VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

EDITOR RELAČNÍCH TABULEK

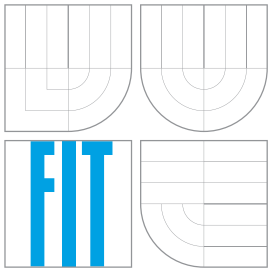
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN BOHÁČ

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

EDITOR RELAČNÍCH TABULEK

EDITOR OF RELATION TABLES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN BOHÁČ
AUTHOR

VEDOUCÍ PRÁCE — Mgr. MARTIN ŠÁRFY
SUPERVISOR

BRNO 2008

Abstrakt

Práce pojednává o tvorbě editoru umožňující efektivní prohlížení a úpravu dat databáze. Zmiňuje použité technologie a možné alternativy dostupné na trhu. Dále se věnuje popisu implementace a způsobu řešení implementačních problémů. Závěr práce se věnuje možnostmi rozšíření systému a jeho možné uplatnění v různých oblastech.

Klíčová slova

Editor relačních tabulek, PHP, MySQL, XHTML, CSS, XML, Zend Framework, MVC, relační databáze, metadata

Abstract

The thesis describes development of editor, which allow users to effective browse and manage data of database. It advert used technoligies and possible alternatives accesible on market. Next part attend to describe of implementation and methods of solving implement problems. The rest of this work is set to posibilites of extension and using in different departments

Keywords

Editor of Relation Tables, PHP, MySQL, XHTML, CSS, XML, Zend Framework, MVC, relation database, metadata

Citace

Martin Boháč: Editor relačních tabulek, bakalářská práce, Brno, FIT VUT v Brně, 2008

Editor relačních tabulek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Mgr. Martina Šárfyho.

.....
Martin Boháč
13. května 2008

Poděkování

Velmi rád bych na tomto místě poděkoval Mgr. Martinovi Šárfymu za poskytnutou pomoc a konzultace při tvorbě této práce.

Obsah

Kapitola 1

Úvod

Bakalářská práce popisuje vývoj editoru pro správu relačních tabulek, který umožňuje efektivní prohlížení a úpravu dat databáze. Nejprve bude pozornost věnována motivaci a proč právě Editor relačních tabulek. Dále pak průzkum současného stavu na trhu a srovnání s podobnými systémy.

V další kapitole si zmíníme přesnou specifikaci zadání, co bude editor umět. Provedeme rozbor návrhu řešení a srovnání různých technologií, které mohou být vhodné pro implementaci programu.

Poté bude následovat stručný popis zvolených technologií a nástrojů. Dále se budeme věnovat implementaci a způsobu řešení jednotlivých problémů.

V závěru bude zhodnocen přínos editoru a zamyšlení nad možností budoucího rozšíření.

Kapitola 2

Motivace

2.1 Aktuální stav

2.1.1 Informační systémy - šité na míru

Informační systém je aplikace, která umožňuje spravovat, třídit a analyzovat různé informace menších či větších projetků, pro jednotlivce či celé firmy. Bez kvalitního informačního systému dnes nemůže větší firma nebo projekt efektivně existovat.

Informační systém Vám může umožnit příjem a správu objednávek, komunikace zaměstnanců, řízení práv i analýzu potřebných dat.

Funkce a možnosti informačního systému závisí především na požadavcích klientů (uživatelů), kteří budou systém využívat. Pomocí informačního systému se mohou řídit interní záležitosti firmy, spravovat webové stránky nebo obchod. Důležitou funkcí může být i napojení na účetní systém, dodavatele nebo jiné potřebné služby.

Informační systém je aplikace, popřípadě skupina aplikací, které navzájem spolupracují. Jinak bude navržen informační systém banky a jinak informační systém pro dopravní společnost. Dá se říci, že IS bude mít vždy alespoň dvě vrstvy. První vrstva bude datová založená na databázovém serveru a druhá vrstva aplikační, která zajišťuje funkce mezi uživateli a datovou vrstvou.

Používané technologie při tvorbě informačních systémů závisí na konkrétních požadavcích na konkrétní informační systém. Pro jednodušší projekty využívající internetového prohlížeče může být vhodné spojení databáze MySQL a skriptovacího jazyka PHP. Pro složitější projekty se může využít programovací jazyk C# s napojením na vhodnou databázi(MSSQL, PostgreSQL).

Cena informačního systému závisí na ceně potřebného softwaru (někdy i hardwaru) a ceně za práci programátora nebo týmu. V případě menších projektů v PHP a MySQL je cena softwaru nulová a je potřeba počítat jen s cenou za programátorské práce. V případě použití komerčního softwaru cena projektu roste nahoru.(viz [1])

2.1.2 Excel

Jedná se o tabulkový procesor z kancelářského balíku Microsoft Office. Excel poskytuje velmi mnoho funkcí a umožňuje tak dobrou práci s tabulky. Nepodporuje vytváření relací mezi nimi . Bohužel se nejedná o freeware, ale o placený software. Navíc je vyvíjen jen pro platformy Windows a Mac.

Existují neplacené konkurenční alternativy, jako jsou například produkty Calc od OpenOffice.org nebo český tabulkový procesor Calc z kancelářského balíku 602Office. Tyto neplacené alternativy mají navíc podporu nejen pro systémy Windows, ale i pro Linuxové systémy.

Společná nevýhoda těchto produktů je, že to nejsou síťové aplikace. Práce více lidí nad jedním dokumentem je velice komplikovaná a nesnadná.

2.1.3 phpMyAdmin

Nástroj pro vzdálenou správu MySQL databází přes webové rozhraní. Umožňuje například tvorbu datových polí a tabulek, nastavení práva atd. Vlastnosti phpMyAdmin:

- vytváření a rušení databáze
- vytváření, kopírování, přesouvání, odstranění a změny tabulky
- přidávání, změny a odstranění sloupce v tabulkách
- spouštění jakýchkoli SQL příkazů, včetně dávkového zpracování
- spravování klíčů a indexů tabulek
- nahrávání textových souborů do tabulek
- vytváření exportů tabulek ve formátu MySQL do textových souborů
- import/export ve formátu CSV - administrace více serverů - uživatelé, práva, procesy ad.

Výhodou je, že PhpMyAdmin je dostupný přes webové rozhraní a stačí mít nainstalovaný jen internetový prohlížeč. (viz [4])

2.1.4 Google spreadsheet

Výhody Google Spreadsheetu:

- umožňuje snadnou práci více lidí nad jedním dokumentem,
- umožňuje přístup k tabulkovému procesoru odkudkoli
- pohodlná práce v tabulkovém procesoru v internetovém prohlížeči, kde je možné využívat e-mail, kalendář a další nástroje, takže není nutné přecházet z e-mailu do jiného okna, čekat několik sekund na otevření aplikace a pak pracovat ve zcela jiném prostředí, které spotřebovává další paměťové prostředky,
- aplikace je zdarma
- silné postavení Googlu a předpoklad rychlého rozšíření

Nevýhody tohoto produktu jsou:

- je funkčně velmi omezený
- uživatel musí být online, aby jej mohl používat

- data jsou uložena vzdáleně, mimo disk uživatele
- může docházet k výpadkům
- nutnost zapnuté podpory JavaScriptů

(viz [3, 2])

2.2 Proč Editor relačních tabulek?

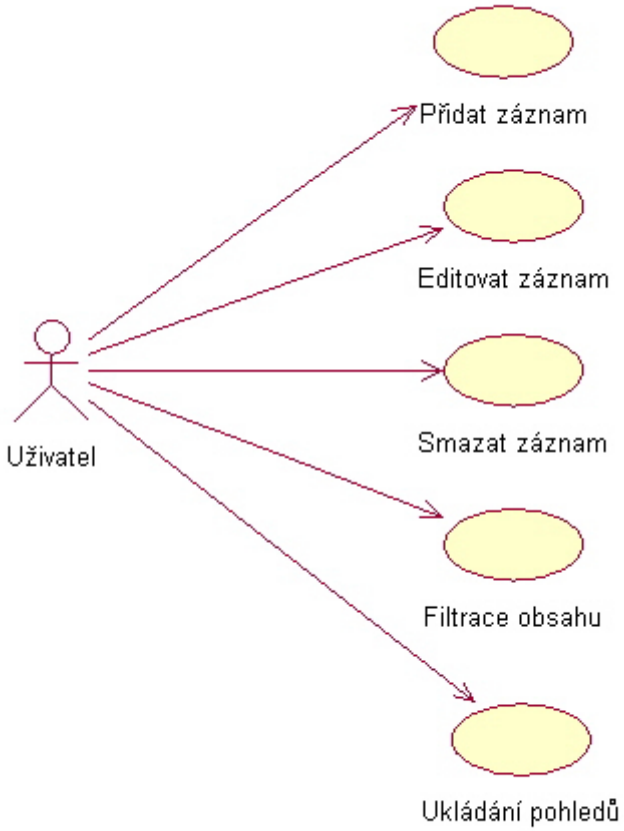
Základem všech těchto tabulkových procesorů, informačních systémů jsou jednoduché tabulky, relace mezi nimi a operace nad nimi. Podle různých potřeb je důležité správně zvolit mezi nabízenými produkty a možnostmi s ohledem na to, jakou funkci by aplikace měla splňovat a pro jaké účely bude využívána.

