

**Zaawansowane programowanie w Javie**

Dokumentacja Projektowa

*Projekt: „Gra wieloosobowa ”*

**Autor:**

Wojciech Wnuk

**Prowadzący zajęcia:**

**mgr inż. Albert Rachwał**

Lublin, 2023

Spis treści

[1. Opis projektu 3](#_Toc153320047)

[2. Struktura aplikacji serwerowej oraz klienckiej 4](#_Toc153320048)

[3. Wykorzystane technologię 6](#_Toc153320049)

[4. Implementacja 7](#_Toc153320050)

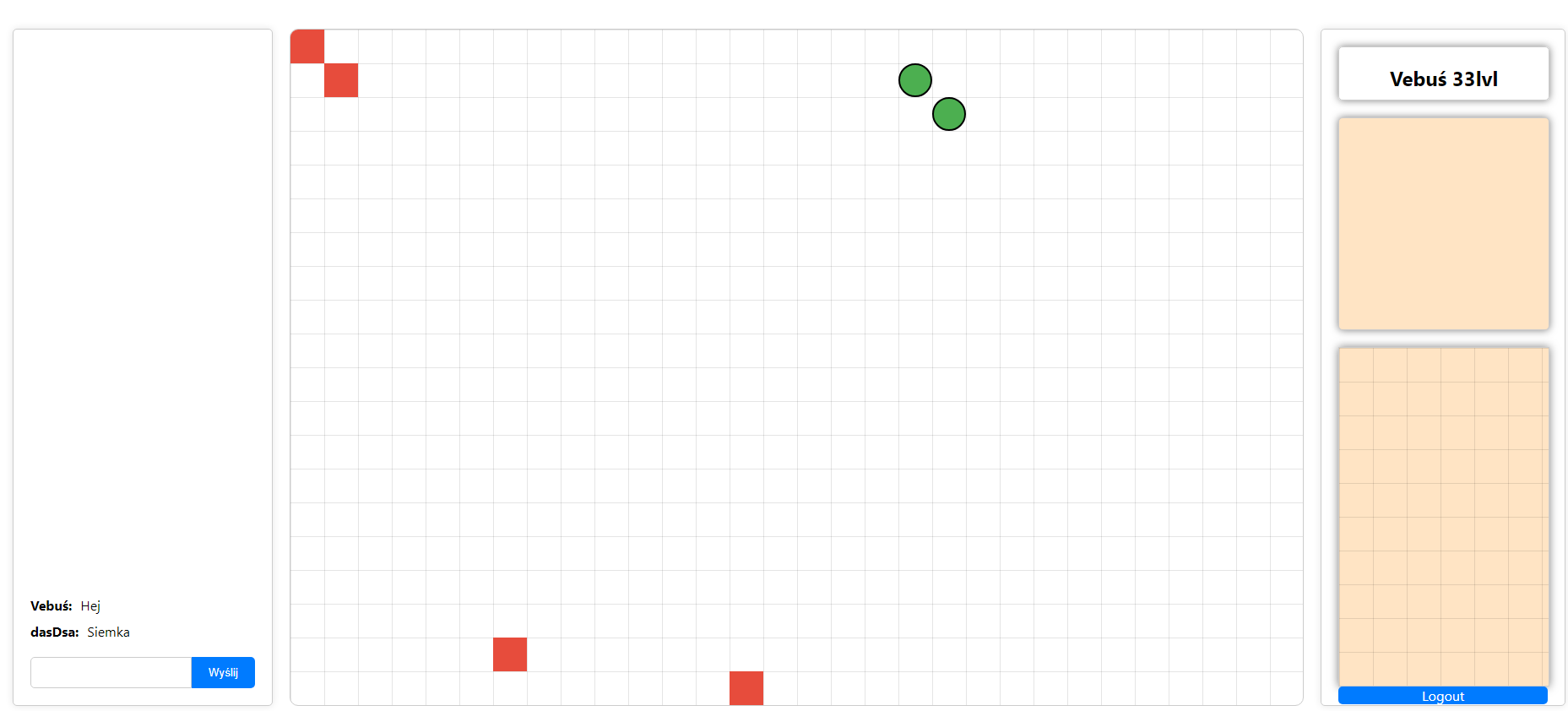
[5. Testy 11](#_Toc153320051)

