SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Alfint - Aplikace pro zpětné testování akcií

Autor: Vojtěch Šíma

Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2024/25

Poděkování

Děkuji všem, co mě psychicky podpořili, když jsem v kódu měl tisíce errorů.

Prohlášení

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým a prezentačním účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2024

. . . . . .

Podpis autora

Obsah

Abstrakt

Tato práce se zabývá vývojem aplikace pro zpětné testování akcií na základě uživatelsky definovaných parametrů. Aplikace je vyvinuta v jazyce Python s využitím frameworku Django a API SimFin a yfinance, což umožňuje kombinovat historická a aktuální data akcií. Hlavní funkcionality zahrnují analýzu historických dat, výpočet klíčových finančních ukazatelů, vizualizaci výsledků a možnost sdílení portfolií mezi uživateli. Dokumentace popisuje architekturu aplikace, implementační detaily a zvolené metody řešení. Kladen je důraz na přesnost výpočtů, přehlednou vizualizaci dat a uživatelskou přívětivost.

Klíčová slova

Python, Django, SimFin, yfinance, zpětné testování, finanční analýza, akcie

SEZNAM POUŽITÝCH ZKRATEK

API	Application Programming Interface,
EBITDA	Earnings Before Interest, Taxes, Depreciation, and Amortization,
EPS	Earnings Per Share,
P/E	Price-to-Earnings Ratio,
ROE	Return on Equity,
REST	Representational State Transfer,
SQL	Structured Query Language,
UI	User Interface.

ÚVOD

Investování a analýza akciového trhu se v posledních letech stávají stále populárnějšími. S rostoucím zájmem o tuto oblast roste i potřeba nástrojů, které by investorům umožnily lépe pochopit trh, analyzovat historická data a testovat své investiční strategie. Tento projekt byl vytvořen jako aplikace, která kombinuje historická data akcií, finanční ukazatele a vizualizace tak, aby uživatelé mohli získat ucelený přehled o výkonu jednotlivých akcií.

Hlavní funkcionalitou aplikace je integrace dat z API SimFin a yfinance, díky čemuž je možné kombinovat historická a aktuální data. Taky byla přidána i možnost sledovat aktuální ekonomické zprávy, což uživatelům umožňuje získat širší kontext pro rozhodování. Součástí aplikace je rovněž zpětné testování strategií a sdílení portfolií mezi uživateli.

Motivací pro vytvoření této aplikace bylo nejen vyřešit nedostatky stávajících nástrojů, ale také lépe pochopit, jak investiční trh funguje. Tento projekt mi umožnil nejen prohloubit mé technické dovednosti, ale také získat nové znalosti v oblasti investování a finanční analýzy.

1 TEORETICKÁ ČÁST

1.1 Co je to akcie

Akcie představuje podíl na vlastnictví společnosti, který držitele opravňuje k účasti na jejím zisku (např. prostřednictvím dividend) a hlasování na valné hromadě. Akciové trhy umožňují obchodování s akciemi, jejichž cena je určována nabídkou a poptávkou. Akcie jsou klíčovým nástrojem pro budování kapitálu jak pro společnosti, tak pro investory.

1.1.1 Výhody investování do akcií

Investování do akcií přináší různé výhody, mezi které patří:

- **Možnost zhodnocení kapitálu:** Dlouhodobé investování do akcií může přinést vyšší výnosy ve srovnání s jinými typy aktiv.
- **Pasivní příjmy:** Prostřednictvím dividend mohou investoři získávat pravidelné příjmy.
- **Diversifikace portfolia:** Akcie umožňují investorům diverzifikovat rizika napříč různými sektory a regiony.

1.2 Principy zpětného testování akcií

Zpětné testování (backtesting) je metoda, která simuluje výkonnost investiční strategie na základě historických dat.

1.2.1 Hlavní kroky zpětného testování

- **Výběr historických dat:** Zahrnuje získání dat o cenách akcií, objemech obchodů a finančních ukazatelích.
- **Definování strategie:** Stanovení pravidel pro nákup a prodej akcií.
- **Simulace obchodování:** Aplikace strategie na historická data za účelem analýzy.
- **Vyhodnocení výsledků:** Analýza výnosů, rizik a dalších metrik strategie.

1.2.2 Výhody zpětného testování

- **Ověření strategie:** Umožňuje otestovat výkonnost strategie před jejím nasazením do praxe.
- **Identifikace rizik:** Odhaluje slabiny strategie.
- **Optimalizace:** Pomáhá doladit parametry strategie pro lepší výsledky.

