Paweł Dymora, Mirosław Mazurek, Bogumił Mroczka

Wykorzystanie multimedialnych rozszerzeń baz danych w dydaktyce przedmiotów informatycznych

Edukacja - Technika - Informatyka nr 3(13), 308-322

2015

Paweł DYMORA, Mirosław MAZUREK, Bogumił MROCZKA
Politechnika Rzeszowska, Polska

Wykorzystanie multimedialnych rozszerzeń baz danych w dydaktyce przedmiotów informatycznych

Wstęp

Początkowo bazy danych służyły do przechowywania głównie alfanumerycznych danych. Były one przechowywane w postaci plików zapisywanych na dysku twardym. Po pewnym czasie rozwiązanie to było niewystarczające, ponieważ wraz ze wzrostem ilości danych pojawiały się problemy z ich przetwarzaniem, co doprowadziło do powstania relacyjnych baz danych pozwalających na minimalizację niepotrzebnego powtarzania się tych samych danych w bazie poprzez zastosowanie powiązań (relacji) pomiędzy zbiorami danych. Obecnie wraz z rozwojem systemów informatycznych zachodzi potrzeba składowania zaawansowanych struktur danych o złożonych typach. Następstwem tego jest rozwój systemów baz danych w kierunku umożliwiającym realizację nowoczesnych rozwiązań. Pojawiają się nowe obiektowe typy danych, jak również typy umożliwiające składowanie danych multimedialnych, tj. zdjęć, muzyki czy filmów. Zmienia się również sposób przetwarzania danych. W prostych systemach wyszukiwanie danych polegało na wybieraniu rekordów, które spełniały określone warunki. W przypadku danych o charakterze multimedialnym zadanie to staje się nietrywialne. Większość dużych producentów systemów bazodanowych wspiera obsługę typów multimedialnych w zakresie przechowywania dużych obiektów binarnych, a także ich przetwarzania i wyszukiwania. Zasady te definiuje standard SQL/MM, który w największym stopniu zaimplementowany został w Oracle, gdzie istnieją m.in. mechanizmy wyszukiwania obrazów na podstawie zawartości. Implementacja tych mechanizmów na poziomie systemu bazodanowego w znaczącym stopniu może odciążyć programistę aplikacji bazodanowych od samodzielnego tworzenia kodu obsługi tych specyficznych rozwiązań. Dlatego znajomość pełnych możliwości, jakie oferują współczesne systemy bazodanowe, jest istotna z punktu widzenia dydaktyki przedmiotów bazodanowych oraz programistycznych. W artykule jako przykład edukacyjny zaprezentowano możliwość budowy interaktywnej aplikacji internetowej opartej na relacyjno-obiektowej bazzie danych wykorzystującej multimedialne typy danych dostępne w bibliotekach Oracle Multimedia oraz technologii JSP [Dymora i in. 2014].

Charakterystyka multimedialnych typów danych w Oracle Multimedia

W pierwszych implementacjach systemy bazodanowe Oracle umożliwiały przechowywanie plików multimedialnych jako typ `BFILE`. Rozwiązanie to polegało na przechowywaniu plików poza bazą danych (w systemie operacyjnym), a w tabeli znajdowała się kolumna typu `BFILE z` referencjami do plików. Obiekt typu `BFILE` umożliwia odczytanie zawartości pliku, sprawdzenie, czy plik istnieje i ile miejsca na dysku zajmuje. Składowanie plików jako `BFILE` umożliwia jedynie transakcyjne przetwarzanie „wskaźnika" do lokalizacji pliku, a nie danych przechowywanych na dysku, dlatego został opracowany nowy typ danych, który umożliwia przechowywanie danych binarnych lokalnie w bazie danych – `BLOB`. Pliki przechowywane w bazie danych jako `BLOB` mogą być już nie tylko odczytywane, ale również przetwarzane. Dalszy rozwój multimedialnych systemów bazodanowych doprowadził do powstania multimedialnych typów obiektowych, które oprócz danych binarnych `LOB` przechowują też informacje opisujące pliki multimedialne, tzw. metadane [Price 2009]. Oracle Database 11g udostępnia specjalne obiektowe typy danych, które umożliwiają składowanie danych binarnych wraz z metadanymi oraz ich przetwarzanie z poziomu proceduralnego języka PL/SQL lub za pomocą aplikacji internetowej wykorzystującej multimedialne typy obiektowe dostępne w bibliotekach Oracle Multimedia [Dymora i in. 2014]. Oracle obsługuje metadane w formacie `EXIF` (ang. *Exchangeable Image File Format*), `IPTC-IIM` (ang. *International Press Telecommunications Council-Information Interchange Model*) oraz `XMP` (*extensible metadata platform*). Typ obiektowy `OrdImage` umożliwia składowanie i przetwarzanie danych graficznych w bazie danych. Udostępnia wiele atrybutów (np. `Source`, `Height`, `Width`, `contentLength`, `fileFormat`, `contentFormat`, `mimeType`, `compressionFormat`), w których przechowywane są metadane obrazu. Posiada także metody służące do ekstrakcji metadanych i przetwarzania obrazów (skalowanie, przycinanie itp.). Operacje, jakie można wykonać na obrazach za pomocą odpowiednich metod obiektu `OrdImage`, to: `compressionFormat`, `Cut`, `fixedScale`, `Mirror`, `Rotate`, `Scale`, `xScale`, `yScale`. Kolejny typ obiektowy udostępniany przez Oracle Multimedia to `OrdVideo`, który umożliwia składowanie i zarządzanie danymi wideo. W skład tego typu wchodzą atrybuty przechowujące metadane o pliku wideo, metody umożliwiające odczytywanie tych metadanych oraz metody służące do przetwarzania danych wideo. Typ obiektowy `OrdVideo` (podobnie jak `OrdImage`) posiada przeładowany konstruktor, który może być wykorzystany do zapisywania nowych rekordów w bazie danych z poziomu języka PL/SQL. Podczas zapisywania obiektu `OrdVideo` do bazy danych przy użyciu polecenia `insert` lub `update` dane przechowywane w `BLOB` są kopiowane do obiektu `OrdVideo` [Price 2009].

