VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

INZERTNÍ PORTÁL S OCHRANOU KUPUJÍCÍHO

ADVERTISING PORTAL WITH BUYER PROTECTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **PETER BABIC**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. VLADIMÍR BARTÍK, Ph.D.**
SUPERVISOR

BRNO 2017

Zadání bakalářské práce/19827/2016/xbabic08

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Babic Peter**
Obor: Informační technologie
Téma: **Inzertní portál s ochranou kupujícího**
Advertising Portal with Buyer Protection
Kategorie: Informační systémy

Pokyny:

1. Seznamte se s principy tvorby informačních systému na webu a přílušnými technologiemi.
2. Navrhněte webový inzertní systém, který bude realizaovat základní operace s prodejci, inzeráty a kupujícími. Kromě toho bude poskytovat ochranu kupujícího tím, že bude prostředníkem mezi kupujícím a prodávajícím s tím, že bude zajišťovat i samotnou platbu. Dále zde bude možné inzeráty filtrovat pomocí jednoduchého rozhraní s mapou. Návrh konzultujte s vedoucím.
3. Navržený systém implementujte.
4. Ověřte funkčnost implementovaného systému na vhodném vzorku dat.
5. Zhodnoťte dosažené výsledky a další možné pokračování tohoto projektu.

Literatura:

- Welling, L., Thomsonová, L.: PHP a MySQL: rozvoj webových aplikací. Vyd. 1. Praha: SoftPress, 2003, 910 s. ISBN 80-86497-60-7.
- Žára, O.: JavaScript - Programátorské techniky a webové technologie, Computer Press, 2015. ISBN: 978-80-251-4573-9

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Bartík Vladimír, Ing., Ph.D.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Hlavným cieľom bakalárskej práce je navrhnúť a implementovať inzertný systém s webovým užívateľským rozhraním, ktorý bude realizovať základné operácie s inzerátmi, predajcami a kupujúcimi. Na rozdiel od existujúcich riešení poskytuje ochranu kupujúceho tým, že plní úlohu prostredníka medzi zúčastnenými stranami. Inzertný portál umožňuje okrem správy informácií o inzerátoch taktiež správu informácií o používateľoch a zaznamenáva ich komunikáciu spojenú s obchodom. Súčasťou portálu je možnosť filtrovať inzeráty pomocou interaktívnej mapy. Systém je postavený na PHP mikroframeworku Slim a pre správu dát využíva databázové systémy MySQL a MongoDB.

Abstract

The main purpose of bachelor thesis is to design and implement an advertising system with a web user interface, which will be able to perform basic operations with advertisements, sellers and buyers. In contradistinction to existing solutions, it provides buyer protection by fulfilling an intermediary role between interested parties. In addition to managing ad information, the Advertising portal also allows a management of information about users and stores their transaction-related communication. Part of the portal is the ability to filter ads using interactive map. The system is built on Slim PHP Microframework and uses MySQL and MongoDB database systems for data management.

Kľúčové slová

Informačný systém, Inzertný portál, PHP, Slim, HTML, CSS, Javascript, MongoDB, Geospatial

Keywords

Information system, Advertising portal, PHP, Slim, HTML, CSS, Javascript, MongoDB, Geospatial

Citácia

BABIC, Peter. *Inzertní portál s ochranou kupujícího.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

Inzertní portál s ochranou kupujícího

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Vladimíra Bartíka Ph.D. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Peter Babic
17. mája 2017

Poďakovanie

Rád by som poďakoval svojmu vedúcemu pánovi Ing. Vladimírovi Bartíkovi Ph.D. za užitočné rady, flexibilitu a ústretovosť pri spolupráci.

Obsah

Kapitola 1

Úvod

Tento dokument vznikol ako bakalárska práca na Fakulte informačných technológií VUT v Brne. Už niekoľko rokov existuje množstvo webových informačných systémov s funkciou inzertného portálu. Tieto portály sú prístupné za určitých podmienok zadarmo, avšak bez záruky na predávaný tovar, či službu. V existujúcich riešeniach podobného zamerania je do istej miery používateľ chránený možnosťou ohodnotiť predajcu. Tieto systémy však neposkytujú dostatočné nástroje pre ochranu ktorejkoľvek zo zúčastnených strán. Z pravidla býva napríklad možné si za účelom pridania jedného inzerátu založiť nový používateľský účet, ktorý môže byť spolu s inzerátom odstránený hneď po prijatí platby za tovar. Ďalej, komunikácia medzi zúčastnenými stranami obchodu býva realizovaná mimo portál a v prípade nespokojnosti niektorej zo strán nemusí mať poškodený používateľ prostriedky pre prípadnú právnu pomoc.

Množstvo používateľov nespokojných so zakúpeným tovarom viedlo k zrodu dohadov o možnostiach ochrany kupujúceho. Preto vznikla myšlienka inzertného portálu, ktorý by okrem bežných služieb vykonával napríklad funkciu sprostredkovateľa platby medzi predajcom a kupcom. Tento nápad je do istej miery inšpirovaný už existujúcim riešením portálu `www.jaspravim.sk`, ktorý sa však zameriava výlučne na predaj a hodnotenie služieb. Podobný systém zameraný na predaj a kúpu tovaru by bol teda v rámci slovenských inzertných portálov prvým svojho druhu.

Obsah práce je rozdelený do niekoľkých celkov popisujúcich priebeh tvorby systému od analýzy a návrhu jeho funkcionality až po detaily implementácie. Prvá časť dokumentu je venovaná základným princípom tvorby webových informačných systémov a príslušných technológii. Nasleduje rozbor možných spôsobov ochrany kupujúceho a z nich vyplývajúcej špecifikácii požiadaviek na systém. Témou ďalšej kapitoly je návrh informačného systému vrátane popisu konceptuálneho modelu dát. V tretej kapitole sú popísané ostatné detaily implementácie systému z hľadiska jeho architektúry, či interakcie s používateľom. Obsah predposledného celku je zameraný na konkrétne použité technológie a spôsob ich integrácie. V závere sú zhrnuté dosiahnuté výsledky, možnosti rozšírenia projektu a jeho prípadné pokračovanie.

Kapitola 2

Princípy tvorby webových informačných systémov

Táto kapitola je venovaná princípom tvorby informačných systémov na webe a súvisiacim používaným technológiám.

2.1 Webový informačný systém

Pri snahe definovať webový informačný systém je najprv nutné špecifikovať informačný systém ako taký. Na webovej stránke www.informacne-systemy.sk je uvedená jedna z možných definícii informačného systému:

„Informačný systém je možné chápať ako systém vzájomne prepojených informácií a procesov, ktoré s týmito informáciami pracujú“ [1].

2.2 Používané návrhové vzory

Pri tvorbe webového informačného systému sa s cieľom vyvíjať udržovateľný kód používajú rôzne návrhové vzory. Medzi najznámejšie patria MVC (Model-View-Controller), MVP (Model-View-Presenter). Tu je nutné podotknúť, že tieto modely nereprezentujú rovnaký návrhový vzor, hoci to tak môže na prvý pohľad vzyerať. Pre tvorbu webových informačných systémov je z vyššie uvedených modelov aktuálne najrozšírenejší model MVC. Každá z troch vrstiev MVC architektúry je určená pre istý typ úloh, ktoré by sa nemali medzi vrstvami presúvať.

Model
Model je vrstva zabezpečujúca komunikáciu s databázou. Mala by to byť jediná vrstva, ktorá má kontakt priamo s databázovým systémom. Možná dodatočná funkcia vrstvy model je taktiež mapovanie dát databázy na objekty.

View
View je statická vrstva, ktorej úlohou je vhodne vizualizovať dáta z databázy. Mala by slúžiť ako zapúzdrenie dát a užívateľsky prijateľnou formou tieto dáta prezentovať. Podľa filozofie architektúry MVC by nemala vrstva View obsahovať žiadnu, prípadne len minimálnu logiku. V kontexte webových systémov by to mohli byť fragmenty HTML kódu, v ktorých sú injektované fragmenty funkčného kódu (napríklad jazykov PHP alebo Python)

s cieľom vkladať už predpripravený text. Komponenty patriace do vrstvy View by nemali obsahovať žiadne zložité výpočty či prácu s databázou.

Controller
Controller je statická vrstva, ktorej úlohou je získať dáta od správnych komponent z vrstvy Model, vykonať nad nimi potrebné akcie a v spracovanej forme ich vizualizovať pomocou správnych komponent z vrstvy View. Vrstva Controller teda reprezentuje hlavnú logiku webovej aplikácie.

MVP
Návrhový vzor MVP je odvodený od návrhového vzoru MVC. Zjednodušene sa dá povedať, že umožňuje v istých prípadoch priame prepojenie vrstiev View a Model, čím sa istá časť funkcionality vrstvy Controller presúva do vrstvy View. Vrstvy View a Controller spolupracujú a vrstva View v modeli MVP stráca atribút statická. Vrstva Controller je v tomto návrhovom vzore premenovaná na Presenter.

V ponímaní architektúry trojvrstvového dátového modelu je možné zjednodušene zaradiť vrstvy Controller (alebo Presenter) a View do prezentačnej vrstvy. Vrstva modelu je v architektúre trojvrstvového dátového modelu súčasťou aplikačnej (alebo tiež funkčnej) vrstvy.

2.3 Používané programovacie jazyky a technológie

Pri tvorbe webových informačných systémov je z hľadiska používaných technológii nutné rozlíšiť časti programovaného systému. V základe by sa dali rozdeliť do troch kategórii:

- strana klienta,
- strana servera,
- databázový systém.

2.3.1 Strana klienta

Na strane klienta je aktuálne najrozšírenejšou technológiou kombinácia jazykov HTML a CSS. Tieto dve technológie sú dnes v každej modernej stránke rozšírené o jazyk Javascript, ktorý dodáva dinamiku stránke na strane klienta.

Pre jazyky CSS a Javascript je vyvíjaných niekoľko nadstavieb či knižníc spoločne nazývaných „frontend frameworky“. Výhodou takýchto frameworkov je, že je v nich z pravidla množstvo často riešených problémov už vyriešených a programátorovi je tak umožnené písať udržovateľnejší kód v kratšom čase. Najznámejšie používané CSS frameworky sú napríklad:

- `Bootstrap`
- `pure.css`
- `w3.css`
- `foundation.css`
- a ďalšie

Medzi najpoužívanejšie Javascript frameworky sa dnes radia napríklad:

- `react.js`
- `angular.js`
- `vue.js`
- `jQuery.js`
- a ďalšie

V prípade moderných Javascript frameworkov sa postupne presadzuje myšlienka generovania HTML kódu pomocou programovacieho jazyka Javascript, čo reprezentuje značne rozdielny koncept od doterajšieho, kde boli fragmenty Javascript kódu injektované do HTML.

2.3.2 Strana servera

Pre programovanie webovej aplikácie na strane servera sú aktuálne najpoužívanejšie programovacie jazyky Java, C# na platforme .NET, PHP, Python, Javascript alebo Go. Každý z jazykov má svoje výhody aj nevýhody. Výber jazyka závisí do značnej miery od charakteru aplikácie a preferencii programátora. Tak, ako na strane klienta, existuje aj pre vývoj na strane servera množstvo používaných frameworkov. Medzi najznámejšie patria:

- `Node.js` (pre Javascript bežiaci na strane servera)
- `Spring` (pre aplikácie v jazyku Java)
- `Laravel` alebo `Symphony` (pre aplikácie v jazyku PHP)
- `Django` alebo `Flask` (pre aplikácie v jazyku Python)
- a ďalšie

Rovnako, ako na strane klienta, je možné frameworky a knižnice úplne vynechať, čo však môže mať za následok pomalý vývoj, či zložitú údržbu kódu. Použitie frameworku však predpokladá istý čas venovaný učeniu sa filozofie frameworku a správneho návrhu aplikácii na ňom postavených.

2.3.3 Databázový systém

Medzi najpoužívanejšie databázové systémy sa ešte do nedávna radili z pravidla relačné systémy (RDBMS). Práca s nimi však v záujme programovať objektovo orientovaným prístupom znamenala nutnosť mapovať dáta z tabuliek databázy na objekty, s ktorými je v systéme manipulované. Tento prístup sa dodržuje do dnes a záznamy z tabuliek sú zapúzdrované funkcionalitou všeobecne známou pod skratkou „CRUD“(Create, Read, Update, Delete). Možnosť manipulovať so záznamami v tabuľke pomocou týchto štyroch operácii sa do istej miery blíži ku konceptu objektovo orientovaného programovania, no nie je možné to tak priamo nazvať.

To viedlo ku vzniku objektovo-relačných databázových systémov (ORDBMS), či neskôr ku vzniku takzvaných NoSQL databáz, ktoré sú schopné uchovávať priamo objekty a pri práci s databázou vracajú dáta napríklad vo formáte JSON.

Medzi najpoužívanejšie databázové systémy pre vývoj webových aplikácii aktuálne patria:

- MySQL alebo SQLite (relačné databázové systémy)
- PostgreSQL (objektovo-relačné databázové systémy)
- MongoDB (NoSQL, alebo objektové databázové systémy)

NoSQL databázy sa vyznačujú rýchlym vykonaním databázového príkazu za cenu značnej miery redundancie dát. Neznamená to však, že relačné databázové systémy nie je vhodné používať. Toto rozhodnutie je ponechané na programátorovi a závisí hlavne od charakteru aplikácie.

Zhrnutie
V tejto kapitole boli zhrnuté aktuálne používané koncepty a technológie spojené s vývojom webových informačných systémov. Nasledujúca kapitola bude venovaná analýze existujúcich informačných systémov v úlohe inzertného portálu a ich vlastnostiam, ďalej možnostiam ochrany kupujúceho a vyvodeniu špecifikácie požiadaviek na navrhovaný systém.

Kapitola 3

Analýza a špecifikácia

Pri vývoji inzertného portálu je nutné najskôr špecifikovať požiadavky na systém. Vzhľadom k množstvu existujúcich riešení môže byť vhodné zvážiť ich vlastnosti a zamerať sa na ich nedostatky. V nasledujúcich častiach sa budem snažiť popísať portály www.bazos.sk a www.bazar.sk z hľadiska ich použiteľnosti a možností ochrany kupujúceho. V závere kapitoly bude zhrnutá vyplývajúca špecifikácia na vyvíjaný systém.

3.1 Existujúce inzertné portály

V súčastnosti je v prevádzke minimálne 100 slovenských inzertných portálov. Najznámejšie z nich sú napríklad www.bazos.sk, alebo www.bazar.sk. Okrem spomínaných systémov patria medzi známe portály napríklad www.123inzercia.sk, www.inzercia.sk, www.burza.sk atď. Prvé dva z vymenovaných systémov majoritne pokrývajú funkcionalitu, ktorú poskytujú všetky ostatné inzertné portály, preto ich použijem ako ilustratívne.

3.2 Vlastnosti inzertných portálov

Inzertné portály sú tak, ako každý informačný systém prístupný verejnosti, navrhnuté s dôrazom na jednoduchosť pri vkladaní aj vyhľadávaní inzerátov. Ak preskúmame existujúce riešenia spomenuté vyššie, bolo by možné ich funkcionalitu rozdeliť na dve hlavné časti a to:

- správa inzerátov,
- správa používateľských účtov.

Pod pojmom správy inzerátov je zahrnutý mechanizmus ich pridávania, kategorizácie a mechanizmus ich vyhľadávania. Všetky inzertné portály taktiež podporujú aspoň jednoduchú správu používateľských účtov. Motiváciou je, aby bolo možné na inzerát reagovať, prípadne používateľa ohodnotiť.

3.3 Správa inzerátov

S cieľom navrhnúť mechanizmus správy inzerátov je vhodné určiť, ktoré informácie bude systém o inzerátoch evidovať. To je možné analyzovať z hľadiska ich vytvárania alebo vyhľadávania.

3.3.1 Vytvorenie inzerátu

Pri vytváraní inzerátov je potrebné premyslieť, akým spôsobom bude inzerát dohľadateľný. Parametre inzerátu, ktoré sú štandardne prístupné v obidvoch systémoch sú *Názov*, *Obsah inzerátu*, *Cena*, *Kategória* a *Inzerent*. Inzerent má taktiež možnosť k inzerátu pridať fotografie, či link na video z youtube. Portály www.bazos.sk a www.bazar.sk umožňujú taktiež zaradiť inzerát do vhodnej podkategórie, ďalej musí inzerent definovať, či sa v prípade pridávaného inzerátu jedná o dopyt alebo ponuku. Okrem toho ponúkajú možnosť uviesť mesto, kde je možné ponúkaný tovar alebo službu vyzdvihnúť. Portál www.bazar.sk umožňuje pri zadávaní inzerátu uviesť taktiež stav tovaru (na výber je *Používaný* alebo *Nový*).

3.3.2 Vyhľadávanie inzerátov

Pre vyhľadávanie inzerátov je použitý filter, ktorý vyplýva z parametrov uvádzaných pri vytvorení inzerátu. Každý zo spomínaných parametrov (*Názov*, *Obsah inzerátu*, *Cena*, *Kategória*, *Lokalita*, *Kategória*) má vo vyhľadávacom formulári miesto a nie je dôvod ho odstraňovať. Pri prehliadaní nájdených inzerátov je úvodný vyhľadávací formulár transformovaný na filtrovací formulár, kde má používateľ možnosť upraviť, či doplniť svoje parametre vyhľadávania. Pri filtrovaní taktiež pribudla možnosť inzeráty zoradiť podľa obľubenosti (počet kliknutí) alebo podľa ceny.

3.4 Správa používateľských účtov

Všeobecne by sa dalo povedať, že systémy www.bazos.sk a www.bazar.sk sú na správu používateľských účtov zamerané len okrajovo. Registráciu používateľa je možné vykonať zároveň s úkonom pridania inzerátu. Registrácia je teda nutná len z dôvodu pridania inzerátu a pochopiteľne v prípade zámeru ohodnotiť iného používateľa. Pre vyhľadávanie, prezeranie, či reakciu na inzerát nie je registrácia vyžadovaná. To, čo by mohlo byť vnímané, ako neodstatok je, že možnosť hodnotenia nie je podmienená uzavretím obchodu, či akýmkoľvek iným kontaktom s inzerujúcou stranou. To môže viesť (podľa zámeru hodnotiaceho používateľa) k umelo vytvorenej dobrej, či naopak zlej domnienke o inzerujúcom používateľovi.

