Univerzita Pardubice
Fakulta elektrotechniky a informatiky

WWW aplikace s využitím relační databáze pro správu sportovního centra

Michal Nosil

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:	**Michal NOSIL**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**WWW aplikace s využitím relační databáze pro správu sportovního centra**

Zásady pro vypracování:

Úkolem bakalářské práce je vytvořit www prezentaci sportovního centra a navrhnout a vytvořit relační databázi pro jeho správu. Cíl teoretické části: - porovnání prezentací tří až pěti prestižních sportovních center a jejich zhodnocení - vyřešení zabezpečení dat. Aplikace musí minimálně umožnit: - Registrace uživatelů - Správa sportovních kurtů (možnost rezervace pro registrované uživatele a zaměstnance) - Správu pořádaných turnajů - Záznamy o půjčeném sportovním zařízení - Přístup dle práv(administrátor, zaměstnanci centra, registrovaní uživatelé s možností správy turnajů, registrovaní uživatelé)

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

Castagnetto, J. a kol. Programujeme PHP profesionálně. Computer Press, 2004. Kout, P. Praktický JavaScript. Zoner Press, 2004. Oppel, A. Databáze bez předchozích znalostí. Computer Press, 2006. Ullman, L. PHP a MySQL.Computer Press, 2004.

Vedoucí bakalářské práce: **RNDr. Iva Rulicová**
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 20. 8. 2009

Michal Nosil

Souhrn

Tato práce se zabývá problematikou návrhu a tvorby rezervačního systému pro sportovní centrum. Rezervační systém byl naprogramován v jazyce PHP, pro databázové schéma byla využita relační databáze MySQL. Grafický návrh byl vytvořen pomocí kaskádových stylů.

Klíčová slova

rezervace, systém, databáze, MySQL, PHP, data

Tittle

World Wide Web application with usage of relational database for sporting centre booking.

Abstract

This thesis is aimed at the issue of designing and creating of a booking system for a sporting center. The booking system was programmed in PHP language and for database, the relational database MySQL was used. The graphic layout was made using cascading style sheets.

Keywords

reservations, system, database, MySQL, PHP, data

Obsah

Seznam obrázků:

Seznam tabulek:

Seznam použitých odborných výrazů

Apache	HTTP Server, je softwarový webový server
CSS	Cascading Style Sheets, kaskádové styly
HTML	Hypertext Markup Language, jazyk pro vytváření statických internetových stránek
HTTP	HyperText Transfer Protokol, internetový protokol
IP adresa	jednoznačná identifikace síťového rozhraní
JavaScript	skriptovací jazyk pro tvorbu dynamického webu
PHP	Hypertext Preprocessor, skriptovací jazyk pro tvorbu dynamických
PHP	internetových stránek
Transakce	skupina příkazů, které převedou databázi z jednoho konzistentního stavu do druhého
URL	Uniform Resource Locator, jednoznačné určení zdroje na Internetu

1. Úvod

Cílem mé bakalářské práce je vytvořit rezervační systém pro sportovní centrum. Systém je navržen tak, aby potenciální zákazník mohl spravovat celé své sportovní centrum elektronicky.

Práce je rozdělena na praktickou a teoretickou část. V praktické části je popsána využitá technologie, návrh databáze a popis jednotlivých stránek. Teoretická část se zabývá zabezpečením dat, popisem možných útoků a způsoby obrany. Dále se teoretická část orientuje na porovnání internetových stránek pěti sportovních center.

2. Zabezpečení dat

2.1. Soubory

Použití souboru jako úložiště dat není vhodné, a to z důvodu nedostatečného zabezpečení. Soubory jsou často málo zabezpečeny, v horších případech nejsou zabezpečeny vůbec. Nejhorší variantou je, když se na serverovém prostoru nachází soubor s uloženými uživatelskými jmény a hesly. V tomto případě stačí zadat útočníkovi prohlížeče adresu s příslušným souborem a dostane se mu výpis všech uživatelů, kteří mají přístup do systému. Tímto případem jsou například soubory *.txt. Použití těchto souborů je dnes spíše historickou záležitostí.

Ještě mnohem nebezpečnější jsou soubory s daty pro vytváření tabulek, nebo dokonce s daty pro naplnění databázových tabulek, např. pro testování aplikace při tvorbě. Pokud by se útočník k takovým datům dostal, dozví se v lepším případě schéma celé databáze, v horším případě i uživatelská jména a hesla. Tyto soubory lze vyhledat např. pomocí internetového vyhledávače Google. Stačí zadat do textu vyhledávače filetype:sql site.cz.

Ukázka sql souboru se strukturou databáze:

```
#MySQL DUMP
#------------------------------
# Jméno databáze: gph
#------------------------------
#
STRUKTURA tabulky: bonifikace
DROP TABLE IF EXISTS `bonifikace`;
CREATE TABLE `bonifikace` (
  `b_id` int(11) NOT NULL auto_increment,
  `rok` int(11) NOT NULL default '0',
  `pivo` int(11) NOT NULL default '0',
  `rum` int(11) NOT NULL default '0',
  `sam_nes` int(11) NOT NULL default '0',
  `sam_vet` int(11) NOT NULL default '0',
  `sam_vyr` int(11) NOT NULL default '0',
  `savle` int(11) NOT NULL default '0',
  PRIMARY KEY  (`b_id`),
  KEY `b_id` (`b_id`)
) TYPE=MyISAM;
```

Zdroj: http://gph.iglu.cz/zaloha_z_10_06_2003-gph.sql

Ukázka vyhledaného souboru, kde jsou uložena data s uživatelskými účty:

```
insert into uzivatele(id,username,password,id_uzivatele_skupiny,vlozeno)
values (1,'administrator',md5('admin'),1,now());
```

Zdroj: http://www.stud.fit.vutbr.cz/~xmlich02/data/stega-creator.sql

Nejlepší obranou je soubory na webový server vůbec neukládat, pokud je to však nevyhnutelné, musí být soubor zabezpečen, např. pomocí hesla nebo IP adresou. Tato ochrana se nastavuje přes konfiguraci serveru Apache v souborech .htaccess nebo .htpasswd. Soubor .htaccess je umístěný někde na webovém prostoru. Tento soubor je serverem zpracováván při každém pokusu o načtení souboru ze složky, v níž je umístěn. Níže je uvedeno několik příkazů, které může .htaccess obsahovat:

- **Options –Indexes -** Zakazuje zobrazení obsahu složky, ve které není stránka index.
- **Deny from** - Zakazuje přístup do složky.

- **Deny from all -** Přístup zamítnut pro všechny.

- **Allow - J**e opakem příkazu Deny. Povoluje tedy přístup ke složce.
- **SetEvnIf -** Je podmíněný příkaz, který je podobný všem ostatním podmínkovým příkazům v různých jazycích.

Soubor .htpasswd obsahuje hesla, která se budou používat. Pokud uživatel zadá heslo jiné, než je uvedeno v tomto souboru, je zobrazen chybový kód.

Samotný soubor .htpasswd obsahuje pouze uživatelská jména a hesla, proto je dále nutné nastavit v .htaccess cestu k souboru s uživatelskými hesly (AuthUserFile), název oblasti, kterou chcete zabezpečit (AuthName), typ autorizace (AuthType), specifikaci omezení přístupu (require) [17].

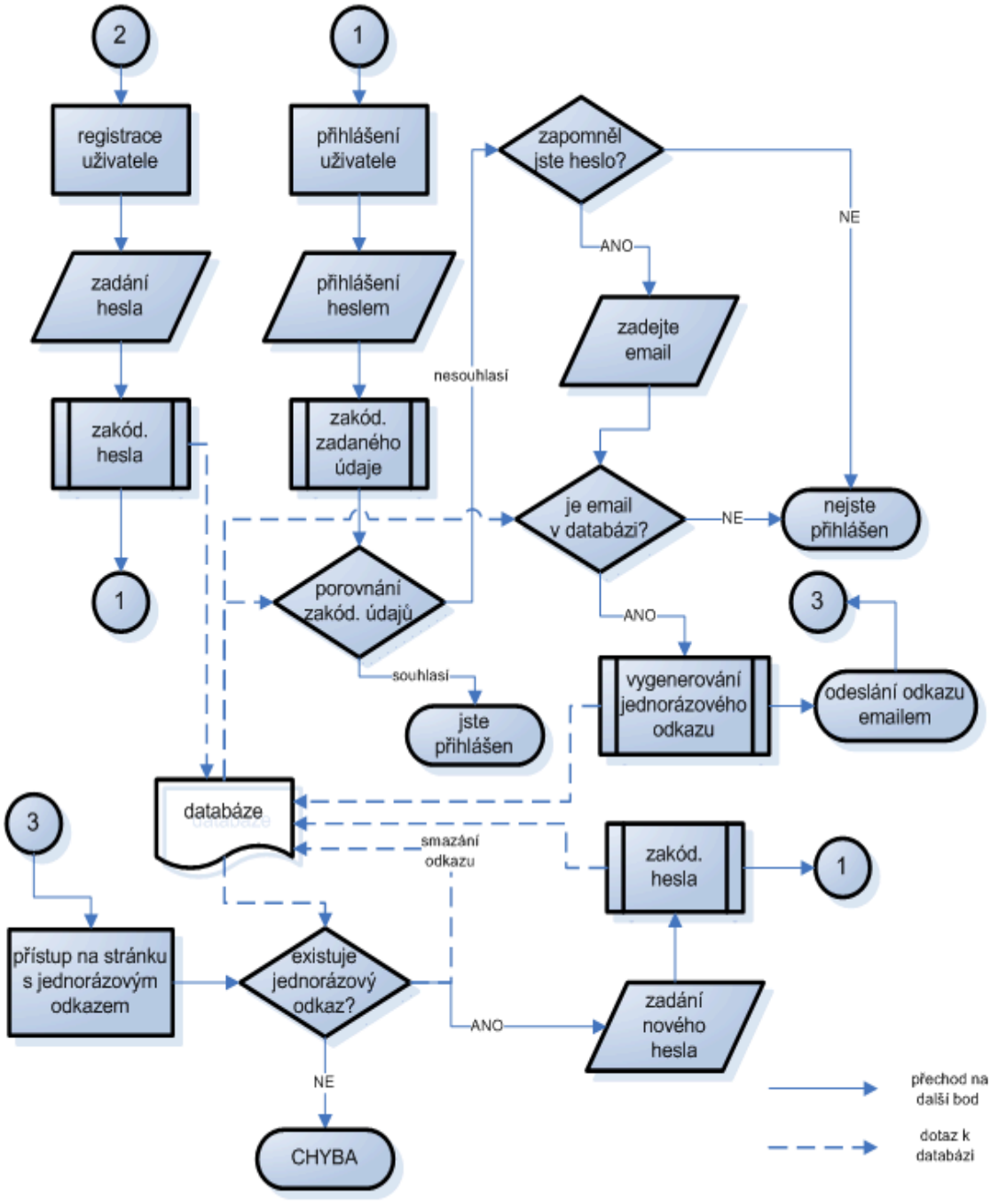
2.2. Hesla

Ověření uživatelského jména a hesla patří k nejčastějším způsobům autentizace webové aplikace. Základní chybou je ukládat hesla bez jakéhokoliv kódování tak, jak ho uživatel zadal. Protože každý, kdo se dostane k databázi, získá automaticky seznam hesel všech uživatelů. Zjištění hesla nemusí znamenat nebezpečí pouze pro aplikaci, ale může způsobit problémy i samotnému uživateli. Pokud má jedno uživatelské jméno a heslo nastavené na email, umožní přístup do internetového bankovnictví. Proto by se každá aplikace měla před uložením do databáze hesla hashovat.

2.2.1. Hashování hesel

Pomocí hashovacích funkcí dochází k lepšímu zabezpečení hesel. Útočník ani administrátor heslo uživatele v běžné textové podobě neznají. Tento nedostatek lze částečně vyřešit následujícím algoritmem [19].