1.3 Databázový model

Databázový model aplikace je klíčový pro ukládání a správu dat o akciích, uživatelích a jejich portfoliích.

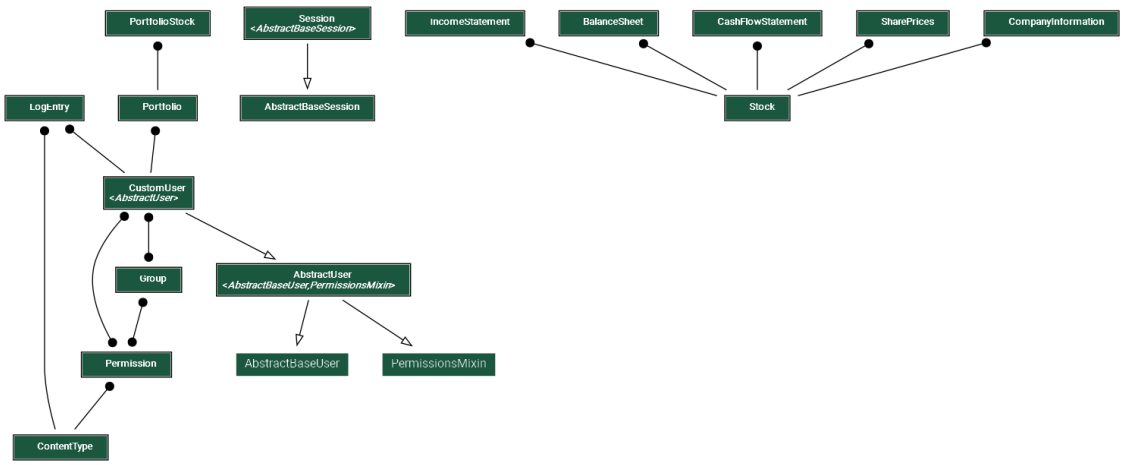
1.3.1 Architektura databáze

Databáze je postavena na relačním modelu s tabulkami propojenými primárními a cizími klíči. Model zahrnuje:

- **Tabulku akcií:** Obsahuje informace o názvu, tickeru, sektoru a dalších vlastnostech.
- **Tabulku historických dat:** Ukládá denní ceny a objemy obchodů.
- **Tabulku portfolií:** Ukládá informace o portfoliích uživatelů a váhách jednotlivých akcií.

1.3.2 Diagram databázového modelu

Diagram znázorňuje vztahy mezi tabulkami a poskytuje přehled o struktuře databáze:



Obrázek 1.1: ER diagram databázového modelu

1.4 Historie zpětného testování

Zpětné testování vzniklo v 70. letech s rozvojem výpočetní techniky. Zpočátku bylo prováděno manuálně, ale s nástupem počítačů a algoritmů se stalo dostupnějším a přesnějším. Dnes umožňuje testovat komplexní strategie na rozsáhlých historických datech.

2 POUŽITÉ TECHNOLOGIE

Tento projekt byl realizován s využitím moderních technologií a nástrojů, které umožnily efektivní vývoj, správu a testování aplikace. Níže jsou popsány klíčové technologie, které byly při vývoji použity.

2.1 Webový framework Django

Django je robustní a škálovatelný framework pro vývoj webových aplikací v Pythonu. Byl zvolen díky:

- **Modulární architektuře**, která usnadňuje rozšiřitelnost aplikace.
- **Integrovanému ORM (Object-Relational Mapping)**, které umožňuje snadnou práci s databází.
- **Vestavěným nástrojům** pro správu uživatelů, zabezpečení a další základní funkce.

Django poskytuje rychlý vývoj, robustní bezpečnostní prvky a čistý kód, což činí aplikaci snadno udržitelnou a rozšiřitelnou.

2.2 Databázový systém SQLite

SQLite je lehký relační databázový systém, který byl zvolen pro tento projekt díky následujícím výhodám:

- **Jednoduché integraci** s Django frameworkem.
- **Nízkým nárokům na údržbu**, jelikož nevyžaduje samostatný databázový server.
- **Přenositelnosti**, což umožňuje snadné sdílení databázových souborů mezi vývojovým a produkčním prostředím.

