TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2646 – Informační technologie

Studijní obor: 1802R007 – Informační technologie

Dynamické webové stránky pro tvůrce stripů a komiksů

Dynamic webpage for creators of strips and comics

Bakalářská práce

Autor: **Jan Brandt**

Vedoucí práce: Ing. Petr Kretschmer

V Liberci 18. 5. 2012

Zadání sem

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Abstrakt

Bakalářská práce má za cíl popsat tvorbu dynamických webových stránek, které mají podobu publikačního systému. Ten se zaměřuje na specifickou skupinu uživatelů, tvůrců webových komiksů a stripů. Zaobírá se především specifickými vlastnostmi, které mají běžně dostupné publikační systémy buď značně nevyvinuté, nebo jim přímo v základu chybí. Jednou takovou vlastností je podpora více jazyků.

Nejdříve dojde na krátký popis několika technologií, které byly při vývoji použity. Jedná se především o programovací jazyky. Práce pak pokračuje v teoretické části, která obsahuje vysvětlení pojmu framework a výběru frameworků pro dva odlišné programovací jazyky, PHP a JavaScript.

Poslední část obsahuje zejména praktické informace k vytvořenému publikačnímu systému. Rozebírán je návrh aplikace a realizace za pomoci zvolených frameworků.

Klíčová slova: publikační systém, Nette framework, GD, dibi, MySQL

Abstract

Bachelor thesis has a goal to describe a creation of dynamic webpages that take form of content management system (CMS). It focuses on specific group of users, creators of comics and strips. Mainly deals with the specific characteristics that are not widely used in CMS or are undeveloped. One such characteristic is support for multiple languages.

First, there is a brief description of several technologies that were used for the development. This consists essentially only a programming languages. Work continues on the theoretical level, where it contains an explanation of the concept of framework and decisions that were made during the selection of frameworks for two different programming languages, PHP and JavaScript.

The last part contains practical information especially about the publication system that was created. Also concept of application design and actual implementation with use of selected frameworks.

Keywords: content management system, Nette framework, GD, dibi, MySQL

Obsah

Seznam obrázků

Seznam tabulek

Zkratky

PHP - PHP: Hypertext Preprocessor

CMS – Content Management System, systém pro správu obsahu

DOM – Document Object Model

HTML – Hypertext Markup Language

XHTML – Extensible Hypertext Markup Language

XML – Extensible Markup Language

HTTP – Hypertext Transfer Protocol

TeX – počítačový program pro sazbu

PostScript – programovací jazyk popisující grafickou podobu tisknutelného dokumentu

SGML – Standard Generalised Mark-up Language

CSS – Cascading Style Sheets, kaskádové styly

ASP – Active Server Pages, skriptovací platforma pro dynamické zobrazování webových stránek

ASP.NET – Součást .NET frameworku, nástupce ASP

Java – programovací jazyk od firmy Oracle (dříve od Sun Microsystems)

JavaScript – skriptovací jazyk určený pro práci na straně klienta, interpretry se nachází v prohlížečích

RDBMS – Relation Database Management System, relační databázový systém

SQL – Structured Query Language, strukturovaný dotazovací jazyk

RSS – Really Simple Syndication (Rich Site Summary), rodina formátů v podobě XML dokumentu určená především pro snadné sdílení nového obsahu webových stránek

MVC – Model, View, Controler, návrhový vzor

MVP – Model, View, Presenter, podobný MVC, ale s odlišnou logikou

WYSIWYG – What You See Is What You Get

URL – Uniform Resource Locator

OOP – Object Oriented Programming, objektově orientované programování

Úvod

Do dnešní moderní doby patří kreslená tvorba už dlouhá léta a posledních několik let zaznamenává díky rozvoji Internetu obrovský rozlet. Poměrně silně rozšířenými se stávají především tzv. webkomiksy, které jsou, jak lze z názvu odvodit, dostupné především skrze globální síť Internet. Nedílnou součástí jsou tedy webové stránky, které umožní tyto komiksy prezentovat a číst. Protože čistě nebo především na komiksy zaměřených publikačních systémů je velmi poskrovnu, rozhodl jsem se v této práci jeden takový vytvořit. Důvodem tohoto rozhodnutí je moje obliba ve čtení a prohlížení webkomiksů a stripů.

Moderní webové stránky mají tendenci poskytovat svůj obsah ve více než jednom jazyce, což může být i případ komiksů. Autor by tak mohl vydávat více jazykových verzí svých komiksů na jedné stránce. Podle tohoto se řídím a v práci se tak zaměřím i na zakomponování této funkce do stránek. Co se týče samotné funkčnosti, tak se inspiruji tím, jak jiné stránky se zaměřením na publikaci webkomiksů vypadají a jak fungují.

Vzhledem k tomu, že mám krátkou osobní zkušenost s frameworky, rozhodl jsem se, že při vývoji využiji jeden framework pro PHP a jeden pro JavaScript. Volba jednoho frameworku z velkého množství není věc jednoduchá, takže osvětlím, i proč jsem zvolil konkrétní framework.

Cílem práce není vytvoření nějakého komplexního publikačního systému, ale jednoduše ovladatelného a úzce zaměřeného.

1 Teorie

1.1 Dynamické webové stránky

Webové stránky mají v dnešní době nezpochybnitelnou roli ve sdílení informací. Pro tyto potřeby byl panem Timem Berners-Leem v akademickém prostředí vytvořen základ pro dnešní jazyk HTML [3], který pomocí značek definuje strukturu obsahu. Berners-Lee postavil HTML na , již v té době používaném, SGML (Standard Generalized Mark-up Language) a rozšířil ho o hypertextové odkazy [3].

Když mluvíme o dynamických webových stránkách, máme tím na mysli stránky, které nejsou tvořeny čistě statickým obsahem. Mezi takovéto stránky patří různé blogy, hry nebo třeba e-shopy. Pro dynamické generování obsahu slouží různé programovací jazyky. Programovací jazyky používané při vývoji stránek se rozdělují na dvě hlavní kategorie, serverové a klientské.

Serverové jazyky se používají na serveru, na kterém jsou umístěny. Jejich hlavním cílem je generování obsahu na straně serveru a práce s daty umístěnými například v databázi. Mezi nejpoužívanější jazyky pro takové účely patří PHP, ASP.NET a Java [8].

Klientské jazyky slouží k programování aplikací nebo jejich částí pracujících na straně klienta. Hlavním a multiplatformním jazykem je v této kategorii interpretovaný jazyk JavaScript. Ten využívá dalších vlastností prohlížečů (např. Document Object Model). Interpretry pro klientské interpretované jazyky jsou součástí většiny prohlížečů.

1.2 Systémy pro správu obsahu

Nejčastěji se s nimi setkáme pod pojmy publikační systémy a CMS (zkratka pro Content Management System). Jedná se o typ dynamických webových stránek, které mají, jak vyplývá z jejich názvu, účel spravovat obsah tvořený uživateli. CMS, jež jsou zaměřeny na publikaci komiksů, jsem našel pouze dva.

ComicPress

Jedná se o rozšíření pro WordPress, který patří mezi nejrozšířenější [6] CMS. Umožňuje WordPressu publikovat komiksy s velmi obsáhlou škálou grafických a funkčních úprav stránek. Naproti tomu přímo nepodporuje více jazykových mutací.

ComicCMS

Jedná se o projekt psaný v čistém PHP, ale stejně jako ComicPress nemá podporu více jazykových mutací a ani dostupnou českou verzi.

1.3 HTML

Vznik HTML se řadí společně se vznikem HTTP a prvního webového prohlížeče do roku 1989 [3]. Mělo jít o náhradu jazyků jako je TeX, PostScript a SGML. První verze standardu vyšla v roce 1991 s číslem 0.9. Poslední dokončená verze je HTML 4.01 a netrpělivě se čeká, až vyjde konečná verze standartu HTML 5. Po HTML 4.01 ještě vznikl standard XHTML, který je přizpůsoben do podoby, při níž lze používat standardní XML parser.

V dnešní době se silně prosazuje trend, kdy se odděluje struktura a obsah od vzhledu. Pomocí HTML se vytváří struktura a obsah a následně pomocí CSS se upravuje jeho vzhled.

DOM

Jedná se o objektově orientovanou reprezentaci HTML nebo XML dokumentu. Jako takový není závislý na žádném jazyku a platformě. Všechny moderní prohlížeče mají zavedenu podporu pro Document Object Model (zkráceně DOM).

Kód 1.1: Příklad HTML stránky

```
1  <html>
2    <head>
3      <title>Ahoj HTML</title>
4    </head>
5    <body>
6      <div class="trida" id="id">
7        <p>Ahoj svete!</p>
8      </div>
9    </body>
10 </html>
```

1.4 CSS

CSS neboli též kaskádové styly je kolekce metod, které popisují, jakým způsobem se mají stránky napsané v HTML, XHTML nebo XML zobrazovat.

V CSS lze pomocí selektorů určit, kterou část chceme ovlivnit. Selektory mohou být dány pomocí:

- **Elementů** – `div`, `p`, `table`, atd.
- **Atributu třídy** HTML – `.trida`, pokud máme definován u elementu atribut `class="trida"`
- **Atributu identifikátoru** v HTML – `#id`, pokud máme definován u elementu atribut `id="id"`

Kód 1.2: Příklad CSS

```
div.trida{ width: 800px; background: black;} /*nastaveni
   sirky a pozadi pro element div tridy trida*/
#id{ height: 600px } /*nastaveni vysky elementu oznaceneho
   identifikatorem id*/
```

1.5 PHP

Jazyk PHP se řadí mezi programovací jazyky, které pracují na straně serveru. Díky své jednoduché syntaxi, jež vychází z takových jazyků jako je C, Java, Pascal či Perl, a bohaté škále funkcí (práce s FTP, generování obrázků či připojení k databázi) se stal velmi oblíbeným. Podle statistik [8] se jedná o nejpoužívanější jazyk ke generování stránek na straně serveru.

Jedná se o interpretovaný jazyk, proto není nutné zdrojové kódy kompilovat. Nicméně díky tomu má oproti kompilovaným jazykům nevýhodu v podobě pomalejší rychlosti zpracování. Podle nezávislého testu [10] jsou většinou interpretované jazyky pomalejší než kompilované. Test ovšem není příliš rozsáhlý, takže se jedná spíše o orientační srovnání.

Trend programovat za pomoci objektů pronikl i do PHP. Poprvé se můžeme setkat s objekty v PHP 3. Největšího rozmachu se ovšem objekty dočkaly v PHP 5, ve kterém doznaly velkých změn a byly takřka kompletně přepracovány. Navíc od PHP 5.3 jsou k dispozici jmenné prostory, které dále rozšiřují možnosti jazyka.

