Informačný systém pre základné a stredné školy

BAKALÁRSKA PRÁCA

Juraj Kerhát

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
KATEDRA INFORMATIKY

9.2.1 Informatika

RNDr. Mária Pastorová

BRATISLAVA 2007

Čestné prehlásenie

Čestne prehlasujem, že túto bakalársku prácu som vypracoval samostatne, len s použitím citovanej literatúry.

….....

Poďakovanie

Ďakujem vedúcej bakalárskej práce RNDr. Márii Pastorovej za cenné rady a pripomienky, ktoré mi pomohli pri písaní tejto práce.

Abstrakt

Cieľom bakalárskej práce bolo vytvoriť webovú aplikáciu Informačný systém pre základné a stredné školy, ktorá by poskytla rozhranie pre komunikáciu medzi zamestnancami školy a jej žiakmi, resp. rodičmi žiakov. K dosiahnutiu tohto cieľa boli použité technológie XHTML, CSS, PHP a MySQL, ktoré spolu poskytujú silný nástroj pre tvorbu dynamických aplikácií dostupných cez sieť Internet.

Kľúčové slová: internetová aplikácia, informačný systém

Obsah

1 Úvod

Informatizácia spoločnosti vo svete napreduje rýchlym tempom. Informačné technológie čoraz viac prenikajú do našeho sociálneho, kultúrneho, či ekonomického života. Komunikačné prostriedky, medzi ktoré patrí Internet poskytujú veľmi silný nástroj pre postupné odstraňovanie geografických, ako aj časových rozdielov medzi účastníkmi komunikácie. Je veľmi dôležité, aby bol tento vývoj podporovaný, ale tiež kontrolovaný a hlavne prospešný pre celú spoločnosť. Vzniká množstvo internetových aplikácií, ktoré majú za cieľ zjednodušiť, či spríjemniť život ich používateľom. Banky poskytujú služby ako internet banking, rôzne televízie poskytujú on-line vysielanie niektorých svojich programov, vzniká množstvo internetových obchodov, vďaka ktorým nemusí zákazník nikam chodiť, ak si chce zakúpiť nejaký tovar a tiež si môže objednať tovar v čase, kedy to jemu vyhovuje, bez nutnosti čakať na otváracie hodiny.

Vytvorenie webovskej aplikácie, ktorá má pomôcť urýchliť, sprehľadniť a v konečnom dôsledku zjednodušiť komunikáciu medzi učiteľmi na školách a ich žiakmi, či rodičmi týchto žiakov je veľmi užitočné a pri pohľade do budúcnosti možno až nevyhnutné. Žiak, ako aj jeho rodič získa prehľad nad svojimi známkami, učiteľ získa väčší priestor pre informovanie žiakov a ich rodičov o dianí na vyučovaní.

V tejto práci sa pokúsim čo najlepšie priblížiť spôsob, akým som postupoval pri vytvorení takejto aplikácie, ktorú som nazval Informačný systém pre základné a stredné školy, akým spôsobom je možné k nej pristupovať, čo môže jednotlivým zúčastneným poskytnúť, ale tiež spôsob, akým môže byť v budúcnosti modifikovaná a rozširovaná.

Bakalárska práca je rozdelená na osem kapitol, pričom prvá a posledná kapitola predstavujú úvod a záver. Druhá kapitola je venovaná analýze Informačného systému a sú v nej uvedené základné funkcie, ktoré by mal takýto systém poskytovať. Tretia kapitola je venovaná návrhu tried a modelu databázy pre tento systém. Vo štvrtej kapitole nájde čitateľ pohľad na potrebu použiteľnosti a prístupnosti internetových aplikácií. Piata kapitola popisuje samotnú realizáciu Informačného systému a tiež informácie o technológiách, ktoré boli pre túto realizáciu vybrané. Šiesta kapitola uvádza možnosti, ako sa dá Informačný systém v prípade potreby modifikovať a rozširovať. Siedma kapitola popisuje spôsob inštalácie tohto systému.

2 Analýza

Pri tvorbe Informačného systému pre základné a stredné školy je potrebné mať na pamäti niekoľko skutočností. Jedná sa o systém, ku ktorému budú pristupovať jeho používatelia prostredníctvom siete Internet. Medzi týchto používateľov patria hlavne učitelia a žiaci, resp. ich rodičia. Znamená to teda, že cieľoví používatelia môžu mať rôzne technické možnosti, ako sú napríklad rýchlosť pripojenia k Internetu, rôzne prehliadače, rôzne rozlišenia obrazovky, či dokonca rôzne zobrazovacie zariadenia. Taktiež to budú používatelia rôznych vekových kategórii s rôznymi znalosťami a skúsenosťami s používaním Internetu a aplikácií na ňom dostupných. Je preto veľmi dôležité venovať patričnú pozornosť prístupnosti a použiteľnosti tohto systému.

Naopak, nie je príliš dôležité sa zameriavať na „vizuálnu krásu“ systému, ktorá by jednak bola pri takom rozmanitom spektre používateľov veľmi ťažko dosiahnuteľná, ale hlavne by mohla narušiť spomínanú prístupnosť, alebo použiteľnosť. Napríklad využitie veľkého množstva, alebo aj malého množstva príliš veľkých, obrázkov by mohlo znemožniť prístup k aplikácii používateľom s pomalým pripojením k Internetu. Navyše, to, čo niekto môže považovať za pekné, niekomu inému môže pripadať neprehľadné a môže mu to znepríjemňovať prácu s týmto systémom. Základným prvkom pri návrhu vzhľadu systému by preto mala byť jednoduchosť a prehľadnosť.

Ako som už spomenul, Informačný systém pre základné a stredné školy je aplikácia určená pre učiteľov a žiakov, resp. rodičov týchto žiakov. Aby tento systém naplnil svoju podstatu, mal by spĺňať určité požiadavky z hľadiska funkcionality.

Z pohľadu žiaka predstavuje tento systém predovšetkým akúsi virtuálnu žiacku knižku. Znamená to, že žiak by mal mať prístup ku všetkým svojim známkam z jednotlivých predmetov. Mal by mať možnosť vidieť dátum obdržanej známky, typ známky (známka z testu, známka za odpoveď a pod.) a prípadne tiež poznámku (krátku textovú správu) učiteľa k danej známke, ak učiteľ „priloží“ takúto poznámku k známke.

Okrem zobrazenia jednotlivých známok by mal poskytovať tento systém informáciu o priemere známok z jednotlivých predmetov aj s porovnaním s priemernou známkou za celú triedu, do ktorej žiak patrí, pre daný predmet.

Ďalšou požadovanou funkciou pre žiakov a ich rodičov je možnosť zanechať

správu učiteľovi. Táto funkcia by mala byť obmedzená len na možnosť zanechať správu učiteľovi niektorého z predmetov, ktoré žiak navštevuje a svojmu triednemu učiteľovi. Mala by slúžiť hlavne rodičovi žiaka, ktorý môže takýmto spôsobom napríklad ospravedlniť svoje dieťa z neprítomnosti na vyučovaní. Aby nemohla byť takáto možnosť zneužitá žiakom k ospravedlňovaniu svojej neprítomnosti na vyučovaní bez vedomia svojho rodiča, je potrebné znemožniť zmazanie takejto správy žiakom, aby jeho rodič si ju mohol kedykoľvek prečítať.

Žiak by mal mať tiež prístup k informácii o tom, do ktorej triedy patrí a či patrí aj do nejakých skupín, keďže na jednotlivých predmetoch môžu byť žiaci rozdelení do skupín.

Užitočné pre žiaka je tiež zobrazenie jeho rozvrhu hodín. Tento by mal obsahovať meno predmetu, deň v týždni, čas vyučovania predmetu, miestnosť, kde sa predmet vyučuje a tiež meno učiteľa, ktorý predmet vyučuje. Je možné zobraziť na hlavnej stránke len stručný rozvrh všetkých predmetov, ktorý by obsahoval len deň v týždni, čas vyučovania a meno predmetu. Na informačnej stránke konkrétneho predmetu by sa potom zobrazoval podrobnejší rozvrh pre tento predmet.

Z pohľadu učiteľa zas predstavuje tento systém akúsi virtuálnu triednu knihu, do ktorej môže zapisovať známky pre žiakov, ktorých učí. Mal by mať preto prístup k formuláru na zapisovanie známok žiakom z tried alebo skupín, ktoré učí. Taktiež by mal mať učiteľ možnosť zmeniť už zapísanú známku a tiež uzavrieť žiakovi konečnú známku z predmetu, ktorý učí, za aktuálny školský polrok.

Ak je učiteľ triednym učiteľom niektorej triedy, mal by mať prístup k informáciam o jednotlivých žiakoch svojej triedy. Tiež by mal mať právo tieto informácie meniť.

