

Informační systém firmy

Information system of company

Sedlář Michal

Bakalářská práce
2005



Univerzita Tomáše Bati ve Zlíně
Fakulta technologická

Vložit oficiální zadání diplomové/bakalářské práce

Na tomto místě bych rád poděkoval vedoucí bakalářské práce Ing. Zdeně Prokopové, CSc., za předmětné připomínky a rady při tvorbě webové aplikace a za cenné poznatky při sestavování tohoto textu.

Prohlašuji, že jsem na celé bakalářské práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně, 08. 06. 2005

.....

jméno diplomanta

Abstrakt

Tato bakalářská práce se v první části zabývá analýzou a stavem stávajícího informačního systému Střední odborné školy a středního odborného učiliště technického v Přerově. Ve druhé části popisuje možnosti řešení tvorby internetové aplikace, zabývá se skriptovacím jazykem PHP a relačními databázemi a to jak obecným návrhem datového typu tak praktickou částí SQL. Ve třetí části je zachyceno praktické řešení www stránek, kde je uvedeno jejich řešení, struktura databáze a popis všech tabulek, jejich atributů včetně použitých datových typů. V poslední části je řešena problematika implementace www stránek na webový server.

The first part of the presented bachelor thesis is dealing with an analysis of existing information system of Technical secondary school in Přerov. The second part describes possibilities to solve creation of internet application, scripting language PHP and relational databases. There were designed data types as well as structure of database in the SQL database system. In the third part are practical solutions of www pages, structure of database, description of tables and their attributes. The last part presents problematic of www pages implementation on to web server.

OBSAH

ÚVOD.....	6
1 ANALÝZA STÁVAJÍCÍHO STAVU SYSTÉMU	8
2 TEORIE.....	10
2.1 WWW A JAZYK HTML	10
2.1.1 World Wide Web.....	10
2.1.2 Jazyk HTML.....	10
2.1.3 CSS a XHTML	11
2.2 DATABÁZE A SQL	12
2.2.1 Historie databází	12
2.2.2 Pojem databáze	13
2.2.3 Datové modely.....	13
2.2.4 Normalizace dat.....	16
2.2.5 Historie jazyka SQL	17
2.2.6 Příkazy jazyka SQL.....	17
2.3 JAZYK PHP	21
2.3.1 Historie PHP	21
2.3.2 Co je to PHP?	21
2.3.3 Příkazy pro výpis dat.....	22
2.3.4 Proměnné.....	22
2.3.5 Větvění programu	22
2.3.6 Příkazy cyklu.....	23
3 VLASTNÍ TVORBA APLIKACE.....	24
3.1 TVORBA DATABÁZE	24
3.1.1 Návrh databáze	24
3.1.2 Popis tabulek	25
3.1.3 Vytvoření tabulek	29
3.1.4 Uložení dat do tabulek.....	31
3.2 TVORBA INTERNETOVÝCH STRÁNEK.....	33
3.2.1 Vytvoření veřejných internetových stránek.....	33
3.2.2 Vytvoření zabezpečených stránek a jejich připojení	41
4 IMPLEMENTACE STRÁNEK NA SERVER.....	50
4.1 WEBHOSTING STRÁNEK SOŠ A SOU TECHNICKÉHO V PŘEROVĚ	50
ZÁVĚR.....	52
SEZNAM POUŽITÝCH MATERIÁLŮ.....	54
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	55
SEZNAM OBRÁZKŮ.....	56

ÚVOD

Internet je médium, které obsahuje neskutečné množství informací. Můžeme na něm najít textové informace, grafické, zvukové i obrazové. Pro tuto svou různorodost je velmi oblíbený, každý si tam najde pouze to co ho zajímá. Díky prakticky neomezenému množství těchto informací a také množství počítačů, které jsou do něj připojeny, mnoho lidí ví co to Internet je a v hojné míře ho používají. Ale co by byl Internet bez nějaké uspořádanosti a přehlednosti těchto informací. Z tohoto důvodu jsou vytvářeny webové stránky. Tyto by měly být co nejpřehlednější a pokud možno co nejlákavější, aby se na ně uživatelé rádi vraceli. Avšak pouze podávání informací není tou jedinou službou Internetu. Můžeme ho využít například ke komunikaci mezi uživateli z různých koutů světa, toto je jeho velkou výhodou. Prakticky není omezen pouze na jistou část uživatelů, ale každý může komunikovat s každým. V poslední době dochází k velkému rozvoji obchodu přes Internet. Můžete klidně sedět doma u počítače a koupit si jakoukoliv věc, aniž byste museli vstát. I přes tyto velké možnosti jsou stále webové stránky službou nejpoužívanější. Výhodou je totiž velké množství poskytovaných informací za příznivou cenu.

Střední odborná škola a střední odborné učiliště technické v Přerově je jednou z několika středních škol v městě Přerově. Proto je velmi důležité, aby se co nejlépe prezentovala na veřejnosti a zaujala případné zájemce o studium. Existuje mnoho způsobů jak se prezentovat a snažit se zaujmout co nejširší okruh veřejnosti. Jedním z nejrozšířenějších způsobů poslední doby je již zmíněný Internet. Z tohoto důvodu je potřeba, aby škola měla na Internetu své vlastní webové stránky, které by ji řádným způsobem reprezentovaly a podávaly o ní co nejvíce informací. Protože boj o případné zájemce je čím dál těžší, musí být podávány co nejobjektivněji a nejlákavěji, aby zájemců oslovili co nejvíce. Trend jde však kupředu a už nezůstává jen u poskytování informací běžným uživatelům, kteří se chtějí dozvědět nějaké informace. Stránky začínají využívat i přímo studenti, rodiče a zaměstnanci k vnitřním záležitostem týkajících se výuky, prospěchu atp. Z tohoto důvodu bylo zapotřebí dát novou podobu stránkám a dát právě možnost k využití pro tyto záležitosti.

Byl jsem požádán, zda bych nemohl vytvořit tyto nové stránky, které by jen neposkytovali informace běžným návštěvníkům, ale mohli je také využívat právě rodiče a vyučující. Problematikou řešení internetových aplikací a stránek jsem se zabýval před mnoha lety. V té době bylo nemyslitelné vytvořit dynamické stránky, které se budou měnit podle akcí

uživatele. Od té doby samozřejmě doba pokročila a technologické možnosti také. Proto mě tato nabídka zaujala a bez většího přemýšlení jsem ji přijal. Začal jsem uvažovat nad tím jakým způsobem se budou stránky tvořit a co po nich budeme požadovat.

1 ANALÝZA STÁVAJÍCÍHO STAVU SYSTÉMU

Internet je medium, které se stále vyvíjí, stále přibývají nové rozhledy co vše se dá s pomocí internetu dosáhnout. Proto by i prezentace na Internetu měla jít kupředu a snažit se držet krok s moderními technologiemi. Dávno jsou pryč doby, kdy se používali statické stránky, které byly schopny jen podávat informace se kterými nebylo možno dále pracovat. A to je případ právě Střední odborné školy a středního odborného učiliště technického v Přerově.

První stránky této školy byly vytvořeny v roce 1996. Jednalo se o velmi jednoduché stránky, které však podávaly velmi přehledné a ucelené informace. V roce 1998 doznaly stránky mnoha změn. Tou hlavní bylo použití rámců v grafickém návrhu. To umožnilo použít v jedné části, konkrétně levé, menu s odkazy, pomocí kterých mohli návštěvníci najít spoustu informací o které měli zájem. Výhodou tohoto řešení bylo, že menu s odkazy zůstávalo stále na stejném místě, jen se měnily zobrazené informace, které uživatel požadoval. To přispělo k velmi přehlednému a ucelenému prohlížení stránek. Díky tomuto bylo možné přidávat mnoho dalších informací, které však přehlednost stránek žádným způsobem nenarušily. Další velkou a nezvratnou výhodou bylo jejich grafické řešení. Nejednalo se v žádném případě o řešení komplikované ani ničím průlomové. Avšak bylo velmi přehledné a příjemné. Až do roku 2004 se stránky žádným výrazným způsobem neměnily. Jedinými změnami byly aktualizace zpráv a množství poskytovaných informací. Během roku 2004 stránky doznaly mnoha změn a inovací, ale jednalo se především o změny po grafické stránce a opět jiné uspořádanosti. Podnětem pro tuto změnu byla především změna názvu školy ze stávajícího Integrovaná střední škola technická, Přerov na současný Střední odborná škola a odborné učiliště technické v Přerově (SOŠ a SOU technické v Přerově). Bylo upuštěno od designu pomocí rámců. Namísto toho jsou stránky tvořeny pomocí tabulek. Tento způsob je oproti rámcům lepší. A to především v tom, že máme větší možnosti v grafickém designu, jelikož jsou stránky členěny do více bloků. Bloky rozumíme řádky a sloupce tabulek. Odkazy jsou také řešeny novým způsobem. Sloupec s odkazy je tvořen obrázkem, který obsahuje předkreslená tlačítka. Funkce odkazu je přidělena těmto tlačítkům pomocí příkazu MAP, který musí obsahovat souřadnice na kterých se má obrázek (tlačítko) chovat jako odkaz. Toto řešení má však několik nevýhod. Pokud bychom chtěli stránky nějakým způsobem rozšiřovat a přidat do menu další odkaz, museli bychom překreslit obrázek s tlačítky. Další nevýhodou je

velikost souborů, které stránky tvoří a namáhavost na velké množství stahovaných obrázků, které tvoří design stránek. Avšak nejen změny v grafickém uspořádání byly těmi jedinými. Výraznou novinkou je možnost prohlížení těch nejdůležitějších informací v několika světových jazycích, konkrétně angličtině a němčině. Ovšem i přes tyto rozšíření mají stránky již od dob jejich prvních provedení jednu shodnou vlastnost. A to, že jsou stále ještě statické. Nedá se s nimi nadále pracovat a uživatel nemá jinou možnost než si stránky jen prohlédnout a zjistit několik nových informací o dění na této škole. Proto uživatelé, kterými jsou zákonní zástupci dětí, které školu již navštěvují, nemají nijak velkou motivaci stránky navštěvovat. V zásadě se na nich dozvědí velmi málo nových informací, které by jim byly k užitku.

2 TEORIE

2.1 WWW a jazyk HTML

Internet je celosvětová počítačová síť, která sdružuje veliké množství počítačů. Ty jsou mezi sebou propojovány buďto velkým množstvím kabelů nebo, v dnešní době již velmi rozšířeným, bezdrátovým spojením. Každým okamžikem jsou další a další počítače přidávány. Každý počítač připojený do Internetu může komunikovat s kterýmkoliv jiným, který mu to nezakáže. Běžný uživatel si toto pod pojmem Internet však nepředstaví. Napadnou ho především služby, které díky němu může využívat např. komunikace, obchodování apod. Internet je tedy neomezeným zdrojem informací. Ty mohou být ve formě textové, grafické, zvukové, ale i obrazové.

2.1.1 World Wide Web

World Wide Web je ve zkratce známo pod názvem WWW. Jedná se o službu Internetu, kterou však nejlépe vystihuje zkrácený výraz Web (z angličtiny „pavučina“). Tato služba umožňuje prohlížet pomocí internetového prohlížeče dokumenty uložené na serveru. Teprve použití této technologie umožnilo zpřístupnit Internet široké veřejnosti a možnost umístit na něj a prohlížet velké množství informací. Web není však omezen pouze jedním jediným dokumentem, pomocí speciálních prvků (odkazů) můžeme jich spojovat neomezené množství. Jedná se tedy o velikou pavučinu, která může nabývat nemalých rozměrů. Stačí myší kliknout na odkaz a ten nás zavede tam, kam směřuje. Takovým odkazům se říká hyperlinky. Což znamená, že se nejedná o obyčejný text, který bychom museli číst postupně, ale můžeme v něm listovat na přeskáčku. Od doby co byl Web spuštěn doznal mnohých změn. S příchodem nových technologií se stále vyvíjí. Už nemusí být stránky pouze textového a obrazového charakteru, mohou obsahovat různé animace, zvuky, dokonce i reagovat na pokyny uživatele.

2.1.2 Jazyk HTML

HTML (Hypertext Markup Language) je značkovací jazyk, pomocí kterého vytváříme zdrojový kód stránky. Nejčastější příponou souborů s jazykem HTML jsou *.html* nebo *.htm*. V nich pomocí značek (tagů) určujeme co má dokument obsahovat, jak má být jeho obsah formátován. Aby se odlišily značky a normální text jsou značky uzavřeny mezi

znaky $\langle \rangle$. Tyto značky mohou být párové a nepárové. Liší se od sebe tím, že párové mají začínající a ukončovací značku. Tyto dvě jsou od sebe odlišné pouze tím, že ukončovací značce se na její začátek zapisuje lomítko $\langle I \rangle \langle /I \rangle$. Většina používaných značek je párových, existuje však i několik nepárových, které se hojně používají např. $\langle BR \rangle$. Každý HTML dokument musí začínat $\langle HTML \rangle$ a končit $\langle /HTML \rangle$ a skládat se ze tří částí. První je určení verze jazyka. Tato informace slouží především prohlížeči, aby věděl podle jakých pravidel má na stránku pohlížet. Druhá je nastavení hlavičky. Tato slouží k poskytnutí doplňujících informací o použité znakové sadě, titulku stránky apod. Tyto informace se na stránce nezobrazují. Pouze text mezi značkami $\langle TITLE \rangle$ a $\langle /TITLE \rangle$ je zobrazen v záhlaví stránky v prohlížeči. Poslední částí je vlastní tělo dokumentu. Uzavírá se mezi $\langle BODY \rangle$ a $\langle /BODY \rangle$. Tuto část prohlížeč provádí a převádí takto napsaný kód do skutečné podoby na stránce. Pro formátování textu slouží značky $\langle FONT \rangle$ a $\langle /FONT \rangle$, ty mohou být dále rozšířeny o další vlastnosti jako barva, velikost, styl atd. Nový odstavec začínáme značkou $\langle P \rangle$ a ukončíme $\langle /P \rangle$. Tabulku vytvoříme pomocí $\langle TABLE \rangle \langle /TABLE \rangle$. Uvnitř můžeme pak libovolně použít značky pro vytvoření řádků $\langle TR \rangle$ a $\langle /TR \rangle$ a sloupců $\langle TD \rangle$ a $\langle /TD \rangle$.

