

**Budowa aplikacji
wielowarstwowych cd
Dostęp do bazy danych w
oparciu o technologię ORM
(Object Relational Mapping)**

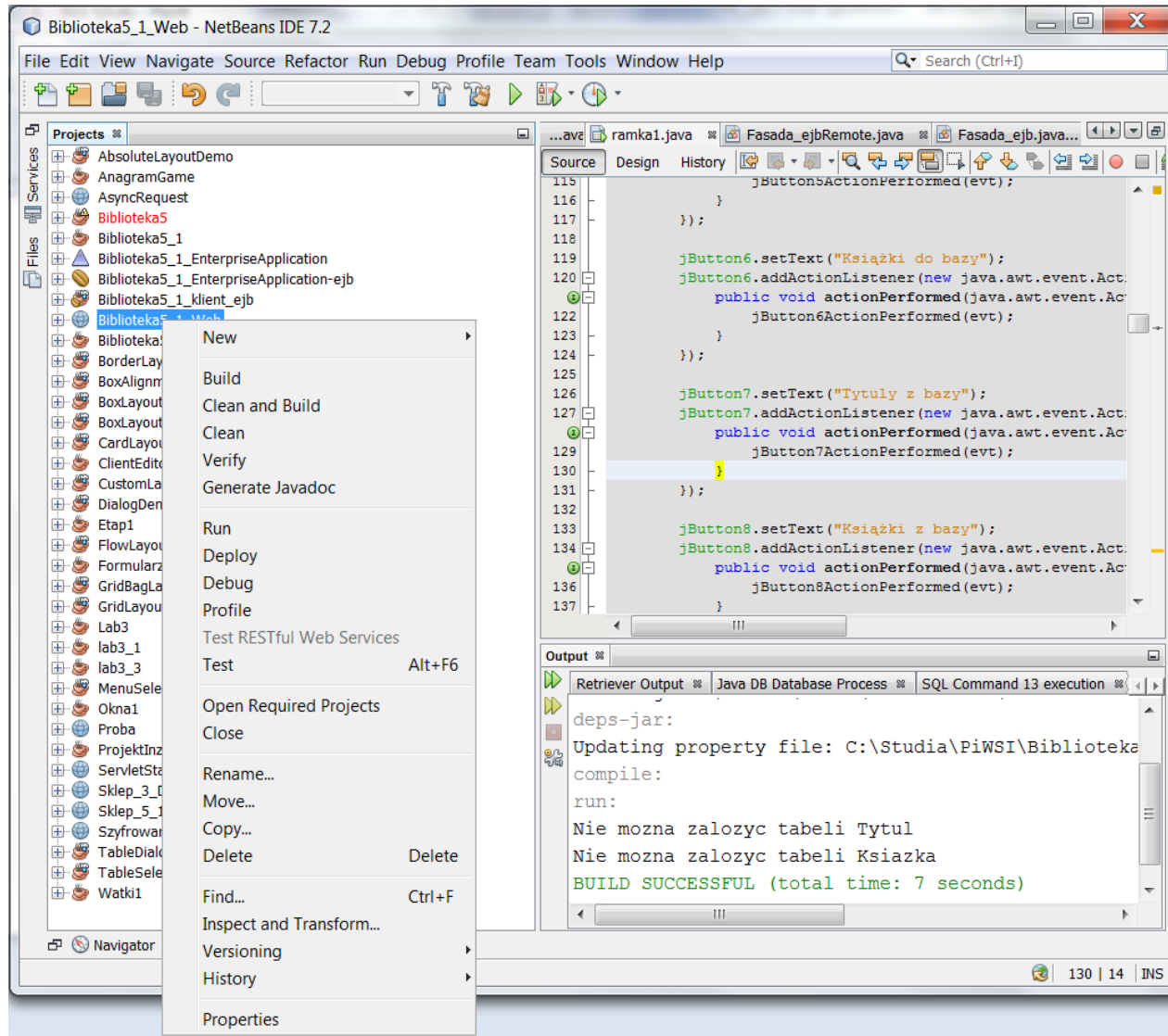
Laboratorium 5

Programowanie komponentowe

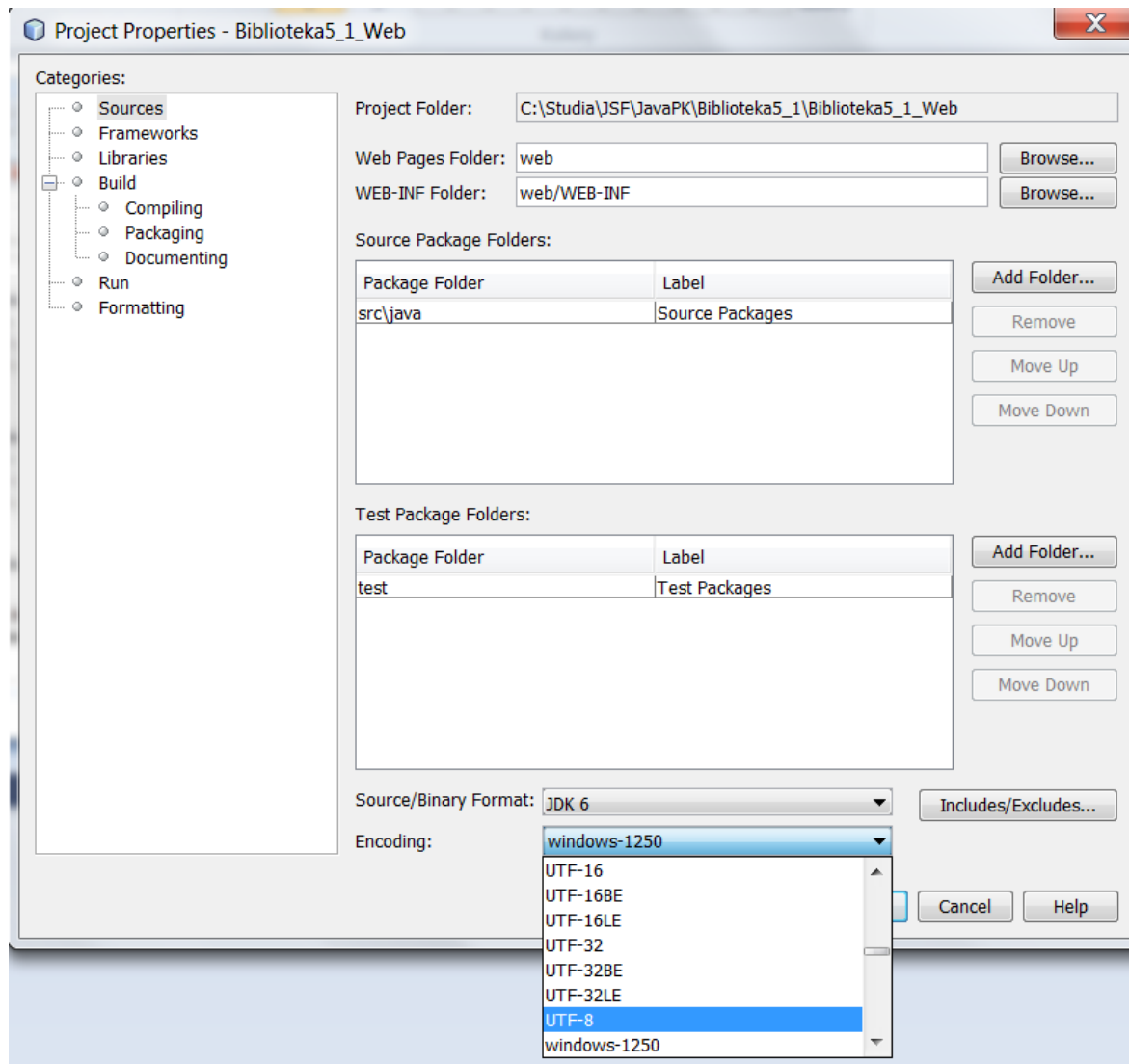
Zofia Kruczkiewicz

Konieczne ustawienie kodowania UTF-8 w projektach tworzonych w ramach laboratorium z Programowania komponentowego w Javie

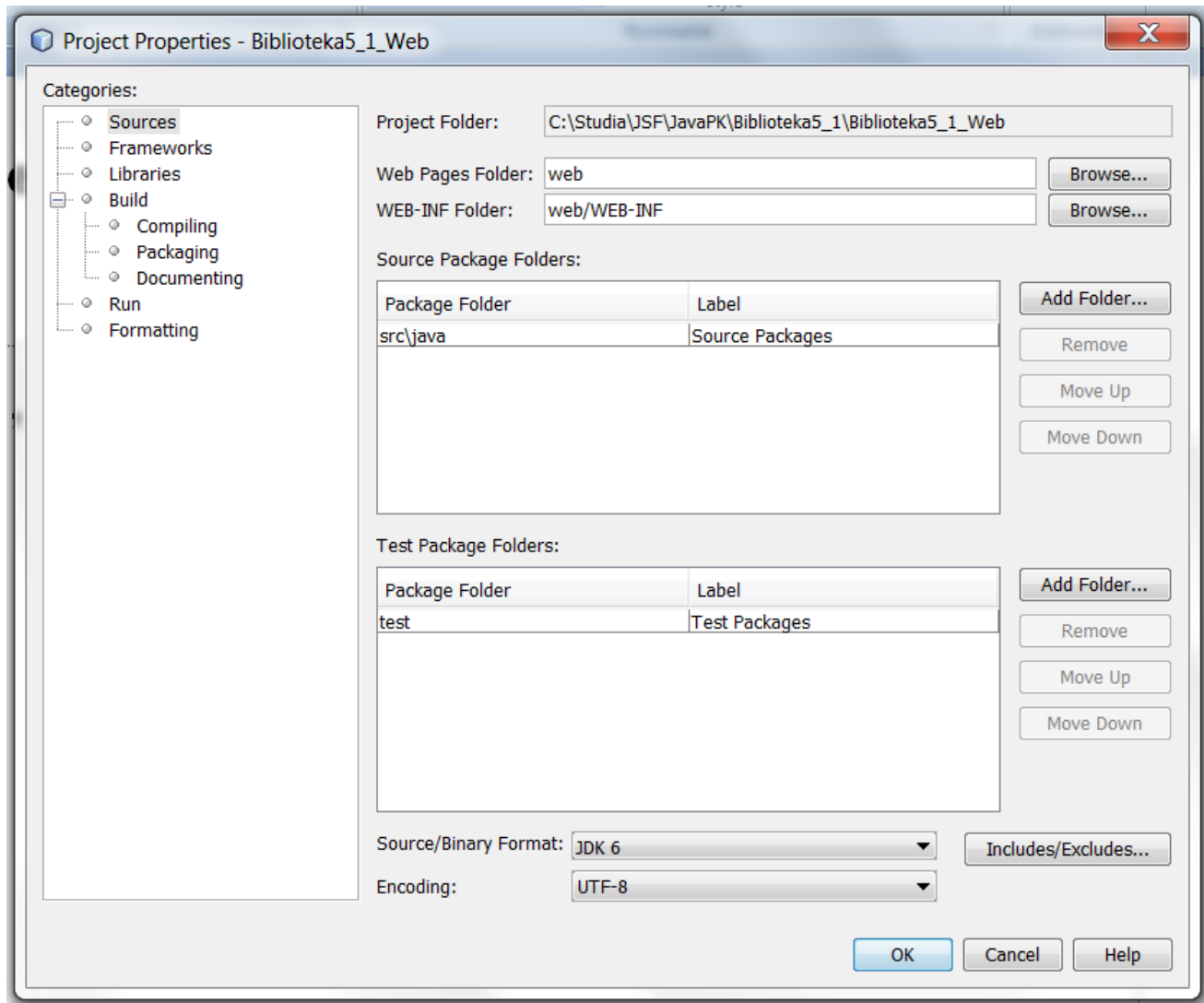
(1) Ustawienie kodowania typu UTF-8 w projektach Java SE oraz Java EE – w zakładce Projects należy prawym klawiszem kliknąć na nazwę projektu i wybrać z listy pozycję Properties



(2) Ustawienie kodowania typu UTF-8 w projektach Java SE oraz Java EE – w formularzu typu Project Properties należy rozwinąć listę Encoding i wybrać pozycję UTF-8

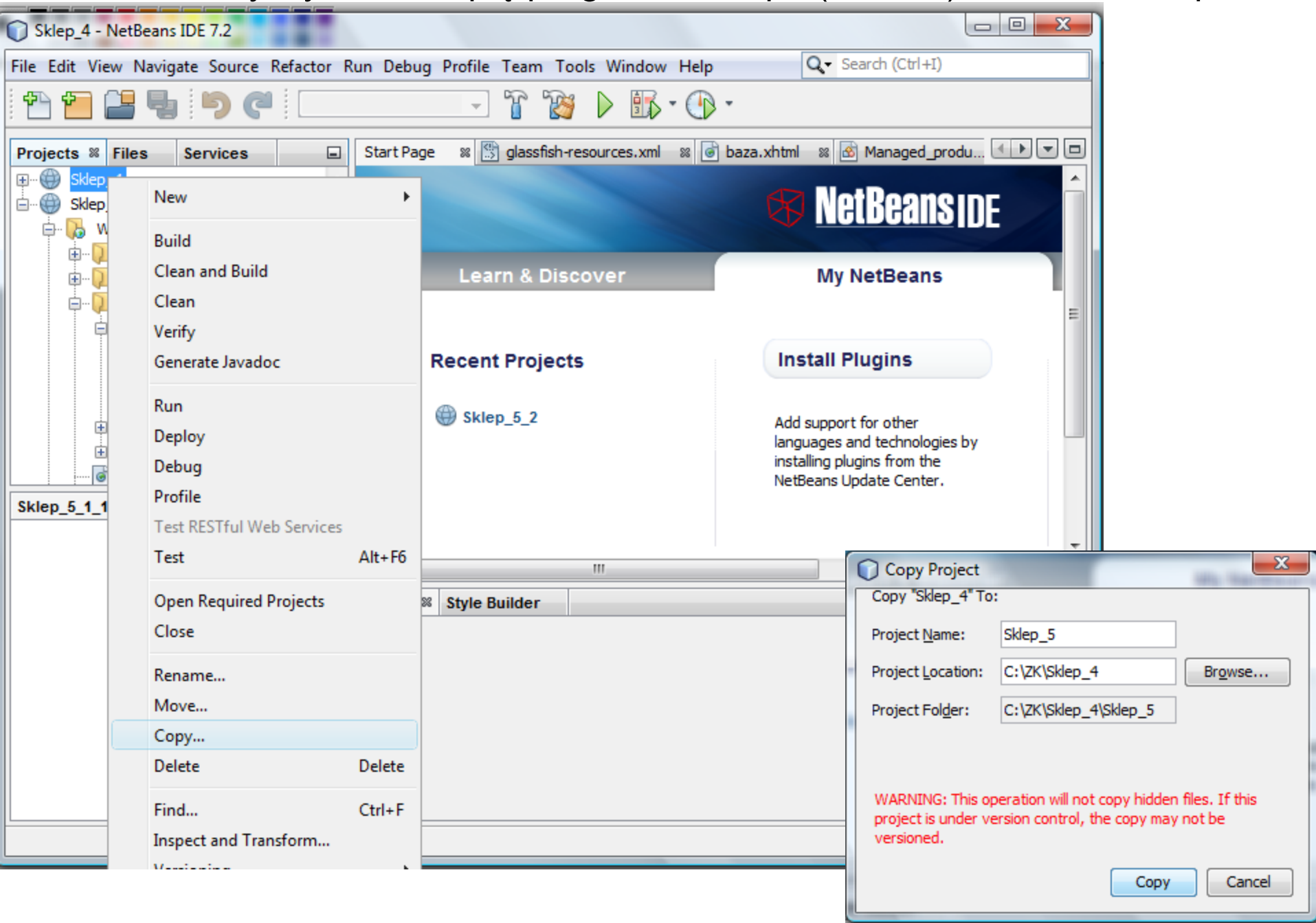


(3) Ustawienie kodowania typu UTF-8 w projektach Java SE oraz Java EE – należy zatwierdzić ustawienia klawiszem OK



