_zak AND
t_pisemky.predmet_id=t_predmety.id_predmet AND t_znamky.zak_id=".$_GET[zak]." AND
t_pisemky.datum>'".$data['od']."' AND t_pisemky.datum<'".$data['do']."' GROUP BY
t_predmety.predmet;
```

Zdrojový kód č. 39 – Příklad SELECTu se spojením několika tabulek dohromady

[30] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Str. 288, 298

4.5.10 Záloha databáze

Poté, co jsou tabulky naplněny daty, je na místě záloha dat, aby v případě nějakého problému data nezmizela. „*MySQL k tomuto účelu poskytuje jednoduchou pomůcku mysqldump, která umí zálohovat jak jednotlivé tabulky, tak celé databáze – tento nástroj je spouštěn z příkazové řádky MySQL. Záloha se běžně provádí každý den.*“[31]

```
mysqldump jmeno_databaze jmeno_tabulky
```

Zdrojový kód č. 40 – Základní syntaxe mysqldump

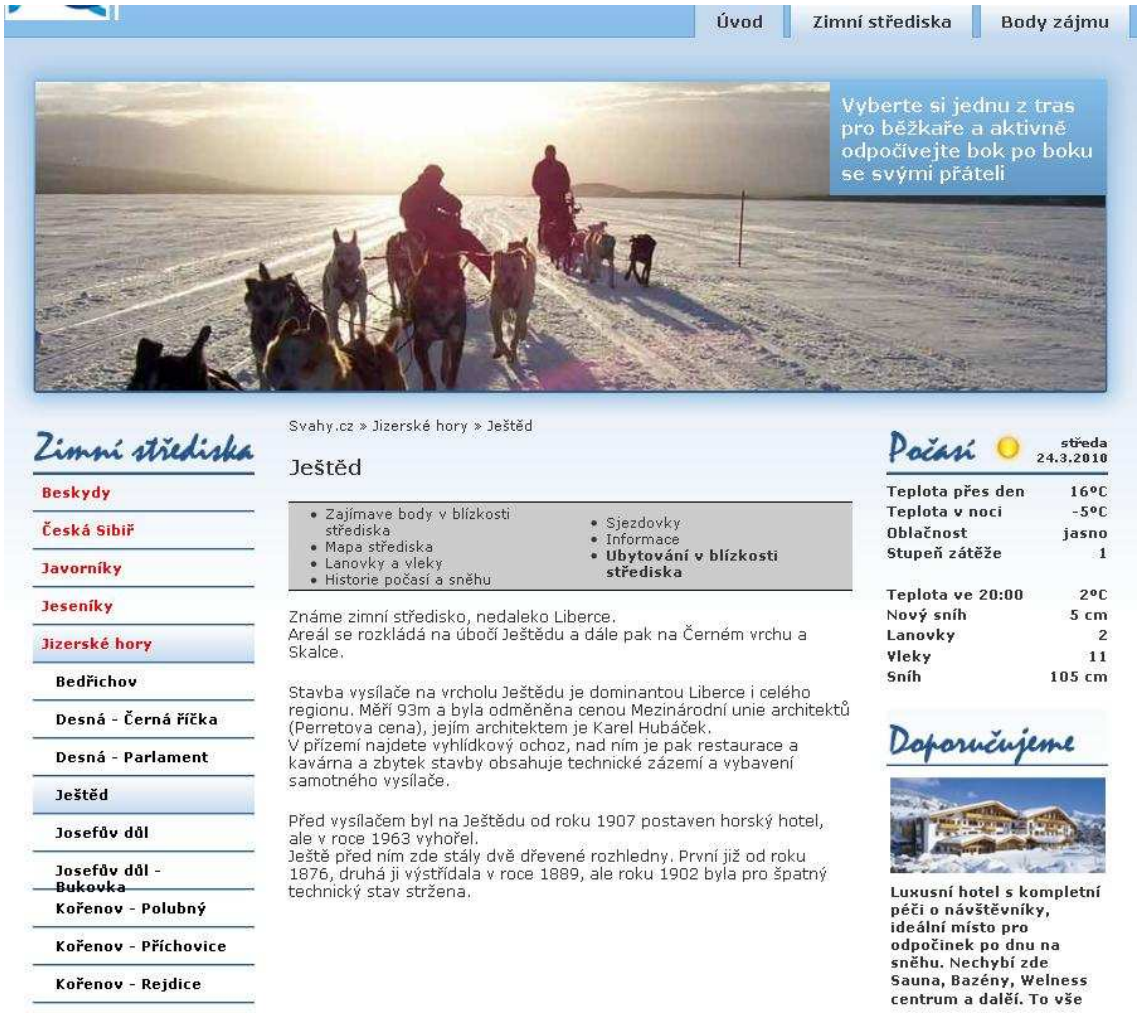
„*Z toho tedy vyplývá, že lze zálohovat jak celou databázi (všechny tabulky), tak i jednotlivé tabulky zvlášť.*“[31]

„*Vyexportovaná databáze bude obsahovat všechny výrazy CREATE potřebné pro vytvoření všech tabulek, takže lze použít tohoto souboru pro kompletní obnovu dat včetně tabulek. Všechny tabulky jsou vyexportovány v takovém formátu, aby bylo velice jednoduché pomocí výrazů INSERT vložit všechna data zpět.*“ [31]

[31] TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky*. Str. 298

5. Realizace a provoz portálu

Spuštění tohoto portálu je plánováno na zimní sezónu 2010/2011. Do této doby bude vývoj portálu plně dokončen a bude ve stavu, jak je naplánován.



Obrázek č. 4 – Vzhled portálu svahy.cz

5.1 Provoz

Provoz bude probíhat na některém z komerčních webhostingových programů, nebo na vlastním linuxovém serveru. Portál je psaný v jazyce PHP s využitím databázového systému MySQL.

5.1.1 WebHosting

Při rozhodování, který webhostingový program zvolit, je zapotřebí řídit se především referencemi ostatních uživatelů internetu o kvalitě těchto služeb. V dosavadní době – v době vývoje portálu - běží celý web na soukromém linuxovém serveru, který je určen pro jiný web. Po dokončení bude přesunut na některý z placených hostingů, případně poběží na speciálním serveru.

5.1.2 PHP + MySQL

Při rozhodování, zda pro tento projekt zvolit technologii PHP nebo ASP.NET bylo na základě srovnání obou technologií, výhod a nevýhod, zvolena technologie PHP. Kromě dosavadních znalostí jazyka PHP hrál proti ASP.NET fakt, že v současné době všechny hostingové společnosti nepodporují zmíněnou platformu .NET a u některých se za tuto službu dokonce i platí, což je, minimálně při vývoji, zbytečné. Nakonec byla tedy zvolena kombinace skriptovacího jazyka PHP a databázového systému MySQL.

5.2 Hlavní účel portálu

Hlavní účel portálu je informovat návštěvníky stránek o jednotlivých zimních střediscích v ČR, o počasí ve střediscích, o sněhovém zpravodajství atd.

5.2.1 Kompletní přehled středisek v ČR