3.5 Možnosti ochrany kupujúceho

Po dôslednom preskúmaní možností ochrany kupujúceho, či predajcu som došiel k záveru, že v existujúcich riešeniach sa prevádzkovateľ portálu úplne vzdáva akejkoľvek zodpovednosti za obsah inzerátov a prípadnú spôsobenú škodu. Vyplýva to jednak zo spôsobu, akým sú portály koncipované, ale taktiež priamo z podmienok používania [15], kde stojí: *„Uzavretím obchodu dochádza teda ku vzniku kúpnej zmluvy medzi konkrétnymi Používateľmi a všetku zodpovednosť za jej riadne plnenie ako aj za prípadné reklamačné konanie a plnenie z neho vyplývajúcich nárokov nesú daní Používatelia.“*
a ďalej:
„Prevádzkovateľ nenesie zodpovednosť za obsah inzerátov, hodnotení ani za žiadne záväzky medzi Používateľmi navzájom, ktoré vzniknú na základe kúpy a predaja jednotlivých inzerovaných tovarov a služieb.“.
Každopádne, tieto portály dávajú používateľom možnosť pridať hodnotenie, prípadne uzavretie obchodu zvážiť na základe už existujúcich hodnotení. Existuje však niekoľko ďalších možností, na základe ktorých by bolo možné kupujúceho používateľa chrániť.

Zaznamenávanie zmien v inzeráte
Jednou z nich je napríklad poskytnúť mechanizmus pre sledovanie zmien v inzeráte. Zmeny v znení inzerátu, prípadne nepravdivé fotografie ponúkaného produktu bývajú často predmetom nedorozumení. Zmeny inzerátu by mohli byť viditeľné z dôvodu ucelenejšieho obrazu o inzeráte, ale taktiež preto, aby mal kupujúci použ9vateľ podklady pre svoju ochranu v prípade podvodu, či nedorozumenia.

Zamknutie inzerátu
Ďalšou možnosťou môže byť takzvané zamknutie inzerátu. Zamknutím inzerátu mám na mysli istotu, že ak inzerent už obdržal nejakú reakciu na svoj inzerát a obchod je už dohodnutý, nebude možné inzerát zmazať ani od obchodu odstúpiť. Pre kupujúceho používateľa by tak vyplývala istota, že sa v prípade nespokojnosti môže odvolať na znenie inzerátu a inzerát bude stále na portáli dohľadateľný. Tento spôsob ochrany úzko súvisí s nasledujúcim bodom.

Používateľské transakcie
Na to, aby v systéme bolo možné rozoznať, ktorý inzerát je už vo fáze obchodu a ktorý nie, je potrebné zaviesť do portálu mechanizmus pre evidovanie používateľských transakcií. Používateľskou transakciou je myslený rozhovor medzi predajcom a kupujúcim spojený s inzerátom. Ak by bol celý rozhovor zúčastnených strán (vrátane prechodov medzi stavmi transakcie) vedený v systéme, obe strany budú mať celý rozhovor k dispozícii od začiatku konverzácie až do ukončenia obchodu. Na základe informácií o všetkých potenciálnych obchodoch k inzerátu vrátane ich stavu by taktiež mohol byť vyriešený problém s nepravdivými, či umelými hodnoteniami. Používateľská transakcia by sa dala využiť ako priestor pre uverejnenie hodnotenia opačnej zo zúčastnených strán, čo by bolo možné len po vykonaní obchodu.

Administrátor v roli prostredníka medzi stranami obchodu
Vyššie spomenuté prídavné funkcie systému by mohli priniesť viac bezpečnosti do verejného inzertného portálu. Ďalšou a poslednoou možnosťou, ktorej sa budem v tejto práci venovať je postavenie administrátora systému do roli prostredníka medzi kupujúcou (platiacou) stranou a medzi predajcom. Ak by platba prebiehala cez tretiu stranu, ktorá by mala prehľad o znení inzerátu a zároveň všetky informácie o zúčastnených používateľoch, pravdepodobne by sa tým mala zvýšiť snaha predajcu o čo najlepšie naplnenie očakávaní kupcu. Ak by mal byť portál orientovaný na ochranu kupujúceho, bude to práve kupujúci, kto rozhodne o spokojnosti s prijatým tovarom a teda aj o príjemcovi platby.

3.6 Špecifikácia požiadaviek

Z prieskumu existujúcich riešení a možných bezpečnostných rozšírení je možné vyvodiť nasledovné požiadavky na funkčnosť inzertného portálu:

- Portál by mal poskytovať aspoň základnú funkcionalitu pridávania inzerátov s možnosťou ich zaradiť do správnej kategórie v závislosti na charaktere inzerátu.
- Inzeráty by malo byť možné vyhľadávať na základe typu (ponuka alebo dopyt), obsahu (hľadané frázy by sa mali nachádzať v texte názvu inzerátu alebo v jeho znení), lokality (zadaním mesta a maximálnej vzdialenosti od neho), kategórie, podkategórie, maximálneho možného stavu a samozrejme cenového rozsahu.
- Zoznam inzerátov by malo byť možné filtrovať aj počas jeho prehliadania s možnosťou ovplyvniť poradie zobrazovaných inzerátov podľa kritérií ako cena alebo počet videní.
- Pridávať inzeráty, či reagovať na ne bude umožnené len prihláseným používateľom. To neplatí pre prehliadanie inzerátov.
- Portál by mal umožňovať evidenciu všetkých používateľských transakcií spojených s inzerátom. Používatelia by mali viesť konverzáciu výlučne cez rozhranie portálu, e-mailový kontakt na zúčastnené strany by mal byť prístupný len administrátorovi.
- Portál by mal rozoznávať stavy transakcie (*otvorená*, *obchod odsúhlasený jednou zo strán*, *obchod odsúhlasený oboma stranami*, *tovar zaplatený*, *tovar doručený*).
- Možnosť ohodnotiť používateľa bude len po dokončení celej používateľskej transakcie. Možnosť budú mať obe zúčastnené strany.
- Ak je používateľská transakcia v stave *obchod odsúhlasený oboma stranami*, nebude možné odstrániť inzerát ani odregistrovať účet inzerenta.
- Systém bude ukladať zmeny v obsahu či vlastnostiach inzerátu a budú prístupné každému účastníkovi používateľskej transakcie spojenej s inzerátom.
- V prípade nevhodnej kategórie bude mať každý používateľ možnosť nahlásiť nevhodné zaradenie inzerátu.
- V prípade nevhodného, či urážlivého znenia inzerátu bude mať každý používateľ možnosť nahlásiť nevhodný inzerát.
- Súčasťou systému bude prehľadné rozhranie administrácie, kde bude možné vyhľadávať transakcie a vidieť ich stav, vyhľadávať používateľov a spravovať ich role a vidieť nahlásené inzeráty.
- Administrátor bude mať možnosť odstrániť, či upraviť inzerát, odregistrovať používateľa, spravovať používateľské role a spravovať stav transakcií.

Zhrnutie

V tejto kapitole boli analyzované vlastnosti existujúcich inzertných portálov a možnosti ochrany kupujúceho. V závere kapitoly boli zhrnuté vyplývajúce požiadavky na navrhovaný inzertný portál. V nasledujúcej kapitole je podrobne rozvedený návrh inzertného portálu z pohľadu modelu dát a používateľského rozhrania.

Kapitola 4

Návrh informačného systému

V tejto kapitole je popísaná dekompozícia celkového systému na menšie podsystémy. Vodítkom k dekompozícii sú do istej miery požiadavky na funkčnosť inzertného portálu definované v predošlej kapitole. Ďalšia časť kapitoly je venovaná vyplývajúcemu konceptuálnemu návrhu relačného modelu dát. V závere kapitoly bude vyhradený priestor pre návrh spôsobu interakcie systému s používateľom.

4.1 Dekompozícia systému

Pri snahe dekomponovať systém na menšie celky je možné ho rozdeliť do niekoľkých častí:

- správa inzerátov,
- správa používateľských účtov,
- správa používateľských transakcii,
- oprávnenia administrátora,
- vlastnosti webového rozhrania.

4.1.1 Správa inzerátov

Z požiadaviek na inzertný portál v prvom rade plynie, že by mal umožňovať filtrovanie inzerátov na základe hľadaného textu. Je teda zrejmé, že systém musí byť schopný ukladať okrem pôvodného názvu a obsahu inzerátu taktiež ich formy, v ktorých by bolo možné efektívne vyhľadávať. Najjednoduchšou variantou takej formy môže byť napríklad text s odstránenou interpunkciou a prevedený do malých písmen.

Vzťah kategórií, subkategórii a inzerátov
Inzerát by mal podľa požiadaviek taktiež patriť do konkrétnej kategórie a subkategórie. Zároveň by pod jednu kategóriu (či subkategóriu) mohlo patriť viac inzerátov. Vzťah subkategórie a inzerátu sa teda dá namodelovať ako vzťah 1:N. Problémom však môže byť vzťah entitných množín kategória a subkategória. Tie majú medzi sebou väzbu M:N, keďže konkrétna subkategória môže spadať pod viaceré kategórie (napríklad subkategória *Káble* môže spadať pod kategórie *Elektronika*, *Telefóny a mobily*, *Hudba a audio*) a zároveň niekoľko subkategórii môže patriť pod jednu kategóriu (napríklad kategória *Vzdelanie* môže

obsahovať subkategórie *Knihy*, *Software* a *Školské potreby*). V prípade dostatočného prvotného návrhu možných kategórii a subkategórii by počas prevádzky systému nemal byť dôvod tento zoznam meniť. Z toho plynie, že by sme sa pri návrhu databázy mohli zamerať skôr na otázku *„Aké dotazy sa budú nad databázou vykonávať?“* (namiesto bežnej otázky *„Aké dáta a vzťahy by mala databáza modelovať?“*). S poťahom na entitné množiny Kategória a Subkategória by najčastejšie (a asi aj jediné) možné dotazy boli:

- *Ktoré subkategórie patria pod konkrétnu kategóriu?*
- *Pod ktoré kategórie patrí konkrétna subkategória?*

S cieľom optimalizovať dotazy pri prehľadávaní inzerátov je niekoľko možností, ako kategórie a subkategórie uložiť. Najmenej efektívny spôsob s ohľadom na vykonávané dotazy by bol uloženie vzťahov do pomocnej prepojovacej tabuľky. Optimalizáciu tohoto prístupu by bola možné vykonať použitím materializovaných pohľadov, ktoré by ako odpoveď na databázový dotaz vracali zoznam subkategórii (prípadne kategórii, v závislosti na charaktere dotazu).

Ďalší spôsob je uviesť ku každej kategórii do samostatného stĺpca zoznam podkategórii, ktoré pod ňu spadajú alebo naopak ku každej podkategórii evidovať zoznam kategórii, pod ktoré spadá. Posledné dva spôsoby sú z hľadiska náročnosti na dotazy ekvivalentné a sú charakteristické rovnakou mierou redundancie. Ako lepšia varianta sa javí tá, kde ku každej kategórii bude na pevno uložený zoznam podkategórii. Motiváciou k tomuto riešeniu je fakt, že systém nepredpokladá, že zoznam kategórii a subkategórii bude predmetom zmien a taktiež to, že zpravidla jediný dotaz, ktorý bude nad zoznamom kategórii vykonávaný, bude: *„Ktoré subkategórie patria pod konkrétnu kategóriu?“*.

Počítanie unikátnych zobrazení inzerátu
Ďalej by mal byť systém schopný pre každý inzerát uchovávať počet unikátnych kliknutí za účelom zoraďovania inzerátov. Unikátnym kliknutím je myslené zobrazenie detailu inzerátu z konkrétnej IP adresy. Za účelom uchovávania týchto údajov je vhodné mať v databáze tabuľku, ktorá má len dva stĺpce a to IP adresy návštevníkov portálu a identifikátor inzerátu, na ktorý daný návštevník klikol. Kombinácia IP adresy používateľa a inzerátu by v tomto prípade tvorila kompozitný primárny kľúč, čím by bola zaručená unikátnosť zobrazení inzerátu.

Ostatné uchovávané vlastnosti inzerátov
Okrem stavu predávaného tovaru a dátumu vytvorenia inzerátu je ešte potrebné uchovávať o inzeráte informáciu, či sa jedná o ponuku alebo dopyt a či inzerát nebol nahlásený ako nevhodný, prípadne nevhodne zaradený. Ďalej musí byť v zázname o inzeráte uvedený zoznam referencii na obrázky z prislúchajúcej fotogalérie, link na video z youtube a štruktúrovaný zoznam hierarchicky zoradených zmien v inzeráte. Ak chceme evidovať každú zmenu v inzeráte, opäť máme na výber niekoľko možností. Jednou z nich je zaznamenávať len informáciu o zmene v inzeráte vykonanej používateľom. To by však znamenalo nutnosť rekonštrukcie aktuálneho obsahu inzerátu z uložených zmien a z jeho pôvodného obsahu pri každej vizualizácii zmien. Ďalšou z možností je do zoznamu zmien v inzeráte po každej vykonanej zmene uložiť pôvodnú verziu jeho znenia. Tým by sa značne skrátil čas vizualizácie zmien, avšak za cenu objemu uložených dát. Vzhľadom ku predpokladu, že zmena v znení inzerátu pravdepodobne nebude patriť medzi najčastejšie úkony používateľa, evidovanie jednotlivých verzii polí v inzeráte (namiesto informáciii o zmenách) bude z hľadiska

času vizualizácie zmien prijateľnejšia varianta. Poľom inzerátu je v tomto prípade myslená určitá vlastnosť evidovaná o inzeráte - teda okrem nadpisu a obsahu to môže byť taktiež zaradenie do kategórie, zmena odkazu na video z youtube a podobne.

4.1.2 Správa miest

So správou inizerátov súvisí aj správa miest, keďže každý inzerát bude priradený práve k jednej obci. Pri vyhľadávaní môže používateľ určiť, z ktorej obce by mali byť filtrované inzeráty a taktiež môže určiť, do akej maximálnej vzdialenosti od danej obce vyhľadávať. Z toho plynie nutnosť udržiavať súradnice miest, čo v systéme umožní získať ich vzájomné vzdialenosti.

Z dôvodu presnejšej identifikácie (ako na strane používateľa, tak na strane portálu) bude taktiež vhodné o mestách uchovávať okrem ich názvu aj poštové smerovacie číslo a okres. Podľa poštového smerovacieho čísla bude možné vyhľadávať mestá vo formulári určenom pre filtrovanie inzerátov.

4.1.3 Správa používateľov

Z požiadaviek na vytváraný inzertný portál vyplýva, že pridať inzerát môže len registrovaný používateľ. Registrácia je taktiež podmienkou možnosti reakcie an inzerát, či možnosti hodnotiť iných používateľov. Otázkou je, aké informácie by mali byť v systéme o používateľovi vedené. S cieľom nájsť odpoveď je vhodné rozdeliť informácie na verejné, prístupné len administrátorovi a neverejné. Medzi verejné informácie by mali patriť údaje, ktoré predstavia používateľa ostatným bez možnosti zneužitia údajov (napríklad meno alebo prezývka). Pre ostatných používateľov nemali byť viditeľné údaje, ktoré sú všeobecne považované za citlivé (napríklad číslo účtu). Tieto by ale mali byť viditeľné administrátorovi portálu, nakoľko administrátor bude zodpovedný za prevod platby na správny účet.

Heslo používateľa by v ideálnom prípade nemalo byť dostupné nikomu okrem samotného používateľa. To by sa dalo docieliť (a v praxi je to najčastejšie riešenie) napríklad ukladaním jednosmerne zašifrovaného hesla. Z toho taktiež plynie nutnosť implementácie mechanizmu pre obnovenie hesla v prípade jeho straty. Z tohto dôvodu je nutné, aby systém s používateľom komunikoval taktiež cez iný kanál okrem rozhrania portálu. Preto je vhodné evidovať napríklad e-mailovú adresu používateľa, ktorá by bola taktiež prístupná len administrátorovi. E-mail môže zároveň slúžiť ako unikátna identifikácia používateľa. Ďalším podstatným neverejným údajom je používateľský IBAN. IBAN je vhodný z dôvodu, že sa jedná o štandardizovaný formát identifikácie bankového účtu, ale taktiež preto, že je relatívne jednoduché ho validovať. Ďalšie verejné informácie spojené s používateľom by mohli byť napríklad počet ním uskutočnených nákupov a predajov na portáli, krátky text, ktorým by sa chcel predstaviť ostatným, či dátum registrácie.

Ďalšia informácia, ktorú je nutné o používateľovi evidovať je jeho rola. V prípade, že má používateľ administrátorské práva, bude mať viac oprávnení ako bežný používateľ.

4.1.4 Správa hodnotení

So správou používateľov úzko súvisí aj spôsob, akým budú vytvárané, či ukladané ich hodnotenia. Z požiadaviek na systém plynie, že hodnotenie používateľa iným používateľom by malo byť viazané len k uzavretej používateľskej transakcii (čiže po uzavretí obchodu). S odstránením používateľskej transakcie alebo inzerátu, ku ktorému je viazaná by však nemalo zanikať. Preto budú hodnotenia umiestnené do samostatnej tabuľky a záznamy transak-

cii sa na ne budú odkazovať. Súčasťou hodnotenia používateľa bude číselné hodnotenie od 1 do 5 hviezdičiek a slovný komentár. V rámci jednej transakcie nebude možné uverejniť viac ako jedno hodnotenie, bude však možné hodnotenie upraviť. Okrem komentára by sa mal každý záznam o hodnotení taktiež odkazovať na záznam hodnotiaceho a hodnoteného používateľa.

4.1.5 Správa používateľských transakcii

K jednému inzerátu môže byť otvorených nula až neobmedzene používateľských transakcii. Jedna transakcia by teda mala byť jednoznačne identifikovaná inzerátom a zákazníkom. Zákazník reprezentuje používateľa, ktorý reaguje na inzerát. Týmto používateľom nemôže byť sám inzerent.