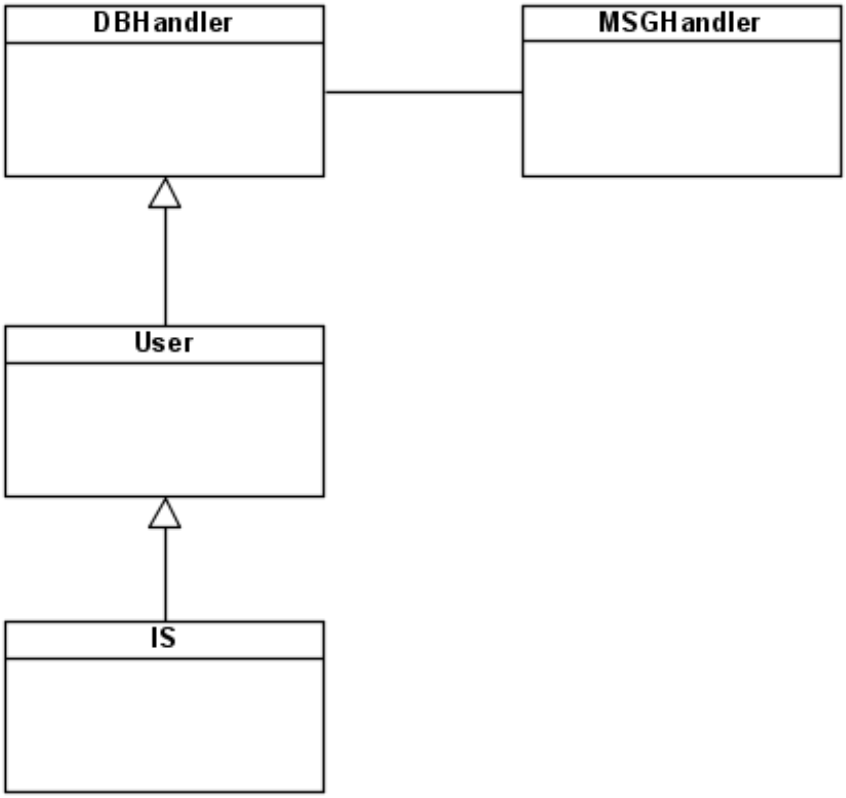
Tak ako žiak, aj učiteľ by mal mať možnosť posielať správy. Tieto by mali byť adresované žiakovi, alebo celej skupine, či triede. Takéto správy môžu predstavovať rôzne oznamy učiteľa pre žiakov, ktorých učí ohľadom predmetu, ktorý ich učí. Takisto by to mohli byť oznamy učiteľa pre rodiča žiaka, prostredníctvom ktorých by rodiča mohol informovať o aktivite, resp. neaktivite jeho dieťaťa na vyučovaní a podobne.

3 Návrh

V tejto kapitole sa zameriam na návrh Informačného systému pre základné a stredné školy. Popíšem návrh tried pre tento systém a takisto uvediem model databázy, s popisom jednotlivých tabuliek databázy.

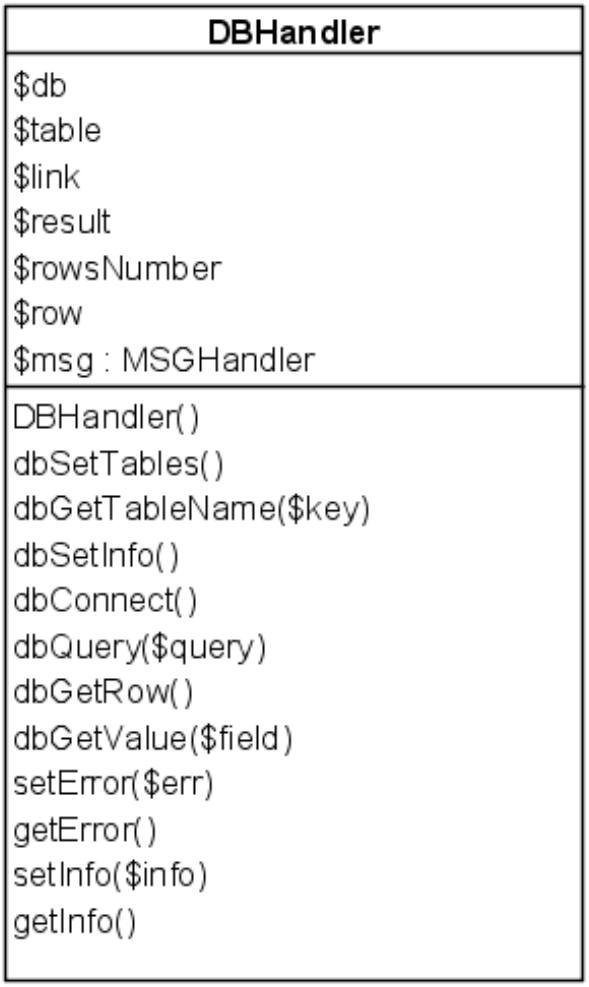
3.1 Návrh tried

Pri návrhu a následne implementácii Informačného systému pre základné a stredné školy som využil objektovo orientované programovanie. Vytvoril som štyri triedy, ktoré tvoria základ tohto systému. Tri z nich vznikli s cieľom, aby mohli byť znovu použité aj pre iné projekty, v ktorých je potrebné, aby používateľ mohol manipulovať s databázou a aby mu systém hlásil rôzne správy chybového, alebo informatívneho charakteru. Sú to triedy DBHandler, MSGHandler a User. Trieda IS (Information System / Informačný systém), ktorá rozširuje triedu User je jediná trieda vytvorená výhradne pre účely Informačného systému pre základné a stredné školy.



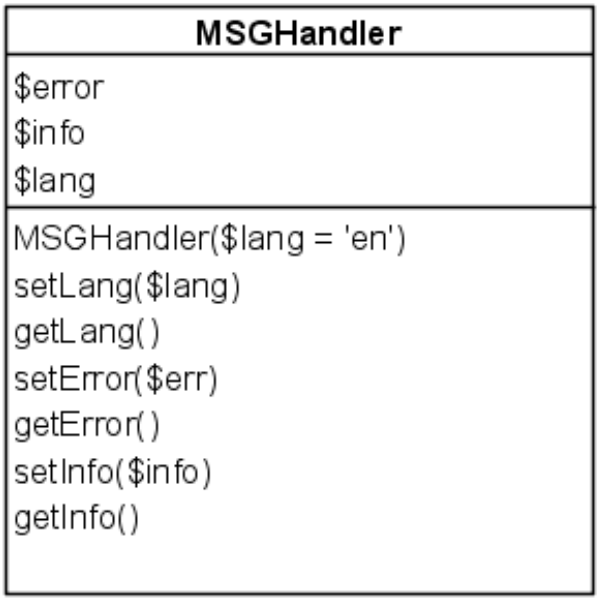
Obr. 1: Diagram tried Informačného systému pre základné a stredné školy

Základnou triedou, ktorá má na starosti komunikáciu s databázou je trieda DBHandler. Táto trieda poskytuje triedam od nej odvodených jednoduché a pohodlné nástroje na pripojenie sa k databáze, výber či vloženie údajov do databázy a vďaka vnorenej triede MSGHandler tiež správu chybových a informačných hlásení.



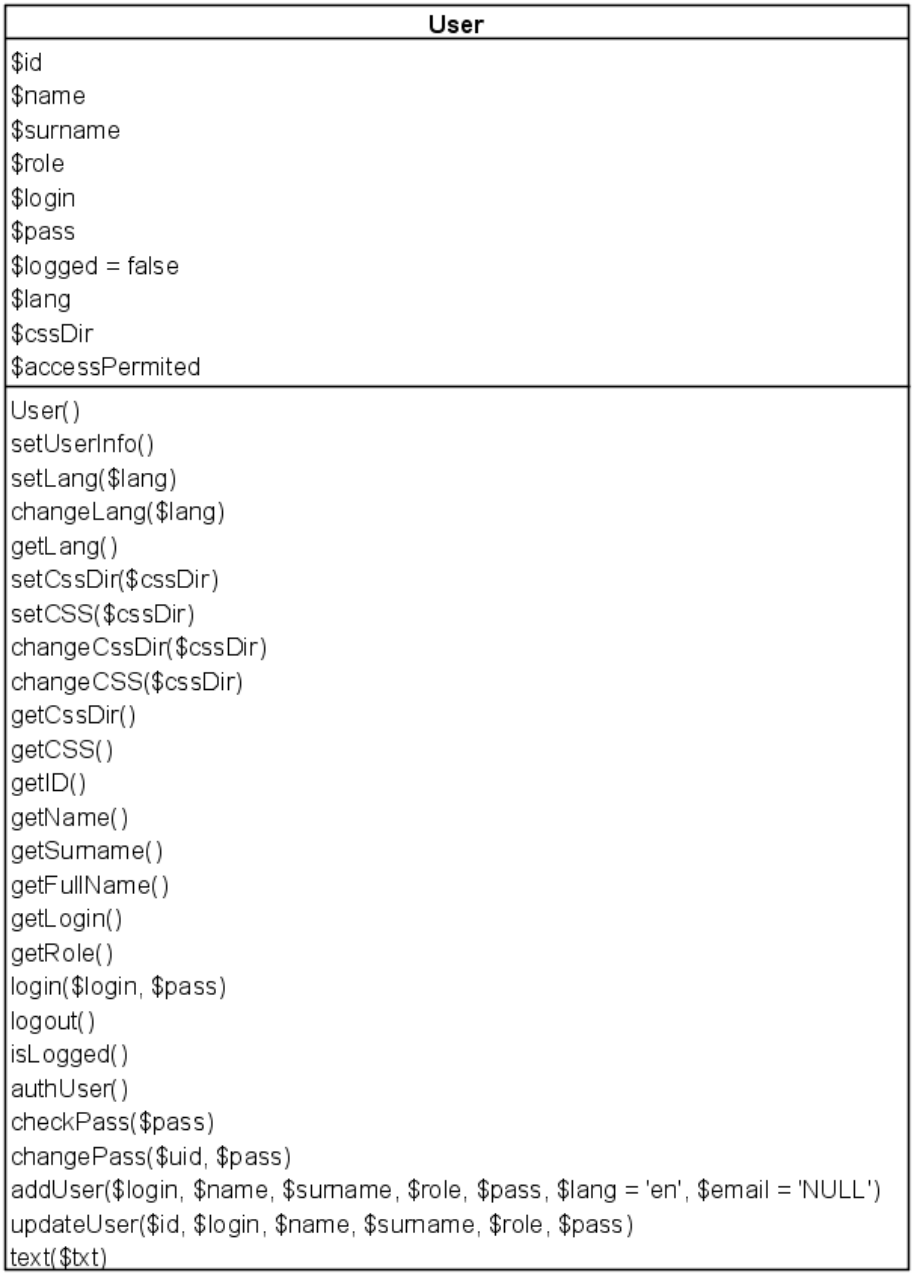
Obr. 2: Atribúty a metódy triedy DBHandler

Počas používania Informačného systému pre základné a stredné školy môže dôjsť k istým situáciám, ktoré znemožnia odoslať klientovi správny (očakávaný) výstup. Napríklad môže zlyhať pripojenie k databáze, alebo môže dôjsť k chybe pri dotaze na databázu a podobne. Je vhodné takéto „nečakané“ situácie rozpoznať a informovať o nich používateľa chybovými správami. Takisto môže byť užitočné poskytnúť používateľovi rôzne správy informatívneho charakteru (nesúvisiace s chybami systému). Napríklad správa s informáciou o nesprávnom vyplnení formulára, správa o úspešnom vykonaní požadovanej akcie a podobne. O manipuláciu s takýmito chybovými a informatívnymi správami sa stará trieda MSGHandler.