V databázi portálu jsou všechna větší lyžařská střediska v ČR. Tato střediska jsou rozdělena v menu podle lokality, kde se nacházejí. Po výběru daného střediska je zobrazeno návštěvníkovi několik informací o vybraném středisku co se týče historie, umístění, velikosti atp. U každého střediska je submenu, ve kterém lze mimo základních informací o středisku zobrazit i podrobnější informace o středisku, dále informace o jednotlivých sjezdovkách, počty lanovek, vleků a délky sjezdových tratí. Nechybí ani mapa střediska a ceník jízdného. Speciálními položkami jsou zde „Historie počasí a sněhu“ a „Zajímavé body v blízkosti střediska“ které jsou následně popsány v kapitole 5.2.3 a v kapitole 5.3.

5.2.2 Počasí a sněhové zpravodajství

V průběhu celého pohybování po webovém portálu je zobrazen panel s počasím. Na tomto panelu jsou vždy zobrazeny údaje o počasí v lokalitě, která je momentálně zvolena.

Dále je v tomto panelu zobrazeno stejným způsobem sněhové zpravodajství. V případě že návštěvník nemá zvolené konkrétní středisko, ale pouze lokalitu, zobrazí se v tomto panelu průměrné teploty, oblačnost, stupeň zátěže i množství sněhu pro celou zvolenou lokalitu. Při výběru konkrétního střediska se pak zobrazí konkrétní informace pro dané středisko.

Počasí jednotlivých lokalit je získáváno službou společnosti Google přes Google API. Tato služba je volaná serverem každé 4 hodiny a veškerá data jsou aktualizována. Proto zobrazené počasí je stále aktuální. Počasí z Google API je získáváno speciální funkcí, která získává potřebné informace a zapisuje je do databáze. Tato funkce je uvedena v příloze 1.

Velice podobným způsobem jsou získávána data o sněhovém zpravodajství od společnosti Sitour, která se zabývá právě monitoringem sněhového zpravodajství, teplot, lyžařských podmínek, aktuálním počtem provozovaných vleků a sjezdových tratí u středisek v ČR. Tato společnost poskytuje zdarma XML soubor s veškerými těmito informacemi, který je volně k vyžití. Je pouze potřeba ho zpracovat. Funkce, která zpracovává tento XML soubor je uvedena v příloze 2.

5.2.3 Historie počasí a sněhu

Již v úvodu této kapitoly byla zmínka o speciální položce v menu. Tou je historie počasí a sněhu. Tato jednoznačně unikátní funkce zmiňovaného portálu monitoruje každé 4 hodiny teplotu a sněhové podmínky jednotlivých středisek a ukládá je do databáze. Při zobrazení této nabídky se na portálu zobrazí graf, ze kterého je patrné jak se měnilo počasí a sněhové podmínky v daném středisku. Je zde na výběr několik časových úseků, pro které lze graf zobrazit. Délky těchto úseků jsou 4 dny, týden, měsíc, 3 měsíce nebo rok.

Mimo zobrazení grafu historie počasí se v těchto místech zobrazuje také předpověď na následující 3 dny.

5.3 Doplňkové funkce

Vzhledem k tomu, že na horách existuje více míst, než pouze lyžařské střediska, obsahuje portál také databázi bodů zájmů (POI).

5.3.1 Seznam bodů zájmů – POI

