

POLITECHNIKA ŚLĄSKA W GLIWICACH

WYDZIAŁ ELEKTRYCZNY

Katedra Energoelektroniki, Napędu Elektrycznego i Robotyki

**PROJEKT INŻYNIERSKI**

**Autentykacja witryny internetowej w oparciu o komponent framework’u Joomla!**

**Website authentication with Joomla! framework component**

Student: **Krzysztof Janusz CHADYNKA**

Nr albumu: 210370

Studia: Stacjonarne I stopnia

Kierunek: Informatyka

Prowadzący: dr inż. Maciej Sajkowski

Recenzent:

Spis treści

[1. Cel projektu 5](#_Toc409571778)

[2. Zakres projektu 6](#_Toc409571779)

[1. Analiza przedmiotu projektu 6](#_Toc409571780)

[2. Opracowanie algorytmu 7](#_Toc409571781)

[1. Proces autentykacji 7](#_Toc409571782)

[3. Opracowanie struktury bazy danych 8](#_Toc409571783)

[3. Założenia projektu 11](#_Toc409571784)

[1. Wykorzystywany język programowania 11](#_Toc409571785)

[2. Model MVC 11](#_Toc409571786)

[3. Wykorzystywane środowisko 11](#_Toc409571787)

[4. Wykorzystywany system bazy danych 11](#_Toc409571788)

[4. Dobór rozwiązań 12](#_Toc409571789)

[5. Część projektowa 13](#_Toc409571790)

[6. Implementacja 14](#_Toc409571791)

[1. Struktura plików i katalogów 14](#_Toc409571792)

[2. Ogólne informacje na temat kodu źródłowego 16](#_Toc409571793)

[3. Część administracyjna komponentu 16](#_Toc409571794)

[4. Część frontowa komponentu 16](#_Toc409571795)

[5. Proces autentykacji 16](#_Toc409571796)

[7. Dokumentacja użytkownika 17](#_Toc409571797)

[7.1. Wymagania sprzętowe 17](#_Toc409571798)

[7.2. Instalacja 17](#_Toc409571799)

[7.3. Zarządzanie aplikacją 20](#_Toc409571800)

[7.4 Część frontowa witryny 26](#_Toc409571801)

[8. Podsumowanie i wnioski 27](#_Toc409571802)

[9. Literatura 28](#_Toc409571803)

Spis ilustracji

[Rysunek 1 Schemat blokowy - proces autentykacji 8](#_Toc409571804)

[Rysunek 2 Struktura bazy danych 11](#_Toc409571805)

[Rysunek 3 Instalacja komponentu - logowanie do panelu administracyjnego 18](#_Toc409571806)

[Rysunek 4 Instalacja komponentu - przejście do odpowiedniej zakładki 19](#_Toc409571807)

[Rysunek 5 Instalacja komponentu - wybór pliku 19](#_Toc409571808)

[Rysunek 6 Instalacja komponentu - informacja o pozytywnym wyniku instalacji 20](#_Toc409571809)

[Rysunek 7 Przesłanie pliku controller.php na zdalny serwer 20](#_Toc409571810)

[Rysunek 8 Położenie komponentu w menu głównym - panel administracyjny 21](#_Toc409571811)

[Rysunek 9 Główny widok podstrony Users Management – panel administracyjny 22](#_Toc409571812)

[Rysunek 10 Główny widok podstrony Roles management – panel administracyjny 22](#_Toc409571813)

[Rysunek 11 Edycja istniejącej roli – panel administracyjny 23](#_Toc409571814)

[Rysunek 12 Dodawanie nowej roli – panel administracyjny 23](#_Toc409571815)

[Rysunek 13 Główny widok podstrony Categories assigned to roles - panel administracyjny 23](#_Toc409571816)

[Rysunek 14 Główny widok postrony Frontend settings - panel administracyjny 24](#_Toc409571817)

[Rysunek 15 Edycja treści w widoku szczegółów użytkownika - panel administracyjny 24](#_Toc409571818)

[Rysunek 16 Dostępne tłumaczenia - panel administracyjny 25](#_Toc409571819)

[Rysunek 17 Dodawanie nowego tłumaczenia - panel administracyjny 25](#_Toc409571820)

[Rysunek 18 Przykładowe tłumaczenie tesktu - panel administracyjny 26](#_Toc409571821)

[Rysunek 19 Dodanie nowej pozycji w menu - panel administracyjny 26](#_Toc409571822)

[Rysunek 20 Podstrona informacji o użytkowniku - część frontowa 27](#_Toc409571823)

[Rysunek 21 Informacja o nieudanej autentykacji - część frontowa 27](#_Toc409571824)

Spis tabel

[Tabela 1 Struktura początkowa tabeli joo\_roles\_info 10](#_Toc409603151)

[Tabela 2 Struktura początkowa tabeli joo\_roles\_role 10](#_Toc409603152)

[Tabela 3 Zestawienie podstron i ich funkcjonalności - panel administracyjny 26](#_Toc409603153)

Spis listing-ów kodu

[Listing 1 Ograniczenie dostępu do pliku 17](file:///G:\Studia\Semestr%20VII\Praca%20inzynierska\projekt\dokumentacja.docx#_Toc409603144)

[Listing 2 Funkcja geturrentCategoryID 18](file:///G:\Studia\Semestr%20VII\Praca%20inzynierska\projekt\dokumentacja.docx#_Toc409603145)

[Listing 3 Funkcja getCurrentRoleID 18](file:///G:\Studia\Semestr%20VII\Praca%20inzynierska\projekt\dokumentacja.docx#_Toc409603146)

[Listing 4 Funkcja checkIfCategoryExists 19](file:///G:\Studia\Semestr%20VII\Praca%20inzynierska\projekt\dokumentacja.docx#_Toc409603147)

[Listing 5 Weryfikacja autentykacji 20](file:///G:\Studia\Semestr%20VII\Praca%20inzynierska\projekt\dokumentacja.docx#_Toc409603148)

# 1. Cel projektu

Celem niniejszego projektu jest zaimplementowanie funkcjonalności pozwalającej administratorowi witryny internetowej na łatwe i przejrzyste zarządzanie rolami użytkowników, a co za tym idzie przydzielanie im odpowiednich uprawnień.

Mechanizm oparty zostanie na jednym z wielu dostępnych na rynku framework’ów – Joomla!. Jest to jeden z najstarszych szkieletów aplikacji internetowych opartym na modelu MVC.

Motywacją do podjęcia takiej tematyki była wykonywana praca zawodowa, opierająca się właśnie na tworzeniu stron internetowych i rozszerzeń dla nich. Do tej pory nie znalazłem dla framework’u Joomla! rozwiązania, które w łatwy i przystępny sposób umożliwiałoby przypisywanie dla użytkowników odpowiednich uprawnień przy korzystaniu z witryny.

Końcowym efektem realizacji projektu powinien być komponent pozwalający na dodawanie nowych, nazwanych przez administratora ról oraz przypisywanie ich odpowiednim użytkownikom. Dla każdej roli będą przypisane kategorie artykułów, do których dana grupa użytkowników będzie miała dostęp. Na podstawie przypisanej roli, przy każdorazowej próbie wyświetlenia artykułu przez danego użytkownika, następować będzie proces autentykacji, który zweryfikuje uprawnienia nadane użytkownikowi, a co za tym idzie zezwoli lub zabroni dostępu do treści.

# 2. Zakres projektu

## 2.1. Analiza przedmiotu projektu

**Na początku tego podrozdziału warto wspomnieć czym właściwie jest autentykacja. W odróżnieniu od autoryzacji, polega ona na nadawaniu odpowiednich ograniczeń lub przywilejów użytkownikom wyłącznie przez administratora systemu (w tym wypadku witryny internetowej) bez jakiejkolwiek interakcji z korzystającym z serwisu. Autoryzacja natomiast oznacza proces rejestracji użytkownika a następnie logowania na podstawie podanych danych, których weryfikacja następuje przy każdorazowej próbie dostępu do serwisu.**

Jak już zostało wspomniane w ramach opisywania przedmiotu projektu w rozdziale pierwszym, aplikacja, której procesu implementacji dotyczy niniejsza praca, zostanie oparta na framework’u Joomla. Będzie stanowić jej rozszerzenie – a konkretnie komponent.

Wybór tematyki i framework’u sugerowany był zapotrzebowaniem na tego typu narzędzie. Joomla! przewiduje co prawda funkcjonalność typów użytkowników, jednak dotyczą one jedynie uprawnień administrowania witryny od strony zaplecza.

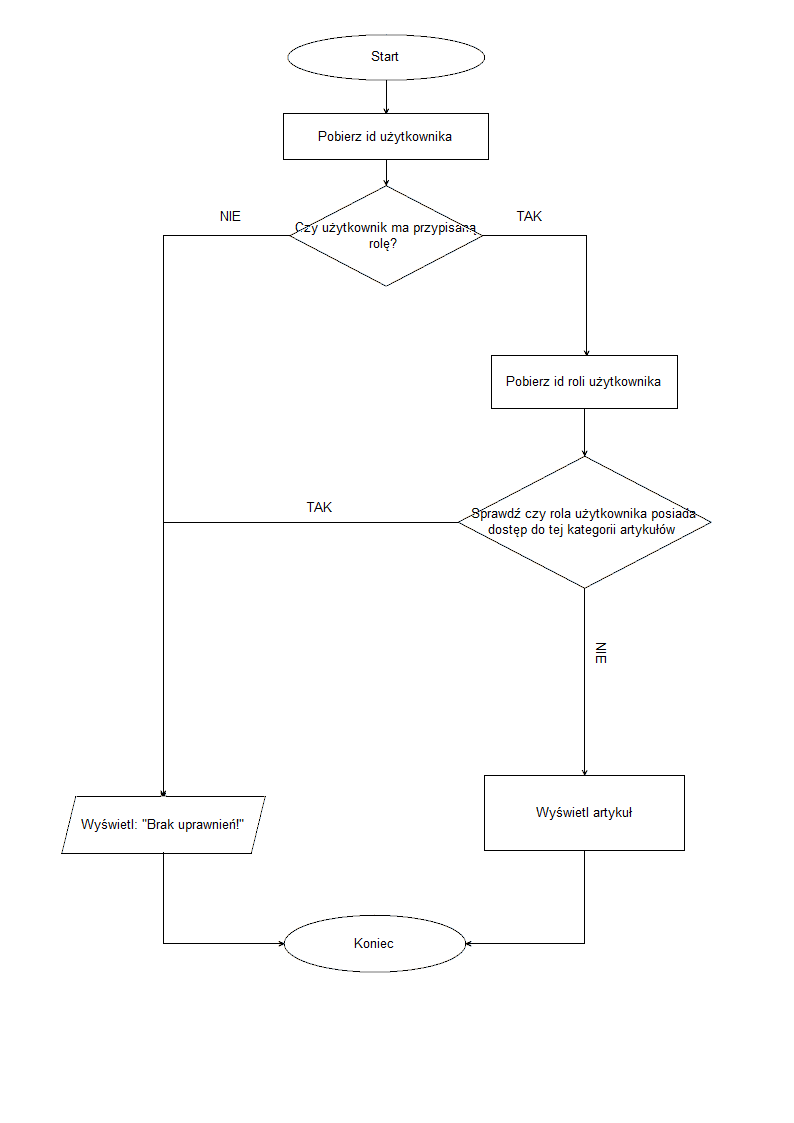
Istnieje także rozwiązanie pozwalające na umieszczenie załączników pobieranych na podstawia wykupionej wcześniej licencji.

