



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



**FAKULTA PODNIKATELSKÁ
ÚSTAV MANAGEMENTU**

FACULTY OF BUSINESS AND MANAGEMENT
INSTITUTE OF MANAGEMENT

ANALÝZA A NÁVRH DISTRIBUČNÍ B2B, B2C APLIKACE

ANALYSIS AND DESIGN OF DISTRIBUTIONAL B2B AND B2C APPLICATION

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. JIŘÍ NOVÁK

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. PETR DYDOWICZ, Ph.D.

BRNO 2010

ZADÁNÍ DIPLOMOVÉ PRÁCE

Novák Jiří, Bc.

Řízení a ekonomika podniku (6208T097)

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách, Studijním a zkušebním řádem VUT v Brně a Směrnicí děkana pro realizaci bakalářských a magisterských studijních programů zadává diplomovou práci s názvem:

Analýza a návrh distribuční B2B, B2C aplikace

v anglickém jazyce:

Analysis and Design of Distributional B2B and B2C Application

Pokyny pro vypracování:

Úvod

Vymezení problému a cíle práce

Teoretická východiska práce

Analýza problému a současné situace

Vlastní návrhy řešení

Závěr

Seznam použité literatury

Přílohy

Seznam odborné literatury:

- BLAŽKOVÁ, M. Jak využít internet v marketingu :krok za krokem k vyšší konkurenceschopnosti. 1. vyd. Praha : Grada, 2005. 156 s. : il. ISBN 80-247-1095-1.
- HLAVENKA, J. Dělejte byznys na Internetu :jak využít Internet k prospěchu firmy i jednotlivce. Vyd. 1. Praha : Computer Press, 2001. vii, 226 s. ISBN 80-7226-371-4.
- HLAVENKA, J. Internetový marketing :Praktické rady, tipy, návody a postupy pro využití Internetu v marketingu. 1.vyd. Praha : Computer Press, 2001. 157 s. ISBN 80-7226-498-2.
- SMIČKA, R. Optimalizace pro vyhledávače - SEO :jak zvýšit návštěvnost webu. Vyd. 1. Kralice na Hané : Zásilkové knihkupectví J. Smičkové, 2004. 126 s. ISBN 80-239-2961-5.
- STUHLÍK, P. Marketing na Internetu. 1.vyd. Praha : Grada Publishing, 2000. 247 s. ISBN 80-7169-957-8.

Vedoucí diplomové práce: Ing. Petr Dydowicz, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2009/2010.

L.S.

PhDr. Martina Rašticová, Ph.D.
Ředitel ústavu

doc. RNDr. Anna Putnová, Ph.D., MBA

V Brně, dne 15.05.2010

Abstrakt

Diplomová práce se zabývá analýzou a návrhem distribuční B2B, B2C aplikace. Stěžejní část práce je zaměřena především na návrh vlastností a funkcí aplikace, volbu vhodných technologií a návrh datového skladu. Dále práce obsahuje návrh systému správy jednotlivých verzí a závazná pravidla pro vývoj a následnou údržbu a rozšiřování aplikace.

Abstract

The diploma thesis is concerned with an analysis and design of distributional B2B and B2C application. The fundamental part is especially focused on design of features and functions of the application, selection of suitable technologies and the project of the data storage. The diploma paper also contains a scheme of administration of particular versions and binding regulations for development and resulting maintenance and expanding of the application.

Klíčová slova

B2B, B2C, elektronický obchod, internetová aplikace, PHP, MySQL, AJAX, SEO, framework, MVC, VCS, projektový management

Keywords

B2B, B2C, e-commerce, internet application, PHP, MySQL, AJAX, SEO, framework, MVC, VCS, project management

Bibliografická citace VŠKP dle ČSN ISO 690

NOVÁK, J. *Analýza a návrh distribuční B2B, B2C aplikace*. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2010. 103 s. Vedoucí diplomové práce Ing. Petr Dydowicz, Ph.D.

Čestné prohlášení

Prohlašuji, že předložená diplomová práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 28. května 2010

.....

Poděkování

Na tomto místě bych chtěl poděkovat mému vedoucímu Ing. Petru Dydowiczovi, Ph.D. za odborné vedení, jeho ochotu, cenné rady, připomínky a konzultace. Také bych rád poděkoval všem vyučujícím na FP, od kterých jsem měl možnost mnohému se naučit.

Obsah

| | |
|---|----|
| Úvod..... | 9 |
| 1 Vymezení problému a cíle práce | 10 |
| 1.1 Metodika a pracovní postup..... | 10 |
| 2 Teoretická východiska práce..... | 12 |
| 2.1 E-commerce..... | 12 |
| 2.2 Aplikace pro internetové obchodování | 12 |
| 2.2.1 Hybridní systém na bázi B2B a B2C..... | 13 |
| 2.3 Význam a cíle internetového obchodování..... | 14 |
| 2.3.1 Výhody a nevýhody internetového obchodu..... | 14 |
| 2.4 Technologie pro návrh a vývoj aplikace..... | 15 |
| 2.4.1 PHP..... | 15 |
| 2.4.2 AJAX | 17 |
| 2.4.3 Framework..... | 17 |
| 2.4.4 Databázový systém MySQL..... | 19 |
| 2.4.5 Normalizace relačních databází..... | 19 |
| 2.4.6 MVC..... | 20 |
| 2.4.7 Stručný pohled do historie internetového obchodování..... | 23 |
| 2.4.8 Pohled do budoucnosti..... | 23 |
| 2.5 Technologie podporující vývoj a údržbu aplikace..... | 24 |
| 2.5.1 Revision Control System..... | 24 |
| 2.5.2 Centralizovaný systém pro správu verzí..... | 25 |
| 2.5.3 Distribuovaný systém pro správu verzí..... | 26 |
| 2.5.4 Principy programátora..... | 27 |
| 2.5.5 Coding standards..... | 28 |
| 2.6 Ekonomické a manažerské metody..... | 29 |
| 2.6.1 SMART..... | 29 |
| 2.6.2 Ganttův diagram..... | 30 |
| 2.6.3 SWOT analýza..... | 30 |
| 3 Analýza problému a současné situace | 31 |
| 3.1 Představení společnosti..... | 31 |
| 3.2 Reference a zákazníci..... | 32 |
| 3.3 Nabízené služby..... | 32 |

| | |
|--|----|
| 3.4 SWOT analýza společnosti..... | 33 |
| 3.5 Analýza současné internetové B2B, B2C aplikace | 34 |
| 3.5.1 SmartShop..... | 35 |
| 3.5.2 ProfiShop..... | 36 |
| 3.5.3 IndividualShop..... | 36 |
| 3.5.4 Technologie současné aplikace..... | 37 |
| 3.6 SWOT analýza stávající B2B, B2C aplikace..... | 39 |
| 3.7 Cílová skupina a účel aplikace..... | 40 |
| 3.7.1 Pro koho je aplikace určena..... | 40 |
| 3.7.2 Účel B2B, B2C aplikace..... | 41 |
| 4 Vlastní návrhy řešení | 42 |
| 4.1 Specifikace veřejného rozhraní nové B2B, B2C aplikace..... | 42 |
| 4.1.1 Typy uživatelů veřejné části aplikace..... | 42 |
| 4.1.2 Katalog..... | 43 |
| 4.1.3 Detail produktu..... | 45 |
| 4.1.4 Akční nabídky, novinky..... | 45 |
| 4.1.5 Vyhledávání..... | 46 |
| 4.1.6 Nákupní košík..... | 46 |
| 4.1.7 Objednávka..... | 47 |
| 4.1.8 Uživatelský účet..... | 48 |
| 4.2 Specifikace administračního rozhraní nové B2B, B2C aplikace..... | 49 |
| 4.2.1 Typy uživatelů administračního rozhraní aplikace..... | 49 |
| 4.2.2 Správa katalogů..... | 50 |
| 4.2.3 Správa produktů..... | 52 |
| 4.2.4 Správa atributů/parametrů..... | 54 |
| 4.2.5 Správa skladového systému..... | 54 |
| 4.2.6 Správa výrobců..... | 55 |
| 4.2.7 Expediční lhůty, záruční lhůty..... | 55 |
| 4.2.8 Správa komentářů a hodnocení produktů..... | 56 |
| 4.2.9 Správa zákazníků..... | 56 |
| 4.2.10 Správa objednávek..... | 57 |
| 4.2.11 Správa obsahu..... | 59 |
| 4.2.12 Správa uživatelských rolí..... | 61 |

| | | |
|--------|--|-----|
| 4.2.13 | Statistické a analytické nástroje..... | 63 |
| 4.2.14 | Bezpečnost aplikace a logování událostí..... | 64 |
| 4.2.15 | Ostatní funkce..... | 67 |
| 4.2.16 | Grafické rozhraní administrační části aplikace..... | 67 |
| 4.3 | PHP Framework jako jádro aplikace..... | 69 |
| 4.3.1 | CakePHP..... | 69 |
| 4.3.2 | Nette Framework..... | 70 |
| 4.3.3 | Zend Framework..... | 72 |
| 4.3.4 | Komponenta datagrid..... | 74 |
| 4.3.5 | Multijazyčnost aplikace..... | 76 |
| 4.4 | Návrh datového skladu..... | 80 |
| 4.4.1 | Návrh datového skladu pro správu katalogu..... | 81 |
| 4.4.2 | Návrh datového skladu pro správu produktů..... | 82 |
| 4.4.3 | Návrh datového skladu pro správu objednávek..... | 83 |
| 4.4.4 | Návrh datového skladu pro správu zákazníků a uživatelských rolí..... | 84 |
| 4.5 | Systémy pro správu verzí (VCS)..... | 85 |
| 4.5.1 | SVN neboli Subversion..... | 86 |
| 4.5.2 | Git..... | 87 |
| 4.5.3 | Mercurial | 88 |
| 4.6 | Závazná pravidla pro vývoj aplikace..... | 89 |
| 4.6.1 | Zdrojové kódy..... | 89 |
| 4.6.2 | Správa verzí..... | 89 |
| 4.7 | Ekonomické zhodnocení a přínosy nového řešení..... | 90 |
| 4.7.1 | Ekonomické vyjádření a harmonogram realizace..... | 91 |
| 4.7.2 | Přínosy nového řešení..... | 93 |
| 4.8 | Strategie zavedení aplikace do praxe..... | 94 |
| | Závěr..... | 95 |
| | Seznam použité literatury..... | 97 |
| | Monografické zdroje..... | 97 |
| | Elektronické zdroje..... | 98 |
| | Seznam obrázků..... | 100 |
| | Seznam příloh..... | 101 |

Úvod

Internet je v dnešní době nejrozsáhlejší počítačovou sítí světa a ve vyspělých zemích již patří mezi základní životní standard. Stanovit počet uživatelů internetu lze jen velmi hrubými odhady, jelikož se toto číslo neustále každým dnem zvyšuje. Současná společnost využívá internet především pro komunikaci, obchodování, zábavu a sdílení informací. A právě internetové obchodování a zábava ženou internet velmi významnou silou neustále kupředu.

On-line obchodování z pohodlí svého domova již dnes využívá poměrně velká skupina internetových uživatelů. Prostřednictvím internetu se tak nakupují věci běžné denní spotřeby až po věci nebo služby ne vždy zcela běžně dostupné. O jednotlivých produktech a službách si lze prostřednictvím internetu zjistit ve většině případů mnohem více informací, než v klasickém kamenném obchodě.

Velmi často se dnes říká, že kdo není na síti jako by neexistoval. A speciálně pro podnikatelské jednotky nabízí internet nový rozměr obchodování a spoustu možností a příležitostí jak dostat své produkty a služby ke konečnému zákazníkovi.

Výsledkem této práce a několika následných obchodních procesů vznikne nástroj, pomocí kterého budou moci další podnikatelské subjekty rozšířit své působení v oblasti internetového obchodování.

1 Vymezení problému a cíle práce

Internetové B2B, B2C aplikace jsou v dnešní době považovány za moderní způsob prodeje výrobků a služeb. Většina podnikatelských jednotek provozuje internetovou aplikaci jako doplněk klasického kamenného obchodu. Ale stále se zvětšuje poměr těch, pro které je internetové obchodování primární činností.

Profesionálně vedený a neustále aktualizovaný internetový obchod je výborným prostředkem, jak s minimálními náklady oslovit široké spektrum potencionálních zákazníků a nabídnout jim své produkty a služby.

Cílem diplomové práce je provést analýzu současného stavu a zpracovat návrh B2B, B2C aplikace, která bude následně implementována a poté nabízena podnikatelským subjektům jako prostředek pro podporu a realizaci internetového obchodování, zvyšování konkurenceschopnosti a loajality zákazníků.

Navržená aplikace musí plně vyhovovat potřebám stávajících i potencionálních zákazníků, mezi které patří především nízká cena, snadná a rychlá správa aplikace s minimalizací manuálních operací, rychlost, modifikovatelnost a budoucí rozšiřitelnost, vysoký důraz na uživatelské rozhraní a interakci se zákazníkem, kompletní multijazyčnost a další.

Vhledem k tomu, že každá podnikatelská jednotka má ještě ve většině případů individuální požadavky, je třeba aplikaci navrhnout tak, aby bylo možné s co nejnižšími náklady a maximální efektivitou tyto požadavky zpracovat a aplikaci nasadit. Tomu má dopomoci především správná volba technologie, stanovení závazných pravidel pro vývoj a údržbu aplikace, ale také vhodná správa verzí, která zefektivní nejen samotný vývoj, ale i následnou údržbu a rozšiřování. Aplikace musí být navržena tak, aby byla schopna naplno využívat moderních a v současné chvíli dostupných prostředků a technologií.

1.1 Metodika a pracovní postup

Metodika a pracovní postup zpracování diplomové práce bude probíhat v následujících krocích. Mezi první kroky bude patřit studium odborné literatury a nashromáždění

potřebných informací z odborné literatury, ale i z ostatních volně dostupných zdrojů. Následovat bude analýza současného stavu společnosti a stávající aplikace. Na základě výsledků této analýzy bude realizována stěžejní část diplomové práce, při které budou využívány především metody modelování, optimalizace, srovnávání a popisu. Jednotlivé metody budou využívány samostatně, případně se budou prolínat.

Pomocí metody popisu budou postupně zpracovány a popsány požadavky na jednotlivé funkce a vlastnosti aplikace a jejich vzájemné souvislosti. Podrobně popisovány budou vlastnosti a funkce jak veřejného, tak i administračního rozhraní aplikace a jejich vzájemná spolupráce.

Především pomocí metod srovnávání pak bude volena vhodná technologie pro implementaci aplikace a správu jednotlivých verzí této aplikace. Budou definovány závazná pravidla pro vývoj a údržbu aplikace. Pro návrh datového skladu budou využity metody datového a funkčního modelování. Při návrhu datového skladu, ale i dalších částí aplikace budou využity i metody optimalizace. Optimalizace bude ve velkém rozsahu využíváno především pro podporu SEO optimalizace pro vyhledávače a dále pro optimalizaci uživatelského rozhraní, výkonnosti a rychlosti celé aplikace.

V závěru práce dojde k sestavení předběžného rozpočtu na aplikaci spolu se stanovením časové a personální náročnosti a také dojde k sepsání jednotlivých přínosů nového řešení.

2 Teoretická východiska práce

Obsah této kapitoly se zaměřuje na teoretická východiska práce, na základě nichž je posléze prováděna analýza a vlastní návrh řešení. Jednotlivé podkapitoly jsou věnovány teoretickým poznatkům týkajících se dané problematiky a obsahují popis použitých pojmů, metod, technologií, postupů a také zdůvodnění jejich použití.

2.1 E-commerce

E-commerce, neboli elektronické podnikání slouží k realizování obchodních transakcí prostřednictvím internetu. Do této oblasti patří vše od marketingové a obchodní komunikace, elektronické on-line platby, elektronické výměny a sběru dat, až po samotný prodej zboží, služeb, ale i informací. Stejně tak jako v jiných odvětví obchodování jsou i zde prováděny veškeré kroky vedoucí k dosažení zisku, zvyšování obratu, zvyšování obchodního podílu na trhu, zvyšování klientské základny, úspore nákladů, tvorbě a zvyšování image společnosti, loajalitě zákazníků a dalších.

Prostřednictvím e-commerce dochází nejen k oboustrannému přesunu informací, ale i samotnému uzavírání obchodních smluv a to vše v prostředí internetu, tedy elektronicky. Právě kvůli internetu se jedná o relativně mladou oblast podnikání, která stále prochází poměrně živým vývojem ať již na poli právním, nebo technologickém.

Podmnožinou elektronického podnikání (e-commerce) je elektronické obchodování, které se dá specifikovat jako „zajištění obchodních aktivit podniku prostřednictvím nejrůznějších informačních a komunikačních technologií“ (3), případně jako „výměnu informací po elektronickém médiu za účelem uzavření obchodu nebo k jeho podpoře“.

(3)

2.2 Aplikace pro internetové obchodování

Aplikace pro internetové obchodování se člení na několik základních modelů, jejichž názvy vycházejí ze vzájemných vztahů mezi prodávajícím a kupujícím. Nejčastěji vyskytující typy jsou tyto:

- ✓ B2B neboli *business to business* – základním znakem tohoto typu obchodování je

vzájemná znalost obou stran (prodávající zná kupujícího) – tento typ se uplatňuje především v distribučních a prodejních sítích – jeden obchodník prodává druhému obchodníkovi

- ✓ B2C neboli *business to customer* – při tomto typu obchodování není znám kupující, je zaměřen na prodej koncovému zákazníkovi, spotřebiteli, nebo alespoň jeho podporu
- ✓ C2C neboli *customer to customer* – v tomto typu se nezná ani jedna ze stran, nachází se například na internetových aukcích (8)

Prostřednictvím aplikace pro internetové obchodování tedy dochází nejen k nabízení zboží, informací o zboží, či poskytování služeb, ale i následnému zprostředkování prodeje nabízeného zboží nebo služby spolu s dalšími souvisejícími službami jako je reklama, doprava, rozšířené služby apod.

Internetový obchod je dnes nejčastějším nástrojem používaným pro elektronické podnikání neboli e-business a je alternativou ke kamennému obchodu. V některých segmentech trhu kamenný obchod téměř nahrazuje a v budoucnosti tomu nebude jinak.

Aplikace navrhovaná v této práci se bude zaměřovat pouze na typy se vztahem B2B a B2C. Především typ B2C měl doposud nejvyšší zastoupení mezi prodanými aplikacemi společností XYZ, s.r.o.

2.2.1 Hybridní systém na bázi B2B a B2C

Poměrně často se dnes setkáváme s požadavkem na tuto kombinaci systémů, která je poskytována prostřednictvím jediné aplikace. Cílem tohoto řešení je správné vyvážení aplikace tak, aby plně uspokojovala obě cílové skupiny. Na návrh jádra tohoto typu aplikace jsou kladeny vyšší požadavky než na oddělené systémy, avšak většina menších klientů, která provozuje svoji obchodní činnost jak prostřednictvím modelu B2B i B2C požaduje sloučení obou modelů elektronického obchodování. Toto sloučení obhajují především snadnější údržbou a také očekávají nižší cenu.

2.3 Význam a cíle internetového obchodování

Hlavním cílem z pohledu provozovatele internetového obchodu, je zvýšit poptávku po nabízených produktech či službách, snížení nákladů na marketing a samotný prodej. Internetový obchod lze ve srovnání s kamenným obchodem provozovat při mnohem nižších nákladech. Nemałym významem internetového obchodu je i prezentace na globálním trhu, kde nabízené produkty či služby mohou vidět milióny návštěvníků.

Hlavním významem z pohledu zákazníka je především neustálá dostupnost internetového obchodu, jeho jednoduchost a možnost realizovat nákup z pohodlí domova.

2.3.1 Výhody a nevýhody internetového obchodu

Hlavní výhody internetového obchodu ze strany provozovatele

- ✓ nižší náklady na provoz v porovnání s kamenným obchodem – odpadají různé transakční náklady, nižší míra prostředníků v prodejním řetězci
- ✓ prezentace globálnímu trhu – nakupovat může kdokoliv a odkudkoliv
- ✓ internetový obchod pracuje 24 hodin denně
- ✓ rostoucí obliba internetového nakupování

Hlavní výhody internetového obchodu ze strany zákazníka

- ✓ úspora času – možnost nákupu kdykoliv, protože obchod nikdy nezavírá
- ✓ úspora financí – nižší ceny díky nižšímu rozpětí nákladů
- ✓ větší informovanost – ke každému produktu lze dohledat recenze, zkušenosti, zákazník není závislý pouze na jednom informačním zdroji (prodavači)
- ✓ vysoké pohodlí – vše z domova, zboží může být dopraveno téměř kamkoliv

Hlavní nevýhody internetového obchodu ze strany zákazníka

- ✓ nemožnost prohlédnutí výrobku
- ✓ omezené způsoby placení – dobírka, platba převodem, jiné se teprve rozšiřují

- ✓ zboží není k dispozici ihned po nákupu
- ✓ anonymita prodeje (8)

2.4 Technologie pro návrh a vývoj aplikace

V dnešní době se pro vývoj internetových aplikací používá několik základních technologií. Od ASP.NET, Ruby on Rails, Javu, až po asi nejrozšířenější a nejdostupnější PHP. Tyto jazyky pak nejčastěji spolupracují s databázovými systémy využívajícími jazyka SQL. Nejvyužívanějším databázovým partnerem PHP je MySQL. Ale stále více se začínají používat i jiné databázové systémy jako jsou PostgreSQL, MS SQL, Oracle, SQLite a jiné.

Nedílnou součástí dnešních aplikací jsou JavaScript a AJAX, jejichž využívání v současných aplikacích razantně vzrůstá. Spolu s daty ve formátu JSON, či XML napomáhají uživatelskému rozhraní a komplexnímu ovládnutí aplikace.

Co se týká zobrazovací části, zde se již delší dobu využívá jazyka (x)HTML, spolu s kaskádovými styly - CSS. V této oblasti by v blízké budoucnosti mohlo dojít po delší době k větším změnám. Očekává se schválení a rozšíření HTML5, což je rozvinuté HTML tak, aby splnilo dnešní požadavky webových aplikací.

V následujících podkapitolách se budu věnovat technologiím, které budou pohánět navrhovanou aplikaci. Jedná se především o PHP, MySQL, JavaScript, AJAX, HTML, CSS a frameworky na těchto technologiích postavené.

2.4.1 PHP

PHP je dnes velmi rozšířená technologie umožňující snadné programování na straně serveru tzv. server-side programming. Stručně lze říci, že skript napsaný v PHP je proveden na serveru podle zadaných kritérií a výsledek je odeslán volajícímu počítači stejným způsobem, jakým se odesílají běžné statické HTML stránky.

PHP podporuje řadu internetových protokolů – HTTP, FTP, POP3, SMTP, IMAP, LDAP a další, dále dokáže komunikovat s velkou většinou dnes používaných databázových serverů např. MySQL, MSSQL, Oracle, PostgreSQL, ODBC, SQLite a další. Obsahuje

taktěž velké množství rozsáhlých knihoven pro zpracování textů, grafiky a práci se soubory.

Toho lze využít k tvorbě různých interaktivních webových stránek a aplikací, které umožňují např. provádění složitých kalkulací, objektově-orientované programování, spolupráci s databázemi, generování obrázků, grafů a dalších. PHP spadá do skupiny Open Source.

Apache, MySQL a PHP je dnes jedna z nejoblíbenějších a nejvyužívanějších kombinací pro provozování internetových aplikací. Najdeme ho totiž na většině serverů po celém světě. Nejrozšířenější verzí PHP je v současnosti verze 5.2.x.

PHP nebylo v počátcích koncipováno jako objektový jazyk. S verzí PHP 5 ovšem došlo k velké změně a podpora objektů byla implementována. S vyššími verzemi postupně narůstají pokročilejší mechanismy a funkce OOP. V červnu roku 2009 byla vydána verze PHP 5.3, která obsahuje zatím asi nejvíce změn ze všech minoritních verzí. Tato verze není prozatím na serverech hostingových společností moc rozšířena, ale to se s vydáním verze 5.3.2 (duben 2010) pravděpodobně změní. U počátečních verzí se obvykle vyskytují problémy, které jsou postupně odstraňovány. A velké množství změn obsažených v této verzi toto nasazení taktěž opozdilo.

Novinky, které PHP 5.3 přináší:

- ✓ jmenné prostory
- ✓ anonymní funkce
- ✓ late static binding
- ✓ `__callStatic`
- ✓ nový garbage collector
- ✓ příkaz `goto`
- ✓ nové rozšíření
- ✓ výkonnostní změny, změny v `php.ini` a spousta dalších

Pro vývoj B2B, B2C aplikace budeme využívat již PHP ve verzi 5.3.

2.4.2 AJAX

AJAX neboli *Asynchronous JavaScript and XML* je ve své podstatě směs technologií, které nám umožňují zbavit se aktualizací stránek, jež představují „mrtvý“ čas při navigaci mezi stránkami. Eliminace aktualizací stránek je však pouze jedním krokem z mnoha, které zavádějí do webových prezentací další vlastnosti jako je např. kontrola dat v reálném čase, technologie přetahování myši (drag-and-drop) a mnoho dalších, které dříve nebyly s webovými aplikacemi spojovány.

AJAX tedy spojuje server-side technologii spolu s client-side verzí. Dokáže v pozadí zpracovat skript na serveru, který je následně odeslán do webového prohlížeče, který jej zobrazí. (8)

2.4.3 Framework

Frameworkem můžeme nazvat softwarovou strukturu, která obsahuje pomocné knihovny, funkce, podpůrné programy a skripty, návrhové vzory a další v podobě kvalitního API. Většina z nich obsahuje také doporučené postupy pro vývoj samotných aplikací. Celý framework pak slouží jako podpora při vývoji nové aplikace. Framework je ve většině případů napsaný ve stejném programovacím jazyce, ve kterém pak vzniká nová aplikace.

Účelem frameworku je především usnadnění práce programátora a zvýšení efektivity jeho práce. Framework ve většině případů také zvyšuje bezpečnost samotné aplikace a její udržitelnost a rozšiřitelnost – i když tyto vlastnosti samozřejmě závisí i na samotném programátorovi a jeho znalostech a zkušenostech. Nemalou výhodou frameworků je i to, že podporují týmovou spolupráci – právě tím kvalitním API.