Listing 1. Konstruktor obiektu typu `OrdVideo` dla danych `BLOB`

```
ORDVideo(SELF IN OUT NOCOPY ORDSYS.ORDVideo,
      data IN BLOB,setproperties IN INTEGER DEFAULT 0)
RETURN SELF AS RESULT
```

Parametry konstruktora to: `data` – dane wideo przechowywane w `BLOB`, `setproperties` – flaga decydująca o tym, czy metoda `setProperties()` ma być wywołana w konstruktorze. Jeśli ten parametr będzie miał wartość 1, to metoda zostanie wywołana, jeśli parametr będzie miał wartość 0 (domyślna), metoda nie zostanie wywołana.

Listing 2. Konstruktor obiektu typu `OrdVideo` dla danych `BFILE`

```
ORDVideo( SELF IN OUT NOCOPY ORDSYS.ORDVideo,
      source_type IN VARCHAR2 DEFAULT 'LOCAL',
      source_location IN VARCHAR2 DEFAULT NULL,
      source_name IN VARCHAR2 DEFAULT NULL,
      setproperties IN INTEGER DEFAULT 0 )
RETURN SELF AS RESULT
```

Powyższy konstruktor tworzy nowy obiekt `OrdVideo` z określonego źródła danych. Jeśli źródło nie zostanie określone, domyślnie zostanie utworzony pusty obiekt `OrdVideo` z lokalnym źródłem danych. Parametry konstruktora: `source` – typ źródła danych. Poprawne wartości: `FILE`, `HTTP`, `LOCAL`, `user-defined`. `source_location` – katalog, z którego plik wideo zostanie zaimportowany do obiektu `OrdVideo`, `source_name` – nazwa źródłowego pliku wideo, `setproperties` – flaga decydująca o tym, czy metoda `setProperties()` ma być wywołana w konstruktorze [Geenwald i in. 2009].

Technologie wykorzystane do budowy aplikacji

Podstawą całego środowiska uruchomieniowego projektowanej aplikacji jest serwer bazy danych Oracle, który jest jednym z najbardziej zaawansowanych systemów bazodanowych, jeżeli chodzi o składowanie, przetwarzanie i zarządzanie danymi multimedialnymi. Jedną z cech baz danych Oracle od wersji 10g są Oracle Multimedia. W skład Oracle Multimedia wchodzą multimedialne typy obiektowe służące do składowania i przetwarzania danych, a także biblioteki JAVA umożliwiające programistom przygotowanie aplikacji internetowych służących do zarządzania multimediami przechowywanymi w bazie danych Oracle'a [Geenwald i in. 2009]. Aplikacje te są dostępne z poziomu przeglądarki internetowej w wersji dla komputerów, jak również urządzeń mobilnych (np. tablet, telefon komórkowy). Oracle udostępnia również programi-

stom różne narzędzia wspomagające projektowanie i konfigurację bazy danych, np. Oracle SQL Developer. Serwery bazy danych Oracle są również przystosowane do składowania i przetwarzania danych multimedialnych stosowanych w medycynie. Służą do tego specjalne typy danych oraz biblioteki wykorzystywane do tworzenia aplikacji [Beynon-Davies 2003].

Oracle Glassfish Server jest serwerem aplikacji kompatybilnym z JavaEE, stworzonym na licencji *open source*. Serwer aplikacji Glassfish może być zainstalowany razem z platformą programistyczną NetBeans IDE, co umożliwia zarządzanie serwerem z poziomu IDE oraz automatyczne instalowanie i uruchamianie tworzonych aplikacji webowych. Istnieje również możliwość zainstalowania serwera Glassfish bez środowiska programistycznego. Oracle Glassfish Server udostępnia konsolę administracyjną uruchamianą w przeglądarce internetowej, która instalowana jest razem z serwerem i dostępna na porcie 4848. Narzędzie to umożliwia zatrzymanie serwera oraz zarządzanie użytkownikami, aplikacjami i modułami webowymi oraz pulami połączeń, które mogą być wykorzystywane przez aplikacje do nawiązywania połączenia z bazą danych. Administrowanie serwerem Glassfish jest możliwe również z poziomu linii poleceń przy użyciu specjalnych narzędzi, takich jak `asadmin`, `appclient`, `schemagen`, `wsgen` i in. [Jendrock i in. 2012].

