Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky
Katedra informatiky

Webová aplikácia

Tvorba internetového obchodu

BAKALÁRSKA PRÁCA

Juraj Prievalský

Študijný odbor: 9.2.1 Informatika
Vedúci práce: RNDr. Mária Pastorová
BRATISLAVA 2007

Prehlásenie:

Týmto prehlasujem, že som bakalársku prácu vypracoval samostatne a všetku použitú literatúru uvádzam.

.....

Touto cestou si dovoľujem poďakovať p. RNDr. Márií Pastorovej za odborné vedenie a rady počas spracovávania bakalárskej práce.

OBSAH

1 ÚVOD

V dnešnom modernom svete sa čoraz viac presúva komunikácia medzi ľuďmi do internetového sveta. A nie je tomu inak ani pri obchodovaní. Už asi nikoho z nás neprekvapí pojem "nakupovanie cez internet". Aj vzhľadom k tomuto fenoménu som sa rozhodol priblížiť, ako sa realizuje tvorba internetového obchodu a jej samotné prevedenie na webe vo forme internetovej aplikácie. Na internete je dostupných množstvo e-obchodov, ja sa v mojej práci pokúsim spojiť prehľadný design s veľkou funkcionalitou a ľahkou spravovateľnosťou, v čom veľa obchodov zaostáva.

Mojim cieľom je spraviť analýzu problému, navrhnúť databázu a jednoduchý framework, vybrať vhodné technológie a v neposlednom rade na dobrom základe naprogramovať komplexnú internetovú aplikáciu - internetový obchod.

Práca je rozdelená do šiestich kapitol a ako príloha je CD. Prvá kapitola je úvodná, stanovujem si v nej ciele práce. V druhej kapitole ukážem dva rôzne pohľady na internetový obchod, z pohľadu koncového používateľa a z pohľadu tvorcu aplikácie - vývojára. Tretia kapitola obsahuje komplexný návrh internetového obchodu. Nosnou časťou je návrh databázy a frameworku, na ktorom bude postavená aplikácia. Z najdôležitejšej funkcionality spomeniem rozdelenie obchodu na dve samostatné časti: back-end - adminovská časť a front-end - pre koncového používateľa, templatový systém - oddelenie designu od funkcionality, podpora jazykových mutácii stránky. Realizácia problému tvorí štvrtú kapitolu. Vyberiem v nej vhodné technológie a samotný postup práce na finálnu tvorbu internetovej aplikácie. Posledné časti práce tvoria záver a zoznam použitej literatúry.

2 ANALÝZA

2.1 Z pohľadu koncového používateľa

Ako by mal vlastne vyzerať moderný virtuálny obchod, v ktorom by sa zákazník nestratil? Množstvo internetových obchodov v súčasnej dobe je už nemoderných, takže sa pokúsim vybrať tie najdôležitejšie požiadavky pre koncového používateľa. Web stránka musí byť predovšetkým graficky prepracovaná, pretože prvý dojem je vizuálny, až potom nastáva samotné "browsovanie" obchodom. Samozrejmosťou je optimalizácia pre najpoužívanejšie webové prehliadače (Internet Explorer, Firefox, Opera). Mala by spĺňať nasledovné body:

- Prehľadná štruktúra stránky - kategórie (podkategórie), výrobcovia, produkty, prihlásenie, registrácia
- Nákupný košík, vkladanie a vyberanie produktov
- Veľké množstvo informácií o produkte, náhľady a obrázky produktov
- Nakupovanie aj bez predregistrácie
- Zľavy a akcie produktov, možnosť zľavniť konkrétnu kategóriu, konkrétny produkt pre všetkých alebo len pre vybraných používateľov
- Multijazyková podpora
- Zmena meny
- Jednoduché a pokročilé vyhľadávanie podľa kritérií

2.2 Z pohľadu vývojára

Pozrime sa na internetový obchod z druhej strany, z pohľadu vývojára. Dôraz sa bude klásť predovšetkým na ľahkú administráciu, modifikovateľnosť kategórií, produktov a výrobcov cez formuláre, kontrola korektnosti vstupných údajov, prehľad nad už existujúcimi používateľmi, ako aj ľahká rozšíriteľnosť

obchodu o novú funkcionalitu. Potrebujeme systém na prezeranie a vybavovanie objednávok. Takisto veľmi dôležitá je bezpečnosť, musíme zabezpečiť korektné prihlasovanie registrovaných používateľov, v admin zóne hierarchiu práv a ošetrovať vstupy, ktoré nám zadáva používateľ, aby sme zamedzili zadávaniu vlastných dopytov v našej databáze. Z hľadiska bezpečnosti potrebujeme ďalej šifrovať používateľské heslá.

3 NÁVRH

3.1 Rozdelenie obchodu

Je nutné aby bol obchod rozdelený na dve samostatné viacpoužívateľské časti:

1. Back-end (komplexná časť pre administráciu obchodu)
2. Front-end (časť pre koncového používateľa)

3.1.2 Back-end

Treba si uvedomiť, že adminovská časť nebude prístupná, ani len viditeľná koncovému používateľovi a teda bežný zákazník bude mať prístup len k polovici aplikácie. Admin zóna je chránená heslom a implementuje systém hierarchie práv pre administráciu. V praxi to znamená, že môže byť viac adminov, pričom každý z nich môže vykonávať len také činnosti, na ktoré má práva. Najvyššiu funkciu má superadmin s neobmedzenými právami, ktorý okrem iného môže aj vytvárať, editovať, zmazať iných adminov a nastavovať im práva.

Adminovská časť musí obsahovať komplexnú správu kategórií, produktov, výrobcov, objednávok, registrovaných používateľov a adminov. Samozrejmosťou je pridávanie a editovanie vo forme formulárov, kontrola korektnosti vstupov a priame ukladanie do databázy. Nevyžaduje pritom takú grafickú náročnosť ako front-end. Prihlasovanie do admin zóny bude

štandardne na adrese stránky s tým, že do adresy prehliadača pridáme `/admin` a nabehne nám prihlasovací formulár s menom a heslom.

