VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

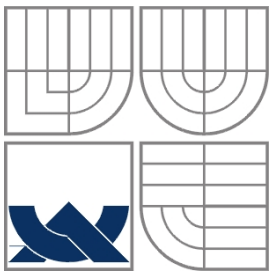
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ÚČETNÍ SYSTÉM

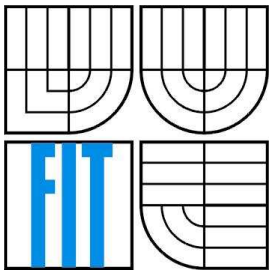
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE TOMÁŠ KAPIČÁK
AUTHOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ÚČETNÍ SYSTÉM
ACCOUNTING SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ KAPIČÁK
AUTHOR

VEDOUCÍ PRÁCE — ING. OTA JIRÁK
SUPERVISOR

BRNO 2010

Abstrakt

Práce se zabývá návrhem a tvorbou informačního systému pro jednoduché účetnictví. V první části je probrána problematika informačního systému a funkce jednoduchého účetnictví. Následující kapitoly se zabývají návrhem a implementací systému. Implementace je provedena v jazycích PHP, XHTML, CSS s využitím Zend Frameworku a různých knihoven. Pro uložení dat je využita databáze MySQL.

Abstract

This work deals with the design and creation of an information system for simple accounting. The first part discussed the issue of information system and the function of simple accounting. The next chapters deal with design and implementation of the system. Implementation is done in PHP, XHTML, CSS using Zend Framework and various libraries. MySQL database is used for data storage.

Klíčová slova

Informační systém, elektronické účetnictví, jednoduché účetnictví, faktura, peněžní deník, framework, PHP, MySQL, Zend, Ajax, MVC systém

Keywords

Information system, electronic accounting, and simple accounting, invoice, cash book, framework, PHP, MySQL, Zend, Ajax, MVC system

Citace

Kapičák Tomáš: Účetní systém, bakalářská práce, Brno, FIT VUT v Brně, 2010

Účetní systém

Prohlášení

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

……
Tomáš Kapičák
19.5.2010

Poděkování

Tímto bych se chtěl poděkovat mému vedoucímu Ing. Otovi Jirákovi za odborné vedení, cenné rady, konzultace, připomínky, které mi přispěli k dokončení bakalářské práce. Poděkovat chci také kamarádovi Bc. Dušanovi Kmeťovi za cenné rady při využívaní Zend Frameworku. Moje poděkování patří také Ing. Kataríne Eľkovej za informace o jednoduchém účetnictví. Děkuji také Mgr. Márii Malákovej za jazykovou korekturu této práce. V neposlední řadě děkuji své rodině a hlavně rodičům za podporu, kterou mi dodávali.

© Tomáš Kapičák, 2010
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů..

Obsah

Obsah 1
1 Úvod 3
2 Účtovný informačný systém 4
2.1 Informačný systém 4
2.2 Jednoduché účtovníctvo 5
2.2.1 Účtovné doklady 5
2.2.2 Peňažný denník 6
2.3 Možnosti a využitie účtovného systému 6
3 Analýza požiadaviek a návrh 8
3.1 Špecifikácia požiadaviek 8
3.1.1 Viac užívateľov 8
3.1.2 GUI šablóny 8
3.1.3 Prijaté / vydané faktúry 8
3.1.4 Tlač dokumentov 9
3.1.5 Adresár 9
3.1.6 Pohyby na účtoch a pokladni, XML import 9
3.1.7 Vyhľadávanie s využitím autocomplete 10
3.1.8 Párovanie platieb s dokladmi 10
3.1.9 Peňažný denník 11
3.1.10 Rozšíriteľnosť pomocou modulov 11
3.2 Diagram prípadov užitia 11
3.3 ER diagram 13
3.4 Grafické rozhranie 16
3.5 Použité technológie 16
3.5.1 Skriptovací jazyk PHP 16
3.5.2 Databázový server MySQL 17
3.5.3 Zend Framework 17
3.5.4 Javascriptová knižnica jQuery 18
3.5.5 XML 19
3.5.6 XHTML 19
3.5.7 Kaskádové stýly CSS 19
3.5.8 PDF a trieda mPDF 19
4 Implementácia 20
4.1 Tvorba grafického návrhu a XHTML šablóny 20
4.2 Vytvorenie databázy 21
4.3 Inštalácia a nastavenie Zend Frameworku 21
4.4 Práca s MVC architektúrou 23
4.4.1 Models 23
4.4.2 Views 23
4.4.3 Controllers 24
4.5 Funkcie účtovného systému s využitým Zend Frameworku 24
4.5.1 Registrácia a prihlásenie užívateľov 25
4.5.2 Vkladanie, editovanie, vymazávanie záznamov 25
4.5.3 Import dát z XML 26
4.5.4 Zobrazovanie zoznamov s filtrovaním 26

4.5.5 Vytváranie faktúr s generovaním do PDF 27
4.5.6 Párovanie platieb 28
4.5.7 Generovanie peňažného denníka 28
4.5.8 Stav účtovníctva podnikateľa 29
4.5.9 Modulové pridanie grafického vzhľad 29
4.5.10 Ostatné doplnkové funkcie 29
5 Testovanie a vylepšenia 30
5.1 Priebeh testovania 30
5.2 Navrhované vylepšenia účtovného systému 31
6 Záver 32
Literatúra 33
Zoznam príloh 34

1 Úvod

Internetový predaj či internetové bankovníctvo je dnes samozrejmosťou. Je to určitá činnosť, ktorá má určité pravidlá a dá sa zautomatizovať a tým šetriť zdroje. Práve tento fakt ma viedol k hľadaniu ďalších možností ako využiť médium Internetu na podobnú činnosť. Prečo nie je zautomatizované aj účtovníctvo? Táto otázka viedla k tvorbe bakalárskej práce – vytvoriť informačný systém, ktorý bude spolu s generovaním dokumentov aj informovať podnikateľa o jeho aktuálnom stave podnikania online.

Popis informačného systému spolu s charakteristikou jednoduchého účtovníctva je popísaný v druhej kapitole. Podstatné je využitie v praxi u podnikateľov, ktoré popisujem na konci kapitoly.

Tretia kapitola sa zaoberá detailnou analýzou požiadaviek a návrhom. Nájdeme tam špecifikáciu vlastností, ktoré účtovný systém má obsahovať spolu s návrhmi funkčnosti. Táto kapitola popisuje návrh grafického vzhľadu systému s využitím GUI šablón. Taktiež tu nájdeme charakteristiku technológií a programovacích jazykov vhodných použiť pri implementácií.

Štvrtá kapitola popisuje riešenie implementácie s využitím Zend Frameworku. Tvorba začínala tvorbou grafického návrhu a pokračovala vytvorením XHTML šablóny. Následne sa na základe ER diagramu vytvárala databáza. Zend Frameworku bolo potrebné nainštalovať, aby mohol byť využívaný pri tvorbe potrebných funkcií systému, ktorých popis nájdeme na konci tejto kapitoly.

Piata kapitola sa zaoberá testovaním potrebným na odchytenie chýb systému a následným ladením. Taktiež tu nájdeme nápady na vylepšenia, ktoré je možné vyriešiť rozšírením systému.

Šiesta záverečná kapitola obsahuje zhodnotenie dosiahnutých výsledkov. Nájdeme v nej zhodnotenie vytváraného projektu s jeho prínosom.

2 Účtovný informačný systém

Účtovný informačný systém má slúžiť na generovanie dokumentov a štatistík potrebných na vedenie jednoduchého účtovníctva. Pre tvorbu takéhoto projektu je potrebné charakterizovať informačný systém a jednoduché účtovníctvo. Vhodné je zvoliť vhodné programovacie jazyky a technológie, aby systém bol vytvorený, čo najelegantnejšie.

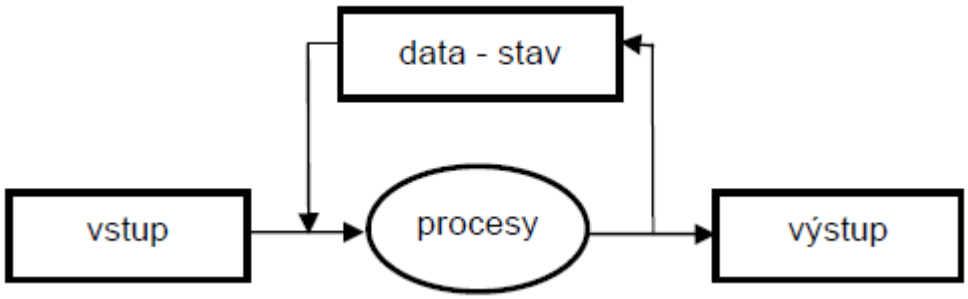
2.1 Informačný systém

Informačný systém[1] označovaný ako IS je systém na zber, udržiavanie, spracovanie a poskytovanie informácií. Skladá sa z dvoch slov a to zo slova informácia a systém. Informácia má viac definícii, ale v jednoduchosti ide o dáta, s ktorými budeme pracovať. V našom prípade informáciou budú dáta o platbách. Systém je možné chápať ako množinu prvkov a ich väzieb medzi sebou, ktoré musia byť definované.

Informačný systém vykonáva nasledujúce činnosti:

- Získavanie informácie.
- Spracovanie informácie.
- Využívanie informácie.
- Uchovávanie informácie.
- Prenos informácie.

Na Obr. 2-1 je znázornená všeobecná schéma informačného systému. Skladá sa z vstupnej a výstupnej časti, odkiaľ sa do systému vkladajú, resp. sa získavajú informácie. Medzi vstupom a výstupom prebieha transformácia týchto dát. Transformačná časť typicky prevádza nad dátami určité operácie – algoritmy. Prebiehajú tu procesy a musíme sa zaoberať spôsobom definícií procesov, ich vzájomnou interakciou, paralelným prevádzaním a pod.



Obr. 2-1 Schéma informačného systému [1]

Typickou časťou informačného systému je spätná väzba, ktorá využíva stav systému. Nie všetky výstupy procesov sú priamo závislé na okamžitých vstupoch, ale typicky závisia na stave systému. Stav systému uchovávajú dáta. Procesy realizujú transformácie často vo forme transakcií.

V súčasnosti chápeme IS ako integrovaný celok hardware, software a dáta, kde významnú úlohu hrá personál, ktorý s IS pracuje.

2.2 Jednoduché účtovníctvo

Jednoduché účtovníctvo[2] podnikateľa zisťuje a zachytáva celý priebeh jednotlivých hospodárskych činností a ich ekonomické výsledky. Účtovníctvo poskytuje podnikateľovi – fyzickej osobe, informácie o:

- Stave, prírastkoch a úbytkoch majetku.
- Príjmoch z podnikateľskej činnosti.
- Výdavkoch vynaložených na dosiahnutie, zabezpečenie a udržanie príjmov.
- Výsledku hospodárenia, ktoré dosiahol za určité časové obdobie, tzn. zisk alebo stratu.

Jednoduché účtovníctvo fyzickej osoby na území Slovenskej republiky sa riadi určitými pravidlami tzv. najvyššími právnymi normami:

- Obchodný zákonník, t. j. zákon č. 513/1991 Zb. v znení neskorších predpisov, ktorý ustanovuje podnikateľom viesť účtovníctvo v rozsahu a spôsobom stanoveným v zákone o účtovníctve.
- Zákon č. 431/2002 Z. z. o účtovníctve v znení neskorších predpisov, ktorý sa týka všetkých účtovných jednotiek v SR, ktoré účtujú v sústave jednoduchého a podvojného účtovníctva.
- Všeobecne záväznou právnou normou pre vedenie JÚ, ktorá doplňuje ZÚ, sú Postupy účtovania v sústave JÚ.

Tieto pravidlá jednoduchého účtovníctva sú zákony, ktoré sú platné určitú dlhšiu dobu. Je možné ich spracovať do automatického systému. V takomto automatizovanom systéme odpadá nutnosť spravovať účtovníctvo účtovníčkou. Ide teda o šetrenie zdrojov. Je potrebné vytvoriť systém tak, aby akceptoval prípadné zmeny v zákonoch alebo vytvorenie nových.

Pre väčšinu malých podnikateľov, ktorí majú len jedno prevádzkové miesto, jeden účet v banke, nenarábajú s ceninami ani s peňažnými prostriedkami v cudzej mene, stačí peňažný denník na úplné a správne účtovanie o stave a pohybe peňažných prostriedkov v hotovosti, na bankovom účte a na ceste.

2.2.1 Účtovné doklady

Podľa § 10 zákona č. 431/2002 Z. z. o účtovníctve v znení neskorších predpisov účtovný doklad je preukázateľný účtovný záznam, ktorý musí obsahovať:

- Označenie účtovného dokladu.
- Obsah účtovného prípadu a označenie jeho účastníkov.
- Peňažnú sumu alebo údaj o cene za mernú jednotku a vyjadrenie množstva.
- Dátum vyhotovenia účtovného dokladu.
- Dátum uskutočnenia účtovného prípadu, ak nie je zhodný s dátumom vyhotovenia.
- Podpisový záznam osoby zodpovednej za účtovný prípad v účtovnej jednotke a podpisový záznam osoby zodpovednej za jeho zaúčtovanie.
- Označenie účtov, na ktorých sa účtovný prípad zaúčtuje v účtovných jednotkách účtujúcich v sústave podvojného účtovníctva, ak to nevyplýva z programového vybavenia.

Pod podpisovým záznamom sa rozumie účtovný záznam, ktorého obsahom je vlastnoručný podpis alebo obdobný preukázateľný účtovný záznam nahrádzajúci vlastnoručný podpis v technickej podobe (napr. v elektronickej forme).

