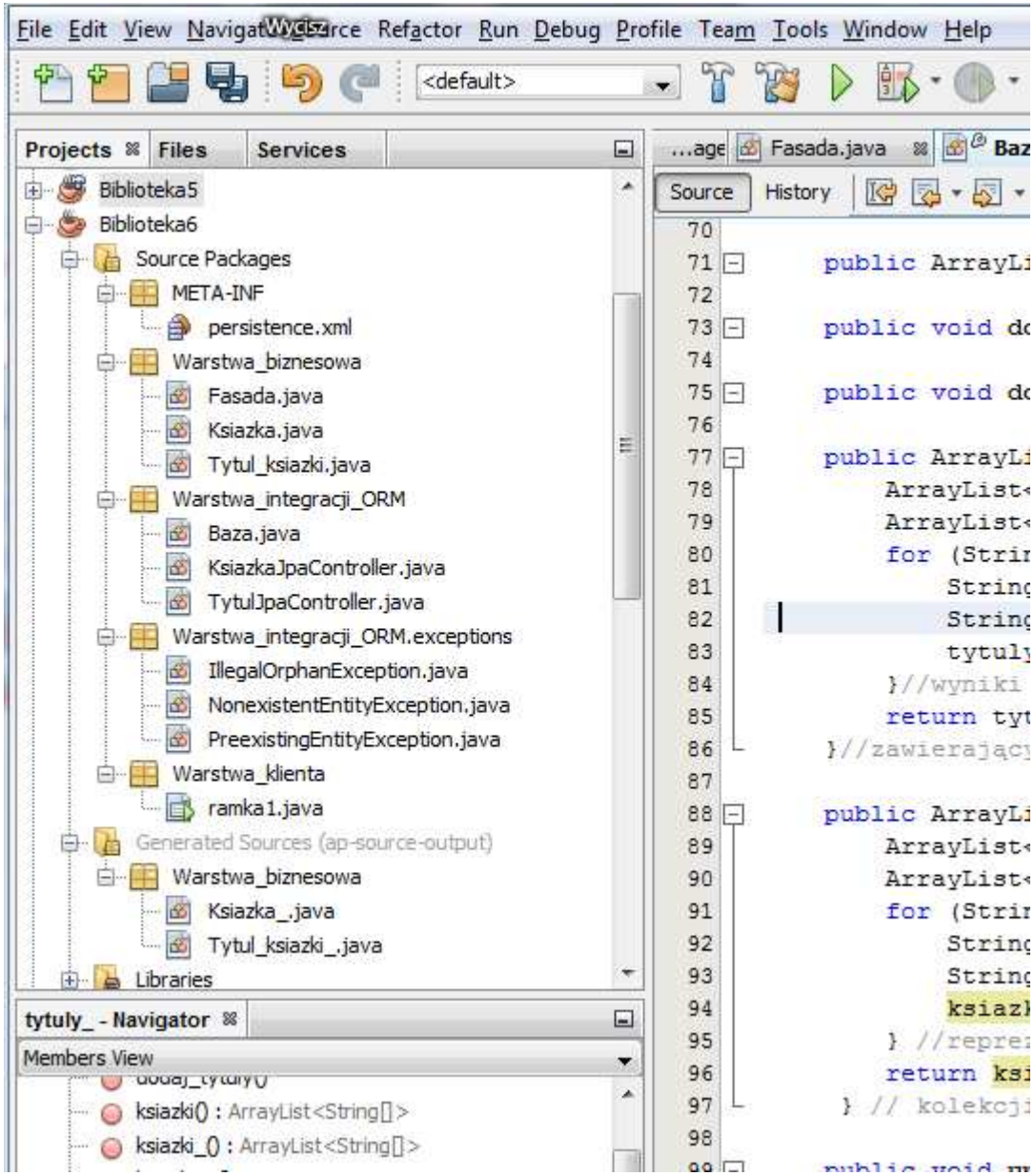
Instrukcja 5
Laboratorium z Podstaw Inżynierii Oprogramowania

Warstwy integracji z bazą danych:

Wzorzec DAO

Technologia ORM

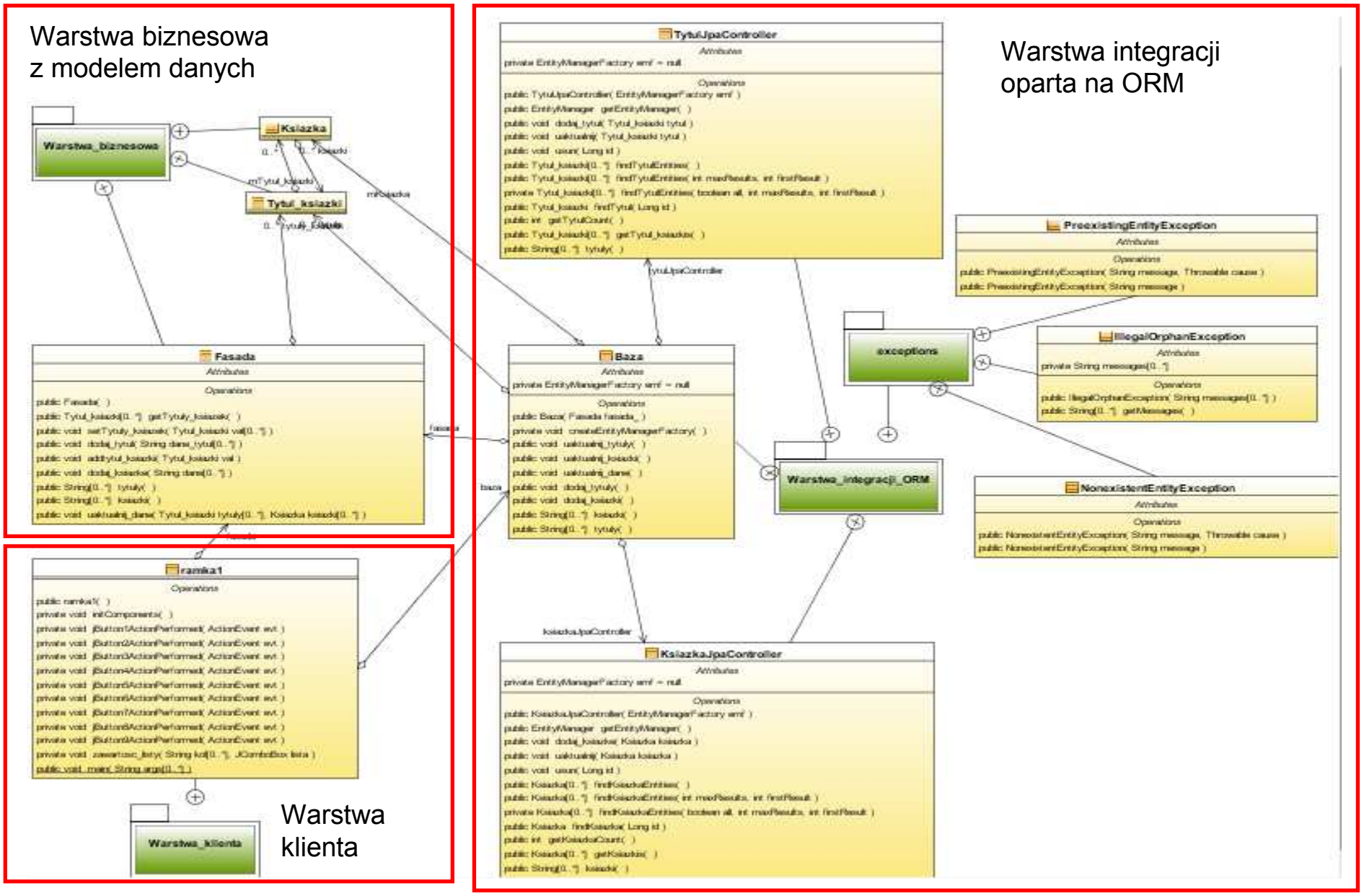
Wykonanie aplikacji trójwarstwowej opartej na wzorcu ORM (JPA)



1. Projekt biblioteka6 – wykonana na kopii programu II z lab4

2. Należy wykonać pustą bazę danych np. Katalog_ksiazek – podobnie jak w poprzednim przykładzie

3. Diagram klas rozmieszczonych w czterech pakietach należących do trzech warstw aplikacji (warstwa integracji typu ORM) – wersja uproszczona



4. Projekt GUI warstwy klienta (program II z lab. 4) – dodano przyciski: **Tytuly do bazy** (zapis tytułów do bazy danych), **Książki do bazy** (zapis książek do bazy), **Tytuly z bazy** (odczyt tytułów z bazy danych i wyświetlenie w liście – Tytuły książek), **Książki z bazy** (odczyt książek z bazy danych i wyświetlenie w liście – Książki), **Dane z bazy** (wykonuje czynności przycisków **Tytuly z bazy** oraz **Książki z bazy**)

5. Definicja klasy ramka – dodano obsługę zdarzeń nowych przycisków: **JButton5-jButton9**

```
package Warstwa_klienta;

import Warstwa_biznesowa.Fasada;
import Warstwa_integracji_ORM.Baza;
import java.util.ArrayList;
import javax.swing.JComboBox;

/**...*/
public class ramka1 extends javax.swing.JFrame {

    private Fasada fasada = new Fasada();
    private Baza baza = new Baza(fasada);

    /**...*/
    public ramka1() {
        try {
            baza.uaktualnij_dane();
        } catch (Exception e) {
            System.out.println(e);
        }
        initComponents();
    }
```