2.3 Finanční API

Pro zajištění přístupu k historickým a aktuálním finančním datům byly použity následující API:

2.3.1 SimFin API

SimFin API poskytuje historické finanční ukazatele a fundamentální data o společnostech. Tato API nám umožnila získávat strukturovaná data o výkonnosti společností, jako jsou tržby, ziskovost, cash flow a další klíčové ukazatele. Využití tohoto API pomohlo efektivně načítat data o akciích pro zpětné testování investičních strategií a pro analýzu trendů v oblasti financí.

2.3.2 yFinance API

yFinance API poskytuje aktuální tržní data o akciích, včetně cen akcií, objemů obchodů, dividendových výnosů a dalších informací. Tato API se ukázala jako cenný nástroj pro získávání živých dat pro analýzu a zobrazení aktuálního stavu trhů.

2.3.3 Finnhub API

Finnhub API poskytuje širokou škálu dat, včetně fundamentálních údajů, historických cen, tržních zpráv a dalších finančních ukazatelů. Tento nástroj byl použit pro získávání podrobných informací o akciových titulech a pro poskytnutí širšího spektra dat pro analýzu výkonnosti a strategické plánování.

2.4 Vizualizace dat pomocí Plotly

Pro vizualizaci finančních dat a výsledků zpětného testování byla použita knihovna **Plotly**. Tato knihovna umožňuje vytvářet interaktivní grafy a diagramy, které jsou užitečné pro analýzu historických cen akcií, porovnání různých strategií a zobrazení výkonnosti portfolia. Výhody použití Plotly zahrnují:

- **Interaktivitu**, která umožňuje uživatelům snadno zkoumat data kliknutím a přiblížením.
- **Různé typy grafů**, včetně časových řad, histogramů a scatter plotů.
- Možnost **dynamického přizpůsobení grafů** na základě uživatelských preferencí.

