VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

AUTOMATICKÉ OBCHODNÍ SYSTÉMY

AUTOMATED TRADING SYSTEMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VÍTĚZSLAV ŠAFÁŘ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAROSLAV ROZMAN, Ph.D.**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

154685

Ústav: Ústav inteligentních systémů (UITS)
Student: **Šafář Vítězslav**
Program: Informační technologie
Název: **Automatické obchodní systémy**
Kategorie: Umělá inteligence
Akademický rok: 2023/24

Zadání:

1. Nastudujte princip obchodování na burze a prostudujte si jednotlivé brokery. Zaměřte se na brokera XTB a nastudujte jeho API pro obchodování.
2. Navrhněte Automatický obchodní systém (AOS), který bude u XTB automaticky provádět obchody podle předem daného plánu. Navrhněte různě komplexní AOS od jednoduchých, které jen provádějí předem dané obchody, až po komplexní, založené na historických burzovních datech a používajících třeba neuronové sítě.
3. Navržený AOS implementujte a otestujte na historických datech.

Literatura:

- Gladiš, D., Naučte se investovat, Grada, 2. vydání, 2016, ISBN 978-80-247-1205-5
- Gladiš, D., Akciové investice, Grada, 2. vydání, 2021, ISBN 978-80-3122-8

Při obhajobě semestrální části projektu je požadováno:
První dva body zadání

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Rozman Jaroslav, Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1.11.2023
Termín pro odevzdání: 9.5.2024
Datum schválení: 6.11.2023

Abstrakt

O obchodování na finančním trhu slyšel v dnešní době snad téměř každý, avšak automatické obchodování je pro většinu stále novinkou. Cílem této bakalářské práce je navrhnout a vytvořit několik automatických obchodním systémů, a to za použití aplikačního programového rozhraní poskytovaného společností XTB, a následně tyto automatické obchodní systémy vyhodnotit na historických datech. Práce představuje čtyři různě komplexní automatické obchodní systémy, které dosahují různých zisků při určitých úrovních rizika. Práce dále demonstruje použitelnost zmíněného aplikačního programového rozhraní společnosti XTB. Nejlepším navrženým systémem byl vyhodnocen systém využívající MACD indikátor, jenž dosahoval průměrného ročního zhodnocení okolo 13.5 % s úrovní rizika ztrát, která se pohybovala okolo 39 %.

Abstract

Trading in the financial market is something almost everyone has heard of these days, but automated trading is still a novelty for most. The aim of this bachelor's thesis is to design and create several automatic trading systems using the application programming interface provided by XTB, and subsequently evaluate these automated trading systems using historical data. The thesis presents four differently complex automated trading systems, achieving various profits at certain risk levels. Furthermore, the thesis demonstrates the usability of the mentioned XTB application programming interface. The best-designed system evaluated was the one utilizing the MACD indicator,which achieved an average annual return of around 13.5 % with a level of risk of loss, approximately 39 %.

Klíčová slova

Automatický Obchodní Systém, AOS, Automatické obchodování, Algoritmické obchodování, XTB, Obchodování, xAPI, Obchodování na finančním trhu, Akcie, Forex. Index, Kryptoměny, ETF, Indikátory, Plovoucí průměr, Jednoduchý plovoucí průměr, Exponenciální plovoucí průměr, Konvergence/divergence plovoucího průměru

Keywords

Automatic Trading System, Automated Trading System, ATS, Automatic Trading, Automated Trading, Algorithmic Trading, XTB, Trading, xAPI, Trading on the Financial Market, Stocks, Forex, Index, Cryptocurrencies, ETF, Indicators, Moving average, Simple Moving average, Exponenctial Moving average, Moving average convergence/divergence

Citace

ŠAFÁŘ, Vítězslav. *Automatické obchodní systémy.* Brno, 2024. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jaroslav Rozman, Ph.D.

Automatické obchodní systémy

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jaroslava Rozmana, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Vítězslav Šafář
1. května 2024

Poděkování

Tímto bych chtěl poděkovat všem, kteří mě při vypracování této práce podporovali a dodávali mi motivaci k jejímu dokončení. Jmenovitě bych chtěl poděkovat svým rodičům a své přítelkyni za veškerou jejich podporu. Také bych chtěl poděkovat svému vedoucímu práce panu Ing. Jaroslavu Rozmanovi, Ph.D. za veškerou pomoc a rady, které mi při vypracování práce poskytl.

Obsah

1 Úvod **7**
1.1 Automatický obchodní systém 8

2 Finanční trhy **9**
2.1 Akcie 9
2.2 Forex 9
2.3 Kryptoměny 10
2.4 Index 10
2.5 Komodity 10
2.6 ETF 10

3 Obchodování na finančním trhu **11**
3.1 Fundamentální a technická analýza 12
3.2 Obchodní strategie 16

4 Aplikační programové rozhraní **18**
4.1 Typy přenosu dat 18
4.1.1 Formát XML 18
4.1.2 Formát JSON 18
4.2 Typy API 19
4.2.1 REST API 19
4.2.2 RPC 20
4.2.3 SOAP API 20
4.2.4 HATEOAS 20

5 XTB API – xAPI **21**
5.1 Formát dat 21
5.2 Wrapper od XTB 22
5.2.1 Tvorba požadavku 22
5.2.2 Důležité příkazy 22
5.2.3 Příklady použití příkazů s využitím wrapperu 24
5.3 Výhody a nevýhody 25

6 Návrh **26**
6.1 Navržené AOS 26
6.2 Architektura systému 30
6.3 Grafické uživatelské rozhraní (Front end) 30
6.4 Server (Back end) 33

6.5 Databáze 33

7 Implementace **35**
7.1 Front end 35
7.1.1 Grafické uživatelské rozhraní 35
7.1.2 Komunikace s back endem 35
7.1.3 Získání kurzů měn 36
7.2 Back end 37
7.2.1 REST API 37
7.2.2 JPA – databáze 38
7.2.3 Zabezpečení 39
7.2.4 AOS 39

8 Testování **44**
8.1 Získání historických dat 44
8.2 Simulace 44
8.3 Porovnání výsledků simulace jednotlivých AOS 45
8.3.1 Automatické nakupování 45
8.3.2 Automatické vyvažování portfolia 46
8.3.3 Automatické obchodování s použitím plovoucích průměrů 49
8.4 Simulovaní na nejobchodovanějších ETF 52
8.5 Porovnání automatických obchodních systémů 53

9 Závěr **54**

Literatura **55**

Seznam obrázků

3.1 Svíčkový graf 13
3.2 Jednoduchý plovoucí průměr za 50 dní 14
3.3 Ukázka trendových kanálů 16

6.1 Schéma funkčnosti automatického nakupování 26
6.2 Schéma funkčnosti automatického vyvažování portfolia bez prodávání 27
6.3 Schéma funkčnosti automatického vyvažování portfolia s prodáváním 28
6.4 Schéma funkčnosti automatického obchodování na základě plovoucích průměrů 29
6.5 Návrh systému 30
6.6 Návrh přihlašovacího okna 31
6.7 Návrh domovského okna 31
6.8 Návrh okna pro vytváření nového příkazu pro vyvažování portfolia 32
6.9 Diagram schéma databáze 34

8.1 Vývoj procentuálního rozložení portfolia bez prodávání 48
8.2 Vývoj procentuálního rozložení portfolia s prodáváním 48
8.3 Demonstrace automatického obchodování na základě indikátoru SMA pro akcie společnosti Pandora A/S 50
8.4 Demonstrace automatického obchodování na základě indikátoru EMA pro akcie společnosti Pandora A/S 51

Seznam tabulek

8.1 Porovnání automatického nakupování při změně objemu finančních prostředků 46
8.2 Porovnání simulací při změně frekvence nakupování (opakovací interval) . . 46
8.3 Porovnání výsledků simulací nad portfoliem ze sedmi technických společností 47
8.4 Porovnání ročního zhodnocení portfolií při měsíčním intervalu obchodování 47
8.5 Porovnání indikátorů SMA a EMA 49
8.6 Porovnání ideálních instrumentů pro indikátory EMA a SMA 49
8.7 Porovnání simulací MACD při změně objemu finančních prostředků 51
8.8 Porovnání simulací na nejobchodovanějších ETF na platformě od XTB . . . 52

Seznam výpisů

4.1 Příklad XML dat 18
4.2 Příklad JSON dat 19
4.3 Příklad dat s odkazy [10] 20
5.1 Formát odesílaných dat 21
5.2 Formát přijatých dat 22
5.3 Formát přijatých dat s chybou 22
5.4 Příklad metody login 24
5.5 Příklad metody logout 24
5.6 Příklad metody getUserInfo 25
7.1 Implementace vytvoření instance OkHttpClient 36
7.2 Příklad tvorby HTTP požadavku s JWT 36
7.3 Příklad třídy s anotací RestController 37
7.4 Příklad přijetí požadavku s HTTP metodou DELETE 38
7.5 Příklad modelu dat tabulky symbol 38
7.6 Příklad rozhraní repozitáře 39
7.7 Příklad získání uživatele z JWT 39
7.8 Příklad cronExpression 39
7.9 Získání příkazů 40

Seznam použitých symbolů

Symbol	Význam	Jednotka
α	Vyhlazovací faktor	[-]
N	Časový interval neboli počet vzorků	[den; ...]
n	Pořadové číslo vzorku ($n \in \{2, N\}$)	[-]
E_n	Exponenciální plovoucí průměr při daném vzorku n	[CZK; ...]
C_n	Uzavírací tržní cena daného vzorku n	[CZK; ...]
V	Objem transakce	[-]
M	Finanční prostředky	[CZK; ...]
P	Nákupní cena instrumentu	[CZK; ...]
R	Kurz převodu z měnu účtu na měnu instrumentu	[-]
L_{step}	Krok přírůstku množství	[-]
L_{min}	Minimální množství, se kterým je možno obchodovat	[-]
M_i	Finanční prostředky potřebné pro dokoupení instrumentu	[CZK; ...]
p_i	Procentuální nastavení instrumentu	[%]
p	Součet procent všech instrumentů	[%]
V_i	Vlastněný objem instrumentu	[-]
m_{V_i}	Aktuální marže instrumentu s vlastněným množstvím	[CZK; ...]
u	Roční zhodnocení (úroková sazba)	[%]
r	Počet let	[-]
Q	Počáteční finanční prostředky	[CZK; ...]
Q_r	Finanční prostředky po r letech	[CZK; ...]

Kapitola 1

Úvod

O obchodování na finančním trhu již v dnešní době slyšel téměř každý a obchodování se stalo dnes mnohem přístupnějším než dříve, což vede k zapojení stále většího počtu jednotlivců do této aktivity. Významným faktorem je široká dostupnost obchodních aplikací, což umožňuje provádět obchody rychle a snadno.

Pro vstup na finanční trh je klíčovým článkem tzv. broker. V dnešní době se zvyšuje jejich počet a mezi nejznámější z nich patří společnosti jako XTB, RoboMarkets, Trading212, eToro, Interactive Brokers, Saxo Bank a další [14].

V rámci finančního trhu má každý jednotlivec rovnocenný přístup k možnostem investování, což poskytuje každému obchodníkovi stejnou šanci a rovnocenné podmínky při rozhodování o svých investičních aktivitách. Jedinou proměnnou v tomto rozhodování jsou finanční prostředky jednotlivce, protože každý jednotlivec obchoduje s rizikem ztráty investovaných prostředků. Mezi instrumenty finančního trhu, do kterých může jednotlivec investovat, patří akcie, forex, index, komodity, kryptoměny a ETF.

V minulosti i v současnosti se mnoho lidí zabývá vývojem automatických obchodních systémů. Představou těchto vývojářů je automatický systém, který bude dosahovat zisků s minimálním rizikem a bude tedy umožňovat vydělávat peníze i lidem, kteří o investování neví téměř nic. Stejně jako u klasického obchodování jsou i pro automatické systémy důležitým článkem brokeři. Avšak systém nedokáže využívat grafické uživatelské rozhraní, které používají obchodníci v obchodních aplikacích na svých chytrých telefonech nebo počítačích. Systém potřebuje aplikační programové rozhraní (API), jenž umožní systému komunikovat a využívat obchodní platformu brokera. Z výše uvedených brokerů umožňují využívat své API společnosti XTB, Trading212, eToro, Interactive Brokers a Saxo Bank.

Cílem této bakalářské práce je navrhnout, implementovat a otestovat několik různě komplexních automatických obchodních systému a následně porovnat jejich úspěšnost na historických datech. V rámci toho se zaměřit na zmíněné aplikační programové rozhraní společnosti XTB a otestovat jeho použitelnost pro vývoj a provoz různě komplexních automatických obchodních systémů. Návrhem systémů se zaobírá kapitola 6, detaily jejich implementace jsou popsány v kapitole 7 a kapitola 8 se zabývá jejich testováním a vyhodnocením jejich kvality s ohledem na poměr ziskovosti vůči podstoupenému riziku ztráty investovaných prostředků. Na základní funkcionalitu zmíněného API je zaměřena kapitola 5.

1.1 Automatický obchodní systém

Automatický obchodní systém (AOS) nebo také algoritmické obchodování [15] získává stále větší pozornost u investorů. Tento speciální software je navržen k automatickému otevírání a uzavírání obchodů na základě předem stanovených pravidel. Jeho účelem je usnadnit manuální práci investorů, minimalizovat rizika ztrát vyplývajících z nepozornosti nebo zaneprázdnění samotného investora a také omezuje vliv emocí na obchodování. Cílem tvůrců všech AOS je vytvořit systém, který bude dlouhodobě dosahovat zisku s minimální mírou rizika. Takové programy provádějí různá rozhodnutí o obchodování v reálném čase. Rozhodnutí mohou představovat:

- Výběr instrumentu finančního trhu, do kterého investovat.
- Kdy otevřít a kdy zavřít pozici. Více o pozicích v kapitole 3.
- Určení množství finanční prostředků pro danou investici.

AOS [9] mohou pracovat za dozoru člověka nebo úplně samostatně. Tvůrce AOS vždy definuje pravidla a strategie, podle kterých se systém následně rozhoduje. Většina AOS používá pouze jednu obchodní strategii. Automatické obchodní systémy se mohou mezi sebou velmi lišit v maličkostech nebo i v zásadních aspektech. Tak mohou vznikat různě komplexní AOS, od těch nejméně komplexních (například automatické nakupování), až po ty velmi komplexní v podobě obrovských neuronových sítí a umělé inteligence.

AOS je možno rozdělit do tří skupin. První skupinu tvoří ty, které pro rozhodování používají fundamentální analýzu, druhou skupinu ty, které používají technickou analýzu a třetí skupinou můžeme označit ty, které používají kombinaci obou druhů analýzy dat.

Kapitola 2

Finanční trhy

Finanční trh [22] je místo, kde se střetává nabídka a poptávka po finančních instrumentech. Jedná se o systém nástrojů, postupů, institucí a vztahů mezi nimi. Finanční trhy se dělí na peněžní a kapitálové trhy. Jednou ze základních částí kapitálového trhu jsou burzy, které organizují trhy s investičními nástroji. Typickým úkazem burz je, že předmět obchodování není při obchodu fyzicky přítomen. Pro automatický obchodní systém jsou důležité:

- burza cenných papírů,
- měnový trh,
- kryptoměnová burza,
- derivátový trh,
- komoditní burza.

2.1 Akcie

Akcie [25] představují cenný papír, pomocí kterého jejich držitelé (akcionáři) získávají majetkové a rozhodovací právo ve vztahu k akciové společnosti. Podle procentuálního poměru akcií vůči ostatním akcionářům mají právo se podílet na řízení této společnosti, jejím zisku a na likvidačním zůstatku při zrušení s likvidací. Obchoduje se s nimi na burze cenných papírů.

2.2 Forex

Forex (zkratka pro foreign exchange) [11] je způsob obchodování s měnami, kdy každý předmět obchodu je měnový pár. Při koupi měnového páru se ve stejný čas nakoupí primární měna a prodá sekundární. Měnový trh je největší a nejlikvidnější na světě.

2.3 Kryptoměny

Kryptoměny [12] jsou digitální měnou. Obchodovatelným kryptoměnám na finančním trhu se přezdívá investiční kryptoměny a mezi nejznámější patří například **Bitcoin**, **Ethereum** a **Litecoin** [2]. Pro udržení integrity a zabránění padělání se používá takzvaná blockchain technologie. Tato technologie je základem fungování kryptoměn.

2.4 Index

Index [3] představuje nástroj, který měří výkonnost nějakého sektoru či segmentu trhu. Může se jednat o část akciového či dluhopisového trhu, různé komodity nebo i úrokové sazby. Měřítka jsou vyvíjena standardizovanými a hlavně transparentními metodami.

Mezi nejdůležitější indexy patří:

- **Dow Jones Industrial Average (u XTB US30)** – 30 velkých společností kótovaných ve Spojených státech na NYSE nebo NASDAQ,
- **S&P 500 (US500)** – 500 největších společností ve Spojených státech amerických,
- **NASDAQ-100 (US100)** – 100 největších nefinančních společností kótovaných na akciovém trhu,
- **FTSE 100 (UK100)** – 100 společností kótovaných na londýnské burze,
- **EURO STOXX 50 (EU50)** – 50 největších a nejlikvidnějších akcií eurozóny,
- **Nikkei 225 (JAP225)** — 225 velkých společností kótovaných v Japonsku,
- **S&P/ASX 200 (AUS200)** – 200 největších společností kótovaných v Austrálii.

2.5 Komodity

Komodity [1] jsou produkty stejné kvality a hodnoty od různých producentů. Dělí se na takzvané **soft** a **hard** komodity. Soft komodity jsou ty, které se pěstují, hard komodity jsou ty, které se těží.

- Soft komodity:
  - káva, cukr, bavlna, kakao ...
- Hard Komodity:
  - zlato, měď, ropa ...

2.6 ETF

ETF (zkratka pro exchange-traded fund) [7] je fond, který umožňuje přístup k portfoliu podnikových akcií, dluhopisů, komodit nebo nemovitostí. Po zakoupení ETF získá kupující malou část tohoto portfolia vytvořeného s cílem sledovat určitý tržní index. Investoři tudíž dosahují částečných výnosů či ztrát, kterých dosahuje dané portfolio.

Kapitola 3

Obchodování na finančním trhu

Obchodování na finančním trhu se může ze začátku zdát jako snadná záležitost, ale po chvíli všichni začátečníci zjistí, že vydělávat si tímto způsobem není tak jednoduché. Úspěšné obchodování vyžaduje dlouhodobé sledování cenových změn, tržních podmínek a aktuálních událostí, které se ve světě dějí a které by mohly ovlivnit vývoj finančního trhu. Také to zahrnuje odhadování růstu či poklesu ceny podle určitých matematických výpočtů či modelů. Podstatné pro obchodování a předpovídání cenového vývoje je opakování se historických jevů, proto investoři sledují aktuální cenové změny a podle historických dat se snaží předpovídat, jaký bude budoucí vývoj.