1.6 GD knihovna

Pro práci s obrázky v PHP existuje několik knihoven, v této práci jsem využil možností knihovny GD. Tuto knihovnu jsem zvolil, protože zvládá práci s textem na mnou požadované úrovni a v porovnání s ostatními knihovnami se s ní setkáme mnohem častěji.

1.7 JavaScript

JavaScript patří mezi interpretované objektově orientované jazyky. Používá se zejména jako skriptovací jazyk na straně klienta, tj. v prohlížeči, ale najdeme i možnosti využití na straně serveru (Node.js). Ve spojení s DOM dokáže JavaScript ovlivňovat vzhled a obsah stránek, což se hodí, pokud chceme nějakým způsobem měnit vzhled v závislosti na interakci s uživatelem.

Kód 1.3: Příklad syntaxe jazyka v PHP 5.3

```
<?php
namespace Zaklad; // jmenny prostor
echo 'Ahoj svete!'; //zobrazi text Ahoj svete!
//OOP
class Trida{
  public $name;
  public function __contruct($name){
    $this->name = $name;
  }
  public function getName(){
    return $this->name;
  }
}
$trida = new Zaklad\Trida('Trida');
echo $trida->getName(); //zobrazi text Trida
?>
```

Kód 1.4: Příklad použití GD v PHP

```
<?php
//nahrani obrazek.jpg do pameti
$image = imagecreatefromjpeg('obrazek.jpg');
$width = imagesx($image); //ziskani sirky
$height = imagesy($image); //ziskani vysky
imagejpeg($image); //vraceni obrazku prohlizeci
//ulozeni obrazku do souboru obrazek2.jpeg
imagejpeg($image,'obrazek2.jpeg')
?>
```

1.8 WYSIWYG editor

Takovýto editor slouží pro editaci textů, např. článků, pokud chceme využít prostředí, které nám poskytují textové preprocesory (jako je Word nebo Writer). WYSIWYG (What You See Is What You Get) editor je využíván ve velkém množství CMS. WYSIWYG editorů dostupných pod licencí, která je slučitelná pro použití v bakalářské práci, je několik. Já jsem se rozhodl pro TinyMCE, se kterým mám už zkušenost.

1.9 MySQL

Dynamické webové stránky obvykle využívají nějakou databázi. Nejčastěji se jedná o kombinaci PHP a právě MySQL. Díky tomu se také jedná o jeden z nejrozšířenějších RDBMS (Relation Database Management System). Pro komunikaci s databází se využívá jazyka SQL (Structured Query Language), který má některé rozdíly [5] oproti standardu ANSI SQL.

Kód 1.5: Příklad SQL dotazu, který nám vrátí obsah tabulky Tabulka

```
SELECT * FROM 'Tabulka'
```

MySQL využívá několik rozdílných úložišť dat, které mají specifické vlastnosti. Nejběžnější jsou MyISAM a InnoDB. InnoDB zvládá cizí klíče a transakce. Nicméně se můžeme setkat se situací, kdy společnost nabízející hosting nepodporuje tento typ úložiště. Problematické může být i špatné nastavení InnoDB, které zaviní například delší provádění dotazů.

2 Framework

V teoretické části došlo k představení dvou programovacích jazyků, které se hojně používají při vývoji dynamických webových stránek. Tyto jazyky ovšem mají své neduhy, které někdy nesouvisí přímo s nimi, ale jsou dány např. různorodostí prohlížečů. V této části osvětluji důvody, proč a jaký framework jsem si pro konkrétní jazyk zvolil.

Pod pojmem framework si můžeme představit soubor knihoven, který usnadňuje tvorbu aplikací. V případě dynamických webových stránek tak dostáváme nástroj, jenž umožňuje zjednodušit a urychlit běžné praktiky při vývoji, a který tak omezuje nutnost programovat pokaždé stejné věci.

Výběr frameworku bývá většinou záležitost, kterou je vhodné si řádně rozmyslet a zauvažovat nad tím, co vlastně od takového frameworku čekáme. Důvodem je, že naučit se pracovat s frameworkem na vysoké úrovni, není snadné.

2.1 PHP framework

PHP je poměrně jednoduchý jazyk s velkým množstvím podporovaných funkcí. Díky frameworku se ušetří čas nad stereotypním programováním, což nám dává možnost se věnovat více programování myšlenky. Bývají mnohdy napsané dle nějakého návrhového vzoru.

V době, kdy jsem se rozhodoval s jakým PHP frameworkem budu pracovat, tu byla otázka, jaký mám zvolit, jelikož jich je velké množství. Hlavním faktorem byla především obecná známost a případně podpora v České republice. Tento faktor bohužel není většinou splněn. Základní požadavky splňovaly tři, a to Zend, CakePHP a Nette. Výběr jsem zohledňoval v závislosti na informacích ze série článků, vydaných na serveru Root.cz [2], a informací dostupných v internetových diskuzích.

- **Zend**: Jedná se bezesporu o jeden z nejvíce rozšířených frameworků pro PHP. Má velmi dobrou dokumentaci (především v angličtině). Nicméně díky své mohutnosti se mi jevil jako poměrně obtížný na úplné zvládnutí a také jeho výkon se podle testu [2] jevil silně závislý na pomocných prostředcích PHP serveru (například v testu [2] použitý eAccelerator).

- **CakePHP**: Další z velmi rozšířených frameworků, který bohužel má odlišný problém než Zend, zpomalený vývoj. Z hlediska rychlosti se nachází na druhém místě. Nejnovější verze má problém s nedostatkem příkladů v češtině, které jsem pro jeho pochopení potřeboval.

- **Nette**: Nejmladší z uvažovaných frameworků, a přesto se stal mým hlavním kandidátem a to hlavně díky svému českému původu (framework je vyvíjen pod záštitou Nette Foundation vedeném Davidem Grudlem). Nyní má už poměrně kvalitní dokumentaci. Silnou stránkou tohoto frameworku je především práce s formuláři, kde lze vytvořit snadno validaci jak v PHP tak zároveň i v JavaScriptu. Dobrý dojem na mne zanechal interní ladící nástroj známý jako Laděnka.

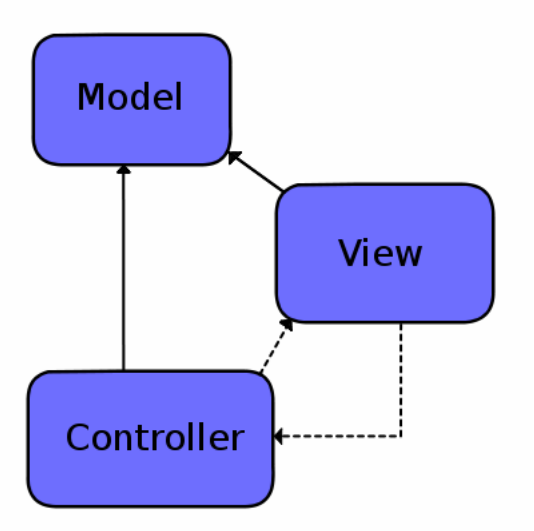
Na základě diskuzí, které jsem si prošel, a toho, že se mi nejlépe pracovalo s Nette, jsem ho použil pro vyvíjený systém.

2.2 Nette

Jak už bylo zmíněno, jedná se o poměrně mladý framework, počátky se datují někde do roku 2004 [4]. Jako open-source byl uvolněn v roce 2008. Momentálně je vyvíjen českou komunitou, která si ho velmi oblíbila. Rovněž si dělá ambice na rozšíření ve světě.

2.2.1 Model-View-Controller

Pro Model-View-Controler se používá zkratka MVC. Jedná se o návrhový vzor pro software, který se hojně používá u architektury webových aplikací. Aplikace se rozděluje na tři oddělené komponenty, jak vyplývá z názvu, a to model, view



Obrázek 2.1: MVC

a controller. Vycházejíc z informací popsaných v [1] mají jednotlivé části následující funkce:

- **Model** má na starost obstarání dat. Data mohou být v různé podobě a můžeme je získávat z různých prostředků, např. databáze.
- **View** (též pohled) se stará o zobrazení získaných informací z modelu.
- **Controller** reaguje na interakci od uživatele a podle toho pracuje s modelem a/nebo s pohledem.

Nette je na tomto návrhovém vzoru postaveno, resp. na jeho obdobě MVP (Model-View-Presenter), kde funkci controllerů zastupují presentery, jak je popsáno v dokumentaci [4].

Models

Jak už bylo zmíněno, tak modely obsahují práci nejčastěji spojenou s databází, ale mohou pracovat například se soubory. Pro práci s databází v modelech jsem použil knihovnu `dibi`.

Dibi

Knihovnu `dibi` jsem upřednostnil před nativní třídou pro práci s databází, `\Nette\Database`, protože se mi s ní při psaní SQL dotazů pracovalo lépe.

Dibi lze používat dvěma způsoby:

- jako statickou třídu
- jako instanci třídy `DibiConnection`

U každého způsobu ještě můžeme zvolit způsob zadávání dotazu pomocí:

- SQL příkazů
- fluent rozhraní

View

V Nette se též používá pojem pohled. Nette pro pohledy využívá šablonovací jazyk „latte“, který byl speciálně vytvořen při vývoji frameworku. Soubory „latte“ mají odpovídající příponu `latte`. Šablony jsou i přes generování rychlé, jelikož probíhá pouze při změně zdrojových kódů nebo první návštěvě pohledu.

Kód 2.1: Ukázka syntaxe latte

```
{block content}
        <span>Promenna: {$promenna}</span>
{\block}
```

V ukázce je vidět použití proměnné $promenna, kterou jsme do šablony poslali z presenteru pomocí kódu ukázaného dále.

Kód 2.2: Odeslání proměnné z presenteru do šablony

```
public function renderDefault(){
        $prom = "Nejaka promenna";
        $this->template->promenna = $prom;
}
```

„Latte“ je natolik pokročilé, že s ním můžeme psát kód, který takřka vypadá jako (X)HTML. Na příkladu je uveden způsob, jakým projít všechny články, které byly do šablony odeslány.

Kód 2.3: Různé způsoby užití maker latte

```
//tradicni foreach makro
<div>
{foreach $articles as $article}
        <span>{$article->name}</span>
{\foreach}
</div>
//a ted za pomoci n:makra
<div n:foreach="$articles as $article">
        <span>{$article->name}</span>
</div>
```

Presenter

Presentery představují třetí z pilířů každé aplikace napsané v Nette. Presentery jsou potomky třídy `Nette\Application\UI\Presenter`. Jejich účelem je dle obr. 2.1 zejména komunikace mezi modelem a pohledem.

