



Platforma do gier wieloosobowych z elementami społecznościowymi

Dokumentacja specyfikacyjna

WIEiT Informatyka niestacjonarna
Nestor Sapuntsov, Kamil Medoń, Daniel Sopel

Promotor pracy
dr inż. Witold Alda

Spis treści

1. Opis serwisów	3
1.1 Serwis Auth	3
1.2 Serwis Lobby	4
1.3 Serwis GameCore	5
1.4 Serwisy Game	6
1.5 Serwis Social	7
1.6 Serwis Gateway	8
2. Obsługa gier	9
2.1. Podstawowa nomenklatura	9
2.1.1. Pojęcia	9
2.1.2. Akcje	9
2.1.3. Wyjaśnienia dodatkowe	10
2.2. Założenia ogólne	11
2.2.1. Protokół interakcji backend-frontend	11
2.3. Interakcja serwisów	11
2.3.1. Rozpoczęcie gry	11
2.3.2. Prowadzenie gry	12
3. Specyfikacja struktury bazy danych	13
3.1. Ogólny opis bazy danych	13
3.2. Opis kolekcji	13
3.2.1. Users	13
3.2.2. Games	13
3.2.3. Tables	13
3.3. Graficzne przedstawienie	13
4. Ekrany	14
2.4.1 Ekran logowania	14
2.4.2 Ekran Rejestracji	14

1. Opis serwisów

1.1 Serwis Auth

Ten serwis został stworzony w celu zagwarantowania odpowiednich procedur logowania, które są kluczowe dla zapewnienia bezpieczeństwa systemu oraz ochrony poufności, integralności i autoryzowanego dostępu do danych.

Dane wejściowe :

- Login : “user123”
- Hasło : “password123”

Dane wyjściowe (udane logowanie) :

- Status logowania : “Powodzenie”
- Token weryfikacyjny

Dane wyjściowe (nieudane logowanie) :

- Status logowanie: “Niepowodzenie”
- Informacja o nieudanym logowaniu

Zalecany sposób działania:

- Serwis będzie w ścisłej współpracy z serwisem Gateway

Dodatkowe informacje specyfikacyjne:

- Użytkownik aby móc korzystać z apki musi się zarejestrować i zalogować. Bez tego nie będzie miał dostępu do serwisu.
- Użytkownik po udanym zalogowaniu ma dostęp do swoich danych, może usunąć swoje konto.

1.2 Serwis Lobby

Ten serwis jest odpowiedzialny za funkcje, towarzyszące rozgrywkom, takie jak połączenie graczy przed rozpoczęciem gry, wybór pożądanej gry, rozpoczęcie nowej gry, czat.

Dane wejściowe :

- Dołączone użytkownicy
- Wybrana przez użytkowników gra
- Wysyłane na czacie wiadomości

Dane wyjściowe:

- Wiadomości na czacie (przekazywane do innych użytkowników w lobby)
- Identyfikator rozpoczętej gry

Zalecany sposób działania:

- Serwis będzie w ścisłej współpracy z serwisami GameCore oraz Social

Dodatkowe informacje specyfikacyjne:

- Użytkownik aby móc zagrać w jakąś grę musi najpierw stworzyć lub dołączyć się do lobby

1.3 Serwis GameCore

Ten serwis odpowiada za przeprowadzanie gier, realizując podstawowe funkcjonalności obsługi i zapisywania ruchów użytkowników, a także używając metody wymagane od serwisów konkretnych gier. W zależności od wybranej przez graczy gry, dany serwis będzie współpracował z serwisem gry, używając go do weryfikacji wykonywanych ruchów oraz do obliczenia możliwych dalszych ruchów. Odpowiednie informacje o powodzeniu ruchu, możliwych dalszych ruchach, stanie stołu, rąk itd. są udostępniane połączonym klientom przez dany serwis.

Dane wejściowe:

- Potoczny stan stołu, rąk graczy
- Wykonywany ruch

Dane wyjściowe (wysyłane do odpowiednich klientów):

- Informacja o powodzeniu / niepowodzeniu ruchu
- Nowy stan stołu, rąk graczy
- Informacja o zakończeniu gry, jeśli takowe nastąpiło

1.4 Serwisy Game

Każdy z takich serwisów ma na celu realizować funkcjonalności dotyczące konkretnych gier i udostępniać te funkcjonalności serwisowi GameCore. Mianowicie ma udostępnić metody do:

- inicjalizacji gry (ustawienie stanu początkowego, rozdanie kart itp.)
- wykonywanie ruchu (wraz ze sprawdzeniem poprawności ruchu i warunków zakończenia gry)

W zależności od gry i jej reguł te metody mogą być w różny sposób zaimplementowane.

