

# XD PROJEKT – Dokumentacja aplikacji

Dariusz Kołodziejczyk, Sebastian Bek, Mikołaj Maliszewski

14 grudnia 2025

## 1 Krótki opis

Aplikacja służy do śledzenia cen akcji CD PROJEKT S.A. w wybranych okresach czasu, umożliwia prezentację wykresów oraz bieżącej ceny papierów wartościowych.

## 2 Cele i założenia projektu

Głównym celem aplikacji jest stworzenie narzędzia umożliwiającego użytkownikowi:

- bieżące monitorowanie kursu akcji CD Projekt S.A.,
- analizę zmian wartości w wybranych okresach czasu,
- zestawienie kursu z poziomem zainteresowania w Internecie (Google Trends),
- pobieranie podstawowych dokumentów finansowych udostępnianych przez spółkę.

## 3 Środowisko i wymagania

Aplikacja została stworzona w języku Python w wersji 3.11.9 i działa w środowisku graficznym opartym na bibliotece CustomTkinter.

Do prawidłowego działania wymagane są następujące biblioteki:

- `yfinance` – pobieranie danych giełdowych z Yahoo Finance,
- `matplotlib` – generowanie wykresów kursu,
- `Pillow` – obsługa grafiki,
- `customtkinter` – tworzenie nowoczesnego interfejsu użytkownika,
- `CTkMessageBox` – obsługa okien komunikatów,
- `tkcalendar` – wybór zakresu dat,
- `pytrends` – komunikacja z Google Trends.
- `openpyxl` - otwieranie plików XSLX.
- `pandas` - otwieranie i działanie na plikach XSLX.

- `pytest` - biblioteka obsługująca testy

Instalacja pakietów za pomocą pliku `req.txt`:

```
pip install -r req.txt
```

## 4 Opis funkcjonalności

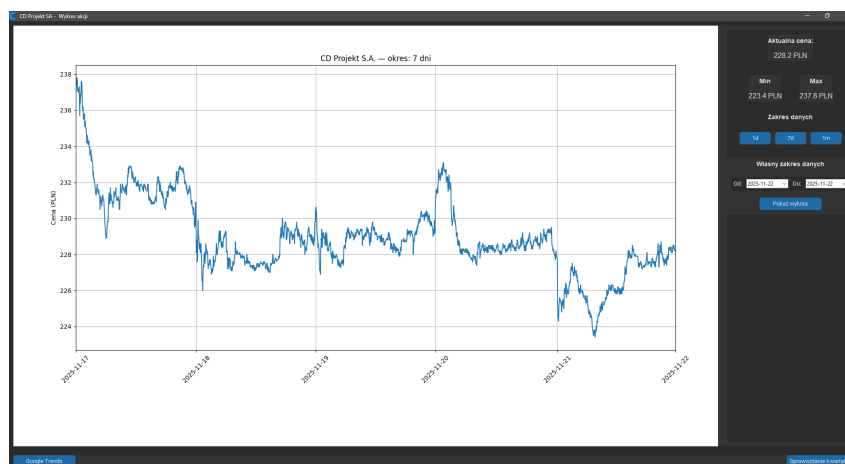
Aplikacja realizuje następujące funkcje:

- pobieranie aktualnej ceny akcji CD Projekt S.A.
- wyświetlanie wykresów kursu z zakresu:
  - ostatniego dnia (1d)
  - ostatniego tygodnia (7d)
  - ostatniego miesiąca (1m)
  - dowolnego zakresu wybranego z kalendarza
- pobieranie i prezentacja wykresów z Google Trends z zakresu:
  - ostatniego dnia (1d)
  - ostatniego tygodnia (7d)
  - ostatniego miesiąca (1m)
- możliwość wyboru słowa kluczowego (np. "CD Projekt", "Cyberpunk 2077", "Wiedźmin")
- możliwość pobierania raportów finansowych spółki w formacie PDF lub XLSX
- wyświetlanie tabeli zawierającej najważniejsze dane liczbowe na temat wybranego kwartału
- zapisywanie pobranych danych w pamięci podręcznej (cache), aby zmniejszyć liczbę zapytań do API