Obrázek 1: Komplexní systém zabezpečení přístupových údajů



Zdroj: http://access.feld.cvut.cz/view.php?cisloclanku=2007080002

Vstupem do hashovací funkce je jakýkoliv řetězec jakékoliv délky, výstupem je řetězec fixní délky, který je nazýván hash (otisk). Mezi nejznámější algoritmy patří: MD4, MD5, SHA-1, SHA-256/224, SHA-512/384).

Ukázka otisku hash funkce MD5:

Původní heslo:
Toto je moje heslo
MD5 otisk:
07f793f8a0518ec9095b41e2f87f7aa9

Hashovací funkce vrací pokaždé stejný otisk, ale z vytvořeného otisku nezískáte původní heslo.

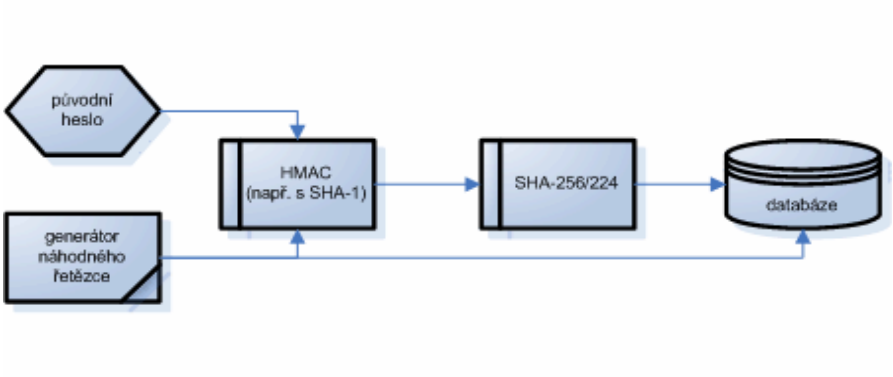
2.2.2. Útoky na zahashovaná hesla

Slovníkový útok je metoda, při které je k zjištění hesla použit hrubou silou způsob pokus/omyl ze seznamu známých slov. Tato metoda může být rozšířena o doplnění číslic na konec slova. Slovníkový útok postupně zkouší všechna slova z běžného jazyka doplněné o číslice na konci. Tyto útoky uspějí v případech, kdy je heslo krátké (7 nebo méně znaků) [15].

Rainbow tables jsou tabulky dvojic, v nichž je uložen řetězec a jeho otisk, tyto tabulky slouží jako online databáze otisků, která v současné době obsahuje miliony záznamů. Princip útoku spočívá v hledání hesel v těchto tabulkách. Pomocí takovéto metody, lze prolomit hashování pomocí funkce MD5 [12].

Dokonalejší metoda ochrany hesla je použití metody „solení“. Princip spočívá v přidávání nějakého řetězce ke vstupu. Výsledkem je, že pro stejná hesla se vytvoří jiný otisk. Řešením je náhodně vygenerovaný řetězec, který je třeba ukládat do samostatného sloupce v databázi. Je tedy proto vhodné, vyvarovat se spojení obyčejných řetězců. Řešením je použít funkci HMAC, která zjišťuje, zda nebyla informace cestou změněna [18].

Obrázek 2. Algoritmus „Solení“



Zdroj: http://access.feld.cvut.cz/view.php?cisloclanku=2007080002

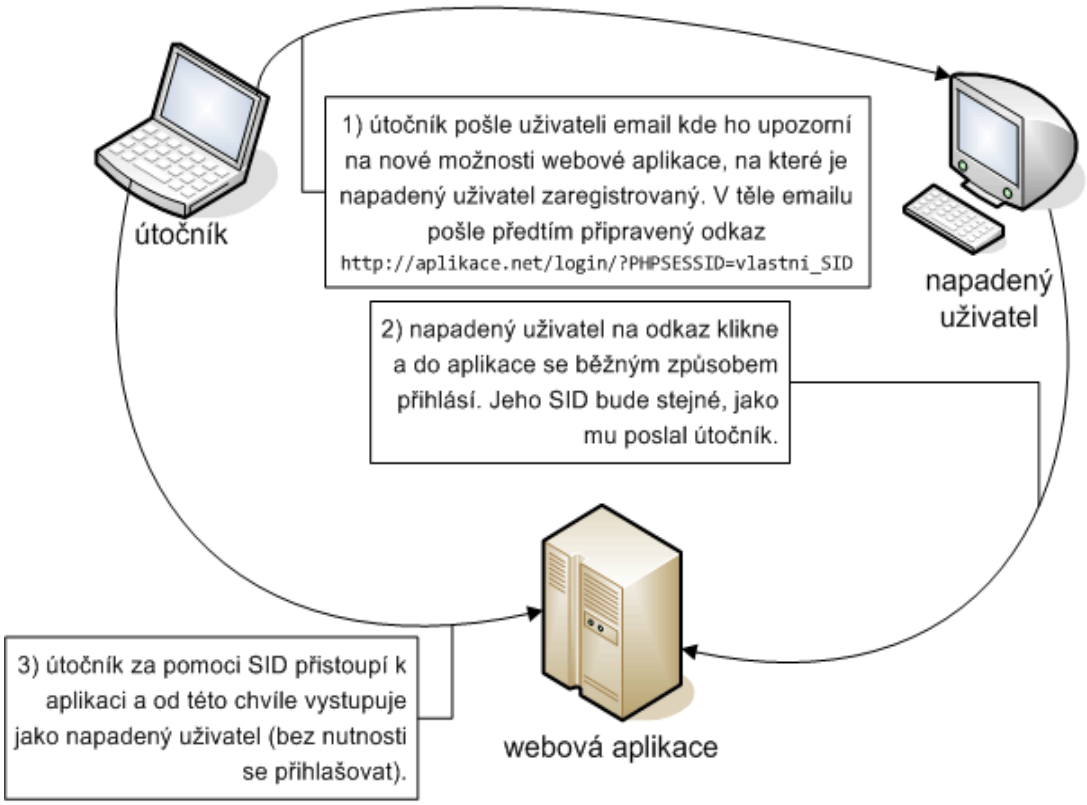
2.3. Session

Uživatelská relace nebo-li session se používá k uložení informací o uživateli při pohybu mezi stránkami. Pokud webová aplikace využívá k přihlášení session, tak se hned po přihlášení vytvoří jednoznačné číslo, které se nazývá SID (session ID). Pomocí SID jsou v aplikaci nastavena uživatelská práva. Např. databáze zjišťuje, zda uživatel s vytvořeným SID má práva pro požadovanou akci.

2.3.1. Útok session fixation

Útočník nastaví SID na předgenerovanou hodnotu a za pomoci např. podstrčeného odkazu přesvědčuje uživatele k zalogování do systému. Jakmile se uživatel přihlásí, použije se session ID poskytnuté útočníkem. Útočník tímto získává přístup do aplikace s ukradeným SID [19].

Obrázek 3: Princip session fixation



Zdroj: http://access.feld.cvut.cz/view.php?cisloclanku=2007080003

Možností ochrany proti útoku session fixation je za pomocí funkce session_regenerate_id, která je zavolána před přihlášením uživatele do systém. Postrčené SID systém zahodí a útočník nové nezná. Funkce session_regenerate_id se nalézá ve verzi PHP serveru 4.3.2 a vyšší.

2.3.2. Útok Sidejacking Session

Principem tohoto útoku je odposlechnutí SID z paketů putujících na síti. V současné době existuje mnoho programů k odchytávání a následnému filtrování paketů např. Wireshark, tcpdump. Sidejacking, které se užívají spíše u bezdrátových sítí, protože zde je možnost odposlechu paketů vyšší. Možnosti rozsahu útoku jsou různé podle typu zabezpečení bezdrátové sítě, respektive podle šifrovacího protokolu. Na klasickém ‚drátovém' připojení je útok možný, pokud je v infrastruktuře sítě hub, protože přeposílá všechnu komunikaci na všechny uzly,

nebo pokud se útočníkovi podaří připojit svůj počítač před router, popřípadě nad ním nějakým jiným způsobem získat kontrolu.

Ochrana před tímto útokem je spíše na správcích sítě, neboť obyčejný vývojář se obvykle nestará o komunikaci na celé síťové infrastruktuře. Jedinou možností ochrany pro vývojáře aplikace je použít pro komunikaci šifrovaný kanál. U webových aplikací je to TLS a jeho předchůdce SSL [16].

2.3.3. Útok Cross-site scripting

Metoda narušení internetových stránek, kdy se útočník snaží využít chyby v bezpečnosti aplikace a vkládá do stránek svůj vlastní skript, který internetový prohlížeč interpretuje jako HTML kód. Pomocí útoku XSS (Cross-site scripting) se útočník může dostat k přihlašovacím údajům, když stránku s přihlášením jednoduše přesměruje na svoji stránku, která je vizuálně identická. Dále se útočník může dostat k uživatelovým cookies a ukrást jeho SID. Cross-site scripting lze rozdělit na Non-persistent, Persistent a DOM based [1].

Non-persistent je jeden z nejběžnějších útoků. Princip spočívá v úpravě URL adresy, která se následně interpretuje do stránky.

Příklad: Útoku Non-persistent

Zdrojový kód:

```
<?php echo $_GET['nadpis']; ?>
```

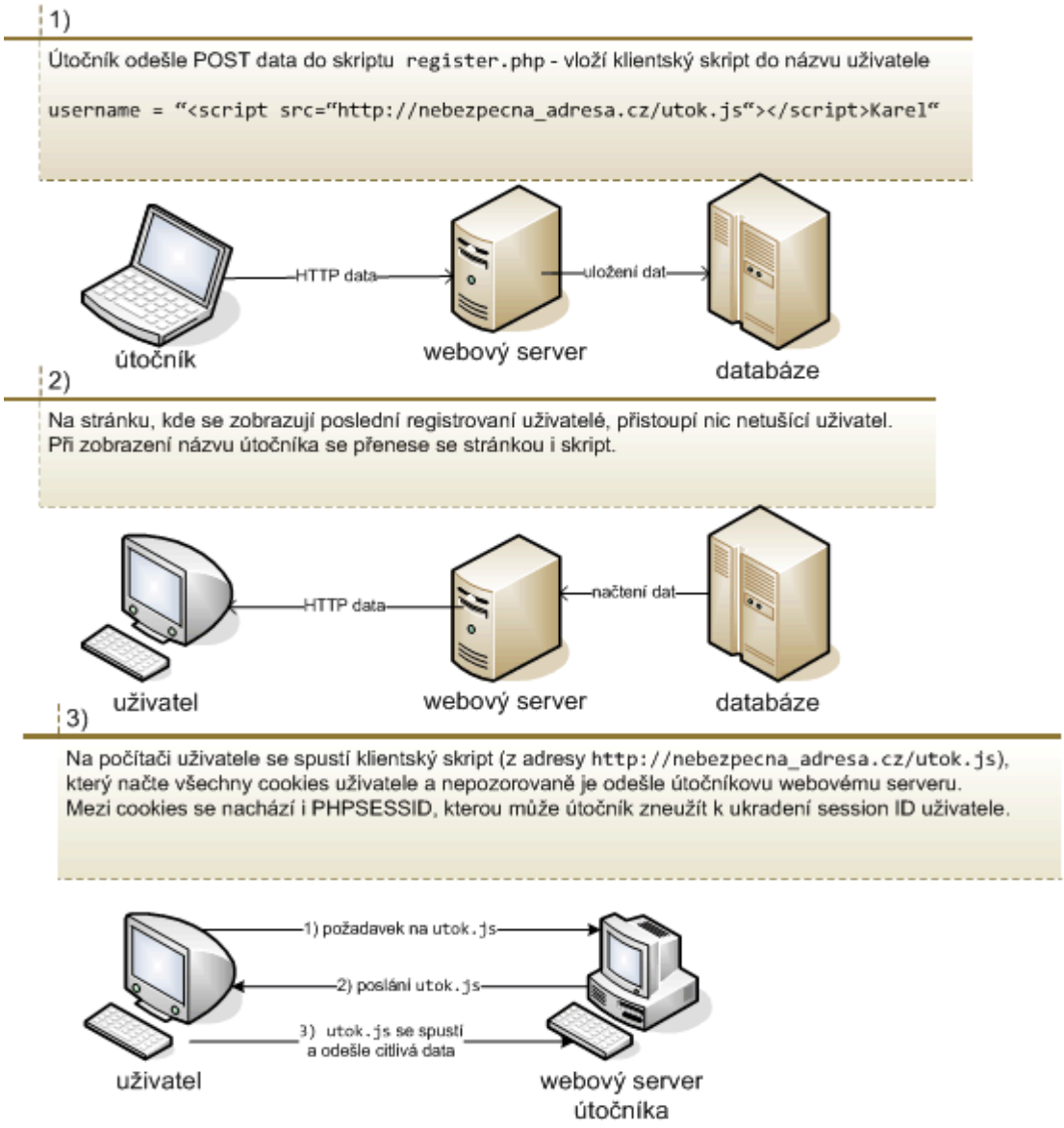
Podstrčená URL adresa:

```
http://URL/stranka.php?nadpis=cokoliv<script>alert('Toto je úspěšný XSS útok.');</script>
```

zdroj: http://cs.wikipedia.org/wiki/Cross-site_scripting

Persistent je útok, který lze aplikovat pouze v případě, kdy je obsah stránky generován z databáze. Útočník vloží Javascript třeba do komentáře, který se uloží do databáze. Data z databáze se zobrazí pokaždé, když si uživatel tuto stránku zobrazí [17].

