



AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

INŻYNIERIA OPROGRAMOWANIA

DOKUMENT SPECYFIKACYJNY

Wish List

Autorzy:

Łukasz GAJEWSKI

Grzegorz SIATKA

Patrycja WRONA

2016

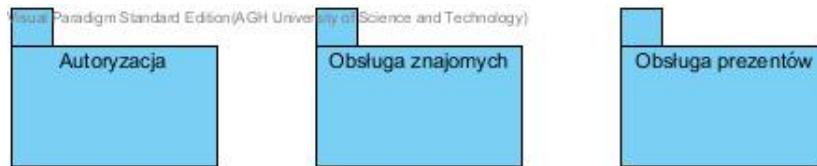
Spis treści

1	Przypadki użycia	2
1.1	Diagramy przypadków użycia	2
1.1.1	Podsystemy	2
1.1.2	Autoryzacja	2
1.1.3	Obsługa prezentów	6
1.1.4	Obsługa znajomych	9
2	Interfejsy	12
2.1	Moduł reprezentacji danych	12
2.2	Moduł persystencji	13
2.3	Moduł autoryzacji użytkownika	15
2.4	Moduł webowy, komunikacji z użytkownikiem	17
2.5	Moduł mobilny, komunikacji z użytkownikiem	18
3	Baza danych	20
3.1	Diagram encji	20
3.2	Użytkownik	20
3.3	Prezent	20
4	Diagramy maszyn stanowych	21
4.1	Maszyna stanowa dla dodawania użytkownika	21
4.2	Maszyna stanowa dla zmiany hasła	22
5	Adresowanie	22
5.1	Użytkownicy	22
5.2	Prezenty	23
5.3	Znajomi	23
6	Moduł mailowy	23
6.1	Protokół SMTP	23

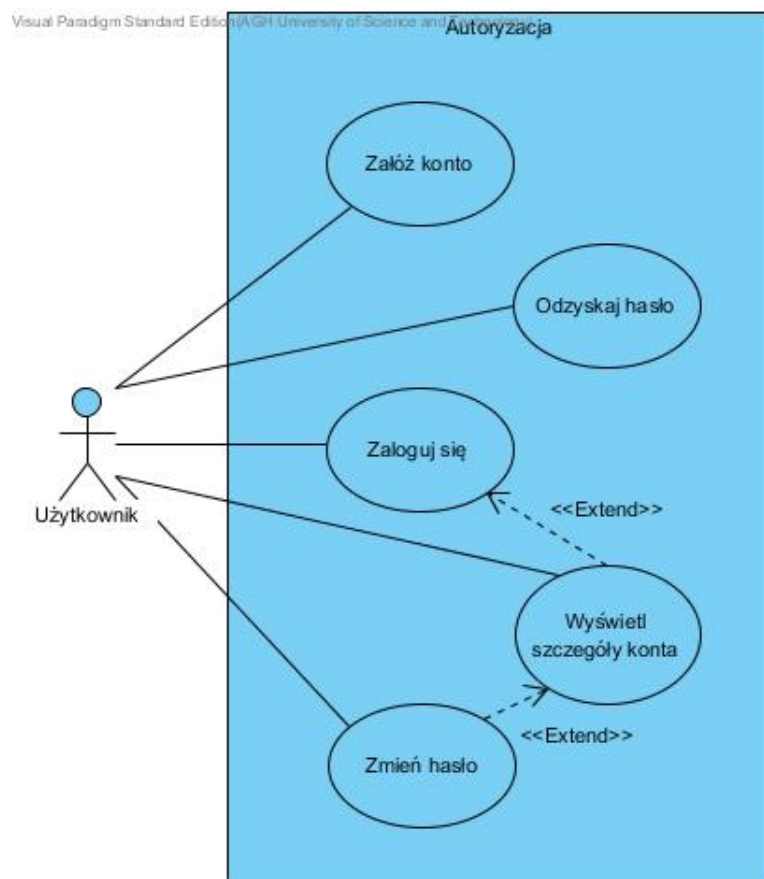
1 Przypadki użycia

1.1 Diagramy przypadków użycia

1.1.1 Podsystemy



1.1.2 Autoryzacja



TYTUŁ: **Założ konto**

AUTOR: Grzegorz Siatka

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik nie jest jeszcze zarejestrowany w systemie

KONTEKST: Dodanie użytkownika do systemu

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Załącz konto*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz zakładania konta.
2. Użytkownik wprowadza dane.
3. Po potwierdzeniu poprawności wprowadzonych danych, utworzone jest konto dla użytkownika w systemie.

SCENARIUSZ ALTERNATYWNY:

- 3a. W przypadku gdy podana w formularzu nazwa użytkownika znajduje się już w systemie, wyświetlany jest komunikat o błędzie, użytkownik nie zostaje dodany do systemu i wracamy do punktu 1.

