POLITECHNIKA WROCŁAWSKA

WYDZIAŁ ELEKTRONIKI

KIERUNEK: INFORMATYKA (INF)

SPECJALNOŚĆ: INŻYNIERIA SYSTEMÓW INTERNETOWYCH (XXX)

PRACA DYPLOMOWA

MAGISTERSKA

Projekt i implementacja wielowarstwowej rozproszonej aplikacji internetowej działającej na platformie Java EE, wspierającej

działalność przedsiębiorstwa handlowego

Design and implementation of a multi-layered distributed web application operating on the Java EE platform, supporting the operation of a commercial enterprise.

AUTOR:

Michał Hawełka

PROWADZĄCY PRACĘ:

dr inż. Zofia Kruczkiewicz

OCENA PRACY:

WROCŁAW, 2018

**Spis treści**

[Spis rysunków 4](#_Toc517856684)

[Spis tabel 5](#_Toc517856685)

[Spis listingów 6](#_Toc517856686)

[Skróty 7](#_Toc517856687)

[1. Wstęp 8](#_Toc517856688)

[1.1. Cel pracy 8](#_Toc517856689)

[1.2. Zakres pracy 9](#_Toc517856690)

[2. Technologie używane do budowy aplikacji internetowych 10](#_Toc517856691)

[2.1. Wielowarstwowa rozproszona aplikacja internetowa 10](#_Toc517856692)

[2.2. Warstwa kliencka 11](#_Toc517856693)

[2.3. Warstwa sieciowa / warstwa prezentacji 11](#_Toc517856694)

[2.3.1. Java Server Faces 12](#_Toc517856695)

[2.3.2. Model View Controller 12](#_Toc517856696)

[2.4. Warstwa biznesowa 13](#_Toc517856697)

[2.4.1. Funkcjonalności Java EE 14](#_Toc517856698)

[2.5. Warstwa integracji 16](#_Toc517856699)

[2.5.1. Java Messaging Service 16](#_Toc517856700)

[2.5.2. Java Persistence API 16](#_Toc517856701)

[2.6. Warstwa danych 16](#_Toc517856702)

[2.6.1. PostgreSQL 16](#_Toc517856703)

[3. Opracowanie projektu wielowarstwowej rozproszonej aplikacji internetowej 17](#_Toc517856704)

[3.1. Opis świata rzeczywistego 17](#_Toc517856705)

[3.1.1. Opis zasobów ludzkich 17](#_Toc517856706)

[3.1.2. Dane techniczne 17](#_Toc517856707)

[3.2. Wymagania funkcjonalne 18](#_Toc517856708)

[3.3. Wymagania niefunkcjonalne 18](#_Toc517856709)

[3.4. Przypadki użycia 18](#_Toc517856710)

[3.5. Model danych 18](#_Toc517856711)

[4. Szczegóły projektowe 19](#_Toc517856712)

[4.1. Diagramy klas 19](#_Toc517856713)

[4.2. Diagramy sekwencji 19](#_Toc517856714)

[5. Szczegóły implementacyjne 19](#_Toc517856715)

[5.1. Architektura systemu 19](#_Toc517856716)

[5.2. Rozproszoność systemu 19](#_Toc517856717)

[5.3. Bezpieczeństwo 19](#_Toc517856718)

[6. Pomiary wydajności i analiza wyników 19](#_Toc517856719)

[6.1. Użyte metryki 20](#_Toc517856720)

[6.2. Środowiska testowe 20](#_Toc517856721)

[6.3. Otrzymane wyniki 20](#_Toc517856722)

[6.4. Analiza wyników 20](#_Toc517856723)

[7. Podsumowanie i wnioski 21](#_Toc517856724)

[Literatura 22](#_Toc517856725)

[Dodatek A 23](#_Toc517856726)

# Spis rysunków

[Rys. 1. Schemat warstw wielowarstwowej aplikacji internetowej [1]. 11](#_Toc517856678)

[Rys. 2. Logo Java Server Faces [2]. 12](#_Toc517856679)

[Rys. 3. Uproszczony schemat MVC [3]. 13](#_Toc517856680)

[Rys. 4. Przykładowa architektura systemu na platformie Java EE [3] 14](#_Toc517856681)

[Rys. 5. Logo Java EE [5] 14](#_Toc517856682)

