SYSTÉM PRO MARKETINGOVÉ ŠÍŘENÍ AGENDY ON-LINE PRODEJE
SYSTEM FOR MARKETING PROPAGATION OF ON-LINE SALE AGENDA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. JIŘÍ DRDLA

VEDOUCÍ PRÁCE
SUPERVISOR
Prof. Ing. Tomáš Hruška, CSc.

BRNO 2015
Abstrakt

Abstract
This paper describes the design of an information system for online sales. There is the problem definition in first chapter, then the requirements are defined and competitive software solutions are analyzed. In the next chapters, individual functions and their implementations are described in detail and result of the chapter is the design of information system.

Klíčová slova
Informační systém, webová aplikace, internetový prodej, ASP.NET MVC, Visual Basic .NET, správa objednávek.

Keywords
Information systém, web application, online sales, ASP.NET MVC, Visual Basic .NET, order management.

Citace
Bc. Jiří Drdla: Systém pro marketingové šíření agendy on-line prodeje, diplomová práce, Brno, FIT VUT v Brně, 2015
**Prohlášení**

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením prof. Tomáše Hrušky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

........................................

Jiří Drdla
20. května 2015

**Poděkování**

Chtěl bych poděkovat vedoucímu své diplomové práce, panu prof. Tomáši Hruškovi, za poskytnuté konzultace a vhodné směrování při práci na projektu a zejména při tvorbě technické zprávy.

© Jiří Drdla, 2015

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

1. Úvod .................................................................................................................................. 4

2. Analýza problému ............................................................................................................. 5
   2.1 Definice požadavků ........................................................................................................ 5
       2.1.1 Shrnutí požadavků .................................................................................................. 7
   2.2 Konkurenční řešení ...................................................................................................... 7
       2.2.1 Manažer prodeje .................................................................................................... 8
       2.2.2 Money S3 ............................................................................................................. 8
       2.2.3 OpenCart ............................................................................................................. 8
       2.2.4 PrestaShop .......................................................................................................... 9
       2.2.5 Zhodnocení konkurenčních produktů ................................................................. 9
   2.3 Diagram případů užití ................................................................................................. 10

3. Technické prostředky ....................................................................................................... 11
   3.1 ASP.NET MVC ........................................................................................................... 11
       3.1.1 Controller podrobně ............................................................................................ 12
   3.2 Entity Framework ....................................................................................................... 13
   3.3 MySQL server ........................................................................................................... 13

4. Návrh .................................................................................................................................. 15
   4.1 Základní rozhraní aplikace ......................................................................................... 15
   4.2 Funkce aplikace ......................................................................................................... 16
       4.2.1 Adresář ................................................................................................................. 16
       4.2.2 Objednávky .......................................................................................................... 17
       4.2.3 Zboží .................................................................................................................... 18
       4.2.4 Zásilky ................................................................................................................ 19
       4.2.5 Reklamace .......................................................................................................... 20
       4.2.6 Import objednávek ............................................................................................... 22
       4.2.7 Import bankovních plateb ..................................................................................... 24
       4.2.8 Uživatelé systému ............................................................................................... 25
   4.3 Databáze ..................................................................................................................... 25
5. Implementace ..................................................................................................................... 29
  5.1 Struktura projektu ........................................................................................................... 29
  5.2 Vzhled aplikace ................................................................................................................ 31
     5.2.1 Bootstrap a jQuery ........................................................................................................ 31
     5.2.2 Použitá rozšíření ......................................................................................................... 32
     5.2.3 Grafická šablona .......................................................................................................... 34
     5.2.4 Struktura pohledů ......................................................................................................... 36
  5.3 Správa uživatelů ................................................................................................................. 37
     5.3.1 ASP.NET Identity ........................................................................................................... 38
     5.3.2 Proces registrace uživatelů ........................................................................................ 39
  5.4 Agendy v systému .............................................................................................................. 39
  5.5 Dashboard ........................................................................................................................ 42
  5.6 Zajímavé části implementace ............................................................................................ 43
     5.6.1 Akce vykonávané na pozadí .......................................................................................... 43
     5.6.2 Generování emailových zpráv a nabídek zboží ........................................................ 45
     5.6.3 Tisk dokumentů .............................................................................................................. 48
     5.6.4 Import a export dat ........................................................................................................ 52
     5.6.5 Zálohování a obnova .................................................................................................... 56
6. Propagace produktu ............................................................................................................ 58
  6.1 Možnosti propagace .......................................................................................................... 58
     6.1.1 Webová prezentace ...................................................................................................... 58
     6.1.2 Google AdWords .......................................................................................................... 59
     6.1.3 Bannerová inzerce ....................................................................................................... 59
     6.1.4 Sociální sítě .................................................................................................................. 59
  6.2 Zvolený způsob propagace ................................................................................................. 60
  6.3 Reálné nasazení .................................................................................................................. 61
7. Závěr .................................................................................................................................... 63
8. Literatura .............................................................................................................................. 64
9. Seznam použitých zkratek a symbolů ................................................................. 68
10. Seznam příloh ........................................................................................................ 69
Kapitola 1

Úvod

Trendem posledních let ve světě maloobchodu je prodej zboží a služeb přes internet. Kromě velkých hráčů na tomto poli působí i celá řada malých internetových prodejců, kteří také tvoří nemalé procento celkového obratu v online prodeji. Nedílnou součástí při každém internetovém prodeji je však velké množství agendy, kterou se musí internetový prodejce zabývat. Větší korporace v tomto ohledu zpravidla zajišťuje robustní informační systémy, jež dokáží mnohé rutinní činnosti ve velké míře automatizovat. Řešení v podobě na míru sestrojeného informačního systému je ovšem pro malé internetové prodeje ve většině případů nepřijatelně nákladné. Předmětem této diplomové práce je návrh a implementace informačního systému, zaměřujícího se právě na tyto menší prodeje. Cílem navrhovaného informačního systému je stejně jako u systémů větších společností usnadnit a zefektivnit činnosti spojené s internetovým prodejem.

Následující kapitola rozebírá řešený problém, definuje požadavky na navrhovaný informační systém a navrhnuté řešení srovnává s konkurenčními produkty. Podrobněji jsou popsány klady a zápory vybraných existujících řešení a nastíněn způsob, jakým bude chybějící funkčnost implementována v tomto projektu.

Kapitola třetí pojednává o technických prostředcích použitých pro implementaci navrženého systému. Popsány jsou zde použité technologie a prostředky, jejich srovnání s konkurencí a je zde zdůvodněn právě jejich výběr.

Čtvrtou kapitolou je návrh informačního systému, podrobně je popsána celková funkčnost systému a samostatně jednotlivé funkce. Součástí kapitol je i nezbytné diagramy komplexně popisující návrh databáze a architekturu systému.

Kapitola pátá se zaměřuje na samotnou implementaci informačního systému. Nejprve jsou představeny její obecné části a principy fungování a poté jsou detailně rozebrány některé její zajímavé pasáže. Složitější části jsou opět graficky popsány pomocí diagramů.

V předposlední kapitole jsou rozebrány některé možnosti propagace, z nichž jedna je zvolena a realizována. Z realizace jsou posléze vyvozeny potřebné závěry a je nastíněn další postup v propagaci produktu.

Závěrem práce jsou shrnuty dosažené výsledky a jsou nastíněny možnosti pro další vývoj aplikace.
Kapitola 2

Analýza problému

V této kapitole se zaměříme na analýzu problému řešeného v tomto diplomovém projektu, určíme si jeho cíle a navrhněné řešení srovnáme s jinými konkurenčními produkty.

Jak již bylo nastíněno v úvodu, navrhnutý informační systém se zaměřuje na menší internetové prodeje, kterým má poskytnout kvalitní a robustní nástroj pro usnadnění rutinních činností spojených s jejich prodejními aktivitami. Mezi základní činnosti, které internetového prodeje denně zaměstnávají, patří správa objednávek, zákazníků, evidence prodávaného zboží a expedice zásilek. K těmto činnostem se často nabízí také vyřizování reklamací jednak od zákazníků, ale i směrem k dodavatelům.

Mnoho z těchto menších internetových prodejců nabízí své zboží kromě svého internetového obchodu také na celé řadě dalších internetových portálů, jako jsou aukční servery, slevové weby nebo internetová inzercce. A právě zde prodejci naráží na problém jak efektivně spravovat objednávky na několika různých prodejních portálech najednou. Zpravidla každý z takových portálů, ať už se jedná o internetový obchod, aukční server nebo slevový portál, nabízí prodejcům rozhraní pro správu zboží a objednávek. Obsluha několika takových rozhraní je však výrazně časově náročnější než objednávky a zboží spravovat z jednoho místa pro všechny portály najednou. A právě takové místo, zastřešující prodej na několika portálech pod jedno rozhraní, se snaží tato diplomová práce navrhovat.

Po několikerém mylném pochopení při představování tohoto projektu je třeba zdůraznit, že se nejedná o internetový obchod, a to ani v žádné jeho obměně. Implementovaný systém bude sloužit pouze prodejcům ke správě prodejní agenty a nikoli jejich zákazníkům. S tím také souvisí plánovaná existence uživatelského rozhraní systému výhradně pro tyto prodeje, ne pro zákazníky. Cílem systému je agregovat data o prodejích na jiných prodejních portálech, kterými mohou být mimo jiné i internetové obchody.

2.1 Definice požadavků

Správa objednávek je sice hlavním, ne však jediným cílem navrhněné webové aplikace. Další podstatnou součástí je evidence prodávaného zboží a s ním souvisejících nákupů. Evidované zboží bude potřeba dále exportovat na prodejní portály popsané níže. Z tohoto důvodu je nutné u zboží uchovávat informace o jeho typu, zařazení do kategorie a je vhodné uvažovat o evidenci produktových obrázků a videí.
Zejména prodejci elektroniky a jiného finančně hodnotnějšího zboží mají potřebu v informačním systému evidovat nejen množství naskladněného zboží, ale také jednotlivé kusy a jejich sérieová čísla či jiné specifické údaje. Tato funkčnost bude tedy do systému rovněž implementovaná, avšak u jednotlivých produktů bude volitelná.


Další podstatnou funkcí je spojenou s prodejními portály je možnost stahovat objednávky zákazníků uskutečněné na těchto prodejních místech. Je to klíčová funkce definující celý informační systém a pro prodejce takřka nejdůležitějším aspektem pro usnadnění práce spojené s prodejem. V závislosti na prodejním portále bude také možné stahovat data zákazníků a přidávat je do adresáře nebo evidovat platby uskutečněné přes daný prodejní portál, viz dále.


Systém bude umožňovat evidovat k objednávkám přijaté bankovní platby a také platby realizované přes platební brány jednotlivých prodejních portálů, jmenovitě například platební brána PayU od Aukro.cz. S evidencí bankovních plateb půjde ruku v ruce také evidence bankovních účtů. Pro vybrané bankovní subjekty a jejich systémy internetového bankovnictví bude v našem informačním systému implementována možnost automatického importu přijatých plateb a spárování s objednávkami.
Systém bude dále disponovat evidenci odeslaných poštovních zásilek s možností tisku nejrůznějších poštovních dokumentů, jako jsou balíkové štítky, podací lístky, poštovní poukázky a další. Evidence odeslaných zásilek není přítomna ani v jednom z uvedených existujících řešení, přitom pro prodejce je přehled o zásilkách také jednou z priorit.

Poslední, ale neméně důležitou součástí navrhovaného systému je samozřejmě možnost registrovat do našeho systému nové prodeje, kteří jej budou používat pro své prodejní aktivity. Systém bude pro běžné používání zpoplatněn, avšak po omezenou dobu jej bude možné využívat bezplatně pro účely vyzkoušení a seznámení se.

2.1.1 Shrnutí požadavků

Z výše uvedených odstavců si tedy v bodech určeme, jaké nároky máme na navrhovaný informační systém:

- Správa objednávek
- Import objednávek z prodejních portálů
- Evidence zboží, včetně parametrů a sériových čísel
- Export zboží na prodejní portály, generování popisů
- Správa reklamací zákazníků a reklamací u dodavatelů
- Evidence odeslaných zásilek
- Tisk poštovních dokumentů a přehledů dle šablon
- Sledování přijatých plateb
- Import přijatých plateb z internetového bankovnictví
- Evidence zákazníků
- Registrace uživatelů systému

Z těchto základních požadavků vzejdou v průběhu návrhu další dílčí potřeby a požadavky, které bude nutné v aplikaci implementovat. Řešení těchto a dalších požadavků si představíme v následujících kapitolách.

2.2 Konkurenční řešení

V této podkapitole si představíme některá potenciálně konkurenční řešení tohoto projektu. Ve srovnání budou představena řešení vhodná pro použití malými internetovými prodejci, nikoli rozsáhlé a drahé informační systémy určené pro korporátní použití.
2.2.1 Manažer prodeje


Zásadní výhodou našeho navrhovaného systému tedy bude napojení na více prodejních portálů, práva reklamací a širší možnosti tisku dokumentů.

2.2.2 Money S3

Tento systém není přímým konkurentem navrhované webové aplikace. Především se jedná o desktopové řešení a ze svého principu ekonomický software. Tento software poskytuje ve svém základu velice pokročilou správu objednávek a nákupů, vedení databáze zákazníků a propracovanou evidenci skladu. Díky množství zásuvných modulů ovšem umožňuje rozšířit svoji funkcionalitu také o evidenci reklamací a import plateb z banky. Cena tohoto řešení včetně DPH přesahuje 5500 Kč včetně DPH. Zcela zvláštní záležitostí je napojení systému Money S3 na nějaký prodejní portál. To je možné realizovat pouze prostřednictvím speciálního datového konektoru v ceně dalších 4999 Kč včetně DPH. Slouží však zejména pro import objednávek a faktur z e-shopu. Neposkytuje možnost evidované zboží do e-shopu exportovat. [10]

Navrhovaný systém nebude bohužel konkurovat v oblastech účetnictví, ale poskytne zejména cenově přijatelnější alternativu pro usnadnění činnosti při práci se zbožím a automatizaci v oblasti exportu nabízeného zboží.

Společnost Cígler Software je provozovatelem ještě jednoho potenciálně konkurenčního produktu a tím je iDoklad. Jedná se o webovou aplikaci určenou výhradně pro tvorbu faktur. Tato aplikace je poskytována zdarma, avšak nenabízí nic jiného než možnost vytváření faktur a evidenci zákazníků.