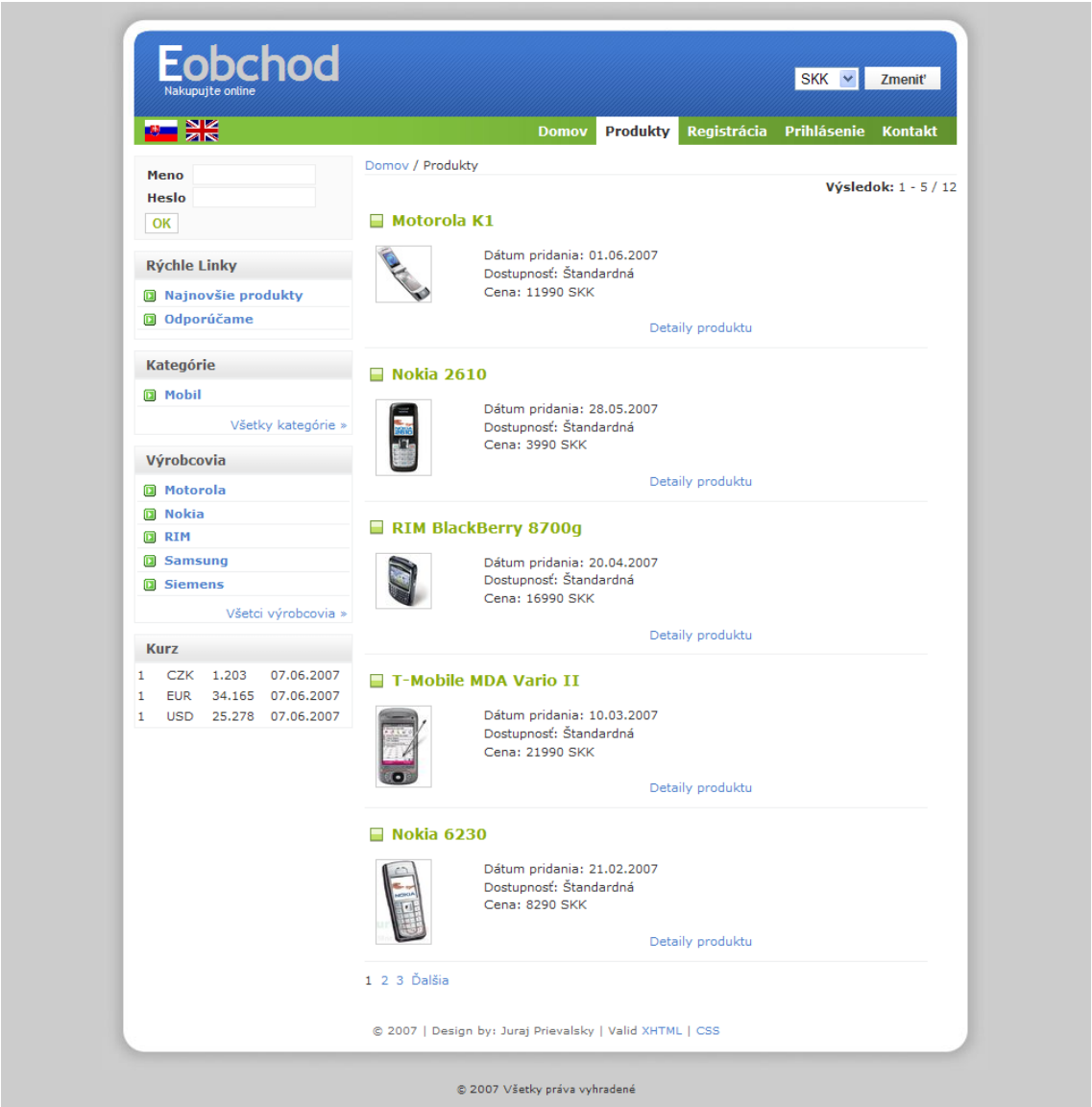
Na obrázku môžete vidieť môj návrh admin zóny po korektnom prihlásení superadmina. Zvolil som si anglický jazyk, vzhľadom k tomu, že chcem aby bol môj obchod čo najuniverzálnejší a toto som považoval za jeden zo štandardov.

Položky v menu:

- categories, products, producers - správa kategórií, produktov a výrobcov pridávanie (Add new...), editácia (Modify), zmazanie (Delete)
- editors - odporúčané produkty, môžeme sem pridať ľubovoľný aktívny produkt (Add to editors)
- orders - prehľad objednávok, zobrazenie (Show), vybavenie (Execute)
- users - prehľad registrovaných užívateľov, vyhľadávanie, editácia, zmazanie
- admins - správa administrátorov, pridanie, zmazanie, nastavenie práv (prístup k položke admins má len superadmin)
- logout - odhlásenie z admin zóny

3.1.2 Front-end

Front-end časť je určená pre koncového používateľa a jej obsah je automaticky generovaný z databázy. Vyžaduje predovšetkým grafickú náročnosť [7]. Môj návrh web stránky vyzerá nasledovne:



Štruktúra stránky:

- **header (hlavička)**
  - logo (odkazujúce na titulnú stránku), slogan
  - zmena meny podľa aktuálnych kurzov NBS

- zmena jazyka (slovenčina, angličtina)
- hlavné menu (domov, produkty, registrácia / nákupný košík, prihlásenie / odhlásenie, kontakt)

- **content (obsah)**
  - sidebar - obsahuje jednotlivé okná:
    - prihlásenie / odhlásenie pomocou formulára
    - / nákupný košík s celkovou sumou a editáciou objednávky
    - rýchle linky - najnovšie a odporúčané produkty
    - kategórie - prvých 5 + odkaz na všetky
    - výrobcovia - prvých 5 + odkaz na všetkých
    - aktuálne kurzy mien NBS
  - main content - aktuálny obsah podľa danej stránky
    - na titulke sa zobrazí top produkt
    - na stránke produktov samotný zoznam s náhľadmi obrázkov, krátkym popisom a stránkovaním
    - zvyšné stránky tvoria registračný formulár / editácia nákupného košíka, prihlasovací formulár / odhlásenie a kontaktné informácie

- **footer (päta)** - copyright

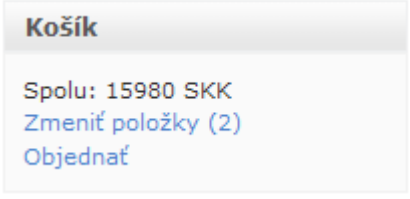
Pozn.: Lomítko znamená, ako sa položka zmení keď je používateľ prihlásený.

Musíme teda rozlíšiť 2 typy používateľov:

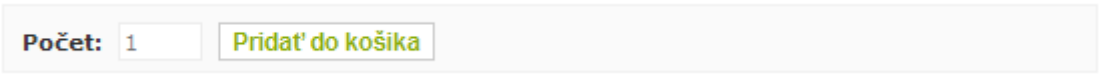
1. neprihlásený - má prístup len k prezeraniu produktov a ich cien
2. prihlásený - môže robiť objednávky

3.2 Nákupný košík

Dôležitú súčasť každého internetového obchodu tvorí virtuálny nákupný košík. Funguje na princípe



vkladania a vyberania produktov s automatickým výpočtom celkovej sumy. Zobrazí sa po prihlásení používateľa s tým, že navyše ku každému produktu pribudne aj formulár s pridaním počtu kusov. Štandardne máme jeden kus, avšak môžeme si pridať naraz aj viac kusov. Pokiaľ nastane situácia, že už nemáme zadaný počet kusov na sklade, vypíšeme používateľovi upozornenie a dovolíme mu pridať maximálny dostupný počet produktov.