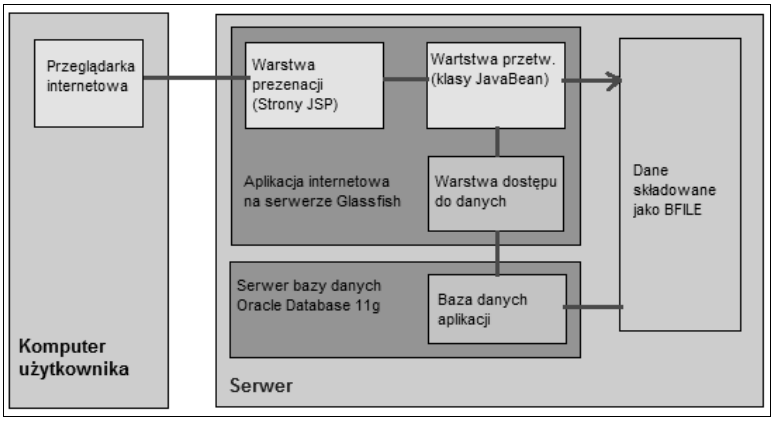
Technologia JavaServer Pages (JSP) umożliwia umieszczanie fragmentów kodu serwletu bezpośrednio w dokumencie tekstowym. Strona JSP jest plikiem tekstowym z rozszerzeniem `.jsp`, który zawiera dwa rodzaje tekstu: dane statyczne w dowolnym formacie tekstowym (np. HTML, XML) oraz elementy JSP, które określają sposób konstruowania dynamicznej treści strony. Pliki JSP podczas pierwszego wyświetlania są kompilowane do serwletów, które są następnie uruchamiane po stronie serwera i generują dynamiczną treść strony w formacie tekstowym. Przy kolejnym odwołaniu do strony JSP uruchamiany jest skompilowany serwlet. Serwlety to klasy języka Java, które przetwarzają żądania użytkownika i generują odpowiedzi [Jendrock i in. 2012]. Zastosowanie obiektów JavaBeans ułatwia oddzielenie widoku od logiki przetwarzania lub formatowania, ponieważ tego typu zadania są wykonywane dla widoku przez komponenty pomocnicze. Tworzenie komponentów JavaBeans i ich wykorzystanie w środowisku JSP jest łatwiejsze i wymaga mniej pracy niż tworzenie własnych znaczników pomocniczych. Zastosowanie takiej strategii pozwala na łatwiejsze zarządzanie aplikacją, ponieważ cała logika przetwarzania znajduje się w obiektach JavaBeans, a strony JSP wywołują jedynie odpowiednie metody tych obiektów, pobierają wygenerowane dane i wyświetlają je w przeglądarce [Alur i in. 2004].

Projekt interaktywnej aplikacji bazodanowej wykorzystującej typy multimedialne

W niniejszym rozdziale zaprezentowana zostanie aplikacja wykorzystująca omawiane mechanizmy. Będzie ona zrealizowana z wykorzystaniem języka

skryptowego JSP i dostępna z poziomu przeglądarki internetowej po uprzednim zalogowaniu. W aplikacji dostępne będą dwa poziomy uprawnień: Użytkownik i Administrator. Administrator aplikacji będzie mógł tworzyć nowe konta użytkowników i przypisywać im odpowiedni poziom uprawnień, edytować dane użytkowników oraz usuwać konta. Ponadto, będzie miał możliwość wyświetlenia zawartości tabel bazy danych, ale nie będzie mógł ich edytować lub usuwać z poziomu aplikacji. Użytkownik aplikacji będzie mógł tworzyć albumy, w których będą zapisywane zdjęcia i pliki wideo. Aplikacja będzie wyświetlać użytkownikowi tylko jego własne albumy, dodanie plików do albumów innych użytkowników nie będzie możliwe. Pliki będzie można wysyłać przy użyciu formularza dostępnego w aplikacji i będą przechowywane w odpowiednich tabelach jako obiektowe typy multimedialne [Horstmann, Cornell 2009].

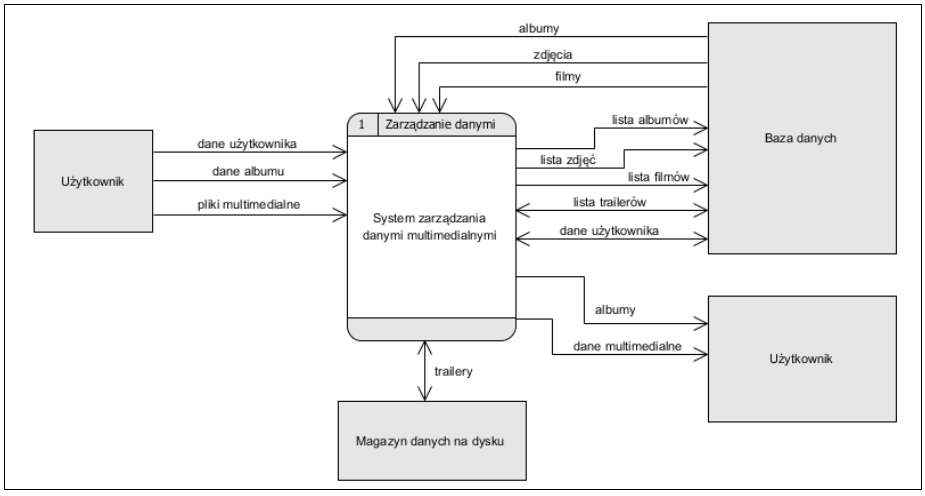
Dla każdego zdjęcia będą dostępne następujące operacje: podgląd, wyświetlenie metadanych, edycja, usunięcie. Dla plików wideo będą dostępne opcje: wyświetl, szczegóły (wyświetlanie metadanych), usuń. Użytkownik będzie mógł edytować zdjęcia przechowywane w bazie danych. Aplikacja będzie udostępniać dwa sposoby zapisu zmian: aktualizacja edytowanego zdjęcia (Zapisz), zapis zmodyfikowanego zdjęcia jako nowe w wybranym albumie (Zapisz jako). Pliki wideo przechowywane w bazie danych będzie można wyświetlać w aplikacji internetowej przy użyciu odtwarzacza wbudowanego w przeglądarkę. Użytkownik będzie mógł dodawać do bazy danych trailery planowanych wycieczek w postaci plików wideo. Pliki te będą przechowywane poza bazą danych jako `BFILE`. Trailery będą mogły być wyświetlane w aplikacji przy użyciu wbudowanego odtwarzacza. Ze względu na wielkość projektu oraz zastosowanie technologii JSP do stworzenia aplikacji zastosowano wzorzec `ViewHelper`.