Přesto nejen podle mého názoru existuje určitá skupina uživatelů, které momentálně dostupné aplikace nemusí vyhovovat. Ať už ve smyslu finanční či časové náročnosti na vývoj u informačních systémů, nepodporování správy dokumentů po síti (více uživateli) u tabulkových procesorů typu Excel, nebo nutnosti provozování na cizím serveru u Google spreadsheet. PhpMyAdmin je sice mocný nástroj pro správu databáze MySQL, ale jeho nasazení například pro správu účetnictví ve firmě nebo pro správu dat v tabulkách více lidmi je silně nevhodné.

Domnívám se, že se najde mnoho uživatelů, ať už studenti, učitelé, menší firmy, kteří pro řešení vlastních projektů potřebují síťovou aplikaci pracující s tabulkami a umožňující základní operace nad nimi. Aplikaci, za kterou by nemuseli platit tisíce či desetitisíce. Jednalo by se o webovou aplikaci, která by na základě definovaných tabulek vytvořila veškerou správu automaticky.

2.3 Use Case diagram

Model případu užití, na obrázku 2.1 nám objasňuje, jaké aktivity mohou uživatelé v editoru provádět.



Obrázek 2.1: Model případu užití

2.3.1 Popis diagramu

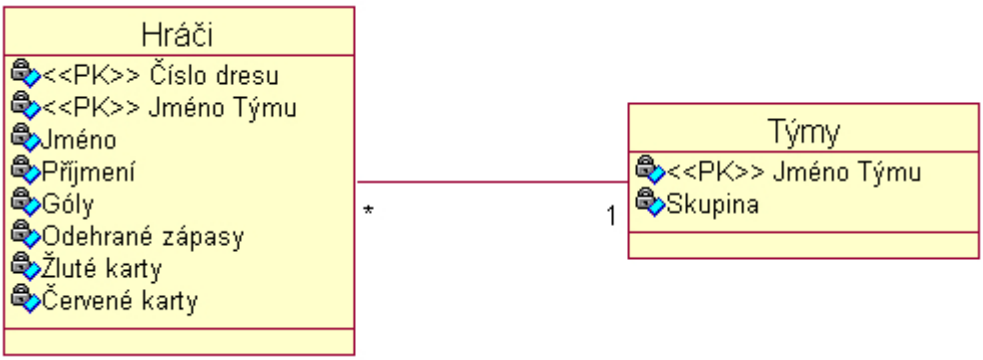
Use Case diagram počítá pouze s rolí uživatele. Je to dáno tím, že veškerá tvorba tabulek a operace nad nimi se provádějí automaticky na základě definovaného metadatového souboru ve formátu xml. Editor předpokládá definování tohoto soboru a umístění do operační složky předem, pověřenou osobou například administrátorem.

Uživatel může provádět všechny standardní operace, jako je přidávání, editace a mazání záznamů. Je možno filtrace dat a zobrazování jen zvolených sloupců. Následně je vytvořený pohled možno uložit a příště již rovnou pracovat s tímto pohledem.

2.4 ER diagram

Vzhledem k obecnosti editoru, který vytváří správu nad tabulkami na základě definovaných metadat v konfiguračním souboru, od kterých se ER diagram odvíjí, uvádím pouze příklady různých variant vztahů, které je možno definovat a které by editor měl zvládat.

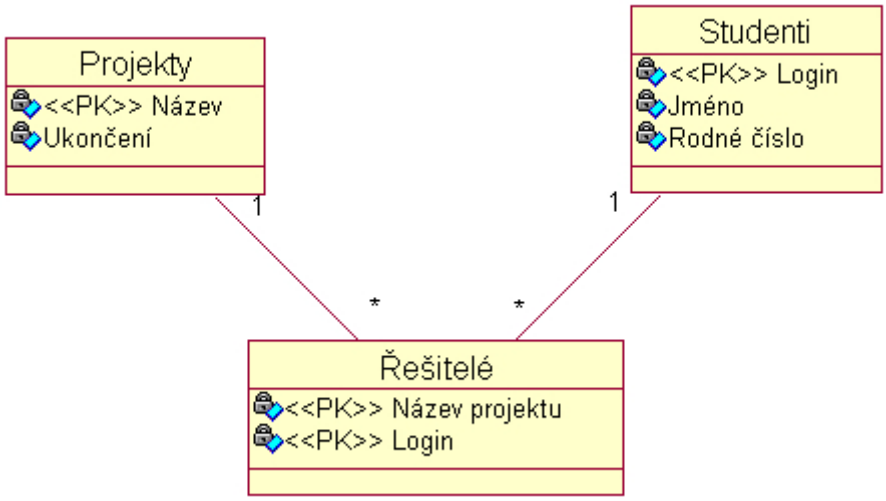
2.4.1 Vztah 1:N



Obrázek 2.2: Ukázka rozložení vztahu 1:N

Jedná se o vztah mezi tabulkami, kdy jeden záznam jedné tabulky (např. Tým), odkazuje na několik záznamů tabulky druhé (např. Hráči z daného týmu) (viz obrázek 2.2).

2.4.2 Vztah M:N



Obrázek 2.3: Ukázka rozložení vztahu M:N

Tento vztah popisuje situaci, kdy každý záznam z obou tabulek je propojen s několika

záznamy tabulky druhé. V takovémto případě je nutno vytvořit pomocnou tabulku, se kterou mají obě tabulky vztah 1:N (viz 2.4.1). V této tabulce se nachází všechny existující vztahy mezi těmito tabulkami (viz obrázek 2.3).

Kapitola 3

Návrh

3.1 Specifikace zadání

Mám navrhnout systém, který na základě schématu relační databáze (metadat) umožní efektivní prohlížení a úpravu dat databáze.

Prohlížení by mělo obsahovat následující funkce :

- formátování výstupu: číslo, datum, text, zobrazení hodnoty z jiné tabulky přes cizí klíč
- filtrování: relační testy i nad více sloupci
- třídění: dle typu sloupce (číselné, abecední, časové), vzestupné, sestupné, stránkování výstupu

Úprava dat spočívá v možnosti přidat, změnit nebo smazat řádek, s možností zadávání hodnot dle typu sloupce: text, číslo, pravdivostní hodnota, možnost statického výčtu hodnot sloupce nebo výčtu hodnot přes cizí klíč.

Volitelným rozšířením jsou základní agregační funkce nad sloupci (výčet hodnot sloupce, součet, průměr,. . .)

3.2 Požadavky na metadata

Metadata(mezidata) jsou strukturovaná data o datech, která popisují obsah, kvalitu, stav a jiné charakteristiky dat. Metadata umožňují interpretovat jiná (primární) data. Komplexnost popisu metadaty se může lišit podle konkrétních potřeb. V projektu by metadata měla být využita pro popis tabulek, vztahů mezi nimi a způsob chování v samotné aplikaci. Mělo by se tedy jednat o metadata popisující způsobem, jako jsou například data based XML dokumenty (viz [11]) popsané v jazyce XML. Takové to dokumenty plní funkci jakési obálky pro přenos či manipulaci s daty, kde hlavním cílem je uvvydolovat] data a zcela jednoznačně pochopit relace mezi nimi. Jedním z typických příkladů tohoto pojetí je bezesporu využívání XML při přenosu tabulek bez ztráty informace o vzájemných relacích.

Původně jsem se také zamýšlel nad použitím XML schéma(viz [10]) k definici metadat XML. XML schéma je také XML dokument, ale se speciálními prvky určující jaké prvky atributy můžeme v XML používat. Nakonec jsem usoudil, že volnost, kterou samotné XML umožňuje , je v případě této aplikace vítaná.