Když se podíváme na klasickou internetovou aplikaci, zjistíme, že se v každé z nich neustále opakují ty stejné části, funkce, metody a postupy. Právě na tyto části se frameworky zaměřují a snaží se je co nejaktuálnějšími a nejefektivnějšími způsoby řešit. Programátor se pak těmito částmi nemusí vůbec zabývat a soustředí se na stěžejní část aplikace. Mezi takovéto části v prostředí internetových aplikací může patřit např. přihlašování uživatelů, uživatelské role, generování url adres, práci s databázemi, šablonovací systém, cachování dat, formuláře a jejich validace a další. Při samotném

vývoji pak nabízí kvalitní ladící a debugovací nástroje, které dokáží ušetřit také nemálo času. Důraz je kladen na znovupoužitelnost jednotlivých kódů.

Takže zjednodušeně se dá říci, že např. při vývoji internetové aplikace pro banku se programátoři mohou zaměřit na to, jak budou prováděny např. bankovní transakce a nemusí řešit navigaci mezi jednotlivými stránkami apod.

V současné době se nejvíce mluví o tzv. Frameworkích pro webové aplikace (Web application framework). S největší pravděpodobností je to dáno velkým rozmachem internetových aplikací a jejich neustálým vývojem, ale také možná určitou nedokonalostí, či nedotažeností skriptovacích a programovacích jazyků (např. PHP).

Mezi nejznámější a nejpoužívanější frameworky pro internetové aplikace patří:

- ✓ Zend Framework
- ✓ Akelos
- ✓ CodeIgniter
- ✓ CakePHP
- ✓ Kohana
- ✓ Symfony
- ✓ Nette Framework
- ✓ Ruby on Rails
- ✓ Yii PHP framework

Rozhodně se nejedná o kompletní seznam existujících frameworků, ten je mnohem rozsáhlejší a neustále vznikají nové a nové. Ve zmíněném seznamu se nacházejí pouze frameworky, o kterých máme ve firmě nějaké povědomí, a jeden z nich bude vybrán pro vývoj nových aplikací. Výběru samotného frameworku se v této práci věnuji v dalších kapitolách ve vlastních návrzích řešení. (4), (14), (20), (29)

2.4.4 Databázový systém MySQL

MySQL je multiplatformní databázový systém, který v nedávné době skoupila společnost Sun Microsystems, což je dceřiná společnost databázového gigantu Oracle Corporation, která se zabývá vývojem nástrojů pro vývoj a správu databází.

Databázový systém MySQL je díky své rychlosti, vysokému výkonu, jednoduchosti a především multiplatformnosti dostupný na velké většině serverů a v dnešní době je hojně využíván na široké škále internetových aplikací malého až středního rozsahu. Dnešní MySQL má v sobě již implementovanou funkcionalitu, kterou dříve mnoho programátorů postrádalo. Jedná se především o transakce, procedury, trigger, pohledy, partitioning, časování událostí (Event Scheduler) a mnoho dalších. Současná a taktéž již nejpoužívanější verze MySQL systému je verze řady 5. Mnou navrhovaná internetová B2B, B2C aplikace bude tuto verzi taktéž vyžadovat.

2.4.5 Normalizace relačních databází

Normalizace je sada pravidel, dle kterých bychom měli postupovat při transformaci struktury entit a relací na strukturu fyzického uspořádání tabulek a relací v databázi. Pomocí procesu nazývaného normalizace odstraníme z databáze všechny zbytečnosti a jiné potenciační problémy, které nabourávají, nebo v budoucnu mohou nabourat integritu dat.

Relační databázový model uspořádává data do tzv. relací (tabulek), které obsahují n-tice (řádky). Tabulky (relace) tvoří základ relační databáze. Tabulka je struktura záznamů s pevně stanovenými položkami (sloupci - atributy). Každý sloupec má definován jednoznačný název, typ a rozsah (doménu). Každý záznam se pak stává n-ticí (řádkem) tabulky. Pokud jsou v různých tabulkách sloupce stejného typu, pak tyto sloupce mohou vytvářet vazby mezi jednotlivými tabulkami.

Relační model přináší celou řadu výhod zejména mnohdy přirozenou reprezentaci zpracovávaných dat, možnost snadného definování a zpracování vazeb apod. (8)

První normální forma

První normální forma nám říká, že relace je v první normální formě, pokud každý její

atribut obsahuje jen atomické hodnoty. Tedy hodnoty z pohledu databáze již dále nedělitelné.

Druhá normální forma

Relace se nachází v druhé normální formě, jestliže je v první normální formě a každý neklíčový atribut je plně závislý na primárním klíči, a to na celém klíči a nejen na nějaké jeho podmnožině. Z čehož vyplývá, že druhou normální formu musíme řešit pouze v případě, že máme vícehodnotový primární klíč.

Třetí normální forma

V této formě se nachází tabulka, splňuje-li předchozí dvě formy a žádný z jejich atributů není tranzitivně závislý na klíči. Jiné vyjádření téhož říká, že relace je ve třetí normální formě, pokud je ve druhé normální formě a všechny neklíčové atributy jsou navzájem nezávislé. Tranzitivní závislost je taková závislost, mezi minimálně dvěma atributy a klíčem, kde jeden atribut je funkčně závislý na klíči a druhý atribut je funkčně závislý na prvním.

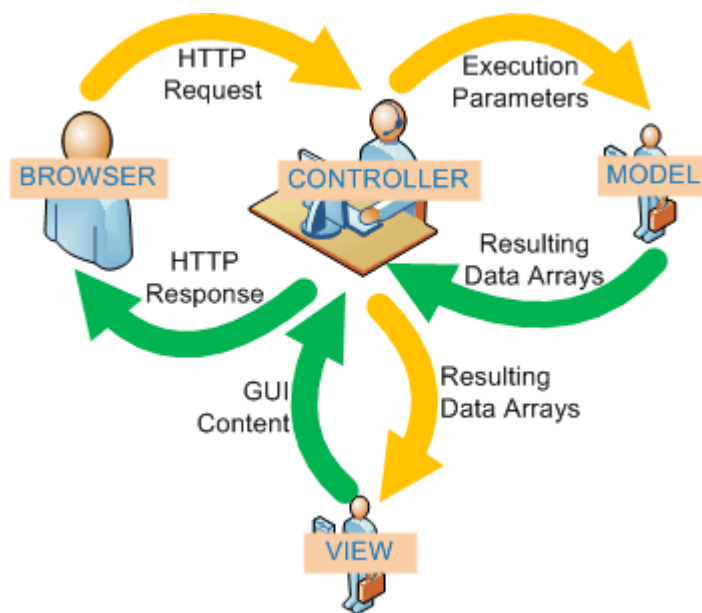
Ne vždy je vhodné normalizaci provádět v plném rozsahu. Samotná normalizace obsahuje ještě další formy, nicméně při návrhu databázové struktury předpokládám využití pouze prvních třech normálních forem. (8)

2.4.6 MVC

„Softwarová architektura MVC (Model-view-Controller) v posledních letech dramaticky nabírá na popularitě, což je nejspíš způsobeno tím, že jednotlivé technologie již mají ranou fázi vývoje za sebou a vedle „základní výbavy“ se tak snaží nabídnout i prostředky pro tvorbu aplikací s kvalitní architekturou.“ (13)

V tomto případě architektury aplikace se jedná o způsob rozdělení aplikace, aplikačních dat, procesů i datových toků do logických celků, stanovení struktury těchto komponent, vzájemných vztahů a interakcí mezi nimi, a to na dostatečně obecné úrovni.

Z tohoto důvodu vznikají, nebo již nějakou dobu existují frameworky, které slouží jako jádro aplikací. Tyto frameworky ve velké většině využívají právě architekturu MVC.



Obrázek 1: Architektura MVC (Model-View-Controller)

Architektura MVC dělí aplikaci na 3 logické části tak, aby je šlo upravovat samostatně a dopad změn byl na ostatní části co nejmenší. Tyto tři části jsou Model, View a Controller. Model reprezentuje data a business logiku aplikace, View zobrazuje uživatelské rozhraní a Controller má na starosti tok událostí v aplikaci a obecně aplikační logiku. (4), (11), (13), (16), (21)

Model

„Model je funkčním a datovým základem celé aplikace. Poskytuje prostředky jak pro přístup k datové základně a stavům aplikace, tak pro jejich ukládání a aktualizaci. Hovoří se o něm jako o softwarovém obrazu procesů reálného světa. Měl by být jako celek zapouzdřený a pro View a Controller nabízet přesně definované rozhraní.

Nejčastěji je implementován pomocí objektových tříd, lze jej však (v přístupech odlišných od OOP) realizovat například i běžnými funkcemi. Na své nižší vrstvě je samozřejmě tvořen nějakou formou úložiště dat.“ (28)

View

„View zobrazuje obsah Modelu, zajišťuje grafický či jiný výstup aplikace. Přes Model

přístupuje k datům a stavům aplikace a specifikuje, jak mají být prezentovány.

Při změně stavu Modelu aktualizuje zobrazení. Změny v Modelu získává buď push přístupem, kde se jenom zaregistruje u Modelu a ten pak sám posílá View notifikace o změnách. Nebo pull přístupem, kdy se View obrací samo na Model vždy, když potřebuje získat nejaktuálnější data.

V případě stand-alone aplikace zajišťuje View zpravidla průběžné překreslování a aktualizaci obsahu zobrazených oken. U webových aplikací generuje View příslušný HTML kód, který je odeslán prohlížeči jako odpověď na požadavek.

Jestliže slouží zobrazený výstup zároveň jako oblast pro zachytávání událostí od uživatele (například kliknutí myši do vykresleného okna), předává View tyto události Controlleru.“ (28)

Controller

„Controller definuje chování aplikace. Zpracovává veškeré vstupy a události pocházející od uživatele. Na jejich základě volá příslušné procesy Modelu, mění jeho stav apod. Podle událostí přijatých od uživatele i podle výsledků akcí v Modelu pak vybírá Controller vhodné View pro další zobrazení.“ (28)

Výhody MVC

- ✓ vysoká komplexnost návrhu a snadná budoucí rozšiřitelnost aplikace, efektivní modularita,
- ✓ znovupoužitelnost kódu. Jako Model lze ve vlastní aplikaci použít standardní knihovny či třídy,
- ✓ rozdělení vývojářských rolí. Jednotlivé části mohou být vyvíjeny samostatně, jen se znalostí předem určených rozhraní mezi nimi. Minimalizuje se dopad modifikací, změny jsou většinou prováděny jen v dané vrstvě,
- ✓ minimalizace duplicitního kódu,
- ✓ čistota a přehlednost výsledného kódu aplikace, snadnější údržba (28)

2.4.7 Stručný pohled do historie internetového obchodování

Když se podíváme do historie internetového obchodování, tak první nákupy prostřednictvím internetu se uskutečnily v 90. letech 20. století na území USA. V Evropě se internetové obchody začaly objevovat o něco později a to především díky technické zaostalosti, nedůvěřivosti zákazníků, ale i pomalejšímu rozšíření platebních karet a on-line transakcí.

První internetové obchody podobné dnešním začaly vznikat až s velkým rozmachem http:// protokolu po letech 1994 – 1995. V těch dobách vznikl také jeden z průkopníků a stálic internetového obchodování Amazon.com. V České republice se internetové obchodování začalo rozšiřovat až po roce 2000. Český zákazník byl v tomto směru poměrně nedůvěřivý.

Nyní zákazníci čekají více než jen dobře odladěnou prodejní internetovou aplikaci s pěknými obrázky, podrobnými informacemi a objednááním dopravy. Samozřejmostí by měla být perfektní logistika a profesionální poprodejní služby. Zákazníky dále zajímá možnost snadné reklamace, možnost vrácení zboží, servisu a dalších služeb, které poskytují kamenní prodejci. Chtějí mít jistotu, že zakoupením zboží se o ně prodejce nepřestane starat.

Když se podíváme do historie z pohledu technologií, první internetové obchody byly tzv. statické – tzn. pro každou stránku byl ručně sepsán zdrojový kód v jazyce HTML. Toto řešení bylo zdoluhavé a údržba a aktualizace těchto zdrojových kódů byla časově velmi náročná. To je při dnešním množství prodávaných položek a jejich intenzitě obměny v některých obchodech téměř nepředstavitelné. Dnešní internetové aplikace jsou ve většině případů generovány dynamicky a tím pádem se dají poměrně snadno a jakkoliv upravovat, samotná data exportovat apod. Datům v databázích nevádí ani častá změna designu internetové aplikace. (23)

2.4.8 Pohled do budoucnosti

V nejbližší době se neočekává nějaká závratná změna v koncepci internetového obchodování. Snad jen ve vzdálenější budoucnosti, kde se mluví např. o nakupování prostřednictvím eye-tracking, kde se bude využívat oční kamera, nebo jiné zařízení,

kteřé bude sledovat pohyb lidského oka. Ovládání a nakupování by pak mohlo probíhat prostřednictvím této technologie.

S největší pravděpodobností se tak bude nejvíce měnit především technologická část aplikací a dále kvalita a množství doplňkových služeb pro zákazníky. Změny na technologické úrovni budou podporovat především uživatelské rozhraní a míru interakce se zákazníkem. V důsledku stále se zvyšující konektivity a navyšování její rychlosti může být zákazníkovi poskytován daleko širší objem informací (video produktů, video-recenze a další). Asi nejzajímavějším tématem pro provozovatele internetových obchodů by mohlo být propojení aplikací se sociálními sítěmi jako je Facebook apod. Tyto aplikace využívají denně milióny uživatelů. (30)

2.5 Technologie podporující vývoj a údržbu aplikace

V této podkapitole se zaměřím na technologie, které souvisí s vývojem samotné aplikace, její následnou udržovatelností a rozšiřitelností. Na základě dosavadních zkušeností se stávajícími aplikacemi musí v tomto směru dojít ve společnosti XYZ, s.r.o. k radikální změně.

2.5.1 Revision Control System

Revision Control System (RCS) je systém sloužící pro uchovávání kompletní historie jakýchkoliv digitálních informací. Verzovat tedy lze jakákoliv digitální data – počítačové soubory.

RSC jsou nejčastěji využívány při vývoji software. Dokáží zaznamenat veškeré změny prováděné ve zdrojových kódech během celého stádia vývoje software. V průběhu vývoje software se lze k jednotlivým verzím (i dávno smazaným) kdykoliv pohodlně vrátit, nebo je jakkoliv využít či modifikovat.

„V praxi si to lze představit tak, že si RCS eviduje údaje typu kdo, kdy a jakým způsobem upravil zdrojové kódy programu. To slouží nejenom k úplnému přehledu všech změn po celou dobu vývoje, ale také možnost zjistit přesný stav sledovaných dat kdykoliv v minulosti. Samozřejmostí je možnost vrátit se k předchozí verzi daného programu v případě, že během dalšího vývoje dojde k chybám. Každé změně provedené

ve zdrojových kódech je přidělováno unikátní číslo, označované většinou jako číslo revize.“ (19) Systémy pro správu většinou neuchovávají úplný stav každé revize, ale pouze rozdíly mezi jednotlivými revizemi. Informační hodnota je stejná a data jsou velmi malá.

Obrovskou výhodou verzovacích systémů je pak možnost kooperace velkého množství vývojářů na jednom projektu. RCS si sám hlídá a řeší situace, kdy dva a více programátorů současně mění stejnou část zdrojového kódu. Pokud systém situaci dokáže vyřešit sám, vše proběhne bez zapojení programátora, jinak musí změny potvrdit, případně upravit sám programátor.

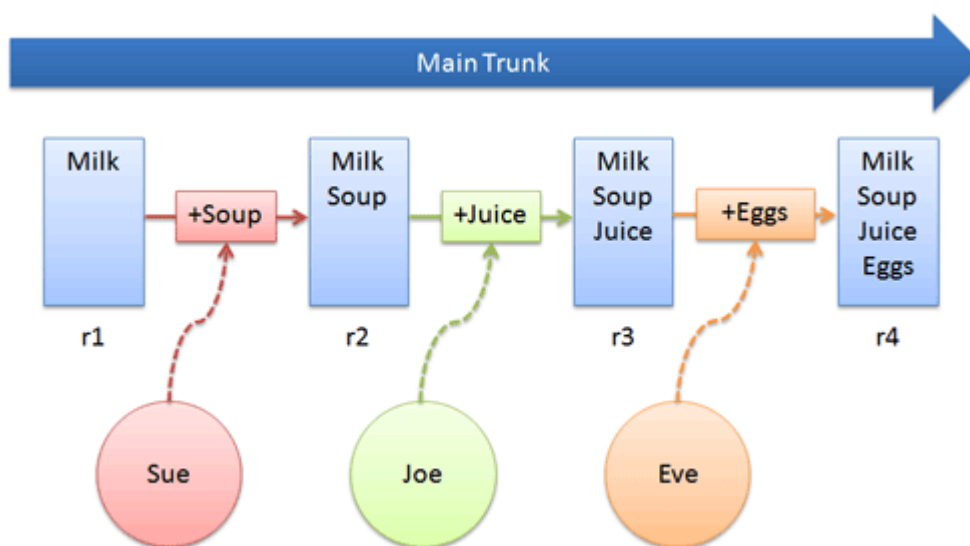
Z výše zmíněných vlastností jasně plynou výhody používání verzovacích systémů. A vyvíjet jakýkoliv větší projekt v týmu se bez použití RCS dá (i vzhledem k dostupnosti) považovat za velmi odvážné a velmi neefektivní.

V současné době existují dva typy systémů pro správu verzí. Prvním z nich je centralizovaný verzovací systém, který je známý a využíván již poměrně dlouho. Jeho princip fungování spočívá v neustálé komunikaci se severem. Naproti tomu existuje distribuovaný verzovací systém, kde může mít každý vývojář kopii celé historie dat lokálně. Tento systém umožňuje ve většině případů rychlejší práci, neboť odpadá neustálá komunikace se serverem.

Většina systémů pro správu verzí spadá do kategorie tzv. open-source software, neboli software s otevřeným kódem. S jistým omezením plynoucím z licenčních smluv daného software se dá říci, že jsou zdarma dostupné a tak náklady na samotný software jsou nulové. (15), (19)

2.5.2 Centralizovaný systém pro správu verzí

Nejznámějšími představiteli centralizovaného verzovacího systému jsou CSV (Comma separated values) a především Subversion.

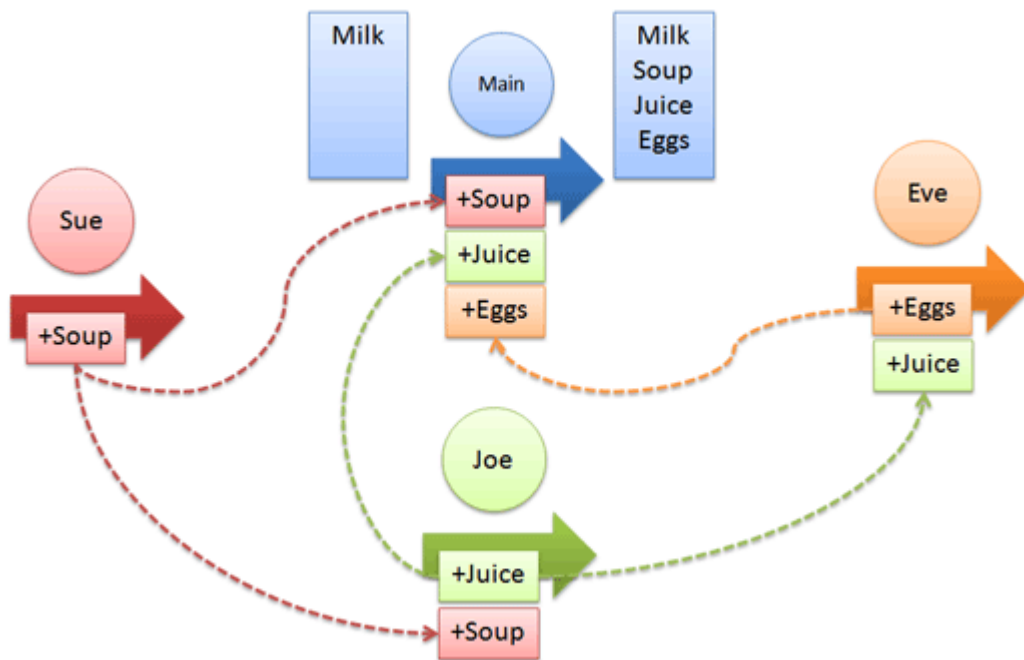


Obrázek 2: Schéma fungování centralizovaného verzovacího systému

Princip fungování centralizovaného systému je znázorněn na obrázku výše. Na serveru je uložena kompletní historie. Každý uživatel má u sebe pouze jednu aktuální verzi. Při jakékoliv změně, pohledu do historie repozitáře se musí uživatel připojit na server. Obrázek dále znázorňuje spolupráci více uživatelů. Ta probíhá např. tak, že uživatel si stáhne aktuální verzi z repozitáře. Lokálně provádí změny, které se mu neverzují. Když je spokojen s výsledkem, musí jej odeslat zpět na server, aby si jej mohli stáhnout ostatní. Sdílet aktuálně rozpracované verze přímo mezi uživateli není přímo možné. Podrobněji se na výhody a nevýhody toho typu zaměřím v dalších kapitolách, při výběru nejlepšího řešení pro naše projekty. Rozhodně se ale jedná o velmi efektivní, spolehlivý a hlavně velmi rozšířený systém. Mnoho významných projektů využívá jako RCS právě Subversion (PHP, Apache Software Foundation, Django, ExtJS, Ruby a spousta dalších. (15), (19), (25)

2.5.3 Distribuovaný systém pro správu verzí

Distribuovaný verzovací systém je mladším systémem, který se snaží eliminovat některé nevýhody centralizovaného systému. Systém je obohacen o řadu nových funkcí, které umožňují efektivněji a hlavně volněji pracovat s repozitářem. Řadí se sem především Git, Mercurial, Bazaar a další.



Obrázek 3: Schéma fungování distribuovaného verzovacího systému

U distribuovaného RCS je možné vytvářet několik „centrálních“ repozitářů a je možné je jednoduše sdílet mezi uživateli. Každý uživatel má u sebe kopii celého stromu a může tak pracovat i bez připojení k serveru. Nic mu tedy nebrání v tom, procházet si např. starší úpravy. Dalším pozitivem distribuovaných RCS je možnost verzovat si data lokálně. Až když jsem spokojen s výsledkem odešlu data do hlavního repozitáře. Výhodou je i to, že když zkolabuje hlavní server, vývojáři mají lokálně kompletní historii. U centralizovaného RCS je toto velký problém a data se musí důkladněji zabezpečit. Na druhou stranu rozšířením funkcí se stal distribuovaný RCS složitějším pro uživatele. (15), (19), (25)

2.5.4 Principy programátora

KISS – *Keep It Simple Stupid* - „vždy jednej v souladu s kontextem a okolnostmi problému. Nekombinuj co není potřeba. Nedělej složitější to, co není. Navrhni vždy řešení, které je efektivní v souladu s okolními podmínkami.“ (24)

DRY – *Don't Repeat Yourself* - „pokud něco děláš podruhé a používáš CTRL+C a

CTRL+V je něco špatně! Neduplikuj kód, pokud jde něco vytknout před závorku, udělej to! Využívej principy OOP.“ (24)

YAGNI – *You Ain't Gonna Need It* - „nestav nic co, není potřeba. Dívej se do budoucnosti v rozumném horizontu. Zvaž jestli někdy využiješ přidanou funkci, zvaž jestli ti v budoucnu spíše nezkomplikuje život.“ (24)

CoC – *Convention over Configuration* - „drž se konvencí, usnadňuje to orientaci v kódu a snižuje nutnost znát dokonale pozadí kódu. Programátor nemusí dělat zbytečná rozhodnutí, pokud se kód chová standardně podle očekávání. Není nutné se učit všechny konfigurační direktivy a postupy jak zajistit požadované chování.“ (24)

2.5.5 Coding standards

Jedná se o sadu pravidel a postupů využívaných při psaní programového kódu. Pravidla jsou důležitá především pro samotné vývojáře, kterým napomáhají v orientaci a porozumění zdrojovým kódům a tím pádem dochází i k eliminaci možných chyb.

```
if (hours < 24 && minutes < 60 && seconds < 60) {
    return true;
} else {
    return false;
}
```

Obrázek 4: Srozumitelně napsaný zdrojový kód

Nyní následuje úplně stejný kód, pouze je nepřehledně napsaný.

```
if(hours<
24 && minutes<
60 && seconds<
60)
{return true
;} else
{return false
;}
```

Obrázek 5: Nepřehledně napsaný zdrojový kód

Do těchto pravidel patří také způsoby pojmenování funkcí, proměnných, databázových tabulek, atributů, využívání malých/velkých písmen, podtržitek, odsazování vnořených bloků a další.

Např. při využívání tzv. velbloudí notace (camel caps), jsou první písmena malé a další slova jsou oddělena velkými písmeny – *getAllResult()* apod. (17)

2.6 Ekonomické a manažerské metody

Část této práce je věnována stručné charakteristice použitých manažerských metod.

2.6.1 SMART

SMART je souhrn pravidel, která pomáhají především v rámci projektového managementu (Project Management) efektivně definovat rámec či cíl projektu a navrhovaného řešení. Pravidla SMART lze ale uplatnit i v jiných oblastech, kde je třeba nějakým efektivním způsobem definovat cíle.

S: specifický (specific)

Navrhované řešení nebo příležitost by měly být přesně popsány. Pokud jsme schopni jednoznačně odpovědět na otázku co je přesně a konkrétně předmětný problém a jak jej hodláme vyřešit, pak jsme problém popsali podle tohoto pravidla.

M: měřitelný (measurable)

Navrhované řešení by mělo být měřitelné. Každý projektový plán by měl obsahovat i kontrolu úspěšnosti našeho řešení, která musí být definována už na začátku.

A: přijatelné (attainable)

Řešení musí odpovídat potřebám svého příjemce. Nemělo by být nedosažitelné, ale na druhou stranu ani triviální.

