Projekt modułu

Ecology Game

Moduł zarządzania użytkownikami (UMM)

Amadeusz Sitnicki
Adam Rosiak

Inżynieria Oprogramowania
poniedziałek, 12:00

2023/24 Z

Cel i założenia

Cel

Udostępnienie użytkownikom oraz pozostałym modułom systemu informatycznego prostych interfejsów dla mechanizmów uwierzytelniania klientów, cyklu życia sesji użytkownika oraz innych aspektów związanych z zarządzaniem użytkownikami.

Założenia

- Moduł ten odbiera od pozostałych elementów systemu informacje takie jak dane użytkowników, ich postęp w grze, aktywność na forum

- Ten moduł umożliwia zewnętrznym w stosunku do niego komponentom utrwalanie danych generowanych podczas użytkowania aplikacji, pośrednicząc w komunikacji z bazą danych

- Jest to jedyny moduł systemu, który wchodzi w bezpośrednią interakcję z bazą danych

Wymagania projektowe

Wymagania funkcjonalne - użytkownika

- Użytkownik może zarejestrować się w systemie
  - Pozwala na utworzenie konta użytkownika powiązanego z danymi podanymi przez użytkownika podczas rejestracji
- Użytkownik może zalogować się do do systemu
  - Pozwala na utworzenie sesji użytkownika zalogowanego i umożliwia użytkownikowi dostęp do funkcji systemu niedostępnych dla użytkownika niezalogowanego
- Użytkownik może edytować część danych powiązanych z jego kontem
  - Pozwala na aktualizację danych takich jak imię, nazwisko oraz adres mailowy
- Użytkownik może wyświetlić profil innego użytkownika lub swój własny
  - Ułatwia graczom odkrywanie społeczności użytkowników systemu
- Użytkownik może dodać innego użytkownika do listy kontaktów
  - Tworzy powiązanie/relację z innym użytkownikiem systemu, która wprowadza dodatkowe możliwości interakcji między dwoma użytkownikami
- Użytkownik może usunąć innego użytkownika z listy kontaktów
  - Usuwa powiązanie/relację z innym użytkownikiem systemu
  - Blokuje dodatkowe możliwości interakcji między dwoma użytkownikami
- Użytkownik może wylogować się z systemu
  - Pozwala na usunięcie sesji użytkownika zalogowanego oraz związanego z nią klucza sesji
- (Opcjonalne) Użytkownik, który zaakceptował wybrane kanały komunikacji otrzymuje powiadomienia o aktywności w systemie w formie powiadomień *push* i/lub na skrzynkę mailową

Wymagania projektowe c.d.

Wymagania funkcjonalne – międzymodułowe

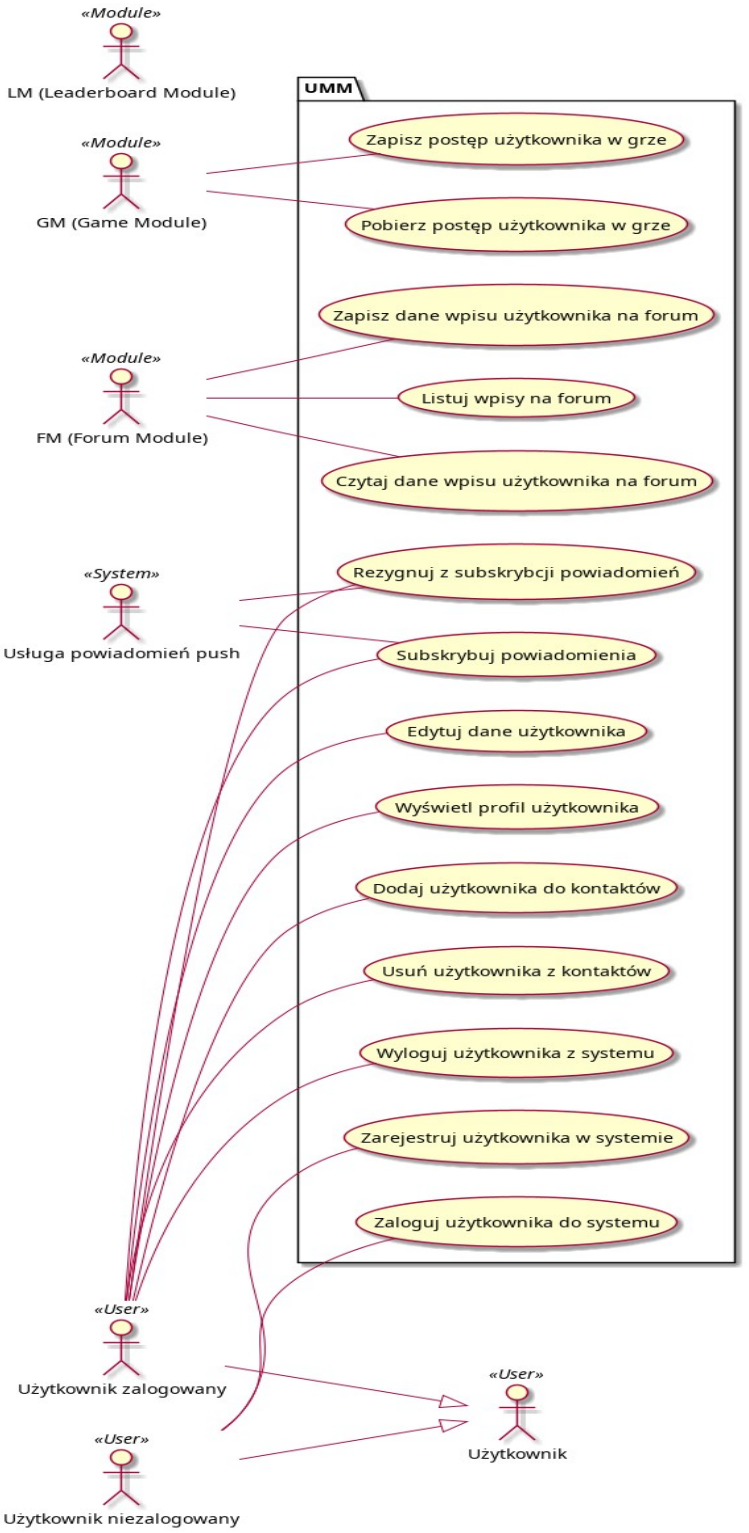
- Wprowadzenie prostego interfejsu udostępniającego dane użytkownika dla pozostałych modułów
  - Moduł pośredniczy w komunikacji z bazą danych
  - Moduł obsługuje współdzielony dostęp do utrwalonych danych
- Wprowadzenie interfejsu utrwalania danych dla modułu “Game Module”
  - Umożliwienie modułowi „Game Module” zapis i odczyt utrwalonych w bazie danych informacji dotyczących postępu użytkownika w grze
- Wprowadzenie interfejsu utrwalania danych dla modułu “Forum Module”
  - Umożliwienie modułowi “Forum” zapis i odczyt utrwalonych w bazie danych informacji dotyczących wpisów na forum
- (Opcjonalne) Wprowadzenie interfejsu dostępu do strumienia zdarzeń aktualizacji stanu użytkownika
  - Umożliwia reagowanie modułów aplikacji na zdarzenia w czasie rzeczywistym
- (Opcjonalne) Wprowadzenie interfejsu udostępniającego możliwość wysyłania powiadomień *push* i/lub wiadomości skrzynki mailowej do użytkowników, którzy wyrazili na to zgodę
  - Zwiększenie liczby kanałów komunikacji między systemem a użytkownikiem

