FINDATA – IMPORTER

DOKUMENTACJA PROJEKTOWA

*Projekt został przygotowany przez:*

|  |  |  |
| --- | --- | --- |
| **Imię i nazwisko** | **Numer albumu** | **Numer grupy dziekańskiej** |
| Bartłomiej Mike | 176509 | KrZZIs2011Io |
| Wojciech Sierewicz | 178391 | KrZZIs2011Io |

Spis treści

[1. Wstęp 1](#_Toc391059386)

[2. Założenia aplikacji 1](#_Toc391059387)

[3. Opis technologii 2](#_Toc391059388)

[4. Architektura aplikacji 2](#_Toc391059389)

[5. Wymagania techniczne i instalacja środowiska 3](#_Toc391059390)

[5.1. JDK 1.7 3](#_Toc391059391)

[5.2. Baza danych PostgreSQL 9.3 3](#_Toc391059392)

[5.3. Serwer aplikacyjny JBoss EAP 6.2 4](#_Toc391059393)

[5.4. Maven 3.2.1 5](#_Toc391059394)

[6. Wdrożenie aplikacji 5](#_Toc391059395)

[6.1. Zbudowanie paczki \*.war 5](#_Toc391059396)

[6.2. Wdrożenie aplikacji na serwer 5](#_Toc391059397)

[7. Opis funkcjonalny 5](#_Toc391059398)

[8. Dokumentacja kodu 6](#_Toc391059399)

# Wstęp

Celem niniejszego dokumentu jest dostarczenie specyfikacji technicznej oraz funkcjonalnej dla aplikacji Findata-Importer. Dokument opisuje założenia projektu, architekturę aplikacji, wymagania techniczne, opis technologii, instrukcję instalacji i wdrożenia aplikacji oraz opis funkcjonalny.

# Założenia aplikacji

Aplikacja Findata-Importer służy do pobierania notowań spółek giełdowych z serwisu <http://findata.co.nz/>, transformacji danych, oraz umieszczeniu tych danych w przygotowanej bazie danych, zgodnie z procesem ETL. Szczegółowe wymagania projektu są opisane w załączonym dokumencie „Wymagania”.

# Opis technologii

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lp.** | **Nazwa technologii** | **Wersja** | **Link do źródeł/ dokumentacji** | **Opis** |
| 1. | Java EE | 7 | <http://www.oracle.com/technetwork/java/javaee/overview/index.html> | Java Enterprise Edition – standard programowania w języku JAVA zawierający biblioteki do tworzenia zaawansowanych aplikacji |
| 2. | JDK | 7u60 | <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html> | Java Development Kit – zestaw podstawowych bibliotek języka JAVA oraz środowisko uruchomieniowe dla aplikacji (kompilator, maszyna wirtualna) |
| 3. | Spring MVC | 4.0.4 | <http://docs.spring.io/spring/docs/4.0.4.RELEASE/spring-framework-reference/htmlsingle/> | Framework służący do tworzenia aplikacji w oparciu o model Model-View-Controller |
| 4. | JPA | 2.1 | <http://docs.oracle.com/javaee/6/tutorial/doc/bnbpz.html> | Java Persistence API – standard definiujący możliwość zarządzania bazą danych w języku programowania JAVA |
| 5. | Hibernate | 4.2.7.SP1 | <http://docs.jboss.org/hibernate/orm/4.2/devguide/en-US/html_single/> | Implementacja JPA – zestaw bibliotek implementujących standard zarządzania bazą danych w języku programowania JAVA |
| 6. | JSF | 2.2 | <http://docs.oracle.com/javaee/7/tutorial/doc/jsf-intro.htm> | Java Server Faces – framework służący do generowania warstwy prezentacji w aplikacji webowej |
| 7. | Primefaces | 4.0 | <http://www.primefaces.org/showcase/> | Zestaw bibliotek będących rozszerzeniem standardu JSF, zawiera zestaw gotowych komponentów do budowania stron internetowych |
| 8. | JSoup | 1.7.3 | <http://jsoup.org/download> | Zestaw bibliotek służących do przetwarzania stron internetowych |
| 9. | Pretty Faces | 2.0.12 | <http://ocpsoft.org/prettyfaces/> | Narzędzie służące do generowania adresów stron internetowych w aplikacji webowej |
| 10. | PostgreSQL | 9.3 | <http://www.postgresql.org/download/> | System zarządzania bazą danych, implementacja języka SQL |
| 11. | Jboss | EAP 6.2 | <http://jbossas.jboss.org/downloads/> | Serwer aplikacyjny, środowisko uruchomieniowe dla aplikacji napisanych w standardzie Java EE |
| 12. | Maven | 3.2.1 | <http://maven.apache.org/download.cgi> | Narzędzie do zarządzania projektami w języku JAVA, umożliwia budowanie aplikacji, dostarczanie wymaganych bibliotek, testowanie, generowanie dokumentacji |
| 13. | Google Chrome | 35.0.1916.153 | <http://www.google.pl/intl/pl/chrome/browser/> | Zalecana przeglądarka internetowa umożliwiająca dostęp do aplikacji |

