**Sprawozdanie z projektu REST API**

**1. Wprowadzenie**

Celem projektu było stworzenie aplikacji internetowej z wykorzystaniem architektury REST. Tematem mojej aplikacji jest baza piłkarzy i drużyn piłkarskich, z możliwością zarządzania danymi oraz przeliczania wartości rynkowych na różne waluty.

**2. Konsumowane API**

- API Transfermarkt: prywatny projekt pobrany z GitHub-a. Jest on uruchamiany lokalnie pod adresem: <http://localhost:8000>  
Szczegóły są dostępne pod adresem: <https://github.com/felipeall/transfermarkt-api>

- NBP Web API: dostępne pod adresem: <http://api.nbp.pl>

**3. Opis aplikacji**

**3.1 Funkcjonalności**

Piłkarze:

- Dodawanie, usuwanie oraz przeglądanie graczy.

- Przeliczanie wartości gracza na inne waluty.

- Konsumpcja API Transfermarkt do pobierania danych o piłkarzach.

Drużyny:

- Dodawanie (na podstawie dodanych piłkarzy), usuwanie oraz przeglądanie drużyn.

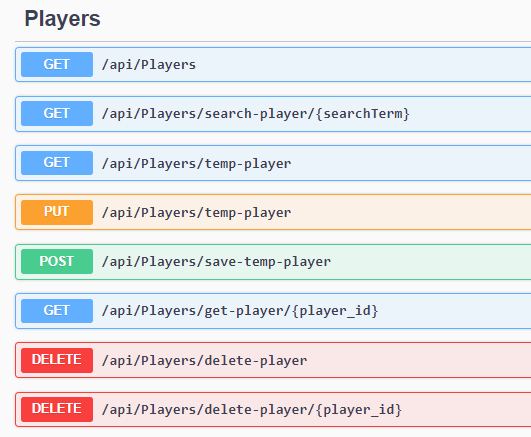
- Przeliczanie wartości drużyny na inne waluty.

Autentykacja:

- Logowanie użytkowników przy użyciu JWT.

**3.2 Endpointy API**

**Piłkarze**Intuicja: Możemy wyszukiwać w API Transfermarkt piłkarza. Następnie zapisywać go w jako gracza tymczasowego. Tylko zapisanego piłkarza tymczasowego można umieścić na liście wszystkich graczy. Ponadto możemy: wyświetlić wszystkich piłkarzy oraz piłkarza zapisanego tymczasowo; usunąć piłkarza; uzyskać dane o piłkarzu na podstawie ID.

****

GET /api/players: Zwraca listę wszystkich dodanych piłkarzy.

GET /api/players/search-player/{searchTerm}: Zwraca dane pierwszego znalezionego w API Transfermarkt piłkarza.

GET /api/players/temp-player: Zwraca dodanego do pliku tymczasowego piłkarza.

PUT /api/players/temp-player: Zapisuje piłkarza do pliku tymczasowego.

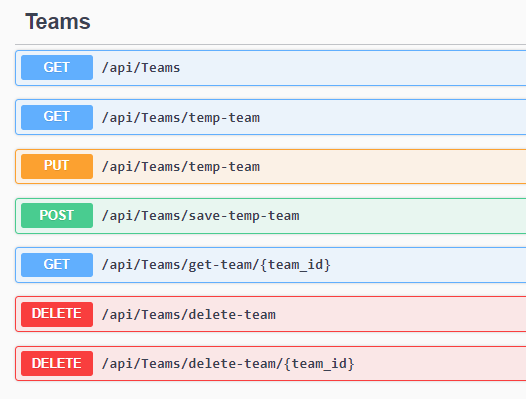
POST /api/players: Dodaje nowego piłkarza (tymczasowego) do pliku z wszystkimi piłkarzami.

GET /api/players/{player\_id}: Zwraca dane konkretnego piłkarza z Trasfermarkt.

DELETE /api/players/delete-player: Usuwa danego piłkarza z pliku.

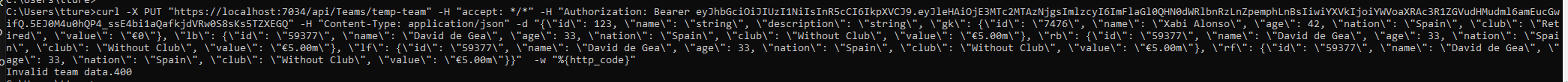
DELETE /api/players/delete-player/{player\_id}: Usuwa piłkarza o danym ID.

