FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Analytický nástroj pro párové obchodování
Student:	Bc. Kamil Maleček
Vedoucí:	Ing. Ondřej Guth, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Prostudujte strategii párového obchodování s akciemi. Prozkoumejte existující nástroje, které pomáhají s touto strategií.
Navrhněte analytický nástroj, který bude umět na základě informací o akciích a statistických ukazatelích vypočítat scoring, tedy ohodnotit a doporučit nejvhodnější kandidáty pro obchodování. Použijte i některý z pokročilejších ukazatelů, například kointegraci.
Implementujte tento nástroj pomocí standardní technologie Java EE 8 a vytvořte aplikaci s přehledným uživatelským grafickým rozhraním, umožňujícím konfiguraci parametrů pro hledání kandidátních párů a zobrazení detailu jednotlivých výsledků. Aplikaci navrhněte a realizujte tak, aby byla snadno rozšiřitelná pro další ukazatele i parametry pro hledání.
Proveďte backtesting s využitím historických dat například ze serveru finance.google.com.
Zhodnoťte výstupy.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 14. prosince 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Analytický nástroj pro párové obchodování

Bc. Kamil Maleček

Katedra softwarového inženýrství
Vedoucí práce: Ing. Ondřej Guth, Ph.D.

6. května 2018

Poděkování

Děkuji vedoucímu diplomové práce Ing. Ondřeji Guthovi, Ph.D. za ochotu a cenné rady. Dále děkuji Ing. Pavlovi Müllerovi za nápad na zajímavé téma.

Prohlášení

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

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

V Praze dne 6. května 2018 .....

Odkaz na tuto práci

Maleček, Kamil. *Analytický nástroj pro párové obchodování*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Diplomová práce se zabývá návrhem a implementací analytického nástroje pro strategii párového obchodování s akciemi. Pro implementaci jsou využity standardní nástroje Java Enterprise Edition, JavaServer Faces a aplikační server GlassFish. Na základě výstupů z testování je v sekci Závěr zhodnocena celá aplikace a jsou poskytnuty návrhy na její vylepšení do budoucna.

Klíčová slova akcie, párové obchodování, analytický nástroj, kointegrace, GlassFish 5, Java EE 8, JSF 2.3, PrimeFaces, OmniFaces, SuanShu

Abstract

This diploma thesis deals with the design and implementation of an analytic tool for stocks trading, namely for the pair trading strategy. The implementation is done by standard technologies available in Java Enterprise Edition, JavaServer Faces and application server GlassFish. In the last chapter, there is a description of pros and cons of the application and some ideas for future improvements are discussed.

Keywords stocks, pair trading, analytic tool, cointegration, GlassFish 5, Java EE 8, JSF 2.3, PrimeFaces, OmniFaces, SuanShu

Obsah

Seznam obrázků

Úvod

Automatizace činností je dnes již běžnou součástí života nás všech, nicméně ještě před nedávnou dobou bylo nepředstavitelné, jak moc lze věci automatizovat a usnadnit tak naše bytí. Pro člověka jde o velice užitečnou věc, která už dávno není jen o zefektivnění procesů a úkolů v práci, či při studiu, ale čím dál více proniká do dalších oblastí našeho života.

Člověk se díky automatizaci některých činností může věnovat i věcem, na které by jinak neměl dostatek času ani energie. Příkladem za všechny je obchodování na burze. Tato činnost byla ještě donedávna přístupná jen pro několik málo vyvolených, zatímco pro ostatní to byl neznámý svět, do kterého nemohli z důvodu chybějícího času, znalostí nebo financí proniknout.

S příchodem automatizačního software logicky přišly i analytické produkty pro různé druhy obchodování na burzách, které slibují snadný vstup do dříve zapovězeného světa pomocí rychlé a automatické aplikace statistických ukazatelů a interpretace výstupů, čímž více, či méně pomáhají uživatelům k touženému zisku. Práce se zabývá analýzou, návrhem a implementací právě takového analytického nástroje.

Cíle práce

Cílem diplomové práce je prostudování strategie párového obchodování s akciemi a dostupného pomocného software pro tuto strategii. Na základě získaných znalostí je úkolem návrh analytického nástroje pro strategii párového obchodování s akciemi, který bude umět vypočítat scoring nalezených akciových párů, tedy jejich ohodnocení a doporučení těch nejvhodnějších pro reálné obchodování. Pro implementaci budou použity standardní nástroje Java Enterprise Edition verze 8 (Java EE 8). Cílem je jednoduché a přehledné uživatelské rozhraní aplikace, které musí umožňovat vyhledávání kandidátních párů na základě parametrů a zobrazování detailu aplikací doporučených akciových párů. Výstupem bude snadno rozšiřitelná aplikace.

Struktura práce

První kapitola diplomové práce obsahuje popis nejenom burzovního obchodování, ale věnuje se druhům investování obecně, jeho principům a možnostem. Práce se týká strategie párového obchodování s akciemi, kterou první kapitola rozebírá do detailu, uvádí její vlastnosti, výhody a nevýhody. Kapitola se věnuje také aktuální nabídce dostupného software, jehož účelem je analýza trhů, a který by mohl být vhodný pro zvolenou strategii párového obchodování s akciemi.

Na základě poznatků, získaných v první části diplomové práce, jsou identifikovány požadavky na novou aplikaci a případy užití. Následuje doménový model. Dle analýzy je poté navržena aplikace nová, speciálně na míru strategie párového obchodování s akciemi. Návrh je proveden nejen z pohledu architektury aplikace a návrhu tříd, ale také z pohledu uživatelského rozhraní, jemuž práce obecně přikládá velkou pozornost. Kapitolu zakončuje model nasazení aplikace.

Následuje kapitola zabývající se realizací. V úvodu jsou popsány použité technologie a problematika analýzy dat o akciích. Zbytek kapitoly se zabývá implementací funkčního prototypu analytického nástroje pro strategii párového obchodování s akciemi. Zmíněny jsou také zajímavé implementační problémy, které bylo potřeba v průběhu vývoje řešit. Nechybí Nielsenova heuristická analýza a ukázka aplikace.

Přímo navazuje kapitola o testování. To bylo provedeno nejenom pomocí jednotkových testů, ale také prostřednictvím reálných uživatelů provádějících konkrétní testovací scénáře. Postřehy z testování byly zapracovány.

V závěru přichází na řadu zhodnocení realizovaného prototypu, porovnání jeho vlastností s již existujícím software a z toho plynoucí návrhy na vylepšení, či rozšíření nového analytického nástroje.

Kapitola 1

Investování obecně

Diplomová práce se zabývá problematikou akciového obchodování na burze, nicméně na začátku je vhodné uvést kontext, vysvětlit si některé základní pojmy a popsat druhy investic. Výčet všech možností investování by stačil na samostatnou práci, a proto práce stručně popisuje pouze některé vybrané způsoby. U investic jsou důležité následující tři parametry:

- Výnos – výše zisku po skončení investice.
- Riziko – vyšší rizikovost přináší větší výnosy, ale také větší ztráty.
- Likvidita – likvidní majetek je snadno a rychle obchodovatelný.

1.1 Mimoburzovní zhodnocování

Jako první kategorii lze vymezit činnosti, které jsou alternativou ke klasickému obchodování na burze.

1.1.1 Sběratelství

Z pohledu historie se jedná bezesporu o jednu z prvních možností zhodnocování financí. Do této kategorie lze zařadit nakupování obrazů a dalšího umění, mincí, výjimečných a vzácných historických předmětů starých, či nových. Lidé, kteří se sběratelství věnují, by se dali povětšinou označit spíše za nadšence s hlubokou znalostí předmětů zájmu. Jakkoliv je tato kategorie zajímavá, je velice vzdálená tématu, a proto se jí práce dále nevěnuje.

1.1.2 Nemovitosti

Nákup bytů a pozemků je poměrně rozšířeným druhem investování, jelikož může přinášet určitý pasivní příjem v podobě nájmu. Tento cíl má však několik podmínek, nemovitost je potřeba pronajímat nájemci s dobrou platební

morálkou. Neplatič může přinést mnoho problémů, které je nutné řešit a majitel zatím přichází o očekávaný příjem z investice. Je potřeba počítat také s opotřebením bytu a dalšími souvisejícími výdaji. Pronajímání nemovitostí je také závislé na lokalitě nemovitosti, proto nákup bytu v centru Prahy lze považovat za lepší investici než například nákup domu v malé vesnici na Vysočině, či v Ústeckém kraji. Závěrem lze říci, že investování do nemovitostí za účelem pasivního příjmu formou nájemného může přinést profit, ale i množství problémů.

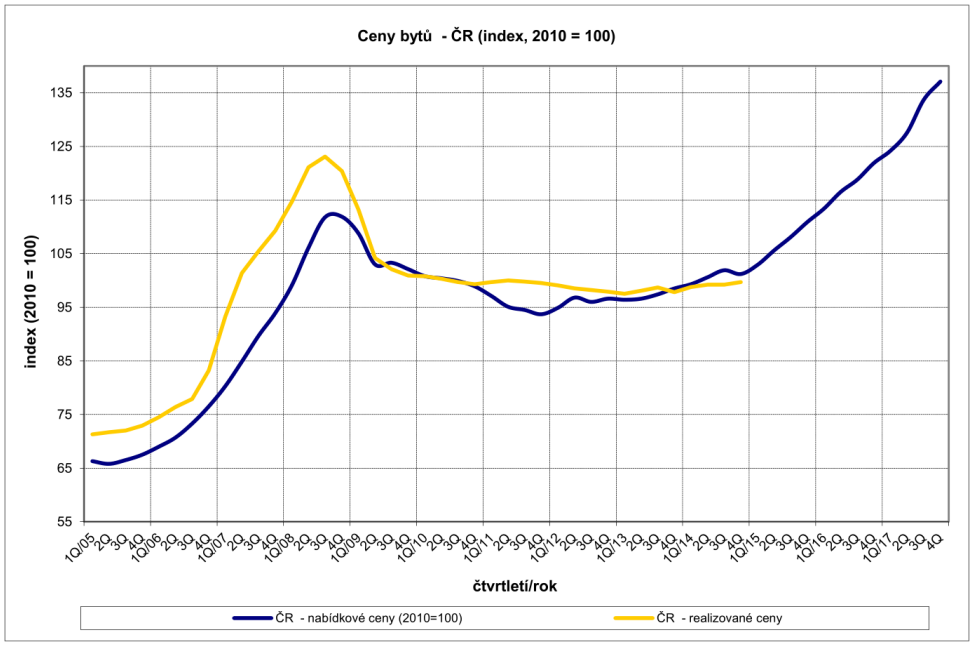
Nakupovat nemovitosti však lze i za jiným účelem, a sice pro očekávaný růst hodnoty nemovitosti [1]. Z grafu 1.1 (data pochází z Českého statistického úřadu [2]) však vyplývá, že výhodnost investice je silně závislá na aktuálním prostředí. Před krizí v roce 2008 poptávka vysoce převyšovala nabídku a ceny stoupaly. Nákup bytu ve třetím kvartálu roku 2008, kdy byla hodnota indexu 123, nebyla úplně výhodná investice, protože ještě v koncem roku 2014 byla hodnota indexu cca 100, tedy propad o 23 bodů hodnoty indexu, respektive o 23 % průměrné ceny bytů v ČR. S výhledem do dnešních dní (začátek 2018) lze předpokládat, že se reálná cena bytů vyšplhá zpátky takovým způsobem, že i byt koupený na konci roku 2008 bude mít vyšší hodnotu. V budoucnu se dá očekávat splasknutí bubliny a pád cen, stejně jako po roce 2008. Vzhledem ke snížené poptávce se však ČNB (Česká národní banka) rozhodla nastartovat ekonomiku a přimět lidi k investování snížením úrokových sazeb až na pouhých 0,05 %, jak ukazuje graf repo sazby 1.2 (data pochází z ČNB [3]). Repo sazba v důsledku ovlivňuje cenu půjček [4]. Začalo být výhodnější investovat za cenu levné půjčky, než nechat peníze na spořícím účtu s v podstatě nulovým úrokem. Vliv na cenu má také nižší poptávka na venkově nebo naopak vysoká poptávka v Praze, způsobená mimo jiné často problematickou výstavbou nových bytů.

1.1.3 Fyzické komodity

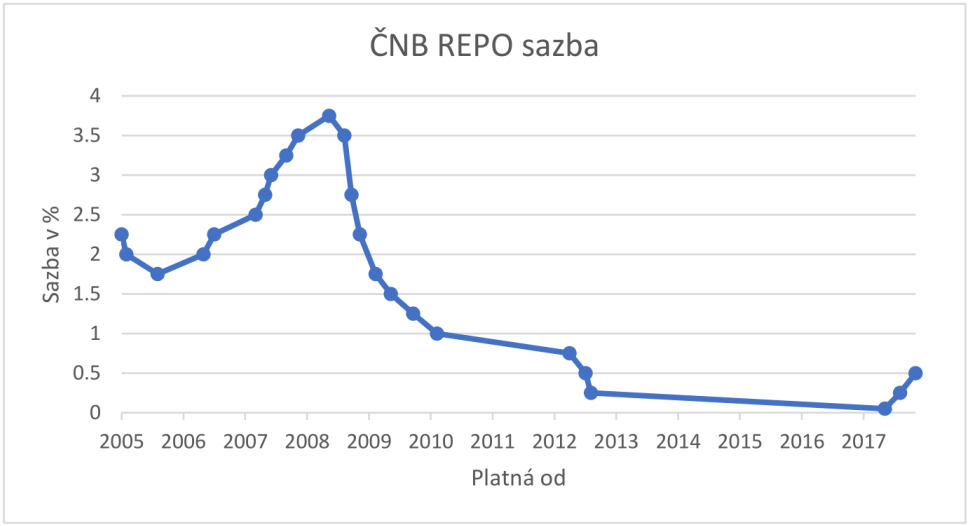
Komodity lze obchodovat více způsoby:

- Nákup samotné fyzické komodity.
- Ve formě takzvaných futures kontraktů, kterým se věnuje kapitola 1.2.2.1.
- Nepřímo nákupem akcií těžařských společností, více viz kapitola 1.2.1.1.
- Nákupem na burze obchodovaných fondů (anglicky Exchange Trated Funds, zkráceně ETF, více v kapitole 1.2.1.3), které nakupují fyzickou komoditu. Například fond SPDR Gold Shares GLD, který aktuálně vlastní přibližně 1300 tun zlata v hodnotě přes 71 miliard dolarů [5].

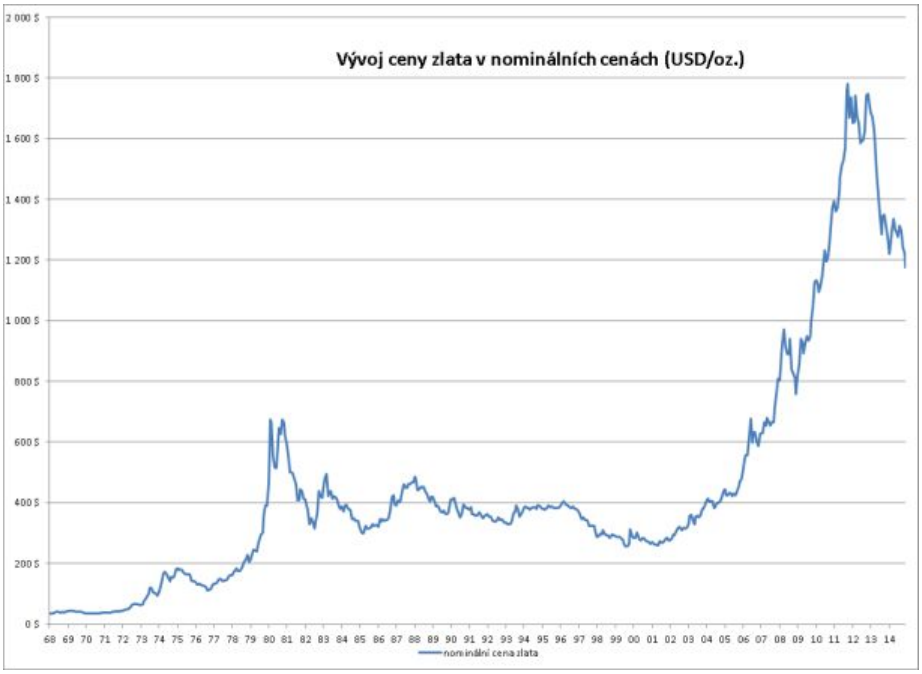
První možnost se týká především drahých kovů, typicky zlata. Argumentem investorů jdoucích touto cestou je často ochrana proti hyperinflaci, tedy stavu, kdy inflace přesáhne 100 % během určitého období, a s tím souvisejícím děním na trhu, kdy měna rychle ztrácí hodnotu a národní banky nestíhají tisknout



Obrázek 1.1: Ceny bytů ČR [2]



Obrázek 1.2: ČNB REPO sazba [3]



Obrázek 1.3: Nominální cena zlata [9]

nové bankovky s mnohem vyšší nominální hodnotou než je běžné [6]. Jako příklad lze uvézt německou ekonomiku kolem roku 1923 viz [7] nebo situaci v Zimbabwe mezi roky 2007 až 2009 viz [8]. Tento argument podporuje graf 1.3, zobrazující reakci ceny zlata na ropnou krizi v roce 1979 a finanční krizi v roce 2007. Z obrázku 1.4, který zobrazuje reálné ceny (z roku 1968) zlata očištěné o index spotřebitelských cen USA, vyplývá, že nákup zlata v období krize 1980 až 1981 opravdu nebyl žádnou ochranou před inflací. Nakupovat zlato v době krize, kdy cena stoupá a je výše než dlouhodobý průměr, se nevyplatí [9].

1.1.4 Termínovaný vklad

Jedná se o bankovní produkt, který spočívá v uložení obnosu peněz na určitou, předem sjednanou dobu. S penězi nelze po dobu trvání vkladu nijak manipulovat, v případě předčasného výběru přicházejí sankce. Odložený peněžní obnos je po celou dobu úročen lépe, než v případě běžného účtu. Zřízení vkladu je typicky zdarma. Mezi nevýhody patří znehodnocování úroků vlivem inflace. Z úroků platíme daň. Vklad má vysokou likviditu, nízké riziko, ale také nízké výnosy.

Obrázek 1.4: Reálná cena zlata [9]

1.1.5 Podílové fondy

Investor za základě vkladu obdrží od správce fondu takzvané podílové listy, jejichž cena se v čase mění a pro vlastníka představuje zisk, nebo ztrátu. Existuje více druhů podílových fondů:

- Akciové – rizikovější, s vyšším výnosem. Spíše dlouhodobá investice.
- Dluhopisový – střední rizikovost i míra výnosu.
- Peněžní – malá míra rizika, ale také nízké výnosy. Krátkodobá investice.
- Smíšený – vlastnosti závisí na skladbě fondu.

Výhodou je, že o portfolio se starají společnosti, jež mají mnoho zkušených analytiků, kteří vytvářejí široká portfolia za účelem rozložení rizika. Další výhodou (oproti termínovaným vkladům) je možnost výběru peněz kdykoliv. Stinnou stránkou podílových fondů jsou poplatky. Lze se setkat se vstupním, výstupním a poplatkem za správu fondu [10]. Z dat [11] asociace SPIVA (S&P Indices Versus Active Funds), která porovnává výsledky akciových indexů oproti podílovým fondům, vyplývá, že v roce 2017 mělo 56,46 % všech podílových fondů v U.S. horší výsledky než index Standard and Poor's 1500 (S&P 1500), který zahrnuje hodnotu přibližně 90 % všech obchodovatelných akcií na NYSE (New York Stock Exchange). V roce 2011 to bylo dokonce 83,88 %.

1.1.6 FOREX

Mezinárodní mimoburzovní obchodní systém pro směnu měnových párů se nazývá FOREX (Foreign Exchange). Jeho střední kurzy se považují za oficiální

světové kurzy. Jedná se o devizový trh bez centrální regulace, který spojuje banky, pojišťovny, investiční fondy a brokerské společnosti. Trh se vyznačuje svou rychlostí, největší likviditou na světě a také tím, že se jedná o jediný trh otevřený 24 hodin denně, 5 dní v týdnu – obchodní den začíná v Sydney a přes Tokio, Londýn končí v New Yorku, kdy už v Sydney začíná nový den. Je tak možné kdykoliv reagovat na aktuální vývoj. Denně proběhnou transakce v hodnotě vyšší než 5,1 biliónů amerických dolarů. Odhaduje se, že okolo 5 % objemu obchodů na trhu provádí korporace a vlády, které nakupují a prodávají zboží nebo služby v zahraničí a zbylých 95 % obchodů tvoří investice s cílem dosažení zisku na pohybu měn, jejich vzestupu i poklesu [12]. Princip je jednoduchý – nakoupit měnu za nižší hodnotu a prodat ji za hodnotu vyšší. Pro FOREX je typické použití pákového efektu (anglicky leverage), kdy za nízkou marginální částku lze obchodovat s několikanásobně větším objemem a dosahovat vyšších zisků, ale i ztrát. Výši páky určuje broker, u FOREX trhu jde o poměr například 1:50, ale i hodnoty přesahující 1:100 [12] [13]. FOREX prostředí je obecně rizikovější, zvyšuje se s pákou. To přináší také vyšší ziskovost.

1.1.7 Shrnutí

Sběratelství, nákup nemovitostí a investice do fyzických komodit, či podílových fondů se dají označit za tzv. Buy And Hold strategii. Ta spočívá v nákupu a dlouhém držení nakoupeného prvku, z čehož vyplývá, že strategie nebere v potaz krátkodobé pohyby cen a další ukazatele. Z pohledu investování jde o strategii, která vyžaduje minimum času, ale zhodnocení se děje v dlouhém časovém horizontu [14]. Rozhodujícím prvkem je tedy načasování nákupu i prodeje vzhledem k aktuálnímu prostředí.

1.2 Burzovní trhy

Burza (anglicky exchange) je obchodní místo, trh, na němž se podle striktních pravidel soustřeďuje nabídka a poptávka. Ještě v nedávné době bylo obchodování založené na telefonování a zprostředkování obchodu člověkem, který byl fyzicky přítomný na burzovním parketu a vykonával příkazy obchodníků pro nákup a prodej. Pro zajímavost, burzovní parket na NYSE (New York Stock Exchange) má rozlohu 11 285 metrů čtverečních [15]. Dnes, v době internetu převažuje elektronické obchodování, kdy můžeme obchodovat odkudkoliv prostřednictvím brokera, což je prostředník (jednotlivec, či firma), vlastnící licenci, mezi kupujícím a prodávajícím, který provádí nákupy a prodeje zadané investorem. Broker poskytuje své služby za poplatky, nicméně spolu s příchodem elektronického obchodování vzniklo mnoho brokerů s nízkými poplatky, což umožnilo vstup na burzu téměř každému. Broker však může za další poplatky poskytovat i jiné služby, jako například odborné poradenství.

Na burze lze obchodovat s různými podkladovými aktivy a také jejich deriváty [16] [17].

1.2.1 Podkladová aktiva

Obchodování s podkladovými aktivy probíhá tak, že se obchod uzavře v určitém momentu za aktuální tržní cenu a v tom samém okamžiku dojde i k reálnému vypořádání obchodu. Existuje několik druhů podkladových aktiv, dále budou zmíněny ty nejběžnější.

