VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM FIRMY CYKLOSPORT PAVKA

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE JIŘÍ KOTULAN
AUTHOR

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM FIRMY CYKLOSPORT PAVKA

INFORMATION SYSTEM OF THE FIRM CYKLOSPORT PAVKA

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

JIŘÍ KOTULAN

VEDOUCÍ PRÁCE
SUPERVISOR

ING. LUKÁŠ STRYKA

BRNO 2008

Abstrakt

Tato práce se zabývá tvorbou modulárního informačního systému na základě požadavků zadavatelské firmy. Systém je implementován pomocí jazyka Delphi 2007 a PHP5, postavených nad databázemi Interbase 2007 a MySQL 5. Cílem práce je vytvořit funkční prototyp informačního systému, skládajícího se z modulu pro řízení skladu, modulu pro řízení práce servisní dílny a modulu pro internetový obchod. Návrh tohoto systému je popsán pomocí prostředků modelovacího jazyka UML. V hlavní části práci je pak rozebírána implementace jednotlivých modulů.

Klíčová slova

Informační systém, databáze, internetový obchod, řízení skladu, řízení práce servisu, Delphi, PHP, JavaScript, CSS, MySQL.

Abstract

This thesis is concerned with creation of modular information system on the basis of submitter firm's requirements. The system is implemented by using Delphi 2007 and PHP5, built on databases Interbase 2007 and MySQL 5. It's aim is create functional prototype of information system, consisting of program unit for inventory control, program unit for managing service shop and program unit for e-shop. Design of this system is described using UML language. Core of the thesis is formed of implementation analysis of each single program unit.

Keywords

Information system, database, e-shop, inventory control, managing service shop, Delphi, PHP, JavaScript, CSS, MySQL.

Citace

Kotulan Jiří: Informační systém firmy Cyklosport Pavka. Brno, 2008, bakalářská práce, FIT VUT v Brně.

Informační systém firmy Cyklosport Pavka

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Lukáše Stryky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Kotulan
19.5.2008

Poděkování

Tímto bych rád vyjádřil svůj velký dík Ing. Lukáši Strykovi za jeho ochotu a trpělivost při tvorbě této práce.

Obsah

Obsah 1
1 Úvod 3
2 Cyklosport Pavka a požadavky na systém 4
2.1 Cyklosport Pavka 4
2.2 Požadavky kladené na systém 4
2.2.1 Obecné požadavky 4
2.2.2 Modul pro řízení skladu 4
2.2.3 Modul pro řízení práce servisní dílny 5
2.2.4 Modul pro internetový obchod 5
2.2.5 Uživatelská práva 6
3 Návrh 8
3.1 Use case diagram 8
3.2 ER diagram 10
4 Implementace 12
4.1 Modul pro řízení skladu a servisní dílny 12
4.1.1 DataModule1 12
4.1.2 FormLogin 13
4.1.3 FormVedouciho 13
4.1.4 FormPokladna 13
4.1.5 FormZbozi 14
4.1.6 FormDetail 15
4.1.7 FormUpravaComb 17
4.1.8 FormUpravaKategorie 17
4.1.9 FormUpravaVyrobce, FormUpravaDodavatel 17
4.1.10 FormFaktura 17
4.1.11 FormServis 20
4.1.12 FormStavyServisu 21
4.1.13 FormDetailServisu 21
4.1.14 FormObjednavky 23
4.1.15 FormStavyObjednavky 23
4.1.16 FormDetailObjednavky 23
4.2 Databázová část 24
4.2.1 Synchronizace 24
4.3 Modul pro internetový obchod 24

4.3.1 index.php 24
4.3.2 base.php 25
4.3.3 support.php 25
4.3.4 stred.php 26
4.3.5 menu.php 26
4.3.6 page.php 26
4.3.7 style.css 28
4.3.8 skript.js 29
5 Závěr 30
Literatura 31
Seznam příloh 32

1 Úvod

Význam elektronických informačních systémů neustále roste. Možností jejich uplatnění je stále více, a tak jsou informační systémy v elektronické podobě aplikovány v čím dál větším rozsahu. Firmy je používají k zjednodušení a zpřehlednění administrátorských prací, mohou je také využít jako náhradu za stávající systémy v papírové podobě.

Cílem této bakalářské práce je implementovat modulární informační systém pro firmu Cyklosport Pavka. Systém je složen z modulu pro řízení skladu, modulu pro řízení práce servisní dílny a modulu pro internetový obchod. Modul pro řízení skladu nahrazuje stávající evidenci zboží, doposud vedenou v papírové podobě. Modul pro práci servisní dílny umožňuje zákazníkům komfortnější způsob využívání služeb servisní dílny. Modul pro internetový obchod je klasická služba umožňující nákup zboží bez nutnosti fyzické návštěvy obchodu, včetně zjišťování aktuální dostupnosti zboží a podobně.

Práce je rozdělena do pěti kapitol. Kapitola první je Úvod. Ve druhé kapitole je krátce představena zadavatelská firma Cyklosport Pavka a pomocí neformální specifikace jsou uvedeny požadavky kladené na systém ze strany této firmy. Třetí kapitola podává rozbor návrhu systému použitím prostředků modelovacího jazyka UML a to pomocí diagramu užití a ER diagramu. Kapitola čtvrtá obsahuje popis samotného systému z hlediska implementace modulu pro řízení skladu, modulu pro řízení práce servisní dílny a modulu internetového obchodu. Ke každému modulu jsou uvedeny soubory se zdrojovými kódy s jejich popisem. Kapitolou pátou je závěr, ve kterém je shrnut postup a výsledky práce na tomto projektu. Jsou v něm také zmíněny další plánovaná rozšíření tohoto systému.

2 Cyklosport Pavka a požadavky na systém

V této kapitole je krátce představena zadávající firma Cyklosport Pavka, následuje bližší popis jednotlivých požadavků na tento informační systém, kladených ze strany této firmy.

2.1 Cyklosport Pavka

Firma Cyklosport Pavka je po řadu let velmi oblíbený brněnský obchod s cyklistickými potřebami. Její majitel, p. Jaroslav Pavka, je bývalým závodním cyklistou. Přednostmi této firmy, díky kterým si firma získala svou oblibu, jsou vstřícnost, ochota a osobní přístup k zákazníkovi. Firma si ovšem až doposud vystačila bez vymožeností informačních technologií, a tak bylo jen otázkou času, než firma přejde na novější technologie, zavede elektronicky řízenou evidenci skladu a zprovozní modernější podobu internetových stránek.

2.2 Požadavky kladené na systém

Pro definování neformální specifikace jsem zaměstnancům připravil dotazníkové formuláře, na jejichž základě vznikla prvotní neformální specifikace. Avšak vzhledem k tomu, že zadavatel neměl se vznikajícím typem systému žádné zkušenosti, ukazoval se původní návrh sestavený z prvotních požadavků jako nevyhovující, a proto byl postupně pozměňován a doplňován, až se neformální specifikace po určité době ustálila do následující podoby.

2.2.1 Obecné požadavky

Cyklosport Pavka potřebuje systém, který by mu umožnil správu zboží na skladě, výdej tohoto zboží včetně tisku faktur a správu zakázek servisu. Firma také požaduje zprovoznění jednoduchého internetového obchodu. Firma doposud vedla veškerou evidenci pouze „papírově“ a rozhodla se pro přejití na správu dat v elektronické podobě. Od systému očekává, že jí pomůže usnadnit a zpřehlednit administrativní práce, zpříjemní stávajícím zákazníkům kontakt s firmou a i s pomocí internetového obchodu získá nové zákazníky.

2.2.2 Modul pro řízení skladu

Nejdůležitější částí systému je modul pro řízení skladu. Jeho úkolem je evidovat veškeré potřebné informace o prodávaném zboží, jako je jeho cena, výše DPH, upřesňující popis produktu, konkrétní

dodavatel včetně obvyklé doby dodání, délka záruky, datum vložení produktu do systému, případně i velikost slevy a cena ve slevě. Systém by také měl uchovávat informaci o počtu kusů na skladě. Pojmem „zboží na skladě“ je v souvislosti s touto firmou myšleno zboží uložené přímo na prodejně, firma nedisponuje externím skladem. Jako součást tohoto modulu jsou v požadavcích specifikovány funkce umožňující výdej zboží ze skladu při prodeji, včetně procedury pro tisk faktur sloužících jako pokladní doklad. Číslování faktur by mělo být nastaveno dle data vystavení a pořadového čísla faktury toho dne. Samotné účetnictví firmy je však vedeno externími aplikacemi.

2.2.3 Modul pro řízení práce servisní dílny

Tento modul je určen pro podporu práce servisních techniků. Má tedy obsahovat funkce pro správu servisních zakázek, umožňující servisním technikům vést přehlednou evidenci jednotlivých zakázek.

