VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

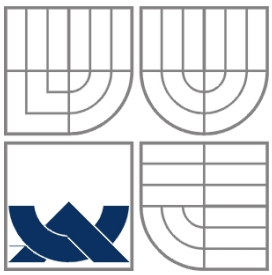
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SKLADOVÝ SYSTÉM MENŠÍ FIRMY

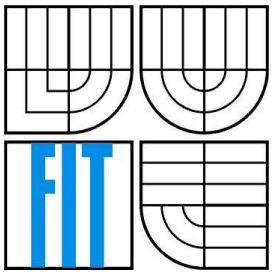
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE LADISLAV HEZOUČKÝ
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SKLADOVÝ SYSTÉM MENŠÍ FIRMY

STORE INFORMATION SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

LADISLAV HEZOUČKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

MGR. TOMÁŠ MASOPUST

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Hezoučký Ladislav**
Obor: Informační technologie
Téma: **Skladový systém menší firmy**
Kategorie: Databáze

Pokyny:

1. Seznamte se s principy tvorby dynamických www stránek
2. Analyzujte požadavky na tento systém. Pro analýzu využijte modelovacích technik jazyka UML.
3. Seznamte se s prostředím databázového systému MySQL a skriptovacími jazyky PHP, Perl a problematikou počítačových sítí.
4. Systém navrhněte tak, aby umožňoval zobrazovat statistiky zboží podle různých měřítek (cena, značka, nejprodávanější značka, nejprodávanější zboží vůbec atd.) a uměl rovněž vystavit fakturu na odebírané zboží. Tu bude umět exportovat do formátu pdf případně i jiného. Studujte rovněž užití různých javascript editorů pro editaci v systému.
5. Seznamte se důkladně s požadavky kladenými na skladový systém a analyzujte je. Podle výsledků analýzy navrhněte koncepci skladového systému.
6. Navrženou koncepci realizujte.
7. Zhodnoťte dosažené výsledky a diskutujte možné pokračování v projektu.

Literatura:

- Kanisová, H., Müller, M.: UML srozumitelně. Computer Press, 2004.
- Welling, L., Thomsonová, L.: PHP a MySQL - rozvoj webových aplikací - 2. vydání, SOFTPRESS, 2003.
- Škultéty Rastislav: JavaScript, programujeme internetové aplikace, Computer Press 2001.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 5.

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

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

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

Vedoucí: **Masopust Tomáš, Mgr.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

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

L.S.

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. **Pan**

Jméno a příjmení: **Ladislav Hezoučký**
Id studenta: 84375
Bytem: Bezručova 1391, 544 01 Dvůr Králové nad Labem
Narozen: 08. 09. 1984, Dvůr Králové n. L.
(dále jen "autor")

a

2. **Vysoké učení technické v Brně**

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Skladový systém menší firmy
Vedoucí/školitel VŠKP: Masopust Tomáš, Mgr.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Práce je zaměřena na problematiku skladového systému menší firmy. Cílem je vytvořit funkční systém, který umožní zachytit podstatu operací na skladu (příjmu a výdeje zboží, vystavení objednávek na zboží, vystavení faktur). Systém je implementován pomocí HTML, PHP, JavaScriptu a MySQL.

Klíčová slova

Skladový systém, informační systém, databáze, MySQL, PHP, Javascript, HTML

Abstract

This bachelor's thesis deals with store information system. The main goal of this thesis is to develop a functional information system which will provide to catch the substance of operations in the stock (stock receipt and stock out, draw purchase orders, draw invoice). The system is implemented by using programming languages HTML, PHP and JavaScript. Database MySQL was choosen as a data storage.

Keywords

Store system, information system, database, MySQL, PHP, Javascript, HTML

Citace

Ladislav Hezoučký: Skladový systém menší firmy, bakalářská práce, Brno, FIT VUT v Brně, 2007

Skladový systém menší firmy

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Mgr. Tomáše Masopusta.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Ladislav Hezoučký
15.5.2007

Poděkování

Chtěl bych poděkovat svému vedoucímu práce panu Mgr. Tomáši Masopustovi za jeho rady a odbornou pomoc při řešení mojí bakalářské práce.

Obsah

1 Úvod

Cílem této bakalářské práce je vytvořit informační systém skladu menší firmy. V dnešní době, kdy informace a jejich správné zpracování a interpretace mají čím dál tím větší váhu, vzrůstá důležitost informačních systémů. Bez nadsázky můžeme říci, že se s nimi setkáváme každý den. Pro mnoho firem a institucí se jejich služby staly nepostradatelnými. Například ve školách si s jejich pomocí studenti registrují předměty a mohou nahlížet na podrobnosti k jednotlivým předmětům,jindy v komerční sféře si s jejich pomocí můžeme objednat lístky na své oblíbené divadelní představení. Jak vidíme úkoly informačních systémů mohou být různorodé, od prezentace firmy po přehled v jejím hospodaření. Hlavním úkolem skladového systému je zachytit podstatu operací na skladu (příjmu a výdeje zboží, vystavení objednávek na zboží, vystavení faktur).

V kapitole druhé jsou uvedeny základní informace o technologiích, které byly při implementaci skladového systému použity. Kapitola třetí popisuje požadavky na funkce a role uživatelů výsledného informačního systému. V kapitole čtvrté je popsán postup transformace E-R diagramu na schéma relační databáze. Kapitola pátá se zabývá zajímavými místy implementace systému. V šesté kapitole jsou popsány funkce výsledného informačního systému.

1.1 Definice informačního systému

Informační systém je otevřený systém, používající jako nosič systému konceptuální zdroje, konkrétně informace. Tím rozumíme, že informační systém nenakládá s hmotnými zdroji, ale zdroji nehmotnými. Nakládáním rozumíme provádění transformačních funkcí nad konceptuálními zdroji. Jako nosič systému si můžeme představit např. školu, úřad nebo výrobní podnik, proto se nosič často označuje jako výsek reálného světa nebo svět objektů.

Transformace nad konceptuálními zdroji nečiníme libovolně, ale tyto operace modelují skutečné chování jiného fyzického systému (firmy, školy…). Informační systém na nehmotné úrovni homomorfně modeluje svůj fyzický vzor, pro jehož řízení je obvykle vytvořen. Vzhledem ke skutečnosti, že modelem nikdy nemůžeme postihnout veškeré chování a vlastnosti vzoru, je model pořízen na vhodné úrovni abstrakce pro daný problém.

Aby mohl informační systém modelovat skutečnou činnost fyzického systému a byl otevřený, musí obsahovat dvě základní části:

1. modelovací prostředí, v něm vždy modelujeme data i procesy. Modelovacím prostředí stavu modelu je nejčastěji databáze, protože poskytuje dostatečně kapacitní a zároveň persistentní (tj. v čase přetrvávající) prostředí pro udržování stavu modelu. Modelování je prováděno pomocí některého z typů systému pro řízení báze dat např.:

- relačním modelem (nejčastější řešení),
- objektovým modelem,
- síťovým modelem, případně jinak.

2. komunikační prostředí, které pomáhá zajistit otevřenost systému (vzhledem k okolí). Komunikace probíhá přes kontaktní body, jejichž lokalizace může být libovolná (např. po celé zeměkouli), proto musíme řešit problém připojení. To může být následujících typů:
   - místní,
   - přes počítačovou síť (lokální, Internet),
   - přímou linkou a jiným specializovaným zařízením.

2 Použité technologie

Nyní bych se zmínil o technologiích, které jsou byly při tvorbě informačního systému použity.

2.1 PHP

PHP (Hypertext Preprocessor,původně Personal Home Page) je skriptovací jazyk, používaný hlavně k programování dynamických internetových stránek. Nejčastěji je začleňován do struktury jazyka HTML, XHTML či WML, což je velmi výhodné pro tvorbu internetových aplikací.