TYTUŁ: Odzyskaj hasło

AUTOR: Patrycja Wrona

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie

KONTEKST: Odzyskanie hasła użytkownika

MINIMALNA GWARANCJA: Baza danych pozostanie spójna i nienaruszona

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Przypomnij hasło*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz odzyskiwania hasła.
2. System wysyła maila z linkiem do strony z formularzem zmiany hasła na podany adres e-mail.
3. System wyświetla użytkownikowi formularz zmiany hasła.
4. Użytkownik wprowadza dane.
5. Nowe hasło użytkownika zostaje zapisane do bazy. Użytkownik jest zalogowany.

SCENARIUSZ ALTERNATYWNY:

2a. W przypadku gdy podany w formularzu adres e-mail nie znajduje się w systemie, wyświetlany jest komunikat o błędzie i wracamy do punktu 1.

TYTUŁ: Zaloguj się

AUTOR: Grzegorz Siatka

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie

KONTEKST: Zalogowanie użytkownika w celu skorzystania z systemu

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Zaloguj się*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz logowania.
2. Użytkownik wprowadza dane logowania.
3. Po potwierdzeniu poprawności wprowadzonych danych i sprawdzeniu istnienia konta w bazie użytkownik zostaje zalogowany do systemu.

SCENARIUSZ ALTERNATYWNY:

3a. W przypadku gdy podana w formularzu nazwa użytkownika nie znajduje się w systemie lub gdy wprowadzone zostało błędne hasło, wyświetlany jest komunikat o błędzie, użytkownik nie zostaje zalogowany do systemu i wracamy do punktu 1.

TYTUŁ: Wyświetl szczegóły konta

AUTOR: Łukasz Gajewski

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie

KONTEKST: Wyświetlenie profilu użytkownika w systemie

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Profil*

GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi szczegółowe dane konta.

TYTUL:**Zmień hasło**

AUTOR: Patrycja Wrona

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie

KONTEKST: Zmiana hasła użytkownika

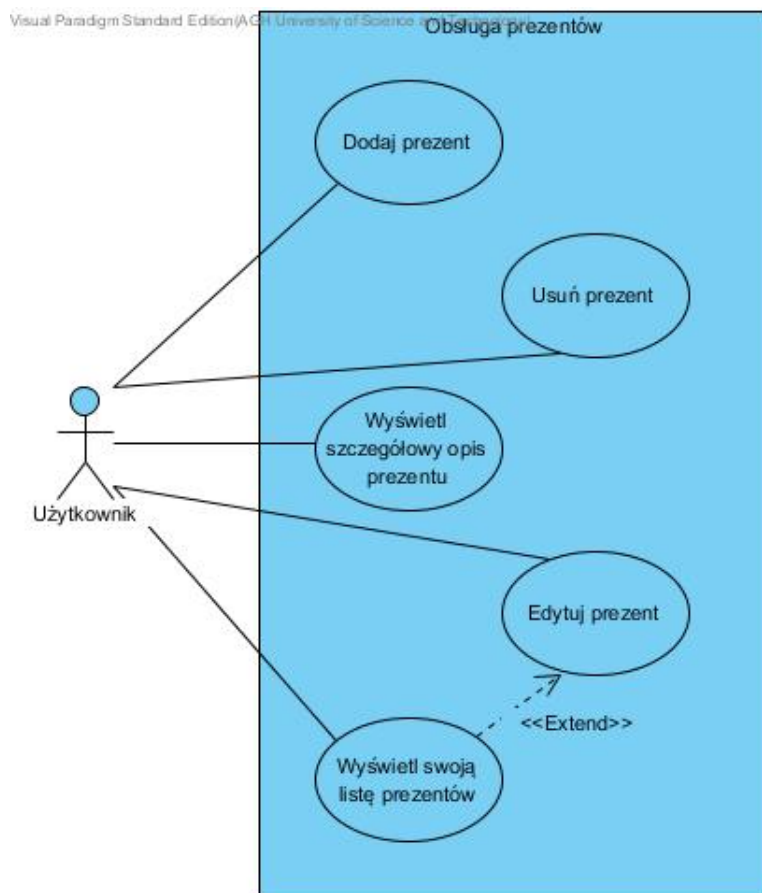
MINIMALNA GWARANCJA: Baza danych pozostanie spójna, brak zmian w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Zmień hasło*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz zmiany hasła.
2. Użytkownik wprowadza dane.
3. Nowe hasło użytkownika zostaje zapisane do bazy.

1.1.3 Obsługa prezentów



TYTUŁ:Dodaj prezent

AUTOR: Łukasz Gajewski

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie

KONTEKST: Dodanie prezentu do listy prezentów użytkownika

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Dodaj prezent*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz dodawania prezentu.
2. Użytkownik wprowadza nazwę i opis prezentu.
3. Po potwierdzeniu poprawności wprowadzonych danych, prezent zostaje dodany do listy prezentów i zostaje zapisany w bazie.

SCENARIUSZ ALTERNATYWNY:

3a. W przypadku gdy podana w formularzu nazwa prezentu znajduje się już w systemie, wyświetlany jest komunikat o błędzie, prezent nie zostaje dodany do bazy i wracamy do punktu 1.