Pro jednotlivé zakázky uložené v systému by měly být ukládány atributy:

- jméno servisního technika zpracovávající danou zakázku
- datum a čas přijetí zakázky
- stav zakázky
- identifikace zákazníka (nepovinné)
- předpokládaná doba opravy
- předpokládané datum dokončení opravy
- datum a čas dokončení zakázky
- seznam servisních prací prováděných na zakázce

Tomuto modulu by také mělo být umožněno propojení s modulem internetového obchodu, dovolující zákazníkům pohodlně přes internet zjišťovat stav zpracování svých zakázek.

2.2.4 Modul pro internetový obchod

Hlavní určením tohoto modulu je především umožnění zákazníkům komfortnější kontakt s obchodem, a to formou nákupu zboží přes internet, zjišťováním aktuální dostupnosti zboží na prodejně, a v neposlední řadě kontrolou stavu zákazníkových zakázek. K tomuto modulu zákazníci přistupují přes webové rozhraní.

Katalog zboží zobrazovaný v internetovém obchodě bude tříděn buďto dle výrobce nebo dle kategorie, do které je zboží zařazeno. U každého zboží by mělo být umožněno zobrazovat následující atributy:

- název zboží
- výrobce
- kategorie
- cena (včetně DPH)

- sazba DPH
- procentuální výše slevy
- cena ve slevě
- záruka
- zda je zboží skladem
- obvyklá dostupnost
- slovní popis

Zákazníci se pro pohodlnější práci mohou na těchto stránkách registrovat, je to také podmínkou pro využívání většiny výhod tohoto modulu. Mělo by jim být umožněno vkládat zboží do košíku, definovat si (i více) adres, prohlížet si historii objednávek a zakázek servisu včetně zjištění aktuálního stavu. Při vytvoření objednávky musí zákazníci specifikovat, zda si zboží hodlají vyzvednout osobně v obchodě, nebo využijí služeb některé přepravní firmy.

2.2.5 Uživatelská práva

Ve firmě je zaměstnáno 6 pracovníků na různých pozicích, je proto třeba do aplikace zaimplementovat správu uživatelů s rozlišnými uživatelskými právy. Firma by si přála program, který by umožňoval přístup do systému po přihlášení loginem a heslem.

Vedoucí firmy

Přístup do všech funkcí systému je umožněn uživateli s právy vedoucího firmy. Uživatel s těmito právy může jako jediný například spravovat jednotlivé položky zboží – upravovat ceny a další atributy. Je mu také umožněn přístup ke všem dále uvedeným funkcím určených jen pro konkrétní typ zaměstnance.

Prodavači

Systém dále podporuje přihlášení uživatelů s právy prodavače. Hlavní funkcí prodavačů je prodej zboží zákazníkům. Možnosti prodeje spočívají v přidávání položek prodeje na fakturu a její následný tisk. Těmto uživatelům je tedy umožněno prohlížet seznam prodávaného zboží včetně všech jeho detailů, avšak nemohou tyto hodnoty měnit. Vydávané faktury nemusejí být tvořeny jen ručním přidáváním jednotlivých položek na fakturu, na pokladně se pomocí faktur taktéž vydávají zakázky servisu a provádí se platba za ně. S právy prodavače má tedy uživatel přístup k seznamu zakázek servisu, a je mu umožněno tyto zakázky vyfakturovat. Obdobným způsobem má být řešeno vydávání přijatých objednávek. Prodavačům je k dispozici seznam všech objednávek v systému včetně jeho detailů. Tito uživatelé také mohou tyto objednávky realizovat (včetně vyfakturování).

Servisní technici

Třetím typem jsou uživatelé s právy servisního technika. Jejich hlavní náplní je provádění servisních oprav a s tím související přijímání objednávek. Tito uživatelé proto mohou vytvářet nové zakázky servisu a ukládat v nich provedené změny. Obdobně jako ostatní uživatelé mohou prohlížet seznam prodávaného zboží a v případě potřeby je jim umožněno přidávat tyto položky k existujícím zakázkám.

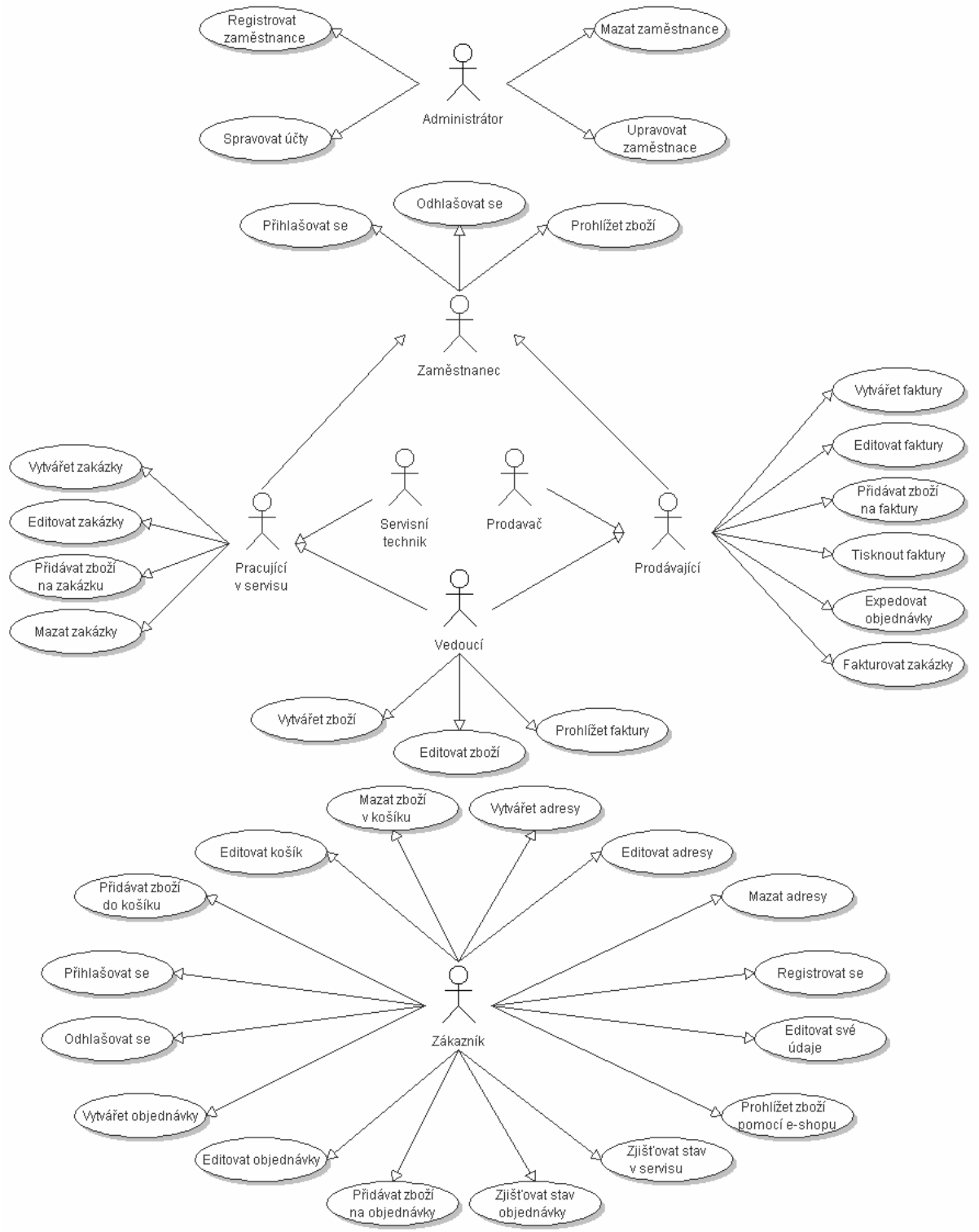
3 Návrh

Pro návrh tohoto informačního systému byly použity prostředky modelovacího jazyka UML.

3.1 Use case diagram

Use case diagram, neboli diagram typových úloh, či také diagram užití, je typ diagramu tříd omezený na elementy aktéra a typovou úlohu. Popisuje služby systému (typové úlohy) a jejich vzájemné vztahy. Vyjadřuje interakci (provázanost) typových úloh s aktérem, jinak řečeno jaké služby může aktér využívat. Aktér je vlastně zákazník, kterému IS poskytuje služby – převzato z [10].

