AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

KATEDRA INFORMATYKI STOSOWANEJ

Praca dyplomowa inżynierska

Serwis internetowy dla studentów zaimplementowany w architekturze SOA

Internet service for students implememented in SOA architecture

Autor: *Marek Ryznar*
Kierunek studiów: *Informatyka*
Opiekun pracy: *dr inż. Mirosław Gajer*

Kraków, 2015

Oświadczam, świadomy(-a) odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Serdecznie dziękuję . . . tu ciąg dalszych podziękowań np. dla promotora, żony, sąsiada itp.

Spis treści

1. Wprowadzenie

1.1. Przedmowa - Przerobic na ladny tekst

- Cos o tym ze dawnej studenci musieli zadupcac kserowac notatki, spisywac od znajomych. No ale zaznaczyc to ze musieli zadupcac po to wszystko
- nagle buch - internet sie stal - wszystkie wymienione wyzej problemy mozna rozwiazac nie ruszajac sie z fotela.
- Gdy sie nie wie kiedy kolos - piszesz na portalu społecznościowym, forum itd. (Tutaj moze jak sie da to ladnie wstawic to przyklady portali fejs,twitter,fora itd)
-A gdy potzrebujesz notatki to albo wysyla Ci ktos mailem, albo udostepnia na dysku internetowym, co tez ulatwia zdecydowanie zycie (No i tutaj opisac dyski internetowe? ze google wiedzie prym bo ma maila i google dysk, dodatkowo napisac o innych istaniejacych rozwiazaniach (dropbox, oneDrive itd)).
- Tutaj trzeba bedzie napisac o tym ze to jest wazna rzecz w życiu studenta, ze bez tego to jak bez reki.
- Ze wszystkie opisane powyzej rozwiazania sa eksta itd, (ale wg mojego researchu?) nie ma rozwiazania dedykowanego stricte dla studentow tj portalu na ktorym mogli by wymieniac swobodnie pliki, zapisywac terminy w kalendarzu itd.
- I tutaj BUM - ja chce cos takiego zrobic
• Dlaczego autor chciał zająć się danym tematem?
Przykladowe odniesienie do bibliografii

1.2. Cel pracy

- Celem pracy jest stworzenie serwisu internetowego opartego na architekturze SOA.
- Gdzies tu trzeba opisac co bedzie sie zawieralo w pracy, ze strona www i jaka funkcjonalnosc na tej stronie
- jaki serwer
- No i tuataj w sumie wspomiec, ze wybralem taki temat bo chcialem poznac technologie SOA? albo wspomiec o tym gdzies indziej
- NO i tutaj po kolei opisac co robie w pracy, z grubsza z czego korzystam
- Byc moze co w ktorym rozdziale tak z grubsza

- To wszystko razem powinno zajac wystarczajaco miejsca

    - Co autor chciał osiągnąć, czego się spodziewał w efekcie realizacji pracy?
- Czego czytelnik dowie się po przeczytaniu pracy?

2. Podstawy teoretyczne

Rozdział ten przedstawia podstawy teoretyczne projektu. W pierwszym porozdziale znajduje się krótkie wprowadzenie do architektóry SOA, na której oparty jest serwis. W następnym opisany jest język implementacji. Trzeci podrozdział prezentuje formę przechowywania danych w serwisie. Ostatni punkt zaznajamia nas z przenaczeniem serwerów aplikacyjnych i konkretnym zastosowaniu w implementowanym projekcie.

2.1. SOA

Rozwinięciem skrótu SOA jest *Service Oriented Architecture* czyli architektura zorientowana na usługi. Według książki *SOA Design Principles for Dummies* SOA składa się z trzech podstawowych aspektów:

- **Usługa** - czyli powtarzalne zadanie biznesowe np. Dodawanie plików na serwer lub tworzenie nowego konta.
- **Orientacja na usługi** - Sposób postrzegania produktu jako zbiór usług połączonych w całość.
- **SOA** - Podejście architektoniczne bazujące na zasadach orientacji na usługi.