Transakcia by mala byť z dôvodu vyhľadávania identifikovaná unikátnyym identifikátorom, ktorý by bol použitý ako variabilný symbol pre prípad, že sa obe zúčastnené strany dohodnú na uzavretí obchodu. Ďalšou uchovávanou informáciou o stave používateľskej transakcie by teda mala byť informácia o tom, ktorá zo strán s obchodom súhlasila. Reprezentovaná bude jedným zo štyroch stavov, a to:

- obchod neodsúhlasený žiadnou zo strán,
- obchod odsúhlasený stranou zákazníka,
- obchod odsúhlasený stranou inzerenta,
- obchod odsúhlasený oboma stranami.

Stav transakcie bude pochopiteľne viditeľný pre obidve zúčastnené strany aj pre administrátora. Zo strany administrácie portálu bude možné pristúpiť k inzerátu, upraviť ho a prípadne aktualizovať stav transakcie v prípade obdržania platby, či odoslania platby na účet používateľa. O tom, na koho účet budú preposlané peniaze kupujúceho rozhoduje podľa požiadaviek na systém kupujúci, avšak administrátor má možnosť skontrolovať správnosť úsudku kupujúceho.

Z vyššie uvedeného plynie, že okrem informácie o tom, ktorá zo strán akceptovala obchod, je potrebné taktiež evidovať údaj o tom, či bol obchod zaplatený a či bol kupujúci spokojný s obdržaným tovarom. Informácia o spokojnosti kupujúceho môže obsahovať jednu z troch hodnôt a to:

- tovar zatiaľ nebol doručený,
- tovar bol doručený a kupujúci **je** spokojný,
- tovar bol doručený a kupujúci **nie je** spokojný.

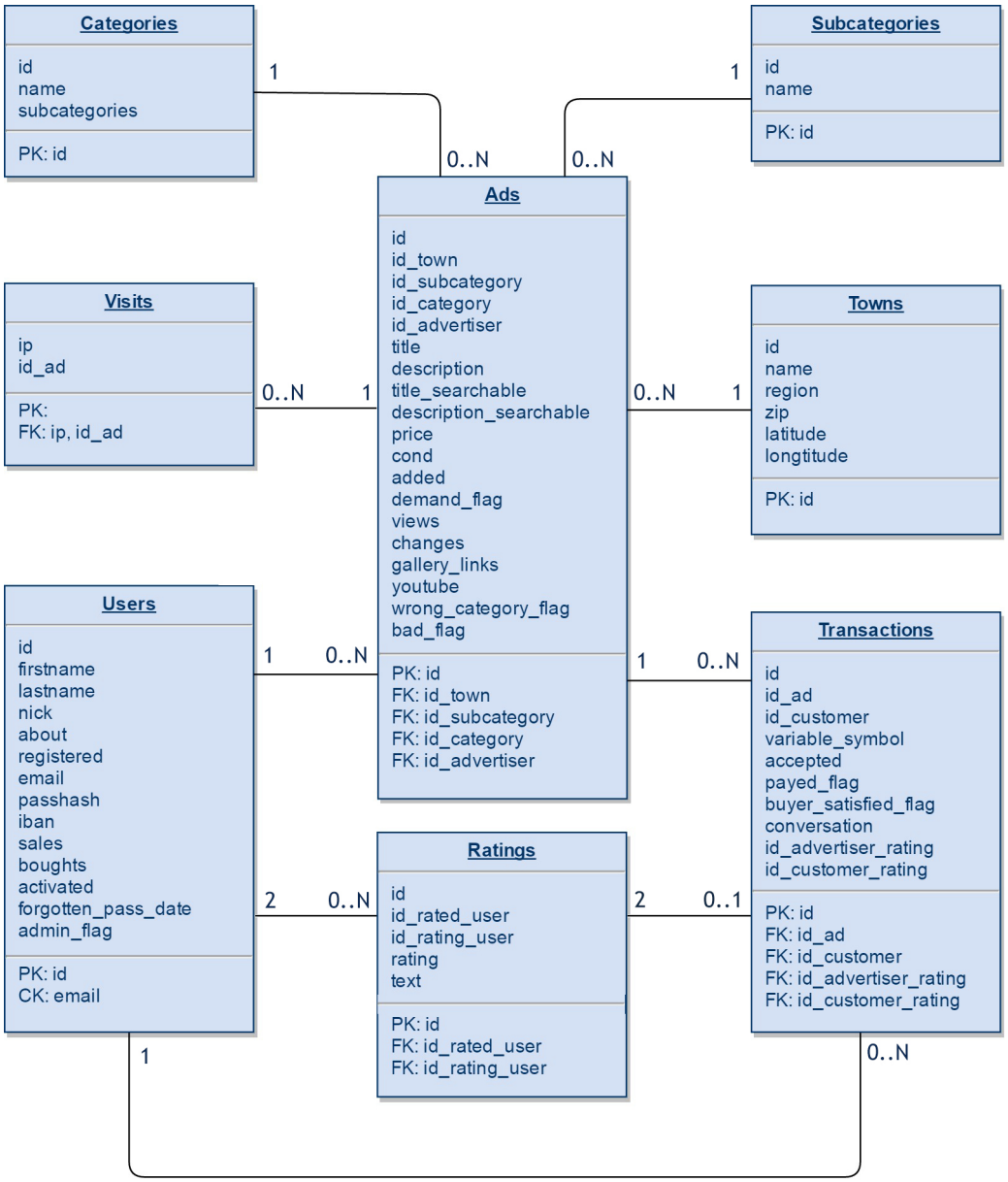
Ako už bolo spomenuté v sekcii o správe hodnotení, súčasťou používateľskej transakcie by mali byť referencie na vzájomné hodnotenia medzi zúčastnenými stranami. Možnosť ohodnotiť druhého používateľa však musí byť podmienená stavom transakcie. Tá musí byť zaplatená a tovar prijatý kupujúcim musí byť ohodnotený.

Posledná významná časť používateľskej transakcie je konverzácia vedená medzi účastníkmi. Z dôvodu transparentnosti konverzácie by systém zúčastneným stranám na seba nemal poskytnúť žiadny priamy kontakt a tak by mala byť celá ich konverzácia realizovaná rozhraním portálu a zaznamenávaná. Taký systém môže byť inšpirovaný riešením, ktoré je

použité pre evidenciu zmien v inzeráte. Tento prístup je dokonca vhodný z hľadiska vizualizácie, nakoľko správy používateľskej transakcie a zmeny v inzeráte sa môžu chronologicky striedať. Je teda možné vizualizovať zmeny v inzeráte rovnakým spôsobom ako jednotivé správy spojené s transakciou.

4.2 Konceptuálny dátový model

Táto podčasť bude venovaná konceptuálnemu návrhu dát. Vodítkom pri návrhu sú jednak požiadavky na systém ale tiež predošlé časti. Z vyššie uvedených predpokladov na obsah a štruktúru dát spravovaných systémom je možné odvodiť nasledujúci ER diagram:



Obr. 4.1: ER diagram

4.3 Používateľské rozhranie portálu

Témou webového používateľského rozhrania je možné sa zaoberať z viacerých hľadísk. V nasledujúcej časti je popísaný návrh užívateľského rozhrania z hľadiska obsahu a rozloženia jednotlivých podstránok, interakcie s používateľom a tiež z hľadiska responzivity systému.

4.3.1 Špecifikácia rozloženia podstránok

Inzertný portál sa ako komplexný systém skladá z viacerých podstránok. Pri návrhu podstránok je vhodné vyjsť z funkcii, ktoré by mal portál vykonávať. Tieto funkcie sú zhrnuté v predošlých častiach a vyplývajú z požiadaviek na systém.

Inzertný portál s ochranou kupujúceho by mal teda obsahovať aspoň týchto 8 podstránok:

- úvodná stránka portálu s formulárom pre vyhľadávanie inzerátov
- zoznam inzerátov s formulárom pre filtráciu inzerátov,
- detail inzerátu,
- používateľská transakcia,
- registračný a prihlasovací formulár,
- detail používateľa,
- formulár pre pridanie inzerátu,
- panel administrácie,
- stránka nenájdená (chybné presmerovanie).

4.3.2 Úvodná stránka portálu

Podľa špecifikácie požiadaviek na systém má byť základnou súčasťou domovskej stránky portálu vyhľadávací formulár. Ten by mal byť doplnený o interaktívnu mapu, ktorá bude slúžiť na selekciu miest, pre ktoré majú byť inzeráty vyhľadávané. Mapa má však slúžiť hlavne ako doplnok vyhľadávacieho formuláru, preto pole pre názov mesta, rovnako ako pole pre vzdialenosť od uvedeného mesta musí byť taktiež súčasťou formulára. Mapa a formulár by mali byť prepojené a automaticky reflektovať zmenu v opačnom prvku používateľského rozhrania (formulár by sa mal aktualizovať po zmene výberu miest na mape a naopak, výber miest na mapke by sa mal aktualizovať po zmene mesta, či vzdialenosti od neho v poliach formulára). Systém bude taktiež poskytovať funkcionalitu pre automatickú lokalizáciu používateľa (podmienenú jeho súhlasom).

Ďalšou súčasťou úvodnej stránky budú interaktívne ikony reprezentujúce kategórie inzerátov, v ktorých je možné vyhľadávať. Táto vlastnosť úvodnej stránky je inšpirovaná existujúcimi inzertnými portálmi. Ikony kategórii budú taktiež slúžiť len ako doplnok formulára. Podobne, ako v prípade miest, bude ekvivalentné pole súčasťou vyhľadávacieho formulára a obsah formulára bude reflektovať používateľom zvolenú ikonu a naopak.
Obsah a rozloženie úvodnej stránky portálu na rôznych zariadeniach je v prílohe B.

4.3.3 Zoznam inzerátov

Po vyhľadaní by mal používateľ vidieť zoznam inzerátov ako výsledkov hľadania v prehľadnom zozname. Zároveň by mal mať kedykoľvek možnosť zmeniť parametre vyhľadávania vo formulári určenom pre filtráciu inzerátov.

Jednotlivé náhľady inzerátov budú organizované do riadkov na šírku obrazovky a budú obsahovať základné informácie o inzeráte vrátane náhľadovej fotografie. Medzi základné

informácie o inzeráte patria informácie, podľa ktorých je možné inzeráty zoradiť a ostatné základné informácie popisujúce inzerát. V náhľade inzerátu budú teda viditeľné tieto položky:

- názov (v prípade limitácie veľkosťou zobrazovacieho zariadenia časť názvu ukončená tromi bodkami),
- obsah inzerátu (alebo jej časť ukončená tromi bodkami),
- meno inzerenta,
- cena,
- dátum pridania,
- počet kliknutí na inzerát.

Filtračný formulár bude umiestnený nad zoznamom inzerátov a bude obsahovať všetky polia, ktoré obsahuje vyhľadávací formulár na úvodnej stránke. Okrem týchto polí bude obsahovať výber subkategórie a možnosť zvoliť parameter zoradenia inzerátov, ktorá bude implicitne nastavená na dátum pridania inzerátu.
Obsah a rozloženie zoznamu inzerátov na rôznych zariadeniach je v prílohe B.

4.3.4 Detail inzerátu

Po výbere inzerátu bude používateľ presmerovaný na jeho detail. Tu by mali byť zobrazené všetky dostupné informácie uvedené k inzerátu v ich plnom znení. Vzhľad detailu inzerátu bude rozdelený do troch hlavných častí a to ľavý panel, obsah a pravý panel.

Ľavý panel slúži pre náhľady podobných inzerátov. Cieľ náhľadu nie je, aby inzerát vynikol ale aby používateľ kliknutím na inzerát s cieľom zobraziť jeho detail nestratil kontext vyhľadávania. V jednotlivých náhľadoch inzerátov v ľavom paneli sú prístupné tieto informácie o inzeráte:

- názov inzerátu,
- cena inzerátu,
- počet kliknutí na inzerát.

Pravý panel obsahuje sumárne informácie o inzeráte teda údaje ako cena, stav ponúkaného tovaru (alebo maximálny stav v prípade, že sa jedná o dopyt), lokalita, meno predajcu a jeho priemerné hodnotenie, dátum pridania a počet zobrazení inzerátu. Pod sumárnymi informáciami by v pravom paneli mali byť panel vedľajších používateľských akcii s inzerátom. Jedná sa o možnosti:

- zdieľať inzerát,
- nahlásiť nevhodný inzerát,
- nahlásiť nevhodnú kategóriu.

Hlavnou časťou podstránky je obsah umiestnený medzi ľavý a pravý panel. Ten bude obsahovať všetky ostatné informácie o inzeráte vrátane interaktívnej fotogalérie a priestoru

pre video z youtube. Na vrchu obsahu bude zvýraznený názov inzerátu nasledovaný fotogalériou. Pod fotogalériou bude samotný obsah inzerátu a náhľad pre video z youtube s možnosťou ho prehrať. V prípade, že prihlásený používateľ je inzerent alebo administrátor, bude vedľa názvu inzerátu možnosť inzerát upraviť alebo zmazať.

Pod obsahom inzerátu bude prihlásený používateľ v prípade, že sa jedná o inzerenta, vidieť zoznam používateľských transakcii otvorených k aktuálnemu inzerátu. V prípade, že sa nejedná o inzerenta bude prihlásený používatľ vidieť pod inzerátom celý priebeh jeho používateľskej transakcie spojenej s aktuálnym inzerátom (prípadne prázdnu) ukončenú možnosťou reagovať na inzerát. V prípade, že používateľ nie je prihlásený, bude na mieste transakcie výzva k prihláseniu. Obsah spodnej časti transakcie s možnosťou reagovať bude generovaný automaticky v závislosti od stavu transakcie a role prihláseného používateľa v danej transakcii (používateľ môže plniť úlohu kupcu alebo predajcu nezávisle na tom, či je inzerent alebo zákazník). Používateľ by totiž mal mať v istých stavoch transakcie možnosť uzavrieť obchod, v inom stave transakcie možnosť odvolať obchod, ohodnotiť prijatý tovar a podobne, preto možné používateľské akcie budú súčasťou spodnej časti transakcie a budú generované individuálne.

4.3.5 Formulár pre pridanie inzerátu

S cieľom pridať inzerát by mal mať používateľ možnosť zvoliť si všetky parametre inzerátu spojené so správnym zaradením inzerátu. Všetky tieto polia sú ekvivalentom vlastností inzerátov viditeľných v jeho detaile. Využitie metódy „ťahaj a pusť“je dnes je už štandardom všetkých moderných aplikácii pracujúcich s nahrávaním súborov cez formulár. Časť formulára vyhradená pre pridávanie fotografii bude tento prístup spolu s alternatívnym tradičným prístupom (s použitím prieskumníka) podporovať.
Obsah a rozloženie formulára pre vytvorenie inzerátu na rôznych zariadeniach je v prílohe B.

4.3.6 Registračný a prihlasovací formulár

Registračný formulár by mal obsahovať polia reprezentujúce informácie uchovávané o používateľovi. Zo vzhľadu formulára by malo byť zrejmé, ktoré polia sú povinné, ktoré sú voliteľné, ktoré informácie budú verejné a ktoré súkromné. V prípade obmedzení na používateľský vstup by mali byť obmedzenia taktiež jasne viditeľné (napríklad „zadajte minimálne 8 alfanumerických znakov“). Z dôvodu predchádzania automatickým registráciám bude po úspešnej registrácii nového používateľa jeho účet deaktivovaný. Aktivácia bude možná kliknutím na link, ktorý úžívateľ v zápätí prijme e-mailom. Je teda v záujme používateľa, či e-mailová adresa, ktorú pri registrácii uviedol skutočne existuje. Súčasťou registračného formulára bude miesto pre podmienky používania portálu. Pod nimi bude vstupný prvok formulára určený pre odsúhlasenie podmienok.

Prihlasovací formulár bude obsahovať len dva polia, a to e-mail používateľa a heslo. Okrem toho bude mať používateľ možnosť požiadať o obnovu hesla. Táto akcia by opäť využila používateľský e-mail, kde by systém automaticky odoslal unikátny odkaz na stránku pre obnovu hesla. Systém teda predpokladá, že zabezpečenie účtu je v najlepšom prípade tak silné, ako je silné zabezpečenie e-mailového konta používateľa.
Obsah a rozloženie registračného formulára na rôznych zariadeniach je v prílohe B.

4.3.7 Detail používateľa

Detail používateľa je možné zobraziť kliknutím na odkaz smerujúci k jeho detailu. Taký odkaz je dostupný ako meno používateľa v používateľskej transakcii, meno inzerenta v detiale inzerátu, meno hodnotiaceho používateľa zozname hodnotení atď.

V prípade detailu používateľa bude podobne, ako pri detaile inzerátu, podstránka rozdelená do troch podčastí. Ľavý panel bude opäť obsahovať zoznam inzerátov s rozdielom, že sa bude jednať o inzeráty používateľa, ktorého detail je práve zobrazovaný. Pravý panel bude obsahovať zoznam hodnotení používateľa. Každé z hodnotení bude obsahovať text hodnotenia, počet hviezdičiek a meno hodnotiaceho používateľa, ktoré bude slúžiť ako odkaz na detail hodnotiaceho používateľa. Obsah umiestnený medzi ľavým a pravým panelom bude obsahovať všetky verejné informácie, ktoré o sebe používateľ uviedol pri registrácii (meno, priezvisko, prezývka a niekoľko slov o sebe). Okrem informácii uvedených používateľom bude podstránka zobrazovať taktiež doterajšie štatistiky používateľa vedené systémom (dátum registrácie, počet uzavretých obchodov a priemerné číselné hodnotenie).

V prípade, že prihlásený používateľ zobrazí svoj vlastný detail, bude mať podobne, ako pri svojich inzerátoch, možnosť svoj profil aktualizovať a doplniť o sebe informácie. Okrem možnosti upraviť profil bude môcť taktiež vykonať odregistráciu svojho účtu. Tá však bude podmienená uzavretím všetkých jeho používateľských transakcii. Právo odregistrovať používateľa bude mať tiež administrátor zobrazujúci detail používateľa.
Obsah a rozloženie podstránky detailu používateľa na rôznych zariadeniach je v prílohe B.