Pro v této práci tvořený modulový informační systém byl sestaven následující diagram užití:

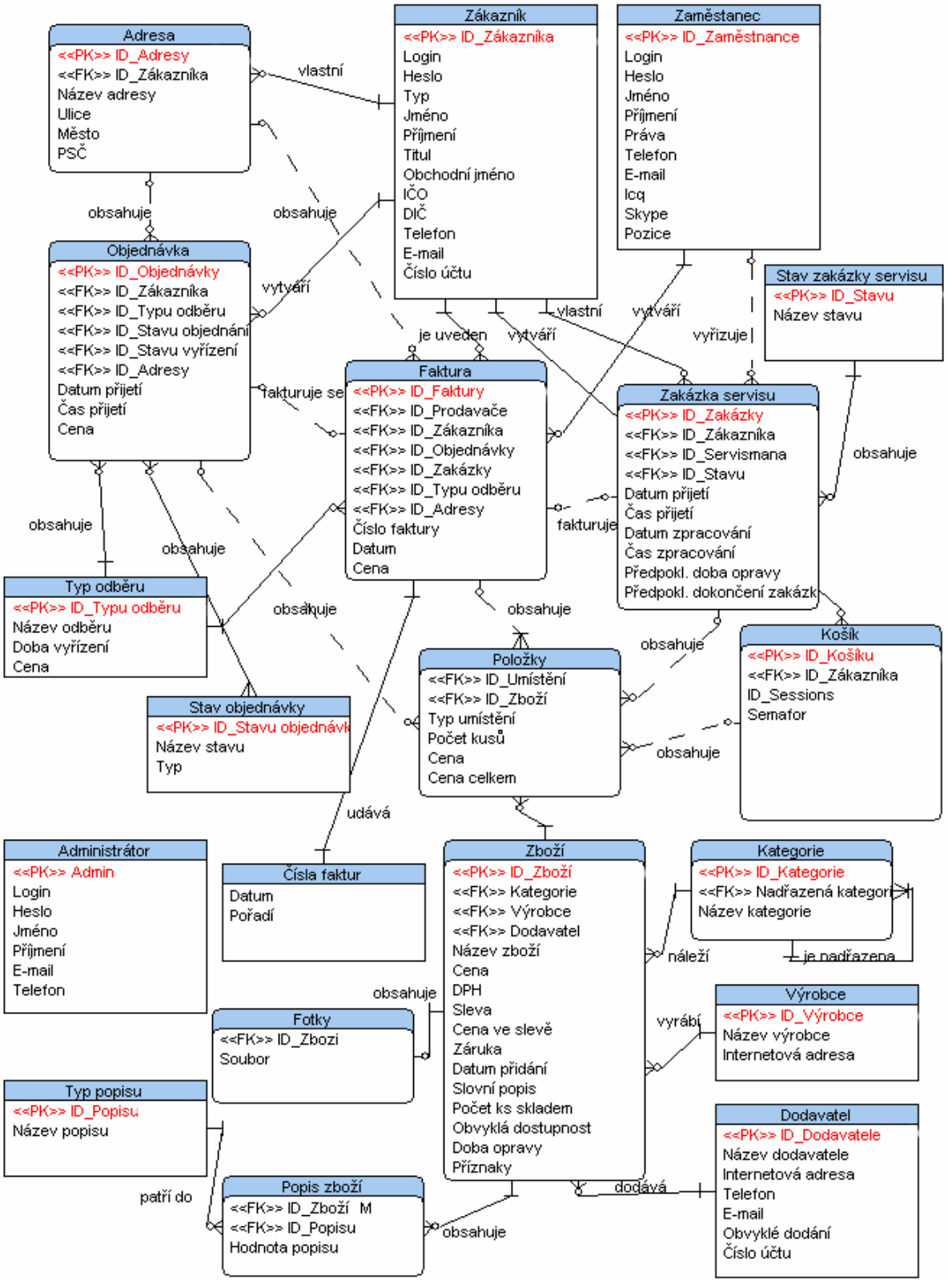


Obrázek 3.1: Use case diagram informačního systému

V uvedeném diagramu užití je zobrazeno 5 hlavních aktérů (vedoucí, servisní technik, prodavač, zákazník a administrátor). Dle diagramu lze rozpoznat, které úlohy jsou specifické pro konkrétního aktéra, či naopak které sdílí 2 aktéři společně (například vedoucí a prodavač – úlohy pro „prodávajícího“).

3.2 ER diagram

ER diagram je grafickým vyjádřením ER modelu. Používá se k zobrazení dat, která budou uložena v databázi.



Obrázek 3.2: ER diagram implementovaného informačního systému

Popis některých tabulek ER diagramu:

Zákazník

Podle hodnoty atributu „Typ“ je určeno, zda se jedná o fyzickou či právnickou osobu. Na tom závisí, zda jsou u uživatele sledovány atributy „Jméno“, „Příjmení“ a „Titul“ (v případě osoby fyzické) nebo atributy „Obchodní jméno“, „IČO“ a „DIČ“ (v případě právnické osoby).

Protože uživatel může mít definováno více adres, jsou jeho případné adresy uloženy v samostatné tabulce.

Čísla faktur

Tato tabulka slouží ke správnému generování čísla faktury pro potřeby firmy a její účetnictví. Primárním klíčem této tabulky je kombinace atributů „Datum“ a „Pořadí“, přičemž atribut „Datum“ odpovídá datu vystavení faktury a atribut „Pořadí“ udává, kolik faktur již bylo pro odpovídající datum vystaveno.

Položky

Pomocná tabulka pro vztah m..n. Slouží k uložení položek zboží na fakturách, objednávkách, v zakázkách servisu a v košíku. Atribut „Typ umístění“ určuje, pro jakou konkrétní tabulku jsou tyto data ukládána.

Současně je pro daný záznam tabulky uložena hodnota „Cena“, což zaručuje, že je v databázi uložena cena zboží platná v době operace se zbožím.

Typ popisu

Tato tabulka může uchovávat typy často se opakujících hodnot popisu zboží nebo typy popisu, pro který nabývá konkrétní zboží více hodnot. Tento způsob uložení také umožňuje podle těchto hodnot zboží porovnávat.

Popis zboží

Tabulka, ve které jsou pro konkrétní zboží uloženy konkrétní hodnoty pro daný typ popisu. Například pokud prodávané kolo je dostupné ve velikosti rámu 17“ a 19“ palců, je v tabulce Typ popisu uložena hodnota „Velikost rámu“ a v tabulce Popis zboží jsou 2 záznamy – obě hodnoty „ID_Zbozi“ ukazuji na ID tohoto kola, obě hodnoty „ID_Popisu“ ukazují na ID Typu popisu hodnoty „Velikost rámu“ a liší se v atributu „Hodnota popisu“ – první záznam má hodnotu „17“ a druhý „19“.

Kategorie

Atribut „Nadřazená kategorie“ nabývá hodnoty NULL, pokud daná kategorie je hlavní, v opačném případě je tato hodnota rovna ID nadřazené kategorie.

4 Implementace

Při implementaci projektu došlo k programovému spojení modulu pro řízení skladu a modulu pro řízení práce servisní dílny, a to především z důvodu jednodušší obsluhy ze strany budoucích uživatelů. Moduly jsou vzájemně propojené v několika úrovních, celkově tvoří kompaktní celek. Pro jejich implementaci byly po zvážení všech kladů a záporů jednotlivých implementačních prostředí zvoleny produkty firmy CodeGear, a to vývojové studio Delphi for Microsoft Windows verze 11.0.2627.5503, obsahující programovací jazyk Delphi 2007. Databázová část těchto modulů je postavena nad databází Interbase 2007 verze 8.0.0.123. Pro vytvoření databáze a prvotní naplnění daty bylo použito programu IBConsole verze 8.1.0.256.

Modul pro internetový obchod byl implementován pomocí jazyka PHP verze 5.1.4, s užitím zrcadla Apache verze 2.0.54, postavené nad databází MySQL verze 5.0.15. Pro správu databáze bylo použito phpMyAdmin verze 2.8.1.

4.1 Modul pro řízení skladu a servisní dílny

Pro tvorbu uživatelského rozhraní byl ve vývojovém prostředí RAD Studio zvolen projekt typu VCL Forms Application. Tento typ dovoluje vzájemné přepínání jednotlivých formulářů, určených pro konkrétní funkce programu.

Při implementaci těchto modulů v jazyce Delphi byla literatura [1], [2], [3] a [4] využívána jako manuál pro tento jazyk.

4.1.1 DataModule1

Jako základní formulář je veden formulář DataModule1. Jsou na něm umístěny všechny nevizuální komponenty programu, především komponenty potřebné pro propojení s databází Interbase. Patří mezi ně:

- komponenta databáze třídy TIBDatabase – reprezentuje spojení s Interbase databází
- komponenta třídy TIBTransaction – umožňuje provádění transakcí v databázi uvedené v TIBDatabase
- komponenta třídy TIBQuery – používá SQL příkazů k získání dat z jedné nebo více tabulek Interbase databáze; TIBQuery způsobí spuštění (provedení) příslušného dotazu a nabízí k dispozici jeho výsledky
- komponenta třídy TDataSource – zpřístupňuje data získaná z databáze pomocí dotazu v TIBQuery

Tento formulář neobsahuje žádné svoje funkce či procedury, slouží pouze pro ostatní formuláře jako zdroj dat a zpřístupnění databáze.