R: reálné (realistic)

Řešení musí být realistické. Při úvaze o tomto pravidle bychom si měli položit otázku, zdali je možné navrhované řešení vůbec realizovat a dosáhnout požadovaných výsledků.

T: časový rámec (timed)

Další otázkou, kterou bychom si měli položit při vymezení navrhovaného řešení je časový rámec pro uvedení řešení v praxi. Tato definice navrhovaného řešení poskytuje odpověď na otázku, kdy bude současný problém řešen. (18)

2.6.2 Ganttův diagram

„Ganttův diagram (Gantt chart) lze vhodně využít k zobrazení časové náročnosti a posloupnosti jednotlivých částí projektu. Pro řízení a kontrolu projektu je potřeba přiměřeně detailní a zároveň realistické plánování. Kromě návaznosti jednotlivých dílčích částí projektu sledujeme i míru plnění těchto úkolů a celkovou časovou náročnost. Ganttův diagram zde slouží jako vizuální přehled o průběhu sledovaného procesu.“ (22)

2.6.3 SWOT analýza

SWOT analýza hodnotí silné (Strengths), slabé (Weaknesses) stránky společnosti, hrozby (Threats) a příležitosti (Opportunities) spojené s podnikatelským záměrem, projektem, strategií nebo i restrukturalizací procesů.

Díky ní dokážeme komplexně vyhodnotit fungování firmy, projektu, nalézt problémy nebo nové možnosti růstu. SWOT je součástí strategického (dlouhodobého) plánování společnosti.

Analýza spočívá v rozboru a hodnocení současného stavu firmy (vnitřní prostředí) a současné situace okolí firmy (vnější prostředí). Ve vnitřním prostředí hledá a klasifikuje silné a slabé stránky firmy. Ve vnějším prostředí hledá a klasifikuje příležitosti a hrozby pro firmu.

V rámci SWOT analýzy je vhodné hledat vzájemné synergie mezi silnými a slabými stránkami, příležitostmi a silnými stránkami apod. Tyto synergie pak vzápětí mohou být použity pro stanovení strategie a rozvoje firmy.

SWOT analýzu je možné využít jako silný nástroj pro stanovení a optimalizaci strategie společnosti, projektu nebo zlepšování stávajícího stavu či procesů. (27)

3 Analýza problému a současné situace

V této kapitole se zaměřuji na představení firmy, zákazníků a jejich produktů. Celá práce je zpracovávána ve spolupráci se společností XYZ, s.r.o., se kterou již přibližně dva roky úspěšně spolupracuji. Tato společnost mimo jiné vyvíjí a distribuuje B2B, B2C aplikace pro potřeby jednotlivých zákazníků. Za dobu působení na českém trhu se jí podařilo prodat desítky těchto aplikací. Klienti prostřednictvím těchto aplikací nejčastěji poskytovali prodej elektroniky, počítačových komponent, oblečení, dětských potřeb, potravin a zdravotních doplňků, cyklistických potřeb a dalších. Většina z nich produkty nabízí koncovému zákazníkovi, ale někteří přišli s požadavkem nabízet své zboží jak pro koncové zákazníky, tak i dalším obchodníkům prostřednictvím jedné aplikace.

Pro tyto účely společnost vytvořila aplikaci, kterou do dnešní doby s různými úpravami pro jednotlivé klienty nabízí. Samotnou aplikaci nabízí i k distribuci jiným prodejčům, kteří z následného prodeje získávají provize. Tato aplikace vznikla již před více jak dvěma lety a při jejím vývoji nebyly z dnešního pohledu zvoleny správné technologie. Aplikace byla postavena na staré verzi PHP, na šablonovacím systému, který se přestal o rok později vyvíjet a tak velmi těžkopádně splňuje rostoucí požadavky zákazníků. V té době se bohužel chybně vůbec neuvažovalo např. o masovém nasazení AJAXových funkcí apod. Když k tomu připočteme způsob, jakým byla tato aplikace rozšiřována a udržována (nebyl v tom žádný systém, jednotlivé verze se míchaly atd.), dospějeme k závěru, že nejefektivnějším řešením bude pravděpodobně její kompletní přepracování.

V následujících odstavcích se pokusím rozebrat a analyzovat současný stav této aplikace, zda je možné s ní dále pracovat a rozšiřovat ji, nebo bude lepší ji přepracovat. Na základě této analýzy pak navrhnou jednotlivé kroky vedoucí k postavení konkurenceschopné a především efektivně modifikovatelné aplikace, která opět zahájí svůj životní cyklus a firmě bude generovat zisk. Tyto závěry se pokusím podložit i finančním vyjádřením.

3.1 Představení společnosti

Společnost XYZ, s.r.o. vznikla v roce 2007 jako společnost zabývající se tvorbou

internetových stránek, aplikací, software na míru a outsourcingu IT. Sídlo společnosti se nachází v Praze. Z původních tří zakládajících členů se společnost rozrostla na osm lidí, které nyní zaměstnává, případně s nimi úzce spolupracuje. Jedná se o kolektiv mladých a ambiciózních lidí, kteří se snaží prosadit na poli IT a internetových aplikací.

Společnost si zakládá především na důkladném pochopení potřeb zákazníka. Na každého zákazníka je nasazen jeho osobní project manager, který je v úzkém kontaktu s programátory a grafiky. Díky této vazbě probíhá celý proces od získání požadavku po jeho konečnou realizaci velmi rychle a především v souladu se zákaznickými požadavky.

3.2 Reference a zákazníci

Během svého působení společnost vytvořila již desítky internetových stránek, aplikací a e-shopů. Mezi klienty společností patří například Palace Cinemas Czech s.r.o., či PYRAMIDA Průhonice s.r.o. a další. Subdodavatelsky společnost spolupracuje s grafickými studii Kafka Design, s.r.o., queue, s.r.o. a dalšími.

V poslední době se společnost zaměřuje také na vývoj jednodušších MRP systémů provozovaných v prostředí internetového prohlížeče, které nahrazují v menších firmách složité a finančně nákladné systémy.

Z globálního pohledu se společnost zaměřuje spíše na drobnější živnostníky a menší firmy, kterým pak nabízí komplexní služby a dlouhodobou spoluprací úspěšně podporuje jejich podnikatelskou činnost.

3.3 Nabízené služby

- ✓ grafika a design – tvorba profesionálních logotypů, budování corporate identity, tvorba profesionálních bannerů různých velikostí jak pro digitální média, tak i pro běžné použití, tvorba různých tiskovin, profesionální fotografování zboží a další
- ✓ webdesign a tvorba internetových prezentací a jejich hostování, CMS
- ✓ internetový marketing, zvyšování návštěvnosti internetových prezentací, SEO, SEM, PPC kampaně

- ✓ tvorba, prodej a údržba internetových B2B, B2C aplikací
- ✓ analýza, návrh a tvorba software na míru
- ✓ počítačový servis

Stěžejní činností společnosti je tvorba, prodej a údržba internetových B2B, B2C aplikací a analýza, návrh a tvorba software na míru spolu s marketingem.

3.4 SWOT analýza společnosti

Silné stránky

- ✓ mladý a perspektivní pracovní kolektiv
- ✓ časová flexibilita a maximální důraz na potřeby zákazníka
- ✓ obrovská chuť se rozvíjet
- ✓ osobní kontakt se zákazníkem
- ✓ výhodná poloha sídla společnosti
- ✓ nízké provozní náklady
- ✓ široké zázemí a komplexní servis

Slabé stránky

- ✓ málo rozvinutý projektový management
- ✓ roztržitost a nejednotnost zdrojových kódů většiny aplikací
- ✓ nedostatek kvalifikovaného personálu
- ✓ omezená schopnost konkurovat velkým firemním řešením
- ✓ nižší stupeň využívání moderních dostupných technologií

Příležitosti

- ✓ zisk a podpora větších obchodních partnerů
- ✓ zahraniční zákazníci, státní správa

- ✓ zvýšit kvalitu internetového marketingu
- ✓ zavedení projektového managementu a modelů odpovědnosti

Hrozby

- ✓ snižující se platební morálka některých zákazníků
- ✓ ztráta klíčových pracovníků
- ✓ globální útlum ekonomiky a s ním související poptávka po produktech a službách
- ✓ nestále rostoucí konkurence

Společnost těží především z toho, že sídlí v hlavním městě České republiky, kde je koncentrována velká skupina potencionálních i stávajících zákazníků. Je zde ovšem daleko silnější konkurence a tak je třeba své produkty a služby neustále zdokonalovat. Tím, že je společnost schopna zajistit komplexní servis ať již prostřednictvím vlastních sil, nebo prostřednictvím obchodních partnerů má mírnou výhodu.

S růstem společnosti a množstvím zákazníků se projeví i některé slabé stránky. Asi největším problémem je projektový management. Rozsáhlejší a dlouho trvající projekty jsou vedeny s nízkou efektivitou a ve finále nepřináší tak vysoký zisk, jaký by měly.

3.5 Analýza současné internetové B2B, B2C aplikace

Jak jsem již zmiňoval v kapitole nabízené služby, je vývoj a distribuce internetových B2B a B2C aplikací pro společnost stěžejní činností. Této činnosti bylo v počátku věnováno velké úsilí, aby vznikla do dnešní doby stále využívaná aplikace. Na vývoji této aplikace jsem se také nemalým dílem podílel.

Samotná aplikace vznikala v počátcích zahájení činnosti společnosti a již v té době byla vyvíjena jako hlavní předmět podnikatelské činnosti. V průběhu následujících dvou let procházela aplikace neustálým vývojem a úpravami. Jednalo se především o úpravy, které vyžadovalo více zákazníků a bylo pravděpodobné, že další zákazníci budou mít stejný požadavek.

V současné chvíli můžeme v nabídce společnosti XYZ, s.r.o. nalézt tři varianty B2B,

B2C aplikace. Všechny varianty vycházejí z jedné verze internetové B2B, B2C aplikace, jen jsou v nich různě kombinovány jednotlivé moduly a rozšíření. Každá z nich se dá ve finální fázi v podstatě jakkoliv modifikovat, ale na základě podobných preferencí zákazníků vznikly 3 hlavní varianty.

Jedná se o tyto 3 varianty:



Obrázek 6: Grafická podoba krabicových verzí stávajícího řešení aplikace

3.5.1 SmartShop

Velmi dobře vybavený e-shop s důrazem na maximální ovladatelnost, na minimální provozní náklady a na automatizaci administrativních úkonů. Je určen klientům, kteří chtějí pohodlně podnikat bez omezení. Je velmi vhodný jako rozšíření kamenného obchodu.

- ✓ e-shop s neomezeným počtem položek
- ✓ automatický fakturační systém
- ✓ kvalitní redakční systém
- ✓ správa reklamní kampaně
- ✓ statistiky návštěvnosti

- ✓ tvorba e-shopu na splátky
- ✓ základní SEO optimalizace

Cena nejnižší SmartShop verze začíná na 14 900 Kč bez DPH. Tato verze je připravena a se dá v podstatě ihned začít využívat.

3.5.2 ProfiShop

Velmi dobře vybavený e-shop s integrovaným skladovým systémem. Byl vytvořen s důrazem na maximální ovladatelnost, na minimální provozní náklady a na automatizaci administrativních úkonů. ProfiShop je určen klientům, kteří chtějí pohodlně podnikat bez omezení a kteří nemají svůj kamenný obchod. Obsahuje integrovaný skladový systém a kompletní správu dokladů pro daňovou evidenci či účetnictví.

- ✓ e-shop s neomezeným počtem položek
- ✓ integrovaný skladový systém
- ✓ podrobné ekonomické statistiky
- ✓ automatický fakturační systém
- ✓ kvalitní redakční systém
- ✓ správa reklamní kampaně
- ✓ statistiky návštěvnosti
- ✓ základní SEO optimalizace

Cena této varianty se je na hranici 24 900 Kč bez DPH. I tato varianta je připravena a je možné ji v podstatě okamžitě nasadit.

3.5.3 IndividualShop

Vlastnosti a funkce této varianty jsou velmi individuální a v podstatě se odvíjí od požadavků každého klienta. Každý klient má možnost si zvolit požadované moduly a funkce ať již z existujících, nebo mu jsou doprogramovány, či přizpůsobeny přesně podle jeho požadavků. Cena nejnižší varianty IndividualShopu začíná na 17 890 Kč.

U této varianty se nám dokonce stalo i to, že jsme museli přeprogramovat přes polovinu zdrojových kódů, abychom splnili velmi individuální požadavky klienta. Právě u klientů vyžadujících tuto verzi jsou nejvíce znát slabiny stávajícího systému a použitých technologií při jejím vývoji.

3.5.4 Technologie současné aplikace

Aplikace vznikala před více jak dvěma a lety a tenkrát bylo rozhodnuto, že se bude vyvíjet pro PHP ve verzi 4 a vyšší. Bylo to především z důvodu, že společnost v té době ještě nedisponovala serverem pro hostování aplikací na vlastní platformě a někteří ze zákazníků hostovali aplikace na svých serverech, nebo na serverech hostingových společností, kde bylo PHP pouze čtyřkové verze. Dalším důvodem bylo to, že ne všichni vývojáři uměli plnohodnotně vyšší verzi PHP využívat a spousta tříd a skriptů, které firma v tu dobu využívala, byly programovány taktéž pro PHP čtyřkové verze. A tyto skripty a dokonce i části dřívějších aplikací byly využity při vývoji aplikace.

Jako databázové úložiště je využíván systém MySQL verze 3.2 a vyšší a v počátcích nebyly využívány žádné rozšířené funkce tohoto databázového systému. Formát úložiště dat byl typu MyISAM. V průběhu životnosti aplikace došlo k zakomponování nových knihoven pro práci s databází (dibi) a tyto funkce (transakce, relační integrita atd.) se začaly využívat. Zároveň postupně docházelo k přechodu na nový formát úložiště (InnoDB), který tyto nové funkce podporoval. Ale i tak nejsou zakomponovány v celém systému.

Pro JavaScriptovou obsluhu byly programovány vlastní funkce a stejně jako u PHP nebylo využito žádného frameworku. I zde byl v průběhu životnosti aplikace JavaScriptový framework nasazen (jQuery). Co se týká AJAXu, tak jeho využívání nebylo při vývoji vůbec uvažováno. AJAXové prvky byly zařazeny až na základě požadavků zákazníků a to za pomoci JSON struktury a jQuery. Na masové využívání AJAXu ale bohužel aplikace nebyla stavěna.

Informační technologie se v tomto směru opět výrazně posunuly a PHP 4 je již z dnešního pohledu historie. Dnešní aplikace se vyvíjejí s podporou frameworků, které se starají o běžné události, obstarávají stále se opakující funkce, napomáhají hlídat

bezpečnost a velmi usnadňují samotný vývoj, udržovatelnost a rozšiřitelnost aplikace.

Největším problémem současné aplikace je její současný „rozházený“ stav. Ve firmě nalezneme desítky různých verzí na různých FTP apod. a nikdo není schopen říci, kde je nejaktuálnější verze, případně co která verze obsahuje. Tento stav je způsoben tím, že pro vývoj a údržbu aplikace nebyl využit žádný systém pro správu verzí. Dohledat tak některé funkce, případně chyby je téměř nadlidský výkon a tak u některých zákazníků se opakují problémy, případně se programují části, které již byly jednou vyřešeny.

Druhým zásadním problémem je to, že každá část aplikace je psána jinak, neexistuje žádná dokumentace apod. To vzniklo tím, že se nikdo nesjednotil jednotlivé styly práce jednotlivých programátorů. Kdyby existovaly nějaké firemní coding standards, výsledný kód by mohl být daleko přehlednější a snáze udržovatelnější.

Tyto problémy se v podstatě netýkají samotného zákazníka, ale velkou měrou snižují finální zisk za prodanou aplikaci. Zákazníci byli s funkčností a možnostmi aplikace vždy velmi spokojeni a naplňovala jejich požadavky. Pokud by firma neplánovala rozšířit okruh zákazníků o další odvětví, stávající aplikace by stále mohla vyhovovat. Samozřejmě zvyšující požadavky zákazníků by musely být stále zapracovávány a složitost těchto funkcí by nadále snižoval zisk a dále zneprůhledňoval aplikaci. Pro společnost je ale důležité nabízet zákazníkovi stále co nejvyšší úroveň služeb, proto je tento problém velmi aktuální a je třeba se mu postavit.

Nedostatky současné aplikace spočívají především v zastaralé technologii a jí přizpůsobenému návrhu. Při návrhu aplikace se neuvažovalo s možností provozovat aplikaci v několika jazycích, nebo s nasazením AJAXových událostí. Na druhou stranu aplikace v době kdy vznikla plně vyhovovala potřebám tehdejších zákazníků. Ovšem technologický vývoj a s ním spojené měnící se potřeby zákazníků jsou nezadržitelné.

Na aplikaci se podepsala také absence jakéhokoliv projektového managementu, který by řídil další vývoj a definoval kroky, jakými se s aplikací bude dále pracovat. Do aplikace se začalo postupně přidávat velké množství funkcí, které kolikrát nebyly ani potřeba a každý programátor je realizoval svým stylem. Tím se z celé aplikace stal velmi nepřehledný kus kódu, který již nelze efektivně a s nízkými náklady spravovat.

V průběhu životního cyklu aplikace nebylo využito žádného sofistikovaného systému pro správu verzí, který by alespoň částečně udržoval aplikaci a hlídal jednotlivé úpravy a verze.

3.6 SWOT analýza stávající B2B, B2C aplikace

Silné stránky

- ✓ existující, zákazníkem ověřená a okamžitě použitelná aplikace
- ✓ vysoká míra optimalizace pro vyhledávače
- ✓ dobře navržené uživatelské rozhraní aplikace
- ✓ nízká cena

Slabé stránky

- ✓ staré technologie, nízká flexibilita
- ✓ nižší zabezpečení aplikace
- ✓ prolínání různých programátorských stylů, prolínání technologií
- ✓ nákladná údržba a rozšiřitelnost, neexistující správa verzí
- ✓ nemožnost multijazyčné verze bez zásadních změn v aplikaci
- ✓ možnost nasadit aplikaci jen jako B2B, nebo B2C samostatně

Příležitosti

- ✓ klient opouští kamenný obchod a prodává pouze prostřednictvím internetu, kde soustřeďuje veškeré síly
- ✓ zvyšování obrátu a konkurenceschopnosti na základě připraveného řešení
- ✓ multijazyčná aplikace získá i zahraniční podnikatelské subjekty pro které je stávající řešení nevyhovující
- ✓ dodávka obchodu na míru, minimální odchylky od požadavků zákazníka
- ✓ vyšší kvalita nabízeného produktu

Hrozby

- ✓ ekonomická krize a změna požadavků zákazníka
- ✓ rostoucí konkurence internetových aplikací
- ✓ snaha realizovat systém v co nejkratším období
- ✓ špatné manažerské vedení projektu

Při návrhu nové aplikace je třeba se věnovat nejen samotnému programování, ale k projektu přistupovat komplexně. Je třeba velmi přesně navrhnout jednotlivé funkce, definovat možnosti jednotlivých modulů, promyslet způsob vývoje, označování verzí a správu jednotlivých verzí, navrhnout standardní postupy pro práci se zdrojovými kódy aplikace - coding standards apod., navrhnout datový model s tím, že je třeba od základu počítat s neomezeným množstvím jazykových mutací veřejné části aplikace. Administrační rozhraní postačí v českém a anglickém jazyce. Pokud budou tyto činnosti provedeny s náležitou péčí, je možné eliminovat všechny slabé stránky a využít některých příležitostí.

3.7 Cílová skupina a účel aplikace

Před samotným návrhem aplikace je třeba najít odpovědi na následující otázky. Jedině tak budeme schopni navrhnout a vytvořit aplikaci, která bude schopna uspokojit jak potřeby zákazníka, tak potřeby společnosti XYZ, s.r.o.

3.7.1 Pro koho je aplikace určena

Před návrhem nové aplikace je třeba znát odpověď na několik základních otázek, bez kterých nelze správně nastavit mantinely vývoje nové aplikace. První otázka zní:

„Kdo je/bude našim zákazníkem a komu primárně bude naše B2B, B2C aplikace nabízena?“

Cílem společnosti XYZ, s.r.o. není soupeření s aplikacemi od velkých IT společností, které se soustředí především na větší zákazníky. Aplikace těchto společností jsou většinou vyvíjeny několik let, na jiných platformách a na jejich rozvoj a údržbu plyne

daleko větší objem finančních prostředků. Stejně tak jako tým, který se stará o vývoj a správu těchto aplikací bude podstatně větší. Naše aplikace je nabízena především menším podnikatelským subjektům a drobným živnostníkům. Tyto podnikatelské jednotky ve většině případů od aplikace vyžadují základní funkcionalitu, přehledné a intuitivní uživatelské rozhraní a kvalitní zákaznickou podporu. Tito zákazníci nechtějí funkcemi nabitou aplikaci, která je neúměrně finančně nákladná a přes 80% jejich vlastností pro své účely nikdy nevyužijí a spíše jim znesnadňují samotné ovládání aplikace. Tyto funkce se obvykle hodí pro marketingové účely a pro propagaci aplikace.

Na druhé straně jsou pak velmi jednoduché aplikace, které mají funkcionalitu nedostatečnou. Tyto aplikace jsou většinou nabízeny velmi levně, případně zdarma, ale jejich správa a údržba je velmi neefektivní. Navrhovaná B2B, B2C aplikace by měla zaujmout přibližně středovou pozici mezi těmito extrémy.

3.7.2 Účel B2B, B2C aplikace

„K jakému účelu má primárně tato aplikace sloužit? S čím si má poradit?“

Aplikace by měla být schopná si poradit s prodejem různého druhu zboží. Nejprodávanějším druhem prodáváného sortimentu prostřednictvím stávající aplikace u zákazníků společnosti XYZ, s.r.o. je oblečení, elektronika, hardware a software, sportovní potřeby, dětské potřeby a hračky, potraviny a doplňky, zdravotnické potřeby a další.

Každé z uvedených druhů zboží má své specifické atributy a je potřeba, aby s nimi aplikace dokázala pracovat a zákazníkovi nabídla vždy to, co vyžaduje.

4 Vlastní návrhy řešení

Vlastní návrh B2B a B2C aplikace se dělí do několika základních částí. Jedná se o návrh funkcí veřejného rozhraní aplikace, návrh funkcí administračního rozhraní aplikace, volba technologií potřebných k efektivní implementaci, návrh datového skladu aplikace a sestavení doporučených postupů při implementaci, údržbě a rozšiřování aplikace.

Vlastní návrhy vycházejí především z dosavadních zkušeností a znalostí, poznatků a připomínek od stávajících klientů společnosti XYZ, s.r.o. a samozřejmě i z ověřených a vyzkoušených částí stávající aplikace. To vše bude obohaceno o nové funkce a současné trendy ve vývoji těchto aplikací tak, aby byla do budoucna zajištěna konkurenceschopná aplikace, která bude firmě schopna generovat zisk a budovat loajalitu zákazníků.

Internetová aplikace musí tedy poskytovat dostatečné množství kvalitních informací nejen návštěvníkům, zákazníkům, ale i vyhledávačům. Kvalita poskytovaných informací tak bude ve velké míře závislá na kvalitě šablony a možnostech grafického návrhu, ale také na množství a kvalitě zadaných údajů do databáze přes administrační rozhraní.

4.1 Specifikace veřejného rozhraní nové B2B, B2C aplikace

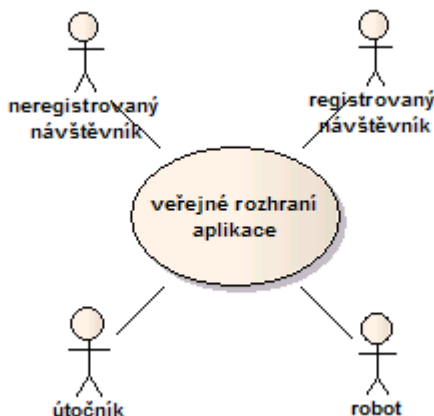
Veřejné rozhraní je část aplikace, která se zobrazí návštěvníkovi při navštívení internetových stránek, respektive internetové domény provozovatele, ale také určitá část po jeho přihlášení pod svým účtem. Po úspěšné přihlášení uživatele může dojít k individualizaci obsahu, prodejních cen, prodejního množství apod. Individualizaci obsahu bude možno ovlivnit prostřednictvím administračního rozhraní a zákaznických skupin.

Veřejné rozhraní aplikace bude kompletně multijazyčné. Přeloženy budou jak jednotlivé formuláře, texty, ale i url adresy. Jednotlivé jazykové mutace bude možno v průběhu provozu internetové aplikace přidávat či ubírat.

4.1.1 Typy uživatelů veřejné části aplikace

Jelikož se jedná o internetovou aplikaci, bude k ní mít neustále přístup několik typů

uživatelů. Ne každý typ uživatele je žádoucí, proto je důležité, aby aplikace typ uživatele včas rozpoznala a podle toho se i zachovala.



Obrázek 7: Typy uživatelů veřejné části aplikace

Neregistrovaný návštěvník je typem návštěvníka, pro kterého je aplikace určena. Většina následujících návrhů aplikace je cílených právě na něj a na jeho vyšší typ, tedy registrovaného a přihlášeného zákazníka.

Registrovaný zákazník vychází ze zákazníka neregistrovaného, jen bude mít navíc řadu funkcí a možností. Od správy vlastního účtu až po přehled jednotlivých objednávek, které uskutečnil. Pro tohoto návštěvníka bude také možno individualizovat obsah.