PHP skripty jsou prováděny na straně serveru, k uživateli je přenášen pouze výsledek jejich činnosti. Syntaxe jazyka kombinuje hned několik programovacích jazyků (Perl, C, Pascal a Java). Jazyk PHP je nezávislý na platformě, skripty fungují bez úprav na mnoha různých operačních systémech. Součástí jazyka jsou rozsáhlé knihovny funkcí pro zpracování textu, grafiky, práci se soubory, přístup k většině databázových serverů (např. MySql, ODBC, Oracle,...) a podporu řady internetových protokolů (HTTP, SMTP, SNMP, FTP, …)

PHP se stalo velmi oblíbeným jazykem a hlavně díky své jednoduchosti. V kombinaci s databázovým serverem (především MySQL nebo PostrgreSQL) a webovým serverem Apache je často používán k tvorbě internetových aplikací. Díky častému nasazení na serverech se vžila zkratka LAMP – tedy spojení Linux, Apache, MySql a PHP nebo Perl.

2.1.1 Historie

Od roku 1994 se stal jazyk PHP nejčastějším prostředkem pro tvorbu dynamicky generovaných www stránek. Jeho tvůrce Rasmus Lerdorf jej vytvořil pro svou osobní potřebu přepsání z Perlu do jazyka C. Sada skriptů byla vydána ještě v témže roce pod názvem Personal Home Page Tools, zkráceně PHP. Zatím poslední verze je 5.2.1. z roku 2007.

2.1.2 Ukázka kódu

Jako ukázku zde uvádím skript PHP, který vypíše na HTML stránku aktuální datum. PHP kód je oddělen speciálními značkami <? a ?>. Příkaz echo vypisuje text na standardní výstup a příkaz Date vrací aktuální datum.

```
<HTML>
    Dnešní datum je <?echo date('Y\-m\-d')?>
</HTML>
```

Jsou možné následující kombinace HTML a PHP kódu:

- Celý soubor může být jeden dlouhý PHP kód. Pak začíná a končí znaky <? a ?>.
- V souboru může být kód PHP jen n jednom určitém místě.
- V souboru ale může být více míst obsahujících PHP kód.
- Soubor se sice může tvářit jako PHP skript, ale ve skutečnosti je to normální HTML soubor a PHP kód v něm vůbec není obsažen..

2.1.3 Typické vlastnosti jazyka PHP

Jazyk PHP je dynamicky typový, tzn. že datový typ proměnné se určí v okamžiku přiřazení hodnoty. Díky tomu má PHP dva typy porovnání, '==' stejný jako v C, a '===' který platí jen když jsou oba dva výrazy stejného datového typu.

Ukázka k porovnání:

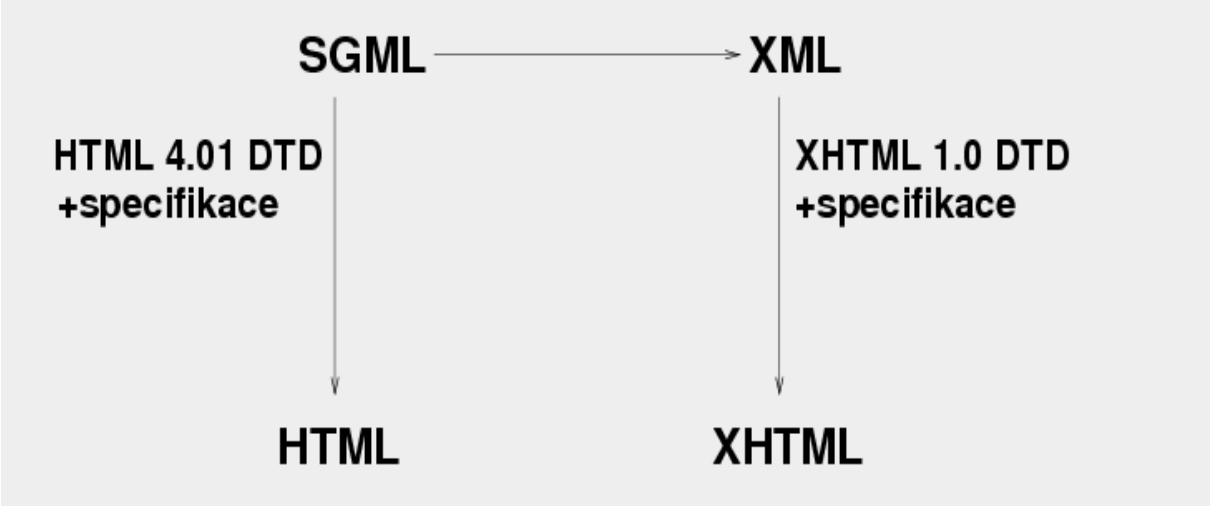
```
// Test porovavani
 $cislo = 5;
 $retezec = '5';

 // Toto porovnani ('==') plati diky automaticke typove konverzi
 if ($retezec == $cislo)
 // Porovnani pomoci '===' neplati, protoze nejsou stejne datove typy
if ($retez === $cislo)
```

Pole jsou heterogenní, mohou tedy obsahovat jakékoli údaje, stejně tak jako jejich indexy. Řetězce lze uzavírat jak do uvozovek (obsah je parsován), tak do apostrofů (obsah není parsován).

2.2 HTML

HTML (Hyper Text Markup Language) je značkovací jazyk pro hypertext. Patří mezi jazyky pro vytváření stránek v systému World Wide Web, umožňujícího publikaci stránek po Internetu.

Jazyk HTML je jednou z vývojových větví dříve vyvinutého univerzálního značkovacího jazyku SGML (Standard Generalized Markup Lanquaqe). Vztah jazyka HTML s jazykem SGML a jeho ostatními vývojovými větvemi vidíme na obrázku 2.1. Další větví jazyka SGML je jazyk XML. Jednou z jeho vývojových větví je jazyk XHTML následník jazyka HTML.



Obrázek 2.1: Vztah SGML, HTML, XML a XHTML

2.2.1 Historie

V roce 1989 pracoval Tim Berners- Lee na propojeném informačním systému pro CERN, výzkumné centrum fyziky poblíž Ženevy ve Švýcarsku. V té době byly dokumenty vytvářeny obvykle pomocí Tex, Poscriptu a SGML. Berners-Lee chtěl vytvořit jednodušší jazyk pro tvorbu dokumentů a proto v roce 1990 navrhl jazyk HTML a protokol pro jeho přenos v síti – HTTP (Hyper Text Transfer Protocol).

V roce 1991 CERN zprovoznil svůj web. Spolu s tím NCSA (National Center for Supercomputer Applications) vybídlo své odborníky k vyvinutí prohlížeče Mosaic. Jeho vývoj byl dokončen v roce 1993, byl určen pro počítače PC a Macintosh. Jednalo se o první prohlížeč s grafickým uživatelským rozhraním a dosáhl obrovského úspěchu. Následoval velký rozvoj webu a nastala potřeba definovat standardy pro jazyk HTML.

Verze jazyka:

- Verze 0.9 – Vydána zhruba v roce 1991. Nepodporuje grafický režim (Tim Berners-Lee)
- Verze 2.0 – Zachycovala stav jazyka v polovině roku 1994. Standard vydala komunita IETF (Internet Engineering Task Force). Je to první verze, která odpovídá syntaxi SGML. Přidává k původní specifikaci interaktivní formuláře, podpora grafiky.
- Verze 3.2 – Vydána 14. ledna 1997 a zachycovala stav jazyka v roce 1996. Připravovaná verze HTML 3.0 nebyla nikdy přijata jako standard pro svoji přílišnou složitost a také proto, že žádná firma nebyla schopná naprogramovat podporu verze do svého prohlížeče. Standard už vydalo konsorcium W3C, stejně jako následující verze. K jazyku byly přidány tabulky, zarovnávání textu a stylové elementy pro ovlivňování vzhledu.

- Verze 4.0 – Vydána 18. prosince 1997. Do specifikace jazyka přidala nové prvky pro tvorbu tabulek, formulářů a nově byly standardizovány rámy (frames). Tato verze je snahou dosáhnout původního účelu - prvky by měly vyznačovat význam (sémantiku) jednotlivých částí dokumentu, vzhled má být ovlivňován připojovanými styly. Některé prezentační elementy byly zavrženy.
- Verze 4.01 –Tato verze opravuje některé chyby verze předchozí a přidává některé nové tagy. Je to poslední verze HTML, které se již dále nevyvíjí, protože má být nahrazeno novějším XHTML, jehož základem je právě tato poslední verze HTML.

