

Informační systém biskupství

Bc. Marek Soldán



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2015/2016

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Marek Soldán**

Osobní číslo: **A12480**

Studijní program: **N3902 Inženýrská informatika**

Studijní obor: **Informační technologie**

Forma studia: **kombinovaná**

Téma práce: **Informační systém biskupství**

Téma anglicky: **An Information System for a Bishopric**

Zásady pro vypracování:

1. Provedte vyhodnocení současného stavu informačního systému.
2. Vypracujte analýzu požadavků pomocí metod inženýrství požadavků.
3. Navrhněte model systému a datový model.
4. Analyzujte potřebu uživatelského rozhraní a navrhněte jej.
5. Provedte návrh architektury systému.
6. Realizujte navrhovaný systém pomocí prototypové implementace.
7. Vypracujte návrh možného dalšího rozvoje systému.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. VRANA, Ivan a Ila NEUSTADT. Projektování informačních systémů s UML: objektově orientovaná analýza a návrh prakticky. Vyd. 1. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2007, 147 s. ISBN 978-80-213-1817-5.
2. ARLOW, Jim a Ila NEUSTADT. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. Brno: Computer Press, 2007. ISBN 978-802-5115-039.
3. PECINOVSKÝ, Rudolf a Ila NEUSTADT. OOP – learn object oriented thinking and programming: objektově orientovaná analýza a návrh prakticky. Vyd. 1. Řepín: Tomáš Bruckner, 2013, 147 s. Academic series. ISBN 978-80-904661-8-0.
4. LAVIN, Peter a Ila NEUSTADT. OOP – learn object oriented thinking and programming: koncepty, techniky a kód. 1. vyd. Praha: Grada, 2009, 211 s. Průvodce (Grada). ISBN 978-802-4721-378.
5. ROBINSON, Simon a Ila NEUSTADT. CSharp: programujeme profesionálně. Vyd. 1. Brno: Computer Press, 2003, xxx, 1130 s. Programmer to programmer. ISBN 80-251-0085-5.
6. CHLAPEK, Dušan, Václav ŘEPA a Iva STANOVSKÁ. Analýza a návrh informačních systémů. 1. vyd. Praha: Oeconomica, 2011. ISBN 978-80-245-1782-7.
7. VLASÁK, Rudolf a Soňa BULÍČKOVÁ. Základy projektování informačních systémů. 1. vyd. Praha: Karolinum, 2003. ISBN 80-246-0727-1.

Vedoucí diplomové práce:

Ing. Radek Šilhavý, Ph.D.

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

5. února 2016

Termín odevzdání diplomové práce:

20. května 2016

Ve Zlíně dne 5. února 2016

doc. Mgr. Milan Adámek, Ph.D.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

Jméno, příjmení: Marek Soldán

Název diplomové práce: Informační systém biskupství

Prohlašuji, že


- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

19. 5. 2016


podpis diplomanta

ABSTRAKT

Tato diplomová práce se věnuje informačním systémům a jejich využití v prostředí biskupství a diecéze. Důvodem pro realizaci jednotného informačního systému byla aktuální potřeba Arcibiskupství olomouckého digitalizace některých procesů. Cílem práce je analýza jednotlivých procesů, zpracování požadavků, vytvoření modelu systému a jeho prototypová implementace. Hlavním očekávaným přínosem je zavedení jednotného systému pro zaměstnance arcibiskupství a větší efektivita pravidelně prováděných úkonů.

Klíčová slova: informační systém, biskupství, diecéze, model, OOP, UML, PHP, MySQL, Nette framework

ABSTRACT

The diploma thesis deals with information systems and their use in bishopric and dioceses. The actual need Archbishopric in Olomouc of digitalization of certain processes was the reason for the realization of the thesis. The aim is to analyze individual process, process requests and create a model of system and implement its prototype. The main expected benefit of the thesis is the implementation of an integrated system for employees of archbishopric which leads to greater efficiency regularly performed operations.

Keywords: information system, bishopric, diocese, model, OOP, UML, PHP, MySQL, Nette framework

Chtěl bych na tomto místě poděkovat všem, kteří mě jakkoli podpořili. Děkuji vedoucímu mé diplomové práce Ing. Radku Šilhavému, Ph.D. za jeho rady a připomínky. Děkuji kolegům v práci za pomoc a náměty, kterými mě inspirovali. Děkuji své manželce za podporu po celou dobu studia a zejména v posledních týdnech při dokončování diplomové práce.

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST.....	10
1 CÍLE A MOTIVACE PRÁCE	11
2 SEZNÁMENÍ S PROBLÉMOVOU DOMÉNOU	12
2.1 VYMEZENÍ POJMŮ.....	12
2.2 STRUKTURA DIECÉZE	13
2.2.1 Diecézní zaměstnanci.....	13
2.2.2 Kuriální zaměstnanci.....	13
2.3 INFORMAČNÍ SYSTÉMY PROVOZOVANÉ NA ARCIBISKUPSTVÍ	14
2.3.1 Systém pro mzdy a personalistiku.....	14
2.3.2 Ekonomický software.....	14
2.3.3 Systém pro evidenci diecéze, spisová služba.....	15
2.3.4 Docházkový systém	15
2.3.5 Zhodnocení současného stavu.....	15
3 INFORMAČNÍ SYSTÉM.....	16
3.1 DEFINICE IS.....	16
3.2 ARCHITEKTURA IS PODLE ÚROVNĚ ŘÍZENÍ	17
3.3 TECHNOLOGICKÁ ARCHITEKTURA IS	19
3.4 PROJEKTOVÁNÍ INFORMAČNÍCH SYSTÉMŮ	20
3.4.1 Metodiky vývoje softwaru	21
4 PROSTŘEDKY PRO ŘEŠENÍ PROBLEMATIKY.....	23
4.1 UML.....	23
4.2 HTML A CSS.....	24
4.3 PHP.....	24
4.4 MYSQL	24
4.5 NETTE FRAMEWORK	25
4.6 BOOTSTRAP.....	26
4.7 PROGRAMOVÉ VYBAVENÍ.....	26
II PRAKTICKÁ ČÁST	27
5 POŽADAVKY NA NOVÝ INFORMAČNÍ SYSTÉM	28
5.1 EKONOMICKÉ ODDĚLENÍ	28
5.2 MZDOVÉ ODDĚLENÍ.....	29
5.3 EVIDENCE ARCIDIECÉZE	29
5.4 STAVEBNÍ ODDĚLENÍ.....	29
5.5 PŘEHLED POŽADAVKŮ.....	30
5.6 DALŠÍ POŽADAVKY NA SYSTÉM.....	32
6 ZHODNOCENÍ SOUČASNÉHO STAVU A NÁVRH ŘEŠENÍ.....	34

6.1	ZHODNOCENÍ SOUČASNÉHO STAVU	34
6.2	ROZŠÍŘENÍ NĚKTERÉHO ZE STÁVAJÍCÍCH SYSTÉMŮ.....	34
6.3	NÁVRH VLASTNÍHO SYSTÉMU.....	35
7	NÁVRH MODELU SYSTÉMU	36
7.1	AKTÉŘI SYSTÉMU	36
7.2	SPECIFIKACE PŘÍPADŮ UŽITÍ	39
7.3	MODEL TŘÍD.....	43
8	NÁVRH DATOVÉHO MODELU	44
8.1	MODUL INVENTUR.....	45
8.2	MODUL FONDŮ	46
8.3	MODUL VÝKAZŮ PRÁCE	47
8.4	MODUL HLÁŠENÍ O ÚKONECH V DUCHOVNÍ SPRÁVĚ	48
8.5	MODUL HLÁŠENÍ OPRAV	50
9	NÁVRH UŽIVATELSKÉHO ROZHRAŇÍ	52
9.1	UŽIVATELSKÉ ROZHRAŇÍ	52
9.2	WIREFRAME	53
9.3	NÁVRH UŽIVATELSKÉHO ROZHRAŇÍ.....	54
10	IMPLEMENTACE PROTOTYPU	56
10.1	NETTE FRAMEWORK.....	56
10.2	ADRESÁŘOVÁ STRUKTURA APLIKACE	57
10.3	FORMULÁŘE.....	58
10.4	MODEL	59
10.5	PRESENTERS	60
10.6	ROUTER.....	62
10.7	LATTE	63
11	ZHODNOCENÍ ŘEŠENÍ A NÁVRH MOŽNÉHO ROZŠÍŘENÍ.....	65
11.1	DALŠÍ MOŽNOSTI ROZŠÍŘENÍ SYSTÉMU	65
	ZÁVĚR	67
	SEZNAM POUŽITÉ LITERATURY.....	68
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	70
	SEZNAM OBRÁZKŮ	72
	SEZNAM TABULEK.....	73
	SEZNAM PŘÍLOH.....	74

ÚVOD

Tato diplomová práce pojednává o možnosti využití informačních systémů v prostředí biskupství. Motivací k této práci byla žádost zadavatele o vytvoření informačního systému pro pracovníky biskupství. V první fázi nebylo přesně definováno, které agendy by měl systém pokrývat. Součástí práce je právě specifikace procesů na jednotlivých odděleních a návrh systému tak, aby co nejlépe odpovídal potřebám jednotlivých pracovníků.

V teoretické části práce je nejdříve věnována pozornost samotnému pojmu informační systém a nastínění některých oblastí z tvorby informačních systémů. Dále je vymezena problémová doména a jsou popsána některá specifika církevního prostředí. Na závěr teoretické části jsou shrnuty nástroje, které byly při tvorbě systému využity.

Praktická část práce potom v jednotlivých krocích shrnuje samotný návrh systému. Po detailnějším seznámení s procesy na konkrétních odděleních byl vyhotoven soupis požadavků, které by informační systém měl pokrýt. Požadavky byly přesněji specifikovány v rámci případů užití a byly také vymezeny jednotlivé role. Značná pozornost byla věnována návrhu datového modelu. Ten musí umožňovat výměnu dat s některými stávajícími systémy. Návrh uživatelského rozhraní byl s pracovníky oddělení konzultován, aby rozhraní co nejlépe vyhovovalo lidem, kteří s ním budou pracovat. Navržený systém byl implementován jako prototyp a v současné době probíhá testovací provoz.

Autor diplomové práce je již několik let zaměstnancem IT oddělení na Arcibiskupství olomouckém. Je s danou problematikou do značné míry obeznámen, a proto se také rozhodl znalosti získané studiem využít a navrhnout informační systém přesně pro potřeby arcibiskupství.

I. TEORETICKÁ ČÁST

1 CÍLE A MOTIVACE PRÁCE

Primární motivací této práce je aktuální potřeba Arcibiskupství olomouckého zavést informační systém. Bylo by samozřejmě ideální, kdyby se podařilo pokrýt všechny požadavky jedním systémem. To ale není reálně možné. Proto jsou na arcibiskupství postupně zaváděny jednotlivé informační systémy pro konkrétní účely (ekonomický, stavební aj.). Církevní prostředí je natolik specifické, že standardně nabízené systémy nepokrývají kompletní problematiku všech oddělení.

Cílem práce je také zefektivnění práce na některých odděleních. Procesy, které na biskupství běží již několik stovek let, a data, která jsou zpracovávána, jsou řešena pouze na papíře bez podpory informačních technologií. Až dodatečně se potom digitalizují a je snaha s nimi dále pracovat. V některých případech dokonce dochází k absurdním situacím, kdy odpovědný pracovník pořídí data v elektronické podobě, vytiskne je a v papírové podobě zašle na příslušné oddělení ke zpracování. Tam jsou data následně opisována do počítače. Snahou této práce je podobným situacím zamezit, práci s daty optimalizovat a pokusit se (i díky kvalitnější zpracovanému datovému modelu) nabídnout analytický pohled na zpracovávaná data.

Z tohoto důvodu, a z pohledu pracovníka oddělení informačních technologií Arcibiskupství olomouckého, se autor diplomové práce rozhodl využít znalosti získané studiem na vysoké škole. Zrealizoval projekt zavedení informačního systému, který by mohl současné systémy, které jsou na arcibiskupství v Olomouci provozovány, doplnit. Účelem práce je tedy vytvořit přehledný a uživatelsky přívětivý systém, který pomůže zaměstnancům lépe zvládnout jejich každodenní činnost.

Výsledkem této práce je analýza současného stavu, návrh datového modelu, uživatelského rozhraní a prototypová implementace systému. Protože je diplomová práce zároveň zakončením studijního programu Informační technologie, návrh a realizaci takového systému by měl autor jako absolvent vysoké školy zvládnout.

2 SEZNÁMENÍ S PROBLÉMOVOU DOMÉNOU

Informační systém bude navrhován a realizován na Arcibiskupství olomouckém. Tato organizace funguje již téměř tisíc let. Za tak dlouhou dobu se ustálila řada procesů zpracování dat. Tyto postupy jsou osvědčené, funkční a v některých případech vycházejí z církevního práva či jsou přímo dané předpisy nadřazených subjektů. Cílem této práce tedy nebude definování a zavádění nových procesů, ale identifikace stávajících procesů a jejich maximální podpora s využitím informačního systému.

2.1 Vymezení pojmů

Protože se v diplomové práci objevuje řada pojmů z církevního prostředí, které nemusí být všeobecně známy, nebo mohou být někdy zaměňovány, jsou zde pro úplnost vyjmenovány a definovány.

(Arci)Diecéze – územně-správní jednotka, v jejímž čele stojí (arci)biskup. Pro potřeby této práce není potřeba řešit rozdíly mezi diecézí a arcidiecézí. Diecéze je obecný pojem, který pokrývá i všechny vlastnosti arcidiecéze. Odlišnosti mezi těmito subjekty mají význam pouze z pohledu církevního práva.

(Arci)Biskupství – úřad a sídlo (arci)biskupa. Biskup stojí v čele diecéze a je jejím statutárním zástupcem.

Diecézní kurie –,Diecézní kurie sestává z těch zařízení a osob, které pomáhají biskupovi v řízení celé diecéze, hlavně ve vedení pastorační činnosti, ve výkonu správy diecéze a ve výkonu soudní moci“ [14]. Někdy bývá označována jako konzistoř, nebo jen zkráceně kurie.

Děkanát (vikariát) – je územní jednotka, menší než diecéze, sestávající se z několika farností. V čele stojí děkan (vikář). Jednotlivé farnosti jsou děkanátu správně podřízeny. Děkanát není samostatnou právnickou osobou. Na jednotlivých děkanátech také pracují laičtí pracovníci. Ti pomáhají děkanovi a správcům farností s administrativou.

Farnost – je základní církevní správní jednotka. Spravuje ji farář nebo administrátor. Velká většina farností je vymezena územně. Může zahrnovat území města, části města, vesnice či několika vesnic. Farnost spravuje kostely na svém území. Farnost má samostatnou právní subjektivitu.

2.2 Struktura diecéze

V Olomoucké arcidiecézi, ve které bude systém zaváděn, je více než 420 farností a přes 20 děkanátů.

Zaměstnanci biskupství se z našeho pohledu dělí do dvou skupin. V první skupině jsou zaměstnanci, kteří pracují přímo na jednotlivých děkanátech a ve farnostech. Ti se nacházejí mimo budovu kurie a pracují po celém území arcidiecéze. Jedná se v součtu o více než 500 zaměstnanců. Druhou skupinu tvoří zaměstnanci úřadu (kurie), kteří čítají něco málo přes sto lidí. Rozumí se tím zaměstnanci, kteří sídlí v jedné budově a přímo spolupracují s biskupem. Ti mají na starosti koordinaci činností a správu diecéze.

2.2.1 Diecézní zaměstnanci

Podstatná část zaměstnanců pracuje mimo arcibiskupství v Olomouci. Jsou to kněží, pastorační pracovníci, účetní, techničtí administrátoři. Tito lidé pracují samostatně, mají na starost pastorační ve farnostech, účetnictví farností, správu budov apod. Stále jsou ale podřízeni a metodicky vedeni zaměstnanci arcibiskupského úřadu.

Kněží, kteří jsou statutárními zástupci jednotlivých farností (právnických osob), mají vzhledem k farnosti funkci administrátora či faráře. Jsou proto oprávněni za farnost vykonávat veškeré související činnosti. Ostatní pracovníci mohou být vykonáním některých činností dočasně pověřeni.

Z pohledu používání informačních systémů se bohužel jednotlivé děkanáty a farnosti různí. Objevují se ale snahy o sjednocení a případný centrální provoz některých systémů. V současné době probíhá například centralizace účetního softwaru.

2.2.2 Kuriální zaměstnanci

Druhou skupinou jsou zaměstnanci, kteří jsou přímo součástí diecézní kurie. Všichni tito pracovníci se nacházejí v podstatě v jedné budově, jsou připojeni do společné LAN a všichni jsou součástí AD¹. Situace se tady moc neliší od běžné firmy. Je zde odbor právní, ekonomický, správní, stavební atd. Tyto odbory samozřejmě neslouží jen úřadu, ale celé diecézi.

¹ Active Directory – konkrétní implementace LDAP na Windows.

Dále se zde nacházejí tzv. *pastorační centra (oddělení)*, která koordinují pastorační činnost v jednotlivých děkanátech.

Správu ICT má na starosti samostatné oddělení, které poskytuje zaměstnancům vybavení, podporu a servis. Toto oddělení má v popisu práce všechny činnosti související se správou sítě, softwarového i hardwarového vybavení a se zaváděním nových systémů.

2.3 Informační systémy provozované na arcibiskupství

V následujícím textu jsou zevrubně popsány systémy, které jsou v současnosti na biskupství provozovány.

2.3.1 Systém pro mzdy a personalistiku

Na osobním a mzdovém oddělení je již přes deset let využíván systém od společnosti Vema a.s. I při relativně velkém množství zaměstnanců a z toho vyplývající velikosti zpracovávané agendy, se tento software jeví jako robustní a spolehlivé řešení. Bohužel není software propojený s žádným dalším systémem, a nelze proto sdílet například data o zaměstnancích apod.

2.3.2 Ekonomický software

Pro účtování a evidenci drobného majetku je využíván účetní software Pohoda. Na tento software se přešlo v nedávné době. Kolem roku 2012 již přestával vyhovovat stávající program (provozovaný pouze jako neserverová aplikace). Nový program pro účetnictví si po odzkoušení několika variant zvolili sami pracovníci ekonomického odboru. Systém je provozován serverově, takže do aplikace mají nově přístup i zaměstnanci jiných oddělení, kteří sice nemusí mít právo zápisu, ale mohou nahlížet do agend, které jim přísluší. Pomocí vzdáleného přístupu se také připojují účetní z jednotlivých děkanátů a vedou zde účetnictví za příslušné farnosti.

Nad účetním softwarem existuje nadstavba, která využívá metod data miningu a umožňuje tak vedoucím pracovníkům mít přehled nejen nad každou farností jako samostatnou jednotkou, ale i nad cashflow celé organizace či jednotlivých děkanátů.

Je to první systém, který je na arcibiskupství využíván v takovémto rozsahu. Tento informační systém měl zpočátku problémy se zaváděním, zejména kvůli neochotě pracovníků přejít na centrální software (někteří museli změnit účetní program, sjednotila se účetní osnova atd.). Přesto je tento systém velmi cenným přínosem pro chod arcibiskupství.

2.3.3 Systém pro evidenci diecéze, spisová služba

Systém pro evidenci diecéze (Katalog) je na každém biskupství jedním z klíčových. Zde se uchovávají údaje o jednotlivých farnostech, děkanátech, jmenování, jmenovacích dekretch, kněžích a jejich funkcích a všech jejich změnách. Obsahuje vlastně celou historii diecéze. Tato evidence byla až do počátku devadesátých let minulého století vedena pouze v papírové podobě. S nástupem osobních počítačů se objevila snaha také tuto agendu digitalizovat. Každé biskupství tehdy řešilo problém svépomocí. Po dvaceti letech přestaly pochopitelně systémy jednotlivých biskupství splňovat požadavky, kterých na ně bylo kladeno čím dál více. Ve stávající podobě nebylo možné systémy již dále používat.

Proto se většina biskupství shodla na společném postupu řešení tohoto problému. Analýzu prováděla dohromady všechna česká a moravská biskupství za pomoci externích spolupracovníků, kteří mají s danou problematikou zkušenosti. Programování a údržbu nyní zajišťují konkrétní pracovníci. Spisovou službu by bylo sice možné zakoupit samostatně jako hotový produkt, ale tím by se značně zkomplikovalo její propojení s Katalogem, což je v prostředí biskupství nezbytné.

Oba systémy se v současné době ladí a doplňují se nové funkce.

2.3.4 Docházkový systém

Pro zaměstnance, kteří pracují na arcibiskupství, je k dispozici docházkový systém firmy Advanced Network Technology, s.r.o.. Ten umožňuje evidenci docházky na pracoviště, plánování absencí, sledování aktuální přítomnosti apod. Výstupy ze systému slouží hlavně mzdové účtárně pro získávání měsíčních přehledů pro účtování a vedoucím pracovníkům pro kontrolu docházky.

2.3.5 Zhodnocení současného stavu

V předchozích kapitolách byly popsány některé informační systémy, které se na arcibiskupství provozují. Nejedná se o velký ERP systém, ale o několik samostatných, úzce zaměřených, systémů. Zdaleka však nepokrývají veškerou problematiku, kterou jednotlivá oddělení řeší. Některé úkony jsou zbytečně prováděny v papírové podobě. V současné době také narůstá objem informací, které jsou vyměňovány mezi pracovníky kurie a externími pracovníky. Cílem této práce je doplnit provozované systémy tak, aby pokrývaly i některé další agendy.

3 INFORMAČNÍ SYSTÉM

Tato kapitola má za cíl vymezit samotný pojem informačního systému a pojmy, které s informačním systémem přímo souvisejí. Ve stručnosti bude popsána architektura informačního systému a metodiky, které slouží k návrhu systému.

3.1 Definice IS

Informační systém je v dnešní době velmi často užívaný termín. Může mít velmi rozdílné významy, proto musíme na začátku vymezit oblast, ve které se budeme pohybovat. V odborné literatuře se nacházejí například tyto definice.

„Informační systém lze definovat jako soubor lidí, metod a technických prostředků zajišťující sběr, přenos, uchování, zpracování a prezentaci dat, za účelem prezentace informací pro potřeby uživatelů činných v systémech řízení.“ [8]

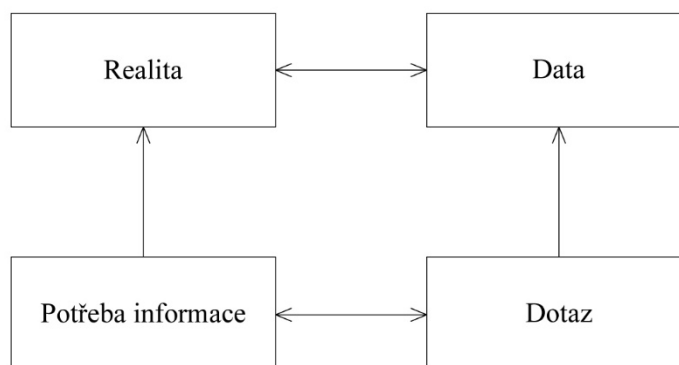
„Informační systém je obecně podpůrný systém pro systém řízení. Jestliže chceme projektovat systém řízení jako takový, musíme znát, jaké jsou cíle, a informační systém řešit tak, aby tyto cíle podporoval.“ [9]

Z výše uvedených definic je patrné, že přestože je pojem informační systém v dnešní době většinou spojován s výpočetní technikou, nemusí to tak být vždy. Podle této definice je nutné za informační systém pokládat i kuchařskou knihu či kartotéku.