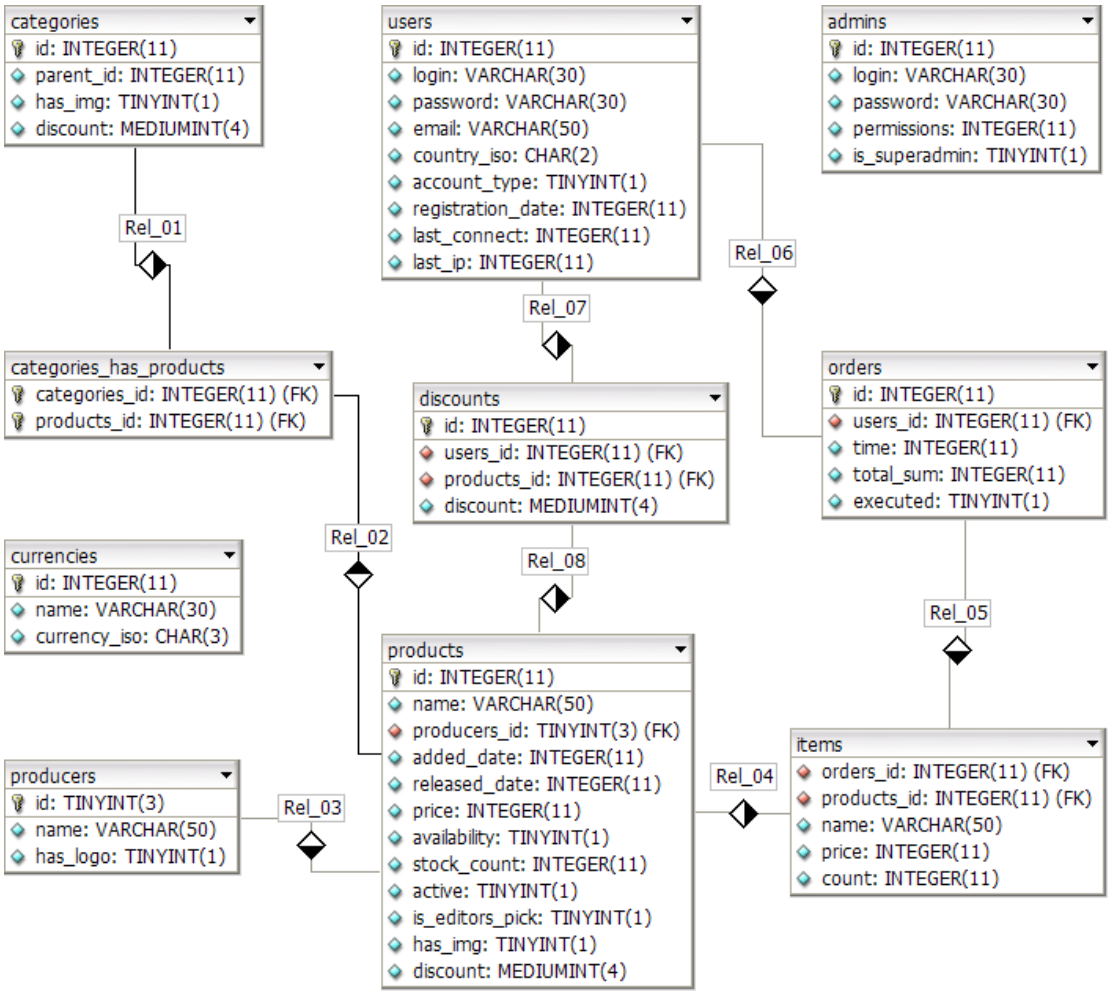
Počas nákupu môžeme kedykoľvek zmeniť položky v košíku:

Keď sa rozhodneme nákup ukončiť, tak klikneme na objednať a zobrazí sa nám ešte samotná objednávka na potvrdenie:

Po potvrdení sa košík vyprázdni a odošle sa objednávka. Košík som sa rozhodol zrealizovať pomocou session, to znamená, že aktuálne položky nákupy budú uchovávané v session premennej - produkt_[id produktu] = [počet kusov v košíku]. Až po objednaní sa obsah košíka uloží do databázy a session premenné sa vyprázdnia.

3.3 Návrh databázy

Základom internetovej aplikácie je správne navrhnutá štruktúra databázy. Môj návrh pozostáva z 10 tabuliek:



1. admins - administrátori

- id
- login - prihlasovacie meno
- password - prihlasovacie heslo, hashované pomocou MD5
- permissions - práva pre admin zónu vo forme súčtu jednotlivých práv, každá sekcia v admin zóne bude mať číslo práv a porovnaním bitového súčinu s aktuálnymi právami admina zistíme oprávnenie prístupu
- is_superadmin - či má admin neobmedzené práva - 0 alebo 1

2. categories - kategórie

- id
- parent_id - id rodičovskej kategórie
- has_img - či má kategória aj obrázok - 0 alebo 1
- discount - zľava na produkty v danej kategórii v percentách * 100 (aby sme mohli dať zľavu aj v nie celých percentách)

3. categories_has_products - produkty prislúchajúce ku kategóriám

- categories_id - id kategórie
- products_id - id produktu

4. currencies - používané meny

- id
- name - názov meny
- currency_iso - trojmiestna skratka meny, napr. SKK

5. discounts - zľavy

- id
- users_id - id používateľa, ktorému chceme dať zľavu
- products_id - id produktu, ktorý chceme zľavniť
- discount - samotná zľava (v percentách * 100)

6. items - položky v objednávke

- orders_id - id objednávky
- products_id - id produktu
- name - názov produktu
- price - cena, za ktorú bol objednaný produkt
- count - počet objednaných kusov

7. orders - objednávky

- id
- users_id - id používateľa, ktorý spravil objednávku

- time - čas objednávky (timestamp - počet sekúnd od 1.1.1970)
- total_sum - celková suma objednaných produktov
- executed - či bola objednávka vybavená - 0 alebo 1