2.2.2 Popis jazyka

Jazyk HTML je charakterizován množinou značek a jejich atributů definovaných pro danou verzi. Části textu dokumentu jsou uzavřeny mezi značkami a tím je jim určen význam (sémantika) obsaženého textu. Názvy jednotlivých značek jsou uzavřeny mezi úhlovými závorkami (< a >). Část dokumentu uzavřená mezi značkami tvoří tzv. element (prvek) dokumentu. Součástí obsahu elementu můžou být i další vnořené elementy. Atributy jsou doplňujícími informacemi, kterými upřesňujeme vlastnosti elementu.

2.2.3 HTML značky

Značky (také nazývané tagy) jsou obvykle párové, ve specifikaci XHTML se vyskytují už jen párové značky. Rozlišujeme počáteční a koncové značky. Koncová značka má před názvem značky lomítko. Značky používané v jazyku HTML můžeme podle významu rozdělit do třech základních skupin:

- Strukturální značky – rozvrhují strukturu dokumentu. Patří mezi ně například značky pro odstavec <p>, nadpisy (<h1>,<h2>). Udávají formu dokumentu.
- Popisné (sémantické) značky – popisují povahu obsahu elementu. Mezi tyto značky patří např. <address>, <title>. Současným trendem je orientace na sémantické značky, díky jimž je usnadněno automatizované zpracovávání dokumentů a vyhledávání informací z nich. Výsledkem této snahy je v současné době jazyk XML.
- Stylistické značky – pomocí nich je určen vzhled elementu při zobrazení. Jako příklad můžeme uvést značku pro tučné písmo (<b>). Současným trendem je tento druh značek nepoužívat, místo nich se používá definice vzhledu pomocí kaskádových stylů oddělujících vzhled od obsahu dokumentu.

2.2.4 Budoucnost HTML

Vývoj jazyku HTML byl ukončen jeho verzí 4.01. Dalším pokračováním vývoje jazyků pro popis dokumentů je jazyk XHTML.

Existuje množství důvodů pro tuto změnu. Překotným vývoj prohlížečů umožnil vznik množství dokumentů, které neodpovídají specifikaci HTML. Současné prohlížeče tolerují chybné značkování a mnoho dalších chyb. Tento princip je však příliš náročný pro aplikace vznikající pro malá méně výkonná zařízení – např. mobilní telefony. Tyto aplikace potřebují dokumenty s přesně definovanou strukturou a přesnými pravidly.

Vývoj pokračuje jiným jazykem, ale specifikace HTML platí dál. Je tedy možné podle ní vytvářet nové dokumenty a aplikace všude tam, kde tento jazyk postačuje.

2.3 CSS

CSS (Cascading Style Sheets) je jazyk pro popis zobrazení stránek napsaných v jazycích HTML, XHTML a XML. Jazyk navrhla standardizační organizace W3C. Prozatím byly vydány specifikace CSS1, CSS2 a CSS3.

Hlavní myšlenkou pro vytvoření jazyka bylo umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Byl to důsledek potřeby zobrazovat jeden dokument na více zařízeních, popřípadě více různých druzích prohlížečů, pro tento účel se ukázala definice vzhledu pomocí jazyka HTML nedostatečné.

2.3.1 Výhody CSS

- Rozsáhlejší možnosti – oproti jazyku HTML nabízí CSS rozsáhlejší možnosti formátování. Např. pro formátování bloku textu – určení vzdálenosti od jejich elementu či okraje stránky nemá HTML žádnou vlastnost. Oproti tomu v CSS k tomuto účelu můžeme použít vlastnosti margin a padding.
- Konzistentní styl – Na všech stránkách webové prezentace by měly být všechny nadpisy stejné úrovně, seznamy, zvýrazněné části textu apod. stejného stylu. To je s použitím formátovacích možností jazyka HTML obtížné – u každého objektu v každém dokumentu prezentace bychom museli vzhled objektu znovu nastavovat. S použitím CSS vytvoříme soubor stylu, který připojíme k HTML dokumentu. Ve všech dokumentech s takto připojeným stylem budou objekty stejného vzhledu.
- Oddělení struktury a stylu
- Dynamická práce se styly - provedení změny stylu webu, který by pro formátován vzhledu používal jen možnosti HTML by bylo velmi pracné, museli bychom najít a nahradit všechny

značky, popřípadě změnit jejich atributy. Při používání CSS stačí k změně stylu webu přepsat jediný soubor a to soubor stylů

- Formátování XML dokumentů
- Kratší doba načítání stránky – soubor CSS se uloží do mezipaměti prohlížeče a pokud není změněn, tak se načítá pouze jednou a zobrazení stránek se tím urychlí
- CSS umožňuje definovat různé styly pro různá výstupní zařízení. Tvůrce dokumentu má tak možnost určit jak bude vypadat zobrazení na papíře, v prohlížeči či na PDA.

2.3.2 Nevýhody

Hlavní nevýhodou CSS je zatím stále špatná podpora v majoritních prohlížečích. Různé prohlížeče interpretují stejný CSS kód jinak. V některých přídapech je obtížné vytvořit CSS kód tak, aby se zobrazoval na všech (popř. na vybraných) prohlížečích stejně. Nicméně situace se značně zlepšuje.

2.3.3 Syntaxe

Stylový předpis se skládá z posloupnosti pravidel. Každé pravidlo určuje vzhled elementu či skupiny elementů. Pravidlo začíná selektorem, který specifikuje skupinu elementů. Selektor je následován seznamem deklarací, které určují vzhled vybrané skupiny elementů. Celý seznam je uzavřen ve složených závorkách a jednotlivé deklarace jsou odděleny středníkem (tj. za poslední deklarací středník už být nemusí).

2.3.4 Příklad stylu:

```
h3 {                /* vzhled nadpisu třetí úrovně */
   text-align: center;  /* zarovnaní textu      */
   font-size: 5pt    /* velikost fontu 5 bodů      */
  }
p  {                 /* styl odstavce */
   margin: 10px;       /* okraj šířky 10 pixelů */
   }
```

2.4 Javascript

Javascript je interpretovaný programovací jazyk se základními objektově orientovanými schopnostmi. Jeho autorem je Brendan Eich ze společnosti Netsccape.

Syntaxí je Javascript podobný jazykům C, C++ a Java. S programovacím jazykem Java jej však kromě podobné syntaxe spojuje už jen název, mnohé myšlenky přebírá z jazyka Perl JavaScript

byl v červenci 1997 standardizován asociací ECMA (Europen Computer Manufacturers Association) a v srpnu 1998 ISO (International Standards Organization). Název standardizované verze JavaScriptu je ECMAScript.

V současné době se používá jako interpretovaný programovací jazyk pro www stránky, vkládaný přímo do HTML kódu stránky. S jeho pomocí jsou např. ovládány některé interaktivní prvky GUI (Grafického uživatelského rozhraní) tlačítka,seznamy, textová pole nebo tvořeny efekty obrázků. Programy v JavaScriptu jsou obvykle spouštěny až po stažení www stránky z Internetu (na straně klienta), na rozdíl od jiných interpretovaných jazyků (např. PHP a ASP), které svoji činnost vykonávají na serveru a klient pouze obdrží výsledek.

Kvůli bezpečnosti nepodporuje Javascript na straně klienta práci se soubory a mechanismy pro práci se sítí (umí pracovat s URL a e-mailovými adresami, ale neumožňuje navázat spojení s jakýmkoli jiným hostitelem na sítí).

2.4.1 Ukázka kódu:

```
<script>
 function Validate(Form) {
   if(Form.nove_heslo.value != Form.kontrola_heslo.value){
                    alert("Hesla se museji shodovat !!!");
                    Form.nove_heslo.focus();
                    return false;
                  }
                  else
                  {
    return true;
   }
 }
</script>
```

3 Analýza požadavků

Úkolem bakalářské práce bylo seznámit se s požadavky kladenými na skladový systém menší firmy a navrhnout jeho koncepci a tu pak realizovat. Analýzou jsem dospěl k následujícím požadavkům na skladový systém a jeho funkčnost.