1.2.1.1 Akcie

Akcie je cenný papír vydaný akciovou společností, představující podíl jejich majitele na kapitálu akciové společnosti, a tím i oprávnění společníka akciové společnosti podílet se na zisku společnosti formou dividend, právo podílet se na řízení společnosti hlasováním na valných hromadách nebo právo na úpis dalších akcií při zvýšení základního kapitálu. Akcionář neručí za závazky společnosti. Akcie některých společností se obchodují na burzách cenných papírů, kde se v rámci nabídky a poptávky určuje jejich tržní hodnota. Lze si například zakoupit akcie Apple nebo Facebook na burze New York Stock Exchange nebo akcie ČEZu na Burze cenných papírů Praha. Diplomová práce se zabývá právě obchodováním na akciových burzách [18] [19]. Charakteristickým znakem je vysoká rizikovost i výnosnost.

1.2.1.2 Akciové indexy

Burzovní index je ukazatelem vývoje akciového trhu jako celku. Reflektuje současný stav vývoje trhu, tak i dlouhodobý vývoj trhu a jeho tendence. Svůj vlastní index má každý burzovní či mimoburzovní trh. Hodnota se vypočítává různými metodami z hodnot akcií vybraných společností nebo akcií vybraných dle určitých kritérií. Mezi nejznámější patří Dow Jones Industrial Average (DJIA) či Standard and Poor's 500 (S&P 500) index, který je vypočítáván jako vážený průměr kurzů akcií 500 vybraných společností v USA. S&P 500 přinesl velké zhodnocení dlouhodobých investic v posledních letech (období do 27. května 2016). Pětiroční výnos byl 57,67 %, desetiroční 63,94 % a výnos za dvacetileté období byl 209,31 %. Jako evropského zástupce lze uvést DAX, který se počítá z 30 největších a nejlikvidnějších německých společností, jako například: Allianz, Siemens, Volkswagen apod., které se obchodují na Frankfurt Exchange [18] [20]. Do akciových indexů lze investovat více způsoby:

- Podílové listy fondu, který přesně kopíruje vývoj daného indexu 1.1.5.
- Nákup investičních certifikátů, navázaných na určitý index.
- Obchodovat je formou futures derivátů viz 1.2.2.1.

1.2.1.3 Exchange Trated Funds

Na burze obchodované fondy se liší od běžných fondů nabízených různými bankami 1.1.5. ETF totiž vydaly své vlastní akcie a obchoduje se s nimi stejně jako s cennými papíry jiných akciových společností. Nákupem ETF akcií investor získává kompletní portfoliu fondu [18]. Existuje více druhů:

- Indexové ETF, které jsou složeny z akcií a portfoliu odpovídá určitému akciovému indexu viz kapitola 1.2.1.2.
- Komoditní ETF byly již zmíněny v kapitole 1.1.3. Jedná se o ETF, které nejsou složeny z akcií, ale z futures (viz kapitola 1.2.2.1) či jiných komoditami zajištěných aktiv.
- Měnové ETF kopírují vybrané měny, nejčastěji také v podobě futures kontraktů.
- Sektorové ETF jsou obdoba indexových ETFs, akorát nesledují index, ale konkrétní sektor, jako například energetický, finanční a podobně.
- ETF orientující se na určité regiony nebo státy [21].

1.2.1.4 Komodity

Jedná se o investici do zboží, které je vyráběné ve velkém množství různými výrobci a je obchodováno bez rozdílu v kvalitě. Kvalita je garantována bez ohledu na dodavatele komodity. Mezi nejdůležitější komodity patří:

- Kukuřice – základní plodina obsažená ve velkém množství potravin ve formě kukuřičného škrobu. Slouží také pro krmení dobytka.
- Zlato – šperky, elektronika.
- Ropa, zemní plyn – doprava, výroba potravin a v podstatě veškerého dalšího zboží.
- Bavlna a mnoho dalších...

Komodity byly zmíněny v kapitole 1.1.3, avšak pouze jako možnost investice do komodity jako takové. Druhým, již dříve zmíněným způsobem obchodování s komoditami je ETF z předchozí kapitoly 1.2.1.3. Zmíněna byla i nepřímá investice, a sice do akcií těžařských a dalších výrobních společností. Následovat bude kapitola 1.2.2.1 o futures kontraktech, které jsou dnes nejrozšířenějším způsobem obchodování s komoditami [5] [18].

1.2.1.5 Dluhopisy

Dluhopis je dluhový cenný papír, jehož koupě znamená, že kupující půjčuje eminentovi (společnosti, státu) dluhopisu určitou částku za určitých podmínek. Jedná se o zastupitelný cenný papír, s nímž je spojeno právo na vyplacení dlužné částky, stanovených výnosů a povinnost emitenta splnit veškeré závazky v den splatnosti dluhopisu. Dluhopisy mohou být vydány i se splatností například 15 let. Běžně lze nakupovat dluhopisy státní nebo podnikové. Investice do státních dluhopisů jsou považovány za bezpečnou investici s nízkou mírou výnosnosti [18] [22].

1.2.2 Deriváty

Princip obchodování s deriváty podkladových aktiv se od přímého obchodování s aktivy liší tím, že podmínky pro nákup či prodej podkladového aktiva se uzavřou v současnosti, avšak obchod se za takových podmínek zrealizuje až v budoucnu v přesně stanovený datum. Jde o takzvané termínované kontrakty, pevné nebo opční. Dále jsou vysvětleny jednotlivé typy derivátů, avšak pouze krátce, protože deriváty se tématu práce netýkají [18].

1.2.2.1 Futures

Futures kontrakt je dohoda dvou stran o nákupu či prodeji standardizovaného množství podkladového aktiva v předem specifikované kvalitě za danou cenu a k určitému budoucímu datu. Futures umožňují obchodovat nejenom s dříve zmíněnými komoditami 1.1.3, ale i s měnovými páry či akciemi. ETF může být také složeno z futures viz kapitola 1.2.1.3. Rozdíl oproti forwards z následující kapitoly je, že futures se obchodují pouze na burzách [18] [23].

1.2.2.2 Forwards

Forward kontrakt je velice podobný futures kontraktu, jedná se také o termínovanou dohodu dvou stran o nákupu či prodeji podkladového aktiva v určitém čase v budoucnosti za předem stanovenou cenu. Forwards však nepodléhají tak striktním pravidlům jako futures, nejsou totiž obchodovány na regulované burze, ale na over-the-counter (OTC) mimoburzovních trzích, které mají volnější pravidla. Existovaly v podstatě už dávno v minulosti, kdy zajišťovaly zemědělcům, že v budoucnu po sklizni prodají svoji úrodu a umožní jim tak třeba zbavit se svých závazků vůči těm, kteří jim svěřili prostředky pro setbu [18] [23].

1.2.2.3 Opce

Opce jsou v podstatě předkupní práva. Nákupem opce se získává předkupní právo koupit či prodat určitou věc v určitém termínu v budoucnosti a za

určitou předem dohodnutou cenu. Opce dává právo kontrakt uskutečnit během sjednané doby nebo termínu a nikoliv povinnost – na rozdíl od futures, kde obchod musí být podle dohodnutých podmínek realizován. Subjekt, který opci poskytuje, se nazývá vypisovatel. Druhá strana rozhodnutá pro nákup opce složí za opci určitou částku, která se nazývá opční premium. Tato částka jde rovnou vypisovateli a zůstává mu i v případě expirace dané opce. Analogií z běžného života může být složení zálohy za účelem rezervace zatím nedokončeného bytu. Složením zálohy (odpovídá opčnímu premiu) si rezervujeme právo na nákup bytu za dohodnutou cenu v určitém čase v budoucnosti (tedy opci). Pokud se rozhodneme dům nekoupit, přicházíme o zálohu [18] [24].

1.2.2.4 Spready

Spready jsou rozdílem mezi nákupní a prodejní cenou určitého podkladového aktiva. Může se jednat o ceny akcií, dluhopisů, měn, měnových párů, komodit, futures i opcí. V rámci těchto obchodů se spekuluje právě na změnu rozdílu v ceně. Velmi zajímavou variantou spreadů je obchodování komoditních spreadů či futures spreadů, jež ovlivňuje určitá sezónnost. Díky ní se zvyšuje pravděpodobnost úspěchu a výnosu. Na základě dlouhodobých historických dat se dá totiž velice dobře předpokládat, kdy nastane pokles či nárůst rozdílu v ceně. Obchodování je celkově pomalejší, ale také méně náročné na čas [25] [18].

1.3 Párové obchodování

V roce 1980 Nunzio Tartaglia ze společnosti Morgan Stanley vytvořil základ strategie párového obchodování. Jeho pracovní skupina se mimo jiné zabývala hledáním takových párů aktiv, jejichž ceny měly tendenci se pohybovat společně a takové páry byly označovány za obchodovatelné. Párové obchodování je tržně neutrální strategie, to znamená, že výsledky portfolia nezávisí na aktuálním vývoji trhu. Neutrální tržní portfolia jsou tvořena pouze dvěma aktivy s tím, že cílem je vydělat na rozdílu mezi cenami aktiv (spread). Na rozdíl od tradičního přístupu „nakup a drž“, jenž je založen na dlouhodobém držení nakoupených aktiv a na předpokladu, že trhy z dlouhodobé perspektivy nadále porostou, vstup do pozice v případě párového obchodování znamená současný nákup (long pozice) jednoho aktiva a prodej (short pozice) druhého aktiva. Prodej aktiva, které nevlastníme znamená jeho vypůjčení prostřednictvím brokera. Současný nákup a prodej tvoří zajištění, tedy alespoň částečnou ochranu před ztrátou. Pokud je hodnota spreadu odchýlena od střední hodnoty, vstupujeme do pozice a spekulujeme na její návrat, tedy sbližování (convergence), nebo rozbíhání (divergence) cen akcií. Existuje více druhů párového obchodování, nicméně diplomová práce se bude dále zabývat pouze statistickou arbitráží s akciemi jako podkladovým aktivem [26] [27]. Výhodami strategie jsou velké množství titulů k obchodování, již zmíněné zajištění před ztrátou, či nesměrovost strategie.

Výhodou může být i nižší časová náročnost. Naopak mezi nevýhody patří vyšší nároky na kapitál a potřeba vnímat výsledky v delším časovém horizontu (oproti například FOREX trhu). Potřeba sledovat informace týkající se obchodovaných titulů pak není ojedinělá pro párovou strategii.

1.3.1 Statistická arbitráž

Jako statistická arbitráž je označováno obchodování, které je založené na statistické analýze, jež pomáhá vybírat páry k obchodování. Dobře vybrané akciové páry jsou klíčem k úspěchu. Obchodník typicky pracuje s akciemi ze stejného sektoru, či odvětví, tedy akciemi firem, které si alespoň částečně konkurují. Akcie musí mít vysokou korelaci, být dostatečně likvidní. Denní ceny akcií tvoří časové řady. Pro párové obchodování není důležitý krátkodobý vývoj časových řad páru, ale cílem je nalezení takového páru, jehož časové řady se v dlouhodobém horizontu vrací k rovnovážnému stavu. Pokud mají dvě (a více) časových řad tendenci dlouhodobě mezi sebou udržovat konstantní rozdíl, respektive pohybovat se spolu, označujeme je jako kointegrované. Koncept kointegrace představil Clive W. J. Granger v roce 1981 [28], avšak klíčová je práce „Co-integration and error correction: Representation, estimation and testing“ z roku 1987 [29]. Od té doby je předmětem zkoumání aplikace kointegrace na reálná data z oblasti ekonomie, párové obchodování nevyjímaje. Clive W. J. Granger spolu s Robert F. Engle obdrželi v roce 2003 Nobelovu cenu v oblasti ekonomie právě za metody spojené s analýzou ekonomických časových řad. Analýza a návrh strategie párového obchodování lze rozdělit do tří kroků:

1. Výběr párů akcií, které by mohly být kointegrované.
2. Ověření hypotézy z předchozího kroku, a sice otestování kointegrace časových řad doporučených párů.
3. Výpočty pravidel, respektive signálů pro úspěšné obchodování [27].

Diplomová práce má za cíl vytvořit analytický nástroj, který ohodnotí a doporučí páry k obchodování. To odpovídá prvním dvěma bodům, které jsou dále rozvedeny.

Výběr párů

Tento proces je možné provést na základě údajů o akciích anebo aplikováním řady pravidel na historická data, která by určila potenciálně kointegrované páry. Jako vhodnější se jeví první možnost, protože cílem této fáze je především rychlé uspořádání množiny kandidátních párů pomocí jejich ohodnocení, neboli provedení scoringu. Požadavek na rychlost tohoto procesu je na místě, protože s ohledem na množství akcií na burzách se může počet kandidátních párů pohybovat až v řádech miliónů. Příkladem takového ohodnocení může být

korelace časových řad, respektive výpočet Pearsonova korelačního koeficientu [27]. Skóre (vzdálenost d) kandidátního páru akcií A, B se potom vypočítá ze vztahu:

$$d(A,B) = \rho_{A,B} = \frac{\mathrm{cov}\,(A,B)}{\sigma_A \sigma_B}$$

Testování kointegrace

Dalším krokem je samotné otestování kointegrace, zejména u párů s vysokým ohodnocením. Žádné ohodnocení však nedosahuje kvalit samotného testu kointegrace, tudíž páry s nižším ohodnocením nejsou zahazovány. Nejpoužívanější jsou Engle-Granger test a Johansenův test. Cílem je nalezení lineární kombinace nestacionárních časových řad, která je stacionární. Podmínkou testů jsou tedy nestacionární časové řady na vstupu. To lze ověřit hledáním jednotkového kořenu řady, jehož přítomnost znamená nestacionaritu řady. K hledání jednotkového kořene se používá Augmented Dickey Fuller (ADF) test, který bývá často doplněn dalšími testy, jako třeba Kwiatkowski, Phillips, Schmidt, Shinn (KPSS) testem stacionarity [30] [31] [32] [33].

1.4 Dostupný software

V současné době existuje mnoho analytického software a každý výrobce tvrdí, že jeho produkt je nejlepší. Někteří brokeři navíc nabízejí svá vlastní řešení. Jejich výhodou je, že jsou často bezplatné (samozřejmě po otevření účtu u brokera). Tato cesta má však i odvrácenou stranu a tou je závislost na platformě daného brokera. Stejné je to povětšinou se zdrojem dat, svázání se s daty od určitého brokera nemusí být tím nejlepším řešením. Data budou sice kvalitní a aktuální, nicméně broker si za ně také nechá odpovídajícím způsobem zaplatit. Dalším aspektem při výběru software může být komunita uživatelů. Ta se týká již zaběhnutých řešení a rozhodně jde o plus. Výrazně usnadňuje získávání informací, návodů, best practices a také řešení případných problémů či dotazů. Ještě důležitější roli může komunita hrát v případě platformy, která umožňuje vývoj a instalaci vlastní modulů za účelem rozšíření funkcionalit. Kapitola krátce přiblíží vybrané analytické aplikace s podporou párového obchodování [34].

1.4.1 Sierra Chart

Sierra Chart je velice oblíbený program pro zobrazování grafů. Vyniká stabilitou, jednoduchostí používání, možnostmi nastavení grafů a také kvalitní dokumentací. Podporuje všechny druhy trhů: akcie, futures, indexy, FOREX a další. Jeho nespornou výhodou je možnost integrace s externími službami, jak poskytovateli dat, tak brokery a lze z něj také zadávat obchodní příkazy. Jako příklad lze uvézt integraci s Interactive Brokers [35] (včetně dat) nebo

Obrázek 1.5: Sierra Chart prostředí [36]

CTS T4 Trading Platform Service za příplatek $10 za měsíc. Cena samotného software začíná na $24 měsíčně za základní verzi, s předplatným se měsíční poplatek snižuje. Software vyžaduje systém Microsoft Windows XP a novější [36] [37]. Ukázka prostředí na obrázku 1.5

1.4.2 Track'n Trade

Software od společnosti Gecko je velmi populárním analytickým programem. V nabídce jsou verze pro futures, FOREX a také pro akciové trhy. Právě poslední zmíněná verze je pro diplomovou práci zajímavá. Program obsahuje denní, týdenní a měsíční grafy, sleduje všechny populární akcie nebo ETF. Pro analýzu nabízí nespočet indikátorů, jako například SSTO (Slow Stochastics), MACD (Moving Average Convergention/Divergention). V případě nákupu prémiového balíčku jsou v ceně další doplňky, jako simulátor nastavených parametrů na reálných historických datech. Cena je v případě základní verze $197, v případě prémiové $997. K tomu je potřeba připočítat cenu za data, která při využití výhodného předplatného na dva roky činí $23 měsíčně. Je k dispozici 14denní trial verze bez nutnosti zadat platební kartu. Software podporuje pouze Microsoft Windows 7 a novější [38]. Ukázka prostředí na obrázku 1.6.

1.4.3 TradeStation

TradeStation je kompletní obchodní platforma a broker v jednom. Nabízí historická a real-time data všech akciových, futures, opčních a FOREX trhů

Obrázek 1.6: Track'n Trade prostředí [38]

[39] a k nim množství grafů a ukazatelů. Platforma poskytuje až 30 let historie dat pro akcie. Výhodou je vytváření vlastních doplňků díky programovacímu jazyku EasyLanguage. Výrobce k plnohodnotné desktopové aplikaci poskytuje i další možnosti přístupu na účet. TradeStation Web Trading slouží pro přístup přes webový prohlížeč, k dispozici jsou také mobilní aplikace pro operační systémy iOS a Android. Veškerý software má sjednocené moderní uživatelské rozhraní viz obrázek 1.7. Software je zdarma k brokerskému účtu bez dalších poplatků. Od roku 2018 je software poskytován zdarma i k účtu u Interactive Brokers (U.K.) Limited [35] [40].

1.4.4 AmiBroker

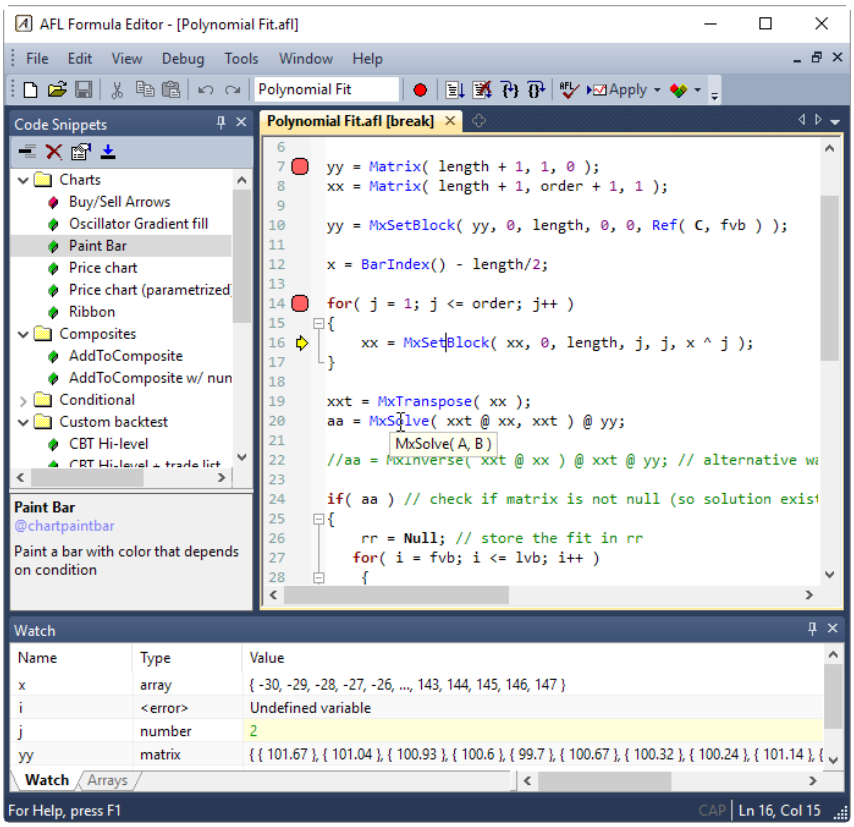
Analytický nástroj AmiBroker, primárně určený pro akciové trhy, se mírně odlišuje od předchozích programů, a sice tím, že je více orientován na backtesting a optimalizace obchodních strategií. Vyniká především svou rychlostí (napsán v jazyce C++) a díky AmiBroker Formula Language (AFL) jazyku, který umožňuje psát pokročilé skripty, je vysoce customizovatelný (náhled na obrázku 1.8). Stejně jako jiné nástroje poskytuje nespočet grafů a funkcí, dále umožňuje integraci s různými datovými zdroji, opět třeba s Interactive Brokers [35]. Náhled prostředí na obrázku 1.9. Pro spuštění programu postačuje sice i Microsoft Windows 2000, nicméně podporován je pouze právě Microsoft Windows. V nabídce jsou dvě verze, Standard, která má omezenou funkcionalitu a limit na alokování si prostředků, stojí $279. Naopak verze Professional

Obrázek 1.7: Trade Station Global prostředí [39]

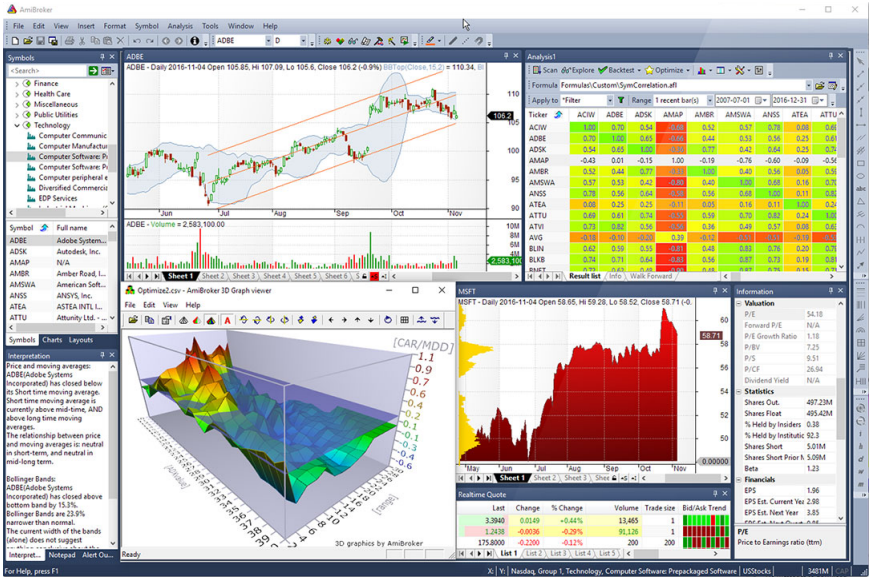
nemá žádná omezení a stojí $339 [41].

1.4.5 Shrnutí

Dle nabytých poznatků lze konstatovat, že na trhu existuje pokročilý software pro všechny možné druhy obchodování. Nevýhodou existujících programů je to, že jsou buď placené, anebo vyžadují otevřený účet od brokera. Jinak řečeno, není k dispozici bezplatný software pro bezplatnou analýzu dat a přípravu vlastní obchodní strategie. Při výběru té nejlepší platformy záleží na požadavcích uživatele. Je třeba si položit otázky, zda se chce uživatel vázat na konkrétního brokera, s čím vlastně chce obchodovat, jaká data mu budou dostačovat a zda potřebuje nespočet ukazatelů pro jeho strategii. Zájemce by měl také zvážit, zda alespoň do budoucna uvažuje o nějakém vlastním rozšíření aplikace, pokud ano, komunita a snadná tvorba vlastních modulů by měla hrát důležitou roli při výběru analytického software.



Obrázek 1.8: AmiBroker vývojové prostředí [41]



Obrázek 1.9: AmiBroker prostředí [41]

Kapitola 2

Analýza a návrh

Tato kapitola v úvodu identifikuje funkční i nefunkční požadavky na novou aplikaci. Následuje seznam případů užití spolu s jeho mapováním na funkční požadavky. Kapitola dále pokračuje doménovým modelem, návrhem uživatelského rozhraní a modelem architektury aplikace. Návrhový model tříd spolu s modelem nasazení zakončují tuto kapitolu.