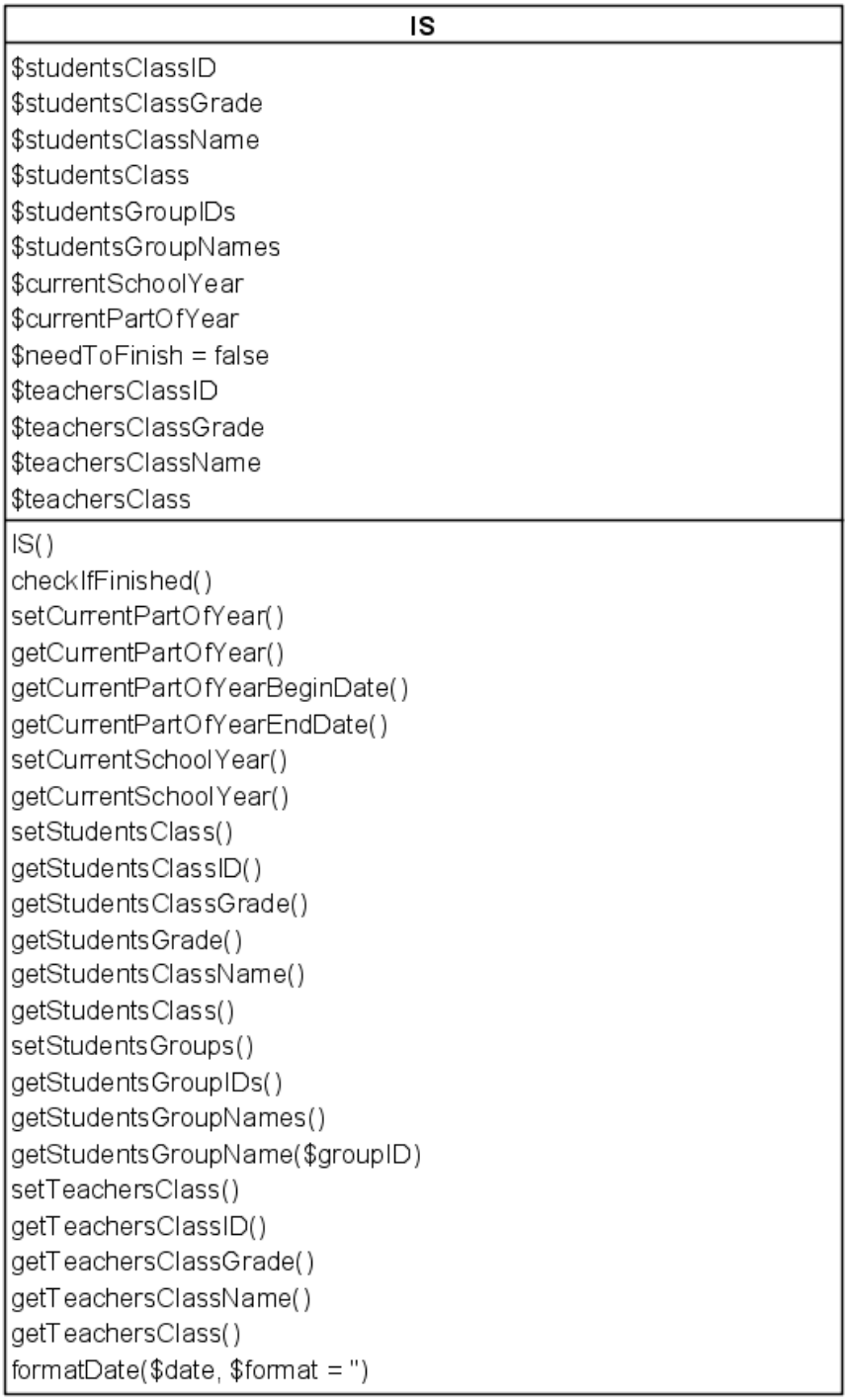
Obr. 3: Atribúty a metódy triedy MSGHandler

Vlastnosti a správanie používateľa špecifikuje trieda User. Keďže používateľ potrebuje komunikovať s databázou, je táto trieda odvodená od triedy DBHandler. Umožňuje používateľovi prihlásiť sa do systému a ďalej s ním pracovať, resp. pracovať so systémom bez prihlásenia, ale s obmedzenými právomocami.



Obr. 4: Atribúty a metódy triedy User

Triedu User rozširuje trieda IS (Information System / Informačný Systém), ktorá obsahuje atribúty a metódy vytvorené špeciálne pre Informačný systém pre základné a stredné školy.

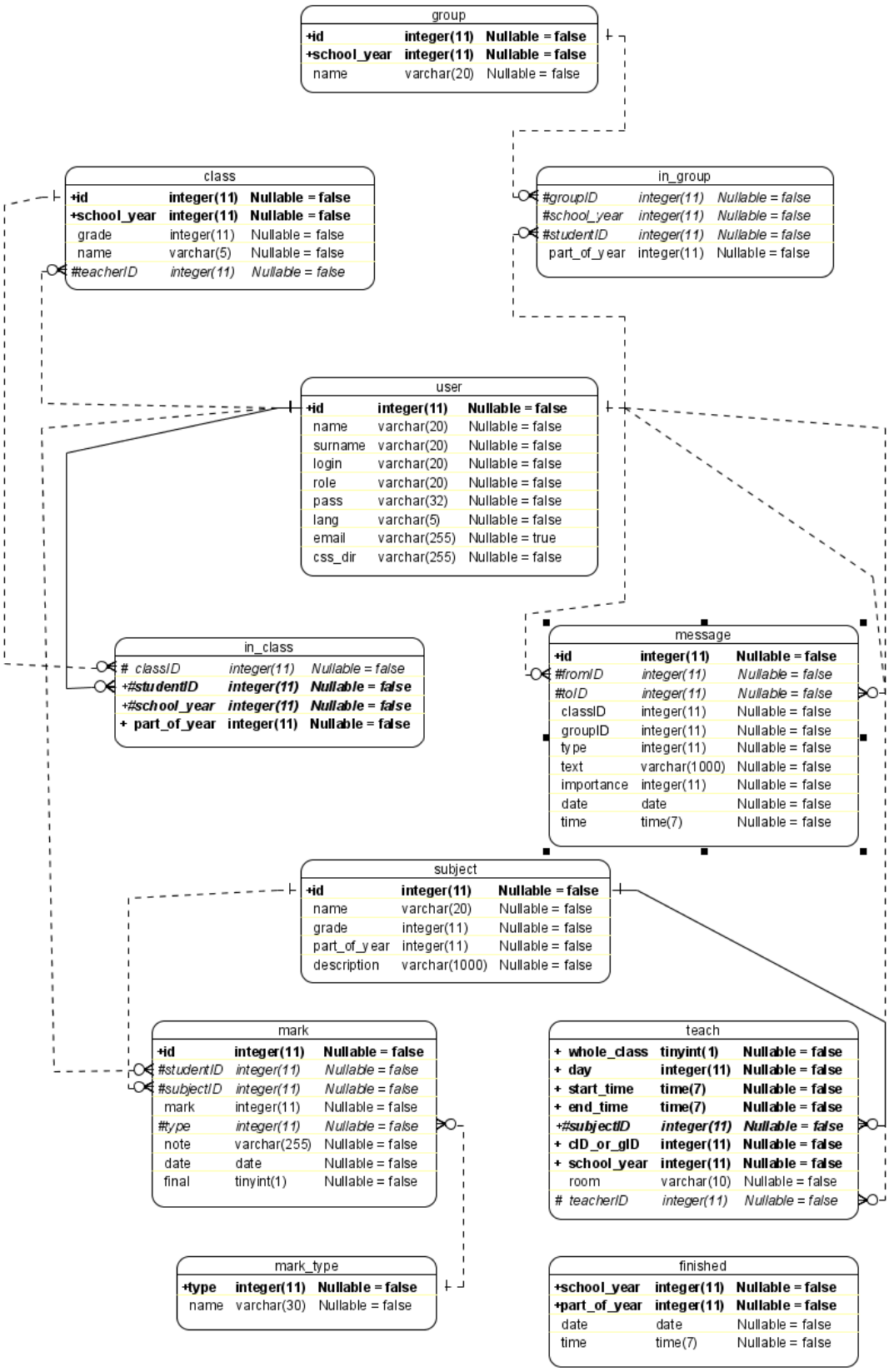


Obr. 5: Atribúty a metódy triedy IS

3.2 Návrh databázy

Skôr, ako som začal pracovať na samotnej aplikácii Informačného systému pre základné a stredné školy, bolo nevyhnutné navrhnúť model databázy, v ktorej bude tento systém uchovávať potrebné údaje. Je potrebné ukladať informácie o jednotlivých používateľoch (ziakoch, uciteloch, …), o triedach, skupinách, predmetoch, známkach a podobne. Vzniklo preto niekoľko tabuliek, ktoré sú navzájom rôzne poprepájané.