3.3 Vhodné technologie

3.3.1 PHP

PHP (Personal Home Page Tools, později Personal Home Page Construction Kit nakonec Hypertext Preprocessor) je serverový skriptovací programovací jazyk navržený pro programování dynamických webových stránek. Vše se provádí na straně serveru, kde se jako výstup generuje HTML(XHTML aj.). PHP je multiplatformní a Open Source, tedy volně šiřitelná technologie. PHP je programovací jazyk umožňující procedurální i objektově orientované programování. Je to dynamicky typový jazyk. Syntaxe jazyka kombinuje hned několik programovacích jazyků (Perl, C, Pascal a Java) a nechává tak programátorům určitou volnost v syntaxi. Díky tomu a své jednoduchosti použití je PHP velmi oblíbeným programovacím jazykem u vývojářů.

3.3.2 PHP frameworky a MVC

PHP frameworky

Díky velkému rozšíření PHP se rozmohli i PHP frameworky. Frameworky se snaží poskytnout programátorovi efektivnější způsob řešení svých projektů a aby nemuseli psát lehce obměněný kód stále dokola. Pokud se porozhlédneme i jiné programovací jazyky mají své frameworky, například u Ruby je to již velmi známý a rozšířený framework Ruby on Rails. Já se ale budu v této části věnovat frameworkům PHP, jelikož jsem se rozhodl řešit projekt v tomto jazyce.

Podle Wikipedie [8] je Framework softwarová struktura sloužící jako podpora při programování, vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovnu API, návrhové vzory nebo doporučené postupy při vývoji.

Obecně můžeme rozlišit frameworky buď na sady skriptů a knihoven pokrývající všemožné potřeby (např. Zend Framework, PRADO) nebo na skripty vytvářející jednu konkrétní webovou aplikaci (např. CakePHP, CodeIgniter a opět Zend Framework). Principem druhé skupiny frameworků je aplikování určitého návrhového vzoru nejčastěji MVC (Model-View-Controller), který se používá pro oddělení rozdílných částí aplikace a kde všechny požadavky míří na jeden bootstrap soubor index.php, jenž řídí celou aplikaci.

Zend framework

Zend Framework je nový PHP MVC framework přímo od společnosti Zend Technologies Ltd., která stojí za jádrem PHP od verze 4. Zend Framework je soubor knihoven, které mají výrazně usnadnit práci všem tvůrcům PHP aplikací. Důvody vzniku Zend Frameworku byl vytvořit standard v tvorbě pokročilejších PHP aplikací s využitím objektově orientovaného programování (OOP) při jejich tvorbě. Všechny komponenty jsou napsány v PHP5. Od toho se odvíjí vyžadování minimální verze PHP 5.1.4. Jedná se o open source framework.

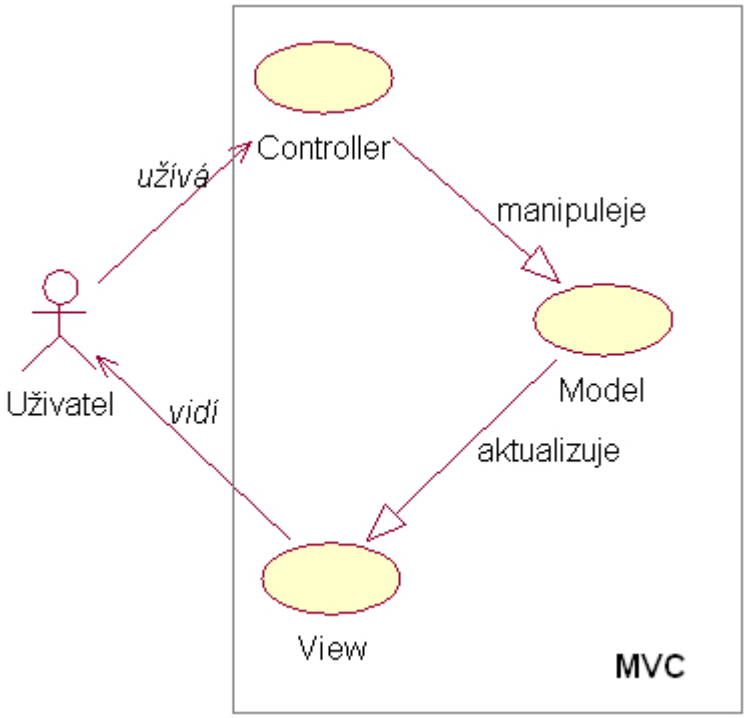
Mezi další výhody ZF patří například bohatá dokumentace, nebo že není invazivní a tudíž lze používat jen jednotlivé části frameworku. Také výhodu jsou existující vývojová prostředí, například Zend Studio nebo Zend Studio for Eclipse.

MVC (Model-View-Controller)

MVC neboli Model-View-Controller je softwarová architektura, oddělující data, jejich zobrazení a řídící logiku do tří nezávislých komponent. Model je datové úložiště, s nímž aplikace

pracuje. Controller (řadič) komunikuje s modelem i pohledem, které řídí na základě události způsobené uživatelem. View (pohled) zobrazuje data modelu uživateli. (viz [9])

Ukázka 3.1 komunikace jednotlivých částí architektury Model-View-Controller.



Obrázek 3.1: Model-View-Controller

3.3.3 XHTML

Roku 1995, bylo vydáno HTML 2.0 (HyperText Markup Language - hypertextový značkovací jazyk), internetovou standardizační organizací jako RFC. Tento jazyk byl jakousi podmnožinou univerzálního značkovacího jazyka SGML. Od té doby prošel jazyk určitým vývojem přes verzi HTML 3.2, u které byly začleněny mnohé prvky pro formátování vzhledu, až k současné specifikaci HTML 4.01, která obsahuje kompletní popis jazyka a všechny následující specifikace jazyků pro tvorbu webu, které doposud vyšly, se na ni odkazují.

Na základě potřeby definovat HTML jazyka jako podmnožinu jazyka XML (eXtensible Markup Language - rozšiřitelný značkovací jazyk), které W3C (World Wide Web Consortium) prosazuje jako hlavní a jediný značkovací jazyk nejen pro web, vznikl nově vytvořený jazyk s názvem XHTML (eXtensible HyperText Markup Language - rozšiřitelný hypertextový značkovací jazyk). Specifikace je stejná jako specifikace HTML 4.01, ale na rozdíl od něj jsou zde pouze integrována pravidla XML. Tato specifikace rozděluje všechny prvky XHTML 1.0 do modulů, které je možno modifikovat a skládat nové kompletní značkovací jazyky. World Wide Web Consortium zatím uznalo XHTML 1.1 (definováno pomocí modulů) a XHTML Basic (pouze základní moduly, k použití hlavně na mobilních telefonech, PDA a podobně) (viz [6]).

V projektu je použita verze XHTML 1.0 Transitional se kterou pracuje výše zmíněný

Zend framework.

3.3.4 CSS

Historie podle [7] CSS (Cascading Sytle Sheets) neboli kaskádové styly vznikly jako souhrn metod pro úpravu vzhledu stránek. První návrh normy byl zveřejněn v roce 1994, v roce 1996 byla pak vydána specifikace CSS 1, v roce 1998 CSS 2. Nyní se pracuje na verzi CSS 3.

Díky CSS můžeme formátovat obsah HTML, XHTML a XML dokumentů. Styly umožňují přesně určit, jak bude který element vypadat. Jediným zápisem pro příslušný element můžeme definovat jednotný vzhled elementu pro celý dokument, nebo naopak můžeme určit odlišné formátování i pro jediný výskyt určitého elementu. Tato možnost zkracuje a zpřehledňuje kód. Umožňuje snadné úpravy a změny ve vzhledu na jednom místě, než několika změnám přímo úměrným počtu výskytu upravovaného elementu.

Kaskádové styly jsou mocným nástrojem pro úpravu vzhledu webových aplikací a díky podpoře Zend Frameworkem bude tato technologie v projektu využita.

3.3.5 MySQL

MySQL je systém, od švédské společnosti TcX,pro správu relačních databází (viz[5]). Tento systém je založen na jazyku SQL, standardu relačních databázových systémů. Jazyk SQL využívá k manipulaci s daty, případně jejich zobrazení či vytvoření. Kromě výhody, že se jedná o freeware a má otevřený kód (součástí Open Source Movement), je další nespornou výhodou jeho nezávislost na operačním systému.