4.3.8 Panel administrácie

Panel administrácie bude dostupný ako položka používateľského menu v navigácii len pre prihlásených používateľov s právami administrátora. Po zobrazení panelu administrácie bude panel rozdelený na tri stĺpce. V ľavom stĺpci bude možné filtrovať používateľov podľa e-mailovej adresy a pridávať, či odoberať im administrátorské práva. V strednom stĺpci bude možnosť filtrovať transakcie podľa variabilného symbolu a upravovať ich stav. Pravý stĺpec bude zobrazovať zoznam nahlásených inzerátov s možnosťou kliknúť na inzerát alebo ho priamo odstrániť zpomedzi zoznamu nahlásených inzerátov.
Obsah a rozloženie panelu administrácie je v prílohe B.

4.3.9 Chybné presmerovanie

K podstránke chybného presmerovania budú viesť všetky URL adresy nerozpoznané systémom a všetky dotazy na server, na ktoré používateľ nemá oprávnenie (pokus zmazať alebo editovať cudzí inzerát, upraviť cudzí profil a podobne). Podstránka chybného presmerovania bude jednoduché oznámenie o neexistujúcej stránke s možnosťou presmerovania na úvodnú podstránku portálu.

4.3.10 Kompatibilita a responzivita

V spojení s webovými informačnými systémami je pojmom kompatibilita myslená kompatibilita webových prehliadačov [13], čo v skratke znamená miera zhody správania a vykreslovania HTML elementov. V záujme prístupnosti systému je najlepší možný prípad, aby sa webové rozhranie správalo rovnako na všetkých zariadeniach a webových prehliadačoch. Podľa štatistík portálu gs.statcounter.com má v čase písania tejto práce majoritný podiel na svetovom trhu prehliadačov Chrome od spoločnosti Google (53.72%). Prehliadače Sam-

sung Internet od firmy Samsung a Edge a Internet Explorer od spoločnosti Microsoft majú na trhu naopak minimálny podiel (v poradí 3.37%, 1.71%, 3.92%). Hoci využitie týchto prehliadačov má klesajúci charakter, stále majú na celkovom trhu značný podiel (spolu 9.0%).

Pod pojmom responzivita webu rozumieme, že sa prvky webovej stránky preusporiadajú a vykreslia podľa veľkosti okna prehliadača tak, aby boli čo najviac optimalizované a komfortne použiteľné na akomkoľvek type zariadenia. [6] Pojmom responzivita sa oplatí zaoberať už len z dôvodu, že v čase písania tejto práce je podľa štatistiky používania zariadení na portáli gs.statcounter.com 51.79% webov navštevovaných z mobilných telefónov a toto číslo ma vzrastajúcu tendenciu. Svojim spôsobom je dnes responzivita už štandardom pri tvorbe webových stránok. To platí nie len z dôvodu použiteľnosti webov ale taktiež z dôvodu optimalizácie umiestnenia webu medzi výsledkami vyhľadávania.

4.3.11 Responzivita inzertného portálu

Ak chceme spraviť návrh portálu responzívnym, na začiatku je nevyhnutné určiť, ktoré elementy by mali byť viditeľné pri akom rozlíšení.

Úvodná podstránka obsahuje tri hlavné časti (vhľadávací formulár, mapa a zoznam kategórii vo forme ikoniek). Pri zobrazení na počítači budú viditeľné všetky tri časti. Pri zobazení na užších obrazovkách (napríklad tablet) môžu byť ikony kategórií skryté a mapa bude mať stále dostatočný rozmer na to, aby s ňou bolo možné pracovať. Pri zobrazení na širokých obrazovkách telefónov (prípadne na telefónoch orientovaných horizontálne) môže byť mapa presunutá pod vyhľadávací formulár, nakoľko stále dáva zmysel ju použiť. Polia formulára budú roztiahnuté na celú šírku obrazovky. V prípade zobrazenia na horizontálne orientovanej obrazovke úzkeho telefónu neostáva na displeji dostatočný priestor pre použitie mapy Slovenska, preto v takom režime dáva zmysel mapu skryť.

Ostatné podstránky obsahujúce formulár budú taktiež na mobilných zariadeniach preusporiadané takým spôsobom, aby vstupné polia formulára zaberali celú šírku displeja a boli zaradené pod seba. Podstránka zobrazujúca zoznam inzerátov bude pri zobrazení na počítači zobrazovať náhľadové fotografie inzerátov vedľa samotného textu inzerátu, zatiaľ čo pri zobrazení na telefónoch a tabletoch bude náhľadová fotografia vyobrazená ako pozadie konkrétneho inzerátu.

V prípade zobrazenia detailu inzerátu alebo používateľa na mobilnom zariadení budú skryté panely obsahujúce podobné inzeráty alebo inzeráty zobrazovaného používateľa. Šírka pravého panelu a hlavného obsahu bude zaberať celú šírku obrazovky a tieto dva celky sa usporiadajú pod seba.

Jednotlivé časti panelu administrácie sa pri mobilnom zobrazení taktiež usporiadajú pod seba a každá z nich sa roztiahne na celú šírku zobrazujúceho zariadenia.

4.3.12 Formuláre a validácia na strane používateľa

Pre validáciu užívateľských vstupov na strane klienta existujú minimálne dva dôvody:

- Používateľ má okamžitú spätnú väzbu od portálu a môže vstup opraviť bez toho, aby čakal na odpoveď servera a musel vstup prepisovať.
- Validáciou na strane klienta sa zníži počet neúspešných (a teda zbytočných) požiadaviek na server, čo má za následok menšiu záťaž servera.

Na strane klienta budú pri registrácii validované vstupy: Meno, Priezvisko, e-mail a IBAN. Počas pridávania inzerátu budú na strane servera validované tieto vstupy: Kombinácia kategórie a podkategórie, názov inzerátu, obsah inzerátu, mesto, cena, stav. Na strane servera budú validované všetky vstupy zadávané prí registrácii aj pri pridávaní inzerátu. Používateľ bude upozornený napríklad na to, že zadal neexistujúci e-mail alebo že sa pokúša k inzerátu pridať link na neexistujúce video.

Zhrnutie
V tejto kapitole boli podrobne definované detaily navrhovaného informačného systému z pohľadu konceptuálneho návrhu dát aj z pohľadu používateľského rozhrania. Ďalšie kapitoly sú venované detailom implementácie portálu.

Kapitola 5

Implementácia portálu

V tejto kapitole je popísaný návrh a postup implementácie jednotlivých modulov systému. Moduly systému sú rozdelené do skupín podľa toho, či patria na stranu klienta alebo na stranu servera.

Pri návrhu aplikácie na strane servera bola použitá architektúra Model View Controller (MVC), ktorá je dnes už štandardom pri vývoji jednoduchých ale aj zložitejších webových aplikáci. Chris Pitt v publikácií [10] uvádza: *„Model View Controller (MVC) sa stáva určujúcou architektúrou frameworkov pre vývoj webových stránok kvôli stabilite, rozšíriteľnosti a predvídateľnosti, ktoré prináša do vývoja. Nepredstavuje len základné oddelenie databázy, funkcie systému a prvkov rozhrania, ale zahŕňa široký rozsah úvah pre tvorenie vysoko výkonných, škálovateľných a bezpečných aplikácii.“*

Na strane servera bol použitý skriptovací jazyk PHP. Rozhodol som sa tak z niekoľkých dôvodov. Prvým z nich je, že jazyk PHP je ľahko použiteľný spolu so značkovacím jazykom HTML a je možné tieto dva jazyky voľne kombinovať. Má jednu z najprehľadnejších dokumentácii na internete a veľkú používateľskú základňu. Okrem toho je veľmi dobre prispôsobený pre jednoduchú prácu s väčšinou používaných databázových systémov [14].
Na strane klienta som použil kombináciu jazykov HTML5, CSS3 a JavaScript.

5.1 Strana servera

Nasledujúca sekcia obsahuje popis detailov významných častí implementácie na strane servera. Pozornosť je obzvlášť venovaná návrhu architektúry, zvolenému databázovému systému a bezpečnosti.

5.1.1 Objektovo orientovaný návrh

V záujme udržiavateľného kódu a jednoduchej manipulácie s dátami som zvolil pre vývoj systému objektovo orientovaný návrh. Z toho plynie nutnosť odlíšiť objekty v navrhovanom systéme. V informačnom systéme založenom na arhcitektúre MVC sa štandardne rozlišujú minimálne tieto tri kategórie objektov:

- model,
- kontrolér,
- entita.

Modely
Objekty patriace do kategórie Model (modely) štandardne slúžia ako rozhranie pre prístup k databáze. Ich úlohou je namapovať dáta z databázy (entity) na objekty kategórie Entita a naopak, objekty kategórie Entita premapovať na dáta v databáze. Tu sa dostávame k pojmu objektovo-relačného mapovania (popísaný ďalej). Z tohoto dôvodu som triedy modelov vo svojej práci odlíšil príponou „Mapper“k ich názvu.

Entity
S modelmi úzko súvisia objekty typu Entita. Entity sú objekty, ktoré by mali svojim charakterom (atribútmi a funkciami) reflektovať konkrétne záznamy v databáze. Sú vstupom pre funkcie v objektoch triedy Model a sú taktiež ich výstupom. Výsledkom tohoto prístupu je, že aplikácia počas celého jej behu pracuje výlučne s objektami.

Kontroléry
Poslednou významnou triedou sú objekty z triedy Kontrolér. Ich hlavnou úlohou je spracovať dotaz na server, vytvoriť správne inštancie modelov, získať dáta z databázy a správne ich vizualizovať pomocou šablón. Šablónou je myslený predpripravený fragment HTML kódu s miestami určenými pre náhradu spracovanými dátami z databázy. Vizualizácia dát však nie je len prostá náhrada textu v HTML kóde. Je ňou taktiež spracovanie dát, ich prípadná úprava a odoslanie HTTP odpovede s vygenerovaným HTML telom klientskej strane. Príkladom môže byť hodnotenie používateľa. Model z databázy vráti okrem iných dát číselné hodnotenie s hodnotou 3 (3 hviezdičky). Kontrolér neodošle klientovi číslo 3, ale 3 vykreslené hviezdičky v HTML kóde. Každý objekt z triedy kontrolér má teda svoje atribúty reprezentujúce prislúchajúcu HTML šablónu a dáta, ktoré pomocou šablóny vizualizuje.

5.1.2 Objektovo-relačné mapovanie

Súčasťou filozofie MVC architektúry je oddelenie databázy a databázových dotazov od riadiacej logiky systému. K tomu slúži mechanizmus objektovo-relačného mapovania. Jedná sa o mechanizmus, ktorý umožňuje adresovať, sprístupňovať a manipulovať s objektmi bez nutnosti mať povedomie o tom, ako sú dané objekty spojené so zdrojom ich dát [12].

S cieľom písať udržovateľný kód a zachovávať jednotné rozhranie pre prácu s dátami v databáze je prístup objektovo-relačného mapovania vhodný. Keď objekt typu model vykoná dotaz nad databázovým systémom, s výsledkom dotazu sa nepracuje ako s poľom ale ako s objektom. K tomu je nutné zapúzdriť riadky výsledku dotazu do entít, ktorých členské premenné zodpovedajú jednotlivým stĺpcom mapovanej tabuľky.

5.2 Použité databázové systémy

Pre Takmer pre celú aplikáciu som použil relačný databázový systém MySQL. Vzhľadom k charakteru aplikácie a spravovaným dátam je tento prístup vhodnejšou variantou ako NoSQL technológie. S databázovým systémom MySQL je taktiež jednoduché pracovať v jazyku PHP, nakoľko jazyk PHP poskytuje viacero moderných a bezpečných knižníc, ktoré abstrahujú beh aplikácie od samotného databázového systému.

Mestá s ich súradnicami som uložil okrem MySQL databázy taktiež do samostatnej databázy MongoDB, ktorá je jedným z hlavných predstaviteľov NoSQL databázových systémov. Motiváciou k využitiu NoSQL technológie je schopnosť MongoDB databázy spravovať dáta reprezentujúce body v dvojrozmernom priestore a vykonávať dotazy nad nimi. Data-

báza obsahuje takzvaný geopriestorový doplnok („geospatial feature“), pomocou ktorého je schopná počítať vzdialenosti medzi uloženými bodmi z ich súradníc vo vormáte latitude a longtitude. Táto funkcionalita je vhodná pre získanie zoznamu miest v určitom okolí bodu určeného geografickými súradnicami.

5.2.1 Šifrovanie dát

Jediný údaj, ktorého zverejnenie by mohlo používateľa reálne poškodiť je jeho heslo. Je teda nevyhnutné ho v databáze ukladať zašifrované. Šifrovacie algoritmy sa v základe rozdeľujú na dva typy a to obojsmerné a jednosmerné. V prípade obojsmerných šifrovacích algoritmov je možné získať zo šifry (zašifrovaného pôvodného reťazca, taktiež nazývaného „hash“) pôvodný šifrovaný reťazec. V prípade jednosmerných tomu tak nie je a teda neexistuje spôsob, ako zo šifry rekonštruovať pôvodný reťazec. Pre ukladanie hesla je teda vhodnou alternatívou jednosmerný šifrovací algoritmus, nakoľko najbezpečnejšie uložené heslo je také, ktoré nevie prečítať ani vlastník portálu.

Za účelom jednosmerného šifrovania je dnes k dispozícii množstvo kryptovacích algoritmov, z ktorých niektoré sa stávajú zastaralými, iné sú preferovanými. Aktuálne sa za najbezpečnejší jednosmerný šifrovací algoritmus považuje „`CRYPT_BLOWFISH`“. Jazyk PHP poskytuje pre šifrovanie (alebo hashovanie) funkciu `password_hash()`. Funkcia automaticky vygeneruje tzv. soľ a pomocou aktuálne najbezpečnejšieho šifrovacieho algoritmu (v čase písania tejto práce je to `CRYPT_BLOWFISH`) jednosmerne zašifruje heslo, ktoré dostane ako vstupný parameter. Soľ je náhodne generovaný reťazec, ktorý je pripojený k šifrovanému heslu pred zašifrovaním. Doposiaľ bolo zvykom soľ pridávať a ukladať manuálne, no podľa manuálu k funkcii `password_hash()` je táto metóda zastaralá a odporúča sa nechať si reťazec označovaný ako soľ vygenerovať funkciou automaticky.

5.2.2 Ochrana pred útokom formou SQL injekcie

Ďalšou oblasťou bezpečnosti je ochrana databázy pred útokmi fromou SQL injekcie. *„Útoky formou SQL injekcie predstavujú vážnu bezpečnostnú hrozbu pre Webové aplikácie, pretože dovoľujú útočníkom získať neobmedzený prístup ku databázam, ktoré tvoria základ aplikácií a k potenciálne citlivým informáciám, ktoré tieto databázy obsahujú“* [5]. Definícia SQL injekcie podľa publikácie **Advanced SQL injection in SQL server applications** z roku 2002: *„SQL Injekcia nastáva, keď je útočník schopný vložiť sled SQL príkazov do „dotazu“manipuláciou dátovým vstupom do aplikácie“* [4]. Jazyk PHP pre prácu s databázou poskytuje minimálne dve moderné a bezpečné vrstvy abstrahujúce databázový systém. Sú to aplikačné rozhranie „MySQL Improved Extension“, ktoré by malo nahradiť pôvodne používané rozhranie „Original MySQL API“. To sa stalo zastaralým a vo verzii PHP 7.0.0 bolo odstránené. Druhou variantou je použitie vstavaného ovládača pre prácu s MySQL databázou „MySQL Functions (PDO_MYSQL)“. V mojej práci som použil práve druhú variantu pre jej jednoduchosť a univerzálnosť. Zabrániť SQL injekcii je v nej možné kombináciou funkcionality takzvaných „Pomenovaných parametrov“a „Pripravených dotazov“. To v praxi znamená, že s použitím tohto ovládača je možné pred vykonaním databázového dotazu najprv dotaz skompilovať a potom vykonať. Skompilovaný dotaz obsahuje pomenované parametre, ktoré sa pri vykonávaní dotazu nahradia skutočnými hodnotami parametrov. Výhoda tohto prístupu je, že v prípade, že sa ten istý databázový dotaz vykonáva viackrát s rôznymi parametrami, je skompilovaný len raz, čo celkovo skracuje čas práce s databázou.

Príklad ochrany pred SQL injekciou Uvažujme nasledovnú situáciu. Chceme z data-

bázy vybrať záznam o používateľovi, ktorého identifikátor sme prijali z formulára. Prijatý identifikátor systém uloží do premennej s názvom `$userId`. SQL dotaz by teda znel napríklad takto:

```
$sql = 'SELECT * FROM Users WHERE id = ' + $userId;
```

V prípade, že útočník zadá do vstupného poľa formulára hodnotu `"105 OR 1=1"`, bude prirodzene výsledok takého dotazu celá databáza používateľov:

```
$sql = 'SELECT * FROM Users WHERE id = 105 OR 1=1';
```

Pomocou PDO ovládača a pomenovaných parametrov by sme databázový dotaz vykonali nasledovným spôsobom:

```
$sql = 'SELECT * FROM Users WHERE id = :user_id';

$stmt = $dbh->prepare($sql);

$stmt->execute(['user_id' => $userId]);
```

V tomto prípade bude dotaz najprv skompilovaný a nechá sa v ňom miesto pre pomenovaný parameter, ktorý do dotazu vložíme cez funkciu `execute()` ako prvok asociatívneho poľa (vstupný parameter funkcie `execute()`). V prípade pokusu o útok bude výsledok rovný výsledku nasledovného dotazu:

```
$sql = "SELECT * FROM Users WHERE id = '105 OR 1=1'";
```

Vidíme, že parameter je vložený do databázového dotazu ako reťazec, čím nám jazyk PHP pomáha brániť sa SQL injekciám a zároveň (v prípade opakovaného vykonávania toho istého dotazu) optimalizuje dobu prístupu k databáze.

