



**POLSKO-JAPOŃSKA WYŻSZA SZKOŁA  
TECHNIK KOMPUTEROWYCH**

**Wydział Informatyki**

**Katedra Metod Programowania**

Programowanie Aplikacji Biznesowych

**Maciej Szczęsny**

Nr albumu 5014

**Implementacja tematycznego portalu społecznościowego  
z wykorzystaniem Struts2**

Praca inżynierska

Promotor

dr Krzysztof Barteczko

Warszawa, lipiec 2012

## Spis treści:

<b>WSTĘP .....</b>	<b>4</b>
<b>1 KONTEKST PROBLEMU .....</b>	<b>5</b>
1.1 PORTAL SPOŁECZNOŚCIOWY.....	5
1.2 PORTAL TEMATYCZNY NA PRZYKŁADZIE PORTALU KULINARNEGO.....	6
1.3 KULINARNY PORTAL SPOŁECZNOŚCIOWY .....	11
<b>2 ZAŁOŻENIA WZGLĘDEM SYSTEMU .....</b>	<b>16</b>
2.1 ZAKRES .....	16
2.2 UŻYTKOWNICY SYSTEMU .....	16
2.3 WYMAGANIA FUNKCJONALNE .....	17
2.4 WYMAGANIA NIEFUNKCJONALNE.....	21
2.5 WYMAGANIA DOTYCZĄCE INTERFEJSU .....	22
2.6 DIAGRAMY PRZYPADKÓW UŻYCIA .....	24
2.7 DIAGRAM KLAS .....	28
<b>3 ZASTOSOWANE TECHNOLOGIE .....</b>	<b>29</b>
3.1 FRAMEWORK STRUTS 2 .....	29
3.2 PLATFORMA JAVA SERVLETS I JAVA SERVER PAGES .....	30
3.3 KONTENER SERVLETÓW TOMCAT .....	31
3.4 BAZA DANYCH MYSQL.....	32
3.5 TECHNOLOGIE FRONTENDOWE .....	32
<b>4 IMPLEMENTACJA.....</b>	<b>34</b>
4.1 ARCHITEKTURA SYSTEMU .....	34
4.2 MODUŁY SYSTEMU .....	44
4.3 WALIDACJA DANYCH.....	47
4.4 LOKALIZACJA APLIKACJI .....	51
4.5 IMPLEMENTACJA WARSTWY DANYCH .....	53
4.6 IMPLEMENTACJA INTERFEJSU .....	57
<b>5 WDROŻENIE I ROZWÓJ.....</b>	<b>58</b>
5.1 TESTOWANIE APLIKACJI .....	58

5.2	REALIZACJA ZAŁOŻEŃ I MOŻLIWOŚCI ROZWOJOWE .....	63
6	LITERATURA .....	65
7	ZAŁĄCZNIKI .....	67

Celem niniejszej pracy jest sprawdzenie czy technologia Struts2 może być wykorzystana do stworzenia tematycznego portalu społecznościowego. Pod względem tematycznym zdecydowano się na portal kulinarny, który umożliwi wizytującym przeglądanie przepisów kulinarnych i wyszukiwanie potraw poprzez wpisanie posiadanych składników. Jak wspomniano, serwis ma implementować elementy serwisu społecznościowego, czyli gromadzić miłośników jedzenia wokół wspólnego zainteresowania poprzez zakładanie profili, dodawanie potraw (i ich zdjęć), komentowanie przepisów innych użytkowników, otrzymywanie punktowej oceny swoich potraw, zdobywanie punktów doświadczenia, które to punkty będą miały odzwierciedlenie w hierarchii użytkowników.

Kolejne rozdziały przedstawiają opis dziedziny problemowej, tj. analizę portali społecznościowych i kulinarnych, a także analizę połączenia cech tych dwóch rodzajów portali (rozdział drugi). W następnych rozdziałach sformułowano szczegółowe wymagania, które ma spełniać kulinarny portal społecznościowy (rozdział trzeci) i opisano zastosowane technologie (rozdział czwarty). Rozdział piąty przedstawia architekturę systemu, jego implementację i moduły funkcjonalne. W rozdziale szóstym skupiono się na przeprowadzonych testach aplikacji i analizie realizacji pierwotnych założeń. Dodatek (załączony na płycie CD) zawiera ponadto instrukcję instalacji, jak również opis obsługi aplikacji.

Produkt programistyczny będący wynikiem niniejszej pracy dowiódł, iż framework Struts2 może być wykorzystany do implementacji tematycznego portalu społecznościowego. Zaimplementowano bowiem system zgodny z wymaganiami charakterystycznymi dla takiego portalu, a określonymi w toku analizy podobnych serwisów obecnych w Internecie.

## 1 KONTEKST PROBLEMU

### 1.1 PORTAL SPOŁECZNOŚCIOWY

Rozwój portali społecznościowych związany jest z trendem jaki powstał w Internecie w roku 2001 – Web 2.0 – zakładającym, że treść portalu jest generowana przez samych jego użytkowników (tzw. *user-generated content*). Odbywa się to poprzez komentarze, blogi, fora. Użytkownicy wchodzą ze sobą w interakcje, wyrażają opinie, dzielą się informacjami, oceniają, wymieniają multimediami, wspólnie podejmują działania. Jednym słowem to oni, a nie wyspecjalizowani pisarze, dziennikarze, producenci, są motorami napędowymi tychże portali. [19]

Na tej właśnie fali powstało wiele serwisów społecznościowych. Niewątpliwie najpopularniejszym z nich jest amerykański Facebook, który pojawił się w sieci w 2004 roku. Wg danych z grudnia 2011 portal ten posiada 845 milionów aktywnych użytkowników, z czego ponad połowa loguje się do niego codziennie; 425 milionów użytkowników odwiedza Facebooka wykorzystując telefon lub tablet ([5]). Statystyki te wyraźnie pokazują, że dla wielu internautów portale społecznościowe są główną formą kontaktu ze znajomymi. Jednocześnie popularność Facebooka sprawia, że implementowane przez niego funkcje można traktować jako kanon funkcji społecznościowych gwarantujących sukces.

Facebook, obok Google+ i polskiego Grono.net, to portale społecznościowe o charakterze ogólnym. Istnieją też portale skierowane do konkretnej grupy odbiorców – np. Nasza Klasa skierowana do uczniów/studentów. Niektóre portale skupiają się na dzieleniu się określonymi treściami, np. YouTube. Równie popularne są społecznościowe serwisy tematyczne, które służą dzieleniu się opinii na określone tematy – przykładem może być tu polski Filmweb skupiający się na filmach i ludziach kina. Wśród takiego rodzaju portali możemy również wyróżnić kulinarne serwisy społecznościowe, o których będzie mowa w rozdziale 1.3.

## 1.2 PORTAL TEMATYCZNY NA PRZYKŁADZIE PORTALU KULINARNEGO

Wśród informacji, które można odnaleźć w Internecie są również te związane z gotowaniem i potrawami. Na pewno każdy z nas trafił na stronę z przepisami – czy to przez przypadek czy celowo. W rzeczywistości raczej trudno tego uniknąć, jako że liczba takich portali w Internecie jest naprawdę spora. Możemy wyróżnić samodzielne portale kulinarne ([allrecipes.com](http://allrecipes.com), [przepisy.net](http://przepisy.net)) lub powiązane z popularnymi serwisami ([shine.yahoo.com/channel/food](http://shine.yahoo.com/channel/food), [kuchnia.wp.pl](http://kuchnia.wp.pl)). Niektóre z nich są tworzone przez społeczność ([www.grouprecipes.com](http://www.grouprecipes.com), [portalkucharski.pl](http://portalkucharski.pl)), inne prowadzone w formie blogu przez jedną osobę ([hazelcooks.com](http://hazelcooks.com)). W obecnych czasach każdy z serwisów kulinarnych posiada elementy społecznościowe, chociażby za sprawą takich funkcji jak: możliwość zakładania profilu, logowania, manipulacji hasłem, edycji profilu (Rysunek 1.1), dodawania przepisów (Rysunek 1.2) czy komentarzy. Możemy zatem wyróżnić funkcje związane czysto z przeglądaniem potraw i funkcje społecznościowe.

### Aktualizuj informacje o sobie

Prosimy teraz o uzupełnienie w formularzu danych niezbędnych lub przydatnych do korzystania z serwisu. Dzięki nim będziesz mógł zalogować się, nawiązywać znajomości i uczestniczyć w tworzeniu serwisu Portalkucharski.pl.

#### 1 Wpisz swój nowy email

Podaj prawidłowy adres e-mail, gdyż na ten adres zostanie przysłany link potwierdzający i aktywujący rejestrację. Twój e-mail służy tylko do rejestracji i nie będzie widoczny dla innych.

#### 2 Potwierdź swój nowy email

Ten sam e-mail jeszcze raz, nie kopiuj tylko starannie wpisz.

#### 3 Podaj nowe hasło

max 30 znaków

Min 6 znaków

Pasek pokazuje siłę zabezpieczenia Twojego hasła.

#### 4 Potwierdź nowe hasło

max 30 znaków

#### 5 Twoje dane teleadresowe

Dla 1000 naszych najlepszych użytkowników wydawana jest w formie papierowej gazetka. Mamy nadzieję że będziesz jednym z nich i chciałbyś ją dostawać do domu dlatego podaj prawidłowe dane.

Podaj swoje imię

Podaj swoje nazwisko

Wpisz nazwę ulicy gdzie mieszkasz

Nr domu

Nr lokalu

Wpisz kod pocztowy

Podaj miasto w jakim mieszkasz

Podaj kraj

Podaj województwo w jakim mieszkasz

Podaj datę urodzenia

#### 6 Zaznacz swoją płeć

☐

Kobieta

☒


Mężczyzna

**Aktualizuj**

Rysunek 1.1 - edycja profilu w portalu portalkucharski.pl

## Aby dodać przepis, uzupełnij formularz:

Wpisz tytuł przepisu

 Dodaj zdjęcie potrawy

Wpisz składniki

Wpisz opis przygotowania potrawy

Wybierz kategorię potrawy

Bezmięsne

Czy to jest danie trudne

bardzo łatwe

Jaki jest koszt potrawy?

bardzo tanie

Jaki jest czas przygotowania potrawy?

w 5 minut

Kiedy podać potrawę?

na śniadanie

☐ czy pomocny będzie blender

Wpisz tagi po przecinku lub wybierz z listy poniżej:

:) Babka Bezmięsne Biszkopt Ciasta Ciastka Ciasto Czekolada Czosnek Deser Desery i ciasta Dorsz Grzyby Inne Jabłka Jajka Jajko Jedzenie Kalorie Krem Kurczak Makaron Migdały Mięśne Mięso Mąka Napoje Owoce **POLECAM** Pierogi Pomidory Przetwory Ryba Ryby Ryż Sałata Sałatka Sałatki i surówki **Smacznego!** Sos Sosy Truskawki Warzywa Wino Zapiekanka Zupa Zupy alkohol ananas bakalie boczek cebula chleb ciasteczka cukier cukinia **danie główne** desery dieta drób galaretka kakao kapusta kawa kielbasa kokos kukurydza lody majonez mak marchew masło miód mleko naleśniki obiad ogórki orzechy papryka pieczarki piernik piwo pizza placek **przyprawy** schab ser sernik smacznego szpinak szynka tort tuńczyk twaróg zdrowie ziemniaki Łosoś **Świąteczne** śmietana **święta Bożego Narodzenia**

Rysunek 1.2 - dodawanie potrawy w serwisie kuchnia.wp.pl

Jeśli chodzi o funkcje związane z przeglądaniem potraw, twórcy portali kulinarnych umożliwili dotarcie do interesującego nas dania na kilka sposobów. Tradycyjną formą są katalogi przepisów, w których to potrawy są podzielone ze względu na rodzaj (*zupy* czy *ryby* w portalu [kuchnia.wp.pl](http://kuchnia.wp.pl)), pochodzenie geograficzne (*Rosja* czy *Indie* w portalu [gotowanie.onet.pl](http://gotowanie.onet.pl)) czy chociażby porę dnia (*śniadanie* czy *obiad* w portalu [przepisy.net](http://przepisy.net)). Niektóre portale oferują jednoczesną selekcję wyników dla wszystkich podanych wyżej kryteriów (Rysunek 1.3) lub nawet kryteriów związanych z poziomem trudności czy czasem przygotowania (Rysunek 1.4).



## Przepisy na śniadanie

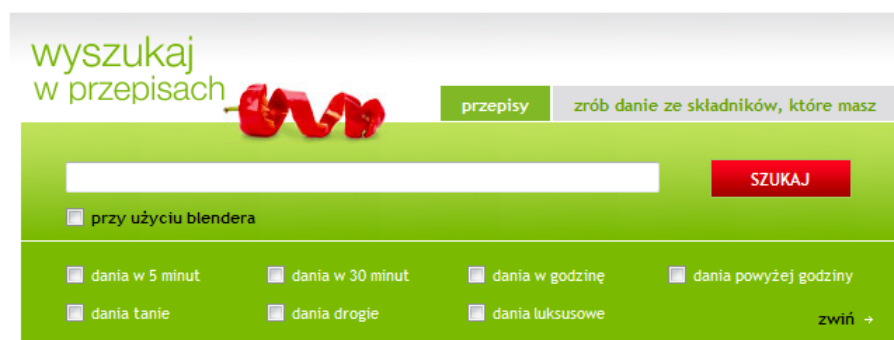


Rysunek 1.3 - filtracja wyników w portalu przepisy.net



Rysunek 1.4 - filtracja wyników w portalu ugotuj.to

Oczywistym sposobem na odnalezienie potrawy jest wyszukiwarka implementowana przez większość serwisów kulinarnych. Poziom szczegółowości w takich wyszukiwarkach może być różny (Rysunek 1.5), a wyniki wyszukiwania mogą być sortowalne (Rysunek 1.6).



Rysunek 1.5 - wyszukiwarka dań w portalu kuchnia.wp.pl



Rysunek 1.6 - sortowanie wyników w portalu epicurious.com

Istnieją również wyszukiwarki wypływające wyniki dla podanych składników, które się już posiada (Rysunek 1.7). Wyszukiwarki te ([kuchnia.wp.pl](http://kuchnia.wp.pl), [gotowanie.onet.pl](http://gotowanie.onet.pl), [portalkucharski.pl](http://portalkucharski.pl)) podają jednak wyniki przybliżone – niestety by przygotować wyszukaną potrawę, trzeba zazwyczaj parę składników jeszcze dokupić.

**Wpisz więcej składników, które masz w kuchni, aby skomponować potrawę**

wpisz więcej składników →

znajdź przepis

Rysunek 1.7 - wyszukiwarka po składnikach w portalu kuchnia.wp.pl

Bardzo ciekawym rozwiązaniem jest również umieszczanie w widocznym miejscu na stronie odnośników do najnowszych i/lub najlepiej ocenianych przepisów (Rysunek 1.8).

#### Najnowsze przepisy

Ostatnio dodane

- [Faworki](#)
- [Papryka faszerowana](#)
- [Makaron ze szpinakiem i serem szopskim](#)
- [Tuńczykowe szalerństwo](#)
- [Coconut muffins](#)
- [Drugie śniadanie od serca](#)
- [Placek na maślanie z owocami](#)

[więcej »](#)

#### Najpopularniejsze przepisy

Oceny internautów

- [Szarlotka z kruszonką](#) ★★★★★
- [Rolada z wędzonego łososia](#) ★★★★★
- [Sałatka makaronowa](#) ★★★★★
- [Przekładaniec brzoskwinowy](#) ★★★★★
- [Rolada z indyka z morelami i śliwkami](#) ★★★★★
- [Medaliony wieprzowe z sosem winogronowym](#) ★★★★★
- [Pierogi z kaszą gryczaną i mięsem](#) ★★★★★

[więcej »](#)

Rysunek 1.8 - najnowsze i najlepiej oceniane przepisy w portalu gotowanie.onet.pl

Opisane powyżej funkcjonalności są związane z samym przeglądaniem dań, czyli tym na czym bazują portale kulinarne. Podążając jednak za myślą Web 2.0 twórcy tych witryn założyli, iż przepisy będą generowane przez samych użytkowników. Sposób, w jaki to osiągnęli przedstawiono w rozdziale 1.3.

### 1.3 KULINARNY PORTAL SPOŁECZNOŚCIOWY

Jak stwierdzono w rozdziale 1.2 każdy portal kulinarny w obecnym czasie implementuje w pewnym stopniu funkcjonalności społecznościowe. Te podstawowe związane są z możliwością założenia profilu i dodawania przepisów, co jest zgodne z trendem Web 2.0. Niektóre portale idą jednak dalej – funkcje społecznościowe są w ich przypadku bardzo rozbudowane. Zauważono bowiem, iż za pomocą tych funkcjonalności, dany użytkownik buduje własną tożsamość i co za tym idzie angażuje się w społeczność tworząc treść portalu, a o to przecież chodzi, by portal sam siebie napędzał.

Już po utworzeniu profilu i zalogowaniu pojawiają się przed użytkownikiem nowe możliwości. Przeglądając dania, może dodawać je do obserwowanych (like’owanych), oceniać, komentować, dodawać zdjęcia, śledzić przepisy autora (Rysunek 1.9).

## Heroin Chicken Wings

From [lillyann](#) 4 days ago

[zobacz profil autora](#)

60 minutes to make | Serves 56



“ Why are these chicken wings prefaced with heroin? Once you try these, you'll understand the name – utterly, totally addictive! You'll impress the heck out of your friends – and wish you'd made more!

chicken baked lowcarb easy appetizer bar more...