2.1.3 CSS a XHTML

Novou možností, jak formátovat své stránky je použití CSS (Cascading style sheet) kaskádových stylů. Jedná se o oddělení formátování od samotného obsahu. To znamená, že textová informace bude dostupná i bez jakéhokoliv formátování. Nejpoužívanějším způsobem je uvádět CSS vlastnosti do externího souboru, který se potom připojí k dané stránce. Zápis $\langle link rel="stylesheet" type="text/css" href="externi_soubor.css" \rangle$ do hlavičky souboru zajistí připojení externího souboru se stylpisem ke stránce. Výhodou tohoto řešení je, že nemusíme stylpis uvádět na každé stránce. Jediným souborem jsme schopni přeformátovat veškeré stránky našeho webu. Druhý způsob, méně používaný, je ten, kdy v hlavičce dokumentu vymezíme prostor pro definování stylu párovými značkami $\langle style \rangle$ a $\langle /style \rangle$. Velkou výhodou CSS je jednoduchá úprava formátování. Přípona *.css* je příponou těchto externích souborů. S kaskádovými styly CSS velmi dobře spolupracuje XHTML kód. Vychází z běžného HTML a XML, avšak jsou zde důležité odlišnosti. Tento kód je tzv. *case sensitive*, zjednodušeně řečeno to znamená, že $\langle h1 \rangle$ a $\langle H1 \rangle$ nejsou stejné zápisy. V XHTML se v zásadě všechny tagy píšou malými písmeny. Počáteční a koncový tag tvoří tzv. *element*. Každý element musí být uzavřen, tedy i u nepárových tagů musí být

uzavírací značka ve formě `<br/>`. Tagy se nesmějí za žádných okolností křížit. XHTML je děleno do několika norem podle přísnosti dodržování pravidel formátování. Ve striktní normě jsou veškeré tagy pro formátování zakázány a veškeré formátování je již přenecháno CSS. Existují tyto čtyři XHTML normy.

- XHTML 1.0 Transitional – nejměkčí norma, obsahuje možnosti, které jsou v dalších zakázány. Jedná se především o přechodnou normu z HTML na XHTML.
- XHTML 1.0 Frameset – u této normy máme ještě možnost použití rámců při tvorbě webu. To z toho důvodu, že se někde bez rámců neobejít nelze.
- XHTML 1.0 Strict – nejprísnejší norma z norem třídy 1.0. Zakazuje prakticky jakékoliv formátování stránky.
- XHTML 1.1 – prakticky se neliší od normy XHTML 1.0 Strict. Zakazuje jakékoliv formátování stránky, to vše přenechává na CSS.

Teorie WWW a HTML byla čerpána s následujícími zdroji [2], [3], [4], [11] a [12].

2.2 Databáze a SQL

2.2.1 Historie databází

Počátek databází a způsobu zpracování dat s pomocí databází spadá do 60. let minulého století. Zasloužila se o to firma IBM, která vytvořila první databázový systém, který byl ojedinělý především tím, že používal hierarchické členění dat. To o deset let položilo základ tvorbě databází založených na hierarchických strukturách, které byly různými způsoby propojovány. Přestože byly schopny řešit mnohé složité problémy, začali se v polovině 70. let objevovat relační databázové systémy. Tyto systémy využívaly pro vztah mezi daty relační algebru. V 80. letech se velice rychle začal prosazovat jazyk SQL, který je s relačními databázemi spjat až do současnosti. V současné době se však vývoj posunul směrem k objektovému přístupu zpracování dat tzv. *objektovým databázím*. Jejich předností je možnost uchování širokého spektra dat, jako textová data, obrazová nebo zvuková. V oblasti implementace se objevují dva trendy. První vychází z plně objektových databází, jedná se především o nové systémy. Druhý trend je zaznamenán spíše v zapracování nových přístupů do již existujících databází. Jelikož tento způsob stále

Síťový model byl obvyklý v 70. letech na velkých sálových počítačích. Základní myšlenka spočívá v existenci velkého množství samostatných segmentů s rozdílnou strukturou. Mezi těmito segmenty existují vzájemné vazby, které tvoří síť. Pomocí tohoto modelu je velmi obtížné a náročné udržovat rozsáhlejší databáze. Značnou nevýhodou je také to, že v případě potřeby se obtížně mění její struktura.

Hierarchický model dat

U této jsou data organizována do tzv. stromové struktury. Každý záznam představuje pomyslný uzel v této stromové struktuře. Vzájemný vztah mezi záznamy je typu rodič/potomek. Jeden rodič může mít několik potomků. Potomci mohou dědit některé vlastnosti od svého rodiče a pak je předávat dál svým potomkům. Nalezení dat v hierarchické struktuře probíhá směrem k potomkům a také směrem k rodičům. Nevýhodou je řešení vztahu M:N, který se řeší např. pomocí virtuálního spojení.

Objektový model dat

V posledních letech se začínají používat objektové databáze. Jedná se o systém podobný síťovému modelu. Rozdílem je to, že jednotlivá data popisují tzv. objekty, které mohou mít různou strukturu a mohou také obsahovat metody, tedy odkazy na úseky s nimi svázané. Mezi objekty pak existují vazby připomínající síť ze síťového modelu. Nejpoužívanějším modelem je však stále relační model, avšak existuje něco jako mezistupeň mezi relačním a objektovým modelem. Tento se jmenuje objektově-relační model.

Relační model dat

Tento model je v dnešní době nejpoužívanějším z těchto modelů. Relační datový model sdružuje data do tzv. *entit*. Tyto entity tvoří základ relační databáze. Každá z entit obsahuje *atributy*, které udávají vlastnosti entity. Mezi entitami mohou být různé *vztahy* [13].

Entity

Entitou rozumíme cokoliv o čem potřebujeme uchovávat informace. Nejčastěji je v relačních databázích entita prezentována tabulkou, můžeme mít entitu např. student, město atp. V datovém modelu musí mít každá entita nějaký identifikátor, který zajistí jednoznačnou identifikaci záznamů v entitě. Tento identifikátor se v databázích označuje jako primární klíč.

Atributy

Obecně si lze pod pojmem atribut představit vlastnost entity. Například u entity *student* může být atribut příjmení, jméno apod. Atribut je tedy prvkem, který blíže definuje entitu.

Dělíme je na:

- Základní, tento atribut není odvoditelný z jiných atributů
- Odvoditelné, jsou to atributy jejichž hodnoty jsou nějakým způsobem odvozené ze základního nebo odvoditelného atributu např. nějakou aritmetickou operací

Velmi významnou vlastností atributů je *volitelnost*. V rámci volitelnosti může atribut nabývat dvou hodnot, kterých mohou dosáhnout. Pokud žádný záznam (řádek tabulky) nesmí být prázdný hovoříme o tzv. *totálním atributu*. Příkladem může být rodné číslo. Je jasné, že každý člověk má rodné číslo a proto nemůže být řádek s touto hodnotou prázdný. Naopak opačným příkladem je tzv. *parciální atribut*. Říká, že není nutností, aby v tabulce byly všechny hodnoty.

Vztahy

Jednotlivé entity mohou také vstupovat do vzájemných vztahů. Entity, které do těchto vztahů vstupují nazýváme účastníky vztahu. Nejčastějším případem jsou vztahy dvou entit, které se také nazývají *binární vztahy*. V případě, že do vztahu vstupuje entit více nazývají se tyto vztahy *vícenásobné*.

Důležité charakteristiky:

- Kardinalita, jedná se o minimální nebo maximální počet entit v rámci jednoho vztahu. Nejčastěji se označuje 1:1, 1:N, N:M
- Exkluzivita, udává to, že pro jednu entitu může být pouze jeden ze vztahů
- Parcialita, volitelnost znamená, že vztah se nemusí vždy týkat všech výskytů

Vztahy 1:1 nejsou moc časté, přesto se občas vyskytují. Nejčastěji se s těmito vztahy setkáváme tam, kde je potřeba snížit počet atributů v relaci.

Nejběžnějším vztahem mezi entitami je 1:N, jedna k více. V tomto vztahu můžeme jeden záznam s první entity asociovat s několika a nebo žádným záznamem z druhé entity. Totéž platí v případě N:1.

Poměrně často se také vyskytuje vztah N:M, tedy více k více. Příkladem může být například to, že každý student může mít zapsáno několik předmětů a zároveň na jeden

předmět může být zapsáno studentů více. V relačních databázích se však toto spojení nedá řešit přímo. Řešením je tzv. *spojovací tabulka*, která je prostředníkem mezi účastníky vztahu.

2.2.4 Normalizace dat

Původně byla tato technika navržena zejména v souvislosti s relačními databázemi. Cílem je dosáhnout co nejpřesnější podoby reálných entit. Zejména se používá pro ověření návrhu logické struktury datového modelu. Soustřeďuje se na odstranění těch záznamů, které by mohly narušit konzistenci datové základny.

Normalizace dat představuje omezující pravidla aplikovaná na datové struktury, jako jsou tabulky nebo atributy. Každé z pravidel je soustřeďováno na odstranění některého z nedostatků, duplicit nebo nekonzistencí. Existuje prakticky jen jedno řešení, jak odstranit nedostatky datových struktur. Tím řešením je rozdělit větší datové struktury na více menších.

Existuje celkem 7 tzv. normálních forem, tedy rad či doporučení, které je vhodné dodržet pro to, aby databáze tvořená relačními tabulkami byla efektivní a přitom s ní bylo možné snadno pracovat. Praktický význam však mají především první tři formy.

První normální forma

V případě této normy platí to, že se v datové struktuře nesmí opakovat stejné položky. Je velmi důležité, aby každý atribut byl skalární tzn. aby nebyl dále dělitelný. Například jméno a příjmení nesmí být určeny jako jeden atribut, ale musí být zvlášť atribut jméno a zvlášť příjmení.

Druhá normální forma

Tato forma řeší problematiku primárních klíčů. Primární klíč jednoznačně odlišuje jednotlivé řádky tabulky od sebe a to tak, že stejný řádek se v dané databázi nesmí vícekrát objevit. Příkladem primárního klíče může být například rodné číslo, číslo zaměstnance apod. V mnoha případech však primární klíč v relaci neexistuje, je však možné primární klíč uměle vyrobit (např. číslování řádek tabulky).

Třetí normální forma

Pokud tabulky spadají do této formy znamená to, že neobsahují redundantní (nadbytečné) data. Nadbytečná data jsou především ta, která se v tabulce stále opakují. Redundanci lze

z tabulky odstranit tím způsobem, že tabulku rozdělíme na několik tabulek, které jsou mezi sebou propojeny a přijímají data jedna od druhé [13].

2.2.5 Historie jazyka SQL

Počátky jazyka SQL (Structured Query Language) spadají do 70. let, kdy byl vyvinut společností IBM jazyk SEQUEL (Structured English Query Language) s úmyslem přivést na svět účinný nástroj pro manipulaci s daty uloženými v relačních databázích. Do té doby nebyla práce s daty snadná. Prvořadým cílem nového jazyka bylo jeho snadné použití a syntaxe blízká anglickému jazyku, aby s ním mohla komunikovat spousta uživatelů. V začátcích to neměl jazyk SEQUEL vůbec jednoduché. Vše se ovšem změnilo, když byl poprvé použit jako jazyk pro přístup k datům především v databázích *Oracle* firmy *Relation Software, Inc.* (dnešní *Oracle Corporation*) a nebo i jiných firem. Mezi nejznámější patří především *Progress*, *Informix* a *SyBase*. Později byl přejmenován na dnes známé SQL. Prakticky od této doby se začal zabydlovat skoro v každém databázovém systému. Reakcí na to se pokusil Americký standardizační institut (ANSI) vytvořit jazyk RDL. Ten se však neprosadil, protože samotné SQL se stalo standardem. To bylo důvodem, proč ANSI v roce 1986 založilo nový standard vycházejícího z tohoto jazyku, dostal název SQL86.

V průběhu následujících let se ukázalo, že tento standard není dostačující. Objevilo se spousta problémů, které SQL86 neumělo řešit a nebyly v něm obsaženy některé důležité prvky. Jednalo se zejména o integritu dat. Proto byl v roce 1992 přijat nový standard SQL-92 označovaný také jako SQL2. Základním rozšířením bylo přidání primárních klíčů a definice integritních omezení dat v tabulkách. Zatím nejnovějším standardem je SQL3, které se zaměřuje na objektové databázové systémy.

Prakticky každá databáze obsahuje standardy, ale obvykle nejsou implementovány vždy všechny požadavky normy. Proto je důležité vědět, že jazyk SQL je sice universálním nástrojem k ovládání mnohých databází, ale jen skupinou základních a nejpoužívanějších příkazů. Poznatky této kapitoly byly čerpány z [6] , [9] a [13].

2.2.6 Příkazy jazyka SQL

Příkazy jazyku SQL lze rozdělit do několika dílčích skupin. Konkrétně rozlišujeme dvě základní skupiny příkazů. První jsou příkazy pro definici dat. Těmito příkazy vytváříme

struktury databáze - tabulky, indexy a další. Takto vytvořené struktury můžeme přidávat, měnit a mazat. Tuto skupinu příkazů označujeme DDL (*Data Definition Language*). Druhou skupinu značíme DML (*Data Manipulation Language*). Jsou to příkazy pro získání dat z databáze a pro jejich úpravy.

Vzhledem k tomu, že jazyk SQL má určitá pravopisná pravidla, existuje také slovní zásoba slov používaných v jazyce SQL. Tato označujeme jako slova vyhrazená, protože mají zvláštní význam. Prvním slovem v každém příkazu musí být vždy vyhrazené slovo. Obvykle tohle slovo udává co se požaduje po databázi. Slova, která následují za vyhrazeným slovem jsou argumenty, které určují parametry toho co se má udělat.

Datový typ

Určuje vlastnost každého sloupce. Rozeznáváme tři základní typy *číslo*, *text*, *datum a čas*.

Další rozšíření těchto základních typů je uvedeno v mnoha odborných literaturách např. [5], [6].

Integritní omezení

Integritní omezení jsou taková, která se kontrolují na straně serveru. Při návrhu tabulky můžeme rozhodnout o určitých omezeních jednotlivých položek.

Existují tyto základní integritní omezení:

- PRIMARY KEY – primární klíč, jeho hodnota musí být jedinečná
- UNIQUE – unikátní hodnota, tato zamezuje duplikátní položky
- DEFAULT <hodnota> - pokud uživatel nezadal žádnou hodnotu, sama se nastaví předdefinovaná
- NOT NULL – udává, že hodnota musí být nutně zadána (nesmí být prázdná)

Příkazy pro definici dat (DDL)

Do této skupiny příkazů patří:

- CREATE – vytváření nových objektů
- ALTER – změny existujících objektů
- DROP – odstraňování objektů

CREATE

Tento příkaz slouží především k vytvoření tabulky.