Požadavky na systém:

- Jednotlivým pozicím ve skladovém hospodářství firmy musí odpovídat i role uživatelů
- Při provádění běžných úloh jako příjem či výdej zboží se počty zboží na skladu automaticky aktualizují (uživatel kromě vystavení příjemky nebo výdejky nemusí vykonávat žádnou další činnost)
- Musí umožňovat zobrazení a získávání informací o prodávaném zboží (např. nejprodávanější značka)

Funkce systému:

- Přidávání uživatelů a jejich rolí
- Pohledy podle rolí uživatelů
- Vystavení faktury na objednané zboží (export do formátu PDF)
- Evidence skladového zboží (např. změna ceny)
- Vyhledávání ve zboží na skladě
- Evidenci dodavatelů a odběratelů
- Vystavování dokladů o příjmu a výdeji zboží
- Zobrazení statistik o prodávaném zboží (např. nejprodávanější značka, počet prodaných kusů jednotlivého zboží)
- Přihlašování uživatelů pomocí uživatelského hesla a jména
- Automatické odhlášení uživatelů po určité době nečinnosti

Role uživatelů:

- Administrátor
- Skladník
- Ekonom

Na základě analýzy požadavků jsem vytvořil následující use case diagramy pro jednotlivé role uživatelů viz. obr. 3.1,3.2 a 3.3. Ukazují jednotlivé případy užití, které jsou možné pro dané role.

Případy užití:

- pro roli ekonoma

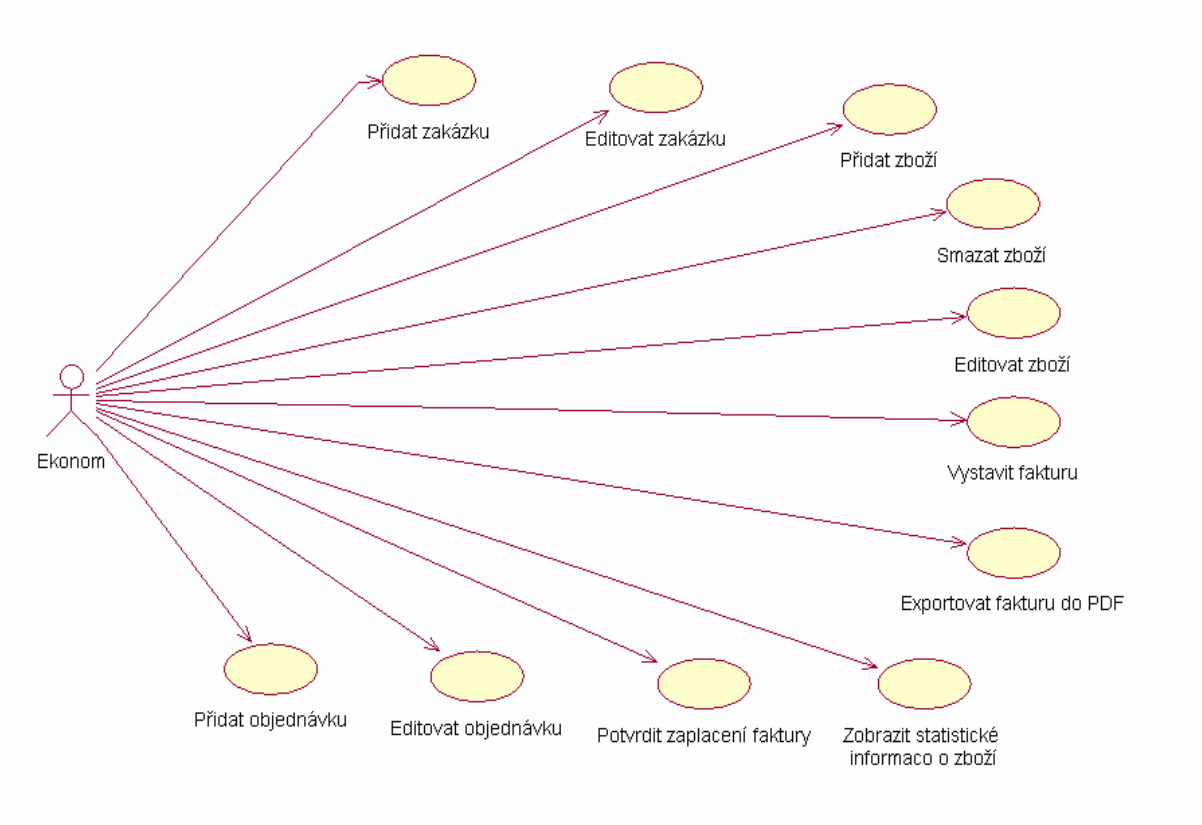
  Ekonom by měl mít možnost plně pracovat s objednávkami a zakázkami (přidávání, mazání, editace). Na zpracovanou zakázku může vystavit fakturu. Na základě účetních dokladů označuje faktury za již zaplacené. Vystavenou fakturu může exportovat do formátu PDF. Poslední možnou činností ekonoma je správa zboží a zobrazení statistik jeho prodeje.

- pro roli skladník

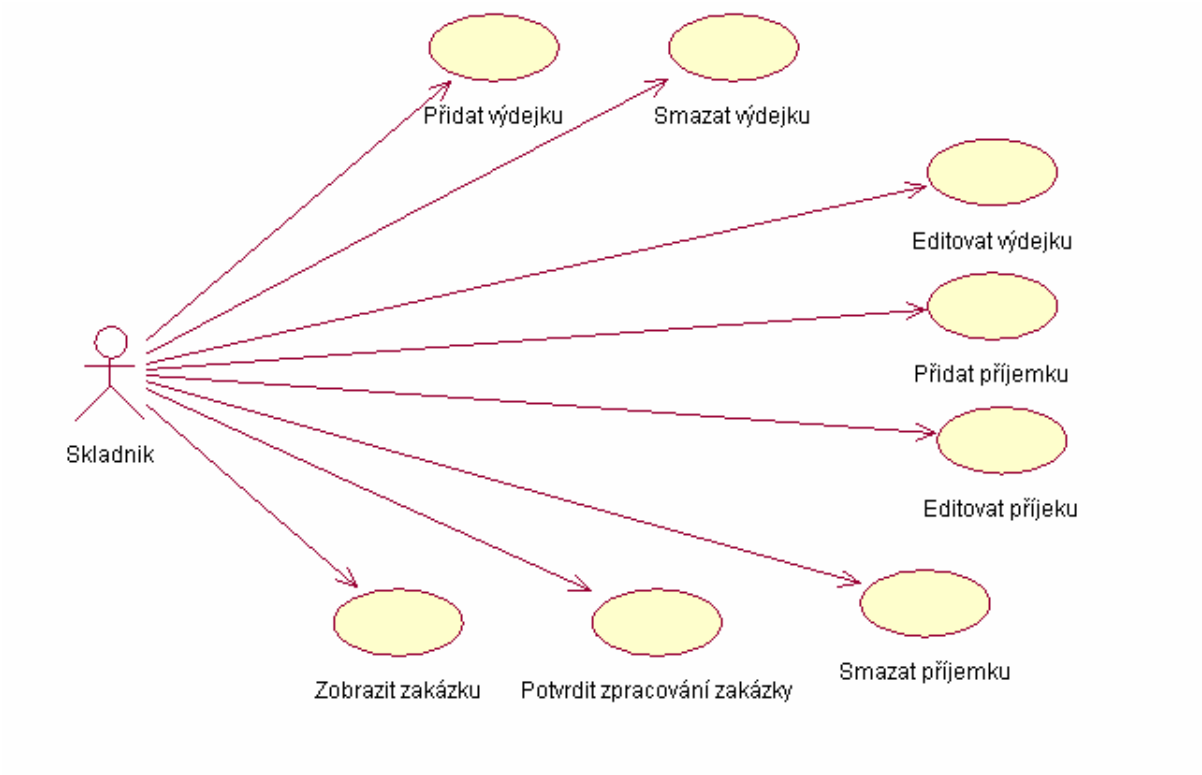
  Hlavní náplní práce uživatele s rolí skladníka je příjem a výdej zboží. Může tedy vydávat, mazat a editovat příjemky a výdejky. Má možnost nechat si zobrazit zboží na zakázce, aby věděl, jaké zboží má vydat ze skladu. Po vydání tohoto zboží může označit zakázku za zpracovanou.