Účtovná jednotka je povinná vyhotoviť účtovný doklad bez zbytočného odkladu po zistení skutočnosti, ktorá sa ním preukazuje, a to tak, aby bolo možno určiť obsah každého jednotlivého účtovného prípadu.

Existuje množstvo účtovných dokladov, najčastejšie sa vyskytujúcimi účtovnými dokladmi sú hlavne:

- Faktúra.
- Výpis z bankového účtu.
- Príjmový pokladničný doklad,
- Výdavkový pokladničný doklad.
- Príjemka.
- Výdajka.

Interné účtovné doklady si vyhotoví účtovná jednotka podľa svojich podmienok, nie je presne stanovený formulár, musia však byť v účtovnom doklade všetky náležitosti podľa zákona o účtovníctve. Informácie a v tejto kapitole sú voľne prevzaté z [2].

2.2.2 Peňažný denník

Peňažný denník[2] je v sústave jednoduchého účtovníctva účtovnou knihou, v ktorej sa zachytáva pohyb peňažných prostriedkov v hotovosti (na základe pokladničných dokladov) a na účtoch v bankách (na základe výpisu z bankového účtu), príjmy a výdaje v požadovanom členení, predovšetkým pre daňové účely.

Zápisy v peňažnom denníku sa uskutočňujú v časovom slede na základe účtovných dokladov. Peňažný denník musí obsahovať minimálne:

- Prehľad o peniazoch v hotovosti v členení na príjmy a výdavky.
- Prehľad o peniazoch na bankových účtoch v členení na príjmy a výdavky.
- Prehľad o priebežných položkách v členení na príjmy a výdavky. Vznikajú pri prevodoch peňažných prostriedkoch medzi bankovými účtami alebo pokladňou a bankovým účtom.
- Prehľad o príjmoch, ktoré súvisia s podnikateľskou činnosťou a sú zahrňované do základu dane (napr. príjmy z predaja výrobkov, tovaru, služieb).
- Prehľad o výdavkoch, ktoré súvisia s podnikateľskou činnosťou a zahrňujú sa do základu dane (napr. nákup materiálu, energie, vyplatenie miezd zamestnancom, zákonné poistenie do zdravotnej a sociálnej poisťovne (§ 19 zákona č. 595/2003 Z. z. o dani z príjmov v znení neskorších predpisov).
- Prehľad o ostatných príjmoch, ktoré neovplyvňujú daňový základ (napr. DPH zaplatené odberateľmi, dotácie od úradu práce).
- Prehľad o ostatných výdavkoch, ktoré nesmú ovplyvniť daňový základ. (napr. peňažné dary, manká a škody presahujúce prijaté náhrady, výdavky na osobnú spotrebu podnikateľa).

2.3 Možnosti a využitie účtovného systému

Keďže na Slovensku pracuje značný počet ľudí na živnosť, potrebujú títo podnikatelia spravovať svoje účtovníctvo. Viacero takýchto podnikateľov vykazuje do roka iba 12 faktúr, mesačne teda jednu ako svoju výplatu. Ich náklady sú minimálne, väčšinou majú len jeden bankový účet, kde prijímajú svoje platby. Každý takýto podnikateľ, ak nie je zdatný v účtovníctve, musí kontaktovať účtovníka, aby mu účtovníctvo za určitý poplatok spracoval.

Možnosťou spracovať účtovníctvo účtovným systémom euctovnicka.sk poskytneme daným podnikateľom možnosť výberu. Spracovaním účtovníctva na internete majú účtovníctvo kedykoľvek k dispozícii. Môžu z akéhokoľvek miesta pripojeného na internete vytvárať a zasielať faktúry, či ich iba prezerať. Adresár kontaktov si môžu pozrieť napríklad aj v mobile, a tak mať údaje o svojich obchodných partneroch vždy poruke. Navyše ak majú iba príjmy a náklady im platí zamestnávateľ,

vytvorením faktúr môžu mať hneď spracované celé účtovníctvo na celý rok. Systém je možné využívať na vytváranie faktúr, prípadne na vytváranie si svojho adresára odberateľov.

Systém je možné rozvinúť do rozsiahleho portálu, kde v budúcnosti všetci podnikatelia budú zasielať svoje doklady interne v danom systéme a tak nebude môcť dochádzať k napríklad nezaúčtovaniu platieb faktúr, daňovým únikom a podobne. Verím, že takýto globálny účtovný systém na nás čaká a je iba otázkou času, kedy budeme na to pripravení a ochotní ho prijať.

3 Analýza požiadaviek a návrh

Pre správne navrhnutie systému je vhodné analyzovať požiadavky na systém. Je nutné vedieť, čo chceme dosiahnuť, preto si rozoberieme základnú špecifikáciu požiadaviek na funkčnosť systému a následne navrhneme spôsob riešenia.

3.1 Špecifikácia požiadaviek

Projekt má zadané požiadavky, ktoré má splňovať. V týchto podkapitolách sa na ne bližšie pozrieme. Okrem tých požiadaviek budú v systéme implementované aj iné požiadavky, tie však nie sú pre beh portálu prioritné.

3.1.1 Viac užívateľov

Účtovný informačný systém má spracovávať účtovníctvo viacerým podnikateľom. Je preto potrebné vytvoriť registráciu daného podnikateľa, ktorého účtovníctvo bude system spracovávať. Je potrebné zadať správne základné údaje ako prihlasovacie meno, ktoré bude e-mail podnikateľa, heslo pre prihlásenie, názov firmy, fakturačná adresa pre potreby vystavovania faktúr, IČO a DIČ ako identifikátor podnikateľa. Je potrebné dáta validovať. Po úspešnej registrácií sa je možne prihlásiť a dostať sa do systému konkrétneho účtovníctva podnikateľa. Do tejto časti sa dá dostať iba po správnom zadaní svojho prihlasovacieho mena a hesla.

3.1.2 GUI šablóny

Účtovný informačný systém má slúžiť na generovanie dokumentov a štatistík potrebných na vedenie. Webová stránka má mať možnosť zmeny svojho vzhľadu. Návštevník stránky môže meniť farebnosť stránky podľa vlastného uváženia a chuti. Je potrebné vytvoriť dostatok farebných variácií. Výber grafického vzhľadu by mal byť uchovaný aj po zatvorený stránky. Na túto činnosť sú vhodné Sessions, nakoľko sa ukladajú na strane servera a uchovávajú sa aj po zatvorení prehľadávača.

3.1.3 Prijaté / vydané faktúry

V účtovnom systéme je potrebné vytvoriť možnosť vytvárania faktúr. Údaje na faktúre o dodávateľovi sú čerpané z registračných údajov užívateľa. Údaje o odberateľovi je potrebné zadať. Adresa sídla odberateľa a predmet fakturácie spolu so sumou sú údaje vyžadované. Bez nich nie je možné faktúru vytvoriť. Ostatné údaje ako IČO, DIČ, poštová adresa a e-mail sú nepovinné. Ak sú však zadané, je potrebné ich validovať. Pre vytvorenie faktúry je potrebné pridať do nastavení podnikateľa naskenovanú pečiatku s podpisom. Bez nej nie je faktúra platná. Je taktiež potrebné vyplniť bankové účty v nastaveniach, ak je možné platbu faktúry uhradiť na bankový účet.

Číslovanie faktúr je potrebne vytvárať postupne. Užívateľ si môže zadať číslo faktúry ľubovoľné podľa svojho systému číslovania faktúr. Ak číslo faktúry nezadá, systém mu automaticky priradí číslo o jedno vyššie ako je číslo naposledy vytvorenej faktúry. Sú definované pred pripravené číslovania vo formáte „YYYY0000”, kde YYYY je aktuálny rok. Variabilný symbol je možne zadať. Ak sa nezadá, bude variabilný symbol rovnaký ako číslo faktúry. Systém následne vytvorí faktúru, ktorú si je možne prezrieť vo formáte HTML alebo PDF. Systém musí faktúru spracovať ako príjem, keďže faktúry sa vystavujú iba za vykonané služby alebo predávaný tovar.

Faktúru je možne odoslať na e-mail odberateľa. Pokiaľ sa vyplní korektne e-mail odberateľa, bude mu na e-mail zaslaná PDF faktúra. Faktúra je automaticky zasielaná aj na e-mail dodávateľa.

Faktúry je potrebné aj vkladať do systému, ak už boli vytvorené v inom systéme ako euctovnicka.sk. Vkladajú sa iba informácie o faktúrach, nevytvárajú sa. Prijaté faktúry sú faktúry, ktoré vytvárajú náklady. Vystavené faktúry predstavujú tržby – príjem. Pri vkladaní informácií o príjmoch a nákladoch nie je potrebné rozpisovať údaje tak ako pri vytváraní faktúr. Pre účely účtovníctva stačia údaje ako číslo dokladu, celková suma, dátum dodania služby, prípadne variabilný symbol. Príjmy a náklady si je možne prezerať v prehľadoch.

3.1.4 Tlač dokumentov

Každú faktúru je vhodné vytlačiť. Najideálnejšie je faktúru vytlačiť do súboru PDF. Súbor PDF sa generuje pomocou triedy *mPDF*, spomínanú v predchádzajúcej kapitole. Tlačiť je možne aj HTML verziu. V zozname vystavených faktúr je možne zvoliť si možnosť vytlačenia z HTML s následnou automatickou výzvou na tlač alebo vytlačenia faktúry do PDF súboru. Dokumenty sa na server neukladajú, nakoľko by mohli k nim mat prístup neoprávnení užívatelia. Preto sa vždy generujú až po overení práv užívateľa.

3.1.5 Adresár

Adresár je zoznam kontaktov, ktoré je možné využívať pri generovaní faktúr. Do adresára sa vkladajú údaje o odberateľoch podnikateľa. Povinný je názov kontaktu ako identifikátor kontaktu a sídlo odberateľa, pre vytváranie faktúr. Ďalšie pomocné údaje ako poštová adresa, IČO, DIČ, IČ-DPH, e-mail, čísla bankových účtov, poznámka ku kontaktu sú voliteľné. Pre rýchlejšiu prácu s kontaktmi je možne nastaviť kontaktu vlastnosť zobrazovania v bočnom menu na každej podstránke. Označením tejto možnosti sa bude kontakt zobrazovať na každej podstránke a tak prístup ku kontaktu bude okamžitý, taktiež aj vytváranie faktúr bude rýchlejšie. Zoznam kontaktov je možne prezerať cez filtrované vyhľadávanie s autocomplete funkciou.

3.1.6 Pohyby na účtoch a pokladni, XML import

Podnikateľ môže prijímať platby prevodom na bankový účet alebo príjmom v hotovosti, teda do pokladne. Rovnakým spôsobom môže prevádzať finančné prostriedky za náklady. Tieto údaje je potrebné do systému zadať, nakoľko od práve týchto prijatých alebo odoslaných platieb sa vykazuje účtovníctvo za konkrétny rok.

Pre pridávanie pohybov na bankovom účte je potrebne bankový účet pridať do účtovníctva podnikateľa v nastaveniach. Pre pridávanie dát o pohyboch sú povinné údaje variabilný symbol, celková cena, dátum prijatia platby a typ platby (príjem alebo výdavok). Pri pohyboch na bankových účtoch sa ukladajú informácie aj o účtoch odosielateľa a ak podnikateľ má viac bankových účtov, tak si volí na ktorý účet sa pohyb vzťahuje. Pri hotovostných platbách sa zadáva informácia o tom, komu sa platba platila, resp. kto ju uhrádzal. Všetky dáta zadané vo formulári je potrebné validovať, podobne ako pri validovaní iných odosielaných údajov cez formulár.

Internetové bankovníctvo prináša možnosť generovania výpisov z účtov aj vo formáte XML. Tento dokument je vhodné využiť na vkladanie dát o pohyboch na účte. Jednoduchým vygenerovaním XML pohybov na účtu v Tatra banke a následným importom do účtovného systému sa vložia pohyby do účtovníctva podnikateľa. Pokiaľ podnikateľ importuje XML súbor s dátami o pohyboch na jeho bankovom účte a daný bankový účet nie je pridaný v nastaveniach, systém ho pridá automaticky za neho. Podnikateľ, ktorý ma inú banku, alebo nemá internet banking môže

pohyby vkladať ručne cez formulár. Pri vkladaní XML súboru do systému je potrebné overovať, či je súbor vôbec XML správny.

3.1.7 Vyhľadávanie s využitím autocomplete

Všetky záznamy o vytvorených, vystavených či prijatých faktúrach, záznamy o nákladoch a príjmoch, o pohyboch na bankových účtoch alebo pokladni budú zobrazované v prehľadných tabuľkách. V týchto záznamoch je potrebné vyhľadávať. Užívateľ bude môcť vyhľadávať pomocou rozšíreného vyhľadávania, kde môže vyhľadávať cez rôzne vlastnosti položiek. Nie je obmedzený nutnosťou pamätať si číslo dokladu pri dokumentoch, ale môže hľadať na základe PSČ dodávateľa, sumy pohybu alebo napríklad variabilného symbolu. Systém mu ponúka možnosť zobraziť iba tie záznamy, ktorých dátumy sa nachádzajú v nastaviteľnom časom intervale.

Navyše pri vyhľadávaní bude napomáhať našepkávací systém autocomplete. Tento systém automaticky kontroluje hodnoty v záznamoch, ktorých časť návštevník zadáva a dáva mu možnosť výberu z už existujúcich hodnôt. Táto funkcia bude riešená pomocou pluginu jQuery, ktorý sa priamo pýta konkrétnej tabuľky na existujúce hodnoty.

Záznamy budú rozdelené po stránkach. Pomocou ďalšieho pluginu jQuery je možné radiť priamo v HTML tabuľke zobrazovane údaje. Tento plugin sa volá Tablesorter a teší sa veľkej obľube užívateľov. Zoraďuje riadky v tabuľke abecedne alebo číselne podľa zvoleného stĺpca.

