POLITECHNIKA KRAKOWSKA
IM. TADEUSZA KOŚCIUSZKI
WYDZIAŁ FIZYKI MATEMATYKI I INFORMATYKI
KIERUNEK INFORMATYKA

BARTŁOMIEJ KWIATEK

SYSTEM CENTRALNEGO UWIERZYTELNIANIA, AUTORYZACJI I ZARZĄDZANIA UŻYTKOWNIKAMI SERWISU INTERNETOWEGO

PRACA INŻYNIERSKA
STUDIA STACJONARNE

Promotor: dr Jaromir Smagłowski

Kraków 2012

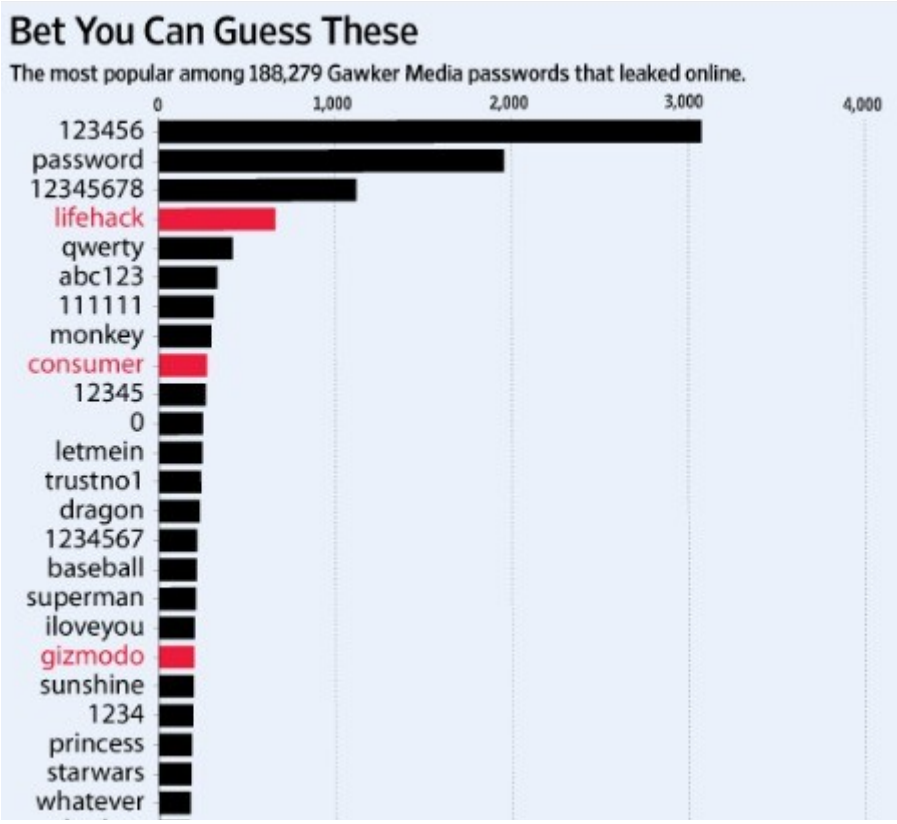
Spis treści

Wstęp....4
1. Definicja problemu....4
2. Cel i zakres pracy....5
Rozdział 1. Zasady działania systemu....6
1. Definicje....6
2. Konto użytkownika....7
3. Uwierzytelnianie i autoryzacja....7
4. Architektura OpenID i standard oAuth....8
5. Hasła do konta użytkownika....9
6. Protokół LDAP....10
7. Połączenia szyfrowane....10
Rozdział 2. Projekt aplikacji....12
1. Założenia....12
2. Architektura systemu....13
3. Opis technologii....14
4. Opis działania aplikacji....15
5. Wygląd aplikacji....20
Zakończenie....28
Indeks ilustracji....29
Bibliografia....30

Wstęp

1. Definicja problemu

Większość użytkowników internetu posiada wiele kont w różnych serwisach. Zmuszeni są przez to do pamiętania wielu loginów (nazw logowania) oraz haseł powiązanych z tymi loginami. Dla własnej wygody bardzo często używamy takiego samego hasła do wielu usług. Niestety dość często stosowane przez nas hasła są bardzo proste (zob. [1] i [2]), przez co bezpieczeństwo zabezpieczonego konta jest małe. Więcej informacji o złożoności haseł i sposobach tworzenia dobrego hasła można znaleźć w internecie (zob. [3], [4]).



Ilustracja 1: Lista najpopularniejszych haseł [2]

Jest kilka rozwiązań tego problemu:

1. Użytkownik może korzystać z serwisu centralnego logowania, np. używając konta w usłudze OpenID.
2. Możliwe jest również logowanie za pomocą już uwierzytelnionego konta, np. logowanie za pomocą oAuth z portalu Facebook® .
3. Istnieją również serwisy internetowe zapamiętujące wszystkie hasła użytkownika. Aby móc się zalogować, wystarczy podać jedno hasło główne. Sposób ten wymaga od użytkownika zaufania danemu podmiotowi oferującemu usługę.

4. Podobne podejście wykorzystują przeglądarki internetowe, z tą różnicą, że dane przechowywane są zazwyczaj tylko na komputerze osobistym i nie muszą być chronione hasłem głównym. Jest to dość dobry sposób, ale trzeba mieć pewność, że przechowywane dane nie są wrażliwe (np. dostęp do banku) lub hasło główne jest odpowiednio mocne.
5. Najmniej polecanym rozwiązaniem i prawdopodobnie najczęściej stosowanym jest używanie takiego samego (lub bardzo podobnego) hasła do wszystkich usług.

Dla firm, które oferują swoim użytkownikom wiele usług wymagających uwierzytelniania, jednym z lepszych rozwiązań jest wdrożenie systemu typu AAA. Umożliwi to zarządzanie i kontrolę wszystkich użytkowników (klientów) oraz ułatwi i przyspieszy działania tych użytkowników w tych oferowanych usługach.

2. Cel i zakres pracy

W mojej pracy prezentuję aplikację do centralnego zarządzania użytkownikami i obsługi ich transakcji oraz zagadnienia związane z tym tematem.

Rozdział pierwszy przedstawia ogólną problematykę mojej pracy. Opisuję w nim dane, jakie są przechowywane na serwerze i przetwarzane przez aplikację. Omawiam zasady działania procedur uwierzytelniania i autoryzacji użytkownika. Przestawiam również istniejące rozwiązania wykorzystujące te procedury. Na końcu przybliżone są kwestie bezpieczeństwa – zarówno przechowywania danych, jak i dostępu do nich oraz połączenia z aplikacją.

Następny rozdział opisuje mój pomysł działania aplikacji. Pojawiają się założenia i opis architektury oraz przede wszystkim zaprezentowane jest działanie utworzonej aplikacji. Przedstawiam również technologię, za pomocą której aplikacja została napisana.

W zakończeniu pokazane są wady i zalety stosowania zaprezentowanej aplikacji oraz podsumowanie.

Rozdział 1. Zasady działania systemu

1. Definicje

1.1. Aktorzy

- usługodawca, serwer AAA – aplikacja internetowa dostarczająca usługi AAA (ang. *Authentication, Authorization i Accounting*), czyli uwierzytelniania, autoryzacji i historii konta;
- użytkownik – osoba fizyczna posiadająca konto u usługodawcy;
- klient, aplikacja kliencka – aplikacja internetowa oferująca własne określone usługi; może korzystać z zasobów usługodawcy w imieniu użytkownika po uzyskaniu autoryzacji;
- system (serwis) centralnego zarządzania kontem użytkownika (logowania) – aplikacja rozproszona składająca się z aplikacji klienta i aplikacji serwera;

1.2. Pojęcia

- uwierzytelnianie – odpowiada na pytanie: kim jest ten użytkownik (zob. poniżej punkt 3.1);
- autoryzacja – kontrola dostępu; definiuje co może (lub nie może) zrobić dany użytkownik (zobacz poniżej punkt 3.2);
- historia konta – odpowiada na pytanie: co ta osoba robi; obsługuje historię oraz rozliczenia konta użytkownika (zob. poniżej punkt 2);
- logowanie – uwierzytelnianie użytkownika (nie należy mylić z logowaniem zdarzeń w historii konta użytkownika);

1.3. Akronimy

- LDAP – ang. *Lightweight Directory Access Protocol* – dosł. Lekki Protokół Usług Katalogowych (więcej na stronie 10);
- PHP – ang. *PHP: Hypertext Preprocessor*; obiektowy, skryptowy język programowania (więcej na stronie 14);
- MVC – architektoniczny wzorzec projektowy do organizowania struktury aplikacji posiadających graficzne interfejsy użytkownika; zakłada podział aplikacji na trzy główne części: model (problem, logiki aplikacji), widok (wyświetlanie danych w ramach interfejsu użytkownika) i kontroler (odbiera dane użytkownika, jest łącznikiem między modelem i widokiem).

1.4. Prawa własności znaków firmowych stron trzecich

Wszelkie wymienione w tym dokumencie znaki towarowe i nazwy firm bądź produktów są własnością odpowiednich podmiotów. Dotyczy to w szczególności:

- Facebook®, Google™, Microsoft®, oAuth, National Geographic™, OpenID, PayPal™, Symantec™, Yahoo!®.

2. Konto użytkownika

Konto użytkownika w sposób jednoznaczny wiąże użytkownika z daną usługę informatyczną. Składa się z identyfikatora i hasła. Nowemu użytkownikowi przyznawane są domyślne uprawnienia. W systemie może również istnieć konto gościa, które posiada standardowy zestaw uprawnień dla użytkownika niezarejestrowanego. [5]

Zakładanie konta użytkownika powinno być proste i bezpieczne. Proste, czyli nie powinno się wymagać od użytkownika zbyt dużej ilości danych, bo długie formularze mogą skutecznie odstraszyć. Bezpieczne zarówno pod względem przesyłania danych (połączenia szyfrowane) jak i wymagające dodatkowego potwierdzenie np. przez e-maila, SMS-a lub telefonicznie, rzadziej w formie listu tradycyjnego.

Po zalogowaniu się na swoje konto, użytkownik ma dostęp do swoich danych – może je zmieniać lub całkowicie zamknąć konto. Powinna pojawić się też możliwość podglądu historii konta. Zarządza historią oraz rozliczenia konta użytkownika to przede wszystkim informacje o rozliczeniach finansowych, oraz lista uwierzytelnień i błędów autoryzacji. [6]

3. Uwierzytelnianie i autoryzacja

3.1. Uwierzytelnianie

Procedura uwierzytelniania odpowiada na pytanie, kim jest dany użytkownik (ang. *Who is this person?*). W procesie tym, podmiot tożsamości zostaje uwierzytelniony poprzez przedstawienie swojej cyfrowej tożsamości. Ta cyfrowa tożsamość to identyfikator i odpowiednie poświadczenia (więcej na stronie 9). [6] [7]

Procedura uwierzytelniania (często również autoryzacji) może czasem wymagać tzw. podwójnego uwierzytelniania. Użytkownik jest wtedy proszony o podanie dodatkowego kodu (poświadczenia) dostępu do swojego konta. Rozwiązania tego typu stosują głównie banki i inne instytucje finansowe. Zabezpieczenia tego typu można również spotkać na stronach popularnych serwisów internetowych (zob. [8] i [9]).

3.2. Rodzaje autoryzacji

Procedura autoryzacji definiuje, czy dany użytkownik jest uprawniony o wykonania określonego działania (ang. *What is this person authorized to do?*). Z reguły uprawnienia te są określane już przy uwierzytelnianiu. Autoryzacja może zależeć o wielu czynników, takich jak: czas (pora dnia) dostępu do zasobu, fizyczna lokalizacja użytkownika lub ograniczenie wielodostępu użytkownika. [6]

a) Autoryzacja dla własnego serwisu

Jeżeli podmiot posiadający serwis centralnego logowania posiada jakiś serwis wymagający autoryzacji, to może zaproponować użytkownikowi uwierzytelnianie i autoryzację za pomocą własnego serwisu centralnego logowania.

Przykładem zastosowania tego rodzaju autoryzacji jest dostęp do serwisu YouTube za pomocą konta Google™.

b) Autoryzacja dla strony trzeciej

Gdy strona trzecia (jakaś aplikacja kliencka) wymaga autoryzacji, użytkownik wybiera swój zaufany serwis do potwierdzenia jego tożsamości. Może to być dowolny serwis centralnej tożsamości typu OpenID. Użytkownik autoryzuje dostęp do swoich danych dla serwisu strony trzeciej i w ten sposób uzyskuje możliwość korzystania z serwisu bez konieczności dodatkowego potwierdzania swojej tożsamości.

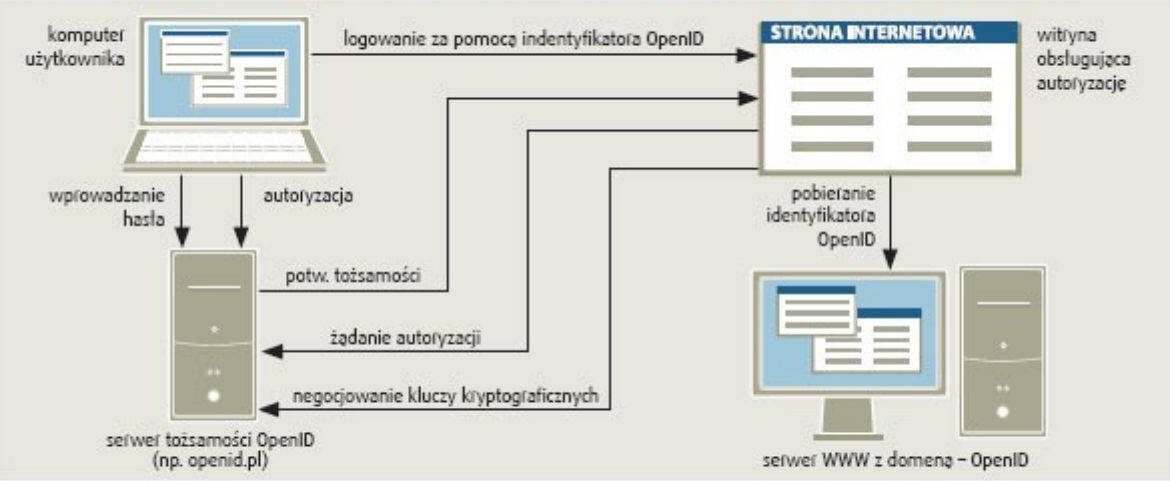
Przykładem zastosowania tego rodzaju autoryzacji jest dostęp do serwisu National Geographic™ za pomocą konta Facebook®.