Informační systém, kterým se bude zabývat tato práce, se týká systému využívaného v managementu organizace. Takový informační systém se může skládat z následujících součástí. Technické prostředky – tzv. hardware. Jedná se o počítačové vybavení, síťové prvky a veškeré periferie. Technologické prostředky – do této kategorie se řadí veškeré programové vybavení související s informačním systémem. Organizační prostředky – tzv. orgaware [7]. Pod tímto pojmem chápeme veškerá nařízení a předpisy, které se týkají informačního systému, například legislativu, vnitřní předpisy a směrnice, metodické pokyny apod. Lidská složka, někdy označovaná jako peopleware [7], zahrnuje osoby, které s daným systémem jakkoli interagují. Okolí systému představuje prostředí, do kterého je systém zasazen. Mohou sem patřit jiné systémy, se kterými komunikuje apod. [10].

Při návrhu je důležité soustředit se nejen na daný konkrétní software, ale zároveň uvažovat o budoucích uživateli a ostatních, výše zmíněných, prostředcích, jako samotné součásti systému.

Na závěr této kapitoly je zařazen informatický pohled. „*Informační systém je systém, umožňující účelné uspořádání sběru, uchování, zpracování a poskytování informací.*“ [13] IS je z tohoto pohledu kolekcí dat, která jsou získána pozorováním části reálného světa. Uživatel je může využívat v podstatě dvěma způsoby. Může do systému data zadávat nebo je ze systému získávat. Tento popis informačního systému je pro potřeby této práce nejvýstižnější.



Obr. 1. Informatický pohled na IS.

3.2 Architektura IS podle úrovně řízení

Informační systémy můžeme rozdělit podle použití v podnicích (Obr. 2). Je zřejmé, že jiné systémy se budou využívat při výrobě, jiné při prodeji, jiné při analýze, jiné při rozhodování. Každá úroveň řízení má své charakteristiky. Jednotlivé transakční systémy se mohou velmi lišit, podle konkrétních činností, které podnik vykonává. Mohou sem patřit pokladní systémy, terminály, čtečky kódu – tedy systémy, které jsou přímo odpovědny za pořizování dat. Často se u těchto systémů setkáváme s pojmem OLTP.² Jedná se o způsob uložení dat tak, aby byla co nejjednodušeji zabezpečena manipulace s těmito daty.

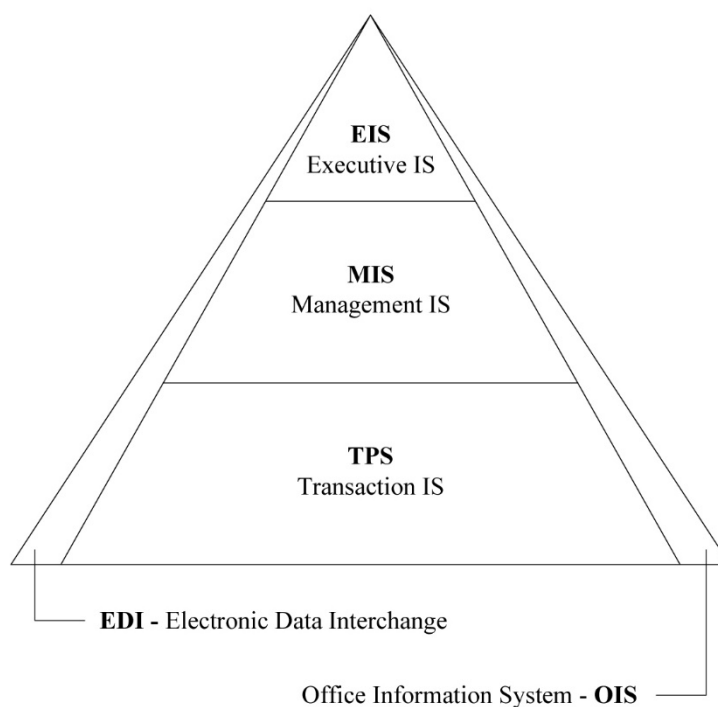
Informační systémy na úrovni managementu dále zpracovávají data, která byla pořízena na nižší úrovni. Slouží k monitorování a řízení jednotlivých podnikových procesů. Pracovníci na této úrovni mají možnost přistupovat už k předzpracovaným (agregovaným) datům, která

² Online Transaction Processing

jim umožňují provádět pokročilé analýzy. Pro uložení agregovaných a často také dopočítávaných dat se využívá technologie OLAP.³ Na této úrovni se stále pracuje s daty z vnitřního prostředí.

Třetí skupinou jsou systémy využívané vrcholovým managementem. Zásadně se od předchozí skupiny liší tím, že využívají i data okolního prostředí a agregace dat je již na velmi vysoké úrovni. Takovéto systémy slouží jako podpora strategického řízení a mohou pomoci například ke zvýšení konkurenceschopnosti [13].

Pro úplnost ještě zbývá krátce zmínit systémy, které prochází celou pyramidou. OIS⁴ je soubor programů, které umožňují digitalizaci kancelářské práce (textové a tabulkové editory, elektronická pošta...) a EDI⁵ definuje standardy, podle kterých je možné vyměňovat zprávy mezi aplikacemi.



Obr. 2. Rozdělení informačních systémů.

³ Online Analytical Processing

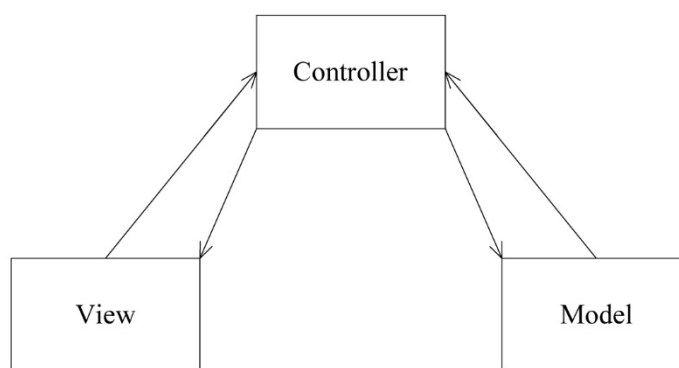
⁴ Office Information System

⁵ Electronic Data Interchange

3.3 Technologická architektura IS

Informační systémy lze také dělit podle jejich technologické architektury. Nejedná se přitom ještě o konkrétní technologii implementace. Jde o způsob, jakým je aplikace navržena. Za všechny můžeme zmínit například Service Oriented Architecture (SOA). Jde o architekturu, která definuje systém jako skupinu nezávislých komponent, které poskytují služby. Dalším příkladem architektury může být Model Driven Architecture. Ta se vyznačuje tím, že klade důraz na oddělení aplikační logiky od implementační platformy.

Velmi rozšířenou architekturou, zvláště u webových aplikací, je třívrstvá architektura. V tomto přístupu je systém rozdělený na tři části – model, view, controller. Základem je oddělení dat, logiky aplikace a výstupu. Třívrstvá architektura samozřejmě není jediná možná architektura a u velkých ERP systémů se dost často využívá i více vrstev. Princip třívrstvé architektury funguje následujícím způsobem:



Obr. 3. Schéma MVC architektury.

První částí je **Model**. Ten je většinou chápán jako datová struktura. Kromě toho, že uchovává data, měl by zároveň popisovat vztahy tak, jak jsou v reálném světě. Nejedná se tedy pouze o systém uložení dat. Model také obsahuje business pravidla a validační pravidla.

View se stará o zobrazení dat z modelu a ovládacích prvků uživateli. V ideálním případě by tato vrstva neměla obsahovat žádnou aplikační logiku. Výhodou tohoto přístupu je, že při programování různých pohledů (např. pro rozdílná zařízení) nevznikají duplicity v definici pravidel.

Controller se stará o logiku celé aplikace. Jak je vidět na předchozí stránce (Obr. 3), má vazbu jak na view, tak na model. Pokud chce uživatel přes presentační vrstvu něco zobrazit,

může přistoupit přímo k modelu. Pokud ovšem uživatel chce provést nějakou změnu v datech, je potřeba, aby se tak dělo právě přes controller [11].

Princip architektury, postavené na MVC, lze popsat na příkladu webové aplikace. Model by v tomto případě tvořila nejen samotná databáze, nýbrž i metody, které se starají o logiku či validaci dat. Model nemá informace o tom, kam data posílá, a jak budou data zobrazena. Naopak právě model by se měl starat o kontrolu dat, zda jsou konzistentní, jestli má heslo správnou délku, e-mail správný tvar apod. View zajišťuje správné zobrazení dat uživateli. Měl by obsahovat jen minimální množství logiky (např. cyklus pro výpis řádků tabulky). Často se nepoužívá jen čistý značkovací jazyk (HTML/XHTML), ale jazyk speciálně upravený (Smarty, Latte). Zprostředkovatelem mezi modelem, view a uživatelem je právě controller (PHP, ASP...). Na základě vstupu uživatele (buďto přímo skrze URL nebo přes zprostředkovaně přes pohled) si controller vyžádá od modelu data, která dále předá konkrétnímu view. Může také data získaná od uživatele do modelu uložit.

3.4 Projektování informačních systémů

Projektování informačních systémů je komplexní proces. Tento proces zahrnuje soubor metodik a nástrojů [19]. Ty slouží k úspěšnému navržení, zavedení a provozování informačního systému v podniku. Nejedná se tedy pouze o technickou realizaci daného systému, ale o všechny činnosti, které předchází nasazení informačního systému. V této kapitole jsou shrnuty důležité principy, které je doporučeno při navrhování dodržovat, a také jsou zde uvedeny nástroje, které mohou pomoci s projektováním.

Projektování informačního systému se neobejde bez modelu. Model je abstrakcí části reálného světa, který zanedbává nepodstatné detaily, a tím zjednodušuje pochopení reality [1]. Modely mohou být zcela obecného charakteru, kterým může být i slovní popis toho, jak by měl systém vypadat a co by měl dělat, až po velmi konkrétní (např. databázový model). Každý model má v určité části projektu své místo.

3.4.1 Metodiky vývoje softwaru

Existuje několik metodik, kterými lze vývoj informačního systému řídit.⁶ „Metodika je doporučený souhrn etap, přístupů, zásad, postupů, pravidel, dokumentů, řízení, metod technik a nástrojů pro tvůrce IS, který pokrývá celý životní cyklus IS.“ [6] Každá metodika v sobě obsahuje jednotlivé metody, techniky a nástroje. Obecně lze metodiky rozdělit na rigorózní a agilní. V posledních letech jsou agilní metodiky velmi oblíbené, ale zdaleka ne pro každý projekt je jejich využití vhodné.

Rozdíly mezi jednotlivými přístupy nejlépe shrnuje tabulka (Tab. 1) [20]. Z uvedené tabulky lze odvodit, že rigorózní metodiky (OPEN, RUP, EUP...) jsou vhodnější pro projekty většího rozsahu a ty projekty, u kterých je možné hned na začátku jasně definovat požadavky a cíle, kterých je potřeba dosáhnout. Agilní metodiky (XP, Scrum, FDD...) začaly vznikat v devadesátých letech. Vymezují se proti klasickým metodikám a jsou charakteristické tím, že dokáží velmi pružně reagovat na měnící se požadavky a systém je možné rychle přizpůsobovat.

Tab. 1. Porovnání rigorózních a agilních metodik.

	Rigorózní metodiky	Agilní metodiky
Náplň metodiky	Procesy, které se zaměřují na formálnost, pohlíží na lidi jako na sekundární faktor	Praktiky, které se zaměřují na znalosti jednotlivců, lidé jsou klíčovým faktorem
Podrobnost metodiky	Podrobný popis činností a procesů	Pouze základní struktura
Kvalita	Zaměření na kvalitu procesů a předpoklad, že kvalitní procesy povedou ke kvalitnímu výsledku	Zaměření na hodnotu pro zákazníka a vysokou kvalitu produktu

⁶ Jedná se o metodiky užívané pro jakýkoli software, nejsou vázány jen na informační systémy.

Předvídatelnost	Předpokládá předvídatelnost budoucnosti, klade důraz na anticipaci (sběr požadavků a plánování předem)	Klade důraz na adaptaci na změny (přírůstkové shromažďování požadavků, plánování pro iteraci)
Změny	Změny podléhají řízení změn a je snaha změny minimalizovat	Snaha změny umožnit s ohledem na nové znalosti
Definovatelnost procesu vývoje softwaru	Vývoj softwaru je definovaný proces, je možné jej opakovat bez problémů	Vývoj softwaru je empirický proces, nemůže být stejně opakován
Hodnota pro zákazníka	Zaměření na vlastní procesy, ne na hodnotu pro zákazníka	Nejvyšší prioritou je spokojenost zákazníka
Participace zákazníka na projektu	Jen v počátečních a koncových fázích	Zákazník je řídicím subjektem během celého projektu
Rozsah řešení	Snaha zabudovat všechny funkce, které by mohl zákazník v budoucnu využít	Pokryty pouze požadované funkce, požadavek minimalizace
Lidský faktor	Sekundární, role lidí jako zaměnitelné součástky	Primární, využívá individualit
Specialisté x generalisté	Vyžaduje úzké specializace lidí	Sdílení znalostí a spolupráce v týmu, spíše generalisté
Jednoduchost	Výsledkem spíše složité řešení	Důraz na jednoduchost
Modelování	Klade důraz na modelování, zejména modelování předem – „big design in front of“	Agilní modelování, nejde o model samotný, ale o proces modelování
Dokumentace	Rozsáhlá dokumentace	Podstatná není dokumentace, ale pochopení
Způsob vývoje	Vodopádový, iterativní a přírůstkový s dlouhými iteracemi	Přírůstkový vývoj s velmi krátkými iteracemi
Forma komunikace	Převážně písemná	Osobní

4 PROSTŘEDKY PRO ŘEŠENÍ PROBLEMATIKY

Dříve než bude přistoupeno k řešení problematiky konkrétního informačního systému, jsou zde krátce shrnuty některé technologie, se kterými se bude čtenář v praktické části setkávat. Jedná se prostředky k návrhu (UML, Wireframe) a realizaci (HTML, PHP MySQL...) systému.

V současné době se při vývoji aplikací v PHP lze velmi často setkat s různými typy *frameworků*. Přesný popis toho, co je to framework, nejspíš neexistuje. Často se uvádí, že se jedná o soubor knihoven nebo též vývojovou platformu. Následující citace se podle názoru autora diplomové práce nejvíc blíží skutečnosti: „*Framework bych definoval jako ucelený soubor tematicky zaměřených knihoven + policydecisions.*“ [15] Framework nám poskytuje množství knihoven, které lze pro tvorbu aplikace využít. To by sice bylo možné nahradit souborem vhodně vybraných knihoven, ale framework poskytuje i metodiku, jak aplikaci psát.

4.1 UML

UML (Unified Modeling Language) je modelovací jazyk, sloužící k modelování především softwarových systémů. Díky jeho univerzálnosti má ale mnohem širší použití. Jazyk UML však není sám o sobě metodikou. Jazyk nám nabízí nástroj, vizuální syntaxi, pomocí které můžeme modelovat zamýšlený systém. Je navržen tak, aby jej bylo možné využít v CASE nástrojích (Computer-aided software engineering).

S nástupem objektově orientovaných programovacích jazyků přišlo také několik jazyků, které nabízely vizuální modelování. Ovšem v roce 1997 došlo k tomu, že jazyk UML byl přijat sdružením OMG⁷ jako průmyslový standard. Ostatní jazyky začaly postupně upadat. Jazyk UML prošel několika změnami a doplněními a ve verzi 2 je dnes snad nejrozšířenějším jazykem sloužícím k modelování systémů [2].

Jak již bylo zmíněno, jazyk není spojen s žádnou konkrétní metodikou, životním cyklem ani konkrétním programovacím jazykem. Je to skutečně univerzální nástroj, který napomáhá

⁷Object Management Group (<http://www.omg.org/>)

tvorbě modelu systému. Má nástroje, které umožňují zobrazit požadavky, které jsou s klientem většinou řešeny ústní či písemnou formou. Díky tomu také pomáhá při komunikaci softwarového inženýra s klientem při vyjasňování požadavků.

4.2 HTML a CSS

Jazyk HTML (Hyper Text MarkupLanguage) je značkovací jazyk pro tvorbu webových stránek. Byl navržen v roce 1991. V první verzi by nešlo napsat nic, co si dnes představujeme pod pojmem web. První verze obsahovala pouze tagy pro rozčlenění textů do logických úrovní a pro vložení obrázků a hypertextových odkazů. Pod patronátem konsorcia W3C [22] byly postupně doplněné značky, které umožňují vkládat formuláře, tabulky aj. V dnešní době je již aktuální verze 4.01 a pracuje se na verzi HTML5.

Kaskádové styly (CSS [23]) se využívají k popisu vzhledu HTML dokumentů. Nabízí více možností než samotné formátování HTML a zároveň „čistí“ HTML strukturu, která potom nemusí obsahovat formátovací tagy, ale pouze popis struktury dokumentu. Jednotlivým tagům se přiřadí třídy či jednoznačné identifikátory, kterým následně v CSS nastavíme požadované formátování.

4.3 PHP

PHP [24] je jedním z nejrozšířenějších skriptovacích jazyků [4] sloužící zejména k tvorbě interaktivních webových stránek. Je to multiplatformní, nekompilovaný, dynamicky typovaný jazyk. Kvůli své poměrně volné syntaxi, nezávislosti na konkrétním systému, vynikající podpoře velkého množství databázových systémů a také kvůli open source licenci je stále velmi oblíbený. PHP podporuje od verze 5 plně práci s objekty a od verze 5.3 umožňuje též využití jmenných prostorů.

4.4 MySQL

MySQL [10] je relační databázový systém. Stejně jako PHP je multiplatformní a je také poskytován jako svobodný software (GPL Licence). MySQL nabízí několik různých typů úložišť (MyISAM, HEAP, InnoDB...). Pro tento projekt bylo zvoleno úložiště InnoDB. Hlavním důvodem je podpora cizích klíčů.

4.5 Nette Framework

Frameworků pro PHP existuje celá řada. Když začínal autor této diplomové práce s návrhem aplikace, neměl zkušenosti s žádným z nich a doposud si při psaní jiných aplikací vystačil se samotným PHP doplněným několika knihovnamí. Vybrat framework bez předchozí zkušenosti není nic jednoduchého. Existuje jich nepřeborné množství a rozdíly mezi nimi nejsou velké. Nejčastěji lze při hledání narazit na Zend Framework⁸, Symfony⁹, Laravel¹⁰, CakePHP¹¹ a v českém prostředí na velmi oblíbený Nette Framework. Právě Nette se často uvádí jako nejrozšířenější v České republice s největší komunitou u nás, přestože ve světě až tak známý není.

Nakonec byl zvolen výše zmíněný Nette, a to hlavně z následujících důvodů: rozsáhlá a aktivní komunita na fóru, Tracy¹², vlastní šablonovací systém a v neposlední řadě proto, že velká část dokumentace je v českém jazyce. Značnou nevýhodou by se mohlo na první pohled zdát, že Nette je dílem jediného člověka¹³. Postupně se však kolem frameworku utvořila velká komunita vývojářů, kteří pomáhají framework vylepšovat, píšou doplňky a v neposlední řadě jsou též velmi aktivní na fóru.

Je dobré uvést několik konkrétních funkcionalit, které Nette nabízí. Není záměrem ho porovnávat s ostatními frameworky, ale cílem je spíše zdůraznit, co využití frameworku nabízí programátorovi na rozdíl od psaní kódů bez něj. Jako velmi užitečné se jeví zejména: [21]

- Podpora architektury MVC.
- Zabezpečení aplikace. Nette automaticky ošetřuje všechny vstupy a výstupy. Velmi tak snižuje riziko narušení aplikace útoky jako XSS, CSRF, session hijacking, session stealing atd.
- Vlastní šablonovací systém Latte.
- Vlastní databázová vrstva.
- Velké množství znovupoužitelných komponent.

⁸ Snad nejkomplexnější framework s mnoha možnostmi nastavení. Pro svou rozsáhlost ne úplně vhodný pro menší projekty. [16]

⁹ Skládá se z jednotlivých knihoven, lze spravovat pomocí Composeru. [16]

¹⁰ V roce 2015 vůbec nejoblíbenější framework mezi vývojáři. [17]

¹¹ CakePHP má výborné zabezpečení, vlastní ORM knihovnu [16]

¹² Knihovna, která velmi usnadňuje ladění PHP kódu

¹³ Autorem Netteframeworku je David Grudl.

4.6 Bootstrap

Pro uvažovaný systém byl vybrán tzv. front-end framework, v tomto případě Bootstrap [12]. Jedná se o sadu komponent, které jsou velmi dobře využitelné při grafickém zpracování webové aplikace. Nabízí zabudovanou podporu pro responzivní web (gridsystem), jednoduchou definici stylů pro všechny běžné HTML prvky (text, formulářové prvky, tabulky...) a většinu dnes používaných prvků webových stránek (menu, štítky, záložky). Dále existuje i několik komponent využívajících JavaScript, které ještě rozšiřují možnosti uživatelského rozhraní.

Bootstrap samozřejmě nemůže nahradit práci grafika na velkých webových projektech, ale pro potřeby informačního systému jej lze považovat za dostatečný. Pro běžné uživatele bude vzhled dostatečně přívětivý a programátorovi umožní se plně soustředit na návrh grafického rozhraní, místo toho, aby se zabýval implementací a optimalizací pro jednotlivé prohlížeče.

4.7 Programové vybavení

Při zpracování diplomové práce byly využity i další podpůrné nástroje. Pro zpracování modelu tříd to byl ArgoUML, jednoduché nákresy byly vytvořeny v grafickém editoru GIMP. Model databáze byl zpracován v softwaru MySQL Workbench a jako vývojové prostředí pro implementaci byl zvolen program NetBeans.

II. PRAKTICKÁ ČÁST

5 POŽADAVKY NA NOVÝ INFORMAČNÍ SYSTÉM

Autor diplomové práce je zaměstnán olomouckém arcibiskupství od roku 2008, a je proto dobře obeznámen s potřebami jednotlivých oddělení. Ve spolupráci s vedoucím ICT oddělení byly vytipovány nejaktuálnější problémy, jejichž řešení by měl nový informační systém podpořit. Za účelem zjištění očekávání od zavedení nového IS se konalo několik schůzek s pracovníky oddělení, kterých se bude zavedení nového systému týkat. Na základě společných analýz byly vybrány činnosti, které bude IS pokrývat. V této kapitole jsou tyto činnosti stručně popsány.

Cílem této kapitoly je vytvoření uceleného seznamu požadavků na budoucí informační systém.

5.1 Ekonomické oddělení

Na základě společného šetření se zaměstnanci ekonomického oddělení vyvstaly dvě činnosti, které jsou zde zpracovávány ručně, a jejichž alespoň částečné zautomatizování by pracovníkům ušetřilo spoustu času. Jde o evidenci inventur jednotlivých farností a správu plateb do tzv. fondů.

Evidence inventur je v jednotlivých farnostech vedena v programu MS Excel. Není to ideální řešení, ale dle pracovníků dostačující. Základním požadavkem je, aby inventurní soupisy z jednotlivých farností byly centrálně evidovány na arcibiskupství. V současnosti se tak děje pouze v počítači jednoho z pracovníků, který udržuje aktuální soubory s inventurami v jednom adresáři. Odpovědní pracovníci z farností mu zasílají aktualizované soubory emailem či fyzicky na optickém nosiči. Jakákoli kontrola aktuálnosti takových souborů je velmi pracná.

Příspěvky do fondů jsou agenda, která si zaslouží větší pozornost. Jedná se o platby subjektů (jednotlivých farností) dle ročního předpisu. Tyto jsou zasílány na účet arcibiskupství nebo přinášeny v hotovosti na pokladnu. Je potřeba rozlišit a udržovat informace o tom, do kterého fondu který subjekt kolik přispěl. Doposud tyto platby odpovědná pracovnice vyhledávala ve výpisech z banky a ručně zadávala do softwaru, který byl naprogramován přímo pro správu fondů v jazyce FoxPro v prostředí MS-DOS. Ten má několik zásadních omezení. První je čistě technického rázu – v nových operačních systémech jej nelze nativně spouštět, a je proto potřeba využívat emulátoru DOSBox. Pro zadání nového fondu či pře-

chodu na nový rok je potřeba zasahovat do kódu. Systém taky neumožňuje hromadné načítání plateb. Další velkou nevýhodou je, že program běží pouze na jednom počítači, a tím pádem vznikají také problémy, pokud není k dispozici dotyčná pracovnice. Není též možnost nahlížet do systému z jednotlivých farností či děkanátů pro kontrolu plateb.