3.1.8 Párovanie platieb s dokladmi

Doklady sú pridávané cez vystavené / prijaté faktúry, resp. cez príjmy / náklady. Platby sú evidované buď v pokladni alebo na bankovom účte. Každý doklad má mať svoj pohyb. Opačne to však už neplatí, nakoľko pri jednoduchom účtovníctve je možné využívať na účely podnikania aj osobný účet, kde môžu byť evidované aj položky osobnej spotreby. Položky osobnej spotreby netvoria základ dane, preto nie je potrebné ich dokladovať. Doklady je však vhodné previazať s platbami, aby bolo zrejmé, ktorý doklad bol kedy zaplatený a do ktorého účtovného roku patrí a hlavne, že je finančný pohyb dokladovaný dokladom.

Každá platba je prvotne charakterizovaná ako osobná spotreba, teda neovplyvňuje základ dane. Až po previazaní sa stáva položkou v podnikaní, ktorá môže ovplyvniť základ dane. Pri párovaní platieb platí jedno pravidlo, že výška sumy pohybu nemôže byť menšia ako suma na doklade. Vyššia samozrejme môže byť a rozdiel sa berie ako príjem neovplyvňujúci základ dane (tzv. díško). V opačnom prípade nie je doklad plne uhradený.

Každý doklad a pohyb je možné previazať. Pri párovaní je potrebné nájsť odpovedajúce dva záznamy, ktoré k sebe patria - doklad a finančný pohyb. Vhodné je opäť našepkávať záznamy, ktoré sa hodia k previazaniu. Našepkávač vyberá záznamy, ktoré majú dátum prijatia platby alebo vystavenia v rozmedzí 15 dní od smerovaného dátumu párovaného záznamu a zároveň hľadá rovnakú výšku sumy v oboch záznamoch. Samozrejme berie ohľad na typ platby. Je možne previazať iba kreditné pohyby s príjmom a debetné pohyby s nákladmi. Našepkávač funguje ako filter, ktorý filtruje zo záznamov, ktoré previazané nie sú. Tento filter je možné meniť a tak nájsť aj iné záznamy ako našepkávač ponúkne, ak je to potrebné. Po nájdení správneho odpovedajúceho záznamu sa môžu záznamy previazať. Pri párovaní sa nastavuje aj zaradenie finančného pohybu, ktorý sa využije pri generovaní peňažného denníka.

3.1.9 Peňažný denník

Každý finančný pohyb na účtoch podnikateľa a v pokladni musí byť zaevidovaný v peňažnom denníku. Aj osobná spotreba, teda nespárovaný finančný pohyb, ale aj spárovaná platba s dokladom. Peňažný denník obsahuje viacero stĺpcov popísaných v kapitole 2.2.2.

Keďže každá nespárovaná platba je osobná spotreba (bez spárovaného dokladu nemôžeme nič dokladovať), je potrebné iba spárované záznamy zaradzovať do konkrétnych stĺpcov peňažného denníka. Ak ide o bankový pohyb, bude hodnota pohybu na účte v stĺpci bankové účty a to buď ako príjem alebo výdavok, podľa hodnoty pohybu. Prirodzene je záporná hodnota (debet) výdavok a kladná hodnota (kredit) príjem. Následne je potrebné zaradiť pohyb ako položku ovplyvňujúcu alebo neovplyvňujúcu základ dane a výber prípadne aj konkretizovať zámerom. Ak pohyb je na pokladni, pohyb sa zaznamená v stĺpci hotovosť a následné sa tiež učí ovplyvniteľnosť základu dane a prípadne aj spresní zámer. Toto priradzovanie sa vykonáva spolu s párovaním platieb s dokladmi. Peňažný denník prakticky iba spracováva dáta vložene v systéme, neupravuje ich.

3.1.10 Rozšíriteľnosť pomocou modulov

Systém je vhodné rozširovať a v prípade, že bude projekt využívaný bude potrebné vytvárať moduly, ktoré sa budú do systému implementovať. Pre začiatok demonštrujem modelové využívanie na rozšíriteľnosti grafických šablón webu. Najvhodnejším spôsobom realizácie bude vloženie nahranie ZIP súboru s kaskádovými štýlmi a obrázkami na server. ZIP súbor sa rozbalí do adresára, ktorý bude aktívny a bude automaticky rozoznávať novo pridané moduly a automaticky ich bude načítavať. Vkladanie bude možné v nastavení.

3.2 Diagram prípadov užitia

Diagram prípadov užitia (Use Case Diagram) znázorňuje chovanie systému z pohľadu používateľa. Definuje systém vzťahy s okolím. Cieľom je podať určitý obraz navrhovanej funkčnosti systému.

Na Obr. 3-1 je znázornený diagram prípadov užitia pre mnou vytváraný systém. Aktérmi modelu sú:

- Visitor (neprihlásený návštevník stránky).
- User (registrovaný a prihlásený).

Visitor je neprihlásený návštevník webu, ktorý nie je prihlásený v systéme. Jeho možnosti sú prezerať informačné a prezentačné stránky, meniť grafickú farebnosť webu, registrovať sa a následne sa prihlásiť. Po úspešnom prihlásení sa stáva z Visitora aktér User.

User má možnosť taktiež vyberať farebnosť webu spoločne s Visitorom, navyše však môže pracovať s účtovným systémom, ktorý mu prihlásením identifikoval jeho profil. Môže vystavovať faktúry svojim klientom, pridávať informácie o nákladoch a príjmoch, pridávať informácie o finančných pohyboch v pokladni a bankových účtoch a taktiež má možnosť importu pohybov z bankové účtu. Ďalšími funkciami prihláseného aktéra sú párovanie platieb s finančnými pohybmi, nastavovať aktuálny pracovný účtovný rok, prezeranie štatistík o nákladoch a príjmoch v roku účtovníctva a generovanie peňažného denníka a taktiež vytváranie vlastného adresára kontaktov svojich klientov. Samozrejmosťou je možnosť odhlásenia zo systému.

Obr. 3-1 Diagram prípadov užitia (Use Case Diagram)

3.3 ER diagram

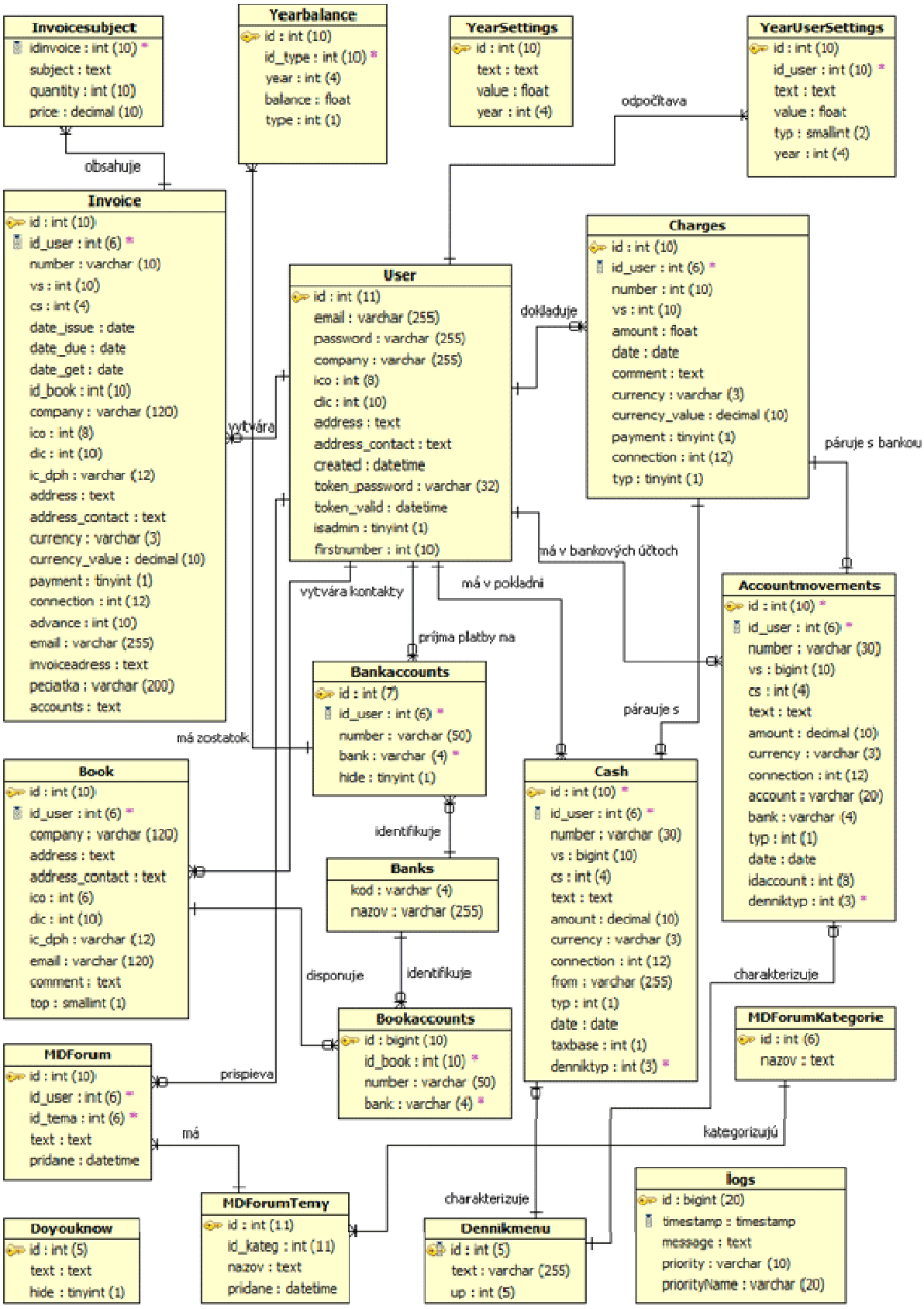
Informačný systém bude pracovať s databázou. Pre jej správne navrhnutie je vhodné si navrhnúť ER diagram, ktorý popisuje jej ontológiu. ER diagram projektu je zobrazený na Obr. 3-2. ER diagram je rozsiahly. Obsahuje všetky využívane tabuľky. Presnejší rozbor ER diagramu pomôže k jednoduchému vytvoreniu databázy. ER diagram tvoria tieto hlavné entity:

- **User** je podstatná entita systému, tabuľka, ktorá reprezentuje užívateľa – podnikateľa. Obsahuje primárny kľuč s názvom *id*, ktorý jednoznačne určuje užívateľa. Pomocou tohto atribútu identifikujem všetky atribúty prislúchajúce tomuto užívateľovi. Ostatné atribúty reprezentujú vlastnosti užívateľa ako napríklad jeho e-mailovu adresu – *email*, heslo – *password* alebo adresu sídla – *address*.
- **Charges** je tabuľka dokladov. Reprezentuje informácie o príjmoch a nákladoch. Primárny kluč je *id*. Cudzím kľúčom je *id_user*, cez ktorý je previazaná tabuľka User. Údaje o dokladoch sú reprezentovane atribútmi ako napríklad *number* – číslo dokladu, *amount* výška sumy uvedená na doklade a *date* – dátum dodania služby. Hodnota v atribúte *typ* reprezentuje príjem pre hodnotu 0 a pre hodnotu 1 je riadok v tabuľke nákladom. Pomocou *connection* na párovanie s finančnými pohybmi v Cash alebo Accountmovements. Entita User je vo vzťahu s entitou Charges s kardinalitou 1:0..n.
- **Cash** reprezentuje tabuľku informácií o pohyboch v pokladni. Primárny kľuč je *id*. Cudzí kľuč je id*_user* a *connection*. S entitou User je táto entita vo vzťahu s kardinalitou 1:0..n. Hodnota atribútu *typ* informuje o tom, či je platba kreditná alebo debetná. Hodnota pohybu *amount* je vždy kladné číslo. Tabuľka obsahuje okrem atribútov informujúcich o pohyboch na pokladni aj atribúty *denniktyp* – informácia o zaradení pohybu v peňažnom denníku, *connection* – prepojenie na Charges, ktoré pre zjednodušenie a urýchlenie otázok na server sú vkladané aj do atribútov reprezentujúcich finančné pohyby. Medzi entitami Charges a Cash je kardinalita 1:0..1.
- **Accountmovements** je atribút, ktorý reprezentuje pohyby na bankových účtoch. Kredity aj debety. Hodnota atribútu *typ* informuje o tom, či je platba kreditná alebo debetná. Hodnota pohybu *amount* je vždy kladné číslo. Pomocou cudzieho kľúča *connection* je párovaný pohyb s dokladom v Charges. *Denniktyp* reprezentuje informáciu o zaradení pohybu v peňažnom denníku. Medzi entitami Charges a Accountmovements je kardinalita 1:0..1.
- **Book**, tabuľka kontaktov, v požiadavkách nazvaný adresár. Obsahuje informácie o kontaktoch, ktoré si užívateľ identifikovaný atribútom *id_user* pridal do svojho adresára. Každý kontakt je identifikovaný primárnym kľúčom *id*. Atribút *top* nastavený na 1 reprezentuje kontakt, ktorý sa bude zobrazovať na každej podstránke ako rýchly kontakt. Jeden kontakt môže mať priradených viac bankových účtov, ktoré reprezentuje entita s primárnym kľúčom *id* Bookaccounts. Entita Book je vo vzťahu s entitou Bookaccounts s kardinalitou 1:0..n.
- **Bookaccounts** obsahuje cudzí kľúč *bank* a *id_book*, ktorý k bankovému účtu zaraďuje kontakt, ktorému patrí. Užívateľ môže mať priradených 0 až neobmedzene účtov, preto je kardinalita 1:0..n medzi User a BookAccounts.
- **Invoice** je najrozsiahlejšia entita ER diagramu, čo sa atribútov týka. Obsahuje atribúty reprezentujúce informácie o vystavenej faktúre priamo cez účtovný systém. Každá faktúra je identifikovaná primárnym kľúčom *id*. Cudzí kľuč *id_user* identifikuje užívateľa, ktorý faktúru vystavil a je taktiež aj dodávateľom. Keďže údaje o dodávateľovi sa môžu zmeniť, je potrebné zadať údaje o dodávateľovi aj do tejto tabuľky. Atribút pečiatka reprezentuje

aktuálnu pečiatku pri vytváraní faktúry. Každá vytvorená faktúra v systéme musí automaticky vytvoriť aj prijem v tabuľke Charges. Kardinalita medzi entitami User a Invoice je 1:0..n.