4. Architektura OpenID i standard oAuth

Pojęcia te są bardzo często ze sobą mylone. Dzieje się tak, ponieważ uwierzytelnianie i autoryzacja użytkownika są ściśle ze sobą powiązanymi procedurami.

Protokół OpenID służy do uwierzytelniania użytkownika za pomocą identyfikatora (specjalnego adresu URL). Pomysłodawcą tego mechanizmu jest Brad Fitzpatrick, twórca znanego portalu blogowego LiveJournal. Sposób działania protokołu jest dość prosty. Podczas uwierzytelniania, serwer OpenID zwraca do aplikacji, do której użytkownik się loguje, jego nazwę tego użytkownika. Umożliwia to aplikacji używanie danych użytkownika. Użytkownik może również wybrać, jakie dane z jego konta zostaną udostępnione dla danej aplikacji. [10]

Warto zauważyć, że do OpenID Foundation (czyli organizacji zajmującej się specyfikacją OpenID) należą mn. takie firmy jak Google™, Microsoft®, PayPal™, Yahoo!® czy Symantec™. Dzięki temu, standard protokołu OpenID ma bardzo szeroki zasięg. [11]



Ilustracja 2: Przebieg uwierzytelniania za pomocą identyfikatora OpenID [11]

Standard oAuth obsługuje autoryzację użytkownika po jego uwierzytelnieniu. Jest zatem usługą dopełniającą funkcjonalność protokołu OpenID. Należy jednak pamiętać, że oAuth może działać bez użycia OpenID. Więcej informacji na temat uwierzytelniania oAuth na stronie 17. [12]

Protokół OpenID i standard oAuth można użyć we własnej aplikacji – istnieją gotowe biblioteki programistyczne w większości popularnych języków programowania.

5. Hasła do konta użytkownika

a) Zasady bezpieczeństwa

Hasło dostępowe jest bardzo ważnym składnikiem całej procedury uwierzytelniania. Hasło to *tajne słowo lub kod, pełniące funkcję środka zapobiegającego nieautoryzowanego dostępu do danych użytkownika*. [13] Bardzo ważne jest bezpieczne przechowywanie haseł i algorytmów po stronie serwera. Podstawowe założenia to:

- nieprzechowywanie hasła w formie jawnej;
- szyfrowanie hasła za pomocą mocnej funkcji jednostronnej;
- wymuszanie odpowiedniej złożoności hasła na użytkowniku;
- odseparowanie danych aplikacji od danych użytkowników.

Więcej informacji na temat bezpieczeństwa w dokumencie RFC4086 [14].

b) Rodzaje poświadczeń

Hasło użytkownika jest jednym z kilku typów poświadczenia użytkownika. Ogólnie można wyróżnić następujące typy poświadczenia to:

- hasła stałe – zapisane w bazie danych hasło użytkownika;

- hasła jednorazowe – lista haseł udostępniana np. w formie zdrapek lub SMS-em;
- tokeny – czyli urządzenia generujące unikalny kod;
- certyfikaty cyfrowe;
- numery telefonów;

W zależności od potrzeb aplikacja może wymagać jednego lub więcej z tych poświadczeń w celu uwierzytelnienia bądź autoryzacji.

c) Tworzenie hasła

Warto zapoznać się ze sposobami łamania i wykradania haseł, aby móc utworzyć dobre hasło. Najczęściej stosowanymi metodami łamania hasła jest atak słownikowy i siłowy. Używane są również bazy danych zawierających *hashe* wielu słów. Inną metodą jest pośrednie lub bezpośrednie szpiegowanie użytkownika. [15] Więcej informacji na temat haseł można znaleźć w internecie oraz niezliczonych artykułach prasowych (zob. [16], [17], [18] i [19]).

6. Protokół LDAP

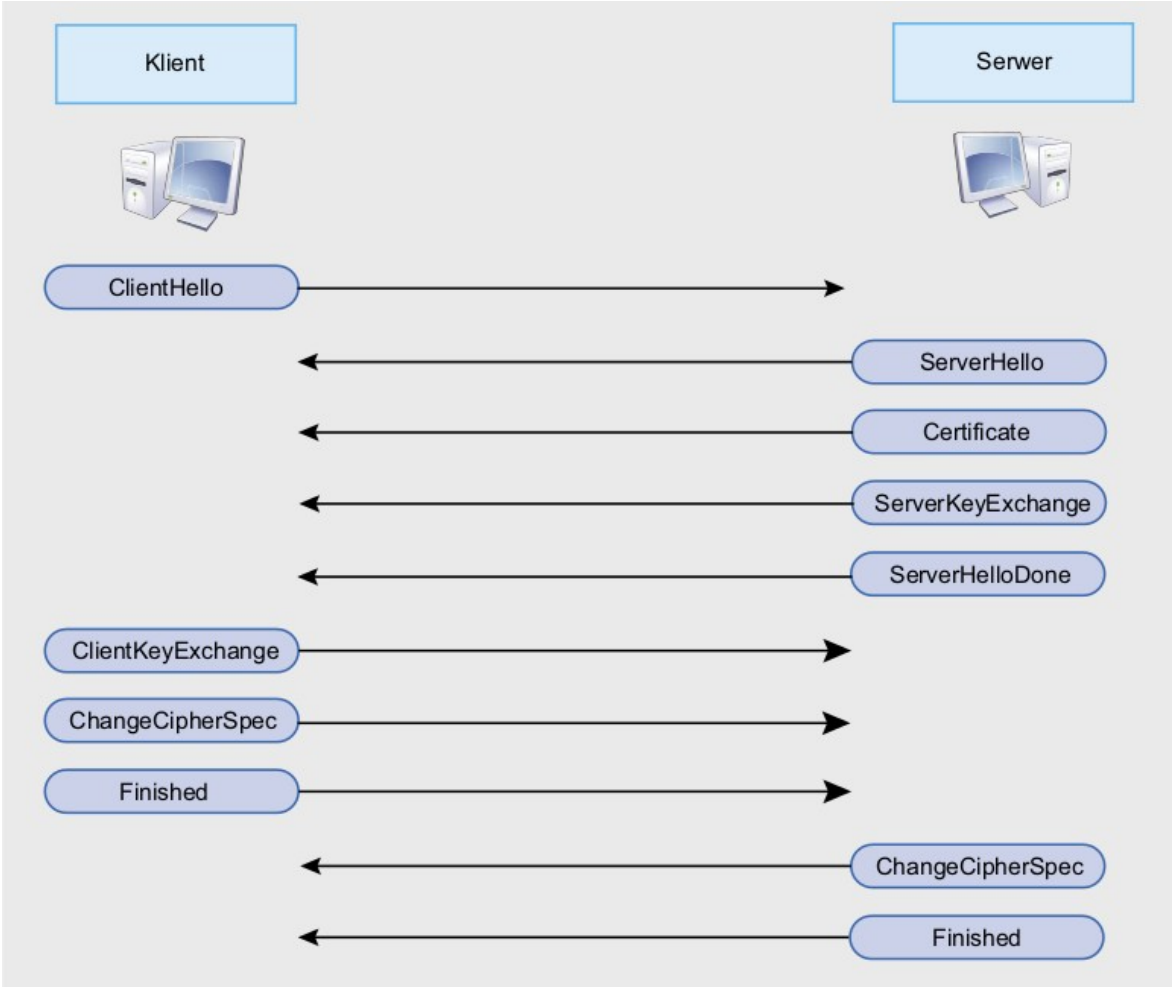
LDAP (ang. *Lightweight Directory Access Protocol*) to *nieskomplikowany protokół dostępu do katalogów*. Powstał jako uproszczona wersja protokołu DAP, wykorzystywanego do uzyskania dostępu do usług katalogowych w standardzie X.500. [20]

Protokół ten wymaga, by dane były grupowane w strukturze przypominającej drzewo katalogów. Każdy obiekt jest jednoznacznie identyfikowany poprzez swoje położenie w drzewie. Katalogi są natomiast wyspecjalizowaną bazą danych przeznaczoną do czytania, przeglądania i przeszukiwania. Zwykle zawierają informację opartą na deskryptorach i atrybutach, ale nie oferują mechanizmów transakcji ani innych usług zapewnianych przez tradycyjne systemy bazodanowe. Ponadto, katalogi są zaprojektowane, tak by dawać szybką odpowiedź na zapytanie. Dodatkowo mają możliwość tworzenia replik informacji w celu uczynienia ich bardziej dostępnymi oraz aby podnieść poziom niezawodności. Protokół ten umożliwia również skalowalność i zapewnia bezpieczeństwo danych. [21] [22]

7. Połączenia szyfrowane

Jedno z największych niebezpieczeństw podczas używania aplikacji internetowych jest związane z przesyłaniem danych pomiędzy użytkownikiem i serwerem. Według specyfikacji protokołu HTTP, wszelkie informacje przy jego użyciu przesyłane są w formie jawnej. Stwarza to możliwość bezproblemowego przeczytania danych użytkownika. Nie stanowi to problemu podczas zwykłego przeglądania zasobów internetu. Jest natomiast nie pożądane, gdy użytkownik korzysta z serwisów społecznościowych, kont pocztowych, bankowości internetowej czy innych zawierających

dane wrażliwe. Dlatego wszelkie działania obsługujące uwierzytelnianie, autoryzację i zarządzanie kontem powinny być wykonywane przy użyciu bezpiecznego, szyfrowanego połączenia. Dla protokołu HTTP stosuje się mechanizm szyfrowania SSL znany jako HTTPS. Dodatkową zaletą używania połączeń szyfrowanych jest możliwość weryfikacji i sprawdzenia tożsamości odwiedzanej witryny internetowej. [23]



Ilustracja 3: Nawiązywanie połączenia klient-serwer poprzez SSL [23]

Rozdział 2. Projekt aplikacji

1. Założenia

1.1. Bezpieczeństwo połączeń

Aby zapewnić odpowiednie bezpieczeństwo danych podczas używania systemu, aplikacja powinna spełniać kilka następujących warunków.

1. Dostęp do serwisu centralnego logowania odbywa się przy użyciu połączenia szyfrowanego (HTTPS). Wymaga to poprawnego wygenerowania i podpisania certyfikatów serwera.
2. Komunikacja z bazą danych użytkowników ma być realizowana w sposób niwelujący możliwość nadużyć. To założenie umożliwia LDAP.

1.2. Bezpieczeństwo danych użytkownika

System ma wykluczać możliwość kradzieży elektronicznej tożsamości.

1. Po kilkukrotnym błędnym logowaniu konto użytkownika zostaje wyłączone. Na adres e-mailowy jest przesłana wiadomość o tym zdarzeniu wraz z instrukcją aktywacji konta.
2. Każdorazowa zmiana hasła wymaga podania również starego (aktualnego) hasła.
3. Zmiana kontaktowego adresu e-mailowego musi być dodatkowo potwierdzona i umożliwiać cofnięcie tej zmiany.

Ilustracja 4: Wygląd aplikacji: lista zarejestrowanych na serwerze aplikacji klienckich

1.3. Podstawowe funkcjonalności

Każdy użytkownik ma mieć możliwość zarządzania swoim profilem. Chodzi tu przede wszystkim o możliwość edycji swoich danych personalnych, takich jak nazwa, adres e-mail czy hasło. Ważne jest również zarządzanie swoimi sesjami (aktywnymi uwierzytelnieniami) oraz autoryzacją serwisów, w szczególności anulowanie danej usłudze dostępu do danych użytkownika.

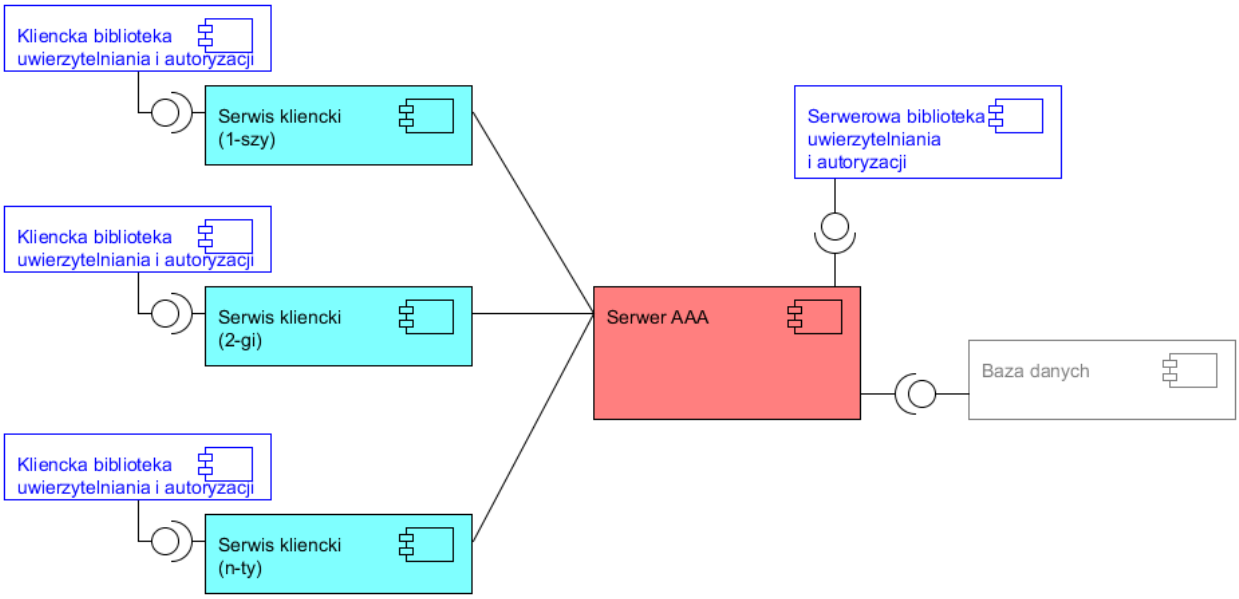
Dodatkowo użytkownik powinien mieć możliwość ustawienia parametrów subskrypcji zdarzeń (np. powiadamianie o błędnych uwierzytelnianiach, czy autoryzacjach).