5.2 Mzdové oddělení

Na mzdovém oddělení se počítají mzdy pro cca 250 laických pracovníků z diecéze. Získávání podkladů pro výpočet mezd se stále řeší papírovou formou. Pracovníci v diecézi vyplňují dva formuláře – výkaz odpracované doby a výkaz popisu práce. Výkaz odpracované doby nechají jednotliví zaměstnanci podepsat vedoucím a odesílají na mzdové oddělení arcibiskupství. Tady se údaje pro výpočet mzdy ručně opisují do mzdového softwaru.

5.3 Evidence arcidiecéze

Pracovnice tohoto oddělení má mimo jiné na starosti zpracování statistických údajů o duchovních úkonech. Ve farnostech se každoročně vyplňují formuláře o provedených úkonech a zasílají se na arcibiskupství. Zde se údaje opíší do tabulkového procesoru (MS Excel) a dále se s nimi pracuje v této podobě. Formuláře mají okolo jednoho sta položek, a to je při více než čtyřech stech farnostech nezanedbatelné množství, které se musí opsat po počítače. Paradoxem je, že v některých farnostech se dokonce formuláře vyplňují elektronicky, tisknou se a znovu se potom digitalizují.

5.4 Stavební oddělení

Toto oddělení má na starosti poměrně rozsáhlou agendu. Kromě jiných činností mají pracovníci stavebního oddělení na starosti evidenci oprav církevních objektů ve všech farnostech. Pracovníci, kteří jsou za tuto činnost odpovědní, podrobně vyplňují, která stavba se bude opravovat, kde se nachází a jaký druh opravy je v plánu realizovat. Na základě těchto informací má stavební oddělení jednak přehled o tom, co se během roku opravuje, a taky podle tohoto přehledu může finančně vypomáhat poskytnutím daru či půjčky. Při každé změně plánu oprav (schválení či neschválení dotace apod.) je potřeba z diecéze informovat pracovníky stavebního oddělení. Je nutné udržovat tento seznam aktualizovaný.

5.5 Přehled požadavků

Po analýze jednotlivých agend byl vytvořen seznam funkčních požadavků, které jednotlivá oddělení na IS mají. Požadavky jsou shrnuty v následující tabulce (Tab. 2).

Tab. 2. Tabulka funkčních požadavků.

ID	Definice požadavku
	Obecné požadavky
FR1.01	Uživatel se musí přihlásit do systému.
FR1.02	Uživatel musí mít možnost změnit heslo.
FR1.03	Uživatel musí mít možnost nechat si zaslat nové heslo na e-mail.
FR1.04	Administrátor systému bude jednotlivým uživatelům přiřazovat role.
FR1.05	Farnost bude mít možnost delegovat oprávnění pro dílčí činnosti na jednotlivé uživatele.
	Evidence inventur
FR2.01	Uživatel musí mít možnost vložit soubor s inventárním soupisem.
FR2.02	Inventurní soubory se vkládají za jednotlivé farnosti.
FR2.03	Pracovníci musí mít možnost vkládat soubory odkudkoli z diecéze.
FR2.04	Odpovědný pracovník biskupství musí vidět aktuální stav inventur.
FR2.05	Odpovědný pracovník bude mít možnost hromadně upozornit farnosti, které nemají aktuální inventuru.
	Příspěvky do fondů
FR3.01	Systém bude evidovat platby od jednotlivých subjektů.
FR3.02	Pracovnice pokladny bude zadávat platby hromadně (importem bankovního výpisu) i jednotlivě.
FR3.03	Pracovnice pokladny bude sama spravovat fondy (přidávat, měnit, odebírat).
FR3.04	Systém musí nabízet různé typy pohledů: podle fondů, podle přispěvatelů, a to vždy s ohledem na zadaný rok.

FR3.05	Systém musí umožňovat export jednotlivých pohledů do excelu pro další zpracování.
FR3.06	Příspěvatelé budou mít přístup k zobrazení vlastních příspěvků a zobrazení případných přeplatků či nedoplatků.
FR3.07	Děkani budou mít přehled o platbách za farnosti patřící do děkanátu.
	Výkazy práce
FR4.01	Zaměstnanci budou do systému zadávat odpracovanou dobu a vykonanou práci.
FR4.02	Vedoucí pracovník musí mít přístup k výkazům práce podřízených zaměstnanců.
FR4.03	Vedoucí pracovník musí mít možnost schválit potvrzený výkaz práce podřízených zaměstnanců.
FR4.04	Mzdová účetní musí mít přístup k odpracovaným hodinám všech pracovníků, ne však k popisu vykonané práce.
FR4.05	Supervizoři musí mít možnost přístupu k popisu vykonané práce vybraných pracovníků.
FR4.06	Systém bude umožňovat export sald kont pracovní doby za všechny pracovníky tak, aby odpovídal formátu pro import do mzdového softwaru.
FR4.07	Systém bude umožňovat hromadný tisk za všechny pracovníky a příslušný měsíc pro archivaci.
	Hlášení o úkonech v duchovní správě
FR5.01	Správci jednotlivých farností musí mít možnost vyplňovat údaje o úkonech v duchovní správě. Výkazy se odevzdávají ročně.
FR5.02	Odpovědný pracovník musí mít přehled o farnostech, zdali mají vyplněné údaje, nebo ne.
FR5.03	Odpovědný pracovník bude mít přístup jak k jednotlivým výkazům, tak k celkovému přehledu.

FR5.04	Odpovědný pracovník musí mít možnost exportovat celkový přehled do souboru MS Excel.
FR5.05	Odpovědný pracovník (administrátor) bude mít možnost operativně měnit jednotlivé položky pro každý rok.
	Hlášení oprav
FR6.01	Pracovníci v děkanátech musí do systému zadávat všechny plánované opravy církevních objektů.
FR6.02	U každé opravy je třeba zadat, o jaký objekt se jedná, kde se nachází, co se na něm bude opravovat a z jakých zdrojů je plánováno financování.
FR6.03	U každého objektu musí být možnost uvést, zda je památkově chráněný.
FR6.04	Typy oprav a zdroje financování jsou jasně definované a je potřeba, aby byly zadány pokaždé stejně.
FR6.05	Oprávněný uživatel bude mít možnost spravovat typy oprav a zdroje financování.
FR6.06	Systém musí umět zajistit, aby odpovědný pracovník biskupství měl možnost zjistit, kdy byla provedena konkrétní aktualizace u daného objektu.
FR6.07	Systém musí umět zadat, že se v dané farnosti žádný objekt neopravuje.

5.6 Další požadavky na systém

V předchozích kapitolách byly popsány požadavky, které mají na systém jednotliví odpovědní pracovníci. Na tomto místě jsou uvedeny některé nefunkční požadavky¹⁴, které vyplynuly většinou z analýzy plánovaného systému pracovníky ICT oddělení. Jedná se o požadavky, které nezahrnují konkrétní funkcionalitu systému, ale dále pomáhají upřesnit daný systém.

¹⁴ Dle UML nefunkční požadavky specifikují omezení pro daný systém.

Požadavky byly definovány s ohledem na znalosti prostředí a uživatelů a na případný budoucí rozvoj informačního systému. Mimo jiné bylo potřeba rozhodnout, jestli systém bude provozován jako webová nebo desktopová aplikace. Protože jsou v poslední době čím dál častěji využívána mobilní zařízení (telefony, tablety), byl zvolen webový přístup, aby nebylo do budoucna nutné udržovat větší množství klientských aplikací pro různé typy systémů.

Tab. 3. Tabulka nefunkčních požadavků.

ID	Definice požadavku
NR1.01	Systém bude implementován jako webová aplikace.
NR1.02	Systém bude napsán v jazyce PHP s využitím uložení dat v databázi MySQL.
NR1.03	Systém musí být dostupný v prostředí internetu.
NR1.04	Uživatelé se do systému budou přihlašovat zadáním e-mailu a hesla.

6 ZHODNOCENÍ SOUČASNÉHO STAVU A NÁVRH ŘEŠENÍ

V předchozích kapitolách bylo popsáno, jaké systémy se na arcibiskupství provozují. Dále byly zaznamenány požadavky jednotlivých oddělení. Po dohodě s vedoucími pracovníky bylo rozhodnuto výše zmíněné agendy implementovat do informačního systému. Nabízely se dvě možnosti – rozšířit jeden či více ze stávajících systémů, nebo navrhnout vlastní systém.

6.1 Zhodnocení současného stavu

Jak již bylo uvedeno, na arcibiskupství není provozován centrální ERP systém a zavedení hlavního systému není vedením arcibiskupství plánováno. Je provozováno několik vzájemně oddělených informačních systémů. Ty jsou specializované pro určitá oddělení a většina z nich byla zakoupena (či jsou pronajata) jako konkrétní produkt.

Stále více se uvažuje o zavedení nového informačního systému, který by pokrýval nově vznikající požadavky. Jedním ze zásadních důvodů pro zavedení nového či rozšíření některého ze stávajících systémů je fakt, že neustále narůstá objem informací, které se vyměňují mezi externími pracovníky a pracovníky kurie. Tato skutečnost se musí při volbě řešení zohlednit.

6.2 Rozšíření některého ze stávajících systémů

Prvním krokem bylo prověření, zda by rozšíření některého ze stávajících systémů nemohlo pokrýt výše zmíněné agendy. Nejvhodnějším kandidátem byl na první ekonomický software Pohoda. Velmi rychle se ale objevily obtíže. Firma žádný modul, který by splňoval požadavky, nenabízí. Programování nového modulu by nedávalo smysl. Snad jen příspěvky do fondů by byl svým zaměřením software schopný pokrýt, ale vznikl by problém s přístupovými právy. Je potřeba, aby do systému mělo přístup cca 500 lidí a každý by měl mít možnost pracovat s jinými daty. Takto ale není software koncipován. Pracuje na systému účetních jednotek a přístupu k nim, což z praktického hlediska pro toto využití nevyhovuje.

Jako velmi dobrá možnost pro zpracování výkazů práce se jeví využití stávajícího docházkového systému. Ovšem i zde se vyskytlo několik problémů. Systém spolupracuje s terminály, na kterých si zaměstnanci kurie značí průchody přes vrátnici. Bylo by tedy potřeba nakoupit terminály na všechna ostatní pracoviště. Ale na některých z nich jsou pouze dva

nebo tři zaměstnanci. Investovat tolik peněz do rozšíření systému by bylo minimálně rozpočetné řešení. Pokud bychom se rozhodli systém využít pro externí zaměstnance i bez zavedení terminálů, museli by si zaměstnanci vypisovat do systému pracovní dobu „ručně.“ To by na jednu stranu bylo řešení, není to však příliš uživatelsky přívětivé řešení. Systém je primárně určen pro spolupráci s terminály a není uzpůsoben pro pravidelné zadávání průchodů přes aplikaci. Dalším problémem je, že systém neobsahuje evidenci pracovní náplně, což je jeden z klíčových požadavků vedení.

Kvůli těmto problémům bylo oddělením ICT rozhodnuto, že je nutné najít jiné, komplexnější řešení.

6.3 Návrh vlastního systému

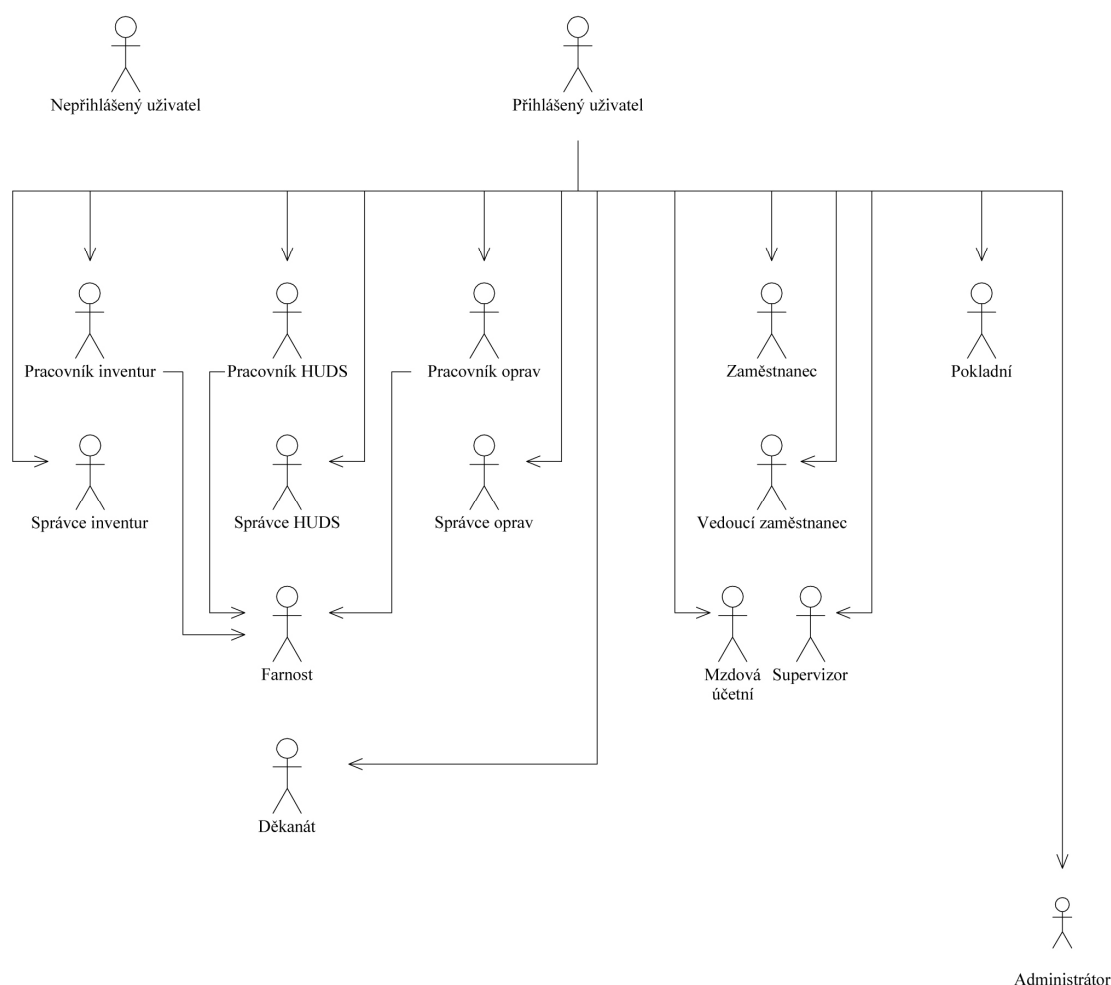
Z výše uvedeného vyplývá, že návrh a vývoj vlastního systému se jeví jako lepší varianta. A to především z následujících důvodů:

- Vlastní IS bude nejlépe pokrývat specifické požadavky biskupství.
- Protože systém nebude svým rozsahem odpovídat klasickému ERP, ale bude spíše doplňovat stávající systémy, náklady na analýzu a implementaci by nemusely být nikterak vysoké.
- Pokud bude systém realizován přímo pracovníky arcibiskupství, bude výhodou znalost prostředí. V projektu by se hned od začátku dalo počítat s dalšími možnými požadavky.
- Dalším argumentem, proč výše zmíněné agendy řešit jednotným systémem, je společná množina budoucích uživatelů. Přestože požadavky na systém vycházejí z různých oddělení, budou systém využívat převážně externí pracovníci (co se týče zadávání dat). Pro zaměstnance úřadu kurie bude systém poskytovat především výstupy z těchto dat.

7 NÁVRH MODELU SYSTÉMU

V této kapitole je popsána struktura informačního systému, přičemž nejprve jsou identifikováni jednotliví aktéři, kteří budou se systémem interagovat. Dále jsou navrženy jednotlivé případy užití a diagram tříd. K modelování jsou využita grafická znázornění pomocí UML, které vychází z knihy UML a unifikovaný proces vývoje aplikací [2].

7.1 Aktéři systému



Obr. 4. Přehled aktérů systému.

V informačním systému je definováno 16 aktérů (Obr. 4). Podstatnou část uživatelů v diecézi, kteří se budou do systému přihlašovat, tvoří role *Farnost*. Tato role má možnost přistupovat k velké části funkcionality. Někde však správce farnosti potřebuje delegovat konkrétní činnosti na jiné pracovníky. Není žádoucí umožnit pracovníkům přístup ke všem

částí systému za konkrétní farnost. Proto jsou v systému role, které umožňují přístup k dílčím činnostem, ke kterým má farnost oprávnění. Z tohoto důvodu se některé role budou překrývat, a tak k jedné činnosti bude mít oprávnění více rolí.

Role *Farnost* a *Děkanát* jsou svým způsobem specifické. Nelze je v systému jednotlivě přiřazovat, ale jsou již přiřazeny konkrétním uživatelským účtům.

Tento přehled není exaktním popisem, co přesně daný aktér může v systému dělat. Slouží k přiblížení a hlubšímu pochopení problematiky. Konkrétní případy užití jsou popsány v další podkapitole.

Nepřihlášený uživatel

Protože se jedná o informační systém, který neobsahuje veřejnou část, nepřihlášený uživatel nemá v podstatě žádná práva. Může se do systému pouze přihlásit, nebo si požádat o zaslání nového hesla, pokud své zapomněl. Registrace umožněna není, všechny uživatele musí do systému zadat administrátor.

Přihlášený uživatel

Po úspěšném ověření uživatelského jména a hesla je uživatel přihlášený do systému. Může upravovat své osobní a kontaktní údaje. V systému existuje několik rolí a přihlášený uživatel může mít přidělenou žádnou, jednu nebo více rolí. Na základě těchto rolí může přistupovat k jednotlivým částem systému.

Farnost

Farnost je právnická osoba, která má specifickou úlohu. Umožňuje vykonávat všechny činnosti, které souvisí s farní agendou. Jde zejména o hlášení oprav, zadávání úkonů v duchovní správě, vkládání inventurních soupisů, přístup k přehledu zaplacených příspěvků do fondů a některé další.

Děkanát

Děkanát je z pohledu uspořádání diecéze správní funkce. V systému má přehled o vyplněných opravách, hlášení o úkonech v duchovní správě, inventurních soupisech, zaplacených příspěvcích do fondů za ty farnosti, které do daného děkanátu patří. Do systému nic nekládá.

Pracovník inventur

Tato role slouží k přehledu a nahrávání inventurních soupisů. Uživatel může mít oprávnění spravovat více farností.

Správce inventur

Správce inventur kontroluje a udržuje databázi odevzdaných inventárních soupisů. Má přístup ke všem odevzdaným souborům, vidí taky poslední aktualizaci souboru a má možnost hromadně upozornit ty farnosti, které nemají delší dobu aktualizovaný inventární soupis.

Pokladní

Pokladní se stará o evidenci plateb do fondů. Může zadávat, měnit a mazat platby. Taktéž může přidávat a odebírat jednotlivé fondy. Má možnost hromadně importovat příchozí platby na účet. Dále má také oprávnění zobrazovat platby podle fondů, příspěvateľů, data přijetí či času vložení. Může také předdefinované sestavy exportovat do formátu CSV a XLS.

Zaměstnanec

Zaměstnanec se v tomto systému rozumí uživatel, který vyplňuje výkaz práce. Má možnost vytvořit, smazat, vyplnit a potvrdit pracovní list. Může si také zobrazit zbývajících dny dovolené.

Vedoucí zaměstnanec

Vedoucí zaměstnanec má pod sebou jednoho či více zaměstnanců, kterým je oprávněn schvalovat výkazy práce. Pokud zaměstnanec potvrdil chybně vyplněný výkaz práce, vedoucí zaměstnanec jej může odemknout, a umožnit tak zaměstnanci ho znova upravovat.

Supervizor

Supervizor se stará o kontrolu pracovních činností jednotlivých zaměstnanců. Má přístup k té části výkazu práce, kam zaměstnanec vyplňuje činnosti, které v dané dny dělal. Nemá ale přístup k docházce konkrétního zaměstnance.

Mzdová účetní

Uživatel v této roli má přístup k docházce všech zaměstnanců. Může si jednotlivě či hromadně zobrazit výkazy práce a exportovat je do formátu PDF. Též může exportovat salda kont pracovní doby kvůli importu do mzdového systému.

Pracovník oprav

Pracovník oprav může zadávat požadavky na opravy v jednotlivých farnostech. Má též možnost zadat, že farnost na daný rok neplánuje žádnou opravu.

Správce oprav

Správce má přehled o celkovém stavu vyplněných plánovaných oprav. Vidí také ty farnosti, které ještě nezadaly žádnou opravu na daný rok a má možnost je na tuto skutečnost upozornit e-mailem. Má přístup k celkovým částkám na opravy dle jednotlivých zdrojů financování. Může upravovat číselníky druhů oprav a zdrojů financování.

Pracovník HUDS¹⁵

Pracovník vytváří výkazy o úkonech v duchovní správě, vyplňuje je a potvrzuje. Má možnost zpětně přistupovat ke všem vyplněným výkazům. Může si také jednotlivé výkazy exportovat do formátu PDF.

Správce HUDS

Správce si může jednotlivé výkazy zobrazit a exportovat do formátu PDF. Kontroluje také, které farnosti již výkaz potvrdily, a na nepotvrzené výkazy může upozornit e-mailem. Také si může všechny údaje za daný rok exportovat ve formátu MS Excel pro další zpracování.

Administrátor

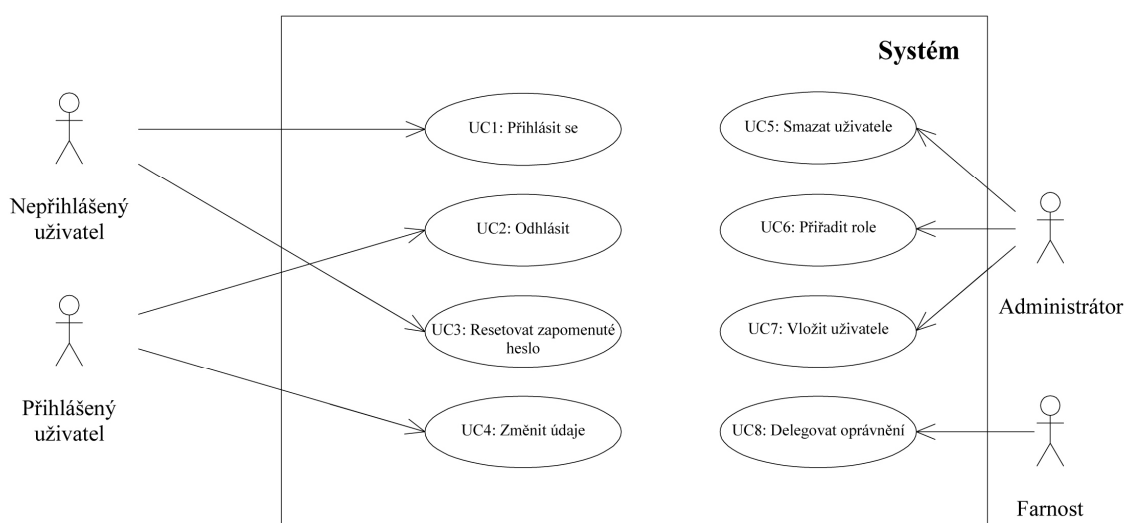
Administrátor v první řadě vkládá, upravuje, odstraňuje uživatele a přiřazuje jednotlivým uživatelům role. Dále může upravovat všechny číselníky.

V širším slova smyslu je administrátor ten, kdo systém instaluje a má přístup ke zdrojovým kódům a k databázi.