2.2.3 OpenCart

Systém s tímto názvem je v prvé řadě open source řešením internetového obchodu. Pro naše porovnání se však zaměříme na administrační část systému, která může potenciálně nabízet podobnou funkčnost jako námi navrhovaný informační systém. [11]
OpenCart opět poskytuje propracovanou správu objednávek, evidenci zboží a zákazníků. Pomocí zásuvných modulů lze dodat také evidenci nákupů, tisk dle přednastavených tiskových šablon a v omezené míře též správu reklamací. Chybí však jakékoli napojení na české bankovní systémy a rovněž nejsou dostupné datové konektory pro žádný z českých prodejních portálů. Jakkoli by šla tato funkčnost více či méně doplnit, u zásuvných modulů je prakticky vždy problém s jejich funkčností napříč jednotlivými verzemi systému.

Oproti tomuto systému tedy navrhovaná webová aplikace nabídne napojení na další prodejní portály, správu bankovních plateb, zásilek a poskytne širší možnosti tisku.

2.2.4 PrestaShop

Stejně jako u systému OpenCart se jedná především o e-shopové řešení a potenciální konkurence tkví zejména v jeho administrační části. Ta stejně jako u OpenCart nabízí velice dobrou správu objednávek, zboží a zákazníků. Díky rozšířenosti větší než má systém OpenCart, existuje také širší nabídka zásuvných modulů. Ty do kází přidat mezi stávající funkcionalitu reklamace, zásilky, širší možnosti tisku i napojení na prodejní portály jako Aukro.cz nebo Slevomat.cz. Problemem však je stále chybějící podpora pro napojení na systémy internetového bankovnictví nebo vytváření popisu zboží na základě parametrů. Výhodou systému PrestaShop i výše zmíněného OpenCart je, že jsou open-source a i pro komerční účely zdarma. [12]

Oproti systému PrestaShop tedy navrhované řešení nabízí napojení na systémy internetového bankovnictví, automatizovanou tvorbu popisu zboží na základě parametrů a dle zkušeností také kvalitnější podporu tisku. Ať už půjde o tisk poštovních dokumentů k zásilkám nebo například tisk faktur.

2.2.5 Zhodnocení konkurenčních produktů

Výše bylo představeno pouze několik produktů, které by v budoucnu mohly být konkurencí pro navrhovanou webovou aplikaci. Jedná se samozřejmě o úzký výběr a na trhu existuje mnoho jiných podobných řešení.

Současné existující produkty lze obecně rozdělit do dvou kategorií. Tou první je skupina software primárně zaměřená na vedení účetnictví, která funguje spojeně s prodejem implementuje spíše okrajově a tím pádem i dosti omezeně. Takovým případem je právě Money S3 a jemu podobné ekonomické systémy.

Druhou kategorií je skupina aplikací, z nichž většina existuje v online verzi, která se zaměřuje více na činnosti spojené se samotným prodejem, ale jejich použití je spojeno pouze s jediným prodejním portálem. Příkladem takové aplikace je Manažer prodeje, který lze použít pouze k administraci prodejů na Aukro.cz. Představená řešení v podobě open-source internetových obchodů sice v některých případech poskytují napojení i na jiné prodejní portály kromě vlastního
prodejního rozhraní, avšak z osobních zkušeností je takové napojení vždy poněkud násilné ne vždy bezproblémové.

Cílem projektu je tedy vytvořit webovou aplikaci, která dokáže agregovat data o prodejích z více prodejních portálů a tyto portály také pomocí jednotného rozhraní plnit nabídkou zboží prodejce. Aplikace by měla být zaměřena primárně na usnadnění samotných prodejních procesů a méně řešit otázky vedení účetnictví.

2.3 Diagram případů užití

V předchozích podkapitolách jsme si specifikovali požadavky na navrhovaný informační systém a tyto požadavky zkonfrontovali s několika existujícími řešeními. Pro lepší představu při následném návrhu informačního systému si navržené požadavky vizualizujeme. K tomu použijeme diagram případů užití, výsledek je vidět na obrázku 2.1.

Obrázek 2.1: Diagram případů užití pro navrhovaný informační systém
Kapitola 3

Technické prostředky

Kapitola třetí se zaměřuje na představení nástrojů použitých při návrhu a následné implementaci projektu. Podrobněji rozebrány zde budou použité technologie a frameworky a použitý databázový backend.

3.1 ASP.NET MVC

"ASP.NET MVC is a radical shift for web developers using Microsoft platform. It emphasizes clean architecture, design patterns, and testability, and doesn’t try to conceal how the Web works.” [33]


Obrázek 3.1: Schéma vzoru MVC [15]
Model představuje reprezentaci informací, se kterými se v aplikaci pracuje. Jeho obsahem jsou rovněž operace nad těmito daty. View (pohled) je určen k převodu dat modelu do podoby vhodné k reprezentaci uživateli. A ve finále Controller (řadič) je součást, která zodpovídá za reakce na požadavky uživatele, zajišťuje změny v modelu a vytváření pohledu.


3.1.1 Controller podrobně


Obrázek 3.2: Adresa požadavku

Nyní si pojďme adresu z obrázku rozebrat a vysvětlit si na ní funkci controlleru. První část tvoří standardně doménové jméno webového serveru, za prvním lomítkem vidíme název controlleru, který náš požadavek bude zpracovávat, za dalším lomítkem název Akce, tedy metody, do níž se zasílaný požadavek předá, a nakonec parametry. Ukázka na obrázku tedy říká, že požadujeme stránku pro editaci objednávky s identifikačním číslem 3. Samotný kód controlleru pak může vypadat následovně:

```csharp
Public Class OrderController
    Inherits Controller
    ...
    Function Edit(ByVal Id As Integer?) As ActionResult
    ...
    End Function
End Class
```
Logika ASP.NET pomocí reflexe [16] z požadavku automaticky zjistí, na který controller a kterou jeho akci je požadavek cílen, vytvoří instanci požadovaného controlleru a jeho metodu zavolá i s předávanými parametry.

Předávaným parametrem nemusí být pouze hodnoty primitivních datových typů. ASP.NET disponuje funkcí zvanou model binding, která přijaté parametry dovoluje interně konvertovat a volané metodě předat ve formě objektu. Metoda využívající funkci model binding může vypadat následovně.

| Function SendEmail(Model As Order, IdTemplate as Integer) As ActionResult |
| End Function |

Na příkladu vidíme akci pro zaslání emailové zprávy k objednávce, přičemž objednávka je předávána již ve formě objektu, avšak z webového prohlížeče byla odeslána jako HTML formulář, tedy jako kolekce dvojic klíč-hodnota. Model binding je velice mocný nástroj, usnadňující mnoho rutinních činností. Na druhou stranu se zatím nedokáže vypořádat například s hierarchickou strukturou objektů. Jak byl tento problém vyřešen v tomto projektu, se dozvíme v kapitole 5.4. Předmětem této práce však není popis funkce ASP.NET, představeny byly pouze funkce nezbytné pro další čtení této práce. Více o technologii je možné se dozvědět z odkazované literatury.

3.2 Entity Framework


3.3 MySQL server

Jako databázový backend poslouží volně šiřitelný databázový systém MySQL od společnosti Oracle. Jedná se o jednu z nejpoužívanějších relačních databází, která bude pro účely projektu plně postačovat. [4] Při vybírání databázové technologie se nabízely též technologie od Microsoftu, avšak zejména kvůli technickým omezením jejich bezplatných verzí jsem vsadil právě na MySQL. Jednou z alternativ bylo použít Microsoft SQL Server v bezplatné Express verzi. Ta však nese velice citelné omezení na velikost databáze a to 2GB. To může být problém například při ukládání objemných obrazových dat k evidovanému zboží, které si popíšeme dále v kapitole 4. Další variantou bylo použití Microsoft SQL
Serveru Compact. Ten je sice taktéž dostupný bezplatně, je to ovšem pouze souborová databáze pouze se základní funkcíností. Nedovoluje například definovat pohledy, triggery a další.
Kapitola 4

Návrh

V této kapitole se zaměříme na návrh samotné aplikace a jednotlivých programových funkcí a to podle požadavků specifikovaných v kapitole 2 – Analýza problému. Každá navrhovaná funkcionalitě bude věnována speciální podkapitola a funkčnost bude v případě potřeby reprezentována odpovídajícími diagramy.

4.1 Základní rozhraní aplikace

Do základního rozhraní aplikace vstupuje uživatel po přihlášení do systému. Na náčrtu v obrázku 4.1 je patrné, jak by toto rozhraní mělo vypadat. Obrazovka aplikace se dělí do třech hlavních částí, kterými jsou horní lišta, levý postranní panel a individuální obsah, zaujímající největší prostor. Tímto obsahem může být například tabulkové zobrazení položek při výběru agendy nebo detail konkrétní vybrané položky.

Následující a všechny další náčrty obrazovek byly vytvořeny v programu Balsamiq Mockups. [40] Jedná se pouze hrubé náčrty s omezenou množinou ovládacích prvků, výsledná podoba grafického rozhraní se může lišit.

Obrázek 4.1: Návrh rozhraní webové aplikace
Horní lišta v aplikaci zobrazuje logo informačního systému, v její pravé horní části budou informace o přihlášeném uživateli, ale zejména obsahuje sadu tlačítek pro přepínání agend v aplikaci. Těmito agendami jsou například seznam objednávek evidovaných v aplikaci, výpis předmětů, odeslaných základí nebo evidovaných reklamací.

Levý postranní panel v sobě bude sdružovat funkce dostupné pro aktuálně vybraný pohled. Pro příklad, pokud bude v horní liště zvolený pohled **Objednávky**, v levém postranním panelu budou dostupné volby pro přidání nové objednávky, editace nebo smazání aktuálně označené objednávky a dále například funkce pro zobrazení celkové ceny označených objednávek, tisk faktur nebo třeba import a export.


Obrazovky detailu pro všechny typy položek budou mít podobná grafická rozhraní, jak bude vidět na obrázcích v následujících podkapitolách. V horní části obrazovky bude zobrazen název otevřené položky a cesta v rámci agendy. Pod pouto informací bude přítomna řada voleb. Po levé straně to budou tlačítka pro uložení provedených změn a pro zavření detailu bez uložení. Vpravo od nich se poté budou nacházet volby specifické pro konkrétní otevřenou položku. Pod těmito volbami se budou skrývat akce, které bude možné s položkou provádět. Spodní část bude poté vyhrazena jednotlivým vstupním polím, kde budou zobrazeny detailní informace.

### 4.2 Funkce aplikace

Tato podkapitola shrnuje jednotlivé navržené funkce aplikace, naznačuje jejich budoucí grafické rozhraní a složitější procesy modeluje pomocí adekvátních diagramů. Rozehraný jsou postupně správa objednávek, zboží, nákupů, zákazníků a reklamací, dále import objednávek a export produktů na prodejní portály a také import plateb z internetového bankovnictví.

#### 4.2.1 Adresář

Adresář slouží k evidenci zákazníků, kteří u prodejce v minulosti nakoupili nebo teprve nakoupili. U zákazníků je nezbytné evidovat dvě adresy – fakturační a doručovací, dále IČO a DIČ a kontaktní údaje jako email a telefon. Stejně jako u objednávek lze zákazníky přidávat do systému ručně nebo automatickým importem z některého prodejního portálu. Teoreticky není nezbytné uchovávat

Na obrázku 4.2 vidíme, jak by mělo ve výsledku vypadat okno pro zobrazení detailu evidovaného zákazníka, říkejme mu například Karta zákazníka.

Obrázek 4.2: Návrh okna detailu zákazníka

4.2.2 Objednávky

Správa objednávek je jednou z hlavních devíz navrhované webové aplikace a to z toho důvodu, že právě objednávky jsou záležitostí, se kterými budou internetoví prodejci pracovat nejčastěji. Objednávky je možné přidávat buď ručně, nebo automatickým importem z některého prodejního portálu, jak bude ukázáno v dalších podkapitolách. Při zobrazení objednávek budou dostupné standardní CRUD operace, tedy přidání, čtení, editace a mazání objednávky. Dále pak tisk dle zvolené tiskové šablony, funkce pro hromadnou změnu stavu, funkce pro zobrazení součtu cen označených objednávek a speciální volba Vyřídit objednávky.

Volbou Vyřídit objednávky je funkčnost využívaná vždy při expedování objednaného zboží. Tato volba zajistí odeslání informačního emailu zákazníkovi s informací o expedování zboží, změnu stavu vybraných objednávek na Vyřízené a označení souvisejících zásilek jako Odeslané.

Na obrázku 4.3 nalezneme první grafický návrh okna pro zobrazení detailu objednávky. V jeho horní části najdeme prvky pro uložení změn, zavření bez uložení a funkce dostupné pro aktuálně zobrazenou objednávku. Pod těmito prvky následují již volby specifické pro objednávku. Těmito volbou pro změnu stavu objednávky, způsobu dopravy a platby, dále fakturační a doručovací adresa zákazníka, kontaktní údaje nebo poznámka. Spodní část okna detailu poté vyplňuje
editovatelný seznam položek objednávky. Seznam bude možné ručně editovat sadou funkcí dostupných v liště nad ním.

Obrázek 4.3: Návrh okna detailu objednávky

4.2.3 Zboží

Evidence prodávaného zboží je opět důležitou součástí agendy maloobchodního prodejce. U zboží je nutné sledovat jeho skladové pohyby, tedy nákupy a prodeje a počet kusů na skladě. Do systému budou produkty vstupovat prostřednictvím evidovaných nákupů, viz dále, a znovu vystupovat pomocí objednávek zákazníků, více v předchozí kapitole. Kromě toho se v naší aplikaci zaměříme i na evidenci parametrů zboží nebo produktových obrázků. Webová aplikace bude pro své uživatele vytvářet souhrnné statistiky pro konkrétní položky zboží, jako jsou obrat, počty nakoupených a prodaných kusů, vývoj cen nebo ve vztahu k evidovaným reklamacím a také poruchovost.

Potřebnými jednoduchými parametry pro evidenci zboží bude samozřejmě jeho název, typ, aktuální prodejní cena a nezbytná poznámka. Rozložení jednotlivých grafických prvků v okně formuláře pro detail zboží naleznete na obrázku 4.4.

definována zvláštní šablona parametrů. V šablone s názvem Stoly poté budou definovány parametry jako materiál, dekor, rozložitelnost, rozměry, atd. Jakmile pak prodejce začne evidovat nový produkt typů Stůl, bude možné u něj tyto parametry vyplnit a následně také použít například pro automatické vytváření popisů zboží, viz dále v této kapitole.

Obrázek 4.4: Návrh okna detailu zboží

4.2.4 Zásilky

Podstatnou záležitostí pro internetové prodejce je evidence zásilek. Jelikož takový prodejce většinu svého prodeje realizuje na dálku, je pro něj důležité mít přehled také o odchozích zásilkách. Z tohoto důvodu bude evidence zásilek přítomna i v naší navrhované aplikaci. Zásilky bude možné v systému vytvářet ručně, ale i na základě existujících objednávek, zákazníků nebo reklamací. V zobrazení zásilek na hlavní stránce aplikace budou opět k dispozici základní CRUD operace a k nim dále možnost tisku nejrůznějších poštovních dokumen tů dle tiskových šablon a export zásilek do formátu CSV či XML pro použití v systémech zásilkových služeb.