# Spis tabel

[Tab. 1. Przykład podpisu tabeli 4](#_Toc465685652)

# Spis listingów

[Listing. 1. Początkowe żądanie HTTP 13](#_Toc517768200)

[Listing. 1. Początkowe żądanie HTTP 16](#_Toc517768201)

# Skróty

**JEE** (ang. *Java Enterprise Edition*)

**XML** (ang. *eXtensible Markup Language*)

**SOAP** (ang. *Simple Object Access Protocol*)

**JPA** (ang. *Java Persistence API*)

**JSF** (ang. *Java Server Faces*)

**JMS** (ang. *Java Messaging System*)

**MDB** (ang. *Message-Driven Bean*)

**EJB** (ang. *Enterprise Java Bean*)

1. Wstęp

Rozproszone, wielowarstwowe aplikacje internetowe pełnią w dzisiejszych czasach olbrzymią rolę. Pozwalają one na używanie wielu różnych technologii, zapewniają skalowalność i bezpieczeństwo. Te aspekty sprawiły, że są one często wybierane przez przedsiębiorstwa handlowe, które dzięki aplikacjom wielowarstwowym są w stanie efektywnie zarządzać swoim inwentarzem, finansami czy obsługą zamówień. Każdy z wymienionych tematów jest tematem zdecydowanie rozległym i wymagającym dobrze napisanego systemu odpornego na zewnętrzne ataki, a także pozwalającego na łatwą rozbudowę.

* 1. Cel pracy

Celem tej pracy jest sprawdzenie jakie możliwości mogą zaoferować swoim klientom przedsiębiorstwa korzystające z systemu wielowarstwowego używającego technologii Java EE. Obecnie na rynku jest dostępnych również kilka innych technologii, ale to Java EE ma za sobą już całkiem sporą historię i jest jednym z wiodących wyborów wśród systemów internetowych. Oprócz samych możliwości technologii zostały sprawdzone również kryteria wydajnościowe. Porównane zostały parametry wielowarstwowego systemu rozproszonego, a także prostej architektury ze scentralizowaną bazą danych. Zostały wykonane następujące zadania:

* Opracowanie i wykonanie projektu wielowarstwowej rozproszonej aplikacji internetowej wspierającej przedsiębiorstwo handlowe, działającej na platformie Java EE z wykorzystaniem technologii JMS
* Implementacja projektu wielowarstwowej rozproszonej aplikacji internetowej Java EE z wykorzystaniem technologii JMS
* Opracowanie metryk umożliwiających ocenę wydajności wykonanej aplikacji
* Wykonanie pomiarów wydajności wykonanej aplikacji
* Analiza wyników i ocena wydajności
  1. Zakres pracy

W rozdziale drugim krótko przedstawione zostały technologie używane do budowy aplikacji internetowych. Skupiono się głównie na frameworku Java EE. W dalszej części rozdziału wyszczególniono również najczęściej stosowane technologie do budowy warstwy prezentacji, warstwy logiki biznesowej czy warstwy integracji (na przykład Java Persistence API czy Java Messaging Service).

W trzecim rozdziale opisany został proces opracowania koncepcji systemu – od modelu biznesowego, przez wymagania jakie stawiane są przed systemami internetowymi, aż do opracowania modelu konceptualnego. Pojawiły się tam między innymi odpowiednie przypadki użycia, które powinna obsługiwać rozproszona aplikacja internetowa, która docelowo ma być wykorzystywana przez przedsiębiorstwo handlowe. Zaprezentowane zostały również przykładowe schematy bazy danych.

Kolejny rozdział to podsumowanie fazy projektowania systemu. Abstrakcja przyjęta w rozdziale poprzednim zyskuje tutaj uściślenie. Zaprezentowane zostały diagramy klas czy dokładne diagramy sekwencji. Trochę miejsca zostało również poświęcone warstwie prezentacji. Pojawiają się projekty formularzy, które będą używane przez stronę klienta.

W rozdziale piątym przedstawione zostały szczegóły implementacyjne. Oprócz najważniejszych fragmentów kodu źródłowego umieszczone zostały tam dowody na wielowarstwowość, czyli rozdzielenie poszczególnych warstw od siebie. Architektura systemu, czyli opis warstw i topologia oprogramowania to również zawartość tegoż rozdziału. Na koniec ujęty został temat bezpieczeństwa w tej konkretnej implementacji systemu internetowego, a więc odporności na awarie czy prawa dostępu użytkowników.