8. producers - výrobcovia

- id
- name - meno výrobcu
- has_logo - či má výrobca aj logo - 0 alebo 1

9. products - produkty

- id
- name - názov produktu
- producers_id - id výrobcu
- added_date - reálny dátum pridania (timestamp)
- released_date - dátum pridania vo front-ende, slúži na systém automatického vydávania produktov v zadaných dátumoch (timestamp)
- price - cena
- availability - dostupnosť (štandardná, prémiová ...)
- stock_count - počet kusov na sklade
- active - či je produkt aktívny - 0 alebo 1
- is_editors_pick - či patrí produkt do nami vybranej edície - 0 alebo 1
- has_img - či má produkt obrázok - 0 alebo 1
- discount - zľava na produkt (v percentách * 100)

10. users - registrovaní používatelia

- id
- login - prihlasovacie meno
- password - prihlasovacie heslo, hashované pomocou MD5
- email - emailová adresa
- country_iso - iso kód krajiny - dvojmiestny, napr. sk
- account_type - typ konta (štandardné, prémiové ...)
- registration_date - dátum registrácie (timestamp)
- last_connect - dátum posledného prihlásenia (timestamp)

- last_ip - ip adresa, z ktorej sa používateľ naposledy prihlásil

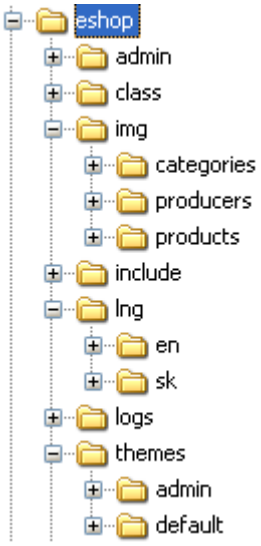
3.4 Návrh frameworku

Framework je prostredie, v ktorom je organizovaná a napísaná ďalšia aplikácia. Je napísaný v rovnakom programovacom jazyku ako aplikácia. V princípe je to súbor knižníc a kódu usporiadaných tak, aby pokrývali čo najviac funkčných požiadaviek spoločných pre rôzne aplikácie. Pri webových aplikáciách to môžu byť napríklad prístup k databáze, cachovanie údajov, logovanie aktivít, templatový systém, jazyková podpora. Framework má za úlohu ušetriť programátorovi čas tak, aby sa pri vývoji venoval len špecifickým požiadavkám pre aplikáciu, ktorú práve vyvíja a ktoré nejde zovšeobecniť.

Návrh môjho frameworku pre internetový obchod:

Adresáre:

- `admin` - adminovská časť
- `class` - jednotlivé triedy (zdrojové kódy aplikácie)
- `img` - obrázky produktov, kategórií, výrobcov ...
- `include` - konfiguračné súbory
- `lng` - jazykové mutácie stránky
- `logs` - logy
- `themes` - grafické mutácie stránky (templaty - zdrojové kódy web stránok)



3.4.1 Templatový systém

Templatový systém je jednou zo základných vecí, ktoré by mal spĺňať internetový obchod. Znamená možnosť oddeliť prezentačnú vrstvu (HTML kód) web stránky a programový kód používaný na danej stránke. Teda úplné oddelenie designu od funkcionality, ktoré uvažujeme len pri front-end časti, keďže back-end nevyžaduje grafickú náročnosť. V praxi sa bude celý design načítavať z jedného adresára v `themes` a pri prechode obchodu na iný design

nám stačí zmeniť tento adresár. Predvolený design bude v adresári `/themes/default/`.

3.4.2 Jazykové verzie

V dnešnej dobe je už samozrejmosťou, že stránka sa dá jedným kliknutím prepnúť na inú jazykovú verziu. A čo by to bol za obchod, keby sme nemohli mať aj zahraničných zákazníkov. Myšlienka je v podstate veľmi jednoduchá, všetky texty zo stránky vytiahneme do jazykových súborov a podľa vybraného jazyka ich načítame zo správneho adresára. V mojom frameworku napr. adresár `/lng/en/`. Pre pridanie ďalšieho jazyka len preložíme jeden jazykový súbor a pridáme nový jazyk v konfiguračnom súbore.

4 REALIZÁCIA

4.1 Technológie

4.1.1 Web

Na tvorbu web stránok sa používa štandardne HTML jazyk (hypertext markup language). Posledné trendy nám však predstavujú XHTML [5] (extensible hypertext markup language) s prísnejšou syntaxou, ktorý je aplikáciou XML. Ja som sa rozhodol pre verziu XHTML 1.0 Transitional s kombináciou CSS.

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

Kaskádové štýly - CSS (cascading style sheets) slúžia k oddeleniu obsahovej časti web stránky od vzhľadovej časti. Ak je vzhľad web stránky definovaný výhradne pomocou CSS, je potom veľmi jednoduché meniť kompletný vzhľad prezentácie iba zmenou definície CSS.

```
<link rel="stylesheet" href="style.css" type="text/css" />
```

Pre kódovanie stránky som si zvolil UTF-8 a aby nebol problém s diakritikou pri načítavaní dát z databázy, aj tam som si zvolil to isté kódovanie pre spojenie.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

4.1.2 Funkcionalita

Na výber z funkcionálneho hľadiska som mal viacero technológií:

- ASP.NET
- JSP
- ColdFusion
- PHP

ASP.NET (Active Server Pages .NET) je technológia nazývaná aj platforma či prostredie spoločnosti Microsoft slúžiaca na tvorbu web aplikácií. Dá sa povedať, že ASP.NET je nasledovníkom staršej ASP technológie, ktorá v polovici 90. rokov priniesla vďaka jednoduchému modelu generovania HTML kódu na strane servra doslova revolúciu, čo sa týka tvorby web aplikácii. Niečo podobné, teda ďalší výrazný krok vpred v oblasti tvorby web aplikácií, si Microsoft sľubuje aj od ASP.NET. Mnohé veci, ktoré boli problémom pri tvorbe najmä veľkých, náročných web aplikácií, sú riešené v ASP.NET omnoho lepšie. ASP.NET je súčasťou .NET Frameworku. Táto technológia je postavená na CLR (Common Language Runtime), čím rieši rôzne doterajšie problémy ako napríklad problém kompatibility rôznych programovacích jazykov. .NET Framework podporuje základne programovacie jazyky Visual Basic .NET, C#.NET, C++.NET a J#.NET, avšak existujú voľne stiahnuteľné kompilátory aj pre ďalšie jazyky ako napríklad Perl alebo Delphi. Ako ASP.NET sa obvykle označujú dva aplikačné modely, Web Forms a ASP.NET Web services. Pre úplnosť dodajme, že súčasťou .NET Frameworku sú i ďalšie aplikačné modely, možno tu vytvárať plnohodnotné windows aplikácie (Win Forms), konzolové

aplikácie bežiace na príkazovom riadku, windows služby, alebo i aplikácie pre mobilné non-PC zariadenia (Mobile Internet Toolkit) . Z toho, okrem iného, vyplýva i fakt, že všetky tieto aplikačné modely zdieľajú ten istý, spoločný programový model, a je tu teda ten istý spôsob kompilácie a vykonávania kódu, ten istý model, čo sa týka bezpečnosti, multithreadingu, či ošetrovania výnimiek. Možno ich programovať v tom istom programovacom jazyku, používajú sa tie isté základné knižnice napr. na prácu s reťazcami, XML alebo čo sa týka sieťovania resp. prístupu k dátam.