U zásilek bude možné definovat adresu příjemce, jeho kontaktní údaje, dále cenu zásilky, dobírkovou částku, cenu poštovného, číslo účtu a variabilní symbol pro zaslání dobírkové částky. Nesmějí chybět ani podací údaje jako číslo zásilky, typ zásilky a podstatná může být i hmotnost.
Většinu těchto údajů je možné při vytváření zásilky přejmout z objednávky, respektive od zákazníka, případně od evidované reklamace.

Jak již bylo nastíněno výše, evidence zásilek slouží nejen pro vytváření přehledu o odeslaných zásilkách, ale rovněž pro možnost tisku nejrůznějších dokumentů k těmto zásilkám. Počítá se s možností tisku tří druhů balíkových štítků (2 obecné a 1 dle vzoru České pošty [13]), s tiskem poštovních poukázek (Poštovní dobírková poukázka A a C od České pošty [14]), podacího lístku a podacího archu (rovněž po vzoru České pošty). Sada připravených šablon bude samozřejmě uživatelsky rozšířitelná a tak přizpůsobitelná na míru každému prodejci. V budoucnu se počítá s vytvořením dalších defaultních šablon pro všechny prodejce a pro další přepravce. Na obrázku 4.5 vidíme předpokládaný vzhled formuláře pro zásilku.

![Obrázek 4.5: Návrh okna detailu zásilky](image)

4.2.5 Reklamace

Důležitou navrhovanou součástí webové aplikace je evidence reklamací. Tuto funkci zároveň nenabízí ve výchozím stavu žádné ze zmiňovaných konkurenčních řešení. Zejména pro maloobchodní prodejce techniky různého druhu je to však funkce velice podstatná. Funkce se bude velice podobat funkci evidence objednávek. V pohledu Reklamace v hlavním okně aplikace bude
možné znovu provádět standardní CRUD operace a spolu s nimi funkce jako tisk reklamačního protokolu, odeslání emailu o vyřízení reklamace, hromadná změna stavu a export.

Reklamační protokol a vzhled odesílaného emailu budou definovány pomocí šablon. Šablony budou uživatelsky editovatelné pomocí voleb v nastavení, viz dále.

V rámci reklamace bude nutné evidovat fakturační a dodací adresu zákazníka, jeho kontaktní údaje, krátký identifikátor reklamace a poznámku. Tato pole budou dostupná v horní části okna formuláře. Spodní část okna bude vyhrazena dvěma oddílům pro definici přijatého a vydaného zboží v rámci reklamace. V levé části bude zboží k reklamaci přijaté, v pravé poté zboží z reklamace vydané. Tyto položky bude možné do reklamace přidávat buď ručně, nebo na základě předmětů evidovaných v naší webové aplikaci. Celou reklamaci bude také možné vytvořit na základě nějaké již existující objednávky. V tomto případě se do reklamace přejme adresa s kontaktními údaji zákazníka a položkami, nebo z evidovaného zákazníka a v tomto případě se přejme fakturační a doručovací adresa a jeho kontaktní údaje.

Obrázek 4.6: Návrh okna detailu reklamace

Na obrázku 4.6 vidíme, jak by mělo ve výsledku vypadat okno pro reklamaci zákazníka. V jeho horní části nalezne volby specifické pro aktuálně otevřenou reklamaci a těmi jsou Tisk (například pro tisk reklamačního protokolu), Zákazník pro výběr zákazníka, Email pro odeslání informačního emailu a Historie pro zobrazení dříve prováděných úkonů v reklamaci.
4.2.6 Import objednávek

Import objednávek je jednou z nejdůležitějších funkcí v navrhované aplikaci, jelikož právě ta dokáže velice usnadnit práci spojenou s prodejem zboží. Pod importem objednávek si můžeme představit funkci, která z prodejních portálů, kterých prodejce ke svému prodeji využívá, stáhne objednávky, data a informace o platbách do naší webové aplikace. Takovým prodejním portálem může být například výše zmínovaný Aukro.cz. Na tomto portálu má prodejce, uživatel naší aplikace, svůj profil, a pokud zákazník zakoupí některý z jeho vystavených předmětů, importem objednávek se tato informace přenesete do námi navrhované webové aplikace. Na obrázku 4.7 naleznete diagram interakce reprezentující způsob stahování objednávek a dat zákazníků z prodejního portálu.

![Diagram interakce pro import objednávek](image)

Obrázek 4.7: Diagram interakce pro import objednávek

V následující podkapitole si navrhneme konkrétní způsob importu objednávek pro jeden z prodejních portálů a tím bude Aukro.cz. Import z dalších navrhovaných prodejních řešení, jako je například e-shopový systém OpenCart, se bude řešit obdobně.
4.2.6.1 Specifikace případů užití

Jelikož se bude jednat o jednu z komplikovanějších operací v našem informačním systému, při níž může docházet k chybám, využijeme jednu z modelovacích technik jazyka UML a tuto operaci si podrobně popíšeme ve specifikaci případů užití. Specifikace tohoto konkrétního případu užití vychází z diagramu případů užití v kapitole 2.

<table>
<thead>
<tr>
<th>ID:</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Název:</td>
<td>Stažení nových objednávek z prodejního portálu</td>
</tr>
<tr>
<td>Popis:</td>
<td>Informační systém stáhne z prodejního portálu data objednávek do vlastní databáze</td>
</tr>
<tr>
<td>Primární účastníci:</td>
<td>Prodejce</td>
</tr>
<tr>
<td>Sekundární účastníci:</td>
<td>Prodejní portál</td>
</tr>
<tr>
<td>Předpoklady:</td>
<td>Žádné</td>
</tr>
</tbody>
</table>
| Následné podmínky: | 1. Do databáze systému jsou vloženy nové objednávky z prodejního portálu  
2. Systém zobrazí seznam nových objednávek |
| Akce pro spuštění: | Prodejce zvolí Stáhnout nové objednávky |
| Hlavní tok: | 1. Systém provede přihlášení k prodejnímu portálu.  
2. Systém provede dotaz na nové objednávky a data zákazníků.  
3. Prodejní portál vráťí informace o nově přijatých objednávkách a zákaznících.  
4. Systém vloží přijatá data do databáze.  
5. Systém zobrazí seznam evidovaných objednávek. |
| Alternativní toky: | Žádné |
| Výjimky: | Storno operace  
Selhání přístupu do databáze  
Selhání komunikace s prodejním portálem |
| Frekvence: | Zpravidla denně, dle potřeby |
| Speciální požadavky: | Žádné |

Tabulka 4.1: Specifikace případu užití pro import objednávek

Jak už bylo zmíněno v odstavci výše, při případu užití může docházet k chybám. Některé možné chyby si popíšeme výjimkami z případu užití a to rovněž v konvenci jazyka UML.

<table>
<thead>
<tr>
<th>ID:</th>
<th>1.E.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Název:</td>
<td>Stažení nových objednávek z prodejního portálu: Storno operace</td>
</tr>
<tr>
<td>Popis:</td>
<td>Prodejce ukončí případ použití</td>
</tr>
<tr>
<td>Primární účastníci:</td>
<td>Prodejce</td>
</tr>
<tr>
<td>Sekundární účastníci:</td>
<td>Žádní</td>
</tr>
<tr>
<td>Předpoklady:</td>
<td>Žádné</td>
</tr>
</tbody>
</table>
| Následné podmínky: | 1. Databáze systému zůstává v původním stavu  
2. Seznam nových objednávek není zobrazen |
| Akce pro spuštění: | Prodejce zvolí v hlavním toku případu užití Žrušit |
Tabulka 4.2: Výjimka z případu užití 1

<table>
<thead>
<tr>
<th>ID:</th>
<th>1.E.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Název:</td>
<td>Stažení nových objednávek z prodejního portálu: Selhání přístupu do databáze</td>
</tr>
<tr>
<td>Popis:</td>
<td>Systém neotevře spojení s databází nebo spojení ztratí</td>
</tr>
<tr>
<td>Primární účastníci:</td>
<td>Prodejce</td>
</tr>
<tr>
<td>Sekundární účastníci:</td>
<td>Žádní</td>
</tr>
<tr>
<td>Předpoklady:</td>
<td>Při přístupu k databázi došlo k chybě</td>
</tr>
</tbody>
</table>
| Následné podmínky: | 1. Nové objednávky nebyly staženy  
2. Seznam nových objednávek není zobrazen |
| Akce pro spuštění: | Selhání přístupu do databáze ve 4. nebo 5. kroku hlavního toku případu užití. |
| Tok: | 1. Systém informuje prodejce o selhání přístupu do databáze  
2. Systém se vrátí do stavu před spuštěním případu užití |
| Frekvence: | Zřídka |

Tabulka 4.3: Výjimka z případu užití 2

<table>
<thead>
<tr>
<th>ID:</th>
<th>1.E.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Název:</td>
<td>Stažení nových objednávek z prodejního portálu: Selhání komunikace s prodejním portálem</td>
</tr>
<tr>
<td>Popis:</td>
<td>Systém se nedokáže spojit s prodejním portálem nebo obdrží chybnou odpověď</td>
</tr>
<tr>
<td>Primární účastníci:</td>
<td>Prodejce</td>
</tr>
<tr>
<td>Sekundární účastníci:</td>
<td>Prodejní portál</td>
</tr>
<tr>
<td>Předpoklady:</td>
<td>Při komunikaci s prodejním portálem došlo k chybě</td>
</tr>
</tbody>
</table>
| Následné podmínky: | 1. Nové objednávky nebyly staženy  
2. Seznam nových objednávek není zobrazen |
| Akce pro spuštění: | Selhání komunikace s prodejním portálem |
| Tok: | 1. Systém informuje prodejce o selhání přístupu do databáze  
2. Systém se vrátí do stavu před spuštěním případu užití |
| Frekvence: | Zřídka |

Tabulka 4.4: Výjimka z případu užití 3

4.2.7 Import bankovních plateb

Import bankovních plateb, stejně jako import objednávek, spočívá v napojení navrhovaného systému na jiný systém třetí strany. V tomto případě to bude internetové bankovnictví některého z vybraných bankovních subjektů. Napojení na systém internetového bankovnictví bude spočívat ve využití API bankovního systému a stažení příchozích plateb.
Systém provede připojení k systému internetového bankovnictví a provede přihlášení uživatele, dále z vlastní databáze načte identifikátor poslední importované platby a skrze API vyhledá platby novější než je poslední zaznamenaná. Informace o platbách uloží do vlastní databáze a bezpečně se odhlásí ze systému elektronického bankovnictví. Z bezpečnostních důvodů se v navrhovaném informačním systému nebude ukládat heslo pro přihlášení do bankovnictví. Při každém importu je uživatel ručně zadá.

Importované platby se po importu automaticky spárují s přijatými objednávkami. Pro párování bude použitý variabilní symbol platby, jméno odesílatele nebo zpráva pro příjemce platby. V systému bude dále přístupná funkce pro zpracování objednávek na základě stavu zaplacení. Jakmile budou k objednávce připárovány platby s celkovou hodnotou stejnou nebo vyšší než je celková cena objednávky, objednávka může změnit svůj stav na K odeslání nebo podobný, dle definice uživatele.

4.2.8 Uživatelé systému

Aby navrhovaný informační systém byl pro prodejce v ůbec použitelný, je samozřejmě nezbytné mít možnost registrovat do systému nové prodejce. Účet prodejce v naší webové aplikaci bude dvojího druhu, a to účet v bezplatné zkušební verzi, nebo účet placený.

Bezplatná verze aplikace bude pro uživatele omezena pouze časově, a to pravděpodobně na 30 dnů. Po tuto dobu nebude fungování aplikace nijak omezeno a prodejce si tak bude moci vyzkoušet všechny aspekty programu tak, jako by jej používal v jeho placené verzi. Po uplynutí této doby bude uživatel při přihlášení do systému zvána k přechodu na placenou verzi aplikace. K registraci uživatele ve zkušební době nebude nutné uvádět jakékoliv kontaktní údaje, pouze emailovou adresu, heslo a případně jméno, které se bude zobrazovat uživateli při přihlášení.

Placená verze aplikace bude pro uživatele zatížena měsíčními poplatky ve výši přibližně 100 Kč měsíčně. Aplikaci si bude možné předplatit volitelně na 3, 6 nebo 12 měsíců dopředu a po tuto dobu bude aplikace poskytovat plnou funkčnost. Při přechodu z bezplatné verze programu nebo při registraci nového platícího zákazníka bude kromě emailové adresy a hesla vyžadována také jeho fakturační adresa a telefonní kontakt.

Předmětem této práce bude pouze implementace systému v plném rozsahu tak, jak by ji používal uživatel v plné verzi aplikace. Vytvoření bezplatné zkušební doby je jedním z cílů v budoucím rozvoji projektu.