5.1. Część definicji z lab4

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String s1, s2, s3, s4, s5;
    Object zrodlo = evt.getSource();
    if (zrodlo == jButton1) {
        s1 = jTextField1.getText();
        s2 = jTextField2.getText();
        s3 = jTextField4.getText();
        s4 = jTextField5.getText();
        s5 = jTextField6.getText();
        String[] tytul = {s1, s2, s3, s4, s5};
        if (!s1.equals("") && !s2.equals("") && !s3.equals("")
                && !s4.equals("") && !s5.equals("")) {
            fasada.dodaj_tytul(tytul);
        }
    }
}
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String s1, s2, s3, s4, s5;
    Object zrodlo = evt.getSource();
    if (zrodlo == jButton2) {
        s1 = jTextField5.getText();
        s2 = jTextField7.getText();
        String[] ksiazka = {s1, s2};
        if (!s1.equals("") && !s2.equals("")) {
            fasada.dodaj_ksiazke(ksiazka);
        }
    }
}
```

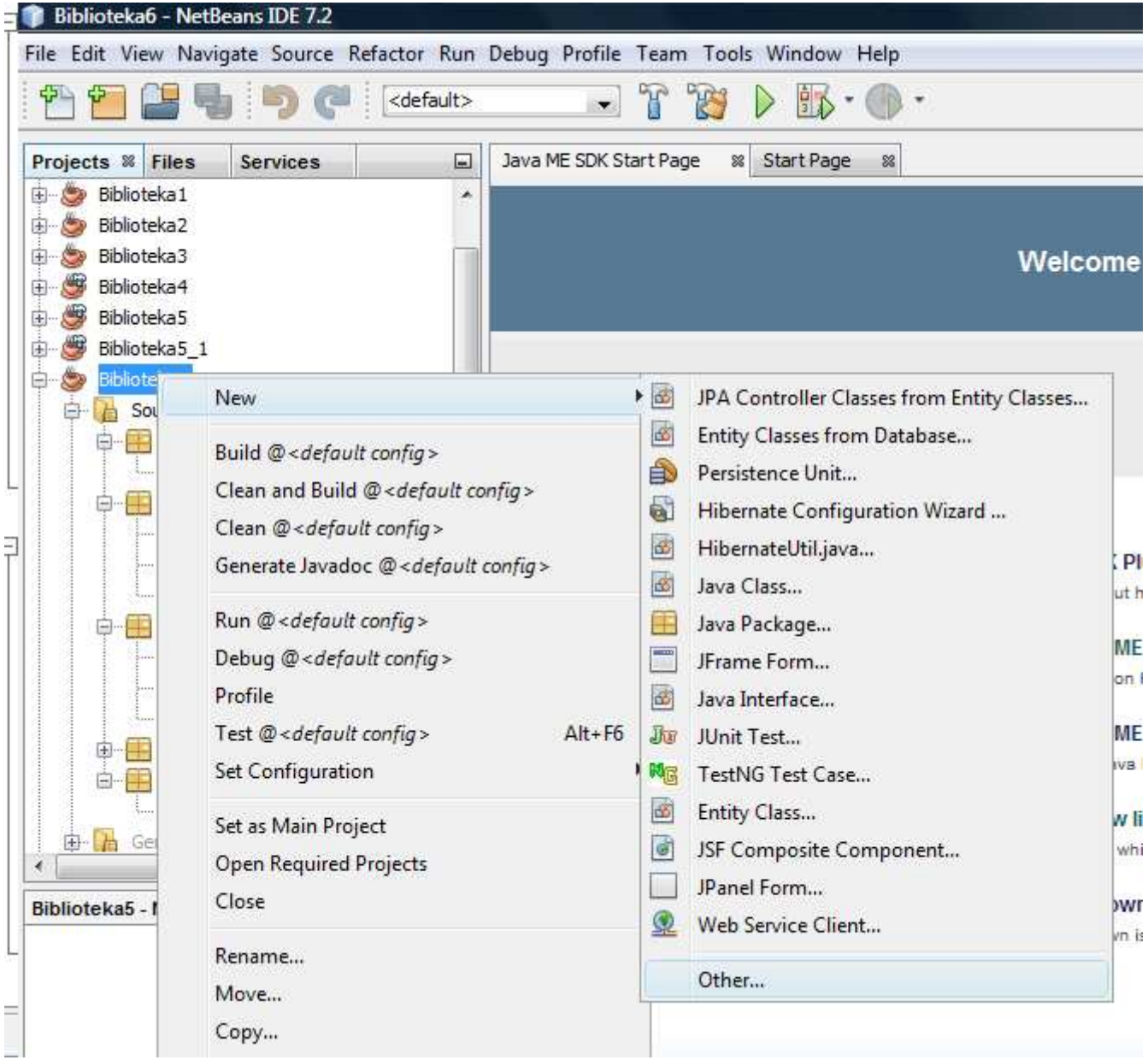
5.2. Część definicji z lab4 oraz obsługa nowych przycisków: **JButton5-jButton6**

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    zawartosc_listy(fasada.tytuly(), jComboBox1);
}
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    zawartosc_listy(fasada.ksiazki(), jComboBox2);
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        baza.dodaj_tytuly();
    } catch (Exception e) {
        System.out.println(e);
    }
}
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        baza.dodaj_ksiazki();
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

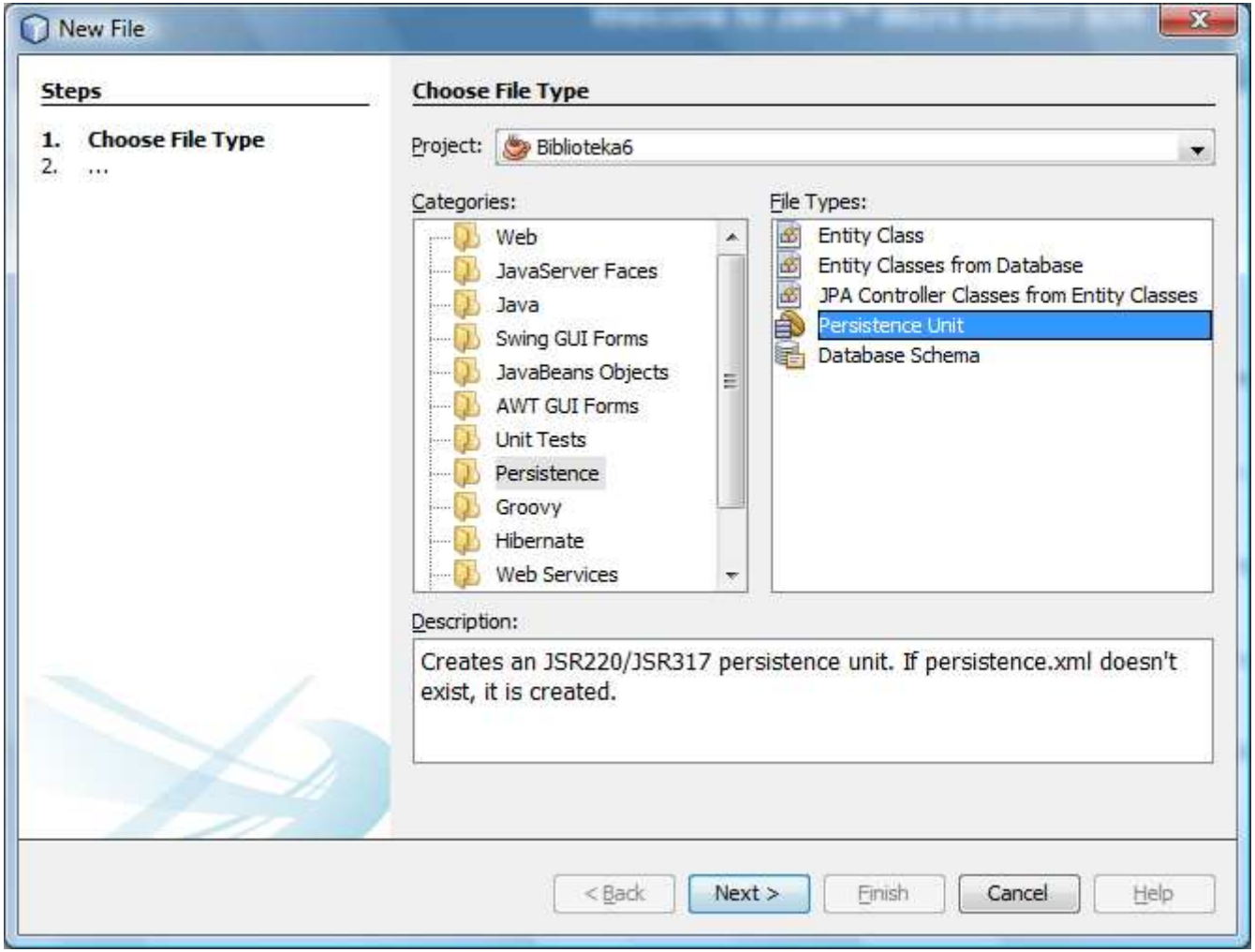
5.3. Obsługa nowych przycisków: JButton7-jButton9