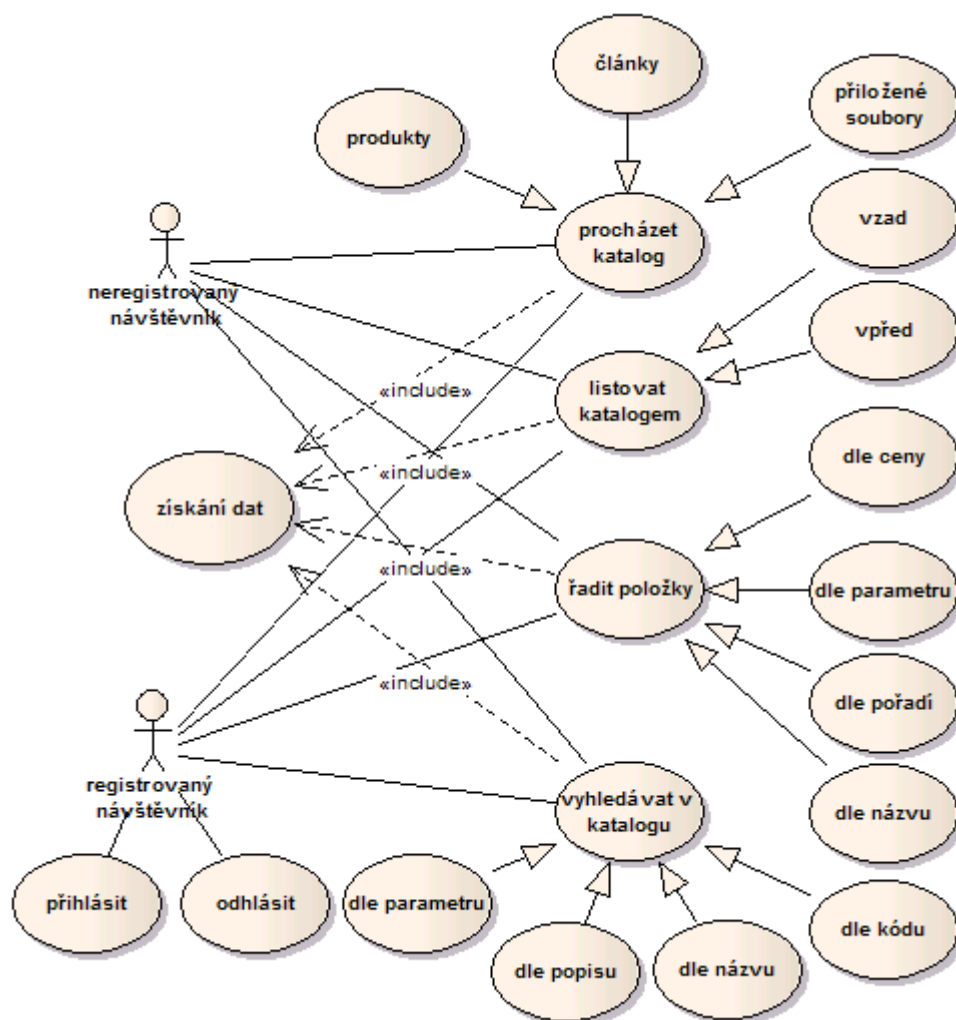
Pro aplikaci nejrizikovější skupinou uživatelů je typ útočník. Je důležité, aby měl tento typ uživatele co nejméně možností aplikaci napadnout a v případě podezřelého chování došlo k zaznamenávání jeho činností a okamžitému upozornění správce aplikace.

Za robota je považován software, který sám prochází internetem a ukládá (indexuje) si obsah jednotlivých stránek do databází. Obsah pak nabízí prostřednictvím vyhledávače. Tomuto typu uživatele bude aplikace poskytovat optimalizovaný obsah, aby měl svoji práci co nejjednodušší a dokázal si uložit veškerý požadovaný obsah.

4.1.2 Katalog

Katalog bude stěžejní částí celé B2B, B2C aplikace. Do katalogu budou moci být zařazeny jednotlivé kategorie, jejich podkategorie, články, ale i jednotlivé stránky.

Kategorie budou tvořit stromovou strukturu s teoreticky neomezenou úrovní podkategorií. Do kategorií a podkategorií se budou vkládat produkty, které budou moci následně jednotlivý návštěvníci nakupovat. Celý katalog si bude moci nadefinovat správce či provozovatel aplikace.



Obrázek 8: Use Case diagram katalogu veřejné části aplikace

Po kliknutí na některou část katalogu se vypíše seznam položek, které jsou k dané části přiřazeny nebo jsou přiřazeny do některé z podkategorií. Na tento výpis bude možno aplikovat filtry, řadit ho dle různých parametrů a v případě většího množství položek dojde k jeho rozdělení na několik stránek, mezi kterými bude možno přecházet. Pokud se bude jednat o kategorie s produkty, budou jednotlivé položky obsahovat základní

informace jako je cena, název, obrázek, základní popis, parametry apod. Tyto informace již budou závislé čistě na klientovi a jeho potřebách. Po kliknutí na položku dojde k zobrazení detailní stránky položky.

Veškeré operace typu řazení, listování a vyhledávání v katalogu budou realizovány prostřednictvím AJAXu.

4.1.3 Detail produktu

Stránka s detailem produktu bude poskytovat podrobné informace o produktu a jeho variantách v přehledné formě dle individuálního grafického podkladu internetové aplikace provozovatele. Účelem této stránky je v přehledné formě poskytnout potenciálnímu zákazníkovi co nejpřesnější informace o dané položce.

- ✓ cena a další poplatky - autorský, recyklační poplatek apod.
- ✓ volitelné parametry/varianty/atributy položky - barva, velikost, hmotnost apod.
- ✓ popis položky
- ✓ skladová dostupnost, rychlost dodání
- ✓ fotografie, manuál, recenze, videa apod.
- ✓ možnost vložit položku do košíku, možnost volby množství
- ✓ související produkty

Každá položka bude moci obsahovat obrázky k jednotlivým variantám a návštěvník si mezi těmito obrázky bude moci pomocí AJAXových funkcí listovat. Před vložením do košíku si bude moci navolit dostupnou variantu a množství. Přes administrační rozhraní bude možné jednotlivým skupinám zákazníků individualizovat detail produktu, jeho prodejní ceny, slevy, prodávané množství. Samotná operace vložení do košíku bude probíhat opět na pozadí pomocí AJAXu.

4.1.4 Akční nabídky, novinky

Akční nabídky, novinky, položky označené jako top produkt nebo zajímavá cena bude

možno při výpisu odlišit od klasických položek. O odlišení od běžných položek se bude starat především šablonovací systém s podporou kaskádových stylů a grafiky. Všechny takto označené, nebo do těchto skupin začleněné položky bude možno zobrazovat na vstupní stránce internetového obchodu, případně jinak dle specifických potřeb klienta. Zobrazované novinky, akční nabídky a další budou definována v administrační části.

4.1.5 Vyhledávání

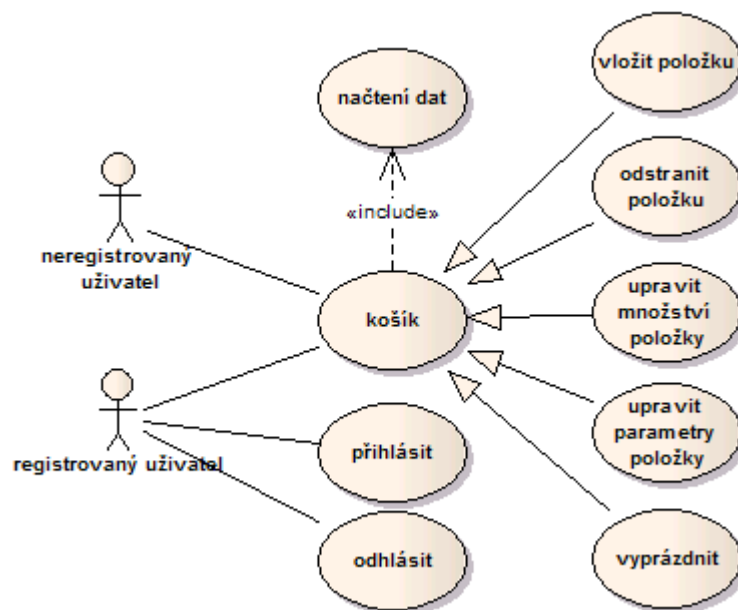
Z internetových průzkumů vyplývá, jak důležité je potencionálnímu zákazníkovi podstrčit správné informace. Pokud návštěvník během krátké chvíle nenalezne informace které hledá, velmi často odejde ke konkurenci. A pokud budeme chtít získat takto ztraceného zákazníka nazpět, bude třeba vynaložit mnohem více úsilí a finančních prostředků.

V aplikaci bude možnost využít základního vyhledávání dle názvu, nebo kódu produktu a pak rozšířeného fulltextového vyhledávání, pomocí kterého se bude prohledávat kompletní obsah internetové aplikace. V rozšířeném vyhledávání bude možno zadávat i další parametry jako je výrobce, předem definované parametry jako např. barva, velikost atd., které budou závislé na druhu prodáváného sortimentu.

Nalezené výsledky se budou následně v přehledné formě vypisovat a bude možno v nich listovat, či si je dle parametrů seřadit.

4.1.6 Nákupní košík

Košík má v internetové aplikaci v podstatě stejný význam jako ten, který používáme při klasickém nákupu v kamenném obchodě. Tento elektronický bude mít navíc několik funkcí, které usnadní nákupní proces zákazníka. Jedná se především o informační funkci, která bude zákazníka v každém okamžiku informovat o celkové sumě položek v něm umístěných.



Obrázek 9: Use Case diagram nákupního košíku

Zákazník si prostřednictvím košíku bude moci taktéž měnit variantu a kupované množství požadovaného produktu. V případě, že se rozhodne, že dané položky nenakoupí, bude moci košík kompletně vyprázdnit, nebo bude mít možnost odstranit položky jednotlivě. Na základě položek v košíku pak bude možné vystavit objednávku, čímž dojde k zakoupení všech produktů v košíku.

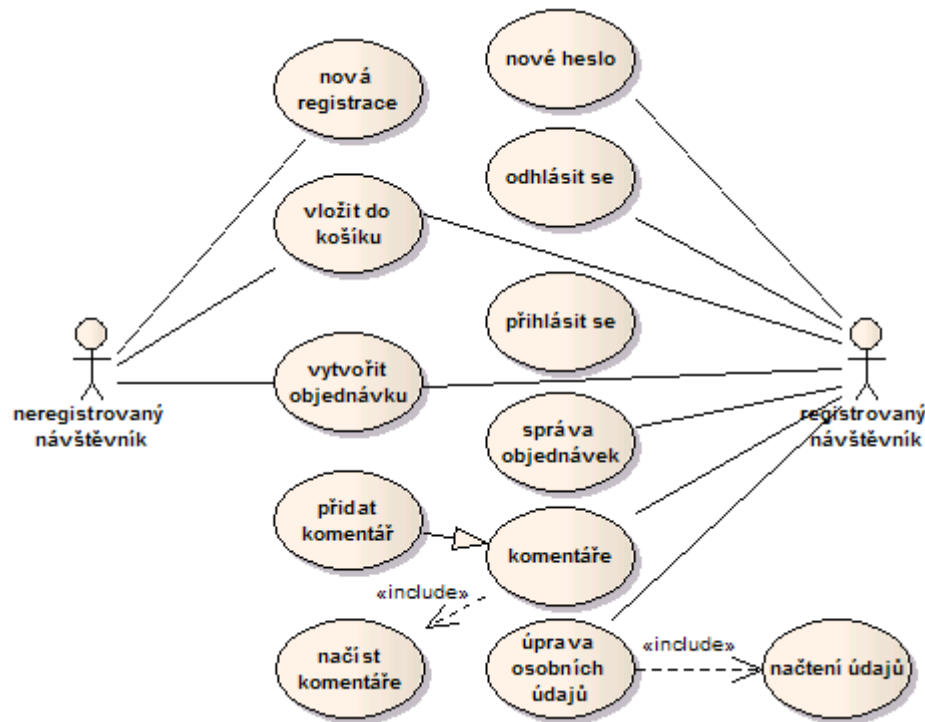
4.1.7 Objednávka

Jakmile zákazník vloží do košíku požadované produkty, může přistoupit k vytvoření objednávky. Pokud se bude jednat o nepřihlášeného zákazníka, bude muset vyplnit fakturační a dodací údaje a v případě, že se bude jednat o přihlášeného zákazníka, budou tyto údaje předvyplněné s možností je změnit. Po výběru požadovaného způsobu platby a dodání bude moci zákazník odeslat objednávku. Objednávka se uloží do databáze interní systém vygeneruje potvrzující email s přehledem objednávky a zadaných údajů, který odešle na emailovou adresu zadanou zákazníkem. V případě, že se bude jednat o registrovaného zákazníka, bude moci po přihlášení sledovat průběh vyřizování objednávky, případně ji taktéž prostřednictvím grafického rozhraní aplikace stornovat.

4.1.8 Uživatelský účet

V internetové aplikaci budou moci nakupovat jak zaregistrovaní a přihlášení návštěvníci, tak i neregistrovaní. V případě, že bude aplikace provozována i jako B2B řešení, bude registrace a přihlašování zákazníků nutností.

Každému návštěvníkovi, který vstoupí na stránky internetové aplikace a nebude přihlášen z předchozího přístupu, bude automaticky přidělena role hosta - *guest*. Jakmile se zákazník přihlásí, jeho role se automaticky změní na *customer*. Provozovatel aplikace si pak bude moci vytvářet další skupiny, do kterých pak bude moci jednotlivé zákazníky zařazovat a individualizovat jim zobrazovaný obsah a nabízené produkty.



Obrázek 10: Use Case diagram uživatelských účtů

Po přihlášení zákazníka budou načteny jeho osobní údaje, které bude moci modifikovat, bude si moci ke svému účtu přidávat nové fakturační a doručovací adresy. Přihlášený návštěvník bude dále moci komentovat články a produkty napříč katalogem. Dále bude mít k dispozici grafické rozhraní s přehledem svých objednávek a stavů, ve kterém se objednávky nacházejí. Ke všem vyřízeným objednávkám si bude moci stáhnout

vygenerované PDF doklady – objednávku, fakturu, dodací a záruční list.

Každý návštěvník se bude moci prostřednictvím svého emailu přihlásit k odběru novinek a akčních nabídek, které mu budou v pravidelných intervalech zasílány.

4.2 Specifikace administračního rozhraní nové B2B, B2C aplikace

V následujících částech se budu věnovat návrhu jednotlivých modulů administračního rozhraní aplikace. Pokusím se přesně specifikovat požadované vlastnosti jednotlivých částí a také určitou představu finální funkcionality. Při návrhu aplikace je kladen důraz především na budoucí rozšiřitelnost, snadnou údržbu, ale i jednoduchost a přehlednost aplikace. Celá aplikace se bude skládat z modulů, které bude možno skládat jako skládanku přesně podle potřeb zákazníka. Volitelné moduly aplikace bude možno deaktivovat, tak aby zákazník dostal jen to, co požaduje. To je důležité hned z několika hledisek. Když zákazníkovi nabídneme i funkce, které nevyžadoval, dojde k několika nežádoucím efektům. Prvním z nich je snížení přehlednosti aplikace pro zákazníka. Druhým nežádoucím efektem je následná rozšiřitelnost, údržba a aktualizace aplikace. Při každém z těchto procesů se bude muset brát v potaz i ta rozšířená funkčnost, což přinese další komplikace a náklady navíc.

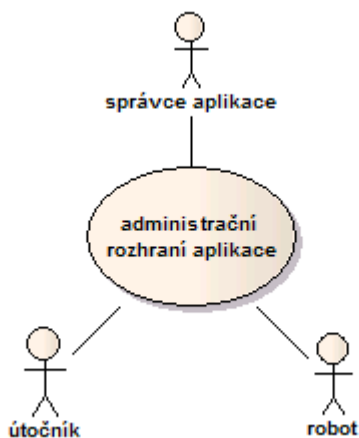
Velmi důležitým aspektem celé aplikace je kompletní multijazyčnost. Pro administrační rozhraní bude dostačující české a anglické rozhraní s tím, že primárním jazykem bude čeština. Pro uživatelské rozhraní jsou nutností další světové jazyky. Důležité je, aby aplikace nebyla pouze jednojazyčná, např. anglicky, nebo německy, ale aby si návštěvník mezi jazykovými variantami mohl zvolit. Samozřejmostí pak musí být automatická detekce podle prohlížeče návštěvníka. Právě této potřebě je třeba vhodně navrhnout administrační rozhraní aplikace.

Velký důraz bude kladen také na implementaci marketingových nástrojů, jako jsou optimalizace pro návštěvníky (SEO, SEM), sledování oblíbenosti produktů, nastavování slev a akcí k produktům, správa bannerů, sledování návštěvnosti a jejich analýzy a další.

4.2.1 Typy uživatelů administračního rozhraní aplikace

K administrační části aplikace bude mít přístup několik typů uživatelů. Je důležité, aby

byl typ uživatel včas rozpoznán a aplikace se k němu na základě toho zachovala. Administrační rozhraní je neveřejnou částí aplikace.



Obrázek 11: Typy uživatelů administrační části aplikace

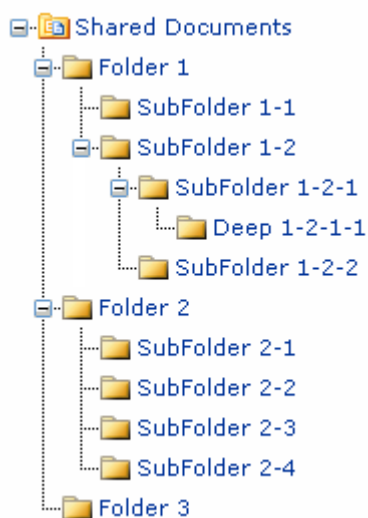
Správce aplikace je jediným žádoucím typem uživatele, který bude mít možnost aplikaci obsluhovat. Tento typ uživatele bude rozpoznán na základě autentizace a následně bude autorizován provádět jemu přidělené činnosti.

Robot je v administrační části nežádoucím typem uživatele. Pro systém neznamena riziko a jeho odmítnutí či přesměrování bude probíhat pouze dle vhodných příkazů např. nofollow apod.

Pro aplikaci nejrizikovější skupinou uživatelů je typ útočník. Je důležité, aby měl tento typ uživatele co nejméně možností aplikaci napadnout a v případě podezřelého chování došlo k zaznamenávání jeho činností a okamžitému upozornění správce aplikace.

4.2.2 Správa katalogů

Správa katalogů bude stěžejní částí nové aplikace. S tímto modulem bude v aplikaci spolupracovat, nebo s ním bude provázána většina dalších modulů. Prostřednictvím uživatelského rozhraní nad tímto modulem bude možno vytvářet libovolné množství stromových struktur různých katalogů, nebo navigačních menu. Samozřejmostí katalogu musí být i poskytování kvalitních dat pro SEO optimalizaci veřejné části aplikace.



Obrázek 12: Stromová
struktura katalogu

U každé kategorie půjde nastavit její typ, pomocí něž se určí druh obsahu, který půjde do kategorie zařadit. Typy kategorií budou následující:

- ✓ *produkty*
- ✓ *článek*
- ✓ *soubory*
- ✓ *články*

Produkty – do tohoto typu kategorie půjde následně přiřadit libovolný počet produktů, ale i dalších podkategorií různých typů, do kterých půjde opět vkládat libovolný obsah dle typu. Spolu s typem *článek* se bude jednat o nejvyužívanější typ kategorie u jednodušších typů prodávaných aplikací.

Soubory – kategorie tohoto typu nebude mít až tak široké využití. Využívat se bude především ke vkládání různých ceníků, propagačních materiálů apod.

Článek – do kategorií tohoto typu bude možno zařadit pouze jediný článek. Tato kategorie se pak bude chovat jako statická stránka.

Články – tento typ kategorie bude určen pro vkládání libovolného množství článků

a dalších podkategorií. Články budou po určitém množství, které bude konfigurovatelné, vždy stránkovány.

Kategorie v katalogu půjde upravovat, mazat, měnit jejich pořadí, bude existovat možnost zobrazení a skrytí pro veřejnost. Výchozími údaji pro jednotlivé kategorie v katalogu budou – název, url, popis, obrázek, klíčová slova. Možnost nahrávat obrázky i ke kategoriím bude využívána především u designově laděných aplikací. Důležitou vlastností bude skrývání kategorií. Tím, že dojde ke skrytí kategorie musí automaticky dojít ke skrytí veškerého obsahu, který pod danou kategorií spadá.

Veškeré úpravy, mazání, vytváření, řazení, skrývání kategorií bude implementováno za pomoci AJAXu, aby byla zajištěna uživatelská přívětivost.

4.2.3 Správa produktů

Modul správa produktů bude kompletně obsluhovat jednotlivé produkty. Správce aplikace bude mít možnost produkty vytvářet, upravovat, mazat, zobrazovat či skrývat veřejnosti, měnit jejich pořadí. Vzhledem k tomu, že každý produkt půjde zařadit do několika různých kategorií (typ *produkty*), je třeba, aby bylo možné měnit pořadí a zvýrazňovat produkty i napříč kategoriemi (v jedné kategorii budeme chtít produkt na prvním místě, v jiné např. na třetím místě apod.). Toho dosáhneme přidělováním tzv. prioritních bodů, v hodnotách od 0 do 999, kde nejvyšší prioritu bude mít produkt s nejvyšším bodovým číslem.

Jelikož je potřeba, aby byla aplikace použitelná u širšího spektra zákazníků, pro jednotlivé produkty bude v základní konfiguraci možno zadávat tyto údaje: EAN kód, kód, název, url adresa, description tag, klíčová slova, tagy, popis, detailní popis, výrobce, záruční lhůta, expediční lhůta, zobrazit od data, skrýt od data, stav skladu, prodejní množství, nákupní cena, původní cena, aktuální cena, sazba DPH, recyklační poplatek, autorský poplatek. Jednotlivé produkty bude možno řadit do skupin, jako jsou novinky, top produkt, speciální nabídka, zajímavá cena, výprodej, akce a ostatní. Takto označené produkty pak půjde ve veřejné části graficky různě odlišovat a dále s nimi pracovat dle požadavků zákazníka.

Dalšími důležitými vlastnostmi u produktů bude možnost definování atributů/parametrů

a následného přiřazení těchto atributů/parametrů k produktům. Vytvořený atribut/parametr bude svázán s kategorií – jelikož je předpoklad, že v kategorii budou produkty se shodnou strukturou atributů/parametrů. Samozřejmě každý atribut/parametr bude možno přiřadit k libovolnému množství kategorií. Tím, že se produkt začlení do kategorie, dojde k načtení přidělených atributů/parametrů a možnosti jejich doplnění.

K produktu a jeho atributům/parametrům pak bude možné nahrávat obrázky. Tímto způsobem pak bude možno dosáhnout toho, že si provozovatel internetové aplikace bude moci vytvořit například barevné varianty produktů a ke každé variantě nahrát vlastní fotografii - viz další část „Správa atributů/parametrů“. Rozměry obrázků budou vždy pevně definovány při instalaci aplikace a budou vycházet z individuálního grafického návrhu pro klienta. Aplikace bude všechny nahrávané obrázky automaticky upravovat na definované rozměry s tím, že bude zachovávat jejich výchozí poměr stran, aby nedošlo k deformaci. V případě, že bude nahrávaná fotografie menších rozměrů než definované, nebo bude v jiném poměru, bude obrázek vyplněn výchozí barvou. Do každého obrázku bude možno vložit ochranný vodoznak.

Poslední vlastností produktů bude možnost definování skupin produktů, které bude možno dále používat jako související produkty. Ke každému produktu půjde přiřadit libovolné množství jiných produktů, které pak bude možno ve veřejné části zobrazit např. jako alternativu k danému produktu.

Modul bude obsahovat také výpis všech produktů, který bude implementován pomocí konfigurovatelného datagridu, který umožní správci snadné vyhledávání, třídění a řazení.

Jelikož se k jednotlivým produktům bude zadávat velké množství informací, je důležité, aby v průběhu úprav fungovalo automatické ukládání na pozadí. Pokud bude vytvářen nový produkt, automaticky bude ukládán jako koncept. Produkty uložené jako koncepty bude možno administrovat stejným způsobem jako plnohodnotně uložené produkty, s tím rozdílem, že je nebude možno zobrazit ve veřejné části aplikace a jejich editaci bude moci provádět pouze správce, který je jako koncepty vytvořil.

4.2.4 Správa atributů/parametrů

Fungování a význam atributů/parametrů jsem nastínil již v podkapitole modulu „správa produktů“. V této části upřesním požadovanou funkčnost, možnosti a využití tohoto modulu.

Primárním účelem této části aplikace je vytváření a editování atributů/parametrů. Pod tím si můžeme představit např. hmotnost, barva, výkon atd. Důležité je, aby bylo možno tyto parametry definovat pro libovolné produkty. Výhody plynoucí z využití atributů/parametrů spočívají především v tom, že produktům lze pak na základě atributů/parametrů přiřadit například různé obrázky, rozšíří se možnost pokročilého vyhledávání, nebo bude možno poskytovat dodatečné informace na základě atributů/parametrů. Samozřejmostí pak je to, že si zákazník ve veřejné části aplikace může u produktu zvolit požadovanou variantu a není třeba dodatečné komunikace k dokončení obchodní operace.

V systému budou existovat dva druhy atributů/parametrů. Prvním bude typ „číselník“, pro který se nadefinují hodnoty a z nich se pak bude vybírat pomocí checkboxů. Příkladem použití tohoto druhu jsou např. barevné varianty a pevné definování dostupných barev (červená, modrá, zelená atd.). Druhou možností bude typ „hodnota“, kde se zadá název atributu/parametru a výchozí hodnota. Tuto hodnotu pak bude možno u každého produktu libovolně upravovat.

Modul bude obsahovat také výpis všech atributů/parametrů, který bude implementován pomocí konfigurovatelného datagridu, který umožní správci snadné vyhledávání, třídění a řazení.

4.2.5 Správa skladového systému

V aplikaci bude možno volitelně využívat interního skladového systému. Interní skladový systém bude určen především pro menší klienty, kteří nepoužívají externí aplikace a importují z nich skladové informace k jednotlivým položkám. Skladový systém umožní zaznamenávat jednotlivé operace jako je nákup a prodej u jednotlivých položek v časové souslednosti. Provozovatel aplikace si bude moci zvolit, zda se bude využívat metody FIFO, nebo metody průměrných cen. V průběhu provozu již nebude

možné bez vynulování skladu metodu změnit. Pokud se provozovatel rozhodne skladového systému využívat, dokáže mu systém poskytnout potřebné údaje pro účetnictví nebo daňovou evidenci. V případě, že se bude jednat o plátce DPH, systém mu bude automaticky počítat DPH na vstupu i na výstupu.