Obrázek 4: Příklad perzistentního útoku



Zdroj: http://access.feld.cvut.cz/view.php?nazevclanku=zabezpeceni-webovych-aplikaci-i-klientske-skriptovaci-jazyky&cisloclanku=2007090001

Útok DOM based je hodně podobný okamžitému útoku. Rozdíl je v tom, že tento útok zneužívá existující klientský (tzn. lokální) skript. Lokální skripty mají obvykle rozšířená oprávnění, jako je například přístup k souborům na disku. Útočník se tak může dostat k citlivým datům aplikace [17].

Nejúčinnější obranou proti těmto útokům je vypnutí Javascriptu, ale tím může dojít k nežádoucí změně grafické úpravy některých internetových stránek.

Dalším způsobem obrany je použití funkce htmlspecialchars**,** která nahradí citlivé znaky, jako je např. „<“ za „&lt‘. Funkce htmlspecialchars zabrání interpretaci podstrčených značek.

Možnost obrany je také pomocí funkce strip tag, která umožňuje odebrat z zadaného řetězce html tagy.

Pokud ale potřebujete některé tagy používat, musíte si sestavit regulární výraz, který odstraní nežádoucí tagy. Tato metoda není příliš bezpečná, protože sestavení takového výrazu je dost obtížné.

2.3.4. Správná tvorba SID

Pro zamezení útoku hrubou silou, tedy možnosti odhadnutí SID, je dobré při každé návštěvě použít nově náhodně vygenerovanou SID. V současné době se již nedoporučuje generování pomocí MD5, a to z důvodu, že je to zastaralá metoda a lze ji odhadnout ve velmi krátké době. Důležité je zamezit možnosti, že dva různí uživatelé budou mít stejnou SID. Řešením je, přidat do algoritmu pro tvorbu uživatelské session uživatelovu IP adresu.

Dále není vhodné tvořit SID ze smysluplných údajů, jako je heslo, uživatelské jméno a další citlivé informace. SID by mělo obsahovat co nejvíce znaků. Čím je SID delší, tím více možností musí útočník vyzkoušet. Další pravidlo pro tvorbu SID je omezení časové platnosti. Při omezení platnosti se velmi snižuje možnost útoku hrubou silou. Obvykle se nastavuje doba vypršení někde mezi 5 a 30 minutami. Pokud není session aktivní, je vhodné ji průběžně obměňovat.

2.4. Databáze

Databáze se v dnešní době objevuje téměř všude. Výjimku netvoří ani webové aplikace. Tyto aplikace komunikují s uživatelem přes webové rozhraní téměř vždy. Databáze je tedy jedno z nejcitlivějších míst celé aplikace. Pokud se útočníkovi podaří databázi nabourat, může získat přístup nejen k uživatelským účtům, ale i citlivým datům, jako jsou rodná čísla, čísla účtu, čísla občanského průkazu atd. Navíc tato data může editovat, mazat a také přidávat záznamy. Útočníkovi se dokonce může podařit smazat i celou databázi.

2.4.1. SQL Injection

Cílem útoku SQL Injection je nalézt místo, kde aplikace posílá SQL dotaz do databáze např. přihlašovací formulář. Princip útoku je ve vložení vlastního dotazu

do vstupních dat, jako jsou např. formuláře (POST/GET) nebo parametry URL. V následujících příkladech jsou popsány různé možnosti útoku [5].

Příklad SQL injection pro vypsání tabulky s uživateli:

Původní dotaz v skriptu:

SELECT * FROM uzivatele WHERE id = '$id'

Výsledný dotazu:

SELECT * FROM uzivatele WHERE id =1 OR 1=1--

V původním dotazu je v podmínce číslo uživatele. Toto číslo je obvykle lehce zjistitelné z výpisu o uživateli. Místo sloupce id lze použít jakýkoliv jiný sloupec. Velmi jednoduše se dá zjistit přezdívka (nick) uživatele např. z internetové diskuze. Dotaz, který je používán, musí být syntakticky správný, proto je tedy nutné nejprve ukončit původní podmínku a dále pak pokračovat svým kódem. Konec dotazu z příkladu je za řetězcem `1`, dále následuje vždy platná podmínka 1=1. V podmínce je možné použít prakticky cokoliv, pokud výsledek bude mít hodnotu „pravda“. Je-li použit místo číselných hodnot textový řetězec, je nutné ho uzavřít mezi znaky apostrofů např. `a` = `a`. Konec kódu je zakončen “--“, což je v jazyce SQL komentář. Pomocí „--“ je ošetřen případ, kdy originální dotaz pokračuje např. řazením (order by). Horší případ je použití příkazu delete, kdy útočník může smazat data z tabulky [2].

Příklad: Použití SQL injection pro smazání celé tabulky:

Původní dotaz v skriptu :

SELECT * FROM uzivatele WHERE id = '$id'

Výsledný dotaz:

SELECT * FROM uzivatele WHERE id = 1 OR 1=1; DROP TABLE uzivatele

Některé SQL servery podporují zpracování více dotazů. Tuto vlastnost předpokládá i výše uvedený příklad. První dotaz již byl vysvětlen. Dále bude pokračováno od znaku středník. Tento znak slouží pro ukončení SQL dotazu, lze ho tedy použít jako oddělovač mezi dotazy. Útočník jednoduše za pomoci „DROP TABLE“ smaže celou tabulku „uzivatele“.

Příklad získání uživatelských jmen a hesel za pomoci UNION:

Původní dotaz:
SELECT nick, email FROM uziv WHERE id = '$id' LIMIT 1
Výsledný dotaz:
SELECT nick, email FROM uziv WHERE id = 1 UNION SELECT heslo AS nick, nick AS email FROM uzivatele --' LIMIT 1

První část dotazu byla již byla vysvětlena. Následuje vysvětlení dotazu od slova „UNION“. Použitím „UNION“ dojde ke sjednocení výsledků dvou dotazů. Druhá část dotazu vypíše uživatelská jména a hesla. Dotaz je však omezen pouze na výpis jednoho záznamu. Díky tomu je možnost útoku pouze ve skriptech, kde se aplikuje cyklické vypisování [4].

2.4.2. Obrana proti SQL injection

Principů ochrany je mnoho a dělí se podle použití databázového serveru.
Mezi obecné patří:

- Nastavení délky vstupních dat u formulářů.
- Testování vstupních dat za pomoci regulárních výrazů, např. uživatelské jméno nebude obsahovat znaky: „ =, <, >, --, /, ?, ‘,“. Dále je možné testovat, zda vstupní data neobsahují slova: „select, update, drop, delete“.
- Testování pomocí funkce, která analyzuje, zda číselné vstupy neobsahují textové řetězce např. v PHP is_integer.
- Nahrazení znaku ['] znakem [''] u textových vstupů, což útočníkovi znemožní ukončit SQL dotaz.
- Zamezení výpisu chybových hlášek u databázového serveru.

V PHP lze ošetřovat SQL injection pomocí funkcí mysql_real_escape_string, addslashes nebo direktivou magic_quotes_gpc**.** Tyto funkce, včetně uvedené direktivy, vloží zpětné lomítko před specifické znaky, čímž se do databáze uloží jako apostrof a není interpretován jako znak uvozující konec řetězce [3]**.**

Přiklad použití funkce mysql_real_escape_string pro ošetření SQL injection:

```
$name_bad = "' OR 1'";
$name_bad = mysql_real_escape_string($name_bad);
$query_bad = "SELECT * FROM customers WHERE username = '$name_bad'";
výpis proměnné „query_bad“ :
SELECT * FROM customers WHERE username = '\' OR 1\'
```

Zdroj: http://www.tizag.com/mysqlTutorial/mysql-php-sql-injection.php

Funkce addslashes je velice podobná funkci mysql_real_escape_string, rozdíl je v pouze v podpoře znakových sad. Ve státech, kde se používá kódování UTF – 8, je plně dostačující [7].

Direktiva magic_quotes_qpc je po instalaci standartně vypnutá a od PHP6 je již zrušena. Pozor na případ, kdy je direktiva zapnuta a vstupní řetězec je ošetřován funkcemi addslashes nebo mysql_real_escape_string, protože tehdy může dojít k vložení dvou znaků zpětného lomítka za sebou [8].

3. Porovnání prestižních sportovních center

Vybrat 5 nejprestižnějších sportovních center není při množství, které je dostupné na internetu, jednoduchá věc. Hlavním kritériem mého výběru byla odlišnost (např. netradiční design, počet kurtů) internetových stránek sportovních center.

3.1. Hcentrum

Grafické zpracování stránek Hcentra je průměrné. Mezi nedostatky patří zobrazení ceníku s jednorázovými vstupy, kde se tabulka při nastaveném rozlišení 1200 x 800 pixelů nezobrazuje celá. Ve spodní části je sice zobrazen posuvník, ale to vede k nepřehlednosti cen. Další nedostatek je v odkazu na 3D galerii. Zde se nalézá velký počet voleb, ale tyto volby nejsou funkční. Stránky jsou velmi často aktualizovány. I přes tyto nedostatky jsou stránky velmi přehledné, přehlednost je zčásti narušena umístěním odkazu s mapou stránek, který se nachází v patičce. Stránky, ale neprošly testem validity.

Přihlášení do online rezervací mají pouze členové klubu, čímž je zajištěno, že neexistuje možnost uskutečnit útok na registrační formulář. Bez přihlášení není možné dostat se k zobrazení vytížení sportovišť a relaxačních zařízení. Velmi příjemná je dvoujazyčná verze těchto stránek. Pro uživatele může být matoucí způsob pohybu mezi měsíci v kalendáři. Na těchto stránkách nebylo možno otestovat, zda lze vložit rezervaci s neaktuálním datem, protože do této sekce mají přístup pouze registrovaní uživatelé [20].

3.2. Sportcentrum Cibulka

Tyto stránky vyčnívají na první pohled svým netradičním designem, který je na úvodní stránce velice povedený. V případě, že je kliknuto na některý z odkazů, dojde k značnému zmenšení hlavního menu a text s odkazy se stane skoro nečitelný. Pokud má uživatel těchto stránek pomalejší internetové připojení, dochází k chvilkovému nežádoucímu rozvržení celé stánky. Důvěru nevzbuzuje ani prázdná stránka s obchodem.Některé fotky např. „4. turnaj SPL 2004 - 4. 12. 2004“ jsou v náhledu v nežádoucí kvalitě. Po detailním zobrazení fotek je vše naprosto v pořádku. Struktura stránek není úplně vyhovující, protože informace o otevírací době jsou pod odkazem na „Ceník“. Příjemným bonusem je ale zobrazení mapy, kde se centrum nalézá. Test validity objevil 93 chyb, což je mnoho.