TYTUŁ: Usunąć prezent

AUTOR: Łukasz Gajewski

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie, posiada prezenty na jego liście prezentów

KONTEKST: Usunięcie prezentu z listy prezentów użytkownika

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Usunąć prezent*

GŁÓWNY SCENARIUSZ:

1. System wyświetla komunikat o usunięciu prezentu z listy.
2. Prezent zostaje usunięty z listy prezentów i bazy.

TYTUŁ: Wyświetl szczegółowy opis prezentu

AUTOR: Łukasz Gajewski

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie, posiada prezenty na jego liście prezentów

KONTEKST: Wyświetlenie szczegółów prezentu użytkownika

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika na wybrany prezent na swojej liście

GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi szczegółowe dane dotyczące prezentu.

TYTUŁ: Edytuj prezent

AUTOR: Patrycja Wrona

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie, posiada prezenty na jego liście prezentów

KONTEKST: Edytowanie szczegółów prezentu

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Edytuj prezent*

GŁÓWNY SCENARIUSZ:

1. System wyświetla formularz edycji prezentu.
2. Użytkownik wprowadza nowe dane dotyczące prezentu.
3. Po potwierdzeniu poprawności wprowadzonych danych, szczegóły prezentu zostają zmienione a dane zapisane w bazie.

SCENARIUSZ ALTERNATYWNY:

- 3a. W przypadku gdy podana w formularzu nazwa prezentu znajduje się już w systemie, wyświetlany jest komunikat o błędzie, prezent nie zostaje zedytowany do bazy i wracamy do punktu 1.

TYTUŁ: Wyświetl swoją listę prezentów

AUTOR: Patrycja Wrona

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany i zalogowany w systemie

KONTEKST: Wyświetlenie listy prezentów użytkownika

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik loguje się do systemu

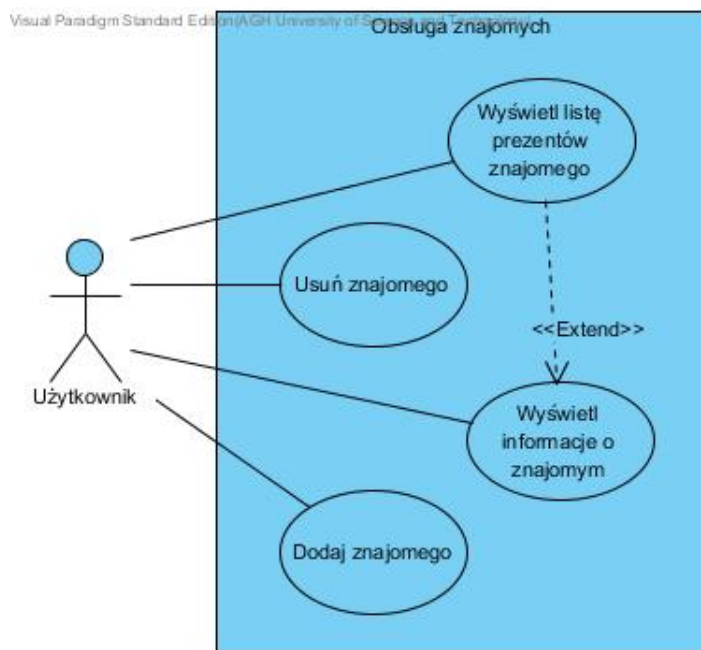
GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi listę prezentów przez niego dodanych.

SCENARIUSZ ALTERNATYWNY:

- 1a. W przypadku gdy lista prezentów jest pusta, wyświetlany jest stosowny komunikat.

1.1.4 Obsługa znajomych



TYTUŁ: Wyświetl listę prezentów znajomego

AUTOR: Grzegorz Siatka

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie i ma innych użytkowników dodanych do znajomych

KONTEKST: Wyświetlenie listy prezentów znajomego użytkownika

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik wybiera znajomego z listy znajomych i klika przycisk *Lista prezentów*

GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi listę prezentów znajomego.

SCENARIUSZ ALTERNATYWNY:

1a. W przypadku gdy lista prezentów jest pusta, wyświetlany jest stosowny komunikat.

TYTUŁ:Dodaj znajomego

AUTOR: Grzegorz Siatka

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie

KONTEKST: Dodanie użytkownika do listy znajomych

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik klika przycisk *Dodaj znajomego*

GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi formularz dodawania znajomego.
2. Użytkownik wprowadza nazwę użytkownika, którego chce dodać do znajomych.
3. Użytkownik o podanej nazwie po zaakceptowaniu zapytania zostaje dodany do listy znajomych użytkownika.

SCENARIUSZ ALTERNATYWNY:

- 2a. W przypadku gdy podany w formularzu użytkownik nie znajduje się w systemie użytkownik nie zostaje dodany do grona znajomych.