ALTER

Po nějaké době, kdy je naše databáze v provozu, můžeme usoudit, že bychom chtěli upravovat formát našich tabulek (přidávat sloupce, odebírat, měnit datové typy apod.). K tomu slouží příkaz *Alter*. Spolu s tímto jsou spjaty i příkazy na to co chceme vlastně udělat. Můžeme přidávat nové sloupce, pokud bude třeba nebo měnit vlastnosti již vytvořených. Pro vytvoření nového sloupce bychom použili příkaz ADD. Jestliže bychom potřebovali změnit vlastnosti již vytvořených sloupců, použije se k tomu příkazu CHANGE.

Možností pro které se dá příkaz *Alter* použít je samozřejmě mnohem více např. přejmenování tabulky, sloupce, mazání sloupce apod.

DROP

Slouží k mazání již vytvořených tabulek. Použití příkazu *Drop* je však do jisté míry nebezpečné. Měli bychom ho použít až po dobrém zvážení. Většina systémů se totiž neptá, jestli si přejeme tabulku opravdu odstranit, ale rovnou ji odstraní. Dalším nebezpečím je to, že pokud jsou v dané tabulce uložena nějaká data, tak o tyto data přijdeme.

Příkazy pro manipulaci s daty (DML)

Do této kategorie příkazů patří zejména tyto:

- SELECT – vybírá data z databáze, umožňuje výběr podmnožiny a řazení dat
- INSERT – vkládá do databáze nová data
- UPDATE – umožňuje opravovat data v databázi
- DELETE – odstraňuje data (záznamy) z databáze

SELECT

Jedná se o jeden z nejdůležitějších příkazů jazyku SQL. Slouží k výběru záznamů z databáze. Tento příkaz se sestává ze dvou základních klauzulí.

- **SELECT**, tato zajistí získání požadovaných dat (sloupců) z databáze. Pokud si přejeme získat z databáze veškerá data (sloupce) použijeme k tomu symbol *
- **FROM**, ta nám říká odkud, z které tabulky, chceme tyto data získat

Tabulka však může obsahovat velké množství dat. Pokud si nepřejeme, aby byly z databáze získány veškeré informace, ale jen žádané, použijeme namísto znaku * přímo názvy těch atributů, které si přejeme vypsát. V případě, že bychom nechtěli vypisovat všechny řádky použijeme navíc podmínky **WHERE**, která nám zajistí vypsání právě jen námi požadovaných řádků. Dalším důležitým rozšířením příkazu *Select* je rozšíření o **ORDER BY**. To nám říká, že bychom chtěli získané řádky seřadit. Získaná data můžeme seřadit sestupně pomocí modifikátoru **DESC** a nebo vzestupně pomocí **ASC**.

INSERT

Tento příkaz nám umožňuje do již existující tabulky vkládat nové záznamy (data). Celý příkaz se skládá ze dvou hlavních částí. *Insert into* (znamená vložit kam) a *Values* (jaká data).

UPDATE

Slouží k aktualizaci záznamů v tabulce. Svou syntaxí je podobný příkazu *Insert*. Je zde však jeden zásadní rozdíl. A to ten, že po jeho použití se pozmění všechny záznamy v tabulce. Pokud bychom nezadali podmínku, kde všude chceme hodnotu v atributu měnit změnila by se u všech záznamů. Proto je na místě použít podmínku **WHERE**, která nám vybere pouze ty řádky, které chceme opravdu měnit.

DELETE

Tento příkaz je pravým opakem příkazu *Insert*. Slouží k odstranění údajů z databáze. Z tabulky může vymazat úplně všechny údaje a nebo také jen ty, které si sami vybereme pomocí již zmíněné podmínky **WHERE**. Pokud tuto podmínku neuvedeme odstraní z tabulky veškeré údaje, ale neodstraní tabulku samotnou.

Praktická použití příkazů jazyka SQL můžeme nalézt v mnoha literaturách např. [6] nebo na internetu např. [9].

2.3 Jazyk PHP

2.3.1 Historie PHP

Historie jazyku PHP se začala psát roku 1994, kdy Rasmus Lerdorf napsal jednoduchý systém pro evidenci přístupu ke svým stránkám. Nejprve tento systém napsal v Perlu, ale pro snížení zatížení serveru ho přepsal do jazyku C. Tento jeho systém se stal brzy populárním a proto ho začal rozšiřovat a dal mu názvem Personal Home Page Tools. V roce 1997, po rozšíření tohoto jazyku o možnost pracovat s SQL dotazy, zobrazování jejich výsledku na stránkách a práci s formuláři získal tento jazyk označení PHP/FI 2.0. Pod tímto názvem začal být šířen, jako jednoduchý programovací jazyk, do celého světa. Nevýhodou této verze však bylo, že pracovala pouze na operačním systému UNIX. Proto bylo v roce 1998 vytvořeno PHP 3.0, které již pracovalo i na operačních systémech Windows. Po této verzi přišla nová s označením PHP 4.0. Tato se vyznačuje podstatně vyšším výkonem a spolehlivostí. Nejnovější verzí je verze PHP 5.0, která má vylepšené jádro ZEND Engine.

2.3.2 Co je to PHP?

PHP je univerzální skriptovací jazyk, který se v posledních letech stává velice oblíbeným. Je obzvlášť vhodný pro vývoj webových aplikací, protože ho lze použít přímo v kódu HTML. Jeho základní odlišností od jazyků jako je např. JavaScript je ten, že PHP se provádí na straně serveru naproti tomu JavaScript se provádí na straně klienta. PHP je na serveru závislé, protože na něm běží jeho interpreter, který skripty vykonává. To že je na straně serveru má své výhody, ale samozřejmě také nevýhody. Výhodou je to, že ke zdrojovému kódu se nedostane nikdo jiný než jeho autor. Pokud si uživatel nechá zobrazit zdrojový kód, tak se mu ukáže pouze kód co je na straně klienta, ten na straně serveru mu je nedostupný. Naproti tomu JavaScript má výhodu v jeho dynamičnosti a reagování na změny jako je například pohyb kurzoru apod. Z tohoto důvodu je velmi vhodné kombinovat PHP s JavaScriptem nebo mu podobným. Samotný PHP kód se zapisuje přímo do HTML kódu. Musí v něm však být jednoznačně oddělen. Pro označení počátku kódu PHP se používá kombinace znaků `<? nebo <?php` a pro jeho konec `?>`. Pak tento soubor má příponu `*.PHP`, `*.PHP3` nebo `*.PHP4`. Interpreter pak pracuje tak, že HTML příkazy rovnou ukládá do výsledné stránky. Pokud během tohoto procesu narazí na skript PHP, tak

ho provede a jeho výsledek zapíše do stránky HTML. Skriptovací jazyk PHP poskytuje velkou podporu databází (MySQL, MS SQL server, Oracle, Informix a mnoho dalších). PHP může také sloužit jako brána k mnohým běžným službám, které jsou na internetu běžně používané. Jako například, FTP, POP3, SMTP, HTTP atd. Nezvratnou výhodou je to, že kód stránek můžeme psát v kterémkoliv textovém editoru. Nejjednodušším je poznámkový blok, nemusíme tedy utrácet spousty peněz za drahé programy renomovaných společností.

2.3.3 Příkazy pro výpis dat

Nejčastěji používané příkazy jsou *Echo* a *Print*. Oba slouží k výpisu dat v prohlížeči. Hlavním rozdílem mezi nimi je fakt, že *Print* je na rozdíl od *Echa* funkce s platnou návratovou hodnotou. Pokud *Print* proběhne v pořádku vrací hodnotu 1, v opačném případě 0. Naproti tomu *Echo* nevrací žádnou hodnotu, jen zobrazí výsledek na obrazovku. Z toho důvodu, že *Echo* nevrací žádnou hodnotu, je rychlejší.

2.3.4 Proměnné

Jedná se o místo v paměti, které odkazuje na pojmenovanou hodnotu. Pokud si například potřebujeme po nějakou dobu, po kterou se bude provádět část kódu, zapamatovat určitou hodnotu, uložíme ji do proměnné a tu libovolně pojmenujeme. Každá proměnná musí začínat znakem \$. V PHP není třeba proměnné nijak předem deklarovat. Vytvoří se v okamžiku jejího použití ve skriptu.

2.3.5 Větvení programu

Jedná se o příkazy, které slouží k porovnávání hodnot proměnných a podle výsledku porovnání určují, jak má program reagovat. Z tohoto důvodu hovoříme o větvení.

Příkaz IF

Tento použijeme, pokud potřebujeme nějakou určitou hodnotu otestovat, nebo zjistit jestli se od této hodnoty liší. Nejjednodušším použitím je provedení části kódu po splnění jisté podmínky. Podmínku *If* můžeme rozšířit o další větve a tou je použití *Else*. Doslovný význam znamená „jinak“. Takže to vyřeší problém, když nebude splněna podmínka *If* provede se tedy *Else*.

Příkaz SWITCH

Někdy je zapotřebí, abychom vybrali z většího množství větví programů než nám dává možnost If / else. Proto místo použití několikrát za sebou těchto příkazů použijeme *Switch*.

2.3.6 Příkazy cyklu

Cyklem rozumíme sekvenci příkazů, které se provádí v jisté smyčce opakovaně. Ovšem každý cyklus musí jednou skončit, a tak je potřeba programu sdělit, kolikrát se má cyklus provést a nebo se má opakování vázat na splnění určité podmínky.

Cyklus WHILE

Jedná se o cyklus s podmínkou na začátku. Dokud je splněna podmínka na začátku, cyklus se provádí. Pokud nejsme schopni již zajistit splnění podmínky, ukončí se.

Cyklus FOR

Tento cyklus je cyklem se stanoveným počtem opakování. Jedná se o cyklus s nejnáročnějším zápisem. Obsahuje tři parametry. Počáteční řídicí proměnnou (počítadlo), logickou podmínku (musí být splněna, aby se cyklus provedl) a třetím je navýšení řídicí proměnné. Většinou je navýšení voleno o 1.

Praktická použití těchto příkazů najdeme např. v [1], [5] a mnoha dalších.

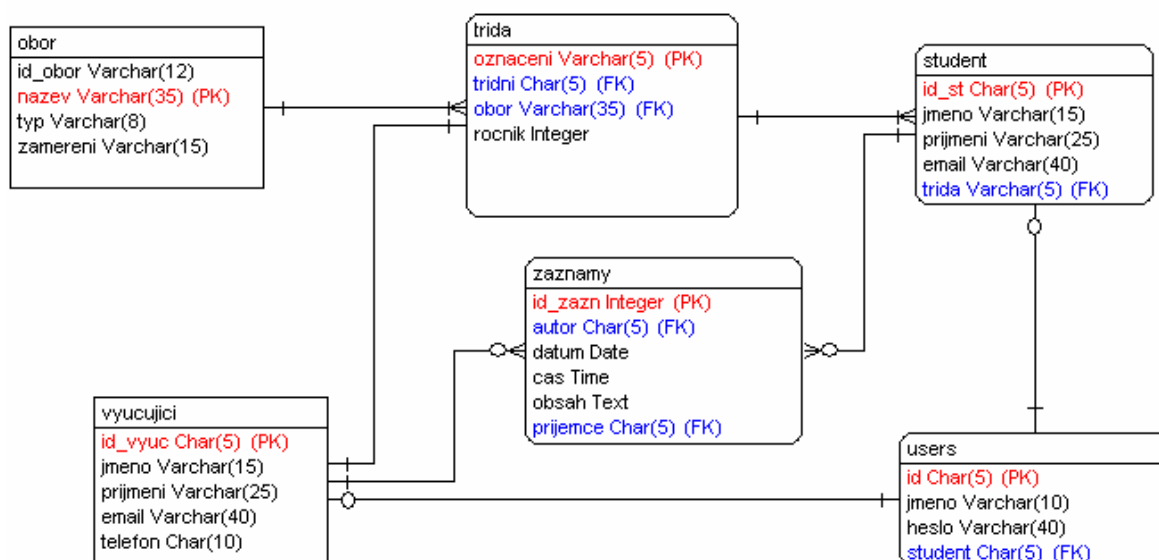
3 VLASTNÍ TVORBA APLIKACE

Celou aplikaci tvoří dva základní stavební kameny: databáze a vlastní internetové stránky. Stránky jsou tvořeny pomocí jazyka HTML v kombinaci se skriptovacím jazykem PHP. Tento skriptovací jazyk je použit z toho důvodu, že velmi dobře podporuje relační databáze. Vlastní databáze tohoto systému je vytvořena v aplikaci MySQL.

3.1 Tvorba databáze

3.1.1 Návrh databáze

Jako první bylo zapotřebí navrhnout strukturu databáze. Tou rozumíme návrh jednotlivých entit a vztahů mezi nimi. Entitami rozumíme tabulky, ve kterých jsou námi uložena data. Byla navržena databáze obsahující 6 tabulek. V pěti z nich jsou uloženy informace o studentech, oborech, třídách, vyučujících a uživateli. Šestá slouží k ukládání záznamů vyučujících zákonným zástupcům studentů. Na Obr. 2 je ERD (entity relation diagram) diagram této databáze, který názorně ukazuje její strukturu a vztahy mezi entitami. Tento ERD diagram byl vytvořen pomocí aplikace CASE studio 2.



Obr. 2. ERD diagram

3.1.2 Popis tabulek

Tabulka obor

V této tabulce jsou uloženy veškeré obory, které jsou na SOŠ a SOU technickém v Přerově vyučovány. Obsahuje 4 atributy, které uložený obor charakterizují. V první řadě je to identifikace oboru pomocí jeho identifikačního čísla *id_obor*. V tomto atributu je uloženo identifikační číslo oboru, které obor na dané škole má. Byl použit datový typ Varchar s maximální délkou 12 znaků. Tento typ byl použit především proto, že každý obor může mít identifikační číslo o jiné délce. Musí být u každého oboru zadán, nesmí být prázdný. Druhým polem je *nazev*. Obsahuje přesný název oboru. Tento je primárním klíčem této tabulky. Každý obor nese nějaký svůj název a proto musí být vždy vyplněn. Jako datový typ byl použit Varchar o maximální délce 35 znaků. Tato délka je plně postačující pro zadání názvu oboru. Třetím atributem charakterizující jednotlivé obory je *typ*. Jelikož na této škole jsou vyučovány jak obory studijní, tak obory učební, mělo by být rozeznatelné o jaký typ studia se jedná. To je vyřešeno polem *typ*. Může tam být uložena hodnota *studijní* a nebo *učební*. Byl použit datový typ Varchar, který může mít maximální délku 8 znaků. Posledním atributem je *zamereni*. Na SOŠ a SOU technickém v Přerově se vyučují obory zaměřené na optiku, strojírenství nebo elektro. Proto do *zamereni* bylo uloženo na co se daný obor specializuje. Na něj bylo také použito datového typu Varchar. Tentokrát s maximální délkou 15 znaků, která mu plně postačuje.

