Mini Projekt - Bazy Danych II

Aplikacja Aukcyjna - Backend i Frontend

Autorzy:

- **Bartosz Ludwin**
- **Filip Malejki**
- **Mateusz Pawliczek**

Link do repozytorium GitHub: https://github.com/FalcO0O/Bazy-danych-II

Wprowadzenie

Projekt stanowi implementację **pełnej aplikacji aukcyjnej** składającej się z backendu oraz frontendu. Umożliwia on tworzenie aukcji oraz licytowanie ich przez użytkowników. Informacje są przechowywane w systemie bazodanowym, w którym znajdują się dane o użytkownikach, aukcjach oraz logi transakcji.

Technologie

Projekt został zrealizowany przy użyciu następujących technologii:

Backend

- **Python + FastAPI**
- **Baza danych**: MongoDB
- **Komunikacja z bazą**: Motor (asynchroniczny klient MongoDB) + PyMongo

Frontend

- **Interfejs użytkownika** dostępny pod adresem **localhost:3000**

Środowisko uruchomieniowe

- **Docker** z Docker Compose

Dodatkowo wykorzystano biblioteki do:

- obsługi zapytań HTTP,
- szyfrowania haseł,
- tworzenia i walidacji tokenów JWT.

Uruchomienie aplikacji

Aplikacja składa się z backendu i frontendu, które są uruchamiane jednocześnie za pomocą Docker Compose:

```
docker compose up
```

Po uruchomieniu:

- **Frontend** będzie dostępny pod adresem: **http://localhost:3000**
- **Backend API** będzie dostępny zgodnie z konfiguracją Docker Compose

Funkcjonalności

API aplikacji aukcyjnej umożliwia:

- Rejestrację użytkownika i logowanie do systemu.
- Obsługę tokenów JWT i mechanizmu odświeżania tokenów (refresh token).
- Tworzenie nowych aukcji lub ich zamknięcie przez zalogowanych użytkowników.
- Edycję i usuwanie aukcji przez administratora.
- Licytowanie aukcji przez użytkowników z weryfikacją poprawności oferty.
- Ręczne zakończenie aukcji przez administratora.
- Generowanie raportów zawierających:
  - zestawienie aktywnych i zakończonych aukcji,
  - statystyki użytkowników,
  - informacje o przepływach finansowych.

- Logowanie działań użytkowników (logowanie, rejestracja konta)
- Logowanie zmian aukcji (utworzenie, zamknięcie, licytowanie)

Backend zapewnia poprawne wykonanie powyższych operacji poprzez wykorzystanie **systemu transakcji**. Jest to szczególnie istotne w przypadku konkurencyjnych operacji, takich jak licytacje, gdzie wiele ofert może być składanych jednocześnie.

Frontend zapewnia intuicyjny interfejs użytkownika umożliwiający korzystanie ze wszystkich funkcjonalności aplikacji aukcyjnej bez konieczności bezpośredniego korzystania z API.

Możliwe udoskonalenia

Aktualnie aukcje są kończone ręcznie przez właściciela aukcji lub administratora. Możliwym usprawnieniem byłoby wprowadzenie mechanizmu automatycznego zamykania aukcji na podstawie ustalonej daty zakończenia.

Testowanie Race Condition

W celu weryfikacji działania systemu transakcji użytego w endpointach licytacji, przygotowano dedykowany program testujący symulujący warunki wyścigu (race condition).

Testy można uruchomić na osobnej, testowej bazie danych, co zapewnia bezpieczeństwo i izolację środowiska.

Wymagania

Do uruchomienia testu potrzebny jest interpreter **Python**. Można go pobrać ze strony: https://www.python.org/downloads/

Opis testu

Test obejmuje następujące kroki:

- Rejestrację lub logowanie trzech kont testowych.

- Utworzenie aukcji przez użytkownika A.
- Trzykrotne synchroniczne licytowanie tej aukcji przez użytkowników B i C (równocześnie).
- Zamknięcie aukcji przez użytkownika A.

Uruchomienie testu

Test wymaga aktywnej bazy danych. Włącz ją przed wykonaniem poniższych komend:

```
docker compose up
```

Aby uruchomić test:

1. Przejdź do folderu `testing` :

   ```
   cd testing
   ```

2. Pobierz wymagane dependencje jeśli jeszcze tego nie zrobiłeś:

   ```
   pip install httpx asyncio
   ```

3. Uruchom skrypt:

   ```
   py racetest.py
   ```

Test zostanie wykonany, a wyniki pojawią się w konsoli.