4.3 Databáze

<table>
<thead>
<tr>
<th>Název entitní množiny</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>BANK_ACCOUNT</td>
<td>Reprezentuje evidované bankovní účty, Na bankovní účty se vážou bankovní platby a objednávky.</td>
</tr>
<tr>
<td>BANK_PAYMENT</td>
<td>Bankovní platby k evidovaným objednávkám. Do této tabulky jsou ukládány automatickým importem.</td>
</tr>
<tr>
<td>BANK_PAYMENT_STATUS</td>
<td>Tabulka obsahující výčet možných stavů bankovní platby. Budou zde přítomny stav jako Přijata, Spárována s objednávkou, Nespárována, atd.</td>
</tr>
<tr>
<td>BUY</td>
<td>Tabulka jednotlivé nákupy u dodavatelů evidované prodejcem.</td>
</tr>
<tr>
<td>CUSTOMER</td>
<td>Reprezentace evidovaných zákazníků.</td>
</tr>
<tr>
<td>ORDER</td>
<td>Reprezentuje objednávky evidované v našem informačním systému, je základem celé aplikace.</td>
</tr>
<tr>
<td>ORDER_DELIVERY</td>
<td>Obsahuje výčet možných způsobů dopravy objednaného zboží (Poštou, Kurýrem, Osobní převzetí, ...)</td>
</tr>
<tr>
<td>ORDER_HISTORY</td>
<td>Obsahuje záznamy o změnách v objednávkách, například změna stavu, přidání čísla zásilky, atd.</td>
</tr>
<tr>
<td>ORDER_PAYMENT</td>
<td>Obsahuje výčet možných způsobů platby objednaného zboží (Předem, Dobírkou, Hotově, ...)</td>
</tr>
<tr>
<td>ORDER_STATUS</td>
<td>Obsahuje výčet možných stavů objednávky (Přijata, K odeslání, Odeslána, ...). Obsahuje parametry jednotlivých stavů. Další parametry se bude řídit importem objednávek i import platě.</td>
</tr>
<tr>
<td>PACKAGE</td>
<td>Nese informace o evidovaných zásilkách k objednávkám a reklamacích.</td>
</tr>
<tr>
<td>PRODUCT</td>
<td>Představuje evidované produkty v nabídce prodejce.</td>
</tr>
<tr>
<td>PRODUCT_BUY</td>
<td>Vazební tabulka mezi předměty a nákupy, řeší M:M problém a reprezentuje v podstatě položky objednávek.</td>
</tr>
<tr>
<td>PRODUCT_ORDER</td>
<td>Vazební tabulka mezi předměty a objednávkami, podobně jako PRODUCT_BUY.</td>
</tr>
<tr>
<td>SERVIS_CUSTOMER</td>
<td>Reprezentuje evidované reklamace zákazníků.</td>
</tr>
<tr>
<td>SERVIS_CUSTOMER_BULK</td>
<td>Sdružuje reklamace zákazníků, čímž umožňuje reklamovat více kusů zboží najednou.</td>
</tr>
<tr>
<td>SERVIS_CUSTOMER_STATUS</td>
<td>Obsahuje možné stavy reklamací zákazníků. Stejně jako u objednávek jsou tyto stavy editovatelné.</td>
</tr>
<tr>
<td>SERVIS_DEALER</td>
<td>Reprezentuje evidované reklamace u dodavatelů.</td>
</tr>
<tr>
<td>SERVIS_DEALER_BULK</td>
<td>Podobně jako SERVIS_CUSTOMER_BULK dovoluje sdružovat reklamace u dodavatelů, čímž rovněž poskytuje možnost reklamovat více kusů zboží najednou.</td>
</tr>
<tr>
<td>SERVIS_DEALER_STATUS</td>
<td>Obsahuje možné stavy reklamací u dodavatelů, stavu budou opět uživatelsky editovatelné.</td>
</tr>
<tr>
<td>SETTING</td>
<td>Tabulka obsahující pouze jediný řádek a nesoucí uživatelské nastavení aplikace. Data v ní budou určovat parametry importu dat z prodejních portálů, nastavení prodejce, atd.</td>
</tr>
<tr>
<td>TEMPLATE_EMAIL</td>
<td>V této tabulce budou uloženy šablony emailů zasílaných zákazníkům při odeslání zboží, vyřízení reklamace, atd.</td>
</tr>
</tbody>
</table>
V této tabulce budou uloženy šablony tiskových formulářů, na jejichž základě se budou tisknout balíkové štítky k zásilkám, reklamační protokoly, souhrny objednávek, atd.

**VIEW**
Tabulka VIEW je zvláštní tabulka, která slouží pro ukládání nastavení pohledů. Pokud například prodejce vyfiltřuje zobrazení objednávek pouze na ty se stavem Přijata, tato informace se uloží do této tabulky a při příštím spuštění aplikace bude filtr stále aktivní.

**VIEW_COLUMN**
Obsahuje uživatelské nastavení sloupců v pohledech. Jak bylo napsáno výše, sloupec v pohledech budou uživatelsky přizpůsobitelné, skrývatelné, atd. Pořadí, atribut viditelnosti a uživatelsky nastavená šířka sloupce bude uložena v této tabulce.

**Tabulka 4.5: Popis entitních množin**

V souvislosti na napojení systému na prodejní portál Aukro.cz budou v databázi modelovány také následující entitní množiny určené k evidenci dodatečných dat právě z tohoto portálu.

<table>
<thead>
<tr>
<th>Název entitní množiny</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHOP_AUKRO_ACCOUNT</td>
<td>Obsahuje uživatelem definované účty na portálu Aukro.cz.</td>
</tr>
<tr>
<td>SHOP_AUKRO_CUSTOMER</td>
<td>Informace o uživatelích prodejního portálu Aukro.cz, tedy zákaznicích.</td>
</tr>
<tr>
<td>SHOP_AUKRO_DEAL</td>
<td>Reprezentuje jednotlivé nákupy provedené kupujícími na Aukro.cz.</td>
</tr>
<tr>
<td>SHOP_AUKRO_ITEM</td>
<td>Ukládá informace o nabídkách, které uživatel systému (prodejce) zveřejnil na portálu Aukro.cz. V těchto nabídkách mohou následně zákazníci vytvářet objednávky.</td>
</tr>
<tr>
<td>SHOP_AUKRO_PAYMENT</td>
<td>Eviduje platby uskutečněné přes platební systém PayU, používaný portálem Aukro.cz.</td>
</tr>
<tr>
<td>SHOP_AUKRO_SELL</td>
<td>Obsahuje jednoduché informace o počtech kusů zboží aktuálně vystaveného v nabídkách na Aukro.cz.</td>
</tr>
<tr>
<td>SHOP_AUKRO_SETTING</td>
<td>Podobně jako tabulka SETTING obsahuje tato tabulka také pouze jediný řádek s nastavením importu objednávek a exportu zboží na Aukro.cz.</td>
</tr>
</tbody>
</table>

**Tabulka 4.6: Popis entitních množin pro práci s Aukro.cz**

Z navržených entitních množin je patrné, že žádná z nich nenese informaci o uživatelích systému. Je to z toho důvodu, že data o uživatelích (prodejcích) budou uložena ve zcela oddělené databázi a také pro data každého uživatele bude vyhrazena speciální databáze.

Řešení v podobě zvláštní databáze pro každého uživatele bylo zvoleno z následujících důvodů. Oproti řešení, kde by byla data všech uživatelů uložena v jedné databázi, tento přístup výrazně zjednodušuje návrh databáze a její výsledné schéma. V případě jedné databáze by pro každou její tabulku bylo nutné specifikovat atribut identifikující uživatele, kterému patří daný záznam, a to
společně s cizím klíčem realizujícím vazbu na tabulku uživatelů. Dalším a hlavním důvodem je budoucí napojení na data ve webové aplikaci prostřednictvím desktopové aplikace.

V případě, že uživateli (prodejce) bude mít zájem o přístup k datům skrze externí aplikace, tuto možnost si bude moci nastavení explicitně povolit. Jejím povolením dojde k vytvoření nového databázového uživatele, jemuž bude povolen přístup z veřejného síťového hostingu a jemuž bude přiděleno oprávnění nakládat pouze s vlastní databází. Skrze tohoto uživatele se následně prodejce může do systému připojit prostřednictvím desktopové aplikace. Tímto bude prodejce odstíněn od dat jiných uživatelů webové aplikace, což primárně zvýší bezpečnost uložených informací.

4.4 Diagram tříd


V první řadě je to třída `Payment`, ze které je teprve zděděna třída `BankPayment` odpovídající stejnojmenné tabulce v databázi. Tato třída je v aplikaci přítomna z důvodu plánovaného napojení na různé platební brány, elektronické peněženky, atd. Každá nově implementovaná platební metoda poté bude implementovat třídu zděděnou právě z této mateřské třídy Payment.

Další změnou je rozhraní `Bankimport` a jeho realizace nazvaná `FioBankImport`. Tato třída slouží k importu plateb ze systému internetového bankovnictví, konkrétně pro FIO Banku.

Poslední změnou je přidané rozhraní `Shop`, jehož realizace nám reprezentuje prodejní portál. Pomocí dvou metod `ExportProducts()` a `GetNewOrders()` bude možné exportovat produkty z nabídky na prodejní portál, respektive stahovat z portálu nově objednané objednávky. V tomto projektu je implementována jedna jeho realizace a tou je třída `ShopAukro`, která reprezentuje prodejní portál Aukro.cz.

Diagram tříd reprezentuje pouze doménový model navrhované aplikace, nezaměřuje se na třídy spojené s obsluhou transakcí, zobrazení atd.
Kapitola 5

Implementace

Kapitola pátá s názvem Implementace nám popíše některé zajímavé pasáže implementace systému. V kapitole si nejpře probereme strukturu projektu, dále se zaměříme na způsob implementace grafického rozhraní, přičemž si představíme použitý framework a jeho rozšíření. Následovat bude podkapitola o implementaci správy uživatelů a rovněž představení frameworku řešícího tuto problematiku. Dále si ukážeme, jak jsou v systému obecně řešeny jednotlivé agendy a představíme si také přehled pro prodeje, tzv. dashboard. Poslední podkapitola se bude věnovat některým zajímavým a složitějším částem implementace. Nelze popsat všechny implementované detaily, proto se zaměříme výhradně na tyto zajímavé řešení úseky. Jejich příkladem může být implementace úloh běžících na pozadí, řešení problému tisku prostřednictvím jednoúčelové aplikace na klientovi nebo generování emailových zpráv a popisů zboží na základě textových šablon.

5.1 Struktura projektu

Pro implementaci systému byla zvolena technologie ASP.NET MVC, jejíž použití již také předurčuje implementaci projektu v prostředí Microsoft Visual Studio, konkrétně ve verzi 2013. Toto vývojové prostředí je se samotnou technologií ASP.NET velice úzce provázané, z čehož lze při samotné implementaci vytěžit mnoho výhod. Již samotné založení nového ASP.NET MVC projektu nám vytvoří základní strukturu složek a vzorových modelů, pohledů a kontrolérů. Strukturu projektu je samořejmě možné libovolně modifikovat, nicméně pokud se jí rozhodneme držet, Visual Studio a samotné ASP.NET nám některé další činnosti výrazně usnadní. Příkladem může být široké použití reflexe [16], jmenovitě například vyhledání požadovaného kontroléru ve složce Controllers dle adresy požadavku nebo vyhledání pohledu dle názvu kontroléru a zvolené akce ve složce Views bez nutnosti explicitně zadávat plnou cestu.

V následující tabulce si stručně popíšeme všechny součásti projektu v pořadí, v jakém je vidíme na obrázku výše.

<table>
<thead>
<tr>
<th>Název součásti</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>My Project</td>
<td>Konfigurační soubory projektu.</td>
</tr>
<tr>
<td>App_Data</td>
<td>Složka obsahuje databázi uživatelů systému (prodejců).</td>
</tr>
<tr>
<td>App_Start</td>
<td>Složka s konfiguračními skripty aplikace, spouštěnými po startu webového serveru.</td>
</tr>
<tr>
<td>Controllers</td>
<td>Složka obsahující třídy kontrolérů v našem systému. V této složce se implicitně hledají kontroléry při příjmu požadavku.</td>
</tr>
<tr>
<td>Fonts</td>
<td>Složka obsahující vektorové definice fontů použitých v grafickém rozhraní aplikace.</td>
</tr>
<tr>
<td>Helpers</td>
<td>Speciální třídy určené ke generování částí HTML kódu v pohledech [17].</td>
</tr>
<tr>
<td>Models</td>
<td>Třídy modelu, které se nenacházejí v odděleném projektu Model. Jsou to například třídy reprezentující uživatele aplikace.</td>
</tr>
<tr>
<td>Plugins</td>
<td>Především rozšíření klientské části aplikace napsaná v jazyce JavaScript.</td>
</tr>
<tr>
<td>Scripts</td>
<td>Knihovny JavaScriptu vytvořené pro klientskou část aplikace.</td>
</tr>
<tr>
<td>Styles</td>
<td>Sada CSS stylů pro definici celkového vzhledu aplikace.</td>
</tr>
<tr>
<td>Tasks</td>
<td>Obsahuje třídy reprezentující úlohy zpracovávané na pozadí systému. Příkladem je import objednávek, odesílání emailů nebo inicializace databáze.</td>
</tr>
<tr>
<td>ViewModels</td>
<td>Třídy tzv. ViewModelů, jejichž instance reprezentují data zobrazovaná uživateli na konkrétní webové stránce, viz [18].</td>
</tr>
<tr>
<td>Views</td>
<td>Složka obsahující pohledy (z terminologie MVC), tedy HTML kód zobrazovaný uživateli.</td>
</tr>
</tbody>
</table>
Tabulka 5.1: Struktura projektu

<table>
<thead>
<tr>
<th>Název</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>favicon.ico</td>
<td>Ikona aplikace.</td>
</tr>
<tr>
<td>Global.asax</td>
<td>Hlavní třída ASP.NET aplikace, zde probíhá volání inicializačních skriptů.</td>
</tr>
<tr>
<td>package.config</td>
<td>Konfigurační soubor pro NuGET balíčkovací systém také používaný v tomto projektu.</td>
</tr>
<tr>
<td>Startup.vb</td>
<td>Konfigurační skript pro autentizaci uživatelů systémem ASP.NET Identity.</td>
</tr>
<tr>
<td>Web.config</td>
<td>Konfigurační soubor webové aplikace.</td>
</tr>
</tbody>
</table>

5.2 Vzhled aplikace

V širokém spektru internetových aplikací současného internetového prostředí je vzhled aplikace jedním z hlavních atributů, který může zákazníka přilákat a stejně tak dobře odradit od používání aplikace. Nyní však k samotné implementaci. ASP.NET MVC od základů změnilo způsob, jakým jsou webové stránky uživateli generovány. Dřívější technologie WebForms obsahovala spoustu předpřipravených grafických komponent, které bylo možné jednoduše v grafickém editoru nakládat na webovou stránku. Všechny komponenty se až při požadavku na server konvertovaly do odpovídajícího HTML kódu a příslušných knihoven JavaScriptu a kódu v CSS. Toto řešení přinášelo velice rychlou tvorbu bohatého uživatelského rozhraní, které však bylo velmi těžké nějakým způsobem programově graficky přizpůsobit. Podobně je na tom nyní například technologie Java Server Faces z balíku Java EE.

ASP.NET MVC se od tohoto způsobu odprošťuje a vrací se k základům tvorby webových stránek. Webová stránka, neboli pohled v terminologii MVC, je zpravidla čistý HTML kód. Dynamické chování webových stránek na straně klienta pak zajišťuje podpůrné knihovny JavaScriptu a vzhled definují CSS styly, které si programátor sám zvolí. Toto řešení je sice náročnější a zdlouhavější na programování, nicméně zaručuje mnohem větší průhlednost kódu a prakticky neomezené možnosti grafického přizpůsobení stylu aplikace.

Tvořit dnes celé grafické rozhraní svépomocí by však vyžadovalo nemalé úsilí a pro rozsáhlejší projekty stovky hodin práce. ASP.NET MVC se s tímto vypořádává díky podpoře použití různých grafických webových frameworků. Jedním z takových frameworků je například Bootstrap [19] používaný i v tomto projektu.