Data skladového systému bude možno importovat data z běžně využívaných účetních, nebo skladových aplikací a to např. ve formě XML. V této chvíli se plánuje implementace dat ze systému Pohoda, pro které již má společnost XYZ ze stávající verze aplikace připravené rozhraní. V případě individuální potřeby zákazníka nebude problém podporu dalších programů doimplementovat.

4.2.6 Správa výrobců

Správce aplikace bude mít možnost vytvořit si vlastní databázi výrobců, ke kterým bude možno doplnit libovolné údaje. Účelem modulu je především rozšíření poskytování podrobnějších informací o produktu, ale i o samotném výrobcí.

Se zavedením výrobců se pak rozšíří možnosti vyhledávání podle výrobců. Každému výrobcí bude možno zadat tyto údaje: název výrobce, podrobnější informace, description tag, klíčová slova, url adresa, www stránky, kontakty, logo výrobce. Ve veřejné části aplikace pak bude možné pro každého výrobce generovat samostatnou stránku se všemi podstatnými údaji daného prodejce.

Modul bude obsahovat také výpis všech výrobců, který bude implementován pomocí konfigurovatelného datagridu, který umožní správci snadné vyhledávání, třídění a řazení.

4.2.7 Expediční lhůty, záruční lhůty

Tento modul bude opět provázaný s modulem „správa produktů“ a bude poskytovat možnosti dodacích, záručních lhůt pro jednotlivé produkty. Správce aplikace si bude moci navolit libovolný počet variant dodacích lhůt, ke kterým bude moci přidat i informační popis pro zákazníky. Záruční lhůty budou fungovat na stejném principu, s tím, že se u nich bude definovat ještě samotná záruční lhůta. Ta se bude zadávat v měsících, kvůli možnosti zadávání kratších lhůt, ale i snadnější přepočitatelnosti.

Důležitou vlastností tohoto modulu je možnost definovat tyto lhůty pro jednotlivé skupiny zákazníků. Je třeba, aby si správce mohl definovat rozdílné expediční lhůty např. pro obchodníky, konečné spotřebitele a další.

Výpis všech expedičních i záručních lhůt bude implementován opět pomocí konfigurovatelného datagridu, který umožní správci snadné vyhledávání, třídění a řazení.

4.2.8 Správa komentářů a hodnocení produktů

Jelikož ve veřejné části aplikace bude možnost hodnotit a komentovat produkty, je třeba, aby bylo možné tyto komentáře a hodnocení produktů spravovat. Správce tak bude mít možnost odstranit či editovat např. nevhodné komentáře. U každého produktu bude možnost uzavřít komentáře či hodnocení. Další uživatelé pak již nebudou mít možnost zanechat odezvu. Toto opatření bude existovat především kvůli snadnější údržbě a vyšší přehlednosti komentářů.

U komentářů se budou zaznamenávat základní údaje o návštěvníkovi – ip adresa, jméno, email pro odpověď, komentář.

Při každém přihlášení správce do tohoto modulu mu budou zobrazeny všechny nové příspěvky od jeho posledního přihlášení.

4.2.9 Správa zákazníků

Správa zákazníků je dalším důležitým modulem aplikace. Tato část bude mít na starosti všechny zaregistrované zákazníky a jejich správu. Každý zákazník bude mít svoji vlastní kartu, ze které bude možno pohotově získat veškeré údaje týkající se daného zákazníka. Mezi tyto údaje patří: přihlašovací údaje, osobní údaje, údaje o firmě, fakturační a doručovací údaje, odebírání novinek, skupina zákazníka, individuální slevy, jazykové rozhraní. Tyto údaje bude mít možnost správce měnit. Správce bude mít také možnost zakázat zákazníkovi přístup do aplikace (zákazník se nebude moci přihlásit a využívat výhod přihlášeného zákazníka).

Dále bude tato karta obsahovat statistické údaje o počtu, četnosti a časech přihlášení

zákazníka, přehled jeho objednávek, celkovou útratu, komentáře a hodnocení od tohoto zákazníka.

V systému bude možnost vytvářet skupiny zákazníků, do kterých bude možno následně přiřadit libovolné zákazníky. Skupiny zákazníků budou sloužit především k odlišení různých typů zákazníků – např. konečný zákazník, distributor apod. Těmto skupinám pak bude možno globálně nastavit např. různé ceny, slevy či akce ať již na prodané produkty a služby, nebo na způsoby platby a dodání. Na základě těchto skupin bude možno pak ovlivnit i veřejnou část aplikace. Např. obchodníkům se může zobrazovat jiný způsob výpisu produktů než konečnému zákazníkovi, nebo mohou mít možnost nakupovat po velkých dávkách oproti kusům apod. Nově registrovaný zákazník bude automaticky zařazen do výchozí skupiny.

Prostřednictvím „správy zákazníků“ bude možné obsluhovat i databázi odběratelů novinek a rozesílání novinek. Do této databáze se může zařadit jak registrovaný, tak neregistrovaný návštěvník veřejné části internetové aplikace. Přes administrační rozhraní bude možno odběratele odstranit, případně přeradit do jiné skupiny zákazníků. Newslettery bude možné odesílat jak všem zaregistrovaným návštěvníkům, tak i jednotlivým skupinám zákazníků, nebo jednotlivcům. Newslettery se bude možno vytvářet prostřednictvím jednoduchého grafického rozhraní. Do každého newsletteru musí jít vložit i produkty z katalogů.

Přehled zákazníků, odběratelů novinek a newsletterů bude realizován opět pomocí konfigurovatelného datagridu, aby bylo umožněno přehledné vyhledávání, filtrování a řazení záznamů.

4.2.10 Správa objednávek

Přes modul „správa objednávek“ budou spravovány všechny objednávky, možnosti dopravy, možnosti platby, slevové kupóny, statusy objednávek a slevy z výše objednávek. K výpisu objednávek bude opět použit konfigurovatelný datagrid, který umožní přehledné vyhledávání, řazení či filtrování.

Každá objednávka bude mít svoji detailní stránku, kde budou přehledně zobrazeny veškeré údaje týkající se dané objednávky. Ke každé objednávce se budou na kliknutí

generovat doklady ve formátu PDF. Bude to jednak list s přehledem objednávky, faktura, dodací a záruční list. Tyto doklady budou přímo před-připravené k tisku.

Systém číslování objednávek bude mít na starosti interní mechanismus. Výchozí číslování bude mít tuto podobu: 1004010001. První dvojčíslí značí poslední dvě čísla aktuálního roku, druhé dvojčíslí značí aktuální měsíc, třetí dvojčíslí je dnem měsíce a poslední čtyři čísla se automaticky inkrementují. V případě individuálních požadavků klienta bude tento mechanismus manuálně upraven. Upravovat se bude především tam, kde bude internetová aplikace propojena s jinou aplikací, nebo účetním systémem apod. V některých případech bude dokonce generování dokladů probíhat prostřednictvím aplikací druhých stran a B2B, B2C aplikace bude jen zprostředkovávat data.

Objednávkám bude možno přiřazovat statusy, které si správce v systému definuje. Statusy objednávek mají nejen usnadnit filtrování a především informovat zákazníka o stavu jeho objednávky. V aplikaci budou existovat i pevně definované výchozí statusy objednávek, u kterých bude možno měnit jen informační část (informační text). Tyto výchozí statusy budou navíc obohaceny o funkci automatického rozesílání informačního emailu. Informační část emailu (předmět a tělo emailu) bude taktéž editovatelná a bude existovat možnost zakázat automatické odesílání tohoto emailu. Výchozími statusy pro objednávky budou tyto:

- ✓ Nová objednávka – tento status budou automaticky nabývat všechny nové objednávky. Ve výchozím nastavení se bude po vytvoření odesílat automatický email s přehledem objednaného zboží, platebními, dodacími a jinými údaji.
- ✓ Vyřízená objednávka – tento status bude přiřazován manuálně správcem objednávek. Má za úkol informovat zákazníka, že jeho objednávka byla vyřízena a je připravena k vyzvednutí, nebo byla expedována.
- ✓ Stornovaná objednávka – tento status bude taktéž přiřazován manuálně správcem objednávek. Automaticky zákazníkovi odešle informaci o stornování jeho objednávky.

Pokud bude objednávce udělen status „Vyřízená objednávka“, bude zaznamenán aktuální datum a dojde k automatickému výpočtu data splatnosti pro fakturu.

Prostřednictvím modulu „správa objednávek“ bude možno také definovat možnosti platby a způsoby dopravy. Tyto dva parametry budou vzájemně závislé. Je třeba, aby byla možnost nastavovat libovolné kombinace možnosti plateb a způsobů dopravy a jejich cen. Např. bankovní převod/dobírka a doručení balíku prostřednictvím České pošty, a.s./osobní převzetí atp.

Jednotlivé možnosti platby a způsoby dopravy bude možno také přiřadit jednotlivým skupinám zákazníků. Je třeba, aby si správce mohl definovat rozdílné možnosti platby a způsoby dopravy např. pro obchodníky, konečné spotřebitele a jiné skupiny.

Modul „správa objednávek“ bude mít možnost využívat uživatelské účty a jejich skupiny z modulu, kterému se věnuji o několik částí dále. Jedná se o modul „správa uživatelských rolí“, prostřednictvím nějž budou vytvářeny uživatelské skupiny. Každé uživatelské skupině pak mohou být povoleny moduly a operace, které bude moci provádět. A v souvislosti s objednávkami a skupinami zákazníků pak bude možno vytvořit např. skupinu správců, jež bude mít na starosti objednávky. Z této skupiny pak ještě půjde každému správci nastavit správu objednávek jen od určitých skupin zákazníků – např. objednávky od konečných zákazníků, objednávky od prodejců apod. K jiným objednávkám uživatel přístup mít nebude.

Do modulu „správa objednávek“ budou zařazeny ještě slevové kupóny. Ke každému slevovému kupónu bude možno zadat informační popis, maximální dobu platnosti, výši slevy v procentech a jedinečný kód. Jedinečný kód si bude generovat sama aplikace. Slevový kupón bude možné okamžitě deaktivovat.

Poslední součástí patřící do modulu „správa objednávek“ budou procentuální slevy z objednávek. Tyto slevy budou taktéž provázané se skupinami zákazníků. Díky této komponentě bude existovat možnost nastavení procentuálních slev podle výše objednávky a to v předem definovaných intervalech. Správce pak bude moci nastavit např. 5% slevu pro objednávky od konečných spotřebitelů v cenovém rozmezí 5 – 10 tisíc apod.

4.2.11 Správa obsahu

Prostřednictvím modulu „správa obsahu“ bude mít správce možnost ovlivnit jednotlivé

stránky, či části stránek veřejné části internetové aplikace. Správce tak bude moci vytvářet, editovat a mazat jednotlivé stránky, vydávat články, informovat zákazníky prostřednictvím novinek apod.

První možností je tedy vytváření jednotlivých stránek, které se zařadí ke kategoriím typu *článek*. Pro tento typ obsahu se budou ukládat tyto údaje: název, url adresa, datum poslední změny, text stránky.

Druhým typem obsahu je typ *články*, který bude identický s předchozím typem *článek*, jen bude navíc obsahovat krátký úryvek – tedy jakýsi úvodní zkrácený text, který se bude zobrazovat ve výpisu článků. Tento typ obsahu bude ve veřejné části připomínat blogový styl výpisu článků.

Samostatnou obsahovou částí budou novinky, prostřednictvím nichž bude moci provozovatel aplikace upozorňovat návštěvníky a zákazníky na probíhající novinky, akce a další události. Pro každou novinku se budou zadávat tyto údaje: datum vystavení, název novinky, url adresa, krátký úryvek a kompletní text novinky.

Do kategorií typu *soubory* bude možné nahrávat jakékoliv soubory ke stažení. Díky provázanosti s modulem „správa katalogu“ je možné vytvářet jednoduché souborové systémy. K nahrávaným souborům se budou zadávat údaje jako je název souboru, popisné informace. U souborů bude existovat ještě možnost zvolit typ uživatelské skupiny, pro které bude nahrávaný soubor viditelný. Obvykle se to bude využívat např. pro nahrávání ceníků pro různé typy zákazníků (konečný zákazník, prodejce apod.).

V modulu „správa obsahu“ bude také možné spravovat propagační a reklamní bannery. Přes administrační rozhraní bude možné nahrávat a mazat flashové a obrázkové bannery. Obsluha zobrazování bannerů ve veřejné části pak bude každému klientovi individuálně nastavena – většinou bude vycházet z grafického zpracování aplikace.

Všechny typy článků, textů, souborů ke stažení a novinek bude možno editovat, mazat a bude je možno nastavit jako neveřejné – nebudou moci být zobrazeny ve veřejné části aplikace. Pro přehledný výpis, vyhledávání a filtrování záznamů bude využít opět konfigurovatelný datagridu.

4.2.12 Správa uživatelských rolí

Internetové aplikace jsou specifické tím, že k nim přistupuje velké množství uživatelů. Ovšem ne každý uživatel může mít přístup ke všem částem aplikace. A vzhledem k tomu, že se jedná o internetovou aplikaci, ke které může přistupovat v podstatě kdokoli, je velmi důležité, aby aplikace byla velmi dobře zabezpečena a obsahovala vhodně navrženou správu uživatelských rolí.

Prostřednictvím správy uživatelských rolí bude možno vytvářet uživatelské skupiny, přidělovat a odebírat jim příslušná oprávnění, přiřazovat do těchto skupin uživatele, přidělovat a odebírat oprávnění jednotlivým uživatelům, nebo uživatelům úplně blokovat přístup do aplikace. Celý tento systém musí fungovat tak, aby k jednotlivým informacím mohli přistupovat pouze vybraní uživatelé a mohli s nimi nakládat pouze dovořeným způsobem. Celá správa uživatelských rolí vychází ze zásady minimálních oprávnění (principle of least privilege).

Vzhledem k tomu, že ne každý klient požaduje detailní správu uživatelských rolí, nebo mu obsah aplikace spravuje přímo dodavatelská firma, budou v systému předdefinované základní uživatelské skupiny, kterým bude nastaveno výchozí oprávnění. Předdefinováno bude následujících 6 uživatelských skupin.

- ✓ **Guest** – účet s nejnižším oprávněním, běžný návštěvník internetové aplikace, může prohlížet stránky, katalog a produkty, vkládat produkty do košíku, může provádět objednávky. Nemá přístup do administrační části aplikace.
- ✓ **Customer** – účet se základním oprávněním, registrovaný zákazník, má stejné oprávnění jako předchozí účet, navíc může spravovat svůj účet (adresy, kontakty), prohlížet historii objednávek, vidí přehled stavů svých objednávek, má přístup k PDF dokladům vygenerovaných aplikací. Nemá přístup do administrační části aplikace.
- ✓ **Editor** – účet s rozšířeným oprávněním, má stejná oprávnění jako účet *Customer* s tím, že má navíc přístup do administrační části aplikace, konkrétně do správy katalogu a obsahu. Uživatel s tímto oprávněním může přidávat, editovat a mazat stránky, novinky, soubory ke stažení, bannery, produkty a jejich komentáře, ale i

samotnou strukturu katalogu. Do ostatních částí administračního rozhraní přístup nemá.

- ✓ **Seller** – účet s rozšířeným oprávněním, má stejná oprávnění jako účet *Customer* s tím, že má přístup do administrační části aplikace, konkrétně do modulu „správa objednávek“ a jeho částí týkajících se samotných objednávek. Bude moci přidávat, editovat a mazat statusy objednávek a dále potvrzovat, vyřizovat a stornovat objednávky. Navíc bude mít přístup do katalogu a k jednotlivým produktům, ale nebude mít možnost je jakkoliv editovat či mazat.
- ✓ **Admin** – účet s plným oprávněním, má stejná oprávnění jako všechny předchozí účty. Navíc má možnost vytvářet, editovat a odstraňovat v ostatních částech aplikace. Jedná se především o uživatelské skupiny, zákazníky, nastavení plateb, možností dopravy apod. Jedná se o nejvyšší druh uživatelského oprávnění, které může být provozovateli/správci aplikace nastaveno.
- ✓ **Superadmin** – účet s maximálním oprávněním. Uživatel s tímto druhem účtu nemá žádná omezení. Jedná se o speciální účet, který bude sloužit především pro konfiguraci celé aplikace, prohlížení chybových zpráv a logů. Tento typ účtu nebude možné přidělovat jednotlivým uživatelům, bude vždy individuálně a napevno nastaven při prodeji aplikace. Prostřednictvím tohoto účtu bude možno internetovou aplikaci i zablokovat či odblokovat. Uživatelské přístupy bude mít dispozici firma, která aplikaci prodala, nebo nainstalovala.

Jednotlivé uživatelské role a jejich oprávnění budou moci být v průběhu provozu aplikace jakkoliv modifikovány, nebo rozšířeny. Z výše uvedeného popisu rolí vyplývá, že jednotlivé uživatelské role mohou tvořit různé hierarchie.

Přihlašování do administračního rozhraní bude probíhat prostřednictvím formulářové autentizace. Uživatel zadá přístupové údaje (přihlašovací jméno a heslo) a na základě shody mu bude nastavena identita a přidělena uživatelská oprávnění. Jelikož bude celá aplikace postavena na architektuře MVC, bude přidělování oprávnění spočívat v sestavení tzv. access control listu, ve kterém se definují všechny role, zdroje a jednotlivá oprávnění. Uživatelské role a principy fungování již byly popsány o odstavec výše. Za

zdroje budou považovány jednotlivé akce, které budou definované ve všech Controllerech.

Všechny uživatelské účty budou ukládány centrálně v databázi a budou obsahovat následující údaje. Přihlašovací jméno, heslo, jméno, příjmení, tituly, kontaktní údaje, fakturační údaje, doručovací údaje, datum vytvoření účtu, IP adresa při přihlášení, datum přihlášení. Pro kontaktní, fakturační a doručovací údaje se bude ukládat název firmy, ulice, psč, město, telefon, email, skype apod.

Bezpečnost autentizačních a autorizačních procesů bude mít na starosti aplikace, použitý framework a jeho vlastnosti. Pro ukládání a zabezpečení jednotlivých hesel se bude využívat dnes poměrně bezpečného a doporučeného postupu. Tento postup spočívá v uložení zašifrovaného hesla a přidané salt. Šifrování bude probíhat prostřednictvím funkce sha(). Útočník by pro úspěšné získání hesel musel ukořistit jak databázi hesel, tak i přidávanou salt (bude uložena odděleně a mimo databázi) a ještě by musel úspěšně prolomit jednosměrný šifrovací mechanismus sha().

4.2.13 Statistické a analytické nástroje

V aplikaci budou implementovány také statistické, analytické nástroje. Prostřednictvím statistických nástrojů bude možné sledovat především návštěvnost veřejné části aplikace, aktivitu jednotlivých návštěvníků, četnosti prodeje jednotlivých položek a jejich vkládání do košíku, návštěvnost jednotlivých stránek a částí aplikace. Pokud bude klient využívat interního skladového systému, bude také moci detailně prohlížet skladovou historii jednotlivých položek a přehled o platbách DPH.

Samostatnou součástí statistických a analytických nástrojů bude grafické rozhraní pro sledování a tvorbu přehledů o objednávkách. Objednávky bude možno zobrazovat dle zvoleného období, dle zákazníka, nebo dle skupin zákazníků. Tyto statistiky budou zobrazeny jak pomocí grafů tak i číselně.

Do aplikace budou také přímo integrovány komplexní nástroje pro sledování a analýzu návštěvníků od společnosti Google – nástroje Google Analytics. Prostřednictvím tohoto nástroje získá provozovatel velmi komplexní statistiky, na základě nichž může provádět různá opatření a kroky vedoucí ke zvyšování prodeje.

Nástroje Google Analytics budou provozovatele aplikace informovat především o množství návštěvníků, jejich době strávené v aplikaci a také stránkách, které jsou pro návštěvníky nejzajímavější, odkud jednotliví zákazníci nejčastěji přicházejí a spoustu dalších údajů. Tento nástroj bude také dopomáhat při SEO optimalizaci a kontrole její efektivnosti.

4.2.14 Bezpečnost aplikace a logování událostí

Bezpečnost internetové aplikace je velmi důležitá hlavně kvůli tomu, že skladuje a manipuluje s osobními údaji jednotlivých uživatelů. Důležitá je fyzická ochrana serverů, na kterých poběží samotná aplikace a databázových serverů, na kterých budou uložena její data. Většina aplikací bude provozována prostřednictvím firemních serverů, které jsou neustále monitorovány, a bezpečnosti je věnovaná náležitá péče. Problém může nastat, pokud se bude aplikace umísťovat na soukromé servery zákazníků. Zde se bude muset k bezpečnosti přistupovat individuálně.

Bezpečnost internetové aplikace bude zajištěna v několika úrovních. První úroveň je samotný framework, který při správném využívání velmi napomáhá eliminovat většinu bezpečnostních rizik. Např. tvorba formulářů pomocí tříd a metod frameworku odolných proti všem do dnes známým způsobům útoku apod.

Samotný framework ovšem není všemocný a tak je zapotřebí mít proškolený a zkušený tým pracovníků, kteří jsou s bezpečností internetových aplikací v dostatečné míře seznámeny a neustále své znalosti v této ale i jiných oblastech obohacují. Bezpečnostní díry a rizika mohou vznikat také až díky dalším úpravám a rozšířením aplikace, proto je třeba dbát na jejich důkladný návrh a implementaci a ne jen na minimalizaci časové a finanční náročnosti.

Pro samotnou implementaci, údržbu a rozšiřování aplikace budou platit následující doporučení.

Validace uživatelských vstupů a výstupů – jedná se o jedno z nejkritičtějších míst každé internetové aplikace. Je tedy třeba důsledně kontrolovat, zda uživatelské vstupy a výstupy obsahují pouze to, co obsahovat mají. Každý vstup a výstup je třeba vždy přesně definovat a nastavit mu co nejužší filtr a nepotřebné údaje eliminovat.

Zásada minimálních oprávnění – každý uživatel internetové aplikace by měl mít oprávnění přistupovat pouze k těm částem, které skutečně potřebuje. Tato oprávnění by měla být vždy co nejméně možná. U uživatelských účtů s vyšším oprávněním je vhodné rozdělit pravomoci na potřebnou úroveň, což může minimalizovat případné škody.

Ošetření neočekávaných událostí – nikdy se nedá spoléhat na 100% funkčnost aplikace, nebo jednotlivých součástí. Např. může být chvíli nedostupný databázový server apod. a aplikace musí být na takovéto události připravena. V případě jakéhokoliv selhání je třeba, aby se aplikace zachovala tak, že minimalizuje případné škody – veškeré požadavky a operace budou zamítnuty, zamezí se přístupu a manipulaci s daty.

Důkladné prověření používaných komponent – aplikace bude využívat komponenty a knihovny třetích stran. Výhody plynoucí z využívání cizích knihoven jsou zřejmé. Zvýší se efektivita a ušetří čas věnovaný vlastní implementaci a testování a minimalizuje se množství chyb. Veřejně dostupné knihovny jsou obvykle již odladěné a prověřené v praxi a jejich případné chyby již byly pravděpodobně odstraněny. Vždy je tyto knihovny před jejich nasazením důkladně prověřit a seznámit se se všemi jejich možnostmi a funkcemi tak, aby nemohly být nijak zneužity, případně neznemožnily fungování celé aplikace.

Jednoduché bezpečnostní mechanismy – všechny bezpečnostní mechanismy aplikace by měly být pokud možno co nejpřehlednější, aby uživatele neodradily od používání, nebo je nenutily obcházet. Tato zásada platí jak pro všechny části aplikace – tedy pro uživatelské účty, tak i pro objekty, metody, funkce a komponenty související s bezpečností aplikace.

Na paměti je třeba mít vždy to, že aplikace je tak bezpečná, jako jeho nejslabší část. Těmto slabým místům je tedy třeba věnovat vždy minimálně zvýšenou pozornost.

Logování událostí bude významným pomocníkem pro zvýšení bezpečnosti aplikace. Logování událostí bude v aplikaci napomáhat při odhalování případných útoků a bezpečnostních rizik, dále bude zaznamenávat chyby, které se vyskytly v průběhu fungování aplikace, ale bude také zaznamenávat informace o uživateli a jejich činnostech v aplikaci.

Každý log záznam bude vždy obsahovat informace o uživateli, IP adresu, čas události a popis činnosti. Zaznamenávat se budou následující činnosti:

- ✓ editace a mazání jakýchkoliv dat v databázi
- ✓ editace a mazání jakýchkoliv dat mimo databázi
- ✓ autentizační události
- ✓ veškeré činnosti prováděné v administračním rozhraní
- ✓ chyby v aplikaci
- ✓ debugovací informace (v případě potřeby)

K jednotlivým logům bude mít ve výchozím nastavení přístup přes administrační rozhraní aplikace pouze uživatel zařazený do skupiny „admin“ a výše. Výpis logů bude realizován prostřednictvím konfigurovatelného datagridu, aby bylo možné data přehledně řadit a filtrovat.

Zálohování dat aplikace bude dalším z bezpečnostních prvků. Jelikož dnes každá trochu kvalitnější hostingová společnost provádí zálohy minimálně jednou za 24 hodin, není prioritou vyvíjet a implementovat složité zálohovací mechanismy. Zálohovat se budou pouze data týkající se zákazníků a jejich objednávek. Interval této zálohy půjde nastavit prostřednictvím administrační části aplikace a účtu typu „Superadmin“. Zálohovaná data z databáze se budou ukládat do složek a souborů na disku, kde bude uložena aplikace. Z bezpečnostního hlediska je důležité zamezit do těchto složek přístup z internetu. Soubory obsahující zálohovaná data se budou pojmenovávat podle času provedení zálohy. Pro zvýšení bezpečnosti a snížení možnosti neoprávněné manipulace s těmito daty budou soubory ukládány do zip archivů chráněných heslem.

K přihlídnutí k dosavadním zkušenostem se zálohováním a obnovováním aplikací ze zálohy nebude aplikace obsahovat grafické rozhraní pro načtení a zpracování zálohovaných souborů. V případě potřeby bude probíhat obnovení databáze prozatím pouze manuálním způsobem.