Do rezervací se bez přihlašovacích údajů, o které si uživatel může zažádat, není možné dostat. To je velký nedostatek, protože uživatel musí volat do centra a nejprve zjišťovat, jaký termín je volný [21].

3.3. Sportcentrum Ivanovice

Grafické rozhraní těchto stránek je přímočaré. Důležité informace pro zákazníka (otevírací doba, telefonní číslo) jsou umístěny tak, aby byly viditelné pořád, bez ohledu na to, kde se návštěvník nalézá. Povedená je volba virtuální prohlídka, která umožňuje seznámit se všemi sportovišti. Odkazy jsou velmi přehledné a jsou strukturovány s vhodnou velikostí textu. Problém těchto internetových stánek je v nefunkčních odkazech např.
„http://www.sportcentrum-ivanovice.cz/html/cs/sportovni_aktivity.phtml“.

Rezervace jsou vyřešeny podobně jako u Hcentra, tedy nepřihlášení uživatelé si mohou zobrazit vytíženost kurtu [22].

3.4. Squashcentrum Chomutov

Internetové stránky squashcentra Chomutov se vyznačují příjemnými barvami a jednoduchým rozvržením. Nepříjemné je rozhození layout při odkazu na „Rozpis cvičení“ pod sekcí kola. Tento odkaz způsobí přepsání kraje layout tabulkou „SÁL KOLA“. Také členění internetových stránek není příliš kvalitní. Odkaz na online rezervace se zobrazí hned na úvodní stránce, ale po přesunu na jinou stránku tohoto webu lze online rezervace nalézt pouze pomocí mapy stránek. Squashcentrum Chomutov také nemá všechny odkazy plně funkční např. odkaz, který je pod squash → amatérská liga, se zobrazí jako zdrojový skript. Testem na validitu tyto stránky neprošly úspěšně.

Rezervace lze vytvořit telefonicky, ale také online po úspěšném zaregistrování. Online registrace je velmi rychlá a nežádá žádné zbytečné údaje. Hned po registraci je zobrazen odkaz pro přihlášení. Zobrazení online rezervací je velmi přehledné s velmi příjemnými barvami. Velmi vítána je možnost nápovědy, kde je např. uvedeno, do kdy je možno rezervaci zrušit. Systém rezervací ale není přehledný, např. odkaz pro zrušení rezervace je umístěn na konci stránky, nenabízí možnosti zobrazení aktuálních uživatelských rezervací.

Velkým nedostatkem rezervačního systému je, že není ošetřeno vkládání rezervací s neaktuálním datem a časem. Dále není vhodné, že u potvrzení vybrané rezervace, je opět vkládáno jméno uživatele, který je již registrován [23].

3.5. A-Sport

Tyto internetové stránky jsou graficky zpracovány velmi průměrně a mají jednoduchou strukturou. Mezi odkazy chybí např. fotogalerie sportovního centra. Všechny odkazy jsou plně funkční. Velkou nevýhodou těchto internetových stránek je, že jsou tvořeny staticky. Tuto metodu tvorby bych očekával u základní prezentace firmy, ale ne u stránek sportovního centra. Validita těchto stránek není bez chyby.

Rezervaci lze uskutečnit pouze telefonicky, a to bez možnosti zobrazení vytíženosti kurtů. Online rezervace sportovní centrum A-Sport nepodporuje, čímž může přicházet o zákazníky, kteří preferují rezervace online [24].

Zhodnocení sportovních center vystihují následující tabulky.

Tabulka 1: Zobrazení počtu chyb na stránkách sportovních center.

Název	Počet chyb při zobrazování	Počet chybných odkazů	Počet chyb po kontrole validity
Hcentrum	1	0	13
Squashpark-Cibulka	3	1	96
Sportcentrum-Ivanovice	0	2	13
Squashcentrum-Chomutov	2	2	15
A-sport	0	0	20

Tabulka 2: Mé hodnocení internetových stánek sportovních center. K hodnocení je využito známkování: 1 – nejlepší, 5 – nejhorší.

Název	Grafické zpracování	Přehlednost	Funkčnost
Hcentrum	2	1	1
Squashpark-Cibulka	4	3	3
Sportcentrum-Ivanovice	1	3	2
Squashcentrum-Chomutov	1	4	2
A-sport	3	2	1

Tabulka 3: Možnosti rezervačních systému sportovních center, x – ano, - ne.

Název	Prohlídka rezervací	Online registrace	Online rezervace
Hcentrum	**x**	-	**X**
Squashpark-Cibulka	-	-	-
Sportcentrum-Ivanovice	**x**	-	**X**
Squashcentrum-Chomutov	**x**	**x**	**X**
A-sport	-	-	-

4. Analýza sportovního centra

Cílem je vytvořit internetovou aplikaci, která umožní správu sportovního centra. Aplikace obsahuje čtyři uživatelské role: návštěvník, běžný uživatel, dále uživatel s právem tvorby turnaje a poslední je administrátorský učet. Tento účet se stará o správu uživatelů, turnajů, sportovních potřeb atd.

K vytvoření této aplikace jsem si vybrat databázi MySQL pro její velkou podporu poskytovatelů internetových stránek. Pro zobrazení dat je nutné vytvářet obsah stránek dynamicky. Tuto vlastnost využívá i programovací jazyk PHP, jejž jsem si pro napsání své bakalářské práce vybral. Jazyk PHP zajistí odeslání a následné přijetí dotazu z databázového serveru. Tato data zpracuje na serveru a odešle je klientovi, kde jsou zobrazena jako HTML stránka.

4.1. Požadavky

Aplikace by měla být z pohledu zákazníka přehledná. Zákazník nesmí mít pocit, že se na stránkách ztratil. V případě zrušení jeho rezervace, např. z důvodu pořádání turnaje, musí být informován. Dále by měl být informován o nejnovějších nabídkách sportovního centra.

Jako v každé internetové aplikaci je velmi důležitá bezpečnost, která zaručuje nejen ochranu osobních údajů při registraci, ale hlavně zabraňuje neoprávněnému vniknutí do aplikace.

Je také nutné zachovat referenční integritu v databázi, aby nedošlo k tomu, že jedna rezervace byla přidána jinému zákazníkovi.

4.2. Uživatelé systému

4.2.1. Návštěvník

Uživatel, který nemá v systému vytvořen účet, má možnost zobrazení rezervací, výsledků turnajů a žebříčku. Dále může vkládat příspěvky do internetové diskuze. Poslední návštěvníkovou volbou je registrace do systému.

4.2.2. Běžný uživatel

Tento uživatel má nastaveno nejméně práv. Po úspěšné registraci a následném přihlášení do systému, může využívat rezervační systém. Má tedy k dispozici možnosti vytvoření, zobrazení a smazání rezervace jak sportovních kurtů, tak i sportovních potřeb. Dále se může přihlásit na pořádaný turnaj. Systém samozřejmě

umožňuje tomuto uživateli i odhlášení z turnaje. Má také možnost vložit příspěvek do internetové diskuze. Poslední funkcí systému pro běžného uživatele je změna osobních údajů a hesla.

4.2.3. Uživatel s rozšířenými právy

Má stejná práva jako běžný uživatel, navíc má právo vytvářet turnaje.

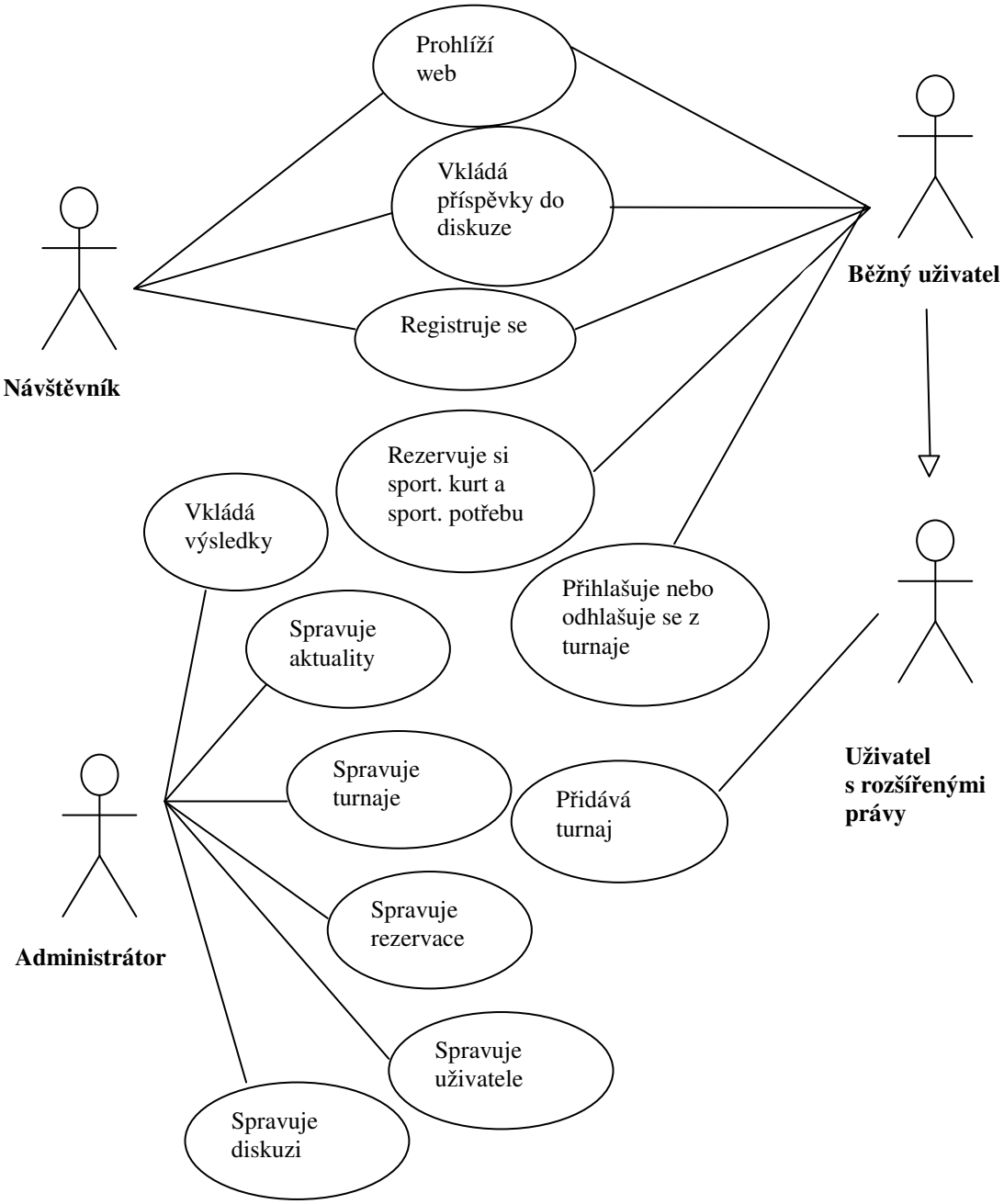
4.2.4. Administrátor

Uživatel s administrátorským účtem využívá všechny funkce aplikace. Disponuje právem přidání, smazání a přidělování práv uživatelům. Aplikace administrátorovi umožňuje smazání turnaje, pokud se tento turnaj nebude konat. Může mazat jednotlivé příspěvky v diskuzi, např. je-li v příspěvku uveden nějaký vulgární výraz. Má na starosti vkládání výsledků po odehraném turnaji. Spravuje sportovní potřeby, které si mohou zákazníci rezervovat. Může vkládat rezervace neregistrovaných uživatelů do aplikace např. po telefonické domluvě. Administrátor zodpovídá za vkládání novinek na úvodní stránce.