Inną ważną funkcjonalnością jest możliwość zarządzania przez administratora całego systemu użytkownikami, w taki sam sposób jak to może zrobić użytkownik ze swoim kontem. Dzięki temu będzie możliwe zdalne sprawdzenie sesji, wylogowanie, zablokowanie lub wyłączenie danego użytkownika.

Administrator zarządza również zarejestrowanymi w systemie aplikacjami klienckimi. Może zobaczyć użytkowników, którzy autoryzowali danej aplikacji dostęp do swoich danych oraz anulować tę autoryzację.

2. Architektura systemu

Cały system składa się z aplikacji usługodawcy (umożliwia przeprowadzenie operacji na koncie klienta) oraz aplikacji klienta – instalowana na stronie internetowej, która ma korzystać z zasobów aplikacji usługodawcy.



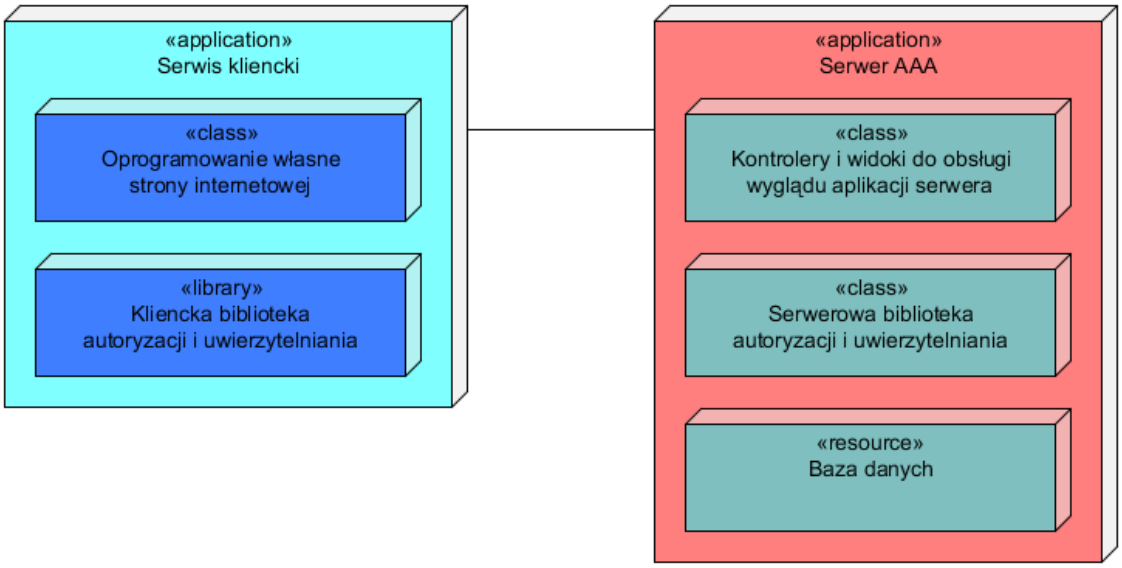
Ilustracja 5: Diagram komponentów

2.1. Aplikacja usługodawcy

Podstawowym zadaniem aplikacji centralnej jest komunikacja pomiędzy użytkownikiem a bazą danych użytkowników oraz komunikacja z aplikacją kliencką. Przy pomocy dodatkowej biblioteki, uwierzytelnia użytkownika i autoryzuje aplikację klienta w systemie.

2.2. Aplikacja klienta

Aplikacja kliencka serwisu internetowego to biblioteka programistyczna, zawierająca podstawowe narzędzia do komunikacji z aplikacją centralną.



Ilustracja 6: Diagram struktury

3. Opis technologii

Do tworzenia aplikacji internetowych można używać wielu języków programowania bądź skryptowych. Jednym z najpopularniejszych (zob. [24]) jest PHP[1] stworzony w 1994 roku przez Rasmusa Lerdorfa. Język ten został zaprojektowany głównie do generowania stron internetowych w czasie rzeczywistym, lecz możliwe jest również wykonywanie skryptów z linii poleceń oraz pisanie samodzielnych aplikacji z interfejsem graficznym. Dużą zaletą PHP jest budowa hybrydowa – aplikacje można pisać obiektowo i strukturalnie. Jednak z drugiej strony, język ten nie jest w pełni obiektowy, co przy dużych projektach jest sporą wadą. Należy jednak zaznaczyć, że ze względu na sporą popularność, PHP jest dobrze udokumentowanym językiem – zarówno w formie elektronicznej (oficjalna dokumentacja, poradniki, fora dyskusyjne) jak i w formie papierowej (niezliczona ilość publikacji). [25]

1 Nazwa tego języka programowania jest rekursywnym akronimem – PHP: Hypertext Preprocessor.

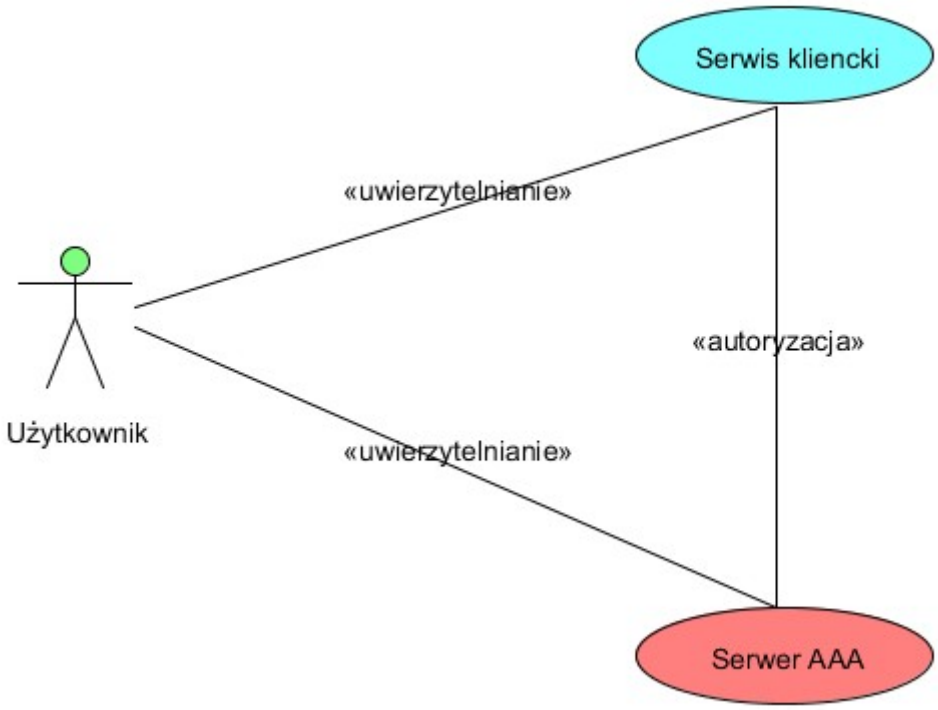
Budowa zaawansowanych aplikacji jest często wspierana przy użyciu frameworka. W porównaniu z innymi frameworkami PHP, Kohana jest szybsza i łatwiejsza w obsłudze – daje to dość dużą swobodę w tworzeniu aplikacji. Framework ten powstał w 2007 roku na bazie CodeIgnitera, podstawą jest architektura MVC w PHP 5. [26]

4. Opis działania aplikacji

4.1. Relacje pomiędzy elementami systemu

Użytkownik uwierzytelnia się na serwerze AAA oraz w serwisie klienckim. Natomiast proces autoryzacji po uzyskaniu zgody przez użytkownika, przebiega pomiędzy serwerem AAA i serwisem klienckim.

- Proces uwierzytelniania działa jako autorska aplikacja zarządzania użytkownikami. Istnieje możliwość rozszerzenia uwierzytelniania, tak by działało przy użyciu zewnętrznych operatorów OpenID oraz utworzenia własnego serwera OpenID.
- Proces autoryzacji jest przeprowadzany przy użyciu standardu oAuth w wersji 1.0a.



Ilustracja 7: Diagram relacji pomiędzy elementami systemu

4.2. Schemat przebiegu uwierzytelniania

a) Uwierzytelnianie w aplikacji serwera

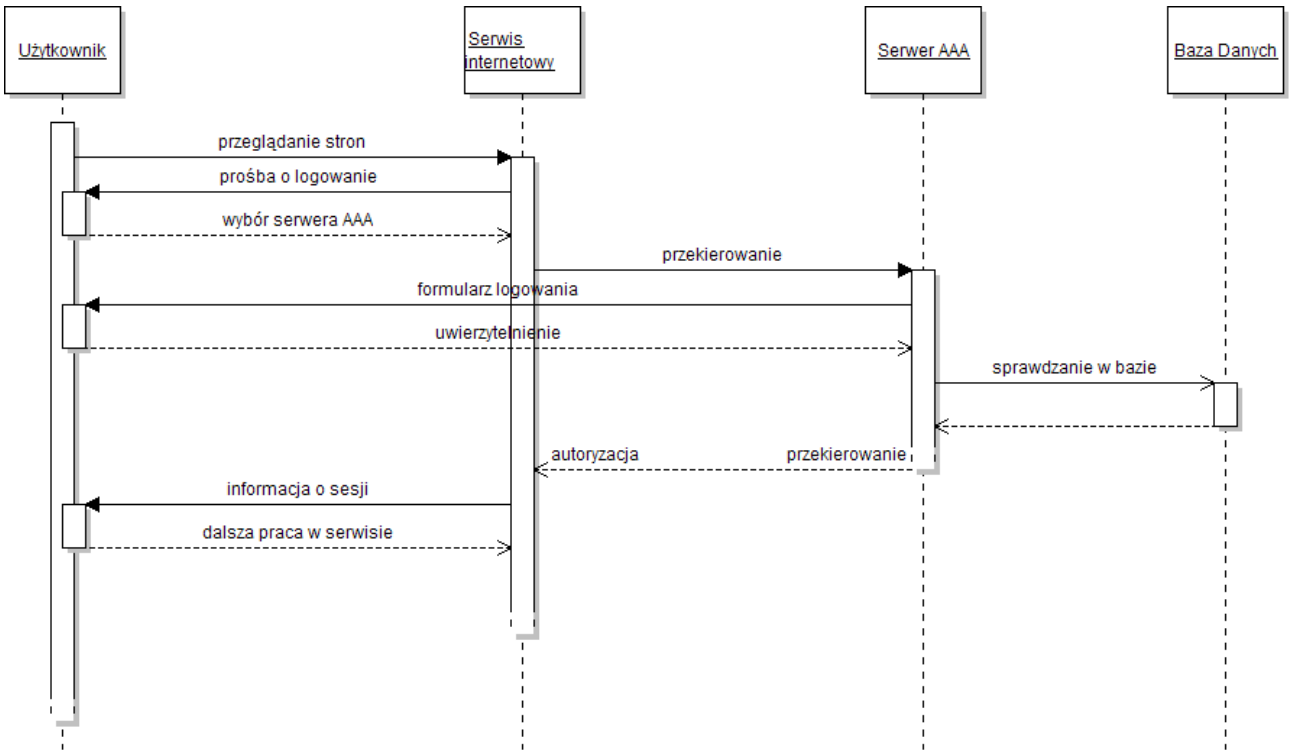
W procesie uwierzytelniania uczestniczy 2 aktorów: użytkownik i aplikacja serwera. Aby uczestniczyć w procesie uwierzytelniania, użytkownik musi być zarejestrowany w bazie serwera.

Uwierzytelnianie składa się z 2 części. Po pierwsze użytkownik wprowadza swoje dane logowania (adres e-mail i hasło) w formularzu na stronie serwera i przesyła je do serwera. Po drugie serwer sprawdza poprawność wysłanych danych.