1. Zastosowanie komponentu **h:selectOneMenu** (typu Drop Down List) na stronie **lista_produkow.xhtml** do wyboru promocji

1.1. Należy wykonać kopię programu Sklep4 (z lab. 3) o nazwie Sklep5



1.2. Dodanie do strony **dodaj_produk2.xhtml** komponentu typu **h:selectOneMenu** – do wprowadzania promocji

```
<h:outputLabel value="#{bundle['jsf.dodaj_produk2.podaj_promocja']}"  
               for="promocja" />  
<h:selectOneMenu id="promocja"  
                 value="#{managed_produk.promocja}"  
                 title="#{bundle['jsf.dodaj_produk2.podaj_promocja']}"  
                 required="true"  
                 requiredMessage="#{bundle['jsf.dodaj_produk2.podaj_promocja_blad']}">  
    <f:selectItems value="#{managed_produk.itemsAvailableSelectOne}"/>  
</h:selectOneMenu>
```


1.3. Dodanie metody w klasie **Fasada_warstwy_biznesowej** do generowania modelu listy rozwijanej, zawierającej wykaz promocji do wyboru z listy

```
public ArrayList<Integer> findAll() {  
    ArrayList<Integer> pom = new ArrayList();  
    pom.add(new Integer(0));  
    pom.add(new Integer(10));  
    pom.add(new Integer(20));  
    pom.add(new Integer(50));  
    return pom;  
}
```

1.4. Dodanie do klasy **Managed_produk**t metody obsługującej wybór z listy – właściwość **managed_produk.itemsAvailableSelectOne** dla znacznika **f:selectItems value**

```
public SelectItem[] getItemsAvailableSelectOne() {  
    return JsUtil.getSelectItems(fasada.findAll(), true);  
}
```

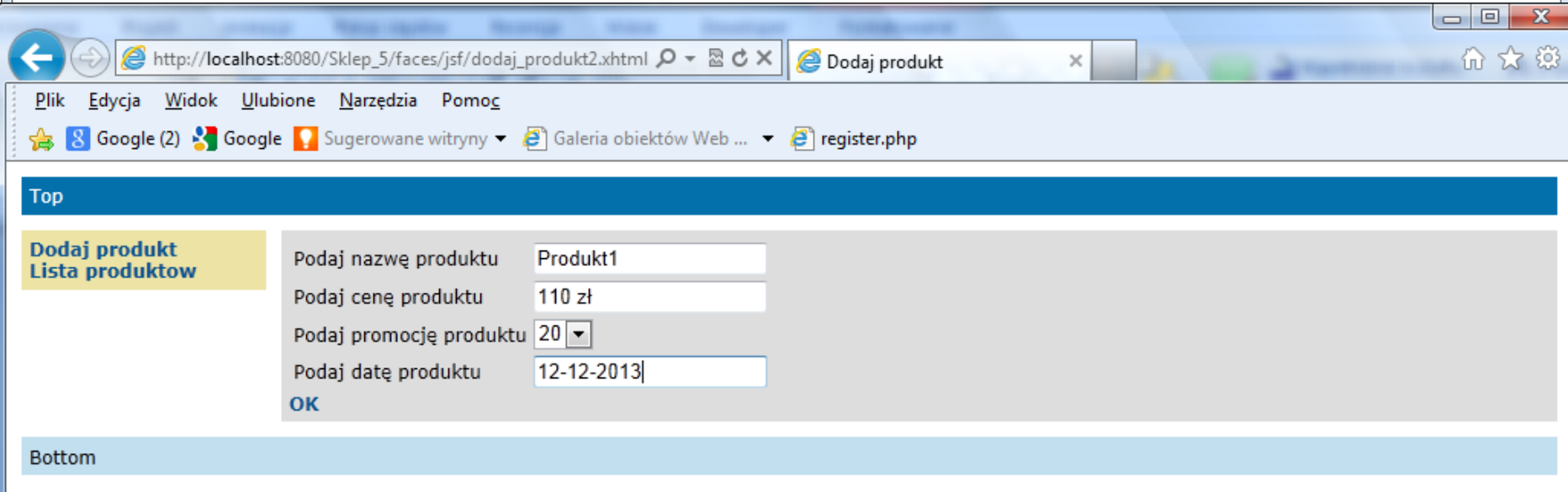
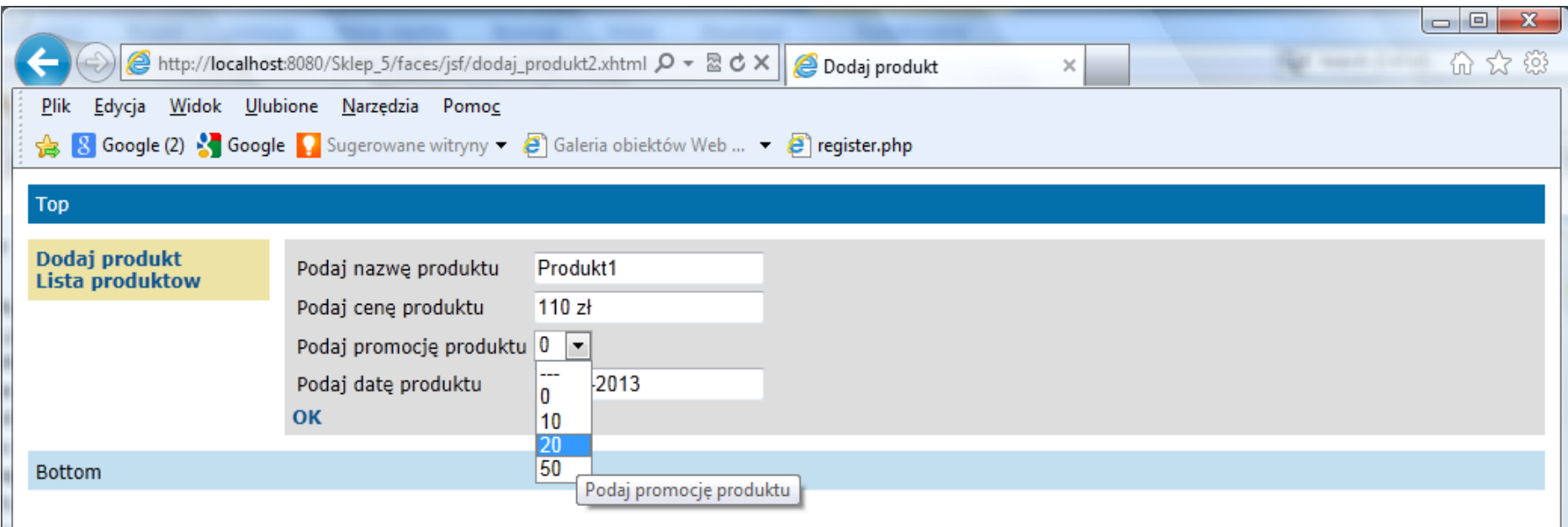
1.5. Należy dodać pakiet **jsf.util**, zdefiniować nową klasę **JsfUtil** i wstawić następujący kod metody **getSelectItems** do tej klasy

```
package jsf.util;

import java.util.List;
import javax.faces.model.SelectItem;

public class JsfUtil {
    public static SelectItem[] getSelectItems(List<?> entities, boolean selectOne) {
        int size = selectOne ? entities.size() + 1 : entities.size();
        SelectItem[] items = new SelectItem[size];
        int i = 0;
        if (selectOne) {
            items[0] = new SelectItem("", "---");
            i++; }
        for (Object x : entities) {
            items[i++] = new SelectItem(x, x.toString());
        }
        return items;
    }
}
```

1.6. Prezentacja działania programu (1)



1.7. Prezentacja programu (2)

Top

Dodaj produkt
Lista produktów

Nazwa produktu Produkt1
Cena netto produktu 110 zł
Promocja produktu 20
Data produkcji produktu Thursday, December 12, 2013 12:00:00 AM GMT
Cena brutto produktu 88.0

[Powrót](#)

Bottom

Top

Dodaj produkt
Lista produktów

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł

Bottom

2. Dodanie stronicowania zawartości komponentu **dataTable** na stronie **lista_produkow.xhtml**.

Kod dotyczący stronicowania oparty na kodzie wygenerowanym podczas tworzenia stron JSF na podstawie klas typu Entity (pakiet jsf.util) - przykład 3 przedstawiony na wykładzie:

http://zofia.kruczkiewicz.staff.iiar.pwr.wroc.pl/wyklady/javapk/PK_2.pdf

Wykład jest oparty na materiałach umieszczonych na stronach:

[Java EE & Java Web Learning Trail.](#)

[Tutorial Java EE 6](#)

2.1. Dodanie stronicowania stron – fragment pliku `lista_produkow.xhtml` z dodanym kodem znaczników odpowiedzialnym za stronicowanie (zaznaczony kolorem czerwonym – przed znacznikiem `dataTable`) – pierwszy znacznik `commandLink` służy do stronicowania „wstecz” a drugi do stronicowania „do przodu”

```
<ui:define name="content">
  <h:form styleClass="jsfcrud_list_form">
    <h:panelGroup id="messagePanel" layout="block">
      <h:messages errorStyle="color: red" infoStyle="color: green" layout="table"/>
    </h:panelGroup>
    <h:outputText escape="false" value="#{bundle['jsf.lista_produkow.pusta']}"
      rendered="#{managed_produkow.pagination.itemsCount == 0}"/>
    <h:panelGroup rendered="#{managed_produkow.pagination.itemsCount > 0}">
      <h:outputText value="#{managed_produkow.pagination.pageFirstItem + 1}
        ..#{managed_produkow.pagination.pageLastItem + 1}
        /#{managed_produkow.pagination.itemsCount}"/>&nbsp;
      <h:commandLink
        action="#{managed_produkow.previous}"
        value="#{bundle['jsf.lista_produkow.poprzedni']} #{managed_produkow.pagination.pageSize}"
        rendered="#{managed_produkow.pagination.hasPreviousPage}"/>&nbsp;
      <h:commandLink
        action="#{managed_produkow.next}"
        value="#{bundle['jsf.lista_produkow.nastepny']} #{managed_produkow.pagination.pageSize}"
        rendered="#{managed_produkow.pagination.hasNextPage}"/>&nbsp;
    </h:panelGroup>
  </h:form>
</ui:define>
```

2.2. Zmiana czasu życia obiektu typu **Managed_produk** do czasu trwania sesji za pomocą adnotacji **@SessionScoped**

```
package jsf;
```

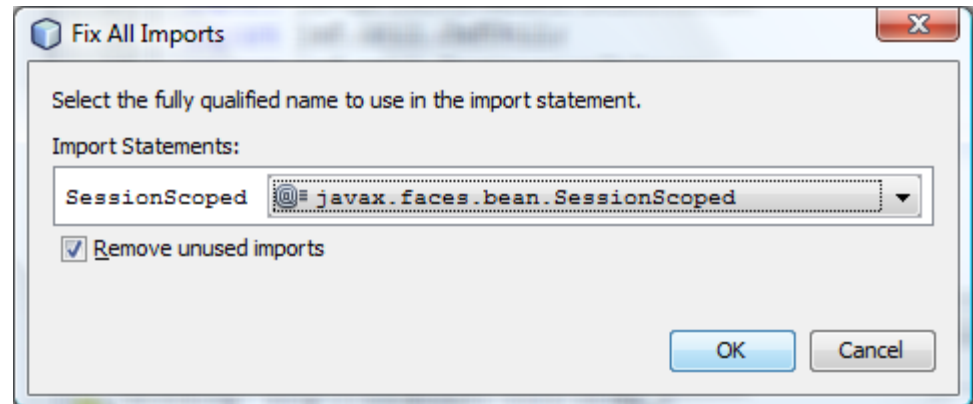
```
import Warstwa_biznesowa.Fasada_warstwy_biznesowej;  
import dto.Produkt_dto;  
import java.util.Date;  
import javax.ejb.EJB;  
import javax.faces.bean.ManagedBean;  
import javax.faces.bean.SessionScoped;  
import javax.faces.convert.NumberConverter;  
import javax.faces.model.DataModel;  
import javax.faces.model.ListDataModel;  
import javax.faces.model.SelectItem;  
import jsf.util.JsfUtil;  
import jsf.util.PaginationHelper;
```

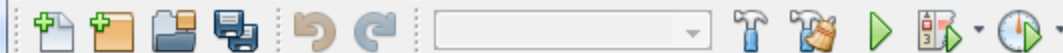
```
@ManagedBean
```

```
@SessionScoped
```

```
public class Managed_produk {
```

Podczas wprowadzania importu brakujących klas za pomocą **Fix Import** (wybrana pozycja po kliknięciu prawym klawiszem na powierzchnię okna edytora)