2.1 Model požadavků

Na základě poznatků z předchozí kapitoly, zejména kapitol o párové strategii 1.3 a průzkumu existujícího analytického software 1.4.5 lze stanovit funkční i nefunkční požadavky na novou aplikaci. Další požadavky plynou přímo ze zadání diplomové práce. Shrnutím je obrázek 2.1.

2.1.1 Funkční požadavky

Funkční požadavky slouží k vymezí všech primárních funkcí systému.

F1 – Načtení dat o akciích

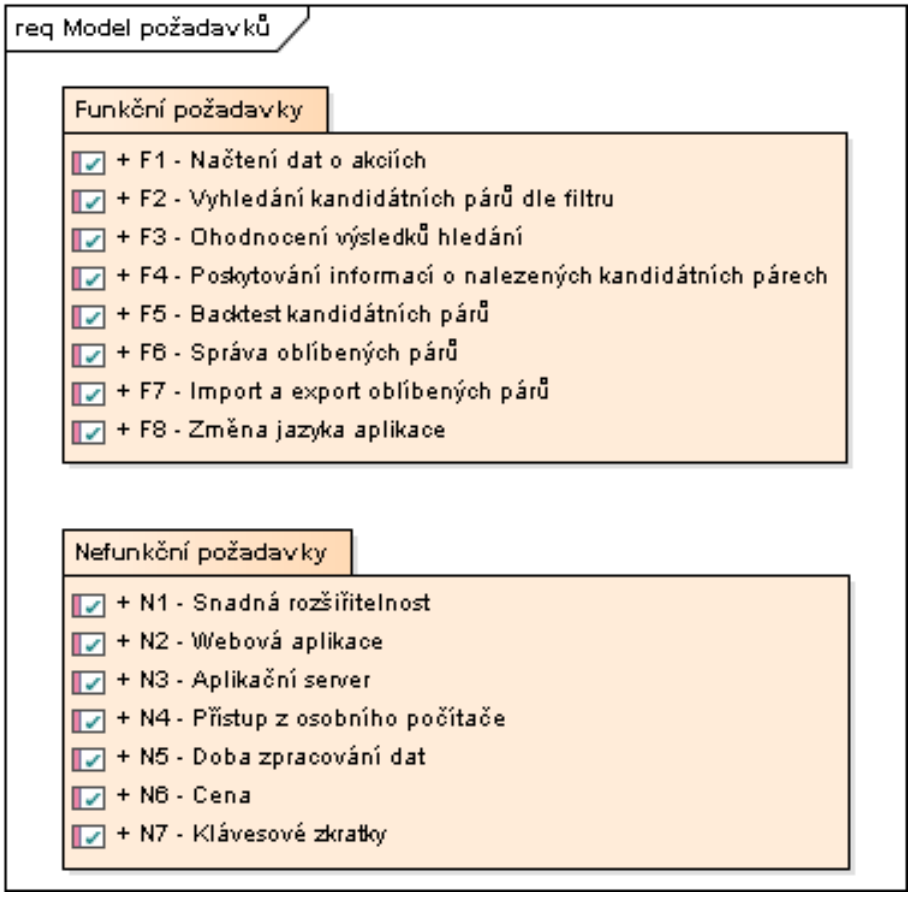
Aplikace musí umět načítat aktuální denní data o akciích z internetu pomocí standardních protokolů.

F2 – Vyhledání kandidátních párů dle filtru

Hledat kandidátní páry musí být možné specifikací více parametrů a zmenšit tak množinu výsledků.

F3 – Ohodnocení výsledků hledání

Nalezené kandidátní páry musí být automaticky ohodnocené podle takového klíče, který co možná nejlépe vyjádří kvalitu každého nalezeného páru.



Obrázek 2.1: Model požadavků

F4 – Poskytování informací o nalezených kandidátních párech

Aplikace musí umět dopočítat, či jinak získat relevantní informace o kandidátním páru a ty přehledně zobrazit, například formou grafů. Kapitola 1.3 popisuje takové informace.

F5 – Backtesting kandidátních párů

Backtesting, neboli testování strategie na reálných historických datech, musí být součástí aplikace.

F6 – Správa oblíbených párů

Aplikace musí podporovat přidání a odebrání kandidátních párů ze seznamu nalezených kandidátních párů do seznamu oblíbených párů.

F7 – Import a export oblíbených párů

Export oblíbených párů musí být dostupný a musí být prováděn ve formátu JSON s pevně danou strukturou, která bude umožňovat následný import dat bez nutnosti dalších úprav souborů.

F8 – Změna jazyka aplikace

Aplikace musí být dostupná ve dvou jazycích, a sice českém i anglickém. Výjimku mohou tvořit slova odborná nebo ustálená, případně slova nepřeložitelná.

2.1.2 Nefunkční požadavky

Nefunkční požadavky nesouvisí přímo s primární funkčností systému, ale i tak jsou důležité pro vývoj aplikace.

N1 – Snadná rozšiřitelnost

Aplikace musí být snadno rozšiřitelná o nové poskytovatele dat o akciích, například využitím API brokera. Tento požadavek přímo plyne ze zadání práce.

N2 – Webová aplikace

Výsledná aplikace bude dostupná ve formě webové aplikace. Zadání diplomové práce specifikuje použití technologie Java EE verze 8.

N3 – Aplikační server

Webová aplikace bude nasaditelná na běžně dostupný open source aplikační server GlassFish verze 5 [42].

N4 – Přístup z osobního počítače

Aplikace bude přístupná z klientského počítače pomocí webového prohlížeče.

N5 – Doba zpracování dat

Doba odezvy webové aplikace nesmí překročit 2 sekundy. Výjimku tvoří načítání dat z externích zdrojů, kdy je doba silně závislá na konkrétní službě. Nicméně i v takovém případě musí aplikace odpovídajícím způsobem včas reagovat.

N6 – Cena

Aplikaci musí být možné používat zcela zdarma. S tím souvisí požadavek N1, který zajišťuje snadnou výměnu některých částí aplikace za třeba kvalitnější, avšak placené služby.

N7 – Klávesové zkratky

Podpora klávesových zkratek pro pohyb v seznamech výsledků a oblíbených položek musí být implementována. Konkrétně směrové klávesy nahoru a dolů.

2.1.3 Shrnutí

Na základě výčtu funkčních i nefunkčních požadavků lze konstatovat, že ideální analytický nástroj pro párové obchodování s akciemi by byla aplikace s jednoduchým a přehledným uživatelským rozhraním, disponující vhodnými daty a potřebnými ukazateli pro strategii párového obchodování (viz 1.3). Strategie nepotřebuje nejaktuálnější data v jednotkách sekund ani minut, ale plně dostačují denní data. Aplikace by měla být snadno integrovatelná s různými datovými zdroji. V neposlední řadě musí být provozovatelná zdarma. Tyto požadavky nenaplňuje žádný existující software.

Většina z nich vyžaduje od uživatele zaplatit určitou částku, za kterou sice poskytuje univerzálnost pro různé druhy obchodování i strategií a k nim nespočet možností, naprostá většina z nich však není potřeba. Vzhledem k placeným službám brokerů je cílem diplomové práce vytvoření plně funkčního prototypu čistě analytické aplikace, to znamená bez funkcionality sloužící k napojení na brokera a s tím související zadávání obchodních příkazů.

2.2 Případy užití

V této sekci jsou představeny případy užití (use cases), které slouží k popisu funkcionality systému. Účastníkem je uživatel aplikace. Tabulka 2.1 obsahuje seznam případů užití a jejich mapování na požadavky definované v kapitole 2.1.

Tabulka 2.1: Přehled realizace požadavků případy užití

	Název případu užití	**Požadavek**
UC1	Zobrazení dostupných burz	F1
UC2	Načtení historických dat o akciovém páru	F1
UC3	Nalezení akciových párů v rámci sektoru burzy	F2
UC4	Nalezení akciových párů v rámci odvětví burzy	F2

Tabulka 2.1: Přehled realizace požadavků případy užití

UC5	Zadání údajů o akciích za účelem filtrování kandidátních párů	F2
UC6	Identifikace nejlepších kandidátních párů ve výsledcích hledání	F3
UC7	Srovnání kvality dříve uložených párů s výsledky hledání	F3
UC8	Zobrazení přehledu vybraného kandidátního páru	F4
UC9	Zobrazení metadat o vybraném kandidátním páru	F4
UC10	Zobrazení vypočtených statistických údajů o kandidátním páru	F4
UC11	Zobrazení grafu vývoje volume (objem obchodů) akcií kandidátního páru	F4
UC12	Zobrazení grafu vývoje korelace akcií kandidátního páru	F4
UC13	Zobrazení grafu vývoje cen akcií kandidátního páru	F4
UC14	Provedení backtestu kandidátního páru	F5
UC15	Definování strategie obchodování pro účely backtestu páru	F5
UC16	Zobrazení výstupních ukazatelů z backtestu kandidátního páru	F5
UC17	Zobrazení grafu vývoje zisku/ztráty pro definovanou strategii obchodování na historických datech páru	F5
UC18	Přidání kandidátního páru z výsledků hledání do oblíbených párů	F6
UC19	Odebrání kandidátního páru z oblíbených párů	F6
UC20	Řazení oblíbených párů v seznamu dle kvality	F6
UC21	Export oblíbených položek párů	F7
UC22	Import oblíbených položek párů	F7
UC23	Přepnutí jazyka aplikace	F8

2.3 Doménový model

V této sekci jsou popsány entity doménového modelu, jejich důležité atributy a vztahy. Diagram se vztahy mezi entitami lze najít na obrázku 2.2.

Stock Code

Třída reprezentuje symbol akcie. Má dva atributy, a sice kód akcie a metadata, která obsahují všechna metadata získaná aplikací a týkající se dané akcie.

Pair Candidate

Základem této třídy reprezentující kandidátní pár jsou dva atributy typu StockCode. Důležitým atributem je skóre hodnotící kvalitu páru.

Stock List

Třída reprezentující seznam akcií dostupných na konkrétní burze identifikované jménem. Obsahuje seznam dostupných symbolů akcií a jejich zařazení do odvětví a sektorů.

Stock List Element

Třída reprezentuje položku v seznamu akcií třídy Stock List. Je vázán na Stock Code. Obsahuje atributy sektor a odvětví v rámci konkrétní burzy.

Stock Filter

Jedná se o třídu obsahující atributy odpovídající vyhledávacímu formuláři. Například název burzy, zvolené odvětví, sektor nebo minimální a maximální hodnotu cen akcií, jejich poměru a další atributy. Nechybí ani časový stop loss (maximální doba obchodu).

Day Price

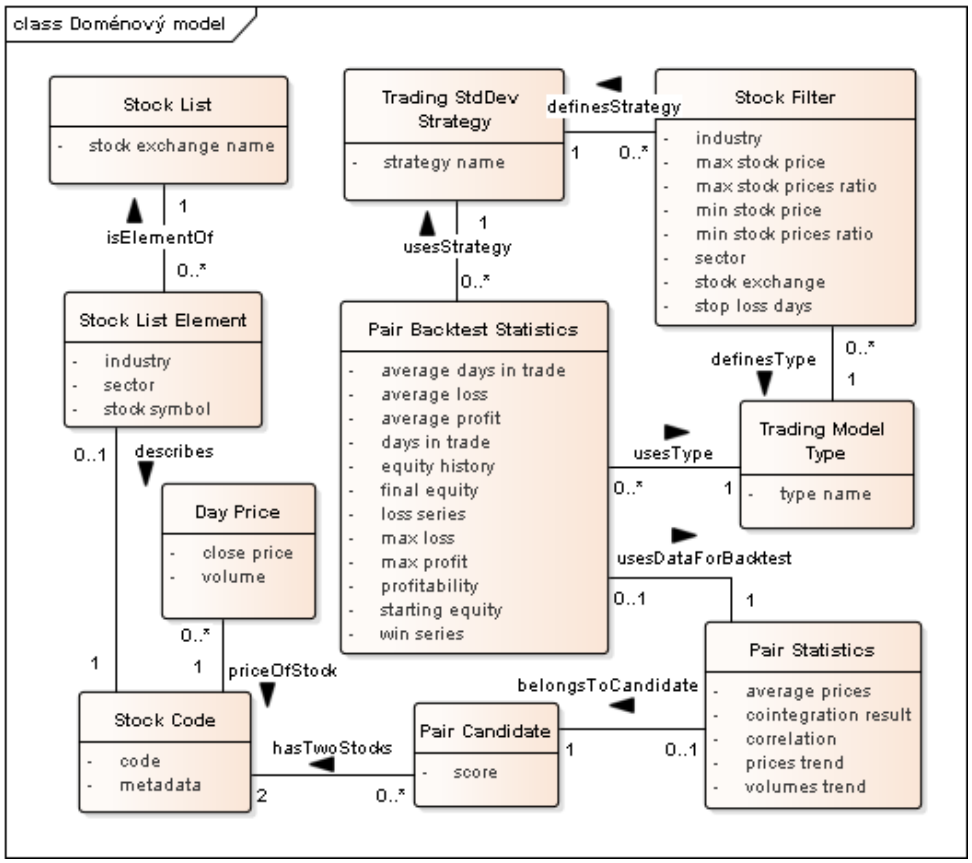
Třída uchovává načtená denní data pro akcii, respektive Stock Code. Obsahuje především cenu akcie na konci dne spolu s volume (objem obchodů) pro konkrétní den.

Pair Statistics

Třída reprezentující vypočítané statistiky o kandidátním páru. Konkrétně vývoj a průměrnou hodnotu cen, objemů obchodů anebo korelaci. Obsahuje také výstup testu kointegrace.

Pair Backtest Statistics

Jedná se o statistiky spojené s backtestem a jeho výstupem. Mezi atributy patří výnosnost, maximální a průměrný zisk, případně ztráta, průměrná délku obchodu a nejdelší série zisků, či ztrát. Dále stav účtu před začátkem a na konci backtestu, spolu s jeho vývojem během backtestu.



Obrázek 2.2: Doménový model

Trading Model Type

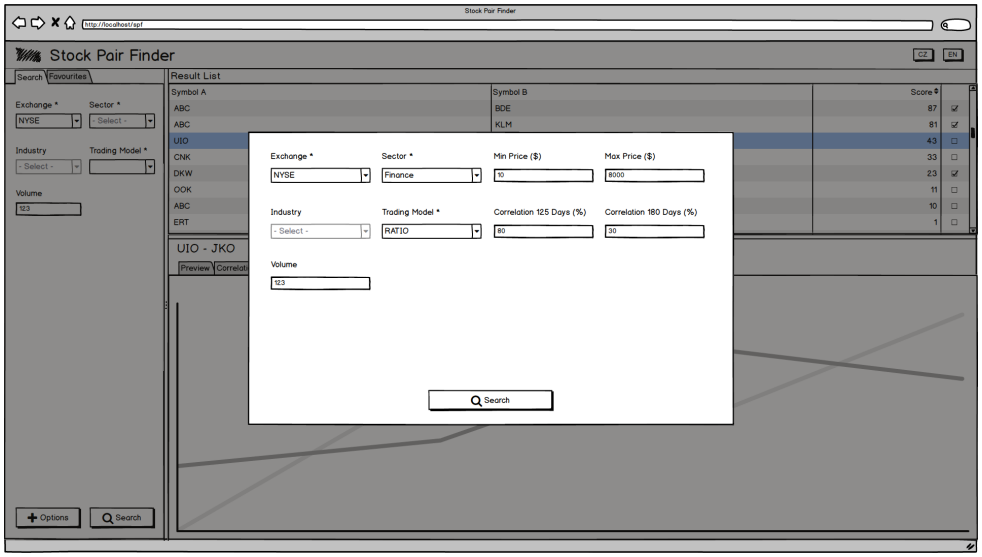
Enum určující typ strategie pro backtest. Na výběr jsou modely pro použití rozdílu, nebo poměru cen akcií páru.

Trading StdDev Strategy

Enum definující momenty vstupů a výstupů do pozic (realizace obchodu). Na výběr je vstupování do pozice při odchylce dvojnásobku směrodatné odchylky od klouzavého průměru zvoleného Trading Model Type a výstupy na hodnotě klouzavého průměru, nebo plus, respektive mínus hodnota směrodatné odchylky.

2.4 Uživatelské rozhraní

Cílem této sekce je návrh co možná nejintuitivnějšího grafického uživatelského rozhraní (GUI). Tento požadavek je obzvláště důležitý s ohledem na



Obrázek 2.3: Wireframe: vyhledávací dialog

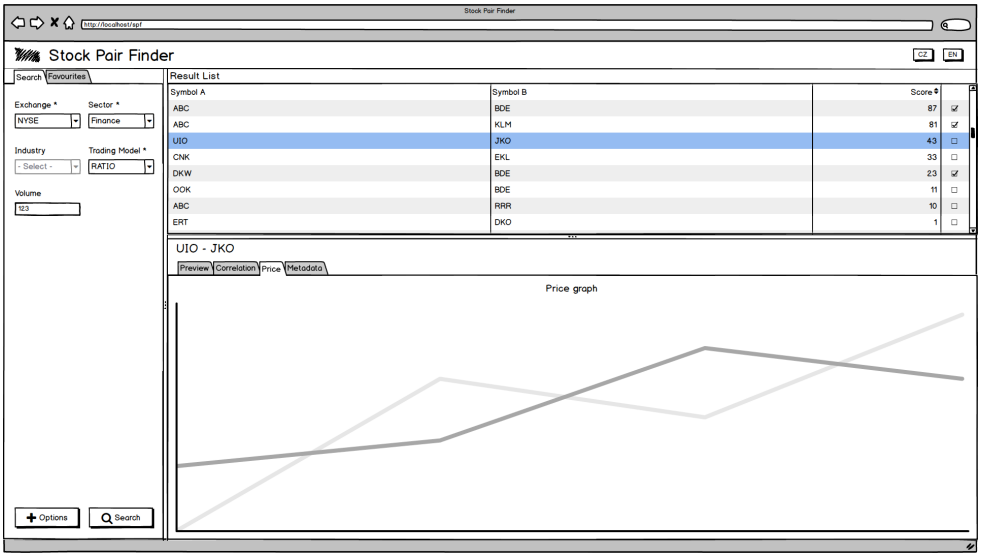
fakt, že aplikace necílí pouze na IT profesionály a zkušené uživatele. Aplikace by měla být snadno ovladatelná i pro běžného uživatele zvyklého používat klasické desktopové aplikace.

2.4.1 Wireframes

Při návrhu jakéhokoliv uživatelského rozhraní (anglicky user interface, UI) je vhodné začít vytvořením wireframes. Výhodou je jednoduchost návrhu a snadné odhalení případných nedostatků, jejichž odstranění ve fázi návrhu je nejenom snadnější, ale především mnohem méně nákladné než ve fázi vývoje, či dokonce po uživatelském otestování hotové aplikace. Wireframes lze realizovat jednoduše pomocí papíru a tužky nebo pomocí specializovaného software. V této diplomové práci byl použit nástroj Balsamiq Mockups 3 for Desktop od společnosti Balsamiq Studios [43]. Software umožňuje jednoduchou tvorbu wireframes viz prvotní náhledy obrazovek popsané níže.

Obrazovka 2.3 zobrazuje rozšířený vyhledávací dialog (klasický vyhledávací formulář lze vidět vlevo pod dialogem). Tato obrazovka slouží nejenom pro zobrazení dostupných burz a spuštění vyhledávání na základě filtru, ale také obsahuje políčka pro definici strategie pro následný backtesting. Dialog tedy pokrývá UC1, UC2, UC3, UC4, UC5 a také UC15.

Druhý wireframe na obrázku 2.4 představuje obrazovku aplikace po provedení vyhledávání. Po vynechání levého vyhledávacího panelu popsaného u předchozího modelu, lze vidět seznam výsledků, respektive ohodnocených párů seřazených dle skóre. To pokrývá především UC6. V pravém horním rohu se nacházejí tlačítka pro změnu jazyka v souladu s UC23.



Obrázek 2.4: Wireframe: seznam výsledků

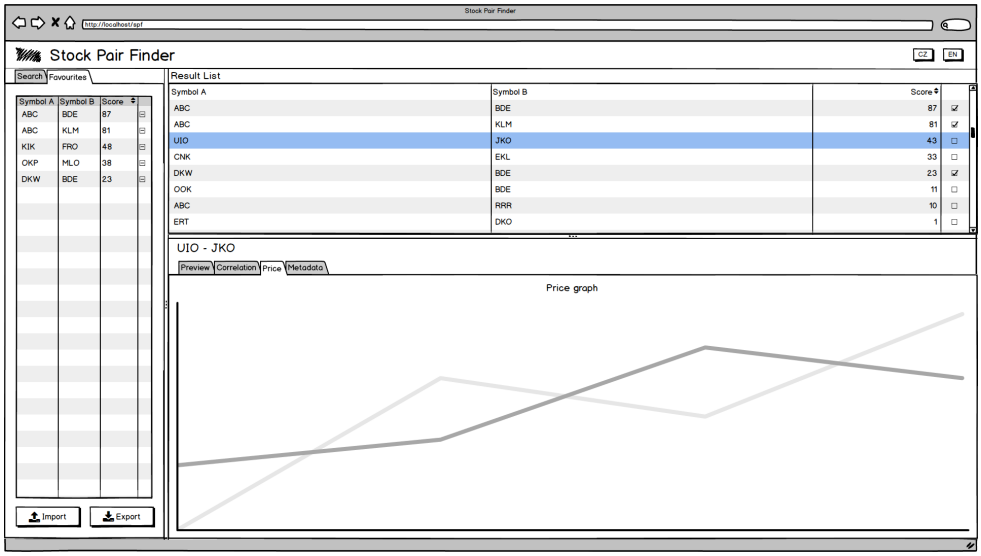
Hlavní část obrazovky však zaujímá detail vybraného páru ze seznamu výsledků. Je zde několik panelů, které slouží k zobrazování přehledů a grafů o kandidátním páru. Na obrázku je ukázka vývoje cen akcií páru. Tyto panely tedy pokrývají UC8, UC9, UC10, UC11, UC12, UC13, UC16, UC17 a také UC14, protože z panelu přehledu je možné spustit backtest.

Poslední ukázka uživatelského rozhraní aplikace, obrázek 2.5, se odlišuje panelem v levé části obrazovky. Ten slouží pro správu oblíbených, přidání ze seznamu výsledků (oblíbené páry označeny vpravo v seznamu výsledků), odebrání páru z oblíbených a jejich řazení. To pokrývá UC18, UC19, UC20. Panel umožňuje také export oblíbených a import dříve uložených párů. Tento panel tedy pokrývá i UC7, UC21 a UC22.

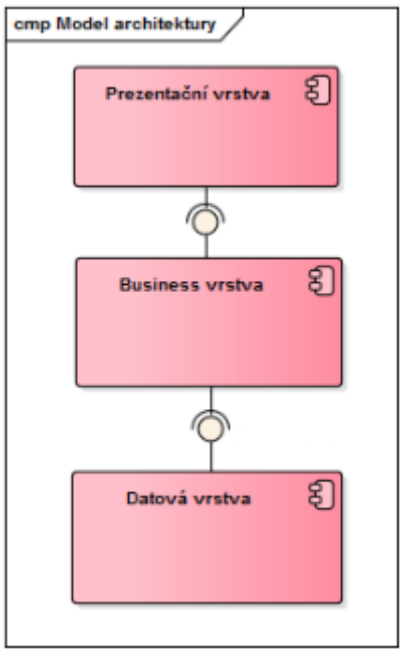
2.5 Model architektury

Architektura aplikace je navržena jako třívrstvá. Komunikaci mezi vrstvami zajišťují rozhraní (interface). Všechny třídy dodržují Dependency Inversion princip [44] a nejsou závislé na konkrétních implementacích rozhraní. To výrazně usnadňuje výměnu jednotlivých vrstev, respektive jejich implementací. Nefunkční požadavek N1 vyžaduje snadnou rozšiřitelnost o další poskytovatele dat. Toho lze v navržené třívrstvé architektuře dosáhnout výměnou implementace datové vrstvy, která se stará o načítání dat.