Routování

Aby mohla aplikace vytvářet „pěkná URL“ (též známé pod anglickým „cool URL“), bylo nutné v souboru bootstrap.php vytvořit takzvanou „routu“ (podle anglického route, česky cesta). Ta definuje podobu URL a podle její formy se bude v aplikaci volat presenter, modul a pohled s parametry. Pokud některá část nebude odpovídat žádné z definic, dojde k chybě. K vytváření cest se používá třída `Nette\Application\Routers\Route`.

Tohoto využívají presentery a šablony pro odkazy, které tak nemusíme složitě generovat. Pro ukázku si nejdříve nadefinujeme „routu“:

```
$container->router[] = new Route('<presenter=Default>/<action
   =default>[/<id>]');
```

Kód 2.4: Ukázka tvorby odkazů v šablonách a presenterech

```
//Pomoci n:makra v sablone
<a n:href="Article:show $articleId">Clanek</a> //vytvori odkaz
    napriklad na clanek s $articleId=10 v podobe /article/show/10
//V presenteru pomoci metody link
public function actionDefault(){
                ...
        $url = $this->link('Article:show', $articleId); //vytvori odkaz
             napriklad na clanek s $articleId=10 v podobe /article/show
            /10
                ...
}
```

2.2.2 Moduly

V složitějších aplikacích se můžeme setkat s potřebou rozdělit aplikaci na menší celky. Pro takové potřeby byly v Nette zavedeny moduly. V adresářové struktuře musí být moduly ve stejnojmenné složce. Presentery jsou tím také ovlivněny, a proto musí mít definované jmenné prostory ve tvaru `<NazevModulu>Module`, v případě submodulů `<NazevModulu>Module\<NazevSubmodulu>Module`.

2.2.3 ACL

Jedná se o zkratku pro Access control list, což by se dalo do češtiny přeložit jako Seznam správy přístupu. ACL slouží ke správě libovolně velké specifikace oprávnění.

Princip ACL stojí na třech základních komponentách:

- Role
- Zdroje
- Operace

Každému uživateli je přidělena jedna nebo více rolí, příkladem rolí může být *guest* (nepřihlášený uživatel) nebo *administrator* (administrátor stránek). Zdroje bývají objekty, které jsou chráněny, například obrázky, komentáře atd. Operace představují akce, které chceme provést nad zdrojem (například smazat, upravit či

zobrazit). Role mohou vytvářet hierarchii, ve které potomek dědí práva od předka a rozšiřuje je o vlastní. ACL podle toho vytváří strom, podle kterého se nastavují práva přístupu a oprávnění.

V Nette je ACL součástí frameworku, takže nebylo nutné vytvářet vlastní. Pro nepřihlášené uživatele používá framework roli *guest*. Ostatní role už musí být definované programátorem.

2.3 JavaScript framework

Kvůli problémům, které vládnou kolem stále odlišných implementací JavaScriptu v prohlížečích, jsem se rozhodl použít framework. Při výběru jsem se řídil především rozšířeností [7] a také tím, jak snadno se dá s frameworkem pracovat. V úvahu přicházela dvojice frameworků, jQuery nebo Prototype.

Z hlediska rychlosti jsou si rovny. Horší dokumentace a také silný vliv Ruby na straně Prototype vedly k tomu, že jsem raději zvolil jQuery. To používá velké množství programátorů, lze tak najít velké množství návodů a doplňujících pluginů.

2.4 jQuery

Framework jQuery je podle své dokumentace [9] rychlý a stručný. Jeho selektory jsou velmi podobné těm v CSS, což usnadňuje práci a snižuje množství informací, které programátor musí znát. Při práci s frameworkem se pracuje nejčastěji s objektem daným předpisem `$(document)`. Alternativní verze zápisu je `jQuery(document)`.

Příklad syntaxe při použití jQuery:

```
//HTML kod
<a href="">Link</a>
//jQuery kod
$(document).ready(function() {
   $("a").click(function() {
     alert("Hello world!");
   });
 });
```

3 Návrh

Během návrhu jsem se řídil průzkumem, při němž jsem prošel několik webových stránek, které uveřejňují komiksy. Na základě tohoto průzkumu jsem si poznamenal několik funkcí, které by takové stránky měly ovládat, a také to, co od nich očekávám. Dále jsem se rozhodl, že chci publikační systém něčím rozšířit. Tedy nejde pouze o inspiraci, ale též o vylepšení.

3.1 Požadované funkce aplikace

Komiksy

Aplikace se točí hlavně kolem komiksů, takže základem je jejich správa. Komiksy lze rozdělit do sérií, které mohou mít více jazykových mutací. V sériích je možné komiksy dále jemněji třídit podle kapitol. Sérii lze určit, zda obsahuje komiksy, které se čtou v euroamerickém stylu (zleva doprava), nebo komiksy ve stylu manga (zprava doleva). Tato volba ovlivní pouze zobrazení upozornění, jenž u mangy informuje o správném směru čtení.

Články

Články mají pouze doplňující funkci, takže se jedná o systém, který nepočítá se složitým členěním. Článku lze přidělit pouze jednu kategorii. Pro editaci textů se používá WYSIWYG editor TinyMCE.

Archiv

Musí mít grafické ztvárnění ve stylu kalendáře či seznamu již dosud vydaných komiksů v jedné sérii. Z toho musí být jasné, kdy jaký komiks vyšel, včetně odkazu

na něj.

Komentáře

Jelikož moderní stránky velmi často obsahují tuto funkci, rozhodl jsem se umožnit přidávání komentářů ke komiksům. Komentáře se zobrazují hned pod komiksem.

RSS

RSS (Really Simple Syndication) chci použít, protože se objevuje na skoro všech moderních stránkách a sám ho využívám pro sledování obsahu velkého množství stránek. Kanály budou vytvářené pro všechny série a pro jednotlivé série.

Editace textů

Provedení by mělo být co nejjednodušší, jelikož není přímým cílem vytvářet editor ve stylu GIMPu či Photoshopu. Tato funkce není něco, co jsem během průzkumu nalezl jako běžné či aspoň přítomné v redakčních systémech použitých pro různé stránky s webkomiksy. Editor je určen pro úpravu dialogů, které jsou tvořené především v horizontálním směru.

Jazykové mutace komiksů a článků

K moderním webovým stránkám patří také podpora více než pouze jednoho jazyka, a tak autor komiksů dostává do rukou možnost vydávat svou tvorbu v různých jazycích. Samozřejmě totéž platí i pro uživatelské rozhraní. Komiksy jsou rozdělovány podle jazykových mutací sérií. U článků jsou jazykové mutace řešeny u samotných článků i u kategorií.

3.2 Uživatelé a role

K publikačním systémům patří správa uživatelů. Vzhledem k tomu, že chci využít možností ACL v Nette frameworku, rozhodl jsem se rozdělit uživatele do skupin, které představují role. Pro snadnou správu jsem si určil 4 skupiny uživatelů.

- ***Host***: Nejběžnější uživatel stránek, patří sem kdokoli, kdo není přihlášený.

- ***Registrovaný***: Nejnižší uživatelská práva na stránkách, patří sem kdokoli, kdo se zaregistruje. Jediná povolená akce oproti hostovi je správa vlastních uživatelských údajů.
- ***Autor***: Práva pro uživatele na této úrovni jsou rozšířena o tvoření sérií, kapitol a přidávání komiksů. Má přístup pouze k vlastní tvorbě.
- ***Administrátor***: Nejvyšší uživatelská práva, může zasahovat do dění celé stránky. Oproti autorovi mu přibude možnost spravovat fonty pro editor.

Stránky by měly uživatelům umožnit se zaregistrovat, a tak jim zajistit, že vybraná přezdívka nebude používána v komentářích někým jiným. Administrátor pak bude moci změnit uživatelům role, například na autora.

4 Realizace

4.1 Adresářová struktura aplikace

Adresářová struktura vychází z frameworkem doporučené [4]. Nicméně jsem provedl menší změny s ohledem na snadnou instalaci a orientaci. Běžně doporučovaná struktura je taková, kdy máme soubory týkající se přímo webových stránek ve složce www. Ta obsahuje například JavaScript, CSS soubory nebo třeba obrázky použité pro vzhled webových stránek. Vzhledem k tomu, jak funguje většina hostingů, bylo toto nutné změnit a tak vznikla následující adresářová struktura:

```
/
├── app
├── libs
├── temp
├── log
├── css
├── js
├── img
├── fonts
├── theme
│   ├── css
│   ├── js
│   └── img
├── .htaccess
└── index.php
```

4.2 Aplikace

Po zvážení jsem rozhodl dělit aplikaci na moduly v závislosti na funkcích, které jednotlivé části mají. Podle toho vzniklo větší množství modulů a některé submoduly.

Jelikož Nette využívá možností OOP, rozdělil jsem aplikaci takovým způsobem, že všechny *Frontend* presentery dědí od *BasePresenter* a *Backend* presentery od *SecuredPresenter*, který dědí od *BasePresenteru*. *SecuredPresenter* pokaždé ověřuje, zdali je uživatel přihlášen. Díky tomuto jsou některé proměnné dostupné pro všechny presentery nebo pouze *Backend* presentery. *Backend* presentery určují například, zda má uživatel práva k přístupu. Z tohoto důvodu se ve skoro všech složkách *templates* nachází dvě složky, *Frontend* a *Backend*, které obsahují šablony pro odpovídající presentery.

4.2.1 Součásti aplikace

Jelikož se při vývojovém režimu objevuje `Debug Bar`, zvolil jsem několik komponent, které ho rozšiřují. Díky `NetteTranslatoru` v `Debug Baru` máme položku, která umožňuje překládat rovnou při vývoji stránek. Má ovšem pár omezení, například problémy s překlady `flashMessages`. Ty se neobjeví v nabídce, dokud nejsou vyvolány. Další rozšíření má samo `dibi`, které ukazuje, jaké SQL dotazy byly na aktuální stránce provedeny. Šikovným pomocníkem je i `Component Tree`, který zobrazuje velké množství informací vázajících se k aktuální stránce, třeba parametry presenteru a akce či obsažené komponenty a jejich případná hierarchie. `ServicesPanel` zase seskupuje informace spojené se službami, například `translator` nebo `router`.