Pro návštěvníky je připravena databáze bodů zájmů kde u každého bodu zájmu jsou uvedeny základní informace o umístění, popisu, kategorie daného bodu zájmu, GPS souřadnice a galerie fotografií. Základní informace o bodech zájmu lze v první fázi portálu zadávat pouze přes administrační prostředí celého portálu. Foto z místa bodu zájmu do fotogalerie může vložit kterýkoli návštěvník. Tyto fotky podléhají schválení administrátorem portálu. Do budoucna je v plánu zavedení přidávání bodů zájmů samotnými uživateli portálu.

Bohužel v současné době tato databáze obsahuje pouze pár bodů zájmu spíše pro potřeby testování. Databáze se bude rozšiřovat neustále i za běhu portálu.

5.3.2 Vzdálenosti mezi jednotlivými POI

Další zajímavou funkcí portálu je počítání vzdáleností mezi jednotlivými body zájmu. Tyto vzdálenosti se počítají pouze jednou a to vždy při vložení nového bodu zájmu. Po vložení se automaticky pomocí uvedených GPS souřadnic u bodů zájmů dopočítá vzdálenost od nového bodu zájmu ke všem již v minulosti zadaným. Tento systém umožňuje plánované zobrazování 20ti nejbližších POI od momentálně zvoleného bodu. Tudíž návštěvník může listovat mezi jednotlivými body zájmu nejen ze seznamu POI, ale také vždy pomocí nejbližších bodů zájmů od bodu zájmu, který si momentálně prohlíží.

5.4 Ubytování

Seznam ubytovacích zařízení je v dnešní době standardem u podobně zaměřených portálů jako má být tento. Proto nebude chybět ani zde. Implementace databáze ubytování a jejího zobrazování v současné době není hotová. Je naplánována do budoucna k začátku zimní sezóny 2010/2011. Nicméně již v současnosti je vize, že bude zaměřena na kvalitu, nikoli na kvantitu. Bude probíhat přímým oslovením poskytovatelů ubytování, aby informace byly pokud možno 100% správné a úplné. Pro tyto účely je také připraven panel „Doporučujeme“ v pravé části stránek. Zde bude vždy náhodně vybráno některé z ubytovacích zařízení z lokality, kterou momentálně návštěvník prohlíží.

5.4.1 Základní databáze ubytovacích zařízení

Jak již bylo zmíněno, z pohledu ubytovacích zařízení bude brán zřetel na kvalitu, nikoli na kvantitu. Cílem popisovaného portálu není mít částečné informace o tisíci ubytovacích zařízení, ale naopak kompletní informace třeba pouze o desítkách ubytování.

5.4.2 Vzdálenosti od jednotlivých středisek a POI

Stejně jako jsou u bodů zájmů uvedeny GPS souřadnice, jsou taktéž uvedeny u jednotlivých středisek a ubytování. Proto i v těchto místech portálu budou uvedeny z daného ubytovacího zařízení vzdálenosti do 5ti nejbližších středisek a klasicky již zmíněných 20ti nejbližších bodů zájmů. Tyto informace mají návštěvníkům nabídnout službu plánování návštěv různých středisek, výletů k bodům zájmů – ke kompletnímu přehledu, co se nachází v nejbližším okolí od potencionálně vybraného ubytování.

5.5 Administrační sekce pro ubytovací zařízení