- pro roli administrátora

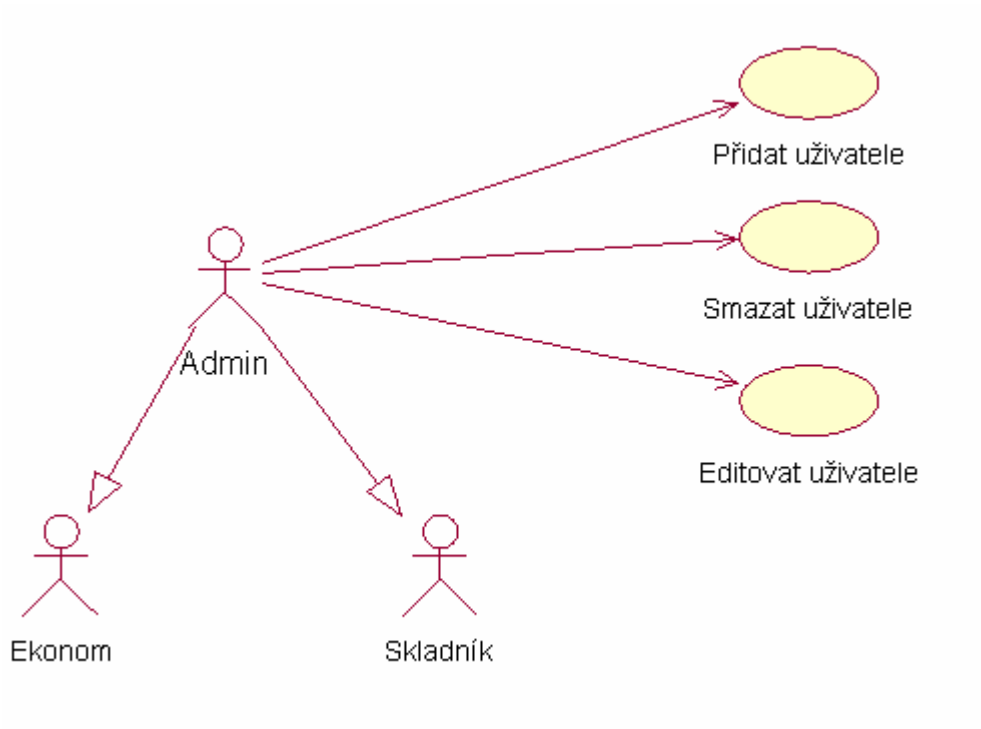
  Uživatel s rolí administrátora může mimo správy uživatelů provádět i operace uživatelů rolí skladník a ekonom.



Obrázek 3.1: Use Case diagram pro roli Ekonom



Obrázek 3.2: Use Case diagram pro roli skladník



Obrázek 3.3: Use Case diagram pro roli admin

4 Vytvoření tabulek relační databáze

Po analýze požadavků jsem přešel k prvnímu kroku implementace informačního systému a to převodu E-R diagramu na tabulky relační databáze.

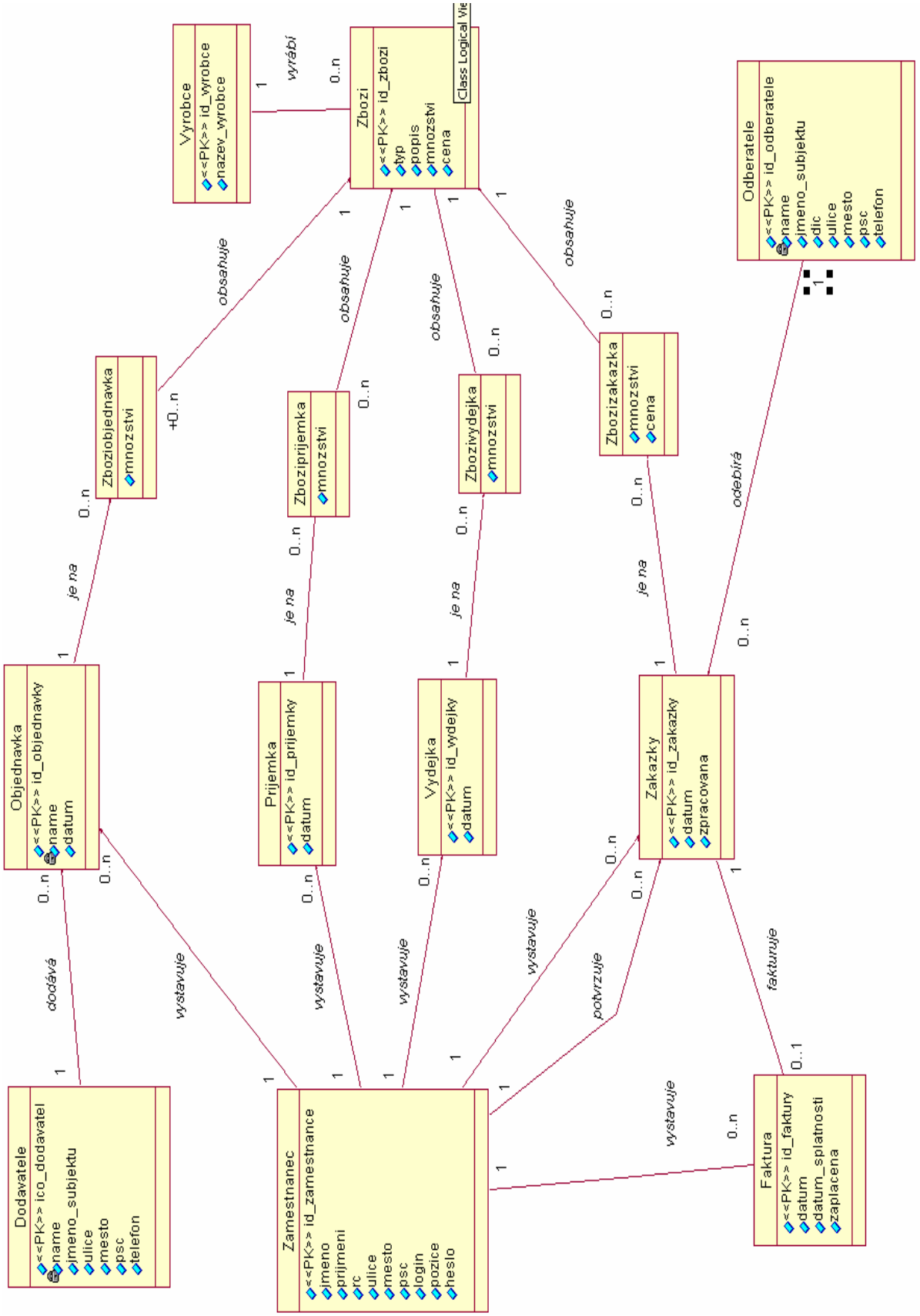
4.1 Konceptuální modelování

Konceptuální modelování (v databázové terminologii také někdy označované jako konceptuální návrh) patří do etapy analýzy požadavků. Jeho cílem je analyzovat požadavky na data, která budou uložena v databázi. Je založeno na objektech (konceptech) aplikační domény, pro kterou softwarový systém vyvíjíme. V tomto případě, kdy vytváříme skladový systém, budeme modelovat data prezentující zaměstnance, zboží, vnitropodnikové doklady a vztahy mezi nimi. Někdy je tato úroveň modelování označována také jako sémantické modelování.

Výsledkem konceptuálního modelování (návrhu) je konceptuální model. Ten je obvykle vytvořen pomocí tzv. entitně-vztahového (entity-relationship) modelu, který se postupně od svého zavedení v polovině 70. let stal základním modelem reprezentujícím požadavky na data uložená v databázi, z něhož se vychází návrh tabulek relační databáze. Prezentujeme ho zpravidla v podobě ER diagramu. Jedná se o model, který je představitelem modelovacích technik strukturované analýzy. Podobnou roli plní pro objektově-orientované analýzy diagram tříd.

Nyní si vysvětleme rozdíl mezi relačním modelem dat a konceptuálním modelem. Relační model dat popisuje obecnou strukturu, integritní omezení a operace nad daty v relační databázi bez ohledu na konkrétní aplikační doménu. Na rozdíl od konceptuálního modelu, který modeluje data konkrétní aplikační domény, v našem případě skladu.