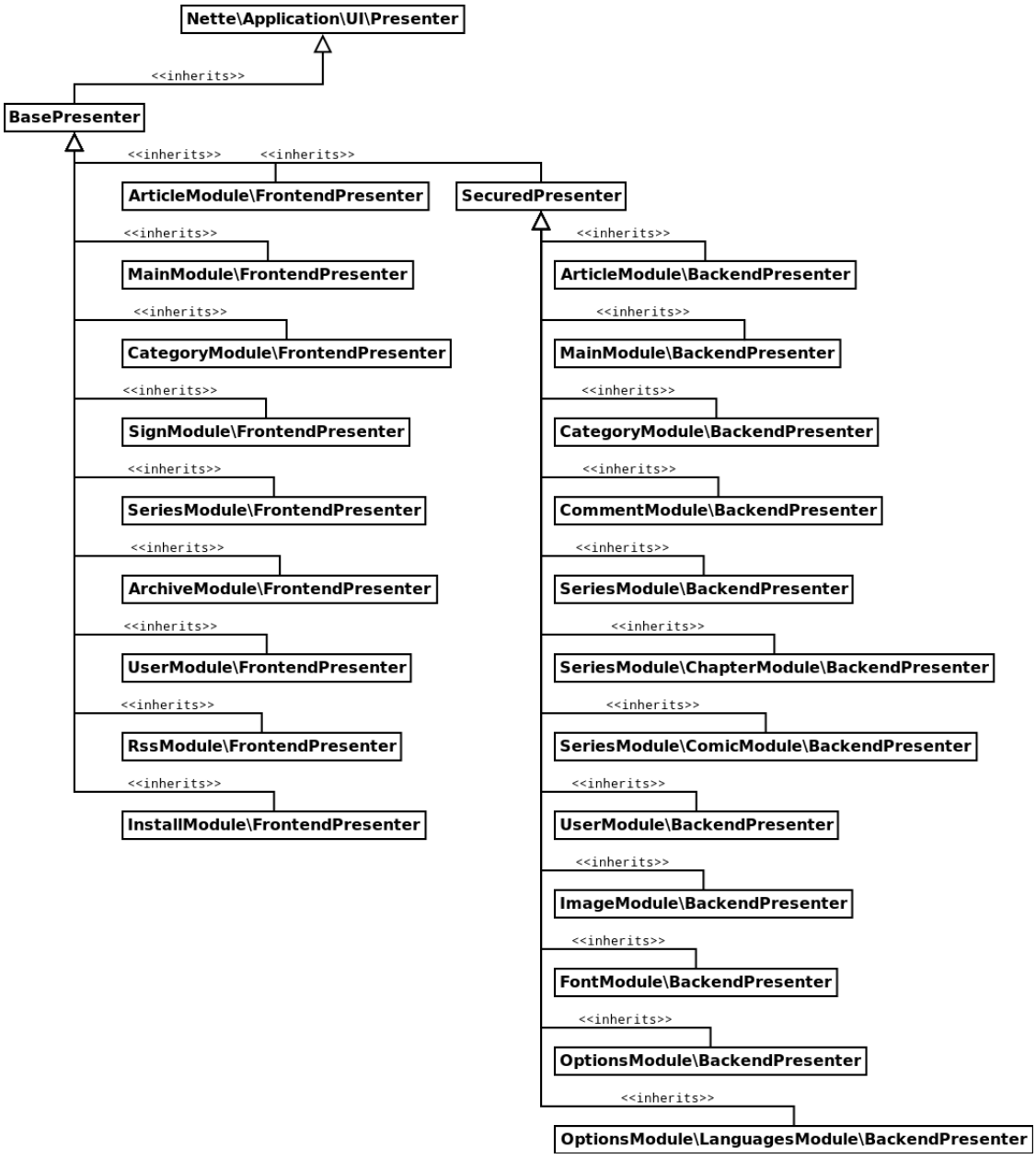
Jelikož jsem se potýkal s problémy při vytváření jednoduchého a přitom hezkého designu, zvolil jsem nakonec `Twitter Bootstrap`. Ten je volně použitelný, což bylo hlavní kritérium. Jeho použití ovšem s sebou přineslo několik změn, které bylo nutné zohlednit. Týkalo se to zejména formulářů a navigace.

4.2.2 Požadavky aplikace

Aplikace potřebuje pro své fungování několik prostředků. Většina se odvíjí od Nette frameworku. Požadavek na databázi jsem určil na základě možností většiny webových hostingů, které mají nějakou nabídku hostování zdarma.

- PHP 5.3+
- GD 2+

- MySQL 5.1+



Obrázek 4.1: UML diagram *presenterů* s jejich jmennými prostory

4.2.3 Moduly

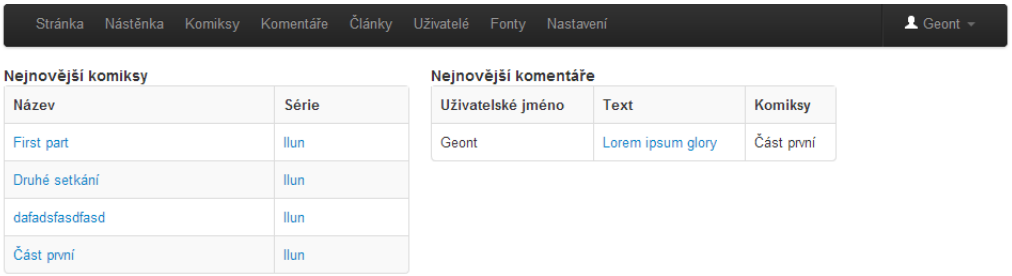
Každý modul má stejnou adresářovou strukturu, která obsahuje složky *models*. Do ní se ukládají modely potřebné pro modul. Dále je zde složka *presenters*, ve které se ukládají *presentery*, přesněji *FrontendPresenter* a *BackendPresenter*. Nakonec je tu složka *templates*, ve které jsou uloženy všechny šablony potřebné pro modul.

Kód 4.1: Definice „route“ v aplikaci

```
$container->router[] = new Route('index.php', array('module' => 'Main', 'presenter'
    => 'Frontend', 'action' => 'default'), Route::ONE_WAY);
$container->router[] = new Route('admin/<module=Main>/<action=default>[/<id>[/<
    text_id>]]',array('presenter'=>'Backend'));
$container->router[] = new Route('<lang=cs [a-z]{2}>/<module=Main>/<action=default
    >[/<id [0-9]+>][/<url>[/<comicUrl>]]',array('presenter'=>'Frontend'));
```

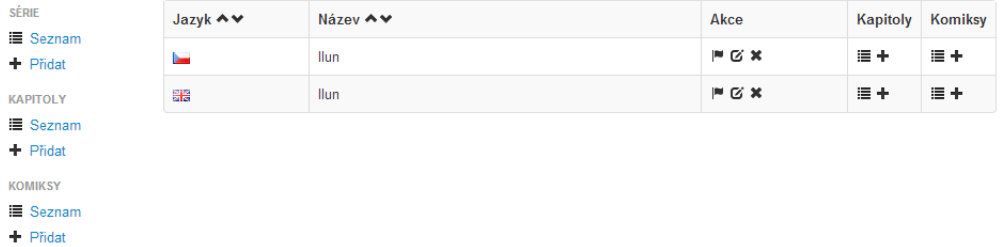
Main

Modul obstarává vzhled a obsah základní stránky (jak v uživatelské části, tak v administraci). V administraci je hlavním cílem modulu zobrazení základního obsahu, kde nejsou funkce, které ovlivňují obsah stránek. Najdeme tu tedy odkazy na poslední zveřejněné komiksy a nejnovější komentáře. V uživatelském prostředí zase zobrazuje nejnovější komiks a případně i poslední články v modulu `Nastavení` určené kategorii.



Obrázek 4.2: Hlavní stránka v administraci

Series



Obrázek 4.3: Administrace sérií

Základem aplikace jsou série, takže tu máme modul, který s nimi pracuje. V administraci obstarává přidávání, úpravy a mazání jednotlivých sérií a jejich jazykových

variant. V uživatelském prostředí má úlohu zejména pro zobrazení komiksů v sérii a jazyce.

Přidat – nám umožňuje přidání série. Musíme zadat její název a jazyk, který vybereme z dostupných jazyků. Pak máme možnost označit sérii za veřejnou. Důležité je i vybrat typ série. Na výběr je euroamerický nebo manga.

Upravit – poskytuje možnost upravení stejných informací jako při přidávání.

Smazat – nám smaže sérii a všechen její obsah.

Přidat jazykovou variantu – nás odkáže na stejnou stránku jako **Přidat**, ale s menší změnou v URL, kdy nám přibude číslo skupiny, do které bude nová série spadat. Podle toho se odvíjí i volby jazyků, kde chybí ve skupině už jednou použité jazyky.

Seznam – odkazuje na seznam kapitol nebo komiksů, které patří k sérii.

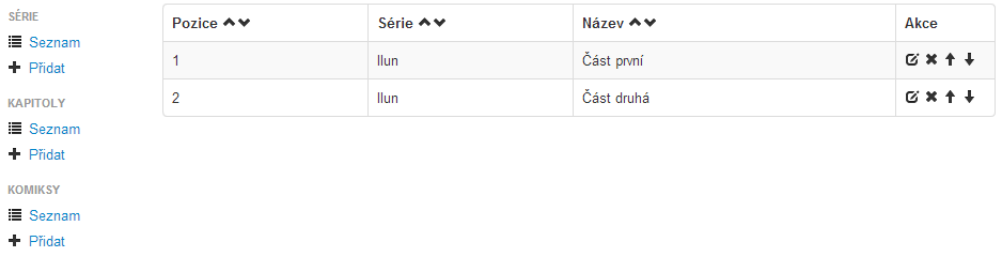
Přidat (v sekci kapitola nebo komiks) – nás odkáže na stránku, kde můžeme přidat kapitolu/komiks.

Obrázek 4.4: Zobrazení komiksu návštěvníkům

Tabulka 4.1: Uživatelská práva modulu `Series`

Akce	**Registrovaný**	**Autor**	**Administrátor**
Přidat	Ne	Ano	Ano
Upravit, smazat, seznam, přidat jazykovou variantu, Přidat kapitolu/komiks	Ne	Vlastní	Ano

Chapter



Obrázek 4.5: Administrace kapitol

Submodul v administraci pomáhá spravovat kapitoly série. Seznam kapitol se ukazuje pouze pro konkrétní sérii. Komiksy, jež nejsou zařazené do některé kapitoly, jsou v administraci stále přístupné a zobrazují se v uživatelském prostředí, pokud jsou zveřejněné.

Přidat – po nás vyžaduje jazyk, sérii, do níž kapitola má patřit, a název kapitoly. Pozice je automaticky nastavena na konec série.