4.1.2 FormLogin

Úvodní formulář, který je zobrazen při spuštění aplikace, slouží pro přihlašování uživatelů do systému. Při přihlašování je kontrolováno uživatelské jméno a heslo. V případě úspěšného přihlášení jsou do globální proměnné uložena práva uživatele. Pokud je rozpoznaný uživatel veden s právy vedoucího, provede se přepnutí na formulář vedoucího, v opačném případě se zobrazí formulář pokladny.

4.1.3 FormVedouciho

Tento formulář slouží pro přístup k funkcím, které jsou přístupné pouze uživatelům s právy vedoucího. Kromě toho je uživatelům umožněno přepnout se do formuláře pokladny, který slouží i ostatním uživatelům jako výchozí bod pro provádění rozličných úkolů.

4.1.4 FormPokladna

Při aktivaci tohoto formuláře je podle práv přihlášeného uživatele nastavena pro jednotlivá tlačítka vlastnost „Enabled“, pomocí čehož je uživatelům povolen či zakázán přístup ke konkrétní funkci.

Obrázek 4.1: Formulář „Pokladna“ uživatele s právy servisního technika

4.1.5 FormZbozi

Formulář FormZbozi zobrazuje seznam všeho zboží v databázi firmy. Seznam je zobrazován v přehledové tabulce s výchozím nastavením zobrazování všech dostupných sloupců. Pro zobrazení této tabulky je použita komponenta třídy TDBGrid. Naplnění daty je implementováno v následujícím pořadí:

- komponentě IBQueryZbozi je do vlastnosti „SQL“ pomocí její funkce „Add“ postupně přidáván textový řetězec SQL dotazu
- po naplnění vlastnosti „SQL“ požadovaným řetězcem se nastaví vlastnost „Active“ této komponenty na hodnotu „True“, což způsobí vykonání vloženého SQL dotazu
- výsledek tohoto dotazu je přístupný přes komponentu DataSourceZbozi
- komponentě DBGridZbozi je pevně nastavena vlastnost „DataSource“ na hodnotu „DataSourceZbozi“, což znamená, že jsou v tabulce vždy zobrazovány aktuální data provedeného SQL dotazu

Pro vetší přehlednost si uživatel může zobrazování sloupců filtrovat dle jeho libosti tak, jak mu to v dané situaci vyhovuje. To je implementováno vynecháním částí původního textového řetězce SQL dotazu při vkládání do komponenty IBQueryZbozi.

Uživateli je umožněno filtrovat také zobrazovaný seznam, a to podle několika kritérií, které je možné vzájemně kombinovat. Toho je docíleno rozšířením textového řetězce přidáním odpovídající části SQL dotazu.

Mezi další procedury zpříjemňující uživateli práci s tímto formulářem patří procedura DBGridZboziTitleClick, která po kliknutí na titulek tabulky způsobí seřazení tabulky dle sloupce, na který bylo kliknuto (střídavě vzestupně a sestupně).

Pokud je uživatel v režimu tvoření faktury, je tento formulář použit pro přidávání položek na fakturu. To je implementováno přes tlačítko ButtonPridatZbozi:

- Po kliknutí na toto tlačítko je nejprve pomocí následujícího úseku kódu zkontrolováno, zda počet položek na aktuální faktuře nepřesahuje číslo 39 (z důvodu kapacity tištěné faktury):

```
DataModule1.IBQueryUpravyFaktury.SQL.Clear;
DataModule1.IBQueryUpravyFaktury.SQL.Add ('SELECT COUNT(idZbozi) FROM
Polozky WHERE typUmisteni = ''F'' AND idUmisteni = ' + idFaktury);
DataModule1.IBQueryUpravyFaktury.Active := True;
if DataModule1.DataSourceUpravyFaktury.DataSet.Fields[0].AsInteger > 39
then
  Application.MessageBox('Počet položek na faktuře přesahuje maximální
  počet (40 položek)','Chyba', MB_OK + MB_ICONERROR)
else begin ...
```

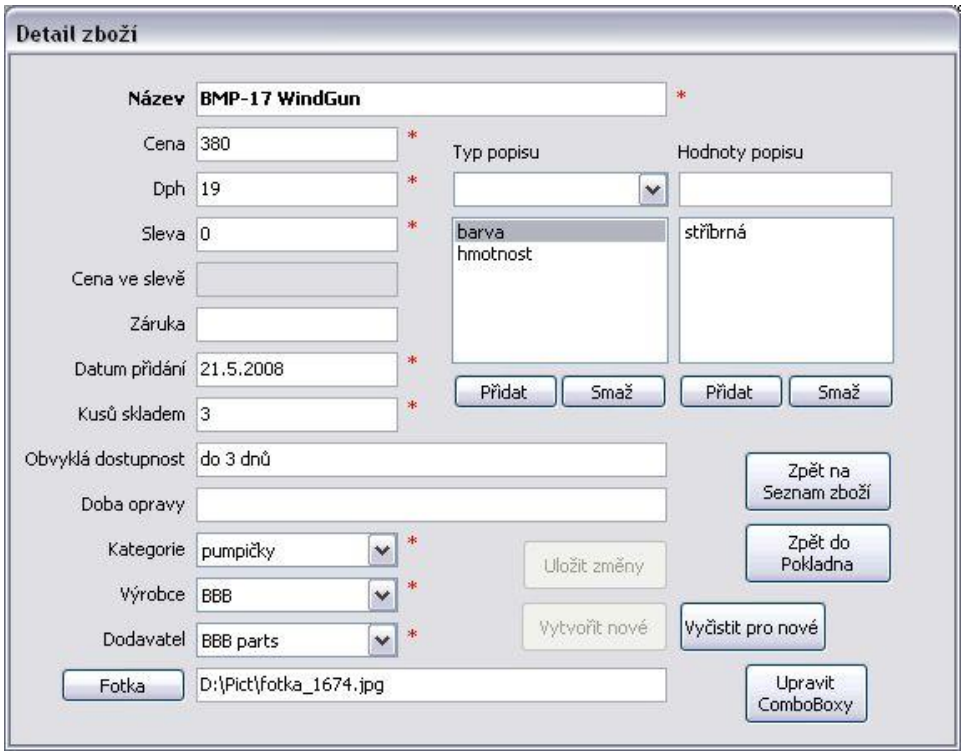
- Pokud je počet položek na faktuře v pořádku, je ze zobrazované tabulky zjištěno jednoznačné ID zvoleného zboží.
- Poté je uživatel dotázán na požadovaný počet kusů vkládané položky.
- Následuje kontrola, zda vkládané zboží již není na faktuře vloženo. V kladném případě je zobrazeno upozornění na tuto skutečnost, v opačném případě je do databázové tabulky Polozky vložen nový řádek s právě získanými údaji.
- Nedojde však k potvrzení transakce. To nastane, až uživatel ve formuláři FormFaktura potvrdí vytvoření nové faktury (příkazem Commit komponenty IBTransaction1). Pokud k tomu nedojde, je tato transakce stornována (příkazem Rollback komponenty IBTransaction1).

Pokud má uživatel tento formulář aktivní a je v režimu editace či vyváření nové zakázky servisu, je obdobným způsobem řešeno přidávání položek na danou zakázku.

Uživatel s právy vedoucího může v tomto formuláři také jednotlivé zboží mazat (pokud mu to umožní referenční integrita dat databáze) a zvolením konkrétní položky má možnost zvolené zboží upravovat – pomocí odkazu na formulář FormDetail. Ostatním uživatelům (tedy s právy prodavače či servisního technika) je umožněno detaily konkrétního zboží pouze prohlížet.

4.1.6 FormDetail

Prohlížení i editace vlastností konkrétní položky zboží je implementováno pomocí formuláře FormDetail. Při aktivaci tohoto formuláře jsou načteny všechny dostupné hodnoty zvoleného zboží a podle práv přihlášeného uživatele jsou povolena či naopak zakázána jednotlivá tlačítka na formuláři umožňující provádět změny. Mezi atributy, které uživatel s odpovídajícími právy může editovat, patří hodnoty popisu zboží (pomocí skupin komponent „Typ popisu“ a „Hodnoty popisu“).



