|  |
| --- |
| **Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie** |
| Opracowanie aplikacji internetowej umożliwiającej interaktywną naukę języka angielskiego |
|  |
| Praca inżynierska |
| Jakub Jaśkowiec |
| Promotor: dr inż. Mirosław Gajer |

Kraków, 2012

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki

KATEDRA AUTOMATYKI

Spis treści

[1. Wstęp 3](#_Toc313056967)

[1.1. Cel pracy 3](#_Toc313056968)

[1.2. Zawartość pracy 3](#_Toc313056971)

[2. Część teoretyczna 5](#_Toc313056972)

[2.1. Podstawowe pojęcia używane w pracy 5](#_Toc313056973)

[2.2. Technologie użyte w projekcie 5](#_Toc313056977)

[2.2.1. Język programowania - Ruby 5](#_Toc313056978)

[2.2.2. Ruby on Rails 7](#_Toc313056984)

[2.2.3. Gemy 8](#_Toc313056985)

[2.3. Baza danych 11](#_Toc313056990)

[2.4. Wykorzystane API 11](#_Toc313056991)

[2.5. Diagramy UML 12](#_Toc313056992)

[2.5.1. Diagramy Use Case 12](#_Toc313056993)

[2.5.2. Diagramy klas 12](#_Toc313057002)

[2.6. Innowacyjność procesu nauczania 13](#_Toc313057003)

[3. Część praktyczna 14](#_Toc313057004)

[3.1. Widok startowy 14](#_Toc313057005)

[3.2. Tworzenie nowego słowa 14](#_Toc313057006)

[3.2. Zarządzanie kolekcjami 17](#_Toc313057011)

[3.3. Prezentacja kolekcji 18](#_Toc313057012)

[4. Podsumowanie 19](#_Toc313057016)

[Bibliografia 19](#_Toc313057017)

# Wstęp

## Cel pracy

Niniejsza praca ma na celu zaprezentowanie aplikacji internetowej umożliwiającej interaktywną naukę języka angielskiego oraz szczegółowe omówienie technologii, w jakiej została ona wykonana.

Głównym założeniem programu jest umożliwienie użytkownikowi uczenia się słów oraz zwrotów z języka angielskiego. Nauka powinna przebiegać w taki sposób, aby wymagała jak najmniejszego wysiłku, jednocześnie umożliwiając jak najlepsze zapamiętywanie prezentowanych treści.

Głównymi funkcjonalnościami aplikacji są:

1. Umożliwienie użytkownikowi tworzenia własnych słów wraz z ich szczegółowym opisem
2. Podział tworzonych słów na kolekcje tematyczne
3. Prezentacja stworzonych kolekcji w taki sposób, aby uczący mógł zapamiętać jak najwięcej słów

Pomysł na aplikację narodził się podczas aktywności na jednym z forów języka angielskiego. Umieszczone przeze mnie autorskie prezentacje wywołały pozytywny odzew wśród użytkowników. Aplikacja bazuje na wykorzystanym wtedy pomyśle, automatyzując proces tworzenia treści oraz wprowadzając nowe możliwości.



## Zawartość pracy

Pierwsza część pracy to opis oraz analiza technologii użytej w projekcie. Zostanie dokonana analiza technologii Ruby on Rails, przyczyny jej użycia, zalety oraz wady. Przedstawione zostaną także inne technologie użyte w projekcie, używane gemy (pluginy) oraz zewnętrzne API.

Druga część to prezentacja aplikacji. Przedstawione zostaną funkcjonalności oraz scenariusze użycia programu, udokumentowane obrazami w formie zrzutów ekranu. Pokazana zostanie przydatność aplikacji w procesie uczenia się nowych słów oraz innowacyjność wykorzystanych w niej metod.

# Część teoretyczna

## Podstawowe pojęcia używane w pracy

Aplikacja, program, projekt – odnosi się tutaj do aplikacji internetowej, która została stworzona na potrzeby tej pracy

Słowo, wyrażenie – wyraz lub kilka wyrazów, którymi użytkownik może zarządzać przy pomocy aplikacji



## Technologie użyte w projekcie

### Język programowania - Ruby

**Ruby** – jest zorientowanym obiektowo interpretowanym językiem skryptowym stworzonym w 1995 w Japonii przez Yukihiro "Matza" Matsumoto.