5.2.1 Bootstrap a jQuery

Bootstrap je frameworkem určeným k tvorbě klientské části webové aplikace. Jedná se o knihovnu JavaScriptu a CSS stylů pro vytváření uživatelského rozhraní aplikace. Bootstrap je úzce provázaný se samotným HTML kódem webové stránky, kdy vzhled jednotlivých komponent se určuje na základě atributů class jednotlivých HTML elementů. Obrovskou výhodou v použití právě tohoto frameworku je důraz na mobilní zobrazení aplikace. Pomocí frameworku je možné definovat vzhled pro různé šířky webového prohlížeče a tak webovou stránku přizpůsobovat pro zobrazení...
počítači, tabletu či mobilním telefonu. To celé v rámci jedné webové stránky bez nutnosti interakce s webovým server.

Další podstatnou součástí projektu je knihovna jQuery, která zastává spoustu rutinních činností při programování klientské části aplikace v jazyce JavaScript. Exemplárním příkladem pomoci knihovny je například získávání dat ze serveru pomocí technologie AJAX. Namísto nutnosti konstruovat objekt XMLHttpRequest [20], předávat mu všechny potřebné parametry a ošetřovat všechny stavky, které mohou nastat v průběhu zpracování požadavku, použijeme knihovní funkci $.ajax(). Parametrem pouze předáme adresu, parametry požadavku a definujeme chování v případě úspěchu a neúspěchu. Samotná práce se nám výrazně zjednoduší. Nad zmiňovanou knihovní funkcí je v tomto projektu implementována knihovna Task.js, sloužící pro zpracovávání akcí běžících na pozadí. Tuto knihovnu si blíže představíme v kapitole 5.6.1, jelikož se jedná o jednu z velice zajímavých částí projektu.

5.2.2 Použitá rozšíření

Bootstrap framework nám sám o sobě poskytuje bohatý nástroj pro vytvoření kompletního vzhledu aplikace. Pokud však půjdeme hlouběji do funkčnosti klientské části aplikace, narazíme na problémy, jejichž řešení nám samotný Bootstrap nenabídne. V této situaci připadá v úvahu použití rozšíření třetích stran, které nám klientskou část aplikace o požadovanou funkčnost doplní. Funkčnost, kterou nám samotný Bootstrap framework zajistit nedokáže, jsou například tabulky podporující řazení a filtrování, textové pole pro vyběr data a další. V této podkapitole si popíšeme rozšíření, která jsou v projektu použita.

5.2.2.1 Bootstrap Table

Ač se může zdát, že se jedná o součást výše zmiňovaného Bootstrap frameworku, opak je pravdou. Projekt Bootstrap Table je zcela oddělenou součástí, pouze určenou pro použití se zmiňovaným frameworkem.

Jedná se o rozšíření pro klientskou část webové aplikace, které dodává dynamickost běžným HTML tabulkám. Rozšíření se skládá z JavaScript knihovny a sady CSS stylů, které nad běžnými tabulkami implementují tuto funkčnost. Výsledné řešení přidává do tabulek možnost řadit zobrazená data, filtrovat je pomocí textového vstupu od uživatele, stránkovat tabulky s mnoha řádky, zobrazovat nebo skrývat sloupce a formátovat v ních zobrazené údaje. Podstatnou součástí je možnost získávání data na žádost uživatele pomocí technologie AJAX nebo vybírat řádky pomocí zaškrťovacích polí. [21]

Tato funkce je v projektu využita při zobrazování všech tabulkových výpisů položek jednotlivých agend. Více o implementaci výpisů v agendách se dočteme v kapitole 5.4.
5.2.2.2 Bootbox

Bootbox rozšiřuje front-end webové aplikace o zobrazování jednoduchých dialogových oken uživateli. Již samotný Bootstrap dovoluje zobrazování dialogových oken, toto rozšíření se však zaměřuje na zobrazování zcela jednoduchých informativních nebo dotazovacích dialogů. Takovéto dialogy je pak možné zobrazit jediným voláním funkce v JavaScriptu, bez nutnosti definovat další HTML kód. [22]

![Obrázek 5.2: Vzhled dialogového okna rozšíření Bootbox](image)

Bootbox dialogová okna jsou v projektu použita například při potvrzení smazání položky, při požadavku na vstup od uživatele v průběhu importu dat z prodejního portálu a podobně.

5.2.2.3 pace

Rozšíření s názvem Pace přidává funkčnost, která jen a pouze zlepšuje uživatelský zážitek z aplikace. Funkčnost tohoto rozšíření spočívá v zobrazení nahrávacího dialogu uživateli po dobu, kdy se součásti aplikace teprve načítají do webového prohlížeče.

Většina internetových prohlížečů ve snaze působit co nejvíce svížně zobrazuje uživateli webové stránky již v průběhu jejich načítání. U složitějších aplikací to má za následek postupné naplňování zobrazené stránky uživatelskými prvky a jejich následné automatické přemístování či transformování dle aktuálně načtených CSS stylů a knihoven JavaScriptu. Toto chování webové stránky ne příliš přídává na uživatelské přívětivosti aplikace, a proto existují rozšíření jako právě Pace, která načítání a transformace prvků stránky před uživatelem skryjí. [23]

Implementačně se prakticky jedná o malou knihovnu JavaScriptu a malé sady CSS stylů pro zobrazení průběhu načítání. Inkluze této knihovny do dokumentu stránky by měla být provedena co možná nejvýše v kódu HTML. Je to z toho důvodu, aby prohlížeč co nejdříve provedl její zpracování, zobrazil požadovaný indikátor průběhu a před uživatelem dočasně skryl prvky webové stránky.
5.2.2.4 Bootstrap Datepicker


Obrázek 5.3: Rozšíření Bootstrap DatePicker v zobrazení objednávky

V tomto projektu je prvek použit ve většině formulářů, kde se nachází vstupní pole pro zadávání data.

5.2.2.5 Morris.js

Rozšíření Morris.js je v projektu použité pouze na jediném místě a to v dashbordu, tedy na úvodní stránce. Toto rozšíření implementuje opět pomocí JavaScriptu a CSS zobrazení grafu. V dashbordu zobrazuje obraty, tedy sumu nákupů a objednávek za dané období.

Oproti jiným konkurenčním řešením vyniká knihovna Morris.js jednoduchostí zobrazených grafů a to s důrazem na grafické ztvárnění. Velice dobře tak zapadne do celkového grafického stylu aplikace. Umožňuje zobrazit klasické spojnicové grafy, ale také sloupcové či koláčové typy grafů. Zobrazované grafy mohou být responzivní, tedy reagující na pohyb myši nad nimi, a tím zobrazující dodatečné informace pro uživatele.

5.2.3 Grafická šablona

Po několika pokusech vytvořit design webové aplikace vlastními silami bylo přistoupeno na variantu použití některé z existujících grafických šablon. Na internetu lze najít mnoho povedených grafických šablon, které dodají aplikaci profesionální vzhled, jehož vytvoření v takovém rozsahu, jaký poskytuje hotová šablona, by vyžadovalo desítky až stovky hodin práce a zejména grafické tvůrčí nadání.

Grafické ztvárnění aplikace vychází webové šablony s názvem Nifty vybrané v internetovém katalogu Bootswrap, viz [25]. Tento a další internetové katalogy nabízejí stovky šablon vhodných

Dle srovnávací tabulky vycházejí nejlépe grafická téma s názvy Nifty, BePro a Lander v2. Konečný výběr tedy rozhodne především celkový dojem z grafického ztvárnění šablony a ten subjektivně nejlépe splňuje grafické téma Nifty.

<table>
<thead>
<tr>
<th></th>
<th>Light Blue</th>
<th>Ace Admin</th>
<th>Nifty</th>
<th>BePro</th>
<th>Lander v2</th>
<th>Prism</th>
<th>King Admin</th>
<th>Beyond</th>
<th>Centaurus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export dat z tabulky</td>
<td>❌</td>
<td>✔️</td>
<td>✔️</td>
<td>❌</td>
<td>✔️</td>
<td>❌</td>
<td>✔️</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Vícečádrové výběr v tabulce</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>❌</td>
<td>✔️</td>
<td>✔️</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Scrollable dialogy</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>❌</td>
<td>✔️</td>
</tr>
<tr>
<td>Vyhledávací box</td>
<td>✔️</td>
<td>❌</td>
<td>✔️</td>
<td>❌</td>
<td>❌</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Nahrávací obrazovka</td>
<td>✔️</td>
<td>❌</td>
<td>❌</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>❌</td>
</tr>
<tr>
<td>Full-screen tabulky v mobilním zobrazení</td>
<td>❌</td>
<td>❌</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Zdroj</td>
<td>[1]</td>
<td>[1]</td>
<td>[2]</td>
<td>[2]</td>
<td>[3]</td>
<td>[3]</td>
<td>[1]</td>
<td>[1]</td>
<td>[1]</td>
</tr>
</tbody>
</table>

*Tabulka 5.2: Porovnání grafických šablon*

Zdroje:

Grafická šablona je distribuovaná ve formě ZIP archivu, který obsahuje vzorové stránky v čistém HTML kódu a k nim stylové CSS předpisy a nezbytný JavaScript pro některé ovládací prvky. Jelikož se jedná o sadu statických webových stránek, vybraná šablona pomáhá pouze s grafickým ztvárněním aplikace. Veškerou logiku i konkrétní rozložení uživatelských prvků je třeba naprogramovat. Použití hotové grafické šablony bylo zvoleno jen a pouze za účelem dosažení
profesionálního vzhledu výsledného produktu, nikoli z důvodu usnadnění práce, jak by se na první pohled mohlo zdát.


5.2.4 Struktura pohledů

Technologie ASP.NET MVC obsahuje funkci, která dovoluje generování pohledů, tedy webových stránek zobrazovaných uživateli, na základě tzv. layoutů. Layouty (opět z názvosloví ASP.NET MVC) nám definují grafický rámec aplikace, v jehož konkrétní definované části se posléze mění uživatelem požadovaný obsah. Layouty mohou být definovány v hierarchické struktuře, což je také případ naší aplikace. Hierarchii pohledů v aplikaci naznačuje obrázek 5.4.

Obrázek 5.4: Hierarchie pohledů

Jak je patrné z obrázku výše, v aplikaci jsou dva druhy grafického rozložení obrazovky. Jedno grafické rozložení je vytvořeno pro proces registrace a přihlašování do systému, druhé poté pro samotný vzhled aplikace.

Vzhled aplikace je dán rozložením v souboru _Layout.vbhtml, z něhož je dále odvozen pohled _LayoutGrid.vbhtml, který do původního layoutu přidává tabulkový výpis položek zvolené agendy. Pro každou agendu (tedy objednávky, zásilky, reklamace, atd.) je poté definován pouze jednoduchý pohled s názvem Index.vbhtml, který obsahuje pouze informace o tom, jaká data do pohledu vložit.
Toto řešení umožňuje vzhled měnit pouze na jediném místě pro všechny agendy, tím výrazně zlepšuje udržovatelnost kódu a bezesporu naplňuje princip DRY v softwarovém inženýrství [27].

Z výchozího rozložení _Layout.vbhtml jsou odvozena také rozložení pro zobrazení položek jednotlivých agend, tedy rozložení pro zobrazení objednávky, zásilky, reklamace, atd. Pro každou agendu existuje zvláštní pohled určený pro vytváření nové položky v agendě, nazvaný Index.vbhtml, a pohled pro editaci existující položky nazvaný Edit.vbhtml. Oba tyto pohledy však využívají společnou část v podobě definice vstupních polí a jiného obsahu ze souboru Content.vbhtml a sami definují pouze specifické odlišnosti pro vytváření resp. upravování položek agendy.

Zvláštní rozložení je definováno pro přihlašovací a registrační obrazovku v systému. V rámci zjednodušení a zpřehlednění procesu registrace a přihlašování bylo vhodné vytvořit rozložení specifické pouze pro tyto činnosti. Rozložení reprezentuje soubor s názvem _LayoutLogin.vbhtml, ze kterého jsou odvozeny pohledy Login.vbhtml a Register.vbhtml pro přihlašování, resp. registraci do systému.

Obrázek 5.5: Přihlašovací obrazovka systému

5.3 Správa uživatelů

Správa a zabezpečené uživatelských účtů je kamenem úrazu mnoha volně dostupných informačních systémů. Je třeba zajistit nejen samotné zapsání uživatelů do databáze systému a následné ověřování přihlašovacích údajů, ale také zabezpečit celý přenos citlivých údajů od uživatele a jejich bezpečné uložení. Trendem poslední doby je poté přihlašování k webovým aplikacím skrze aplikace třetích stran, jako jsou například různé sociální sítě. Přihlašováním přes
aplikace třetích stran vzniká v systému další potenciální slabina v zabezpečení, o níž se vývojář musí postarat.

Naštěstí však existují některá hotová komplexní řešení, která dokážou výše zmíněné problémy vyřešit takřka zcela ve své režii. Jedním z takových řešení je framework ASP.NET Identity, použitý i v tomto projektu.

5.3.1 ASP.NET Identity

ASP.NET Identity je nástupcem technologie ASP.NET Membership [28] z roku 2005, která sloužila ke správě uživatelů ve webových aplikacích založených na technologií WebForms. Postupem času a se stoupající provázaností internetových aplikací právě se sociálními sítěmi se tento framework stal limitujícím a společnost Microsoft přišla s řešením v podobě nového frameworku s názvem Identity. [32]


![Obrázek 5.6: Schéma databáze pro správu uživatelů](image)

Co se týká samotného zabezpečení, framework se stará o ukládání hesla v podobě hashe, podporuje ověření zadaných uživatelských údajů skrze SMS bránu či e-mail, implementuje protokol...
OAuth pro přihlašování skrze služby třetích stran, blokování přístupu po několika neúspěšných přihlášeních, atd. [29]

Informace o uživatelích systému, tedy o prodejcích, jsou v systému uložena odděleně od uživatelských dat, je pro ně vyhrazena zvláštní databáze. V našem případě posloužila pro ukládání dat o uživatelích databáze MSSQL Express, standardně dodávaná s instalací Visual Studia. Výsledné schéma databáze pro správu uživatelů vidíme na obrázku 5.7.

Zabezpečení přenosu citlivých informací mezi webovým prohlížečem uživatele a informačním systémem zajišťuje šifrovaný SSL kanál.

5.3.2 Proces registrace uživatelů


Obrázek 5.7: Uživatel přihlášený v systému

5.4 Agendy v systému

Položky jednotlivých agend evidovaných ve webové aplikaci, tj. objednávky, zásilky, reklamace, zboží, nákupy, zákazníci a platby, jsou implementovány velice podobným způsobem. To zajišťuje jednoduchost práce se systémem napříč agendami. Kliknutím na název agendy v levém postranním panelu se přechází na tabulkový výpis položek evidovaných v rámci této agendy. Tabulkový výpis je patrný z obrázku 5.8.