- **user** - uchováva všetky informácie o jednotlivých používateľoch, ako napríklad meno, priezvisko, prihlasovacie meno, heslo, ale tiež jazyk a štýl, v ktorom sa majú stránky Informačného systému danému používateľovi zobrazovať
- **class** – v tejto tabuľke sú uložené informácie o jednotlivých triedach, ktoré sú na škole – školský rok, ročník (v danom školskom roku), meno, ID používateľa, ktorý je triednym učiteľom danej triedy
- **group** – informácie o vytvorených skupinách
- **in_class** – táto tabuľka určuje, ktorí žiaci patria do ktorej triedy (pre konkrétny školský rok a jeho časť)
- **in_group** – tabuľka in_group informuje o tom, ktorí žiaci patria do ktorých skupín
- **subject** – informácie o predmetoch, ktoré sa na škole učia
- **teach** – táto tabuľka zaznamenáva informácie o tom, kedy (ktorý deň a o akom čase) sa učí konkrétny predmet (v konkrétnom školskom roku), v ktorej miestnosti sa učí, ktorej triedy, resp. skupiny sa to týka a kto ho učí
- **mark** – informácie o známkach – z ktorého predmetu je námka, pre ktorého žiaka, typ známky (presnejšie identifikátor typu známky), poznámka učiteľa k známke, dátum a tiež informácia, či sa jedná o finálnu známku z predmetu
- **mark_type** – pomenovanie typov známok (odpoveď, písomka, …)
- **message** – správy používateľov – obsahuje identifikátor používateľa, ktorý napísal správu, identifikátor používateľa, pre ktorého je správa určená, alebo identifikátor triedy, ak je správa určená pre celú triedu, alebo identifikátor skupiny, ak je spáva určená pre skupinu, text správy, dátum a čas uloženia správy
- **finished** – informácie o tom, ktoré časti školského roku (resp. školské roky) už boli ukončené – školský rok, časť školského roku, dátum a čas ukončenia



Obr. 6: Model databázy Informačného systému pre základné a stredné školy

4 Použiteľnosť a prístupnosť

Informačný systém pre základné a stredné školy je aplikácia určená pre žiakov a učiteľov základných a stredných škôl. Jej hlavným cieľom je poskytnúť akési rozhranie pre komunikáciu medzi zúčastnenými cez Internet. To si vyžaduje, aby mali všetci žiaci aj učitelia pohodlný prístup k tejto aplikácii bez akéhokoľvek obmedzenia. Jedinou nutnou podmienkou pre prístup k tomuto systému by malo byť pripojenie k sieti Internet. Naopak, nie je žiadané, aby boli jednotliví zúčastnení obmedzovaní nutnosťou výberu konkrétneho operačného systému, webového prehliadača, rozlíšenia obrazovky a podobne.

4.1 Štandardy

V súčasnosti existuje mnoho rôznych inetrnetových preliadačov. Tvorcovia každého z nich sa snažia poskytnúť používateľom rôzne možnosti ovládania, rôzne grafické prvky, ale tiež rôzne možnosti rozšírenia, s cieľom odlíšiť sa od svojich konkurentov. Zároveň ale tvorcovia týchto prehliadačov implementujú rôzne algoritmy na spracovanie a zobrazovanie webových dokumentov, ktoré dostávajú od serverov. Aby nedošlo k situácii, kedy by bolo potrebné pri tvorbe internetových aplikácií vopred stanoviť, pre ktorý internetový prehliadač bude daná aplikácia určená a tomu prispôsobovať formát dokumentov, ktoré bude server posielať prehliadaču, je nutné webové dokumenty štandardizovať.

O štandardizáciu webových dokumentov sa stará konzorcium W3C (The World Wide Web Consortium). Pre účely prezentácie na Internete vznikol typ dokumentu HTML (HyperText Markup Language), ktorý popisuje štandard HTML 4.01. Tento štandard je v súčasnosti zastaralý a nie je odporúčané ho ďalej používať pri tvorbe nových webových dokumentov. Niektoré z hlavných dôvodov sú, že takýto typ dokumentu je príliš voľný a povoľuje rôzne konštrukcie elementov a tiež rôzne atribúty pre niektoré elementy, ktoré slúžia napríklad len na vizuálnu úpravu daného elementu.
Je napríklad povolené neuzatvárať elementy ukončovacími tagmi:

```
<p>Toto je neuzavretý paragraf
```

```
<p>A tu už začína ďalší paragraf (ktorý už je uzavretý)</p>
```

Takisto je povolené neuzatvárať prázdne elementy:

```
Obrázok: <img src="obrazok.jpg" alt="Obrazok">
Koniec riadku: <br>
```

Nezáleží na veľkosti písmen v menách tagov pre jednotlivé elementy:

```
<p>Paragraf 1</p>
<P>Paragraf 2</P>
<P>Paragraf 3</p>
```

Dokonca nie je nutné vkladať všetky elementy do jedného koreňového elementu (<html>):

```
<head><title>Toto je nadpis dokumentu definovaný v jeho
hlavičke.</title></head>
<body>Toto je telo dokumentu.</body>
```

Takáto voľnosť je významná hlavne pre tvorcov webových dokumentov, ktorým uľahčuje prácu tým, že ich neobmedzuje nutnosťou dodržiavať presne stanovené pravidlá. Na druhej strane ale vytvára problém s dosahovaním prístupnosti takýchto dokumentov. Prehliadače, ktoré majú tieto dokumenty zobrazovať sa musia vysporiadať s tým, že niektoré elementy v danom dokumente sú uzavreté a iné nie, že niektoré mená tagov sú napísané malými písmenami a iné veľkými a pod. Napríklad sa musia rozhodnúť, ako zobraziť nasledujúcu časť dokumentu:

```
<p>Text 1<em>Text 2</em>
```

V uvedenom príklade chýba paragrafu ukončovací tag. Ale kam ho treba doplniť? Nie je totiž jasné, akým spôsobom sú tieto elementy do seba včlenené. Majú sa zobraziť nasledovným spôsobom?:

```
<p>Text 1</p><em>Text 2</em>
```

Alebo skôr nasledovne?:

```
<p>Text 1<em>Text 2</em></p>
```

Keďže každý z prehliadačov môže riešiť takéto nejednoznačnosti „po svojom“, môže sa dokument zobraziť v rôznych prehliadačoch rôzne a u niektorých to môže byť výrazne odlišné od pôvodného zámeru autora dokumentu.
Zároveň používanie atribútov určených len na vizuálnu úpravu vzhľadu dokumentu nemusí byť podporované niektorými prehliadačmi, napríklad v mobilných telefónoch, alebo iných zariadeniach.

Oveľa striktnejším typom dokumentov je XHTML (eXtensible HyperText Markup Language). Tento typ predstavuje akýsi prechod medzi HTML (HyperText Markup Language) a XML (eXtensible Markup Language). Definuje elementy využívané v HTML (resp. len ich podmnožinu takú, ktorá neobsahuje elementy slúžiace len na vizuálne formátovanie textu), ale tak ako XML, vyžaduje dodržiavanie prísnych syntaktických pravidiel. Je nevyhnutné, aby všetky elementy boli uzavreté:

```
<p>Uzavretý paragraf</p>
```

To platí aj pre prázdne elementy:

```
Obrázok: <img src="obrazok.jpg" alt="Obrazok" />
Koniec riadku: <br />
```

Mená všetkých elementov je nutné písať malými písmenami:

```
<p>Paragraf</p>
```

Takisto mená atribútov je nutné písať malými písmenami. Navyše všetky použité atribúty musia obsahovať hodnotu (nie sú povolené skrátené formy atribútov), ktorá musí byť uzavretá v úvodzovkách (resp. apostrofoch):

```
<img src="obrazok.jpg" alt='Obrazok' />
```

Všetky elementy musia byť potomkom jedného koreňového elementu (<html>):

```
<html>
<head>...</head>
<body>...</body>
</html>
```

4.2 Validita

Aby vedel prehliadač určiť, aký typ dokumentu práve dostal od servera, s ktorým komunikuje, a podľa ktorých pravidiel ho má zobraziť, je potrebné tento typ, aj štandard, ktorý ho popisuje, deklarovať na začiatku dokumentu. Niektoré z možných deklarácií typu dokumentu sú:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Samotná deklarácia typu dokumentu však nestačí k tomu, aby bol dokument „dobrý“. Je potrebné, aby tiež spĺňal všetky podmienky, ktoré určuje špecifikácia daného typu dokumentu uvedená v deklarácii typu. Pre kontrolu validity dokumentu je možné použiť napríklad nástroj vyvinutý samotným konzorciom W3C, The W3C Markup Validation Service, dostupný na stránkach http://validator.w3.org/. Tento nástroj poskytuje možnosť skontrolovať dokument zadaním URL adresy, na ktorej sa nachádza, ale tiež uploadovaním lokálne uloženého dokumentu či priamym vložením obsahu dokumentu do textového poľa formulára pre kontrolu správnosti (validity) dokumentu.

Všetky stránky Informačného systému pre základné a stredné školy sú dokumenty validné podľa štandardu XHTML 1.1. Ich obsah je rozdelený pomocou elementov DIV (division) na niekoľko základných častí, ktoré sú identifikované pomocou atribútu ID. Napríklad element DIV, ktorý predstavuje hlavičku stránky má definovaný atribút ID s hodnotou „header“. Podobne pre element DIV s navigáciou stránky je definovaný atribút ID s hodnotou „navigation“, pre DIV s hlavným obsahom stránky je to atribút ID s hodnotou „main“ atď. Takéto identifikovanie uľahčuje napríklad rozširovanie systému o nové kaskádové štýly. V rámci týchto častí sú definované len také elementy, ktoré štandard XHTML 1.1 povoľuje a ktoré sú pre zobrazenie určitej časti stránky určené. Napríklad pre zoznam sú použité elementy UL a LI a nie napríklad element P s textovými položkami oddelenými elementom BR, ani tabuľka (element TABLE), či iné elementy, ktoré na to nie sú určené.