```
private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        zawartosc_listy(baza.tytuly(), jComboBox1);
    } catch (Exception e) {
        System.out.println(e);
    }
}
private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        zawartosc_listy(baza.ksiazki(), jComboBox2);
    } catch (Exception e) {
        System.out.println(e);
    }
}
private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    jButton7ActionPerformed(evt);
    jButton8ActionPerformed(evt);
}
private void zawartosc_listy(ArrayList<String> kol, JComboBox lista) {
    lista.removeAllItems();
    for (String s : kol) {
        lista.addItem(s);
    }
}
```

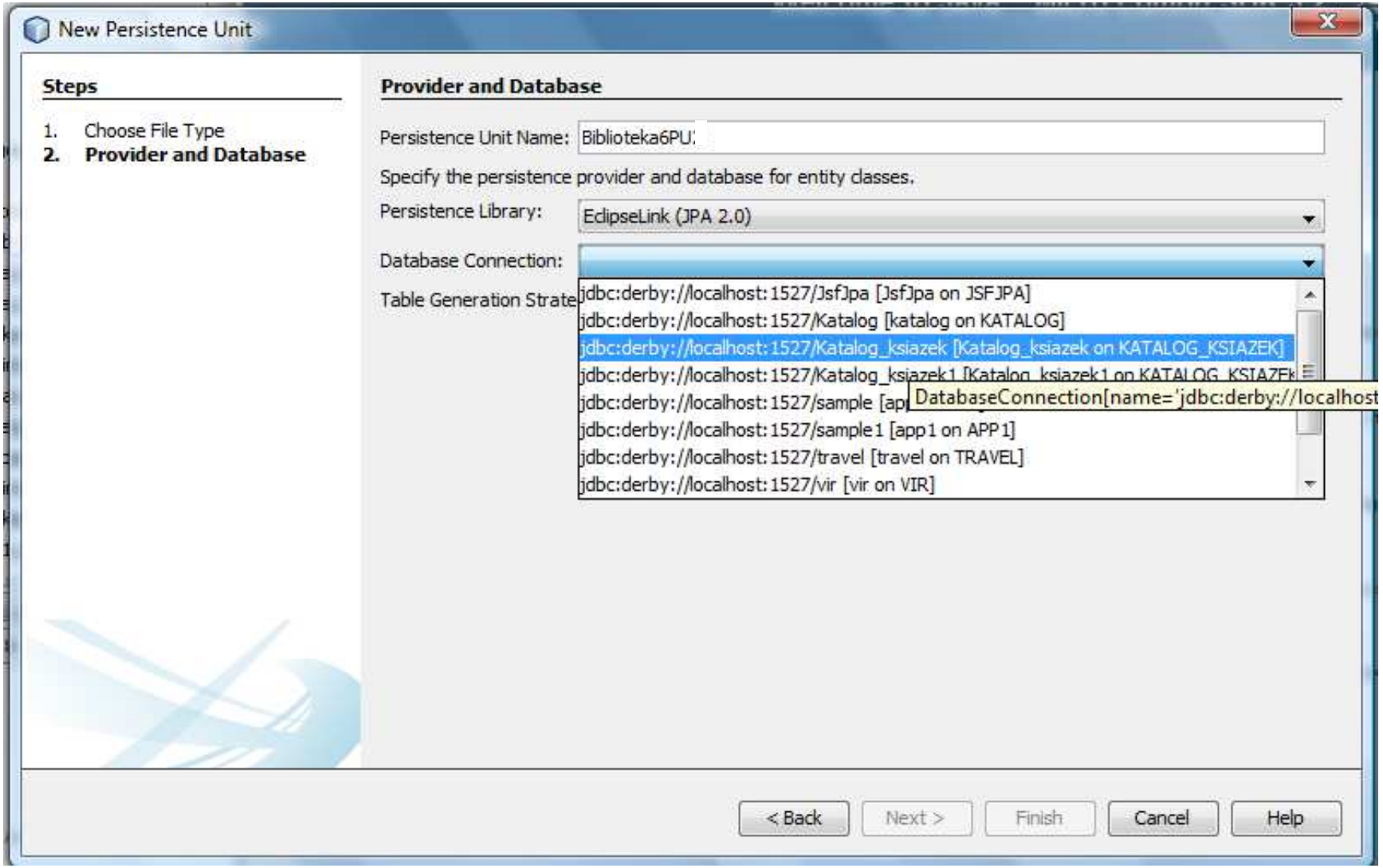
6. Dodanie pliku persistence.xml definiującego proces ORM (JPA)



6.1. Dodanie pliku persistence.xml definiującego proces ORM (JPA)



6.2. Dodanie pliku persistence.xml definiującego proces ORM (JPA)



6.3. Przekształcenie klasy Tytul_ksiazki na typ Entity - w celu utrwalania jej w bazie danych technologią ORM (JPA)

```
package Warstwa_biznesowa;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import javax.xml.bind.annotation.XmlTransient;
/**...*/
@Entity
public class Tytul_ksiazki implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Basic(optional = false)
    @Column(name = "ID_TYTUL")
    private Long idTytul;
    @Column(name = "TYTUL")
    private String tytul;
    @Column(name = "AUTOR_NAZWISKO")
    private String nazwisko;
    @Column(name = "AUTOR_IMIE")
    private String imie;
    @Column(name = "ISBN")
    private String ISBN;
    @Column(name = "WYDAWNICTWO")
    private String wydawnictwo;
    @OneToMany(mappedBy = "mTytul_ksiazki", cascade=CascadeType.ALL)
    private Collection<Ksiazka> mKsiazka;
```

```
public Tytul_ksiazki()                              {    }
public Tytul_ksiazki(int i)                         { mKsiazka=new ArrayList();
                                                      idTytul=null;    }
public Tytul_ksiazki(Long idTytul)                  { this.idTytul = idTytul;  }
public Long getIdTytul()                            { return idTytul; }
public void setIdTytul(Long idTytul)                { this.idTytul = idTytul; }
public String getTytul()                            { return tytul; }
public void setTytul(String tytul)                  { this.tytul = tytul; }
public String getNazwisko()                         { return nazwisko;  }
public void setNazwisko(String autorNazwisko)       { this.nazwisko = autorNazwisko;  }
public String getImie()                             {  return imie; }
public void setImie(String autorImie)               {  this.imie = autorImie; }
public String getISBN()                             { return ISBN;  }
public void setISBN(String isbn)                    { this.ISBN = isbn;  }
public String getWydawnictwo()                      {  return wydawnictwo;  }
public void setWydawnictwo(String wydawnictwo) {   this.wydawnictwo = wydawnictwo;  }
@XmlTransient
public Collection<Ksiazka> getMKsiazka()                              {  return mKsiazka; }
public void setMKsiazka(Collection<Ksiazka> ksiazkaCollection) {  this.mKsiazka = ksiazkaCollection;  }
@Override
public int hashCode() {
    int hash = 0;
    hash += (idTytul != null ? idTytul.hashCode() : 0);
    return hash;}
@Override
public String toString() {
    String pom = "Tytul: " + getTytul();
    pom += " Autor: " + getNazwisko() + " " + getImie();
    pom += " ISBN: " + getISBN();
    pom += " Wydawnictwo: " + getWydawnictwo();
    return pom;    }
@Override
public boolean equals(Object ob)  {//your code here
    String isbn2 = ((Tytul_ksiazki) ob).getISBN();
    return ISBN.equals(isbn2); }
```

6.4. Przekształcenie klasy Tytul_ksiazki na typ Entity - w celu utrwalania jej w bazie danych technologią ORM (JPA) - cd

6.5. Przekształcenie klasy Tytul_ksiazki na typ Entity cd – ta część definicji służy do realizacji usług warstwy biznesowej aplikacji

```
    public void dodaj_ksiazke(String dane[]) // your code here
    {
        Ksiazka nowa= new Ksiazka(1);
        if (nowa != null) {
            nowa.setNumer(Integer.parseInt(dane[1]));
            addKsiazka(nowa);
        }
    }

    public void addKsiazka(Ksiazka nowa) {
        if (!this.mKsiazka.contains(nowa)) {
            this.mKsiazka.add(nowa);
            nowa.setTytul_ksiazki(this);
        }
    }

    public ArrayList<String> ksiazki() {
        ArrayList<String> ksiazki = new ArrayList();
        Iterator<Ksiazka> it = mKsiazka.iterator();
        while (it.hasNext()) {
            ksiazki.add(it.next().toString());
        }
        return ksiazki;
    }
}
```

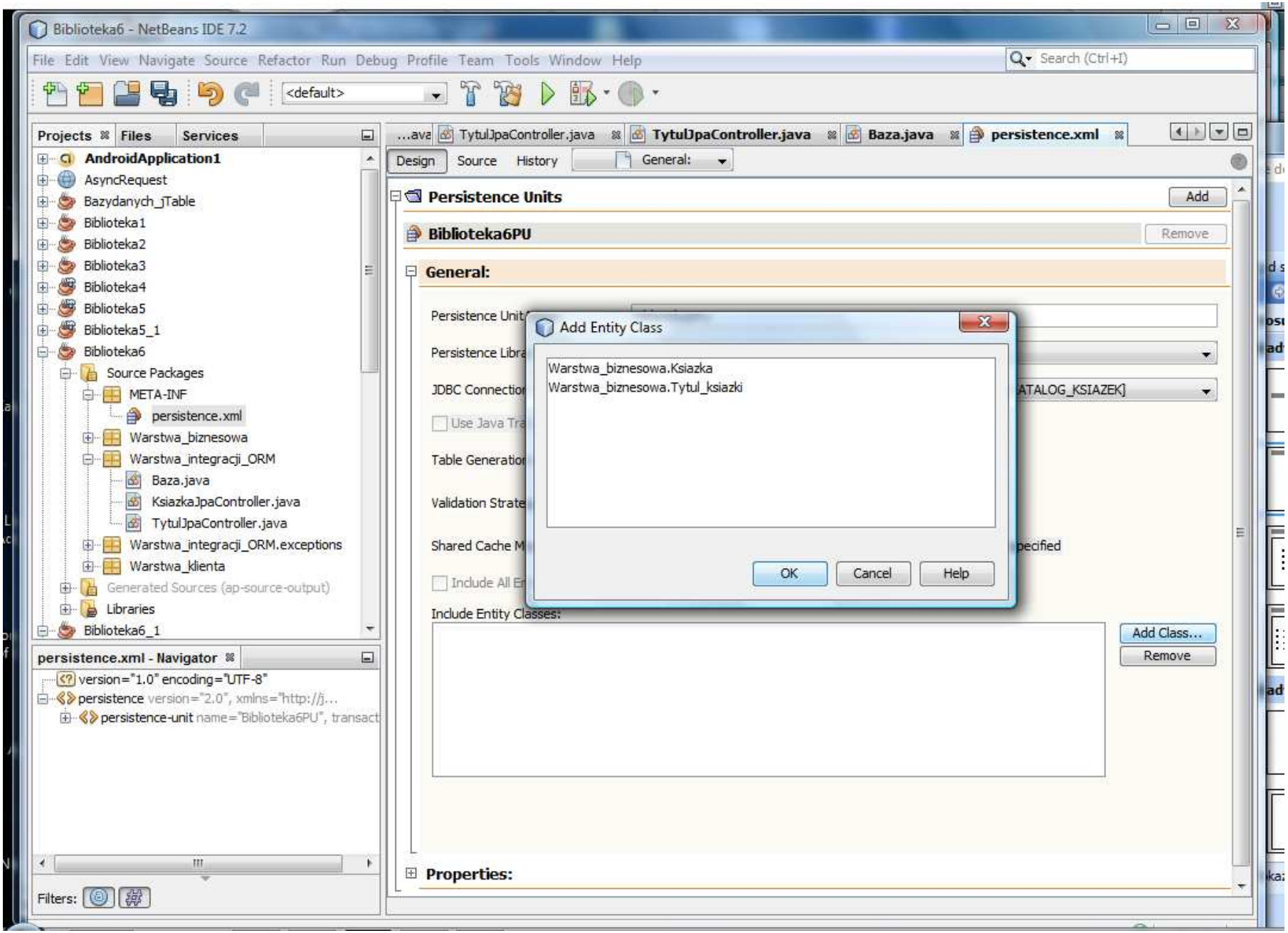
6.6. Przekształcenie klasy Ksiazka na typ Entity - w celu utrwalania jej w bazie danych technologią ORM (JPA)