Podobný styl výpisu se používá v mnoha dalších podobných systémech a uživatelé s nimi pracující jsou na něj úspěšně přiznáni. V horní části prostoru pro tabulkový výpis je pro každou agendu uveden její název a pod ním navigační lišta, tzv. breadcrumb. [30] Největší část okna však vyplňuje zmiňovaný výpis položek. Tabulkový výpis zajišťuje webové rozšíření Bootstrap Table, více o tomto rozšíření je uvedeno v kapitole 5.2.2.1. Pro účely této aplikace bylo rozšíření lehce přizpůsobeno. Na řádky tabulky lze nyní kliknut a fungují jako odkaz na konkrétní objednávku. Řádky tabulky je možné označovat pomocí zaškrtnavacích polí a pomocí horní tlačítkové
lišty s nimi hromadně pracovat. Pokud není označen žádný řádek, tlačítka v horní liště zůstávají neaktivní, což lépe informuje uživatele o aktuálně proveditelných akcích.

Obrázek 5.8: Tabulkový výpis položek agendy

Data do tabulky jsou získávána pomocí technologie AJAX, nejsou tedy součástí webové stránky již při načtení. Tím je umožněno například provádět refresh dat v tabulce bez nutnosti nahrávat celou stránku.

Rozšíření Bootstrap Table nabízí mnoho funkcí, z nichž některé jsou použité i v tomto projektu. Zejména se jedná o vyhledávací pole, které na základě uživatelem zadaného textu vyhledává mezi aktuálně zobrazenými objednávkami. Dále je to možnost řádky tabulky konvertovat do seznamového zobrazení, které je zvláště vhodné pro malé displeje mobilních telefonů. V projektu byl použit i uživatelský výběr sloupců a stránkování tabulky.

Obrázek 5.9: Zobrazení položky agendy
Kliknutím na řádek tabulky se přechází na zobrazení konkrétní položky, které můžeme vidět na obrázku 5.9. Zobrazení položky agény využívá stejné rozložení jako tabulkový výpis, pouze se liší svým obsahem. Horní část středového prostoru opět obsahuje název aktuálně zobrazené položky a pod ním navigační lištu. Následuje lišta tlačítek pro práci s aktuálně zobrazenou položkou, přičemž v její levé části jsou umístěny volby pro uložení a návrat zpět, na pravé straně jsou to již akce pracující v dané položce.

Obrázek 5.10: Položky objednávky

Nyní si vezměme pro příklad objednávku, jakožto položku z agény objednávek. Pro zobrazení objednávaného zboží je využita opět tabulka implementovaná rozšířením Bootstrap Table. Zajímavým způsobem bylo nutné vyřešit naplnění tabulky daty, tedy informacemi o objednaném zboží a rovněž předávání těchto informací zpět serveru při uložení objednávky. Za tímto účelem je ve formuláři přítomno skryté textové pole, v němž jsou informace o všech objednaných položkách uloženy ve formátu JSON. Po načtení stránky se text uložený v tomto poli naparsuje do podoby kolekce objektů představujících objednané zboží a ta předá tabulce pro zobrazení. Při uložení objednávky, případně při jiné akci, se kolekce z tabulky zpět serializuje do formátu JSON, uloží se do avizovaného skrytého pole a je odeslána jako standardní položka formuláře. Tímto řešením se také předchází problému s funkcí model binder, popsaném v kapitole 3.1.1.

Editace položek objednávky probíhá pouze na straně klienta, za pomocí modálních oken z Bootstrap frameworku. Na obrázku 5.10 vidíme výpis položek objednávky a na obrázku 5.11 příklad editace jedné položky.

Obrázek 5.11: Editace položky objednávky
5.5 Dashboard

Jednou z obrazovek informačního systému je dashboard. Je to první obrazovka, kterou uživatel uvidí po vstupu do systému a slouží k poskytnutí základního přehledu o aktuálním stavu prodejů.

"A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance.” [34]

Jak už bylo zmíněno v návrhu, webová aplikace cílí především na malé internetové prodeje všech kategorií. Z tohoto důvodu by i dashboard měl být jednoduchý, na první pohled srozumitelný a měl by poskytovat pouze takovou škálu informací, ve které se lze rychle zorientovat. Ukázkou dashboardu z implementované webové aplikace naleznete na obrázku 5.12.

Obrázek 5.12: Dashboard

Hlavní část dashboardu zaujímá graf nákupů a prodejů v měsících za poslední rok. Pod ním lze nalézt vývoj počtu objednávek za stejné období a počet objednávek realizovaných zákazníky v aktuálním dni. K přehlednosti a čistotě celého řešení přispívá fakt, že konkrétní hodnoty se zobrazují po přejetí kurzoru myši. Vpravo od grafu jsou umístěny 4 widgety informující o nedokončených činnostech evidovaných v systému. Jsou to informace o počtu nevyřízených objednávek, počtu
neodeslaných zásilek, počtu aktuálně vystavených nabídek na prodejních portálech a počtu nevyřízených reklamací. Na widgety je možné kliknout, přičemž uživatel je přesměrován na výpis položek dané agendy.

Obzvlášť spodní části dashboardu je opět několik widgetů, tentokrát bez funkce odkazu, které dávají základní přehled o finanční stránce obchodů. Jejich obsahem je celkový stav bankovních účtů, celková suma objednávek odeslaných na dobírku, u nichž očekáváme přijetí dobírkové částky, dále průměrný zisk za měsíc a průměrná marže za poslední měsíc.

Oproti dashboardům jiných informačních systémů může tento dashboard působit poněkud stroze a dojmem, že poskytuje malé množství informací. Jednoduchost dashboardu je však primárním účelem.

5.6 Zajímavé části implementace

V této podkapitole si podrobněji popíšeme některé zajímavé části z implementace projektu.

5.6.1 Akce vykonávané na pozadí

Akce jako import objednávek, export zboží nebo i odesilání emailů zákazníkům mohou trvat déle než je odezva systému na běžné požadavky a jejich sekvenční provádění by mohlo mít neblahý vliv na uživatelskou přívětivost systému. Z tohoto důvodu byl v aplikaci navržen a implementován systém pro zpracovávání akcí běžících na pozadí.

Tento systém dovoluje uživateli spustit požadovanou akci, zobrazuje mu její průběh a také informuje o dokončení akce. Specialitou je poté možnost vyžádat si od uživatele vstup v průběhu prováděné akce. Systém se skládá ze serverové a klientské části a blíže si jej popíšeme v této podkapitole.

Klientská část systému pro vykonávání akcí na pozadí je knihovna Tasks.js napsaná v jazyce JavaScript a sada informačních dialogů v jazyce HTML využívající Bootstrap framework. Zmiňovaná knihovna implementuje speciální objekt Start, jehož atribut start() dovoluje zahájit akci vykonávanou na pozadí, přičemž zobrazí modální dialogové okno pro informování uživatele o průběhu. Samotná metoda vyžaduje jediný parametr a tím je objekt specifikující požadavek na prováděnou akci. Tento požadavek zahrnuje adresu pro spuštění úlohy, dále seznam parametrů zasílaných serveru. Jeho součástí je také definice obslužných rutin pro případ úspěchu i neúspěchu prováděné akce a také dvojice informačních textů, které se uživateli po dokončení akce zobrazí v závislosti na stavu dokončení úlohy.

Spuštění následně probíhá zasíláním požadavku na server pomocí technologie AJAX. Následuje periodické dotazování serveru na stav úlohy a po jejím dokončení převzetí výsledku. Přenos dat probíhá ve formátu JSON. Následující obrázek ilustruje sekvenci zasílaných zpráv při provádění akce na pozadí včetně jejich konkrétní podoby.
Obrázek 5.14: Diagram tříd pro akce běžící na pozadí


Správu všech úloh zajišťuje objekt třídy `TaskPerformer`. Tento objekt je v aplikaci přítomen pouze v jedné instanci a obsluhuje tak všechny úlohy všech uživatelů. Jeho úkolem je však pouze přijímat požadavky na nově zpracovávané úlohy a ty řadit do fronty a spouštět, dále na žádost klienta vracet stav požadované úlohy a v případě dokončení opět na žádost vrátit její výsledek a úlohu z fronty odstranit.

5.6.2 Generování emailových zpráv a nabídek zboží

Jednou z vlastností určené už při návrhu webové aplikace je dát prodejci možnost, aby svým zákazníkům zasílal e-mailové zprávy a to jak čistě textově, tak i ve formátu HTML. Další vlastností systému má být generování popisů evidovaného zboží na základě jeho parametrů. Obě tyto činnosti spolu zdánlivě nesouvísí, avšak opak je pravdu.

V obou případech jde o generování dokumentů ve formátu prostého textu nebo HTML na základě dat evidovaných v aplikaci. Pro tento účel byl v aplikaci navržen systém šablak, který...

Aby bylo možné emaily na základě emailových šablon vytvářet, je zapotřebí v emailové šablone definovat místa, do kterých dosadit požadovaný text z evidované objednávky, reklamace, atd. Místa pro vkládání označíme speciálními tagy. Tagy budou celkem třech druhů a všechny si je popíšeme v následujících podkapitolách. Tyto tagy řeší prakticky 3 problémy, které při generování textových zpráv na základě šablony nastávají. Těmito základními problémy jsou:

- Jak do šablony vložit konkrétní informaci?
- Jak určitou část textu šablony zopakovat?
- Jak určitou část textu šablony za určitých podmínek vynechat?
V příloze 4 poté najdeme ukázku celé emailové šablony, jak je použita v reálném nasazení. Nyní si představme jednotlivé tagy podrobněji.

5.6.2.1 Jednoduché tagy

Jednoduchými tagy se rozumí jednoduché textové hodnoty jako je například název zboží, cena objednávky, jméno zákazníka atd. Tyto tagy jsou vždy ohraničeny hranatými závorkami, například \[title\] pro název zboží nebo \[id\] pro číslo objednávky. Seznam veškerých tagů, použitelných pro jednotlivé agendy, najdeme v příloze číslo 3 této práce.

Následuje ukázka použití jednoduchých tagů na příkladu vkládání názvu zboží a jeho krátkého popisu při generování popisu ve formátu HTML.

```html
...<div id="obsah00in">
  <h2>
    Prodám [title]
  </h2>
  <p>
    [parameters:short_description]
  </p>
...```

5.6.2.2 Vkládací tagy

Někdy je zapotřebí některé části šablony ve výsledném dokumentu zobrazit nebo naopak skrýt v závislosti na hodnotě některého atributu. K tomuto slouží speciální tag \casid_insert. Příklad použití vidíme zde:

```html
...<casid_insert ifdef="parameters:description">
  <p>
    [parameters:description]
  </p>
</casid_insert>
...```

Výše uvedený kód nám do výsledného dokumentu vloží odstavec s popisem zboží ([parameters:description]), ale pouze pokud je tento parametr pro dané zboží definován (ifdef="parameters:description"). Není-li parametr description ve zboží přítomen nebo je prázdný, odstavec se do výsledné šablony vůbec nevloží.
5.6.2.3 Opakovací tagy

Pokud potřebujeme určitou část kódu ve výsledném dokumentu opakovat, například pro všechny parametry zboží nebo pro všechny položky objednávky, které chceme vypsat jako jednotlivé řádky tabulky, použijeme speciální tag \texttt{casid\_repeat}. Použití může vypadat podobně jako kód uvedený níže.

\begin{verbatim}
...<table>
 <tbody>
 <casid\_repeat source=\textquotesingle{}parameters\textquoteright{} tag=\textquotesingle{}param\textquoteright{} hideonempty=\textquotesingle{}true\textquoteright{}>
 <tr>
 <td>
 <strong>[name]</strong>
 </td>
 <td style=\textquotesingle{}width: 100%;\textquoteright{}>
 [value]
 </td>
 </tr>
 </casid\_repeat>
 </tbody>
</table>
...
\end{verbatim}

Výše uvedený kód nám do výsledného dokumentu vkládá všechny námi definované parametry zboží (\texttt{source=\textquotesingle{}parameters\textquoteright{}}) označené jako \texttt{param} (\texttt{tag=\textquotesingle{}param\textquoteright{}}). Vkládá je jako jednotlivé řádky HTML tabulky. Konkrétní místo vložení názvu parametru je pak označeno pomocí \texttt{[name]} a místo pro vložení hodnoty jednoduchým tagem \texttt{[value]}.

5.6.3 Tisk dokumentů

Dalším aspektem implementované webové aplikace je tisk. Jedná se zejména o tisk dokumentů souvisejících se zásilkami, jako jsou balíkové štítky, poštovní poukázky, podací archy, atd. Kromě nich je však možné tisknout dokumenty také k položkám jiných agend. V tomto případě se jedná například o faktury k objednávkám či reklamační protokoly k evidovaným reklamacím. Bylo by možné namítnout, že tyto dokumenty je možné generovat stejně jako výše uvedené emailové zprávy a zvláštní zacházení s nimi tedy není zapotřebí. Problémem u tiskových položek je ovšem formát těchto dokumentů, u nichž běžný HTML kód interpretovaný prohlížečem nevyhovuje.

Důvodem je lehce rozdílná interpretace HTML kódu napříč prohlížeči. Tento problém je zcela zanedbatelný například při tisku faktur nebo v reklamačních protokolech. Komplikace však nastává v případě tisku poštovních dokumentů, které zpravidla tiskneme na předem připravené formuláře.
Podívejme se na obrázek 5.16, kde je vyobrazena *Poštovní dobírková poukázka A* jako příklad jednoho z takových předtištěných formulářů.

**Obrázek 5.16: Poštovní dobírková poukázka A**

Z obrázku je patrné, že tisknutí textu jako je číslo účtu, variabilní symbol a další musí být velice přesné proto, aby jednotlivé znaky byly přesně zacíleny do svých jednotlivých polí. Dosáhnutí takové přesnosti je takřka nemožné, pokud je text generovaný pomocí obyčejné HTML šablony a tisknutý na připravený formulář skrze webový prohlížeč. Experimentálně bylo zjištěno, že ne každý prohlížeč interpretuje například rozteč znaků u čísla účtu stejně a prohlížeče se taktéž rozchází v celkovém umístění textových řádků ve vytištěném dokumentu. Rovněž byla zvažována možnost vytvoření dokumentu ve formátu PDF a jeho následný tisk v aplikaci na straně klienta. Tato varianta však také nepřinesla požadované výsledky.