Projects Files Services

Sklep_5

- Web Pages
 - WEB-INF
 - jsf
 - dodaj_produk2.xhtml
 - lista_produk2ow.xhtml
 - rezultat2.xhtml
 - resources
 - index2.xhtml
 - template.xhtml
- Source Packages
 - <default package>
 - Bundle.properties
 - Warstwa_biznesowa
 - Fasada_warstwy_biznesowej.java
 - dto
 - jpa
 - jsf
 - Managed_produk2.java
 - jsf.util
 - JsfUtil.java
 - PaginationHelper.java
- Libraries
 - JDK 1.6 (Default)
 - GlassFish Server 3.1.2
 - Enterprise Beans
 - Configuration Files

Sklep_5_1_1

```
...html dodaj_produk2.xhtml Managed_produk2.java Managed_produk2.java...
Source History
1  /*...*/
5  package jsf;
6
7  import Warstwa_biznesowa.Fasada_warstwy_biznesowej;
8  import dto.Produkt_dto;
9  import java.util.Date;
10 import javax.ejb.EJB;
11 import javax.faces.bean.ManagedBean;
12 import javax.faces.bean.SessionScoped;
13 import javax.faces.convert.NumberConverter;
14 import javax.faces.model.DataModel;
15 import javax.faces.model.ListDataModel;
16 import javax.faces.model.SelectItem;
17 import jsf.util.JsfUtil;
18 import jsf.util.PaginationHelper;
19
20 @ManagedBean
21 @SessionScoped
22 public class Managed_produk2 {
23     @EJB
24     private Fasada_warstwy_biznesowej fasada;
25     private DataModel items;
26     private int stan = 1;
27     private Produkt_dto produkt_dto = new Produkt_dto();
28     private NumberConverter number_convert = new NumberConverter();
29     private PaginationHelper pagination;
```

2.3. Kod, który należy dodać i zmodyfikować w klasie **Managed_produk**t

```
public DataModel getItems() {
    if (items == null) {
        items = getPagination().createPageDataModel(); }
    return items;
}

public String dodaj_produk() {
    fasada.utworz_produk(produkt_dto);
    dane_produk();
    recreateModel();
    getPagination().nextPage();
    return "rezultat2";
}

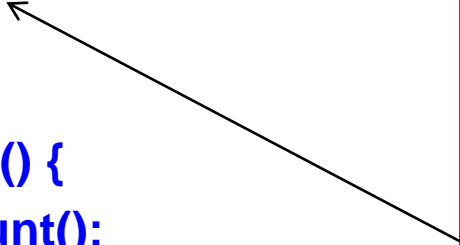
public void dane_produk() {
    stan = 1;
    produkt_dto = fasada.dane_produk();
    if (produkt_dto == null) {
        produkt_dto = new Produkt_dto();
        stan = 0; }
}
```

```
public PaginationHelper getPagination() {
    if (pagination == null) {
        pagination = new PaginationHelper(3) {

            @Override
            public int getItemCount() {
                return getFasada().count();
            }

            @Override
            public DataModel createPageDataModel() {
                int[] range = {getPageFirstItem(), getPageLastItem() + 1};
                return new ListDataModel(getFasada().findRange(range));
            }
        };
    }
    return pagination;
}
```

Ustalenie wartości atrybutu **pageSize**, w obiekcie klasy dziedziczącej po klasie **PaginationHelper** (definicję klasy podano na kolejnych slajdach) z zdefiniowanymi metodami **getItemCount** oraz **createPageDataModel**



```
private void recreateModel() {  
    items = null;  
}
```

Metoda usuwająca zawartość bieżącej strony – umożliwia metodzie **getItems** utworzenie zawartości wybranej strony

```
public String next() {  
    getPagination().nextPage();  
    recreateModel();  
    return "lista_produktow";  
}
```

Metoda obsługująca wybór następnego strony

```
public String previous() {  
    getPagination().previousPage();  
    recreateModel();  
    return "lista_produktow";  
}
```

Metoda obsługująca wybór poprzedniej strony

2.4. Do pakietu **jsf.util** należy dodać klasę **PaginationHelper**, zawierającą następujący kod:

```
package jsf.util;
import javax.faces.model.DataModel;

public abstract class PaginationHelper {
    private int pageSize;
    private int page;

    public PaginationHelper(int pageSize) {
        this.pageSize = pageSize;
    }

    public abstract int getItemCount();
    public abstract DataModel createPageDataModel();

    public int getPageFirstItem() {
        return page * pageSize; }
}
```

pageSize - rozmiar strony – czyli liczba wierszy komponentu,
dataTable
page – numer strony, startowa wartość równa 0

pageSize*page – liczba pozycji przypisanych do wszystkich stron, a jednocześnie numer danej wyświetlanej jako pierwsza pozycja na ostatniej stronie

```
public int getPageLastItem() {  
    int i = getPageFirstItem() + pageSize - 1;  
    int count = getItemsCount() - 1;  
    if (i > count) {  
        i = count; }  
    if (i < 0) {i = 0; }  
    return i;  
}
```

← Indeks ostatniej pozycji,
czyli numer ostatniej
danej, wyświetlany na
ostatniej stronie

```
public boolean hasNextPage() {  
    return (page + 1) * pageSize + 1 <= getItemsCount();  
}
```

← Sprawdzenie,
czyli liczba
danych wymaga
utworzenie
następnej strony

```
public void nextPage() {  
    if (hasNextPage()) {  
        page++;  
    }  
}
```

← Numer ostatniej strony

```
public boolean isHasPreviousPage() {  
    return page > 0;  
}
```

Sprawdzenie, czy istnieje poprzednia strona (pierwsza strona ma numer 0)

```
public void previousPage() {  
    if (isHasPreviousPage()) {  
        page--;  
    }  
}
```

Pobranie numeru poprzedniej strony

```
public int getPageSize() {  
    return pageSize;  
}
```

Pobranie rozmiaru strony

```
public void setPage() {  
    this.page = getItemCount()/pageSize;  
} //aktualizacja w dniu 26.05.2013  
}
```

Aktualizacja liczby stron

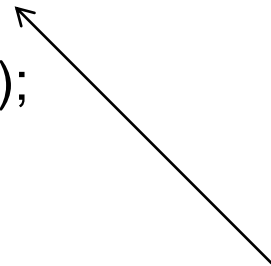
2.5. Kod, który należy dodać do klasy **Fasada_warstwy_biznesowej**

```
public int count() {  
    return produkty.size();  
}
```



Pobranie liczby danych

```
public ArrayList<Produkt_dto> findRange(int[] range) {  
    ArrayList<Produkt_dto> pom = new ArrayList();  
    for (int i = range[0]; i < range[1]; i++) {  
        pom.add(produkt_transfer(getProdukty().get(i)));  
    }  
    return pom;  
}
```



Pobranie podzbioru danych potrzebnych do wyświetlenia na stronie za pomocą metody **findRange**. Tablica **range** zawiera dwa elementy: pierwszy zawiera numer pierwszego elementu, drugi element zawiera numer ostatniego elementu z kolekcji **produkty**, które wyznaczają podzbiór danych pobieranych do wyświetlenia na stronie. Pobrane elementy z kolekcji produktu są przekształcone na obiekty transferowe typu **Produkt_dto**.

2.6. Uzupełnienie zawartości pliku **Bundle.properties**

jsf.lista_produkow.poprzedni=Poprzedni

jsf.lista_produkow.nastepny=Nastepny

2.7. Prezentacja stronicowania (1)

The screenshot shows a web browser window with the address bar containing `http://localhost:8080/Sklep_5/faces/jsf/lista_produkow.xhtml`. The page title is "Lista produktów". The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar also shows search engines like Google and Sugerowane witryny, along with a "register.php" icon.

At the top of the page, there is a blue bar with the text "Top". Below it, on the left, is a yellow button labeled "Dodaj produkt" and a blue link "Lista produktów". To the right of the button, the text "1..1/1" indicates the current page number.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł

At the bottom of the page, there is a light blue bar with the text "Bottom".

The screenshot shows the same web browser window as above, but the page content has changed to show the second page of the product list. The address bar and browser interface are identical.

The "Top" bar and the "Dodaj produkt" button are present. The text "1..2/2" indicates the current page number.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł
2	Produkt2	220 zł	20 %	czwartek, 12-12-2013	176 zł

The address bar now shows `http://localhost:8080/Sklep_5/faces/jsf/dodaj_produkow2.xhtml`.

The screenshot shows the same web browser window, now displaying the third page of the product list. The address bar and browser interface are identical.

The "Top" bar and the "Dodaj produkt" button are present. The text "1..3/3" indicates the current page number.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł
2	Produkt2	220 zł	20 %	czwartek, 12-12-2013	176 zł
3	Produkt3	330 zł	10 %	czwartek, 10-10-2013	297 zł

2.8. Prezentacja stronicowania (2)

The screenshot shows a web browser window with the URL `http://localhost:8080/Sklep_5/faces/jsf/lista_proc`. The browser's address bar and tabs are visible. The page content includes a menu bar with options like 'Plik', 'Edycja', 'Widok', 'Ulubione', 'Narzędzia', and 'Pomoc'. Below the menu, there are search engines (Google) and a 'register.php' link. A blue 'Top' bar is present. On the left, there is a yellow button labeled 'Dodaj produkt Lista produktów'. The main content area displays a table with the following data:

4..4/4 Popzedni 3					
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
4	Produkt4	440 zł	50 %	czwartek, 12-12-2013	220 zł

At the bottom of the page, there is a light blue 'Bottom' bar.

The screenshot shows the same web browser window as above, but with a different page of the product list. The pagination indicator shows '1..3/4 **Nastepny 3**'. The table contains the following data:

1..3/4 Nastepny 3					
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł
2	Produkt2	220 zł	20 %	czwartek, 12-12-2013	176 zł
3	Produkt3	330 zł	10 %	czwartek, 10-10-2013	297 zł

The rest of the browser interface, including the menu bar and search engines, remains the same as in the previous screenshot.

2.9. Prezentacja stronicowania (3)

The screenshot shows a web browser window with the URL `http://localhost:8080/Sklep_5/faces/jsf/lista_proc`. The browser's address bar and tabs are visible. The page content includes a blue header with "Top", a yellow button labeled "Dodaj produkt Lista produktów", and a table with 2 rows of product data. The table has 6 columns: Id produktu, Nazwa produktu, Cena netto produktu, Promocja produktu, Data produkcji produktu, and Cena brutto produktu. The table is followed by a blue footer with "Bottom".

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
4	Produkt4	440 zł	50 %	czwartek, 12-12-2013	220 zł
5	Produkt5	550 zł	10 %	czwartek, 10-10-2013	495 zł

The screenshot shows a web browser window with the same URL as the previous image. The page content is similar, but the table now displays 3 rows of product data. The table has 6 columns: Id produktu, Nazwa produktu, Cena netto produktu, Promocja produktu, Data produkcji produktu, and Cena brutto produktu. The table is followed by a blue footer with "Bottom".

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
4	Produkt4	440 zł	50 %	czwartek, 12-12-2013	220 zł
5	Produkt5	550 zł	10 %	czwartek, 10-10-2013	495 zł
6	Produkt6	660 zł	10 %	środa, 05-06-2013	594 zł

2.10. Prezentacja stronicowania (4)

The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/Sklep_5/faces/jsf/lista_prod`. The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar also shows several tabs: "Dno i obciac...", "Serwer poczto...", and "Lista produ...". The browser's toolbar includes icons for Google (2), Google, Sugerowane witryny, Galeria obiektów Web, and register.php.

The page content features a blue header with "Top" and a yellow button labeled "Dodaj produkt Lista produktów". Below this is a pagination indicator "7..7/7" and a "Poprzedni 3" link. A table displays product information:

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
7	Produkt7	770 zł	10 %	środa, 05-06-2013	693 zł

The page concludes with a blue footer containing "Bottom".

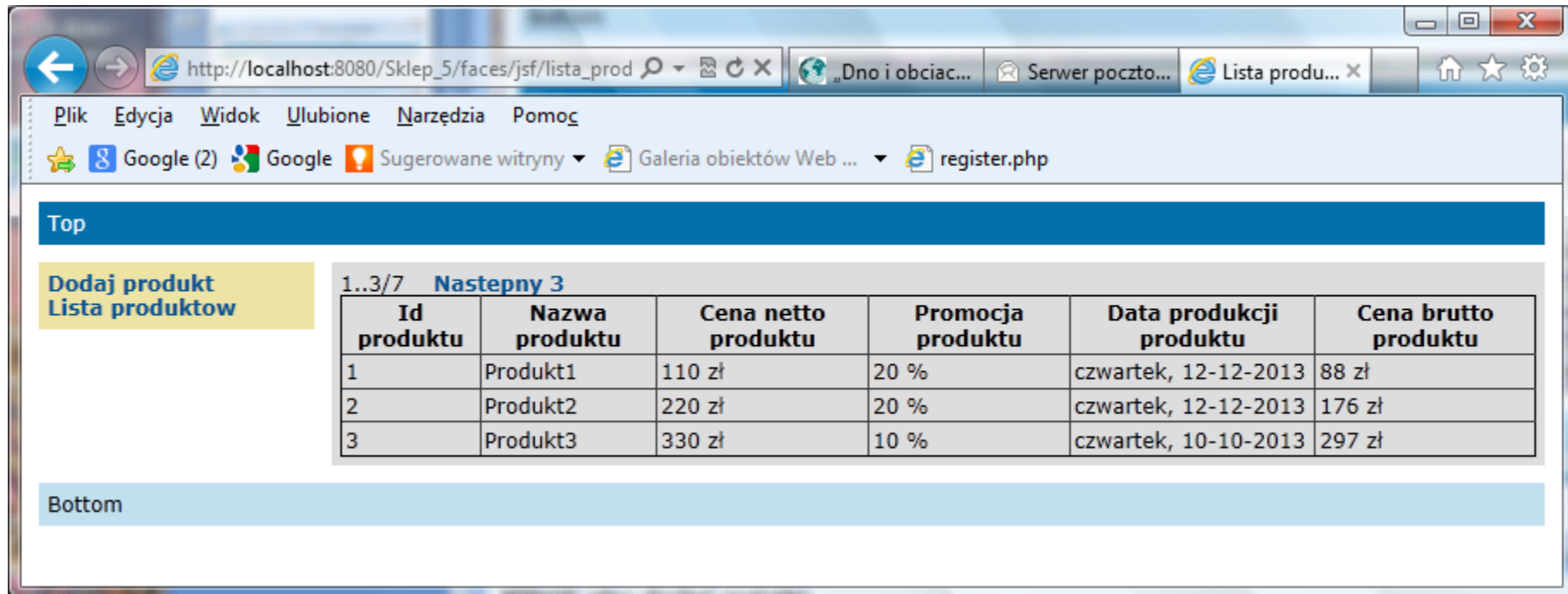
The screenshot shows a web browser window displaying a product list page with pagination "4..6/7". The browser's address bar shows `http://localhost:8080/Sklep_5/faces/jsf/lista_prod`. The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar also shows several tabs: "Dno i obciac...", "Serwer poczto...", and "Lista produ...". The browser's toolbar includes icons for Google (2), Google, Sugerowane witryny, Galeria obiektów Web, and register.php.

