Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Návrh a implementace evidenčního systému pro nakladatelství

Václav Štengl

Vedoucí práce: Ing. David Buchtela, Ph.D.

3. května 2015

Poděkování

Rád bych poděkoval vedoucímu této bakalářské práce Ing. Davidu Buchtelovi, Ph.D. za poskytnuté rady.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 3. května 2015 .....

Odkaz na tuto práci

Štengl, Václav. *Návrh a implementace evidenčního systému pro nakladatelství*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato bakalářská práce je zaměřena na návrh, implementaci a zhodnocení dosažených ekonomicko-manažerských přínosů webové aplikace, která má za úkol nahradit stávající způsob vedení skladové evidence v knižním nakladatelství.

V práci lze nalézt potřebný teoretický základ, který je následovaný výčtem požadavků kladených na funkčnost a rozsah aplikace. Dále soupis existujících řešení pro vedení skladové evidence a zhodnocení jejich vhodnosti pro potřeby nakladatelství. Navazuje sekce obsahující podrobnější popis návrhu a implementace aplikace. Včetně použité architektury, výběru použitých technologií a stručného popisu implementace. Práce na závěr obsahuje testování aplikace, zhodnocení ekonomicko-manažerských přínosů a instalační příručku.

Klíčová slova evidenční systém pro knižní nakladatelství, open source evidenční systém, evidence zásob, skladová evidence

Abstract

This thesis is focused on design, implementation and evaluation of economical and administrative benefits achieved by a web application. This application is supposed to replace the old system for inventory management used in book publisher's.

The thesis contains the necessary theorectical basis and list of requirements for functionality and the extent of the application. Followed by description and evaluation of existing solutions for inventory management administration. Next section covers description of the design and implementation, including the utilized architecture and technologies. In the end, the thesis contains a section about the application testing, evaluation of economical and administrative benefits and installation manual.

Keywords inventory management for books's publisher, open sourse inventory management, inventory management

Obsah

Seznam obrázků

Seznam tabulek

Úvod

„IS je jedním z článků konkurenceschopného fungování firmy“ [1]. Vhodný IS napomáhá firmám ke značným časovým, finančním, personálním (případně i dalším) úsporám a zvýšení efektivity při běžném fungování firmy. IS však nemusí přinášet užitek pouze formou úspor nebo zvýšení efektivity. Může napomoci udržet a nebo získat nové zákazníky a tím napomoci rozvoji firmy. Všechny vyjmenované výhody jsou pro firmy vítané zlepšení a pomáhají firmám získat lepší postavení než jejich konkurence a tím následně upevnit a rozšířit své podnikání.

Tato bakalářská práce se zabývá návrhem a vytvořením IS, který by přinesl některé z uvedených výhod knižnímu nakladatelství. IS v kontextu této práce je open source webová aplikace, která má za cíl nahradit stávající způsob vedení skladové evidence v knižním nakladatelství. Pod návrhem a vytvořením IS se ukrývá spousta operací. Na samotném počátku je potřeba získat a následně analyzovat uživatelské požadavky na aplikaci. Na základě těchto požadavků vytvořit návrh aplikace a vybrat nejvhodnější technologie a frameworky [1] pro uskutečnění implementace. Dalším krokem je samotná implementace a neopominutelné testování aplikace. Pro snadné nasazení aplikace v reálném prostředí je potřeba k aplikaci dodat příručku pro instalaci.

[1] Struktura, která napomáhá při vytváření projektů.

Kapitola 1

Cíl práce

Cílem práce je ze shromážděných a analyzovaných požadavků navrhnout, implementovat, otestovat a zhodnotit ekonomicko-manažerské přínosy webové aplikace, která po svém dokončení a nasazení do provozu nahradí stávající způsob vedení skladové evidence v knižním nakladatelství.

Pro uskutečnění tohoto cíle je nutné splnit zejména zadané požadavky na funkcionalitu. Hlavní požadavky na funkcionalitu jsou následující: evidování dokladů o příjmu zásob na sklad, evidování dokladů o výdeji zásob ze skladu, export dat ve formátu PDF a CSV, vedení skladové evidence ve více skladech, oceňování zásob pomocí metod FIFO a váženého průměru. Pro splnění všech uvedených požadavků je nutné uskutečnit několik fází vývoje. Fáze jsou popsány v jednotlivých kapitolách práce. Kapitoly jsou uspořádány v chronologickém pořadí fází.

Prní kapitola se zabývá nastudováním problematiky skladové evidence v souladu se zákony České republiky a nese název **Skladová evidence** 2. Další kapitola se nazývá **Rešerše** 3 a obsahuje soupis možných řešení skladové evidence pro knižní nakladatelství. Kapitola **Analýza a návrh** obsahuje dvě následující fáze. První fáze je nazvána **Analýza požadavků** 4.1. Zde jsou popsány finální požadavky a jejich rozdělení do skupin podle popsaných kritérií. Druhá fáze nese název **Návrh** 4.3 a obsahuje popis použité metodiky pro vývoj, architekturu aplikace a role uživatelů. Následuje kapitola **Realizace** 5, která obsahuje popis výběru technologií a samotné implementace aplikace. Pro ověření funkčnosti zdrojových kódů následuje kapitola **Testování** 6. Zde jsou uvedeny použité druhy testů a scénáře pro průchod akceptačními testy. Kapitola **Zhodnocení ekonomicko-manažerských přínosů** 7 obsahuje přínosy, kterých lze dosáhnout využíváním aplikace a náklady spojené s provozem a vývojem aplikace. Pro pohodlné nasazení aplikace jsem přidal kapitolu **Instalační příručka** 8

Kapitola 2

Skladová evidence

2.1 Definice pojmů

Následující definice pojmů jsou využity při popisu skladové evidence.

Účetní jednotka dle [2] Účetní jednotka je subjekt, který je definován v zákoně o účetnictví:

- Fyzická osoba
  - jeden konkrétní člověk
  - platí daň z příjmu fyzických osob
  - do obchodního rejstříku se zapisuje nepovinně
  - objektem obchodněprávních vztahů je sám živnostník, který firmu řídí a sám v ní pracuje
- Právnická osoba
  - společnost několika osob, která je oprávněná vstupovat do právních vztahů a jednat svým jménem
  - platí daň z příjmu právnických osob
  - do obchodního rejstříku se zapisuje povinně
  - objektem obchodněprávních vztahů je právnická osoba, která firmu řídí, ale nemusí v ní pracovat

Účetní období dle [2] období, za které účetní jednotka vyhotovuje účetní závěrky

- trvá zpravidla jeden rok (respektive 12 po sobě jdoucích kalendářních měsíců)
- může být kratší - například při zániku účetní jednotky, ale i delší
- při změně typu účetního období
- následující účetní období musí přímo navazovat na předchozí

2.2 Skladová evidence

Skladová evidence je proces zabývající se oceňováním, zaznamenáváním pohybů a stavu zásob na skladech spadajících pod účetní jednotku. Pro přehlednost se tato evidence obvykle dělí podle jednotlivých skladů a jednotlivých účetních období. Stav skladů na konci předchozího účetního období se přenáší jako počáteční stav pro následující účetní období.

V následujících částech této kapitoly si postupně projdeme co jsou zásoby, jaké existují typy pohybů na skladě a jakými způsoby se zásoby oceňují. Účtování zásob je mimo rámec této práce.

2.3 Co vše je zásobou?

Dle [3, p. 11] se za zásoby označují následující položky:

- Položka materiál zejména obsahuje:
  1. suroviny, to je základní materiál, které při výrobním procesu přecházejí zcela nebo zčásti do výrobku a tvoří jeho podstatu,
  2. pomocné látky, které přecházejí také přímo do výrobku, netvoří však jeho podstatu, například lak na výrobky,
  3. látky, kterých je zapotřebí pro zajištění provozu účetní jednotky, například mazadla, palivo, čisticí prostředky,
  4. náhradní díly včetně náhradních dílů určených k výměně komponenty,
  5. obaly a obalové materiály, pokud nejsou účtovány jako dlouhodobý majetek nebo zboží,
  6. další movité věci s dobou použitelnosti jeden rok a kratší bez ohledu na výši ocenění,
  7. samostatné hmotné movité věci a soubory hmotných movitých věcí s dobou použitelnosti delší než jeden rok, považované za drobný hmotný majetek, o kterém účetní jednotka účtuje jako o zásobách,
  8. pokusná zvířata.
- Položka nedokončená výroba a polotovary obsahuje
  1. produkty, které prošly jedním nebo několika výrobními stupni a nejsou již materiálem, nejsou však dosud hotovým výrobkem; tato položka rovněž obsahuje nedokončené činnosti, při nichž nevznikají hmotné produkty,
  2. odděleně evidované produkty, to je polotovary, které dosud neprošly všemi výrobními stupni a budou dokončeny nebo zkompletovány do hotových výrobků v dalším výrobním procesu účetní jednotky.

- Položka výrobky obsahuje věci vlastní výroby určené k prodeji nebo ke spotřebě uvnitř účetní jednotky.
- Položka mladá a ostatní zvířata a jejich skupiny obsahuje zvířata a jejich skupiny včetně jatečných zvířat, která nejsou vykazována jako „Dospělá zvířata a jejich skupiny“ nebo „Materiál“ nebo „Zboží“.
- Položka zboží obsahuje movité věci a zvířata, nabyté za účelem prodeje, pokud účetní jednotka s těmito věcmi a zvířaty obchoduje. Položka obsahuje dále výrobky vlastní výroby, které byly aktivovány a předány do vlastních prodejen, a zvířata vlastního chovu, která dospěla, byla aktivována a jsou určena k prodeji s výjimkou jatečných zvířat. Položka obsahuje též nemovité věci, které účetní jednotka, jejímž předmětem činnosti je nákup a prodej nemovitých věcí, nakupuje za účelem prodeje a sama je nepoužívá, nepronajímá a neprovádí na nich technické zhodnocení.
- Položka poskytnuté zálohy na zásoby obsahuje krátkodobé a dlouhodobé zálohy a závdavky poskytnuté na pořízení zásob.

2.4 Co vše evidujeme u zásob?

Pro potřeby podnikání můžeme evidovat prakticky jakékoliv potřebné údaje. Minimálně je však dle [4] nutné evidovat:

- přesný název,
- datum pořízení,
- datum naskladnění,
- datum vyskladnění,
- datum převodu do jiného skladu nebo převedení materiálu ke zpracování jiné účetní jednotce,
- ocenění a údaje o množství.

2.5 Druhy pohybů zásob na skladě

Pohyby na skladě lze rozdělit na dva základní druhy. První je **příjem** zásob a druhý jejich **výdej**. Existují další druhy pohybů na skladě, ale ty můžeme pro potřeby této práce uvažovat jako kombinaci těchto dvou základních druhů. Pro každý uskutečněný pohyb na skladě je potřeba vytvořit záznam o tomto pohybu podle následujícího výčtu dle [4]:

- Pro každý příjem zásob vytvoříme příjemku zásob.

- Pro každý výdej zásob vytvoříme výdejku zásob.
- Při převodu zásob mezi sklady účetní jednotky vytvoříme převodku mezi sklady (pro potřeby této práce můžeme uvažovat jako výdej zásob z jednoho a příjem zásob pro druhý sklad).
- Při vrácení zboží vytvoříme vratku (pro potřeby této práce můžeme uvažovat jako příjem na sklad).

2.6 Oceňování zásob

Účetní jednotka oceňuje zásoby **pořizovací cenou**. Tato cena vzniká z ceny pořízení zásob a je navýšena o vedlejší náklady. Dle [3] jsou takové náklady zejména přepravné vyúčtované dodavatelem nebo provedené účetní jednotkou, provize, clo a pojistné. Tímto postupem jsme stanovili pořizovací cenu pro nově pořízené zásoby. Tuto cenu však musíme pomocí oceňovací metody sloučit s pořizovací cenou zásob, které již máme na skladě.

2.6.1 Oceňovací metody

V rámci této práce uvedu metody oceňování, které se vyskytly v požadavích na IS.

Vážený aritmetický průměr Pořizovací cenu zásob na skladě P získáme podle vzorce:

$$P = \frac{ps * ms + pn * mn}{ms + mn} \tag{2.1}$$

Kde jednotlivé proměnné představují:

ms Množství zásob jednoho druhu na skladě.

mn Množsví zásob jednoho druhu v příjemce zásob na sklad.

ps Pořizovací cena jednoho druhu zásob na skladě za jednotku.

pn Pořizovací cena jednoho druhu zásob v příjemce zásob na sklad za jednotku.

FIFO Pořizovací cena zásob na skladě se určuje dle pořizovací ceny nejstarších zásob stejného druhu na skladě. Po vyčerpání nejstarších zásob se cena aktualizuje podle nových nejstarších zásob stejného druhu.

2.7 Využití skladové evidence

Zejména slouží pro potřeby účetní jednotky.

- Pro lepší **přehlednost a správu** skladovaných zásob.