Java Server Pages (JSP) je nástroj od spoločnosti Sun Microsystems pre písanie dynamických stránok založený na jazyku Java, funkcionalitou veľmi podobný ASP alebo PHP. Jedná sa vlastne o HTML stránky, do ktorých je pomocou špeciálnych značiek vložený kód v Jave, ktorý sa uskutočňuje pri vykonávaní dopytu na strane servra. Na rozdiel od PHP a ASP, ktoré nemajú skoro žiadnu typovú kontrolu, JSP majú silnú typovú kontrolu z Javy. Druhou hlavnou odlišnosťou je kompilácia stránok do tzv. servletov, čo sú špeciálne triedy v jazyku Java, ktoré následne komunikujú s web servrom. JSP by sa teda dali charakterizovať aj ako nástroj na písanie servletov. Pretože servlety sú triedy v jazyku Java, môžu využívať ďalšie triedy, ktoré neboli písane pomocou JSP a pochopiteľne tak ako v celej Jave jednoducho používať aj predtým napísané triedy. To umožňuje rýchly vývoj aplikácií a čiastočne aj oddelenie designu od výkonného kódu. JSP taktiež umožňujú používanie "sessions", teda možnosť uchovávania dát konkrétneho používateľa.

ColdFusion je aplikačný server a framework na tvorbu softvéru a dynamických web stránok. Typickou črtou je prepojenosť so skriptovacím jazykom CFML (ColdFusion Markup Language) ale aj inými na strane servera ako Actionscript. Hlavnými službami sú konverzia html do pdf a flashpaper, validácia formulárov na strane klienta, platformovo-nezávislé databázové dopyty cez ODBC alebo JDBC. Ďalej vyhľadávanie dát v enterprise systémoch ako Active Directory, LDAP, POP, HTTP, FTP, indexovanie súborov, XML parsovanie, dopytovanie, validácia a množstvo ďalších. ColdFusion bol pôvodne produktom firmy Allaire, ktorú odkúpila Macromedia a nakoniec získaný spoločnosťou Adobe.

PHP [8] je populárny open source skriptovací jazyk, ktorý umožňuje spracovanie dát a následné vygenerovanie obsahu web stránky na strane servera, ktorý je odoslaný webovému prehliadaču. Obvykle sa PHP používa v spojení s databázou a je možné pomocou neho vytvoriť takmer akúkoľvek zložitú webovú aplikáciu. Spolupracuje s najrozšírenejšími webovými servermi. Architektúra Linux, Apache, MySQL, PHP (zaužívaná skratka je LAMP) sa stala veľmi obľúbenou v internetovom odvetví. Ja som sa rozhodol pre najnovšiu verziu PHP 5.

4.1.3 Databázový systém

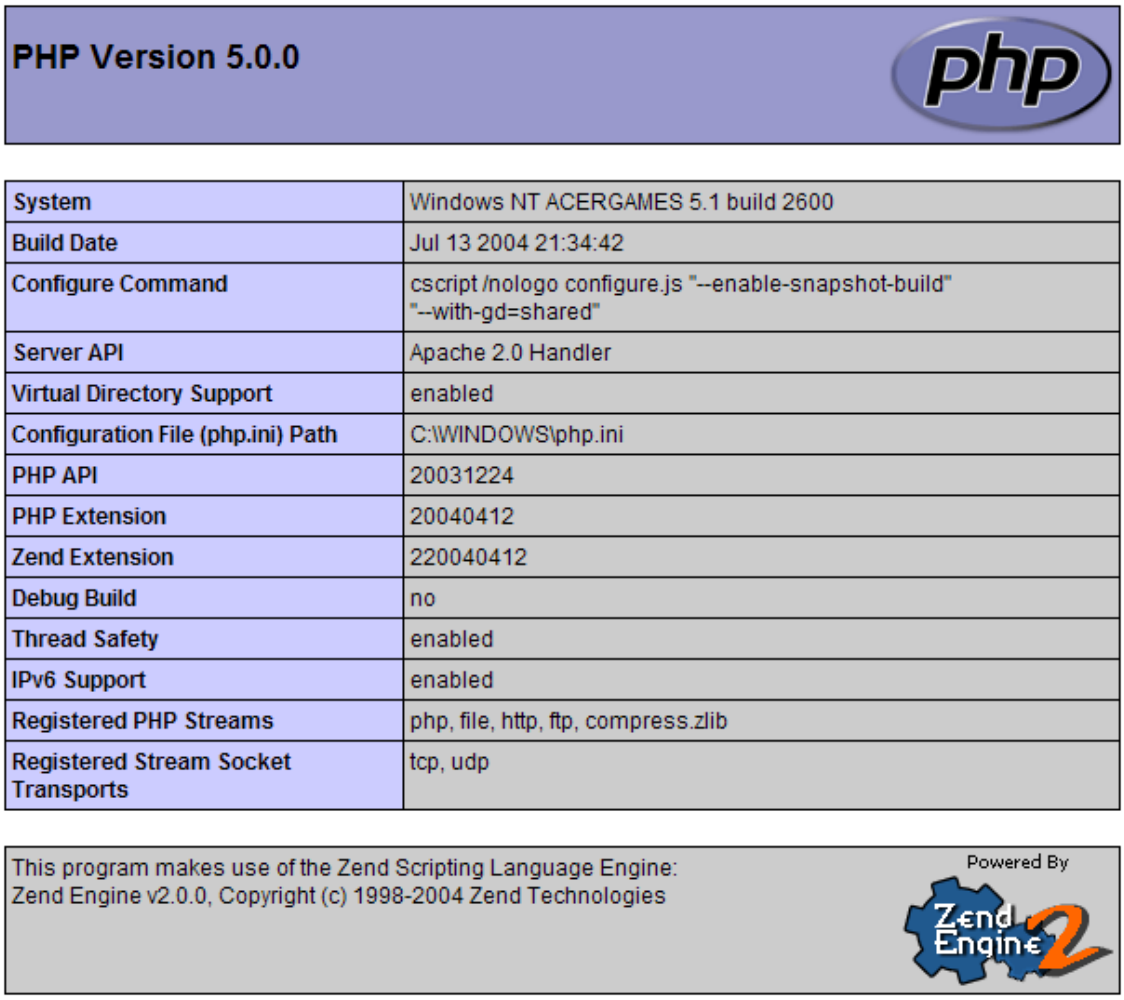
- MySQL
- PostgreSQL

PHP dokáže spolupracovať s relačnými databázami, ako MySQL, Oracle, IBM DB2, Microsoft SQL Server, PostgreSQL a SQLite, pričom si stále zachováva jednoduchú a priamočiaru syntax. Internetové aplikácie obvykle využívajú databázu MySQL [10], ktorá je rýchla a jednoduchá pre nasadenie v daných podmienkach. Samozrejme nie je problém využiť aj iné databázy.