Upravit – nám umožní změnit jazyk, sérii, do níž kapitola má patřit, a název kapitoly.

Smazat – způsobí smazání kapitoly, ale nikoliv komiksů, které jsou v ní. Komiksy osiří a budou považovány za nezařazené.

Pozice zmenšit/zvětšit – změní pozici kapitoly požadovaným směrem.

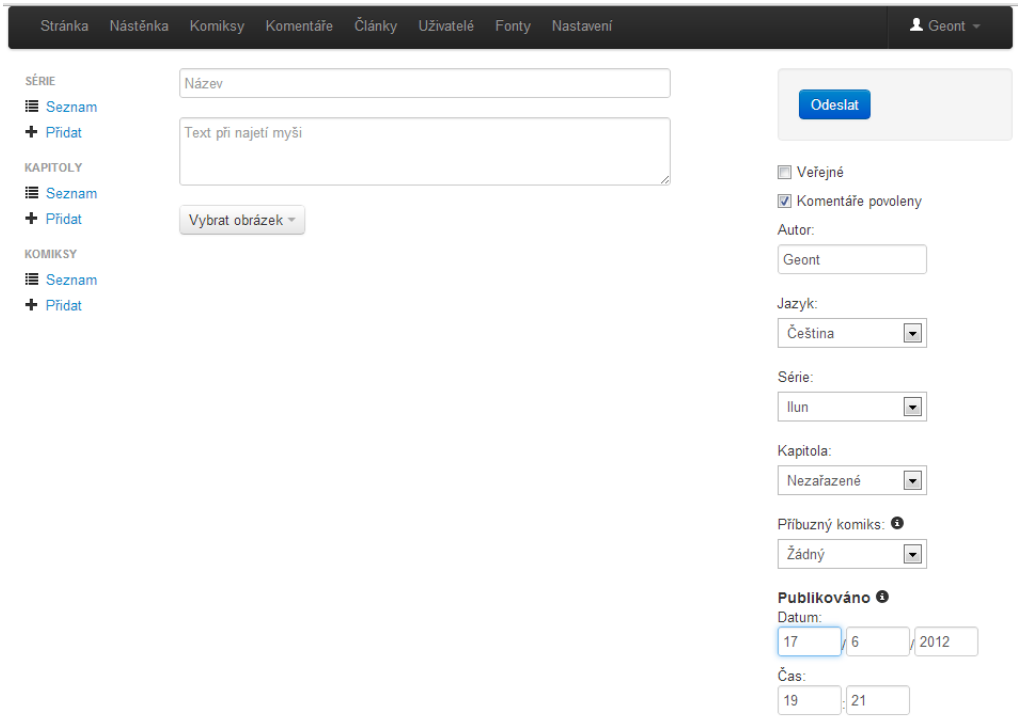
Tabulka 4.2: Uživatelská práva modulu `Chapter`

Akce	**Registrovaný**	**Autor**	**Administrátor**
Přidat	Ne	Ano	Ano
Upravit, smazat	Ne	Vlastní	Ano

Comic

Tento submodul má pouze administrační část, která se stará o správu komiksů v sérii. Komiksy jsou seřazeny podle zadaného data publikace.

Přidat po nás bude požadovat řadu informací, ale ne všechny jsou povinné. Mezi povinné patří jazyk, série, do níž kapitola má patřit, název, jméno autora (předvyplněno jako přezdívka osoby, která komiks vkládá), zda chceme komiks rovnou zveřejnit, zda obsahuje tento konkrétní díl obsah pouze pro dospělé, datum publikování a obrázek, který představuje komiks. Nepovinné údaje jsou kapitola, pod niž komiks spadá, *hover* text, který se zobrazí při najetí ukazatele myši na obrázek a příbuzný komiks. Příbuzný obrázek určuje pozici obrázku v sérii a také umožňuje použít původní nebo upravený obrázek příbuzného komiksu při volbě obrázku.



Obrázek 4.6: Přidání komiksu

Pro nahrání obrázku slouží `plupload`, který je umístěn v modal okně. Po nahrání se okno zavře a stejně jako v případě zvolení obrázku z příbuzného komiksu dojde k jeho okamžitému zobrazení. Součástí nahrání obrázku na server je kontrola existence pomocí hashe obsahu souboru nikoliv jeho názvu, zdali tento soubor už je použit. Pokud není ještě v databázi, tak se vytvoří a přidá do databáze hash a cesta k obrázku. V opačném případě se nahraný soubor smaže a použije se hash exis-

tujícího.

Během smazání komiksu se kontroluje, zda je obrázek použit ještě u jiného komiksu. Pokud tomu tak skutečně je, obrázek se nesmaže.

Upravit – umožňuje upravit skoro všechny údaje jako při přidávání, ale neumožňuje znovu nahrát obrázek.

Smazat – smaže komiks i s obrázkem, pokud ten není nikde jinde používán. Ověřuje se podle hashe vytvořené z obsahu.

Upravit texty – přesune autora nebo administrátora do editoru textů, kde může přidávat a upravovat texty v obrázku.

Tabulka 4.3: Uživatelská práva modulu `Comic`

Akce	**Registrovaný**	**Autor**	**Administrátor**
Přidat	Ne	Ano	Ano
Upravit, smazat	Ne	Vlastní	Ano

Article

Modul se stará v administraci o správu článků a v uživatelském prostředí o jejich zobrazení.

Obrázek 4.7: Administrace článků

Přidat – nám umožní vložit článek. Pro vložení potřebujeme znát název, v jakém jazyce článek je, kategorii, zdali ho označit jako veřejný, text článku, pro který je využit `TinyMCE`, a nakonec krátký popis článku.

Přidat jazykovou variantu – dělá to samé jako **Přidat** s tím rozdílem, že očekává v URL číslo skupiny, pod kterou patří nový článek v dalším jazyce.

Upravit – umožňuje upravit všechny údaje v článku.

Smazat – smaže článek.

Akce	Registrovaný	Autor	Administrátor
Přidat	Ne	Ano	Ano
Upravit, smazat, přidat jazykovou variantu	Ne	Vlastní	Ano

Category

Modul `Category` slouží ke správě kategorií pro články a v uživatelském prostředí k zobrazení článků dané kategorie.

Obrázek 4.8: Administrace kategorií

Přidat – nám umožní vložit kategorii, kterou následně můžeme použít třeba při vytváření menu nebo pro zobrazení na domovské stránce. Pro vložení potřebujeme znát název a v jakém jazyce má kategorie být.

Přidat jazykovou variantu – dělá to samé jako **Přidat** s tím rozdílem, že očekává v URL číslo skupiny, pod kterou patří nová kategorie v dalším jazyce.

Upravit – umožňuje upravit všechny údaje kategorie.

Smazat – smaže kategorii a všechny k ní náležící články se stanou *Nezařazené* (lze k nim stále přistupovat přes administraci).

Tabulka 4.4: Uživatelská práva modulu `Category`

Akce	Registrovaný	Autor	Administrátor
Přidat	Ne	Ano	Ano
Upravit, smazat, přidat jazykovou variantu	Ne	Vlastní	Ano

Font

Modul `Font` – je čistě pro administraci a plní funkci správy fontů využívaných editorem textů. Tento modul může používat pouze administrátor.

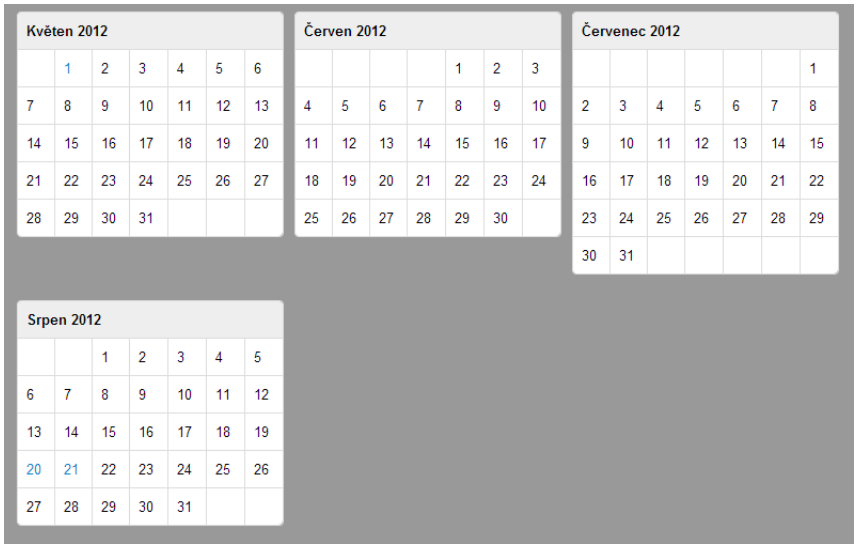
Přidat – po nás vyžaduje název fontu, který se bude ukazovat v editoru, podporu jazyků z aktuálně dostupných a nakonec soubor typu `ttf`.

Upravit – nám umožní upravit název a podporu jazyků u aktuálně dostupných. To lze využít například, pokud jazyk přidáme až po přidání fontu a víme, že je font pro tento jazyk použitelný.

Smazat – smaže soubor a záznam v databázi.

Archive

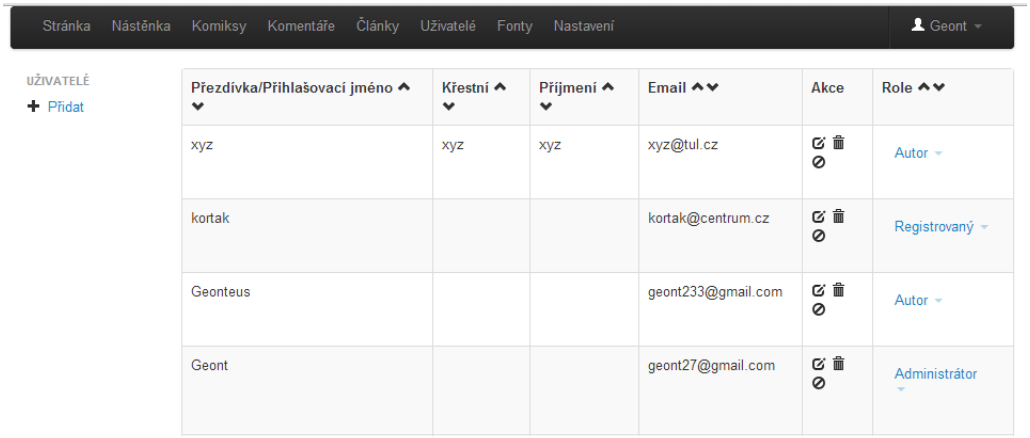
Obsahuje pouze uživatelskou část, která zobrazuje pro série archiv již vyšlých komiksů v podobě kalendáře. Kalendář je vytvářen skrze stejnojmennou komponentu.