5.3 Validácia

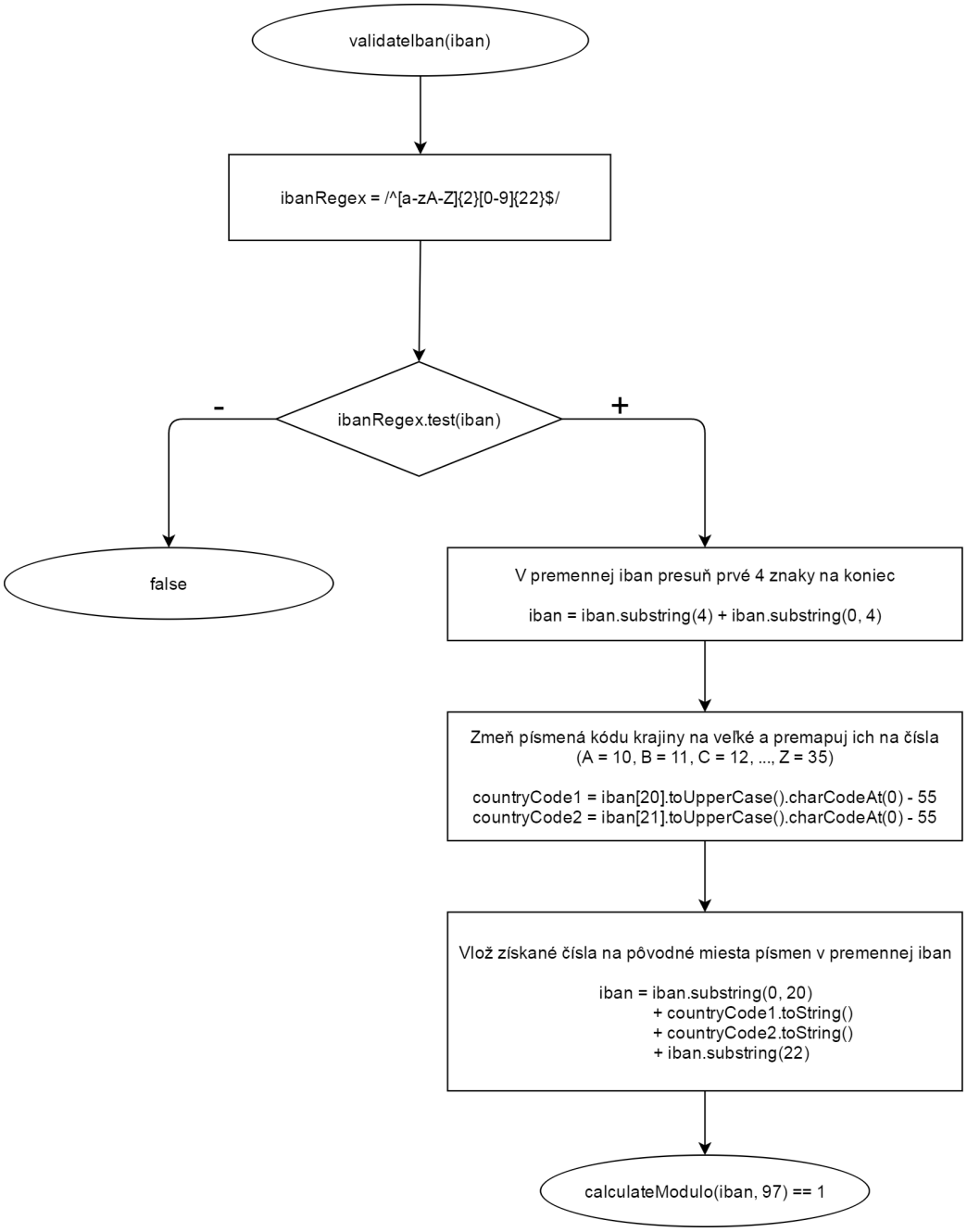
Pred uložením do databázy je nutné vkladané dáta z bezpečnostných dôvodov validovať aj na strane servera. Ak chceme označiť niektorý zo vstupov za validný, je najskôr potrebné určiť parametre, na základe ktorých chceme vstup validovať. Vo väčšine prípadov sa jedná štandardne o povinný alebo nepovinný vstup, vstup obsahujúci výlučne písmená, vstup obsahujúci e-mailovú adresu alebo číselný vstup. Medzi neštandardné typy vstupov patria napríklad heslo alebo IBAN používateľa. Pre validáciu väčšiny vstupov som použil vstavané funkcie jazyka PHP:

- `ctype_alpha()` pre kontrolu, či vstup obsahuje len písmená (napríklad validácia mena a priezviska pri registráii)
- `is_numeric()` pre kontrolu číselnej hodnoty
- `count()` pre kontrolu dĺžky reťazca
- regulárne výrazy (napríklad pre validáciu zložitosti hesla)

Pre validáciu e-mailu som v systéme použil kombináciu funkcii `filter_var()`, pomocou ktorej je možné zvalidovať formát e-mailovej adresy a `checkdnsrr()`, pomocou ktorej je možné overiť, že doména, na ktorej je e-mailová adresa skutočne existuje.

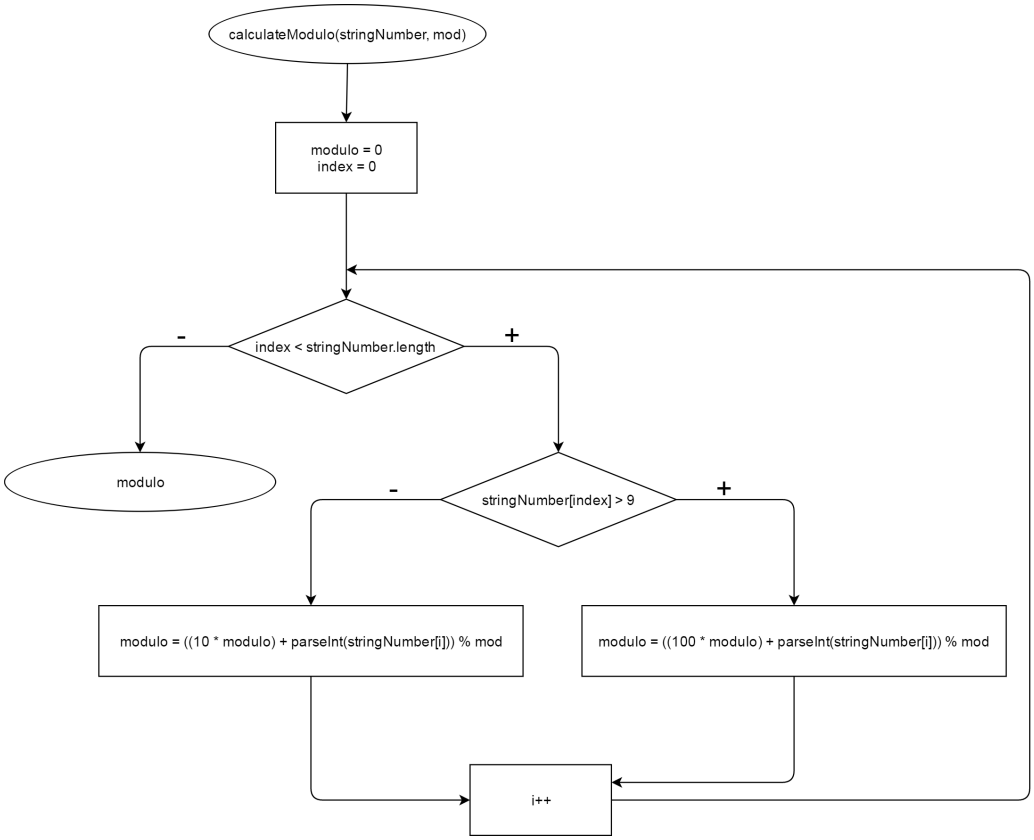
Celkom špecifický bol postup validácie vstupného poľa IBAN (Medzinárodné číslo bankového účtu). Nástroj pre validáciu slovenských čísel IBAN som implementoval na základe špecifikácie jeho formátu. Slovenský IBAN má z pravidla 24 znakov, z ktorých prvé dva sú SK (alebo CZ v prípade Českej varianty). Nasledujú 2 čísla kontrolného súčtu, 4 čísla reprezentujúce kód banky, 6 číslic predstavujúcich predčíslie účtu a 10 číslic samotného čísla účtu. Vzor validného slovenského IBANu je napríklad `SK68 0720 0002 8919 8742 6353`.

Podľa zdroja [2], kde sú popísané jednotlivé kroky validácie IBANu, som implementoval jednoduchý validátor, ktorým je používateľský vstup validovaný na strane klienta aj na strane servera. Algoritmus validácie vyjadruje nasledovný diagram:



Obr. 5.1: IBAN validátor

Súčasťou validácie IBANu je nutnosť počítať zvyšok po delení čislom 97 z čísla, ktoré má veľkosť od 65 do 128 bitov. Ani jeden z jazykov Javascript alebo PHP bez prídavných modulov nemajú možnosť uchovať 128-bitovú celočíselnú premennú. Preto je súčasťou validácie aj modul pre výpočet zvyšku po delení z ľubovoľne veľkého celého čísla číslom menším ako 100. Vstup musí byť teda uložený v reťazcovej premennej. Nasledujúci diagram ilustruje priebeh tejto pomocnej funkcie:



Obr. 5.2: výpočet zvyšku po delení

5.4 Strana klienta

Strana klienta (front-end) bola implementovaná pomocou kombinácie jazykov HTML, CSS a Javascript. Venovať pozornosť návrhu klientskej časti aplikácie je dôležité minimálne z niekoľkých dôvodov a to:

- optimalizácia z hľadiska výkonu,
- optimalizácia z hľadiska sieťovej komunikácie,
- optimalizácia pre rôzne typy zariadení.

5.4.1 Optimalizácia kódu na strane klienta

Podľa článku [9] spolu prvé dva z uvedných bodov úzko súvisia. Optimalizovať výkon webovej aplikácie je možné dodržiavaním niekoľký zásad, ktoré sú považované za nepísané pravidlá pri tvorbe webových aplikácii. Patrí medzi ne snaha o čo najmenší objem stránky, minimalizácia počtu stiahnutých súborov, správne umiestnenie referencii na skripty a dokumenty popísujúce kaskádové štýly (CSS).

Pri návrhu portálu som sa snažil čo najviac obmedziť objem stránky, preto sú na nej prítomné len elementy, ktoré sú naozaj využívané. Zmenšenie rozsahu HTML kódu urýchľuje okrem sťahovania stránky tiež prácu jazyka Javascript s DOM štruktúrou. Stiahnuté sú len súbory (obrázky a skripty), ktoré aktuálne zobrazená stránka skutočne využíva. To

je docielené implementáciou na strane servera. Skripty sú sťahované až po načítaní celej webstránky, čo je docielené umiestnením referencií na skripty na koniec tela stránky. Umiestnenie referencii na CSS súbory na začiatok stránky urýchľuje vykreslovanie elementov stránky (napríklad z dôvodu, že obrázky na stránke majú vopred stanovené rozmery a stránka sa môže vykresľovať paralelne s ich sťahovaním).

Optimalizácia Javascript kódu
Aj fragmenty Javascript kódu je možné niekoľkými spôsobmi optimalizovať. Prvá a najjednoduchšia forma optimalizácie je odstrániť časti Javascript kódu na miestach, kde je možné pre ten istý účel použiť možnosti technológií CSS3. Takými miestami sú napríklad jednoduché animácie vlastností elementov na stránke. Základ tejto optimalizácie spočíva v tom, že ak sú webové animácie v prostredí prehliadača natívne podporované, tak animácie založené na CSS sú typicky spracovávané na vláknach nazývaných kompozičné („compositor thread“). Tie sú odlišné od hlavného vlákna prehliadača, ktoré sa stará o štýlovanie, rozmiestnenie, vykresľovanie a vykonávanie Javascript kódu. To znamená, že ak v prehliadači beží nejaká náročná úloha na hlavnom vlákne, tieto animácie môžu byť vykonávané bez prerušenia [7]. Architektúra používajúca kompozitné vlákna tvorí základ vykresľovacieho modulu v moderných prehliadačoch. Detailný pohľad na ich účel a fungovanie je však nad rámec tejto práce. Ďalšou možnosťou optimalizácie Javascript kódu je ukladanie často používaných medzivýsledkov (napríklad výsledkov prehľadvania DOM štruktúry dokumentu) do premenných.

5.4.2 Optimalizácia zobrazenia na rôznych zariadeniach

Web by mal byť podľa požiadaviek na systém zobraziteľný na všetkých zariadeniach. Pre organizáciu štruktúry stránky bol použitý štandard HTML5 vrátane moderných významových HTML značiek, čo vedie k stručnejšiemu a čistejšiemu kódu. HTML elementy sú na stránke rozmiestnené tak, aby bolo možné s nimi manipulovať v prípade zobrazenia na ľubovoľnom type obrazovky. Rôzne rozlíšenia a pomery strán sú pokryté CSS dotazmi na zobrazovacie zariadenie („media queries“). Správne zobrazenie na všetkých typoch zariadení a responzívny návrh je teda vyriešený čisto pomocou funkcionality CSS3.

5.4.3 Validácia na strane klienta

V prípade štandardných prehliadačov je do istej miery možné nasmerovať používateľa validáciou jeho vstupu. Štandard HTML5 podporuje isté druhy validácie hodnôt (napríklad validácia e-mailovej adresy, validácia číselného vstupu, maximálneho alebo minimálneho počtu znakov a podobne). V prehliadačoch, ktoré nepodporujú validácie HTML formulárov je možné validovať vstupy pomocou validačných funkcii v jazyku Javascript. To je do istej miery aj nutné, keďže niektoré špecifické hodnoty validácie vyžadujú zložitejší systém validácie (napríklad heslo alebo IBAN). Pre validáciu na strane klienta sú použité tie isté validačné algoritmy ako na strane servera. Správne fungovanie portálu taktiež predpokladá, že používateľ nemá v prehliadači vypnuté vykonávanie Javascript kódu.

Zhrnutie
V tejto kapitole boli podrobne preberané detaily týkajúce sa implementácie systému spolu s motiváciou k uvedeným riešeniam. Ďalšia kapitola je venovaná zoznamu použitých technológii, spôsobu ich integrácie a taktiež motivácii k ich použitiu.

Kapitola 6

Použité technológie

V tejto kapitole budú podrobne rozvedené všetky moduly inzertného portálu a popísané technológie použité pri ich implementácii vrátane motivácie k ich použitiu. Technológie použité pri tvorbe systému sa opäť dajú rozdeliť do dvoch kategórii a to technológie na strane servera a technológie na strane klienta.

6.1 Technológie na strane servera

Po návrhu architektúry a výbere programovacieho jazyka som mal na výber jednu z dvoch možností:

1. Naprogramovať systém od základu vrátane každého modulu,
2. Postaviť jadro aplikácie na niektorom z existujúcich PHP frameworkov.

Prvá možnosť predstavuje pracnú činnosť, z istého uhla pohľadu aj znovuvynaliezanie už objavených techník a hlavne neistý koniec, ktorý môže v prípade zlého počiatočného návrhu viesť k neprehľadnému, nebezpečnému, či neefektívne navrhnutému kódu.

Druhá možnosť predpokladá istý čas venovaný štúdiu zvoleného frameworku, jeho filozofie a predpokladanej štruktúry vytvorenej aplikácie. Nevýhodou tejto možnosti môže byť robustnosť a všestrannosť zvoleného frameworku. To môže znamenať množstvo prídavnej funkcionality, ktorá nemusí byť nikde vo výslednom systéme využitá, avšak bude prítomná. Správne použitie frameworku by však pravdepodobne vyústilo k produkcii čistého a udržovateľného kódu písaného podľa zaužívaných a známych štandardov.

6.1.1 Mikroframework Slim

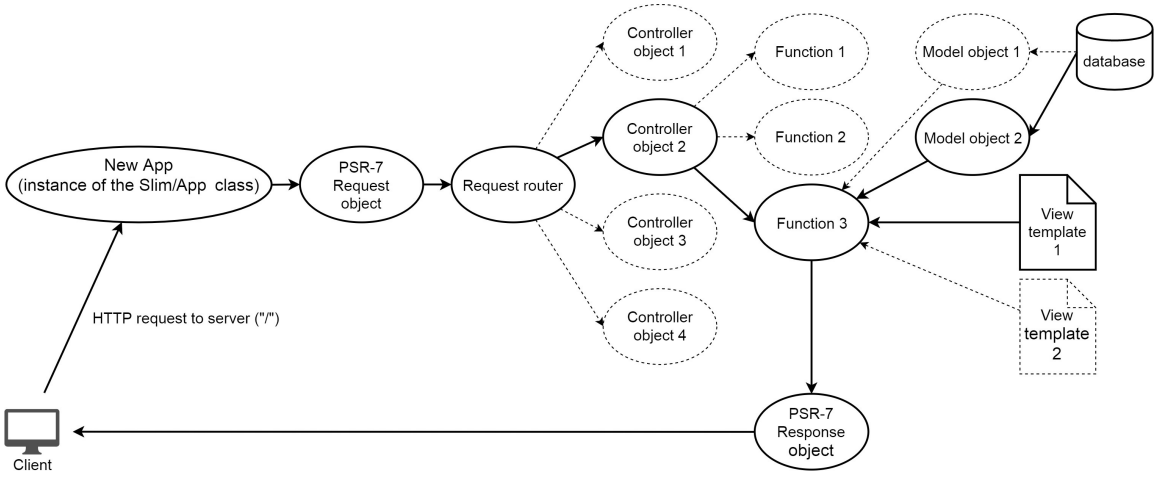
Ako kompromis medzi dvomi vymenovanými riešeniami som pre implementáciu systému zvolil mikroframework Slim. Motiváciou pre toto rozhodnutie bola hlavne jeho modularita. Slim ako odľahčený mikroframework vytvára len základ pre ostatné moduly, ktorých výber je ponechaný na preferenciách vývojára. Slim má vstavanú len základnú sadu funkcii pre prácu s HTTP dotazmi na server a odpoveďami klientovi. Podľa dokumentácie na oficiálnej stránke mikroframeworku má Slim rýchly a dobre optimalizovaný smerovací systém (Request router) [8].