[Add yours](#)

[dodaj własne zdjęcie potrawy](#)

[śledź przepisy dodane przez autora](#)

### Ingredients

24 [chicken wing](#) drumettes (about 2-1/2 lb/1 kg)  
1 cup grated Parmesan  
2 tbsp [dried parsley](#)  
1 tbsp dried leaf [oregano](#)  
2 tsp [paprika](#)  
1 tsp [salt](#)  
1/2 tsp [pepper](#)  
About 1/3 cup melted [butter](#)

### How to make it

First, preheat the oven to 350°F.

Cut the wings up into "drumettes". (Freeze the pointy "tips" for soup – they make great broth!)

Then combine the grated cheese and the seasonings. Line a shallow baking pan with foil. (Do not omit this step, or you'll still be scrubbing the pan for days!)

Melt the butter in a shallow bowl or pan.

Dip each "drumette" in butter, roll in the seasoned cheese, and arrange in the foil lined pan. Bake for 30-45 minutes at 350°F.

Then ... kick yourself that you didn't make a double recipe!!

[oceniaj potrawę](#)

### People Who Like This Dish 243

Colleen5  
nowhere, us

Knudsenrick  
toronto, ca

Confiocco  
nowhere, us

Mzstyl3z  
nowhere, us

Momheenda  
nowhere, us

Niceboy\_alpha  
mumbai, india

Pywhacket  
nowhere, us

Sexy  
nowhere, us

Kimsmomma  
erlanger, ky

Liisuzi  
nowhere, us

Plus 233 others  
From around the world!

[skomentuj potrawę](#)

### Reviews & Comments 63

[All Comments](#)

[Your Comments](#)



What do you think?

[Add a Link?](#)

### The Cook



Lillyann

Chicago, IL

[Subscribe to My Recipes](#)

Like lillyann's recipes?  
Never miss an upload!

### The Rating

Delicious!

★★★★★  
Reviewed by 32 people

I love chicken wings ... my favorite part of the bird! Thanks.



[linebb956](#)

in La Feria loved it

These sound wonderful. will be making a double batch.



[chervilyn](#)

in Salem loved it

Wow! Great pic, drooling just by looking at it



[liezel](#)

in Bloemfontein loved it

[A Few More Reviews](#)

### The Groups



Party Food

315 members

[PUBLIC](#)



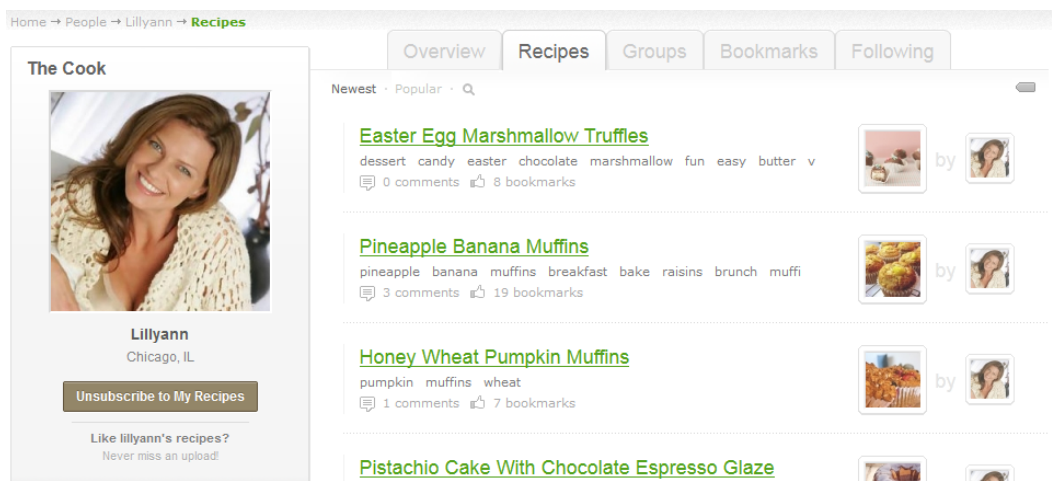
Tailgate Time

129 members

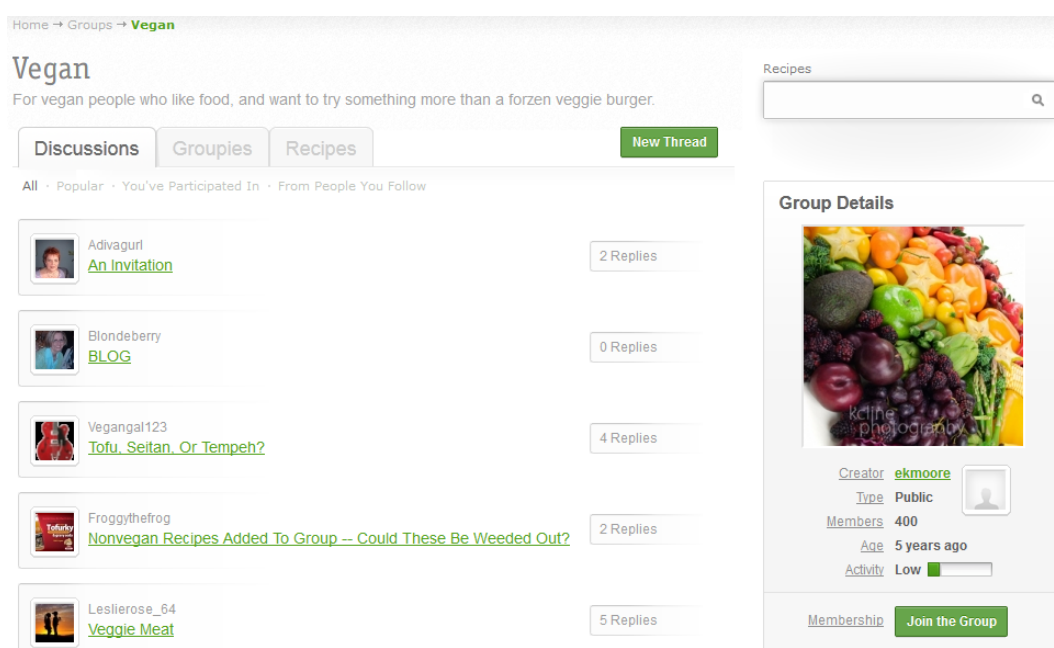
[PUBLIC](#)

Rysunek 1.9 - funkcjonalności dostępne dla zalogowanego użytkownika w portalu grouprecipes.com

Przeglądając profil użytkownika (Rysunek 1.10), mamy dostęp do przepisów przez niego dodanych; grup, do których należy (użytkownicy mogą skupiać się w grupy, w których mogą dyskutować na różne tematy – Rysunek 1.11); potraw, które polubił; użytkowników, którzy śledzą jego aktywność.

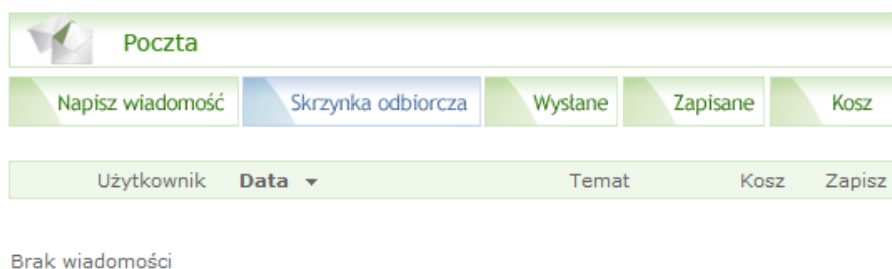


Rysunek 1.10 - profil użytkownika w portalu grouprecipes.com

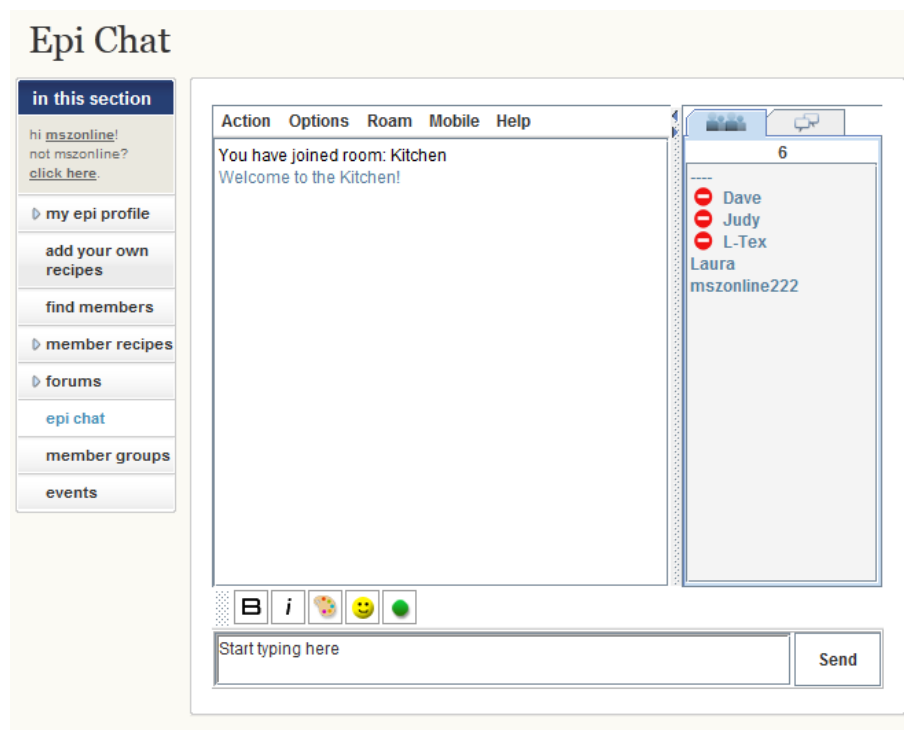


Rysunek 1.11 - strona grupy w portalu grouprecipes.com

Niektóre portale udostępniają również dla zalogowanych użytkowników miejsca do prowadzenia rozmów (Rysunek 1.13) czy wprowadzają możliwość korespondencji prywatnej (Rysunek 1.12).



Rysunek 1.12 - korespondencja prywatna w portalu portalkucharski.pl



Rysunek 1.13 - chat w portalu epicurious.com

Inne z kolei, celem zachęcania użytkowników do publikowania treści, wprowadzają rankingi aktywności. Określone czynności w portalu (np. dodanie przepisu) skutkują doliczeniem pewnej liczby punktów do już zgromadzonych. Niejednokrotnie skutkuje to przyznaniem wyższej rangi w hierarchii użytkowników (Rysunek 1.14). Punkty mogą być też wymieniane na nagrody.

Rysunek 1.14 - punkty za aktywność w portalu portalkucharski.pl

Powyższa analiza pokazuje, iż portale kulinarne są prześięknięte typowymi funkcjonalnościami znanymi z portali społecznościowych. Tworząc własny portal kulinarny nie można ignorować tych cech, bez których, można śmiało stwierdzić, nie ma on po prostu bytu. Przy obecnym trendzie w Internecie każdy portal kulinarny musi być jednocześnie portalem społecznościowym. Następny rozdział skupia się na sformułowaniu wymagań biznesowych dla takiego właśnie portalu internetowego, którego możliwość implementacji z wykorzystaniem frameworku Struts2 postanowiono sprawdzić.

## 2 ZAŁOŻENIA WZGLĘDEM SYSTEMU

### 2.1 ZAKRES

Portal będzie dostępny dla wszystkich. Niezalogowani użytkownicy (tj. goście) będą mogli wyszukiwać przepisy na 3 różne sposoby: wpisując posiadane składniki, przeglądając katalog potraw lub wyszukując słowa kluczowe w nazwach potraw.

Zalogowani użytkownicy będą mieli większe uprawnienia. Nie tylko będą mogli komentować i oceniać potrawy innych użytkowników, ale również dodawać własne przepisy zdobywając tym samym punkty doświadczenia. Punkty doświadczenia będą przyznawane za określone osiągnięcia (np. dodanie zdjęcia potrawy), co będzie odzwierciedleniem aktywności danego użytkownika.

Portal będzie kontrolowany przez super-użytkowników, którzy będą aprobować „treść” stworzoną przez zwykłych użytkowników, edytować profile użytkowników lub blokować użytkowników za niepoprawne zachowanie na stronie.

### 2.2 UŻYTKOWNICY SYSTEMU

#### 2.2.1 GOŚĆ

Gość jest zwykłym wizytującym stronę, który może tylko przeglądać przepisy.

#### 2.2.2 UŻYTKOWNIK

Użytkownik jest gościem, który stworzył profil na stronie i uzyskał określone przywileje, przede wszystkim możliwość prowadzenia aktywnego życia na portal poprzez dodawanie własnych potraw, przeglądanie potraw innych użytkowników i wypowiadanie się na ich temat.

#### 2.2.3 ADMINISTRATOR

Administrator jest super-użytkownikiem, który ma prawo edytować profile innych użytkowników, blokować potrawy przez nich dodane, jak również aprobować komentarze.



## 2.3 WYMAGANIA FUNKCJONALNE

### 2.3.1 OPIS STRUKTUR

#### 2.3.1.1 UŻYTKOWNICY

System powinien przechowywać informacje na temat użytkowników. Obowiązkowymi przechowywanymi danymi użytkownika są: adres e-mail i hasło (używane do logowania), nick, data urodzenia (te cztery wypełniane w momencie rejestracji nowego użytkownika), poziom uprawnień, czas i data rejestracji, liczba punktów doświadczenia (odzwierciedlająca rangę użytkownika), liczba punktów użyta do redempcji. Inne dane użytkownika to: awatar, kraj, strona WWW, płeć. Po stworzeniu profilu, status użytkownika określony jest jako „nieaktywny” – email z kodem aktywacyjnym wysyłany na adres email podany przy rejestracji. By użytkownik stał się „aktywny” musi kliknąć w link wysłany w mailu.

#### 2.3.1.2 RANGI

Użytkownikom będą przypisywane odpowiednie rangi odzwierciedlające liczbę zdobytych punktów doświadczenia. Przechowywanymi danymi rangi są: nazwa, opis, awatar, liczba punktów potrzebna do osiągnięcia danej rangi. System będzie również przechowywać informację o tym, kiedy ranga została przypisana użytkownikowi.

#### 2.3.1.3 POTRAWY

System powinien przechowywać informacje o potrawach. Przechowywanymi danymi potrawy są: nazwa, opis przygotowania, liczba kalorii, czas przygotowania, poziom trudności, wielkość (dla ilu ludzi), średnia ocena. Lista składników powinna być dostępna dla każdej potrawy. Każdej potrawie powinna zostać przypisana kategoria (np. „zupa”, „deser”). Powinno być możliwe dodanie zdjęcia potrawy (czas dodania, użytkownik dodający, status – zaaprobowany lub nie przez administratora - powinny być znane). Użytkownik dodający daną potrawę, jak również czas i data dodania powinny być przechowywane w systemie. Potrawy mogą być zablokowane przez administratora.

---

#### 2.3.1.4 SKŁADNIKI

System powinien przechowywać informację o składnikach. Przechowywanymi danymi składników są: nazwa i jednostka.

---

#### 2.3.1.5 KOMENTARZE

Komentarze mogą być dodawane do potraw. System powinien przechowywać informację o tym kto i kiedy dodał dany komentarz. Komentarze mogą być zablokowane przez administratora.

---

#### 2.3.1.6 OCENY

Oceny (1 do 5) mogą być dawane potrawom. System powinien przechowywać informacje kiedy ocena została dodana do potrawy i przez kogo (dany użytkownik może tylko raz ocenić daną potrawę).

---

#### 2.3.1.7 OSIĄGNIĘCIA

Osiągnięciami są na przykład: „dodanie potrawy”, „dodanie zdjęcia potrawy”, „otrzymanie pozytywnej oceny”. Przechowywanymi przez system danymi są: nazwa, opis, liczba punktów doświadczenia. Data przypisania danego osiągnięcia powinna być znana. Kilka osiągnięć może być przypisane jednemu użytkownikowi.

---

#### 2.3.1.8 NAGRODY

Nagrody mogą być odbierane przez użytkowników posiadających punkty doświadczenia. Danymi nagrody przechowywanymi w systemie powinny być: nazwa, opis, koszt (w punktach doświadczenia). Data redempcji, jak również adres (podany przez użytkownika), na który ma być wysłana nagroda powinny być znane.

---

#### 2.3.1.9 OBRAZY

Obrazami są zdjęcia potraw lub awatarami użytkowników lub rang. System powinien przechować informacje o statusie (zaaprobowany lub nie przez administrator – zawsze zaaprobowany dla awatara rangi), lokalizacji systemowej, jak również czasie dodania.

---

## 2.3.2 FUNKCJONALNOŚĆ SYSTEMU

---

### 2.3.2.1 GOŚĆ

Każdy gość:

- może wyszukiwać potrawy przez wpisanie składników, które posiada;
- może wyszukiwać potrawy po słowach zawartych w nazwie potrawy;
- może wyszukiwać potrawy przeglądając katalog potraw (po kategoriach);
- może przeglądać szczegóły potrawy, jak również komentarze, zdjęcia i średnią ocenę, ale tylko nick użytkownika, który dodał daną potrawę.

---

### 2.3.2.2 UŻYTKOWNIK

Każdy użytkownik:

- posiada prawa gościa;
- może dodawać własne potrawy i ich zdjęcia;
- może oceniać potrawy innych użytkowników;
- może komentować potrawy innych użytkowników;
- może dodać daną potrawę do listy potraw do zrobienia, przygotowanie potrawy jest odnotowywane przez system w momencie dodania zdjęcia przez użytkownika;
- może poprosić o dodanie nowego składnika (notyfikacja mailowa powinna zostać wysłana do administratora, administrator zdecyduje czy dodać dany składnik ręcznie);
- może edytować swój profil;
- może przeglądać profile innych użytkowników;
- może zresetować swoje hasło.