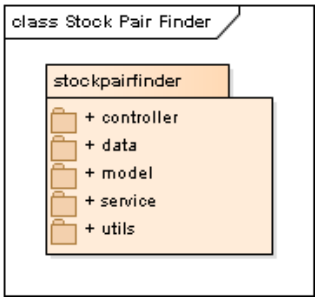
Prezentační vrstva obsahuje třídy a komponenty, které zajišťují prezentování informací uživateli, přebírání požadavků od uživatele a jejich předávání dále do business vrstvy.



Obrázek 2.5: Wireframe: oblíbené položky



Obrázek 2.6: Model architektury



Obrázek 2.7: Balíček *stockpairfinder*

Business vrstva obsahuje veškerou logiku aplikace, spolu s výpočty, k jejichž výstupům přistupuje prezentační vrstva pomocí rozhraní business třídy. Prezentační vrstva je tak nezávislá na konkrétní implementaci. Umístění veškeré logiky do business vrstvy také výrazně zjednodušuje její jednotkové testování.

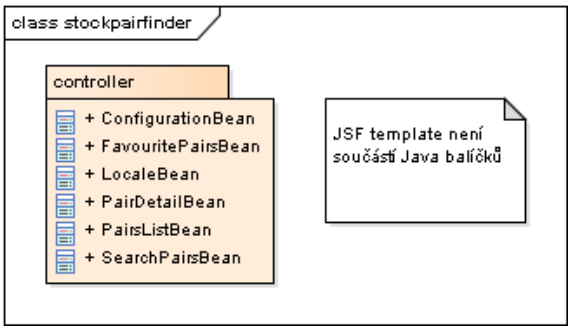
Datová vrstva obsahuje třídy a komponenty pro získávání dat o dostupných burzách a pro načítání denních dat o konkrétních akciích. Vrstva zajišťuje také import a export oblíbených kandidátních párů (funkční požadavek F7).

2.6 Návrhový model tříd

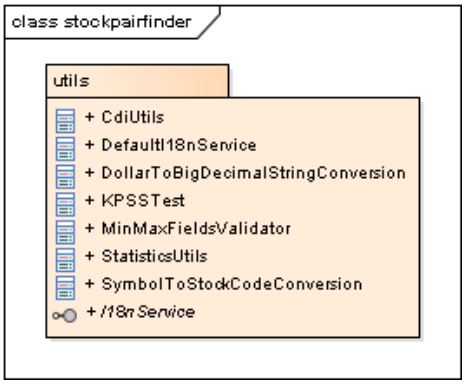
Tato sekce obsahuje popis jednotlivých balíčků, rozhraní a tříd, které realizují navrženou aplikaci Stock Pair Finder. Hlavní balíček *stockpairfinder* zobrazuje diagram 2.7.

Prezentační vrstva se skládá ze dvou komponent. Jednak z balíčku *controller*, viz obrázek 2.8, který obsahuje managed beans a zároveň reprezentuje Controller v MVC (Model–view–controller) vzoru. Druhou komponentou patřící do prezentační vrstvy jsou JavaServer Faces (JSF) pages, které nejsou součástí Java balíčků, ale v modelu MVC reprezentují View. JSF pages používají templates, to znamená, že stránka je rozdělena do několika logických celků, jež spojuje *main-template.xhtml*, který obsahuje například hlavičku stránky, či definuje layout, tedy rozdělení stránky na jednotlivé celky, které mají vlastní kontrolery právě v balíčku *controller*. Například *LocaleBean.java* se stará o nastavený jazyk prostředí nebo *PairDetailBean.java* stojí za stránkou zobrazující detail vybraného akciového páru.

Balíček *utils* obsahuje pomocnou funkcionalitu, která je používána v celé aplikaci. Na obrázku 2.9 je přehled tříd z balíčku. Jedná se o třídy poskytující matematické nástroje anebo o rozhraní *I18nService* spolu s implementací, které slouží pro načítání přeložených textů z *resource-bundle* v závislosti na aktuálním jazyku aplikace. Rozhraní je používáno jak v JSF pages za účelem překladu uživatelského rozhraní, tak například v *controller* balíčku pro lokalizaci chybových hlášek zobrazovaných uživateli.



Obrázek 2.8: Balíček *controller*

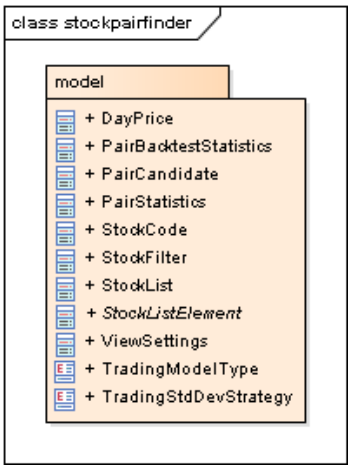


Obrázek 2.9: Balíček *utils*

Balíček *model* 2.10 obsahuje třídy reprezentující datový model z kapitoly 2.3. Vztahy mezi třídami zobrazuje diagram 2.11. Pro přehlednost byly odstraněny getters a setters.

Balíček *service* reprezentuje business vrstvu, obsahuje tedy rozhraní a jejich implementace realizující logiku. Výčet rozhraní a implementací zobrazuje obrázek 2.12. Každé rozhraní se stará o určitý logický celek, jako například *BacktestService.java* poskytuje metody pro backtest anebo rozhraní *StockDataService.java*, jenž slouží pro obsluhu dat a jejich načítání z datové vrstvy. Využití rozhraní prezentační vrstvou lze vidět na obrázku 2.13, kde jako příklad slouží *PairDetailBean.java* a *SearchPairsBean.java* z balíčku *controller*.

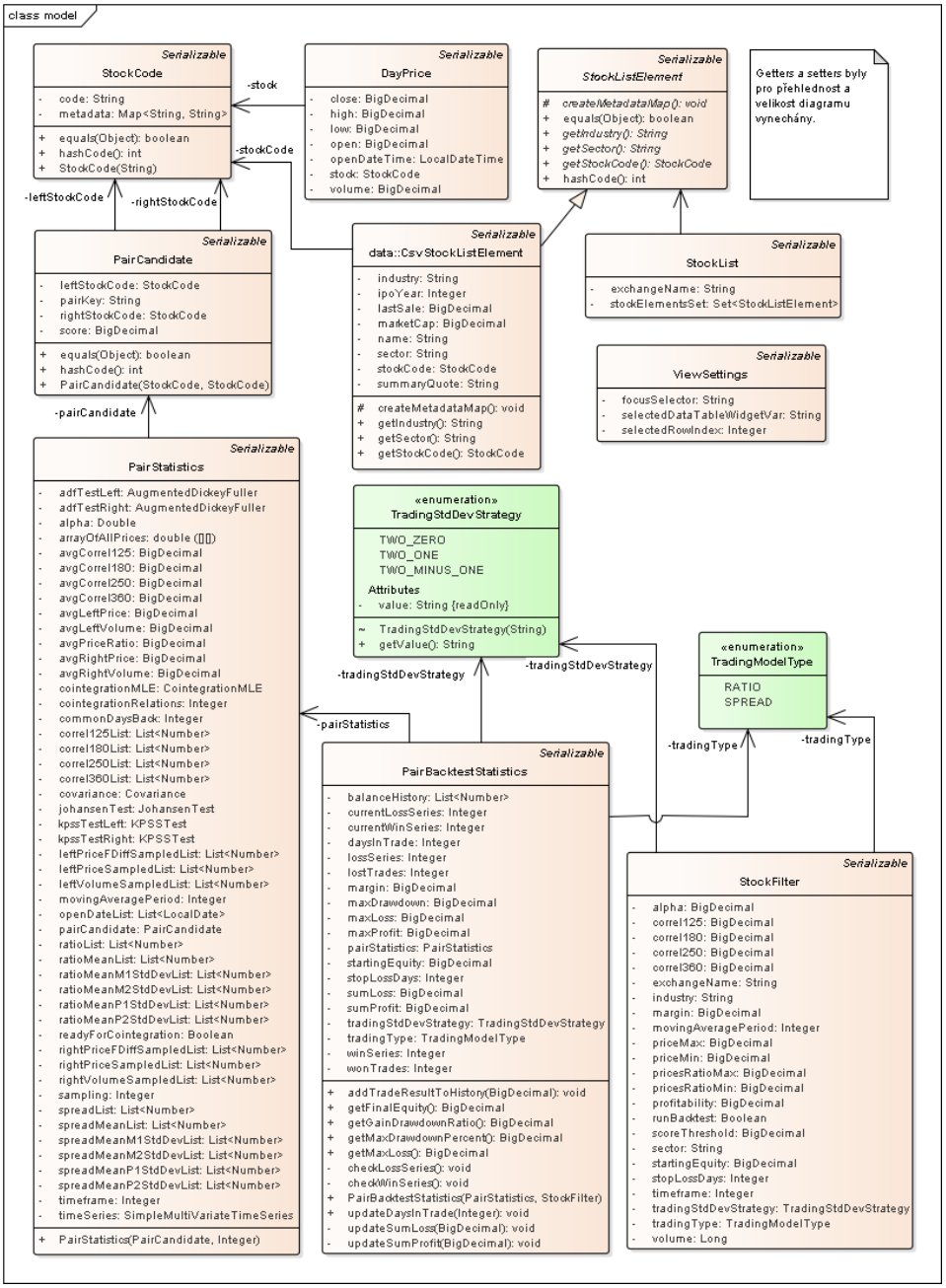
Datovou vrstvu reprezentuje balíček *data*. Jak znázorňuje obrázek 2.14, balíček obsahuje celkem tři rozhraní. *JsonService.java* je používané rozhraním *FavouritePairsService.java* a slouží pro import a export oblíbených kandidátních párů do formátu JSON. Druhým rozhraním je *StockListLoader.java*, které obstarává načítání seznamu burz a na nich dostupných akcií. Třetí rozhraní *StockLoader.java* má za úkol načtení denních dat pro vybrané akcie. Poslední dvě rozhraní, respektive jejich implementace byly blíže rozebírány v kapitole 3.3.1. Obrázek 2.15 přináší bližší pohled na třídy balíčku *data*.



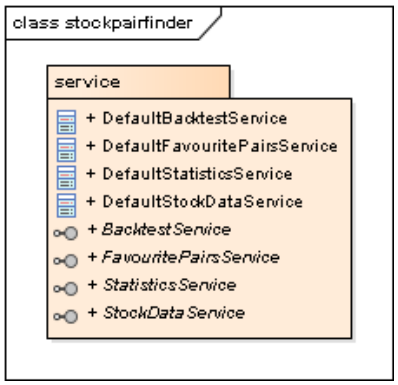
Obrázek 2.10: Balíček *model*

2.7 Model nasazení

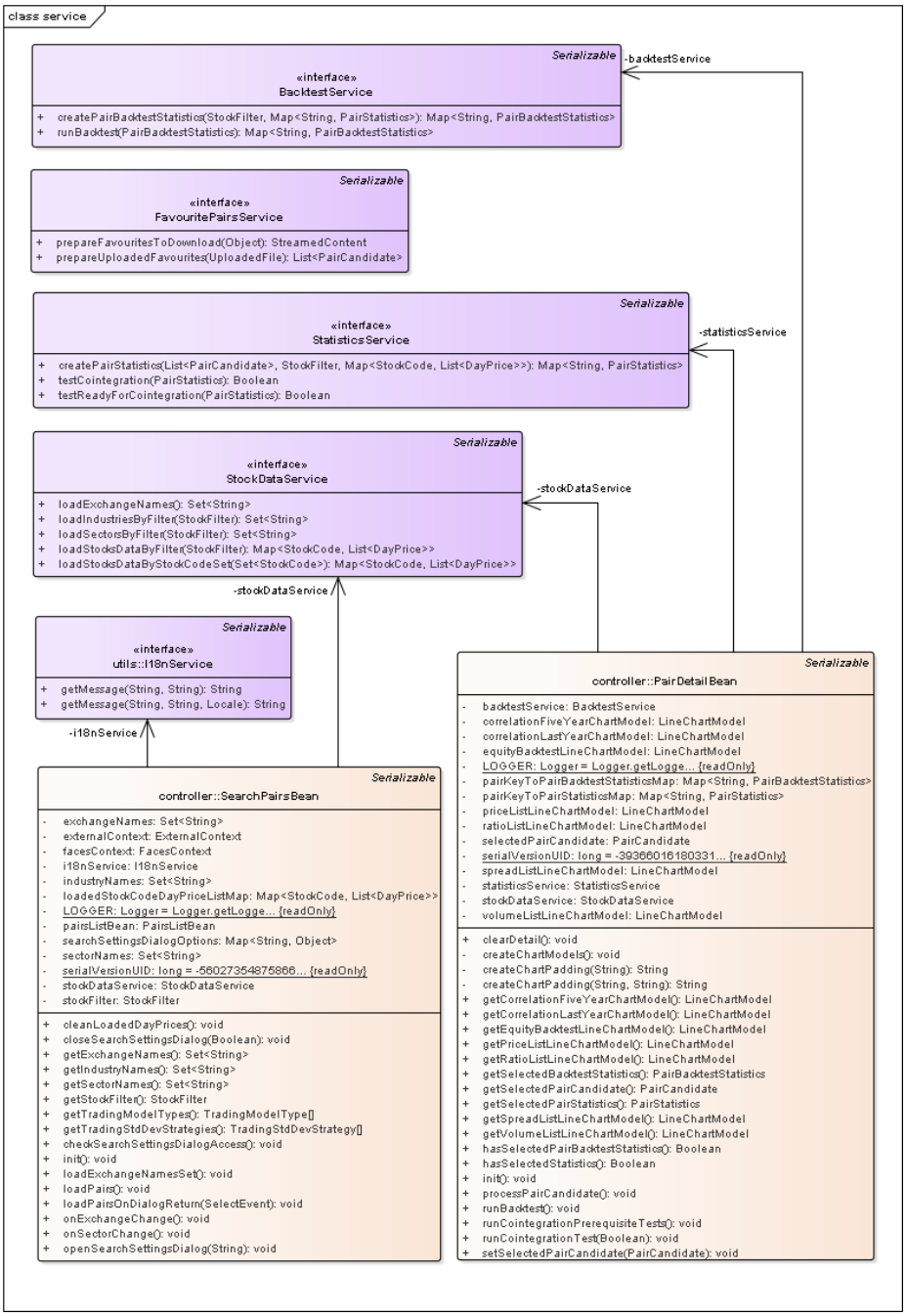
Dle požadavků je aplikace přístupná přes webové rozhraní prohlížeče, není tedy potřeba žádné instalace na osobní počítač a zároveň je tím zaručena nezávislost na platformě. Aplikace StockPairFinder.war běží na aplikačním serveru GlassFish 5. Konfigurace aplikace se provádí úpravou souboru *config.properties* v adresáři *WEB-INF*. Spuštěná aplikace je přístupná na adrese *http://server/spf*. Model nasazení je znázorněn na obrázku 2.16.



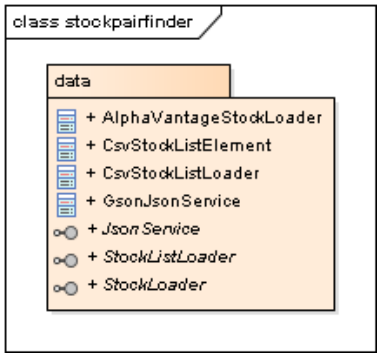
Obrázek 2.11: Diagram balíčku *model*



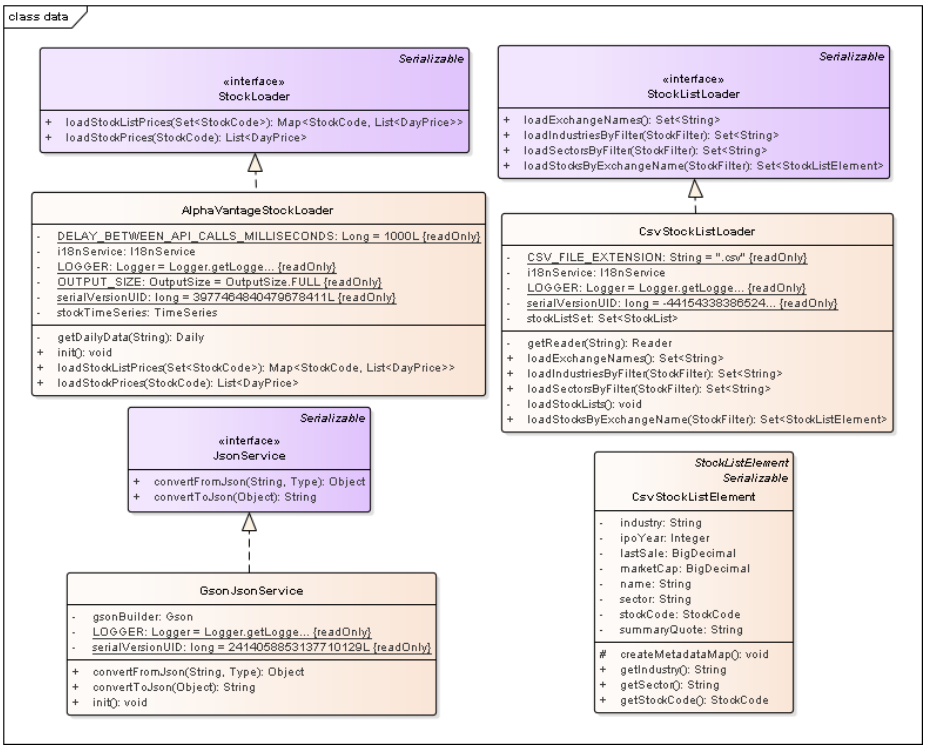
Obrázek 2.12: Balíček *service*



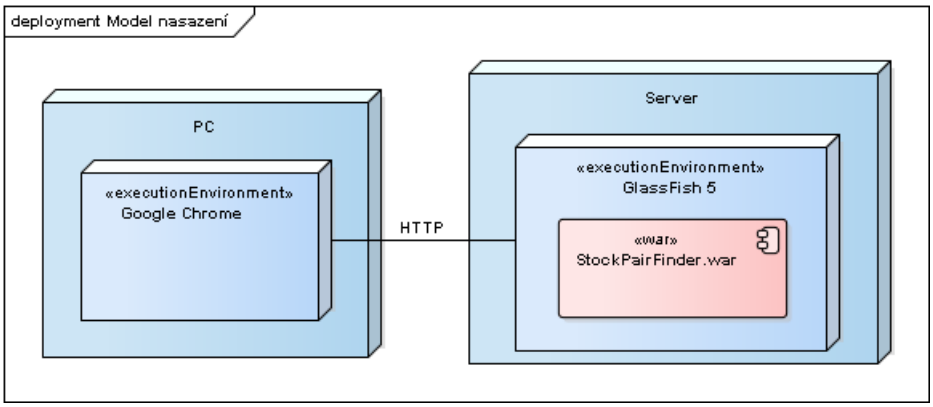
Obrázek 2.13: Diagram závislostí *PairDetailBean.java, SearchPairsBean.java*



Obrázek 2.14: Balíček *data*



Obrázek 2.15: Diagram balíčku *data*



Obrázek 2.16: Model nasazení

Kapitola 3

Realizace

Kapitola se věnuje realizaci analytického nástroje dle návrhu z kapitoly předchozí. Nejdříve jsou vybrány vhodné technologie i s ohledem na zadání diplomové práce. Následuje problematika analýzy časových řad cen akcií. Dále se kapitola věnuje zajímavým úskalím, která doprovázela implementaci. V závěru kapitoly je provedena Nielsenova heuristická analýza a konečně ukázka výsledné aplikace.

3.1 Technologie

Základní požadavky na technologie jsou jasně specifikované. Plyne z nich, že cílem je webová aplikace naprogramovaná pomocí jazyka Java EE 8. Jedná se o aktuálně poslední verzi Java EE, která obsahuje mnoho novinek. Nefunkčním požadavkem také je, že aplikace bude nasaditelná na open source aplikační server GlassFish aktuální verze 5, která implementuje Java EE verze 8. Jedná se o referenční implementaci [42]. Pro správu projektu byl použit nástroj Apache Maven verze 3.5 [45].

Při implementaci aplikace byla použita specifikace JSR-346 Contexts and Dependency Injection for the Java EE [46] (CDI), která je ve verzi 2.0 součástí Java EE 8. Koncept dependency injection je v podstatě implementace tzv. inversion of control principu, kde dependency (závislost) je objekt poskytující službu a injection (dosazení) je předání reference na závislost konzumentovi, neboli závislému objektu, který dosazený objekt použije (anotace *@Inject*). CDI odstraňuje pevné vazby mezi jednotlivými komponentami, a tak se výměna implementace určitého rozhraní obejde zcela bez zásahu do ostatních tříd. K tomu slouží anotace *@Alternative* a konfigurační soubor *beans.xml* [47]. Nezávislost komponent navíc usnadňuje testování. Životní cyklus, tedy tvorba a destrukce, komponent (beans) je řízen pomocí CDI kontejneru v rámci aplikačního serveru. Například kontrolery mohou použít anotaci *@SessionScoped* pro existenci instance pouze v rámci konkrétní session, naopak třídy servisní vrstvy typicky použijí anotaci *@ApplicationScoped* pro jednu instanci v rámci

CDI kontejneru pro všechny session. CDI beans jsou integrovány s Unified Expression Language (EL), který umožňuje snadný přístup k instancím tříd s anotací *@Named* přímo z JSP, respektive JSF stránek. Například výraz *value="${myBean.myField}"* umožňuje přistoupit k proměnné anebo výraz *${myBean.addNewOrder('orderName')}* volá metodu s parametrem [48] [49].

Pro tvorbu uživatelského rozhraní se přímo nabízí technologie JSF [50], která postupně nahrazuje technologii JavaServer Pages (JSP) a navíc se v rámci Java EE 8 objevila v nové verzi 2.3. Avšak pro intuitivnější rozhraní a přímočařejší vývoj je vhodné JSF doplnit snadno integrovatelnými knihovnami. S ohledem na funkční požadavky a wireframe byly vybrány následující knihovny.

PrimeFaces

PrimeFaces je jedna z nejpopulárnějších UI knihoven v Java EE světě. Používají ji velké firmy, banky i univerzity [51]. Oblíbená je především pro svoji jednoduchost, rychlost, množství předem připravených komponent, snadnou integrovatelnost se Spring Framework [52] a nakonec i pro velkou komunitu uživatelů [53]. Vývoje se řídí heslem „A good UI component should hide complexity but keep the flexibility“. Diplomová práce používá verzi 6.1, která je kompatibilní s JSF 2.3. Knihovna je open source [54].

PrimeFaces Themes

Jedná se o zdarma dostupnou knihovnu s předem připravenými tématy pro PrimeFaces. Vedle toho si lze zakoupit již hotová a modernější témata, nebo si vytvořit vlastní design. To vše díky integraci PrimeFaces knihovny s jQuery ThemeRoller CSS frameworkem [55] [56].

PrimeFaces Extensions

Jak už název napovídá, jde o nadstavbu PrimeFaces a přidává nové grafické komponenty a utility, které chybí komunitě kolem PrimeFaces. Diplomová práce používá verzi 6.1.1, která je kompatibilní s PrimeFaces verze 6.1. Knihovna je open source [57].