[6. Podsumowanie 13](#_Toc153320052)

# Opis projektu

Projekt zaliczeniowy jaki zaimplementowano na potrzeby laboratoriów jest prostą grą wieloosobową stworzoną przy użyciu Frameworka Spring na serwerze oraz Frameworka React na kliencie, z wykorzystaniem technologii umożliwiających komunikację w czasie rzeczywistym. Gra została wyposażona w takie widoki jak:

* Widok strony rejestracji
* Widok strony logowania
* Widok dodawania przeciwnika
* Widok gry

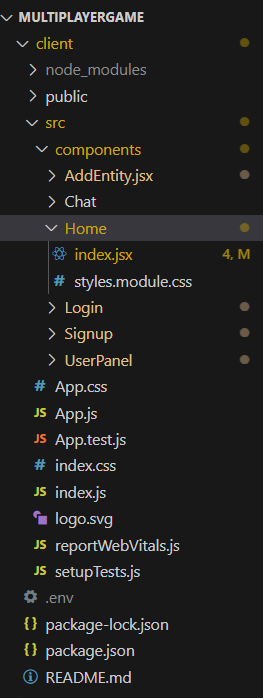


Rys 1. Widok strony gry

Gra polega na prostym zbieraniu punktów przez różnych użytkowników. Każdy gracz posiada swoją postać (zielone kropki), którymi porusza się za pośrednictwem klawiatury. Celem gracza jest zebrać jak najwięcej czerwonych kwadratów, które po zebraniu pojawiają się od nowa na losowo wygenerowanych koordynatach. Całość oczywiście jest widoczna dla wszystkich użytkowników jednocześnie – każdy ruch jakiegokolwiek użytkownika, zebranie czerwonego kwadratu oraz chat są natychmiastowo synchronizowane dla każdego użytkownika korzystając z zaawansowanych technologii w celu zminimalizowania opóźnienia. Dodatkowo użytkownik z rolą „Admin” dostaje opcję takie jak dodawanie nowego „entity” (kwadraty) oraz edycje aktualnego. Dane takie jak pozycje graczy oraz entity są przechowywane na bieżąco w bazie danych lecz w przypadku chatu zachowana jest ulotność danych.

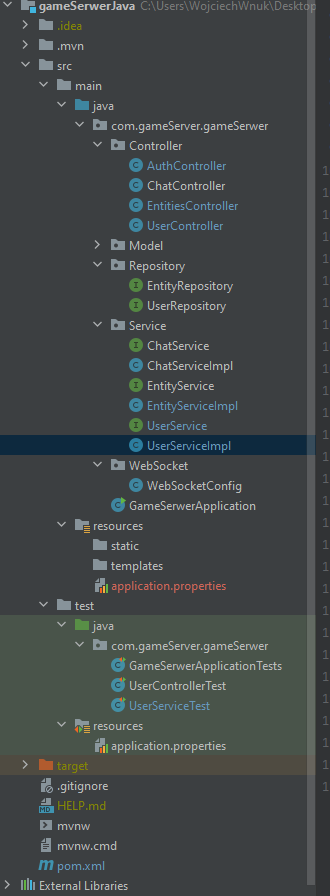
# Struktura aplikacji serwerowej oraz klienckiej

Zaimplementowana aplikacja kliencka podzielona jest na komponenty, cztery główne będące jednocześnie podstronami oraz dwa – Chat oraz UserPanel będące elementami Home.



Rys. 2 Struktura aplikacji klienta

Natomiast aplikacja serwerowa została odpowiednio podzielona na kontrolery, modele, repozytoria, serwisy oraz konfigurację WebSocketa. Dodatkowo w wydzielonej części projektu znajduje się katalog zawierający testy – zarówno jednostkowe jak i integracyjne.