**Drużyny – wymagają autoryzacji**Intuicja: Podobnie jak z piłkarzami mamy drużynę tymczasową. Możemy wstawić do niej dane (tylko piłkarzy z listy dodanych piłkarzy), a następnie umieścić je w pliku metodą POST. Ponadto możemy wyświetlić: wszystkie drużyny, drużynę tymczasową, wybraną, po ID, drużynę; usunąć drużynę podając jej pełne dane lub podając tylko jej ID.

****

GET /api/teams: Zwraca listę wszystkich drużyn.

GET /api/teams/temp-team: Zwraca drużynę zapisaną w pliku tymczasowym.

PUT /api/teams/temp-team: Zapisuje daną drużynę w pliku tymczasowym. Sprawdzane jest również, czy piłkarz na pozycji bramkarza faktycznie jest bramkarzem. Jeśli nie zwracany jest kod 400  


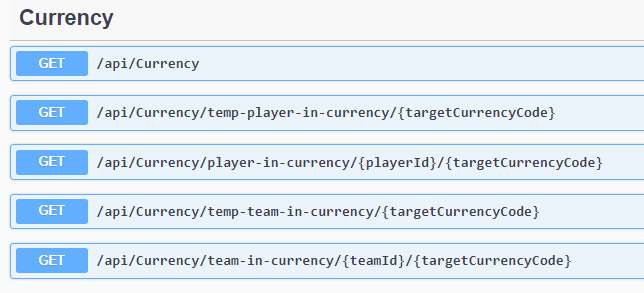
POST /api/teams/save-temp-team: Dodaje drużynę tymczasową do pliku ze wszystkimi drużynami.

GET /api/teams/{team\_id}: Zwraca dane konkretnej drużyny.

DELETE /api/teams/delete-team: Usuwa daną drużynę.

DELETE /api/teams/delete-team/{team\_id}: Usuwa drużynę o danym ID.

**Waluty**Intuicja: Możemy: uzyskać kursy oraz skróty kodowe obsługiwanych walut; przeliczyć wartości piłkarza i drużyny tymczasowej na inną walutę; przeliczyć wartość piłkarza lub drużyny o danym ID na inną walutę (o ile są zapisani w pliku).

****

GET /api/currency: Zwraca aktualne kursy walut z NBP.

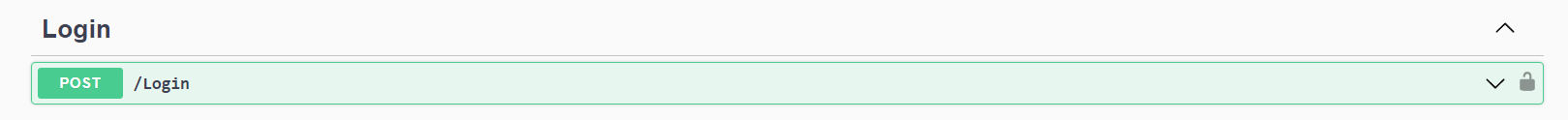
GET /api/currency/temp-player-in-currency/{targetCurrencyCode}: Przelicza wartość tymczasowego piłkarza na podaną walutę.

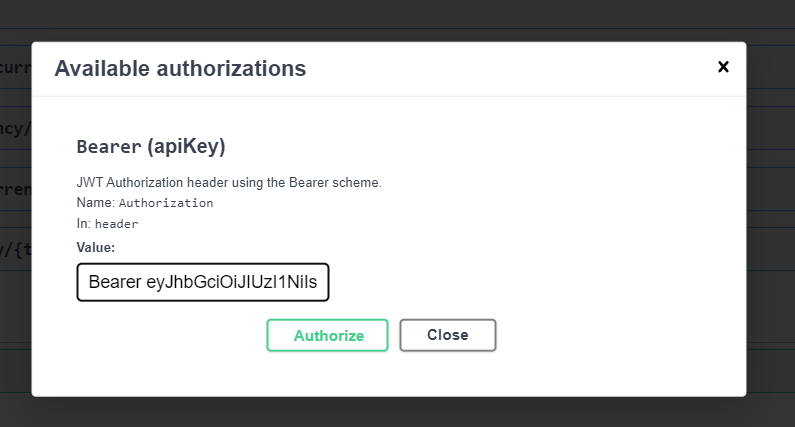
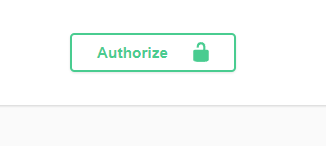
GET /api/currency/player-in-currency/{playerId}/{targetCurrencyCode}: Przelicza wartość wybranego piłkarza na podaną walutę.