Architektura zorientowana na usługi posiada następujące właściwości:

- **Luźne powiązanie usług**
- **Abstrakcyjne usługi**
- **Autonomiczność usług**
- **Możliwość wielokrotnego użycia usług**
- **Bezstanowość usług**
- **Dzielenie usług na komponenty**

[1] SOA jest architekturą wspierającą modularność więc produkty pisane w tej architekturze łatwo jest rozwijać poprzez dodawanie nowych usług.

- a tutaj moze ze moze na projekt studencki to nie jest najlepsze rozwiazanie, ale jezeli mialo by to sie rozwinac to to jest super ekstra pomysl i dlaczego tak jest! (Tutaj zapewne wpomiec, ze soa jest modularne i bez problemu mozna dodawac kolejne moduly).

2.2. Java

Język programowania Java został stworzony na początku lat dziewięćdziesiątych przez grupę inżynierów z firmy Sun zwaną "Green Team". Ideą tworzonego języka była praca w rozwijającym się środowisku internetu [2]. Java została uformowana na podstawie języka C++ przy czym wymusza ona programowanie obiektowe. Jest jednym z najbardziej popularnych języków programowania i każdy kto miał jakąkolwiek styczność z programowaniem przynajmniej o nim słyszał (Tutaj mam nadzieje, że Paulinka mi zrefaktorujesz to zdanie).

Java dzięki swojej platformie programistycznej Java EE (Java Enterprise Edition) pozwala pisać multi platformowe, wielowarstwowe niezawodne i bezpieczne aplikacje sieciowe. Dzięki wielu darmowo udostępnianym bibliotekom i frameworkom wielu programistów decyduje się na wybranie tej właśnie platformy [3]. W swojej pracy główne frameworki platformy Java EE pomagające w tworzeniu opisywanej aplikacji to:

– **JSF** - Java Server Faces, który upraszcza tworzenie internetowego interfejsu użytkownika, dzięki udostępnianym tagom, z których można korzystać tworząc strony www. Posiada także Managed Beany czyli klasy odpowiedzialne za połączenie logiki aplikacji z widokiem czyli ze stronami internetowymi.

– **Hibernate** - jest to biblioteka dla języka Java dostarczającą framework do realizacji warstwy dostępu do danych. Zapewnia translację danych z relacyjnej bazy danych na klasy Java i vice versa (Object/Relational Mapping). Zastępuje bezpośrednie metody dostępu do danych za pomocą wysokopoziomowych metod operujących na zmapowanych obiektach.

2.3. Przechowywanie danych

Wirtualne dane można przechowywać na wiele sposobów, najprostszym z nich jest przechowywanie danych lokalnie czyli na dysku twardym stacji roboczej, aczkolwiek w tym wypadku istnieje zagrożenie utraty tych danych w razie zepsucia się komputera. W obawie przed utratą danych użytkownicy często szukają sposobu na umieszczenie ich zapasowej kopii w internecie. Istnieje wiele sposobów na przechowywanie naszych plików w internecie:

– **Dyski sieciowe** - jest to typ urządzenia przechowującego dane, które zapewnia dostęp do nich w lokalnej sieci LAN.

– **Dyski internetowe** - umożliwiają przechowywanie plików w "internecie", tzn. korzystają między innymi z chmur internetowych do zapisu danych.

– **Bazy danych** - z tego rozwiązania korzysta się raczej w przypadku ustrukturyzowanych danych np. Informacje o użytkownikach (login użytkownika serwisu w postaci krótkiego pola tekstowego lub rok jego urodzenia w postaci liczby naturalnej). Pomimo tego niektóre bazy danych udostępniają możliwość przechowywania danych w postaci binarniej czyli bez podania ich typu co pozwala na zapisywanie danych bez wymuszonego ich typu.

Do implementacji aplikacji użyto bazy danych. Wybór był podyktowany tym, że platforma Java EE udostępnia framework Hibernate, który pozwala na prosty dostęp do bazy danych. Wybrany system do zarządzania relacyjną bazą danych to PostgreSQL, system ten jest wolnodostępny. Do przechowywania danych o użytkownikach i ich kalendarzach wystarczą podstawowe typy danych dostępne we większości sytemów zarządzających bazą danych. Do przechowywania większych plików takich jak całe książki w dowolnym formacie służy typ danych BLOB czyli Binary Large Objects co jak sama nazwa wskazuje służy do przechowywania większych plików binarnych.