- **Invoicesubject** je pomocný atribút pre atribút Invoice so vzájomným vzťahom 0..n:1. Atribúty reprezentujú informácie o predmetoch faktúry. Tabuľka nemá primárny kľuč. Potrebný je iba cudzí kľuč *idinvoice* na previazanie s faktúrami. Celková suma na faktúre je suma atribútov *price*, kde je atribút *idinvoice* zhodný.
- **Bankaccounts** definuje atribúty pre reprezentáciu bankových účtov podnikateľov, užívateľov. Primárny klúč je *id*, cudzí kluč *id_user* a *bank*. *Hide* atribút slúži na uchovanie účtu v histórii. Entita User má s touto entitou kardinalitu 1:0..n.
- **Yearbalance** reprezentuje zostatky k 1. januáru účtovného roka. Primárny kľuč je *id*, cudzí kluč *id_type*, cez ktorý je tabuľka previazaná s Bankaccounts. Vzájomný vzťah medzi Yearbalance a Bankaccounts je 1:0..n nakoľko ak nie je uvedený zostatok v Yearbalance, zostatok je 0€.
- **YearUserSetting** je entita pre odpočítateľné položky, ktoré si chce dať do účtovníctva individuálne sám podnikateľ. *Id* je primárny kľuč, cudzím kľúčom je *id_user*. *Typ* definuje, či ide o navýšenie základu dane alebo o zníženie o hodnotu vyjadrenú v atribúte *value*. Atribút *year* reprezentuje rok, pre ktorý platí dané nastavenie. Vzťah s tabuľkou User je 1:0..n.
- **Dennikmenu** reprezentuje stĺpce peňažného denníka. Primárny kľúč je *id*. Táto entita je vo vzťahu s entitou Cash a s entitou Accountmovements s kardinalitou 1:0..n.
- **Banks** reprezentuje všetky banky na slovenskom trhu. Obsahujú kód banky *kod*, ktorý ich identifikuje. Táto entita Banks je vo vzťahu s entitou Bookaccounts a s entitou BankAccounts s kardinalitou 1:0..n.

Ostatné entity sa využívajú na doplnenie systému. Entita **logs** reprezentuje informácie o prípadných chybách počas behu systému. **Doyouknow** je entita pre tipy a triky, ktoré sa budú zobrazovať náhodné. **YearSetting** entita reprezentuje informácie o všeobecných daňových úľavách pre všetkých podnikateľov využívajúcich jednoduché účtovníctvo. Prepojené entity začínajúce na MD sú entity pre vytvorenie prídavného fóra. Príspevky reprezentované entitou **MDForum** sú radené v témach **MDForumTemy**. Témy sú radene v kategóriách **MDForumKategorie**.



Obr. 3-2 ER diagram účtovného systému

3.4 Grafické rozhranie

Tvorba grafického vzhľadu by mala prejsť taktiež analýzou. Je potrebné vytvoriť prehľadný a jednoduchý grafický návrh, kde by štruktúra webu nemala zbytočné miasť návštevníka stránky.

Neprihláseného návštevníka je vhodné osloviť a vysvetliť podstatu webu a ponúknuť mu registráciu. Ľudia neradi čítajú, a preto použijem na prezentovanie systému s jeho demonštrovaním krátke video. Človek si rád pozrie video, je to pre neho jednoduchšie. Je dôležité, aby návštevník najskôr pochopil o čom projekt je a aby ho aj hneď vyskúšal a neodkladal to na neurčito. Registrovaným je potrebné poskytnúť možnosť prihlásenia.

Pre prihláseného užívateľa je podstatný prehľad. Prehľad tak ako v štruktúre webu, tak aj vo svojom účtovníctve. Je potrebné zvážiť, na ktoré odkazy sa bude často klikať a ponúknuť ich návštevníkovi na každej podstránke bez zbytočných kategorizovaných preklikávaní. Takéto odkazy sú odkazy na vytváranie dokumentov, ktoré sa bude vytvárať často. Taktiež je vhodné ponúknuť zobrazovanie niektorých kontaktov z adresára na každej podstránke, aby návštevník mohol okamžite pristupovať ku kontaktov vybraných klientov. Štatistky účtovníctva budú poskytované na každej podstránke, aby pri každej práci so systémom mal podnikateľ viditeľnú spätnú väzbu, ako sa zmena prejavila v jeho hlavných údajoch účtovníctva.

Vzory grafik pre web sú stránky google.com a facebook.com. Sú to jednoduché a pritom často využívane weby. Nezaťažujú návštevníka a ponúkajú mu grafickú jednoduchosť a prehľad, čo návštevník spolu s možnosťou meniť farebnosť webu ocení.

3.5 Použité technológie

Médium, cez ktoré bude možné využívať účtovný systém euctovnicka.sk je celosvetová sieť Internet. Prostredníctvom webového prehliadača je možné so systémom plne pracovať bez nutnosti inštalácie ďalších podporných programov. Navyše sa väčšina operácií spracováva na serveri, klientove zdroje sa šetria, resp. sa môžu využiť inak. V nasledujúcich podkapitolách je popísaný výber programovacích jazykov a technológií.

3.5.1 Skriptovací jazyk PHP

PHP[3] (skratka pre PHP Hypertext Preprocessor) je skriptovací jazyk zabudovaný na strane serveru. To znamená, že pracuje vnútri dokumentu HTML a požičiava mu tak schopnosť generovania požadovaného obsahu. Dôvody prečo nevyužiť iné alternatívy ako ASP, Cold Fusion, Perl, Java, Python alebo staré dobré scripty shell/awk/sed sú:

- Jednoduchosť.
- Najprirodzenejší spôsob práce s databázou.
- Nezávislosť na platforme.
- Open Source.
- Vyššia prístupnosť, ak je súčasťou webového serveru.

Názov PHP bol pôvodne odvodený od “Personal Home Page” v roku 1994, keď Rasmus Lerdort dal dohromady kombináciu skriptov v Perle. Tieto scripty boli vydané ako balíček, po ktorom bol značný záujem. Pridali sa vlastnosti jazyka C a v neskorších verziách rozšírené o možnosť používať objekty. Jedna zo zaujímavých vlastností PHP je, že umožňuje viac ako bežný skriptovací jazyk. Vďaka modulárnemu návrhu možno PHP používať aj na vývoj aplikácii s užívateľským rozhraním (GUI)[3].

V spolupráci s frameworkom je práca PHP ešte jednoduchšia. PHP beží na takmer všetkých najrozšírenejších operačných systémoch, vrátane UNIXu, Linuxu, Windowsu a Mac OS X. Operačný systém návštevníka stránky nemá vplyv na PHP, keďže PHP využíva komunikáciu klient-server.

3.5.2 Databázový server MySQL

MySQL je kompaktný, jednoducho použiteľný databázový server, ideálny pre malé a stredné aplikácie. Je to implementácia klient-server, ktorá sa skladá z deamonu serveru mysqld a množstva rôznych klientskych programov. Je dostupný pre rôzne platformy UNIX aj Windows. Na unixových platformách používa rozdeľovanie do podporcesov (vlákien), ktoré z neho robia vysoko výkonný dostupný databázový server. Databázové aplikácie môžu byť vytvárané v rade programovacích jazykov ako PHP, C Perl atď.

MySQL je rýchlejšia ako mnoho iných komerčných databáz a má veľmi jednoduchú správu. Keďže je relačná databáza Open Source, pre všetky platformy je definovaná zdarma [3].

3.5.3 Zend Framework

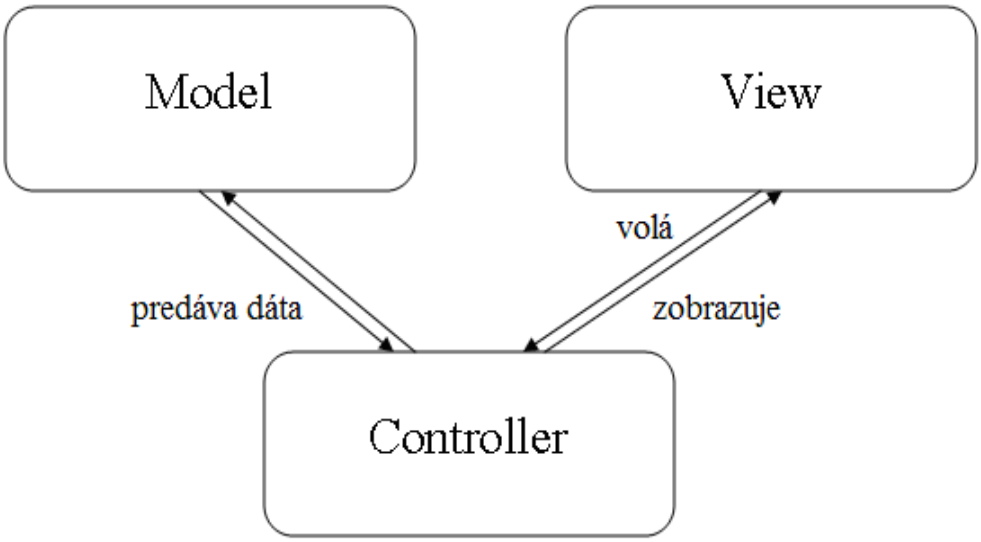
Zend Framework[4] označovaný aj ako ZF je Open Source technológia pre vývoj aplikácií v PHP. Je objektovo orientovaný webový aplikačný framework. Obsahuje množstvo nezávislých komponentov, ktoré umožňujú vyvíjať PHP aplikácie ľahšie s udržateľnejším kódom pre budúce úpravy a vylepšenia. Komponenty sú nezávislé, nakoľko užíva modulárnu architektúru a vyhovujú direktíve E_STRICT.

Jeho jadro tvoria základne potrebné moduly a následne si vývojár môže vložiť iba tie komponenty, ktoré bude využívať. Jadro tvoria tieto základné komponenty:

- Zend_Controller – srdce MVC systému, ovládač.
- Zend_View – formátovanie dát návštevníkovi stránky.
- Zend_Db – práca s models, s dátami.
- Zend_Config – nastavuje konfiguračné parametre.
- Zend_Filter – filtrovanie aj overovanie dát.
- Zend_Validate – validovanie dát.
- Zend_Registry – alternatíva k používaniu globálneho úložiska.
- Zend_Acl – autentizácia.
- Zend_Auth – autorizácia.
- Zend_Session – práca so sessions.

Komponenty tvoriace jadro Zend Frameworku využívajú Model-controller-view architektúru, označovaný ako MVC architektúra. Je vhodný pre tvorbu aplikácií, kde je oddelený vzhľad internetovej stránky od logiky a controllerov. Hlavné časti MVC teda tvoria: Zend_Db (Model), Zend_View (View), Zend_Controller (Controller). Tieto časti sú oddelené aj štruktúrou adresárov. Obr. 3-3 Ilustruje spoluprácu jednotlivých častí.

Model je časť aplikácie, ktorá pracuje s dátami. Tieto dáta predáva Controlleru. View je časť aplikácie, ktorá má na starosti formátovanie dát užívateľovi. View prijíma konkrétne výsledné dáta od Controllera, ktoré sa už majú zobraziť v určitom formáte. Tento formát je práve definovaný vo View. Controller je ovládač práce medzi modelom a viewom, tak aby pri špecifickom požiadavku dostal užívateľ správne údaje v správnej forme. Volá od Modelu špecifické dáta, ktoré sa v Controlleri spracujú a dajú zobraziť vo View. V Controlleri sú definované akcie, ktoré sú volané prevažne z View.



Obr. 3-3 Ilustrácia Model-Controller-View systému

Dôvody prečo som si vybral práve Zend Framework sú:

- Rozsiahla dokumentácia.
- Rozsiahla komunita.
- Podporia množstva komponentov.
- Jednoduchosť vývoja.
- Umožní rýchly vývoj aplikácií.
- Je známejší ako český Nette.

Zend Framework od svojho vzniku v roku 2005 prešiel rôznymi vylepšeniami. Obsahuje podporu pre multi-databázové systémy zahrnuje MySQL, Oracle, IBM DB2 MSSQL Server, PostgreSQL, SQLite a Informix Dynamix Server. Jednoduchou zmenou v konfigurácii dokáže začať komunikovať s iným databázovým serverom. Podporuje viacero techník prác s dátami a prispôsobuje sa trendom.

Zend Framework má nasledovné požiadavky:

- PHP verzie 5.1.4 (alebo vyššia).
- WEB server so zapnutou funkciou mod_rewrite.

Server eutovnicka.sk tieto požiadavky spĺňa.

3.5.4 JavaScriptová knižnica jQuery

jQuery[8] je knižnica pre JavaScript, obsahujúca množstvo náročnejších funkcií, ktoré je možné jednoduchým importovaním a zavolaním vykonať. Keďže je JavaScriptová knižnica, pracuje na strane klienta. Kladie doraz na interakciu medzi JavaScriptom a HTML. Pomocou jQuery v niektorých prípadoch môže klientovi uľahčiť a spríjemniť niektoré funkcie, ako napríklad vyhľadávanie s našepkávaním alebo radenie stĺpcov v HTML tabuľkách. Je to opäť Open Source knižnica.