Szósty rozdział skupia się na testowaniu powstałego systemu. Zawiera dokładny plan testów wraz z danymi wejściowymi i oczekiwanymi rezultatami. Do tego pojawiają się zestawienia wyników poszczególnych rodzajów systemów w zależności od użytej topologii, a także analiza wyników przeprowadzonych testów.

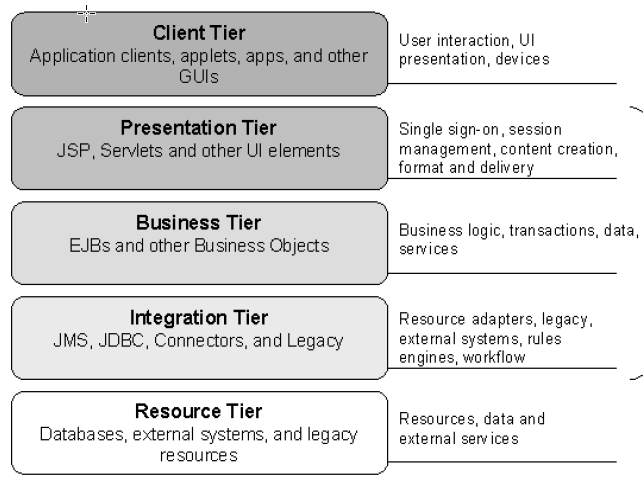
1. Technologie używane do budowy aplikacji internetowych
   1. Wielowarstwowa rozproszona aplikacja internetowa

Rozproszoną aplikacją internetową nazywany jest program, który działa na wielu komputerach w sieci. Tego typu aplikacje porozumiewają się ze sobą w celu wykonania jakiegoś zadania (na przykład utrzymywania spójności danych, wymiany informacji czy podziału odpowiedzialności). Aplikacje najczęściej są budowane na architekturze klient-serwer, gdzie n serwerów komunikuje się ze sobą. Klient odpowiada za interakcję z użytkownikiem, za wysokopoziomowy dostęp do danych, serwer z kolei odpowiada za logikę biznesową, komunikację z innymi serwerami, a także z warstwą danych (np. baza danych).

Wielowarstwowość aplikacji internetowej jest powiązana w pewien sposób z rozproszonością. Większość aplikacji rozproszonych jest aplikacjami wielowarstwowymi, ale nie wszystkie aplikacje wielowarstwowe są rozproszone. Najczęściej spotykany model aplikacji wielowarstwowej składa się z pięciu warstw:

* Warstwy danych (Pliki, bazy danych)
* Warstwy integracji (Web serwisy, JMS, DAO, JPA, MDB)
* Warstwy logiki biznesowej (Session EJB)
* Warstwy sieciowej/prezentacji (Serwlety, JSP, JSF)
* Warstwy klienckiej (Przeglądarka, aplet Java)

Tego typu podział ma zapewnić aplikacji wysoką skalowalność i ułatwić jej utrzymanie w przyszłości. Poszczególne warstwy są od siebie logicznie oddzielone w celu uzyskania możliwości na łatwiejszą rozbudowę systemu czy zmniejszenia podatności na błędy.



Rys. . Schemat warstw wielowarstwowej aplikacji internetowej [1].

* 1. Warstwa kliencka

Warstwa kliencka to punkt wejściowy do aplikacji. To tutaj użytkownik wchodzi w interakcję z systemem. Najczęściej ta warstwa aplikacji jest uruchamiana na maszynie użytkownika lub w przeglądarce internetowej. Odpowiada ona za wyświetlanie zawartości użytkownikowi, wstępną walidację wprowadzanych danych, a także za komunikację z serwerem.

* 1. Warstwa sieciowa / warstwa prezentacji

Warstwa sieciowa przechowuje wszystkie komponenty potrzebne do uruchomienia warstwy klienckiej, a także odpowiada za zarządzanie sesjami użytkowników. Popularniejszą nazwą jest warstwa prezentacji, ale w przypadku aplikacji tworzonych na platformie Java EE można mówić o warstwie sieciowej (głównie ze względu na użycie takich technologii jak serwlety Java, Java Server Pages czy Java Server Faces).

