

1. Sprawozdanie SSBD	2
1.1 Strona tytułowa	3
1.1.1 Konta aplikacji	5
1.1.2 Opis funkcjonalności podstawowej i dodatkowej	6
1.2 Sprawozdanie wstępne (15 pkt)	7
1.2.1 1 Dziedzina wykorzystania 1 pkt	8
1.2.2 2 Moduły funkcjonalne, aktorzy, przypadki użycia 3 pkt	11
1.2.3 3 Struktury relacyjnej bazy danych 2 pkt	26
1.2.4 4 Użytkownicy bazodanowi 2 pkt	38
1.2.5 5 Identyfikacja obiektów encji 2 pkt	39
1.2.6 6 Diagram klas encji 1 pkt	43
1.2.7 7 Diagramy sekwencji oraz identyfikacja transakcji bazodanowych 3 pkt	44
1.2.8 8 Konfiguracja uwierzytelniania w aplikacji 1 pkt	56
1.3 Sprawozdanie szczegółowe (25 pkt)	57
1.3.1 9 Diagram UML klas komponentów EJB/CDI 4 pkt	58
1.3.2 10 Model bezpieczeństwa komponentów EJB/CDI 5 pkt	60
1.3.3 11 Identyfikacja transakcji aplikacyjnych 6 pkt	69
1.3.4 12 Zgłasiane wyjątki 4 pkt	85
1.3.5 13 Interfejs użytkownika 3 pkt	87
1.3.6 14 Zabezpieczenia interfejsu użytkownika 3 pkt	99
1.3.7 15 Zmiany - projekt szczegółowy	103
1.4 Sprawozdanie końcowe (10 pkt)	104
1.4.1 16 Zabezpieczenie transmisji HTTP 1 pkt	105
1.4.2 17 Weryfikacja poprawności danych 2 pkt	107
1.4.3 18 Obsługa błędów 2 pkt	115
1.4.4 19 Wersje językowe interfejsu użytkownika 1 pkt	119
1.4.5 20 Wykaz akcji dostępnych z interfejsu użytkownika 4 pkt	133
1.4.6 21 Zmiany - projekt końcowy	177

Sprawozdanie SSBD

SPRAWOZDANIE

Z REALIZACJI LABORATORIUM PRZEDMIOTU

SIECIOWE SYSTEMY BAZ DANYCH

EDYCJA 2023

Strona tytułowa

Projekt SSBDO1: "Apteka internetowa"

Członkowie grupy projektowej i funkcje projektowe:

- Jakub Glegoła 236535, Funkcje pełnione w projekcie: **lider**, kontrola zgodności. [MOA.1](#), [MOA.2](#), [MOA.19](#), [MOA.21](#), [MOA.23](#)
- Antoni Jończyk 236551, Funkcja pełniona w projekcie: dokumentacja. [MOA.15](#), [MOA.18](#), [MOA.20](#), [MOA.22](#)
- Wiktor Malavasi 236594, Funkcja pełniona w projekcie: baza danych. [MOA.3](#), [MOA.4](#), [MOA.5](#), [MOA.14](#)
- Michał Pianka 229979, Funkcja pełniona w projekcie: kontrola jakości. [MOA.7](#), [MOA.8](#), [MOA.9](#), [MOA.10](#)
- Jan Rubacha 236641, Funkcje pełnione w projekcie: wdrożenie, kontrola jakości. [MOA.11](#), [MOA.12](#), [MOA.13](#)
- Piotr Strachota 236660, Funkcja pełniona w projekcie: architektura. [MOA.6](#), [MOA.16](#), [MOA.17](#)

Zasoby:

- Zarządzanie projektem, system rejestracji zadań JIRA: <https://atlas.it.p.lodz.pl/jira>
- Repozytorium projektu Bitbucket: <https://atlas.it.p.lodz.pl/bitbucket>
- System gromadzenia i wymiany informacji Confluence: <https://atlas.it.p.lodz.pl/confluence>
- Witryna aplikacji : <https://team-1.proj-sum.it.p.lodz.pl/>
- Zarządzanie serwerem aplikacyjnym: <https://10.31.201.3:4848>

Macierz decyzyjna przypadków użycia

P.U.	Opis	Gość	Administrator	Farmaceuta	Pacjent	System
MOK.1	Zarejestruj	V				
MOK.2	Utwórz konto		V			
MOK.3	Zablokuj konto		V			
MOK.4	Odblokuj konto		V			
MOK.5	Dolącz poziom dostępu do konta		V			
MOK.6	Odlącz poziom dostępu od konta		V			
MOK.7	Zmień własne hasło		V	V	V	
MOK.8	Zmień hasło innego użytkownika		V			
MOK.9	Edytuj dane własnego konta		V	V	V	
MOK.10	Edytuj dane konta innego użytkownika		V			
MOK.11	Wyloguj		V	V	V	
MOK.12	Usuń konto w przypadku braku aktywacji					V
MOK.13	Wyświetl informacje o koncie		V	V	V	
MOK.14	Wyświetl listę kont		V			
MOK.15	Wyświetl informacje o koncie innego użytkownika		V			
MOK.16	Zresetuj hasło	V				
MOK.17	Zaloguj się na konto	V				
MOA.1	Wyświetl listę leków	V		V	V	
MOA.2	Wyświetl stronę ze szczegółami leku			V	V	
MOA.3	Dodaj lek do koszyka				V	
MOA.4	Wyświetl koszyk				V	
MOA.5	Zmień liczebność leku w koszyku				V	
MOA.6	Usuń lek z koszyka				V	
MOA.7	Złoż zamówienie				V	
MOA.8	Wycofaj złożone zamówienie				V	
MOA.9	Wyświetl kolejkę zamówień oczekujących			V		
MOA.10	Usuń zamówienia z kolejki zamówień oczekujących			V		

MOA.11	Wprowadź zawartość dostawy			V		
MOA.12	Wyświetl listę zamówień oczekujących na potwierdzenie			V		
MOA.13	Potwierdź zamówienie			V		
MOA.14	Odwolaj zamówienie			V		
MOA.15	Pokaż statystyki zakupów			V		
MOA.16	Cyklicznie sprawdź kolejkę i aktualizuj ją				V	
MOA.17	Wyświetl listę złożonych przez siebie zamówień zamówień				V	
MOA.18	Wyświetl listę wszystkich zamówień			V		
MOA.19	Dodaj lek do oferty			V		
MOA.20	Edytuj dane leku			V		
MOA.21	Dodaj kategorię			V		
MOA.22	Wyświetl kategorie			V		
MOA.23	Edytuj kategorię			V		

Spis rozdziałów sprawozdania

Sprawozdanie wstępne

- [1 Dziedzina wykorzystania](#)
- [2 Moduły funkcyjne, aktorzy, przypadki użycia](#)
- [3 Struktury relacyjnej bazy danych](#)
- [4 Użytkownicy bazodanowi](#)
- [5 Identyfikacja obiektów encji](#)
- [6 Diagram klas encji](#)
- [7 Diagramy sekwencji oraz identyfikacja transakcji bazodanowych](#)
- [8 Konfiguracja uwierzytelniania w aplikacji](#)

Sprawozdanie szczegółowe

- [9 Diagram UML klas komponentów EJB](#)
- [10 Schemat bezpieczeństwa komponentów EJB](#)
- [11 Identyfikacja transakcji aplikacyjnych](#)
- [12 Zgłoszone wyjątki](#)
- [13 Interfejs użytkownika](#)
- [14 Zabezpieczenia interfejsu użytkownika](#)
- [15 Zmiany - projekt szczegółowy](#)
- [Sprawozdanie końcowe](#)
- [16 Zabezpieczenie transmisji HTTP](#)
- [17 Weryfikacja poprawności danych](#)
- [18 Obsługa błędów](#)
- [19 Wersje językowe interfejsu użytkownika](#)
- [20 Opis przypadków użycia MVCv2](#)
- [21 Zmiany - projekt końcowy](#)

...

Konta aplikacji

Administrator

Login: admin123

Hasło: P@ssw0rd

Poziom dostępu: Admin

Farmaceuta

Login: chemist123

Hasło: P4\$\$w0Rd

Poziom dostępu: Chemist

Opis funkcjonalnosci podstawowej i dodatkowej

Sprawozdanie wstępne (15 pkt)

Sprawozdanie wstępne

1 Dziedzina wykorzystania 1 pkt

Przeznaczenie aplikacji

- Aplikacja ma za zadanie zarządzanie apteką internetową, klienci zakupić mogą produkty medyczne, farmaceuci obsługują dostawy oraz zatwierdzają zamówienia wymagające recepty. Klienci apteki mogą zakładać konta oraz zarządzać nimi. Farmaceuta, by uzyskać konto musi mieć je utworzone przez administratora, następnie może on je edytować. Klient może przeglądać produkty, dodawać je do koszyka oraz zakupić je, w razie wymaganej recepty składana jest ona razem z zamówieniem. Farmaceuta może dodawać nowe produkty, aktualizować ich stan, zatwierdzać zamówienia wymagające recepty, dodawać kategorie. Aplikacja ma za zadanie umożliwić klientom skorzystanie z usług apteki bez wychodzenia z domu.
- Konkurencja zachodzi dla następujących zasobów: leków (Medication), zamówień (Order) oraz kont (Account).
- Aplikacja zapewnia możliwość szybkiego i efektywnego składania i modyfikacji zamówień przez wielu różnych użytkowników przy zachowaniu spójności danych. Aby to uzyskać, musi zapewnić obsługę transakcji wspólnie przejętych. Są to cechy charakterystyczne modelu OLTP.
- Aplikacja jest bezstanowa, komunikacja pomiędzy warstwami prezentacji i logiki biznesowej odbywa się poprzez REST API w formacie JSON.
- Stos technologiczny dla warstwy logiki biznesowej:

Nazwa	Wersja
Java	17
EJB	3.2
Lombok	1.18.26
Hibernate	6.2.0.CR4
Payara	6.2023.2
Postgresql	15.3
Jakarta	9.1.0
JPA	3.1.0
java JWT	0.11.5
Jackson Databind	2.14.2
Jersey	3.1.1

Słownik nazw obiektów w modelu danych

nazwa	opis	przechowywane dane	zależności
AccessLevel	obiekt bazowy obiektów przechowujących dane do	rola (Role)	jest w relacji wiele-jeden z obiektem Account
AdminData	reprezentuje poziom dostępu administratora		powiązany z Account ponieważ dziedziczy z AccessLevel
ChemistData	reprezentuje poziom dostępu farmaceuty	numer licencji	powiązany z Account ponieważ dziedziczy z AccessLevel
PatientData	reprezentuje poziom dostępu pacjenta	imię, nazwisko, pesel, nr. telefonu, PESEL i NIP	powiązany z Account ponieważ dziedziczy z AccessLevel
Account	reprezentuje konto użytkownika	<ul style="list-style-type: none">• login• hash hasła• czy konto jest aktywne• czy konto jest zarejestrowane	obiekt jest w relacji jeden-wiele z obiektami AccessLevel, przy czym może zawierać maksymalnie jedną instancję z każdej podklasy
Medication	reprezentuje lek	<ul style="list-style-type: none">• nazwa• kategoria (Category)• cena• ilość na stanie	jest w relacji wiele-jeden z Category, i jeden-wiele z OrderMedication oraz OrderShipment

Order	reprezentuje zamówienie	<ul style="list-style-type: none"> wiele leków wraz z ich ilościami (poprzez ShipmentMedication) datę zamówienia odnośnik do pacjenta (klasa PatientData) odnośnik do farmaceuty (klasa ChemistData) informację o tym czy zamówienie znajduje się w kolejce. 	jest w relacji jeden-wiele z prescription i OrderMedication, oraz wiele-wiele z PatientData i ChemistData
OrderMedication	reprezentuje pozycję w zamówieniu	lek (Medication) oraz jego ilość	jest w relacji wiele-jeden z Order oraz z Medication
Prescription	reprezentuje receptę	nr. recepty	jest w relacji wiele-jeden z Order
Shipment	reprezentuje dostawę do apteki	datę dostawy oraz wiele leków wraz z ilościami (poprzez ShipmentMedication)	jest w relacji jeden-wiele z ShipmentMedication
ShipmentMedication	reprezentuje pozycję w dostawie	lek (Medication) oraz jego ilość	jest w relacji wiele-jeden z Shipment oraz z Medication
Category	reprezentuje kategorię leku	angielska i polska nazwa oraz informacja o tym czy leki z danej kategorii są na receptę	jest w relacji jeden-wiele z Medication

Słownik pozostałych pojęć i obiektów

nazwa	opis

2 Moduły funkcjonalne, aktorzy, przypadki użycia 3 pkt

Lista modułów funkcjonalnych

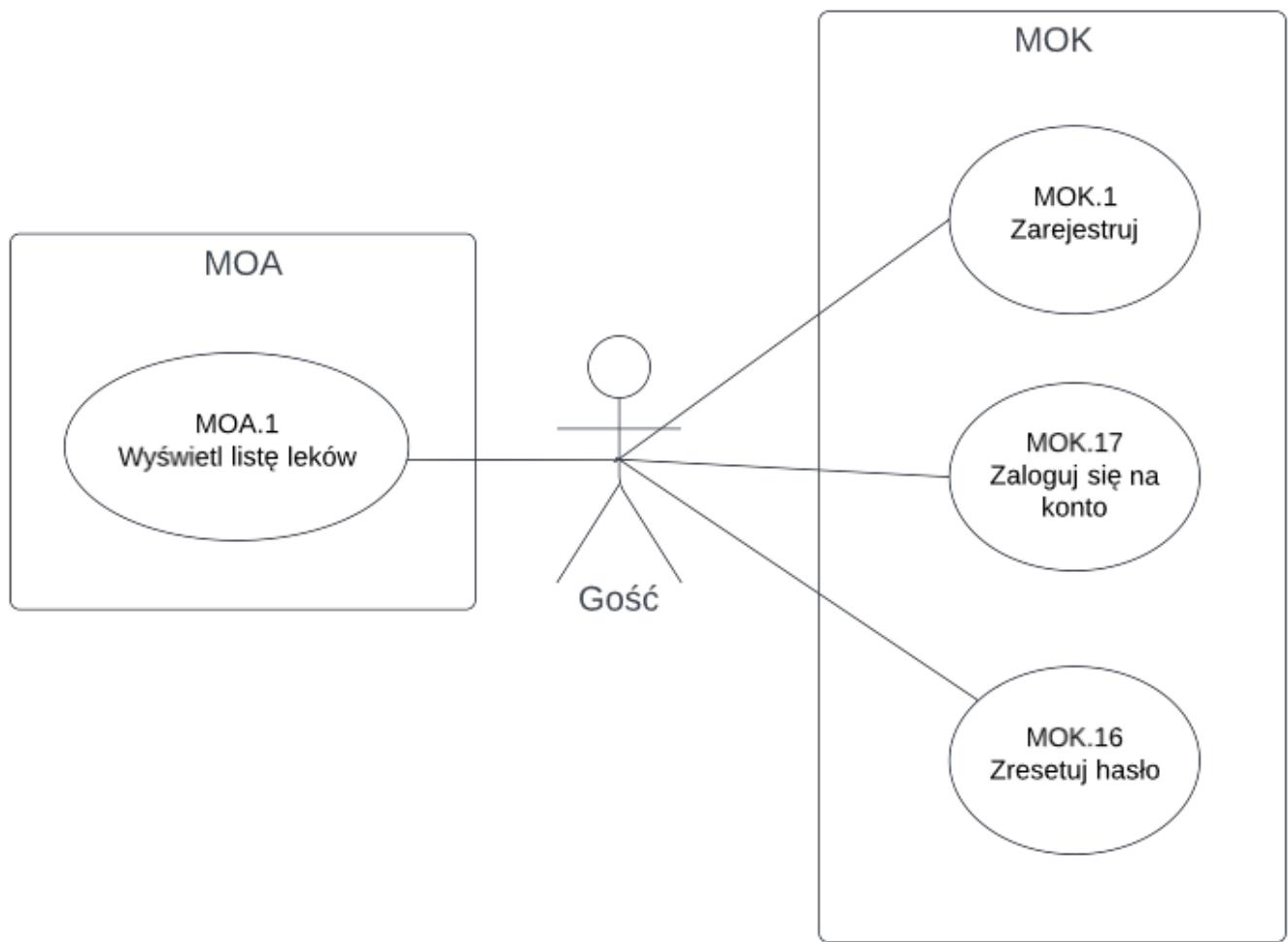
- MOK (Moduł Obsługi Klienta)
- MOA (Moduł Obsługi Apteki)

Lista poziomów dostępu (aktorów)

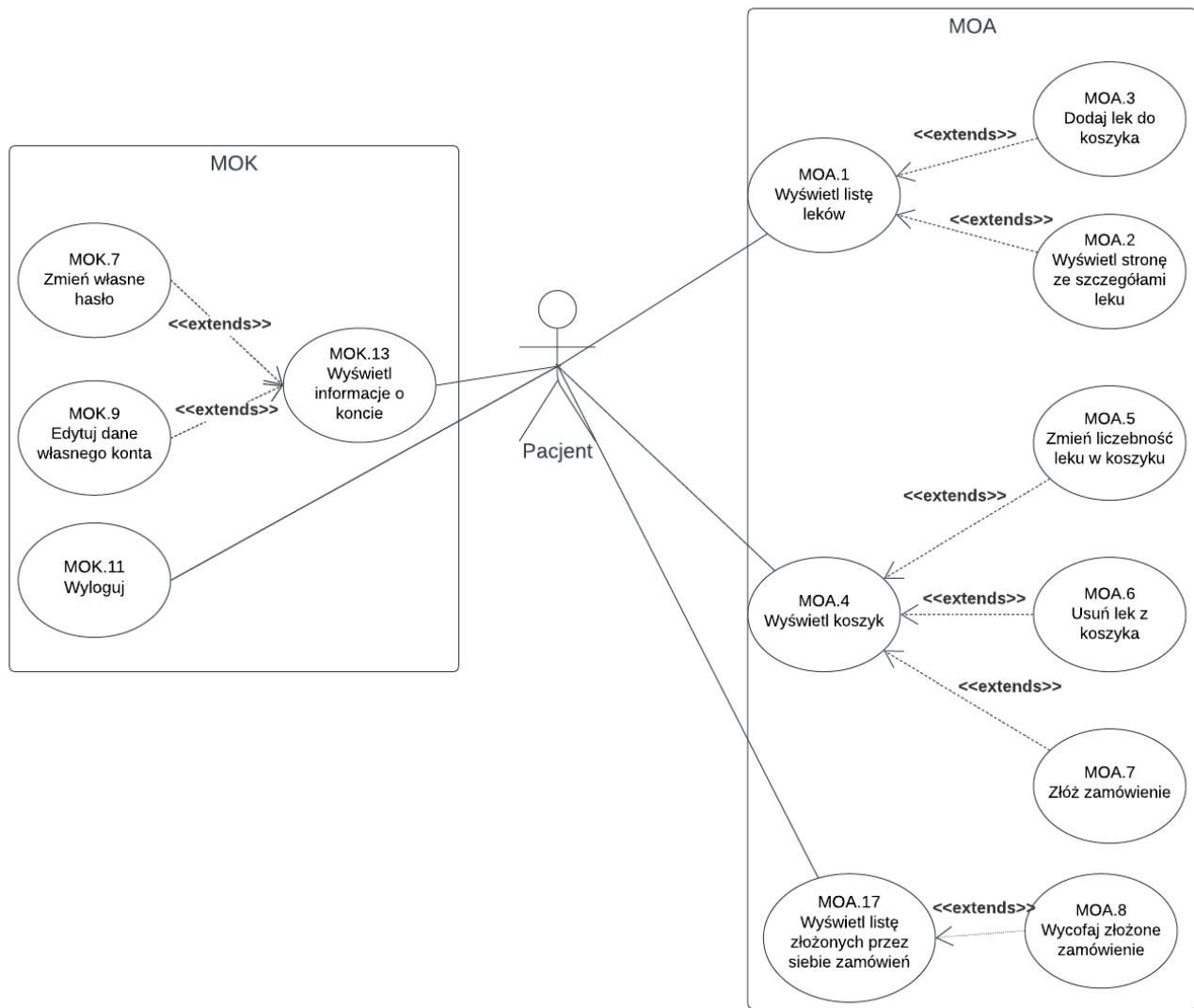
Nazwa poziomu dostępu	Charakteryzacja roli poziomu dostępu w modelu biznesowym aplikacji	Sekwencja czynności prowadzących do utworzenia funkcjonującego konta o danym poziomie dostępu	Wykluczanie
Gość	Wszyscy użytkownicy, którzy mogą wejść na stronę internetową	-	Wyklucza powiązanie z innymi poziomami dostępu
Pacjent	Zwykły użytkownik korzystający z usług apteki internetowej, może zamawiać leki za pośrednictwem strony internetowej	Pacjent samodzielnie rejestruje się podając wymagane dane. Po poprawnej rejestracji system wysyła email z prośbą o potwierdzenie, po którego potwierdzeniu, użytkownik uzyskuje dostęp do konta	Wyklucza powiązanie z poziomem dostępu Farmaceuta
Administrator	Poziom dostępu odpowiedzialny za obsługę kont użytkowników	Konto tworzone przez administratora, bądź tworzony z poziomu bazy danych	Brak
Farmaceuta	Zajmuje się zarządzaniem zamówieniami w aptece internetowej	Konto tworzone przez administratora	Wyklucza powiązanie z poziomem dostępu Pacjent
System	Działania automatyczne aplikacji	Brak	Wyklucza powiązanie z resztą poziomów dostępu

Diagramy przypadków użycia

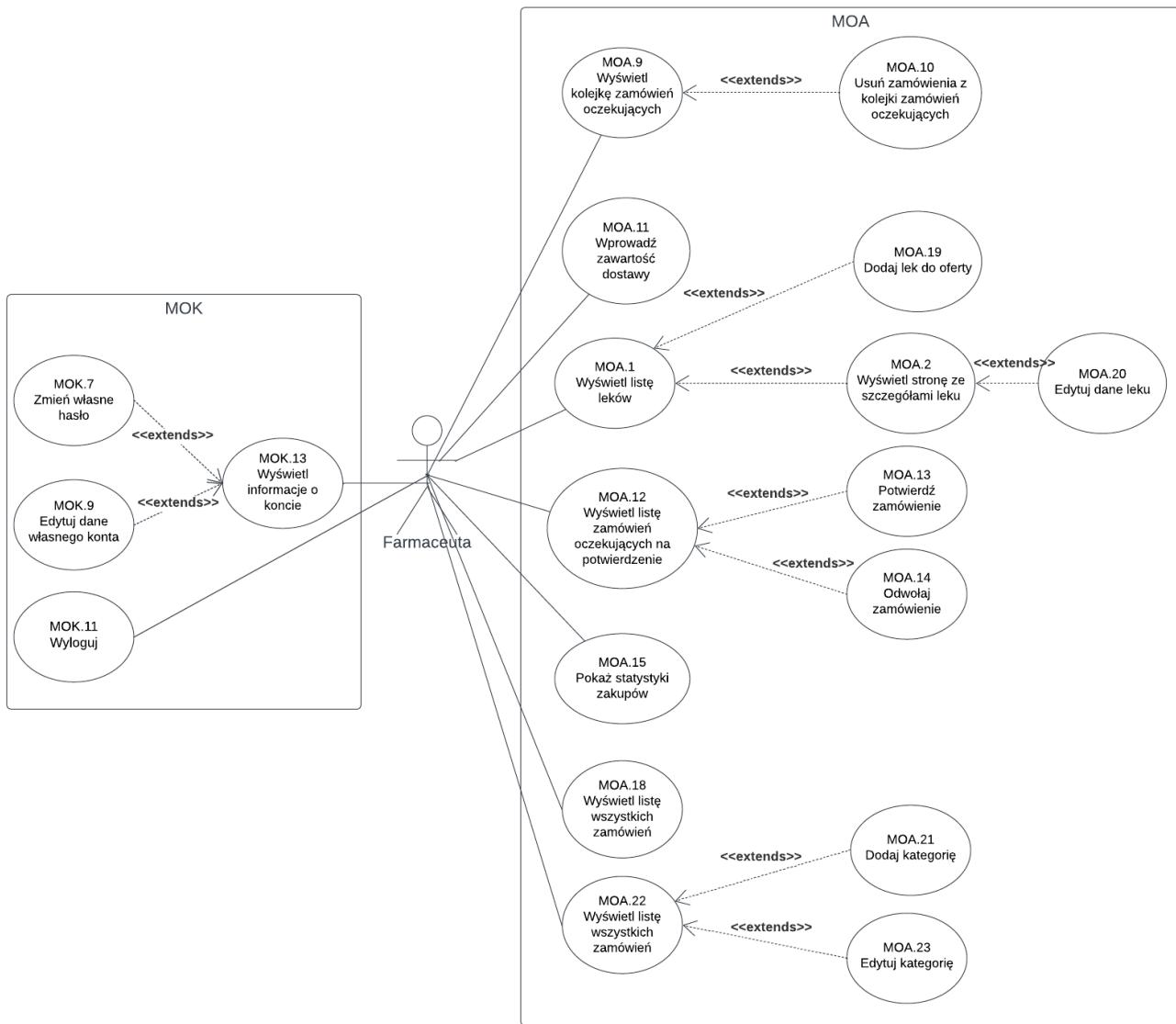
- Forma: odrębne diagramy użycia dla wszystkich poziomów dostępu (aktorów) aplikacji, włącznie z użytkownikiem anonimowym tzw. Gość (nie uwierzytelnionym) oraz System, który nigdy nie może być przypisany do kont użytkowników.
- Na każdym diagramie należy grupować przypadki użycia należące do jednego modułu funkcjonalnego, dla każdego przypadku użycia należy podać akronim modułu i unikalny numer względem tego modułu.
- Diagramy muszą być niezależne. Jeżeli przypadek użycia jest dostępny dla wielu poziomów dostępu, powinien znaleźć się na wszystkich stosownych diagramach.
- Jeżeli dla poszczególnych przypadków użycia występuje relacja rozszerzania, uszczegóławiania to powinna ona być uwzględniona na diagramie.
- Dla każdego przypadku użycia oprócz scenariusza głównego należy wymienić i krótko opisać wszystkie zidentyfikowane scenariusze błędów, a dla każdego z nich wyraźnie należy zaznaczyć przyczynę i lokalizację wystąpienia błędu lub problemu z przetwarzaniem.



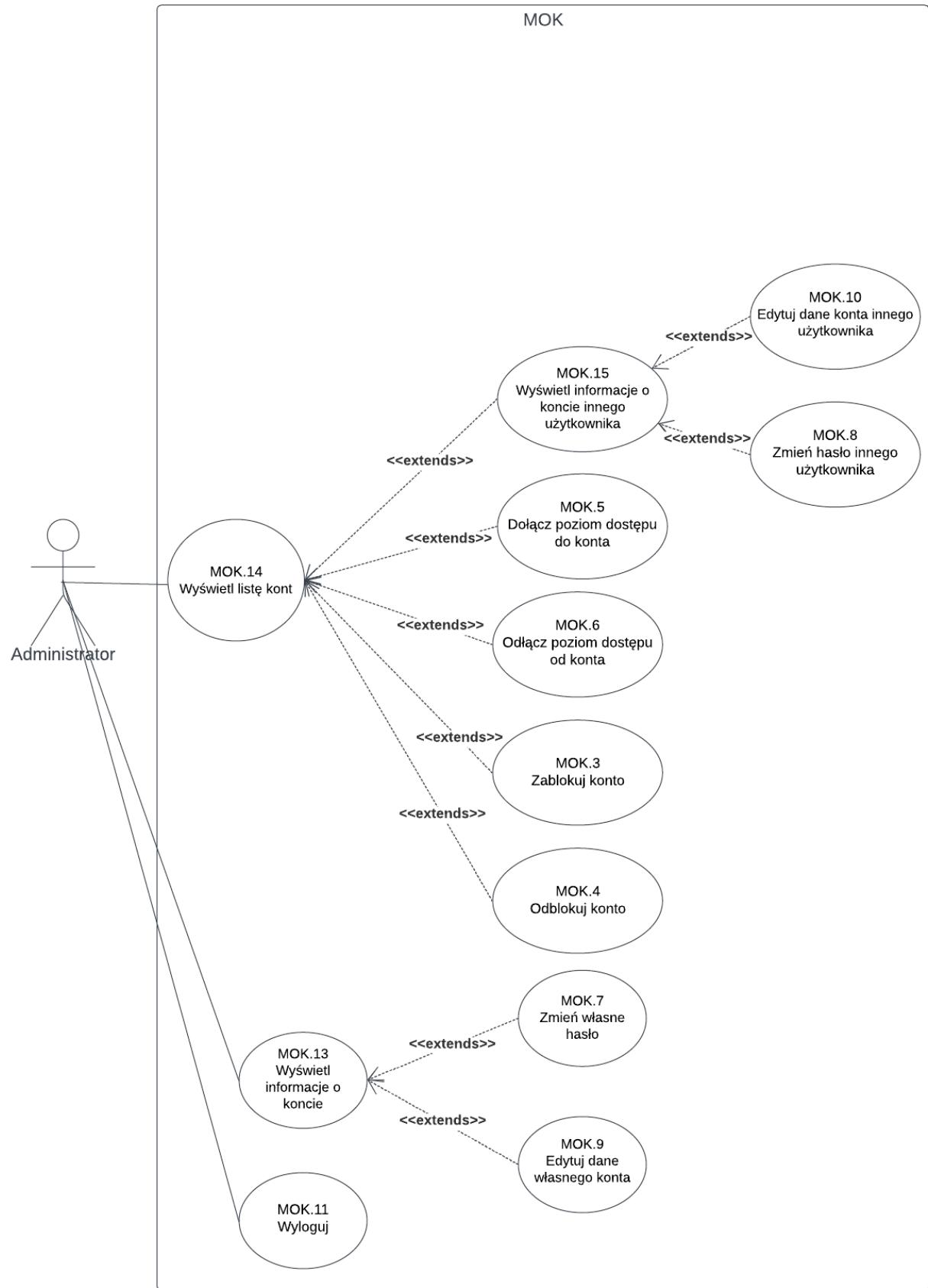
1.2.1. Diagram przypadków użycia dla aktora Gość



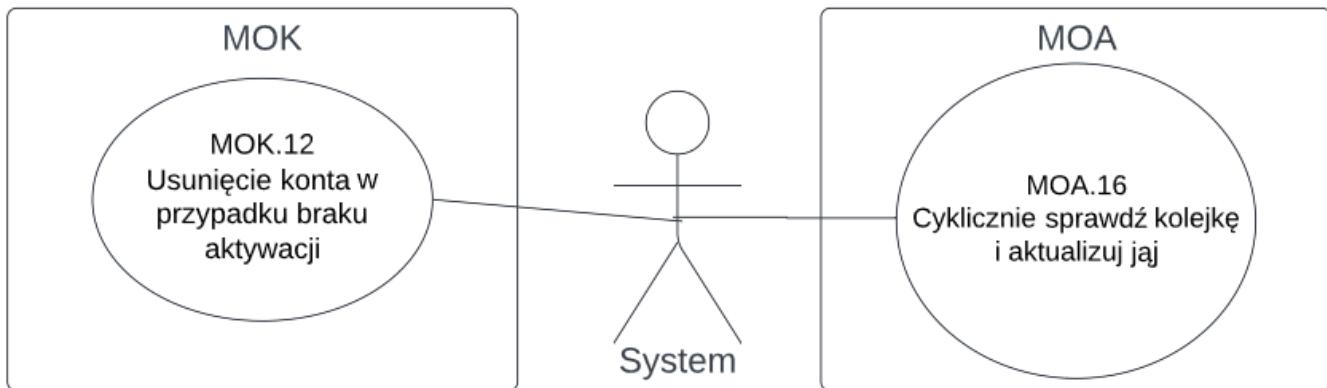
1.2.2. Diagram przypadków użycia dla aktora Pacjent



1.2.3. Diagram przypadków użycia dla aktora Farmaceuta



1.2.4. Diagram przypadków użycia dla aktora Administrator



1.2.4. Diagram przypadków użycia dla aktora System

Nazwa przypadku użycia	MOK.1 Zarejestruj
Aktorzy	Gość
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję zarejestrowania się w systemie 2. Aplikacja wyświetla formularz rejestracji 3. Użytkownik podaje niezbędne do rejestracji dane i zatwierdza formularz 4. Aplikacja potwierdza zgodność podanych danych 5. Aplikacja dodaje konto nowego użytkownika 6. Aplikacja wysyła mail aktywacji na skrzynkę pocztową użytkownika 7. Aplikacja wyświetla komunikat o poprawnym otworzeniu konta 8. Użytkownik aktywuje konto poprzez link z maila
Rozszerzenia	4a. Podane dane są niepoprawne i użytkownik musi ponownie podać dane

Nazwa przypadku użycia	MOK.2 Utwórz konto
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję utwórz konto 2. Aplikacja wyświetla formularz rejestracji 3. Użytkownik wybiera poziom dostępu i podaje niezbędne do rejestracji dane, oraz zatwierdza formularz 4. Aplikacja potwierdza zgodność podanych danych 5. Aplikacja dodaje nowe konto
Rozszerzenia	4a. Podane dane są niepoprawne i użytkownik musi ponownie podać dane

Nazwa przypadku użycia	MOK.3 Zablokuj konto
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wyszukuje użytkownika na liście i na danej pozycji wybiera opcję zablokowania konta 2. Aplikacja sprawdza czy dany użytkownik nie jest zablokowany 3. Aplikacja blokuje konto 4. Użytkownik otrzymuje powiadomienie o poprawnie wykonanej czynności
Rozszerzenia	2a. Konto zostało już zablokowane - Użytkownik otrzymuje powiadomienie o braku możliwości zablokowania konta

Nazwa przypadku użycia	MOK.4 Odblokuj konto
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wyszukuje użytkownika na liście i na danej pozycji wybiera opcję odblokowanie konta 2. Aplikacja sprawdza czy dany użytkownik nie jest odblokowany 3. Aplikacja odblokowuje konto 4. Użytkownik otrzymuje powiadomienie o poprawnie wykonanej czynności
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.5 Dołącz poziom dostępu do konta
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wyszukuje użytkownika na liście i z menu wybiera opcję przydzielania poziomu dostępu 2. Aplikacja pobiera listę poziomów dostępu dla użytkownika 3. Aplikacja wyświetla listę możliwych poziomów dostępu do wyboru 4. Użytkownik wybiera rolę, którą chce dodać z listy, uzupełnia dane roli i zatwierduje formularz 5. Aplikacja wyświetla komunikat aby potwierdzić wykonanie czynności 6. Użytkownik potwierdza dołączenie poziomu dostępu 7. Aplikacja przydziela poziom dostępu do konta 8. Aplikacja wyświetla komunikat o poprawnym przypisaniu dostępu
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.6 Odłącz poziom dostępu od konta
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wyszukuje użytkownika na liście i z menu wybiera opcję przydzielania poziomu dostępu 2. Aplikacja pobiera listę poziomów dostępu dla użytkownika 3. Aplikacja wyświetla listę możliwych poziomów dostępu do wyboru 4. Użytkownik wybiera rolę, którą chce odłączyć z listy i zatwierduje swój wybór 5. Aplikacja wyświetla komunikat aby potwierdzić wykonanie czynności 6. Użytkownik potwierdza dołączenie poziomu dostępu 7. Aplikacja przydziela poziom dostępu do konta 8. Aplikacja wyświetla komunikat o poprawnym przypisaniu dostępu
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.7 Zmień własne hasło
Aktorzy	Administrator, Farmaceuta, Pacjent

Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję zmiany własnego hasła 2. Aplikacja wyświetla formularz zmiany hasła 3. Użytkownik wypełnia formularz zmiany hasła i zatwierdza je 4. Aplikacja wyświetla prośbę o potwierdzenie zmiany hasła 5. Użytkownik potwierdza operację 6. Aplikacja potwierdza poprawność nowego hasła 7. Aplikacja weryfikuje wartość starego hasła 8. Aplikacja aktualizuje hasło użytkownika 9. Aplikacja wysyła komunikat o pomyślnej zmianie hasła i wylogowuje użytkownika
Rozszerzenia	<ol style="list-style-type: none"> 5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj" 6a. Hasło nie spełnia kryteriów - użytkownik musi wprowadzić hasło jeszcze raz 7a Stare hasło jest niepoprawne

Nazwa przypadku użycia	MOK.8 Zmień hasło innego użytkownika
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję zmiany hasła na stronie z informacjami o koncie wybranego użytkownika 2. Aplikacja wyświetla formularz zmiany hasła 3. Użytkownik wypełnia formularz zmiany hasła i zatwierdza je 4. Aplikacja wyświetla prośbę o potwierdzenie zmiany hasła 5. Użytkownik potwierdza operację 6. Aplikacja potwierdza zgodność podanych danych 7. Aplikacja aktualizuje hasło użytkownika 8. Aplikacja wysyła komunikat o pomyślnej zmianie hasła
Rozszerzenia	<ol style="list-style-type: none"> 5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj" 6a. Hasło nie spełnia kryteriów - użytkownik musi wprowadzić hasło jeszcze raz - aplikacja wyświetla komunikat o błędzie

Nazwa przypadku użycia	MOK.9 Edytuj dane własnego konta
Aktorzy	Administrator, Farmaceuta, Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję edycji danych 2. Aplikacja wyświetla formularz zmiany danych 3. Użytkownik zmienia dane i zatwierdza formularz 4. Aplikacja wyświetla prośbę o potwierdzenie zmiany danych 5. Użytkownik potwierdza operację 6. Aplikacja potwierdza poprawność danych 7. Aplikacja aktualizuje dane użytkownika 8. Aplikacja wyświetla komunikat o pomyślnej zmianie danych
Rozszerzenia	5a Podane dane są niepoprawne i użytkownik musi ponownie podać dane - aplikacja wyświetla komunikat o błędzie

Nazwa przypadku użycia	MOK.10 Edytuj dane konta innego użytkownika
------------------------	---

Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję edytowania konta innego użytkownika 2. Aplikacja wyświetla formularz do zmiany danych 3. Użytkownik zmienia dane i zatwierdza formularz 4. Aplikacja wyświetla prośbę o potwierdzenie zmian 5. Użytkownik potwierdza operację 6. Aplikacja potwierdza zgodność podanych danych 7. Aplikacja aktualizuje dane konta, wyświetla komunikat o pomyślnej aktualizacji danych oraz wysyła e-mail informujący o zmianie danych
Rozszerzenia	6a. Wprowadzone dane są niepoprawne - aplikacja wyświetla komunikat o niepoprawności danych

Nazwa przypadku użycia	MOK.11 Wyloguj
Aktorzy	Administrator, Farmaceuta, Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wylogowania 2. Aplikacja wyświetla komunikat o potwierdzeniu wylogowania 3. Użytkownik potwierdza wylogowanie 4. Aplikacja porzuca token JWT 5. Użytkownik zostaje przekierowany na stronę główną
Rozszerzenia	3a. Użytkownik anuluje wylogowanie

Nazwa przypadku użycia	MOK.12 Usuń konto w przypadku braku aktywacji
Aktorzy	System
Scenariusz główny	<ol style="list-style-type: none"> 1. Aplikacja oczekuje określony czas na potwierdzenie rejestracji przez użytkownika 2. Aplikacja usuwa konto jeśli minie określony czas
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.13 Wyświetl informacje o koncie
Aktorzy	Administrator, Farmaceuta, Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia informacji o swoim koncie 2. Aplikacja wyświetla informacje o koncie użytkownika
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.14 Wyświetl listę kont
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia listy kont 2. Aplikacja wyświetla listę wszystkich kont
Rozszerzenia	brak

Nazwa przypadku użycia	MOK.15 Wyświetl informacje o koncie innego użytkownika
Aktorzy	Administrator
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wyszukuje użytkownika na liście i na danej pozycji wybiera opcję informacji o koncie 2. Aplikacja wyświetla informacje na temat wybranego konta

Rozszerzenia	brak
Nazwa przypadku użycia	MOK.16 Zresetuj hasło
Aktorzy	Gość
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik zgłasza chęć zresetowania hasła 2. Aplikacja wyświetla formularz do wprowadzenia adresu e-mail powiązanego z kontem użytkownika 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyszukuje konto o podanym adresie email 5. Aplikacja wysyła na podany adres e-mail link do resetowania hasła 6. Użytkownik wchodzi na stronę podaną w wiadomości email 7. Aplikacja wyświetla formularz związanego z ustawniem nowego hasła 8. Użytkownik wypełnia i zatwierdza formularz 9. Aplikacja sprawdza poprawność podanych w formularzu danych 10. Aplikacja ustawia nową wartość hasła użytkownika 11. Aplikacja wyświetla komunikat o poprawnej zmianie hasła i przekierowuje użytkownika na stronę
Rozszerzenia	<ol style="list-style-type: none"> 4a. Aplikacja wyświetla komunikat o niepoprawnym adresie email 9a Podane hasła nie spełniają kryteriów - aplikacja wyświetla komunikat o niespełnieniu kryteriów

Nazwa przypadku użycia	MOK.17 Zaloguj się na konto
Aktorzy	Gość
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownika zgłasza chęć zalogowania się 2. Aplikacja wyświetla formularz logowania 3. Użytkownik podaje dane potrzebne do logowania i zatwierdza formularz 4. Aplikacja potwierdza zgodność podanych danych 5. Aplikacja wyświetla stronę główną poziomu dostępu użytkownika
Rozszerzenia	<ol style="list-style-type: none"> 4a. Podane dane są niepoprawne i należy ponownie podać dane 5a. Użytkownik ma przypisany więcej niż jeden poziom dostępu <ol style="list-style-type: none"> 1. Aplikacja wyświetla listę z poziomami dostępu użytkownika 2. Użytkownik wybiera jeden z poziomów dostępu poprzez kliknięcie 3. Aplikacja wyświetla stronę główną wybranego poziomu dostępu

Nazwa przypadku użycia	MOA.1 Wyświetl listę leków
Aktorzy	Gość, Farmaceuta, Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia listy leków 2. Aplikacja wyświetla listę leków
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.2 Wyświetl stronę ze szczegółami leku
Aktorzy	Farmaceuta, Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera lek z listy i klikna na niego 2. Aplikacja przekierowuje użytkownika na stronę wybranego leku

Rozszerzenia	brak
--------------	------

Nazwa przypadku użycia	MOA.3 Dodaj lek do koszyka
Aktorzy	Pacjent
Scenariusz główny	1. Użytkownik wybiera opcję dodania danego leku z listy do zamówienia 2. Aplikacja dodaje lek do zamówienia użytkownika
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.4 Wyświetl koszyk
Aktorzy	Pacjent
Scenariusz główny	1. Użytkownik wybiera opcję wyświetlenia kompletowanego zamówienia 2. Aplikacja wyświetla listę produktów dodanych do zamówienia
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.5 Zmień liczebność leku w koszyku
Aktorzy	Pacjent
Scenariusz główny	1. Użytkownik wybiera opcję zmiany ilości leku w koszyku 2. Aplikacja wyświetla formularz zmiany liczebności leku 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyświetla prośbę o potwierdzenie zmiany danych 5. Użytkownik potwierdza zmianę danych 6. Aplikacja weryfikuje formularz 7. Aplikacja aktualizuje ilość leku w zamówieniu 8. Aplikacja wyświetla komunikat o poprawnej zmianie ilości leku
Rozszerzenia	5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj" 6a. Podana wartość jest niepoprawna - aplikacja wyświetla komunikat o złej wartości

Nazwa przypadku użycia	MOA.6 Usuń lek z koszyka
Aktorzy	Pacjent
Scenariusz główny	1. Użytkownik kliką przycisk usuń przy wybranej pozycji z zamówienia 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza usunięcie pozycji z zamówienia 4. Aplikacja usuwa dany lek z koszyka 5. Aplikacja wyświetla komunikat o usunięciu leku z koszyka
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.7 Złoż zamówienie
Aktorzy	Pacjent

Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję złożenia zamówienia 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza złożenie zamówienia 4. Aplikacja potwierdza możliwość wykonania zamówienia 5. Aplikacja wyświetla komunikat o poprawnym złożeniu zamówienia
Rozszerzenia	<p>3a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"</p> <p>4a. Leki w zamówieniu nie są dostępne na stanie, system dodaje zamówienie do kolejki zamówień oczekujących</p> <p>4b. Zamówienie zawiera leki na receptę, system dodaje zamówienie do listy do zatwierdzenia</p>

Nazwa przypadku użycia	MOA.8 Wycofaj złożone zamówienie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera zamówienie z listy swoich zamówień i kliką przycisk wycofania zamówienia - jeśli zamówienie jest w kolejce do realizacji 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza wycofanie złożonego zamówienia 4. Aplikacja usuwa zamówienie z kolejki zamówień 5. Aplikacja wyświetla komunikat o poprawnym wycofaniu zamówienia
Rozszerzenia	3a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"

Nazwa przypadku użycia	MOA.9 Wyświetl kolejkę zamówień oczekujących
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia kolejki zamówień oczekujących 2. Aplikacja wyświetla listę zamówień oczekujących
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.10 Usuń zamówienie z kolejki zamówień oczekujących
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik kliką przycisk usunięcia wybranego zamówienia z kolejki 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza wycofanie złożonego zamówienia 4. Aplikacja usuwa wybrane zamówienie z kolejki 5. Aplikacja wyświetla komunikat o usunięciu zamówienia z kolejki
Rozszerzenia	3a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"

Nazwa przypadku użycia	MOA.11 Wprowadź zawartość dostawy
Aktorzy	Farmaceuta

Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik kliką przycisk wprowadź zawartość dostawy 2. Aplikacja wyświetla formularz wprowadzania zawartości dostawy 3. Użytkownik wprowadza leki oraz ich liczebność, które przyszły w dostawie 4. Użytkownik przesyła formularz wprowadzania zawartości dostawy 5. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 6. Aplikacja zapisuje zawartość dostawy 7. Aplikacja wyświetla komunikat o pomyślnie wykonanej operacji
Rozszerzenia	3a. Użytkownik dodaje za pomocą osobnego formularza dane leków, które doszły w dostawie ale nie ma ich w systemie

Nazwa przypadku użycia	MOA.12 Wyświetl listę zamówień oczekujących na potwierdzenie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia kolejki zamówień oczekujących na potwierdzenie 2. Aplikacja wyświetla listę zamówień oczekujących na potwierdzenie
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.13 Potwierdź zamówienie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera z listy zamówienie i kliką przy nim przycisk potwierdzenia 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza wycofanie złożonego zamówienia 4. Aplikacja potwierdza zamówienie 5. Aplikacja wyświetla komunikat o potwierdzeniu zamówienia
Rozszerzenia	3a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"

Nazwa przypadku użycia	MOA.14 Odwołaj zamówienie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera z listy zamówienie i kliką przy nim przycisk odwołania 2. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 3. Użytkownik potwierdza wycofanie złożonego zamówienia 4. Aplikacja potwierdza zamówienie 5. Aplikacja wyświetla komunikat o odwołaniu zamówienia
Rozszerzenia	3a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"

Nazwa przypadku użycia	MOA.15 Pokaż statystyki zakupów
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję pokazania statystki zakupów 2. Aplikacja wyświetla statystyki sprzedaży
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.16 Cyklicznie sprawdź kolejkę i aktualizuj ją
Aktorzy	System

Scenariusz główny	<ol style="list-style-type: none"> 1. Aplikacja porównuje stan magazynowy z kolejką zadań oczekujących na realizację 2. Aplikacja zatwierdza zamówienie, które mogą być skompletowane
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.17 Wyświetl listę złożonych przez siebie zamówień zamówień
Aktorzy	Pacjent
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik zgłasza chęć wyświetlenia listy złożonych zamówień 2. Aplikacja wyświetla listę złożonych zamówień
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.18 Wyświetl listę wszystkich zamówień
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia wszystkich zamówień 2. Aplikacja wyświetla listę wszystkich zamówień
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.19 Dodaj lek do oferty
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję dodania leku do oferty 2. Aplikacja wyświetla formularz dodania leku 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 5. Użytkownik zatwierdza akcję 6. Aplikacja sprawdza poprawność danych w formularzu 7. Aplikacja zapisuje dodany lek 8. Aplikacja wyświetla komunikat o poprawnym dodaniu leku
Rozszerzenia	<ol style="list-style-type: none"> 5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj" 6a. Dane w formularzu są niepoprawne i trzeba je wprowadzić ponownie - aplikacja wyświetla komunikat o niepoprawnych danych

Nazwa przypadku użycia	MOA.20 Edytuj dane leku
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję edytowania leku do oferty 2. Aplikacja wyświetla formularz edytowania leku 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 5. Użytkownik zatwierdza akcję 6. Aplikacja sprawdza poprawność danych w formularzu 7. Aplikacja aktualizuje edytowany lek 8. Aplikacja wyświetla komunikat o poprawnym edytowaniu leku

Rozszerzenia	<p>5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"</p> <p>6a. Dane w formularzu są niepoprawne i trzeba je wprowadzić ponownie - aplikacja wyświetla komunikat o niepoprawnych danych</p>
--------------	--

Nazwa przypadku użycia	MOA.21 Dodaj kategorie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję dodania kategorii 2. Aplikacja wyświetla formularz dodania leku 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 5. Użytkownik zatwierdza akcję 6. Aplikacja sprawdza poprawność danych w formularzu 7. Aplikacja zapisuje dodany lek 8. Aplikacja wyświetla komunikat o poprawnym dodaniu leku
Rozszerzenia	<p>5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"</p> <p>6a. Dane w formularzu są niepoprawne i trzeba je wprowadzić ponownie - aplikacja wyświetla komunikat o niepoprawnych danych</p>

Nazwa przypadku użycia	MOA.22 Wyświetl kategorie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję wyświetlenia wszystkie kategorie 2. Aplikacja wyświetla listę wszystkich kategorii
Rozszerzenia	brak

Nazwa przypadku użycia	MOA.23 Edytuj kategorie
Aktorzy	Farmaceuta
Scenariusz główny	<ol style="list-style-type: none"> 1. Użytkownik wybiera opcję edytowania kategorii 2. Aplikacja wyświetla formularz edytowania kategorii 3. Użytkownik wypełnia i zatwierdza formularz 4. Aplikacja wyświetla okno z prośbą potwierdzenia akcji 5. Użytkownik zatwierdza akcję 6. Aplikacja sprawdza poprawność danych w formularzu 7. Aplikacja aktualizuje edytowaną kategorię 8. Aplikacja wyświetla komunikat o poprawnym edytowaniu kategorii
Rozszerzenia	<p>5a. Użytkownik anuluje zmianę hasła klikając przycisk "anuluj"</p> <p>6a. Dane w formularzu są niepoprawne i trzeba je wprowadzić ponownie - aplikacja wyświetla komunikat o niepoprawnych danych</p>

3 Struktury relacyjnej bazy danych 2 pkt

Struktury tabel

Struktury tabel

```
CREATE TABLE public.access_level (
    access_level_role character varying(31) NOT NULL,
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    active boolean DEFAULT true NOT NULL,
    account_id bigint NOT NULL
);

ALTER TABLE public.access_level OWNER TO ssbd01admin;

CREATE SEQUENCE public.access_level_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.access_level_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.access_level_id_seq OWNED BY public.access_level.id;

CREATE TABLE public.account (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    active boolean NOT NULL,
    confirmed boolean DEFAULT false NOT NULL,
    email character varying(50) NOT NULL,
    incorrect_login_attempts integer,
    language character varying(255) NOT NULL,
    last_negative_login timestamp(6) without time zone,
    last_positive_login timestamp(6) without time zone,
    logical_address character varying(255),
    login character varying(50) NOT NULL,
    password character varying(255) NOT NULL
);

ALTER TABLE public.account OWNER TO ssbd01admin;

CREATE SEQUENCE public.account_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.account_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.account_id_seq OWNED BY public.account.id;
```

```

CREATE TABLE public.admin_data (
    work_phone_number character varying(255) NOT NULL,
    id bigint NOT NULL
);

ALTER TABLE public.admin_data OWNER TO ssbd01admin;

CREATE TABLE public.category (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    is_on_prescription boolean NOT NULL,
    category_name character varying(255) NOT NULL
);

ALTER TABLE public.category OWNER TO ssbd01admin;

CREATE SEQUENCE public.category_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.category_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.category_id_seq OWNED BY public.category.id;

CREATE TABLE public.chemist_data (
    license_number character varying(255) NOT NULL,
    id bigint NOT NULL
);

ALTER TABLE public.chemist_data OWNER TO ssbd01admin;

CREATE VIEW public.glassfish_auth_view AS
SELECT account.login,
       account.password,
       al.access_level_role
  FROM (public.access_level al
        JOIN public.account ON ((al.account_id = account.id)))
 WHERE ((account.confirmed = true) AND (account.active = true) AND (al.active = true));

ALTER TABLE public.glassfish_auth_view OWNER TO ssbd01admin;

CREATE TABLE public.medication (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    medication_name character varying(255) NOT NULL,
    price numeric(12,2) NOT NULL,

```

```

stock integer NOT NULL,
category_id bigint NOT NULL,
CONSTRAINT medication_price_check CHECK ((price >= (0)::numeric)),
CONSTRAINT medication_stock_check CHECK ((stock >= 0))
);

ALTER TABLE public.medication OWNER TO ssbd01admin;

CREATE SEQUENCE public.medication_id_seq
START WITH 1
INCREMENT BY 1
NO MINVALUE
NO MAXVALUE
CACHE 1;

ALTER TABLE public.medication_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.medication_id_seq OWNED BY public.medication.id;

CREATE TABLE public.order_medication (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    quantity integer NOT NULL,
    medication_id bigint NOT NULL,
    order_id bigint NOT NULL,
    CONSTRAINT order_medication_quantity_check CHECK ((quantity >= 1))
);
ALTER TABLE public.order_medication OWNER TO ssbd01admin;

CREATE SEQUENCE public.order_medication_id_seq
START WITH 1
INCREMENT BY 1
NO MINVALUE
NO MAXVALUE
CACHE 1;

ALTER TABLE public.order_medication_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.order_medication_id_seq OWNED BY public.order_medication.id;

CREATE TABLE public.patient_data (
    nip character varying(255) NOT NULL,
    first_name character varying(50) NOT NULL,
    last_name character varying(50) NOT NULL,
    pesel character varying(255) NOT NULL,
    phone_number character varying(255) NOT NULL,
    id bigint NOT NULL
);
ALTER TABLE public.patient_data OWNER TO ssbd01admin;

CREATE TABLE public.patient_order (

```

```

        id bigint NOT NULL,
        created_by character varying(255),
        creation_date timestamp(6) without time zone NOT NULL,
        modification_date timestamp(6) without time zone,
        modified_by character varying(255),
        version bigint,
        in_queue boolean NOT NULL,
        order_date timestamp(6) without time zone NOT NULL,
        chemist_data_id bigint,
        patient_data_id bigint NOT NULL,
        prescription_id bigint
    );
}

ALTER TABLE public.patient_order OWNER TO ssbd01admin;

CREATE SEQUENCE public.patient_order_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.patient_order_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.patient_order_id_seq OWNED BY public.patient_order.id;

CREATE TABLE public.prescription (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    prescription_number character varying(255) NOT NULL,
    patient_data_id bigint NOT NULL
);
ALTER TABLE public.prescription OWNER TO ssbd01admin;

CREATE SEQUENCE public.prescription_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.prescription_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.prescription_id_seq OWNED BY public.prescription.id;

CREATE TABLE public.shipment (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    shipment_date timestamp(6) without time zone NOT NULL
);

```

```

ALTER TABLE public.shipment OWNER TO ssbd01admin;

CREATE SEQUENCE public.shipment_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.shipment_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.shipment_id_seq OWNED BY public.shipment.id;

CREATE TABLE public.shipment_medication (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    quantity integer NOT NULL,
    medication_id bigint NOT NULL,
    shipment_id bigint NOT NULL,
    CONSTRAINT shipment_medication_quantity_check CHECK ((quantity >= 1))
);
ALTER TABLE public.shipment_medication OWNER TO ssbd01admin;

CREATE SEQUENCE public.shipment_medication_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.shipment_medication_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.shipment_medication_id_seq OWNED BY public.shipment_medication.id;

CREATE TABLE public.token (
    id bigint NOT NULL,
    created_by character varying(255),
    creation_date timestamp(6) without time zone NOT NULL,
    modification_date timestamp(6) without time zone,
    modified_by character varying(255),
    version bigint,
    code character varying(100) NOT NULL,
    expiration_date timestamp(6) without time zone NOT NULL,
    used boolean DEFAULT false NOT NULL,
    token_type character varying(255) NOT NULL,
    was_previous_token_sent boolean DEFAULT false NOT NULL,
    account_id bigint NOT NULL
);
ALTER TABLE public.token OWNER TO ssbd01admin;