Rys. 3 Struktura aplikacji serwerowej

# Wykorzystane technologię

Do zaimplementowania aplikacji wykorzystano szereg zaawansowanych technologii, skorzystano między innymi z:

* WebSocket
* NodeJS
* JavaScript
* Java
* Spring
* React
* Lombok
* MongoDB
* JUnit

Kluczowymi z wymienionych są Websocket dzięki, któremu można było zaimplementować aktualizacje danych w czasie rzeczywistym oraz MongoDB – szybka i elastyczna baza danych, która pomogła zminimalizować opóźnienie do nieznacznych wartości.

# Implementacja

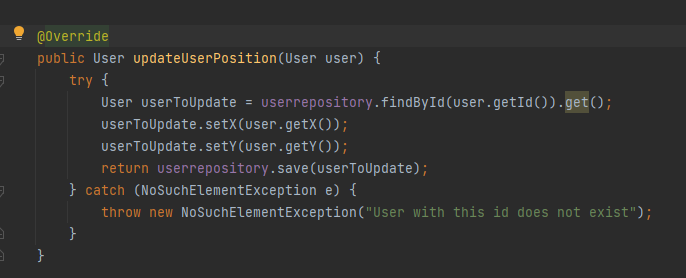
W tej części skupiono się głównie na pokazaniu części serwerowej – w końcu to ona jest tą ważniejszą częścią tego projektu. Jednakże aby dopełnić projekt zaprezentowane zostaną także kluczowe elementy aplikacji klienckiej.

Spośród kontrolerów zostanie omówiony najważniejszy pod względem systemu – UserController w którym wykonywane są wszelkie endpoiny dotyczące użytkownika. Oczywiście omawiany kontroler zawiera wszelkie działania CRUD lecz zostaną omówione jedynie najbardziej rozbudowane punkty końcowe.

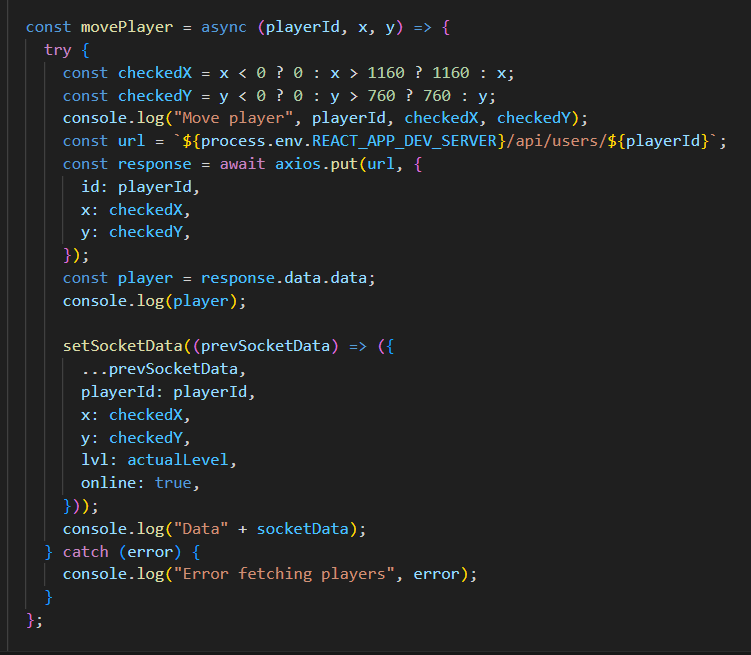


Rys. 4 Endpoint odpowiadający za aktualizacje pozycji graczy

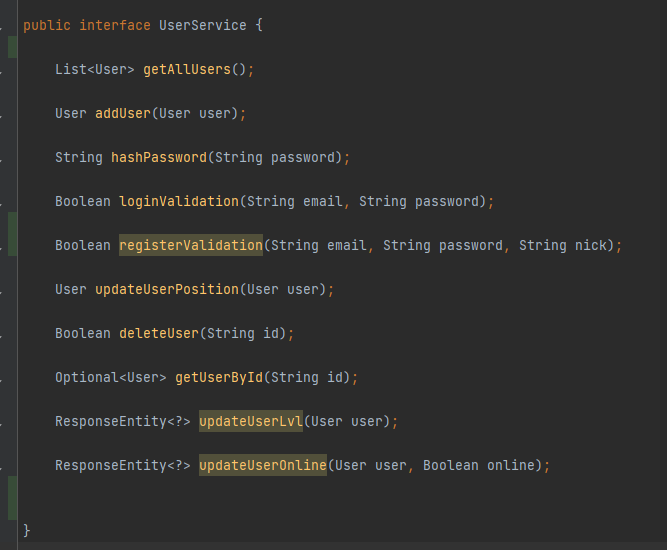
Fragment kodu z rysunku 4 otrzymuje dane użytkownika, a następnie aktualizuje jego pozycje dodatkowo rozsyłając te dane do wszystkich użytkowników pozwalając na dostarczanie aktualnych pozycji graczy.