Problem, który jest przedmiotem projektu, polega na zaimplementowaniu rozwiązania, które w łatwy i przystępny dla administratora systemu sposób pozwoli na bezpośrednie przypisanie ról, na podstawie których dany użytkownik otrzyma dostęp do kategorii artykułów.

Założeniem projektu jest jak najbardziej optymalna ze względu na zasoby implementacja kodu źródłowego, a także przystępność i prostota użytkowania gotowego komponentu dla użytkowników nie mających większego pojęcia o zagadnieniach informatycznych.

## 2.2. Opracowanie algorytmu

### 2.2.1. Proces autentykacji



Rysunek 1 Schemat blokowy - proces autentykacji

Jak wynika z powyższego schematu, proces autentykacji odbywa się następująco:

1. Na początku zostaje pobrane ID użytkownika obecnie zalogowanego w naszej witrynie.
2. Na podstawie pobranego ID przeszukujemy odpowiednią tabelę w bazie danych, aby sprawdzić czy zalogowanemu użytkownikowi przypisana jest jakaś rola.
3. Jeżeli wynik powyższego warunku okazał się negatywny, kończymy proces wyświetlając komunikat a nieudanej autentykacji. W przeciwnym wypadku pobieramy z tej samej tabeli ID roli przypisanej do użytkownika .
4. Przeszukujemy następnie tabelę przechowującą informacje o kategoriach dostępnych dla danej roli i sprawdzamy czy dla zalogowanego użytkownika możliwe jest przeglądanie artykułów z bieżącej kategorii.
5. Jeżeli wynik powyższego warunku okazał się pozytywny wyświetlamy artykuł i kończymy proces. W przeciwnym wypadku wyświetlamy komunikat o nieudanej autentykacji i kończymy proces

## 2.3. Opracowanie struktury bazy danych

Framework Joomla! dysponuje już gotową bazą danych, na której oparte jest działanie witryny internetowej. Do każdej nazwy tabeli dodawany jest, ustawiony przez administratora, lub programistę, prefix tabel (w przypadku mojej witryny jest to przedrostek *joo\_*).   
Ingerencja w bazę danych będzie ograniczała się jedynie do utworzenia czterech dodatkowych tabel:

* roles\_info(id, header, description, image, user\_info, role\_info, categories\_info)
* roles\_user (user\_id, role\_id)
* roles\_role (role\_id, name)
* roles\_categories (role\_id, category\_id)

Gdzie znaczenie poszczególnych tabel jest następujące:

1. **roles\_info (a właściwie joo\_roles\_info)**

Jest to tabela przechowująca dane, które zostaną wyświetlone w części frontowej komponentu, wyświetlającej informacje na temat użytkownika, przypisanej mu roli, oraz powiązanych z tą rolą kategorii.

Informacje w tabeli zapisane zostały w postaci aliasów, które podczas ładowania na stronę zostaną automatycznie przetłumaczone (proces tłumaczenia zostanie opisany w rozdziale implementacyjnym).

Znaczenie poszczególnych aliasów jest następujące:

* header – nagłówek strony
* description – krótki opis funkcji, jaką spełnia komponent
* user\_info – tekst poprzedzający dane identyfikujące użytkownika
* role\_info – tekst poprzedzający rolę użytkownika
* categories\_info – tekst poprzedzający listę kategorii dostępnych dla przypisanej użytkownikowi roli

Tabela już przy instalacji wypełniana jest przykładowymi aliasami:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **header** | **description** | **image** | **user\_info** | **role\_info** | **categories\_info** |
| COM\_  ROLES\_  INFO\_ HEADER | COM\_ ROLES\_ INFO\_ DESC |  | COM\_  ROLES\_  INFO\_  USERINFO | COM\_  ROLES\_  INFO\_  ROLE\_INFO | COM\_ROLES\_  INFO\_  CAT\_INFO |

Tabela 1 Struktura początkowa tabeli joo\_roles\_info

1. **roles\_user (a właściwie joo\_roles\_user)**

Jest to tabela przechowująca ID identyfikujące użytkownika (pobrane z tabeli joo\_users) oraz identyfikator roli mu przypisanej

1. **roles\_role (a właściwie joo\_roles\_role)**

Jest to tabela przechowująca dostępne role użytkowników wraz z ich identyfikatorami.

Przy instalacji tworzone są od razu trzy role użytkowników („student”, „pracownik” i „vip”), tak więc początkowa struktura tej tabeli przedstawia się następująco:

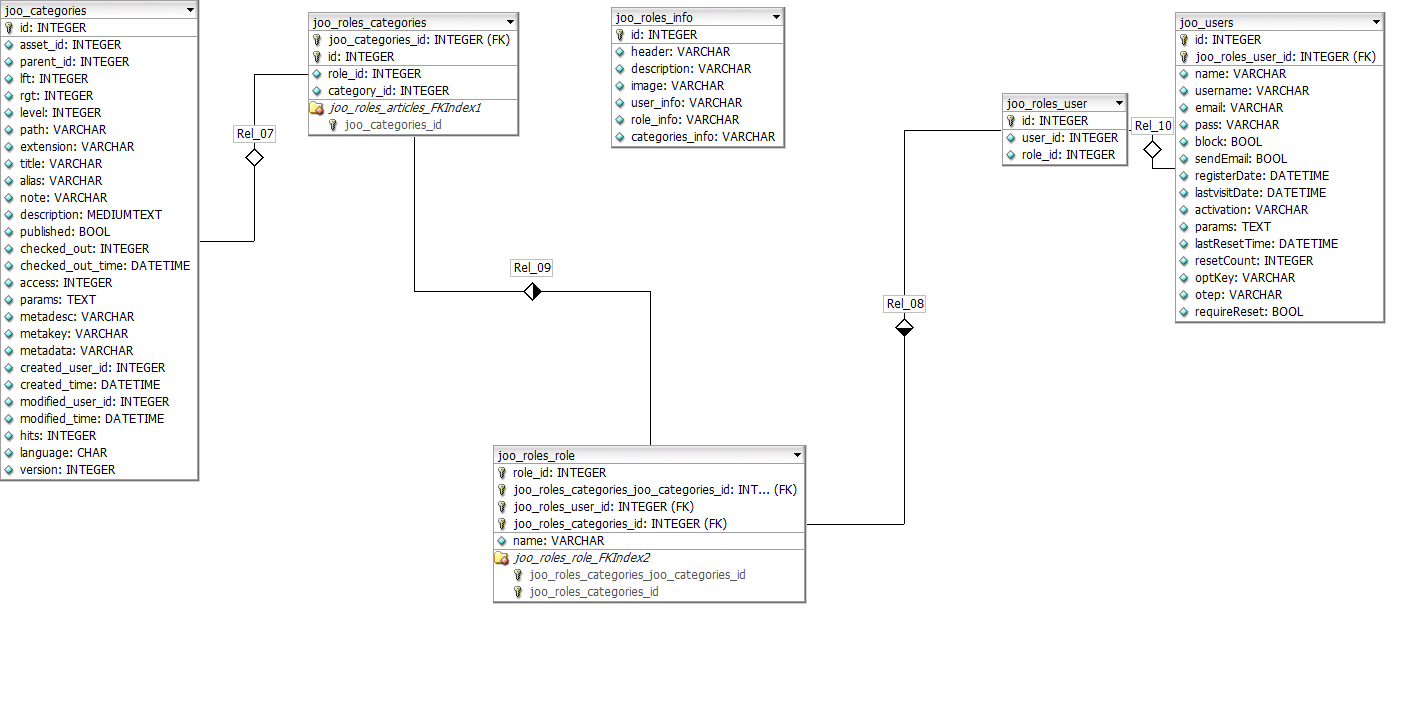
|  |  |
| --- | --- |
| **role\_id** | **Name** |
| 1 | Student |
| 2 | Pracownik |
| 3 | Vip |

Tabela 2 Struktura początkowa tabeli joo\_roles\_role

1. **roles\_articles (a właściwie joo\_roles\_articles)**

Jest to tabela przechowująca połączone w pary identyfikatory roli wraz z przypisanymi do nich identyfikatorami kategorii artykułów (pobranych z tabeli joo\_categories), na dostęp do których zezwala dana rola.

Ostatecznie struktura wykorzystywanych tabel bazy danych (wraz z odpowiednimi relacjami) przedstawia się następująco (*Rysunek 1*):



Rysunek 2 Struktura bazy danych

z czego tabele *joo\_categories* oraz *joo\_users* są tabelami predefiniowanymi framework’u. Na ich podstawie pobierane są informacje o kategoriach artykułów oraz o użytkownikach.

# 3. Założenia projektu

## 1. Wykorzystywany język programowania

Implementacja projektu zostanie oparta o język programowania PHP, gdyż w takim języku obsługiwane są skryptu framework’u Joomla!

## 2. Model MVC

## 3. Wykorzystywane środowisko

Jak już wspomniałem kilkakrotnie powyżej, projekt zostanie oparty na framework’u Joomla!.

Dla celów tworzenia projektu używać będę oprogramowania pozwalającego na utworzenie lokalnego serwera *WebServ*, a także oprogramowania *NetBeans*, w którym dokonam implementacji całego kodu projektu.

Jako systemu kontroli wersji projektu zdecydowałem się użyć repozytorium *GitHub.*

## 4. Wykorzystywany system bazy danych

Jako systemu bazy danych używać będę systemu MySQL, co także narzuca wykorzystywany framework. Jednak nawet gdyby nie było to niezbędne zdecydowałbym się także na system MySQL.

# 4. Dobór rozwiązań

W poprzednich rozdziałach zaznaczone zostało, iż projekt będzie stanowił implementację komponentu. W dalszej części tego rozdziału nastąpi wyjaśnienie dokonanego wyboru.

Joomla! obsługuje trzy podstawowe typy rozszerzeń:

* moduły,
* dodatki (wtyczki),
* komponenty.

Moduł jest najprostszym typem rozszerzenia. Stanowi niewielką funkcjonalność służącą najczęściej do reprezentacji danych (np. moduł reklamy), lub prostego pobierania danych – w tym wypadku najczęściej stanowią część składową komponentu (przykładowo moduł logowania). Moduły stanowią także reprezentacje menu na stronie – aby wyświetlić dodatkowe menu należy najpierw przypisać je do modułu.

Dodatek (wtyczka) stanowi pewną funkcjonalność wykonującą operację na gotowych elementach strony, świadczą usługi na rzecz komponentów i modułów. Są najbardziej wyspecjalizowanymi rozszerzeniami.