```
package Warstwa_biznesowa;
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
/**...*/
@Entity
public class Ksiazka implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Basic(optional = false)
    @Column(name = "ID_KSIAZKA")
    private Long idKsiazka;
    @Column(name = "NUMER")
    private int numer;
    @JoinColumn(name = "ID_TYTUL_", referencedColumnName = "ID_TYTUL")
    @ManyToOne
    private Tytul_ksiazki mTytul_ksiazki;

    public Ksiazka()                                    {    }
    public Ksiazka(int i)                               { idKsiazka = null; }
    public Long getIdKsiazka()                          { return idKsiazka; }
    public void setIdKsiazka(Long idKsiazka)            { this.idKsiazka = idKsiazka; }
    public int getNumer()                               { return numer; }
    public void setNumer(int numer)                     { this.numer = numer; }
    public Tytul_ksiazki getTytul_ksiazki()             { return mTytul_ksiazki; }
    public void setTytul_ksiazki(Tytul_ksiazki idTytul) { this.mTytul_ksiazki = idTytul; }
```

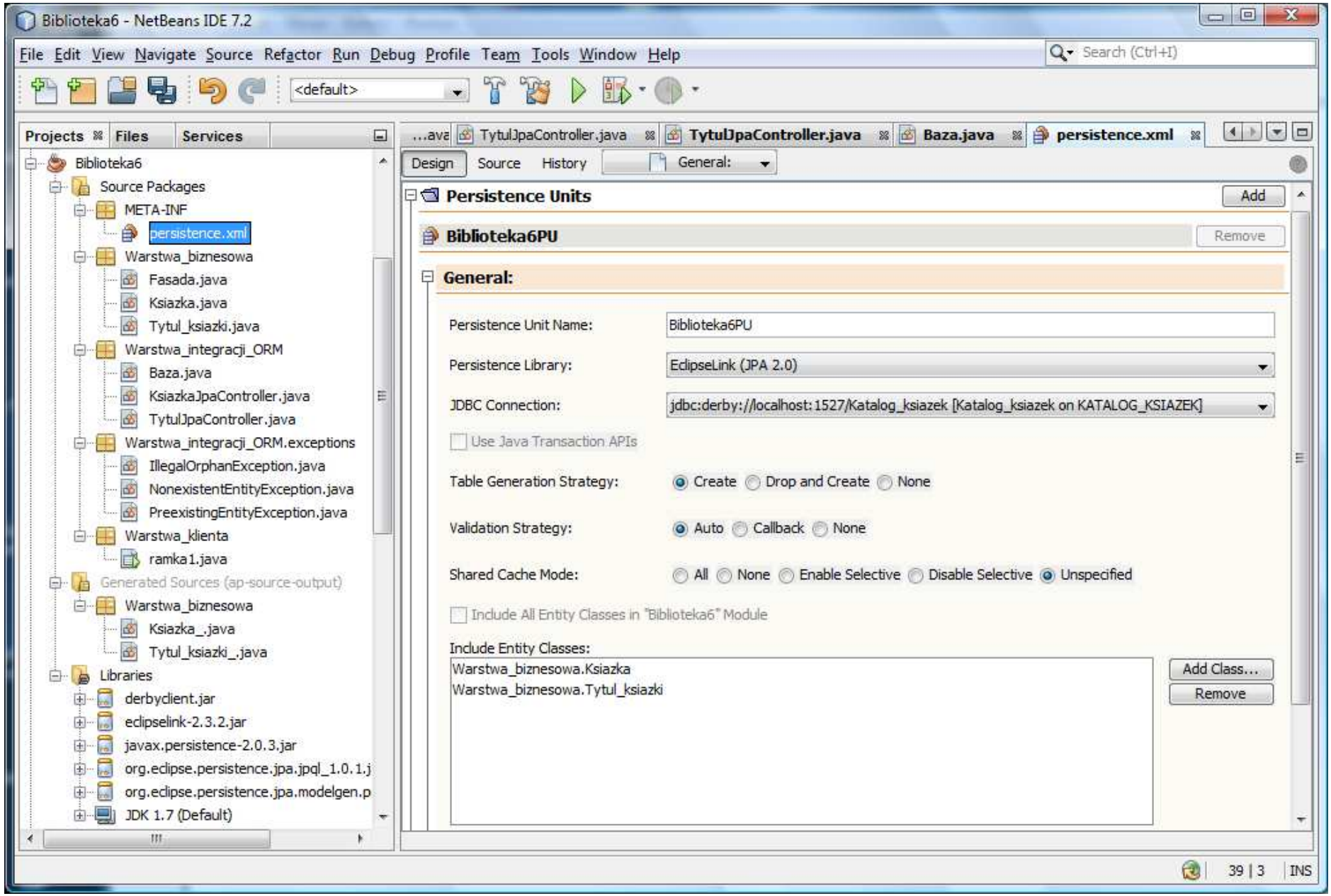
6.7. Przekształcenie klasy Ksiazka na typ Entity - w celu utrwalania jej w bazie danych technologią ORM (JPA) cd

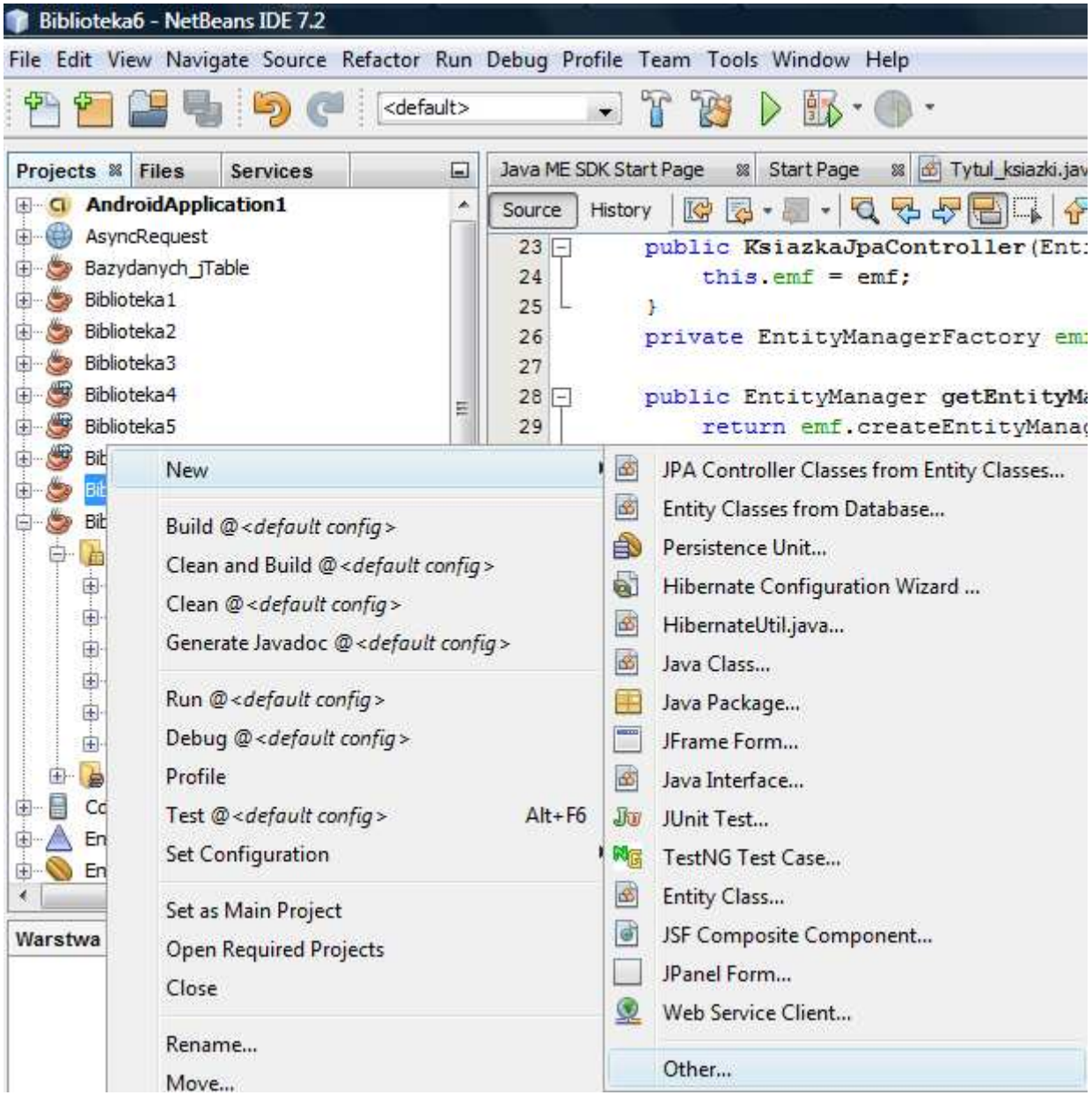
```
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idKsiazka != null ? idKsiazka.hashCode() : 0);
        return hash;
    }

    public boolean equals(Object ob) // your code here
    {
        return numer == ((Ksiazka) ob).getNumer();
    }