4.3 Kódovanie UTF-8

Každý XHTML dokument by mal informovať prehliadač, ktorý ho má zobraziť, o tom, aké kódovanie textu je v ňom použité. Bolo by totiž nežiadané, keby prehliadač nesprávne zobrazil niektoré znaky a dokument by sa tak stal nečitateľným (a teda nepoužiteľným). Aby sme zvolili ten správny typ kódovania, musíme si najprv uvedomiť, aký jazyk, resp. aké znaky budú v dokumente použié. Keďže snahou pri vývoji Informačného systému pre základné a stredné školy bolo poskytnúť možnosť pridávania rôznych jazykových verzií tohto systému, je potrebné použiť kódovanie, ktoré podporuje znaky rôznych jazykov. Práve takým kódovaním je kódovanie UTF-8 (Unicode Transformation Format-8).

5 Realizácia

V tejto kapitole sa pokúsim priblížiť spôsob, akým som realizoval Informačný systém pre základné a stredné školy. Uvediem výber technológií, ktoré som pri tejto realizácii využil, popíšem jednotlivé triedy, ktoré som vytvoril, ich atribúty a metódy.

5.1 Výber technológií

Pre implementáciu Informačného systému pre základné a stredné školy bolo potrebné najskôr zvoliť vhodné nástroje. Tento systém nie je len zoskupením statických dokumentov, ktoré by boli uložené niekde na serveri a na žiadosť klienta bez zmeny odoslané cez Internet, aby ich mohol klient prečítať a zobraziť. Jedná sa o plne dynamickú aplikáciu, ktorá môže získavať rôzne údaje uložené v databáze, resp. vkladať informácie do databázy. Údaje získané z databázy, alebo od klienta môže ďalej spracovávať a na základe výsledkov spracovania sa ďalej rozhodovať, ktoré príkazy sa vykonajú v ďalšom kroku. Nakoniec sa vygeneruje výstupný dokument, ktorý sa odošle klientovi na zobrazenie.

Ja som si ako hlavný nástroj zvolil skriptovací jazyk PHP, konkrétne jeho verziu 5. Rekurzívna skratka PHP predstavuje „PHP: Hypertext Preprocessor“. PHP je Open Source skriptovací jazyk so všeobecným zameraním, no predovšetkým vhodný pre vývoj webovských aplikácii, ktorý môže byť vložený priamo do HTML. Je to skriptovací jazyk na strane servera. Môže byť použitý na všetkých najpoužívanejších operačných systémoch, akými sú napríklad Linux, rôzne varianty Unixu (napríklad HP-UX, Solaris, OpenBSD), Microsoft Windows, Mac OS X, RISC OS a iné. PHP tiež podporuje väčšinu dnešných webových serverov. Patria k nim napríklad Apache, Microsoft Internet Information Server, Personal Web Server, servre Netscape a iPlanet, Oreilly Website server, Caudium, Xitami, OmniHTTPd a mnohé ďalšie. Pre väčšinu serverov má PHP samostatný modul, pre ostatné, podporujúce CGI štandard, dokáže PHP pracovať ako CGI procesor. [1]

Navyše, okrem voľnosti pri výbere operačného systému a webového servera, poskytuje PHP tiež voľnosť pri rozhodovaní sa medzi procedurálnym, alebo objektovo orientovaným programovaním, alebo spojením procedurálneho a objektovo

orientovaného pogramovania. Práve PHP 5, ktoré je použité aj v Informačnom systéme pre základné a stredné školy, má na rozdiel od predchádzajúcich verzií plnú podporu objektovo orientovaného programovania.

Ďalšou veľkou výhodou jazyka PHP je, že podporuje veľké množstvo databázových systémov. Medzi najznámejšie patria napríklad Informix, InterBase, FrontBase, Direct MS-SQL, MySQL, Oracle, PostgreSQL a mnohé ďalšie. Pre realizáciu Informačného systému pre základné a stredné školy som použil databázový systém MySQL 5.1. MySQL je Open Source databázový systém, ktorý je svojou rýchlosťou, spoľahlivosťou a jednoduchosťou použitia vhodným databázovým systémom pre takú aplikáciu, akou je Informačný systém pre základné a stredné školy.

5.2 Vytvorené triedy

Ako som už spomenul v predchádzajúcich kapitolách, základ Informačného systému pre základné a stredné školy tvoria štyri triedy, ktoré definujú potrebnú funkcionalitu tohto systému. V nasledujúcom texte bližšie popíšem jednotlivé atribúty týchto tried, ako aj význam a princíp jednotlivých metód definovaných pre tieto triedy.

5.2.1 Trieda DBHandler

Informácie o nastaveniach databázy si trieda DBHandler uchováva v atribúte $db. Tento atribút je reprezentovaný ako asociatívne pole textových reťazcov. Jeho hodnoty predstavujú meno servera, na ktorom sa nachádza databáza, meno databázy, meno používateľa, ktorý má právo pracovať s databázou a heslo pre tohto používateľa. Konštruktor triedy DBHandler (metóda DBHandler()) zavolá privátnu metódu dbSetInfo(), ktorá prečíta súbor dbsettings.php a podľa informácií v tomto súbore nastaví hodnoty atribútu $db.

Atribút $table je podobne, ako atribút $db reprezentovaný ako asociatívne pole textových reťazcov. Tieto uchovávajú mená tabuliek v databáze. Konštruktor triedy DBHandler zavolá metódu dbSetTables(). Táto podľa informácií v súbore dbsettings.php nastaví hodnoty atribútu $table. Každá položka poľa $table je zložená z prefixu tabuliek, mena tabuľky a postfixu tabuliek. Prefix aj postfix je rovnaký pre každú tabuľku. Využitie prefixu a/alebo postfixu umožňuje implementovať systém aj s použitím databázy, v ktorej už existujú nejaké tabuľky (napr. pre nejaký iný systém), bez nutnosti premenovania tabuliek, ktorých mená sa zhodujú s menami už

existujúcich tabuliek. Na znázornenie metódy dbSetTables() uvediem jednoduchý príklad:
Chceme vytvoriť jednoduchú aplikáciu, ktorá má obsahovať len jednu tabuľku v databáze s názvom „user“, ktorá bude uchovávať informácie o používateľoch našej aplikácie. Máme však prístup len k jednej databáze, ktorú už ale využíva iná naša aplikácia a zhodou okolností aj tá uchováva informácie o používateľoch v tabuľke s menom „user“. My samozrejme nechceme pomiešať informácie o používateľoch jednej aplikácie s informáciami o používateľoch druhej aplikácie. Taktiež je možné, že stĺpce existujúcej tabuľky „user“ sa nezhodujú so stĺpcami tabuľky „user“, ktorú chceme vytvoriť, pretože jednotlivé aplikácie môžu vyžadovať iný typ informácií o svojich používateľov. Musíme preto k novej tabuľke pridať prefix, alebo postfix, alebo aj prefix, aj postfix. Rozhodneme sa, že použijeme prefix „new_“. V databáze teda vytvoríme novú tabuľku s menom „new_user“. Následne v súbore dbsettings.php nastavíme hodnotu premennej $tablePrefix na „new_“, hodnotu premennej $tablePostfix na „“ (prázdny reťazec) a hodnotu premennej $table['user'] na „user“. Keď sa potom zavolá metóda dbSetTables(), tak tá nastaví hodnotu atribútu $table['user'] tak, že spojí premenné $tablePrefix, $table['user'] a $tablePostfix do jedného reťazca. Hodnota tohto atribútu teda bude „new_user“.

Meno tabuľky uchované v atribúte $table je možné zistiť zavolaním metódy dbGetTableName($key). Táto metóda má jeden parameter, ktorý určuje kľúč v asociatívnom poli $table a vracia hodnotu prvku $table[$key]. Dôvodom pre vznik tejto metódy je hlavne to, že pri prípadnej zmene mien tabuliek, resp. ich prefixov alebo postfixov v databáze nie je potrebné prepisovať všetky dotazy na tabuľky so zmenenými menami. Stačí zmeniť len hodnoty príslušných premenných v súbore dbsettings.php. Taktiež to umožňuje každému, kto sa rozhodne využiť aplikáciu (pri Informačnom systéme pre základné a stredné školy je to konkrétna škola), pomenovať príslušné tabuľky „po svojom“ bez nutnosti zasahovať do kódu aplikácie.

Aby mohla aplikácia komunikovať s databázou, je potrebné sa najprv k databáze pripojiť. To zahŕňa vytvorenie spojenia s databázovým serverom pomocou mena a hesla používateľa a následne výber databázy, s ktorou má aplikácia komunikovať. Aby počas komunikácie medzi klientom (aplikáciou) a databázovým serverom cez vytvorené spojenie nedošlo k nežiadúcej zmene posielaných údajov z dôvodu preloženia údajov do nesprávnej znakovej sady, je tiež potrebné nastaviť znakovú sadu spojenia. Spojenie s databázovým serverom umožňuje v PHP funkcia mysql_connect, ktorej je možné zadať ako parametre meno servera, na ktorom je databáza uložená, meno používateľa, ktorý k nej má prístup a heslo pre tohto používateľa. Návratová hodnota funkcie mysql_connect je identifikátor spojenia, ak sa spojenie podarilo vytvoriť, alebo FALSE, ak sa to nepodarilo. Výber databázy vykoná funkcia mysql_select_db, ktorá potrebuje dva parametre. Jeden určuje meno databázy, ktorú chceme vybrať a druhý je identifikátor spojenia s databázovým serverom. Keď už je vytvorené spojenie s databázou, je možné nastaviť znakovú sadu spojenia. Slúži na to dotaz „SET CHARACTER SET ...“, kde tri bodky treba nahradiť menom znakovej sady. Napríklad pre znakovú sadu UTF8, ktorá je tiež použitá v Informačnom systéme pre základné a stredné školy, vyzerá tento dotaz nasledovne: „SET CHARACTER SET UTF8“. Na posielanie MySQL dotazov databázovému serveru slúži v PHP funkcia mysql_query s dvoma parametrami: samotný dotaz a

identifikátor spojenia.