Díky spouštění ve více vláknech je všem klientům serveru MySQL přiřazen vlastní proces. Je tím dosaženo velké rychlosti serveru. MySQL splňuje i vysoké nároky kladené na zabezpečení dat před zneužitím nepovolanými osobami. Lze jasně stanovit, jaká data jsou pro kterého uživatele. Také ze strany hostingových služeb je MySQL v drtivé většině podporován.

Zejména z důvodu snadné dostupnosti a podpory ze strany PHP a Zend frameworku, je pro projekt využit právě tento databázový systém.

Kapitola 4

Implementace

4.1 Formát XML

Základem editoru je soubor s metadaty, který definuje schéma relační databáze. Pro tento účel byl zvolen značkovací jazyk XML i pro jeho dobrou podporu PHP ve formě nástroje `simpleXML`. Název souboru, ve kterém editor předpokládá definované tabulky má název config_tables.xml. Editor umožňuje tvorbu pohledů nad tabulkami. Tyto pohledy se ukládají taktéž do xml souboru s názvem views.xml.

4.1.1 Formát definice tabulek

```
<?xml version="1.0"?>
<dbmodel>
<tabulka>
    <numberINT type="number" primarykey="auto_increment"></numberINT>
    <numberFLOAT type="number" format="%4.2f"></numberFLOAT>
    <stringPK type="string" primarykey="true"></stringPK>
    <string type="string" ></string>
    <stringTEXTAREA type="string" form="textarea"></stringTEXTAREA>
    <enum type="enum">'A','B','C','D'</enum>
    <datum type="datetime" format="dd.month yyyy"></datum>
    <cas type="datetime" format="hh:mm:ss"></cas>
  </tabulka>
<ukazka_AgFunc>
    <sloupec1 type="number" primarykey="auto_increment" AgFunc="suma">
    </sloupec1>
    <sloupec2 type="number" AgFunc="prumer"></sloupec2>
    <sloupec3 type="number" AgFunc="min"></sloupec3>
    <sloupec4 type="number" AgFunc="max"></sloupec4>
  </ukazka_AgFunc>
</dbmodel>
```

4.1.2 Formát automaticky generovaných pohledů

```
<?xml version="1.0"?>
<dbmodel>
```

```
 <ukazka_AgFunc view="1">
    <sloupec1 name="sloupec1" type="number" primarykey="auto_increment"
              AgFunc="suma" visibility="1" filter_text="" filter_comp="all"/>
    <sloupec2 name="sloupec2" type="number" AgFunc="prumer" visibility="1"
              filter_text="30" filter_comp="more"/>
    <sloupec3 name="sloupec3" type="number" AgFunc="min" visibility="1"
              filter_text="" filter_comp="all"/>
    <sloupec4 name="sloupec4" type="number" AgFunc="max" visibility="0"
              filter_text="" filter_comp=""/>
  </ukazka_AgFunc>
  <datove_typy view="1">
    <numberINT name="numberINT" type="number" primarykey="auto_increment"
              visibility="1" filter_text="" filter_comp="all"/>
    <numberFLOAT name="numberFLOAT" type="number" format="%4.2f" visibility="1"
              filter_text="" filter_comp="all"/>
    <stringPK name="stringPK" type="string" primarykey="true" visibility="1"
              filter_text="" filter_comp="all"/>
    <string name="string" type="string" visibility="1" filter_text=""
              filter_comp="all"/>
    <stringTEXTAREA name="stringTEXTAREA" type="string" form="textarea"
              visibility="0" filter_text="" filter_comp=""/>
    <enum name="enum" type="enum" enumvalues="Array" visibility="1"
              filter_text="B" filter_comp="eq"/>
    <datum name="datum" type="datetime" format="dd.month yyyy" visibility="1"
              filter_text="" filter_comp="all"/>
    <cas name="cas" type="datetime" format="hh:mm:ss" visibility="1"
              filter_text="" filter_comp="all"/>
  </datove_typy>
</dbmodel>
```

4.1.3 Datové typy a možné atributy

Pro zjednodušení definování tabulek existují čtyři základní datové typy (string, number, datetime, enum), které je možno definovat. Tyto typy je pak dále možno specifikovat atributy sloupce, na základě kterých je následně rozhodnuto v jakém datovém typu budou uloženy v databázi.

jmeno_tabulky

- label=”Jméno tabulky” - díky tomuto atributu je možné nastavit alternativní jméno tabulky, pod kterým bude přístupna v levém horním menu editoru
- jump=”30” - umožňuje nastavit maximální počet zobrazených záznamů (řádků) - na základě tohoto atributu se generuje menu nad a pod tabulkou, odkazující na bloky záznamů o velikosti této hodnoty

společné atributy

- type=”string|number|datatime|enum”

- primarykey=”true” - definuje, že sloupec bude primárním klíčem tabulky, čili v databázi nemůže být použit typ text, nýbrž je použit varchar(64).
- foreignkey=”tabulka.sloupec” - odkazuje na primární klíč cizí tabulky
- show=”tabulka.sloupec” - lze si zvolit který sloupec z cizí tabulky bude zobrazen
- visibility=”1|0” - určuje zda bude sloupec zobrazen

string

- form=”textarea” - tímto atributem lze definovat typ elementu ve formuláři, defaultně je element typu text

number

- min= určí minimální hodnotu, kterou je možno zadat
- max= určí maximální hodnotu, kterou je možno zadat
- format=”%4.2f|%d” - tento atribut ovlivňuje, zda se číslo bude ukládat do databáze jako typ int, defaultně int(11), nebo typ float
- AgFunc=”suma|prumer|min|max” - zvolí agregační funkci nad sloupcem

datetime

- format=”yyyy-mm-dd|yyyy-mm-dd|yyyy|dd.mm.yyyy|dd.mm.yy|day|day|dd.month yyyy| dd.month yyyy|dd.mm Fr|dd.mm |worddate” `worddate` - určí zda se bude jednat o slovní vyjádření data (dneska| zítra| včera| před 2 dny| za 2 dny)
- format=”hh:mm:ss|hh:mm|wordtime” `wordtime` - určí zda se bude jednat o slovní vyjádření času (Před 4 minutami| Za 50 minut| Za 37 sekund)

enum Je výčtový typ neobsahující, žádné další speciální atributy. Na rozdíl od ostatních datových typů, je potřeba zadat jeho hodnoty, kterých může nabývat. Tyto hodnoty se zapisují do hodnoty definovaného sloupce. Jednotlivé hodnoty jsou v jednoduchých uvozovkách a odděleny čárkou.

Příklad:

```
<nazevSloupce type="enum">'A','B','C','D','E','F','G','H'</nazevSloupce>
```

4.2 Editor - využití Zend Framework

Vzhledem k nevýhodě PHP, že se jeho kód prolíná s kódem HTML a kvůli všeobecným výhodám frameworků, jsem se rozhodl řešit projekt pomocí některého z nich. Z dostupných frameworků jsem si nakonec vybral Zend Framework. Zejména díky velké komunitě, která ho vyvíjí, se jedná o velmi mocný nástroj, který podle mého názoru bude v budoucnu mnohem více využíván.

Jak již bylo řečeno Zend framework dodržuje MVC architekturu, proto při popisu editoru budu postupovat podle těchto částí.

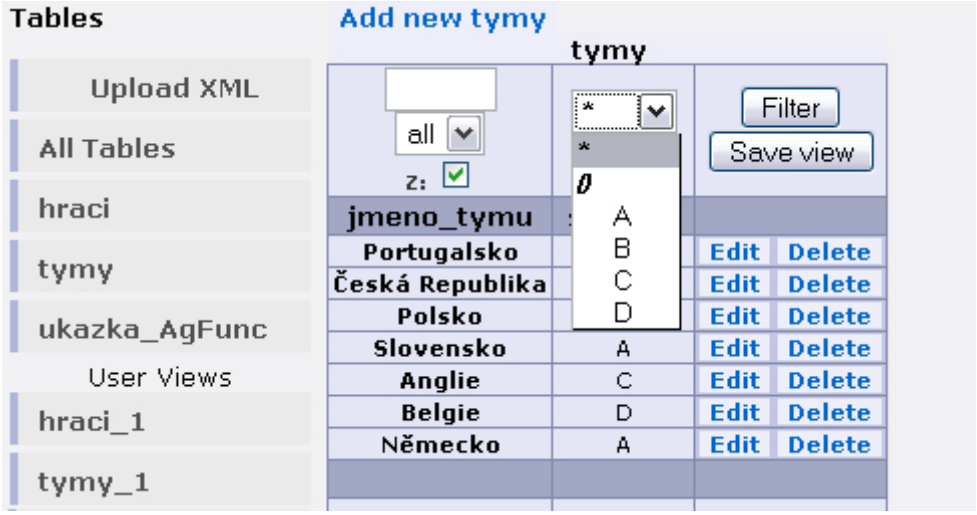
4.2.1 Instalace

Z důvodu použití Zend frameworku při řešení editoru, jsou kladeny určité minimální nároky, které musí webový server splňovat. Je požadována minimální verze PHP 5.1.4 nebo vyšší a webový server musí mít zapnutou podporu mod_rewrite. Dále je nutno mít nainstalovaný databázový systém MySQL 5.0 nebo vyšší.