PostgreSQL je objektovo-relačný databázový systém a používa dopytovací jazyk SQL pre výber a modifikáciu údajov. Dáta sú reprezentované ako množina tabuliek, ktoré spájajú cudzie kľúče. Jedna z charakteristík oproti jeho hlavne voľne šíriteľným konkurentom je programovateľnosť. PostgreSQL umožňuje oveľa jednoduchšiu tvorbu aplikácií z reálneho sveta využívajúcich údaje z databázy.

Ja som sa rozhodol pre použitie databázy MySQL, ktorá je voľne šíriteľná. Pre administráciu som si zvolil veľmi šikovný nástroj - phpMyAdmin [11]. Ponúka používateľom ucelený systém pre komplexnú manipuláciu s databázami a tabuľkami MySQL. Systémovým administrátorom poskytuje nástroje pre správu používateľov a ich práv. Z funkcionality spomeniem editáciu dát, zmeny v štruktúre tabuľky, export a import štruktúry a dát, vyhľadávanie dát a operácie nad celými tabuľkami. Je naprogramovaný v PHP.

4.2 PHP 5



PHP Version 5.0.0

System	Windows NT ACERGAMES 5.1 build 2600
Build Date	Jul 13 2004 21:34:42
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS\php.ini
PHP API	20031224
PHP Extension	20040412
Zend Extension	220040412
Debug Build	no
Thread Safety	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, http, ftp, compress.zlib
Registered Stream Socket Transports	tcp, udp

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.0.0, Copyright (c) 1998-2004 Zend Technologies

Powered By Zend Engine 2

PHP bolo spočiatku navrhnuté ako niekoľko skriptov v jazyku Perl, neskôr prepísaných do jazyka C. Autor Rasmus Lerdorf ich napísal v roku 1994. O rok neskôr svoje skripty zverejnil pod názvom "Personal Home Page Tools". Kombináciou s ďalším jeho programom Form Interpreter vzniklo PHP/FI. Zeev Suraski a Andi Gutmans, dvaja izraelskí vývojári prepísali syntaktický analyzátor (parser) v roku 1997 na novšiu verziu, ktorá sa stala základom PHP 3. Hneď po zverejnení verzie PHP 3 bola spustená aj oficiálna stránka. V roku 1999 sa Suraski a Gutmans opäť pustili do prepísania jadra, už pod názvom Zend engine. Založili spoločnosť Zend Technologies, ktorá sa odvtedy podieľa na ďalšom vývoji PHP. V máji 2000 bolo vydané PHP 4, ktorého jadro tvoril nový "Zend Engine 1.0". 13. júla 2004 bola vydaná verzia PHP 5 s jadrom "Zend Engine 2".

V PHP 5 [1] je nový objektový model. Obsluha objektov bola kompletne prepísaná a umožňuje lepšiu výkonnosť a funkcionalitu:

```
<?php
class SimpleClass
{
    // member declaration
    public $var = 'a default value';

    // method declaration
    public function displayVar() {
        echo $this->var;
    }
}
?>
```

- objekty sú vždy odovzdávané referenciou
- nový konštruktor triedy __construct()
- deklarácia premenných pomocou kľúčového slova “var” bola zavrhnutá (deprecated) a nahradila ju trojica “public, private, protected”
- volanie rodičovského konštruktora triedy pomocou parent::__construct()
- deštruktor triedy __destruct() vykoná kód pri deinicializácií objektu
- kopírovanie objektov - “clone”
- statické premenné - “static”
- konštanty - “const”
- prístupnosť metód taktiež pomocou “public, private, protected”
- triedy aj metódy môžu byť definované ako “final”, triedy nemožno extendovať a metódy prepísať
- funkcia autoload - zabezpečí automatické načítanie triedy, ktorú používame v kóde

```
function __autoload($class_name) {
        require_once "/php/classes/{$class_name}.inc.php";
}
```

- objekty môžu mať 3 čarovné metódy:
  __sleep() - limituje premenné objektu pri serializácií,
  __wakeup() - obnoví premenné objektu po deserializácií,
  __toString() - konverzia objektu na reťazec

- volania metód ako aj prístupnosť premenných triedy môžu byť preťažené (overloaded) pomocou __call(), __get() a __set() metód a sú volané len vtedy ak metóda alebo premenná neexistuje

- abstraktná trieda - “abstract”, môže obsahovať abstraktné metódy, pričom každá z nich musí byť implementovaná rozširujúcou triedou (extended class)

- interfejs - “interface”, definuje rozhranie, ktoré musia implementujúce triedy poskytovať, pričom trieda môže implementovať aj viac interfejsov

- iterácia objektu - objekt musí implementovať metódy: rewind, current, key, next, valid

- výnimky - slúžia na ošetrenie možných chýb v programe, buď použijeme: try { kód programu } catch (výnimka) { } alebo funkciu: set_exception_handler()

4.3 Programovanie aplikácie

4.3.1 Framework

Jadro frameworku tvorí súbor `/include/global.php`, ktorý inicializuje takmer všetky ostatné časti.

Zdrojový kód global.php:

```
<?php
include("defines.php");
include("const.php");
include("sql_defines.php");
include("functions.php");
require_once("db.php");

function __autoload($class_name) {
	require_once(CLASSES_DIR."/".$class_name.".class.php");
}

if (check_pirates($_SERVER['HTTP_USER_AGENT'])) {
	exit();
}

if (!check_robots($_SERVER['HTTP_USER_AGENT'])) {
	session_start();
}

if (isset($_GET['lang'])) {
	$_LANG = $_GET['lang'];
} elseif (isset($_POST['lang'])) {
	$_LANG = $_POST['lang'];
} elseif (isset($_SESSION['lang'])){
	$_LANG = $_SESSION['lang'];
} else {
	$_LANG = DEFAULT_LANG;
}

if (isset($_GET['currency'])) {
	$_CURRENCY = $_GET['currency'];
} elseif (isset($_POST['currency'])) {
	$_CURRENCY = $_POST['currency'];
} elseif (isset($_SESSION['currency'])) {
	$_CURRENCY = $_SESSION['currency'];
} else {
	$_CURRENCY = DEFAULT_CURRENCY;
}
```

```
if (!@include_once(LNG_DIR."/".$_LANG."/common.php")) {
      $_LANG = DEFAULT_LANG;
      require_once(LNG_DIR."/".$_LANG."/common.php");
}

$_SESSION['lang'] = $_LANG;
$_SESSION['currency'] = $_CURRENCY;
?>
```

Popis kódu:

Vložené súbory (pomocou include, require_once). Rozdiel medzi nimi je v tom, že require pri chybe zastaví vykonávanie ďalšieho kódu. Once znamená, že súbor môžeme vložiť len raz.