Sprawdzanie poprawności danych przebiega następująco. Serwer szuka w bazie danych użytkownika o podanym loginie. Następnie porównuje przesłane hasło z pobranym z bazy. W kolejnym kroku sprawdzany jest status użytkownika. Jeżeli wszystko przebiega poprawnie, to następuje uwierzytelnienie użytkownika – zapisanie tokenu sesji w bazie danych. W przeciwnym wypadku zostaje wyświetlony stosowny komunikat o błędzie.

b) Wylogowanie z serwera

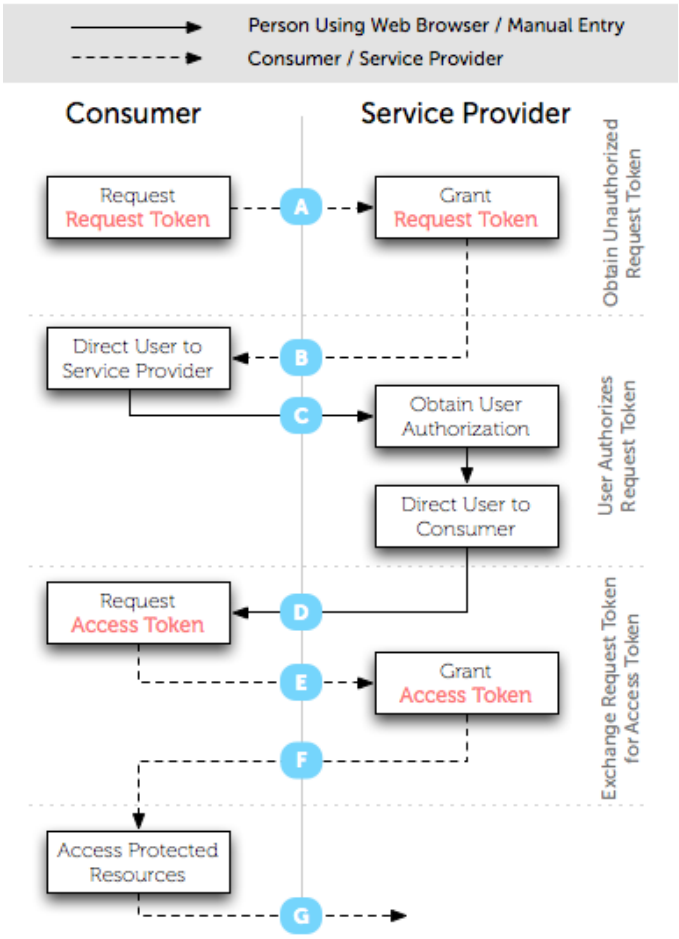
Wylogowanie się z serwera, czyli cofnięcie uwierzytelnienia dla użytkownika z określonego miejsca dostępu następuje poprzez kliknięcie przez użytkownika w odpowiedni link w panelu zarządzania kontem po stronie serwera. Serwer przetwarza to żądanie, usuwając token sesji z bazy danych.



Ilustracja 8: Ogólny schemat uwierzytelniania i autoryzacji - diagram sekwencji

4.3. Schemat przebiegu autoryzacji

W całym procesie autoryzacji oAuth uczestniczy 3 aktorów: użytkownik, aplikacja kliencka oraz aplikacja serwera. Aby uczestniczyć w procesie autoryzacji, użytkownik i aplikacja kliencka muszą być zarejestrowani w bazie serwera.



Ilustracja 9: Przebieg autoryzacji oAuth [12]

a) Działania przed podjęciem autoryzacji

1. Jeżeli użytkownik jest uwierzytelniony u klienta, to znaczy, że klient ma autoryzację do danych użytkownika i posiada jego numer ID. Procedura autoryzacji nie musi być przeprowadzana.
2. Po wysłaniu żądania autoryzacji przez użytkownika, klient wysyła do serwera żądanie sprawdzenia czy użytkownik jest uwierzytelniony. Serwer może przetworzyć żądanie na 3 sposoby:
    1. Jeżeli użytkownik nie jest uwierzytelniony, to następuje przekierowanie na formularz logowania. Po uwierzytelnieniu serwer zwraca do klienta numer ID użytkownika.
    2. Jeżeli uwierzytelnianie zostanie przerwane, to serwer zwraca do klienta kod błędu.
    3. Jeżeli użytkownik jest uwierzytelniony na serwerze, to serwer zwraca do klienta numer ID użytkownika.
3. Jeżeli klient dostał numer ID użytkownika, to inicjuje autoryzację, która przebiega według standardu oAuth. W przeciwnym wypadku wyświetla stosowny komunikat o błędzie.

b) Autoryzacja oAuth

Autoryzacja działa na podstawie wymiany i podpisywania tokenów: tokenu zapytania (*request token*) i tokenu dostępu (*access token*). Każdy z tych kluczy ma określony „czas życia” (TTL). Na diagramie powyzej linią kreskową zaznaczono przebieg wykonywany bez udziału użytkownika.

Na żądanie aplikacji klienckiej (A) serwer generuje nieautoryzowany *request token* i przesyła go (B) z powrotem do klienta. W kolejnym kroku (C) klient przekierowuje użytkownika na stronę serwera, gdzie użytkownik ma wyrazić zgodę na pobieranie jego danych przez aplikację kliencką z serwera. Po akceptacji serwer przekierowuje (D) użytkownika na stronę klienta, który wyświetla informacje o pozytywnym (lub nie) przebiegu autoryzacji tokena. Ostatnim krokiem procesu autoryzacji jest wygenerowanie *access tokena* (E) przez serwer i zwrócenie go (F) do klienta. Na podstawie *access tokena* klient będzie mógł pobierać (G) dane użytkownika z serwera.

c) Anulowanie uwierzytelnienia

Użytkownik może anulować uwierzytelnianie dla danej aplikacji klienta bez anulowania uwierzytelnienia w aplikacji serwera. W tym celu użytkownik musi wybrać odpowiednią opcję w panelu zarządzania swoim kontem po stronie klienta. Klient przetwarza to żądanie, usuwając token sesji z bazy danych.

Należy pamiętać, że cofnięcie uwierzytelnienia nie jest jednoznaczne z cofnięciem autoryzacji. Po cofnięciu uwierzytelnienia, użytkownik może ponownie zalogować się do aplikacji klienckiej bez ponownego potwierdzania autoryzacji dostępu (szczegóły powyżej w punkcie a).

d) Anulowanie autoryzacji

Aby anulować autoryzację aplikacji klienta, użytkownik musi wybrać odpowiednią opcję w panelu zarządzania swoim kontem po stronie klienta lub serwera. Proces anulowania przebiega jako zwykłe zapytanie klient-serwer z wykorzystaniem klucza dostępu (*access token*), przy czym klient wysyła dodatkowy parametr TTL o wartości ujemnej.

4.4. Operacje na koncie użytkownika

a) Schemat zakładania konta

Aby założyć konto na serwerze, użytkownik musi posiadać adres e-mail. W formularzu na stronie ma wprowadzić ten adres i wymyślone przez siebie hasło oraz wysłać formularz. Konto nie zostanie jednak wysłane od razu, gdyż wymagane jest potwierdzenie chęci założenia konta za pomocą specjalnego kodu przesłanego na adres e-mail.

b) Schemat aktywacji konta

Gdy użytkownik trzykrotnie wpisze błędne hasło dostępowe do swojego konta, to jego konto zostaje zablokowane. Na kontaktowy adres e-mail zostaje przesłana informacja o blokadzie wraz z linkiem aktywującym konto. Taki link można również wygenerować za pomocą formularza aktywacji konta użytkownika. Wystarczy podać login lub kontaktowy adres e-mail.

c) Schemat usuwania konta

Usunięcie konta również odbywa się za pomocą linku potwierdzającego wysyłanego na kontaktowy adres e-mail. Aby usunąć swoje konto na serwerze, użytkownik musi wprowadzić w formularzu swój login lub kontaktowy adres e-mail.

d) Schemat zmiany hasła

Gdy użytkownik zapomni swoje hasło, może zlecić wygenerowanie nowego. W formularzu należy wprowadzić login lub kontaktowy adres e-mail. Po przesłaniu formularza na kontaktowy adres e-mail zostanie przesłane nowe hasło oraz link aktywujący to hasło.

Po potwierdzeniu nowego hasła i zalogowaniu do panelu, użytkownik może ustawić swoje hasło, korzystając z odpowiedniego formularza. Wymaga on wprowadzenia starego hasła oraz dwukrotnie nowego. Minimalna długość hasła to 8 znaków.

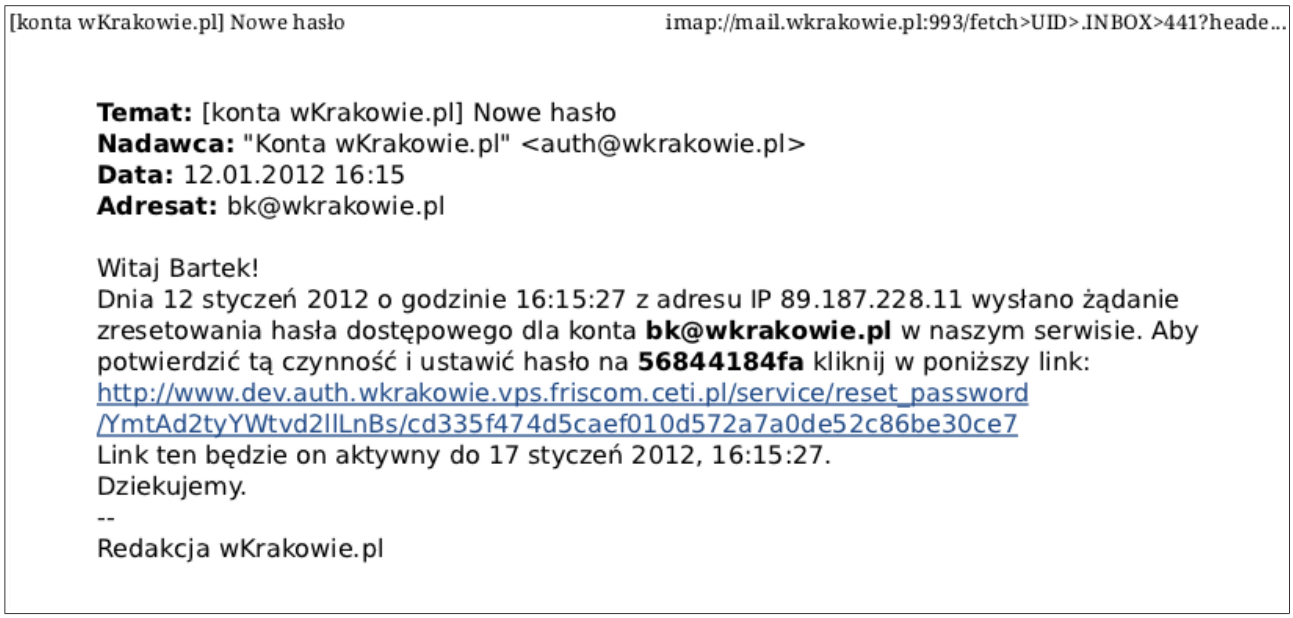
e) Schemat zmiany adresu e-mail

Na nowy adres zostaje przesłany link potwierdzający zmianę adresu (link ten ma krótką datę ważności). Natomiast na stary adres zostaje przesłana informacja o zmianie wraz z linkiem anulującym zmianę adresu kontaktowego (link ten ma długą datę ważności).

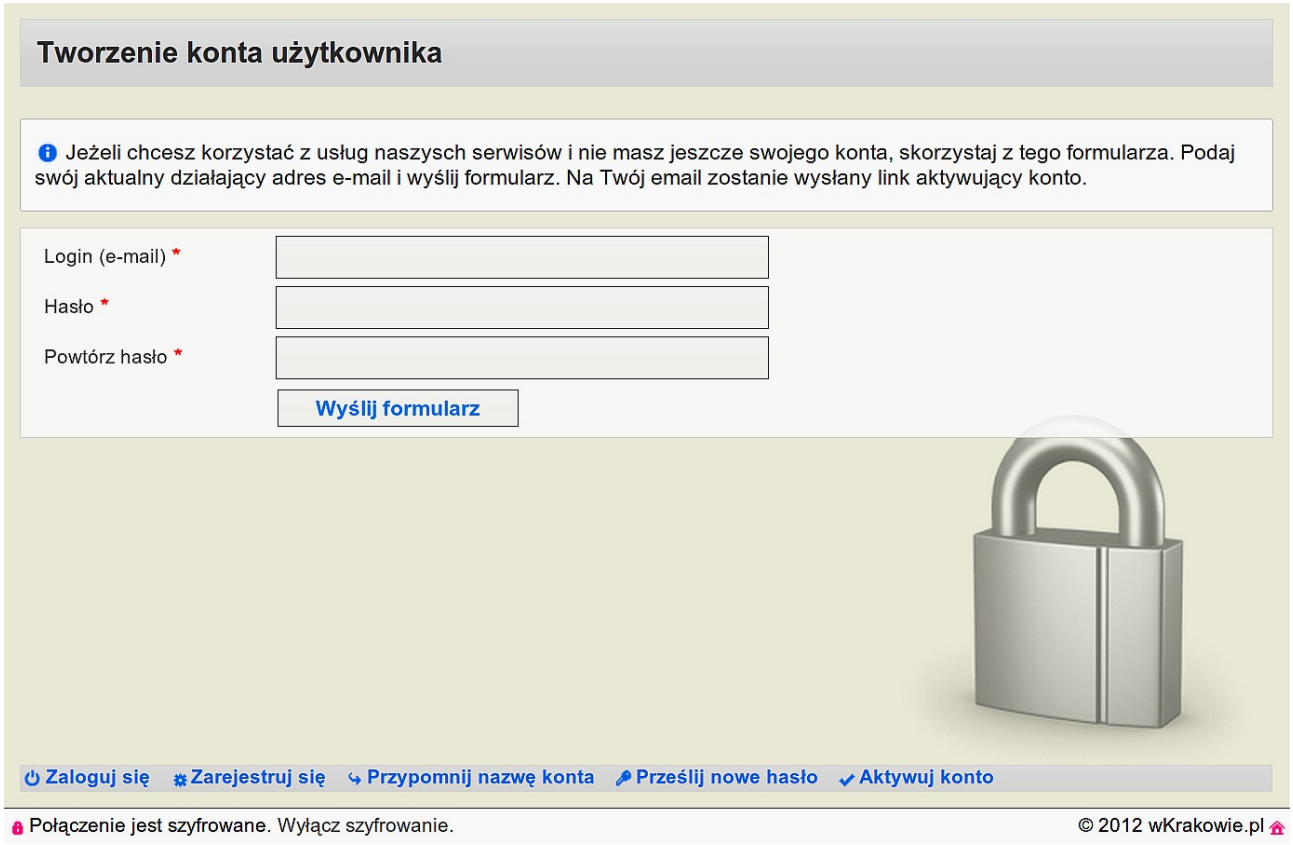
4.5. Kody wysyłane na e-mail

Wszelkie operacje na koncie wymagające dodatkowego zabezpieczenia są potwierdzane za pomocą kodów jednorazowych. Podczas zlecania akcji serwer generuje parę kluczy (publiczny i prywatny) przypisane do danego użytkownika i wysyła na kontaktowy adres e-mail użytkownika link zawierający klucz publiczny. Użytkownik klikając w ten link, otwiera stronę serwera, który przetwarza żądanie, porównując nazwę użytkownika i klucz publiczny z danymi zapisanymi w bazie danych.