    public String toString() // your code here
    {
        String pom = mTytul_ksiazki.toString();
        pom += " Numer: " + getNumer();
        return pom;
    }
}
```

6.8. Dodanie do pliku persistence.xml definiującego proces ORM (JPA) klas typu Entity



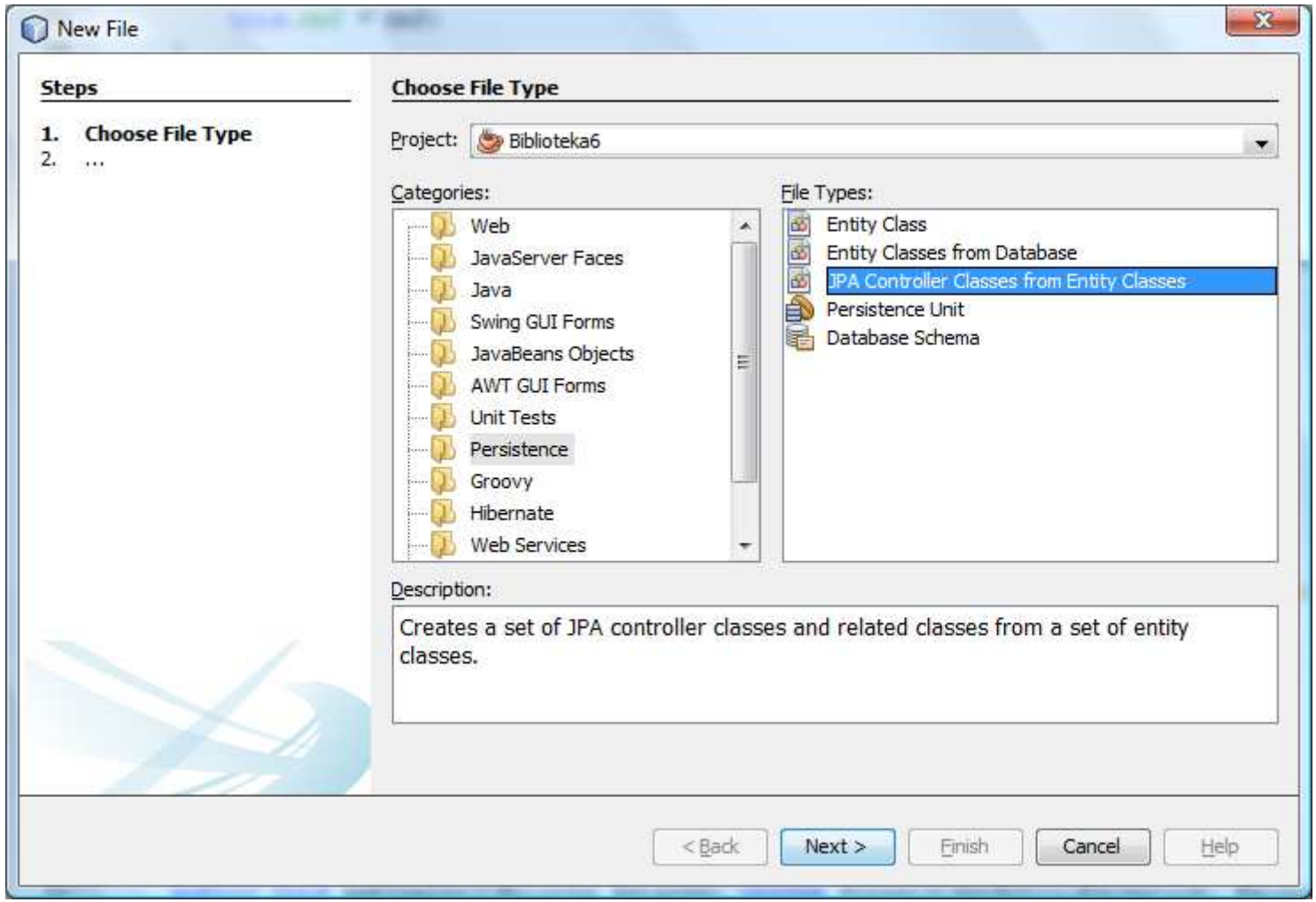
6.9. Dodanie do pliku persistence.xml definiującego proces ORM (JPA) klas typu Entity



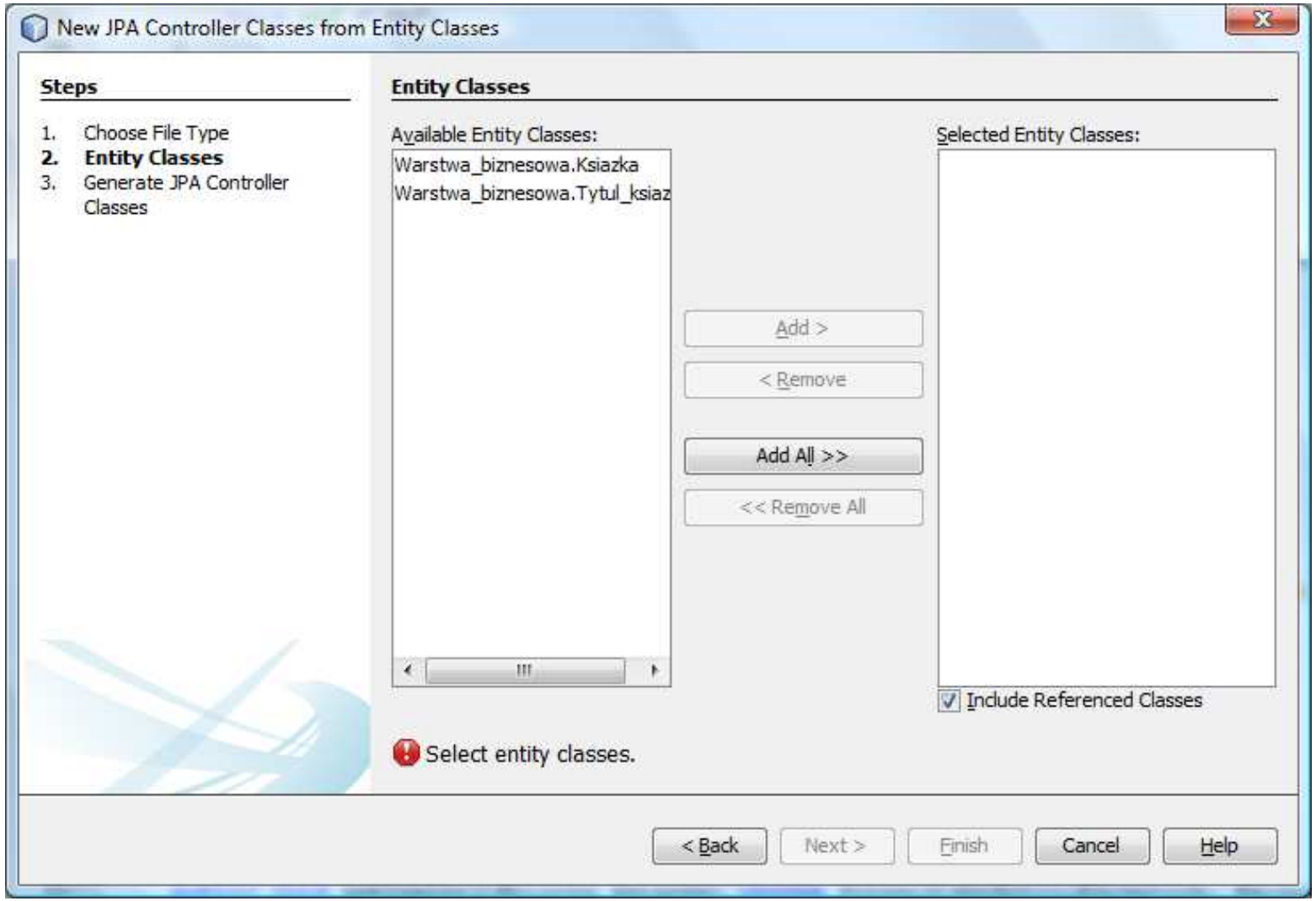


7. Dodanie kontrolerów do utrwalania klas typu Entity w warstwie integracji

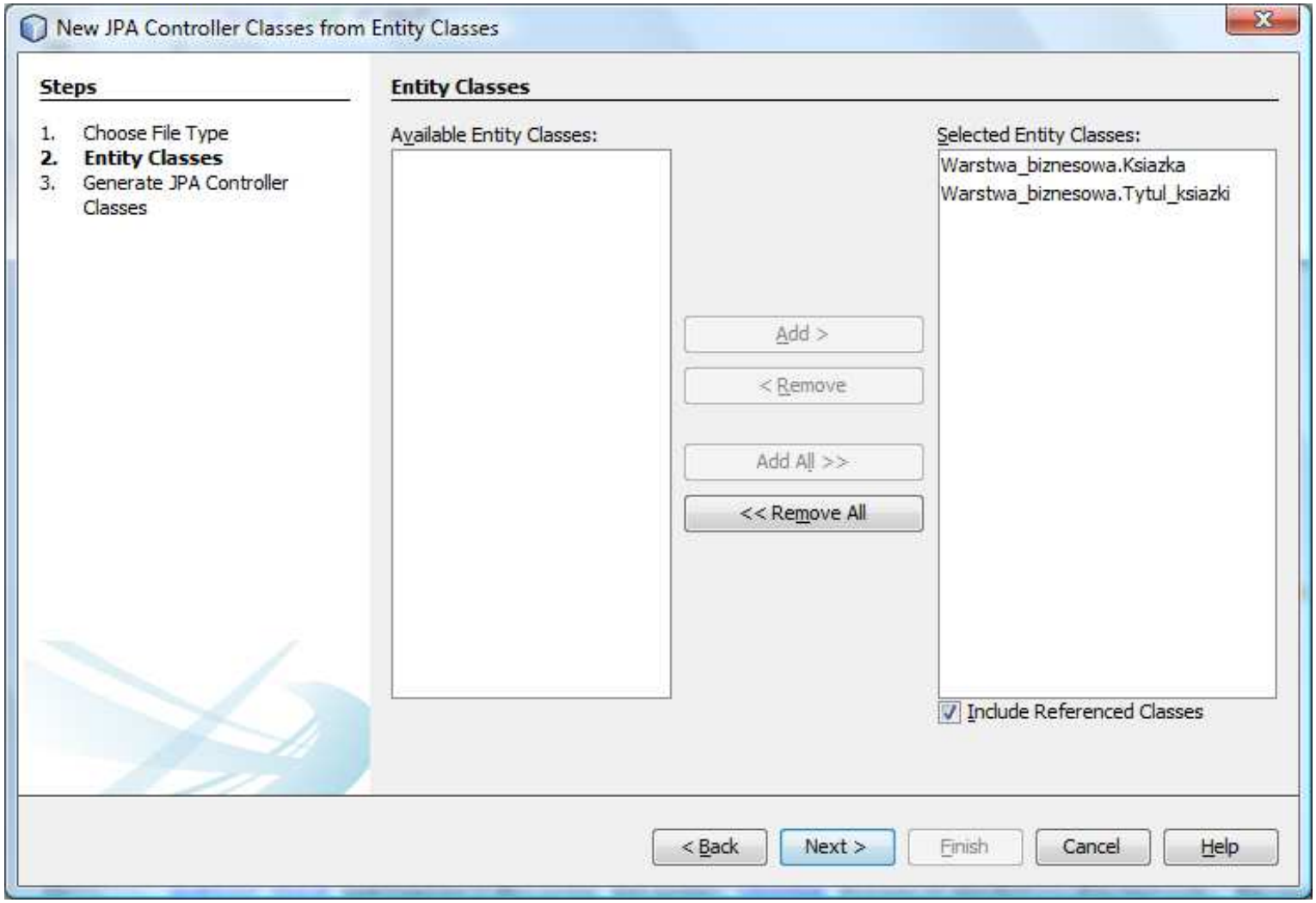
7.1. Dodanie kontrolerów w warstwie integracji do utrwalania klas typu Entity cd



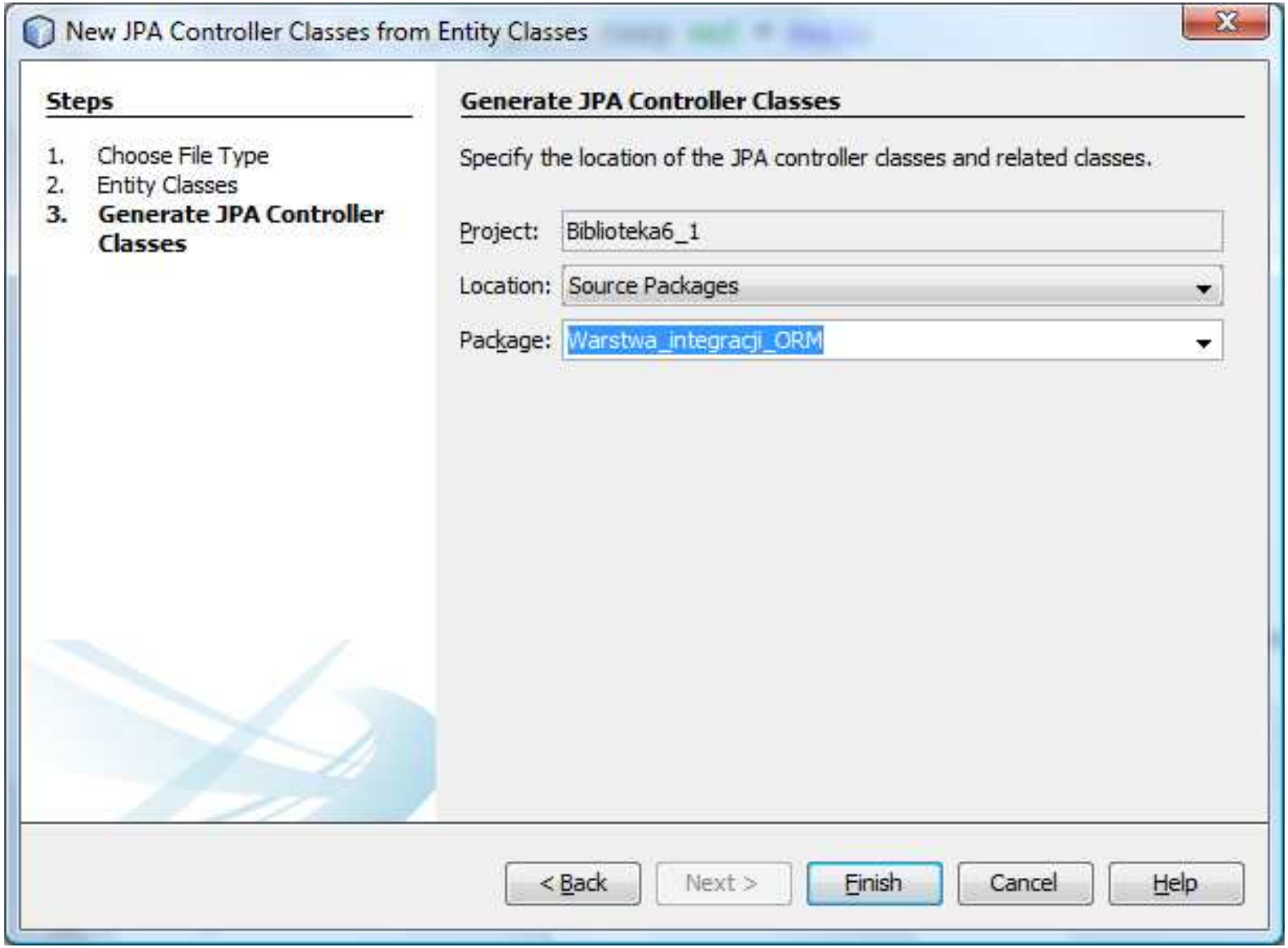
7.2. Dodanie kontrolerów w warstwie integracji do utrwalania klas typu Entity cd



7.3. Dodanie kontrolerów w warstwie integracji do utrwalania klas typu Entity cd



7.4. Dodanie kontrolerów w warstwie integracji do utrwalania klas typu Entity cd



7.5. Dodanie kontrolerów do utrwalania klas typu Entity – wygenerowany kod