Przykładowe wyniki:

```
[!] test-user-xkawofngyh131 already exists, trying to login...
[!] test-user-jjkoelfmah125 already exists, trying to login...
[!] test-user-kdrihnekog232 already exists, trying to login...
✅ Auction created: 6845fca72bda3f4afc7ac07c
[BID] Status: 200, Amount: 150, Response: {"id":"6845fca72bda3f4afc7ac07e", ...
[BID] Status: 400, Amount: 150, Response: {"detail":"Kwota oferty musi być wyższa ...
[BID] Status: 200, Amount: 200, Response: {"id":"6845fca92bda3f4afc7ac080", ...
[BID] Status: 400, Amount: 200, Response: {"detail":"Kwota oferty musi być wyższa ...
[BID] Status: 200, Amount: 250, Response: {"id":"6845fca92bda3f4afc7ac082", ...
[BID] Status: 400, Amount: 250, Response: {"detail":"Kwota oferty musi być wyższa ...
✅ Auction 6845fca72bda3f4afc7ac07c closed.
```

Jak widać, system transakcji zapobiega konfliktom — tylko jedna oferta o danej kwocie jest zaakceptowana, a pozostałe są odrzucane.
W przypadku błędu transakcyjnego mogłoby dojść do zaakceptowania tej samej kwoty wielokrotnie.

Zapytania do Bazy - `Dokumentacja`

`ZAPYTANIA: AUTORYZACJA`

Rejestracja Użytkownika

`POST /register`

Opis:
Odpowiada za **rejestrację nowych użytkowników**. Waliduje unikalność e-maila, hashuje hasło i zapisuje dane użytkownika do bazy danych (bez hasła). Nowi użytkownicy otrzymują domyślną rolę "user".

Akceptowane dane (JSON):

```
{
    "username": "string",
    "email": "user@example.com",
    "password": "string"
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Pomyślna rejestracja** użytkownika.
- `400 Bad Request` : **Adres e-mail już zarejestrowany**.

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "username": "string",
    "email": "user@example.com",
    "role": "string"
}
```

Logowanie Użytkownika

`POST /login`

Opis:
Uwierzytelnia użytkownika na podstawie podanego e-maila (jako `username`) i hasła. Po pomyślnej weryfikacji generuje **`access_token`** (krótkożyjący) i **`refresh_token`** (dłużej żyjący, do odnawiania `access_token`). `refresh_token` jest zapisywany w bazie danych użytkownika.

Akceptowane dane (Form-data - `application/x-www-form-urlencoded`):

- `username` : Adres e-mail użytkownika.
- `password` : Hasło użytkownika.

Kody odpowiedzi HTTP:

- `200 OK` : **Pomyślne zalogowanie**, zwrócono tokeny.
- `400 Bad Request` : **Nieprawidłowy e-mail lub hasło**.

Format danych zwracanych (JSON):

```
{
    "access_token": "string",
    "refresh_token": "string",
    "token_type": "bearer"
}
```

Odświeżanie Tokena Dostępu

`POST /token/refresh`

Opis:
Pozwala **odnowić wygasły** `access_token` za pomocą `refresh_token`. Weryfikuje `refresh_token` i jego obecność w bazie. Jeśli jest ważny, generuje **nowy** `access_token` **i nowy** `refresh_token`, usuwając stary i dodając nowy do bazy.

Akceptowane dane (JSON):

```
{
    "refresh_token": "string"
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Nowe tokeny wygenerowane pomyślnie**.
- `401 Unauthorized` : **Nieprawidłowy token odświeżania** (np. wygasły, zmodyfikowany, nieznany).

Format danych zwracanych (JSON):

```
{
    "access_token": "string",
    "refresh_token": "string",
    "token_type": "bearer"
}
```

Wylogowanie z Sesji

`POST /logout`

Opis:
Bezpiecznie **kończy konkretną sesję użytkownika** przez usunięcie podanego `refresh_token` z bazy danych. Wymaga **uwierzytelnienia żądania** za pomocą ważnego `access_token` w nagłówku autoryzacji.

Akceptowane dane (JSON):

```
{
    "refresh_token": "string"
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Pomyślnie wylogowano** z bieżącego urządzenia.
- `400 Bad Request` : **Token odświeżania jest nieprawidłowy** lub już unieważniony.
- `401 Unauthorized` : **Brak lub nieprawidłowy** `access_token` w nagłówku autoryzacji.