---

### 2.3.2.3 ADMINISTRATOR

Każdy administrator:

- posiada prawa użytkownika;
- może edytować i zastrzegać profile użytkowników;
- może dodawać i edytować składniki;
- może dodawać i edytować kategorie potraw;
- może aprobować wygrywane zdjęcia potraw i awatarów;
- może zablokować komentarze;
- może zablokować potrawy.

---

#### 2.3.2.4 WYMAGANIA DODATKOWE DOTYCZĄCE APROBAT

- Dodanie profilu musi być zaaprobowane przez użytkownika poprzez kliknięcie linku z kodem aktywacyjnym wysłanym na adres email.
- Dodanie potrawy nie musi być zaaprobowane przez administrator, ale administrator może zablokować daną potrawę.
- Dodanie zdjęcia potrawy musi być zaaprobowane przez administratora.
- Dodanie zdjęcia awatara użytkownika musi być zaaprobowane przez administratora.
- Dodanie komentarza do potrawy nie musi być zaaprobowane przez administratora, ale administrator może zablokować dany komentarz.

---

#### 2.3.2.5 WYMAGANIA DODATKOWE DOTYCZĄCE OSIĄGNIĘĆ, RANG I NAGRÓD

- Osiągnięcia będą przypisywane do użytkowników za określone aktywności (np. dodanie potrawy). Liczba punktów doświadczenia za dane osiągnięcie będzie dodawana do liczby punktów doświadczenia zgromadzonych do tej pory przez użytkownika.
- Osiągnięcie określonej liczby punktów doświadczenia będzie powodowało przypisanie określonej rangi danemu użytkownikowi, co wyniesie go wyżej w hierarchii użytkowników.
- Punkty doświadczenia będą mogły być wymieniane przez użytkownika na nagrody.

## 2.4 WYMAGANIA NIEFUNKCJONALNE

Lp.	Obszar	Cecha	Miara
1	<b>Wydajność</b>	Czas ładowania strony (w sekundach)	<= 10
2		Liczba jednoczesnych użytkowników	<= 50
3	<b>Lokalizacja</b>	Implementacja aplikacji w dowolnym języku	Implementacja w języku polskim i w języku angielskim
4	<b>Testowanie funkcjonalne</b>	Testy akceptacyjne dla FF/Linux i IE/Windows	100% testów wykonanych, 100% zaliczonych przypadków testowych
5	<b>Dokumentacja</b>	Dokument opisujący wymagania biznesowe, szczegóły implementacyjne i testy akceptacyjne	Sporządzony
6		Instrukcja obsługi dla użytkownika i administratora	Sporządzona
7		Instrukcja instalacji systemu	Sporządzona
8	<b>Bezpieczeństwo</b>	Hasło użytkownika/administratora	Przechowywane w bazie w formie zaszyfrowanej, o odpowiedniej złożoności
9		Połączenie dla zalogowanego użytkownika	HTTPS
10	<b>Niezawodność</b>	Dostępność systemu	24/7
11		Niedostępność po zgłoszeniu usterki (w godzinach)	<= 8
12	<b>Konserwacja</b>	Kopia zapasowa bazy danych	Przeprowadzana codziennie
13	<b>Oprogramowanie i sprzęt</b>	Przeglądarki zainstalowane na kliencie	Przynajmniej jedna z: FF 4.0+, IE 8+ z włączoną obsługą JavaScript
14		Konfiguracja serwera	Minimum: Dual Core 2x1.5 Mhz, 2048 MB RAM, 20 GB powierzchni dyskowej, Linux 3.0, JDK 1.7, Apache Tomcat 7, MySQL Server 5.5

## 2.5 WYMAGANIA DOTYCZĄCE INTERFEJSU

Interfejs użytkownika (Rysunek 2.1) zakłada istnienie 5 obszarów:

- nagłówek (1) - ma posiadać graficzne tło, przedstawiać nazwę strony, która ma być linkiem do strony domowej;
- górne menu (2) - menu poziome, które ma się składać z trzech odnośników kierujących do trzech różnych sposobów wyszukiwania dań;
- lewy pasek nawigacyjny (3) – ma posiadać różny zestaw odnośników, w zależności od tego czy odwiedzający jest gościem, użytkownikiem czy administratorem; dla administratora i użytkownika na górze tego paska ma się znaleźć: zdjęcie zuploadowane przez użytkownika (lub w przypadku jego braku - zdjęcie rangi), nick użytkownika, data jego rejestrację oraz nazwa rangi użytkownika.
- obszar roboczy (4) – będzie się zmieniać w zależności od wywołanej w danym momencie akcji;
- stopka (5) - ma zawierać informacje o prawach autorskich.



Rysunek 2.1 - projekt interfejsu użytkownika

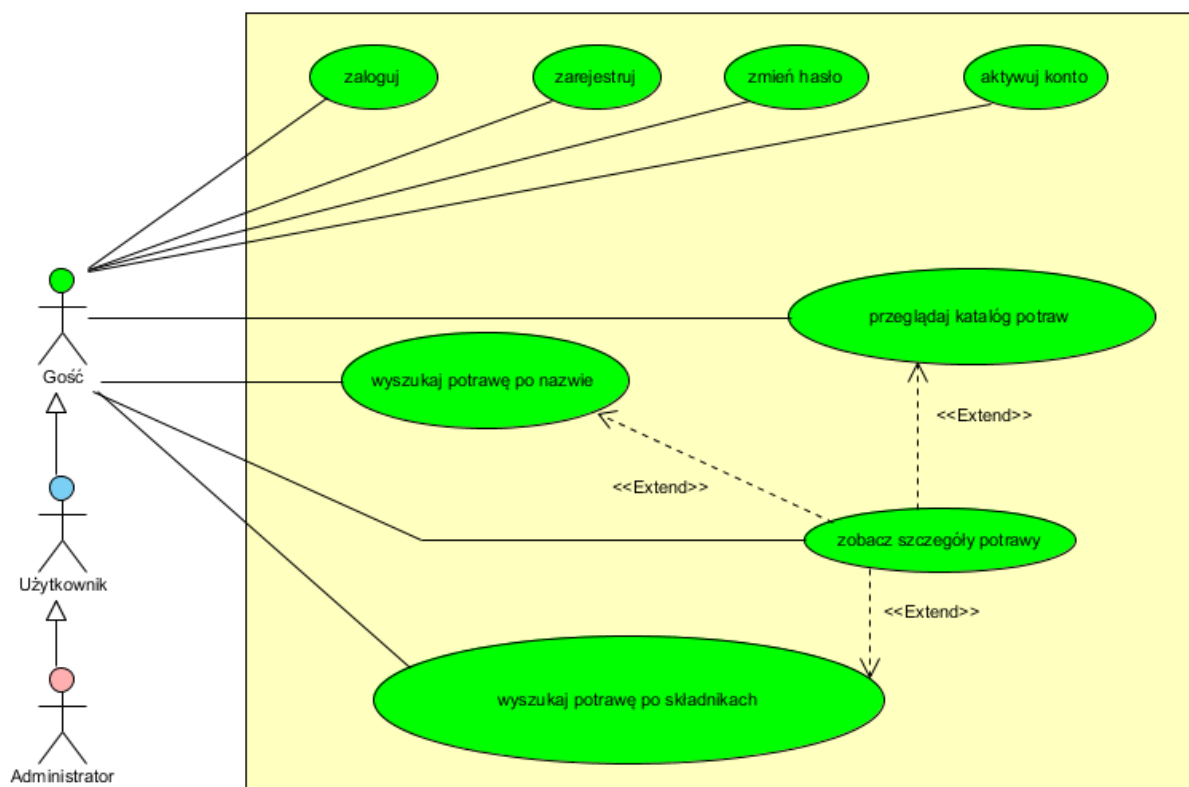
Podczas projektowania interfejsu użytkownika nacisk powinien być położony na zasady użyteczności:

- Pola formularza muszą być odpowiednio duże i dokładnie opisane,
- Błędy powstałe przy uzupełnianiu pól formularza powinny być odpowiednio widoczne – zaznaczone kolorem czerwonym,
- Tekst powinien być czytelny poprzez zastosowanie odpowiedniego doboru wysokości i koloru czcionki, koloru tła i wysokości linii,
- Odnośniki powinny się odróżniać od pozostałego tekstu,
- Link (przycisk) wylogowujący powinien się szczególnie wyróżniać.

Oprawa graficzna powinna być prosta, aczkolwiek implementacja systemu powinna pozwolić na jej znaczne urozmaicenie – zabieg ten miałby zostać osiągnięty jedynie poprzez podmianę odpowiednich plików graficznych i zmiany w arkuszach stylów.

## 2.6 DIAGRAMY PRZYPADKÓW UŻYCIA

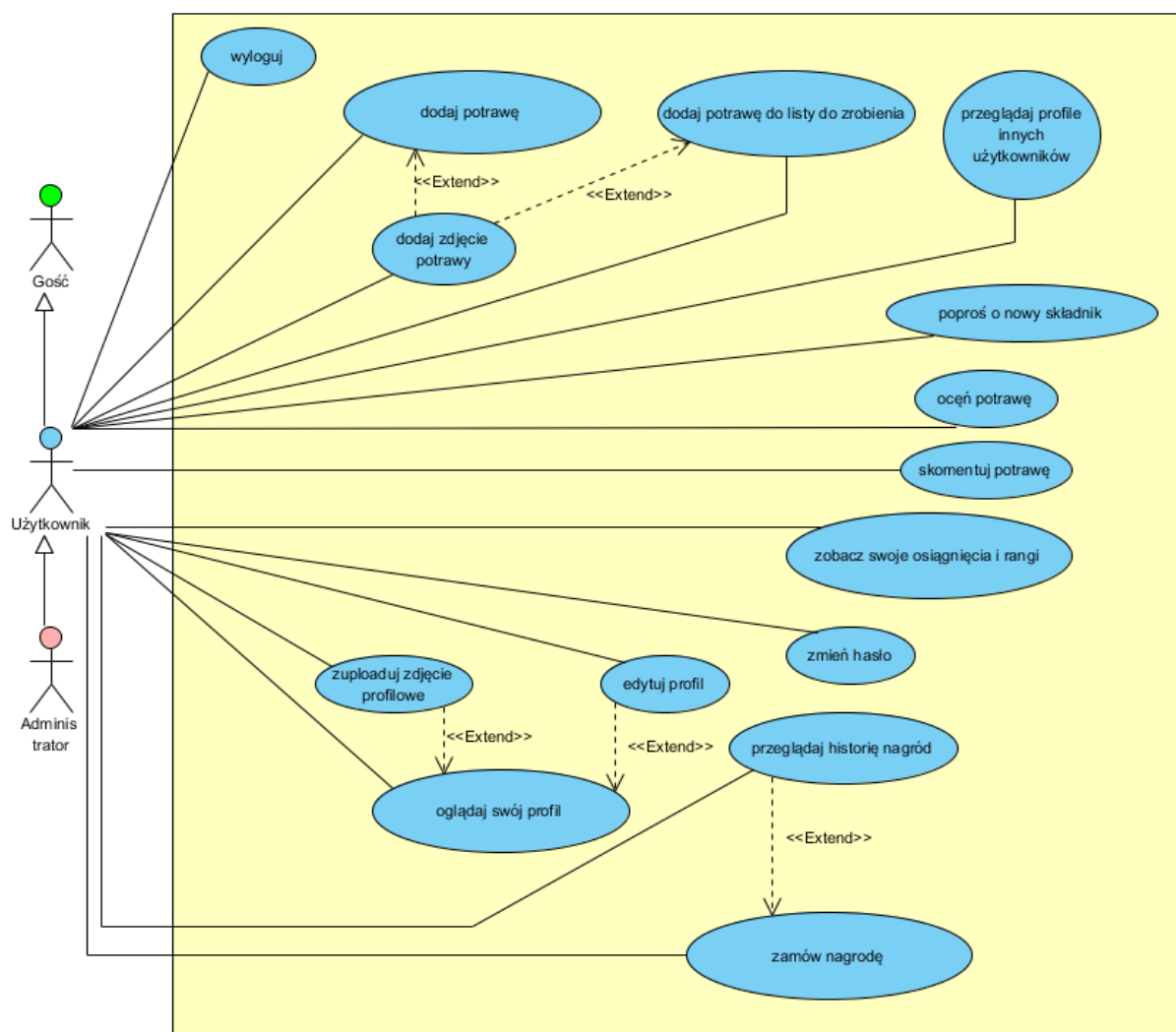
### 2.6.1 GOŚĆ



Rysunek 2.2 – diagram przypadków użycia dla gościa

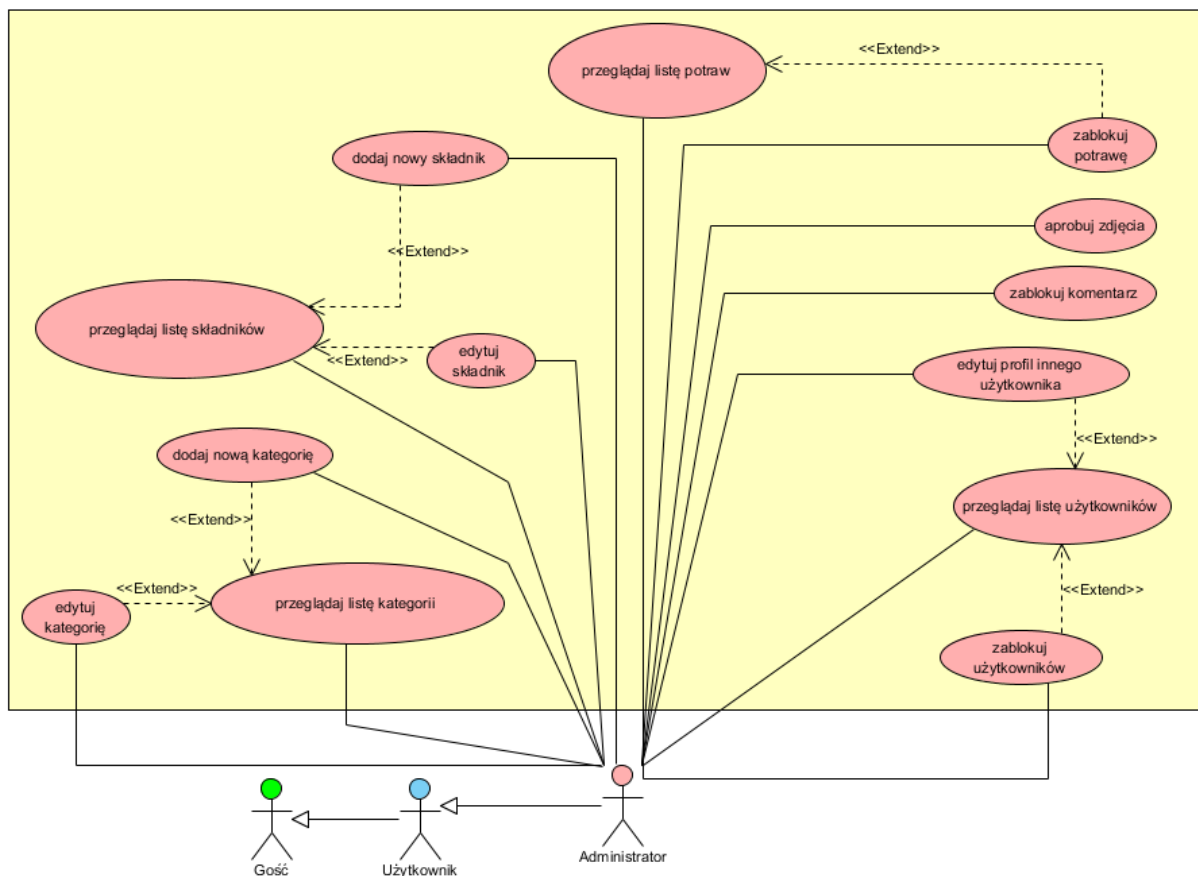


## 2.6.2 UŻYTKOWNIK



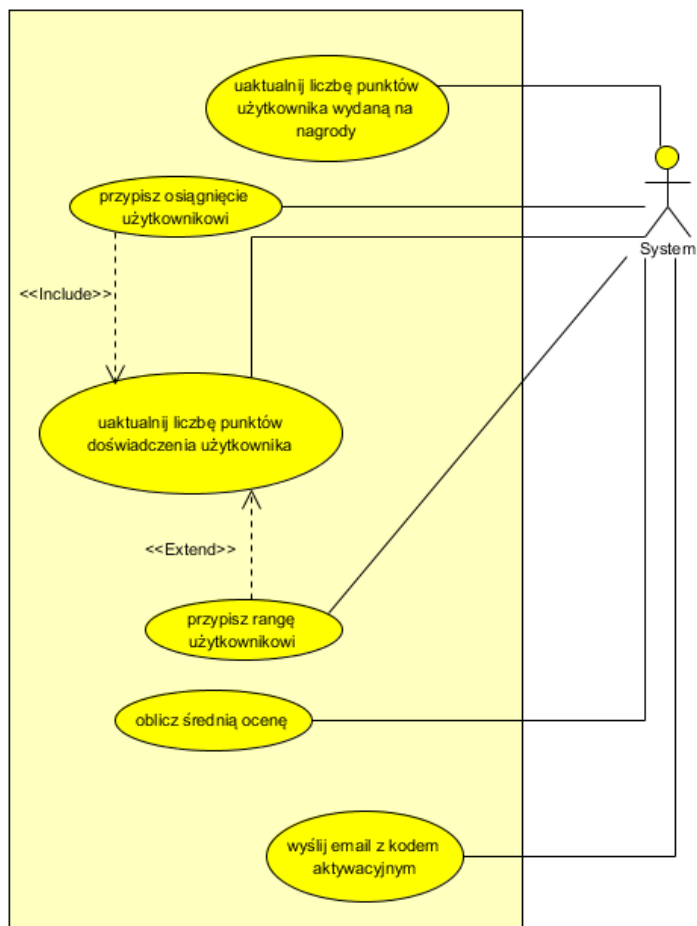
Rysunek 2.3 – diagram przypadków użycia dla użytkownika