OmniFaces

OmniFaces je utility knihovna, jejímž účelem je usnadnění práce s JSF 2. Knihovna je open source a je kompatibilní s PrimeFaces [58]. Diplomová práce používá verzi 3.0 s podporou Java EE 8, tedy i CDI 2.0 a JSF 2.3.

ChartistJSF

Jedná se o knihovnu založenou na oblíbené JavaScript knihovně Chartist.js [59] pro tvorbu jednoduchých responzivních grafů. ChartistJSF je v podstatě

nadstavba nad knihovnou PrimeFaces a je tak snadno použitelná v moderních aplikacích založených na JSF. Knihovna PrimeFaces sama o sobě poskytuje grafové komponenty, nicméně na dnešní dobu poměrně zastaralé a obtížně modifikovatelné. Výhodou ChartistJSF je použití SVG formátu a s tím související snadná modifikovatelnost pomocí JavaScriptu (JS) a CSS. ChartistJSF navíc podporuje JS rozšíření pro původní Chartist.js. Knihovna je kompatibilní s JSF Ajax API a sama poskytuje eventy k zachycení [60]. Nevýhodou knihovny jsou omezené možnosti, které sice mohou být přidány pomocí JS rozšíření, ale existují i pokročilejší JS knihovny pro tvorbu grafů.

Testování

Jelikož testování je důležitou součástí všech projektů, byly vybrány dvě knihovny za účelem jednotkového otestování výsledné aplikace:

- JUnit [61] – knihovna sloužící k implementaci a spouštění jednotkových testů. Poskytuje mnoho funkcí pro porovnávání hodnot.
- Mockito [62] – framework pro tvorbu mocků. Mock je zástupný objekt, který se jeví jako normální instance určité třídy, avšak jeho chování lze definovat. Po vytvoření nemá mock žádné hodnoty. Slouží pro definování chování všech objektů, které testovaný úsek kódu potřebuje pro účely testovacího scénáře.

Z požadavků dále vyplývá nutnost integrace s poskytovateli dat a jejich Application Programming Interface (API). Java EE 8 sama o sobě poskytuje mnoho nástrojů pro integraci pomocí standardních protokolů s externími službami, tudíž tento požadavek nevyžaduje použití žádné další knihovny.

3.2 Analýza dat

Jak bylo zmíněno v kapitole o párové strategii 1.3, cílem implementace je scoring kandidátních párů a testování kointegrace. Vzhledem k potřebě několika matematických, či statistických nástrojů, které neobsahuje standardní Java, bylo rozhodnuto o použití externí knihovny. Tento krok nejenom usnadňuje práci, ale zároveň výrazně zmenšuje prostor pro programátorské chyby, obzvláště v případě sofistikovanějších nástrojů, jako například testů kointegrace. S ohledem nejenom na licenci, kde byl cílem otevřený software, ale také na podporu a zájem komunity, který značí určitou kvalitu, byla vybrána knihovna SuanShu [63].

Jedná se o vysoce výkonnou matematickou knihovnu poskytující nástroje a algoritmy pro lineární algebru, optimalizace, analýzu časových řad, zpracování digitálního signálu a mnoho dalšího. Spolu s aplikací AlgoQuant [64], nástrojem pro tvorbu automatických tradovacích programů, jde o produkt čínské společnosti Numerical Method Inc. [65], jejíž tým je složen z profesorů

a doktorů aplikované matematiky, statistiky a počítačového inženýrství. Mezi zákazníky patří finanční domy, nadnárodní společnosti, či akademické instituce. Společnost se vyznačuje individuálním přístupem k zákazníkům a možností optimalizace knihoven na míru. Obě knihovny jsou logicky placené, avšak počátkem roku 2018 byla uvolněna starší verze knihovny SuanShu, konkrétně z 6.6.2012, jako open source pod licencí Apache 2.0 [66]. Tento nástroj obsahuje téměř vše potřebné pro potřeby diplomové práce.

Scoring

Po načtení a zpracování denních dat o akciích jsou vytvořeny kandidátní páry. Těch může být velké množství a je potřeba ohodnotit jejich kvalitu a vybrat ty, které se jeví jako nejlepší. Scoring je zároveň naplněním funkčního požadavku F3 z kapitoly 2.1.1. Použit byl příklad z kapitoly 1.3.1, a sice výpočet hodnoty korelace spolu s výstupem backtestu a případného úspěchu v testu kointegrace. K implementaci byla použita třída *Covariance.java* z knihovny SuanShu, která poskytuje i hodnotu korelace.

Kointegrace

Provedení testu kointegrace je druhým krokem při vytváření strategie párového obchodování, jak zmiňuje kapitola 1.3.1. V této kapitole bylo vycházeno ze zdrojů [30], [32], [33]. Před samotným testem je potřeba otestovat obě vstupní časové řady, zda splňují podmínku nestacionarity. K tomu slouží dříve zmíněný test jednotkového kořene ADF doplněný testem stacionarity KPSS.

ADF test je založený na testování nulové hypotézy, že časová řada má jednotkový kořen, proti alternativní hypotéze, že řada je stacionární. Pro testy jednotkových kořenů je potřeba specifikovat trend vstupní řady. Ceny akcií nevykazují rostoucí ani klesající trend a proto se používá alternativní hypotéza o stacionaritě bez trendu. ADF test je k dispozici v knihovně SuanShu, kdy na vstupu očekává časové řady, parametr udávající přítomnost trendu v řadě a také parametr řád zpoždění. Jeho hodnota je poměrně důležitá, protože příliš nízká může vést k chybným výsledkům (neodstraněná korelace), naopak příliš vysoká hodnota vede ke snížení síly testu. Použita byla hodnota závislá na velikosti časové řady a sice doporučení, nazývané také jako pravidlo palce (rule of thumb):

$$p_{max} = \left[12 \cdot \left(\frac{T}{100} \right)^{\frac{1}{4}} \right]$$

KPSS je test stacionarity, který testuje nulovou hypotézu, že řada je stacionární, oproti alternativní hypotéze, že řada má jednotkový kořen (je nestacionární) [67]. Test KPSS však není k dispozici v knihovně SuanShu a byl proto implementován podle zdrojových kódů open source statistického modulu Statsmodels pro jazyk Python [68]. KPSS test byl implementován ve variantě, kdy je nulovou hypotézou stacionarita okolo konstanty, tedy pro časové řady

bez trendu. Stejně jako ADF test je vyžadován parametr řád zpoždění, pro který byla použita stejná hodnota jako v případě ADF testu.

Jasné závěry dávají následující kombinace výsledků testů:

- ADF: nezamítnutí nulové hypotézy, KPSS: zamítnutí nulové hypotézy. Oba testy značí, že časová řada má jednotkový kořen, podmínka nestacionarity je splněna. Lze tedy pokračovat testem kointegrace.

- ADF: zamítnutí nulové hypotézy, KPSS: nezamítnutí nulové hypotézy. Oba testy značí, že časová řada je stacionární.

Rozdělení testových statistik testů není normální, kritické hodnoty jsou uvedeny ve zdrojích, konkrétně v [32]. Pro vyhodnocení testů je potřeba znát hladinu významnosti α, tedy maximální chybu se kterou je zamítána nulová hypotéza. Tuto hodnotu může uživatel zadat ve vyhledávacím formuláři, případně je načtena defaultní hodnota z konfiguračního souboru *config.properties* v adresáři *WEB-INF*.

V případě rozporuplných výsledků, případně za účelem potvrzení závěrů, je možné provést další testy jako Phillips–Perron (1988), Elliot-Rothenberg-Stock (1996) nebo Ng-Perron (2001). Tyto testy však nejsou v diplomové práci implementované. Použitá verze knihovny SuanShu je nenabízí, jejich implementace by tedy byla možná stejným způsobem jako v případě KPSS testu.

Pro otestování samotné kointegrace byl zvolen Johansenův test, který je k dispozici v knihovně SuanShu. Použita byla varianta testování maximálního vlastního čísla, která testuje nulovou hypotézu, že počet kointegračních vektorů je r oproti alternativě $r + 1$. Test byl opět nastaven pro vstupní data bez trendu. Pokud je počet nalezených kointegračních vztahů vyšší než nula, jsou vstupní časové řady považovány za kointegrované.

Testování kointegrace je vzhledem k časové náročnosti prováděno automaticky pouze u dobře ohodnocených párů. Hranici tohoto skóre udává vyhledávací formulář, případně defaultní hodnoty opět z konfiguračního souboru *config.properties*. Pro ostatní páry je dostupné manuální spuštění ADF a KPSS testů, v případě úspěchu je k dispozici možnost spuštění Johansenova testu. Tlačítka spuštění spolu s výsledky jsou zobrazovány na stránce detailu páru, a sice v panelu Přehled. Úspěšný test kointegrace automaticky navýší skóre testovaného kandidátního páru.

3.3 Implementace

Sekce přiblíží vybrané části implementace a také připomene problémy, které vývoj doprovázely. Některé z nich souvisely s použitím nejnovějších verzí technologií Java EE 8.

3.3.1 Poskytovatel dat

První funkční požadavek z kapitoly 2.1 říká, že aplikace musí umět načítat aktuální denní data o akciích, konkrétně jejich close ceny (ceny na konci dne, anglicky End-of-Day, respektive EOD). Požadavek F2 zase vyžaduje možnost hledání akcií dle filtru. Dohromady s požadavkem F5, tedy provádění backtestu, budou potřeba i data historická. Nakonec požadavek N6 v podstatě neumožňuje použití jakkoliv placených dat.

Aplikace tedy potřebuje dva základní druhy dat:

- Seznam podporovaných burz a symbolů akcií na nich obchodovaných spolu s údaji o odvětví a průmyslu.
- Aktuální i historická denní data o těchto akciích. Čím více historie, tím lépe.

Ještě před zahájením implementace bylo potřeba najít zdroj těchto dat splňující všechny požadavky. Obecně platí, že pro snadné a rychlé programové získávání strukturovaných dat ze služeb třetích stran je nejvhodnější použití jejich API, pokud možno RESTful. Některé služby API neposkytují vůbec anebo s nedostačující funkcionalitou pro účely diplomové práce. Získávání aktuálních denních dat jiným způsobem než pomocí API, je nepřípustné, respektive krajní řešení. Zadání práce zmiňuje server finance.google.com jako vhodného kandidáta pro poskytování dat.

Google a jeho Finance služby dlouhá léta poskytovaly veřejné a zdarma dostupné RESTful API, které poskytovalo všechna potřebná data pro tuto práci. Analytické nástroje tento způsob získávání dat také bez výjimky podporovaly, což je určitá záruka kvality dat. Podle dostupných internetových diskuzí ho používali i běžní lidé a jejich vlastnoručně napsaný software. Bez ohledu na výhody i nevýhody API je nutné vzít v potaz, že provoz API měl být ukončen v říjnu roku 2012. Nicméně Google nechal servery dále v provozu, ale bez podpory. V případě problému většího rázu měly být servery vypnuty úplně [69]. API bylo plně funkční až do druhé poloviny roku 2017. Dle internetových diskuzí od září roku 2017 začaly API postihovat výpadky, či změny formátů odpovědí, což mohlo souviset s příchodem nového Google Finance prostředí [70], které však žádné nové API nepřináší. Aktuálně, tedy v březnu roku 2018, Google Finance API stále určitým způsobem funguje, avšak je kompletně bez dokumentace a dle diskuzí se nedá považovat za stabilní službu. Navíc stále platí hrozba náhlého ukončení celého API kdykoliv v budoucnu.

Bezplatnou alternativou k Google Finance API bylo po dlouhá léta Yahoo! Finance a jeho API pro získávání dat historických, či intra-denních. V květnu roku 2017 začalo Yahoo! měnit funkčnost těchto API. Výsledkem je, že Yahoo! Intraday API neodpovídá vůbec, stejně jako Historical API pro stahování EOD cen ve formátu CSV [71]. Podle vyjádření moderátora oficiálního Yahoo! fóra opravdu došlo k ukončení poskytování služeb [72]. Existují snahy pro

získávání dat přímo ze stránek Yahoo! Finance ve formátu CSV, viz například projekt [73], nicméně výstupy se liší od dříve poskytovaných a formát dat se v poslední době také mění. V současnosti, tedy březen 2018, nelze Yahoo! Finance považovat za vhodný a spolehlivý zdroj potřebných dat.

Jako funkční alternativy splňující všechny požadavky byly nalezeny tyto služby:

Investors Exchange

Investors Exchange (IEX) je relativně nová burza, založená v březnu roku 2012, která pomalu roste, nicméně už nyní poskytuje bezplatné API s kvalitní dokumentací. Není potřeba žádná registrace. Bohužel API je orientováno spíše na real-time data a historii cen poskytuje zatím pouze pětiletou. Jedná se o možnou variantu a do budoucna zajímavou možnost i s ohledem na propojení s IEX burzou [74].

Alpha Vantage

Alpha Vantage je poskytovatel zdarma dostupného API pro real-time a historická data (měsíční, týdenní, denní i data s až minutovým intervalem). Historie obsahuje až 20 let zpět. API poskytuje data o akciích, měnách, či kryptoměnách. Pro použití API je potřeba API klíč, jehož získání je podmíněno bezplatnou registrací vyžadující pouze emailovou adresu. Autoři slibují doživotní přístup [75]. API se volá pomocí HTTP GET požadavků a URL parametrů, jak lze vidět na vzorovém dotazu 3.1, který se dotazuje na symbol MSFT. Dotaz musí obsahovat funkci (*TIME_SERIES_DAILY* je pro denní data), symbol a API klíč. Odpověď lze získat ve formátu JSON i CSV, defaultní je JSON. Nepovinný, ale pro diplomovou práci důležitý, je parametr *outputsize*, jehož hodnota *full* zaručí odpověď s celou dostupnou historií pro dotazovaný symbol.

Alpha Vantage API má však i řadu nevýhod, a sice:

- API neposkytuje seznam podporovaných burz a na nich dostupných symbolů ani dalších metadat. Dokumentace se o těchto věcech nikde nezmiňuje. V souladu s funkčním požadavkem F2 jsou pro navrženou aplikaci taková data potřeba. Dle internetových diskuzí [76], jsou plně podporovány US burzy, pro ostatní se musí provést dotaz se symbolem v jiném formátu. Vzhledem k snadné dostupnosti potřebných seznamů v CSV formátu přímo ze stránek NASDAQ [77], tedy seznamu akcií, odvětví a průmyslů na vybraných US burzách, se práce omezuje na největší newyorské burzy, a sice NASDAQ, NYSE, AMEX. Nefunkční požadavek N1 požaduje snadnou rozšiřitelnost, takže se jedná v podstatě o dočasné omezení.

Kód 3.1: Příklad volání Alpha Vantage API

```
https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&
    symbol=MSFT&apikey=demo

// response
{
"Meta Data": {
"1. Information": "Daily Prices (open, high, low, close) and
    Volumes",
"2. Symbol": "MSFT",
"3. Last Refreshed": "2018-03-26",
"4. Output Size": "Compact",
"5. Time Zone": "US/Eastern"
},
"Time Series (Daily)": {
"2018-03-26": {
"1. open": "90.6100",
"2. high": "94.0000",
"3. low": "90.4000",
"4. close": "93.7800",
"5. volume": "56260185"
},
...
}
```

- Nelze se dotázat na více symbolů v jednom dotazu. Je tedy nutné se dotazovat na každý symbol zvlášť.

- S předchozím bodem souvisí limit pro dotazování se API, který sice není v dokumentaci uveden, ale doporučením je jedno volání za jednu sekundu. Při testování API se tento limit opravdu začne po několika rychlejších voláních uplatňovat a API začne na nespecifikovanou dobu vracet chybové zprávy namísto dat.

- Kvalita dat. Dokumentace nezmiňuje, odkud Alpha Vantage data bere, tato informace není k nalezení ani nikde jinde. Dle reakcí v internetových diskuzích data odpovídají [78]. Avšak v průběhu tvorby a testování aplikace bylo odhaleno pár nedostatků. Například data o akciích firmy Apple Inc. (symbol AAPL). Z porovnání dat, která vrací Alpha Vantage API (ukázka viz kód 3.2) a dostupných dat přímo na webu Nasdaq.com (ukázka na obrázku 3.1), je patrné, že historická data se shodují pouze do 6. června 2014. To představuje poměrně velký problém, zejména pro backtesting strategie. Pro účely diplomové práce ojedinělé chyby nevadí, nicméně jde o důkaz, že ani na tento zdroj dat se nelze úplně spolehnout.

Date	Open	High	Low	Close / Last	Volume
06/10/2014	94.73	95.05	93.57	94.25	62,563,070
06/09/2014	92.7	93.88	91.75	93.7	75,295,450
06/06/2014	92.8428	93.0371	92.0671	92.2243	87,403,076
06/05/2014	92.3143	92.7671	91.8014	92.4785	75,796,583
06/04/2014	91.0628	92.5557	90.8728	92.1171	83,735,285

Obrázek 3.1: AAPL data z Nasdaq.com, kolem 6.6.2014

Shrnutí

Jak již bylo řečeno dříve, aplikace potřebuje dva druhy dat. Po průzkumu dostupných řešení vyhovujících stanoveným požadavkům, bylo rozhodnuto o použití zdroje Alpha Vantage API. Tomu však předchází načtení dat o podporovaných burzách, jejich sektorech, odvětvích a obchodovatelných symbolech po startu aplikace. Tyto informace jsou uchovávány ve formě CSV souborů a v aplikaci konfigurovány pomocí souboru *config.properties* v adresáři *WEB-INF*. Pro snazší mapování CSV souborů na objekty byla použita knihovna uniVocity [79], ukázka použití viz kód 3.4. U třídní proměnné *lastSale* si lze všimnout širokých možností knihovny a sice použití vlastní konverze z dat ve formátu *$40.000* na objekt *BigDecimal*. Vzhledem k nefunkčnímu požadavku N1 musí jít pouze o vzorovou, snadno nahraditelnou implementaci. Za tímto účelem byla vytvořena abstraktní třída *StockListElement.java*, která reprezentuje načtený akciový symbol spolu se souvisejícími daty. Prostřednictvím abstraktních metod třída vynucuje povinné atributy, jako symbol, sektor a průmysl konkrétního záznamu (na nich jsou také založeny metody *hashCode()* a *equals()*). Konkrétní implementace jsou potom realizovány jako potomci této třídy a následně přetypovány na rodiče. Další atributy specifické pro konkrétní implementaci jsou zachovány ve formě mapy metadat a uživatel je tak bude mít vždy k dispozici. Načítání CSV souborů pomocí knihovny je příklad takové implementace. Kód abstraktní třídy zobrazuje kód 3.3 a potomka viz 3.4.

Získávání denních dat o akciích z Alpha Vantage API je také pouze implementací rozhraní *StockLoader.java* v souladu s požadavkem N1 o rozšiřitelnosti. S ohledem na limit API jednoho požadavku za sekundu, vztahujícího se na celou aplikaci, tedy všechny session, je implementace rozhraní anotována *@ApplicationScoped* a metoda s logikou dotazování a čekání označena jako *synchronized*. Pro samotnou komunikaci s API byla použita zdarma dostupná knihovna alphavantage4j [80]. Její funkcionalitu používá datová vrstva a data předává vyšším vrstvám v podobě instancí třídy *DayPrice.java*.

Kód 3.2: AAPL data z Alpha Vantage API, kolem 6.6.2014

```
https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&
symbol=AAPL&outputsize=full&apikey=demo
...
"2014-06-10": {
"1. open": "94.7300",
"2. high": "95.0500",
"3. low": "93.5700",
"4. close": "94.2500",
"5. volume": "62777000"
},
"2014-06-09": {
"1. open": "92.7000",
"2. high": "93.8800",
"3. low": "91.7500",
"4. close": "93.7000",
"5. volume": "75414997"
},
"2014-06-06": {
"1. open": "649.9000",
"2. high": "651.2600",
"3. low": "644.4700",
"4. close": "645.5700",
"5. volume": "12497800"
},
"2014-06-05": {
"1. open": "646.2000",
"2. high": "649.3699",
"3. low": "642.6100",
"4. close": "647.3500",
"5. volume": "10850200"
},
"2014-06-04": {
"1. open": "637.4400",
"2. high": "647.8900",
"3. low": "636.1100",
"4. close": "644.8200",
"5. volume": "11981500"
},
...
```

Kód 3.3: Abstraktní třída (*StockListElement.java*)

```
public abstract class StockListElement implements Serializable {

private static final long serialVersionUID = 1067215454631309929L;

public abstract StockCode getStockCode();

public abstract String getSector();

...

@Override
public boolean equals(Object obj) {
// based on StockCode
}

@Override
public int hashCode() {
// based on StockCode
}
...
}
```

3.3.2 Uživatelské rozhraní

Při realizaci uživatelského rozhraní na základě požadavků a wireframes z kapitoly 2.4 byl kladen důraz na jednoduchost a využití standardních dostupných řešení, to znamená vyhnutí se vlastním implementacím, které by přinesly možnou budoucí nekompatibilitu a chyby. Vzhledem ke zvoleným technologiím v kapitole 3.1, tedy JSF 2.3 a knihovnám PrimeFaces, PrimeFaces Extensions a OmniFaces se přímo nabízí použití předem připravených a customizovatelných komponent. Nejzajímavější z nich budou dále připomenuty.

JSF 2.3

Spolu s příchodem JSF verze 2.2 se změnily doporučené namespace pro standardní knihovny v JSF view souborech. Tato změna je poměrně důležitá, protože nová funkcionalita je dostupná pouze při použití nových namespace, jako například tag *<f:viewAction .../>*, který před JSF 2.2 neexistoval. V tomto konkrétním případě jde o substituci *xmlns:f="http://java.sun.com/jsf/core"* za *xmlns:f="http://xmlns.jcp.org/jsf/core"*.

Poslední verzi 2.3 doprovází další konfigurační změna. Možnost specifikovat verzi JSF pomocí souboru *faces-config.xml*, jako například *<faces-config version="2.3"...>*, již není funkční tak, jako tomu bylo ve verzích předchozích. Nově je třeba specifikovat verzi anotací v Java třídě jak zobrazuje kód 3.5.

Kód 3.4: Implementace načítání z CSV (*CsvStockListElement.java*)

```
public class CsvStockListElement extends StockListElement
    implements Serializable {

private static final long serialVersionUID = 73247043861326657L;

@Parsed(field = "Symbol")
@Convert(conversionClass = SymbolToStockCodeConversion.class)
private StockCode stockCode;

@Parsed(field = "Sector", defaultNullRead = StringUtils.EMPTY)
private String sector;

@Parsed(field = "LastSale", defaultNullRead = "0")
@NullString(nulls = {"n/a"})
@Convert(conversionClass
= DollarToBigDecimalStringConversion.class)
private BigDecimal lastSale;

...

@Override
public StockCode getStockCode() {
if (stockCode != null && stockCode.getMetadata().isEmpty()) {
// collect other implementation specific data
createMetadataMap();
}
return stockCode;
}

@Override
public String getSector() {
return sector;
}
...
}
```

Kód 3.5: Konfigurace JSF 2.3 (*ConfigurationBean.java*)

```
import javax.faces.annotation.FacesConfig;
import javax.enterprise.context.SessionScoped;
import static javax.faces.annotation.FacesConfig.Version.JSF_2_3;

@FacesConfig(version = JSF_2_3)
@SessionScoped
public class ConfigurationBean {
...
}
```

Kód 3.6: Konfigurace lokalizace (*faces-config.xml*)

```
<application>
  <locale-config>
    <default-locale>en</default-locale>
    <supported-locale>cs</supported-locale>
  </locale-config>

  <resource-bundle>
    <base-name>cz.malecek.stockpairfinder.pairDetail</base-name>
    <var>i18n_pairDetail</var>
  </resource-bundle>
</application>
```

Specifikace JSF 2.3 bere na vědomí pouze tento způsob, což se přirozeně týká všech implementací, tedy i JSF 2.3 kompatibilního serveru GlassFish verze 5, který používá implementaci Mojarra [81]. JSF 2.3 navíc defaultně běží v jakémsi pseudo JSF 2.2 kompatibilním režimu a tak je nutné, v případě potřeby, novou verzi explicitně zapnout [82] [83].