4.2.2 Adresářová struktůra

```
Editor/
   /app
     /controllers
       IndexController.php - zde je třída controller, obsahující action public
                             funkce, na které je pak možno se odkazovat přes url
     /forms
       FilterForm.php - formulář pro filtrování obsahu tabulek a ukládání pohledů
       AllForm.php - formulář generovaný pro jakoukoli tabulku, umožňující
                     editaci, přidání a mazání záznamů
     /models
       xml.php - zde je definována třída, mající všechny metody pro obsluhu
                 definovanych tabulek ve formátu xml
       viewXML.php - třída pro správu užiatelsky definovaných pohledů a jejich
                     ukládání do souboru formátu xml
       MyTable.php - definována obecná třída pro načtení libovolné tabulky
                     z˜databáze
     /views
       /filters
       /helpers
       /layouts
         layout.phtml
       /scripts
         /index - šablony pohledu, pojmenované podle přidružené akce
           index.phtml
           element.phtml
           edit.phtml
           delete.phtml
           add.phtml
     config.ini - zde jsou nastaveny všechny důležité údaje pro přihlášení
                  do databáze
   /lib
     /Zend
   /www
     /images
     /scripts
     /styles
       main.css - kaskádový styl starající se o˜vzhled prostředí (editoru)
     index.php - hlavní soubor, nazývaný bootstrap soubor, na který jsou
                 přesměrovány všechny požadavky pomocí souboru .htaccess
     views.xml - do tohoto xml souboru editor ukládá, uživatelsky definované
```

```
            pohledy tabulek
config_tables.xml - v˜tomto xml souboru jsou definované tabulky nad kterými
                    má editor pracovat
.htaccess - obsahuje přepisovací pravidla, umožňující jakoukoli url
            přesměrovat na bootstrap soubor, čili index.php
```



Obrázek 4.1: Ukázka levého menu a filtrace nad sloupcem typu enum

4.2.3 Stručný popis vygenerované tabulky

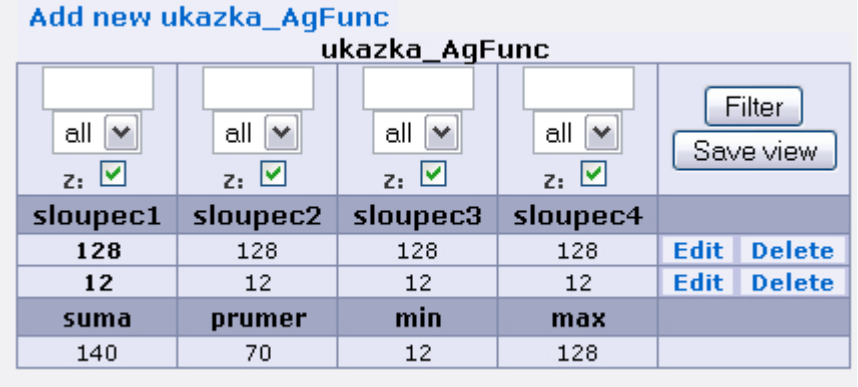
Na obrázku 4.1 vidíme ukázku jednoduché tabulky vygenerované na základě metadat definovaných v config_tables.xml. Jedná se o tabulku mající dva sloupce, kde primárním klíčem je jméno týmu. Editor při generování tabulek rozpozná sloupce, které jsou primárním klíčem a tyto sloupce zvýrazní tučnou barvou. U každého řádku umožní jeho editaci a smazání. V horní části tabulky je odkaz pro přidání nového záznamu.

Dále se v horní části nad každým sloupcem nachází formulářové elementy umožňující filtraci dat (viz sekce**??**), kde jsou jednotlivé elementy popsány blíže.

Krom filtrace formulář umožňuje vytvořený pohled nad tabulkou uložit a následně se na pohled odkazovat přes levé dolní menu (viz obrázek 4.1). Tvorba pohledů je detailněji popsána v sekci Pohledy4.2.11.

4.2.4 Agregační funkce

Editor nám umožňuje definovat nad číselnými sloupci agregační funkce (suma, průměr, minimální hodnotu, maximální hodnotu). Jejich užití vidíme na obrázku 4.2 , kde výsledky těchto funkcí jsou zobrazeny vždy pod daným sloupcem. Výhodou konfigurace agregační funkce v xml souboru je možnost pohodlné změny zvolené funkce v jinou.



Obrázek 4.2: Ukázka různých agregačních funkcí, které lze aplikovat nad sloupci.

4.2.5 Práce s tabulkami

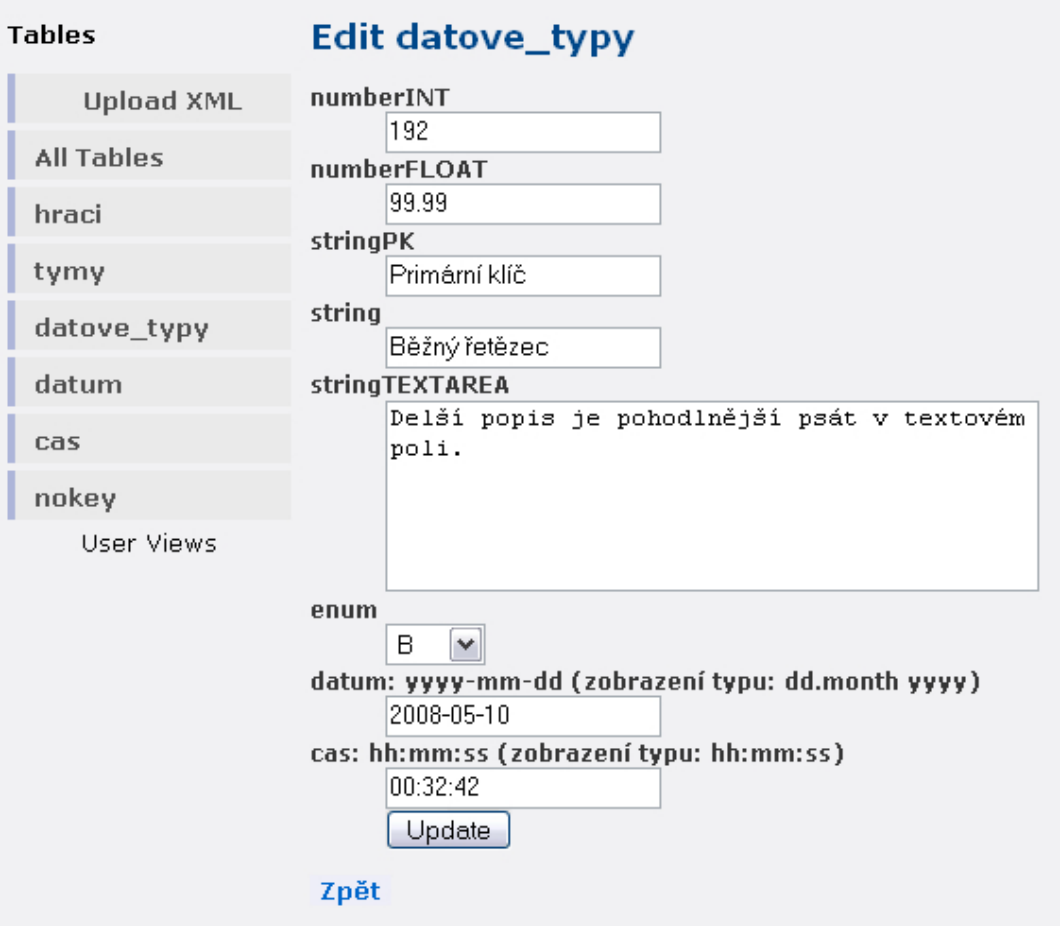
Vytváření a mazání tabulky v databázi

Po definování tabulky v xml souboru, se odkaz s jejím jménem automaticky objeví v levém horním menu. Také se automaticky zjistí, zda daná tabulka již neexistuje v databázi. Pokud ne, editor tuto informaci sdělí uživateli a nabídne možnost tabulku vytvořit. Pokud vytvoření proběhlo úspěšně, informuje o tom uživatele. Následně, po kliknutí na odkaz na tabulku v pravém horním menu, se zobrazí náhled tabulky. Pokud chceme tabulku smazat z databáze, editor nám to umožní pod odkazem AllTables, kde pod každou tabulkou je odkaz provádějící mazání dané tabulky z databáze. Pro úplné odstranění tabulky z editoru je nutné tuto tabulku smazat v konfiguračním souboru, popřípadě pokud jsou uloženy pohledy je nutno smazat i pohledy ze souboru views.xml, kde jsou uloženy.