7.2 Specifikace případů užití

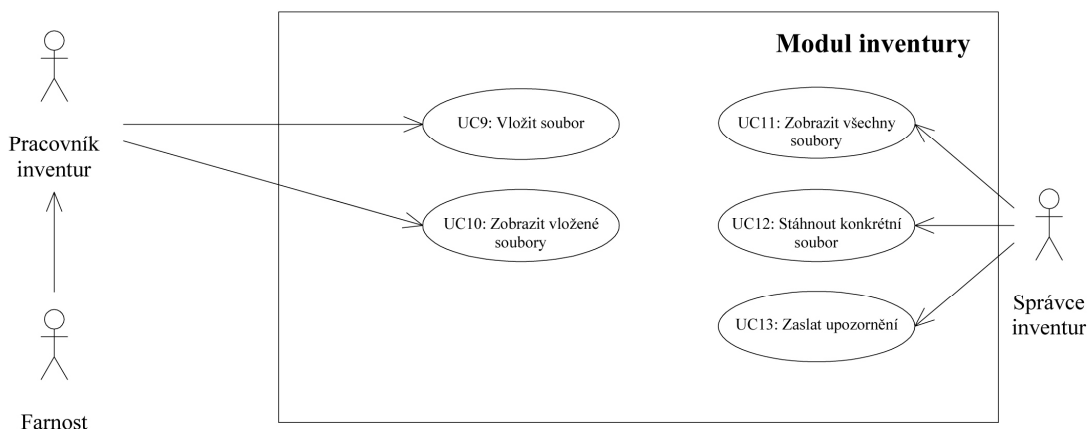
V předchozí části práce jsou popsáni jednotliví aktéři systému. Následuje uvedení jednotlivých diagramů případů užití s krátkým popisem. Diagramy případů užití znázorňují jednotlivé funkční požadavky a aktéry. Každý aktér může přistupovat k jiné části systému.

¹⁵HUDS – hlášení o úkonech v duchovní správě



Obr. 5. UC diagram systémové části.

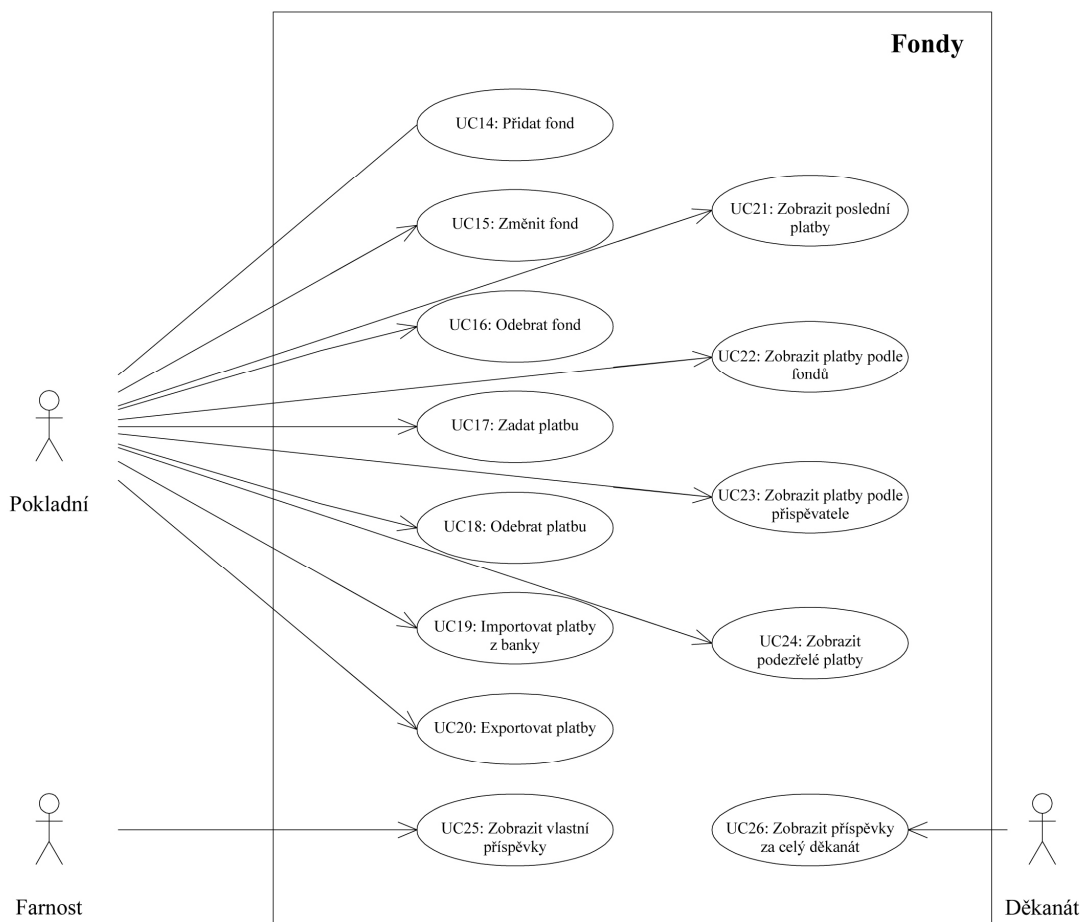
První diagram (Obr. 5) zahrnuje případy užití, které jsou společné pro celý systém. Jedná se o přihlášení a odhlášení uživatele, zaslání e-mailu pro nastavení nového hesla. *Administrátor* může přidávat uživatele a již existujícím uživatelům měnit role. Kvůli evidenci není možné uživatele odebírat, *administrátor* má možnost uživatelům zakázat přihlášení do aplikace.



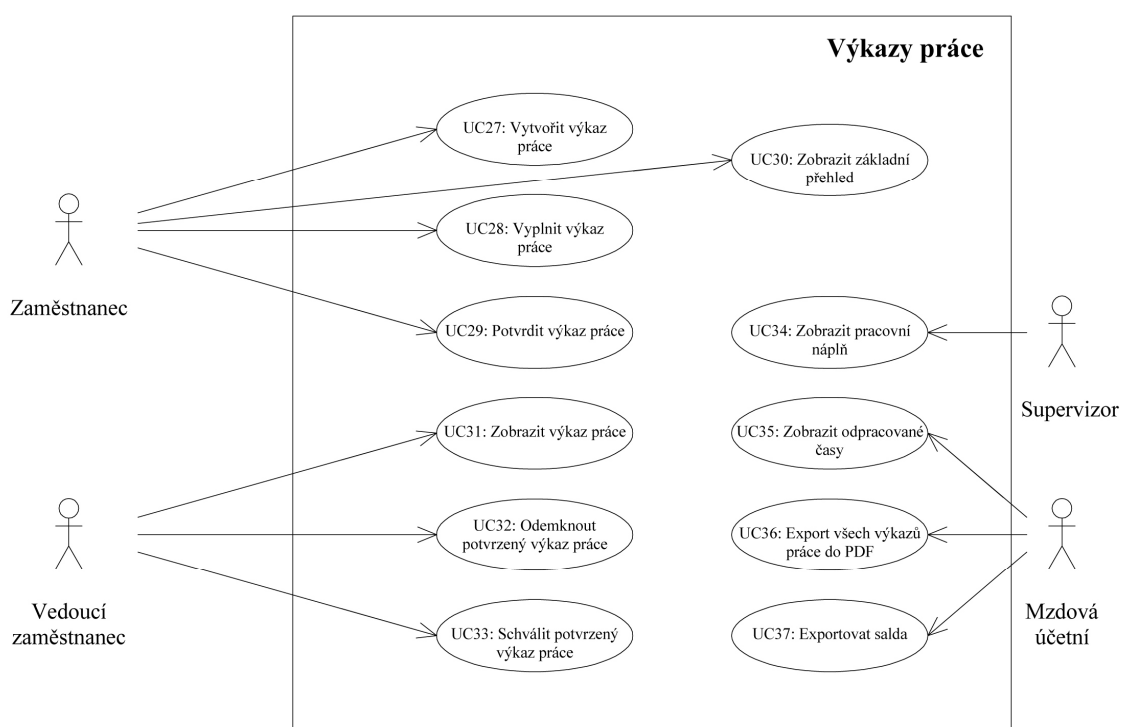
Obr. 6. UC diagram modulu inventury.

Další diagram (Obr. 6) znázorňuje jednoduchý modul inventur, který slouží k evidenci inventurních soupisů za jednotlivé subjekty. Za každý subjekt je vždy aktuální jeden soupis. Kvůli zpětnému přehledu není žádoucí jednotlivé záznamy mazat, proto má *pracovník inventur* pouze možnost vkládat nové soubory. *Správce inventur* může ke všem inventurám přistupovat, a také má možnost zaslat upozornění subjektům, které mají zastaralý inventurní soupis. Za vkládání inventurních souborů jsou odpovědné *farnosti*. Ty mohou tuto roli delegovat na jiné uživatele systému.

Modul fondů slouží k evidenci příspěvků od farností do jednotlivých fondů. Tuto agendu zpracovává jeden člověk, a proto diagram (Obr. 7) přiřazuje většinu případů užití *Pokladní*. Ta má na starost celou agendu (spravuje fondy, zadává příspěvky atd.). Ostatní aktéři (*Farnost*, *Děkanát*) mají přístup pouze k přehledům.

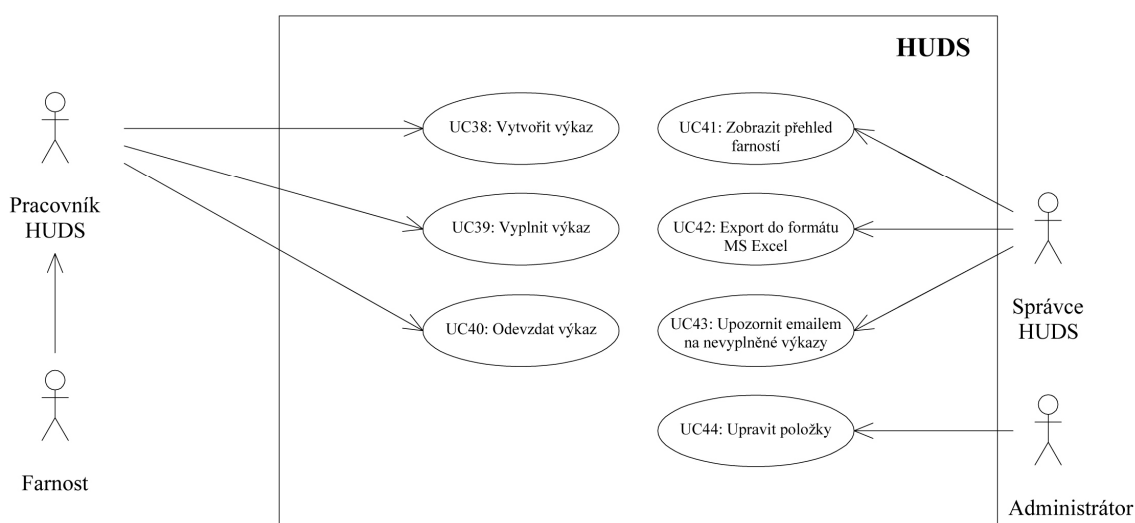


Obr. 7. UC diagram modulu fondy.



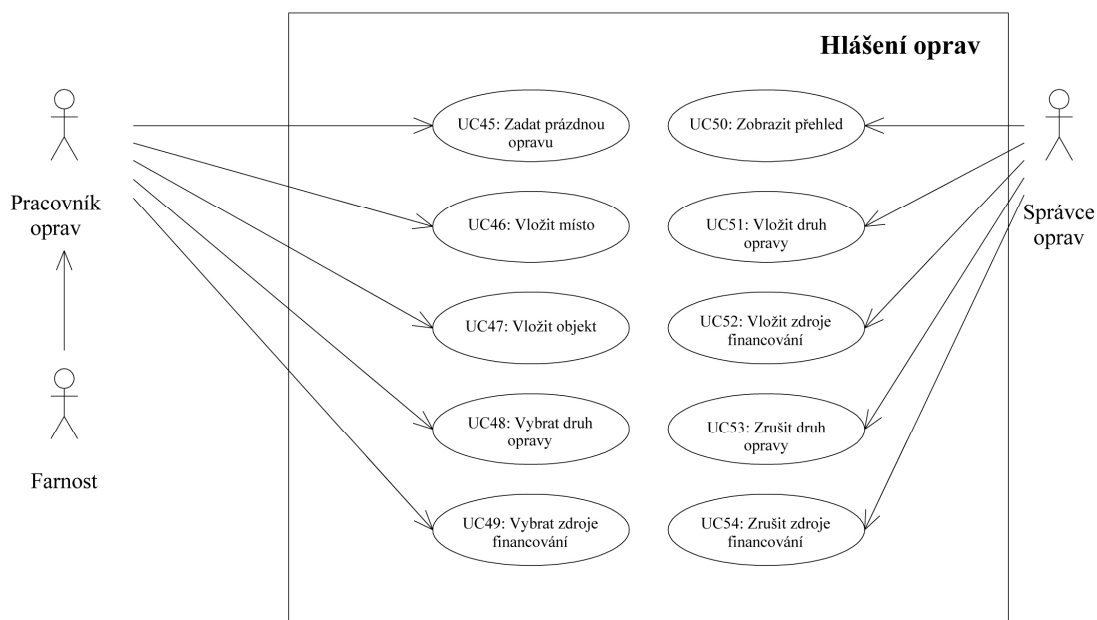
Obr. 8. UC diagram modulu výkazy práce.

Výkazy práce (Obr. 8) jsou jedinou částí systému, ve které nefigurují role *Farnost* a *Děkanát*. Nejsou na tomto uspořádání nijak závislé. *Zaměstnanec* vyplňuje pracovní listy, *Vedoucí zaměstnanec* je potom schvaluje. Až po schválení se pracovní listy zobrazují *Mzdové účetní*. Role *Supervizora* slouží ke kontrole práce jednotlivých zaměstnanců, proto má přístup pouze k soupisu vykonané práce zaměstnanců.



Obr. 9. UC diagram modulu hlášení úkonů v duchovní správě.

Výkazy o úkonech v duchovní správě vyplňují jednotlivé farnosti či delegovaní pracovníci. Po potvrzení (odevzdání výkazu) jsou zobrazeny *Správci*. Ten potom může data dále hromadně zpracovávat.



Obr. 10. UC diagram modulu hlášení oprav.

Hlášení oprav je prováděno jednotlivými *farnostmi* (či oprávněnými pracovníky). Každá oprava se v systému skládá z několika částí, aby bylo možné efektivně agregovat data. Proto jsou úkony rozděleny na více UC. *Správce oprav* může upravovat číselníky a zobrazit celkový přehled.

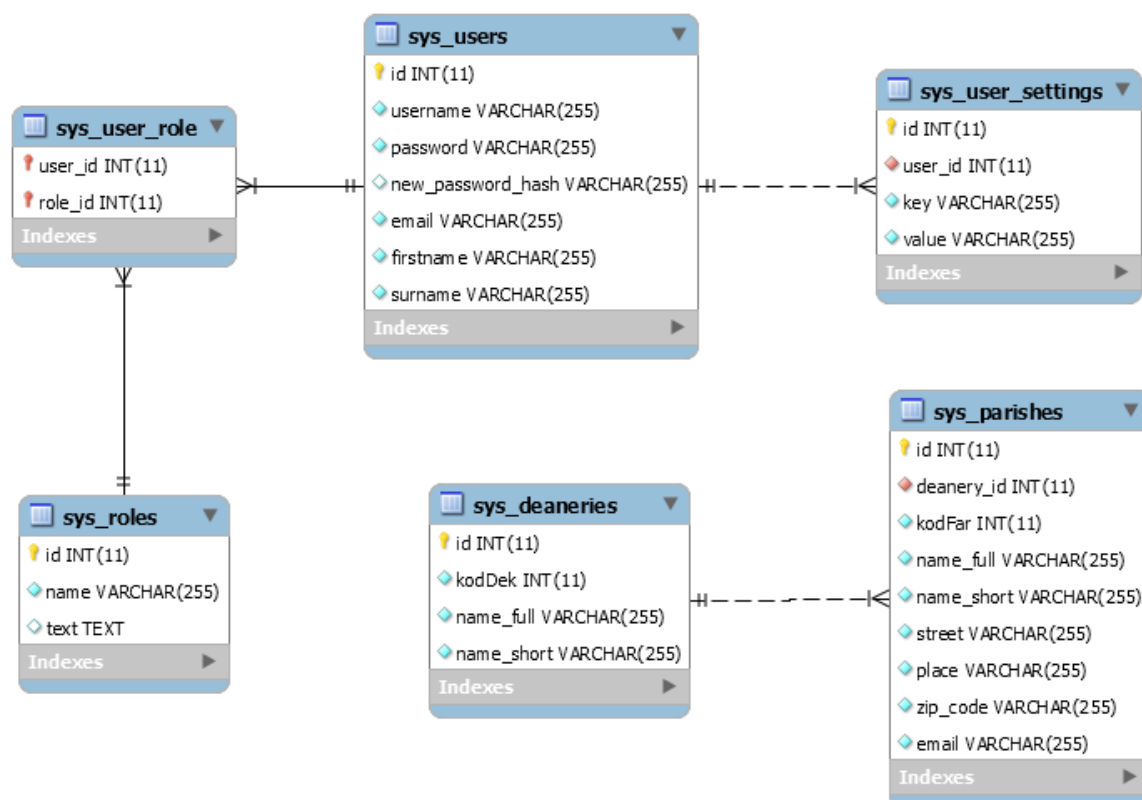
V příloze (P II) jsou podrobně popsány scénáře k jednotlivým případům užití. V tabulce (P III) bylo průběžně sledováno, zda jsou všechny požadavky pokryty alespoň jedním případem užití.

7.3 Model tříd

Jako další krok návrhu byl vytvořen model tříd (P IV). Ten popisuje třídy, jejich základní atributy a metody a také vztahy mezi jednotlivými třídami. Slouží jako rozšířené znázornění objektů problémové domény a jejich základních atributů a metod. Nejedná se o model, který znázorňuje konkrétní implementaci.

8 NÁVRH DATOVÉHO MODELU

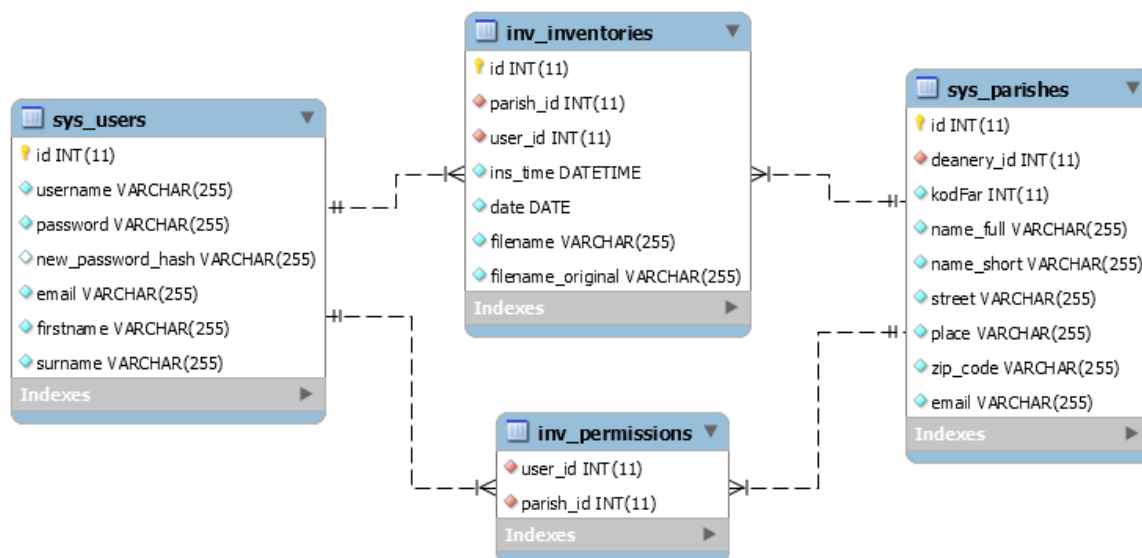
Informační systém bez možnosti ukládat data by jen těžko posloužil svému účelu. Proto je potřeba věnovat návrhu datového modelu dostatečnou pozornost. Pro tento systém byl zvolen způsob uložení dat pomocí relační databáze. Konkrétně se bude jednat o databázový systém MySQL. Aby byla databáze přehlednější a související tabulky mohly být při výpisu řazeny u sebe, jsou u názvů tabulek použity prefixy. Ty rozdělují tabulky podle jednotlivých funkčních celků (modulů). Náčrt celého datového modelu je v příloze (P V).



Obr. 11. Schéma systémové části databáze.

Na obrázku (Obr. 11) je vidět základní (systémová) část datového modelu. Konkrétně se jedná o tabulky uživatelů, rolí, farností a děkanátů. Nejsou zde znázorněny vazby na tabulky v dalších modulech. Do systému se o subjektech ukládají jen nejnutnější informace, které jsou pro funkci systému nezbytné. Konkrétní evidence zaměstnanců, farností a děkanátů je na arcibiskupství vedena v jiných systémech. Sloupce `sys_parishes.kodFar`, `sys_deaneries.kodDek` a `sys_users.osCislo` slouží k propojení na externí systémy. Jeden uživatel může mít více rolí, proto tabulka `sys_user_has_role`, která ukládá ID role a ID uživatele.

8.1 Modul inventur

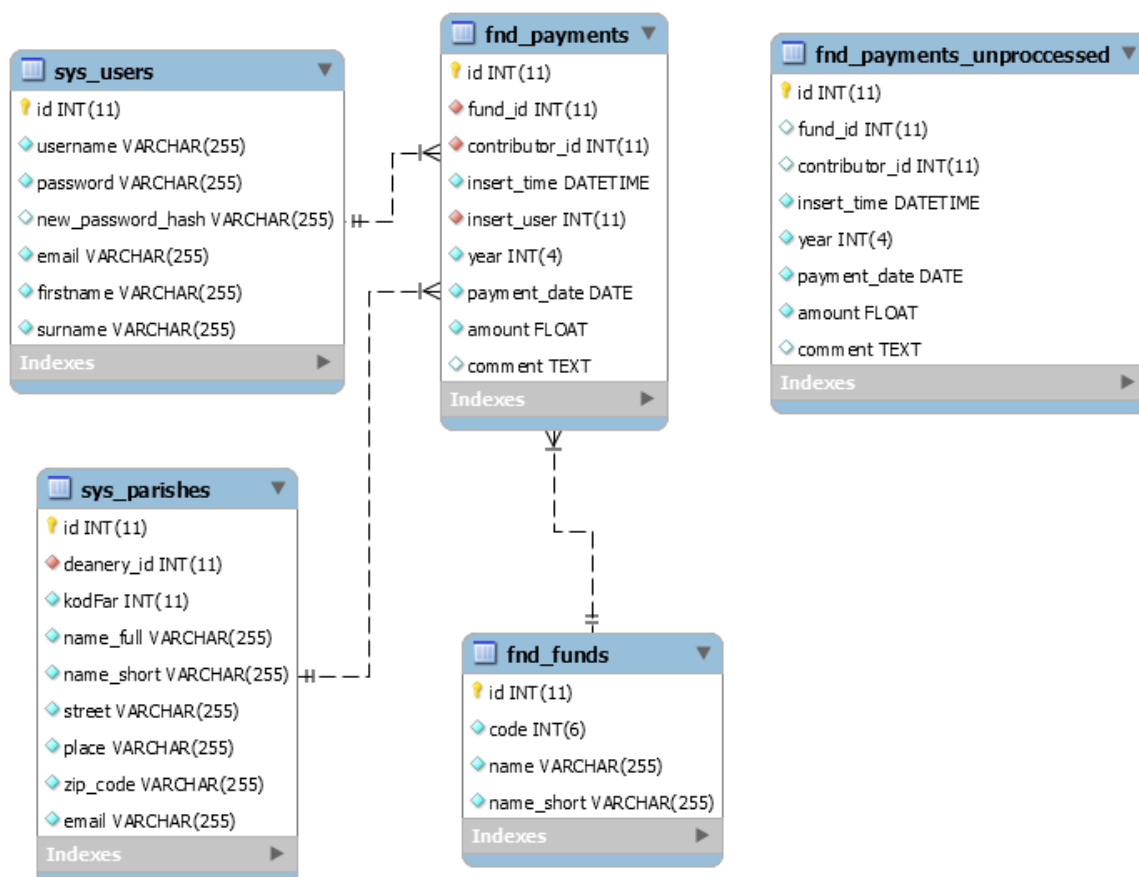


Obr. 12. Schéma modulu inventur s vazbou na farnosti a uživatele.

