



AKADEMIA GÓRNICZO-HUTNICZA  
IM. STANISŁAWA STASZICA W KRAKOWIE

INŻYNIERIA OPROGRAMOWANIA

DOKUMENT SPECYFIKACYJNY

---

## Wish List

---

*Autorzy:*

Łukasz GAJEWSKI

Grzegorz SIATKA

Patrycja WRONA

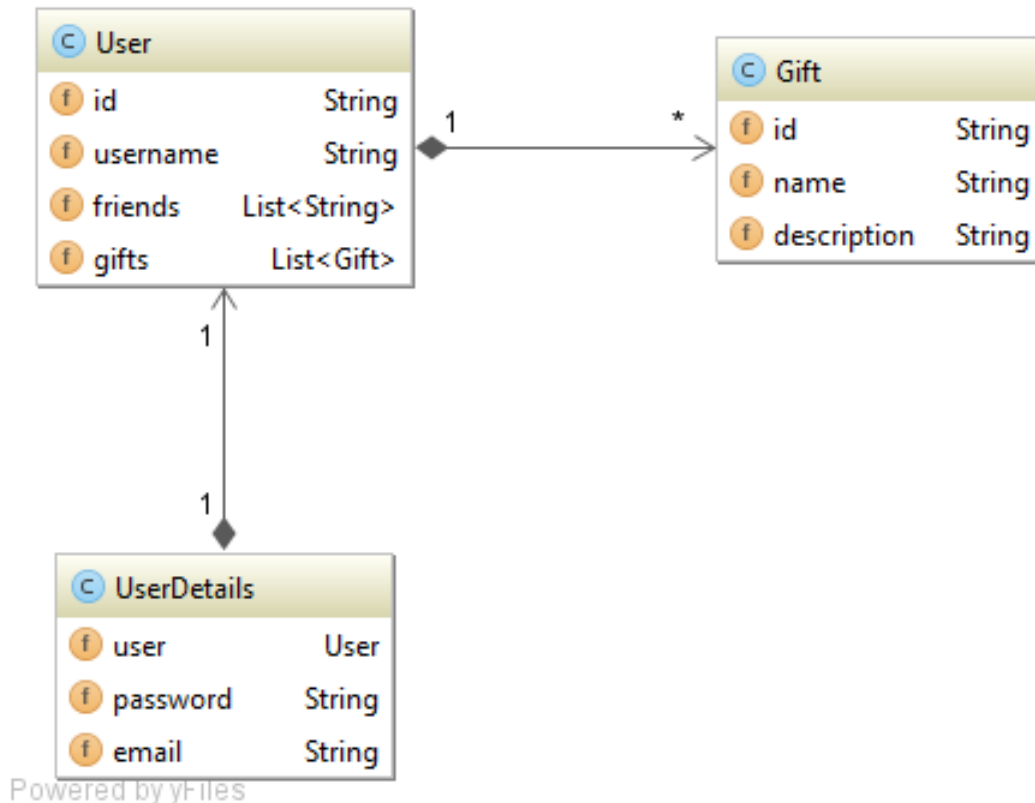
2016

# Spis treści

<b>1</b>	<b>Interfejsy</b>	<b>2</b>
1.1	Moduł reprezentacji danych . . . . .	2
1.2	Moduł persystencji . . . . .	3
1.3	Moduł autoryzacji użytkownika . . . . .	5
1.4	Moduł webowy, komunikacji z użytkownikiem . . . . .	7
1.5	Moduł mobilny, komunikacji z użytkownikiem . . . . .	8
<b>2</b>	<b>Specyfikacje</b>	<b>10</b>
2.1	Baza danych . . . . .	10
2.1.1	Diagram encji . . . . .	10
2.1.2	Użytkownik . . . . .	10
2.1.3	Prezent . . . . .	11
2.2	Diagramy maszyny stanowej . . . . .	11
2.2.1	Dodanie użytkownika . . . . .	11
2.2.2	Zmiana hasła . . . . .	12
2.3	Adresowanie . . . . .	12
2.3.1	Użytkownicy . . . . .	12
2.3.2	Prezenty . . . . .	12
2.3.3	Znajomi . . . . .	13
2.4	Moduł mailowy . . . . .	13
2.4.1	Protokół SMTP . . . . .	13
<b>3</b>	<b>Przypadki użycia</b>	<b>14</b>
3.1	Diagramy przypadków użycia . . . . .	14
3.1.1	Podsystemy . . . . .	14
3.1.2	Autoryzacja . . . . .	14
3.1.3	Obsługa prezentów . . . . .	18
3.1.4	Obsługa znajomych . . . . .	21