```
import Warstwa_integracji_ORM.exceptions.NonexistentEntityException;
import Warstwa_biznesowa.Tytul_ksiazki;
import java.io.Serializable;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Query;
import javax.persistence.EntityNotFoundException;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Root;
/**...*/
public class Tytul_ksiazkiJpaController implements Serializable {

    public Tytul_ksiazkiJpaController(EntityManagerFactory emf) {...}
    private EntityManagerFactory emf = null;

    public EntityManager getEntityManager() {...}
    public void create(Tytul_ksiazki tytul_ksiazki) {...}
    public void edit(Tytul_ksiazki tytul_ksiazki) throws NonexistentEntityException, Exception {...}
    public void destroy(Long id) throws NonexistentEntityException {...}
    public List<Tytul_ksiazki> findTytul_ksiazkiEntities() {...}
    public List<Tytul_ksiazki> findTytul_ksiazkiEntities(int maxResults, int firstResult) {...}
    private List<Tytul_ksiazki> findTytul_ksiazkiEntities(boolean all, int maxResults, int firstResult)
        {...}
    public Tytul_ksiazki findTytul_ksiazki(Long id) {...}
    public int getTytul_ksiazkiCount() {...}
}
```

7.6. Dodanie kontrolerów do utrwalania klas typu Entity – wygenerowany kod

```
import Warstwa_integracji_ORM.exceptions.NonexistentEntityException;
import Warstwa_biznesowa.Ksiazka;
import java.io.Serializable;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Query;
import javax.persistence.EntityNotFoundException;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Root;

/**...*/
public class KsiazkaJpaController implements Serializable {

    public KsiazkaJpaController(EntityManagerFactory emf) {...}
    private EntityManagerFactory emf = null;

    public EntityManager getEntityManager() {...}
    public void create(Ksiazka ksiazka) {...}
    public void edit(Ksiazka ksiazka) throws NonexistentEntityException, Exception {...}
    public void destroy(Long id) throws NonexistentEntityException {...}
    public List<Ksiazka> findKsiazkaEntities() {...}
    public List<Ksiazka> findKsiazkaEntities(int maxResults, int firstResult) {...}
    private List<Ksiazka> findKsiazkaEntities(boolean all, int maxResults, int firstResult)
    {...}
    public Ksiazka findKsiazka(Long id) {...}
    public int getKsiazkaCount() {...}
}
```

7.7. Dodanie kontrolerów do utrwalania klas typu Entity i wygenerowany pakiet z klasami wyjątków do obsługi utrwalania metodą JPA

```
import java.util.ArrayList;
import java.util.List;

public class IllegalOrphanException extends Exception {
    private List<String> messages;
    public IllegalOrphanException(List<String> messages) {...}
    public List<String> getMessages() {...}
}

public class NonexistentEntityException extends Exception {
    public NonexistentEntityException(String message, Throwable cause) {...}
    public NonexistentEntityException(String message) {...}
}


public class PreexistingEntityException extends Exception {
    public PreexistingEntityException(String message, Throwable cause) {...}
    public PreexistingEntityException(String message) {...}
}
```

7.8. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **create** na **dodaj_tytul** oraz jej kod po zmodyfikowaniu w klasie **TytulJpaVController**

```
public void dodaj_tytul(Tytul_ksiazki tytul) throws PreexistingEntityException,
    Exception {
      EntityManager em = null;
      try {
          em = getEntityManager();
          em.getTransaction().begin();
          if (tytul.getIdTytul() == null) {
              em.persist(tytul);
              em.getTransaction().commit();
          }
      } catch (Exception ex) {
          if (findTytul(tytul.getIdTytul()) != null) {
              throw new PreexistingEntityException("Tytul " + tytul + " already exists.", ex);
          }
          throw ex;
      } finally {
          if (em != null) {
              em.close();
          }
      }
  }
```

7.9. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **edit** na **uaktualnij** oraz jej kod po zmodyfikowaniu w klasie **TytulJpaVController**