Z datového pohledu je nejjednodušší modul inventur. Ten obsahuje jen dvě tabulky `inv_inventories` a `inv_permissions`. Tabulka `inv_permissions` slouží k udělení přístupu uživateli ke konkrétní farnosti. Tabulka `inv_inventories` obsahuje záznamy o jednotlivých odevzdaných souborech s inventurami. Samotné soubory nejsou v databázi uloženy. Popis vybraných atributů:

- `parish_id` – cizí klíč, odkazuje do tabulky farností.
- `user_id` – cizí klíč, slouží k identifikaci uživatele, který soubor nahrál.
- `filename` – název fyzického souboru.

8.2 Modul fondů



Obr. 13. Schéma modulu fondů s vazbou na farnosti a uživatele.

Modul pro evidenci fondů obsahuje čtyři tabulky. Tabulka `fnd_funds` obsahuje seznam jednotlivých fondů, do kterých se provádějí platby. Kvůli kompatibilitě s předchozím softwarem a možnosti využití vlastního ID fondu byl přidán atribut `code`. Ten obsahuje identifikaci fondu. Slouží například k identifikaci plateb při importu.

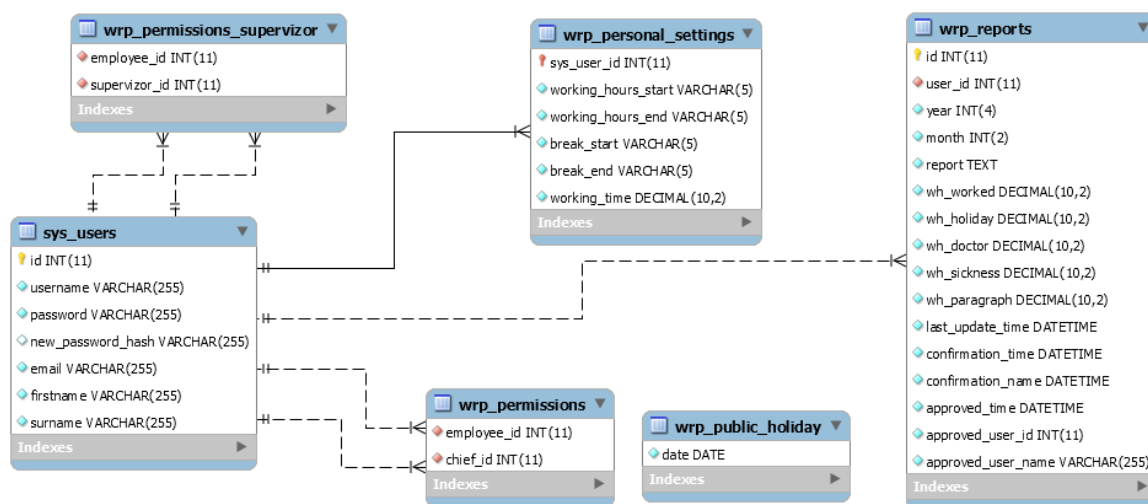
Platby jsou evidovány v tabulce `fnd_payments` s atributy:

- `fund_id` – cizí klíč, slouží k identifikaci fondu.
- `parish_id` – cizí klíč, odkazuje na farnost, která platbu provedla.
- `year` – rok, za který byl příspěvek nahrán. Nestačí evidovat datum vložení, příspěvky mohou být posílány zpětně.

Kvůli integritním omezením do tabulky `fnd_payments` nelze ukládat platby, které nemají jednoznačně přiřazený fond a farnost. Při importu však dochází k situacím, kdy je špatně zadán variabilní či specifický symbol, které slouží k identifikaci fondu a farnosti. Pokud je

platba vyhodnocena jako podezřelá (odpovídá alespoň jeden symbol, platba přišla z účtu farnosti...), je uložena do tabulky `fnl_payments_unprocessed`. Ta má obdobnou strukturu jako jako tabulka `fnl_payments` s tím rozdílem, že neobsahuje integritní omezení.

8.3 Modul výkazů práce



Obr. 14. Schéma modulu výkazů práce s vazbou na uživatele.

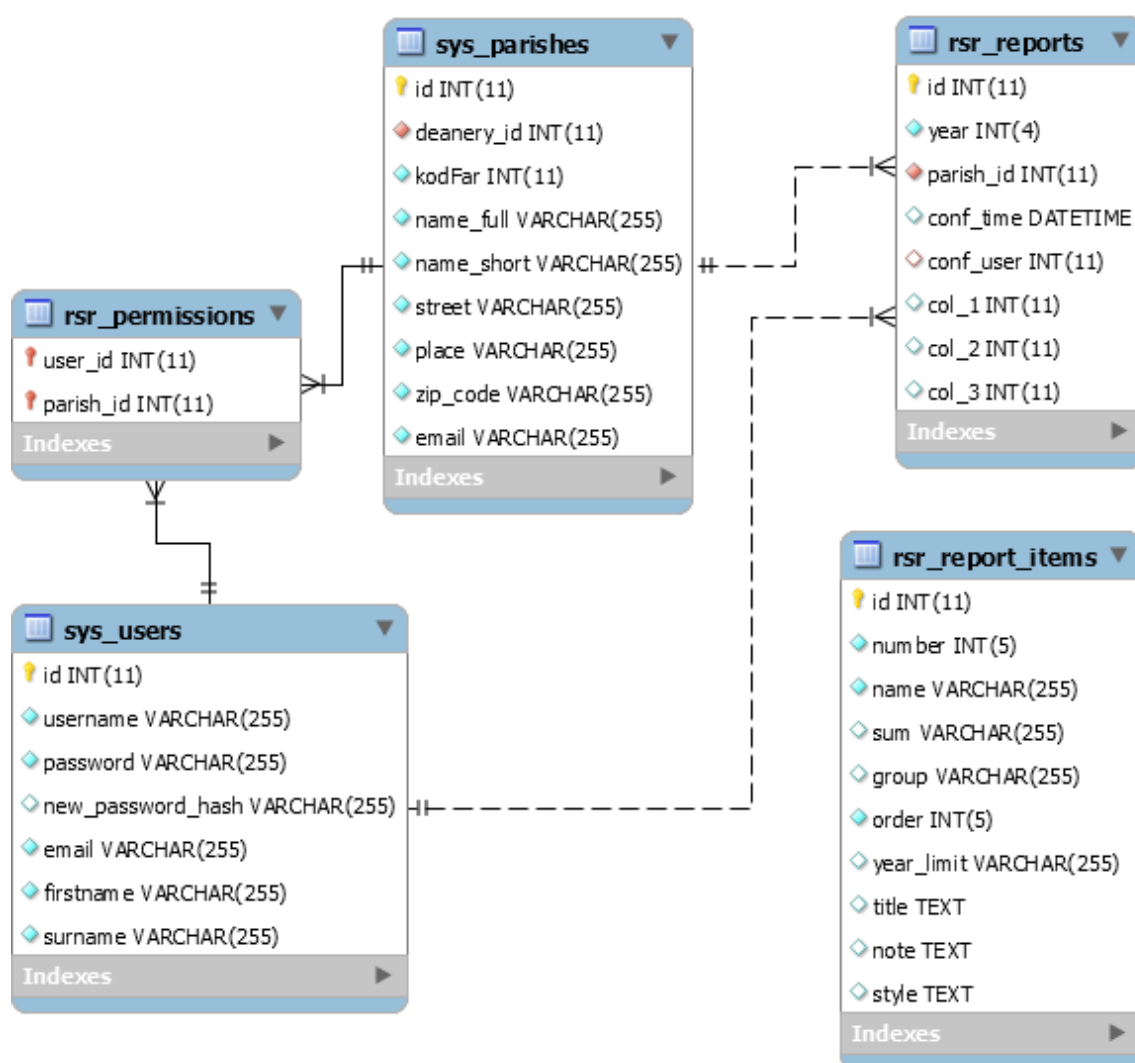
Modul výkazů práce je jediná část systému, která není vázána na farnosti. Výkazy se odevzdávají za jednotlivé zaměstnance. Základní tabulkou je `wrp_reports`, která obsahuje všechny výkazy práce zaměstnanců. Výkazy jsou ukládány za každý měsíc. Popis tabulky:

- `user_id` – cizí klíč, odkazuje do tabulky `sys_users`. Identifikuje uživatele, který vyplňuje výkaz.
- `year`, `month` – rok a měsíc, za který je výkaz vyplněný.
- `report` – ukládá serializované vícerozměrné pole. Obsahuje všechna pole z formuláře výkazu práce.
- `wh_worked`, `wh_holiday`, `wh_doctor`, `wh_sickness`, `wh_paragraph` – tyto atributy slouží k evidenci sald kont pracovní doby (odpracováno, dovolená, lékař, nemoc, paragraf).
- `approved_time`, `approved_user_id`, `approved_user_name` – jsou atributy, které uchovávají informace o vedoucím, který výkaz práce potvrdil.

Tabulka `wrp_public_holiday` obsahuje kalendářní data vyhlášená jako státní svátky. Tabulka `wrp_personal_settings` slouží k uložení pracovní doby a úvazku uživatele pro snadnější vyplňování výkazů.

Dalšími tabulkami jsou `wrp_permissions` a `wrp_permissions_supervisor`. Obě mají stejnou strukturu a obsahují pouze dva cizí klíče. Ty ukazují do tabulky `sys_users`. V tabulkách jsou evidována oprávnění vedoucích uživatelů a supervizorů k jednotlivým pracovníkům.

8.4 Modul hlášení o úkonech v duchovní správě



Obr. 15. Schéma modulu HUDS s vazbou na farnosti a uživatele.

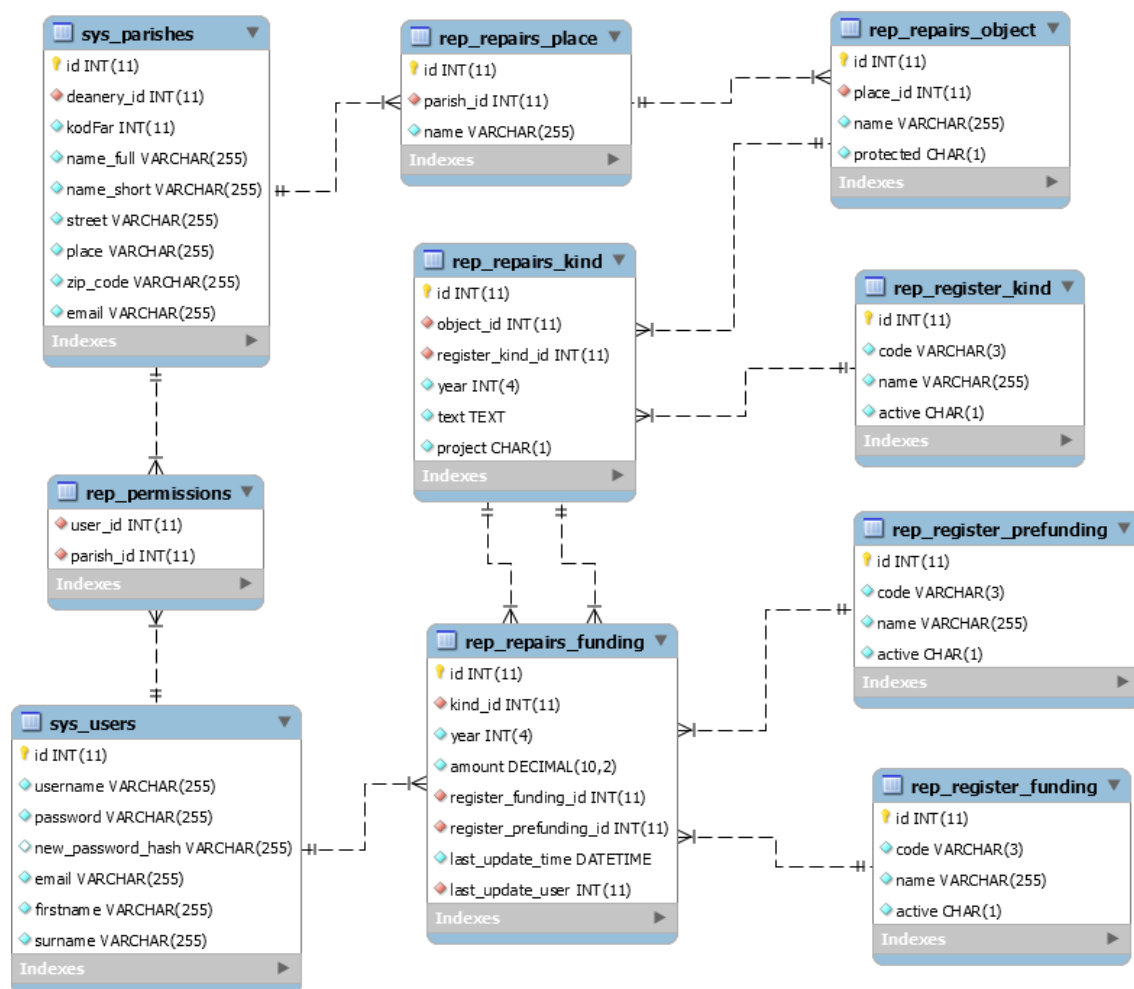
Všechna hlášení jsou uložena v jedné tabulce `rsr_reports`. Tato tabulka má jen několik málo pevných atributů. Jedná se o:

- `id` – automaticky generované číslo, slouží jako primární klíč.
- `parish_id` – cizí klíč, identifikace farnosti, za kterou je výkaz vyplňován.
- `year` – rok, za který je výkaz vyplňován. Každý rok vyplňuje farnost jeden výkaz.
- `ins_user, ins_time` – informace o tom, kdo a kdy výkaz vytvořil.
- `up_user, up_time` – informace o poslední úpravě výkazu.
- `conf_user, conf_time` – informace o potvrzení výkazu.

Další atributy, které tabulka obsahuje, mají tvar `col_xxx`, kde `xxx` je nahrazeno číslem. Tyto atributy označují jednotlivé položky hlášení. Protože je k nim vyžadován obsáhlejší popis, mohou se měnit či přidávat. Veškeré další informace o položkách jsou uloženy v tabulce `rsr_reports_items`. Ta má následující atributy:

- `id` – PK, slouží ke spárování s tabulkou `rsr_reports`.
- `number` – číslo, které se používá k označení ve formuláři.
- `name` – název položky.
- `sum` – některá pole slouží jen jako součet jiných. Pokud tento atribut obsahuje vzorec, je ve formuláři místo pole pro zadání hodnoty vypsán součet.
- `group` – atribut určuje grafické seskupení jednotlivých položek ve vykresleném formuláři.
- `order` – pořadí jednotlivých položek.
- `year_limit` – tento atribut může obsahovat výčet roků. Pokud jsou některé leto-počty uvedeny, zobrazuje se položka ve formuláři jen v uvedených letech.
- `title, note` – doplňující informace k některým položkám.
- `style` – může volitelně obsahovat definici CSS stylů pro přesnější naformátování položky.

8.5 Modul hlášení oprav



Obr. 16. Schéma modulu hlášení oprav s vazbou na farnosti a uživatele.

Hlášení oprav se opět provádí za jednotlivé farnosti minimálně jednou ročně a každý záznam hlášení se skládá ze čtyř částí. Tomu je uzpůsoben i datový model (Obr. 16). Tabulka `rep_repairs_place`:

- `id` – primární klíč.
- `parish_id` – cizí klíč, identifikátor farnosti.
- `name` – název místa (obce), ve kterém oprava probíhá.

Na jednom místě se může opravovat jeden či více objektů. Na místo opravy navazuje tabulka `rep_repairs_object`:

- `id` – primární klíč.
- `repairs_place_id` – cizí klíč, identifikátor místa opravy.

- `name` – název objektu.
- `protected` – označení, zda je objekt památkově chráněn.
- `project` – označení, zda je na opravu objektu zpracováván projekt.

Na každém objektu je možné provádět v jednom roce více oprav. Pro možnost opravy přesně třídit a vyhledávat jsou druhy oprav vybírány z číselníku. Tuto skutečnost zachycuje tabulka `rep_repairs_kind`:

- `id` – primární klíč.
- `repairs_object_id` – cizí klíč, identifikátor objektu, kde probíhá oprava.
- `register_repair_kind_id` – cizí klíč, odkaz do číselníku druhů oprav.
- `text` – možnost přesněji specifikovat opravu.

Poslední tabulkou, která se vyplňuje při zadávání oprav, je tabulka `rep_repairs_funding`:

- `id` – primární klíč.
- `repairs_kind_id` – cizí klíč, identifikátor druhu opravy, ke kterému se financování vztahuje.
- `register_funding_id` – cizí klíč, odkaz do číselníku zdrojů financování.
- `register_prefunding_id` – cizí klíč, odkaz do číselníku zdrojů předfinancování.
- `year` – rok, kdy se oprava provádí.
- `amount` – částka, která bude použita z tohoto zdroje.
- `last_update_time` – čas poslední změny.
- `last_update_user` – cizí klíč, odkazuje na uživatele, který poslední změnu provedl.

Poslední tři tabulky v modulu hlášení oprav jsou číselníky. Jedná se číselník druhů oprav (`rep_register_repair_kind`), zdrojů financování (`rep_register_funding`) a zdrojů předfinancování (`rep_register_prefunding`). Struktura je následující:

- `id` – primární klíč.
- `code` – kód (ID), který používají pracovníci stavebního oddělení.
- `name` – název položky číselníku.
- `active` – označení, zda je položku možno využívat pro zadávání.

9 NÁVRH UŽIVATELSKÉHO ROZHRAŇÍ

Následující kapitola se potýká s návrhem uživatelského rozhraní. Nejprve je samotný pojem definován a dále jsou uvedeny některé nástroje, které je možné k tvorbě uživatelského rozhraní využít. Poslední část této kapitoly se věnuje konkrétnímu návrhu grafického rozhraní pro vyvíjený informační systém.

9.1 Uživatelské rozhraní

„Uživatelské rozhraní je kombinací technologií a prostředků, které umožňují uživateli komunikovat s počítačem“ [18]. Takto lze chápat uživatelské rozhraní v nejširším slova smyslu. Dále se v této práci se bude pod pojmem uživatelské rozhraní myslet výhradně grafické uživatelské rozhraní (GUI¹⁶) v prostředí webové aplikace. Při návrhu aplikace se dnes velmi často využívá metodika UXD.¹⁷ Návrh uživatelského rozhraní je zaměřen primárně na uživatele. Cílem návrhu tedy není jen dobře vypadající aplikace, ale především aplikace, která bude pro uživatele přínosná, funkční, efektivní, bude mít jednoduché a intuitivní ovládání a uživatel se rychle naučí, jak aplikaci používat [18].

Dobře navržené GUI uživatelům umožňuje rychlou orientaci v aplikaci. Neklade na ně nároky, aby nejdříve museli studovat manuál a učit se příkazy jako u textově ovládaných aplikací. Protože dobře navržené uživatelské rozhraní zvyšuje použitelnost aplikace, je zapotřebí věnovat návrhu dostatečnou váhu. Uživatelské rozhraní by mělo být navrhováno shora dolů, tzn. nejdříve je potřeba navrhnout aplikaci jako celek (rozložení, umístění navigace...) a až poté se zabývat jednotlivými detaily (ovládací prvky, formuláře...).

Při návrhu uživatelského rozhraní je vhodné se držet zavedených zvyklostí. Zvyšuje se tím použitelnost programu a schopnost uživatelů se v aplikaci rychle orientovat. Pokud je například na webových stránkách zvykem umísťovat menu nahoru a doleva, není dobré bezdůvodně vymýšlet jiné umístění.

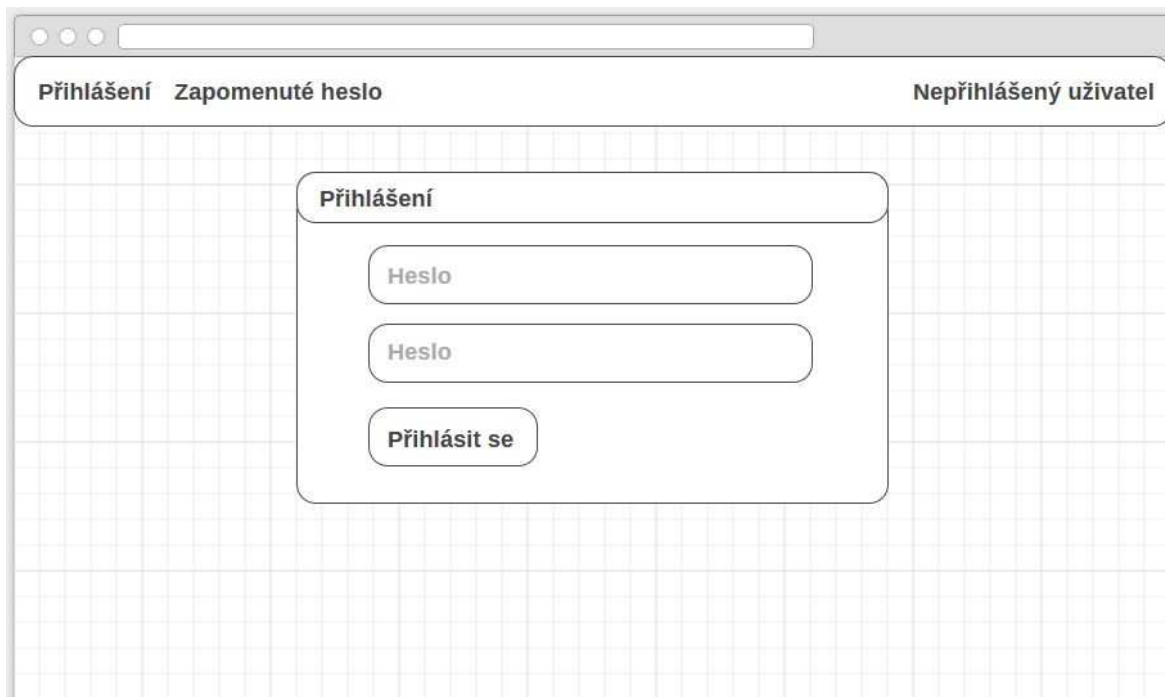
Nespornou výhodou architektury MVC, která je zmíněna v kapitole 3.2, je fakt, že implementace uživatelského rozhraní je naprosto oddělena od ostatních částí aplikace (datového

¹⁶ Graphic user interface

¹⁷ User Experience Design

modelu a prezentační logiky). Z tohoto důvodu je mj. možné naprosto oddělit role kodéra a programátora aplikace.

9.2 Wireframe



Obr. 17. Wireframe přihlašovací stránky.

Velmi užitečným nástrojem pro tvorbu uživatelského rozhraní je wireframe.¹⁸ Ten jednoduchým způsobem ukazuje základní (při podrobnějším návrhu i detailní) rozvržení jak celé aplikace, tak jejích jednotlivých částí. Nespornou výhodou je, že pro tvorbu základního wireframe postačí tužka a papír. Naskýtá se tak možnost rychle navrhnout rozmístění jednotlivých částí webu bez jakékoli potřeby specifikovat konkrétní technologie.