Kod źródłowy Rubiego jest kompilowany przez interpreter w momencie wykonania programu. Wiąże się z tym szereg wad i zalet.

Wadą tego typu rozwiązania jest szybkość – ponieważ kod źródłowy jest interpretowany w momencie wykonania oznacza to, że działa wolniej niż odpowiadający mu kod skompilowanej aplikacji. Inną możliwą wadą jest to, że każdy użytkownik używający aplikacji może zobaczyć jej kod źródłowy. Nie jest to problem dla projektów typu Open Source, ale dla niektórych aplikacji komercyjnych może to być efekt niepożądany.

Zaletą języków interpretowanych jest możliwość przenoszenia ich pomiędzy systemami operacyjnymi oraz różnymi architekturami. Skompilowana aplikacja będzie działać jedynie na systemie operacyjnym oraz architekturze na którą została skompilowana. Do uruchomienia aplikacji napisanej w języku Ruby wystarczy zainstalowany interpreter oraz kod źródłowy aplikacji.

Kolejną zaletą jest możliwość pisania oraz wykonywania kodu w czasie rzeczywistym przez interpreter. Jest to ważna funkcjonalność dla wielu programistów.

Inne istotne cechy języka:

Ruby jest:

- zorientowany obiektowo

wszystko jest obiektem (włączając typy prymitywne, takie jak liczby całkowite, nil-wartości)

- dynamicznie typowany

sprawdzanie zgodności typów odbywa się w czasie wykonywania programu

Ruby posiada:

- duck typing

rozpoznawanie obiektu odbywa się nie na podstawie deklaracji typu, lecz poprzez badanie metod udostępnionych przez obiekt

- garbage collector (odśmiecacz pamięci)

zarządzanie dynamiczną pamięcią wykonywane jest automatycznie, programista nie musi zwalniać przydzielonej pamięci

- domknięcia (closures)

elementy kodu mogą być traktowane jako obiekty

- obsługę wyjątków

obsługa sytuacji wyjątkowych możliwa jest poprzez mechanizm obsługi wyjątków

- system pluginów

dzięki systemowi zarządzania pakietami RubyGems, ułatwione jest zarządzanie zewnętrznymi pluginami

- Interactive Ruby Shell

powłoka, dzięki której możliwe jest programowanie w linii poleceń

- implementację na wszystkich popularnych platformach

Ruby działa m.in. na systemach Windows, Linux oraz Mac.

Aplikacja jest stworzona w języku Ruby w wersji 1.9.2. Jest kompatybilna z wersją Ruby 1.8.7.



### Ruby on Rails

**Ruby on Rails** – jest frameworkiem do tworzenia aplikacji internetowych w języku Ruby. Został stworzony w 2004 roku przez Davida Heinemeier Hanssona. Udostępniony jest na zasadzie licencji Open Source.

Ruby on Rails korzysta z wzorca projektowego Model-View-Controller. Dzięki temu struktura aplikacji jest ściśle zdefiniowana, pozwala to na oddzielenie logiki aplikacji od interfejsu użytkownika. Za poszczególne moduły odpowiadają:

Model – ActiveRecord

Moduł odpowiedzialny za logikę aplikacji oraz komunikację z bazą danych.

Komunikacja z bazą danych jest standardowo realizowana przy pomocy systemu ORM, którym jest ActiveRecord. Framework umożliwia użycie innych systemów ORM.

Kod odpowiedzialny za logikę aplikacji umieszczony jest w plikach reprezentujących obiekty bazodanowe.

Kontroller – ActionController

Kontrolery odpowiedzialne są za sterowanie aplikacją i przepływ danych pomiędzy modelami a widokami

Widok – ActionView

Widoki tworzą interfejs użytkownika generowany przy użyciu danych z kontrolera oraz kodu HTML zapisanego w plikach widoków.

Framework dostarcza narzędzia, które ułatwiają programiście podstawowe czynności związane z tworzeniem aplikacji webowych. Jednym z takich narzędzi jest scaffolding, który pozwala na automatyczne generowanie kodu umożliwiającego uruchomienie podstawowej aplikacji.

Ruby on Rails posiada zintegrowane środowisko do testowania aplikacji. Domyślnym modułem testującym jest Test/Unit, jednak ze względu na rosnącą popularność narzędzia RSpec (używanego m.in. do Test Driven Development), został on użyty w tej aplikacji.