Tabulka vyucujici

Tato tabulka obsahuje 5 atributů, které charakterizují vyučující, kteří na této škole pracují. V první řadě to je identifikační číslo vyučujícího *id_vyuc*. Toto číslo vyučujícího jednoznačně identifikuje. Stejně identifikační číslo nemůže mít více vyučujících. Narušila by se tím jednoznačná identifikace každého z nich. Proto je *id_vyuc* použito jako primární klíč této tabulky. Jako datový typ je použit Char s maximální délkou 5 znaků. Toto číslo je přidělováno správcem sítě na SOŠ a SOU technickém v Přerově a má tvar např. v0001. Tento atribut nesmí být prázdný, každý vyučující musí mít *id_vyuc* přiděleno. Druhým v řadě je *jmeno*. Bylo použito datového typu Varchar o maximální délce 15 znaků. Takto zvolená délka je dostačující pro uložení křestního jména vyučujícího do databáze. Tento parametr nesmí být stejně jako *id_vyuc* prázdný. Každý člověk má své křestní jméno a proto i zde musí být vyplněno. Dalším parametrem charakterizující vyučujícího je

prijmeni. Stejně jako u *jmena* je použit datový typ Varchar. Tentokrát však o maximální délce 25 znaků. A to proto, že příjmení vyučujícího může být různě dlouhé a tato délka by měla plně postačovat našim potřebám. Další shodnou vlastností se *jmenem* je ta, že také nesmí být tento parametr prázdný, protože každý má své příjmení. Následují atributy, které už nutně vyplněné být nemusí. Jedná se o *email* a *telefon*. Jsou to především kontaktní informace na daného vyučujícího. Na atribut *email* bylo použito datového typu Varchar s maximální délkou 40 znaků. Kontaktní e-maily na vyučující mají tvar *prijmeni@kourilkova8.cz*, proto je omezení 40 znaků. 26 je použito na část adresy před @ a zbylé znaky jsou použity na část @kourilkova8.cz. 26 proto, že na škole může být zaměstnáno více vyučujících se stejným příjmením a aby byly e-mailové adresy od sebe jednoznačně rozeznatelné přidá se k příjmení ještě jeden znak (např. první písmeno jména). Na parametr *telefon* je použito datového typu Char o délce 10 znaků. To proto, že formát telefonních čísel má následující formát: XXX/YYYYYY.

Tabulka trida

Tato tabulka už získává některá data z jiných tabulek. Obsahuje 4 atributy. Prvním je *oznaceni*. Jedná se o jednoznačné označení třídy a ročníku. Většinou se toto označení skládá ze 3 znaků, kde první dva znaky definují obor a třetí ročník např. Op2 (Optik, 2. ročník). Výjimkou, ale není označení složené ze 2, 4 nebo 5 znaků. Proto je také použit datový typ Varchar o maximální délce právě 5 znaků. Tento parametr nesmí být v žádném případě prázdný a musí být jedinečný, proto byl také definován jako primární klíč. Druhým atributem je *tridni*. Jedná se o určení třídního učitele, který k třídě náleží. U tohoto parametru se jedná o cizí klíč. Což znamená, že získává hodnotu z jiné tabulky. Konkrétně je získáváno *id_vyuc* z tabulky *vyucujici*. Protože musí být datový typ shodný s datovým typem *id_vyuc* v tabulce *vyucujici* je i zde použit Char s délkou maximálně 5 znaků. Stejně jako atribut *oznaceni* nesmí i tento být prázdný, protože každá třída musí mít svého třídního vyučujícího. V pořadí třetím je *obor*. Shodně jako předchozí *tridni* se jedná o atribut, který získává data z jiné tabulky. V tomto případě se jedná konkrétně o získání názvu oboru. Proto je z tabulky *obor* získáván atribut *nazev* do námi definovaného atributu *obor* v tabulce *trida*. Tento je tedy cizím klíčem. Stejně jako v tabulce *obor* musí mít i zde shodný datový typ a proto je zde taktéž datový typ Varchar s maximální délkou 35 znaků. Tento parametr stejně jako předešlé dva nesmí být prázdný. Posledním je *rocnik*. Ten je datového typu Integer. Na SOŠ a SOU technickém v Přerově jsou vyučovány tříleté nebo

čtyřleté (maturitní) obory. Proto v parametru ročník může být hodnota 1, 2, 3 popřípadě jedná-li se o maturitní obor i 4.

Tabulka student

Tato se skládá z 5-ti atributů, které nám charakterizují jednotlivé studenty. Prvním je *id_st*, jedná se o jedinečnou identifikaci studenta nebo studentky. Jedno identifikační číslo nesmí používat více studentů a musí být vždy zadáno, proto je tento parametr primárním klíčem tabulky *student*. Byl použit datový typ Char o maximálně 5-ti znacích. Toto identifikační číslo přiděluje studentům správce počítačové sítě ve formátu např. s0001. Dalšími parametry tabulky *student* jsou *jmeno* a *prijmeni*. Oba tyto jsou datového typu Varchar a nesmí být prázdné. Parametr *jmeno* může mít maximální délku 15 znaků a *prijmeni* může mít znaků až 25. Čtvrtým atributem je *email*. Tento je samozřejmě také nepovinný, protože všichni studenti nemusí mít nutně svůj vlastní email. U tohoto byl použit datový typ Varchar o maximálně 40-ti znacích. Každý student nebo studentka co SOŠ a SOU technické v Přerově navštěvuje musí být zařazen do nějaké třídy, podle oboru, který studuje. Proto je tu poslední atribut *trida*, ve kterém je uložen údaj do které třídy daný student nebo studentka chodí. Údaje o třídách, jejich označení, jsou získávány z tabulky *trida*. Proto také atribut *trida* v tabulce *student* byl charakterizován jako cizí klíč. Datový typ tohoto atributu musí být shodný s datovým typem atributu *oznaceni* z tabulky *trida*. Proto je také tento typu Varchar s délkou maximálně 5 znaků.

Tabulka user

Tato tabulka bude obsahovat veškeré uživatele, kteří se budou moci přihlásit na stránkách do jejich zabezpečené části. Jedná se především o administrátora, vyučující a zákonné zástupce studentů. Tabulka *user* obsahuje 4 atributy. Prvním je *id*. Tento je použit jako primární klíč této tabulky. Toto identifikační číslo přiděluje uživatelům správce sítě. Jedná-li se o vyučujícího je jako *id* použito *id_vyuc* z tabulky *vyucujici*. V případě že se jedná o zákonného zástupce, tak mu správce sítě přidělí identifikační číslo ve tvaru např. p0001. Administrátor, tedy správce sítě, má identifikační číslo ve formátu např. a0001. Aby mohlo být přijato *id_vyuc* z tabulky *vyucujici* musí mít atribut *id* v tabulce *user* stejný datový typ. Proto byl použit datový typ Char o délce znaků maximálně 5. Druhým atributem je atribut *jmeno*. Tento je stejně jako *id* přidělován správcem sítě, uživatelé nemají možnost si toto jméno žádným způsobem měnit. Slouží jako přihlašovací jméno do chráněné oblasti stránek. Byl definován datový typ Varchar o délce maximálně 6 znaků.

Vyučující mají jména ve formátu např. V03508 a zákonní zástupci studentů mají jména ve formátu např. T03453. Tento atribut nesmí být prázdný a musí být unikátní, proto je u toho atributu vlastnost *unique*. Dva uživatelé totiž nesmějí mít stejné uživatelské jméno. V pořadí třetím atributem je *heslo*. Slouží jako heslo k přihlašovacímu jménu. Heslo si také nemá možnost uživatel sám zvolit, to mu přiděluje správce sítě sám. Je použit datový typ *Varchar* o maximální délce 40 znaků. Heslo je v tabulce zašifrováno pomocí funkce MD5. Hlavním důvodem pro zašifrování hesla byla ochrana proti vstupu neoprávněných osob do databáze. Pokud by nebylo heslo zašifrováno, mohli by si tito narušitelé zjistit uživatelská jména a hesla a napadat zabezpečené sekce stránek. Atribut *heslo*, stejně jako předchozí dva, nesmí být prázdný. Posledním parametrem této tabulky je *student*. Tento atribut nemusí být nutně zadán. Jedná-li se o vyučujícího, tak tento atribut nezasadáme. Pokud se však jedná o zákonného zástupce musíme studenta zadat. A to jako *id_st* z tabulky *student*. Znamená to tedy, že tento atribut je cizím klíčem. Stejně jako je v tabulce *student* použit u *id_st* datový typ *Char* s maximální délkou 5 znaků, musí být stejně tak tento datový typ použit i u atributu *student* v tabulce *users*.

Tabulka zaznamy

Tato tabulka slouží k ukládání sdělení, které napíše vyučující zákonným zástupcům studentů. Ti pak mají možnost si tyto sdělení prohlížet. Tato tabulka obsahuje 6 atributů, které charakterizují a popisují sdělení. Prvním je *id_zazn*. Jedná se o identifikační číslo vloženého záznamu. Toto nesmí zůstat prázdné. U tohoto atributu je přidělena vlastnost *autoincrement* což znamená, že se identifikační číslo záznamu bude přidělovat automaticky po vložení nového záznamu. Jako datový typ byl použit *Integer*. Druhým atributem v řadě je *autor*. Jedná se o cizí klíč, který získává data z tabulky *vyucujici*. Z této tabulky se získává *id_vyuc*, které jednoznačně identifikuje, kdo je autorem vloženého záznamu. Proto musí být použit stejný datový typ jako je u *id_vyuc* v tabulce *vyucujici*. Třetím v pořadí je *datum*. Do tohoto se ukládá aktuální datum, kdy bylo sdělení vloženo do tabulky. Proto bylo použito datového typu *Date*. Datum se vkládá ve formátu YYYY-MM-DD. Dalším je *cas*, který stejně jako datum určuje dobu, kdy byl záznam vložen. To z toho důvodu, aby si mohl zákonný zástupce zjistit spolu s datem, kdy bylo sdělení přidáno a nemusel přemýšlet zda si toto sdělení už četl nebo nikoliv. Na tento atribut bylo použito datového typu *Time*. Ten do něj ukládá aktuální čas ve formátu HH:MM:SS. Následujícím atributem je *obsah*. Jedná se o atribut datového typu *Text* a proto se do něj ukládá přímo text sdělení, které daný vyučující napsal konkrétnímu rodiči. Napsané sdělení může mít až 65 535

znaků. Posledním parametrem tabulky *zaznamy* je *prijemce*. Do tohoto se ukládá příjemce, tedy student, kterému je sdělení určeno, aby si ho jeho zákonný zástupce mohl přečíst. Jedná se o cizí klíč. Ukládá se do něj tedy atribut *id_st* z tabulky *student*. Datový typ se musí shodovat s typem v tabulce *student*, proto je jím Char s maximální délkou 5 znaků.

3.1.3 Vytvoření tabulek

Strukturu naší databáze již máme navrženu, proto můžeme přistoupit k dalšímu kroku. Tím je vlastní tvorba databáze. Ta byla tvořena pomocí nástroje PhpMyAdmin 2.6.0. Jedná se o webovou aplikaci napsanou v jazyce PHP, která je určena ke správě databáze MySQL. Je distribuován zdarma pod GNU licencí. Pracovní okno aplikace PhpMyAdmin je na Obr. 3. V levé části máme možnost vybrat si databázi. Po výběru databáze se nám v této části ukáže seznam entit, které tato databáze obsahuje. V pravé části okna už tvoříme přímo nové entity, atributy, vztahy atd.

Prvním krokem fyzické tvorby naší databáze bylo vytvoření jednotlivých entit (tabulek) a ke každé z nich vytvoření atributů. Jako první byla vytvořena tabulka *obor* o čtyřech attributech. Atributy této tabulky jsou *id_obor*, *nazev*, *typ*, *zamereni*. Všem těmto byly nastaveny datové typy a atribut *nazev* byl zvolen jako primární klíč této tabulky. Atributu *id_obor* byla přidělena vlastnost unique. To proto, že identifikační číslo každého oboru je jeho jednoznačným identifikátorem.

Jako druhá byla vytvořena tabulka *vyucujici*. Této tabulce byly vytvořeny atributy *id_vyuc*, *jmeno*, *prijmeni*, *email* a *telefon* a byly jim nastaveny zvolené datové typy. Jako primární klíč byl určen *id_vyuc*, tento parametr je pro každého vyučujícího jedinečný.

Třetí tabulkou, kterou bylo potřeba vytvořit, byla tabulka *trida*. Ta mohla být vytvořena až poté, co byly vytvořeny tabulky *obor* a *vyucujici*, protože byly potřeba i atributy těchto tabulek. Jako první byl vytvořen atribut *oznaceni*, byl mu nastaven zvolený datový typ a byl vybrán jako primární klíč této tabulky. Druhým vytvořeným atributem byl *tridni*. Ten svou hodnotu získává z tabulky *vyucujici*. Proto musel být *tridni* nastaven jako cizí klíč k tabulce *vyucujici*. Vazba mezi těmito tabulkami je 1:N. Znamená to, že na jeden obor může být N tříd. Jako datový typ musel být zvolen stejný jako v tabulce *vyucujici* u *id_vyuc*. Třetím parametrem byl *obor*. U tohoto musel být taktéž nastaven cizí klíč, tentokrát však k tabulce *obor*, protože se získává z této tabulky název oboru. Vazba mezi těmito tabulkami je 1:1, jeden vyučující může být třídním učitelem pouze pro jednu třídu.

Datový typ atributu *obor* musí být shodný s datovým typem *nazev* v tabulce *obor*. Posledním atributem tabulky *trida* je *rocnik*, kterému byl nastaven požadovaný datový typ, konkrétně Integer.

V pořadí čtvrtou tabulkou, která byla vytvořena je tabulka *student*. Parametry charakterizující studenty jsou *id_st*, *jmeno*, *prijmeni* a *email*. Tyto byly také vytvořeny a byly jim nastaveny námi požadované datové typy a jako primární klíč byl použit *id_st*. Posledním atributem této tabulky je *trida*. Tento je použit jako cizí klíč a získává informace o označení tříd z tabulky *trida*. Vztah mezi těmito tabulkami je 1:N. To proto, že do jedné třídy může chodit N studentů, ale jeden student může být členem pouze jedné třídy.