Komponent jest najbardziej złożonym rozszerzeniem – stanowią rodzaj niewielkich aplikacji, składających się z części administracyjnej i prezentacyjnej. Działanie całej witryna uruchomionej na silniku Joomla! opiera się na funkcjonowaniu różnych komponentów. Przykładem jest komponent służący do rejestracji użytkowników (*com\_registration*). W części prezentacyjnej istnieje możliwość zarejestrowania się zainteresowanego gościa witryny. Po stronie administratora możliwa jest edycja nowo utworzonego konta, a także dodanie użytkownika z tego poziomu.

Aplikacja opisywana w projekcie będzie stanowić dodatkową funkcjonalność składającą się z części administracyjnej i frontowej, co powoduje wybór komponentu, jako typu rozszerzenia na którym zostanie oparta.

# 5. Część projektowa

# 6. Implementacja

## 6.1. Struktura plików i katalogów

* **com\_roles**
  + **admin**
    - **controllers**
      * *role.php*
      * *roles.php*
      * *user.php*
      * *users.php*
    - **language**
      * **en-GB**
        + *en-GB.com\_roles.ini*
        + *en-GB.com\_roles.sys.ini*
    - **models**
      * **forms**
        + *role.xml*
        + *user.xml*
      * *categories.php*
      * *role.php*
      * *roles.php*
      * *user.php*
      * *users.php*
    - **sql**
      * **updates**
        + **mysql**

*0.0.1.sql*

*Index.html*

* + - * + *index.html*
      * *install.mysql.utf8.sql*
      * *uninstall.mysql.utf8.sql*
      * *index.html*
    - **tables**
      * *role.php*
      * *roles.php*
      * *index.html*
    - **views**
      * **categories**
        + **tmpl**

*default.php*

*default\_head.php*

*default\_body.php*

*default\_foot.php*

* + - * **role**
        + **tmpl**

*edit.php*

* + - * + *view.html.php*
      * **roles**
        + **tmpl**

*default.php*

*default\_head.php*

*default\_body.php*

*default\_foot.php*

* + - * + *view.html.php*
      * **user**
        + **tmpl**

*edit.php*

* + - * + *view.html.php*
      * **users**
        + **tmpl**

*default.php*

*default\_head.php*

*default\_body.php*

*default\_foot.php*

* + - * + *view.html.php*
    - *controller.php*
    - *roles.php*
    - *index.html*
  + **site**
    - **language**
      * **en-GB**
        + *en-GB.com\_roles.ini*
    - **views**
      * **roles**
        + **tmpl**

*default.php*

*index.html*

* + - * + *view.html.php*
        + *index.html*
      * *index.html*
    - *controller.php*
    - *roles.php*
    - *index.html*

## 6.2. Ogólne informacje na temat kodu źródłowego

Kazdy plik z kodem zródłowym (niezależnie czy jest to widok, model, kontroler, czy też klasa pomocnicza) rozpoczyna się nagłówkiem zabraniającym dostępu do niego osobom niepowołanym.

Listing Ograniczenie dostępu do pliku

*defined('\_JEXEC') or die('Restricted access');*

## 6.3. Część administracyjna komponentu

Część administracyjna komponentu odpowiada za pobieranie i edycję danych przetwarzanych podczas działania rozszerzenia.

Pierwszą, bardzo istotną kwestią za jaką odpowiada ta część komponentu jest instalacja. To właśnie w tej części deklarowane są początkowe zapytania do bazy danych (

## 6.4. Część frontowa komponentu

## 6.5. Proces autentykacji

Cały proces autentykacji odbywa się w pliku */administrator/components /com\_content/controller.php*, który na potrzeby projektu został zmodyfikowany. Poniżej zamieszczone zostały zmienione lub dodane na potrzeby autentykacji fragmenty kodu:

private function getCurrentCategoryID($art\_id)

{

$db = JFactory::getDbo();

$query = $db->getQuery(true);

$query->select('catid')->from('#\_\_content')->where('id='.$art\_id);

$db->setQuery($query);

return $db->loadResult();

}

Listing Funkcja geturrentCategoryID

Powyzsza funkcja odpowiada za pobranie ID bieżącej kategorii z bazy danych na podstawie, przekazanego jako argumentu funkcji, ID artykułu.

Listing Funkcja getCurrentRoleID

private function getCurrentRoleID($user\_id)

{

$db = JFactory::getDbo();

$query = $db->getQuery(true);

$query->select('role\_id')->from('#\_\_roles\_user')->where('user\_id='.$user\_id);

$db->setQuery($query);

return $db->loadResult();

}

Powyzsza funkcja odpowiada za pobranie z bazy danych ID roli przypisanej użytkownikowi na podstawie ID użytkownika przkazanego jako argument funkcji

private function checkIfCategoryExists($role\_id, $cat\_id)

{

$db = JFactory::getDbo();

$query = $db->getQuery(true);

$query->select('category\_id')->from('#\_\_roles\_categories')->where('role\_id='.$role\_id);

$db->setQuery($query);

$results = $db->loadObjectList();

foreach ($results as $result)

{

if ($result->category\_id == $cat\_id)

{

return true;

break;

}

}

return false;

}

Listing Funkcja checkIfCategoryExists

Powyzsza funkcja odpowiada za konkretny proces autentykacji. Jako argumenty pobierane są: ID roli użytkownika oraz ID kategorii. Następnie z tabeli *joo\_roles\_categories* pobierane są te wpisy, które posiadają ID roli równe przekazanemu. Następnie w pętli następuje sprawdzanie, czy w którymś z pobranych wierszy znajduje się przekazane ID kategorii. Jeżeli tak, funkcja zwraca wartość true.

Ostateczne sprawdzenie dostępu użytkownika do danej kategorii wygląda następująco:

Listing Weryfikacja autentykacji

if ($role\_id != 0 && $cat\_id != 0)

{

if ($this->checkIfCategoryExists($role\_id, $cat\_id))

parent::display($cachable, $safeurlparams);

else

JFactory::getApplication()-> enqueueMessage(JText::\_('COM\_ROLES\_ACCESS\_DENIED\_MESSAGE'), 'Error');

}

Na początku następuje sprawdzenie czy ID roli użytkownika oraz ID kategorii zostały określone. Jeżeli tak, to wywoływana jest opisana wyżej funkcję (Listing 4) i na podstawie zwróconej przez nią wartości udostępnia się lub blokuje daną kategorię.

# 7. Dokumentacja użytkownika

## 7.1. Wymagania sprzętowe

Aby móc korzystać z komponentu, serwer na którym będzie on zainstalowany musi spełniać następujące wymagania:

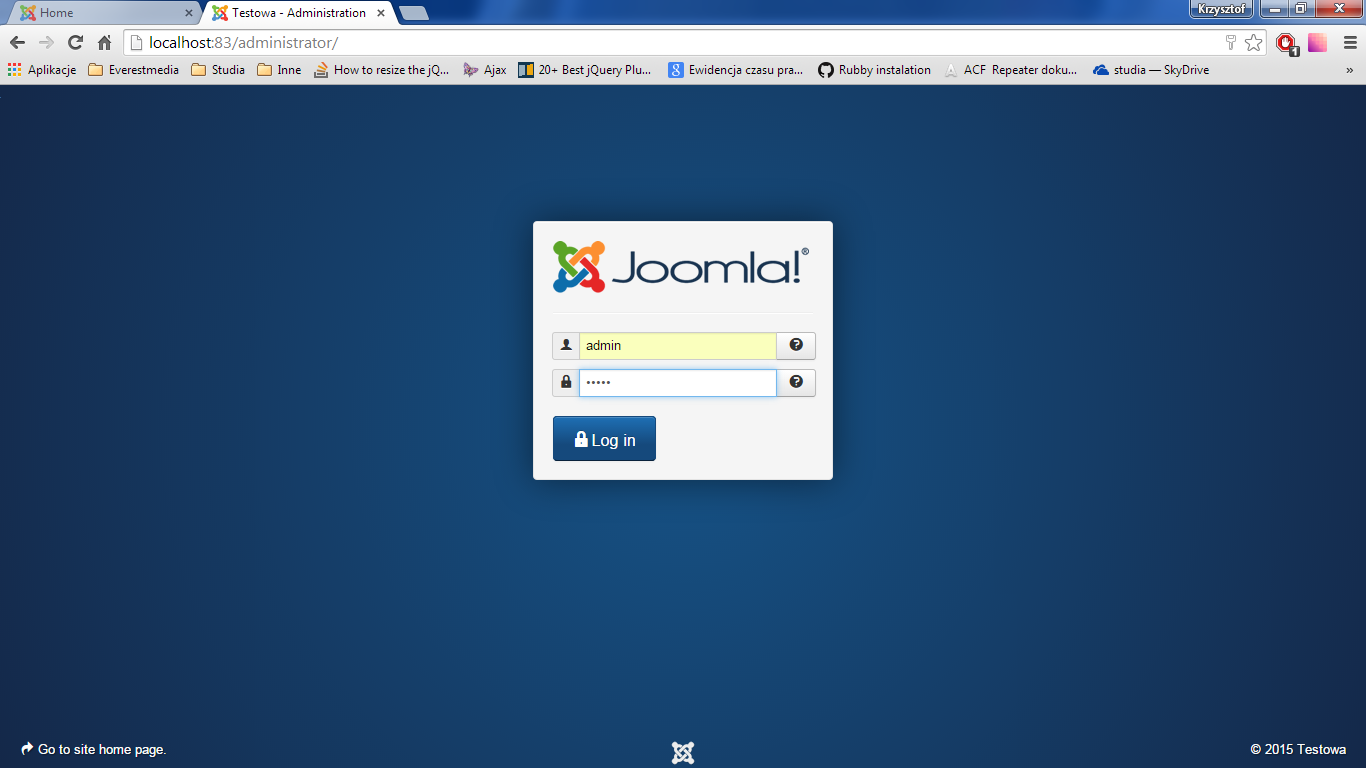
* obsługa PHP w wersji 5.3.10 lub wyższej
* Obsługa bazy danych w standardzie MySQL 5.1 (lub nowsza)
* Na serwerze musi być zainstalowana witryna internetowa oparta na framework’u Joomla! wersji 3.1 lub wyższej (Proces instalacji szczegółowo opisany jest na stronie producenta – [*www.wiki.joomla.pl*](http://www.wiki.joomla.pl)).

## 7.2. Instalacja

Po zainstalowaniu framework’u na serwerze przechodzi się do instalacji komponentu.