Format danych zwracanych (JSON):

```
{
    "message": "string"
}
```

`ZAPYTANIA: UŻYTKOWNICY`

Lista Użytkowników

`GET /users/`

Opis:
Ten endpoint umożliwia **pobranie listy wszystkich zarejestrowanych użytkowników** w systemie.

Dostęp do tej funkcji jest **ściśle ograniczony** wyłącznie do użytkowników posiadających rolę **administratora**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę wszystkich użytkowników**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "username": "string",
        "email": "user@example.com",
        "role": "string"
    }
]
```

Pobieranie Profilu Użytkownika

`GET /users/{user_id}`

Opis:
Endpoint służy do **pobierania szczegółowych danych profilu konkretnego użytkownika**. Użytkownicy z rolą **administratora** mogą pobrać profil **dowolnego** użytkownika. Zwykli użytkownicy mogą pobrać **wyłącznie swój własny** profil.

Parametry ścieżki:

- `user_id` : Unikalny identyfikator użytkownika (string).

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **profil użytkownika**.
- `400 Bad Request` : Podano **nieprawidłowy format identyfikatora użytkownika**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Użytkownik **nie posiada uprawnień** do pobrania wskazanego profilu (np. zwykły użytkownik próbuje pobrać profil innej osoby).
- `404 Not Found` : Użytkownik o podanym `user_id` **nie został znaleziony**.

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "username": "string",
    "email": "user@example.com",
    "role": "string"
}
```

Historia Licytacji Użytkownika

`GET /users/{user_id}/bids`

Opis:
Ten endpoint umożliwia **pobranie listy wszystkich licytacji (ofert)** złożonych przez wskazanego użytkownika. Dostęp do tej historii jest możliwy **wyłącznie dla właściciela konta** lub dla użytkownika posiadającego rolę **administratora**.

Parametry ścieżki:

- `user_id` : Unikalny identyfikator użytkownika (string), którego historię licytacji chcemy pobrać.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę licytacji** danego użytkownika.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Użytkownik **nie posiada uprawnień** do wglądu w historię licytacji (np. zwykły użytkownik próbuje zobaczyć licytacje innej osoby).

- `404 Not Found` : Użytkownik o podanym `user_id` **nie został znaleziony** (choć endpoint nie sprawdza bezpośrednio istnienia użytkownika, brak ofert dla ID nie zwróci błędu 404 dla użytkownika).

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "auction_id": "string",
        "user_id": "string",
        "amount": 0.0,
        "timestamp": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
    }
]
```

Zapytania: Aukcje

Tworzenie Aukcji

POST /auctions

Opis:
Ten endpoint umożliwia **tworzenie nowych aukcji**. Dostęp jest ograniczony do **zalogowanych użytkowników**. System automatycznie przypisuje `owner_id` (ID twórcy aukcji) oraz ustawia `current_price` na wartość `starting_price` podaną w żądaniu. Rekord `created_at` jest również ustawiany automatycznie na aktualny czas.

Akceptowane dane (JSON):

```
{
    "title": "string",
    "description": "string (optional)",
    "starting_price": 0.0
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Aukcja została pomyślnie utworzona**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "title": "string",
    "description": "string",
    "owner_id": "string",
    "current_price": 0.0,
    "created_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
}
```

Lista Aktywnych Aukcji

`GET /auctions`

Opis:
Endpoint zwraca **listę wszystkich aktualnie aktywnych aukcji** dostępnych w systemie.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę aktywnych aukcji**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "current_price": 0.0,
        "created_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
    }
]
```


Pobieranie Danych Aukcji

`GET /auctions/{auction_id}`

Opis:
Ten endpoint służy do **pobierania szczegółowych informacji o konkretnej aukcji** na podstawie jej unikalnego identyfikatora.

Parametry ścieżki:

- `auction_id` : Unikalny identyfikator aukcji (string).

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **szczegóły aukcji**.
- `400 Bad Request` : Podano **nieprawidłowy format identyfikatora aukcji**.
- `404 Not Found` : Aukcja o podanym `auction_id` **nie została znaleziona**.