Je zrejmé, že v celej aplikácii bude pri každom pokuse o komunikáciu s databázovým serverom (ak ešte nie je vytvorené spojenie) potrebné vykonať všetky spomínané kroky: vytvorenie spojenia, výber databázy a nastavenie znakovej sady spojenia. To predstavuje opakované písanie tých istých príkazov v kóde aplikácie. Preto som vytvoril triede DBHandler metódu dbConnect(), ktorá nemá žiadne parametre a vykoná všetky potrebné operácie pre pripojenie sa k databáze. Najprv skontroluje, či už náhodou nie je spojenie vytvorené (či má atribút $link nastavenú hodnotu). Ak spojenie ešte vytvorené nie je, zavolá funkciu mysql_connect, pričom ako parametre použije príslušné hodnoty atribútu $db a identifikátor vytvoreného spojenia uloží ako hodnotu atribútu $link. Následne vyberie databázu pomocou funkcie mysql_select_db s menom databázy (príslušná hodnota atribútu $db) a identifikátorom spojenia (hodnota atribútu $link). Nakoniec nastaví znakovú sadu spojenia na UTF8 funkciou mysql_query, ktorej pošle ako parametre dotaz „SET CHARACTER SET UTF8“ a hodnotu atribútu $link.

Komunikácia medzi klientom a databázovým serverom prebieha tak, že klient pošle serveru dotaz a server mu vráti výsledok aplikácie tohto dotazu na databázu. Toto zabezpečuje v PHP už spomínaná funkcia mysql_query. Túto funkciu nahrádza metóda triedy DBHandler dbQuery($query), ktorá má len jeden parameter $query, ktorý predstavuje dotaz na databázu. Táto metóda zavolá funkciu mysql_query s parametrom $query a atribútom $link ako parametrami tejto funkcie. Navyše uloží výstupnú hodnotu tejto funkcie do atribútu $result a zavolá tiež funkciu mysql_num_rows s parametrom $result, aby zistila počet riadkov výslednej tabuľky dotazu, ktorý uloží ako hodnotu atribútu $rowsNumber. Vďaka tomu nie je potrebné vytvárať v kóde aplikácie lokálne premenné $result a $rowsNumber pre každý dotaz.

Aby sme tiež nemuseli vytvárať lokálnu premennú pre získavanie riadkov výslednej tabuľky dotazu na databázu, obsahuje trieda DBHandler atribút $row, ktorého hodnota predstavuje jeden riadok výslednej tabuľky. Hodnotu tohto atribútu nastavuje metóda dbGetRow(), ktorá nemá žiadne parametre a ako hodnotu atribútu $row nastaví výslednú hodnotu funkcie mysql_fetch_array. Tejto funkcii pošle ako parameter hodnotu atribútu $result. V atribúte $row je teda uložené asociatívne pole, v ktorom kľúč predstavuje meno stĺpca výslednej tabuľky a hodnota pre daný kľúč predstavuje hodnotu v danom riadku a stĺpci výslednej tabuľky. Pre získanie tejto hodnoty je potrebné zavolať matódu dbGetValue($field) s parametrom $field, ktorá vráti hodnotu poľa $row pre kľúč $field.

Trieda DBHandler ešte obsahuje metódy setError($err), getError(), setInfo($info) a getInfo(), ktoré sú ale len premenovaním volania príslušných metód objektu typu MSGHandler, ktorý trieda DBHandler obsahuje (atribút $msg). Tieto metódy popíšem pri popise triedy MSGHandler.

5.2.2 Trieda MSGHandler

Trieda MSGHandler je jednoduchá trieda, ktorá obsahuje tri atribúty: $error, $info a $lang. Atribút $lang uchováva informáciu o tom, aký jazyk má byť použitý pre správy. Nastavuje ho metóda setLang($lang), ktorá je volaná konštruktorom MSGHandler($lang = 'en'). Konštruktor triedy MSGHandler má jeden voliteľný parameter $lang, ktorý ak nie je zadaný, použije sa hodnota „en“. Znamená to, že štandardný jazyk pre správy je angličtina. Metóda getLang() vracia hodnotu atribútu $lang.

Pre nastavovanie správ slúžia metódy setError($err) a setInfo($info), ktoré sú navzájom veľmi podobné. Metóda setError($err) nastavuje chybové správy a metóda setInfo($info) nastavuje informatívne správy. Najprv podľa hodnoty atribútu $lang prečítajú súbor príslušného jazyka, uložený v adresári „lang“ (o jazykoch bude ešte zmienka neskôr). Tento súbor obsahuje jedno asociatívne pole $err_msg s chybovými správami a jedno asociatívne pole $info_msg s informatívnymi správami. Metóda setError($err) teda nastaví hodnotu atribútu $error na hodnotu $err_msg[$err] a metóda setInfo($info) nastaví hodnotu atribútu $info na hodnotu $info_msg[$info]. Metódy getError() a getInfo() slúžia na návrat hodnoty atribútu $error, resp. $info.

5.2.3 Trieda User

Konštruktor triedy User najprv zavolá konštruktor triedy DBHandler, ktorú trieda User rozširuje. Následne zavolá metódu authUser(), ktorá má na starosti autentifikáciu používateľa. Tú vykoná tak, že skontroluje, či sú v premennej $_SESSION uložené informácie o používateľovi. Ak áno, porovná ich s údajmi v databáze a ak sa zhodujú, vráti TRUE, inak vráti FALSE. Ak vráti metóda authUser() hodnotu TRUE, priradí konštruktor triedy User hodnotu TRUE atribútu $logged. Ak je používateľ prihlásený (atribút $logged má nastavenú hodnotu TRUE), zavolá funkciu setUserInfo(), ktorá nastaví hodnoty atribútov $id, $name, $surname, $role, $login, $lang a $cssDir podľa hodnôt premenných $_SESSION['id'], $_SESSION['name'], $_SESSION['surname'], $_SESSION['role'], $_SESSION['login'], $_SESSION['lang'] a $_SESSION['cssDir']. Okrem atribútov $lang a $cssDir sa ostatné atribúty nastavia priamym priradením príslušnej hodnoty premennej $_SESSION. Atribút $lang sa nastaví zavolaním metódy setLang($lang), ktorej sa ako parameter pošle hodnota $_SESSION['lang'] a atribút $cssDir sa nastaví zavolaním metódy setCSS($cssDir), ktorej sa ako parrameter pošle hodnota

$_SESSION['cssDir']. Ak používateľ nie je prihlásený, nastaví konštruktor triedy User len hodnoty atribútov $lang a $cssDir. To vykoná tak, že najprv zistí, či je nastavená hodnota premennej $_COOKIE['lang']. Ak áno, zavolá metódu setLang($lang) a ako parameter jej pošle túto hodnotu, inak zavolá metódu setLang($lang) s parametrom „en“. Takisto zistí, či je nastavená hodnota premennej $_COOKIE['cssDir']. Ak áno, zavolá metódu setCSS($cssDir) a ako parameter jej pošle túto hodnotu, inak zavolá metódu setCSS($cssDir) s parametrom „default“.

Metóda setLang($lang) najprv zistí, či existuje v adresári „lang“ súbor s menom $lang a s príponou „.php“. Ak tento súbor existuje, nastaví hodnotu atribútu $lang na hodnotu parametra $lang, inak nastaví hodnotu atribútu $lang na „en“. Nakoniec zavolá metódu vnorenej triedy MSGHandler msg->setLang($lang) ktorej pošle ako parameter hodnotu atribútu $lang. Metóda changeLang($lang) umožňuje zmeniť jazyk používateľa v závislosti na tom, či je prihlásený, alebo nie. Najprv zavolá vyššie popísanú metódu setLang($lang). Ak je používateľ prihlásený zmení hodnotu poľa „lang“ v tabuľke „user“ pre tohto používateľa a taktiež zmení hodnotu premennej $_SESSION['lang']. Ak nie je používateľ prihlásený, nastaví táto metóda hodnotu premennej $_COOKIE['lang'] zavolaním funkcie setcookie. Metóda getLang() len vracia hodnotu atribútu $lang.

Metódy setCssDir($cssDir), changeCssDir($cssDir) a getCssDir() sa vykonávajú analogicky ako metódy setLang($lang), changeLang($lang) a getLang(). Metóda setCssDir($cssDir) nekontroluje existenciu súboru, ale existenciu adresára s menom $cssDir v adresári „css“. Metódy setCSS($cssDir), changeCSS($cssDir) a getCSS() predstavujú len alternatívne názvy pre metódy setCssDir($cssDir), changeCssDir($cssDir) a getCssDir().

Na získanie informácií o používateľovi slúžia metódy getID(), getName(), getSurname(), getLogin() a getRole(), ktorých návratovými hodnotami sú hodnoty atribútov $id, $name, $surname, $login a $role. Okrem týchto metód je možné použiť tiež metódu getFullName(), ktorá spojí do jedného reťazca hodnoty atribútov $name a $surname, medzi ktoré vloží medzeru a tento reťazec vráti ako návratovú hodnotu.

Keďže je potrebné, aby mal používateľ možnosť sa prihlásiť, resp. odhlásiť, vytvoril som metódy login($login, $pass) a logout(). Metóda login najprv skontroluje, či už náhodou nie je používateľ prihlásený a ak áno, skončí. Inak zašifruje parameter $pass pomocou funkcie md5 a výsledok vloží ako hodnotu atribútu $pass. Následne skontroluje, či existuje v databáze používateľ s príslušným prihlasovacím menom a heslom. Ak nájde v databáze hľadaný záznam, vloží informácie o používateľovi do premennej $_SESSION a do príslušných atribútov triedy User. Nakoniec nastaví hodnotu atribútu $login na TRUE. Ak hľadaný záznam v databáze nenájde, nastaví chybovú správu o nesprávnych prihlasovacích údajoch. Metóda logout() zruší hodnoty $_SESSION a nastaví hodnotu atribútu $logged na FALSE. Pre získanie hodnoty atribútu $logged je možné zavolať metódu isLogged().