TYTUŁ:Usuń znajomego

AUTOR: Grzegorz Siatka

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie i ma innych użytkowników dodanych do znajomych

KONTEKST: Usunięcie użytkownika z listy znajomych

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik wybiera znajomego z listy znajomych i klika przycisk *Usuń znajomego*

GŁÓWNY SCENARIUSZ:

1. Użytkownik zostaje usunięty z listy znajomych.

TYTUŁ: Wyświetl informacje o znajomym

AUTOR: Patrycja Wrona

TYP PRZYPADKU: systemowy

AKTOR GŁÓWNY: Użytkownik

WARUNEK POCZĄTKOWY: Użytkownik jest zarejestrowany w systemie i ma innych użytkowników dodanych do znajomych

KONTEKST: Wyświetlenie informacji o znajomym

MINIMALNA GWARANCJA: Brak zmian w systemie w przypadku błędu

ZDARZENIE INICJUJĄCE: Użytkownik wybiera znajomego z listy znajomych

GŁÓWNY SCENARIUSZ:

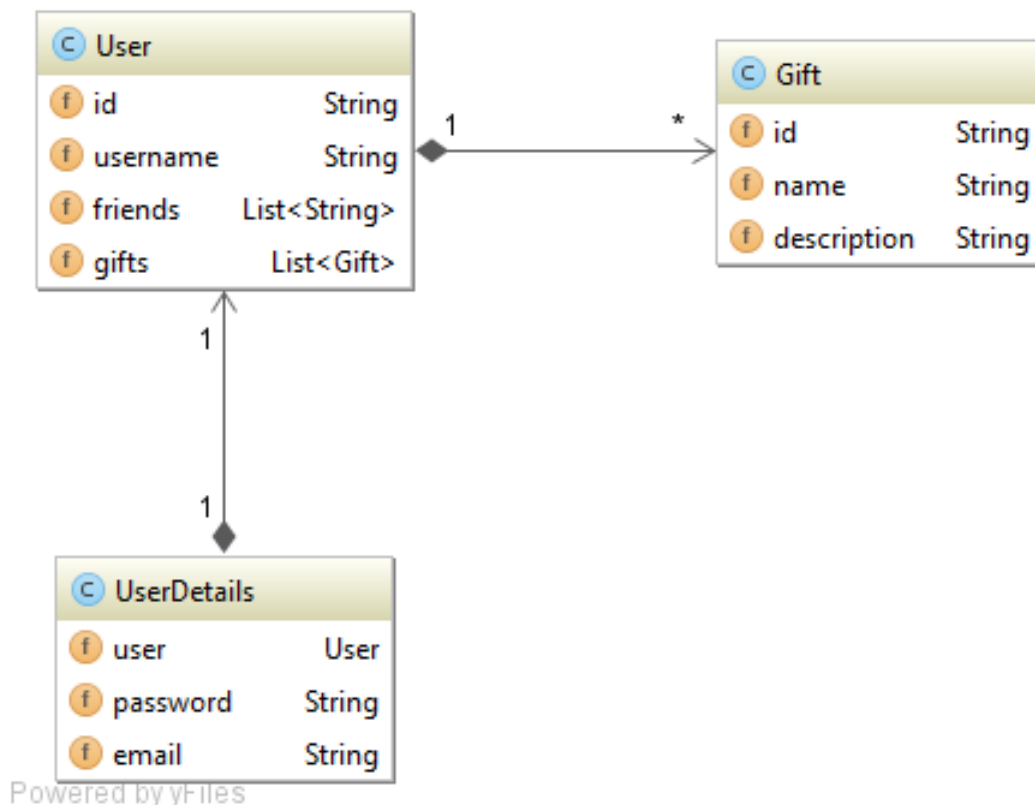
1. System wyświetla użytkownikowi informacje dotyczące wybranego znajomego.

2 Interfejsy

2.1 Moduł reprezentacji danych

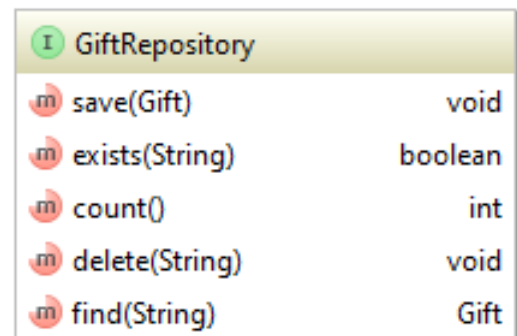
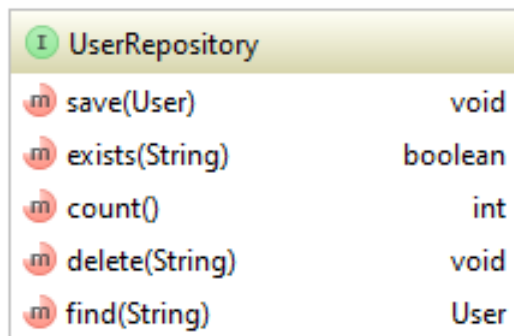
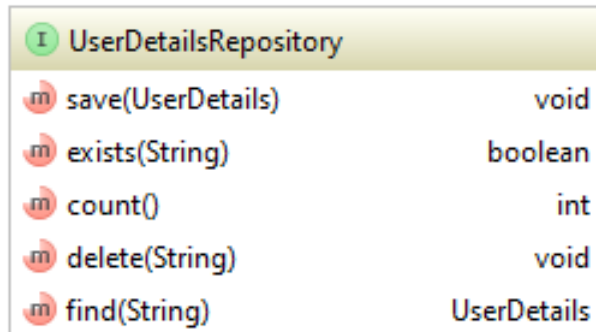
Odpowiedzialny za reprezentację świata rzeczywistego w środowisku wirtualnym. Określa strukturę i zależności między poszczególnymi elementami:

- User - użytkownik w systemie o określonym identyfikatorze, posiadający swój 'username' oraz mający wskazanie na listę swoich znajomych, wraz z prezentami przypisanymi do siebie.
- UserDetails - szczegóły dotyczące użytkownika, niewidoczne dla innych modułów z powodów danych wrażliwych.
- Gift - encja odwzorowująca prezent ze świata rzeczywistego. Posiada identyfikator, nazwę oraz opis.



2.2 Moduł persystencji

Służy do utrwalania oraz pobierania potrzebnych elementów z bazy danych. Wspiera podstawowe operacje wykonane na bazie danych (CRUD).



Interfejsy:

void save(UserDetails) - zapisuje w bazie danych dane użytkownika dotyczące logowania

boolean exists(String) - określa czy w bazie danych istnieje użytkownik o podanym identyfikatorze

int count() - zwraca wielkość kolekcji

void delete(String) - usuwa użytkownika o danym identyfikatorze z bazy danych

UserDetails find(String) - wyszukuje użytkownika na podstawie podanego identyfikatora

void save(User) - zapisuje w bazie danych dane użytkownika dotyczące prezentów i

znajomych

boolean exists(String) - określa czy w bazie danych istnieje użytkownik o podanym identyfikatorze

int count() - zwraca wielkość kolekcji

void delete(String) - usuwa użytkownika o danym identyfikatorze z bazy danych

User find(String) - wyszukuje użytkownika na podstawie podanego identyfikatora

void save(Gift) - zapisuje prezent przekazany jako parametr w bazie danych

boolean exists(String) -

int count() - zwraca wielkość kolekcji prezentów

void delete(String) - usuwa prezent o danym identyfikatorze z bazy danych

Gift find(String) - wyszukuje prezent na podstawie podanego identyfikatora

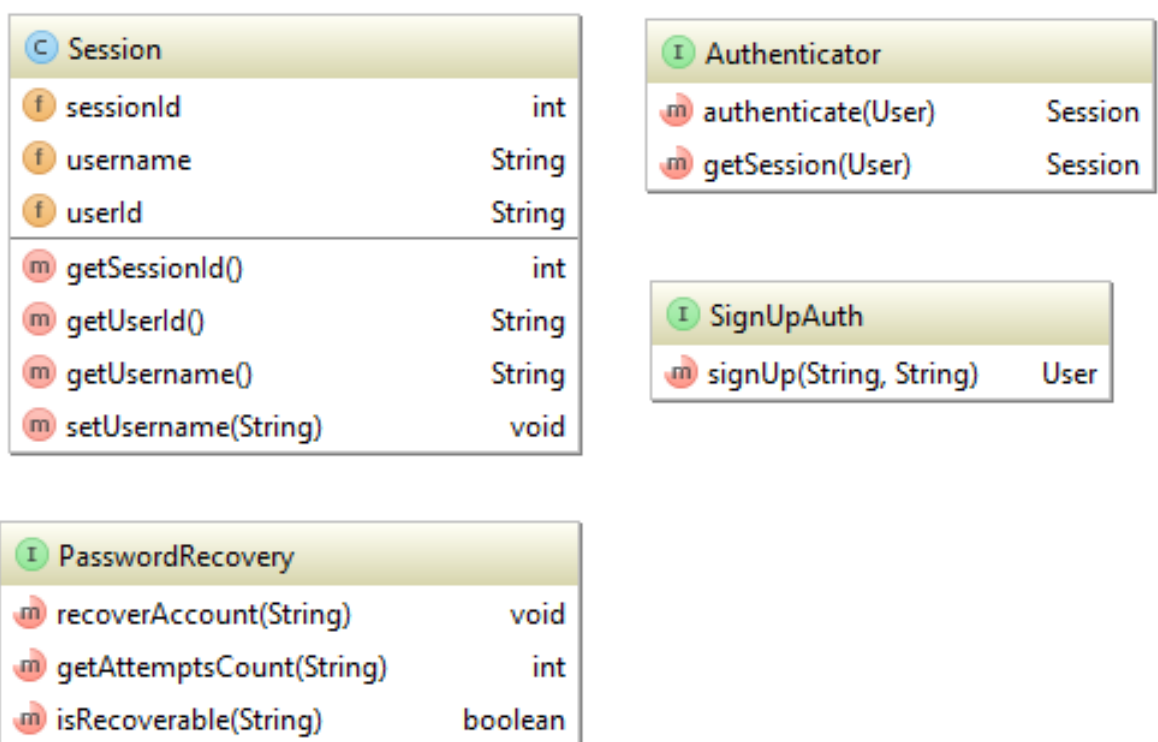
Moduł persystencji pozwala na:

- Zapisywanie w bazie danych
- Wyszukiwanie z bazy
- Pobieranie rozmiaru kolekcji o określonym typie dokumentów
- Znajdowanie encji w bazie danych na podstawie podanego identyfikatora
- Określenie, czy w bazie danych istnieje obiekt z podanym identyfikatorem

2.3 Moduł autoryzacji użytkownika

Dzięki niemu użytkownik będzie w stanie założyć konto uprawniające go do korzystania z aplikacji. Umożliwia identyfikację użytkownika za pomocą wybranego loginu oraz hasła, a w przypadku utraty lub zapomnienia hasła będzie możliwość odzyskania go przy pomocy adresu mailowego, dla którego utworzono konto.