Syntax je navrhnutá pre jednoduchšiu navigáciu dokumentu, výber DOM (Document Object Model) elementov, pracovať s CSS, vytváranie animácií, spracovanie udalostí a vývoj Ajax aplikácií. Poskytuje možnosti pre vývojárov na vytváranie pluginov postavených na tejto JavaScript knižnici. Veľkou výhodou je vytvorenie veľkej jQuery komunity, kde sa vytvárajú prípadne chýbajúce pluginy.

3.5.5 XML

XML znamená Extensible Markup Language (rozšíriteľný jazyk so značkami). Jeho špecifikácia je popísaná odporučením konzorcia World Wide Web Consortium. Jazyk XML nám dovoľuje vytvárať štruktúrované dokumenty veľmi flexibilným spôsobom. Je ho možné využiť k vytváraniu dokumentov. XML podobne ako HTML slúži na výmenu dát a na použitie na webe. XML sa pokúša oddeliť obsah a jeho reprezentáciu. Dokumenty XML obsahujú informácie v štruktúrovanej forme, kde si môžeme definovať svoje vlastné značky, ktoré štruktúru určujú [3].

3.5.6 XHTML

XHTML (eXtensible HyperText Markup Language) je značkovací jazyk pre tvorbu webových stránok. Jazyk využíva tagy pre definovanie formátu obsahu webu podobne ako jeho predchodca, jednoduchší HTML. Písmeno "X" v skratke XHTML označuje prírastok HTML do veľkej rodiny rozšíriteľných, samo popisných značkovacích jazykov, schopných spoločne spolupracovať. Jedná sa o rodinu jazykov XML popísanú vyššie [4].

XHTML prakticky vyžaduje použitie externých JavaScript scriptv a oddelenie CSS. Prináša aj niektoré nové pravidlá pre písanie tagov, napríklad všetky tagy musia byť ukončené, písané malými písmenami a hodnoty atribútov musia byť uzavreté v úvodzovkách. Kód napísaný v XHTML je prehľadnejší a zrozumiteľnejší ako kód v staršom HTML.

3.5.7 Kaskádové štýly CSS

Kaskádové štýly CSS (Cascading Style Sheets) priniesli zmenu v písaní kódu HTML šablón. Od vzhľadu webov vytvorených v rámoch (frames) cez tabuľkové layouty sa dospelo k názoru, že vzhľad sa oddelí od kódu. Realizujú to práve CSS, kaskádové štýly. Jednoduchou zámenou súboru štýlu je možné zmeniť celý vzhľad web stránky. Kaskádové sa nazývajú, pretože je možné vrstvenie definícií štýlov, ale platia iba tie posledné, ak sa niektoré upravia. Prinášajú nové možnosti pre formátovanie obsahu. Nevýhodou je však rôzna podpora u niektorých webových prehľadávačov. Môže tak dôjsť k rôznym vzhľadom webu, avšak aj toto je možné ošetriť určitými trikmi. Používanie CSS je už bežný štandard pri tvorbe webových stránok.

3.5.8 PDF a trieda mPDF

PDF (Portable Document Format) je hardwarovo aj softwarovo nezávislý formát. Slúži na ukladanie dokumentov, v ktorých môžu byť obsiahnuté text, obrázky a hypertextové obrázky. Jedná sa o otvorený štandard, ktorý je vďaka svojej nezávislosti veľmi obľúbený a rozšírený. Je zaručené, že sa dokument zobrazí rovnako na akejkoľvek platforme.

Trieda mPDF je PHP trieda na generovanie PDF súborov z HTML šablón. Podporuje UTF-8 kódovanie. Je veľmi praktická, nakoľko nepotrebuje vytvárať nové šablóny pre generovanie PDF súborov ako v prípade generovanie PDF súborov v Zende.

Pomocou tejto triedy je možné PDF dokument ukladať na server alebo zasielať ako prílohu na e-mail. Vie spracovať aj obrázky uvedené v HTML kóde. Pri písaní validných šablón pre generovanie PDF môžeme natrafiť na problémy s CSS štýlmi, ktoré nemusia byť správne knižnicou správne zobrané. Nakoľko mPDF pracuje hlavne s HTML tagmi, je v tomto prípade použitie starých neXHTML značiek vhodné. Šablóna slúži iba na generovanie PDF a nie na zobrazovanie webu. Význam validity tu zohráva nepodstatnú úlohu.

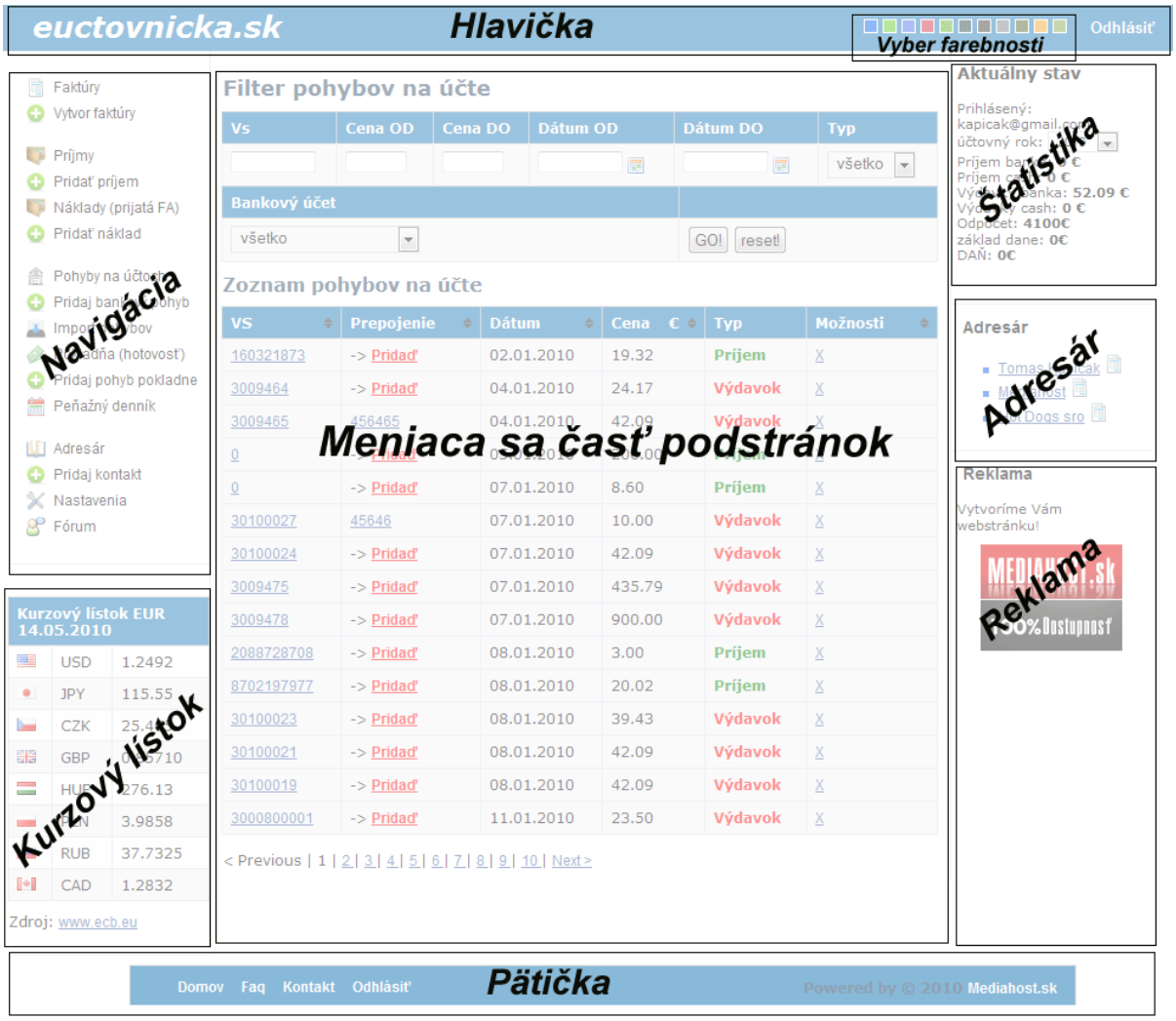
4 Implementácia

Táto kapitola opisuje implementáciu systému podľa návrhu z predošlej kapitoly. Postup implementácie je chronologicky popísaný v podkapitolách.

4.1 Tvorba užívateľského rozhrania a XHTML šablóny

Implementácia začína pri tvorbe grafického návrhu. Začať práve grafikou je vhodné z dôvodu následnej implementácie jadra systému priamo do grafiky, resp. XHTML šablóny. Po dôkladnej analýze máme prehľad ako sa systém bude využívať. Práve preto je vhodné jednotnú štruktúru, pre väčšinou podstránok hneď na začiatku implementácie. Vytvárajú sa dva grafické návrhy. Jeden pre časť neprihlásených užívateľov a druhá pre prihlásených, ktorý pracujú s účtovníctvom. Všetky návrhy majú aj rôzne farebné mutácie.

Grafika neprihláseného užívateľa obsahuje prezentačné video a formulár pre registrovanie a prihlásenie, prípadne odkazy na statické stránky. Spoločnú časť s prihláseným užívateľom je horná lišta, kde sa nastavuje grafická farebnosť webu s logom.



Obr. 4-1 Štruktúra užívateľského rozhrania prihláseného

Návrh pre prihláseného je 3-stĺpcový design s hornou lištou a pätičkou. Štruktúra je znázornená na Obr.4-1. V prvom stĺpci je hlavná navigácia webu. Pod ňou je tabuľka s aktuálnym kurzovým lístkom spolu s tipmi a trikmi. Druhý, najrozsiahlejší stĺpec bude využívaný pre dáta konkrétnej podstránky. V treťom pravom stĺpci sú štatistiky s formulárovým výberom aktuálneho účtovného roka. Nasleduje adresár s rýchlymi kontaktmi adresára s odkazom na vytvorenie faktúry s údajmi adresára. Pod kontaktmi je priestor pre prípadnú reklamu.

Po vytvorení grafického návrhu v grafickom programe Adobe Photoshop CS bol výsledný .psd súbor narezaný na vhodné obrázky a bol vytvorený validný XHML kód. Valídna stránka sa hlavne rýchlejšie načítava – rýchlejšie sa zobrazí v okne prehliadača www stránok. Vytvorené šablóny sú až na embed bez videa z youtube.com valídne.

4.2 Tvorba databázy

Na základne dobre vypracovaného ER diagramu je vytvorenie databázy s tabuľkami veľmi jednoduchá úloha. Počet tabuliek je 19, tak ako počet entít v ER diagrame. Primárne kľúče majú nastavené extra vlastnosti auto_increment, teda je zaručená jednoznačná identifikovateľnosť. Tabuľky nahrávam na webhostingový server pre euctovnicka.sk. Štruktúra tabuliek bez dát je k dispozícií na www.euctovnicka.sk/public/sql.sql. K dátam pristupujem cez phpMyAdmin na adrese http://phpmyadmin.euctovnicka.sk. Dátové typy boli popísané aj v ER diagrame, spolu s vhodne zvolenými indexmi. Verzia MySQL na serveri je: 5.1.45.

Názvy tabuliek je potrebné písať bez podtržníkov, nakoľko pri práci s models je tento znak špeciálny znak, ktorý má funkciu ako separátor. Názvy stĺpcov sa riadia pravidlami pri vytváraní MySQL dotazov. Nie je vhodné nazvať stĺpce ako niektoré významové slova „do“, „where“ alebo znakové názvy, ktoré v niektorých prípadoch bez použitia ,`' môžu robiť problémy.

4.3 Inštalácia a nastavenie Zend Frameworku

Pre využívanie Zend Frameworku je potrebné nainštalovať daný systém na server, keďže väčšinou nebýva interne nainštalovaný. Je potrebné preto nahrať súbory frameworku na server so všetkými knižnicami, ktoré bude systém využívať. Základom je vhodne navrhnúť adresárovú štruktúru. K využívaniu architektúry MVC je patrične navrhnutá základná adresárová štruktúra na Obr. 4-2.

```
www_root/
    /application
        /configs
        /controllers
        /forms
        /layouts
        /models
        /views
            /helpers
            /scripts
    /library
    /public
        /css
        /images
        /js
```

Obr. 4-2 Základná adresárová štruktúra projektu

V hlavnom adresári `www_root` sú vytvorené tri hlavné podadresáre. Podadresár `application` pre prevažne využívanie MVC architektúry, kde sú samostatné adresáre pre súbory `model`, `view` a `controller`. Zložka `forms` slúži pre ukladanie formulárov. V `layouts` sa vkladajú HTML grafické šablóny. Konfigurácia Zendu sa nastavuje v adresári `configs`. Všetky knižnice nahrávam do podadresára `library`. V tomto adresári je nahratý aj súbor knižníc Zendu. Pre zjednodušenie práce s objektmi je doinštalovaný do Zendu systém nazvaný *Miwo*. Pomocou neho sa jednoduchšie a hlavne rýchlejšie deklarujú triedy spolupracujúce s databázou a jednoduchšie sa pristupuje k objektom cez rôzne pomocné funkcie. Ide o určité rozšírenie Zendu na základe skúsenosti s opakovateľnosťou prác s týmto frameworkom. Obrázky, JavaScript a CSS súbory sú uložené vo vlastných adresároch podadresára `public`.

Pre vloženie knižnic Zendu do adresára pre knižnice je ich potrebné najskôr stiahnuť. Na stránke http://framework.zend.com/download je k dispozícií Zend Framework. Stačí stiahnuť minimálnu verziu s adresárom library a celý ho vložiť do podadresára `library`.