The page content features a blue header with "Top" and a yellow button labeled "Dodaj produkt Lista produktów". Below this is a pagination indicator "4..6/7" and links for "Poprzedni 3" and "Następny 3". A table displays product information:

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
4	Produkt4	440 zł	50 %	czwartek, 12-12-2013	220 zł
5	Produkt5	550 zł	10 %	czwartek, 10-10-2013	495 zł
6	Produkt6	660 zł	10 %	środa, 05-06-2013	594 zł

The page concludes with a blue footer containing "Bottom".

2.11. Prezentacja stronicowania (5)



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/Sklep_5/faces/jsf/lista_prod`. The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar contains several icons, including Google and a search engine. The main content area features a blue header with "Top" and a yellow sidebar with "Dodaj produkt" and "Lista produktów". The main content displays a table of products with pagination information "1..3/7" and "Następny 3". The table has six columns: "Id produktu", "Nazwa produktu", "Cena netto produktu", "Promocja produktu", "Data produkcji produktu", and "Cena brutto produktu". The table contains three rows of product data. A blue footer with "Bottom" is visible at the bottom of the page.

Top

Dodaj produkt
Lista produktów

1..3/7 **Następny 3**

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu
1	Produkt1	110 zł	20 %	czwartek, 12-12-2013	88 zł
2	Produkt2	220 zł	20 %	czwartek, 12-12-2013	176 zł
3	Produkt3	330 zł	10 %	czwartek, 10-10-2013	297 zł

Bottom

3. Wprowadzenie warunkowego renderowania strony [rezultat2.xhtml](#) w celu zapewnienia wieloużywalności tej strony

3.1. Dodanie do komponentu **dataTable** na stronie **lista_produkow.xhtml** kolumny z przyciskiem obsługującym operację **Rezultat**

```
<h:column>
```

```
<f:facet name="header">
```

```
  <h:outputText value="&nbsp;"/>
```

```
</f:facet>
```

```
<h:commandLink action="#{managed_produk.prepareView}"  
                value="#{bundle['jsf.lista_produk.rezultat']}"/>
```

```
</h:column>
```

Podany znacznik **h:column** należy umieścić jako ostatni w komponencie **dataTable**

3.2. Wprowadzenie renderowanych przycisków na stronie **rezultat2.xhtml** – w przypadku wywołania tej strony ze strony **dodaj_produkt2.xhtml** zmienna **powrot** jest równa 1 i następuje powrót do strony **index2.xhtml**, a w przypadku wywołania ze strony **lista_produktow.xhtml** zmienna **powrot** jest równa 0 – wtedy ze strony powraca się z powrotem do strony **lista_produkt.xhtml**

```
<h:commandButton id="powrot1"
```

```
    value="#{bundle['jsf.rezultat2.akcja']}"
```

```
    action="/faces/index2"
```

```
    rendered="#{managed_produkt.powrot!=0}"/>
```

```
<h:commandButton id="powrot2"
```

```
    value="#{bundle['jsf.rezultat2.akcja']}"
```

```
    action="#{managed_produkt.powrot}"
```

```
    rendered="#{managed_produkt.powrot==0}"/>
```

3.3. Dodanie do klasy **Managed_produk**t obsługi przycisku umożliwiającego przejście do strony **rezultat2.xhtml** ze strony **lista_produk**tow.xhtml (metoda **prepareView**) i powrót ze strony **rezultat2.xhtml** do strony **lista_produk**tow.xhtml (metoda **powrot**) oraz właściwości **powrot**

```
private int powrot = 1;
```

```
public int getPowrot() {  
    return powrot;  
}
```

```
public String prepareView() {  
    produkt_dto = (Produkt_dto) getItems().getRowData();
```

```
powrot = 0;
```

```
    stan=1;  
    return "rezultat2";
```

```
}
```

```
public String powrot() {
```

```
powrot = 1;
```

```
    produkt_dto = new Produkt_dto();  
    return "lista_produk
```

```
}
```

Właściwość **powrot** pozwala renderować przyciski umożliwiające powrót ze strony **rezultat2.xhtml** do strony **index2.xhtml** (**powrot = 1**) lub **lista_produk**tow.xhtml (**powrot = 0**).

```
public String dodaj_produkt() {
    fasada.utworz_produkt(produkt_dto);
    powrot = 1;
    dane_produktu();
    recreateModel();
    getPagination().nextPage();
    return "rezultat2";
}
```

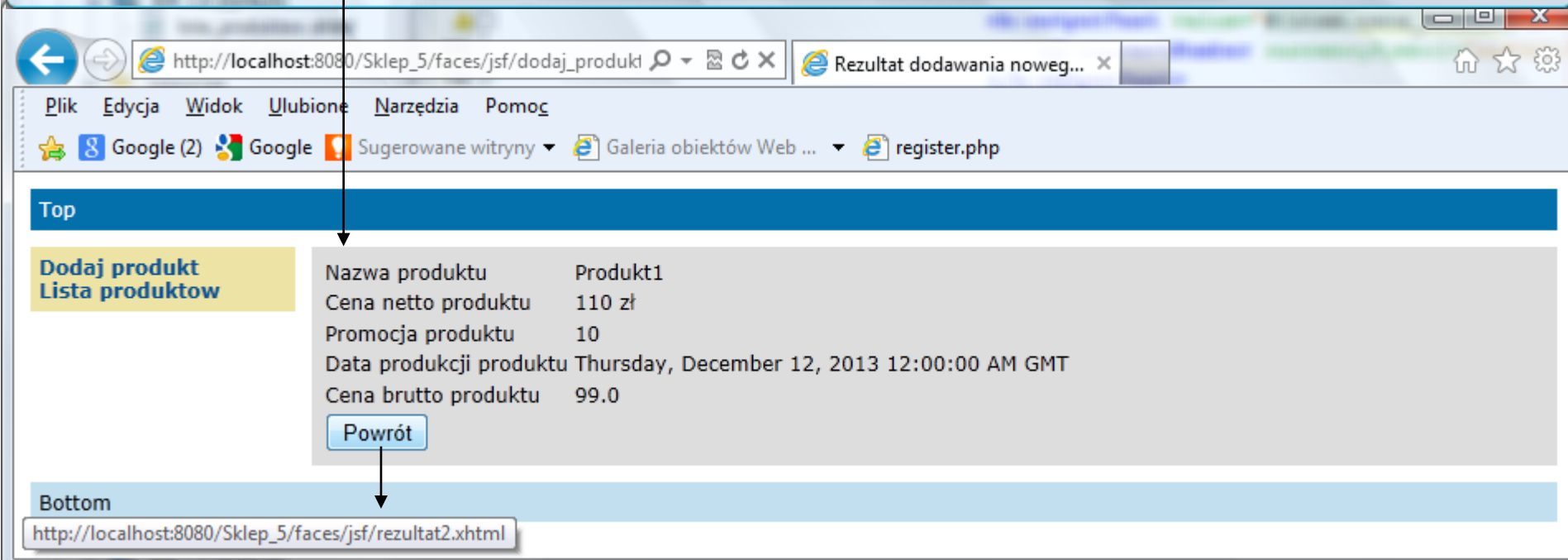
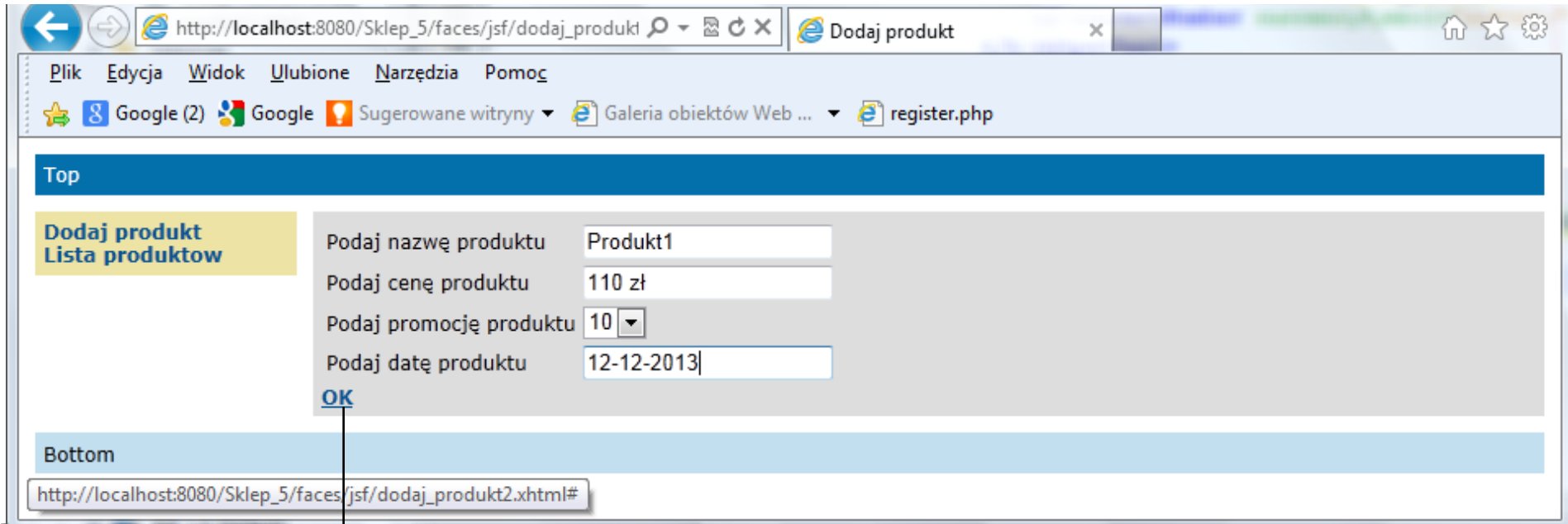
```
public DataModel getItems() {
    if (items == null) {
        items = getPagination().createPageDataModel();
    }
    powrot = 1;
    return items;
}
```

Zmienna **powrot** ma nadaną wartość 1 w przypadku opuszczenia strony **rezultat2.xhtml** za pomocą bloku „left” szablonu np. i przejście do strony **dodaj_produkt2.xhtml** lub **lista_produktow.xhtml** z renderowaniem właściwych przycisków.

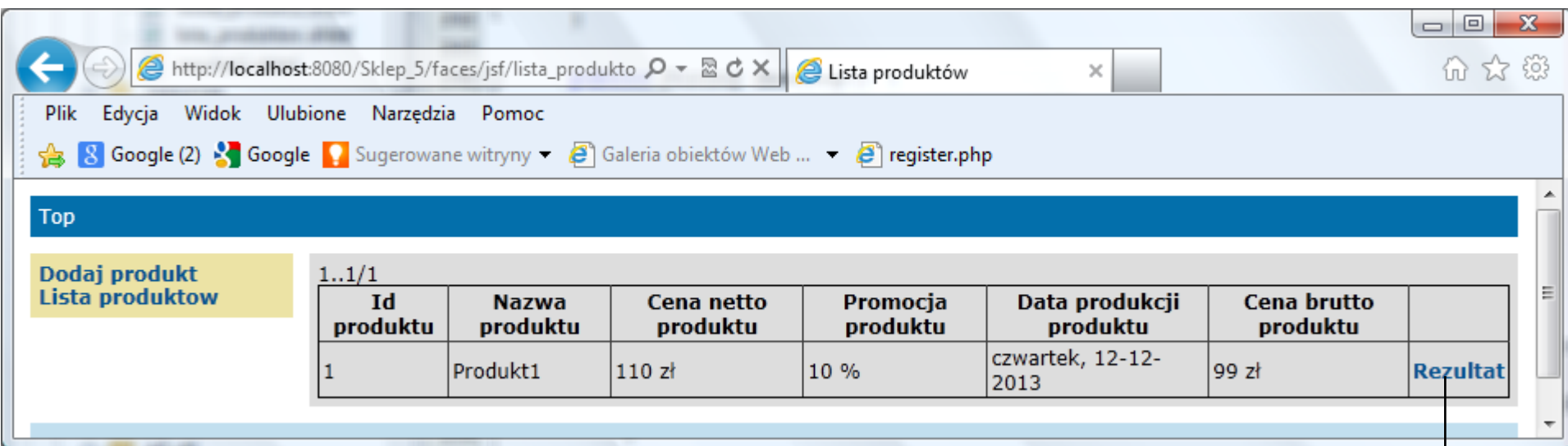
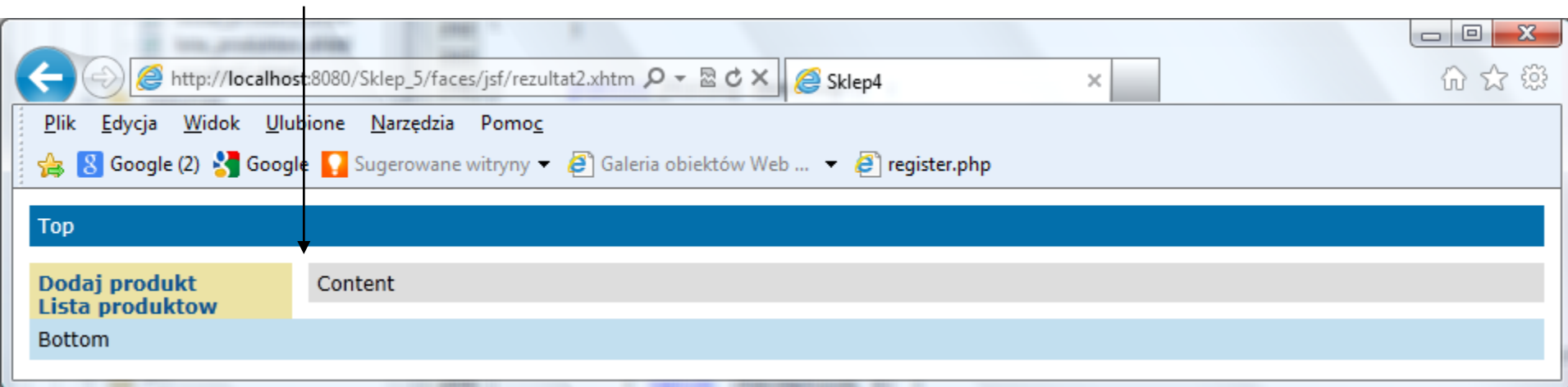
3.4. Uzupełnienie zawartości pliku **Bundle.properties**

