Jak powinien wyglądać proces pozyskiwania danych zewnętrznych, w jakiej formie? Jakie dane można pobierać?

Warto w pierwszej kolejności zajrzeć do dokumentacji API.

Należy się zapoznać się ze strukturami danych, które chcemy pozyskać. Można to zrobić, np. poprzez sprawdzenie, jakie informacje (w tym format danych) są dostępne w odpowiedzi na zapytania. W praktyce można użyć Postmana (służy do testowania API) i zobaczyć, jakie dane otrzymuje się w odpowiedzi.

Trzeba się również zapoznać z warunkami korzystania z zewnętrznych usług API, aby się upewnić, czy nie są naliczane opłaty za korzystanie z ich usług lub czy usługa nie nakłada ograniczeń np. w postaci ilości zapytań na jednostkę czasu.

W przypadku iTunes Search API zwracałam uwagę na informacje o autorze, tytule, dacie wydania i cenie ebooka. W przypadku NBP API kluczowe dla mnie były kursy walut, numery tabeli

Jak wczytać do procesu dane wejściowe?

Dane wejściowe można wczytać do procesu poprzez formularz lub interfejs. W tym zadaniu stworzyłam interaktywny formularz, który został zaimplementowany w kodzie HTML przy użyciu Flask. Formularz ten jest częścią strony internetowej i pozwala użytkownikowi wprowadzać dane: autor i tytuł e-booka. Kod obejmuje również walidację danych- czy obie rubryki zostały wypełnione (jeżeli nie, to na ekranie użytkownika pojawia się informacja, że wprowadzone dane są niekompletne).

Jakie rozwiązania i struktury danych wykorzystuje API apple.com i NBP, jak ich użyć do celu w zadaniu?

<u>iTunes Search API:</u>

Endpoint:
URL: https://itunes.apple.com/search
Metoda: GET

Parametry Zapytania:
term: Fraza wyszukiwania, np. autor i tytuł ebooka.
entity: Typ wyszukiwania, w tym przypadku "ebook".
limit: Ograniczenie liczby wyników.

Struktura Odpowiedzi (JSON):
resultCount: Liczba wyników.
results: Lista wyników, gdzie każdy wynik zawiera informacje o ebooku.
artistName: Autor ebooka.
trackName: Tytuł ebooka.
releaseDate: Data wydania.
collectionPrice: Cena ebooka.

<u>NBP API:</u>

Endpoint:
URL: https://api.nbp.pl/api/exchangerates/rates/A/{currency_code}/?format=json
Metoda: GET

Parametry Zapytania:
currency_code: Kod waluty, np. "USD".

Struktura Odpowiedzi (JSON):

rates: Lista kursów walut.
mid: Średni kurs waluty.

Użycie w kontekście zadania:

Komunikacja z iTunes Search API:
Użytkownik wprowadza dane do formularza (autor, tytuł). Skrypt Pythona wysyła zapytanie do iTunes Search API używając wprowadzonych danych. Następnie otrzymuje odpowiedź API zawiera informacje, a następnie parsuje i wyodrębnia potrzebne informacje (autor, tytuł, data wydania, cena e-booka).

Komunikacja z NBP API:
Skrypt Pythona wysyła zapytanie do NBP API, otrzymuje kurs waluty i wyodrębnia wartość kursu (mid). W dalszej części skrypt przelicza cenę e-booka na polskie złote.

Czy zadanie pozyskiwania danych jest jednoznacznie sformułowane?

Tak, zadanie jest jasne w zakresie pobierania danych z iTunes Search API i NBP API dla określonych autorów/tytułów e-booków i kursu walut.

Jak wpłynąłby na rozwiązanie dodatkowy warunek: API NBP jest płatne za każdy wysłany request?

Jeśli API NBP jest płatne za każde zapytanie, należy uwzględnić to w kosztach projektu. Wpływa to na optymalizację liczby zapytań do API, a także na planowanie budżetu projektu. Można to zrobić w postaci: ustalenia limitów zapytań, wykorzystania mechanizmów cache'owania (aby ograniczyć liczbę zapytań do API), buforowania wyników lub korzystania z lokalnej pamięci podręcznej. Warto także rozważyć dostępność alternatywnych źródeł kursów walut.

Jak wykonać przeliczenia w sposób optymalny?

Należy kontrolować dokładność przeliczeń, unikając nadmiernego zaokrąglania, jednocześnie stosując się do zasad finansowych. Warto wprowadzić mechanizm buforowania wyników przeliczeń (jeśli użytkownik ponownie zapyta o tę samą pozycję, sprawdźmy, czy wynik nie jest już dostępny w buforze). Jeśli to możliwe, można zminimalizować ilość zapytań poprzez pobieranie danych zbiorczo zamiast pojedynczo dla każdego e-booka. Jeżeli dane o kursach walut zmieniają się rzadko, można rozważyć lokalne przechowywanie wcześniej pobranych kursów.

Struktura danych

Bazę danych można zaimplementowałam w PostgreSQL. Baza danych obejmuje tabelę z danymi wynikowymi, takimi jak autor, tytuł, oryginalna cena, data wydania, kurs waluty, przeliczona cena na PLN i numer tabeli.