- Statistické **zpracování informací**.
- Pro provedení **inventury**. Jelikož je nutné porovnat skutečný stav skladu a evidovaný stav.
- Pro **účtování zásob**.

Z těchto důvodů je potřeba, aby skladová evidence byla vedena **pečlivě, nepřetržitě a průkazně**.

Kapitola 3

Rešerše

3.1 Řešení bez využití SW

Skladovou evidenci je možné zaznamenávat pomocí skladových karet. To však znamená ruční vyplnění příslušného typu pohybu na skladě do skladové karty a následné uchování karty pro pozdější využití. Tento způsob vedení skladové evidence se používal hlavně v minulosti a dnes je na ústupu. Hlavní důvody pro přechod na jiná řešení jsou zejména:

- možná chybovost, nečitelnost záznamu, nesnadná oprava chyby,
- ztráta skladové karty nebo její poškození,
- potřeba úložného prostoru pro uchovávání skladových karet.

3.2 Řešení využívající SW

3.2.1 Tabulkový procesor

Prvním možným řešením pro vedení skladové evidence je použití tabulkového procesoru. Pro představu bych uvedl asi nejznámějšího zástupce v této oblasti a tím je MS Excel [5]. Tento program umožňuje velmi snadno a rychle začít vést skladovou evidenci. Nejsou potřeba větší finanční výdaje ani podrobné znalosti programu. Stačí jen pár kliknutí a základy skladové evidence jsou na světě.

Toto řešení však skrývá mnohá úskalí, která na první pohled nemusí být patrná. Velké množství záznamů v jednom souboru nebo mnoho souborů s menším počtem záznamů budou nepřehledné a neefektivní pro práci. Uživatel může snadno přijít o svá data kvůli nepozornosti, pádu programu nebo smazáním souboru s daty. Tyto nedostatky je možné odstranit pomocí zálohy souboru. Pak je ovšem potřeba pravidelně provádět zálohování originálního souboru.

Dále chybí kontrola vstupu. Pro větší firmy navíc toho řešení nemusí být vhodné, jelikož může vzniknout potřeba pracovat se záznamy z několika stanic zároveň.

3.2.2 Obený účetní SW pro vedení skladové evidence

Toto řešení zahrnuje využití SW, který se zabývá vedením skladové evidence bez bližší specifikace skladovaných zásob. Takový SW je možné využít pro evidenci prakticky jakýkoliv zásob. Výše zmíněná obecnost takového SW však přináší i problémy.

- Mohou chybět potřebné atributy u skladovaných zásob.
- Poskytnutá funkcionalita může být pro potřeby firmy nedostatečná.

Úprava takového účetního SW pro potřeby firmy se zdá být poměrně cenově náročná a ne vždy možná. Proto se místo úprav účetního SW využívají další prostředky, které doplňují požadované vlastnosti. Například již zmíněný tabulkový procesor.

Následují příklady ekonomických SW. Zhodnocení jednotlivých ekonomických SW je provedeno oproti požadavkům nakladatelství.

3.2.2.1 STEREO - ekonomický software

STEREO ekomický software [6] umožňuje vést skladovou evidenci jako jednu ze svých funkcionalit. Skladová evidence v tomto programu je určena pro co nejširší zákaznické spektrum.
Zhodnocení:

- ✓ Příznivá cena za poskytnutou funkcionalitu. STEREO obsahuje mnohem více než jen vedení skladové evidence.
- ✗ Záznamy zásob, které lze pomocí tohoto programu vytvořit jsou pro potřeby nakladatelství nedostatečně detailní.
- ✗ Není možné vyhledávat podle atributů evidovaných u zásob.
- ✗ STEREO nepodporuje práci z více stanic zároveň. Pro sdílení dat je potřeba sdílet soubor, který vznikne uložením aktuálního stavu.
- ✗ Nepodporuje import dat do programu.
- ✗ Pro plnohodnotné použití programu je nutné program zakoupit. Licence se dá pořídit od 2 000 Kč. U programu existuje i verze na vyzkoušení programu, ta však omezuje počet vytvořených záznamů.
- ✗ Není open source. Určený jen pro platformu Windows [7].

3.2.2.2 DUEL Modul SKLADY

Systém evidence skladového hospodářství DUEL [8] podporuje dle [8]:

- ✓ Evidování neomezeného množství skladů.
- ✓ Práci se všemi sklady nad jedním dokladem.
- ✓ Přehledné sledování všech operací se skladovými zásobami.
- ✓ Evidování skladových zásob v průměrných i pevných cenách, v cenách bez daně nebo s daní.
- ✓ Využívá client-server model pro komunikaci.
- ✗ Absence potřebných atributů u zásob.
- ✗ Není open source. Určený jen pro platformu Windows.
- ✗ Pořizovací cena za licenci je od 3 000 Kč pro jeden počítač.

3.2.2.3 Další aplikace

Na českém trhu existuje více aplikací, které se věnují problematice vedení skladové evidence. Tyto aplikace poskytují podobnou funkcionalitu jako zmíněné produkty za podobnou cenu.

3.2.3 SW určený přímo pro nakladatelství

Následující řešení se bude zabývat SW, který je určen přímo pro nakladatelství (ne nutně knižní). Tyto SW nejvíce vyhovují zadaným požadavkům nakladatelství. Na druhou stranu ovšem narůstá pořizovací cena, jelikož tento SW je určen pro úzké spektrum zákazníků.

3.2.3.1 HELIOS

Nejrozšířenějším zástupcem takovového SW v České republice je HELIOS [9]. „Jedním z hlavních přínosů informačních systémů HELIOS je možnost evidence výroby, zejména nosičů zvukových a zvukově obrazových záznamů — CD, DVD a knih či rozmnoženin uměleckých děl.“ [9] Systém se zdá být profesionálním řešením pro problematiku skladové evidence pro knižní nakldatelství. Na druhou stranu cena takového řešení rozhodně nebude zanedbatelná.

3.3 Zvolené řešení

Na základě výše uvedených možností řešení jsem se rozhodl pro návrh a implementaci nového systému. Toto řešení bude koncipováno jako zdarma dostupná open source webová aplikace. Takové řešení přináší následující výhody:

- ✓ Řešení přímo na míru požadavkům nakladatelství.
- ✓ Pro přístup k systému stačí jen prohlížeč a připojení k internetu.
- ✓ Odpadá potřeba vytvářet zálohy souboru s obsaženou evidencí.
- ✓ Je možná práce více uživatelů zároveň.

Kapitola 4

Analýza a návrh

4.1 Požadavky - úvod

Tato sekce se věnuje požadavkům kladených na aplikaci.

4.1.1 Dělení požadavků

Požadavky na IS popisují:

- budoucí funkcionalitu,
- zabezpečení, výkonnost, dostupnost systému,
- dodržování legislativy,
- mnoho dalších potřebných aspektů pro úspěšné navržení a naimplementování IS.

Pro přehlednost se požadavky rozdělují do kategorií. V každé kategoii jsou pak shromážděny požadavky podle specifickch kritérií. Minimální dělení požadavků je dle [10].

4.1.1.1 Požadavky na funkce (funkční požadavky)

Tento druh požadavků vyjadřuje požadaveky na funkcionalitu systému.

Příklad: IS bude evidovat záznamy o knihách. Záznamy budou zadávány ručně.

4.1.1.2 Požadavky na rozhraní

Tento druh požadavků specifikuje rozhraní poskytovaná systémem a rozhraní, se kterými má systém interagovat. Požadavky v této kategorii se obvykle týkají uživatelských, SW, HW a komunikačních rozhraní.

Příklad: Požadavek na GUI, tedy jak má vypadat rozhraní pro interakci s uživateli.

4.1.1.3 Nefunkční požadavky

Často také nazvané obecné požadavky. Obvykle se jedná o požadavky na použité technologie, zabezpečení, výkon, odezvu, dostupnost atd.

Příklad: Systém v jeden okamžik dokáže obsloužit maximálně 500 uživatelů.

4.1.1.4 Další typy požadavků

Tento druh požadavků obsahuje požadavky, které se nevešly do předchozích kategorií. Nejčastěji se jedná o požadavky ohledně legislativy, vícejazyčnosti.

Příkladem: Požadavek na dodržení vyhlášek České republiky ohledně metod oceňování zásob.

4.2 Požadavky na aplikaci

4.2.1 Funkční požadavky

4.2.1.1 Kniha

Entita reprezentující skutečnou knihu. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **book**.

FK.1 Vytvoření záznamu.

FK.2 Upravení záznamu (všech atributů).

FK.3 Smazání záznamu, pokud není obsažen v jiných záznamech.

FK.4 Zobrazení detailu pro existující záznam:

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů příjemek, výdejek a skladovaných knih obsahujících aktuální záznam.

FK.5 Vytvoření vydání knihy k existujícímu záznamu:

- Nové vydání může mít identické parametry jako původní vydání.
- Zachycení vztahu mezi novým a starým vydáním.

FK.6 Výpis všech vydání knihy pro zadanou knihu.

FK.7 Zobrazení všech záznamů v systému s vyhledáváním podle parametru.

4.2.1.2 Dodavatel/Odběratel

Entita reprezentující odběratele/dodavatele v aplikaci. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **customer**.

FDO.1 Vytvoření záznamu.

FDO.2 Upravení záznamu (všech atributů).

FDO.3 Smazání záznamu, pokud není obsažen v jiných záznamech.

FDO.4 Zobrazení detailu pro existující záznam:

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů příjemek a výdejek obsahujících aktuální záznam.

FDO.5 Zobrazení všech záznamů v systému s vyhledáváním podle parametru.

4.2.1.3 Sklad

Entita reprezentující sklad v aplikaci. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **warehouse**.

FS.1 Vytvoření záznamu.

FS.2 Upravení záznamu (všech atributů vyjma metody oceňování zásob).

FS.3 Zobrazení detailu pro existující záznam:

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů příjemek, výdejek, skladovaných knih obsahujících aktuální záznam.

FS.4 Zobrazení všech záznamů v systému s vyhledáváním podle parametru.

4.2.1.4 Přijemka na sklad

Entita reprezentující příjem zásob na sklad. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **receipt**.

FP.1 Vytvoření záznamu:

- Vyplnění atributů příjemky.
- Přidání 1 až n záznamů přijatých knih ve specifickém množství a ceně.
- Přepočtení stavu skladu a ovlivněných záznamů dle metody oceňování skladu.

FP.2 Upravení záznamu:

- Všech atributů záznamu.
- Při úpravě je možno mazat, přidávat a upravovat záznamy přijatých knih.
- Přepočtení stavu skladu a ovlivněných záznamů dle metody oceňování skladu.

FP.3 Smazání záznamu:

- Aktualizace stavu skladu.

FP.4 Zobrazení detailu pro záznam:

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů s vydanými knihami a vypočtenou celkovou cenou pro každý záznam.
- Zobrazení celkové ceny a množství.

FP.5 Zobrazení a vyhledávání záznamů podle zadaného parametru pro aktivní sklad.

FP.6 Zobrazení a vyhledávání záznamů podle zadaného parametru globálně.

4.2.1.5 Výdejka ze skladu

Entita reprezentující výdej zásob ze skladu. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **release_note**.

FV.1 Vytvoření záznamu:

- Vyplnění atributů výdejky.
- Přidání 1 až n záznamů vydaných knih ve specifickém množství.
- Přepočtení stavu skladu a ovlivněných záznamů dle metody oceňování skladu.

FV.2 Upravení záznamu:

- Všech atributů záznamu.
- Při úpravě je možno mazat, přidávat, upravovat záznamy z vydaných knih.
- Přepočtení stavu skladu a ovlivněných záznamů dle metody oceňování skladu.

FV.3 Smazání záznamu a aktualizace stavu skladu.

FV.4 Zobrazení detailu pro existující záznam.

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů s vydanými knihami a vypočtenou celkovou cenou pro každý záznam.
- Zobrazení celkové ceny a množství.

FV.5 Zobrazení a vyhledávání záznamů podle zadaného parametru pro aktivní sklad.

FV.6 Zobrazení a vyhledávání záznamů podle zadaného parametru globálně.

4.2.1.6 Skladované knihy

Reprezentují knihy na skladě. Obsahují historii skladových pohybů, pořizovací cenu vypočtenou pomocí oceňovací metody a další informace. Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **stored_book**.

FSK.1 Zobrazení detailu:

- Zobrazení všech atributů záznamu.
- Zobrazení záznamů příjemek a výdejek obsahujících aktuální záznam.

FSK.2 Zobrazení a vyhledávání záznamů podle zadaného parametru pro aktivní sklad.

FSK.3 Zobrazení a vyhledávání záznamů podle zadaného parametru globálně.

4.2.1.7 Export dat

FE.1 Export příjemek ze zadaného časového intervalu. Formát výstupu PDF a CSV.

FE.2 Export výdejek ze zadaného časového intervalu. Formát výstupu PDF a CSV.

