

# XD PROJEKT – Aplikacja w pythonie

## 1. Krótki opis

Aplikacja służy do śledzenia cen akcji CD PROJEKT S. A. w wybranych okresach czasowych w formie wykresu i prezentuje aktualną cenę.

## 2. Środowisko i wymagania

Aplikacja została napisana w pythonie w wersji 3.11.9. Do działania potrzebne są dodatkowo biblioteki/pakiety:

- yfinance
- matplotlib
- Pillow
- Customtkinter
- CtkMessageBox
- Tkcalendar
- Pytrends

Pakiety można zainstalować na raz dzięki plikowi req.txt komendą  
`pip install -r req.txt`

## 3. Opis funkcjonalności

Uruchomienie aplikacji poprzedza splash screen wyświetlający się przez 2,5 sekundy + ewentualny dodatkowy czas potrzebny na pobranie danych do wykresu.

Gdy dane się załadują wyświetla się okno z wykresem i przyciskami do zmiany zakresu. Domyślnie ustawiony zakres to 1 tydzień.

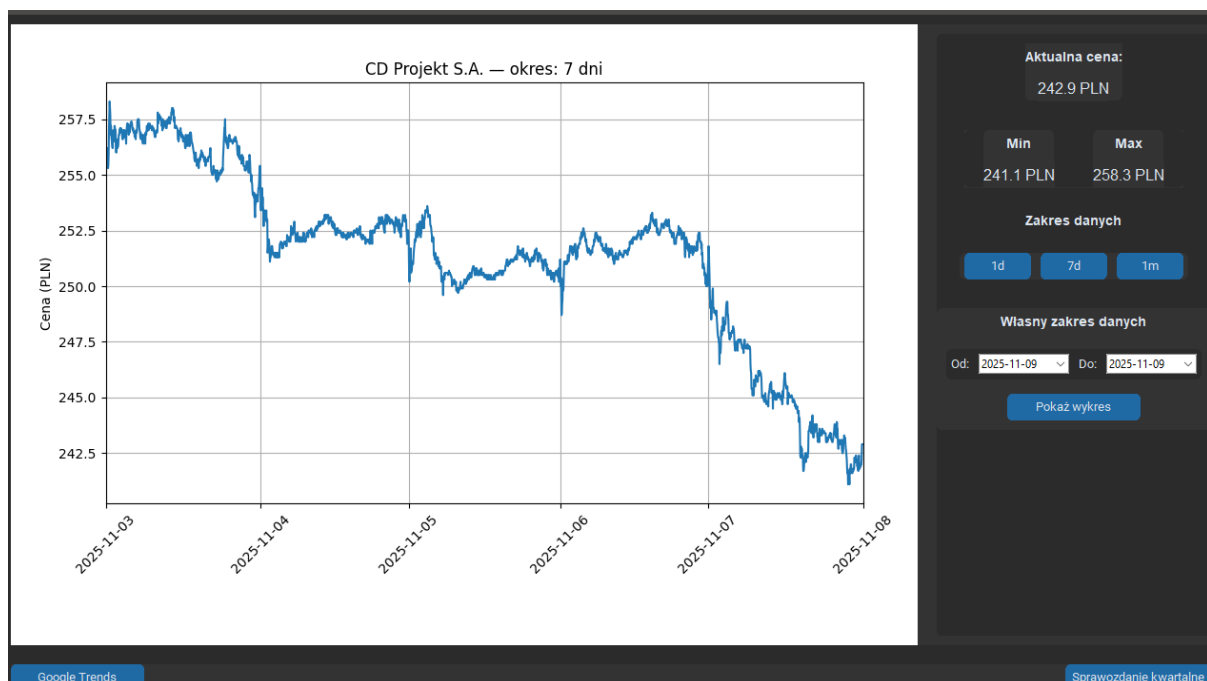
Użytkownik może zmieniać zakres z spośród 3 opcji: 1 dzień, 1 tydzień, 1 miesiąc. Dodatkowo użytkownik może wybrać dowolny zakres dat za pomocą specjalnego menu.

Wyświetlany jest również wykres popularności fraz związanych z CD Projekt SA przy użyciu serwisu Google Trends.