Přidání, editace a mazání záznamů

Editor umožňuje pohodlnou správu tabulek. Funkce editace a mazání záznamů je umožněno tlačítky nacházejících se vždy na konci každém řádku zobrazující záznam. Přidání nového záznamu nám umožňuje odkaz vlevo nahoře. Tento odkaz, podobně jako u editace, zobrazuje formulář pro danou tabulku. Ukázku editace máme na obrázku 4.3, kde máme ukázkový formulář obsahující všechny formulářové elementy, které lze na základě informací definovaných v konfiguračním xml souboru zadat. Formulář pro editaci a ukládání záznamů vypadá shodně.

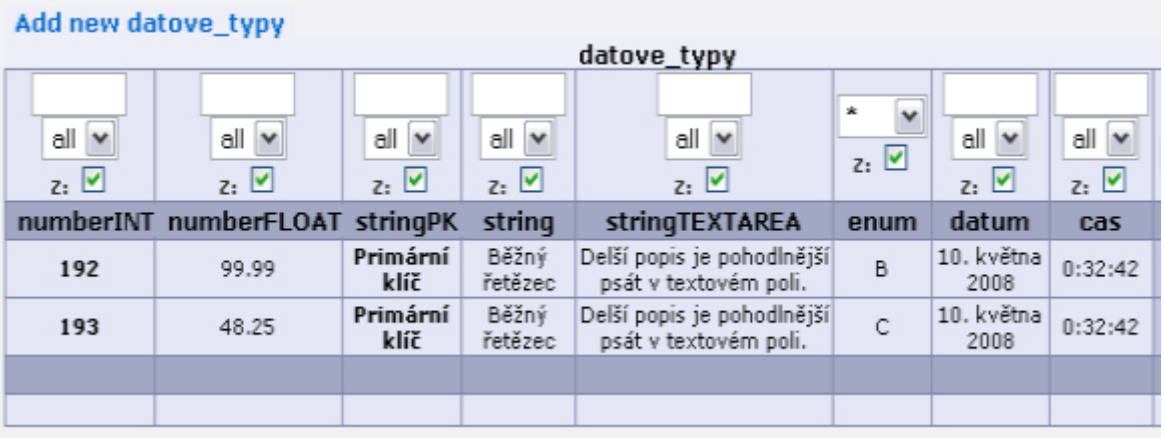
Při implementaci editace záznamů bylo nutno ošetřit problém se změnou primárního klíče.Díky použití metod Zend frameworku a vzhledem k předávání primárních klíčů metodou GET, pro identifikaci konkrétního záznamu, nastal problém po odeslání formuláře. Hodnoty předávané metodou GET měly větší prioritu, než hodnoty primárních klíčů zasílaných metodou POST. Problém byl řešen dočasným přejmenováním nových klíčů před jejich odesláním a následným přejmenováním zpět po jejich přijetí.



Obrázek 4.3: Ukázka editace tabulky různých datových typů.

4.2.6 Datové typy, jejich možnosti a zobrazení

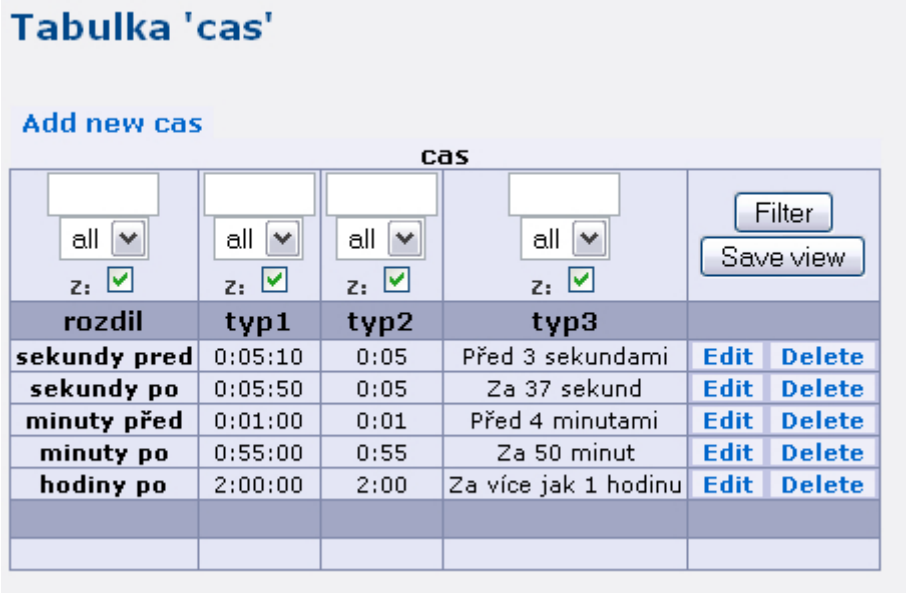
Ukázková tabulka obsahující všechny datové typy 4.1.3 je zobrazena na obrázku 4.4. Datový typ float se zobrazuje podle nastavení v atributu `format`, který je zde nastaven způsobem `%4.2f`. V tabulce se nacházejí dva primární klíče, jeden typ `int` a druhý typu `string`, které jsou v tabulce zvýrazněny tučnou barvou.



Obrázek 4.4: Ukázka tabulky s různými datovými typy.

4.2.7 Práce s časem

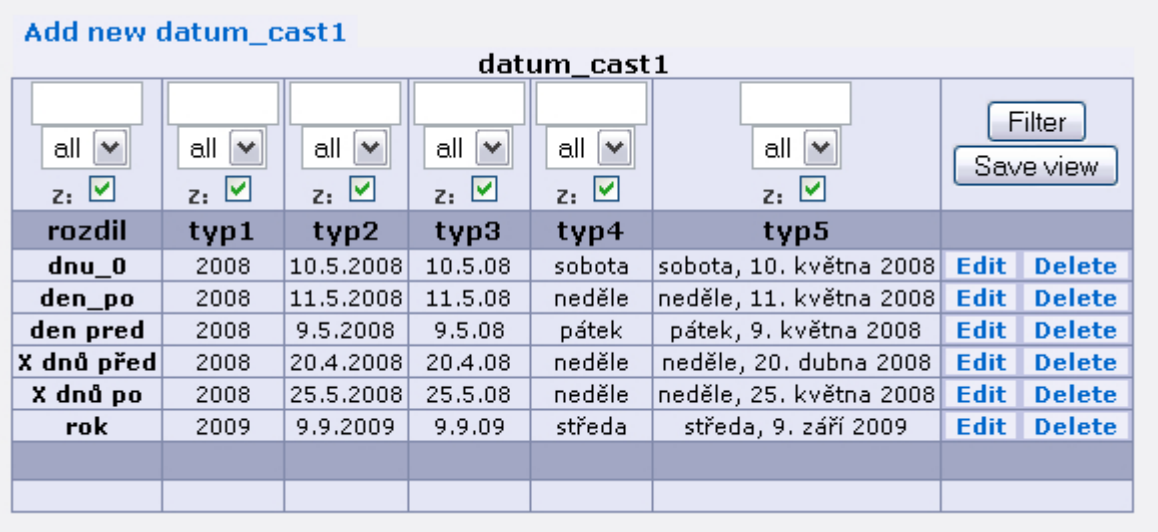
Počet formátu času není už tak početný jako u data. Editor rozlišuje tři typy formátu času, stejně jako datum je určen atributem `format`. Popsány jsou v článku 4.1.3 a ukázka jejích zobrazení je na obrázku 4.5.