### 5.6.3.1 Princip funkce

Z důvodů popsaných výše bylo upuštěno od snahy využívat k tisku funkce internetového prohlížeče a byl navržen systém tiskových šablón ve formátu XML, pomocí nichž se výsledné dokumenty vytváří. Stejně jako u emailových šablón probíhá generování výsledného dokumentu na straně serveru. Další zpracování je ale od zasílání emailů rozdílné. Po vygenerování dokumentu je výsledný tiskový formulář, také ve formátu XML, zaslán zpět klientovi jako soubor. Uživatel si tento soubor stáhne skrze webový prohlížeč a otevře pomocí speciální malé desktopové aplikace, která je taktéž součástí projektu. Desktopová aplikace následně pouze vezme soubor, vytvoří z něj dokument a ten odešle k tisku. Proces tisku dokumentu ukazuje sekvenční diagram na obrázku 5.17.

Toto řešení se může zdát poněkud komplikované, ale pro přesný tisk do předpřipravených formulářů je opravdu nezbytné. V budoucím nasazení do aplikace přibude možnost tisknout dokumenty, u nichž tato přesnost není vyžadována přímo z okna webového prohlížeče. V současné verzi projektu však tuto funkci neuvažujeme.
5.6.3.2 Struktura XML formuláře

Nyní si pojďme blíže představit strukturu tiskové šablony ve formátu XML. Popsány zde budou všechny elementy, které mohou být obsaženy v tiskové šablone. Pro lepší představu fungování se však nejprve podívejme na obrázek 5.18, který nám graficky znázorňuje strukturu tiskové šablony.


V obrázku 5.18 jsou naznačeny některé typy elementů, které mohou bloky Header, Expander a Footer obsahovat. Výčet všech elementů, jejich možných atributů a přijatelných dceřiných elementů, které mohou být jejich obsahem, naleznete v příloze 3.
5.6.3.3 Zpracování dokumentu na straně klienta

Nyní se pojďme zaměřit na zpracování dokumentu na straně klienta. Jak jsme se dozvěděli výše, do tiskové aplikace přichází archív s uživatelem zvolenou tiskovou šablounou ve formátu XML a datovou reprezentací tiskových položek, například objednávek, rovněž ve formátu XML. Přijetím tohoto archívu dojde k naparsování tiskové šablony do hierarchické struktury objektů, které reprezentují jednotlivé elementy XML stromu. Podívejme se na diagram tříd, ze kterých hierarchie objektů vychází, najdeme jej v příloze 2g. Na obrázku 5.19 vidíme velice zjednodušenou hierarchickou strukturu objektů pro tiskový dokument.

Po naparsování XML dokumentu jsou objekty naplněny daty z přiložených objednávek a následuje procedura tisku. Při tisku se prochází hierarchická struktura objektů od kořenového uzlu a tím je objekt Page. Stromovou strukturou se prochází algoritmem Pre-order. [31] Při průchodu
uzlem je tento uzel vytisknut na interní canvas a pokračuje se na jeho prvního následníka. Po průchodu celého stromu je připravený canvas odeslán k tisku na uživatelem zvolené tiskové zařízení. Všechny tyto operace již probíhají mimo webovou aplikaci, tedy na počítači uživatele pomocí jednoduchého desktopového rozhraní.

Obrázek 5.19: Hierarchická struktura objektů popisující dokument

Zmiňovaná desktopová aplikace má velice spartánské grafické rozhraní, zahrnující pouze volbu pro načtení archívu z aplikace, která je následně okamžitě odeslána na tisk. Zamýšleným rozšířením pro budoucí vývoj je v každém případě integrace aplikace do operačního systému. To zahrnuje registraci přípony .CPR souborů stahovaných prostřednictvím webového prohlížeče do systému a jejich automatické otevírání tiskovou aplikací.

5.6.4 Import a export dat

V této podkapitole si popíšeme, jakým způsobem probíhá napojení implementované webové aplikace na externí systémy. Těmito systémy jsou služby třetích stran jako portály pro nabízení evidovaného zboží nebo systémy internetového bankovnictví.

Problém, který řeší import dat z různých zdrojů, je agregace těchto nehomogenních dat s různou strukturou a uložení do nám implementovaného systému. Data z externích systémů jsou na požadovaný formát transformována procedurálně, tedy přímo v kódu programu. Obecně lze říci, že data z různých zdrojů přicházejí do webové aplikace v tak rozdílných formátech, že vytváří jen nějaký sjednocující adaptér by pouze zvýšilo složitost bez dalšího užitku. Kupříkladu data z internetového bankovnictví přicházejí do aplikace v čistém XML, portál Aukro.cz nabízí přístup ke svým datům prostřednictvím protokolu SOAP a například Česká pošta vyžaduje data o zásilkách v prostém CSV formátu. V rámci budoucího vývoje při rozšířování počtu podporovaných externích systémů bude však vhodně nějakou vhodnou míru abstrakce nad procesy importu a exportu dat v aplikaci implementovat.
Export dat, v našem případě export zboží na prodejní portály, řeší problém opačný. Tedy, kterak zajistit transformaci dat ze struktur používaných v naší webové aplikaci na struktury požadované prodejním portálem. Tato transformace rovněž probíhá procedurálně, stejně jako proces importu popsaný výše.

5.6.4.1 Import objednávek z Aukro.cz

Aukro.cz je prodejní portál, ze kterého navrhovaná aplikace umí stahovat objednávky zákazníků. Základem pro komunikaci s Aukro.cz je jím poskytované webové API, viz [8]. Webové API je vlastně SOAP služba implementovaná nad protokolem HTTP. Po přihlášení do této služby může prodejce spravovat svůj účet stejně jako skrze webové rozhraní, avšak čistě datově orientováno.

Přistupme tedy ke konkrétnímu popisu implementace importu objednávek z tohoto portálu. K portálu je nutné se nejdříve přihlásit, přihlášení nám zajistí vzdálená metoda `doLoginEnc()`. Návratovou hodnotou této metody je tzv. `SessionString` s platností 3 hodiny. Ten slouží při používání dalších metod jako identifikace přihlášeného uživatele. Po úspěšném přihlášení je potřeba stáhnout seznam událostí uskutečněných v našem portálu od minulého importu objednávek a to pomocí metody `doGetSiteJournalDeals()`. Tato metoda vrací seznam událostí jako je prodej zboží, vyplnění `Formuláře voleb přepravy` a zaplacení zboží. Jsou to však pouze metadata a informace o konkrétních prodaných produktech a zákaznících je nutné stáhnout zvlášť.

Začněme stažením informací o prodaných předmětech. K tomu slouží metoda `doGetItemsInfo()`, která vrací název prodaného zboží, cenu, počet kusů a další volitelné informace. Takto stažené produkty by se měly ideálně namapovat na evidované zboží v naší databázi. To se však nemusí stát například v tom případě, že prodejce předmět na portál vystaví ručně. V tomto případě se tedy prodávané zboží uloží do naší databázi.

Dále provedeme stažení dat zákazníků metodou `doGetPostBuyData()`. V naší webové aplikaci budou staženými daty buď aktualizována data již existujících zákazníků, nebo budou vytvořené zákazníci noví.

Zákazník může při prodeji libovolně vyplnit tzv. `Formulář voleb přepravy`, který informuje prodejce o zvoleném způsobu dopravy a platby zakoupeného zboží. Data vyplněných formulářů stáhneme z webové služby metodou `doGetPostBuyFormsDataForSellers()` a pomocí nich vyplníme potřebné údaje v námi vytvořených objednávkách.

Po stažení všech těchto údajů je uživateli zobrazena souhrnná informace o realizovaných prodejích a je vyzván k potvrzení uložení stažených dat. Potvrzením dojde k transformaci všech přijatých dat na formát potřebný pro uložení v databázi aplikace a data jsou v ní uchována.

5.6.4.2 Export zboží na Aukro.cz

Základem exportu zboží na portál Aukro.cz je metoda doNewAuctionExt() poskytovaná webovým API portálu. Pro vystavení nové nabídky je zapotřebí se opět na portál přihlásit, přihlášení
probíhá metodou `doLoginEnc()` a její návratovou hodnotou je již dříve uvedený `SessionString`. Tím je zapotřebí se identifikovat při volání dalších metod.


Správnost vystavení nabídky můžeme ověřit voláním metody `doVerifyItem()`, již předáme identifikátor námi zvolený při vystavování nabídky.

### 5.6.4.3 Import bankovních plateb z FIO banky

Import plateb z internetového bankovnictví FIO banky probíhá prostřednictvím bankou poskytovaného aplikačního rozhraní. Toto rozhraní je založeno na architektuře REST [41] postavené nad protokolem HTTP. Jedná se o jednoduché rozhraní, které dovoluje zadávat systému internetového bankovnictví příkazy, přičemž systém odpovídá formou textu s předem zvolenou strukturou. V našem případě byl zvolen formát XML, dostupné jsou však i další formáty jako CSV, JSON, GPC, atd. Celá komunikace mezi webovou aplikací a serverem banky je zabezpečena pomocí protokolu SSL s minimálně 128 bitovým šifrováním.

![Sekvenční diagram importu plateb z FIO banky](image)

**Obrázek 5.21: Sekvenční diagram importu plateb z FIO banky**

Přístup k datům z internetového bankovnictví prostřednictvím API musí být nejprve povolen majitelem účtu nebo osobou s patřičným oprávněním. Tato osoba musí ve svém internetovém
bankovnictví vygenerovat 64 znakový token, kterým se při následné komunikaci se serverem webové aplikace identifikuje. Tato činnost musí být provedena manuálně. V dalším textu předpokládejme, že tento token již byl vygenerován.

Aplikační rozhraní banky poskytuje systém s takzvanou zarážkou. Představme si, že platby evidované bankou jsou uloženy v jednom dlouhém seznamu. Vyžádáním seznamu plateb prostřednictvím aplikačního rozhraní dojde k nastavení zarážky na poslední v seznamu plateb, které byly aplikací vypsány. Každým dalším stažením plateb poté dochází ke stažením pouze takových, které jsou umístěny pod zarážkou. To jest takových, které od posledního stažení v systému přibyly.

Zarážku je však možné v případě potřeby posunout na libovolnou jinou transakci na základě znalosti jejího identifikátoru.

Nyní však přistupme k samotnému importu plateb. Prvním krokem je nastavení zarážky na poslední námi evidovanou platbu. To provádíme z toho důvodu, že pokud by býval při posledním importu nastal nějaký problém, zarážka by mohla být nastavena na jinou platbu než je námí poslední evidovaná. Po nastavení zarážky následuje stažení seznamu posledních plateb, a pokud proběhne úspěšně, obdržený XML dokument je napravován a informace o platbách jsou uloženy do databáze webové aplikace. Na obrázku 5.21 vidíme proces importu bankovních plateb včetně přesné podoby zasílaných příkazů.

5.6.5 Zálohování a obnova

Webová aplikace přináší uživateli možnost zálohovat si svá data, tedy přesněji veškeré svoje evidované objednávky, zboží, nákupy atd. Zálohu si uživatel může stáhnout ve formě souboru přímo z prostředí webové aplikace. Tento soubor může později sloužit pro návrat ke starším verzím dat prodeje, může být použit pro přenos dat různými uživatelskými účty v aplikaci, ale především bude sloužit k migraci mezi desktopovou a webovou verzí aplikace.

Jak jsme se dozvěděli výše, při registraci se pro každého uživatele vytváří zvláštní databáze, v níž jsou uložena výlučně jeho data. Schéma této databáze je shodné se schématem desktopové verze této aplikace. Prodejce tak má možnost pomocí zálohy přenést data z lokální desktopové verze aplikace do webového rozhraní nebo opačně. Toto přináší zejména pro podniky klíčovou vlastnost, kdy prodejce má možnost mít svá data zcela pod svoji kontrolou, pokud si to bude přát.

Kapitola 6

Propagace produktu

V této kapitole si shrneme možné možnosti propagace navrhaného webové aplikace tak, aby se snáze dostala ke svým zákazníkům – maloobchodním prodejcům. Dále si představíme metody, které byly k propagaci produktu zvoleny a provedeme zhodnocení jejich účinnosti. Popišeme si rovněž detaily nasazení aplikace v komerčním prostředí, což bylo také jedním z cílů této diplomové práce.

6.1 Možností propagace

V následující podkapitole si blíže představíme některé možnosti, které můžeme využít k propagaci výsledného produktu. Omezíme pouze na metody propagace produktu cestou internetu. Propagace produktu tradičními metodami, jako je například inzertní kampaň v médiích, bývá pro jednotlivce vyvíjející internetovou aplikaci zpravidla finančně nerealizovatelná.

6.1.1 Webová prezentace

Zcela nezbytnou položkou při propagaci navrhané webové aplikace je její internetová prezentace. Na webové prezentaci musejí potenciální zákazníci být informováni především o účelu poskytovaného produktu a zpravidla je také vstupní branou do samotné aplikace. Prezentace by měla být laděna v podobném grafickém duchu jako finální produkt, takové řešení zajistí dojem větší komplexnosti a lepšího uživatelského zážitku.

Kromě klasické textové specifikace funkcí by hlavní stránka webové prezentace měla obsahovat interaktivní schéma fungování aplikace, které by mělo na první pohled poskytnout povědomí o její samotné funkci. Důležitou součástí webové prezentace jsou také materiály, které případného zájemce blíže seznámí s funkcemi aplikace a poskytnou informace, zda produkt splňuje očekávané požadavky. Moderním přístupem v poskytování těchto informací jsou online videa. Divákové poskytují mnohem snazší cestu k požadovaným informacím než klasické textové popisy a zjednodušování je, možná naneštěstí, obecným trendem dnešní doby.
6.1.2 Google AdWords


V českých poměrech jsou to vyhledávače Seznam.cz [38] a Google [39], kteří společně zastávají celou dominantní podíl na trhu. V případě tohoto projektu by pak placená reklama měla být přednostně zobrazována ve výsledcích vyhledávání klíčových slovních spojení, jako je správa objednávek, tisk poštovních poukázek, apod.

6.1.3 Bannerová inzerce


Umístění banneru na takové weby je pak otázkou individuální dohody s provozovatelem a je obecně těžké o takové formě spolupráce předem odvodit její obecný průběh.

6.1.4 Sociální sítě

Fenoménem posledních deseti let je prezentace jednotlivců, firem i zájemcův skupin na sociálních sítích. Pro aplikace, jako je ta vytvořená v rámci diplomové práce, je hlavním důvodem prezentace v této oblasti především možnost udržování trvalého styku se zákazníky. Primárně tedy neslouží k přímému získávání nových uživatelů, ale spíše k vytváření komunity okolo projektu, informování současných zákazníků o novinkách v aplikaci a udržování kontaktu v podobě diskuze.

