TECHNICKÁ UNIVERZITA V LIBERCI

FAKULTA MECHATRONIKY, INFORMATIKY A MEZIOBOROVÝCH STUDIÍ

Studijní program: B2646 - Informační technologie

Studijní obor: 1802R007 - Informační technologie

Recenzní server – WWW aplikace
Web application for customers reviews

Bakalářská práce

Autor: Jan Švàger

Vedoucí práce: Mgr. Jiří Vraný, Ph.D.

V Liberci 17. 5. 2013

Stránka se zadáním, nechat prázdné…

1. Seznamte se zásadami responsive designu WWW aplikací a problematikou nasazení databáze CouchDB pro WWW aplikaci.
2. Navrhněte www aplikaci pro víceuživatelský recenzní systém.
3. Na základě návrhu tento systém vytvořte. Aplikace by měla být dobře ovladatelná na různých typech zařízení od mobilu po desktop PC.

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.

. . . .

Jan Švágr

V Liberci 17. 5. 2013

Abstrakt

Práce se zabývá vytvořením webové aplikace jako víceuživatelského serveru, který odráží reálné zkušenosti uživatelů s produkty. Aplikace se zaměřuje především na optimalizaci pro mobilní zařízení za využití responsive designu, použití nerelační dokumentové databáze CouchDB a na strukturu aplikace ve skriptovacím jazyce PHP s využitím Nette Framework.

Klíčová slova: Responsive Design, CouchDB, PHP, Nette Framework

Abstract

This work deals with creating a web application like multi-user server that reflects the real user experience with the products. The application focuses on optimizing for mobile devices using responsive design, the use of non-relational document database CouchDB and the structure of the application in a scripting language PHP with using the Nette Framework.

Keywords: Responsive Design, CouchDB, PHP, Nette Framework

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Mgr. Jiřímu Vranému, Ph.D. za jeho spolupráci, trpělivost a cenné rady. Dále bych chtěl poděkovat své rodině, která mě po celou dobu studia podporovala. Děkuji také dobrovolníkům, kteří se podíleli na testování aplikace.

Obsah

1 Úvod

Cílem této práce je poskytnout ucelený popis, který odráží reálný vývoj víceuživatelského recenzního serveru jako webové aplikace. Ta je vytvořena pomocí skriptovacího jazyku PHP s využitím Nette Framework. Jako datové úložiště využívá nerelační dokumentovou databázi CouchDB a je plně optimalizována pro zobrazení a interakci na mobilních zařízeních.

Práce se nejdříve zaměřuje na základní pojmy a použité technologie, mezi něž patří především Nette Framework a CouchDB. Následuje krátká příprava práce, která popisuje současný stav recenzí na internetu a popisuje podobné existující aplikace. Zmíněny jsou také základní cíle, které by měla aplikace splňovat. Poté se již dostává na řadu návrh a implementace aplikace samotné, kde je dbán důraz na základní strukturu a implementaci klíčových částí, na kterých je aplikace postavena. Předposlední kapitola se zaměřuje na optimalizaci pro mobilní zařízení, která vychází z metody Mobile first a Responsive designu. V poslední části jsou znázorněny výsledky práce v podobě uživatelského rozhraní.

2 Základní pojmy a použité technologie

V této části jsou vysvětleny základní teoretické pojmy a principy týkající se této práce, na které budou pozdější kapitoly dále navazovat. První část obsahuje stručný popis k použitému jazyku PHP. V další části je popsána architektura použitého Nette Framework. Poté jsou vysvětleny důležité vlastnosti a pojmy ohledně použité databáze CouchDB.

2.1 PHP

PHP (Hypertext Preprocessor) je skriptovací programovací jazyk určený především pro programování dynamických internetových stránek a webových aplikací. Skripty jsou prováděny na straně serveru – k uživateli je přenášen až výsledek jejich činnosti. PHP je nezávislý na platformě, rozdíly v různých operačních systémech se omezují na několik systémově závislých funkcí a skripty lze většinou mezi operačními systémy přenášet bez jakýchkoli úprav. Více informací lze nalézt v manuálu PHP [3].

2.2 Nette Framework

Pro tvorbu webové aplikace byl využit Nette Framework (dále jen Nette), jehož autorem je David Grudl [4]. Nette je postaven na architektuře Model-View-Controller (MVC). V této podkapitole bude tato architektura (včetně jejích částí) stručně popsána. Zmíněn je také šablonovací systém Latte.

2.2.1 Model-View-Controller

MVC je softwarová architektura, která vznikla z potřeby oddělit u aplikací s grafickým rozhraním kód obsluhy (controller) od kódu aplikační logiky (model) a od kódu zobrazujícího data (view). Tím aplikaci zpřehledňuje, usnadňuje budoucí vývoj a umožňuje testování jednotlivých částí zvlášť.

Model je datový a zejména funkční základ celé aplikace. Je v něm obsažena aplikační logika. Akce uživatele představuje akci modelu. Model si spravuje svůj vnitřní stav a ven nabízí pevně dané rozhraní. Voláním funkcí tohoto rozhraní můžeme zjišťovat či měnit jeho stav.

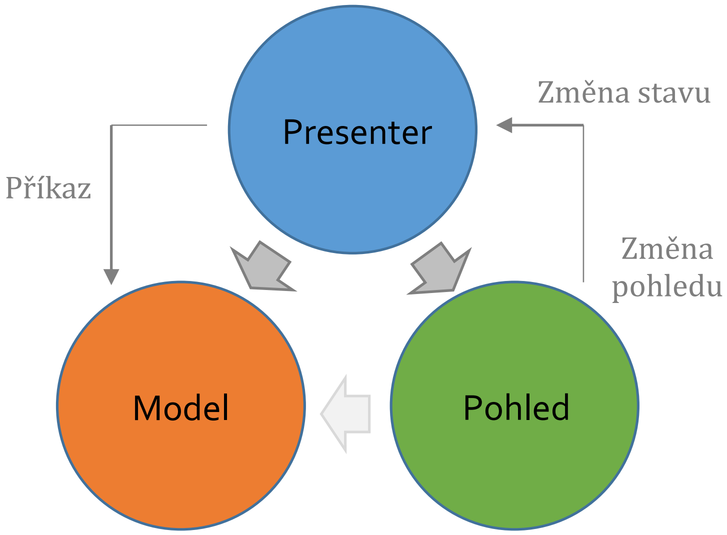
View, tedy pohled, je vrstva aplikace, která má na starost zobrazení výsledku požadavku. Obvykle používá šablonovací systém, díky němuž může snadno zobrazit data získaná z modelu.

Controller je řadič, který zpracovává požadavky uživatele, na jejich základě volá patřičnou aplikační logiku (tj. model), a poté požádá view o vykreslení dat.

2.2.2 Presenter

Presenter [5] v Nette má podobnou roli jako controller v MVC. Vybírá pohled (view) a předává mu model, nebo data z modelu. Udržuje stav persistentních proměnných. Především však zpracovává reakce uživatele. Ty se dají rozdělit na tři typy:

- změna pohledu
- změna stavu
- příkaz modelu



Obr. 1: Architektura Model–View–Controler v Nette Framework

2.2.3 Šablonovací systém Latte

U větších projektů je vhodné mít oddělenou aplikační vrstvu aplikace od té prezentační. Aby byla práce efektivní, mělo by být cílem, aby mohl grafik nebo kodér webu pracovat na projektu bez obavy, že poškodí aplikační kód, a naopak, aby grafik nemusel žádat o spolupráci programátora kvůli změnám, které se týkají prezentace dat. Tohoto oddělení lze dosáhnout využitím šablonovacího systému Latte, který je základní součástí Nette Framework.

2.3 CouchDB

Apache CouchDB (dále jen CouchDB) [1] je open source dokumentově orientovaný databázový systém napsaný v programovacím jazyku Erlang. Jeho tvůrcem je bývalý vývojář z IBM Damien Kratz.

CouchDB definují především její unikátní vlastnosti:

- Dokumentově orientovaná
- Restful HTTP
- Map/Reduce pohledy
- Distribuovaná
- Odolná proti pádu

2.3.1 Dokumentově orientovaná

CouchDB používá pro zápis dokumentů i výsledky dotazů notaci JavaScript Object Notation (dále jen JSON). JSON je jednoduchý a odlehčený formát pro výměnu dat. Data jsou ukládána **bez** pevně daného **schématu**, což umožňuje vytvářet volné struktury. To se velmi dobře hodí pro rychlý vývoj, nebo pokud mají data podobu reálných dokumentů.