Brokeři obvykle ve svých mobilních či webových aplikacích nabízí použití funkcí, které využívají technické indikátory pro usnadnění rozhodování investorů. Mezi nejpoužívanější technické indikátory patří:

- Exponential Moving Average (EMA),
- Moving Average Ribbon (MAR),
- Stochastic Oscillator (SO),
- Volume Density (VD),
- Relative Strange Indicator (RSI).

Dalšími užitečnými funkcemi, které brokeři nabízí, jsou automatické uzavírací pokyny **Stop Loss** a **Take Profit** [33]. Tyto funkce přidávají komfort investorům, protože zabraňují příliš velkým ztrátám nebo promeškání možnosti realizovat zisk. Stop Loss (v překladu zastavení ztrát) je automatický pokyn, která uzavře určitý obchod, pokud tržní cena postoupí proti pozici za určitý předem nastavený bod. Naopak Take Profit (v překladu získat zisk) je automatický pokyn, který uzavře obchod, pokud tržní cena postoupí za určitý bod ve směru pozice.

Pozice představuje investici do instrumentu finančního trhu. Otevřením pozice se zainvestují finanční prostředky do instrumentu za aktuální tržní cenu a do vlastnictví investora připadá určité množství daného instrumentu. Při uzavření pozice se toto množství za aktuální tržní cenu prodává. Pozice se dělí na **dlouhé** a **krátké**.

Dlouhá pozice [29] označuje situaci, kdy investor vydělává na růstu tržní ceny instrumentu. Při otevření dlouhé pozice si investor nakoupí určité množství instrumentu a při uzavření pozice toto množství prodá. Jeho ziskem se tudíž stává rozdíl mezi nákupní cenou při otevření a prodejní cenou při uzavření.

Pravým opakem dlouhé pozice je krátká pozice [32], což je termín pro situaci, kdy investor vydělává na poklesu ceny instrumentu. Investor si vypůjčí od brokera instrument, který otevřením pozice prodává a při uzavření pozice nakoupí zpět a vrací brokerovi. Ziskem z této transakce je pokles mezi cenou, za kterou instrument prodal, a cenou, za kterou následně tento instrument opět nakoupil. Při otevření krátké pozice se brokerovi platí poplatek za vypůjčení.

Další velmi využívanou možností obchodování na finančním trhu je používání **finanční páky**, kterou však nenabízí každý z brokerů a ne u všech instrumentů. Finanční páka [30] umožňuje dosáhnout vyšších zisků s nižším kapitálem. Funguje na principu vypůjčení si kapitálu od brokera v určitém poměru, který je stanoven pro daný instrument. Tento poměr je obvykle omezen hodnotou 1:20 u akcií a komodit, ale při obchodování měn tento poměr dosahuje až 1:200. Zjednodušeně řečeno, při poměru 1:20 investor obchoduje s 20krát větším objemem finančních prostředků, než by si mohl dovolit na základě svého kapitálu. To mu umožňuje dosáhnout potenciálně 20krát vyšších zisků, ale současně s sebou nese vysoké riziko 20krát vyšších ztrát. Při špatném využívání finanční páky může investor ve velmi krátkém čase přijít o veškerý svůj kapitál, proto se také doporučuje používat výše zmíněný pokyn stop loss pro zamezení rizika ztráty veškerých finančních prostředků investora. Pro otevření pozice s využitím finanční páky musí investor zaplatit určitou sumu, která mu bude při prodeji vrácena. U finanční páky 1:20 se právě jedná o $\frac{1}{20}$ ceny celkového objemu nakoupených instrumentů. Tato částka se označuje jako marže [28] a volné finanční prostředky investora se nazývají volná marže, protože se jedná o kapitál použitý k zaplacení marže při otevření nové pozice. Marže je vždy stejná pro otevření dlouhé i krátké pozice.

3.1 Fundamentální a technická analýza

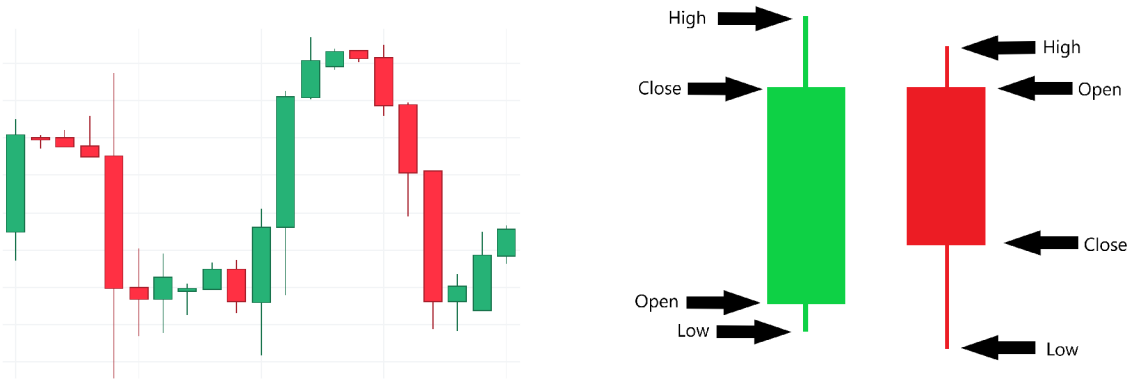
Fundamentální analýza [26] představuje studii finanční síly společnosti podle historických dat. Historická data tvoří data, která nejsou jednoznačně spjata s cenou instrumentu na trhu. Jsou to záležitosti jako řízení společnosti, konkurenční postavení v oboru, pověst společnosti, dividendová sazba, platební historie a záznamy o regulaci. Při fundamentální analýze jsou používány indikátory [18]:

- **EPS** – Indikuje poměr zisku společnosti a počtu vydaných akcií.
- **ROE** – Měří finanční výkon a poskytuje metriku pro hodnocení návratnosti investic.
- **TOAT** – Ukazuje efektivnost využití aktiv při generování hodnoty prodeje nebo příjmu společnosti.
- **P/E poměr** – Měří poměr současné tržní ceny jedné akcie vzhledem k zisku na akcii.
- **B/M poměr** – Měří poměr zaknihované hodnoty společnosti a tržní hodnoty.
- **INTE poměr** – Indikuje schopnost společnosti splácet úroky ze svých dluhů.
- **SGAE** – Představuje provozní náklady společnosti jako jsou reklama, marketingové řízení a administrativa.
- **SIZE** - Indikuje rozsah a růst firmy.

Pod technickou analýzu naopak spadají data, která jsou jednoznačně spjata s cenou instrumentu. Základem technické analýzy jsou cena a množství instrumentu, se kterým se obchodovalo za určitý časový interval. Nejdůležitějšími daty proto jsou:

- **Open** – První cena, za kterou byla vytvořena transakce v určitém intervalu.
- **High** – Nejvyšší cena, za kterou byla vytvořena transakce v určitém intervalu. Představuje bod, kdy byl rozdíl v počtu prodejních a nákupních transakcí nejvyšší ve prospěch prodejních transakcí.
- **Low** – Nejnižší cena, za kterou byla vytvořena transakce v určitém intervalu. Představuje bod, kdy byl rozdíl v počtu prodejních a nákupních transakcí nejvyšší ve prospěch nákupních transakcí.
- **Close** – Poslední cena, za kterou byla vytvořena transakce v určitém intervalu.
- **Volume** – Množství daného instrumentu, se kterým se obchodovalo za určitý interval času.
- **Bid** – Cena, za kterou se daný instrument prodá v aktuální chvíli.
- **Ask** – Cena, za kterou se daný instrument nakoupí v aktuální chvíli.

Tyto data jsou obsahem grafů, které jsou pro investory nejpoužívanější metodou jejich zobrazení. Grafy, nejčastěji svíčkové grafy, zobrazují vývoj ceny v jednotlivých časových intervalech. Svíčkový graf je tvořen z tzv. svíček, kdy každá svíčka zobrazuje 4 hodnoty v určitém časovém intervalu (Open, Close, High a Low). Tento graf je vyobrazen na obrázku 3.1(a) (získaný z webové aplikace od XTB) s popisem svíček na obrázku 3.1(b).



(a) Svíčkový graf

(b) Popis svíček

Obrázek 3.1: Svíčkový graf

Velmi důležitým indikátorem pro technickou analýzu [4] je takzvaný plovoucí průměr, který ukazuje průměrnou cenu instrumentu za předem stanovený časový interval. Tento indikátor se používá pro rozhodování, kdy otevřít dlouhou a kdy krátkou pozici nebo naopak kdy je uzavřít. Funguje na principu vytvoření imaginární linie, která vždy představuje průměrnou cenu od dané chvíle do minulosti o nastavený časový interval, viz obrázek 3.2. Pokud je tato linie překročena, tak se rozhoduje, zda pozici otevřít nebo uzavřít. Pokud je linie překročena směrem na nižší hodnotu, uzavírá se dlouhá pozice a otevírá se krátká. Při opačném jevu se naopak otevírá dlouhá a zavírá krátká pozice.

Mezi nejpoužívanější indikátory používající plovoucí průměr patří:

- **Simple Moving Average (SMA)** - Indikátor používající jednoduchý plovoucí průměr, který byl vysvětlen výše.

Obrázek 3.2: Jednoduchý plovoucí průměr za 50 dní

- **Exponential Moving Average (EMA)** - Indikátor [23] používající exponenciální plovoucí průměr, který značí využití váhy, kdy novější data mají vyšší váhu než data starší. Pro výpočet EMA je potřeba stanovit posun a časový interval, které jsou obvykle specifikovány uživatelem. Časový interval zde indikuje počet vzorků, ze kterých se bude počítat plovoucí průměr a zároveň takzvaný „vyhlazovací faktor“. Vyhlazovací faktor je použit právě pro již zmíněnou simulaci váhy. Vzorec pro výpočet vyhlazovací faktoru je velmi prostý:

$$\alpha = \frac{2}{N+1},$$

kde

  - α – vyhlazovací faktor,
  - N – časový interval neboli počet vzorků. Pokud je použita například denní vzorkovací frekvence, tak N je počet dní, ze kterých je vyhlazovací faktor počítán.

Vzorec pro výpočet EMA za pomocí vyhlazovací faktoru zní následovně:

$$E_n = \alpha \times C_n + (1 - \alpha) \times E_{n-1},$$

kde

  - n – pořadové číslo vzorku,
  - E_n – exponenciální plovoucí průměr při vzorku n,
  - C_n – uzavírací tržní cena vzorku n,
  - E_{n-1} – exponenciální plovoucí průměr při předchozím vzorku.

Pro výpočet prvního exponenciálního plovoucího průměru E_1 je použit jednoduchý plovoucí průměr se stejným časovým intervalem, který byl použit pro výpočet vyhlazovacího faktoru.

- **Moving Average Convergence/Divergence (MACD)** - K výpočtu indikátoru konvergence a divergence plovoucího průměru [4] je zapotřebí rychlý a pomalý exponenciální průměr. Základní nastavení používá 12denní průměr jako rychlý a 26denní průměr jako pomalý za předpokladu použití denní vzorkovací frekvence. Rozdílem těchto dvou plovoucích průměrů (rychlý mínus pomalý) právě vzniká již zmíněný indikátor. Ten osciluje okolo nuly, a pokud je v kladných hodnotách, tak 12denní průměr je vyšší jak 26denní, což značí nárůst poptávky oproti nabídce na trhu, tedy vzrůst ceny. Jestliže se nachází v záporných hodnotách, tak indikuje přesně opačný jev. Při rozhodování o otevírání a uzavírání pozic se používá ještě signalizační linie, která značí 9denní plovoucí průměr MACD (nejedná se o plovoucí průměr ceny instrumentu). Rozhodující je průsečík MACD a signalizační linie. Pokud MACD naroste nad signalizační linii, tak vzniká signál pro otevření dlouhých pozic. Když MACD spadne pod signalizační linii, vzniká signál pro jejich uzavření. Toto lze však opačně použít i pro krátké pozice.

SMA, EMA a MACD se řadí mezi takzvané zaostávající indikátory [4], neboť nepředvídají vývoj instrumentu do budoucna. Pouze ukazují, jak si daný instrument vede v aktuální chvíli oproti nepříliš dávné minulosti. Použití těchto indikátorů může vysoce zredukovat riziko vysokých ztrát při obchodování, avšak jejich použitím investor nevyužívá veškerý potenciál svého obchodování a může se ošidit o vyšší zisk. Protože plovoucí průměr neupozorní investora na důležité změny ve vývoji jistého instrumentu ihned, způsobuje tím pozdní uzavření a pozdní otevření pozic.

Z tohoto důvodu vznikly takzvané vedoucí indikátory [4]. Tyto indikátory nezobrazují pouze aktuální stav oproti minulosti, ale snaží se i předpovídat budoucí vývoj. K tomu se nejčastěji využívá měření, jak moc je daný instrument překoupený (představuje stav trhu, kdy je vysoký rozdíl mezi nákupními a prodejními transakcemi s nadváhou nákupních transakcí neboli převažuje poptávka nad nabídkou) nebo přeprodaný (představuje opačný jev, tzn. převažuje nabídka nad poptávkou). Následně se předpokládá, že pokud je nyní instrument „přeprodaný“, tak v budoucnu se bude cena instrumentu navracet do vyšších hodnot. Příkladem takového indikátoru je stochastický oscilátor.

Ani vedoucí indikátory samozřejmě nejsou dokonalé a mají své nedostatky. Největším nedostatkem je vysoký náběh daného instrumentu, protože kdyby vždy platilo, že „překoupený“ instrument se navrátí do nižších hodnot, tak by instrumenty dlouhodobě nerostly na ceně.

Při rozhodování o použití indikátorů, se využívají trendové kanály [20]. Trendový kanál tvoří dvě rovnoběžné přímky, které obě putují ve směru trendu. Pokud postupně stoupá tržní cena, vzniká takzvaný rostoucí trendový kanál. Pravým opakem je klesající trendový kanál, kdy tržní cena naopak klesá. Zajímavým případem je trend trhu, kdy cena neroste ani neklesá. Takový trend je indikován trendovým kanálem jdoucím do strany. Ukázka těchto kanálů je na obrázku 3.3. Jak bylo zmíněno výše, tak vedoucí indikátory nejsou velmi užitečné, pokud trh prudce klesá nebo naopak stoupá, tedy když současný trend indikuje rostoucí nebo klesající kanál. Při rostoucím nebo klesajícím kanálu je vhodný čas pro použití zaostávajících indikátorů a naopak při trendovém kanálu jdoucím do strany je ideální použití vedoucích indikátorů. Určování trendů v minulosti je z historických dat velmi snadné, avšak předpovídání do budoucna je již složitější záležitostí.

Obrázek 3.3: Ukázka trendových kanálů

3.2 Obchodní strategie

Při obchodování je také velmi důležité, jakou obchodní strategii [17] investor používá. Mezi ty nejznámější patří:

- **Poziční obchodování (Position Trading)** – Představuje strategii, kdy investor zaujímá pozici ve směru hlavního trendu instrumentu po delší dobu. Jedná se o metodu buy-and-hold (v překladu nakup-a-drž). Tato metoda neurčuje přesnou dobu držení dané pozice, může se jednat o několik měsíců či let. Poziční obchodování se také obvykle spojuje s obchodními strategiemi, jako jsou obchodování podle trendů, obchodování proti trendům a obchodování s průlomy.
  - Velkou výhodou této strategie je, že investor nemusí stále sledovat denní změny v ceně, ale zajímají ho pouze dlouhodobé změny.
  - Mezi nevýhody patří dlouhodobá vázanost peněz k dané pozici a krátkodobá nenávratnost této pozice. Zisk se může objevit až po několika letech, ale je tu také možnost, že se neobjeví nikdy, a pozice navyšuje s časem pouze větší ztráty.
- **Swingové obchodování (Swing Trading)** – Jedná se o velmi podobný přístup jako je pozičního obchodování, avšak s rozdílem, že swingové obchodování se soustředí jak na krátkou, tak dlouhou pozici určitého instrumentu. Správným a včasným střídáním těchto pozic je možno dosáhnout vyšších a rychlejších zisků než u pozičního obchodování. Obvykle se tyto pozice drží po dobu několika dní.
  - Největší výhodou je možnost využít růst i pokles ceny v prospěch investora. Při správném použití investor v obou těchto případech dosahuje zisku.
  - Naopak nevýhodou je, že investor musí neustále sledovat a předvídat krátkodobé změny a měnit svou pozici daného instrumentu. Při obchodování s více instrumenty je téměř nemožné stihnout sledovat a předpovídat veškeré změny, tudíž je velmi časté, že investor nevyužije každou šanci na dosažení zisku.

- **Denní obchodování (Day Trading)** – Velmi oblíbeným způsobem obchodování je právě denní obchodování. Jak již z názvu vyplývá, jedná se o jednodenní obchodování. To však neznamená, že investor pouze otevře pozici na začátku dne a uzavře ji až na konci dne. Většina investorů během dne otevře a uzavře mnoho různých pozic, ale vždy na konci dne skončí tím, že všechny otevřené pozice za tento den uzavře. Takto jeho pozice neovlivní další změny, které se mohou stát v následujících dnech, ale pouze změny, které se dějí během daného dne.
  - Výhodou tohoto přístupu je, že investor není vystaven rizikům změny v cenách, které nestihne zaznamenat, a tudíž eliminuje velké riziko ztrát. Také investor na konci každého dne může přesně vyčíslit svůj zisk či ztrátu, což u jiných přístupů zcela možné není.
  - Mezi nevýhody patří neustálá potřeba sledovat změny v cenách a trendech, což může být pro některé investory příliš náročné a stresující. Rovněž velký počet otevřených transakcí každý den může vést k větším nákladům na poplatky či provizi brokerům.
- **Obchodování na základě cenové akce (Price Action Trading)** – Pro začínající investory se může jevit složitějším. Představuje přístup, kdy investor sleduje změny v ceně na určitém časovém rámci a podle toho se rozhoduje, kterou cenovou akci provede.
  - Výhodou tohoto přístupu je možnost pouze sledovat různé grafy a hledat různé vzory, na základě kterých následně otevírat pozice. Investor nemusí tudíž využívat žádné funkce či složité výpočty pro rozhodování.
  - Složitost hledání vzorů v grafech je považováno za nevýhodu tohoto přístupu, jelikož je k tomu zapotřebí určitých znalostí a zkušeností.
- **Obchodování na základě událostí (News Trading)** – Jedná se o postup, u kterého investor nesleduje změnu ceny, ale sleduje fundamentální události ve světě. Protože některé z těchto událostí mají vliv na finanční trh, lze jejich sledováním předpovědět poklesy a růsty v ceně instrumentu. Jedná se o jednu z hlavních strategií používající fundamentální analýzu dat.
  - Výhodou je, že investor může dosahovat vysokých zisků, pokud dokáže odhadnout správný dopad fundamentálních událostí na finanční trh.
  - Nevýhoda tohoto postupu spočívá ve vyšší míře rizika, které investor podstupuje. Finanční trh se může ve vztahu k některým událostem chovat nepředvídatelně a cena se může v následujících dnech neustále různě navyšovat či klesat. Investor v takové situaci nemůže jednoznačně určit, jak se cena bude nadále vyvíjet.