FE.3 Export vystavené faktrury z výdejky do formátu PDF:

- Možnost stanovení jiné slevy a DPH pro každou vydanou knihu.
- Výpočet splatnosti faktury dle data vystavení faktury a navýšené o splatnost faktur uvedené u odběratele.
- Výpočet ceny bez DPH po slevě pro každý záznam vydané knihy obsažené ve výdejce.
- Informace o odběrateli budou převzaty z výdejky.

FE.4 Export výdejky do PDF.

4.2.1.8 Převod aktuálního stavu skladu do jiného skladu

FTS.1 Aplikace musí podporovat převod aktuálního stavu skladu A do skladu B. Převodem stavu se rozumí vytvoření příjemky pro sklad B z aktuálního stavu skladu A. Následuje přepočet stavu skladu B. Metoda oceňování zásob na skladech se nemusí shodovat.

4.2.1.9 Uživatel

Atributy jsou k nalezení v obrázku 4.3.5 v tabulce s názvem **users**.

FUS.1 Registrace nového uživatele.

FUS.2 Přihlášení uživatele.

FUS.3 Odhlášení uživatele.

FUS.4 Upravení osobních údajů a hesla.

FUS.5 Vygenerování hesla po zadání uživatelského jména a emailu.

FUS.6 Smazání svého účtu.

FUS.7 Aktivní sklad:

- Možnost přepnout pracovní sklad.
- Aplikace si pro uživatele bude pamatovat pracovní sklad mezi přihlášeními.

4.2.1.10 Admin

FA.1 Aktivace/blokace uživatele.

FA.2 Smazání skladu.

FA.3 Nastavení firemních údajů.

FA.4 Změna rolí uživatelů.

FA.5 Smazání uživatele.

4.2.2 Nefunkční požadavky

N.1 Výsledný systém bude mít podobu webové aplikace.

N.2 Při vývoji aplikace je nutné využít SW balíček LAMP [11].

N.3 Validace dat proběhne ihned po odeslání na server.

4.2.3 Požadavky na rozhraní

V této kategorii se nevyskytly žádné požadavky.

4.2.4 Další typy požadavků

D.1 Aplikace musí dodržovat oceňování zásob popsané v [3, p. 37-38] minimálně pomocí metod:

- FIFO
- Vážený průměr - „Pokud jsou u stejného druhu zásob využity způsoby ocenění cenou, která vychází z ocenění jejich úbytků, pak v rámci jednoho analytického účtu zásob je nutno používat pouze jeden způsob ocenění; pokud je využit vážený aritmetický průměr, počítá se nejméně jednou za měsíc.“[3, p. 37-38]

Aplikace musí aktualizovat cenu při každém vytvoření/upravení/smazání příjemky a výdejky.

D.2 Aplikace musí být lokalizována pro český jazyk. Další jazyky nejsou vyžadovány.

4.3 Návrh

4.3.1 Volba metodiky pro vývoj

Vzhledem k rozsahu práce jsem se rozhold využít metodiku zvanou vodopád [12, slide. 6]. Metodika je vhodná pro projekty, které splňují níže uvedené podmínky:

- Všechny nebo velká většina požadavků na systém je známá na počátku vývoje.
- Požadavky se během vývoje nemění a nebo jen v omezené míře.
- Jedná se spíše o menší až středně velký projekt.

Podmínky pro použití vodopádu jsou dodrženy. Pojďme si tedy představit metodiku vodopád podrobněji.

4.3.2 Vodopád

Jedná se o sekvenční metodiku, která se využívá najčastěji při vývoji IS. Dle [12, slide. 6] je základní dělení fází vodopádu následovné:

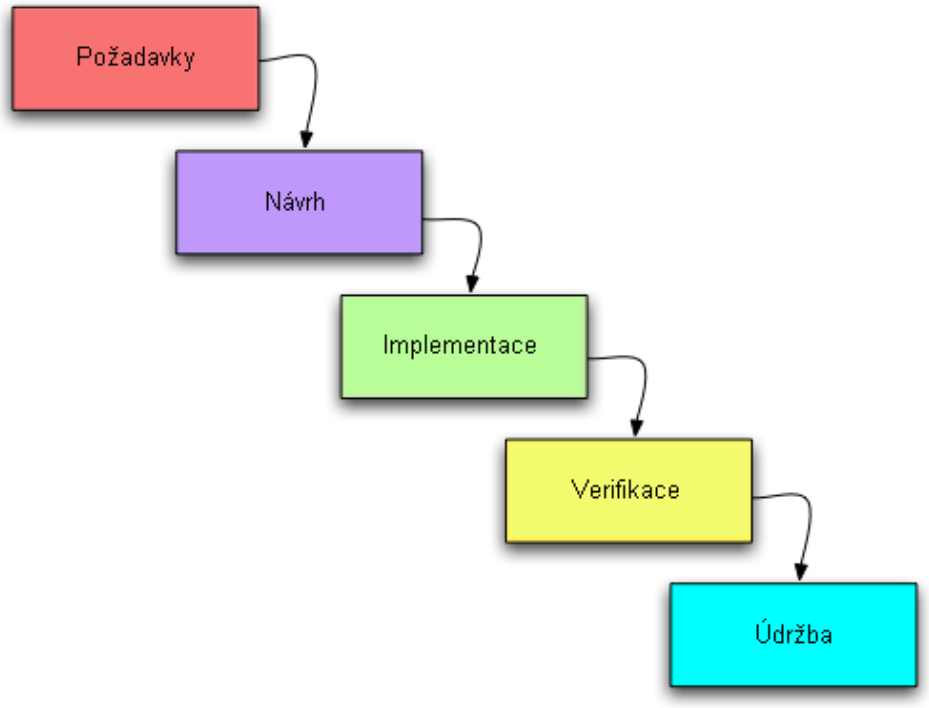
1. Analýza požadavků
2. Design

3. Implementace

4. Testování

5. Provoz a údržba

Odtud pak vodopád získal svůj název. Při vývoji se obvykle již nevracíme do předchozích fází. Až na vyjímky, kdy se můžeme vrátit maximálně o jednu fázi a to je jen v případě, že je to nezbytně nutné. Proces vývoje tedy připomíná pomyslný vodopád. Pro znázornění postupu vývoje při využití metodiky jsem přiložil obrázek 4.1.



Obrázek 4.1: Životní cyklus vodopádu [13]

4.3.3 Architektura

Pro potřeby této práce jsem se rozhodl využít architektonický vzor Model-view-presenter [14] (dále jen MVP). MVP rozděluje aplikaci na tři kooperující součásti.

Model reprezentuje a spravuje informace, se kterými v aplikaci pracujeme pomocí datových struktur. Tyto datové struktury poskytuje presenteru, který se stará o jejich následné využití. Pro snadnější přístup k informacím model obvykle poskytuje CRUD [15] operace. CRUD operace vypadají následovně:

Create Persistence dat.

Retrieve Vyhledání a navrácení požadovaných dat.

Update Aktualizace dat.

Delete Vymazání dat.

View prezentuje uživateli informace, které obdržel od presenteru a zachytává požadavky od uživatele. Požadavky od uživatele následně deleguje presenteru ke zpracování.

Presenter působí jako supervizor pro view and model. Reaguje na požadavky uživatelů, předává view potřebná data z modelu a dává příkazy k aktualizaci dat spravovaných modelem. Komunikace mezi jednotlivými částmi MVP je znázorněna na obrázku 4.2.

4.3.4 Uživatelé systému - role

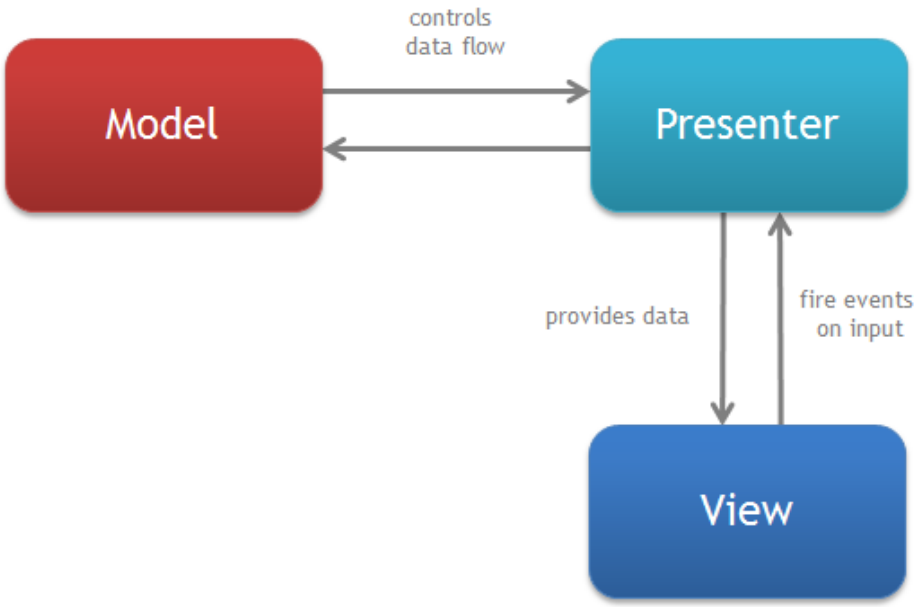
Role uživatelů jsou uvedeny od rolí s nejnižšími pravomocemi (první uvedená role má nejnižší pravomoce). Platí, že uživatel s vyšší rolí má všechna oprávnění uživatelů s nižšími rolemi.

Anonymní uživatel Jakýkoliv návštěvník aplikace, který se nepřihlásil. Pro tohoto uživatele jsou dostupné jen stránky s přihlášením, registrací a obnovou hesla.

Nahlížení Uživatel s pravomocemi nahlížení do existujících záznamů (vyjma seznamu uživatelů) a provádění exportů z aplikace. Akce typu vytvoření, upravení, smazání nejsou pro tohoto uživatele dostupné.

Uživatel Smí vytvářet, upravovat, mazat záznamy. Nevztahuje se na mazání skladů a správu uživatelů.

Admin V menu je takovému uživateli přístupná **administrátorská sekce**. Dovoluje spravovat uživatele systému, smazat sklad nebo provést převod aktuláního stavu skladu do jiného skladu.



Obrázek 4.2: Komunikace MVP [16]

4.3.5 Databázový model

Pro zachycení všech potřebných vztahů a informací jsem vytvořil databázové schéma zobrazené na obrázku 4.3.5.

Obrázek 4.3: Databázový model (vytvořeno pomocí[17])

Kapitola 5

Realizace

5.1 Volba technologií (serverová strana)

Volba technologií je podstatnou součástí při vývoji IS. Zvolené technologie musí být pro účely IS vhodné. Existuje mnoho kritérií, kterými se můžeme při volbě použitých technologií řídit. Při výběru technologií jsem se držel následujících kritérií:

- Výsledný IS musí mít formu webové aplikace.
- Systém bude open source řešení.
- Persistence musí být zajištěna pomocí databáze.
- Pro vytvoření IS je nutné využít SW balíček LAMP.

Balíček LAMP značí:

L = Linux [18] Open source operační systém, který je vhodný pro vývoj a testování aplikací všeho druhu.

A = Apache HTTP server [19] Jedná se o nejrozšířenější webový server. Vhodný nástroj pro vývoj a testování webových aplikací na lokálním počítači.

M = MySQL [20] Jeden z celosvětově nejpoužívanějších open source databázových managerů pro relační databáze.

P = PHP | Perl | Python Programovací jazyk pro tvorbu samotných webových aplikací. V tomto případě máme na výběr z několika jazyků. Každý z nich má své silné a slabé stránky a je nemalý problém určit vhodný jazyk pro implementaci webové aplikace.

5.1.1 Volba programovacího jazyka

PHP

+ snadný jazyk
+ mnoho předpřipravených funkcí pro web. vývojáře
+ kvalitní frameworky
+ rozsáhlá komunita vývojářů
+ již mám zkušenosti s PHP a jeho frameworky
- při špatné údržbě kódu se kód může stát nepřehledným

Python

+ jednotná syntaxe
+ již mám zkušenost s Pythonem
- novější verze jsou nekompatibilní se staršími

Perl

+ víceúčelový jazyk
+ mnoho cest k řešení problému
- pomalý vývoj
- bez řádné dokumentace se kód stává nečitelný velmi rychle
- ještě více cest k vytvoření chyb

Po důkladném zvážení všech pro a proti jsem se rozhodl pro PHP. Nekompatibilita různých verzí Pythonu by při nasazování systému mohla být problematická a nebo způsobit nefunkčnost části systému. Perl kvůli svému pomalému vývoji, špatné čitelnosti kódu při nedostatečné dokumentaci a poměrně snadnému způsobení chyb také neuspěl.