jsf.lista_produktow.rezultat=Rezultat

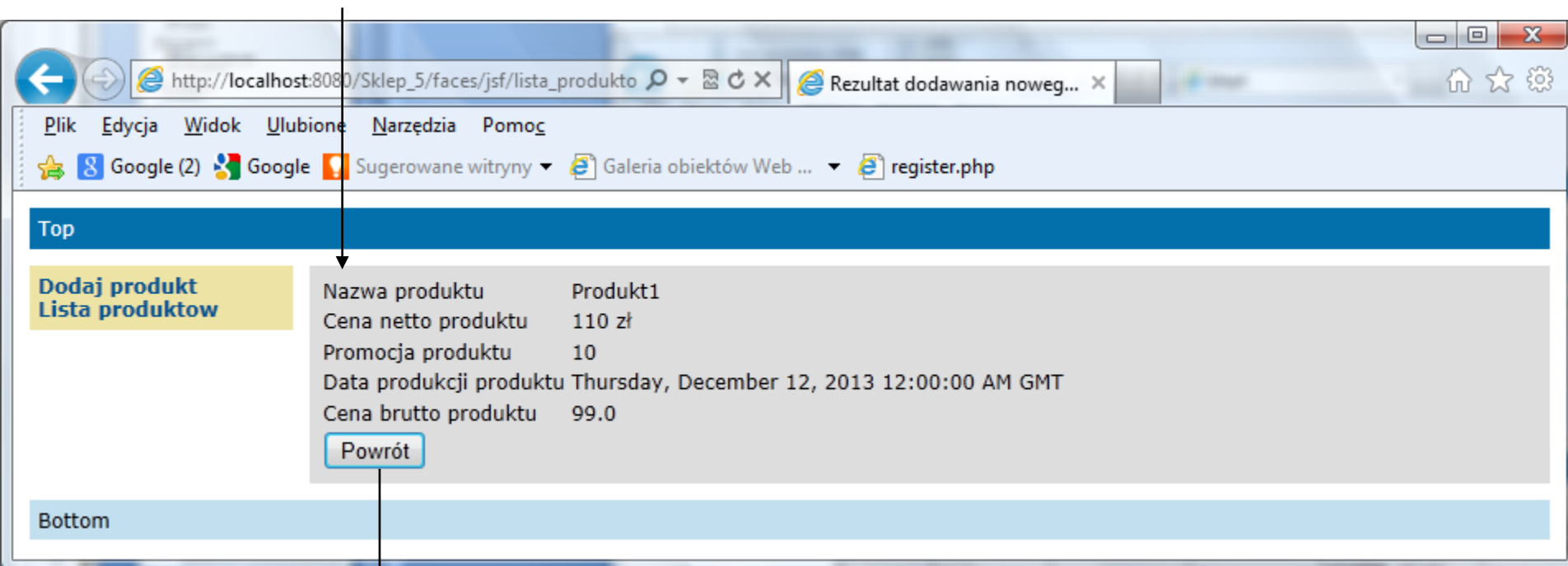
3.5. Prezentacja wieloużywalności strony **rezulta2.xhtml** (1)



3.6. Prezentacja wieloużywalności strony **rezultat2.xhtml** (2)



3.7. Prezentacja wieloużywalności strony [rezultat2.xhtml](#) (3)



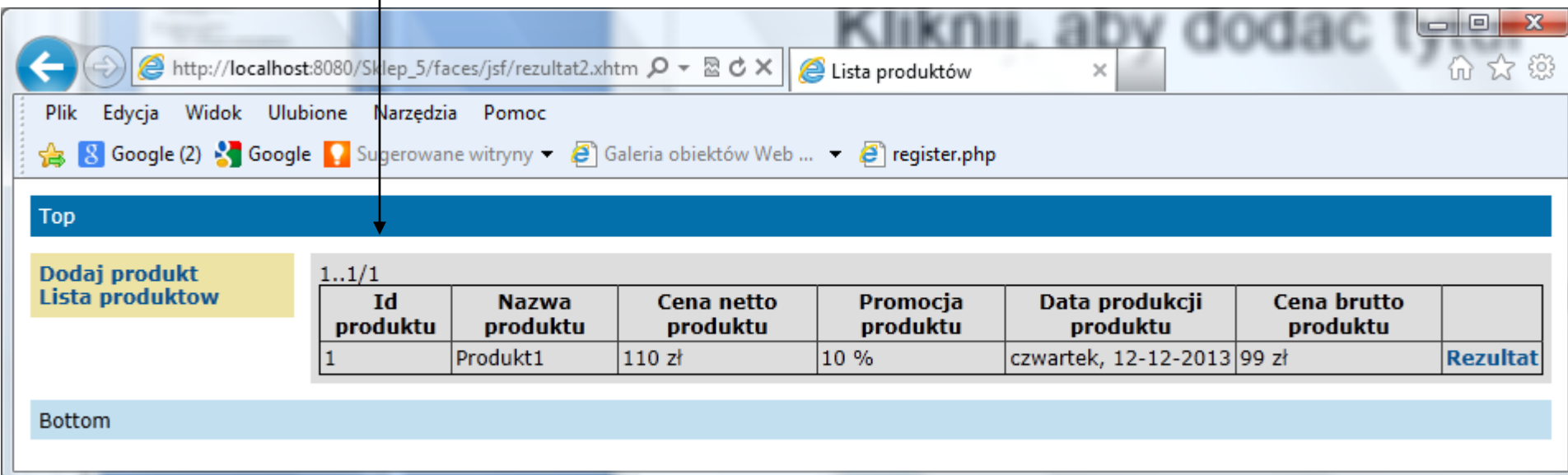
Top

Dodaj produkt
Lista produktów

Nazwa produktu Produkt1
Cena netto produktu 110 zł
Promocja produktu 10
Data produkcji produktu Thursday, December 12, 2013 12:00:00 AM GMT
Cena brutto produktu 99.0

Powrót

Bottom



Top

Dodaj produkt
Lista produktów

1..1/1

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat

Bottom

4. Wprowadzenie warunkowego renderowania strony [dodaj_produk2.xhtml](#) w celu zapewnienia wieloużywalności tej strony

4.1. Dodanie przycisku wywołującego stronę **dodaj_produk2.xhtml** w celu modyfikacji danych produktu z wybranego wiersza tabeli typu **dataTable** na stronie **lista_produkow.xhtml**.

```
<h:column>
```

```
<f:facet name="header">
```

```
    <h:outputText value="&nbsp;"/>
```

```
</f:facet>
```

```
<h:commandLink action="#{managed_produk.prepareView}"  
                value="#{bundle['jsf.lista_produkow.rezultat']}" />
```

```
<h:outputText value=" " />
```

```
<h:commandLink action="#{managed_produk.prepareEdit}"  
                value="#{bundle['jsf.lista_produkow.edycja']}" />
```

```
<h:outputText value=" " />
```

```
</h:column>
```

4.2. Dodanie nowych komunikatów do pliku [Bundle.properties](#)

Produkt_zmieniony=Zmiana danych produktu

Blad_modyfikacji=Nie dokonano zmiany danych produktu

jsf.lista_produkow.edycja=Edycja

4.3. Dodanie zmiennej **zmiana** do klasy **Managed_produk**t w celu renderowania strony **dodaj_produk2.xhtml**

@ManagedBean

@SessionScoped

```
public class Managed_produk {
```

```
    @EJB
```

```
    private Fasada_warstwy_biznesowej fasada;
```

```
    private DataModel items;
```

```
    private int stan = 1;
```

```
    private Produkt_dto produkt_dto = new Produkt_dto();
```

```
    private NumberConverter number_convert = new NumberConverter();
```

```
    private PaginationHelper pagination;
```

```
    private int powrot = 1;
```

```
    private int zmiana = 1;
```

```
public int getZmiana() {
```

```
    return zmiana;
```

```
}
```

```
public String prepareEdit() {  
    produkt_dto = (Produkt_dto) getItem().getRowData();  
    zmiana = 0;  
    return "dodaj_produk2";  
}
```

```
public String update() {  
    try {  
        getFasada().edit(produkt_dto);  
        produkt_dto=new Produkt_dto();  
        zmiana = 1;  
        recreateModel(); //dodano w dniu 26.05.2013  
        JsfUtil.addSuccessMessage(ResourceBundle.getBundle("/Bundle").getString("Produkt_zmieniony"));  
        return "lista_produk2";  
    } catch (Exception e) {  
        JsfUtil.addErrorMessage(e, ResourceBundle.getBundle("/Bundle").getString("Blad_modyfikacji"));  
        return null;  
    }  
}
```

Metoda **prepareEdit**, wywołana po kliknięciu na link **Edycja** na stronie **lista_produk2.xhtml** nadaje wartość 0 zmiennej **zmiana**, co powoduje przejście ze strony **lista_produk2.xhtml** do strony **dodaj_produk2.xhtml** i powrót po edycji do strony **lista_produk2.xhtml**

```
public DataModel getItems() {  
    if (items == null) {  
        items = getPagination().createPageDataModel();    }  
    zmiana = 1;  
    powrot = 1;  
    return items;  
}
```

```
public String dodaj_produkt() {  
    fasada.utworz_produkt(produkt_dto);  
    powrot = 1;  
    zmiana=1;  
    dane_produktu();  
    recreateModel();  
    getPagination().nextPage();  
    return "rezultat2";  
}
```

4.4. Zmiany na stronie **dodaj_produkt_2.xhtml**

```
<h:inputText
  id="nazwa"
  title="#{bundle['jsf.dodaj_produkt2.podaj_nazwa']}"
  value="#{managed_produkt.nazwa}"
  required="true,,
  requiredMessage="#{bundle['jsf.dodaj_produkt2.podaj_nazwa_blad']}"
  disabled="#{managed_produkt.zmiana==0}" >
</h:inputText>
```

Dodanie atrybutu **disabled** pozwala pokazać widok komponentu na stronie jako nieaktywny – nie można wprowadzić nazwy produktu

```
<h:commandLink action="#{managed_produkt.dodaj_produkt}"
  value="#{bundle['jsf.dodaj_produkt2.akcja']}"
  rendered="#{managed_produkt.zmiana==1}"/>
<h:commandLink action="#{managed_produkt.update}"
  value="#{bundle['jsf.dodaj_produkt2.akcja']}"
  rendered="#{managed_produkt.zmiana==0}"/>
```

Dodawanie nowego produktu

Modyfikacja danych produktu

4.5. Metoda **edit** w klasie **Fasada_warstwy_biznesowej** wywołana podczas obsługi zdarzenia kliknięcia na przycisk OK. na stronie **dodaj_produk2.xhtml**, gdy renderowany jest przycisk dla wartości zmiennej **zmiana=0** i wywołana jest metoda **update** z klasy **Managed_produk**

```
public void edit(Produkt_dto o) {  
    for (Produkt p : getProdukty()) {  
        if (p.equals_(new Long(o.getId())) != null) {  
            p.setCena(o.getCena());  
            p.setData_produkcji(o.getData_produkcji());  
            p.setPromocja(o.getPromocja());  
            break;  
        }  
    }  
}
```

4.6. Metoda **equals_**, dodana do klasy **Produkt** w celu wyszukania obiektu typu Produkt do edycji – wywołana z metody edit z klasy **Fasada_warstwy_biznesowej**

```
public Produkt equals_(Object o)
{
    if(id.equals(o))
        return this;
    else return null;
}
```


4.7. Dodane metody do klasy **JsfUtil** z pakietu **jsf.util** do obsługi błędów np. w metody edit klasy **Fasada_warstwy_biznesowej**

```
public static void addErrorMessage(Exception ex, String defaultMsg) {
    String msg = ex.getLocalizedMessage();
    if (msg != null && msg.length() > 0) {
        addErrorMessage(msg);
    } else {
        addErrorMessage(defaultMsg);    }
}

public static void addErrorMessage(String msg) {
    FacesMessage facesMsg =
        new FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
    FacesContext.getCurrentInstance().addMessage(null, facesMsg);
}

public static void addSuccessMessage(String msg) {
    FacesMessage facesMsg =
        new FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
    FacesContext.getCurrentInstance().addMessage("successInfo", facesMsg);
}
```

4.8. Prezentacja procesu modyfikacji danych (1)

The image displays two screenshots of a web browser showing a data modification process. The top screenshot shows a table of products, and the bottom screenshot shows the 'Dodaj produkt' form.