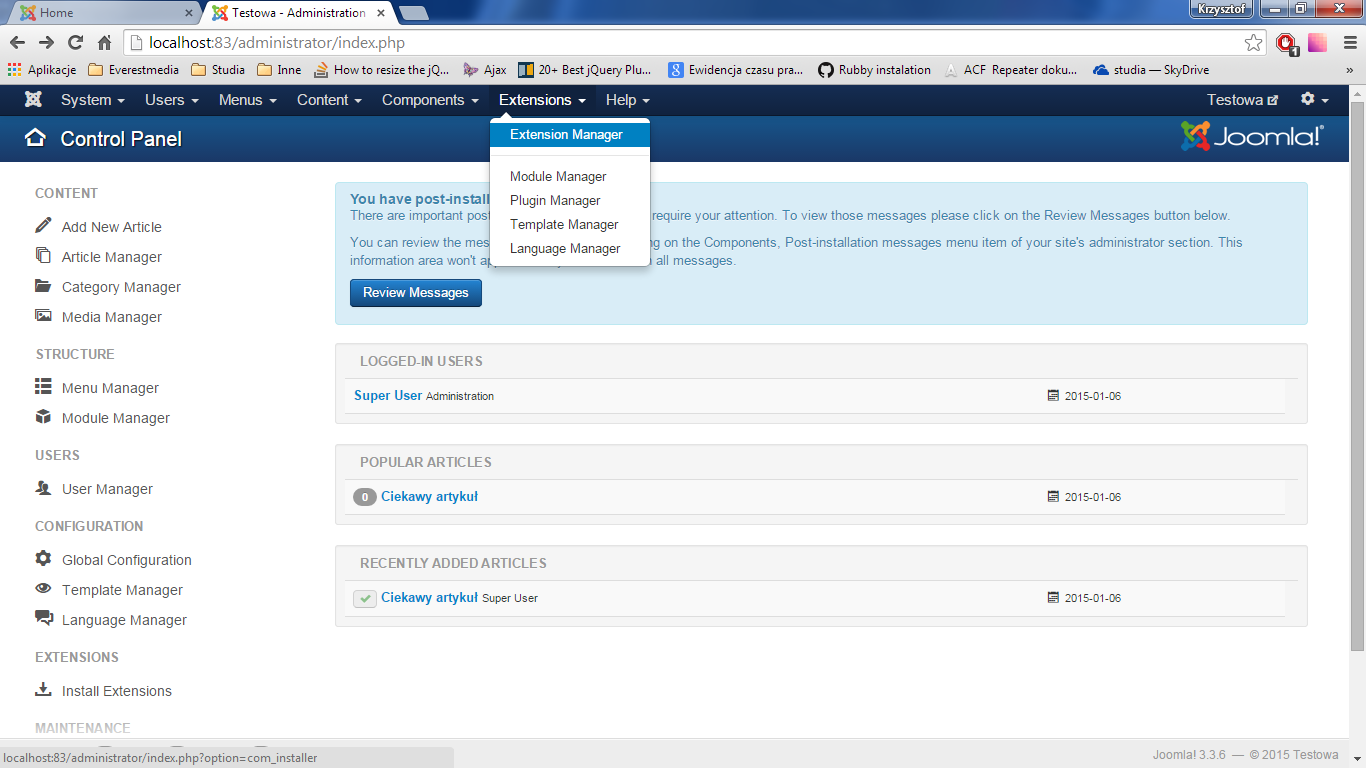
W tym celu należy:

1. zalogować się do panelu administracyjnego, korzystając z danych podanych podczas instalacji Joomla!. :



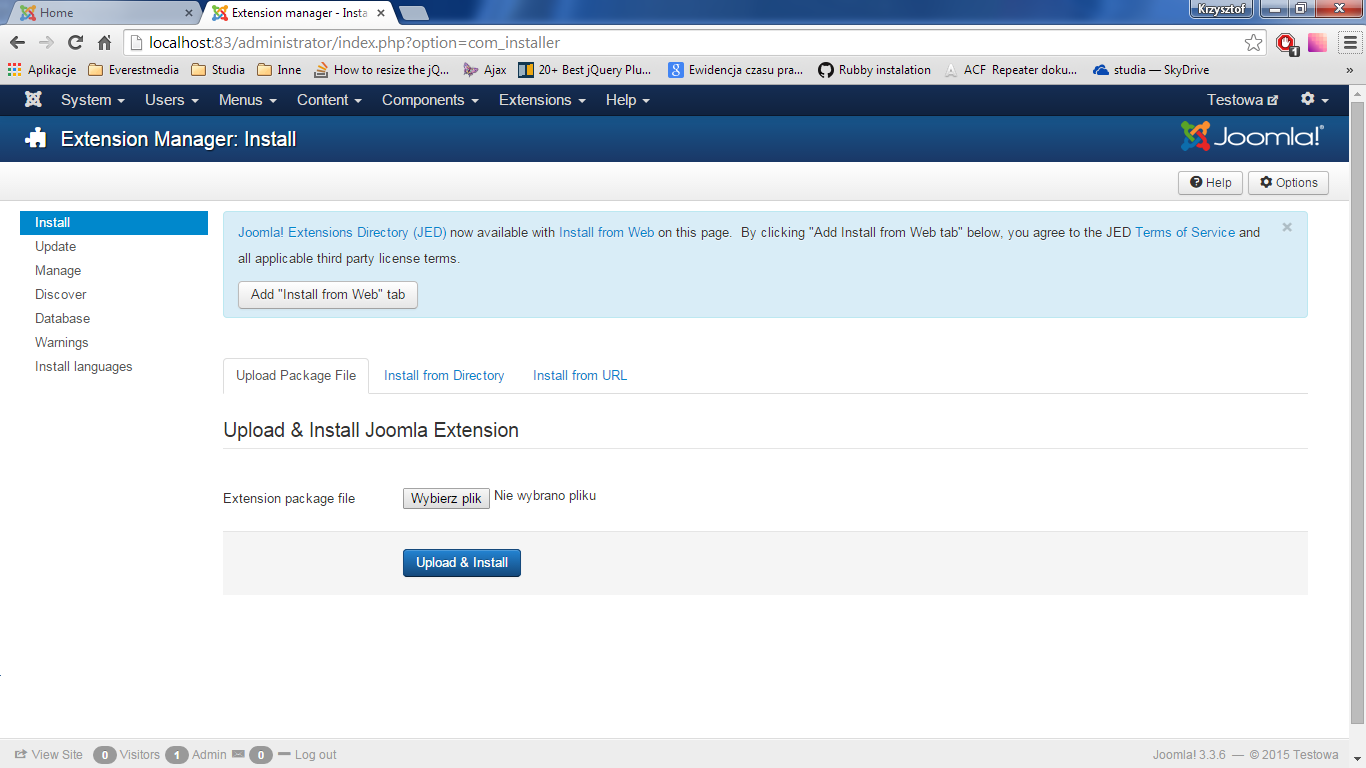
Rysunek 3 Instalacja komponentu - logowanie do panelu administracyjnego

1. Wybrać z górnego menu pozycję Extensions -> Extension Manager:



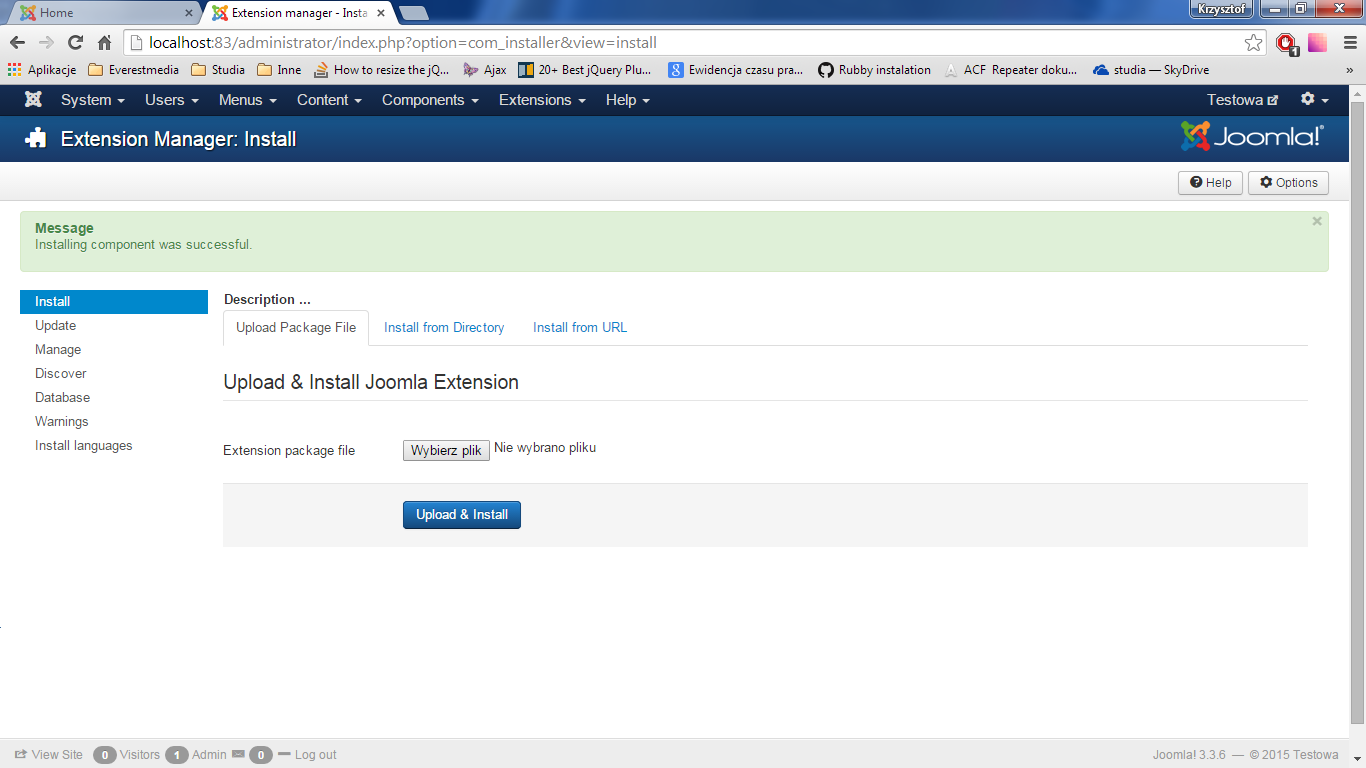
Rysunek 4 Instalacja komponentu - przejście do odpowiedniej zakładki