Pro usnadnění práce a zpřehlednění zdrojového kódu navíc nepoužiji jen čisté PHP, ale jeden z PHP frameworků. Využití frameworku má nepopřitelné výhody. Framework obsahuje spoustu předpřipravené funkcionality pro usnadnění vývoje aplikací, napomáhá znovupoužití kódu, přidání knihoven třetí strany je snadné a obsahuje výborné debugovací prostředky.

5.1.2 Volba frameworku

Při volbě frameworku pro bakalářskou práci jsem vybíral celkem ze tří PHP frameworků. Konkrétně z Nette [21], Symfony2 [22] a Zendu [23]. Při výběru frameworku jsem si pro každý framework vytvořil seznam slabých a silných stránek.

Zend Je určen primárně pro větší projekty.

- ✓ rozsáhlé knihovny
- ✗ verze 1 je zastaralá, verze 2 nedokončená
- ✗ pomalý, pokud není přítomen Zend Optimizer nebo Accelerator

Nette Určeno spíše pro menší projekty.

- ✗ pomalý vývoj
- ✗ lokální framework, zejména Česká republika
- ✗ pro práci s databází využívá vlastní systém
- ✗ nepřehledná dokumentace

Symfony2 Určeno zejména pro střední a velké projekty.

- ✓ využívaní anotací
- ✓ využívá Doctrine 2 [24] (objektově-relační mapování objektů)
- ✓ Twig šablony [25]
- ✗ použití mnoha různých anotací může být nepřehledné a chyba se hledá obtížně

V mém rozhodování mě upevnil navíc tento odstavec: Zend 1 je dnes již velmi zastaralý, Zend 2 je nevyspělý a nevidím u něj jasnou vizi. Framework Yii se stále drží ActiveRecordu a nevyužívá Dependency Injection. U Nette mi zase vadí pomalý vývoj, lokálnost, mizerná dokumentace a nepodpora některých moderních postupů (anotace, AOP, controllery jako services atd.). Komunitu Symfony2 tvoří tisíce vývojářů. Framework se rychle rozvíjí, má obsáhlou dokumentaci a mnoho rozšiření (bundles). Jeho jednotlivé komponenty v poslední době začaly využívat nové verze známých open-source řešení. [26] Ve výsledku jsem tedy zvolil framework Symfony2.

5.1.3 Symfony2

Populární Request and Response PHP framework.

5.1.3.1 Modulatira

Symfony2 se skládá z jednotlivých komponent, které se v kontextu frameworku nazývají bundle. Jednotlivé bundly jsou celky, které shromažďují logicky spolu související funkcionalitu. Není tedy problém použít jen část frameworku pro své potřeby nebo přidat potřebou funkcionalitu přidáním vlastních bundlů a nebo bundlů třetí strany. Do bundlů se dělí i vlastní kód aplikace.

5.1.3.2 Filozofie

Symfony2 se snaží ponechat developerům co největší volnost. Prakticky vše lze ovlivnit pomocí konfiguračních souborů a samotné konfigurace lze provést několika různými způsoby.

5.1.3.3 Nejdůležitější bundly Symfony2

Symfony2 obsahuje velké množství technologií. Při práci na aplikaci budu pracovat zejména s následujícími technologiemi.

Doctrine2
Doctrine je PHP projekt, který obsahuje několik knihoven pro ulehčení práce s databází a datovými strukturami zvanými entity. Přesněji řečeno Doctrine2 se orientuje na persistenci dat z entit a následné mapování dat z databáze zpět na entity. Tento proces nazýváme ORM. Pomocí Doctrine2 lze také vyjádřit vztahy mezi entitami. Stručný popis entit a vztahů naleznete 5.3.1.

Twig
Šablonovací engine Twig slouží pro snadné generování požadovaného výstupu. Více v sekci 5.3.5

5.1.3.4 Další potřebné bundly

V požadavcích na aplikace se vyskytl požadavek na export dat do formátu PDF.

KnpLabs/KnpSnappyBundle
Pro export do PDF jsem zvolil bundle **KnpLabs/KnpSnappyBundle** [27], který do apliakce přináší služby pro generování PDF a JPG formátů. Tento bundle je wrapper pro **wkhtmltopdf** [28]. Wkhtmltopdf je WYSIWYG nástroj určený pro příkazovou řádku, který provádí konverzi HTML do PDF a obrázkových formátů. Nástroj je open source pod licencí LGPLv3.

Všechny exporty do formátu PDF v této aplikaci využívají služby výše popsaného bundlu pro konverzi HTML a CSS vytvořeného pomocí Twig šablon do formátu PDF.

5.2 Volba technologií (klientská strana)

5.2.1 HTML

Pro vytvoření obsahu webových stránek jsem se rozhodl použít značkovací jazyk HTML. Jazyk slouží jen pro definici obsahu stránky nikoliv jejího vzhledu nebo chování.

5.2.2 CSS

Jazyk určený pro úpravy vzhledu stránky, avšak ne jejího obsahu. Obvykle se používá pro změnu vzhledu stránek vytvořených pomocí HTML, XHTML a XML. Nezabývá se tedy obsahem stránky, ale pouze jejím vzhledem.

5.2.3 JavaScript

HTML a CSS zajistí obsah a vzhled stránek budoucí aplikace. Pro dosažení plnohodnotné interakce s uživatelem a dynamického chování stránek využiji JavaScript. JavaScript je dynamický jazyk, který dokáže měnit obsah stránek, a tak lépe interagovat s uživatelem. Dále snižuje potřebnou dobu pro přenos a objem přenesených dat díky asynchronní komunikaci (AJAX). AJAX místo přenosu a načtení celé stránky přenáší a načítá jen části stránky.

JavaScript se v aplikaci vyskytne hlavně při potvrzování důležitých operací a při vyhledávání položek, které je realizováno pomocí AJAXu.

5.2.4 HTML, CSS, a JS frameworky

Pro efektivnější práci a profesionálnější vzhled jsem využil v dnešní době již prakticky standard pro stylování webových stránek HTML, CSS a JS framework zvaný Bootstrap [29].

5.2.4.1 Bootstrap

Tento framework obsahuje:

- Velké množství předpřipravených CSS tříd pro snadné pozicování a stylování HTML tagů.
- Komponenty pro vytváření částí webových stránek. Příkladem:
  - navigace,
  - stránkování,
  - upozornění a mnoho dalších.
- Předpřipravené části kódu pro jazyk JavaScript, které napomáhají interakci s uživatelem.

5.2.4.2 jQuery

JavaScript knihovna pro zjednodušení práce s JavaScriptem. Zjednodušuje práci zejména ve vyhledávání a přístupu k HTML elementům, zpracovávání eventů a využívání AJAXu.

Tím končí výčet zvolených technologií pro budoucí implementaci a započíná samotná implementace.

5.3 Implementace

Tato sekce obsahuje stručný popis využitých postupů při implementaci s několika ukázkami.

5.3.1 Entity

Doctrine2 mapuje objekty (využité v aplikaci pro reprezentaci dat) k přidružené relační databázi. Tyto objekty se nazývají entity. Entity jsou v této práci vytvořeny pomocí anotovaných [2] PHP tříd a vztahy mezi entitami a databází jsou vyjádřeny pomocí anotací zdrojového kódu. Data jsou v entitách dostupná jako proměnné. Takové proměnné pak mohou být primitivní typy [3] nebo instance tříd včetně dalších entit. Následující podsekce popisují jak vytvit entity spravované pomocí Doctrine2.

5.3.1.1 Anotace třídy

Nejdříve je nutné použít anotace na samotnou PHP třídu. Tím docílíme, že Doctrine2 začne tuto třídu spravovat jako entitu.

```
use Doctrine\ORM\Mapping as ORM;
/**
 * @ORM\Entity()
 */
```

Označí třídu jako Entitu spravovanou Docrine2.

```
/**
 * @ORM\Entity(repositoryClass="BookRepository")
 * @ORM\Table(name="book")
 */
```

Určí název tabulky, ze které má Doctrine2 čerpat data a třídu, která bude slouži jako repository pro tuto entitu.

[2] Anotace jsou dodatečné/přidané informace k dokumentu.
[3] Například integer, float

5.3.1.2 Anotace proměnných

V případě, že se jedná o primitivní typ, pak anotací určíme název sloupce (nepovinné) v tabulce a případně další potřebné upřesnění.

```
/**
  * @var float
  * @ORM\Column(type="float")
  */
protected $retailPrice;
```

Jedná-li se o entitu, pak je potřeba určit vztahem OneToOne, OneToMany, ManyToMany a k nim inverzními názvy vztah mezi těmito entitami. Následně jméno cílové entity, kterou chceme mít uloženou v proměnné a sloupec, který bude sloužit pro navázání vztahu. Pokud má být vztah oboustranný je potřeba vyplňit atribut inversedBy.

```
//ManyToMany vztah mezi entitami User a Role

//Entity User
/**
  * ManyToMany(targetEntity="Role", inversedBy="users")
  */
protected $roles;

//Entity Role
/**
  * ManyToMany(targetEntity="User", mappedBy="roles")
  */
protected $users;
```

Aby vše fungovalo, je potřeba pro každý atribut vytvořit get a set metody.

Následující příkazy vytvoří databázi a její strukturu podle anotovaných tříd.

1. php app/console doctrine:database:create
2. php app/console doctrine:schema:update --force

Tato sekvence příkazů vytvoří databázi a upraví ji podle anotovaných tříd. Pokud potřebujete jen upravit schéma databáze, stačí využít poslední příkaz.

5.3.2 Repository

Repository jsou objekty pro správu entit v databázi. Jsou reprezentovány PHP třídami, které jsme určili anotací entit a musí být následníci třídy **EntityRepository**. Jejich účelem je shromažďovat úkony potřebné od databáze do jednoho místa. Takové úkony jsou obvykle persistence dat, výběr dat z DB s podmínkami, filtry, uspořádáním.

Tato praktika zvyšuje znovupoužitelnost a přehlednost kódu, jelikož požadavky pro databázi najdeme na jednom místě a ne všude možně. Při implementaci jsem do repozitory přidal i CRUD operace.

Příklad PHP třídy, která by mohla sloužit jako repository.

```
use Doctrine\ORM\EntityRepository;

/**
 * Class DefaultRepository
 */
class DefaultRepository extends EntityRepository {
 //class content
 //query etc..
}
```

5.3.3 Service

Service (služba) je spolu logicky související funkcionalita, kterou jsme shromáždili a rozhodli se ji poskytovat k užití. V Symfony2 se službou může stát prakticky jakákoliv PHP třída. Stačí ji zaregistrovat v souboru services.yml. Služba může být dostupná v celé aplikaci, nebo jen v určitých bundlech. Pokud potřebujeme pro správnou funkčnost naší služby jiné služby, pak uvedeme názvy těchto služeb v services.yml jako argumenty služby.

Příklad zaregistrování služby **book_service**.

```
File: app/config/services.yml

book_service:
    class: Cvut\Fit\Service\BookService
    arguments: [@doctrine.orm.entity_manager]
```

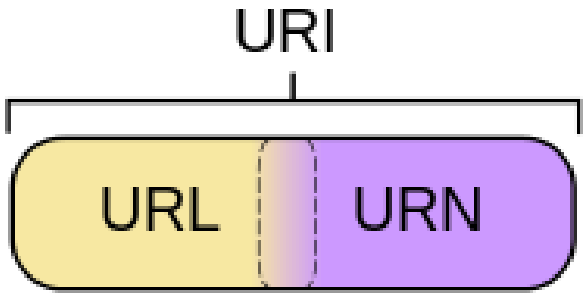
5.3.4 Controller

V Symfony2 je Controller PHP třída, která slouží pro obsluhu příchozích uživatelských požadavků. Tato třída musí být následníkem třídy Controller implementovaném v Symfony2. V Controlleru máme přístup ke kontejneru se službami a můžeme je využívat pro naše potřeby.

Stejně jako v předchozích případech se i zde využívají anotace. V tomto případě popisují, kterou metodu Controlleru využít pro příchozí požadavek. Příchozí požadavek obsahuje řetězec **URI**, který se skládá z podřetězců **URL** a **URN**.

URL Specifikuje umístění zdroje a formu reprezentace.

URN Specifikuje zdroj, který je požadován.



Obrázek 5.1: Znázornění složení URI [30]

Pomocí anotace určíme pro jakou hodnotu URN se má metoda vykonat. Metoda uvedená v příkladu se vykoná pro URN s hodnotou "/book/create".

```
// src/Cvut/Fit/StorageEvidence/Controller/BookController.php

namespace Cvut\Fit\StorageEvidence\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Cvut\Fit\StorageEvidenceBundle\Form\Book\BookType;
use Cvut\Fit\StorageEvidenceBundle\Entity\Book;

class BookController extends Controller {
   /**
    * @Route ("/book/create")
    */
  public function createAction(Request $request) {
    $form = $this->createForm(new BookType(), new Book());
    $twig = 'CvutFitStorageEvidenceBundle:Book\create.html.twig';

    if($form->isSubmitted() && $form->isValid()) {
      //data persistence or another actions
    }

    return $this->render($twig, array(
        'form' => $form->createView()
    ));
  }

}
```

5.3.5 Twig template

Twig template (Twig šablona) zabraňuje opakování kódu. Šablonu naplňíme statickým obsahem a pomocí výpisu proměnných, podmínek, maker, filtrů a průchodů cykly dosáhneme různých výstupů při použití stejné šablony. Je možné generovat nespočet různých formátů výstupů. Nejčastěji se jedná o formáty HTML, XTML, JSON, XML.