GET /api/currency/temp-team-in-currency/{targetCurrencyCode}: Przelicza wartość tymczasowej drużyny na podaną walutę.

GET /api/currency/team-in-currency/{teamId}/{targetCurrencyCode}: Przelicza wartość drużyny na podaną walutę. *(Wymaga autoryzacji)*

**Logowanie**

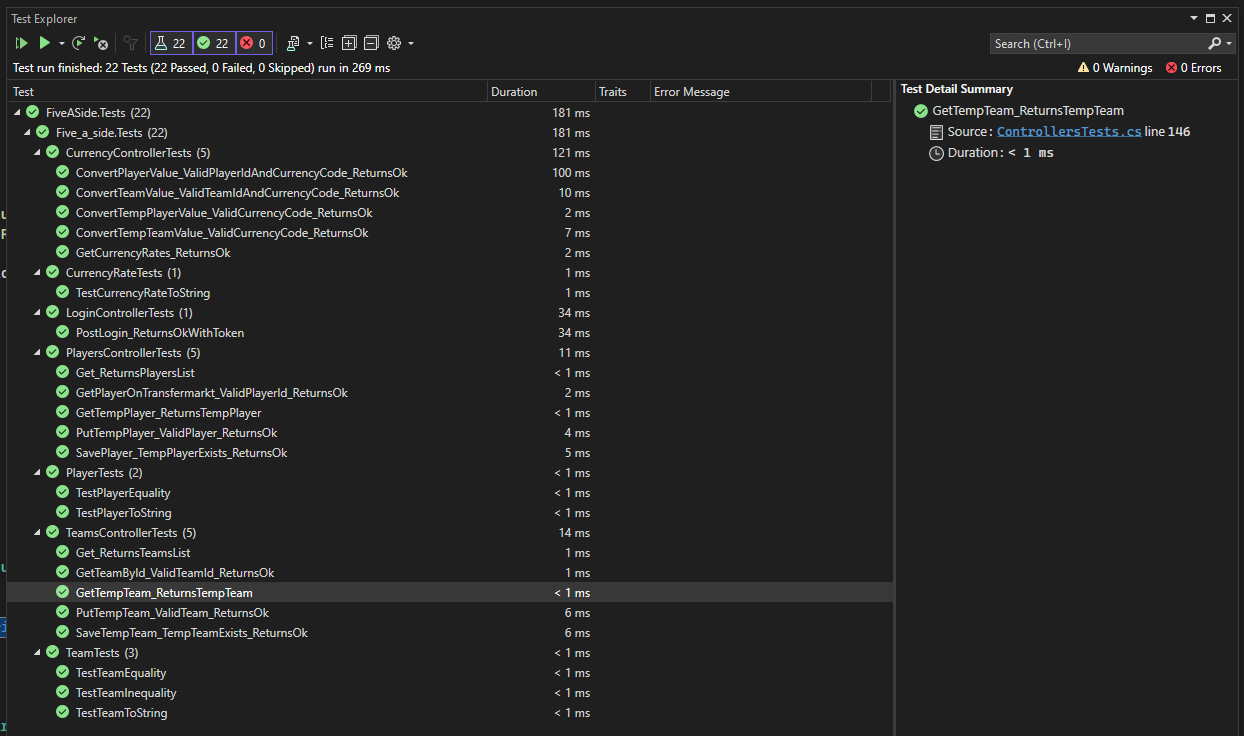
****

POST /Login: Generuje token JWT potrzebny do autoryzacji użytkownika. Można go potem użyć w odpowiednim panelu:  


**3.3 Autentykacja i autoryzacja**

Aplikacja wykorzystuje JWT do autentykacji i autoryzacji użytkowników. Endpoint logowania (POST /login) generuje token JWT, który jest wykorzystywany do uwierzytelniania przy kolejnych żądaniach.

**3.4 Testy jednostkowe**



Testy jednostkowe zostały napisane dla wszystkich głównych funkcjonalności aplikacji.

Testy obejmują:

* Sprawdzenie poprawności modeli: Player, Team, CurrencyRate;
* Sprawdzanie poprawności endpointów graczy i drużyn;
* Testowanie przeliczania wartości graczy i drużyn na różne waluty;
* Weryfikacja działania autoryzacji za pomocą JWT.