```

```

CREATE SEQUENCE public.token_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;

ALTER TABLE public.token_id_seq OWNER TO ssbd01admin;

ALTER SEQUENCE public.token_id_seq OWNED BY public.token.id;

ALTER TABLE ONLY public.access_level ALTER COLUMN id SET DEFAULT nextval('public.access_level_id_seq'::regclass);

ALTER TABLE ONLY public.account ALTER COLUMN id SET DEFAULT nextval('public.account_id_seq'::regclass);

ALTER TABLE ONLY public.category ALTER COLUMN id SET DEFAULT nextval('public.category_id_seq'::regclass);

ALTER TABLE ONLY public.medication ALTER COLUMN id SET DEFAULT nextval('public.medication_id_seq'::regclass);

ALTER TABLE ONLY public.order_medication ALTER COLUMN id SET DEFAULT nextval('public.order_medication_id_seq'::regclass);

ALTER TABLE ONLY public.patient_order ALTER COLUMN id SET DEFAULT nextval('public.patient_order_id_seq'::regclass);

ALTER TABLE ONLY public.prescription ALTER COLUMN id SET DEFAULT nextval('public.prescription_id_seq'::regclass);

ALTER TABLE ONLY public.shipment ALTER COLUMN id SET DEFAULT nextval('public.shipment_id_seq'::regclass);

ALTER TABLE ONLY public.shipment_medication ALTER COLUMN id SET DEFAULT nextval('public.shipment_medication_id_seq'::regclass);

ALTER TABLE ONLY public.token ALTER COLUMN id SET DEFAULT nextval('public.token_id_seq'::regclass);

ALTER TABLE ONLY public.access_level
    ADD CONSTRAINT access_level_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.account
    ADD CONSTRAINT account_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.admin_data

```

```

ADD CONSTRAINT admin_data_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.medication
    ADD CONSTRAINT category_index UNIQUE (category_id);

ALTER TABLE ONLY public.category
    ADD CONSTRAINT category_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.patient_order
    ADD CONSTRAINT chemist_data_index UNIQUE (chemist_data_id);

ALTER TABLE ONLY public.chemist_data
    ADD CONSTRAINT chemist_data_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.shipment_medication
    ADD CONSTRAINT medication_index UNIQUE (medication_id);

ALTER TABLE ONLY public.order_medication
    ADD CONSTRAINT medication_index_om UNIQUE (medication_id);

ALTER TABLE ONLY public.medication
    ADD CONSTRAINT medication_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.order_medication
    ADD CONSTRAINT order_index UNIQUE (order_id);

ALTER TABLE ONLY public.order_medication
    ADD CONSTRAINT order_medication_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.patient_order
    ADD CONSTRAINT patient_data_index UNIQUE (patient_data_id);

ALTER TABLE ONLY public.prescription
    ADD CONSTRAINT patient_data_index_prescription UNIQUE (patient_data_id);

ALTER TABLE ONLY public.patient_data
    ADD CONSTRAINT patient_data_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.patient_order
    ADD CONSTRAINT patient_order_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.patient_order
    ADD CONSTRAINT prescription_index UNIQUE (prescription_id);

```

```

ALTER TABLE ONLY public.prescription
ADD CONSTRAINT prescription_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.shipment_medication
ADD CONSTRAINT shipment_index UNIQUE (shipment_id);

ALTER TABLE ONLY public.shipment_medication
ADD CONSTRAINT shipment_medication_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.shipment
ADD CONSTRAINT shipment_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.token
ADD CONSTRAINT token_pkey PRIMARY KEY (id);

ALTER TABLE ONLY public.chemist_data
ADD CONSTRAINT uk_4f10w9rn7xyf5iyhu832yyt5 UNIQUE (license_number);

ALTER TABLE ONLY public.account
ADD CONSTRAINT uk_5vxwyorsr92jce3ore6h93k6q UNIQUE (login);

ALTER TABLE ONLY public.patient_data
ADD CONSTRAINT uk_e4vdrsqh4hno0qx1q7wjhc4fa UNIQUE (phone_number);

ALTER TABLE ONLY public.token
ADD CONSTRAINT uk_ibgcuaiv8i3op4ck6lg08jjxs UNIQUE (code);

ALTER TABLE ONLY public.patient_data
ADD CONSTRAINT uk_kkkhwobiyl2jf2nvqhosaucd UNIQUE (pesel);

ALTER TABLE ONLY public.patient_data
ADD CONSTRAINT uk_kw3y4hqfonff1wbunk9up5e UNIQUE (nip);

ALTER TABLE ONLY public.category
ADD CONSTRAINT uk_lroeo5fvfdeg4hpicn4lw7x9b UNIQUE (category_name);

ALTER TABLE ONLY public.medication
ADD CONSTRAINT uk_nwsh0tbac94tvrhxk5jy2bchc UNIQUE (medication_name);

ALTER TABLE ONLY public.account
ADD CONSTRAINT uk_q0uja26qgu1atulenwup9rxyr UNIQUE (email);

```

```

ALTER TABLE ONLY public.prescription
  ADD CONSTRAINT ukl3f9254wwc2bwulllsinmwwmc UNIQUE (patient_data_id, prescription_number);

ALTER TABLE ONLY public.access_level
  ADD CONSTRAINT ukr0n7odr6w5omf264fj9ira603 UNIQUE (access_level_role, account_id);

ALTER TABLE ONLY public.chemist_data
  ADD CONSTRAINT fk2518a73nwmjgbrutrbk39pxlk FOREIGN KEY (id) REFERENCES public.access_level(id);

ALTER TABLE ONLY public.prescription
  ADD CONSTRAINT fk32lg8tjt168velulecaifjqod FOREIGN KEY (patient_data_id) REFERENCES public.patient_data(id);

ALTER TABLE ONLY public.patient_order
  ADD CONSTRAINT fk6ir7prtux5xlf05o1pw5ynpmj0 FOREIGN KEY (patient_data_id) REFERENCES public.patient_data(id);

ALTER TABLE ONLY public.shipment_medication
  ADD CONSTRAINT fk7rop3obtwh294s9hp5dvh3bml FOREIGN KEY (medication_id) REFERENCES public.medication(id);

ALTER TABLE ONLY public.medication
  ADD CONSTRAINT fka6f8yod9ghvgau0wu9usasst1 FOREIGN KEY (category_id) REFERENCES public.category(id);

ALTER TABLE ONLY public.order_medication
  ADD CONSTRAINT fkbsoyeu0wubwchlyah237te0q9 FOREIGN KEY (medication_id) REFERENCES public.medication(id);

ALTER TABLE ONLY public.patient_order
  ADD CONSTRAINT fkbu7dad0yosvq7aykg9hjkh06 FOREIGN KEY (chemist_data_id) REFERENCES public.chemist_data(id);

ALTER TABLE ONLY public.shipment_medication
  ADD CONSTRAINT fkdpshyk5k6kt2wp4kf1ugab4 FOREIGN KEY (shipment_id) REFERENCES public.shipment(id);

ALTER TABLE ONLY public.admin_data
  ADD CONSTRAINT fkeph7dr69cfma0jravsg8fwhua FOREIGN KEY (id) REFERENCES public.access_level(id);

ALTER TABLE ONLY public.patient_data
  ADD CONSTRAINT fkeyt8nn072vqk2h2vxfl2mgfbp FOREIGN KEY (id) REFERENCES public.access_level(id);

ALTER TABLE ONLY public.token
  ADD CONSTRAINT fkftkstvcfb74ogw02bo5261kno FOREIGN KEY (account_id) REFERENCES public.account(id);

ALTER TABLE ONLY public.access_level
  ADD CONSTRAINT fkl6ljrwl5w8nhxvrrqt0pu4uoou FOREIGN KEY (account_id) REFERENCES public.account(id);

```

```
ALTER TABLE ONLY public.order_medication
ADD CONSTRAINT fkoydqp2wk8j7wcs33a58wukhjt FOREIGN KEY (order_id) REFERENCES public.patient_order(id);
```

```
ALTER TABLE ONLY public.patient_order
ADD CONSTRAINT fkqko3hvaac6yftkkoclbx4kvv FOREIGN KEY (prescription_id) REFERENCES public.prescription(id);
```

Dane inicjujące

Dane inicjujące

```
CREATE VIEW public.glassfish_auth_view AS SELECT account.login, account.password, al.access_level_role FROM (public.access_level al JOIN public.account ON ((al.account_id = account.id))) WHERE ((account.confirmed = true) AND (account.active = true) AND (al.active = true));

REVOKE USAGE ON SCHEMA public FROM PUBLIC;
GRANT ALL ON SCHEMA public TO PUBLIC;

GRANT SELECT,INSERT,UPDATE ON TABLE public.access_level TO ssbd01mok;
GRANT SELECT ON TABLE public.access_level TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.account TO ssbd01mok;
GRANT SELECT ON TABLE public.account TO ssbd01moa;

GRANT SELECT,INSERT,UPDATE ON TABLE public.admin_data TO ssbd01mok;
GRANT SELECT ON TABLE public.admin_data TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.category TO ssbd01moa;

GRANT SELECT,INSERT,UPDATE ON TABLE public.chemist_data TO ssbd01mok;
GRANT SELECT ON TABLE public.chemist_data TO ssbd01moa;

GRANT SELECT ON TABLE public.glassfish_auth_view TO ssbd01glassfish;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.medication TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.order_medication TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.patient_data TO ssbd01mok;
GRANT SELECT ON TABLE public.patient_data TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.patient_order TO ssbd01moa;

GRANT SELECT,INSERT,DELETE ON TABLE public.prescription TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.shipment TO ssbd01moa;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.shipment_medication TO ssbd01moa;

GRANT SELECT, UPDATE ON TABLE public.account_id_seq TO ssbd01mok;

GRANT SELECT, UPDATE ON TABLE public.access_level_id_seq TO ssbd01mok;

GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.token TO ssbd01mok;
GRANT SELECT,INSERT,DELETE,UPDATE ON TABLE public.token_id_seq TO ssbd01mok;

insert into account (active, confirmed, created_by, creation_date, email, language, last_negative_login, last_positive_login, logical_address, login_incorrect_login_attempts, modification_date, modified_by, password, version) values (true, true, null, now(), 'admin@o2.pl', 'en', null, null, null, 'admin123', 0, null, null, 'b03ddf3ca2e714a6548e7495e2a03f5e824eaac9837cd7f159c67b90fb4b7342', 0);
insert into account (active, confirmed, created_by, creation_date, email, language, last_negative_login, last_positive_login, logical_address, login_incorrect_login_attempts, modification_date, modified_by, password, version) values (true, true, null, now(), 'chemist@o2.pl', 'en', null, null, null, 'chemist123', 0, null, null, '52d7a0431ddd469b9e0929a09ef67fefef99e3511893edb0c2fa0b09892df1e52', 0);

insert into access_level (account_id, active, created_by, creation_date, modification_date, modified_by, version, access_level_role) values (1, true, null, now(), null, null, 0, 'ADMIN');
insert into access_level (account_id, active, created_by, creation_date, modification_date, modified_by, version, access_level_role) values (2, true, null, now(), null, null, 0, 'CHEMIST');
insert into access_level (account_id, active, created_by, creation_date, modification_date, modified_by, version, access_level_role) values (1, true, null, now(), null, null, 0, 'CHEMIST');

insert into admin_data (id, work_phone_number) values (1, '123456789');
insert into chemist_data (id, license_number) values (2, '456456');
insert into chemist_data (id, license_number) values (1, '123123');
```


4 Użytkownicy bazodanowi 2 pkt

Lista użytkowników bazodanowych

Użytkownik	Hasło	Moduł	Opis
postgres	postgrespassword	-	Użytkownik z prawami administratora.
ssbd01admin	admin	mok, moa	Użytkownik, który jest właścicielem wszystkich tabel w bazie danych.
ssbd01mok	mokpassword	mok	Konto stworzone na potrzeby modułu obsługi kont (mok).
ssbd01moa	moapassword	moa	Konto stworzone na potrzeby modułu obsługi apteki (moa).
ssbd01glassfish	glassfishpassword	-	Użytkownik stworzony na potrzeby utworzenia puli połączeń w celu odczytania widoku i przeprowadzenia przez serwer aplikacyjny procesu uwierzytelnienia.

Tabela uprawnień użytkowników bazodanowych

Nazwa	Typ	postgres	ssbd01admin	ssbd01mok	ssbd01moa	ssbd01glassfish
account	tabela	a,r,w,d	a,r,w,d	a,r,w,d	r	
account_id_seq	sekwencja	a,r,w,d	a,r,w,d	r, w		
access_level	tabela	a,r,w,d	a,r,w,d	a,r,w	r	
access_level_id_seq	sekwencja	a,r,w,d	a,r,w,d	r, w		
admin_data	tabela	a,r,w,d	a,r,w,d	a,r,w	r	
category	tabela	a,r,w,d	a,r,w,d		a,r,w,d	
chemist_data	tabela	a,r,w,d	a,r,w,d	a,r,w	r	
glassfish_auth_view	widok	a,r,w,d	a,r,w,d			r
medication	tabela	a,r,w,d	a,r,w,d		a,r,w,d	
order_medication	tabela	a,r,w,d	a,r,w,d		a,r,w,d	
patient_data	tabela	a,r,w,d	a,r,w,d	a,r,w,d	r	
patient_order	tabela	a,r,w,d	a,r,w,d		a,r,w,d	
prescription	tabela	a,r,w,d	a,r,w,d		a,r,d	
shipment	tabela	a,r,w,d	a,r,w,d		a,r,w,d	
shipment_medication	tabela	a,r,w,d	a,r,w,,d		a,r,w,d	
token	tabela	a,r,w,d	a,r,w,d	a,r,w,d		
token_id_seq	sekwencja	a,r,w,d	a,r,w,d	r, w		

Legenda:

r – Select, prawo do odczytu

a – Insert, prawo do wstawiania wierszy

w – Update, prawo do aktualizacji wierszy

d – Delete, prawo do usuwania wierszy

Puste pole oznacza brak uprawnień.

5 Identyfikacja obiektów encji 2 pkt

Zestawienie klas encji

Informacje o polach klas encyjnych przedstawione są w następującym formatowaniu:

atrybut klasy encyjnej modelu obiektowego → oznaczenie odpowiadającej kolumny w tabeli → ograniczenia i dodatkowe informacje

Klasa **Account** → tabela **account**:

Klasa Account reprezentuje konto użytkownika w systemie. Przechowuje ona informacje takie jak login, hasło, adres email, poziomy dostępu oraz informacje dotyczące logowania. Celem klasy jest umożliwienie zarządzania kontami użytkowników w systemie.

- a. `Set<AccessLevel> accessLevels` → jest klucz obcy dla kolumny `BIGINT account_id` w tabeli `access_levels`, związek jeden do wielu, relacja dwukierunkowa
- b. `private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY` → ograniczenie `not null` → klucz główny, generowany automatycznie z użyciem strategii `GenerationType.IDENTITY`
- c. `private String login → login VARCHAR(255)` → ograniczenie `not null`, ograniczenie `unique`
- d. `private String email → email VARCHAR(255)` → ograniczenie `not null`, ograniczenie `unique`, ograniczenie typu wyrażenie regularne `"^/[A-Za-z0-9+-.]+@[.]+\$/"`
- e. `private String password → email VARCHAR(255)` → ograniczenie `not null`, ograniczenie typu wyrażenie regularne `"^/[0-9a-f]{64}\$/"`
- f. `private Boolean active → active BOOLEAN` → ograniczenie `not null`
- g. `private Boolean confirmed → confirmed BOOLEAN DEFAULT FALSE` → ograniczenie `not null`, domyślana wartość `false`
- h. `private Locale language → language VARCHAR(255)` → ograniczenie `not null`
- i. `private Date lastPositiveLogin → last_positive_login TIMESTAMP WITHOUT TIME ZONE`
- j. `private Date lastNegativeLogin → last_negative_login TIMESTAMP WITHOUT TIME ZONE`
- k. `private String logicalAddress → logical_address VARCHAR(255)`

Klasa abstrakcyjna **AccessLevel** → tabela **access_level**:

Klasa bazową służącą do implementacji różnych poziomów dostępu w systemie. W kontekście bazy danych, ww. klasa pełni rolę tabeli pośredniczącej pomiędzy kontami użytkowników a poziomami dostępu. Wykorzystuje się do tego strategię dziedziczenia JPA `IInheritanceType.JOINED`, co oznacza, że każdy poziom dostępu posiada swoją osobną tabelę, a dane związane z kontami użytkowników są przechowywane w tabeli głównej, a informacje o dostępach użytkowników są przechowywane w tabelach poziomów dostępu, z którymi są powiązane kluczami obcymi.

- a. `private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY` → ograniczenie `not null` → klucz główny, generowany automatycznie z użyciem strategii `GenerationType.IDENTITY`
- b. `private Role role → access_level_role VARCHAR(255)` → kolumna dyskryminatora, jako wartość przyjmuje nazwę klasy danego poziomu dostępu → ograniczenie `not null`, ograniczenie `JPA updatable = false`, do mapowania wykorzystano adnotację `@Enumerated(EnumType.STRING)`
- c. `private Boolean active → active BOOLEAN DEFAULT TRUE` → ograniczenie `not null`, domyślana wartość `true`
- d. `private Account account → account_id BIGINT` → odwołuje się w ramach klucza obcego do kolumny `id` z tabeli `accounts`, ograniczenie `JPA updatable = false`

Klasa **AdminData** → tabela **admin_data**:

Klasa, która reprezentuje poziom dostępu administratora w systemie. Klasa ta dziedziczy atrybuty i relacje z klasą `AccessLevel`

- a. `private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY` → ograniczenie `not null` → klucz główny, generowany automatycznie z użyciem strategii `GenerationType.IDENTITY`, klucz dostępny poprzez dziedziczenie z klasą `AccessLevel`

Klasa **ChemistData** → tabela **chemist_data**:

Klasa, która reprezentuje poziom dostępu aptekarza w systemie. Klasa ta dziedziczy atrybuty i relacje z klasą `AccessLevel`

- a. `private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY` → ograniczenie `not null` → klucz główny, generowany automatycznie z użyciem strategii `GenerationType.IDENTITY`, klucz dostępny poprzez dziedziczenie z klasą `AccessLevel`
- b. `private String licenseNumber → license_number VARCHAR(255)` → ograniczenie `not null`

Klasa **PatientData** → tabela **patient_data**:

Klasa, która reprezentuje poziom dostępu pacjenta w systemie. Klasa ta dziedziczy atrybuty i relacje z klasą `AccessLevel`

- a. `private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY` → ograniczenie `not null` → klucz główny, generowany automatycznie z użyciem strategii `GenerationType.IDENTITY`, klucz dostępny poprzez dziedziczenie z klasą `AccessLevel`

- b. private String pesel → pesel VARCHAR(255) → ograniczenie not null, ograniczenie JPA updatable = false, ograniczenie typu wyrażenie regularne "^(\\d{10}-\\d{3})\$"
- c. private String firstName → first_name VARCHAR(255) → ograniczenie not null
- d. private String lastName → last_name VARCHAR(255) → ograniczenie not null
- e. private String phoneNumber → phone_number VARCHAR(255) → ograniczenie not null, ograniczenie typu wyrażenie regularne "^(\\d{10}-\\d{3})\$"
- f. private String NIP → nip VARCHAR(255) → ograniczenie not null, ograniczenie typu wyrażenie regularne "^(\\d{10}-\\d{3}-\\d{2}-\\d{3}-\\d{2}-\\d{2})\$"

Klasa Medication → tabela medication :

Klasa **medication** reprezentuje fizyczny obiekt leku. Przechowuje ona informacje takie jak nazwa leku, dostępną ilość w magazynie, cenę za sztukę. Jest powiązana z encją **category** informującą o kategorii leku.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private String name → medication_name VARCHAR(255) → ograniczenie not null, ograniczenie unique
- c. private Integer stock → stock INTEGER → ograniczenie not null, ograniczenie dla minimalnej wartości jako zero
- d. private BigDecimal price → price DECIMAL → ograniczenie not null, ograniczenie dla minimalnej wartości jako zero, ograniczenie dopuszczalnych wartości liczbowych do 10 cyfr przed kropką i 2 cyfr po kropce
- e. private Category category → category_id BIGINT → odwołuje się w ramach klucza obcego do kolumny id z tabeli category, ograniczenie not null

Klasa Category → tabela category:

Klasa **Category** reprezentuje kategorię dla danego leku. Oprócz nazwy kategorii zawiera także informację, czy leki o danej kategorii są do kupienia tylko na receptę.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private String name → category_name VARCHAR(255) → ograniczenie not null oraz unique
- c. private boolean isOnPrescription → is_on_prescription BOOLEAN → ograniczenie not null

Klasa Shipment → tabela shipment:

Klasa **Shipment** reprezentuje dostawę leków i zawiera informacje o dacie wysyłki oraz listę leków przypisanych do przesyłki.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private Date shipmentDate → shipment_date TIMESTAMP WITHOUT TIME ZONE → ograniczenie not null, ograniczenie JPA updatable = false
- c. private List<ShipmentMedication> shipment_medications (relacja zmapowana po stronie klasy ShipmentMedication)

Klasa ShipmentMedication → tabela shipment_medication:

Klasa **ShipmentMedication** stanowi klasę encyjną i nie ma bezpośredniego odzwierciedlenia w logicznym modelu, za to zawiera informacje o jednym leku w jednej dostawie.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private Shipment shipment → shipment_id BIGINT → ograniczenie not null, odwołuje się w ramach klucza obcego do kolumny id z tabeli shipment, związek wiele do jednego
- c. private Medication medication → medication_id BIGINT → ograniczenie not null, odwołuje się w ramach klucza obcego do kolumny id z tabeli medication, związek wiele do jednego
- d. private Integer quantity → INTEGER → ograniczenie not null, ograniczenie dla minimalnej wartości jako jeden

Klasa Order → tabela patient_order:

Klasa **Order** reprezentuje zamówienie pacjenta w aptece. Zamówienie zawiera informacje o kupowanych lekach, danych pacjenta oraz o ewentualnych recepcie i danych przypisanego aptekarza do weryfikacji recepty.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private Boolean inQueue → in_queue BOOLEAN → ograniczenie not null
- c. private Date orderDate → order_date TIMESTAMP WITHOUT TIME ZONE → ograniczenie not null
- d. private List<OrderMedication> patient_order_medications (relacja zmapowana po stronie klasy OrderMedication)
- e. private Prescription prescription → prescription_id BIGINT → odwołuje się w ramach klucza obcego do kolumny id z tabeli prescription , opcjonalny związek jeden do jednego
- f. private PatientData patientData → patient_data_id BIGINT → ograniczenie not null, odwołuje się w ramach klucza obcego do kolumny id z tabeli patient_data, związek wiele do jednego

- g. private ChemistData chemistData → chemist_data_id BIGINT → odwołuje się w ramach klucza obcego do kolumny id z tabeli *chemist_data*, opcjonalny związek wiele do jednego

Klasa *OrderMedication* → tabela *order_medication*:

Klasa *OrderMedication* stanowi klasę encyjną i nie ma bezpośredniego odzwierciedlenia w logicznym modelu, za to zawiera informacje o jednym leku w jednym zamówieniu.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private Order order → patient_order_id BIGINT → odwołuje się w ramach klucza obcego do kolumny id z tabeli *patient_order*, związek wiele do jednego
- c. private Medication medication → medication_id BIGINT → odwołuje się w ramach klucza obcego do kolumny id z tabeli *medication*, związek wiele do jednego
- d. private Integer quantity → quantity INTEGER → ograniczenie not null, ograniczenie dla minimalnej wartości jako jeden

Klasa *Prescription* → tabela *prescription*:

Klasa *Prescription* reprezentuje obiekt recepty, niezbędnej do kupna leków o określonej kategorii. Numer recepty (*prescription_number*) jest unikalny tylko w połączeniu z ID pacjenta, dlatego w tej klasie znajduje się związek wiele do jednego z encją *patient_data*.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private PatientData patientData → patient_data_id BIGINT → ograniczenie not null oraz unique (razem z polem *prescription_number*), odwołuje się w ramach klucza obcego do kolumny id z tabeli *patient_data*, związek wiele do jednego
- c. private String prescriptionNumber → prescription_number VARCHAR(255) → ograniczenie not null oraz unique (razem z polem *patient_data_id*)

Klasa *Token* → tabela *token*:

Klasa *token* reprezentuje obiekt tokenu, zawierającego 8 znakowy kod, niezbędny do zatwierdzenia zmiany hasła/emailu.

- a. private Long id → id BIGINT GENERATED BY DEFAULT AS IDENTITY → ograniczenie not null → klucz główny, generowany automatycznie z użyciem strategii GenerationType.IDENTITY
- b. private String code → code Varchar(255) → ograniczenie not null oraz unique, oraz zakaz aktualizacji wartości → kod do aktywacji konta lub emailu
- c. private bool isUsed → is_used BOOLEAN → ograniczenie not null → wartość informująca czy kod został użyty
- d. private bool wasPreviousTokenSent → was_previous_token_sent BOOLEAN → ograniczenie not null → wartość informująca czy token został wysłany
- e. private Account account → account_id BIGINT → ograniczenie not null oraz zakaz aktualizacji wartości → odwołanie do klucza obcego do kolumny id z tabeli *accounts*, związek wiele do jednego
- f. private Date date → expiration_date Date → ograniczenie not null oraz zakaz aktualizacji wartości → data wygaśnięcia tokenu

Związki między obiektami klas encji

		Klasa podstawowa												
		AbstractEntity	Account	AccessLevel	AdminData	ChemistData	PatientData	Medication	Shipment	ShipmentMedication	Order	OrderMedication	Prescription	Category
Klasa powiązana	AbstractEntity													
Klasa powiązana	Account	<ul style="list-style-type: none"> • createdBy; 1-1; PR; NM • modifiedBy; 1-1; M 	*-1; NM											
	AccessLevel		1-*; PR, MR, RM; M											
	AdminData													
	ChemistData													
	PatientData													
	Medication													
	Shipment													
	ShipmentMedication									1-*; PR, RM; M				
	Order													
	OrderMedication											1-*; PR, RM; M		
	Prescription											1-1; PR; NM		
	Category									*-1; PR, RF; M				

Legenda:

Pole przedstawia związek pomiędzy "kласą podstawową" a "klasą powiązaną", gdzie właściwym związkiem jest klasa podstawowa i związek przebiega od klasy podstawowej do klasy powiązanej.

Zawartość pola opisuje charakter związku za pomocą następującego schematu:

1. Liczność związku:

- 1-1 - związek jeden do jednego,
- 1-* - związek jeden do wielu,
- *-1 - związek wiele do jednego.

2. Operacje kaskadowe:

- PR - CascadeType.PERSIST,
- RM - CascadeType.REMOVE,
- MR - CascadeType.MERGE,
- RF - CascadeType.REFRESH.

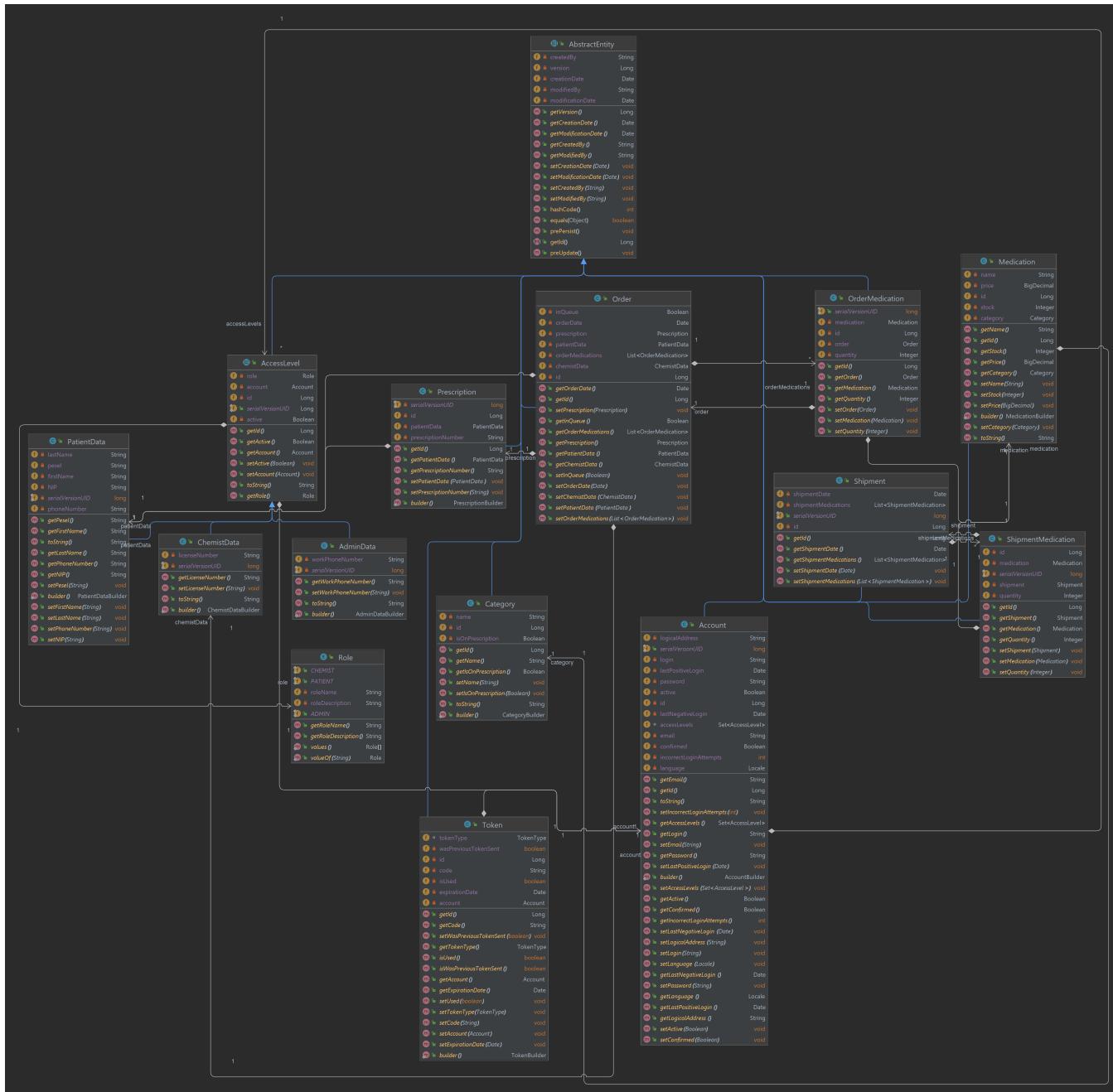
3. Czy relacja może ulec zmianie po utworzeniu:

- M - tak
- NM - nie

W przypadku występowania w klasie podstawowej wielu związków do tej samej klasy powiązanej takowe związki zostały wypunktowane oraz dodano nazwę atrybutu.

6 Diagram klas encji 1 pkt

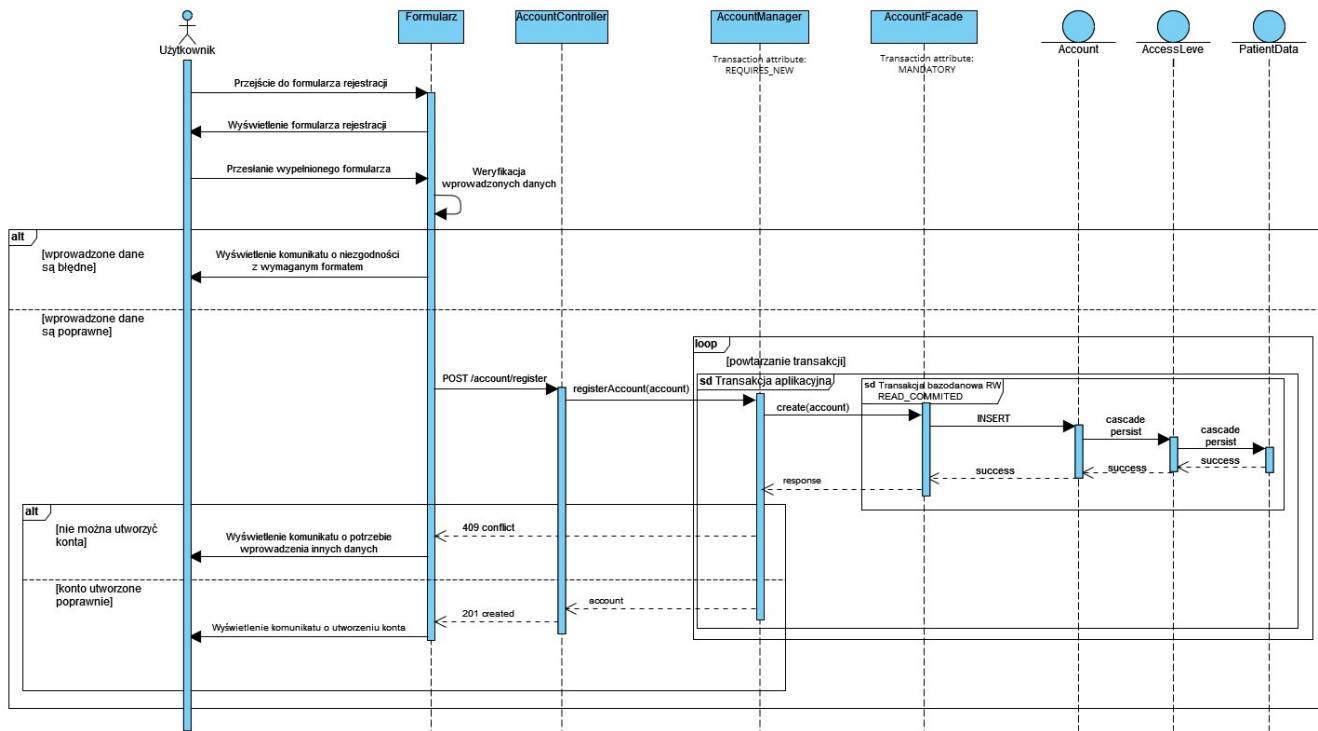
Diagram klas encji



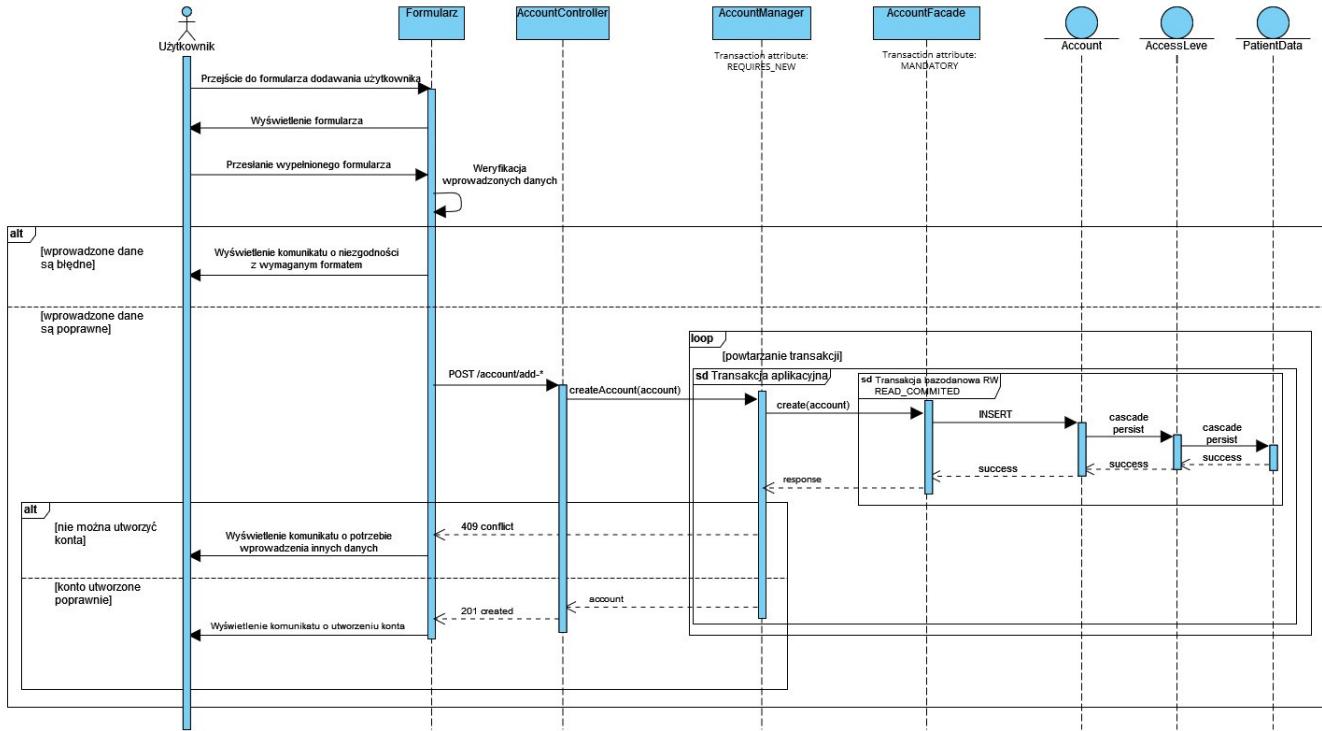
7 Diagramy sekwencji oraz identyfikacja transakcji bazodanowych 3 pkt

Na diagramach, żeby zaoszczędzić powtórzeń, skorzystano z notacji *, która oznacza odwołanie do jednej klasy lub encji spośród: PatientData, ChemistData, AdminData. W nazwach funkcji oznacza odwołanie do jednej z 3 metod zawierających zamiast znaku * 'Patient', 'Chemist' lub 'Admin'.

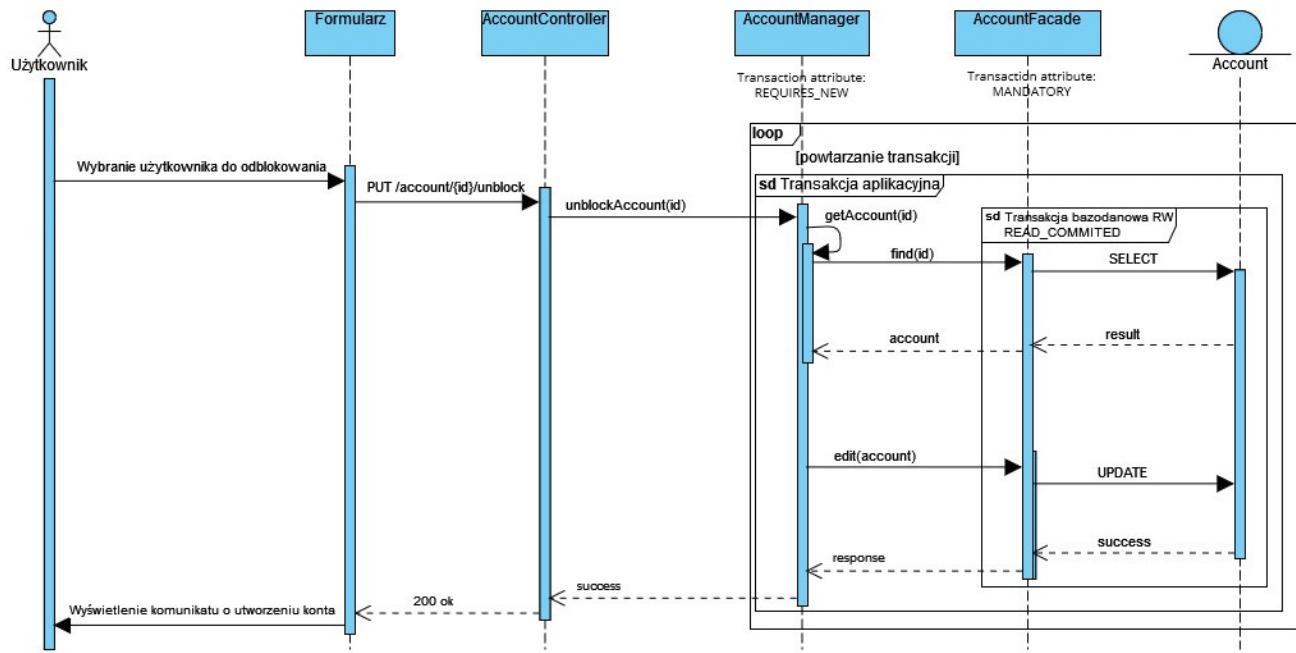
Formularz na diagramach to określenie na aplikację SPA.



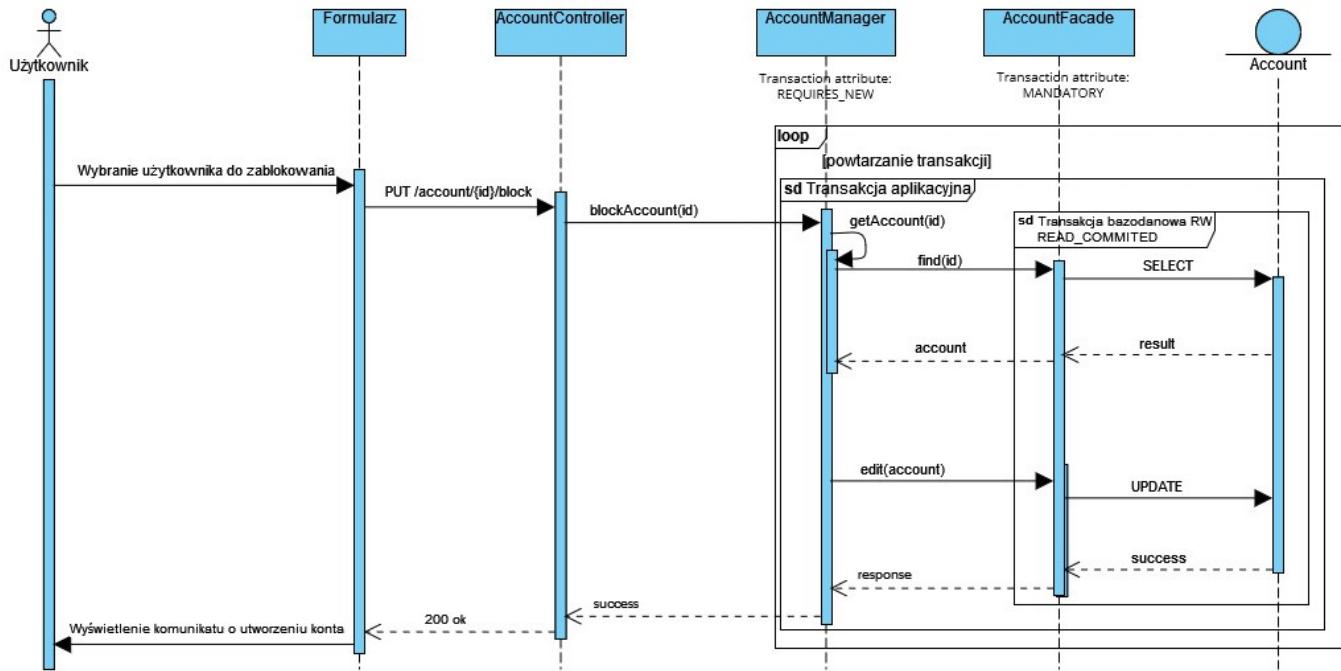
7.1 Diagram sekwencji dla przypadku użycia MOK 1 Zarejestruj.



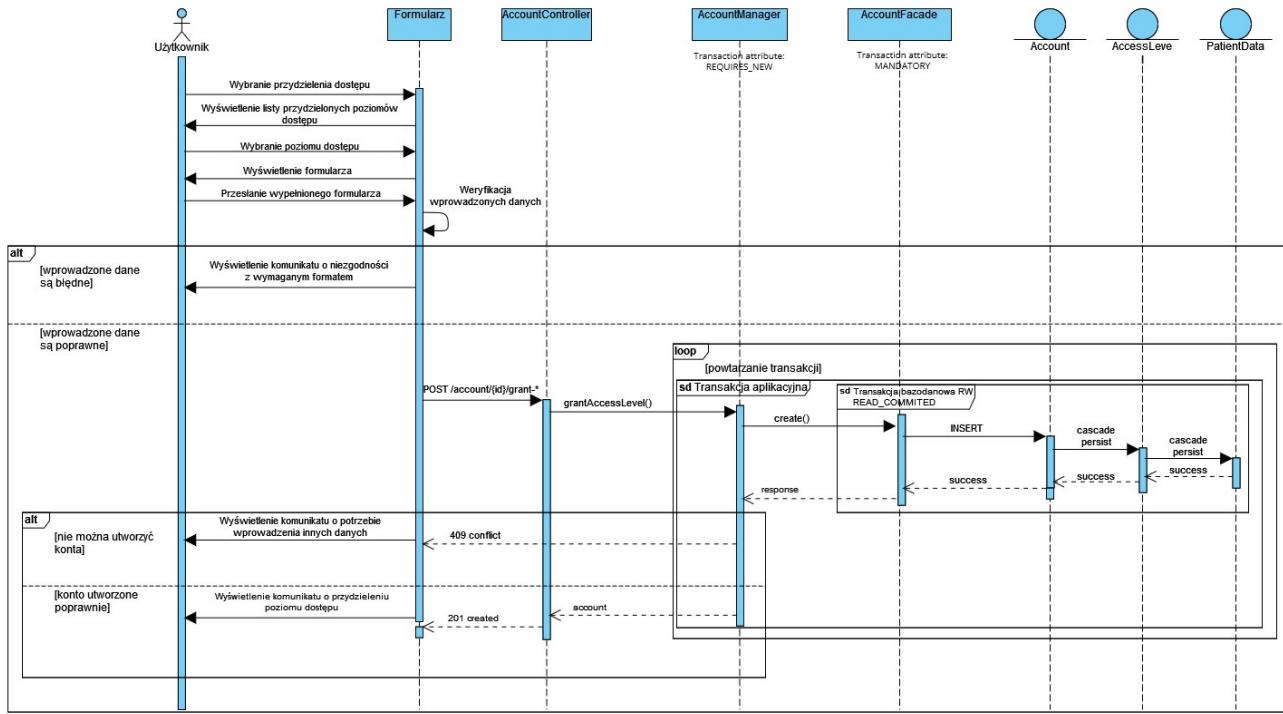
7.2 Diagram sekwencji dla przypadku użycia MOK 2 Utwórz konto.



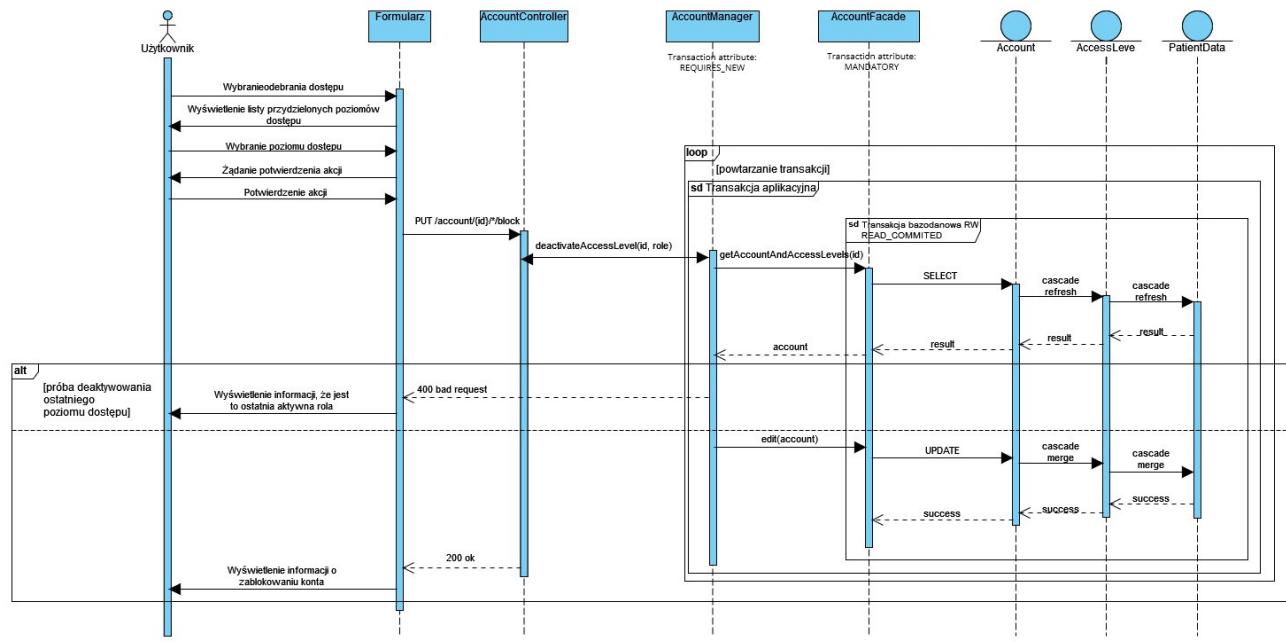
7.3 Diagram sekwencji dla przypadku użycia MOK 3 Zablokuj konto.



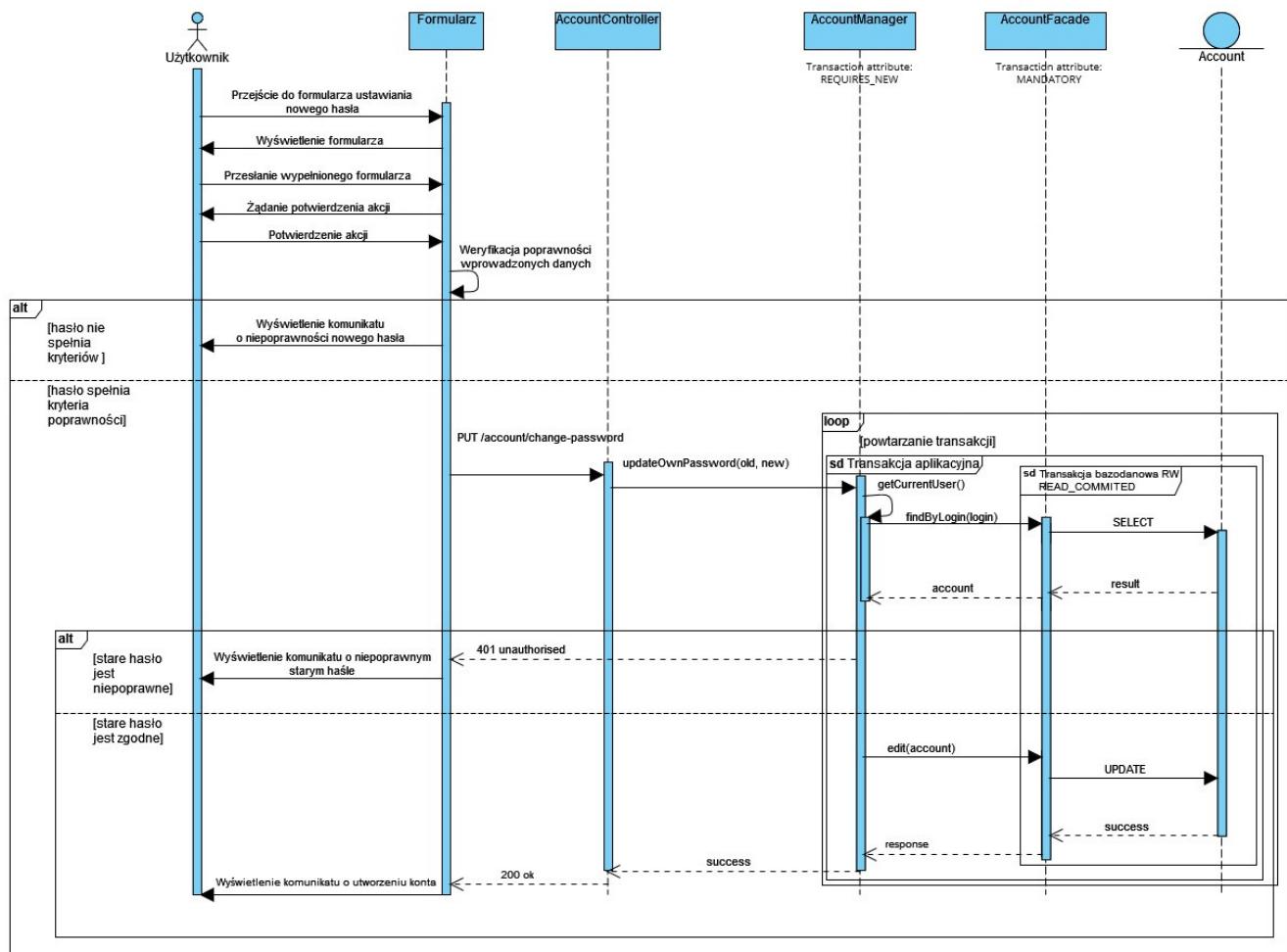
7.4 Diagram sekwencji dla przypadku użycia MOK 4 Odblokuj konto.



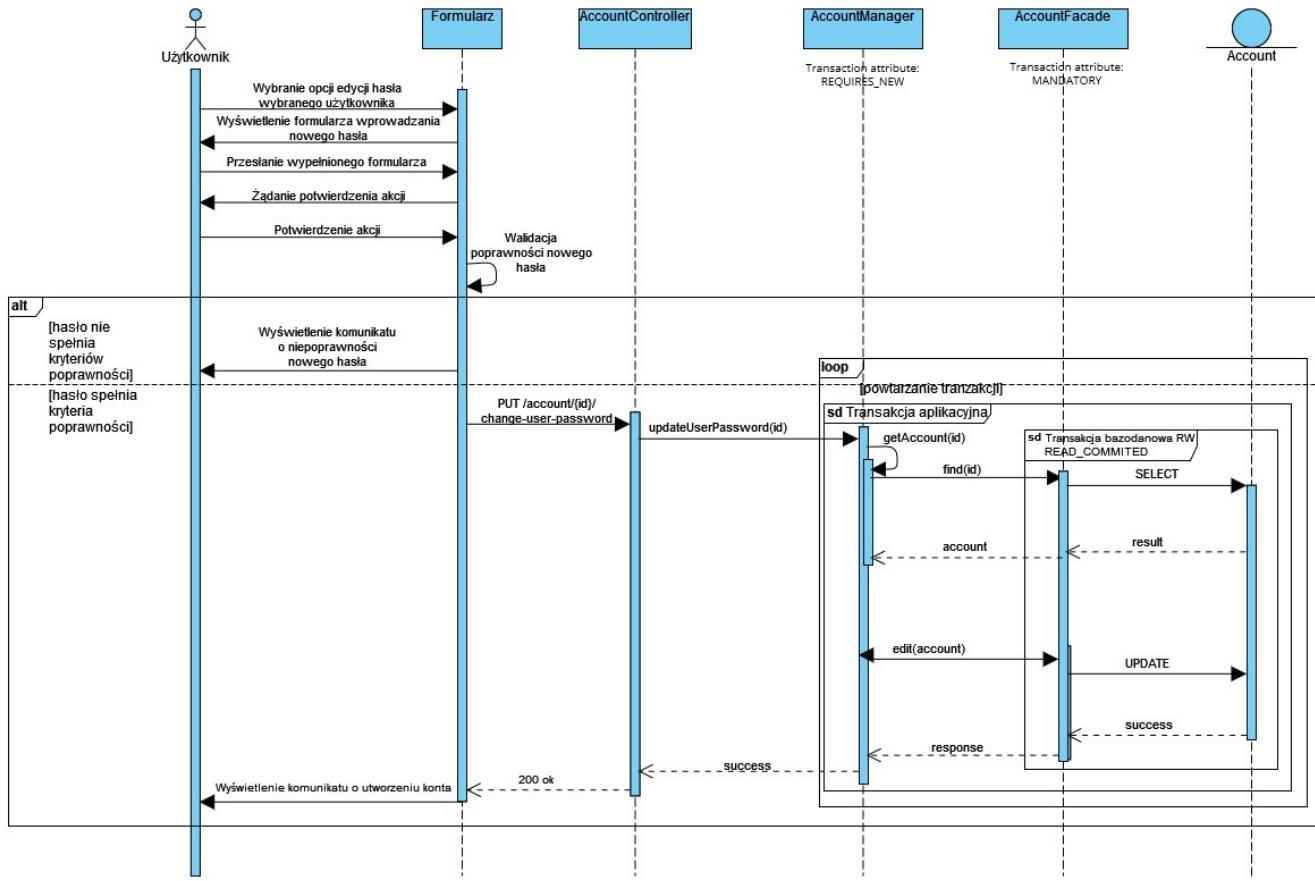
7.5 Diagram sekwencji dla przypadku użycia MOK 5 Dołącz poziom dostępu do konta.