Top Screenshot: Lista produktów

URL: http://localhost:8080/Sklep_5/faces/jsf/li:

1..2/2

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja
2	Produkt2	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja

Bottom Screenshot: Dodaj produkt

URL: http://localhost:8080/Sklep_5/faces/jsf/li:

Podaj nazwę produktu

Podaj cenę produktu

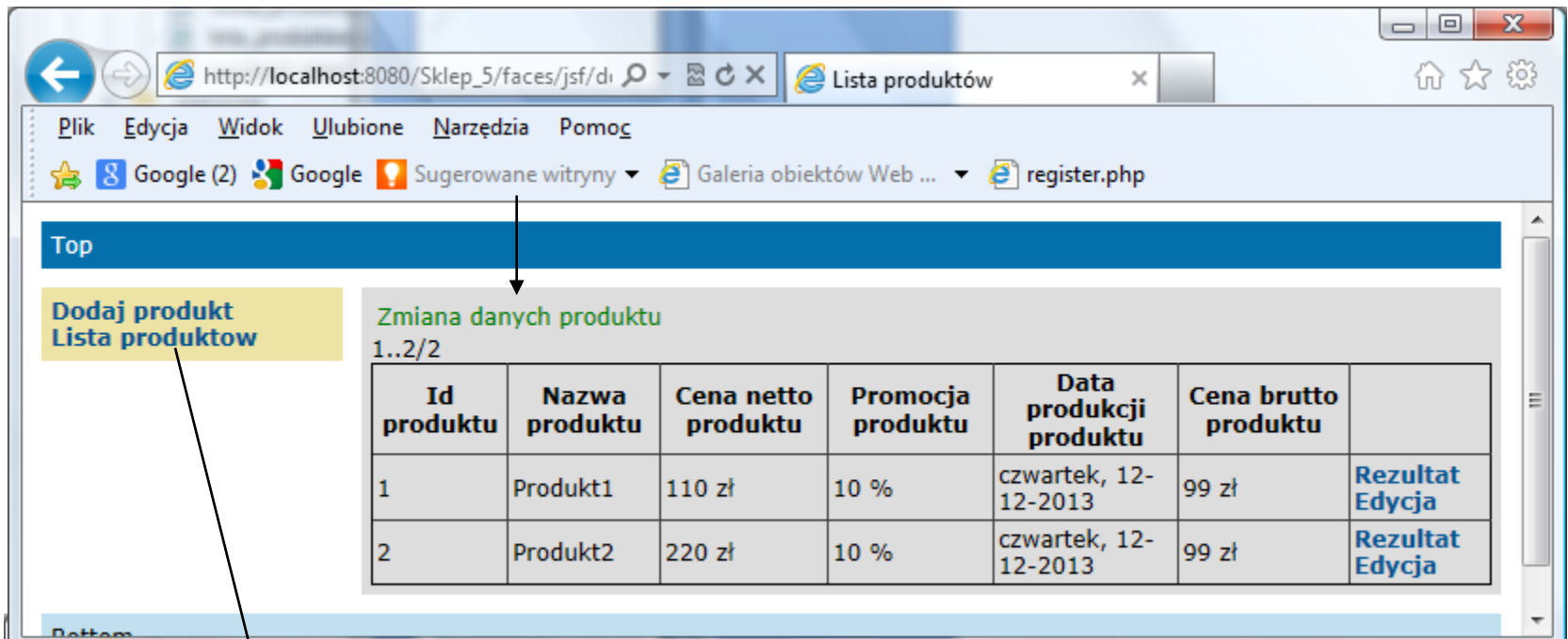
Podaj promocję produktu

Podaj datę produktu

[OK](#)

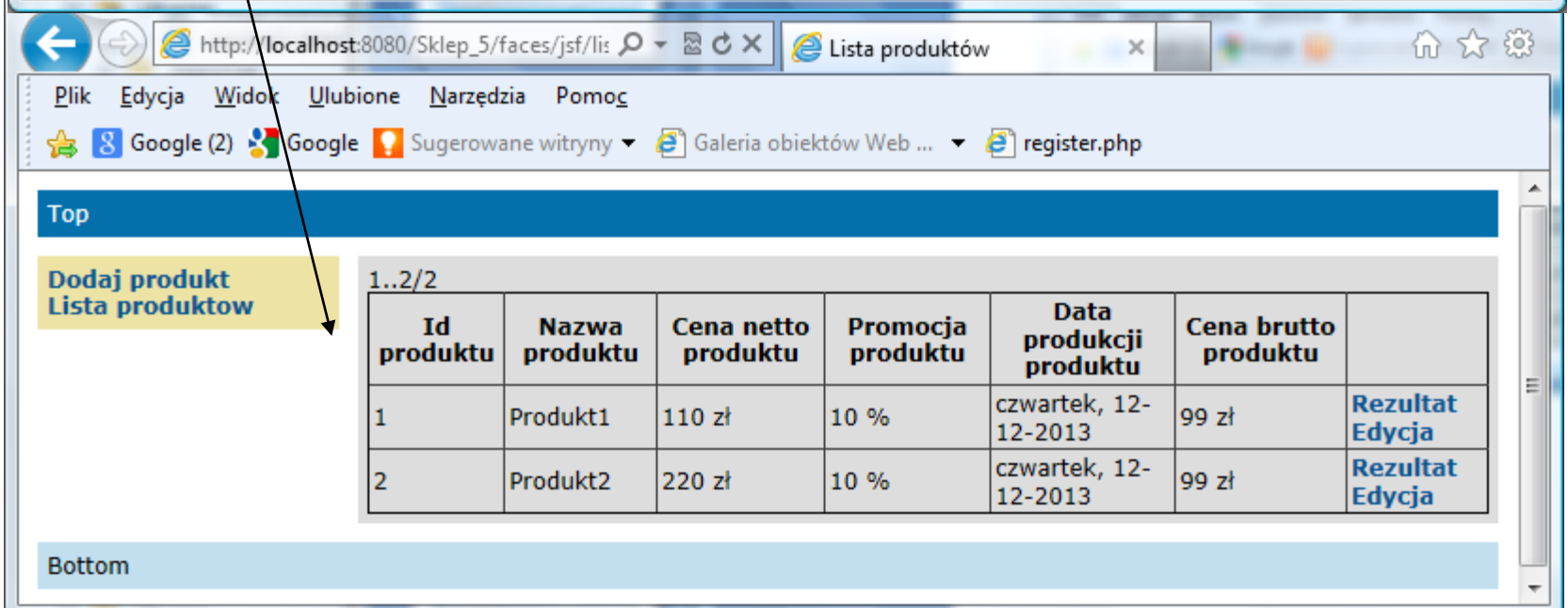
An arrow points from the 'Edycja' link in the top screenshot to the 'Dodaj produkt' form in the bottom screenshot.

4.9. Prezentacja procesu modyfikacji danych (1)



The screenshot shows a web browser window with the URL `http://localhost:8080/Sklep_5/faces/jsf/di`. The page title is "Lista produktów". The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar shows "Lista produktów" and "register.php". The page content includes a "Top" navigation bar, a yellow button labeled "Dodaj produkt Lista produktów", and a section titled "Zmiana danych produktu" with a sub-header "1..2/2". Below this is a table with 7 columns: "Id produktu", "Nazwa produktu", "Cena netto produktu", "Promocja produktu", "Data produkcji produktu", "Cena brutto produktu", and an empty column. The table contains two rows of data. A black arrow points from the "Zmiana danych produktu" section to the "Dodaj produkt" button in the second screenshot.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja
2	Produkt2	220 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja



The screenshot shows a web browser window with the URL `http://localhost:8080/Sklep_5/faces/jsf/lis`. The page title is "Lista produktów". The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The address bar shows "Lista produktów" and "register.php". The page content includes a "Top" navigation bar, a yellow button labeled "Dodaj produkt Lista produktów", and a section titled "1..2/2" with a sub-header "1..2/2". Below this is a table with 7 columns: "Id produktu", "Nazwa produktu", "Cena netto produktu", "Promocja produktu", "Data produkcji produktu", "Cena brutto produktu", and an empty column. The table contains two rows of data. A black arrow points from the "Dodaj produkt" button in this screenshot to the "Zmiana danych produktu" section in the first screenshot.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja
2	Produkt2	220 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja

5. Dodanie na stronie
lista_produkow.xhtml przycisku do
usuwania danych

5.1. Dodanie przycisku do usuwania produktu wybranego w wierszu tabeli typu **dataTable** na stronie **lista_produkow.xhtml**

```
<h:column>
<f:facet name="header">
    <h:outputText value="&nbsp;" />
</f:facet>
<h:commandLink action="#{managed_produk.prepareView}"
                value="#{bundle['jsf.lista_produkow.rezultat']}" />
<h:outputText value=" " />
<h:commandLink action="#{managed_produk.prepareEdit}"
                value="#{bundle['jsf.lista_produkow.edycja']}" />
<h:outputText value=" " />
<h:commandLink action="#{managed_produk.destroy}"
                value="#{bundle['jsf.lista_produkow.usun']}" />
</h:column>
```

5.2. Uzupełnienie zawartości pliku **Bundle.properties**

Usunieto_produk=Produkt został usuniety

Blad_usuwania=Produkt nie został usuniety

jsf.lista_produkow.usun=Usun

5.3. Dodanie metod do obsługi usuwania danych w klasie **Managed_produk**t

```
public String destroy() {
    produkt_dto = (Produkt_dto) getItems().getRowData();
    performDestroy();
    recreateModel();
    return "lista_produkutow";
}

private void performDestroy() {
    try {
        getFasada().remove(produkt_dto);
        JsUtil.addSuccessMessage(ResourceBundle.getBundle("/Bundle").getString("Usunieto_produk"));
    } catch (Exception e) {
        JsUtil.addErrorMessage(e, ResourceBundle.getBundle("/Bundle").getString("Blad_usuwania"));
    }
}
```

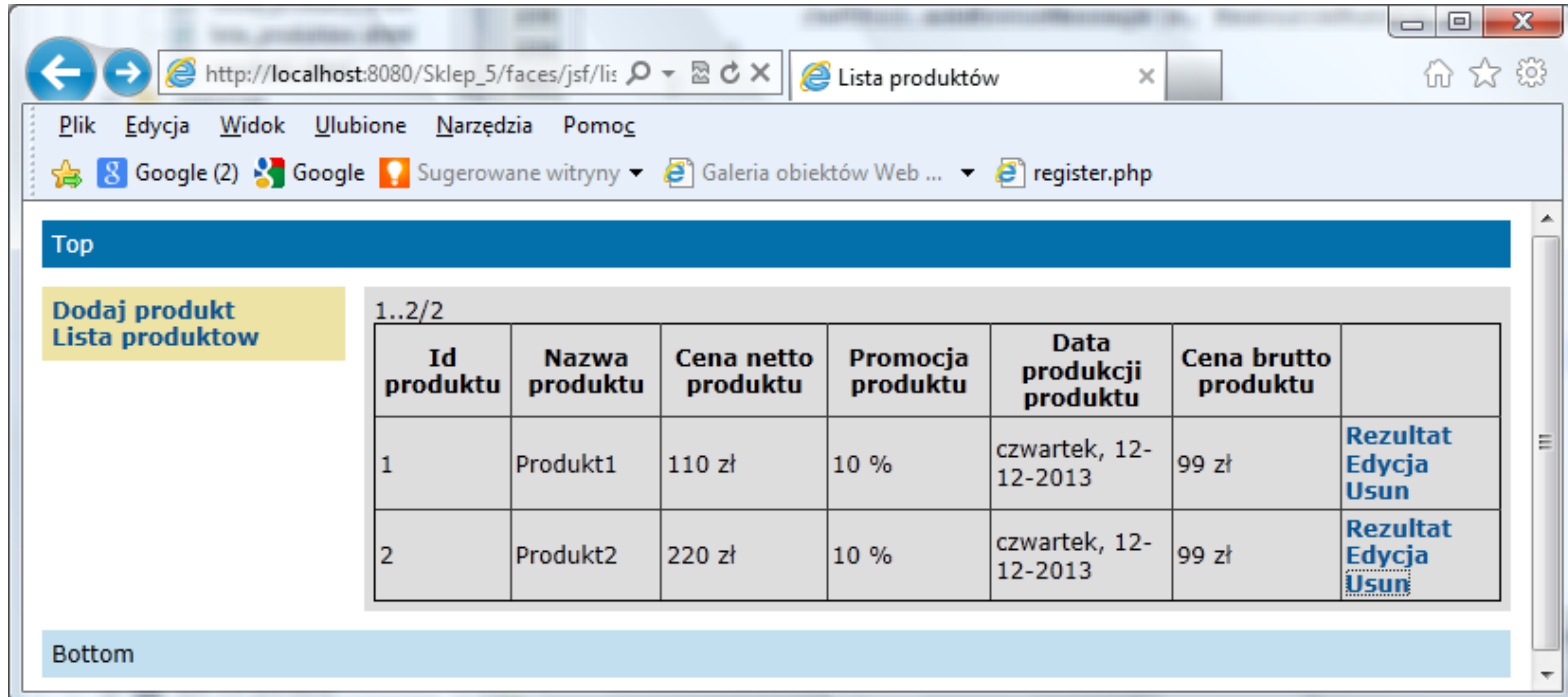
5.4. Dodanie metod **remove** do klasy **Fasada_warstwy_biznesowej** –modyfikacja kodu metody **utworz_produkt**. Metoda **max_klucz** wyznacza wartość unikatowego klucza

```
public void utworz_produkt(Produkt_dto produkt_dto) {
    Produkt produkt = wykonaj_produkt(produkt_dto);
    dodaj_produkt(produkt);
}
Produkt wykonaj_produkt(Produkt_dto produkt_dto) {
    Produkt produkt = new Produkt();
    max_klucz(); //dodano w dniu 26.05.2013
    produkt.setId(new Long(klucz));
    produkt.setNazwa(produkt_dto.getNazwa());
    produkt.setCena(produkt_dto.getCena());
    produkt.setPromocja(produkt_dto.getPromocja());
    produkt.setData_produkcji(produkt_dto.getData_produkcji());
    return produkt;
}
public void remove(Produkt_dto p) {
    Produkt produkt = wykonaj_produkt(p);
    getProdukty().remove(produkt);
}
void max_klucz() { //dodano w dniu 26.05.2013
    long max = 0;
    for (Produkt p : produkty)
        if (p.getId() > max) max = p.getId();
    klucz = max + 1; }
```

Refaktoryzacja metody **utworz_produkt** przez dodanie metody **wykonaj_produkt**, która również została zastosowana w kodzie nowej metody **remove**

Metoda **max_klucz** wyznacza wartość kolejnego największego klucza zapewniając jego unikatowość