Opět je však potřeba zvažit potřebu takové prezentace vzhledem k zaměření konkrétní aplikace. Prezentace na sociálních sítích může být vhodná pro projekty určené k masovému použití, což navrhovaná aplikace není. Správa profilu na sociální síti rovněž vyžaduje poměrně velké úsilí a zpravidla každodenní monitorování reakcí uživatelů a odpovídání na jejich příspěvky.

Z pohledu provozovatele aplikace je hlavní slabinou celého systému prezentace na sociálních sítích především nemožnost moderovat a regulovat uživatelské příspěvky. Toto může vyústit v nezcela příznivou situaci pro provozovatele, kdy negativní obsah několika málo příspěvků může vážným způsobem poškodit reputaci celého projektu.
6.2 Zvolený způsob propagace

Původním záměrem zamýšleným v době návrhu webové aplikace byla realizace propagace výsledného produktu prostřednictvím webové prezentace a její zviditelnění v podobě reklamní kampaně ve vyhledávači Google. Neplánované prodloužení doby práce na samotné webové aplikaci však nedovolilo tento časově poměrně náročný plán realizovat. Výsledkem byla volba způsobu propagace, který by byl realizovatelný ještě v průběhu tvorby samotného produktu.


Obrázek 6.1: Anketní otázka na webové stránce


Pojdeme se tedy zaměřit na krátké zhodnocení nasbíraných dat. Z grafu je patrné, že majoritní skupina respondentů má zájem na tvorbě webové verze aplikace, konkrétně 65,3 % dotázaných. Druhá největší skupina čítající 24,5 % dotázaných odpověděla neutrálně a posledních 10,2 % respondentů vyjádřilo nesouhlas nad tvorbou online verze.
Respondentům odpovídajícím na anketní otázku kladně byl po zaškrtnutí volby nabídnut vstupní box pro zadání emailové adresy, na kterou mohou dostávat informace o novinkách ohledně přípravy webové verze. Z celkového počtu 32 kladně hlasujících vyplnilo svou emailovou adresu 26 dotázaných. Všem z nich byl automaticky odeslán email jako poděkování za poskytnutý hlas a následně v průběhu dvou měsíců dvě emailové zprávy informující o průběhu implementace a předpokládaném datu dokončení. Celkem 3 uživatelé následně kladli dotaz v odpovíde na zaslaný email, čímž dále potvrdili svůj zájem o online verzi.

### 6.3 Reálné nasazení

Jedním z bodů zadání bylo otestovat webovou aplikaci v reálném nasazení. Toto nasazení aplikace proběhlo v internetovém obchodě Vortex Computers, zabývajícím se prodejem výpočetní techniky a jiné elektroniky [36] a dále ve společnosti OnStyle s.r.o., zaměřující se na výrobu a prodej rytých šperků [37]. V obou těchto případech byla aplikace používána paralelně s již existující instalací desktopové aplikace, přičemž pracovala nad stejnými daty. Cílem nasazení bylo především získání zpětné vazby z reálného použití.
V obou případech proběhlo nasazení bez obtíží a bylo získáno velké množství podnětů pro aktuální i budoucí vývoj celé webové aplikace. Výhodou byla možnost použití stávajícího ověřeného desktopového systému, pokud by z jakéhokoli důvodu webová verze nefungovala.

Níže naleznete stručný výpis některých podnětů pro další vývoj webové aplikace. Přijímány byly podněty ze všech možných oblastí a pro prezentaci vybíráme ty zajímavější:

- Přizpůsobit zobrazení detailů položek agend pro monitory s velmi vysokým rozlišením. V současné době dochází na extrémně velkých displejích k nepříjemnému roztažení ovládacích prvků přes celou obrazovku.
- Implementovat agendu pro evidenci vypůjčeného zboží.
- Na hlavní stránce zobrazit komponentu pro přímé spojení s technickou podporou.
- Implementovat napojení na další prodejní portály a bankovní systémy.
- Tisk přehledu zboží, které je aktuálně skladem.
Kapitola 7

Závěr

Cílem této práce bylo navrhnut informační systém pro podporu maloobchodního prodeje. V úvodní části byl problém analyzován, dále byly specifikovány požadavky na navrhovaný informační systém, požadavky byly zkonzentrovány s jinými existujícími řešeními a dále bylo přistoupeno k návrhu. V návrhu byly podrobně popsány jednotlivé funkce programu, doplněny grafickými návrhy obrazovek a diagramy vytvořenými v konvenci jazyka UML. V poslední části návrhu se zaměřujeme na návrh databáze a tříd v aplikaci. Následuje obsáhlá kapitola popisující implementaci webové aplikace a dále kapitola o možnostech propagace produktu.

V tomto diplomovém projektu se podařilo vytvořit webovou aplikaci implementující až na výjimky všechnu požadovanou funkčnost. Těmito výjimkami, které se do doby odevzdání projektu nepodařilo zdárně nasadit je evidence sériových čísel a evidence parametrů u zboží.

V průběhu testování při reálném nasazení v obou společnostech byly zaznamenávány četné podněty pro rozšíření funkčnosti aplikace nad rámec této diplomové práce. Z důvodu pozitivních ohlasů na tento projekt by bylo velice vhodné tuto práci dále rozvíjet a přivézt k nasazení u většího počtu klientů.
Literatura


## Seznam použitých zkratek a symbolů

<table>
<thead>
<tr>
<th>Zkratka</th>
<th>Celý název</th>
<th>Vysvětlení</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
<td>Rozhraní pro programování aplikací</td>
</tr>
<tr>
<td>AJAX</td>
<td>Asynchronous JavaScript and XML</td>
<td>Technologie pro interaktivní webové stránky</td>
</tr>
<tr>
<td>ASP</td>
<td>Active Server Pages</td>
<td>Technologie pro tvorbu webových aplikací</td>
</tr>
<tr>
<td>CRUD</td>
<td>Create-Read-Update-Delete</td>
<td>Zkratka pro 4 základní operace nad daty</td>
</tr>
<tr>
<td>CSV</td>
<td>Comma-separated values</td>
<td>Hodnoty oddělené čárkami, formát souboru</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
<td>Značkovací jazyk pro webové stránky</td>
</tr>
<tr>
<td>MSSQL</td>
<td>Microsoft SQL</td>
<td>Databázový server</td>
</tr>
<tr>
<td>MVC</td>
<td>Model-View-Controller</td>
<td>Vzor pro vytváření webových aplikací</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
<td>Protokol pro výměnu strukturovaných dat</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
<td>Dotazovací jazyk pro práci s databází</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
<td>Grafický jazyk pro vizualizaci</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
<td>Rozšířitelný značkovací jazyk</td>
</tr>
</tbody>
</table>
Seznam příloh

Příloha 1  ER diagram databáze uživatele
Příloha 2  Konceptuální diagramy tříd
   Příloha 2a  Konceptuální diagram tříd pro objednávky
   Příloha 2b  Konceptuální diagram tříd pro zboží
   Příloha 2c  Konceptuální diagram tříd pro reklamace
   Příloha 2d  Konceptuální diagram tříd pro bankovní platby
   Příloha 2e  Konceptuální diagram tříd pro portál Aukro.cz
   Příloha 2f  Konceptuální diagram tříd pro nastavení a pohledy
   Příloha 2g  Konceptuální diagram tříd pro tisk dokumentů
Příloha 3  Seznam XML elementů pro vytváření tiskových šablon
Příloha 4  Příklad emailové šablony
Příloha 1 – ER diagram databáze uživatele
Příloha 2a – Konceptuální diagram tříd pro objednávky
Příloha 2b – Konceptuální diagram tříd pro zboží
Příloha 2c – Konceptuální diagram tříd pro reklamace
Příloha 2d – Konceptuální diagram tříd pro bankovní platby
Příloha 2e – Konceptuální diagram tříd pro portál Aukro.cz
Příloha 2f – Konceptuální diagram tříd pro nastavení a pohledy
Příloha 2g – Konceptuální diagram tříd tisk dokumentů
## Příloha 3 – Seznam XML elementů pro vytváření tiskových šablon

<table>
<thead>
<tr>
<th>Název elementu</th>
<th>Povolené atributy</th>
<th>Povolené vnitřní elementy</th>
</tr>
</thead>
</table>
| **Page**  
kořenový element dokumentu | Title Název dokumentu  
Action Akce prováděné před tiskem (např. vygenerování číslo faktury) | Header, Footer, Expander |
| **Header** | | |
| **Footer** | | Stejné s elementem Box |
| **Expander** | | |
| **Box**  
oblast pro vkládání dalších elementů, je základem hierarchie | BorderWidth Šířka elementu  
FontSize Velikost písma  
Font Typ písma  
FontWeight Styl písma: {bold | italic | normal}  
Align Zarovnání textu  
Positon Pozice: souřadnice X;Y  
Size Velikost: šířka;výška  
Height Šířka  
Width Výška | Text, Line, Box, Table, Image, Textarea |
| **Line**  
čárový oddělovač | Start počáteční bod  
End koncový bod  
Width šířka čáry | - |
| **Text**  
textová hodnota | Value zobrazovaný text  
Suffix textové zakončení (např. „Kč“)  
FontSize Velikost písma  
Font Typ písma  
FontWeight Styl písma: {bold | italic | normal}  
Align Zarovnání textu  
Positon Pozice: souřadnice X;Y  
Size Velikost: šířka;výška  
Height Šířka  
Width Výška | - |
| **Image**  
vložený obrázek | Source Zdroj obrázku:  
{"[owner_logo]"|  
"[owner_signature]"}  
Positon Pozice: souřadnice X;Y  
Size Velikost: šířka;výška  
Height Šířka  
Width Výška | - |
| **Textarea**  
textová oblast, dovoluje vložit formátovaný text | FontSize Velikost písma  
Font Typ písma  
FontWeight Styl písma: {bold | italic | normal}  
Align Zarovnání textu  
Positon Pozice: souřadnice X;Y  
Size Velikost: šířka;výška  
Height Šířka  
Width Výška | - |
### Table

**Table**

<table>
<thead>
<tr>
<th>Tabulkové zobrazení položek</th>
<th>Source</th>
<th>Zdroj dat</th>
</tr>
</thead>
</table>
| **TableHeader**
  definice stylu hlavičky tabulky | Background | Barva pozadí |
|                         | Padding | Odsazení od kraje |
|                         | FontSize | Velikost písma |
|                         | Font | Typ písma |
|                         | FontWeight | Styl písma: {bold | italic | normal} |
|                         | Align | Zarovnání textu |
|                         | Positon | Pozice: souřadnice X;Y v bodech |
|                         | Size | Velikost: šířka;výška v bodech |
|                         | Height | Šířka |
|                         | Width | Výška |
| **TableRows**
  definice stylu řádků tabulky | FontSize | Velikost písma |
|                         | Font | Typ písma |
|                         | FontWeight | Styl písma: {bold | italic | normal} |
|                         | Align | Zarovnání textu |
|                         | Positon | Pozice: souřadnice X;Y v bodech |
|                         | Size | Velikost: šířka;výška v bodech |
|                         | Height | Šířka |
|                         | Width | Výška |
| **TableColumns** | - | - |

### TableLine

<table>
<thead>
<tr>
<th>Ohraničení buněk tabulky</th>
<th>Type</th>
<th>Typ čáry: {border, separator}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Orientation</td>
<td>Orientace čáry: {top, left, right, bottom, horizontal, vertical}</td>
</tr>
</tbody>
</table>

### TableColumn

<table>
<thead>
<tr>
<th>definice obsahu sloupce tabulky</th>
<th>HeaderText</th>
<th>Text v hlavičce sloupcu</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Width</td>
<td>Šířka sloupce</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Typ hodnoty ve sloupci</td>
</tr>
<tr>
<td></td>
<td>Suffix</td>
<td>Textový dodatek k hodnotám</td>
</tr>
<tr>
<td></td>
<td>Align</td>
<td>Zarovnání ve sloupci</td>
</tr>
</tbody>
</table>
Příloha 4 – Příklad emailové šablony

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/1999/REC-html401-19991224/strict.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=cp1250">
  <title>Vortex computers</title>
</head>
<body align="center" style="font-family: Tahoma, Arial; font-size: 10px; color: #FFF;">
<table style="background-color: #036;">
  <tr>
    <td colspan="2" style="padding: 5px;">
    </td>
  </tr>
  <tr>
    <td align="left" style="font-weight: bold; padding: 10px;" valign="top">Vaše objednávky odesláno.</td>
  </tr>
  <tr>
    <td align="left" style="border-radius: 5px; background-color: #036; color:#FFF; font-size: 12px; font-weight: bold; padding: 8px 10px;" valign="top">Detaily objednávky</td>
  </tr>
  <tr>
    <td align="left" style="padding: 10px;" id="items">
      ID objednávky: <span style="color: #036; font-weight: bold;" id="ID"></span><br />
      Datum objednání: [CREATE_DATE]<br />
      Způsob platby: <strong>[PAYMENT]</strong><br />
      Způsob dopravy: <strong>[DELIVERY]</strong><br />
      Číslo zásilky: <strong>[BARCODE]</strong><br /> 
      E-mail: <strong>[EMAIL]</strong><br />
      Telefon: <strong>[PHONE]</strong><br />
    </td>
  </tr>
  <tr>
    <td style="width: 100%; font-family: Verdana, sans-serif; font-size: 11px;">
      <table style="text-transform: uppercase; color: #FFFFFF;">
        <tr style="background-color: #036; border-top-left-radius: 5px; padding: 8px;">
          <td style="background-color: #036; border-top-right-radius: 5px; padding: 8px;">
            Fakturační adresa</td>
        </tr>
        <tr style="vertical-align: top;">
          <td style="padding: 8px; background-color: #EEEEEE;" [INV_COMPANY] [INV_NAME] [INV_SURNAME] [INV_STREET] [INV_COUNTRY] [POST_COMPANY] [POST_NAME] [POST_SURNAME] [POST_STREET] [POST_COUNTRY] </td>
        </tr>
      </table>
    </td>
  </tr>
  <tr>
    <td style="padding: 15px; padding-bottom: 30px;">
      <table style="width: 100%; font-family: Verdana, sans-serif; font-size: 11px;">
        <th align="left" style="padding: 5px 8px;">Číslo</th>
        <th align="right" style="width: 15%; padding: 5px 8px;">Sob platby</th>
        <th align="right" style="width: 15%; padding: 5px 8px;">Sob dopravy</th>
        <th align="right" style="width: 20%; padding: 5px 8px;">Celkem s DPH</th>
        <th align="right" style="width: 15%; padding: 5px 8px;">Množství</th>
      </table>
    </td>
  </tr>
</table>
</body>
</html>
<table>
<thead>
<tr>
<th>TITLE</th>
<th>PRICE</th>
<th>QUANTITY</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poštovné:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Celkem:</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tento email byl automaticky odeslán programem Casid - správa objednávek.

© 2015 Vortex computers