**4. Konfiguracja aplikacji**

4.1 Konfiguracja Program.cs

Aplikacja została skonfigurowana z wykorzystaniem technologii ASP.NET Core. Konfiguracja obejmuje m.in. ustawienia autoryzacji JWT, rejestrację kontrolerów oraz konfigurację Swagger do dokumentacji API.

4.2 Użyte biblioteki i narzędzia

ASP.NET Core: Framework do budowy aplikacji webowych.

Newtonsoft.Json: Biblioteka do pracy z JSON.

Moq: Biblioteka do tworzenia mock-ów w testach jednostkowych.

Swagger: Narzędzie do dokumentacji i testowania API.

**5. Implementacja**

**5.1 Piłkarze**

Plik PlayersController.cs zawiera implementację kontrolera odpowiedzialnego za operacje na piłkarzach

**5.2 Drużyny**

Plik TeamsController.cs zawiera implementację kontrolera odpowiedzialnego za operacje na drużynach.

**5.3 Waluty**

Plik CurrencyController.cs zawiera implementację kontrolera odpowiedzialnego za przeliczanie wartości graczy i drużyn na różne waluty, korzystając z API NBP.

**5.4 Autentykacja**

Plik LoginController.cs zawiera implementację kontrolera odpowiedzialnego za generowanie tokenów JWT dla użytkowników.

**6. Testy jednostkowe**

Testy jednostkowe zostały napisane w plikach: ModelssTests.cs; ControllersTests.cs. Przykładowe testy obejmują:

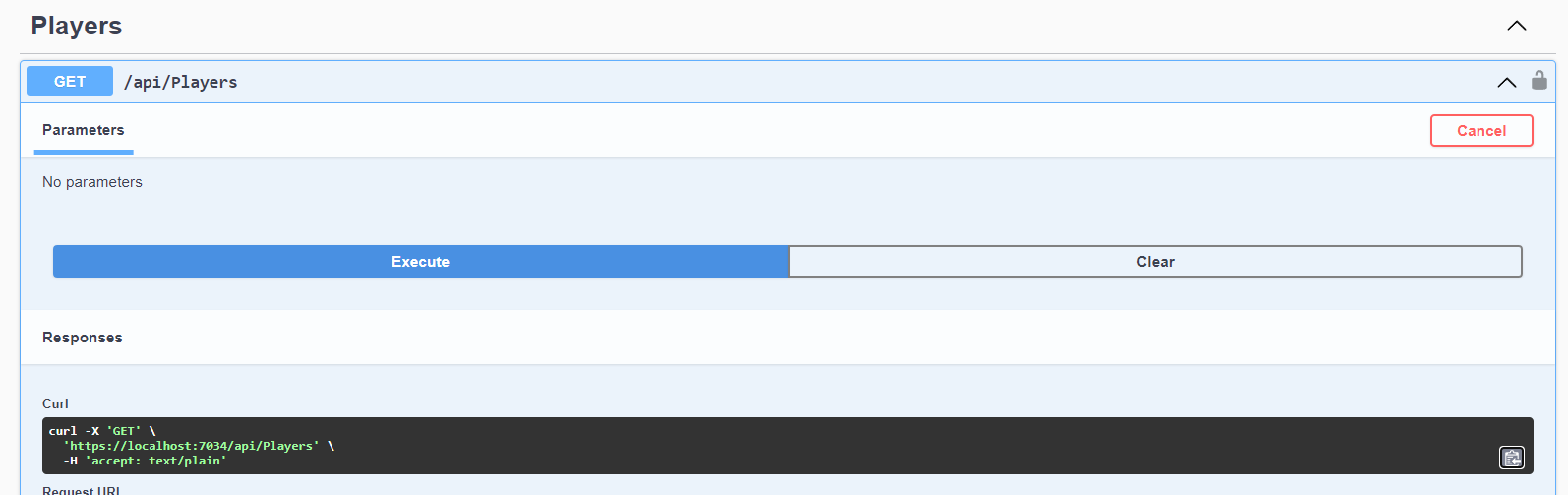
Sprawdzenie poprawności tworzonych modeli, ich porównywania i zapisywania jako ciągi znaków.

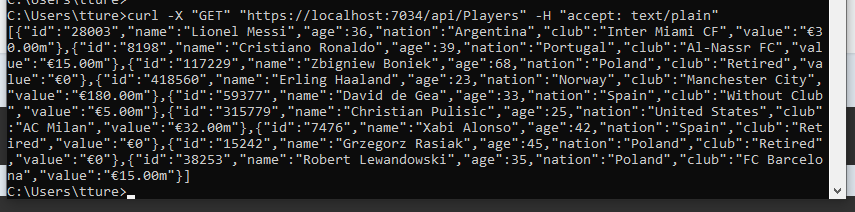
Sprawdzanie poprawności zwracanych danych przez endpointy graczy i drużyn.