# Architektura aplikacji

Aplikacja Findata-Importer jest aplikacją webową. Oznacza to, że aplikacja działa w architekturze klient-serwer. Źródło aplikacji znajduje się na serwerze aplikacyjnym, a dostęp do niej możliwy jest poprzez przeglądarkę internetową.

Aplikacja jest zbudowana na klasycznym modelu Model-View-Controller. Takie rozwiązanie zapewnia skalowalność aplikacji oraz możliwość podmiany modułu w przyszłości bez konieczności ingerencji w pozostałe. Rozwiązanie zostało zaimplementowane za pomocą frameworku Spring MVC, który pozwala na łatwe tworzenie skalowalnej aplikacji umożliwiając m.in. korzystania z mechanizmu Dependency Injection.

W module „Model” znajdują się elementy związane z zarządzaniem danymi w aplikacji. Moduł ten udostępnia funkcjonalności związane z bazą danych (CRUD) oraz osobny podmoduł służący do zarządzania danymi pobranymi z ze strony Internetowej. Użytymi technologiami są tutaj przede wszystkim JPA oraz jego implementacja Hibernate. Są to biblioteki zarządzające persystencją danych, czyli tworzeniem, odczytem, aktualizacją oraz usuwaniem encji bazodanowych. W tym module jest również wykorzystane narzędzie JSoup służące do parsowania kodu html. To za jego pomocą aplikacja ma możliwość uzyskania danych znajdujących się na stronie internetowej.

Moduł „Controller” jest odpowiedzialny za logikę biznesową aplikacji. Znajdują się w nim abstrakcyjne „usługi”, które umożliwiają zastosowanie zaprojektowanych funkcjonalności związanych z aplikacją. Moduł ten łączy pozostałe moduły, będąc „interfejsem” pomiędzy warstwą danych, a warstwą prezentacji. W tym module został zaimplementowany proces ETL. Każdy z kroków tego procesu ma swój osobny „podmoduł” oraz dla każdego z nich jest stworzona osobna „usługa” będący zarządzanym JavaBean’em, kontrolowanym przez kontener Springa. Usługi te udostępniają metody, za pomocą których realizowany jest cały proces ETL.

Warstwą prezentacji zarządza moduł „View”. Moduł ten umożliwia interakcję użytkownika z aplikacją. W nim zawierają się elementy służące do pobierania od użytkownika żądań dla aplikacji oraz prezentowania rezultatów tych żądań. Jako, że aplikacja jest aplikacją webową, warstwa ta opiera się na tworzeniu stron internetowych udostępniających zaprojektowane funkcjonalności. Narzędziem, które posłużyło do budowy tej warstwy jest m.in. JSF. Umożliwia on tworzenie plików .xhtml, w których zawarta jest część statyczna oraz dynamiczna, tworzona za każdym razem w zależności od odpowiedzi z serwera. Dodatkowo została wykorzystana biblioteka Primefaces, która udostępnia wiele gotowych komponentów do budowy dynamicznej strony internetowej (pola „input”, listy rozwijane, kalendarze do wyboru daty, tabele, panele, itp.)