Po vytvoření těchto čtyř tabulek byla jedna z nejdůležitějších částí databáze hotová. Ovšem bylo zapotřebí zajistit, aby se k těmto datům na stránkách dostal pouze člověk tomu oprávněný. Proto musela být vytvořena další tabulka, která by obsahovala seznam uživatelů. Tato tabulka byla pojmenována *users*. A obsahuje atributy *id*, *jmeno*, *heslo* a *student*. Atributům *id*, *jmeno* a *heslo* je nastaven žádaný datový typ a jako primární klíč je použit *id*. U *jmena* se muselo také zvolit Unique, protože se v tabulce nesmí vyskytovat více uživatelů se stejným jménem. Posledním atributem této tabulky je *student*. Tento je cizím klíčem k tabulce *student*, kde získává *id_st*. Vztah mezi těmito tabulkami je 1:1, ovšem jen za jistých podmínek. Tou podmínkou je, pokud uživatelem je zákonný zástupce. Pokud uživatelem bude kdokoliv jiný vztah zaniká (1:0).

Poslední tabulkou, kterou bylo potřeba vytvořit je tabulka, kterou budou používat především vyučující a to k tomu, aby mohli posílat různá sdělení zákonným zástupcům studentů. Bylo zapotřebí vytvořit atributy, které by obsahovali veškeré informace, které budou identifikovat zápis. Proto byly vytvořeny *id_zazn*, *autor*, *datum*, *cas*, *obsah* a *prijemce*. Primárním klíčem byl zvolen *id_zazn*. Atribut *autor* je cizím klíčem k tabulce *vyucujici*. Bude se do něj ukládat identifikační číslo autora, tedy *id_vyuc*. Vztah mezi těmito tabulkami je 1:N. Aktuální datum vložení záznamu bude uloženo do *datum* s datovým typem Date a aktuální čas do *cas* o datovém typu Time. Sdělení které bude napsáno vyučujícím uloženo do atributu *obsah*. Jemu byl nastaven datový typ Text a proto může vyučující napsat sdělení v délce až 65 535. Posledním a velmi důležitým atributem této tabulky je *prijemce*. Tento je cizím klíčem k tabulce *student*, kde získává identifikační

číslo studenta, tedy *id_st*. Vztah mezi těmito tabulkami je naopak N:1. Jeden student může mít zapsáno několik záznamů.

3.1.4 Uložení dat do tabulek

Po vytvoření všech tabulek můžeme přejít k dalšímu kroku. Tím je uložení veškerých dat do tabulek a naplnit tak databázi daty. Nejprve uložíme do tabulky *obor* veškeré obory které se na SOŠ a SOU technickém v Přerově vyučují. Každý obor má své vlastní identifikační číslo, proto do *id_obor* toto číslo uložíme. Název daného oboru byl uložen do atributu *nazev*. Do *typ* byl ke každému oboru uložen typ studia. Na této škole se vyučují jak obory maturitní (studijní), tak obory učební. Proto zde ke každému oboru bylo uloženo o jaký typ studia se jedná (studijní nebo učební). Posledním atributem charakterizující daný obor je *zamereni*. Zde ke každému oboru bylo uloženo na co se obor zaměřuje. Zaměřuje-li se na strojírenství bylo uloženo strojírenství. Dalšími možnostmi jsou optika a nebo elektro.

Po uložení veškerých dat (oborů) do této tabulky bylo započato ukládání dat do tabulky *vyucujici*. V této tabulce jsou tři atributy, které nesmějí zůstat nezadané. Správce sítě sám přidělil každému z vyučujících identifikační číslo. To má tvar např. v0001 a bylo uloženo do *id_vyuc*. Jméno a příjmení vyučujících bylo uloženo do *jmeno* a *prijmeni*. Dalšími daty, která jsme mohli do této tabulky uložit jsou kontaktní informace na konkrétního vyučujícího. Těmito kontaktními informacemi je e-mail a telefon. Tyto data už nemusí být nutně zadána, protože každý vyučující tyto kontakty mít nemusí. Ovšem v případě e-mailu má většina vyučujících e-mail školní a to ve tvaru *prijmeni@kourilkova8.cz*.

Na Střední odborné škole a středním odborném učilišti technickém v Přerově je v momentální době 18 tříd. Příští rok se bude počet tříd rozšiřovat. Všechny tyto třídy musíme uložit do databáze *trida*. Každá třída má své typické označení, které ji jednoznačně charakterizuje a určuje ročník. Toto označení bylo uloženo do *oznaceni*. Každá z tříd by měla mít svého třídního učitele, proto do *tridni* uložíme *id_vyuc* z tabulky *vyucujici*, který je třídním učitelem té konkrétní třídy. Další informací kterou by měla zvolená třída mít je obor na který se daná třída zaměřuje. Proto byla do tabulky *trida* ke každé třídě uložena data s názvem oboru na který se daná třída zaměřuje. Posledními daty která zbývají ke každému oboru uložit je její ročník.

Nyní můžeme přikročit k plnění daty o studentech tabulku *student*. Na škole studuje přes 500 studentů a všechny tyto studenty bylo potřeba do databáze, konkrétně do tabulky *student*, uložit. Jméno a příjmení studentů uložíme do *jmeno* a *prijmeni*. Správce sítě sám každému ze studentů přidělil jejich identifikační číslo. To může mít tvar např. s0001. Jedno identifikační číslo nesmí být přiděleno dvěma studentům, protože jednoznačně identifikuje každého z nich. Spousta studentů vlastní v dnešní době také e-mail, proto ho můžeme uložit do sloupce *email*. Hlavním důvodem je možný kontakt se studentem. Posledním parametrem, který charakterizuje každého studenta je třída do které chodí. Student který školu navštěvuje musí být žákem nějaké třídy, proto tento atribut nesmí být prázdný. Možnosti výběru tříd se získává s tabulky *trida*.

Velmi důležitou tabulkou je tabulka *users*. Tato obsahuje všechny uživatele, kteří budou moci vstupovat do zabezpečených částí stránek. V této tabulce musí být uloženi zákonní zástupci, vyučující a samozřejmě administrátor. Jednotlivým uživatelům bylo správcem sítě přiděleno *id*, *jmeno* a také *heslo*. Atribut *student* je vyplněný pouze tehdy, jedná-li se o zákonného zástupce. Administrátor ani vyučující tento atribut vyplněný mít nebudou. Pokud byla přidávána data o vyučujících jako *id* bylo použito *id_vyuc* z tabulky *vyucujici*, *jmeno* bylo použito např. ve formátu V03506. Heslo které také bylo uloženo k jednotlivým uživatelům bylo při uložení zašifrováno pomocí funkce MD5. Důvodem pro zašifrování byla ochrana proti zjištění jména a hesla neoprávněnými uživateli. Jelikož byl zadáván vyučující atribut *student* nebyl vyplněn. Zákonným zástupcům bylo přiděleno *id* ve tvaru např. p0001. Jako jméno bylo použito např. T03451. Heslo bylo stejně jako u vyučujících zašifrováno funkcí MD5. Jelikož se jedná o zákonného zástupce, tak mu byl také přidělen *student* který k němu náleží.

Poslední tabulkou je tabulka *zaznamy*. Do této tabulky však nebyly dopředu ukládány žádná data. Budou se ukládat během provozu stránek v závislosti na tom, kdy bude potřebovat některý z vyučujících zapsat sdělení některému ze zákonných zástupců studentů.



Obr. 3. Pracovní okno aplikace phpMyAdmin 2.6.0

3.2 Tvorba internetových stránek

Vlastní internetové stránky byly vytvořeny ve dvou krocích. Prvním krokem bylo vytvoření stránek tak, aby na ně měl přístup kdokoli a dozvěděl se na nich veškeré informace, které souvisí se Střední odbornou školou a středním odborným učilištěm technickým v Přerově. Tyto informace by měli být co nejpřehledněji prezentované. V druhém kroku byla k těmto stránkám vytvořena a připojena ta část, která zajišťuje vstup do zabezpečených oblastí stránek a samotné stránky, které mají být zobrazeny pouze pro oprávněné uživatele. K tvorbě internetových stránek bylo použito aplikace Notepad++ v2.8, který je šířen zdarma pod GNU licenci.

3.2.1 Vytvoření veřejných internetových stránek

Tyto stránky budou přístupné široké veřejnosti. Bude mít možnost si na nich přečíst informace kterýkoliv uživatel který je navštíví. Proto by na těchto stránkách měly být pouze informace, které jsou jen informativní a nemohou se žádným způsobem zneužít.

Byly možné dva způsoby, jak tyto stránky vytvořit. Prakticky se jedná o rozdíl v postupu při jejich tvorbě. Prvním je ten, kdy nejprve vytvoříme hlavní stránku a pak k ní jednotlivé stránky postupně přidáváme. V tom druhém případě bychom vytvořili nejprve veškeré informační stránky a až nakonec stránku hlavní ve které by se jednotlivé stránky spojily v jeden celek. Byl vybrán druhý způsob, podle kterého budou stránky vytvořeny. Prvním krokem tedy bylo vytvořit veškeré stránky, které budou poskytovat co nejvíce informací. Bylo tedy zapotřebí přepsat veškeré informativní texty do HTML kódu. Na Obr. 4 je názorně vidět jak tyto stránky poskytující informace budou vypadat.

Domov mládeže

Celá škola a tedy i domov mládeže je výhodně umístěn u zimního stadionu, v blízkosti obchodního domu Prior, kina Hvězda a také středu města. Je součástí komplexu SOŠ a SOUT a zabezpečuje ubytování žáků SOŠ a SOUT a žáků jiných středních škol v Přerově i ubytování sportovců a účastníků různých zájezdů.

Ti jsou ubytováni ve dvou a třílůžkových pokojích v samostatné čtyřpatrové budově. Kapacita domova mládeže je 120 lůžek. Na každém poschodí jsou koupelny se sprchami, společenské místnosti s televizory a studovny

Pro ubytované je dále k dispozici:

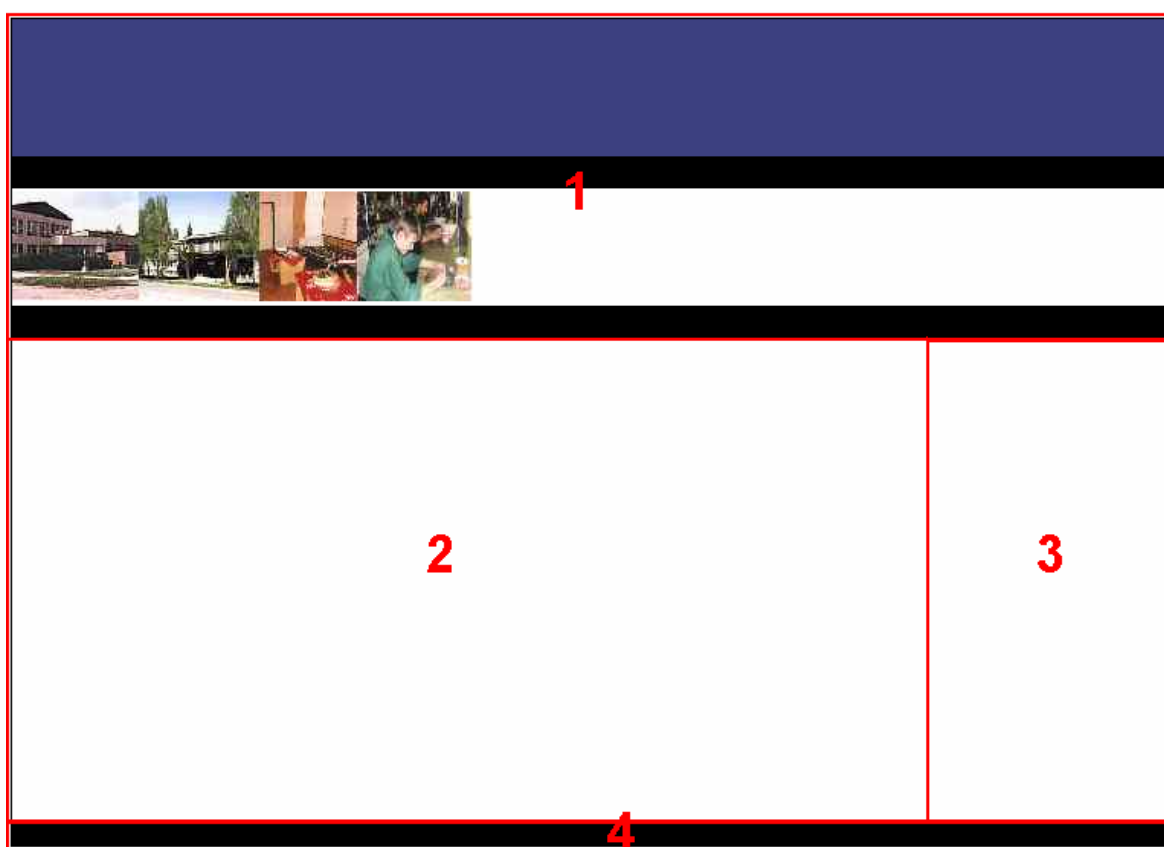
- velká společenská místnost s klavírem, kterou lze pronajímat veřejnosti,
- posilovna,
- venkovní hřiště s umělým povrchem,
- sportovní náradí.

Hlavními akcemi výchovy mimo vyučování jsou výchovné besedy, setkání se zajímavými lidmi, návštěvy kulturních představení, zábavné večery, diskotéky a sportovní soutěže. Je možné se zapojit do zájmových kroužků: vaření, ruční práce, fotokroužek, taneční, sportovní (sebeobrana, míčové hry atd..) Kvalitní strava je poskytována v jídelně SOŠ a SOUT, která je rovněž v areálu školy společně se školní kuchyní.

Obr. 4. Stránka poskytující informace o domovu mládeže

Museli se vytvořit stránky o historii SOŠ a SOU technického v Přerově, kontaktech na důležité zaměstnance školy. Dalším blokem bylo vytvoření stránek o členění školy jako celku. Ta se dělí na tři větší části. Na školu samotnou, kde je organizována výuka a praxe, domov mládeže a školní jídelnu. Jednou z nejrozsáhlejších oblastí je oblast studia. Do této oblasti byla zahrnuta jak organizace výuky, tak i obory vyučované na této škole. V momentální době je na této škole vyučováno 7 oborů, v příštím roce bude výuka rozšířena o další 3 obory. Proto musely být vytvořeny stránky oborů, které by poskytovaly co nejvíce informací případným zájemcům. Všechny doposud vytvořené stránky používají jednotnou grafickou úpravu. To především z toho důvodu, aby měl web jednotnou úpravu

a nebyla každá stránka jiná. Každá z vytvořených stránek má svůj nadpis, aby návštěvník věděl kterou část webu si právě prohlíží. Tyto nadpisy mají červenou barvu, velikost písma 3 a styl písma Verdana popřípadě Arial, pokud nemá uživatel písmo Verdana v systému nainstalováno. Na vlastní texty je také použito písmo Verdana popřípadě Arial, ovšem velikost textu je 2 a barva černá. Barva černá byla zvolena pro dobrou čitelnost, protože pozadí stránek je bílé. Po takto vytvořených stránkách můžeme přikročit k vytvoření vlastní grafické podoby webu. Ten byl vytvořen pomocí CSS a XHTML. Na Obr. 5 je vidět grafický návrh internetových stránek, který se skládá ze čtyř základních částí ohraničených červeně a očíslovaných.