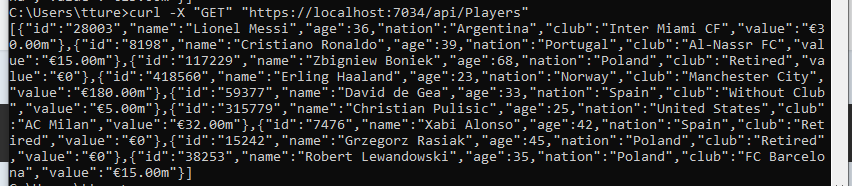
Testowanie przeliczania wartości na różne waluty.

Weryfikacja działania autoryzacji za pomocą JWT.

**7. Obsługa przez terminal**

Dostęp do API Five-a-side może odbywać się również z wiersza poleceń. Wówczas zalecam korzystać z komand sugerowanych w Swagger-ze, np.:  


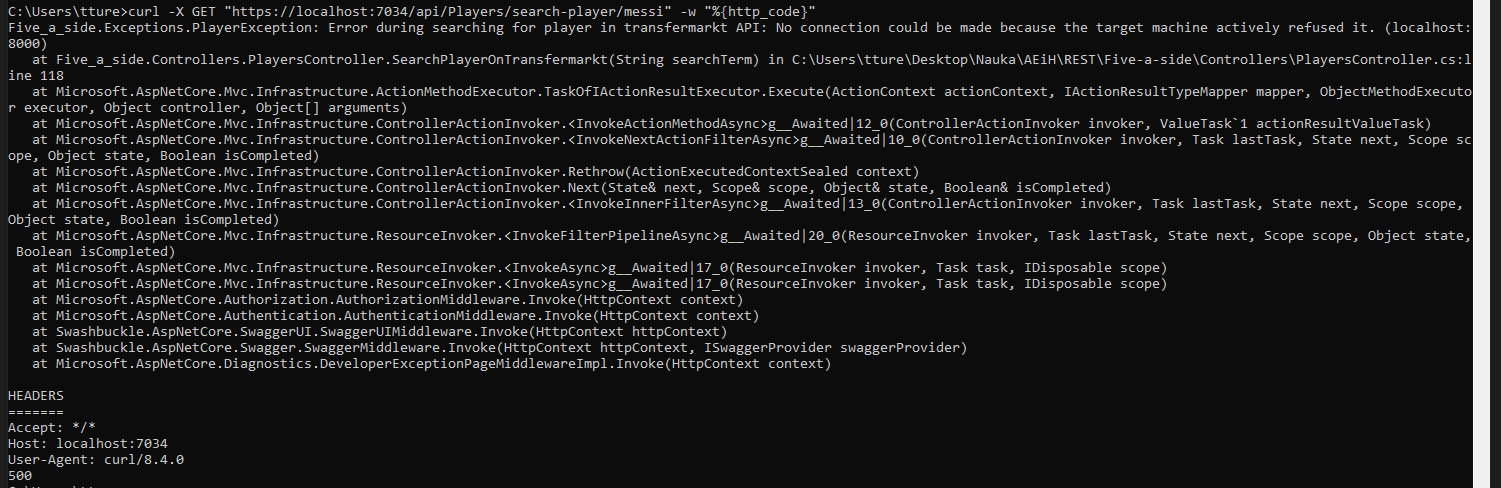


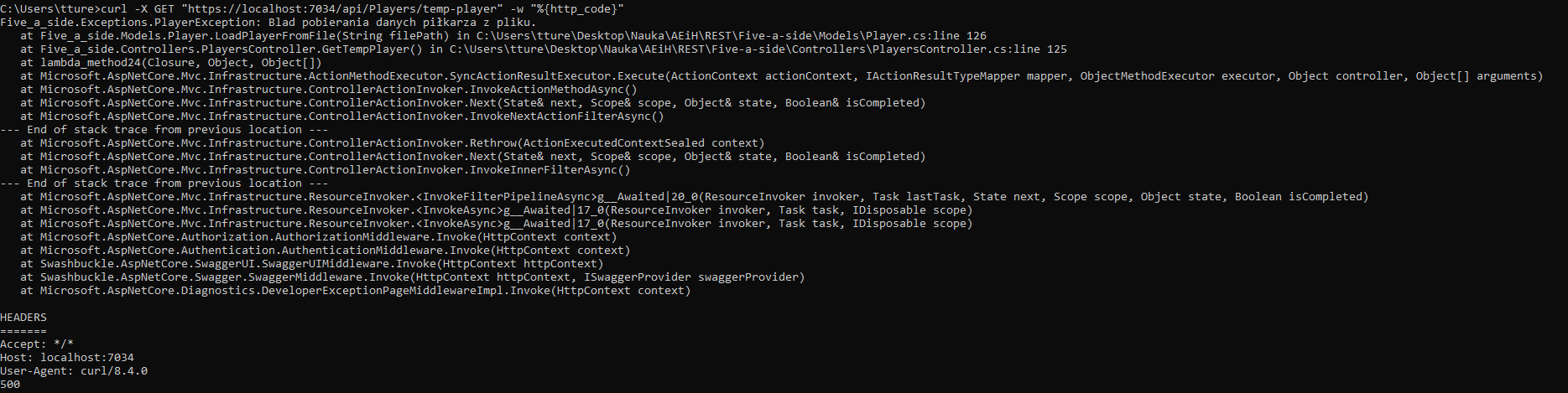