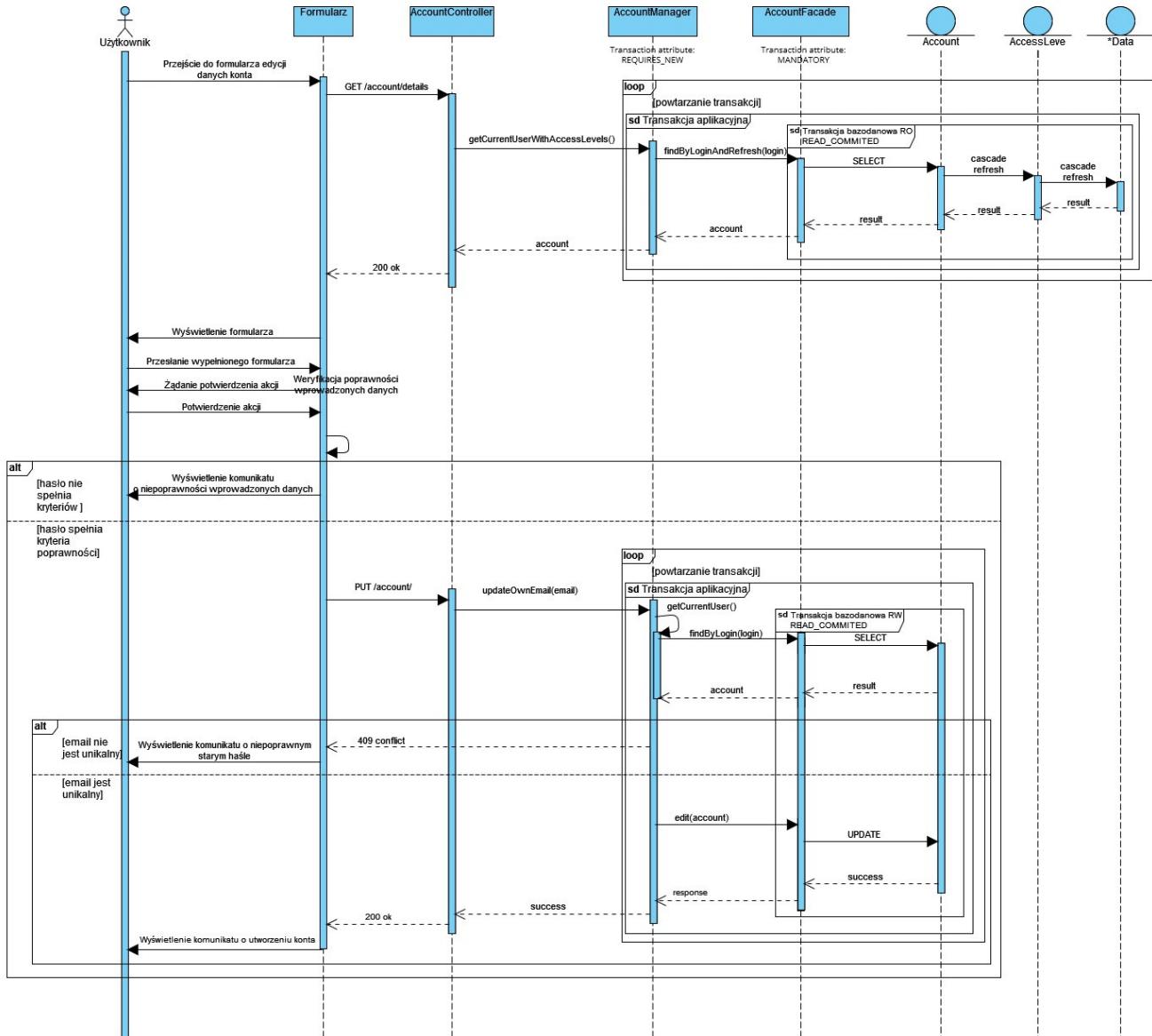
7.6 Diagram sekwencji dla przypadku użycia MOK 6 Odłącz poziom dostępu od konta.



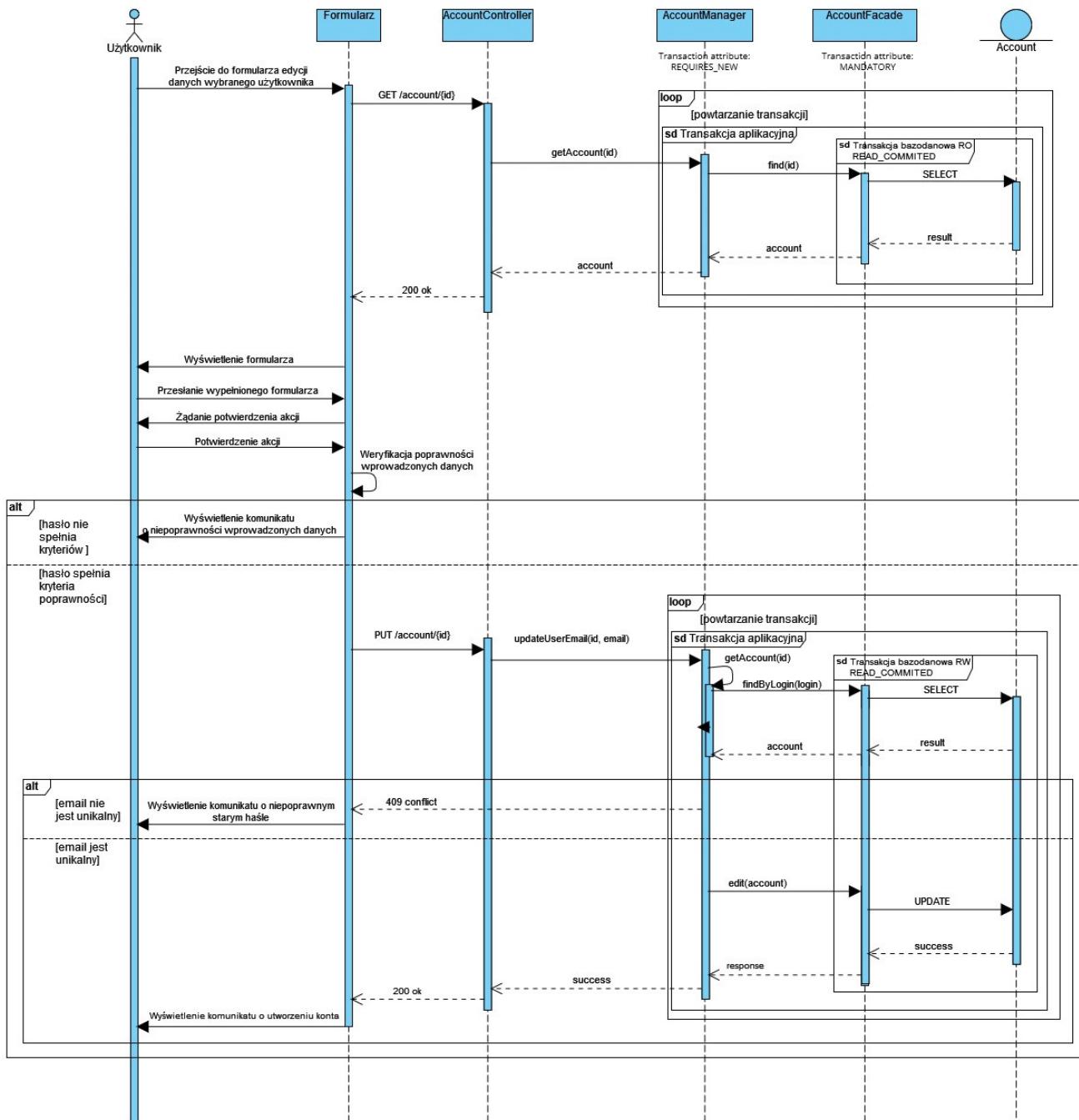
7.7 Diagram sekwencji dla przypadku użycia MOK 7 Zmień własne hasło.



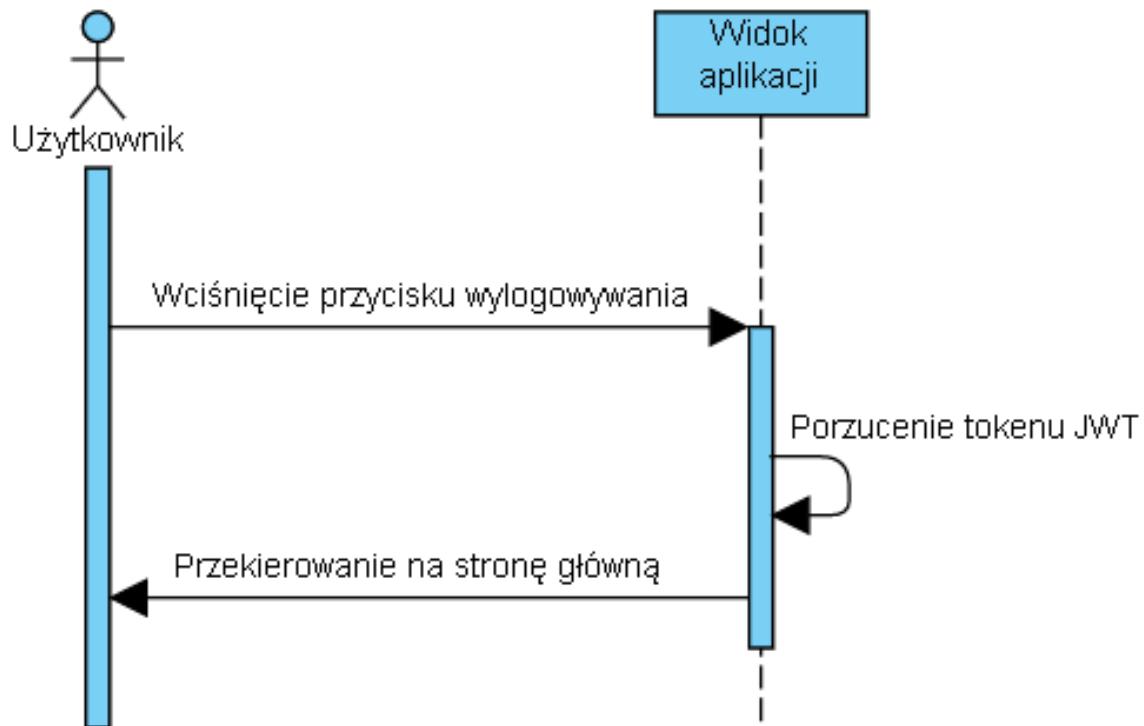
7.8 Diagram sekwencji dla przypadku użycia MOK 8 Zmień hasło innego użytkownika.



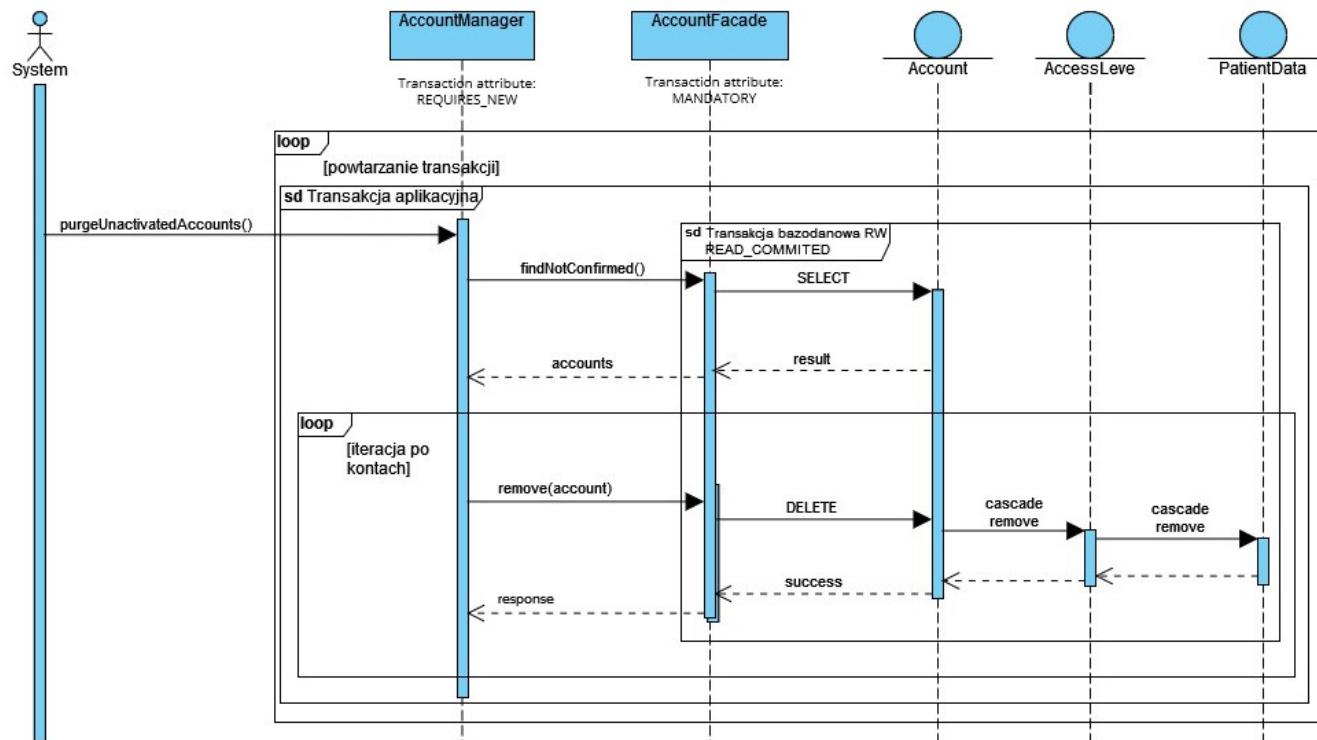
7.9 Diagram sekwencji dla przypadku użycia MOK 9 Edytuj dane własnego konta.



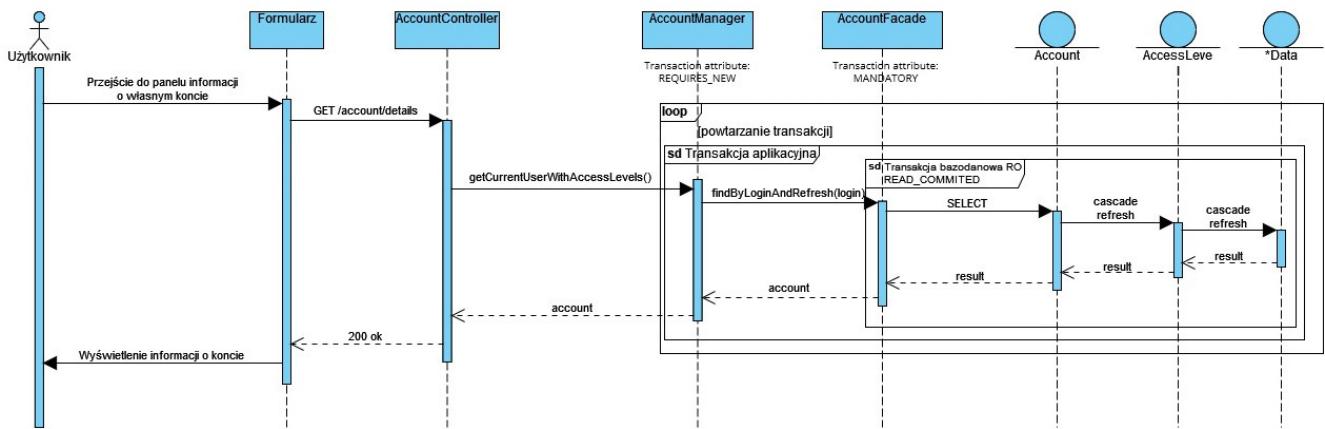
7.10 Diagram sekwencji dla przypadku użycia MOK 10 Edytuj dane konta innego użytkownika.



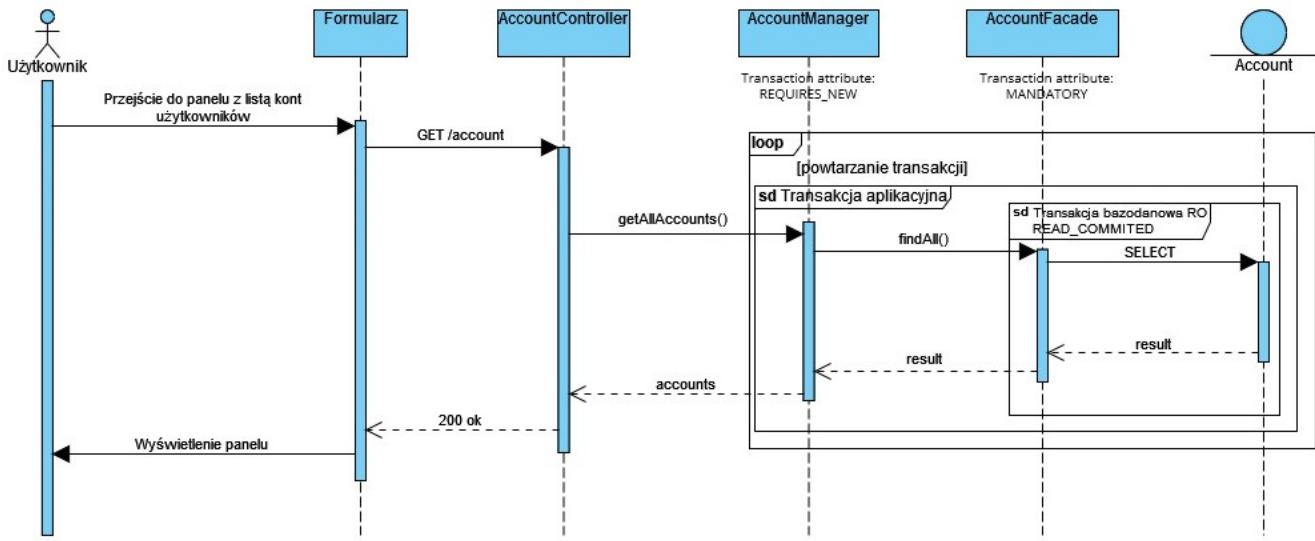
7.11 Diagram sekwencji dla przypadku użycia: MOK 11 Wyloguj



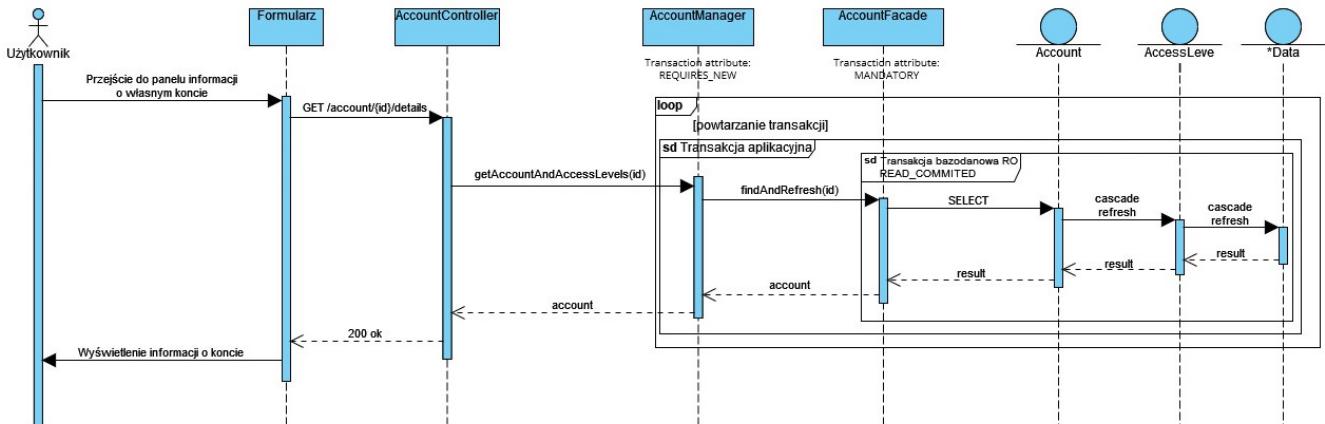
7.12 Diagram sekwencji dla przypadku użycia MOK 12 Usuń konto w przypadku braku aktywacji.



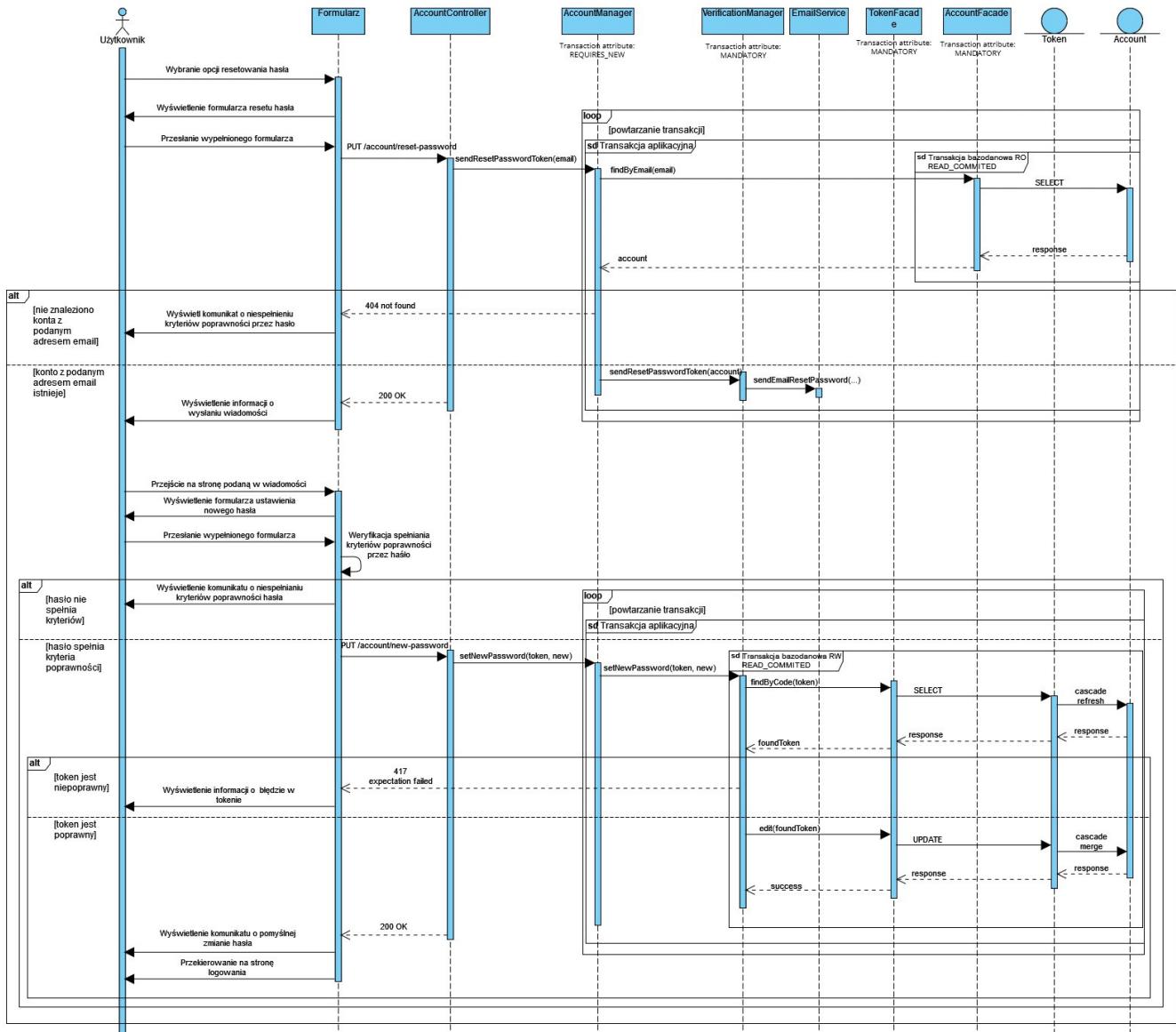
7.13 Diagram sekwencji dla przypadku użycia MOK 13 Wyświetl informacje o koncie.



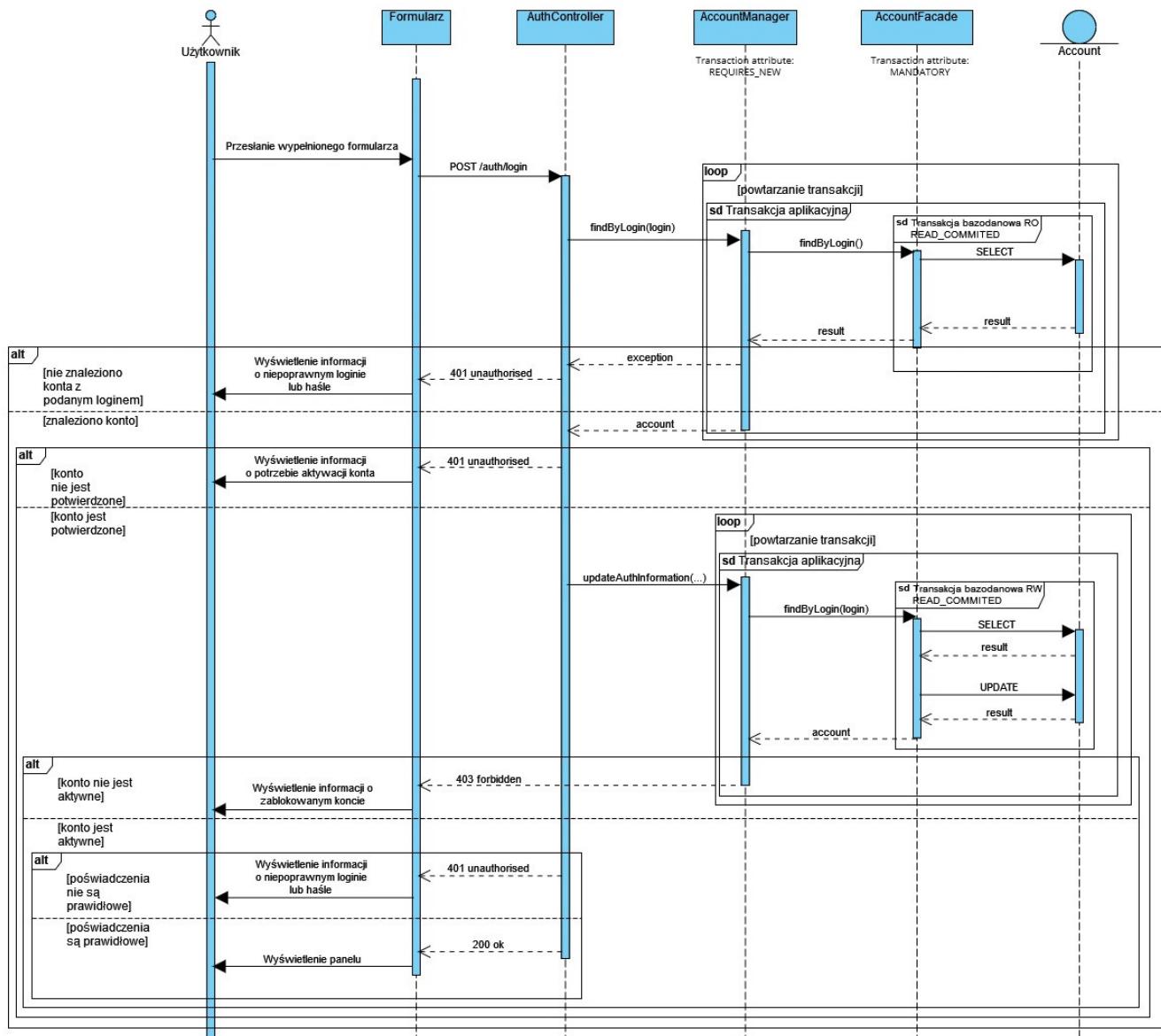
7.14 Diagram sekwencji dla przypadku użycia MOK 14 Wyświetl listę kont.



7.15 Diagram sekwencji dla przypadku użycia MOK 15 Wyświetl informacje o koncie innego użytkownika.



7.16 Diagram sekwencji dla przypadku użycia MOK 16 Zresetuj hasło.



7.17 Diagram sekwencji dla przypadku użycia MOK 17 Zaloguj się na konto.

8 Konfiguracja uwierzytelniania w aplikacji 1 pkt

API będzie chronione tokenem JWT. będzie on tworzony po poprawnym uwierzytelnieniu i będzie miał okres ważności równy 30 minut.

- TokenJWT składa się z trzech sekcji:
 - nagłówka (header), ładunku(payload) i podpisu (signature)
 - Nagłówek zawiera typ tokenu (JWT) oraz algorytm szyfrowania (HS256)
 - Ładunek zawiera:
 - identyfikator użytkownika
 - wydawca tokenu
 - czas wygaśnięcia tokenu
 - czas wydania tokenu
 - odbiorcę tokenu
 - Podpis to sekcja, w której znajduje się zaszyfrowany nagłówek i ładunek.

rejestracja kont

- Admin: konto jest tworzone przez istniejącego Admina poprzez podanie loginu i hasła.
- Farmaceuta: konto jest tworzone z poziomu panelu administratora poprzez podanie loginu hasła i numeru licencji.
- Pacjent: konto jest tworzone przez gościa poprzez podanie danych użytkownika, loginu, hasła i adresu e-mail.

Rejestracja konta skutkuje utworzeniem krotki w tabeli account , oraz powiązaną z nią encję odpowiedniego poziomu dostępu (krotka w tabeli access_level i jednej z: patient_data , admin_data i chemist_data. w przypadku kont tworzonych przez administratora wartość flagi confirmed jest od razu ustawiona na true , w przeciwnym razie jest to false

weryfikacji konta po rejestracji samodzielnej

Na podany przez użytkownika adres e-mail wysyłany jest link aktywacyjny, skorzystanie z niego ustawia flagę confirmed na true . Można zalogować się jedynie na konto którego flaga confirmed jest ustawiona na true .

usuwanie nieaktywnych kont po upływie limitu czasu na aktywację zarejestrowanego konta.

Jeżeli pacjent nie skorzysta z linku w przeciągu 24 godzin jego konto wraz z przypisanymi do niego poziomami dostępu zostaną usunięte. (automatyczne usuwane są konta tworzone wcześniej niż 24 godziny temu (wartość pola creationDate) z flagą confirmed równą false)

blokowania i odblokowania konta

Administrator może aktywować lub dezaktywować konta wszystkich użytkowników korzystających (zmieniana jest flaga active w tabeli account). Na konto z flagą active równą false nie można się zalogować.

przydzielania i odbierania poziomów dostępu

Administrator może nadawać i odbierać poziomy dostępu kontom użytkowników (krotki w tabeli access_level).

Użytkownicy mogą aktywować lub dezaktywować przypisane do swojego konta poziomy dostępu (flaga active w tabeli access_level)

Sprawozdanie szczegółowe (25 pkt)

Sprawozdanie szczegółowe

9 Diagram UML klas komponentów EJB/CDI 4 pkt

Poprawny diagram UML komponentów EJB/CDI (klas i interfejsów), oddzielny dla każdego modułu funkcjonalnego z wykazaną przynależnością klas do pakietów.

Diagram musi eksponować:

- zależności występujące pomiędzy poszczególnymi klasami i interfejsami
- typ komponentu
- wyjątki weryfikowalne deklarowane przez poszczególne metody

Diagram MOK

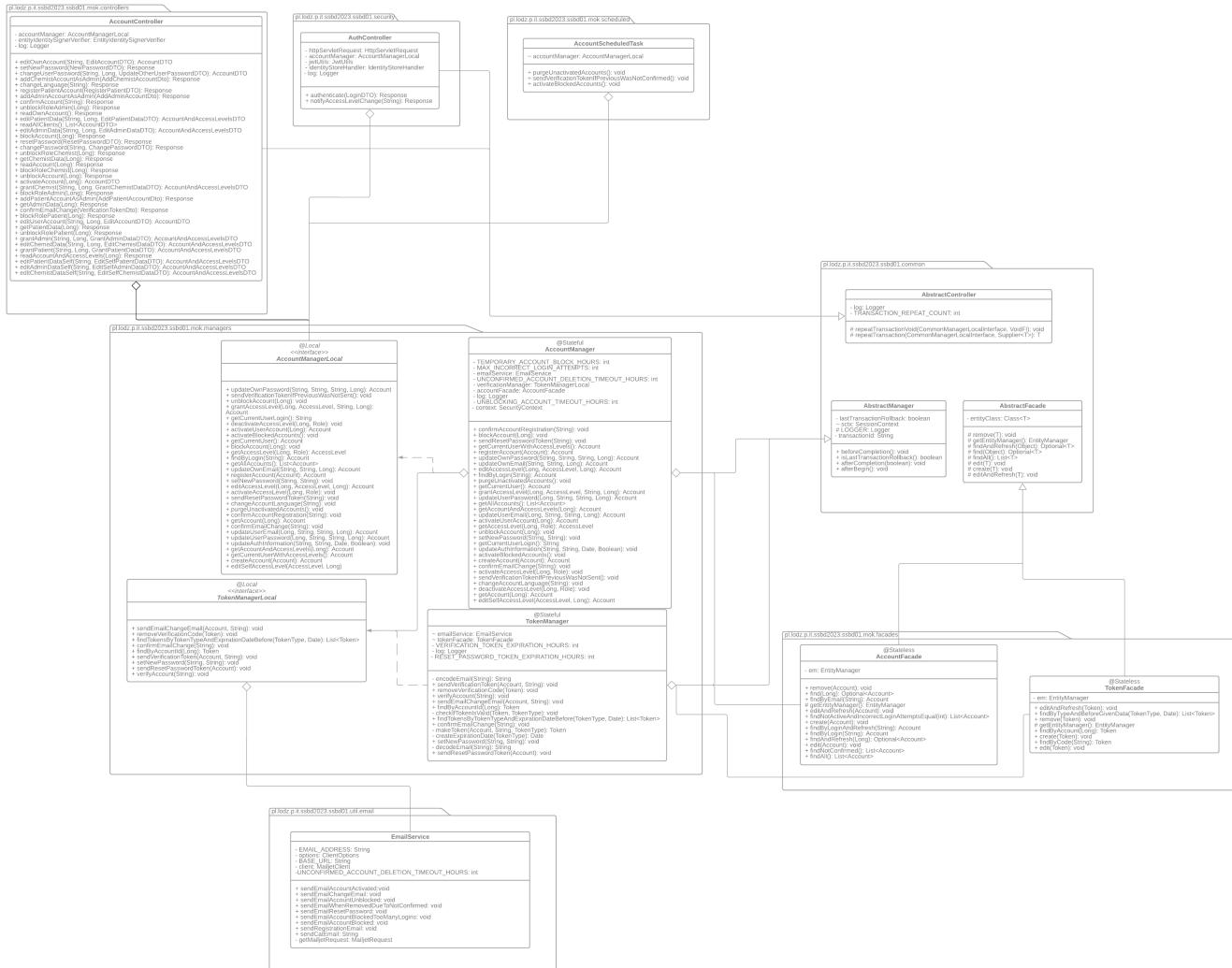
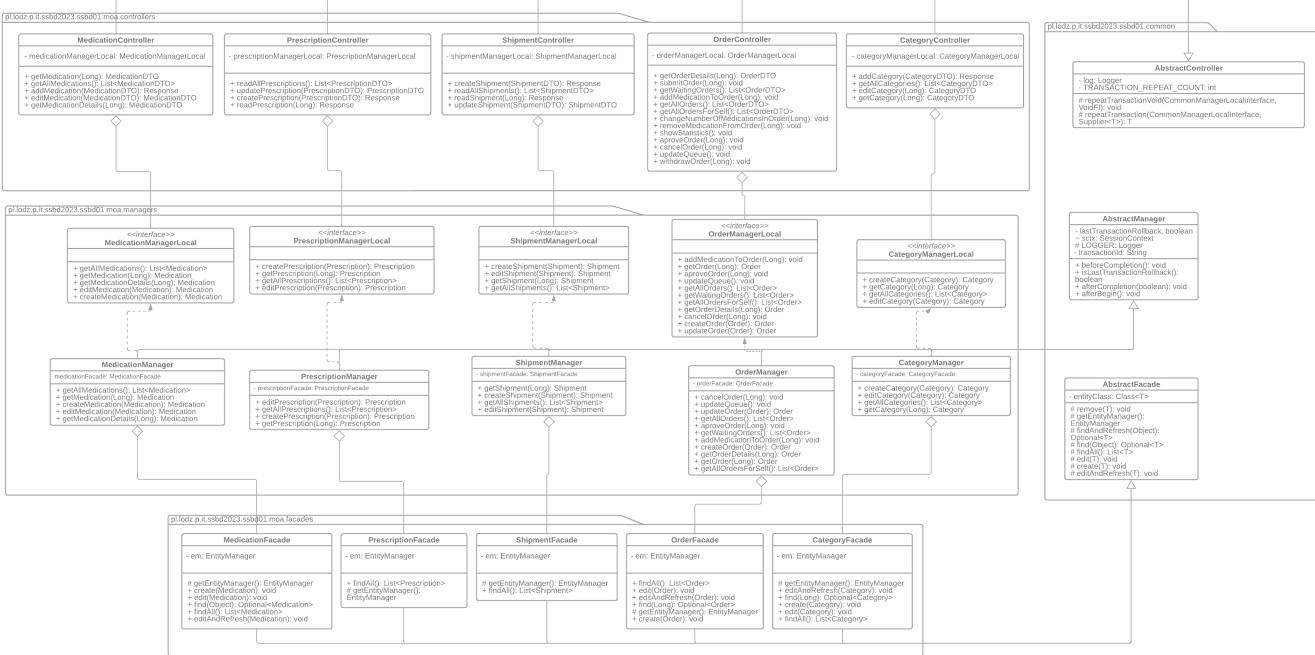
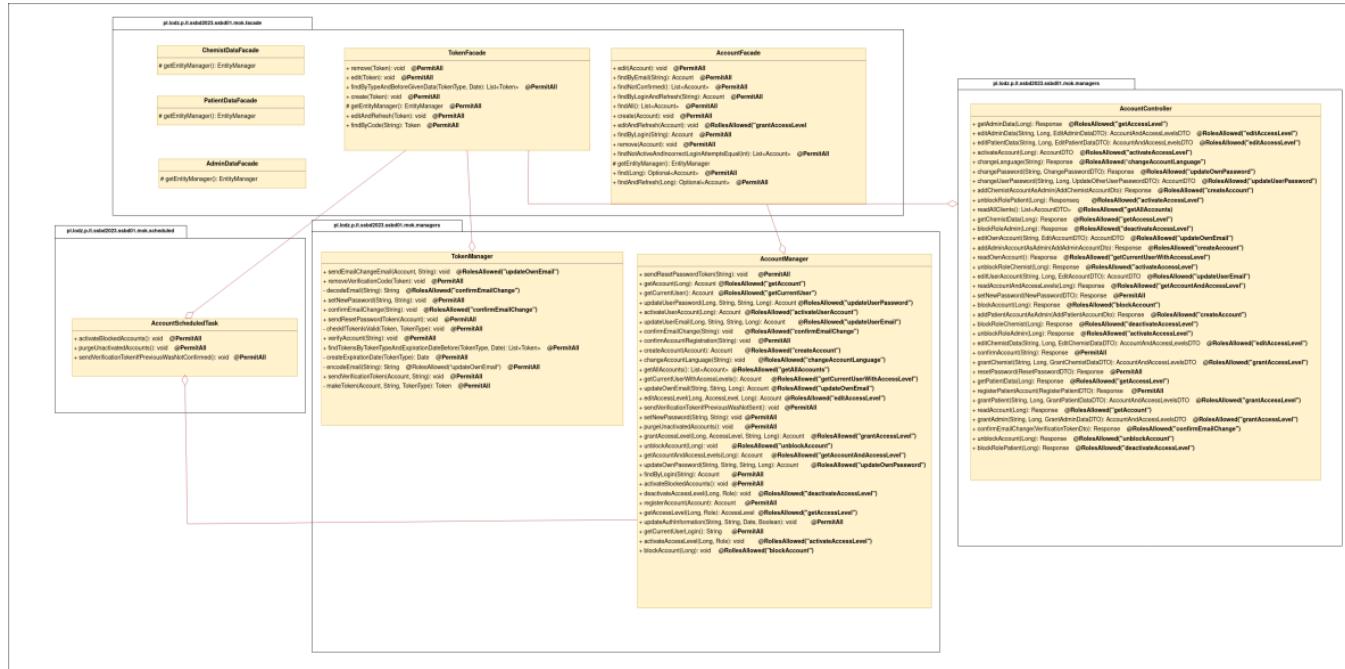


Diagram MOA



10 Model bezpieczeństwa komponentów EJB/CDI 5 pkt

Schemat bezpieczeństwa komponentów EJB dla MOK



Schemat bezpieczeństwa komponentów EJB dla MOA

Przypadek użycia	Na tym etapie metody kontrolerów nie są wykorzystywane z tego powodu zabraniamy do nich dostępu	Na tym etapie metody managerów nie są wykorzystywane z tego powodu zabraniamy do nich dostępu	Na tym etapie metody fasad nie są jeszcze w pełni zaimplementowane
MOA.1	<pre>@RolesAllowed("getAllMedications") public List<MedicationDTO> getAllMedications()</pre>	<pre>@RolesAllowed("getAllMedications") public List<Medication> getAllMedications()</pre>	<pre>@RolesAllowed("getAllMedications")) public List<Medication> findAll()</pre>
MOA.2	<pre>@RolesAllowed("getMedicationDetails") public MedicationDTO getMedicationDetails(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("getMedicationDetails") public Medication getMedicationDetails(Long id) @PermitAll public Medication getMedication(Long id)</pre>	<pre>@PermitAll public Optional<Medication> findAndRefresh(Object id)</pre>
MOA.3	Implementacja na froncie	Implementacja na froncie	Implementacja na froncie
MOA.4	Implementacja na froncie	Implementacja na froncie	Implementacja na froncie
MOA.5	Implementacja na froncie	Implementacja na froncie	Implementacja na froncie
MOA.6	Implementacja na froncie	Implementacja na froncie	Implementacja na froncie

MOA.7	<pre>@RolesAllowed("createOrder") public Response submitOrder(@Valid CreateOrderDTO createOrderDTO)</pre>	<pre>@RolesAllowed("createOrder") public void createOrder(Order order, EtagVerification etagVerification) @RolesAllowed("createOrder") private boolean checkAllMedicationsAvailable(Order order) @RolesAllowed("createOrder") private boolean checkIsOnPrescription(Order order) @RolesAllowed("createOrder") private void decreaseMedicationStock(Order order, EtagVerification etagVerification)</pre>	<pre>@RolesAllowed("createOrder") public void create(Order order) @PermitAll public Medication findByName(String name)</pre>
MOA.8	<pre>@RolesAllowed("withdraw") public Response withdrawOrderById(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("withdraw") public void withdrawOrder(Long id)</pre>	<pre>@RolesAllowed("withdraw") public void withdrawOrder(Long id, Long userId)</pre>
MOA.9	<pre>@RolesAllowed("getWaitingOrders") public List<OrderDTO> getWaitingOrders()</pre>	<pre>@RolesAllowed("getWaitingOrders") public List<Order> getWaitingOrders()</pre>	<pre>@RolesAllowed("getWaitingOrders") public List<Order> findWaitingOrders()</pre>
MOA.10	<pre>@RolesAllowed("deleteWaitingOrdersById") public Response deleteWaitingOrderById(@PathParam("id") Long id) @RolesAllowed("approvedByPatient") public Response approvedByPatient(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("deleteWaitingOrdersById") public void deleteWaitingOrderById(Long id) @RolesAllowed("approvedByPatient") public void approvedByPatient(Long id) @RolesAllowed("approvedByPatient") private void decreaseMedicationStock(Order order)</pre>	<pre>@PermitAll public Optional<Order> find(Long id) @RolesAllowed("deleteWaitingOrdersById") public void deleteWaitingOrdersById(Long id)</pre>
MOA.11	<pre>@RolesAllowed("createShipment") public Response createShipment(@Valid CreateShipmentDTO shipmentDTO)</pre>	<pre>@RolesAllowed("createShipment") public void createShipment(Shipment shipment, EtagVerification etagVerification)</pre>	<pre>@RolesAllowed("createShipment") public void create(Shipment shipment)</pre>
MOA.12	<pre>@RolesAllowed("getOrdersToApprove") public List<OrderDTO> getOrdersToApprove()</pre>	<pre>@RolesAllowed("getOrdersToApprove") public List<Order> getOrdersToApprove()</pre>	<pre>@RolesAllowed("getOrdersToApprove") public List<Order> findNotYetApproved()</pre>
MOA.13	<pre>@RolesAllowed("approveOrder") public Response approveOrder(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("approveOrder") public void approveOrder(Long id)</pre>	<pre>@PermitAll public Optional<Order> find(Long id) @PermitAll public void edit(Order order)</pre>
MOA.14	<pre>@RolesAllowed("cancelOrder") public Response cancelOrder(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("cancelOrder") public void cancelOrder(Long id)</pre>	<pre>@RolesAllowed("cancelOrder") public void cancelOrder(Long id, Long chemistId)</pre>
MOA.15	Brak Implementacji	Brak Implementacji	Brak Implementacji
MOA.16	<pre>@RolesAllowed("updateQueue") public Response updateQueue()</pre>	<pre>@RolesAllowed("updateQueue") public void updateQueue()</pre>	<pre>@RolesAllowed("updateQueue") public List<Order> findAllOrdersInQueueSortByOrderDate() @PermitAll public List<Shipment> findAllNotAlreadyProcessed()</pre>
MOA.17	<pre>@DenyAll public List<OrderDTO> getAllOrdersForSelf()</pre>	<pre>@DenyAll public List<Order> getAllOrdersForSelf()</pre>	

MOA.18	<pre>@RolesAllowed("getAllOrdersForSelf") public Response getAllOrdersForSelf()</pre>	<pre>@RolesAllowed("getAllOrdersForSelf") public List<Order> getAllOrdersForSelf(Account account)</pre>	<pre>@PermitAll public List<Order> findAllByPatientId(Long id)</pre>
MOA.19	<pre>PermitAll public Response addMedication(@Valid AddMedicationDTO addMedicationDTO)</pre>	<pre>@RolesAllowed("createMedication") public Medication createMedication(Medication medication, String categoryName)</pre>	<pre>@RolesAllowed("createMedication") public void create(Medication medication) @RolesAllowed("createCategory") public void create(Category category)</pre>
MOA.20	Brak Implementacji	Brak Implementacji	Brak Implementacji
MOA.21	<pre>@RolesAllowed("createCategory") public Response addCategory(@NotNull @Valid CategoryDTO categoryDto)</pre>	<pre>@RolesAllowed("createCategory") public Category createCategory(Category category)</pre>	<pre>@PermitAll public Category findByName(String name) @RolesAllowed("createCategory") public void create(Category category)</pre>
MOA.22	<pre>@RolesAllowed("getAllCategories") public List<GetCategoryDTO> getAllCategories() @PermitAll public Response getCategory(@PathParam("id") Long id)</pre>	<pre>@RolesAllowed("getAllCategories") public List<Category> getAllCategories() @PermitAll public Category getCategory(Long id)</pre>	<pre>@PermitAll public Optional<Category> find(Long id) @PermitAll public List<Category> findAll()</pre>
MOA.23	<pre>@RolesAllowed("editCategory") public CategoryDTO editCategory(@HeaderParam("If-Match") @NotEmpty String etag, @PathParam("id") Long id, @Valid EditCategoryDTO editCategoryDTO)</pre>	<pre>@RolesAllowed("editCategory") public Category editCategory(Long id, Category category, Long version)</pre>	<pre>@PermitAll public Optional<Category> find(Long id) @PermitAll public void edit(Category category)</pre>

Fragment deskryptora wdrożenia aplikacji web.xml zawierający definicje ról
<pre>(...) <security-role> <role-name>ADMIN</role-name> </security-role> <security-role> <role-name>CHEMIST</role-name> </security-role> <security-role> <role-name>PATIENT</role-name> </security-role> <security-role> <role-name>ANONYMOUS</role-name> </security-role> (...) <security-role> <role-name>getAllAccounts</role-name> </security-role> <security-role> <role-name>updateQueue</role-name> </security-role> <security-role> <role-name>getCurrentUser</role-name> </security-role> <security-role> <role-name>getCurrentUserWithAccessLevels</role-</pre>

```

    name>
</security-role>
<security-role>
    <role-name>changeAccountLanguage</role-name>
</security-role>
<security-role>
    <role-name>getAccountAndAccessLevels</role-name>
</security-role>
<security-role>
    <role-name>grantAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>getAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>deactivateAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>getAccount</role-name>
</security-role>
<security-role>
    <role-name>createAccount</role-name>
</security-role>
<security-role>
    <role-name>editAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>activateUserAccount</role-name>
</security-role>
<security-role>
    <role-name>blockAccount</role-name>
</security-role>
<security-role>
    <role-name>updateUserPassword</role-name>
</security-role>
<security-role>
    <role-name>updateOwnPassword</role-name>
</security-role>
<security-role>
    <role-name>updateOwnEmail</role-name>
</security-role>
<security-role>
    <role-name>updateUserEmail</role-name>
</security-role>
<security-role>
    <role-name>confirmEmailChange</role-name>
</security-role>
<security-role>
    <role-name>activateAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>unblockAccount</role-name>
</security-role>
<security-role>
    <role-name>notifyAccessLevelChange</role-name>
</security-role>
<security-role>
    <role-name>editSelfAccessLevel</role-name>
</security-role>
<security-role>
    <role-name>editSelfAccessLevelPatient</role-
name>
</security-role>
<security-role>
    <role-name>editSelfAccessLevelChemist</role-
name>
</security-role>
<security-role>
    <role-name>editSelfAccessLevelAdmin</role-name>
</security-role>
<security-role>
    <role-name>getAllOrdersForSelf</role-name>
</security-role>
<security-role>
    <role-name>getWaitingOrders</role-name>
</security-role>
<security-role>
    <role-name>getOrdersToApprove</role-name>
</security-role>
<security-role>
    <role-name>createOrder</role-name>
</security-role>
<security-role>
    <role-name>addMedicationToOrder</role-name>

```

```

</security-role>
<security-role>
    <role-name>changeNumberOfMedicationsInOrder</role-name>
</role-name>
</security-role>
<security-role>
    <role-name>createMedication</role-name>
</security-role>
<security-role>
    <role-name>getAllMedications</role-name>
</security-role>
<security-role>
    <role-name>getOrderDetails</role-name>
</security-role>
<security-role>
    <role-name>readAllShipments</role-name>
</security-role>
<security-role>
    <role-name>readShipment</role-name>
</security-role>
<security-role>
    <role-name>createShipment</role-name>
</security-role>
<security-role>
    <role-name>updateShipment</role-name>
</security-role>
<security-role>
    <role-name>deleteWaitingOrdersById</role-name>
</security-role>
<security-role>
    <role-name>createCategory</role-name>
</security-role>
<security-role>
    <role-name>getMedicationDetails</role-name>
</security-role>
<security-role>
    <role-name>withdraw</role-name>
</security-role>
<security-role>
    <role-name>approveOrder</role-name>
</security-role>
<security-role>
    <role-name>getAllCategories</role-name>
</security-role>
<security-role>
    <role-name>approvedByPatient</role-name>
</security-role>
<security-role>
    <role-name>editCategory</role-name>
</security-role>
<security-role>
    <role-name>cancelOrder</role-name>
</security-role>

(...)
```

Deskryptor glassfish_web.xml zawierający mapowanie grup oraz tożsamości użytkowników na role aplikacji

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org//DTD GlassFish
Application Server 3.1 Servlet 3.0//EN"
"http://glassfish.org/dtds/glassfish-web-app_3_0-1.dtd">
<glassfish-web-app error-url="">
    <context-root>/</context-root>

    <security-role-mapping>
        <role-name>getAllAccounts</role-name>
        <group-name>ADMIN</group-name>
    </security-role-mapping>

    <security-role-mapping>
        <role-name>getCurrentUser</role-name>
        <group-name>PATIENT</group-name>
        <group-name>CHEMIST</group-name>
        <group-name>ADMIN</group-name>
    </security-role-mapping>

    <security-role-mapping>
```

```

<role-name>getCurrentUserWithAccessLevels</role-name>
<group-name>PATIENT</group-name>
<group-name>CHEMIST</group-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>changeAccountLanguage</role-name>
<group-name>PATIENT</group-name>
<group-name>CHEMIST</group-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>getAccountAndAccessLevels</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>grantAccessLevel</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>getAccessLevel</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>deactivateAccessLevel</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>activateAccessLevel</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>getAccount</role-name>
<group-name>PATIENT</group-name>
<group-name>CHEMIST</group-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>createAccount</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>editAccessLevel</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>activateUserAccount</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>blockAccount</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>unblockAccount</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>updateUserPassword</role-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>updateOwnPassword</role-name>
<group-name>PATIENT</group-name>
<group-name>CHEMIST</group-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
<role-name>updateOwnEmail</role-name>

```

```

<group-name>PATIENT</group-name>
<group-name>CHEMIST</group-name>
<group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>updateUserEmail</role-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>confirmEmailChange</role-name>
    <group-name>PATIENT</group-name>
    <group-name>CHEMIST</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>notifyAccessLevelChange</role-name>
    <group-name>PATIENT</group-name>
    <group-name>CHEMIST</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>editSelfAccessLevel</role-name>
    <group-name>PATIENT</group-name>
    <group-name>CHEMIST</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>createOrder</role-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>getAllOrdersForSelf</role-name>
    <group-name>PATIENT</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>getOrderDetails</role-name>
    <group-name>PATIENT</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>getWaitingOrders</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>getOrdersToApprove</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>addMedicationToOrder</role-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>changeNumberOfMedicationsInOrder</role-name>
    <group-name>PATIENT</group-name>
    <group-name>ADMIN</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>createMedication</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>getAllMedications</role-name>
    <group-name>CHEMIST</group-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>readAllShipments</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

```

```

<security-role-mapping>
    <role-name>readShipment</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>updateQueue</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>createShipment</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>updateShipment</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>deleteWaitingOrdersById</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>createCategory</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>approveOrder</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>

<security-role-mapping>
    <role-name>getMedicationDetails</role-name>
    <group-name>CHEMIST</group-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>withdraw</role-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>getAllCategories</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>approvedByPatient</role-name>
    <group-name>PATIENT</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>editCategory</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>
<security-role-mapping>
    <role-name>cancelOrder</role-name>
    <group-name>CHEMIST</group-name>
</security-role-mapping>
</glassfish-web-app>

```

Mechanizm tworzenia dziennika zdarzeń w TracingInterceptor

```

@Log
public class TrackerInterceptor {

    @Resource private SessionContext sessionContext;

    @AroundInvoke
    public Object intercept(InvocationContext context) throws Exception {
        StringBuilder message = new StringBuilder("Method: ");
        Object result;
        try {
            message
                .append(context.getMethod().getName())
                .append(" Class: ")
                .append(context.getTarget().getClass().getCanonicalName())
                .append(" user: ")
                .append(sessionContext.getCallerPrincipal().getName()).append(" ");
            log.fine(message.toString());
        } catch (Exception e) {
            message.append("finished with exception: ").append(e);
            log.log(Level.SEVERE, message.toString(), e);
            throw e;
        }
        message.append("finished successfully");
        log.info(message.toString());
        return result;
    }
}

```

Przykład klasy AccountManager korzystającej z Interceptora