```
public void uaktualnij(Tytul_ksiazki tytul) throws NonexistentEntityException, Exception {
    EntityManager em = null;
    try {
        em = getEntityManager();
        em.getTransaction().begin();
        Tytul_ksiazki persistentTytul = em.find(Tytul_ksiazki.class, tytul.getIdTytul());
        persistentTytul.setTytul(tytul.getTytul());
        persistentTytul.setNazwisko(tytul.getNazwisko());
        persistentTytul.setImie(tytul.getImie());
        persistentTytul.setISBN(tytul.getISBN());
        persistentTytul.setWydawnictwo(tytul.getWydawnictwo());
        em.getTransaction().commit();
    } catch (Exception ex) {
        String msg = ex.getLocalizedMessage();
        if (msg == null || msg.length() == 0) {
            Long id = tytul.getIdTytul();
            if (findTytul(id) == null) {
                throw new NonexistentEntityException("The tytul with id " + id +
                                                     " no longer exists.");
            }
        }
        throw ex;
    } finally {
        if (em != null) {  em.close();  }
    }  }
```

7.10. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **destroy** na **usun** oraz jej kod po zmodyfikowaniu w klasie **TytulJpaVController**

```
public void usun(Long id) throws NonexistentEntityException {
    EntityManager em = null;
    try {
        em = getEntityManager();
        em.getTransaction().begin();
        Tytul_ksiazki tytul;
        try {
            tytul = em.getReference(Tytul_ksiazki.class, id);
        } catch (EntityNotFoundException enfe) {
            throw new NonexistentEntityException("The tytul with id " + id +
                                                 " no longer exists.", enfe);
        }
        em.remove(tytul);
        em.getTransaction().commit();
    } finally {
        if (em != null) {
            em.close();
        }
    }
}
```

7.11. Dodanie kontrolerów do utrwalania klas typu Entity – dodanie metody **tytuly()** w klasie **TytulJpaVController** zwracajacej dane odczytane z bazy danych metodą **getTytul_ksiazkis** i przekształcającej na dane łańcuchowe np. do prezentacji

```
public List<Tytul_ksiazki> findTytulEntities() {  return findTytulEntities(true, -1, -1);  }

   private List<Tytul_ksiazki> findTytulEntities(boolean all, int maxResults, int firstResult) {
      EntityManager em = getEntityManager();
      try {
         Query q = em.createQuery("select object(o) from Tytul_ksiazki as o");
         if (!all) {
            q.setMaxResults(maxResults);
            q.setFirstResult(firstResult);
         }
         return q.getResultList();
      } finally {
         em.close();  }  }

public Tytul_ksiazki[] getTytul_ksiazkis() {
                        return (Tytul_ksiazki[]) findTytulEntities().toArray(newTytul_ksiazki[0]); }

 public ArrayList<String> tytuly() {
      ArrayList<String> tytuly = new ArrayList();
      Tytul_ksiazki[] tytuly_ = getTytul_ksiazkis();
      for (Tytul_ksiazki t : tytuly_) {
         tytuly.add(t.toString());
      }
      return tytuly;   }
```

7.12. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **create** na **dodaj_ksiazke** oraz jej kod po zmodyfikowaniu w klasie **KsiazkaJpaVController**

```
public void dodaj_ksiazke(Ksiazka ksiazka) throws PreexistingEntityException,
    Exception {
      EntityManager em = null;
      try {
         em = getEntityManager();
         em.getTransaction().begin();
         if (ksiazka.getIdKsiazka() == null) {
            em.persist(ksiazka);
            em.getTransaction().commit();
         }
      } catch (Exception ex) {
         if (findKsiazka(ksiazka.getIdKsiazka()) != null) {
            throw new PreexistingEntityException("Ksiazka " + ksiazka +
                                                 " already exists.", ex);
         }
         throw ex;
      } finally {
         if (em != null) {
            em.close();
         }
      }
   }
```

7.13. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **edit** na **uaktualnij** oraz jej kod po zmodyfikowaniu w klasie **KsiazkaJpaVController**

```
public void uaktualnij(Ksiazka ksiazka) throws NonexistentEntityException, Exception {
    EntityManager em = null;
    try {
        em = getEntityManager();
        em.getTransaction().begin();
        Ksiazka persistentKsiazka = em.find(Ksiazka.class, ksiazka.getIdKsiazka());
        persistentKsiazka.setNumer(ksiazka.getNumer());
        em.getTransaction().commit();
    } catch (Exception ex) {
        String msg = ex.getLocalizedMessage();
        if (msg == null || msg.length() == 0) {
            Long id = ksiazka.getIdKsiazka();
            if (findKsiazka(id) == null) {
                throw new NonexistentEntityException("The ksiazka with id " + id +
                                                     " no longer exists.");
            }
        }
        throw ex;
    } finally {
        if (em != null) {
            em.close();
        }
    }
}
```

7.14. Dodanie kontrolerów do utrwalania klas typu Entity – zmiana nazwy metody **destroy** na **usun** oraz jej kod po zmodyfikowaniu w klasie **KsiazkaJpaVController**

```
public void usun(Long id) throws NonexistentEntityException {
        EntityManager em = null;
        try {
            em = getEntityManager();
            em.getTransaction().begin();
            Ksiazka ksiazka;
            try {
                ksiazka = em.getReference(Ksiazka.class, id);
            } catch (EntityNotFoundException enfe) {
                throw new NonexistentEntityException("The ksiazka with id " + id +
                                                     " no longer exists.", enfe);
            }
            em.remove(ksiazka);
            em.getTransaction().commit();
        } finally {
            if (em != null) {
                em.close();
            }
        }
    }
```

7.15. Dodanie kontrolerów do utrwalania klas typu Entity – dodanie metody **ksiazki()** w klasie **KsiazkaJpaVController** zwracajacej dane odczytane z bazy danych metodą **getKsiazkis** i przekształcającej na dane łańcuchowe np. do prezentacji

```
public List<Ksiazka> findKsiazkaEntities() {   return findKsiazkaEntities(true, -1, -1);   }

private List<Ksiazka> findKsiazkaEntities(boolean all, int maxResults, int firstResult) {
        EntityManager em = getEntityManager();
        try {
            Query q = em.createQuery("select object(o) from Ksiazka as o");
            if (!all) {
                q.setMaxResults(maxResults);
                q.setFirstResult(firstResult);
            }
            return q.getResultList();
        } finally {
            em.close();
        }    }

public Ksiazka[] getKsiazkis() {  return (Ksiazka[]) findKsiazkaEntities().toArray(new Ksiazka[0]);  }

public ArrayList<String> ksiazki() {
        ArrayList<String> ksiazki = new ArrayList();
        Ksiazka[] ksiazki_ = getKsiazkis();
        for (Ksiazka k : ksiazki_) {
            ksiazki.add(k.toString());
        }
        return ksiazki;    }
```

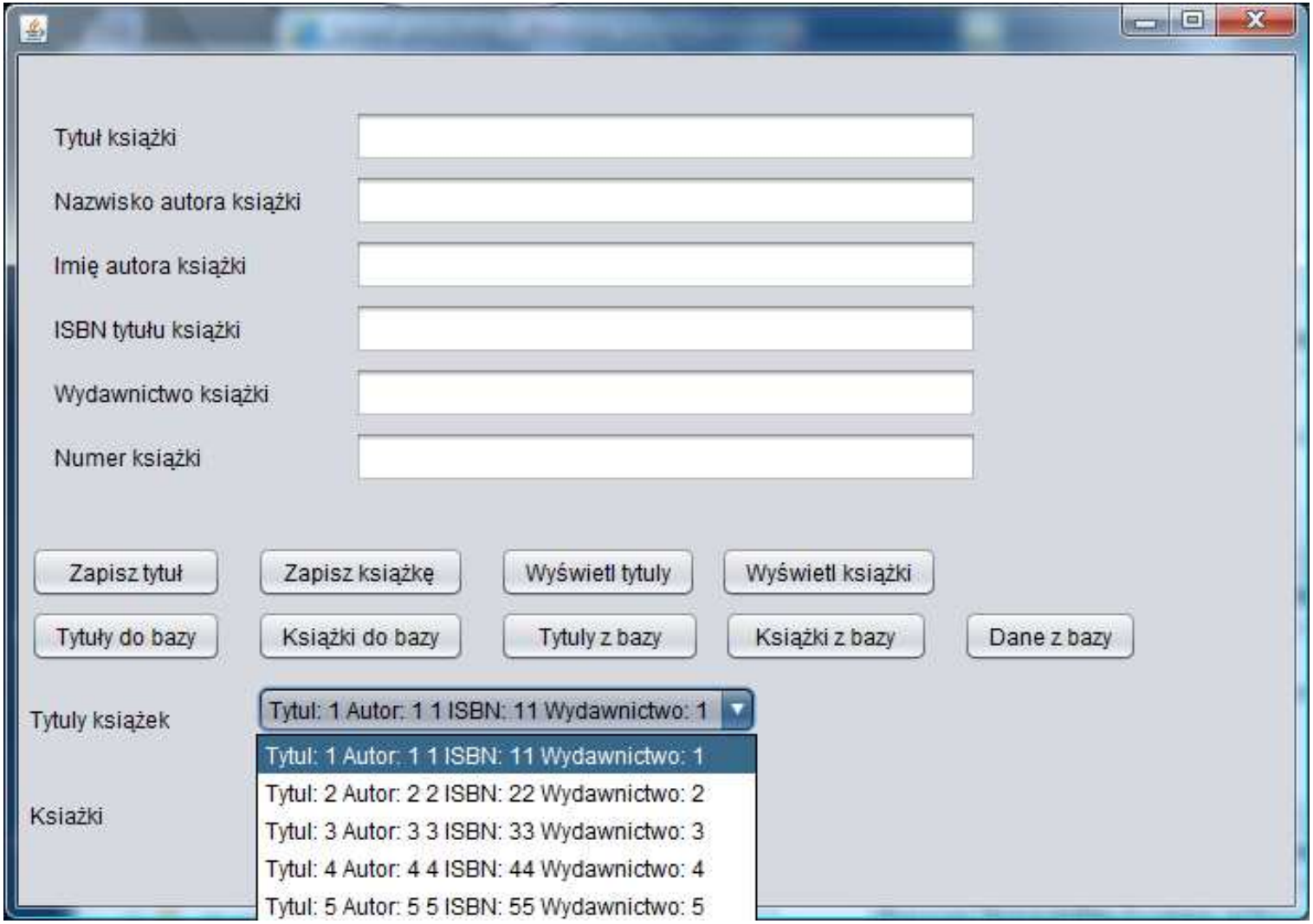
8. Dodana klasa Baza w warstwie integracji – fasada warstwy integracji

