ChatGPT

Rozbudowa modułu kolekcji kart z użyciem API TCGGO

Planując moduł „Moja kolekcja”, warto wydzielić następujące elementy: **rejestrację kont użytkowników**, mechanizm tworzenia i zarządzania kolekcją (grupą posiadanych kart), **wyszukiwarkę kart** oraz funkcję wyliczania wartości kolekcji na podstawie danych cenowych z API TCGGO. Konta mogą być realizowane np. poprzez prostą rejestrację (login/hasło lub logowanie przez Google/Facebook) – bez zbierania dodatkowych danych osobowych. Dla każdego użytkownika przechowujemy w bazie tylko minimalne info (np. unikalne ID i hasło), zaś powiązane z kontem kolekcje (nazwa kolekcji, lista kart, ilość sztuk) przechowujemy w osobnych tabelach.

- **Zakładanie kont:** prosta rejestracja, np. tylko nazwa użytkownika i hasło. Ewentualnie logowanie OAuth (Google/Facebook), które eliminuje konieczność ręcznej rejestracji.
- **Tworzenie kolekcji:** użytkownik może mieć jedną lub kilka kolekcji kart. Struktura bazy: tabela `Collections` (id, user_id, name), tabela `CollectionItems` (collection_id, card_id, quantity).
- **Przegląd kolekcji:** lista dodanych kart z możliwością usuwania/edycji.
- **Wyszukiwarka kart:** interfejs do wpisania nazwy karty lub przeglądania według zestawów.
- **Wycena kolekcji:** sumowanie wartości cenowych kart (np. *lowest near-mint* z Cardmarket lub inna cena) pomnożonych przez ilość sztuk.

Integracja z API TCGGO

Wszystkie dane o kartach będą pochodzić z **Pokemon TCG API (TCGGO)** dostarczanego przez RapidAPI. Bazowy URL to `https://pokemon-tcg-api.p.rapidapi.com`. Zgodnie z dokumentacją dostępne są m.in. następujące operacje REST:

- **GET /cards?search=...** – wyszukiwanie kart po nazwie (np. `?search=Pikachu`) [1].
- **GET /cards** – ogólna lista kart (bez parametru `search` zwraca wiele kart, można też użyć paginacji).
- **GET /cards/{cardId}** – pobiera szczegóły pojedynczej karty według `cardId`.
- **GET /episodes** – pobranie listy wszystkich zestawów (expansions) [2]. W kontekście Pokemon TCG „episodes” to zestawy wydawnicze (np. *Crown Zenith*).
- **GET /episodes/search?search=...** – wyszukiwanie zestawów po nazwie [3].
- **GET /episodes/{id}/cards** – pobranie wszystkich kart z konkretnego zestawu (według jego ID) [4].
- **GET /products**, **GET /products/search**, **GET /episodes/{id}/products** – analogiczne operacje dla produktów (booster packs, sety), jeśli potrzebne.

Dokumentacja TCGGO wymienia m.in. te ścieżki (lista operacji GET jest widoczna w doceliste API) [5] [6]. Każde żądanie musi zawierać nagłówek lub parametr `rapidapi-key` z kluczem API (dostęp zdobywa się przez rejestrację na RapidAPI i wykupienie planu – również dostępny jest plan darmowy) [1]. Przykładowe żądanie do wyszukania karty to np.:

```
GET https://pokemon-tcg-api.p.rapidapi.com/cards?search=Charizard&rapidapi-key=TWÓJ_KLUCZ
```

Dane zwracane przez API

Odpowiedź JSON z API zawiera szczegółowe informacje o kartach. Każdy obiekt karty ma m.in. pola takie jak `id`, `name`, `slug`, `rarity` oraz obiekt `prices`, który agreguje ceny z różnych źródeł [7] [8]. Przykładowo:

- `prices.cardmarket`: zawiera informacje o cenach z serwisu Cardmarket (EUR) – są tam m.in. `lowest_near_mint` (najniższa cena w stanie Near Mint), `30d_average` (średnia cena z 30 dni) oraz ceny dla ocenionych egzemplarzy (sekcja `graded` z PSA/CPS).
- `prices.tcg_player`: zawiera ceny z rynku TCGPlayer (USD) – `market_price` i `mid_price`.