Do uruchomienia aplikacji wymagany jest odpowiedni serwer. W środowiskach deweloperskich dominuje serwer WEBrick oraz Mongrel, natomiast w środowisku produkcyjnym najpopularniejszym rozwiązaniem jest użycie modułu Phusion Passenger dla serwera Apache lub nginx.

Aplikacja jest stworzona w środowisku Ruby on Rails w wersji 3.0.3.

### Gemy

**Gem** – to spakowana aplikacja lub biblioteka w języku Ruby. Posiada nazwę (np. rake) oraz wersję (np. 0.4.16).

Zarządzanie gemami na komputerze odbywa się przy pomocy polecenia *gem*. Umożliwia ona między innymi instalowanie, usuwanie oraz wyszukiwanie dostępnych pakietów.

RubyGems to system zarządzania gemami.

Cechy RubyGems:

- łatwa instalacja oraz usuwanie pakietów i ich zależności

- zarządzanie i kontrola lokalnymi pakietami

- zarządzanie zależnościami

- wyszukiwanie oraz przeglądanie lokalnych i zdalnych pakietów

- możliwość instalacji wielu wersji tych samych pakietów

- interfejs webowy umożliwiający przeglądanie dokumentacji dotyczącej zainstalowanych pakietów

Zalety RubyGems:

- łatwość pobierania oraz instalacji bibliotek języka Ruby

- łatwe zarządzanie zależnościami

- usuwanie bibliotek, które nie są już w użyciu

W projekcie wykorzystano szereg gemów, dzięki którym możliwe było rozszerzenie funkcjonalności aplikacji oraz ułatwienie pracy programistycznej:

- Rspec

Jest popularnym narzędziem używanym do Test Driven Development w języku Ruby. Umożliwia tworzenie m.in. testów jednostkowych, funkcjonalnych oraz wydajnościowych.

W niniejszej pracy został użyty zamiast standardowego narzędzia testującego wbudowanego w Ruby on Rails (Test Unit). Zostały w nim stworzone głównie testy funkcjonalne.

Fragment testu sprawdzającego poprawność akcji odpowiedzialnej za wyświetlenie listy słów:

*describe "GET index" do*

*it "assigns all expressions as @records" do*

*expression = Expression.create! valid\_attributes*

*get :index*

*assigns(:records).should eq([expression])*

*end*

*end*

- Haml

Haml (HTML Abstraction Markup Language) jest językiem znaczników używanym do prostego i przejrzystego opisywania HTML. Został stworzony, aby obejść wiele niedoskonałości tradycyjnych systemów szablonów używając zarazem eleganckiej składni. Haml umożliwia zastąpienie standardowego systemu szablonów używanych w Ruby on Rails (RHTML).

Główne założenia języka:

- wcięcie reprezentuje zagnieżdżenie (brak tagów zamykających)

- tagi HTML są reprezentowane jako ‘%’ oraz nazwa tagu (np. %div )

- atrybuty HTML zapisywane są w postaci hasha z języka Ruby (np. {:class => ‘klasa’} )

Porównanie składni standardowego systemu szablonów RHTML z HAML:

|  |  |
| --- | --- |
| RHTML | HAML |
| *<div id="profile">*  *<div class="left column">*  *<div id="date"><%= print\_date %></div>*  *<div id="address">*  *<%= current\_user.address %>*  *</div>*  *</div>*  *<div class="right column">*  *<div id="email">*  *<%= current\_user.email %></div>*  *<div id="bio"><%= current\_user.bio %></div>*  *</div>*  *</div>* | *#profile*  *.left.column*  *#date= print\_date*  *#address= current\_user.address*  *.right.column*  *#email= current\_user.email*  *#bio= current\_user.bio* |

- Active Scaffold

<http://activescaffold.com/>

- Wordnik

Jest to oficjalny gem portalu wordnik.com. Umożliwia dostęp do Wordnik API, które jest wykorzystywane do pobierania danych do opisu słowa. Strona internetowa: <https://github.com/wordnik/wordnik-ruby>

-Paperclip

Wspiera obsługę plików graficznych. Umożliwia zapisywanie na serwerze, skalowanie oraz odczytywanie plików graficznych wybranych przez użytkowników.