1. Wybrać plik z dysku (komponent *com\_roles* w formacie .zip), po czym kliknąć przycisk *Upload & Install*:



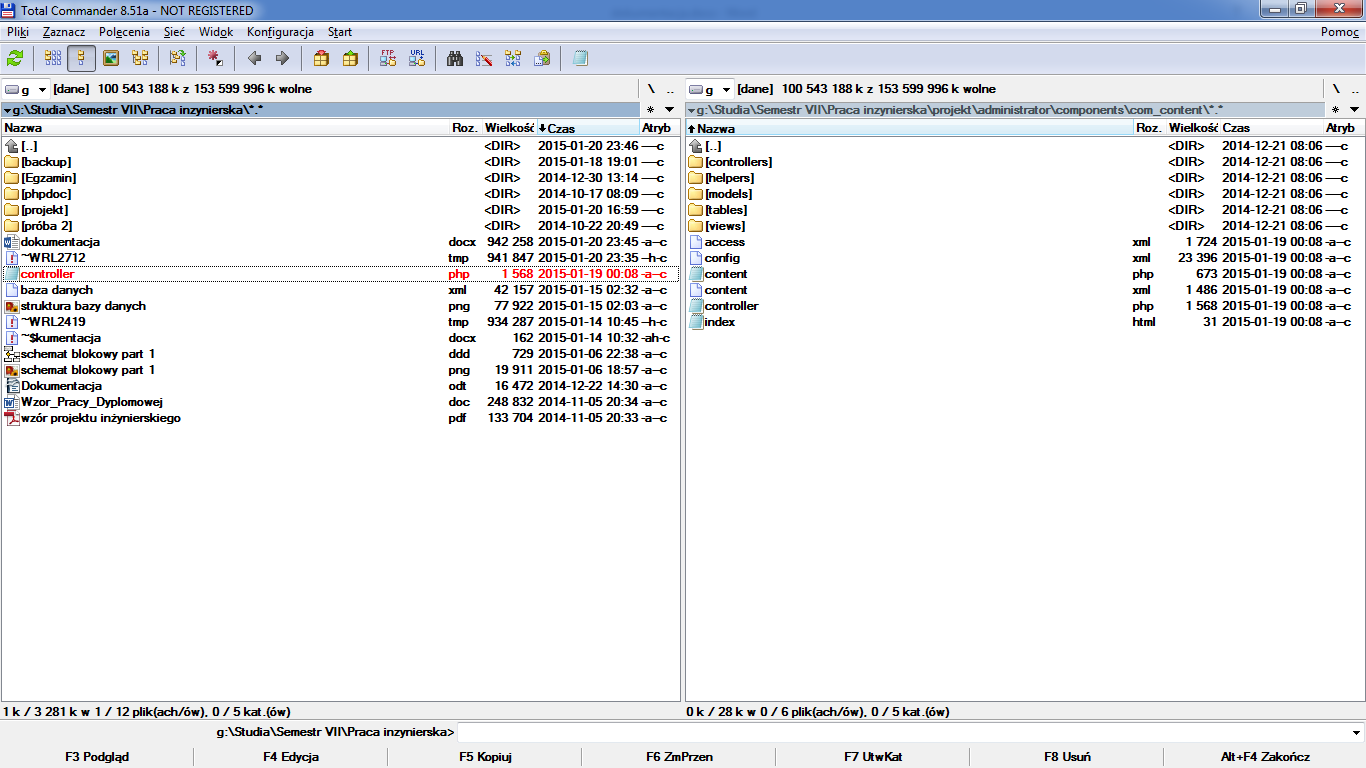
Rysunek 5 Instalacja komponentu - wybór pliku

1. Jeżeli podczas instalacji nie wystąpił żaden błąd w górnym obszarze strony pojawi się komunikat informujący o zakończonej sukcesem instalacji komponentu:



Rysunek 6 Instalacja komponentu - informacja o pozytywnym wyniku instalacji

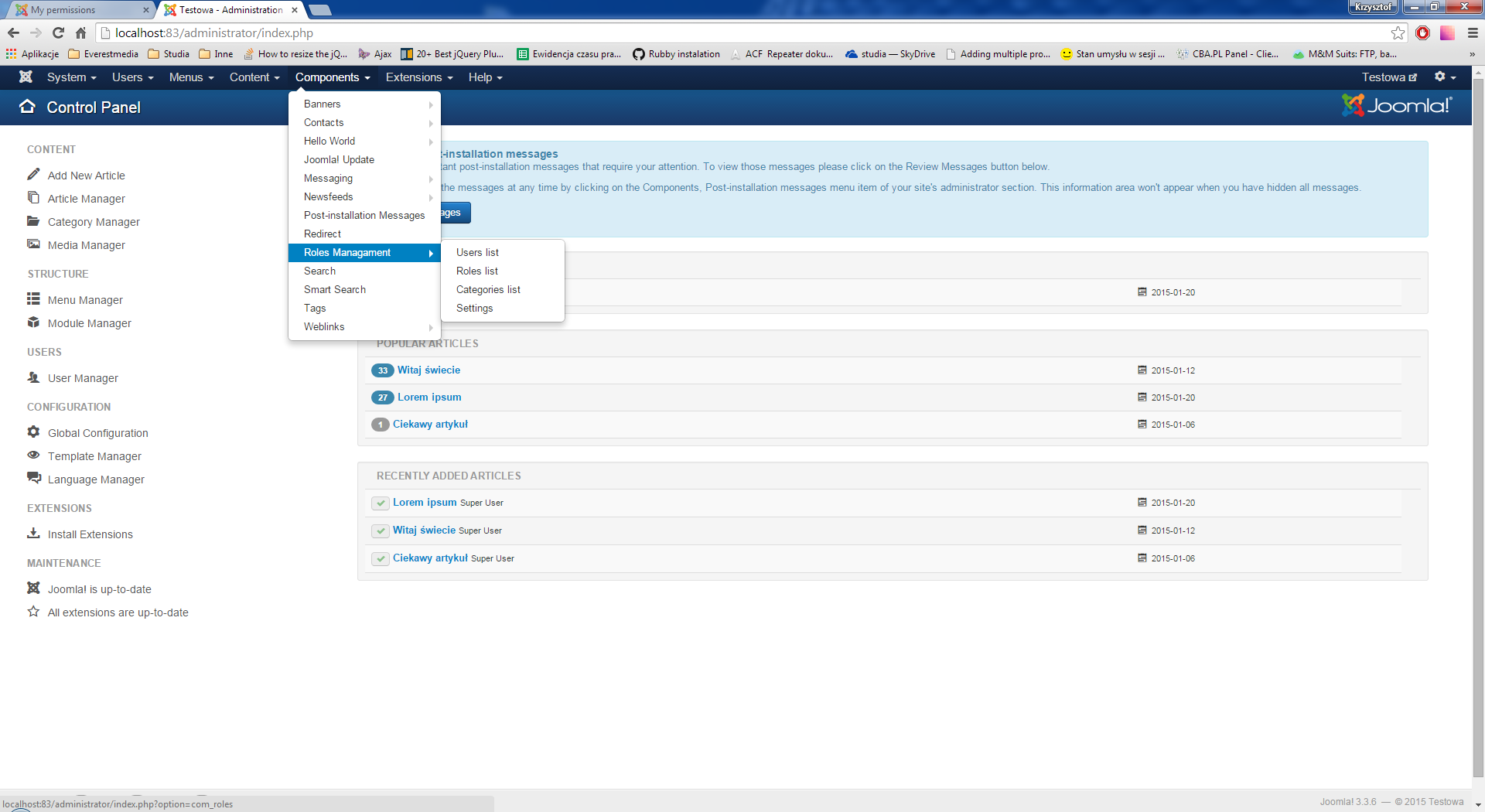
1. Po zakończonej instalacji należy za pomocą protokołu narzędzia obsługującego protokół FTP (np. Total Commander) przesłać na zdalny serwer plik *controller.php (*plik ten znajduje się także na płycie cd) do katalogu *administrator/components/com\_content*. Spowoduje to nadpisanie istniejącego już w tej lokalizacji pliku o identycznej nazwie. Czynność tą wykonuje się, ponieważ w pliku *controller.php* zostały zaimplementowane dodatkowe funkcjonalności niezbędne do prawidłowego działania instalowanego komponentu.



Rysunek 7 Przesłanie pliku controller.php na zdalny serwer

## 7.3. Zarządzanie aplikacją

Po udanej instalacji komponentu w menu głównym w zakładce *Components* pojawi się dodatkowa pozycja *Roles Management*



Rysunek 8 Położenie komponentu w menu głównym - panel administracyjny

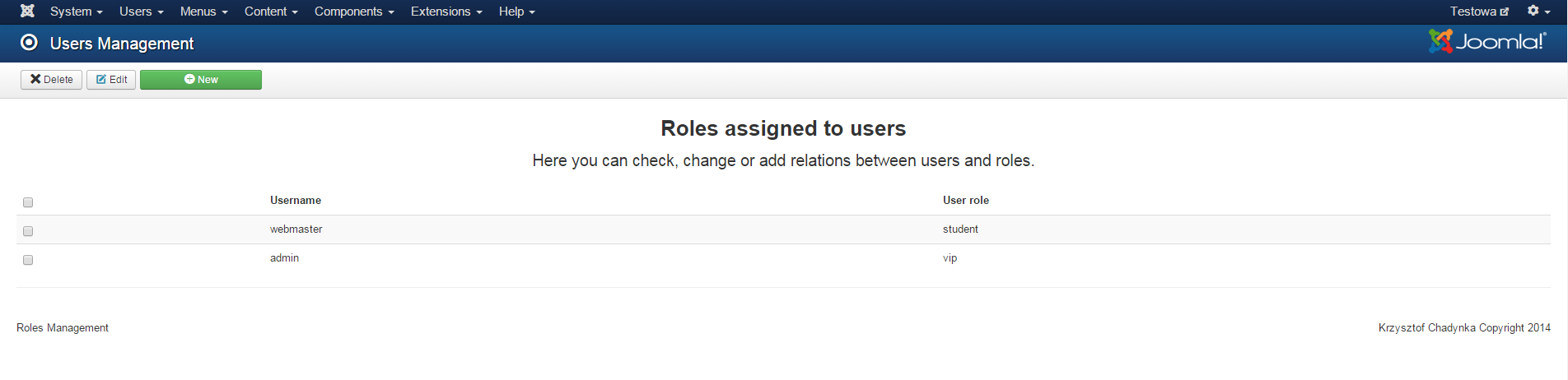
Jak zostało udokumentowane na powyższym zrzucie ekranu (*Rysunek 8)* dostępne są cztery elementy menu, z których każdy odnosi się do podstrony odpowiedzialnej za fragment funkcjonalności komponentu. Zestawienie tych funkcjonalności zostało przedstawione w tabeli:

|  |  |
| --- | --- |
| **Pozycja w menu (Podstrona)** | **Funkcjonalność** |
| Users list (Users management) | Wyświetla listę użytkowników do których zostały przyporządkowane strony. Umożliwia dodawanie nowych powiązań oraz usuwanie i edycję już istniejących. |
| Roles list (Roles management) | Wyświetla listę dostępnych ról, które można przypisać do zarejestrowanych użytkowników. Umożliwia dodawanie nowych pozycji oraz usuwanie i edycję już istniejących. |
| Categories list (Categories assigned to roles) | Wyświetla powiązania pomiędzy rolami, a kategoriami artykułów, czyli informacje o tym do których kategorii mają dostęp użytkownicy z odpowiednią rolą. Umożliwia dodawanie nowych powiązań oraz usuwanie i edycję już istniejących. |
| Settings (Frontend settings) | Wyświetla ustawienia informacji wyświetlanych na stronie w widoku informacji o użytkowniku. Umożliwia edycję ustawień. |

Tabela 3 Zestawienie podstron i ich funkcjonalności - panel administracyjny

**Obsługa poszczególnych podstron:**

1. **Users management**

****

Rysunek 9 Główny widok podstrony Users Management – panel administracyjny

Jak zostało udokumentowane na powyższym zrzucie ekranu (Rysunek 9) główny widok prezentuje listę powiązań użytkowników z rolami.