## 5 Zdjęcia aplikacji

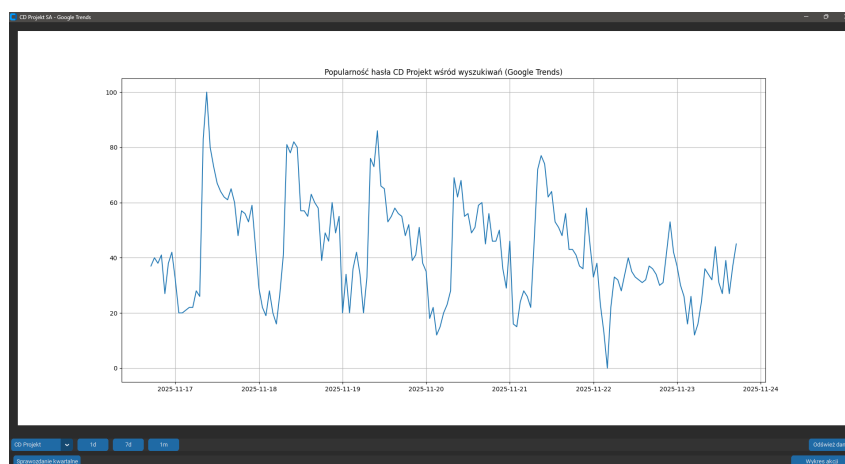
Poniżej przedstawiono screeny pokazujące 3 ekrany aplikacji

## 5.1 Ekran1: Wykres akcji



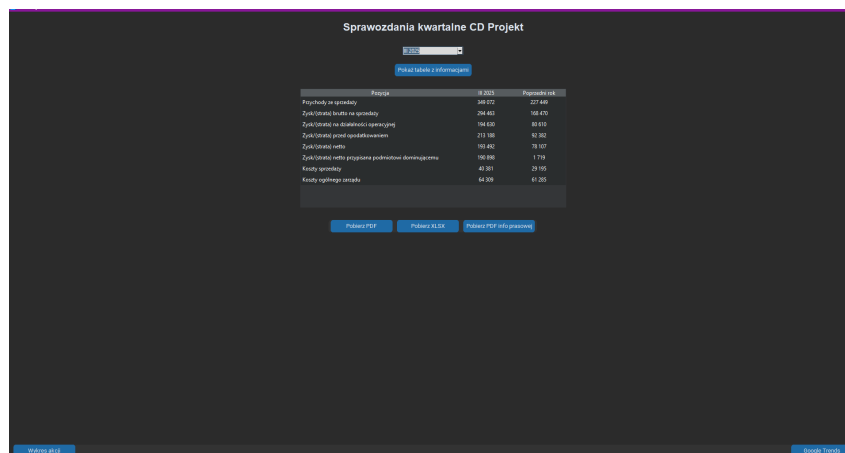
Rysunek 1: Ekran do wyświetlania wykresów akcji CD Projekt S.A. gdzie użytkownik może wybierać ich zakres

## 5.2 Ekran2: Wykres Google Trends



Rysunek 2: Ekran do wyświetlania wykresów Google Trends gdzie użytkownik może wybierać ich zakres i słowo kluczowe

## 5.3 Ekran3: Obsługa sprawozdań kwartalnych



Rysunek 3: Ekran do zarządzania sprawozdaniami kwartalnymi, gdzie użytkownik może je pobrać z wybranego kwartału

## 6 Struktura kodu

Aplikacja podzielona jest na moduły logiczne zgodnie z zasadą rozdziału odpowiedzialności.

### 6.1 Główna klasa aplikacji – App

Klasa ta inicjalizuje wszystkie ekrany aplikacji oraz zarządza ich wyświetlaniem. Po uruchomieniu programu użytkownik widzi ekran wykresu kursu akcji.

### 6.2 Ekran wykresów giełdowych – Screen1

Odpowiada za:

- wyświetlanie wykresu kursu,
- wybór zakresu danych (1 dzień, 1 tydzień, 1 miesiąc),
- podawanie zakresu użytkownika za pomocą kalendarza,
- aktualizację:
  - bieżącej ceny akcji,
  - minimalnej i maksymalnej ceny w zakresie.

### 6.3 Ekran Google Trends – Screen2

Udostępnia:

- wybór słowa kluczowego (np. CD Projekt, Wiedźmin),

- wygenerowanie wykresu popularności,
- odświeżanie danych,
- wykorzystanie cache w celu minimalizacji obciążenia API.