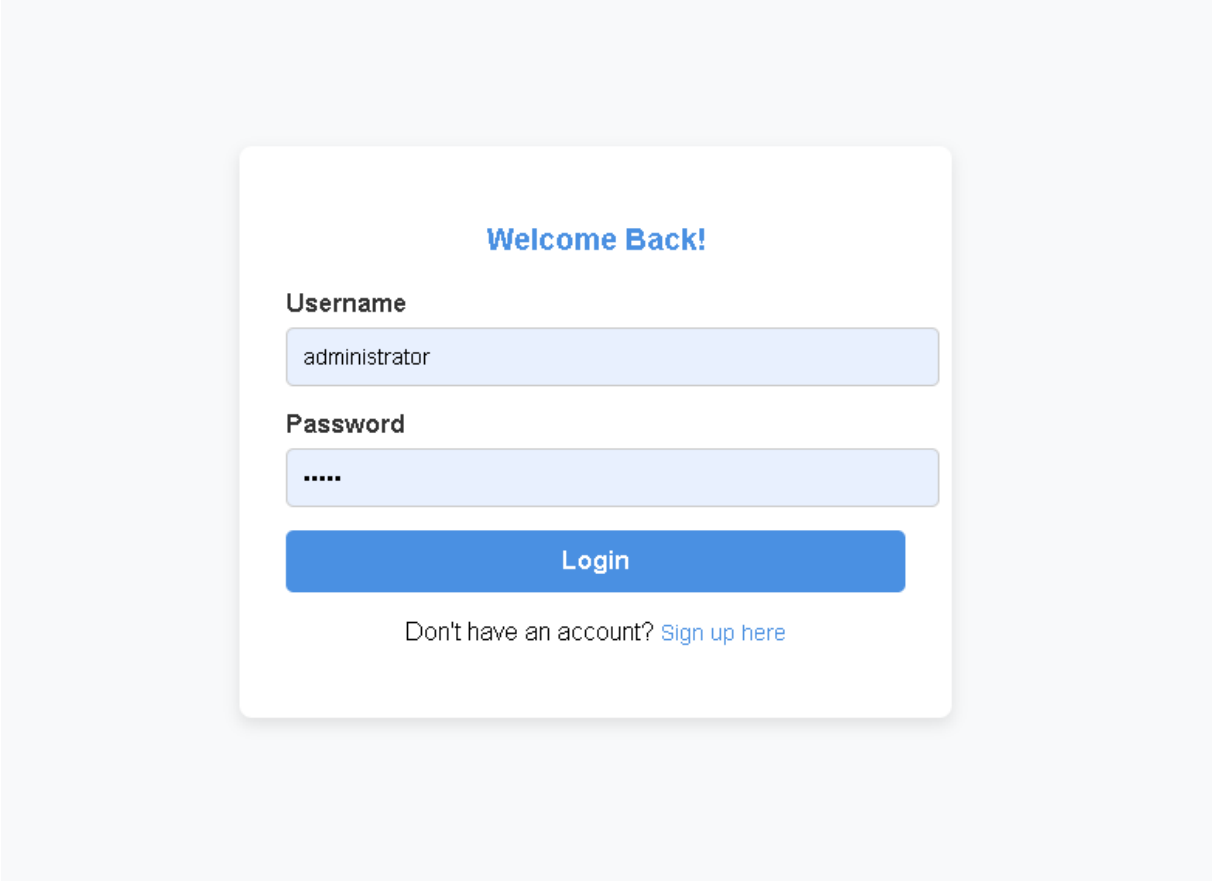
3 NÁVRH ŘEŠENÍ

3.1 Frontend

Frontend aplikace je navržen s cílem poskytnout uživatelům přehledné, intuitivní a interaktivní prostředí pro správu jejich portfolií a analýzu akcií. Důraz byl kladen na jednoduchost uživatelského rozhraní, které umožní snadnou navigaci mezi různými funkcemi aplikace.

3.1.1 Přihlašování uživatele

Uživatelé se do aplikace přihlašují pomocí uživatelského jména a hesla. Tento proces je implementován s využitím Django autentizačního systému. Po úspěšném přihlášení uživatel získá přístup ke svému portfoliu, kde může spravovat akcie a jejich váhy.

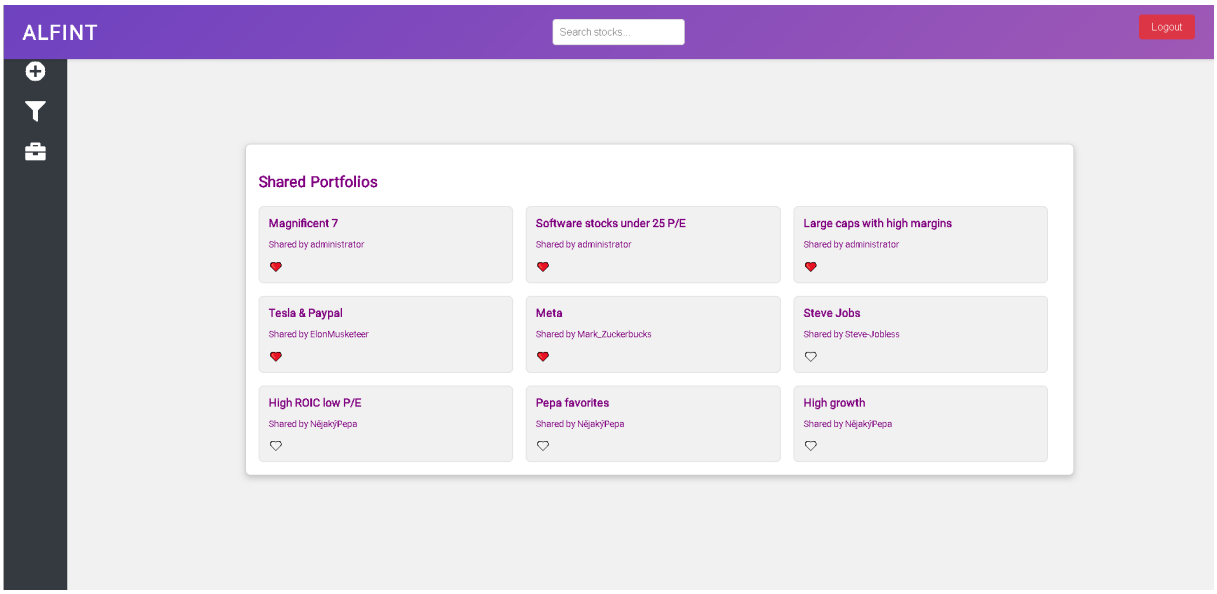


Obrázek 3.1: Login

3.1.2 Sdílení portfolia

Jednou z klíčových funkcí aplikace je možnost sdílení portfolia mezi uživateli. Uživatel může označit své portfolio jako veřejné, což umožní ostatním uživatelům podívat se na jeho výběr akcií. Tento proces je implementován pomocí tlačítka „Sdílet portfolio“, které mění stav portfolia