* + 1. Java Server Faces

Java Server Faces (JSF) to technologia, która została opracowana w celu ułatwienia tworzenia warstwy sieciowej i warstwy prezentacji w aplikacjach sieciowych. Jej główne założenia to uniezależnienie implementacji od wyboru języka, protokołów czy urządzeń klienckich. JSF zapewnia własne urządzenie renderujące, które może być używane przez poszczególne urządzenia klienckie, dzięki czemu programista może w łatwy sposób przygotować rozwiązania dla dużej grupy odbiorców.



Rys. 2. Logo Java Server Faces [2].

Cechy Java Server Faces:

* Prostota użytkowania
* Rozdzielenie logiki i warstwy prezentacji
* Użycie wzorca MVC (Model View Controller)
  + 1. Model View Controller

Wzorzec Model-View-Controller jest już używany od dłuższego czasu, ale wraz z rozpowszechnieniem internetu zwiększyła się jego popularność, a sam wzorzec ewoluował. Założenia wzorca prezentują się następująco:

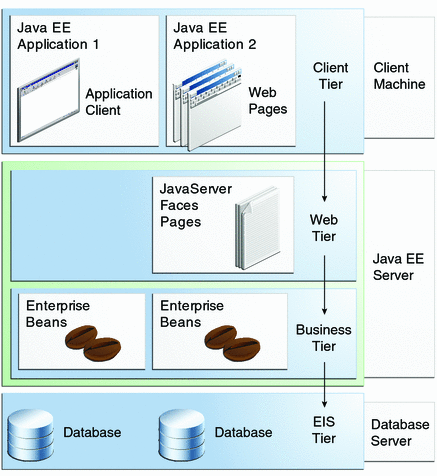
* Model – to opis lub reprezentacja danych. Nie są to właściwe dane. Model to pewnego rodzaju abstrakcja, pozwalająca na przykład na używanie wielu baz danych niezależnie od ich schematu.
* Widok (View) – warstwa prezentacji, to co widzi użytkownik. Widok zapewnia również interfejs do przechwytywania danych wejściowych od użytkownika.
* Kontroler (Controller) – kontroluje przepływ informacji między modelem i widokiem. Decyduje o tym, jakie dane są wyciągane z bazy danych przez model i jakie dane są przekazywane do widoku. Implementuje też logikę biznesową.



Rys. . Uproszczony schemat MVC [3].

* 1. Warstwa biznesowa

Warstwa biznesowa to warstwa, w której odbywa się większość przetwarzania danych biznesowych. Jest w tym zawarta cała logika biznesowa, a także takie funkcjonalności jak transakcje czy usługi. W przypadku aplikacji na platformie Java EE warstwa ta składa się najczęściej z Enterprise Java Beans (EJB).



Rys. . Przykładowa architektura systemu na platformie Java EE [3]

* + 1. Funkcjonalności Java EE

Z punktu widzenia wielowarstwowej aplikacji internetowej najważniejszą funkcjonalnością platformy Java EE są Enterprise Java Beans (EJB). Są to komponenty działające po stronie serwera aplikacji zawierające logikę biznesową. Wykorzystuje się je głównie do budowania złożonych aplikacji. Jest tak dlatego, że EJB pozwalają na łatwą rozbudowę programu, a także na jego rozproszenie. Same EJB są uruchamiane w tzw. kontenerach EJB, które zapewniają użytkownikowi szereg usług już na początku, bez potrzeby ich implementacji. Do tego typu usług można zaliczyć na przykład obsługę transakcji czy bezpieczeństwo.



Rys. . Logo Java EE [5]

W przypadku systemu dla przedsiębiorstwa handlowego EJB jest dobrym wyborem z kilku powodów:

* Pozwala na obsługę różnego typu klientów
* Aplikacje oparte na EJB są skalowalne
* Pozwala na zaawansowane przetwarzanie transakcyjne

EJB wyróżnia dwa typy komponentów – Sesyjne (Session Beans) i Komunikatowe (Message-Driven Beans). Oba typy są użyteczne w projektowaniu wielowarstwowej rozproszonej aplikacji internetowej.