Na przykład odpowiedź dla karty może zawierać:

```
{
  "id": 3852,
  "name": "Giratina VSTAR",
  "rarity": "Rare Secret",
  "prices": {
    "cardmarket": {
      "currency": "EUR",
      "lowest_near_mint": 157.21,
      "30d_average": 192.79,
      "graded": {
        "psa": {"psa10": 279, "psa9": 184},
        "cgc": {"cgc10": 344}
      }
    },
    "tcg_player": {
      "currency": "USD",
      "market_price": 146.69,
      "mid_price": 163.71
    }
  },
  "episode": {"name": "Crown Zenith", "series": {"name": "Sword & Shield"} },
  "artist": {"name": "AKIRA EGAWA"},
  "image": "...",
  "tcggo_url": "https://www.tcggo.com/pokemon/..."
}
```

(Zacytowany fragment pokazuje strukturę pola `prices` [7] [8]). W odpowiedziach dostępne są też inne pola: `episode` (nazwy zestawu, logotyp), `artist` (autor ilustracji), `image` (URL obrazka karty) czy `tcggo_url` (link do strony TCGGO).

Przykładowy przebieg działania

1. Pobranie listy zestawów: Po wejściu do modułu kolekcji można wyświetlić listę zestawów Pokemon, np. łącząc się z `GET /episodes` [2]. Na podstawie zwróconych danych (np. nazwa, rok wydania, logo) można zbudować menu lub katalog, podobnie jak na stronie TCGGO. Opcjonalnie można udostępnić wyszukiwarkę zestawów używając `GET /episodes/search?search=nazwa` [3].

2. Przegląd kart z zestawu: Po wybraniu konkretnego zestawu wykonujemy `GET /episodes/{id}/cards`, co zwróci wszystkie karty ze wskazanego expansion. Następnie wyświetlamy listę tych kart (np. miniaturki, nazwy, rarity, aktualna cena) – dane te możemy wziąć z odpowiedzi API (pole `prices`) [7] [8]. Na interfejsie „Przeglądaj karty" można także udostępnić sortowanie według ceny lub trendów, podobnie jak robi to TCGGO.

3. Wyszukiwanie kart: Pozwól użytkownikowi wpisywać fragment nazwy karty i wywołaj `GET /cards?search=fraza` [1]. API zwróci karty pasujące do nazwy. Wyświetlamy wyniki z ich nazwami, obrazkami i cenami. Można też posłużyć się `GET /cards/{cardId}` dla pełnych detali pojedynczej karty (np. po kliknięciu).

4. Dodawanie do kolekcji: Gdy użytkownik znajdzie kartę, może dodać ją do swojej kolekcji. Wówczas zapisujemy w bazie: `card_id` (dokładnie wartość `id` z API), opcjonalnie `name`, `rarity` (dla łatwiejszego wyszukiwania), i liczbę sztuk. Warto przechować także sumę kosztu przy dodaniu (faktycznie użyte, albo bieżącą cenę).

5. Wycena kolekcji: Aby obliczyć wartość kolekcji, dla każdej karty pobieramy jej aktualną cenę (np. `prices.cardmarket.lowest_near_mint` lub średnią 30-dniową) z API i mnożymy przez ilość sztuk. Sumę prezentujemy użytkownikowi. Przykład: jeśli kolekcja zawiera 3x „Giratina VSTAR" po cenie 157,21 EUR każda, to wartość to ok. 471,63 EUR (lub w innej walucie po przeliczeniu). Pole `prices` w odpowiedzi API zawiera dokładnie te dane [7]. Dodatkowo można pobrać wartości zweryfikowanych egzemplarzy (PSA/CPS), jeśli chcesz wspierać kolekcje kart kolekcjonerskich.

Podsumowanie

Integracja opiera się na pobieraniu danych z **Pokemon TCG API (TCGGO)** przez REST. Należy najpierw uzyskać klucz z RapidAPI i w każdym żądaniu dodać go jako `rapidapi-key`. Następnie moduł udostępnia interfejs podobny do strony TCGGO – przegląd zestawów (expansions), listę kart, szczegóły karty i przycisk „Dodaj do kolekcji". Dane do wyceny kart (w tym aktualne ceny z rynków EU/US) pochodzą bezpośrednio z API (np. pola `prices.cardmarket.lowest_near_mint`, `prices.tcg_player.market_price` itd. w odpowiedzi JSON) [7] [8]. Dzięki temu można dynamicznie aktualizować wartość zbioru kart. Często cytowana dokumentacja TCGGO potwierdza dostępność potrzebnych endpointów i struktury danych [5] [1], co pozwala na płynną realizację opisanych funkcji.

Źródła: Dokumentacja API TCGGO (RapidAPI) pokazuje listę endpointów i przykłady użycia [5] [1]. Strona informacyjna PokemonTCG API (TCGGO) zawiera przykładową odpowiedź JSON z polami cenowymi (Cardmarket, TCGPlayer) [7] [8]. Te źródła stanowią bazę implementacyjną dla opisanego modułu.

---

1 2 3 4 7 8 PokemonTCG API - The Ultimate TCG Pricing API
https://pokemon-api.com/

5 6 TCGGO API Documentation
https://tcgapi.apidocumentation.com/