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "title": "string",
    "description": "string",
    "owner_id": "string",
    "current_price": 0.0,
    "created_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
}
```

Składanie Oferty (Licytacji)

`POST /auctions/{auction_id}/bid`

Opis:
Endpoint umożliwia **złożenie nowej oferty (licytacji)** na wskazaną aukcję. Kwota oferty musi być **wyższa niż aktualna cena** aukcji. Operacja jest wykonywana w transakcji, z mechanizmem ponawiania w przypadku błędów przejściowych. Dostęp jest ograniczony do **zalogowanych użytkowników**.

Parametry ścieżki:

- `auction_id` : Unikalny identyfikator aukcji (string), na którą składana jest oferta.

Akceptowane dane (JSON):

```
{
    "amount": 0.0
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Oferta została pomyślnie złożona**.
- `400 Bad Request` : **Nieprawidłowy identyfikator aukcji** lub **kwota oferty jest niższa lub równa bieżącej cenie**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `404 Not Found` : Aukcja o podanym `auction_id` **nie została znaleziona**.
- `500 Internal Server Error` : Wystąpił błąd serwera (np. przekroczono liczbę prób transakcji).

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "auction_id": "string",
    "user_id": "string",
    "amount": 0.0,
    "timestamp": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
}
```

Zamykanie Aukcji

`POST /auctions/{auction_id}/close`

Opis:
Endpoint ten umożliwia **zamknięcie aktywnej aukcji**. Po zamknięciu, aukcja jest przenoszona do kolekcji `history`, a wszystkie powiązane oferty są usuwane. Zwycięzca aukcji jest wyłaniany na podstawie najwyższej oferty (jeśli istnieją oferty). Dostęp do tej funkcji jest ograniczony do **właściciela aukcji** (`owner_id`) lub **administratora**. Cała operacja wykonywana jest w ramach transakcji.

Parametry ścieżki:

- `auction_id` : Unikalny identyfikator aukcji (string) do zamknięcia.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : **Aukcja została pomyślnie zakończona**.
- `400 Bad Request` : Podano **nieprawidłowy format identyfikatora aukcji**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Użytkownik **nie posiada uprawnień** do zakończenia tej aukcji.
- `404 Not Found` : Aukcja o podanym `auction_id` **nie została znaleziona**.

Format danych zwracanych (JSON):

```
{
    "message": "string",
    "winner_id": "string (lub null, jeśli brak ofert)",
    "final_price": 0.0
}
```

Edycja Danych Aukcji przez Administratora

`PATCH /auctions/{auction_id}`

Opis:
Endpoint umożliwia **modyfikację danych aktywnej aukcji** wyłącznie przez **administratora**. Dozwolone pola do edycji to `title`, `description` i `starting_price`. **Zmiana `starting_price` jest niemożliwa, jeśli na aukcję zostały już złożone jakiekolwiek oferty**.

Parametry ścieżki:

- `auction_id` : Unikalny identyfikator aukcji (string) do edycji.

Akceptowane dane (JSON):
Przyjmuje obiekt JSON zawierający pola do zaktualizowania.

```
{
    "title": "string (optional)",
    "description": "string (optional)",
    "starting_price": 0.0 (optional)
}
```

Kody odpowiedzi HTTP:

- `200 OK` : **Aukcja została pomyślnie zaktualizowana**.
- `400 Bad Request` : Brak danych do aktualizacji, podano **nieprawidłowe pole** do edycji, **nieprawidłowy identyfikator aukcji**, lub próbowano **zmienić cenę startową przy istniejących ofertach**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.
- `404 Not Found` : Aukcja o podanym `auction_id` **nie została znaleziona**.

Format danych zwracanych (JSON):

```
{
    "id": "string",
    "title": "string",
    "description": "string",
    "owner_id": "string",
    "current_price": 0.0,
    "created_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm"
}
```

Zapytania: Raporty

Historia Wszystkich Aukcji

`GET /reports/history`

Opis:
Endpoint umożliwia **pobranie pełnej historii wszystkich aukcji**, które zostały zakończone. Dostęp do tej funkcji jest **wyłącznie dla użytkowników z rolą administratora**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę zakończonych aukcji** wraz ze szczegółami.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "created_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm",
        "closed_at": "YYYY-MM-DDTHH:MM:SS.mmmmmm",
        "winner_id": "string (lub null)",
        "final_price": 0.0 (lub null)
    }
]
```