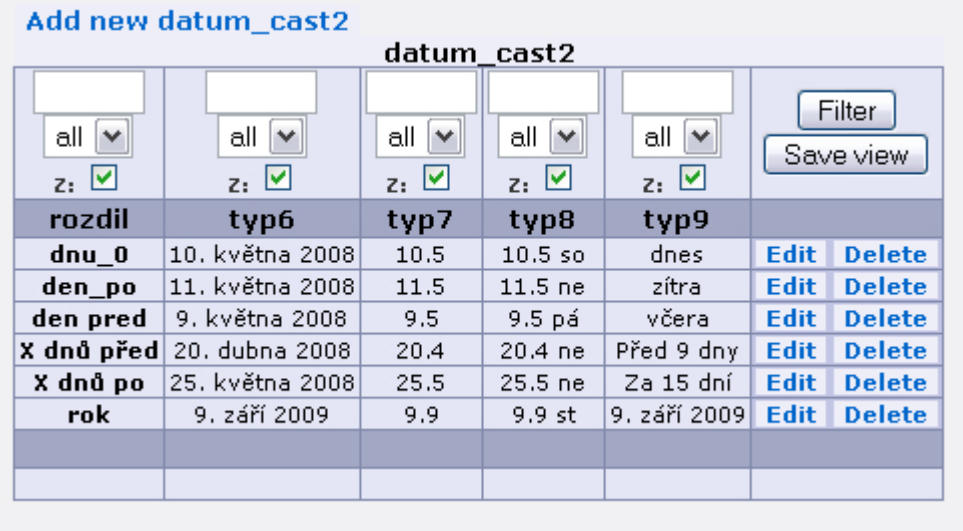
Obrázek 4.5: Ukázka zobrazení různých formátu času

4.2.8 Práce s datumem

Možností zobrazení dat je celá řada z důvodu kombinací číslic a jmen měsíců, kratších či delších vyjádření a podle toho, která část nás zajímá (dny, měsíce, roky) nebo podle informace, jakou by nám datum měl sdělovat (dnes, zítra, za 14 dní). Editor umožňuje definovat 9 různých typů zobrazení datumu v atributu `format` u datového typu datetime, který je společný jak pro datum, tak čas. Jeho rozlišení závisí právě na hodnotě atributu `format`, jeho hodnoty jsou popsány v článku 4.1.3.



Obrázek 4.6: Ukázka možností různých formátů zobrazení dat.



Obrázek 4.7: Ukázka dalších formátů zobrazení dat.

4.2.9 Cizí klíč

Aby byl editor mocným nástrojem pro správu tabulek, je zapotřebí možnost propojování tabulek přes cizí klíč. To nám umožňuje dříve popsaný (viz 4.1.3) atribut sloupce tabulky `foreignkey`, který obsahuje cestu ke sloupci cizí tabulky. Při přidání či editaci záznamu, obsahujícího sloupec s cizím klíčem, jsou nejprve načtena data tohoto sloupce z databáze a ve formuláři zobrazena ve formě výběrového menu. Je to z důvodu, že sloupec nemůže nabývat jiných hodnot, než jsou udány právě v tomto sloupci cizí tabulky.

4.2.10 Filtrace dat

Pro tabulku, obsahující mnoho záznamů v databázi, je důležitá možnost filtrace dat. Pokud je například počet záznamů ve stovkách či tisících, zobrazení všech dat najednou by mohlo být nejen časově náročné a práce s těmito daty nepohodlná.

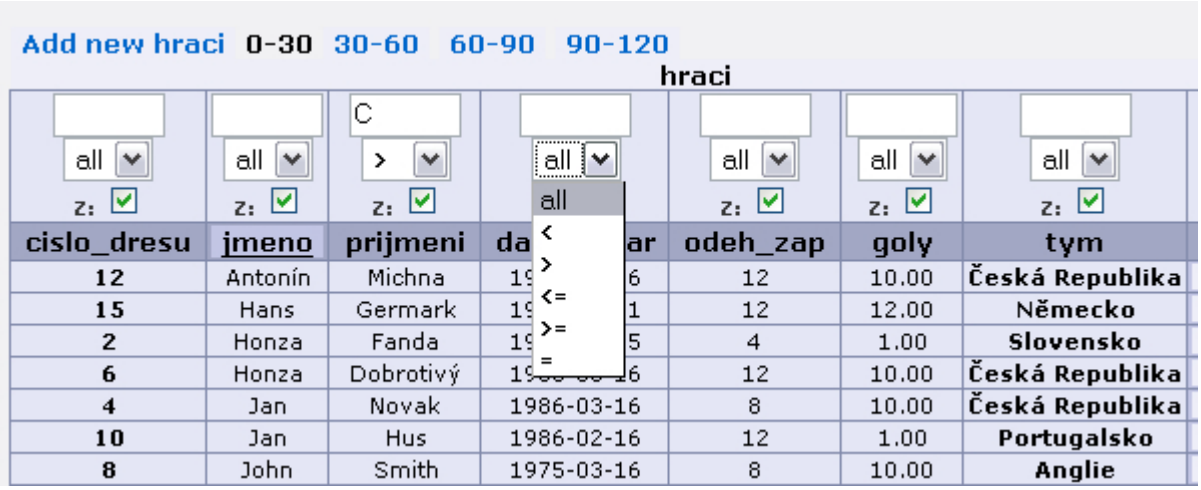
Zlepšení práce s takovouto tabulkou by měla napomoct možnost filtrace a zobrazování záznamů po částech. Tabulku s více záznami vidíme na obrázku 4.8, kde jsou rozděleny na úseky po třiceti. Hodnota 30 záznamů je implicitní, ale pokud by se nacházelo v databázi několik tisíc položek, prohlížení po 30 by se nedalo považovat za efektivní či příjemné. Proto existuje možnost tuto hodnotu změnit atributem `jump` u jména tabulky (viz sekce 4.1.3).

Dalším užitečným prvkem je seřazování dat podle zvoleného sloupce, například podle jména, jak je také ukázáno na obrázku 4.8. Dále zde vidíme vyrolované výběrové menu, kterým si nad každým sloupcem můžeme zvolit, zda zobrazené hodnoty se mají rovnat, být menší nebo větší hodnotě textového pole nacházejícího se nad výběrovým menu.

U sloupců typu `enum` je místo textového políčka výběrové menu, ukázané u tabulky `tymy` (viz 4.1). Je to z toho důvodu, že sloupec nemůže nabývat jiné hodnoty než jsou v něm obsaženy. U tabulek obsahujících cizí klíče jsem se rozhodl, že ponechám textové pole. Je to z důvodu možnosti velkého počtu záznamů, které tabulka může obsahovat. V takovém případě je efektivnější napsat hodnotu manuálně do textového pole.

Třetím formulářovým elementem nad každým sloupcem je checkbox, jehož zatržení rozhoduje, zda sloupec bude zobrazen či nikoli.

Formulář obsahuje dvě tlačítka, jedno pro uložení aktuálního pohledu a druhé pro již zmíněné filtrování dat.



Obrázek 4.8: Ukázka části tabulky s více záznamy a filtrace nad sloupcem

4.2.11 Pohledy

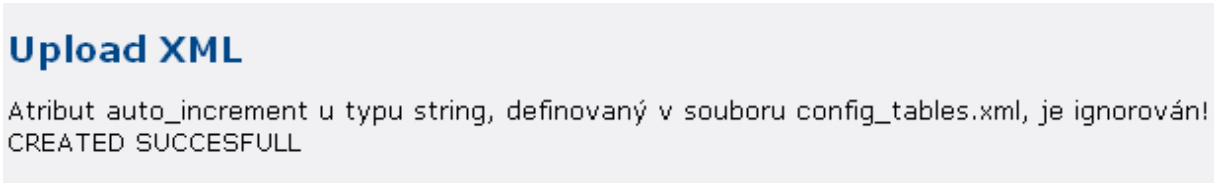
U tabulek, které obsahují mnoho sloupců, se práce s nimi může stát dost nepřehledná. Navíc tabulky často obsahují sloupce, u kterých nepotřebujeme znát jejich hodnotu, ale přesto jsou pro tabulku důležité. Takovými sloupci mohou být například primární klíče. Jedna tabulka může uchovávat data, zahrnující veškeré informace o něčem. My ale v daný moment můžeme potřebovat jen určitá data. Pro tyto situace nám editor umožňuje tvorbu pohledů. Pohled vytvoříme stisknutím tlačítka Save view nacházejícího se v pravém horním rohu tabulky (viz obrázek 4.1). Tím zajistí uložení aktuálního nastavení zobrazení tabulky, do zvláštního xml souboru pro pohledy. Do souboru se pomocí xml atributů ukládá nastavení filtračního formuláře. Na základě informací v tomto xml souboru se vygeneruje levé dolní menu pod oddělovačem User Views. Toto menu nás odkazuje na uložené pohledy. Práce s uloženým pohledem je stejná, jako s původní tabulkou. Editace, mazání a přidávání záznamů probíhá se stejnou tabulkou v databázi, jako při úplném zobrazení. Jedinou odlišností je opětovné uložení pohledu, které vede k aktualizaci zobrazeného pohledu, nikoli k vygenerování pohledu nového. Pokud bychom chtěli vygenerovat další pohled nad tabulkou, na který bychom se odkazovali opět z levého dolního menu, museli bychom uložení provést opět ze základního (kompletního) zobrazení. Mazání pohledů je možno přímo v konfiguračním souboru views.xml.