Kapitola 4

Aplikační programové rozhraní

Aplikační programové rozhraní neboli API (Application Programming Interface) [5] umožňuje softwarovým programům komunikovat s jinými softwarovými programy. API nabízí souhrn příkazů a pravidel pro ovládání specifického softwaru. Samotné API musí definovat metodu připojení a následnou metodu přenosu dat. Pro připojení k API se používají takzvané APIkeys, které je možné získat od společnosti poskytující dané API. Avšak tyto klíče jsou u některých společností zpoplatněny.

4.1 Typy přenosu dat

4.1.1 Formát XML

Nejpoužívanějším formátem reprezentace přenášených dat je v dnešní době XML[27]. Jedná se o strukturovaný dokumentový formát, který obsahuje textová data. Závisí na přiřazení dat ke štítku. Používá stromovou strukturu dat, pomocí čehož docílí snadného seskupování dat do skupin. Štítek je tvořen z otevírací části, těla a zavírací části, kdy název štítku se nachází v otevírací i zavírací části, viz výpis 4.1. Nevýhodou XML formátu je vysoký objem přenesených dat, i když samotná data zprávy mají velikost znatelně menší.

```
<Airport City="New York">
    <Arrival-time>10:07:12</Arrival-time>
    <Departure-time>10:42:53</Departure-time>
    <Date>2023-12-31</Date>
    <Gate>12</Gate>
    <Gate>8</Gate>
</AirPort>
```

Výpis 4.1: Příklad XML dat

Ve výše uvedeném příkladu byl počet přenesených dat 169 bytů, i když data samotná mají pouhých 55 bytů. Největším rozdílem oproti jiným možným reprezentacím je přiřazení atributů ke štítkům, pomocí kterých se následně přijatá data mohou filtrovat či vyhledávat. Pomocí atributů je možno přidat ke štítkům dodatečné informace.

4.1.2 Formát JSON

Další používanou možností reprezentace přenesených dat je formát JSON (JavaScript Object Notation) [27]. Jedná se opět o strukturovaný dokumentový formát, který obsahuje

textová data. JSON taktéž používá stromovou strukturu, ale na rozdíl od XML používá klíče na místo štítků. Klíče neumožňují použití atributů, tudíž veškeré klíče v podstromu musí být unikátní. Oproti formátu XML tolik nenavyšuje velikost přenesených dat. Ukázka viz výpis 4.2.

```
{ "Airport":
    { "_City": "New York",
      "Arrival-time": "10:07:12",
      "Departure-time": "10:42:53",
      "Date": "2023-12-31",
      "Gate": [ "12",
                "8"
              ]
    }
}
```

Výpis 4.2: Příklad JSON dat

Při použití JSON formátu je stejná zpráva uložena na 154 bytech, což oproti 169 bytům není velký rozdíl, avšak u větších zpráv je již tento rozdíl znatelnější.

4.2 Typy API

API jsou v dnešní době používána ve velké míře a existuje jich několik typů. Mezi ty nejvyužívanější typy patří RESTful, RPC, SOAP a HATEOAS [6].

4.2.1 REST API

REST (Representational State Transfer) [21] [6] [8] je architektonický styl, který definuje sadu pravidel a omezení. RESTful API je takové API, které vyhovuje všem těmto pravidlům a omezením a které pro komunikaci využívá optimálně HTTP požadavky. Hlavními pravidly jsou:

- **Reprezentace zdroje** – Každý zdroj je unikátně identifikován pomocí svého URI (Uniform Resource Identifier) a je přístupný pomocí standardních metod HTTP.
- **Bezstavová komunikace** – Každý požadavek od klienta na server obsahuje veškeré informace potřebné pro vykonání požadavku, takže server si nijak neukládá stav obsluhy uživatele a pouze obsluhuje jednotlivé požadavky.
- **Operace** – Používají se přesně definované operace. Nejčastějšími jsou:
  - GET – získávání zdroje či zdrojů,
  - POST – vytváření nového zdroje či zdrojů,
  - PUT – aktualizace zdroje,
  - PATCH – částečná aktualizace zdroje,
  - DELETE – odstranění zdroje.
- **Návratové kódy** – Pro zjednodušení komunikace používá návratové kódy HTTP.
- **Reprezentace dat** – Data jsou reprezentována ve stanoveném formátu. Nejčastěji se jedná o JSON nebo XML.

4.2.2 RPC

Komunikační model RPC (Remote Procedure Call) [8] spočívá ve vzdáleném volání procedur. RPC model také skrývá detaily komunikace mezi klientem a serverem. Často bývá kombinován s REST architektonickým stylem a vznikají takzvané hybridní API.

4.2.3 SOAP API

SOAP (Simple Object Access Protocol) [21] je protokol pro výměnu strukturovaných informací v distribuovaných a decentralizovaných prostředích. Klíčovými aspekty jsou:

- Použití XML formátu dat.
- Komunikace může probíhat pomocí HTTP, HTTPS, TCP, UDP nebo i SMTP. Nejčastěji se používá HTTP nebo HTTPS.
- Zprávy mají přesně danou strukturu, která obsahuje 3 části:
  - Záhlaví – obsahuje volitelná data (například Content-Type a Content-Length),
  - Tělo – obsahuje data zprávy ve formátu XML,
  - Obálka – charakterizuje SOAP zprávu, určuje její začátek a konec.
- Jazyková a platformní nezávislost.
- Definuje své vlastní zabezpečení v podobě WS-Security.
- Může implementovat princip vzdáleného volání procedur (RPC).

4.2.4 HATEOAS

HATEOAS (Hypermedia As The Engine Of Application State) [10] je v podstatě nadstavba nad architekturou REST. HATEOAS API vyhovuje všem pravidlům a omezením stanoveným REST architekturou, avšak navíc přidává dynamickou navigaci ke zdrojům. HTTP odpovědi obsahují navíc ještě odkazy (hypermedia) na dodatečné zdroje, které jsou získávány pomocí vzájemného propojení všech zdrojů. Toto umožňuje uživateli pracovat s API bez jakýchkoliv předchozích znalostí o daném API.

```
"departmentId": 10,
"departmentName": "Administration",
"locationId": 1700,
"links": [
    {
        "href": "10/employees",
        "rel": "employees",
        "type" : "GET"
    }
]
```

Výpis 4.3: Příklad dat s odkazy [10]

Kapitola 5

XTB API – xAPI

X OpenHub API (xAPI) je programové rozhraní použitelné pro komunikaci s rozhraním společnosti XTB pro obchodování na finančních trzích. Toto API, specificky REST API, používá výše zmíněný JSON formát dat pro komunikaci mezi uživatelem naprogramovanou aplikací a XTB serverem. Pro připojení se používá takzvané „čisté síťové připojení“, ale pro následné obchodování se již používá SSL protokol, který zajišťuje bezpečnou komunikaci. xAPI se může připojit ke dvěma různým serverům:

- **DEMO** – Komunikuje na portu 5124 a 5125 pro streamovou komunikaci.
- **REAL** – Komunikuje na portu 5112 a 5113 pro streamovou komunikaci.

Celá tato kapitola byla převzata z [5].

5.1 Formát dat

Data při komunikaci s xAPI jsou ve formátu JSON, jak je již zmíněno výše. Avšak tento formát musí mít předem stanovenou strukturu, aby xAPI následně tyto data dokázalo zpracovat.

Při odesílání požadavku na server musí data obsahovat klíče „command“ a „arguments“, kde hodnota přiřazená ke klíči „arguments“ je obvykle dále rozvětvená na více klíčů s hodnotami. Viz výpis 5.1. Hodnota u klíče „prettyPrint“ specifikuje, zda se obsah zprávy má vypsat na standardní výstup v takzvané „člověkem čitelné“ podobě. Tato hodnota může být však u požadavku vynechána.

```
{
        "command": "commandName",
        "arguments": {
                "arg1Name": 10,
                "arg2Name": "Some text",
                ...
        },
        "prettyPrint": true
}
```

Výpis 5.1: Formát odesílaných dat

Při přijetí odpovědi ze serveru data vždy obsahují klíč „status“ s hodnotou, která určuje, zda byl požadavek přijat, a následně obsahuje data odpovědi, která jsou variabilní v závislosti na typu požadavku. Viz výpis 5.2.

```
{
        "status": true,
        "returnData": JSON value
}
```

Výpis 5.2: Formát přijatých dat

V případě, že status obsahuje hodnotu „false“, tak při vykonání požadavku nastala chyba a tato chyba je následně identifikována podle hodnoty u klíče „errorCode“ a popsána v hodnotě u klíče „errorDescr“[1]. Viz výpis 5.3.

```
{
        "status": false,
        "errorCode": "E123",
        "errorDescr": "Error description"
}
```

Výpis 5.3: Formát přijatých dat s chybou

5.2 Wrapper od XTB

XTB taktéž nabízí různé „wrappery“, které obsahují již předem naprogramované funkce pro ovládání výše zmíněného xAPI. Jsou dostupné pro programovací jazyky Java, Python a C#. Tato práce se bude soustředit na „wrapper“ v jazyce Java.

5.2.1 Tvorba požadavku

Pro tvorbu požadavku se zde nachází třída `APICommandFactory`, jenž nabízí metody pro vytvoření požadavku pro kterýkoli příkaz, jaký xAPI dokáže zpracovat. `APICommandFactory` pouze vytvoří předem předdefinovanou kostru příkazu v JSON formátu, kterou umožňuje pomocí argumentů funkce naplnit požadovanými daty. Pro komunikaci je také zapotřebí instance třídy `SyncAPIConnector` (dále jen konektor), která umožňuje zabezpečenou komunikaci se serverem.

5.2.2 Důležité příkazy

Příkazy jsou rozděleny na **synchronní** a **asynchronní (streamové)**. Synchronní příkazy ve vrácené odpovědi od serveru obsahují veškerá data připravená pro další použití. Speciální variantou synchronních příkazů jsou příkazy `Login` a `Logout`. Tyto příkazy nevrací žádná data pro další použití, pouze změní stav konektoru z neautorizovaného na autorizovaný nebo naopak. Toto umožňuje vykonávat příkazy pod určitým obchodním účtem vytvořeným u XTB. Mezi další synchronní příkazy patří:

[1] Více informací o chybových kódech a zprávách viz http://developers.xstore.pro/documentation/#error-messages.

- `getAllSymbols` – Získání dat o všech dostupných symbolech, se kterými je možno obchodovat na platformě XTB. Odpověď obsahuje velký objem dat, takže by se tento příkaz měl používat v minimálním měřítku a pro získání pouze malé skupiny symbolů je lepší používat vícenásobně příkaz `getSymbol` (například pro 1 až 10 symbolů).
- `getSymbol` – Získání dat o konkrétním instrumentu. Při vytváření tohoto příkazu je potřeba jako argument zadat pouze zkratku instrumentu (například AAPL.US).
- `getChartLastRequest` – Získání svíčkových dat za poslední období. Období je určeno pomocí argumentem specifikovaného času od až po nynější dobu. Pro získání svíčkových dat za určitý úsek času (specifikovaný časem od - do) je možno použít příkaz `getChartRangeRequest`.
- `getCurrentUserData` – Získání dat o aktuálně přihlášeném uživateli. Tento příkaz je možné použít například pro získání měny, ve které je účet veden.
- `getMarginLevel` – Získání dat ohledně finanční prostředků uživatele, včetně použitelných prostředků, celkové hodnoty účtu a úrovně marže.
- `getMarginTrade` – Získání přesné hodnoty marže k vytvoření transakce pro daný instrument a množství.
- `getTradeRecords` – Získání dat o specifických transakcích, kdy tyto transakce mohou být aktuální či historické.
- `getTrades` – Získání dat o všech nebo pouze aktuálně otevřených transakcích. Při vytváření tohoto příkazu se musí specifikovat povinný argument `openned0nly`, který určuje, zda chceme získat pouze aktuálně otevřené transakce.
- `getTradesHistory` – Získání dat o historických transakcích za určité období. Tento příkaz pouze zjednodušuje funkcionalitu předchozího příkazu a umožňuje filtrovat data podle intervalu času.
- `getTradingHours` – Získání dat o času, kdy je možno se specifickým instrumentem obchodovat. Tato data se nenachází v odpovědi žádného jiného příkazu, proto je nutno před vytvořením jakékoliv transakce zkontrolovat, zda je s tímto instrumentem v danou chvíli možno obchodovat pomocí tohoto příkazu.
- `tradeTransaction` – Příkaz pro vytvoření transakce. Umožňuje vytvořit transakci pro dlouhou či krátkou pozici. Při vytváření transakce lze také specifikovat **Stop loss** a **Take profit**.
- `tradeTransactionStatus` – Pro získání stavu určité transakce. Tento příkaz existuje, protože příkaz `tradeTransaction` ve své odpovědi nesděluje stav dané transakce.

Druhou skupinou jsou **asynchroní (streamové)** příkazy. Funkčnost streamových příkazů je velmi odlišná od těch synchronních, protože server neposílá na tyto příkazy žádnou odpověď. Data jsou po přihlášení k odběru daného tématu získána pomocí asynchronně běžícího vlákna, které je vytvářeno při vytvoření konektoru a připojeno na specifický `streamSessionId`, což je číslo unikátně identifikující streamovou komunikaci pro daného přihlášeného uživatele. Server následně vysílá veškerou streamovou komunikaci na portu 5113 (port 5125 pro DEMO server) a přihlášený uživatel má autorizaci přistupovat k datům,

která jsou identifikována pomocí uživateli přiřazeného `streamSessionId`. Příkazy vždy mají svoji opačnou verzi, která naopak odběr daného tématu odhlásí. Wrapper od XTB nabízí třídu `StreamingListener`, jejíž instance monitoruje streamovou komunikaci a obsahuje metody pro obsluhu přijatých streamových dat. Tyto metody originálně neobsahují žádnou implementaci a vývojář používající tento wrapper musí jejich implementaci doplnit. XTB taktéž doporučuje používání streamových příkazů pro získávání aktualizovaných dat. Mezi důležité streamové příkazy patří:

- `getBallance` – Přihlášení se k odběru dat ohledně finančních prostředků na účtu přihlášeného uživatele. Data jsou ze serveru zasílána při kterékoliv změně v těchto datech.
- `getCandles` – Přihlášení se k odběru svíčkových dat specifického instrumentu. Interval příchodu každých svíčkových dat je 1 minuta.
- `getTickPrices` – Přihlášení se k odběru dat o změně v ceně specifického instrumentu. Při přihlášení lze specifikovat minimální dobu mezi každými dvěma odpověďmi.
- `getTrades` – Přihlášení se k odběru dat ohledně změny v otevřených transakcích. Může se jednat například o uzavření transakce či změnu **Stop loss** nebo **Take profit**.
- `getTradeStatus` – Přihlášení se k odběru dat ohledně změny stavu otevřených transakcí.

5.2.3 Příklady použití příkazů s využitím wrapperu

Pro používání API od XTB není potřeba získávat žádný APIkey, ale uživatel se musí přihlásit ke svému účtu za pomocí identifikačního čísla účtu, hesla a zvolení serveru. Z tohoto důvodu je používání API zcela bezplatné.

Po přihlášení jsou programu přiřazeny porty pro synchronní a streamovou komunikaci a identifikační číslo sezení. Tyto údaje jsou uloženy v konektoru, proto je nezbytný při vytváření každého požadavku. Použití příkazu pro přihlášení je vyobrazeno na výpisu 5.4.

```
public boolean login(ServerEnum server, Credentials credentials) {
    try {
        connector = new SyncAPIConnector(server); //Zvoleni serveru
        //Prihlaseni pomoci ID a hesla "credentials"
        APICommandFactory.executeLoginCommand(connector, credentials);
        return true;
    } catch (Exception e) { //Vyjimka pokud udaje nebyly prijaty
        return false; //Neni jiny zpusob jak otestovat spravnost udaju
    }
}
```

Výpis 5.4: Příklad metody login

Po odhlášení by se ještě měla odpojit streamová komunikace, protože k ní již není přístup. Viz výpis 5.5.

```
public void logout() {
    try {
        APICommandFactory.executeLogoutCommand(connector);
```

```
            connector.disconnectStream();
        } catch (Exception e) {
            exit(-1); //Selhani pripojeni k~serveru
        }
    }
```

Výpis 5.5: Příklad metody logout

Další velmi užitečnou metodou je metoda implementující použití příkazu pro získání údajů o uživateli. Součástí těchto údajů je například měna účtu. Implementace této metody lze vidět níže na výpisu 5.6.

```
public CurrentUserDataResponse getUserInfo() {
    try {
        return APICommandFactory
            .executeCurrentUserDataCommand(connector);
    } catch (Exception e) {
        return null;
    }
}
```

Výpis 5.6: Příklad metody getUserInfo

5.3 Výhody a nevýhody

Při studování funkcionality xAPI jsem narazil na některé okolnosti, které mi usnadnili nebo naopak ztížili a znepříjemnili práci. Proto zde uvádím mnou zjištěné výhody a nevýhody.

Výhody:

- Vysoký počet HTTP požadavků za sekundu.
- Čitelná a přesná dokumentace.
- Při vytvoření prodejní transakce xAPI automaticky uzavírá nejdéle držené pozice, což velmi zjednodušuje implementaci některých AOS.

Nevýhody:

- Ikdyž možnost obchodovat s frakčními akciemi podporuje webová aplikace xStation, tak xAPI tuto možnost nepodporuje. Frakční akcie je část akcie, která je menší než jedna celá akcie.
- Pro otevření transakce je nutné zadat množství instrumentu, přičemž získaná cena z xAPI je vždy v měně instrumentu a pro převod do měny účtu je zapotřebí použít nástroj třetí strany.
- Při získávání měny instrumentu, který je zařazen pod kryptoměny, se vždy od xAPI získá značka kryptoměny (například BTC pro bitcoin), ikdyž cena kryptoměny je ve většině případů v amerických dolarech.
- XTB nemá uživatelskou podporu pro xAPI.