Obr. 5. Grafický návrh stránek

Hlavičku stránek (1), která ještě dále bude obsahovat čtyři vnořené bloky, *hlavní část* (2) pro vlastní obsah webu, *pravou část* (3), kterou si připravíme pro přihlášení uživatelů a *patičku* (4) určenou copyrightu a pro další informace. Nejprve bylo zapotřebí vytvořit v hlavní adresáři, ve kterém již byly uloženy vytvořené stránky oborů, struktury atd., podadresáře s názvem *images*, do kterého budeme ukládat veškeré obrázky, a *styles*, kde bude uložen soubor se stylovým předpisem pro vzhled webu. Tento předpisový soubor bude mít název *styl.css*. Díky němu se bude řídit celý vzhled webu. Ještě před tím bylo

zapotřebí připravit si ilustrační obrázek. Protože tento web je koncipován jako web s pevnou šířkou, nesmí šířka obrázku ani webu přesáhnout 760px, aby se nezobrazoval vodorovný posuvník. Po konečných úpravách má obrázek rozměry 302x72px a je ve formátu JPG. Byl nazván *obrazek.jpg* a uložen do adresáře images. Protože se jedná o pevný grafický návrh, bylo by dobré umístit celý web horizontálně na střed obrazovky. Abychom pro každý blokový prvek nemuseli definovat umístění horizontálně na střed, byl vložen celý obsah webu do jednoho bloku *div id="hlavni* a ten následně ve stylovém předpisu donucen, aby byl horizontálně na středu obrazovky. Všechny blokové elementy, ze kterých se skládá výsledná podoba stránek byly označeny pomocí identifikátoru *ID*. Ten slouží k tomu, abychom mohli všechny elementy uvnitř takto označených bloků stylovat. V rámci jedné stránky se nesmí stejný identifikátor objevovat vícekrát. Část BODY v dokumentu bude pak vypadat takto:

```
<body>
<!-- hlavní slouží k umístění prezentace horizontálně na střed obrazovky -- >
<div id="hlavni">
<!-- hlavička webu, obsahuje čtyři vnořené elementy DIV -- >
<div id="hlavicka">
<div id="horni"></div>
<div id="jazyky">(...zde můžeme vkládat kódy (odkazy) na jiné jazyky...)</div>
<!-- - obrázek webu -- >
<div id="obrazek">(...vložíme obrazek.jpg...)</div>
<!-- - obrázek webu -- >
<!-- - rozcestné menu - - >
<div id="menu">(...vložíme JavaScript s menu...)</div>
<!-- - rozcestné menu - - >
</div>
<!-- hlavička webu, obsahuje čtyři vnořené elementy DIV -- >
<!-- - vlastní obsah webu -- >
<div id="obsah"></div>
<!-- - vlastní obsah webu -- >
<!-- - přihlášení k zabezpečenému webu -- >
<div id="prihlaseni"></div>
<!-- - přihlášení k zabezpečenému webu -- >
<!-- - copyright -- >
<div id="copyright">© 2005 www.kourilkova8.cz</div>
<!-- - copyright -- >
</div>
<!-- hlavní slouží k umístění prezentace horizontálně na střed obrazovky -- >
</body>
```

Tímto byl vytvořen základní grafický návrh webu a mohl být uložen jako *index.php*. Tomuto návrhu mohly být nyní přiděleny vlastnosti a celkový vzhled. To bylo provedeno v již vytvořeném souboru *styl.css*. Nejprve museli být nadefinovány vlastnosti, které budou platné pro celý web. Pro element BODY byly tyto vlastnosti nastaveny takto:

```
body {  
  font-family: Verdana, sans-serif;  
  font-size: 80%;  
  background: #ffffff;  
  margin: 0;  
  padding: 10px 0;  
  text-align: center;  
  min-width: 770px;  
}
```

Všechny tyto parametry nyní přiblížím. Nejprve bylo nutné nastavit písmo celého webu, proto byl použit *font-family*. Bylo vybráno písmo Verdana, protože se dobře čte a je jedno z nejpoužívanějších na internetu a byl doplněn o obecnou definici. To z toho důvodu, že může nastat případ, kdy uživatel toto písmo nebude mít nainstalováno, prohlížeč sám vybere ze systému uživatele podobný font a použije ho. Toto uvedení obecné definice je povinné. Dalším parametrem je *font-size*, který nastavuje velikost písma platnou pro celý web. Pokud bychom tento parametr neuvedli hrozilo by riziko, že v každém prohlížeči bude velikost písma jiná. Třetím parametrem je *background*. Již z názvu je patrné, že se jedná o pozadí. Tento by měl být také vždy zadán, protože ne každý prohlížeč má nastavenou takovou výchozí barvu pozadí, jakou bychom očekávali. Jako barva pozadí celého webu byla zvolena barva bílá. Parametr *margin* určuje okraj, i když je nastaven na nulu, je lepší ho i přesto uvést, protože ne všechny prohlížeče mají jako výchozí hodnotu nulu. Takhle máme nulu zajištěnu pro všechny prohlížeče. Dalším je *padding*, jedná se o výplň. Byla nastavena na hodnoty 10px a 0. Znamená to, že hrana vlastního webu bude o 10px odsazená shora i zdola. *Text-align* slouží k zarovnání textu. Bylo nastaveno na střed a to především kvůli tomu, aby se nám prvek *div id="hlavni"* umístil na střed obrazovky. Vlastnost *min-width* byla použita především pro uživatele Mozilly. Internet explorer tuto vlastnost přehlíží, ale pro Mozilu je velmi důležitá. Pokud bychom si totiž zmenšili více okno prohlížeče stránka by zůstala stále uprostřed, avšak části textu by se nám ořezali. Tato definice tomu však zabrání. Pokud tomu nebude v některé části webu dáno jinak bude toto nastavení vždy použito.

Dalším krokem je definice hlavního prvku *div id="hlavni"* a zajištění jeho umístění, tedy i celé stránky, na střed obrazovky.

Celá definice tohoto bloku bude vypadat takto:

```
#hlavni {  
  width: 760px;  
  margin: 0 auto;  
  text-align: left;  
  border: 1px solid #000000;
```

```
}
```

Křížek # ve stylovém předpisu označuje ty prvky, které mají přiřazen identifikátor ID a nastavuje jim námi požadované vlastnosti. Parametr *width* definuje šířku hlavního prvku. Tedy šířku našeho okna, které navrhujeme. Byla zvolena šířka 760px. Celý web bude tedy široký 760 pixelů. Tento rozměr vyhovuje i rozlišení 800x600. *Margin: 0 auto* udělá totéž co *text-align: center* v definici BODY a to umístí celý blok na střed především v prohlížečích Opera a Mozilla. Konkrétně nastaví horní a dolní okraj na 0 a levý a pravý automaticky dopočítá. Parametr *text-align* slouží k zarovnání textu zpět doleva. Poslední vlastností je *border*. Této vlastnosti byly nastaveny parametry 1px solid #000000, které jsou příčinou toho, že celý blok (web) bude ohraničen černou linkou o šířce 1px.

Dalším, který byl nadefinován je div id="horni" Definice tohoto bloku je následující:

```
#horni {  
  height: 65px;  
  background: #3c4080;  
  font-size: 120%;  
  color: #ffffff;  
  text-align: center;  
  padding: 25px 0 0px 0;  
}
```

Height říká, jak má být tento blok vysoký. Byla určena výška 65px a barva pozadí tmavě modrá, parametrem *background*. Do tohoto bloku bude zapsán název školy, tedy Střední odborná škola a střední odborné učiliště technické v Přerově. Protože bylo pozadí tohoto bloku nastaveno na tmavě modrou barvu, musí být text názvu světlý. Proto byl název školy napsán bílou barvou pomocí *color: #ffffff* a jeho velikost určena na 120%. Celý název je potom zarovnán na střed a díky *padding* posunut od horní hrany tohoto bloku o 25 pixelů.

Následujícím krokem bylo vytvoření černého proužku pod blokem s názvem školy. Do tohoto budou vloženy odkazy na vícejazyčné verze stránek. Vlastnosti tomuto proužku dávají následující parametry:

```
#jazyky {  
  height: 20px;  
  background: #000000;  
  border-top: 1px solid #000000;  
}
```

Výška tohoto proužku je 20 pixelů a barva pozadí je černá. Později budou do něj vloženy odkazy na verze stránek v cizích jazycích. Konkrétně v němčině a angličtině. Pod takto vložený proužek byl vložen tentýž se stejnými vlastnostmi. Jediným rozdílem je, že se

tento blok bude jmenovat `div id="menu"` a bude v něm zobrazováno, namísto odkazů na jiné jazykové verze, menu, které bude sloužit k jednoduchému prohlížení stránek.

Poslední částí hlavičky je dekorační obrázek, který jednotvárnost webu trochu oživí. Tento obrázek byl vložen mezi vytvořené černé proužky s odkazy na jiné jazyky a menu. Blok byl pojmenován `div id="obrazek"`. Tento je řešen následující částí kódu ve stylovém předpisu:

```
#obrazek {  
  height: 75px;  
  background: #ffffff url("../images/obrazek.jpg") no-repeat;  
  border-top: 2px solid #ffffff;  
}
```

Výška tohoto bloku je 75 pixelů a pozadí je bílé barvy. Do takto vytvořeného bloku vložíme obrázek, který jsme si uložili pod názvem *obrazek.jpg* a určíme mu tu vlastnost, aby se neopakoval (*no-repeat*). To proto, aby byl vložen jen jednou i když by byla možnost, že by se do tohoto bloku vlezl vícekrát. V souladu s grafickým návrhem, který je uveden na Obr. 5 musíme nad obrázek ještě umístit dvoupixelový rámeček.

Nyní bylo možné vytvořit dva plovoucí prvky, které jsou jedněmi z nejdůležitějších částí našeho webu. Podle Obr. 5 byl nastaven pravý plovoucí sloupec na šířku 150px. Výšku sice neznáme, protože se může měnit v závislosti na množství obsahu, ale musíme určit nějakou minimální hodnotu, aby náš web měl ucelenou minimální výšku. K tomu se používá parametr *min-height*. Jeho nevýhodou je, že na rozdíl od jiných prohlížečů (Mozilla, Opera atd.) ho Internet explorer ignoruje. Proto musel být použit i *_height*, který už Internet explorer nepřehlédne. Tato výška byla nastavena na hodnotu 317px. Pozadí tohoto bloku je bílé barvy a celý je plovoucím vpravo. Takto vypadá výsledný kód, který byl před chvílí popsán:

```
#prihlaseni {  
  width: 150px;  
  float: right;  
  text-align: center;  
  background: #ffffff;  
  min-height: 317px;  
  _height: 317px;  
}
```

Druhým plovoucím prvkem je sloupec, který bude sloužit ke zobrazování textu jednotlivých stránek. Tento sloupec máme označený v kódu stránky jako `div id="obsah"`. Šířka tohoto okna byla zvolena 598px. Toto okno je, na rozdíl od předchozího, plovoucím vlevo. Ještě bylo důležité zajistit to, aby se text nezobrazoval přímo u okrajů, byl by špatně

čitelný. Proto je v tomto bloku použit bílý rámeček, který nám tento problém vyřešil. Zde je uveden výsledný kód:

```
#obsah {
width: 598px;
float: left;
border-left: 5px solid #ffffff;
border-right: 5px solid #ffffff;
border-top: 5px solid #ffffff;
border-bottom: 10px solid #ffffff;
}
```

Poslední částí, kterou zbývalo vytvořit byl copyright. Tu máme ve stránce označenu jako div id="copyright". Tato část nám uzavírá celou stránku. Obsahuje internetovou adresu školy a © 2005. Barva pozadí patičky (copyrightu) je černá a použitý text je psán bílou barvou. Velmi důležité je použití definice *clear: both*. Tato vlastnost nám ukončí veškerá obtékání na stránce a umístí copyright až dolů. Definice tohoto je následující:

```
#copyright {
clear: both;
color: #ffffff;
background: #000000;
}
```

Takto vytvořen styl bude definicí celého návrhu webu. Ještě je zapotřebí menu, které bude vloženo do černého proužku s názvem menu. Toto je vyřešeno pomocí Javascriptu. Jedná se o rozbalovací menu, které se po přejetí kurzoru myši rozbalí a po opuštění oblasti menu opět schová. V závislosti na množství odkazů je menu rozčleněno do několika vrstev a tím se rozsáhlost značně sníží. Informační stránky, na které odkazují položky menu, jsou zobrazovány na stránce v div id="obsah". Do této definice byl napsán následující skript:

```
<?php
$filename = $stranka . ".php";
if (File_Exists ($filename))
{
include $filename;
}
else
{
include "chyba.php";
}
?>
```

Pokud uvedená stránka existuje, tak bude zobrazena ve zmíněném bloku. Pokud tomu však není bude spuštěn soubor chyba.php, která uvědomí uživatele, že uvedená stránka neexistuje. Na Obr. 6 je vidět již kompletní grafické řešení webu, který jsme vytvořili.



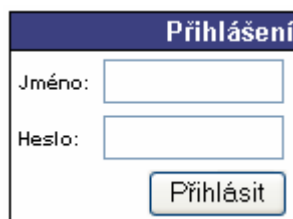
Obr. 6. Hotový návrh webu

3.2.2 Vytvoření zabezpečených stránek a jejich připojení

Na tyto stránky budou mít přístup pouze uživatelé, kteří tomu budou oprávněni. Těmi uživateli mohou být zákonní zástupci studentů, vyučující a samozřejmě administrátor. Každý z těchto možných uživatelů bude mít možnost provádět různé operace. Vyučující budou moci psát zákonným zástupcům různá sdělení, která se týkají jejich dítěte. Naproti tomu zákonný zástupce si může tyto sdělení přečíst. Administrátor bude obstarávat veškeré úkony spojené se správou databáze. Bude moci přidávat nové studenty, třídy, obory atd., upravovat již existující a nebo odebírat již nepotřebné.