Snahou je poskytnout kompletní informace o ubytovacích zařízeních. Proto zde bude pro ubytovací subjekty k dispozici administrační část, ve které každý vyplní formulář, který by měl pokrýt kompletní informace, které návštěvníky zajímají.

5.5.1 Přidávání ubytovacích zařízení

Stejně jako je plánováno přidávání bodů zájmů od uživatelů, bude k dispozici volné vkládání do databáze ubytovacích zařízení. Po vložení nového zařízení bude stejně jako u bodů zájmů potřeba kontrola ze strany administrátora, zdali uvedené informace jsou dostatečné a zdali vyhovují kompletnosti informací o vloženém středisku. V případě, že informace nebudou vyplněny podle představ administrátora, bude poskytovatel ubytovaní vyzván k doplnění informací, nebo jeho možnost ubytování nebude na portálu zveřejněna.

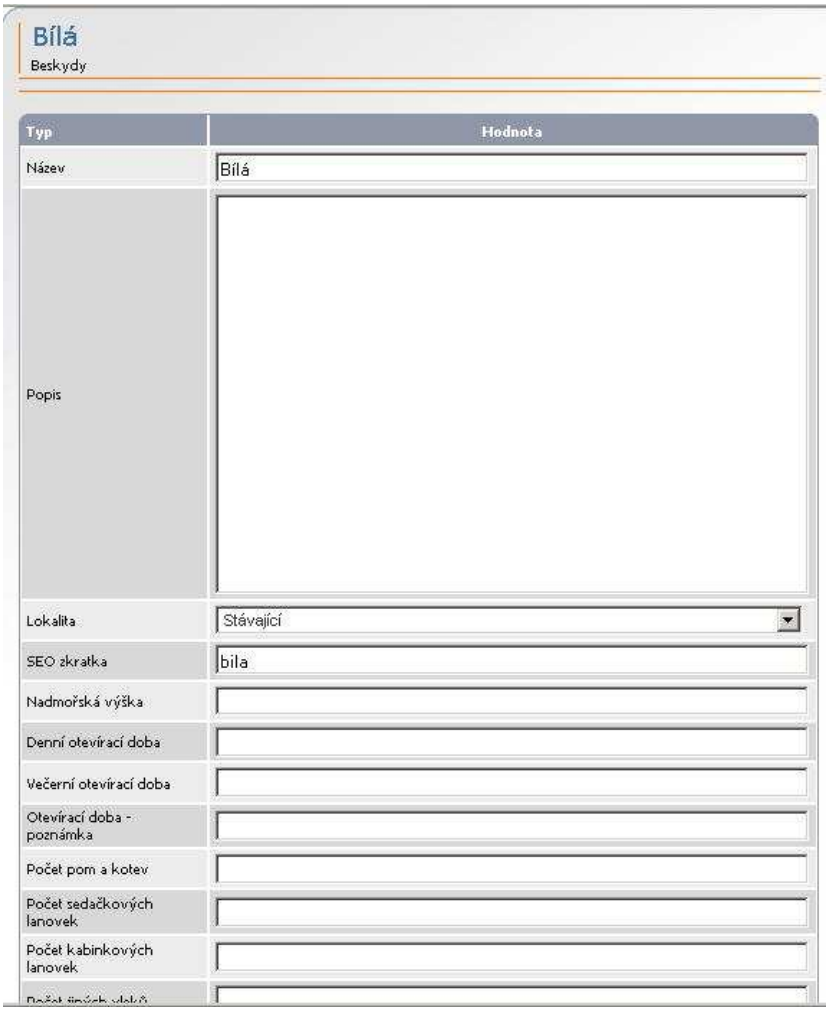
5.6 Administrace portálu

Administrační rozhraní portálu je graficky zcela odlišné. Je zaměřeno na přehlednost a jednoduchou správu celého obsahu webového portálu. Obsahuje jednotlivé kategorie, ve kterých lze upravovat veškeré informace.

Obrázek č. 5 – Vzhled administračního rozhraní portálu svahy.cz

5.6.1 Administrace středisek

Administrace středisek v současné době umožňuje pouze editaci informací o středisku – toto lze vidět na obrázku. Jak bylo uvedeno v jednotlivých podkapitolách bakalářské práce v části popisující funkce portálu, některé funkce se plánují až pro sezónu 2010/2011. Proto v administraci jsou připraveny další pole pro přidání administrování např. zmíněných fotografií, které budou moci uživatelé sami vkládat.



Obrázek č. 6 – Vzhled administračního rozhraní portálu svahy.cz – editace střediska

5.6.2 Administrace POI

Administrace bodů zájmů je momentálně možná pouze administrátorem portálu přes hlavní administrační prostředí. Pro jednotlivé návštěvníky je v současné době pouze možnost vkládání fotografií s popiskem k jednotlivým bodům zájmu. Možnost uživatelům vkládat samotné body zájmu je plánováno až po zaběhnutí portálu.

6. Porovnání s existujícími portály

Podobně zaměřených portálů je relativně mnoho. Proto je snahou nabídnout návštěvníkům přímo informace, které je zajímají. Nabídnout je v přehledné formě pro všechny věkové kategorie návštěvníků bez ohledu na zručnost využívání internetu.

6.1 Výhody nového portálu

Mezi hlavní výhody patří již zmíněné kompletní a přehledně uspořádané informace o jednotlivých střediscích. Dále pak informace o sněhu a teplotě. Největší silou tohoto portálu bude přehledné procházení bodů zájmu. Také informace o ubytování, které budou zaměřeny na kvalitu jak již bylo také popsáno, poskytnou návštěvníkům kvalitní a pravdivé informace o ubytování.

6.2 Nevýhody nového portálu

Asi největší nevýhodou je v současné době velice malá databáze bodů zájmů. Další nevýhodou je, že informace o ubytování je teprve ve fázi vývoje, a proto není ve stránkách zahrnut. Jak již bylo několikrát zmíněno v průběhu bakalářské práce, portál se plánuje spustit pro sezónu 2010/2011, proto stále chybí některé části portálu. Do této doby budou všechny uvedené nevýhody snad kompletně odstraněny.