Obrázek 4.9: Zobrazení archivu

User

Modul umožňující práci s uživatelskými údaji. Administrátor zde může upravovat vlastní a cizí údaje nebo přidávat nové uživatele. Ostatní uživatelé (registrovaní a autoři) zde mohou upravovat své vlastní údaje. Pro uživatelské prostředí funguje pouze zobrazení několika informací o uživateli a možnost se zaregistrovat.



Obrázek 4.10: Administrace uživatelů

Přidat – funguje podobně jako registrace a vyžaduje po nás řadu informací, přihlašovací jméno, které se zároveň chová jako přezdívka, heslo a jeho potvrzení, jazyk prostředí užívaném v případě administrace a email uživatele. Doplňující informace jsou křestní jméno a příjmení a zdali splňuje podmínky, že je ve své zemi uznán jako dospělý.

Upravit – umožní změnit údaje, které jsme zadali, kromě přihlašovacího jména.

Zabanovat/odbanovat – dokáže nastavit status „ban“ (česky zákaz) uživateli. Ten mu znemožní se přihlásit a tak přispívat do komentářů či na stránky. Při pokusu o přihlášení se zobrazí zpráva o tom, že byl uživateli udělen „ban“.

Aktivovat – se používá v případě nových registrací. Ty jsou automaticky neaktivní a administrátor je musí ručně aktivovat. Jinak by se uživatel nemohl přihlásit.

Smazat – smaže uživatele, ale nikoliv jím vytvořený obsah, to musí učinit administrátor nebo uživatel před smazáním.

Role – umožňuje administrátorovi změnit roli, kterou má uživatel na stránkách.

Tabulka 4.5: Uživatelská práva modulu `User`

Akce	**Registrovaný**	**Autor**	**Administrátor**
Přidat	Ne	Ne	Ano
Upravit	Vlastní	Vlastní	Ano
Smazat, zabanovat/odbanovat, aktivovat, změnit práva	Ne	Ne	Ano

Comment

Jedná se o modul dostupný pouze pro administraci, ve které umožňuje spravovat komentáře. Komentáře v uživatelském prostředí jsou tvořeny pomocí komponenty `Comments`, jež umožňuje jejich vkládání a zobrazení. Zasahovat do komentářů může administrátor a autor, který je omezen na komentáře u jeho komiksů.

Obrázek 4.11: Administrace komentářů

Upravit – umožní upravit komentář.

Smazat – smaže komentář.

Options

V administraci má tento modul funkci správy nastavení. Zásadní vlastností modulu je, že umožňuje měnit název stránky dostupných skrz uživatelské prostředí a také možnost obnovovat dostupné jazyky. Do tohoto modulu má přístup pouze administrátor.

Rss

Tento modul v uživatelském prostředí vytváří RSS kanály za pomoci komponenty, která byla pro tento účel vytvořena. Každá série má svůj kanál a navíc existuje kanál pro všechny série.

Sign

Tento modul slouží v uživatelském prostředí čistě k přihlašování a ověřování uživatelů. Pokud sem byl uživatel přesunut z nějaké stránky z důvodu třeba dlouhé nečinnosti, tak bude přesunut po přihlášení zpět na stránku, kterou se snažil navštívit, pokud má práva k jejímu přístupu.

Kód 4.2: Ukázka vygenerovaného RSS kódu

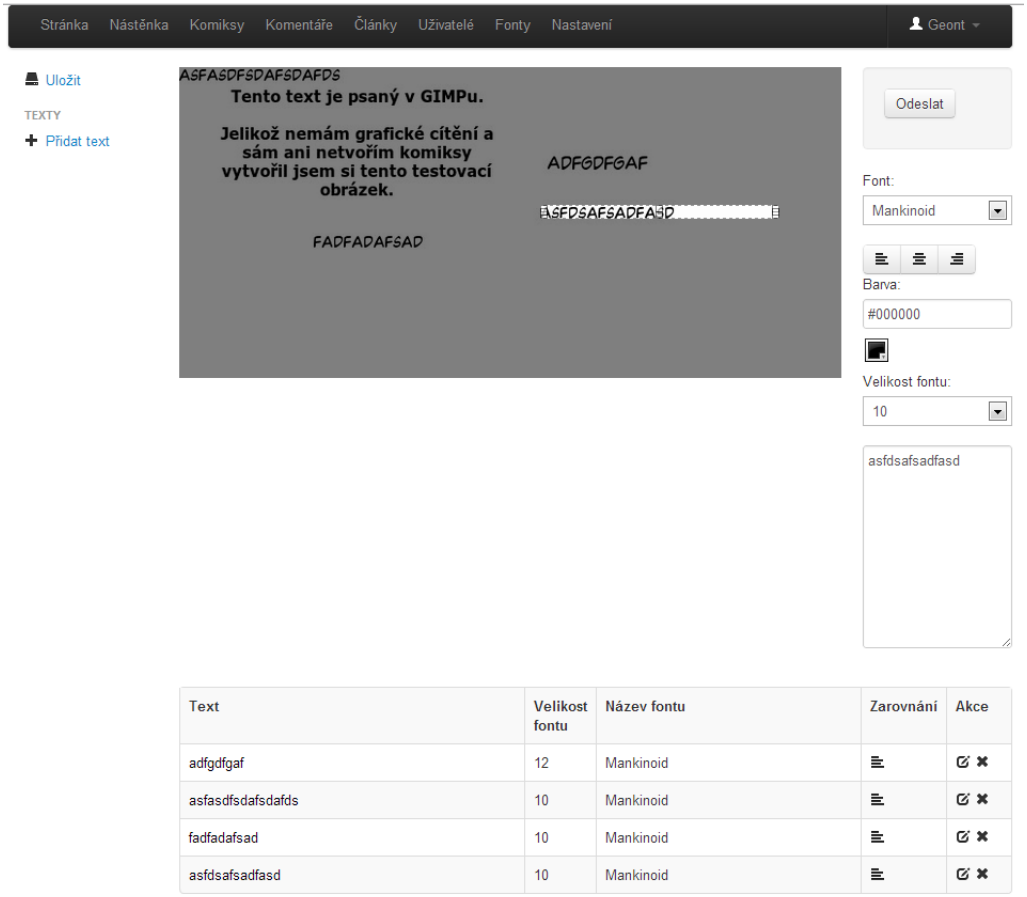
```
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <title>llun</title>
    <description>News for comic llun</description>
    <link>/cs/series/show/llun</link>
    <lastBuildDate>2012-06-11 08:45:56</lastBuildDate>
    <item>
          <pubDate>2012-05-02 00:00:00</pubDate>
          <title>Druhe setkani</title>
          <link>/cs/series/show/llun/druhe-setkani</link>
          <guid>/cs/series/show/llun/druhe-setkani</guid>
    </item>
    <item>
          <pubDate>2012-02-01 00:00:00</pubDate>
          <title>Cast prvni</title>
          <link>/cs/series/show/llun/cast-prvni</link>
          <guid>/cs/series/show/llun/cast-prvni</guid>
    </item>
  </channel>
</rss>
```

Image

Modul má pouze administrativní prostředí, díky kterému umožňuje poměrně zajímavou funkci přidávání textu do obrázků. Editor je to velmi jednoduchý, neumožňuje nijak složité operace s textem. K dispozici jsou pouze fonty, které byly přidány skrze webové rozhraní.

Za editorem stojí kombinace jQuery pluginu `imgAreaSelect`, který slouží k výběru obdélníkové oblasti, kde se má nacházet text, a knihovny GD, která zpracovává zápis textů do obrázku. GD nemá nijak velké možnosti při práci s textem, nicméně i to málo se ukázalo jako dostačující. Podporován je pouze horizontální text, jelikož jsem došel k závěru, že při editaci komiksu není nutné mít podporu pro vertikální text. Tento názor podpírám tím, že jsem editor vytvářel za účelem doplňování textů dialogů. Algoritmus přidávání textů do obrázků využívá několik funkcí GD (`imagettfbbox` a `imagettftext`). Lze tedy použít pouze ttf fonty. Pomocí těchto funkcí na základě zadaných údajů vykresluje text do obrázku.

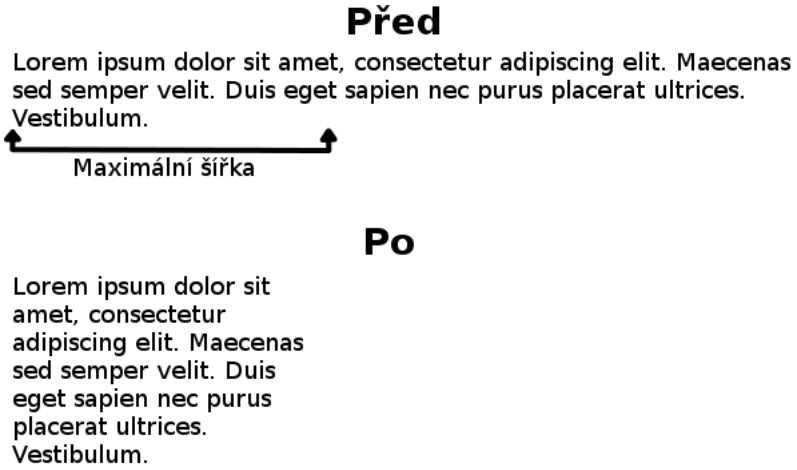
V databázi jsou uloženy informace, které slouží k vytváření textu. Konkrétně



Obrázek 4.12: Editace textů v obrázku

se jedná o velikost textu v bodech (points, pt), barvu v RGB, x a y pozici levého horního rohu, maximální šířku textu, zarovnání textu a font. Pokud chceme vykreslit či vygenerovat obrázek i s texty, provede se pro každý záznam v databázi sekvence kroků, kterou popisuji v následujícím textu.

Nejprve se rozdělí text na řádky podle znaku nového řádku, a tím je znak \n. Takto jsme získali prvotní řádky, které ale ještě musí projít další fází, kontrolou jejich šíře. V tomto momentě využíváme možnosti funkce `imagettfbbox`, která vrací souřadnice rohů obdélníku, který obsahuje text. Zajímají nás pouze horizontální souřadnice levého a pravého horního rohu. Jejich rozdíl nám dává šířku řádku, tu porovnáme s maximální šířkou. V případě, že přesahuje, tak se pokusí rozdělit řádek po slovech na více řádků, které podmínku maximální šířky splňují. Pokud nastane situace, kdy máme jedno nebo víc slov, které přesahují maximální šíři, tak dojde k přizpůsobení na jejich maximální šířku. Algoritmus se následně spustí znovu s touto novou šířkou. Tato hodnota se uloží do databáze.