Include:

`defines.php` - všetky zadefinované konštanty v aplikácií

`const.php` - univerzálne konštanty

`sql_defines.php` - konštanty názvov tabuliek v databáze

`functions.php` - univerzálne funkcie

Require_once:

`db.php` - inicializácia databázy

Funkcie:

`__autoload()` - zabezpečí automatické načítanie použitej triedy

`check_pirates()` - kontrola web kopírovacích nástrojov

`check_robots()` - kontrola vyhľadávacích enginov, ak ich nenájde môžeme inicializovať session pomocou `session_start()`

Globálne prístupné premenné:

- `$_DB` - databázový objekt, inicializovaný v súbore `db.php`
- `$_LANG` - aktuálny jazykový iso kód, prenášaný cez session, ak neexistuje, načíta sa predvolený

- `$_LNG` - aktuálne jazykové pole, ktoré sa načíta z adresára `/lng/$_LANG/`
- `$_CURRENCY` - aktuálny kód meny, prenášaný cez session, ak neexistuje, načíta sa predvolený

Templatový systém

Inicializácia templatového objektu je v súbore `/common.php`. Nastavíme kódovanie stránky, nadpis, jazykovú premennú, header a footer.

```
$template = new Template(THEME, THEMES_DIR, REL_PATH);
$template->SetCharset(PAGE_CHARSET);
$template->SetPageTitle(DEFAULT_TITLE);
$template->SetLangVar("_LNG", $_LANG);
.
.
.
$template->SetHeader("header.tpl", $params);
$template->SetFooter("footer.tpl", $params);
```

- `$params` - pole parametrov, ktoré sa pošlú do templatu a nahradia príslušnú premennú - `$TPL['index']`

Použitie templatového systému:

```
$output = $template->ParseRow("content.tpl", $params);
```
- vráti html kód do premennej $output

```
$template->AddContent("content.tpl", $params);
```
- vyskladávanie finálnej stránky, pridávanie obsahu

```
echo $template->Build();
```
- zobrazenie finálnej web stránky

Jazykové verzie

Aktuálny jazykový súbor máme načítaný v poli `$_LNG`, po nastavení tejto jazykovej premennej pri vytváraní templatového objektu sa všetky výskyty kódu typu

```
<?php echo $_LNG['name']; ?>
```

nahradia prekladom daného indexu podľa aktuálneho jazyka.

4.3.2 Finálna aplikácia

Prvou fázou bolo naprogramovanie back-end časti a to z jednoduchého dôvodu. Triedy, ktoré takto vytvorím, môžem použiť aj vo front-ende, len ich výstup bude viac graficky naformátovaný. V adminovskej časti nám stačí vylistovať zoznamy do normálnych tabuliek. Treba si uvedomiť, že v admin časti budeme mať len 2 typy súborov - listy a formuláre. Listy na zobrazenie zoznamu kategórií, produktov, výrobcov, objednávok, užívateľov, adminov z databázy a formuláre na pridávanie a editovanie týchto položiek priamo do databázy. Prístup do admin zóny je chránený autentifikáciou cez session. Mal som ešte na výber HTTP autentifikáciu, ale keďže nepotrebujeme chrániť iné ako php súbory, tak som ju nepreferoval.

Druhou fázou bolo naprogramovanie front-end časti. Tu som mal už veľa práce uľahčenej, pretože väčšina tried sa použila tých istých. Zoznamy sa nezobrazili vo forme jednoduchých tabuliek, ale pekne graficky nadesignované. Postup je asi nasledovný: Najprv som spravil kompletný design web stránky vo Photoshope, narezal som ho do XHTML + CSS a potom postupne rozsekal na jednotlivé šablóny (templaty) v mojom frameworku. Popri práci som musel dodržiavať aj optimalizáciu pre jednotlivé prehliadače, pretože niektoré parametre v CSS má každý prehliadač špecifické.

4.4 Bezpečnosť

Pri prevádzke dynamických internetových stránok existujú rôzne bezpečnostné hrozby od SQL injection, cez XSS ...

SQL injection je hackerská metóda ktorá využíva nedostatky v kontrole vstupných dát k tomu aby mohol hacker podstrčiť namiesto očakávaných informácií vlastné a prebrať tak kontrolu nad databázou nad rámec svojich práv. Prebratím kontroly je myslená úplná kontrola nad databázou kde hacker môže databázu kopírovať, mazať, pridávať ľubovoľné údaje alebo kradnúť dôležité informácie. Platí pravidlo: "Neverte údajom od používateľa!". K problémom s SQL injection nastáva aj vtedy keď sa nestaráme poriadne alebo vôbec o informácie ktoré má obdržať náš SQL dopyt z adresy URL. Napríklad: skript očakáva číslo ale my mu pošlete písmeno, v prípade že skript nevie na túto situáciu pružne reagovať zobrazí chybové hlásenie, ktoré môže byť znakom, že niečo nie je v poriadku.

Vhodné opatrenia pre bezpečnosť údajov zadaných používateľom:

- Vždy predtým ako nejaké údaje uložíme do databázy, preženieme ich funkciou addslashes(). Tá potlačí význam niektorých špeciálnych znakov ktoré by mohli spôsobiť problémy. Pri získavaní údajov z databázy musíte potom použiť funkciu stripslashes() ktorá odstráni spätné lomítka a údaje budú v pôvodnom stave.

- V súbore PHP.INI existujú direktívy magic_quotes_gpc a magic_quotes_runtime. Magic_quotes_gpc je funkcia magických úvodzoviek ktorá všetkým údajom prichádzajúcim z GET, POST a COOKIEs automaticky vkladá spätné lomítka. Pri tejto direktíve nie je na škodu vytvoriť si vlastnú funkciu add_slashes() ktorá by nezávisle na tom, či je magic_quotes_gpc zapnuté alebo nie.
    - funkcia zistí, či je magic_quotes_gpc zapnuté alebo nie a podľa toho aplikuje na reťazec addsleshes() alebo nie

```
function gpc_addslashes($str)
{
        return (get_magic_quotes_gpc() ? $str : addslashes($str));
}
```

Druhá direktíva magic_quotes_runtime ošetruje dáta z databázy.