## 6.4 Ekran sprawozdań – Screen3

Pozwala na:

- wybór kwartału z listy raportów,
- pobranie:
  - raportu finansowego PDF,
  - raportu XLSX z wynikami finansowymi,
  - informacji prasowej w PDF.
- wyświetlenie tabeli z najważniejszymi danymi na temat danego kwartału

## 6.5 Moduł danych – CDPdata

Zawiera logikę:

- pobierania danych z Yahoo Finance,
- obsługi Google Trends poprzez PyTrends,
- zapisu i wczytywania cache w formacie JSON.

## 6.6 Moduł wykresów – CDPplot

Odpowiada za renderowanie wykresów biblioteki matplotlib i osadzanie ich w komponentach interfejsu.

## 6.7 SplashScreen

Odpowiada za wyświetlanie startowego ekranu i ładowanie danych przed uruchomieniem aplikacji.

# 7 Opisy funkcji

### CDPdata

- **GetCdpData(period)** - pobiera dane o kursie z serwisu Yahoo Finance (yfinance) dla danego okresu
- **getCustomCDPData(start\_time, end\_time)** - pobiera dane o kursie dla wybranego przez użytkownika zakresu
- **GetCurrentPrice()** - pobiera i wyświetla aktualną cenę akcji

- **getMinMaxPrice(data)** - pobiera najniższą i najwyższą cenę w danym zakresie
- **ensure\_\_cache\_\_dir()** - upewnia się że istnieje folder plików JSON jeśli nie tworzy go
- **load\_\_cache()** - wczytuje cache a jeśli plik jest uszkodzony zwraca pusty
- **save\_\_cache(cache)** - zapisuje dane do pliku JSON
- **invalidate\_\_trends\_\_plot(keyword, period)** - usuwa cache przy odświeżaniu wykresów
- **timeframe\_\_for(period)** - zamienia nazwy okresów na te używane przez pytrends
- **df\_\_from\_\_entry(entry)** - tworzy DataFrame ze wczytanego cache
- **getTrendsData(period)** - wczytuje dane z pliku JSON a jeśli nie ma pobiera dane z Google Trends
- **choose\_\_folder()** - wyświetla okno wyboru folderu do jakiego ma zostać zapisany wybrany przez użytkownika plik
- **download\_\_file(url: str, default\_\_name: str)** - pobiera plik z internetu do wybranego przez użytkownika folderu

## CDPplot

- **CreateCdpPlot(frame, period)** - tworzy i wyświetla wykres cen
- **createCustomDataCdpPlot(frame, start\_\_date, end\_\_date, data)** - tworzy wykres dla wybranego przez użytkownika zakresu
- **createTrendsPlot(frame, period, data, keyword)** - tworzy wykres zainteresowania z Google Trends

## SplashScreen

- **\_\_init\_\_(self)** - ustawia wysokość szerokość i zawartość splash screena
- **center\_\_x(self, window\_\_width), center\_\_y(self, window\_\_width)** - pomocnicze funkcje do ustawienia splash screena na środku ekranu
- **load\_\_data\_\_with\_\_delay(self)** - pobiera dane podczas splash screena i czeka co najmniej 2,5 sekundy
- **open\_\_main\_\_app(self, data)** - zamyka splash screen i otwiera główne okno aplikacji

## CDPQuarter

- **getQuarterTableData(selected\_\_quarter)** - wyciąga konkretne dane z pliku XLSX sprawozdania kwartalnego dla określonego kwartału i analogicznego kwartału z poprzedniego roku

## App

- **\_\_init\_\_(self, master, preloaded\_data, preloaded\_trends)** - ustawia wysokość szerokość i zawartość głównego okna aplikacji
- **showFrame(self, frame, text)** - zmienia widoczny ekran i nazwę aplikacji

## Screen1

- **\_\_init\_\_(self, master, preloaded\_data, preloaded\_trends)** - ustawia ramki okna wykresu akcji
- **showPlot(self, period)** - aktualizuje wykres po kliknięciu któregoś z przycisków
- **showCustomDatePlot(self)** - aktualizuje wykres po wpisaniu własnego zakresu
- **updatePriceLabel(self)** - aktualizuje cenę po kliknięciu przycisku
- **updateMinMaxLabels(self, data)** - aktualizuje min i max ceny po kliknięciu przycisku
- **showError(self, title, message, icon)** - pokazuje okienko z błędem