Není samozřejmě nutné vytvářet wireframe na papíře. Dnes existuje spousta softwarových nástrojů, které jej umožňují sestavit v digitální podobě, od úplně základních, které jsou vlastně jen specializovanou verzí kreslicího softwaru¹⁹, až po velmi sofistikované, které umožňují tvorbu HTML prototypů.²⁰ Některé nástroje je možné využít jako online aplikaci.

¹⁸ Do češtiny se často překládá jako „drátěný model“

¹⁹ Například wireframe.cc – dostupný online na adrese <https://wireframe.cc/>

²⁰ HTML prototyp je funkční HTML model, který obsahuje funkční prvky a slouží prezentaci, testování a odhalování chyb. Používá se v zásadě jen u rozsáhlejších projektů.

Tvorba wireframe by měla být jedním z prvních kroků (po slovní či textové definici základní funkcionality) návrhu uživatelského rozhraní. V této fázi projektování se ještě neřeší do podrobností grafická podoba zamýšlené aplikace.

9.3 Návrh uživatelského rozhraní.

Následující text rozebírá návrh konkrétního GUI pro vyvíjený informační systém. Protože systém nabízí naprosté minimum funkcionality před přihlášením uživatele, není potřeba tuto část detailněji rozebírat. Jedná se o dvě stránky s velmi podobnými formuláři. Jeden slouží k přihlášení uživatelů do aplikace (Obr. 17) a druhý k zaslání zapomenutého hesla pomocí e-mailové adresy, kterou má uživatel zadanou v systému.

Po přihlášení do aplikace zůstane zachována horní lišta, která poskytuje navigaci mezi jednotlivými moduly a v pravé části zobrazuje informace o aktuálně přihlášeném uživateli. Také se zobrazí lišta v levé části. Ta slouží jako navigace pro aktuálně vybraný modul.

Grafické uživatelské rozhraní je navrženo s ohledem na obvyklé zvyklosti. Jedná se o firemní informační systém. Z tohoto důvodu systém neobsahuje přehnané množství grafických elementů, které by odváděly pozornost. Systém je spíše kolekcí tabulek a formulářů. Primárním cílem při návrhu GUI byla snaha o minimalizaci počtu kroků při provádění jednotlivých úkonů.

Pro práci v tomto informačním systému je velmi podstatný rok. Například příspěvky do fondů nebo hlášení o úkonech v duchovní správě se sčítají po kalendářních letech. Proto je důležité, aby byl zvolený rok neustále zobrazen a aby byla možnost ho rychle změnit. Na obrázku (Obr. 18) je ukázka obrazovky s přehledem odevzdaných inventurních souborů. Vlevo je zobrazen aktuálně zvolený rok, pod kterým jsou jednotlivé položky navigace. V tomto konkrétním případě se jedná o modul správy uživatelů.

Fondy

Výkazy práce

Opravy

HUDS

Inventury

Uživatelé

Odhlásit

Uživatel: Marek Soldán

Rok: 2016

(změnit)

Správa inventur

Děkanát

Celkový přehled

Přehled odevzdaných souborů

Děkanát	Farnost	Datum inventarizace	Datum poslední aktualizace	Stáhnout soubor
Hranice	Běloutín	Farnost nedodala inventární soubor!		
Konice	Bílá Lhota	15. 5. 2016	15. 5. 2016 11:04:19	↓
Přerov	Beňov	Farnost nedodala inventární soubor!		
Uherské Hradiště	Babice u Uherského Hradiště	2. 5. 2016	2. 5. 2016 21:56:16	↓
Uherské Hradiště	Bílovice	Farnost nedodala inventární soubor!		
Uherský Brod	Bánov	2. 5. 2016	2. 5. 2016 21:54:16	↓

Obr. 18. Ukázka celkového přehledu inventurních souborů.

Na dalším obrázku je ukázka navrženého rozhraní pro modul hlášení oprav. Jedná se o část, ve které se zadávají způsoby financování. V horní části stránky navigace v rámci systému, vlevo menu týkající se zvoleného modulu. Dále je vidět výpis vybraného druhu opravy, objektu, místa a farnosti. Hlavní část zabírá formulář pro zadání financování.

Fondy	Výkazy práce	Opravy	HUDS	Inventury	Uživatelé	Odhlásit	Uživatel: Marek Soldán
-------	--------------	--------	------	-----------	-----------	----------	------------------------

Rok: 2016 (změnit)	<h2>Opravy - druhy oprav</h2>
------------------------------	-------------------------------

Zvolená oprava **statické zajištění, včetně sanace** v roce **2014**
u objektu **Kostel sv. Mikuláše** v místě **Babice u UH**, ve farnosti **Babice u Uherského Hradiště**.

Rok	Financování	Částka	Předfinancování	Poslední aktualizace
-----	-------------	--------	-----------------	----------------------

Předfinancování
Zdroj financování
Rok:
Částka:

Vyberte předfinancování...
Vyberte předfinancování...
1 - půjčka banka
2 - půjčka farnost
3 - půjčka AO
4 - žádné

Uložit

Obr. 19. Ukázka modulu hlášení oprav.

Obdobným způsobem je navržen celý systém. Moduly sice poskytují uživatelům rozdílnou funkcionalitu, ale způsob ovládání je v co největší možné míře jednotný, aby byl co nejlépe srozumitelný a zapamatovatelný.

10 IMPLEMENTACE PROTOTYPU

Tato kapitola se věnuje implementaci základního prototypu aplikace. Nejprve je popsána základní struktura aplikace a dále jsou nastíněny implementační techniky. Systém byl implementován v jazyce PHP ve verzi 5.6.15 za použití Nette frameworku ve verzi 2.3. Jako front-end framework byl zvolen Twitter Bootstrap verze 3.3.6. Jako datové úložiště je použita relační databáze MySQL, verze 5.5.

Implementovaný prototyp není plně funkčním systémem. Je ukázkou, jak bude informační systém vypadat, a nabízí základní ukázkou funkcionality. Slouží k představení zadavateli a uživatelům za účelem vznášení připomínek. Díky této zpětné vazbě lze systém odladit tak, aby co nejlépe odpovídal požadavkům. Tato kapitola neslouží jako exaktní popis celé implementace, ale klade si za cíl nastínit základní principy a logiku celé aplikace.

Systém byl navržen a implementován za využití objektově orientovaného přístupu. Toto programovací paradigma popisuje systém jako soubor vzájemně komunikujících objektů. Objekt je chápán jako samostatná entita, která obsahuje vlastní funkcionalitu a data [3]. Objekty jsou instancemi jednotlivých tříd. Třidu je možné vnímat jako určitý předpis pro vytvoření objektu. Atributy popisují vnitřní stav objektu a metody jsou zvenku viditelné funkce, které mají přístup k vnitřním atributům objektu [1]. Dalšími základními rysy OOP jsou dědičnost, polymorfismus, zapouzdření, abstrakce, skládání.

10.1 Nette framework

Pro implementaci informačního systému byl zvolen Nette framework. Ten nabízí (oproti implementaci čistě v PHP) několik velmi užitečných nástrojů. Za zmínku jistě stojí ladící nástroj Tracy. Na produkčním serveru nám umožňuje logovat chyby a výjimky, na vývojovém serveru jejich přímé, přehledné zobrazení. PHP v případě chyby vypisuje jen strohé informace. Tracy oproti tomu chybové hlášení vizualizuje a vypíše přehledně i s tou částí kódu, kde chyba nastala (Obr. 20). Také umožní zobrazit přímo v prohlížeči další užitečné informace – např. aktuální hodnoty proměnných či informace o konfiguraci serveru.

Parse Error

syntax error, unexpected '}'

Source file ▼

File: ...app\presenters\RepairsPresenter.php:53

```
43:     public function renderDefault()  
44:     {  
45:  
46:     }  
47:  
48:     public function renderPlaces()  
49:     {  
50:         if($this->getUser()->isInRole('Farnost'))  
51:         {  
52:             $myParishes = $this->getUser()->getId()  
53:         }  
54:         else  
55:         {  
56:             $myParishes = $this->repairsRepository->getAvaibleParishes($this->getUser()->getId());  
57:         }
```

Obr. 20. Ukázka výpisu chyby.

Tracy mimo jiné obsahuje tzv. debugger bar. Je to malá lišta, která zobrazuje základní informace o tom, co aplikace zpracovává. Je možnost vidět identitu přihlášeného uživatele, SQL dotazy, které byly položeny, jak dlouho trvalo zpracování konkrétní úlohy a v neposlední řadě máme možnost si do debugger baru nechat vypsát jakékoli proměnné. Na obrázku (Obr. 21) můžeme vidět ukázkou. Zleva jsou jednotlivé údaje o času zpracování, aktuálním presenteru a akci, času a počtu SQL dotazů. Po kliknutí na jednotlivé údaje se můžeme dozvědět další detaily. Například je možnost si prohlédnout konkrétní dotazy, které byly kladeny do databáze či údaje aktuálně přihlášeného uživatele.

TRACY 325.1 ms Repairs:objects 38.0 ms/4

Obr. 21. Debugger bar.

10.2 Adresářová struktura aplikace

Adresářová struktura aplikace vychází z principů Nette. Základní struktura vypadá následovně:

- `app/` – zde se nachází celý zdrojový kód aplikace.
- `log/` – adresář pro zápis logů na produkčním serveru.
- `temp/` – dočasné soubory.
- `vendor/` – adresář pro nahrání frameworku a dalších knihoven.
- `www/` – kořenová složka webové aplikace.

Důležité je, aby byl webový server nakonfigurovaný tak, že jediná přístupná složka z internetu je `www`. V té se nacházejí obrázky, CSS soubory, soubory JavaScriptu atp. Také se zde nalézá soubor `index.php`. Ten volá základní soubor celé aplikace, kterým je `app/bootstrap.php`, který slouží k načtení konfiguračních souborů, knihoven, nastavení vývojového či produkčního módu. Adresář `app/`, ve kterém se nachází samotná aplikace, obsahuje:

- `config/` - konfigurační soubor.
- `forms/` - soubory se třídami formulářů.
- `model/` - třídy modelu.
- `presenters/` - třídy presenterů a šablony.
- `router/` - třídy pro konfiguraci URL adres.
- `bootstrap.php`

V adresáři `config/` se nalézají dva soubory. V souboru `config.neon` se nachází konfigurace aplikace – registrace služeb, rozšíření, připojení k databázi apod. Soubor `config.local.neon` funguje obdobně. Slouží k nastavením, které se liší na serverech, kde je provozován (produkční, vývojový...) – například obsahuje údaje o připojení k databázi.

10.3 Formuláře

Adresář `forms/` obsahuje soubory s definicemi jednotlivých formulářů. Nette poskytuje velmi dobrou podporu pro tvorbu formulářů.

Na obrázku (Obr. 22) je ukázka metody části zdrojového kódu. Konkrétně se jedná o přihlašovací formulář a jeho metodu `create()`. Formulář vytvořený pomocí třídy `Nette\Forms\Form` se automaticky postará o validaci jak na straně klienta, tak na straně serveru. Zároveň zajišťuje automatickou ochranu proti XSS. Všimněme si, že formulářová třída neobsahuje v tomto případě zpracování události po úspěšném odeslání formuláře. V presenteru, kde se tato třída volá, je potřeba přidat metodu, která zajistí zpracování formuláře (Obr. 23). Za zmínku stojí volání metody `setRenderer(new Bs3FormRenderer)`, která má za úkol vykreslení formuláře pro formátování pomocí Twitter Bootstrapu.


```
public function create()
{
    $form = new Form;
    $form->setRenderer(new Bs3FormRenderer);

    $form->addText('username')
        ->setRequired('Prosím vyplňte své uživatelské jméno.')
        ->setAttribute('placeholder', 'E-mail / Uživatelské jméno');

    $form->addPassword('password')
        ->setRequired('Prosím vyplňte své heslo.')
        ->setAttribute('placeholder', 'Heslo');

    $form->addSubmit('send', 'Přihlásit');

    $form->onSuccess[] = function($form) {
        $this->onFormSuccess();
    };
    return $form;
}
```

Obr. 22. Ukázka zdrojového kódu přihlašovacího formuláře.

```
public function signInFormSucceeded($form)
{
    $values = $form->values;

    try {
        $this->getUser()->login($values->username, $values->password);
        $this->redirect('Homepage:');
    } catch (Nette\Security\AuthenticationException $e) {
        $form->addError('Nesprávné přihlašovací jméno nebo heslo.');
```

Obr. 23. Ukázka zdrojového kódu zpracování formuláře.

Tento přístup poskytuje velkou výhodu. Jeden formulář je možné použít na více místech aplikace a pokaždé je možné jej zpracovat jinak. Obdobným způsobem jsou v aplikaci tvořeny všechny další formuláře.

10.4 Model

Další část aplikace se nachází v adresáři `model/`. Zde jsou uloženy soubory, které obsahují definice tříd modelu. Ten zajišťuje veškerou komunikaci s datovým úložištěm. Všechny třídy modelu jsou potomkem třídy `Nette\Object`. První je definována třída `BaseRepository`. Ta obsahuje jedinou metodu – konstruktor, která zajišťuje předání spojení na

databázi. Všechny další třídy modelu jsou potomkem třídy `BaseRepository`. Na obrázku (Obr. 24) je ukázka třídy `ParishesRepository`.

```
public function getName($id, $type = NULL)
{
    $colName = NULL;
    if($type == 'full'){
        $colName = self::TABLE_PARISHES_COLUMN_NAME_FULL;
    }
    else{
        $colName = self::TABLE_PARISHES_COLUMN_NAME_SHORT;
    }
    $parish = $this->database->table(self::TABLE_PARISHES_NAME)
        ->select($colName)
        ->wherePrimary($id)
        ->fetch();
    return $parish->$colName;
}
```

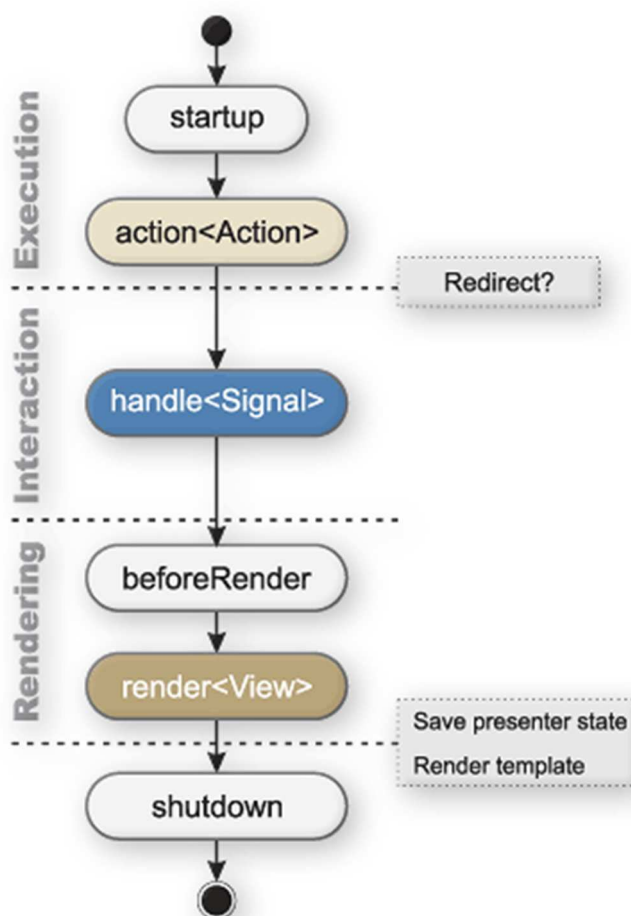
Obr. 24. Ukázka zdrojového kódu části modelu farností.

Konkrétně funkce `getName($id, $type = NULL)` vrací název farnosti na základě primárního klíče. Pro komunikaci s datovým úložištěm slouží třída `Nette\Database`. Ta slouží k připojení k databázi, umožňuje jednoduché vytváření SQL dotazů za pomoci vlastních metod nebo přímého zápisu v jazyce SQL. Třída `Nette\Database` je vlastně nadstavbou nad PDO.²¹

10.5 Presenters

V adresáři `presenters/` jsou umístěny třídy `presenterů` a šablony pro zobrazení obsahu. Presenter řídí činnost celé aplikace. Zpracovává vstupy od uživatele, kontroluje uživatelská oprávnění, komunikuje s modelem, provádí přesměrování a připravuje data pro zobrazení v šabloně. Životní cyklus presenteru v Nette je graficky znázorněn na obrázku (Obr. 25) [21].

²¹ PDO (*PHP Data Objects*) – rozhraní pro práci s SQL databází.



Obr. 25. Životní cyklus presenteru v Nette.

Na obrázku je vidět šest metod, které jsou volány ve specifickém pořadí. Metoda `action<Action>` se využívá v případech, kdy není potřeba nic vykreslovat, například zapsání údajů do databáze. Na základě výsledku je možné rozhodnout o tom, jaká šablona se vykreslí. K tomu slouží metoda `render<View>`, která se stará o sestavení dat a následné předání do šablony. Ta je zobrazena po ukončení životního cyklu.

```

public function beforeRender()
{
    parent::beforeRender();
    $this->template->leftMenuItems = array(
        'Základní přehled' => 'Funds:overview',
        'Nový příspěvek' => 'Funds:Payment',
        'Import z výpisu' => 'Funds:importBank',
        'Přehled fondů' => 'Funds:fundsOverview',
        'Přehled farností' => 'Funds:parishesOverview',
    );
}

```

Obr. 26. Ukázka zdrojového kódu metody `beforeRender()`.

Systém obsahuje základní presenter (BasePresenter), od kterého jsou odvozeny všechny další presentery. Každý presenter v navrhovaném systému obsahuje metodu `beforeRender()`, která je využívána k nadefinování položek menu (Obr. 26). Uvedme ještě příklad metody `action<Action>`. Metoda `actionDownload($invId)` si nejdříve vyžádá od modelu název souboru, poté odešle soubor ke stažení, zaregistruje flash zprávu a nakonec přesměruje uživatele (Obr. 27). Flash zprávy jsou součástí použitého frameworku a slouží k informování uživatelů o činnosti systému. Zůstávají uloženy po určitý časový úsek a jejich vypsaní zpráv má na starosti šablona.

```
public function actionDownload($invId) {  
    $filename = $this->inventoriesRepository->getFilename($invId);  
    $response = new Responses\FileResponse  
        (__DIR__.self::UPLOAD_DIRECTORY.$filename, $filename);  
    $this->sendResponse($response);  
    $this->flashMessage('Soubor byl odeslán ke stažení!', 'alert-info');  
    $this->redirect('Inventories:default');  
}
```

Obr. 27. Ukázka zdrojového kódu metody pro stažení souboru.

10.6 Router

```
public static function createRouter()  
{  
    $router = new RouteList();  
    $router[] = new Route('<presenter>/<action>[/<id>]',  
        'Homepage:default');  
    return $router;  
}
```

Obr. 28. Ukázka zdrojového kódu routeru.

V předchozí kapitole bylo v presenteru použito přesměrování. Konkrétně se provádí pomocí metody `$this->redirect('Inventories:default')`. Jak je vidět, nepřesměrovává se na konkrétní URL. Metoda `redirect` přijímá argument ve formě `Presenter:akce`. Celá aplikace je tedy nezávislá na konkrétních tvarech URL a v případě potřeby je možné jednoduše všechna URL v aplikaci změnit. Nastavení routování se provádí v souborech, které se nacházejí ve složce `router/`. V navrhovaném systému vše obstarává jednoduchá routa (Obr. 28), která jako parametry přijímá dva textové řetězce – masku cesty a výchozí akci. Masku cesty je tvořena dvěma povinnými údaji (`<presenter>` a

`<action>`) a nepovinným `<id>`. Díky tomu, že jsou zadány výchozí hodnoty, není v podstatě nutné zadávat ani první dva parametry. Pokud nebudou zadány, budou automaticky použity hodnoty, které byly vyplněny jako výchozí.

10.7 Latte

Nette Framework nabízí také vlastní šablonovací systém. Ten je poslední částí použitého třívrstvého modelu. Slouží k naformátování výstupu a zobrazení dat uživateli. Nemá přístup k jiným datům, než které mu předá presenter. Všechny výstupy jsou při výpisu automaticky ošetřeny escapováním²² proti XSS. Latte obsahuje makra (podmínky, cykly, definice a výpisy proměnných) pro zpracování dat a filtry (modifikace řetězců, formátování hodnot), kterými lze zobrazení dat ovlivnit.

Na příkladu (Obr. 29) je šablona vykreslující tabulku s příspěvky do fondů. Proměnná `$payments` je do šablony předána presenterem. Jak je vidět, struktura se příliš neliší od obyčejného HTML souboru.

²² „Escapování je převod znaků majících v daném kontextu speciální význam na jiné odpovídající sekvence.“
[25]

```

<table class="table table-bordered table-condensed table-striped">
  <tr>
    <th>Datum vložení</th>
    <th>Fond</th>
    <th>Příspěvatel</th>
    <th>Částka</th>
    <th>Datum platby</th>
  </tr>
  {foreach $payments as $pay}
    <tr>
      <td class="text-right">
        <a n:href="Funds:Payment $pay->id">
          {$pay->insertTime | date:'d. m. Y H:i:s'}
        </a>
      </td>
      <td>{$pay->fundId} - {$pay->fundName}</td>
      <td>{$pay->parishName}</td>
      <td class="text-right">{$pay->amount | number:2:',':' '} Kč</td>
      <td class="text-right">{$pay->paymentDate | date:'d. m. Y'}</td>
    </tr>
  </foreach>
  {if !$payments}
    <tr>
      <td colspan="5">Nejsou k dispozici žádné záznamy</td>
    </tr>
  </if>
</table>

```

Obr. 29. Ukázka zdrojového kódu šablony v Latte.

11 ZHODNOCENÍ ŘEŠENÍ A NÁVRH MOŽNÉHO ROZŠÍŘENÍ

Celý navržený systém je v současné chvíli nasazen pro testovací provoz. Do systému byly importovány údaje o farnostech, děkanátech a zaměstnancích. Vybraným zaměstnancům byla přidělena práva a mohou se do systému přihlásit. Probíhá naplňování systému reálnými daty z jednotlivých agend. To není možno provést automatizovaně, protože data v digitální podobě neexistují. Po získání dostatečného množství dat bude probíhat vyhodnocení a úprava společných výstupů.

Agenda inventur již byla spuštěna v reálném provozu. Pracovníci se k systému postupně přihlašují a odevzdávají aktuální inventurní soupisy. Zde zatím nenastaly žádné problémy, které by nebylo možno operativně vyřešit. Další agendy budou postupně uváděny do provozu v následujících měsících.

Hodnocení uživatelů je veskrze pozitivní. Když pomineme obecnou nedůvěru pracovníků k novým systémům, tak nejvíce vyzdvihují jednoduché a intuitivní ovládání. Velmi dobře je také vnímán systém jako webová aplikace, a tím daná dostupnost z jakéhokoli počítače bez nutnosti instalace. Velký přínos si uvědomují někteří vedoucí pracovníci. Pro ně znamená informační systém velký potenciál do budoucna díky tomu, že zavedl přihlašování všech pracovníků do jednoho systému.

Systém se jeví jako vhodně navržený a všechny požadavky, které vzešly z testování, se zatím daří zdárně zapracovávat. Datový model je dostatečně přizpůsobený požadavkům a zatím do něj nebylo potřeba zasahovat.

11.1 Další možnosti rozšíření systému

Již v těchto fázích zavádění informačního systému se objevují další podněty, co by se dalo v systému zpracovávat. Jednou z možností je právě nahrazení stávajícího modulu inventur. Ten by nesloužil jen jako úložiště souborů, ale jako plnohodnotný inventarizační modul. Tomu by ale musela předcházet podrobná analýza.

Dalším modulem, který by systém mohl svou strukturou pokrýt, je zadávání výsledků ze sčítání účastníků bohoslužeb. Zde se jedná o zpracování zhruba tří set tisíc údajů. Sčítání probíhá v jednotlivých kostelích v celé diecézi.