6.3 Cílová skupina lidí

Cílovou skupinou lidí pro které je portál připravován jsou především lidé, kteří provozují zimní sporty v rámci ČR. Chtějí se dozvědět informace o střediscích, kde si nejlépe užijí zimní sporty, případně zde naleznou možnosti alternativního využití času v blízkosti zimních středisek.

6.4 Porovnání s konkurenčním portálem ceskehory.cz

Portál ceskehory.cz je v provozu již déle než 10 let. I přes to má oproti nově zakládanému portálu svahy.cz na první pohled nemoderní design. Informace na portálu jsou relativně nepřehledně uspořádané a těžko se v nich návštěvník orientuje. Na druhou stranu zde mají velice zajímavý a rozsáhlý obsah zaměřený nejen na zimní sezónu na sjezdové lyžování, ale také na běžky a v letních měsících na cyklistiku a turistiku. Na tomto portálu mají přesný opak toho, co je cílem portálu svahy.cz. Seznam ubytování mají zaměřený na kvantitu, nikoli na kvalitu. Dalším faktem je, že tento portál spravuje informace nejen o českých horách, ale i o horách na Slovensku a o horách a pobřežích v Chorvatsku. Portál svahy.cz bude zaměřen minimálně v první fázi pouze na ČR.

6.5 Porovnání s konkurenčním portálem skinet.cz

Portál skinet.cz má zajímavý design ovšem velikou slabostí je nepřehledné rozdělení částí webu. Na tomto portálu jsou všechny informace zobrazeny najednou. Nevýhodou portálu je, že obsahuje pouze základní informace o střediscích, počasí a sněhové informace. Obrovskou výhodou naopak je, že není zaměřen pouze na ČR, ale poskytuje informace o střediscích v Německu, Francii, Švýcarsku, Rakousku, Slovensku, Itálii a Slovinsku.

6.6 Porovnání s konkurenčním portálem lyzovani.cz

Tento portál je zaměřen trošku jiným směrem. Informace o střediscích jsou na spíše jako doplňkové informace. Portál je zaměřen na služby jako je poskytování ubytování, poskytování zájezdů na lyžování, online prodej nového i bazarového zboží. Katalog firem se zaměřením na zimní radovánky od půjčoven lyžařského vybavení až po cestovní kanceláře poskytující zájezdy. Dále jsou tu nejrůznější články např. o skialpinismu, závodech na lyžích, závodnících atd.

6.5 Porovnání s konkurenčním portálem lyzuj.cz

Tento poslední porovnávaný portál poskytuje také spíše zájezdy jednotlivých cestovních kanceláří do různých středisek po celé Evropě, kde jako doplňkové informace jsou uvedeny střediska, do kterých jsou tyto zájezdy vypravovány

s nejzákladnějšími informacemi o těchto střediscích. Jsou zde taktéž informace o počasí, sněhovém zpravodajství atd. Nechybí ani online rezervace vybraných ubytovacích subjektů, kteří spolupracují s tímto portálem. Ovšem informace o lyžařských střediscích a ubytovacích zařízeních jsou minimální a z portálu pouze odkazují na jednotlivé stánky zimních středisek a ubytovacích zařízení.