Vytvoření okna pro přihlášení a vytvoření přihlašovacího skriptu

Grafický návrh přihlašovacího okna byl navržen tak, aby ničím stránky nenarušoval. Na Obr. 7 je ukázáno takto vytvořené okno. Toto okno bude později umístěno do pravé části webu. Pro přihlášení jsme povinni vyplnit přihlašovací jméno a heslo.



The image shows a simple web form for login. It has a title bar that says 'Přihlášení'. Below the title bar, there are two text input fields. The first is labeled 'Jméno:' and the second is labeled 'Heslo:'. Below these two fields is a button with the text 'Přihlásit'.

Obr. 7. Přihlašovací okno

Hlavní částí tohoto okna jsou dvě formulářová pole a tlačítko pro potvrzení vstupu. První z těchto polí je typu *text*. To proto, aby mohlo být zadáno přihlašovací jméno a uživatel viděl jaké jméno vyplnil. Políčko pro zadávání hesla je typu *password*. Je velmi důležité použít tento typ hlavně z toho důvodu, aby nebylo možné přechíst zadávané heslo neoprávněnou osobou. Místo znaků, které jsou zadávány do tohoto políčka, jsou vidět jen tečky.

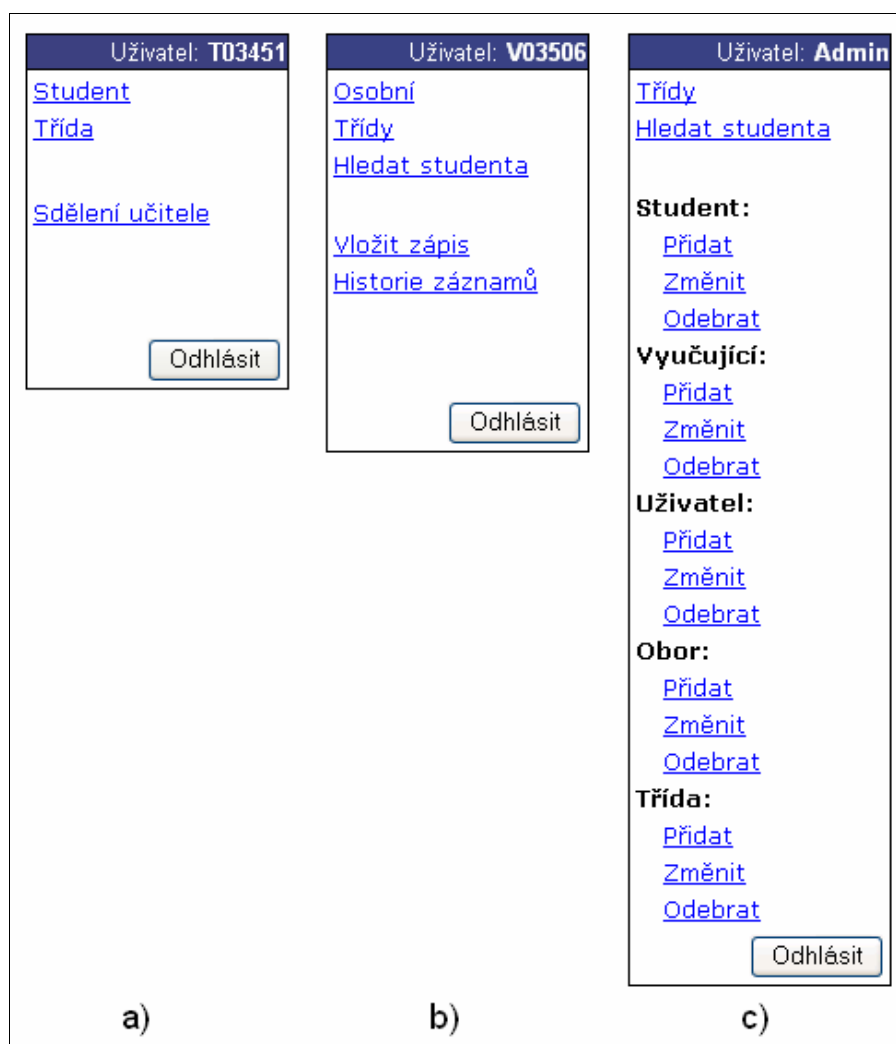
Vlastní přihlašovací skript je vytvořen pomocí skriptovacího jazyku PHP a tvoří ho několik souborů. `Log_data.php`, `log_hlaseni.php`, `log_in.php`, `log_menu.php`, `log_opravneni.php` a `log_prihlasen.php`. Soubor `log_data.php` obsahuje veškerá nastavení přihlášení. Pokud bychom se rozhodli kdykoliv měnit čas po který může být uživatel nečinný stačí to změnit v tomto souboru. Standardně je tento čas nastaven na 10 minut tj. 600 sekund. Totéž platí i u změny v databázi. Pokud bychom se rozhodli změnit název tabulky s uživateli nebo změnit názvy atributů (jméno a heslo) stačí tuto změnu provést v tomto souboru. Ten obsahuje také námi již vytvořené přihlašovací okno. Vyplněné informace o *Jménu* a *Heslu* jsou posílána metodou POST do souboru `log_in.php`. V tomto se zkontroluje, zda uživatel zadal přihlašovací jméno a heslo. Pokud některou z těchto položek vynechal je z tohoto souboru odkázán na další (`log_menu.php`), který zajistí vypsání zprávy ze souboru `log_hlaseni.php` o tom, že zapomněl zadat jméno nebo heslo a zobrazí mu znovu přihlašovací formulář. Pokud uživatel jméno a heslo zadal spojí se tento skript s databází a v ní zjišťuje zda uživatel vyplnil správné jméno a heslo. Pokud nebylo vyplněno správné jméno nebo heslo je tento stav opět poslán souboru `log_menu.php` a ten se postará o to, aby uživatele informoval, že nezadal správné jméno či heslo. Jestliže bude

zjištění v databázi kladně vyřízeno a uživatel zadal správné jméno i heslo uloží se do session *ID* uživatele kterému náleží příslušné jméno a heslo. S touto session pak dále pracujeme. Nepracuje se tedy s uživatelským jménem, ale s *id* uživatele, které je uloženo v naší databázi v tabulce *users*. V tom případě, že je uživatel přihlášen, se musí do session uložit také doba po kterou může být uživatel nečinný a také aktuální čas. V tomto případě může být doba nečinnosti maximálně 600 sekund. Poslední vlastností, která musí být uložena je informace o stavu uživatele. Tedy, že je aktivní. To z toho důvodu, aby skript poznal je-li uživatel přihlášen či ne a mohl by mu nabídnout menu pro přihlášené uživatele. Poté se soubor s tímto skriptem opouští a přechází se k souboru *log_prihlasen.php*. První co musí provést je zjištění o jakou kategorii uživatelů se jedná a podle toho se mu přihlašovací okno změní. Jelikož každá kategorie uživatelů má jiné možnosti musí se mezi nimi rozlišovat. Jako rozlišovací parametr slouží ID jednotlivých uživatelů. Jak již bylo řečeno do session se uloží ID přihlášeného uživatele. Zjistí se první znak, který je právě rozlišujícím jednotlivých skupin. Zjištění je provedeno následujícím způsobem:

```
$znak_id=substr("$id_user", 0,1);
```

Tedy *Substr* zjistí z textového řetězce daný počet znaků, konkrétně první, a uloží ho do proměnné *\$znak_id*. S ohledem na výsledek, který je v této proměnné uložen se provádí větvení programu na jednotlivé větve.

Jedná-li se o zákonného zástupce změní se mu přihlašovací okno do tvaru, který je na Obr. 8 a), vyučujícímu Obr. 8 b) a administrátorovi do tvaru na Obr. 8 c).



Obr. 8. Nabídka po přihlášení

Možnosti zákonných zástupců

Zákonní zástupci mají tři možnosti, které mohou po přihlášení provést. Jako první se mohou podívat na informace jestli jsou opravdu přihlášení k účtu svého dítěte. Zde se dozví také jeho e-mail, pokud ho jejich dítě má. Pokud si přejí zjistit kontaktní informace na třídního učitele, slouží jim k tomu odkaz *Třída*. Zde mají možnost zjistit označení třídy, kterou dítě navštěvuje a obor, který studuje. Najdou tam také seznam jeho spolužáků spolu s e-maily a samozřejmě jméno třídního učitele a kontakt na něj. Všechny tyto informace jsou získávány z databáze pomocí SQL dotazů. Jestliže bylo vyučujícím vloženo nějaké sdělení mohou si ho přečíst po kliknutí na odkaz *Sdělení učitele*. Dozví se co je v daném sdělení za informace, kdy bylo zapsáno a jméno vyučujícího který jej udělil. SQL dotaz na zobrazení sdělení s databáze je následující:

```
"SELECT date_format(zaznamy.datum, '%d.%m.%Y')as datum, zaznamy.cas,
zaznamy.obsah, concat(vyucujici.prijmeni, ' ', vyucujici.jmeno) AS jmeno FROM zaznamy"
```

```
INNER JOIN vyucujici ON zaznamy.autor=vyucujici.id_vyuc
WHERE prijemce='$stud'
ORDER BY datum DESC, cas DESC"
```

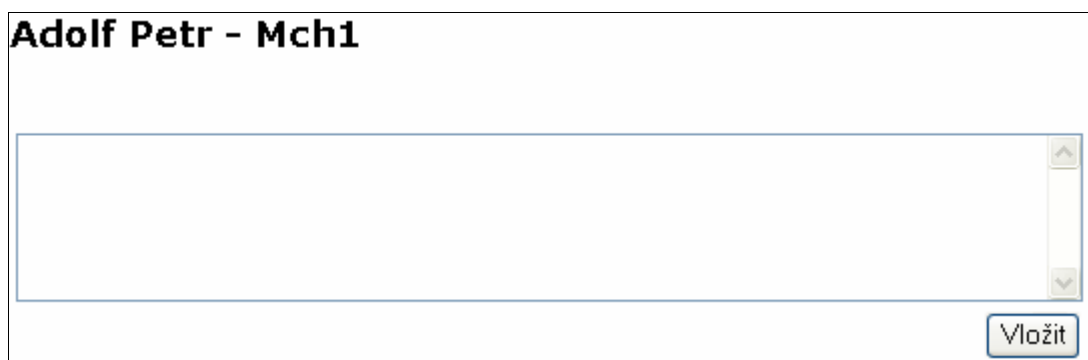
V proměnné \$stud je vždy uloženo ID studenta, kterému je sdělení určeno. A proto může být zákonnému zástupci zobrazeno pouze sdělení pro jeho dítě.

Možnosti vyučujících

Vyučující má na rozdíl od zákonného zástupce možnost prohlédnout si informace přímo o sobě. Dozví se své jméno, e-mail a telefon. K zjištění těchto informací z databáze slouží následující SQL dotaz:

```
"SELECT concat(prijmeni, ' ', jmeno) AS jmeno, email, telefon FROM vyucujici
WHERE id_vyuc='$id_user'"
```

Tyto informace si může zkontrolovat a případně nějaké změny nebo chyby může apelovat na administrátora (správce sítě), aby tyto věci uvedl na pravou míru. Další možností je odkaz na třídy. Zde si může vybrat kteroukoliv třídu, která se na škole vyskytuje a prohlédnout si seznam studentů této třídy. Každému studentovi zde může také vložit zápis, stejně jako tomu je u samostatného odkazu v pravém menu. Jestliže se rozhodne vyučující zapsat sdělení některému ze studentů, vybere si ho ze seznamu studentů rozdělených podle tříd, může tak učinit ve formuláři, který se mu následně poskytne. Tento formulář je vidět na následujícím obrázku.

The image shows a web browser window with a title bar. The page content has a header "Adolf Petr - Mch1" in bold. Below the header is a large, empty text input field with a vertical scrollbar on the right side. At the bottom right of the form area is a button labeled "Vložit".

Obr. 9. Formulář vložení zápisu vyučujícího

Takto napsané sdělení se uloží do databáze do tabulky *zaznamy* díky tomuto kódu:

```
"INSERT INTO `zaznamy` ( `autor` , `datum` , `cas` , `obsah` , `prijemce` )
VALUES ('$id_user', NOW( '0000-00-00' ) , NOW( '00:00:00' ) ,
'" . htmlspecialchars($text) . "' , '$id' )"
```

Jako autor bude zapsán konkrétní vyučující, který zápis provedl. Datum a čas bude vloženo automaticky aktuální v době vložení sdělení a za příjemce bude označen student, kterému byl zápis udělen. Zákonný zástupce konkrétního studenta si pak toto sdělení může

přečíst. Pokud ovšem vyučující zapomene do které třídy chodí student, kterému chtěl zápis udělit, může si podle jména a příjmení vyhledat jen ty studenty, které potřebuje a udělit jim zápis. Zjistí-li po čase, že udělal chybu v zápisu nebo se rozhodne jej úplně odstranit, má k tomu možnost v *Historii záznamů*. Zde si může prohlédnout veškerá sdělení co napsal a pokud uzná za vhodné je odstranit nebo změnit. Jestliže zvolí odstranění bude vyzván, aby tuto možnost ještě potvrdil. Může se totiž stát, že by kliknul na odstranění omylem a zpráva by byla ihned odstraněna, což by nebylo žádané. Okno s historií záznamů je na Obr. 10.

Historie záznamů	
27.05.2005 16:40:08 Komu: Adolf Petr - Mch1	Akce: Změnit Smazat
Bylo zapsáno sdělení!	

Obr. 10. Historie záznamů

Jestliže se rozhodne změnit již zapsaný záznam tuto možnost zde má. Po změně se přepíše staré sdělení sdělením novým v databázi a aktualizuje se i datum a čas vložení na nové.

Možnosti administrátora

Administrátor je osoba, která má na starosti správu celého webu. Z tohoto důvodu musí mít možnost web upravovat a samozřejmě taky upravovat databázi, která s webem souvisí. Bude data do databáze přidávat, měnit data v ní již uložená a odstraňovat stará a nepotřebná. Akce, které může provádět jsou přehledně rozčleněny do jednotlivých skupin viz. Obr. 8. Museli být vytvořeny skripty, které veškeré jeho požadavky budou provádět. Jsou to *pridat.php*, *zmenit.php* a *odebrat.php*. Soubor *pridat.php*, jak již název napovídá, bude sloužit k ukládání nových dat do databáze. Lze přidávat obor, třídu, studenty, vyučující a uživatele. Po vybrání položky, kterou chce administrátor přidat, se mu zobrazí formulář do kterého stačí jen vyplnit informace a kliknout na tlačítko vložit. Tyto budou uloženy do databáze. Příklad je vidět na Obr. 11. Ve formulářovém okně u *ID vyučujícího* je již před vyplněno na které písmeno musí ID začínat. Po vyplnění všech povinných údajů budou data uložena do databáze. Pokud některý z povinných zapomene administrátor zadat nebudou data do databáze uložena a dojde k upozornění.