Typickým příkladem z CouchDB: The Definitive Guide [1] je ukládání vizitek, kde každá vizitka obsahuje jiné informace, např. může obsahovat jedno nebo více adres či telefonních čísel. Tohoto lze v CouchDB na rozdíl od relačních databází docílit velmi jednoduše.

Každý dokument obsahuje svůj jedinečný identifikátor (dále pouze docid), který může CouchDB generovat, nebo může být použit vlastní. Generované docid zaručuje unikátnost a předchází konfliktům. CouchDB obsahuje také verzovací systém, který přiděluje dokumentům revizi při každé změně dokumentu. Vychází z principu neustálého přidávání dat – nikdy nemaže stará data. Díky tomuto mechanismu je velmi odolná proti pádům.

Příklad uložení dokumentu typu vizitky v CouchDB:

```
{
   "_id": "card/jan-svager",
   "_rev": "1-9ff54b49889ebbe65fb8be509d794992",
   "type": "card",
   "first_name": "Jan",
   "last_name": "Švágerˮ,
   "phones": {
       "home": "+420 777 888 999",
       "work": "+420 666 888 999",
       "mobile": "+420 555 666 999"
   },
   "addresses": {
       "work": {
           "street": "Studentská",
           "number": "2",
           "city": "Liberec",
           "country": "Czech Republic"
       }
   },
   "created_at": [ 2013, 5, 2, 13, 41, 33 ]
}
```

2.3.2 Restful HTTP

K datům v CouchDB se přistupuje pomocí architektury REST, která umožňuje operace CRUD pomocí standardních dotazů HTTP:

- **Create**: PUT *http://domain.tld:port**/db***
- **Read**: GET *http://domain.tld:port**/db/docid***
- **Update**: PUT *http://domain.tld:port**/db/docid***
- **Delete**: DELETE *http://domain.tld:port**/db/docid***

REST je architektura rozhraní navržená především pro distribuované prostředí. Chování většiny požadavků v CouchDB nezávisí na předchozích požadavcích. Odpovědi na jednotlivé požadavky jsou typicky opět ve formátu JSON.

Typickým příkladem použití dotazů HTTP je pomocí curl zavolání adresy serveru na určitém portu, čímž se zároveň ověří, že je CouchDB v provozu:

```
curl http://127.0.0.1:5984/
```

Odpověď ve formátu JSON pak vypadá následovně:

```
{"couchdb":"Welcome","version":"1.2.0"}
```

2.3.3 Map/Reduce

Pro zajištění struktury datům uloženým v CouchDB lze vytvořit, podobně jako u databází relačních, pohledy. Pohled se skládá z map a reduce funkce, kde reduce je volitelná.

Každý pohled lze vytvořit pomocí **Javascriptové** funkce, která může být libovolně složitá. Z důvodu nákladnosti takové operace nad rozsáhlou databází může CouchDB pohledy **indexovat** a tyto indexy obnovovat, jak jsou dokumenty přidávány, mazány a obnovovány. Toto je velmi mocný indexovací mechanismus udělující bezprecedentní kontrolu nad systémem v porovnání s většinou ostatních databázových systémů.

Map prohledává jednotlivé datové záznamy (v případě CouchDB se jedná o dokumenty) jeden po druhém. Výstupem funkce map je **seřazená** kolekce párů klíč / hodnota. To zjednodušuje získávání dat z určitého rozsahu klíčů, i když záznamů jsou tisíce či miliony. Jako klíč může být zvoleno i pole více klíčů. Každý řádek kromě klíče obsahuje také hodnotu. Pomocí reduce funkce lze takto setříděné řádky velmi snadno zredukovat a získat například součet všech hodnot či počet položek se stejným klíčem.

Příklad použití map funkce:

```
function(doc) {
	if(doc.type == "card"){
		emit(doc.type, 1);
	}
}
```

Funkce vrátí všechny dokumenty typu card s klíčem card a s hodnotou 1. Na výsledek této map funkce lze použít jednoduchou reduce funkci:

```
function(keys, values) {
       return sum(values);
}
```

Výsledkem je JSON, který jako hodnotu vrací celkový počet dokumentů typu card:

```
{
   "rows": [ {"key": "card", "value": 20} ]
}
```

2.3.4 Distribuovaná

CouchDB se velmi dobře škáluje, a to různými směry. Databázové jádro je napsané v Erlangu, takže není problém je provozovat na velkých serverech i na embedded systémech. Důležitou součástí je **on-line** i **off-line replikace**. CouchDB lze provozovat v clusteru, ale i jako úložiště osobních dat na několika různých zařízeních (stolní počítač, notebook, telefon, PDA, webový server), která se spolu synchronizují.

2.3.5 Odolná proti pádu

Jak již bylo zmíněno, CouchDB je také velmi odolná proti pádu. Tato vlastnost vychází ze způsobu ukládání dat, kdy jsou data **pouze přidávána**. Nevýhodou toho přístupu je stále rostoucí velikost úložiště, kterou lze částečně omezit voláním garbage collectoru. CouchDB však staví na myšlence, že diskový prostor je oproti výpočetnímu výkonu mnohem levnější, a proto není problém tento prostor snadno rozšířit.

2.4 Mobile first

Mobile first je jedním ze způsobů webdesignu, jehož návrh začíná pro mobilní zařízení. Jakmile je tento návrh hotov, může se přejít k návrhu pro PC. Mobile first umožňuje:

- připravit se na explozivní růst a nové příležitosti ohledně mobilních zařízení,
- zaměřit se pouze na části, které jsou skutečně důležité:
    - zjednodušení obsahu,
    - určení priorit,
    - použitelnost.

Důležitým faktorem je, že web jako takový již neexistuje pouze na stolním PC, jako tomu mohlo být v letech minulých. Každým dnem se aktivuje a přibývá mnoho mobilních zařízení, díky nimž lze již velmi snadno přistupovat k webovým stránkám takřka odkudkoliv. Web se rozrostl a nadále rozrůstá do nepřeberného množství různých zařízení. Mezi ně patří dnes již typicky například mobilní telefony,

Obr. 2: Univerzální web.
Převzato z bradfrostweb.com [7]

tablety, elektronické čtečky či televize. Tento trend bude pravděpodobně stále pokračovat.

Zobrazení webové stránky optimalizované pouze na jeden druh zařízení tak pro uživatele přináší několik možných problémů:

- stránka se nemusí zobrazovat naprosto korektně
- ovládání uživatelského prostředí není přizpůsobené zařízení

Mobilní web je tedy především univerzální, přístupný z jakéhokoliv zařízení a měl by umožňovat získat jednotné informace (Obr. 2). Jedním z cílů této webové aplikace je právě její mobilní univerzálnost.

3 Příprava práce

Kapitola popisuje současný stav situace před vývojem recenzního serveru, zmiňuje již existující podobná řešení a také základní cíle, které byly pro aplikaci vytyčeny. Aplikace se jmenuje '*zkusil*' a je dostupná z url: *http://zkusil.cz*.

3.1 Současný stav

Většina recenzí produktů je v současné době umístěna především na odborných stránkách nebo na jiných tematicky zaměřených webech. K takovému webu se uživatel musí nejdříve dopátrat, a to nejčastěji přes internetový vyhledávač. Výsledky vyhledávání navíc nemusí odpovídat požadavkům uživatele. To pro něj přináší jen další ztracený čas. Samotné recenze jsou na mnoha webech totožné nebo velmi podobné – přidaná hodnota spočívá především v porovnávání technických vlastností produktu.

3.2 Podobné existující aplikace

Částečně přijatelná jsou existující řešení velkých internetových portálů, mezi něž patří především alza.cz a heureka.cz. Portály uvádějí u každého produktu krátké hodnocení nebo recenzi uživatelů. Ty jsou většinou doplněny klady a zápory daného produktu.