Moduł autoryzacji przechowuje dane tymczasowe w postaci sesji, w celu określenia tożsamości użytkownika i dalszego zarządzania danymi bezpośrednio związanymi z kontem.



Interfejsy:

int getSessionId() - zwraca Id obecnej sesji w postaci liczby całkowitej

String getUserId() - zwraca Id użytkownika w postaci String'a

String getUsername() - zwraca nazwę użytkownika

void setUsername(String) - ustawia nazwę użytkownika na tą podaną jako parametr

void recoverAccount(String) - wysyła maila z linkiem do odzyskiwania konta na adres przekazany jako parametr

int getAttemptsCount(String) - zwraca liczbę prób odzyskiwania hasła dla danego użytkownika

boolean isRecoverable(String) - określa czy hasło jest możliwe do odzyskania, zwraca true lub false

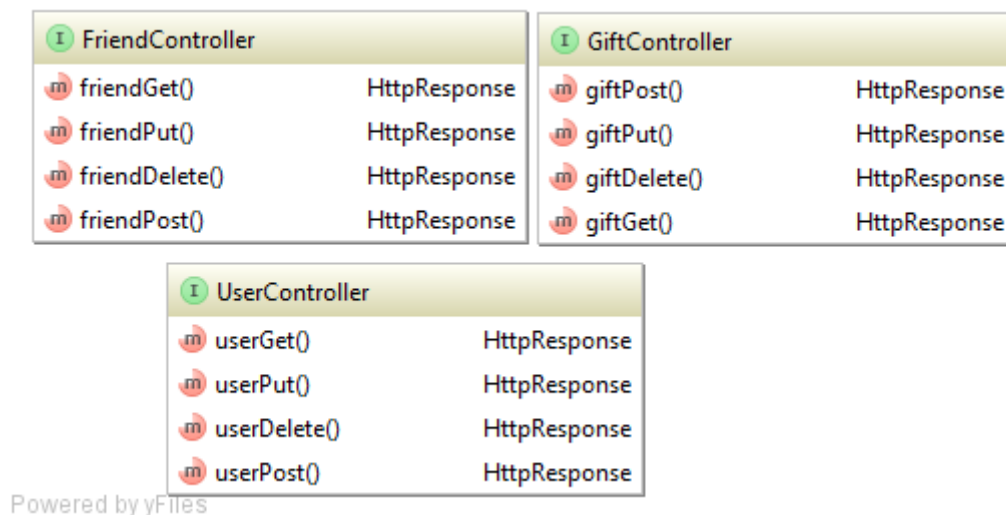
Session authenticate(User) - przeprowadza autentykację użytkownika przekazanego jako parametr, zwraca obiekt sesji

Session getSession(User) - zwraca obiekt sesji dla danego użytkownika

User signUp(String, String) - przeprowadza autentykację użytkownika przy logowaniu na podstawie przekazanej nazwy i hasła

2.4 Moduł webowy, komunikacji z użytkownikiem

Pozwala na zarządzanie prezentami oraz kontem użytkownika i jego znajomymi poprzez odpowiednie metody, które działają w charakterze 'servlet' i pozwalają na odwzorowanie adresów URL w celu otrzymania skonfigurowanej odpowiedzi przeznaczonej dla określonego użytkownika.



Interfejsy (wszystkie metody zwracają odpowiedź HTTP zależną od powodzenia akcji):

HttpServletResponse friendGet() - zwraca instancję User'a reprezentującą znajomego o ID przekazany jako parametr metody

HttpServletResponse friendPut() - aktualizuje dane znajomego w bazie danych na te przekazane jako parametr metody

HttpServletResponse friendDelete() - usuwa użytkownika przekazanego jako parametr z listy znajomych

HttpServletResponse friendPost() - dodaje nowego znajomego przekazanego jako parametr

HttpServletResponse userGet() - zwraca instancję User'a o ID przekazany jako parametr metody

HttpServletResponse userPut() - aktualizuje dane użytkownika w bazie danych na te przekazane jako parametr metody

HttpServletResponse userDelete() - usuwa użytkownika przekazanego jako parametr z bazy danych

HttpResponse userPost() - dodaje nowego użytkownika przekazanego jako parametr

HttpResponse giftGet() - zwraca instancję Gift'u reprezentującą prezent o ID przekazany jako parametr metody









HttpResponse giftPut() - aktualizuje dane prezentu w bazie danych na te przekazane jako parametr metody





HttpResponse giftDelete() - usuwa prezent przekazany jako parametr z bazy danych

HttpResponse giftPost() - dodaje nowy prezent przekazany jako parametr

2.5 Moduł mobilny, komunikacji z użytkownikiem

Udostępnia metody do zarządzania prezentami oraz kontem użytkownika po stronie platformy natywnej.