```

@Stateful
@TransactionAttribute(TransactionAttribute.Type.REQUIRES_NEW)
@Interceptors({GenericManagerExceptionsInterceptor.class, TrackerInterceptor.class})
@Log
@DenyAll
public class AccountManager extends AbstractManager implements AccountManagerLocal,
SessionSynchronization[...]

```

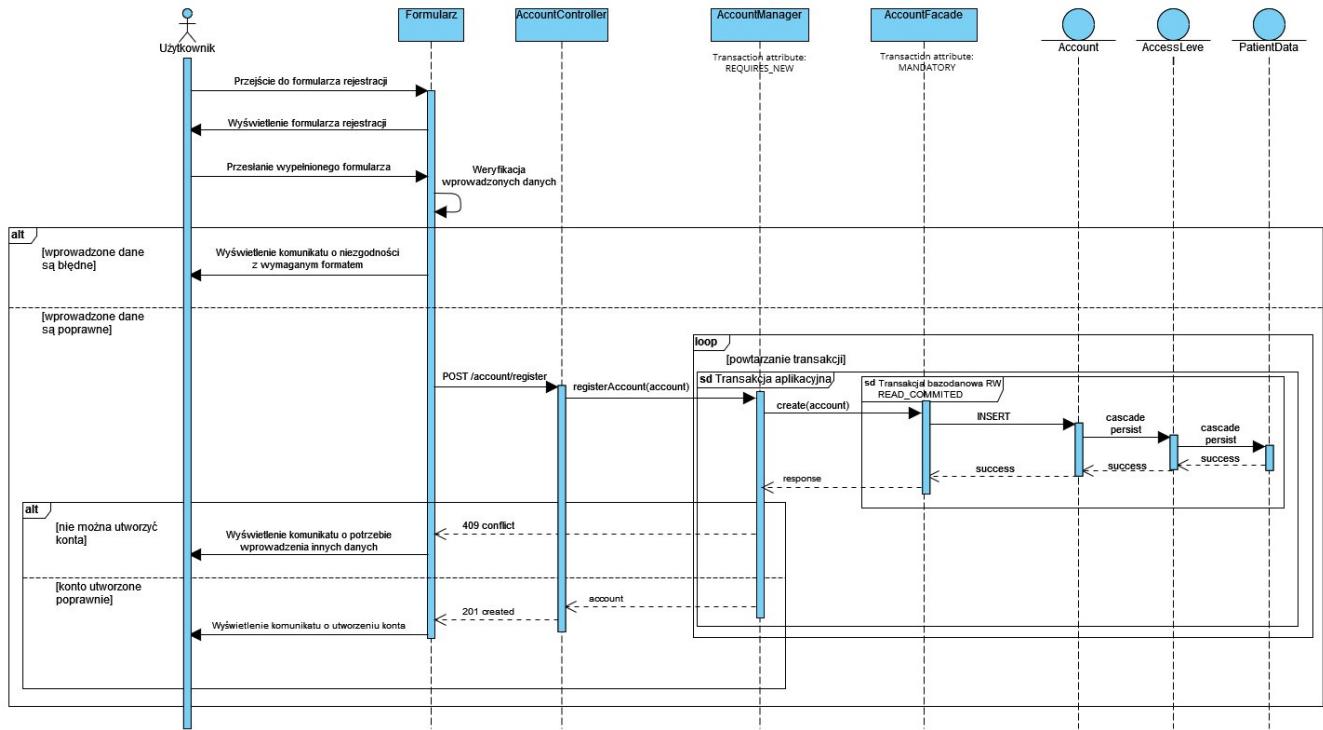
11 Identyfikacja transakcji aplikacyjnych 6 pkt

Transakcje aplikacyjne występujące w aplikacji

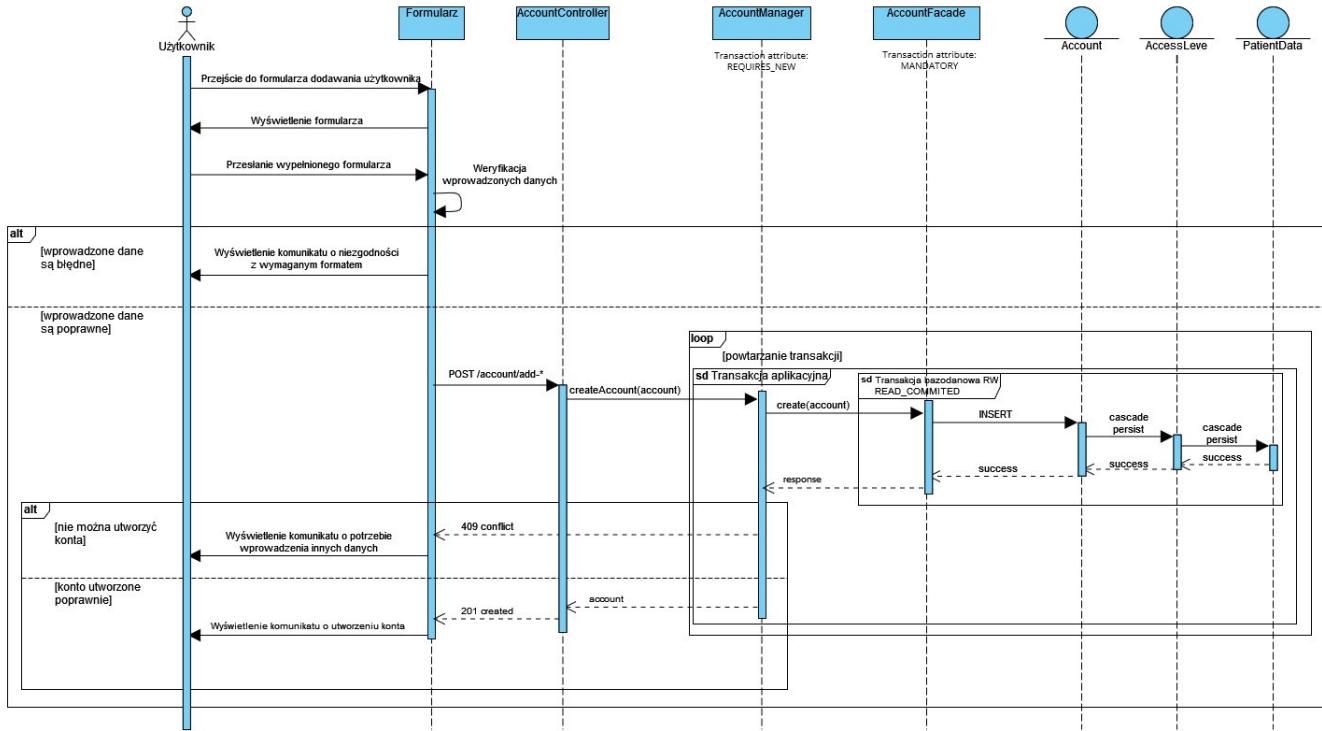
Transakcje aplikacyjne zostały zaznaczone na diagramach sekwencji.

W niektórych miejscach zastosowano uproszoną notację, że to manager zwraca odpowiedź formularzowi. Jest to oczywiście nieprawdą. Ten zapis oznacza, że w danym miejscu w managerze jest rzucany wyjątek, który potem po przechwyceniu z zmapowaniem jest zamieniany na odpowiedź z odpowiednim kodem błędu.

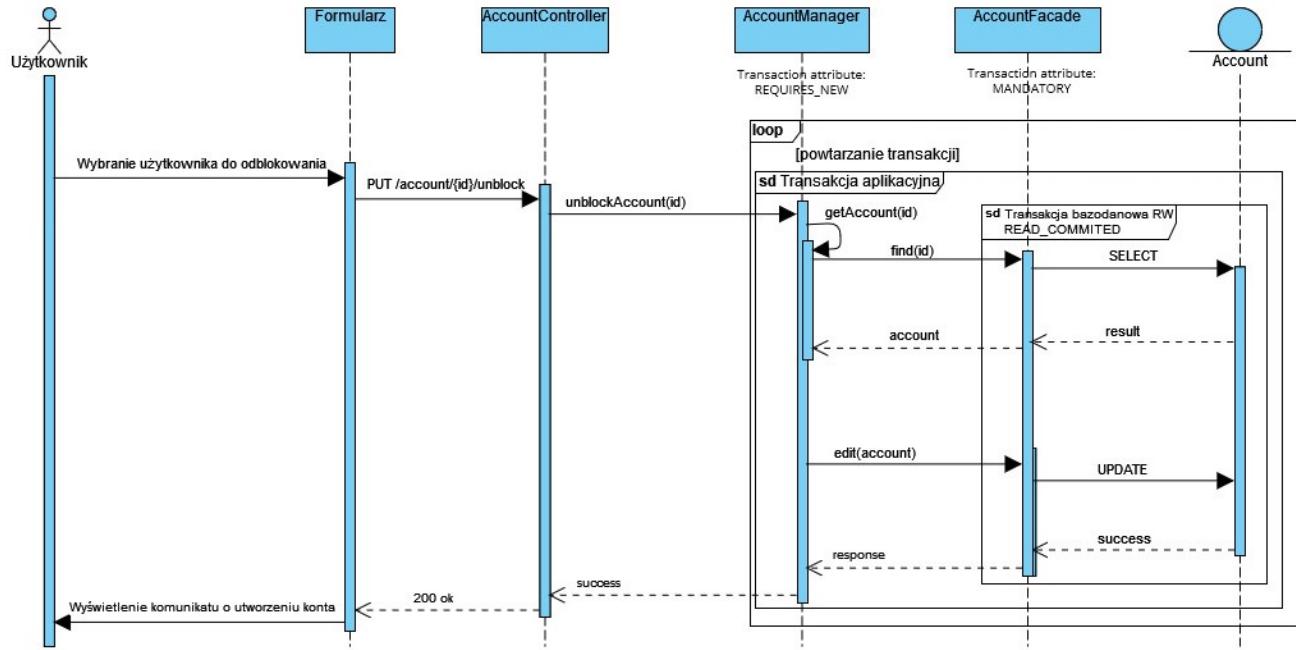
Analogicznie jak w części 7. dokumentacji zapis z * oznacza Patient, Chemist lub Admin.



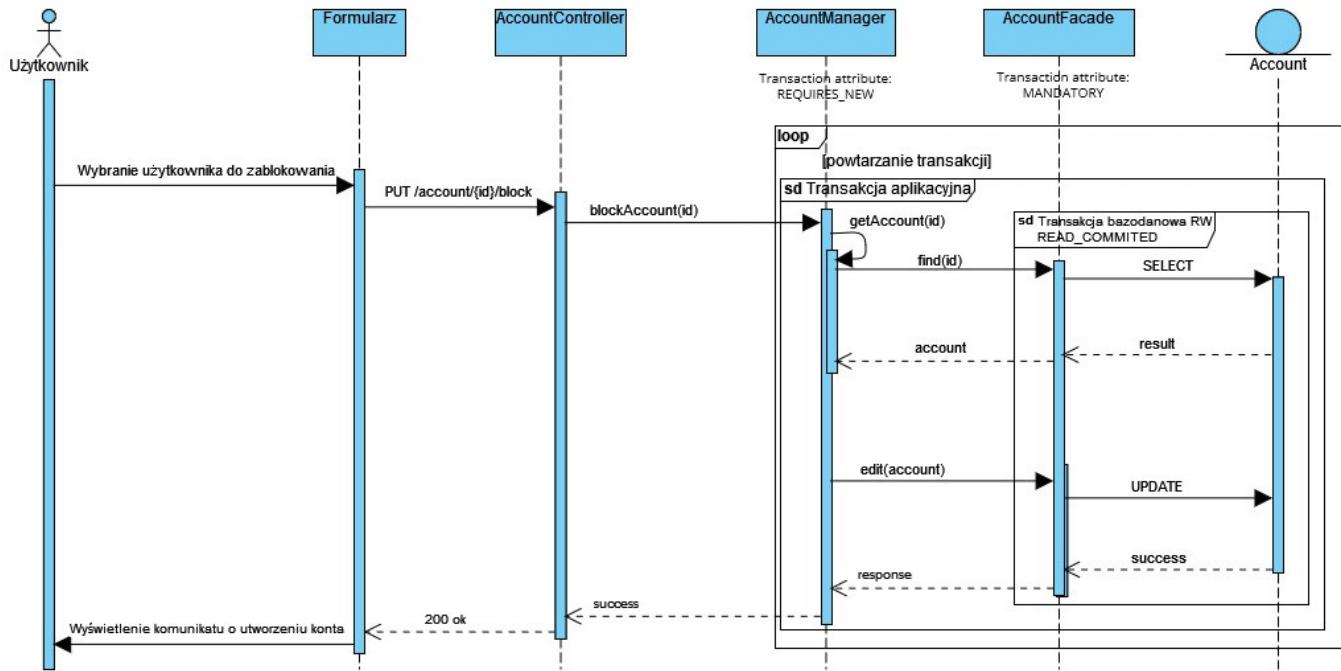
11.1 Diagram sekwencji dla przypadku użycia MOK 1 Zarejestruj.



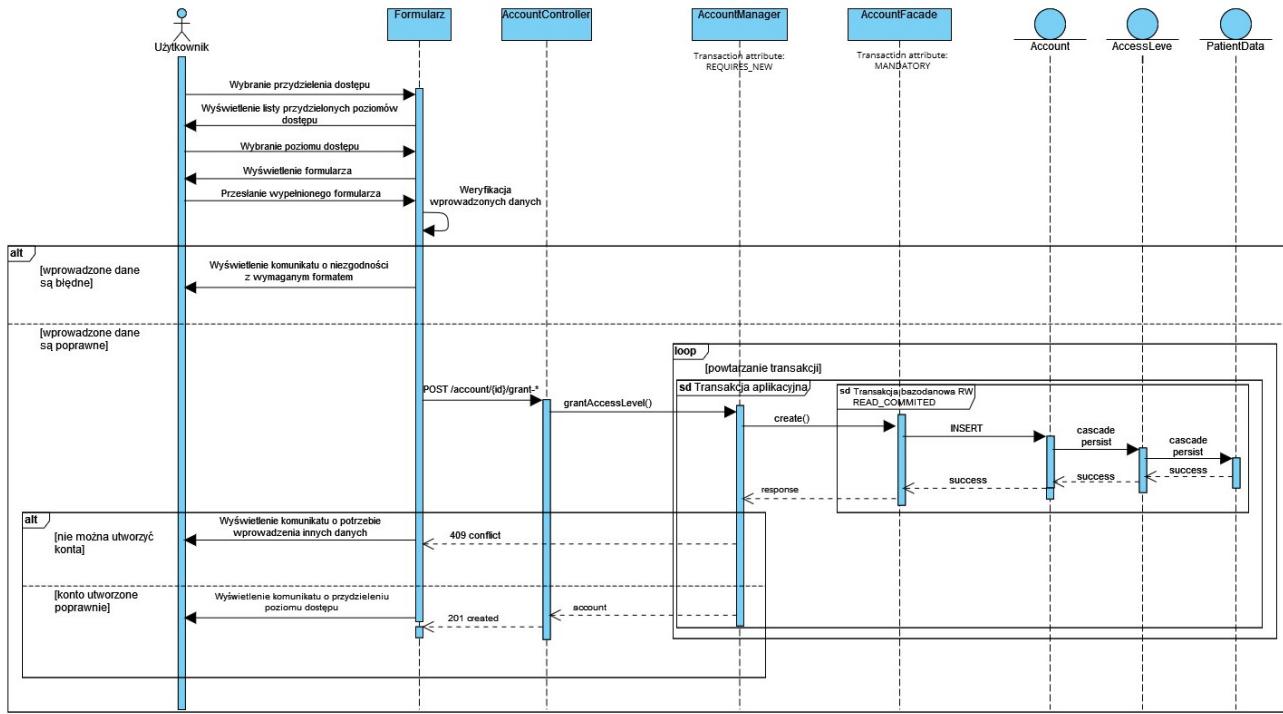
11.2 Diagram sekwencji dla przypadku użycia MOK 2 Utwórz konto.



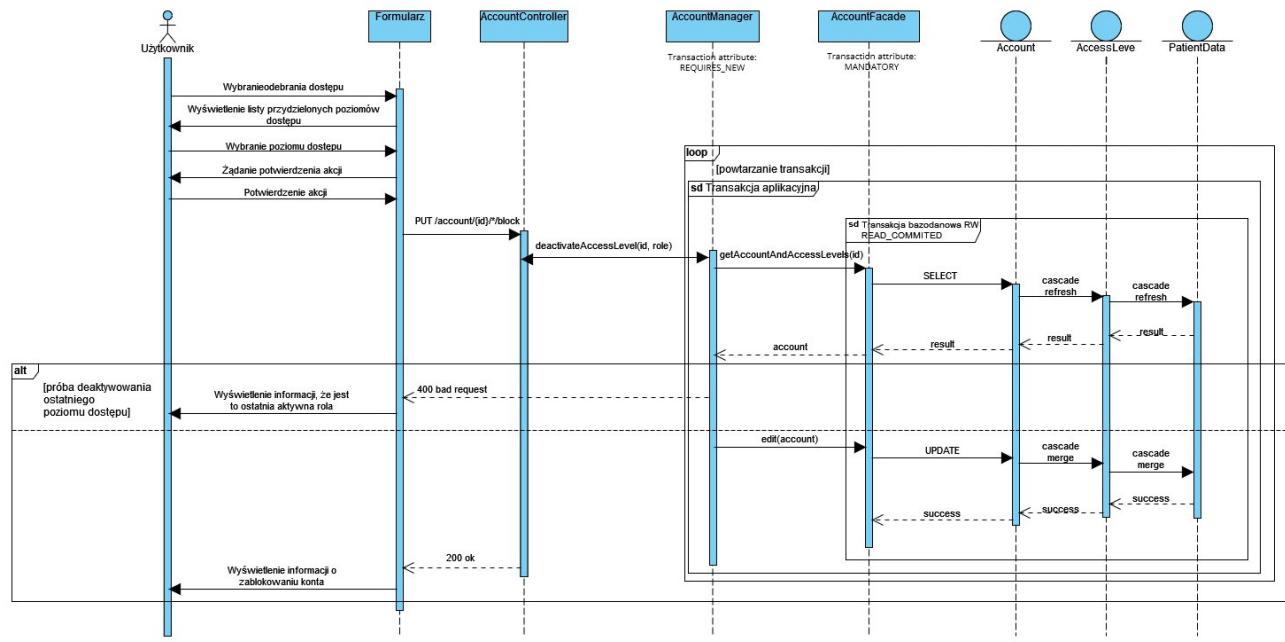
11.3 Diagram sekwencji dla przypadku użycia MOK 3 Zablokuj konto.



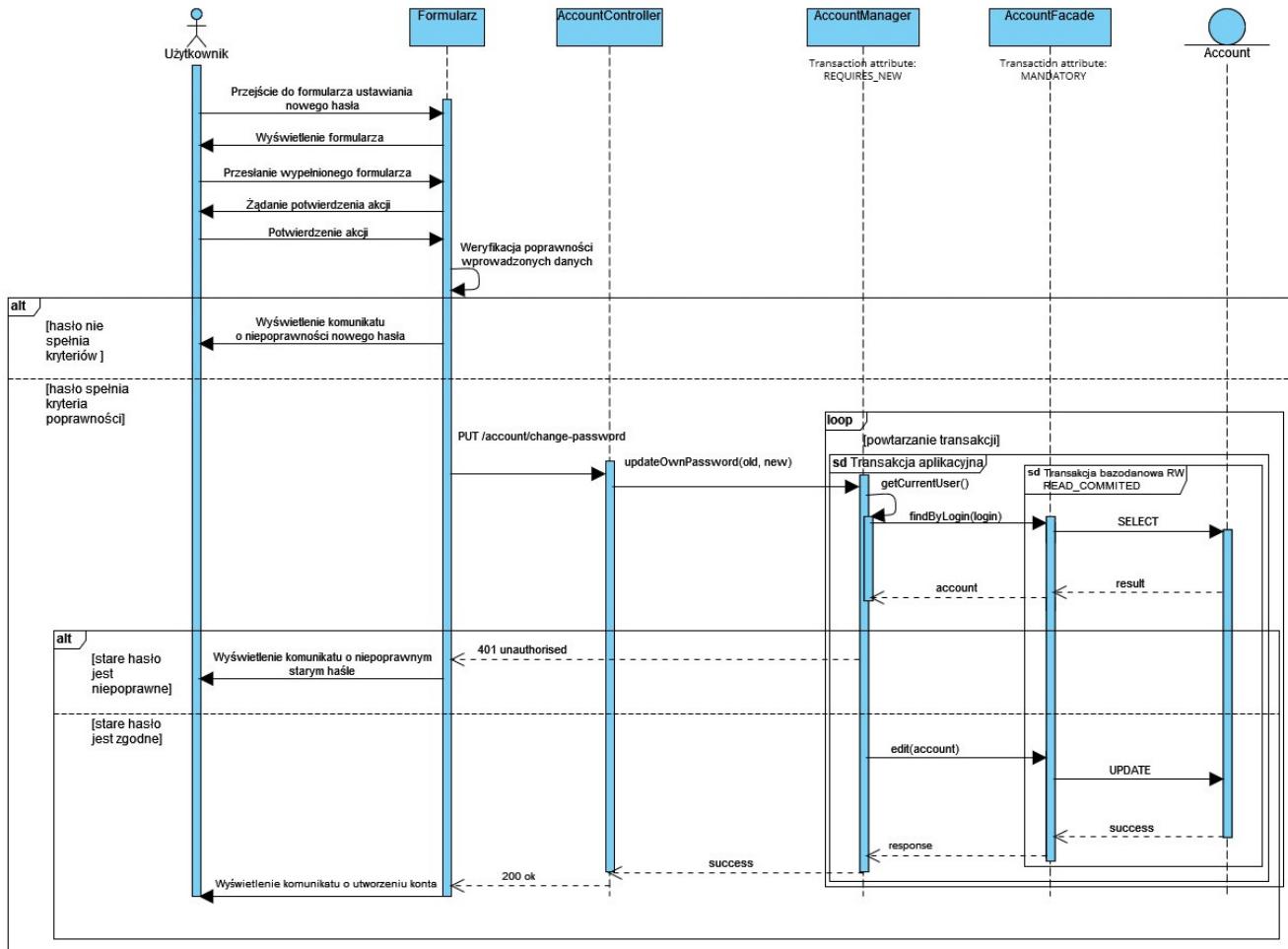
11.4 Diagram sekwencji dla przypadku użycia MOK 4 Odblokuj konto.



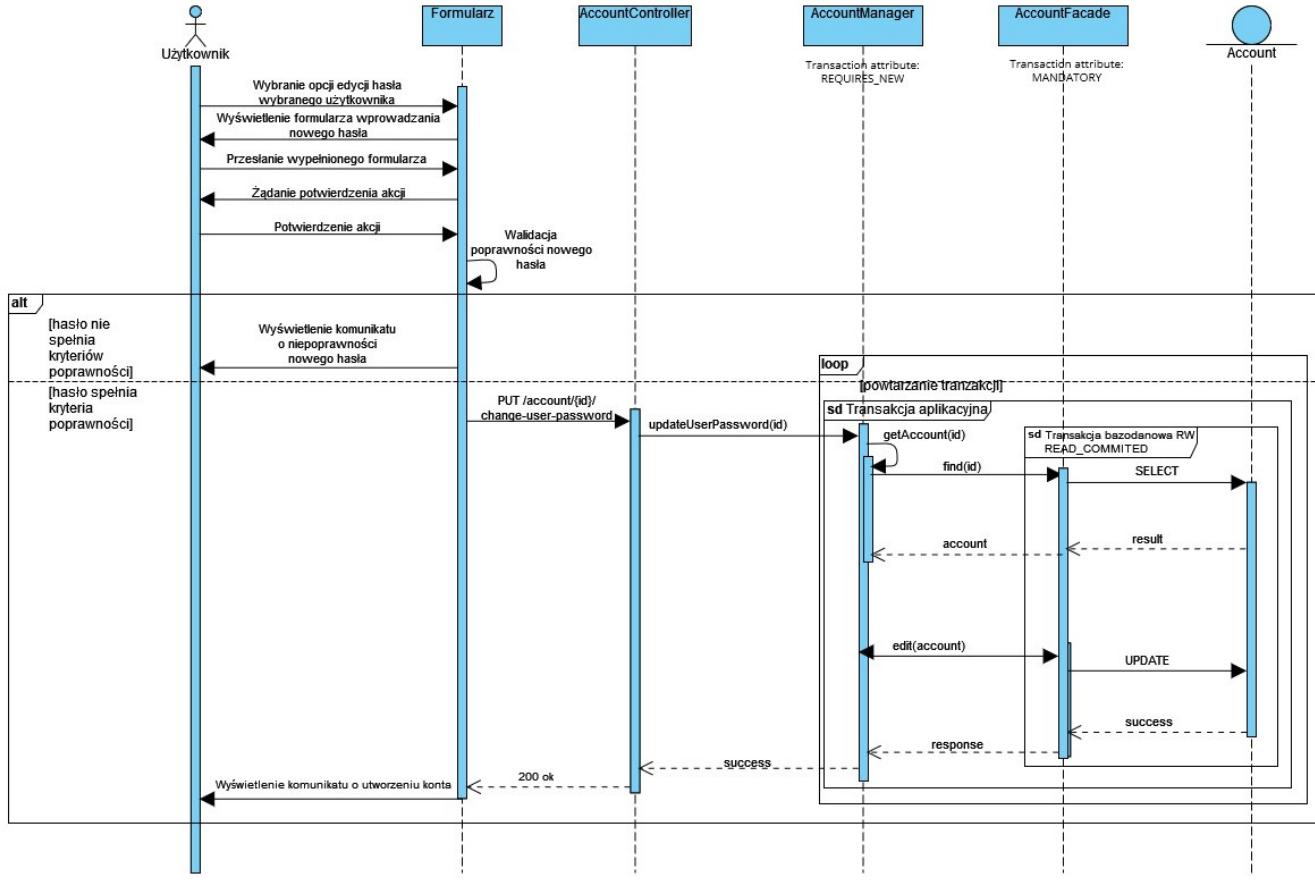
11.5 Diagram sekwencji dla przypadku użycia MOK 5 Dołącz poziom dostępu do konta.



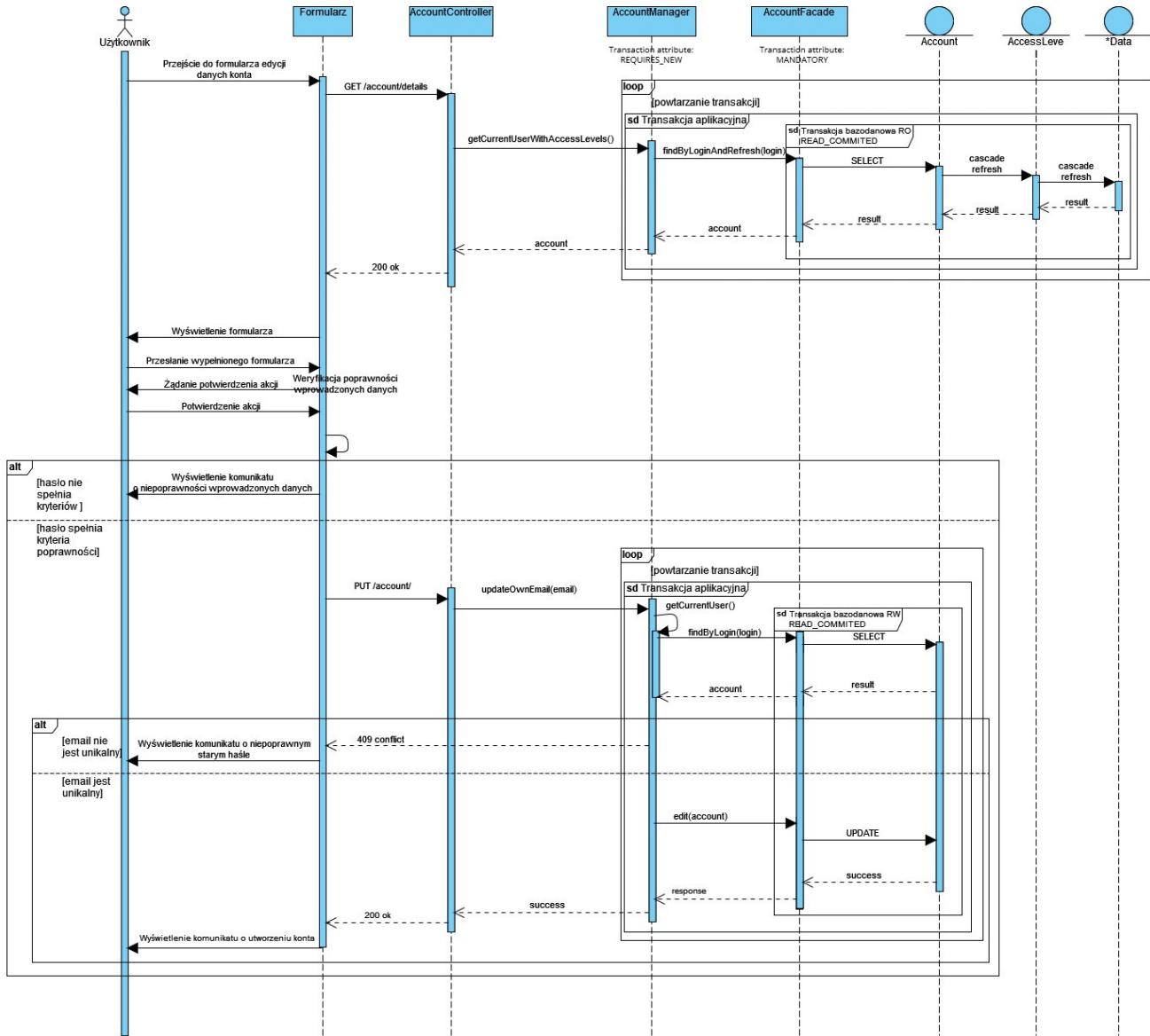
11.6 Diagram sekwencji dla przypadku użycia MOK 6 Odłącz poziom dostępu od konta.



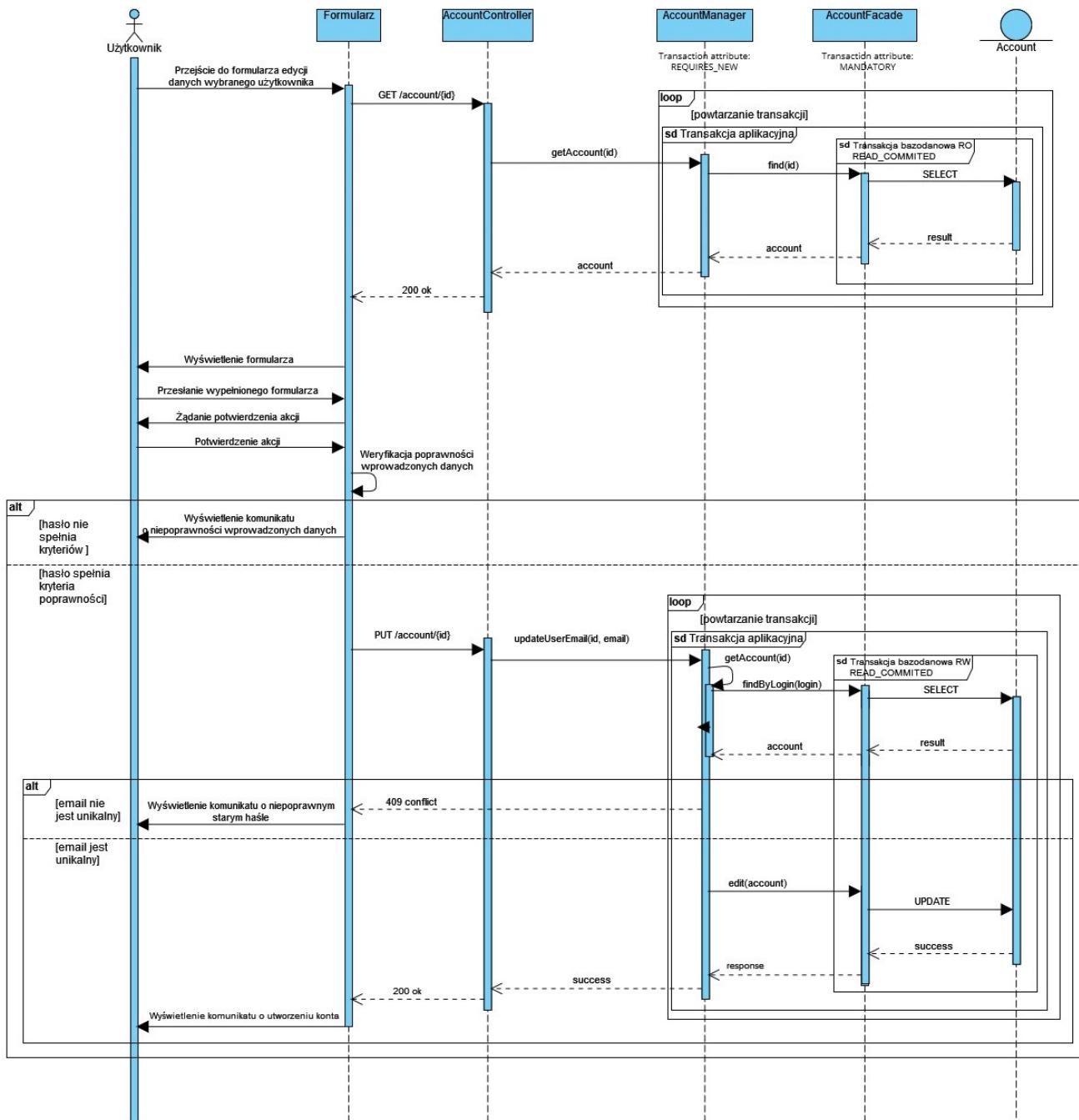
11.7 Diagram sekwencji dla przypadku użycia MOK 7 Zmień własne hasło.



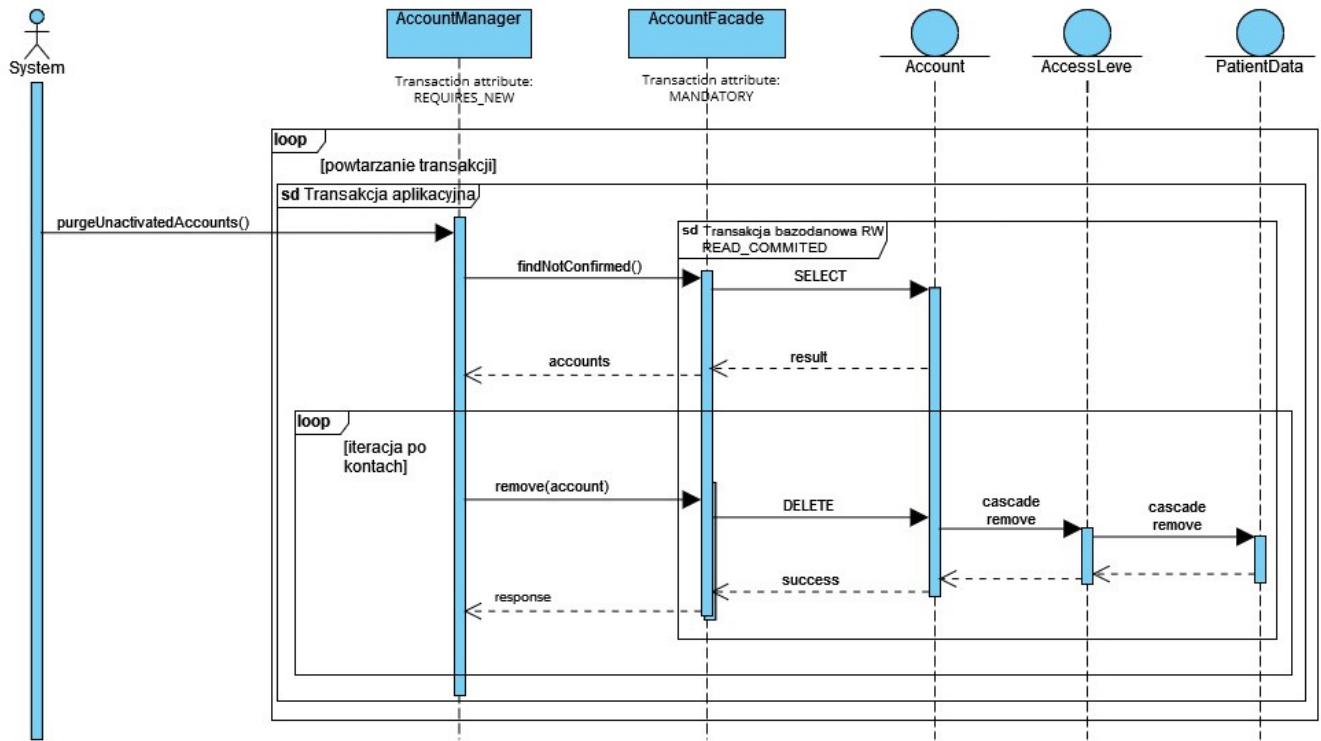
11.8 Diagram sekwencji dla przypadku użycia MOK 8 Zmień hasło innego użytkownika.



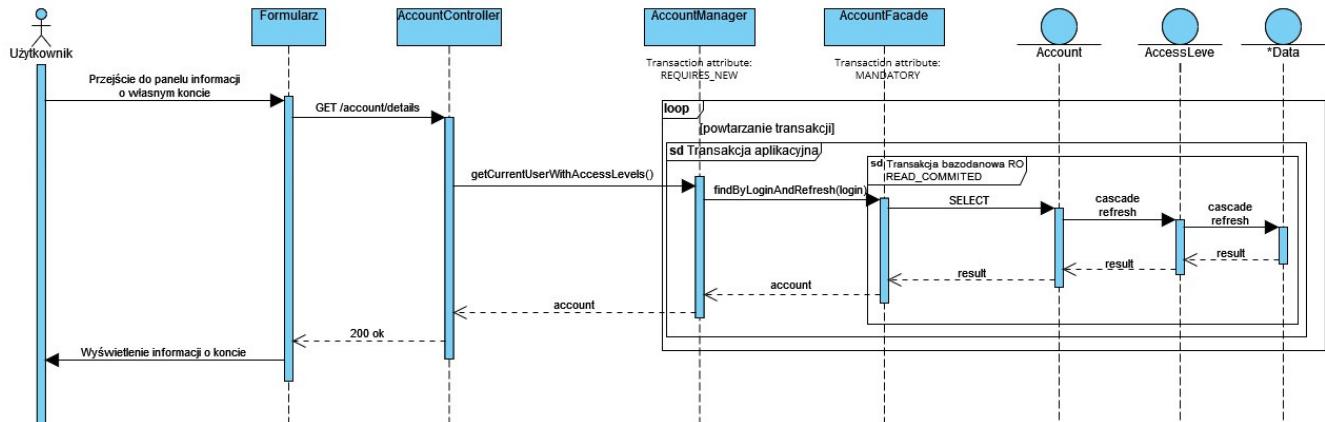
11.9 Diagram sekwencji dla przypadku użycia MOK 9 Edytuj dane własnego konta.



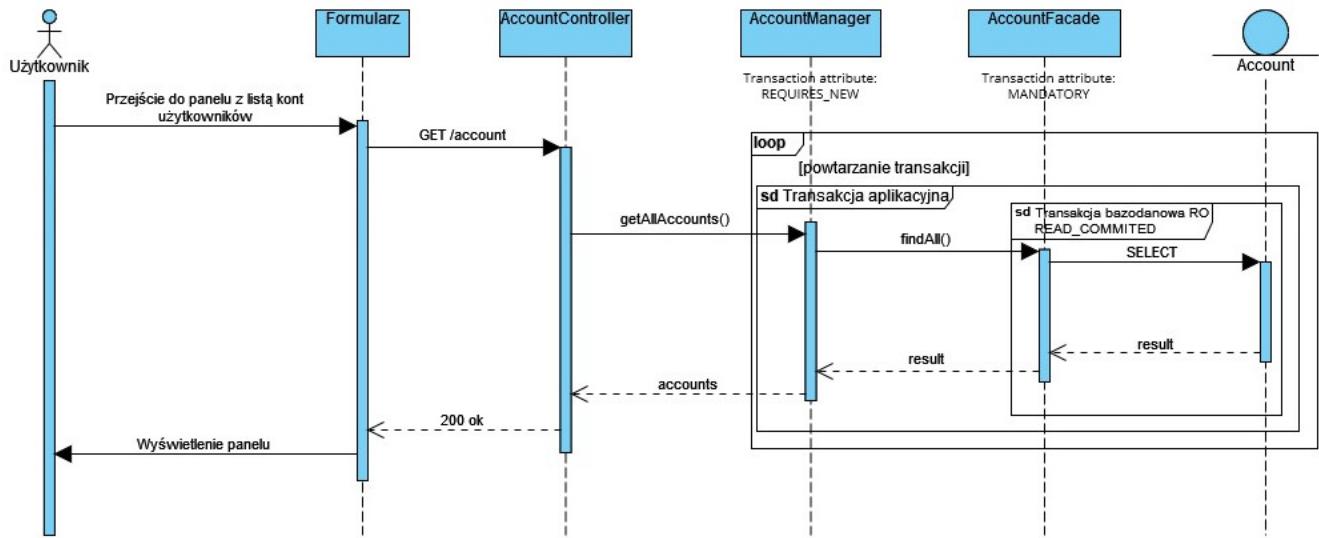
11.10 Diagram sekwencji dla przypadku użycia MOK 10 Edytuj dane konta innego użytkownika.



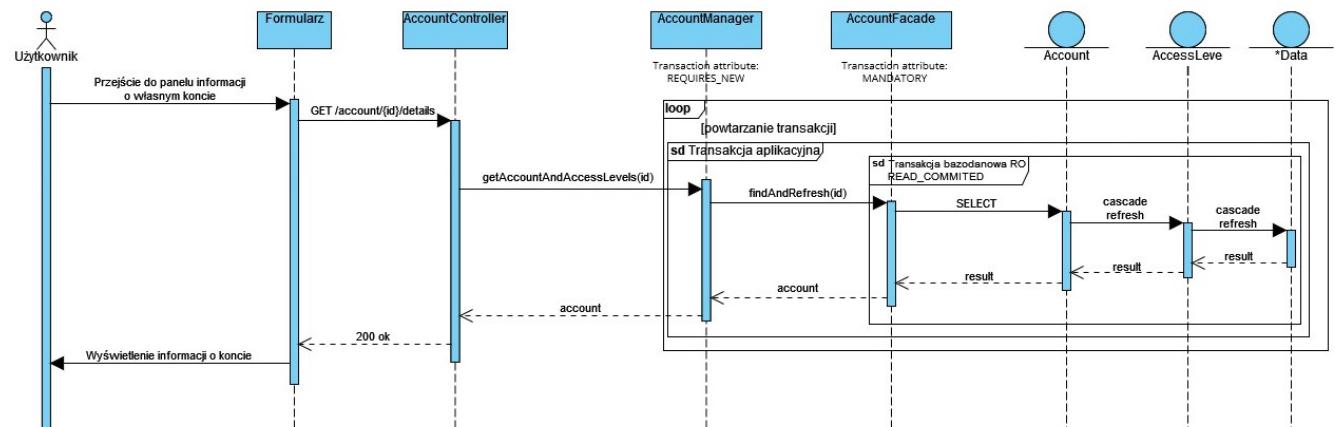
11.11 Diagram sekwencji dla przypadku użycia MOK 12 Usuń konto w przypadku braku aktywacji.



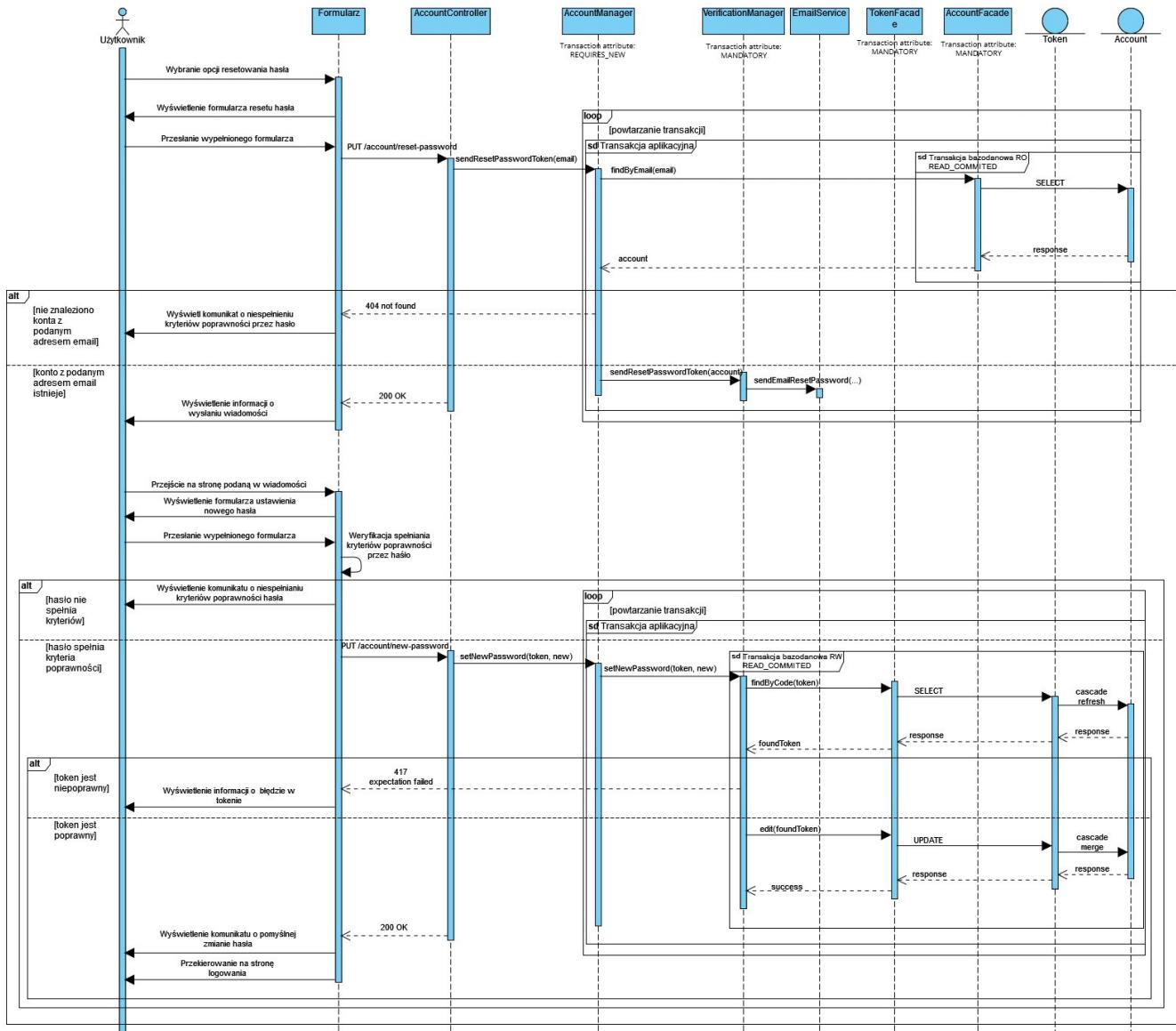
11.12 Diagram sekwencji dla przypadku użycia MOK 13 Wyświetl informacje o koncie.



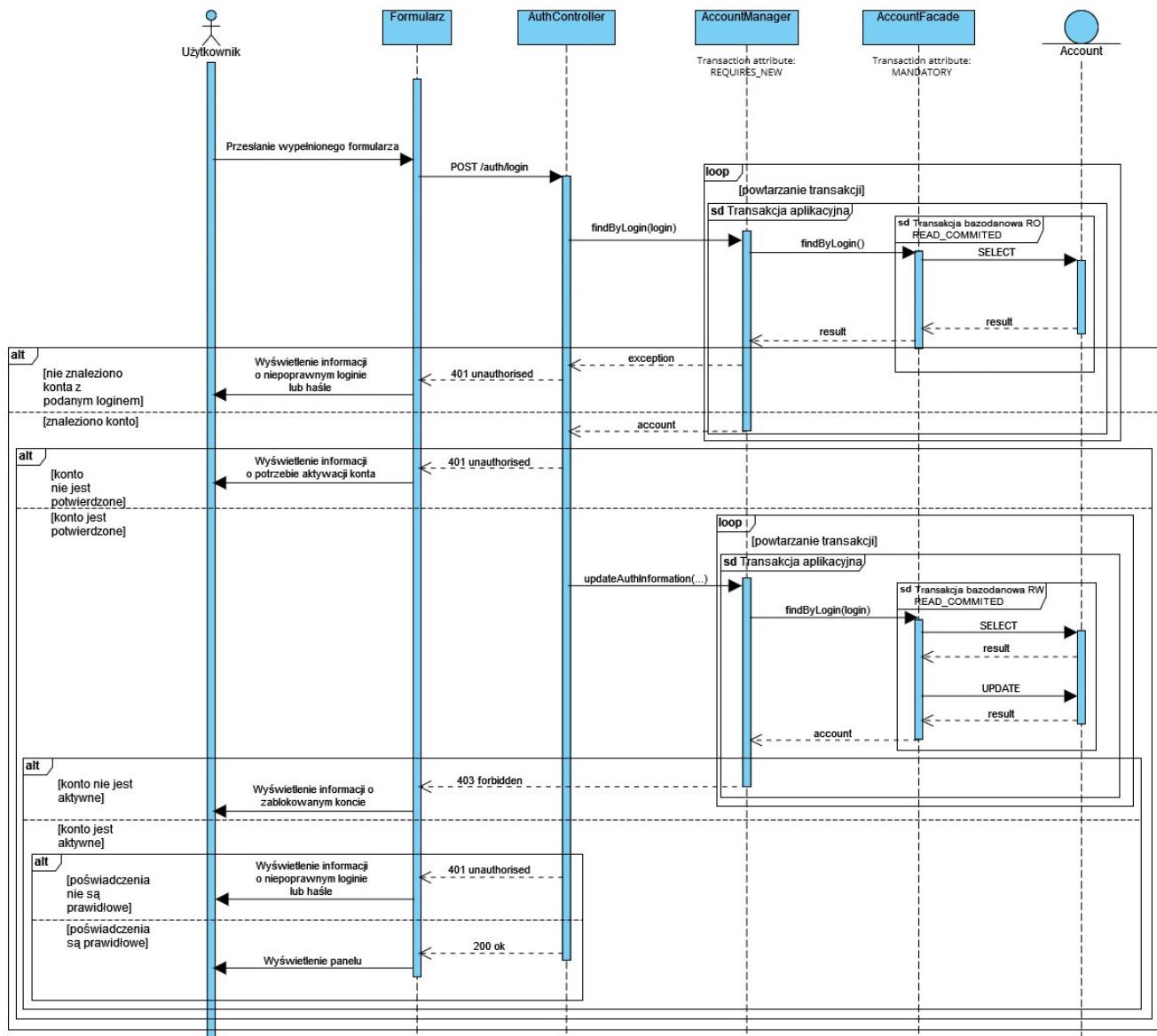
11.13 Diagram sekwencji dla przypadku użycia MOK 14 Wyświetl listę kont.



11.14 Diagram sekwencji dla przypadku użycia MOK 15 Wyświetl informacje o koncie innego użytkownika.



11.15 Diagram sekwencji dla przypadku użycia MOK 16 Zresetuj hasło.



11.16 Diagram sekwencji dla przypadku użycia MOK 17 Zaloguj się na konto.

Mechanizm działania blokady optymistycznej

Dla każdego typu encji, która może podlegać edycji zastosowano mechanizm, który zapobiega błędem wynikającym z jednoczesnej edycji tej samej encji.

AccountManager

```
@Override  
@RolesAllowed("updateUserEmail")  
public Account updateUserEmail(Long id, String email, String login, Long version) {  
    Account account = getAccount(id);  
    if(!account.getLogin().equals(login)) {  
        throw ApplicationException.createMismatchedPayloadException();  
    }  
    if(!account.getVersion().equals(version)) {  
        throw ApplicationException.createOptimisticLockException();  
    }  
    account.setEmail(email);  
    account.setConfirmed(true);  
    account.setModifiedBy(getCurrentUserLogin());  
    accountFacade.edit(account);  
    return account;  
}
```

W linii 8 widać, że jest weryfikowana wartość pola wersji. Jeśli nie jest ona zgodna z tą pobraną z bazy danych zgłoszany jest wyjątek aplikacyjny OptimisticLockException.

Aby zapobiec przypadkowi, że użytkownik zmieniłby ręcznie wartość pola wersji obliczany jest na jego podstawie, oraz innego pola, podpis cyfrowy. W przypadku edycji konta innego użytkownika (z którego pochodzi powyżej przytoczona funkcja) owo drugie pole stanowi wartość loginu edytowanego konta. W połączeniu z weryfikacją wartości w linii 5 daje nam gwarancję, że nie jest możliwe przesłanie tego samego ciała żądania w celu edycji innego konta.

Mechanizm obliczający wartość podpisu został przedstawiony poniżej.

EntityIdentitySignerVerifier

```
public String calculateEntitySignature(SignableEntity entity) {  
    try {  
        JWSSigner signer = new MACSigner(ETAG_SECRET);  
  
        JWSObject jwsObject =  
            new JWSObject(  
                new JWSHeader(JWSAlgorithm.HS256), new Payload(entity.getSignablePayload()));  
        jwsObject.sign(signer);  
        return jwsObject.serialize();  
    } catch (JOSEException e) {  
        throw ApplicationException.createEtagCreationException();  
    }  
}
```

Ta funkcja jest wywoływana za każdym razem jak wykonywany jest odczyt pojedynczej encji. Wynik jej jest przesyłany w nagłówku **ETag**. Funkcja getSignablePayload jest zdefiniowana dla każdej encji która może podlegać edycji.

Mechanizm weryfikujący wartość podpisu wygląda natomiast następująco:

EntityIdentitySignerVerifier

```
public void checkEtagIntegrity(SignableEntity entity, String etag) {  
    if (!verifyEntityIntegrity(entity, etag)) {  
        throw ApplicationException.createEtagNotValidException();  
    }  
}  
  
public boolean verifyEntityIntegrity(SignableEntity entity, String tag) {  
    try {  
        final String header = JWSSObject.parse(tag).getPayload().toString();  
        final String signablePayload = entity.getSignablePayload();  
        return validateEntitySignature(tag) && signablePayload.equals(header);  
    } catch (ParseException e) {  
        e.printStackTrace();  
        return false;  
    }  
}
```

Mechanizm ponawiania odwołanych transakcji aplikacyjnych

W każdym miejscu gdzie ma zostać wywołana metoda managera, a wraz z tym utworzona nowa transakcja, opakowywana jest ona w funkcję anonimową i, w zależności czy ma zwracać jakąś wartość czy nie, jest przekazywana jako argument do jednej z dwóch poniżej zaprezentowanych funkcji:

AbstractController

```
protected <T> T repeatTransaction(CommonManagerLocalInterface service, Supplier<T> method) {  
    int retryTXCounter = TRANSACTION_REPEAT_COUNT;  
    boolean rollbackTX = false;  
  
    T result = null;  
    do {  
        try {  
            result = method.get();  
            rollbackTX = service.isLastTransactionRollback();  
        } catch (EJBTransactionRolledbackException e) {  
            rollbackTX = true;  
        }  
    } while (rollbackTX && --retryTXCounter > 0);  
  
    if (rollbackTX && retryTXCounter == 0) {  
        throw new TransactionException("Transaction failed");  
    }  
    return result;  
}
```

AbstractController

```
protected void repeatTransactionVoid(CommonManagerLocalInterface service, VoidFI voidFI) {
    int retryTXCounter = TRANSACTION_REPEAT_COUNT;
    boolean rollbackTX = false;

    do {
        try {
            voidFI.action();
            rollbackTX = service.isLastTransactionRollback();
        } catch (EJBTransactionRolledbackException e) {
            rollbackTX = true;
        }
    } while (rollbackTX && --retryTXCounter > 0);

    if (rollbackTX && retryTXCounter == 0) {
        throw new TransactionException("Transaction failed");
    }
}
```

Na tą chwilę parametr TRANSACTION_REPEAT_COUNT ma wartość 3. Powyższe metody znajdują się w klasie AbstractController, z której dziedziczy każdy kontroler.

Kod wszystkich metod realizujących scenariusz pobrania wszystkich kont użytkowników

AccountController

```
@GET
@Path("/")
@Produces(MediaType.APPLICATION_JSON)
public List<AccountDTO> readAllClients() {
    List<Account> accounts =
        repeatTransaction(accountManager, () -> accountManager.getAllAccounts());
    return accounts.stream().map(AccountConverter::mapAccountToAccountDto).toList();
}
```

Funkcja repeatTransaction została zaprezentowana wyżej jako część mechanizmu do powtarzania transakcji.

AccountManager

```
@Override
@RolesAllowed("getAllAccounts")
public List<Account> getAllAccounts() {
    return accountFacade.findAll();
}
```

AccountFacade

```
@Override
@PermitAll
public List<Account> findAll() {
    return super.findAll();
}
```

AbstractFacade

```
protected List<T> findAll() {  
    CriteriaQuery cq = getEntityManager().getCriteriaBuilder().createQuery();  
    cq.select(cq.from(entityClass));  
    return getEntityManager().createQuery(cq).getResultList();  
}
```

Rejestrowanie transakcji aplikacyjnych w dzienniku zdarzeń

Program korzysta z trzech metod rejestrujących w dzienniku zdarzeń utworzenie, moment przed zakończeniem i moment po zakończeniu transakcji. Metody te są zdefiniowane w klasie abstrakcyjnej, z której następnie dziedziczy każda klasa managera.

W każdej z metod rejestrujących jest umieszczona informacja o loginie użytkownika na rzecz którego została wykonana transakcja (uzyskany za pomocą sctx.getCallerPrincipal().getName()), numerze identyfikacyjnym transakcji oraz klasie na rzecz której transakcja została wykonana. Dodatkowo rejestrowany jest również moment rozpoczęcia transakcji oraz jej wynik zakończenia (zatwierdzenie bądź wycofanie).

Ciało owej klasy zostało przedstawione poniżej.