Wymagania projektowe c.d.

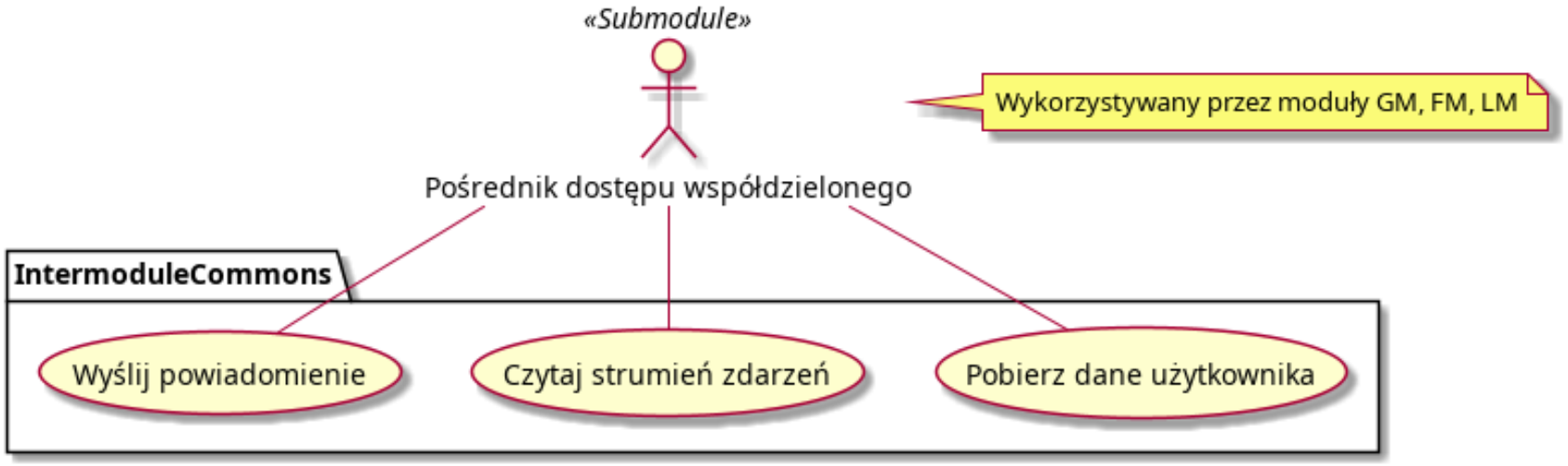
Wymagania niefunkcjonalne

- Bezpieczeństwo danych
  - hasła są przechowywane w bezpiecznej formie
- Bezpieczeństwo użytkowników
  – zmiany danych profilu są możliwe tylko dla użytkownika, który posiada aktywną sesję użytkownika zalogowanego powiązanego z profilem.
- Zabezpieczenie przed przechwyceniem danych
  - skonfigurowane jest bezpieczne szyfrowane połączenie z wykorzystaniem kryptografii asymetrycznej
  - połączenie niezabezpieczone szyfrowaniem z wykorzystaniem kryptografii asymetrycznej jest odrzucane przez system
- Integracja projektu z bazą danych SQLite w celu przechowywania danych użytkowników
  - przechowywanie danych profilu użytkownika oraz danych udostępnianych przez pozostałe moduły w bazie danych
- Implementacja intuicyjnego w obsłudze GUI
- Trwałość zapisywanych danych
  - Na etapie wdrożeniowym systemu, zatrzymanie/zresetowanie instancji kontenera z działającą aplikacją nie powoduje utraty danych