Rys. 5 Metoda updateUserPosition w UserServiceImpl



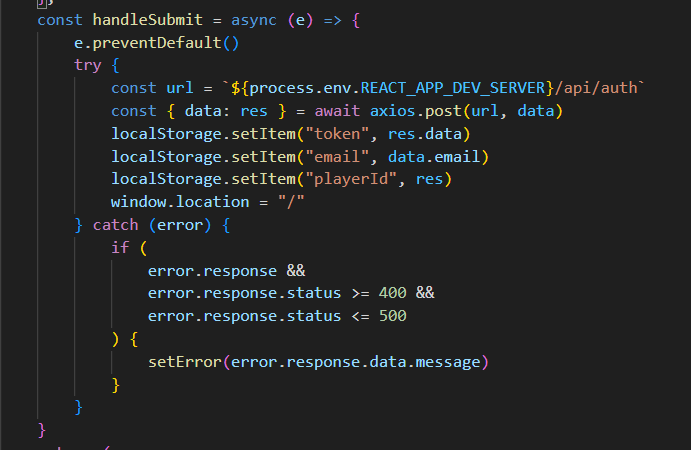
Rys. 6 Obsługa poruszania graczy po stronie serwerowej



Rys. 6 Interfejs UserService

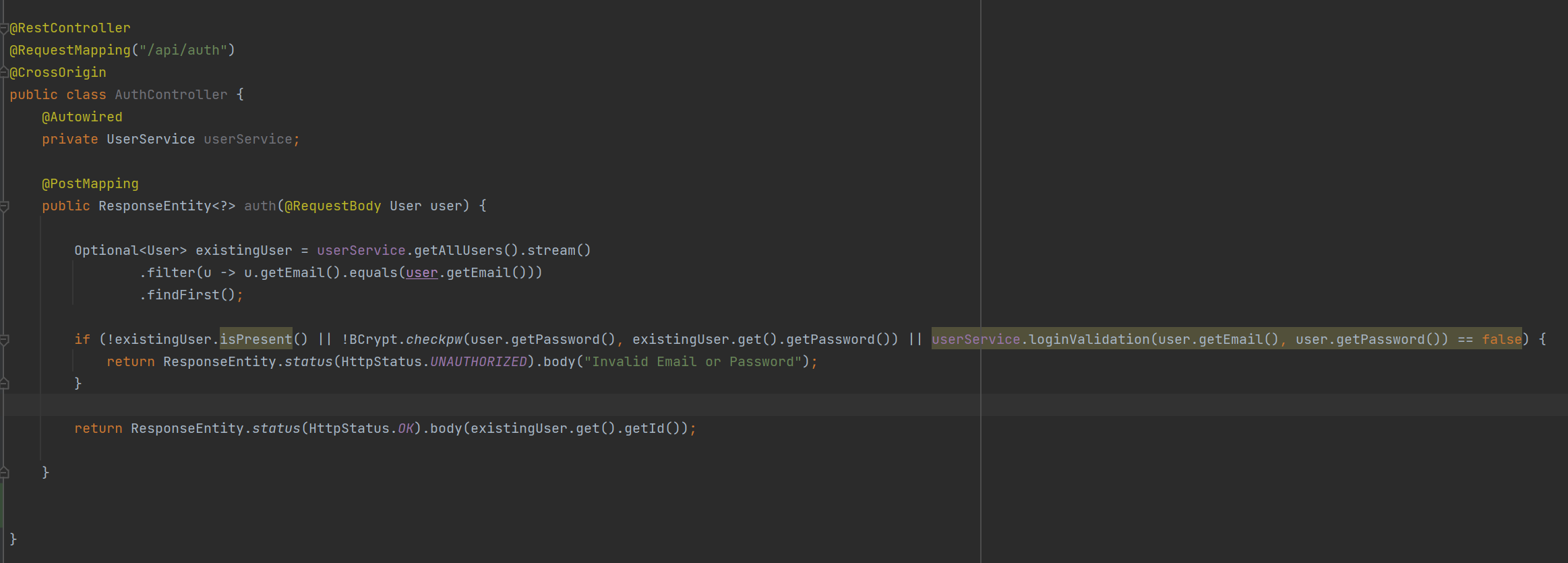
Aplikacja oczywiście została wyposażona w walidacje dla wszelkich danych, które mogą powodować błędy. Do przedstawienia przykładowej walidacji posłużymy się przejściem przez cały proces zalogowania użytkownika.