* + - 1. Session Bean

Ziarenka sesyjne mają za zadanie wykonać konkretną czynność dla klienta. Nie są współdzielone, a ich stan nie wykracza poza daną sesję, a co za tym idzie nie jest reprezentowany w warstwie danych (np. w bazie danych). Wyszczególnia się dwa podtypy ziarenek sesyjnych:

* Stanowe (stateful) – to ziarenka, które reprezentują stan sesji z konkretnym klientem. Zapisywane są w nich informacje dotyczące działań i zapytań klienta (np. klient nie musi za każdym razem pobierać listy przedmiotów, bo ta jest zapisana w ziarenku).
* Bezstanowe (stateless) – ziarenka używane głównie do wykonywania metod logiki biznesowej (np. web service’y)
  + - 1. Message-Driven Bean

Listing. 1. Początkowe żądanie HTTP

GET /script/Articles/Latest.aspx HTTP/1.1  
Host: www.codeproject.com  
Connection: keep –alive  
Cache -Control: max-age=0  
Accept: text/html ,application/xhtml+xml,application/xml|  
User -Agent: Mozilla/5.0 ...  
Accept -Encoding: gzip ,deflate ,sdch  
Accept -Language: en-US...  
Accept -Charset: windows -1251,utf -8...

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id dapibus enim. Etiam lobortis pulvinar enim in maximus.

Przykład listy numerowanej:

* + - 1. Wstrzykiwanie zależności (Dependency Injection)
  1. Warstwa integracji

Warstwa integracji odpowiada za odpowiednie przygotowanie danych dla warstwy danych. Jest tutaj dokonywane nie tylko mapowanie struktur danych aplikacji na bazodanowe, ale również odbywa się tutaj komunikacja z innymi serwerami czy bazami danych. Integracja może przebiegać synchronicznie lub asynchronicznie.

* + 1. Java Messaging Service

CZYM JEST, DLACZEGO WAŻNE Z PUNKTU WIDZENIA TEMATU PRACY Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id dapibus enim. Etiam lobortis pulvinar enim in maximus.

* + 1. Java Persistence API

KRÓTKO CZYM JEST Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id dapibus enim. Etiam lobortis pulvinar enim in maximus.

* 1. Warstwa danych

Warstwa danych to najczęściej baza danych. Przechowywane są tu wszystkie istotne informacje niezbędne do właściwego działania aplikacji. Bazy danych mogą być skonfigurowane na różne sposoby, w aplikacjach internetowych stosuje się na przykład:

* Centralną bazę danych – wszystkie serwery korzystają z jednej bazy danych
* Synchronizowaną bazę danych - każdy serwer ma swoją własną bazę, która jest synchronizowana przy każdej aktualizacji bazy dowolnego serwera
* Klastry danych – każdy serwer posiada swoją własną bazę, przy odczycie danych dane są zbierane z różnych serwerów.
  + 1. PostgreSQL

KRÓTKO CZYM JEST Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id dapibus enim. Etiam lobortis pulvinar enim in maximus.

1. Opracowanie projektu wielowarstwowej rozproszonej aplikacji internetowej

Sklep z multimediami to idealne miejsce do wykorzystania wielowarstwowej rozproszonej aplikacji internetowej. Tego typu architektura pozwala na obsługę wielu oddziałów sklepu i ciągłe utrzymywanie aktualnego stanu magazynów przy jednoczesnym zabezpieczeniu przed awariami.

* 1. Opis świata rzeczywistego
     1. Opis zasobów ludzkich

Pracownik sklepu może aktualizować bazę produktów poprzez dodawanie lub usuwanie dostępnych egzemplarzy poszczególnych multimediów. Każda pozycja posiada tytuł, cenę, opis, liczbę egzemplarzy i dane magazynu, w którym się znajdują. W zależności od rodzaju multimediów (gra wideo, film, album muzyczny) wyróżniane są dodatkowe informacje takie jak na przykład gatunek, nośnik czy producent. Dodatkowo każda pozycja jest opisany identyfikatorem ułatwiającym ewidencję posiadanego inwentarza i wyszukiwanie wszystkich dostępnych egzemplarzy.

Klient ma możliwość wyszukiwania produktów, dodawania ich do koszyka i składania zamówienia. Ma również możliwość podejrzenia swoich wszystkich, a także anulowania aktywnych zamówień.