6.1.2 Jadro mikroframeworku

Slim framework, na rozdiel od ostatných frameworkov nedisponuje žiadnym mechanizmom pre objektovo-relačné mapovanie, žiadnou vstavanou funkcionalitou pre prácu s pohľadmi, žiadnymi funkciami pre validáciu a podobne. Podľa dokumentácie ho tvoria len štyri hlavné celky a to:

- Request router,
- Implementácia rozhrania PSR-7 ServerRequestInterface pre prácu s dotazmi na server,
- Implementácia rozhrania PSR-7 ResponseInterface pre prácu s odpoveďou servera klientovi,
- Kontajner závislostí, ku ktorým môže pristupovať každá trieda s prístupom ku objektu aplikácie.

Tok aplikácie od obdržania dotazu až po návrat odpovede klientovi je ilustrovaný na nasledujúcom diagrame:



Obr. 6.1: Životný cyklus Slim aplikácie

6.1.3 Rozhranie PSR-7

Rozhranie PSR-7 predstavuje štandard pre abstrakciu správ HTTP protokolu pre PHP aplikácie [11]. Tento štandard je používaný väčšinou moderných PHP frameworkov.

Implementáciu rozhrania PSR-7, ktorá je vstavaná vo Frameworku Slim je možné nahradiť ľubovoľnou inou (prípadne vlastnou) implementáiou. Pri práci s HTTP správami budeme s ohladom na tok aplikácie rozlišovať dve kategórie správ:

- Dotaz (PSR-7 Request Object),
- Odpoveď (PSR-7 Response Object).

PSR-7 Request objekt
Pri konštrukcii PSR-7 Request objektu je spracovávaná HTTP správa transformovaná do ekvivalentnej objektovej reprezentácie. Novo vzniknutý objekt potom disponuje niekoľkými prístupovými metódami, ktoré robia prácu s protokolom HTTP omnoho jednoduchšou.

Výber najpoužívanejších verejných prístupových metód pre PSR-7 objekt typu Request:

- `isGet()` pre identifikáciu metódy HTTP dotazu,
- `isPost()` pre identifikáciu metódy HTTP dotazu,
- `getUri()` pre získanie objektu reprezentujúceho URI parameter HTTP dotazu. Ten opäť disponuje niekoľkými funkciami pre identifikáciu vlastností URL (napríklad parametrov, fragmentov a podobne),
- `getParsedBody()` pre získanie spracovaného tela dotazu. Výsledkom je asociatívne pole dvojíc názov a hodnota (napríklad z obsahu formulára),
- `getUploadedFiles()` pre získanie prístupu k súborom odoslaným cez formulár.

PSR-7 Response objekt
HTTP správa typu Response je rozhraním Slim aplikácie vytvorená automaticky pri jej spustení. Ak je k prijatému HTTP dotazu priradená obslužná funkcia (pomocou registrovaných ciest v module Request router), je pri jej volaní automaticky vytvorený aj prázdny objekt typu PSR-7 Response, ktorý je obslužnej funkcii predaný ako parameter. Vo funkcii je teda možné priamo pristupovať k Response objektu a cez jeho prístupové funkcie mu nastaviť obsah jeho tela, kód stavu (napríklad `200`, `404`, `302`...) a tak ďalej. Response objekt je pri výstupe z obslužnej funkcie automaticky vrátený klientovi.

Výber najpoužívanejších verejných prístupových metód pre PSR-7 objekt typu Response:

- `getStatusCode()` pre identifikáciu kódu stavu odpovede,
- `withStatus(302)` pre nastavenie kódu stavu odpovede,
- `withHeader('Content-type', 'application/json')` pre nastavenie atribútu v hlavičke HTTP správy,
- `withAddedHeader('Allow', 'PUT')` pre pridanie atribútu do hlavičky HTTP správy,
- `withoutHeader('Allow')` pre odstránenie atribútu z hlavičky HTTP správy,
- `getBody()` pre prístup k telu HTTP správy,
- `getBody()->write()` pre zápis do tela HTTP správy,
- `withJson()` pre nastavenie dát vo formáte JSON telu HTTP správy. Vstupný argument prístupovej funkcie je asociatívne pole, ktoré je automaticky prevedené do formátu JSON a serializované. Táto metóda sa najčastejšie používa v prípade implementácie REST servera.

6.1.4 Kontajner závislostí

Počas vytvárania inštancie Slim aplikácie je aplikácii sprístupnený takzvaný kontajner závislostí. Kontajner závislostí je modul frameworku Slim, ktorý zhrňuje a robí pre aplikáciu prístupnými jej závislosti. Za závislosť aplikácie sú považované služby, moduly, parametre konfigurácie a všetky ostatné súčasti systému, ku ktorým je vhodné mať počas celej doby behu aplikácie prístup. Takými závislosťami sú aj samotné PSR-7 Request a Response objekty, ale taktiež napríklad inštancia PDO objektu pre prístup k databáze, modul pre

zaznamenávanie behu aplikácie (logovanie udalostí), informácie o používateľskej relácii (session) a podobne. V kontajneri závislostí aplikácie Slim sú tieto závislosti injektované pred samotným vytvorením inštancie aplikácie. Kontajner je potom predaný konštruktoru aplikácie Slim, ktorá potom môže počas celého svojho behu k týmto závislostiam ľubovoľne pristupovať.

Tento prístup prispieva k čistote kódu, nakoľko inštanciu jednej závislosti spraví prístupnou pre všetky ostatné moduly aplikácie a nie je potrebné vytvárať samostatnú inštanciu pre každé jej použitie. To je vhodné napríklad v prípade inštancie PDO objektu, ktorý nie je potrebné počas behu aplikácie vytvárať odznova pred každým dotazom na databázový systém. Kontajner závislostí však nie je úplne vhodné používať pre moduly, ktorých použitie je špecifické pre úzku skupinu úloh (napríklad šifrovanie dát). Z logických dôvodov nedáva zmysel pri každom spustení aplikácie vytvárať inštanciu triedy zabezpečujúcej šifrovanie a dešifrovanie dát, ak nutnosť šifrovať, či dešifrovať dáta nastáva len pri registrácii a prihlásení používateľa (s cieľom zašifrovať používateľom zadané heslo). Preto moduly, ktoré sú špecifické len pre úzku sadu úloh je vhodné vytvoriť počas behu aplikácie a optimalizovať tak dobu vytvárania Slim aplikácie.

6.1.5 Prepojenie modulov aplikácie

V návrhu portálu inšpirovanom MVC architektúrou tvoria hlavnú časť kontroléry. Kontroléry sú inštancie triedy Controller, ktorých hlavnou úlohou je získať a pripraviť dáta z databázy, pripraviť šablónu pohľadu a pomocou nej vizualizovať dáta používateľsky prívetivým spôsobom a odoslať vizualizované dáta v HTTP odpovedi klientovi. V aplikácii inzertného portálu som triedy kontrolérov organizoval podľa tabuliek v databáze, s ktorými kontrolér pracuje. Najvýznamnejšie triedy odvodené od triedy Controller, ktoré v implementácii inzertného port8lu rozlišujem sú:

- `Ads` (pre prácu s inzerátmi),
- `Authentication` (pre prácu s registráciou, prihlasovaním a správou hesiel),
- `Transactions` (pre prácu s používateľskými transakciami),
- `Users` (pre prácu s používateľskými transakciami),
- `Towns` (pre prácu so zoznamom miest),
- `Restful` (pre prácu s asynchrónnymi dotazmi klienta, ktoré nepredpokladajú aktualizovanie celej zobrazovanej podstránky, ale len jej časti).

Každá trieda odvodená od triedy Controller má implementované svoje vlastné funkcie s individuálnym určením v závislosti od HTTP dotazu. To, ktorá funkcia kontroléra je pri prijatí HTTP dotazu volaná je definované práve v module Request router Slim aplikácie. Úlohou tohoto modulu je mapovanie HTTP dotazov zodpovedajúcich určitému vzoru na konkrétne verejné funkcie kontrolérov. Pred zavolaním zvolenej obslužnej funkcie kontroléra je vytvorená inštancia daného kontroléra, ktorej je predaná referencia na inštanciu bežiacej aplikácie. Tým je zabezpečený prístup inštancie vytvoreného kontroléra ku celému kontextu aplikácie vrátane kontajnera závislostí. Následne je volaná funkcia určená v module Request router, ktorej sú ako parametre predané inštancie objektov PSR-7 Request aj PSR-7 Response popísaných vyššie.

6.1.6 Šablónovací nástroj Twig

Šablónovacím nástrojom rozumieme softvér, ktorý kombinuje šablóny a dáta s cieľom vytvoriť výsledný dokument [3]. V architektúre MVC má tento nástroj uplatnenie vo vrstve pohľadov (View). Mikroframework Slim v základe žiadnym takým nástrojom nedisponuje. Je však jednoduché doň akýkoľvek modul pridať s použitím systému Composer, ktorý predstavuje správcu závislostí pre PHP aplikácie. Výber šablónovaieho nástroja je ponechaný na preferenciách programátora. Je taktiež možné ho úplne vynechať a injektovať fragmenty PHP kódu priamo do HTML kódu. Pri rozhodovaní, ktorý šablónovací nástroj použiť je na výber z niekoľkých najpoužívanejších:

- twig,
- smarty,
- blade.

Každý z vymenovaných robí funkčný kód injektovaný do HTML kódu prehľadnejším tým, že definuje vlastnú čitateľnejšiu syntax, z ktorej sú potom šablóny prekompilované do natívneho PHP kódu. Pri tvorbe systému som sa rozhodol pre šablónovací nástroj Twig, pretože je schopný zapisovať vykresľovať odpoveď priamo do inštancie triedy PSR-7 Response. Je teda jednoducho integrovateľný s frameworkom. Ďalším dôvodom je, že integrácia Twig-u a mikroframeworku Slim je dobre dokumentovaná na oficiálnej stránke mikroframeworku. Posledným dôvodom pre voľbu šablónovacieho nástroja Twig bolo, že umožňuje fragmentáciu a ľubovoľné zanorovanie šablón, čo robí vizualizáciu dát oveľa pohodlnejšou a vedie k lepšej udržovateľnosti a znovupoužiteľnosti kódu.

6.1.7 Práca s e-mailami

Súčasťou funkcionality vytváraného inzertného portálu je možnosť nechať si odoslať celý doterajší priebeh používateľskej transakcie na e-mail. Pre tento účel, ale taktiež z dôvodu registrácie, či pomoci pri obnove hesla je nutné v systéme mať spoľahlivý modul pre odosielanie e-mailov. Opäť je jednou z možností využiť funkcionalitu čistého PHP. Vzhľadom k rôznorodosti používaných e-mailových služieb a rôznym druhom zabezpečenia som pri implementácii systému použil modul PHPmailer, ktorý je dostupný z repozitára PHP balíčkov Packagist pod LGPL 2.1 licenciou (čo znamená, že je možné modul voľne používať bez možnosti ho meniť). Inštalácia modulu bola vykonaná pomocou správcu závislostí Composer.

Modul PHPmailer sprístupňuje odosielanie e-mailov tým, že zapúzdruje celú klientskú časť SMTP komunikácie so SMTP serverom do objektu, v ktorom je možné cez jednoduché aplikačné rozhranie nastaviť parametre ako odosielateľ, SMTP server, príjemcovia, príjemcovia kópie, e-mailová adresa pre odpoveď, predmet, telo, prílohy a podobne. Odoslanie e-mailu je po inicializácii objektu vykonané pomocou zavolania jeho členskej funkcie `send()`, ktorá v prípade neúspechu vráti hodnotu `0` a vo svojej verejnej premennej `ErrorInfo` uloží informácie o chybe.

6.2 Technológie na strane klienta

Na strane klienta sa z hľadiska použitých technológii oplatí venovať pozornosť používaným Javascript frameworkom a knižniciam, ktoré sa spolu s CSS frameworkmi spoločne radia do kategórie tzv. „frontend“ technológií. V nasledujúcej časti sú zhrnuté významné frontend technológie použité pri implementácii inzertného portálu.

6.2.1 CSS framework W3.css

Aj keď by bolo možné vzhľad portálu navrhnúť aj bez použitia CSS frameworkov, použil som framework W3.css z niekoľkých dôvodov. Prvým a najdôležitejším je responzivita na všetkých zariadeniach a prehliadačoch. Zjednodušil som tým návrh optimálneho zobrazenia napriek prípadnej nekompatibilite rôznych prehliadačov. Ďalším dôvodom je sada použiteľných CSS animácii, moderných preddefinovaných farieb, panelov, tlačidiel, písiem a iných prvkov používateľského rozhrania stránky. Vedľajší efekt použitia CSS frameworku je teda tiež jednotný vizuálny koncept.

6.2.2 Javascript knižnica W3.js

Knižnica W3.js zjednodušuje prácu s dinamicky generovanými skupinami elementov v HTML. Príkladom môže byť zoznam možností pre formulárový vstupný prvok výberu, riadky tabuľky atď. Podobné prvky štruktúry je síce možné generovať pomocou jazyka Javascript bez použitia externých knižníc avšak táto knižnica robí generovanie výrazne prehľadnejším. Vstupom pre funkciu generovania sú HTML element, ktorý slúži ako vzor pre prvky, ktoré majú byť dinamicky generované a dáta vo formáte JSON. Funkcionalitu tejto knižnice som použil pre implementáciu väčšiny dinamicky sa meniacich zoznamov (napríklad zoznam subkategórií prislúchajúcich pod aktuálne zvolenú kategóriu vo formulári).

6.2.3 Nahrávanie súborov pomocou Dropzone.js

Pridanie nového inzerátu zahŕňa možnosť pridať k nemu taktiež 0 až 6 fotografií. Pre moderný spôsob (ťahaj a pusť) pridávania fotografií som použil externú Javasript knižnicu Dropzone.js dostupnú pod MIT licenciou (čo v skratke znamená, že je možné ju voľne používať, meniť a že vlastník produktu neručí za jej funkčnosť). Táto knižnica robí nahrávanie súborov na server jednoduchým. Na pozadí používa technológiu AJAX (viac o AJAX v ďalšej časti), je do veľkej miery prispôsobiteľná a je možné ju použiť bez ostatných prídavných knižníc (ako napríklad jQuery).

Po odoslaní súboru na server je na strane servera súbor uložený do dočasnej zložky a je pre súbor vygenerované unikátne meno, ktoré je v prípade úspechu odoslané naspäť na stranu klienta. Na strane klienta je táto odpoveď servera (unikátne meno) spracovaná a pridaná ku skrytému textovému prvku formulára určeného pre pridanie inzerátu. Po odoslaní formulára sú serverom spracované názvy súborov z tohto skrytého prvku a súbory sú premiestnené do trvalej zložky, ktorej obsah je už priamo zviazaný s inzerátom. Tak, ako je možné obrázky na server posielať je možné ich cez AJAX dotazy na server taktiež mazať.

6.2.4 Javascript a AJAX

Pre všetky ostatné účely som použil čistú formu jazyka Javascript s kombináciou asynchrónnych volaní funkcií AJAX. Vzhľadom k jednoduchosti stránky je funkcionalita čistého jazyka Javascript postačujúca. Asynchrónne volania funkcií AJAX sú vhodné z dôvodu obmedzenia sieťovej komunikácie. Pomomcou AJAX je možné odoslať dotaz na server, spracovať odpoveď a aktualizovať len tú časť webovej stránky, ktorá sa zmenila. To má za následok odľahčenie prehliadača, dátovej komunikácie aj servera. V prípade rozšírenej funkcionality Javascript pomocou niektorej z knižníc (napríklad W3.js) je možné zo servera odoslať len relevantné informácie vo formáte JSON alebo XML a neposielať celé fragmenty HTML kódu. Prijaté dáta vo formáte JSON môžu byť potom vizualizované stranou klienta.

Tento princíp funkcionality systému má názov REST API (Representational state transfer) a je prístupom, ktorým sa ideológia mikroframeworku Slim vyznačuje.

6.2.5 Mapa na úvodnej obrazovke

Pri návrhu mapy na úvodnej obrazovke som využil aplikačné rozhranie máp spoločnosti Google pre Javascript. Je pomocou neho možné definovať, ktorá časť mapy sveta má byť zobrazená a do akej úrovne by mala byť priblížená. Okrem základnej funkcionality však umožňuje pomocou konfiguračnej štruktúry (vo formáte JSON) definovať, ktoré vrstvy mapy majú byť viditeľné. Túto vlastnosť som využil pre odfiltrovanie informácii z mapy, ktoré s ohľadom na výber miest nie sú potrebné (napríklad pohoria, rieky, národné parky, budovy atď.). Okrem tejto funkcionality je možné vyznačovať do mapy grafické objekty a to automaticky alebo poskytnúť používateľovi možnosť kresliť do mapy ručne. Možnosť kresliť do máp Google je zhrňovaná pod ich modul `google.maps.drawing.DrawingManager`. Aplikačné rozhranie tohto modulu umožňuje manipuláciu s kreslenými objektami, nastavenie alebo zistenie ich vizuálnych vlastností (rozmery, farba, okraj, výplň, priesvitnosť) a geografických vlastností (polomer v metroch, geografické súradnice atď.). Pomocou máp Google bolo teda možné vykresliť kruh reprezentujúci oblasť na mape Slovenska, pod ktorú patria mestá, z ktorých chce používateľ systému inzeráty vyhľadávať. Po nakreslení kruhu do mapy alebo zmene jeho polohy je vyvolaná udalosť, ku ktorej je priradená jej obslužná funkcia. Tá zistí súradnice kruhu a zvolí najbližšie položené mesto z MongoDB databázy. Názov najbližšieho mesta je v zápätí nastavený ako vybrané mesto vo vyhľadávacom formulári. Tak isto systém reaguje aj na zmenu polomeru kruhu. Po zmene mesta alebo maximálnej vzdialenosti v poliach vyhľadávacieho formulára je zmena taktiež reflektovaná vlastnosťami kruhu na mape.

Zhrnutie
V tejto kapitole boli vymenované a popísané všetky dôležité technológie použité pri implementácii inzertného portálu. Ďalšia kapitola bude venovaná overovaniu správnej funkčnosti systému. Budú v nej postupne vymenované oblasti a postupy testovania inzertného portálu zo strany používateľa aj zo strany servera.

Kapitola 7

Overovanie funkčnosti systému

Obsah tejto kapitoly je venovaný overovaniu funkčnosti implementovaného systému z nasledovných hľadísk:

1. Konzistencia portálu,
2. Konzistencia používateľských transakcii,
3. Bezpečnosť systému,
4. Funkčnosť používateľského rozhrania.