Obrázek 4.13: Text se zadanou maximální šířkou před průchodem algoritmem

Další část se skládá z výpočtu šířek nově vzniklých řádků a celkové výšky textu. Jako poslední věc je procházení řádků textu, nastavení offsetu od levé strany, který je ovlivněn zarovnáním textu. Následuje výpočet vertikálního offsetu tj. mezer mezi řádky. Jeho přidání do celkové výšky textu a nakonec zápis samotného textu podle udaných parametrů.

Install

Modul Install slouží k instalaci aplikace. Instalace je prováděna ve třech krocích. První krok nás informuje o tom, že všechny tabulky, které se budou shodovat s názvy tabulek aplikace, budou nahrazeny těmi aplikačními. Zbývající kroky už probíhají s nově vygenerovanou databází, do které ukládají potřebné informace. Mezi tyto informace patří základní jazyk, název stránek a založení administrátorského účtu, který má nastaven jako svůj jazyk základní jazyk stránek.

4.3 Databáze

Databáze tvoří základ většiny dnešních webových aplikací, proto je důležité se zabývat její strukturou. Protože jsem rozhodl použít MySQL, musel jsem zvážit, jaké úložiště dat v něm použít. Pro všechny tabulky jsem použil úložiště dat MyISAM z důvodů větší podpory hostingů. Některé hostingy zdarma nepodporují InnoDB a tak by byl mařen záměr o co nejmenší problémy pro provoz v široké škále hostingů.

Vzhledem k vícejazyčnosti je v databázi použito kódování `utf8_unicode_ci`.

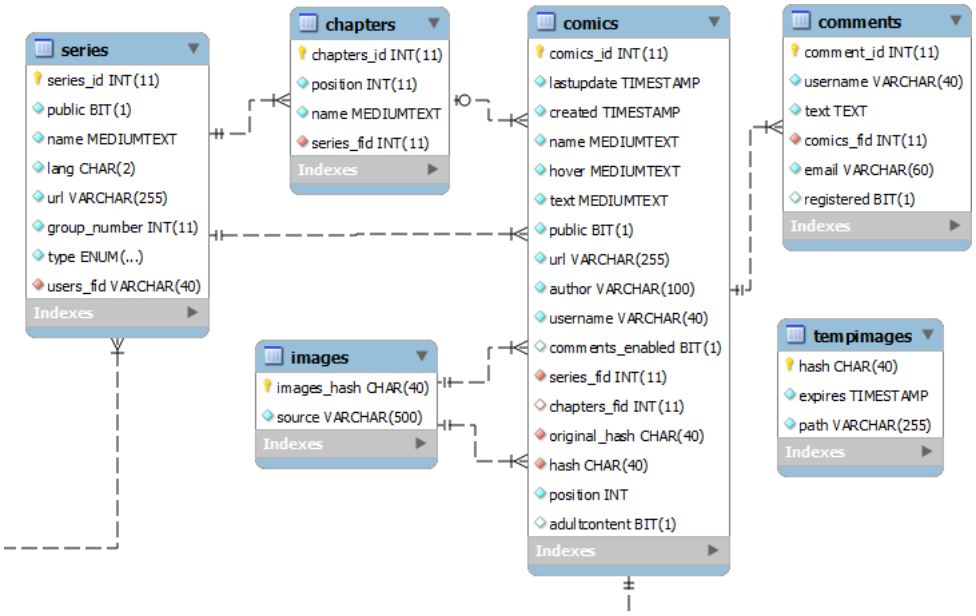
Komiksy

Komiksy jsou definovány pomocí čtveřice tabulek, `series`, `chapters`, `comics` a `images`. Každá z tabulek odpovídá obsahu jednotlivých částí. Série rozhodují o tom, jaký jazyk mají komiksy, které do ní patří. Série se dělí do skupin. Skupinou je myšleno seskupení sérií, ve kterém máme k dispozici různé jazykové verze.

Kapitoly nám dávají možnost, jak seskupovat v sérii jednotlivé komiksy do celků. Zásadní pro tabulku `chapters` je, že obsahuje atribut `position`, která vyjadřuje pozici v sérii. Spojení různých jazykových verzí probíhá skrze série.

`Images` obsahuje hash vytvořený z obsahu obrázku a cesty k němu. Za primární klíč jsem zvolil hash, jelikož ten by měl jednoznačně identifikovat soubor. V tabulce `comics` nalezneme dva hashe, jeden slouží k odkazování na aktuální obrázek a druhý, `original_hash`, na původní. Ve většině případů budou duplicitní, ale v momentě, kdy využijeme možnosti editovat texty, tak se při uložení změn do obrázku a vytvoření dalšího fyzického souboru, budou tyto hashe lišit.

V tabulce `comics` kromě již zmíněných atributů `hash` a `original_hash` nalezneme například cizí klíče na sérii a kapitolu, do které patří.



Obrázek 4.14: Tabulky spojené s komiksy a komentáři

V průběhu vývoje se databáze postupně měnila, zpočátku byla přítomna tabulka

`groups`, která měla sloužit ke spojování jazykových verzí sérií a komiksů. Po zralé úvaze jsem nakonec došel k závěru, že se tím stane obtížnější udržení konzistence databáze a přehledu o tom, co seskupuje, a bude tedy lepší tuto tabulku vynechat. Místo toho jsem zajistil seskupování pomocí atributu `group_number` nebo `position`.

Dočasné obrázky

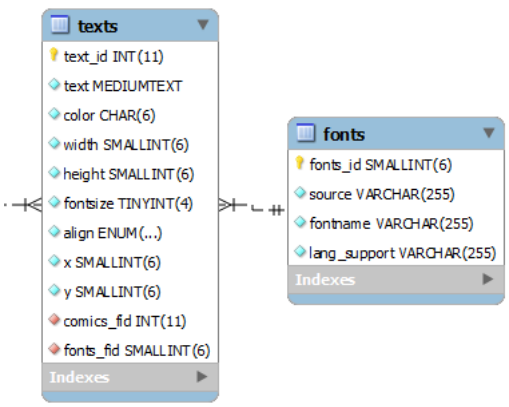
Tabulka `tempimages` slouží k ukládání informací o nahraných souborech, které nebyly ještě použity pro nějaký komiks. Jelikož se ukládá informace o tom, kdy mají soubory vypršet, je zajištěno, že nebudou soubory uloženy věčně. Kontrola platnosti probíhá pokaždé, když je nějaký nový obrázek nahrán. Pokud platnost obrázku vypršela bude smazán.

Komentáře

Komentáře jsou tvořeny pouze jednou tabulkou `comments`, která je propojena pomocí cizího klíče s komiksy. Obsahuje identifikátor, jméno toho, kdo komentář přidal, text zprávy, e-mail a cizí klíč na komiks.

Texty

Texty jsou reprezentovány dvojicí tabulek, `texts` a `fonts`. V první jsou všechny texty a obsahuje dva cizí klíče, a to pro komiks a font. Druhá tabulka obsahuje název fontu, cestu k souboru, kterou používá skript pro tvoření textů v obrázcích, a jazyky, které font podporuje.



Obrázek 4.15: Tabulky spojené s texty

Články

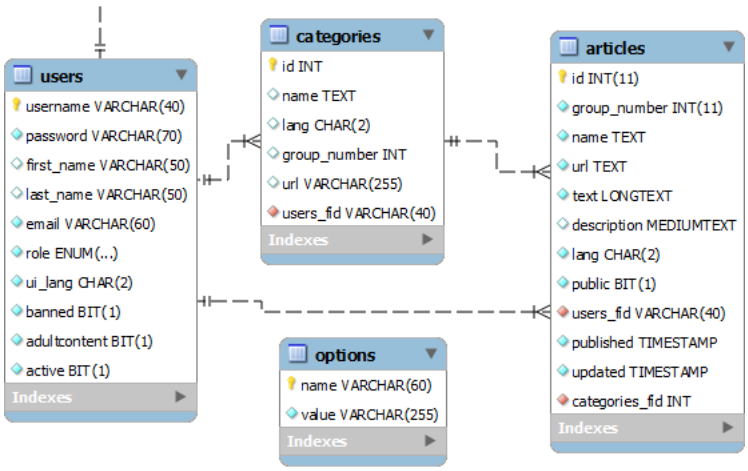
Články mají pouze dvě tabulky, `articles` a `categories`. Tabulka `articles` představuje články, které se stejně jako série seskupují do skupin jazykových verzí. Jediný cizí klíč se váže k uživateli, který článek vytvořil. Podobně jako série i články obsahují atribut `group_number`, který se stará o spojování různých jazykových variant téhož článku. Kapitoly mají pouze vlastní název a stejně jako série a články se seskupují do skupin podle stejného atributu.

Uživatelé

Uživatele představuje jedna tabulka, `users`, do které se o nich ukládají informace a jejich osobní nastavení.

Nastavení

Jednoduchá tabulka `options` na ukládání nastavení celé aplikace. Je tvořena pouze dvojicí atributy, názvem a hodnotou.



Obrázek 4.16: Tabulky articles, categories, users a options

4.4 Vícejazyčnost aplikace

Vzhledem k tomu, že jsem vytvářel aplikaci, která by zvládla pracovat jako vícejazyčná, využil jsem možnosti použít `NetteTranslator`. Ten je vytvořen pro práci v Nette frameworku, ve kterém implementuje rozhraní `ITranslator`.

Díky tomu jsem mohl využít lokalizační možnosti frameworku a zároveň jsem získal nástroj, který mi v lokalizaci pomohl. Lokalizační soubory jsou běžně používané gettextem, který je implementovaný v `NetteTranslatoru` takovým způsobem, že při změně v lokalizačních souborech nemusí dojít k restartu serveru, jak je to u implementace v PHP nutné.

Pro přidání dalšího jazyka musíme nejdříve vytvořit jazykový soubor, kde se bude nacházet překlad. Tento soubor lze vytvořit za pomoci `NetteTranslatoru`, ale to není moc vhodné, jelikož je pak nutné projít všechny stránky a zobrazit všechny oznámení. Druhým způsobem je vzít si například přiložený `PO` soubor a využít programu `Poedit`, který umožňuje vytváření a generování `MO` souborů, jež používá aplikace, a přeložit tak aplikaci do nového jazyka. Jako poslední krok je obnovení stavu jazyků v administraci v sekci `Nastavení`.