4.3. Use Case diagram

Pro analýzu aplikační logiky, tedy pro popis interakcí mezi uživatelem a systémem, jsem nakreslil Use Case diagram.

Obrázek 5: Use Case Diagram



5. Zvolená technologie

5.1. PHP

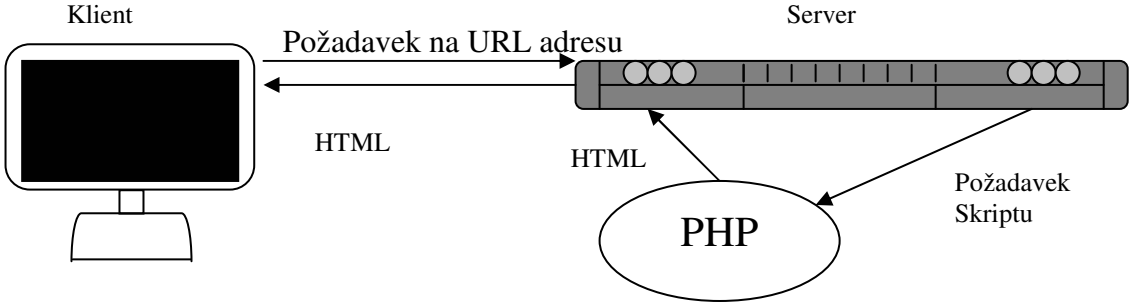
Označení PHP bylo původně zkratkou anglické fráze „Personal Home Page“. Autor této technologie je Rasmus Lerdorf, který ji vytvořil pro sledování návštěvnosti svých stránek.

S postupným nárůstem užitečnosti a možnosti této technologie se ujal název „PHP: Hypertext Preprocessor“.

Syntaxe jazyka PHP vychází ze známých programovacích jazyků jako jsou Perl, Java, a C. Stejně jako uvedené programovací jazyky je PHP nezávislý na platformě a je Open Source.

Skripty PHP jsou zpracovávány na straně serveru a výsledek skriptů je přenášen ke koncovému uživateli. Hlavní myšlenka PHP je v dynamickém generování obsahu HTML stránek. Protože jazyk PHP se používá s kombinací databázového serveru, je vygenerovaný obsah je velmi často výsledkem nějakého databázového dotazu [20].

Obrázek 6: Princip generování html stránek pomocí PHP



5.2. MySQL

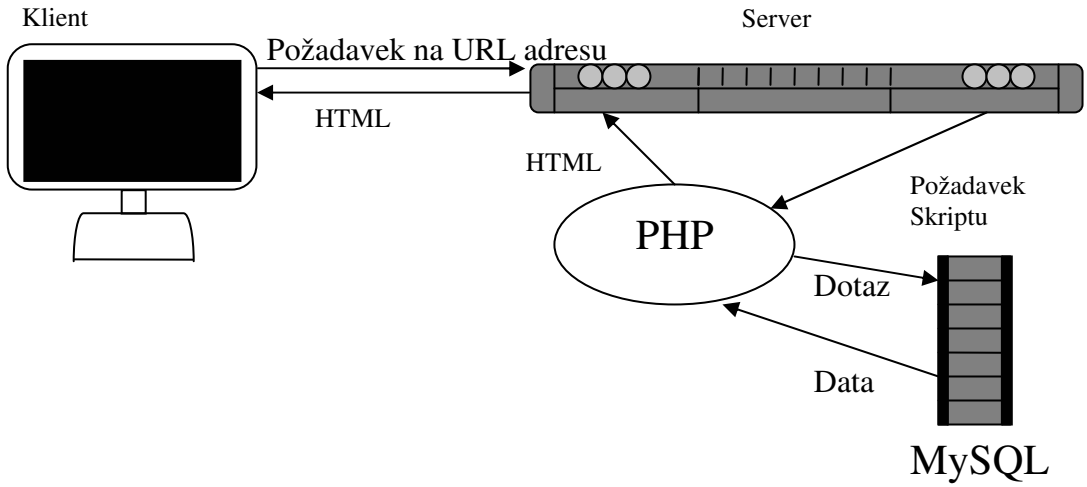
Je databázový systém, který byl vytvořen firmou MySQL_AB a mezi jeho autory patří Michael „Monty“ Widenius a David Axmark.

MySQL je multiplatformní databáze s bezplatnou licencí GPL. Komunikace s databází probíhá pomocí jazyka SQL. Stejně jako u jiných SQL databází se jedná o dialekt tohoto jazyka, který obsahuje některá rozšíření.

MySQL se velmi často používá v kombinaci s jazykem PHP. Princip generování HTML stránky za pomocí PHP a MySQL je zobrazen na obrázku č. 7. MySQL

od verze 5 začalo podporovat pohledy, triggery a uložené procedury, a tím ještě více v vzrostla oblíbenost této databáze [20].

Obrázek 7: Princip generování html stránek pomocí PHP



5.3. CSS

CSS je zkratka anglického názvu „Cascading Style Sheet“, česky tabulky kaskádových stylů, které umožňují grafickou úpravu stránek psaných v jazycích HTML, XHTML a XML.

Jazyk byl navržen standardizační organizací W3C. Zatím byly vydány tři úrovně specifikace CSS1, CSS2, a CSS3. CSS3 se v současné době rozšiřuje.

Kaskádové styly slouží k definování vzhledu jednotlivých elementů nebo skupiny elementů. Jejich velkou výhodou je oddělení obsahu HTML dokumentu a grafické struktury. Kód je přehlednější a nemusíme formátovat každý element zvlášť, stačí pro něj vytvořit v CSS vzhled pouze jednou.

5.4. Javascript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk. Autor Javascriptu je Brendan Eich z tehdejší společnosti Netscape.

JavaScript je prováděn na straně klienta, tím je obvykle internetový prohlížeč. Zapisuje se přímo do HTML kódu. Tato technologie se obvykle používá v kombinaci s kaskádovými styly a událostmi jednotlivých elementů HTML např. „onclick“ nebo „mouseover“. JavaScript pracuje s jednotlivými komponentami prohlížeče a stránkami pomocí objektového modelu dokumentu (DOM) tak, že zpracovává atributy objektů dokumentu a jejich hodnoty.

6. Návrh databáze

Aplikace není navržena pro určitý počet sportovních kurtů a ani pro předem daný typ sportovišť. Tedy prvním kritériem při tvorbě databáze byla univerzálnost. Dalším kritériem byla bezpečnost, tedy dodržení integrity dat a zabezpečení proti neoprávněnému zásahu.

6.1. Architektura

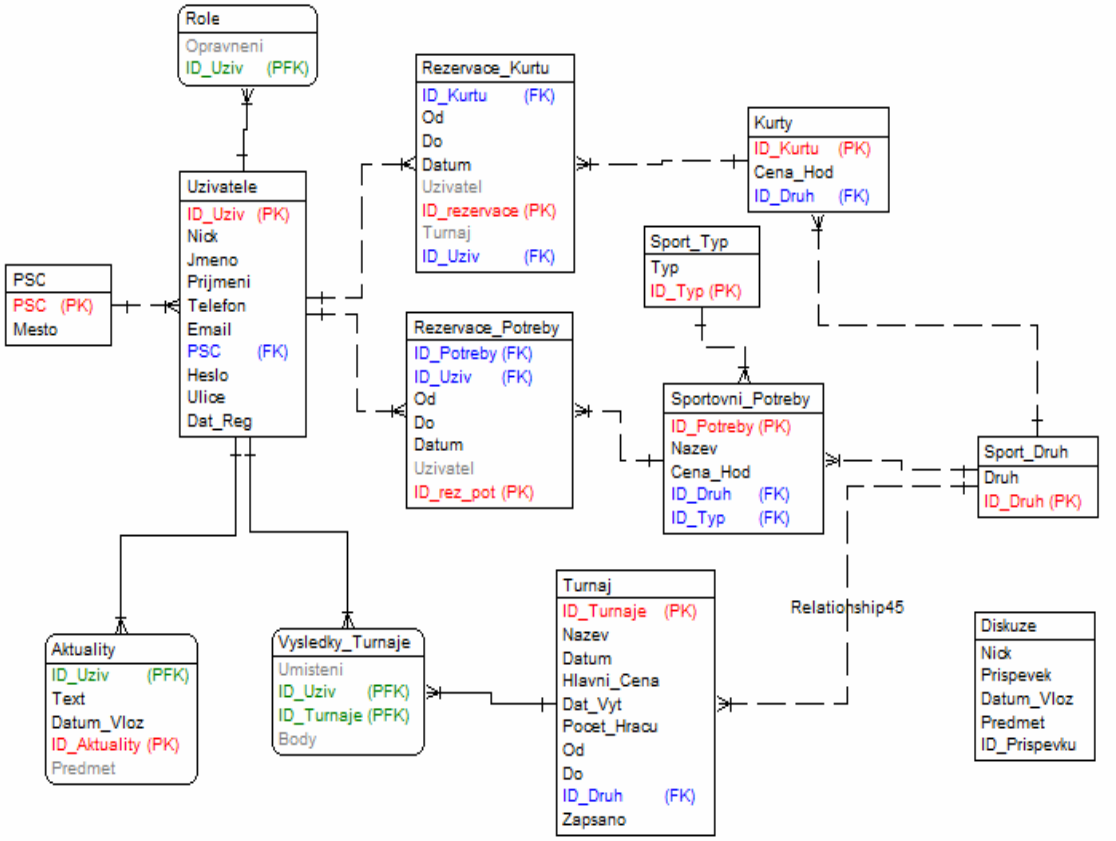
Aplikace využívá data poskytovaná z databáze. Z důvodu bezpečnosti byl vytvořen přístup, který má nastavena jen nejnutnější uživatelská práva.

Pro přístup k těmto datům jsou využity pohledy. Pohledy skryjí celou databázovou strukturu. Ve většině případů spojují více tabulek. Pomocí pohledů lze vyfiltrovat pouze některé záznamy.

Pro změny v databázi (vložení, úprava nebo smazání) je vytvořena databázová procedura. V procedurách je zapouzdřena změna uskutečněná ve více tabulkách najednou. U těchto změn je většinou využívána transakce. Provedou se tedy všechny změny, nebo žádné. U transakce nesmí nastat případ, kdy se vykoná jen její část.

6.2. E-R Diagram

Obrázek 8: Vytvořený E-R Diagram



6.3. Základní popis tabulek

Uvedené tabulky splňují třetí normální formu. Zde následuje detailnější popis tabulek, které jsou využity v aplikaci.