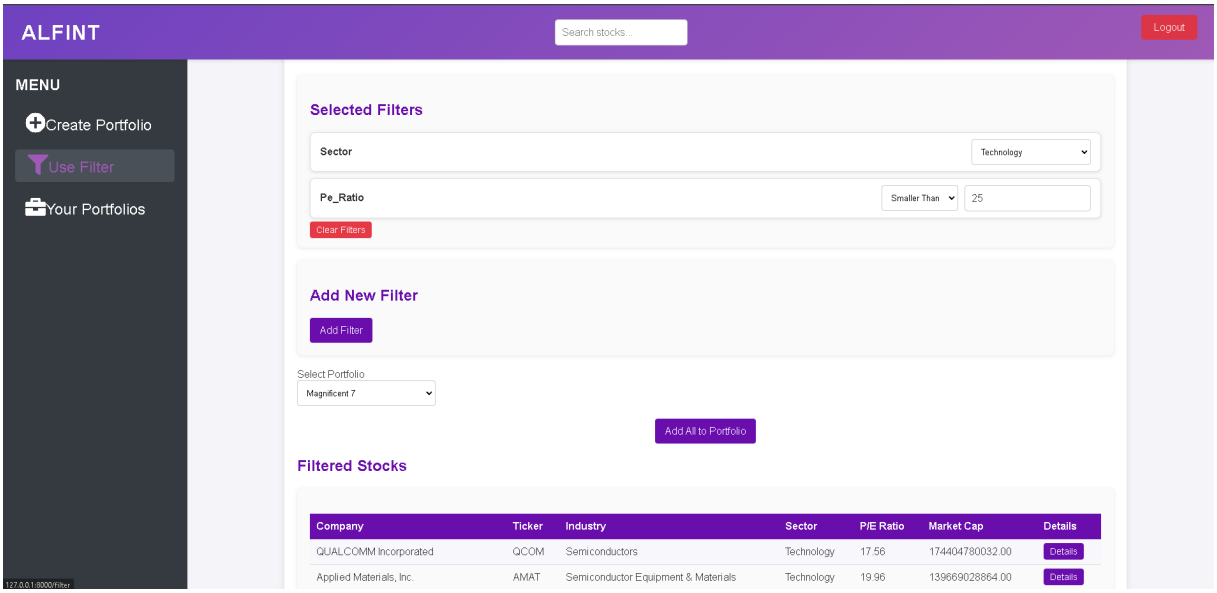
na veřejný. Po sdílení portfolia je možné získat odkaz pro jeho zobrazení.



Obrázek 3.2: Sdílení portfolií

3.1.3 Filtr akcií

Frontend obsahuje filtrování akcií podle různých parametrů, jako je sektor, průmysl, a finanční ukazatele. Filtr se nachází v bočním panelu a umožňuje uživatelům rychle najít akcie, které odpovídají jejich kritériím. Po aplikaci filtrů se výsledky automaticky aktualizují.