Tato řešení mají několik výhod:

- recenze jsou populární
  - uživatelé je sledují
  - uživatelé je tvoří
- recenze jsou krátké, informativní

A nevýhod:

- recenze často obsahují pouze klady a zápory
- odráží především krátkodobou zkušenost s produktem nebo závady
- pouze alza.cz poskytuje i mobilní přístup k recenzím

3.3 Záměr práce

Tato webová aplikace má za úkol vytvořit komunitní recenzní server, který odráží především reálné zkušenosti uživatelů s produkty. Uživatelé by měli tyto zkušenosti detailně popsat a uvést klady i zápory. Taková recenze bude ohodnocena ostatními uživateli, a tak může být jednoduše označena za nejpřínosnější, čímž ulehčí výběr ostatním. Obsah serveru bude tvořen především samotnými uživateli a měl by obsahovat různé druhy produktů, které budou snadno dohledatelné. Dalším důležitým aspektem je optimalizace pro různé druhy zařízení od mobilního telefonu po desktop PC, čímž bude umožněn přístup ještě více uživatelům.

3.4 Základní cíle aplikace

Před vývojem aplikace byly vytyčeny základní cíle, které by aplikace měla splňovat:

- Bezpečnost
- Dostupnost
- Použitelnost
- Datová nenáročnost
- Robustnost
- Rozšiřitelnost
- Srozumitelnost

Jednotlivé cíle jsou popsány v následujících podkapitolách.

3.4.1 Bezpečnost

Nejzákladnějším prvkem každé webové aplikace by měla být právě bezpečnost. Aplikace bude psána převážně v jazyce PHP, kde je většina běžných bezpečnostních chyb již známa, a tak by neměly být v žádném případě opomenuty. K tomuto účelu bude využit Nette Framework, který eliminuje výskyt bezpečnostních děr a jejich zneužití, jako je např. XSS, CSRF, session hijacking, session fixation atd.

3.4.2 Dostupnost

Aplikace by měla být snadno dostupná na jakémkoliv zařízení od mobilního telefonu po desktop PC. K tomuto účelu budou využity tyto metody webdesignu:

- Fluid layout
- Responsive design
- Mobile first

Použití těchto metod umožňuje také snadnou přizpůsobitelnost, která vychází z využití relativních jednotek. Úpravy webu, které nemění celkovou strukturu, jsou poté otázkou změny několika CSS vlastností.

3.4.3 Použitelnost

Dalším důležitým kritériem je jednoduchá použitelnost celého webu, která bude vycházet z návrhu Mobile First. Ten umožňuje zaměřit se pouze na prvky stránky, které jsou skutečně důležité. Takto by měla být vytvořena velmi jednoduchá navigace a pouze minimální množství dalších ovládacích prvků. Hlavním cílem je nerušit uživatele zbytečnými informacemi a zaměřit se především na obsah.

3.4.4 Datová nenáročnost

Velmi důležitým prvkem při optimalizaci pro mobilní zařízení je datová nenáročnost. Většina stávajících datových tarifů má omezený limit. Proto je důležité, aby aplikace přenášela pouze nezbytný objem dat. K tomuto účelu se skvěle hodí využití databáze CouchDB, která přenáší data přes HTTP protokol pomocí formátu JSON. S CouchDB lze také velmi snadno vytvářet cache. Dále budou všechny ikony celého webu umístěny v jediném souboru, čímž je značně omezen počet dotazů na server.

3.4.5 Robustnost

Aplikace bude velmi dobře odolná proti pádu. Tohoto cíle bude docíleno využitím databáze CouchDB. Ta data pouze přidává – stará data nemaže, ani nemění.

V případě kolize se dokument pouze neuloží a proces se může opakovat bez jakéhokoliv pádu.

3.4.6 Rozšiřitelnost

Vývoj aplikace bude zaměřen s ohledem na budoucí rozšiřitelnost. Aplikační část by měla být modulární a neměla by klást překážky při budoucím rozšiřování. Použití CouchDB je v tomto ohledu velmi vhodné, jelikož je velmi jednoduché CouchDB škálovat – umožňuje rozložit zátěž na více serverů a snadno mezi nimi replikovat.

3.4.7 Srozumitelnost

Aplikace bude dbát na srozumitelnost kódu. Jednotlivé metody a proměnné budou udávat svůj účel vhodným názvem. Funkčnost každé třídy a metody bude popsána pomocí zavináčových anotací.

4 Návrh a implementace

Tato kapitola zmiňuje návrh aplikace pro uživatelské rozhraní, které je optimalizováno pro mobilní telefony a zařízení s větší obrazovou plochou. Dále je rozebrána samotná struktura navržené aplikace a nakonec implementace jejích klíčových částí.

4.1 Uživatelské rozhraní

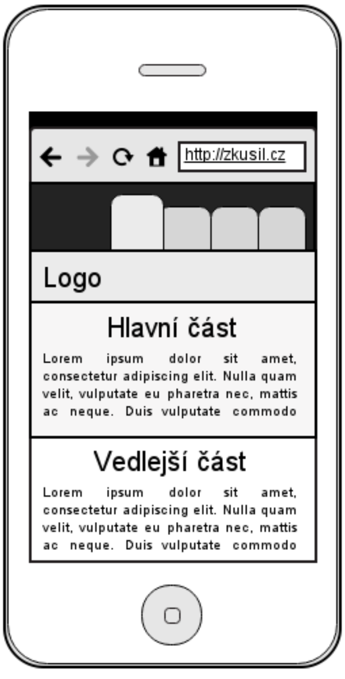
V této části je popsán návrh a rozložení webové aplikace, která se skládá z několika základních částí:

- navigační menu
- hlavní lišta
- hlavní část
- vedlejší část

Z těchto částí je seskládán výsledný návrh, který je optimalizován pro jednotlivá zařízení.

4.1.1 Mobilní telefony

Podle metody Mobile first započal návrh u mobilních telefonů. Je zde velmi omezený prostor, proto je důležité zaměřit se pouze na hlavní priority. Mezi ně patří především navigační menu a obsahová část.



Obr. 3: Návrh pro mobilní telefony

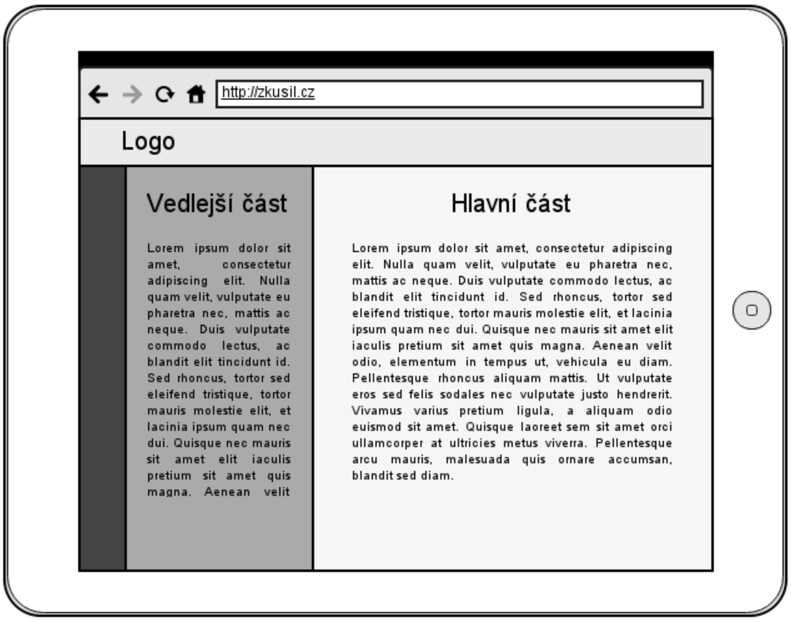
Pro mobilní telefony bylo navrženo menu v podobě záložek. Ty jsou úzce spojené s hlavní lištou. V základu se na hlavní liště zobrazuje pouze logo. Pokud se uživatel přihlásí, zobrazí se také jeho jméno a profilová fotografie. Pokud uživatel klepne na jinou záložku, zobrazí se pro ní dané položky menu. Aktivní záložka je zobrazena stejnou barvou jako hlavní lišta a nepřerušuje ji žádná linka, jako tomu je na Obr. 3. Ostatní záložky jsou zobrazeny barvou tmavší, čímž je znázorněna jejich neaktivita.

Obsahová část aplikace je rozdělena na hlavní a vedlejší část. Ty jsou pro mobilní telefony zobrazovány pod sebou.

4.1.2 Tablety a PC

Tablety a stolní počítače se liší především velikostí obrazovky, na které je aplikaci možné zobrazit. Návrh pro mobilní telefony v tomto případě již není optimální, a to především také kvůli typicky širokoúhlému zobrazení.