Kapitola 6

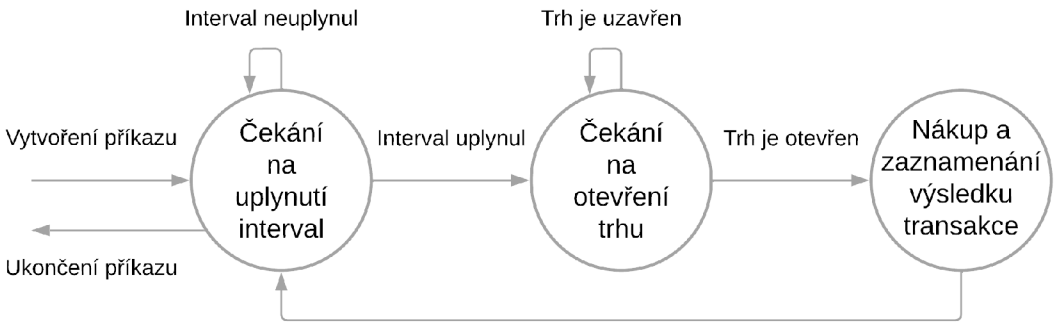
Návrh

V této kapitole je dopodrobna popsán návrh celého navrženého systému a jednotlivých AOS. V rámci návrhu jsou zmíněny použité technologie a zamýšlená funkcionalita systému jako celku.

6.1 Navržené AOS

Všechny navržené AOS se liší v zásadních nebo pouze okrajových aspektech. Navrženými systémy jsou:

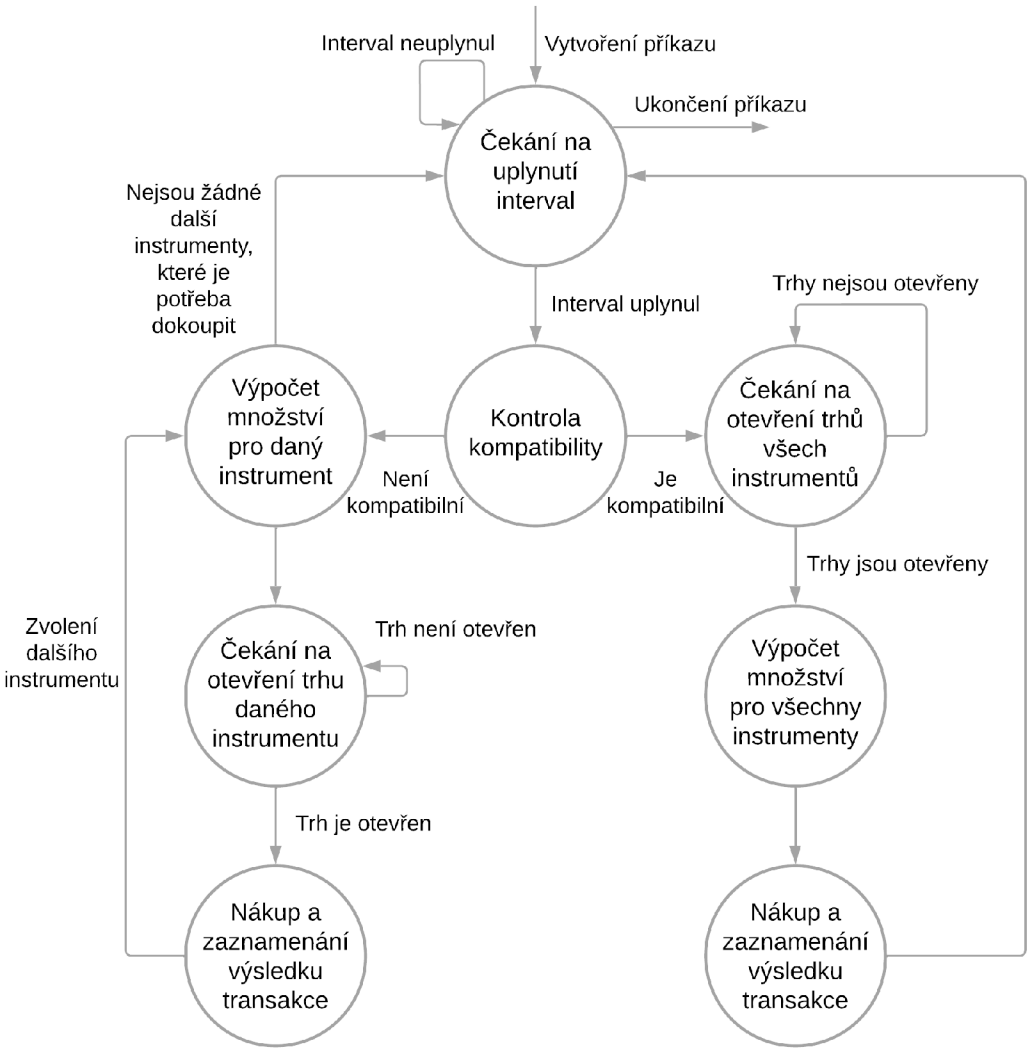
- **Automatické nakupování instrumentu** – AOS je navržen tak, aby nakupoval v předem stanovených intervalech a s předem stanovenou finanční částkou pro každou investici. Také je možné buď stanovit ukončující datum investování, kdy se příkaz uzavře a už se nebude nadále vykonávat, nebo příkaz ukončit ručně. Účelem tohoto AOS je postupné a rovnoměrné investování se strategií dlouhodobého držení investic. Jednoduchý algoritmus lze vidět na obrázku 6.1.



Obrázek 6.1: Schéma funkčnosti automatického nakupování

- **Automatické vyvažování portfolia bez prodávání** – tento AOS se velmi podobá předchozímu, avšak rozdíl tkví ve výpočtu množství, které se bude nakupovat. Tento výpočet bude záviset na procentuálním rozložení portfolia, které stanoví sám uživatel. Následně se při každém nákupu bude snažit dokupovat instrumenty tak, aby procentuální rozložení portfolia odpovídalo. Zde však vzniká problém s kompatibilitou těchto

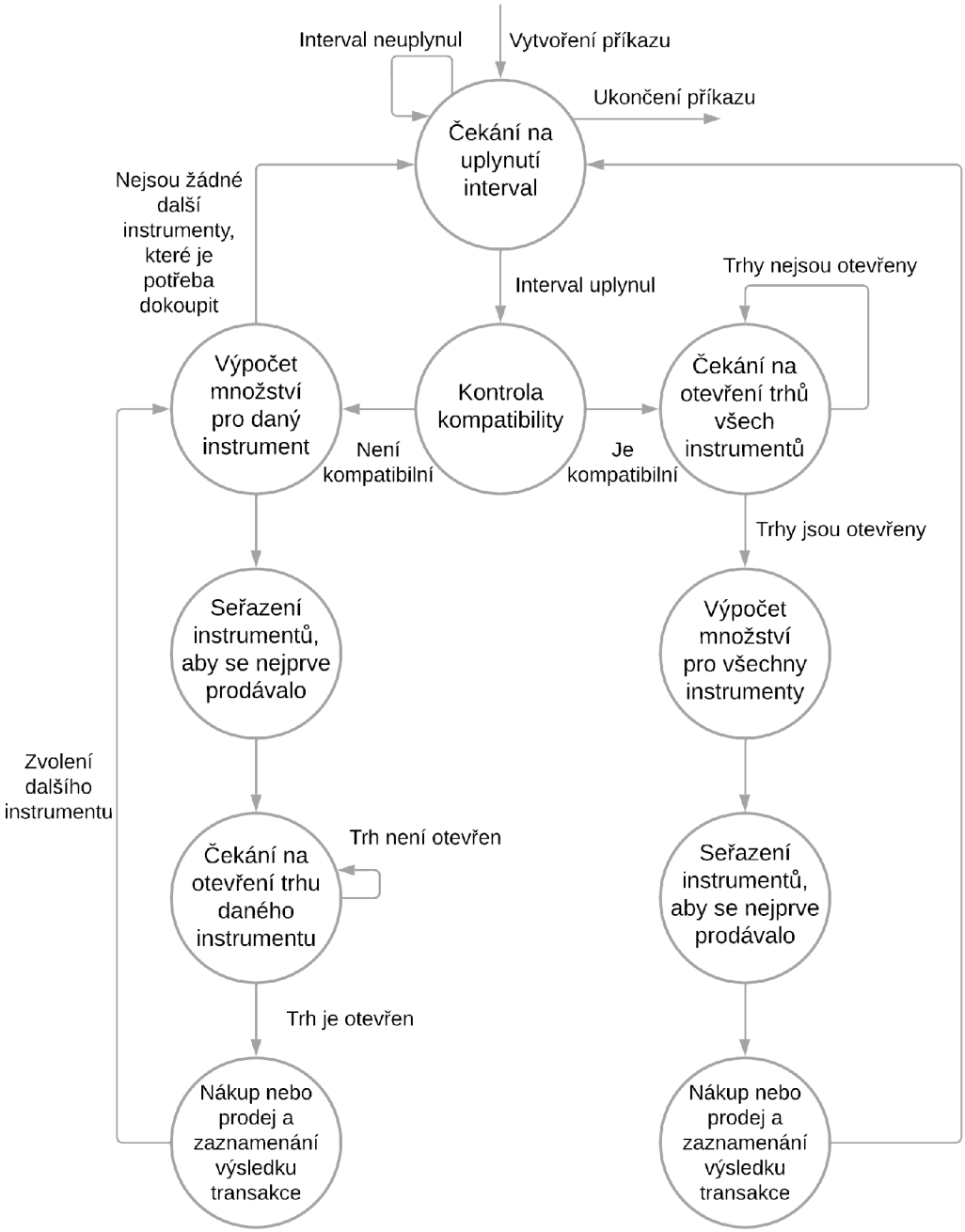
instrumentů, protože pokud instrumenty nelze všechny nakupovat ve stejnou chvíli, musí se množství vypočítávat postupně a dokupovat v různých časech. Kompatibilita bude zjištěna při vytváření příkazu. Účelem tohoto AOS je dokupovat ty instrumenty, které klesají na hodnotě, a držet ty, které na hodnotě nabývají. Tento komplexnější algoritmus je vyobrazen na obrázku 6.2.



Obrázek 6.2: Schéma funkčnosti automatického vyvažování portfolia bez prodávání

- **Automatické vyvažování portfolia s prodáváním** – další AOS v pořadí je ve svém základu stejný jako předchozí, jen je zde přidána možnost prodat instrumenty, které rostou na hodnotě, namísto jejich držení. Prodávání však bude probíhat pouze u instrumentů držených déle jak 3 roky, a to z důvodu časového testu. Tříletý časový test implikuje, že příjem z instrumentu, specificky z akcie, drženého více jak 3 roky je osvobozen od daně z příjmu [34]. Další novou funkcí u tohoto systému je seřazení instrumentů tak, aby se nejprve prodávalo, a až následně nakupovalo. Důvodem vytvoření této funkce je potřeba využití peněz získaných z prodávání. Účelem tohoto

AOS je prodávat instrumenty, které rostou na hodnotě, a za získané finanční prostředky dokupovat instrumenty, které na hodnotě naopak klesají. Algoritmus tohoto systému je vyobrazen níže na obrázku 6.3.



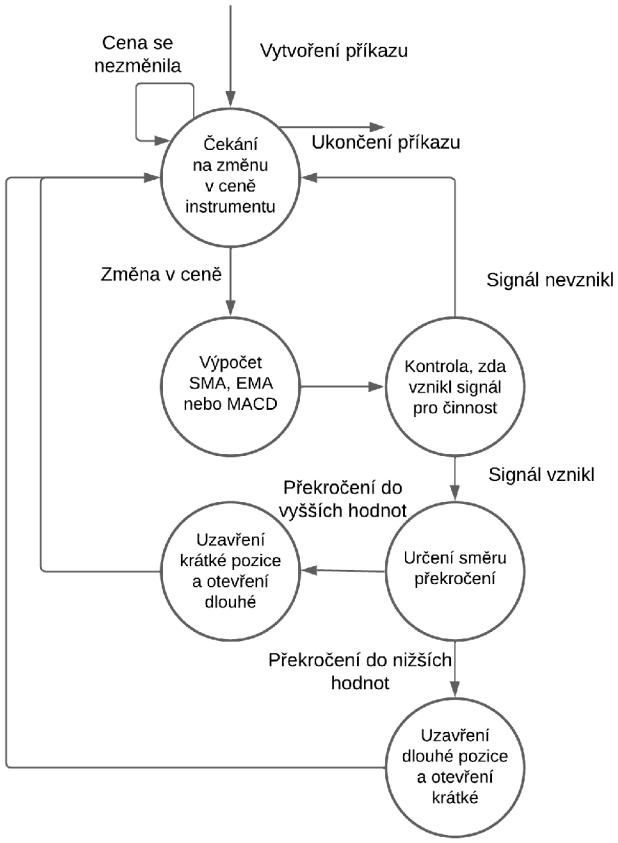
Obrázek 6.3: Schéma funkčnosti automatického vyvažování portfolia s prodáváním

- **Automatické obchodování na základě plovoucích průměrů** – jak již bylo zmíněno v podkapitole 3.1, tak pro rozhodování mohou AOS používat různé indikátory. Jelikož zatím nebyl zmíněn jediný AOS, který indikátory pro rozhodování používá,

navrhl jsem AOS obchodující na základě plovoucích průměrů. Ten specificky používá pro rozhodování indikátory s plovoucím průměrem jako jsou SMA, EMA a MACD. Zvolení jednoho z nich již bude na uživateli. Kromě zvolení indikátoru bude uživatel muset doplnit dodatečná nastavení, která zahrnují:

- Ukončovací datum příkazu.
- Počet vzorků pro výpočet SMA a EMA. MACD má předem stanovený počet vzorků.
- Časový interval mezi jednotlivými vzorky.

Hlavním rozdílem oproti předchozím AOS je v tomto případě obchodování s veškerými finančními prostředky účtu. Je tedy na uživateli, jestli bude chtít v průběhu běhu AOS na účet přidávat či odebírat finanční prostředky. Dalším rozdílem je nenastavování opakovacího intervalu, protože AOS bude sledovat pohyb ceny instrumentu konstantně a při každé změně se rozhodovat, zda otevřít či uzavřít krátkou a dlouhou pozici. Algoritmus sice vypadá jednodušeji oproti předchozím, ale AOS považuji za nejkomplexnější z důvodu použití indikátorů, jejichž výpočet je mnohem složitější než výpočty u předchozích AOS, viz obrázek 6.4.



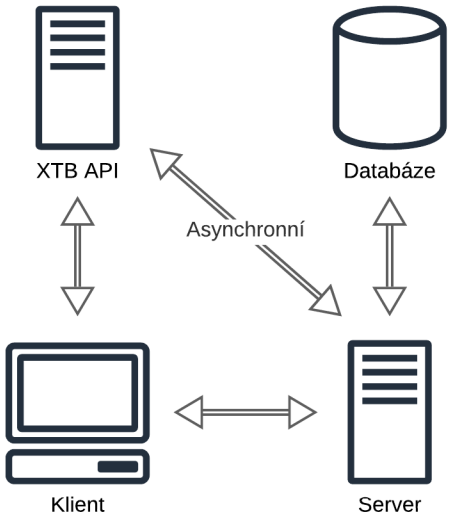
Obrázek 6.4: Schéma funkčnosti automatického obchodování na základě plovoucích průměrů

6.2 Architektura systému

Návrh systému se skládá ze 3 částí:

- uživatelská aplikace – implementovaná v jazyce Java za použití knihovny JavaFX,
- server – implementován v jazyce Java s použitím konstrukce (framework) Spring Boot a Java Persistence API (JPA) pro správu databáze,
- MySQL Databáze.

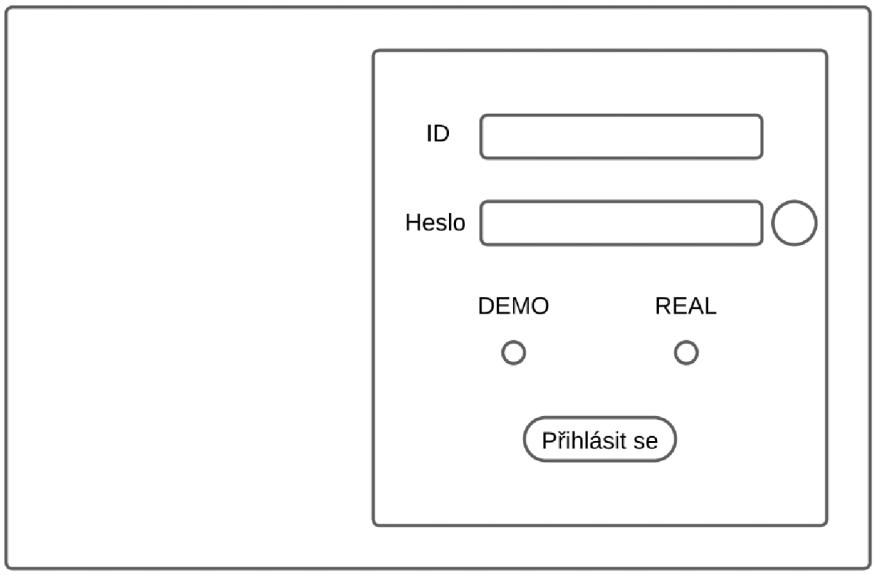
Uživatelská aplikace umožňuje komunikaci s xAPI pro potřeby uživatele a komunikaci se serverem za použití HTTP požadavků. Server využívá xAPI asynchronně vůči uživateli a má zároveň přístup k MySQL databázi s veškerými důležitými daty, viz obrázek 6.5.



Obrázek 6.5: Návrh systému

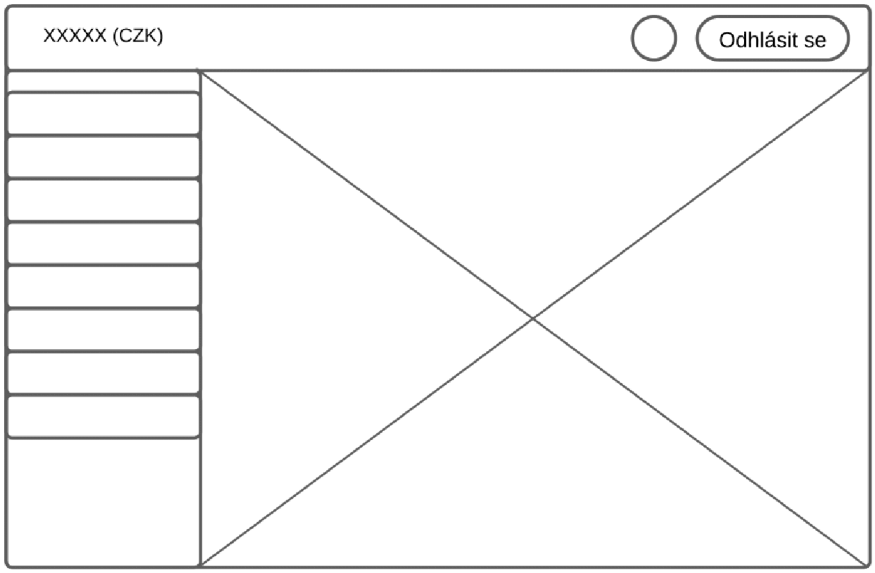
6.3 Grafické uživatelské rozhraní (Front end)

Aplikace je navržena tak, aby byla uživatelsky přívětivá a umožňovala použití většiny funkcí podporovaných aplikačním programovým rozhraním xAPI. Pro práci s aplikací uživatel musí nejprve vlastnit demo nebo reálný účet u brokera XTB. S tímto účtem se následně pomocí identifikačního čísla a hesla bude moci do aplikace přihlásit, proto úvodní obrazovkou po spuštění aplikace je dialog pro přihlášení, viz obrázek 6.6. Toto dialogové okno obsahuje textová pole pro zadání identifikačního čísla a hesla, tlačítko pro zobrazení hesla, přepínače pro výběr režimu (demo nebo reálný účet) a tlačítko pro přihlášení.