```

class AbstractManager

public abstract class AbstractManager {

    protected static final Logger LOGGER = Logger.getGlobal();

    @Resource SessionContext sctx;

    private String transactionId;

    private boolean lastTransactionRollback;

    @TransactionalAttribute(NOT_SUPPORTED)
    public boolean isLastTransactionRollback() {
        return lastTransactionRollback;
    }

    public void afterBegin() {
        transactionId =
            Long.toString(System.currentTimeMillis())
            + ThreadLocalRandom.current().nextLong(Long.MAX_VALUE);
        LOGGER.log(
            Level.INFO,
            "Transaction TXid={0} begin at {1}, " + " identity: {2}",
            new Object[] {
                transactionId, this.getClass().getName(), sctx.getCallerPrincipal().getName()
            });
    }

    public void beforeCompletion() {
        LOGGER.log(
            Level.INFO,
            "Transaction TXid={0} before completion at" + " {1} identity: {2}",
            new Object[] {
                transactionId, this.getClass().getName(), sctx.getCallerPrincipal().getName()
            });
    }

    public void afterCompletion(boolean committed) {
        lastTransactionRollback = !committed;
        LOGGER.log(
            Level.INFO,
            "Transaction TXid={0} ended at {1}" + "by {3}, identity {2}",
            new Object[] {
                transactionId,
                this.getClass().getName(),
                sctx.getCallerPrincipal().getName(),
                committed ? "CONFIRMATION" : "CANCELLATION"
            });
    }
}

```

12 Zgłoszane wyjątki 4 pkt

Należy zaprezentować w formie listy lub tabeli wszystkie wyjątki biznesowe reprezentujące błędy specyficzne dla logiki aplikacji. Dla każdego z takich wyjątków należy określić co najmniej:

- miejsce/miejsca zgłoszenia wyjątku
- możliwe przyczyny wystąpienia wyjątku (np. niespełnianie założonych asercji przez parametry metody; obsługa innego wyjątku, w szczególności związanego z działaniem mechanizmu blokad optymistycznych - w takim przypadku należy w jednym zdaniu scharakteryzować okoliczności zgłoszenia tego wyjątku)
- stwierdzenie, czy wyjątek jest obsługiwany w warstwie logiki biznesowej (w takim przypadku należy podać klasę komponentu i metodę zawierającą stosowny segment obsługi wyjątku) czy też propaguje do warstwy widoku (w takim przypadku, jeśli obsługa wyjątku polega na wyeksponowaniu użytkownikowi odpowiedniego komunikatu, należy podać klucz zasobu zawierającego treść komunikatu)
- stwierdzenie, czy wystąpienie wyjątku prowadzi do odwołania aktualnie prowadzonej transakcji aplikacyjnej).

W taki sam sposób należy zaprezentować obsługę wyjątków systemowych (zgłoszanych przez kontener EJB). W tym przypadku lista wyjątków musi obejmować co najmniej:

- wyjątek/wyjątki związane z kontrolą dostępu do wywoływania metod biznesowych komponentów EJB/CDI
- wyjątek/wyjątki związane z błędnym zakończeniem transakcji aplikacyjnych
- pozostałe wyjątki

Realizacja: wszystkie planowane wyjątki aplikacyjne muszą być zadeklarowane w oddzielnym pakiecie klas.

Należy zwrócić uwagę na to, że wyjątkiem może zakończyć się operacja zestawiania / rozłączania związku między obiektami encji.

Prezentowane rozwiązania muszą być zaimplementowane w module MOK.

Wyjątek biznesowy	Reprezentowany błąd	Miejsca zgłoszania wyjątku	Możliwe przyczyny wystąpienia wyjątku	Czy wyjątek jest obsługiwany w warstwie logiki biznesowej czy też propaguje do warstwy widoku?	Czy wystąpienie wyjątku prowadzi do odwołania aktualnie prowadzonej transakcji aplikacyjnej?
TokenException	Błędy związane z tokenami (np. do zmiany hasła)	TokenManager	Otrzymany token jest błędny, nie istnieje w bazie danych, został już użyty lub wygasł.	Propagowany do warstwy widoku. exception.token.already-used exception.token.bad-type exception.token.not-found exception.token.expired	Tak
AccountApplicationException	Błędy związane z zarządzaniem kontami	AuthController, AccessLevelConverter, AccountManager, AccountFacadeExceptionInterceptor	Próba utworzenia nowego konta bądź modyfikacji pól które muszą być unikatowe kolidującymi z innym kontem (email, login, pesel nip lub numer telefonu) Próba zalogowania się na niepotwierdzone konto Próba dezaktywacji własnego konta. Próba wprowadzenia niedozwolonej zmiany w koncie. (np. usunięcie lud dezaktywacji ostatniego poziomu dostępu lub dodania kolejnego, który już istnieje na danym koncie, a także niepoprawnego poziomu dostępu)	Propagowany do warstwy widoku. exception.account.duplicate. email exception.account.duplicate. login exception.account.duplicate. pesel exception.account.duplicate. nip exception.account.duplicate. phone exception.account.constraint-violation exception.account.not-confirmed exception.account.no-such-access-level exception.account.duplicate-access-level exception.account.deactivate. self exception.account.deactivate. last-access-level	Tak

ApplicationException	różne błędy: błędy związane z etagiem, językiem, autoryzacją, zarządzaniem kontami, niepoprawnymi żądaniami 	AccountFacadeException nInterceptor AccessLevelFinder TokenFacadeInterceptor AccountManager GenericManagerExceptionInterceptor GenericFacadeException nInterceptor AuthController EntityIdentitySignerVerifier	Błąd podczas zmiany hasła, brak encji w bazie danych, brak autoryzacji do wykonania jakiejś czynności, odmowa dostępu, bra Pusty lub niepoprawny etag, błąd w trakcie tworzenia etaga Próba zmiany języka konta na niepoprawny.	Propagowany do warstwy widoku. exception.general exception.persistence exception.access-denied exception.unauthorised exception.password.not-changed exception.not-found exception.method-not-allowed exception.etag.empty exception.etag.not-valid exception.etag.creation exception.language.not-found miejscza obsługi wyjątku: AuthController. notifyAccessLevelChange() GenericFacadeExceptionsInterceptor.intercept() GenericManagerExceptionsInterceptor.intercept()	Tak
ApplicationExceptionEntityNotFound	brak szukanej(ych) encji w bazie danych	AccountFacadeException nInterceptor TokenFacadeInterceptor AccessLevelFinder AccountManager	zapytanie odwołuje się do nieistniejącej encji (np. konta lub poziomu dostępu)	w zależności od miejsca zgłoszenia obsługiwany w metodzie AuthController.authenticate() lub propagowany do warstwy widoku jako exception.entity-not-found	Tak
ApplicationExceptionOptimisticLock	Wystąpienie blokady optymistycznej	AccountManager GenericFacadeException nInterceptor	W przypadku otrzymania wersji encji starszej niż najnowsza (oznacza to że nastąpił konflikt edycji)	Propagowany do warstwy widoku exception.optimistic-lock	Tak
AuthApplicationException	Błąd uwierzytelnienia	AccountManager AuthController	Użytkownik nie może się zalogować z powodu błędego logatu lub hasła, albo ponieważ jego konto jest zablokowane.	Propagowany do warstwy widoku exception.auth.bad-credentials exception.auth.blocked-account	Tak

Wyjątek systemowy	Reprezentowany błąd	Miejsce zgłoszenia wyjątku	Możliwe przyczyny wystąpienia wyjątku	Czy wyjątek jest obsługiwany w warstwie logiki biznesowej czy też propaguje do warstwy widoku?	Czy wystąpienie wyjątku prowadzi do odwołania aktualnie prowadzonej transakcji aplikacyjnej?
EJBTransactionRolledbackException	Transakcja została wycowana	Serwis, Fasada	Wystąpił błąd w trakcie transakcji aplikacyjnej	jest obsługiwany przez metody repeatTransaction i repeatTransactionVoid klasy AbstractController ,	ten wyjątek jest rzucany jako konsekwencja odwołania transakcji aplikacyjnej (Tak)
EJBAccessLevelException	Nie można wykonać metody z powodu braku autoryzacji	Serwis, Fasada	Użytkownik spróbował wykonać akcję do której nie ma uprawnień	jest obsługiwany przez interceptor GenericManagerExceptionsInterceptor i konwertowany do Application Exception ('exception.access-denied')	Tak

13 Interfejs użytkownika 3 pkt

Wykorzystywane są dwa szablony stron - dla użytkowników zalogowanych oraz niezalogowanych.

Główną różnicą między ww. szablonami jest występujący u góry strony pasek nawigacji, który zawiera inne opcje w zależności od tego, czy użytkownik jest zalogowany oraz jaką obecnie posiada rolę.

Szablon strony dla użytkownika zalogowanego

```
(... importy zostaj pominitte)

function PublicLayout({children}) {
    return (
        <>
            <PublicNavbar/>
            {children}
        </>
    );
}

export default PublicLayout;
```

Stałe wykorzystywane w aplikacji front-end

```
export const JWT_TOKEN = "jwtToken";

export const ROLES = {
    ADMIN: "ADMIN",
    PATIENT: "PATIENT",
    CHEMIST: "CHEMIST",
}

export const languages = [
    {
        code: 'en',
        name: 'English',
        country_code: 'gb',
    },
    {
        code: 'pl',
        name: 'Polski',
        country_code: 'pl'
    },
    {
        code: 'cs',
        name: 'eský',
        country_code: 'cs'
    }
]
```

Górny pasek nawigacji występujący w szablonie PublicLayout

```
(... importy zostaj pominitte)

const guestTheme = createTheme({
    palette: {
```

```

        primary: blue,
    },
    typography: {
        fontFamily: [
            'Lato',
            'sans-serif',
        ].join(','),
    },
});
};

export default function PublicNavbar() {
    const navigate = useNavigate();
    const { t } = useTranslation();
    const [anchorEl, setAnchorEl] = React.useState(null);
    const open = Boolean(anchorEl);
    const currentLanguage = i18n.language;
    const handleClick = (event) => {
        setAnchorEl(event.currentTarget);
    };
    const handleClose = (event) => {
        event.stopPropagation();
        setAnchorEl(null);
    };

    return (
        <ThemeProvider theme={guestTheme}>
            <Box sx={{ flexGrow: 1 }}>
                <AppBar position="static">
                    <Toolbar>
                        <Typography onClick={()=>navigate(Pathnames.public.home)} variant="h6" component="div" sx={{ flexGrow: 1, cursor: 'pointer' }}>
                            {t('internet_pharmacy')}
                        </Typography>
                        <Button color="inherit" onClick={() => navigate(Pathnames.public.signup)}>
                            {t('sign_up')}
                        </Button>
                        <Button color="inherit" onClick={() => navigate(Pathnames.public.login)}>
                            Login
                        </Button>
                        <IconButton
                            size="large"
                            edge="start"
                            color="inherit"
                            aria-label="menu"
                            sx={{ mr: 2 }}
                            id="basic-button"
                            aria-controls={open ? 'basic-menu' : undefined}
                            aria-haspopup="true"
                            aria-expanded={open ? 'true' : undefined}
                            onClick={handleClick}
                        >
                            <LanguageIcon />
                            <Menu
                                id="basic-menu"
                                anchorEl={anchorEl}
                                open={open}
                                onClose={handleClose}
                                MenuListProps={{
                                    'aria-labelledby': 'basic-button',
                                }}
                            >
                                {languages.map(({ code, name, country_code }) => (
                                    <MenuItem disabled={code === currentLanguage} key={country_code} onClick={() => {
                                        i18n.changeLanguage(code)
                                        setAnchorEl(null);}}>
                                        {name}
                                    </MenuItem>
                                )));
                            </Menu>
                        </IconButton>
                    </Toolbar>
    );
}

```

```

        </AppBar>
    </Box>
    <ThemeProvider>
);
}

```

Dokładny opis kodu:

Ten kod definiuje komponent funkcjonalny `PublicNavbar`, który renderuje pasek nawigacji dla publicznych, niezalogowanych użytkowników w aplikacji. Zawiera on nazwę aplikacji, przyciski rejestracji i logowania, oraz menu rozwijane z wyborem języka.

Na początku kodu definiowany jest `guestTheme` za pomocą funkcji `createTheme`. Motyw określa ustawienia palety kolorów i typografii dla paska nawigacji użytkownika niezalogowanego.

Informacje o dostępnych językach aplikacji (polski, angielski i czeski) zawiera przedstawiona powyżej, eksportowana zmienna `languages`.

W komponencie używane są `hooki`, takie jak `useNavigate` i `useTranslation`, które umożliwiają nawigację po aplikacji i tłumaczenie tekstów na ww. języki.

Po kliknięciu na ikonkę kuli ziemskiej, wyświetlane jest rozwijane menu z ww. językami. Po wybraniu jednego z nich, wywoływana jest funkcja `i18n.changeLanguage` do zmiany aktualnej wersji językowej aplikacji.

Szablon strony dla użytkownika zalogowanego

```

(... importy zostaj pominitte)

function AuthLayout({children}) {
    return (
        <>
            <AuthNavbar/>
            {children}
        </>
    );
}

export default AuthLayout;

```

Górny pasek nawigacji występujący w szablonie AuthLayout

```

(... importy zostaj pominitte)

const guestTheme = createTheme({
    palette: {
        primary: blue,
    },
    typography: {
        fontFamily: ["Lato", "sans-serif"].join(","),
    },
});

const adminTheme = createTheme({
    palette: {
        primary: red,
    },
    typography: {
        fontFamily: ["Lato", "sans-serif"].join(","),
    },
});

const chemistTheme = createTheme({
    palette: {
        primary: purple,
    },
});

```

```

typography: {
    fontFamily: ["Lato", "sans-serif"].join(","),
},
});

const patientTheme = createTheme({
  palette: {
    primary: pink,
  },
  typography: {
    fontFamily: ["Lato", "sans-serif"].join(","),
  },
});
};

export default function AuthNavbar() {
  const navigate = useNavigate();
  const {t} = useTranslation();
  const userRole = useSelector((state) => state.user.cur);
  const currentLanguage = i18n.language;
  const user = useSelector((state) => state.user);
  const login = useSelector((state) => state.user.sub);
  const roles = useSelector((state) => state.user.roles);
  const dispatch = useDispatch();
  const [currentTheme, setCurrentTheme] = useState(guestTheme);
  const [anchorElRole, setAnchorElRole] = React.useState(null);
  const [anchorElLanguage, setAnchorElLanguage] = React.useState(null);

  const handleClick = (event) => {
    setAnchorElRole(event.currentTarget);
  };

  const handleClose = (event) => {
    event.stopPropagation();
    setAnchorElRole(null);
  };
}

useEffect(() => {
  if (userRole === ROLES.ADMIN) {
    setCurrentTheme(adminTheme);
  } else if (userRole === ROLES.CHEMIST) {
    setCurrentTheme(chemistTheme);
  } else if (userRole === ROLES.PATIENT) {
    setCurrentTheme(patientTheme);
  }
}, [userRole]);

const [dialogOpen, setDialogOpen] = useState(false);

const accept = () => {
  dispatch(logout());
  navigate(Pathnames.public.login);
};

const reject = () => {
  setDialogOpen(false);
};

const location = useLocation();

let current = "";

return (
  <ThemeProvider theme={currentTheme}>
    <Box sx={{flexGrow: 1}}>
      <AppBar position="static">
        <Toolbar>
          <Typography
            onClick={() => navigate(Pathnames.auth.landing)}
            variant="h6"
            component="div"
            sx={{flexGrow: 1, cursor: "pointer"}}

```

```

>           {t("internet_pharmacy")}
</Typography>
{userRole === ROLES.ADMIN && (
    <Button
        color="inherit"
        onClick={() => navigate(Pathnames.admin.accounts)}
    >
        {t("accounts")}
    </Button>
)
}
<Box>
    <Typography>{login}</Typography>
    <Typography>{userRole}</Typography>
</Box>
<IconButton color="inherit" onClick={(e) => {
    setAnchorElRole(e.currentTarget);
}}>
    <ManageAccountsIcon/>
    <Menu
        id="simple-menu"
        anchorEl={anchorElRole}
        open={Boolean(anchorElRole)}
        onClose={(e) => {
            e.stopPropagation();
            setAnchorElRole(null);
        }}
        menuListProps={{
            "aria-labelledby": "basic-button",
        }}>
        {roles.map((role, index) => (
            <MenuItem
                disabled={role === userRole}
                key={role}
                onClick={(event) => {
                    dispatch(
                        changeLevel({
                            sub: user.sub,
                            roles: user.roles,
                            index: index,
                            exp: user.exp,
                        })
                });
                handleClose(event);
            }}>
            {role}
        </MenuItem>
        ))}
    </Menu>
</IconButton>
<IconButton
    color="inherit"
    onClick={() => {
        navigate(Pathnames.auth.self);
    }}
>
    <AccountCircle/>
</IconButton>
<IconButton
    size="large"
    edge="start"
    color="inherit"
    aria-label="menu"
    // sx={{mr: 2}}
    id="basic-button"
    aria-controls={Boolean(anchorElRole) ? 'basic-menu' : undefined}
    aria-haspopup="true"
    aria-expanded={Boolean(anchorElRole) ? 'true' : undefined}
    onClick={(e) => {
        setAnchorElLanguage(e.currentTarget)
    }}

```

```

    >
      <LanguageIcon/>
      <Menu
        id="basic-menu"
        anchorEl={anchorElLanguage}
        open={Boolean(anchorElLanguage)}
        onClose={(e) => {
          e.stopPropagation();
          setAnchorElLanguage(null);
        }}
        menuListProps={{
          'aria-labelledby': 'basic-button',
        }}
      >
        {languages.map(({code, name, country_code}) => (
          <MenuItem disabled={code === currentLanguage} key={country_code} onClick={

() => {
          changeLanguage(code)
          i18n.changeLanguage(code)
          setAnchorElLanguage(null);
        }}>
          {name}
        </MenuItem>
      ))}
    </Menu>
  </IconButton>
  <IconButton
    color="inherit"
    onClick={() => {
      setDialogOpen(true);
    }}
  >
    <Logout/>
  </IconButton>
  <ConfirmationDialog
    open={dialogOpen}
    title={t("confirm_logout")}
    actions={[
      {label: t("logout"), handler: accept, color: "primary"},
      {label: t("cancel"), handler: reject, color: "secondary"},
    ]}
    onClose={() => setDialogOpen(false)}
  />
</Toolbar>
</AppBar>
<Breadcrumbs separator=">" aria-label="breadcrumb">
  {location.pathname
    .split("/")
    .filter((crumb) => crumb !== "")
    .map((crumb) => {
      if (!isNaN(crumb)) {
        return null; // Ignore numbers in the URL
      }
      current += `${crumb}`;
      const capitalizedCrumb =
        crumb.charAt(0).toUpperCase() + crumb.slice(1);
      return (
        <Link
          underline="hover"
          color="inherit"
          href={current}
          sx={{fontSize: "18px"}}
        >
          {t(capitalizedCrumb)}
        </Link>
      );
    })}
</Breadcrumbs>
</Box>
</ThemeProvider>
);

```

```
}
```

Dokładny opis kodu:

Ten kod definiuje komponent *AuthNavbar*, który reprezentuje pasek nawigacji dla stron użytkowników zalogowanych.

Na początku kodu, podobnie jak w komponencie *PublicNavbar*, znajdują się deklaracje motywów, tym razem jednak dotyczą one także określonych poziomów dostępu (*a* *dminTheme*, *chemistTheme*, *patientTheme*). W zależności od poziomu dostępu ww. pasek nawigacji przyjmuje inny kolor.

Wewnątrz komponentu *AuthNavbar* także używane są *hooki* oraz selektory *Redux*, takie jak *useSelector* i *useDispatch*, aby pobierać informacje m.in. o zalogowanym użytkowniku i dostępnych rolach.

Wyświetlana jest nazwa użytkownika i jego obecna rola oraz ikony umożliwiające nawigację do innych stron - edycji własnego konta dla każdego poziomu dostępu oraz wyświetlenia wszystkich kont i ich późniejszej edycji przez użytkowników posiadających rolę ADMIN oraz oczywiście opcja wylogowania z aplikacji.

Podobnie jak w opisanym wcześniej *PublicNavbar*, także i tutaj istnieje możliwość zmiany języka na polski, czeski lub angielski. W tym wypadku jednak zmiana ta zostaje odzwierciedlona w informacjach użytkownika w bazie danych.

Ponadto na początku paska nawigacyjnego są wyświetlane *breadcrumbs*, które informują użytkownika o jego aktualnej lokalizacji na stronie.

Odpowiednie przypisanie szablonu do danej strony następuje w komponencie *RoutesComponent*:

Komponent RoutesComponent

```
(... importy zostaj pominitte)

export const RoutesComponent = () => {

    const user = useSelector((state) => state.user);

    const dispatch = useDispatch();
    const token = localStorage.getItem(JWT_TOKEN);
    try {
        if (user.exp === "" && token) {
            const decoded_token = jwtDecode(token);

            if (decoded_token) {
                dispatch(loginDispatch(decoded_token));
            }
        }
    } catch (error) {
        console.error(error);
    }

    return (
        <Routes>
            {publicRoutes.map(({path, Component}) => (
                <Route
                    key={path}
                    path={path}
                    element={
                        <PublicLayout>
                            <Component/>
                        </PublicLayout>
                    }
                />
            ))}
            {
                user.cur === ROLES.PATIENT &&
                PatientRoutes.map(({path, Component}) => (
                    <Route
                        key={path}
                        path={path}
                        element={
                            <AuthLayout>
                                <Component/>
                            </AuthLayout>
                        }
                    />
                ))
            }
        )
    )
}
```

```

        } />
    ) ) )
{
    user.cur === ROLES.ADMIN &&
AdminRoutes.map(( {path, Component} )) => (
    <Route
        key={path}
        path={path}
        element={
            <AuthLayout>
                <Component/>
            </AuthLayout>
        } />
    ) ) )
{
    user.cur === ROLES.CHEMIST &&
ChemistRoutes.map(( {path, Component} )) => (
    <Route
        key={path}
        path={path}
        element={
            <AuthLayout>
                <Component/>
            </AuthLayout>
        } />
    ) ) )
{
    (user.cur === ROLES.ADMIN || user.cur === ROLES.PATIENT || user.cur === ROLES.CHEMIST) &&
AuthRoutes.map(( {path, Component} )) => (
    <Route
        key={path}
        path={path}
        element={
            <AuthLayout>
                <Component/>
            </AuthLayout>
        } />
    ) ) )
<Route path="*" element={
    <PublicLayout>
        <Error/>
    </PublicLayout>
} />
</Routes>
)
}

```

Dokładny opis kodu:

Ten kod definiuje komponent funkcyjny *RoutesComponent*, który jest odpowiedzialny za *renderowanie tras (routes)* w aplikacji, w zależności od roli użytkownika.

Na początku kodu, identycznie jak w sytuacji w *AuthNavbar*, używane są *useSelector* i *useDispatch*, aby uzyskać dostęp do informacji o obecnie zalogowanym użytkowniku.

Wewnątrz komponentu odczytywana jest wartość tokenu z *local storage* (*localStorage.getItem(JWT_TOKEN)*) i dekodowana przy użyciu funkcji *jwtDecode*. Jeśli token został zdekodowany poprawnie, wywoływana jest akcja *loginDispatch*, która aktualizuje stan użytkownika.

Następnie renderowane są trasy (*route*) przy użyciu komponentu *Routes*. Dla publicznych tras, jako szablon wykorzystywany jest wspomniany wyżej *PublicLayout*.

Analogicznie dla tras związanych z rolą pacjenta (*PATIENT*), admina (*ADMIN*) i aptekarza (*CHEMIST*), wykorzystywany jest *AuthLayout*, jeśli rola użytkownika zalogowanego (*user.cur*) odpowiada określному poziomowi dostępu.

Dodatkowo, dla tras wymagających autoryzacji, niezależnie od roli, również jako szablon jest wykorzystywany *AuthLayout*.

Na końcu, w przypadku braku dopasowania do żadnej trasy, renderowany jest komponent *Error*, zawierający informacje o błędzie, gdzie jako szablon został użyty *PublicLayout*.

Z komponentem RoutesComponent związane są inne komponenty:

Pathnames, który przypisuje endpointy do określonych pól obiektu

Obiekt Pathnames

```
export const Pathnames = {
  public: {
    login: '/login',
    signup: '/sign-up',
    home: '/',
    confirmAccount: '/confirm-account/:token',
    resetPassword: '/accounts/reset-password',
  },
  auth: {
    landing: '/landing',
    self: '/accounts/self',
  },
  admin: {
    accounts: '/accounts',
    editSingleAccount: '/accounts/edit/:id',
    editChemist: '/edit-chemist/:id',
    details: '/accounts/:id/details',
    account: '/accounts/:id',
  },
  patient: {
  },
  chemist: {
  }
}
```

oraz *Routes, które wiążą ww endpointy z odpowiadającymi stronami:

Obiekty *Routes

```
(... importy zostay pominit)

export const publicRoutes = [
  {
    path: Pathnames.public.home,
    Component: Home,
  },
  {
    path: Pathnames.public.login,
    Component: Login,
  },
  {
    path: Pathnames.public.signup,
    Component: SignUp,
  },
  {
    path: Pathnames.public.resetPassword,
    Component: ResetPassword
  },
  {
    path: Pathnames.public.confirmAccount,
    Component: ConfirmAccount
  },
]
]

export const AuthRoutes = [
  {
    path: Pathnames.auth.self,
    Component: AccountDetails
  },
  {
    path: Pathnames.auth.landing,
    Component: Landing
  },
]
]

export const PatientRoutes = []

export const AdminRoutes = [
  {
    path: Pathnames.admin.account,
    Component: SingleAccount
  },
  {
    path: Pathnames.admin.editSingleAccount,
    Component: EditSingleAccount
  },
  {
    path: Pathnames.admin.editChemist,
    Component: EditChemist
  },
  {
    path: Pathnames.admin.details,
    Component: SingleAccount
  },
  {
    path: Pathnames.admin.accounts,
    Component: AllAccounts
  },
]
]

export const ChemistRoutes = []
```

Cała logika związana z mechanizmem doboru strony głównej po poprawnym zakończeniu procesu uwierzytelniania w aplikacji, dla konta z kilkoma poziomami dostępu znajduje się w komponencie *Login.js*. Ze względu iż zdecydowana większość kodu znajdującego się w tym komponencie nie ma wpływu na ww. logikę, zaprezentowane zostaną jedynie fragmenty wykorzystujące ją bezpośrednio.

Logika związana z mechanizmem doboru strony głównej po poprawnym zakończeniu procesu uwierzytelniania w aplikacji, dla konta z kilkoma poziomami dostępu

```
const dispatch = useDispatch();
const {t} = useTranslation();
const [dialogOpen, setDialogOpen] = useState(false);
const userRoles = useSelector((state) => state.user.roles);
const user = useSelector((state) => state.user);

signInAccount(login, password).then((response) => {
    setLoading(false)
    const jwt = response.data.jwtToken;
    localStorage.setItem(JWT_TOKEN, jwt);
    const decodedJWT = jwtDecode(jwt);
    dispatch(loginDispatch(decodedJWT));
    const roles = decodedJWT.roles;
    if (!roles.includes(","))
        navigate(Pathnames.auth.landing)
    else {
        setDialogOpen(true)
    }
}).catch((error) => {
    (... obsuga kodu błędów)
})
}

return (
    (... cz kodu nie zwizana bezporednio z logik przenoszenia)

    <Dialog open={dialogOpen} onClose={() => {
        setDialogOpen(false)
    }}>
        <DialogTitle>{t("choose_role")}</DialogTitle>
        <DialogActions>
            {
                userRoles.map((role, index) => {
                    return <Button onClick={() => {
                        dispatch(changeLevel({
                            sub: user.sub,
                            roles: user.roles,
                            index: index,
                            exp: user.exp,
                        }));
                        navigate(Pathnames.auth.landing);
                        setDialogOpen(false)
                    }}>{role}</Button>
                })
            }
        </DialogActions>
    </Dialog>

```

Dokładny opis kodu:

Ten kod odpowiada za fragment procesu logowania użytkownika przy użyciu funkcji *signInAccount(login, password)*, która wysyła żądanie do API oraz zwraca odpowiedź (response) po zakończeniu żądania.

Następnie, z odpowiedzi otrzymywany jest token JWT (`response.data.jwtToken`), który jest zapisywany w lokalnej pamięci przeglądarki (`localStorage.setItem(JWT_TOKEN, jwt)`). Po tej operacji, token jest dekodowany przy użyciu funkcji `jwtDecode` i rezultat jest przekazywany do akcji `loginDispatch`, która aktualizuje stan użytkownika.

Następnie sprawdzane są role użytkownika (`decodedJWT.roles`). Jeśli użytkownik ma tylko jedną rolę, następuje przeniesienie do strony głównej użytkownika zalogowanego (`Landing.js`). W przeciwnym przypadku, ustawiany jest stan `dialogOpen` na `true`, co powoduje otwarcie dialogu wyboru roli.

Wewnątrz dialogu wyświetlane są przyciski reprezentujące różne role użytkownika (`userRoles`). Po kliknięciu na przycisk, wywoływana jest akcja `changeLevel`, która aktualizuje poziom dostępu użytkownika, a następnie także następuje przeniesienie ww. strony głównej użytkownika zalogowanego (`Landing.js`)

14 Zabezpieczenia interfejsu użytkownika 3 pkt

Należy przedstawić - w formie zacytowania odpowiednich fragmentów konfiguracji lub kodu aplikacji - konfigurację kontroli dostępu użytkownika do widoków z interfejsem użytkownika.

RoutesComponent.js

```
export const RoutesComponent = () => {

    const user = useSelector((state) => state.user);

    const dispatch = useDispatch();
    const token = localStorage.getItem(JWT_TOKEN);
    try {
        if (user.exp === "" && token) {
            const decoded_token = jwtDecode(token);

            if (decoded_token) {
                dispatch(loginDispatch(decoded_token));
            }
        }
    } catch (error) {
        console.error(error);
    }

    return (
        <Routes>
            {publicRoutes.map(({path, Component}) => (
                <Route
                    key={path}
                    path={path}
                    element={
                        <PublicLayout>
                            <Component/>
                        </PublicLayout>
                    }
                />
            ))}
            {
                user.cur === ROLES.PATIENT &&
                PatientRoutes.map(({path, Component}) => (
                    <Route
                        key={path}
                        path={path}
                        element={
                            <AuthLayout>
                                <Component/>
                            </AuthLayout>
                        }
                    />
                )));
            }
            {
                user.cur === ROLES.ADMIN &&
                AdminRoutes.map(({path, Component}) => (
                    <Route
                        key={path}
                        path={path}
                        element={
                            <AuthLayout>
                                <Component/>
                            </AuthLayout>
                        }
                    />
                )));
            }
            {
                user.cur === ROLES.CHEMIST &&
                ChemistRoutes.map(({path, Component}) => (
                    <Route
                        key={path}
                        path={path}
                        element={
                            <AuthLayout>
                                <Component/>
                            </AuthLayout>
                        }
                    />
                )));
            }
        
```

```

        key={path}
        path={path}
        element={
          <AuthLayout>
            <Component/>
          </AuthLayout>
        }/>
      ))}
    {
      (user.cur === ROLES.ADMIN || user.cur === ROLES.PATIENT || user.cur === ROLES.CHEMIST) &&
      AuthRoutes.map(({path, Component}) => (
        <Route
          key={path}
          path={path}
          element={
            <AuthLayout>
              <Component/>
            </AuthLayout>
          }/>
      ))}

      <Route path="/" element={
        <PublicLayout>
          <Error/>
        </PublicLayout>
      }/>
    </Routes>
  )
}

```

Ponadto należy zaprezentować mapowanie grup oraz tożsamości użytkowników na role aplikacji w warstwie widoku. Informacje te należy zaprezentować w formie tabeli lub wycinka z odpowiedniego deskryptora aplikacji.

UserSlice.js

```
export const userSlice = createSlice({
  name: "user",
  initialState,
  reducers: {
    logout: () => {
      localStorage.removeItem(ACCESS_LEVEL);
      localStorage.removeItem(JWT_TOKEN);
      return initialState;
    },
    login: (state, action) => {
      const data = action.payload;

      if (
        !localStorage.getItem(ACCESS_LEVEL) ||
        localStorage.getItem(ACCESS_LEVEL) === ""
      ) {
        const res = {
          sub: data.sub,
          roles: data.roles.split(","),
          cur: data.roles.split(",")[0],
          exp: data.exp,
        };
        localStorage.setItem(ACCESS_LEVEL, res.cur);
        return { ...state, ...res };
      } else {
        const res = {
          sub: data.sub,
          roles: data.roles.split(","),
          cur: localStorage.getItem(ACCESS_LEVEL),
          exp: data.exp,
        };
        return { ...state, ...res };
      }
    },
    changeLevel: (state, action) => {
      const data = action.payload;
      const res = {
        sub: data.sub,
        roles: data.roles,
        cur: data.roles[data.index],
        exp: data.exp,
      };
      localStorage.setItem(ACCESS_LEVEL, res.cur);
      notifyAccessLevelChange(res.cur).then().catch();
      return { ...state, ...res };
    },
  },
});
```

Należy też przedstawić rozwiązań zapewniające wyliczanie i składowanie haseł przypisanych do kont użytkowników w bazie danych w postaci niejawniej (skrótu) wraz z przykładem jego wykorzystania.

AccountManager.java

```
public Account createAccount(Account account) {
  account.setPassword(HashAlgorithmImpl.generate(account.getPassword()));    <-- "HashAlgorithmImpl.generate
  (account.getPassword()" fragment kodu odpowiedzialny za transformację hasa użytkownika z postaci jawnej do
niejawniej
  account.setCreatedBy(getCurrentUserLogin());
  accountFacade.create(account);
  return account;
}
```

HashAlgorithmImpl.java

```
public class HashAlgorithmImpl implements PasswordHash {                                     <-- Klasa
odpowiedzialna za zamian postaci jawnej do niejawnej oraz weryfikacj zgodnoci dostarczonych danych

    public static String generate(String password) {                                         <-- Funkcja
Funkcja przyjmujaca posta jawn danych oraz zwracajca posta niejawn
        return DigestUtils.sha256Hex(password);
    }

    public String generate(char[] password) {
        return generate(new String(password));
    }

    public static boolean check(String password, String hashedPassword) {                <-- Funkcja sprawdzajaca
czy dostarczona warto jawnia oraz zapisana warto niejawna s zgodne
        String toVerify = generate(password);
        return hashedPassword.equals(toVerify);
    }

    public boolean verify(char[] password, String hashedPassword) {
        return check(new String(password), hashedPassword);
    }
}
```

Zrzut ekranu wartości kolumny "password" w bazie danych, która zapisana jest w niej w postaci niejawnej:

▼ WHERE ORDER BY					
	vage	: last_negative_login	: last_positive_login	: logical_address	: login : password
1		2023-05-22 18:00:48.350000	2023-05-22 18:00:52.856000	0:0:0:0:0:0:1	chemist123 b03ddf3ca2e714a6548e7495e2a03f5e824eaac9837cd7f159c67b90fb4b7342
2	<null>		2023-05-22 18:04:02.910000	0:0:0:0:0:0:1	MGrechuta b03ddf3ca2e714a6548e7495e2a03f5e824eaac9837cd7f159c67b90fb4b7342
3	<null>		2023-05-22 18:11:46.916000	0:0:0:0:0:0:1	admin123 b03ddf3ca2e714a6548e7495e2a03f5e824eaac9837cd7f159c67b90fb4b7342

15 Zmiany - projekt szczegółowy

W rozdziale tym należy wymienić zmiany wprowadzone do sprawozdania wstępnego, zwięzłe je charakteryzując. Dla każdej zmiany wymagane jest zamieszczenie odwołania do strony poddanej modyfikacjom.

Dodano poziom dostępu System. Dodano alternatywną metodę tworzenia konta z poziomem dostępu administratora [2 Moduły funkcjonalne, aktorzy, przypadki użycia 3 pkt](#)

todo opisać zmiany w [3 Struktury relacyjnej bazy danych](#).

Dodano uprawnienia do nowych tabel w [4 Użytkownicy bazodanowi](#).

Dodano stan aktualny struktur bazy danych [5 Identyfikacja obiektów encji](#).

Zmieniono diagram klas encji, dodano encję Token. Rozbudowanie encji o nowe pola i metody [6 Diagram klas encji](#).

Zostały zmienione diagramy sekwencji dla wszystkich przypadków użycia oprócz MOK 11 w rozdziale [7 Diagramy sekwencji oraz identyfikacja transakcji bazodanowych](#).

Sprawozdanie końcowe (10 pkt)

Sprawozdanie końcowe

16 Zabezpieczenie transmisji HTTP 1 pkt

Przedstawić konfigurację aplikacji (fragmenty deskryptora) lub serwera pośredniczącego (proxy) wymuszającą korzystanie z szyfrowanego protokołu HTTPS w komunikacji między przeglądarką WWW a serwerem. Prezentowana konfiguracja musi być w całości zrealizowana w aplikacji.

Konfiguracja aplikacji wymuszająca komunikację za pośrednictwem protokołu HTTPS

Owa konfiguracja jest zapewniona przez serwer pośredniczący nginx. Poniżej przedstawiony jest fragment pliku konfiguracyjnego odpowiadający za przekierowywanie komunikacji HTTP na HTTPS.

Fragment pliku /etc/nginx/nginx.conf

```
(...)
server {
    listen      80;
    listen      [::]:80;
    server_name team-1.proj-sum.it.p.lodz.pl;
    return 301 https://$server_name$request_uri;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;
}

upstream payara {
    server 10.31.201.3:8080;
}

server {
    listen      443 ssl http2;
    listen      [::]:443 ssl http2;

    location /api { # redirect to REST API
        proxy_pass http://payara;
    }

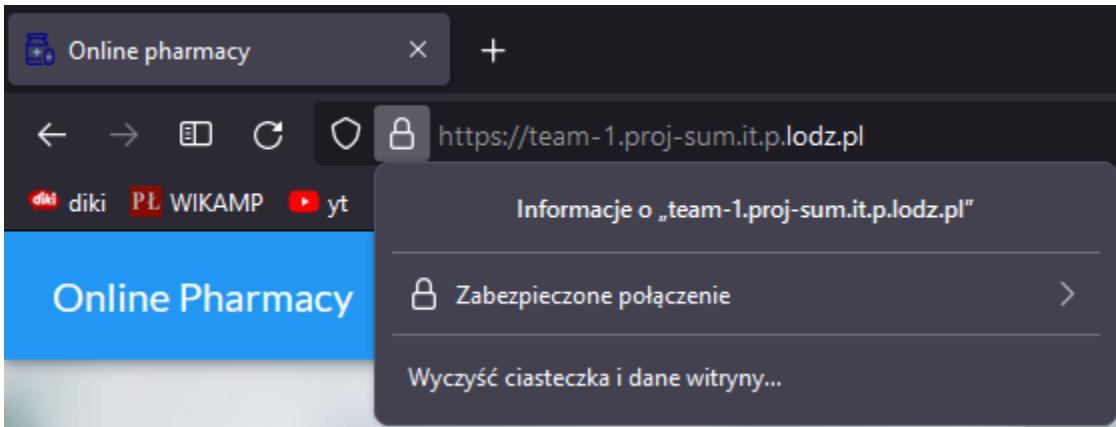
    location / { # redirect to SPA application
        root /var/www/html;
        try_files $uri /index.html;
    }

    ssl_certificate "/etc/nginx/acme.sh/team-1.proj-sum.it.p.lodz.pl_ecc/team-1.proj-sum.it.p.lodz.pl.cer";
    ssl_certificate_key "/etc/nginx/acme.sh/team-1.proj-sum.it.p.lodz.pl_ecc/team-1.proj-sum.it.p.lodz.pl.key";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_ciphers PROFILE=SYSTEM;
    ssl_prefer_server_ciphers on;

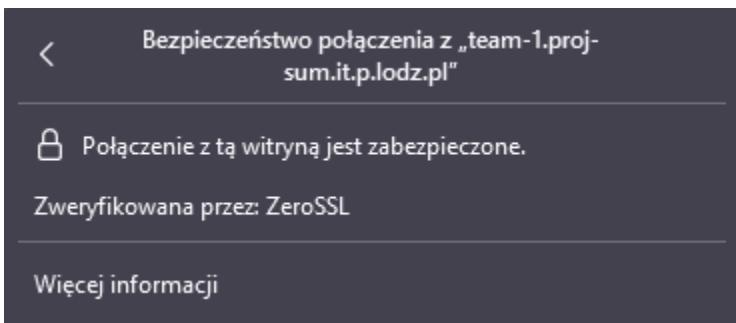
    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;
}
}
```

Nawiązywanie połączenia za pomocą protokołu HTTPS

Ponieważ certyfikat użyty na stronie jest zweryfikowany przez CA ZeroSSL przeglądarka wyświetla informację, iż nawiązane połączenie jest zabezpieczone.



Obraz 16.1 Informacja o zabezpieczonym połączeniu.



Obraz 16.2 Informacja o weryfikacji certyfikatu przez ZeroSSL.

17 Weryfikacja poprawności danych 2 pkt

Metod weryfikacji poprawności wprowadzonych danych na wybranych przykładach:

- 1 przykład jawnie użytego konwertera, o ile takowy występuje w projekcie

W naszym projekcie zamiast domyślnego jsonb, korzystamy z konwertera Jackson

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.14.2</version>
</dependency>
```

Przykładem modyfikacji działania konwertera Jackson poprzez adnotacje jest klasa *AccessLevelDTO*, gdzie wykorzystano adnotację *@JsonIgnore* do zaznaczenia metody jako ignorowanej.

Klasa AccessLevelDTO

```
(... importy zostaj pominitte)

@Override
@EqualsAndHashCode(callSuper = true)
@Getter
@Setter
@NoArgsConstructor
public abstract class AccessLevelDTO extends AbstractEntityDTO implements SignableEntity {

    public AccessLevelDTO(Long id, Long version, Role role, Boolean active) {
        super(id, version);
        this.role = role;
        this.active = active;
    }

    @NotNull
    private Role role;

    @NotNull
    private Boolean active;

    @Override
    @JsonIgnore
    public String getSignablePayload() {
        return String.format("%s.%d", role, getVersion());
    }
}
```

- 1 przykład weryfikacji przez metodę warstwy prezentacji, o ile takowa występuje w projekcie

Do walidacji danych po stronie warstwy prezentacji użyte zostały biblioteki *Yup* oraz *react-hook-form*.

Dokładny opis działania:

signUpSchema jest obiektem *Yup.object().shape()*, który definiuje reguły walidacji dla poszczególnych pól formularza. Każde pole ma określone wymagania dotyczące minimalnej i maksymalnej długości oraz inne specyficzne reguły walidacji, takie jak sprawdzanie czy adres e-mail jest poprawny, czy hasło spełnia określone kryteria (mały literę, wielką literę, cyfrę i znak specjalny).

Obiekt signUpSchema

```
export const signUpSchema = Yup.object().shape({
  name: Yup.string()
    .min(2, "first_name_length_min")
    .max(50, "first_name_length_max")
    .required("first_name_required"),
  lastName: Yup.string()
    .min(2, "last_name_length_min")
    .max(50, "last_name_length_max")
    .required("last_name_required"),
  login: Yup.string()
    .min(5, "login_length_min")
    .max(50, "login_length_max")
    .required("login_required"),
  email: Yup.string().email("email_valid").required("email_required"),
  password: Yup.string()
    .min(8, "password_length_min")
    .max(50, "password_length_max")
    .matches(
      /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/,
      "password_invalid"
    )
    .required("Password is required"),
  confirmPassword: Yup.string()
    .oneOf([Yup.ref("password"), null], "passwords_not_match")
    .required("confirm_password_required"),
  phoneNumber: Yup.string()
    .min(9, "phone_number_length")
    .max(9, "phone_number_length")
    .matches(/^[0-9]+$/, "phone_number_only_digits")
    .required("phone_number_required"),
  pesel: Yup.string()
    .min(11, "pesel_length")
    .max(11, "pesel_length")
    .matches(/^[0-9]+$/, "pesel_only_digits")
    .required("pesel_required"),
  nip: Yup.string()
    .min(10, "nip_length")
    .matches(/^[0-9]+$/, "nip_only_digits")
    .max(10, "nip_length")
    .required("nip_required"),
});
```

Następnie, przy użyciu *hooka useForm* z biblioteki *react-hook-form*, destrukturyzowane są wartości zwrócone przez ten *hook*, takie jak *register*, *handleSubmit* oraz *errors*, które będą wykorzystane do zarządzania formularzem.

```
const {
  register,
  handleSubmit,
  formState: { errors },
} = useForm({
  resolver: yupResolver(signUpSchema),
});
```

Funkcja *onSubmit* jest wywoływana po zatwierdzeniu formularza. W tej funkcji następuje wywołanie funkcji *signUpAccount* (odpowiedzialnej za wysłanie danych rejestracji do serwera) i przekazanie do niej danych z formularza. W przypadku powodzenia rejestracji, wyświetlane jest powiadomienie o sukcesie. W przypadku błędu, odpowiednie komunikaty są wyświetlane w zależności od otrzymanego statusu odpowiedzi serwera.

Funkcja onSubmit

```
const onSubmit = handleSubmit(({name, lastName, login, email, password, phoneNumber, pesel, nip}) => {

    setLoading(true)

    signUpAccount(name, lastName, login, email, password, phoneNumber, pesel, nip).then(
        () => {
            setLoading(false)
            toast.success(t("account_created_check_email"), {
                position: "top-center",
            })
        }
    ).catch(error => {
        setLoading(false)

        if (error.response.status === 400) {
            toast.error(t("invalid_account_data"), {
                position: "top-center",
            })
        } else if (error.response.status === 409) {
            toast.error(t(error.response.data.message), {
                position: "top-center",
            })
        } else {
            toast.error(t("server_error"), {
                position: "top-center",
            })
        }
    })
})
})
```

W renderowaniu komponentu formularza używane są komponenty `TextField` z biblioteki `Material-UI`, które są dostosowane do obsługi walidacji. Na przykładzie pola `name` można zauważyc, że atrybuty `error` i `helperText` są ustawiane na wartości odpowiednio `errors.name` i `t(errors.name?.message)`. Jeśli wystąpi błąd walidacji dla tego pola, to `errors.name` będzie miało wartość `true`, a `t(errors.name?.message)` zostanie użyte do wyświetlenia komunikatu o błędzie w odpowiednim języku. Podobnie jest to realizowane dla pozostałych pól formularza.

Fragment formularza rejestracji konta

```
<Stack spacing={4}>
    <Stack spacing={2} direction="row">
        <TextField
            {...register("name")}
            type="text"
            variant='outlined'
            color='secondary'
            label={t("name")}
            fullWidth
            required
            error={errors.name}
            helperText={t(errors.name?.message)}
        />
        <TextField
            type="text"
            variant='outlined'
            color='secondary'
            label={t("last_name")}
            fullWidth
            required
            error={errors.lastName}
            helperText={t(errors.lastName?.message)}
            {...register("lastName")}
        />
    </Stack>
    <TextField
```

```

        type="text"
        variant='outlined'
        color='secondary'
        label={t("login")}
        fullWidth
        required
        sx={{mb: 4}}
        error={errors.login}
        helperText={t(errors.login?.message)}
        {...register("login")}
    />
    <TextField
        type="email"
        variant='outlined'
        color='secondary'
        label={t("email")}
        fullWidth
        required
        sx={{mb: 4}}
        error={errors.email}
        helperText={t(errors.email?.message)}
        {...register("email")}
    />
    <TextField
        type={passwordShown ? "text" : "password"}
        variant='outlined'
        color='secondary'
        label={t("password")}
        required
        fullWidth
        error={errors.password}
        helperText={t(errors.password?.message)}
        sx={{mb: 4}}
        {...register("password")}
        InputProps={
            {
                endAdornment: <Button onClick={() => setPasswordShown(!passwordShown)}>
                    <VisibilityOffIcon fontSize="small" sx={{color: 'black'}}/> :
                    <VisibilityIcon fontSize="small" sx={{color: 'black'}}/>
                </Button>
            }
        }
    />
    <TextField
        type={confirmPasswordShown ? "text" : "password"}
        variant='outlined'
        color='secondary'
        label={t("confirm_password")}
        required
        fullWidth
        error={errors.confirmPassword}
        helperText={t(errors.confirmPassword?.message)}
        sx={{mb: 4}}
        {...register("confirmPassword")}
        InputProps={
            {
                endAdornment: <Button
                    onClick={() => setConfirmPasswordShown(!confirmPasswordShown)}>
                        <VisibilityOffIcon fontSize="small" sx={{color: 'black'}}/> :
                        <VisibilityIcon fontSize="small" sx={{color: 'black'}}/>
                    </Button>
            }
        }
    />
    <TextField
        type="text"
        variant='outlined'
        color='secondary'
        label={t("phone_number")}
        required
        fullWidth

```

```

        error={errors.phoneNumber}
        helperText={t(errors.phoneNumber?.message)}
        sx={{mb: 4}}
        {...register("phoneNumber")}
    />
    <TextField
        type="text"
        variant='outlined'
        color='secondary'
        label={t("pesel")}
        required
        fullWidth
        error={errors.pesel}
        helperText={t(errors.pesel?.message)}
        sx={{mb: 4}}
        {...register("pesel")}
    />
    <TextField
        type="text"
        variant='outlined'
        color='secondary'
        label={t("nip")}
        required
        fullWidth
        error={errors.nip}
        helperText={t(errors.nip?.message)}
        sx={{mb: 4}}
        {...register("nip")}
    />
    <Box sx={{mb: 2}} display="flex" justifyContent="center" alignItems="center">
        {loading ? <CircularProgress/> : (
            <Button fullWidth onClick={handleSubmit(onSubmit)} type='submit' variant='contained'>
                {t("sign_in")}
            </Button>
        )}
    </Box>
</Stack>

```

Przykład komunikatu o danych nie spełniających opisanych wyżej warunków:

Login *

f

Login must have at least 2 characters

Email *

invalid

Email is not valid

- 1 przykład weryfikacji przez metodę warstwy logiki, o ile takowa występuje w projekcie

Przykład weryfikacji przez metodę warstwy logiki znajduje się w metodzie "*checkIfTokenIsValid*", pochodzącej z klasy *TokenManager*, która sprawdza, czy token jest ważny i spełnia określone warunki.

Dokładny opis sprawdzanych warunków:

1. Pierwszy warunek: *if (token == null)*
 - Sprawdza, czy podany token nie ma wartości *null*.
 - Jeśli token ma wartość *null*, metoda rzuca wyjątek *TokenException.tokenNotFoundException* oznaczający brak znalezienia tokenu.
2. Drugi warunek: *if (token.getExpirationDate().before(new java.util.Date()))*
 - Sprawdza, czy data wygaśnięcia tokena jest wcześniejsza niż bieżąca data.
 - Jeśli token wygasł, metoda rzuca wyjątek *tokenExpiredException* oznaczający wygaśnięcie tokenu.
3. Trzeci warunek: *if (token.isUsed())*
 - Sprawdza, czy token został już użyty.
 - Jeśli token jest już użyty, metoda rzuca wyjątek *tokenAlreadyUsedException* oznaczający, że token został już wykorzystany.
4. Czwarty warunek: *if (type == TokenType.VERIFICATION && account.getConfirmed())*
 - Sprawdza, czy typ tokena to *TokenType.VERIFICATION* (weryfikacja) i czy konto jest potwierdzone.
 - Jeśli token jest typu weryfikacyjnego i konto jest już potwierdzone, metoda rzuca wyjątek *tokenAlreadyUsedException* oznaczający, że token został już użyty.
5. Piąty warunek: *if (token.getTokenType() != type)*
 - Sprawdza, czy typ tokena jest inny niż oczekiwany typ
(Weryfikacja konta, Resetowanie hasła, Potwierdzenie zmiany adresu email)
 - Jeśli typ tokena jest niezgodny z oczekiwany typem, metoda rzuca wyjątek *incorrectTokenTypeException* oznaczający, że typ tokena jest nieprawidłowy.

Przykład weryfikacji przez metodę warstwy logiki

```
private void checkIfTokenIsValid(Token token, TokenType type) {  
    if (token == null) {  
        throw TokenException.tokenNotFoundException();  
    }  
    Account account = token.getAccount();  
  
    if (token.getExpirationDate().before(new java.util.Date())) {  
        throw tokenExpiredException();  
    }  
    if (token.isUsed()) {  
        throw tokenAlreadyUsedException();  
    }  
    if (type == TokenType.VERIFICATION  
        && account.getConfirmed()) {  
        throw tokenAlreadyUsedException();  
    }  
    if (token.getTokenType() != type) {  
        throw incorrectTokenTypeException();  
    }  
}
```

- 1 przykład weryfikacji przez adnotację *Bean Validation*, o ile takowa występuje w projekcie

Kontynuując przykład procesu rejestracji, adnotacje *Bean Validation* zostały wykorzystane np. do weryfikacji pól tych klas *BasicAccountDto* oraz *RegisterPatientDTO* (dzi edziczącej po poprzedniej klasie).

Dokładny opis przykładu weryfikacji dla wybranych pól:

1. Pole login w klasie *BasicAccountDto*:

- `@Size(max = 50, min = 5)` - Wymaga, aby wartość tego pola mieściła się w zakresie od 5 do 50 znaków.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.

2. Pole `password` w klasie `BasicAccountDto`:

- `@Pattern(regexp = "^(?=.*[a-z])(?=.*[A-Z])(?=.*\\d)(?=.*[@$!%*?&])[A-Za-z\\d@$!%*?&]{8,}$")` - Wymaga, aby wartość tego pola spełniała określony wzorzec, który jest wymaganym formatem hasła. Musi zawierać co najmniej jedną małą literę, jedną wielką literę, jedną cyfrę, jeden znak specjalny i mieć długość od 8 znaków.
- `@Size(max = 50, min = 8)` - Wymaga, aby wartość tego pola mieściła się w zakresie od 8 do 50 znaków.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.

3. Pole `email` w klasie `BasicAccountDto`:

- `@Email` - Wymaga, aby wartość tego pola była poprawnym adresem e-mail.
- `@Size(max = 50, min = 5)` - Wymaga, aby wartość tego pola mieściła się w zakresie od 5 do 50 znaków.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.

4. Pole `name` w klasie `RegisterPatientDTO`:

- `@Size(max = 50, min = 2)` - Wymaga, aby wartość tego pola mieściła się w zakresie od 2 do 50 znaków.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.
- nie miała wartości `null`.

5. Pole `phoneNumber` w klasie `RegisterPatientDTO`:

- `@Pattern(regexp = "^\d{9}$", message = "Invalid phone number")` - Wymaga, aby wartość tego pola spełniała określony wzorzec, który jest wymaganym formatem numeru telefonu. Musi składać się z dokładnie 9 cyfr.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.

6. Pole `pesel` w klasie `RegisterPatientDTO`:

- `@Pattern(regexp = "^\d{11}$", message = "Invalid license number")` - Wymaga, aby wartość tego pola spełniała określony wzorzec, który jest wymaganym formatem numeru PESEL. Musi składać się z dokładnie 11 cyfr.
- `@NotNull` - Wymaga, aby wartość tego pola nie miała wartości `null`.