Nového používateľa je možné pridať do databázy zavolaním metódy addUser($login, $name, $surname, $role, $pass, $lang = 'en', $email = 'NULL'). Táto metóda jednoducho vloží informácie o novom používateľovi, ktoré dostane ako

hodnoty svojich parametrov, do tabuľky user v databáze. Upraviť informácie o používateľovi možno pomocou metódy updateUser($id, $login, $name, $surname, $role, $pass). Ak je potrebné zmeniť len heslo používateľa, je možné zavolať metódu changePass($uid, $pass).

Aby bolo možné kontrolovať, ktorá používateľská rola môže, resp. nemôže vidieť obsah konkrétnej stránky, musia byť nastavené prístupové práva k danej stránke pre používateľa (rolu). Tieto práva umožňuje nastaviť metóda setAccessPermissions($roles = 'none'). Táto metóda má jeden parameter $roles, ktorý určuje, ktoré roly majú povolený prístup k stránke. Je to textový reťazec, ktorého hodnoty môžu byť „none“, „all“, „logged“, alebo zoznam s povolenými rolami, oddelenými čiarkami. Hodnota „none“ znamená, že žiadny používateľ nemá prístup k stránke, „all“ znamená, že prístup k stránke majú všetci používatelia, „logged“ znamená, že len prihlásení používatelia majú k stránke prístup a zoznam s rolami určuje, že len roly v tomto zozname majú prístup k stránke. Informácia o tom, či má konkrétny používateľ prístup k stránke, alebo nie uloží metóda setAccessPermissions($roles = 'none') ako hodnotu booleovskému atribútu accessPermited. Hodnotu tohto atribútu možno získať zavolaním metódy accessPermited().

5.2.4 Trieda IS

Aby bolo možné poskytnúť používateľovi správne a aktuálne informácie, je potrebné vedieť, ktorý školský rok a tiež, ktorá časť školského roku (na Slovensku polrok) momentálne prebieha. Tieto informácie uchovávajú v triede IS atribúty $currentSchoolYear a $currentPartOfYear. Hodnoty týmto atribútom nastavujú metódy setCurrentSchoolYear() a setCurrentPartOfYear(). Metóda setCurrentPartOfYear() najskôr prečíta súbor schoolsettings.php. V tomto súbore sa nachádza informácia o počte častí školského roku (štandardne 2), ich meno a dátum (deň a mesiac) začiatku a konca každej z týchto častí. Podľa týchto informácií a podľa aktuálneho dátumu vypočíta aktuálnu časť školského roku, ktorú uloží ako hodnotu atribútu $currentPartOfYear. Následne ešte zavolá metódu checkIfFinished(). Metóda setCurrentSchoolYear() takisto prečíta súbor schoolsettings.php a podľa informácií v ňom uložených a podľa aktuálneho dátumu vypočíta aktuálny školský rok a ten uloží ako hodnotu atribútu $currentSchoolYear. Nakoniec zavolá metódu checkIfFinished(). Metóda checkIfFinished() skontroluje podľa vypočítanej aktuálnej časti školského roku, či je ukončená predchádzajúca časť školského roku, teda, či existuje pre danú časť školského roku (v danom školskom roku) záznam v tabuľke finished v databáze. Ak táto ukončená nie je, nastaví ju ako aktuálnu namiesto tej, ktorá by mala byť aktuálna podľa aktuálneho dátumu. Až keď je predchádzajúca časť školského roku ukončená, začnú sa zobrazovať na stránkach Informačného systému údaje relevantné pre časť školského roku, ktorá je skutočne aktuálna. Ukončenie časti školského roku má na starosti administrátor (používateľská rola admin).

Informácie o tom, ktorú triedu žiak navštevuje nastavuje metóda setStudentsClass(). Tá nastaví hodnoty atribútom $studentsClassID, $studentsClassGrade, $studentsClassName a $studentsClass (atribút $studentsClass predstavuje textový reťazec zložený z $studentsClassGrade a $studentsClassName oddelených bodkou, napríklad „1.A“, „1.B“ a podobne). Keď už sú tieto informácie nastavené, je možné k nim pristupovať pomocou metód getStudentsClassID(), getStudentsClassGrade(), getStudentsGrade() (táto metóda je len alternatívnym pomenovaním metódy getStudentsClassGrade()), getStudentsClassName() a getStudentsClass(). Okrem toho, že každý žiak patrí do niektorej triedy, môže tiež patriť do nejakej skupiny, keďže na niektorých predmetoch je potrebné žiakov rozdeliť do skupín. Informácie o skupinách, do ktorých žiak patrí nastavuje metóda setStudentsGroups(). Tá nastaví atribúty $studentsGroupIDs a studentsGroupNames. Pre získanie ich hodnôt slúžia metódy getStudentsGroupIDs(), getStudentsGroupNames() a getStudentsGroupName($groupID).

Každá trieda má svojho triedneho učiteľa. Je preto vhodné uchovávať informácie o triede, pre ktorú je používateľ triednym učiteľom (ak je daný používateľ triednym učiteľom niektorej triedy). Nastavenie takýchto informácií v triede IS má na starosti metóda setTeachersClass(). Pre ich opätovné získanie slúžia metódy getTeachersClassID(), getTeachersClassGrade(), getTeachersClassName() a getTeachersClass().

6 Možnosti rozšírenia

Informačný systém pre základné a stredné školy nie je aplikácia určená pre jednu konkrétnu školu. Naopak, cieľom jej vzniku bolo vytvoriť aplikáciu, ktorá by bola flexibilná a použiteľná pre rôzne školy. Keďže rôzne školy môžu mať tiež rôzne dodatočné požiadavky na tento systém, je dôležité, aby bol jednoducho modifikovateľný, resp. rozšíriteľný. V nasledujúcom texte priblížim spôsob, akým som riešil zjednodušenie prípadných modifikácií, resp. rozšírení v troch hlavných oblastiach, u ktorých je potreba takýchto úprav najpravdepodobnejšia.

6.1 Jazyky

V súčasnosti existujú mnohé školy, ktoré poskytujú vyučovanie aj v inom než slovenskom jazyku. Taktiež prebiehajú rôzne výmenné pobyty študentov medzi školami na Slovensku a v zahraničí. Bolo by preto nerozumné neumožniť podporu viacerých jazykov v Informačnom systéme pre základné a stredné školy. Navyše tento systém nie je určený výhradne pre slovenské školy.

Pre realizáciu jazykovej podpory som najskôr vytvoril pod koreňovým adresárom Informačného systému adresár s menom „lang“. Tento adresár slúži na uskladňovanie „jazykových súborov“. Sú to súbory, ktoré obsahujú všetky texty použité na stránkach Informačného systému preložené do jednotlivých jazykov. Pre každú jazykovú verziu existuje práve jeden takýto súbor. Napríklad pre anglickú verziu je to súbor „en.php“, pre slovenskú verziu „sk.php“. Každý takýto súbor definuje tie isté premenné. Sú to asociatívne polia textových reťazcov, ktorých hodnoty sú rôzne slová, slovné spojenia, alebo celé vety. Konkrétne sa jedná o tri premenné. Premenná $err_msg uchováva chybové správy, premenná $info_msg informatívne správy a premenná $text všetky ostatné texty použité na stránkach Informačného systému. S týmito premennými manipulujú triedy MSGHandler a User. V triede User je definovaná metóda setLang($lang), ktorá zistí, či v adresári „lang“ existuje súbor s menom $lang a príponou „.php“ (hodnota $lang môže byť „en“, „sk“, ...). Ak taký súbor existuje, nastaví hodnotu atribútu $lang na hodnotu parametra $lang, inak nastaví hodnotu atribútu $lang na „en“. Odteraz, ak bude niektorá metóda triedy User potrebovať hodnotu premennej s textom, ktorý má byť zobrazený na

niektorej stránke Informačného systému, bude ju hľadať v súbore s menom $lang a príponou „.php“ v adresári „lang“. Keďže aj metódy triedy MSGHandler manipulujú s „jazykovými súbormi“, zavolá metóda setLang($lang) tiež metódu triedy MSGHandler s rovnakým menom a ako parameter jej pošle hodnotu atribútu $lang triedy User. Metóda setLang($lang) triedy MSGHandler len nastaví hodnotu atribútu $lang (v triede MSGHandler) na hodnotu parametra $lang.

Vyššie popísané metódy slúžia na nastavenie jazyka v objektoch typu User a MSGHandler. Pri prechode z jednej stránky Informačného systému na inú stránku tohto systému tieto objekty zanikajú a je potrebné ich znovu vytvoriť. To ale znamená, že je tiež potrebné znovu nastaviť jazyk používateľa. Aby bolo možné určiť, aký jazyk je treba používateľovi nastaviť, je potrebné túto informáciu pre konkrétneho používateľa niekde uchovávať. Tu však treba rozlišovať medzi používateľom, ktorý je prihlásený do systému a medzi neprihláseným (anonymným) používateľom. Prihlásený používateľ je taký používateľ, o ktorom sú uložené informácie v databáze, preto je možné k týmto informáciam pridať ďalšiu, ktorá bude predstavovať jazyk pre tohto používateľa. V databáze to predstavuje stĺpec „lang“ v tabuľke „user“. Ak je teda používateľ prihlásený (sú nastavené príslušné hodnoty premennej $_SESSION), nastaví sa pri vytváraní objektov typu User a MSGHandler jazyk v týchto objektoch (ich atribúty $lang) na hodnotu, ktorá je uložená v databáze pre daného používateľa. Čo ale v prípade, že používateľ nie je prihlásený? Akým spôsobom si má systém pamätať, ktorý jazyk tento používateľ používa? Keďže sa jedná o anonymného používateľa, nie je možné uchovávať informácie o ňom v databáze. Riešením je premenná $_COOKIE. Hodnotu tejto premennej uchováva internetový prehliadač na strane klienta ako textový súbor. Nastavenie jazyka neprihlásenému (anonymnému) používateľovi v tomto prípade teda znamená nastaviť jazyk pre internetový prehliadač, cez ktorý tento používateľ pristupuje k Informačnému systému.