Obrázek 4.2: Formulář „Detail Zboží“ uživatele s právy vedoucího

Tento formulář slouží také uživatelům s právy vedoucího pro zadávání nového zboží. Při požadavku na jeho vytvoření je zkontrolováno vyplnění všech nutných atributů a pokud je vše v pořádku, provede se zápis nového zboží do databázové tabulky. Poté je režim tohoto okna změněn na editační (jsou tedy detekovány změny atributů a je umožněno editovat hodnoty popisu zboží pomocí skupin komponent „Typ popisu“ a „Hodnoty popisu“). Záznam o rozpoznané změně atributu je veden pomocí pole „`b: array [1..14] of boolean`“, kde každý atribut má svůj index standardně definovaný jako false, a při detekované změně je atributu odpovídající index změněn na true.

V editačním režimu je mimo jiné umožněno zadávat procentuální výši slevy. To je implementováno pomocí reakce na událost „OnExit“ komponenty „EditSleva“. Při deaktivaci zaměření tohoto políčka je vyvolána procedura definovaná pro tuto událost, která způsobí vypočtení hodnoty ceny po slevě na základě zadaných procent slevy. Daný fragment kódu (podmínky řeší, zda sleva je či není zadaná):

```
if ((NOT(EditSleva.Text = '0')) and (NOT(EditCena.Text = ''))) then
begin
  EditCenaVeSleve.Text   :=   IntToStr   (Round(StrToInt(EditCena.Text)   *
  (100-StrToInt(EditSleva.Text)) / 100 ));
end;
if EditSleva.Text = '0' then EditCenaVeSleve.Clear;
```

4.1.7 FormUpravaComb

Tento formulář slouží uživatelům s právy vedoucího pro editaci databázových tabulek Kategorie, Vyrobce, Dodavatel a TypPopisu. Pro každou tabulku je těmto uživatelům umožněno mazat existující záznamy, vytvářet záznamy nové a také pomocí dvojkliknutí na již existující záznam tabulky zvolený záznam editovat.

4.1.8 FormUpravaKategorie

Formulář slouží pro editaci existujících záznamů tabulky Kategorie. Při aktivaci jsou pro zvolený záznam načteny jeho hodnoty a zobrazeny ve formuláři. Při stisku tlačítka ButtonUlozit dojde k zapsání hodnot do databázové tabulky. Pokud je jako nadřazená kategorie zvoleno „žádná“, je pro odpovídající atribut tabulky uložena hodnota „NULL“. V opačném případě je uloženo ID zvolené kategorie.

4.1.9 FormUpravaVyrobce, FormUpravaDodavatel

Obdobné formuláře jako FormUpravaKategorie.

4.1.10 FormFaktura

Formulář FormFaktura je určen pro tvorbu a tisk faktur za:

- zboží prodávané zákazníkovi osobně na prodejně
- dokončené zakázky servisu (včetně položek s tím souvisejících)
- expedované objednávky

Přístup do tohoto formuláře mají jen zaměstnanci firmy s uživatelskými právy vedoucí či prodavač.

Aktivace formuláře pro novou fakturu

Při prvotní aktivaci tohoto formuláře pro novou fakturu jsou automaticky provedeny následující kroky:

- komponentám, kterým se během tvorby faktury mění vlastnost „Enabled“, jsou tyto hodnoty nastaveny na výchozí
- podle přihlášeného uživatele jsou vyplněny políčka „Jméno“ a „Příjmení“ prodavače
- vyplněno políčko „Číslo faktury“
- nastaveno ID této nové faktury

Číslo faktury je uváděno ve formátu rrrrmmddccc, (r..rok, m..měsíc, d..den, c..pořadové číslo faktury vydané týž den – například 20080418004). Toto číslo je vytvořeno jako spojení 2 textových řetězců (proměnných „cisloFaktury1“ a „cisloFaktury2“). První řetězec (cisloFaktury1) je vytvořen pomocí funkce „`FormatDateTime('yyyymmdd',Now)`“, která vrátí aktuální datum v datovém typu string. Pro druhý řetězec (cisloFaktury2) je nejdříve zjištěn počet záznamů v tabulce CislaFaktur, které mají datum dne, kdy je faktura tvořena. Pokud je zjištěný počet roven nule, je proměnné „cisloFaktury2“ přiřazena hodnota „001“ (jedná se o první fakturu v daném dni). Pokud je zjištěný počet vyšší než nula, je do proměnné „cisloFaktury2“ přiřazen řetězec odpovídající číslu o 1 větší, než je maximální hodnota sloupce „pořadí“ pro záznamy s datem téhož dne. Pomocí následujícího úseku kódu je zajištěno, aby proměnná „cisloFaktury2“ byla textový řetězec o právě 3 znacích:

```
cisloFaktury2 := ('000' + IntToStr(DataModule1.
  DataSourceUpravyFaktury.DataSet.Fields[1].AsInteger + 1));
cisloFaktury2 := RightStr (cisloFaktury2, 3);
```

Nastavení ostatních komponent při aktivaci a také dostupné operace s tvořenou fakturou se odvíjí podle toho, zda se faktura vytváří jako prázdná (typicky pro osobní odběr), jako vyfakturování zakázky servisu nebo jako faktura z objednávky.

Prázdná faktura

Pokud je faktura vytvářena jako prázdná, jsou při aktivaci formuláře naplněny ComboBoxy komponent pro Typ odběru a Login zákazníka všemi dostupnými hodnotami z databáze a zobrazeny výchozí hodnoty pro tyto účely. Prodavač poté může dvěma způsoby vyplnit fakturační údaje dle přání zákazníka:

- hodnoty vypíše ručně – příslušná editační pole aktivuje pomocí volby některého z tlačítek třídy TRadioButton
- v případě registrovaného zákazníka může pomocí volby některé z hodnot ComboBoxů „Login zákazníka“ či „Adresa“ vybrat údaje zaregistrované zákazníkem

V případě potřeby také může prodavač změnit způsob odběru.

Faktura ze zakázky servisu

Jestliže je faktura vytvořena za zakázku servisu, která nemá definovaného zákazníka, jsou možnosti zadávání fakturačních údajů obdobné jako v případě tvorby nové prázdné faktury. Pokud ovšem zákazník na zakázce servisu definován je, jsou údaje na faktuře o údaje tohoto zákazníka (jméno, příjmení, seznam dostupných adres apod.) doplněny automaticky a není možná jejich úprava (kromě fakturační adresy).

Zakázce, která je tímto fakturována, je změněna hodnota stavu zpracování na hodnotu „vydáno“.

Na závěr aktivace tohoto formuláře jsou na faktuře doplněny položky zboží, odpovídající položkám zakázky servisu, pro kterou je tato faktura tvořena. Je také provedena kontrola, zda pro vkládanou položku je skladem odpovídající počet kusů. Pokud ne, je přidán jen dostupný počet kusů a uživateli je zobrazena informace o vzniklé situaci.

Faktura z objednávky

Postup při aktivaci formuláře pro fakturu z objednávky je obdobný jako pro fakturu tvořenou ze zakázky servisu – společné údaje jsou překopírovány, včetně položek zboží.

Operace s fakturou

Přidávání jednotlivých položek zboží je možné přes tlačítko „Přidat“ (ButtonPridatPolozku), který přepne aktuálně zobrazený formulář FormFaktura na formulář FormZbozi, umožňující vkládání jednotlivých položek zboží na fakturu (viz výše). Aby se při návratu z formuláře FormZbozi na formulář FormFaktura opětovně neprovedly operace související s aktivací tohoto formuláře, jsou v programu pro jednotlivé případy aktivace formuláře definovány proměnné typu boolean, které zjišťují provedení jen požadovaných úseků kódu.

Položkám na faktuře může prodavač také upravovat počet prodávaných kusů nebo je z faktury mazat.

Potvrzení vytvoření faktury

Vytvoření faktury v databázi (a uložení v systému), včetně potvrzení všech databázových transakcí souvisejících s touto fakturou, nastane až poté, co uživatel potvrdí vznik faktury přes tlačítko „Vytvořit fakturu“ (ButtonVytvoritFakturu). To způsobí provedení příkazu „Commit“ pro komponentu IBTransaction1, která obsahuje seznam všech v databázi prováděných transakcí.

Potvrdit vytvoření faktury je ovšem možné pouze, pokud faktura obsahuje nejméně jednu položku – vytvořit fakturu s prázdným seznamem položek nejde.

Opačný efekt než příkaz „Commit“ má příkaz „Rollback“ (stejné komponenty), který způsobí stornování všech změn v databázi souvisejících s touto fakturou – provádí se po potvrzení volby tlačítka „Storno“ (ButtonStorno).