- Devise

(logowanie opcjonalne)

- Coffee-rails

- Sass

- Mysql

Umożliwia używanie bazy danych MySql w Ruby on Rails. Opis bazy danych został przedstawiony w dalszej części pracy.

Javascript – Jquery - Coffeescript

JS, LIVE()

Interfejs został stworzony w języku angielskim. Dzięki temu używanie programu jest możliwe nie tylko przez ludzi polskojęzycznych, ale także innych obcokrajowców pragnących skorzystać z aplikacji w celu nauki języka angielskiego.

Przyczyny użycia Ruby on Rails:

- nowoczesny framework

- dobrze nadaje się do tego typu aplikacji

- duże wsparcie ze strony społeczności



## Baza danych

Mysql, co jest przechowywane

## Wykorzystane API

Podczas procesu tworzenia nowego słowa, aplikacja wykorzystuje dwa zewnętrzne API:

- Google Image Search API

Wykorzystywane jest w celu pobrania obrazów reprezentujących słowo.

Strona główna projektu: <http://code.google.com/apis/imagesearch/>

- Wordnik API

API serwisu internetowego Wordnik.com umożliwia pobieranie definicji, przykładów użycia oraz synonimów dla danego słowa.

Używanie API umożliwia gem „wordnik”, który jest używany w akcji „load data” kontrolera „expressions\_controller”.

Przykładowe użycie (pobieranie definicji):

@definitions = Wordnik.word.get\_definitions(@query, :useCanonical => 'true').map { |d| d["text"]}

## Diagramy UML

### Diagramy Use Case

Visio



### Diagramy klas

## Innowacyjność procesu nauczania

Dlaczego aplikacja jest innowacyjna do nauki:

- podczas opis słowa stara się korzystać z różnych technik zapamiętywania:

zapamiętywanie wzrokowe – obrazek

zapamiętywanie przy pomocy słuchu – plik z wymową

budowanie sieci skojarzeń poprzez synonimy

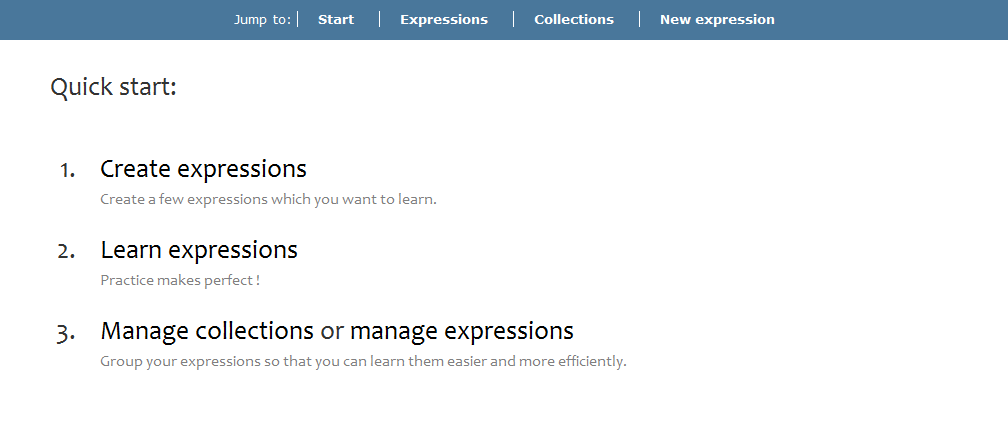
zapamiętywanie kontekstu w jakim słowo zostało użyte – przykłady użycia

definicja - wytłumaczenie

- podczas przeglądania kolekcji (TODO)

# Część praktyczna

## Widok startowy



Rysunek

## Tworzenie nowego słowa

Proces tworzenia nowego słowa został zaprojektowany tak, aby użytkownik na podstawie sugerowanych danych mógł samodzielnie opisać interesujące go słowo. Umożliwia to lepsze zapamiętywanie nowego słownictwa, gdyż pobudza kreatywność oraz pozwala stworzyć własne skojarzenia.

Na opis słowa składają się następujące elementy:

- nazwa,

- obraz,

- definicja,

- przykłady użycia,

- synonimy,

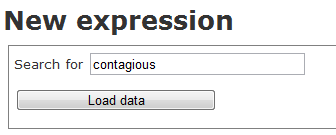
- notatki.

Dodatkowo, każde słowo można umieścić w kolekcji.