Dostępne są trzy akcje:

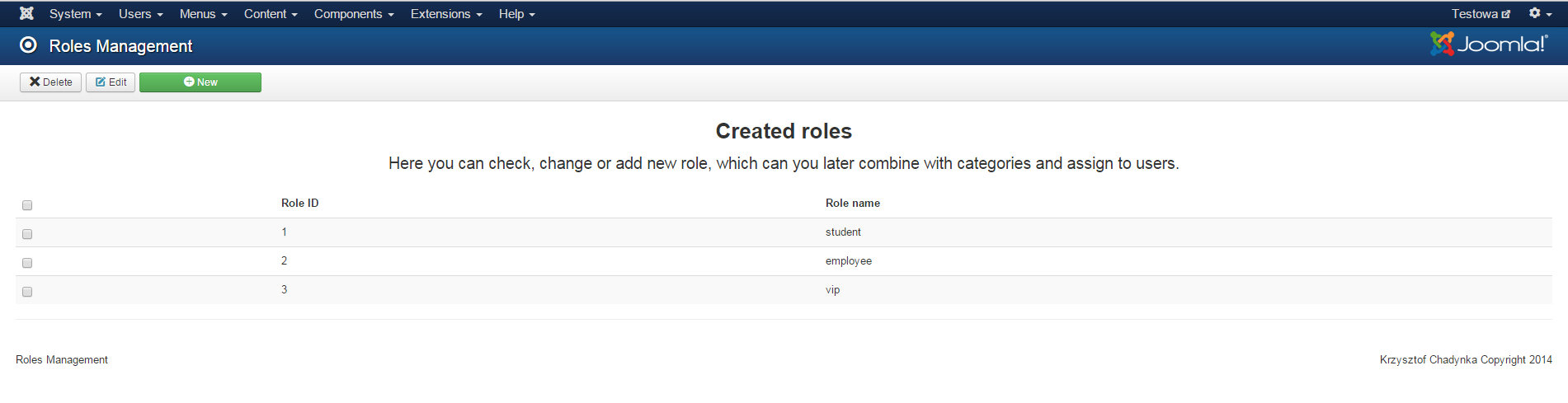
* 1. Usuwanie (należy zaznaczyć element, a następnie wybrać opcję *Delete*)
  2. Edycja (należy zaznaczyć element, a następnie wybrać opcję *Edit*)

**SCREEN!!!**

* 1. Dodawanie nowego użytkownika (Należy wybrać opcję *New)*

**SCREEN!!!**

1. **Roles list**

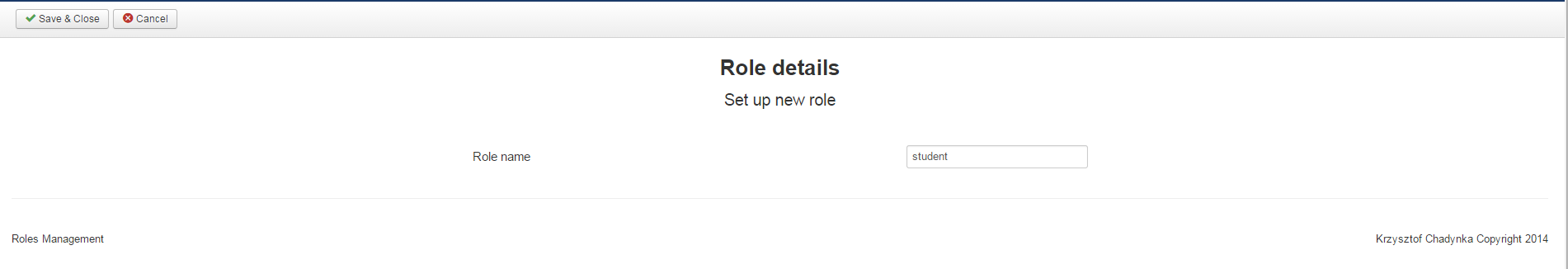
****

Rysunek 10 Główny widok podstrony Roles management – panel administracyjny

Jak zostało udokumentowane na powyższym zrzucie ekranu (Rysunek 9) główny widok prezentuje listę dostępnych ról.

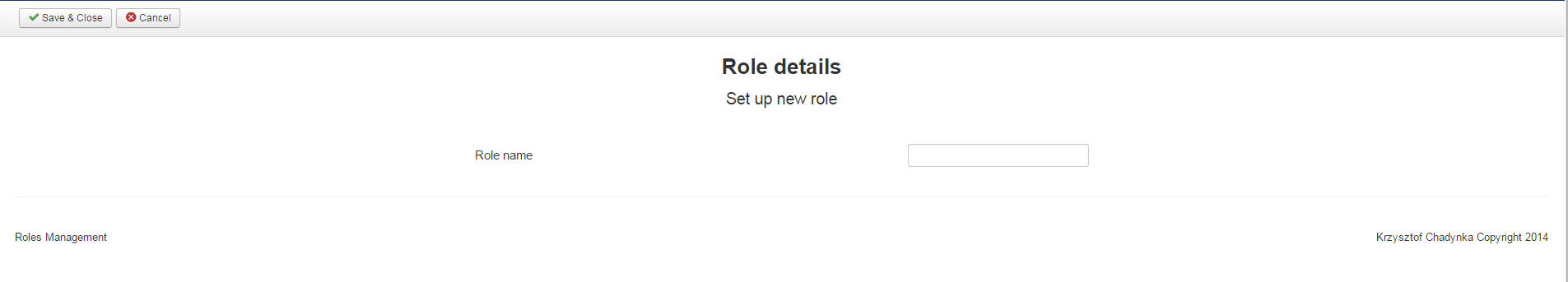
Dostępne są trzy akcje:

* 1. Usuwanie (należy zaznaczyć element, a następnie wybrać opcję *Delete*)
  2. Edycja (należy zaznaczyć element, a następnie wybrać opcję *Edit*)



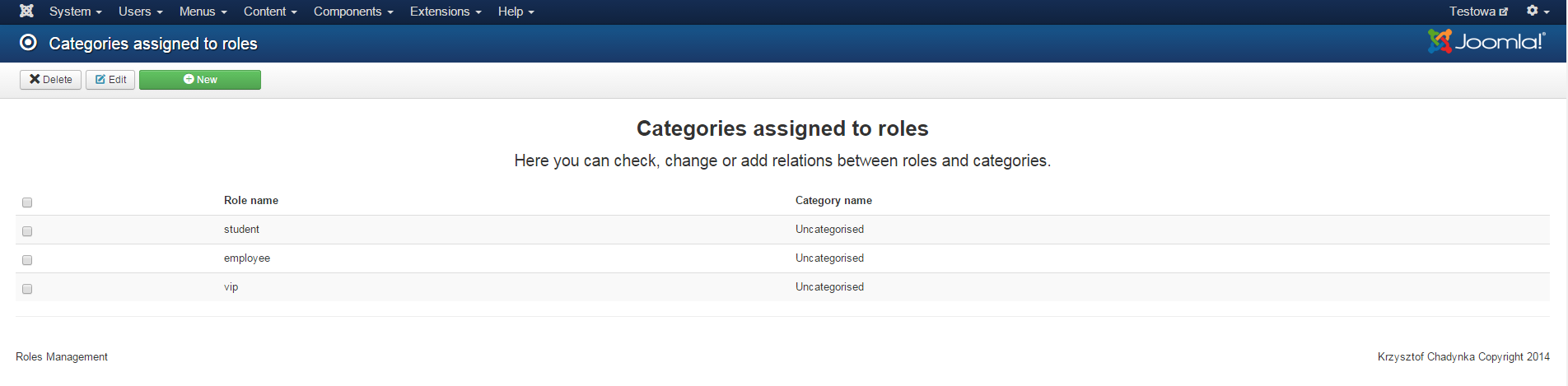
Rysunek 11 Edycja istniejącej roli – panel administracyjny

* 1. Dodawanie nowej roli(Należy wybrać opcję *New)*



Rysunek 12 Dodawanie nowej roli – panel administracyjny

1. **Categories assigned to roles**

****

Rysunek 13 Główny widok podstrony Categories assigned to roles - panel administracyjny

Jak zostało udokumentowane na powyższym zrzucie ekranu (Rysunek 9) główny widok prezentuje listę dostępnych powiązań pomiędzy rolami a kategoriami.

Dostępne są trzy akcje:

* 1. Usuwanie (należy zaznaczyć element, a następnie wybrać opcję *Delete*)
  2. Edycja (należy zaznaczyć element, a następnie wybrać opcję *Edit*)

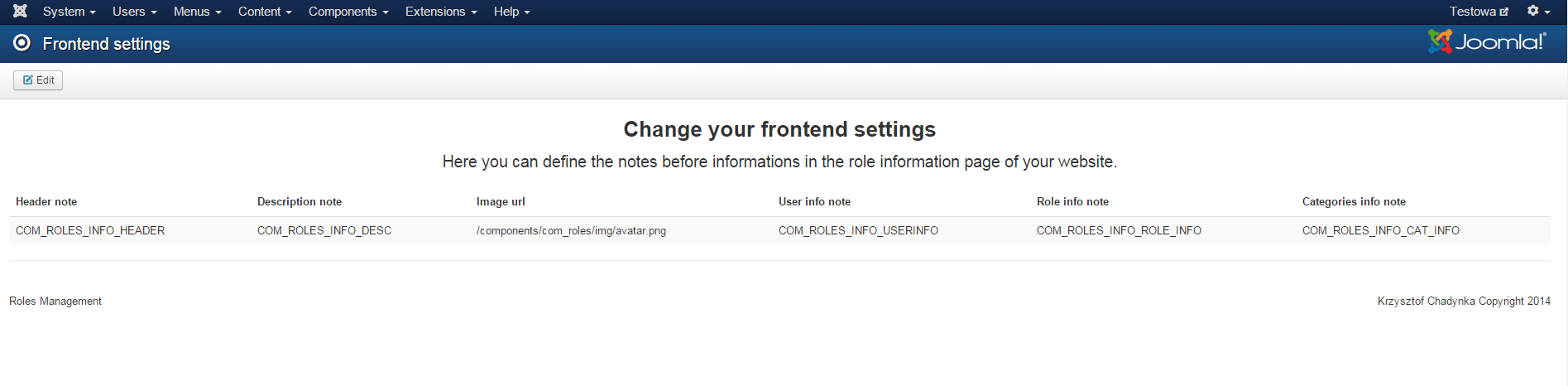
**SCREEN!!!**

* 1. Dodawanie nowego powiązania pomiędzy rolą, a kategorią (Należy wybrać opcję *New)*