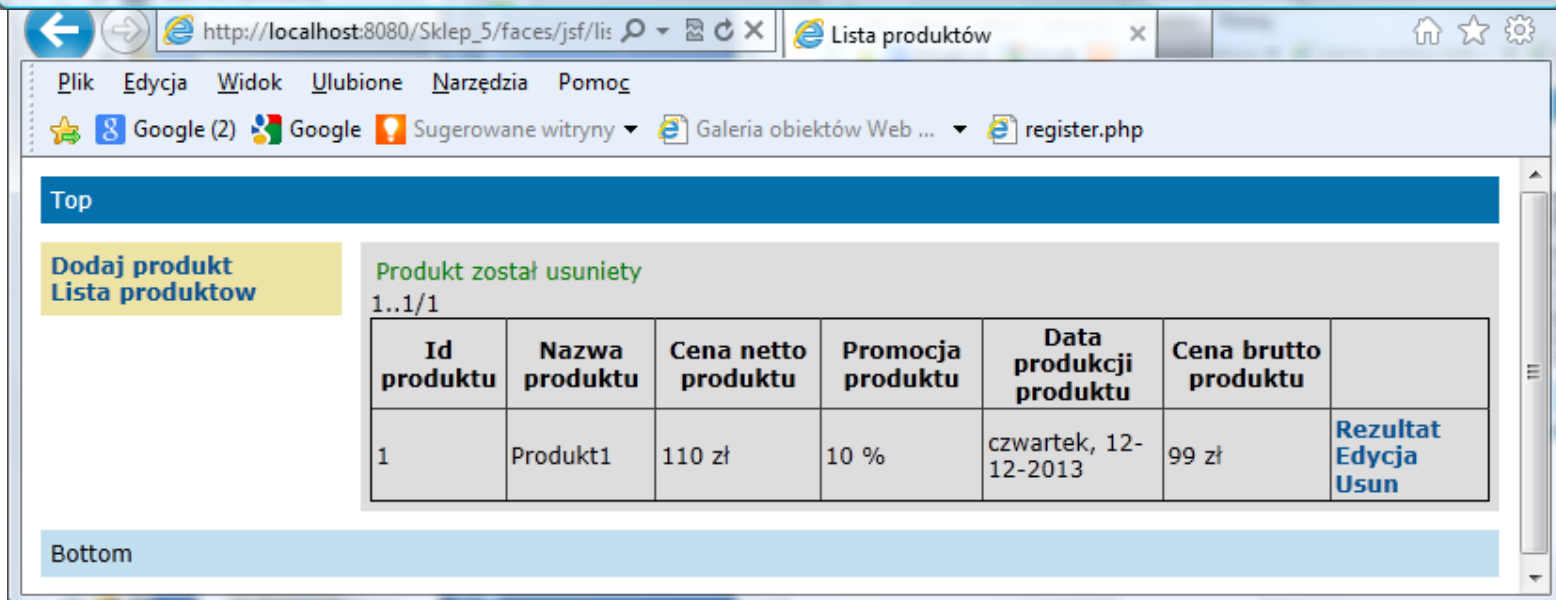
5.5. Prezentacja usuwania tytułu książki (1)



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/Sklep_5/faces/jsf/li...` and the page title "Lista produktów". The browser interface includes a menu bar with "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". Below the menu bar are search engines (Google) and a "Sugerowane witryny" dropdown. The main content area has a blue header "Top" and a yellow button "Dodaj produkt Lista produktów". A table displays two products:

1..2/2						
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun
2	Produkt2	220 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun

The bottom of the page features a blue bar labeled "Bottom".



The screenshot shows the same web browser window after a product has been deleted. The address bar and menu bar are identical. The main content area now displays a green message: "Produkt został usunięty". Below the message, the table shows only one product:

1..1/1						
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun

The bottom of the page features a blue bar labeled "Bottom".

5.6. Przeniesienie panelu komunikatów **<h:messages** do części Bottom szablonu (plik **template.xhtml**)

```
<div style="height:150px">
```

```
<div id="left">
```

```
<h:link outcome="/faces/jsf/dodaj_produkt2" value="Dodaj produkt"/> <br/>
```

```
<h:link outcome="/faces/jsf/lista_produktow" value="Lista produktow"/>
```

```
</div>
```

```
div id="content" class="left_content">
```

```
<ui:insert name="content">Content</ui:insert>
```

```
</div>
```

```
</div>
```

```
<div id="bottom" class="left_content">
```

```
<h3>Status</h3>
```

```
<h:panelGroup id="messagePanel" layout="block">
```

```
<h:messages errorStyle="color: red" infoStyle="color: green"  
layout="table"/>
```

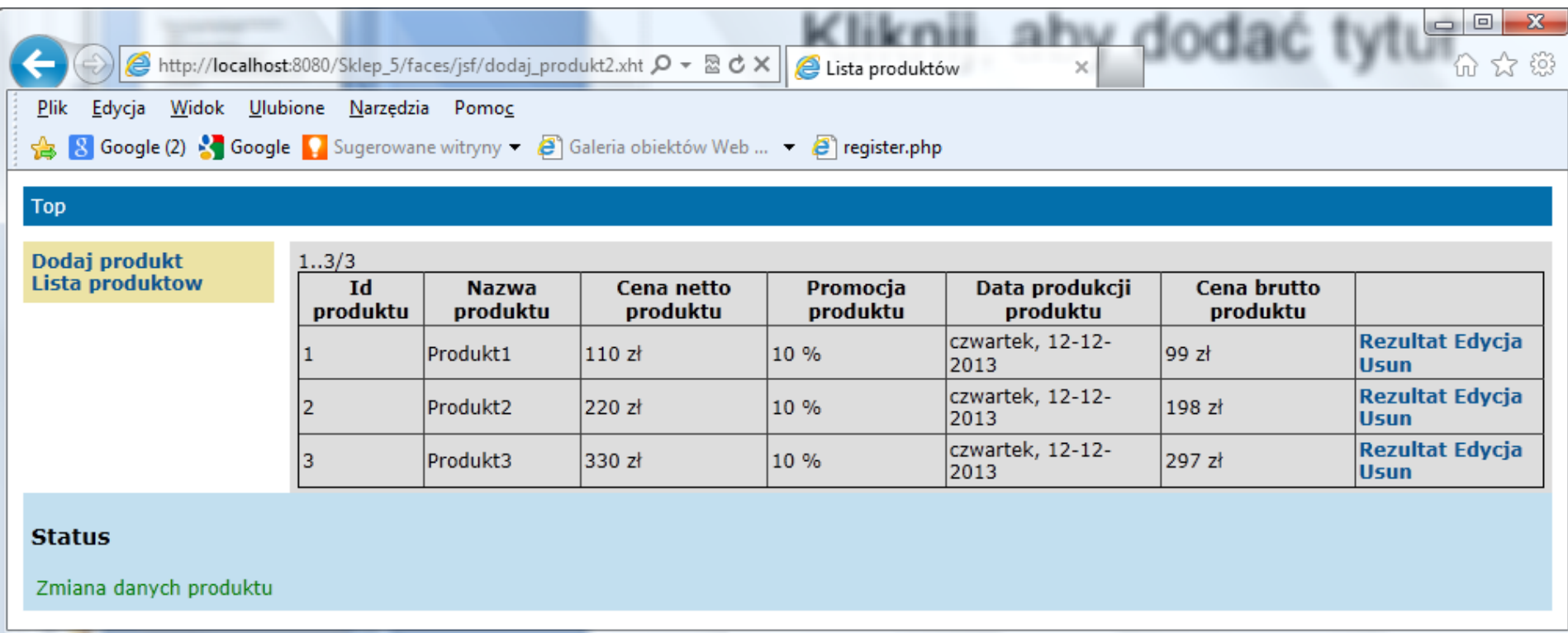
```
</h:panelGroup>
```

```
</div>
```

5.7. Wyeliminowanie panelu wiadomości **<h:messages** ze strony **lista_produkow.xhtml** – teraz komunikaty wyświetlane są w części **Bottom** o nazwie **Status** (poprzedni slajd)

```
<ui:define name="content">  
  <h:form styleClass="jsfcrud_list_form">  
    <h:panelGroup id="messagePanel" layout="block">  
      <h:messages errorStyle="color: red" infoStyle="color:  
green"  
      layout="table"/>  
    </h:panelGroup>
```

5.8. Prezentacja usuwania produktu (1)



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/Sklep_5/faces/jsf/dodaj_produk2.xht`. The browser's menu bar includes 'Plik', 'Edycja', 'Widok', 'Ulubione', 'Narzędzia', and 'Pomoc'. The address bar contains 'Lista produktów'. The page content includes a 'Dodaj produkt' button and a table of products.

Dodaj produkt
Lista produktów

1..3/3

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun
2	Produkt2	220 zł	10 %	czwartek, 12-12-2013	198 zł	Rezultat Edycja Usun
3	Produkt3	330 zł	10 %	czwartek, 12-12-2013	297 zł	Rezultat Edycja Usun

Status
Zmiana danych produktu

5.9. Prezentacja usuwania tytułu książki (2)

The screenshot shows a web browser window with the URL `http://localhost:8080/Sklep_5/faces/jsf/lista_produkow.xh1`. The page title is "Lista produktów". The browser's address bar shows the URL and the page title. The browser's menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The browser's search bar shows "Google (2)", "Google", "Sugerowane witryny", "Galeria obiektów Web ...", and "register.php".

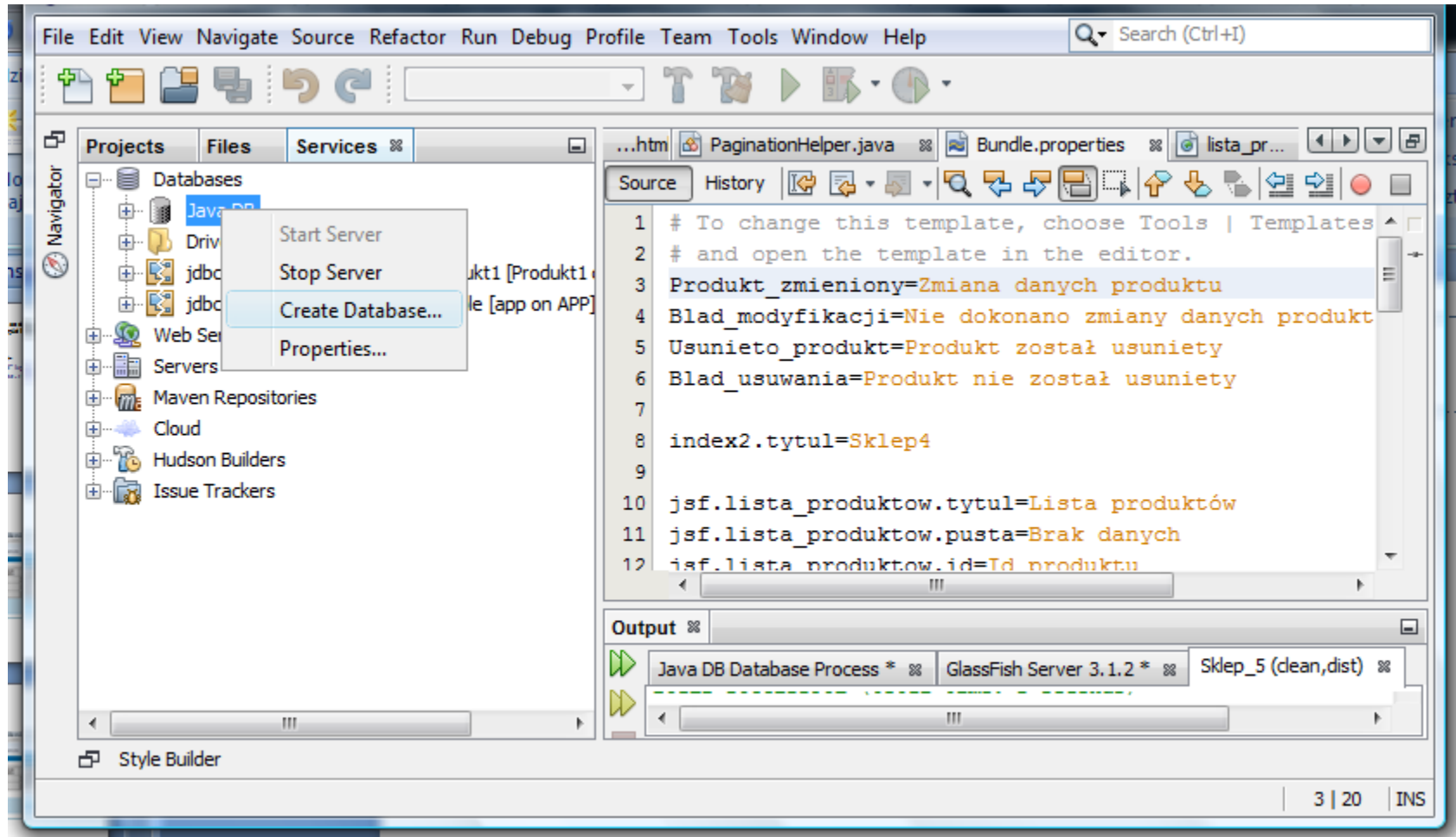
The page content includes a "Top" link, a "Dodaj produkt" button, and a "Lista produktów" link. The product list is displayed in a table with the following data:

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
1	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun
3	Produkt3	330 zł	10 %	czwartek, 12-12-2013	297 zł	Rezultat Edycja Usun

Below the table, there is a "Status" section with the message "Produkt został usuniety".