Obrázek 6.6: Návrh přihlašovacího okna

Po přihlášení se uživateli zobrazí domovské okno, jejíž součástí je postranní navigační menu a horní menu se zobrazenou hodnotou představující zůstatek finančních prostředků uživatele a tlačítka pro odhlášení a zobrazení notifikací, viz obrázek 6.7.

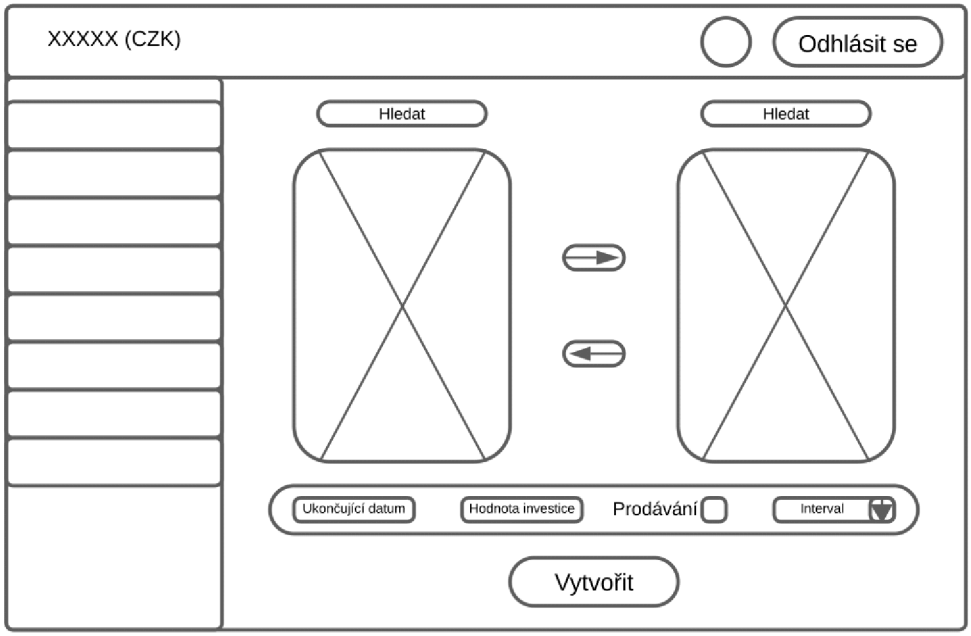


Obrázek 6.7: Návrh domovského okna

Uživatel se následně v aplikaci naviguje pomocí již zmíněného postranního navigačního menu a zobrazuje obsah do vytyčené oblasti vpravo, viz obrázek 6.7.

Podle návrhu aplikace se serverem komunikuje pomocí HTTP požadavků a metod GET, POST, PUT, PATCH a DELETE. Samotné ovládání AOS se dělí na vytváření automatického příkazu a jeho následné zobrazení s možností úpravy.

Jako příklad uvádím návrh okna pro vytvoření portfolia a nastavení AOS, které automaticky vyvažuje instrumenty v portfoliu, viz obrázek 6.8. Navržené okno obsahuje dvě tabulky, kde jedna z nich obsahuje výpis všech instrumentů a druhá představuje nově tvořené portfolio. Instrumenty se mezi tabulkami přesouvají pomocí šipek umístěných mezi nimi a po přesunutí bude uživatel muset zvolit procentuální rozložení. Uživateli se bude zobrazovat, kolik procent již má rozděleno a kolik mu ještě zbývá rozdělit. Po vytvoření portfolia je potřeba doplnit dodatečné informace, jejichž součástí je ukončující datum, finanční prostředky, které budou při každém vykonání příkazu investovány, interval opakování a zda se má využívat AOS s prodejem či nikoliv. Okno také obsahuje možnost vyhledávat v tabulkách pro snadnější a přívětivější použití.



Obrázek 6.8: Návrh okna pro vytváření nového příkazu pro vyvažování portfolia

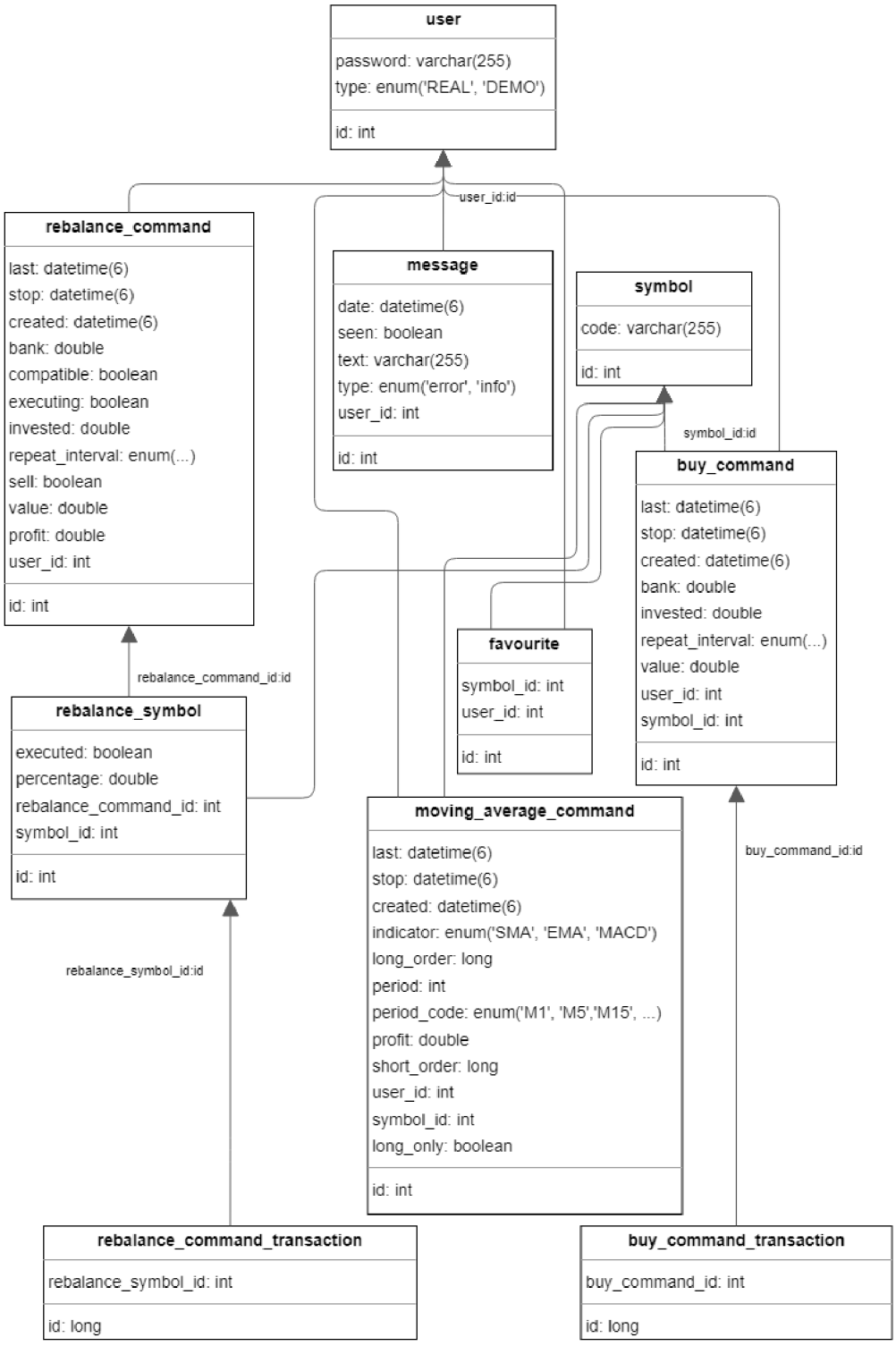
6.4 Server (Back end)

Návrh serveru je velmi stručný, jelikož jeho implementace zahrnuje veškerá již zmíněná AOS. Back end bude implementován s využitím Java Spring Boot frameworku, technologií JPA pro správu databáze a JSON Web Token (dále jen JWT) pro zabezpečení. Server také obsahuje implementaci REST API pro komunikaci s front endem neboli grafickým uživatelským rozhraním.

- **Java Spring Boot** – jedná se o open-source framework, který zjednodušuje a urychluje vývoj Java aplikací. Nejdůležitějšími technologiemi pro tuto práci, které framework přidává, jsou Spring Data JPA a Spring Data REST. Více viz Spring Boot dokumentace [24].
- **JPA** – jedná se o API, které umožňuje přístup k persistentním datům neboli databázi. Definuje interní mapování objektů na relační databáze. Za pomocí JPA může back end zacházet s databází jako s kolekcí objektů. Více viz [13].
- **JWT** – představuje formu identifikace a autentizace uživatelů pomocí tokenů, které jsou zakódované a po dekódovaní z nich vznikají data ve formátu JSON obsahující identifikující informace o uživateli. Každý token je také podepsán kryptografickým hešem za použití algoritmů HMAC a SHA256. Token je serverem přiřazen uživateli vždy po jeho přihlášení do daného systému a je následně zasílán při každém HTTP požadavku. Více viz [19].

6.5 Databáze

Databáze je navržena tak, aby obsahovala veškeré důležité informace ohledně jednotlivých záznamů (příkazů) pro AOS a uživatelů. Implementována je pomocí Spring Data JPA, jak již bylo zmíněno výše. Hlavní tabulkou v databázi je tabulka uživatele (user), na kterou jsou navázány všechny zbývající tabulky. Uživatel má své seznamy záznamů jednotlivých AOS (rebalance_command, buy_command, moving_average_command), seznam notifikací (messages) a seznam oblíbených instrumentů (favourite). Protože příkaz pro nakupování (tabulka buy_command) se vždy vztahuje pouze na jeden instrument, tak zde vzniká vazba na tabulku „symbol“ a cizí klíč se ukládá v tabulce „buy_command“. Stejný princip platí i u tabulky „moving_average_command“. Oproti tomu v případě příkazu pro vyvažování portfolia vzniká vazba na tabulku „rebalance_symbol“, která obsahuje cizí klíč, protože těchto symbolů musí jednotlivé příkazy obsahovat více. Zároveň každý záznam v tabulce „rebalance_symbol“ obsahuje seznam transakcí, které byly pro tento symbol ve vztahu s daným příkazem vytvořeny, aby bylo možné následně zjistit stav a zisk portfolia. Ze stejného důvodu si seznam transakcí udržuje také každý záznam v tabulce „buy_command“. Viz obrázek 6.9.



Obrázek 6.9: Diagram schéma databáze

Kapitola 7

Implementace

Tato kapitola obsahuje důkladný popis implementace důležitých částí, úryvky a popis kódu. Implementace se skládá pouze ze 2 částí, protože implementace back endu zahrnuje implementaci jednotlivých AOS, jejichž součástí je implementace databáze.

7.1 Front end

Front end systému představuje grafické uživatelské rozhraní, které umožňuje uživateli pracovat s xAPI a ovládat AOS komunikací s back endem. Uživatelská aplikace byla implementována v jazyce Java za použití knihovny **JavaFX**.

7.1.1 Grafické uživatelské rozhraní

Implementace každé scény je realizována s využitím `FXML` souboru a s ním spojeného ovladače, který implementuje rozhraní `Initializable`. Rozhraní `Initializable` nabízí možnost inicializovat data při vytvoření instance třídy. Jelikož instance třídy ovladače se bude vždy automaticky vytvářet při načítání FXML souboru pro změnu scény, tak se s instancí ovladače nedá jinak pracovat. Z tohoto důvodu byl také použit návrhový vzor „Singleton“ neboli jedináček, pomocí kterého se mohla předávat data mezi ovladači. Pro vytváření FXML souboru byl použit nástroj Gluon Scene Builder, který nabízí grafické rozhraní pro tvorbu JavaFX grafického rozhraní.

Třídy využívající návrhový vzor jedináček fungují na principu existence pouze jedné instance této třídy za celý běh programu. Tohoto efektu je docíleno použitím privátního konstruktoru a statické metody `getInstance()`, která vrací instanci této třídy. Tato metoda v případě prvního zavolání neboli zavolání, kdy instance ještě neexistuje, instanci vytvoří a následně navrátí. Tento efekt je možno využít pro uložení objektu do instance v dané scéně a následného získání tohoto objektu v inicializační funkci ovladače jiné scény.

7.1.2 Komunikace s back endem

Pro komunikaci s back endem jsou používány HTTP požadavky na portu 443 z důvodu využití HTTPS. Pro příjem těchto požadavků je na back endu implementováno REST API, viz níže v oddíle 7.2.1. K vytváření těchto požadavků je použita knihovna **okhttp3**. Pro ověření komunikace přes HTTPS byl využit soubor Java KeyStore, soubor s příponou **.jks**, který obsahuje „self-signed“ certifikát (certifikát podepsaný sám sebou). Implementace vytvoření klienta pro odesílání požadavků za použití HTTPS viz výpis 7.1.

```
private final OkHttpClient client;
private Server() throws Exception {
    KeyStore trustStore = KeyStore.getInstance("JKS");
    try (InputStream trustStoreInput = new FileInputStream("server.jks")) {
        trustStore.load(trustStoreInput, "Bakalarka2023".toCharArray());
    }
    TrustManagerFactory trustManagerFactory = TrustManagerFactory
            .getInstance(TrustManagerFactory.getDefaultAlgorithm());
    trustManagerFactory.init(trustStore);
    X509TrustManager trustManager = (X509TrustManager) trustManagerFactory
            .getTrustManagers()[0];
    SSLContext sslContext = SSLContext.getInstance("TLS");
    sslContext.init(null,
                    new TrustManager[]{trustManager},
                    new SecureRandom());
    SSLSocketFactory sslSocketFactory = sslContext.getSocketFactory();
    client = new OkHttpClient.Builder()
            .sslSocketFactory(sslSocketFactory, trustManager)
            .build();
}
```

Výpis 7.1: Implementace vytvoření instance OkHttpClient

Z důvodu využívání technologie JWT pro autentizaci je nezbytné v každém HTTP požadavku posílat i JWT, viz výpis 7.2.

```
public String get(String endpoint) throws IOException {
    Request request = new Request.Builder()
            .url(BASE_URL + endpoint)
            .addHeader("Authorization", "Bearer " +
                    UserInterface.getInstance().getToken())
            .get()
            .build();

    return executeRequest(request);
}
```

Výpis 7.2: Příklad tvorby HTTP požadavku s JWT

7.1.3 Získání kurzů měn

Jak již bylo zmíněno výše v podkapitole 5.3, tak jednou z nevýhod xAPI je nemožnost získání použitého kurzu měn. Proto pro získávání kurzů měn byl použit portál **revolut.com**, který nabízí API s omezeným počtem požadavků zdarma. Tak bylo možno získávat kurzy s využitím HTTP GET požadavku ve tvaru `https://www.revolut.com/api/exchange/quote/amount=1&country=GB&isRecipientAmount=false&fromCurrency=USD&toCurrency=CZK`, přičemž odpověď obsahuje přesný kurz pro převod z měny USD do měny CZK.

7.2 Back end

Back end v tomto systému představuje server, na kterém poběží jednotlivé AOS asynchronně vůči uživateli, takže obsahuje implementaci všech navržených AOS, viz oddíl 7.2.4. Server také neustále přijímá HTTP požadavky od front endu za pomocí implementovaného REST API a řeší logiku zabezpečení této komunikace pomocí HTTPS a JWT, viz oddíl 7.2.3. Byl implementován v jazyce Java za použití frameworku Java Spring Boot, přičemž hlavními použitými technologiemi tohoto frameworku jsou Spring Data REST a Spring Data JPA, viz oddíl 7.2.1 a 7.2.2. Spring Boot z velké části používá pro implementaci anotace. Anotace vždy začíná znakem „@“ (například „@Override“. Tyto anotace jsou velkou součástí implementace REST API a JPA.

7.2.1 REST API

Jak funguje a co by REST API mělo obsahovat a splňovat, bylo zmíněno v oddílu 4.2.1. Proto tato část se zaměřuje na popis implementace tohoto API za použití Spring Data REST.

Pro vytvoření REST API musí back end obsahovat ovladače pro přijetí HTTP požadavků. Tyto ovladače jsou třídy, které se označují anotací `@RestController`. Každý ovladač by měl kontrolovat pouze podskupinu všech možných požadavků zaslaných front endem. Pro rozdělení skupin se používá anotace `@RequestMapping`, která vyžaduje parametr ve formátu `String` neboli řetězec. Tento řetězec představuje skupinu požadavků. Požadavky, které budou do této skupiny patřit, vždy musí obsahovat tento řetězec, viz výpis 7.3.

```
@RestController
@RequestMapping("buyCommand") //https://localhost:443/buyCommand/...
public class BuyCommandController {
    private final BuyCommandService buyCommandService; //Obsluha
}
```

Výpis 7.3: Příklad třídy s anotací RestController

Ovladač by však měl požadavky pouze přijímat. Jejich obsluhu již řeší třída s anotací `@Service`, přičemž každý ovladač by měl mít vlastní obsluhu, se kterou pracuje. V tomto případě se jedná o **BuyCommandService** viz výpis 7.3. Tato třída již obslouží přijatý požadavek.

U implementace přijetí požadavku je potřeba specifikovat:

- mapování HTTP metody – `@GetMapping, @PostMapping, @PutMapping, ...`,
- datový typ odpovědi,
- proměnné předané v URL požadavku – `@PathVariable`,
- proměnné předané v těle požadavku – `@RequestBody`,
- proměnné v hlavičce požadavku – `@RequestHeader`,
- zavolání obslužné metody.