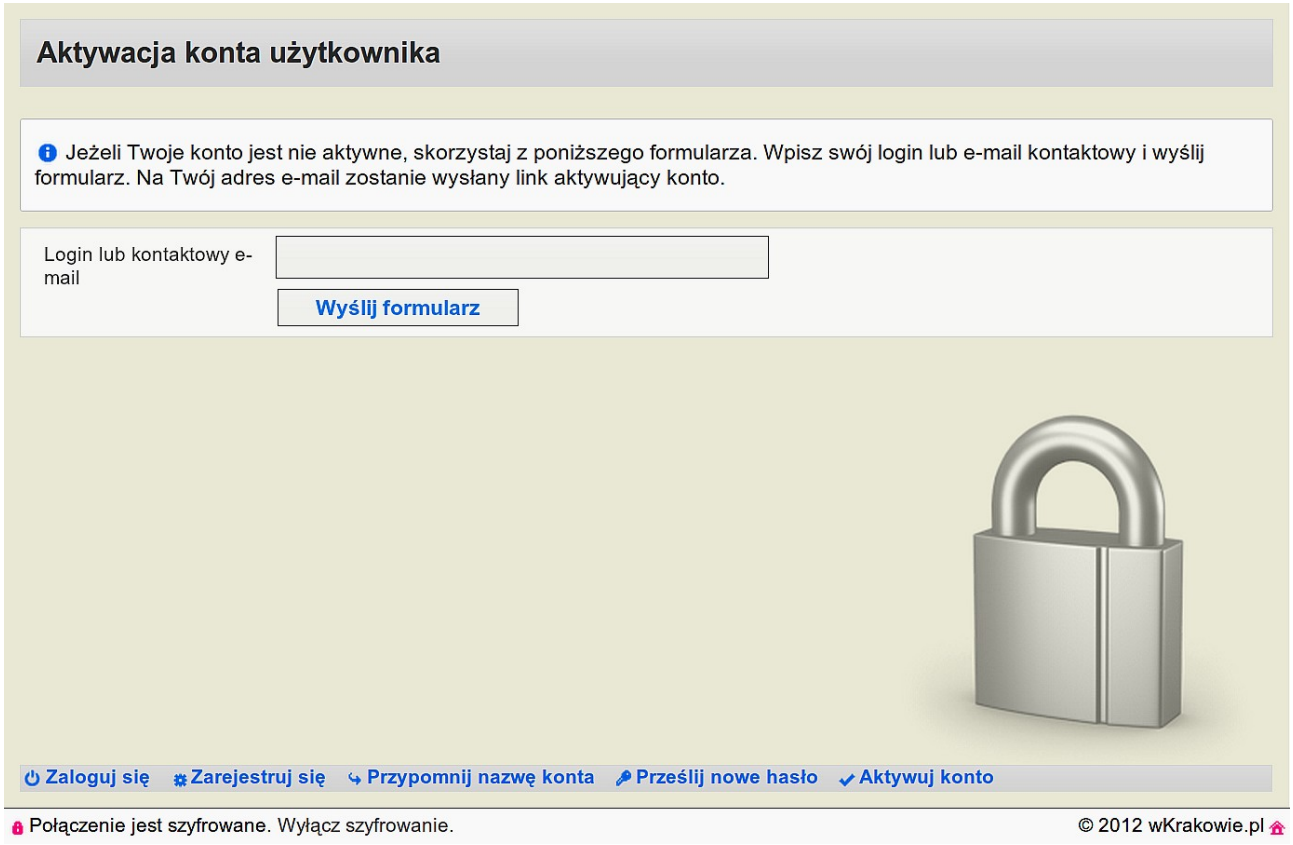
5. Wygląd aplikacji



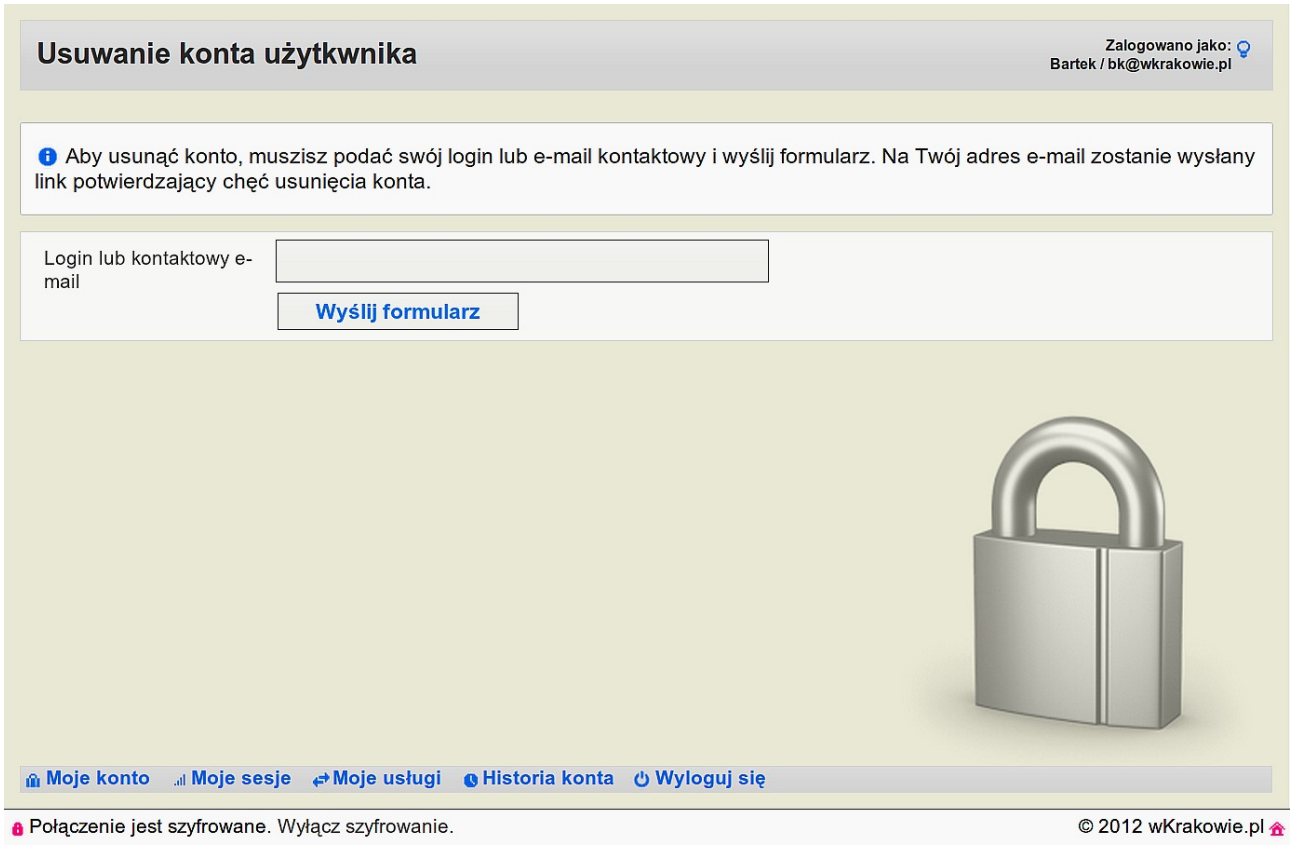
Ilustracja 10: Przykładowy e-mail - nowe hasło



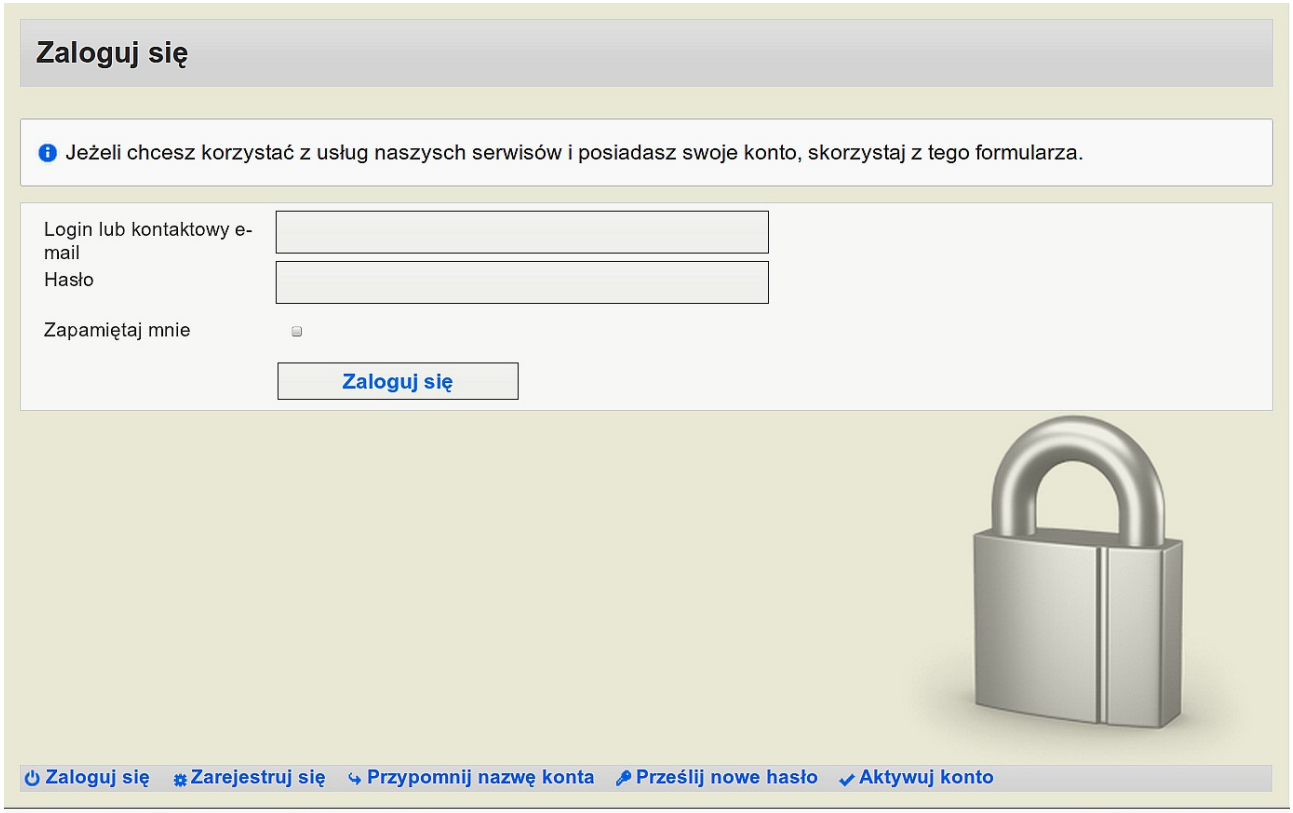
Ilustracja 11: Wygląd aplikacji: tworzenie konta użytkownika



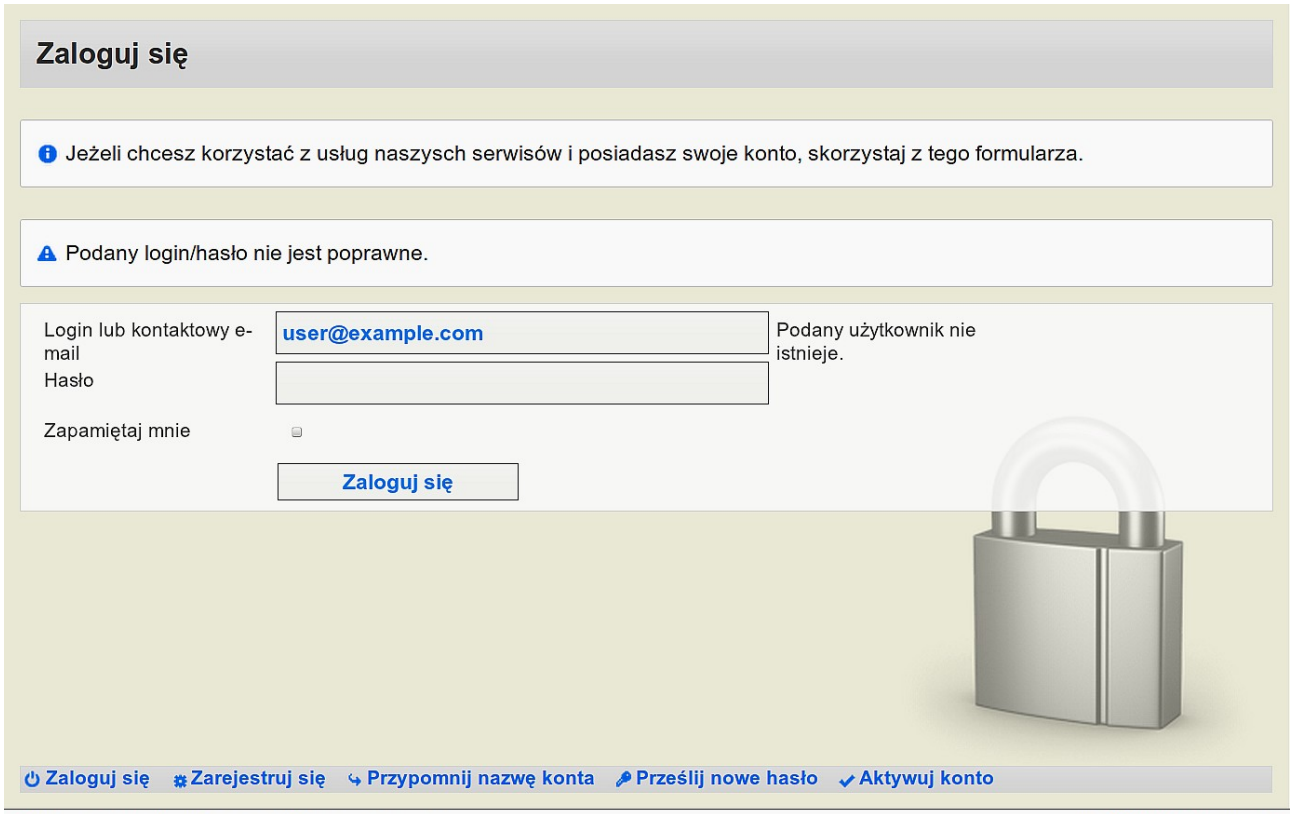
Ilustracja 12: Wygląd aplikacji: aktywacja konta użytkownika