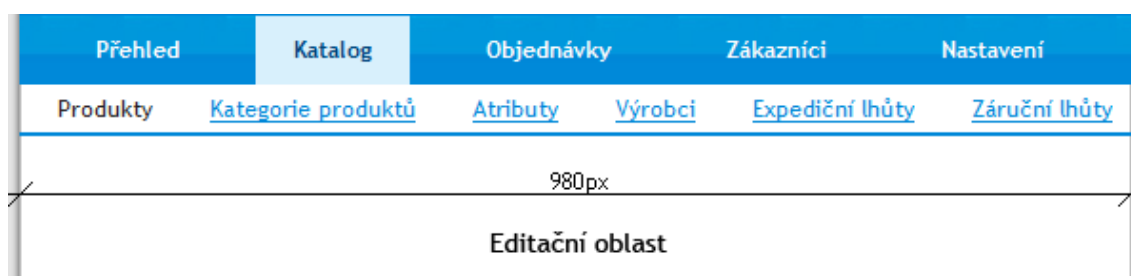
4.2.15 Ostatní funkce

Prostřednictvím uživatelského účtu typu superadmin bude možné přes administrační rozhraní provádět základní konfiguraci aplikace. Bude možné editovat veškeré údaje o prodejci – tedy název, kontaktní a fakturační údaje, dále bude možné nastavit emailové adresy ze kterých budou chodit automatické emaily zákazníkům, ale také obsah těchto automatických emailů. V případě, že bude provozovatel plátcem DPH, bude možné nastavovat jednotlivé sazby DPH.

Jelikož bude celá aplikace multijazyčná, může provozovatel požadovat zobrazování cen nejen v základní měně – např. v Kč, ale i v jiných měnách. Přes administrační rozhraní bude možné zadat třímístný kód měny a aplikace si aktuální kurz načte přes kurzovní lístek ČNB. Bude možné nastavit také četnost aktualizace kurzovního lístku, nebo kurzovou přírážku – tedy částku, o kterou se má kurz ČNB automaticky upravit.

4.2.16 Grafické rozhraní administrační části aplikace

Vzhledem k tomu, že se bude s administračním rozhraním aplikace poměrně často pracovat a bude se prostřednictvím něj editovat velké množství údajů, je důležité, aby byla pro editaci určena co největší obrazová plocha a ovládací prvky do této plochy co nejméně zasahovaly.



Obrázek 13: Ukázka návrhu grafického rozhraní v administrační části aplikace

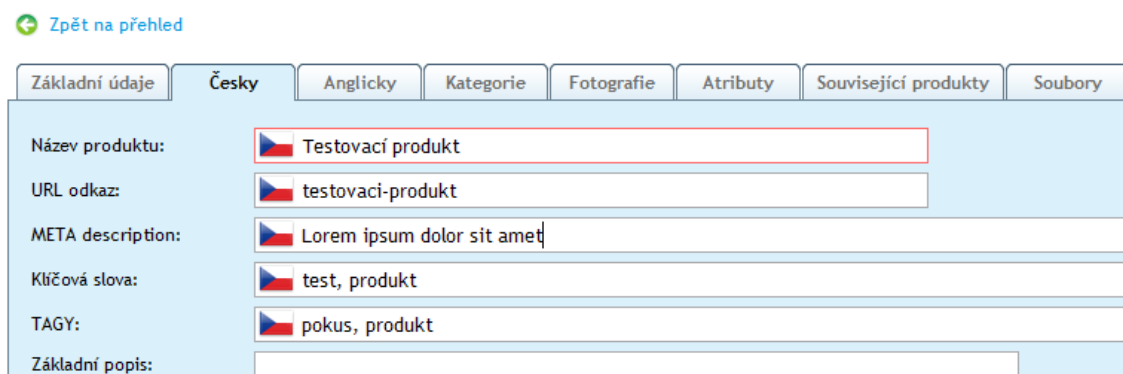
Ovládací prvky administračního rozhraní budou umístěny vždy u horního okraje stránky. V první řadě budou tlačítka pro vstup do jednotlivých modulů a po zvolení daného modulu se zobrazí druhý řádek, který bude obsahovat navigační tlačítka pro jednotlivé funkce daného modulu. Tento návrh navigační části zajistí dostatek prostoru

pro editační oblast, neboť bude možné využít šířku stránky v maximální míře.

Pod navigační oblastí se bude nacházet ještě drobečková navigace, která bude zobrazovat úroveň zanoření, ve které se správce aplikace právě nachází. Pod drobečkovou navigací již bude následovat editační oblast. V editační oblasti se budou nacházet veškeré formuláře, výpisy a další prvky, prostřednictvím nichž bude mít správce možnost ovlivňovat chování a obsah aplikace.

Vzhledem k tomu, že veřejné rozhraní má být multijazyčné, bude se prostřednictvím administračního rozhraní zadávat velké množství údajů. Aby byla zachována přehlednost, bude se využívat systému kartiček, který umožní, aby byly údaje pro každý jazyk umístěny na vlastní podstránce.

Následující obrázek demonstruje využití systému kartiček u editace produktu. Po přidání další jazykové mutace se pouze zvýší počet kartiček a nedojde tak ke snížení přehlednosti editovaných údajů.



The screenshot shows an administrative interface for editing a product. At the top left, there is a link labeled "Zpět na přehled" with a left-pointing arrow. Below this is a horizontal navigation bar with several tabs: "Základní údaje", "Česky", "Anglicky", "Kategorie", "Fotografie", "Atributy", "Související produkty", and "Soubory". The "Česky" tab is currently selected. The main content area is a form with the following fields:

| | |
|-------------------|---|
| Název produktu: | <input type="text" value="Testovací produkt"/> |
| URL odkaz: | <input type="text" value="testovací-produkt"/> |
| META description: | <input type="text" value="Lorem ipsum dolor sit amet"/> |
| Klíčová slova: | <input type="text" value="test, produkt"/> |
| TAGY: | <input type="text" value="pokus, produkt"/> |
| Základní popis: | <input type="text"/> |

Obrázek 14: GUI administrační části - systém kartiček

Pomocí kartiček bude možné členit i další editovatelný obsah a rozčlenit ho např. podle na sobě závislých částí.

Pro editaci větších textových částí jako jsou samostatné stránky, popisy produktů, kategorií apod. bude využíván grafický WYSIWYG editor CKeditor a pro upload souborů pro obsah spravovaný tímto editorem bude zakoupena licence AJAXového upload manageru CKFinder.

4.3 PHP Framework jako jádro aplikace

Jak již bylo výše zmiňováno, aplikaci bude pohánět PHP framework, jehož účelem bude především zvýšit bezpečnost, produktivitu práce, správu a údržbu a snadnou rozšiřitelnost aplikace. V dnešní době existuje několik desítek takovýchto frameworků a každý z nich má své výhody i nevýhody. Snadno lze mezi těmito frameworky najít i takové, které se pro tuto aplikaci nehodí, nebo které již delší dobu nikdo nevyvíjí a nemají tak do budoucna žádnou perspektivu. Vývoj, údržbu, správu a rozšiřování aplikace by takovýto framework mohl spíše zkomplikovat. Tato kapitola se zabývá porovnáním a následným výběrem vhodného frameworku pro aplikaci a s největší pravděpodobností také pro další a na této aplikaci nezávislé aplikace vznikající ve společnosti XYZ, s.r.o.

Výkon a rychlost jednotlivých frameworků jsme neměřili, vycházeli jsme z vlastních poznatků a z poznatků programátorů a testerů, kteří své zkušenosti sdílí na odborných stránkách a diskuzích po internetu.

Nejdůležitějším kritériem pro nás byla snadná použitelnost, kvalitní a dostupná dokumentace spolu s komunitou, která dokáže rychle poradit, nebo alespoň ukázat směr jakým se vydat, ale také aktuálnost a nadčasovost řešení a na závěr naše osobní zkušenosti s frameworkem. Do užšího výběru byly zařazeny pouze frameworky, se kterými jsme realizovali alespoň jeden projekt a to z toho důvodu, aby se realizace projektu nezarazila právě na neznalosti frameworku. Samozřejmě s přechodem na nový způsob vývoje aplikací bude třeba zvýšit kvalifikovanost zaměstnanců, což se v novém projektu promítne jak v časovém měřítku, tak i v nákladových položkách.

4.3.1 CakePHP

Oficiální stránky frameworku: <http://cakephp.org>. CakePHP je určen pro PHP 4 i 5. Jedná se o poměrně rozšířený framework, což nahrává kvalitní podpoře, existenci velkého množství rozšíření a doplňků, přehledné a kvalitní dokumentaci. Pozitivní je i to, že má tento framework fanoušky i v České republice a jsou tedy k nalezení i návody v češtině. Aktuální verze frameworku je 1.3.0.

CakePHP obsahuje poměrně kvalitní ORM knihovnu pro práci s daty a spolupracuje většinou dnes využívaných databází. Jedná se o klasický MVC framework, jehož výhody spočívají především v rychlosti a malé paměťové náročnosti a velmi kvalitní podpoře a dokumentaci. K frameworku je dále možné stáhnout velké množství komponent a pluginů – např. komponenty pro práci s PDF, e-maily, soubory, grafy a další. Množství doplňků se neustále zvětšuje.

Mírnou slabinou jsou šablony, které se chovají jako klasické php soubory. Jelikož v našem případě budou šablony editovat i lidé s minimálními znalostmi PHP mohou se jim šablony jevit jako nepřehledné.

```
<table>
  <?php foreach($rows as $row): ?>
    <tr>
      <td><?php echo htmlspecialchars($row->name); ?></td>
      <td><?php echo htmlspecialchars($row->surname); ?></td>
    </tr>
  <?php endforeach; ?>
</table>
```

Obrázek 15: Zápis cyklu foreach v šablonách CakePHP

S CakePHP máme ve společnosti nejméně zkušeností. Prostřednictvím tohoto frameworku byl realizován pouze jeden projekt, který byl v okamžiku vstupu společnosti do projektu již z menší části rozpracován.

Výhody CakePHP

- ✓ velká rozšířenost, komunita, podpora, dokumentace
- ✓ jednoduchost, strmá křivka učení
- ✓ velké množství doplňků
- ✓ existence ORM knihovny

4.3.2 Nette Framework

Jedná se o objektově orientovaný framework od českého autora Davida Grudla.

Oficiální stránky frameworku jsou k nalezení na adrese <http://nettephp.com>. Nette Framework je vyvíjen čistě pro PHP 5, konkrétně pro PHP 5.2 a vyšší a jeho aktuální verze je 0.9.4. Nette Framework je vydáván i ve verzi pro PHP 5.3 a vyšší, proto je možné využívat pokročilejších programovacích technik a nových vlastností vyšší verze PHP – např. namespaces. Největší předností Nette frameworku je zaměření na bezpečnost, nízká paměťová náročnost a vysoká rychlost. Pro vývoj aplikací pak obsahuje velmi kvalitní ladící nástroje, které usnadňují vývojářům jejich práci. Mírnou nevýhodou frameworku je absence databázové vrstvy, nebo kvalitního ORM. Tuto slabinu ovšem snižuje existence dibi knihovny od stejného autora, která se dá s Nette Frameworkem výborně propojit.

Největším slabinou frameworku bývala dokumentace. Na ní se ovšem nyní poměrně intenzivně pracuje a tak je každým dnem lepší. Pozitivní na Nette Frameworku je také jeho rozšíření a aktivní komunita v České republice.

Pro Nette Framework již v tuto chvíli existuje poměrně velké množství zajímavých doplňků počínaje kurzovním lístkem, generováním PDF, knihovnami pro rekurzivní menu struktury až po datagrid. Nové doplňky prostřednictvím komunity neustále vznikají. Samozřejmostí je možnost využívat knihovny např. z velmi populárního Zend Frameworku.

Nette Framework má poměrně propracovanou práci se šablonami a výsledný kód je velmi krátký a přehledný i pro kodéry s minimální znalostí PHP.

```
<table>
  <tr n:foreach="$rows as $row">
    <td>{ $row->name}</td>
    <td>{ $row->surname}</td>
  </tr>
</table>
```

Obrázek 16: Zápis cyklu foreach pomocí šablon v Nette Frameworku

S Nette Frameworkem máme ve společnosti XYZ, s.r.o. největší zkušenosti. S jeho podporou již bylo realizováno 6 internetových aplikací a u všech byly zaznamenány jeho pozitivní přínosy.

Výhody Nette

- ✓ vysoký důraz na bezpečnost
- ✓ vysoká rychlost a nízká paměťová náročnost, minimalizovaná verze
- ✓ podpora namespaces – možnost přehlednějšího kódu, zabránění kolizí tříd
- ✓ propracovaný systém šablon
- ✓ výborná podpora AJAXu a dalších WEB 2.0 technologií
- ✓ velmi kvalitní OOP návrh

4.3.3 Zend Framework

Stejně jako Nette Framework je Zend objektivě orientovaným frameworkem, který slouží především pro efektivní vývoj internetových aplikací. Jedná se pravděpodobně o nejpopulárnější a nejrozšířenější PHP framework vůbec.

Oficiální stránky frameworku jsou <http://framework.zend.com>. Nejnovější dostupnou verzí je verze 1.10, která vyžaduje minimálně PHP 5.2.4. Zend je taktéž plně objektivě orientovaný framework, který má po celém světě velmi silnou komunitu. Právě proto pro něj existuje obrovské množství doplňků a rozšíření. Co se týká dokumentace, tak ta je na velmi vysoké úrovni – ať již oficiální, nebo komunitní na různých odborných serverech a diskuzních fórech.

Asi největší nevýhoda Zend Frameworku pro naše účely spočívá v jeho nižší rychlosti, vysoké paměťové náročnosti a velké rozsáhlosti. Na všechny tyto problémy existuje řešení – ať již v podobě eAcceleratoru či jiných nástrojů pro cachování skriptů, nebo manuální zredukování nepotřebných modulů a doplňků pouze na potřebné části, ne vždy může být toto řešení optimální a minimálně zvyšuje režii.

Z pohledu budoucnosti vypadá zajímavě aktuálně vyvíjená druhá větev, která bude později vydaná jako verze 2.0. Vývojáři jsou si vědomi současného stavu frameworku, ale také reagují na požadavky uživatelů, kteří vyžadují např. namespaces apod. a tak se pustili do kompletního refactoringu frameworku. Pro naše účely je tato vývojová verze ovšem nevhodná.

Zend Framework neobsahuje v základu žádný propracovaný šablonovací systém. Není ovšem problém do něj implementovat externí šablonovací systém, čímž ale opět dojde ke zvětšení již tak velkého systému a snížení údržby a možnosti jednoduché aktualizace. Způsob zápisu kódu v šablonách ve výchozí verzi je tedy taktéž v podobě klasických php skriptů – tedy méně přehledný a obsáhlejší.

```
<table>
<?php foreach ($rows as $row): ?>
  <tr>
    <td><?php echo $this->escape($row->name); ?></td>
    <td><?php echo $this->escape($row->surname); ?></td>
  </tr>
<?php endforeach; ?>
</table>
```

Obrázek 17: Zápis cyklu *foreach* v šablonách Zend Frameworku

I se Zend Frameworkem máme ve společnosti XYZ, s.r.o. zkušenosti, když pomocí něj byly realizovány dva projekty. Projekty byly menšího rozsahu než ty u Nette Frameworku, ale všeobecně můžeme říci, že se nám se Zendem pracovalo o něco méně pohodlněji, než právě s Nette Frameworkem.

Výhody Zend Frameworku

- ✓ vyzrálost a ověřenost
- ✓ obrovské množství knihoven, doplňků a rozšíření v podstatě na cokoliv
- ✓ kvalitní dokumentace, obrovská komunita
- ✓ podpora externích knihoven
- ✓ podpora AJAXu

Využívání jakéhokoliv frameworku v našich aplikacích sebou přinese i nové úkoly. Každá aplikace, každý projekt, ale i každý framework má svůj životní cyklus, v tomto případě postupný vývoj. To platí i u všech ostatních frameworků a pro nás to znamená jediné. To co funguje v dnešní verzi frameworku již nemusí fungovat ve verzi zítřejší. V jednotlivých verzích nemusí být vždy zachována zpětná kompatibilita, i když se o to

vývojáři snaží. A zůstat na stále stejné verzi frameworku také není ideální, jelikož budeme ochuzeni o nové funkce, bezpečnostní a optimalizační zlepšení. Z tohoto důvodu budou nově kladeny mnohem vyšší požadavky na jednotlivé programátory. Jejich novým úkolem bude sledovat a zachycovat vývoj daného frameworku a aktivně se v něm neustále zdokonalovat.

Vyzrállost dnešních frameworků znamená to, že ať si vybereme jakýkoliv, většinou chybu neuděláme. Pro firmu XYZ bylo důležité, aby měla s frameworkem již nějaké pozitivní zkušenosti, dále k danému frameworku existovala kvalitní dokumentace a ochotná komunita a v neposlední řadě aby se nejednalo o již „mrtvý“ projekt, který se již dále nerozšiřuje.

Výběru frameworku mohli ovlivnit všichni programátoři ve společnosti a všichni se shodli na využití Nette frameworku. Výběr byl podpořen především jejich zkušenostmi z posledních projektů, na kterých byl tento framework taktéž využit. Ocenili u něj především rychlou křivku učení, skvěle navržené rozhraní pro velmi často využívané prvky a skvělou podporu AJAXu, skvělé doplňky a rozšíření a kvalitní systém šablon.

Nette framework pak bude podporovat JavaScriptový framework jQuery, který je ve firmě používán již delší dobu. jQuery bude mít na starosti především interaktivní část aplikace a bude tak aktivně působit mezi uživatelem aplikace a její serverovou částí.

4.3.4 Komponenta datagrid

Datagrid je grafická komponenta umožňující zobrazit určitá data prostřednictvím tabulky a následně dovoluje tyto data filtrovat, třídít, zvýrazňovat či jinak s nimi pracovat. Nejčastěji tímto způsobem zobrazuje data načtená z databáze.

V našem případě bude datagrid stěžejní komponentou využívanou v administračním rozhraní aplikace. Tato komponenta bude využita na všech stránkách zobrazující větší množství souvisejících dat – viz požadavky na administrační rozhraní.

| | Zákazník ↕ | Adresa ↕ | Datum ↕ | Stav ↕ | Kredit ↕ | Akce |
|-------------------------------------|----------------------------------|--------------------------|----------------------|--------|----------------------|-------------|
| | <input type="text" value="Inc"/> | <input type="text"/> | <input type="text"/> | ? ▾ | <input type="text"/> | Filtrovat 🗄 |
| <input type="checkbox"/> | Diecast Classics Inc. | 7586 Pompton St. | 07/20/2004 | | 100 600 Kč | |
| <input type="checkbox"/> | Muscle Machine Inc | 4092 Furth Circle | 07/07/2004 | | 138 500 Kč | |
| <input type="checkbox"/> | Gift Depot Inc. | 25593 South Bay Ln. | 06/28/2004 | | 84 300 Kč | |
| <input checked="" type="checkbox"/> | Down Under Souvenirs, Inc | 162-164 Grafton Road | 06/03/2004 | | 88 000 Kč | |
| <input checked="" type="checkbox"/> | Tekni Collectables Inc. | 7476 Moss Rd. | 05/18/2004 | | 43 000 Kč | |
| <input type="checkbox"/> | Land of Toys Inc. | 897 Long Airport Avenue | 05/07/2004 | | 114 900 Kč | |
| <input type="checkbox"/> | Super Scale Inc. | 567 North Pendale Street | 05/04/2004 | | 95 400 Kč | |
| <input checked="" type="checkbox"/> | Oulu Toy Supplies, Inc. | Torikatu 38 | 04/12/2004 | | 90 500 Kč | |
| <input type="checkbox"/> | Vitachrome Inc. | 2678 Kingston Rd. | 04/05/2004 | | 76 400 Kč | |
| <input type="checkbox"/> | Motor Mint Distributors Inc. | 11328 Douglas Av. | 04/03/2004 | | 72 600 Kč | |

Vybrané: Proved' 3 Položky 21 až 30 z 44 | Zobraz: | Resetovat stav

Obrázek 18: Ukázka komponenty datagrid pro Nette od Romana Sklenáře

Datagrid je poměrně snadno konfigurovatelný a v podstatě s minimálním úsilím připravíme přehledný výpis dat. Samozřejmostí datagridu je i podpora lokalizace, proto jej lze bez obav využívat i ve vícejazyčných systémech. Následující kód demonstruje jeho základní použití. Nejlepším způsobem je využití tzv. továrničky.

```
protected function createComponentGrid($name) {
    $grid = new DataGrid;
    $grid->bindDataTable($this->dao->getAll());
    $this->addComponent($grid, $name);
    $grid->itemsPerPage = 15;
}
```

Obrázek 19: Základní konfigurace datagridu

Samotné vykreslení datagridu v šabloně pak obstará následující kód uvedený níže. Při nahrání správných JavaScriptových kódů má takto definovaný datagrid plnou AJAXovou podporu a zajišťuje tak velmi pozitivní uživatelský požitek.

```
{* vykreslení v šabloně *}
@{widget grid}
```

Obrázek 20: Vykreslení datagridu v šabloně

Datagrid byl uvolněn pod licenci *New BSD License*, což nám umožňuje jeho plné nasazení do naší aplikace.

4.3.5 Multijazyčnost aplikace

Hned v úvodní části návrhu byl zmíněn poměrně zásadní požadavek na celou aplikaci a to kompletní multijazyčnost celé aplikace s tím, že pro administrační rozhraní bude dostačující jako výchozí čeština a pak angličtina. Veřejnou část aplikace musí být možno provozovat s libovolným počtem jazykových mutací.

V aplikaci budou tři typy dat, které bude třeba překládat. Prvním typem dat jsou texty uživatelského prostředí – tedy formuláře, chybové hlášky, ovládání aplikace a další. Tyto texty budou uloženy převážně v šablonách. Druhým typem dat jsou uživatelská data, neboli dynamický obsah, který bude ukládán v databázi. Uživatelská data budou ukládána prostřednictvím grafického administračního rozhraní a tak je důležité. Posledním typem dat jsou url adresy, které jsou z části závislé i na uživatelských datech v databázi. Část url, která nevychází z databáze, se definuje v tzv. routeru, nebo jiného automatického překladače, který zajistí pro každý jazyk správný překlad.

Multijazyčnost uživatelského prostředí

Pro řešení tohoto problému existuje v dnešní době velké množství postupů, kde každý má své výhody a nevýhody a ne každý je vhodný pro daný typ a velikost aplikace. Doposud jsme pro multijazyčné aplikace a jejich uživatelské prostředí využívali tzv. asociativních polí – viz následující ukázka.

```

$translator = array(
    "cs" => array(
        "naviCatalog" => "Katalog",
        "naviProduction" => "Výroba",
        "naviAboutUs" => "O nás",
    ),
    "en" => array(
        "naviCatalog" => "Catalogue",
        "naviProduction" => "Production",
        "naviAboutUs" => "About us",
    ),
);

```

Obrázek 21: Nástin řešení překladů pomocí asociativního pole

Vhodnost využití tohoto řešení některé zdroje uvádějí při překládání cca 500 – 1000 položek, jiné až do výše 4 – 5 tis. položek. Při tomto množství je toto řešení velmi rychlé a paměťově nenáročné. Nevýhoda tohoto řešení spočívá v tom, že neexistuje žádná možnost ukládání plurálů, pravidel a dalších překladů a je tedy třeba ji vymyslet a implementovat. Dříve se plurály a odlišná pravidla překládání u internetových aplikací téměř neřešily, důležité bylo, že to bylo tak nějak přeložené. U dnešní profesionální aplikace je takové řešení již nepřijatelné.

Mezi dalšími způsoby řešení této problematiky dnes nejčastěji nalezneme využívání jazykových XML dokumentů, ukládání dat uživatelského rozhraní přímo do databáze, nebo nástroj GetText.

Pro naši aplikaci a překlady uživatelského prostředí budeme využívat GetText. Jedná se o velmi rozšířenou technologii využívanou především při vývoji open source aplikací. Jeho výhodou je nativní podpora PHP, vysoká rychlost a ověřenost mnoha profesionály a časem. Vysoká rychlost tohoto řešení spočívá především v tom, že jsou překlady mimo server uloženy do binární podoby, server si je poté načte do sdílené paměti a tam je ponechá a nemusí je tak stále dokola načítat. Další výhodou GetTextového řešení je možnost provádět překlad bez jakéhokoliv zásahu do aplikace – může jej tedy překládat jazykový odborník, který nepotřebuje mít přístup do aplikace.

Použití GetTextového překladače v Nette frameworku je velmi jednoduché, v podstatě

jej stačí někde v centrální části – nejlépe v nějakém BasePresenteru zaregistrovat a předat mu typ jazyka, který od něj požadujeme.

```
abstract class BasePresenter extends Presenter {
    public function beforeRender() {
        $this->template->setTranslator(
            new GettextTranslator(APP_DIR.'/locale/cs_CZ/locale.mo'), 'cs'
        );
    }
}
```

Obrázek 22: Příklad registrace GetText translátoru v BasePresenteru

Druhým požadavkem na správnou funkčnost překladů je trochu rozdílný zápis v šablonách aplikace. Aby framework rozeznal, že má text překládat, použije se speciální makro zápis – viz následující obrázek.

```
<div id="header">
    <h1>{_'Hello world!'}</h1>
    <p>{_'You have %d new e-mail.', 1}</p>
</div>
```

Obrázek 23: Příklad zápisu textu v šabloně určeného k překladu

Nette framework samozřejmě zahrnuje podporu např. pro překládání formulářů a dalších součástí frameworku. Překládat pomocí GetTextu je možné i různé doplňky a rozšíření, pokud s tím bylo v jejich návrhu počítáno. Stačí pouze správně nastavit translátor.

Samotný překlad pak bude probíhat pomocí externích aplikací – např. Poedit, Gted nebo jiných. Na jednu stranu je to výhoda, na druhou stranu nebude možné tyto texty měnit např. přes administrační rozhraní, jelikož budou uloženy v binární podobě. Jelikož se jedná o texty formulářů, chybových hlášek apod. a tyto texty se budou měnit maximálně při instalaci nové aplikace u zákazníka, je toto řešení vyhovující.