# 1 Interfejsy

## 1.1 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.

## 1.2 Moduł persystencji

I UserDetailsRepository		
m	save(UserDetails)	void
m	exists(String)	boolean
m	count()	int
m	delete(String)	void
m	find(String)	UserDetails

I UserRepository		
m	save(User)	void
m	exists(String)	boolean
m	count()	int
m	delete(String)	void
m	find(String)	User

I GiftRepository		
m	save(Gift)	void
m	exists(String)	boolean
m	count()	int
m	delete(String)	void
m	find(String)	Gift

Powered by yFiles

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

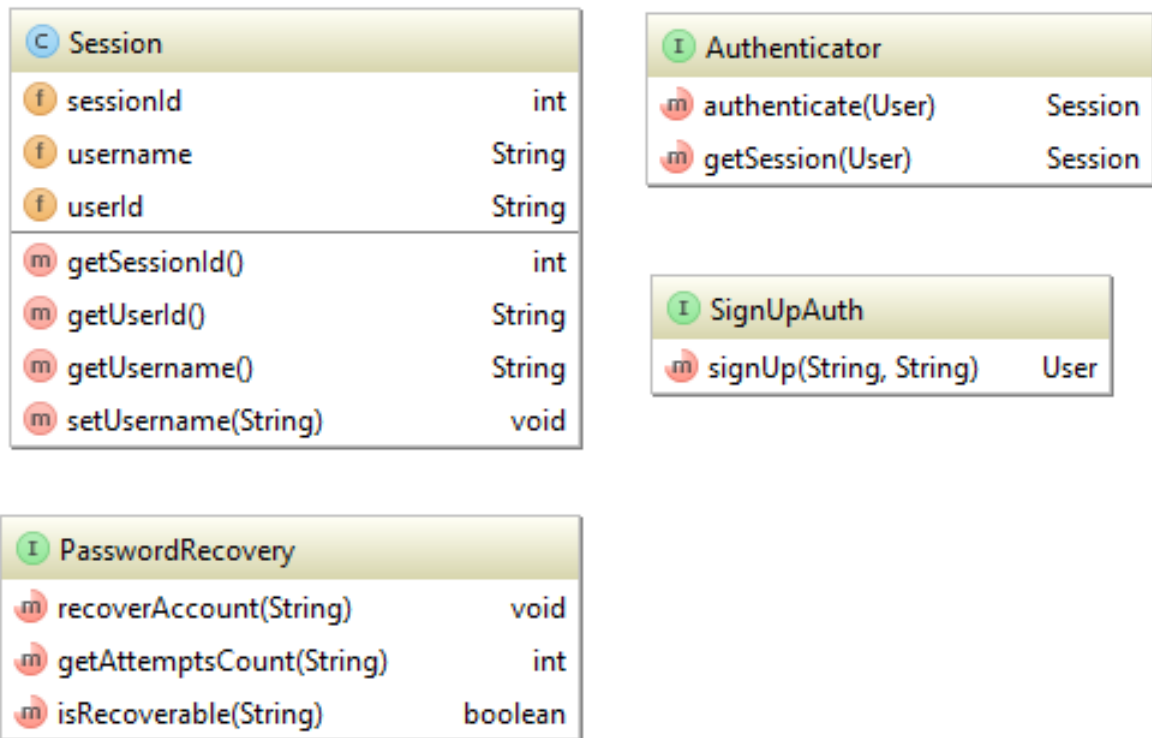
**void delete(String)** - usuwa prezent o danym identyfikatorze z bazy danych