Należy pamiętać o zamianie na cudzysłów w systemie Windows. Można również pominąć flagę ustawiającą Header-y.  


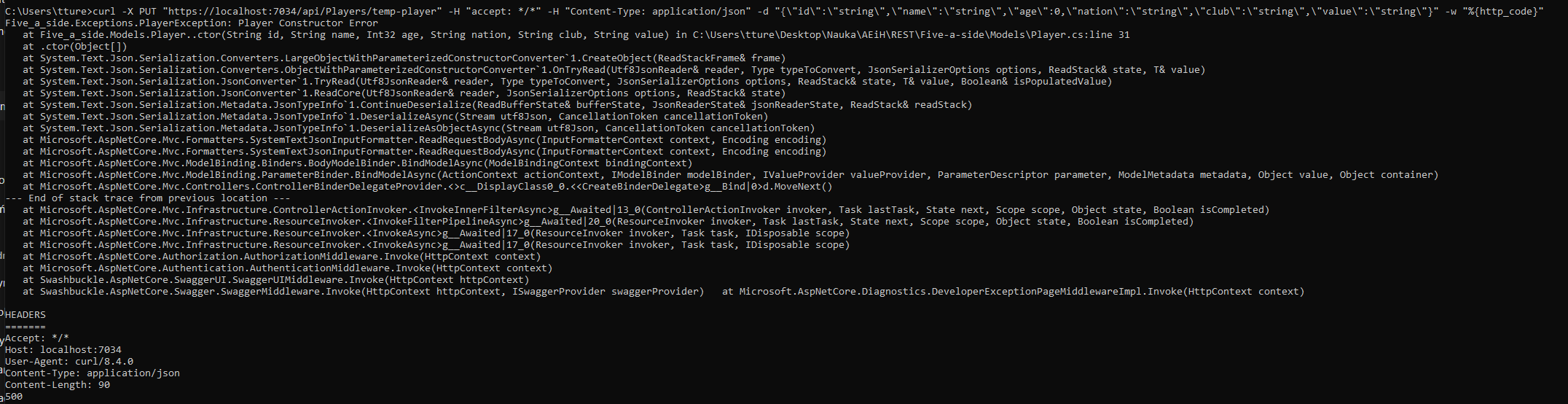
**8. Obsługa błędnych danych**

Wywołanie list piłkarzy, gdy nie ma pliku lub nikogo na liście generuje pustą listę:  

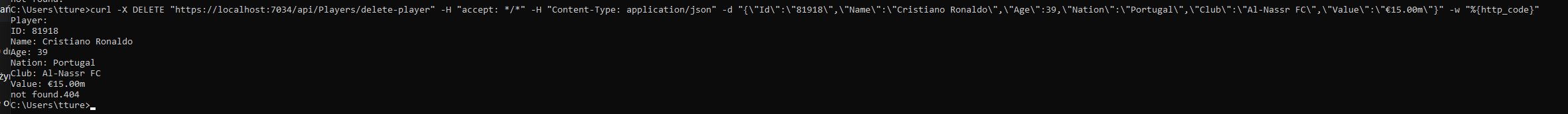
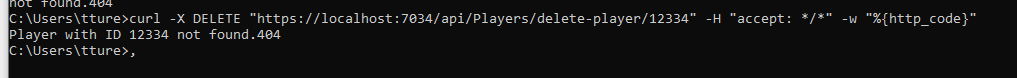