```
package Warstwa_integracji_ORM;
import Warstwa_biznesowa.Fasada;
import Warstwa_biznesowa.Ksiazka;
import Warstwa_biznesowa.Tytul_ksiazki;
import java.util.ArrayList;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
/**...*/
public class Baza {

    private TytulJpaController tytulJpaController;
    private KsiazkaJpaController ksiazkaJpaController;
    private EntityManagerFactory emf = null;
    private Fasada fasada;
    private Tytul_ksiazki tytuly[];
    private Ksiazka ksiazki[];

    public Baza(Fasada fasada_) {
        fasada = fasada_;
        createEntityManagerFactory();
        tytulJpaController = new TytulJpaController(emf);
        ksiazkaJpaController = new KsiazkaJpaController(emf);
    }
    private void createEntityManagerFactory() {
        if (emf == null) {
            emf = Persistence.createEntityManagerFactory("Biblioteka6PU");
        }
    }
    public void uaktualnij_tytuly() throws Exception{
        tytuly = tytulJpaController.getTytul_ksiazkis();
    }
    public void uaktualnij_ksiazki() throws Exception{
        ksiazki = ksiazkaJpaController.getKsiazkis();
    }
```

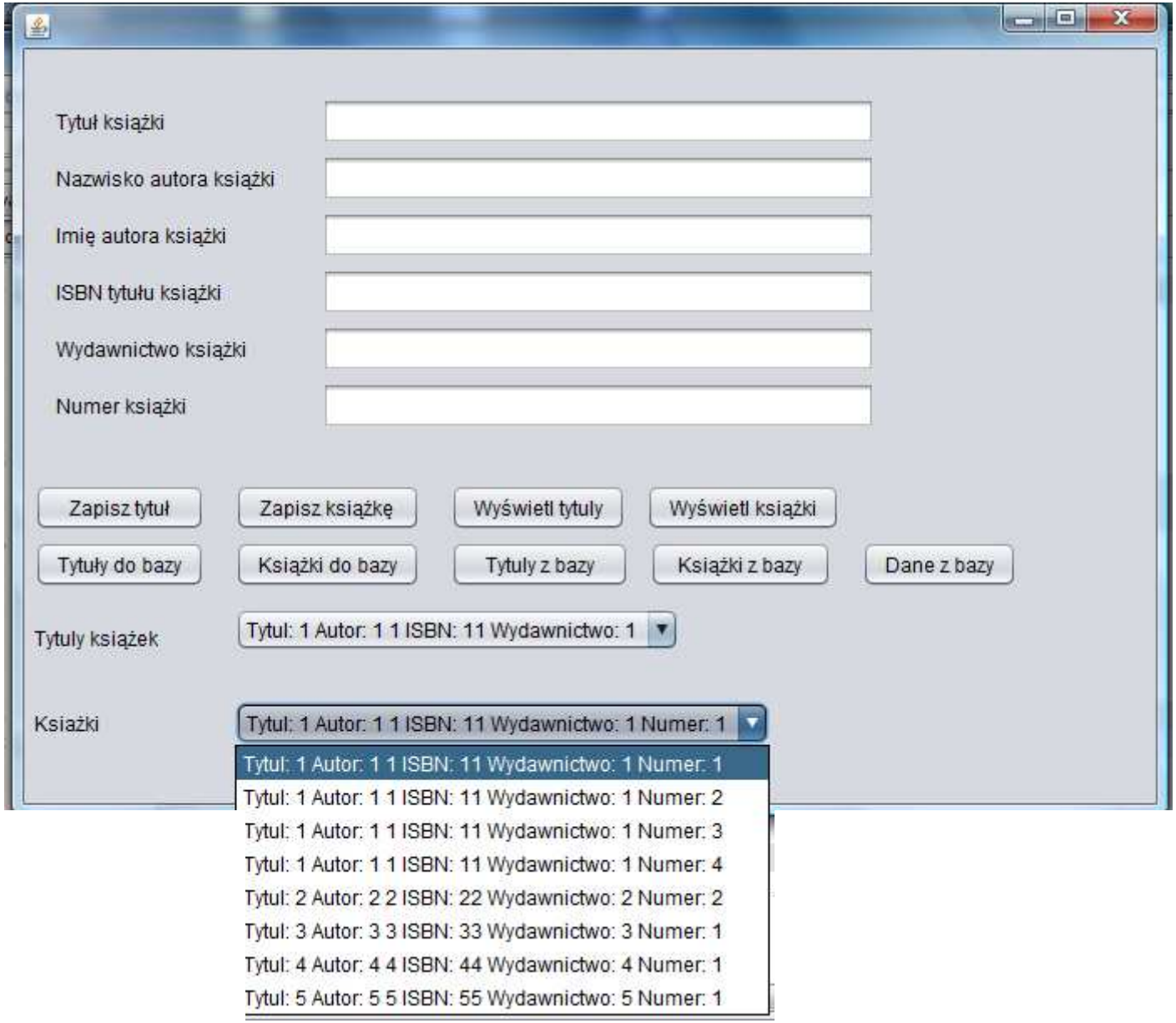
```
    public void uaktualnij_dane() throws Exception{
        uaktualnij_tytuly();
        uaktualnij_ksiazki();
        fasada.uaktualnij_dane(tytuly, ksiazki);
    }

    public void dodaj_tytuly() throws Exception{
        try {
            for (Tytul_ksiazki t : fasada.getTytuly_ksiazek()) {
                tytulJpaController.dodaj_tytul(t);
            }
        } catch (Exception e) {
        }
    }

    public void dodaj_ksiazki() throws Exception{
        try {
            for (Tytul_ksiazki t : fasada.getTytuly_ksiazek()) {
                for (Ksiazka k : t.getMKsiazka()) {
                    ksiazkaJpaController.dodaj_ksiazke(k);
                }
            }
        } catch (Exception e) {
        }
    }

    public ArrayList<String> ksiazki() throws Exception {
        return ksiazkaJpaController.ksiazki();
    }

    public ArrayList<String> tytuly() throws Exception {
        return tytulJpaController.tytuly();
    }
}
```

8.1 Dodana klasa Baza w warstwie integracji – fasada warstwy integracji cd

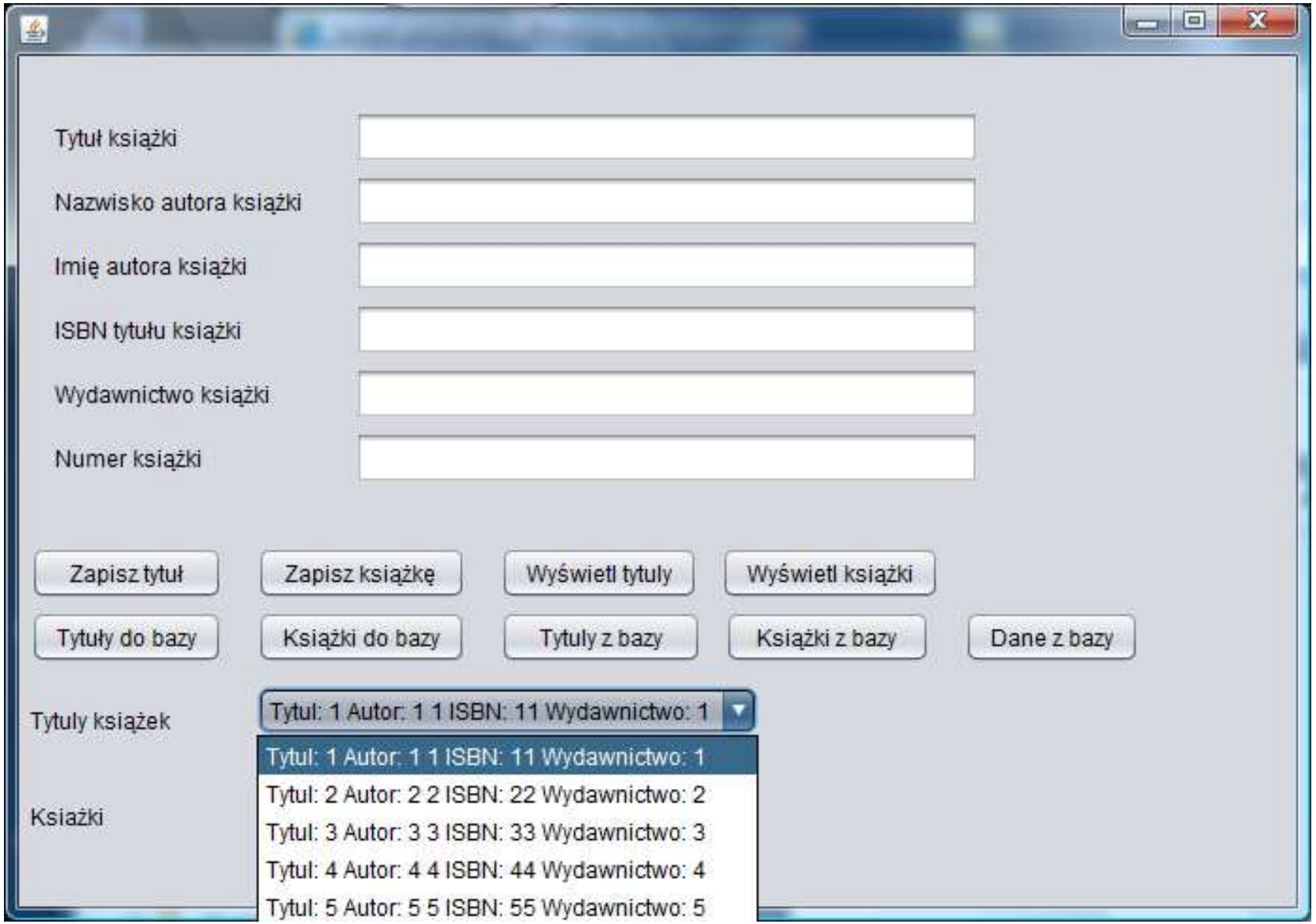
9. Wyświetlenie tytułów z aplikacji



9.1. Wyświetlenie książek z aplikacji



9.2. Wyświetlenie tytułów z bazy danych



9.3. Wyświetlenie książek z bazy danych