Příklad implementace přijetí požadavku viz výpis 7.4.

```
@DeleteMapping("/{id}") //https://localhost:443/buyCommand/{id}
public boolean deleteBuyCommand(@PathVariable int id) {
    return buyCommandService.deleteBuyCommand(id); //zavolani metody
}
```

Výpis 7.4: Příklad přijetí požadavku s HTTP metodou DELETE

7.2.2 JPA – databáze

Součástí tohoto oddílu o implementaci JPA je i implementace databáze, protože Spring Data JPA dokáže potřebné tabulky v databázi vytvořit.

JPA z databáze dělá kolekci objektů. Z tohoto důvodu první fází implementace JPA jsou modely objektů neboli dat v databázi. Modely dat tabulek v databázi jsou stejné jako v návrhu databáze viz obrázek 6.9. Pro správnou funkcionalitu musí tyto modely obsahovat správné vazby mezi sebou a na tyto vazby se používají anotace `@OneToOne, @ManyToOne, @OneToMany a @ManyToMany`. Jako příklad je uveden model dat tabulky „symbol“ viz výpis 7.5.

Model musí být označen anotacemi `@Entity` a `@Table` s parametrem udávající název tabulky v databázi. Anotace `@Id` označuje atribut, který je použit jako primární klíč, a `@GeneratedValue` specifikuje funkci pro generování této hodnoty. Další použitou anotací je `@Column`, která je implicitní, ale z důvodu příkladu je zde uvedena. Tato anotace zdůrazňuje, že atribut bude mít sloupec v databázi. Poslední použitou anotací je `@OneToMany`, která specifikuje vazbu 1:n a ze záznamu v tabulce „symbol“ vytváří rodičovský záznam pro záznamy v tabulce „favourite“. Proto je atribut datového typu `List<>` a fyzicky v databázi sloupec nemá. U anotace `@OneToMany` je potřeba specifikovat mapování a popřípadě „fetch“ a „cascade“. „Fetch“ specifikuje, jak budou data načítána z databáze. V případě `FetchType.LAZY` označuje načtení těchto hodnot pouze v případě jejich použití. „Cascade“ definuje mazání synovských záznamů v databázi při smazání rodičovského záznamu. `CascadeType.ALL` definuje smazání všech synovských záznamů.

```
@Entity
@Table(name = "symbol")
public class Symbol {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column
    private String code;

    @OneToMany(fetch = FetchType.LAZY, mappedBy = "symbol",
            cascade = CascadeType.ALL)
    private List<Favourite> favourites;
    ...
}
```

Výpis 7.5: Příklad modelu dat tabulky symbol

Po vytvoření modelu je ještě zapotřebí vytvořit repozitář, který je implementován jako rozhraní a obsahuje Java Persistence Query Language (JPQL) dotazy. Označuje se anotací

`@Repository`. Repozitář je napojen na model pomocí rozšíření rozhraní `JpaRepository<>`, které jako parametry přijímá třídu modelu a datový typ primárního klíče. Viz výpis 7.6.

```
@Repository
public interface BuyCommandRepository
       extends JpaRepository<BuyCommand, Integer> {

    @Query("SELECT b FROM BuyCommand b WHERE b.user.id = ?1")
    List<BuyCommand> getBuyCommandsForUser(int id);
}
```

Výpis 7.6: Příklad rozhraní repozitáře

7.2.3 Zabezpečení

Jak již bylo výše uvedeno, tak zabezpečení probíhá za pomocí šifrované HTTPS komunikace a autentizace uživatelů přes JWT. JWT je generován při přihlášení uživatele do front endové aplikace. V HTTP požadavku je zasláno uživatelské ID a heslo, jenž je následně ověřeno za pomocí xAPI. Pokud jsou tyto údaje validní a ještě nejsou v databázi, tak jsou do ní uloženy. Po uložení dat do databáze se uživateli vygeneruje JWT, který je zaslán v odpovědi tohoto požadavku. JWT obsahuje ID uživatele a důležité údaje pro zabezpečení, které se generují implicitně. JWT také autorizuje uživatele, aby mohl zasílat veškeré ostatní HTTP požadavky, které back end nabízí. Přitom jediný požadavek, který může uživatel zaslat bez JWT, je právě požadavek pro přihlášení.

JWT je používán i pro ověření uživatele při vykonávání požadavků, aby uživatel mohl vykonávat požadavky pouze nad záznamy v databázi, které jsou spojeny s ním. Příklad získání uživatele z JWT viz výpis 7.7.

```
User user = (User) SecurityContextHolder.getContext()
       .getAuthentication().getPrincipal();
```

Výpis 7.7: Příklad získání uživatele z JWT

HTTPS je zajištěno pomocí SSL na straně serveru a TLS na straně aplikace. Protože během vývoje je aplikace a server spouštěn na stejném zařízení, byl vytvořen „self-signed“ certifikát a uložen v souboru **keystore.p12**. SSL je následně implementováno v konfiguračním souboru pro Spring Boot. V tomto případě se jedná o soubor **application.properties**.

7.2.4 AOS

Pro automatické vykonávání AOS byla využita knihovna **scheduling** obsažená v Spring Boot frameworku. Avšak pro vykonávání v určitých intervalech byl použit **cronExpression**, který zajistí vykonávání naplánované metody v cyklu. Příklad využití viz výpis 7.8. CronExpression je následně kombinován s anotací `@Scheduled` z již zmíněné knihovny.

```
@Scheduled(cron = "0/30 * * * * *") // kazdych 30 sekund
public void execute() {}
```

Výpis 7.8: Příklad cronExpression

Automatické nakupování instrumentu

Implementace postupovala podle návrhu a tedy nejprve se při každém vykonání metody volané za pomocí cronExpression získávají veškeré příkazy z databáze, které jsou potřeba vykonat. Jedná se o příkazy, u kterých čas od jejich posledního vykonání je vyšší než stanovený interval, viz výpis 7.9.

```
return buyCommandRepository.findAll()
        .stream()
        .filter(buyCommand -> {
            Date today = new Date();
            if (today.after(buyCommand.getStop())) {
                return false; //Prikaz byl ukoncen
            } else if(buyCommand.getLast() == null) {
                return true; //Prvni vykonani prikazu
            }
            return (today.getTime() - buyCommand.getLast()
                    .getTime() > buyCommand.getRepeat_interval()
                            .getMilliseconds());
        })
        .collect(Collectors.toList());
```

Výpis 7.9: Získání příkazů

Pokud existují takové příkazy, je nezbytné back end připojit k xAPI s údaji o daném uživateli a zjistit, zda je daný trh otevřen. V případě, že je trh otevřen, se musí vypočítat maximální množství instrumentu, které je možné nakoupit, aby se mohla provést transakce. Výsledek transakce se zaznamená do notifikací uživatele a u příkazu se změní datum posledního vykonání. Pokud je trh uzavřen, tak se příkaz přeskočí a bude tedy znovu kontrolován při dalším vykonávání metody. Objem transakce je zjištěn použitím vzorce, který započítává aktuální nákupní cenu instrumentu, jeho finanční páku a kurz převodu měny účtu na měnu instrumentu. S finanční pákou se zde musí počítat, protože uživatel si může s určitými finančními prostředky zakoupit větší množství daného instrumentu, pokud finanční páka tohoto instrumentu není rovna 100%. Použitý vzorec vypadá tedy následovně

$$V = \lfloor \frac{M}{P \times R \times L_{step}} \rfloor \times L_{step},$$

kde

- V – objem transakce,
- M – finanční prostředky,
- P – nákupní cena instrumentu,
- R – kurz převodu z měnu účtu na měnu instrumentu,
- L_{step} – krok přírůstku množství.

Avšak použití tohoto vzorce je závislé pouze na instrumentech, které se nenakupují s marží. Pro výpočet množství k nakoupení xAPI nabízí pro instrumenty nakupující se

s marží příkaz `getMarginTrade`, který získá marži instrumentu vyčíslenou v měně účtu. Tuto marži lze již jednoduše použít pro výpočet množství za pomocí vzorce

$$V = \lfloor \frac{M \times L_{min}}{m_{min} \times L_{step}} \rfloor \times L_{step},$$

kde

- L_{min} – minimální množství, se kterým je možno obchodovat.

Při implementaci tohoto AOS vznikl problém s investováním příliš malého objemu finančních prostředků do instrumentu s vysokou tržní cenou. Toto je zapříčiněno nedostatkem xAPI, které neumožňuje nákup frakčních akcií. Pro vyřešení tohoto problému si každý příkaz v databázi ukládá přebytečné finanční prostředky z předchozího vykonání, které následně může použít při následujících vykonání příkazu. Tedy pokud uživatel chce investovat určitý objem finančních prostředků měsíčně do instrumentu, který má minimální nákupní cenu vyšší než je daný objem, tak se AOS samo přizpůsobí a bude si finanční prostředky ukládat, dokud tyto finanční prostředky nebudou stačit k vykonání transakce.

Automatické vyvažování portfolia bez prodávání

Získávání příkazů je implementováno obdobně jako u automatického nakupování instrumentu. Dalším krokem však je test kompatibility, který rozděluje příkazy na 2 skupiny. Implementace kompatibilních příkazů je jednoduší, protože zde nevzniká potřeba vykonávat tento příkaz vícekrát. U kompatibilních příkazů se ověří otevřenost všech trhů. Pokud jsou všechny trhy otevřeny, vypočítá se, kolik je zapotřebí finančních prostředků pro dokoupení každého instrumentu tak, aby odpovídal procentuálnímu nastavení. Toho je docíleno použitím vzorce

$$M_i = \frac{M \times p_i}{p} - V_i \times P \times R,$$

kde

- M_i – finanční prostředky potřebné pro dokoupení instrumentu,
- M – celkové volné finanční prostředky,
- p_i – procentuální nastavení instrumentu,
- p – součet procentuálních nastavení všech instrumentů,
- V_i – vlastněný objem instrumentu,
- P – nákupní cena instrumentu,
- R – kurz převodu z měnu účtu na měnu instrumentu.

Tento vzorec opět platí pouze pro instrumenty bez marže. Pro instrumenty s marží je použit upravený vzorec

$$M_i = \frac{M \times p_i}{p} - m_{V_i},$$

kde nově přidaný symbol

- m_{V_i} – aktuální marže instrumentu s vlastněným množstvím.

Po výpočtu finančních prostředků se instrumenty seřadí tak, aby se nejprve nakupoval instrument s nejvyšším počtem potřebných finančních prostředků. Toto seřazení je důležité v případě nedostatečných finančních prostředků pro nakoupení všech instrumentů. AOS vždy nejprve nakupuje instrument, kterému schází nejvíce finančních prostředků, aby odpovídal procentuálnímu nastavení. Tímto postupem se vždy nejprve nakupuje instrument, který nejvíce klesá na hodnotě. Následně se provede postupný nákup za použití stejného vzorce jako u předchozího AOS, viz oddíl 7.2.4. Nákup se provádí pouze u instrumentů, jejichž potřebné finanční prostředky jsou dostačující k zakoupení nejmenšího možného množství.

Problém však přichází při implementaci příkazu nad nekompatibilním portfoliem. Pro řešení takovýchto příkazů byl přidán do databáze k záznamům v tabulce `rebalance_symbol` sloupec `executed`. Tento sloupec indikuje, zda již byl daný symbol v příkazu nakoupen nebo ne. Toto je důležité z důvodu několikanásobného vykonávání daného příkazu, protože se musí vykonat v různých časech. Zároveň bylo nutné přidat sloupec `executing` do tabulky `rebalance_command`, který představuje proces vykonávání daného příkazu. To je podstatné proto, aby se vícenásobně nepřičítali finanční prostředky k použitelným finančním prostředkům. Jediný rozdíl oproti kompatibilním příkazům je vypočítání potřebných finančních prostředků pro dokoupení a seřazení před ověřením otevřenosti jednotlivých trhů, protože se následně podle seřazení trhy kontrolují jednotlivě. Vykonání transakce je poté stejné, přičemž na konci vykonání transakce se instrumenty zaznamenají jako vykonané nebo se zaznamenají jako vykonané hned při výpočtu, pokud jejich potřebné finanční prostředky jsou záporné, aby nevznikalo čekání na otevření zbytečných trhů.

Jedinou možností, jak způsobit prodávání instrumentů u tohoto AOS, je změna nastavení uživatelem, přičemž tato změna obsahuje kompletní odstranění instrumentu z portfolia. Při takové změně je AOS nucen uzavřít veškeré pozice držené s tímto instrumentem a finanční prostředky rozdělit podle nově nastaveného portfolia.

Automatické vyvažování portfolia s prodáváním

Jak již bylo zmíněno v návrhu, tak tato implementace AOS pouze rozšiřuje předchozí systém o prodávání instrumentů jako možnost lepšího a znatelnějšího vyvažování instrumentů. Prodávání se však vztahuje pouze na akcie a ETF, protože s ostatními instrumenty se namísto nakupování a prodávání obchoduje stylem otevírání krátkých a dlouhých pozic.

Implementace většiny částí systému proběhla obdobně jako u předchozího AOS. Mezi rozdílné části patří výpočet objemu portfolia, seřazení instrumentů a vykonání transakce pro instrumenty s objemem představujícím vyšší procentuální část, než bylo nastaveno uživatelem.

U implementace výpočtu objemu portfolia musel být u každého instrumentu vypočítán vlastněný objem, který splňuje časový test. Tento objem je tudíž osvobozen od daně z příjmu a je možné ho prodat za účelem vyvážení portfolia.

K implementaci seřazení nestačilo řadit instrumenty sestupně z důvodu potřeby upřednostnění prodeje před nákupem, aby finanční prostředky získané z prodeje mohly být využity k nákupu. Proto byla vytvořena metoda, která řadí veškeré instrumenty s kladnými potřebnými finančními prostředky sestupně, ale na začátek tohoto seřazeného seznamu jsou přidány veškeré instrumenty se zápornými potřebnými finančními prostředky. Toto seřazení implikuje upřednostnění prodávání instrumentů před nakupováním.

Vykonání transakcí bylo implementováno obdobně jako u předchozího AOS, avšak musela být přidána implementace prodeje instrumentů se zápornými potřebnými finančními

prostředky. Tato implementace se v základu neliší oproti implementaci nákupu, pouze namísto omezení objemu transakce rozpočtem je omezena maximálním množstvím, které je možno prodat z důvodu časového testu.

Problém s prodáváním vzniká při používání AOS současně s ručním obchodováním nebo jiným AOS na stejném instrumentu. Protože u akcií a ETF nelze uzavřít specifickou pozici, ale vždy se uzavře ta nejdéle držená, nelze hlídat, aby dané AOS prodávalo pouze množství, které je přiřazeno k danému AOS. Řešením daného problému je používání AOS na samostatném účtě, na kterém nebudou probíhat jiné obchody.

U řešení problému s nekompatibilním portfoliem se na rozdíl od předchozího AOS musí provádět transakce ve dvou fázích, a to nejprve prodávání a následně nakupování. Dokud se neprovede prodej všech instrumentů, které jsou potřeba prodat, tak se nezačnou žádné instrumenty nakupovat.

Automatické obchodování na základě plovoucích průměrů

Implementace tohoto AOS se již velmi liší oproti předchozím třem, protože na rozdíl od vyčkání předem stanoveného intervalu, musí AOS neustále sledovat změny v cenách zvoleného instrumentu. Před kontrolou se získají svíčková data všech potřebných instrumentů pro všechny AOS všech uživatelů a nad těmito daty se spočítají veškeré různě nastavené indikátory. Pokud daný indikátor vydá signál k činnosti, zjistí se v jakém směru tento signál vznikl a rozhodne se o otevření či uzavření dlouhých pozic. Jestliže instrument podporuje i krátké pozice, otevírají a zavírají se i krátké pozice. Přičemž krátké pozice se otevírají při zavírání dlouhých a naopak.

Při implementaci vznikl stejný problém s instrumenty podporující pouze dlouhé pozice jako u předchozího AOS. Řešení tohoto problému je obdobné, tedy užívání AOS na separátním účtě samostatně, což je doporučováno i z důvodu využívání veškerých finančních prostředků účtu. A v případě použití více AOS na stejném účtě, by totiž vždy fungovalo pouze jedno z nich.

Implementované algoritmy byly již popsány v podkapitole 3.1, ale jejich použití se lehce liší. Z důvodu možnosti nastavení vzorkovací frekvence neboli časového intervalu mezi vzorky byl výpočet plovoucích průměrů adaptován. Perioda již neznačí počet dní pro výpočet, jak bylo zmíněno v podkapitole 3.1, ale značí počet vzorků pro výpočet. Pokud je tedy perioda nastavena na 60 a vzorkovací frekvence je 1 minuta, tak pro výpočet budou použity vzorky za posledních 60 minut.

Avšak z důvodu podpory použití malých intervalů vznikl problém s použitím velkého množství vzorků (například v minutových intervalech). Cena může konvergovat okolo rozhodujícího bodu, což vede ke vzniku několika nákupních a prodejních signálů, čímž mohou vznikat ztráty z důvodu rozdílné prodejní a nákupní ceny instrumentu. Bohužel řešení tohoto problému není jednoznačné, protože nelze předem určit, zda vytvořený signál ve velmi krátké době od předchozího je správné nebo špatné rozhodnutí systému. Možným řešením je implementace limitace vytvoření signálu na minimální možné době od předchozího signálu. Avšak stanovení této doby není opět jednoznačné. V implementaci byl zvolen časový interval jedné hodiny, což způsobilo zamítnutí velkého množství správných signálů. Postupně byl interval snižován až na 5 minutový interval, který naplňoval předpokládanou funkčnost u indikátorů se vzorky v 5 minutových intervalech. Z tohoto zjištění vyplývá implementace intervalu, který je roven vzorkovací frekvenci. Pro vyšší vzorkovací frekvence (den, týden, měsíc) byl tento interval limitován maximální hodnotou jednoho dne.

Kapitola 8

Testování

Automatické testování probíhalo pouze nad jednotlivými AOS a ostatní testování bylo už jen manuální. I když automatické testování probíhalo pouze nad AOS, tak představovalo velkou část této práce. Protože všechny AOS měly být testovány nad historickými daty, zvolil jsem testování za pomocí simulace s vlastním časem. Implementace všech AOS musela být upravena pro práci s daty ze souborů.

8.1 Získání historických dat

K získání historických dat byl využit xAPI za použití příkazu `getChartLastRequest`. Specificky, pokud se tento příkaz použije s parametrem počátečního času nastaveným na "0"a denním intervalem mezi vzorky, získají se všechna historická data daného instrumentu. Tato data byla následně uložena v `csv` souborech pro další použití. Protože hlavním zaměřením této práce je vytvoření AOS použitelných pro obchodování s akciemi, mezi staženými daty jsou veškeré akcie, se kterými lze obchodovat pomocí xAPI.