Rys. 1. Model projektowanej aplikacji

Warstwa prezentacji (widok) składa się ze stron JSP znajdujących się w katalogach: `web\common` – strony wspólne dla wszystkich użytkowników, np. komuni-

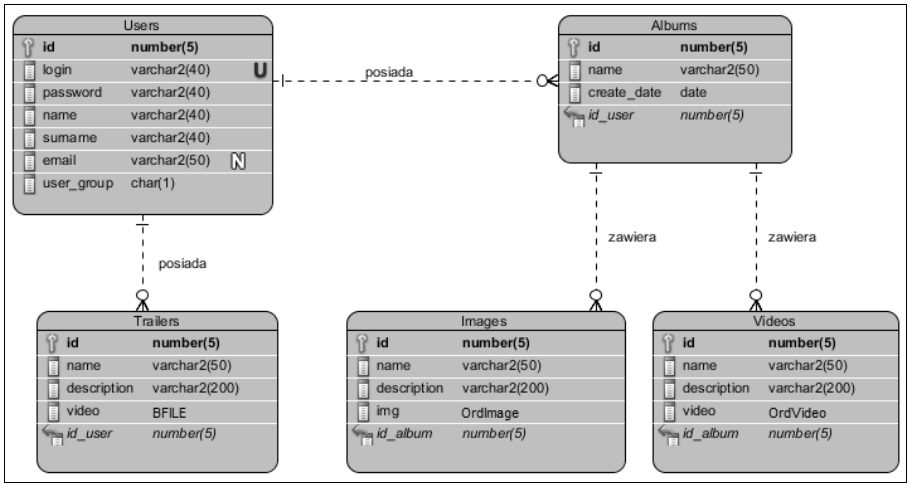
katy o błędach, `web\secure\user` – strony użytkownika, `web\secure\admin` – strony administratora aplikacji. Warstwa przetwarzania (helper) składa się z klas JavaBean znajdujących się w katalogu `src\java\bean`. Za komunikację z bazą danych odpowiada warstwa dostępu do danych, która składa się z serwisów oraz obiektów DTO [Horstmann, Cornell 2009]. Na rys. 1 przedstawiono model projektowanej aplikacji. Użytkownik łączy się z serwerem aplikacji, na którym znajduje się aplikacja internetowa. Warstwa widoku przejmuje żądania użytkownika i kieruje przetwarzanie do obiektów JavaBean, które pobierają lub zapisują dane w bazie przy użyciu metod serwisów z warstwy dostępu do danych i przygotowują dane dla widoku. Strony JSP pobierają następnie dane z obiektów JavaBean i wyświetlają je użytkownikowi. W przypadku uploadu pliku składowanego jako BFILE warstwa przetwarzania tworzy nowy plik w katalogu na dysku, natomiast warstwa dostępu do danych dodaje nowy rekord do tabeli [Wilson 2006]. Rysunek 2 przedstawia kontekstowy diagram przepływu danych, gdzie możemy wyróżnić jeden proces reprezentujący całą aplikację, a także przepływ danych pomiędzy aplikacją i magazynami danych oraz użytkownikiem zalogowanym do aplikacji. Na diagramie zostały wyszczególnione dwa magazyny danych: system bazodanowy oraz plikowy magazyn danych przechowujący dane składowane poza bazą danych w systemie plików serwera. Baza danych aplikacji przechowuje dane multimedialne na dwa sposoby.



Rys. 2. Kontekstowy diagram przepływu danych

W przypadku zdjęć i plików wideo przypisanych do albumu dane przechowywane są w tabelach bazy danych przy użyciu multimedialnych typów Oracle. Trailery natomiast składowane są poza bazą danych jako BFILE – w tabeli bazy danych znajduje się odwołanie do plików na dysku. Poniżej przedstawiono pole-

cenia SQL umożliwiające stworzenie tabel do przechowywania danych dla obu przypadków. Szczegółowy diagram ERD [Jakieła, Litwin 2011] bazy danych przedstawia rys. 3 oraz skrypty z listingów nr 3.



Rys. 3. Diagram ERD

Typy multimedialne Oracle umożliwiają wykonywanie różnych operacji na danych przechowywanych w bazie danych, m.in. istnieje możliwość ekstrakcji metadanych. W celu odczytania podstawowych metadanych obrazu można użyć następującego polecenia SQL z listing nr 4.