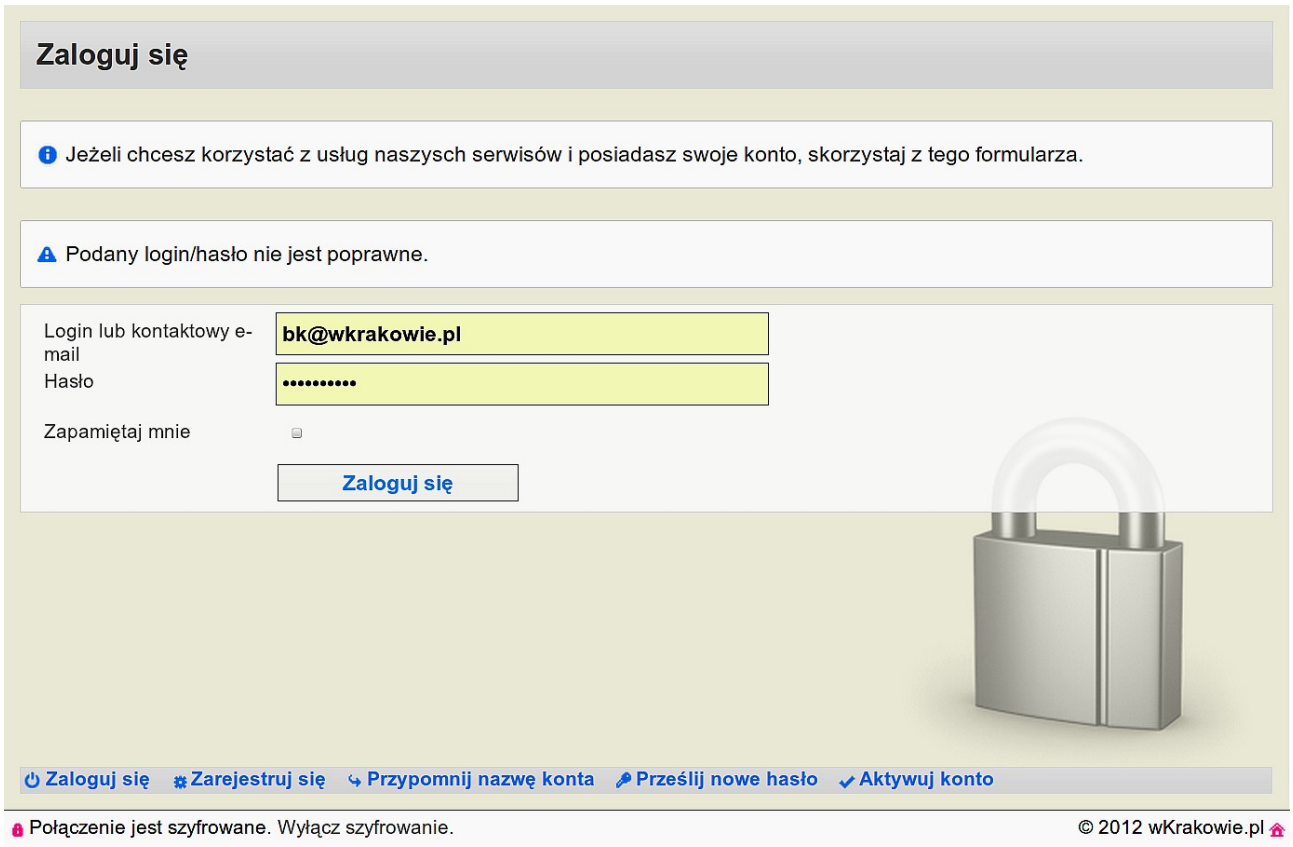
Ilustracja 13: Wygląd aplikacji: usuwanie konta użytkownika



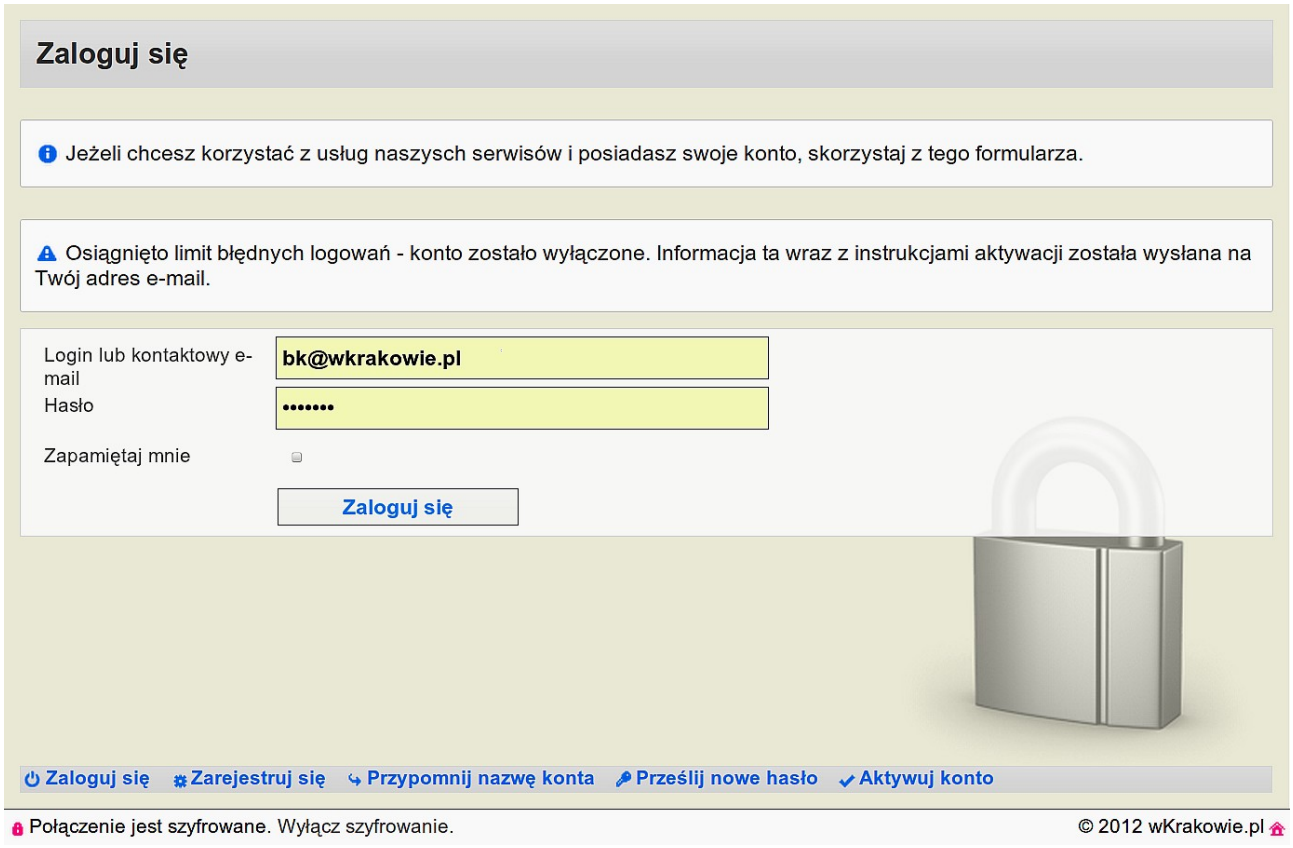
Ilustracja 14: Wygląd aplikacji: formularz logowania



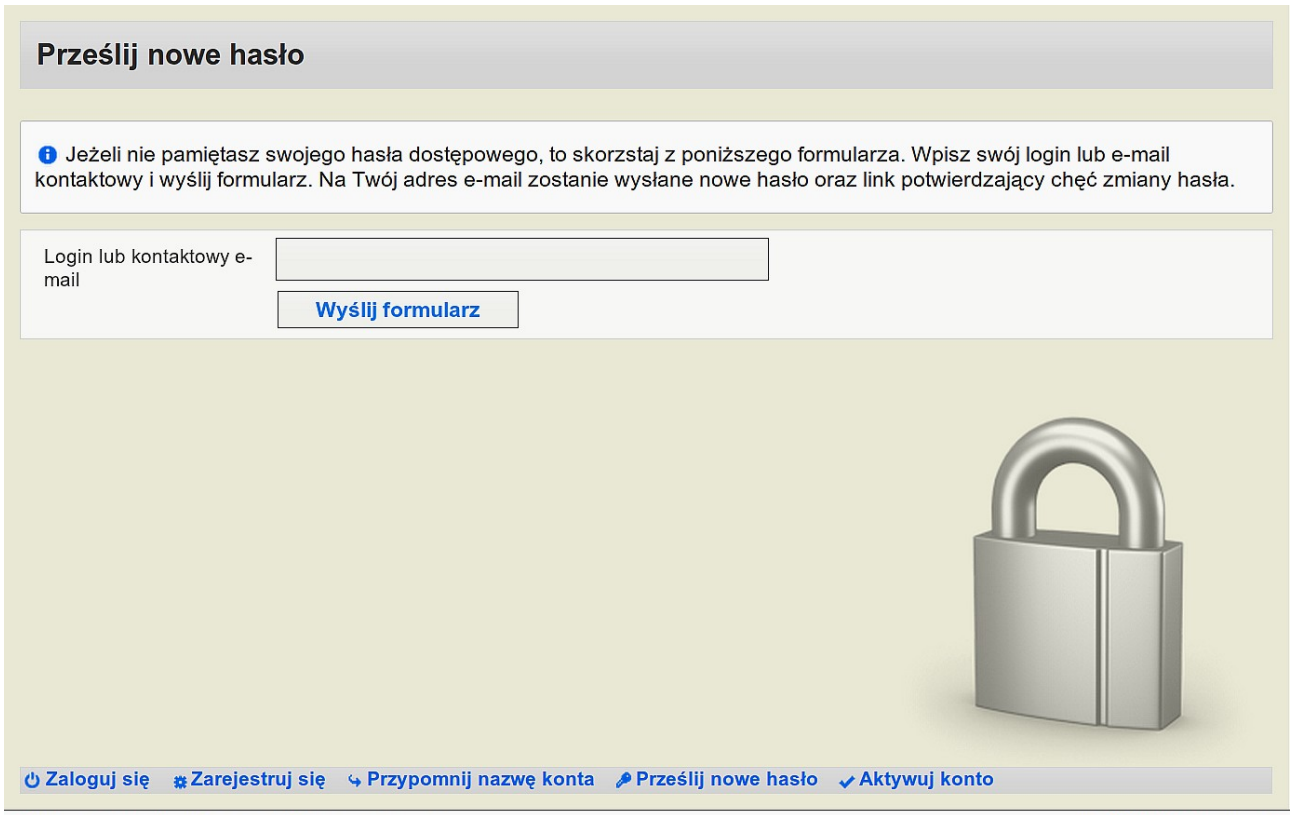
Ilustracja 15: Wygląd aplikacji: formularz logowania - komunikat o braku użytkownika