**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

## 1.3 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

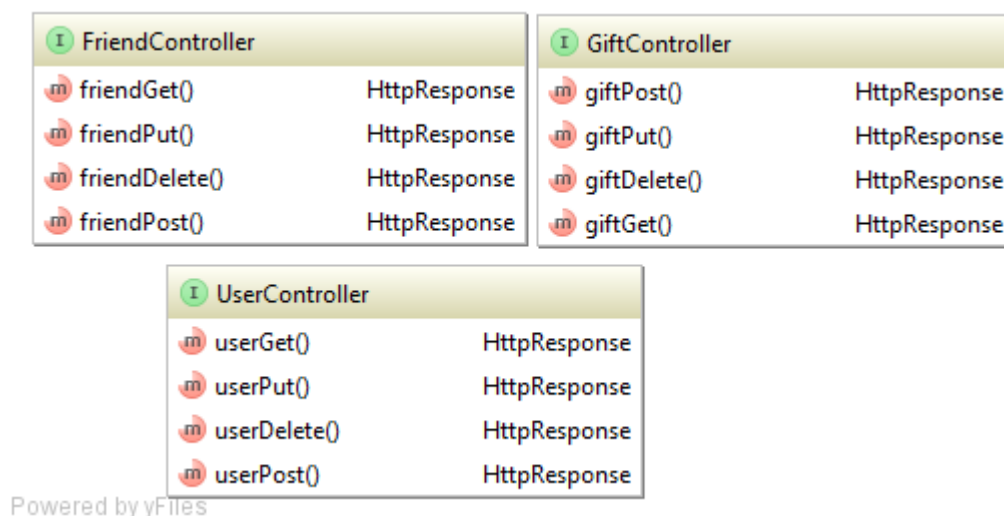
**boolean isRecoverable(String)** - określa czy hasło jest możliwe do odzyskania, zwraca true lub false

**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

## 1.4 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 przekazanym 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 przekazanym 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

## 1.5 Moduł mobilny, komunikacji z użytkownikiem

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

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

Powered by yFiles

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

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

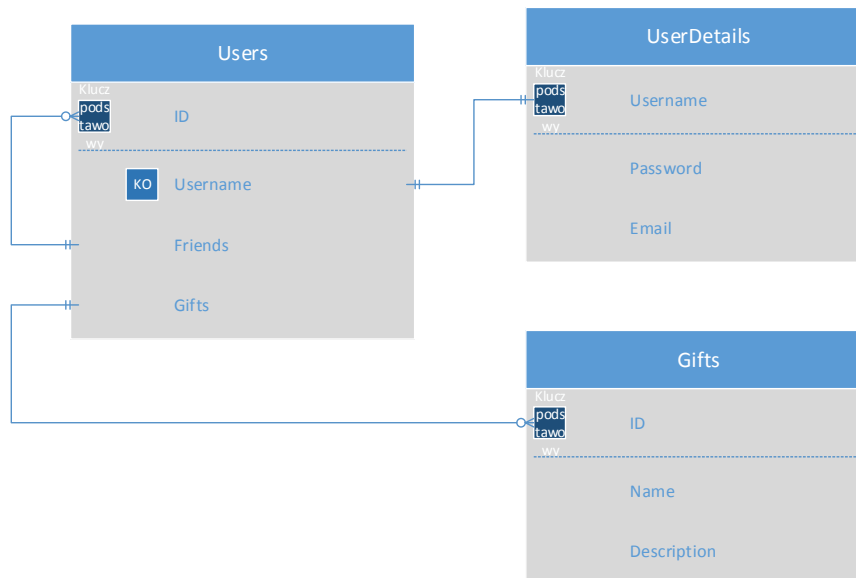
**void deleteGift(Gift)** - usuwa prezent przekazany jako parametr z bazy danych