4.2.12 Chybové hlášky

Vzhledem k dodržování architektury MVC, je nutno veškeré výstupy na obrzovku tisknout v časti pohledu (view). Proto všechny korektní hlášky jsou směrovány na tuto část. Editor vytváří správu tabulek na základě údajů v konfiguračním souboru, který píše pověřená osoba (administrátor). Ta se ovšem může dopustit chyb, na které musí editor co nejlépe zareagovat a chybu nahlásit.

U Závažných chyb, bylo třeba dát pozor na předání chybové hlášky od místa jejího vzniku až ke komponentě view. Proto veškeré mnou definované třídy a formuláře obsahují veřejnou proměnou status, ve kterém se nacházejí případné chybové hlášky. Tuto proměnou Controller testuje zda došlo k chybě a poté předává komponentě view, která chybovou

Obrázek 4.9: Chybová hláška neidentifikovaného cizího klíče



Obrázek 4.10: Definování auto increment u nečíselného sloupce je ignorován

Obrázek 4.11: Pokus přidání záznamů do tabulky s cizím klíčem, kde cizí tabulka je prázdná

hlášku vypíše.

Jedna ze závažných, ale určitě velmi častých chyb, které se programátor(administrátor) dopustí, je nedodržení xml formátu v konfiguračním souboru. Může se např. jednat o pouhé zapomenutí ukončovacího lomítka a funkce pro zpracování xml souboru zahlásí chybu. Tato chyba je nepříjemná, protože na základě údajů v tomto souboru uvedených se generuje postranní menu i náhledy tabulek. Proto tato chyba vede k ukončení další činnosti a vypsání chybové hlášky `Chyba v souboru config_tables.xml`.

Jinou, i když méně závažná chyba je neuvedením primárního klíče v definici tabulky. Méně závažná z toho důvodu, že neovlivňuje ostatní tabulky ani chod editoru. Proto je možné snadno nahlásit chybu jak je uvedeno na obrázku 4.9. Hlášen je také pokus přidání záznamu do tabulky se sloupcem odkazujícím na cizí tabulku, která neobsahuje žádné záznamy. Je upozorněn, jak je uvedeno na obrázku 4.11, aby nejprve zadal data v cizí tabulce. Poté teprve bude umožněno vkládat data i do tabulky s cizím klíčem.

Další chybou, se kterou jsem se při implementování setkal, a bylo nutno ošetřit, bylo udání hodnoty auto_increment v atributu primarykey u sloupce datového typu string. Následkem bylo na první pohled úspěšně vykonaný sql dotaz pro vytvoření tabulky v databázi. Ve skutčnosti ovšem tabulka vytvořená nebyla. Proto editor na tuto chybu reaguje způsobem, že údaj auto_increment u slopců jiných datových typů než number ignoruje. Poupraveným (bez auto_increment) sql příkazem tabulku v databázi vytvoří a o všem korektně informuje stejně jako na obrázku 4.10.

Kapitola 5

Návrh na rozšíření

To co dělá tento editor mocným nástrojem, jsou atributy u sloupců a tabulek, které je schopen zpracovat a uživatel má možnost definovat. Čím více může uživatel ovlivnit automaticky generované věci, tím lépe uživatel dosáhne, co potřebuje. Proto se v následující části zaměřím na popis některých atributů a myšlenek, které z časových důvodů nebylo možno realizovat. Prvním návrhem je možnost spojování tabulek efektivnějším způsobem. Aplikace sice umožňuje v tabulce vytvářet relace mezi tabulkami i ve vztahu M:N, problém však nastává při jejich zobrazení. Je možno zobrazit pouze jednotlivé tabulky, kde vidíme odkazy na cizí tabulku (hodnotu jejího primárního klíče), nikoli však sloupce, které cizí tabulka obsahuje. Tato funkce by mohla být implementována pomocí atributu `show`, který by mohl obsahovat ty sloupce z cizí tabulky, které bychom chtěli zobrazit.

Dalším návrhem pro budoucí obecnější využití, je naimplementovat možnost logování uživatelů s následnou možností nahrání na server svého vlastního konfiguračního souboru přímo z prostředí webové aplikace, nikoli za nutnosti použití FTP klienta. Tím by se aplikace mohla stát využívána kýmkoli a každý by v případě potřeby měl ihned k dispozici správu tabulek, které potřebuje.

Také by editor mohl nabízet přidání, popřípadě mazání, jen určitého sloupce bez nutnosti vytvářet tabulku znovu. V současné době můžeme pouze měnit atributy, které přímo neovlivňují údaje v datbázi, jako je například atribut `format`, `form`, `jump`, dokonce i `foreignkey` a další. Pokud bychom chtěli přidat sloupec nyní, bylo by nutné nejprve již vytvořenou tabulku smazat a znovu vytvořit s novým sloupcem.

Kapitola 6

Závěr

Tato bakalářská práce popisovala vývoj aplikace nazvanou Editor relačních tabulek pro správu dat databáze na základě schématu relační databáze.

Popis byl rozdělen do tří hlavních částí a to analýzy, návrhu a implementaci. V analýze jsme se nejprve věnovali průzkumu aktuálního stavu na trhu a porovnávali jsme různé alternativy k vyvíjenému programu. Tato část byla zakončena motivačním článkem, zmiňující hlavní důvody pro vznik nové aplikace. Byly také uvedeny možné případy užití editoru. Druhá část, věnována návrhu aplikace, nejprve uvedla specifikaci zadání a následně požadavky na metadat. Poté se věnovala možnostmi technologií, které by byly vhodné použít pro vývoj aplikace. Zde se zejména zaměřila na PHP fameworky, ze kterých byl nakonec jeden vybrán a použit pro vývoj aplikace. Nejrozsáhlejší byla poslední část, která popisovala vývoj, některé naimplementované části editoru a standardní chybové hlášky, které museli být v editoru implementovány.

Díky možnosti pracovat na tomto tématu v rámci bakalářské práce, jsem se dobře seznámil s technologiemi pro mě do té doby neznámé a rozšířil znalosti z oblasti webových technologií. Zejména práce se Zend frameworkem, který dodržuje architekturu MVC byla pro mě velmi poučná a do budoucna velice užitečná zkušenost. Také jsem prohloubil znalosti ohledně jazyka XML a funkcích PHP zpracovávající informace definované v tomto jazyce.

Před závěrem této práce byla ještě zvláštní pozornost věnována možným rozšířením a návrhem na možné uplatnění v budoucnu.

Mě, jakožto řešiteli tohoto projektu, při implementaci neustále napadali nové možnosti rozšíření a jaké další funkce by editor mohl nabídnout. Proto věřím, že aplikace nalezne své uplatnění a bude se i nadále rozvíjet.

Literatura

[1] Martin Březovský. Informační systémy na zakázku. http://www.brezovsky.net/page/informacni-systemy.html.

[2] Google. Basic information. http://docs.google.com/support/spreadsheets.

[3] Jiří Kodera. Porazí google office microsoft?, 2006.

[4] phpMyAdmin.cz. phpmyadmin. http://phpmyadmin.cz.

[5] Martin Simbartl. Databázový systém mysql (1.). http://www.pcsvet.cz/art/article.php?id=1255.

[6] Martin Snížek. Xhtml - vývoj (x)html a jeho možnosti. http://interval.cz/clanky/xhtml-vyvoj-x-html-a-jeho-moznosti/.

[7] WEB TVORBA. *Úvod do CSS*.

[8] Wikipedie. Framework. http://cs.wikipedia.org/wiki/Framework.

[9] Wikipedie. Model-view-controller. http://cs.wikipedia.org/wiki/MVC.

[10] Wikipedie. Xml schema. http://en.wikipedia.org/wiki/XMLSchema.

[11] Petr Žižka. Nativní xml databáze - nástin teorie. http://interval.cz/clanky/nativni-xml-databaze-nastin-teorie/, 2003.