Klasa BasicAccountDto

```

@Data
@NoArgsConstructor
@AllArgsConstructor
public class BasicAccountDto {

    @Size(max = 50, min = 5)
    @NotNull
    private String login;

    @NotNull
    @Pattern(regexp = "^(?=.*[a-z])(?=.*[A-Z])(?=.*\\d)(?=.*[@$!%*?&])[A-Za-z\\d@$!%*?&]{8,}$")
    @Size(max = 50, min = 8)
    private String password;

    @NotNull
    @Email
    @Size(max = 50, min = 5)
    private String email;

    private String language;
}

```

Klasa RegisterPatientDTO rozszerzająca klasę BasicAccountDto

```
@Data
@NoArgsConstructor
@AllArgsConstructor
public class RegisterPatientDTO extends BasicAccountDto {

    @NotNull
    @Size(max = 50, min = 2)
    private String name;

    @NotNull
    @Size(max = 50, min = 2)
    private String lastName;

    @NotNull
    @Pattern(regexp = "^\d{9}$", message = "Invalid phone number")
    private String phoneNumber;

    @NotNull
    @Pattern(regexp = "^\d{11}$", message = "Invalid license number")
    private String pesel;

    @NotNull
    @Pattern(regexp = "^\d{10}$", message = "Invalid NIP")
    private String nip;

    @Builder
    public RegisterPatientDTO(
        @NotNull String login,
        @NotNull String password,
        @NotNull String email,
        String language,
        String name,
        String lastName,
        String phoneNumber,
        String pesel,
        String nip) {
        super(login, password, email, language);
        this.name = name;
        this.lastName = lastName;
        this.phoneNumber = phoneNumber;
        this.pesel = pesel;
        this.nip = nip;
    }
}
```

18 Obsługa błędów 2 pkt

Zaprezentować sposób informowania użytkownika o błędach pojawiających się w warstwie logiki poprzez przedstawienie przykładowych segmentów obsługi. W wyżej zamieszczonym kodzie przechwytywane są błędy zwracane przez logowanie do systemu. Najpierw porównywany jest kod błędu, a następnie na podstawie wiadomości zawartej w błędzie, wyświetlany jest komunikat, w odpowiedniej wersji językowej, informujący o błędzie.

- przykładowego wyjątku aplikacyjnego (biznesowego)

OrderManager.java

```
(...)
@POST
@Path("/login")
@PermitAll
public Response authenticate(@Valid LoginDTO loginDto) {
    Account account;
    try {
        account = repeatTransaction(accountManager, () -> accountManager.findByLogin(loginDto.getLogin()));
    } catch (ApplicationExceptionEntityNotFound e) {
        throw AuthApplicationException.createInvalidLoginOrPasswordException();
    }

    if (!account.getConfirmed()) {
        throw AccountApplicationException.createAccountNotConfirmedException();
    }
}

(...)
```

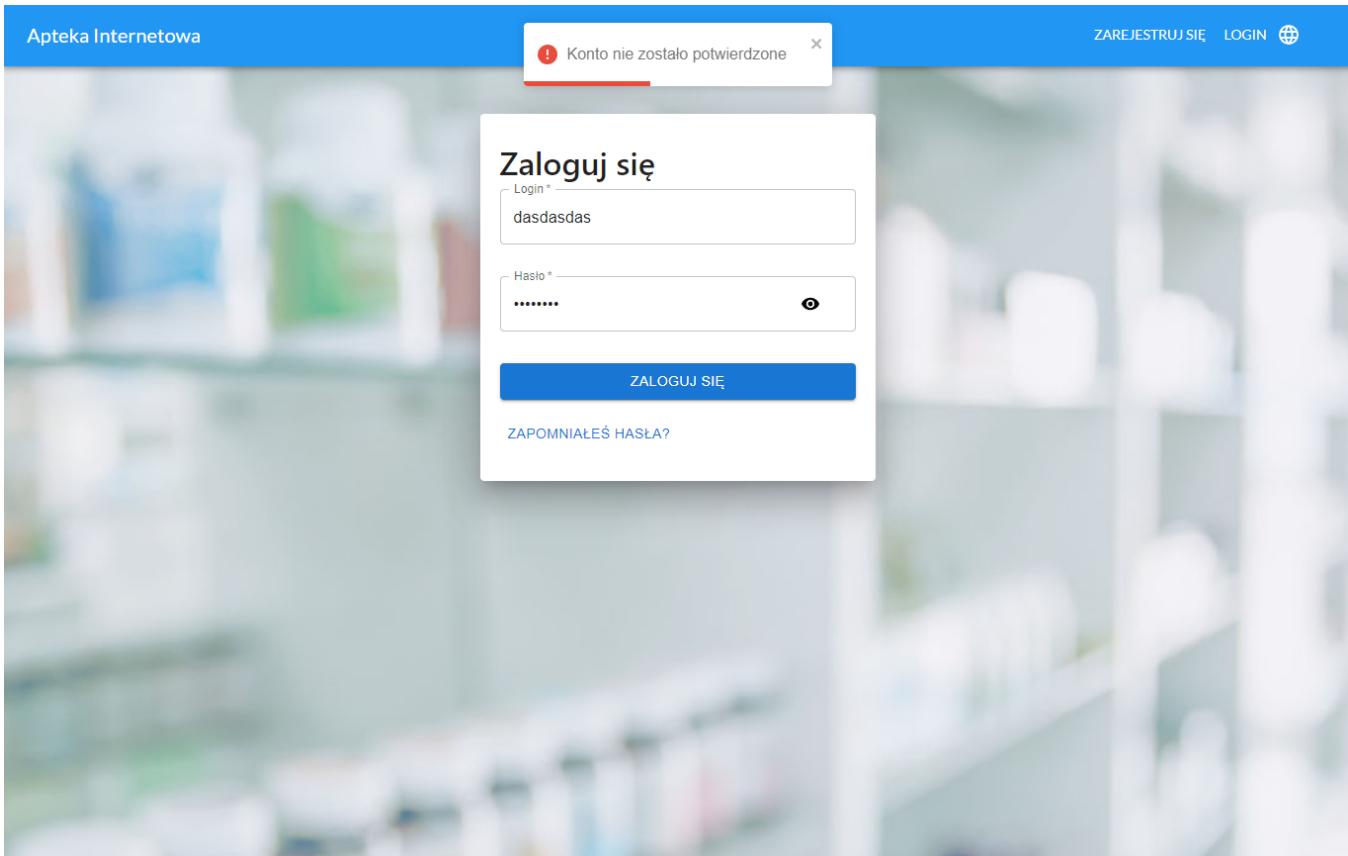
W wyżej zamieszczonym kodzie, w 9 linii przechwytywany jest wyjątek `ApplicationExceptionEntityNotFound`, który następnie jest mapowany na odpowiedź(`Response`) dopasowaną do działania aplikacji po stronie widoku (zawierają odpowiedni "message", na podstawie którego dobierany jest odpowiedni komunikat widoczny dla użytkownika, oraz "code" zawierający kod błędu).

Login.js

```
(...)
).catch((error) => {
    if (error.response.status === 403) {
        toast.error(t(error.response.data.message), {
            position: toast.POSITION.TOP_CENTER,
        });
        setLoading(false)

    } else if (error.response.status === 401) {
        toast.error(t(error.response.data.message), {
            position: toast.POSITION.TOP_CENTER,
        });
        setLoading(false)
    } else {
        toast.error(t("server_error"), {
            position: toast.POSITION.TOP_CENTER,
        });
        setLoading(false)
    }
})
(...)
```

W wyżej zamieszczonym kodzie, warstwa widoku przechwytuje zwracaną odpowiedź. Następnie sprawdza zwrócony kod błędu oraz wyświetla okno błędu z odpowiednim komunikatem.



- przykładowego wyjątku systemowego (zgłoszonego przez kontener EJB)

MedicationManager.java

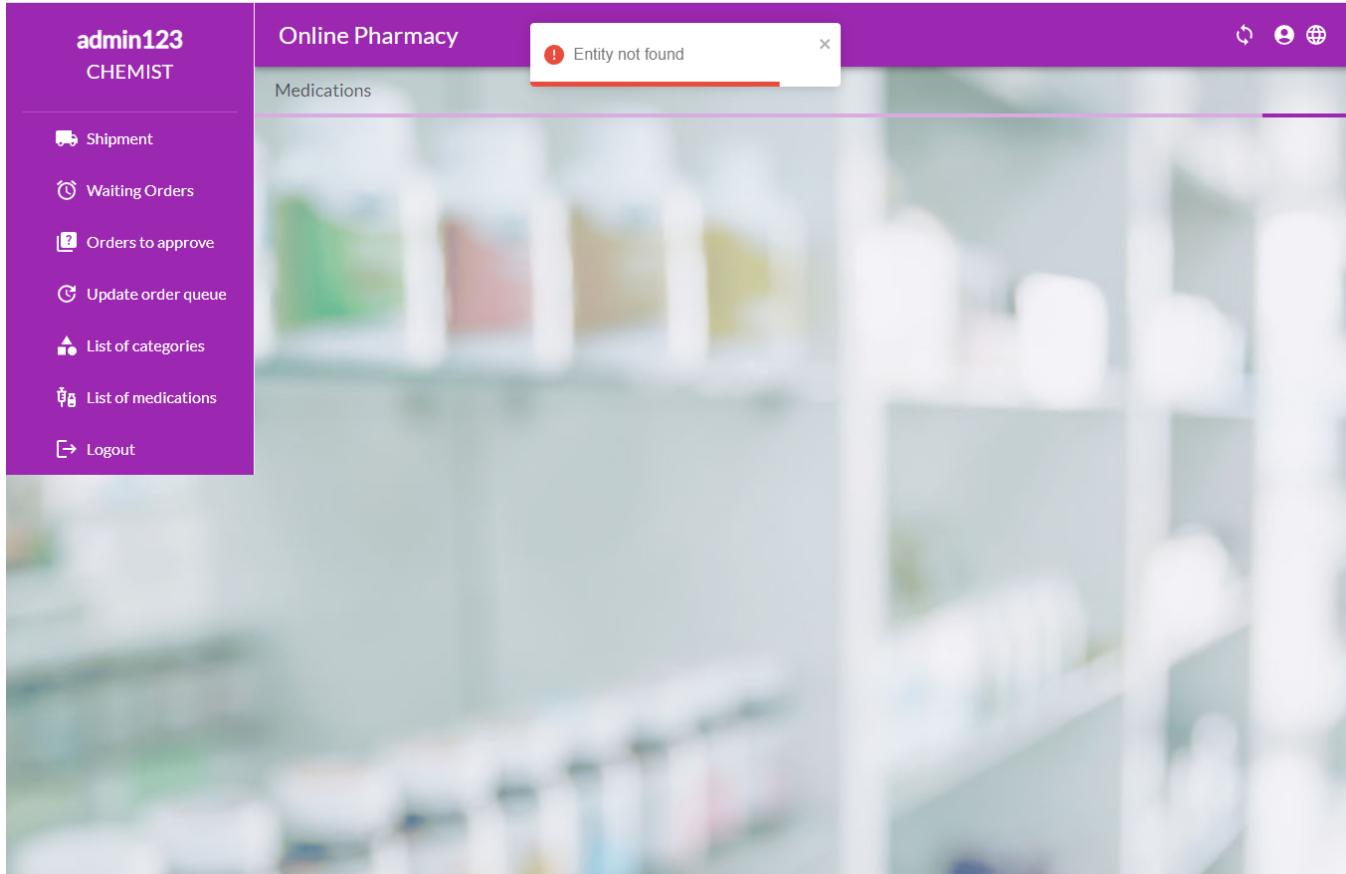
```
(...)
    @Override
    @RolesAllowed("getMedicationDetails")
    public Medication getMedicationDetails(Long id) {
        Optional<Medication> medication = medicationFacade.findAndRefresh(id);
        if (medication.isEmpty()) {
            throw ApplicationException.createEntityNotFoundException();
        }
        return medication.get();
    }
(...)
```

W wyżej zamieszczonym kodzie, w 6 linii sprawdzany jest warunek, czy szukany lek został znaleziony, jeśli nie, w następnej linii mapowany jest on na odpowiedź (Response) dla warstwy widoku (zawierają odpowiedni "message", na podstawie którego dobierany jest odpowiedni komunikat widoczny dla użytkownika, oraz "code" zawierający kod błędu).

MedicationDetails.js

```
(...)
    .catch((error) => {
        console.log(error.response.data.message)
        toast.error(t(error.response.data.message), { position: "top-center" });
    })
(...)
```

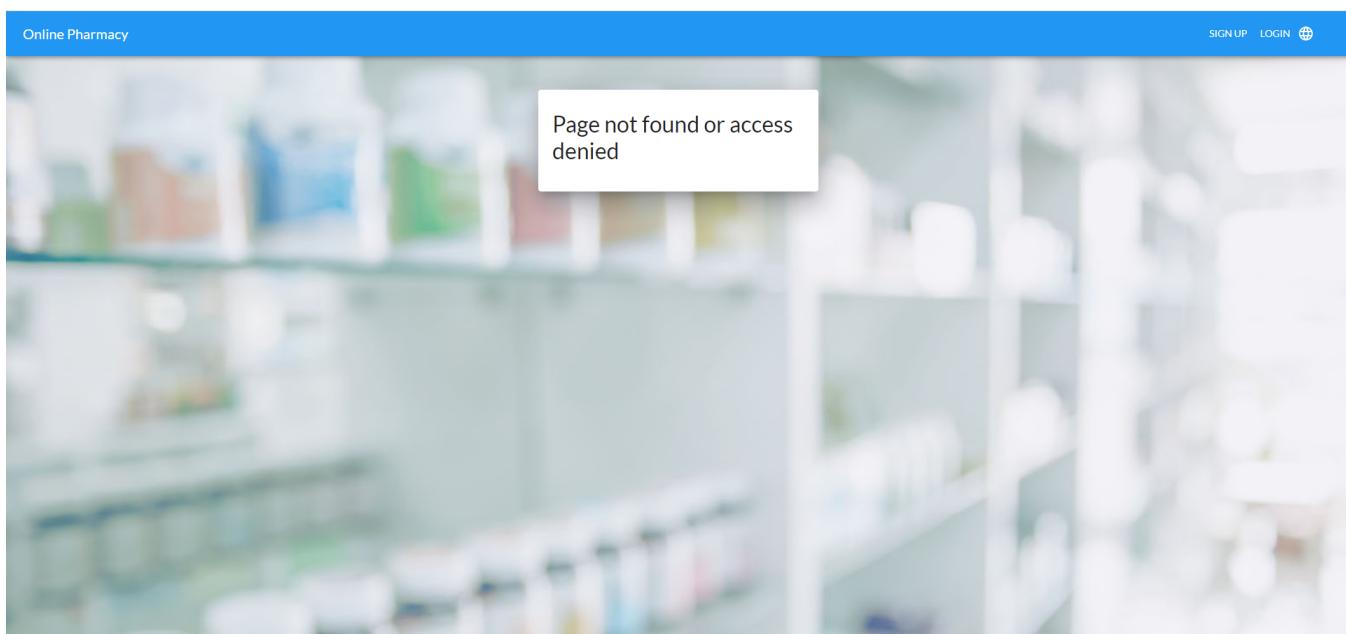
Wyżej wymieniony kod przechwytuje odpowiedź z warstwy logiki i wyświetla odpowiedni komunikat



Opisać konfigurację reakcji na standardowe błędy zgłoszane przez serwer aplikacji dla wybranego kodu błędu HTTP (np. 401 - żądanie nie autoryzowane).

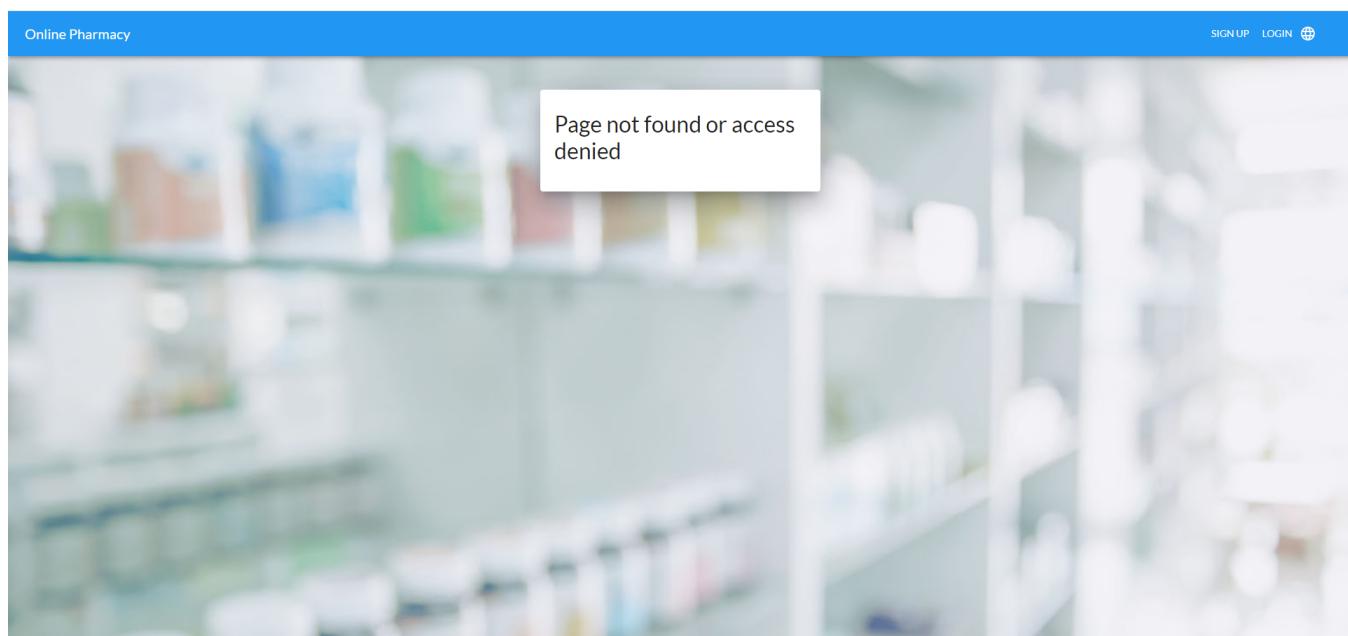
Wybrany błąd: **401**

W razie wystąpienia błędu 401, reakcją aplikacji będzie wyświetlenie domyślnej strony błędu:



Opisać konfigurację domyślnej reakcji na błędy wykonania aplikacji nie przewidziane przez mechanizmy opisane powyżej (tzw. domyślna strona błędu).

W razie wystąpienia błędu 404, reakcją aplikacji będzie wyświetlenie domyślnej strony błędu:



19 Wersje językowe interfejsu użytkownika 1 pkt

Przedstawić konfigurację zastosowanych mechanizmów zapewniających internacjonalizację interfejsu użytkownika oraz powiadomień e-mail wysyłanych przez aplikację względem preferencji językowych użytkownika. Należy zaprezentować gdzie w profilu konta użytkownika przechowywana jest wersja językowa.

Na przykładzie odpowiedniego kodu należy też wyjaśnić dobór wersji językowej dla:

- komunikatu związanego z błędym formatem danych (komunikatu błędu validatora)
- komunikatu przedstawianego użytkownikowi w sytuacji obsługi wyjątku zgłoszonego w warstwie logiki
- wyświetlanego poziomu dostępu dla uwierzytelnionego użytkownika

Wymagane jest zdefiniowanie w aplikacji co najmniej dwóch wersji językowych: polskiej oraz domyślnej (bez kodu języka). Wersja domyślna może przy tym definiować tylko jeden komunikat tak, aby była możliwość zweryfikowania działania mechanizmu - pozostałe komunikaty mogą w tej wersji zostać wyświetlone w postaci nie odnalezionych kluczy.

W klasie encyjnej account znajduje się pole language, które jest typu Locale.

```
@Basic(optional = false)
private Locale language;
```

19.1. Pole language z klasy encyjnej Account

account						
WHERE ORDER BY						
	on	active	confirmed	email	incorrect_login_attempts	language
1	0	• true	• true	chemist@o2.pl		0 en
2	0	• true	• true	patient@o2.pl		0 en
3	7	• true	• true	admin@o2.pl		0 en

19.2. Tabela account w bazie danych - kolumna language wskazująca na język

W warstwie logiki znajduje się klasa i18n, definiuje zestaw stałych reprezentujących klucze do zasobów w plikach lokalizacyjnych.

Fragment klasy i18n

```
package pl.lodz.p.it.ssbd2023.ssbd01.common;

import java.util.Locale;
import java.util.ResourceBundle;

public class i18n {
    public static final String EXCEPTION_UNKNOWN = "exception.unknown";
    public static final String EXCEPTION_GENERAL = "exception.general";
    public static final String EXCEPTION_PERSISTENCE = "exception.persistence";
    public static final String EXCEPTION_ACCESS_DENIED = "exception.access-denied";
    public static final String EXCEPTION_UNAUTHORISED = "exception.unauthorised";
    public static final String EXCEPTION_PASSWORD_NOT_CHANGED = "exception.password.not-changed";
    public static final String EXCEPTION_NOT_FOUND = "exception.not-found";
    public static final String EXCEPTION_LANGUAGE_NOT_FOUND = "exception.language.not-found";
    public static final String EXCEPTION_METHOD_NOT_ALLOWED = "exception.method-not-allowed";
    public static final String EXCEPTION_MISMATCHED_PAYLOAD = "exception.mismatched.payload";
    public static final String EXCEPTION_ETAG_CREATION = "exception.etag.creation";
    public static final String EXCEPTION_ACCOUNT_CONSTRAINT_VIOLATION =
        "exception.account.constraint-violation";
    public static final String EXCEPTION_ACCOUNT_NOT_CONFIRMED =
        "exception.account.not-confirmed";
    public static final String EXCEPTION_ACCOUNT_NO SUCH ACCESS LEVEL =
        "exception.account.no-such-access-level";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_ACCESS_LEVEL =
        "exception.account.duplicate.access-level";
    public static final String EXCEPTION_ACCOUNT_DEACTIVATE_LAST_ACCESS_LEVEL =
        "exception.account.deactivate.last-access-level";
    public static final String EXCEPTION_ACCOUNT_DEACTIVATE_SELF =
        "exception.account.deactivate.self";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_EMAIL =
        "exception.account.duplicate.email";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_LOGIN =
        "exception.account.duplicate.login";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_PHONE_NUMBER =
        "exception.account.duplicate.phone";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_NIP =
        "exception.account.duplicate.nip";
    public static final String EXCEPTION_ACCOUNT_DUPLICATE_PESEL =
        "exception.account.duplicate.pesel";
    public static final String EXCEPTION_AUTH_BAD_CREDENTIALS = "exception.auth.bad-credentials";
    public static final String EXCEPTION_AUTH_BLOCKED_ACCOUNT = "exception.auth.blocked-account";

    public static final String EXCEPTION_TOKEN_EXPIRED = "exception.token.expired";
    public static final String EXCEPTION_TOKEN_NOT_FOUND = "exception.token.not-found";
    public static final String EXCEPTION_TOKEN_ALREADY_USED = "exception.token.already-used";
    public static final String EXCEPTION_TOKEN_BAD_TYPE = "exception.token.bad-type";

    public static final String MAIL_ACCOUNT_REMOVED SUBJECT = "mail.account.removed.subject";
    public static final String MAIL_ACCOUNT_REMOVED_BODY = "mail.account.removed.body";
    public static final String MAIL_ACCOUNT_BLOCKED SUBJECT = "mail.account.blocked.subject";
    public static final String MAIL_ACCOUNT_BLOCKED_BODY = "mail.account.blocked.body";
    public static final String MAIL_ACCOUNT_UNBLOCKED SUBJECT = "mail.account.unblocked.subject";
    public static final String MAIL_ACCOUNT_UNBLOCKED_BODY = "mail.account.unblocked.body";
    public static final String MAIL_ACCOUNT_BLOCKED_TOO_MANY_LOGINS_BODY =
        "mail.account.unblocked.too-many-logins.body";
    public static final String MAIL_ACCOUNT_ACTIVATED SUBJECT = "mail.account.activated.subject";
    public static final String MAIL_ACCOUNT_ACTIVATED_BODY = "mail.account.activated.body";
    public static final String MAIL_ACCOUNT_REGISTER SUBJECT = "mail.account.register.subject";
    public static final String MAIL_ACCOUNT_REGISTER_BODY = "mail.account.register.body";
```

Fragment klasy EmailService, która używa funkcji getMessage() z klasy i18n, do której przekazujemy między innymi to jaką wiadomość mamy wysłać oraz język w którym będzie dana wiadomość

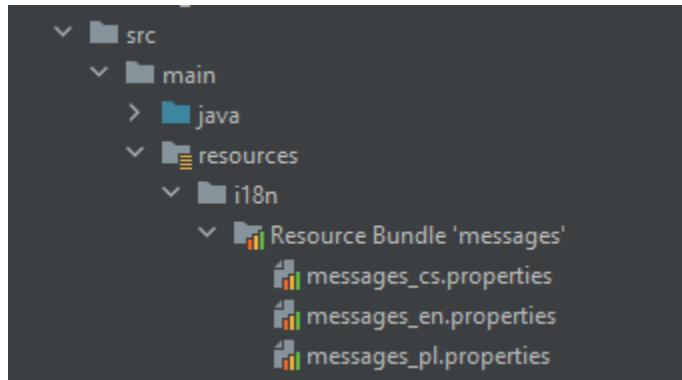
Fragment klasy EmailService

```
public void sendEmailAccountBlocked(String email, String name, Locale locale) {
    String subject = i18n.getMessage(i18n.MAIL_ACCOUNT_BLOCKED SUBJECT, locale);
    String body = i18n.getMessage(i18n.MAIL_ACCOUNT_BLOCKED_BODY, locale, name);

    MailjetRequest request = getMailjetRequest(email, name, subject, body);
    try {
        client.post(request);
    } catch (MailjetException e) {
        e.printStackTrace();
    }
}

public void sendEmailAccountUnblocked(String email, String name, Locale locale) {
    String subject = i18n.getMessage(i18n.MAIL_ACCOUNT_UNBLOCKED SUBJECT, locale);
    String body = i18n.getMessage(i18n.MAIL_ACCOUNT_UNBLOCKED_BODY, locale, name);

    MailjetRequest request = getMailjetRequest(email, name, subject, body);
    try {
        client.post(request);
    } catch (MailjetException e) {
        e.printStackTrace();
    }
}
```



19.3. Wycinek Intellij, pokazanie zasobów i18n, które zawierają mapowanie kluczy na języki.

Fragment zasobu messages_pl.properties

```
mail.account.removed.subject=Twoje konto zosta\u0142o usuni\u0119te
mail.account.removed.body=Szanowny(a) %s, Twoje konto zosta\u0142o usuni\u0119te z powodu braku aktywacji.
Prosz\u0119 spr\u00f3buj utworzy\u0107 nowe konto.
mail.account.blocked.subject=Twoje konto zosta\u0142o zablokowane
mail.account.blocked.body=Szanowny(a) %s, Z przykro\u015bcie informujemy, \u017ce Twoje konto zosta\u0142o zablokowane.
mail.account.unblocked.subject=Twoje konto zosta\u0142o odblokowane
mail.account.unblocked.body=Szanowny(a) %s, Twoje konto zosta\u0142o odblokowane. Mo\u017cesz ponownie si\u0119 zalogowa\u0107.
mail.account.unblocked.too-many-logins.body=Szanowny(a) %s, Twoje konto zosta\u0142o zablokowane z powodu zbyt wielu pr\u00f3b logowania.\n
Je\u015bli to nie by\u0142e u015b/-a\u015b Ty, skontaktuj si\u0119 niezw\u0142ocznie z dzia\u0142em IT.

mail.account.activated.subject=Twoje konto zosta\u0142o aktywowane
mail.account.activated.body=Szanowny(a) %s, Twoje konto zosta\u0142o aktywowane. Mo\u017cesz teraz si\u0119 zalogowa\u0107.
mail.account.register.subject=Aktywacja konta
mail.account.register.body=Witamy w Aptece Online, drogi(a) %s! Aby aktywowa\u0107 swoje konto, kliknij poni\u017cszy link. B\u0119dzie on wa\u017cny przez kolejne 24 godziny: %s

mail.password.reset.subject=Zresetuj swoje has\u0142o
mail.password.reset.body=Szanowny(a) %s, kliknij %s, aby zresetowa\u0107 has\u0142o
mail.email.change.body=Szanowny(a) %s kliknij %s, aby zatwierdzic email
mail.email.change.subject=Zresetuj swoj email
```

Fragment zasobu messages_en.properties

```
mail.account.removed.subject=Your account has been deleted
mail.account.removed.body=Dear %s, your account has been deleted due to lack of activation, please try creating another one.
mail.account.blocked.subject=Your account has been blocked.
mail.account.blocked.body=Dear %s, We are sorry to inform that your account has been blocked.
mail.account.unblocked.subject=Your account has been unblocked.
mail.account.unblocked.body=Dear %s, your account has been unblocked. You can log in again.
mail.account.unblocked.too-many-logins.body=Dear %s, your account has been blocked due to too many login attempts.\n
If that wasn't you please contact the IT department forthwith.

mail.account.activated.subject=Your account has been activated.
mail.account.activated.body=Dear %s, Your account has been activated. You can log in now.
mail.account.register.subject=Account activation
mail.account.register.body=Welcome to Online Pharmacy, dear %s! To activate your click the following link, it'll be valid for the next 24 hours: %s

mail.password.reset.subject=Reset your password
mail.password.reset.body=Dear %s, click %s to reset your password
mail.email.change.body=Dear %s, click %s to validate your email
mail.email.change.subject=Reset of your email
```

Metoda managera odpowiedzialna za zmianę języka

```

@Override
@RolesAllowed("changeAccountLanguage")
public void changeAccountLanguage(String language) {
    try {
        LanguageType.valueOf(language);
    } catch(InvalidArgumentException e) {
        throw ApplicationException.createLanguageNotFoundException();
    }
    Locale locale = Locale.forLanguageTag(language);
    Account account = getCurrentUser();
    account.setLanguage(locale);
    accountFacade.edit(account);
}

```

19.4. Metoda changeAccountLanguage z klasy AccountManager

Implementacja zmiany języka w warstwie widoku

```

<LanguageIcon/>
<Menu
  id="basic-menu"
  anchorEl={anchorElLanguage}
  open={Boolean(anchorElLanguage)}
  onClose={(e) => {
    e.stopPropagation();
    setAnchorElLanguage(null);
  }}
  menuListProps={{
    'aria-labelledby': 'basic-button',
  }}
>
  {languages.map(({code, name, country_code}) => (
    <MenuItem disabled={code === currentLanguage} key={country_code} onClick={() => [
      changeLanguage(code),
      i18n.changeLanguage(code),
      setAnchorElLanguage(null),
    ]}>
      {name}
    </MenuItem>
  )));
</Menu>

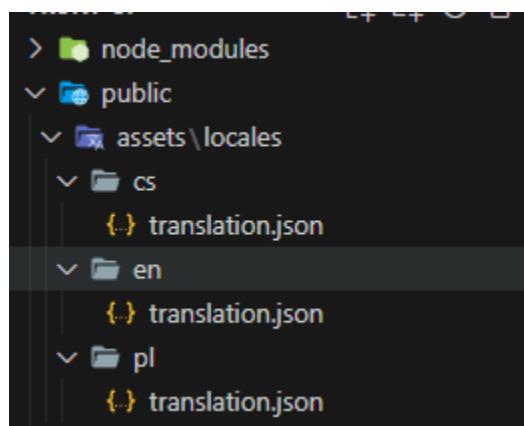
```

19.5. Skrawek kodu z funkcji AuthNavbar

Konfiguracja i18n w warstwie widoku, aby zmienić język sprawdzane są ciasteczka. Wskazujemy również ścieżkę do plików translation.json, gdzie znajdują się pary klucz - wartość dla tłumaczeń.

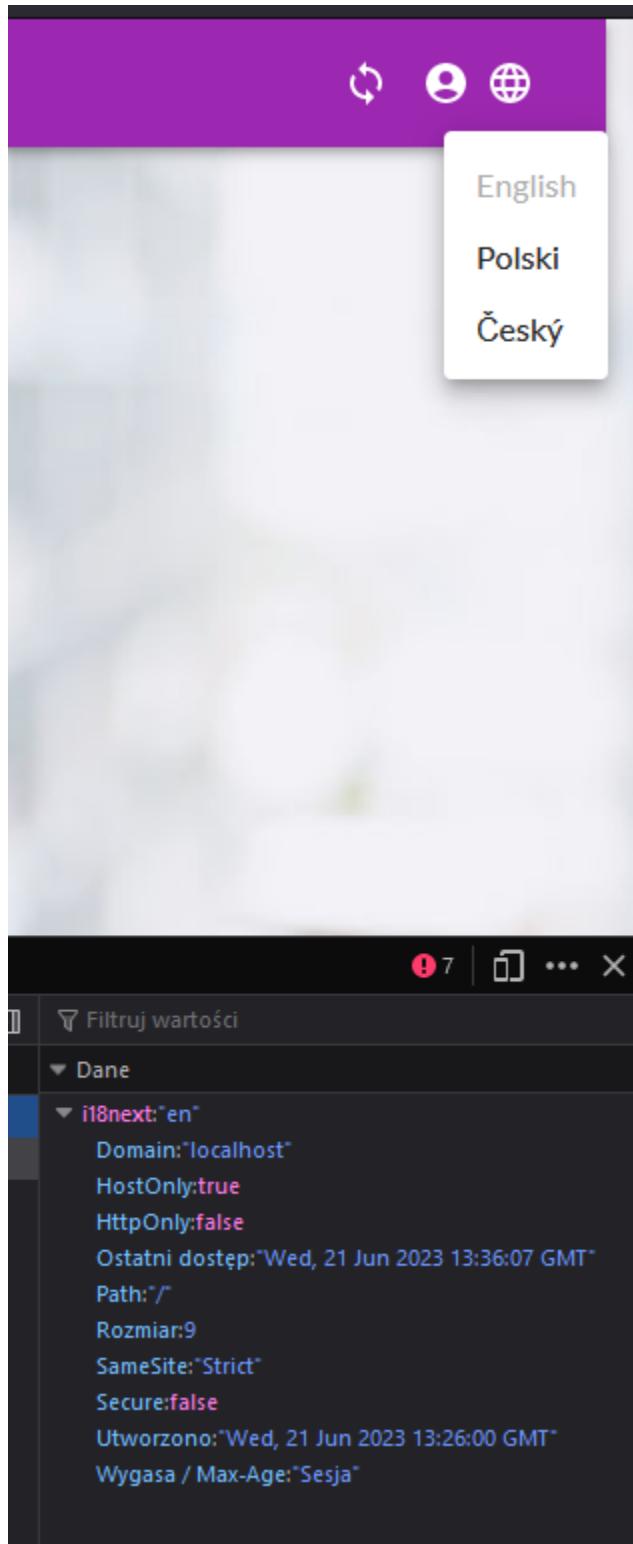
```
i18n
  .use(initReactI18next) // passes i18n down to react-i18next
  .use(LanguageDetector)
  .use(HttpApi)
  .init({
    supportedLngs: ["en", "pl", "cs"],
    backend: {
      loadPath: "/assets/locales/{{lng}}/translation.json",
    },
    fallbackLng: "en",
    detection: {
      order: [
        "cookie",
        "htmlTag",
        "localStorage",
        "sessionStorage",
        "navigator",
        "path",
        "subdomain",
      ],
      caches: ["cookie"],
    },
  },
```

19.6. Skrawek kodu z pliku App.js



19.7. Struktura plików translation.js dla danych języków

Aktywny język jest wyszarzony. Dane o używanym języku znajdują się w ciasteczku i18next.



19.8. Wykazanie zmiany języka w interfejsie użytkownika i ciasteczek dotyczących języka

Wykazanie, że informacje o weryfikacji danych są zależne od wybranego języka

X ZAMKNIJ

Utwórz nowy lek

Imię *

Imię musi mieć co najmniej 2 znaki

Ilość * 5

Cena * 43

Nazwa kategorii * Kategoria

DODAJ LEK

19.9. Walidacja pola imię - język ustawiony na polski

X CLOSE

Create new medication

Name *

Name must have at least 2 characters

Stock * 5

Price * 43

Category name * Kategoria

ADD MEDICATION

19.10. Walidacja pola imię - język ustawiony na angielski

Wycinek z kodu dotyczącego weryfikacji dodania leku - w cudzysłowie znajdują się klucze, które są przypisywane do wartości w plikach translation.json

```

    ✓ export const medicationSchema = Yup.object().shape({
      name: Yup.string()
        .min(2, "name_length_min")
        .max(50, "name_length_max")
        .required("name_required"),
      stock: Yup.number()
        .min(1, "stock_min")
        .max(1000000, "stock_max")
        .required("stock_required"),
      price: Yup.number()
        .min(0.01, "price_min")
        .max(1000000, "price_max")
        .required("price_required"),
      categoryName: Yup.string()
        .min(2, "category_name_length_min")
        .max(50, "category_name_length_max")
        .required("category_name_required"),
    });
  
```

19.11. Ustawienia walidacji dodawania leku z pliku Validation.js

Wycinek żądania - w odpowiedzi dla pobrania detali konta otrzymujemy język konta użytkownika

Stan	Metoda	Domena	Plik	Inicjator	Typ	Przesłano	Rozmiar
200	POST	localhost:8080	login	bundle:ic:141232 (xhr)	json	699 B	159 B
200	POST	localhost:8080	CHEMIST	bundle:ic:141232 (xhr)	xml	506 B	0 B
200	GET	localhost:8080	details	bundle:ic:141232 (xhr)	json	1 kB	371 B

Wyszukaj właściwość: Nieprzetworzone


```

    id: 1
    version: 8
    active: true
    confirmed: true
    email: "admin@o2.pl"
    login: "admin123"
    signablePayload: "admin123.8"
    accessLevel: [<--, <--]
    language: "en"
  
```

19.12. Wykazanie, że odpowiedź żądania zwraca pole z językiem

Następnie zmieniamy język aplikacji w zależności od tego jaki język znajduje się w polu language(język)

```

useEffect(() => {
  const fetchData = async () => {
    try {
      const response = await getSelfAccountDetails();
      if (response.data.language === "en") {
        i18n.changeLanguage("en");
      } else if (response.data.language === "pl") {
        i18n.changeLanguage("pl");
      } else if (response.data.language === "cs") {
        i18n.changeLanguage("cs");
      } else {
        i18n.changeLanguage("en");
      }
    } catch (error) {
      console.error(error);
    }
  };
  fetchData();
}, []);

```

19.13. Obsługa ustawienia języka z odpowiedzi na żądanie get

Zależność klucz - wartość w pliku translation.json - polski oraz angielski

```

public > assets > locales > pl > {} translation.json > ...
254   "bad_prescription_number": "Zły numer",
255   "bought_successfully": "Zakup udany!",
256   "exception.order.prescription.already-
257   "exception.order.prescription.required-
258   "CHEMIST": "Farmaceuta",
259   "PATIENT": "Pacjent",
260   "ADMIN": "Administrator"

```

19.14. Przykładowe tłumaczenie poziomów dostępu - język polski

```

public > assets > locales > en > {} translation.json
254   "bad_prescription_number": "Bad
255   "bought_successfully": "Bought
256   "exception.order.prescription.already-
257   "exception.order.prescription.required-
258   "CHEMIST": "Chemist",
259   "PATIENT": "Patient",
260   "ADMIN": "Admin"
261 }
262

```

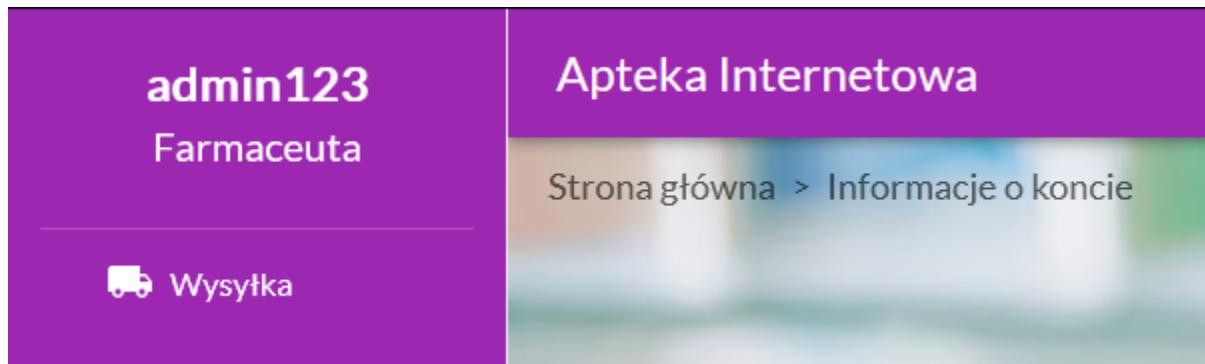
19.15. Przykładowe tłumaczenie poziomów dostępu - język angielski

Wyświetlanie roli użytkownika z panela bocznego

```
return (
  <>
  <Sidebar
    className="text-white"
    backgroundColor={theme.palette.primary.main}
  >
    <Menu iconShape="square">
      <Box mb="25px">
        <Box textAlign="center">
          <Typography
            variant="h5"
            fontWeight="bold"
            sx={{ m: "20px 0 0 0" }}
          >
            {login}
          </Typography>
          <Typography variant="h6">{t(userRole)}</Typography>
        </Box>
      </Box>
    </Menu>
  </Sidebar>
)
```

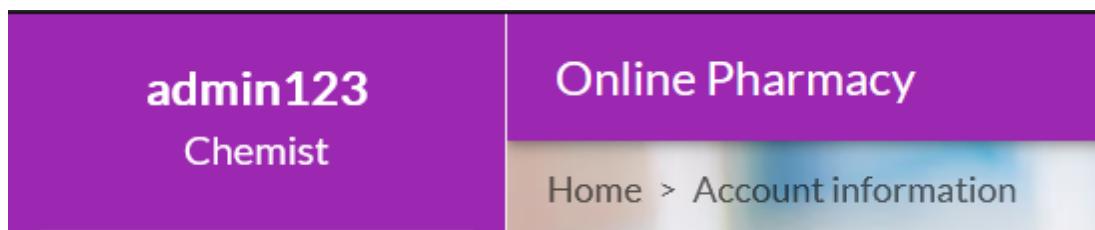
19.16. Fragment kodu przedstawiający używanie zmiennej `userRole`, która przechowuje poziom dostępu użytkownika, jako klucza do tłumaczenia

Przykład roli konta dla języka polskiego



19.17. Tłumaczenie roli użytkownika - język polski

Przykład roli konta dla języka angielskiego



19.18. Tłumaczenie roli użytkownika - język angielski

Na fragmencie kodu widoczne jest obsługuwanie wyjątków z warstwy logiki - dokładnie wtedy, kiedy status odpowiedzi ma wartość 409. Otrzymany błąd jest traktowany klucz i tak jak reszta kluczy nadawana jest mu wartość w plikach `translation.json`

Fragment kodu - obsługa błędów przy dodawaniu

```
const doSubmit = function({ name, isOnPrescription }) {
    setLoading(true);

    createCategory(name, isOnPrescription)
        .then(() => {
            setLoading(false);
            toast.success(t("category_created"), {
                position: "top-center",
            });
        })
        .catch((error) => {
            setLoading(false);

            if (error.response.status === 400) {
                toast.error(t("invalid_category_data"), {
                    position: "top-center",
                });
            } else if (error.response.status === 409) {
                toast.error(t(error.response.data.message), {
                    position: "top-center",
                });
            } else {
                toast.error(t("server_error"), {
                    position: "top-center",
                });
            }
        });
    onClose();
}
```

Dlatego otrzymujemy zależne od języka wyskakujące okna z błędami w języku, które ma ustalone dane konto

The screenshot shows a mobile application interface. At the top, there is a purple header bar. Below it, a white alert box with a red border and a red exclamation mark icon displays the text "Category already exists". In the background, there is a blurred list of categories. At the top of this list are two icons: a circular icon with a minus sign and a plus sign. The main list has a purple header row with three columns: "Name", "Is on prescription?", and "Edit". Below this, there are seven rows of data, each containing a category name, its status, and an "EDIT" button.

Name	Is on prescription?	Edit
painkillers	No	EDIT
antidepressant	Yes	EDIT
vitamins	No	EDIT
antibioticsS	Yes	EDIT
sda	Yes	EDIT
rew	Yes	EDIT
dawdaw	Yes	EDIT
dasdasdasdasd	Yes	EDIT

19.19. Obsługa błędu dodania tej samej kategorii - język angielski

Kategoria już istnieje

Imię	Na receptę?	Edytuj
painkillers	Nie	EDYTUJ
antidepressant	Tak	EDYTUJ
vitamins	Nie	EDYTUJ
antibioticsS	Tak	EDYTUJ
sda	Tak	EDYTUJ
rew	Tak	EDYTUJ
dawdaw	Tak	EDYTUJ
dasdasdasdasd	Tak	EDYTUJ

19.20. Obsługa błędu dodania tej samej kategorii - język polski

20 Wykaz akcji dostępnych z interfejsu użytkownika 4 pkt

Role mające prawo do uruchamiania akcji dla danego przypadku użycia.

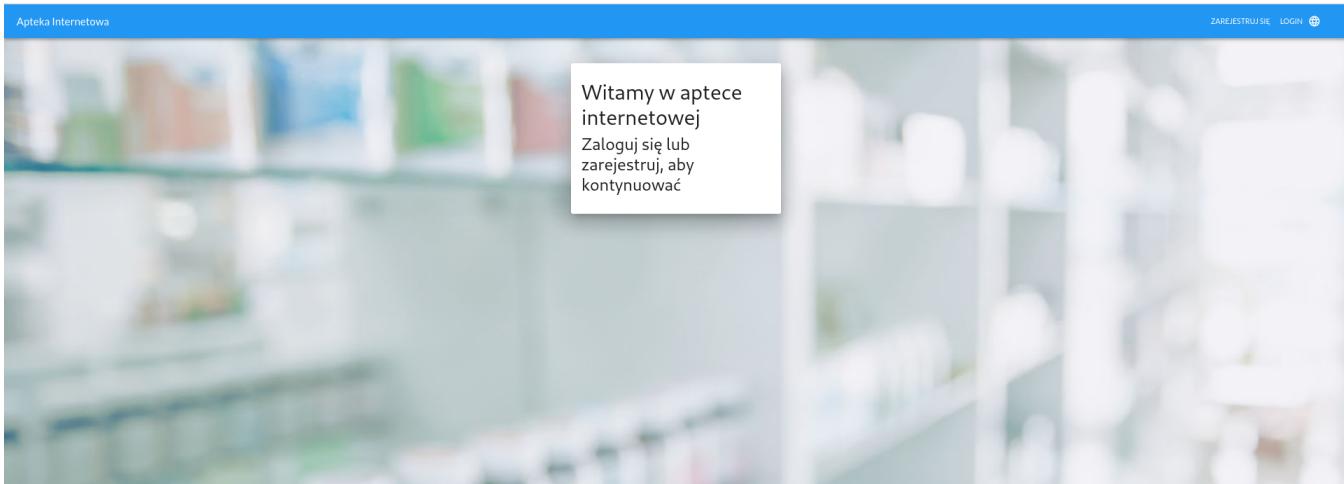
1	MOK.1	Gość
2	MOK.2	Administrator
3	MOK.3	Administrator
4	MOK.4	Administrator
5	MOK.5	Administrator
6	MOK.6	Administrator
7	MOK.7	Administrator, Farmaceuta, Pacjent
8	MOK.8	Administrator
9	MOK.9	Administrator, Farmaceuta, Pacjent
10	MOK.10	Administrator
11	MOK.11	Administrator, Farmaceuta, Pacjent
12	MOK.12	System
13	MOK.13	Administrator, Farmaceuta, Pacjent
14	MOK.14	Administrator
15	MOK.15	Administrator
16	MOK.16	Gość
17	MOK.17	Gość
18	MOA.1	Gość, Farmaceuta, Pacjent
19	MOA.2	Farmaceuta, Pacjent
20	MOA.3	Pacjent
21	MOA.4	Pacjent
22	MOA.5	Pacjent
23	MOA.6	Pacjent
24	MOA.7	Pacjent
25	MOA.8	Pacjent
26	MOA.9	Farmaceuta
27	MOA.10	Farmaceuta
28	MOA.11	Farmaceuta
29	MOA.12	Farmaceuta
30	MOA.13	Farmaceuta
31	MOA.14	Farmaceuta
32	MOA.15	Farmaceuta
33	MOA.16	System
34	MOA.17	Pacjent
35	MOA.18	Farmaceuta

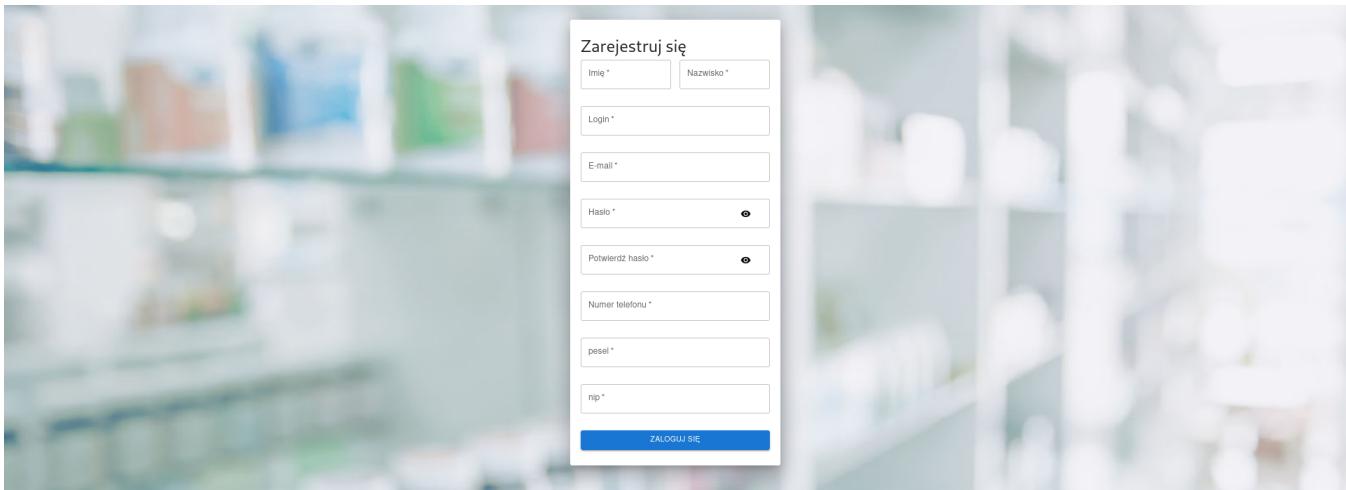
36	MOA.19	Farmaceuta
37	MOA.20	Farmaceuta
38	MOA.21	Farmaceuta
39	MOA.22	Farmaceuta
40	MOA.23	Farmaceuta

MOK.1 Zarejestruj

	Controller	Manager	Facade	Manager	Facade	Manager
Kolejność wywołania metod	1	2	3	4	5	6
Komponent	AccountController	AccountManager	AccountFacade	TokenManager	TokenFacade	EmailService
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful	Stateless	Stateful
Metoda	registerPatientAccount	registerAccount	create	sendVerificationToken	create	sendRegistrationEmail

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne zarejestrowanie konta pacjenta	/account/register	POST	201 (Created)	scenariusz pozytywny
Negatywne - brak wymaganego ciała żądania	/account/register	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywne - nieprawidłowe dane rejestracji pacjenta	/account/register	POST	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - błąd metody http	/account/register	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - duplikacja jednego z pól o wymaganej unikatowej wartości	/account/register	POST	409 (Conflict)	exception.account.duplicate.email exception.account.duplicate.login exception.account.duplicate.phone exception.account.duplicate.nip exception.account.duplicate.pesel
Negatywne - błąd wewnętrzny serwera	/account/register	POST	500 (Internal Server Error)	exception.general





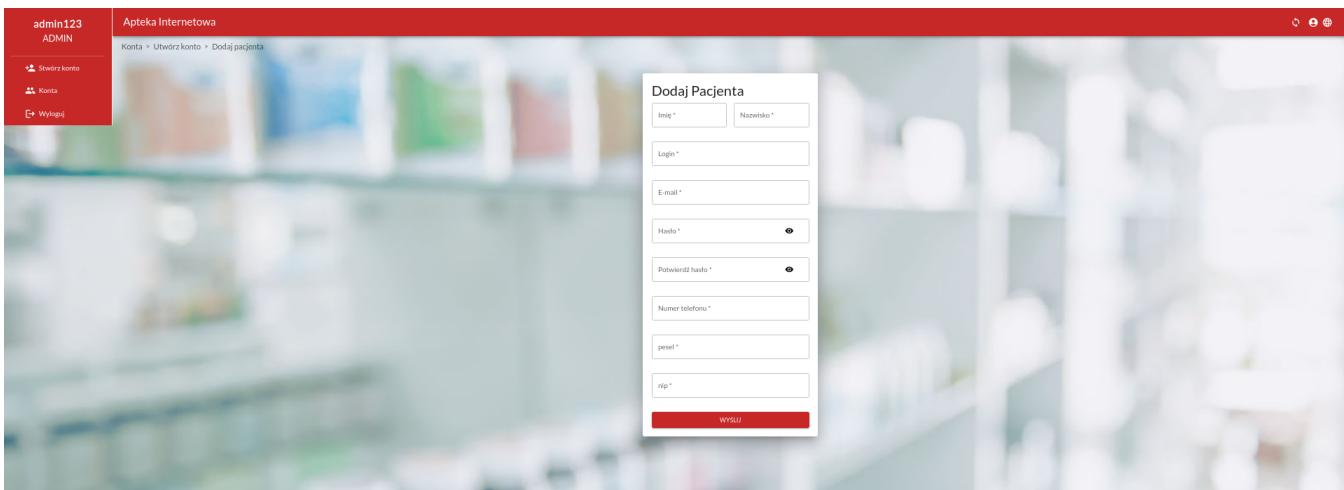
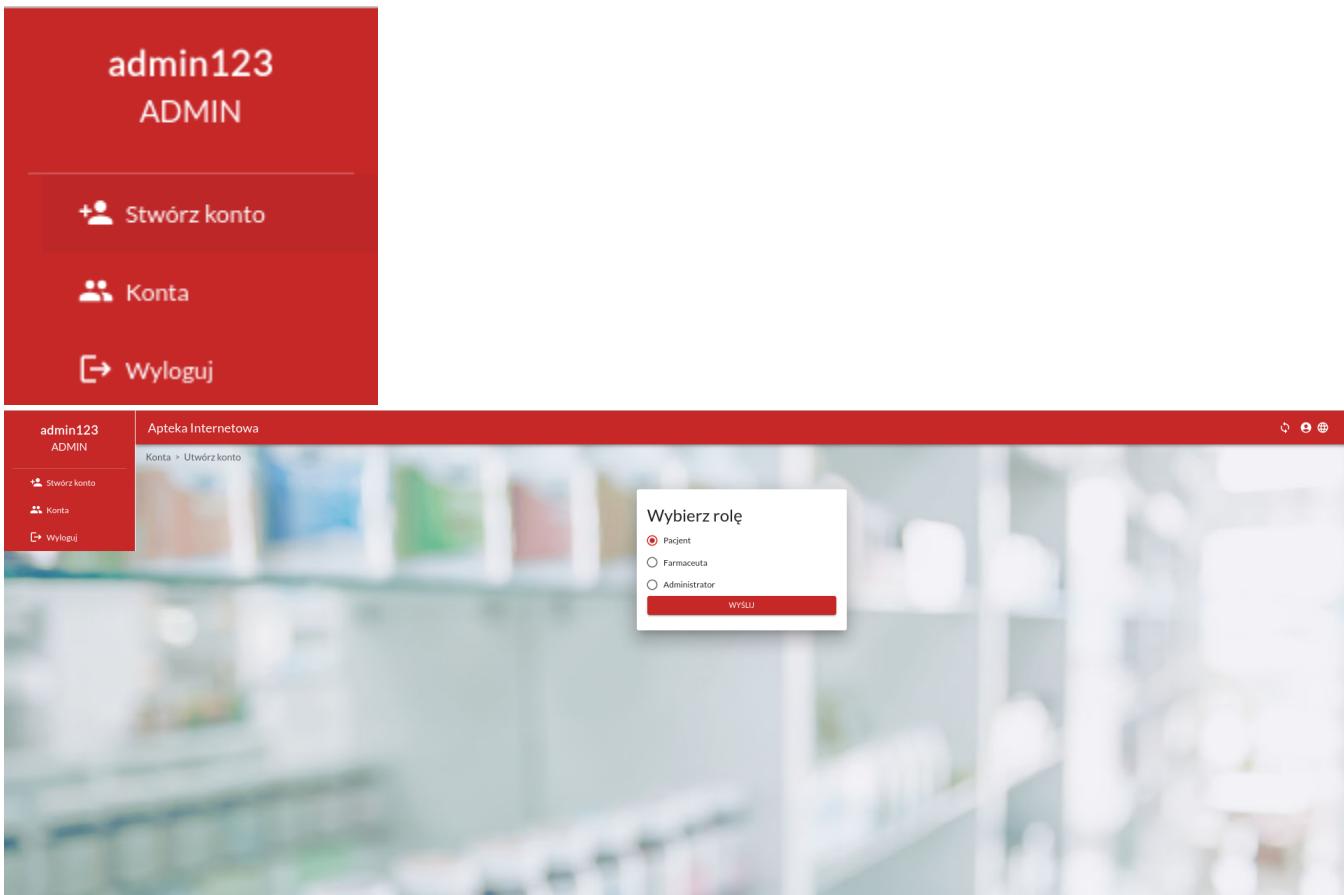
MOK.2 Utwórz konto

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	addPatientAccountAsAdmin lub addChemistPatientAccountAsAdmin lub addAdminAccountAsAdmin	createAccount	create

Poniższe scenariusze są analogiczne dla utworzenia poziomu dostępu administratora (/add-admin) oraz farmaceuty (/add-chemist) - różnice stanowią wartości pól specyficzne dla danego poziomu dostępu (np. *workPhoneNumber* dla administratora lub *licenseNumber* dla farmaceuty)

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne utworzenie konta pacjenta przez administratora	/add-patient	POST	201 (Created)	scenariusz pozytywny
Negatywne - brak wymaganego ciała żądania	/add-patient	POST	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - nieprawidłowe dane rejestracji pacjenta	/add-patient	POST	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - nieistniejący poziom dostępu	/add-patient	POST	400 (Bad Request)	exception.account.no-such-access-level
Negatywne - tylko administrator utworzyć konto	/add-patient	POST	403 (Forbidden)	exception.access-denied
Negatywne - błąd metody http	/add-patient	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieprawidłowy JWT	/add-patient	POST	401 (Unauthorised)	exception.unauthorised
Negatywne - duplikacja jednego z pól o wymaganej unikatowej wartości	/add-patient	POST	409 (Conflict)	exception.account.duplicate.email exception.account.duplicate.login exception.account.duplicate.phone exception.account.duplicate.nip exception.account.duplicate.pesel exception.account.duplicate.access-level

Negatywne - duplikacja poziomu dostępu	/add-patient	POST	409 (Conflict)	exception.account.duplicate.access-level
Negatywne - błąd wewnętrzny serwera	/add-patient	POST	500 (Internal Server Error)	exception.general



Wybierz rolę

- Pacjent
- Farmaceuta
- Administrator

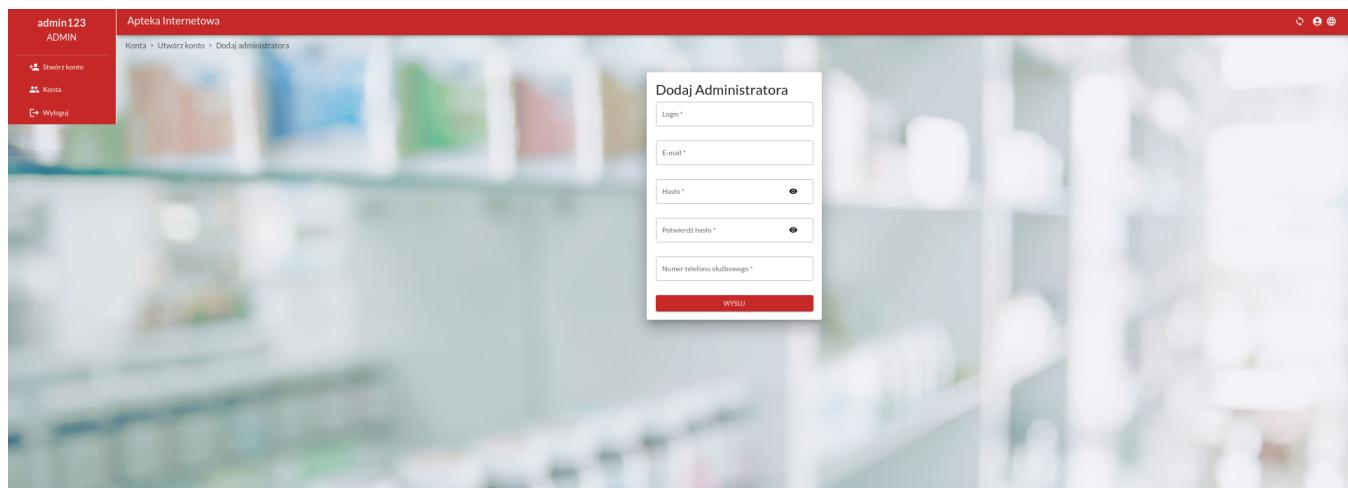
WYSUJ

The screenshot shows a web application interface for managing accounts. On the left, there's a sidebar with a red background containing the user information "admin123" and "ADMIN". Below this are three menu items: "Stwórz konto" (Create account), "Konta" (Accounts), and "Wyloguj" (Logout). The main content area has a blurred background of a pharmacy interior. At the top, a navigation bar reads "Apteka Internetowa" and "Konta > Utwórz konto > Dodaj aptekarza". A modal window titled "Dodaj Farmaceuta" is open, containing fields for "Login*", "Email*", "Hasło*" (password), "Potwierdź hasło*" (confirm password), and "Numer lizencji*". A red "WYSUJ" button is at the bottom of the modal.