Součástí aplikace je i třída `LocaleInfo`, která nám poskytuje informace o různých jazycích použitelných pro lokalizaci aplikace.

Vzhledem k tomu, že se využívá i cizí kód (TinyMCE), tak je nutné zajistit i překlad těchto částí. TinyMCE má své překlady ve složce `/js/tinymce/langs/`.

Vícejazyčnost dynamických textů, jako jsou články a komiksy, je řešena pomocí databáze a v textu práce se jí zabývám v sekci týkající se databází.

4.4.1 Datum a čas

Prvotní nastavení formátování datumu a času společně s časovou zónou nalezneme v třídě `LocaleInfo`. Uživatel si tato nastavení může změnit v administraci v rámci modulu `Nastavení`. Nastavení jsou aplikována podle uživatelových preferencí (administrace) nebo jazykové mutaci stránek (uživatelská část). Všechna data ukládaná do databáze jsou uložena v zóně UTC kvůli konzistenci.

Závěr

V bakalářské práci se zabývám tvorbou jednoduše ovladatelného systému specializovaného na publikaci webových komiksů a/nebo stripů. V teoretické části popisuji základní technologie, které patří k tvorbě dynamických webových stránek. Výsledek praktické části je vidět na testovací webové stránce, kterou jsem vytvořil pomocí vytvořeného systému. Nalezneme ji na adrese `llun-test.g6.cz` (admin - přihlašovací jméno: geont a heslo: geont258).

Jako PHP framework jsem si zvolil Nette, se kterým se mi díky novinkám ve verzi 2 pracovalo dobře. Rozhodnutí rozdělit kód do modulů napomohlo k snadnější orientaci v kódu. Použitý framework pro JavaScript je jQuery, který má velké množství dostupných pluginů. Několik pluginů bylo využito pro potřeby administrace.

Systém umožňuje správu komiksů, kapitol a sérií. Série pak může mít více jazykových mutací, což nám umožní vydávat komiksy ve více jazycích. Využití knihovny GD pro přidávání textů do obrázků se ukázalo jako dostatečné.

S výslednou podobou aplikace jsem spokojený. Splňuje zadání a požadavky, které byly na ní kladeny v průběhu vývoje. Během vývoje došlo k několika problémům (např. špatné navržení modulů a databáze), ale ty se mi podařilo vyřešit. Vývoj aplikace ovšem nemusí být konečný, jelikož jsou věci, které by bylo možné přidat. Příkladem funkce, která by byla vhodná, je podpora addonů či pluginů podobná takové, jakou mají pokročilé CMS (např. Wordpress).

Literatura

[1] BERNARD, B. *Seriál MVC a další prezentační architektury* [online]. 2009. URL: <http://www.zdrojak.cz/serialy/mvc-a-dalsi-prezentacni-vzory/>.

[2] DANĚK, P. *Seriál Velký test PHP frameworků* [online]. 2008. URL: <http://www.root.cz/serialy/velky-test-php-frameworku/>.

[3] LONGMAN, A. W. *A history of HTML* [online]. 1998. URL: <http://www.w3.org/People/Raggett/book4/ch02.html>.

[4] Nette foundation. *Dokumentace Nette frameworku* [online]. 2012. URL: <http://doc.nette.org/cs/>.

[5] Oracle Corporation. *MySQL Differences from Standard SQL* [online]. 2012. URL: <http://dev.mysql.com/doc/refman/5.1/en/differences-from-ansi.html>.

[6] Q-Success. *Usage of content management systems for websites* [online]. 2012. URL: <http://w3techs.com/technologies/overview/content_management/all>.

[7] Q-Success. *Usage of JavaScript libraries for websites* [online]. 2012. URL: <http://w3techs.com/technologies/overview/javascript_library/all>.

[8] Q-Success. *Usage of server-side programming languages for websites* [online]. 2012. URL: <http://w3techs.com/technologies/overview/programming_language/all>.

[9] The jQuery Foundation. *Dokumentace jQuery frameworku* [online]. 2012. URL: <http://docs.jquery.com/Main_Page>.

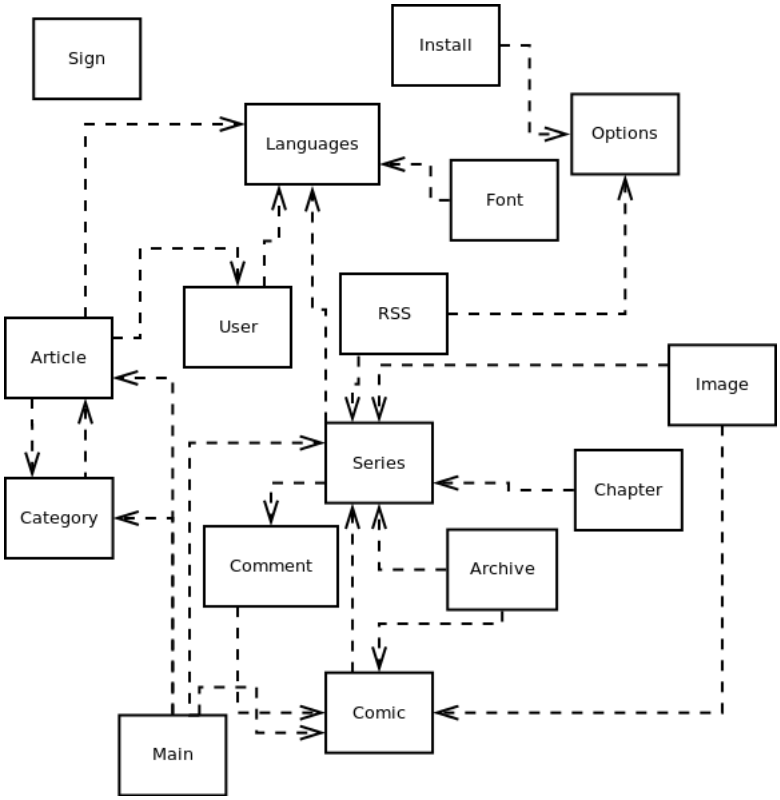
[10] ZAHARIEV, I. *C++ vs. Python vs. Perl vs. PHP performance benchmark* [online]. 2012. URL: <http://blog.famzah.net/2010/07/01/cpp-vs-python-vs-perl-vs-php-performance-benchmark/>.

Příloha A – Obsah CD

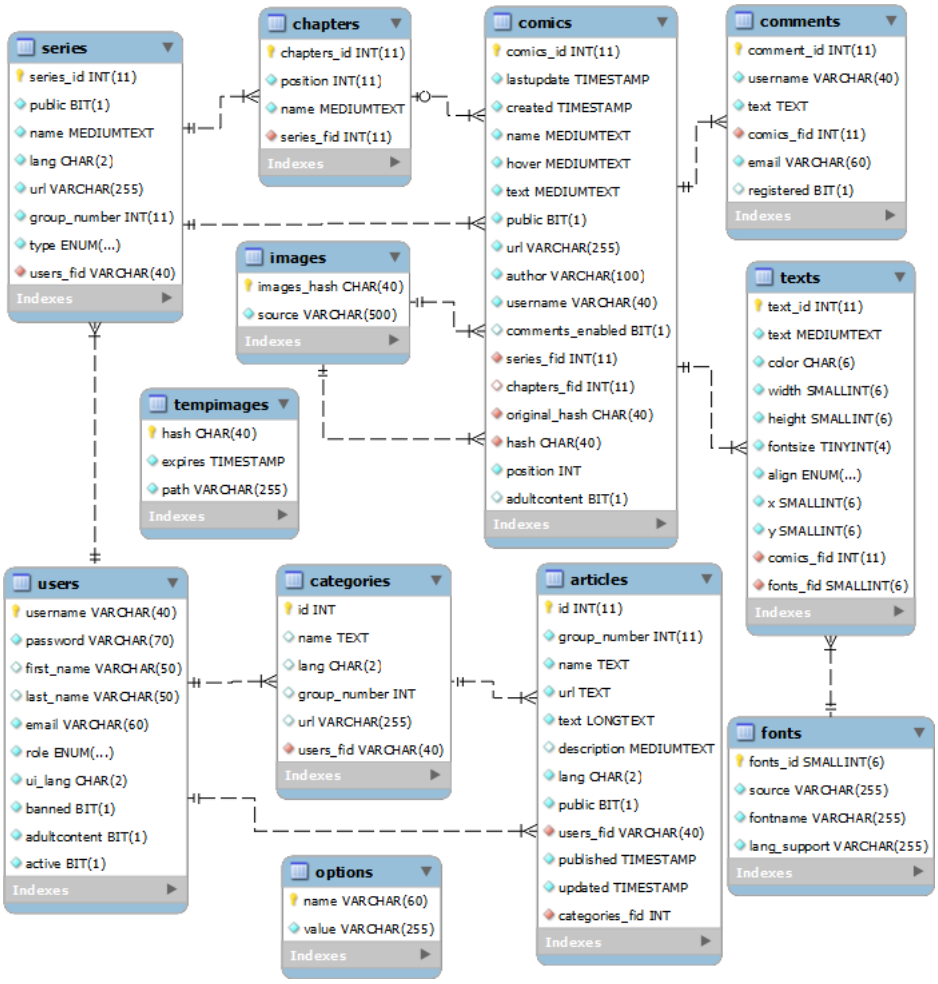
obsah CD

CD
- `comicPublish` ..... Publikační systém se základní šablonou
- `Databáze` ..... Soubory pro MySQL Workbench
- `Bakalářská práce` ..... Zdrojové kódy bakalářské práce
- `Text práce`

Příloha B – Diagram závislostí mezi moduly



Příloha C – EER diagram databáze



Příloha D – Postup instalace

Nejdříve musíme zajistit, aby složky temp a log měly nastaveny práva na `0766`. Poté musíme nastavit přístupové údaje k databázi. Soubor, který obsahuje nastavení databáze, se nachází v adresáři `/app/config` a nazývá se `database.neon`.

Kód 4.3: Ukázkový příklad obsahu souboru `database.neon`

```
database:
        host: localhost
        database: llun
        user: geont
        password: easypassword
```

Podpora databáze je zaručena pouze pro MySQL, ale dibi dokáže komunikovat i s jinými databázemi beze změny SQL. Nicméně u jiných databází není zaručena stoprocentní funkčnost. Pokud bychom chtěli změnit typ databáze, musel by se změnit řádek `driver` v souboru `config.neon`.

V momentě, kdy nastavíme databázi, můžeme pokračovat v instalaci. Při prvním spuštění stránek se objeví instalátor, který reprezentuje modul Install. Ten v několika krocích provede základní instalaci. Většinu údajů lze později změnit v administraci.