Tisk faktury

Po úspěšném vytvoření faktury (a jejím uložení v databázi) je povoleno tlačítko pro tisk (ButtonTisknout). To způsobí:

- přenastavení vizuálních vlastností formuláře a jeho komponent na hodnoty vhodné pro tisk (viditelnost netisknutých komponent, velikost formuláře a jeho barva, nastavení tabulky položek a podobně)
- zobrazení dialogového okna pro nastavení parametrů tisku
- tisk formuláře
- opětovné přenastavení vizuálních vlastností formuláře a jeho komponent, tentokrát na standardní hodnoty

Tiskárna, na kterou bude faktura vytisknuta, je volena uživatelem v zobrazeném dialogovém okně. Tamtéž se mění i některé další parametry tisku.

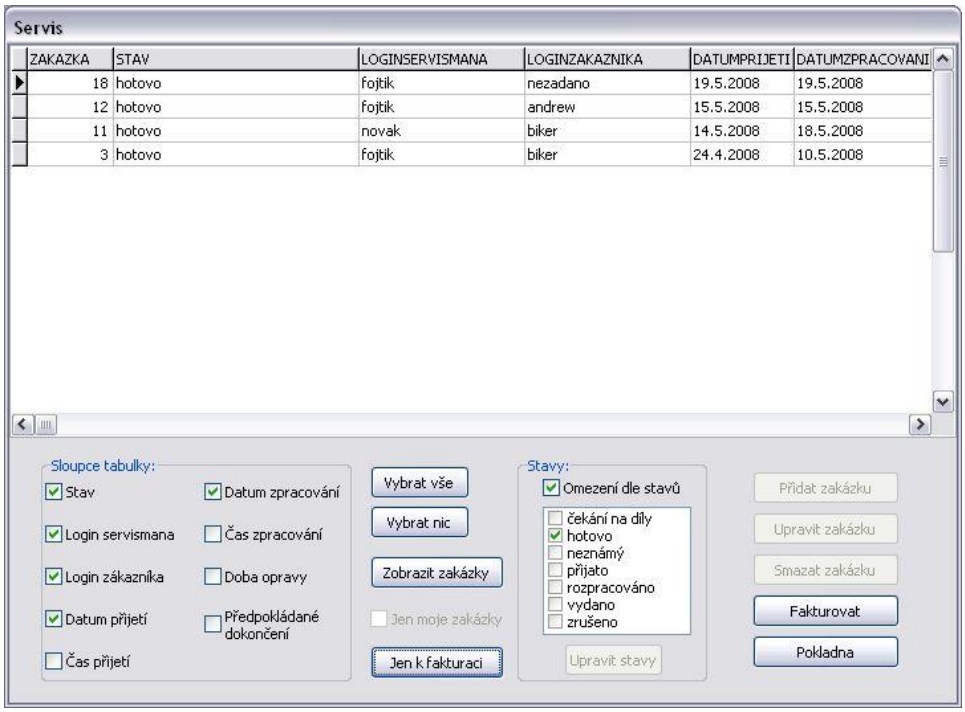
4.1.11 FormServis

Formulář zobrazující seznam zakázek servisu. Ústřední formulář pro modul řízení práce servisní dílny. Při aktivaci tohoto formuláře je podle práv přihlášeného uživatele nastavena vlastnost „Enabled“ tlačítkům, která jsou povolována podle uživatelských práv.

Pro komfortnější práci se seznamem zakázek je obdobně jako u formuláře FormZbozi umožněno zobrazovaný seznam zakázek filtrovat, a tedy zobrazovat zakázky:

- podle stavu vyřízení zakázky (každý uživatel v tomto formuláři)
- jen ty, které má přihlášený servisní technik přidělené (jen uživatelé s právy servisního technika)
- jen ty, které jsou určeny k fakturaci – tedy se stavem vyřízení „hotovo“ (jen uživatelé s právy prodavače či vedoucího)
- ke všem výše zmíněným filtrům je možné definovat, které sloupce zobrazované tabulky se mají zobrazit a které nikoliv

Obdobně jako u formuláře FormZbozi je také možno zobrazovaná data třídit dle hodnot ve sloupci kliknutí na titulek daného sloupce.



Obrázek 4.3: Formulář „Servis“ uživatele s právy prodavače

Tlačítko „Upravit stavy“ (ButtonSeznamStavu) zobrazuje formulář FormStavyServisu (viz dále).

Pomocí tlačítka „Smazat“ mohou uživatelé s příslušnými právy zakázky servisu mazat. Pro editaci existujících a vytváření nových zakázek slouží přes tlačítka “Upravit zakázku“ a „Smazat zakázku“ zobrazovaný formulář FormDetailServisu.

4.1.12 FormStavyServisu

V tomto formuláři může uživatel s právy vedoucího, obdobně jako na formuláři FormUpravaComb, editovat existující stavy zakázek servisu, může je mazat a také má možnost vytvářet stavy nové.

4.1.13 FormDetailServisu

Podle způsobu volání tohoto formuláře z formuláře FormServis je umožněna editace konkrétní zakázky servisu či její vytvoření.

Vytvoření nové zakázky

Při aktivaci formuláře pro účel vytvoření nové zakázky servisu je nastaveno nové číslo zakázky, automaticky předvyplněno políčko servisního technika přebírajícího zakázku (dle přihlášeného uživatele) a datum a čas přijetí (dle aktuálního času). Přes tlačítko „Přidat“ (ButtonPridatPolozku) je uživatel přepnut do formuláře FormZbozi, umožňující přidávání položek na zakázku (automaticky je

nastaven filtr zobrazování položek ve FormZbozi tak, aby zobrazil jen položky související se servisem).

V tabulce DBGridZakazkyServisu je zobrazen sloupec „Doba opravy“, obsahující pro položky oprav předpokládanou dobu strávenou prováděním dané opravy. Servisní technik může na základě těchto hodnot udělat předběžný odhad ohledně dokončení zakázky a má možnost tyto předpokládané hodnoty uložit do databáze. Tyto hodnoty si pak může zákazník zjistit přes internet.

Vytvoření zakázky je dokončeno po potvrzení volby tlačítka „Vytvořit novou“ (ButtonNovaZakazka), čemuž předchází kontrola vyplnění všech povinných údajů. Po dokončení vytvoření nové zakázky je formulář zobrazen v režimu editace zakázky.

Editace zakázky

Při zobrazení formuláře FormDetailServisu v editačním režimu jsou nejprve vyplněna všechna editační políčka formuláře dle hodnot uložených v databázi, včetně tabulky DBGridZakazkyServisu zobrazující položky zvolené zakázky.

Uživatel má možnost měnit některé atributy, včetně přiřazení zakázce datum a čas dokončení. Pro toto přiřazení je nejprve nutné povolit uložení data a času dokončení pomocí tlačítek ButtonDatumZpracovani a ButtonCasZpracovani, čímž se povolí komponenty DateTimePickerDatumZpracovani a DateTimePickerCasZpracovani, umožňující pohodlné zadávání data a času. Jako výchozí hodnoty jsou těmto komponentám přiřazeny hodnoty aktuálního data a času. Ukázka kódu právě popsaného procesu, včetně detekce změny hodnoty v komponentě:

```
if ButtonDatumZpracovani.Caption = 'Povolit' then
begin
  EditDatumZpracovani.Visible := false;
  DateTimePickerDatumZpracovani.Visible := true;
  ButtonDatumZpracovani.Caption := 'Zakázat';
  DateTimePickerDatumZpracovani.Date := Now;
  if isReadyZakazka then
  begin
    ButtonUlozit.Enabled := true;
    s[6]:=true;
  end;
end else
```

Ke změně počtů kusů položky na zakázce slouží tlačítko „Změnit # ks“ (ButtonUpravitPolozku). Ke smazání položky je určeno tlačítko „Smazat“ (ButtonSmazatPolozku).

Pokud je během zobrazení formuláře FormDetailServisu v editačním režimu detekována změna některé komponenty, je pro index odpovídající této komponentě v poli „`s: array [1..10] of`

`boolean`“ změněna hodnota z „false“ na „true“ a je povoleno tlačítko „Uložit změny“ (ButtonUlozit).

Tlačítko „Vyčistit pro nové“ (ButtonClear) slouží pro přepnutí do režimu tvorby nové zakázky (včetně s tím souvisejícího vyčištění editačních políček).

4.1.14 FormObjednavky

Formulář zobrazující seznam objednávek, vytvořených zákazníky pomocí modulu pro elektronický obchod. Formulář je přístupný jen pro uživatele s právy vedoucího nebo prodavače. Jako všechny ostatní výpisy, je pro komfortnější práci i tento seznam zobrazovaný pomocí tabulky DBGridObjednavky možno filtrovat.

Pokud je uživatel veden s právy vedoucího, tradičně může přes tlačítko „Upravit stavy“ (ButtonSeznamStavuObj) zobrazit formulář (FormStavyObjednavky), umožňující mazat a editovat stávající stavy objednávek, stejně jako vytvářet stavy nové.