# Wymagania techniczne i instalacja środowiska

## JDK 1.7

* Instalacja

Instalacja środowiska JDK powinna zostać przeprowadzona zgodnie z instrukcją <http://docs.oracle.com/javase/7/docs/webnotes/install/index.html> dla wybranego systemu.

Należy też utworzyć zmienną systemową JAVA\_HOME wskazującą na katalog z zainstalowanym JDK.

## Baza danych PostgreSQL 9.3

* Instalacja

Instalacja serwera powinna zostać przeprowadzona zgodnie z instrukcją <https://wiki.postgresql.org/wiki/Detailed_installation_guides> dla wybranego systemu.

* Konfiguracja

Po poprawnej instalacji serwera należy utworzyć nową bazę danych o nazwie „findata” oraz nowego użytkownika o nazwie „findata” z hasłem dostępu „findata123”. Sposób postępowania znajduje się w dokumentacji serwera PostgreSQL.

Następnie, dla tak przygotowanej bazy, należy uruchomić skrypt tworzący schemat bazy danych dla aplikacji. Skrypt znajduje się w pliku:

{APP}/sources/db/create\_schema.sql

## Serwer aplikacyjny JBoss EAP 6.2

* Instalacja

Instalacja serwera aplikacyjnego powinna być przeprowadzona zgodnie z instrukcją: [https://access.redhat.com/site/documentation/en-US/JBoss\_Enterprise\_Application\_Platform/6/html/Installation\_Guide/chap-Installing\_JBoss\_Enterprise\_Application\_Platform\_6.html](https://access.redhat.com/site/documentation/en-US/JBoss_Enterprise_Application_Platform/6/html/Installation_Guide/chap-Installing_JBoss_Enterprise_Application_Platform_6.html%20) dla wybranego systemu.

* Konfiguracja

Skopiować katalog {APP}/sources/conf/postgresql do katalogu {JBOSS\_HOME}/modules/system/layers/base/org ({APP} wskazuje na rozpakowane archiwum z aplikacją, a {JBOSS\_HOME} wskazuje na katalog z serwerem na dysku)

W pliku {JBOSS\_HOME}/standalone/configuration/standalone należy dodać w sekcji

<subsystem xmlns="urn:jboss:domain:datasources:1.1">

następujące wpis:

<xa-datasource jta="true" jndi-name="java:/findata" pool-name="PostgresDS" enabled="true" use-java-context="true">

<xa-datasource-property name="ServerName">

localhost

</xa-datasource-property>

<xa-datasource-property name="PortNumber">

5432

</xa-datasource-property>

<xa-datasource-property name="DatabaseName">

findata

</xa-datasource-property>

<driver>postgresql</driver>

<security>

<user-name>findata</user-name>

<password>findata123</password>

</security>

</xa-datasource>

oraz:

<drivers>

<driver name="postgresql" module="org.postgresql">

<xa-datasource-class>org.postgresql.xa.PGXADataSource</xa-datasource-class>

</driver>

</drivers>

Następnie należy dodać użytkownika administracyjnego. W tym celu należy uruchomić w konsoli polecenie add-user w katalogu {JBOSS\_HOME}/bin i postępować zgodnie z poleceniami. Nazwa i hasło mogą być dowolne, spełniające wymagania serwera.

## Maven 3.2.1

* Instalacja

Instalacja serwera powinna zostać przeprowadzona zgodnie z instrukcją <http://maven.apache.org/download.cgi#Installation> dla wybranego systemu.

# Wdrożenie aplikacji

## Zbudowanie paczki \*.war

Aby stworzyć paczkę aplikacji webowej (.war) należy skorzystać z narzędzia Maven. W tym celu uruchamiamy konsolę i przechodzimy do katalogu {APP}/sources/src/findata (gdzie {APP} jest folderem z rozpakowaną aplikacją), w której znajdują się źródła aplikacji. Następnie wykonujemy polecenie:

mvn clean install

Gdy proces budowania paczki zostanie pomyślnie zakończony, w katalogu {APP}/sources/src/findata/target zostanie utworzony plik „findata.war”.