W rozwiązaniu końcowym takich serwisów jest tyle, ile jest zaimplementowanych gier.

Dane wejściowe:

- Potoczny stan stołu, rąk graczy
- Wykonywany ruch

Dane wyjściowe (udostępniane do GameCore):

- Informacja o powodzeniu / niepowodzeniu ruchu
- Informacja o zakończeniu gry, jeśli takowe nastąpiło

1.5 Serwis Social

Serwis będzie odpowiedzialny za obsługę kontaktów użytkowników, zarządzanie historią i wysyłaniem wiadomości (czat).

Dane wejściowe :

- Nazwa odbiorcy

- Wiadomość

Dane wyjściowe :

- Odpowiedź odbiorcy
- Obrazek
- Zaproszenie do gry
- Prezentacja rankingu

Zalecany sposób działania modułu :

- Moduł będzie w ścisłej współpracy z serwisem GameCore

Dodatkowe informacje specyfikacyjne:

- Użytkownik będzie mógł bezpośrednio wysyłać wiadomości do innych użytkowników
- Użytkownik będzie miał wgląd w historię czatów i Czaty grupowe

1.6 Serwis Gateway

Ten moduł będzie odpowiedzialny za pojedyncze i skonsolidowane weryfikację danych na backendzie. Tylko ten serwis będzie dostępny z zewnątrz. Głównie w ramach autoryzacji.

Dane wejściowe :

- Dane logowania użytkownika

Dane wyjściowe :

- Status tokenu

- Docelowy adres
- Ewentualne dodatkowe dane

Dane wyjściowe (brak dostępu) :

- Status wejścia : “Niepowodzenie”
- Informacja o braku odpowiedniej ilości uprawnień
- Ewentualny opis błędu

Zalecany sposób działania modułu :

- Moduł będzie w ścisłej współpracy z wszystkimi modułami aplikacji, głównie modułem Auth

Dodatkowe informacje specyfikacyjne:

- Użytkownik każdą akcję będzie wykonywał przez moduł Gateway
- Gateway będzie mógł w przyszłości służyć jako pewnego rodzaju load balancer
- Gateway będzie mógł wstępnie zweryfikować nie tylko dane autoryzacyjne ale ogólne dla dalszej logiki biznesowej

2. Obsługa gier

2.1. Podstawowa nomenklatura

W celu maksymalnego zuniwersalizowania podejścia i uogólnienia wykorzystywanych metod, narzędzi, sposobu przechowywania i przetwarzania danych itd. przy implementacji wprowadza się pojęcia oraz akcje, wspólne dla wszystkich (a przynajmniej dla zdecydowanej większości) gier karcianych, przedstawione poniżej.