- **Role** – v této tabulce jsou uživatelské role. Tuto tabulku využívá tabulku uzivatele.
- **Uzivatele** – tabulka obsahuje informace o registrovaných uživatelích a také o pracovnících centra. Sloupce v této tabulce jsou číslo uživatele, jméno, příjmení, uživatelské jméno, telefon, email, heslo, ulice a datum registrace. Tabulka je propojena s tabulkami Vysledky_Turnaje, Rezervace_Kurtu, Rezervace_Potreby a Aktuality.
- **PSC** – zde jsou uložena poštovní směrovací čísla s příslušnými názvy měst.
- **Rezervace_Kutru** – obsahuje informace o rezervaci jednotlivých kurtů. Jsou zde uložena data: datum, začátek rezervace, konec rezervace, číslo kurtu, číslo uživatele a nepovinný atribut uzivatel. Tento atribut je využíván při telefonické rezervaci, kdy tento údaj zapisuje do tabulky zaměstnanec sportovního centra.
- **Kurty** – informace o sportovních kurtech. Tabulka má tyto sloupce: číslo kurtu, cenu za hodinu a číslo sportovního druhu.
- **Sport_Druh** – zde jsou uloženy druhy sportovišť. Tato tabulka obsahuje sloupce: číslo sportovního druhu a název druhu (tenis, squash atd.). Tabulka je využívána tabulkou kurty a také sportovní potřeby.
- **Rezervace_Potreby** – zde jsou uvedena v podstatě stejná data, jako v tabulce Rezervace_Kurtu, tedy sloupce s názvy datum, začátek a konec, číslo rezervované sportovní potřeby, číslo uživatele a nepovinný atribut uživatel.
- **Sportovní_Potreby** – tabulka uchovává informace o sportovních potřebách. Sloupce v této tabulce jsou: název sportovní potřeby, cena za hodinu. Tato tabulka je propojena s tabulkami Sport_Druh a Sport_Typ.
- **Sport_Typ** –v této tabulce jsou uloženy typy sportovních potřeb např. raketa, boty, míček atd.
- **Turnaj** – tabulka obsahuje data o pořádaných turnajích. K tomu využívá sloupce: název turnaje, čas, kdy se turnaj koná, začátek, konec, hlavní cenu, datum vytvoření v systému, maximální počet přihlášených hráčů. Tato tabulka využívá tabulku Sport_Druh

- **Vysledky_Turnaje** – obsahuje informace o výsledkách jednotlivých turnajů, tedy kolik a jaké umístění získal přihlášený hráč.
- **Aktuality** – tabulka s uloženými aktualitami má sloupce: text, datum vložení, předmět. Tabulka je spojena s tabulkou uzivatele.
- **Diskuze** – tabulka sloužící pouze pro ukládání příspěvků do internetové diskuze. Sloupce v této tabulce jsou: příspěvek, nick (přezdívka), datum vložení, předmět.

6.4. Pohledy

Využívání pohledů místo umístěných databázových dotazů ve zdrojovém kódu zvyšuje bezpečnost aplikace. Další výhodou je zkrácení kódu, protože pohled zapouzdřuje spojení mezi tabulkami. V názvech pohledu byly použity anglické názvy tabulek a sloupců.

Příklad pohledu, který spojuje tabulky Sport_druh, Sport_typ a Sportovni_potreby:

```
create or replace view Sport_equipment as
(select typ as type, sport_druh.Druh as kind, sportovni_potreby.Nazev as title,
sportovni_potreby.Cena_Hod as price, sportovni_potreby.ID_Potreby as id_p
from sportovni_potreby, sport_typ, sport_druh
where sportovni_potreby.ID_Potreby  and
sportovni_potreby.ID_Typ = sport_typ.ID_Typ and sportovni_potreby.ID_Druh =
sport_druh.ID_Druh)
```

6.5. Procedury

Pomocí procedur lze opět docílit větší bezpečnost v databázi. V uložených procedurách jsou vykonávány různé změny v databázi, které jsou při volání procedury ve zdrojovém kódu skryty. V aplikaci používám uložené procedury k jakýmkoliv změnám v databázi, tedy ke změně smazání a vkládání dat do tabulek.

Přiklad procedury, která upravuje uživatelská data v systému:

```
CREATE PROCEDURE Update_U(id_u int, nick char(20), jmeno char(20), prijmeni
char(20), telefon int, email char(30), psc int, ulice char(20))
begin
UPDATE uzivatele SET Nick = nick, jmeno = jmeno, prijmeni = prijmeni, telefon = telefon,
email = email, psc = psc, ulice = ulice
        where id_uziv = id_u;
end;
```

Příklad procedury pro vložení turnaje do databáze:

```
CREATE PROCEDURE Insert_T(naz char(20), dat date, hl_cen char(20), dat_vy date, poc
int(11),zac time, kon time, dr char(10))
begin
INSERT INTO turnaj (Nazev, Datum, Hlavni_cena,Dat_Vyt, Pocet_Hracu, Od, Do, Druh)
VALUES (naz, dat, hl_cen, dat_vy, poc, zac, kon, dr);
end;
```

Příklad procedury, která smaže uživatele ze systému:

```
CREATE PROCEDURE Delete_U(id_u int)
begin
  delete from uzivatele where ID_Uziv=id_u;
  delete from rezervace_kurtu where ID_Uziv=id_u;
  delete from rezervace_potreby where ID_Uziv=id_u;
end;
```

6.6. Indexy

Pro optimalizaci databáze jsem vytvořil indexy v tabulce diskuze u sloupce id_prispevku, protože neobsahuje primární klíč, a dále u tabulek Rezervace_kurtu a Rezervace_potreby, kde jsou použity sloupce datum, od a do. Tyto sloupce jsou velmi často používány v databázových dotazech. Ostatní indexy jsou vytvořeny automaticky, protože se jedná o sloupce s primárními a cizími klíči.

Příklad vytvořených indexu:

```
Create Index Rez_kurt ON Rezervace_Kurtu (Datum,Od,Do);
Create Index Rez_pot ON Rezervace_Potreby (Datum,Od,Do);
```

7. Zabezpečení aplikace

Pro bezpečnost aplikace je velmi důležité, aby nedošlo k neoprávněnému vniknutí do aplikace nebo ke ztrátě citlivých dat.

7.1. Ošetřování formulářů

Jak jsem již zmiňoval, formuláře jsou velmi často cílem internetových útoků, a proto je důležité, je dostatečně zabezpečit. Ve formuláři je nejprve ošetřeno, zda jsou všechna vstupní data vyplněna. K tomu jsem využil PHP funkci empty. Jsou-li vstupní data čísla, jsou testována funkcí Je_Cislo. Tato funkce ověří, zda vložený vstup obsahuje čísla a zda má přijatelnou délku, např. délka telefonního čísla je 9 znaků (předpoklad, že jde o česká telefonní čísla).

Ukázka funkce Je_Cislo:

```
function Je_Cislo($cislo, $pocet)
{
 $bool = false;
 $cis = strlen($cislo);
 if(is_numeric($cislo))
   {
    if ($cis==$pocet)
      {
       $bool = true;
      }};
    return $bool;
}
```

Pokud vstupní data obsahují textové znaky, jsou testována funkcí Ochrana_dat_zapis. V této funkci je řetězec testován proti běžným útokům.

Ukázka funkce Ochrana_dat_zapis:

```
function Ochrana_dat_zapis($text)
{
  $text = strip_tags($text);
  $text = trim($text);
  $text =  htmlspecialchars($text,ENT_QUOTES);
  $text = addslashes($text);
  return $text;
}
```

Dále testuji u některých vstupních dat, zda vložené údaje mají smysl. V tomto testování jsem využil regulární výrazy, např. aby nedošlo k vložení čísla do pole příjmení, také testuji textové pole s e-mailem, zda obsahuje znak @.

Ukázka regulárního výrazu:

```
  if (!EregI("^[a-z0-9]+[a-z0-9\._-]*[a-z0-9]+@[a-z0-9]+[a-z0-9\._-]*[a-z0-9]+\.[a-
z]{2,3}$", $e) && !EregI("\.{2,}", $e) && !EregI("_{2,}", $e) && !EregI("-{2,}", $e))
```

7.2. Zabezpečení session

Pro zabezpečení session jsem nemohl použít žádnou hashovací funkci, protože funkce pro hashování vytvoří otisk řetězce, který nelze získat zpět. Pro zakódování session jsem vytvořil funkci SafeIn, která zakóduje hodnotu session proměnné. Protože používám v rezervačním systému hodnotu session jako identifikační číslo uživatele uložené v databázi. Vytvořil jsem funkci SafeOut, která zakódovanou session dekóduje zpět do původního tvaru.

7.3. Hesla

Kvůli bezpečnosti jsem pro ukládání hesel zvolil metodu dvojitého hashování. Nejprve heslo zahashuji pomocí fukce sha256, která vytvoří 32 bitový otisk. Následně vytvořím pomocí funkce hash otisk výsledku funkce sha256.

Příklad hashování hesla:

```
$heslo = hash('sha256', 'nosilac');
```

8. Vývoj aplikace

Ve své bakalářské práci jsem naprogramoval rezervační systém pro sportovní centra. K implementaci jsem si zvolil výše uvedené technologie PHP, CSS, Javascript a relační databázi MySQL. Aplikace má, jak již bylo uvedeno, tři uživatele, kde každý má své vlastní menu. Toto menu se liší v počtu odkazů podle oprávnění přihlášeného uživatele. Rozvržení stránky obsahuje tři části, v levé je umístěno navigační menu, ve střední části je hlavní stránka a pravá část obsahuje informace o otevírací době a aktuality.

8.1. Adresářová struktura

Z důvodu bezpečnosti nebyly všechny soubory pouze v kořenovém adresáři, ale v aplikaci jsem vytvořil následující adresářovou strukturu:

ADMIN – obsahuje soubory, které může využívat pouze administrátor.

CSS – zde jsou uloženy kaskádové styly.

JS – adresář vytvořenými Javascripty.

SECURE – soubory pro připojení k databázi a soubor s navigací.

USER – v tomto adresáři jsou soubory určené pro běžného uživatele.

USER2 – adresář se soubory uživatele s rozšířenými právy.

MENU – tento adresář obsahuje soubory s typy menu.

8.2. Registrace

V souboru Registrace.php je formulář pro přidání uživatele do systému. Pro ošetřování vstupních řetězců jsou volány výše uvedené funkce. Jsou-li vstupní data v pořádku, zavolá se databázová funkce pro vložení uživatele do systému.

8.3. Přihlašování do systému

Pomocí souboru Login.php systém zkontroluje, zda vložené uživatelské jméno a heslo je uloženo v databázi. Pokud se údaje shodují, zobrazí se text „Přihlášení proběhlo úspěšně“. Po přihlášení se do levého sloupce vloží menu s příslušnými odkazy z adresáře MENU podle oprávnění uživatele. Dále se vytvoří session id_uzivatele, se kterou pracuje celý rezervační systém spolu se správou uživatelů a turnajů.

8.4. Rezervace sportovních kurtů

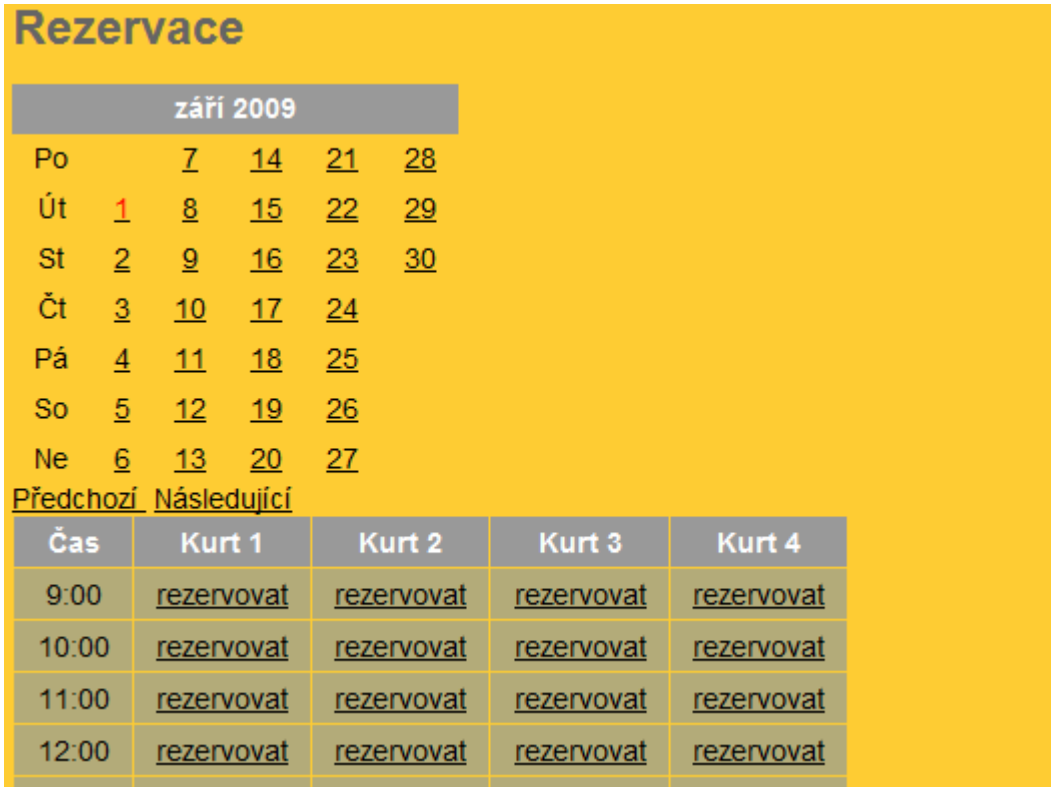
Pro vložení rezervace se volá stránka Zob_Kal.php s URL parametrem typ. Na této stránce se voláním funkce kalendar zobrazí kalendář. Dále je volána funkce vypis_casu, která zobrazuje, zda je kurt volný nebo obsazený viz obr. 7. Zamezení vložení rezervace s neaktuální hodinou ošetřuje funkce aktuální. Protože je tato funkce využívána v přidávání turnaje, má kromě vstupních parametrů čas a datum, parametr pro počet měsíců.