Při volbě tlačítka „Detail objednávky“ (ButtonDetailObjednavky) je zvolená objednávka zobrazena detailněji (pomocí formuláře FormDetailObjednavky).

4.1.15 FormStavyObjednavky

Tento formulář pracuje obdobně jako ostatní formuláře (například FormUpravaComb či FormStavyServisu), umožňující uživatelům s právy vedoucího úpravy možných stavů, které mohou pro určité databázové tabulky systému nastat.

V tomto formuláři je navíc třeba specifikovat, zda je editovaný stav zřízen pro určení stavu objednání (hodnoty v objednávce může měnit zákazník) nebo pro určení stavu vyřízení (hodnoty mění zaměstnanci firmy).

4.1.16 FormDetailObjednavky

Formulář sloužící k detailnějšímu zobrazení konkrétní objednávky elektronického obchodu. Při aktivaci jsou z databáze načteny všechny údaje o dané objednávce a jsou zobrazeny na formuláři. Položky této objednávky až na jednu výjimku nelze měnit. Případnou úpravu položek (jako například vymazání nedostupného zboží) je nutné provést až na faktuře při fakturování této objednávky. Onou výjimkou, kterou prodavač může měnit, je stav vyřízení sloužící pro lepší informovanost zákazníků o stavu zpracování jejich objednávek firmou.

Opět je detekována případná změna provedená uživatelem (i když tentokrát jen u jediné komponenty). Pokud je tato změna rozpoznána, je aktivováno tlačítko „Uložit změny“ (ButtonUlozit), které tuto změnu uloží do databáze.

Vyfakturování této objednávky je možné volbou tlačítka „Fakturovat“ (ButtonFakturovat). Potvrzení této volby způsobí předání atributů této objednávky formuláři FormFaktura. Pokud byla

u objednávky detekována uživatelem provedená změna a nedošlo k potvrzení této změny, je uživatel dotázán, zda chce tuto změnu uložit či nikoliv. Pokud ano, je změna zaznamenána do databáze, pokud ne, je naopak z databáze načtena uložená hodnota.

4.2 Databázová část

Databázi aplikace, běžící na lokálním serveru Interbase, bylo třeba nejdříve vytvořit a nadefinovat jí její tabulky. To umožnil program IBConsole, který umožňuje přehlednou správu databází a spouštění SQL příkazů včetně skriptů. Vytvořeným databázím je možné pomocí tohoto programu nastavit všechny potřebné vlastnosti, jako je uživatelské jméno a heslo, znaková sada apod. Po vytvoření a nastavení vlastností databáze byly databázi definovány tabulky. Nakonec byly tabulky naplněny prvotními údaji.

Při skládání SQL dotazů byly čerpány informace z www stránek [5].

4.2.1 Synchronizace

Protože je databázová část systému uložena na dvou různých serverech, je nutné data na těchto serverech vzájemně synchronizovat.

Synchronizace je zatím pro uživatele poměrně nekomfortní, což ovšem bude v brzké budoucnosti změněno, a to díky přechodu na jediný server, provozovaný přímo firmou Cyklosport Pavka.

4.3 Modul pro internetový obchod

Tento modul slouží jako internetový obchod, umožňuje uživatelům zjišťovat aktuální dostupnost prodávaného zboží a tvoří veřejné rozhraní pro modul servisní dílny, jelikož je přes něj uživatelům umožněno online kontrola stavu jejich zakázek.

Modul je navržen jako internetové stránky v jazyce PHP využívající databázi MySQL pro zobrazování dat. Celek tvoří 6 souborů v jazyce PHP, které generují kód jazyka HTML. Dále obsahuje další 2 pomocné soubory JavaScriptu a kaskádových stylů. Tyto pomocné soubory se starají o grafický vzhled generovaného HTML kódu.

Při programování tohoto modulu byly využívány informace o jazyku PHP z literatury [6], [7], [8] a [9].

4.3.1 index.php

Tento soubor volá ostatní soubory, které se vkládají do tohoto souboru. Sám vykonává pouze 2 funkce.

První funkcí je spouštění Sessions a uložení jeho jedinečného identifikačního čísla (dále jen id) do globální proměnné; toto jedinečné id slouží k identifikaci nepřihlášených návštěvníků stránek, pokud je návštěvník přihlášen, pracuje se přímo s proměnnou uloženou v Sessions.

Funkcí druhou je generování základního HTML kódu (HTML hlavička, HTML body a základní rozvržení stránky pomocí tagů).

4.3.2 base.php

Soubor base.php propojuje zdrojový kód stránek s databází MySQL a zaručuje správné kódování vkládaných a vyjímaných dat databáze. To je nutné z důvodu odlišného kódování internetových stránek a databáze.

4.3.3 support.php

V tomto souboru se provádějí veškeré akční operace. Mezi nejdůležitější patří:

- Přihlašování návštěvníka. Nejprve jsou přijata data z formuláře webové stránky, poté jsou odeslaná data zkontrolována, následuje databázový dotaz s údaji přijatých z formuláře (příkazem MySQL_Query) a výsledek dotazu je zjišťován pomocí funkce MySQL_Num_Rows (vrací počet řádků zadanému SQL dotazu). Pokud je zadaný uživatel v databázi nalezen, je provedeno přihlášení návštěvníka. Zároveň je zkontrolován obsah košíku do té doby nepřihlášeného uživatele, a pokud je neprázdný, je převeden na přihlášeného uživatele.
- Odhlášení návštěvníka. Při odhlášení jsou vyprázdněny proměnné uložené v Sessions.
- Vložení zboží do košíku. Je řešeno obdobně jako přihlašování uživatele, tedy kontrolou vstupních dat a porovnáním těchto dat s databází. V případě, že konkrétní zboží není doposud součástí košíku, je tam vloženo, v opačném případě je aktualizován počet kusů (jako součet původní a nově vkládané hodnoty).
- Vytvoření nového uživatele. Uživatelé jsou rozděleni na dvě skupiny – právnické a fyzické osoby. Na tom závisí vkládání konkrétních dat do databáze.
- Vytvoření nové adresy stávajícího uživatele. Po kontrole vstupních dat je detekováno, zda vkládané jméno adresy patři mezi předdefinované, nebo nikoliv. Pokud ne, je do databáze vložen název adresy definovaný uživatelem.
- Potvrzení objednávky. Při této operaci jsou možné dva stavy – uživatel si zvolil osobní odběr na prodejně a není tedy potřeba žádné adresy, či uživatel zvolil jiný způsob dodání a adresa je tedy nutná. Po vytvoření nové objednávky jsou z košíku zkopírovány data do tabulky Položky. Na závěr této operace jsou položkám v košíku nastaveny semafory na stav odesláno. Pole Semafor slouží k identifikaci stavu objednávek v košíku. Již odeslané položky jsou průběžně mazány, neodeslané zůstávají uživateli stále přístupné.

Ostatní operace umožněné v tomto souboru jsou:

- Odstranění položky z košíku.
- Změna počtu kusů v košíku.
- Úprava adresy.
- Změna hesla.
- Změna osobních údajů.
- Změna stavu objednávky.

Jejich implementace je obdobná jako výše popsané operace.

4.3.4 stred.php

Soubor obsahující kód pro uživatelský panel. Provádí a následně zobrazuje aktuální součet hodnot položek v košíku. Součet je počítán jen z položek, které mají počet kusů větší jako nula. To je kontrolováno příkazem:

```
$dotaz = MySQL_Query ("SELECT SUM(cena*pocet) FROM Kosik WHERE idZakaznika
= '$_SESSION[login]' AND semafor = 'add' AND pocet > '0'");
```

4.3.5 menu.php

Soubor ve značné míře využívá JavaScript. Jeho účelem je zobrazování či skrývání položek menu (podle „Výrobce“ nebo „Kategorie“), které slouží k výpisu všeho zboží pro zvoleného výrobce nebo kategorii (včetně podkategorií). Pro rozpoznání, zda kategorie je hlavní (a nemá žádnou nadřazenou kategorii), je v kódu použita podmínka „`WHERE nadrazena IS NULL`“.

4.3.6 page.php

Souboru jsou předány vstupní proměnné, poslané z předchozí navštívené stránky. Podle těchto proměnných se zvoleno provedení konkrétní části kódu. Která část se provede, závisí na vyhodnocení podmíněných příkazů vyhodnocujících vstupní podmínky souboru.