Wyszukanie hasła, które się nie kojarzy z żadnym piłkarzem generuje kod 204:  


Wyszukanie hasła, kiedy serwer jest niedostępny generuje błąd i kod 500:  


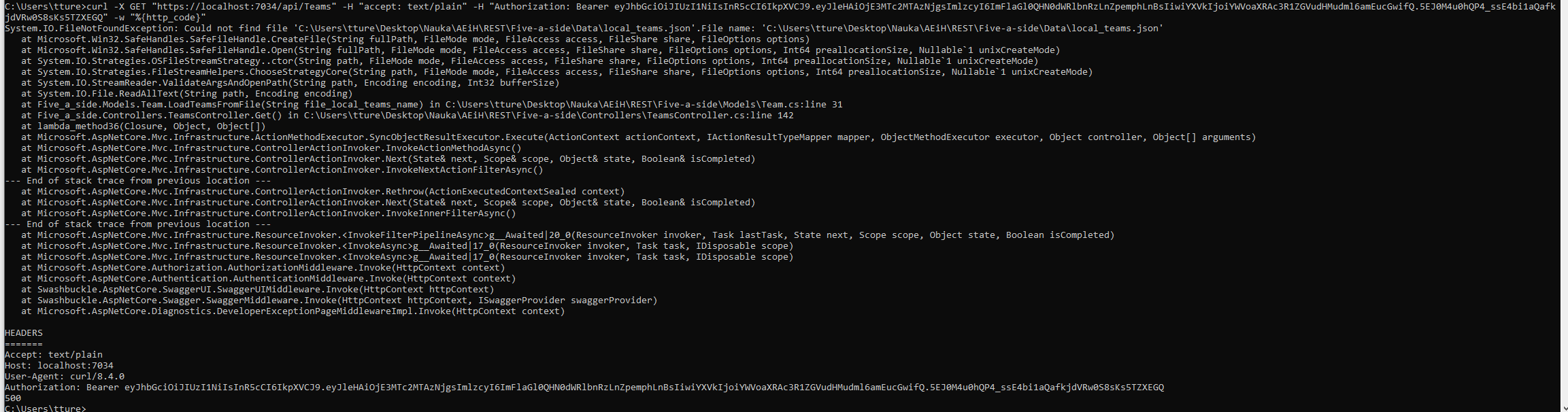
Pobranie piłkarza tymczasowego z błędnymi (z jakiegoś powodu) danymi generuje błąd i kod 500  


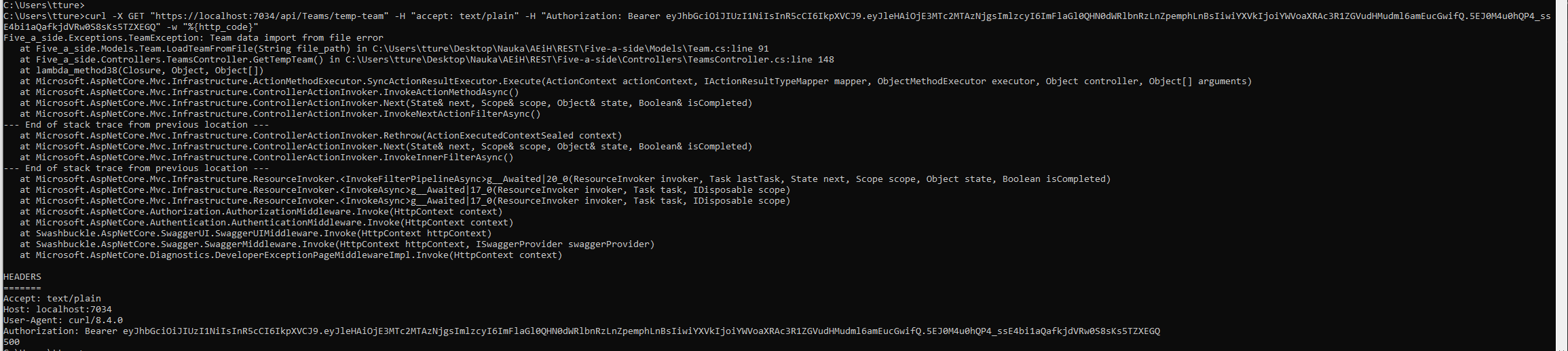
Wstawienie piłkarza o błędnych danych generuje błąd i kod 500  