2.1.1. Pojecia

- Stół
To jest pewna przestrzeń, gdzie odbywa się rozgrywka

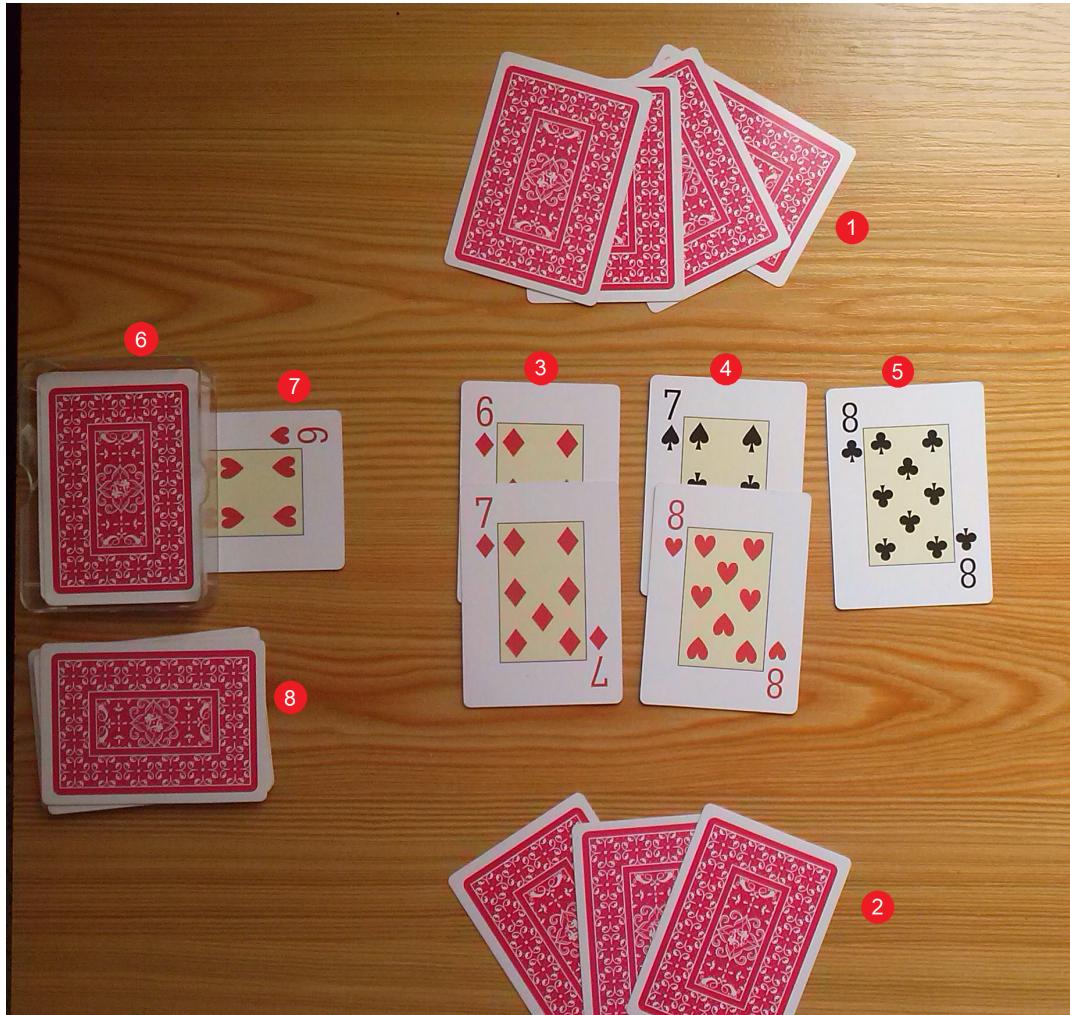
- **Ręka**
Jest zbiorem kart, należącym do pewnego gracza. Użytkownik ma bezpośredni wpływ na swoją rękę, może (w zależności od reguł gry) wykładać lub zabierać karty do ręki, otwierać lub zamykać swoją rękę itd.
- **Stos**
Stos jako byt - to jest samodzielny sekwencyjny układ kart na stole (nawet składający się z jednej karty). Stosy mogą być tworzone lub usuwane w trakcie gry n.p. jako wynik wykonywania ruchów. Karty z jednego stosu mogą być przeniesione do innego, także karty ze stosu mogą być przeniesione do ręki lub na odwrót. Stos może być otwarty lub zamknięty (postawiony grzbietem do dołu lub góry)
- **Karta**
Byt mający jakąś wartość lub znaczenie. Nie występuje na stole samodzielnie, tylko w składzie stosu lub ręki

2.1.2. Akcje

- Przenieść kartę z dowolnego stosu do ręki
- Przenieść kartę z ręki do dowolnego stosu
- Przenieść kartę z jednego stosu do innego
- Otworzyć/zamknąć rękę
- Otworzyć/zamknąć stos
- Stworzyć/usunąć stos
- Potasować stos

2.1.3. Wyjaśnienia dodatkowe

Wykorzystując rozgrywkę w durnia jako przykład dane podejście jest wyjaśnione poniżej



1, 2 - ręce

3, 4, 5 - stosy ataku/bicia (w danym przypadku czasowe)

6 - stos kart do pobrania

7 - stos dodatkowy (w przypadku durnia koncepcyjnie stanowi część stosu do pobrania, jednak technicznie mógłby być reprezentowany jako osobny stos jako byt)
8 - stos kart odrzuconych