2.4. Serwer Aplikacyjny

Serwer aplikacyjny jest kontenerem webowym, który odpowiada za przechowywanie, zdalne uruchamianie i użytkowanie aplikacji Java EE. Ponadto pomaga programiście tworzyć aplikacje oraz umożliwia oddzielenie logiki biznesowej od usług dostarczanych przez dany serwer aplikacyjny np. zabezpieczenia aplikacji [4]. Najbardziej znane serwery aplikacyjne to JBoss, IBM WebSphere, Apache Tomcat, GlassFish. Serwer używany do implementacji opisywanej aplikacji to JBoss AS 7.1.1. Serwer ten został wybrany ponieważ jest darmowy (udostępniany na licencji *GNU Lesser General Public License*), ponadto implementuje pełen zestaw usług JEE. Dodatkowym argumentem przemawiającym za wyborem właśnie tego serwera była jego intergracja z Eclipse (środowiskiem programistycznym w którym wykonany został projekt) za pomocą wtyczki JBossTools co znacznie ułatwiło pracę.

3. Projekt systemu

Rozdział ten w 3 podrozdziałach przedstawia koncepcję implementowanej aplikacji. W podrozdziale pierwszym opisany jest system, w drugim znajduje się specyfikacja wymagań, a w trzecim model systemu. Dokumentacja projektu została częściowo oprata na szablonie dokumentacji projektu stworzonym przez dr Piotra Szweda [5].

3.1. Opis systemu

W tym podrozdziale opisany jest cel systemu, jego granice, lista możliwości systemu oraz jego użytkownicy. Ponadto za pomocą diagramów aktywności przedstawione jest działanie aplikacji.

3.1.1. Cel projektu

Celem projektu jest implementacja serwisu internetowego dedykowanego dla studentów w architekturze SOA. Głównym zadaniem serwisu ma być możliwość przechowywania plików oraz udostępnianie ich innym użytkownikom. Ponadto każdy z użytkowników będzie mógł korzystać z dedykowanego dla niego kalendarza, w którym może zapisywać ważne wydarzenia. Oprócz opisanych wcześniej funkcji użytkownik za pomocą strony powiadomień będzie mógł się dowiedzieć o nadchodzących wydarzeniach oraz o udostępnionych dla niego plikach.

3.1.2. Użytkownicy

Grupę użytkowników stanowią wszyscy korzystającyz serwisu, z wyszczególnieniem:

– **Użytkownik** - Docelowym użytkownikiem serwisu jest student potrzebujący dodać lub uzyskać dostęp do pliku poprzez udostępnienie od innych użytkowników serwisu. Nie jest warunkiem koniecznym, aby użytkownik był studentem.

– **Administrator** - Użytkownik posiadający szystkie uprawnienia, takie jak usówanie użytkowników oraz wgląd do wszystkich plików w systemie.

3.1.3. Granice systemu

Granicą systemu dla użytkownika jak i dla administratora będzie strona internetowa aplikacji. Zawarty będzie tam dostęp do wszystkich funkconalności systemu. Administrator dodatkowo będzie miał dostęp do specjalnego panelu za pomocą którego będzie mógł zarządzać kontami wszystkich użytkowników.