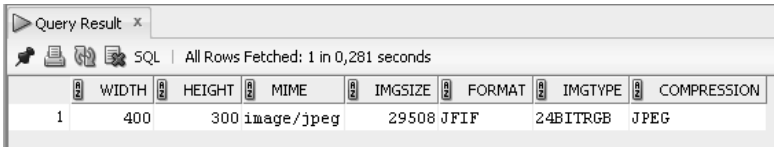
Listing 3. Tworzenie tabel dla danych multimedialnych przechowywanych w bazie danych

```
CREATE TABLE IMAGES (
  ID NUMBER(5) NOT NULL,
  NAME VARCHAR2(50) NOT NULL,
  DESCRIPTION VARCHAR2(200) NOT NULL,
  IMG ORDSYS.ORDIMAGE,
  ID_ALBUM NUMBER(5) NOT NULL,
  CONSTRAINT IMAGES_PK PRIMARY KEY ( ID ) ENABLE);

CREATE TABLE VIDEOS (
  ID NUMBER(5) NOT NULL,
  NAME VARCHAR2(50) NOT NULL,
  DESCRIPTION VARCHAR2(200) NOT NULL,
  VIDEO ORDSYS.ORDVIDEO,
```

```
  ID_ALBUM NUMBER(5) NOT NULL,
  CONSTRAINT VIDEOS_PK PRIMARY KEY (ID) ENABLE);
```

Listing 4. Ekstrakcja metadanych typów multimedialnych

```
select i.img.getWidth() WIDTH,
       i.img.getHeight() HEIGHT,
       i.img.getMimeType() MIME,
       i.img.getContentLength() IMGSIZE,
       i.img.getFileFormat() FORMAT,
       i.img.getContentFormat() IMGTYPE,
       i.img.getCompressionFormat() COMPRESSION
from images i where i.id = &id;
```



Multimedialny typ obiektowy `OrdImage` umożliwia również edycję danych przechowywanych w bazie danych. Obiekt typu `OrdImage` posiada metodę `process()` pozwalającą na wykonywanie różnych operacji na zdjęciu. Poniżej w listingu nr 5 przedstawiono `kod PL/SQL` procedury składowanej edytującej zdjęcie przechowywane w tabeli tymczasowej.

Listing 5. Manipulacja obiektami multimedialnymi

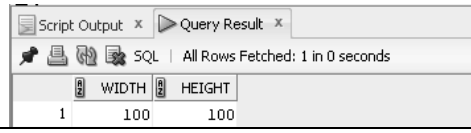
```
CREATE OR REPLACE PROCEDURE editTmpImage (
  command in varchar2,
  tmpImgId in tmpimages.id%TYPE) IS
   tmpImage ORDSYS.ORDIMAGE;
BEGIN
  select ti.img into tmpImage from tmpimages ti where id
         = tmpImgId for update;
  tmpImage.process(command);
  update tmpimages set img = tmpImage where id =
  tmpImgId;
  commit;
```

```
END editTmpImage;

execute editTmpImage('fixedScale=100 100', 85);
```

Powyższe wywołanie procedury składowanej spowoduje przeskalowanie wybranego obrazka do podanego rozmiaru. Aby sprawdzić, czy procedura wykonała się poprawnie, można odczytać jego nowy rozmiar z metadanych:

Listing 6. Przeglądanie właściwości transformowanego obiektu

```
select i.img.getWidth() WIDTH, i.img.getHeight() HEIGHT
from tmpimages i where i.id = 85;
```



Wybrane moduły utworzonej aplikacji

W aplikacji wyróżnić można następujące warstwy logiczne, a mianowicie **warstwę prezentacji** odpowiadającą za wyświetlanie informacji w przeglądarce internetowej. Składa się ona z plików JSP umieszczonych w odpowiednich katalogach w zależności od uprawnień dostępu oraz arkuszy stylów i obrazków. Warstwa prezentacji komunikuje się z warstwą przetwarzania, która obsługuje żądania użytkownika i generuje dane wynikowe. Dane te są następnie pobierane i wyświetlane w przeglądarce [Alur i in. 2004]. Pliki warstwy prezentacji znajdują się w następujących katalogach projektu:

- `web\common` – zawiera części wspólne dostępne zarówno dla użytkowników zalogowanych, jak i niezalogowanych, np. ekran logowania, komunikaty o błędach,
- `web\secure\admin` – zawiera widoki administratora aplikacji,
- `web\secure\user` – zawiera widoki użytkownika aplikacji,
- `web\css` – arkusze stylów,
- `web\img` – pliki graficzne ładowane do widoków.

Warstwa przetwarzania złożona jest ze specjalnych klas, tzw. Beanów, które służą do obsługi zapytań użytkownika i generowania wyników. Każdemu widokowi z warstwy prezentacji odpowiada Bean z warstwy przetwarzania, który zawiera metody obsługujące żądania z danego widoku oraz pola przechowujące dane wprowadzane przez użytownika i dane pobrane z bazy danych podczas przetwarzania żądania. Klasy JavaBeans znajdują się w katalogu

\src\java\bean. Warstwa dostępu do danych odpowiada za komunikację z bazą danych i składa się z serwisów znajdujących się w katalogu src\java\service. Obiekty serwisu składają się z pola connection, które przechowuje połączenie z bazą danych, metod do nawiązywania i zamykania połączenia z bazą danych oraz metod służących do wykonywania operacji na bazie danych. Wysyłają one zapytania do serwera bazy danych i przetwarzają odpowiedź serwera, zwracając dane w postaci obiektu transferu danych lub listy takich obiektów. W katalogu src\java\dto znajdują się obiekty transferu danych dla każdej tabeli bazy danych, które zawierają pola odpowiadające kolumnom w tabeli oraz metody umożliwiające pobieranie i ustawianie zawartości tych pól [Alur i in. 2004].