Na základě analýzy požadavků jsem vytvořil E-R diagram viz obr. 4-4. Jak z něj můžeme vyčíst,skladový systém umožňuje vystavovat pět různých dokladů na zboží (příjemku, výdejku, objednávku, zakázku a fakturu). Každý z dokladů nese záznam o uživateli, který ho vystavil. Aby na dokladu mohlo být uvedeno více jak jeden druh zboží, je mezi každou entitou představující doklad a entitou zboží vazební tabulka M:M, která toto umožňuje. Diagram dále obsahuje entity Odberatele a Dodavatele, které prezentují právnické nebo fyzické osoby, které obchodují s naší firmou. Dodavatelé se vztahují k objednávkám a odběratelé k zakázkám. Entita zbozi představuje zboží uložené na skladu.



Obrázek 4.4: *E-R diagram*

4.2 Převod E-R diagramu na schéma relační databáze

Pravidla pro transformaci E-R diagramu na schéma relační databáze:

- **Odstranění složených a vícehodnotových atributů**

  Jedná se o přípravný krok, který zajistí, že následná transformace povede na normalizované tabulky. V případě složeného atributu je úprava jednoduchá. Spočívá v rozložení atributu na jednotlivé jeho položky (např. atribut adresa bychom rozložili na ulici, město a jeho PSČ). U vícehodnotového atributu máme dvě možnosti úpravy. Můžeme omezit počet hodnot (např. každý zaměstnanec bude mít pouze tři možná telefonní čísla) nebo zavedeme novou entitní množinu.

- **Transformace silných entitních množin**

  Pokud se tyto množiny nevyskytují ve struktuře generalizace/specializace transformaci provádíme následujícím způsobem. Každé silné entitním množině odpovídá tabulka navrhované databáze. Sloupce tabulky odpovídají atributům entitní množiny, identifikátor entitní množiny bude primárním klíčem tabulky. Jednotlivé řádky tabulky představují jednotlivé entity entitní množiny.

- **Transformace vztahových množin**

  Pravidla pro transformaci vztahových množin.

  1. Vztahy s kardinalitou 1:1 reprezentujeme v databázi rozšířením jedné z tabulek pro entity, mezi kterými je transformovaný vztah, o sloupec, který je cizím klíčem odkazujícím se na primární klíč druhé tabulky. Pokud má transformovaná vztahová množina nějaké atributy, rozšíříme tabulku s cizím klíčem také o sloupce odpovídající těmto atributům.
  2. Vztahy s kardinalitou 1:M reprezentujeme v databázi rozšířením tabulky pro entitu, která má na svém konci transformovaného vztahu kardinalitu M, o sloupec, který je cizím klíčem odkazujícím se na primární klíč druhé tabulky. Má-li transformovaná vztahová množina nějaké atributy, rozšíříme tabulku s cizím klíčem také o sloupce odpovídající těmto atributům.
  3. Vztahy s kardinalitou M:M reprezentujeme v databázi tak, že přidáme novou tabulku, která bude obsahovat jako cizí klíče primární klíče tabulek reprezentujících entitní množiny, mezi kterými je transformovaný vztah, a případné sloupce atributů vztahové množiny. Primární klíč této nové tabulky bude složený a bude tvořen primárními klíči obou tabulek a případně ještě jedním nebo více atributy vztahové množiny, pokud samotná kombinace hodnot primárních klíčů není v tabulce unikátní.

4.3 Normalizace databáze

Mezi hlavní problémy chybného návrhu patří zejména:

- opakující se informace (redundance)
- nemožnost reprezentovat určitou informaci
- složitá kontrola integritních omezení

Všem třem nedostatkům se můžeme vyvarovat pomocí procesu normalizace. V extrémním případě bychom mohli provést kompletní logický návrh pouze použitím normalizace. Takový postup by byl při návrhu rozsáhlých databází značně nepraktický. V praxi normalizaci používáme nejčastěji ke kontrole kvality návrhu a úpravám při zjištění nedostatků schématu databáze navrženého transformací konceptuálního diagramu. Normalizace výrazně pomáhá vyvarovat se chyb logického návrhu relační databáze, ať už je prováděn transformací z konceptuálního modelu nebo intuitivně přímo při návrhu jednoduchých databází.

Myšlenka normalizace schématu relačních databází se opírá o dva základní koncepty:

1. Hierarchii tzv. normálních forem, které odrážejí kvalitu logického návrhu, schématu relace z hlediska výše uvedených nedostatků.
2. Postupnou dekompozici schématu relace, které vykazuje nedostatky.

Nejčastěji se uvádí šest normálních forem. Každá z nich definuje vlastnosti, které relace musí mít, aby normální formu splňovala. První tři (1NF, 2NF a 3NF) a postup zvaný normalizace zavedl již na začátku 70. let minulého století E.F.Codd. Ten později přidal další normální formu, která se označuje jako Boyce-Coddova (BCNF). Ještě později vznikly 4NF, 5NF. Těchto šest normálních forem tvoří hierarchii v pořadí, jak byly uvedeny. Znamená to, že má-li relace splňovat k-tou normální formu, musí splňovat nejen podmínky k normální formy, ale i k-1 normální formy.

4.4 Tabulky vzniklé převodem E-R diagramu na schéma relační databáze

Název tabulky	Popis
zamestnanec	Obsahuje údaje o zaměstnancích
dodavatele	Informace o dodavatelích skladu
odberatele	Informace o odběratelích skladu
vyrobce	Seznam výrobců zboží
zbozi	Informace o zboží na skladu
zakazky	Obsahuje informace o zakázkách skladu
zbozizakazka	Informace o druhu a počtu zboží na zakázce
faktura	Obsahuje informace o vystavených fakturách
vydejka	Informace o výdejce zboží ze skladu
zbozivydejka	Obsahuje druh a množství zboží na výdejce
prijemka	Informace o příjemce zboží na sklad
zboziprijemka	Obsahuje druh a množství zboží na příjemce
objednavka	Informace o objednávce na zboží
zboziobjednavka	Obsahuje druh a množství zboží na objednavce

Tabulka 4.1: Tabulky vzniklé převodem E-R diagramu.

5 Implementace

V této části bych se zmínil o zajímavostech z hlediska implementace. Jako technologie pro implementaci informačního systému byly zvoleny následující HTML, PHP a databázový systém MySql (o těchto technologiích jsem se podrobněji zmiňoval v kapitole 2).

5.1 Kontrola hodnot v polích formulářů

Hodnoty v polích formulářů pro zadávání údajů například o novém zaměstnanci můžeme kontrolovat už v prohlížeči klienta pomocí Javascriptu před odesláním na server. Umožňují nám to regulární výrazy pomocí nichž můžeme definovat očekávanou podobu zadávaných údajů.

Příklad definice regulárního výrazu v Javascriptu:

```
re = new RegExp("^([A-Za-z ]{2,30})$");
if (!re.test(Form.jmeno.value))
{
  alert("Jmeno neni dobre zadano.");
    Form.jmeno.focus;
 return false;
}
```

Vytvořili jsme regulární výraz, který kontroluje, zda textové pole obsahuje řetězec o délce 2-30 znaků složený z velkých a malých písmen od a-z. Pokud zadané pole obsahuje jiné hodnoty vypíše na obrazovku varování a umístí kurzor do tohoto pole. Jako výsledek funkce na kontrolu hodnot formuláře vratí hodnotu false, čímž zabrání odeslání formuláře.

5.2 Export faktury do formátu PDF

Jedním z problémů, který jsem musel ve své práci řešil byl export faktury do formátu PDF.
PDF (Portable Dokument Format – Formát pro přenositelné dokumenty) je souborový formát, který byl vyvinut firmou Adobe pro ukládání dokumentů nezávisle na softwaru i hardwaru, na kterém byly vytvořeny. Soubory typu PDF mohou obsahovat text i obrázky, díky vlastnostem formátu je zajištěno, že se libovolný dokument zobrazí na všech zařízeních stejně. Existuje mnoho volně dostupných prohlížečů pro různé platformy, zřejmě nejznámějším je oficiální prohlížeč Adobe Reader.

Pro implementaci exportu do PDF jsem použil volně šiřitelnou a poměrně propracovanou knihovnu fpdf pro jazyk PHP.