I FriendController	I UserController
 getFriend() User	 getUser() User
 updateFriend(User) void	 updateUser(User) void
 deleteFriend(User) void	 deleteUser(User) void
 exists(User) boolean	 exists(User) boolean

I GiftController
 getGift() Gift
 updateGift(Gift) void
 deleteGift(Gift) void
 exists(Gift) boolean

Powered by yFiles

Interfejsy:

User getFriend(String) - zwraca instancję User'a reprezentującą znajomego o ID przekazany jako parametr metody

void updateFriend(User) - aktualizuje dane znajomego w bazie danych na te przekazane jako parametr metody

void deleteFriend(User) - usuwa użytkownika przekazanego jako parametr z listy znajomych

boolean exists(User) - sprawdza czy istnieje znajomy przekazany jako parametr, zwraca true lub false

User getUser(String) - zwraca instancję User'a o ID przekazanym jako parametr metody

void updateUser(User) - aktualizuje dane użytkownika w bazie danych na te przekazane jako parametr metody

void deleteUser(User) - usuwa użytkownika przekazanego jako parametr z bazy danych

boolean exists(User) - sprawdza czy istnieje w bazie danych użytkownik przekazany jako parametr, zwraca true lub false

Gift getGift(String) - zwraca instancję Gift'u o ID przekazanym jako parametr metody

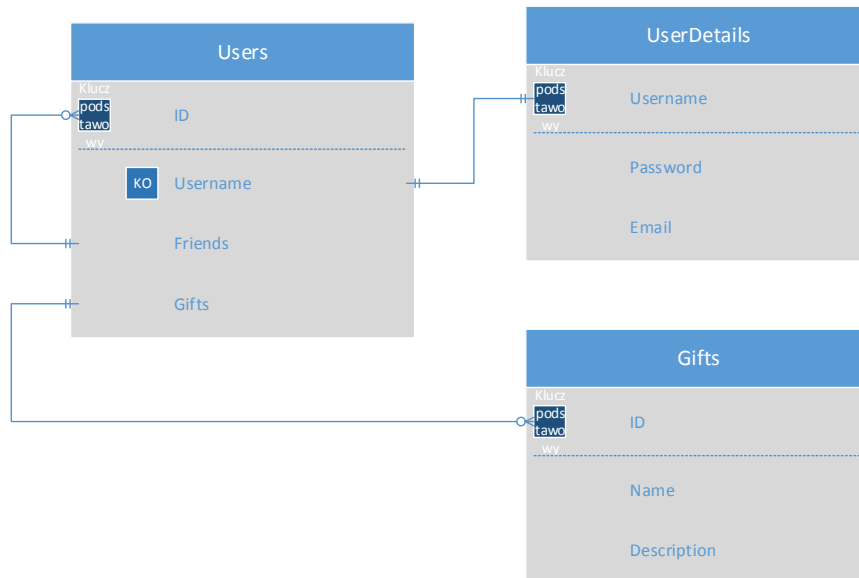
void updateGift(Gift) - aktualizuje dane prezentu w bazie danych na te przekazane jako parametr metody

void deleteGift(Gift) - usuwa prezent przekazany jako parametr z bazy danych

boolean exists(Gift) - sprawdza czy istnieje w bazie danych prezent przekazany jako parametr, zwraca true lub false

3 Baza danych

3.1 Diagram encji



3.2 Użytkownik

- Użytkownik jest identyfikowany w systemie za pomocą nazwy.
- Każdy użytkownik ma unikalną nazwę użytkownika przypisaną do unikalnego adresu e-mail.
- Hasło użytkownika jest zapisywane w bazie danych w postaci zaszyfrowanej, aby zapobiec wyciekowi wrażliwych danych.
- Użytkownik jest powiązany z dokładnie jedną listą prezentów, na której może znajdować się wiele prezentów.
- Użytkownik może mieć wielu innych użytkowników na liście swoich znajomych.