* + 1. Dane techniczne

Klient i pracownik sklepu mają dostęp do sklepu z poziomu przeglądarki internetowej. Zakłada się, że sklep będzie w stanie obsłużyć ponad 1000 użytkowników jednocześnie, a baza danych będzie zawierać kilkadziesiąt tysięcy produktów. Sklep ma kilka oddziałów, więc potrzebna jest integracja danych znajdujących się w więcej niż jednym miejscu. Aplikacja ma być ubezpieczona na ewentualne awarie bazy danych.

* 1. Wymagania funkcjonalne

Warstwa danych to najczęściej baza danych. Przechowywane są tu wszystkie istotne informacje niezbędne do właściwego działania aplikacji. Bazy danych mogą być skonfigurowane na różne sposoby, w aplikacjach internetowych stosuje się na przykład:

* 1. Wymagania niefunkcjonalne

Warstwa danych to najczęściej baza danych. Przechowywane są tu wszystkie istotne informacje niezbędne do właściwego działania aplikacji. Bazy danych mogą być skonfigurowane na różne sposoby, w aplikacjach internetowych stosuje się na przykład:

* 1. Przypadki użycia

Warstwa danych to najczęściej baza danych. Przechowywane są tu wszystkie istotne informacje niezbędne do właściwego działania aplikacji. Bazy danych mogą być skonfigurowane na różne sposoby, w aplikacjach internetowych stosuje się na przykład:

* 1. Model danych

Warstwa danych to najczęściej baza danych. Przechowywane są tu wszystkie istotne informacje niezbędne do właściwego działania aplikacji. Bazy danych mogą być skonfigurowane na różne sposoby, w aplikacjach internetowych stosuje się na przykład:

1. Szczegóły projektowe
   1. Diagramy klas
   2. Diagramy sekwencji
2. Szczegóły implementacyjne
   1. Architektura systemu
   2. Rozproszoność systemu

Listing. 1. Początkowe żądanie HTTP

GET /script/Articles/Latest.aspx HTTP/1.1  
Host: www.codeproject.com  
Connection: keep –alive  
Cache -Control: max-age=0  
Accept: text/html ,application/xhtml+xml,application/xml|  
User -Agent: Mozilla/5.0 ...  
Accept -Encoding: gzip ,deflate ,sdch  
Accept -Language: en-US...  
Accept -Charset: windows -1251,utf -8...

* 1. Bezpieczeństwo

1. Pomiary wydajności i analiza wyników

Do przetestowania:

- czas dodania/edycji nowego elementu do bazy danych za pomocą synchronizacji JMSowej i połączenia z centralną bazą danych.

- czas wylistowania wszystkich przedmiotów z lokalną bazą synchronizowaną JMSami i centralną bazą danych

* 1. Użyte metryki
  2. Środowiska testowe
  3. Otrzymane wyniki
  4. Analiza wyników

1. Podsumowanie i wnioski

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id dapibus enim. Etiam lobortis pulvinar enim in maximus. Aliquam erat volutpat. Integer maximus est turpis, ut bibendum ligula accumsan et. Ut eget vestibulum libero. Aliquam erat volutpat. Nullam placerat mauris a lectus tincidunt, et aliquet turpis aliquam. Etiam in malesuada lacus. Proin dignissim augue sit amet auctor elementum. Suspendisse potenti. Vivamus suscipit vulputate massa ac molestie. Suspendisse a justo porttitor, commodo mi at, placerat risus. Integer lobortis augue ac neque suscipit, vel sodales lacus fringilla.

# Literatura

[1] M. Bickley, C. Slominski. A MySQL-based data archiver: preliminary results. Proceedings of ICALEPCS07, Paz. 2007. http://www.osti.gov/scitech/servlets/purl/922267 [dostęp dnia 20 czerwca 2015].

[2] <http://www.oracle.com/technetwork/java/javaee/overview-140548.html>

[3] <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

[3] J. Jędrzejczyk, B. Sródka. Segmentacja obrazów metodą drzew decyzyjnych. Raport instytutowy, Politechnika Wrocławska, Wydział Elektroniki, 2007.

<http://wazniak.mimuw.edu.pl/images/1/1e/ZAWWW-2st1.2-w04.tresc-1.0czb.pdf>

[4] <https://docs.oracle.com/javaee/7/index.html>

# Dodatek A