7. Závěr

Cílem bakalářské práce bylo vytvoření portálu, který má být primárně zaměřený na zimní střediska v ČR se vším, co k nim patří. Výsledný portál má sloužit k získání kompletních informací o jednotlivých lokalitách.Od základních informací jednotlivých středisek, přes aktuální počasí a sněhové zpravodajství včetně krátkodobé předpovědi počasí, až po přehled služeb, které středisko poskytuje svým návštěvníkům.

Portál byl naprogramovaný v jazyce PHP v kombinaci s databázovým systémem MySQL. Nejprve byl vytvořen grafický návrh celého portálu a poté převeden do kaskádových stylů CSS. Po úspěšném funkčním a validním zhotovení statické XHTML stránky s použitím kaskádových stylů a nutným přihlédnutím na správné zobrazení ve všech internetových prohlížečích, byla první část pro tvorbu portálu připravena.

Důvod, proč byl použit čistý jazyk PHP a nebyl použit žádný z existujících redakčních systémů, případně některý z opensource projektů pro portály je požadavek na originalitu portálu a specifické rozložení webu, kterému nevyhovoval žádný opensource projektů.

Zobrazení dat o počasí a sněhovém zpravodajství je na portálu zajištěno pomocí grafů, které jsou generovány automaticky prostřednictvím GD knihovny v PHP. Poskytování informací o bodech zájmu je v současné době v počáteční fázi testování. Databáze bodů zájmů je proto pouze demonstrativní pro odladění chyb svázaných s poskytováním informací o nejbližších jiných bodech zájmu. Tato databáze se bude zaplňovat postupně i po spuštění portálu pro veřejnost. Stejně jako po spuštění portálu pro veřejnost se bude pracovat na databázi ubytovacích zařízení s funkcí zobrazování náhodných ubytovacích objektů na stránkách portálu.

8. Seznam literatury

1. ULLMAN, Larry. *PHP a MySQL, názorný průvodce tvorbou dynamických WWW stránek, první vydání*. Computer Press, Brno 2004. Str. 534. ISBN: 80-251-0063-4
2. TANSLEY, David. *PHP a MySQL – Vytváříme Dynamické Webové stránky.* Softpress s.r.o, Praha 2003. Str. 480. ISBN: 80-86497-40-2
3. WELLING, Luke, THOMSONOVÁ, Laura. *PHP a MySQL – rozvoj webových aplikací, Druhé vydání.* Softpress s.r.o, Praha 2004. Str. 910. ISBN: 80-86497-60-7
4. LACKO, Luboslav. *PHP a MySQL Hotová řešení.* CP Books, Brno 2005. Str. 299. ISBN: 80-251-0397-8
5. CASTAGNETTO, Jesus a spol. *PHP Programujeme profesionálně, První vydání.* Computer Press, Brno 2001. Str. 656. ISBN: 80-7226-310-2
6. BRÁZA, Jiří. *PHP 4 – učebnice základů jazyka, První vydání.* Grada Publishing, Praha 2002. Str. 224. ISBN: 80-247-0442-0
7. MACH, Jakub. *PHP pro úplné začátečníky, Druhé vydání.* Computer Press, Brno 2003. Str. 167. ISBN: 80-7226-834-1
8. GUTMANS, Andi a spol. *Mistrovství v PHP 5, První vydání.* CP Books, Brno 2005. Str. 655. ISBN: 80-251-0799-X

9. Přílohy

Příloha 1 – seznam zdrojových kódů

Příloha 2 – seznam obrázků

Příloha 3 – Funkce získávání počasí přes Google API