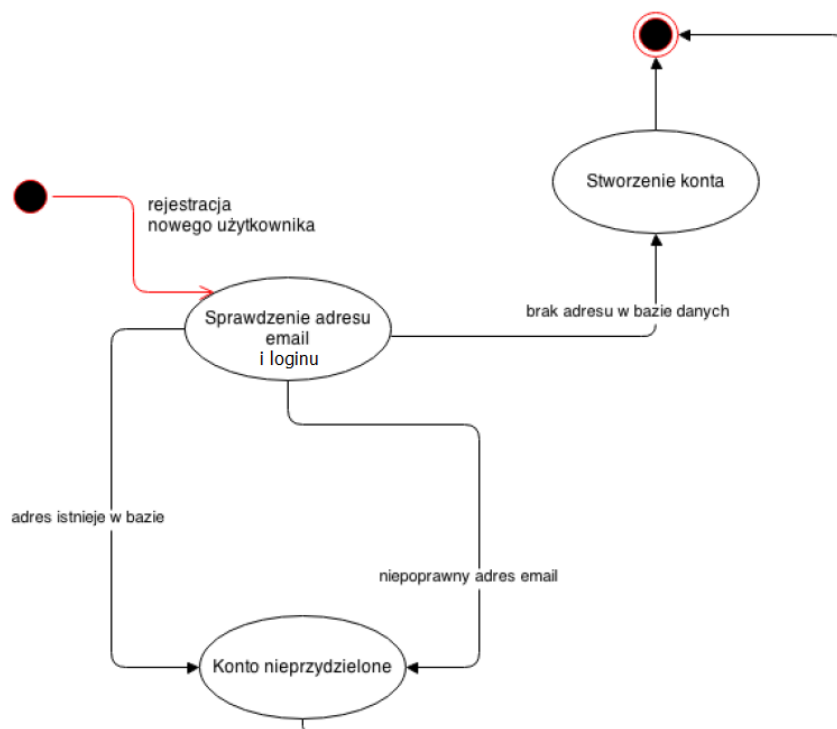
3.3 Prezent

- Prezent jest identyfikowany w systemie za pomocą ID.

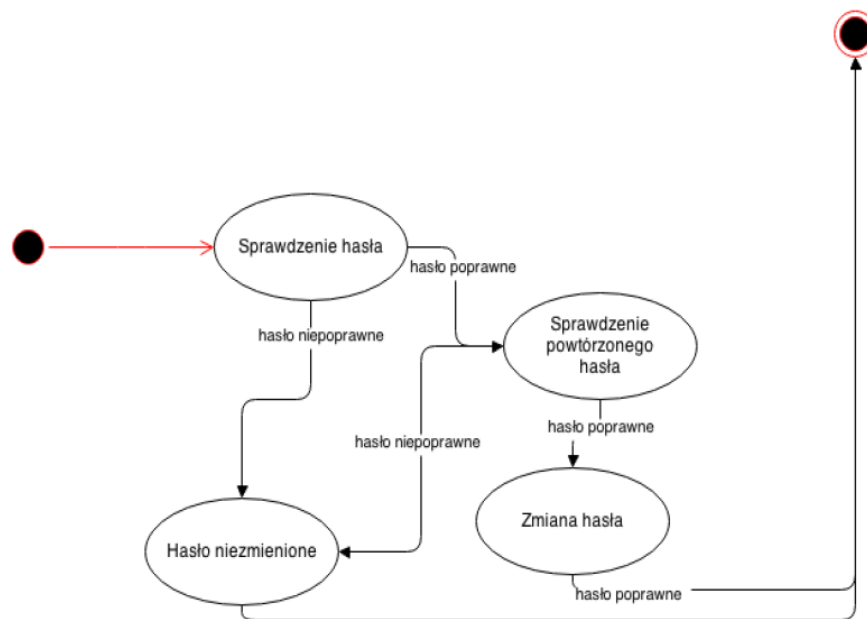
- Prezent nie musi mieć unikalnej nazwy.
- Prezent składa się z nazwy i opisu.
- Na liście prezentów może znajdować się wiele prezentów.

4 Diagramy maszyn stanowych

4.1 Maszyna stanowa dla dodawania użytkownika



4.2 Maszyna stanowa dla zmiany hasła



5 Adresowanie

Adresowanie podstron jest istotnym elementem aplikacji, który pozwala na uporządkowanie zarządzania dostępem do stron, a także na kategoryzację dostępnych metod. Zastosowanie takiego samego prefiksu dla powiązanych ze sobą usług, wpływa na skuteczne zwiększenie bezpieczeństwa (reguły w Spring Security).

5.1 Użytkownicy

- Wszystkie adresy funkcji serwisowych dotyczących użytkowników zaczynają się prefiksem `/users`
- W zależności od rodzaju operacji (dodawanie, usuwanie, aktualizowanie danych użytkownika itd.) będą używane requesty odpowiedniego typu (POST, GET, PUT, DELETE).

5.2 Prezenty

- Wszystkie adresy funkcji serwisowych dotyczących prezentów zaczynają się prefiksem '/gifts'
- Podobnie jak w przypadku użytkowników w zależności od rodzaju operacji (dodawanie, usuwanie, aktualizowanie prezentów itd.) będą używane requesty odpowiedniego typu (POST, GET, PUT, DELETE).

5.3 Znajomi

- Wszystkie adresy funkcji serwisowych dotyczących użytkowników zaczynają się prefiksem '/friends'
- Tutaj również została zastosowana konwencja taka jak w pozostałych przypadkach.

6 Moduł mailowy

6.1 Protokół SMTP

Protokół SMTP służy w sieci Internet do wysyłania poczty. Serwer obsługujący ten protokół, w terminologii programów - klientów poczty elektronicznej nazywany jest najczęściej serwerem poczty wychodzącej.