## Screen2

- **\_\_init\_\_(self, master, preloaded\_data, preloaded\_trends)** - ustawia ramki okna wykresów Google Trends
- **showTrendsPlot(self, period)** - tworzy wykres dla Google Trends z określonego zakresu
- **refreshTrends(self)** - odświeża dane używane do rysowania wykresu Google Trends

## Screen3

- **\_\_init\_\_(self, master, preloaded\_data, preloaded\_trends)** - ustawia ramki okna służącego do obsługi sprawozdań kwartalnych
- **onQuarterButtonClick(self)** - wyświetla tabelę z danymi po kliknięciu przycisku
- **showQuarterTable(self, extracted, selected\_quarter)** - tworzy tabelę z zebranych danych
- **download\_pdf(self)** - pobiera plik PDF
- **download\_xlsx(self)** - pobiera plik XLSX
- **download\_press\_pdf(self)** - pobiera dane prasowe w formacie PDF
- **showError(self, title, message, icon)** - pokazuje okienko z błędem

## 8 Zagrożenia

### 8.1 Błąd 429 Google Trends

Wynika z zbyt dużej liczby zapytań do serwisu Google Trends w krótkim czasie.

**Rozwiązanie:** korzystanie z zapisanych danych w JSON (cache) oraz ograniczenie liczby zapytań

### 8.2 Blokowanie Yahoo Finance

Serwis Yahoo Finance wprowadza dzienne limity około 2000 zapytań na IP

**Rozwiązanie:** ograniczanie częstotliwości zapytań, cache danych historycznych oraz stosowanie interwałów

### 8.3 Obsługa plików JSON (cache)

Możliwe błędy serializacji i deserializacji, np. uszkodzony plik JSON lub niepoprawny format danych

**Rozwiązanie:** obsługa wyjątków przy wczytywaniu i zapisie JSON, atomowy zapis do pliku tymczasowego, backup uszkodzonego pliku

### 8.4 Path traversal w cache

Niepoprawnie sformatowane słowa kluczowe mogą prowadzić do zapisania/usunięcia plików poza katalogiem cache

**Rozwiązanie:** sanitacja słów kluczowych (tylko litery, cyfry i podkreślenia), użycie `os.path.basename` oraz bezpieczne tworzenie nazw plików

### 8.5 Pobieranie plików z internetu

Pobieranie plików PDF/XLSX bez weryfikacji rozszerzenia, MIME type czy rozmiaru może prowadzić do pobrania złośliwego pliku

**Rozwiązanie:** sprawdzanie typu i rozmiaru pliku, obsługa wyjątków, zapis w bezpiecznym folderze

### 8.6 Brak timeoutów w zapytaniach sieciowych

Funkcje `requests.get`, `yfinance.download` i Google Trends mogą blokować aplikację przy problemach sieciowych

**Rozwiązanie:** ustawienie timeoutów i obsługa wyjątków przy pobieraniu danych

### 8.7 Duże rozmiary danych

Pobieranie dużych zbiorów danych (Google Trends, Yahoo Finance) może prowadzić do problemów z pamięcią

**Rozwiązanie:** ograniczenie liczby pobieranych danych lub filtrowanie niepotrzebnych kolumn



## 8.8 Interfejs użytkownika

Brak pełnej kontroli nad ścieżką folderu wyboru w Tkinter może prowadzić do zapisu w systemowych katalogach

**Rozwiązanie:** domyślny folder w katalogu aplikacji oraz walidacja ścieżki

## 9 Przeprowadzone testy

### 9.1 Testy jednostkowe

#### 9.1.1 Cache (test\_cache.py)

Testy te sprawdzają poprawność działania mechanizmu cache w izolacji od reszty systemu.

- **test\_save\_and\_load\_cache** – weryfikuje, że zapisany cache jest poprawnie odczytywany bez modyfikacji danych.
- **test\_invalid\_json\_moves\_file** – symuluje uszkodzony plik JSON i sprawdza, czy zostaje on przeniesiony, a cache zwraca pusty słownik.
- **test\_invalidate\_removes\_entry** – testuje funkcję unieważniania pojedynczego wpisu cache.