Bude-li úspěšně dokončeno nasazení systému a systém bude dále provozován, bude zapotřebí také optimalizovat uživatelské rozhraní tak, aby odpovídalo principům responzivního webu.

Z hlediska IT oddělení se začíná diskutovat o dvou možných úpravách. Jednou z nich je nasazení komplexní ORM vrstvy, která by usnadnila práci s databází. V současné chvíli je v podstatě veškerá komunikace s databází zajišťována přímo SQL dotazy. Druhou je implementace modulu, který by umožnil komplexnější správu rolí, oprávnění a zdrojů. Pokud bude systém dále rozšiřován, budou obě úpravy nevyhnutelné.

ZÁVĚR

Cílem práce bylo navrhnout informační systém použitelný pro arcibiskupství, který by pokrýval specifické potřeby vyskytující se v církevním prostředí.

V teoretické části práce byly nejprve popsány systémy, které již na arcibiskupství fungují. Dále byla provedena analýza některých činností, které jsou na arcibiskupství pravidelně vykonávány a které by mohl nový informační systém zefektivnit. Cílem bylo zejména zjednodušit výměnu informací mezi odděleními. V diplomové práci bylo vybráno pět agend, jejichž zpracování by měl zamýšlený systém pokrýt. Konkrétně se jedná o evidenci fondů, výkazy práce, hlášení oprav, hlášení o úkonech v duchovní správě a evidenci inventur.

Po zvážení jednotlivých možností bylo rozhodnuto, že informační systém bude navržen a provozován jako webová aplikace a tomu byl podřízen i výběr implementační technologie. Byla zvolena velmi populární kombinace - skriptovací jazyk PHP (s využitím Nette frameworku) a MySQL jako datové úložiště. Na základě požadavků jednotlivých oddělení byly definovány role, pod kterými bude možné do systému přistupovat. Zároveň byly specifikovány jednotlivé případy užití tak, aby pokrývaly všechny požadavky.

Za hlavní přínos práce je možné považovat samotné zavedení systému i přesto, že je systém zatím v testovacím provozu. Uživatelé velmi oceňují možnost vyřizovat agendy elektronicky, bez nutnosti zpracovávat data ručně na papíře. Protože ze strany vedoucích pracovníků je patrná ochota na tento systém kompletně přejít, dá se předpokládat, že po odladění některých detailů bude zavádění systému úspěšně dokončeno a bude pokračovat jeho rozvoj.

Z hlediska managementu organizace je velkým přínosem, že se všichni externí zaměstnanci přihlašují do jednotného systému. Taková možnost zatím na arcibiskupství nebyla a externí zaměstnanci museli komunikovat výhradně pomocí e-mailové či klasické korespondence. Zavedení navrženého informačního systému se jeví jako velmi dobrá výchozí pozice k případným budoucím rozšíření o další funkcionality.

SEZNAM POUŽITÉ LITERATURY

- [1] VRANA, Ivan. *Projektování informačních systémů s UML*. Vyd. 1. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2008, 147 s. ISBN 978-80-213-1817-5.
- [2] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. Brno: Computer Press, 2007, 567 s. ISBN 978-802-5115-039.
- [3] PECINOVSKÝ, Rudolf. *OOP – learn object oriented thinking and programming*. Řepín: Tomáš Bruckner, 2013, xxiv, 502 s. Academic series. ISBN 978-80-904661-8-0.
- [4] LAVIN, Peter. *PHP - objektově orientované: koncepty, techniky a kód*. 1. vyd. Praha: Grada, 2009, 211 s. Průvodce (Grada). ISBN 978-80-247-2137-8.
- [5] ROBINSON, Simon. *C#: programujeme profesionálně*. Brno: Computer Press, 2003, xxx, 1130 s. Programmer to programmer. ISBN 80-251-0085-5.
- [6] CHLAPEK, Dušan, Václav ŘEPA a Iva STANOVSKÁ. *Analýza a návrh informačních systémů*. Praha: Oeconomica, 2011. ISBN 978-80-245-1782-7.
- [7] VLASÁK, Rudolf a Soňa BULÍČKOVÁ. *Základy projektování informačních systémů*. Praha: Karolinum, 2003. ISBN 80-246-0727-1.
- [8] TVRDÍKOVÁ, Milena. *Zavádění a inovace informačních systémů ve firmách*. 1.vyd. Praha: Grada Publishing, 2000, 110 s. ISBN 80-716-9703-6.
- [9] TIETZE, Petr. *Strukturální analýza: úvod do projektu řízení*. Praha: Grada, 1992, 224 s. Nestůjte za dveřmi. ISBN 8085424452.
- [10] *MySQL* [online]. [cit. 2016-05-04]. Dostupné z: <https://www.mysql.com/>
- [11] BOREK, Bernard. Úvod do architektury MVC. In: *Zdroják.cz* [online]. 2009 [cit. 2015-04-12]. Dostupné z: <http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/>
- [12] *Bootstrap - The world's most popular mobile-first and responsive front-end framework*. [online]. [cit. 2016-05-04]. Dostupné z: <http://getbootstrap.com/>
- [13] HRONEK, Jiří. *Informační systémy* [online]. Olomouc, 2007 [cit. 2015-04-25]. Dostupné z: <http://phoenix.inf.upol.cz/esf/ucebni/infoSys.pdf>. Učební text. Univerzita Palackého.

- [14] *Kodex kanonického práva: Úřední znění textu a překlad do češtiny : Latinsko-české vyd. s věc. rejstř.* Praha: Zvon, 1994. ISBN 80-7113-082-6.
- [15] MAJDA, David. Knihovny vs. frameworky. In: *David Majda* [online]. 2009 [cit. 2016-04-10]. Dostupné z: <http://majda.cz/blog/265>
- [16] MONUS, Anna a Jan POKORNÝ. 10 nejlepších PHP frameworků pro vývojáře. In: *Interval.cz | Svět Internetu, Technologií a Bezpečnosti* [online]. 2015 [cit. 2016-04-10]. Dostupné z: <https://www.interval.cz/clanky/10-nejlepsich-php-frameworku-pro-vyvojare/>
- [17] SKVORC, Bruno. *The Best PHP Framework for 2015: SitePointSurveyResults*. In: *SitePoint – Learn HTML, CSS, JavaScript, PHP, Ruby & Responsive Design* [online]. 2015 [cit. 2016-04-10]. Dostupné z: <http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>
- [18] *Návrh uživatelského rozhraní webové aplikace. Grafická a multimediální laboratoř VŠE* [online]. 2000-2016 [cit. 2016-04-11]. Dostupné z: <http://gml.vse.cz/data/oppa-webdesign/ui.html>
- [19] ŠIMONOVÁ, Stanislava, Renáta MYŠKOVÁ a Pavel JIRAVA. *Projektování informačních systémů - UML, procesní řízení: pro kombinovanou formu studia*. Vyd. 1. Pardubice: Univerzita Pardubice, 2006. ISBN 8071948950.
- [20] BUCHALCEVOVÁ, Alena. *Metodiky vývoje a údržby informačních systémů: kategorizace, agilní metodiky, vzory pro návrh metodiky*. 1. vyd. Praha: Grada, 2005. Management v informační společnosti. ISBN 8024710757.
- [21] *Rychlý a pohodlný vývoj webových aplikací v PHP | Nette Framework* [online]. NetteFoundation, 2016 [cit. 2016-04-13]. Dostupné z: <https://nette.org/>
- [22] *W3C HTML* [online]. [cit. 2016-05-04]. Dostupné z: <https://www.w3.org/html/>
- [23] *Cascading Style Sheets* [online]. [cit. 2016-05-04]. Dostupné z: <https://www.w3.org/Style/CSS/>
- [24] *PHP: Hypertext Preprocessor* [online]. [cit. 2016-05-04]. Dostupné z: <http://php.net/>
- [25] GRUDL, David. Escapování - definitivní příručka. In: *PhpFashion* [online]. 2009 [cit. 2016-05-01]. Dostupné z: <https://phpfashion.com/escapovani-definitivni-pri-rucka>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AD	Active Directory
ASP	Active Server Pages
CASE	Computer-aided Software Engineering
CSRF	Cross-site Request Forgery
CSS	Cascading Style Sheets
CSV	Comma-separated Values
EDI	Electronic Data Interchange
ERP	Enterprise Resource Planning
EUP	Enterprise Unified Process
FDD	Feature Driven Development
GPL	General Public License
GUI	Graphical User Interface
HTML	HyperText Markup Language
HUDS	Hlášení o úkonech v duchovní správě
ICT	Information and Communication Technologies
IS	Information System
LAN	Local Area Network
MVC	Model View Controller
OIS	Office Information System
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
OMG	Object Management Group
OOP	Object-oriented Programming
ORM	Object Relational Mapping

PDF	Portable Document Format
PDO	PHP Data Objects
PHP	PHP Hypertext Preprocessor
RUP	Rational Unified Process
SOA	Service Oriented Architecture
SQL	Structured Query Language
UC	Use Case
UML	Unified Modeling Language
URL	Uniform Resource Locator
UXD	User Experience Design
XP	Extreme Programming
XSS	Cross-site Scripting

SEZNAM OBRÁZKŮ

Obr. 1. Informatický pohled na IS.	17
Obr. 2. Rozdělení informačních systémů.....	18
Obr. 3. Schéma MVC architektury.	19
Obr. 4. Přehled aktérů systému.	36
Obr. 5. UC diagram systémové části.	40
Obr. 6. UC diagram modulu inventury.	40
Obr. 7. UC diagram modulu fondy.	41
Obr. 8. UC diagram modulu výkazy práce.	42
Obr. 9. UC diagram modulu hlášení úkonů v duchovní správě.....	42
Obr. 10. UC diagram modulu hlášení oprav.	43
Obr. 11. Schéma systémové části databáze.	44
Obr. 12. Schéma modulu inventur s vazbou na farnosti a uživatele.....	45
Obr. 13. Schéma modulu fondů s vazbou na farnosti a uživatele.....	46
Obr. 14. Schéma modulu výkazů práce s vazbou na uživatele.....	47
Obr. 15. Schéma modulu HUDS s vazbou na farnosti a uživatele.	48
Obr. 16. Schéma modulu hlášení oprav s vazbou na farnosti a uživatele.....	50
Obr. 17. Wireframe přihlašovací stránky.....	53
Obr. 18. Ukázka celkového přehledu inventurních souborů.	55
Obr. 19. Ukázka modulu hlášení oprav.	55
Obr. 20. Ukázka výpisu chyby.....	57
Obr. 21. Debugger bar.	57
Obr. 22. Ukázka zdrojového kódu přihlašovacího formuláře.....	59
Obr. 23. Ukázka zdrojového kódu zpracování formuláře.....	59
Obr. 24. Ukázka zdrojového kódu části modelu farností.	60
Obr. 25. Životní cyklus presenteru v Nette.....	61
Obr. 26. Ukázka zdrojového kódu metody beforeRender().....	61
Obr. 27. Ukázka zdrojového kódu metody pro stažení souboru.....	62
Obr. 28. Ukázka zdrojového kódu routeru.	62
Obr. 29. Ukázka zdrojového kódu šablony v Latte.	64

SEZNAM TABULEK

Tab. 1. Porovnání rigorózních a agilních metodik.....	21
Tab. 2. Tabulka funkčních požadavků.....	30
Tab. 3. Tabulka nefunkčních požadavků.....	33

SEZNAM PŘÍLOH

- P I CD-ROM se zdrojovými kódy a textem práce
- P II Scénáře případů užití
- P III Matice sledovatelnosti požadavků
- P IV Model tříd
- P V Schéma databáze

PŘÍLOHA P I: CD SE ZDROJOVÝMI KÓDY A TEXTEM PRÁCE

Struktura souborů na přiloženém CD:

- \DOC\ - text diplomové práce
- \SQL\ - skripty pro vytvoření databáze
- \SRC\ - zdrojové kódy navrženého systému
- readme.txt

PŘÍLOHA P II: SCÉNÁŘE PŘÍPADŮ UŽITÍ

Případ užití: Přihlásit se
ID: UC1
Účastníci: Nepřihlášený uživatel
Vstupní podmínky: 1. Nepřihlášený uživatel přejde na stránku pro přihlášení.
Tok událostí: 1. Systém zobrazí formulář pro přihlášení. 2. Uživatel zadá e-mail a heslo. 3. Systém ověří, zda je e-mail zadáný v databázi a jestli se shoduje hash hesla. 4. Pokud údaje souhlasí, uživatel bude přihlášen do systému.
Následné podmínky: 1. Uživatel byl přihlášen k systému.
Alternativní tok: 3.1 Pokud údaje nesouhlasí, systém vypíše chybové hlášení. 3.2 Systém zobrazí formulář pro přihlášení.

Případ užití: Odhlásit
ID: UC2
Účastníci: Přihlášený uživatel
Vstupní podmínky: 1. Uživatel je přihlášený v systému.
Tok událostí: 1. Uživatel zvolí v menu možnost odhlásit se. 2. Systém odhlásí přihlášeného uživatele. 3. Systém zobrazí formulář pro přihlášení.
Následné podmínky: 1. Uživatel je odhlášen ze systému.
Alternativní tok:

Případ užití: Resetovat zapomenuté heslo
ID: UC3
Účastníci: Nepřihlášený uživatel
Vstupní podmínky: 1. Uživatel přejde na stránku pro zaslání nového hesla.
Tok událostí: 1. Systém zobrazí formulář pro zaslání nového hesla. 2. Uživatel zadá e-mailovou adresu. 3. Systém vyhledá e-mailovou adresu v databázi. 4. Pokud e-mailová adresa existuje, systém uloží do databáze hash pro nastavení nového hesla. 5. Systém zašle e-mail s odkazem pro nastavení nového hesla.
Následné podmínky: 1. Uživateli byl odeslán e-mail s možností nastavení nového hesla.
Alternativní tok: 4.1 Pokud e-mailová adresa není uložena v databázi, systém zobrazí chybové hlášení.

Případ užití: Změnit údaje
ID: UC4
Účastníci: Přihlášený uživatel
Vstupní podmínky: 1. Uživatel přejde na stránku pro změnu údajů.
Tok událostí: 1. Systém vyhledá v databázi údaje přihlášeného uživatele podle ID. 2. Systém zobrazí formulář pro změnu údajů s uloženými údaji. 3. Uživatel může upravit jméno, přímení, e-mail, heslo. 4. Uživatel stiskne potvrzovací tlačítko. 5. Systém uloží údaje z formuláře do databáze.
Následné podmínky: 1. Údaje o uživateli byly změněny.
Alternativní tok: 5.1 Pokud uživatel zadal e-mail, který je již v systému registrován, systém vypíše chybové hlášení. 5.2 Systém znova zobrazí formulář pro změnu údajů.

Případ užití: Smazat uživatele
ID: UC5
Účastníci: Administrátor
Vstupní podmínky: 1. Uživatel je přihlášený do systému s rolí administrátora.
Tok událostí: 1. Uživatel přejde na stránku se seznamem uživatelů. 2. Uživatel zvolí uživatele pro smazání. 3. Systém ověří, zda je možné uživatele smazat. 4. Systém odstraní uživatele z databáze.
Následné podmínky: 1. Smazaný uživatel se nemůže přihlásit do systému.
Alternativní tok: 4.1 Pokud uživatele není možné smazat, uloží systém do databáze údaje, že uživatel byl smazaný.

Případ užití: Přiřadit role
ID: UC6
Účastníci: Administrátor
Vstupní podmínky: 1. Uživatel je přihlášený do systému s rolí administrátora. 2. Uživatel přejde na stránku se seznamem uživatelů.
Tok událostí: 1. Administrátor zvolí možnost změnit role u požadovaného uživatele. 2. Systém zobrazí dostupné role s vyznačením již přiřazených rolí uživatele. 3. Administrátor označí požadované role a odešle formuláře. 4. Systém uloží požadované změny do databáze. 5. Systém přesměruje administrátora zpět na přehled uživatelů.
Následné podmínky: 1. Uživatel má nastavené nové role.
Alternativní tok:

Případ užití: Vložit uživatele
ID: UC7
Účastníci: Administrátor
Vstupní podmínky: 1. K systému je přihlášený uživatel s oprávněním administrátora.
Tok událostí: 1. Administrátor zadá příkaz vložit uživatele 2. Systém zobrazí formulář pro vložení uživatele 3. Administrátor vyplní údaje nového uživatele a odešle formulář. 4. Systém zkontroluje, zda jsou zadané povinné údaje. 5. Systém zkontroluje, zda uživatel se stejným e-mailem již v systému není. 6. Systém vloží nového uživatele do systému a odešle notifikační e-mail.
Následné podmínky: 1. Uživatel byl vložen do systému.
Alternativní tok 1: 5.1 Pokud nebyly zadány všechny požadované údaje, systém vypíše chybové hlášení. 5.2 Systém zobrazí formulář pro vložení uživatele.
Alternativní tok 2: 6.1 Pokud je již e-mail v systému registrován, systém vypíše chybové hlášení. 6.2 Systém zobrazí formulář pro vložení uživatele.

Případ užití: Delegovat oprávnění
ID: UC8
Účastníci: Farnost
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Farnost.
Tok událostí: 1. Farnost zvolí možnost delegovat oprávnění. 2. Systém zobrazí formulář pro delegování oprávnění. 3. Farnost vyplní e-mail uživatele, který má být delegován. Farnost zvolí role, ke kterým bude uživatel oprávněn. 4. Systém uloží oprávnění uživatele do databáze.
Následné podmínky: 1. Oprávnění bylo uživateli uděleno.
Alternativní tok: 4.1 Pokud e-mail není v systému registrován, systém vypíše chybové hlášení. 4.2 Systém zobrazí formulář pro delegování uživatele.

Případ užití: Vložit soubor
ID: UC9
Účastníci: Pracovník inventur
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Pracovník inventur.
Tok událostí: 1. Pracovník inventur zvolí možnost správa inventur. 2. Systém zobrazí seznam dostupných farností, ke kterým má Správce inventur oprávnění. 3. Pracovník inventur zvolí požadovanou farnost. 4. Systém zobrazí přehled odevzdaných inventur a formulář pro vložení nového souboru. 5. Pracovník inventur vybere soubor pro vložení a doplní další údaje. 6. Systém zkontroluje, zda vybraný soubor má požadovaný formát. 7. Systém nahraje soubor do datového úložiště a uloží údaje o souboru.
Následné podmínky: 1. Inventurní soubor je uložen v systému.
Alternativní tok 1: 2.1 Pokud uživatel nemá udělen přístup k žádné farnosti, systém vypíše chybové hlášení.
Alternativní tok 2: 7.1 Pokud soubor nemá požadovaný formát, systém neumožní nahrát soubor a vypíše chybové hlášení.

Případ užití: Zobrazit vložené soubory
ID: UC10
Účastníci: Pracovník inventur
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Pracovník inventur.
Tok událostí: 1. Pracovník inventur zvolí možnost prohlížet inventurní soupisy. 2. Systém zobrazí seznam dostupných farností, ke kterým má Správce inventur oprávnění. 3. Pracovník inventur zvolí požadovanou farnost. 4. Systém vypíše seznam odevzdaných inventurních soupisů. 5. Pracovník inventur má možnost stažení jednotlivých souborů s inventurami.
Následné podmínky:
Alternativní tok: 2.1 Pokud uživatel nemá udělen přístup k žádné farnosti, systém vypíše chybové hlášení.

Případ užití: Zobrazit všechny soubory
ID: UC11
Účastníci: Správce inventur
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Správce inventur.
Tok událostí: 1. Správce inventur zvolí možnost prohlížení inventur. 2. Systém vypíše seznam všech farností s informací, kdy ve zvoleném roce byl odevzdán soubor s inventurami.
Následné podmínky:
Alternativní tok:

Případ užití: Stáhnout konkrétní soubor
ID: UC12
Účastníci: Správce inventur
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Správce inventur.
Tok událostí: 1. Správce inventur zvolí možnost prohlížení inventur. 2. Systém vypíše seznam všech farností. 3. Správce inventur vybere farnost. 4. Systém zobrazí celou historii odevzdaných inventurních souborů. 5. Správce inventur má možnost stáhnout kterýkoli odevzdaný soubor.
Následné podmínky:
Alternativní tok:

Případ užití: Zaslát upozornění
ID: UC13
Účastníci: Správce inventur
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Správce inventur.
Tok událostí: 1. Správce inventur vybere možnost zaslání upozornění. 2. Systém zobrazí formulář pro zaslání upozornění. 3. Správce inventur zadá datum, kdy naposledy měla proběhnout aktualizace inventurních souborů. 4. Systém vyhledá farnosti, které stanovenou podmínku nesplňují. 5. Systém zobrazí seznam farností a formulář pro zadání doplňujícího textu. 6. Správce inventur doplní text a potvrdí odeslání e-mailu. 7. Systém rozešle e-maily zvoleným farnostem.
Následné podmínky:
Alternativní tok: 5.1 Pokud všechny farnosti splňují zadanou podmínku, systém informuje správce inventur. 5.2 Systém zobrazí formulář pro zaslání upozornění.

Případ užití: Přidat fond
ID: UC14
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost spravovat fondy. 2. Systém vypíše přehled fondů a zobrazí formulář pro zadání nového fondu. 3. Pokladní zadá požadované údaje do formuláře a potvrdí odeslání. 4. Systém ověří, zda byly zadány všechny povinné údaje. 5. Systém vloží fond do systému.
Následné podmínky: 1. Nový fond byl vložen do systému.
Alternativní tok: 5.1 Pokud nebyly zadány všechny povinné údaje, systém informuje Pokladní vypsáním chybové hlášky.

Případ užití: Změnit fond
ID: UC15
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost spravovat fondy. 2. Systém vypíše přehled fondů a zobrazí formulář pro zadání nového fondu. 3. Pokladní zvolí fond, který si přeje změnit. 4. Systém zobrazí editační formulář s údaji o požadovaném fondu. 5. Pokladní změní požadované údaje a potvrdí odeslání formuláře. 6. Systém uloží informace do databáze.
Následné podmínky: 1. Informace o fondu byly aktualizovány.
Alternativní tok: 6.1 Pokud nebyly vyplněny požadované údaje, systém vypíše chybové hlášení. 6.2 Systém znovu zobrazí editační formulář.

Případ užití: Odebrat fond
ID: UC16
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost správa fondů. 2. Systém vypíše seznam dostupných fondů. 3. Pokladní vybere fond, který chce odebrat. 4. Systém zkontroluje, zda se v daném fondu nachází příspěvky. 5. Pokud fond neobsahuje příspěvky, systém vymaže fond.
Následné podmínky: 1. Fond byl vymazán ze systému.
Alternativní tok: 5.1 Pokud fond obsahuje příspěvky, fond nelze vymazat a systém zobrazí chybové hlášení. 5.2 Systém vypíše seznam dostupných fondů.

Případ užití: Zadat platbu
ID: UC17
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost zadat platbu. 2. Systém zobrazí formulář pro zadání platby. 3. Pokladní vyplní údaje o platbě a odešle formulář. 4. Systém ověří, zda byly požadované údaje zadány správně. 5. Systém uloží platbu.
Následné podmínky: 1. Platba byla vložena do systému.
Alternativní tok: 5.1 Pokud nebyly údaje zadány správně, systém vypíše chybové hlášení. 5.2 Systém zobrazí formulář pro zadání platby.

Případ užití: Odebrat platbu
ID: UC18
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost zobrazit platby. 2. Systém zobrazí seznam posledních plateb. 3. Pokladní vybere platbu pro odstranění. 4. Systém zobrazí potvrzovací dialog. 5. Pokladní potvrdí odstranění platby. 6. Systém vymaže platbu ze systému a zobrazí seznam posledních plateb.
Následné podmínky: 1. Platba byla vymazána ze systému.
Alternativní tok: 5.1 Pokud pokladní nepotvrdí smazání platby, systém odstranění neprovede. 5.2 Systém zobrazí seznam posledních plateb.