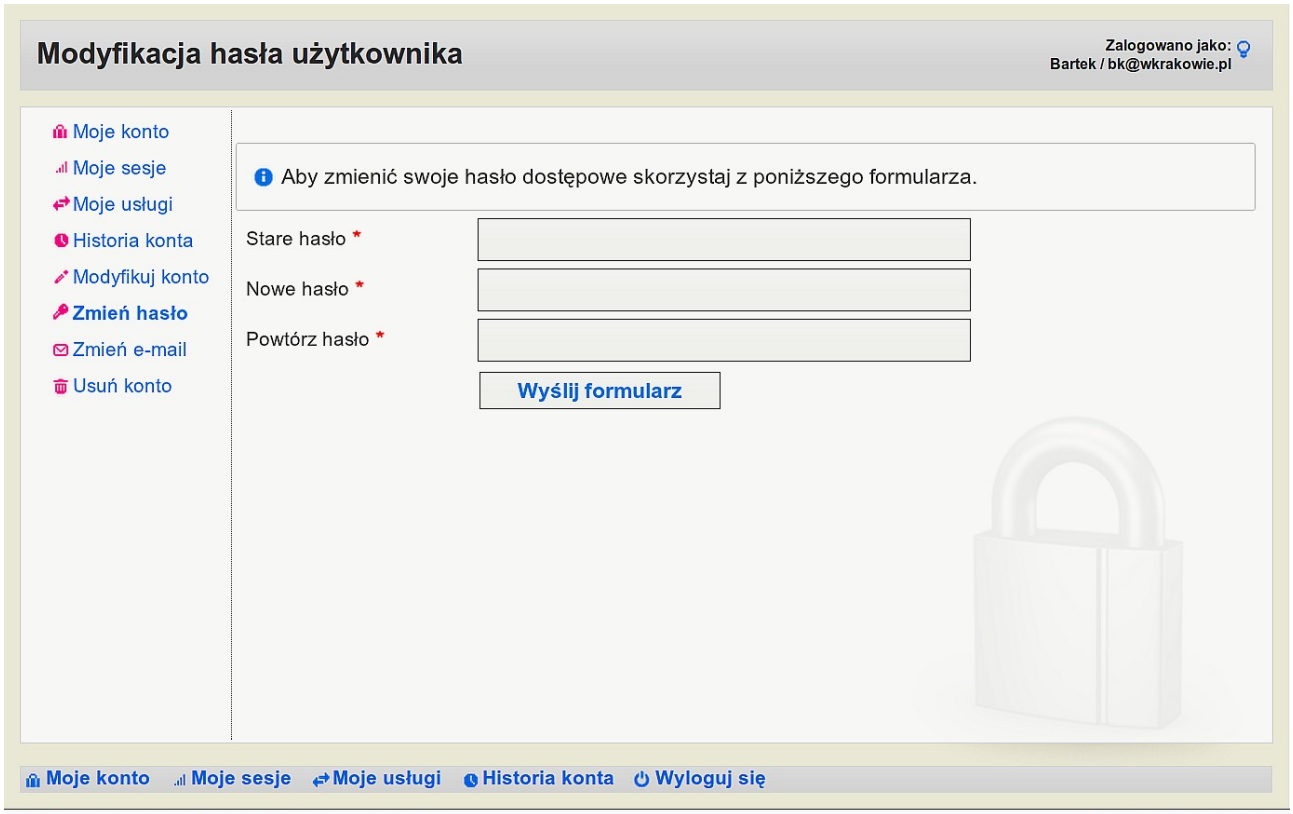
Ilustracja 16: Wygląd aplikacji: formularz logowania - komunikat o błędzie logowania



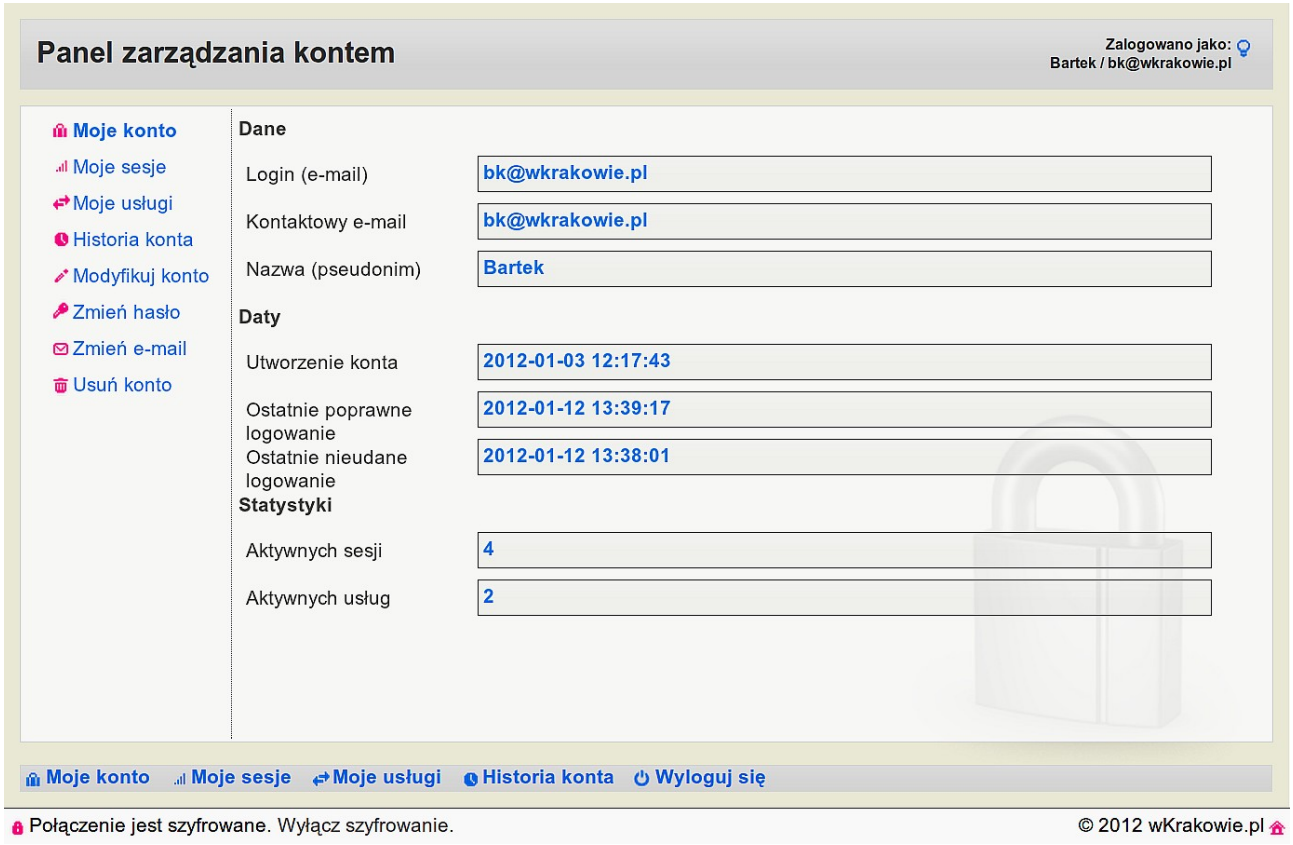
Ilustracja 17: Wygląd aplikacji: formularz logowania - komunikat o blokadzie konta



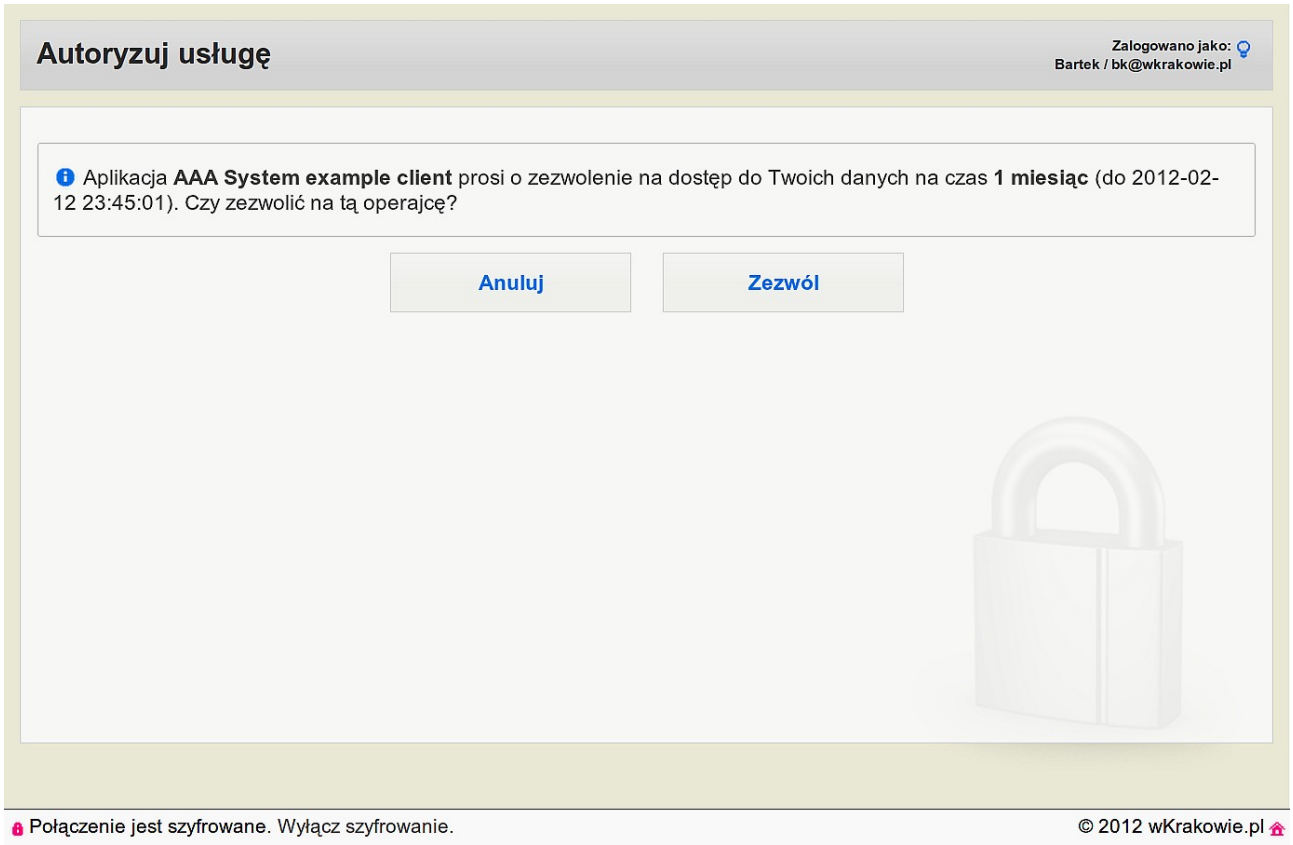
Ilustracja 18: Wygląd aplikacji: resetowanie hasła konta użytkownika



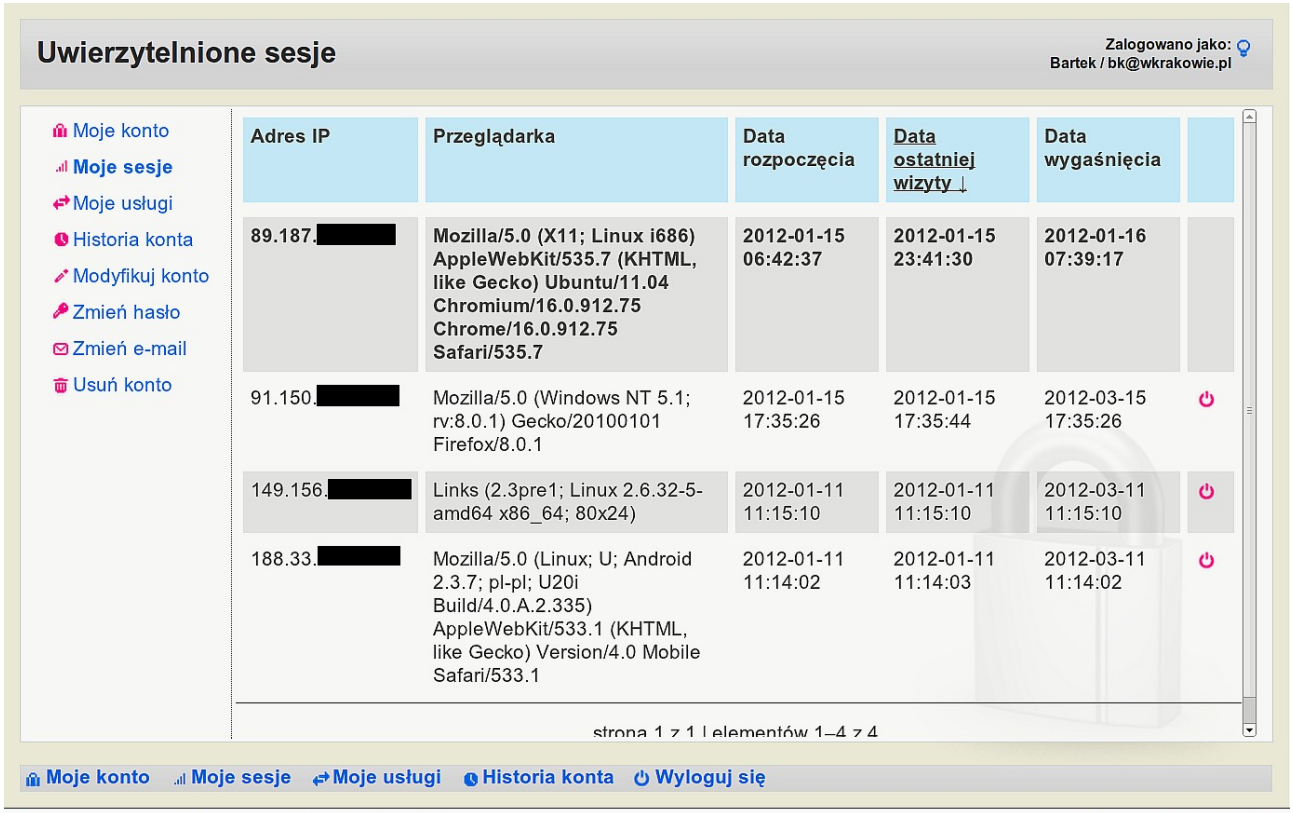
Ilustracja 19: Wygląd aplikacji: modyfikacja hasła konta użytkownika