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

## 2 Specyfikacje

### 2.1 Baza danych

#### 2.1.1 Diagram encji



#### 2.1.2 Użytkownik

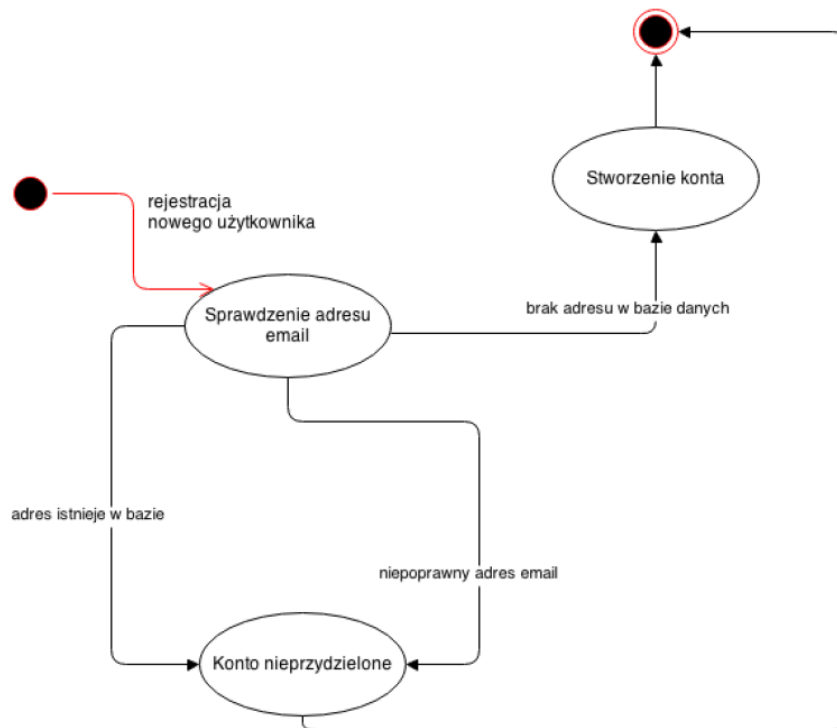
- Użytkownik jest identyfikowany w systemie za pomocą nazwy.
- Każdy użytkownik ma unikalną nazwę użytkownika przypisaną do unikalnego adresu e-mail.
- Hasło użytkownika jest zapisywane w bazie danych w postaci zaszyfrowanej, aby zapobiec wyciekowi wrażliwych danych.
- Użytkownik jest powiązany z dokładnie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

### 2.1.3 Prezent

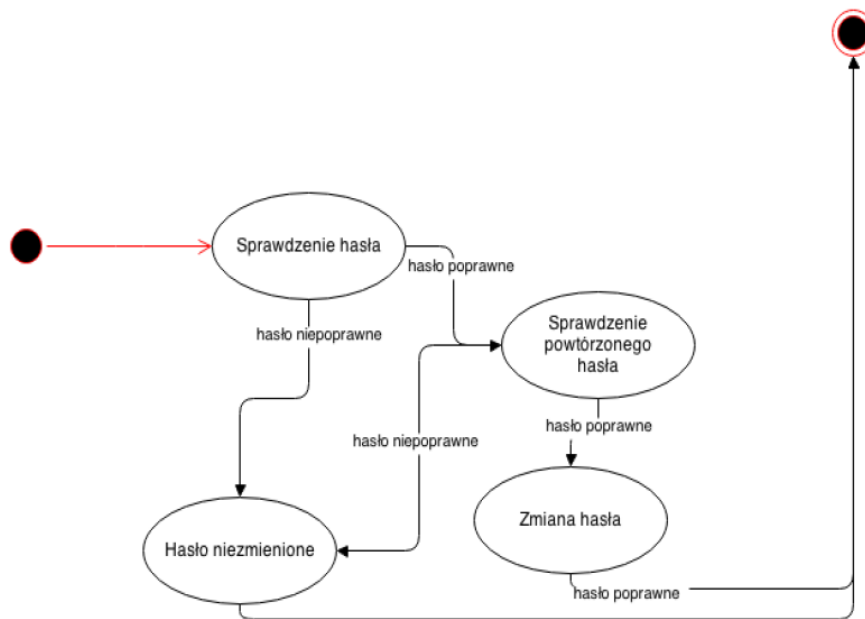
- Prezent jest identyfikowany w systemie za pomocą ID.
- Prezent nie musi mieć unikalnej.
- Prezent składa się z nazwy i opisu prezentu.
- Na liście prezentów może znajdować się wiele prezentów.