W takim układzie wyimaginowana sekwencja gry wygląda następująco:

1. Gracz dolny robi ruch, wykładając szóstkę. Z punktu widzenia technicznego to oznacza:
 - a. Tworzenie nowego stosu (3)
 - b. Przeniesienie karty z ręki do stosu 3

2. Gracz górny bije szóstkę siódemką:
 - a. Przeniesienie karty z ręki do stosu 3
3. Gracz dolny dokłada siódemkę. Z punktu widzenia technicznego to oznacza:
 - a. Tworzenie nowego stosu (4)
 - b. Przeniesienie karty z ręki do stosu 4

Koniec rundy w takim układzie składałby się z przeniesienia wszystkich kart ze stosów 3-5 do stosu 8 (zakładając że gracz górny bije ósemkę) i usunięcia tych stosów. Następnie przed następną rundą odbywa się przeniesienie odpowiedniej ilości kart ze stosu 6 do rąk graczy.

Zakłada się że wymienionych powyżej podstawowych encji i akcji wystarczy do realizacji dowolnych gier karcianych. W zależności od poszczególnej gry i jej reguł niektóre akcje będą dostępne do wykonywania przy konkretnym stanie stołu, a niektóre nie.

2.2. Założenia ogólne

2.2.1. Protokół interakcji backend-frontend

W przypadku gier wieloosobowych istnieje konieczność zapewnienia tego, że wszyscy uczestnicy posiadają informację o aktualnym stanie gry. W danym przypadku to znaczy że jeśli jeden użytkownik zrobi którykolwiek ruch, czym zmieni stan stołu, to pozostali klienci (w tym przypadku klient - to aplikacja służąca frontendem) muszą również na bieżąco odświeżyć stan gry u siebie. Najbardziej sensownym rozwiązaniem tu jest wybranie protokołu WebSocket, polegającego na otwarciu długotrwałego kanału, na którym istnieje możliwość wysyłania wiadomości w obie strony w dowolnym momencie.

2.3. Interakcja serwisów

Poniżej jest opisana kolejność i sposób możliwych interakcji serwisów w ramach obsługi gry.

2.3.1. Rozpoczęcie gry

Będąc połączeni do konkretnego “pokoju”, czyli lobby (użytkownik uważa się połączonym do lobby w przypadku, gdy ma otwarte połączenie WebSocket z serwisem Lobby), użytkownicy ustalają grę, w którą chcieliby zagrać.

W momencie, gdy wszyscy sygnalizują o tym, że są gotowi zaczynać, serwis Lobby, używając odpowiednią metodę serwisu GameCore i przekazując informację o wybranej grze i uczestnikach, zaczyna nową grę. Uzyskując odpowiedź od GameCore, Lobby przekazuje do użytkowników informacje do połączenia się z grą.

Użytkownik (a de facto jego klient - aplikacja frontendowa) wykonuje połączenie WebSocket z serwisem GameCore. W momencie, gdy wszyscy klienci są podłączeni, GameCore oblicza stan początkowy stołu, używając do tego metody inicializacji gry odpowiedniego serwisu Game, i udostępnia informacje o stanie stołu odpowiednim klientom (n.p. nie udostępniając zawartości zamkniętych rąk innym uczestnikom).

2.3.2. Prowadzenie gry

W ramach gry użytkownicy mogą wykonywać ruchy, składające się z dowolnej ilości i złożonych w dowolny sposób podstawowych akcji. Użytkownicy wysyłają swoje pożądane ruchy do GameCore na otwartym połączeniu WebSocket. Odpowiedzialnością GameCore tuż jest odpowiednia obsługa kolejności ruchów (n.p. gdy kilka użytkowników "jednocześnie" zadecydują na jakiś ruch).

W momencie, gdy GameCore otrzymuje od klienta pożądany ruch, on najpierw sprawdza czy ten ruch da się wykonać. Dla tego on używa metody do wykonania ruchu odpowiedniego serwisu Game, przekazując stan stołu i sam ruch.

Jeśli ruch się udaje, GameCore przekazuje informację o sukcesie z powrotem do klienta, zapisuje nowy stan stołu i udostępnia go pozostałym klientom wraz z informacją o wykonanym ruchu.