## Wdrożenie aplikacji na serwer

Po utworzeniu paczki .war uruchamiamy serwer Jboss poleceniem

{JBOSS\_HOME}/bin/standalone

Następnie uruchamiamy przeglądarkę internetową i przechodzimy na adres <http://localhost:9990>. Jest to system zarządzania serwerem. Wybieramy z menu po lewej strony pozycję „Manage Deployments”, następnie klikamy w przycisk „Add”, wybieramy utworzoną wcześniej paczkę .war i potwierdzamy nasz wybór. Po zamknięciu okna dialogowego, na liście pojawia się wybrana wcześniej paczka .war. W celu uruchomienia aplikacji należy zaznaczyć tą paczkę oraz nacisnąć przycisk „En/Disable” oraz potwierdzić.

Gdy proces uruchamiania aplikacji zakończy się pomyślnie, przechodzimy na stronę <http://localhost:8080/findata>.

# Opis funkcjonalny

Aplikacja Findata-Importer jest aplikacją implementującą proces **ETL**. Proces ten składa się z 3 kroków: **E**xtract, czyli pobranie danych ze źródła, **T**ransform, czyli konwersja danych z formatu źródłowego do formatu wyjściowego oraz **L**oad, czyli umieszczenie przekonwertowanych danych w bazie.

Pierwszym krokiem po uruchomieniu aplikacji jest wybór przez użytkownika, czy proces ETL zrealizować jako całość, czy w osobnych krokach. Wybór ten jest możliwy poprzez wybranie jednej z dwóch opcji w panelu widocznym na stronie głównej. Każda z tych opcji aktywuje następny panel.

Jeśli użytkownik wybiera opcję realizacji całego procesu, zostaje pokazany panel z przyciskiem do aktywacji procesu. Po naciśnięciu tego przycisku następuje wywołanie usługi w module „Controller”, która sama wywołuje usługi wszystkich trzech procesów, jeden po drugim. Jeśli proces zostanie zakończony pomyślnie, aktywowany jest panel z rezultatem wykonania procesu. W panelu tym znajduje się przede wszystkim tabela, która prezentuje notowania giełdowe zapisane do bazy danych. Oprócz tego, użytkownik dostaje informacje na temat nowych spółek, które pojawiły się w danej giełdzie od ostatniego uruchomienia procesu.

Jeśli użytkownik wybiera opcję realizacji procesu w osobnych krokach, zostaje pokazany panel z trzema przyciskami: „Extract”, „Transform” oraz „Load”. Przyciski te aktywują odpowiadający krok w procesie ETL.

1. Po naciśnięciu przycisku „Extract” następuje wywołanie usługi „Extract” z modułu „Controller”. Jeśli proces został zakończony pomyślnie, zostaje aktywowany panel z rezultatem procesu. Jest to tabela zawierająca dane pobrane ze strony internetowej <http://findata.co.nz/>, które nie zostały jeszcze przetransformowane.
2. Po naciśnięciu przycisku „Transform” następuje wywołanie usługi „Transform” z modułu „Controller”. Usłudze tej przekazywana jest lista danych z poprzedniego procesu. Jeśli proces został zakończony pomyślnie, zostaje aktywowany panel z rezultatem procesu. Są to tabele zawierające przetransformowane dane do encji bazodanowych.
3. Po naciśnięciu przycisku „Load” następuje wywołanie usługi „Load” z modułu „Controller”. Usłudze tej jest przekazywana lista encji bazodanowych utworzonych w poprzednim procesie. Dane są zapisywane do bazy danych. Jeśli proces został zakończony pomyślnie, zostaje aktywowany panel z rezultatem procesu. Jest to tabela z notowaniami giełdowymi wprowadzonymi do bazy danych w tym procesie. Dodatkowo są zaprezentowane dane dotyczące zmian w liście spółek lub giełd w stosunku do ostatniego wywołania procesu.

Aby zapewnić spójność procesu, odpowiednie przyciski są odblokowywane dopiero kiedy ich użycie ma sens, to znaczy, jeśli dane nie zostały.

Każda z dwóch opcji realizacji procesu ETL może zostać zresetowana i rozpoczęta od nowa za pomocą przycisku „Reset”.

# Dokumentacja kodu

Dokumentacja kodu w formacie javadoc jest umieszczona pod adresem:

<http://wizard.uek.krakow.pl/~s176509/doc/findata/>