Wybierz rolę

- Pacjent
- Farmaceuta
- Administrator

WYSUJ



MOK.3 Zablokuj konto

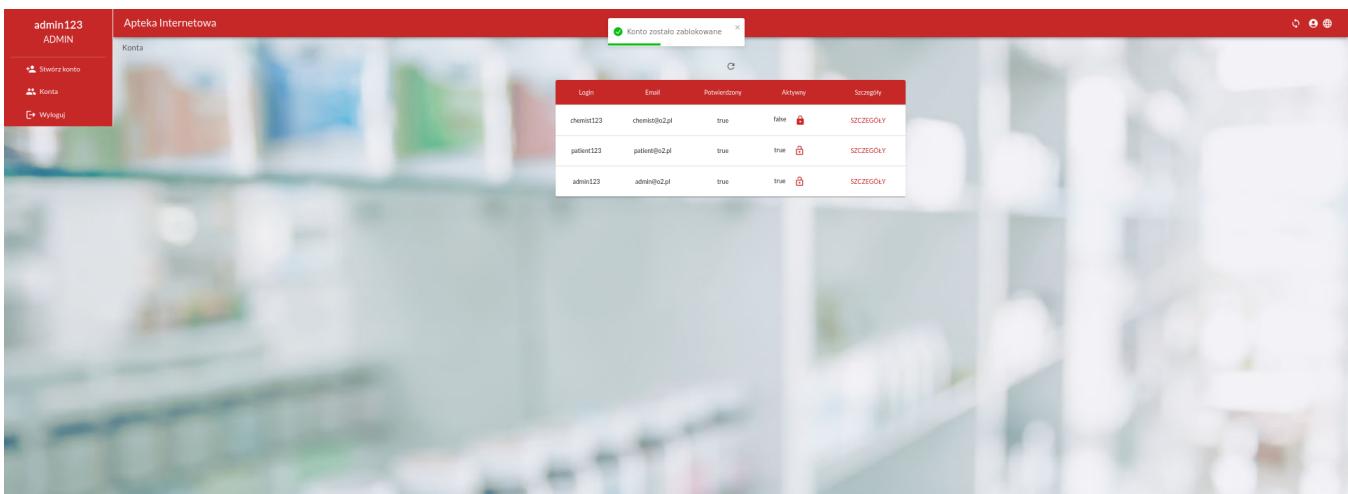
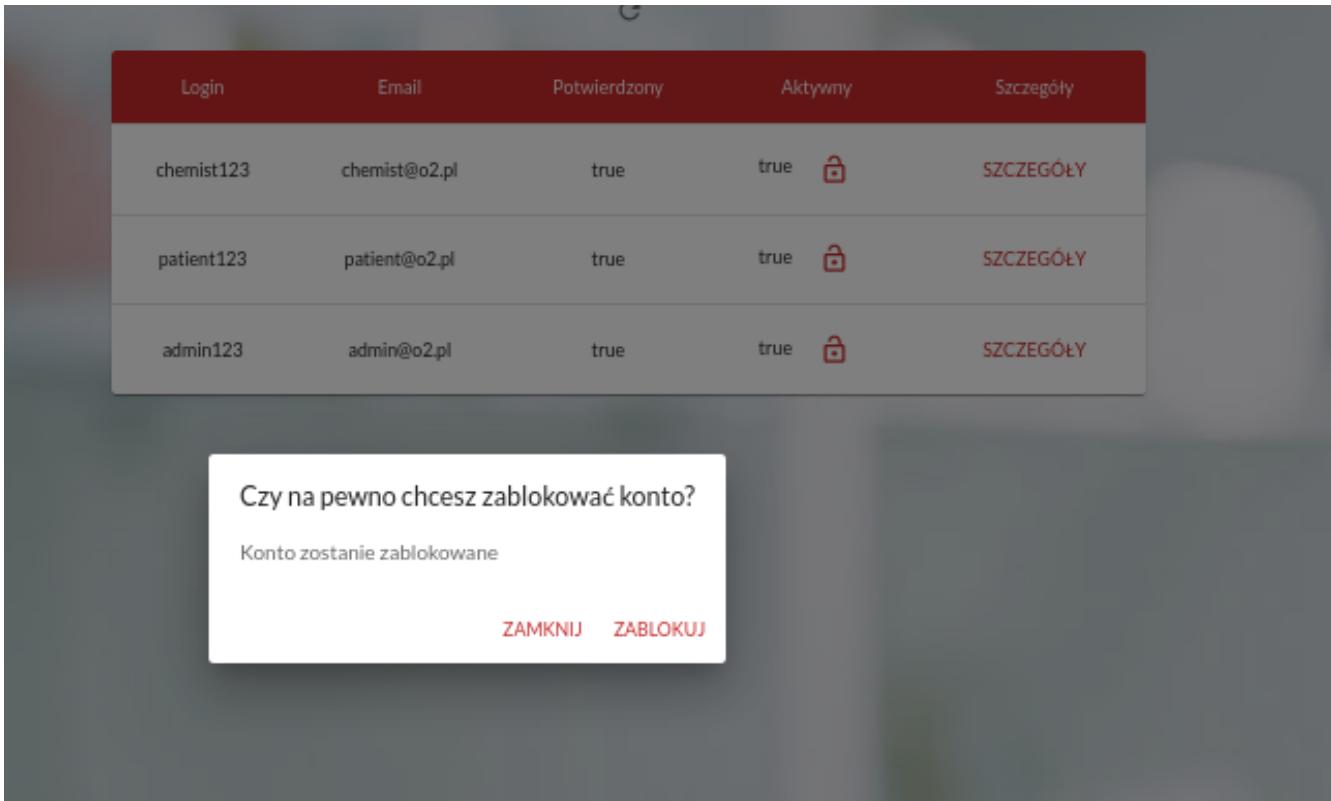
	Controller	Manager	Facade	Manager
Kolejność wywołania metod	1	2	3	4
Komponent	AccountController	AccountManager	AccountFacade	EmailService
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful
Metoda	blockAccount	blockAccount	edit	sendEmailAccountBlocked

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne - zablokowanie konta	/account/{id}/block	PUT	200 (OK)	scenariusz pozytywny
Negatywne - nieprawidłowy JWT	/account/{id}/block	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - nie znaleziono konta	/account/{id}/block	PUT	409 (Conflict)	exception.entity-not-found
Negatywne - błąd metody http	/account/{id}/block	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - tylko administrator może zablokować konto	/account/{id}/block	PUT	403 (Forbidden)	exception.access-denied
Negatywne - błąd wewnętrzny serwera	/account/{id}/block	PUT	500 (Internal Server Error)	exception.general

Po kliknięciu w Konto.

Login	Email	Powielony	Aktywny	Szczegóły
chemist123	chemist@o2.pl	true	true	
patient123	patient@o2.pl	true	true	
admin123	admin@o2.pl	true	true	

Po kliknięciu w kıldkę.

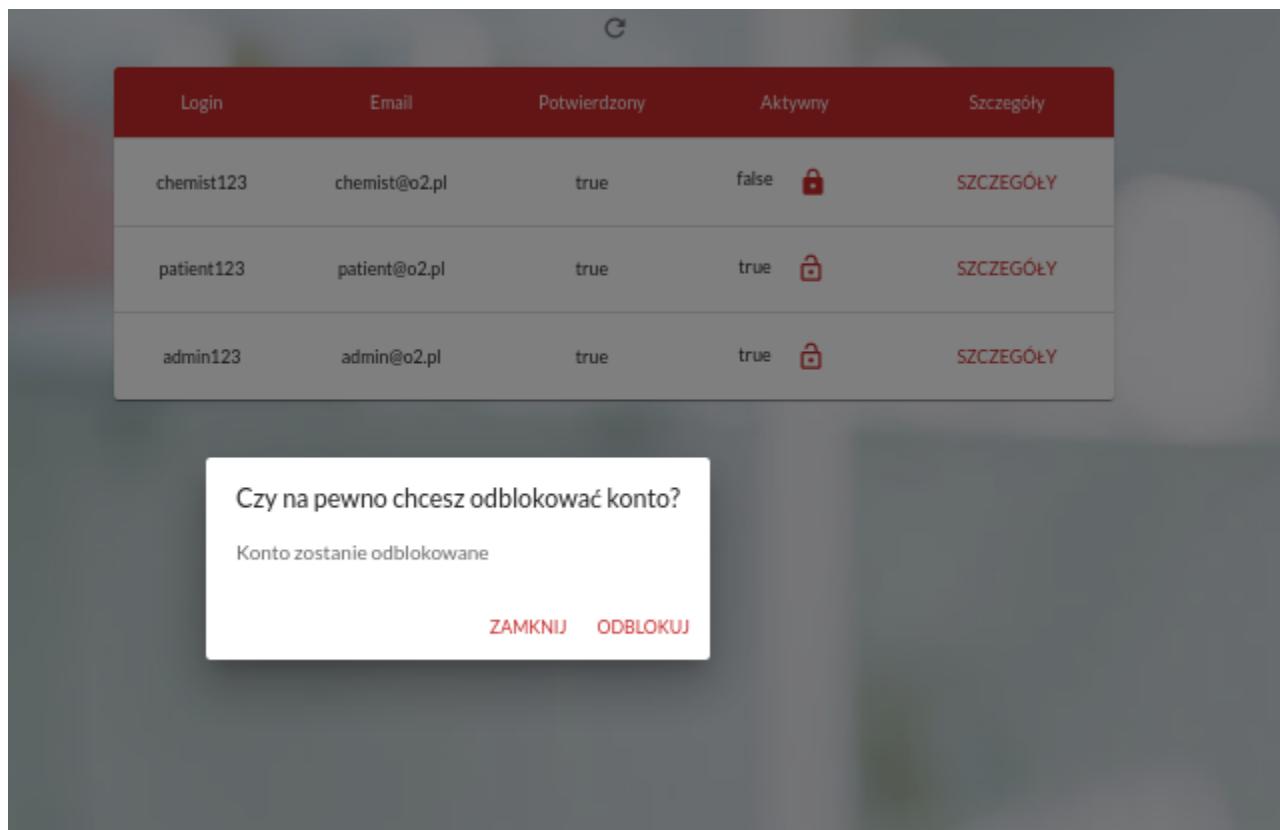


MOK.4 Odblokuj konto

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozitywne - odblokowanie konta	/account/{id}/unblock	PUT	200 (OK)	scenariusz pozytywny
Negatywne - nieprawidłowy JWT	/account/{id}/unblock	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - nie znaleziono konta	/account/{id}/unblock	PUT	409 (Conflict)	exception.entity-not-found
Negatywne - błąd metody http	/account/{id}/unblock	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - tylko administrator może odblokować konto	/account/{id}/unblock	PUT	403 (Forbidden)	exception.access-denied

Negatywne - błąd wewnętrzny serwera	/account/{id}/unblock	PUT	500 (Internal Server Error)	exception.general
-------------------------------------	-----------------------	-----	-----------------------------	-------------------

	Controller	Manager	Facade	Manager
Kolejność wywołania metod	1	2	3	4
Komponent	AccountController	AccountManager	AccountFacade	EmailService
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful
Metoda	unblockAccount	unblockAccount	edit	sendEmailAccountUnblocked



The screenshot shows the main interface of the "Apteka Internetowa" application. On the left, there is a sidebar with a user menu (admin123, ADMIN) and navigation links for "Stwórz konto", "Konta", and "Wyloguj". The main content area is titled "Apteka Internetowa" and shows a list of accounts under the heading "Konta". A success message "Konto zostało odblokowane" is displayed in a green box at the top right. The account list table has columns: Login, Email, Potwierdzony, Aktywny, and Szczegóły. The table contains three rows of data corresponding to the users listed in the modal dialog.

Login	Email	Potwierdzony	Aktywny	Szczegóły
chemist123	chemist@o2.pl	true	false	SZCZEGÓŁY
patient123	patient@o2.pl	true	true	SZCZEGÓŁY
admin123	admin@o2.pl	true	true	SZCZEGÓŁY

	Controller	Manager	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4	5
Komponent	AccountController	AccountManager	AccountManager	AccountFacade	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateful	Stateless	Stateless
Metoda	grantPatient lub grantChemist lub grantAdmin	grantAccessLevel	getAccount	find	edit

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne dodanie poziomu dostępu pacjenta	/account/{id} /patient	POST	201 (Created)	scenariusz pozytywny
Negatywne - brak wymaganego ciała zapytania	/account/{id} /patient	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywne - nieprawidłowe dane poziomu dostępu pacjenta	/account/{id} /patient	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywne - nieprawidłowy JWT	/account/{id} /patient	POST	401 (Unauthorised)	exception.unauthorised
Negatywne - duplikacja jednego z pól o wymaganej unikatowej wartości	/account/{id} /patient	POST	409 (Conflict)	exception.account.duplicate.email exception.account.duplicate.login exception.account.duplicate.phone exception.account.duplicate.nip exception.account.duplicate.pesel
Negatywne - błąd metody http	/account/{id} /patient	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - tylko administrator może dołączyć poziom dostępu	/account/{id} /patient	POST	403 (Forbidden)	exception.access-denied
Negatywne - równoległa modyfikacja konta	/account/{id} /patient	POST	409 (Conflict)	exception.optimistic-lock
Negatywne - nie znaleziono konta	/account/{id} /patient	POST	409 (Conflict)	exception.entity-not-found
Negatywne - nieprawidłowa wartość eTag	/account/{id} /patient	POST	400 (Bad Request)	exception.etag.not.valid
Negatywne - pusta wartość eTag	/account/{id} /patient	POST	400 (Bad Request)	exception.etag.empty
Negatywne - błąd wewnętrzny serwera	/account/{id} /patient	POST	500 (Internal Server Error)	exception.general

Podobnie jak w przypadku utworzenia konta przez administratora, powyższe scenariusze oraz kontekst są analogiczne dla poziomu dostępu administratora (/account/{id}/admin) oraz farmaceuty (/account/{id}/chemist) - różnice stanowią wartości pól specyficzne dla danego poziomu dostępu (np. *workPhoneNumber* dla administratora lub *licenseNumber* dla farmaceuty)

	Controller	Manager	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4	5
Komponent	AccountController	AccountManager	AccountManager	AccountFacade	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateful	Stateless	Stateless

Metoda	unBlockRoleAdmin lub unBlockRoleChemist lub unBlockRolePatient	activateAccessLevel	getAccountAndAccessLevel	findAndRefresh	edit
--------	--	---------------------	--------------------------	----------------	------

Apteka Internetowa

Konta

Login	Email	Potwierdzony	Aktivny	Szczegóły
patient123	patient@o2.pl	true	true	Szczegóły
admin123	admin@o2.pl	true	true	Szczegóły
chemist123	chemist@o2.pl	true	true	Szczegóły

Wybieramy szczegóły.

Apteka Internetowa

Konta > Details

Szczegóły konta

Potwierdzony	Aktivny
true	true

Email
patient@o2.pl

ZMIEN EMAIL

ZMIEN HASŁO

DODAJ ROLĘ

Login
patient123

Pozj. dostępu
PATIENT
R

Imię
Jan

Nazwisko
Kowalski

Numer telefonu
721545784

Pesel
22344078801

[EDYTU DETALE KONTA](#)

Dodaj rolę.

Szczegóły konta

Potwierdzony — true Aktywny — true

Email
patient@o2.pl

ZMIEN EMAIL

ZMIEN HASŁO

Select Role — CHEMIST

licenseNumber * — 123456

DODAJ POZIOM DOSTĘPU

POWRÓT

Login
patient123

Poziom dostępu
PATIENT

Imię

Szczegóły konta

Potwierdzony — true Aktywny — true

Email
patient@o2.pl

ZMIEN EMAIL

ZMIEN HASŁO

DODAJ ROLE

Login
patient123

Poziom dostępu
CHEMIST PATIENT

Numer licencji
123456

Imię
Jan

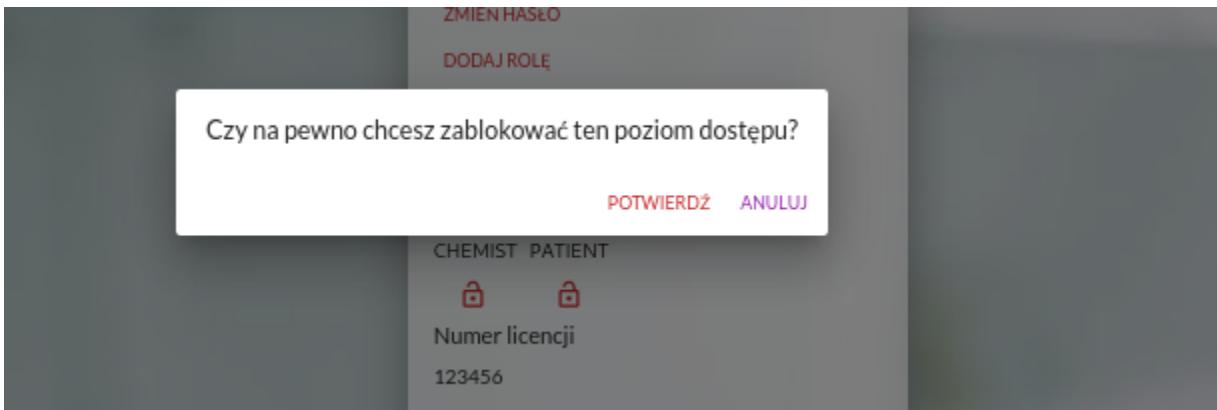
Nazwisko
Kowalski

Numer telefonu
721545784

Pesel
22344678801

EDYTUJ DETALE KONTA

Blokowanie poziomu dostępu.



Szczegóły konta

Potwierdzony	true	Akttywny	true
--------------	------	----------	------

Email
patient@o2.pl

ZMIEN EMAIL

ZMIEN HASŁO

DODAJ ROLE

Login
patient123

Poziom dostępu
CHEMIST PATIENT

🔒	🔒
---	---

Numer licencji
123456

Imię
Jan

Nazwisko
Kowalski

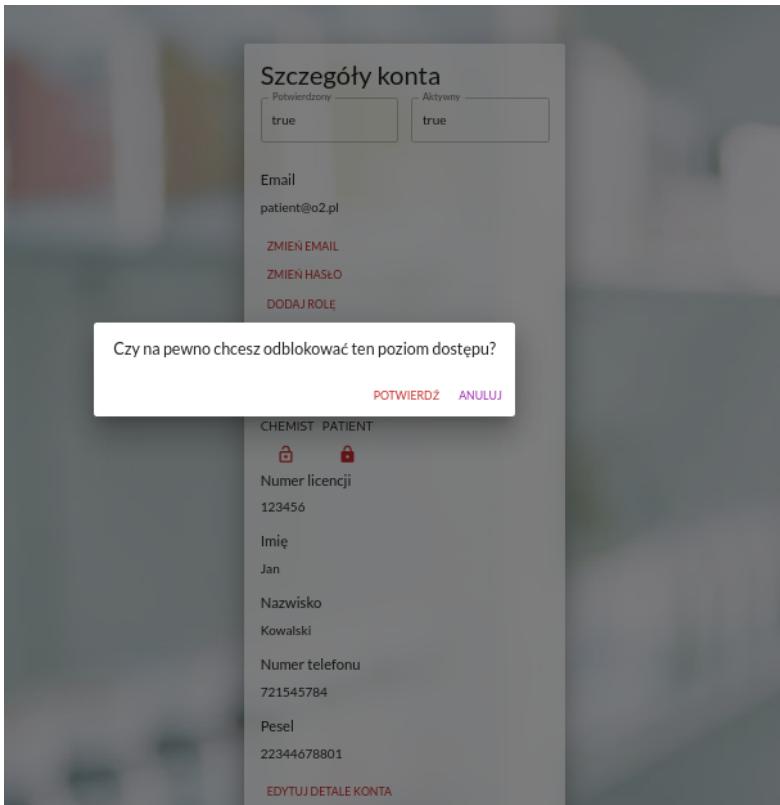
Numer telefonu
721545784

Pesel
22344678801

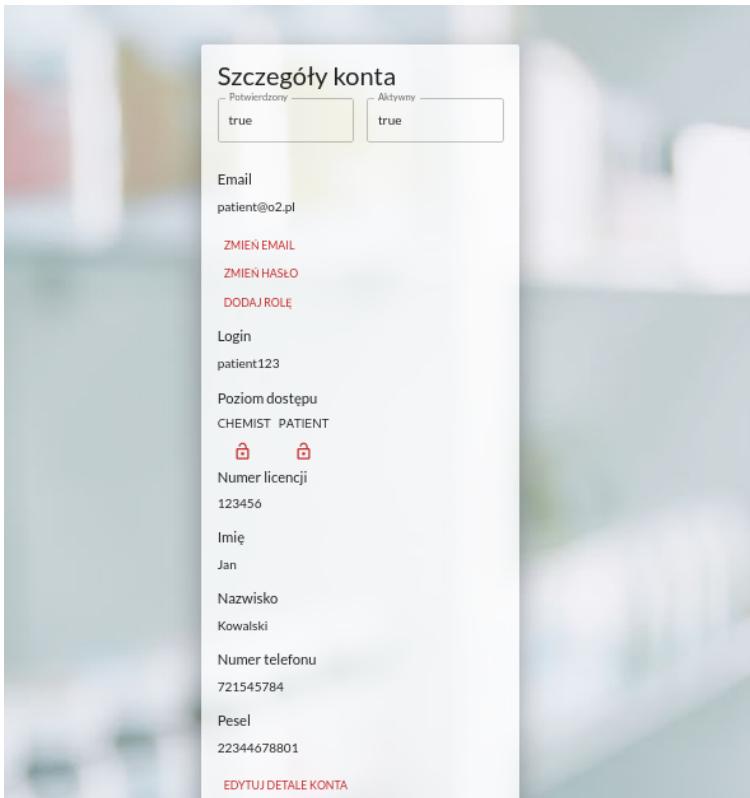
[EDYTUJ DETALE KONTA](#)

MOK.6 Odłącz poziom dostępu od konta

	Controller	Manager	Facade	Facade	Facade
Kolejność wywołania metod	1	2	3	4	5
Komponent	AccountController	AccountManager	AccountFacade	AccountFacade	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless	Stateless
Metoda	BlockRoleAdmin lub BlockRoleChemist lub BlockRolePatient	deactivateAccessLevel	getAccountAndAccessLevel	findAndRefresh	edit



Po kliknięciu zamkniętej kłódki i potwierdzeniu.



Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
------------------	--------------	-------------	---------------------	---------------------------

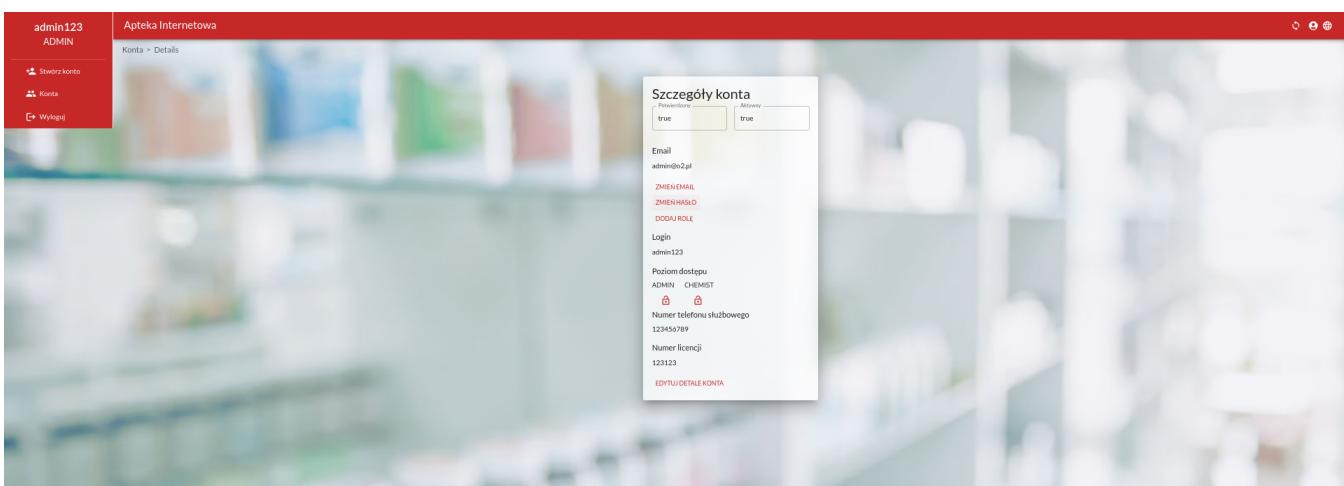
Pozytywne - zablokowanie poziomu dostępu konta	/account/{id}/patient/block	PUT	204 (No Content)	scenariusz pozytywny
Negatywne - nieprawidłowy JWT	/account/{id}/patient/block	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - nie znaleziono konta	/account/{id}/patient/block	PUT	409 (Conflict)	exception.entity-not-found
Negatywne - tylko administrator może odłączyć poziom dostępu	/account/{id}/patient/block	PUT	403 (Forbidden)	exception.access-denied
Negatywne - błąd metody http	/account/{id}/patient/block	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nie można zablokować jedynego poziomu dostępu danego konta	/account/{id}/patient/block	PUT	400 (Bad Request)	exception.account.deactivate.last-access-level
Negatywne - błąd wewnętrzny serwera	/account/{id}/patient/block	PUT	500 (Internal Server Error)	exception.general

Przypadek blokowania farmaceuty jest analogiczny - różnicę stanowi kontekst (/account/{id}/chemist/block). W przypadku blokowania poziomu dostępu administratora, oprócz zmiany kontekstu (/account/{id}/admin/block) pojawia się jeszcze jeden scenariusz:

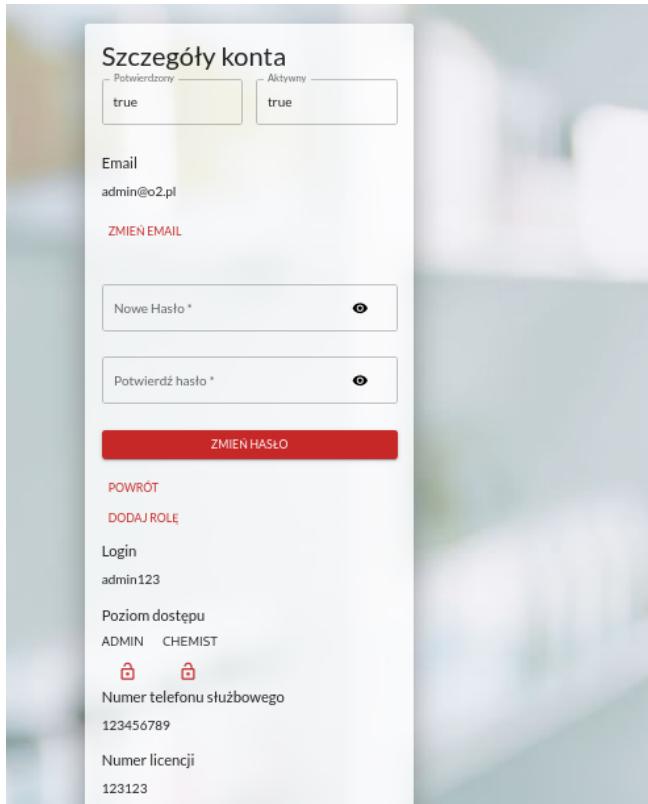
Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Negatywne - administrator nie może odebrać poziomu dostępu swojemu własnemu kontu	/account/{id}/admin/block	PUT	204 (No Content)	exception.account.deactivate.self

MOK.7 Zmień własne hasło

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	changePassword	updateOwnPassword	edit



Klikamy zmień hasło.

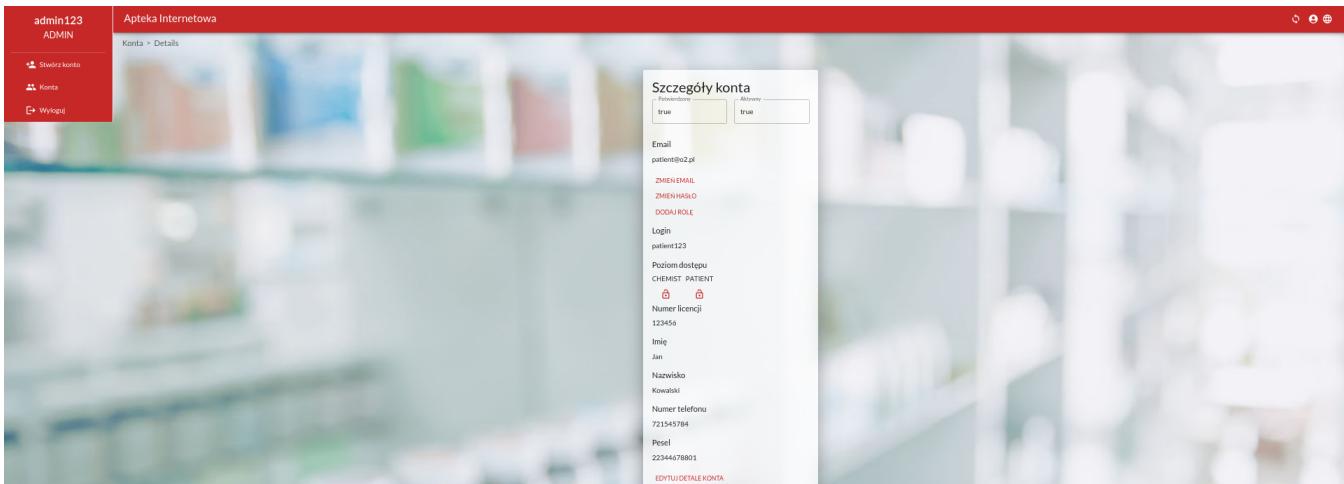


Klikamy zmień hasło i zatwierdź.

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne - zmiana własnego hasła	/account/change-password	PUT	200 (OK)	scenariusz pozytywny
Negatywne - hasło takie samo jak poprzednie	/account/change-password	PUT	409 (Conflict)	exception.password.not-changed
Negatywne - nie znaleziono konta	/account/change-password	PUT	404 (Not Found)	exception.entity-not-found
Negatywne - niepoprawny login lub hasło	/account/change-password	PUT	401 (Unauthorized)	exception.auth.bad-credentials
Negatywne - nieprawidłowy JWT	/account/change-password	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - równoległa modyfikacja konta	/account/change-password	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - nieprawidłowa wartość eTag	/account/change-password	PUT	400 (Bad Request)	exception.etag.not-valid
Negatywne - pusta wartość eTag	/account/change-password	PUT	400 (Bad Request)	exception.etag.empty
Negatywne - błąd metody http	/account/change-password	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - ładunek żądania odwołuje się do innej encji	/account/change-password	PUT	400 (Bad Request)	exception.mismatched.payload
Negatywne - błąd wewnętrzny serwera	/account/change-password	PUT	500 (Internal Server Error)	exception.general

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	changeUserPassword	updateUserPassword	edit

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacionalizacji
Pozytywne - zmiana hasła użytkownika przez administratora	/account/{id}/change-user-password	PUT	200 (OK)	scenariusz pozytywny
Negatywne - nie znaleziono konta	/account/{id}/change-user-password	PUT	404 (Not Found)	exception.entity-not-found
Negatywne - tylko administrator może zmieniać hasło innego użytkownika	/account/{id}/change-user-password	PUT	403 (Forbidden)	exception.access-denied
Negatywne - równoległa modyfikacja konta	/account/{id}/change-user-password	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - nieprawidłowy JWT	/account/{id}/change-user-password	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - błąd metody http	/account/{id}/change-user-password	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieprawidłowa wartość eTag	/account/{id}/change-user-password	PUT	400 (Bad Request)	exception.etag.not-valid
Negatywne - pusta wartość eTag	/account/{id}/change-user-password	PUT	400 (Bad Request)	exception.etag.empty
Negatywne - ładunek żądania odwołuje się do innej encji	/account/{id}/change-user-password	PUT	400 (Bad Request)	exception.mismatched.payload
Negatywne - błąd wewnętrzny serwera	/account/{id}/change-user-password	PUT	500 (Internal Server Error)	exception.general



Klikamy zmień hasło.

The screenshot shows a user profile page titled "Szczegóły konta". It displays the following information:

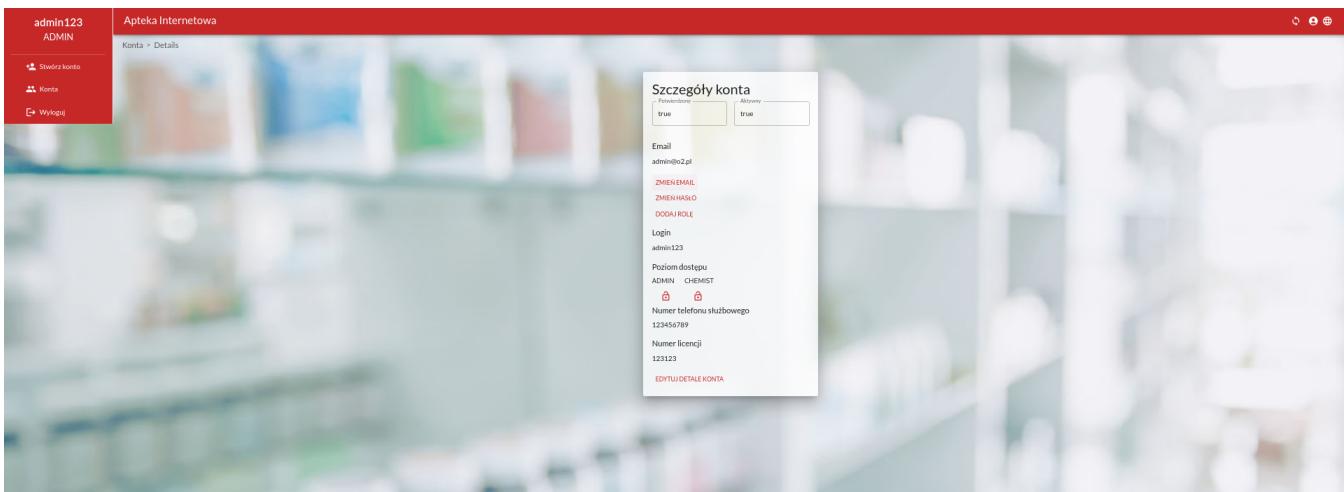
- Potwierdzony**: true
- Aktywny**: true
- Email**: patient@o2.pl
- ZMIEN EMAIL** (button)
- Nowe Hasło ***: masked input field containing "*****"
- Potwierdź hasło ***: masked input field containing "*****"
- ZMIEN HASŁO** (button)
- POWRÓT**
- DODAJ ROLE**
- Login**: patient123
- Poziom dostępu**: CHEMIST PATIENT (with two padlock icons)
- Numer licencji**: 123456
- Imię**: Jan

Klikamy zmień hasło potwierdzamy.

MOK.9 Edytuj dane własnego konta

	Controller	Manager	Facade	Manager	Facade	Manager
Kolejność wywołania metod	1	2	3	4	5	6
Komponent	AccountController	AccountManager	AccountFacade	TokenManager	TokenFacade	EmailService
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful	Stateless	Stateful
Metoda	editOwnAccount	updateOwnEmail	edit	sendEmailChangeEmail	create	sendEmailChangeEmail

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozitywne - edytowanie danych własnego konta	/account	PUT	200 (OK)	scenariusz pozytywny
Negatywne - duplikacja jednego z pól o wymaganej unikatowej wartości	/account	PUT	409 (Conflict)	exception.account.duplicate.email
Negatywne - nieprawidłowy JWT	/account	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - równoległa modyfikacja konta	/account	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - błąd metody http	/account	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieprawidłowa wartość eTag	/account	PUT	400 (Bad Request)	exception.etag.not-valid
Negatywne - pusta wartość eTag	/account	PUT	400 (Bad Request)	exception.etag.empty
Negatywne - błąd wewnętrzny serwera	/account	PUT	500 (Internal Server Error)	exception.general



Klikamy zmień Email.

Szczegóły konta

Potwierdzony
Aktywny

Email
 admin@o2.pl

Nowy Email *

Potwierdź Email *

[ZMIEN EMAIL](#)

[POWRÓT](#)
[ZMIEN HASŁO](#)
[DODAJ ROLE](#)

Login
 admin123

Poziom dostępu
 ADMIN CHEMIST

Numer telefonu służbowego
 123456789

Numer licencji
 123123

MOK.10 Edytuj dane konta innego użytkownika

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykł życia	Stateful, RequestScoped	Stateful	Stateless

Metoda	editAdminData lub editPatientData lub editChemistData	editAccessLevel	edit
--------	---	-----------------	------

The screenshot shows a user interface for managing accounts. On the left, there's a sidebar with a user profile (admin123, ADMIN), navigation links (Stwórz konto, Konta, Wyloguj), and a blurred background image of medicine bottles. The main content area has a title 'Szczegóły konta'. It contains sections for 'Pielęgnacyjny' (true) and 'Aktywny' (true). Below these are fields for 'Email' (patient@o2.pl), 'ZMIEN EMAIL', 'ZMIEN HASŁO', 'DODAJ RÓŁĘ', 'Login' (patient123), 'Poziom dostępu' (CHEMIST PATIENT), 'Numer licencji' (123456), 'Imię' (Jan), 'Nazwisko' (Kowalski), 'Numer telefonu' (721545784), and 'Pesel' (22344678801). At the bottom right of this panel is a red button labeled 'EDYTUJ DETALE KONTA'.

Klikamy edytuj detale konta.

The screenshot shows a modal dialog titled 'Edytuj detale konta'. It has two main sections: 'EDYTUJ DANE APTEKARZA' and 'EDYTUJ DANE PACJENTA'. Both sections contain fields for 'Imię *' (Name) with 'Jan' entered, 'Nazwisko *' (Surname) with 'Kowalski' entered, 'Numer telefonu *' (Phone number) with '721545784' entered, and 'Pesel *' (Social Security Number) with '22344678801' entered. The 'NIP *' field is also present in the patient section with '2234557890' entered. Red buttons labeled 'EDYTUJ DANE APTEKARZA' and 'EDYTUJ DANE PACJENTA' are located at the bottom of their respective sections.

Podobnie jak w początkowych przypadkach, także i tutaj edycja konta przez administratora jest analogiczna dla pozostałych poziomów dostępu. Zmianie ulega kontekst - dla farmaceuty (/account/{id}/chemist) oraz dla administratora (/account/{id}/admin)

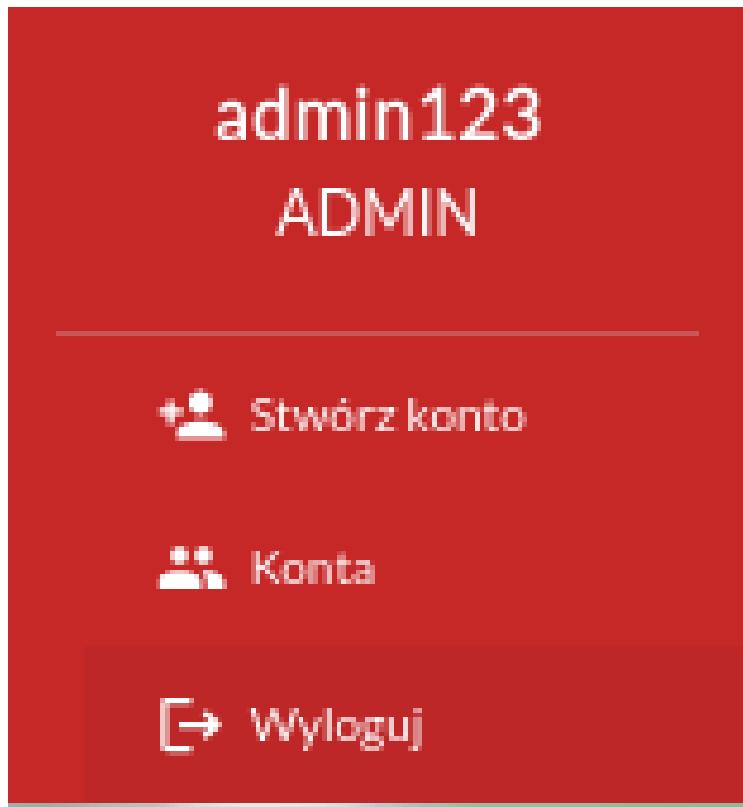
Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozatywne - edytowanie danych innego konta użytkownika	/account/{id}/patient	PUT	200 (OK)	scenariusz pozytywny

Negatywne - duplikacja jednego z pól o wymaganej unikatowej wartości	/account/{id} /patient	PUT	409 (Conflict)	exception.account.duplicate. email exception.account.duplicate.login exception.account.duplicate.phone exception.account.duplicate.nip exception.account.duplicate.pesel
Negatywne - błąd metody http	/account/{id} /patient	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieprawidłowy JWT	/account/{id} /patient	PUT	401 (Unauthorised)	exception.unauthorised
Negatywne - nie znaleziono konta	/account/{id} /patient	PUT	404 (Not Found)	exception.entity-not-found
Negatywne - tylko administrator może edytować dane innego konta użytkownika	/account/{id} /patient	PUT	403 (Forbidden)	exception.access-denied
Negatywne - równoległa modyfikacja konta	/account/{id} /patient	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - nieprawidłowa wartość eTag	/account/{id} /patient	PUT	400 (Bad Request)	exception.etag.not-valid
Negatywne - pusta wartość eTag	/account/{id} /patient	PUT	400 (Bad Request)	exception.etag.empty
Negatywne - błąd wewnętrzny serwera	/account/{id} /patient	PUT	500 (Internal Server Error)	exception.general

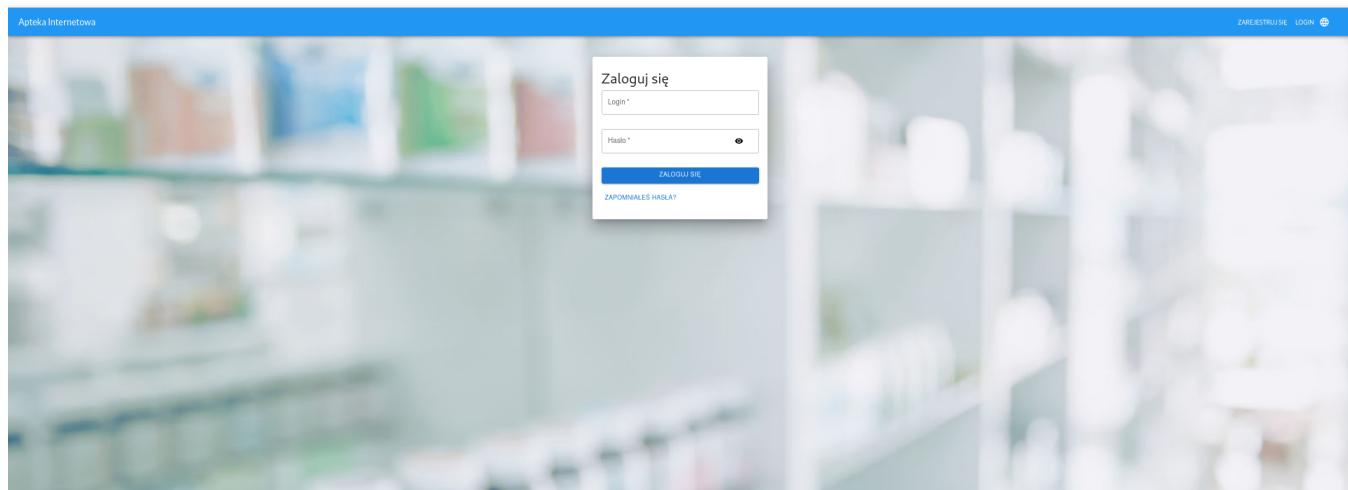
Po edycji i zatwierdzeniu jesteśmy przekierowani do poprzedniego okna.

MOK.11 Wyloguj

Zrealizowane na poziomie widoku.



Po kliknięciu wyloguj i potwierdzeniu.



MOK.12 Usuń konto w przypadku braku aktywacji

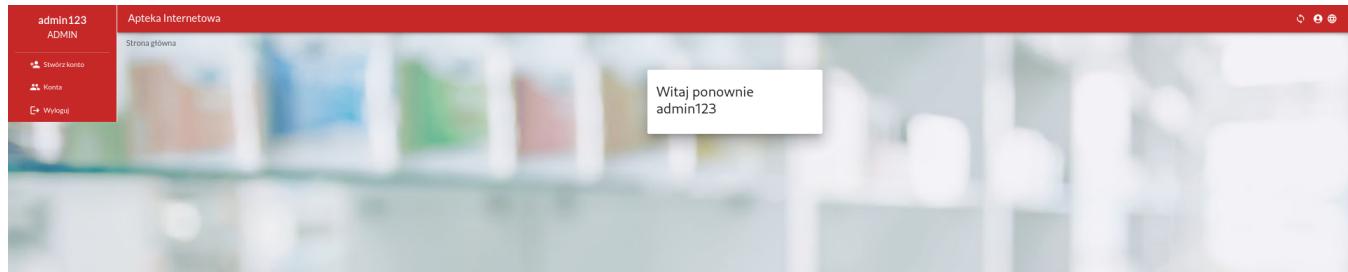
Realizacja tylko w backendzie.

	Scheduler	Manager	Facade	Facade	EmailService
Kolejność wywołania metod	1	2	3	4	5
Komponent	AccountScheduledTask	AccountManager	AccountFacade	AccountFacade	EmailService
Cykl życia	ApplicationScoped	Stateful	Stateless	Stateless	Stateful
Metoda	purgeUnactivatedAccounts	purgeUnactivatedAccounts	findNotConfirmed	remove	sendEmailWhenRemovedDueToNotConfirmed

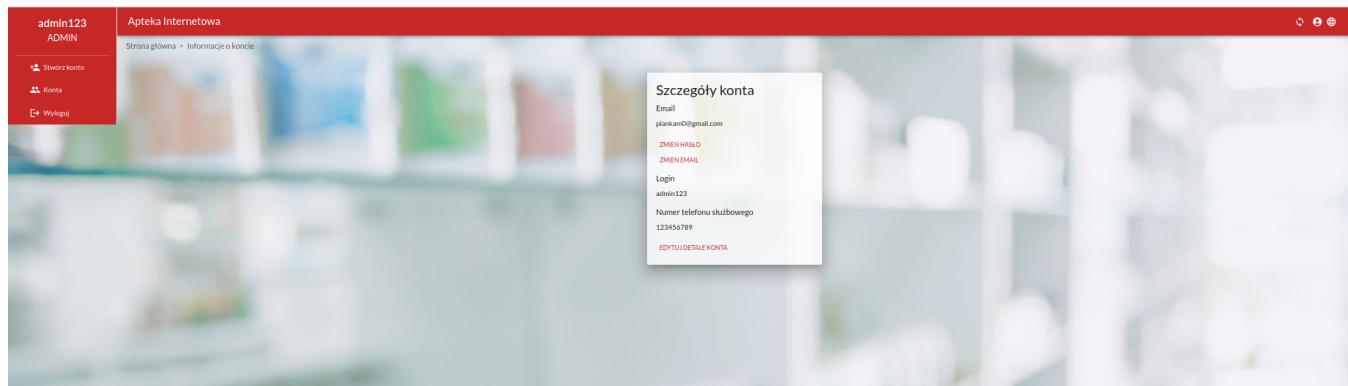
MOK.13 Wyświetl informacje o koncie

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	readOwnAccount	getCurrentUserWithAccessLevels	findByLoginAndRefresh

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie informacji o koncie	account/details	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	account/details	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	account/details	GET	403 (Forbidden)	exception.access.denied
Negatywny - niezalogowany użytkownik	account/details	GET	401 (Unauthorized)	exception.etag.creation
Negatywne - błąd wewnętrzny serwera	account/details	GET	500 (Internal Server Error)	exception.general



Po kliknięciu ikony ludzika w prawym górnym rogu.



Admin może wyświetlić informacje o koncie również poprzez Konta → Szczegóły tj. pokazano w poprzednich przypadkach użycia.

MOK.14 Wyświetl listę kont

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless

Metoda	readAccount	getAccount	find
--------	-------------	------------	------

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie informacji o koncie	account/{id}	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	account/{id}	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	account/{id}	GET	403 (Forbidden)	exception.access.denied
Negatywne - błąd wewnętrzny serwera	account/{id}	GET	500 (Internal Server Error)	exception.general

Po kliknięciu w Konta.

MOK.15 Wyświetl informacje o koncie innego użytkownika

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	AccountController	AccountManager	AccountFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	readAllClients	getAllAccounts	findAll

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie informacji o koncie	account/	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	account/	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	account/	GET	403 (Forbidden)	exception.access.denied
Negatywne - błąd wewnętrzny serwera	account/	GET	500 (Internal Server Error)	exception.general

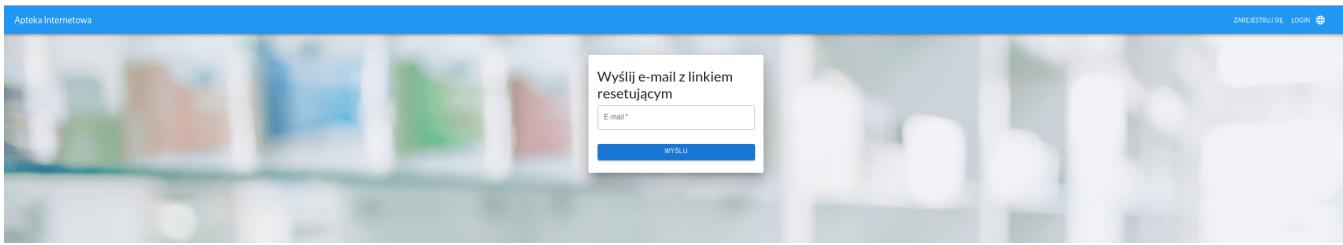
Tj. ukazano w poprzednich przypadkach użycia klikamy Konta → Szczegóły.