Twig šablony obsahují tři druhy bloků speciální syntaxe:

- {% blok vyhodnocení %}
- {{ proměnná k výpisu }}
- {# jednořádkový komentář #}

Vše co není umístěno v těchto blocích se považuje za statický obsah a je součástí výpisu. Příklad Twig šablony 5.3.6.

5.3.6 Formuláře a jejich validace

Vytvoření nového formuláře je v Symfony2 poměrně snadné. Stačí vytvořit PHP třídu, která je následníkem třídy **AbstractType**. V této třídě je následně nutné naimplementovat 3 funkce. Viz příklad.

```
// src/Cvut/Fit/StorageEvidenceBundle/Form/Book/BookType

namespace Cvut\Fit\StorageEvidenceBundle\Form\Book;

use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolverInterface;

class BookType extends AbstractType {

    public function buildForm(FormBuilderInterface $builder, array $options) {
        $builder->add('title', 'text');
    }


    public function setDefaultOptions(OptionsResolverInterface $resolver) {
        $resolver->setDefaults(array(
            'data_class' => 'Cvut\Fit\StorageEvidenceBundle\Entity\Book',
        ));
    }

    public function getName() {
        return 'create_book';
    }
}
```

Funkce **buildForm** vytvoří formulář dle obsahu metody. V tomto případě bude formulář obsahovat textové pole a jeho obsah bude po odeslání uložen do proměnné **title**. Proměnnou title musí obsahovat třída uvedená v metodě **setDefaultOptions**. Jedná se o třídu, která je k nalezení v asociativním poli pod názvem **data_class**. Tato třída nemusí být entita. Ve většině případů však je. Následuje ukázka takové třídy.

```
// src/Cvut/Fit/StorageEvidenceBundle/Entity/Book.php

use Symfony\Component\Validator\Constraints as Assert;
use Symfony\Component\Validator\Context\ExecutionContextInterface;

namespace Cvut\Fit\StorageEvidenceBundle\Entity;

class Book {
    /**
     * @var string
     * @Assert\NotBlank()
     */
    protected $title;

    /**
     * @Assert\Callback
     */
    public function validate(ExecutionContextInterface $c) {
        //additional validation
    }
    //get, set methods
}
```

Formulář využijeme při požadavku od uživatele v metodě Controlleru. Použití viz ukázka metody Controlleru 5.3.4

V metodě Controlleru vytvoříme formulář a uložíme jej do proměnné $form. Proměnná $twig obsahuje cestu k šabloně, kterou využijeme pro vytvoření stránky s formulářem.

```
return $this->render($twig, array(
  'form' => $form->createView()
));
```

Zpřístupní twig šabloně proměnnou s názvem **form**, která po vyspání vytvoří HTML reprezentaci formuláře. Zjednodušená ukázka takové šablony:

```
//src/Cvut/Fit/StorageEvidenceBundle/Resources/views/Book/create.html.twig
<!DOCTYPE html>
<html>
    <head>
        <title>Skladova evidence</title>
    </head>
    <body>
      <h1>Nova kniha</h1>
      {{ form(form) }} {# vypis prommene form pomoci funkce form() #}
    </body>
 </html>
```

5.3.6.1 Validace

Po přijetí vyplněného formuláře dochází k validaci obsahu v metodě Controlleru. Ukázka 5.3.4.

```
if($form->isSubmitted() && $form->isValid()) {
```

```
    //data persistence or another actions
}
```

Metoda **isValid()** provede validaci formuláře. Validuje se podle podmínek uvedených v třídě sdružené k formuláři. Příklad třídy 5.3.6.

@Assert\NotBlank() - zajišťuje, že záznam nezůstane prázdný. Celý seznam assertů naleznete na adrese: `http://symfony.com/doc/current/book/validation.html`. Funkce anotovaná pomocí **@Assert\Callback** slouží pro vykonání složitějších validací.

5.3.7 Shrnutí

Na každý požadavek od uživatele framework Symfony2 vytvoří odpověď. Životní cyklus požadavku až k odpověďi je znázorněn na obrázku 5.3.7. Pojďme si obrázek probrat postupně po jednotlivých zvýrazněných prvcích.

Request Příchozí požadavek od uživatele.

request Objekt representující požadavek uživatele.

Resolve Controller Framework podle URN v požavavku určí metodu Controlleru pro obsluhu požadavku.

Controller Inicializace Controlleru.

Resolve arguments Argumenty obsažené v URN jsou převedeny do požadovaných formátů

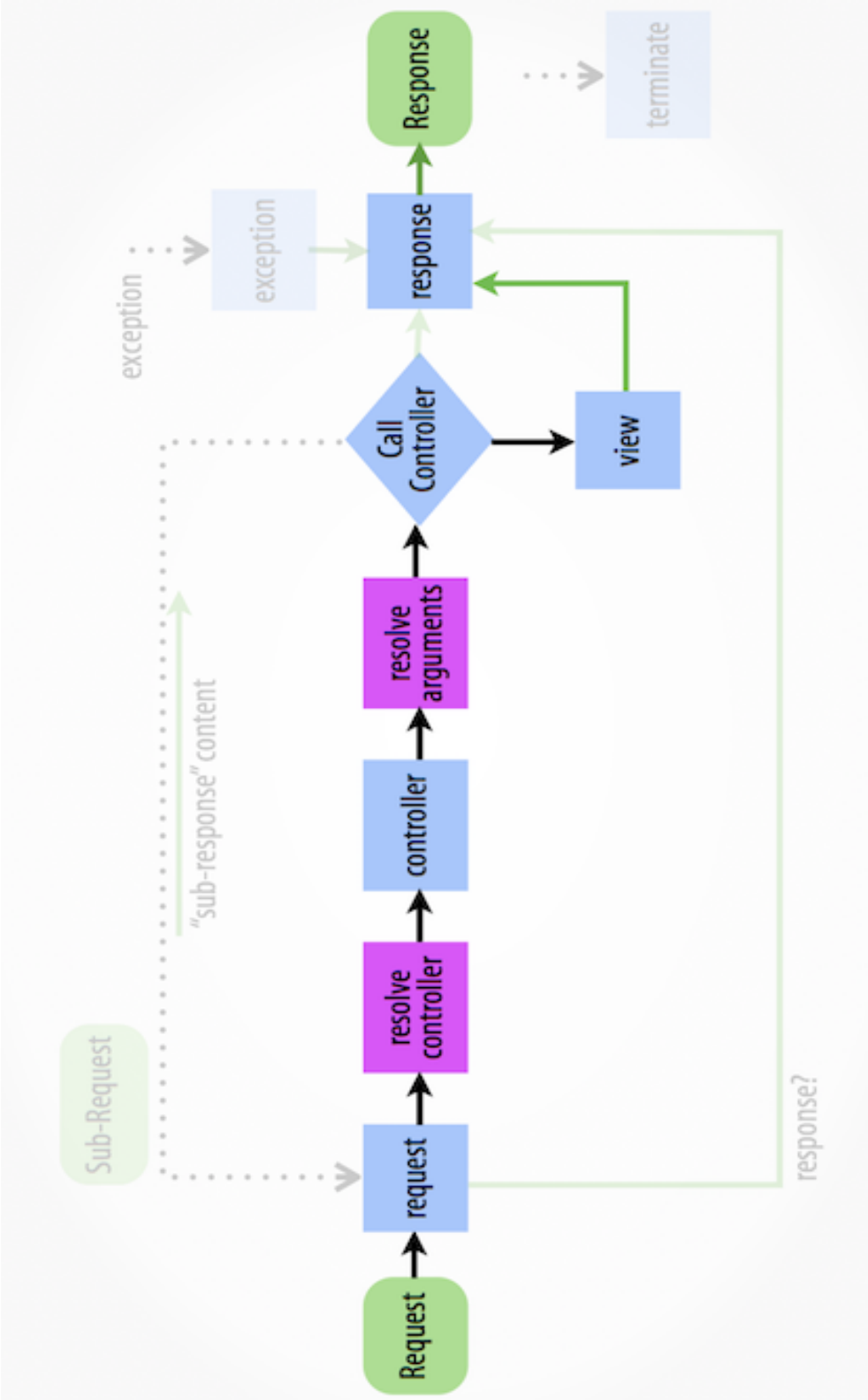
Call Controller Vykonání příslušné metody Controlleru a obsloužení požadavku uživatele. Pro obsluhu požadavku máme k dispozici vše uvedené v 5.3.

Veškerá potřebná funkcionalita a business logika aplikace pro obsloužení požadavku je přístupná pomocí služeb. To platí i pro repository. Dokážeme tedy provádět operace nad databází, vytvářet a validovat formuláře a mnohé další.

View Ze zpracovaných dat vygeneruje výstup v požadovaném formátu.

response Vytvoření objektu reprezentující odpověď pro uživatele.

Response Kompletní odpověď je odeslána zpět uživateli.



Obrázek 5.2: Od požadavku až k odpovědi [31]

Kapitola 6

Testování

6.1 Druhy využitých testů

Při testování aplikace jsem využil unit a akceptační testy.

6.1.1 Unit testy

Pro otestování funkcionality, která se špatně testuje pomocí akceptačních testů jsem zvolil unit testy. Pokrytí:

- Aktualizace stavu skladu po vytvoření příjemky.
- Aktualizace stavu skladu po vytvoření výdejky.
- Aktualizace stavu skladu po upravení příjemky.
- Aktualizace stavu skladu po upravení výdejky.
- Aktualizace stavu skladu po smazání příjemky.
- Aktualizace stavu skladu po smazání výdejky.
- Aktualizace stavu skladu po převedení stavu skladu A do skladu B.
- Výpočet splatnosti faktury dle data vystavení faktury a navýšené o splatnost faktur uvedené u odběratele.
- Výpočet ceny bez DPH po slevě pro každý záznam vydané knihy.

Testování pomocí unit testů proběhlo úspěšně.

6.1.2 Akceptační testy

Akceptační testy jsou popsány pomocí scénářů pro jejich provedení.

6.2 Funkční požadavky

6.2.1 Scénáře

Menu je umístěné na horní části stránky. Je však dostupné až po přihlášení uživatele. Všechny operace kromě registrace, přihlášení a vygenerování nového hesla vyžadují, aby byl uživatel přihlášen v aplikaci. Některé akce navíc vyžadují specifickou roli viz 4.3.4

6.2.1.1 Uživatel

Před započetím testování ostatní funkcionality je potřeba vytvořit uživatele pro systém.

Registrace nového uživatele Na stránce přihlášení do aplikace zvolte registraci. Vyplňte vyžadované údaje.

Přihlášení uživatele Na stránce přihlášení do aplikace vyplňte uživatelské jméno a heslo. Do systému se nelze přihlásit, dokud administrátor neaktivuje účet. Pokud administrátorský účet neexistuje, je potřeba v databázi nastavit účet aktivní manuálně.

Odhlášení uživatele V menu vyberte položku **Odhlásit**.

Upravení osobních údajů a hesla V menu klikněte na své uživatelské jméno (pravá část menu). Na zobrazené stránce zvolte **Upravit údaje** pro upravení osobních údajů kromě hesla. Pro upravení hesla zvolte **Změnit heslo**. Objeví se stránka s aktálními údaji. Údaje přepište dle libosti a změny uložte.

Vygenerování hesla po zadání uživatelského jména a emailu Na stránce přihlášení do systému vyberte obnovení hesla. Na zobrazené stránce zadejte uživ. jméno a platný email, který jste zadali při registraci. Nové heslo obdržíte pomocí emailu.

Smazání svého účtu V menu klikněte na své uživatelské jméno. Na zobrazené stránce zvolte možnost **Smazat účet**.

Aktivní sklad Pro provedení je vyžadována existence skladů.

- **Přepnutí pracovního skladu** V menu klikněte na **Aktivní sklad** a ze seznamu skladů vyberte požadovaný sklad k aktivaci.
- **Aplikace si pro uživatele bude pamatovat aktivní pracovní sklad mezi přihlášeními** Aktivujte uživateli sklad. Aktivní sklad je zobrazen na pravé straně menu. Uživatele odhlašte a přihlašte do aplikace a porovnejte hodnotu aktivních skladů.

6.2.1.2 Podobné scénáře pro více požadavků

Vytvoření záznamu V menu vyberte rozbalovací položku **Vytvořit** a z nabídky vyberte jednu z položek. Pro vytvoření položky je potřeba:

Kniha Vyplňit povinné pole formuláře.