### 2.6.3 ADMINISTRATOR



Rysunek 2.4 – diagram przypadków użycia dla administratora

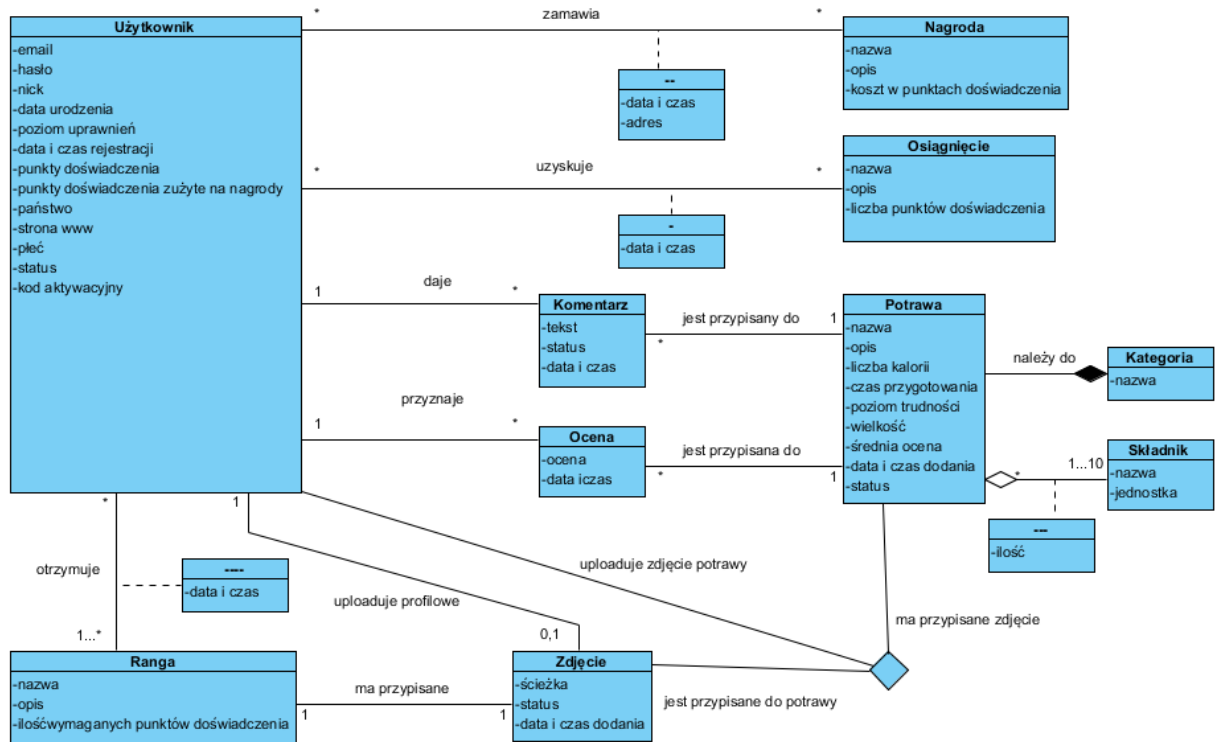
## 2.6.4 SYSTEM



Rysunek 2.5 - diagram przypadków użycia dla systemu

## 2.7 DIAGRAM KLAS

Na podstawie wymagań funkcjonalnych sporządzono następujący diagram klas:



Rysunek 2.4 - diagram klas

## 3 ZASTOSOWANE TECHNOLOGIE

### 3.1 FRAMEWORK STRUTS 2

Temat pracy narzuca wykorzystanie Struts2, dlatego opis tego frameworku zostanie przedstawiony na początku. Otóż, Struts jest open-source'owym niezależnym od platform frameworkiem używanym do tworzenia zarówno komercyjnych jak i niekomercyjnych aplikacji internetowych pisanych w Javie. Rozszerza javowe API servletów celem ułatwienia deweloperom implementacji architektury model-view-controller (MVC).

Pomimo faktu, iż Struts posiada silną konkurencję wśród javowych frameworków MVC (np. Spring MVC, Stripes, Wicket, Play!, Tapestry) niektóre z jego cech czynią go bardzo dobrym wyborem przy projektowaniu i implementacji aplikacji webowych [14]. Po pierwsze, Java jako język programowania jest już dość dojrzała, bo osiągnęła taki poziom rozwoju, w którym większość nie-biznesowej logiki istnieje w postaci bibliotek. Ponadto Struts2 umożliwia zastosowanie wtyczek (pluginów), które mogą być dołączane w razie potrzeby bez konieczności ładowania całej funkcjonalności do samego frameworku. Struts2 podąża również za popularnym ostatnio trendem „konwencja ponad konfiguracją”. Konfiguracja została wyeliminowana gdzie tylko możliwe, np. nazwy klas mogą dostarczać mapowania na akcje lub zwracane wartości rezultatów mogą dostarczać nazw stron JSP, które mają zostać wygenerowane, czy chociażby użycie javowych adnotacji zamiast plików XML. Omawiany framework bardzo dobrze radzi sobie z konwersją danych (konwersja opartych na łańcuchach znakowych wartości pól formularzy do typów prymitywnych i obiektów i vice versa). Bardzo ciekawą właściwością jest również tzw. wstrzykiwanie zależności (dependency injection), które polega na wykorzystaniu setterów celem dostarczeniu obiektowi akcji wszystkich innych obiektów, z którymi obiekt akcji musi współpracować by wykonać swoją logikę biznesową.

Opisane powyżej cechy pozwalają dobrze rokować przy implementacji systemu, a z góry narzucony framework nie powinien być traktowany jako ograniczenie a raczej jako właściwe rozwiązanie technologiczne.

## 3.2 PLATFORMA JAVA SERVLETS I JAVA SERVER PAGES

Użycie frameworku Struts2 wymusza na projekcie wykorzystanie technologii Java Servlets i Java Server Pages. Wchodzi ona w skład tzw. Java Platform, Enterprise Edition (Java EE lub Java Korporacyjna), która jest szeroko rozpowszechnioną platformą używaną w programowaniu po stronie serwera. Servlety Javy i JSP dostarczają API przede wszystkim do obsługi zapytań HTTP. Podwalinami tej technologii jest oczywiście język Java, z którego to dziedziczy ona wszystkie pozytywne cechy. Java jest bowiem prosta do nauki, pisania, kompilowania, debugowania w porównaniu z innymi językami, wykorzystuje ponadto dynamiczną alokację pamięci i tzw. „garbage collection”. Język ten jest zorientowany obiektowo, tzn. koncentruje się na tworzeniu obiektów i współpracy między nimi, co pozwala na wielokrotne wykorzystanie tego samego kodu i podejście modułowe. Jest on również niezależny od platformy (dotyczy to zarówno kodu źródłowego jak i binarnego). Łatwe przenoszenie między systemami komputerowymi jest szczególnie ważne przy tworzeniu aplikacji WWW. Java jest językiem interpretowalnym – programy pisane w Javie są kompilowane do tzw. „bytecode’u” i wymagają do uruchomienia interpretera Javy, ale kompilacja odbywa się tylko raz, a bytecode może być uruchamiany na dowolnej platformie. Ważną cechą jest również to, iż Java posiada zintegrowane zaawansowane opcje programowania sieciowego (rozproszonego). Ponadto sprawdzanie w kierunku błędów odbywa się wcześniej, na poziomie kompilatora, w przypadku innych języków podobne błędy wykrywane są dopiero po uruchomieniu. Opisując pozytywne cechy Javy nie wolno zapomnieć o bezpieczeństwo aplikacji – w przypadku Javy kompilator, interpreter, środowisko uruchomieniowe i sam język zostały zaprojektowane i stworzone zgodnie z paradygmatami bezpieczeństwa. Wielowątkowość Javy oznacza ponadto możliwość wykonywania kilka zadań w tym samym czasie w obrębie jednej aplikacji [1].

Zastosowany framework narzuca użycie technologii, której cechy wymieniono powyżej. Cechy te sugerują, iż wykorzystanie języka programowania jakim jest Java, nie jest ograniczeniem a raczej dobrym wyborem pod względem technologicznym.

### 3.3 KONTENER SERVLETÓW TOMCAT

Apache Tomcat jest kontenerem servletów stworzonym przez Apache Software Foundation. Implementuje specyfikacje Oracle Corporation: Java Servlet i JavaServer Pages i dostarcza środowiska webserverowego HTTP (napisanego również w Javie) dla aplikacji javowych.

Kontener servletów jest odpowiedzialny za uruchamianie i zarządzanie servletami. Odpowiada za [2]:

- Wsparcie dla komunikacji – np. tworzenie server socketów, nasłuchiwanie na portach, tworzenie strumieni,
- Zarządzanie cyklem życia – np. ładuje klasy servletów, inicjalizuje servlety, wywołuje metody servletów, decyduje, które instancje servletów mają ulec „garbage collection”,
- Wsparcie dla wielowątkowości – obsługa wątków dla wielu żądań,
- Bezpieczeństwo deklaracyjne – kontener obsługuje tzw. deskryptor wdrożenia (w formie pliku XMLowego), w którym można konfigurować opcje bezpieczeństwa bez konieczności umieszczania ich w kodzie klasy servletu,
- Wsparcie dla JSP – tłumaczenie JSP na kod Javy.

Oczywistym wyborem dla rozpatrywanej aplikacji, jeśli chodzi o kontener servletów był Tomcat, szeroko wykorzystywany samodzielnie lub w połączeniu z takimi serwerami aplikacji jak Apache Geronimo lub JBoss. Wybór ten był również podyktowany następującymi jego cechami: darmowy, open-source’owy, dodatkowo dla wersji 7 wykorzystanej w projekcie: aktualny w aspekcie zgodności z API servletów, zoptymalizowany, z ulepszoną prewencją i detekcją tzw. „memory leak” [6].

### 3.4 BAZA DANYCH MYSQL

MySQL jest niezależnym od platformy systemem zarządzania relacyjną bazą danych (RDBMS - relational database management system), poprzednio sponsorowanym przez szwedzką firmę MySQL AB, obecnie w posiadaniu Oracle Corporation. Starsze wersje MySQL były zgodne ze standardem SQL tylko do pewnego stopnia, w nowszych zaimplementowano obsługę wyzwalaczy (triggers), perspektyw (perspectives), kursorów (cursors), procedur składowanych (stored procedures). Aż do wersji 5.5 domyślnym silnikiem bazodanowym był MyISAM, który nie obsługiwał transakcji i referencyjnych więzów spójności, ale InnoDB, który stał się domyślny w wersji 5.5 implementuje obie te cechy – podążając tym samym za regułami ACID i większością standardów SQL [13].

MySQL 5.5 został wybrany na potrzeby projektu do przechowywania danych ze względu na swą popularność (wykorzystywany przez chociażby takie strony jak: Flickr, Nokia.com, YouTube, Wikipedia, Facebook, Twitter [13]) - a co za tym idzie globalne wsparcie, łatwe zarządzanie za pomocą aplikacji wyposażonych w interfejs graficzny (phpMyAdmin, MySQL Workbench), dobrą wydajność przy dużym stosunku odczyt/zapis (dużo więcej użytkowników przegląda potrawy w porównaniu do tych, którzy je dodają), łatwym dostępem dla aplikacji pisanych w Javie (za pomocą sterownika JDBC).

### 3.5 TECHNOLOGIE FRONTENDOWE

eXtensible HyperText Markup Language (XHTML) jest rozszerzeniem XML-owym szeroko rozpowszechnionego Hypertext Markup Language (HTML) używanego do pisania stron internetowych. Projektowana aplikacja wykorzystuje XHTML 1.0 Transitional, który zachowuje się jak HTML 4 – jest to głównie spowodowane potrzebą odpowiedniego interpretowania przez Internet Explorera, jako że dopiero od wersji 9 IE w pełni go obsługuje. Określenie *Transitional* oznacza że zawiera pewne element prezentacyjne, które zostały wykluczone w wersji strict [17]. Ta wersja XHTMLa została narzucona przez sam framework Struts2.

Kaskadowe arkusze stylów (CSS - Cascading Style Sheets) jest to język używany do opisanie semantyki prezentacyjnej (tzw. look and feel) dokumentów napisanych w (X)HTML (na potrzeby stron internetowych) lub w innych językach znacznikowych. Projekt jest oparty



na CSS 2.1, który jest wspierany przez wszystkie współczesne przeglądarki [4]. CSS w wersji 3 został użyty tylko w celu zaokrąglenia rogów elementów blokowych DIV (niewspierane w IE9).

JavaScript jest opartym na prototypach, dynamicznym, słabo typowanym językiem skryptowym z pierwszo-klasowymi funkcjami, stworzonym przez firmę Netscape. Język ten jest szeroko wykorzystywany po stronie przeglądarki do interakcji z użytkownikiem. Owa interakcja jest uzyskiwana np. poprzez: walidację formularzy (przed wysyłką na serwer), odpowiednią reakcją na zdarzenia, tworzenie i dodawanie elementów DOM „w locie” [7].

JQuery jest niezależnym od przeglądarki open-source’owym frameworkiem JavaScript’owym zaprojektowanym dla łatwiejszego używania czystego JavaScriptu. Jego składnia ułatwia wybór elementów DOM, tworzenie animacji, obsługę zdarzeń i zapytań ajaxowych [9][8]. Do budowy aplikacji wykorzystano JQuery w wersji 1.7.

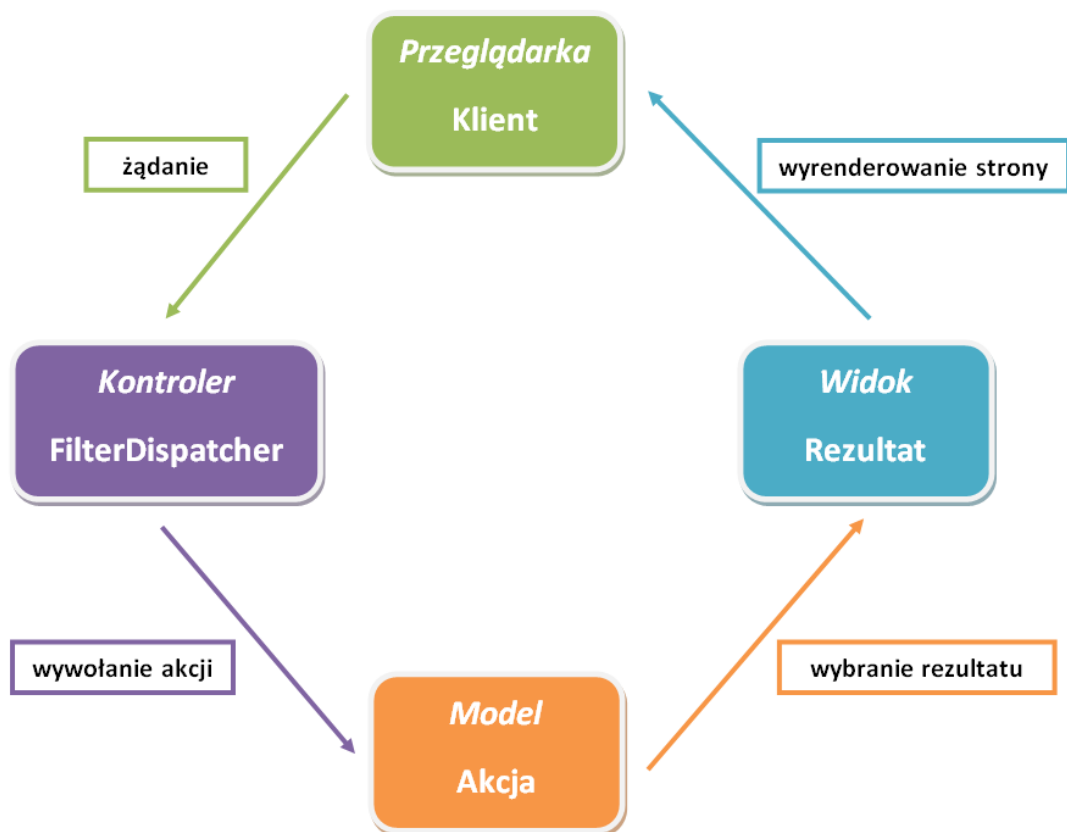
JQueryUI jest biblioteką JavaScriptową zbudowaną na podstawie JQuery stworzoną celem zaimplementowania bardziej skomplikowanych elementów interfejsu użytkownika. Składają się na nią widgety, interakcje i efekty [10]. JQueryUI w wersji 1.8.16 wykorzystano do zaimplementowania funkcji kalendarza dla pól formularzy z datą.

## 4 IMPLEMENTACJA

### 4.1 ARCHITEKTURA SYSTEMU

#### 4.1.1 WZORZEC MODEL-VIEW-CONTROLLER

Framework Struts2 podąża za wzorcem MVC. Separacja we wzorcu MVC pozwala na wdrażanie i zarządzanie dużymi skomplikowanymi aplikacjami poprzez podział na odpowiednie komponenty. We wzorcu MVC odnajdujemy następujące odrębne komponenty: model, widok (*view*) i kontroler (*controller*), które we frameworku Struts2 odpowiadają: akcji (*action*), rezultatowi (*result*) i *FilterDispatcher*’owi [3].