```
<?php
/**
 * GoogleWeather
 *
 * Predpoved pocasia (google)
 *
 * @author   stenley <stenley@webdev.sk>
 * @version   1.1
 * @license   http://opensource.org/licenses/gpl-license.php GNU Public License
 */

class GoogleWeather {
  var $lang = "sk";
  var $country = "Slovakia";
  var $charset = "utf-8";

  function getWeatherObj($city) {
    $city = urlencode($city);
    $data = @file_get_contents("http://www.google.com/ig/api?weather=".$city.",".$this-
>country."&hl=".$this->lang);
    if(!$data) {
      return false;
    }
    $data = iconv("ISO-8859-2",$this->charset,$data);
    return new SimpleXMLElement($data);
  }

  function getWeatherInfo($city) {
    $weather = array();
    $obj = $this->getWeatherObj($city);

    if(is_object($obj)) {
      $data = $obj->xpath("/xml_api_reply/weather/current_conditions");
      if(!empty($data)) {
        $weather[] = array(
          "condition"     => $this->getAttr($data[0]->xpath("condition")),
          "temp_f"        => $this->getAttr($data[0]->xpath("temp_f")),
          "temp_c"        => $this->getAttr($data[0]->xpath("temp_c")),
          "humidity"      => $this->getAttr($data[0]->xpath("humidity")),
          "icon"        => $this->getAttr($data[0]->xpath("icon")),
          "wind_condition"   => $this->getAttr($data[0]->xpath("wind_condition"))
        );
      }
```

```
      $data = $obj->xpath("/xml_api_reply/weather/forecast_conditions");
      if(!empty($data)) {
        foreach($data as $value) {
          $weather[] = array(
            "day_of_week"   => $this->getAttr($value->xpath("day_of_week")),
            "low"      => $this->getAttr($value->xpath("low")),
            "high"      => $this->getAttr($value->xpath("high")),
            "icon"      => $this->getAttr($value->xpath("icon")),
            "condition"   => $this->getAttr($value->xpath("condition"))
          );
        }
      }
    }

    return $weather;
  }

  function getAttr($obj) {
    $attr = $obj[0]->attributes();
    return (string)$attr['data'];
  }
}
?>
```

Příloha 4 – Funkce získávání sněhového zpravodajství z XML souboru společnosti Sitour