Wejścia w systemie:

- formularz nowego konta – użytkownik wypełnia dane: login, hasło, imie, nazwisko, adres e-mail,
- formularz logowania – zalogowanie się do systemu za pomocą loginu i hasła użytkownika,
- formularz edycji konta – zmiana danych użytkownika,
- formularz nowego pliku – załączonik oraz opcjonalnie tytuł i opis pliku,
- formularz edycji pliku - zmiana załącznika, tytułu lub opisu,
- formularz udostępnienia pliku - login użytkownika, któremu ma być udostępniony wybrany plik,
- formularz nowego wydarzenia - wybrana data, nazwa wydarzenia oraz opcjonalny opis,
- formularz edycji wydarzenia - nowe dane do wybranego wydarzenia,

Wyjścia w systemie:

- Strona powiadomień - na tej stronie znajdować się będą informacje o udostępnionych plikach oraz o przypomnienia o nadchodzących wydarzeniach.
- Strona przeglądania własnych plików - użytkownik na tej stronie może przeglądać dodane przez siebie pliki,
- Strona przeglądania udostępnionych plików - tutaj użytkownik będzie mógł przeglądać pliki mu udostępnione,
- Strona kalendarza - Strona odpowiedzialna za wyświetlanie kalendarza.

3.1.4. Lista możliwości

Lista wymagań funkcjonalnych:

- Logowanie,
- Wylogowanie,
- Rejestracja,
- Zarządzanie kontem,

– Zarządzanie materiałami,

– Pobieranie pliku,

– Udostępnianie pliku,

– Zarządzanie kalendarzem,

– Otrzymywanie powiadomień,

Dodatkowe funkcje administratora:

– Zarządzanie kontami użytkowników.

Na kolejnych stronach zostały przedstawione diagramy czynności typowych akcji.

3.2. Specyfikacja i analiza wymagań

Dzięki analizie wymagań funkcjonalnych możemy zidentyfikować oczekiwane zachowania budowanego systemu. Wymaganie funkcjonalne jest to „stwierdzenie, jakie usługi ma oferować system, jak ma reagować na określone dane wejściowe oraz jak ma się zachowywać w określonych sytuacjach. W niektórych wypadkach wymagania funkcjonalne określają, czego system nie powinien robić" [6]. W tym podrozdziale opisane zostaną lista wymagań wunkonalnych oraz niefunkcjonalnych oraz przedstawiony zostanie diagram przypadków użycia z dołączonymi scenariuszami.

3.2.1. Wymagania oprogramowania

Wymagania funkcjonalne zostały wypisane w podpunkcie 3.1.4. Wymagania niefunkcjonalne:

1. Wymaganie organizacyjne:
   – Implementacji : językiem programowania jest Java,
   – Użyteczności : Interfejs graficzny użytkownika jest przejrzysty i łatwy do przyswojenia,
2. Wymagania zewnętrzne:
   – Poufności - Projekt przestrzega wymagań prawnych ustanowionych w Ustawie o Ochronie Danych Osobowych.
   – Bezpieczeństwa : System nie pozwala użytkownikom na nieautoryzowany dostęp do bazy danych