Diagram przypadków użycia



Dodatkowy diagram przypadków użycia



Powiązania z innymi modułami

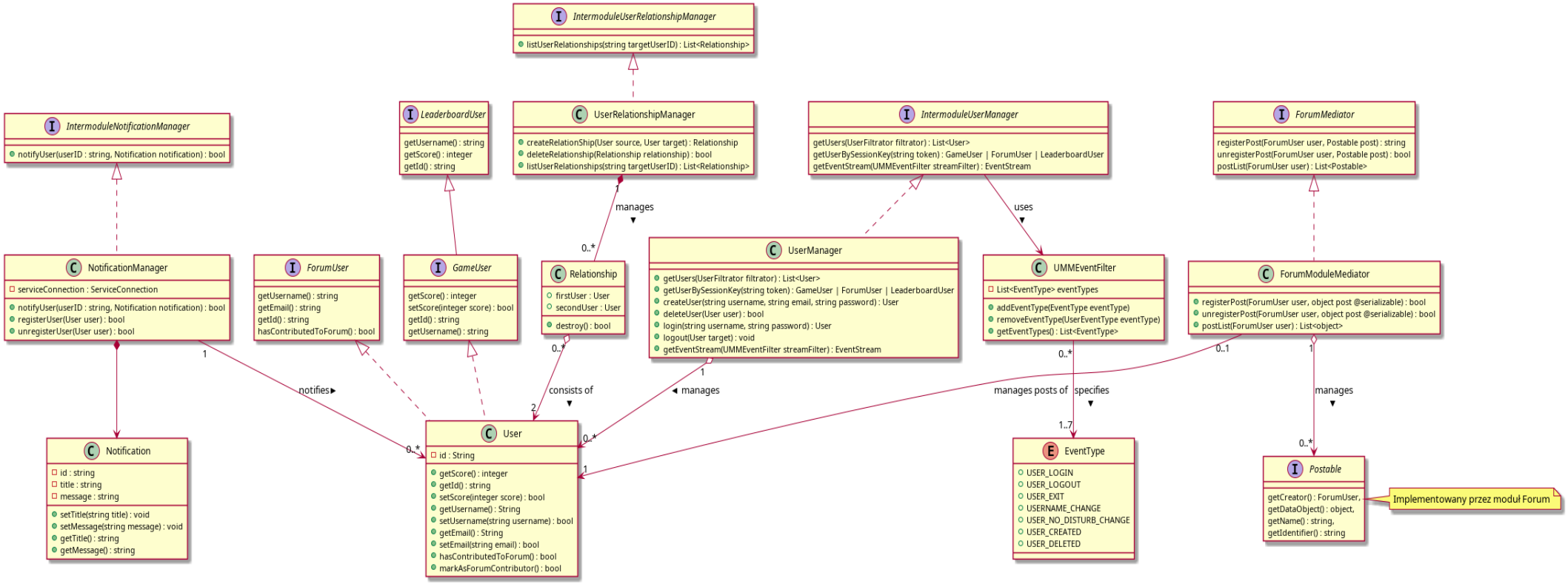
- Powiązanie z modułem „Game Module”
  - Moduł UMM synchronizuje z bazą danych postęp użytkownika w grze obliczany przez moduł GM.
  - Moduł GM może odczytać z modułu UMM dane użytkownika
  - Moduł GM może reagować na zdarzenia otrzymane ze strumienia zdarzeń udostępnianego przez UMM. Chodzi tu o zdarzenia logowania się użytkownika, opuszczenia strony internetowej przez użytkownika, pojawienie się/zniknięcie wpisu użytkownika na forum, opcjonalnie ustawienia przez użytkownika statusu „Nie przeszkadzać”, opcjonalnie zmiany pozycji w tabeli wyników lub opcjonalnie innych.
  - Moduł GM może powiadamiać użytkownika o zdarzeniach dotyczących gry wykorzystując operacje udostępniane przez dedykowany interfejs modułu UMM

- Powiązanie z modułem “Leaderboard Module”
  - Moduł LM może odczytać z modułu UMM dane użytkownika (włącznie z punktacją w grze)
  - Moduł LM może reagować na zdarzenia otrzymane ze strumienia zdarzeń udostępnianego przez UMM. Chodzi tu o zdarzenia utworzenia/usunięcia użytkownika, opcjonalnie logowania się użytkownika, opcjonalnie opuszczenia strony internetowej przez użytkownika, opcjonalnie ustawienia przez użytkownika statusu „Nie przeszkadzać”, opcjonalnie zmiany pozycji w tabeli wyników lub opcjonalnie innych.
  - Moduł LM może powiadamiać użytkownika o zdarzeniach dotyczących tablicy wyników wykorzystując operacje udostępniane przez dedykowany interfejs modułu UMM

Powiązania z innymi modułami c.d.

- Powiązanie z modułem “Forum Module”
  - Moduł UMM synchronizuje z bazą danych informacje dotyczące wpisów użytkowników na forum, generowane przez moduł FM.
  - Moduł FM może odczytać z modułu UMM dane użytkownika
  - Moduł FM może reagować na zdarzenia otrzymane ze strumienia zdarzeń udostępnianego przez UMM. Chodzi tu o zdarzenia logowania się użytkownika, opuszczenia strony internetowej przez użytkownika, utworzenia/usunięcia konta użytkownika, opcjonalnie ustawienia przez użytkownika statusu „Nie przeszkadzać”, opcjonalnie zmiany pozycji w tabeli wyników lub opcjonalnie innych.
  - Moduł FM może powiadamiać użytkownika o zdarzeniach dotyczących forum wykorzystując operacje udostępniane przez dedykowany interfejs modułu UMM

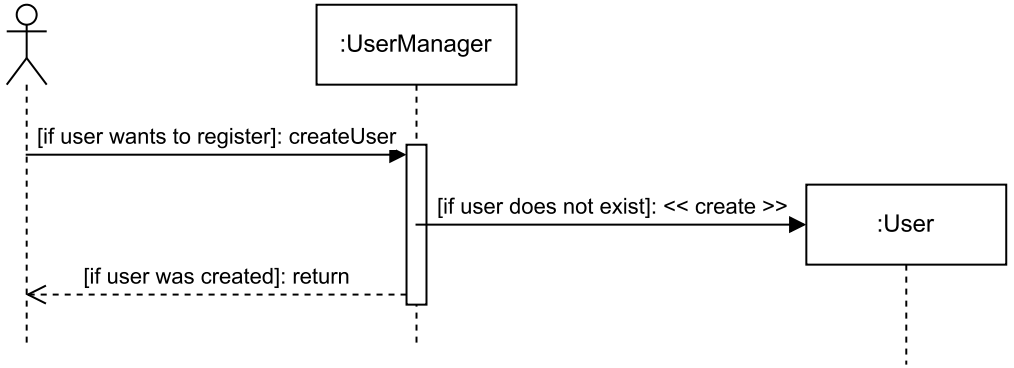
Diagram klas



Scenariusz – Zarejestruj użytkownika w systemie

Poziom ważności	Krytyczny
Typ przypadku użycia	Ogólny, niezbędny
Aktorzy	Użytkownik, Moduł UMM
Krótki opis	Scenariusz przedstawia ciąg zdarzeń wymagany do rejestracji użytkownika w systemie
Warunki końcowe	Istnienie konta użytkownika w systemie
Warunki wstępne	Brak
Główny przepływ zdarzeń	1. Użytkownik rejestruje się 2. Tworzony jest obiekt User 3. Zwracany jest utworzony obiekt
Alternatywne przepływy zdarzeń	1a. Konto istnieje – obiekt nie jest tworzony
Specjalne wymagania	Brak

Diagram sekwencji – Zarejestruj użytkownika w systemie



Scenariusz – Zaloguj użytkownika do systemu

Poziom ważności	Krytyczny
Typ przypadku użycia	Ogólny, niezbędny
Aktorzy	Użytkownik, Moduł UMM
Krótki opis	Scenariusz przedstawia ciąg zdarzeń wymagany do zalogowania się użytkownika do systemu
Warunki końcowe	Utworzenie obiektu użytkownika służącego do manipulacji danymi
Warunki wstępne	Istnienie użytkownika w systemie
Główny przepływ zdarzeń	1. Użytkownik loguje się 2. Tworzony jest obiekt User 3. Zwracany jest utworzony obiekt
Alternatywne przepływy zdarzeń	1a. Konto nie istnieje – obiekt nie jest tworzony 1b. Hasło jest niepoprawne – obiekt nie jest tworzony
Specjalne wymagania	Brak