Každá z vymenovaných oblastí zahŕňa niekoľko testovaných prípadov použitia, ktoré vyplývajú zo špecifikácie požiadaviek na systém.

7.1 Konzistencia portálu

Pod pojmom konzistencia portálu mám na mysli schopnosť systému dodržiavať všetky predpokladané obmedzenia. Znamená to napríklad, že nie je možné zmazať inzerát, pokiaľ je k nemu priradená aspoň jedna otvorená používateľská transakcia (otvorenou je tentokrát myslené, že platba od kupcu bola obdržaná administráciou a tovar zatiaľ nebol doručený kupujúcemu). Ďalej by systém nemal povoliť registráciu dvoch používateľov s tou istou e-mailovou adresou, nemal by povoliť odregistráciu používateľa, ktorý je účastníkom otvorených používateľských transakcií, nemal by evidovať dve transakcie s rovnakým variabilným symbolom a všetky spravované údaje by mali byť pred uložením zvalidované.

7.2 Konzistencia používateľských transakcii

Pojmom konzistencia používateľských transakcii je v tomto prípade myslené, že ak systém funguje správne, všetky používateľské transakcie by mali byť v jednom z definovaných stavov:

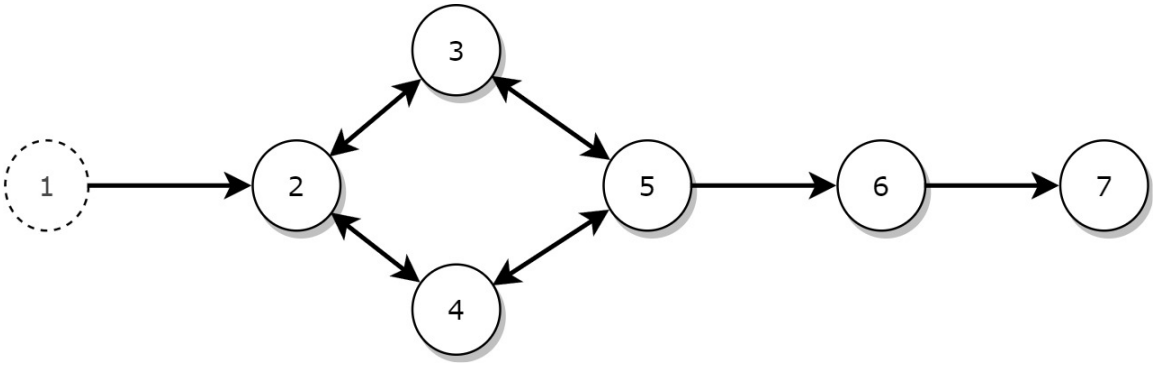
1. Transakcia nebola vytvorená,
2. Transakcia bola vytvorená ale obchod nebol odsúhlasený žiadnou zo strán,
3. Transakcia bola vytvorená a obchod bol odsúhlasený len zákazníkom,
4. Transakcia bola vytvorená a obchod bol odsúhlasený len inzerentom,

5. Transakcia bola vytvorená a obchod bol odsúhlasený oboma stranami, ale platba zatiaľ nebola obdržaná administrátorom,

6. Transakcia bola vytvorená, obchod bol odsúhlasený oboma stranami a platba bola obdržaná administrátorom ale tovar nebol doručený používateľovi v roli kupujúceho,

7. Transakcia bola vytvorená, obchod bol odsúhlasený oboma stranami, platba bola obdržaná, tovar bol obdržaný a ohodnotený kupujúcim používateľom.

Stav používateľskej transakcie môže byť zmenený len v rámci istých obmedzení prechodov medzi stavmi. Nasledujúci diagram ilustruje povolené prechody medzi stavmi používateľskej transakcie:



Obr. 7.1: Možné prechody medzi stavmi používateľskej transakcie

Z diagramu vyplýva, že účastníci transakcie majú možnosť s obchodom súhlasiť hneď po otvorení transakcie a odstúpiť od obchodu len za predpokladu, že platba zatiaľ nebola prijatá administráciou. Toto obmedzenie musí byť reflektované prvkami používateľského rozhrania na strane klienta, no aj kontrolami na strane servera. Z toho vyplýva, že vizualizovaný priebeh používateľskej transakcie by mal obsahovať niektoré prvky spoločné pre obe strany obchodu a niektoré odlišné pre stranu kupcu a predajcu. Napríklad, používateľ v roli kupujúceho by mal vidieť výzvu na zaplatenie a neskôr ohodnotenie tovaru a používateľ v roli predajcu zase výzvu na odoslanie tovaru. Individuálny obsah vizualizovaného priebehu používateľskej transakcie teda predpokladá, že systém správne identifikuje, ktorá strana obchodu je kupujúca a ktorá predávajúca. To závisí od charakteru inzerátu. Medzi ďalšie kritériá, ktoré hovoria o konzistencii transakcie patrí možnosť vytvoriť a aktualizovať hodnotenie obomi stranami a možnosť reagovať správami.

Testovanie konzistencie portálu

Pre overenie konzistencie transakcií aj samotného systému som vytvoril niekoľko fiktívnych používateľských účtov, inzerátov a transakcií. Pomocou vytvorenej vzorky dát som otestoval všetky možnosti (a obmedzenia) prechodu medzi stavmi transakcie a možnosti komunikovať, či hodnotiť používateľov. Ďalej som otestoval obmedzenia spojené s odregistráciou používateľov a mazaním inzerátov. Systém sa vo všetkých prípadoch použitia správa konzistentne a očakávané kontroly naozaj vykonáva.

7.3 Bezpečnosť systému

Bezpečnosť systému som overoval hlavne z hľadiska možných pokusov o používateľskú akciu vyžadujúcu patričné oprávnenia. Takou je z pravidla akcia, ktorá vyžaduje autentifikáciu používateľa. Nasleduje zoznam konkrétnych testovaných prípadov:

- Pokus o zmazanie alebo úpravu cudzieho inzerátu v roli bežného používateľa (prihláseného aj neprihláseného),
- Pokus o zmazanie alebo úpravu cudzieho inzerátu v roli administrátora,
- Pokus o vstup do administrácie v roli administrátora (vrátane administrátorských úkonov),
- Pokus o vstup do administrácie vrátane administrátorských úkonov v roli bežného používateľa (prihláseného aj neprihláseného),
- Pokus o reakciu na inzerát či uzavretie obchodu bez prihlásenia.

Pre otestovanie bezpečnosti systému som využil vzorku dát vytvorenú za účelom testovania predošlého bodu. Systém vo všetkých prípadoch najprv kontroluje oprávnenia prihláseného používateľa na vykonanie požadovanej akcie. V každom z testovacích prípadov systém odpovedal podľa očakávania v súlade s oprávneniami používateľa, z čoho je možné považovať testy na bezpečnosť systému za úspešné. Pre otestovanie bezpečnosti systému som použil doplnok POSTman prehliadača Chrome.

7.4 Funkčnosť používateľského rozhrania

Funkčnosť používateľského rozhrania som hodnotil v dvoch kritériách a to:

- Valídne správanie funkcii Javascript
- Responzivita rozhrania

Valídne správanie funkcií Javascript
S cieľom overiť funkcie Javascript som overoval validáciu používateľských vstupov realizovanú v Javascripte, interakciu prvkov používateľského rozhrania (napríklad správna funkcia fotogalérie inzerátu, rozhrania s mapou, aktualizácia formulárových polí na základe zmeny alternatívnych prvkov rozhrania a naopak), asynchrónne volania AJAX atď. Javascript funkcie fungujú podľa očakávaní a nevypisujú do konzoly žiadne chybové hlásenia spojené so zlyhaním počas vykonávania funkcie.

Responzivita rozhrania
Responzivitu a správne preskupovanie elementov som overil pre každú podstránku systému. Postupne som skúšal rozloženie elementov na stolných počítačoch so základným (1366px x 768px) aj vysokým rozlíšením (1920px x 1080px), ďalej na tablete (1024px x 768px) a mobilných telefónoch Microsoft Lumia a LG G4. Responzivitu som taktiež testoval vo všetkých bežne používaných internetových prehliadačoch. Preskupovanie elementov na stránke sa správalo v prípade všetkých podstránok portálu podľa očakávaní. Príklady rozloženia obsahu vybraných podstránok sú v prílohe B.

Kapitola 8

Záver

Cieľom tejto bakalárskej práce bola tvorba Inzertného portálu s ochranou kupujúceho, ktorého súčasťou je mapa určená pre filtrovanie inzerátov. V dokumente som sa zaoberal návrhom a implementáciou takého informačného systému.

V prvých kapitolách som sa venoval princípom tvorby webových informačných systémov a používaným technológiám. Značná časť práce je venovaná analýze inzertného portálu s ohľadom na existujúce riešenia a možnosti ochrany kupujúceho. Zo záverov v analýze je vytvorený návrh požiadaviek na informačný systém z hľadiska jeho funkcionality a používateľského rozhrania. Zvyšok práce je venovaný implementácii a použitým technológiám spolu s motiváciou k ich použitiu.

S cieľom overiť funkčnosť informačného systému som v databáze portálu vytvoril niekoľko používateľských účtov, inzerátov a používateľských transakcii. Stavy používateľských transakcii sú modelované tak, aby na nich bolo možné overiť a demonštrovať každý možný stav transakcie, inzerátu a celkovú konzistenciu funkcionality systému. Na modelových dátach v systéme som overil, že systém plní svoju funkciu podľa očakávaní. Snímok obsahu databázy, z ktorého je možné testovacie dáta do databázy importovať je dostupný spolu so zdrojovými kódmi pod názvom `DB_test_snapshot.sql`.

V porovnaní s existujúcimi systémami podobného zamerania vytvorený inzertný portál spĺňa štandardné vlastnosti a funkcie inzertných portálov. Na základe overenia funkčnosti a funkcionality pre ochranu kupjúceho sa domnievam, že portál spĺňa predpoklady pre nasadenie v reálnej prevádzke. Ciele práce boli naplnené bez zvyšku.

Projekt slúži ako funkčný základ. Stále však ostáva množstvo priestoru pre jeho zdokonaľovanie, či rozširovanie. Jedným z možných rozšírení je dodať funkcionalitu známu ako „strážny pes“, pomocou ktorej by bolo možné posielať na e-mail používateľovi nové inzeráty, ktoré by spĺňali parametre jeho uložených vyhľadávaní. Medzi možnosti zdokonaľovania systému patria napríklad optimalizačné techniky pre prácu s databázou. Konkrétne môže ísť o analýzu možností úprav s cieľom znížiť počet prístupov k databáze, či o optimalizáciu samotných databázových dotazov. Ďalšími možnými zdokonaleniami systému je integrácia používateľských účtov na sociálnych sieťach a SEO optimalizácia. Z hľadiska prístupnosti systému sa tieto rozšírenia stávajú bežnou súčasťou existujúcich portálov. Preto by boli v prípade nasadenia systému jeho vhodným doplnením, nie však podmienkou.

Literatúra

[1] *Informačné systémy*. 2017, [Online; navštívené 11.05.2017].
URL http://informacne-systemy.sk/co-su-to-informacne-systemy/

[2] *International Bank Account Number*. 2017, [Online; navštívené 11.05.2017].
URL https://en.wikipedia.org/wiki/International_Bank_Account_Number

[3] *Template processor*. 2017, [Online; navštívené 11.05.2017].
URL https://en.wikipedia.org/wiki/Template_processor

[4] Anley, C.: Advanced SQL injection in SQL server applications. 2002.

[5] Halfond, W. G.; Viegas, J.; Orso, A.: A classification of SQL-injection attacks and countermeasures. In *Proceedings of the IEEE International Symposium on Secure Software Engineering*, ročník 1, IEEE, 2006, s. 13–15.

[6] Kadlčík, J.: Responzivní design a technologie pro tvorbu webu. 2016.

[7] Lewis, P.; Thorogood, S.: *Animations and Performance*. 2017, [Online; navštívené 11.05.2017].
URL https://developers.google.com/web/fundamentals/design-and-ui/animations/animations-and-performance

[8] Lockhart, J.; Smith, A.; Allen, R.; aj.: *Router*. 2016, [Online; navštívené 11.05.2017].
URL https://www.slimframework.com/docs/objects/router.html

[9] Morris, B.: *Web page size and browser performance – why it still matters*. 2009, [Online; navštívené 11.05.2017].
URL http://www.ben-morris.com/web-page-size-and-browser-performance-why-it-still-matters/

[10] Pitt, C.: *Pro PHP MVC*. Apress, 2012, ISBN "978-1-4302-4165-2".

[11] Reinink, J.: *PSR-7: HTTP message interfaces*. 2016, [Online; navštívené 11.05.2017].
URL http://www.php-fig.org/psr/psr-7/

[12] Rouse, M.: *object-relational mapping (ORM)*. 2008, [Online; navštívené 11.05.2017].
URL http://searchwindevelopment.techtarget.com/definition/object-relational-mapping

[13] Št'ovíček, M.: Responzivní web design. 2013.

[14] The PHP Group: *What can PHP do?* 2017, [Online; navštívené 11.05.2017].
URL http://php.net/manual/en/intro-whatcando.php

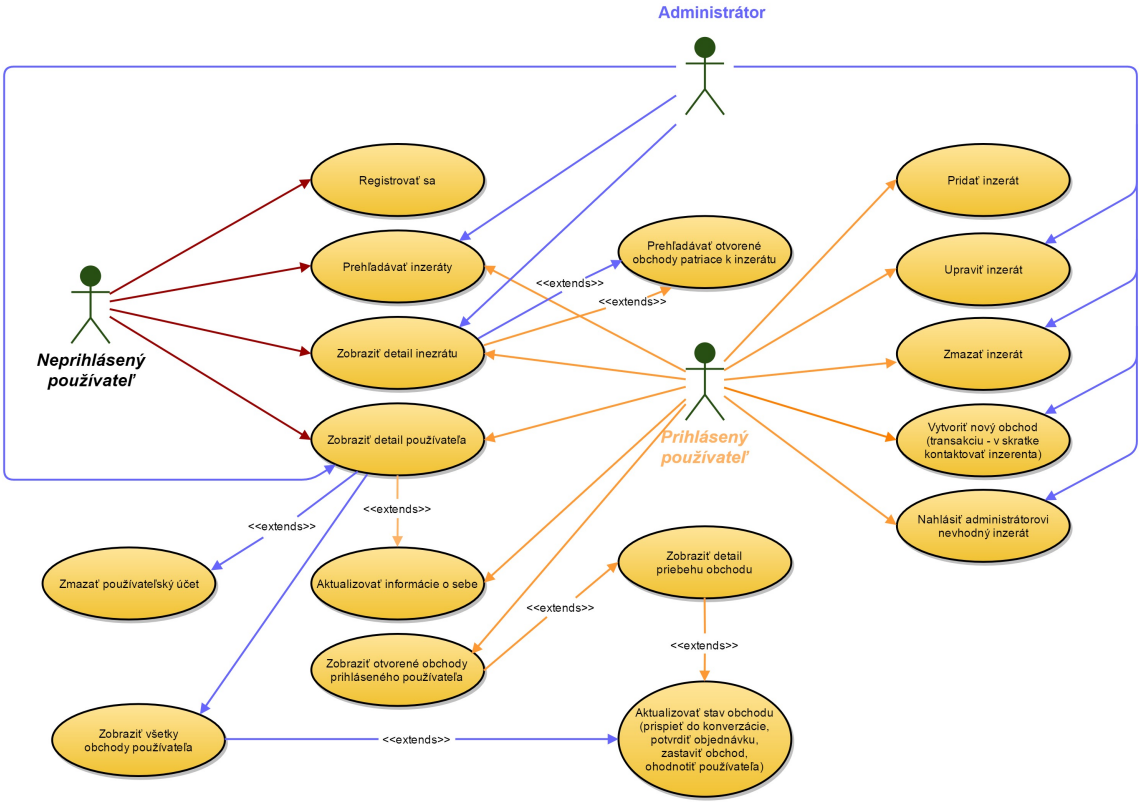
[15] United Classifieds s.r.o.: *Podmienky inzercie na Bazar.sk*. 2017, [Online; navštívené 11.05.2017].
URL https://s.aimg.sk/unitedclassifieds/css/data/bazar.pdf?v=20170315