8.2 Simulace

Dalším krokem testování byla samotná simulace s historickými daty. Simulace postupně prochází historická data den po dni a rozhoduje, zda v daný den otevřít či uzavřít pozici. Toto chování je očekáváno od všech AOS v reálném čase. Nevýhodou výsledků simulace je nemožnost určení rozdílu mezi otevírací a uzavírací cenou instrumentu historických dat. Pro simulaci byl alespoň použit rozdíl v prodejní a nákupní ceně instrumentu v současnosti. Reálný zisk by se tedy ve skutečnosti mohl mírně lišit oproti tomu zjištěnému na konci simulace.

Počátkem simulace byl zvolen den **1. července 2010**. Počáteční bod byl zvolen již při prvním spuštění simulace, protože to byl nejstarší bod, ze kterého bylo možno získat historická data kurzů měn. I když nakonec byla získána historická data i z doby před tímto datem, tak počátek simulace se nezměnil. Veškeré AOS byly následně simulovány s různými parametry nad získanými daty. Výsledky simulací odpovídají datům získaným dne **15. dubna 2024**. Získávání dat převodů měn probíhalo manuálně a byla získána již **27. března 2024**. Problém vznikl u měn PLN a NOK, u kterých se nepodařilo získat historická data a byl pouze použit aktuální kurz převodu měn.

Simulace byla spouštěna na zařízení s operačním systémem Windows 10, procesorem Intel Core i7-7700, diskem Intel SSD 256 GB a 16 GB RAM. Na zmíněném zařízení trvalo získání veškerých výsledků simulací uvedených v této kapitole okolo 60 hodin.

8.3 Porovnání výsledků simulace jednotlivých AOS

Samozřejmě hlavní částí každé simulace je porovnání výsledků. Výsledky byly nejprve porovnány navzájem mezi různými konfiguracemi každého AOS a následně i mezi různými AOS. Mezi simulované AOS patří:

- automatické nakupování,
- automatické vyvažování portfolia bez prodávání,
- automatické vyvažování portfolia s prodáváním,
- automatické obchodování na základě indikátoru SMA,
- automatické obchodování na základě indikátoru EMA,
- automatické obchodování na základě indikátoru MACD.

V tabulkách jsou kromě celkových zisků za simulační období také uváděna roční zhodnocení, která jsou vypočítána za pomocí následujícího vzorce:

$$u = \sqrt[r]{\frac{Q_r}{Q}} - 1,$$

kde

- u – roční zhodnocení (úroková sazba) v procentech,
- r – počet let,
- Q – počáteční finanční prostředky,
- Q_r – finanční prostředky po r letech.

Tento vzorec byl odvozen ze vzorce pro složenou úrokovou sazbu:

$$Q_r = Q * (1 + u)^r.$$

8.3.1 Automatické nakupování

Simulace automatického nakupování byla spuštěna s pěti různými konfiguracemi. Mezi změny v těchto konfiguracích patří objem investovaných finančních prostředků a interval nakupování. Srovnání výsledků simulací, u kterých byl změněn pouze objem finančních prostředků pro investování, neukázalo žádný znatelný rozdíl, což je vidno níže v tabulce 8.1. Procentuální zisky u instrumentů, které jsou uvedeny u více simulací, jsou obdobné, proto je zde uveden pouze průměr z nich. Systém v simulaci se při nedostatku finančních prostředků pro nákup v daný okamžik zachová tak, že si finanční prostředky ukládá pro budoucí použití, dokud finanční prostředky nejsou dostatečné pro minimální nákup v budoucím termínu vykonání.

Finanční prostředky	500 Kč	1000 Kč	5000 Kč
Opakovací interval	1 měsíc		
Průměrný zisk	≈ 91.33 %	≈ 92.32 %	≈ 93.14 %
Roční zhodnocení	≈ 6.62 %	≈ 6.69 %	≈ 6.75 %
Počet instrumentů, které utrpěly ztráty	1222 (34.46 %)	1230 (34.69 %)	1238 (34.91 %)
Instrumenty s nejvyšším ziskem za celé období investování	Nvidia (roční zhodnocení ≈ 786.4 %) Fon SE (≈ 241.1 %) Sears Holdings Corp (≈ 215.3 %)		
	Celsius Holdings Inc (≈ 212.3 %) Tesla Motors (≈ 207.7 %)		Builders F. (≈ 210.8 %) Celsius H. (≈ 210.2 %)

Tabulka 8.1: Porovnání automatického nakupování při změně objemu finančních prostředků

Po prvním porovnání bylo možné říci, že navýšení investovaných finančních prostředků jen mírně zvyšuje zisk, avšak tato změna není velmi znatelná, proto simulace se změnou intervalu nakupování již byly spouštěny se stejným objemem finančních prostředků.

V tabulce 8.2 je vyobrazeno porovnání výsledků simulací se změnou frekvence nakupování.

Finanční prostředky	5000 Kč		
Opakovací interval	1 týden	1 měsíc	3 měsíce
Průměrný zisk	≈ 89.61 %	≈ 93.14 %	≈ 182.67 %
Roční zhodnocení	≈ 6.49 %	≈ 6.75 %	≈ 13.24 %
Počet instrumentů, které utrpěly ztráty	1251 (35.24 %)	1238 (34.91 %)	1290 (37.02 %)
Instrumenty s nejvyšším ziskem za celé období investování	Nvidia Fon SE Sears H. Tesla Builders F.	Nvidia Fon SE Sears H. Builders F. Celsius H.	Nvidia (≈ 2 181.4 %) Tesla (≈ 909.4 %) Builders F. (≈ 631.0 %) Axon Enter. (≈ 604.8 %) Repligen (≈ 406.9 %)

Tabulka 8.2: Porovnání simulací při změně frekvence nakupování (opakovací interval)

Porovnání ukázalo, že frekvence 3 měsíců dosahuje nejvyššího zisku. Počet instrumentů se záporným ziskem se pohybuje okolo 35 % u všech konfigurací, tudíž nevzniká příliš vysoké riziko ztrát při zvolení špatného instrumentu. Akcie společnosti **Nvidia** dosahují nejvyšších zisků v každé konfiguraci, což je způsobeno jejich velmi vysokým růstem na ceně od začátku roku 2024.

8.3.2 Automatické vyvažování portfolia

Tento oddíl obsahuje porovnání automatického vyvažování portfolia bez prodávání i s prodáváním. Testování bylo prováděno nad dvěma portfolii, přičemž jedno z nich je složeno z akcií sedmi technických společností, které nejvíce rostou na ceně v poslední době. Jedná se o akcie společností Tesla, Nvidia, Apple, Microsoft, Alphabet (Google), Meta, Amazon [16]. Druhé portfolio bylo tvořeno z 495 akcií obsažených v indexu S&P500. Pět akcií bylo vyřazeno, protože s nimi nelze obchodovat u brokera XTB. Porovnání výsledků se trochu

liší oproti předchozímu AOS, protože u této simulace je pouze počítán zisk z portfolia jako celku. Níže v tabulce 8.3 je uveden zisk různých konfigurací tohoto AOS.

Prodávání	Finanční prostředky	Opakovací interval	Průměrný zisk	Roční zhodnocení
Ano	500 Kč	1 měsíc	≈ 1410.38 %	≈ 102.22 %
	1000 Kč		≈ 1339.06 %	≈ 97.05 %
	5000 Kč	1 týden	≈ 502.05 %	≈ 36.39 %
		2 týdny	≈ 1006.72 %	≈ 72.97 %
		1 měsíc	≈ 1057.75 %	≈ 76.66 %
		3 měsíce	≈ 1087.57 %	≈ 78.83 %
Ne	500 Kč	1 měsíc	≈ 3459.23 %	≈ 250.72 %
	1000 Kč		≈ 3081.56 %	≈ 223.35 %
	5000 Kč	1 týden	≈ 3002.85 %	≈ 217.64 %
		2 týdny	≈ 2887.69 %	≈ 209.30 %
		1 měsíc	≈ 2995.74 %	≈ 217.13 %
		3 měsíce	≈ 2172.84 %	≈ 157.48 %

Tabulka 8.3: Porovnání výsledků simulací nad portfoliem ze sedmi technických společností

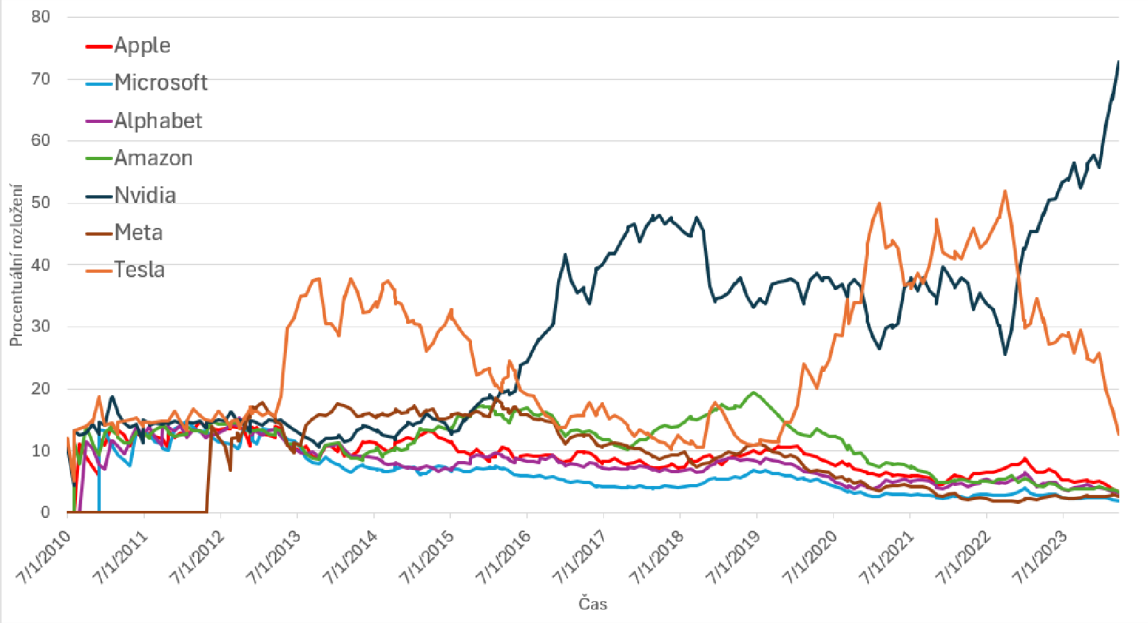
Z tabulky je zřejmé, že vyvažování s prodáváním dosahuje nižších zisků, kdy se tento zisk výrazně snižuje při vysoké frekvenci opakování. U obchodování bez prodávání se ukazuje, že menší intervaly dosahují naopak mírně vyššího zisku.

Výsledná data ze simulací nad portfoliem S&P500 ukázala nepoužitelnost automatického vyvažování na velkém počtu instrumentů. Příčinou je nemožnost nakupování frakčních akcií, čímž při velkém počtu instrumentů a nedostatečném objemu finančních prostředků vzniká problém s nakupováním. Zisk se tedy razantně zvyšuje při zvýšení objemu finančních prostředků. Zisky z několika konfigurací jsou porovnány níže v tabulce 8.4.

Prodávání	Finanční prostředky	Technické portfolio	S&P500 portfolio
Ano	500 Kč	≈ 102.22 %	≈ 1.35 %
	1000 Kč	≈ 97.05 %	≈ 4.20 %
	5000 Kč	≈ 76.66 %	≈ 5.81 %
Ne	500 Kč	≈ 250.72 %	≈ 2.98 %
	1000 Kč	≈ 223.35 %	≈ 5.44 %
	5000 Kč	≈ 217.13 %	≈ 11.57 %

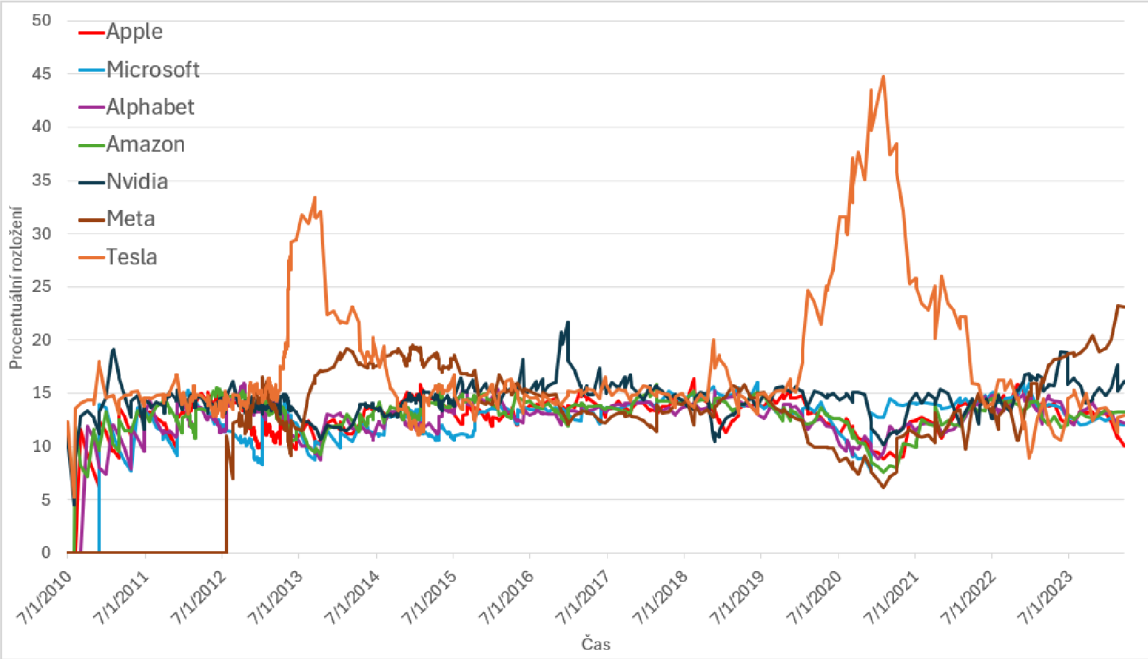
Tabulka 8.4: Porovnání ročního zhodnocení portfolií při měsíčním intervalu obchodování

U automatického obchodování s vyvažováním je velmi důležité, aby instrumenty odpovídaly svému procentuálnímu nastavení. Níže na obrázku 8.1 je možno vidět vývoj portfolia při obchodování bez prodávání. Vyobrazený graf odpovídá nejvýdělečnější konfiguraci z tabulky 8.3, kde je procentuální rozložení instrumentů rovnoměrné.



Obrázek 8.1: Vývoj procentuálního rozložení portfolia bez prodávání

Na grafu lze jednoznačně vidět, že pět ze sedmi instrumentů je velmi pod jejich procentuálním rozložením z důvodu velkého růstu na ceně zbylých dvou instrumentů. Akcie společností Tesla a Nvidia velmi vzrostly na ceně, a tedy jejich podíl na portfoliu se výrazně zvětšil. Tento jev však nelze vidět na obrázku 8.2, z čehož vyplývá, že využitím možnosti prodávání lze udržovat instrumenty na jejich přednastaveném rozdělení. V grafu si nelze nepovšimnout výjimek, kdy vyvažování portfolia nefungovalo. Jedná se o dobu, kdy nebylo možné prodávat instrumenty z důvodu nesplnění časového testu.



Obrázek 8.2: Vývoj procentuálního rozložení portfolia s prodáváním

8.3.3 Automatické obchodování s použitím plovoucích průměrů

Tento oddíl obsahuje porovnání všech tří implementovaných indikátorů využívajících plovoucí průměr. V tabulce 8.5 jsou uvedeny výsledky nejlepších konfigurací s použitím indikátorů SMA a EMA.[1]

Indikátor	Finanční prostředky	Počet vzorků	Průměrný zisk	Roční zhodnocení	Riziko
SMA	50 000 Kč	50	≈ 19.49 %	≈ 1.41 %	63.04 %
		100	≈ 38.00 %	≈ 2.75 %	57.09 %
		200	≈ 71.14 %	≈ 5.16 %	48.07 %
	100 000 Kč	50	≈ 9.71 %	≈ 0.70 %	66.33 %
		100	≈ 33.80 %	≈ 2.45 %	58.04 %
		200	≈ 64.23 %	≈ 4.66 %	49.34 %
EMA	50 000 Kč	50	≈ 10.5 %	≈ 0.76 %	66.69 %
		100	≈ 40.53 %	≈ 2.94 %	58.10 %
		200	≈ 69.98 %	≈ 5.07 %	51.03 %
	100 000 Kč	50	≈ 13.07 %	≈ 0.95 %	65.51 %
		100	≈ 40.59 %	≈ 2.94 %	58.35 %
		200	≈ 70.17 %	≈ 5.09 %	51.08 %

Tabulka 8.5: Porovnání indikátorů SMA a EMA

Z tabulky je evidentní znatelný nárůst na zisku a naopak snížení míry rizika při navyšování počtu vzorků pro výpočet obou indikátorů. Navyšování počátečních finančních prostředků má na výši zisku minimální vliv, přičemž tento vliv snižuje výši zisku při využití indikátoru SMA. Samozřejmě existuje minimální množství finančních prostředků, pro které se použití AOS vyplatí. Avšak pro nalezení takového minimálního množství by musela být simulace spuštěna mnohonásobně vícekrát.

Pro hledání nejlepšího počtu vzorků byly ještě simulovány počty vzorků 300, 400 a 500. Nejlepších výsledků dosáhly simulace nad indikátory s konfigurací s počtem 500 vzorků, která dosáhla ročního zhodnocení 7.23 % a míry rizika 48.10 % při indikátoru EMA a ročního zhodnocení 6.28 % a míry rizika 44.29 % při indikátoru SMA. Ze všech simulací vyplynulo, že vyšší počty vzorků dosahují z dlouhodobého hlediska lepších výsledků.

Instrumenty, které v minulosti byly ideálními pro indikátory SMA a EMA, jsou vypsány v tabulce 8.6. Jedná se o konfiguraci indikátorů s 200 vzorky.

Instrument	SMA (zisk v %)	EMA (zisk v %)
Besi	≈ 380.53 %	≈ 290.70 %
Fingerprint Cards	≈ 294.39 %	≈ 137.68 %
CD project	≈ 285.59 %	≈ 352.71 %
Nvidia	≈ 211.79 %	≈ 114.80 %
Netflix	≈ 209.02 %	≈ 244.10 %
Vipshop	≈ 76.89 %	≈ 440.89%
Hexatronic Group	≈ 45.83 %	≈ 329.12 %
Celldex Therapeutics, Inc.	≈ 101.99 %	≈ 317.13 %

Tabulka 8.6: Porovnání ideálních instrumentů pro indikátory EMA a SMA

[1] Riziko představuje procento instrumentů, které utrpěly ztráty.