Poniżej zostanie omówiony proces tworzenia nowego wyrażenia, na przykładzie słowa ‘contagious’.

- w polu ‘Search for’ wpisujemy słowo / frazę

- klikamy ‘Load data’



Rysunek 2

Wyniki wyszukiwania zostają załadowane do ramki poniżej. Poszczególne pola zostały oddzielone poziomą linią.

Poniżej zaprezentowany zostanie stan poszczególnych pól po załadowaniu wyników.

- Nazwa (Name)

Nazwa, pod jaką słowo zostanie zapisane w systemie. Domyślnie wypełniane jest wyrażeniem wpisanym do pola „Search for”. Po wielokrotnym skorzystaniu z akcji „Load data” wyświetlana jest historia wyszukiwania.

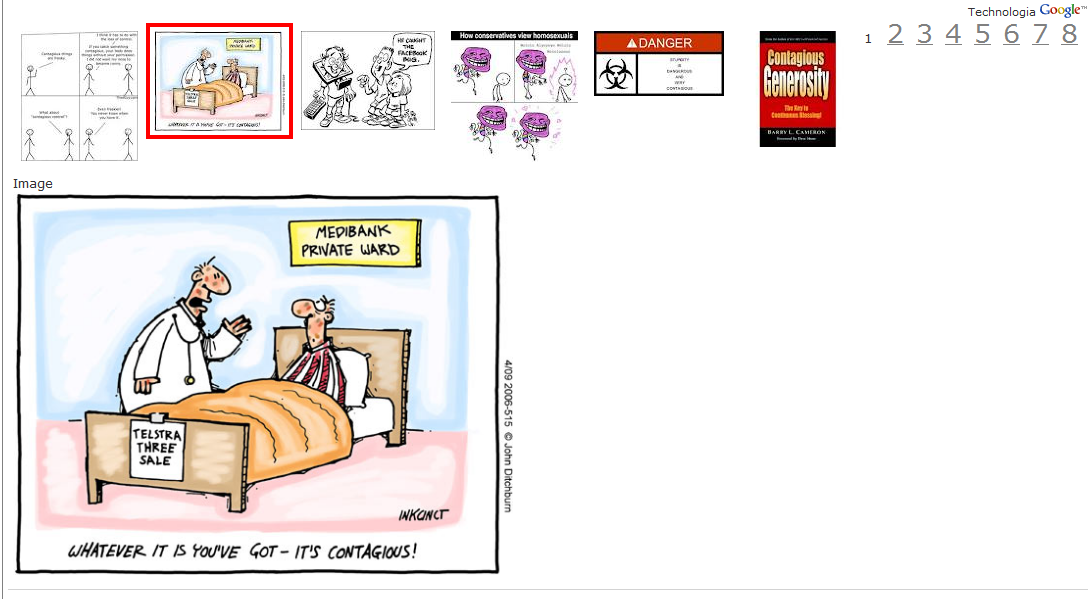
- Obraz (Image)

Obraz, który jest dołączany do słowa.

Obrazy pobierane są z wykorzystaniem technologii Google, przy użyciu Google Images API.

Po załadowaniu danych ukazują się miniatury. Po kliknięciu w miniaturę, zostaje ona otoczona czerwoną ramką, a poniżej zostaje załadowany obraz w rozmiarach, w jakich może być zapisany w systemie. Po prawej widzimy szare cyfry, dzięki którym możemy załadować więcej miniatur.

W ten sposób użytkownik z listy miniatur wybiera obraz, który reprezentuje słowo.

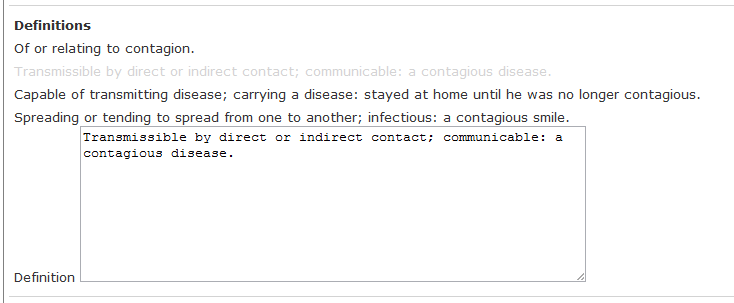


Rysunek 3

- Definicje (Definitions)

Dostępna jest lista definicji do wyboru. Po kliknięciu w definicję, zostaje ona wpisana do pola tekstowego poniżej i oznaczona kolorem szarym.