Aplikace byla pro tyto zařízení navržena tak, že hlavní a vedlejší části jsou nyní umístěny vedle sebe a mohou se libovolně prohazovat či rozšiřovat podle potřeby stránky (Obr. 4). Navigační menu je nyní umístěno na kraji stránky úplně vlevo.



Obr. 4: Návrh pro tablety

4.2 Struktura aplikace

V této části je popsána adresářová, webová a databázová struktura aplikace. Popsány jsou také databázové modely, které zpracovávají a předávají data z databáze do aplikace.

4.2.1 Adresářová struktura

Tato část popisuje adresářovou strukturu aplikační části, která se skládá z následujících položek:

- **/app** – aplikační část
  - **/AdminModule** – administrační část
    - **/presenters** – kontrolery
    - **/templates** – šablony
  - **/FrontModule** – uživatelská část
    - **/presenters** – kontrolery
    - **/templates** – šablony
  - **/components** – komponenty
  - **/config** – konfigurační soubor
  - **/forms** – formuláře
  - **/model** – datová a funkční vrstva aplikace
  - **/presenters** – základní kontrolery
- **/css** – kaskádové styly
- **/images** – obrázky
- **/js** – javascript
- **/libs** – knihovny
- **/log** – logování chyb
- **/temp** – dočasné soubory, cache

Jedná se o výchozí adresářovou strukturu Nette [4], která je upravena pro použití modulů. Je velmi důležité, aby adresář app nebyl veřejně přístupný, jelikož obsahuje veškerý aplikační kód a především konfigurační soubor, který obsahuje důležitá hesla a nastavení.

4.2.2 Webová struktura

Uživatelskou část tvoří presentery, které tvoří základní hierarchii webové aplikace. Ta odpovídá struktuře URL v adresním řádku. Mezi použité presentery a jejich akce patří:

- **/Article** – zobrazuje články
  - /all – výpis všech článků
  - /read – výpis konkrétního článku
- **/Auth** – obsluhuje autentizaci a autorizaci
  - /login – přihlášení
  - /register – registrace
  - /logout – odhlášení
  - /confirm – ověření registrace
- **/Catalog** – zobrazuje katalog produktů
- **/Create** – obsluhuje vytváření
  - /article – články
  - /review – recenze
- **/Error** – obsluhuje chybové stavy
  - 404
  - 405
  - 410
  - 4xx
  - 500
- **/Faq** – zobrazuje často kladené dotazy, obsluhuje jejich zadávání
- **/Homepage** – zobrazuje úvodní stránku
- **/Member** – obsluhuje uživatelskou část
  - all – statistiky všech uživatelů
  - profil – profil jediného uživatele
  - settings – nastavení uživatele
- **/Product**
- **/Search**

Pokud u presenteru akce není uvedena, znamená to, že obsahuje pouze jedinou akci s názvem default, která se v URL neuvádí, pokud není uveden žádný atribut. Samotný presenter tak zastupuje funkci akce.

4.2.3 Databázová struktura

Dokumenty uložené v CouchDB nevyžadují žádné schéma. Pro účely aplikace je však použití určitého schématu nutné. Jednotlivé dokumenty jsou rozděleny podle typu dat, která reprezentují. Každý typ dokumentu obsahuje základní pohled *list* – pro získání všech dokumentů daného typu. Dále také několik dalších pohledů pro získání seřazených či jinak strukturovaných dat. Pohledy jsou získávány pomocí Map/Reduce funkce v CouchDB.

Seznam pohledů použitých v aplikaci:

- **Article** – článek
    - **list** – získá všechny články
    - **created_at** – získá všechny články seřazené podle data vytvoření
    - **post** – získá samotný článek a jeho autora
- **Catalog** – strom položek pro určitou kategorii produktů
    - **list** – získá stromy všech kategorií
- **Comment** – komentář produktu nebo článku
    - **list** – získá všechny komentáře
- **Description** – technické údaje o produktu
    - **list** – získá všechny popisy produktů
- **Faq** – často kladená otázka
    - **list** – získá všechny často kladené otázky
    - **site** – získá všechny často kladené otázky portálu
    - **user** – získá všechny často kladené otázky od uživatelů
- **Review** – uživatelská recenze produktu
    - **list** – získá všechny recenze
    - **review** – získá jedinou recenzi + autora

- **review_best** – získá recenzi s nejlepším hodnocením + autora recenze
- **review_max_plus** – získá recenzi s nejvíce kladnými ohlasy + autora recenze
- **reviews_by_date** – získá všechny recenze pro daný produkt seřazené podle data + všechny autory recenzí
- **reviews_by_name** – získá všechny recenze pro daný produkt seřazené podle jména + všechny autory
- **reviews_by_exps** – získá všechny recenze pro daný produkt seřazené podle hodnocení + všechny autory recenzí

- **User** – účet uživatele
  - **list** – získá všechny uživatele
  - **exps** – získá všechny uživatele seřazené podle množství zkušenostních bodů
  - **registration** – získá všechny uživatele seřazené podle data registrace
  - **username** – získá všechny uživatele seřazené podle uživatelského jména
  - **name** – získá všechny uživatele seřazené podle jména
  - **fb_id** – získá všechny uživatele s Facebook účtem

4.2.4 Výchozí rozložení šablony

Základní struktura webu, která se odvíjí od návrhu, je rozdělena na hlavní a vedlejší část. Ta je strukturována především v šabloně, která vykresluje výsledný HTML kód. Pro tyto hlavní části jsou použity bloky *#mainbar* a *#sidebar*. Díky nim lze v šablonách pro specifickou stránku snadno určit, jaký obsah se má umístit do hlavní a vedlejší části. Výchozí šablona je určena především pro části stránky, které se opakují v rámci celé webové aplikace.

Zjednodušené rozložení výchozí šablony:

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <meta name="robots" content="{$robots}" n:ifset="$robots">
      <title>{block title|striptags}Hlavní stránka{/block}|zkusil.cz</title>
      <link rel="shortcut icon" href="{$basePath}/images/favicon.png">
      {block head}{/block}
   </head>
   <body>
      <div id="wrapper">
         {include 'header.latte'}
         <section class="content {$presenter->name}">
            <div id="mainbar">
               <div class="wrap">
                  {include #mainbar}
               </div>
            </div>
            <aside id="sidebar">
               <div class="wrap">
                  {include #sidebar}
               </div>
            </aside>
         </section>
         {include 'footer.latte'}
      </div>
   </body>
</html>
```

Ve zjednodušené šabloně jsou znázorněny především důležité části. Jsou vynechány části, které linkují nebo používají Javascript či CSS. Blok *title* umožňuje definovat dynamický titulek stránky. Zde je nastaven se základní hodnotou, která se zobrazí, pokud nebude uveden jiný specifický titulek. Blok *head* je určen pro vkládání skriptů či css stylů, které jsou specifické pro danou stránku. Dále je v těle stránky vložena nejdříve hlavička stránky, její hlavní sekce *#mainbar*, *#sidebar* a nakonec také patička. Klíčové slovo *include* zaručuje vložení daného bloku nebo šablony do stránky.

4.3 Implementace klíčových částí aplikace

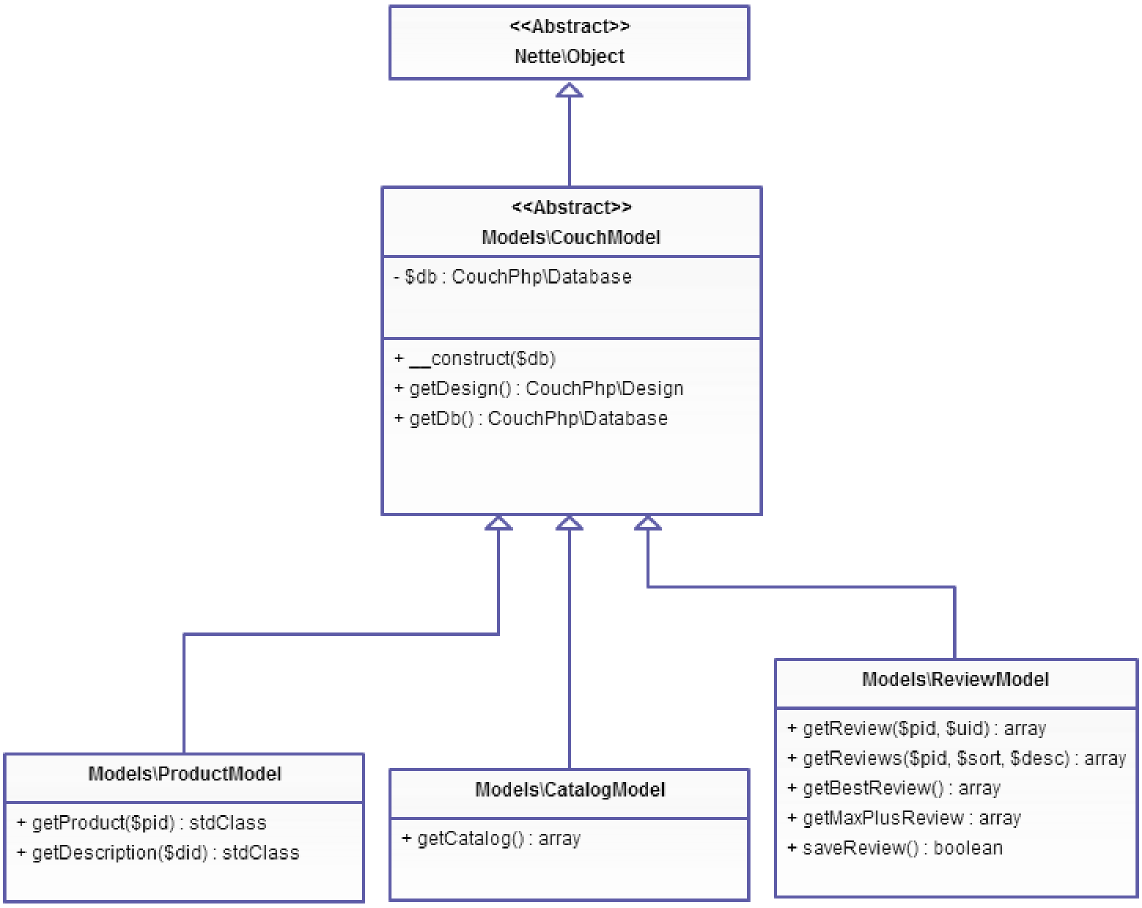
V této sekci je rozebrána implementace důležitých součástí aplikace. V první části je popsán způsob, jakým se získávají data v aplikaci pomocí modelů. V další části je popsán způsob agregace takto získaných dat. Poté je rozebrán seznam pro řízení práv a přístupů, který tvoří základní bezpečností složku aplikace.

4.3.1 Databázové modely

Pomocí pohledů lze data z CouchDB efektivně získávat. Pro získání dat v aplikaci byly navrženy a implementovány databázové modely, které odpovídají typům dokumentů v databázi. V modelu jsou data nejčastěji získávána právě díky pohledům nebo samostatně podle id dokumentu. Aplikace obsahuje následující databázové modely:

- **CouchModel**
- ArticleModel
- CatalogModel
- FaqModel
- ProductModel
- ReviewModel
- UserModel

Všechny použité databázové modely jsou potomci základní abstraktní třídy CouchModel (viz. Obr. 5), která obstarává veškeré závislosti, které jsou předávány automaticky v konfiguračním souboru. Třída získá připojení k databázi a automaticky nastaví pohledy podle názvu třídy budoucího potomka.



Obr. 5: Ukázkový UML diagram databázových modelů

Tímto způsobem lze velmi jednoduše, efektivně a modulárně vytvářet nové modely a zaměřit se čistě na zpracování pohledů. Jednotlivé modely se předávají do presenterů pomocí inject metod. Ty předávají závislosti automaticky a to ve chvíli, kdy jsou skutečně potřeba.

4.3.2 Agregace databázových dat

Pokud je potřeba získat data z CouchDB s určitou strukturou, lze toho docílit pomocí pohledů. V případech, kdy je potřeba získat agregovaná data z více typů dokumentů, není řešení zcela evidentní. Důležitým faktorem je v požadovaném dokumentu uchovávat referenční identifikátory na ostatní dokumenty, které jsou při získávání dat také potřeba. Identifikátory se poté využijí při budování map funkce.

Pro získání všech závislých dokumentů obsahuje CouchDB specifickou vlastnost, která se nazývá **linkování**. Při znalosti identifikátoru tak lze nalinkovat jakýkoliv dokument. Identifikátor se uvádí jako hodnota objektu s klíčem *_id*. Pro získání dat je však potřeba požadavek volat s parametrem *include_docs=true*. V opačném případě se při požadavku získá pouze linkovaný identifikátor.

Praktickým příkladem z aplikace je získání dat pro samostatnou recenzi. Zde je potřeba získat dokument, který uchovává data o samostatné recenzi a dále dokument, který uchovává data o autorovi recenze. Map funkce při použití nalinkování vypadá následovně:

```
function(doc) {
	if(doc.type == 'review'){
		emit([doc.pid, doc.uid, 0], null)
		emit([doc.pid, doc.uid, 1], {_id: doc.uid})
	}
}
```

Jak je z výše popsané map funkce patrné, hodnoty se získávají pouze pro dokumenty typu *review.* Jako klíč emit funkce je použito pole klíčů, kde *pid* je identifikátor produktu a *uid* je identifikátor uživatele. Jako poslední klíč je uvedeno číslo, které značí, jestli se jedná o data recenze nebo uživatele. V tomto případě číslo 0 značí samotnou recenzi a číslo 1 data uživatele. Nakonec je jako hodnota funkce emit použit nalinkovaný identifikátor, kterým je v tomto případě identifikátor dokumentu uživatele.

Poté již stačí zavolat výsledný HTTP požadavek s atributem *include_docs=true* následujícím způsobem:

```
../_view/review?startkey=["lg-nexus-4","user-svagis"]
&endkey=["lg-nexus-4","user-svagis",{}]&include_docs=true
```

Výsledkem je JSON jako pole o dvou prvcích, kde první prvek je dokument s recenzí a druhým prvkem je dokument s uživatelem recenze. Nalinkovaná data jsou uložena pod položkou *doc*:

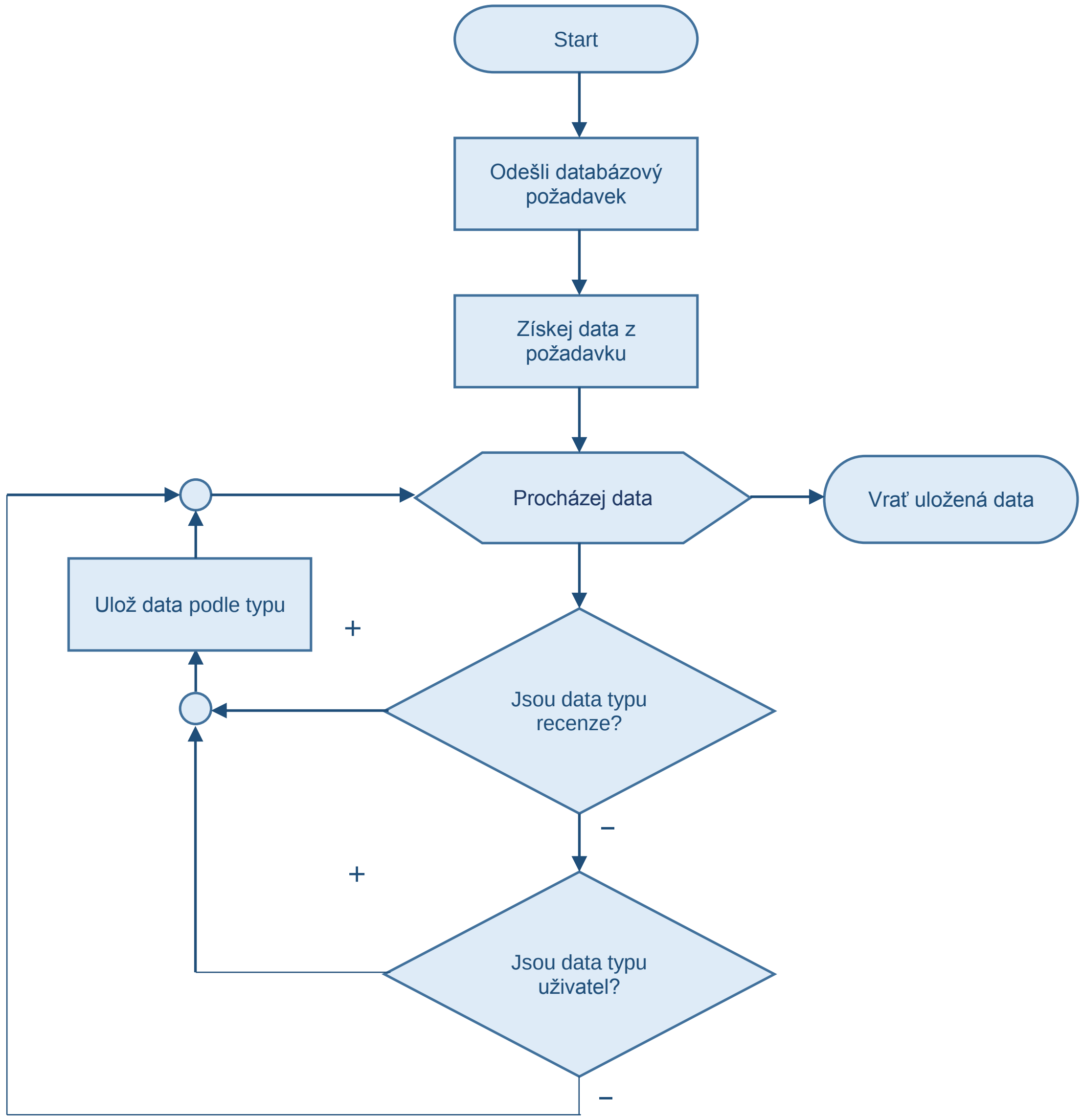
```
{
```