**SCREEN!!!**

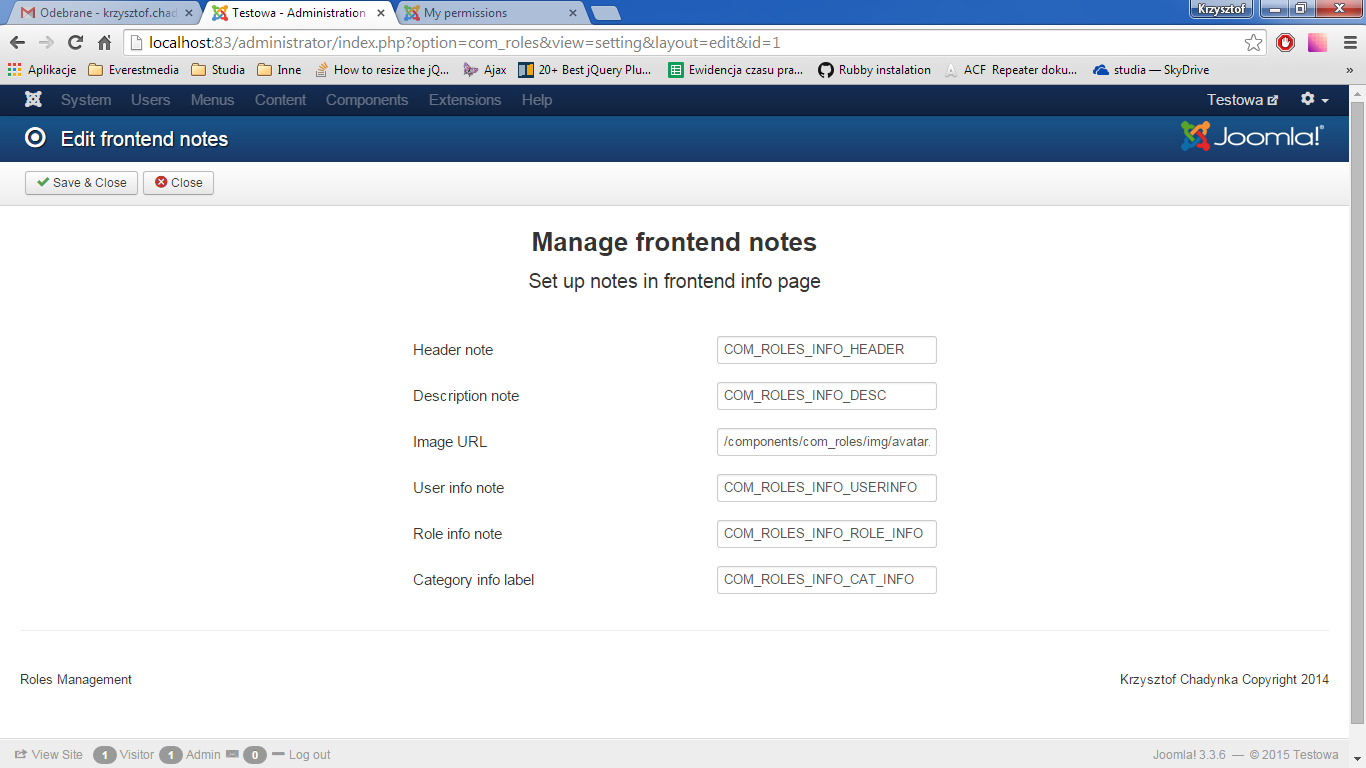
1. **Frontend settings**

Jak zostało już wspomniane w tabeli (Tabela 3) podstrona to daje możliwość edycji treści w widoku szczegółów użytkownika po stronie użytkownika korzystającego z administrowanej witryny.



Rysunek 14 Główny widok postrony Frontend settings - panel administracyjny

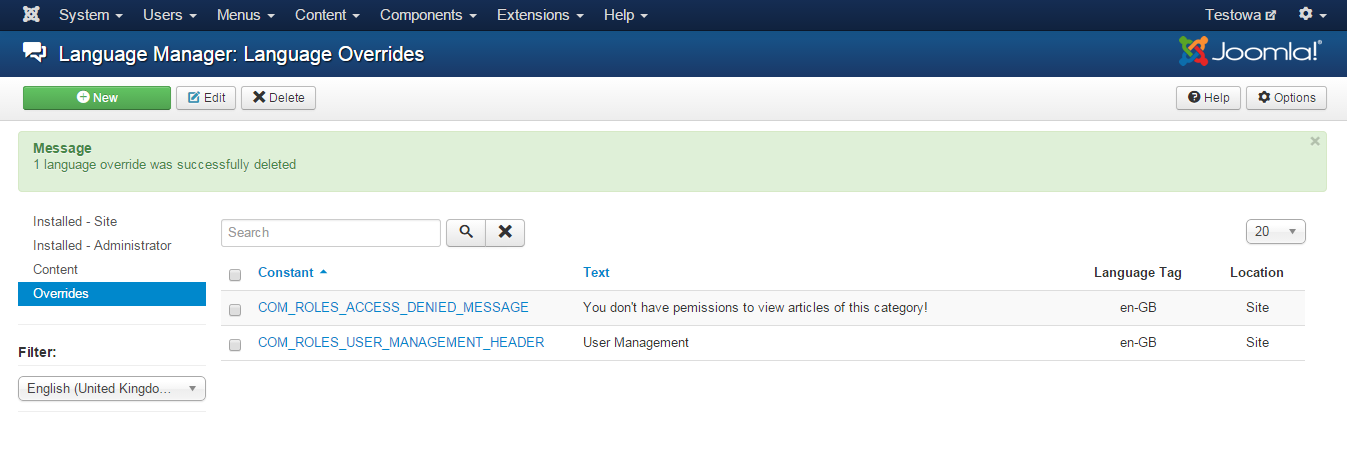
Dostępna jest opcja edycji poszczególnych treści (należy wybrać opcję *Edit)*



Rysunek 15 Edycja treści w widoku szczegółów użytkownika - panel administracyjny

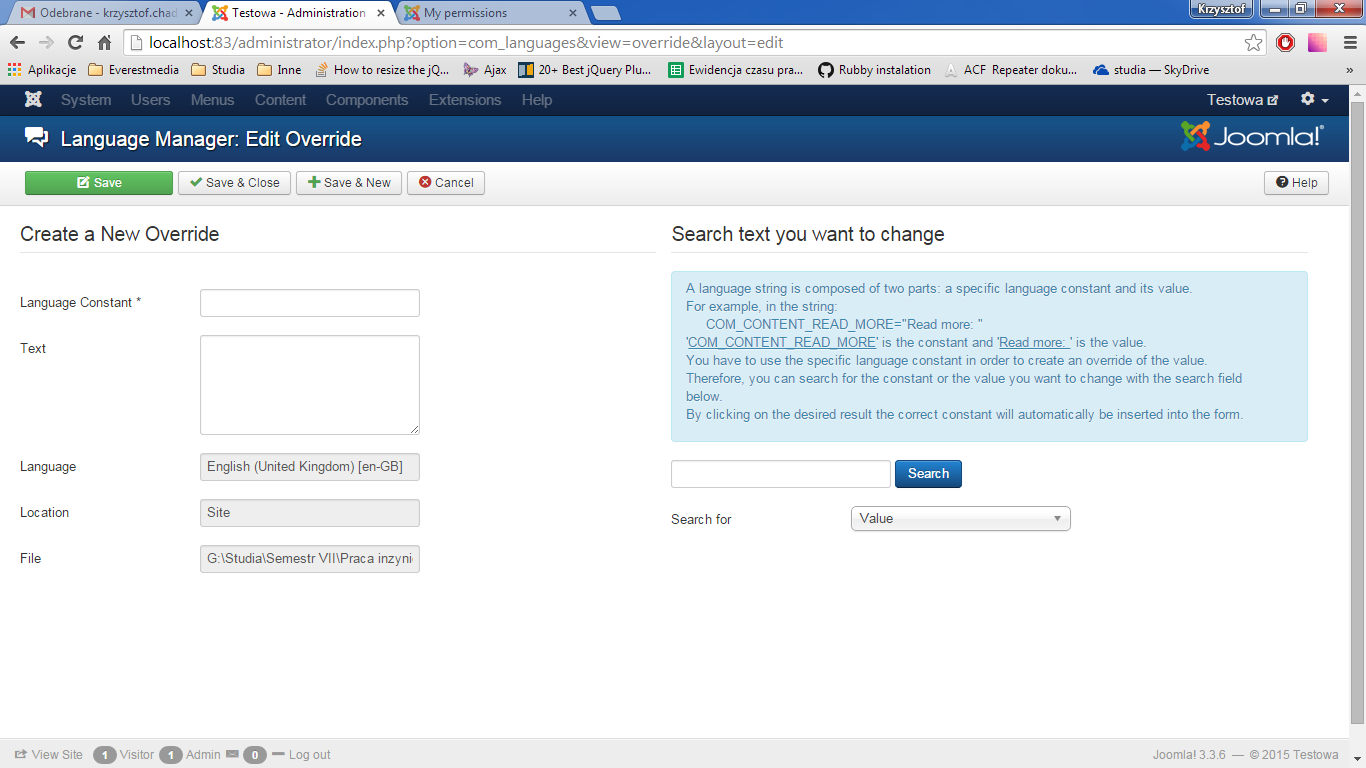
Każda treść (poza adresem url zdjęcia) została zapisana jako sekwencja, która zostanie przetłumaczona przez System Tłumaczeń framework’u Joomla!. Aby jednak tłumaczenie nastąpiło należy dodać daną treść i jej tłumaczenie do bazy:

* 1. Należy w górnym meny wybrać *Extension -> Language Manager*
  2. Z menu po lewej stronie wybrać opcję *Overrides*



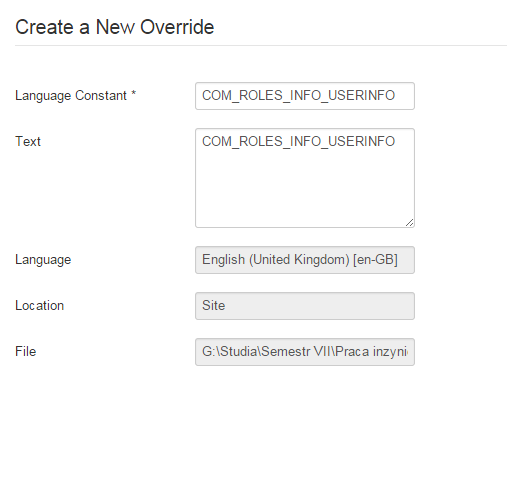
Rysunek 16 Dostępne tłumaczenia - panel administracyjny

* 1. Wybrać opcję New



Rysunek 17 Dodawanie nowego tłumaczenia - panel administracyjny

W polu *Language Constant* wpisuję się sekwencje do przetłumaczenia, natomiast w polu *Text* zostaje wprowadzony tekst, które będzie tłumaczeniem.