Diagram sekwencji – Zaloguj użytkownika do systemu

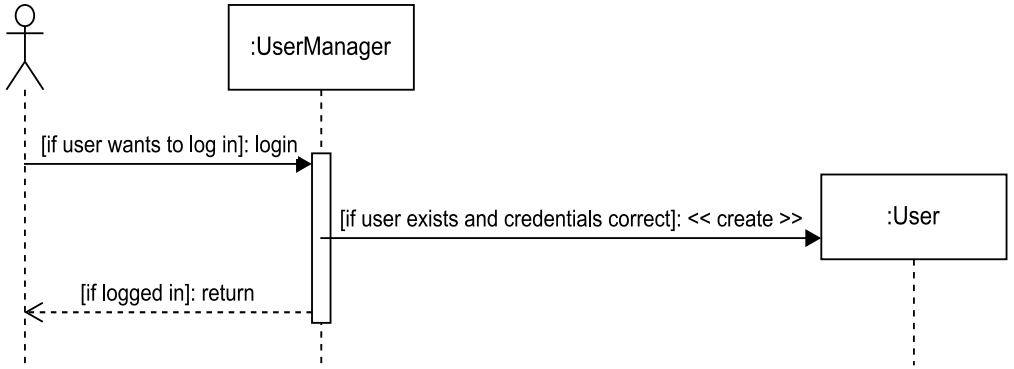


Diagram maszyny stanu – Zaloguj użytkownika do systemu

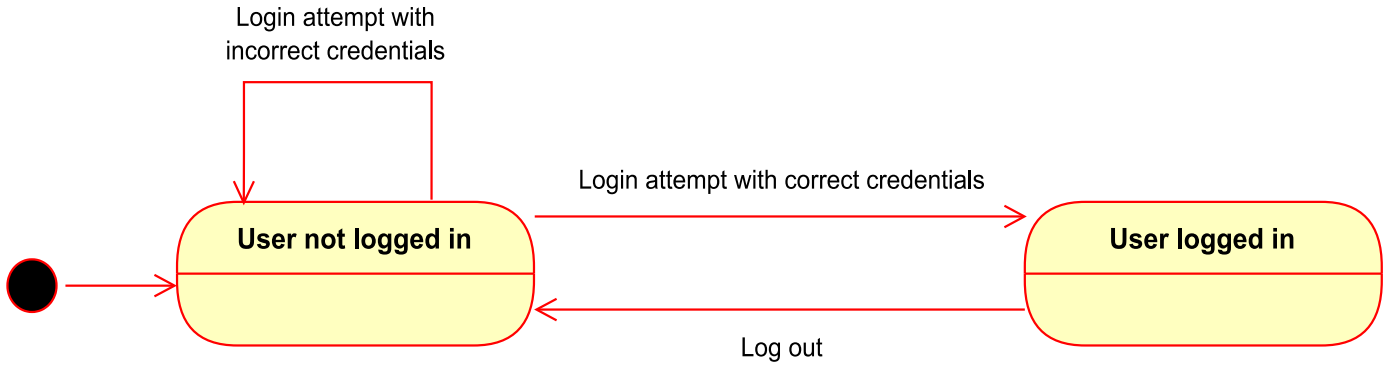
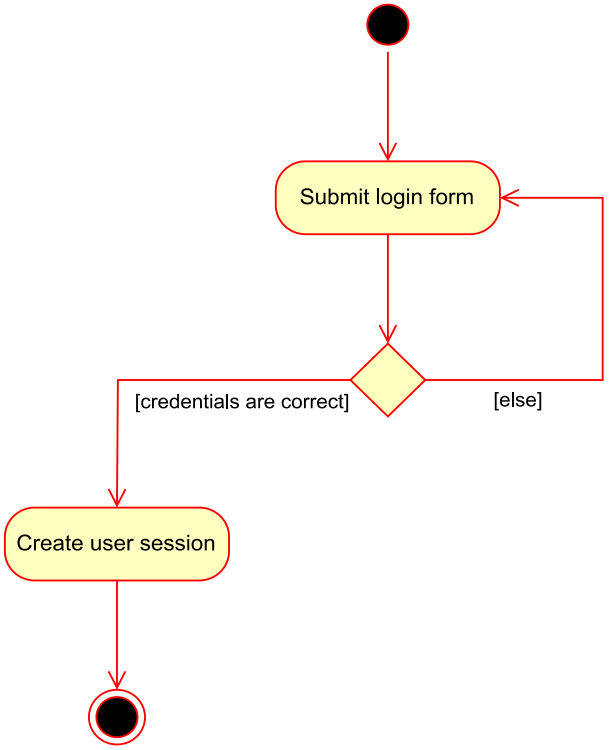


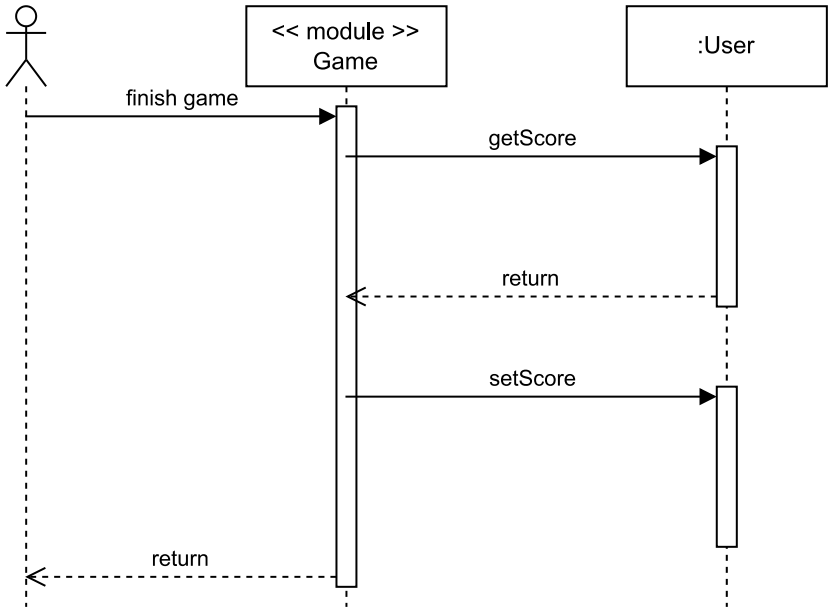
Diagram czynności – Zaloguj użytkownika do systemu