Ukázka generování dokumentu v PDF

```
<?
define('FPDF_FONTPATH','./font/');
require('./fpdf.php');

$pdf=new FPDF();
$pdf->Open();
$pdf->AddPage();
$pdf->SetFont('Arial','B',10);
$pdf->Cell(30,8,'Ahoj');
$pdf->Output();
?>
```

Pro generování textu potřebujeme mít nastaven nějaký font, proto jako první ukládáme do konstanty FPDF_FONTPATH cestu k adresáři s fonty. Pokud by nám standardní fonty nestačily je možnost vytvořit si svoje vlastní. Na druhém řádku skriptu je použit příkaz require pro vložení knihovny FPDF. Na dalším řádku vytvoříme objekt $pdf. Pomocí metody Open () založíme nový dokument. Abychom mohli vkládat text, potřebujeme do dokumentu přidat stránku. To provedeme pomocí metody AddPage(). Pomocí metody SetFont nastavíme vlastnosti fontu pro námi vkládaný text – Arial, Bold a velikost 10 bodů. K vytvoření textu použijeme metodu Cell(). Jejím užitím vytvoříme textový objekt 30x8 a do něj vložíme text „Ahoj“. Pomocí metody Output() dokončíme tvorbu PDF dokumentu a odešleme výsledek prohlížeči.

Nyní krátce k metodám Cell a Output:

Cell(float w [, float h [, string txt [, mixed border [, int ln [, string align [, int fill [, mixed link]]]]]]])

Slouží k vytvoření textového objektu.

Parametry:

- povinný “w”- šířka textového objektu
- nepovinný “h” – výška textového objektu
- nepovinný “txt”- text vkládaný do textového objektu

- nepovinný “border”- určuje zda bude viditelný okraj. 0 = bez okraje, 1 = s okrajem. Také je možnost vytvořit okraj pouze určité strany objektu: “L” = levý, “T” = horní, “R”= pravý a “B” = dolní. Ohraničení okrajů lze mezi sebou kombinovat.
- nepovinný “ln” – určuje pozici kam bude vložen další objektu. 0 = vpravo, 1 = na začátek následujícího řádku, 2 = pod tento objekt
- nepovinný “align” – nastavuje zarovnání textu. “L” vlevo, “C” na střed, “R” vpravo
- nepovinný “fill” – určuje zda je použíta (1) nebo nepoužita (0) barva pozadí
- nepovinný "link" - URL nebo identifikátor vrácený metodou AddLink().

Output([string file [, boolean download]])

Odešle vygenerované PDF prohlížeči.

Parametry:

- nepovinný “file” – určuje jméno souboru. Vyplňujeme jen v případě, pokud budeme chtít uložit soubor na serveru nebo ho odeslat klientovi ke stažení.
- nepovinný "download" určuje zda bude PDF dokument uložen na serveru (false) nebo u klienta (true).

Příklady užití:

```
$pdf->Output("faktura.pdf",true) - odešle klientovi soubor faktura.pdf.
$pdf->Output("faktura.pdf") - uloží na serveru soubor faktura.pdf.
$pdf->Output() - prohlížeč načte plug-in a zobrazí obsah dokumentu PDF.
```

6 Skladový systém

Do skladového systému se uživatel musí přihlásit pomocí svého loginu a hesla, existují tři druhy uživatelů a tím tedy i práv k tomuto informačnímu systému.

Informační systém obsahuje tyto tři role uživatelů:

- Admin
- Ekonom
- Skladník

6.1 Přihlášení do systému

K této části systému má volný přístup každý uživatel lokální sítě. Přihlášení probíhá pomocí formuláře na hlavní stránce systému. Data odeslaná tímto formulářem jsou na serveru ověřena porovnáním s uživatelskými daty uživatele uloženými v databázi. Pokud se jedná o ověřený přístup je uživateli umožněno přihlášení do informačního systému a důležitá data uživatele jsou uložena do proměnných typu session.

V session proměnných je uloženo identifikační číslo zaměstnance, druh jeho role a čas přihlášení. Tímto máme zaručenu jednoznačnou identifikaci uživatele po celou dobu jeho práce v systému. Při odhlášení ze systému jsou všechny tyto proměnné odstraněny.

6.2 Správa uživatelů

K této části systému má přístup pouze uživatel s oprávněním admina.

6.2.1 Přidání nového uživatele

Pro přidání nového uživatele musí admin zadat jeho jméno, příjmení, rodné číslo, adresu, login a druh role (oprávnění uživatele). Všechny vyjmenované položky jsou povinné. Heslo je při vytvoření uživatele nastaveno na jeho rodné číslo a je mu doporučeno, aby si ho při prvním přihlášení do systému změnil. Všechna pole jsou při odeslání formuláře testována a pokud nejsou správně vyplněna, formulář se neodešle na server.

Jsou-li všechna pole řádně vyplněna a dojde k odeslání formuláře na server, následuje test na přítomnost loginu v databázi. Pokud již databáze login obsahuje, je na to admin na tuto skutečnost upozorněn a umožníme mu vrátit se na formulář pro zadávání údajů o uživateli. Když databáze login neobsahuje, je do ní vložen záznam o novém uživateli..

6.2.2 Editace a mazání uživatelů

Editovat lze všechny parametry uživatele s výjimkou id_zamestnance. Jedná se totiž o primární klíč tabulky. Při mazání uživatelů jsme upozorněni pomocí okna Javascriptu, zda opravdu chceme uživatele smazat.

6.3 Správa zakázek a faktur

K zakázkám mají plný nebo částečný přístup všichni uživatelé informačního systému. Uživatelé s právy administrátora mají plný přístup ke všem možnostem práce se zakázkami. Ekonomové mohou zakázky mazat, editovat nebo přidávat. Dále je jim umožněna editace zboží na zakázce, nemohou však označit zakázku za zpracovanou. Skladníkům je umožněno nahlédnout na obsah zakázky, aby mohli vydat ze skladu určené zboží. Mohou také označit zakázku za zpracovanou. K sekci faktury mají přístup pouze administrátoři a ekonomové.

Uživatel s právy administrátora nebo ekonoma může zakázky přidávat, mazat a editovat všechny jejich atributy vyjma id_zakazky, které je primárním klíčem tabulky. Při přidávání zboží na zakázku kontrolujeme, zda se již dané zboží na zakázce nenachází. Nachází-li se námi přidávané zboží na zakázce, jsme na tuto skutečnost upozorněni a vráceni zpět k formuláři na zadávání zboží na zakázku. Pokud je zakázka zpracována, může na ní ekonom pomocí tlačítka vystavit fakturu. Po jeho stisknutí se do tabulky faktura vloží nový záznam s aktuálním datem, obsahující id_zamestnance, který fakturu vystavil a je do něj uloženo id_zakazky jako cizí klíč.

Faktury můžeme mazat, ale tato činnost není moc pravděpodobná. Snad jen při nečekaných změnách v zakázce, která už byla zpracována a vystavena na ni faktura. Ekonom může na základě účetních dokladů označit fakturu za zaplacenou. Poslední možnou operací je export faktury do formátu PDF. Technické řešení bylo popsáno v předchozí kapitole. Zaměřme se tedy co faktura obsahuje po formální stránce.

Na faktuře jsou uvedeny všechny následující náležitosti:

- Obchodní jméno, sídlo nebo místo podnikání, popř. bydliště nebo místo podnikání plátce, který uskutečňuje zdanitelné plnění.
- Daňové identifikační číslo plátce, který uskutečňuje zdanitelné plnění.
- Obchodní jméno, sídlo nebo místo podnikání, popř. bydliště nebo místo podnikání plátce, pro něhož se uskutečňuje zdanitelné plnění.
- Daňové identifikační číslo plátce, pro něhož se uskutečňuje zdanitelné plnění.
- Pořadové číslo dokladu.
- Rozsah a předmět zdanitelného plnění.

- Datum vystavení dokladu.
- Datum uskutečnění zdanitelného plnění.
- Výši ceny bez daně celkem.
- Základní nebo sníženou sazbu daně.