Wydatki Użytkowników

`GET /reports/user-spending`

Opis:
Endpoint generuje raport **sumarycznych wydatków każdego użytkownika**, który wygrał przynajmniej jedną aukcję. Wyniki są sortowane malejąco według sumy wydatków. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę użytkowników z ich całkowitymi wydatkami** i liczbą wygranych aukcji.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "user_id": "string",
        "username": "string",
        "email": "user@example.com",
        "total_spent": 0.0,
        "won_count": 0
    }
]
```


Top Zwycięzcy Aukcji

`GET /reports/top-winners`

Opis:
Endpoint zwraca listę użytkowników z **największą liczbą wygranych aukcji**. Domyślnie lista obejmuje top 10 zwycięzców. Możliwe jest określenie limitu wyników. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Parametry zapytania (query parameters):

- `limit` : Liczba wyników do zwrócenia (domyślnie `10`).

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę użytkowników z największą liczbą wygranych aukcji**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "user_id": "string",
        "username": "string",
        "won_count": 0,
        "total_spent": 0.0
    }
]
```


Całkowity Przepływ Gotówki

`GET /reports/total-cashflow`

Opis:
Endpoint oblicza i zwraca **całkowitą wartość pieniężną wygenerowaną przez wszystkie zakończone aukcje**, które miały zwycięzcę. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **sumę całkowitego przepływu gotówki**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON):


```
{
    "total_cashflow": 0.0
}
```

Aukcje Wysokiej Wartości

`GET /reports/high-value-auctions`

Opis:
Endpoint zwraca listę zakończonych aukcji, których **cena końcowa przekroczyła określoną minimalną wartość**. Domyślna wartość minimalna to `1000.0`. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Parametry zapytania (query parameters):

- `min_price` : Minimalna cena końcowa aukcji do uwzględnienia (domyślnie `1000.0`).

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę aukcji o wysokiej wartości końcowej**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "created_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "closed_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "winner_id": "string",
        "final_price": 0.0
    }
]
```

Aukcje z Ostatniego Tygodnia

`GET /reports/last-week-auctions`

Opis:
Endpoint zwraca listę aukcji, które **zostały utworzone w ciągu ostatnich 7 dni**. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę aukcji utworzonych w ciągu ostatniego tygodnia**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "created_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "closed_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "winner_id": "string",
        "final_price": 0.0
    }
]
```

Aukcje z Ostatniego Miesiąca

`GET /reports/last-month-auctions`

Opis:
Endpoint zwraca listę aukcji, które **zostały utworzone w ciągu ostatnich 30 dni**. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę aukcji utworzonych w ciągu ostatniego miesiąca**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):

```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "created_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "closed_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "winner_id": "string",
        "final_price": 0.0
    }
]
```

Aukcje z Ostatnich 6 Godzin

`GET /reports/last-6h-auctions`

Opis:
Endpoint zwraca listę aukcji, które **zostały utworzone w ciągu ostatnich 6 godzin**. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **listę aukcji utworzonych w ciągu ostatnich 6 godzin**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.

- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON Array):


```
[
    {
        "id": "string",
        "title": "string",
        "description": "string",
        "owner_id": "string",
        "created_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "closed_at": "YYYY-MM-DDTHH:MM:SS+0000",
        "winner_id": "string",
        "final_price": 0.0
    }
]
```


Statystyki Aukcji

`GET /reports/auctions-stats`

Opis:
Endpoint dostarcza **zagregowane statystyki dotyczące aukcji**, w tym liczbę aktualnie aktywnych aukcji oraz liczbę aukcji, które zostały już zakończone. Dostęp do tego raportu jest **wyłącznie dla administratorów**.

Akceptowane dane:
Brak.

Kody odpowiedzi HTTP:

- `200 OK` : Zwraca **statystyki dotyczące aktywnych i zakończonych aukcji**.
- `401 Unauthorized` : Użytkownik nie jest uwierzytelniony.
- `403 Forbidden` : Uwierzytelniony użytkownik **nie posiada uprawnień administratora**.

Format danych zwracanych (JSON):

```
{
    "auctions_active": 0,
    "auctions_closed": 0
}
```