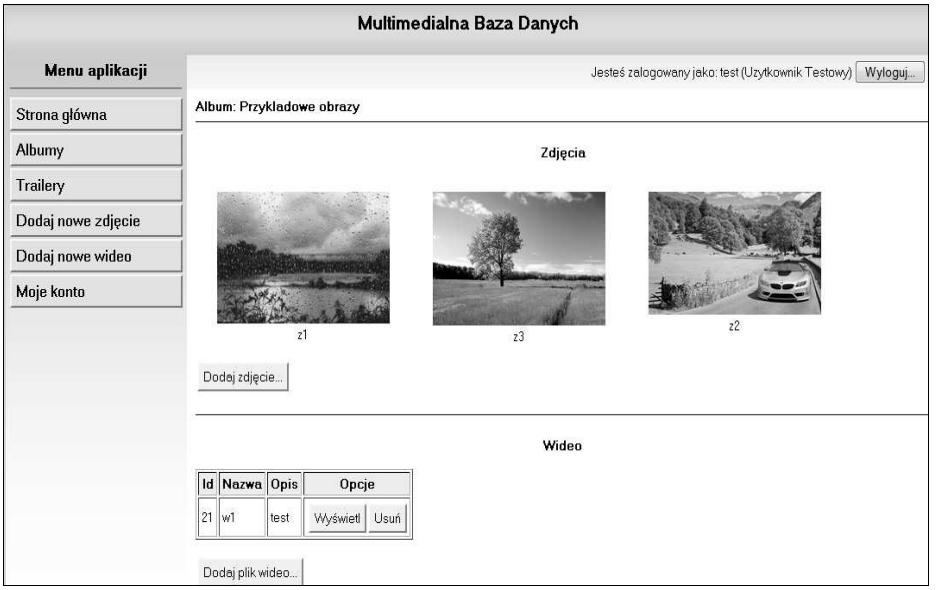
Do nawiązywania połączeń z bazą danych w warstwie dostępu do danych służą klasy, które znajdują się w katalogu src\java\database. Klasa DatabaseConnectionFactory zawiera statyczną metodę getDatabaseConnection(), która odczytuje parametry połączenia z pliku konfiguracyjnego aplikacji, a następnie tworzy nowy obiekt klasy OracleConnection, przekazując w konstruktorze odczytane parametry połączenia, i zwraca null, jeśli połączenie z bazą danych się nie powiedzie, lub obiekt klasy Connection, jeśli połączenie zostało nawiązane [Horstmann, Cornell 2009]. Tworzenie połączenia z bazą danych oraz wykonywanie zapytań SQL realizowane jest za pomocą sterownika JDBC typu Thin, który jest dołączany do aplikacji jako biblioteka odbc6.jar [Price 2009]. Za uwierzytelnianie użytkowników odpowiada klasa serwisowa AuthService. Znajdują się w niej metody login() oraz logOut(). W polu user przechowywane są dane aktualnie zalogowanego do aplikacji użytkownika. Podczas otwierania aplikacji główny plik – index.jsp, który znajduje się w katalogu web i pełni funkcję kontrolera, tworzy obiekt klasy AuthService i zapisuje go w sesji. Jeśli żaden użytkownik nie jest zalogowany, skrypt index.jsp przekierowuje użytkownika do okna logowania. W przeciwnym wypadku następuje przekierowanie pod odpowiedni adres w zależności od poziomu uprawnień (administrator: /secure/admin/?p=main, użytkownik: /secure/user/?p=main). Proces logowania użytkownika polega na sprawdzeniu, czy użytkownik o podanym loginie i haśle istnieje w bazie danych. W tym celu wykonywane jest zapytanie SQL, a następnie sprawdzana jest odpowiedź serwera bazy danych. Jeżeli serwer zwróci dane użytkownika, to oznacza to, że użytkownik istnieje w systemie, a w formularzu został wprowadzony poprawny login i hasło. Pobrane z bazy danych dane użytkownika zapisywane są w polu user obiektu AuthService, który jest przechowywany w sesji. Ze względu na to, że w bazie danych przechowywane są skróty haseł, hasło wprowadzone przez użytkownika przetwarzane jest algorytmem SHA1, a następnie do zapytania wstawiany jest skrót hasła. Do otrzymywania skrótu hasła przekazywanego jako String służy metoda hashPassword(), która znajduje się w klasie

serwisowej `UserService`. Ponadto, mechanizm logowania wzbogacono o dodawanie do hasła losowo generowanej liczby, tzw. `sól`, która zapobiega atakom z wykorzystaniem tęczowych tablic.