Dodavatel/Odběratel Vyplňit povinné pole formuláře.

Příjemka Je nutné mít aktivovaný sklad a vytvořené záznamy knih. Pro příjemku s více jak jedním záznamem přijaté knihy využijte tlačítko **Přidat záznam**. Záznamy je možné opět smazat pomocí tlačítka **Odstranit** u záznamu.

Výdejka Je nutné mít aktivovaný sklad a dostatek knih na skladě. Pro výdejku s více jak jedním záznamem využijte tlačítko **Přidat záznam**. Záznamy je možné opět smazat pomocí tlačítka **Odstranit** u záznamu.

Sklad Vyplňit povinné pole formuláře.

Záznam uložíte pomocí tlačítka pod formulářem.

Upravení záznamu Předpokládáme existenci záznamu. V menu vyberte rozbalovací položku **Přehledy globální** a z nabídky vyberte jednu z položek. Následně klikněte na tlačítko **Upravit** u záznamu v zobrazeném seznamu.

Kniha Lze upravit všechny atributy pomocí formuláře.

Odběratel/Dodavatel Lze upravit všechny atributy pomocí formuláře.

Sklad Lze upravit všechny atributy kromě oceňovací metody.

Příjemka Pro upravení záznamu jsou vyžadovány vytvořené záznamy knih a knihy na skladě. Atributy příjemky lze změnit pomocí polí formuláře. Záznamy přijatých knih lze upravit přepsáním hodnot, odstraněním nebo přidáním záznamu.

Výdejka Pro upravení záznamu jsou vyžadovány vytvořené záznamy knih a knihy na skladě. Atributy výdejky lze změnit pomocí polí formuláře. Záznamy přijatých knih lze upravit přepsáním hodnot, odstraněním nebo přidáním záznamu.

Změny uložte tlačítkem pod formulářem.

Smazání záznamu Předpokládáme existenci záznamu. V menu vyberte rozbalovací položku **Přehledy globální** vyberte jednu z následujících položek:

- Příjemky

- Výdejky
- Knihy
- Odběratelé/Dodavatelé

Pro smazání záznamu pak využijte tlačítko **Smazat**, které je u každé položky v seznamu.

Zobrazení všech záznamů v systému s vyhledáváním dle parametru V menu vyberte rozbalovací položku **Přehledy globální** a z nabídky vyberte jednu z položek. Vyhledávání je možné pro:

- Příjemky
- Výdejky
- Skladované knihy
- Knihy
- Odběratelé/Dodavatelé
- Sklady

Vyhledávání se provede zapsáním hledaného řetězce do textového pole a stisknutím klávesy enter.

Vyhledání a zobrazení záznamů dle parametru pro aktivní sklad V menu vyberte rozbalovací položku **Přehledy pro sklad** a z nabídky vyberte jednu z následujících položek dle potřeby:

- Příjemky
- Výdejky
- Skladované knihy

Vyhledávání se provede zapsáním hledaného řetězce do textového pole a stisknutím klávesy enter.

Zobrazení detailu pro existující záznam V menu vyberte rozbalovací položku **Přehledy globální** a vyberte jednu z položek. Pro zobrazení detailu využijte tlačítka **Detail**. V případě Knihy se detail zobrazí kliknutím na text ve sloupci **Titul**. Všechny atributy položky jsou zobrazeny na výsledné stránce. Další informace jsou dostupné následujícím způsobem:

Příjemka Stránka obsahuje záznamy přijatých knih doplněné o výpočty.

Výdejka Stránka obsahuje záznamy vydaných knih doplněné o výpočty.

Skladovaná kniha Záznamy příjemek a výdejek jsou obsaženy v záložkách pod atributy.

Kniha Zobrazení záznamů příjemek, výdejek a skladovaných knih je vyřešeno pomocí záložek pod atributy.

Odběratelé/Dodavatelé Záznamy příjemek a výdejek jsou k nalezení jako záložky pod atributy.

Sklady Záznamy příjemek, výdejek, skladovaných knih jsou k nalezení jako záložky pod atributy.

6.2.2 Kniha

Pro následující scénáře je společný počátek. V menu rozklikněte rozbalovací položku **Přehledy globální** a vyberte **Knihy**. Tato stránka obsahuje všechny záznamy knih v systému.

Vytvoření vydání knihy k existujícímu záznamu Ze seznamu knih si jednu vyberte a klikněte na tlačítko **Nové vydání**. Ve formuláři upravte potřebné údaje a uložte.

Výpis všech vydání knihy pro zadanou knihu Kniha, která je vydáním jiné knihy, má podtržený text ve sloupci **Vydání**, po kliknutí na tento text se zobrazí všechny vydání původní knihy.

6.2.3 Dodavatel/Odběratel

Popis scénářů najdete zde 6.2.1.2.

6.2.4 Sklad

Popis scénářů najdete zde 6.2.1.2.

6.2.5 Skladovaná kniha

Zobrazení a vyhledávání záznamů podle zadaného parametru pro aktivní sklad Přechod na tuto stránku je možný dvěma způsoby. První je stiskem tlačítka **Index** a druhý **Přehledy pro sklad** a následně z výběru **Skladované knihy**. Zbytek scénářů najdete zde 6.2.1.2.

6.2.6 Příjemka na sklad

Popis scénářů požadavků najdete zde 6.2.1.2.

6.2.7 Výdejka ze skladu

Popis scénářů požadavků najdete zde 6.2.1.2.

6.2.8 Export

Soubor bude ke stažení nad formulářem.

Export příjemek ze zadaného časového intervalu. Formát PDF a CSV Vyberte v menu položku **Export dat** a v rozbalovacím menu **Příjemky od - do**. V zobrazeném formuláři je potřeba vyplňit počáteční a koncové datum výběru. Dále rozsah výběru exportu z možností:

- aktuální sklad
- všechny sklady

a formát exportu PDF nebo CSV.

Export výdejek ze zadaného časového intervalu. Formát PDF a CSV Vyberte v menu položku **Export dat** a v rozbalovacím menu **Výdejky od - do**. V zobrazeném formuláři je potřeba vyplňit počáteční a koncové datum. Dále rozsah výběru exportu z možností:

- aktuální sklad
- všechny sklady

a formát exportu PDF nebo CSV.

Export vystavené faktrury z výdejky do formátu PDF Vyberte v menu položku **Export dat** v rozbalovacím menu a zvolte **Vystavená faktura**. Ze seznamu dostupných výdejek si jednu zvolte a klikněte na **Faktura**. Ve formuláři vyplňte požadované údaje a nechte vygenerovat soubor pomocí tlačítka pod formulářem.

- **Možnost stanovení jiné slevy a DPH pro každou vydanou knihu** - U každého záznamu vydané knihy je textové pole pro stanovení nové hodnoty DPH
- **Výpočet splatnosti faktury dle data vystavení faktury a doby splatnosti faktur uvedené u odběratele** otestováno pomocí unit testů.
- **Výpočet ceny bez DPH po slevě pro každý záznam vydané knihy** otestováno pomocí unit testů.
- **Zobrazení celkové ceny a množství knih** otestováno pomocí unit testů.
- **Informace o odběrateli budou převzaty z výdejky** ověřitelné z vygenerovaného dokumentu.

Export výdejky do PDF Vyberte v menu položku **Export dat** v rozbalovacím menu zvolte **Výdejka**. Ze seznamu dostupných výdejek si jednu zvolte a klikněte na **Výdejka**. Do formuláře zadejte potřebné údaje a nechte vygenerovat výsledné PDF pomocí tlačítka pod formulářem.

6.2.9 Převod aktuálního stavu skladu do jiného skladu

Aplikace musí podporovat převod aktuálního stavu skladu A do skladu B. Převodem stavu se rozumí vytvoření příjemky pro sklad B z aktuálního stavu skladu A. Následuje přepočet stavu skladu B. Metoda oceňování zásob na skladech se nemusí shodovat.
Vytvořte 2 sklady a příjemky dle potřeby. V menu klikněte na **Admin sekce** a následně na **Sklady** a zvolte **Přenést aktuální sklad**. Ve formuláři vyberte zdrojový a cílový sklad. Přenesení příjemek lze zkontrolovat zobrazením příjemek pro cílový sklad. Aktualizace stavu skladu byla testována unit testy.

6.2.10 Admin

Pro uskutečnění následujících scénářů je potřeba přejít do **administrátorské sekce** aplikace. Do této sekce se dostanete pomocí stejnojmenné položky v menu.

- **Aktivace/blokace uživatele** V menu vyberte **Správa uživatelů** a v seznamů uživatelů klikněte na tlačítko **Aktivovat** nebo **Zablokovat**.
- **Smazání skladu** V menu zvolte položku **Sklady** a z rozbalovího menu **Smazání skladu**. Následně vyberte ze seznamu sklad ke smazání.
- **Nastavení firemních údajů** V menu zvolte **Nastavení firmy** a upravte potřebné údaje.
- **Změna rolí uživatelů** V menu vyberte **Správa uživatelů** a v seznamů uživatelů klikněte na tlačítko **Změna role** a vyberte pro uživatele novou roli.
- **Smazání uživatele** V menu vyberte **Správa uživatelů** a v seznamů uživatelů klikněte na tlačítko **Smazat**.

6.3 Nefunkční požadavky

Nefunkční požadavky N.1, N.2, N.3 splněny.

6.4 Další typy požadavků

Další typy požadavků D.1 a D.2 splněny.

Kapitola 7

Ekonomocko-manažerské zhodnocení

7.1 Náklady

Náklady spojené s vývojem aplikace jsem vyčíslil na základě pracnosti jednotlivých etap souvisejících s vývojem aplikace. Pracnost jednotlivých etap je uvedena v mhr (man hour). Na pokrytí nákladů spojených s vývojem aplikace uvažujme průměrnou sazbu 320 Kč/mhr. Sazba zahrnuje i povinné odvody zdravotního a sociálního pojištění. Jednotlivé etapy spolu s náklady na jejich provedení jsou k nalezení 7.1.

Tabulka 7.1: Náklady na vývoj aplikace

Etapa	Pracnost	Náklady
Nastudování problematiky	30 mhr	9600 Kč
Shromaždování a analýza požadavků	20 mhr	6400 Kč
Návrh	30 mhr	9600 Kč
Implementace	160 mhr	51200 Kč
Testování	30 mhr	9600 Kč
Nasazení aplikace	5 mhr	1600 Kč
Celkem	275 mhr	88000 Kč

Náklady pro koncové uživatele aplikace se skládají pouze z nákladů na provoz aplikace, jelikož aplikace je pojatá jako open source řešení dostupné zdarma. Náklady spojené s provozem aplikace se skládají z web. hostingu a pronájmu domény. Finanční částku potřebnou pro provoz aplikace na jeden rok jsem uvedl do tabulky 7.2.

Tabulka 7.2: Roční náklady na provoz aplikace

Služba	Trvání	Cena bez DPH	Cena včetně DPH
Hosting (PHP, MySQL, 1GB)	1 rok	588.00 Kč	708.00 Kč
Registrace domény (.cz)	1 rok	125.00 Kč	151.00 Kč
Celkem		713.00 Kč	859.00 Kč

7.2 Přínosy

Nakladatelsví využitím aplikace při vedení skladové evidence a provádění souvisejících procesů dosáhne následujících přínosů:

7.2.1 Úspora času

Nakladatelsví dosáhne úspory času při vykonávání procesů spojených se skladovou evidencí, jelikož funkcionalita systému je přímo vytvořená podle požadavků nakladatelsví.

7.2.2 Dostupnost

K využívání apliakce stačí jen webový prohlížeč a zařízení s připojením k internetu. Není potřeba instalovat žádný speciální SW na klientské stanice. Aplikace se tak stává velmi dobře dostupnou a je možné s ní pracovat prakticky z každého stolního PC s připojením k internetu. S aplikací může navíc v jeden okamžik pracovat více uživatelů.

7.2.3 Úspora finančních prostředků

Využíváním aplikace dosáhne nakladatelství úspory finančních prostředků díky:

- Vyšší efektivitě při vykonávání procesů souvisejících s vedením skladové evidence.
- Omezením zbytečných operací jako jsou přenosy dat mezi jednotlivými stanicemi a nebo programy.
- Není potřeba nakupovat SW pro vedení skladové evidence a tím dojde k úspoře finančních prostředků.
- Odpadá potřeba instalovat SW pro vedení skladové evidence na klientské stanice a případná potřeba změny IT infrastruktury firmy.

7.2.4 Rozdělení oprávnění

Aplikace umožňuje uživatelům přiřadit různá oprávnění pro práci s aplikací. Můžeme tak například povolit jen prohlížení záznamů a export dat pro některé uživatele.

7.2.5 Další přínosy

S využitím aplikace budou data uložena na jednom místě. Tím snižujeme šanci ztráty nebo případného smazání souboru s daty. Zálohování dat se stává snadnější.