Případ užití: Importovat platby z banky
ID: UC19
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost importu z banky. 2. Systém zobrazí formulář pro výběr souboru. 3. Pokladní vybere soubor s výpisem z banky. 4. Systém ověří, zda soubor má správný formát. 5. Systém provede import nalezených plateb. 6. Systém zobrazí platby, které byly identifikovány jako možné příspěvky, které ale nebylo možné přesně přiřadit.
Následné podmínky: 1. Identifikované platby byly uloženy do systému.
Alternativní tok: 5.1 Pokud soubor nemá správný formát, systém vypíše chybové hlášení a zobrazí formulář pro výběr souboru.

Případ užití: Exportovat platby
ID: UC20
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost exportovat platby. 2. Systém nabídne formulář pro přípravu exportu. 3. Pokladní zvolí období, fondy a děkanáty, za které chce export provést a odešle formulář. 4. Systém zpracuje požadovaný export a nabídne soubor pro uložení.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit poslední platby
ID: UC21
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí akci zobrazit poslední platby. 2. Systém zobrazí seznam plateb podle poslední změny u každé platby.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit platby podle fondů
ID: UC22
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel je přihlášený do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost zobrazení přehledu podle fondů. 2. Systém vypíše všechny dostupné fondy. 3. Pokladní zvolí fond pro zobrazení přehledu. 4. Pokladní zadá období, pro které si přeje přehled zpracovat. 5. Systém vypíše jednotlivé platby do fondů v zadaném období. 6. Systém vypíše sumář plateb za zvolené období.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit platby podle příspěvatele
ID: UC23
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost zobrazení přehledu podle farností. 2. Systém vypíše všechny dostupné farnosti. 3. Pokladní zvolí farnosti pro zobrazení přehledu. 4. Pokladní zadá období, pro které si přeje přehled zpracovat. 5. Systém vypíše jednotlivé platby od farnosti v zadaném období. 6. Systém vypíše sumář plateb za zvolené období.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit podezřelé platby
ID: UC24
Účastníci: Pokladní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pokladní.
Tok událostí: 1. Pokladní zvolí možnost import z výpisu. 2. Systém zobrazí formulář pro import a všechny podezřelé platby. 3. Pokladní zvolí možnost upravit platbu 4. Systém zobrazí formulář pro editaci platby. 5. Pokladní doplní chybějící údaje a zvolí uložit.
Následné podmínky: 1. Platba byla přesunuta do tabulky plateb.
Alternativní tok: 3.1 Pokladní zvolí možnost odstranit. 3.2 Systém vymaže podezřelou platbu z databáze. 3.3 Pokračuje se hlavním scénářem od bodu 2.

Případ užití: Zobrazit vlastní příspěvky
ID: UC25
Účastníci: Farnost
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Farnost.
Tok událostí: 1. Farnost zvolí možnost zobrazit vlastní příspěvky. 2. Systém zobrazí sumář za jednotlivé fondy, do kterých farnost provedla platby. 3. Farnost zvolí konkrétní fond. 4. Systém vypíše jednotlivé platby do zvoleného fondu.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit příspěvky za celý děkanát
ID: UC26
Účastníci: Děkanát
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Děkanát.
Tok událostí: 1. Děkanát zvolí možnost zobrazit příspěvky za děkanát. 2. Systém zobrazí sumář za jednotlivé fondy, do kterých farnosti provedly platby. 3. Děkanát zvolí konkrétní fond. 4. Systém vypíše jednotlivé platby za všechny odpovídající farnosti do zvoleného fondu.
Následné podmínky:
Alternativní tok:

Případ užití: Vytvořit výkaz práce
ID: UC27
Účastníci: Zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Zaměstnanec.
Tok událostí: 1. Zaměstnanec zvolí možnost vytvoření nového výkazu práce. 2. Systém zobrazí možnost volby měsíce. 3. Zaměstnanec zvolí měsíc a potvrdí volbu. 4. Systém vloží prázdný výkaz práce do systému. 5. Systém zobrazí všechny dostupné výkazy práce.
Následné podmínky: 1. Výkaz práce byl vložený do systému.
Alternativní tok: 4.1 Pokud již výkaz pro zvolený měsíc existuje, systém vypíše chybové hlášení.

Případ užití: Vyplnit výkaz práce
ID: UC28
Účastníci: Zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Zaměstnanec.
Tok událostí: 1. Zaměstnanec zvolí možnost upravit pracovní list z nabídky pracovních listů. 2. Systém zobrazí formulář pro konkrétní měsíc. 3. Zaměstnanec vyplní požadované údaje a odešle formulář. 4. Systém provede validaci dat a uloží změny.
Následné podmínky: 1. V systému byl aktualizován pracovní list.
Alternativní tok: 4.1 Pokud dojde k chybě při validaci údajů, systém zobrazí chybové hlášení. 4.2. Systém znova zobrazí formulář pracovního listu.

Případ užití: Potvrdit výkaz práce
ID: UC29
Účastníci: Zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Zaměstnanec.
Tok událostí: 1. Zaměstnanec zvolí možnost zobrazit přehled výkazů práce. 2. Systém zobrazí seznam vložených výkazů práce. 3. Zaměstnanec zvolí možnost potvrdit výkaz práce. 4. Systém zkontroluje, zda je výkaz práce řádně vyplněn. 5. Systém potvrdí výkaz práce.
Následné podmínky: 1. Výkaz práce byl potvrzen.
Alternativní tok: 4.1 Pokud není výkaz správně vyplněn, zobrazí systém chybové hlášení. 4.2 Systém zobrazí výkaz práce. 4.3 Zaměstnanec provede změny ve formuláři a odešle. 4.4 Tok pokračuje na 4. bod základního toku.

Případ užití: Zobrazit základní přehled
ID: UC30
Účastníci: Zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Zaměstnanec.
Tok událostí: 1. Zaměstnanec zvolí možnost základní přehled. 2. Systém zobrazí přehled výkazů práce s informacemi o poslední úpravě, datu potvrzení a datu schválení a nabídkou možných operací. 3. Systém zobrazí zbývající dny dovolené pro daný rok.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit výkaz práce
ID: UC31
Účastníci: Vedoucí zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Vedoucí zaměstnanec.
Tok událostí: 1. Vedoucí zaměstnanec zvolí možnost zobrazit výkazy práce. 2. Systém vypíše seznam všech výkazů práce podřízených pracovníků. 3. Vedoucí zvolí požadovaný výkaz práce. 4. Systém zobrazí výkaz práce s možnostmi. 5. Pokud byl výkaz práce potvrzen, zobrazí systém možnosti schválení a odemčení pracovního výkazu.
Následné podmínky:
Alternativní tok:

Případ užití: Odemknout potvrzený výkaz práce
ID: UC32
Účastníci: Vedoucí zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Vedoucí zaměstnanec. 2. Uživatel se musí nacházet na příslušném výkazu práce.
Tok událostí: 1. Vedoucí zaměstnanec zvolí možnostodemknout výkaz práce. 2. Systém zobrazí formulář pro popis problému. 3. Vedoucí zaměstnanec může doplnit důvody odemčení. 4. Systém uloží odemčený výkaz i s komentářem. 5. Systém odešle e-mail s informacemi podřízenému pracovníkovi.
Následné podmínky: 1. Výkaz práce byl odemčen. 2. Byl odeslán e-mail.
Alternativní tok:

Případ užití: Schválit potvrzený výkaz práce
ID: UC33
Účastníci: Vedoucí zaměstnanec
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Vedoucí zaměstnanec. 2. Uživatel se musí nacházet na příslušném výkazu práce.
Tok událostí: 1. Vedoucí zaměstnanec zvolí možnost schválit výkaz práce. 2. Systém uloží změny.
Následné podmínky: 1. Výkaz práce byl schválen.
Alternativní tok:

Případ užití: Zobrazit pracovní náplň
ID: UC34
Účastníci: Supervizor
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Supervizor.
Tok událostí: 1. Supervizor zvolí možnost zobrazit pracovní náplň. 2. Systém vypíše seznam uživatelů, ke kterým má supervizor oprávnění. 3. Supervizor vybere požadovaného uživatele. 4. Systém vypíše pracovní náplň zvoleného zaměstnance.
Následné podmínky:
Alternativní tok:

Případ užití: Zobrazit odpracované časy
ID: UC35
Účastníci: Mzdová účetní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Mzdová účetní.
Tok událostí: 1. Mzdová účetní zvolí možnost zobrazit zaměstnance. 2. Systém vypíše seznam všech zaměstnanců a vložených výkazů práce. 3. Mzdová účetní zvolí konkrétní výkaz práce. 4. Systém zobrazí odpracované časy za konkrétní měsíc.
Následné podmínky:
Alternativní tok:

Případ užití: Export všech výkazů práce do PDF
ID: UC36
Účastníci: Mzdová účetní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Mzdová účetní.
Tok událostí: 1. Mzdová účetní zvolí možnost zobrazit zaměstnance. 2. Systém vypíše seznam všech zaměstnanců a vložených výkazů práce. 3. Mzdová účetní zvolí možnost exportovat všechny výkazy práce. 4. Systém provede výběr a vytvoří PDF soubor. Výkazy práce, které nebyly potvrzeny nebo schváleny, budou výrazně označeny. 5. Systém nabídne výsledný PDF soubor ke stažení.
Následné podmínky:
Alternativní tok:

Případ užití: Exportovat salda
ID: UC37
Účastníci: 1. Mzdová účetní
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Mzdová účetní.
Tok událostí: 1. Mzdová účetní zvolí možnost zobrazit zaměstnance. 2. Systém vypíše seznam všech zaměstnanců a vložených výkazů práce. 3. Mzdová účetní zvolí možnost exportovat salda kont pracovní doby. 4. Systém sestaví soubor do požadovaného formátu. 5. Systém nabídne soubor ke stažení.
Následné podmínky:
Alternativní tok:

Případ užití: Vytvořit výkaz
ID: UC38
Účastníci: Pracovník HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník HUDS.
Tok událostí: 1. Pracovník HUDS zvolí možnost vložit nový výkaz. 2. Systém vypíše seznam dostupných farností, ke kterým má pracovník oprávnění. 3. Pracovník zvolí požadovanou farnost. 4. Systém nabídne formulář pro vložení výkazu. 5. Pracovník HUDS vyplní rok a odešle formulář. 6. Systém uloží výkaz.
Následné podmínky: 1. V systému je uložen nový výkaz.
Alternativní tok: 2.1 Systém vypíše hlášení, že přihlášený uživatel nemá oprávnění vyplňovat výkazy k žádné farnosti.

Případ užití: Vyplnit výkaz
ID: UC39
Účastníci: Pracovník HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník HUDS. 2. Uživatel se musí nacházet na seznamu výkazů.
Tok událostí: 1. Pracovník HUDS zvolí možnost upravit výkaz u konkrétního výkazu. 2. Systém zobrazí formulář výkazu HUDS pro konkrétní rok. 3. Pracovník vyplní položky a odešle formulář. 4. Systém provede validaci dat a formulář uloží. 5. Systém zobrazí seznam výkazů.
Následné podmínky: 1. Výkaz práce byl změněn.
Alternativní tok: 4.1 Pokud validace neproběhla v pořádku, systém vypíše chybové hlášení. 4.2 Systém znova zobrazí formulář výkazu HUDS.

Případ užití: Odevzdat výkaz
ID: UC40
Účastníci: Pracovník HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník HUDS. 2. Uživatel se musí nacházet na seznamu výkazů.
Tok událostí: 1. Pracovník HUDS u konkrétního výkazu zvolí možnost odevzdat výkaz. 2. Systém zkontroluje, zda jsou vyplněna všechna pole. 3. Systém uzamkne výkaz pro další úpravy.
Následné podmínky: 1. Výkaz HUDS byl uložen a uzamčen pro další úpravy.
Alternativní tok: 2.1 Pokud nejsou všechny údaje vyplněny, nelze výkaz uložit. 2.2 Systém zobrazí formulář pro úpravu výkazu HUDS.

Případ užití: Zobrazit přehled farností
ID: UC41
Účastníci: Správce HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce HUDS.
Tok událostí: 1. Správce HUDS zvolí možnost zobrazit přehled farností. 2. Systém vypíše přehled všech děkanátů. U děkanátu je uvedeno, kolik obsahuje farností a kolik formulářů bylo vytvořeno a kolik bylo potvrzeno. 3. Správce HUDS zvolí příslušný děkanát. 4. Systém zobrazí přehled všech farností ve zvoleném děkanátu s informací o výkazech HUDS.
Následné podmínky:
Alternativní tok:

Případ užití: Export do formátu MS Excel
ID: UC42
Účastníci: Správce HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce HUDS.
Tok událostí: 1. Pracovník HUDS zvolí možnost exportovat výkazy do MS Excel. 2. Systém zobrazí možnost zadání roku. 3. Pracovník HUDS zadá požadovaný rok. 4. Systém provede export po jednotlivých děkanátech na každý list. 5. Systém nabídne stažení souboru.
Následné podmínky:
Alternativní tok:

Případ užití: Upozornit emailem na nevyplněné výkazy
ID: UC43
Účastníci: Správce HUDS
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce HUDS.
Tok událostí: 1. Správce HUDS zvolí možnost upozornit e-mailem na nevyplněné výkazy. 2. Systém zobrazí formulář pro odeslání upozornění. 3. Správce HUDS zadá rok a zprávu. 4. Systém provede validaci zadaných údajů a odešle e-mail s upozorněním farnostem, které nemají potvrzené výkazy.
Následné podmínky:
Alternativní tok: 3.1 Pokud validace neproběhne v pořádku, systém vypíše chybové hlášení.

Případ užití: Upravit položky
ID: UC44
Účastníci: Administrátor
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Administrátor.
Tok událostí: 1. Administrátor zvolí možnost upravit položky. 2. Systém vypíše seznam editovatelných položek. 3. Administrátor vybere položku k editaci. 4. Systém zobrazí formulář k editaci položky. 5. Administrátor provede požadované změny.
Následné podmínky: 1. Položka výkazu byla upravena.
Alternativní tok:

Případ užití: Zadat prázdnou opravu
ID: UC45
Účastníci: Pracovník oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník oprav.
Tok událostí: 1. Pracovník oprav zvolí možnost zadávání oprav. 2. Systém zobrazí seznam dostupných farností. 3. Pracovník oprav zvolí požadovanou farnost. 4. Systém zobrazí formulář pro zadání prázdné opravy na zvolený rok. 5. Pracovník oprav potvrdí, že chce zadat prázdnou opravu. 6. Systém uloží u farností prázdnou opravu.
Následné podmínky: 1. Prázdná oprava byla vložena do systému.
Alternativní tok:

Případ užití: Vložit místo
ID: UC46
Účastníci: Pracovník oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník oprav.
Tok událostí: 1. Pracovník oprav zvolí možnost zadávání oprav. 2. Systém zobrazí seznam dostupných farností. 3. Pracovník oprav zvolí požadovanou farnost. 4. Systém zobrazí formulář pro zadání místa, kde se nachází opravovaný objekt. 5. Pracovník oprav zadá místo. 6. Systém uloží místo.
Následné podmínky: 1. Místo bylo uloženo do systému
Alternativní tok: 5.1 Pokud není místo zadáno správně, systém zobrazí chybové hlášení. 5.2 Systém zobrazí formulář pro zadání místa.

Případ užití: Vložit objekt
ID: UC47
Účastníci: Pracovník oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník oprav. 2. Pracovník oprav se musí nacházet na místě opravy.
Tok událostí: 1. Pracovník oprav zvolí možnost přidat objekt. 2. Systém zobrazí formulář pro vložení objektu, který se bude opravovat. 5. Pracovník oprav zadá název objektu a zvolí, zda je objekt památkově chráněný. 6. Systém uloží objekt.
Následné podmínky: 1. Objekt byl uložen do systému.
Alternativní tok: 5.1 Pokud není objekt zadán správně, systém zobrazí chybové hlášení. 5.2 Systém zobrazí formulář pro zadání objektu.

Případ užití: Vybrat druh opravy
ID: UC48
Účastníci: Pracovník oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník oprav. 2. Pracovník oprav se musí nacházet na požadovaném objektu.
Tok událostí: 1. Pracovník oprav zvolí možnost přidat druh opravy. 2. Systém zobrazí formulář pro přidání druhu, který se bude opravovat. 5. Pracovník oprav vybere požadovaný druh opravy ze seznamu. 6. Systém uloží druh opravy.
Následné podmínky: 1. Druh opravy byl přidán k objektu.
Alternativní tok:

Případ užití: Vybrat zdroje financování
ID: UC49
Účastníci: Pracovník oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Pracovník oprav. 2. Pracovník oprav se musí nacházet na požadovaném druhu opravy u objektu.
Tok událostí: 1. Pracovník oprav zvolí možnost vložení financování. 2. Systém zobrazí formulář pro vložení financování. 3. Pracovník oprav vybere způsob financování, předfinancování, zadá částku a odešle formulář. 4. Systém uloží údaje o financování.
Následné podmínky: 1. Do systému byl přidán druh financování.
Alternativní tok: 3.1 Pokud nejsou správně zadané údaje, systém vypíše chybové hlášení. 3.2. Systém zobrazí formulář pro vložení financování.

Případ užití: Zobrazit přehled
ID: UC50
Účastníci: Správce oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce oprav.
Tok událostí: 1. Uživatel zvolí možnost zobrazit přehled oprav. 2. Systém vypíše všechny zadané opravy podle děkanátů a farností v daném roce.
Následné podmínky:
Alternativní tok:

Případ užití: Vložit druh opravy
ID: UC51
Účastníci: Správce oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce oprav.
Tok událostí: 1. Správce oprav zvolí možnost upravit druhy opravy. 2. Systém vypíše seznam druhů opravy a zobrazí formulář pro vložení nového. 3. Správce oprav zadá ID a název položky a odešle formulář. 4. Systém provede validaci údajů a uloží druh opravy.
Následné podmínky: 1. Do systému byl uložen nový druh opravy.
Alternativní tok: 4.1 Pokud nebyla validace úspěšná, systém vypíše chybové hlášení.

Případ užití: Vložit zdroje financování
ID: UC52
Účastníci: Správce oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce oprav.
Tok událostí: 1. Správce oprav zvolí možnost upravit zdroje financování. 2. Systém vypíše seznam zdrojů financování a zobrazí formulář pro vložení nového zdroje. 3. Správce oprav zadá ID a název položky a odešle formulář. 4. Systém provede validaci údajů a uloží zdroj financování.
Následné podmínky: 1. Do systému byl uložen nový zdroj financování.
Alternativní tok: 4.1 Pokud nebyla validace úspěšná, systém vypíše chybové hlášení.

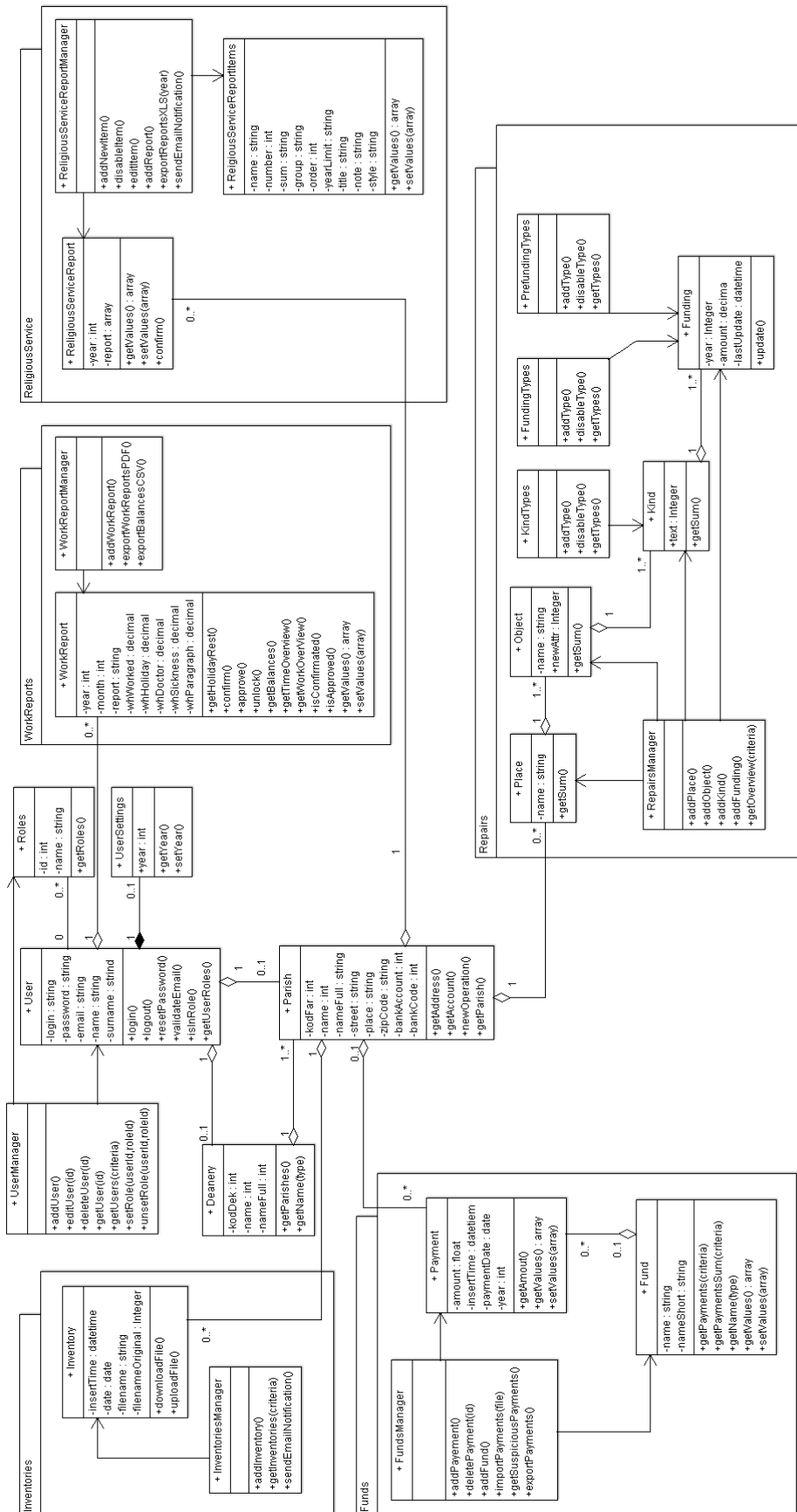
Případ užití: Zrušit druh opravy
ID: UC53
Účastníci: Správce oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce oprav.
Tok událostí: 1. Správce oprav zvolí možnost upravit druhy opravy. 2. Systém vypíše seznam druhů opravy. 3. Správce oprav zvolí druh opravy pro zrušení. 4. Systém zneplatní druh opravy pro další využití. Ze systému nebude fyzicky odstraněn.
Následné podmínky: 1. Druh opravy byl zneplatněn.
Alternativní tok:

Případ užití: Zrušit zdroje financování
ID: UC54
Účastníci: Správce oprav
Vstupní podmínky: 1. Uživatel musí být přihlášen do systému s oprávněním Správce oprav.
Tok událostí: 1. Správce oprav zvolí možnost upravit zdroje financování. 2. Systém vypíše seznam zdrojů financování. 3. Správce oprav zvolí zdroj financování pro zrušení. 4. Systém zneplatní zdroj financování pro další využití. Ze systému nebude fyzicky odstraněn.
Následné podmínky: 1. Zdroj financování byl zneplatněn.
Alternativní tok:

PŘÍLOHA P III: MATICE SLEDOVATELNOSTI POŽADAVKŮ

[illegible]

PŘÍLOHA P IV: MODEL TRŽÍD



PŘÍLOHA P V: SCHÉMA DATABÁZE