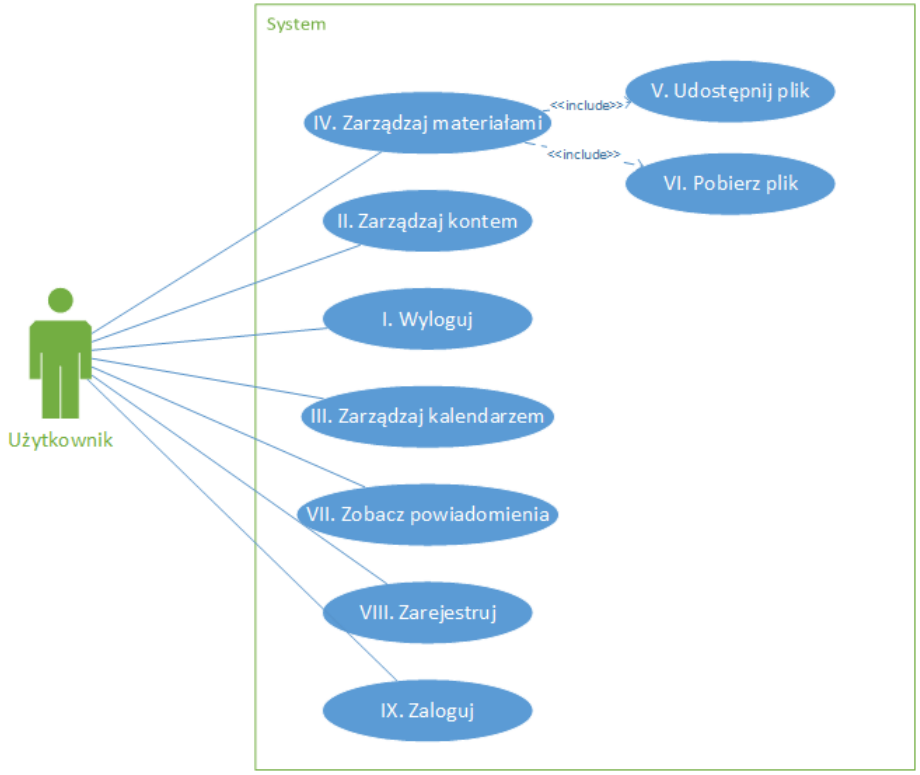
3.2.2. Ryzyka operacyjne

Ryzykiem w implementowanym systemie może być niedostateczne zabezpieczenie bazy danych lub pojawienie się nieobsłużonych błędów. //TODO: Poszukać i dodać tu jeszcze cos

3.2.3. Przypadki użycia

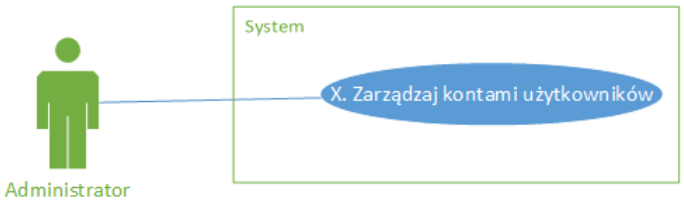
Dzięki przypadkom użycia możemy okeślić jak użytkownik będzie korzystał z systemu. Dzięki diagramowi możemy zobaczyć jakie czynności może wykonywać dany użytkownik [7]. Opisy czynności zawartych na diagramie znajdują się w scenariuszach przypadków użycia.

Poniższe diagramy prezentują możliwe interakcje użytkowników z systemem. W poniższym dia-



Rys. 3.1. Diagram przypadków użycia dla użytkownika

gramie pominięto przypadki użycia, które posiada zwykły użytkownik, lecz nie należy zapominać, że administrator również je posiada.



Rys. 3.2. Diagram przypadków użycia dla administratora

Scenariusze przypadków użycia:

Tabela 3.1. Wyloguj

Aktorzy:	Użytkownik, Administrator
Zakres:	System
Poziom:	Sytemowowy
Udziałowcy i ich cele:	Użytkownik chce się wylogować.
Zdarzenie wyzwalające	Użytkownik wciska przycisk “Wyloguj”.
Warunki wstępne:	Użytkownik jest zalogowany.
Warunki końcowe dla sukcesu:	Użytkownik jest wylogowany.
Warunki końcowe dla niepowodzenia:	Użytkownik jest zalogowany.

Scenariusz główny:

1. Użytkownik wciska przycisk wyloguj widoczny w prawym górnym rogu strony.
2. Pojawia się okno z napisem: “Czy na pewno chcesz się wylogować?”
3. Użytkownik wciska przycisk “tak”.
4. Użytkownik jest wylogowany i przekierowany na stronę główną systemu.

Scenariusz alternatywany:

3.a. Użytkownik wciska “nie”.

3.a.1. Okno z napisem znika, użytkownik dalej jest zalogowany.

Tabela 3.2. Zarządzaj kontem

Aktorzy:	Użytkownik, Administrator
Zakres:	System
Poziom:	Sytemowowy
Udziałowcy i ich cele:	Użytkownik chce zmienić login lub hasło.
Zdarzenie wyzwalające	Użytkownik wciska przycisk “Zarządzaj kontem”.
Warunki wstępne:	Użytkownik jest zalogowany.
Warunki końcowe dla sukcesu:	Login lub hasło użytkownika są zmienione.
Warunki końcowe dla niepowodzenia:	Login lub hasło użytkownika nie są zmienione.

3.3. Model systemu

nad tym tytulem pomyslec!