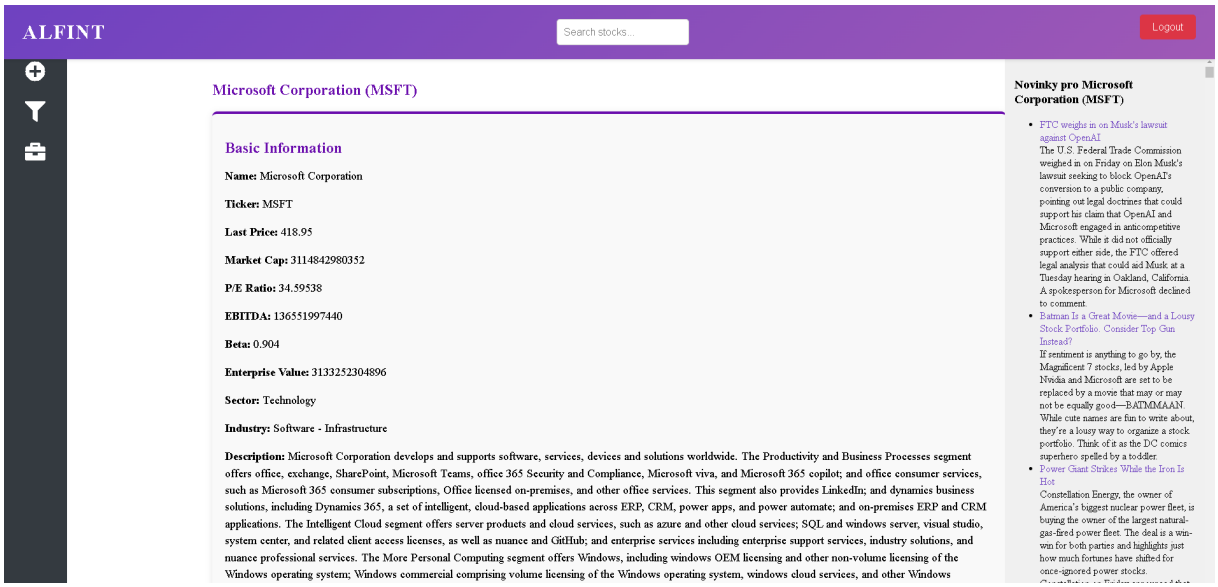
Obrázek 3.3: Filtrování akcií

3.1.4 Vyhledávání akcií

Vyhledávání akcií je klíčovou funkcí aplikace, která umožňuje uživatelům rychle najít konkrétní akcie podle tickeru nebo názvu společnosti. Po zadání názvu nebo tickeru se na stránce automaticky zobrazí výsledky, které odpovídají zadanému hledanému výrazu.

3.1.5 Detail akcie

Stránka s detailem akcie poskytuje uživatelům všechny relevantní informace o konkrétní akcii, včetně historických cen, finančních ukazatelů, grafů a nejnovějších zpráv týkající se dané akcie. Grafy ukazují vývoj vybraných finančních ukazatelů v průběhu času, což pomáhá uživatelům při analýze výkonu akcie.



Obrázek 3.4: Detail akcie

3.1.6 Grafy a vizualizace

Aplikace používá grafy pro zobrazení historických dat akcií, jako jsou ceny, P/E poměr, ROE, a další indikátory. Grafy jsou interaktivní a umožňují uživatelům detailně analyzovat trendy a historické výkony jednotlivých akcií.

3.2 Backend

Backend aplikace je zodpovědný za správu dat, autentizaci uživatelů, a logiku aplikace. Používá Django jako framework pro vývoj webových aplikací a Python pro zpracování dat.

Obrázek 3.5: Graf

3.2.1 Přihlašování uživatele

Backend je zodpovědný za autentizaci uživatele pomocí Django autentizačního systému. Po přihlášení je uživatelovi přiřazena relace, která uchovává informace o jeho stavu přihlášení. Kód pro přihlášení uživatele vypadá takto:

```
def signup(request):
    if request.method == 'POST':
        form = CustomUserCreationForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)
            return redirect('main_page')
    else:
        form = CustomUserCreationForm()
    return render(request, 'signup.html', {'form': form})
```

Kód 3.1: Kód pro přihlášení uživatele

3.2.2 Správa portfolií

Backend umožňuje správu portfolií, která obsahují akcie a jejich váhy. Uživatelé mohou vytvářet nová portfolia, přidávat akcie a měnit jejich váhy. Tento proces je řízen pomocí API, které komunikuje s databází.

```
@login_required
def view_portfolio(request, portfolio_id):
    portfolio = get_object_or_404(Portfolio, id=portfolio_id)
    portfolio_stocks = PortfolioStock.objects.filter(portfolio=portfolio)

```

```
    # Výpočet vážené návratnosti portfolia
    avg_return = calculate_portfolio_return(portfolio)

    # Načítání a příprava dat pro graf
    prices_data = {}
    stocks = []
    for stock_item in portfolio_stocks:
        stock = Stock.objects.filter(ticker=stock_item.ticker).first()
        if not stock:
            continue

        share_prices = SharePrices.objects.filter(stock).order_by('date')
        stock_return = None
        if share_prices.exists():
            first_price = share_prices.first().close_price
            last_price = share_prices.last().close_price
            if first_price and last_price and first_price != 0:
        stock_return = ((last_price - first_price) / first_price) * 100

        stocks.append({
            'stock': stock,
            'return': stock_return,
            'weight': stock_item.weight,  # Váha akcie
        })
```