Ukázka funkce aktualni:

```
function aktualni ($hodina,$datum,$pocet_mesicu)
{
 $akt=false;
 $dat = explode("-", $datum);
 $mes = $dat[1] - $pocet_mesicu;
  if ($dat[0] > date('o'))
  {
   $akt=true;
  }
  else
   {
    if ($mes > date("n"))
     {
      $akt=true;
     }
    else
     if ($dat[2] > date("j") && $mes == date("n"))
      {
       $akt=true;
      }
      else
      if ($hodina > date("G") && $dat[2] == date("j"))
         $akt=true;
   }
 return $akt;
 }
```

V případě, že je datum a čas rezervace v pořádku, přebírá vložení rezervace databázová funkce Insert_Rez. Jedná-li se o rezervaci po telefonu, tedy pro uživatele, který není registrován, musí rezervaci do systému vložit administrátor. Tento typ vložení rezervace je vytvořen na stránce Rezervace_Zam.php, která je z důvodu bezpečnosti v adresáři admin. Ta obsahuje stejné funkce i grafické rozvržení, rozdíl je pouze v přesměrování na stránku Vlozit_Rez.php po vybrání rezervace.
Na přesměrované stránce je formulář, kde se vkládá jméno uživatele. Po jeho vložení a potvrzení tlačítkem se zavolá databázová funkce.

Obrázek 9: Stránka s rezervacemi



Systém umožňuje prohlídku rezervací. Prohlídka je naprogramována v souboru Zobraz_Rez.php. Tato stránka ošetří, zda je uživatel přihlášen. Pokud uživatel přihlášen není, zobrazí se text „Nejste přihlášeni“ a odkaz pro přihlášení a registraci. Je-li uživatel přihlášen, zobrazí se jeho aktuální rezervace. Administrátor má možnost výpisu rezervací jiných uživatelů. Na stránce Vyhledat_Rez je vytvořeno vyhledávání rezervací. Protože je tato stránka využívána pro zrušení rezervace, je při kliknutí na odkaz zobrazení rezervace poslána proměnná akce. Je-li na tuto stránku poslána proměnná akce s hodnotou jedna, systém rozpozná, že jde o zobrazení rezervací. Na stránce je textové pole pro vyhledávání rezervace a dále

možnost výběru kritéria vyhledávání. V případě, že systém nalezne v databázi stejný záznam jako ten, který byl vložen do textového pole, je tento záznam vypsán.

Zrušení rezervace je v souboru Zruseni_Rez.php. Je-li uživatel přihlášen, zobrazí se tabulka s jeho rezervacemi. Poslední volba této tabulky je možnost vybrání jednotlivé rezervace a po následném stisknutí dojde k porovnání zatržené rezervace s rezervacemi v databázi.

Ukázka kódu ze stánky Zruseni_Rez.php:

```
if (isset($_POST['odeslat']))
 {
  $inc=0;
  $result = mysql_query("select id_r from vypis_rez where ID_U='$id';");
  while ($row = mysql_fetch_array($result, MYSQL_NUM))
    {
    $id_rezervace[$inc]=$row[0];
    $inc++;
    }
  for ($i=0; $i<=$inc; $i++)
    {
      if ($_POST[$id_rezervace[$i]] ==1 )
        {
        mysql_query("call delete_r('$id_rezervace[$i]');");
        }
    }
 }
```

Administrátorský účet může zrušit jakoukoli aktuální rezervaci. Tuto rezervaci musí nejprve vyhledat. O vyhledávání jsem se již zmiňoval při zobrazení rezervací. Systém volá stejnou stránku, pouze hodnota URL proměnné akce je dvě. Stránka je totožná, jediný rozdíl je při zobrazení výpisu nalezené rezervace. Rezervace je zobrazena jako odkaz, kde po následném kliknutí je uživatel přesměrován na stránku Zruseni_Rezervace.php. Zde jsou údaje o rezervaci a tlačítko pro smazání. Kliknutím na tlačítko je volána databázová funkce pro smazání rezervace.

Rezervační systém zahrnuje i možnost rezervovat sportovní potřebu. O tuto volbu se starají stránky Vyb_Rez.php a Vyb_Pot.php. Nejprve se budu zabývat stránkou Vyb_Rez.php. Tato stránka nejdříve ověří přihlášení uživatele, dále zjistí, jestli má

uživatel vytvořenou rezervaci sportovního kurtu. V případě, že má uživatel více druhů rezervací, pak se při první návštěvě zobrazí druh rezervace, která je v databázi uložena jako první. Možnost přepnutí je v odkazech, které jsou generovány z databáze a předávány jako proměnná typ v URL adrese.

Ukázka kódu ze stánky Vyb_Pot.php:

```
$dotaz = mysql_query("select kind from equipment where dat>='$date' and ID_U = '$id'
group by kind order by kind;");
$inc =0;
  while ($row = mysql_fetch_array($dotaz))
    {
      $druh[$inc] = $row[0];
      $inc++;
    };
for ($i=0;$i<=$inc;$i++)
  {
    echo "<a
href='index.php?str=Vyb_Rez&amp;typ=$druh[$i]'>$druh[$i]</a>  ";
  };
```

Pod těmito odkazy je tabulka s vytvořenými rezervacemi a volbou pro vybrání rezervace. Pokud uživatel potvrdí výběr tlačítkem, je přesměrován na stránku Vyb_Pot.php s proměnnou id_rez. Na této stránce se zjišťuje začátek, konec a datum rezervace. Dále se v databázi hledají sportovní potřeby, které jsou v daném časovém období volné. Následně se tyto potřeby zobrazí. Po vybrání sportovních potřeb se tyto sportovní potřeby porovnávají s těmi, které jsou uloženy v databázi. Pokud jsou záznamy stejné, je volána Insert_Rez_P, která vloží příslušná data do databáze.

8.5. Žebříček hráčů

Možnost podívat se na žebříček hráček, hráčů je vytvořena ve stránce Zobrazit_Zebricek.php. Zde se zobrazí výpis z databáze s pořadím, jménem, příjmením a počtem získaných bodů. Pro zobrazení detailního výpisu výsledků jednotlivého hráče stačí kliknout na jeho příjmení.

8.6. Osobní údaje

Každá aplikace by měla umožňovat změnu uživatelských údajů. V mé aplikaci k tomu slouží stránky Zmena_Uziv.php a Zmena_Hesla.php. Stránka Zmena_Uziv.php slouží k editaci uložených uživatelských údajů. Na stránce je umístěn formulář, kde se v textových polích zobrazí uložená data z databáze.

Stránka Změna_Hesla.php, jak je již z názvu patrné, obsahuje formulář pro změnu hesla. Tato stránka nejprve porovná staré heslo s heslem uloženým v databázi a je-li totožné provádí testování, zda jsou obě nová hesla stejná.. Pokud jsou hesla stejná, nahradí staré heslo novým.

8.7. Sportovní potřeby

Jak bylo již zmiňováno, je v aplikaci začleněná správa sportovních potřeb. Tato volba je možná jen pod administrátorským účtem. Vytvořené stránky pro správu jsou Pridat_Sport_Pot.php a Smazat_Sport_Pot.php. Na první uvedené stránce se nalézá formulář pro přidání sportovních potřeb, které jsou po stisknutí tlačítka „Přidat“ vloženy do databáze.

Pro smazání slouží stránka Smazat_Sport_Pot.php viz obr. 8 Na této stránce se nejprve nabízí volby sportovního druhu (squash, tenis a badminton), které jsou výsledkem databázového dotazu. Níže jsou na stránce uvedeny volby pro typ sportovní potřeby. Tato volba je opět výsledkem dotazu z databáze. Pokud je uživatel na této stránce poprvé, je vybrán typ i druh sportovní potřeby podle abecedního pořadí uloženého v databázi. Pro přehlednost je pod těmito volbami zobrazení, pod jakým typem a druhem se uživatel nachází. Následuje tabulka s vybranými sportovními potřebami. Poslední sloupec v tabulce je určen pro zaškrtnutí sportovní potřeby, kterou chce uživatel smazat. Po stisknutí tlačítka dojde k smazání vybraných sportovních potřeb.

Obrázek 10: Stránka pro smazání sportovní potřeby

8.8. Turnaje

První možností v této sekci je zobrazení výsledků turnaje. Na stránce Zobrazit_Vysledky.php je vytvořen nejprve výběr druhu turnaje (squash, badminton nebo tenis.). Pokud je uživatel na této stránce poprvé, je druh vybrán jako první záznam z databáze. Dále následuje tabulka s detailním výpisem turnaje, kde je poslední sloupec určen pro zatrhnutí turnaje. Je-li vybrán turnaj a uživatel klikne na potvrzovací tlačítko, dojde k zobrazení výsledku vybraného turnaje. Uživatel má z této stránky možnost prokliknutí na stránku se zobrazením uživatele na žebříčku.

Do turnaje se může přihlásit každý registrovaný uživatel. Přihlášení do turnaje je vytvořeno ve stránce Prih_Turnaj.php, kde jsou zobrazeny druhy turnajů, na které je možno se přihlásit. Po zatržení vybraného turnaje a stisknutím potvrzovacího tlačítka, dojde k přihlášení uživatele k turnaji.

Přidávat turnaje může uživatel s rozšířenými právy a administrátor. Pro přidání turnaje je naprogramovaná stránka Pridat_Turnaj.php. Na této stránce je formulář s textovými poli a s volbami pro druh turnaje. Jsou-li správná vstupní data ve formuláři a uživatel stiskne tlačítko pro přidání turnaje, je turnaj vložen do databáze. Následně je volána funkce pro zjištění vytíženosti kurtů dne, kdy je turnaj pořádán. Pokud jsou v den pořádání turnaje nějaké rezervace, jsou zrušeny a uživatelům, kteří tuto rezervaci měli, je poslán omluvný email.

Volba smazaní turnaje je k dispozici pouze administrátorovi aplikace. Smazání je vytvořeno v stránce Smazat_Turnaj.php. Na této stránce se zobrazuje výpis z tabulky

turnaje, kde jsou uloženy údaje o turnajích. Zobrazují se pouze turnaje, které mají aktuální datum. Stránka Smazat_Turnaj.php obsahuje možnosti pro vybrání turnaje a po stisknutí tlačítka "Smazat" dojde k jeho smazání.

8.9. Správa uživatelů

Tato volba je přístupná pouze pro administrátora.. Přidat do systému se samozřejmě může každý uživatel, pokud zvolí odkaz obsahující registrační formulář. Tato stránka je volána i pod možností "Přidat" v odkazu "Uživatele."