Podpora jazyků

Funkční požadavek F8 určuje podporu českého a anglického jazyka. Pro splnění požadavku je potřeba nejprve nakonfigurovat podporované lokalizace v souboru *faces-config.xml* jak zobrazuje kód 3.6. Součástí je ukázka konfigurace *resource-bundle*, který uchovává překlady textů, konkrétně v souborech *cz.malecek.stockpairfinder.pairDetail_en.properties* a *pairDetail_cs.properties*. Použití v *xhtml* kódu lze vidět například v kódu 3.18. Pro účely lokalizace knihovny PrimeFaces, jako například validačních zpráv, či konfigurace kalendáře, se používají další soubory, konkrétně *PrimeFaces_en.js*, respektive *PrimeFaces_cs.js* pro české texty. Pro použití stačí skripty přidat do kódu stejným způsobem jako jiné skripty. Ukázku obsahu souborů zobrazuje kód 3.7.

Kód 3.7: Zkrácená ukázka lokalizace PrimeFaces (*Primefaces_en.js*)

```
PrimeFaces.locales ['en'] = {
  closeText: 'Close',
  dayNames: ['Sunday', 'Monday', 'Tuesday', 'Wednesday',
    'Thursday', 'Friday', 'Saturday'],
  dayNamesShort: ['Sun', 'Mon', 'Tue', 'Wed', 'Tue', 'Fri', 'Sat'],
  dayNamesMin: ['S', 'M', 'T', 'W ', 'T', 'F ', 'S'],
  day: 'Day',
  allDayText: 'All Day',
  messages: {
    // optional for Client Side Validation
    'javax.faces.component.UIInput.REQUIRED':
      '{0}: Validation Error: Value is required.',
    'javax.faces.converter.NumberConverter.CURRENCY':
      '{2}: \'{0}\' could not be understood as a currency value.',
    'javax.faces.converter.NumberConverter.NUMBER':
      '{2}: \'{0}\' could not be understood as a date.',
    'javax.faces.converter.NumberConverter.NUMBER_detail':
      '{2}: \'{0}\' is not a number. Example: {1}',
    // optional for JSR-303 Bean Validation integration
    'javax.validation.constraints.Min.message':
      'Fill in value greater than or equal to {0}.',
    'javax.validation.constraints.NotNull.message':
      'Fill in the field.',
    'javax.validation.constraints.Past.message':
      'Fill in date in the past.',
    'javax.validation.constraints.Pattern.message':
      'Fill in value matching "{0}".',
  }
};
```

Nastavení lokalizace je třeba uchovávat na úrovni session. K tomu slouží třída *LocaleBean.java*, která si pamatuje nastavenou lokalizaci a v případě potřeby ji mění pro konkrétní session, ukázka viz kód 3.8.

Důležité je nastavení lokalizace i v JSF view jak ukazuje kód 3.9, které slouží pro budoucí požadavky. Lze si všimnou tagu *<html lang>*, který není nezbytný, avšak deklarace jazyka stránky pomáhá vyhledávačům a prohlížečům [84].

Validace

Validace je třetí fází v životním cyklu zpracovávání JSF požadavku jak ukazuje obrázek 3.2. V případě neúspěšné validace se přistoupí rovnou k odeslání odpovědi klientovi. Výchozí chybové zprávy lze konfigurovat a překládat pomocí souborů *PrimeFaces_en.js* a *PrimeFaces_cs.js* viz kód 3.7. JSF aplikace podporují více druhů validací:

Kód 3.8: Ukázka (*LocaleBean.java*)

```
@Named
@SessionScoped
public class LocaleBean implements Serializable {

  private static final long serialVersionUID = -3979336059645538991L;

  private static final Logger LOGGER
    = Logger.getLogger(LocaleBean.class);

  @Inject
  private FacesContext facesContext;

  /**
  * Current {@link Locale}
  */
  private Locale locale;

  /**
  * Set the default locale from faces-config.xml file.
  */
  @PostConstruct
  public void init() {
    locale = facesContext.getCurrentInstance().getApplication().
        getDefaultLocale();
    LOGGER.log(Level.DEBUG, "Setting locale: " + locale);
  }

  /**
  * @return {@link LocaleBean#locale}
  */
  public Locale getLocale() {
    return locale;
  }

  /**
  * @return Language from the {@link LocaleBean#locale}
  */
  public String getLanguage() {
    return locale.getLanguage();
  }

  public void setLanguage(String language) {
    locale = new Locale(language);
    if (facesContext.getViewRoot() != null) {
      facesContext.getViewRoot().setLocale(locale);
      LOGGER.log(Level.DEBUG, "Locale changed to: " + locale);
    }
  }
}
```

Kód 3.9: Lokalizace ve view (*main-template.xhtml*)

```
<!DOCTYPE html>
  <html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:h="http://xmlns.jcp.org/jsf/html"
  xmlns:p="http://primefaces.org/ui"
  xmlns:f="http://xmlns.jcp.org/jsf/core"
  xmlns:pe="http://primefaces.org/ui/extensions"
  xmlns:o="http://omnifaces.org/ui"
  lang="#{localeBean.language}">

  <f:view encoding="UTF-8" contentType="text/html"
    locale="#{localeBean.locale}">
    ...
  </f:view>
</html>
```

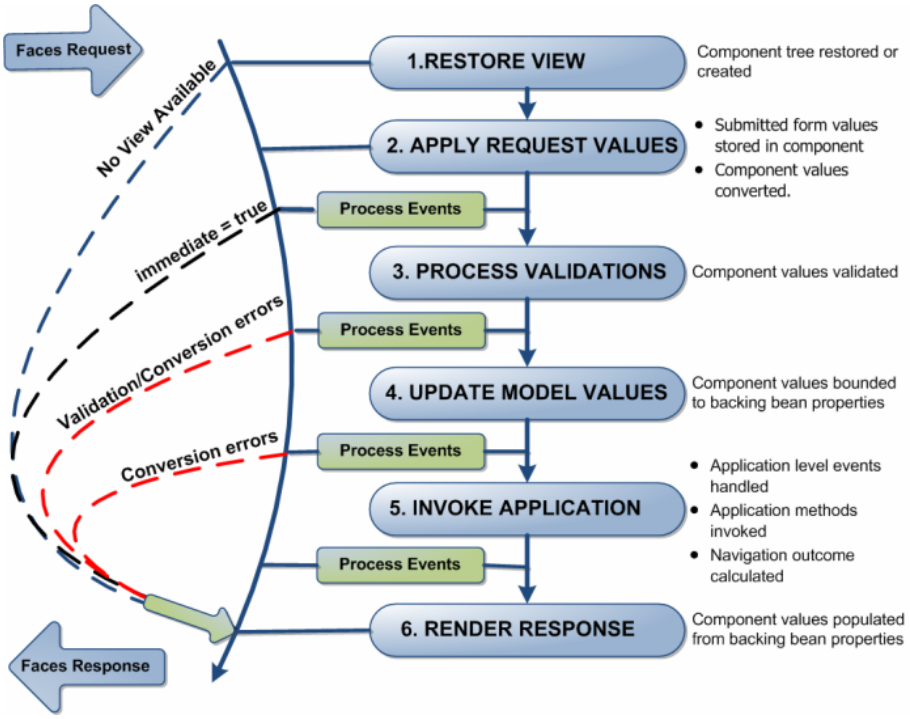
- JSF standardní validátor
- Vlastní JSF validátor
- Volání validační metody
- JSR-303 Bean Validation

JSF má implementováno několik standardních validátorů jako například *RequiredValidator* pro kontrolu povinnosti, jenž je často používán jako atribut *required="true"* políčka, anebo *DoubleRangeValidator* pro kontrolu čísla, zda patří do intervalu. Ukázka použití ve view kódu 3.10.

Pokud nedostačují funkce standardních validátorů, lze vytvořit validátory vlastní, konkrétně implementací rozhraní *javax.faces.validator.Validator*. Nová třída musí mít anotaci *@FacesValidator* a musí implementovat metodu rozhraní *validate(FacesContext, UIComponent, Object value) throws ValidatorException*, která se automaticky volá při zpracovávání požadavku. V případě nevyhovující hodnoty validované komponenty je třeba vyhodit právě *ValidatorException*, která přijímá argument v podobě chybové zprávy pro uživatele. Příklad použití ve view kódu 3.10.

Validační metoda je pouze jiná forma vlastního validátoru, protože metoda musí mít stejnou hlavičku, tedy *myMethod(FacesContext, UIComponent, Object value) throws ValidatorException*, avšak liší se použitím, metoda se nastavuje pomocí atributu *validator="#{myBean.myMethod}"* konkrétní validované komponenty [86].

Poslední způsob validace je použití JSR-303 [87] validací pomocí anotací na úrovni modelových tříd. GlassFish verze 5 již obsahuje implementaci této specifikace [88], případně lze explicitně použít Hibernate Validator implementaci. Na rozdíl od předchozích způsobů se JSR-303 validování nedefinuje v JSF view,



Obrázek 3.2: JSF Lifecycle [85]

Kód 3.10: Použití standardního a vlastního validátoru

```
// usage of the standard validator inside input component
<f:validateDoubleRange minimum="3.0" maximum="4.5"/>

// usage of the custom validator with
// @FacesValidator("emailValidator") inside input component
<f:validator validatorId="emailValidator"/>
```

ale v Java třídách, viz kód 3.11 s ukázkou anotací. Pro správnou funkčnost anotace *@NotNull* je nutná konfigurace v souboru *web.xml* viz kód 3.12 [89]. Tento druh validací je použit v diplomové práci.

Layout

Vzhledem ke kapitole o návrhu uživatelského rozhraní 2.4 má prostředí připomínat klasické programy, jako například emailového klienta. Za tímto účelem byla použita komponenta *Layout* z PrimeFaces Extensions, která umožňuje rozdělení stránky na panely přesně dle potřeby (horní, levý, dolní, pravý a střední). Jednotlivé panely lze skrývat, upravovat jejich velikost anebo vnořovat. Komponenta umožňuje zachycení a reagování na některé události, jako

Kód 3.11: Ukázka použití JSR-303 validací (*StockFilter.java*)

```
@NotNull
private String exchangeName;
@NotNull
private String sector;
private String industry;
...
```

Kód 3.12: Konfigurace pro JSR-303 *@NotNull* anotaci (*web.xml*)

```
<context-param>
  <param-name>
    javax.faces.INTERPRET_EMPTY_STRING_SUBMITTED_VALUES_AS_NULL
  </param-name>
  <param-value>true</param-value>
</context-param>
```

skrytí panelu (nelze bohužel specifikovat pro konkrétní panel, nýbrž pouze globálně, avšak tuto informaci lze zpětně zjistit pomocí PrimeFaces Client Side API). Kombinací těchto možností lze dosáhnout požadovaného rozložení a chování. *Layout* je široce konfigurovatelný, v případě implementované aplikace je konfigurace možná pomocí souboru *config.properties* v adresáři *WEB-INF*. Možnosti komponenty jsou nejlépe demonstrovány přímo na oficiální stránce viz [90].

Dialog Framework

Pro realizaci dialogu s rozšířenými možnostmi hledání jsou v knihovně PrimeFaces dostupná dvě řešení. Klasická komponenta *<p:dialog>* vytvoří div element, který je vygenerován ihned spolu s celou stránkou, a v závislosti na potřebě se mění pouze jeho viditelnost. Druhou možností je použití Dialog Framework, který umožňuje generování dialogu dynamicky za běhu. Takový dialog v podstatě zobrazuje externí stránku, která s původní rodičovskou nemá nic společného. Toto řešení je mnohem flexibilnější, avšak o něco složitější. Protože Dialog Framework je novější (byl přidán do PrimeFaces v roce 2013) a zajímavější, bylo při implementaci použito právě toto řešení [91]. Nezbytnou konfiguraci pro Dialog Framework zachycuje kód 3.13 [92].

Tlačítko pro otevření dialogu ukazuje kód 3.14. Jeho stisknutí zavolá metodu 3.15. Ta vytvoří vhodnou request mapu a dialog otevře.

Vzhledem k tomu, že Dialog Framework pracuje se zvláštní stránkou, je vhodné k ní zamezit přímý přístup. Kontrolu přístupu lze provést programově, konkrétně použitím tagu *<f:viewAction action="#{bean.onLoad}"/>*, který je součástí od JSF 2.2 (konfigurace viz 3.3.2) [93]. Jeho výhodou oproti staršímu řešení v podobě *<f:event type="preRenderView"listener="#{bean.onLoad}"/>* je

Kód 3.13: Konfigurace pro Dialog Framework (*faces-config.xml*)

```
<action-listener>
  org.primefaces.application.DialogActionListener
</action-listener>

<navigation-handler>
  org.primefaces.application.DialogNavigationHandler
</navigation-handler>

<view-handler>
  org.primefaces.application.DialogViewHandler
</view-handler>
```

Kód 3.14: Tlačítko pro otevření dialogu (*leftmenu-template.xhtml*)

```
...
<p:commandButton id="moreSettingsButton"
  value="#{i18n_leftMenu['moreSettings']}"
  action="#{searchPairsBean.openSearchSettingsDialog('leftMenu')}">

  <!--
  dialogReturn event: data could be passed,
  see page 587 in PF 6.1 manual.
  -->
  <p:ajax event="dialogReturn"
    listener="#{searchPairsBean.loadPairsOnDialogReturn}"/>
</p:commandButton>
...
```

Kód 3.15: Otevření dialogu (*SearchPairsBean.java*)

```
/**
* Open search settings dialog.
*
* @param origin name of the page which wants to open dialog
*/
public void openSearchSettingsDialog(String origin) {
  final Map<String, List<String>> requestParameterValuesMap
    = new HashMap<>();
  requestParameterValuesMap.put(
    SEARCH_DIALOG_REQUEST_MAP_ORIGIN_KEY,
  Collections.singletonList(origin));
  RequestContext.getCurrentInstance().openDialog(
    SEARCH_DIALOG_NAME, searchSettingsDialogOptions,
    requestParameterValuesMap);
}
```

Kód 3.16: Přístup na dialog, JSF view (*searchSettingsDialog.xhtml*)

```
...
<f:metadata>
  <!-- Check if dialog is beeing opened from the index page. -->
  <f:viewAction
    action="#{searchPairsBean.checkSearchSettingsDialogAccess}"
    onPostback="false"/>
</f:metadata>

<f:view encoding="UTF-8" contentType="text/html" ...>
...
```

Kód 3.17: Přístup na dialog, Java metoda (*SearchPairsBean.java*)

```
public void checkSearchSettingsDialogAccess() {
  final Map<String, String[]> requestParameterValuesMap =
      externalContext.getRequestParameterValuesMap();
  final String[] hostPageParamValue = requestParameterValuesMap.
      get(SEARCH_DIALOG_REQUEST_MAP_ORIGIN_KEY);

  if (hostPageParamValue == null || hostPageParamValue.length == 0
    || StringUtils.isEmpty(String.valueOf(hostPageParamValue[0]))){
    try {
      LOGGER.log(Level.DEBUG,
        "Dialog access hasn't been done from any allowed page,
        HttpServletResponse.SC_NOT_FOUND will be send.");
      externalContext.responseSendError(
      HttpServletResponse.SC_NOT_FOUND, "Not Found");
    } catch (IOException e) {
      ...
    }
  } else {
    LOGGER.log(Level.DEBUG, "Dialog access has been done from an
        allowed page ('hostPage': " + hostPageParamValue[0] + ")");
  }
}
```

parametr *onPostback="false"*, který zajišťuje, že volání neprobíhá při postback request a není nutné toto ošetřovat v kódu volané metody. Ukázka kódu 3.16 zobrazuje volání metody pro kontrolu přístupu, jejíž kód 3.17 ověří, zda byl do request mapy vložen nějaký řetězec, jenž by identifikoval původ žádosti o otevření dialogu. V případě úspěšného ověření zdroje je dialog zobrazen. Pokud zdroj není identifikován, například pokud měl být dialog otevřen přímým přístupem na stránku dialogu, žádná hodnota nebude k dispozici a bude odeslán response s HTTP statusem 404 Not Found.

Dialog je tedy samostatná xhtml stránka a v případě této práce se její

Kód 3.18: Tlačítka na stránce dialogu (*searchSettingsDialog.xhtml*)

```
<p:commandButton id="closeButtonDialog"
  value="#{i18n_common['close']}"
  action="#{searchPairsBean.closeSearchSettingsDialog('false')}"/>

<p:commandButton id="searchButtonDialog" ajax="false"
  value="#{i18n_leftMenu['search']}" validateClient="true"
  action="#{searchPairsBean.closeSearchSettingsDialog('true')}"/>
```

obsah bude podobat vyhledávacímu formuláři, akorát bude obsahovat více nepovinných vyhledávacích možností. Jak bylo odhaleno v Nielsenově heuristice viz 3.4, kromě tlačítka pro spuštění hledání bude dialog obsahovat i tlačítko pro zavření dialogu. Tlačítko pro spuštění hledání je v podstatě shodné s tlačítkem pro spuštění hledání, které je součástí vyhledávacího formuláře na domovské stránce v levém panelů. Po stisknutí se musí provést validace povinných polí a dalších omezení tak, jak je definováno pro vyhledávací filtr. Pokud nenastanou žádné chyby, je provedeno vyhledávání dle filtru z formuláře. Naproti tomu, u tlačítka pro zavření dialogu chceme zachovat vyplněné hodnoty, i kdyby nesplňovaly validační pravidla, a bez provedení validací se vrátit na domovskou stránku. Ve výchozím stavu se validace provádí při každém odeslání formuláře viz 3.2. K dosažení tohoto rozdílného požadavku na validace bylo použito tagu *<f:validateBean disabled="true"/>*, který obaluje vyhledávací formulář a vypíná pro něj výchozí provádění validací. Tím se zaručí bezproblémový návrat po stisknutí tlačítka k zavření dialogu a v případě tlačítek pro vyhledávání je potřeba validaci explicitně zapnout pomocí atributu *<p:commandButton validateClient="true".../>*. Kód tlačítek je zobrazen v kódu 3.18

Rozlišit chování těchto dvou tlačítek je ale potřeba nejenom na úrovni validace, ale v další fázi zpracovávání, jakou je návrat z dialogu. Univerzální metodu, kterou volají obě tlačítka, avšak s odlišným parametrem (viz kód 3.18), zobrazuje kód 3.19. Metoda uzavře dialog a předá vstupní parametr dále metodě *RequestContext.closeDialog()*. Uzavření dialogu vytvoří *dialogReturn* událost, kterou lze zachytit a reagovat na ní, stejně jako na kteroukoliv jinou událost v rámci JSF Ajax API (volání JavaScriptu, backing bean metod, překreslení oblastí a podobně). V diplomové práci byla tato událost také využita, jak zobrazuje kód 3.14, a po úspěšném návratu z dialogu se na základě hodnoty parametru předaného voláním *RequestContext.closeDialog()* spustí hledání kandidátních párů. Metodu volanou po zachycení události zobrazuje kód 3.20.

3.3.3 Grafy

Práce s knihovnou ChartistJSF je velice jednoduchá, při tvorbě grafů v backing bean jsou grafům přiřazena data, popisky, osy a další nastavení.

Kód 3.19: Zavření dialogu (*SearchPairsBean.java*)

```
/**
* Close search dialog.
*
* @param load flag says if
* {@link SearchPairsBean#loadPairs()} call is needed.
*/
public void closeSearchSettingsDialog(Boolean load) {
  RequestContext.getCurrentInstance().closeDialog(load);
}
```

Kód 3.20: Reakce na *dialogReturn* událost (*SearchPairsBean.java*)

```
/**
* After dialog is closed, check passed param and conditionally
* call {@link SearchPairsBean#loadPairs()}.
*
* @param event from JSF
*/
public void loadPairsOnDialogReturn(SelectEvent event) {
  if ((Boolean) event.getObject()) {
    loadPairs();
  }
}
```

Kód 3.21: Použití chartist-plugin-legend

```
var legendPlugin = [
  Chartist.plugins.legend()
];
```

Zajímavější je přidávání pluginů, které není tak jednoduché jako v případě původní JavaScript knihovny Chartist.js. Typicky je potřeba manuální zásah a úprava zdrojových kódů pluginů (skriptů a stylů) pro potřeby JSF aplikace. Zdrojové kódy pluginů jsou snadno dostupné na GitHub, jako například použitý plugin chartist-plugin-legend [94]. Aktivace pluginů už je snadná, slouží k tomu atribut *plugins="legendPlugin"* JSF tagu *<ct:chart>* knihovny ChartistJSF a odpovídající JavaScript proměnná. V JS je možné specifikovat i vlastní nastavení pluginů, nicméně ukázka kódu 3.21 si vystačí s výchozím nastavením pluginu.

3.3.4 Backtesting

Podpora backtestingu, neboli testování strategie na historických datech bez rizika ztráty skutečného kapitálu, je požadováno funkčním požadavkem F5.

Popis strategie, respektive nezbytné vstupy pro backtesting, jsou definovány ve vyhledávacím formuláři. Na stejném místě lze také povolit, či zakázat automatické provádění backtestingu. V případě zapnutí je backtesting automaticky prováděn pro všechny páry. Pro importované páry lze backtest spustit ručně (po stažení potřebných dat) pomocí tlačítka na stránce detailu vybraného páru v panelu Přehled. Ve stejném panelu jsou přehledně zobrazeny výstupy backtestu a jeho konfigurace (počáteční stav konta, margin, strategie, či stop loss). Skóre páru je upraveno podle výsledného profitu, či ztráty. K dispozici jsou i další výstupní údaje o délce, počtu výdělečných a prodělečných obchodů. Dále také maximální série zisků a ztrát anebo maximální a průměrný zisk, či ztráta a další atributy.

Shrnutí

Uživatelské rozhraní aplikace podporuje vyhledávání akcií dle filtru, nalezené páry jsou poté ohodnoceny viz kapitola 3.2. Pro dobře ohodnocené páry jsou automaticky spuštěny testy kointegrace, respektive jejích podmínek. V případě úspěchu je vylepšeno skóre kandidátního páru. Výsledky automaticky prováděného backtestu jsou také promítnuty do výsledného skóre. Ostatní páry, pro které nebyly automaticky provedeny testy, nejsou zahazovány, naopak jsou v seznamu výsledků a kdykoliv je možné spustit tyto procesy manuálně z obrazovky detailu páru. Manuální zásah je potřeba i v případě naimportovaných párů. Pro ně typicky chybí denní data, jejichž stahování není automatické. V průběhu testování aplikace backtesting v podstatě potvrdil výstupy scoringu. Pro dobře ohodnocené páry byly výstupy backtestingu lepší, než pro páry s nižším ohodnocením.

Stránka detailu vybraného páru obsahuje několik panelů, jako přehled údajů o páru, vypočtených statistikách a výsledcích backtestingu, dále panely s grafy vývoje cen, volume, korelace, spreadu či ratio anebo panel se získanými metadaty. Aplikace umožňuje přepínání mezi českým a anglickým jazykem, stejně jako klávesové zkratky pro pohyb v tabulkách. Tím byly splněny všechny funkční požadavky definované v kapitole 2.1.1.