Kód 3.2: Kód pro správu portfolií

3.2.3 Filtrace akcií

Backend poskytuje funkci pro filtrování akcií podle zadaných parametrů. Filtrované výsledky se následně vrátí na frontend a zobrazí uživatelovi.

```
    for f in filters:
        field = f.get('field')
        operator = f.get('operator', 'gte')
        value = f.get('value')

        if field and value:
            try:
                if field in ['sector', 'industry']:
```

```
                    q_object &= Q(**{f"{field}__iexact": value})
            else:
                q_object &= Q(**{f"{field}__{operator}": float(value)})
            except (ValueError, TypeError):
                continue
    stocks = Stock.objects.filter(q_object).order_by('-market_cap')
```

Kód 3.3: Kód pro filtr akcií

3.2.4 Získávání dat z externích API

Aplikace získává data o akciích a jejich finančních ukazatelích z externích API, jako je SimFin a yfinance. Backend zajišťuje komunikaci s těmito API, načítání dat a jejich ukládání do databáze. Kód pro načítání dat z API vypadá následovně:

```
ef load_simfin_data():
    print("Starting to load SimFin data...")

    try:
        # Načtení dat z API SimFin
    income_data = sf.load(dataset='income', variant='annual', market='us',
    index=['Ticker', 'Fiscal Year'])

    balance_data = sf.load(dataset='balance', variant='annual',
    market='us', index=['Ticker', 'Fiscal Year'])

    cashflow_data = sf.load(dataset='cashflow', variant='annual',
    market='us', index=['Ticker', 'Fiscal Year'])

    shareprices_data = sf.load(dataset='shareprices', variant='daily',
    market='us', index=['Ticker', 'Date'])

        tickers = income_data.index.get_level_values('Ticker').unique()
```

Kód 3.4: Kód pro získávání dat z API

3.2.5 Výpočet finančních ukazatelů

Po získání dat z API backend vypočítá klíčové finanční ukazatele, jako je P/E poměr, ROE, atd. Tyto ukazatele jsou následně uloženy v databázi pro pozdější analýzu.

```
def calculate_ratios():
    print("Calculating financial ratios...")
    stocks = Stock.objects.all()
    for stock in stocks:
        try:
            # Načtení finančních dat
    income_statement = IncomeStatement.objects.filter(stock=stock)
    balance_sheet = BalanceSheet.objects.filter(stock=stock)
    cash_flow_statement = CashFlowStatement.objects.filter(stock=stock)

    if income_statement and balance_sheet:
    # P/E Ratio
if stock.market_cap and income_statement.net_income
and income_statement.net_income != 0:
stock.pe_ratio = round(stock.market_cap / income_statement.net_income, 2)
                    ...
```

Kód 3.5: Kód pro výpočet finančních ukazatelů

3.2.6 Grafy a vizualizace

Backend zajišťuje generování dat pro grafy, které jsou následně odesílány na frontend. K tomu využívá knihovny jako Matplotlib nebo Plotly pro generování vizualizací.

```
def create_price_chart(close_prices):
    dates = [price['date'] for price in close_prices]
    prices = [price['close_price'] for price in close_prices]

    # Vytvoření grafu
    fig, ax = plt.subplots()
    ax.plot(dates, prices, label='Close Price')

    ax.set(xlabel='Date', ylabel='Close Price',
           title='Stock Price Over Time')
    ax.grid()

    buf = BytesIO()
    plt.savefig(buf, format='png')
    buf.seek(0)
    image_base64 = base64.b64encode(buf.getvalue()).decode('utf-8')
```

```
17      buf.close()
18
19      return image_base64
```

Kód 3.6: Kód pro generování grafů

4 ZÁVĚR A ZHODNOCENÍ

4.1 Shrnutí výsledků

Cílem této práce bylo vyvinout aplikaci pro zpětné testování akcií, která umožňuje uživatelům testovat investiční strategie na historických datech a analyzovat výkonnost akcií na základě různých finančních ukazatelů. Po realizaci a testování aplikace lze shrnout následující výsledky:

Aplikace úspěšně integruje dvě klíčová externí API, SimFin a yfinance, která slouží k získávání historických dat a finančních ukazatelů. Mezi těmito ukazateli jsou například P/E, ROE, ROA, a další, které jsou správně vypočítány a uloženy v databázi pro následné použití v analýzách.

Uživatelé mají možnost definovat vlastní investiční strategie a provádět zpětné testování na historických datech, přičemž výsledky jsou generovány a vizualizovány pomocí interaktivních grafů. Tyto grafy zlepšují analýzu a porozumění výsledkům zpětného testování. Uživatelé mohou přidávat akcie do portfolia buď ručně, nebo pomocí filtru, který umožňuje výběr akcií podle různých kritérií.