Rysunek 4.1 - implementacja wzorca MVC we frameworku Struts2 (1)

Struts jest często określany jako *front-controller* MVC. Jest to związane z faktem, iż kontroler jest „z przodu” odgrywając rolę pierwszego komponentu w przetwarzaniu zapytania. Jego rolą jest mapowanie zapytania na odpowiednią akcję. Tę rolę w Struts2 odgrywa *FilterDispatcher*. Obiekt ten jest właściwie filtrem servletowym, który analizuje każde dochodzące do aplikacji zapytanie i decyduje, która akcja powinna być wybrana do jego obsługi. Filtrowy jakimi jest Struts2 jest zarejestrowany w deskrypcji wdrożenia (*web.xml*) – listing 4.1:

```

1:  <filter>
2:      <filter-name>struts2</filter-name>
3:      <filter-class>
4:          org.apache.struts2.dispatcher.FilterDispatcher
5:      </filter-class>
6:  </filter>
7:  <filter-mapping>
8:      <filter-name>struts2</filter-name>
9:      <url-pattern>/*</url-pattern>
10:  </filter-mapping>

```

Listing 4.1 - rejestracja Struts2 w deskryptorze wdrożenia

Klasa filtru jest zdefiniowana w linii 4, a poprzez odpowiednie mapowanie (linia 2 i 8) jest zastosowana do żądań przychodzących ze wszystkich URLi (linia 9).

W celu przekazania frameworkowi informacji jakie URLe mapują do jakich akcji mogą być zastosowane dwa podejścia: XMLowe pliki konfiguracyjne lub Javowe adnotacje. Projekt wykorzystuje pierwsze podejście, a konkretnie XMLowy plik konfiguracyjny `struts.xml` – listing 4.2:

```

1:  <action
2:      name="Activation"
3:      class="culinary.secure.prelogin.Activation"
4:  >
5:      <result name="input" type="tiles">
6:          /secure/prelogin/Activation.tiles
7:      </result>
8:      <result name="success" type="tiles">
9:          /secure/prelogin/RActivated.tiles
10:     </result>
11: </action>
12: <action name="SendComment"
13:     class="culinary.secure.user.SendComment">
14:     <result type="json">sendComment</result>
15: </action>

```

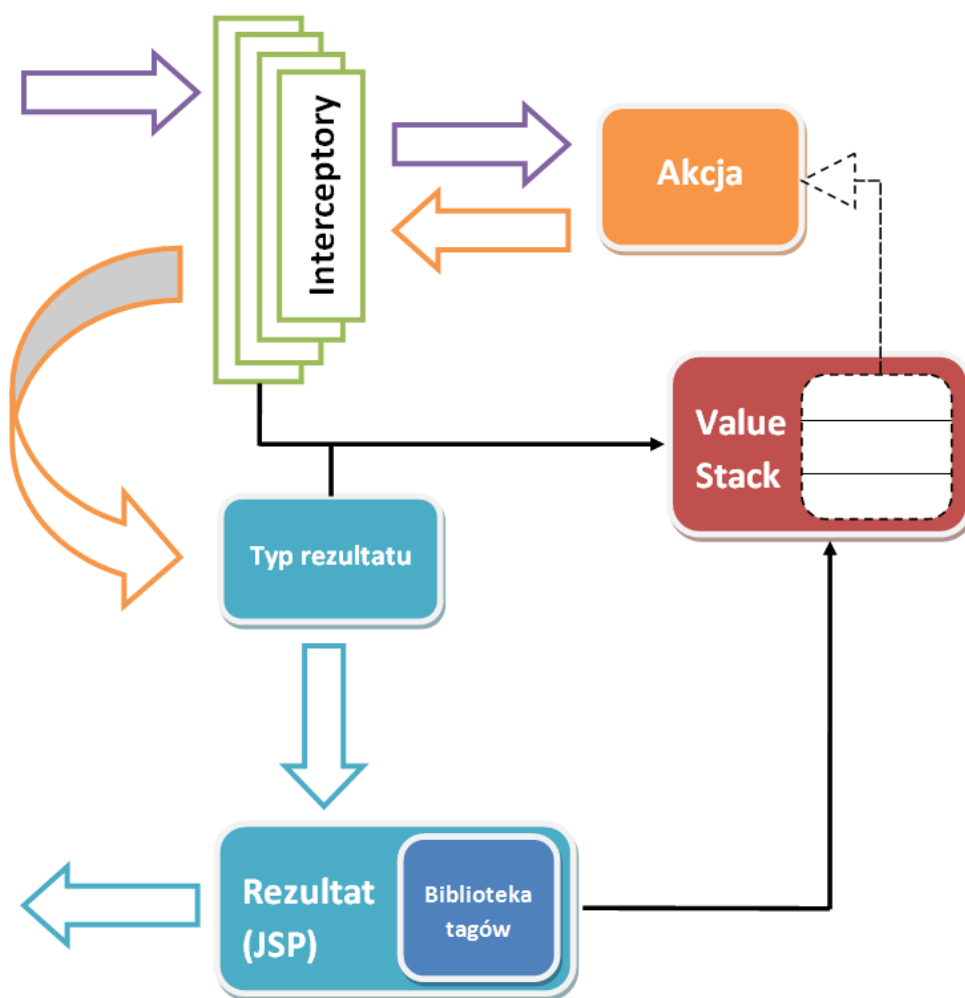
Listing 4.2 - mapowanie akcji w pliku `struts.xml`

Model w Struts2 jest zaimplementowany w postaci akcji. Używając określeń technicznych, możemy opisać model jako wewnętrzny stan aplikacji. Składa się z modelu danych (*data model*) i logiki biznesowej (*business logic*), które to łączą się ze sobą stwarzając monolityczny stan aplikacji. Jeżeli rozważymy proces logowania, zarówno dane z bazy danych jak i logika biznesowa będą wykorzystane podczas autentykacji. Metoda autentykacji będzie dostarczona przez logikę biznesową, która to porówna login i hasło wpisane przez użytkownika z danymi zapisanymi w bazie danych, w efekcie wytwarzając jeden z dwóch stanów: zautentykowany lub niezautentykowany. W chwili kiedy kontroler odnajdzie zmapowaną akcję, wywołuje ją przekazując do niej jednocześnie kontrolę nad dalszym

przetwarzaniem żądania. Wywołanie to przygotowuje zarówno odpowiednie dane, jak również wywoła logikę akcji. Kiedy praca akcji się zakończy, widok zostanie wyrenderowany i przekazany do elementu, który wysłał oryginalne żądanie.

Widok w Struts2 jest reprezentowany przez rezultat, którego zadaniem jest przetłumaczenie wewnętrznego stanu aplikacji na wizualną prezentację, z którą współdziała użytkownik. Najczęściej są to strony JSP, ale mogą to być również inne technologie prezentacyjne, jak Tiles lub XMLowe lub JSONowe rezultaty (w przypadku kiedy mamy do czynienia z żądaniami ajaxowymi). To zadaniem akcji jest wybór odpowiedniego rezultatu, zarówno jego typu (np. JSP, tile, JSON) i nazwy (np. SUCCESS, ERROR, INPUT) – linie 5, 8, 14 na listingu 4.2.

Struts2 dostarcza czystej implementacji idei MVC - aby to osiągnąć używa kilku innych kluczowych komponentów, które biorą udział w procesowaniu żądania. Są to: interceptor, ONGL i ValueStack.



Rysunek 4.2 - implementacja wzorca MVC we frameworku Struts2 (2)

Diagram na rysunku 4.2 pokazuje `FilterDispatcher`a, który wybrał już odpowiednią akcję do obsłużenia żądania, ale zanim żądanie trafi do akcji musi przejść przez stos interceptorów. Warto zwrócić uwagę, iż te same interceptory są odpalane po wykonaniu logiki akcji. Nie oznacza to jednak że za każdym razem gdy są odpalane coś robią, ale mogą. Niektóre z nich wykonują swoją pracę przed wywołaniem akcji, niektóre po. Sposób, w jaki interceptor są zaimplementowane, stwarza możliwości zastosowania wielokrotnie wykorzystywanej logiki, którą można wyłączyć z logiki samej akcji. Struts2 jest wyposażony w listę domyślnych interceptorów (np. dla walidacji i konwersji danych lub uploadu plików), ale można również samemu pisać własne interceptory. Dość zasadne jest posiadanie w swojej aplikacji interceatora, który sprawdzałby czy użytkownik może wywołać daną akcję zanim żądanie do niej dotrze, zakładając, że niektóre akcje są tylko dostępne dla zalogowanych użytkowników. Właśnie taki autentykujący interceptor został zaimplementowany na potrzeby projektu. Został on skonfigurowany w deskrytorze wdrożenia (`web.xml`) – listing 4.3:

```
1: <interceptors>
2:   <interceptor name="authenticationInterceptor"
3:     class="culinary.utils.AuthenticationInterceptor" />
4:   <interceptor-stack name="secureStack">
5:     <interceptor-ref name="authenticationInterceptor" />
6:     <interceptor-ref name="defaultStack" />
7:   </interceptor-stack>
8: </interceptors>
9: <default-interceptor-ref name="secureStack" />
```

Listing 4.3 - konfiguracja interceatora autentykującego w deskrytorze wdrożenia

Linie 2 i 3 definiują klasę interceatora autentykującego, linie 4-7 definiują nowy stos odpalanych interceptorów, który składa się z interceatora autentykującego (linia 5) i stosu domyślnych interceptorów występujących w Struts2 (linia 6). Nowo zdefiniowany stos interceptorów staje się domyślny w linii 9.

W samej klasie interceatora musimy zaimplementować metodę `intercept()` – listing 4.4:

```
1: public String intercept(ActionInvocation actionInvocation)
2:   throws Exception {
3:   Map session =
4:     actionInvocation.getInvocationContext().getSession();
5:   User user = (User) session.get("user");
6:   if (user == null) return Action.LOGIN;
7:   else {
8:     Action action = (Action) actionInvocation.getAction();
9:     if (action instanceof UserAware)
```

```

10:         ((UserAware) action).setUser(user);
11:         return actionInvocation.invoke();
12:     }
13: }

```

Listing 4.4 - implementacja metody `intercept()` w klasie interceptor autentykującego

Na początku używamy obiektu `ActionInvocation` by uzyskać referencję do obiektu (mapy) sesji (linie 3-4), z którego to wyłuskujemy obiekt użytkownika (`User`) przechowywanego pod znanym kluczem (linia 5). Jeżeli obiekt użytkownika jest null, oznacza to, że użytkownik nie został zautentykowany - co jest realizowane przez akcję logowania, która ustawia obiekt użytkownika w sesji – i żądanie jest przekierowywane do akcji logowania (linia 6). Istnienie obiektu użytkownika wskazuje na fakt, iż użytkownik przeszedł pomyślnie proces logowania. W linii 8 uzyskujemy referencję do bieżącej akcji, sprawdzamy czy implementuje interfejs `UserAware` (linia 9), a jeżeli tak, to obiekt użytkownika jest wstrzykiwany w metodę settera (linia 10), tak by był dostępny z wnętrza klasy akcji. W ten sposób każda akcja, która implementuje wspomniany interfejs będzie miała dostęp do obiektu użytkownika. W dalszej części (linia 11) przekazujemy kontrolę do pozostałych interceptorów i akcji.

`ValueStack` jest używany przez framework do przechowywania wszystkich danych aplikacji potrzebnych podczas procesowania żądania. Dane są przesuwane do `ValueStack` przy przygotowaniu do obsługi żądania, przetwarzane tam podczas wykonywania akcji i wyłuskowane stamtąd w chwili renderowania rezultatu. Object-Graph Navigation Language (ONGL) jest to język wyrażeń używany do wyłuskowania i manipulowania danymi przechowywanymi w `ValueStack`. Dane obecne w `ValueStack` są dostępne przez wszystkie fazy obsługi żądania, jako że są przechowywane w kontekście `ThreadLocal` (`ActionContext`). `ActionContext` zawiera nie tylko `ValueStack` ale również obiekt samego żądania, mapę sesji i aplikacji (z Servlet API).

By zrozumieć powyższą ideę, spójrzmy na poniższy przykład na listingu 4.5:

```

1: <s:form action="AddCategory" namespace="/secure">
2:     <s:textfield key="addeitcat.name" name="name" />
3:     <s:submit key="addeitcat.submit" />
4: </s:form>

```

Listing 4.5 - fragment strony JSP z elementami formularza

Nazwa pola formularza (linia 2) jest w rzeczywistości wyrażeniem ONGL, które jest porównywane z danymi w `ValueStack`. Obiekt akcji jest umieszczany w `ValueStack` w

chwili gdy rozpoczyna się obsługiwanie żądania i w związku z faktem, iż klasa akcji implementuje odpowiednie atrybuty i metody (zgodnie z konwencją JavaBeans), wartości z pól formularza są przekazywane do obiektu akcji, wewnątrz którego mogą być użyte – listing 4.6:

```
1: private String name;
2: public String getName() {
3:     return name;
4: }
5: public void setName(String name) {
6:     this.name = name;
7: }
```

Listing 4.6 - implementacja atrybutów i metod w akcji zgodnie z konwencją JavaBeans

Ponadto wartości te są dostępne (poprzez `ValueStack`) na stronach JSP renderowanych jako rezultat – ponownie za pomocą ONGL – listing 4.7:

```
1: <p>
2:     ...
3:     <i><s:property value="name" /></i>
4:     ...
5: </p>
```

Listing 4.7 - fragment strony JSP z użyciem ONGL

---

## 4.1.2 REZULTATY (RESULTS)

---

### 4.1.2.1 REZULTAT TILES (TILES RESULT)

Apache Tiles to framework pozwalający na uproszczenie implementacji interfejsu użytkownika poprzez zastosowanie szablonów. Plugin Struts2 Tiles wprowadza nowy typ rezultatu, który używa XML-owych definicji do renderowania stron.

Listing 4.8 pokazuje w jaki sposób skonfigurowano deskryptor wdrożenia do obsługi pluginu Tiles:

```
1: <context-param>
2:     <param-name>
3:     org.apache.tiles.impl.BasicTilesContainer.DEFINITIONS_CONFIG
4:     </param-name>
5:     <param-value>/WEB-INF/tiles.xml</param-value>
6: </context-param>
7: <listener>
8:     <listener-class>
9:     org.apache.struts2.tiles.StrutsTilesListener
10:    </listener-class>
11: </listener>
```

Listing 4.8 - rejestracja pluginu Tiles w deskrytorze wdrożenia

Użyto parametru kontekstu (linie 1-6) do określenia lokalizacji pliku z definicjami szablonów Tiles i słuchacza kontekstu servletu (linie 7-11) do integracji pluginu z frameworkiem Struts2.

W pliku `struts.xml` (głównym pliku konfiguracyjnym frameworku Struts2) pokazanym poniżej na listingu 4.9, zdefiniowano nowy typ rezultatu (linie 3-4), który może zostać użyty do wyrenderowania określonego rezultatu dla danej akcji (linie 7-9):

```
1: <package name="..." namespace="..." extends="struts-default">
2:   <result-types>
3:     <result-type name="tiles"
4:       class="org.apache.struts2.views.tiles.TilesResult" />
5:   </result-types>
6:   <action name="Register">
7:     <result type="tiles">
8:       /secure/prelogin/Registration.tiles
9:     </result>
10:  </action>
11: </package>
```

Listing 4.9 - definicja nowego typu rezultatu w pliku `struts.xml`

Plik `tiles.xml` zawiera definicje szablonów Tiles. Listing 4.10 pokazuje szczegóły implementacji:

```
1: <definition name="baseLayout"
2:   template="/layout/BaseLayout.jsp">
3:   <put-attribute name="header" value="/secure/Header.jsp" />
4:   <put-attribute name="menu_top"
5:     value="/secure/MenuTop.jsp" />
6:   <put-attribute name="sidebar_left" value="" />
7:   <put-attribute name="content"
8:     value="/unsecure/ShowSearch.jsp" />
9:   <put-attribute name="footer"
10:    value="/unsecure/Footer.jsp" />
11: </definition>
12: <definition name="/secure/admin/ EditedUser.tiles"
13:   extends="baseLayout">
14:   <put-attribute name="sidebar_left"
15:     value="/secure/SidebarLeftUser.jsp" />
16:   <put-attribute name="content"
17:     value="/secure/admin/ EditedUser.jsp" />
18: </definition>
```

Listing 4.10 - implementacja szablonów w pliku `tiles.xml`

Na początku zdefiniowano layout podstawowy (linie 1-11) wykorzystujący plik `BaseLayout.jsp`. Pozostałe szablony rozszerzają layout podstawowy (linia 13) poprzez wstawianie odpowiednich atrybutów (linie 14-17).

Fragment pliku `BaseLayout.jsp` pokazuje w jaki sposób atrybuty są wstawiane (linia 2 listingu 4.11):



```

1: <div id="menu_top">
2:     <tiles:insertAttribute name="menu_top " />
3: </div>

```

Listing 4.11 - fragment strony JSP z atrybutem Tiles

I odpowiadający plik `/secure/MenuTop.jsp` (zmapowany w `tiles.xml` w `baseLayout`) – listing 4.12:

```

1: <%@ page language="java" contentType="text/html; charset=UTF-8"
2:     pageEncoding="UTF-8"%>
3: <%@ taglib prefix="s" uri="/struts-tags"%>
4: <ul>
5:     <li>
6:         <a href="<s:url action='ShowSearch'
7:             namespace='/secure' includeParams='none' />">
8:             <s:text name="menutop.showsearch" />
9:         </a>
10:    </li>
11:    ...
12: </ul>
13: <div class="clearer"></div>

```

Listing 4.12 - plik `/secure/MenuTop.jsp`

Można zauważyć, iż jest to zwykła strona JSP z dyrektywami *page* i *taglib* (linie 1-3). Dyrektywa *taglib* pozwala użyć specyficznych dla framework Struts2 tagów (linie 6-8).