- Niekedy potrebujeme zobrazovať v php aj časti HTML kódu ktorý sa môže vykonať a spôsobiť problémy. Jednoduchým riešením je funkcia strip_tags() ktorá umožňuje automatické odstránenie všetkých tagov. Má aj nepovinný parameter v ktorom môžeme zapísať povolené tagy. Týmto zabránime prípadnému útočníkovi zadávať skripty ktoré by sa mohli vykonať a smerovať naspäť do prehliadača.

- Ak chceme zobrazovať HTML kód ale nechceme aby sa vykonal môžeme použiť funkciu htmlspecialchars() ktorá pretvorí znaky na neškodné HTML entity. Napríklad znak < by premenila na <

- Pokiaľ nejaké údaje dávame funkciám system() alebo exec() mali by sme použiť funkciu escapeshellcmd(). Touto funkciou potlačíme význam všetkých metaznakov a tým zabránime spúšťaniu akýchkoľvek príkazov.

- Ak pracujeme s číselnými hodnotami (napríklad SQL dopyt prijímajúci číselnú hodnotu z URL) mali by sme hodnotu testovať pomocou funkcie intval(). Táto funkcia skúša či hodnota sa dá premeniť na číslo, ak to nebude číselná hodnota vráti 0, dobrá funkcia pri ochrane pred SQL injection.

Cross-site scripting (XSS alebo CSS) využíva slabé miesto vo webovej aplikácii vďaka ktorému je do nej možne vložiť škodlivý kód. Prestavme si webovú aplikáciu s vyhľadávaním. Ak klikneme na tlačidlo hľadaj aplikácia odošle serveru požiadavku. Čo sa ale stane ak do políčka hľadať vložíme nejaký kód? Útočník môže URL upraviť takto:

```
/hladat.php?najdi=<script src="utocnikovweb.sk/xss.js"></script>
```

Týmto spôsobom môže vykonať akýkoľvek javascript. Môže napríklad ukradnúť tzv. „account hijacking“, vďaka ktorej sa útočník môže nalogovať do účtu obeti.

4.5 Validácia

Posledná fáza je otestovanie a naplnenie obsahu obchodu. Ja som sa rozhodol, že na prezentačné účely spravím virtuálny obchod s mobilmi - mobilshop ☺ Nebolo nič jednoduchšie, prihlásil som sa do admin zóny ako superadmin pomocou prihlasovacieho mena a hesla na adrese http://sprite.edi.fmph.uniba.sk/~georgo/admin/. Najprv som si v sekcii kategórie pridal jednotlivé kategórie, keďže budem pridávať len mobilné telefóny, vystačil som si aj s jednou kategóriou. Následne som formulármi s automatickou kontrolou korektnosti vstupu popridával jednotlivé mobily, ich výrobcov, ceny, dostupnosť, zaradenie do kategórií, počet kusov na sklade, popis v jednotlivých jazykoch, obrázky, ktoré sa automaticky upravili na požadovanú veľkosť a dátum vydania. Na záver som sa prepol do front-end časti http://sprite.edi.fmph.uniba.sk/~georgo/index.php , kde sa mi vygeneroval môj práve naplnený mobilshop. Tým bol cieľ mojej práce splnený, mohol som sa zaregistrovať ako bežný zákazník a objednať si tie svoje obľúbené produkty. Objednávky pribudli v admin zóne a mohol som si ich prezerať a vybavovať.

5 ZÁVER

V úvode práce som si stanovil viacero cieľov. Podarilo sa mi spraviť analýzu problému, podľa ktorej som navrhol databázu a jednoduchý framework. Domnievam sa, že naprogramovanie mojej internetovej aplikácie na takto dobre vybudovaných základoch bolo úspešné, o čom svedčí aj funkčné prevedenie na webe: http://sprite.edi.fmph.uniba.sk/~georgo/index.php. Stránku som použil predovšetkým na prezentáciu mojej práce, takže sa môže stať, že v budúcnosti nebude dostupná. Keďže obchod je naprogramovaný maximálne všeobecne, dá sa naplniť ľubovoľným obsahom a teda ja som z mojej verzie na webe spravil veľmi rýchlo a pohodlne internetový obchod s mobilmi. Fantázií sa medze nekladú, takže môžeme mať množstvo menších nezávislých obchodov.

Ako rozšírenie plánujem spraviť jeden veľký hyperobchod, ktorý by spravoval menšie obchody a distribuoval im produkty na pár kliknutí. Všetko so spoločnou databázou a prehľadnými štatistikami o jednotlivých obchodoch, systémom fakturácie každej objednávky. Takisto by bola možnosť načítania produktov z cenníka alebo naopak export, presunu tovaru medzi ľubovoľnými dvoma obchodmi. Keď potrebujeme súrne odísť, tak prerušiť nákup a pokračovať pri ďalšom prihlásení. Zaujímavé je aj konanie pravidelných aukcií produktov alebo darovanie produktu inej osobe.

V súčasnej forme sa dá tovar objednávať len na dobierku, teda platba prebehne až po doručení tovaru. Je potrebné implementovať aj platbu cez internet banking alebo kreditnou kartou, aby sme mohli byť celosvetoví. Uzavrie sa zmluva s bankou alebo providerom kreditných kariet a implementuje sa jeden z ponúkaných modulov na e-platby.

Je to síce hudba budúcnosti, ale asi nikto z nás si netrúfne povedať ako raz budeme nakupovať v budúcnosti, už súčasný vývoj technológií ide míľovými krokmi a môže sa stať, že nakupovať budeme už len virtuálne.

6 ZOZNAM POUŽITEJ LITERATÚRY

1. W. Jason Gilmore: Veľká kniha PHP 5 a MySQL, 2007
2. Ľuboslav Lacko: PHP a MySQL - Hotová řešení, 2006
3. George Schlossnagle: Pokročilé programování v PHP 5, 2004
4. Andi Gutmans, Stig Saether Bakken, Derick Rethans: Mistrovství v PHP 5, 2007
5. Elizabeth Castro: HTML, XHTML a CSS, 2007
6. Rastislav Škultéty: JavaScript - Kapesní přehled, 2006
7. Steve Krug: Webdesign, 2006
8. http://www.php.net/
9. http://www.w3.org/TR/xhtml1/
10. http://www.mysql.com/
11. http://www.phpmyadmin.net/