Pro změnu oprávnění je nejprve zobrazena stránka Vyhledej.php, kde je URL proměnná akce rovna jedné. Na této stránce je textové pole a dále zaškrtávací políčka pro kriteria vyhledávání. Jsou-li data vložená do textového pole shodná s daty v databázi, je zobrazen odkaz na vyhledaného uživatele. Po kliknutí na tento odkaz dojde k přesměrování na stránku Zmena_Role.php. Zde se nalézá podrobný výpis uživatelských dat, textové pole pro vložení oprávnění a tlačítko pro změnu oprávnění. Po stisknutí tlačítka je uživateli nastaveno nové oprávnění.

Poslední možností správy uživatelů je smazaní uživatele. Zde je opět zobrazena již zmiňovaná stránka Vyhledat.php, tentokrát je URL proměnná akce rovna dvěma. Jsou-li záznamy z databáze stejné jako vložený záznam do textového pole, je uživatel přesměrován na stránku Smazat_Uziv.php. Stránka zobrazí podrobnější informace uživateli a tlačítko pro smazání. Po stisknutí tlačítka dojde k smazání uživatele ze systému.

8.10. Aktuality

Stránka Aktuality.php se stále zobrazuje v pravém sloupci rozvržení stránky. Záznamy jsou výsledkem databázového dotazu. Aby byl pravý sloupec přehledný, je dotaz omezen na čtyři nejaktuálnější záznamy.

Další volby pro přidání a úpravu stránky Aktuality může využívat pouze administrátor. Přidání aktuality je naprogramováno ve stránce Pridat_Aktuality.php. Na stránce je umístěn formulář s textovými poli. Html tag textarea neobsahuje atribut pro omezení délky vkládání znaků. Tento nedostatek jsem vyřešil pomocí Javascriptu, který sleduje, kolik volných znaků ještě zbývá. Poměr mezi maximálními a vloženými znaky je zobrazen pod textovým polem. Po vložení všech údajů a stisknutí tlačítka se záznamy vloží do databáze.

Ukázka Javascriptu:

```
var MaxLengthLock = false;
    function MaxLengthCount(fieldObj,fieldMaxLength)
    {
     if (!MaxLengthLock)
     {
       MaxLengthLock = true;
       if (fieldObj.value.length > fieldMaxLength)
       {
         alert("Text je delší než " + fieldMaxLength + " znaků!");
         fieldObj.value = fieldObj.value.substring(0,fieldMaxLength);
       }
       var percentage = parseInt(100 - (( fieldMaxLength - fieldObj.value.length) *
100)/fieldMaxLength);
       document.getElementById(fieldObj.id + "_PBar").style.width =
parseInt((parseInt(fieldObj.offsetWidth)*percentage)/100)+"px";
       MaxLengthLock = false;
     }
    }
```

Pro úpravu aktualit je vytvořena stránka Vybrat_Aktualitu.php. Tato stránka nejprve zobrazí tabulku se čtyřmi nejnovějšími aktualitami. V posledním sloupci je volba pro vybrání aktuality, která je určena pro editaci. Po vybrání aktuality a stisknutí dojde k přesměrování na stránku Upravit_Aktuality.php s parametrem id_a, kde je uloženo identifikační číslo vybrané aktuality. Tato stránka je velmi podobná stránce Pridat_Aktualitu.php. Odlišnost je v zobrazení dat uložených v databázi do textových polí a po stisknutí tlačítka se provede v databázi změna záznamu místo jejich vložení.

8.11. Diskuze

V dnešní době bývá možnost vyjádření v internetové diskuzi téměř na každých internetových stránkách, proto jsem se ji do své aplikace rozhodl zahrnout také. Zobrazení příspěvků internetové diskuze jsem naprogramoval ve stránce Forum.php., která slouží pro zobrazení databázového dotazu z tabulky Diskuze. Tato stránka umožňuje výpis pouze osmi příspěvků na stránku. Je-li počet příspěvků větší, je pod příspěvky vytvořen odkaz na další stránku. Odkaz je předáván na stránku Forum.php s proměnou v URL adrese.

Ukázka kódu ze stránky Forum.php:

```
$pomer = $pocet_zaznamu/$zobrazeni;
$zbytek = $pocet_zaznamu % $zobrazeni;
if ($zbytek <> 0)
  {
  $pomer++;
  }
for ($stranka=1;$stranka<=$pomer;$stranka++)
{
  echo "<a href='index.php?str=Diskuze&amp;stranka=$stranka'>$stranka</a> ";
}
```

Pro přidávání do diskuze je vytvořena stránka Pridat_Forum.php. Zde je formulář pro vložení přezdívky, předmětu a textu. Pole pro text je ošetřeno pomocí již zmiňovaného Javascriptu. Jsou-li vložená data v pořádku a uživatel potvrdí vložení tlačítkem „Vložit“, dojde k volání databázové funkce pro vložení dat na této stránce.

Možnost smazání příspěvku zamezuje tvorbu vulgarismů v diskuzi, toto oprávnění má pouze uživatel s administrátorským účtem. Na stránce Smazat_Forum.php se zobrazí tabulka se sedmi příspěvky. Poslední sloupec tabulky obsahuje zaškrtávací pole pro vybrání příspěvku pro smazání. Pod tabulkou se nachází tlačítko, které po stisknutí smaže příspěvek v databázi.

8.12. Mapa stránek

Pro rychlou orientaci v systému je vytvořena mapa stránek (Mapa_Stranek.php). Tato stránka zobrazuje uživatelské volby v systému. Každý uživatelský typ má vlastní mapu stránek.

9. Závěr

Cílem této práce bylo vytvořit rezervační systém sportovního centra. K vytvoření byl využit jazyk PHP a databázový server MySQL. Webová aplikace je validní, tedy splňuje technická pravidla webových stránek.

Systém rezervací je spolehlivě funkční. Aplikace umožňuje administrátorovi spravovat jakoukoli část dat vložených do databáze. Návštěvník si může prohlédnout vytíženost jakéhokoliv sportovního kurtu. Registrovaný uživatel má možnost vytvoření online rezervace.

Po srovnání s ostatními sportovními centry, která disponují rezervačním systémem, se domnívám, že jsem vytvořil konkurenceschopnou aplikaci. Aplikace neřeší situaci, kdy je ve sportovním centru zaveden systém přeplaceného kreditu, k jehož odečítání dochází po využití rezervace. Tuto možnost ale většina sportovních center nevyužívá. Systém s předplaceným kreditem by bylo do mnou vytvořené aplikace možno doimplementovat.

V teoretické části jsem se dozvěděl mnoho nových informací o útocích do internetových aplikací a seznámil jsem se s možnostmi obrany proti těmto útokům. Pravidla obrany aplikace jsem využil při implementaci ve své bakalářské práci.

Zdroje informací

[1] Cross-site scripting. *Wikipedie, otevřená encyklopedie* [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Cross-site_scripting>.

[2] GLEMSER, Tobias. SQL Injection útoky na PHP a MySQL. Hakin9 [online]. 2005, č. 3 [cit. 2009-07-07]. Dostupný z WWW: <http://www.hakin9.org/upload/hakin9/PDFVersion/sql_cz.pdf>.

[3] MySQL - SQL Injection Prevention [online]. 2003-2008 [cit. 2009-07-09]. Dostupný z WWW: <http://www.tizag.com/mysqlTutorial/mysql-php-sql-injection.php>.

[4] SQL Injection. MySQL [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://dev.mysql.com/tech-resources/articles/guide-to-php-security-ch3.pdf>.

[5] SQL injection. Wikipedie, otevřená encyklopedie [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://en.wikipedia.org/wiki/SQL_injection>.

[6] VRÁNA, Jakub. Addslashes [online]. 2005 [cit. 2009-07-07]. Dostupný z WWW: <http://webtest.spstrutnov.cz/~aj-net/data/links/php_manual_cs/function.addslashes.html>.

[7] VRÁNA, Jakub. Obrana proti SQL Injection [online]. 2005-2008 [cit. 2009-07-07]. Dostupný z WWW: <http://php.vrana.cz/obrana-proti-sql-injection.php>.

[8] VEČEŘA, Zdeněk. Jak na to: SQL injection, magic_quotes_gpc, addslashes() a stripslashes() [online]. 2008-2009 [cit. 2009-07-07]. Dostupný z WWW: <http://blog.zdenekvecera.cz/item/jak-na-to-sql-injection-magic_quotes_gpc-addslashes-a-stripslashes>.

[9] Vyhledávače, SQL injection a ideální nástroje hackerů. POOH.CZ [online]. 2005 [cit. 2009-07-07]. Dostupný z WWW: <http://www.pooh.cz/pooh/a.asp?a=2012768>.

[10] Secure Hash Algorithm. Wikipedie, otevřená encyklopedie [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Secure_Hash_Algorithm>.

[11] Lámání hesel v praxi. Lupa [online]. 2005 [cit. 2009-07-07]. Dostupný z WWW: <http://www.lupa.cz/clanky/lamani-hesel-v-praxi-1/>.

[12] Rainbow table. Wikipedie, otevřená encyklopedie [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Rainbow_table>.

[13] VRÁNA, Jakub. Ukládání hesel [online]. 2005 [cit. 2009-07-07]. Dostupný z WWW: <http://php.vrana.cz/ukladani-hesel.php>.

[14] TICHÝ, Jan. Ukládání hesel v databázi [online]. 2007 [cit. 2009-07-07]. Dostupný z WWW: <http://www.phpguru.cz/clanky/hashovani-hesel>.

[15] TICHÝ, Jan. Sidejacking aneb nasloucháme v sití [online]. 2008 [cit. 2009-07-07]. Dostupný z WWW: <http://www.phpguru.cz/clanky/hashovani-hesel>.

[16] MALÝ, J., KACÁLEK, J. Zabezpečení webových aplikací I. - klientské skriptovací jazyky. Access server [online]. 2007 [cit. 2009-07-07]. Dostupný z WWW: <http://access.feld.cvut.cz/view.php?cisloclanku=2007090001>.

[17] MALÝ, J., KACÁLEK, J. Zabezpečení webových aplikací II. - databáze. Access server [online]. 2007 [cit. 2009-07-07]. Dostupný z WWW: <http://access.feld.cvut.cz/view.php?cisloclanku= 2007080002>.

[18] MALÝ, J., KACÁLEK, J. Zabezpečení webových aplikací III. – ostatní útoky a nastavení prostředí. Access server [online]. 2007 [cit. 2009-07-07]. Dostupný z WWW: <http://access.feld.cvut.cz/view.php?cisloclanku= 2007080003>.

[19] ULLMAN, Larry. PHP a MySQL. [s.l.] : [s.n.], 2004. 534 s.

[20] *Hcentrum* [online]. 2003 [cit. 2009-07-07]. Dostupný z WWW: <http://www.hcentrum.net/>.

[21] *SQUASHPARK Cibulka* [online]. 2009 [cit. 2009-07-07]. Dostupný z WWW: <http://www.squashpark.cz/>.

[22] *SportCentrum Ivanovice* [online]. 2008 [cit. 2009-07-09]. Dostupný z WWW: <http://www.sportcentrum-ivanovice.cz/html/cs/sport_centrum.phtml>.

[23] *Squash a fitness centrum Chomutov* [online]. 2008 [cit. 2009-07-08]. Dostupný z WWW: <http://www.squashcentrum-chomutov.cz/>.

[24] *Asport sportovní centrum* [online]. 2007 [cit. 2009-07-09]. Dostupný z WWW: <http://www.a-sport.cz/>.

Příloha A

Na přiloženém CD je v adresáři application celá adresářová struktura. Stačí obsah této složky zkopírovat do nainstalovaného prostředí. Dále se na CD nalézá soubor database.sql, kde je vyexportována databázová struktura a soubor install.txt popisující instalaci. Složka ADMIN obsahuje soubory, které jsou přístupné jen administrátorovi aplikace. Ve složce USER2 jsou soubory pro uživatele s rozšířenými právy. Dále následuje složka CSS, kde jsou soubory s kaskádovými styly.

MENU je složka, ve které jsou soubory s menu. Toto menu je vloženo po přihlášení do systému.

Složka SECURE obsahuje soubory s přístupem do databáze a některé vytvořené funkce.