MOK.16 Zresetuj hasło

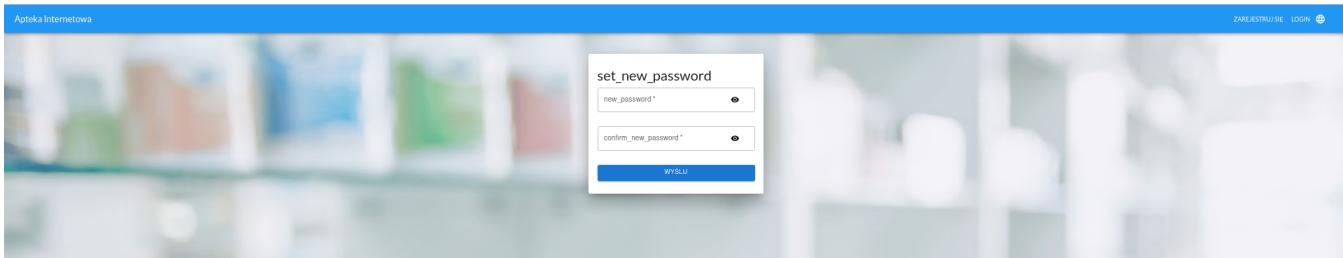
	Controller	Manager	Facade	Manager	Facade	Manager	Controller	Manager	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4	5	6	7	8	9	10	10
Komponent	AccountController	AccountManager	AccountFacade	TokenManager	TokenFacade	EmailService	AccountController	AccountManager	TokenManager	TokenFacade	TokenFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful	Stateful	Stateful	Stateful, RequestScoped	Stateful	Stateful	Stateless	Stateless
Metoda	resetPassword	sendResetPasswordToken	findByEmail	sendResetPasswordToken	create	sendEmailResetPassword	setNewPassword	setNewPassword	setNewPassword	findByCode	edit

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacionalizacji
Poztywne wysłanie linka	account/reset-password	PUT	200 (Ok)	scenariusz pozytywny
Negatywny - błąd metody http	account/reset-password	GET	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - brak wymaganego ciała żądania	account/reset-password	PUT	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - błąd wewnętrzny serwera	account/reset-password	PUT	500 (Internal Server Error)	exception.general
Negatywne - nieprawidłowe dane	account/new-password	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - brak wymaganego ciała żądania	account/new-password	PUT	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywny - błąd metody http	account/new-password	GET	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - błąd wewnętrzny serwera	account/new-password	PUT	500 (Internal Server Error)	exception.general

Klikamy zapomniałeś hasła.



Po kliknięciu wyslij i potwierdzeniu linka.

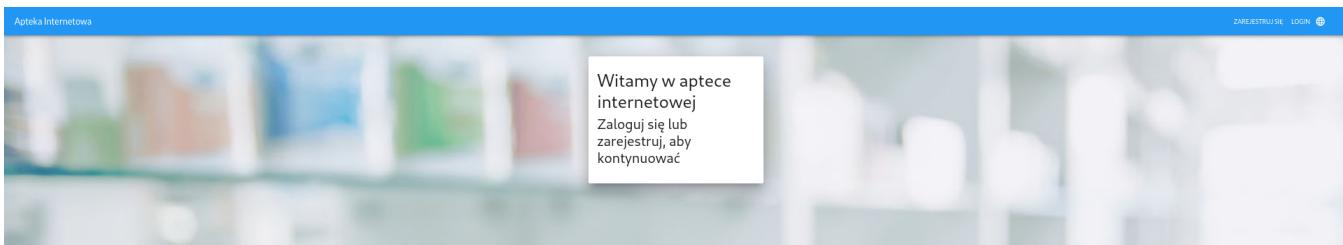


MOK.17 Zaloguj się na konto

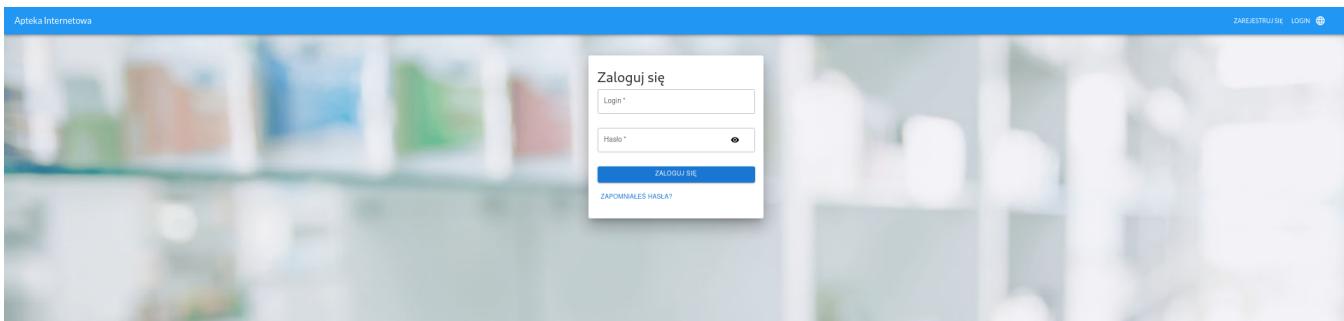
	Controller	Manager	Facade	Manager	Manager	Facade	Manager
Kolejność wywołania metod	1	2	3	4	5	6	7
Komponent	AuthController	AccountManager	AccountFacade	EmailService	AccountManager	AccountFacade	EmailService
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateful	Stateful	Stateless	Stateful
Metoda	authenticate	findByLogin	findByLogin	sendEmailAccountBlockedTooManyLogins	updateAuthInformation	findByLogin	sendEmailAccountBlockedTooManyLogins

Uwaga - metoda numer 4 i 7 zostaje wykonana tylko w określonym przypadku (zbyt wiele niepoprawnych logowań).

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacionalizacji
Pozytywne logowanie	auth/login	POST	200 (Ok)	scenariusz pozytywny
Negatywne - brak wymaganego ciała zapytania	auth/login	POST	500 (Internal Server Error)	exception.general
Negatywne - nieprawidłowe dane	auth/login	POST	401 (Unauthorized)	exception.auth.bad.credentials
Negatywne - próba do zalogowania się na niepotwierdzone konto	auth/login	POST	401 (Unauthorized)	exception.account_not_confirmed
Negatywny - błąd metody http	auth/login	GET	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - błąd wewnętrzny serwera	auth/login	POST	500 (Internal Server Error)	exception.general



Klikamy login.



MOA.1 - Wyświetl listę leków

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	MedicationController	MedicationManager	MedicationFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	getAllMedications	getAllMedications	findAll

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozitywne wyświetlenie leków	medication/	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	medication/	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	medication/	GET	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	medication/	GET	500 (Internal Server Error)	exception.general



Po zalogowaniu na konto pacjenta klikamy w lista leków.

Imię	Cena	Ilość	Na receptę?	Szczegóły
Penicylina	10	10	Tak	SZCZEGÓŁY DODAJ
Ibuprofen	10	10	Nie	SZCZEGÓŁY DODAJ
Aspirin	10	10	Nie	SZCZEGÓŁY DODAJ
Prozac	10	10	Tak	SZCZEGÓŁY DODAJ
Zoloft	20	10	Tak	SZCZEGÓŁY DODAJ
Marsjanki	25	10	Nie	SZCZEGÓŁY DODAJ
Amoksycylina	25	10	Tak	SZCZEGÓŁY DODAJ
Witamina D3	25	10	Nie	SZCZEGÓŁY DODAJ

MOA.2 - Wyświetl stronę ze szczegółami leku

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	MedicationController	MedicationManager	MedicationFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	getMedicationsDetails	getMedicationDetails	findAndRefresh

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie	medication/{id}	GET	201 (Created)	scenariusz pozytywny
Negatywny - wykonanie metody http innej niż GET	medication/{id}	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywny - podanie id nie znajdującego się w bazie	medication/{id}	GET	404 (Not found)	exception.entity.not.found
Negatywne - błąd wewnętrzny serwera	medication/{id}	GET	500 (Internal Server Error)	exception.general

Tj. w przypadku użycia MOA1. Logujemy się → Lista leków → Szczegóły.

Penicylina

Cena Ilość
10 10

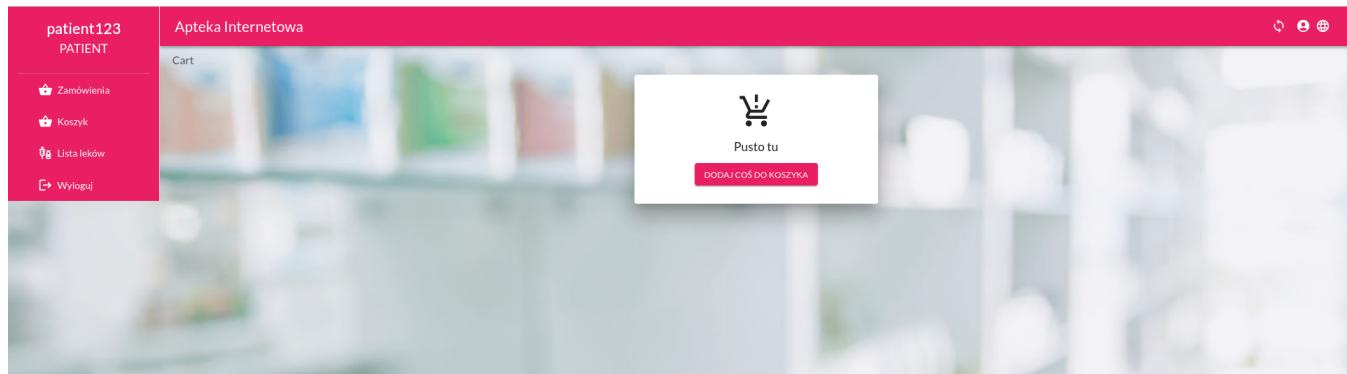
Nazwa kategorii
antibiotics

Na receptę?
Tak

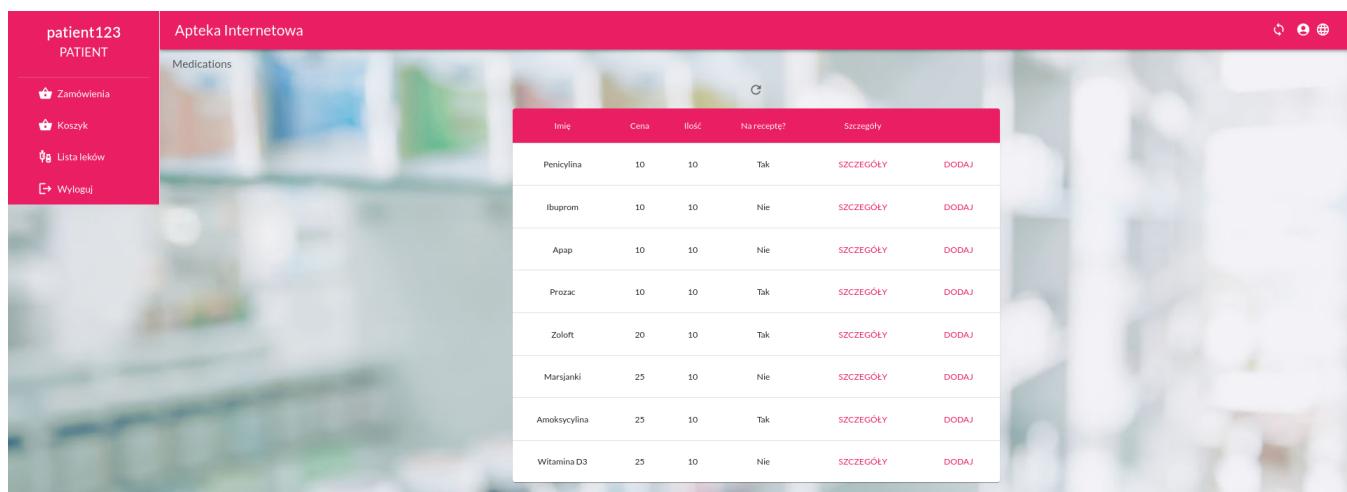
MOA.3 Dodaj lek do koszyka

Realizowane z poziomu widoku.

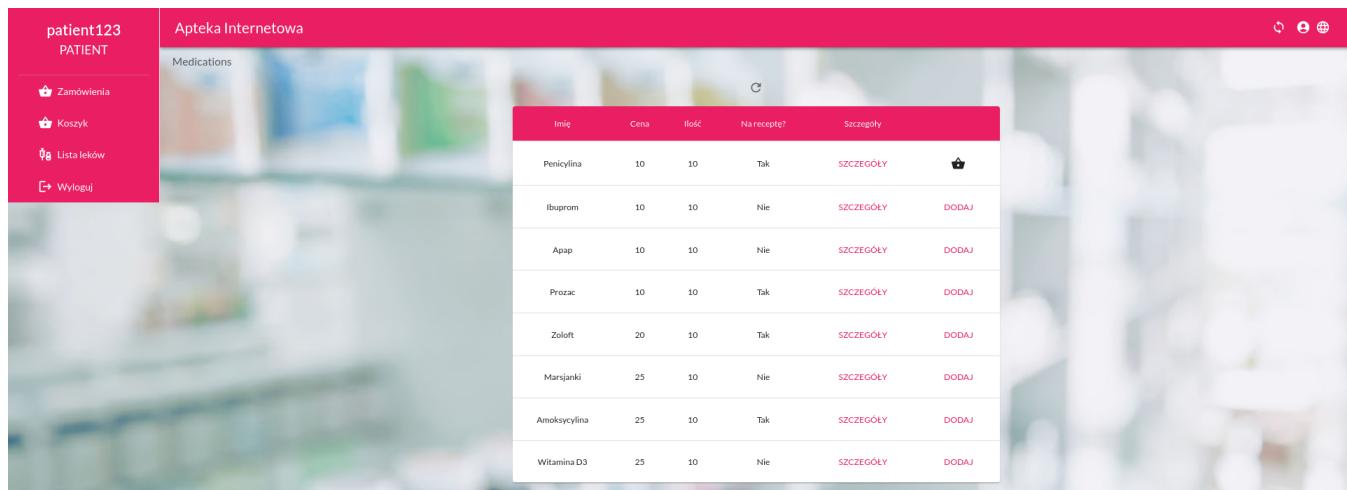
Klikamy koszyk.



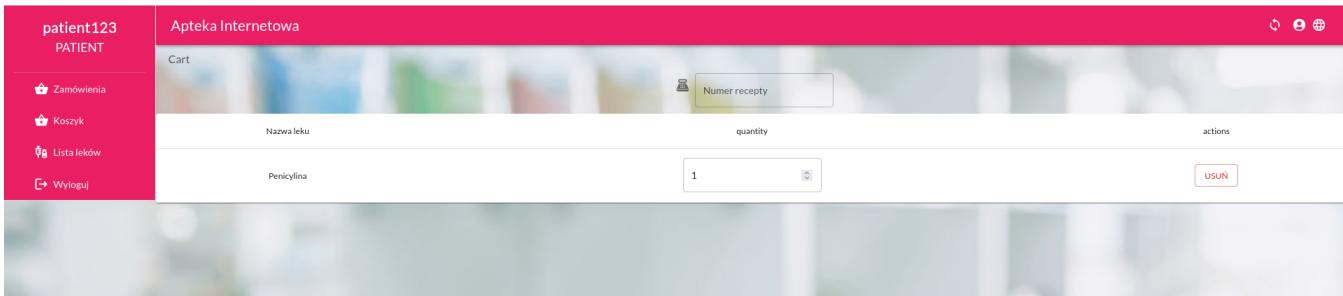
Klikamy dodaj coś do koszyka.



Klikamy dodaj.



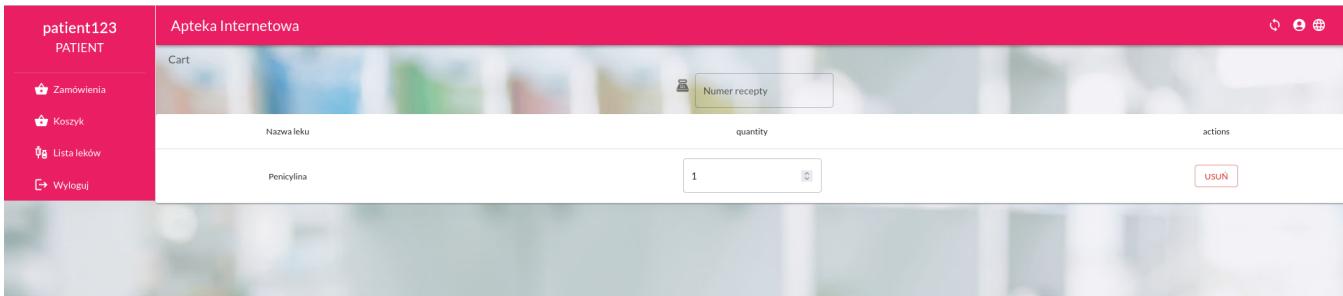
Po dodaniu leku widzimy ikonkę koszyka oznaczającą, że lek znajduje się w koszyku.



MOA.4 Wyświetl koszyk

Realizowane z poziomu widoku.

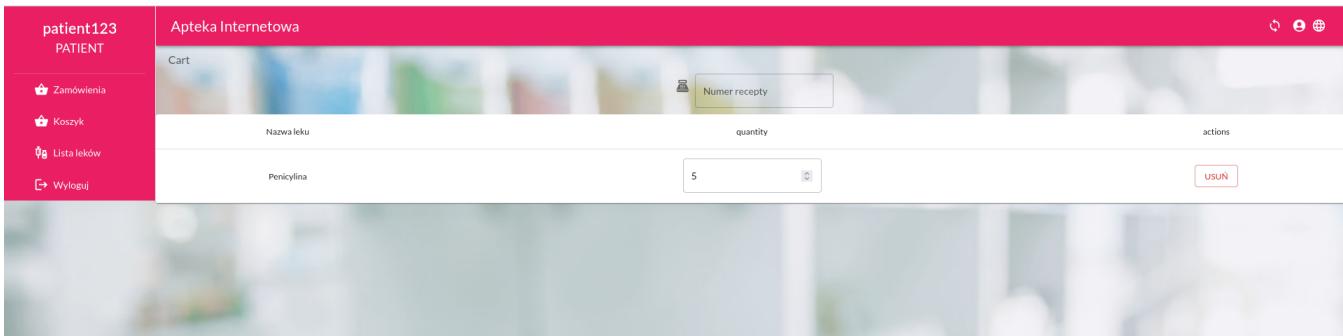
Klikamy koszyk



MOA.5 Zmień liczebność leku w koszyku

Realizowane z poziomu widoku.

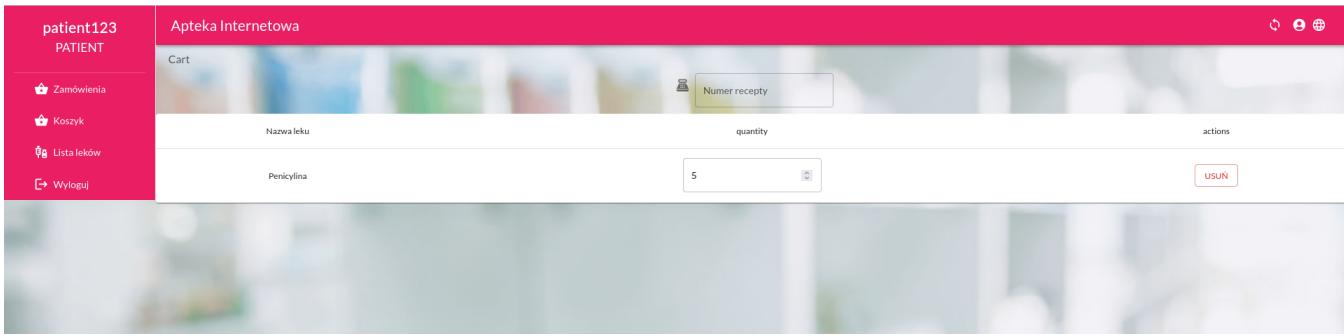
Wykonane poprzez modyfikację pola quantity.



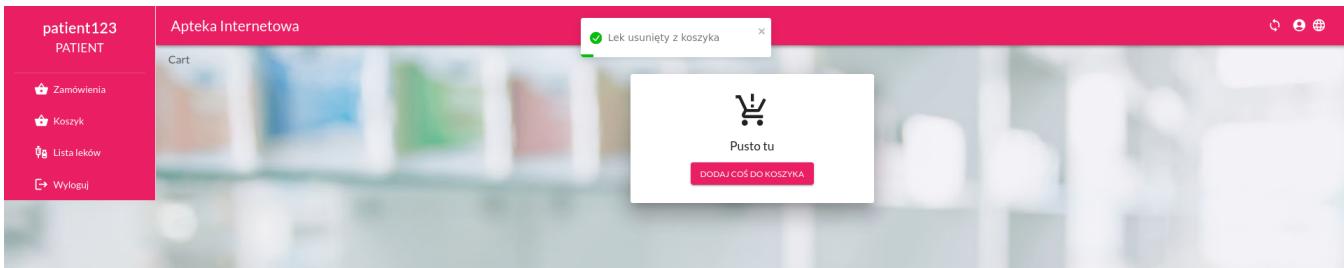
MOA.6 Usuń lek z koszyka

Realizowane z poziomu widoku.

Tj. poprzednio klikamy koszyk i usuń.



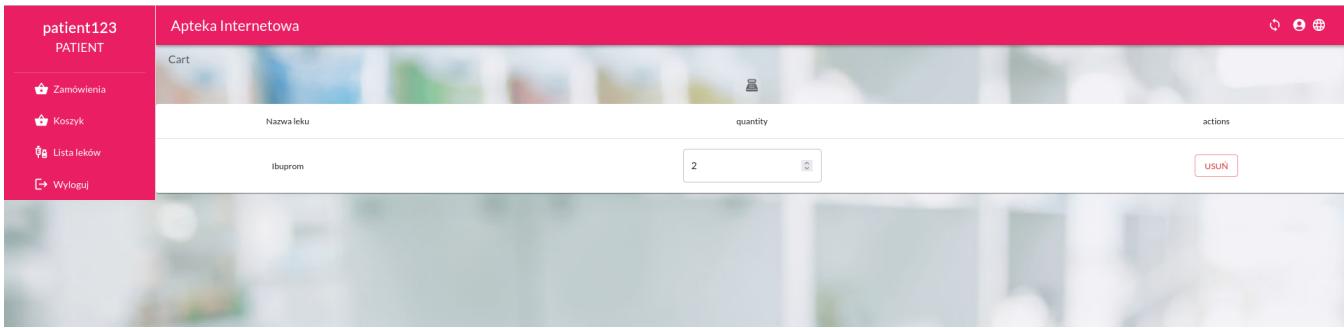
Po usunięciu i potwierdzeniu.



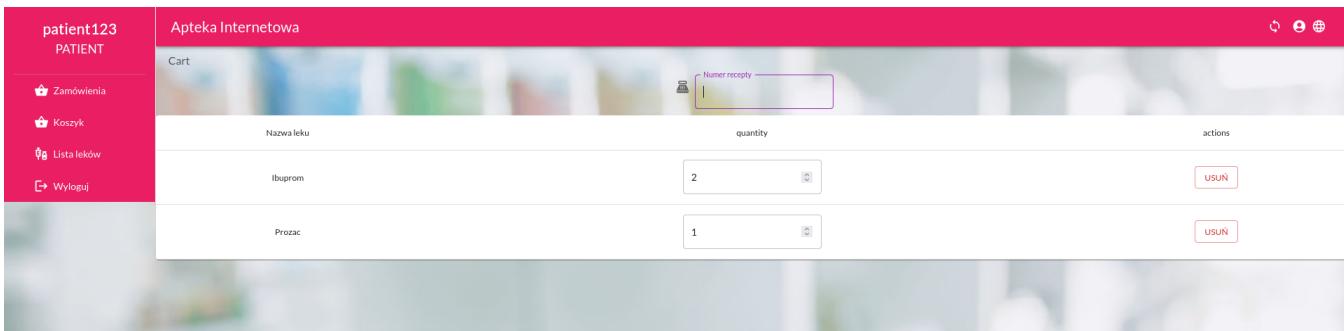
MOA.7 Złoż zamówienie

	Controller	Manager	Manager	Facade	Facade	Manager	Manager	Manager	Facade
Kolejność wywołania metod	1	2	3	4	5	6	7	8	9
Komponent	OrderController	OrderManager	AccountManager	AccountFacade	MedicationFacade	OrderMedication	OrderMedication	OrderMedication	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateful	Stateless	Stateless	Stateful	Stateful	Stateful	Stateless
Metoda	submitOrder	createOrder	getCurrentUserWithAccessLevel	findByLoginAndRefresh	findByName	checkIsOnPrescription	checkAllMedicationsAvailable	decreaseMedicationStock	create

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne złożenie zamówienia	order/submit	POST	201 (Created)	scenariusz pozytywny
Negatywne - brak wymaganego ciała zapytania	order/submit	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywne - nieprawidłowe dane (np. przekazanie leku, który nie istnieje)	order/submit	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywny - brak podanego numeru recepty, a w zamówieniu jest lek na receptę	order/submit	POST	400 (Bad Request)	exception.order.prescription.required
Negatywny - niepoprawna metoda HTTP	order/submit	GET	405 (Method Not Allowed)	exception.method.not.allowed
Negatywny - wykonanie nie będąc zalogowanym	order/submit	POST	401(Unauthorized)	brak - domyślana strona błędu serwera Payara
Negatywny - błąd wersji	order/submit	POST	409 (Conflict)	exception.optimistic-lock
Negatywne - błąd wewnętrzny serwera	order/submit	POST	500 (Internal Server Error)	exception.general



Zamówienie zostanie stworzone po kliknięciu ikonki kasy i potwierdzeniu akcji (jeżeli wśród zamówionych leków jest lek na receptę należy wprowadzić jeszcze numer recepty).



MOA.8 Wycofaj złożone zamówienie

a) Odrzucenie zamówienia

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4
Komponent	OrderController	OrderManager	OrderFacade	OrderFacade
Cykł życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	withdrawOrderById	withdrawOrder	find	withdrawOrder

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne odrzucenie zamówienia	order/{id} /withdraw	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba odrzucenia nieistniejącego zamówienia	order/{id} /withdraw	PUT	404 (Not found)	exception.entity.not.found
Negatywne - próba odrzucenia zamówienia nie będącego w stanie do zatwierdzenia przez pacjenta	order/{id} /withdraw	PUT	403 (Forbidden)	exception.no.permission.to_delete_order
Negatywne - próba odrzucenia przez nieuprawnioną rolę	order/{id} /withdraw	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - błąd wewnętrzny serwera	order/{id} /withdraw	PUT	500 (Internal Server Error)	exception.general

Klikamy w zamówienia.

Stan zamówienia	Data zamówienia	Nazwa leku	Cena leku	Ilość leku	Cena całkowita zamówienia	Numer recepty	Zatwierdż zamówienie	Usuń
W kolejce	21-04-2023 12:43:01	Percytyna	10	2				
		Arop	10	4	80	123456789	<input checked="" type="checkbox"/>	<input type="checkbox"/>
		Prozic	10	2				
W kolejce	21-04-2023 12:43:01	Ibuprofen	10	5				
		Marcinka	25	1	75			
W kolejce	21-04-2023 12:43:01	Marcinka	25	5				
		Zelent	20	4	80	123456788	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Oczekuje na zatwierdzenie przez aptekarza	21-04-2023 12:43:01	Ibuprofen	10	4	40	123456787		
Oczekuje na zatwierdzenie przez aptekarza	21-04-2023 12:43:01	Ibuprofen	10	4	40	123456786		
Do zatwierdzenia przez pacjenta	21-04-2023 12:43:01	Ibuprofen	10	4	40	123456783	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Do zatwierdzenia przez pacjenta	21-04-2023 12:43:01	Ibuprofen	10	4	40	123456784	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Jeżeli zamówienie jest w stanie do zaakceptowania przez pacjenta w kolumnie usuń zamówienie pojawia się ikonka kosza po kliknięciu której i zaakceptowanie akcji zamówienie zostaje odrzucone.

b) Zaakceptowanie zamówienia

	Controller	Manager	Facade	Facade	Manager
Kolejność wywołania metod	1	2	3	4	2
Komponent	OrderController	OrderManager	OrderFacade	OrderFacade	OrderManager
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless	Stateful
Metoda	approvedByPatient	approvedByPatient	find	approvedByPatient	decreaseMedicationStock

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne zaakceptowanie zamówienia	order/{id}/patient-approve	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba zaakceptowania nieistniejącego zamówienia	order/{id}/patient-approve	PUT	403 (Forbidden)	exception.order_not_found
Negatywne - próba zaakceptowania zamówienia nie będącego w stanie do zatwierdzenia przez pacjenta	order/{id}/patient-approve	PUT	403 (Forbidden)	exception.no.permission_to_approve_order
Negatywne - próba odrzucenia przez nieuprawnioną rolę	order/{id}/patient-approve	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - błąd wewnętrzny serwera	order/{id}/patient-approve	PUT	500 (Internal Server Error)	exception.general

Klikamy w zamówienia.

Jeżeli zamówienie jest w stanie do zaakceptowania przez pacjenta w kolumnie zatwierdż zamówienie pojawia się ikonka potwierdzenia po kliknięciu której i zaakceptowania akcji zamówienie zostaje zrealizowane bądź wysłane do zatwierdzenia przez farmaceutę.

MOA.9 Wyświetl kolejkę zamówień oczekujących

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	OrderController	OrderManager	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	getWaitingOrders	getWaitingOrders	findWaitingOrders

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacionalizacji
Pozytywne wyświetlenie zamówień	order/waiting	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	order/waiting	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	order/waiting	GET	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	order/waiting	GET	500 (Internal Server Error)	exception.general

Po zalogowaniu jako farmaceuta i kliknięciu zamówienia oczekującego.

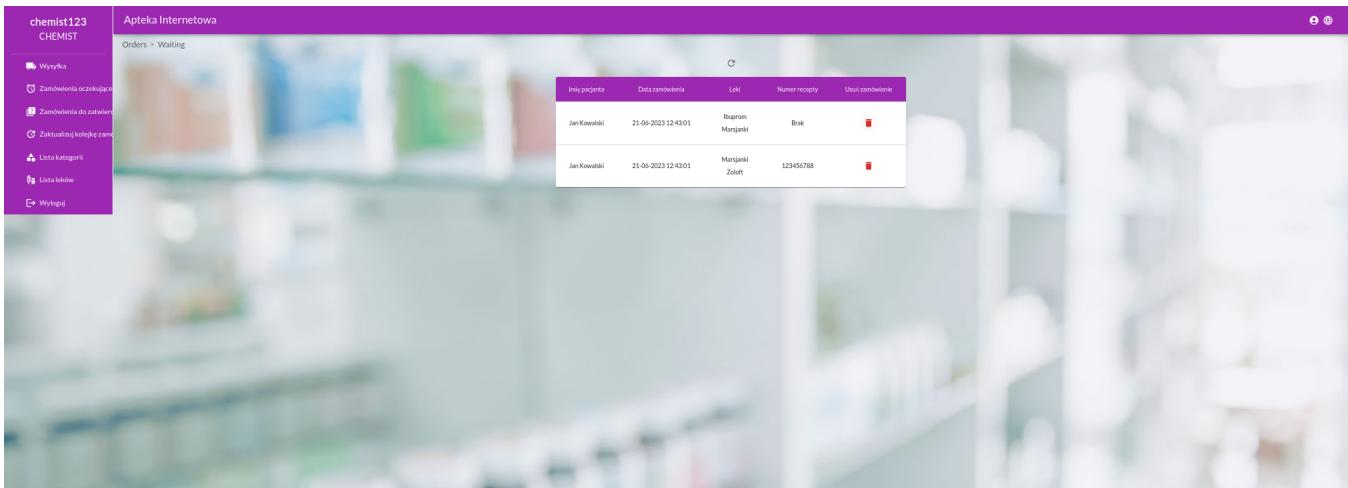
MOA.10 Usuń zamówienia z kolejki zamówień oczekujących

	Controller	Manager	Facade	Facade

Kolejność wywołania metod	1	2	3	4
Komponent	OrderController	OrderManager	OrderFacade	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	deleteWaitingOrdersById	deleteWaitingOrdersById	find	deleteWaitingOrdersById

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne usunięcie zamówienia	order/{id}/waiting	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba usunięcia nieistniejącego zamówienia	order/{id}/withdraw	PUT	404 (Not found)	exception.entity.not.found
Negatywne - próba odrzucenia zamówienia nie będącego w stanie do zatwierdzenia przez pacjenta	order/{id}/withdraw	PUT	403 (Forbidden)	exception.order_not_in_queue
Negatywne - próba usunięcia nie będąc farmaceutą	order/{id}/withdraw	PUT	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	order/{id}/waiting	PUT	500 (Internal Server Error)	exception.general

Po przejęciu do zadań oczekujących i kliknięciu ikonki kosza oraz zatwierdzeniu akcji zamówienie zostaje usunięte z kolejki.



MOA.11 Wprowadź zawartość dostawy

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4
Komponent	ShipmentController	ShipmentManager	MedicationFacade	ShipmentFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	createShipment	createShipment	findByName	create

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne stworzenie dostawy	shipment/createShipment	POST	201 (Created)	scenariusz pozytywny
Negatywne - nieprawidłowa wartość Etag	shipment/createShipment	POST	400 (Bad Request)	exception.etag.not-valid
Negatywne - błąd wersji	shipment/createShipment	POST	409 (Conflict)	exception.optimistic-lock
Negatywne - błąd metody http	shipment/createShipment	GET	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	shipment/createShipment	POST	403 (Forbidden)	exception.access_denied

Negatywne - brak wymaganego ciała zapytania	shipment/createShipment	POST	400 (Bad Request)	brak - domyślana strona błędu serwera Payara
Negatywne - błąd wewnętrzny serwera	shipment/createShipment	POST	500 (Internal Server Error)	exception.general

Po kliknięciu w wysyłka.

Po kliknięciu w wybierz lek.

Listę leków można rozszerzać przyciskiem pokaż więcej bądź filtrować po nazwie leków wpisując nazwę leku w polu nazwa leku.

Po kliknięciu utwórz nowy lek.

Po kliknięciu ikony kalendarza.

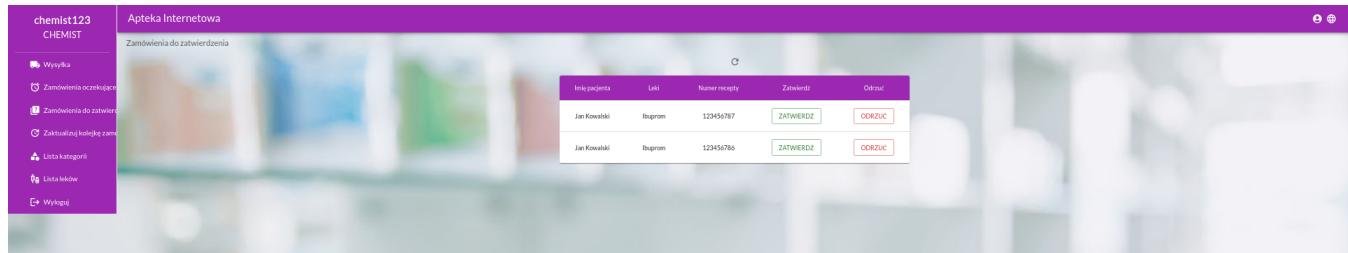
Datę dostawy można również wpisać ręcznie.

MOA.12 Wyświetl listę zamówień oczekujących na potwierdzenie

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	OrderController	OrderManager	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	getOrdersToApprove	getOrdersToApprove	findNotYetApproved

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie zamówień	order/to-approve	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	order/to-approve	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	order/to-approve	GET	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	order/to-approve	GET	500 (Internal Server Error)	exception.general

Klikamy w zamówienia do zatwierdzenia.



MOA.13 Potwierdź zamówienie

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4
Komponent	OrderController	OrderManager	OrderFacade	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	approveOrder	approveOrder	find	edit

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne potwierdzenie zamówienia	order/{id}/approve	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba zatwierdzenia nieistniejącego zamówienia	order/{id}/approve	PUT	404 (Not found)	exception.entity.not.found

Negatywne - wykonanie metody innej niż PUT	order/{id}/approve	GET	405 (Method not allowed)	exception.method.not.allowed
Negatywne - zły stan zamówienia	order/{id}/approve	PUT	400 (Bad request)	exception.order.illegal-state.modification
Negatywne - nieupoważniona rola	order/{id}/approve	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - błąd wewnętrzny serwera	order/{id}/approve	PUT	500 (Internal server error)	exception.general

Przechodzimy do zamówień do zatwierdzenia i klikamy zatwierdź po potwierdzeniu akcji zamówienie przechodzi w stan sfinalizowane.

MOA.14 Odwołaj zamówienie

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4
Komponent	OrderController	OrderManager	OrderFacade	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	cancelOrder	cancelOrder	find	cancelOrder

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne odwołanie zamówienia	order/{id}/cancel	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba odwołania nieistniejącego zamówienia	order/{id}/cancel	PUT	404 (Not found)	exception.entity.not.found
Negatywne - wykonanie metody innej niż PUT	order/{id}/cancel	GET	405 (Method not allowed)	exception.method.not.allowed
Negatywne - zły stan zamówienia	order/{id}/cancel	PUT	400 (Bad request)	exception.order.illegal-state.modification
Negatywne - nieupoważniona rola	order/{id}/cancel	PUT	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - błąd wewnętrzny serwera	order/{id}/cancel	PUT	500 (Internal server error)	exception.general

Przechodzimy do zamówień do zatwierdzenia i klikamy odrzuć po potwierdzeniu akcji zamówienie przechodzi w stan odrzucone przez farmaceutę.

MOA.15 Pokaż statystyki zakupów

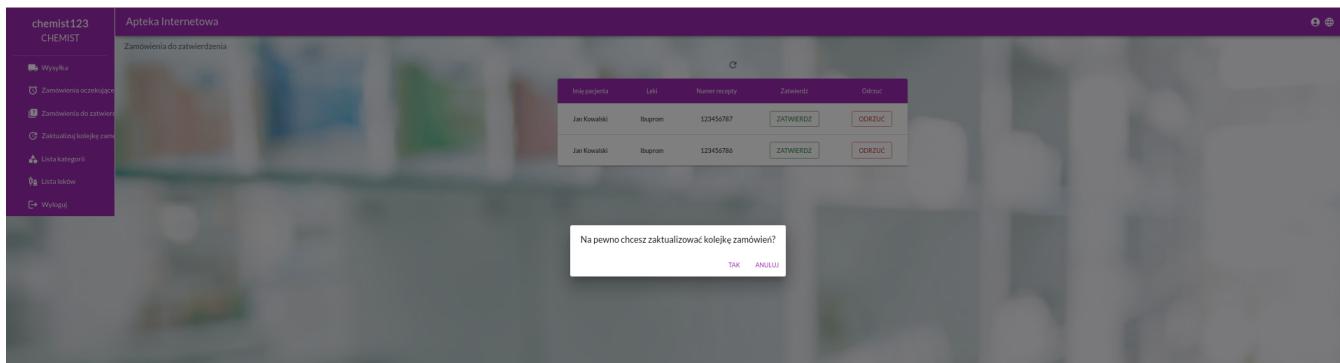
Brak Implementacji.

MOA.16 Cyklicznie sprawdź kolejkę i aktualizuj ją

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	3
Komponent	OrderController	OrderManager	OrderFacade	ShipmentFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	updateQueue	updateQueue	findAllOrdersInQueueSortByOrderDate	findAllNotAlreadyProcessed

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywna aktualizacja kolejki	order/update-queue	PUT	204 (No content)	scenariusz pozytywny
Negatywne - niezgodność wersji	order/update-queue	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - błąd wewnętrzny serwera	order/update-queue	PUT	500 (Internal Server Error)	exception.general

Po kliknięciu zaktualizuj kolejkę zadań oczekujących.



Ta opcja została dodana na potrzeby prezentacji.

MOA.17 Wyświetl listę złożonych przez siebie zamówień

	Controller	Manager	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4	5
Komponent	OrderController	OrderManager	AccountManager	AccountFacade	OrderFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateful	Stateless	Stateless
Metoda	getAllOrdersForSelf	getAllOrdersForSelf	getCurrentUserWithAccessLevel	findByLoginAndRefresh	findAllByPatientId

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie zamówień	order/self	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	order/self	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	order/self	GET	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	order/self	GET	500 (Internal Server Error)	exception.general

Po kliknięciu zamówienia z poziomu pacjenta.

MOA.18 Wyświetl listę wszystkich zamówień

Brak Implementacji.

MOA.19 Dodaj lek do oferty

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	4
Komponent	MedicationController	MedicationManager	CategoryFacade	MedicationFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	addMedication	createMedication	findByName	create

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Clucz internacjonalizacji
Pozytywne dodanie leku	medication/add-medication	POST	200 (Ok)	scenariusz pozytywny
Negatywne - brak wymaganego ciała zapytania	medication/add-medication	POST	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - nieprawidłowe dane leku	medication/add-medication	POST	400 (Bad Request)	brak - domyślna strona błędu serwera Payara
Negatywne - błąd metody http	medication/add-medication	GET	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	medication/add-medication	POST	403 (Forbidden)	exception.access_denied
Negatywne - błąd wewnętrzny serwera	medication/add-medication	POST	500 (Internal Server Error)	exception.general

Jako farmaceuta klikamy lista leków.

Po kliknięciu dodaj następuje zmiana ikonki na koszyk i dodanie leku do oferty.

MOA.20 Edytuj dane leku

Brak Implementacji.

MOA.21 Dodaj kategorie

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	3
Komponent	CategoryController	CategoryManager	CategoryFacade	CategoryFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	addCategory	createCategory	findByName	create

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Poztywne dodanie kategorii	category/add-category	POST	200 (Ok)	scenariusz pozatywny
Negatywne - kategoria już istnieje	category/add-category	POST	409 (Conflict)	exception.category.already-exists
Negatywne - nieprawidłowe dane kategorii	category/add-category	POST	400 (Bad request)	brak - domyślana strona błędu serwera Payara
Negatywne - nieupoważniona rola	category/add-category	POST	403 (Forbidden)	exception.access_denied
Negatywne - błąd metody http	category/add-category	GET	405 (Method not allowed)	exception.method.not.allowed
Negatywne - błąd wewnętrzny serwera	category/add-category	POST	500 (Internal Server Error)	exception.general

Po kliknięciu lista kategorii jako farmaceuta.

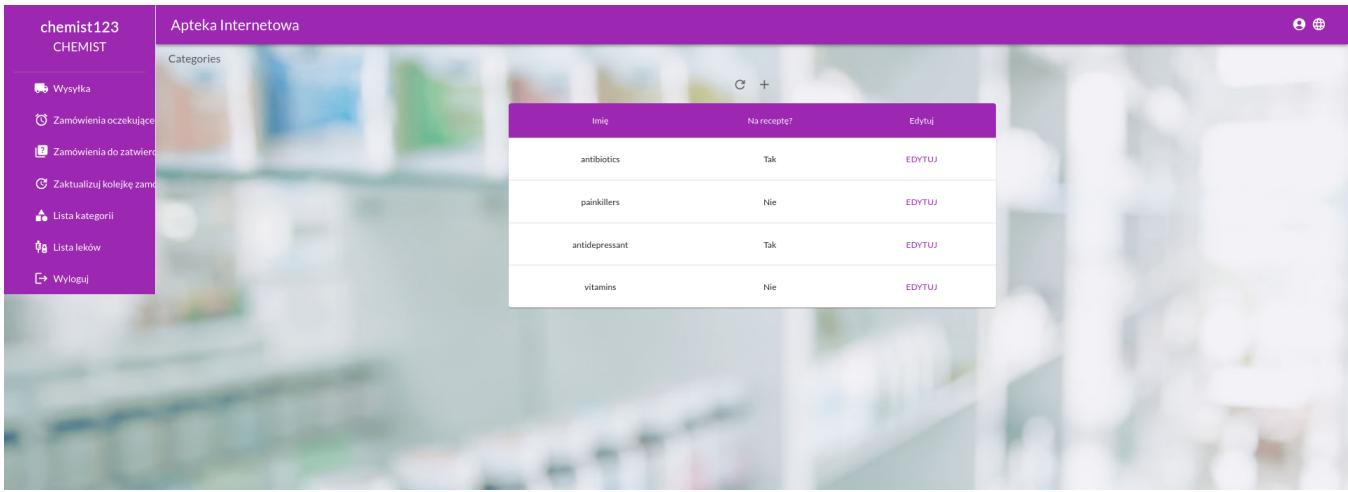
Klikamy ikonkę plus.

MOA.22 Wyświetl kategorie

	Controller	Manager	Facade
Kolejność wywołania metod	1	2	3
Komponent	CategoryController	CategoryManager	CategoryFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless
Metoda	getAllCategories	getAllCategories	findAll

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywne wyświetlenie kategorii	category/	GET	200 (Ok)	scenariusz pozytywny
Negatywne - błąd metody http	category/	POST	405 (Method Not Allowed)	exception.method.not.allowed
Negatywne - nieupoważniona rola	category/	GET	403 (Forbidden)	exception.access.denied
Negatywne - błąd wewnętrzny serwera	category/	GET	500 (Internal Server Error)	exception.general

Po kliknięciu lista kategorii jako farmaceuta.

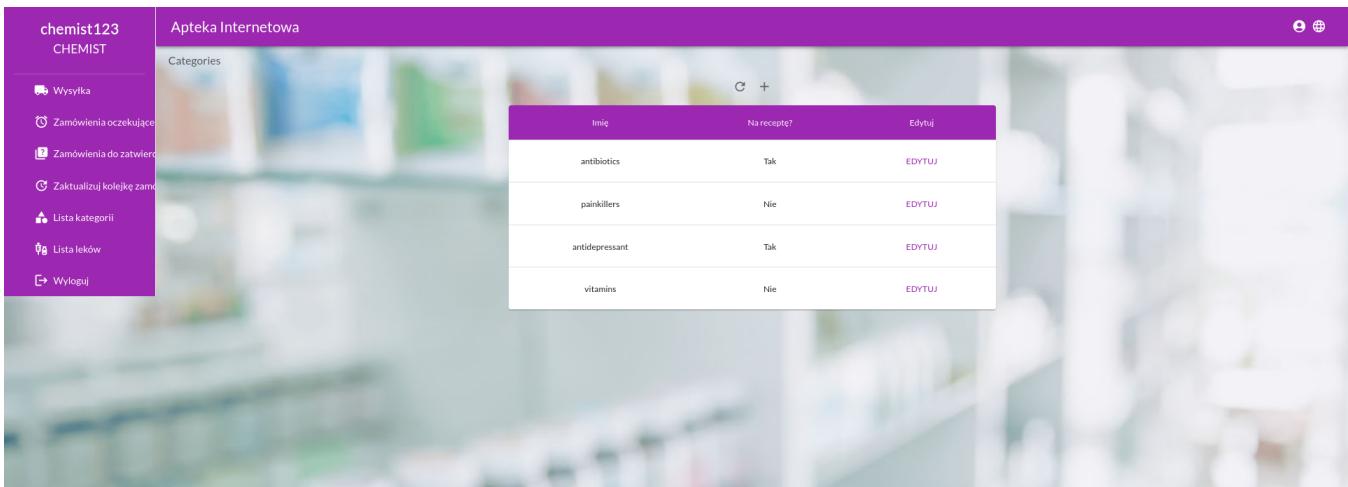


MOA.23 Edytuj kategorie

	Controller	Manager	Facade	Facade
Kolejność wywołania metod	1	2	3	3
Komponent	CategoryController	CategoryManager	CategoryFacade	CategoryFacade
Cykl życia	Stateful, RequestScoped	Stateful	Stateless	Stateless
Metoda	editCategory	editCategory	find	edit

Przypadek użycia	Kontekst URI	Metoda HTTP	Kod odpowiedzi HTTP	Klucz internacjonalizacji
Pozytywna edycja kategorii	category/{id}/edit-category	PUT	200 (Ok)	scenariusz pozytywny
Negatywne - próba edycji nieistniejącej kategorii	category/{id}/edit-category	PUT	404 (Not found)	exception.entity.not.found
Negatywne - wykonanie metody innej niż PUT	category/{id}/edit-category	GET	405 (Method not allowed)	exception.method.not.allowed
Negatywne - niezgodność wersji	category/{id}/edit-category	PUT	409 (Conflict)	exception.optimistic-lock
Negatywne - błąd wewnętrzny serwera	category/{id}/edit-category	PUT	500 (Internal server error)	exception.general

Po kliknięciu lista kategorii jako farmaceuta.



Po kliknięciu edytuj.

Apteka Internetowa

Categories > Edytuj

Edytuj kategorię

Imię * antibiotics

is_on_prescription * Tak

Tak

Nie

EDYTUJ KATEGORIE

21 Zmiany - projekt końcowy

W rozdziale tym należy wymienić zmiany wprowadzone do sprawozdania wstępniego i szczegółowego, zwięzłe je charakteryzując. Dla każdej zmiany wymagane jest zamieszczenie odwołania do strony poddanej modyfikacjom.

5 **Identyfikacja obiektów encji** (jako, iż zmiany opisane w tym punkcie wyjaśniają zmianę występującą także w strukturach relacyjnej bazy danych, kolejność rozdziałów w niniejszym opisie została zamieniona miejscami)

1. Dodanie do encji *Token* wartości typu wyliczeniowego *TokenType* informującego o typie tokenu (weryfikacja konta, resetowanie hasła oraz potwierdzenie zmiany adresu email)
2. Zamiana pola *inQueue* w klasie *Order* na typ wyliczeniowy *OrderState* reprezentujący stan zamówienia (utworzone, w kolejce, oczekujące na zatwierdzenie przez pacjenta, oczekujące na zatwierdzenie przez farmaceutę, sfinalizowane, odrzucone przez pacjenta, odrzucone przez farmaceutę, zatwierdzone przez pacjenta)
3. Dodanie do klasy *ShipmentMedication* pola *processed*, informującego czy dana dostawa została już wprowadzona.
4. Dodanie do klasy *OrderMedication* pola *purchasePrice*, stanowiącego informacje o cenie zakupu danego leku.

3 Struktury relacyjnej bazy danych

```
CREATE TABLE token
(
    id          BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL,
    version      BIGINT,
    creation_date  TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    modification_date  TIMESTAMP WITHOUT TIME ZONE,
    created_by    VARCHAR(255),
    modified_by    VARCHAR(255),
    token_type    VARCHAR(255)           NOT NULL,
    code          VARCHAR(100)          NOT NULL,
    used          BOOLEAN DEFAULT FALSE        NOT NULL,
    was_previous_token_sent BOOLEAN DEFAULT FALSE      NOT NULL,
    account_id    BIGINT              NOT NULL,
    expiration_date  TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    CONSTRAINT pk_token PRIMARY KEY (id)
);
```

```
CREATE TABLE order_medication
(
    id          BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL,
    version      BIGINT,
    creation_date  TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    modification_date  TIMESTAMP WITHOUT TIME ZONE,
    created_by    VARCHAR(255),
    modified_by    VARCHAR(255),
    purchase_price DECIMAL,
    order_id      BIGINT          NOT NULL,
    medication_id BIGINT          NOT NULL,
    quantity      INTEGER         NOT NULL,
    CONSTRAINT pk_order_medication PRIMARY KEY (id)
);
```

```

CREATE TABLE shipment_medication
(
    id          BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL,
    version      BIGINT,
    creation_date  TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    modification_date  TIMESTAMP WITHOUT TIME ZONE,
    created_by    VARCHAR(255),
    modified_by   VARCHAR(255),
    shipment_id   BIGINT              NOT NULL,
    processed     BOOLEAN             NOT NULL,
    medication_id BIGINT              NOT NULL,
    quantity      INTEGER             NOT NULL,
    CONSTRAINT pk_shipment_medication PRIMARY KEY (id)
);

```

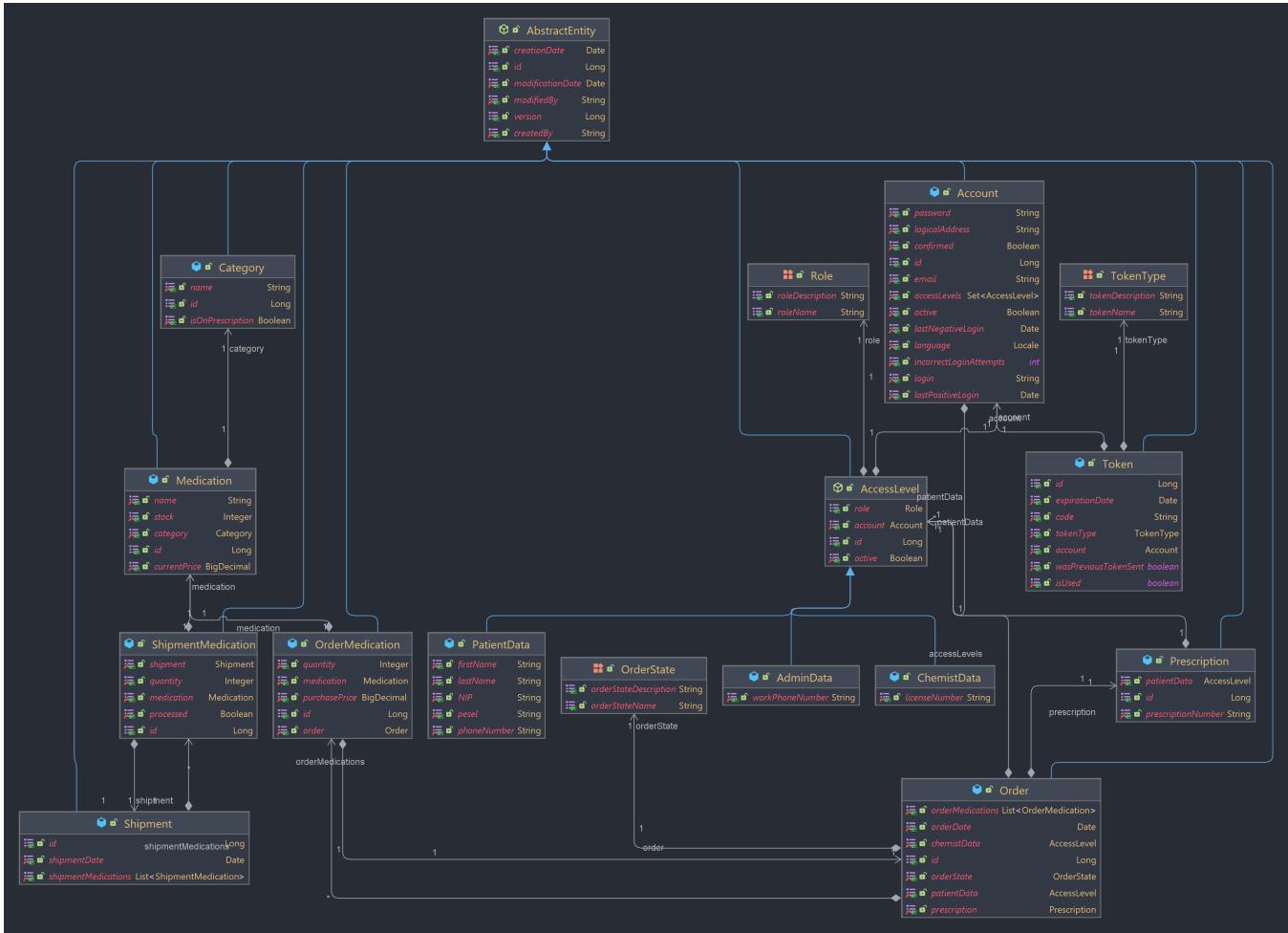
```

CREATE TABLE patient_order
(
    id          BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL,
    version      BIGINT,
    creation_date  TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    modification_date  TIMESTAMP WITHOUT TIME ZONE,
    created_by    VARCHAR(255),
    modified_by   VARCHAR(255),
    order_date    TIMESTAMP WITHOUT TIME ZONE      NOT NULL,
    order_state   VARCHAR(255)        NOT NULL,
    prescription_id BIGINT,
    patient_data_id BIGINT              NOT NULL,
    chemist_data_id BIGINT,
    CONSTRAINT pk_patient_order PRIMARY KEY (id)
);

```

6 Diagram klas encji

Z oczywistych przyczyn, także diagram klas encji uległ zmianie - poniżej zaktualizowany diagram klas encji



W 10 Model bezpieczeństwa komponentów EJB/CDI zaktualizowano deskryptory web.xml oraz glassfish_web.xml oraz zaktualizowano tabelę Schemat bezpieczeństwa komponentów EJB dla MOA.

13 Interfejs użytkownika

W porównaniu do wcześniejszego szablonu strony użytkownika zalogowanego, w którego skład wchodził jedynie komponent typu *navbar*, dla zwiększenia czytelności wprowadzono także komponent typu *sidebar*, gdzie znajdują się akcje dostępne dla wybranego poziomu dostępu.