3.4 Nielsenova heuristická analýza

Již po vytvoření prototypu uživatelského rozhraní je dobré provést heuristickou analýzu. Jde o soubor všeobecných pravidel získaných z dlouhodobých zkušeností uživatelů s obsluhou systémů. K dosažení intuitivního a přehledného ovládání je potřeba tato pravidla ověřit [95]. Právě k tomu slouží Nielsenova heuristická analýza [96].

1) Visibility of system status

P1 – Zablokování prostředí s informací o stavu.

- Popis: V případě, že uživatel přijímá data ze vzdáleného serveru, či načítá data z lokálního disku, není zobrazen momentální stav systému. V případě delší odpovědi zdroje uživatel nemusí vědět, v jakém stavu systém je.

- Řešení: U těchto akcí bude během jejich vykonávání zobrazena informační hláška o stavu systému a bude znemožněno vyvolat další akci (např. opětovné odeslání formuláře).

- Priorita: Vysoká.

2) Match between system and the real Word

Pořadí jednotlivých úkonů je logické a odpovídá zažitým konvencím. Ikony také odpovídají reálnému světu viz vlajky pro změnu jazyka.

3) User control and freedom

P2 – Návrat z vyhledávacího dialogu.

- Popis: V případě, že uživatel otevře vyhledávací dialog s rozšířenými možnostmi, nemá jinou možnost, než spustit vyhledávání kandidátních párů.

- Řešení: Umožnit návrat z vyhledávacího dialogu s rozšířenými možnostmi bez nutnosti spustit vyhledávání (tlačítko zpět).

- Priorita: Vysoká.

4) Consistency and standards

V tomto ohledu uživatelské rozhraní zachovává běžné konvence. Rozmístění a tvar prvků odpovídá běžně používaným standardům.

5) Error prevention

Uživateli není u vstupních políček povoleno zadat nevalidní data (například do *Volume* políčka lze zadat pouze numerické znaky). V případě, že pravidla pro konkrétní políčko jsou složitější, je uživateli vysvětleno, v jakém formátu musí být vstup. Chybové hlášky respektují aktuálně nastavený jazyk prostředí. Všechny povinné položky formulářů jsou označeny běžně používaným červeným znakem *.

6) Recognition rather than recall

Aplikace sice neobsahuje drobečkovou navigaci, ta ale není potřeba vzhledem k tomu, že veškerá činnost probíhá na jedné stránce a všechny platné volby jsou stále viditelné, naopak ostatní skryté, či vizuálně zneplatněné (pro výběr *Industry* je potřeba nejdříve vybrat *Exchange*).

7) Flexibility and efficiency of use

Prostředí aplikace je v podstatě jedna stránka a snaží se suplovat běžně používané programy jako je emailový klient. Pro pokročilejší uživatele jsou k dispozici klávesové zkratky pro rychlejší navigaci v tabulce výsledků a oblíbených párů.

8) Aesthetic and minimalist design

Viz předchozí dva body. Prostředí aplikace obsahuje minimum informací a to ty, které uživatel aktuálně potřebuje.

9) Help users recognize, diagnose, and recover from errors

Veškeré hlášky ve formulářích jsou popsány srozumitelně v běžném jazyce a popisují konkrétní důvod vzniklé chyby.

10) Help and documentation

Jelikož je systém určen po běžné uživatele, mělo by ho být možno používat intuitivně, bez dalších nápověd, které by uživatel musel studovat. Proto systém neobsahuje dokumentaci ani nápovědu.

Shrnutí heuristiky

Po provedení heuristiky byly opraveny všechny nalezené nedostatky (P1 – Zablokování prostředí s informací o stavu, P2 – Návrat z vyhledávacího dialogu). Výsledné uživatelské rozhraní bude dále testováno.

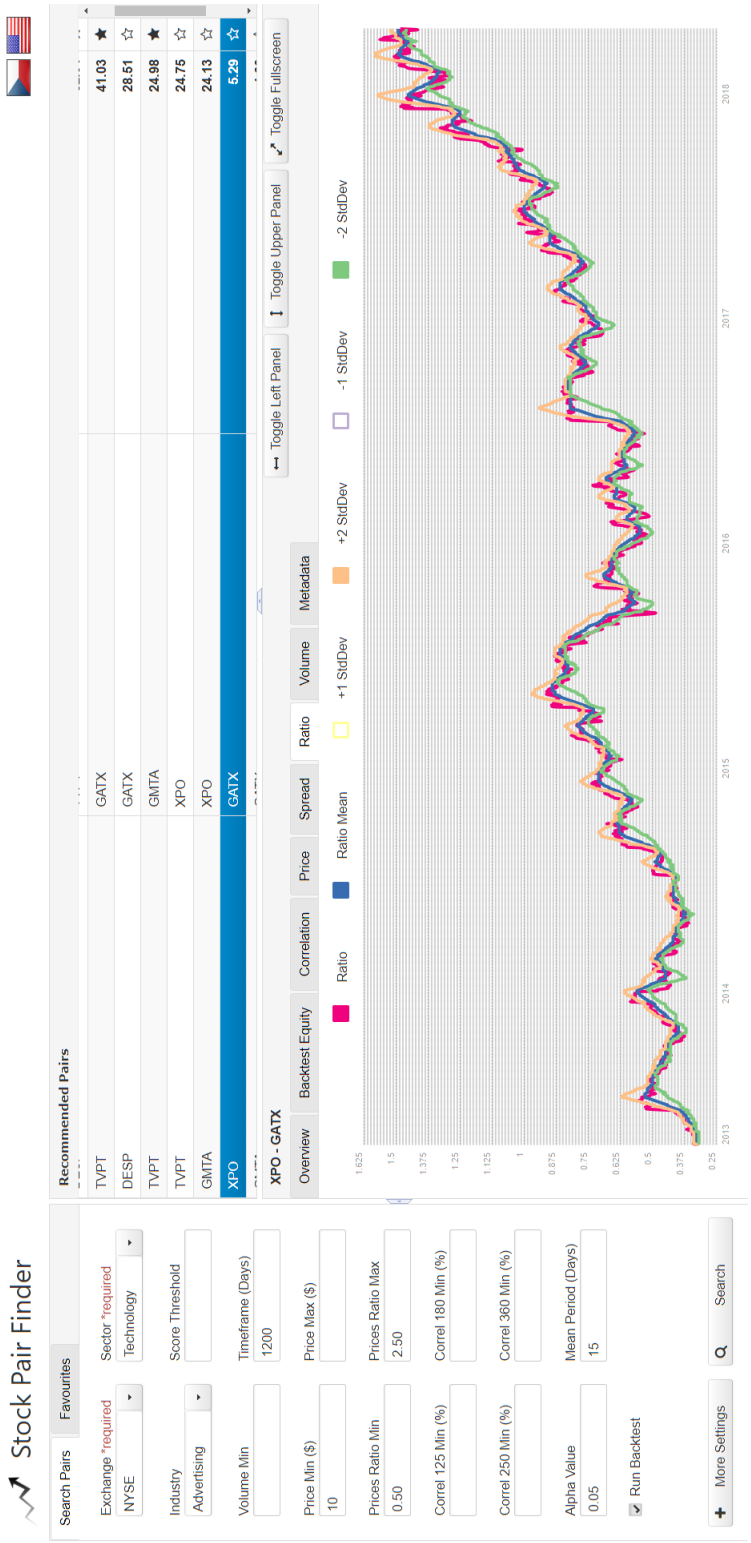
3.5 Ukázka aplikace

Sekce ukazuje finální podobu výsledné aplikace. Nejedná se pouze o výstup implementační fáze, ale jde o výsledek po provedení uživatelského testování a zapracování výstupů z něj. Uživatelskému testování se věnuje následující kapitola 4.2.

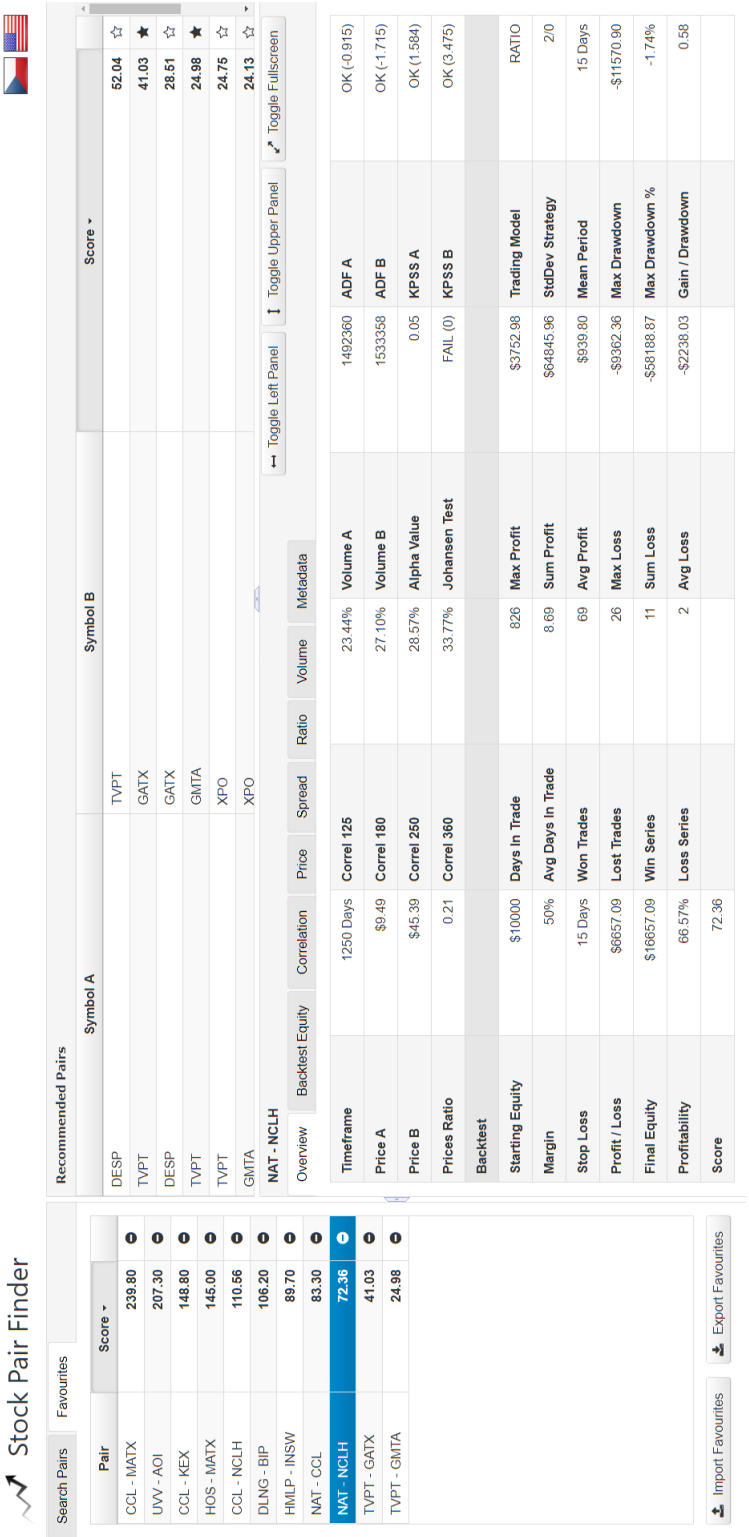
Na obrázku 3.3 lze vidět vyhledávací formulář se seznamem ohodnocených výsledků vyhledávání. Náhled detailu páru zobrazuje panel s vývojem poměru cen, klouzavý průměr této hodnoty (mean) a k němu přičtené, či odečtené násobky klouzavého průměru hodnot standardní odchylky

z mean. Graf slouží k posouzení strategie vstupů a výstupů z obchodů (enum *TradingStdDevStrategy.java*), případně k nalezení strategie vhodnější.

Druhý obrázek 3.4 zobrazuje funkcionalitu oblíbených. Seznam oblíbených obsahuje nejenom vybrané položky z aktuálního seznamu výsledků (označené hvězdičkou), ale i naimportované kandidátní páry. Na detailu vybraného páru lze vidět základní přehled a statistiky, jako například průměrné ceny akcií, či výstup testu kointegrace. Dolní část pak zobrazuje konfiguraci a výstupy provedeného backtestu páru.



Obrázek 3.3: Ukázka aplikace: Výsledky hledání, graf poměru cen



Obrázek 3.4: Ukázka aplikace: Oblíbené, přehled statistik a výsledků backtestu

Kapitola 4

Testování

S předchozí implementační kapitolou úzce souvisí testování aplikace, které je neméně důležitou částí jakéhokoliv vývoje. Kromě jednotkového testování kódu bylo provedeno také uživatelské testování, kdy byl prototyp aplikace spolu s test cases předán vhodnému vzorku uživatelů a byla sledována jejich činnost [97] [98].

4.1 Jednotkové testování

Jako první úroveň testování bylo provedeno unit testování. S použitím knihoven JUnit a Mockito byly otestovány zejména balíčky *controller*, *service* a *utils*. Třídy v nich obsahují důležitou logiku a řídí chod celé aplikace. Provedeny byly jak pozitivní, tak negativní testovací scénáře. Krátké ukázky testů zobrazují kódy 4.1 a 4.2, kde lze vidět použití knihoven pro mockování a kontrolu provedené logiky v testovaném kusu kódu.

4.2 Uživatelské testy

Pro ověření použitelnosti implementovaného prototypu bylo provedeno testování za pomoci reálných uživatelů. Jedná se o velice důležitou část práce, protože jsou zde kladeny vysoké nároky na intuitivnost GUI a také z důvodu, že programátor sám nedokáže nasimulovat chování reálných uživatelů.

4.2.1 Testovací případy

Pro otestování aplikace byly zvoleny 3 testovací scénáře, které zahrnují use case pokrývající funkční požadavky F1, F2, F3, F4, F6 i F7.

Kód 4.1: Ukázka unit testu 1 (*SearchPairsBeanTest.java*)

```
/**
  * Load proper message after unsuccessful loading of exchange names
  * in {@link SearchPairsBean#loadExchangeNamesSet()}.
  */
@Test
public void loadExchangeNamesSetFailTest() {
  doThrow(new RuntimeException()).when(stockDataService).
      loadExchangeNames();
  searchPairsBean.loadExchangeNamesSet();

  verify(stockDataService).loadExchangeNames();
  verify(facesContext).addMessage(any(), any());
  verify(i18nService).getMessage("leftMenu", "stockListLoadFailed");
  assertNotNull(searchPairsBean.getExchangeNames());
}
```

Kód 4.2: Ukázka unit testu 2 (*SearchPairsBeanTest.java*)

```
/**
  * Test dialog return action with false argument
  * {@link SearchPairsBean#loadPairsOnDialogReturn(SelectEvent)}.
  */
@Test
public void loadPairsOnDialogReturnFailTest() {
  SelectEvent selectEvent = mock(SelectEvent.class);
  when(selectEvent.getObject()).thenReturn(Boolean.FALSE);

  searchPairsBean.loadPairsOnDialogReturn(selectEvent);
  verify(selectEvent).getObject();
  verify(pairsListBean, never()).prepareForNewSearching();
  verify(stockDataService, never()).loadStocksDataByFilter(any());
}
```

1) Nalezení korelace páru

- Zadání: Nalezněte kandidátní pár symbolů HOS-GLOG, který lze najít na burze NYSE, v sektoru Consumer Services a pod průmyslem Marine Transportation. Zobrazte graf korelace v posledním roce.
- Předpoklady: Načtený seznam burz, sektorů, průmyslů a symbolů dostupných akcií.
- Očekávaný scénář:
  1. Uživatel je na domovské stránce.
  2. Uživatel vyplní vyhledávací formulář dle zadání.

3. Uživatel stiskne tlačítko hledat.
4. Uživatel pohybem v seznamu nalezne požadovaný záznam.
5. Uživatel klikne na odpovídající řádku v tabulce.
6. Na detailu zvoleného páru uživatel kliknutím vybere záložku korelace.
7. Uživatel nalezne požadovaný graf korelace v posledním roce.

2) Export oblíbeného páru

- Zadání: Nalezněte libovolný kandidátní pár se skóre alespoň 50, pár přidejte do oblíbených a ty exportujte.
- Předpoklady: Načtený seznam burz, sektorů, průmyslů a symbolů dostupných akcií.
- Očekávaný scénář:
  1. Uživatel je na domovské stránce.
  2. Uživatel vyplní povinná pole (burza, sektor, průmysl) a spustí vyhledávání.
  3. Pokud aplikace nenalezne žádné výsledky, uživatel změní průmysl ve vyhledávacím formuláři a opakuje vyhledávání.
  4. Uživatel pohybem v seznamu výsledků nalezne pár se skóre alespoň 50.
  5. Uživatel klikne na symbol hvězdičky v daném řádku tabulky.
  6. Uživatel kliknutím vybere záložku oblíbené v levé části aplikace.
  7. Uživatel stiskne tlačítko pro export oblíbených.

3) Načtení dat importovaného páru

- Zadání: Importujte oblíbené kandidátní páry. Pro libovolný pár načtěte data a zobrazte graf korelace v posledním roce.
- Předpoklady: Načtený seznam burz, sektorů, průmyslů a symbolů dostupných akcií. Připravený validní soubor s oblíbenými pro import na lokálním pevném disku v domovském adresáři.
- Očekávaný scénář:
  1. Uživatel je na domovské stránce.
  2. Uživatel kliknutím vybere záložku oblíbené.
  3. Uživatel stiskne tlačítko pro import oblíbených.

4. Uživatel lokalizuje a vybere předem připravený soubor na disku.
5. Uživatel kliknutím na řádek tabulky oblíbených vybere libovolný kandidátní pár.
6. Uživatel stiskne tlačítko pro zpracování páru na detailu vybraného páru.
7. Na detailu zvoleného páru s načtenými daty uživatel kliknutím vybere záložku korelace.
8. Uživatel nalezne požadovaný graf korelace v posledním roce.

4.2.2 Průběh testování

Cílovou skupinu představují ekonomicky vzdělanější lidé, často vytížení, zvyklí ovládat klasické aplikace pro bussines jako Microsoft Outlook nebo webový prohlížeč. Pro testování byl vybrán vzorek těchto uživatelů, lišících se schopnostmi v IT světě, tedy i zkušeností s ovládáním různých systémů.

Tester 1

- Charakteristika: 24 let, student vysoké školy se zaměřením na IT, pokročilý uživatel
- Scénář 1: Tester postupoval podle očekávaného scénáře. Při žádném kroku se neobjevily delší prodlevy nebo zaváhání. Při druhém kole testování tester navrhl přidání klávesových zkratek pro rychlejší vstup a výstup z celoobrazovkového režimu detailu vybraného páru.
- Scénář 2: Při žádném kroku se neobjevily delší prodlevy nebo zaváhání. Oproti očekávanému scénáři tester rozbalil dialog s rozšířenými možnostmi a zadal hodnotu pro minimální skóre dle zadání, dále pokračoval dle očekávaného scénáře.
- Scénář 3: Tester postupoval podle očekávaného scénáře. Při žádném kroku se neobjevily delší prodlevy nebo zaváhání.

Tester 2

- Charakteristika: 50 let, ekonom, méně zdatný uživatel
- Scénář 1: Tester navrhl zvětšení náhledu vybraného páru na celou obrazovku. Grafy by se staly přehlednější a mohly by být podrobnější. Jinak tester postupoval bez prodlev dle očekávaného scénáře.
- Scénář 2: Tester postupoval podle očekávaného scénáře. Při žádném kroku se neobjevily delší prodlevy nebo zaváhání.

- Scénář 3: Tester znovu připomenul možnost zvětšení náhledu vybraného páru na celou obrazovku. Scénář splnil bez prodlev a dle očekávaného scénáře.

Tester 3

- Charakteristika: 55 let, bankéř, méně zdatný uživatel
- Scénář 1: Na začátku testování přepnul uživatel aplikaci do českého jazyka, kvůli snazší orientaci. Jinak postupoval podle očekávaného scénáře.
- Scénář 2: Tester postupoval podle očekávaného scénáře. Při žádném kroku se neobjevily delší prodlevy nebo zaváhání.
- Scénář 3: Po dokončení importu oblíbených tester očividně nevěděl, co se děje, vznikla prodleva. Vysvětlení je, že očekával potvrzovací zprávu o úspěšném importu oblíbených.

4.2.3 Zhodnocení testování

Uživatelské testování bylo provedeno po Nielsenovo heuristické analýze (viz kapitola 3.4). Během testování bylo identifikováno několik nedostatků, a sice:

Náhled detailu přes celou obrazovku

- Popis: Tester 2 by uvítal zvětšení detailu kandidátního páru, hlavně kvůli grafům. Připomíná možnosti Microsoft Excel.
- Řešení: Přidat tlačítko pro zvětšení náhledu na celou obrazovku a návrat z ní.

Oznámení o úspěšném importu oblíbených párů

- Popis: Tester 3 navrhuje přidání zpráv informujících o úspěšně, či neúspěšně provedené akci, konkrétně po importu oblíbených. Srovnává s bankovními systémy, které o takových věcech informují.
- Řešení: Zobrazování zpráv po dokončení akcí, dojde ke zvýšení informovanosti uživatele o stavu aplikace, viz první bod v Nielsenově heuristické analýze 3.4.

Tyto nedostatky byly po dokončení testování opraveny a aplikace předána uživatelům k opětovnému otestování. Nebyly zjištěny žádné další nedostatky, kromě testera 1 (profil viz 4.2.2), který si stěžoval na absenci klávesových zkratek pro zobrazení a zrušení celoobrazovkového detailu.

Klávesové zkratky pro náhled přes celou obrazovku

- Popis: Tester 1 by uvítal klávesové zkratky pro vstup a výstup z celoobrazovkového režimu. Jakožto pokročilý uživatel je zvyklý nepoužívat pouze myš pro ovládání aplikací.
- Řešení: Přidání podpory klávesových zkratek, a sice klávesa enter pro celoobrazovkový režim při pohybu v tabulkách a klávesa esc pro návrat. Způsobí posílení flexibility a efektivity používání aplikace, viz sedmý bod Nielsenovy heuristické analýzy 3.4.

Nedostatek byl opět zapracován a uživatel byl s řešením spokojen. Ostatní uživatelé již nebyli požádáni o další kolo testování. Ukázku z výsledné aplikace lze najít v kapitole 3.5.

Závěr

Shrnutí

Hlavním cílem diplomové práce bylo na základě nabytých poznatků o strategii párového obchodování s akciovými páry navrhnout novou analytickou aplikaci pro tuto strategii a implementovat ji pomocí jazyku Java EE 8.

V úvodní kapitole byl proveden popis dnes běžných druhů investování a také podrobné seznámení se se strategií párového obchodování s akciemi. Rešeršní část práce završuje přehled dostupného software s podporou akciových párů.

Kapitola zabývající se analýzou a návrhem nové aplikace nashromáždila funkční i nefunkční požadavky. Seznam případů užití pokryl všechny funkční požadavky. Nechybí ani doménový model. Následuje sekce zabývající se návrhem uživatelského rozhraní, který byl proveden pomocí wireframes pokrývajících jednotlivé případy užití. Dále byl vytvořen model architektury a návrhový model tříd. Kapitolu zakončuje model nasazení aplikace.