Scenariusz – Zapisz postęp użytkownika w grze

Poziom ważności	Wysoki
Typ przypadku użycia	Ogólny
Aktorzy	Użytkownik, Moduł UMM, Moduł Game
Krótki opis	Scenariusz przedstawia ciąg zdarzeń wymagany do zapisu nowego wyniku w grze
Warunki końcowe	Zapisanie nowego wyniku
Warunki wstępne	Użytkownik jest zalogowany do systemu
Główny przepływ zdarzeń	1. Użytkownik wykonuje dzienną grę 2. Moduł Game pobiera obecny wynik 3. Moduł Game zapisuje nowy wynik
Alternatywne przepływy zdarzeń	Brak
Specjalne wymagania	Moduł Game potrzebuje wiedzy o obecnym wyniku

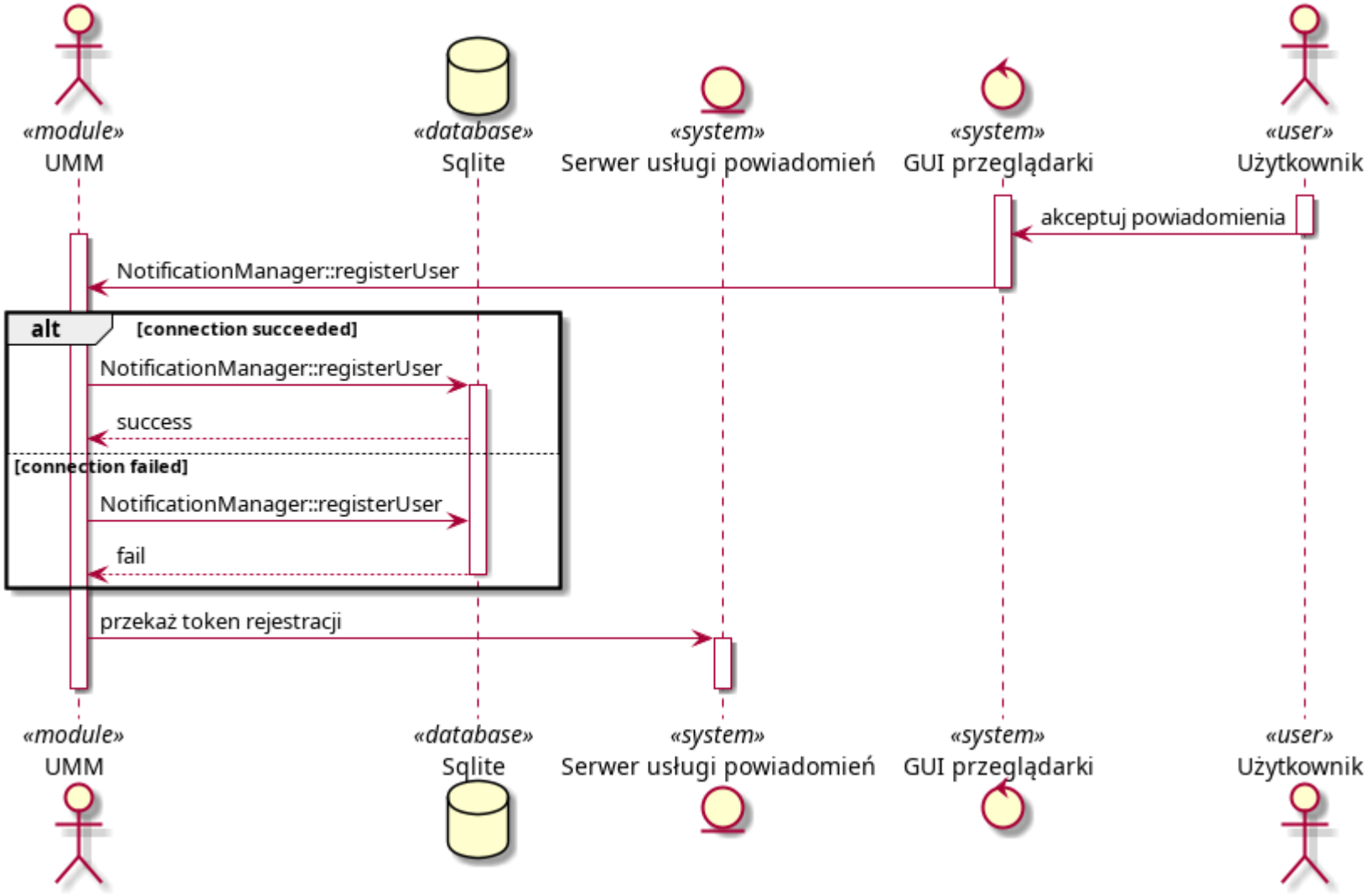
Diagram sekwencji – Zapisz postęp użytkownika w grze



Scenariusz – Subskrybuj powiadomienia push

Poziom ważności	Średni
Typ przypadku użycia	Ogólny
Aktorzy	Użytkownik, Moduł UMM, Serwer usługi powiadomień, przeglądarka
Krótki opis	Rejestracja przeglądarki na urządzeniu użytkownika w celu późniejszego wysyłania powiadomień *push* za pośrednictwem serwera usługi powiadomień (np. Google Cloud Messaging)
Warunki wstępne	Aby rejestracja urządzenia była możliwa, użytkownik musi wyrazić zgodę na otrzymywanie powiadomień
Warunki końcowe	Wysyłane powiadomienień do przeglądarki na urządzenie użytkownika będzie możliwePowiwidoczne dla użytkownika w przeglądarce internetowej. Po kliknięciu powiadomienia, użytkownik zostanie przekierowany do odpowiedniego zasobu o ile aktywność nie wygasła.
Główny przepływ zdarzeń	1. Użytkownik akceptuje powiadomienia w przeglądarce 2. UMM wykonuje operację NotificationManager::RegisterUser 3. UMM przesyła token rejestracji przeglądarki do Serwera usługi powiadomień
Alternatywne przepływy zdarzeń	2a. Użytkownik wykonuje operację NotificationManager::RegisterUser, która zwraca błąd połączenia z bazą danych ;
Specjalne wymagania	Konfiguracja połączenia szyfrowanego z wykorzystaniem kryptografii asymetrycznej