6.4 Správa odběratelů a dodavatelů

K této části informačního systému mají přístup admin a ekonom. Identifikace dodavatelů je prováděna pomocí jejich identifikačního čísla (IČ), to je každému ekonomickému subjektu přiřazeno Českým statistickým úřadem. Předpokládáme, že od nás mohou odebírat zboží i menší živnostníci. U nich je IČ tvořeno jiným způsobem než u právnických osob. Proto v tabulce odběratele v primárním klíči tabulky id_odberatele je uloženo u fyzické osoby její rodné číslo a u právnické osoby její IČ.

Při vytváření nového dodavatele/odběratele je kontrolována správnost všech zadaných polí v formuláři na předpokládaný tvar adresy, telefonního čísla a názvu. Na serveru je testováno, zda se již zadané ico_dodavatele/id_odberatele v databázi nachází. Pokud ano, je uživatel na tuto skutečnost upozorněn a vyzván k návratu k zadávacímu formuláři. Jinak je přidán nový záznam do tabulky dodavatele/odberatele.

6.5 Správa zboží

V této části Informačního systému mohou uživatelé s oprávněním administrátora nebo ekonoma přidávat, měnit nebo mazat jednotlivé druhy zboží na skladu. Součástí každého záznamu o zboží je jeho identifikační číslo, výrobce a typ zboží, jeho cena bez daně a množství na skladě. V ideálním případě bychom začínali se stavem 0 v položce množství u jednotlivých druhů zboží a veškerý úbytek nebo nárůst množství by se konal pomocí výdejek nebo příjemek. Ale systém může být zaveden do již fungující firmy, začínalo by se tedy od stavu z nejbližší inventury. A po každé další inventuře by se počet zboží na skladu aktualizoval podle skutečností zjištěných inventurou.

Systém umožňuje zboží mazat a editovat všechny jeho záznamy mimo atributu id_zbozi, které je primárním klíčem tabulky. V této sekci informačního systému si můžeme nechat zobrazit statistiku zboží. Máme na výber zobrazení nejprodávanějšího zboží nebo značky. A to za určité období, které zvolíme výberem určitého časového intervalu, nebo za dobu existence skladu.

6.6 Správa výdejek a příjemek

S touto sekcí mohou pracovat uživatelé s právy administrátora a skladníka. Výdejky a příjemky jsou obrazem výdeje a příjmu zboží na sklad. Pokud vystavíme výdejku resp. příjemku na určité množství zboží, změní se i počet zboží ve skladu, v informačním systému vyjádřené položkou množství

u jednotlivých druhů zboží v tabulce zbozi.

Příjemky a výdejky mohou být vytvářeny, editovány nebo mazány. Smazat výdejku nebo příjemku však lze pouze v případě, že neobsahuje žádné zboží. Můžeme také na výdejku nebo příjejku přidávat zboží a jeho množství. Při této operaci je kontrolováno, zda se již dané zboží na výdejce, příjemce nenachází. Pokud ano, je přidán nový záznam do tabulky zbozinavydejce/zbozinaprijemce. Množství zboží na příjemce/výdejce lze zpětně editovat.

6.7 Změna hesla

K této sekci má přístup každý z uživatelů. Umožňuje jim změnit si svoje heslo pro přihlášení do systému. Při vytvoření nového uživatele je mu automaticky heslo nastaveno na jeho rodné číslo, proto je mu doporučeno si ho při prvním přihlášení do systému změnit. Heslo by mělo být 5-20 znaků dlouhé a může obsahovat kombinaci číslic, malých a velkých písmen abecedy bez diakritiky.

Při odesílání vyplněného formuláře o změně hesla nejprve kontrolujeme na straně klienta, zda jsou všechny položky formuláře vyplněny a zda jejich obsah formálně souhlasí s požadavky na heslo a nové heslo souhlasí s jeho opakovaným zadáním. Na straně serveru kontrolujeme, zda se staré heslo shoduje s heslem uloženým v databázi pro uživatele. Pokud souhlasí je do databáze uloženo nové heslo.

Přesněji je v databázi uložen pouze otisk hesla. Jedná se o heslo zakódované pomocí algoritmu md5. Abychom zabránili odposlechu pravého hesla, jsou data z formuláře zasílána pomocí tohoto algoritmu. Zvyšujeme tím celkovou bezpečnost systému, pokud uživatel zapomene heslo, nejsme mu schopni původní heslo nijak sdělit. Administrátor pouze může jeho heslo nastavit opět na rodné číslo uživatele.

7 Závěr

Vytvoření skladového systému se skládalo ze dvou hlavních částí. První byla analýza požadavků na výsledný systém a následné vytvoření E-R diagramu na základě této analýzy. Ten byl poté převeden na schéma relační databáze. Druhou částí bylo pomocí vybraných technologií implementovat požadovanou funkčnost systému. Výsledný informační systém má všechny požadované vlastnosti a funkce. Systém se momentálně nachází v první verzi, která slouží především jako testovací.

Většinu funkcí systému se povedlo otestovat vcelku dobře, ale při nasazení systému do reálného provozu by s ním mohly současně pracovat desítky uživatelů. Tato skutečnost se spolu se zatížením databáze předem těžko testuje.

Celkově byla pro mně bakalářská práce velmi přínosná. Vyzkoušel jsem si realizaci většího projektu s daným termínem dokončení a také jsem se důkladně seznámil s vybranými technologiemi po praktické i teoretické stránce.

7.1 Možná rozšíření systému

Jedním z rozšíření funkcí systém by mohlo být větší množství podporovaných formátů pro export faktury, například do formátu určeného pro některý z účetních programů. Systém by mohl být propojen s čtecím zařízením na čárové kódy. Po vytvoření výdejky, popřípadě příjemky, by zboží bylo na tyto doklady přidáváno po načtení svého čárového kódu.

Další možné vylepšení bych viděl v přidání atributu zakázka do tabulek příjemka a výdejka. Tento atribut by nesl informaci, ke které zakázce nebo objednávce se daný doklad vztahuje.

Literatura

[1] Hruška, T. *Informační systémy*. Studijní opora.Brno, FIT VUT v Brně, 2006.

[2] HUGH E., Williams, DAVID, Lane. *PHP a MySQL*. [s.l.] : [s.n.], 2002. 530 s.

[3] *Linux Software* [online]. c2003-2007 [cit. 2007-05-08]. Dostupný z WWW: <http://www.linuxsoft.cz/>.

[4] *World Wide Web Consortium* [online]. c1994-2007 [cit. 2007-05-08]. Dostupný z WWW: <http://www.w3.org/>.

[5] DAVID, Flanagan. *Javascript : Kompletní průvodce*. [s.l.] : [s.n.], 2002. 825 s.

[6] HANA, Kanisová, MIROSLAV, Müller. *UML srozumitelně*. [s.l.] : [s.n.], 2006. 170 s.

[7] Zendulka, J. *Databázové systémy*. Studijní opora. Brno, FIT VUT v Brně, 2006.

[8] *Webguru* [online]. c2001-2003 [cit. 2007-05-08]. Dostupný z WWW: <http://webguru.cz/>.

Seznam příloh

Příloha 1. Ukázka výsledné faktury

Faktura - Danový doklad: 1

Dodavatel:		*Odberatel:*	
BOXS s.r.o.		**Balirny**	
Zahradni 45/2		Bezrucova	
Brno 61200		Dvur	
ICO: 23698569	DIC: CZ23698569	ICO: 7896541236	DIC: CZ7896541236
Banka:	Komercni banka - Brno	Konstantni symbol:	0008
Ucet:	275698741236/0100	Variabilni symbol:	1
Tel.: 523 693 852	web: www.boxs.cz	Popis:	Objednavka zbozi.
Zapsana do OR: MS v Brne, B 5623			
Datum UZP:	1		
Datum vystveni:	1		
Datum splatnosti:	1		

Kod	Nazev zbozi	Mnozsvi	Netto/MJ	Netto	Dan	Celkem brutto
2	Asus A35	5	30000	150000	19%	178500
3	Benq FP73G	3	5100	15300	19%	18207
4	Nokia 3210	10	2000	20000	19%	23800
1	Samsung E12	5	2000	10000	19%	11900

	Netto	DPH	Brutto
Zakladni sazba DPH 19%	195300	37107	232407
Celkem	195300	37107	232407