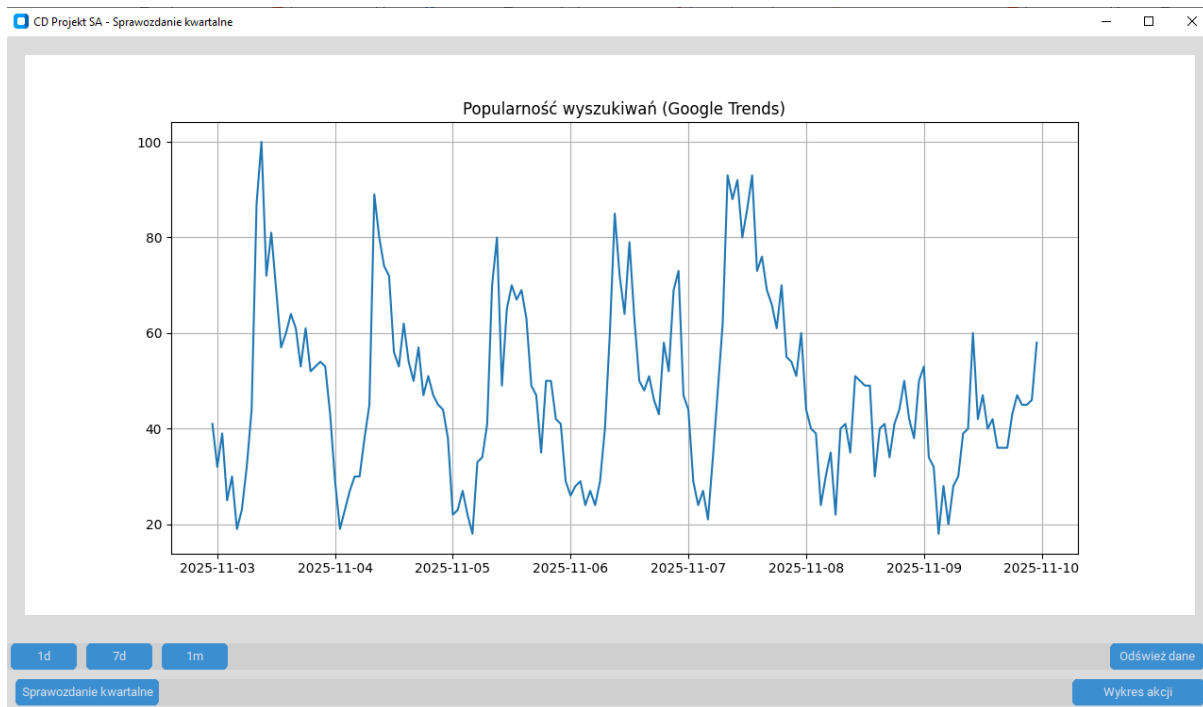
Nad przyciskami wyświetlana jest aktualna cena akcji.



Splash screen



GUI Aplikacji – ekran główny



GUI aplikacji – ekran google trends

## 4. Struktura kodu

### a) Moduły:

#### I. 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 to go tworzy
- `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. Dla czytelności używa powyższych funkcji

## II. CDPplot

- `CreateCdpPlot(frame, period)` - tworzy i wyświetla wykres cen
- `createCustomDataCdpPlot(frame, start_date, end_date, data)` - tworzy i wyświetla wykres dla wybranego zakresu przez użytkownika
- `createTrendsPlot(frame, period, data, keyword)` - tworzy wykres zainteresowania używając danych z serwisu Google Trends dla konkretnego zakresu i słowa kluczowego

## III. 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 zamiast w lewym górnym rogu.
- `load_data_with_delay(self)` - pobiera dane podczas splash screena i jeśli nie minęły 2,5 sekundy, wyświetla go dalej aż miną.

- `open_main_app(self, data)` - zamyka splash screen i otwiera główne okno aplikacji przesyłając jednocześnie pobrane dane

#### IV. 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

#### V. 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 któregoś z przycisków
- `updateMinMaxLabels(self, data)` - aktualizuję min i max ceny po kliknięciu przycisku
- `showError(self, title, message, icon)` - pokazuje okienko z błędem

#### VI. Screen2:

- `__init__(self, master, preloaded_data, preloaded_trends)`
  - ustawia 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

### 5. Aktualizacje

#### I. 27.10.2025

- Rodzielono główny plik na moduły CDPdata (Pobieranie danych), CDPplot (Tworzenie wykresów), Splash i Main.
- Nowe funkcje w module CDPdata:  
getCustomCDPData(start\_time, end\_time),  
getMinMaxPrice(data)
- Nowe funkcje w module CDPplot:  
createCustomDataCdpPlot(frame, start\_date, end\_date, data)
- Nowe funkcje w App: showCustomDatePlot(self) i  
updateMinMaxLabels(self, data)
- Nowe biblioteki: CtkMessageBox i tkcalendar
- Dodano możliwość ustawiania własnego zakresu dat

#### II. 02.11.2025

- Nowa biblioteka: pytrends
- Naprawienie błędów dotyczących własnego ustawiania zakresu dat
- Dodanie funkcjonalności wyświetlania wykresu używającego danych z serwisu Google Trends

#### III. 09.11.2025

- Nowa biblioteka: json
- Podzielenie głównej aplikacji na 3 ekrany (wykres kursu giełdowego, Google Trends, podsumowanie sprawozdania kwartalnego)
- Zmienienie działania Google Trends na zapisywanie w pliku json

#### IV. 16.11.2025

- Dodanie wykresów Trendów dla marek należących do CD Projekt SA (Cyberpunk 2077 i Wiedźmin)

## 6. Opis zagrożeń

### a) Błąd 429 dla API Google Trends

- Powodowany jest wysyłaniem zbyt dużej ilości żądań, ale małej ilości może wystąpić
- Rozwiązaniem jest zapisywanie danych w pliku JSON, by w przypadku zablokowania użytkownika nadal mógł korzystać z funkcji tworzenia wykresów dla starszych danych

### b) Blokowanie ze strony API Yahoo Finances

- Powodowany zbyt dużej ilości żądań do API Yahoo Finances
- Limit dzienny to ok. 2000, więc nie powinno to wpływać negatywnie na korzystanie z aplikacji

### c) Używanie formatu JSON

- Przy serializacji może dojść do błędu ze względu na niekompletne dane
- Przy deserializacji może dojść do dostania błędnych danych
- Potrzebna jest kontrola przy serializacji i deserializacji