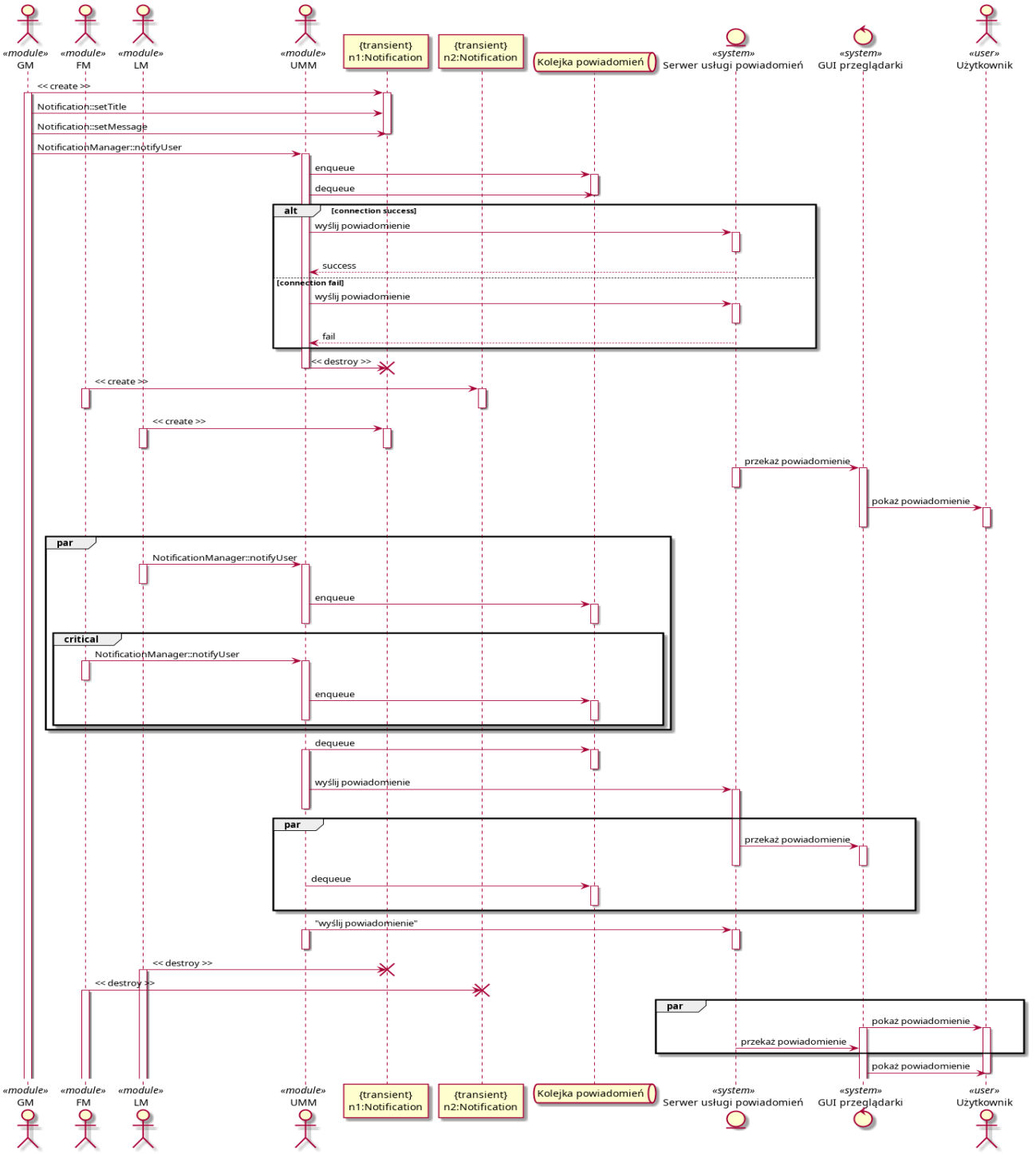
Diagram sekwencji – Subskrybuj powiadomienia push



Scenariusz – wyślij powiadomienie

Poziom ważności	Średni
Typ przypadku użycia	Ogólny
Aktorzy	Użytkownik, Moduł UMM, Moduł Game, Moduł Forum, Moduł Leaderboard, kolejka powiadomień, Serwer usługi powiadomień, przeglądarka
Krótki opis	Wysłanie do przeglądarki na urządzeniu użytkownika powiadomienia *push* za pośrednictwem serwera usługi powiadomień (np. Google Cloud Messaging)
Warunki wstępne	Aby wysłanie powiadomienia było możliwe, użytkownik musi wyrazić zgodę na otrzymywanie powiadomień oraz powinna być wcześniej wykonana subskrybcja powiadomień na poziomie modułu UMM z użyciem operacji NotificationManager::RegisterUser
Warunki końcowe	Powiadomienie będzie widoczne dla użytkownika w przeglądarce internetowej. Po kliknięciu powiadomienia, użytkownik zostanie przekierowany do odpowiedniego zasobu o ile aktywność nie wygasła.
Główny przepływ zdarzeń	1. GM/LM/FM tworzy obiekt klasy Notification 2. GM/LM/FM wykonuje operacje Notification::setTitle oraz Notification::setMessage 3. GM/LM/FM wykonuje operację NotificationManager::notifyUser 4. UMM wysyła parametry powiadomienia gry do Serwera usługi powiadomień, który przekazuje powiadomienie do przeglądarki na urządzeniu klienta
Alternatywne przepływy zdarzeń	3a. NotificationManager::notifyUser zwraca błąd połączenia z serwerem usługi powiadomień
Specjalne wymagania	Konfiguracja połączenia szyfrowanego z wykorzystaniem kryptografii asymetrycznej

Diagram sekwencji – Wyślij powiadomienie



Scenariusz – Rezygnuj z subskrybcji powiadomień push

Poziom ważności	Średni
Typ przypadku użycia	Ogólny
Aktorzy	Użytkownik, Moduł UMM, Serwer usługi powiadomień, przeglądarka
Krótki opis	Anulowanie rejestracji przeglądarki na urządzeniu użytkownika w celu wstrzymania wysyłania powiadomień *push* za pośrednictwem serwera usługi powiadomień
Warunki wstępne	Brak
Warunki końcowe	Wysyłane powiadomienień do przeglądarki na urządzenie użytkownika nie będzie miało miejsca
Główny przepływ zdarzeń	1. Użytkownik wycofuje zgodę na powiedomienia poprzez GUI aplikacji 2. UMM wykonuje operację NotificationManager::unregisterUser
Alternatywne przepływy zdarzeń	1a. Użytkownik wycofuje zgodę na powiadomienia w przeglądarce 2a. Użytkownik wykonuje operację NotificationManager::RegisterUser, która zwraca błąd połączenia z bazą danych
Specjalne wymagania	Konfiguracja połączenia szyfrowanego z wykorzystaniem kryptografii asymetrycznej