Z testování vyplynulo, že indikátory SMA a EMA dosahují podobných průměrných zisků, avšak zisky na daných instrumentech se velmi liší. Některé velmi výdělečné instrumenty pro jeden indikátor nebyly ideálními instrumenty pro druhý indikátor. Příkladem takového instrumentu jsou akcie společnosti Hexatronic Group. Veškeré tyto výsledky jsou závislé na minulosti a nelze jednoznačně určit, zda budoucí cenový vývoj bude stejný. Již výše bylo zmíněno, že jedním z jevů instrumentů finančního trhu je opakování minulosti, proto je zde jistá šance, že použití AOS s touto konfigurací bude na daných instrumentech fungovat podobně jako na historických datech.

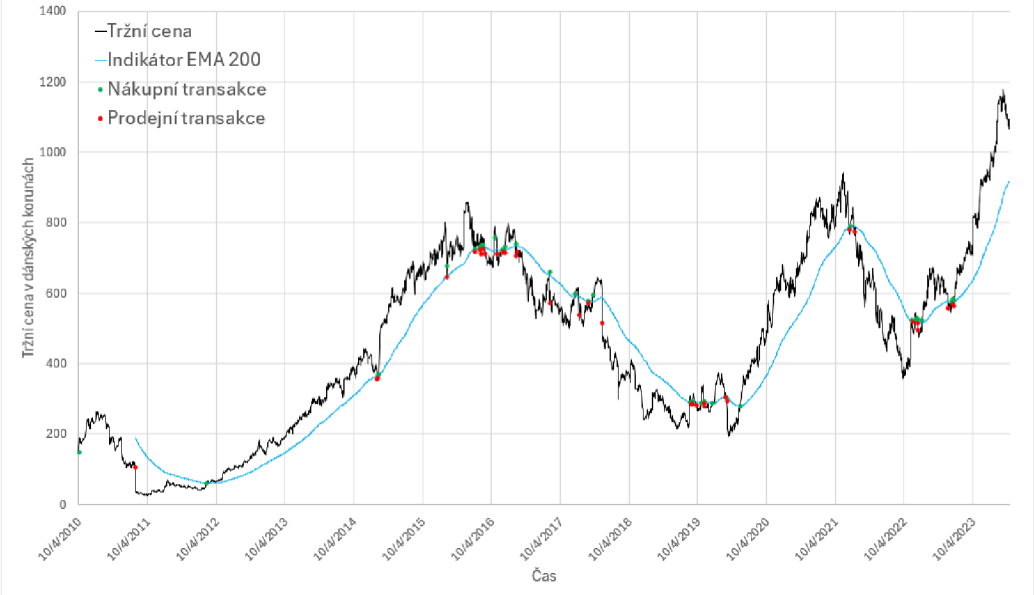
Pro demonstraci automatického obchodování na základě indikátorů SMA a EMA uvádím obrázky 8.3 a 8.4, na kterých lze vidět, kdy systém prováděl nákupní a prodejní transakce. Tato data byla získána simulováním nad tržními cenami akcií společnosti Pandora A/S. Tento instrument byl zvolen, jelikož na grafu vývoje jeho tržní ceny je možné viditelně znázornit potřebné průsečíky tržní ceny a indikátorů.

Z obrázku 8.3 vyplývá, že systém vytváří transakce ve chvíli, kdy tržní cena protne imaginární linii indikátoru SMA. Lze si povšimnout, že občas systém nevytvoří transakce ihned ve chvíli, kdy je tato linie protnuta. Důvodem vzniku tohoto jevu je přístup pouze k datům na začátku a na konci dne, tudíž pokud protnutí linie nastane během dne, tak se transakce provede až při otevření obchodních hodin v dalším dni. Druhým zásadním jevem, který obrázek zobrazuje, je již výše zmíněné konvergování okolo rozhodujícího bodu. Z tohoto důvodu je obvykle při každém protnutí linie vytvořeno několik nákupních i prodejních transakcí, čímž vznikají ztráty způsobené rozdílem nákupní a prodejní ceny instrumentu.

Obrázek 8.3: Demonstrace automatického obchodování na základě indikátoru SMA pro akcie společnosti Pandora A/S

Obrázek 8.4 zobrazuje demonstraci obchodování na základě indikátoru EMA a je na něm možno rozpoznat stejné jevy jako na předchozím obrázku. Velmi zajímavou transakcí je ihned první transakce, která byla vytvořena okamžitě při prvním možném termínu obchodování s tímto instrumentem. Toto bylo způsobeno tím, že indikátor EMA nespecifikuje počet vzorků před použitím vzorce pro výpočet indikátoru. Je možno takto vzniklou transakci považovat za chybovou, avšak já jsem tuto transakci považoval za validní a neopravoval

ji. Pokud by tato transakce nevznikla, dosahoval by instrument v tomto případě mírně vyšších zisků, protože po prvním prodání se instrument dostal do mínusu přibližně 15% a na zbytek obchodování by to nemělo vliv. V takovém případě by první platná transakce byla provedena až 8. listopadu 2012.



Obrázek 8.4: Demonstrace automatického obchodování na základě indikátoru EMA pro akcie společnosti Pandora A/S

Pro celkové porovnání indikátorů je nezbytné si ještě ukázat výsledky simulace obchodování na základě indikátoru MACD, přičemž AOS používající indikátor MACD je komplexně nejsložitějším systémem, který byl v této práci navržen. Jak již bylo uvedeno v podkapitole 3.1, MACD indikátor má předem nastavenou konfiguraci[2], proto jedinou možností uživatelské konfigurace je změna investovaných finančních prostředků a vzorkovací frekvence. Bohužel z důvodu možnosti získání historických dat pouze v denních intervalech nelze simulovat se změnou vzorkovací frekvence, ale pouze se změnou finančních prostředků. Níže v tabulce 8.7 jsou porovnána data ze simulací použití indikátoru MACD s různými finančními prostředky.

Počáteční finanční prostředky	50 000 Kč	100 000 Kč
Průměrný zisk	≈ 186.67 %	≈ 186.04 %
Roční zhodnocení	≈ 13.53 %	≈ 13.48 %
Riziko	≈ 39.17 %	≈ 39.31 %
Instrumenty s nejvyšším ziskem za celé období investování	Nvidia (≈ 1 813.4 %) Hexatronic Group (≈ 1 412.6 %) Tesla (≈ 1 107.1 %) Carvana (≈ 1 007.6 %)	

Tabulka 8.7: Porovnání simulací MACD při změně objemu finančních prostředků

[2]12 vzorků pro rychlý exponenciální plovoucí průměr, 26 vzorků pro pomalý, 9 vzorků pro signalizační linii a denní frekvence vzorkování.

Výsledky simulace obchodování na základě indikátoru MACD překvapily svým vyšším průměrným ziskem a zároveň nižším rizikem oproti ostatním indikátorům. Zisky jednotlivých instrumentů jsou několikanásobně vyšší než u jiných indikátorů. Tento systém tedy dosahuje nejlepších výsledků ze všech AOS, které byly v této práci vytvořeny. Avšak z důvodu možnosti simulace pouze nad vzorky s denním intervalem mezi vzorky je nemožné určit, zda by těchto výsledků systém dosahoval i například nad vzorky s minutovým či hodinovým intervalem. Taktéž lze z tabulky 8.7 vyčíst stejný jev jako u ostatních indikátorů, a to téměř žádnou závislost na výši finančních prostředků.

8.4 Simulovaní na nejobchodovanějších ETF

Již bylo zmíněno, že veškeré testování probíhalo pouze na všech akcií, se kterými lze obchodovat s využitím xAPI. Jako dodatečné testování byly simulovány všechny systémy na nejobchodovanějších ETF na platformě od XTB, které byly získány ze článku [31]. Důvodem těchto simulací byla skutečnost, že pro dosažení výše uvedených výsledků by investor musel vlastnit nereálně vysoké finanční prostředky. Proto mnohem realističtější možností je investice do instrumentu, jako je například S&P500 ETF, který se soustředí na index S&P500 ETF, jenž sleduje 500 největších amerických firem. Celkem bylo simulováno nad instrumenty:

- Xtrackers Dax UCITS ETF (XDAX.DE),
- Amundi Euro Stoxx 50 UCITS ETF (MSE.FR),
- SPDR S&P 500 UCITS ETF (SPY5.UK),
- iShares Core S&P 500 UCITS ETF (CSPX.UK),
- Amundi Euro Stoxx Banks UCITS ETF (BNKE.FR),
- iShares J.P. Morgan USD EM Bond UCITS ETF (IEMB.UK),
- iShares S&P 500 EUR Hedged UCITS ETF (IUSE.UK).

Výsledky simulací byly pouze porovnány mezi různými AOS, protože mezi různými konfiguracemi systémů platí stejné jevy, které šly vidět na výsledcích simulací nad akciemi. V tabulce 8.8 jsou uvedeny nejlepší konfigurace všech AOS. Výsledky dosahují nižších zisků než výsledky simulací na akciích, ale jsou mnohem realističtější z důvodu menšího počtu instrumentů, tudíž není potřeba příliš vysoký objem finančních prostředků.

AOS	**Konfigurace**	**Roční zhodnocení**	**Nejlepší instrument**
Nakupování	každé 3 měsíce 5000 Kč	≈ 5.92 %	CSPX.UK (12.19 %)
Vyvažování (bez)	každý týden 5000 Kč	≈ 5.35 %	—
Vyvažování (s)	každý měsíc 1000 Kč	≈ 4.48 %	—
Obchodování SMA	200 vzorků	≈ 2.49 %	BNKE.FR (8.57 %)
Obchodování EMA	200 vzorků	≈ 6.91 %	CSPX.UK (25.18 %)
Obchodování MACD	předem daná	≈ 7.37 %	CSPX.UK (26.15 %)

Tabulka 8.8: Porovnání simulací na nejobchodovanějších ETF na platformě od XTB

8.5 Porovnání automatických obchodních systémů

Základní cíl každého automatického obchodního systému je dosahovat co nejvyššího zisku s co nejnižší mírou rizika. Z výše porovnaných výsledků simulací lze jednoznačně říct, že žádné z implementovaných AOS nezaručuje vysoký zisk s nulovým rizikem. Bohužel nulového rizika při investování nelze dosáhnout, i když zkušení investoři ho dokáží velmi minimalizovat. Určit, které AOS je nejlepší, není zcela jednoznačně možné. AOS dosahujícím nejlepších výsledků je **automatické obchodování na základě indikátoru MACD**, protože oproti ostatním dosahuje v průměru vyššího zisku a nižší úrovně rizika. Jako jediný dosahoval lepších výsledků vůči pravidelnému investování. Ve většině případů dosahoval až dvojnásobně vyššího ročního zhodnocení, a tudíž dvojnásobného konečného zisku. I když automatické vyvažování bez prodávání nad technickým portfoliem dosahuje mnohokrát vyšších zisků, tak tyto výsledky jsou velmi závislé na zvolení správného portfolia. Simulace automatického obchodování s používání indikátorů SMA a EMA dosahovaly podobných výsledků, které byly ve většině konfigurací nedostačující, aby se vyrovnaly pravidelnému nakupování.

Kapitola 9

Závěr

Cílem této práce bylo navrhnout, implementovat různě komplexní AOS s použitím aplikačního programového rozhraní od společnosti XTB a následně na historických datech otestovat jejich funkčnost.

Celkem byly navrženy čtyři různě komplexní AOS. Implementace proběhla úspěšně a funkčnost implementace všech AOS byla otestována používáním v reálném čase, avšak výsledky tohoto testování jsou velmi limitovány krátkou dobou používání. Z důvodu potřeby velkého množství času pro získání validních výsledků testování byla veškerá implementace otestována na historických datech. Výsledky těchto testů byly již porovnány v předchozí kapitole.

Práce dosáhla svého cíle, kterým bylo zjistit, zda lze využít aplikační programové rozhraní poskytované společností XTB pro automatické obchodní systémy. Využití xAPI je určitě možné, ale z důvodu nemožnosti obchodovat s frakčními akciemi není zcela ideální.

Navržené AOS jsou v praxi použitelné, ale jejich použitím vzniká vysoké riziko ztráty finančních prostředků. Při zvolení správného instrumentu je možné, aby AOS dosahoval lepších výsledků než nezkušený investor. Zkušený investor by ale pravděpodobně dosahoval lepších a stabilnějších výsledků s nižším rizikem.

Všechny navržené systémy nejsou zcela dokonalé. V budoucnu je možné na tuto práci navazovat a návrhy nebo implementaci zdokonalit. Budoucí práce by se mohla zaměřit na snížení úrovně rizika u všech AOS. Jednou z možností je zabránit vysokým ztrátám použitím funkce `Stop Loss`. Tato možnost by však v některých situacích mohla způsobit nevyužití budoucího růstu ceny, a tedy dosahovat nižších zisků.

Literatura

[1] *Komodity: Co jsou komodity a komoditní trhy: Online trading* online. 22. Nov 2023. Dostupné z: https://www.xtb.com/cz/vzdelavani/komodity. [cit. 2.1.2024].

[2] *Kryptoměny* online. Dostupné z: https://www.kurzy.cz/kryptomeny/. [cit. 2.1.2024].

[3] *Obchodování indexů – Co je to akciový index?* online. 22. Nov 2023. Dostupné z: https://www.xtb.com/cz/vzdelavani/obchodovani-indexu-akciovy-index. [cit. 2.1.2024].

[4] Achelis, S. B. *Technical Analysis from A to Z.* McGraw Hill New York, 2001.

[5] Biehl, M. *API Architecture.* API-University Press, 2015.

[6] Biehl, M. *RESTful Api Design.* API-University Press, 2016. 81–102 s.

[7] BlackRock. *ETF VYSVĚTLENO* online. Dostupné z: https://www.blackrock.com/cz/individualni-investori/vzdelavani/co-je-etf#what-are-etfs. [cit. 23.12.2023].

[8] Feng, X.; Shen, J. a Fan, Y. REST: An alternative to RPC for Web services architecture. In: IEEE. *2009 First International Conference on future information networks.* 2009, s. 1–2.

[9] Fleischer, D. *Automated Trading System using Machine Learning.* 2019. Diplomová práce. University of Stavanger, Norway.

[10] Gupta, L. HATEOAS Driven REST APIs. online, Nov 2023. Dostupné z: https://restfulapi.net/hateoas/. [cit. 4.3.2024].

[11] Hakim, H. Forex Trading and Investment. *Diss., Worcester Polytechnic Institute* online, 2012. Dostupné z: https://digital.wpi.edu/downloads/m039k5293. [cit. 23.12.2023].

[12] Horčička, M. *Kryptoměny.* 2019. Bakalářská práce. Masarykova univerzita, Ekonomicko-správní fakulta.

[13] IBM. *Java Persistence API (JPA)* online. 30. Jan 2024. Dostupné z: https://www.ibm.com/docs/en/was-liberty/nd?topic=liberty-java-persistence-api-jpa. [cit. 7.3.2024].

[14] Kudláček, I. P. *Broker Velké srovnání brokerů pro rok 2023. Kde obchodovat akcie, etf, forex, indexy nebo komodity?* online. Redakce Finex, 2023. Dostupné z: https://finex.cz/rubrika/brokeri/. [cit. 16.12.2023].

[15] Ltd, R. *Algoritmické obchodování* online. Dostupné z: https://www.robomarkets.cz/beginners/info/expert-advisors/. [cit. 16.12.2023].

[16] Lu, M. Which Companies Make Up the "Magnificent Seven" Stocks? *Visual Capitalist* online, Dec 2023. Dostupné z: https://www.visualcapitalist.com/magnificent-seven-stocks/. [cit. 6.4.2024].

[17] Miley, S. a Lewis, S. The Top 6 Trading Strategies for 2023. online. Hantec Markets, Aug 2023. Dostupné z: https://hmarkets.com/5-best-trading-strategies-for-every-trader/. [cit. 13.1.2024].

[18] Naknok, S. Firm Performance Indicators as a Fundamental Analysis of Stocks and a Determinant of a Firm's Operation. *International Journal of Economics & Business Administration (IJEBA)*. International Journal of Economics & Business Administration (IJEBA), 2022, sv. 10, č. 1, s. 190–213.

[19] Poddar, R. What is JWT? Understanding JSON Web Tokens. *SuperTokens* online, Mar 2022. [cit. 7.3.2024].

[20] Scott, G. Trending Market: What it Means, How it Works. *Investopedia* online, Jul 2022. Dostupné z: https://www.investopedia.com/terms/t/trending-market.asp. [cit. 15.4.2024].

[21] Soni, A. a Ranga, V. API features individualizing of web services: REST and SOAP. *International Journal of Innovative Technology and Exploring Engineering*, 2019, sv. 8, č. 9.

[22] Strnad, J. *Finanční zprostředkovatelé a finanční trh* online. 2009. Disertační práce. Technická Univerzita v Liberci. Dostupné z: https://dspace.tul.cz/handle/15240/5198. [cit. 23.12.2023].

[23] Sukparungsee, S.; Areepong, Y. a Taboran, R. Exponentially weighted moving average—Moving average charts for monitoring the process mean. *Plos one*. Public Library of Science San Francisco, CA USA, 2020, sv. 15, č. 2, s. e0228208.

[24] Tanzu, V. *Spring Boot* online. Dostupné z: https://spring.io/projects/spring-boot#overview. [cit. 7.3.2024].

[25] Tesárek, R. *Akcie* online. 2015. Dostupné z: http://hdl.handle.net/11025/18064. [cit. 23.12.2023].

[26] Thomsett, M. C. *Getting started in fundamental analysis*. John Wiley & Sons, 2006.

[27] Tiwary, G. P.; Stroulia, E. a Srivastava, A. Compression of XML and JSON API Responses. *IEEE Access*. IEEE, 2021, sv. 9, s. 57426–57439.

[28] XTB. Co je Marže? *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/help-center/obchodni-ucet/co-je-marze. [cit. 13.1.2024].

[29] XTB. Dlouhá pozice (long position). *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/vzdelavani/long-pozice. [cit. 12.1.2024].

[30] XTB. Finanční páka. *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/vzdelavani/financni-paka-xtb. [cit. 12.1.2024].

[31] XTB. Investice do ETF s 0 *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/etf?utm_source=google&utm_medium=cpc&utm_campaign=search_etf_search_campaign_ID634&utm_content=offer_stocks_&_etf&utm_term=etf%202022&gad_source=1&gclid=CjwKCAjwuJ2xBhA3EiwAMVjkVBhhg1IwzcRk1TtqjcipbA3qRt3mQ-nZlbzqxloC63X4jD_tS4ydrBoCXgEQAvD_BwE. [cit. 26.4.2024].

[32] XTB. Krátká pozice (short position). *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/vzdelavani/short-pozice. [cit. 12.1.2024].

[33] XTB. Stop Loss a Take Profit. *XTB S.A.* online. Dostupné z: https://www.xtb.com/cz/vzdelavani/stop-loss-a-take-profit. [cit. 12.1.2024].

[34] Zaňka, Z. Jak na daně z investic a tradingu. *Quastic. cz*, 2017.