Controller Zend Frameworku, *Zend_Controller* je navrhnutý pre podporu internetových stránok s peknými URL. Aby sme toto dosiahli, musíme všetky požiadavky (requesty) vybavovať cez jediný súbor index.php, nazývaný tiež *bootstrapper*. Toto nám poskytne centrálny bod pre všetky stránky a zaručí, že prostredie bude pre aplikáciu správne nakonfigurované. Takéto správanie dosiahneme použitím súboru .htaccess v adresári `www_root` [7]. Môj .htaccess vyzerá nasledovne:

```
SetEnv APPLICATION_ENV development
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} -s [OR]
RewriteCond %{REQUEST_FILENAME} -l [OR]
RewriteCond %{REQUEST_FILENAME} -d
RewriteRule ^.*$ - [NC,L]
RewriteRule ^.*$ index.php [NC,L]
```

Okrem centrálneho presmerovania súborov a adresárov, ktoré prakticky neexistujú na serveri, nastavujem aj prostredie, v ktorom budem pracovať. Mám definované dve prostredia. Prostredie pre tvorbu - development a prostredie pre štandardný beh webu - production. Pri development je vhodné nastaviť vypisovanie zložitejších chybových výziev ako pri production, a preto prostredie development využívať pri testovaní a nastavovaní.

Vlastnosti prostredí nastavujem v podadresári `configs` v súbore *application.ini*. Okrem prostredia tu nastavujem aj iné natavenia a hlavne parametre pre pripojenie k databáze. Využívať sa bude PDO_MYSQL adaptér. Správnym nastavením parametrov *resources.db* je možné komunikovať s databázou. Nastavujú sa tu adresáre pre formuláre (`application/forms`), models (`application/models`), knižnice (`library`), grafické vzhľady (`application/layouts`). Adresárovú štruktúru si takto viem navrhnúť a nastaviť podľa vlastného návrhu.

Keďže všetky fiktívne súbory a adresáre sú presmerované na *index.php*, je potrebné v ňom nastaviť spracovanie dát. V ňom najskôr definujem cesty k MVC architektúre a ďalej vyberám prostredie, v ktorom sa pracuje. Taktiež je potrebne načítať knižnice, s ktorými sa bude pracovať, teda adresár `library`. Vytvára sa tu základný objekt *Zend_Application*, ktorý poskytuje opakované použitie zdrojov. Tiež sa stará o nastavenie prostredia PHP a zavádza autoloading v predvolenom nastavení.

V súbore *bootstrap.php* sú nastavené opakované volané zdroje, ako napríklad nastavenie základného grafického rozhrania ak nie je žiadny definovaný, nastavovanie doctype pre formát zobrazovania (XHTML), prednastavené metatagy, titulky a tiež aj vytváranie modelu views. Taktiež

je tu tak ako inicializácia časti modelu views, aj inicializácia práce s databázou, ktorá bude využívaná v časti models.

Nastavujem tu aj loganie chýb s využitým *Zend_Log*. Ide o informovanie a uchovávanie histórie chýb v systéme. Každá chyba má spoju prioritu. Priorít je 7 a najväčšiu prioritu má CRIT – kritická chyba. Nastavujem tu zasielanie e-mailu v prípade, že ide o vážnu chybu. Chyby najvyšších priorít sú uchovávame aj v databáze v tabuľke log.

4.4 Práca s MVC architektúrou

Práca s MVC architektúrou bola pre mňa novou skúsenosťou. Programovanie je rozdelené do troch časti, pre väčší prehľad a možnosť jednoduchšieho upravovania v budúcnosti. V nasledujúcich troch podkapitolách tejto kapitoly je popis vytvárania súborov MVC architektúry a nastavení v daných súboroch.

4.4.1 Models

Pre prácu s databázou je vhodné vytvoriť triedu na prácu s danou tabuľkou. Väčšinou sa vždy pracne písala deklarácia triedy. S využitým spomínanej Miwo knižnice stačí definovať iba stĺpce tabuľky, primárny kľuč a názov tabuľky. Pomocou týchto údajov bude každý stĺpec tabuľky reprezentovať atribút objektu triedy. Je možné nastaviť a j predefinované hodnoty pre vkladanie do tabuľky. V models je vhodné vytvárať funkcie, ktoré spolupracujú s databázou. Čí už ide o hľadanie najväčšieho čísla v určitom stĺpci tabuľky alebo metódy pre filtrovanie záznamov tabuľky. Samozrejme je tu možné vytvárať aj bežné statické funkcie, ktoré sa do triedy je vhodne zatriediť. Názov súborov models je nutné nastavovať ako názov triedy.

Miwo knižnica má viacero pomocných často využívaných funkcii, ktoré môže využívať aj ostatných častiach MVC architektúry:

- *find($primary_key)* - pomocou ktorej vloží do objektu dáta, kde je primárny kľuč hodnota parametru.
- *save()* – pri zmene dát v objekte dáta uloží do tabuľky.
- *beforeSave()* – udalosť volaná pred uložením modelu, možnosť vložiť chýbajúce hodnoty.
- *beforeInsert()* a *beforeUpdate()* sú funkcie ktoré sú volané pre vložením, resp. aktualizovaním záznamu v tabuľke.

4.4.2 Views

Formát zobrazovaných dát je XHTML kód. Súbor vo views je jednoduchá webová podstránka pracujúca s premennými, ktoré dostala od controllera. Views sú vytvárané v adresári `views` a musia mať rovnakú štruktúru ako controller. Každému controllleru prislúcha rovnako nazvaný adresár vo `views/scripts/`. Každá akcia v controlleri má priradený súbor vo views v adresári daného controllera. Pre správne fungovanie systému MVC je potrebné zachovávať tieto pravidlá.

Pre vykresľovanie sa *Zend_View* objekt nastaví tak, aby skripty hľadal v umiestnení `views/scripts/{názov controlleru}` a implicitne hľadal skript nazvaný podľa akcie s koncovkou phtml. Plná cesta k skriptu views určitého controllera a jeho akcie má formu `views/scripts/{názov_controlleru}/{názov_akcie}.phtml` a vyrenderovaný obsah sa uloží do tela objektu *Response*. Objekt *Response* sa používa na to aby zhromažďoval všetky HTTP hlavičky, telo dokumentu a generované výnimky ako výsledok práce MVC systému. Front controller

potom automaticky pošle hlavičky nasledované telom dokumentu na konci tzv. *dispatchu* (posledný príkaz v *bootstrapperi*)[5].

4.4.3 Controllers

V Zend Frameworku je controller triedou, takže sa musí volať podľa syntaxe {Názov Controlleru}Controller. Táto trieda je zapísaná v súbore `{Názov Controlleru}Controller.php` v špecifikovanom adresári s controllermi. Každá akcia je verejnou funkciou triedy controllera a musí mať názov *{názov akcie}Action*. Akcie reprezentujú určite spracovanie dát. Môže isť o dáta odoslané z formulára alebo vytiahnuté z models. Akcie controllera sú volané cez URL adresu a to v podobe `domena/{názov controlleru}/{názov akcie}`.

V akciách sa nastavujú operácie s dátami a nastavujú sa dáta odoslané do views, s ktorými môžeme v časti view pracovať, resp zobrazovať. Premenné, ktoré chcem zobraziť vo views je potrené zapísať do objektu *$this->view->{názov premennej}*. Vo views bude hodnota prístupná cez *$this->{názov premennej}*. V akciách nastavujem okrem iného aj layout pre zobrazovanie dát vo views, nastavujú sa tu meta tagy, doplňujem titulok web stránky.

4.5 Funkcie účtovného systému s využitým Zend Frameworku

Keďže implementácia systému bola s využitím frameworku, všetky požiadavky som navrhoval s využívaním knižníc Zend Frrameworku. Tento framework ponúka veľké množstvo komponentov, ktoré je vhodné využiť.

Pre nastavovanie grafického užívateľského rozhrania je využívaný komponent *Zend_Layout*. Boli vytvorené základné šablóny, ktoré obsahovali časti formátu a dát podstránok, ktoré sa často neobmieňali. Pre každú podstránku je nastavovaná práve jedna šablóna. Šablóny sú dve hlavné a to jedna pre časť prihláseného a druhá pre neprihláseného, kde sa môže registrovať. Taktiež je možné nastaviť aby šablóna bola prázdna, čo využívam pri generovaní peňažného denníka, kde sa generuje priamo čistý dokument.

Pre ukladanie informácií o návštevníkoch stránky je vhodné využiť Sessions. Ide hlavne o nastavenia, ktoré majú byť nastavené pre užívateľa dlhodobo, teda aj po ukončení práce so systémom. Sessions sú dáta, ktoré sa ukladajú na serveri a teda po ukončení práce s prehľadávačom je možné si ponechať dáta užívateľa na serveri, a vytiahnuť ich pri ďalšej návšteve v budúcnosti. Aj pre prácu so sessions je možné vyžiť možnosti Zendu. Pomocou objektu triedy *Zend_Session* je možné pracovať so Sessions ako s objektom. Tento komponent využívam pri nastaveniach voľby grafického dizajnu stránky. Výber grafickej farebnosti sa ukladá do Sessions a následné ho porovnávam s grafickými štýlmi. Vybratý je iba ten štýl, ktorý prislúcha hodnote v Sessions. Takýmto spôsobom načítavania iba tých štýloch, ktoré sú potrebné sa šetria prenesené dáta, nakoľko sa neprenášajú informácie o štýloch, ktoré nie sú vybraté. *Zend_Sessions* je využívaný aj pri výbere aktuálne účtovného roku.

Zend obsahuje viacero komponentov, ktoré je vhodné využiť pri implementácii. Komponentu pre všeobecné účely logovacia som využil už pri inštalácii Zendu. Je už iba na programátorovi. či si vyberie možnosť vlastného spracovania dát alebo využije na spracovanie určitej udalosti framework.

4.5.1 Registrácia a prihlásenie užívateľov

Registrácia užívateľa je riešená pomocou formulára. Formulár obsahuje iba povinné položky, aby množstvo položiek formulára neodrádzalo návštevníka stránky od registrácie. Dáta formulára sú odosielané do LoginControllera a spracované sú v akcii *registrationAction()*.

Pre overovanie správnosti vložených dát je pre formulár vytvorená trieda pre registračný formulár, ktorá dedí metódy od komponentu *Zend_Form*. V tejto triede definujem elementy formulára a definujem ich požadovanie pre spracovanie formulára. Dôležitú úlohu tú vykonávajú validátory. Pomocou nich je možne overovať hodnoty zadané vo formulári s využitím prednastavených overovaní. Je možné overovať správnosť e-mailu bez potreby tvorby vlastnej funkcie na dané overenie. Taktiež je možné validátori využiť na overenie identickosti hesiel, overenie existencie záznamu v tabuľke alebo dĺžky reťazca.

Pri registrácií je vhodné použiť CAPTCHA reťazec pre zamedzenie generovaných registrácií. *Zend_Captcha_Image* je vhodný komponent pre generovane obrázkov s overovacími reťazcami. Nastavuje sa tu typ písma, ktorý bude použitý, rozmery obrázka a tiež aj noiselevel. Je to úroveň náročnosti prečitaní obrázka robotom. Treba však brať ohľad aj na ľudí s obmedzeniami, takže tento level som nenastavoval až moc vysoko.

Po odoslaní dát formulára, *registrationAction()* je najskôr overovaný CAPTCHA reťazec a následne sú validované hodnoty zaslané vo formulári podľa nastavených validátorov využitých so *Zend_Form*. Pri chybe validácie sa vypíše chybová hláska informujúca o chybe spolu s registračným formulárom. Pokiaľ dáta sú správne, vytvára sa objekt *User*. S využitím funkcie *save()* s knižnice Miwo dáta v objekte zapíše do tabuľky nastavenej v models daného objektu. Týmto je nový užívateľ registrovaný. O úspešnej registrácií je informovaný hláškou systému, ale aj potvrdzujúcim e-mailom, ktorý je odosielaný pomocou komponentu *Zend_Mail* na e-mail nového užívateľa.

Po úspešnej registrácií je možné sa prihlásiť. Pre prihlásenie slúži formulár, do ktorého sa zadávajú prihlasovacie meno (e-mail) a heslo. Dáta formulára prechádzajú validáciou. Pokiaľ sú dáta validné, dochádza k autentizácií pomocou *Zend_Auth* komponent. Využíva sa pri tom adaptér *Zend_Auth_Adapter_DbTable*, pomocou ktorého je elegantnejšie pracovať s tabuľkou registrovaných užívateľov pre potreby autentifikácie. Pokiaľ pre prihlasovacie meno a heslo vyhovuje práve jeden záznam v tabuľke *User*, prihlásenie je úspešné. Ak sa vyskytnú chyby, tak sú vypísané.

Pre overovanie autentifikácie je použitý plugin *Controller_Plugin_Auth*. V ňom sú nastavené controllery a akcie, pre ktoré nie je nutné prihlásenie. Pre prácu s ostatnými akciami je autentifikácia nutná a pri prístupe do týchto časti webu je užívateľ presmerovaný na chybové hlásenie o nutnosti autentifikácie. Pre prístup k informáciám o užívateľovi je vytvorený špeciálny prepojovací controller *Controller_Action*. Inicializujem v ňom premenné pre prácu s údajmi o prihlásenom užívateľovi a aktuálnom účtovnom roku.

4.5.2 Vkladanie, editovanie, vymazávanie záznamov

Keďže účtovný systém pracuje s dátami, je potrebné tieto dáta vkladať. Jedná sa o pridávanie informácií o príjmoch a výdavkoch, pohyboch na bankovom účte a v pokladni, vkladanie kontaktu do adresára. Pre takéto vkladanie dát sa využívajú formuláre. Pre prácu s formulármi je vhodné používať komponent *Zend_Form*, hlavne pre výborné plnenie úlohy filtrovania a validácie vstupných údajov.