```
        "id": "69806aa1c71c47556a17c965ea02bbfc",
        "key": ["lg-nexus-4", "user-svagis", 0 ],
        "value": null,
        "doc" : *nalinkovaná data zde*
    },
    {
        "id": "69806aa1c71c47556a17c965ea02bbfc",
        "key": ["lg-nexus-4", "user-svagis", 1 ],
        "value": {"_id": "user-svagis"},
        "doc" : *nalinkovaná data zde*
    }
```

Tímto způsobem lze získat všechny závislosti, které recenze potřebuje. V tomto případě jsou to pouze data o autorovi, ale obdobným způsobem lze získat i jiné typy dokumentů. Nevýhodou je, že získaná data nejsou plně agregovaná a musí se rozlišit až na straně serveru, což v tomto případě, kdy se získávají data pouze o jedné recenzi, je jednoduché.

V případě, kdy se získávají data o více recenzích, je filtrování na straně serveru složitější. Například pro získání více recenzí a jejich autorů. V takovém případě map funkce vrátí pole recenzí a pole uživatelů, které jsou opět rozlišeny pomocí posledního klíče v poli klíčů s hodnotou 0 nebo 1. Data však nejsou strukturována. Toho je v aplikaci docíleno na straně serveru v databázovém modelu. Princip strukturování je znázorněn na Obr. 6.

Program nejprve odešle databázový požadavek a získá data. Poté jednotlivá data prochází a ukládá podle typu dat společně s jedinečným identifikátorem. Takto projde všechna získaná data a vrátí jejich setříděnou podobu.



Obr. 6: Diagram strukturování dat na straně serveru

4.3.3 Access control list

Access control list (dále jen ACL) [6] je seznam pro řízení práv a přístupu. Jedná se o základní zabezpečení aplikace, které je implementováno již na úrovni společného předka všech presenterů. Práci s tímto seznamem definují jeho role, zdroje a jednotlivá oprávnění, přičemž role a zdroje umožňují vytvářet hierarchie. ACL vychází z principu, že všechna oprávnění jsou nejprve zakázaná a poté postupně přidávána podle role a zdrojů, ke kterým mají mít přístup.

Pro zabezpečení aplikace bylo navrženo statické ACL. Zdroje jsou definovány jako **názvy** presenterů a jednotlivá oprávnění jako **akce** presenterů.

Rozdělení rolí v aplikaci:

- **Guest** – anonymní návštěvník
- **Validate** – neověřený uživatel
- **Member** – běžný uživatel
- **VIP –** uživatel s vyššími oprávněními
- **Editor –** uživatel, který moderuje recenze a články
- **Admin** – uživatel s absolutními právy

Jednotlivé role od sebe vzájemně dědí, např. VIP uživatel má stejná práva jako běžný uživatel, ale navíc má také svá specifická práva. V případě, že uživatel nemá k dané sekci dostatečná oprávnění, je přesměrován na stránku s přihlášením.

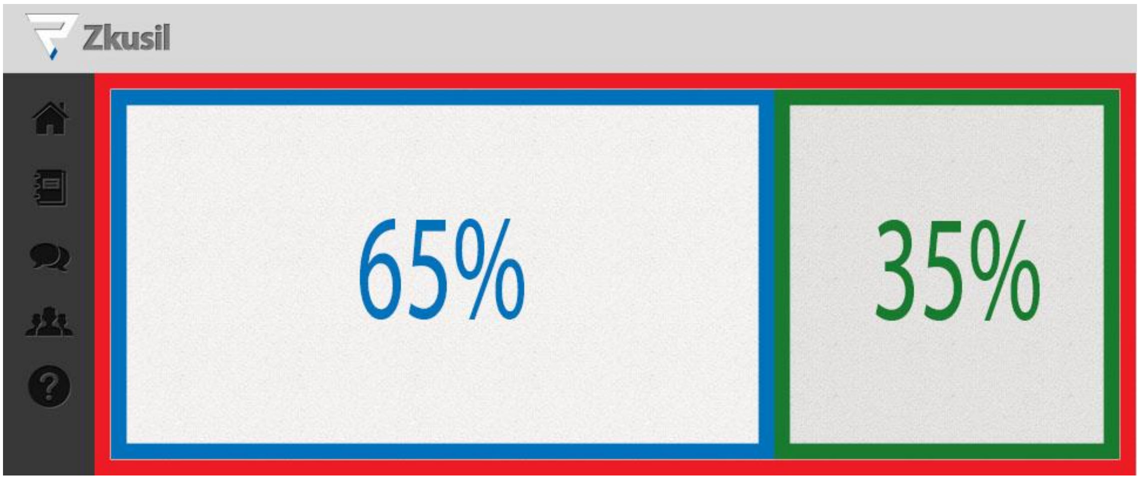
5 Mobilní optimalizace

Cílem optimalizace pro mobilní zařízení v této práci je vycházet z návrhu aplikace pomocí metody Mobile first a následně ji realizovat pomocí dalších metod webdesignu. Mezi tyto metody patří fluid layout a responsive design.

5.1 Fluid layout

Základním prvkem k dosažení mobilního webu je využití fluid layout. Jedná se o rozložení webu, které mění svou velikost v závislosti na velikosti okna prohlížeče nebo displeje zařízení (rozlišení). K tomuto účelu jsou použity relativní jednotky (ems, procenta).

Tato webová aplikace využívá fluid layout především pro procentuální rozdělení sekcí na stránce. Kromě menu je hlavní část umístěna ve dvou sloupcích na zbývající plnou šířku okna. Větší obsahová část je umístěna v levém sloupci o procentuální šířce 65%, kdežto postranní sekce v pravém sloupci má šířku 35% (Obr. 7). Takto si sekce udržují stále **stejný poměr**, ale mají také **flexibilní šířku**. Tím je zaručeno, že se obsah „roztáhne" na každém zařízení.



Obr. 7: Fluid layout v aplikaci

Problém nastává v případě, kdy už je šířka sloupců **velmi malá** nebo naopak **velmi velká**, což způsobuje špatnou čitelnost či kolaps některých elementů. Pro optimální zobrazení a vyřešení problému je využit právě Responsive design.

5.2 Responsive design

Mezi další z prvků webdesignu použitých v této aplikaci patří také Responsive design, který zaručí, že zobrazení stránky bude optimalizováno pro všechny druhy nejrůznějších zařízení (Obr. 8). Především díky vlastnosti Media Queries, která je zahrnuta ve specifikaci CSS3, lze rozpoznat vlastnosti zařízení, na kterém je stránka prohlížena a přizpůsobit tak samotnou stránku a její obsah.

Obr. 8: Přizpůsobení webové stránky na několika zařízeních

Responzivní webdesign má tři základní úrovně:

- Flexibilní struktura
- Flexibilní obrázky
- Media Queries

5.2.1 Flexibilní struktura

Flexibilní struktury se dosahuje především pomocí procentních šířek (viz. Fluid layout). Důležitou součástí této aplikace je kromě použití procentuálních jednotek také použití jednotek em, a to především u velikosti písma. Velikost vyjádřená v pixelech na jednom zařízení nemusí vypadat stejně na zařízení jiném. Pokud jsou relativní jednotky použity vždy i pro margin a padding všech elementů, změna hlavního písma adekvátně změní i tyto hodnoty, a tedy i celé rozložení webové stránky.

Standardní velikost písma v prohlížečích je 16px. Nicméně každý prohlížeč toto základní nastavení nemusí podporovat. Proto je nutné nastavit velikost znovu pomocí CSS, a poté nastavit základní procentuální hodnotu k této standadtní velikosti pro využití relativních jednotek v optimálním poměru 1em = 10px.

Příklad použití:

```
html { font-size: 16px;  /* standart size */ }

body { font-size: 62.5%; /* 1em = 10px */  }
```

S tímto základním nastavením lze již snadno převádět velikosti relativních jednotek vůči absolutním hodnotám pixelů. Pro dopočet je použit vzorec pro Flexibilní strukturu. Kontext je nejčastěji velikost písma rodičovského elementu.

Vzorec:

cílová velikost (px) / velikost kontextu (px) = výsledná velikost (em).

Příklad použití:

```
h1 { font-size: 3em; /* 30px / 10px = 3em */  }
```

Výše popsanými způsoby je vytvořena flexibilní struktura, která reaguje na změny zobrazení jednotlivých zařízení.

5.2.2 Flexibilní obrázky

Technika flexibilních obrázků zajistí, že obrázky se budou přizpůsobovat stejně tak, jako samotná struktura. Princip spočívá v nastavení maximální velikosti obrázku pouze v CSS:

```
img { max-width: 100%; height: auto}
```

5.2.3 Media Queries

K rozpoznání určitého zařízení slouží Media Queries. K tomu využívají tzv. „zlomové body,“ které se aplikují na místa, kde se stávající design rozpadá. Těmito body bývá zpravidla šířka obrazovky zařízení, pro které se optimalizuje. O další přizpůsobení se postará dříve zmíněný Fluid layout.

Příklad použití v aplikaci pro zařízení s vyšším rozlišením:

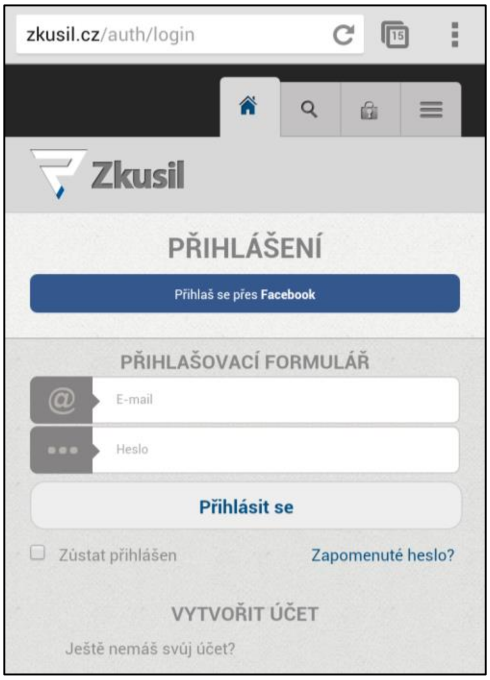
```
@media screen and (min-width: 960px){
      article { font-size: 1.4em; /* 1.4em = 14px */  }
}
```

6 Uživatelské rozhraní

V této části je postupně rozebráno uživatelské rozhraní z pohledu uživatele mobilního zařízení.

6.1 Přihlášení / Registrace

Aplikace předpokládá, že hlavním tvůrcem obsahu budou uživatelé. Jednou z nejdůležitějších částí stránek je proto přihlášení (Obr. 9). Průběh a náročnost takového přihlášení by měla být co nejjednodušší. K tomuto účelu bylo naimplementováno přihlašování přes sociální síť Facebook. To umožňuje přihlásit se do systému jediným klepnutím. Pro uživatele, kteří o toto přihlašování nemají zájem, je tu také klasické přihlašování pomocí kombinace email / heslo.



Obr. 9: Přihlášení k uživatelskému účtu

Pokud uživatel nemá vytvořen účet a registruje se přes Facebook, pouze odsouhlasí zpracování jeho údajů, a je okamžitě zaregistrován. Způsob této registrace nevyžaduje žádné další ověření, jelikož se předpokládá, že totožnost uživatele a jeho účtu je již potvrzena.

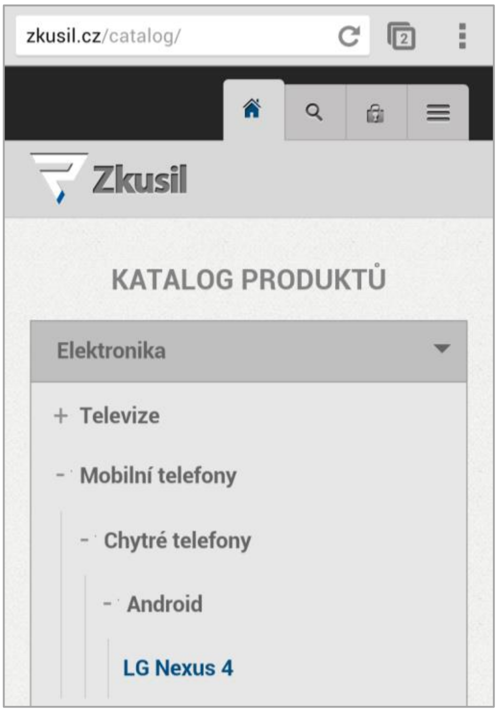
V případě, že uživatel nemá účet a nechce se registrovat přes Facebook, je tu možnost klasické registrace, kde vyplní registrační formulář. Pro jednoduché ověření totožnosti je vyžadováno ověření emailové adresy uživatele. Jakmile uživatel ověří adresu, jeho účet je plnohodnotně vytvořen a proběhne automatické přihlášení.

6.2 Hlavní sekce

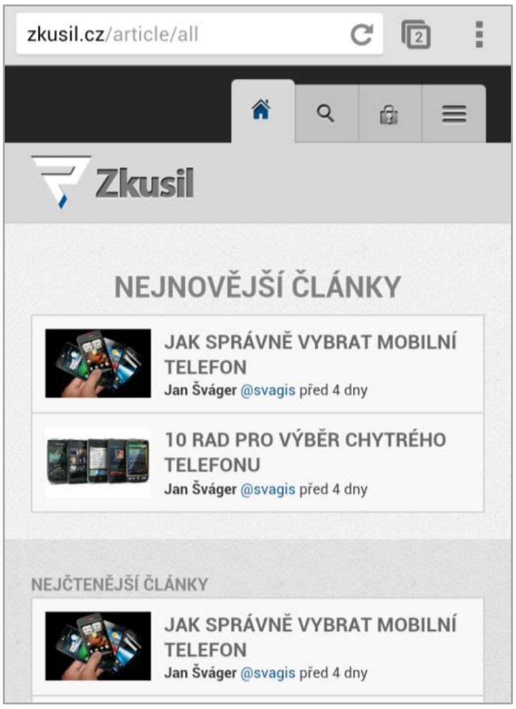
Aplikace má několik základních sekcí, které zároveň slouží jako výchozí menu. Tyto sekce jsou v jednotlivých podkapitolách popsány a znázorněny na obrázcích.

6.2.1 Katalog

Katalog (Obr. 10) slouží pro nalezení produktů podle určité kategorie. Je vytvořen pomocí stromové struktury. V aplikaci se chová jako rozevírací menu. Každá položka odkazuje na svou vlastní recenzi.



Obr. 10: Sekce katalog produktů



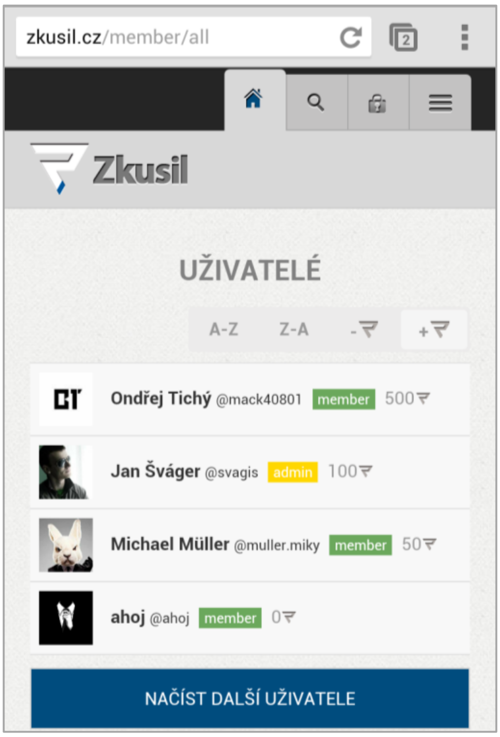
Obr. 11: Sekce články

6.2.2 Články

Uživatelské články (Obr. 11) slouží především jako doplněk ke klasickým recenzím. Mohou obsahovat například různé tipy, jak vybrat správné zařízení atd. Hlavní sekce obsahuje vždy nejnovější články. Vedlejší sekce zobrazuje články, které jsou hodnocené podle určitého kritéria – např. podle hodnocení článku nebo počtu zhlédnutí.

6.2.3 Uživatelé

Sekce s uživateli (Obr. 12) v hlavní části vypisuje seznam všech uživatelů, které je možné třídit podle jména a počtu zkušenostních bodů, které jednotliví uživatelé vlastní. Třídit lze vzestupně i sestupně. Ve vedlejší části jsou uvedeni uživatelé podle specifických kritérii – např. nejlepší uživatelé, s nejvíce recenzemi atd.



Obr. 12: Sekce uživatelé

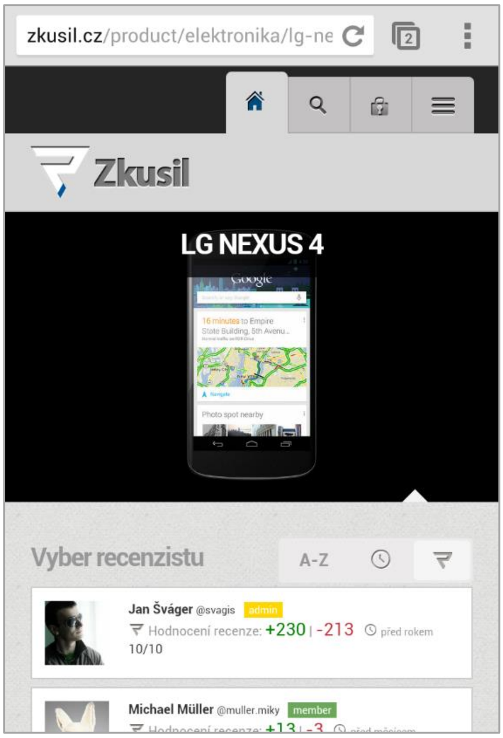