Diagram sekwencji – Rezygnuj z subskrybcji powiadomień push

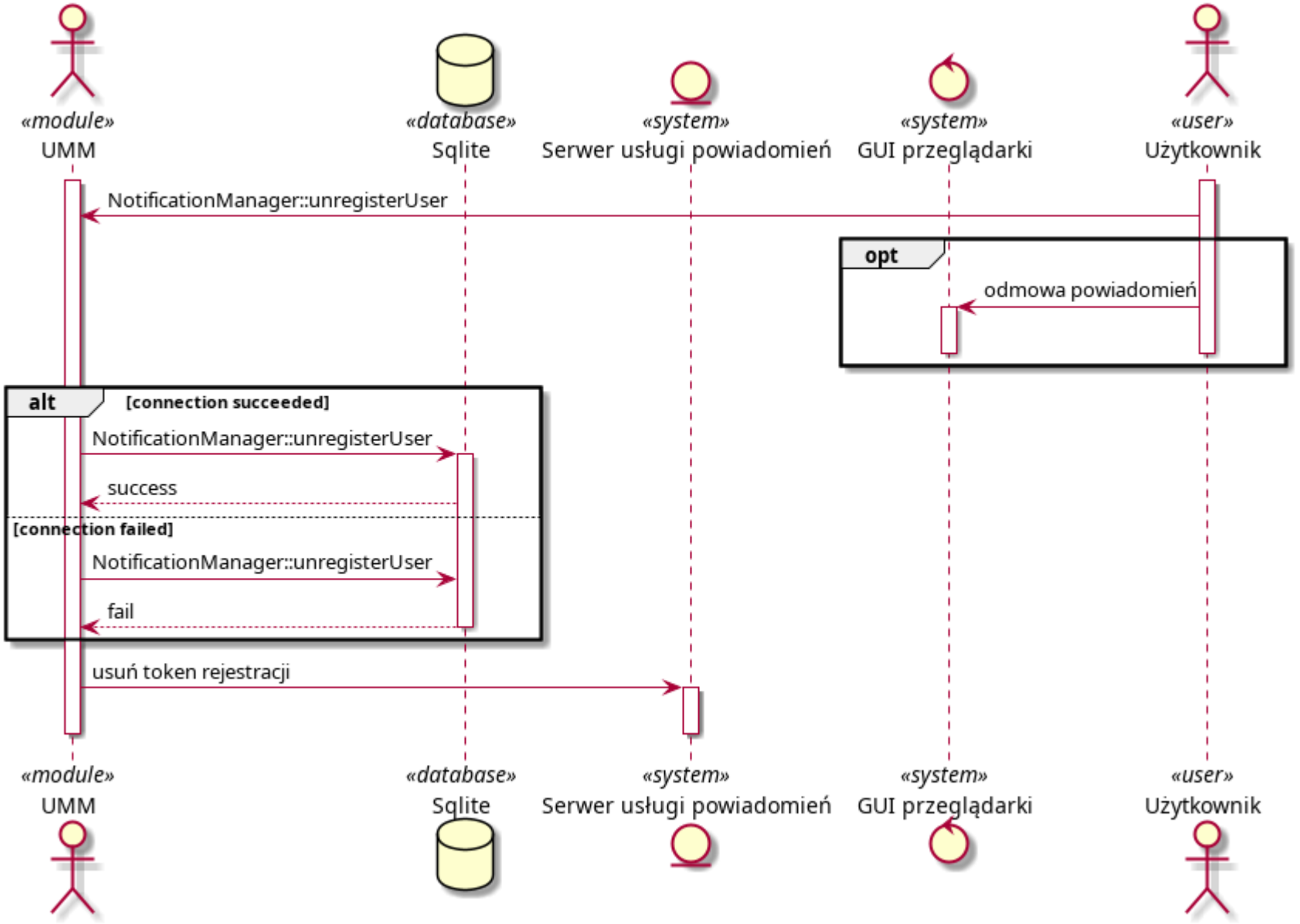
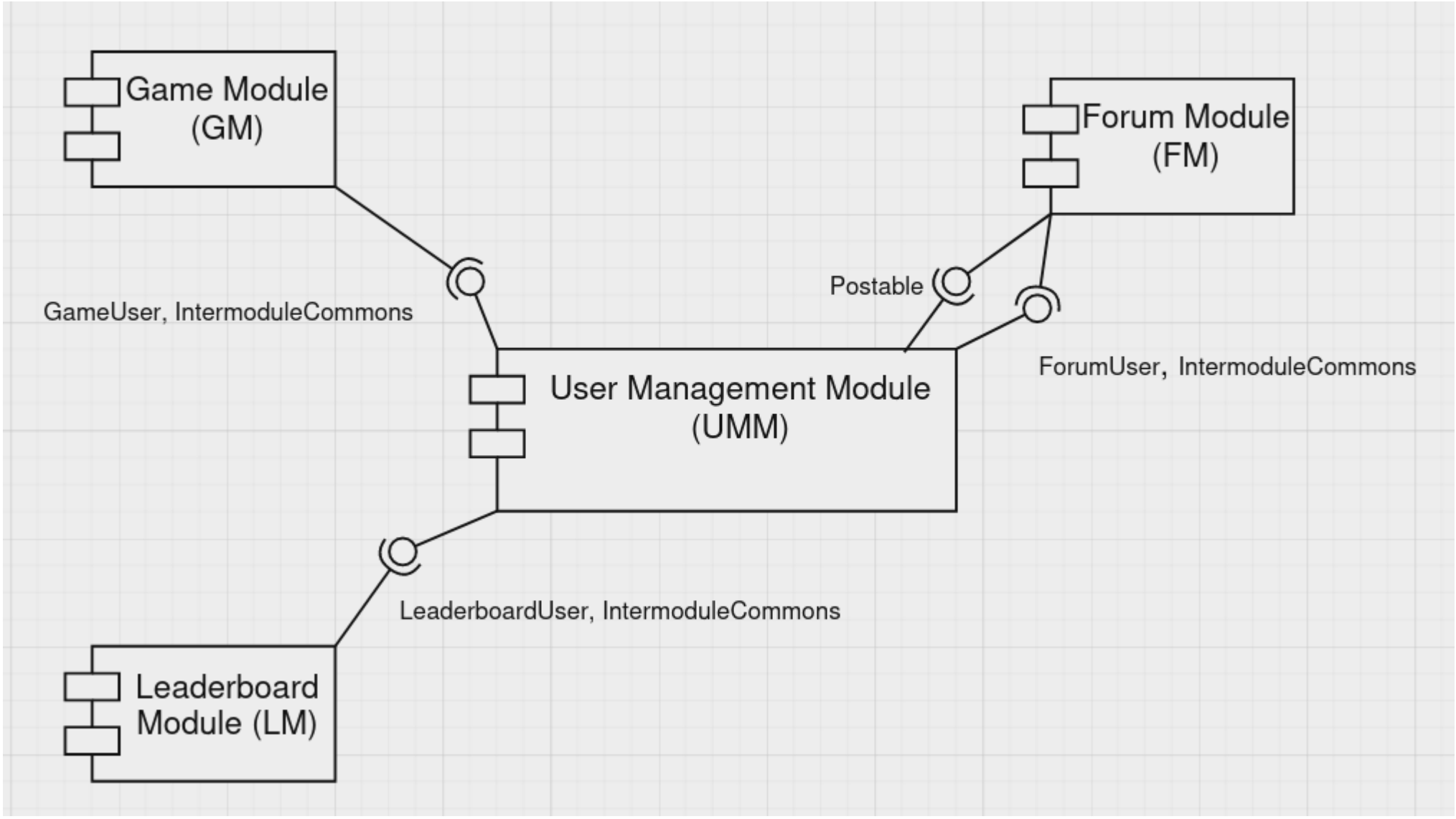