## 2.2 Diagramy maszyny stanowej

### 2.2.1 Dodanie użytkownika



### 2.2.2 Zmiana hasła



## 2.3 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).

### 2.3.1 Użytkownicy

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

### 2.3.2 Prezenty

- Wszystkie adresy funkcji serwisowych dotyczących prezentów zaczynają się prefiksem `/gift`

- Podobnie jak w przypadku użytkowników w zależności od rodzaju operacji (dodawanie, usuwanie, aktualizowanie prezentów itd.) będą używane requesty odpowiedniego typu (POST, GET, PUT, DELETE).

### **2.3.3 Znajomi**

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

## **2.4 Moduł mailowy**

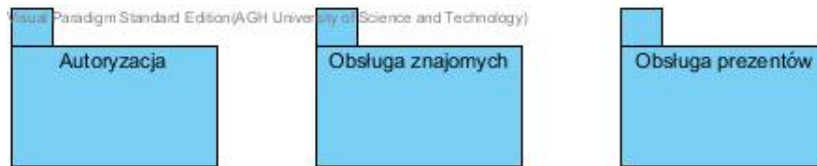
### **2.4.1 Protokół SMTP**

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.

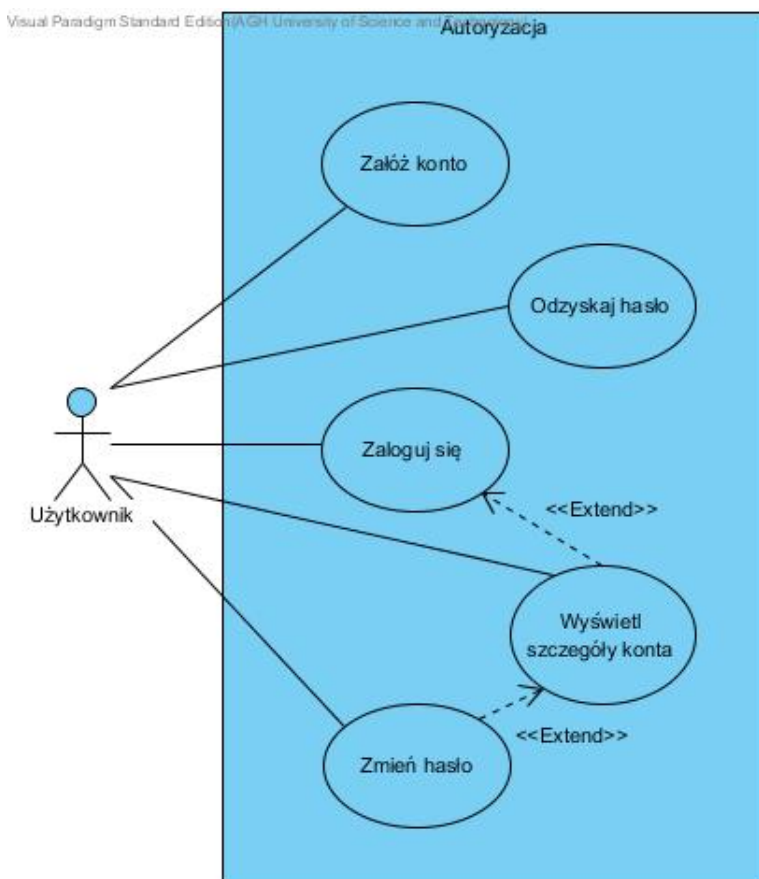
## 3 Przypadki użycia

### 3.1 Diagramy przypadków użycia

#### 3.1.1 Podsystemy



#### 3.1.2 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

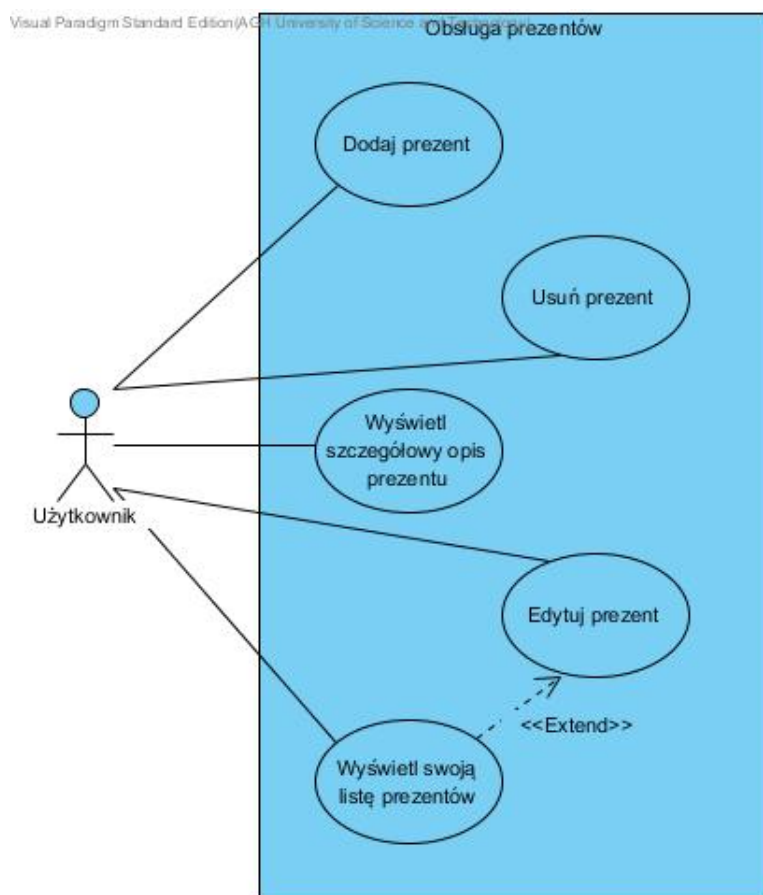