Listing 7. Pobieranie listy albumów z bazy (`service.AlbumsService`)

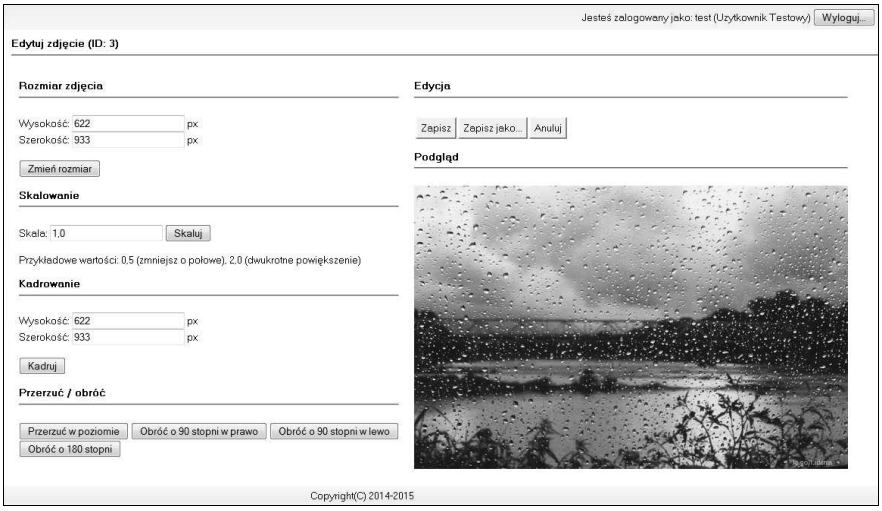
```
public ArrayList<AlbumTO> getAlbumList(Integer userId) {
    ArrayList<AlbumTO> albumList = new ArrayList<>();

    try {
        Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM ALBUMS WHERE ID_USER = " + userId);
        while(rs.next()) {
            AlbumTO albumTO = new AlbumTO();
            albumTO.setFromResultSet(rs);
            albumList.add(albumTO);
        }
        rs.close();
        stmt.close();
    } catch(SQLException ex) {
        Logger.getLogger(getClass().getName()).log(Level.SEVERE, "getAlbumList: SQL Error!", ex);
    }

    return albumList;
}
```

Widok *Albumy* umożliwia wyświetlenie utworzonych albumów. Aplikacja daje możliwość filtrowania listy albumów po nazwie. Dla każdego albumu dostępne są przyciski umożliwiające wyświetlenie zawartości albumu, edycję lub usunięcie. Do pobierania listy albumów z bazy danych służy metoda `getAlbumList()` z serwisu `AlbumsService` przedstawiona w listingu nr 7. Po kliknięciu na przycisku „Otwórz” w widoku *Albumy* wyświetlane są zdjęcia oraz pliki wideo znajdujące się w otwartym albumie (rys. 4). Klikając na zdjęciu, można je otworzyć w widoku *podgląd zdjęcia*, a przycisk „Dodaj zdjęcie” umożliwia przejście do formularza wysyłania zdjęcia na serwer. Poniżej zdjęć znajduje się lista plików wideo, jeśli są w albumie, oraz przycisk „Dodaj plik wideo”, który otwiera formularz wysyłania pliku wideo na serwer. Dla każdego pliku wideo, który znajduje się w wybranym albumie, dostępne są operacje „Otwórz”, „Edytuj” i „Usuń”. Do pobrania danych wyświetlanych w tym widoku wykorzystywane są dwie klasy serwisowe: `ImageService` i `VideoService`. Obiekt JavaBean wywołuje odpowiednie metody tych serwisów w celu załadowania listy zdjęć i plików wideo z bazy danych do pól, z których widok JSP pobiera dane. Do pobrania wideo z bazy danych w celu wyświetlenia go w odtwarzaczu używany jest skrypt JSP, który przyjmuje jako parametr numer `ID` wideo do wyświetlenia. Znajduje się on w katalogu `web\secure\user`. Do parametru `src` odtwarzacza wideo jest podstawiana nazwa skryptu JSP z parametrem, np. `src="./videoFromDb.jsp?id=21"`.