Následující kapitola, zabývající se realizací aplikace, na úvod představuje další vhodné nástroje pro realizaci navržené webové aplikace spustitelné na aplikačním serveru GlassFish 5. Druhá sekce se stručně věnuje problematice analýzy časových řad cen akcií. Kapitola dále popisuje zajímavé problémy, se kterými se autor při implementaci potýkal, což mu přineslo detailní seznámení se s posledními verzemi použitých technologií anebo konceptem kointegrace. Na prototypu uživatelského rozhraní byla provedena Nielsenova heuristická analýza uživatelského rozhraní a její výstupy byly zapracovány. Následuje ukázka prostředí aplikace.

Výstupem je analytická aplikace pro strategii párového obchodování s akciemi, která umožňuje vyhledání akciových párů, jejich ohodnocení, mimo jiné na základě výsledků backtestu, a následné doporučení nejlepších párů uživateli k obchodování. Podporováno je testování kointegrace včetně kontroly potřebných podmínek, stejně jako provádění backtestu na historických datech s možností konfigurace obchodní strategie. Tím byly úspěšně splněny všechny body zadání diplomové práce kromě jediného, a sice explicitního požadavku

na použití historických dat o akciích ze serveru *finance.google.com*, který byl v současné podobě identifikován jako nevhodný, více viz kapitola 3.3.1.

Poslední kapitola se zabývá testováním, a to nejenom jednotkovými testy logiky výsledné aplikace, ale také uživatelským testováním. To proběhlo testery, kteří reprezentují vzorek cílové skupiny uživatelů, tedy lidmi různých profesí, ale s určitými znalostmi a zkušenostmi v oblasti ekonomie. Nedostatky odhalené během provádění testovacích scénářů byly následně odstraněny.

Možnosti rozšíření

Do budoucna se nabízí několik rozšíření, jež v podstatě odrážejí funkcionalitu existujících komplexnějších nástrojů představených v první kapitole.

Hlavním nedostatkem je podpora omezeného počtu akciových burz, která však souvisí se zvoleným poskytovatelem dat a jeho službami. Rozšíření by tedy znamenalo možnost získávání historických dat i od jiných poskytovatelů, typicky placených. Realizace takového rozšíření je díky robustnímu návrhu aplikace v souladu s požadavky poměrně snadná, a sice spočívá v nové implementaci připraveného rozhraní.

Analýza časových řad kandidátních párů také poskytuje prostor ke zlepšení, respektive k implementaci dalších testů nebo ukazatelů. Těch existující aplikace nabízí nespočet, nicméně je nutné podotknout, že nástroje jsou také komplexnější a ne všechna funkcionalita je pro párové obchodování s akciemi nezbytná. Jako inspirace k další analýze akciových párů by mohla sloužit kniha Pairs Trading: Quantitative Methods and Analysis od Ganapathy Vidyamurthy [27].

Logickým rozšířením analytické aplikace je i podpora zadávání obchodních příkazů přímo v aplikaci. Za tímto účelem by bylo potřeba aplikaci rozšířit o službu pro komunikaci s API vybraného brokera a odpovídající úpravu uživatelského rozhraní, například přidání dalšího panelu na obrazovce detailu akciového páru. S napojením na brokera souvisí i použití jeho historických dat, což je snadno realizovatelné jak již bylo zmíněno dříve.

Literatura

[1] Petr Tmej: *Příručka úspěšného obchodování na burze*. 2015, s. 4-5 [cit. 2018-03-16]. Dostupné z: http://aostrading.cz/kniha/

[2] Český statistický úřad: Ceny bytů. [online], [cit. 2018-03-17]. Dostupné z: https://www.czso.cz/csu/czso/ceny_bytu

[3] Česká národní banka: Jak se vyvíjela dvoutýdenní repo sazba ČNB? [online], [cit. 2018-03-17]. Dostupné z: https://www.cnb.cz/cs/faq/jak_se_vyvijela_dvoutydenni_repo_sazba_cnb.html

[4] König, M.: Ekonomie pro neekonomy – Úrokové sazby ČNB. [online], January 2016, [cit. 2018-03-17]. Dostupné z: hhttp://www.kenig.cz/2016/01/22/ekonomie-pro-neekonomy-urokove-sazby-cnb/

[5] Tománek, M.: Jak investovat do komodit? Tři kapitáni, jedna loď. [online], February 2013, [cit. 2018-03-17]. Dostupné z: https://www.penize.cz/komodity-a-futures/249588-jak-investovat-do-komodit-tri-kapitani-jedna-lod

[6] Ekonomika Online: Hyperinflace. [online], [cit. 2018-03-16]. Dostupné z: http://ekonomikaonline.cz/349/hyperinflace/

[7] Redakce Peníze.cz: Hyperinflace v Německu 1923. [online], [cit. 2018-03-16]. Dostupné z: https://www.penize.cz/15896-hyperinflace-v-nemecku-1923

[8] ČTK: Hyperinflace v Zimbabwe: Bochník chleba stojí 35 milionů. [online], December 2008, [cit. 2018-03-16]. Dostupné z: https://byznys.ihned.cz/c1-31306980-hyperinflace-v-zimbabwe-bochnik-chleba-stoji-35-milionu

[9] Zámečník, P.: Není všechno zlato... aneb Nenechte se svést grafy. [online], December 2014, [cit. 2018-03-16]. Dostupné z: https:

//www.investujeme.cz/clanky/neni-vsechno-zlatoundefined-aneb-nenechte-se-svest-grafy/

[10] Měšec.cz: Jak fungují podílové fondy? [online], [cit. 2018-03-16]. Dostupné z: https://www.mesec.cz/financni-portal/ucty/jak-funguji-podilove-fondy/

[11] SPIVA®: *SPIVA® U.S. Scorecard, Year-End 2017*. 2018, [cit. 2016-03-16]. Dostupné z: https://us.spindices.com/documents/spiva/spiva-us-year-end-2017.pdf

[12] Vobořil, T.: Univerzita Forexu: Co je Forex. [online], May 2015, [cit. 2018-03-16]. Dostupné z: https://www.penize.cz/forex/255446-univerzita-forexu-co-je-forex

[13] Štěpán Petřivý: Základní pojmy z forexu. Co je to lot, páka, stop loss? [online], September 2017, [cit. 2018-03-16]. Dostupné z: https://opcebinarni.com/forex-co-je-lot-paka-stop-loss/

[14] Investopedia, LLC.: Buy And Hold. [online], 2018, [cit. 2018-03-11]. Dostupné z: https://www.investopedia.com/terms/b/buyandhold.asp

[15] FXstreet.cz: Burzovní parket. [online], [cit. 2018-03-17]. Dostupné z: https://www.fxstreet.cz/forex-slovnik-pojmu+burzovni-parket.html

[16] Tým TradeandFinance.eu: Burza. [online], [cit. 2018-03-11]. Dostupné z: http://www.tradeandfinance.eu/slovnik-pojmu/burza/

[17] Patria Finance, a.s.: Obecné - Burza. [online], 2018, [cit. 2018-03-11]. Dostupné z: https://www.patria.cz/slovnik/166/burza.html

[18] Petr Tmej: *Příručka úspěšného obchodování na burze*. 2015, s. 26-28 [cit. 2018-03-16]. Dostupné z: http://aostrading.cz/kniha/

[19] Patria Finance, a.s.: Obecné - Burza. [online], 2018, [cit. 2018-03-17]. Dostupné z: https://www.patria.cz/slovnik/2/akcie.html

[20] Investopedia, LLC.: S&P 500 Index: A Performance Analysis of Long-Term Returns. [online], 2018, [cit. 2018-03-17]. Dostupné z: https://www.investopedia.com/articles/markets/061216/sp-500-index-performance-analysis-longterm-returns.asp

[21] Kennedy, M.: Learn the Basics About Commodity ETFs. [online], February 2017, [cit. 2018-03-17]. Dostupné z: https://www.thebalance.com/learn-the-basics-about-commodity-etfs-1214745

[22] Patria Finance, a.s.: Dluhopis (Bond, obligace). [online], 2018, [cit. 2018-03-17]. Dostupné z: https://www.patria.cz/slovnik/98/dluhopis-bond-obligace.html

[23] Tyleček, J.: Futures kontrakty: základní principy a informace. [online], October 2009, [cit. 2018-03-17]. Dostupné z: https://byznys.ihned.cz/c1-38767880-futures-kontrakty-zakladni-principy-a-informace

[24] Nesnídal, T.: Začínáme s opcemi (1). [online], December 2006, [cit. 2018-03-17]. Dostupné z: https://www.financnik.cz/komodity/fin_home/opce-komodity-akcie-forex-1.html

[25] Podhajský, P.: Spready: jak obchodovat komodity jinak (1/3). [online], June 2005, [cit. 2018-03-17]. Dostupné z: https://www.financnik.cz/komodity/zkusenosti/komoditni-spready.html

[26] Hanel, D.: *Statistické arbitráže na akciových trzích*. Diplomová práce, Vysoká škola ekonomická v Praze, Fakulta financí a účetnictví, Katedra bankovnictví a pojišťovnictví, January 2014. Dostupné z: https://vskp.vse.cz/61226

[27] Vidyamurthy, G.: *Pairs Trading: Quantitative Methods and Analysis*. John Wiley & Sons, Inc., první vydání, 2004, ISBN 0-471-46067-2.

[28] C.W.J.Granger: Some Properties of Time Series Data and Their Use in Econometric Model Specification. *Journal of Econometrics*, 1981, [cit. 2018-04-21]. Dostupné z: https://doi.org/10.1016/0304-4076(81)90079-8

[29] Robert F. Engle and C. W. J. Granger: Co-integration and error correction: Representation, estimation and testing. *Econometrica*, 1987, [cit. 2018-04-21]. Dostupné z: https://www.jstor.org/stable/1913236

[30] Eric Zivot: Unit Root Tests. 2006, [cit. 2018-04-21]. Dostupné z: https://faculty.washington.edu/ezivot/econ584/notes/unitroot.pdf

[31] Eric Zivot: Cointegration. 2006, [cit. 2018-04-21]. Dostupné z: https://faculty.washington.edu/ezivot/econ584/notes/cointegration.pdf

[32] Ježková, M.: *Kointegrace a její aplikace ve financích*. Diplomová práce, Masarykova univerzita, Přírodovědecká fakulta, Ústav matematiky a statistiky, 2013. Dostupné z: https://is.muni.cz/th/l5le4/DP_Martina_Jezkova.pdf

[33] Štěpán Chrz: *Provázanost trhů potravin, biopaliv a fosilních paliv: Kointegrační analýza*. Bakalářská práce, Karlova univerzita v Praze, Fakulta sociálních věd, Institut ekonomických studií, 2012. Dostupné z: https://is.muni.cz/th/l5le4/DP_Martina_Jezkova.pdf

[34] Vobořil, T.: Jak si vybrat obchodní software. [online], May 2011, [cit. 2018-03-17]. Dostupné z: https://www.financnik.cz/komodity/zkusenosti/jak-si-vybrat-obchodni-software.html

[35] Interactive Brokers: Interactive Brokers. [online], [cit. 2018-03-17]. Dostupné z: https://www.interactivebrokers.com/en/home.php

[36] Sierra Chart: Sierra Chart - Features. [online], December 2016, [cit. 2018-03-17]. Dostupné z: http://www.sierrachart.com/index.php?page=doc/Features.php

[37] Sierra Chart: Description of Service Packages and Pricing. [online], January 2018, [cit. 2018-03-17]. Dostupné z: http://www.sierrachart.com/index.php?page=doc/Packages.php

[38] Gecko Software: Stocks Charting Software. [online], [cit. 2018-03-17]. Dostupné z: https://www.trackntrade.com/stocks/

[39] TradeStation Group, Inc.: TradeStation. [online], [cit. 2018-03-17]. Dostupné z: https://www.tradestation.com/

[40] TradeStation Group, Inc.: TradeStation International Ltd and Interactive Brokers UK Ltd Plan to Offer Global Trading Account. [online], January 2018, [cit. 2018-03-17]. Dostupné z: https://globenewswire.com/news-release/2018/01/16/1289712/0/en/TradeStation-International-Ltd-and-Interactive-Brokers-UK-Ltd-Plan-to-Offer-Global-Trading-Account.html

[41] AmiBroker.com: AmiBroker Features. [online], 2016, [cit. 2018-03-17]. Dostupné z: https://www.amibroker.com/features.html

[42] Oracle Corporation and/or its affiliates: GlassFish - Download. [online], [cit. 2018-03-17]. Dostupné z: https://javaee.github.io/glassfish/download

[43] Balsamiq Studios: Balsamiq Products. [online], [cit. 2018-03-22]. Dostupné z: https://balsamiq.com/products/

[44] Despoudis, F.: Understanding SOLID Principles: Dependency Inversion. [online], [cit. 2018-05-03]. Dostupné z: https://codeburst.io/understanding-solid-principles-dependency-injection-d570c15560ab

[45] The Apache Software Foundation: Welcome to Apache Maven. [online], March 2018, [cit. 2018-03-17]. Dostupné z: https://maven.apache.org/

[46] Oracle Corporation and/or its affiliates: JSR 346: Contexts and Dependency Injection for JavaTM EE 1.1. [online], [cit. 2018-03-31]. Dostupné z: https://jcp.org/en/jsr/detail?id=346

[47] Oracle Corporation and/or its affiliates: Applying @Alternative Beans and Lifecycle Annotations. [online], [cit. 2018-04-29]. Dostupné z: https://netbeans.org/kb/docs/javaee/cdi-validate.html

[48] Daněček, J.: CDI, BI-EJA. 2016, [cit. 2018-03-31]. Dostupné z: https://edux.fit.cvut.cz/archive/B162/BI-EJA/

[49] Adámek, P.: Efektivní vývoj v Java EE a využití návrhových vzorů (JEEDP). [online], [cit. 2018-03-31]. Dostupné z: https://www.fi.muni.cz/~xadamek2/DesignPatternsTraining

[50] Chris Schalk, Oracle Corporation: Introduction to Javaserver Faces - What is JSF? [online], April 2005, [cit. 2018-03-17]. Dostupné z: http://www.oracle.com/technetwork/topics/index-090910.html

[51] PrimeFaces: Who uses PrimeFaces. [online], [cit. 2018-03-23]. Dostupné z: https://www.primefaces.org/whouses/

[52] Spring Framework: Spring Framework. [online], [cit. 2018-03-23]. Dostupné z: https://projects.spring.io/spring-framework/

[53] PrimeFaces: Why PrimeFaces. [online], [cit. 2018-03-17]. Dostupné z: https://www.primefaces.org/whyprimefaces/

[54] PrimeFaces: Primefaces. [online], [cit. 2018-03-23]. Dostupné z: https://github.com/primefaces/primefaces

[55] The jQuery Foundation: ThemeRoller. [online], [cit. 2018-03-22]. Dostupné z: http://jqueryui.com/themeroller/

[56] PrimeFaces: PrimeFaces Themes. [online], [cit. 2018-03-22]. Dostupné z: https://www.primefaces.org/themes/

[57] PrimeFaces: PrimeFaces Extensions. [online], [cit. 2018-03-17]. Dostupné z: http://primefaces-extensions.github.io/

[58] OmniFaces: OmniFaces. [online], [cit. 2018-03-17]. Dostupné z: http://omnifaces.org/

[59] Chartist.js: Chartist.js on GitHub. [online], [cit. 2018-03-22]. Dostupné z: http://gionkunz.github.io/chartist-js/

[60] Alimam, H.: hatemalimam/ChartistJSF on GitHub. [online], [cit. 2018-03-22]. Dostupné z: https://github.com/hatemalimam/ChartistJSF

[61] JUnit: JUnit - About. [online], [cit. 2018-03-17]. Dostupné z: https://junit.org/junit5/

[62] Mockito: Mockito. [online], [cit. 2018-03-17]. Dostupné z: http://site.mockito.org/

[63] Numerical Method Inc.: SuanShu. [online], [cit. 2018-04-21]. Dostupné z: http://numericalmethod.com/suanshu/

[64] Numerical Method Inc.: AlgoQuant. [online], [cit. 2018-04-21]. Dostupné z: http://numericalmethod.com/algoquant/

[65] Numerical Method Inc.: Welcome to Numerical Method! [online], [cit. 2018-04-21]. Dostupné z: http://numericalmethod.com/up/welcome-to-numerical-method/

[66] Numerical Method Inc.: SuanShu 20120606. [online], [cit. 2018-04-21]. Dostupné z: http://numericalmethod.com/suanshu-20120606/

[67] Andale, S.: KPSS Test: Definition and Interpretation. October 2017, [cit. 2018-04-24]. Dostupné z: http://www.statisticshowto.com/kpss-test/

[68] Statsmodels: Statsmodels. [online], [cit. 2018-04-24]. Dostupné z: http://www.statsmodels.org/stable/index.html

[69] Nelson, J.: How much longer will the Google Finance RESTFUL API function? [online], July 2014, [cit. 2018-03-27]. Dostupné z: https://www.quora.com/How-much-longer-will-the-Google-Finance-RESTFUL-API-function/answer/Jeff-Nelson-32?srid=uzvFF

[70] Schwartz, B.: New Google Finance Has Launched, Dropping The Portfolio Feature. [online], November 2017, [cit. 2018-03-27]. Dostupné z: https://www.seroundtable.com/new-google-finance-24837.html

[71] Janeczko, T.: AmiQuote Yahoo Historical stopped working. [online], May 2017, [cit. 2018-03-27]. Dostupné z: http://www.amibroker.com/kb/2017/05/17/amiquote-yahoo-historical-stopped-working/

[72] Nixon: Is Yahoo! Finance API broken? [online], May 2017, [cit. 2018-03-27]. Dostupné z: https://forums.yahoo.net/t5/Yahoo-Finance-help/Is-Yahoo-Finance-API-broken/td-p/250503/page/3

[73] c0redumb: yahoo_quote_download on GitHub. [online], [cit. 2018-03-27]. Dostupné z: https://github.com/c0redumb/yahoo_quote_download

[74] IEX Group, Inc.: IEX Developer Platform. [online], [cit. 2018-03-27]. Dostupné z: https://iextrading.com/developer/

[75] Alpha Vantage, Inc.: About Alpha Vantage. [online], [cit. 2018-03-27]. Dostupné z: https://www.alphavantage.co/support/#api-key

[76] Srivastava, V.: Alpha Vantage API Review on GitHub. [online], November 2017, [cit. 2018-03-27]. Dostupné z: https://github.com/RomelTorres/alpha_vantage/issues/13#issuecomment-344839209

[77] Nasdaq.com: Company List (NASDAQ, NYSE, & AMEX). [online], [cit. 2018-03-27]. Dostupné z: https://www.nasdaq.com/screening/company-list.aspx

[78] wickenden g: Alpha Vantage API Review on GitHub. [online], August 2017, [cit. 2018-03-27]. Dostupné z: https://github.com/RomelTorres/alpha_vantage/issues/13#issue-251401978

[79] uniVocity team: Welcome to uniVocity-parsers on GitHub. [online], 2018, [cit. 2018-03-29]. Dostupné z: https://github.com/uniVocity/univocity-parsers/

[80] patriques82: alphavantage4j Java wrapper on GitHub. [online], 2018, [cit. 2018-04-27]. Dostupné z: https://github.com/patriques82/alphavantage4j

[81] Oracle Corporation and/or its affiliates: Mojarra JavaServer Faces - Oracle's open source implementation of the JSF standard. [online], 2014, [cit. 2018-03-29]. Dostupné z: https://javaserverfaces.github.io/download.html

[82] Tijms, A.: Unable to inject FacesContext against GlassFish 5 (JSF 2.3, CDI 2.0). [online], [cit. 2018-03-29]. Dostupné z: https://github.com/javaee/glassfish/issues/22094

[83] Tijms, A.: JSF Facelet can not recognize CDI beans. [online], [cit. 2018-03-29]. Dostupné z: https://github.com/javaserverfaces/mojarra/issues/4264

[84] w3schools.com: ISO Language Codes. [online], [cit. 2018-03-29]. Dostupné z: https://www.w3schools.com/tags/ref_language_codes.asp

[85] DevelopersBook.com: JSF Lifecycle. [online], [cit. 2018-03-29]. Dostupné z: http://developersbook.com/jsf/images/JSF-Lifecycle.png

[86] Bhale, P.: Validation in JSF. [online], [cit. 2018-03-30]. Dostupné z: http://incepttechnologies.blogspot.cz/p/validation-in-jsf.html

[87] Oracle Corporation and/or its affiliates: JSR-000303 Bean Validation. [online], [cit. 2018-03-29]. Dostupné z: https://jcp.org/aboutJava/communityprocess/final/jsr303/index.html

[88] Scholtz, B.: JSF 2.0 tutorial with Eclipse and Glassfish. [online], January 2011, [cit. 2018-03-29]. Dostupné z: http://balusc.omnifaces.org/2011/01/jsf-20-tutorial-with-eclipse-and.html#FinetuningValidation

[89] Red Hat community: Hibernate Validator. [online], [cit. 2018-03-29]. Dostupné z: http://hibernate.org/validator/documentation/

[90] PrimeFaces Extensions: Layout. [online], March 2018, [cit. 2018-03-29]. Dostupné z: https://www.primefaces.org/showcase-ext/views/layout.jsf

[91] PrimeFaces.com: Dialog Framework. [online], March 2013, [cit. 2018-03-29]. Dostupné z: https://www.primefaces.org/dialog-framework/

[92] PrimeTek: *PrimeFaces 6.1 manual*. April 2017, s. 585 [cit. 2018-03-29]. Dostupné z: https://www.primefaces.org/docs/guide/primefaces_user_guide_6_1.pdf

[93] Oracle Corporation and/or its affiliates: Faces Core - Tag viewAction. [online], [cit. 2018-03-29]. Dostupné z: https://docs.oracle.com/javaee/7/javaserver-faces-2-2/vdldocs-facelets/f/viewAction.html

[94] Code Yellow B.V.: chartist-plugin-legend on GitHub. [online], [cit. 2018-03-29]. Dostupné z: https://github.com/CodeYellowBV/chartist-plugin-legend

[95] Svoboda, V.: Nielsen's Heuristic Evaluation. [online], December 2011, [cit. 2018-03-22]. Dostupné z: http://blog.vojtasvoboda.cz/nielsens-heuristic-evaluation

[96] Nielsen, J.: 10 Usability Heuristics for User Interface Design. [online], 1995 January, [cit. 2018-03-22]. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/

[97] Štrobl, R.: BI-ATS, Handout č. 1 - Úvod do testování software. 2017, [cit. 2018-03-22].

[98] Štrobl, R.: BI-ATS, Handout č. 2 - Test management, příprava automatizace QA. 2017, [cit. 2018-03-22].

Příloha **A**

Seznam použitých zkratek

AFL AmiBroker Formula Language

AMEX American Stock Exchange

API Application Programming Interface

CDI Contexts and Dependency Injection

CSS Cascading Style Sheets

CSV Comma-separated values

ČNB Česká národní banka

ČSÚ Český statistický úřad

DJIA Dow Jones Industrial Average

EL Unified Expression Language

ETFs Exchange Trated Funds

FOREX Foreign Exchange

GUI Graphical User Interface

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

IT Information technology

Java EE Java Enterprise Edition

JS JavaScript

JSF JavaServer Faces

JSON JavaScript Object Notation

JSP JavaServer Pages

JSR Java Specification Requests

MACD Moving Average Convergention/Divergention

MVC Model–view–controller

NASDAQ National Association of Securities Dealers Automated Quotations

NYSE New York Stock Exchange

OTC Over-the-counter

REST REpresentational State Transfer

S&P Standard and Poor's

SSTO Slow Stochastics

UI User Interface

URL Uniform Resource Locator

US United States

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu disku
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `src` ..... zdrojové kódy Stock Pair Finder
    - `pom.xml` ..... Maven Project Object Model
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `DP_Maleček_Kamil_2018.pdf` ..... text práce ve formátu PDF