W systemie jako definicja zostanie zapisana treść pola tekstowego. Użytkownik może dowolnie wybierać definicje z listy: może wybrać jedną lub wiele definicji, może także wpisać swoje własne, jeśli żadna z proponowanych nie jest satysfakcjonująca.



Rysunek 4

- Przykłady użycia (Examples) / Synonimy (Synonyms)

Ich uzycie jest analogiczne do pola „definicja”

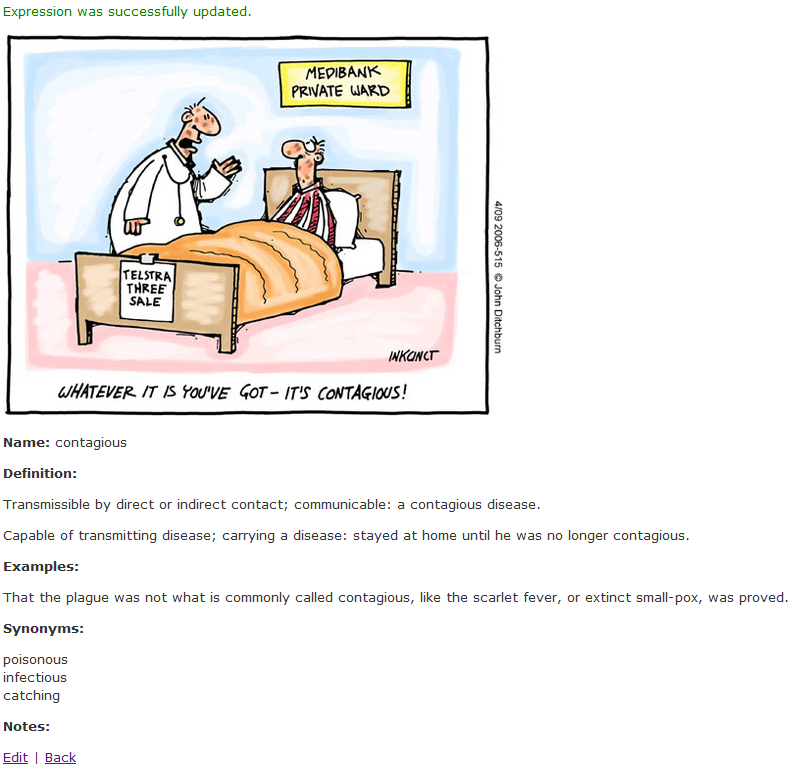
- Kolekcja (Collection)

Umożliwia dodanie słowa do wybranej kolekcji.

- Notatki (Notes)

Umożliwia dodanie własnych notatek dla słowa.

Po zapisaniu danych, użytkownik zostaje przeniesiony na stronę prezentującą stworzone słowo. Widnieją na niej wszystkie dane, które zostały wybrane podczas procesu tworzenia słowa.