```
<?php

header('Content-Type: text/html; charset=utf-8');
error_reporting(E_ALL);

class Strediska
{
 //zde uložená instance XMLReaderu
 private $reader;

 //sem se ukládají veškeré přečtené informace
 private $strediska = Array();

 /*
  * Vyhledá středisko podle hodnoty ve značce <ACRONYM>.
  * Při úspěchu vrací pole se všemi informacemi o středisku,
  * při něúspěchu vrací string s hláškou o nenalezení.
  */
 public function najdiStredisko($acronym)
 {
  if (isset($this->strediska[$acronym]))
    return $this->strediska[$acronym];
  else
    return 'Středisko nenalezeno.';
 }

 /*
  * Vrátí celé pole se všemi informacemi o střediscích
  */
 public function vsechnyStrediska()
 {
  return $this->strediska;
 }

 /*
  * V konstruktoru se zřídí instance XMLReaderu
  * a zpracuje se celý XML dokument.
  */
 public function __construct($file)
 {
  $this->reader = new XMLReader();

  if (!$this->reader->open($file))
  {
   echo 'Nepodařilo se otevřít XML soubor!';
   return;
  }

  $this->parseFile();
```

```
  $this->reader->close();
}

private function parseFile()
{
  //jen pro zkrácení
  $r = &$this->reader;

  //čteme soubor značku po značce
  while ($r->read())
  {
    //pokud narazíme na nové středisko
    if ($r->nodeType == XMLReader::ELEMENT && $r->name == 'STREDISKO')
      $this->addStredisko();
  }
}

private function addStredisko()
{
  //jen pro zkrácení
  $r = &$this->reader;

  //pomocné proměnné
  $nazev = htmlspecialchars($r->getAttribute('NAZEV'));
  $acronym = null;

  //dál procházíme dokument
  while ($r->read())
  {
    //pokud narazíme na konec informací o právě zpracovávaném středisku
    if ($r->nodeType == XMLReader::END_ELEMENT && $r->name == 'STREDISKO')
      break;

    // jinak zpracováváme každou značku (nezajímá nás, co je to za
    // značku -> pružně ukládáme veškeré info o středisku, které se nám dostane pod ruku
    if ($r->nodeType == XMLReader::ELEMENT)
    {
      // značka akronym bude sloužit jako index v poli $this->strediska[acronym],
      // pro každé středisko (předpokládá se, že hodnota acronymu je jedinečná)
      if ($r->name == 'ACRONYM')
      {
        $r->read();
        $acronym = trim($r->value);
        $this->strediska[$acronym] = Array();
        $this->strediska[$acronym]['jmeno'] = $nazev;
      }
      else if ($r->name == 'SJEZDOVKY')
      {
        $this->strediska[$acronym]['sjezdovky'] = Array();
        while ($r->read())
        {
          if ($r->nodeType == XMLReader::ELEMENT && $r->name == 'SJEZD')
```

```
        {
          $r->read();
          $this->strediska[$acronym]['SJEZDOVKY'][] = $r->value;
        }
        else if ($r->nodeType == XMLReader::END_ELEMENT && $r->name == 'SJEZDOVKY')
          break;
       }
      }
      else if ($r->name == 'AKCE')
      {
       $this->akce($acronym, $r);
      }
      else
      {
       //ukládáme každou informaci
       $index = $r->name;
       $r->read();
       $this->strediska[$acronym][$index] = htmlspecialchars(trim($r->value));
      }
     }
    }
   }

   private function akce($acronym, $r)
   {

    while ($r->read())
    {
     if ($r->nodeType == XMLReader::ELEMENT && $r->name == 'AKCE_PART')
     {
      $akce = Array();
      while($r->read())
      {
       $index = null;
       if ($r->nodeType == XMLReader::END_ELEMENT && $r->name == 'AKCE_PART')
         break;
       else if ($r->nodeType == XMLReader::ELEMENT)
       {
         $index = $r->name;
         $r->read();
         $akce[$index] = htmlspecialchars(trim($r->value));
       }
      }
      $this->strediska[$acronym]['AKCE'][] = $akce;
     }
     else if ($r->nodeType == XMLReader::END_ELEMENT && $r->name == 'AKCE')
       break;
    }
   }

}
```