#### 4.1.2.2 REZULTAT JSON (JSON RESULT)

Obiektowa Notacja JavaScript (JSON - JavaScript Object Notation) jest używana do kompaktowej tekstowej serializacji obiektów. Jest ona coraz częściej używana jako środek komunikacji między serwerem i klientem ajaxowym. W ten właśnie sposób mogą zostać zserializowane obiekty jadowe, wysłane w odpowiedzi i zamienione na obiekty JavaScriptowe (tablice) po stronie klienta.

By uzyskać to co zostało opisane powyżej, wprowadzono odpowiednio zaprojektowany rezultat: `culinary.utils.JSONResult` – klasa, która implementuje interfejs `com.opensymphony.xwork2.Result`. Serializacja odbywa się w metodzie `execute()` tej klasy i jest wykonywana za pomocą klas z pakietów: `XStream` [18] (serializacja obiektów jadowych do XML) i `Jettison` [8] (transformacja XML do JSON).

Konfiguracja w `struts.xml` jest analogiczna do tej dla frameworku Tiles – listing 4.13:

```

1: <result-types>
2:     <result-type name="json"
3:         class="culinary.utils.JSONResult" />
4: </result-types>

```

```

5:     <action name="GetDishesPerCategory"
6:         class="culinary.secure.user.GetDishesPerCategory">
7:         <result type="json">dishes</result>
8:     </action>

```

Listing 4.13 - konfiguracja rezultatu JSON w pliku struts.xml

Przykładowa odpowiedź (na zapytanie ajaxowe) w postaci JSON:

```

{"dishes":[{"string-
array":[{"string":[1,"Dish1"]}, {"string":[3,"Dish3"]}]}]}

```

#### 4.1.3 ELEMENTY “NIEWIDOCZNEGO” (UNOBTRUSIVE) JAVASCRIPTU

Główną zasadą programowania w “niewidocznym” JavaScriptcie jest odseparowanie funkcji (tzw. warstwy zachowań) od struktury i prezentacji (tzw. markup). Zgodnie z tym podejściem w całej aplikacji nie ma ani jednego wydarzenia *onclick* dodanego do HTML-owego tagu. Użyto natomiast klas i idków dla elementów DOM celem identyfikacji i dodania odpowiednich wydarzeń [16]. Przykład poniżej na listingu 4.14 (z wykorzystaniem JQuery):

```

1:     <a href="#" class="actionLinkGetDishesPerCategory"
2:         rel='/portal/unsecure/GetDishesPerCategory.action|3'>
3:         Cat3
4:     </a>
5:     ...
6:     <script type="text/javascript" language="JavaScript">
7:         $('#categories').find(".actionLinkGetDishesPerCategory")
8:             .click(function(e) {
9:             var rel = $(this).attr('rel').split('|');
10:            var target = '#categories #trCat'+rel[1]+'_dishes';
11:            $(target).toggle('slow', function() {
12:                if($(target+'.dishesList').is(':visible')==true
13:                    && $(target+'.dishesList').text()=="")
14:                    portal.dish.model.getDishesPerCategory(rel[0],rel[1]);
15:            });
16:            e.preventDefault();
17:            e.stopPropagation();
18:        });
19:    </script>

```

Listing 4.14 - implementacja “niewidocznego” JavaScriptu

Ponadto wprowadzono wyraźniejszą separację funkcji wewnątrz samego JavaScriptu. Funkcja onclickowa (linia 8) może być traktowana jako pseudo-kontroler, który wywołuje funkcję `portal.dish.model.getDishesPerCategory()` odpowiedzialną za uzyskanie danych JSON-owych (poprzez zapytanie ajaxowe) działającą jako pseudo-model i wywołującą funkcję (pseudo-widok) `portal.dish.view.gotDishesPerCategory()`, która dokonuje zmian na stronie bez jej odświeżenia. Listing 4.15 pokazuje szczegóły implementacji tych funkcji:

```

1:   var portal = {
2:       dish : {
3:           model : {
4:               getDishesPerCategory : function(url, txt) {
5:                   ...
6:                   $.ajax({
7:                       ...
8:                       success : function(data) {
9:                           portal.dish.view.gotDishesPerCategory(data, id);
10:                        }
11:                    });
12:                }
13:            },
14:            view : {
15:                gotDishesPerCategory : function(data) {
16:                    var json = data;
17:                    var isEmpty = $.isArray(json.dishes[0]
18:                        ["string-array"][0].string);
19:                    var s = [];
20:                    if (isEmpty) {
21:                        ... json parsing ...
22:                    } else {
23:                        s.push(json.dishes[0] ["string-array"] [0].string);
24:                    }
25:                    $('#trCat' + id + '_dishes td.dishesList')
26:                        .html(s.join('\n'));
27:                }
28:            }
29:        }
30:    };

```

Listing 4.15 - implementacja separacji funkcji w JavaScriptcie

Taka implementacja pozwala na dostęp do funkcji z wykorzystaniem operatora kropki (.), co do pewnego stopnia odzwierciedla koncepcję przestrzeni nazw (namespacing) również opisywaną przy okazji „niewidocznego” programowania w JavaScriptcie.

## 4.2 MODUŁY SYSTEMU

Aplikacja została podzielona na moduły w oparciu o kryterium poziomu dostępu, co przedstawia poniższa tabela:

Lp.	UŻYTKOWNIK SYSTEMU	KATEGORIA APLIKACJI	APLIKACJE
1	GOŚĆ	ZWIĄZANE Z POTRAWAMI	<ul style="list-style-type: none"> <li>▪ Szukanie po składnikach</li> <li>▪ Szukanie po nazwie</li> <li>▪ Katalog potraw</li> <li>▪ Szczegóły potrawy</li> </ul>
2		PRE-LOGIN	<ul style="list-style-type: none"> <li>▪ Logowanie</li> <li>▪ Rejestracja</li> <li>▪ Aktywacja konta</li> <li>▪ Resetowanie hasła</li> </ul>
3	UŻYTKOWNIK	OGÓLNE	<ul style="list-style-type: none"> <li>▪ Szukanie po składnikach</li> <li>▪ Szukanie po nazwie</li> <li>▪ Katalog potraw</li> <li>▪ Szczegóły potrawy</li> <li>▪ Szczegóły użytkownika</li> </ul>
4		Z LEWEGO MENU	<ul style="list-style-type: none"> <li>▪ Dodaj danie</li> <li>▪ Dania moje/do zrobienia</li> <li>▪ Poproś o składnik</li> <li>▪ Katalog nagród</li> <li>▪ Zobacz/edytuj profil</li> <li>▪ Moje osiągnięcia i rangi</li> <li>▪ Zmień hasło</li> </ul>
5	ADMINISTRATOR	ODZIEDZICZONE PO UŻYTKOWNIKU	pkt. 3 i 4
6		DODATKOWE Z LEWEGO MENU	<ul style="list-style-type: none"> <li>• Dodaj/edytuj kategorię/składnik</li> <li>• Przeglądaj użytkowników</li> <li>• Blokuj dania</li> <li>• Aprobuj zdjęcia</li> </ul>

Taki podział został odzwierciedlony w konfiguracji w pliku `struts.xml`. Wszystkie wspomniane powyżej funkcjonalności (akcje) zostały podzielone na 4 pakiety:

Lp.	PAKIET	PRZESTRZEŃ NAZW	PROTOKÓŁ	APLIKACJE
1	<b>UNSECURE (NIEZABEZPIECZONE)</b>	/unsecure	HTTP lub HTTPS	Aplikacje gościa związane z potrawami
2	<b>PRELOGIN</b>	/secure	HTTPS	Aplikacje pre-login gościa
3	<b>USER (UŻYTKOWNIK)</b>	/secure	HTTPS	Aplikacje użytkownika
4	<b>ADMIN (ADMINISTRATOR)</b>	/secure	HTTPS	Aplikacje administratora (bez tych odziedziczonych po użytkowniku)

Użyto różnych stosów interceptorów dla różnych pakietów. Dla przykładu, pakiet User i Admin wykorzystuje `AuthenticationInterceptor`, który sprawdza czy akcje z tych dwóch pakietów są osiągalne przez zalogowanego lub niezalogowanego użytkownika, natomiast dla pakietu Admin interceptor `AuthenticationInterceptorAdmin` sprawdza czy zalogowany użytkownik próbujący dostać się do danej akcji ma przywileje administratora. Interceptory zdefiniowane dla pakietu Admina przedstawiono na listingu 4.16:

```

1: <package name="admin" namespace="/secure"
2:   extends="struts-default">
3:   <interceptors>
4:     <interceptor name="sessionInterceptor"
5:       class="culinary.utils.SessionInterceptor" />
6:     <interceptor name="authenticationInterceptor"
7:       class="culinary.utils.AuthenticationInterceptor" />
8:     <interceptor name="authenticationInterceptorAdmin"
9:       class="culinary.utils.AuthenticationInterceptorAdmin"
10:    />
11:    <interceptor-stack name="secureStackAdmin">
12:      <interceptor-ref name="sessionInterceptor" />
13:      <interceptor-ref name="authenticationInterceptor"
14:    />
15:      <interceptor-ref
16:        name="authenticationInterceptorAdmin" />
17:      <interceptor-ref name="defaultStack" />
18:    </interceptor-stack>
19:  </interceptors>
20:  <default-interceptor-ref name="secureStackAdmin" />
21: </package>

```

Listing 4.16 - interceptory zdefiniowane dla pakietu admin w pliku `struts.xml`

Również wewnętrzna konstrukcja pakietów javowych podąża za tym samym podziałem jaki jest przedstawiony w pliku `struts.xml`. Wyróżniono pakiety:

1. `culinary.unsecure`
2. `culinary.secure.prelogin`
3. `culinary.secure.user`
4. `culinary.secure.admin`
5. `culinary.utils`

Wszystkie te pakiety zawierają klasy akcji, poza `culinary.utils`, który zawiera klasy pomocnicze, takie jak:

- `swoiste interceptory`
- `JSONResult` (Rozdział 4.1.2.2)
- `PasswordIntegrityValidator` (Rozdział 4.3.1.3)
- `SendMailSSL` – używana do wysyłki maili (z kodem aktywacyjnym)
- `SHA256` – używana do zakodowania haseł użytkowników przechowywanych w bazie danych
- `User` – obiekt użytkownika przechowywany w sesji po zalogowaniu
- `SQLServletContextListener` (Rozdział 4.5.4)

## 4.3 WALIDACJA DANYCH

### 4.3.1 WALIDACJA BACKENDOWA

#### 4.3.1.1 METODA VALIDATE() AKCJI

Jednym z domyślnych interceptorów używanych w Strus2 jest interceptor `workflow`, który sprawdza czy docelowa akcja implementuje metodę `validate()` i określa w niej logikę walidującą dla danych przesyłanych jako parametry POST.

Ten rodzaj walidacji został użyty tylko dla akcji dodającej danie (`AddDish`) celem sprawdzenia czy przekazane parametry są nieujemne. Implementacja metody `validate()` na listingu 4.17:

```
1: public void validate() {
2:     if (getPreptime() < 0) {
3:         setPreptime(0);
4:         addFieldError("preptime",
5:             getText("addDish.negTime"));
6:     }
7:     ...
8: }
```

Listing 4.17 - implementacja metody `validate()` akcji

#### 4.3.1.2 METODA EXECUTE() AKCJI

Walidację w metodzie `execute()` wykorzystywano jedynie w przypadkach, kiedy istniała potrzeba konfrontacji przekazywanych danych z zawartością w bazie danych – dla akcji logującej do systemu użytkownika (`Login`) sprawdzanie czy użytkownik z danym adresem e-mail istnieje w bazie danych – listing 4.18:

```
1: public String execute() throws Exception {
2:     ResultSet rs = new SQLQuery(context.getAttribute("dbConn"))
3:         .select("SELECT * FROM USERS WHERE EMAIL='"+email+"'");
4:     if(SQLQuery.getResultSetSize(rs) == 0) {
5:         addFieldError("email",
6:             getText("login.err.email.notexists"));
7:         return INPUT;
8:     }
9:     ...
10: }
```

Listing 4.18 - implementacja metody `execute()` akcji

#### 4.3.1.3 FRAMEWORK WALIDUJĄCY

Struts2 dostarcza łatwej walidacji za pomocą interceptora `validation` (jednego z domyślnych interceptorów), który współpracuje z interceptorem `workflow` odpalonym zaraz po nim. Różnica między nimi polega na tym, że interceptor `workflow` wywołuje metodę `validate()` akcji a interceptor `validation` wykonuje walidację samodzielnie. Logika walidująca dla tego interceptora może zostać umieszczona w pliku XML (taki sposób zastosowano w projekcie) lub zaimplementowana w postaci adnotacji w samej klasie akcji. Pliki XML są nazywane od nazwy klasy, dla której wykonują pracę: *NazwaKlasyAkcji-validation.xml*.

Spójrzmy na zaimplementowaną walidację dla akcji zmiany hasła - `ChangePassword-validation.xml` (listing 4.19):

```
1: <validators>
2:   <field name="password1">
3:     <field-validator type="regex">
4:       <param name="expression">
5:         ((?=.*\d) (?=.*[a-z]) (?=.*[A-Z]) (?=.*[$!@#?]) .{6,10})
6:       </param>
7:       <message key="changepass.err.password.invalid" />
8:     </field-validator>
9:     ...
10:    <field-validator type="passwordintegrity">
11:      <param name="specialCharacters">$!@#?</param>
12:      <message key="changepass.err.password.integrity" />
13:    </field-validator>
14:    ...
15:  </validators>
```

Listing 4.19 - walidacja dla akcji zmiany hasła w pliku `ChangePassword-validation.xml`

Projekt w większości wykorzystuje walidatory pól (field validators) jak pokazano na listingu 4.19. Struts2 posiada wiele wbudowanych typów walidatorów, jak na przykład walidator `regex` (linie 3-8), który to sprawdza czy pole jest zgodne z wyrażeniem regularnym podanym jako parametr (linie 4-6).

Można również tworzyć swoje własne walidatory. Takie walidatory muszą być zarejestrowane w pliku `validators.xml` – listing 4.20:

```
1: <validators>
2:   <validator name="passwordintegrity"
3:     class="culinary.utils.PasswordIntegrityValidator"/>
4: </validators>
```

Listing 4.20 - rejestracja własnej klasy walidatora w pliku `validators.xml`



Klasa `culinary.utils.PasswordIntegrityValidator` została stworzona w celu lepszego zabezpieczenia hasła – wymusza na użytkowniku wybór hasła, które zawiera małą i dużą literę, liczbę oraz znak specjalny. Logika ta jest zaimplementowana w metodzie `validate()` tejże klasy – listing 4.21:

```
1: private String specialCharacters;
2: public String getSpecialCharacters() {
3:     return specialCharacters;
4: }
5: ...
6: static Pattern digitPattern = Pattern.compile("[0-9]");
7: static Pattern letterPattern = Pattern.compile("[a-zA-Z]");
8: public void validate(Object object){
9:     String fieldName = getFieldName();
10:    Object value = this.getFieldValue(fieldName, object);
11:    if (!(value instanceof String)) {return;}
12:    String fieldValue = ((String) value).trim();
13:    Matcher digitMatcher = digitPattern.matcher(fieldValue);
14:    Matcher letterMatcher = letterPattern.matcher(fieldValue);
15:    Matcher specialCharacterMatcher;
16:    ...
17:    Pattern specialPattern = Pattern.compile("[\"
18:                                                + getSpecialCharacters() + "\"]");
19:    specialCharacterMatcher =
20:        specialPattern.matcher(fieldValue);
21:    ...
22:    if (!digitMatcher.find()) {
23:        addFieldError(fieldName, object);
24:    } else if (!letterMatcher.find()) {
25:        addFieldError(fieldName, object);
26:    } else if (!specialCharacterMatcher.find()) {
27:        addFieldError(fieldName, object);
28:    }
```

Listing 4.21 - implementacja klasy własnego walidatora `PasswordIntegrityValidator`

Jak widzimy na listingu 4.19 (linia 11) wybór zestawu znaków specjalnych pozostawiono do definicji w pliku konfiguracyjnym XML. Zestaw ten zostaje wstrzyknięty do obiektu klasy walidującej za pomocą settera – listing 4.21, linie 2-4. Za pomocą pól statycznych zdefiniowano wzorce wyrażeń regularnych, które zostaną użyte w metodzie walidującej – linie 6-7 listingu 4.21. Następnie pobierana jest wartość pola za pomocą wywołania kilku pomocniczych metod – linie 9-12. W liniach 13-20 budujemy odpowiednie `Matcher`'y i weryfikujemy pole (hasło) w kierunku zdefiniowanych znaków. Jeżeli hasło nie posiada określonych znaków, generujemy błąd i dodajemy go do zestawu przechowywanych błędów. Jeżeli workflow walidujący odnajdzie owe błędy, przekieruje użytkownika do strony z polami do uzupełnienia z odpowiednimi alertami błędu.