Dalšími funkcemi aplikace jsou detailní zobrazení akcií, které zahrnuje historické cenové grafy, základní finanční ukazatele, výpočty jako P/E, ROIC, PEG, EPS, a další. V případě, že některé údaje nejsou k dispozici, aplikace správně zobrazuje hodnotu None místo N/A. Uživatelé mohou také přidávat váhy jednotlivým akciím v portfoliu a grafy se automaticky aktualizují podle těchto váh. Navíc je možné přidávat „srdíčko“ k portfoliím, což je jednoduché označení oblíbených portfolií.

Aplikace také umožňuje uživatelům sdílet portfolia, což poskytuje možnost spolupráce a výměny investičních strategií s ostatními uživateli.

Aplikace tedy splňuje své základní cíle a poskytuje funkční nástroj pro zpětné testování akcií a analýzu jejich výkonnosti.

4.2 Možnosti budoucího vývoje

Vzhledem k tomu, že aplikace byla navržena s ohledem na rozšiřitelnost a flexibilitu, existuje několik možností pro její budoucí vylepšení. K těmto možnostem patří:

- **Predikce pomocí AI a strojového učení** – Integrace modelů strojového učení pro predikci budoucího vývoje cen akcií na základě historických dat, klíčových finančních ukazatelů a dalších relevantních faktorů. To by uživatelům pomohlo lépe odhadnout potenciální výnosy a rizika.
- **Kalendář ekonomických událostí** – Přidání interaktivního kalendáře obsahujícího významné ekonomické události, jako jsou zveřejnění výsledků společností, makroekono-

mická data nebo rozhodnutí centrálních bank. Tato funkcionalita by uživatelům poskytla lepší přehled o možných rizicích a příležitostech.

- **Pokročilé analýzy portfolia** – Zavedení nástrojů pro analýzu portfolia, jako je sledování celkové volatility, Sharpeho poměru nebo dalších ukazatelů, které by pomohly uživatelům optimalizovat jejich investice.
- **Sledování sentimentu trhu** – Možnost analyzovat sentiment trhu pomocí zpracování textů (např. z finančních článků a zpráv). Tato funkcionalita by uživatelům pomohla identifikovat, zda převládá pozitivní, neutrální nebo negativní nálada ohledně konkrétní akcie či trhu obecně.
- **Komunitní diskuse a hodnocení** – Integrované fórum nebo sekce pro komentáře by umožnily uživatelům diskutovat o investičních strategiích, ekonomických událostech nebo konkrétních akciích. Možnost hodnotit portfolia a strategie by dále podpořila interakci mezi uživateli.

Aplikace má také potenciál pro komerční využití, zejména pokud bude rozšířena o pokročilé funkce analýzy a optimalizace investičních strategií.

4.3 Závěr

Tato práce se zaměřila na vytvoření aplikace pro zpětné testování akcií, která by poskytovala nástroje pro analýzu akciových trhů na základě historických dat a finančních ukazatelů. Aplikace byla úspěšně navržena, implementována a otestována. Testování prokázalo správnost výpočtů a funkčnost aplikace i při velkém objemu dat.

Díky implementaci flexibilních analytických nástrojů a vizualizací má aplikace vysoký potenciál pro využití v praxi a pro rozšíření o nové funkce v budoucnu. Tento projekt tak představuje silný základ pro další vývoj nástrojů pro analýzu a testování akciových strategií.

Zdrojový kód projektu je dostupný na GitHubu (https://github.com/Vojtik1/Alfint)

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

- **Django Documentation** – Django Software Foundation. Dostupné z: `https://www.djangoproject.com/`
- **yfinance Documentation** – Yahoo Finance API for Python. Dostupné z: `https://pypi.org/project/yfinance/`
- **SimFin API Documentation** – SimFin. Dostupné z: `https://simfin.com/docs`
- **The Intelligent Investor** – Benjamin Graham, HarperBusiness, 2003.
- **Plotly: A Comprehensive Guide** – Plotly Technologies Inc. Dostupné z: `https://plotly.com/python/`
- **ChatGPT** - ChatGPT. Dostupné z: `https://chatgpt.com/`
- **Stack Overflow** - Stack Overflow. Dostupné z: `https://stackoverflow.com/`