Vložte údaje o novém vyučujícím

ID vyučujícího: * (např. v0001)

Jméno: *

Příjmení: *

E-mail:

Telefon: (např. xxx/xxxxxxx)

(* povinné)

Obr. 11. Vložení nového vyučujícího

Vzorový dotaz SQL, který vloží nového vyučujícího do databáze:

```
"INSERT INTO `vyucujici` ( `id_vyuc` , `jmeno` , `prijmeni` , `email` , `telefon` )  
VALUES ( '$id_v', '$jm_v', '$pr_v', '$email_vyucujici', '$telefon_vyucujici' )"
```

Druhým souborem, který slouží především administrátorovi je *zmenit.php*. Díky němu je schopen upravovat již uložená data v databázi. Stejně jako u přidávání, tak i zde může upravovat data o třídách, oborech, vyučujících, studentech a uživateli. Celá akce, kdy chceme nějaký údaj změnit je tvořena dvěma kroky. V prvním kroku jde o zjištění již uložených dat o daném subjektu a ve druhém kroku o změnu těchto dat a uložení zpět do databáze. První krok je názorně předveden na Obr. 12. Jde o změnu údajů studenta Petra Adolfa. Z databáze byly zjištěny veškeré údaje, které o něm víme. A vyplněny do formulářových polí.

Změnit údaje studenta

ID studenta: *

Jméno: *

Příjmení: *

E-mail:

Třída: *

(* povinné)

Obr. 12. Změna údajů studenta

Tyto data můžeme změnit podle toho jak požadujeme. Po potvrzení, tlačítkem *Změnit*, se údaje v databázi nahradí. Musíme však dodržet, aby byla vyplněna všechna povinná pole

formuláře. Pokud bychom toto nedodrželi, nebyly by data v databázi změněna a objevilo by se varování na vyplnění všech povinných údajů. Pokud si rozmyslíme a nebudeme chtít data změnit máme možnost vrátit se zpět.

Posledním souborem, který slouží především administrátorovi je *odebrat.php*. Již z názvu je patrné co tento soubor bude dělat. Bude odstraňovat data, které jsou uložena v databázi. Stejně jako *zmenit.php* pracuje ve dvou krocích. V prvním kroku se znovu ptá jestli si přejeme zmíněný objekt (student, obor atd.) opravdu odstranit. Jestliže odstranění nepotvrdíme, vrátí se zpět k výpisu a žádnou akci neprovede. Pokud ovšem zmíněné odebrání potvrdíme bude proveden druhý krok. Tudíž samotné odebrání, již bez možnosti návratu. SQL dotaz tohoto kroku může vypadat následovně:

```
"DELETE FROM student WHERE id_st='$ide'"
```

Potvrzovací okno odstranění je vidět na Obr. 13. Můžeme si v něm přečíst veškerá data, která a odstraňovaném víme a ujistit se tak, že si přejeme odstranit skutečně tento subjekt.

Opravdu si přejete odebrat tohoto studenta?				
ID	Jméno	Příjmení	E-mail	Třída
s0007	Petr	Adolf	Adas@centrum.cz	Mch1
				Ano Ne

Obr. 13. Potvrzení odebrání studenta z databáze

Nyní máme vytvořený již celý web přístupný jak běžným uživatelům, tak uživatelům zaregistrovaným. Na Obr. 14 je celý vzhled webu (soubor index.html). Takto se stránky zobrazí každému uživateli, který do svého prohlížeče zadá www adresu Střední odborné školy a středního odborného učiliště technického v Přerově.



Obr. 14. Úvodní stránka webu

K tomu, aby bylo možné vyzkoušet veškeré možnosti, které vytvořený web nabízí na lokální stanici, je zapotřebí mít na této stanici nainstalován PHP server např. Apache a databázový server MySQL. Existuje celá řada aplikací, které tyto servery obsahují v jednom balíku a nemusíme je instalovat každou zvlášť. Tyto aplikace také samy zajistí správné nastavení těchto serverů. Jsou to např. PHP Triad, Uniserver apod.

Bez nainstalování a spuštění těchto serverů nebudou stránky na lokální stanici fungovat. Pro nahlédnutí na vytvořené stránky je tu jediný soubor, pro který tyto servery nainstalované být nemusí, *index.html*. Tento můžeme spustit v kterémkoliv internetovém prohlížeči.

4 IMPLEMENTACE STRÁNEK NA SERVER

Implementace stránek na webový server je posledním, ale velmi důležitým krokem. K čemu by nám byly vytvořené stránky, kdybychom je žádným způsobem neprezentovali. Proto je nutné najít poskytovatele, který nám toto umožní. Službě, která se zabývá poskytnutím námi vytvořených stránek do Internetu se říká Webhosting. Existuje celá řada poskytovatelů těchto služeb. Liší se od sebe formou a možnostmi poskytování a samozřejmě také cenami, které si za to účtují. Proto bychom měli pečlivě vybírat takového poskytovatele, který nám zajistí vše co žádáme s vysokou úrovní kvality a za přijatelnou cenu. Samozřejmě bychom neměli opomenout i naše plány do budoucna a ujistit se, že budeme mít možnost si tyto služby dále rozšiřovat.

4.1 Webhosting stránek SOŠ a SOU technického v Přerově

Střední odborná škola a střední odborné učiliště technické v Přerově využívá k webhostingu svých dosavadních stránek Kabelovou televizi Přerov (KTV Přerov). KTV Přerov není však pouze poskytovatelem webhostingu. Zprostředkovává také připojení do Internetu, prostřednictvím rozvodů kabelové televize, po celém území města Přerova. KTV Přerov nabízí celou řadu programů jak u poskytování připojení k Internetu, tak i u webhostingu a mnoho dalších. V této době používá SOŠ a SOU technické v Přerově program s názvem Webhosting STANDARD. Zde jsou uvedeny služby programu Webhosting STANDARD:

- Adresa: www.vase_domena.cz
- Kapacita: 50 MB pro data
- Neomezený FTP přístup
- PHP skripty
- 10 mailboxů o kapacitě 10MB
- Cena bez DPH 168 Kč (s DPH 200 Kč) měsíčně, platné ke dni 2. 5. 2005

Tento program byl doposud plně dostačující pro poskytování webhostingových služeb stávajícím stránkám. Ovšem při přechodu na nové stránky bude nutnost přejít z tohoto programu na jiný. Hlavním důvodem je použití databáze MySQL u těchto stránek. Stávající program Webhosting STANDARD databáze MySQL nepodporuje. Z tohoto

důvodu poskytuje KTV Přerov další program, Webhosting PROFI, který MySQL podporuje. Služby tohoto programu jsou následující:

- Adresa: www.vase_domena.cz
- Kapacita – 100 MB pro data
- Neomezený FTP přístup
- PHP skripty
- 10 mailboxů o kapacitě 10 MB
- Databáze MySQL
- Cena bez DPH 294 Kč (s DPH 350 Kč) měsíčně, platné ke dni 2. 5. 2005

Tato změna programu je naplánovaná na konec měsíce srpna spolu s implementací stránek na server. Hlavním důvodem, proč proběhne implementace až v tomto čase, je ten, že studenti co ve školním roce 2004/2005 studují závěrečný ročník školu opustí a 1. 9. 2005 nastoupí nově přijatí studenti. Zákonným zástupcům těchto studentů již budou na začátku školního roku přidělena jejich přihlašovací jména a hesla spolu s ostatními zákonnými zástupci jejichž děti budou v školním roce 2005/2006 na škole studovat. Informace o programech byly převzaty z <http://www.ktvprerov.cz>.

ZÁVĚR

V této bakalářské práci je řešena problematika tvorby webových stránek s využitím databází. Konkrétně popis řešení tvorby stránek pro SOŠ a SOU technickou v Přerově.

V první části této práce se zabývám aktuálním stavem současných stránek. Poukazuji na jejich vývoj od počátků až po současnost. Je popsáno jaké technologie jsou použity, jejich technické řešení a jaké jsou jejich možnosti.

Druhá část představuje teorii tvorby webových aplikací a poskytuje popis těchto technologií. Představuji značkovací jazyk HTML, jak po stránce teoretické, tak i představením důležitých příkazů sloužících k formátování výsledného dokumentu. Stejně jako u jazyku HTML, tak i jazyk PHP je představen po stránce teoretické. Poukazuji na jeho historii vývoje, tak i jeho praktické využití s ohledem na proměnné, cykly a větvení programů. Poslední částí, kterou se zabývám v teoretické oblasti jsou databáze. Zaměřuji se na jejich technické možnosti a především na popis nejdůležitějších a nejpoužívanějších příkazů. Zejména ohledně tvorby entit, atributů a samozřejmě prací s uloženými daty.

V praktické části řeším již konkrétní tvorbu webových stránek. Zadání se opíralo o požadavky, které byly kladeny SOŠ a SOU technickým v Přerově. Z velké části bylo zapotřebí se držet těchto kladených požadavků.

Nejprve tedy popisuji tvorbu datového modelu plánované databáze dle požadavků zadavatele. Takto navrženou databázi jsem následně vytvořil pomocí aplikace PhpMyAdmin 2.6.0 a naplnil ji veškerými daty, která zadavatel poskytl.

Následně je popsáno vytvoření webových stránek a to ve dvou krocích. V prvním z hlediska všech uživatelů, pomocí kaskádových stylů CSS, a v druhém z hlediska uživatelů, kteří budou mít možnost vstupu do zabezpečených částí. Zabývám se způsobem přihlášení pomocí přihlašovacího jména a hesla, s využitím skriptovacího jazyka PHP a databáze MySQL, a důvody, které nutí uživatele se do zabezpečených sekcí přihlásit. Pohlížím na jednotlivé skupiny těchto uživatelů a pro každou představuji jejich možnosti, které jsem pro ně vytvořil. Jako je zápis sdělení vyučujícími a následné čtení zákonnými zástupci, prohlížení tříd studentů a také práce s daty v databázi určená pro administrátora (správce sítě).

Nakonec představuji možnosti umístění vytvořených stránek na webový server. Poukazují především na stávající způsob umístění a na důvod, proč současný stav změnit. Jedná se především o rozšíření získaných služeb od poskytovatele.

Do budoucna je počítáno s tím, že budou stránky dále rozšiřovány. Především zabezpečená část a širší okruh jejich uživatelů. Tím nejdůležitějším tedy bude rozšíření databáze, konkrétně tabulky *user*, o další uživatele, studenty. Stejně jako ostatní uživatelé, obdrží svá přihlašovací jména a hesla a po přihlášení budou mít možnost zjistit zadání nových úkolů a prací nebo budou moci odevzdat již vypracované.

SEZNAM POUŽITÝCH MATERIÁLŮ

- [1] BRÁZA, Jiří. PHP 4: Praktické příklady. 1. vyd., Praha: Grada, 2003. 224 s. ISBN: 80-247-0441-2
- [2] BROŽA, Petr. Programování WWW stránek pro úplné začátečníky. 1. vyd., Praha: Computer Press, 2000. 164 s. ISBN: 80-7226-278-5
- [3] BROŽA, Petr. Tvorba WWW stránek pro úplné začátečníky. 1. vyd., Brno: Computer Press, 1999. 162 s. ISBN: 80-7226-164-9
- [4] KUČERA, Miroslav. HTML – tipy a triky od profesionálů. 1. vyd., Brno: UNIS Publishing, 2001. 80 s. ISBN: 80-86097-64-1
- [5] MACH, Jakub. PHP pro úplné začátečníky. 2. vyd., Brno: Computer Press, 2003. 169 s. ISBN: 80-7226-834-1
- [6] WILLIAMS, Hugh E., LANE, David. PHP a MySQL. 1. vyd., Praha: Computer Press, 2002. 552 s. ISBN: 80-7226-760-4
- [7] Manuál PHP [online]. [cit. 2005-06-04]. Dostupný z URL: <<http://www.php.net/manual/cs>>.
- [8] PHP krok za krokem [online]. [cit. 2005-02-05]. Dostupný z URL: <<http://www.builder.cz>>.
- [9] Programování: MySQL [online]. [cit. 2005-02-05]. Dostupný z URL: <<http://jaknaweb.com>>.
- [10] Webdesign: PHP [online]. [cit. 2005-05-05]. Dostupný z URL: <<http://www.owebu.cz/main.php>>.
- [11] Jak psát web: o tvorbě internetových stránek [online]. [cit. 2005-23-04]. Dostupný z URL: <<http://www.jakpsatweb.cz>>.
- [12] Web tvorba [online]. [cit. 2005-13-04]. Dostupný z URL: <<http://www.webtvorba.cz>>.
- [13] Root.cz [online]. [cit. 2005-20-05]. Dostupný z URL: <<http://www.root.cz/vyvoj/database>>.
- [14] Kabelová televize Přerov [online]. [cit. 2005-30-05]. Dostupný z URL: <<http://www.ktvprerov.cz>>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ANSI	- Americký standardizační institut
CSS	- Cascading style sheet, kaskádové styly tvorby webu
ERD	- Entity relation diagram, entitě relační diagram
HTML	- Hypertext Markup Language, značkovací jazyk určený k tvorbě webových stránek
MD5	- Šifrovací funkce v MySQL
MySQL	- Aplikace určená k tvorbě relačních databází
PHP/FI 2.0	- Professional Home Page/Form Interpreter verze 2.0, verze jazyka PHP
SOŠ a SOU technické v Přerově	- Střední odborná škola a střední odborné učiliště technické v Přerově
SQL	- Structured Query Language, strukturovaný dotazovací jazyk
WWW	- World wide web
XML	- eXtensible Markup Language

SEZNAM OBRÁZKŮ

Obr. 1. Vztahy mezi pojmy databázové struktury	13
Obr. 2. ERD diagram	24
Obr. 3. Pracovní okno aplikace phpMyAdmin 2.6.0	33
Obr. 4. Stránka poskytující informace o domovu mládeže.....	34
Obr. 5. Grafický návrh stránek	35
Obr. 6. Hotový návrh webu	41
Obr. 7. Přihlašovací okno	42
Obr. 8. Nabídka po přihlášení.....	44
Obr. 9. Formulář vložení zápisu vyučujícího	45
Obr. 10. Historie záznamů	46
Obr. 11. Vložení nového vyučujícího.....	47
Obr. 12. Změna údajů studenta	47
Obr. 13. Potvrzení odebrání studenta z databáze.....	48
Obr. 14. Úvodní stránka webu	49