Po wypełnieniu formularza logowania użytkownik klika przycisk zaloguj powodując wykonanie poniżej przedstawionej akcji:



Rys. 7 Obsługa przycisku zaloguj

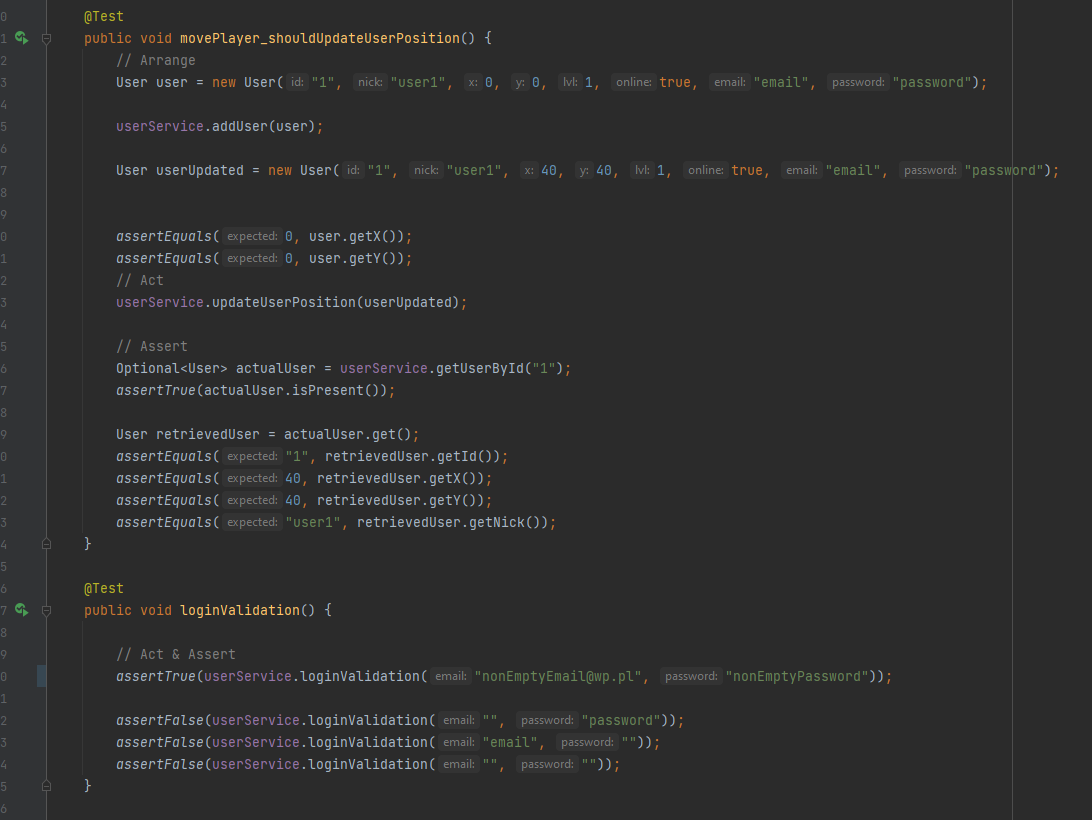
Następnie dane przekazane z formularza lądują na serwerze gdzie odbywa się sprawdzanie danych oraz walidacja.