### 4.3.2 WALIDACJA FRONTENDOWA

W związku z faktem, iż framework Struts2 jest wyposażony w mocne mechanizmy walidacji backendowej, walidacja frontendowa została w dużej mierze ograniczona. Najczęściej polegała ona na nadpisaniu metody wysyłającej formularz za pomocą JQuery – listing 4.22:

```
1:    $('form#AddDish').submit(function() {
2:    ...
3:        if ($('#AddDish_category option:selected').val() == "-1"){
4:            $('#AddDish_category')
5:                .css('border-color','red').css('border-width','3px');
6:            $('#AddDish_category').focus();
7:            return false;
8:        }
9:    ...
10:    return true;
11:    });
```

Listing 4.22 - nadpisanie funkcji formularza submit() za pomocą JQuery

Walidacja po stronie klienta została również użyta przed wywołaniem zapytań ajaxowych – listing 4.23:

```
1:    sendComment : function(url, id, comment) {
2:    ...
3:        if (comment == '') {
4:            $('#ShowDish_comment').css('border', '3px solid red');
5:            $('#ShowDish_comment').focus();
6:            return false;
7:        }
8:        var intRegex = /^[^\r\nąćęłńóśźżĄĆĘŁŃÓŚŹŻa-z ,.A-Z0-9_-]*$/;
9:        if (!intRegex.test(comment)) {
10:            $('#ShowDish_comment_error').show();
11:            return false;
12:        }
13:    ...
14:        $.ajax({...});
15:    }
```

Listing 4.23 - walidacja w JavaScriptcie przed wysłaniem zapytania ajaxowego

## 4.4 LOKALIZACJA APLIKACJI

Dzięki znakomitej obsłudze dla internacjonalizacji Struts2 w bardzo prosty sposób pozwala zaimplementować aplikację w dowolnym języku. `ResourceBundle` wykorzystywany w projekcie został zdefiniowany w pliku `struts.properties`, który to posiada również inne ustawienia konfiguracyjne:

```
1: ...
2: struts.custom.i18n.resources=phrases
3: ...
```

Listing 4.22 – plik konfiguracyjny `struts.properties` z definicją zasobu ze zlokalizowanymi frazami

Ponadto utworzono 2 pliki ze zlokalizowanymi frazami: polski (domyślny) i angielski - odpowiednio `phrases.properties` i `phrases_en.properties`.

```
1: ...
2: login.h3=Logowanie
3: login.h4=Wypełnij formularz
4: login.email=Podaj e-mail
5: login.password=Podaj hasło
6: login.submit=Zaloguj
7: ...
```

Listing 4.23 – fragment `phrases.properties`

```
1: ...
2: login.h3=Login
3: login.h4=Fill in:
4: login.email=E-mail
5: login.password=Password
6: login.submit=Login
7: ...
```

Listing 4.26 – fragment `phrases_en.properties`

W plikach tych znajdują się pary klucz-wartość. Klucza używa się do wyłuskiwania wartości. Na stronach JSP odbywa się to w następujący sposób:

```
1: <h4><s:text name="login.h4" /></h4>
2: <s:form action="PerformLogin" namespace="/secure">
3:   <s:textfield key="login.email" name="email" />
4:   <s:password key="login.password" name="password" />
5:   <s:submit key="login.submit" />
6: </s:form>
```

Listing 4.27 – fragment `Login.jsp`

Lokalizacji można poddać również komunikaty błędów walidacyjnych zdefiniowanych w plikach XML-owych (rozdział 4.3.1.3):

```

1:  ...
2:  <field-validator type="requiredstring">
3:    <message key="addcategory.err.name.required" />
4:  </field-validator>
5:  ...

```

Listing 4.28 – fragment AddCategory-validation.xml

W klasach akcji wykorzystuje się metodę `getText()` nadklasy `ActionSupport`:

```

1:  ...
2:  if (getCategory()<0) {
3:    addFieldError("category", getText("addDish.selectCat"));
4:  }
5:  ...

```

Listing 4.29 – fragment AddDish.java

W obsłudze internacjonalizacji uczestniczy interceptor `i18n` wchodzący w skład domyślnego stosu interceptorów. Jeżeli żądanie HTTP zawiera parametr `request_locale`, wtedy jego wartość jest zapisywana w `ActionContext`, nadpisując w ten sposób domyślny język ustalony na podstawie nagłówków HTTP wysyłanych przez przeglądarkę użytkownika. Interceptor `i18n` w momencie odnalezienia parametru `request_locale` robi coś więcej, a mianowicie celem zachowania wyboru języka ustawia atrybut sesyjny pod kluczem `WW_TRANS_I18N_LOCALE`. Atrybut ten może być zmieniony za pomocą parametru `request_locale` lub programowo.

W aplikacji projektowej język interfejsu może być zmieniony za pomocą dołączenia do adresu URL `request_locale=pl` lub `request_locale=en`. Pierwsza wartość spowoduje zaciągnięcie fraz z pliku `phrases.properties` (w związku z brakiem dedykowanej wersji `phrases_pl.properties`) a druga – `phrases_en.properties`. W zależności od tego w jakim języku będzie wypełniana baza danych, język interfejsu może być adekwatnie dostosowany.

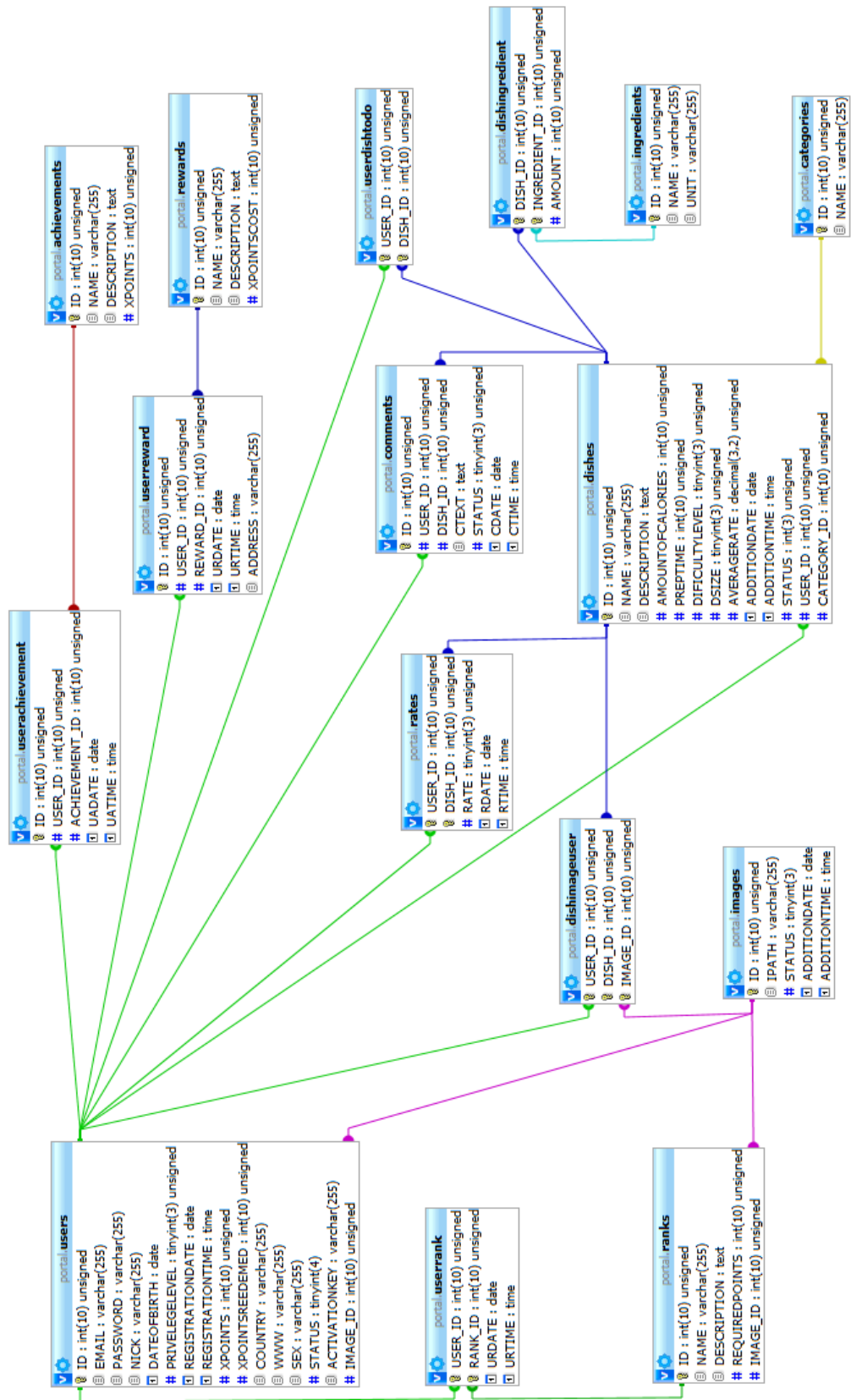
## 4.5 IMPLEMENTACJA WARSTWY DANYCH

### 4.5.1 TŁUMACZENIE DIAGRAMU KLAS NA MODEL RELACYJNY

Przetłumaczenie modelu obiektowego na relacyjny to etap, który poprzedza właściwą konstrukcję struktury bazy danych. Diagram klas, który został skonstruowany w fazie analizy musi zostać przełożony na uboższy projektowy model relacyjny, który bezpośrednio zostanie wykorzystany do określenia tabel i więzów integralności między nimi [15].

Klasy z diagramu skonstruowanego w rozdziale 2.7 zostały zamienione na tabele, atrybuty klas stały się kolumnami. Asocjacje binarne postaci 1 - \* stały się powiązaniem między tabelami w taki sposób, że klucz główny z klasy (tabeli) z licznością 1 stał się kluczem obcym w tabeli (klasie) z licznością \*. Asocjacje z atrybutem zostały zamienione dwiema asocjacjami 1 - \*. Pomędzy oryginalnymi klasami (tabelami) wprowadzono klasę pośredniczącą, w której umieszczono klucze obce. Podobnie przekształcono asocjację 3-arną pomiędzy klasami: Użytkownik, Zdjęcie, Potrawa wprowadzając klasę pośredniczącą z kluczami obcymi i 3 asocjacje binarne 1 - \*. Z obecnymi na diagramie klas agregacją i kompozycją postąpiono dokładnie tak samo jak w przypadku asocjacji. Otrzymany implementacyjny diagram struktury bazy danych przedstawiono w rozdziale 4.5.2.

## 4.5.2 STRUKTURA BAZY DANYCH



Rysunek 4.3 - diagram struktury bazy danych

#### 4.5.3 WYZWALACZE BAZODANOWE JAKO ELEMENTY LOGIKI BIZNESOWEJ

Wyzwalcz (trigger) jest to procedura wykonywana automatycznie jako reakcja na określone zdarzenie w tabeli bazy danych. Ta właściwość bazy danych MySQL została wykorzystana do implementacji pewnej części logiki biznesowej, zgodnie z tabelą poniżej:

Lp.	WYZWALACZ	TABELA	LOGIKA
1	AFTER INSERT	RATES (OCENY)	Wyliczenie średniej oceny dla danej potrawy jest inicjowane po insercji nowego wpisu do tabeli ocen (RATES) – w chwili, gdy użytkownik ocenia daną potrawę. Średnia ocena dania zostaje uaktualniona wyliczoną wartością.
2	AFTER INSERT	DISHES (DANIA)	Po insercji nowego wpisu do tabeli dań (DISHES) – w chwili, gdy użytkownik dodaje nowe danie – odpowiedni rodzaj osiągnięcia zostaje przypisany użytkownikowi (nowy wpis zostaje wstawiony do tabeli USERACHIEVEMENT).
3	AFTER UPDATE	IMAGES (OBRAZY)	W chwili, gdy dany obraz (avatara lub dania) zostaje zaakceptowany przez administrator (pole STATUS uaktualnione w tabeli IMAGES), odpowiednie osiągnięcie zostaje przypisane użytkownikowi, który dodał dane zdjęcie – nowy wpis zostaje wstawiony do tabeli USERACHIEVEMENT.
4	AFTER INSERT	USERACHIEVEMENT	W chwili, gdy nowe osiągnięcie zostaje przypisane użytkownikowi (co jest realizowane przez wyzwalacze w podrozdziałach 4.7.2 i 4.7.3), odpowiednia liczba punktów doświadczenia (charakterystyczna dla danego osiągnięcia) zostaje dodana do liczby punktów doświadczenia już posiadanych przez danego użytkownika – pole z punktami doświadczenia w tabeli USERS zostaje uaktualnione.
5	AFTER UPDATE	USERS (UŻYTKOWNICY)	1) W chwili, gdy użytkownik aktywuje swoje konto (pole STATUS w tabeli USERS zostaje uaktualnione z 0 do 1) najniższa ranga zostaje mu przypisana – nowy wpis jest wstawiany do tabeli USERRANK, 2) W chwili, gdy zostaje uaktualniona liczba punktów dla danego użytkownika (realizowane przez wyzwalacz opisany w podrozdziale 4.7.4) dokonywane jest sprawdzenie czy nowa liczba punktów doświadczenia jest wystarczająca, by przypisać użytkownikowi wyższą rangę – jeśli tak, to nowy wpis jest wstawiany do tabeli USERRANK.
6	AFTER INSERT	USERREWARD	Po tym jak nowy wpis zostaje dodany do tabeli USERREWARDS (w chwili, gdy użytkownik zamawia nagrodę), liczba punktów przeznaczona na nagrodę jest dodawana do liczby punktów doświadczenia wykorzystanych przez użytkownika na redempcję – odpowiednie pole w tabeli USERS zostaje uaktualnione.

#### 4.5.4 DOSTĘP DO BAZY DANYCH

Dostęp do bazy danych jest realizowany za pomocą `ServletContextListener` zarejestrowanego w deskrytorze wdrożenia. Słuchacz ten jest w stanie zareagować na dwa kluczowe wydarzenia w życiu kontekstu servletu – inicjalizacji i destrukcji [2].

Zaimplementowano klasę `culinary.utils.SQLServletContextListener`. Obiekt tej klasy jest notyfikowany w momencie inicjalizacji kontekstu – wyłuskuje wtedy parametry inicjalizujące kontekstu (nazwę użytkownika i hasło do bazy danych oraz sterownik bazy danych), używa ich w celu nawiązania połączenia z bazą danych i zapisuje to połączenie jako atrybut kontekstu (pod nazwą `"dbConn"`), tak by było ono dostępne w obiekcie akcji; oraz w momencie destrukcji kontekstu – zamyka wtedy połączenie z bazą danych.

Obsługa zapytań SQL-owych jest realizowana za pomocą klasy `SQLQuery`, stworzonej tylko w tym celu, zaopatrzonej w metody: `select()`, `insert()`, `update()` i `delete()`. Użycie tej klasy w klasie akcji pokazano na listingu 4.30:

```
1:   ResultSet rs = new
2:   SQLQuery(context.getAttribute("dbConn"))
3:   .select("SELECT * FROM USERS WHERE EMAIL='"+email+"'");
```

Listing 4.30 - przykładowe użycie klasy `SQLQuery`

Implementację metody `select()` w klasie `SQLQuery` przedstawiono poniżej na listingu 4.31:

```
1:   public ResultSet select(String query) throws SQLException {
2:       return stmt.executeQuery(query);
3:   }
```

Listing 4.31 - implementacja metody `select()` w klasie `SQLQuery`

Użyte do połączenia z bazą danych rozwiązanie wydaje się być bardzo zasadne, zważywszy na fakt, iż jest tylko jeden `ServletContext` dla całej aplikacji webowej i wszystkie komponenty mogą go współdzielić.



## 4.6 IMPLEMENTACJA INTERFEJSU

Interfejs został zaimplementowany zgodnie z wymaganiami określonymi w rozdziale 2.5 – rysunek 4.4. Określone obszary: nagłówek (1), górne menu (2), lewy pasek nawigacyjny (3), obszar roboczy (4), stopka (5) zbudowano w oparciu o HTML-owe bloki DIV z odpowiednim formatowaniem CSS-owym zdefiniowanym w pliku `/portal/css/style.css`.

```
1: <head>
2:   <link rel="stylesheet" href="/portal/css/style.css"
3:                                             type="text/css" />
4: </head>
5: <body>
6:   <div id="container">
7:     <div id="header" />
8:     <div id="menu_top" />
9:     <div id="sidebar_left" />
10:    <div id="content" />
11:    <div id="footer">
12:      </div>
13:  </div>
```

Listing 4.32 – struktura HTMLowa portalu



Rysunek 4.4 - implementacja interfejsu użytkownika

## 5 WDROŻENIE I ROZWÓJ

### 5.1 TESTOWANIE APLIKACJI

#### 5.1.1 TESTY FAZY DEVELOPMENTU

Podczas fazy developmentu zostało przeprowadzone testowanie białej skrzynki, zwane również testowaniem szklanej skrzynki. Jest to metoda testowania oprogramowania bazująca na wiedzy o wewnętrznej strukturze aplikacji. Najlepszą praktyką jest, by było ono przeprowadzane podczas fazy kodowania przez samych programistów. Może być ono przeprowadzane również później, podczas fazy testów, ale wymaga wtedy od testera wiedzy programistycznej przy przygotowaniu przypadków testowych – jest to oczywisty wzrost kosztów projektu [11].

W przypadku opisywanej aplikacji testy wykonał sam programista. Każdy moduł aplikacji został poddany tzw. wyczerpującej ścieżce testowania (*exhaustive path testing*) – wszystkie możliwe ścieżki przebiegu kontroli programu zostały sprawdzone [12]. Wykryte błędy zostały niezwłocznie naprawione.

#### 5.1.2 TESTY AKCEPTACYJNE

Testy akceptacyjne to jeden z rodzajów testowania czarnoskrzynkowego, które w prostych słowach traktuje program jak czarną skrzynkę. Funkcje są testowane w ten sposób, że dostarcza się do programu dane (*input*) i obserwuje się jakie dane wychodzą z programu (*output*). Stąd testowanie czarnej skrzynki jest również zwane testowaniem napędzanym wejściem/wyjściem (*input/output-driven testing*). Nie zwraca się uwagi na wewnętrzne zachowanie i strukturę programu. Celem jest znalezienie takich okoliczności, w których program zachowuje się niezgodnie z założeniami zdefiniowanymi w specyfikacji. [11][12]

Testowanie akceptacyjne zazwyczaj przeprowadzane jest jako ostatnie przed wdrożeniem/dostarczeniem systemu [11]. Jego celem jest porównanie oprogramowania z pierwotnymi wymaganiami biznesowymi [12].