Rysunek



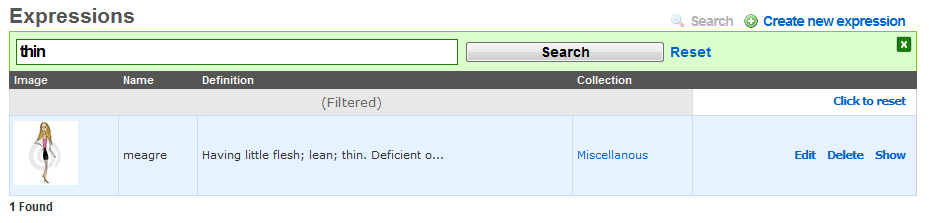
## Zarządzanie danymi



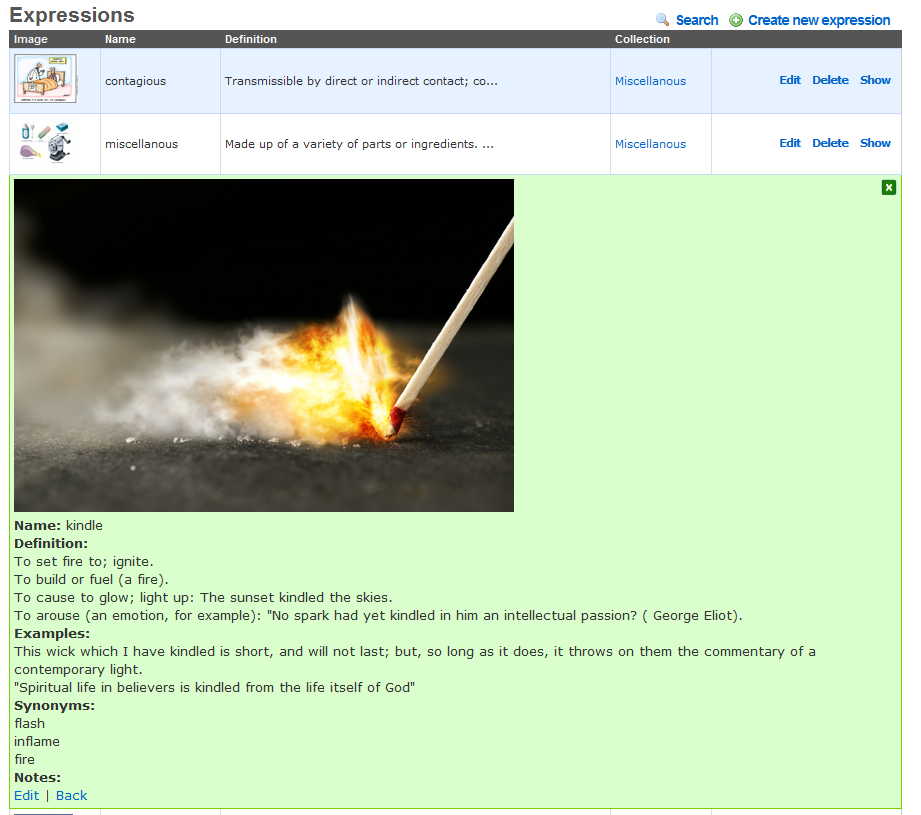
Rysunek



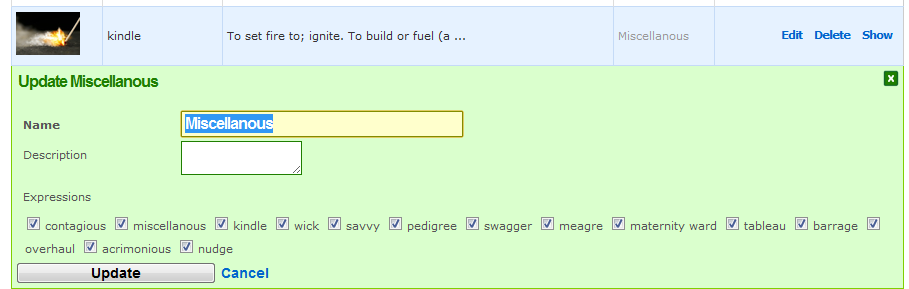
Rysunek



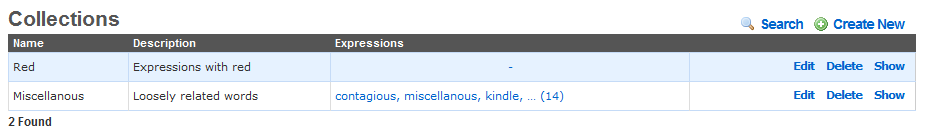
Rysunek



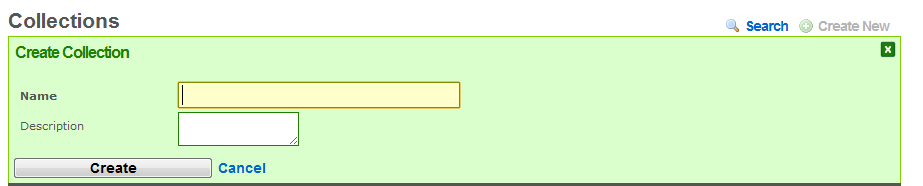
Rysunek



Rysunek



Rysunek



Rysunek



Rysunek

## Prezentacja kolekcji



# Podsumowanie

# Bibliografia

wikipedia.org

[http://docs.rubygems.org](http://docs.rubygems.org/)

<http://www.techotopia.com/index.php/What_is_Ruby%3F>

<http://rspec.info/>