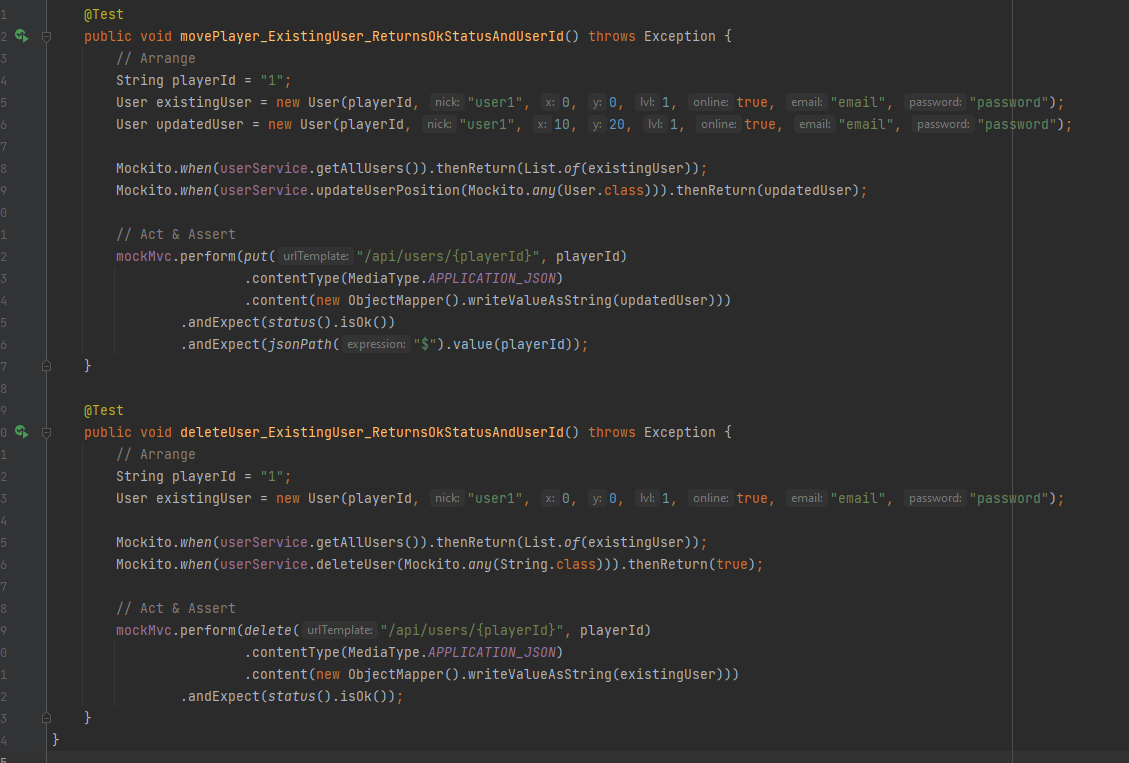
Rys. 8 AuthController

# Testy

Aplikacja serwerowa została wyposażona w testy jednostkowe jak i integracyjne, które są szczególnie ważne przy rozwoju gry, w takim typie aplikacji nie może być miejsca na błędy. Oto przykłady kilku z testów:



Rys. 9 Testy jednostkowe



Rys. 10 Testy integracyjne

# Podsumowanie

Finalnie udało się zrealizować projekt realizujące poniższe wymagania:

* Aplikacja wykonana z wykorzystaniem Frameworka Spring
* Wykorzystanie bazy danych
* Logowanie i rejestracja
* Hashowanie haseł z wykorzystaniem soli i pieprzu
* Operacje CRUD
* Rozbudowane formularze do zapisu oraz edycji danych
* Walidacja danych
* Testy jednostkowe
* Obsługa ról użytkowników\*
* Budowa REST API z oddzielnym front-endem\*\*

A także kilka ponadprogramowych, jedyne co nie zostało uwzględnione – obsługa sesji użytkownika, została zastąpione poprzez użycie ograniczeń nałożonych poprzez aplikacje kliencką.