Diagram komponentów



Realizacja założeń i wymagań

Założenia:

- Moduł ten odbiera od pozostałych elementów systemu informacje takie jak dane użytkowników, ich postęp w grze, aktywność na forum
  Dane użytkowników są zbierane za pomocą funkcji User::setUsername, User::setEmail, GameUser::setScore
- Ten moduł umożliwia zewnętrznym w stosunku do niego komponentom utrwalanie danych generowanych podczas użytkowania aplikacji, pośrednicząc w komunikacji z bazą danych
- Jest to jedyny moduł systemu, który wchodzi w bezpośrednią interakcję z bazą danych
  Moduł zawiera w sobie połączenie z bazą danych, którego nie udostępnia innym modułom

Wymagania funkcjonalne użytkownika:

- Użytkownik może zarejestrować się w systemie
  Zrealizowane poprzez funkcję UserManager::createUser
- Użytkownik może zalogować się do do systemu
  Zrealizowane funkcją UserManager::login
- Użytkownik może edytować część danych powiązanych z jego kontem
  Zrealizowane poprzez funkcje User::setUsername, User::setEmail
- Użytkownik może wyświetlić profil innego użytkownika lub swój własny
  Można wybrać dowolnego użytkownika za pomocą funkcji UserManager::getUsers oraz odpowiedniego filtra
- Użytkownik może dodać innego użytkownika do listy kontaktów
  Realizowane przez funkcję UserRelationshipManager::createRelationship
- Użytkownik może usunąć innego użytkownika z listy kontaktów
  Implementowane funkcją UserRelationship::destroy

Realizacja założeń i wymagań c.d.

Wymagania funkcjonalne użytkownika:

- Użytkownik może wylogować się z systemu
  Realizowane funkcją UserManager::logout

Wymagania funkcjonalne międzymodułowe:

- Wprowadzenie prostego interfejsu udostępniającego dane użytkownika dla pozostałych modułów
  Implementowane interfejsem IntermoduleModuleManager oraz interfejsem ModuleUser dla każdego modułu
  Interfejs IntermoduleUserRelationshipManager konsumowany może być konsumowany przez moduł Leaderboard do wyświetlania osiągnięć znajomych
  Interfejs IntermoduleNotificationManager może być użyty przez moduł Forum do wyświetlania nowych postów
- Wprowadzenie interfejsu utrwalania danych dla modułu “Game Module”
- Wprowadzenie interfejsu utrwalania danych dla modułu “Forum Module”
  Implementowane za pomocą zaenkapsulowanego połączenia z bazą danych oraz realizacji pozostałych założeń i wymagań
- (Opcjonalne) Wprowadzenie interfejsu dostępu do strumienia zdarzeń aktualizacji stanu użytkownika
- (Opcjonalne) Wprowadzenie interfejsu udostępniającego możliwość wysyłania powiadomień *push* i/lub wiadomości skrzynki mailowej do użytkowników, którzy wyrazili na to zgodę
  Realizowane poprzez funkcje IntermoduleUserManager::getEventStream oraz IntermoduleNotificationManager::notifyUser

Wymagania niefunkcjonalne nie mogą być przedstawione na diagramie UML – są to szczegóły implementacyjne

Realizacja powiązań z innymi modułami

- Moduł realizuje powiązanie z modułem Game
  Moduł pozwala na pobranie, jak i zapisanie, postępu użytkownika za pośrednictwem funkcji GameUser::getScore oraz GameUser::setScore

- Moduł realizuje powiązanie z modułem Forum
  Pozwala na zapis, usuwanie i odczyt listy postów na forum za pomocą funkcji ForumMediator::postList, ForumMediator::registerPost, ForumMediator::unregisterPost
  Wykorzystaje implementację interfejsu wymagającego Postable dostarczaną przez moduł Forum w celu utrwalania danych

- Moduł realizuje powiązanie z modułem Leaderboard
  Pozwala na odczyt postępu wszystkich użytkowników poprzez funkcje UserManager::getUsers i LeaderboardUser::getScore

- Dodatkowo moduł udostępnia do dyspozycji każdego z pozostałych modułów trzy interfejsy IntermoduleCommons:

  - IntermoduleUserManager
    Pobieranie i filtrowanie listy użytkowników
    Odczyt strumienia zdarzeń związanych z użytkownikami
  - IntermoduleNotificationManager
    Wysyłanie powiadomień do użytkowników
  - IntermoduleUserRelationshipManager
    Eksploracja powiązań między użytkownikami poprzez wyświetlanie listy użytkowników dodanych do kontaktów przez gracza.