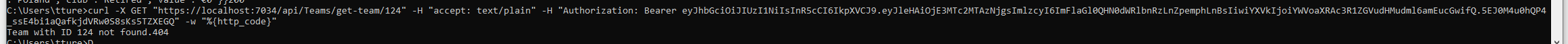
Wyszukanie piłkarza o błędnym numerze ID generuje kod 404  

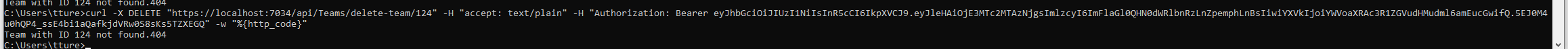

Próba usunięcia piłkarza o danych, które nie są zapisane generuje kod 404  
  


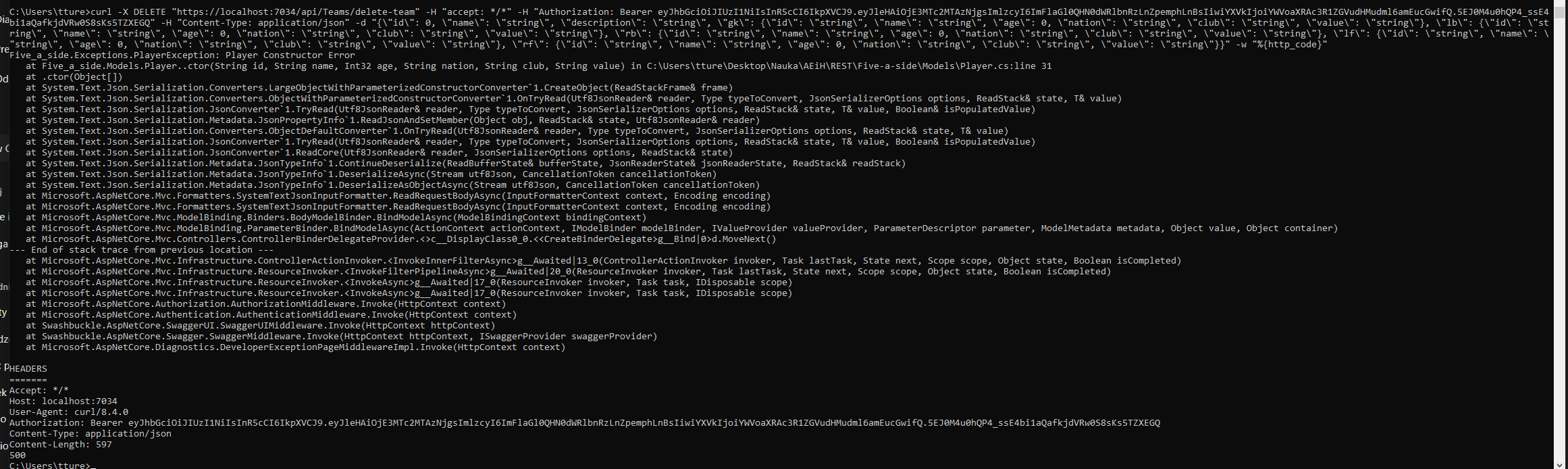
Próba wywołania metod bez autoryzacji generuje kod 401  

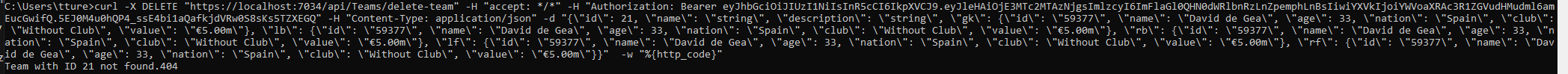

Pobranie drużyn, kiedy nie ma pliku z danymi lub jest on błędnie sformatowany generuje błąd i kod 500  


Pobranie tymczasowej drużyny, kiedy jest ona błędnie sformatowany również generuje błąd i kod 500  


Pobranie drużyny o nieistniejącym ID generuje kod 404  


Usunięcie drużyny o nieistniejącym ID generuje kod 404  


Usunięcie drużyny o nieprawidłowych danych generuje kod 500 lub, jeśli dane zespołu nie pokrywają się z obecnymi w bazie kod 404 i komunikat, że nie znaleziono zespołu z takim ID  




Wywołanie przeliczenia wartości piłkarza na walutę, której kod nie jest obsługiwany (nie występuje na liście od NBP) spowoduje kod 404  


Przeliczeni wartości piłkarza/drużyny spoza listy również spowoduje kod 404  