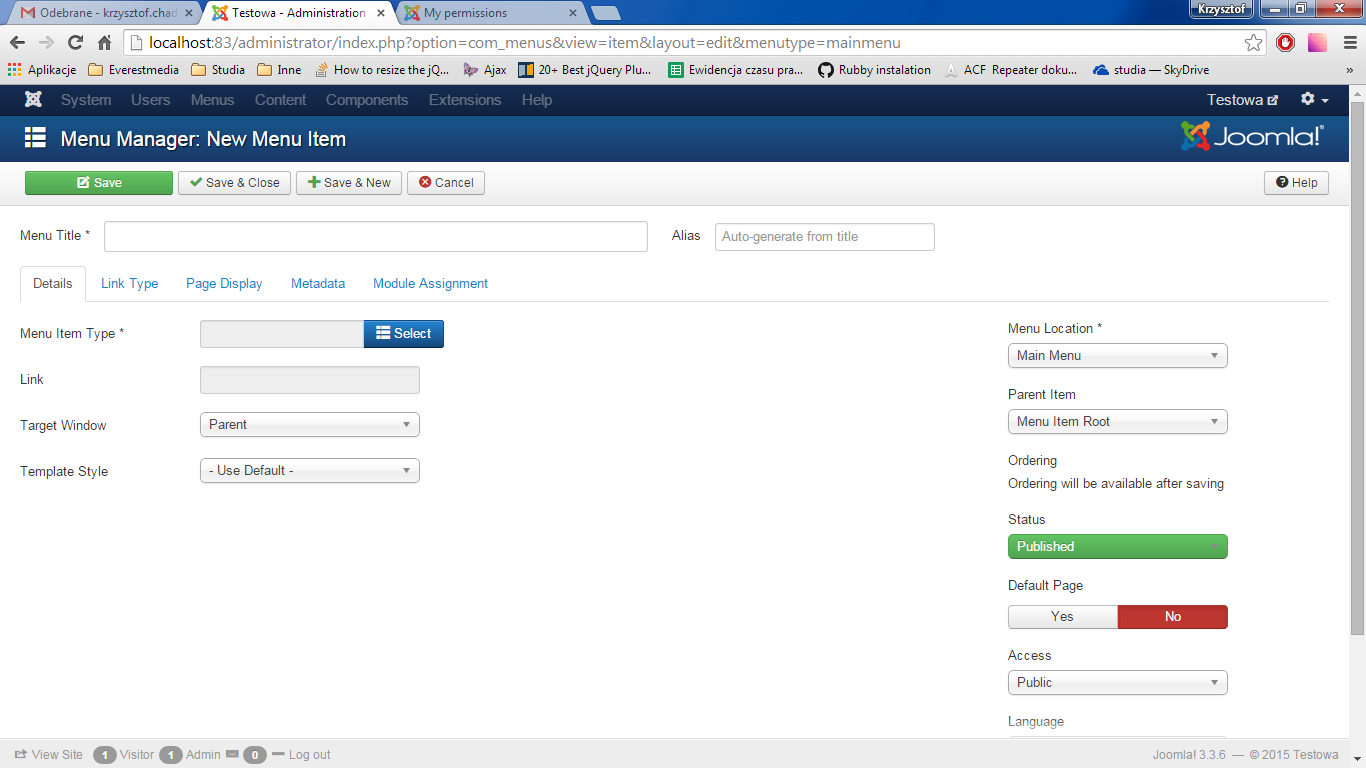


Rysunek 18 Przykładowe tłumaczenie tesktu - panel administracyjny

Można oczywiście pominąć krok tłumaczenia dodając do bazy bezpośrednie treści. Uniemożliwi to jednak automatyczne tłumaczenie strony w przypadku witryny wielojęzycznej.

Aby podstrona ze szczegółami użytkownika została wyświetlona we frontowej części witryny, należy dodać nową pozycję w menu.

W tym celu należy wybrać z menu głównego pozycję *Menus -> (jedno z dostępnych Menu) -> Add new Menu Item*



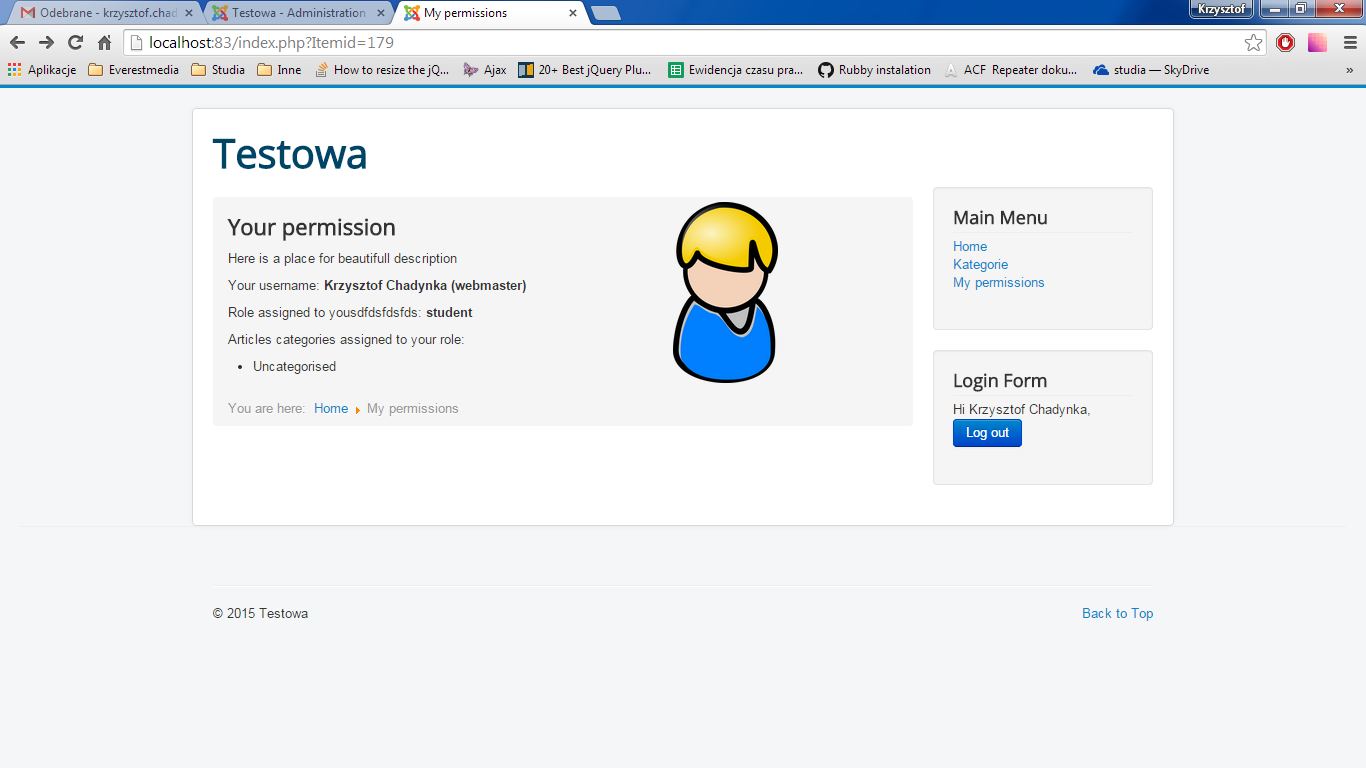
Rysunek 19 Dodanie nowej pozycji w menu - panel administracyjny

W polu *Menu Item Type* wybiera sięopcję *roles -> User info*, natomiast w polu *Menu Title* zostaje wpisana dowolna treść, która będzie wyświetlana jako odnośnik we frontowej części witryny.

## 7.4 Część frontowa witryny

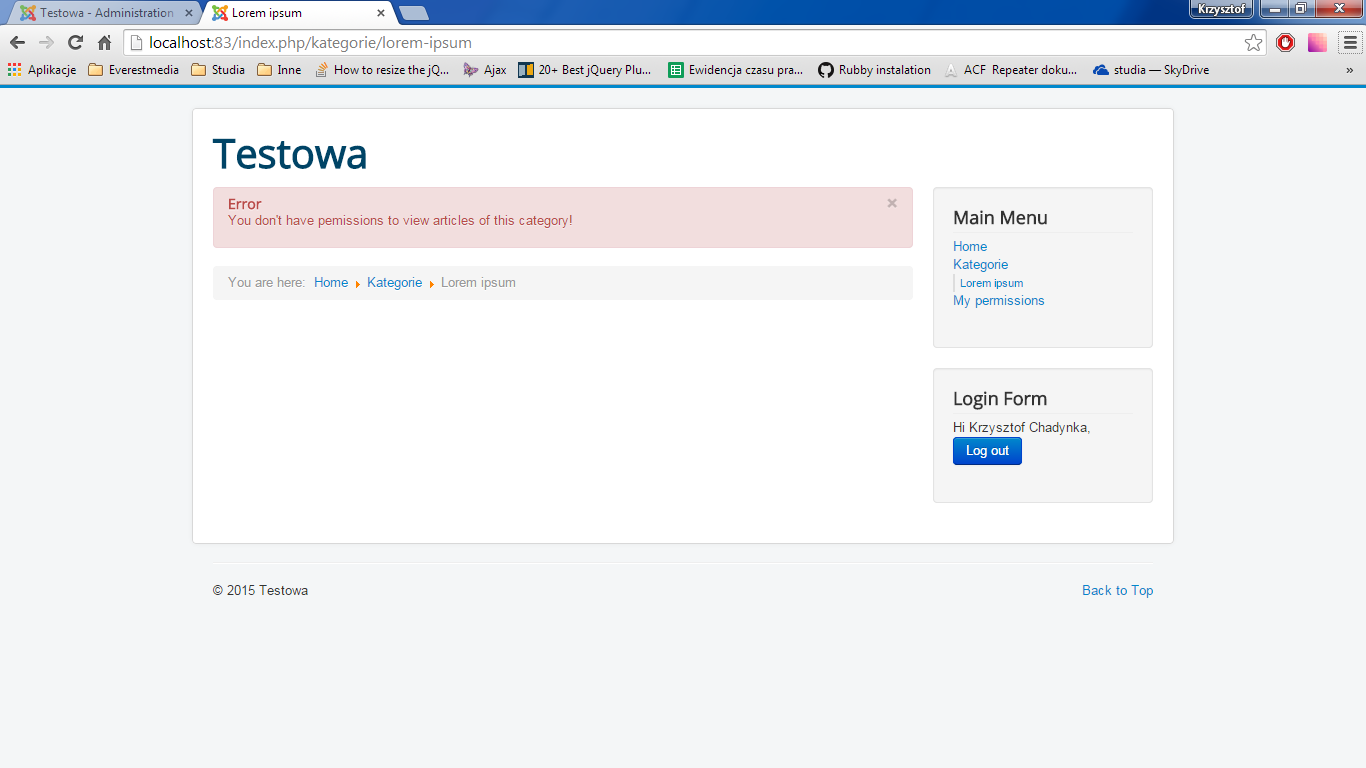
1. **Informacje na temat użytkownika**

Po wyborze opcji z menu (dodanej uprzednio przez administratora witryny) użytkownik zostaje przekierowany na stronę informacji na temat swoich uprawnień.



Rysunek 20 Podstrona informacji o użytkowniku - część frontowa

1. **Informacja o nieudanej autentykacji**



Rysunek 21 Informacja o nieudanej autentykacji - część frontowa

# 8. Podsumowanie i wnioski

# 9. Literatura

1) Dan Rahmel: *Joomla! Profesjonalne tworzenie stron WWW*, wyd. Helion 2010

2) [*www.docs.joomla.org*](http://www.docs.joomla.org)(dostęp z dnia 04.01.2015)

3) [www.wiki.joomla.pl](http://www.wiki.joomla.pl) (dostęp z dnia 06.01.2015)