3.3.1. Architektutra systemu

- najpierw napisac ze soa
- zastanowic sie czy tutaj opis soa czy we wstepie, ale chyba we wstepie, tutaj jedynie można wspomiec ze wczesniej opisany w punkcie X.x.
- Diagram
- byc moze opis

3.3.2. Model bazy danych

- tutaj jakis ladny wstep.
- MODEL ERD - ostateczny

3.3.3. Model klas

- rowniez jakis ladny wstep
- No i diagram klas tutaj ładnie sklepać trzeba będzie

3.3.4. Diagramy sekwencji

- Wstep, ze to wynika z diagramu klas, cos ładnego o diagramach sekwencji - tutaj moze sie wykładami szweda posłużyć
- No i wklepac diagramy sekwencji - moze nie bedzie tak zle jak już beda diagramy klas

3.4. Projekt interfejsu?

Nad tym sie zastanowic czy to robic czy dac dupie siana

4. Implementacja

Kilka zdań dotyczących zawartości poszczególnych podrozdziałów

4.1. Architektura

Opis warstw systemu, rozmieszczenie geograficzne oprogramowania
tutaj pewnie trzeba bedzie napisac ze to na mvc
no i ze podzielone przez ejb na war, ejb no ten koncowy

4.1.1. Dostęp do bazy danych

tutaj subsekcje w jednej dostep do danych dao i hibernate

4.1.2. Widok aplikacji

te dwa ponizsze moze w dodatkowych podsekcjach JSF
przykladowe formularze

4.2. Zabezpieczenia

tutaj cos napisac o tych zabezpieczeniach co u rogiusa bazgrałem

4.3. Testy

To chyba mozna tutaj bo tego nie bedzie duzo i nie ma co osobnego rozdzialu
Moze napisze jakies junity to bedzie cos
mozna napisac o planach testow manualnych itd

5. Wyniki i podrecznik uzytkowania

Albo to upchnac w implementacji albo jak sie wyrobie to uda sie jakis osobny rozdzial

5.1. No idea

6. Podsumowanie

W podsumowaniu pracy należy zebrać wnioski z jej realizacji. Odpowiedzieć na pytania: czy cel pracy został osiągnięty i w jakim stopniu. Co inaczej byłoby realizowane, gdyby autor od nowa zaczął tę pracę? Można tu podać trochę ciekawostek z jej realizacji. Kto i jaki pożytek może mieć z tej pracy? Czy warto kontynuować tą pracę i w jaki sposób? Jakie nowe problemy zostały zidentyfikowane i które z nich mogę być przedmiotem kolejnych prac dyplomowych? Podsumowanie nie ma na celu wykazywać, że stworzony produkt jest idealny i bezbłędny, lecz to, że autor jest rzetelnym i kompetentnym projektantem i analitykiem. Do błędów popełnionych w pracy należy się uczciwie przyznać.
Tutaj bez podrozdzialow.

Bibliografia

[1] Claus T. Jensen. *SOA Design Principles for Dummies*. New Jersey: John Wiley and Sons, Inc., 2013.

[2] *The History of Java Technology*. `http://www.oracle.com/technetwork/java/javase/overview/javahistory-index198355.html`. Oracle.

[3] *JEE - wprowadzenie do Javy Enterprise Edition*. `http://javastart.pl/static/jee/jeewprowadzeniedojavyenterprise-edition/`.

[4] R. Matusik. *Serwery aplikacji*. `http://math.uni.lodz.pl/ radmat/serwery/Wyklad_d_01.pdf`. Uniwersytet Łudzki.

[5] P. Szwed. *Szablon dokumentacji projektowej*. `http://home.agh.edu.pl/ pszwed/wiki/doku.php?id=amo:rup_tailored`. Akademia GórniczoHutnicza. 2015.

[6] K. Zmitrowicz. *Analiza wymagań funkcjonalnych*. `http://wymagania.net/bazawiedzy/36bazawiedzy/88analizawymagan-funkcjonalnych`.

[7] H. Wesołowska. *Jak opisywać przypadki użycia?* `http://analizait.pl/2012/jakopisywacprzypadkiuzycia/`. 2012.