Obr. 13: Sekce často kladené dotazy

6.2.4 Často kladené dotazy

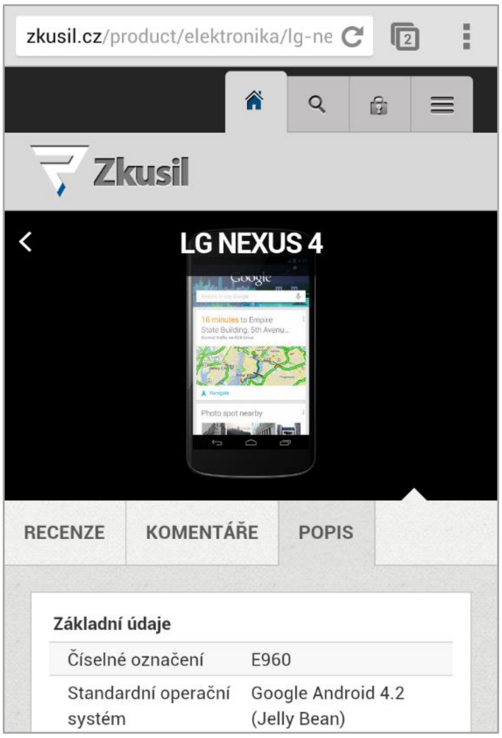
Poslední hlavní sekcí jsou často kladené dotazy (Obr. 13). Ty jsou realizovány jako rozevírací menu v hlavní části. Ve vedlejší části je formulář pro další dotazy uživatelů.

6.3 Recenze

Hlavní zaměření celé webové aplikace spočívá v uživatelských recenzích. Každá recenze má svého vlastního recenzenta, podle něhož je zvolena samotná recenze (Obr. 15). Jednotlivé recenzenty lze řadit podle jména, data vydání a hodnocení jejich recenze.



Obr. 15: Recenze - výběr recenzenta



Obr. 14: Recenze - článek, komentáře, popis

Po zvolení recenzenta se v hlavní části zobrazí stránka s recenzí, která obsahuje samotný článek, komentáře a popis produktu (Obr. 14). Ve vedlejší části je k dispozici komponenta pro hodnocení samotné recenze určená pro ostatní uživatele.

Závěr

Náplní mé práce bylo vytvoření víceuživatelského recenzního serveru s využitím dokumentové databáze CouchDB a s optimalizací pro mobilní zařízení. V práci jsem se zaměřil na řádný vývoj moderních webových a mobilních řešení. Jednotlivé metodiky jsem zkoumal především z tohoto hlediska.

Aplikaci jsem navrhnul způsobem, který vychází z Mobile first – tedy návrh, který začíná u mobilních telefonů. Díky této metodice jsem kladl důraz na skutečně důležité prvky, jednoduchost a srozumitelnost. Tento návrh jsem realizoval především pomocí Responsive designu, díky kterému jsem mohl aplikaci optimalizovat pro různé druhy zařízení od mobilních telefonů po stolní počítače.

Součástí mého návrhu byla také struktura aplikace, která dbá na dostupnost, rozšiřitelnost, použitelnost a bezpečnost. K tomu mi dopomohl i použitý Nette Framework, který většinu základní bezpečnostních chyb eliminuje. Jelikož se jedná především o víceuživatelský systém, implementoval jsem další bezpečnostní vrstvu Access control list. Ta vychází z navržené struktury a pracuje na základní vrstvě aplikace, čímž umožňuje efektivně omezovat přístup podle typu uživatele.

V práci jsem se zaměřil také na efektivní použití databáze CouchDB – především v získávání dat a jejich následného snadného použití v aplikační části. K tomuto účelu jsem navrhnul databázové modely (viz. 4.3.1). Ty řeší také problematičtější agregaci dat při požadavcích na více typů dokumentů.

Podařilo se mi navrhnout systém, který umožňuje uživatelům získat přehled o reálných zkušeností ostatních uživatelů ve formě recenzí, které mohou také vytvářet. Do aplikace jsem zakomponoval systém hodnocení, který uživatelům umožňuje recenze lépe třídit. Jsem přesvědčený, že se mi zadání práce podařilo splnit.

Seznam použité literatury

[1] ANDERSON, J. Chris, Jan LEHNARDT a Noah SLATER. *CouchDB: The Definitive Guide* [online]. 2010 [cit. 2012-10-16]. Dostupné z: http://guide.couchdb.org/

[2] WROBLEWSKI, Luke. Mobile first. New York: A Book Apart, 2010. ISBN 978-193-7557-027.

[3] PHP: *Hypertext preprocessor* [online]. 1997, 2012-10-16 [cit. 2012-10-16]. Dostupné z: http://php.net/

[4] GRUDL, David. NETTE FOUNDATION. *Nette Framework* [online]. 2008, 2013 [cit. 2013-05-09]. Dostupné z: http://nette.org/

[5] Nette Framework: MVC & MVP: *Zdroják.cz* [online]. 2009 [cit. 2013-05-09]. Dostupné z: http://www.zdrojak.cz/clanky/nette-framework-mvc-mvp/

[6] Access control list. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-05-13]. Dostupné z: http://cs.wikipedia.org/wiki/Access_control_list

[7] FROST, Brad. This is the web. *Brad Frost Web* [online]. 2012-03-29 [cit. 2013-05-13]. Dostupné z: http://bradfrostweb.com/blog/post/this-is-the-web/

Seznam obrázků

Obsah CD

Přiložené CD obsahuje:

application/	zdrojové kódy aplikace
app/	vlastní aplikace v jazyce PHP
css/	kaskádové styly
images/	obrázky použité v aplikaci
js/	zdrojové kódy Javascriptu
libs/	použité knihovny
images/	přiložené obrázky a diagramy
thesis/	text vlastní práce

Na CD není umístěn konfigurační soubor z důvodu ochrany přístupových hesel.