Prílohy

Zoznam príloh

Príloha A

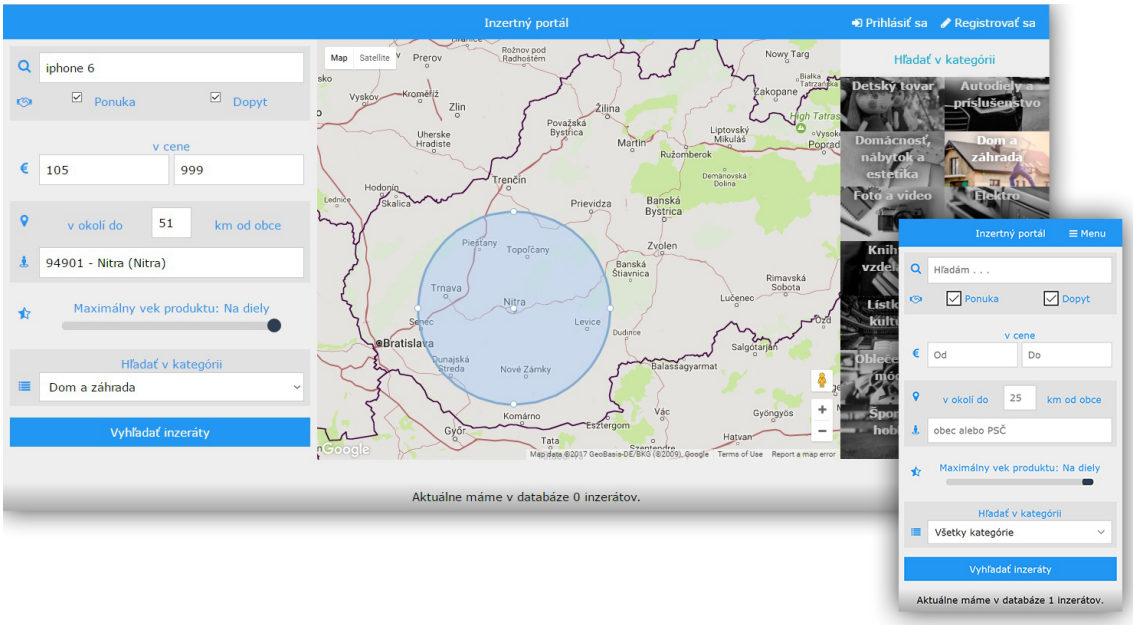
Diagram prípadov použitia



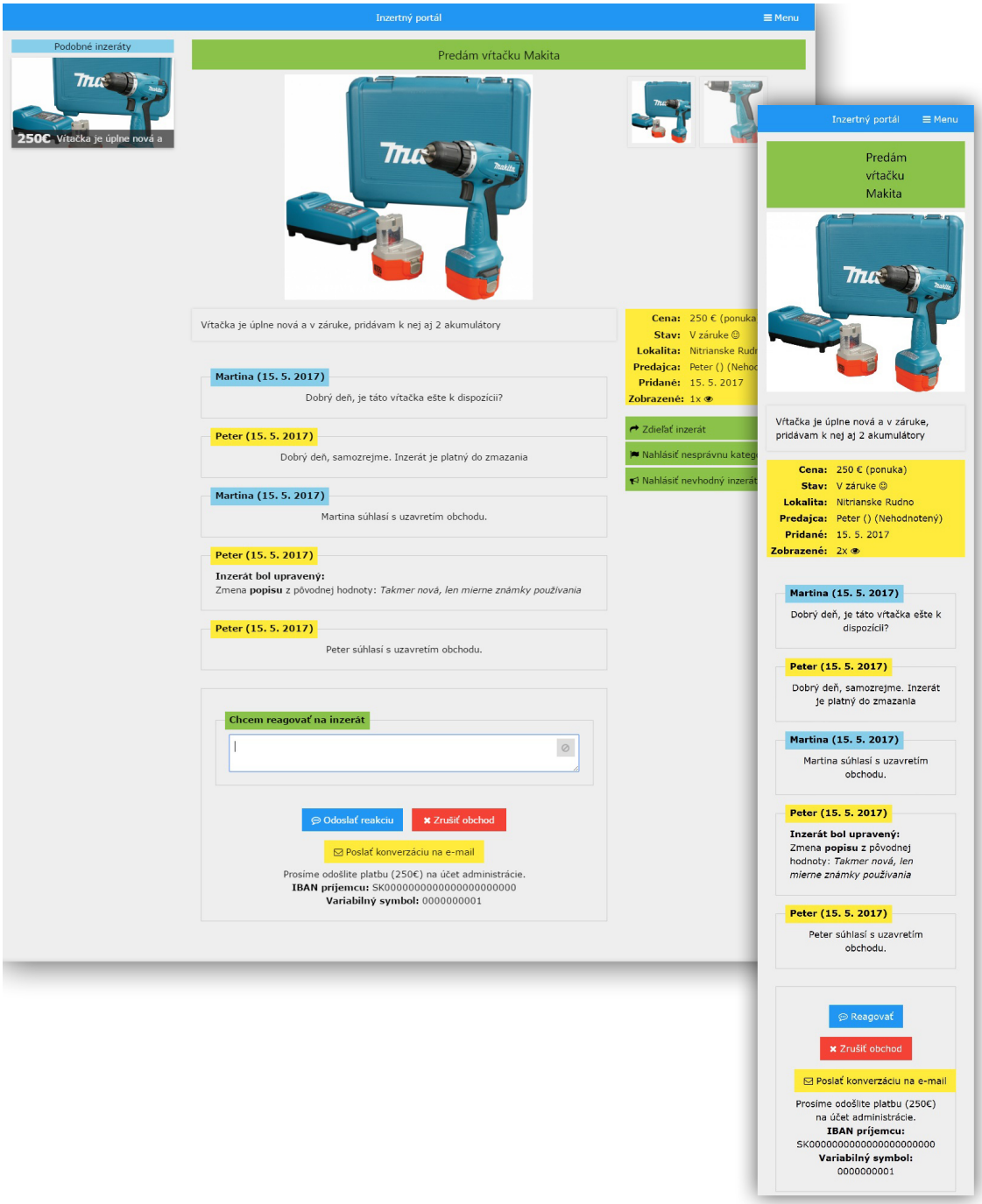
Obr. A.1: Diagram prípadov použitia

Príloha B

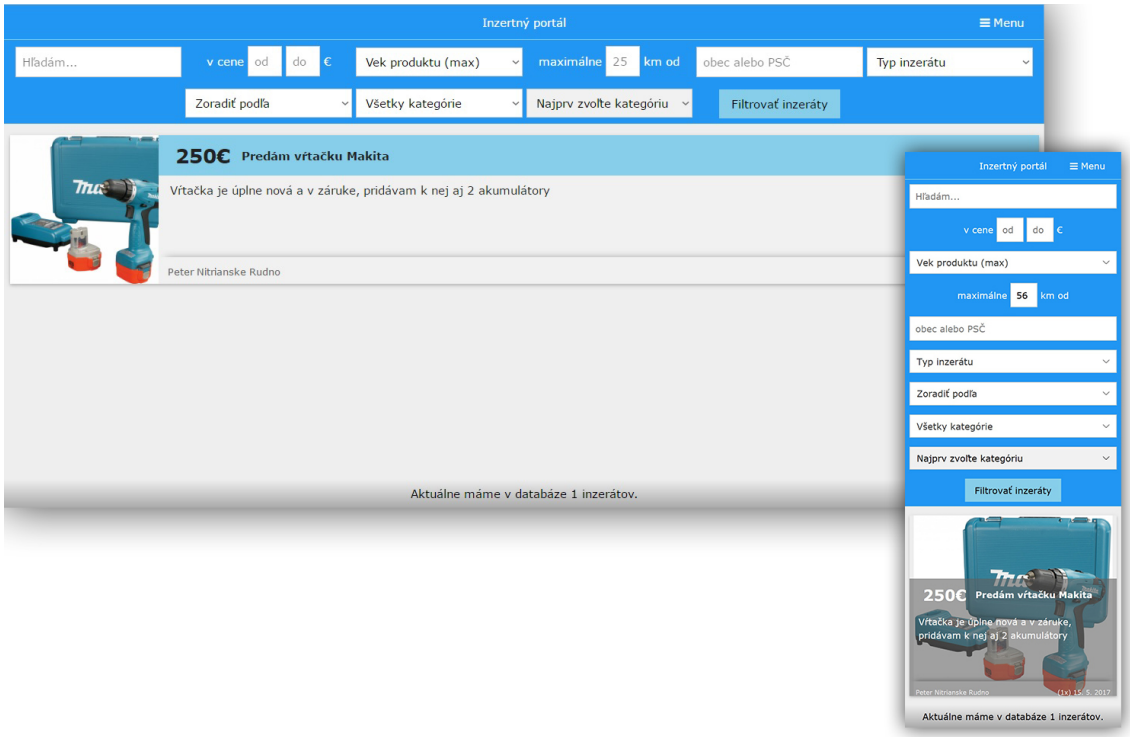
Obsah a rozloženie podstránok systému



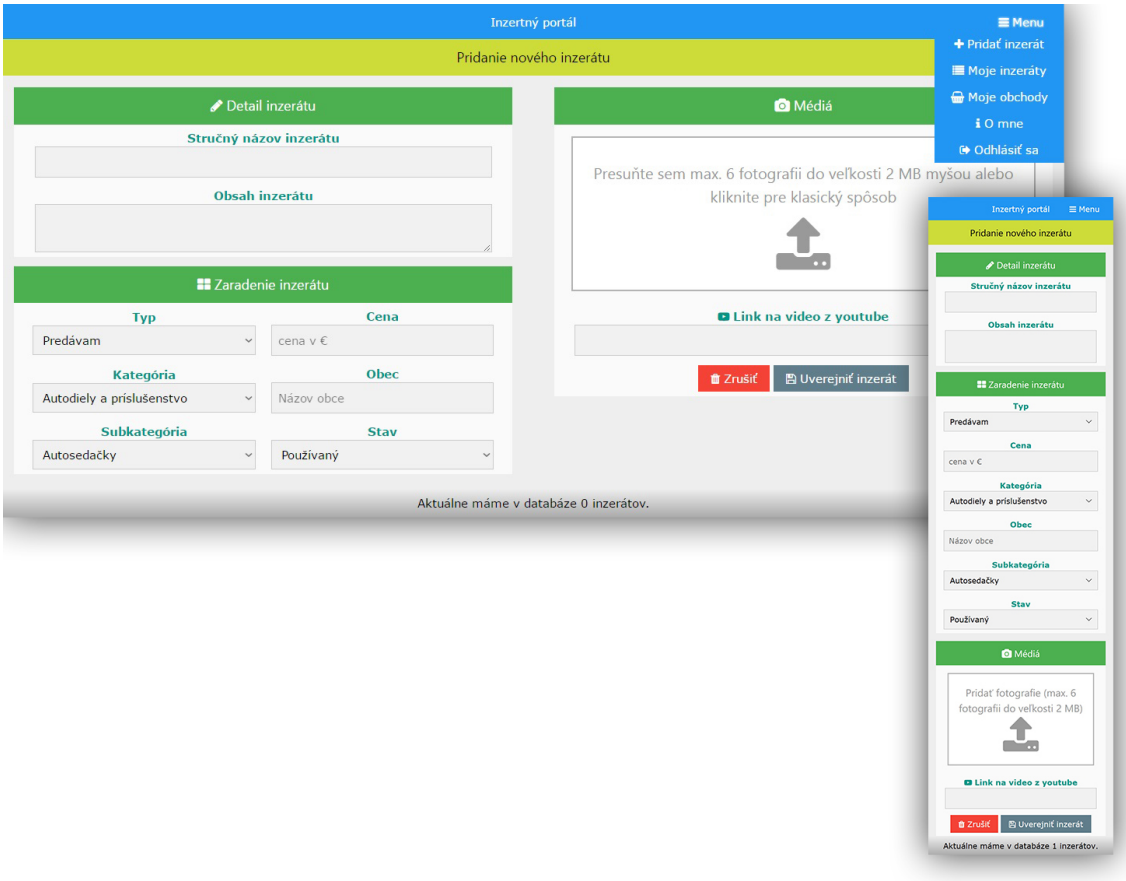
Obr. B.1: Úvodná obrazovka



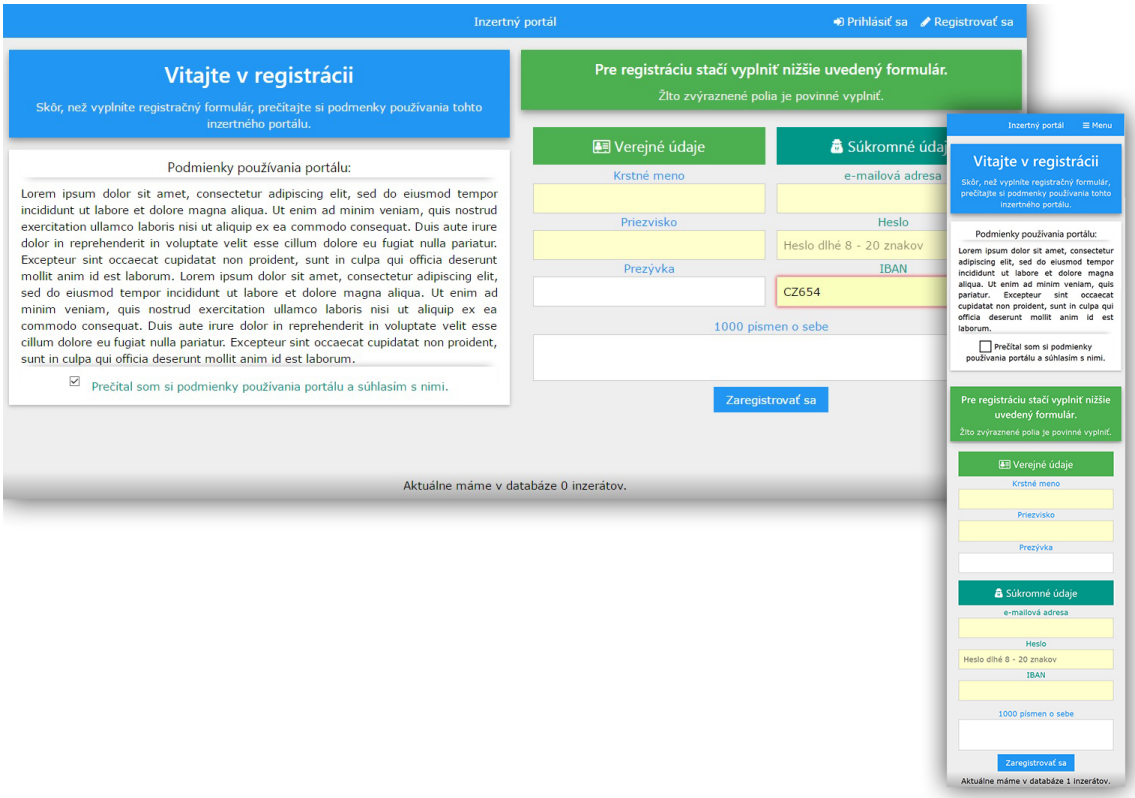
Obr. B.2: Detail inzerátu



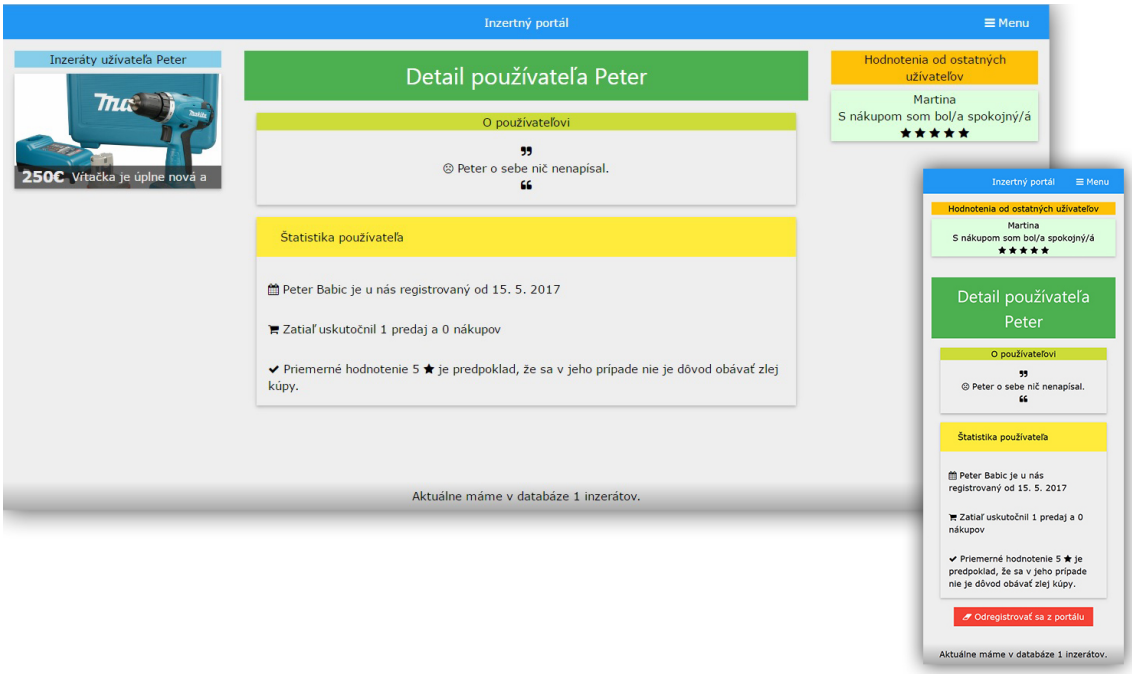
Obr. B.3: Prehľad inzerátov



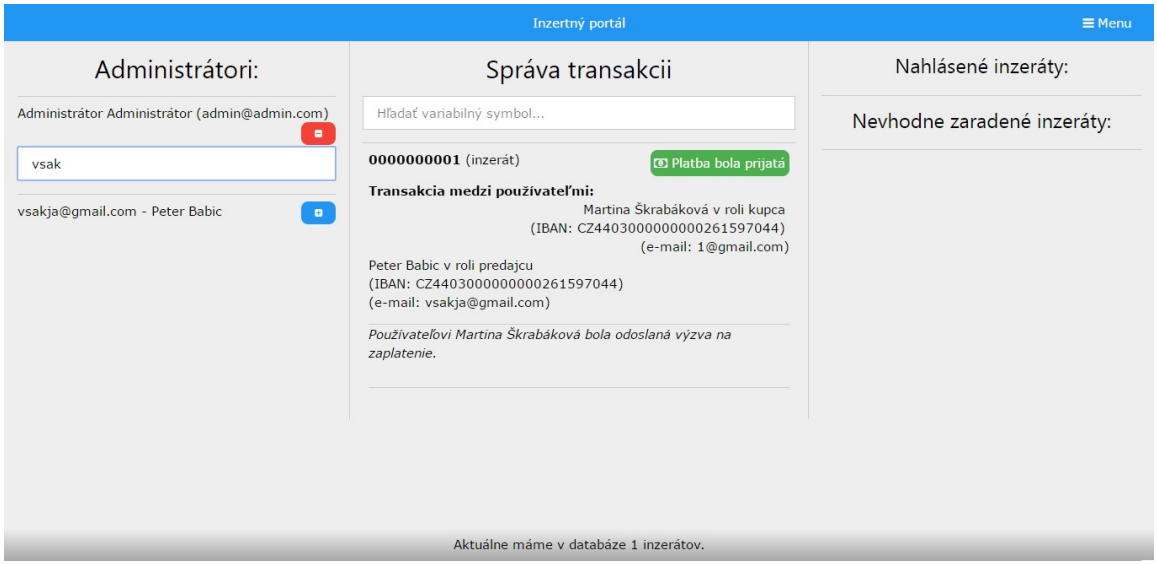
Obr. B.4: Formulár pre pridanie inzerátu



Obr. B.5: Registračný formulár



Obr. B.6: Detail používateľa



Obr. B.7: Panel administrácie