Pre každý formulár je vytvorená trieda v adresári `forms`, v ktorej sú nastavené validátori pre vstupné údaje. Pokiaľ sú údaje valídne, vytvára sa objekt podľa konkrétneho formulára. Metódou *save()* sú dáta v objekte vložené do tabuľky priradenej k danému objektu. Ak nie sú vstupné údaje

valídne, vráti sa informácia o chybe. Vložené záznamy majú svoj identifikátor, pomocou ktorého je možné vždy označiť záznam, ktorý chcem vymazať označiť alebo zobraziť.

Záznamy už vložené v tabuľkách je možné meniť. Využívajú sa na to rovnaké formuláre ako pri vytváraní, avšak pri editácií sú do formulárov vložené dáta z tabuľky pre konkrétne upravovaný záznam. Údaje o konkrétnom zázname vyberám pomocou funkcie *find()*, prípadne vlastných funkcií deklarovaných v models. Údaje s formulárov sú odoslané do akcií príslušných controllerov, ktoré dáta validujú a záznam upravia pomocou SQL príkazu UPDATE.

Pri vymazávaní záznamov je potrebné vždy overiť, či danému užívateľovi patrí konkrétny záznam, ktorý chce vymazať. Pokiaľ áno, môže ho zmazať pomocou príkazu DELETE. Pri prípadom zobrazovaní informácií sú vyťahované konkrétne dáta z tabuľky pomocou funkcie find() a následne zobrazované.

4.5.3 Import dát z XML

Údaje o pohyboch na bankových účtoch je možné vložiť aj automaticky pomocou vloženia XML súboru s pohybmi. Slúži na to odkaz Import pohybov v navigácií, ktorý odkazuje na akciu *importAction AccountmovementsControllera*. Po vložení XML súboru je obsah pomocou štandardnej funkcie PHP *simplexml_load_file($file)* spracovaný do objektu. S hodnotami je možné pracovať ako s hodnotami pola. Tieto hodnoty sú vkladané do tabuľky pre bankové pohyby *Accountmovement.*

Užívateľ má možnosť rozhodnúť, či chce záznamy s rovnakými informáciami preskakovať. Pokiaľ si vyberie ignorovanie zhodných záznamov, tak pri vkladaní sa overuje, či rovnaký záznam už existuje. Dáta sú ukladané iba v prípade, že rovnaký záznam neexistuje.

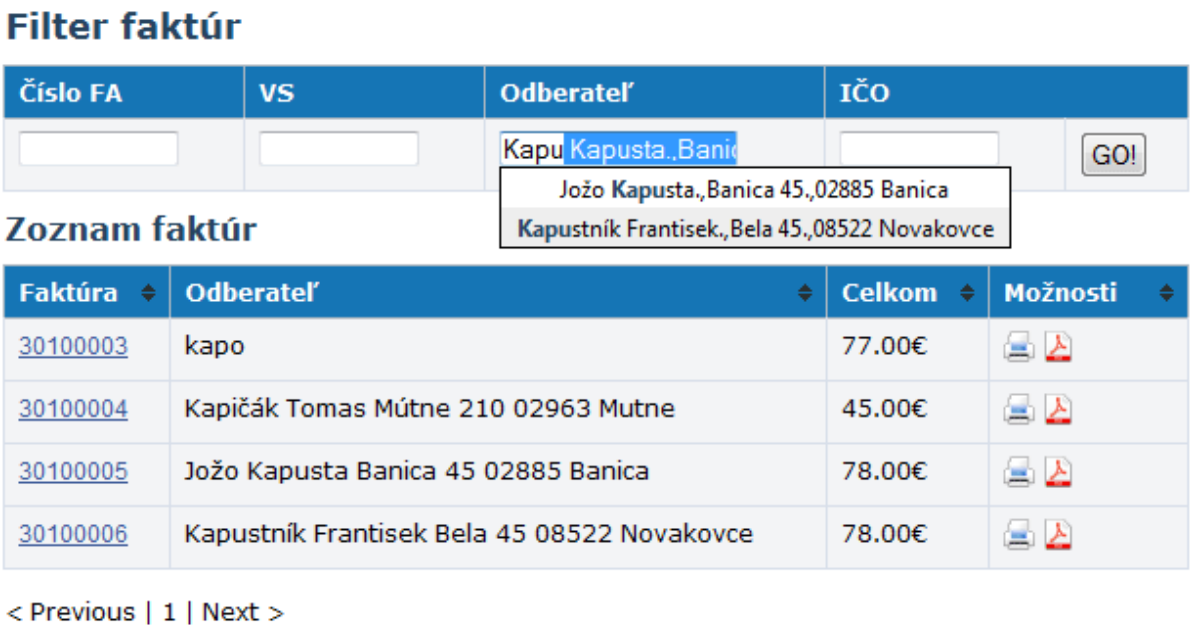
Formát akceptovateľného XML záznamu je formát XML súboru generovaného Tatra bankou. V XML súbore je informácia aj o čísle bankového účtu. Pokiaľ pri vložení XML súboru ešte užívateľ nepridal tento bankový účet, tak systém ho vytvorí – pridá do tabuľky *Bankaccounts* nový záznam s číslom a bankou v XML súbore.

4.5.4 Zobrazovanie zoznamov s filtrovaním

Záznamy z databázy sú zobrazované v zoznamoch v HTML tabuľkách. Ukážka možného zobrazovania záznamov s filtrom je nájdeme na Obr. 4-3. Záznamy je možné radiť podľa vlastného výberu. Pomocou AJAX funkcie *TableSorter* je možné zaradzovať zobrazené záznamy v tabuľkách. Kliknutím na hlavičku stĺpca je možné zoznamy v tabuľke radiť abecedne alebo číselne. Implementácia *TableSorteru* spočíva v načítaní potrebných JavaScriptových súborov a nastavením stĺpcov HTML tabuľky, ktoré budú mať možnosť zaradzovania.

Pri väčšom počte záznamov je vhodné používať stránkovanie. *Zend_Paginator* je komponent frameworku pre prácu so stránkovaním. Pracuje priamo so záznamami, ktoré vráti SQL príkaz. Metódou *setItemCountPerPage($count)* je možné nastavovať počet záznamov na jednu stránku. Ak je zoznamov menej ako počet záznamov na jednu stránku, stránkovanie je neaktívne. V stránkovaní je možné posúvať sa v o jednu stranu vpred aj vzad, prípadne si zvoliť konkrétnu stranu.

Pre zjednodušenie a zrýchlenie hľadania konkrétnych záznamov je ponúknutá možnosť filtrovania. Na základe vložených filtrujúcich dát sú vyhľadané iba tie záznamy, ktoré vyhovujú danému filtru. Na filtrovanie sa používa zložitejšia funkcia, ktorá pozostáva s príkazu SELECT a klauzuly WHERE, v ktorej sú nastavené hodnoty pre filter.



Obr. 4-3 Ukážka zoznamu s filtrovaním s možnosťou autocomplete

Pre zjednodušenie vkladania filtrujúcich hodnôt je použitá ďalšia AJAXová funkcia *autocomplete*. Ide o našepkávanie z možných už existujúcich hodnôt, ktoré obsahujú časť textu, ktorú užívateľ začal písať. Automaticky aj doplňuje najvhodnejší záznam. Ako je možné vidieť na Obr. 4-3, po zadaní „Kapu“ autocomplete ponúkne možnosti z existujúcich hodnôt, ktoré v sebe text „Kapu“ obsahujú.

Autocomplete komunikuje s databázových serverom pomocou PHP scriptu. V scripte je SELECT príkaz, ktorý hľadá v tabuľke záznamy, v ktorých sa časť textu nachádza. Po nájdení tieto hodnoty generuje ako text oddelený riadkami. Tieto riadky sa následne zobrazujú ako možnosti v autocomplete.

4.5.5 Vytváranie faktúr s generovaním do PDF

Pre vytváranie faktúr je použitý formulár s využitým *Zend_Form*. Číslo faktúry sa ponúka vždy o jedno vyššie ako najväčšie v tabuľke *Invoice* pre daného užívateľa. Vstupné dáta sa validujú a v prípade chyby užívateľa informujú hláseniami z *getMessages()*. Po úspešnej validácií sa dáta ukladajú do tabuľky *Invoice*. Ukladajú sa informácie o aktuálnej pečiatke aj o aktuálnych bankových účtoch. Do tabuľky *Charges* sú vložené informácie o novom príjme.

Faktúra sa generuje do PDF formátu pomocou knižnice *mPDF*. Je vytvorená šablóna PDF vo formáte HTML, do ktorej sa vkladajú dáta faktúry. Táto HTML faktúra je vkladaná ako parameter funkcie *WriteHTML($htmlkod)*, ktorá spracuje HTML kód. Funkciou *Output()* je generovaný dokument PDF. Ukážku formátu a štruktúry PDF faktúry nájdeme v prílohe č.1.

Následne je PDF faktúra zasielaná e-mailom na e-mail dodávateľa. Ak je vyplnený e-mail pri odberateľovi, tak je zasielaný aj jemu. E-mail je zasielaný pomocou štandardnej PHP funkcie *mail()* s nastavením pre textové aj HTML prezeranie obsahu s pripojenou PDF prílohou. Faktúru je možne prezerať aj v HTML formáte. Faktúry nie sú ukladané na serveri, ale pri prezeraní v systéme sa generujú. Je to z dôvodu rizika úniku dát na serveri. PDF aj HTML faktúram je pred generovaním overované právo užívateľa k dokumentu.

4.5.6 Párovanie platieb

Každý pohyb v pokladni alebo na účte je možné priradiť k dokladu. V zozname pohybov je možnosť s názvom „Pridať“. Kliknutím na odkaz je možné filtrovať medzi dokladmi rovnakého typu a nájsť odpovedajúci doklad. Automaticky sú však vyhľadané záznamy, ktoré majú rovnakú sumu ako pohyb a sú v časom intervale plus mínus 15 dní od prijatia platby.

Po vybratí odpovedajúceho dokladu sú zobrazené informácie o pohybe aj o doklade pre overenie. Identifikátory o doklade a pohybe sú prenášane cez GET parametre. Na každej podstránke sa testuje, či dané záznamy je možné spárovať. Overuje sa, či záznamy nemajú spárovaný iný dokument, či sú oba rovnakého druhu (kreditový alebo debetový). Párovaný záznam má určený svoj záznam v peňažnom denníku. Tento záznam sa vyberá spolu s príkazom na spárovanie. Pri spárovaní sa informácie o prepojení uchovávajú v tabuľkách v stĺpcoch *connection*. Sú tam odkazy na identifikátory naviazaného dokumentu. Po spárovaní záznamu nie je možné daný záznam spárovať s ďalším.

4.5.7 Generovanie peňažného denníka

Pre generovanie peňažného denníka je potreba vybrať všetky pohyby na účtoch aj v pokladni za aktuálne nastavený účtovný rok. Príklad SQL príkazu, ktorý vyberá všetky pohyby v pokladni aj na bankových účtoch v roku 2010 pre užívateľa s identifikátorom 28 je uvedený na Obr. 4-4. Pomocou príkazu UNION sa zjednocujú záznamy z dvoch tabuliek do jednej. Informácia o tom, či záznam je z pokladne alebo z bankového účtu je definovaná v novom stĺpci *druh*.

```
SELECT 1 AS druh, `id` , `number` , `vs` , `cs` , `text` , `amount` ,
`currency` , `connection` , `typ` , `date` , `denniktyp` , `idaccount`
FROM `Accountmovements`
WHERE `id_user` = '28'
AND date
BETWEEN '2010-01-01'
AND '2010-12-31'
UNION
SELECT 2 AS druh, `id` , `number` , `vs` , `cs` , `text` , `amount`,
`currency` , `connection` , `typ` , `date` , `denniktyp` , 0
FROM `Cash`
WHERE `id_user` = '28'
```

Obr. 4-4 Príklad SQL príkazu na výber riadkov peňažného denníka

Záznamy všetkých pohybov daného užívateľa v aktuálne zvolenom roku sa vypisujú postupne, zoradené podľa dátumu v peňažnom denníku pomocou funkcie *foreach()*. Každý previazaný záznam má svoju hodnotu *denniktyp*, ktorá informuje o stĺpci, v ktorom je finančný pohyb zaradený. Pokiaľ pohyb nie je spárovaný, ide o osobnú spotrebu, teda náklady alebo výdavky neovplyvňujúce základ dane.

Peňažný denník pracuje s aktuálnymi zostatkami na bankových účtoch aj pokladni v danom roku. Tieto údaje sú ukladané v tabuľke *Yearbalance*. Hodnoty sa nastavujú v položke nastavenia v navigácií. Zostatky na účtoch k 1.1. zvoleného roka sú hodnoty, ku ktorým sa pripočítavajú kredity alebo debety pohybov.

Posledným riadkom peňažného denníka je súčet kreditných a debetných operácií na účte. Neberie sa ohľad na ovplyvniteľnosť základu dane.

4.5.8 Stav účtovníctva podnikateľa

Pre informácie o stave účtovníctva sú vyberané sumy nákladov a výdavkov vo vybratom účtovnom roku, ktoré sú spárovane a majú vplyv na základ dane. Na výpočet sumy nákladov je použitý SQL príkaz SELECT, kde sa využíva funkcia SUM. V models *Accountmovements* aj *Cash* sú vytvorené metódy *getsum()* na zistenie súčtu kreditov, resp. debetov pre určený rok.

Celkový základ dane je daný súčtom všetkých príjmov zníženým o súčet všetkých nákladov. Uplatňuje sa tu aj odpočítateľná položka, ktorá je čerpaná pre každý rok z tabuľky *YearSetting*. Zo základu dane je vypočítavaná daň, a to vo výške 19% zo základu dane.