Následující obrázek demonstruje překlad pomocí nástroje Poedit. Za povšimnutí stojí možnost zadání plurálů. Každý jazyk má jiný počet plurálů, i s tímto toto řešení počítá.

| Originální řetězec | Překlad | |
|---------------------------|--------------------------|---------------------|
| 🗨 Hello world! | Hello world! | |
| 🗨 You have %d new e-mail. | Máte %d nový e-mail | |
| Singulár: | You have %d new e-mail. | |
| Plurál: | You have %d new e-mails. | |
| Forma 0 (např. "1") | Forma 1 (např. "2") | Forma 2 (např. "5") |
| Máte %d nových e-mailů. | | |

Obrázek 24: Doplnování překladu pomocí aplikace Poedit

Tento návrh tedy počítá s tím, že má být aplikace snadno rozšiřitelná a udržitelná. V budoucnu tak nebude problém překladač vyměnit za jiný typ, který musí pouze implementovat rozhraní ITranslator.

Multijazyčnost uživatelských dat

Pod uživatelskými, neboli dynamickými daty si můžeme představit např. názvy produktů, texty článků, novinek apod. Tento typ dat bude ukládán do databáze, odkud si je bude načítat sama aplikace na základě vhodných SQL dotazů. Veškeré načítání a práci s daty bude mít na starosti model aplikace.

Multijazyčnost URL adres

Aby byla B2B, B2C aplikace kompletně lokalizovaná, zbývá vyřešit překlad jednotlivých url adres. Url adresy se budou překládat pouze ve veřejné části aplikace a to ať již kvůli vyšší přehlednosti pro uživatele, ale samozřejmě i z důvodu SEO optimalizace. K překladu jednotlivých částí url adres využijeme vlastností frameworku a jeho routeru.

```

// pridani prekladoveho slovníku pro presenter
Route::addStyle('#cs-presenter', 'presenter');

// definovani prekladoveho slovníku pro presenter
Route::setStyleProperty('#cs-presenter', Route::FILTER_TABLE, array(
    'o-nas' => 'AboutUs',
    'vyroba' => 'Production',
    'kontakt' => 'Contact',
));

// inicializace routeru
$router = $application->getRouter();

// ruta, která pracuje s definovaným cs slovníkem
$router[] = new Route('<presenter #cs-presenter>/<action>/<id>', array(
    'lang' => 'cs',
    'presenter' => 'Default',
    'action' => 'default',
    'id' => NULL,
));

```

Obrázek 25: Ukázka definice routeru a překladového slovníku url adres

Pro jednotlivé jazyky lze nadefinovat překladové tabulky, podle níž pak router dle aktuálního jazyka vyhledává. V případě některých pohledů budou url adresy uloženy v databázi, proto se routeru nastaví funkce, pomocí které si tyto url načte a předpřipraví. V tomto případě se bude překladová tabulka cachovat, aby byla zajištěna co nejvyšší rychlost načítání aplikace.

4.4 Návrh datového skladu

Na základě specifikovaných požadavků na B2B, B2C aplikaci a její ukládání a manipulaci s většinou dynamických dat bude využita relační databáze MySQL. Tento typ úložiště se dnes využívá u velké většiny internetových a intranetových aplikací. Do karet mu hraje především jeho poměr cena/výkon, ale samozřejmě i další velmi dobré parametry.

Všechna data se budou ukládat v kódování UTF-8 a jako datový formát tabulek bude použit typ InnoDB, který dokáže pracovat s transakcemi a cizími klíči. Mazání a editace položek bude probíhat pomocí cizích klíčů a funkcí typu ON DELETE CASCADE

apod., čímž dojde k ponechání režie na databázovém, který je pro tyto operace optimalizován.

Databázová struktura aplikace bude poměrně rozsáhlá, což je dáno zvoleným způsobem ukládání multijazyčných dat. Na druhou stranu se jedná o nejrychlejší možné datové úložiště.

Veškerá data z databáze budou načítána tzv. lazy způsobem, tedy až v okamžiku, kdy budou požadována. Tento způsob načítání bude v aplikaci zajišťovat model s pomocí controlleru.

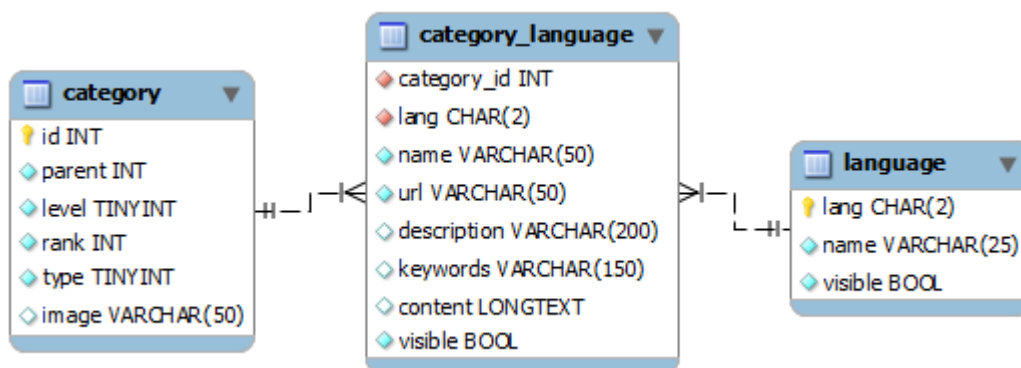
Pro všechny datové tabulky bude definován pevný prefix, který bude zadáván při každé instalaci aplikace. Účelem prefixu je zamezit kolizi nových datových entit se stávajícími. Ne vždy je aplikace instalována do prázdné databáze.

Jednotlivé tabulky jsou většinou dekomponovány tak, aby splňovaly první, druhou, třetí a další normální formu. U některých tabulek dekompozice nebyla zcela provedena, bral se ohled na aplikaci a její spolupráci s daty, případně na její snadnější rozšíření v budoucnosti.

Veškeré názvy tabulek a sloupců datového skladu budou v anglickém jazyce a v jednotném čísle. Tzn. například tabulka produktů se bude jmenovat *product* atd.

4.4.1 Návrh datového skladu pro správu katalogu

Zásadní tabulkou pro celý katalog, do kterého bude možné následně řadit produkty, články a další stránky dle specifikace v kapitole správa katalogu bude tabulka *category*. Jelikož budou ke kategoriím ukládány i multijazyčná data pro několik jazykových verzí, bude existovat tabulka *category_language*, do které budou pod cizím klíčem tato data ukládána. Stejným způsobem budou dekomponovány i další tabulky, do kterých budou multijazyčná data ukládána.

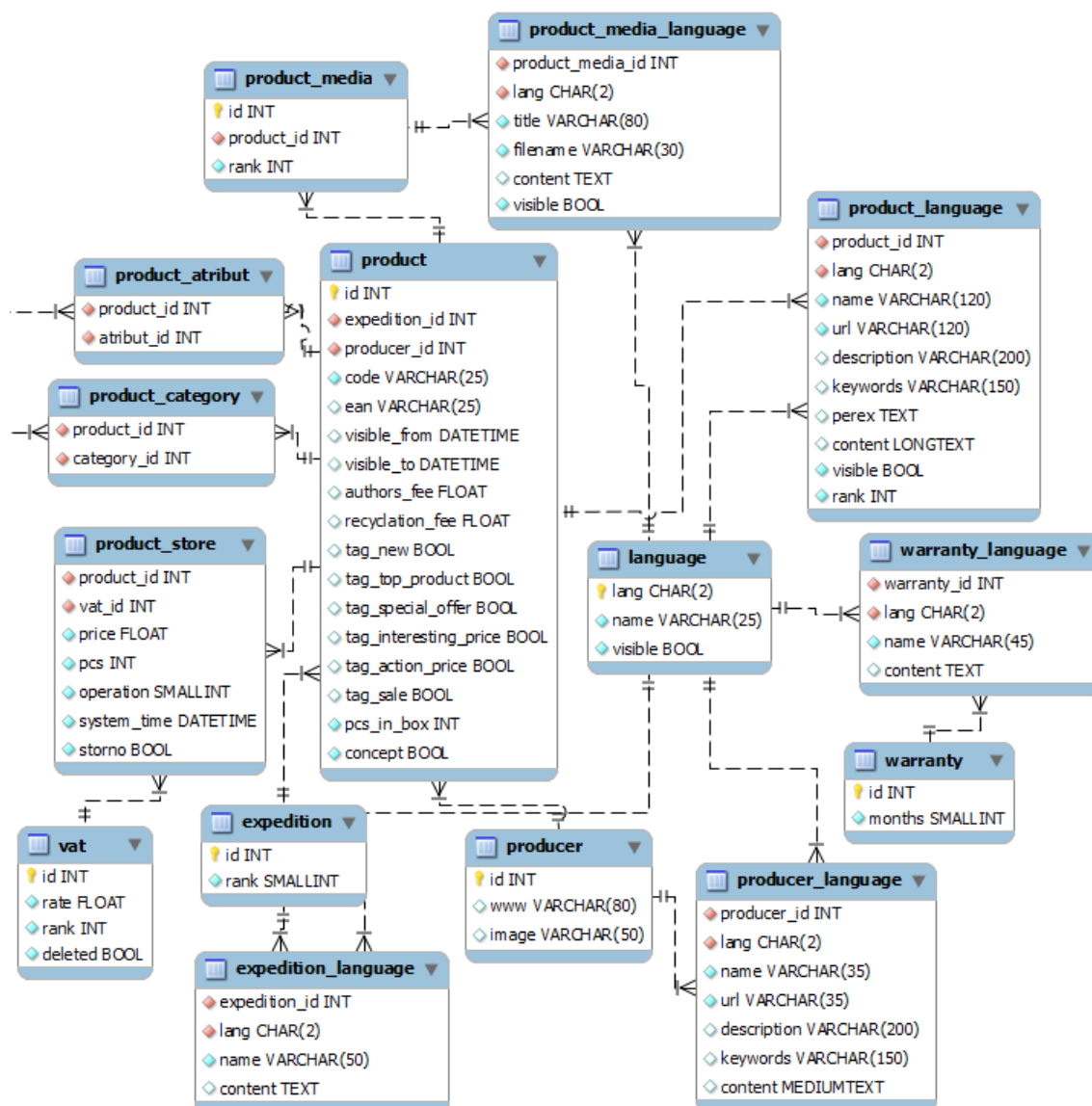


Obrázek 26: Dekompozice multijazyčných dat u katalogu

V tabulce *language* budou uloženy informace o jazykových variantách aplikace. Primární klíč *lang* typu CHAR(2) bude vždy obsahovat zkratku jazyka dle normy ISO 639-1. Ve všech tabulkách obsahující multijazyčné informace tak bude vždy umístěn cizí klíč *lang*, prostřednictvím kterého budou tabulky provázány s tabulkou *language*.

4.4.2 Návrh datového skladu pro správu produktů

Datový sklad pro správu produktů bude velmi rozsáhlý. Je to dáno především ukládáním velkého množství multijazyčných dat. Hlavní tabulkou k produktům je tabulka *product*, která byla dekomponována na tabulku *product_language*, která obsahuje data pro jednotlivé jazyky. Obdobným způsobem jsou řešeny i další tabulky související se ukládáním informací o produktech.



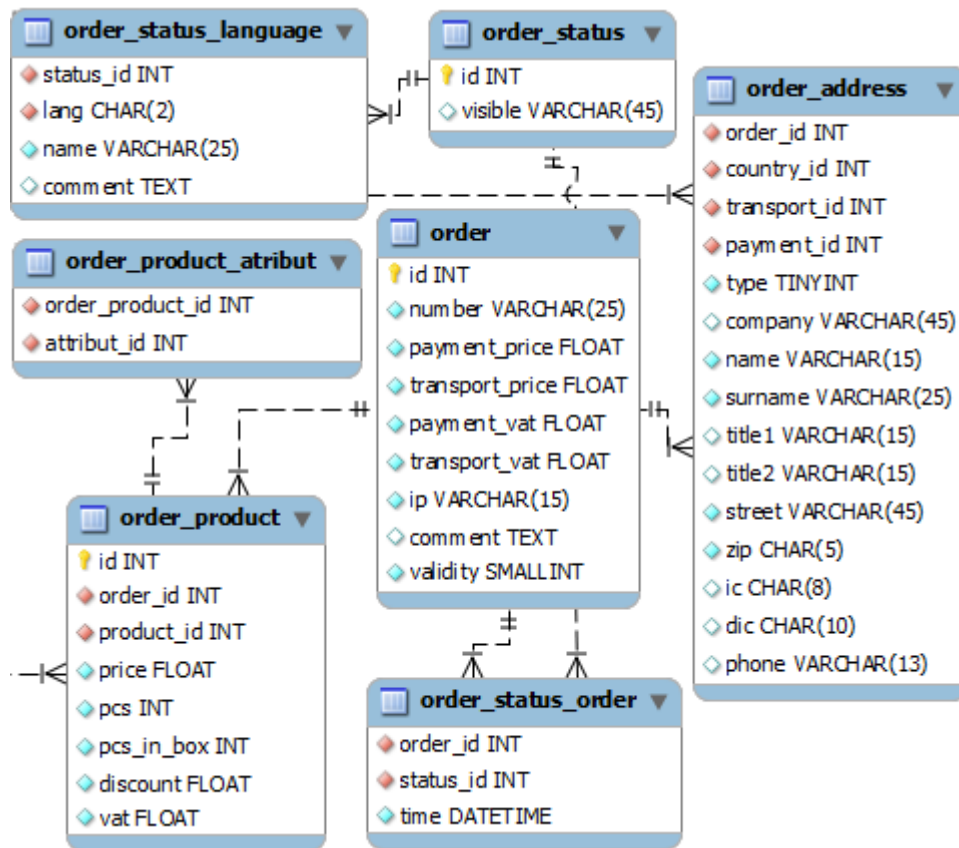
Obrázek 27: Ukázka návrhu datového skladu pro správu produktů

Výše uvedený obrázek nezachycuje všechny tabulky související s tabulkou produktů. Kompletní návrh je k nalezení v přílohách diplomové práce.

4.4.3 Návrh datového skladu pro správu objednávek

Objednávky budou ukládány do tabulky *order*. K tabulce *order* bude existovat tabulka *order_address* v relaci 1:N, do které budou ukládány fakturační a doručovací údaje. Fakturační a doručovací údaje se u objednávek budou archivovat, proto budou ukládány

na dvou místech. Je to z toho důvodu, že si každý uživatel může kdykoliv tyto údaje změnit a tím pádem by došlo i ke změnám na již vyřízených objednávkách. Stejným způsobem jsou řešeny produkty v objednávkách v tabulce *order_product*, kde je ukládána i cena a DPH v době vzniku objednávky.

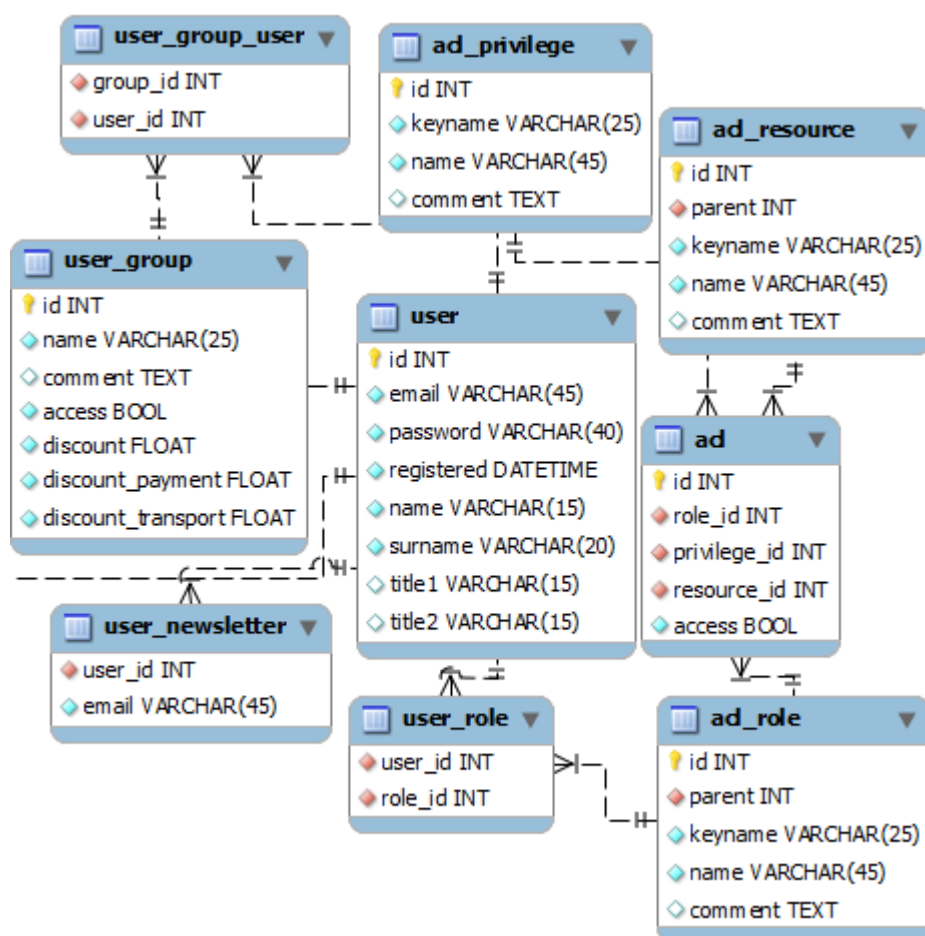


Obrázek 28: Návrhu datového skladu pro správu objednávek

Prostřednictvím tabulky *order_status_order* budou zaznamenávány přechody mezi jednotlivými statusy. U každé změny bude zaznamenáváno i datum provedení.

4.4.4 Návrh datového skladu pro správu zákazníků a uživatelských rolí

Tabulky s předponou *acl* slouží k definování zdrojů, oprávnění a rolí. Jednotlivé role a zdroje mohou tvořit hierarchii, proto jsou definovány cizí klíče *parent*, které odkazují na primární klíč v dané tabulce.



Obrázek 29: Návrhu datového skladu pro správu uživatelů

K tabulce *user* ještě existuje tabulka *user_address* v relaci 1:N, do které se ukládají kontaktní, fakturační a další údaje – viz celý návrh v přílohách diplomové práce.

Na všech výše uvedených obrázcích týkajících se návrhu datového skladu nemusí být kompletní údaje. Z důvodu přehlednosti a možnosti lepší demonstrace byly některé tabulky v návrhu zjednodušeny, nebo vynechány. Kompletní návrh datové struktury pro B2B, B2C aplikaci i spolu s integritním omezením je k nalezení v přílohách diplomové práce.

4.5 Systémy pro správu verzí (VCS)

Na vývoj, údržbu a rozšiřování B2B, B2C aplikace budou nasazeny nové nástroje,

jejichž primárním účelem bude tyto procesy zefektivnit. Většina těchto nástrojů bude poté postupně nasazena i na další firemní projekty.

Krom využívání frameworku nastane zásadní změna i co se týká správy verzí jednotlivých aplikací. V současné chvíli je stav neudržitelný a není možno dohledat nejaktuálnější verzi, nebo verzi distribuovanou jednotlivým zákazníkům. Na různých místech je uloženo velké množství archivovaných projektů, ve kterých se lze již stěží vyznat. A některé podstatnější změny nejsou kolikrát archivovány vůbec.

Druhým a zásadnějším problémem současného stavu je vysoká komplikovanost spolupráce více programátorů na jednom projektu, která bude s využíváním systému pro správu verzím minimalizována.

Je třeba zavést automatizovanou správu verzí, která bude zároveň zaznamenávat veškeré prováděné změny, autora těchto změn a komentáře k těmto změnám. Kdykoliv v budoucnosti pak nebude problémem dohledat jakoukoliv verzi.

Pro každého klienta bude vytvořena speciální vedlejší větev, kde se budou realizovat individuální úpravy. S pomocí VCS nebude problémem tyto verze jednoduše slučovat, porovnávat, případně se navrátit k poslední 100% funkční variantě.

Ve stávající aplikaci bylo zavedeno číslování jednotlivých verzí, nicméně toto číslování měl na starosti programátor, který ho prováděl manuálně. S každou novou větší změnou se měly vytvářet archivy. Bohužel ani jedna z těchto činností takto většinou neřešila.

Úkolem této kapitoly je analyzovat a následně zvolit vhodný systém pro správu verzí, který se bude ve firmě využívat. Po průzkumu dostupných řešení byly do výběru zařazeny následující tři systémy.

4.5.1 SVN neboli Subversion

Subversion je příkladem centralizovaného systému pro správu verzí. Ve společnosti XYZ, s.r.o. byl tento systém použit na tři poslední projekty a relativně se osvědčil. Poměrně razantně se zvýšila rychlost práce, jelikož bylo eliminováno několik mezikroků k získání aktuálních verzí aplikace a navíc přinesl pohodlnou práci více programátorů.

Výhody SVN

- ✓ dostupnost repozitářů, snadná instalace
- ✓ výborná integrace do vývojového prostředí (Eclipse)
- ✓ multiplatformní systém
- ✓ velmi jednoduchý, aktuální zkušenosti

Nevýhody SVN

- ✓ centralizovaný systém
- ✓ nízká rychlost
- ✓ nemožnost pracovat offline
- ✓ jasně daný typ workflow
- ✓ nemožnost verzovat lokální úpravy

4.5.2 Git

Git je dnes velmi populárním typem distribuovaného systému pro správu verzí. Každému vývojáři poskytuje lokální kopii celé historie vývoje. Vývojářům nabízí mnohem více možností než Subversion.

Výhody Gitu

- ✓ distribuovaný systém, dokáže verzovat i lokální obsah
- ✓ vysoká rychlost
- ✓ možnost práce offline
- ✓ umožňuje importovat i exportovat SVN včetně historie
- ✓ umožňuje zásah do historie (např. mazat slepé větve apod.)
- ✓ nástroje jako TortoiseGit jsou ve windos integrovány přímo do exploreru
- ✓ libovolný typ workflow

Nevýhody Gitu

- ✓ primárně určen pro prostředí OS Linux
- ✓ nedotaženost doplňků do vývojových prostředí (např. Eclipse)

4.5.3 Mercurial

Mercurial je stejně jako předchozí Git typem distribuovaného systému pro správu revizí. Oproti Gitu není tolik rozšířený a nabízí o něco méně funkcí.

Výhody systému Mercurial

- ✓ jednoduchý a menší systém
- ✓ výborná funkčnost na různých platformách
- ✓ možnost práce offline
- ✓ zvládá import SVN včetně historie
- ✓ libovolný typ workflow

Nevýhody systému Mercurial

- ✓ nelze zasahovat do historie
- ✓ nedotaženost doplňků do vývojových prostředí (např. Eclipse)

Nakonec byl pro vývoj aplikace a pravděpodobně i pro další firemní projekty vybrán systém Git. Množstvím svých funkcí se jeví jako nejsložitější systém, ale na druhou stranu nabízí největší svobodu při správě jednotlivých verzí. Zvolen byl především kvůli možnostem lokálního verzování, offline práci, rychlosti a možnosti volby libovolného typu workflow u jednotlivých projektů. Nasazení tohoto řešení také podpořily následující internetové stránky.

- ✓ <http://whygitisbetterthanx.gitfu.cz>
- ✓ <http://phpfashion.com/proc-opustit-subversion-svn>

4.6 Závazná pravidla pro vývoj aplikace

Jelikož se jedná o poměrně rozsáhlou aplikaci, na jejímž vývoji se bude podílet několikačlenný tým lidí, je třeba stanovit závazná pravidla, která bude nutné při vývoji a následné údržbě a rozšiřování aplikace platit. Cílem těchto pravidel je zpřehlednit a usnadnit orientaci ve zdrojových kódech a jednotlivých verzích aplikace.

4.6.1 Zdrojové kódy

Všechny zdrojové kódy a jejich komentáře budou v anglickém jazyce. Žádné výjimky nejsou přípustné. Každá proměnná bude mít v komentáři definovaný datový typ, pokud se bude jednat o funkci, bude definována i její návratová hodnota.

Názvy funkcí, názvy proměnných budou začínat malými písmeny a bude se využívat tzv. velbloudí notace. Např. pro načtení produktů podle jejich id se bude funkce jmenovat *getProductById(\$id)* apod. Názvy tříd budou začínat velkými písmeny a dále budou využívat taktéž velbloudí notaci. Samotný název funkce, třídy, nebo proměnné musí být vždy co nejvýstižnější a je třeba se vyvarovat jednopísmenným názvům, které jsou nicneřikající.

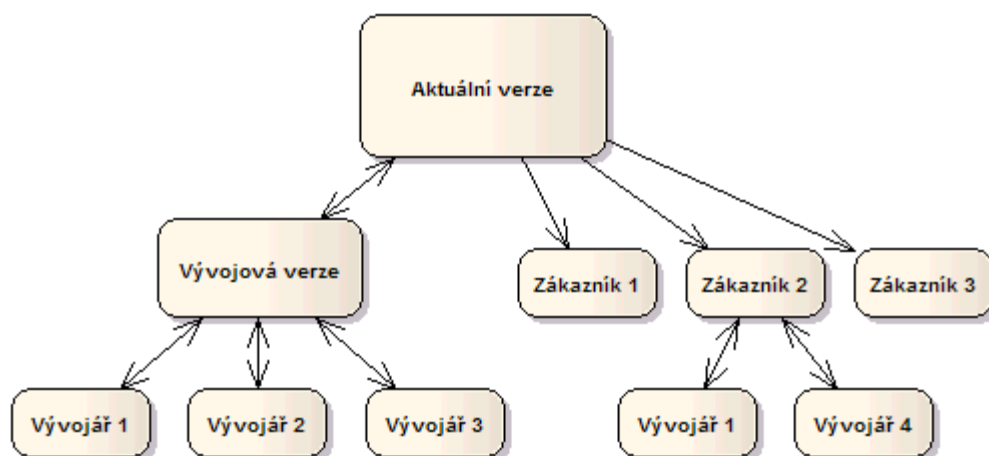
Ve zdrojových kódech se bude využívat stejné odsazení a to následujícím způsobem:

- ✓ v souborech typu html, css - 2 space tab stops
- ✓ v souborech typu php, js, sql a dalších - 1 tab stop

Všechny složitější funkce, postupy či mechanismy budou řádně okomentované tak, aby bylo na první pohled jasné, k čemu daná část kódu slouží, případně na čem je závislá.