Jeśli ruch jednak się nie udaje, to z powrotem do klienta przekazuje informację o niepowodzeniu.

W przypadku, gdy po udanym wykonaniu ruchu gra się kończy (o czym ma zasygnalizować serwis Game w swojej odpowiedzi na wywołanie metody do wykonania ruchu), GameCore oprócz wymienionych powyżej informacji udostępnia również do klientów informacje o wynikach gry.

3. Specyfikacja struktury bazy danych

3.1. Ogólny opis bazy danych

Projekt struktury bazy znacznie różni się od relacyjnej ze względu na możliwą różnorodność danych (spowodowaną przez liczną ilość gier). Z tego powodu docelowo wybrana została baza MongoDB która nie wymaga z góry narzuconych struktur tabel. Skupiając się na głównych elementach aplikacji przewidywane są opisane poniżej kolekcje.

3.2. Opis kolekcji

3.2.1. Users

Dokumenty danej kolekcji zawierają dane logowania użytkownika, podstawowe dane typu imię, udostępnione kontakty, lista znajomych (referencje do innych użytkowników) itd.

3.2.2. Games

Zawiera informacje o grach, uczestnikach, wynikach gry.

3.2.3. Tables

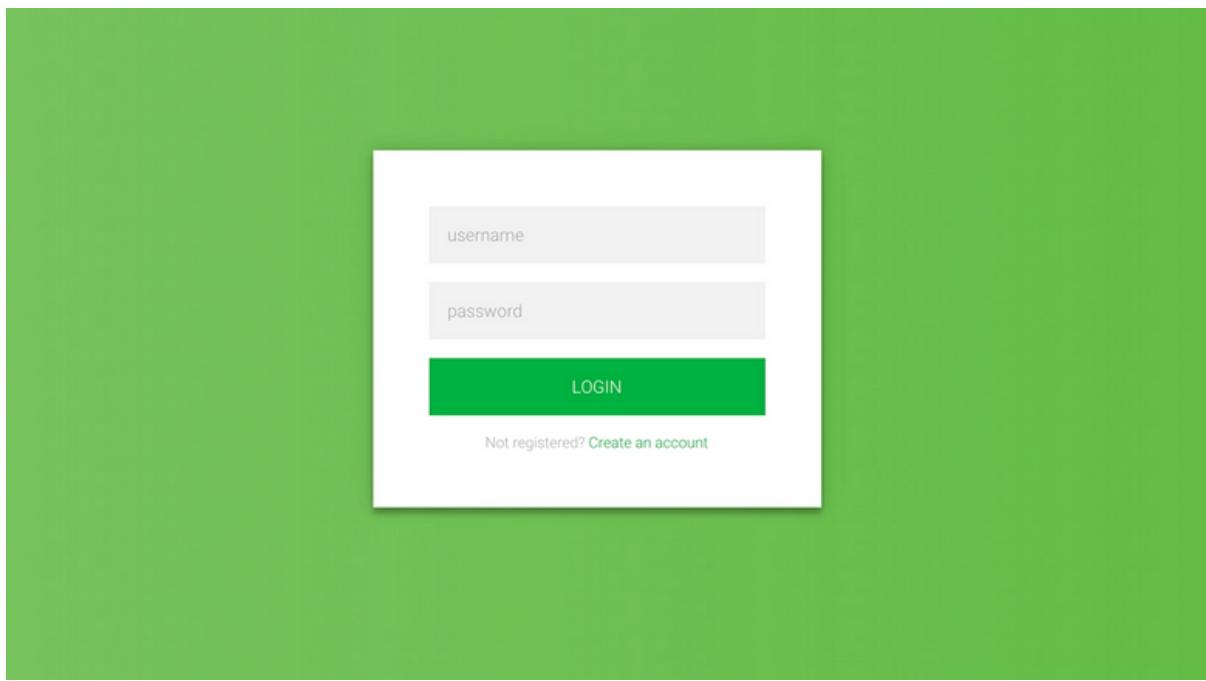
Zawiera informacje o potocznych stanach stołów podczas obsługi gier.

3.3. Graficzne przedstawienie

Ze względu na specyfikę nierelacyjnej bazy danych schemat bazy nie jest wymagany.

4. Ekrany

2.4.1 Ekran logowania



2.4.2 Ekran Rejestracji