MINIMALNA GWARANCJA: Baza danych pozostanie spójna, brak zmian w przypadku błędu

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.

### 3.1.3 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

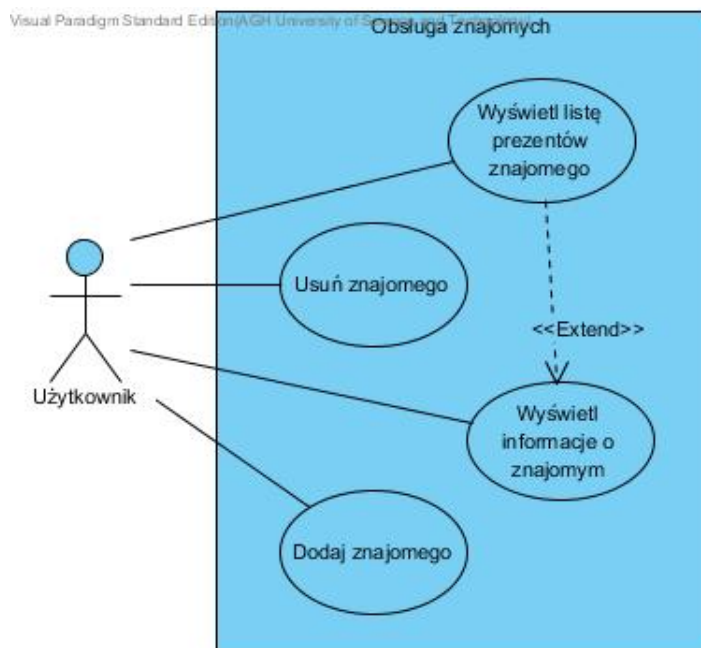
GŁÓWNY SCENARIUSZ:

1. System wyświetla użytkownikowi listę prezentów przez niego dodanych.

SCENARIUSZ ALTERNATYWNY:

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

### 3.1.4 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 zostaje dodany do listy znajomych użytkownika.

**SCENARIUSZ ALTERNATYWNY:**

2a. W przypadku gdy podany w formularzu użytkownik nie znajduje się w systemie, wyświetlany jest komunikat o błędzie, 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. System wyświetla użytkownikowi komunikat o usunięciu znajomego.
2. 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.