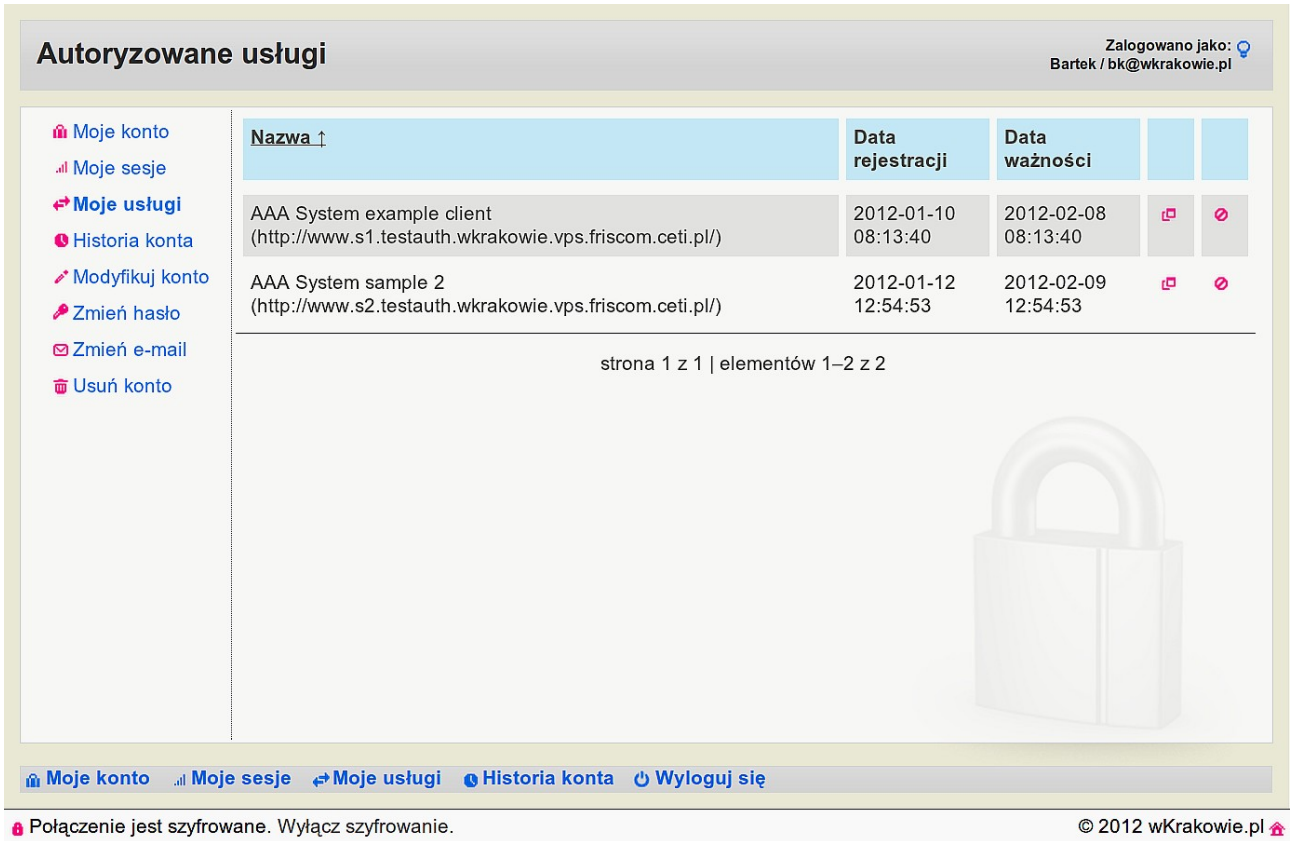
Ilustracja 20: Wygląd aplikacji: panel zarządzania kontem użytkownika



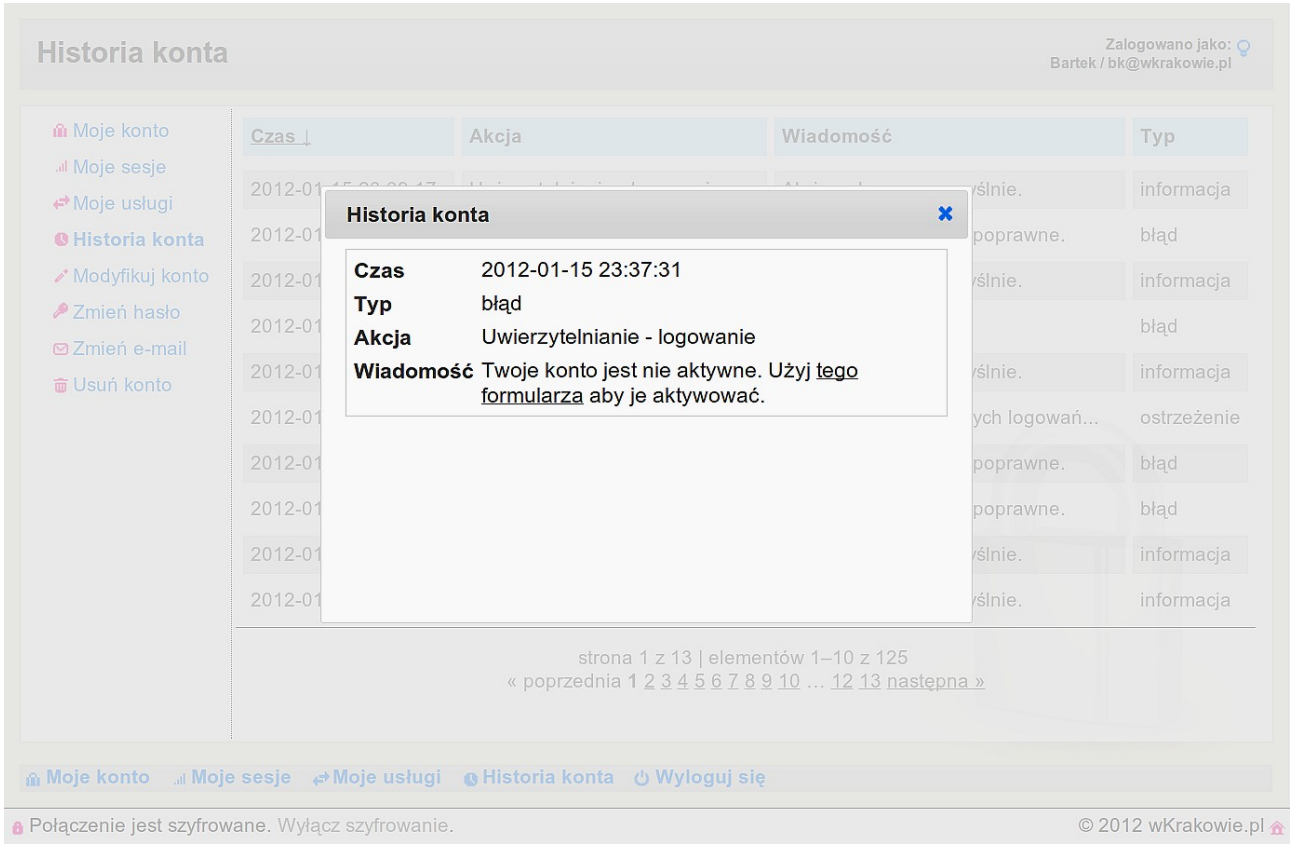
Ilustracja 21: Wygląd aplikacji: formularz autoryzowania usługi (aplikacji klienckiej)



Ilustracja 22: Wygląd aplikacji: lista uwierzytelnionych sesji użytkownika



Ilustracja 23: Wygląd aplikacji: lista autoryzowanych usług (aplikacji) użytkownika



Ilustracja 24: Wygląd aplikacji: historia konta użytkownika

Zakończenie

W niniejszej pracy zaprezentowany został system centralnego uwierzytelniania, autoryzacji i zarządzania kontem użytkownika. Przedstawione zagadnienia bezpieczeństwa i sposobu działania aplikacji, pozwalają na wysnucie wniosków.

Po pierwsze podmiot oferujący centralną aplikację musi być na tyle wiarygodny dla użytkownika, by ten zechciał korzystać z jego usług.

Korzystanie z takiego serwisu wiąże się z szeregiem niebezpieczeństw. Przede wszystkim posiadanie globalnego konta stwarza możliwość łatwego przejęcia internetowej tożsamości danego użytkownika. Jest to spowodowane tym, że takie konto posiada wiele danych użytkownika i informacji o nim. Z drugiej jednak strony, taka centralizacja danych jest niewątpliwą zaletą. Dzięki temu nie trzeba za każdym razem wypełniać formularzy rejestracji i zapamiętywać kolejnego hasła.

Usługa centralnego logowania ułatwia także zarządzaniem i udostępnianiem danych osobowych w różnych serwisach internetowych. Może także służyć jako centralny notatnik prywatnych wiadomości, do którego dostęp jest możliwy z dowolnej innej aplikacji i miejsca. Dodatkowo wszelkie autoryzacje odbywają się za pomocą zaufanej strony trzeciej (serwera AAA), co pozwala na lepszą ochronę danych użytkownika.

Należy również pamiętać, że sama aplikacja musi być odpowiednio zabezpieczona – dotyczy to zarówno danych przechowywanych na serwerze, jak i połączenia z użytkownikiem. Niestety najsłabszym elementem całego systemu jest sam użytkownik, który może przez nie uwagę udostępnić swoje dane nie tam gdzie trzeba. [11]

Indeks ilustracji

Ilustracja 1: Lista najpopularniejszych haseł [2].....4
Ilustracja 2: Przebieg uwierzytelniania za pomocą identyfikatora OpenID [11].....9
Ilustracja 3: Nawiązywanie połączenia klient-serwer poprzez SSL [23].....11
Ilustracja 4: Wygląd aplikacji: lista zarejestrowanych na serwerze aplikacji klienckich.....12
Ilustracja 5: Diagram komponentów.....13
Ilustracja 6: Diagram struktury.....14
Ilustracja 7: Diagram relacji pomiędzy elementami systemu.....15
Ilustracja 8: Ogólny schemat uwierzytelniania i autoryzacji - diagram sekwencji.....16
Ilustracja 9: Przebieg autoryzacji oAuth [12].....17
Ilustracja 10: Przykładowy e-mail - nowe hasło.....20
Ilustracja 11: Wygląd aplikacji: tworzenie konta użytkownika.....20
Ilustracja 12: Wygląd aplikacji: aktywacja konta użytkownika.....21
Ilustracja 13: Wygląd aplikacji: usuwanie konta użytkownika.....21
Ilustracja 14: Wygląd aplikacji: formularz logowania.....22
Ilustracja 15: Wygląd aplikacji: formularz logowania - komunikat o braku użytkownika.....22
Ilustracja 16: Wygląd aplikacji: formularz logowania - komunikat o błędzie logowania.....23
Ilustracja 17: Wygląd aplikacji: formularz logowania - komunikat o blokadzie konta.....23
Ilustracja 18: Wygląd aplikacji: resetowanie hasła konta użytkownika.....24
Ilustracja 19: Wygląd aplikacji: modyfikacja hasła konta użytkownika.....24
Ilustracja 20: Wygląd aplikacji: panel zarządzania kontem użytkownika.....25
Ilustracja 21: Wygląd aplikacji: formularz autoryzowania usługi (aplikacji klienckiej).....25
Ilustracja 22: Wygląd aplikacji: lista uwierzytelnionych sesji użytkownika.....26
Ilustracja 23: Wygląd aplikacji: lista autoryzowanych usług (aplikacji) użytkownika.....26
Ilustracja 24: Wygląd aplikacji: historia konta użytkownika.....27

Bibliografia

1: *Lista najpopularniejszych haseł - warto ich unikać*, http://nt.interia.pl/internet/wiadomosci/news/lista-najpopularniejszych-hasel-warto-ich-unikac,1724194,62 (dostęp 29 listopada 2011)

2: *The top 50 passwords you should never use*, http://nakedsecurity.sophos.com/2010/12/15/the-top-50-passwords-you-should-never-use/ (dostęp 30 listopada 2011)

3: *Fabryka haseł*, http://fabrykahasel.pl (dostęp 29 listopada 2011)

4: *How secure is my password*, http://howsecureismypassword.net (dostęp 29 listopada 2011)

5: Alan Freedman, *Encyklopedia komputerów*, 2004, s. 398

6: *Tech-FAQ*, http://www.tech-faq.com/aaa.html (dostęp 29 listopada 2011)

7: Alan Freedman, *Encyklopedia komputerów*, 2004, s. 887

8: *Niebezpiecznik.pl - Google wprowadza dwuskładnikowe uwierzytelnienie*, http://niebezpiecznik.pl/post/google-wprowadza-dwuskladnikowe-uwierzytelnienie/ (dostęp 28 grudnia 2011)

9: *Niebezpiecznik.pl - Facebook wprowadza nowe zabezpieczenia*, http://niebezpiecznik.pl/post/facebook-wprowadza-nowe-zabezpieczenia/ (dostęp 28 grudnia 2011)

10: *What is OpenID?*, http://openid.net/get-an-openid/what-is-openid/ (dostęp 28 grudnia 2011)

11: redakcja, *Zarządzanie tożsamością elektroniczną*, NEXT, 2008, nr 3, s. 100-103

12: *Introduction - oAuth*, http://oauth.net/about/ (dostęp 28 grudnia 2011)

13: Alan Freedman, *Encyklopedia komputerów*, 2004, s. 272

14: RFC4086 *Randomness Requirements for Security*, Czerwiec 2005 (http://tools.ietf.org/html/rfc4086)

15: Krzysztof Daszkiewicz, *Złam wszystkie hasła*, PC World, 2010, nr 1, s. 112-115

16: redakcja, *Chroń dostęp: Jak tworzyć bezpieczne hasła*, PC Format, 2009, nr 11, s. 54-55

17: Jeży Majdaniec, *10 najważniejszych faktów o hasłach*, CHIP, 2011, nr 3, s. 55-57

18: redakcja, *Pewne hasła*, PC Format, 2009, nr 4, s. 82-83

19: redakcja, *Pamięć do haseł*, NEXT, 2008, nr 12, s. 102-105

20: Alan Freedman, *Encyklopedia komputerów*, 2004, s. 415

21: ITI PK, *Laboratorium Sieci Komputerowych: LDAP - usługa katalogowa*, 19 listopada 2004

22: *Lightweight Directory Access Protocol*, http://pl.wikipedia.org/wiki/LDAP (dostęp 21 grudnia 2011)

23: Robert Tomaszewski, *Protokół HTTPS w Apache*, HAKIN9, 2008, nr 11, s. 48-54

24: *TIOBE Programming Community Index for December 2011*,
http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html (dostęp 22 grudnia 2011)
25: *PHP*, http://pl.wikipedia.org/wiki/PHP (dostęp 22 grudnia 2011)
26: *Nasza Kohana*, http://nasza.kohanaphp.pl (dostęp 21 grudnia 2011)