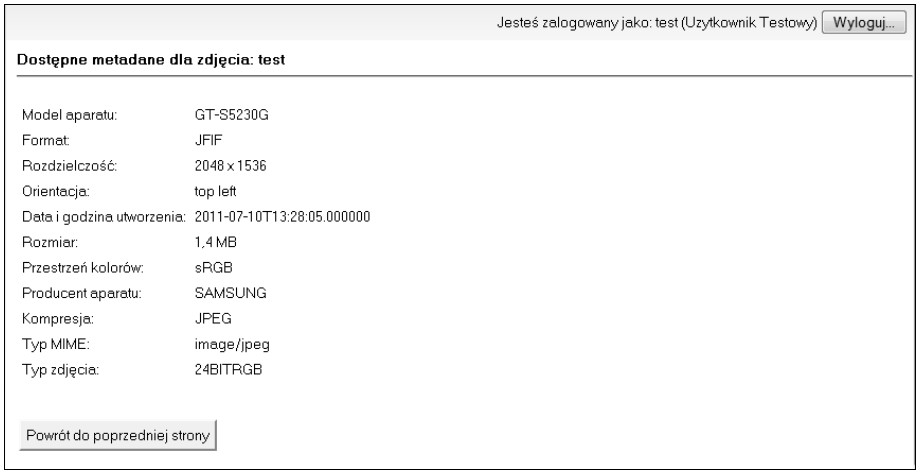
Rys. 4. Widok: podgląd albumu

Widok edycji prezentowany na rys. 5 umożliwia wykonywanie prostych operacji na wybranym zdjęciu: zmiana rozmiaru, skalowanie, kadrowanie, przerzucanie (odbicie lustrzane) i obracanie. Po zakończeniu edycji można zapisać zmodyfikowane zdjęcie jako nowe w wybranym albumie lub zaktualizować edytowane zdjęcie. Użytkownik ma również możliwość anulowania edycji i powrotu do widoku podglądu zdjęcia. Podczas ładowania tego widoku wybrane zdjęcie zostaje skopiowane do tabeli tymczasowej przy użyciu odpowiedniej procedury składowanej. Dzięki temu może być dowolnie edytowane, podczas gdy oryginalne zdjęcie pozostaje niezmodyfikowane. Podczas zapisywania zmian zdjęcie z tabeli tymczasowej jest wstawiane w miejsce oryginału (Zapisz) lub dodawane do tabeli `IMAGES` jako nowy rekord (Zapisz jako). Modyfikowanie tymczasowego zdjęcia polega na wywołaniu procedury składowanej, przekazując jako parametr numer ID zdjęcia tymczasowego (zwrócony przez procedurę kopiującą zdjęcie do tabeli tymczasowej) oraz komendę przetwarzającą. W przypadku, gdy operacja kopiowania zdjęcia do tabeli tymczasowej nie powiedzie się, aplikacja wyświetla odpowiedni komunikat o błędzie. Efektem wystąpienia takiego błędu jest brak możliwości edycji zdjęcia. Użytkownik może wyświetlić metadane zawarte w zdjęciach przechowywanych w bazie danych. Służy do tego opcja „Wyświetl szczegóły” dostępna w widoku podglądu zdjęcia (rys. 6). Aplikacja wyświetla zarówno podstawowe metadane, jak i informacje składowane w formacie XML, jeśli są dostępne. Przycisk „Powrót do poprzedniej strony” umożliwia powrót do widoku podglądu zdjęcia.



Rys. 5. Widok: edycja zdjęcia

Przykład wyświetlania metadanych zdjęcia, w którym dostępne są tylko podstawowe metadane, oraz wyświetlanie informacji o zdjęciu wykonanym przy użyciu telefonu komórkowego, dla którego są dostępne również metadane w formacie XML (EXIF), przedstawia rys. 6.



Rys. 6. Widok: szczegóły zdjęcia, metadane EXIF

Podsumowanie

Współczesne systemy bazodanowe stale rozwijają się, udostępniając nowoczesne technologie składowania danych, m.in. multimedialnych, takich jak zdję-

cia, filmy lub pliki dźwiękowe. Starsze rozwiązania oferowały jedynie przechowywanie plików multimedialnych na dysku poza serwerem bazy danych bez możliwości składowania metadanych. Współczesne rozwiązania pozwalają na przechowywanie dużej ilości danych multimedialnych w tabelach bazy danych, składowanie ich metadanych, a nawet umożliwiają modyfikację przechowywanych danych binarnych, co zostało przedstawione w niniejszym artykule. Zaprezentowana możliwość budowy interaktywnej aplikacji internetowej opartej na relacyjno-obiektowej bazie danych wykorzystującej multimedialne typy danych dostępne w bibliotekach Oracle Multimedia oraz technologii JSP z powodzeniem może być wykorzystana w dydaktyce przedmiotów bazodanowych oraz programistycznych dla uczniów i studentów kierunków informatycznych.

Literatura

Beynon-Davies P. (2003): *Systemy baz danych*, Warszawa.

Alur D., Crupi J., Malks D. (2004): *J2EE. Wzorce projektowe*, Gliwice.

Dymora P., Mazurek M., Maciąg P. (2014): *Wireless Sensor Network Monitoring System Based on Spatial Data Types*, „PAK" nr 10.

Geenwald R., Stackowiak R., Stern J. (2009): *Oracle Database 11g to co najważniejsze*, Warszawa.

Horstmann C.S., Cornell G. (2009): *Java Core. Techniki zaawansowane*, Gliwice.

Jakieła J., Litwin P. (2011): *Bazy danych. Przewodnik architekta informacji*, Rzeszów.

Jendrock E., Evans I., Gollapudi D., Haase K., Srivathsa C. (2012): *JavaEE6. Przewodnik*, Gliwice.

Price J. (2009): *Oracle Database 11g i SQL. Programowanie*, Gliwice.

Wilson G. (2006): *Przetwarzanie danych dla programistów*, Gliwice.

Streszczenie

W artykule przedstawiono właściwości współczesnych baz danych w zakresie mechanizmów składowania i obsługi multimedialnych typów danych. Zaprezentowano możliwość budowy interaktywnej aplikacji internetowej opartej na relacyjno-obiektowej bazzie danych wykorzystującej multimedialne typy danych dostępne w bibliotekach Oracle Multimedia oraz technologii JSP. Tematyka ta może być z powodzeniem wykorzystana w dydaktyce przedmiotów bazodanowych oraz programistycznych.

Słowa kluczowe: nauczanie, przedmioty bazodanowe i programistyczne, typy multimedialne, środowisko Oracle, JSP.

The Database Multimedia Extensions in Teaching Informatics Subjects

Abstract

The paper presents the characteristics of modern databases in terms of storage and handling mechanisms of multimedia data types. Presents the possibility

to build an interactive web applications on the basis of object-oriented relational database that uses multimedia object types available in libraries Oracle Multimedia and JSP programming language. This subject can be successfully used in the teaching the database and programming subjects.

Keywords: teaching, database and programming subjects, multimedia data types, Oracle database system, JSP.