4.6.2 Správa verzí

Závazná pravidla budou platit i pro správu jednotlivých verzí, aby se zabránilo stavu, do kterého se dostala současná aplikace. Jednotlivé verze budou spravovány následujícím způsobem:



Obrázek 30: Správa verzí - workflow

Aktuální verze bude otestovaná verze připravená k nasazení u zákazníka. Do této verze se budou commitovat vývojové verze, které budou ozkoušené a schválené project managerem. Každý vývojář pak bude pracovat s vývojovou verzí, do které může commitovat změny. Vývojáři mohou mezi sebou v případě potřeby sdílet své vlastní repozitáře.

Pro každého nového zákazníka dojde k vytvoření vlastní větve z aktuální verze. Dílčí změny a úpravy týkající se daného zákazníka tak budou prováděny jen v jeho repozitáři. Do repozitáře zákazníka budou mít přístup programátoři, kteří mají tento projekt na starosti. V případě potřeby bude možné změny či rozšíření aplikace provedené v repozitáři zákazníka převést i do vývojové verze a následně do aktuální verze aplikace.

4.7 Ekonomické zhodnocení a přínosy nového řešení

Jak již bylo zmiňováno v analýze stávajícího řešení náklady na údržbu a implementaci stávající internetové aplikace u zákazníka byly již neúměrně vysoké. Ve zdrojových kódech se vyskytuje poměrně velká duplicita, různé části kódu jsou závislé na jiných částech, velké množství stejných úprav je realizováno stále dokola kvůli neexistenci fungující správy verzí, minimální podpora současných požadavků zákazníků apod.

Některé zakázky, u kterých zákazníci vyžadovali rozšířené funkce jako je např. plná podpora AJAXu, multijazyčnost, správa uživatelských rolí a další byly natolik časově a

finančně náročné, že je nebylo možné stávající aplikací v požadované míře efektivnosti a výnosnosti uspokojit.

4.7.1 Ekonomické vyjádření a harmonogram realizace

Analýza, návrh a implementace nové internetové B2B, B2C aplikace s sebou nově přinese relativně velké náklady. Ovšem tím, že se jedná distribuovanou aplikaci, která bude dále prodávána, bude se podíl nákladů na jednu instalaci postupem času snižovat.

Do celkových nákladů vstupují nejen variabilní náklady, kde je uvažováno s průměrnou hodinovou sazbou 300,- Kč u analytika a u vývojářů, u testerů je uvažována poloviční hodinová sazba, ale i fixní náklady v podobě energie, nájmu, telekomunikačních nákladů, odpisů, části ceny používaného software a další. Fixní náklady byly stanoveny na přibližně 10 000,- Kč.

Do níže sestaveného předběžného rozpočtu byly započítány i prvotní náklady na instalaci a konfiguraci systému pro správu verzí a licence WYSIWYG editorů. Je ale velmi pravděpodobné, že tyto technologie budou nově využívány u většiny dalších projektů. Stejným způsobem budou využívány některé části aplikace, proto se dá očekávat, že by mohly být skutečné náklady nižší.

Celková cena návrhu a implementace internetového obchodu i se započítanými fixními náklady je přibližně 195 250,- Kč. Časová náročnost realizace vychází přibližně na 640 hodin s tím, že na vývoji bude po většinu času pracovat tým o třech vývojářích. Jelikož tito vývojáři mají na starosti i běžné projekty, uvažuje se při vývoji s jejich průměrnou dobou strávenou nad projektem na přibližně 4 hodiny denně. Vývoj aplikace nesmí zásadním způsobem ovlivnit, nebo opozdit vývoj stávajících projektů.

Celý projekt byl zahájen 29.3. 2010 a dle zpracovaných návrhů by měl být dokončen přibližně 9. 8. 2010. Jednotlivé procesy, jejich časy zahájení a ukončení spolu s přidělením potřebných pracovníků jsou zachyceny pomocí Ganttova diagramu, který je k nalezení v přílohách diplomové práce.

V následující tabulce jsou uvedeny jednotlivé činnosti, jejich odhadovaná hodinová časová a finanční náročnost a také celková finanční náročnost.

| Popis operace | Čas [hod] | Kč/hod | Celkové náklady |
|---------------------------------|------------------|---------------|------------------------|
| Specifikace požadavků | 20 | 300,- Kč | 6 000,- Kč |
| Analýza a výběr technologií | 16 | 300,- Kč | 4 800,- Kč |
| Náklady na pořízení technologie | | | |
| • systém pro správu verzí | - | - | 8 000,- Kč |
| • licence WYSIWYG editorů | - | - | 28 000,- Kč |
| Návrh aplikace | | | |
| • návrh jednotlivých částí | 40 | 300,- Kč | 12 000,- Kč |
| • návrh rozhraní | 16 | 300,- Kč | 4 800,- Kč |
| • návrh datového skladu | 24 | 300,- Kč | 7 200,- Kč |
| Interní proškolení zaměstnanců | | | |
| • framework, standardy | 3 | 300,- Kč | 900,- Kč |
| • správa verzí, vývojové studio | 4 | 300,- Kč | 1 200,- Kč |
| Implementace aplikace | | | |
| • datový sklad, relace | 8 | 300,- Kč | 2 400,- Kč |
| • administrační rozhraní | 220 | 300,- Kč | 66 000,- Kč |
| • veřejné rozhraní | 160 | 300,- Kč | 48 000,- Kč |
| • optimalizační procesy | 40 | 300,- Kč | 12 000,- Kč |
| • sestavení demo aplikace | 24 | 300,- Kč | 7 200,- Kč |
| Testování | | | |
| • administrační rozhraní | 25 | 150,- Kč | 3 750,- Kč |
| • veřejné rozhraní | 20 | 150,- Kč | 3 000,- Kč |
| Doladění aplikace | 20 | 300,- Kč | 6 000,- Kč |
| Fixní náklady | - | - | 10 000,- Kč |
| Celkem | 640 | | 195 250,- Kč |

Tabulka 1: Předpokládaný rozpočet B2B, B2C aplikace

Návratnost investice, která bude vynaložena na návrh a implementaci nové aplikace bude záviset především na množství prodaných aplikací. Budeme-li uvažovat současné měsíční náklady firmy a stávající trend, kdy prostřednictvím obchodních partnerů, kteří stávající aplikaci distribuují, prodáme přibližně 7 aplikací za rok a sama společnost prodá kolem 14 aplikací za rok, je předpokládána návratnost v horizontu 5 - 7 měsíců.

Prostřednictvím obchodních partnerů se většinou prodává přímo krabicové řešení, u kterého je třeba jen poměrně malé množství modifikací. Průměrná cena těchto aplikací se pohybuje okolo 39 000,- Kč. Z toho je 30% jako provize pro obchodního partnera.

Z množství prodaného přímo společností XYZ je přibližně 3 - 5 aplikací poměrně individuálních s velkým množstvím úprav a rozšíření, které do konečné ceny velmi intenzivně promlouvají. Průměrná cena těchto aplikací se pohybuje kolem 68 000,- Kč.

A právě nepořádek v jednotlivých verzích spolu s množstvím individuálních úprav, které se díky nevhodnému návrhu aplikace a zastaralým technologiím původní aplikace velmi složitě realizovali tak výslednou aplikaci interně výrazně prodražovali. Tento negativní výsledek by měla nová verze aplikace výrazně zlepšit.

Společnost XYZ, s.r.o. chce i nadále zachovat přibližně stejnou cenovou politiku a zákazníkovi budou nabízeny vždy jen ty části aplikace, které požaduje. Díky novému návrhu a použitým technologiím při vývoji a správě bude možno uspokojit i vyšší požadavky zákazníků při nižší časové náročnosti a tedy i nižších nákladech.

4.7.2 Přínosy nového řešení

Největší přínosy se očekávají od využití novějších technologií spolu s nasazením frameworku, které velmi usnadní implementační část, ale i dílčí modifikace a jiná rozšíření u jednotlivých zákazníků. Tato pokročilá technologie společnosti také umožní snížit časovou náročnost těchto procesů. Framework sám o sobě díky svým vlastnostem eliminuje bezpečnostní rizika, podporuje AJAX, DRY, KISS, MVC a znovupoužitelnost kódu. Zákazník pak dostane vysoce interaktivní aplikaci s přehledným uživatelským rozhraním, která splňuje všechny dnes kladené nároky na tento typ aplikací tohoto rozsahu.

Tím, že bude připravena plně multijazyčná aplikace s možností zavedení libovolného počtu jazykových mutací bude možno efektivněji uspokojit i zákazníky s těmito požadavky. Za použití stávající aplikace byla implementace tohoto řešení pro zákazníka, ale i pro společnost poměrně nákladná a neefektivní.

Při návrhu datového skladu bylo uvažováno i s maximální podporou optimalizace pro

vyhledávače – tzv. SEO. V kombinaci s dalšími SEO a SEM nástroji bude aplikace podporovat velmi rozsáhlé možnosti pro komplexní optimalizaci pro vyhledávače.

Nemalé přínosy plynou také z využití systému pro správu verzí, který nejen že usnadní koordinaci práce jednotlivých vývojářů, ale také zamezí vzniku zmatků v jednotlivých verzích aplikace. VCS spolu se zavedením závazných pravidel pro vývoj aplikace by měli celkově zvýšit produktivitu práce.

V konečném důsledku by všechny tyto přínosy měly přinést celkově nižší náklady na nasazení aplikace u zákazníka s tím, že samotnému zákazníkovi přinesou také vyšší užitek. Přesné číselné vyjádření těchto přínosů je velmi obtížné, jelikož se budou odvíjet od poptávky po aplikaci, ale také rozsahu požadovaných modifikací a následného servisu u daného zákazníka.

4.8 Strategie zavedení aplikace do praxe

Po úspěšném dokončení implementace aplikace dojde k vyřazení stávající aplikace. Ta se bude nadále udržovat již pouze u stávajících zákazníků a nebude nadále rozšiřována. Z nové aplikace budou vytvořeny „krabicové“ verze stejným způsobem jako u stávající aplikace. U jednotlivých „krabicových“ verzí dojde k rozšíření o nové vlastnosti. Některé funkce budou zařazeny do všech verzí, jiné bude možné dokupovat za příplatky jako nadstandardní moduly. K tomuto účelu bude upraven a rozšířen stávající konfigurační systém aplikace, který je dostupný na internetových stránkách společnosti.

Prodej aplikace bude probíhat stejným způsobem jako doposud – tedy prostřednictvím externích distributorů, kteří budou mít z prodeje aplikace předem dohodnutý podíl a pak přímo prostřednictvím společnosti XYZ, s.r.o.

Propagaci novinek, vlastností a pokročilých možností nové aplikace bude napomáhat vytvořená demo aplikace, která bude naplněna testovacími produkty. Potencionální zákazník si bude moci tuto aplikaci před zakoupením řádně vyzkoušet. Se zahájením distribuce nové aplikace budou s největší pravděpodobností spojeny i další marketingové akce, tak jako tomu je u stávající aplikace.

Závěr

Internetové podnikání a především internetové obchodování je dnes velmi perspektivní obor, který stále zaznamenává poměrně velký růst. Tento fakt dokládá i pohled na tuzemský internetový trh, kde vedle sebe existují velká nákupní internetová centra, ale také drobné a úzce zaměřené internetové obchody a další neustále vznikají. Se vzrůstajícím množstvím internetových obchodů roste i konkurence a dosažení úspěchu tak nemá zaručen žádný z nich. Zákazníci se dnes řídí především nízkou cenou, kvalitou servisu, referencemi a zkušenostmi, ale také designem, přehledností a jednoduchostí samotné aplikace. A právě design, přehlednost a jednoduchost jsou vlastnosti, které bude nová aplikace, která vzejde z této práce nabízet. Spolu s kvalitním servisem a zákaznickou podporou bude podnikatelským jednotkám nabízeno komplexní řešení pro internetové obchodování.

Diplomová práce byla zpracována jako projekt, na který byly aplikovány jednotlivé prvky projektového managementu. A právě analýza a návrh patří z pohledu projektového managementu mezi nejdůležitější procesy, při kterých dochází k přesným specifikacím jednotlivých částí a funkcí aplikace.

V první části diplomové práce jsou definovány nezbytné teoretické podklady a po nich již následuje rozbor a podrobná analýza současného stavu stávající aplikace. Na základě nevyhovujícího současného stavu se přistoupilo k návrhu a realizaci nové B2B, B2C aplikace, která bude nově splňovat současné i budoucí požadavky jak ze strany zákazníka, tak i ze strany distribuujících společností.

Nově navržená aplikace počítá s plným využitím moderních technologií jako jsou např. AJAX, který bude zajišťovat interaktivní uživatelské prostředí, framework, který usnadní a zefektivní implementaci a následnou údržbu a rozšiřitelnost aplikace a zavádí velké množství bezpečnostních prvků. Velmi významnou inovací je také kompletní podpora multijazyčných verzí aplikace.

Nově navržen byl i systém pro správu verzí, který má zajistit snadnou koordinaci jednotlivých vývojářů na implementacích jednotlivých verzí aplikace, ale také má zvýšit přehlednost jednotlivých verzí a minimalizovat čas vynaložený na hlídání jednotlivých

verzí aplikace. Definovány byly i závazná pravidla pro vývoj a údržbu aplikace, které mají usnadnit a zvýšit efektivitu práce a zvýšit přehlednost zdrojových kódů jednotlivých částí aplikace.

V poslední části práce byly stanoveny jak náklady, tak časová náročnost jednotlivých kroků. Sestaven byl i časový harmonogram realizace projektu. Po dokončení implementace bude aplikace připravena k distribuci k jednotlivým zákazníkům.

Diplomová práce obohatila mé dosavadní znalosti a zkušenosti v této oblasti. Při zpracovávání této práce jsem zúročil některé znalosti získané dosavadním studiem. V průběhu realizace diplomové práce se vyskytly i problémy, kterým se díky novým zkušenostem budu moci v budoucnu vyhnout, případně je eliminovat dříve než nastanou.

Seznam použité literatury

Monografické zdroje

- [1] BLAŽKOVÁ, M. *Jak využít internet v marketingu :krok za krokem k vyšší konkurenceschopnosti*. 1. vyd. Praha : Grada, 2005. 156 s. : il. ISBN 80-247-1095-1.
- [2] DENNIS, L. *Project Management*. 9th rev. edition. Aldershot : Gower Publishing Ltd., 2007. 520 s. ISBN 978-0-566-08772-1.
- [3] DVOŘÁK, J. *Elektronický obchod*. Brno : Vysoké učení technické v Brně, 2004. ISBN 80-214-2600-4.
- [4] FORREST, L. *Pro Zend Framework CMS : Building a full CMS using Advanced Aspects of the Zend Framework*. New York City : Apress, 2009. 240 s. ISBN 978-1-4302-1879-1.
- [5] HLAVENKA, J. *Dělejte byznys na Internetu :jak využít Internet k prospěchu firmy i jednotlivce*. Vyd. 1. Praha : Computer Press, 2001. vii, 226 s. ISBN 80-7226-371-4.
- [6] HLAVENKA, J. *Internetový marketing :Praktické rady, tipy, návody a postupy pro využití Internetu v marketingu*. 1.vyd. Praha : Computer Press, 2001. 157 s. ISBN 80-7226-498-2.
- [7] NĚMEC, V. *Projektový management*. Grada, 2002. 184 s. ISBN 80-247-0392-0.
- [8] NOVÁK, J. *Návrh a optimalizace internetového obchodu*. (Bakalářská práce) Brno: VUT, 2008. 51 s.
- [9] SMIČKA, R. *Optimalizace pro vyhledávače - SEO :jak zvýšit návštěvnost webu*. Vyd. 1. Kralice na Hané : Zásilkové knihkupectví J. Smičkové, 2004. 126 s. ISBN 80-239-2961-5.
- [10] POUR, J., GÁLA, L., ŠEDIVÁ, Z. *Podniková informatika*. 2. přepracované a aktualizované vydání. Praha: Grada, 2009. 496 s. ISBN 978-80-247-2615-1.
- [11] QUENTIN, Z. *Practical Web 2.0 Applications with PHP*. New York : Springer, 2007. 570 s. ISBN 1590599063.
- [12] STUHLÍK, P. *Marketing na Internetu*. 1.vyd. Praha : Grada Publishing, 2000. 247 s. ISBN 80-7169-957-8.

Elektronické zdroje

- [13] BERNARD, B. *Úvod do architektury MVC*. [online]. 2009 [cit. 2010-03-16]. Dostupné z <<http://zdrojak.root.cz/clanky/uvod-do-architektury-mvc/>>.
- [14] DANĚK, P. *Velký test PHP Frameworků*. [online]. 2008 [cit. 2010-04-20]. Dostupné z <<http://www.root.cz/clanky/velky-test-php-frameworku-2008/>>.
- [15] KALID, A. *Intro to Distributed Version Control (Illustrated)*. [online]. 2007 [cit. 2010-03-27]. Dostupné z <<http://betterexplained.com/articles/intro-to-distributed-version-control-illustrated/>>.
- [16] kolektiv dobrovolných přispěvatelů. *Model-view-controller*. [online]. 2001 – 2010 [cit. 2010-03-16]. Dostupné z <<http://cs.wikipedia.org/wiki/Model-view-controller>>.
- [17] kolektiv dobrovolných přispěvatelů. *Programming_style*. [online]. 2001 – 2010 [cit. 2010-04-05]. Dostupné z <http://en.wikipedia.org/wiki/Programming_style>.
- [18] kolektiv dobrovolných přispěvatelů. *SMART*. [online]. 2008 [cit. 2010-04-05]. Dostupné z <<http://www.gewiki.cz/SMART>>.
- [19] kolektiv dobrovolných přispěvatelů. *Verzování*. [online]. 2001 – 2010 [cit. 2010-03-25]. Dostupné z <<http://cs.wikipedia.org/wiki/Verzování>>.
- [20] kolektiv dobrovolných přispěvatelů. *Web application framework*. [online]. 2001 – 2010 [cit. 2010-03-22]. Dostupné z <http://en.wikipedia.org/wiki/Web_application_framework>.
- [21] kolektiv dobrovolných přispěvatelů. *Hlavní přednosti | Nette Framework*. [online] 2010 [cit. 2010-03-22]. Dostupné z <<http://nettephp.com/>>.
- [22] LORENC, M. *Ganttův diagram*. [online] 2007 – 2010 [cit. 2010-03-28]. Dostupné z <<http://lorenc.info/3MA381/graf-ganttuv-diagram.htm>>.
- [23] *Marketingové noviny - Historie elektronických obchodů*. [online]. 2006 [cit. 2010-03-23]. Dostupné z <http://www.marketingovenoviny.cz/index.php3?Action=View&ARTICLE_ID=4391>.
- [24] MATĚNA, R. *Programátorské zkratky aneb principy programátora ve zkratkách*

- DRY, KISS, YAGNI a další.* [online]. 2009 [cit 2010-04-03]. Dostupné z <<http://www.webfaq.cz/clanek/Programatorske-zkratky-aneb-principy-programatora-ve-zkratkach-DRY-KISS-YAGNI-a-dalsi>>.
- [25] MINAŘÍK, K. *Proč je Git lepší než X?*. [online]. 2009 [cit. 2010-03-27]. Dostupné z <<http://whygitisbetterthanx.gitfu.cz/>>.
- [26] PHP Framework benchmark: Zend, CodeIgniter & CakePHP. [online]. 2009 [cit. 2010-04-20]. Dostupné z <<http://leftblank.nl/php-framework-benchmark-zend-codeigniter-cakephp-481.html>>.
- [27] STŘELEČ, J. *SWOT analýza.* [online]. 2006 - 2009 [cit. 2010-03-14]. Dostupné z <<http://www.vlastnicesta.cz/akademie/marketing/marketing-metody/swot-analyza/>>.
- [28] TICHÝ, J. *Architektura aplikace.* [online]. 2004 – 2010 [cit. 2010-03-16]. Dostupné z <<http://www.jantichy.cz/diplomka/pozadavky/architektura>>.
- [29] *Top 10 PHP Frameworks.* [online]. 2010 [cit. 2010-04-20]. Dostupné z <<http://www.phpframeworks.com/news/p/category/top-10-phpframeworks>>.
- [30] VAVŘIČKA, J. *Internetové obchody budoucnosti.* [online]. 2008 [cit. 2010-04-05]. Dostupné z <<http://www.lupa.cz/clanky/internetove-obchody-budoucnosti/>>.

Seznam obrázků






| | |
|---|----|
| Obrázek 1: Architektura MVC (Model-View-Controller)..... | 21 |
| Obrázek 2: Schéma fungování centralizovaného verzovacího systému..... | 26 |
| Obrázek 3: Schéma fungování distribuovaného verzovacího systému..... | 27 |
| Obrázek 4: Srozumitelně napsaný zdrojový kód..... | 28 |
| Obrázek 5: Nepřehledně napsaný zdrojový kód..... | 28 |
| Obrázek 6: Grafická podoba krabicových verzí stávajícího řešení aplikace..... | 35 |
| Obrázek 7: Typy uživatelů veřejné části aplikace..... | 43 |
| Obrázek 8: Use Case diagram katalogu veřejné části aplikace..... | 44 |
| Obrázek 9: Use Case diagram nákupního košíku..... | 47 |
| Obrázek 10: Use Case diagram uživatelských účtů..... | 48 |
| Obrázek 11: Typy uživatelů administrační části aplikace..... | 50 |
| Obrázek 12: Stromová struktura katalogu..... | 51 |
| Obrázek 13: Ukázka návrhu grafického rozhraní v administrační části aplikace..... | 67 |
| Obrázek 14: GUI administrační části - systém kartiček..... | 68 |
| Obrázek 15: Zápis cyklu foreach v šablonách CakePHP..... | 70 |
| Obrázek 16: Zápis cyklu foreach pomocí šablon v Nette Frameworku..... | 71 |
| Obrázek 17: Zápis cyklu foreach v šablonách Zend Frameworku..... | 73 |
| Obrázek 18: Ukázka komponenty datagrid pro Nette od Romana Sklenáře..... | 75 |
| Obrázek 19: Základní konfigurace datagridu..... | 75 |
| Obrázek 20: Vykreslení datagridu v šabloně..... | 76 |
| Obrázek 21: Nástin řešení překladů pomocí asociativního pole..... | 77 |
| Obrázek 22: Příklad registrace GetText translátoru v BasePresenteru..... | 78 |
| Obrázek 23: Příklad zápisu textu v šabloně určeného k překladu..... | 78 |
| Obrázek 24: Doplnění překladu pomocí aplikace Poedit..... | 79 |
| Obrázek 25: Ukázka definice routeru a překladového slovníku url adres..... | 80 |
| Obrázek 26: Dekompozice multijazyčných dat u katalogu..... | 82 |
| Obrázek 27: Ukázka návrhu datového skladu pro správu produktů..... | 83 |
| Obrázek 28: Návrhu datového skladu pro správu objednávek..... | 84 |
| Obrázek 29: Návrhu datového skladu pro správu uživatelů..... | 85 |
| Obrázek 30: Správa verzí - workflow..... | 90 |

Seznam příloh

Příloha č. 1: Názvy úkolů + Ganttův diagram

Příloha č. 2: CD-ROM - Návrh datového skladu aplikace (grafika)

Příloha č. 1: Názvy úkolů + Ganttův diagram

| |  Název úkolu | Doba trvání | Zahájení | Dokončení | Názvy zdrojů |
|----|---|------------------|-----------------|-----------------|----------------------------|
| 1 | | | | | |
| 2 | | | | | |
| 3 | Zahájení projektu | 0 hodin | 29.3. 10 | 29.3. 10 | |
| 4 | Specifikace požadavků | 20 hodin | 29.3. 10 | 2.4. 10 | Analytik |
| 5 | Analýza a výběr technologií | 16 hodin | 5.4. 10 | 8.4. 10 | Analytik |
| 6 | Instalace a konfigurace RCS | 40 hodin | 9.4. 10 | 22.4. 10 | Administrátor |
| 7 | Nákup licencí editorů | 14 dny | 9.4. 10 | 28.4. 10 | Analytik |
| 8 |  Návrh aplikace | 22 dny | 9.4. 10 | 10.5. 10 | |
| 9 | Návrh jednotlivých částí aplikace | 40 hodin | 9.4. 10 | 22.4. 10 | Analytik |
| 10 | Návrh rozhraní | 16 hodin | 23.4. 10 | 28.4. 10 | Analytik |
| 11 | Návrh datového skladu | 32 hodin | 29.4. 10 | 10.5. 10 | Analytik[50%];Vývojář[50%] |
| 12 |  Interní proškolení zaměstnanců | 1,5 dny | 11.5. 10 | 12.5. 10 | |
| 13 | framework, standardy | 4 hodin | 11.5. 10 | 11.5. 10 | Vývojář[50%];Analytik[50%] |
| 14 | správa verzí, vývojové studio | 2 hodin | 12.5. 10 | 12.5. 10 | Analytik[50%];Vývojář[50%] |
| 15 |  Implementace aplikace | 51,33 dny | 12.5. 10 | 22.7. 10 | |
| 16 | datový sklad, relace | 8 hodin | 12.5. 10 | 14.5. 10 | Vývojář |
| 17 | administrační rozhraní | 80 hodin | 14.5. 10 | 11.6. 10 | Vývojář[300%] |
| 18 | veřejné rozhraní | 53,33 hodin | 11.6. 10 | 30.6. 10 | Vývojář[300%] |
| 19 | optimalizační procesy | 40 hodin | 30.6. 10 | 14.7. 10 | Vývojář |
| 20 | sestavení demo aplikace | 24 hodin | 14.7. 10 | 22.7. 10 | Vývojář |
| 21 |  Testování | 6,25 dny | 22.7. 10 | 2.8. 10 | |
| 22 | administrační rozhraní | 25 hodin | 22.7. 10 | 2.8. 10 | Tester |
| 23 | veřejné rozhraní | 20 hodin | 22.7. 10 | 29.7. 10 | Tester |
| 24 | Doladění aplikace | 20 hodin | 2.8. 10 | 9.8. 10 | Vývojář |
| 25 | Připravená verze aplikace | 0 hodin | 9.8. 10 | 9.8. 10 | |

