CoinGecko API

Pobiera, udostępnia na zewnątrz poprzez gRPC, cachuje oraz zarządza danymi z CoinGecko.

1. Uruchamianie

Aby uruchomić gRPC API należy do w katalogu głównym umieścić plik ".github_token" w którym znajduje się token autoryzujący z serwisu GitHub.

W celu uruchomienia należy posiadać zainstalowanego dockera oraz wykonać następujące polecenia w podanej kolejności:

```
docker compose build
docker compose up -d
```

Domyślnie gRPC API jest uruchamiane na porcie 50051, zmiana portu odbywa się w pliku compose.yaml, znajdują się tam również inne zmienne.

W celu zatrzymania aplikacji należy użyć polecenia:

```
docker compose down
```

2. Opis funkcji

2.1 GetHistoricalData():

Parametry wejściowe:

`coin_id` - id crypto coina z MongoDB

`start_date` - data startowa jako timestamp

`end_date` - data końcowa jako timestamp

`fiat_coin` - id fiat coina z MongoDB

Paramtery wyjściowe:

`status` - status operacji, success lub error

`error_message` - jeśli status=error to opis błędu w przeciwnym wypadku puste

`data` - jeśli status=success to słownik, w przeciwnym wypadku puste

Słownik data z przykładowego response:

Pierwszy rekord z tabeli timestamp odpowiada pierwszemu rekordowi z tabeli price.

```
{
    timestamp: [
        1732230268459,
        1732234490890,
        1732238016359
    ],
    price: [
        98441.26161000447,
        98529.80211326091,
        98187.91803696626
    ]
}
```

2.2 GetCoinData

Parametry wejściowe:

`coin_id` - id crypto coina z MongoDB

`fiat_coin` - id fiat coina z MongoDB

Paramtery wyjściowe:

`status` - status operacji, success lub error

`error_message` - jeśli status=error to opis błędu w przeciwnym wypadku puste

`data` - jeśli status=success to słownik, w przeciwnym wypadku puste

Słownik data z przykładowego response:

```
{
   "id":"bitcoin",
   "symbol":"btc",
   "name":"Bitcoin",
   "market_data": {
      "current_price": 351080,
      "market_cap":6948517954631,
      "total_volume":175359671780,
      "high_24h":366881,
      "low_24h":349429,
      "price_change_24h_in_currency":-12089.517051851959,
      "price_change_percentage_24h_in_currency":-3.32889
      }
   }
}
```

2.3 GetAllCoinsPrices

Parametry wejściowe:

Brak

Paramtery wyjściowe:

`status` - status operacji, success lub error

`error_message` - jeśli status=error to opis błędu w przeciwnym wypadku puste

`data` - jeśli status=success to słownik, w przeciwnym wypadku puste

Struktura słownika jest następująca:

W słowniku są pola o nazwach takich jak id crypto z MongoDB, dla każdego pola w środku jest słownik z polami o nazwach odpowiadających id fiat z MongoDB, w każdym z tych pól jest cena danego crypto coina w danym fiat coinie, na przykład:

```
{
    "bitcoin": {
        "usd": 94000,
        "pln": 320000
    },
    "ethereum": {
        "usd": 3200,
        "pln": 12000
    },
}
```

2.4 GetListDataForAllCoins

Parametry wejściowe:

`fiat_coin` - id fiat coina z MongoDB

Paramtery wyjściowe:

`status` - status operacji, success lub error

`error_message` - jeśli status=error to opis błędu w przeciwnym wypadku puste

`data` - jeśli status=success to słownik słowników, w przeciwnym wypadku puste

Słownik data z przykładowego response:

```
{
    "bitcoin": {
        "id":"bitcoin",
```

```
        "symbol":"btc",
        "name":"Bitcoin",
        "market_data":{
            "current_price": 351080,
            "market_cap":6948517954631,
            "total_volume":175359671780,
            "high_24h":366881,
            "low_24h":349429,
            "price_change_24h_in_currency":-12089.517051851959,
            "price_change_percentage_24h_in_currency":-3.32889
            }
        }
    }
}
```


2.5 GetHistoricalCandleData():

Parametry wejściowe:

`coin_id` - id crypto coina z MongoDB

`start_date` - data startowa jako timestamp

`end_date` - data końcowa jako timestamp

`fiat_coin` - id fiat coina z MongoDB

Paramtery wyjściowe:

`status` - status operacji, success lub error

`error_message` - jeśli status=error to opis błędu w przeciwnym wypadku puste

`data` - jeśli status=success to słownik, w przeciwnym wypadku puste

Słownik data z przykładowego response:

Pierwszy rekord z tabeli timestamp odpowiada pierwszemu rekordowi z tabel low, open, high oraz close.


```
{
    timestamp: [
        1732230268459,
        1732234490890,
        1732238016359
    ],
    low: [
        98440,
        98520,
        98180
```

```
    ],
    open: [
        98443,
        98523,
        98183
    ],
    high: [
        98449,
        98559,
        98189
    ],
    close: [
        98448,
        98528,
        98188
    ]
}
```