Kapitola 8

Instalační příručka

8.1 Potřebný SW

Pro běh aplikace je potřeba:

Webový server Příkladem mohu uvést Apache HTTP Server, který je možné stáhnout z `http://httpd.apache.org/download.cgi`. Nejlépe verzi 2.0 a vyšší.

Jazyk PHP Ke stažení na adrese `http://php.net/downloads.php`. Aplikace vyžaduje verzi PHP 5.3 a vyšší.

DB server MySQL Ke stažení na adrese `http://www.mysql.com/downloads/`.

wkhtmltopdf Ke stažení na adrese `http://wkhtmltopdf.org/downloads.html`.

phpMyAdmin (volitelné) Slouží k administraci databázového serveru. Ke stažení na adrese `http://www.phpmyadmin.net/home_page/downloads.php`.

8.2 Postup

1. Nainstalujte a nakonfigurujte MySQL server dle pokynů výrobce.
2. Nainstalujte a nakonfigurujte web. server dle pokynů výrobce.
3. Nainstalujte a nakonfigurujte phpMyAdmin dle pokynů výrobce.
4. Pomocí phpMyAdmin se přihlašte do databáze a vytvořte uživatele pro databázi.
5. Překopírujte složku impl z CD do požadované lokace.
6. Pomocí následující sekvence příkazů stáhněte bundly třetí strany a připravte cache aplikace.

- php composer.phar self-update
- php composer.phar update
- php composer.phar install `--no-dev` `--optimize-autoloader`
- php app/console cache:clear `--env=prod` `--no-debug`

V případě neúspěchu následujte pokyny na adrese `http://symfony.com/doc/current/cookbook/deployment/tools.html`.

7. Pomocí příkazu **php app/check.php** zkontrolujte, že jsou splněny všechny potřebné podmínky pro běh aplikace.
8. V souboru app/config/parameters.yml nastavte hodnoty následujících položek pro správnou funkčnost databáze.
   - database_host:
   - database_port:
   - database_user:
   - database_password:
   - database_name:
9. Pro uvedení databáze do provozu použijte následující příkazy.

   php app/console doctrine:database:drop --force Pro odstranění DB.

   php app/console doctrine:database:create Pro vytvoření DB.

   php app/console doctrine:schema:update --force Pro vytvoření struktury DB.

   Následně databázi inicializujte pomocí souboru **init.sql**.
10. Pro správnou funkčnost zasílání emailů s novým heslem upravte v app/config/pamareters.yml
    - mailer_transport:
    - mailer_host:
    - mailer_user:
    - mailer_password:
    - mailer_sender: "emailová adresa pro odesílání hesel"

    Pokud hodláte využít jiný druh mailu než gmail, pak postupujte dle návodu na adrese `http://symfony.com/doc/current/reference/configuration/swiftmailer.html`.

11. Pro správnou funkčnost exportu do PDF postupujte dle uvedené konfigurace na adrese `https://github.com/KnpLabs/KnpSnappyBundle`. Do souboru app/config/config.yml je potřeba doplňit na konec souboru následující konfiguraci.

```
knp_snappy:
    pdf:
        enabled:    true
        binary:     "umisteni_souboru"
        options:    []
```

12. Pro přihlášení a aktivaci nových uživatelů systému můžete využít předpřipraveného účtu s adminskými právy. Uživatelské jméno: **admin** a heslo: **adminHeslo412487124**.

Závěr

Prvním cílem této práce bylo nastudovat problematiku skladové evidence z vyhlášek České republiky a dodržovat jejich ustanovení. Druhým cílem bylo provést zhodnocení a porovnání existujících řešení v této oblasti. A posledním cílem bylo analyzovat požadavky, navrhnout, implementovat, otestovat a provést zhodnocení ekonomicko-manažerských přínosů webové aplikace, která bude fungovat jako systém pro vedení skladové evidence v knižním nakladatelství.

Všechny stanovené cíle této práce jsou splněny. Před jejich splněním však bylo potřeba ujít dlouhou cestu. Nejdříve jsem nastudoval problematiku ohledně skladové evidence, abych zcela pochopil rozsah práce. Následně jsem shromáždil a analyzoval požadavky na požadovaný systém. Z nich jsem vytvořil finální požadavky na systém a rozdělil je do kategorií. Na základě analyzovaných požadavků jsem provedl zhodnocení a porovnání existujících řešení s požadavky zadavatele. Jelikož z různých důvodů existující řešení zcela nevyhovovala požadavkům nakladatelsví, započal jsem s vývojem nové webové aplikace. Po návrhu aplikace následoval výběr technologií, implmentace a neodmyslitelné testování dokončené aplikace. Na závěr jsem provedl zhodnocení dosažených ekonomicko-manažerských přínosů. A pro snadnější nasazení jsem přidal příručku pro nasazení aplikace na server.

Osobním přínosem z této práce je pro mě zdokonalení se v používání frameworků a dalších knihoven. Dále rozšíření znalostí o export do formátu PDF z HTML a hlubší pochopení problematiky skladové evidence.

Literatura

[1] Ing. Pavel Náplava: *ROZDĚLENÍ IS* [online] , 2. 11. 2012 [cit. 2015-03-29]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-TIS/_media/lectures/06/prednaska06.pdf

[2] Ing. David Buchtela, Ph.D.: *Předmět a cíle účetnictví* [online] , 14. 12. 2014 [cit. 2014-12-14]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-FIP/_media/lectures/bi_fip_p01.pdf

[3] *500/2002 Sb.VYHLÁŠKA ze dne 6. listopadu 2002, kterou se provádějí některá ustanovení zákona č. 563/1991 Sb., o účetnictví, ve znění pozdějších předpisů, pro účetní jednotky, které jsou podnikateli účtujícími v soustavě podvojného účetnictví* [online] , in: SBÍRKA ZÁKONŮ ročník 2002, částka 174, ze dne 5.12.2002 [cit. 2015-04-02]. Dostupné z: http://portal.gov.cz/app/zakony/download?idBiblio=54043&nr=500~2F2002~20Sb.&ft=pdf

[4] Hana Kovalíková: *Problematika zásob a skladového hospodářství - Daňový portál profesionálů a daňových poradců* [online] , [cit. 2015-03-25]. Dostupné z: http://www.danarionline.cz/archiv/dokument/doc-d192v168-problematika-zasob-a-skladoveho-hospodarstvi/

[5] Microsoft Corporation: *Office - Microsoft Store Česká republika* [online] , [cit. 2015-04-02]. Dostupné z: http://www.microsoftstore.com/store/mseea/cs_CZ/cat/Office/categoryID.66226700

[6] KASTNER software: *Stereo - ekonomický software | KASTNER software* [online] , [cit. 2015-04-04]. Dostupné z: http://www.kastnersw.cz/stereo/

[7] Microsoft Corporation: *Windows - Microsoft Windows* [online] , [cit. 2015-03-25]. Dostupné z: http://windows.microsoft.com/en-us/windows/home

[8] Ježek software: *DUEL Modul Sklady | Ježek software* [online] , [cit. 2015-04-10]. Dostupné z: http://www.jezeksw.cz/duel/popis-programu/otazky/sklady

[9] SHERWOOD Media s.r.o.: *Média, nakladatelství a vydavatelství, marketing :: HELIOS - podnikový informační systém, ekonomický a účetní software, systém pro veřejnou správu* [online] , [cit. 2015-04-11]. Dostupné z: http://www.helios.eu/oborova-reseni/sektor-sluzeb/media-nakladatelstvi-vydavatelstvi-marketing/

[10] Tomáš Krátký, Bohumír Zoubek: *Requirements Engineering* [online]. Profinit, s.r.o. , [cit. 2015-04-11]. Dostupné z: http://www.profinit.eu/fileadmin/Content/profinit.eu/Academy/NSWI129/prednasky/02_Requirements.pdf

[11] Bruce Timberlake: *LAMP (Linux, Apache, MySQL, PHP)* [online] , [cit. 2015-04-12]. Dostupné z: http://lamphowto.com/

[12] Tomáš Krátký, Bohumír Zoubek: *Softwarový proces* [online] . Dostupné z: http://www.profinit.eu/fileadmin/Content/profinit.eu/Academy/NSWI129/prednasky/12_SoftwareProcess.pdf

[13] Paul A. Hoadley: *Vodopadovy model.png - Wikimedia Commons* [online] , [cit. 2015-04-16]. Dostupné z: http://commons.wikimedia.org/wiki/File:Vodopadovy_model.png

[14] Boodhoo, J.-P.: Design Patterns: Model View Presenter. [cit. 2015-04-14]. Dostupné z: https://msdn.microsoft.com/en-us/magazine/cc188690.aspx

[15] Jiří Mlejnek: *Integrace aplikací* [online] , 29. 11. 2013 [cit. 2015-04-14]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-SI1/_media/lectures/10/10.prednaska.pdf

[16] Thomas Guerin: *How to build rock-solid apps and keep 100M+ users happy* [online] , 26. 11. 2014 [cit. 2015-04-18]. Dostupné z: http://blog.xebia.fr/2014/11/26/devoxx-how-to-build-rock-solid-apps-and-keep-100m-users-happy/

[17] phpMyAdmin contributors: *phpMyAdmin* [online] , [cit. 2015-04-22]. Dostupné z: http://www.phpmyadmin.net/home_page/index.php

[18] The Linux Foundation: *Linux.com | The source for Linux information* [online] , [cit. 2015-04-17]. Dostupné z: www.linux.com

[19] The Apache Software Foundation: *The Apache HTTP Server Project* [online] , [cit. 2015-04-17]. Dostupné z: http://httpd.apache.org/

[20] The Apache Software Foundation: *MySQL :: The world's most popular open source database* [online] , [cit. 2015-04-17]. Dostupné z: https://www.mysql.com/

[21] Nette Foundation: *Nette Framework* [online] , [cit. 2015-04-07]. Dostupné z: http://nette.org/en/

[22] Fabien Potencier: *Symfony, High Performance PHP Framework for Web Development* [online] , [cit. 2015-04-07]. Dostupné z: http://symfony.com/

[23] Zend Technologies Ltd.: *Zend Framework* [online] , [cit. 2015-04-07]. Dostupné z: http://framework.zend.com/

[24] Doctrine Team: *Doctrine Project* [online] , [cit. 2015-04-09]. Dostupné z: www.doctrine-project.org

[25] Sensio Labs: *Twig - The flexible, fast, and secure PHP template engine* [online] , [cit. 2015-04-22]. Dostupné z: http://twig.sensiolabs.org/

[26] Jiří Koutný: *Symfony2, těší mě! - Zdroják* [online] , 26. 6. 2013 [cit. 2015-04-17]. Dostupné z: http://www.zdrojak.cz/clanky/symfony2-tesi-me/

[27] Matthieu Bontemps, Antoine Hérault: *KnpLabs/KnpSnappyBundle* [online] , [cit. 2015-04-12]. Dostupné z: https://github.com/KnpLabs/KnpSnappyBundle

[28] Jakob Truelsen, Ashish Kulkarni: *wkhtmltopdf* [online] , [cit. 2015-04-12]. Dostupné z: http://wkhtmltopdf.org/

[29] Mark Otto, Jacob Thornton: *Bootstrap* [online] , [cit. 2015-03-25]. Dostupné z: http://getbootstrap.com/

[30] David Torres: *URI Euler Diagram no lone URIs.svg - Wikimedia Commons* [online] , 22. 6. 2010, [cit. 2015-04-19]. Dostupné z: http://commons.wikimedia.org/wiki/File:URI_Euler_Diagram_no_lone_URIs.svg

[31] Fabien Potencier: *The HttpKernel Component (The Symfony Components)* [online] , [cit. 2015-04-15]. Dostupné z: http://symfony.com/doc/current/components/http_kernel/introduction.html