U proměnné $_GET[page] rozlišujeme 4 hlavní hodnoty :

- kategorie
- vyrobci
- kosik
- a všechny ostatní možnosti

kategorie

Pokud je v proměnné $_GET[page] hodnota „kategorie“, chod programu se dále větví dle hodnot v proměnných $_GET[num] (obsahuje hodnotu ID zvolené kategorie) a $_GET[zbozi] (obsahuje hodnotu ID zvoleného zboží):

- $_GET[num] != ‘ ‘ && $_GET[zbozi] == ’’ odpovídá stavu, kdy je zadána kategorie, avšak není vybráno konkrétní zboží. V tomto případě se na stránce zobrazí seznam zboží odpovídající zvolené kategorie.
- $_GET[num] != ‘ ‘ && $_GET[zbozi] != ’’ odpovídá stavu, kdy je vybrána jak kategorie, tak i konkrétní zboží. Na stránce jsou tedy zobrazeny detaily zvoleného zboží.
- ostatní možnosti – v těchto případech se vypíše seznam všech kategorií

vyrobci

Pokud je v proměnné $_GET[page] hodnota „vyrobci“, chová se program obdobně jako v případě, kdy je v této proměnné hodnota „kategorie“.

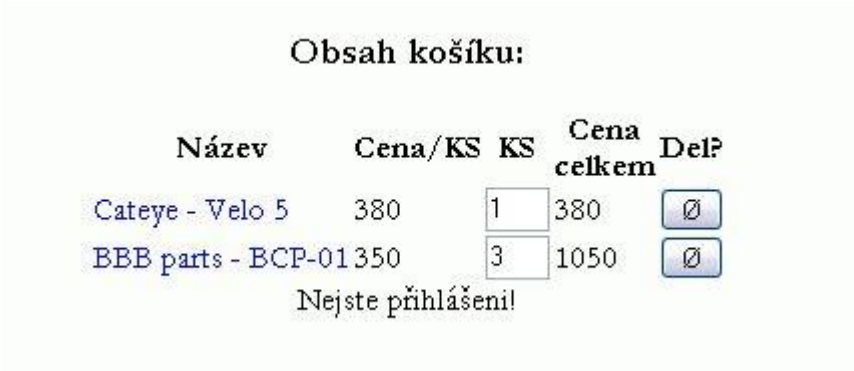
kosik

Pokud je v proměnné $_GET[page] hodnota „kosik“, chod programu se dále větví dle hodnot v proměnné $_GET[akce] (do této proměnné jsou ukládány hodnoty pro identifikaci prováděné akce).

Seznam hodnot proměnné $_GET[akce], kterým odpovídají prováděné operace:

- prihlas – Jednoduchý formulář pro odeslání dat potřebných pro přihlášení uživatele.
- menu – Nabízí přihlášenému uživateli seznam základních operací (úprava osobních údajů, kontrolování objednávek a zakázek servisu a správa adres).
- new_adr – Slouží přihlášenému uživateli k zadávání adres. Uživatel si může pro název adresy zvolit předdefinovanou hodnotu nebo zvolit vlastní typ, což pomocí JavaScriptu zaktivuje políčko pro vlastní název adresy.
- up_adr – Slouží k úpravě adresy. Název již nelze změnit, pouze ostatní údaje.
- up_pi – Slouží k úpravě osobních údajů uživatele – buď jeho hesla nebo osobních údajů uživatele. Pro osobní údaje je rozlišováno, zda je přihlášený uživatel fyzická či právnická osoba – podle toho se aktivují nebo deaktivují příslušná editační políčka.
- registr – Obsahuje formulář pro vytvoření nového uživatele. Mezi části tohoto formuláře patří editační políčka pro login, heslo, kontrola hesla a osobní údaje uživatele. Podle volby právnické či fyzické osoby jsou aktivní příslušná políčka.
- kosik – Vypisuje obsah košíku přihlášených i nepřihlášených uživatelů. Uživatelům je umožněno měnit počet kusů daného zboží v košíku a také toto zboží mazat. Objednat obsah košíku je umožněn jen přihlášeným uživatelům. Nepřihlášení uživatelé se pro

objednání položek v košíku musí nejprve přihlásit (obsah košíku jim bude automaticky převeden).



Obrázek 4.4: Část internetové stránky zobrazující obsah košíku nepřihlášeného uživatele

- objednat – Slouží pro vygenerování formuláře objednávky z obsahu košíku. Pomocí JavaScriptu je také nastavena celková cena objednávky jako součet cen všech položek a ceny dopravného.
- kont_ob – Zobrazuje uživateli seznam jeho objednávek.
- up_ob – Pokud je hodnota proměnné $_GET[typ] prázdná hodnota, program je totožný jako v případě $_GET[akce] = kont_ob (viz výše). Pokud ale hodnota proměnné $_GET[typ] je neprázdná, zobrazí se detail objednávky, včetně přehledné tabulky položek objednávky.
- kont_ser – Zobrazuje seznam uživatelových zakázek servisu
- view_ser – Pokud je hodnota proměnné $_GET[typ] prázdná hodnota, program je totožný jako v případě $_GET[akce] = kont_ser (viz výše). Pokud ale hodnota proměnné $_GET[typ] je neprázdná, zobrazí se detail zakázky servisu.
- všechny ostatní případy – Je zobrazeno stručné menu základních nabídek jako například košík, odhlášení apod.

ostatní možnosti

Pokud je v proměnné $_GET[page] hodnota jiná než „vyrobci“, „kategorie“ nebo „kosik“, je zobrazena domovská stránka.

4.3.7 style.css

Soubor obsahující hodnoty pro kaskádové styly. CSS je jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML. (viz. wikipedia.org). Při programování byly využity www stránky [10].

4.3.8 skript.js

Soubor obsahuje 3 funkce, které jsou volány při různých událostech (onClick a onChange). Každá funkce dostává s jejím voláním předaný jeden parametr a ten používá pro identifikaci měněného (či změněného) objektu.

5 Závěr

V této práci jsem vytvořil funkční prototyp modulárního informačního systému pro firmu Cyklosport Pavka. Systém byl navržen, implementován a testován ve spolupráci s touto firmou. Při jeho vývoji docházelo k častým úpravám požadavků ze strany zadavatelské firmy Cyklosport Pavka, a tak bylo třeba tyto změny průběžně implementovat do již částečně zpracovaného projektu. Tyto požadavky jsou i nadále doplňovány a upravovány, a tak již nyní jsou známy některá další rozšíření, která je třeba do systému zaimplementovat.

Především bude zprovozněn produkční server provozovaný přímo firmou Cyklosport Pavka, který nahradí stávající lokální server Interbase a MySQL server aktuálně běžící u provozovatele webových stránek php5.cz. Jednoznačnou výhodou tohoto řešení bude především odpadnutí nutnosti oba servery vzájemně synchronizovat. Také vznikl požadavek, aby mohl majitel firmy vyhodnocovat statistiky prodeje zboží. I to bude v budoucnosti realizováno. Mezi uvažované rozšíření patří zavedení čtečky čárových kódů, což by vedlo ke zjednodušení práce zaměstnancům.

Díky poznatkům, které jsem nabyl při řešení této práce, se zvýšili mé zkušenosti při tvorbě aplikací tohoto typu, a tak si nyní mohu dovolit vytvářet i náročnější aplikace, na které bych si dříve netroufl. Také jsem pomocí tohoto projektu poznal návrh aplikace v praxi, se skutečnou firmou, řešení reálných požadavků a podobně. Tyto získané zkušenosti se budou v budoucnu určitě hodit.

Literatura

[1] WWW stránky. Stránky o programování v jazyce Delphi na serveru Živě.cz. http://www.zive.cz/Programovani/Delphi/sc-73/default.aspx

[2] Eller, Frank: DELPHI 6: příručka programátora. Praha, Grada, 2002.

[3] Teixeira, Steve: Borland Delphi: průvodce vývojáře. Brno, Mobil Media, 2002.

[4] WWW stránky. Stránky o programování v jazyce Delphi na serveru Programujte. http://programujte.com/index.php?rubrika=26&sekce=87

[5] WWW stránky. Stránky o programování SQL na serveru Interval.cz. http://interval.cz/serialy/sql-structured-query-language

[6] Gilmore, Jason W.: Velká kniha PHP 5 MySQL: kompendium znalostí pro začátečníky i profesionály. Brno, Zoner Press, 2005.

[7] WWW stránky. Stránky o programování v jazyce PHP na serveru PCSvět.cz. http://www.pcsvet.cz/php

[8] WWW stránky. Stránky o programování v jazyce JavaScript na serveru Jak psát web. http://www.jakpsatweb.cz/javascript

[9] WWW stránky. Stránky o programování v jazyce CSS na serveru Jak psát web. http://www.jakpsatweb.cz/css

[10] WWW stránky. Stránky o modelovacím jazyce UML. http://lide.uhk.cz/pdf/student/cermaha1/usecase.htm

Seznam příloh

Příloha 1. CD se zdrojovými texty a dokumentací.