#### 9.1.2 Konwersja danych (test\_df.py)

Testy dotyczą funkcji odpowiedzialnej za konwersję danych JSON do obiektu `DataFrame`.

- **test\_df\_from\_entry\_valid** – poprawna konwersja prawidłowego JSON-a do `DataFrame`.
- **test\_df\_from\_entry\_invalid\_json** – obsługa błędnego JSON-a (zwracane `None`).
- **test\_df\_from\_entry\_empty** – zachowanie funkcji przy pustym wejściu.

### 9.2 Testy integracyjne

#### 9.2.1 Pobieranie danych trendów (test\_gettrends.py)

Testy integracyjne sprawdzające współpracę warstwy cache z logiką pobierania danych (z użyciem atrap API).

- **test\_gettrends\_fetches\_and\_saves** – dane są pobierane z API i zapisywane do cache.
- **test\_gettrends\_uses\_cache** – przy istniejącym cache API nie jest wywoływane.
- **test\_gettrends\_fallback\_on\_error** – w przypadku błędu API następuje fallback do cache.

## 9.3 Testy funkcjonalne (GUI)

### 9.3.1 Ekran trendów (test\_trends\_functionals.py)

Testy funkcjonalne sprawdzające zachowanie interfejsu użytkownika.

- **test\_trends\_screen\_shows\_plot** – sprawdza, czy po akcji użytkownika renderowany jest wykres.
- **test\_refresh\_forces\_reload** – weryfikuje, że odświeżenie wymusza ponowne pobranie danych.
- **test\_trends\_error\_message** – sprawdza wyświetlenie komunikatu błędu w UI.

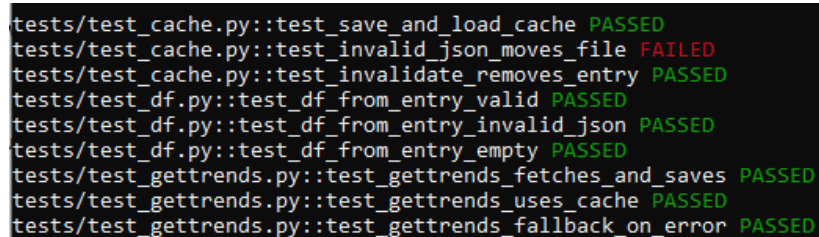
## 9.4 Testy wydajnościowe

### 9.4.1 Wydajność cache (test\_trends\_performance.py)

Test mierzy czas dostępu do danych zapisanych lokalnie w cache.

- **test\_cache\_read\_is\_fast** – weryfikuje, że odczyt dużego DataFrame z cache trwa krócej niż 50 ms.

## 9.5 Screen poprawności testów



```
tests/test_cache.py::test_save_and_load_cache PASSED
tests/test_cache.py::test_invalid_json_moves_file FAILED
tests/test_cache.py::test_invalidate_removes_entry PASSED
tests/test_df.py::test_df_from_entry_valid PASSED
tests/test_df.py::test_df_from_entry_invalid_json PASSED
tests/test_df.py::test_df_from_entry_empty PASSED
tests/test_gettrends.py::test_gettrends_fetches_and_saves PASSED
tests/test_gettrends.py::test_gettrends_uses_cache PASSED
tests/test_gettrends.py::test_gettrends_fallback_on_error PASSED
```

Rysunek 4: Wszystkie testy poza jednym przeszły

# 10 Aktualizacje

## 27.10.2025

- Podział aplikacji na moduły
- Dodanie funkcji do obsługi zakresów dat
- Nowe biblioteki: CtkMessageBox i tkcalendar

## 02.11.2025

- Dodano pytrends
- Obsługa wykresów Google Trends

## **09.11.2025**

- Dodano obsługę zapisu trendów do JSON
- Podział aplikacji na 3 ekrany

## **16.11.2025**

- Dodano wykresy Google Trends dla Cyberpunk 2077 i Wiedźmin
- Dodano pobieranie sprawozdań finansowych CD Projekt SA

## **23.11.2025**

- Poprawienie dokumentacji

## **01.12.2025**

- Dodanie wyświetlania tabeli ze sprawozdania kwartalnego
- Przeprowadzenie testów jednostkowych i integracyjnych

## **08.12.2025**

- Przeprowadzenie testów wydajnościowych i funkcjonalnych