Okrem nastavenia jazyka, ktoré sa vykoná vždy pri vytvorení objektu typu User (pri otvorení niektorej stránky Informačného systému), je veľmi dôležité, aby mal používateľ tiež možnosť si tento jazyk zmeniť. Na to slúži metóda triedy User changeLang($lang). Táto metóda najprv zistí, či je používateľ prihlásený, alebo nie. Ak áno, tak zmení hodnotu v stĺpci „lang“ v databáze pre tohto používateľa (v tabuľke user) na hodnotu parametra $lang a nastaví hodnotu premennej $_SESSION['lang'] (takisto na hodnotu parametra $lang). Ak používateľ nie je prihlásený, použije hodnotu parametra $lang na nastavenie premennej $_COOKIE['lang']. Metóda changeLang($lang) sa zavolá po odoslaní formulára pre zmenu jazyka používateľom. Tento formulár sa zobrazí používateľovi na stránke s nastaveniami („settings.php“). Jeho veľkou výhodou je, že je generovaný dynamicky. To znamená, že zoznam jazykov, ktoré ponúka používateľovi na výber v ňom nie je pevne stanovený, ale je generovany v závislosti od toho, aké „jazykové súbory“ sa nachádzajú v adresári „lang“. Vďaka tomu je vytváranie nových jazykových verzií Informačného systému naozaj veľmi jednoduché. Stačí len vytvoriť nový súbor s menom príslušného jazyka (napríklad „de“ pre nemčinu), vložiť do neho príslušné premenné ($err_msg, $info_msg a $text) s hodnotami preloženými do nového jazyka a tento súbor uložiť do adresára „lang“. Nie je potrebný žiadny zásah do kódu aplikácie. Formulár pre zmenu jazyka ponúkne automaticky spolu s ostatnými jazykmi aj tento nový jazyk používateľovi na výber.

Tu sa naskytuje niekoľko otázok. Prečo pri nastavovaní jazyka rozlišovať medzi prihláseným a neprihláseným používateľom? Prečo nevyužiť premennú $_COOKIE aj pre prihláseného používateľa. Neznamená takéto rozdelenie, že používateľ ak chce zmeniť jazyk, potrebuje to vykonať dva krát, najprv pred tým, ako sa prihlási a potom ešte raz, po prihlásení? Áno, je pravda, že používateľ potrebuje vykonať zmenu jazyka dva krát a takisto je pravda, že by sa to dalo vyriešiť využitím premennej $_COOKIE bez rozlišovania medzi prihláseným a neprihláseným používateľom. Je ale treba uvedomiť si niekoľko skutočností. Je vysoko pravdepodobné, že konkrétny používateľ bude potrebovať zmeniť jazyk Informačného systému len pri prvom prístupe, kedy je ešte nastavený štandardný jazyk, ak mu tento nevyhovuje. Takisto je pravdepodobné, že mnohí používatelia sa budú prihlasovať do tohto systému z rôznych počítačov, napríklad doma, v škole, v internetovej kaviarni a podobne. Ak by bolo nastavenie jazyka aj pre prihláseného používateľa riešené pomocou premennej $_COOKIE a nie pomocou údaju v databáze, znamenalo by to, že by nezáležalo na tom, či by používateľ zmenil jazyk ako prihlásený, alebo ako neprihlásený. Zmena by sa prejavila aj po odhlásení, resp. prihlásení. Čo je ale podstatnejšie, ak by sa používateľ potreboval prihlásiť do systému z iného počítača (presnejšie prehliadača), Informačný systém by použil jazyk, ktorý mohol nastaviť iný používateľ.

6.2 CSS

Ďalšou podporou rozšírenia Informačného systému je možnosť vytvorenia nových kaskádových štýlov (CSS), z ktorých si potom môže používateľ vyberať. Pre tento účel som vytvoril v koreňovom adresári aplikácie Informačného systému adresár „css“. Tento adresár slúži na ukladanie adresárov pre jednotlivé kaskádové štýly. Adresár pre konkrétny štýl musí obsahovať súbor „main.css“ a môže tiež obsahovať napríklad obrázky, ktoré používa. Nastavenie a zmena kaskádových štýlov je vykonávaná analogickým spôsobom ako nastavenie, či zmena používateľského jazyka.

Štandardný kaskádový štýl predstavuje adresár „default“. Cieľom pri vytváraní tohto štýlu bola snaha dosiahnuť jednoduchosť a prehľadnosť stránok Informačného systému. Tento štýl využíva len tri obrázky, z ktorých každý je veľkosti menej ako 1kB. Vďaka tomu je tento systém dostupný aj pre používateľov s pomalým, resp. obmedzeným pripojením k sieti Internet. Ak má škola záujem poskytnúť používateľom aj zložitejšie štýly na zobrazenie stránok Informačného systému, stačí, ak vytvorí v adresári „css“ adresár pre nový štýl a vloží do neho nový súbor „main.css“ a tiež obrázky (resp. adresáre s obrázkami) pre tento štýl. Informačný systém automaticky nájde nový štýl a ponúkne ho používateľovi na výber (na stránke s nastaveniami - „settings.php“). Používateľ sa potom môže sám rozhodnúť, ktorý štýl sa mu viac páči, resp., ktorý štýl je pre neho vhodný vzhľadom na jeho technické možnosti.

6.3 Roly

Informačný systém pre základné a stredné školy nie je určený len pre jeden typ používateľov. Naopak používatelia sú rozdelení na niekoľko typov (roly). Existujú tri základné roly v tomto systéme, a to administrátorska rola (admin), učiteľská rola (teacher) a žiacka rola (student). Každá rola má rôzne práva pre prístup k jednotlivým stránkam informačného systému. Ak by niektorá škola potrebovala vytvoriť novú rolu používateľov, je dôležité, aby nemusela táto škola príliš zasahovať do zdrojových kódov aplikácie. Vytvoril som preto v koreňovom adresári aplikácie Informačného systému adresár „roles“, ktorého podadresáre predstavujú jednotlivé roly. V týchto podadresároch sa nachádzajú časti stránok, ktorých obsah závisí od toho, do ktorej roly používateľ patrí (napríklad navigácia stránky a podobne). Ak chce škola pridať do Informačného systému novú používateľskú rolu, je potrebné vytvoriť adresár s jej menom v adresári „roles“ a vložiť do neho časti stránky, ktoré majú mať rôzny obsah pre rôzne roly, upravené pre túto rolu. Potom už stačí len povoliť prístup pre novú rolu na stránkach, pre ktoré je to žiadané.

7 Inštalácia

Pre inštaláciu Informačného systému pre základné a stredné školy je potrebné mať prístup na server s podporou PHP a MySQL. V databáze je potrebné vytvoriť tabuľky podľa modelu databázy. Je možné ich pomenovať aj inými menami, aké sú uvedené v tomto modeli, prípadne k ich menám pridať prefix a/alebo postfix. Mená stĺpcov v príslušných tabuľkách musia byť pomenované podľa modelu databázy. Ďalej je potrebné v súbore „dbsettings.php“ nastaviť hodnoty jednotlivým premenným: $db['SERVER'] – meno databázového servera, $db['NAME'] – meno databázy, $db['USER'] – meno používateľa, ktorý má prístup k databáze, $db['PASSWORD'] – heslo pre tohto používateľa, $tablePrefix – prefix pre mená tabuliek v databáze, $tablePostfix – postfix pre mená tabuliek v databáze a tiež mená jednotlivých tabuliek v databáze: $table['user'], $table['class'], $table['group'] atď. Ak nie je potrebné použiť pre tabuľky v databáze iné mená, ako sú definované v modeli databázy, ani im pridať prefix, alebo postfix, je možné pre vytvorenie tabuliek v databáze použiť dotaz uložený v súbore „install.sql“. Pre tento prípad stačí v súbore „dbsettings.php“ nastaviť meno databázového servera, meno databázy, meno používateľa a heslo pre tohto používateľa. V súbore „schoolsettings.php“ je možné upraviť meno a počet častí školského roku a tiež dátum (deň a mesiac) začiatku a konca jednotlivých častí.

8 Záver

Informačný systém pre základné a stredné školy môže nepochybne uľachčiť prácu mnohým učiteľom rôznych škôl. Zároveň pomôže žiakom a tiež ich rodičom nestratiť prehľad o známkach z jednotlivých predmetov, či informáciách o dianí na týchto predmetoch, alebo na škole celkovo. Jazyková podpora umožní jednoduchú implementáciu tohto systému na školách rôznych krajín, pričom školy v týchto krajinách môžu mať školský rok rozdelený na rôzny počet častí, resp. tieto časti môžu mať rôzny dátum začiatku, či ukončenia.

Zároveň vysoká podpora prístupnosti umožní pracovať s týmto systémom nie len na osobných počítačoch, ale tiež na rôznych iných zariadeniach schopných pripojiť sa k Internetu a zobrazovaať dokumenty typu XHTML, akými sú napríklad mobilné telefóny, alebo rôzne vreckové počítače (PDA) a podobne. Žiak takto získa možnosť informovať sa o svojich známkach a o dianí v škole kdekoľvek sa nachádza, či je to doma, u priateľov, v električke, v zahraničí a podobne..

Navyše tento systém bude možné stiahnuť z internetu a tiež ho ďalej upravovať pre potreby rôznych škôl, ale tiež iných organizácií, či jednotlivcov. Takisto môže slúžiť ako inšpirácia pre tvorbu podobných systémov v rôznych oblastiach.

Literatúra

[1] PHP manual: http://www.php.net/manual/en/
[2] MySQL 5.1 manual: http://dev.mysql.com/doc/refman/5.1/en/index.html
[3] Hugh E. Williams, David Lane: *PHP a MySQL Vytváříme webové databázové aplikace,* 2002
[4] W3C (World Wide Web Consortium): http://www.w3.org/
[5] W3Schools Online Web Tutorials: http://www.w3schools.com/
[6] UTF-8 and Unicode Standards: http://www.utf-8.com/
[7] Wikipedia, the free encyclopedia: http://www.wikipedia.org/