Příloha A

Seznam použitých zkratek

AJAX Asynchronous JavaScript and XML

CRUD Create Retrieve Update Delete

CSS Cascading Style Sheet

CSV Comma Separated values

DB Databáze

HTML HyperText Markup Language

HW HardWare

FIFO First In First Out

IS Informační Systém

LAMP Linux, Apache HTTP Server, MySQL, PHP

MHR man hour

MVP Model-View-Presenter

ORM Object-Relational Mapping

SW SoftWare

URI Uniform Resource Identifier

URL Uniform Resource Locator

URN Uniform Resource Name

WYSIWYG What You See Is What You Get

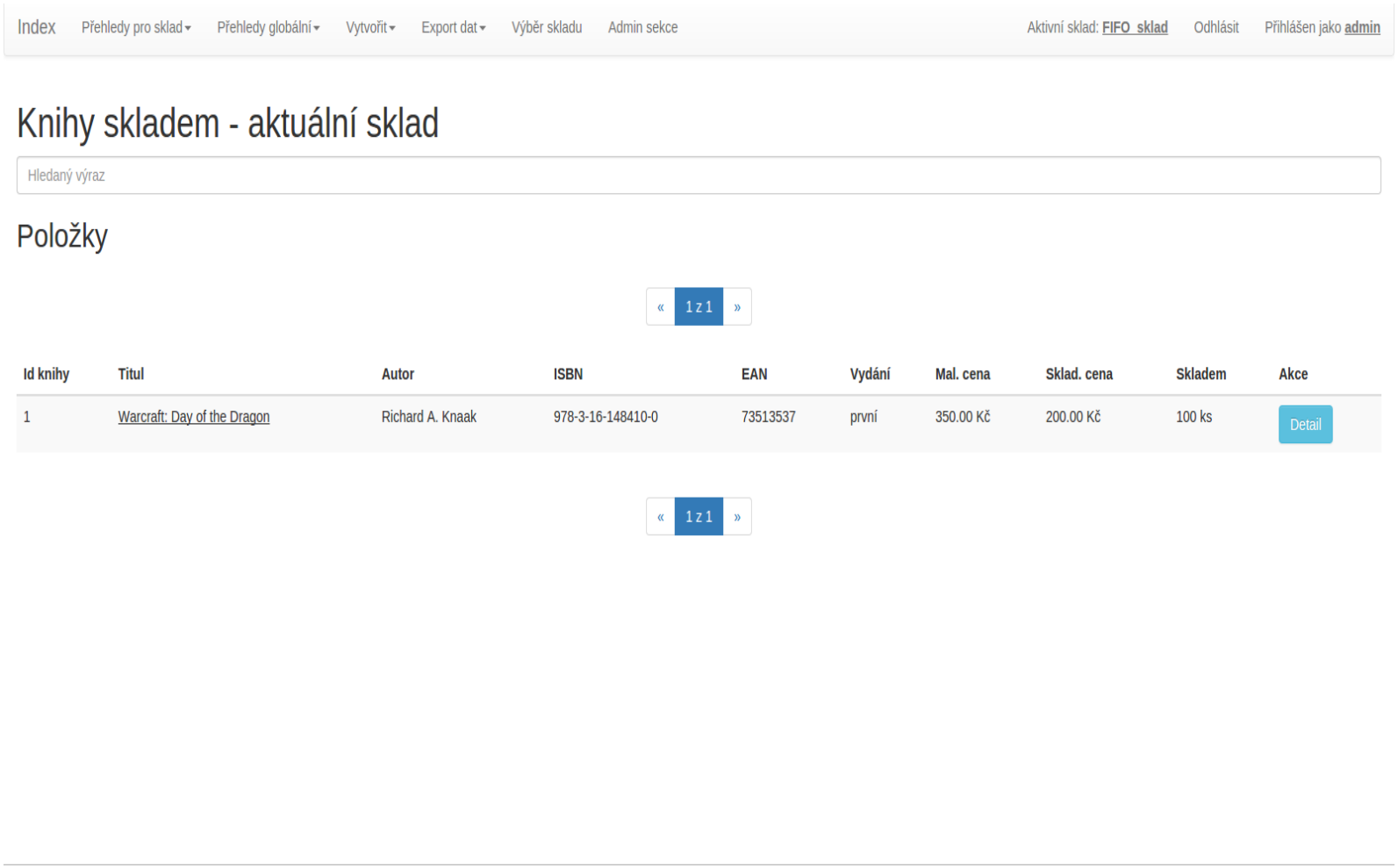
Příloha B

Obsah přiloženého CD

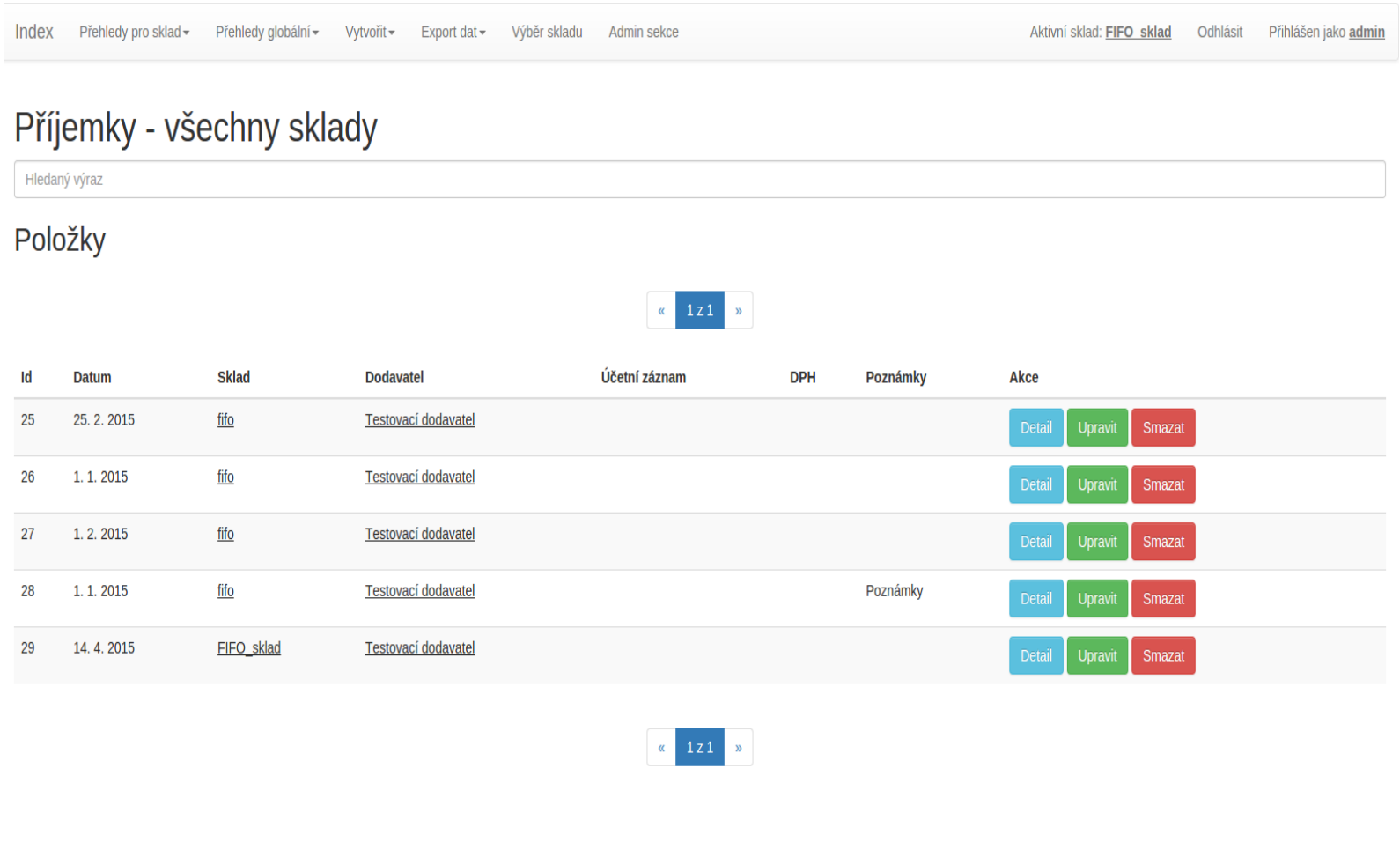
- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
    - `images` ..... použité obrázky
  - `impl` ..... zdrojové kódy implementace
    - `app`
    - `bin` ..... binární soubory
    - `src` ..... zdrojové kódy aplikace
    - `vendor` ..... bundly třetí strany
    - `web`
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF

Příloha C

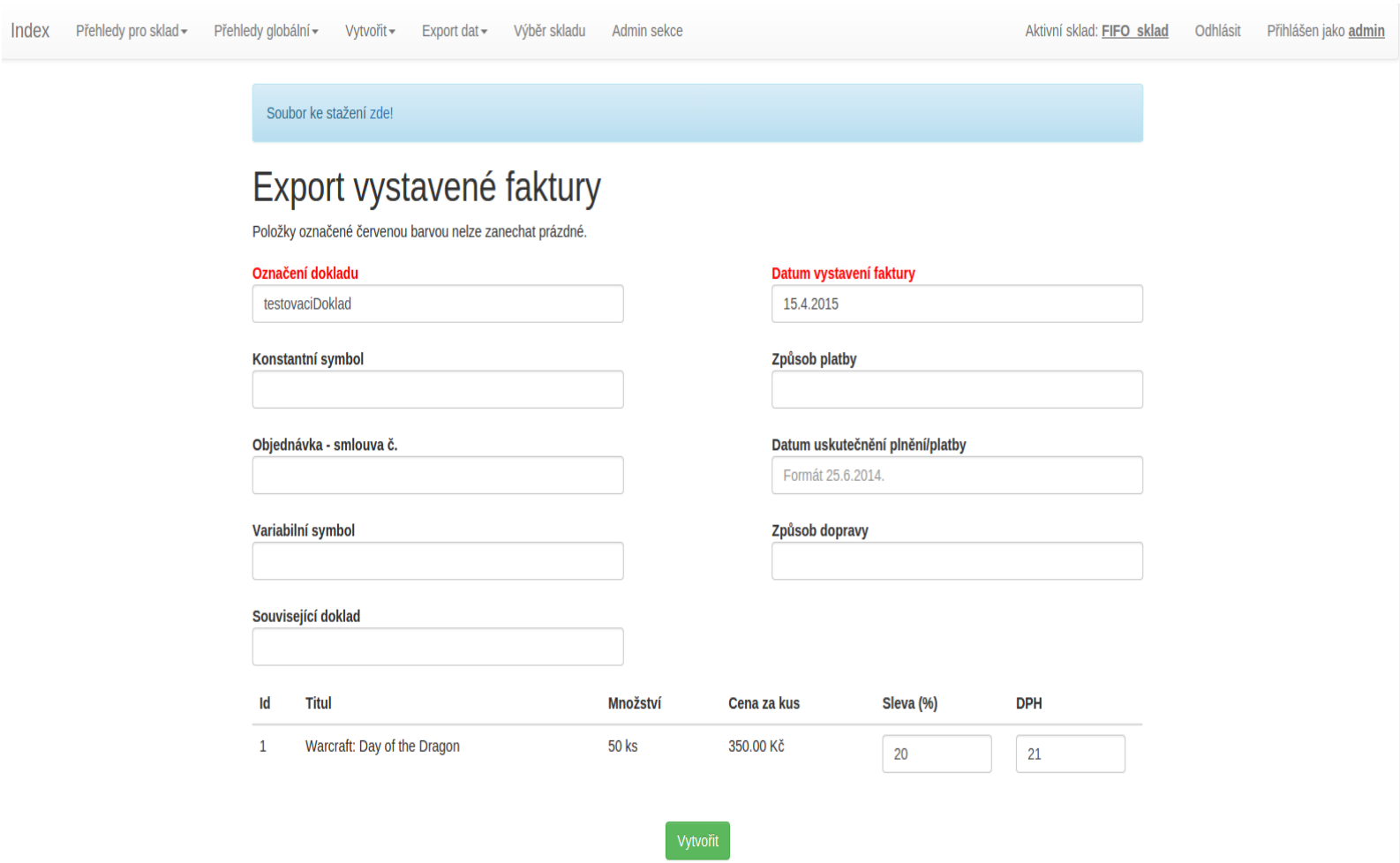
Screenshoty aplikace



Obrázek C.1: Hlavní stránka aplikace



Obrázek C.2: Přehled všech výdejek v aplikaci



Obrázek C.3: Export faktury

Faktura - daňový doklad č. testovaciDoklad

Dodavatel Testovací s.r.o.
Testovací 384
165 00, Praha 7

Telefon:
WWW:
Email:
IČO: 00029947 **DIČ:** CZ00029947
Ban. Účet: **Spec. symb.:**
IBAN: **Fax:**

Zapsán u: Městského soudu v Praze
Dne: 14. 4. 2012 **Zn.** C541247
Dodavatel je plátce DPH

Konstantní symbol:
Číslo dokladu: testovaciDoklad
Vystavil admin
Objednávka - smlouva č.:

Variabilní symbol:
Související doklad:

Odběratel **IČO:** 12345678 **DIČ:** CZ12345678

Testovací dodavatel
Nad Manovkou 327/31
161 00, Praha 6

Příjemce Testovací dodavatel

Nad Manovkou 327/31
161 00, Praha 6

Datum splatnosti: 25. 5. 2015
Způsob úhrady:
Vystaveno: 15. 4. 2015
Datum uskutečnění plnění/platby:
Způsob dopravy

Titul (ceny jsou uvedeny bez DPH)	Množství	Sleva %	Jedn. cena	Cena celkem	DPH
Warcraft: Day of the Dragon	50 ks	20	231.40	11 570.25	21 %

sazba DPH	bez daně celkem	daň celkem	s daní
21 %	11 570.25	2 429.75	14 000.00
cena celkem	**11 570.25**	**2 429.75**	**14 000.00**

Celková částka: **14 000.00 Kč**

Obrázek C.4: Vyexportovaná vystavená faktura