4.5.9 Modulové pridanie grafického vzhľadu

V nastaveniach je možné pridať vlastnú všeobecnú šablónu. Je možné si vytvoriť vlastný modul alebo použiť pripravený. Modul musí obsahovať obrázky a štýly pre grafickú farebnosť webu v správnej adresárovej štruktúre v ZIP formáte. Pridaním ZIP súboru s týmito dátami sa farebnosti webu rozšíria o tento štýl.

Na spracovanie ZIP súboru je vyžívaná knižnica *pclzip*. Po nahratí súboru je súbor rozbalený do aktívneho adresára `public/css/model/` pomocou metódy *extract()*. V tomto aktívnom adresári sú vyhľadávané rozbalené moduly, ktoré sú správne nahraté. Pokiaľ je nájdený graficky rozbalený modul, automaticky s ním pracuje a zobrazuje ho.

4.5.10 Ostatné doplnkové funkcie

Účtovný systém ponúka možnosť prehľadu aktuálneho kurzového lístka Európskej centrálnej banky. Kurzový lístok je spracovávaný z adresy http://www.ecb.europa.eu/stats/eurofxref/eurofxref-daily.xml. Ide o XML súbor, v ktorom sú údaje o hodnotách mien, ktoré sú spracované a zobrazované.

Pre systém je navrhnuté fórum, ktoré je možné aktivovať po úprave zdrojových kódov. Nie je však potrebne pre beh systému a pri spustení systému nie je implementované zobrazovanie.

Pod navigáciou sú zobrazované tipy a triky na pracovanie s účtovníctvom. Texty sú preberané z tabuľky *Doyouknow*.

5 Testovanie a vylepšenia

Pri práci s novou technológiou je prirodzené, že sa vyskytnú problémy a chyby pri implementácií. Rozoberať budem nedostatky systému, ktoré boli následne implementované. Taktiež tu nájdeme popísané nedostatky účtovného systému, ktoré je vhodné navrhnúť ako vylepšenia spolu s nápadmi na rozšírenie.

5.1 Priebeh testovania

Testovanie prebiehalo už v štádiu implementácie, kde nie je možné implementovať bez overenia správnosti svojho myšlienkového postupu. Po dotvorení systému ako celku, boli prevádzané testovania, na základe ktorých sa dopracovávali vlastnosti systému.

Testovanie prebiehalo vytváraním 5-tich užívateľských účtov na portáli www.euctovnicka.sk. Títo užívatelia testovali systém vkladaním, editovaním, mazaním dát a párovaním dokumentov. Prístupy k testovacím účtom sú:

1. Login: test@test.sk, heslo: gigant
2. Login: test1@test.sk, heslo: gigant
3. Login: test2@test.sk, heslo: gigant
4. Login: test3@test.sk, heslo: gigant
5. Login: test4@test.sk, heslo: gigant

Nedostatky boli rôzne. Nešlo hlavne o nedostatky systému, ale skôr o nedostatky po strane užívateľskej. Veľmi často sa stávalo, že po vyhodení chyby pri validovaní dát sa odoslané dáta neuchovávali vo formulári. Tieto námety na vylepšenia boli dopracované následnou implementáciou do systému.

Ďalším nedostatkom, ktorý bol pri práci so systémom zistený je zobrazovanie aktuálnej pečiatky pri generovaní faktúry. V nastaveniach je možné meniť aktuálnu pečiatku. Avšak pri generovaní faktúr bola braná aktuálna pečiatka v dobe vytvorenia faktúry. Po následnej zmene pečiatky sa v zozname faktúr ponúkala možnosť zobrazenia PDF faktúry, ale už s aktuálnom pečiatkou. Nedostatok bol ošetrený najskôr vytvorením tabuľky, v ktorej sa uchovali dáta o faktúrach a pečiatkach, kde k faktúre bola priradená konkrétna pečiatka. Pri optimalizácii bola pečiatka priradená priamo k tabuľke *Invoices*.

Nachádzané boli aj nedostatky, ktoré je vhodné dopracovať pre elegantnejšie a kompletnejšie spracovanie jednoduchého účtovníctva. Tieto návrhy sú popísane v nasledujúcej podkapitole ako vylepšenia systému. Požiadavky na účtovný systém boli pri testovaní potvrdené ako vytvorené a funkčné.

Testovaním prešiel aj XHTML a CSS kód, ktorý sa testoval pod operačným systémom Windows XP a Vista na prehľadávačoch Internet Explorer 8, Firefox 3.5 a Opera 10.53. Web bol otestovaný aj pod Linuxom, konkrétne pod operačnom systéme Ubuntu 9.10 s internetovým prehľadávačom Firefox 3.5. Valídnosť web stránky je možné otestovať na tejto linke: http://validator.w3.org/, kde je možné overiť si valídnosť HTML. Kaskádové štýly je možné overiť na adrese http://jigsaw.w3.org/css-validator/. Vytvorené šablóny sú až na embed tagy videa z youtube.com valídne.

Celkové hodnotenie testov hodnotím kladne, nakoľko boli odstránené nedostatky systému. Vytvorený účtovný systém splňuje požiadavky definované zadaním.

5.2 Navrhované vylepšenia účtovného systému

Počas tvorby, ale aj pri samotnom návrhu sa ráta s prípadným rozšírením projektu. Pre kompletné spracovanie účtovníctva a jeho zjednodušenie sú navrhnuté tieto vylepšenia:

- **Požívať viaceré meny** – pri tvorbe vkladaní informácií o cenách je možne vložiť cenu iba v eurách. Pre prípady, že podnikateľ má účet v inom štáte a prijíma platby aj v inej mene je vhodne dopracovať možnosť vytvárania faktúr aj v inej mene ako EUR.
- **Generovanie ostatných dokumentov pre jednoduché účtovníctvo** – ako príjmové a výdavkové lístky, daňové priznania a ročné zúčtovania. Momentálna verzia dovoľuje iba pracovať s informáciami o týchto dokladoch, ale nevytvára ich.
- **On-line prepojenie s bankovým účtom podnikateľa** – pri tvorbe vkladaní informácií o cenách je možne vložiť cenu iba v eurách. Pre prípady, že podnikateľ má účet v inom štáte a prijíma platby aj v inej mene je vhodne dopracovať možnosť vytvárania faktúr aj v inej mene ako EUR.
- **Spracovanie účtovníčka pre platcov DPH** – niektorí podnikatelia sú platcami DPH pre zahraničie. Takýto podnikateľ musí vydávať štvrťročné výkazy. Tieto výkazy by mohli byť generované systémom.
- **Mzdový systém** – systém na správu miezd zamestnancov.
- **Informátor zmien v zákone** – informátor o zmenách v zákonoch.
- **Prepojenie s internetovým obchodom alebo iným systémom** – vytvorenie systému, ktorý napojením na systém euctovnicka.sk bude automaticky a samostatne spracovávať účtovníctvo.
- **Použitie vo viacerých krajinách s rôznymi zákonmi**.

6 Záver

Cieľom tejto práce bolo implementovať účtovný systém dostupný z internetu, ktorý by umožňoval spracovanie jednoduchého účtovníctva. Tento systém sa podarilo vytvoriť a výsledkom je webový portál www.euctovnicka.sk, ktorý ponúka svoje funkcie všetkým užívateľom internetu zdarma.

Výhodou systému je aj fakt, že je možné priamo v ňom vytvárať faktúry svojim odberateľom a zasielať im ich v elektronickej forme na e-mail. Generovanie faktúr je možné do formátu PDF, ktorý si zachováva svoj formát aj dáta pri prezeraní dokumentu nezávisle na operačnom systéme či verzii internetového prehľadávača. Spolu s adresárom kontaktov odberateľov je účtovný systém užitočný nástroj aj pri spracovávaní informácií o svojich odberateľoch. System splňuje požiadavky so spracovaním informácií o finančných pohyboch a ich dokladoch. S využitím párovania s našepkávaním je možné jednoducho spracovať svoje účtovníctvo a generovať peňažný denník. Dovolím si tvrdiť, že dostupnosťou dát celosvetovo v prostredí internetu je tento systém jedinečný.

Spolupráca s AJAX knižnicami je pre užívateľa systému nápomocná. Našepkávanie pri párovaní a autocomplete pri filtrovaní je prijemným vylepšením funkčného systému. Pomocou rýchleho kontaktu je možné vytvárať faktúru odberateľovi na dve kliknutia.

Účtovný systém bol riadne otestovaný s testovacími dátami, ale aj dátami podnikateľa. Je vhodné dopracovať rozšírenia spomenuté v kapitole č. 5, aby sa systém stal vhodným pre všetkých podnikateľov vyžívajúci systém jednoduchého účtovníctva.

Pracovanie na projekte mi dalo skúsenosti a informácie s prácou účtovníka. Najväčší prínos malo získanie znalostí pracovať so Zend Frameworkom, ktorý budem v budúcnosti využívať pri implementácií rozsiahlejších webov. Budem však zvažovať veľkosť projektu, nakoľko sa mi zdá spracovávanie dát Zend Frameworkom pomalé a náročné na zdroje servera. Som však spokojný s dostupnosťou informácií o komponentoch a manuáloch. Získal som si vzťah k MVC architektúre, ktorá je určite prehľadnejšia ako formát a dáta natlačené v jednom súbore.

Systém je potrebné udržiavať, nakoľko sa zákony menia. Odpočítateľnú položku je potrebné nastavovať pre každý nový rok. Taktiež sa môžu prejaviť chyby neobjavené pri testovaní. Demonštrácia modulového pridania grafických štýlov však ubezpečuje, že systém je možné meniť a pridávať doň nové funkcie.

Som vďačný za informácie, ktoré som prijímal a na ich základe tvoril projekt. Teší ma, že výsledok práce využijem vo vlastnom podnikaní pre svoje účtovníctvo a obdobne ho môžu využívať aj ostatní malí podnikatelia. V tom vidím prínos aj pre spoločnosť.

Literatúra

[1] HRUŠKA, Tomáš; KŘIVKA, Zbyněk. *Informační systémy (IIS,PIS) : Pojem informačního systému Data Procesy Transakce Studijní opora* [online]. listopad 2008. Brno : 2006, listopad 2008 [cit. 2010-05-06]. Dostupné z WWW: <https://www.fit.vutbr.cz/study/courses/WAP/private/opory/OporaIIS2PISPojemDataProcesyTransakce.pdf>.

[2] *Ako podnikať* [online]. 2006, 05. máj 2010 [cit. 2010-05-09]. Dostupné z WWW: <www.akopodnikat.sk>.

[3] CASTAGNETTO, Jesus, et al. *PHP Programujeme profesionálně*. Praha : Computer Press, 2001. 656 s. ISBN 80-7226-310-2.

[4] *Zend Framework* [online]. 2010 [cit. 2010-05-10]. Dostupné z WWW: <http://framework.zend.com/>.

[5] KING, Andrew B. *Zrychlete své WWW stránky*. 2004. Brno : Zoner press, 2004. 446 s. ISBN 80-86815-02-1.

[6] *mPDF* [online]. 2010 [cit. 2010-05-10]. Dostupné z WWW: <http://mpdf.bpm1.com/>.

[7] *BLACKHOLE.sk* [online]. 2007-12-23 [cit. 2010-05-10]. Začíname so Zend Frameworkom – tutoriál [1/3] . Dostupné z WWW: <http://blackhole.sk/topiczaciname-so-zend-frameworkom-tutorial-1-3>.

[8] *JQuery: The Write Less, Do More, JavaScript Library* [online]. 2010 [cit. 2010-05-10]. Dostupné z WWW: <http://jquery.com/>.

Zoznam príloh

Príloha 1. Ukážka vygenerovaného PDF dokumentu
Príloha 2. Návod na spustenie
Príloha 3. CD s elektronickou verziou technickej správy, zdrojovými textami informačného systému, skriptom pre MySQL databázu so zdrojovými kódmi

Príloha 1. Ukážka vygenerovaného PDF dokumentu

Faktúra č. 201100003

Dodávateľ:		**Odberateľ:**	
Tomas Kapicak, mutne 210 02963 Mutne		Ján Novák Ceská 1 6000 Bratislava	
IČO:	41963555	IČO platiteľa:	11111111
DIČ:	1478523690	DIČ platiteľa:	1234567890
		IČ-DPH platiteľa:	SK1234567890
Č. účtu:	124578/0200	Variabilný symbol	
		Konštantný symbol	0308
Dátum:		- vyhotovenia dokladu:	17.05.2010
- splatnosti:	31.05.2010	- dodania služby:	17.05.2010

Zasielame Vám predfaktúru za služby.

Označenie položky	Množstvo	Cena za kus	Celkom
Fakturujem Vám za murárske práce za mesiac apríl	1	1234.00€	1234.00€
Celkom			1234.00€
Celkom k úhrade			**1234.00€**

Dôležité upozornenie: Ak fakturovaný poplatok nebude v plnej výške pripísaný na účet dodávateľa najneskôr do termínu splatnosti, nárok na vykonanie služby sa ruší.
V prípade akýchkoľvek pochybností o správnosti údajov uvedených na tejto faktúre nás prosím kontaktujte e-mailom na t@kapicak.com.

Vystavil:

Ďakujeme Vám za využívanie našich služieb.

Príloha 2. Návod na spustenie

1. Dáta z adresára *webdata* v hlavnom adresári CD nahrať na server podporujúci PHP5 a MySQL5. Prípadne naistalovať webový server s PHP a MySQL na vlastnom počítači.
2. Nastavit v súbore *application.ini* v adresári *application/configs* nastavenia pre prístup k MySQL databáze.
3. Naimportovať do databazy dáta zo súboru *sql.sql* z hlavného adresára na CD.
4. Nastaviť prístupové práva adresárom pre zápis scriptom pre adresáre *public/captcha*, *public/css/model* a *public/images/peciatky*. (napríklad chmod 757 adresar).
5. PHP direktívu allow_url_fopen nastaviť na ON
6. Spustiť web cez internetový prehľadávač.