W celu przeprowadzenia testów akceptacyjnych przygotowano listę przypadków testowych (*test cases*) bazując na kryterium testowania wyczerpującego wejścia (*exhaustive input testing*), tak by wykorzystać każdą wejściową okoliczność jako przypadek testowy – tabela poniżej:

Lp.	Subaplikacja	Dane uwierzytelniające	Przypadek testowy
1	Wyszukiwanie po składniku (niezabezpieczone)	-	Wprowadź różną kombinację składników, kliknij szukaj, sprawdź składniki wyszukanych potraw
2	Wyszukiwanie po nazwie (niezabezpieczone)	-	Wpisz różne nazwy, kliknij szukaj, jeżeli potrawy zostały wyszukane sprawdź czy ich nazwy zawierają szukaną frazę
3	Katalog dań (Niezabezpieczone)	-	Przeglądaj potrawy w katalogu potraw, po wejściu w szczegóły potrawy sprawdź czy jest we właściwej kategorii
4	Szczegóły dania (niezabezpieczone)	-	Przeglądaj potrawy w katalogu potraw, po wejściu w szczegóły potrawy sprawdź czy jest we właściwej kategorii, sprawdź pozostałe szczegóły potrawy
5	Rejestracja	-	Rejestracja nowego użytkownika – cały flow
6	Aktywacja konta	-	Aktywacja konta – użyj klucza wysłanego w 5.
7	Logowanie	-	Zaloguj wykorzystując dane z 5.
8	Resetowanie hasła	-	Zresetuj hasło, a potem zaloguj się nowym hasłem wysłanym w mailu
9	Wyszukiwanie po składniku (zabezpieczone)	Użytkownik	Wprowadź różną kombinację składników, kliknij szukaj, sprawdź składniki wyszukanych potraw
10	Wyszukiwanie po nazwie (zabezpieczone)	Użytkownik	Wpisz różne nazwy, kliknij szukaj, jeżeli potrawy zostały wyszukane sprawdź czy ich nazwy zawierają szukaną frazę
11	Katalog dań (zabezpieczone)	Użytkownik	Przeglądaj potrawy w katalogu potraw, po wejściu w szczegóły potrawy sprawdź czy jest we właściwej kategorii
12	Szczegóły dania (zabezpieczone)	Użytkownik	Sprawdź czy ekran potrawy zawiera sekcje zabezpieczone
13	Dodaj danie/Moje dania	Użytkownik	Wypełnij formularz dodawania potrawy kilkakrotnie z różnymi danymi, sprawdź zachowanie (walidację), przejdź do Moich potraw i sprawdź czy lista potraw jest kompletna
14	Moje dania/Dania do zrobienia	Użytkownik	Przeglądaj potrawy, dodaj nie swoje potrawy do listy potraw do zrobienia, sprawdź czy zostały wpisane na listę potraw do zrobienia w aplikacji Dania moje/do zrobienia
15	Moje dania/ Dania do zrobienia/ Aprobuj zdjęcia	Użytkownik / administrator	Dodaj zdjęcia potraw (swoich jak i z listy do zrobienia), zaloguj się jako administrator, zaaprobuj zdjęcie, zaloguj się jako wcześniejszy użytkownik, sprawdź ekrany potraw czy zdjęcia a pokazują się
16	Poproś o składnik	Użytkownik	Poproś o nowy składnik, sprawdź maila administrator czy dostał notyfikację z prośbą
17	Katalog nagród	Użytkownik	Zamów nową nagrodę, sprawdź historię zamówień, sprawdź czy liczba punktów została uaktualniona
18	Zobacz/edytuj profil	Użytkownik	Obejrzyj swój profil, edytuj go i zapisz, sprawdź czy zmiany są widoczne
19	Zobacz/edytuj profil / Aprobuj zdjęcia	Użytkownik / administrator	Obejrzyj swój profil, zuploaduj zdjęcie, zaloguj się jako administrator, zaaprobuj zdjęcie, zaloguj się jako wcześniejszy użytkownik, sprawdź czy nowe zdjęcie jest widoczne (w lewym pasku nawigacyjnym zamiast zdjęcia rangi)
20	Moje osiągnięcia/ Moje rangi	Użytkownik	Wejdź do aplikacji Moje osiągnięcia/rangi, sprawdź czy liczba naliczonych punktów zgadza się z wcześniejszymi czynnościami (dodanie potrawy, upload zdjęcia i aprobata), sprawdź czy ranga jest przypisana adekwatnie do uzyskany punktów doświadczenia
21	Zmień hasło	Użytkownik	Zmien hasło, wyloguj się, spróbuj zalogować się starym i nowym hasłem
22	Dodaj/edytuj kategorię	Administrator	Obejrzyj obecne kategorie, dodaj nową i sprawdź czy lista kategorii została uaktualniona
23	Dodaj/edytuj składnik	Administrator	Obejrzyj obecne składniki, dodaj nowy i sprawdź czy lista składników została uaktualniona
24	Przeglądaj użytkowników	Administrator	Sprawdź listę użytkowników, edytuj niektórych z nich i sprawdź czy zmiany zostały odzwierciedlone

25	<b>Przeglądaj użytkowników</b>	<b>Administrator /użytkownik</b>	Sprawdź listę użytkowników, zablokuj niektórych z nich, wyloguj się i spróbuj zalogować się zablokowanym użytkownikiem
26	<b>Blokuj dnia</b>	<b>Administrator</b>	Przejdź do aplikacji Blokuj dania, zablokuj niektóre dania i spróbuj wyświetlić szczegóły zablokowanych dań
27	<b>Szczegóły użytkownika</b>	<b>Użytkownik</b>	Przejdź do ekranu ze szczegółami dania, kliknij nickname użytkownika, który dodał potrawę, sprawdź szczegóły użytkownika; zrób to samo dla różnych użytkowników

Testy zostały przeprowadzone w Firefox 8 (Ubuntu 11.10) i Internet Explorer 8 (Windows 7). Wszystkie przypadki testowe zakończyły się sukcesem. Drobne błędy layoutowe, które wykryto podczas testów, zostały naprawione.

### 5.1.3 TESTY OBCIĄŻENIOWE

Testy obciążeniowe sprawdzają odpowiedź program w warunkach zwiększonego obciążenia. Jeżeli zdefiniowane jest obciążenie, które program powinien wytrzymać, test obciążeniowy stara się sprawdzić czy program przestaje odpowiadać poniżej czy powyżej tego zdefiniowanego poziomu [11]. Aplikacje webowe są często testowane pod względem obciążenia. Testy obciążeniowe tych aplikacji mają zapewnić, że dana aplikacja, użyty sprzęt, mogą wytrzymać daną liczbę jednoczesnych użytkowników [12].

Na potrzeby testów obciążeniowych projektu została przygotowana specjalna podstrona (`/portal/stressTest.html`), która wykorzystując JavaScript wysyła zapytania ajaxowe do aplikacji post-login. Jej celem jest zasymulowanie zachowania przeciętnego użytkownika, który przechodzi z jednej subaplikacji do drugiej klikając odnośniki w lewym menu. Po załadowaniu strony JavaScript inicjuje zapytanie ajaxowe do strony logowania (z danymi administratora), dane użytkownika są prependowane do strony testującej, następnie kolejne zapytania ajaxowe są wysyłane do aplikacji, które są dostępne dla administratora (gdyby był zalogowany w normalny sposób), na końcu zapytanie jest wysyłane do strony wylogowania. Zapytania te są wysyłane w odstępach 3 sekund plus czas odpowiedzi na dane zapytanie ajaxowe (co odpowiada czasowi 3 sekund plus czas ładowania się danej strony w przeglądarce). Po tym jak jeden cykl się kończy, zaczyna się nowy, do momentu zamknięcia strony. Strona przedstawia czas inicjalizacji zapytania, nazwę zasobu (akcji) i czas odpowiedzi (Rysunek 5.1).



user: **mszonline**  
 from: **2011-12-11 20:00**  
 rank: **Kucharz Laik**

Log:

13:6:3	PerformLogin	247
13:6:7	ShowAddDish	1502
13:6:11	ShowMyDishes	186
13:6:14	ShowRequestIngredient	184
13:6:18	ShowRewards	1279
13:6:21	ShowChangePassword	180
13:6:26	ShowAddCategory	1559
13:6:29	ShowAddIngredient	221
13:6:34	ShowUsers	1492
13:6:37	ShowBlockDish	177
13:6:41	ShowApprovePhoto	1314
13:6:44	Logout.action	81
13:6:49	PerformLogin	1574
13:6:52	ShowAddDish	435
13:7:2	ShowMyDishes	6516
13:7:7	ShowRequestIngredient	1767
13:7:10	ShowRewards	212
13:7:13	ShowChangePassword	156
13:7:16	ShowAddCategory	332

Rysunek 5.1 – wizualizacja zapytań ajaxowych podczas testu obciążeniowego

W celu zasymulowania pewnej liczby jednoczesnych użytkowników został wykorzystany Internet Explorer w wersji 9, jako że ta przeglądarka może otwierać nowe okna jako nowe procesy (wystarczy użyć `iexplore.exe -noframemerging`), co pozwala na wielokrotne logowanie do aplikacji webowej.

Pięćdziesiąt okien Internet Explorera zostało otwartych z adresem `/portal/stressTest.html` za pomocą odpowiednio spreparowanego pliku batchowego (`stressTest.bat`) – obecny na płycie CD załączonej do projektu. Test trwał 30 minut. Średni czas odpowiedzi dla ostatnich 3 pełnych cykli został policzony – tabela poniżej:

Akcja	Średni czas odpowiedzi
PerformLogin	2886 (ms)
ShowAddDish	2793 (ms)
ShowMyDishes	2257 (ms)
ShowRequestIngredient	2051 (ms)
ShowRewards	2385 (ms)
ShowChangePassword	2194 (ms)
ShowAddCategory	3595 (ms)
ShowAddIngredient	3599 (ms)
ShowUsers	4670 (ms)
ShowBlockDish	2818 (ms)
ShowApprovePhoto	1982 (ms)
Logout	1439 (ms)

Sprawdzono również, iż żadna z odpowiedzi nie trwała dłużej niż 10000ms. Kryterium dotyczące wydajności (50 jednoczesnych użytkowników, którzy otwierają strony w mniej niż 10 sekund) okazało się być spełnione (Rozdział 2.4).

Konfiguracja sprzętowa dla serwera użytego podczas testu obciążeniowego przedstawia się następująco: Dual Core 2x1.5 Mhz, 2048 MB RAM, 20 GB powierzchni dyskowej. Konfiguracja ta spełnia wymaganie dotyczące minimalnej konfiguracji sprzętowej serwera przedstawionej w rozdziale 2.4.

## 5.2 REALIZACJA ZAŁOŻEŃ I MOŻLIWOŚCI ROZWOJOWE

Biorąc pod uwagę wymagania funkcjonalne i нефункционалне, udało się zaimplementować system zgodny z założeniami. Prawidłową implementację wymagań funkcjonalnych potwierdziły przeprowadzone testy akceptacyjne (Rozdział 5.1.2). Testy obciążeniowe (Rozdział 5.1.3) wykazały zgodności pod względem wymagań dotyczących wydajności, oprogramowania i sprzętu. Sporządzono również szczegółową dokumentację dotyczącą samej instalacji systemu, jak również instrukcje obsługi dla poszczególnych poziomów użytkowników (Gość, Użytkownik, Administrator) – odnajdziemy ją na płycie CD jako dodatek do niniejszej pracy.

Projekt dowiódł, iż jest możliwe stworzenie tematycznego portalu społecznościowego w oparciu o framework Struts2. Co prawda framework sam w sobie narzucił użycie określonej technologii po stronie serwera (Java), ale jednocześnie dostarczył wystarczających możliwości do implementacji systemu określonego przez wymagania charakterystyczne dla kulinarnego serwisu społecznościowego (Rozdział 2). Wybrane rozwiązanie po stronie serwera było na tyle elastyczne, by wybór bazy danych, kontenera servletów, bibliotek javascriptowych czy konstrukcji layoutu w oparciu o własny szkielet HTML i arkusze stylów pozostawić po stronie klienta/programisty.

Zaimplementowany system jest systemem kompletnym, bo zgodnym z wymaganiami biznesowymi, które były określane dla serwisu gotowego do wdrożenia produkcyjnego. Pomimo starannie przeprowadzonej analizy biznesowej zawiera on jednak kilka niedociągnięć. Pod względem funkcjonalnym brakuje bardzo istotnej funkcji edycji potrawy – w zaimplementowanym systemie nie jest to możliwe ani przez użytkownika, który dodał potrawę, ani przez administratora – administrator może jedynie zablokować potrawę. Funkcja edycji potrawy powinna zostać dodana przed otwarciem serwisu szerszemu gronu użytkowników. Implementacja tej funkcjonalności przy użyciu Struts2 jest możliwa.

Aplikacja kuleje także pod względem estetycznym. Portale kulinarne posiadają zazwyczaj kwiecistą grafikę, która ma zainteresować odbiorcę i zachęcić go do przeglądania *pięknych* potraw. Projekt nie koncentrował się jednak na tym – w tym aspekcie powinien być ulepszony. Ulepszenie to będzie jednak bardzo proste – zgodnie z wymaganiami dotyczącymi interfejsu użytkownika określonymi w rozdziale 2.5, urozmaicenie grafiki ma się odbyć jedynie poprzez podmianę odpowiednich grafik i zmiany w arkuszach stylów. Taka



konstrukcja portalu była do osiągnięcia dzięki frameworkowi Struts2, który pozostawił swobodę w tym zakresie.

Biorąc pod uwagę możliwości rozwoju systemu, na pewno polem do popisu są funkcje społecznościowe. W opracowanym portalu interakcje członków społeczności ograniczają się do przeglądania, oceniania i komentowania potraw innych oraz przeglądania średnio rozbudowanych profili użytkowników. Zaimplementowano co prawda system nagradzania za aktywność i hierarchię użytkowników, ale zabrakło np. korespondencji prywatnej, miejsc do dyskusji na tematy nie związane z tematyką portalu, ankiet, sond, chatroomów. Funkcje te na pewno zwiększyłyby poczucie przynależności do społeczności. Wstępna analiza ekspercka pokazała, iż funkcjonalności te dzięki użytemu frameworkowi mogą być włączone do serwisu dość sprawnie.

Reasumując można stwierdzić, iż implementacja kulinarnego portalu społecznościowego z wykorzystaniem frameworku Struts2 była stosunkowo łatwa i intuicyjna. Wiele wbudowanych funkcji frameworku w szybki i skuteczny sposób umożliwiło przełożenie wymagań biznesowych na produkt programistyczny. Możliwości jakie stwarza framework Struts2 pozwalają również optymistycznie patrzeć w kierunku rozwoju portalu. Przy małych modyfikacjach (poprawa strony wizualnej i dodanie funkcji edycji potrawy) system jest gotowy do wdrożenia produkcyjnego.



- [1] Advantages of Java  
<http://www.webdotdev.com/nvd/content/view/1042/204/>
- [2] Basham Bryan, Sierra Kathy, Bates Bert, Head First Servlets & JSP, Second Edition. O'Reilly Media, 2008.
- [3] Brown Don, Davis Chad Michael, Stanlick Scott, Struts 2 in Action. Manning Publications, 2008.
- [4] CSS  
<http://en.wikipedia.org/wiki/Css>
- [5] Facebook has over 845 million users  
<http://www.zdnet.com/blog/facebook/facebook-has-over-845-million-users/8332>
- [6] Is Apache Tomcat 7 in your future?  
<http://www.tomcatexpert.com/blog/2010/07/29/tomcat-7-your-future>
- [7] JavaScript  
<http://en.wikipedia.org/wiki/JavaScript>
- [8] Jettison  
<http://jettison.codehaus.org>
- [9] JQuery  
<http://jquery.com>
- [10] JQueryUI  
<http://jqueryui.com/>
- [11] Kaner Cem, Falk Jack, Nguyen Hung Quoc, Testing Computer Software, Second Edition. Wiley, 1999.
- [12] Myers Glenford J., Badgett Tom, Thomas Todd M., Sandler Corey, The Art of Software Testing, Second Edition. Wiley, 2004.
- [13] MySQL  
<http://dev.mysql.com/doc/refman/5.5/en/index.html>  
<http://en.wikipedia.org/wiki/MySQL>
- [14] Roughley Ian, Practical Apache Struts2 Web 2.0 Projects. Apress, 2007
- [15] Trzaska Mariusz, Modelowanie i implementacja systemów informatycznych. Wydawnictwo PJWSTK, 2008.
- [16] Unobtrusive JavaScript

[http://en.wikipedia.org/wiki/Unobtrusive JavaScript](http://en.wikipedia.org/wiki/Unobtrusive_JavaScript)

[17] XHTML

<http://en.wikipedia.org/wiki/Xhtml>

[18] XStream

<http://xstream.codehaus.org>

[19] Web 2.0

[http://pl.wikipedia.org/wiki/Web 2.0](http://pl.wikipedia.org/wiki/Web_2.0)

Na załączonej do pracy płycie CD umieszczono:

- Elektroniczną wersję pracy pisemnej w formacie DOCX (MS Word) oraz PDF
- Instrukcje instalacji i obsługi aplikacji w formacie DOCX (MS Word) oraz PDF
- Pliki źródłowe systemu w postaci pliku `portal.war`
- Skrypt SQL tworzący bazę danych dla projektu w postaci `initialQuery.sql`
- Plik batchowy użyty podczas testu obciążeniowego `stressTest.bat`
- Archiwum Apache Tomcat 7.0.23
- Archiwum Java Development Kit 7