6. Zapis produktów do bazy danych metodą ORM (Object Relational Mapping)

6.1 . Zapis do bazy danych – należy utworzyć pustą bazę danych w zakładce **Services** (prawy przycisk myszy na pozycji JavaDB po rozwinięciu pozycji Database i wybór **Create Database**)

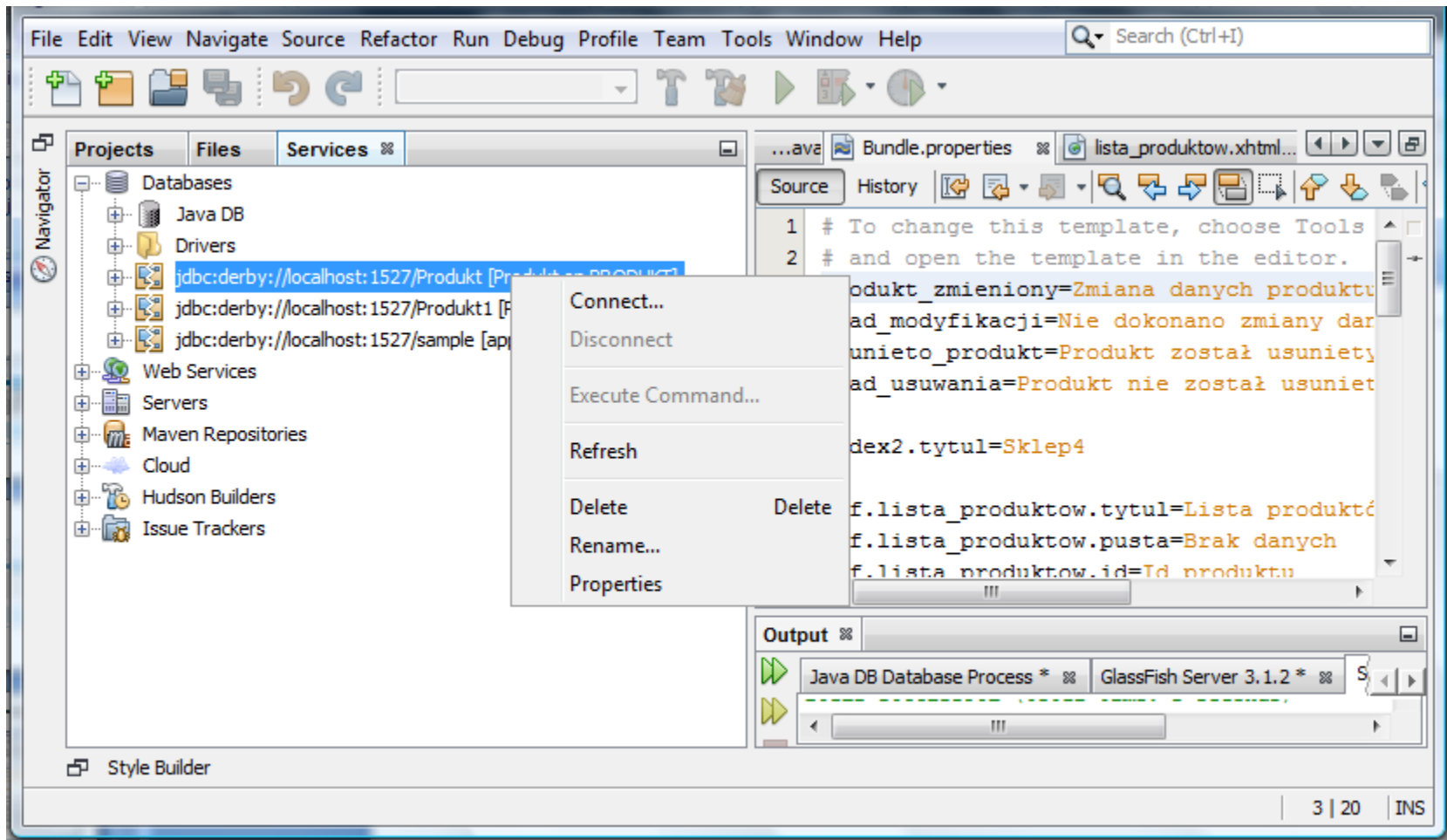


6.2. Wpisanie nazwy bazy danych (Database Name), loginu (User Name) i hasła (Password)

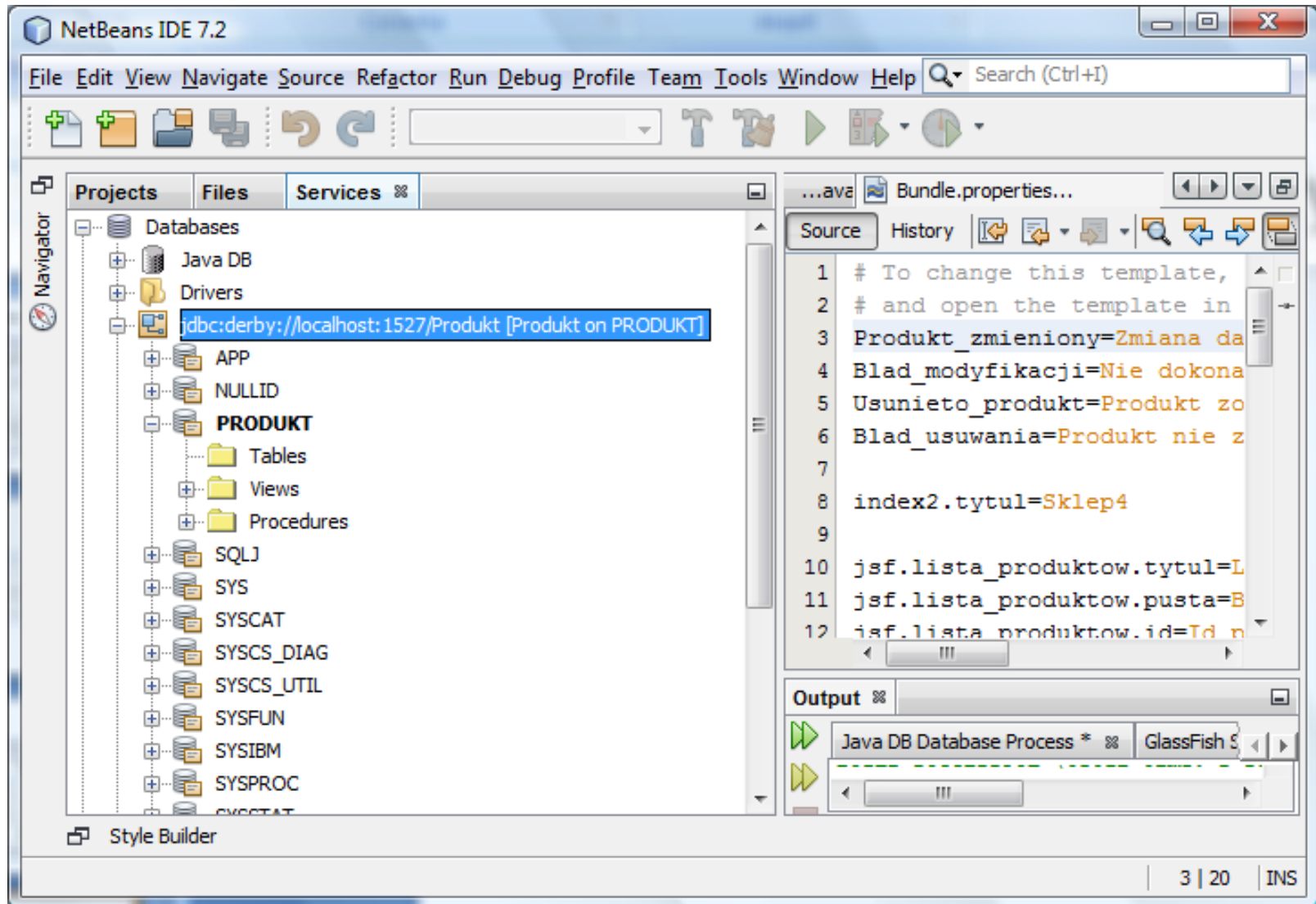
The screenshot shows a dialog box titled "Create Java DB Database" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- Database Name:** A text input field containing the text "Produkt".
- User Name:** A text input field containing the text "Produkt".
- Password:** A password input field with ten black dots representing the masked password.
- Confirm Password:** A password input field with ten black dots representing the masked password.
- Database Location:** A text field containing the path "C:\Users\kruczkiewicz\.netbeans-derby".
- Properties...** A button located to the right of the Database Location field.
- OK** and **Cancel** buttons are located at the bottom right of the dialog.

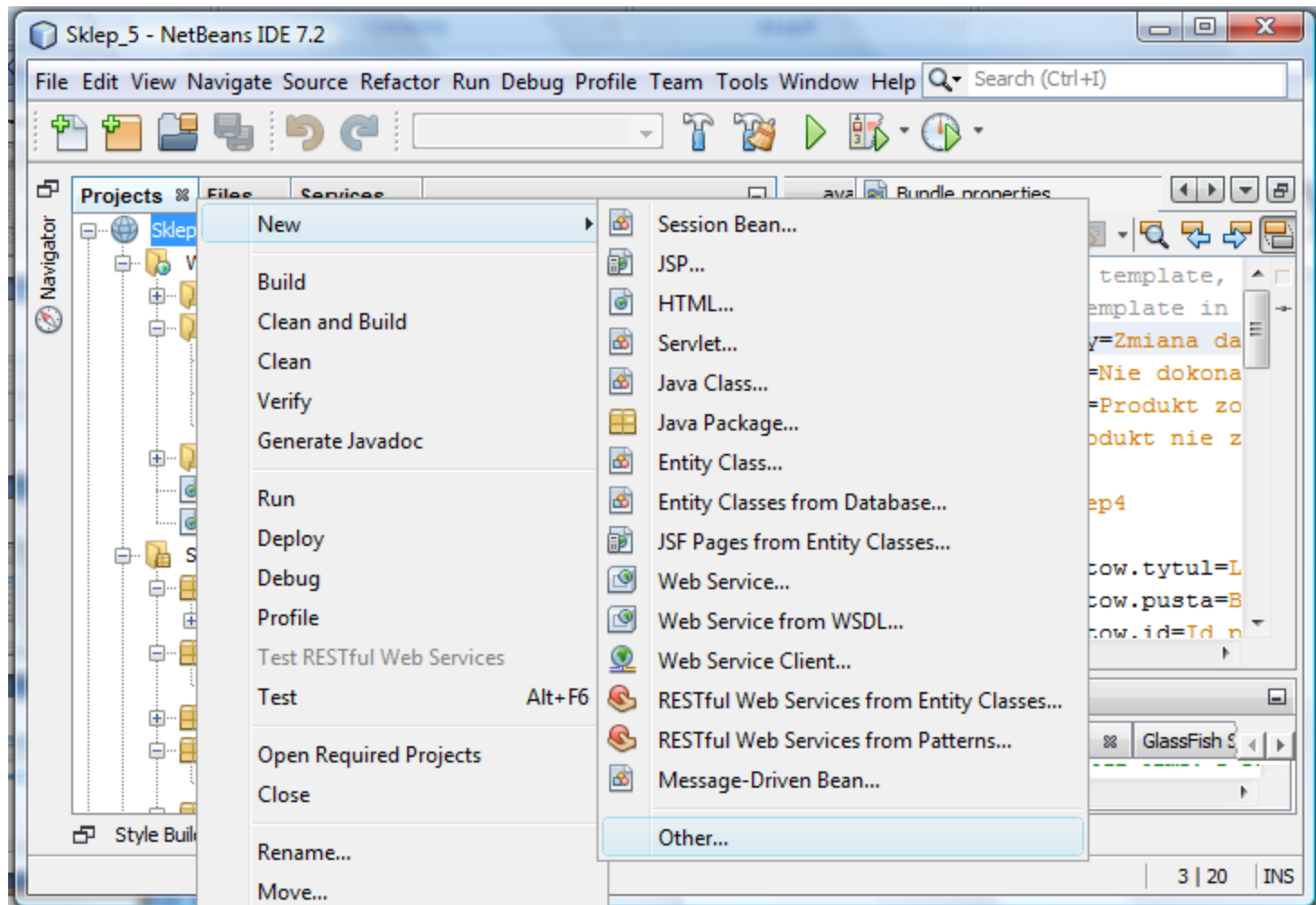
6.3. Połączenie z pustą bazą danych (wybór pozycji Connect po kliknięciu prawym klawiszem myszy na połączeniu)



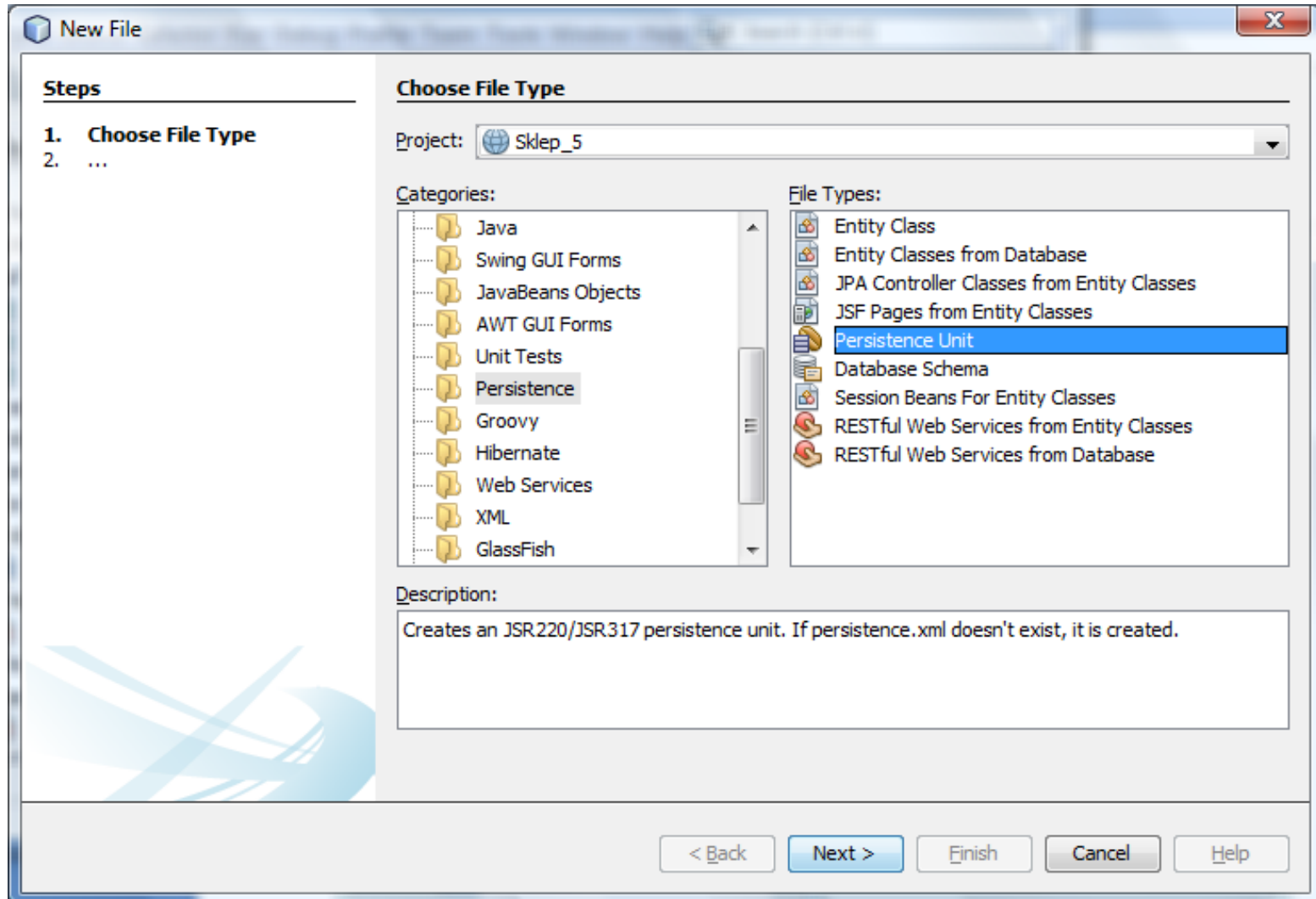
6.4. Widok utworzonej pustej bazy danych **Produkt**



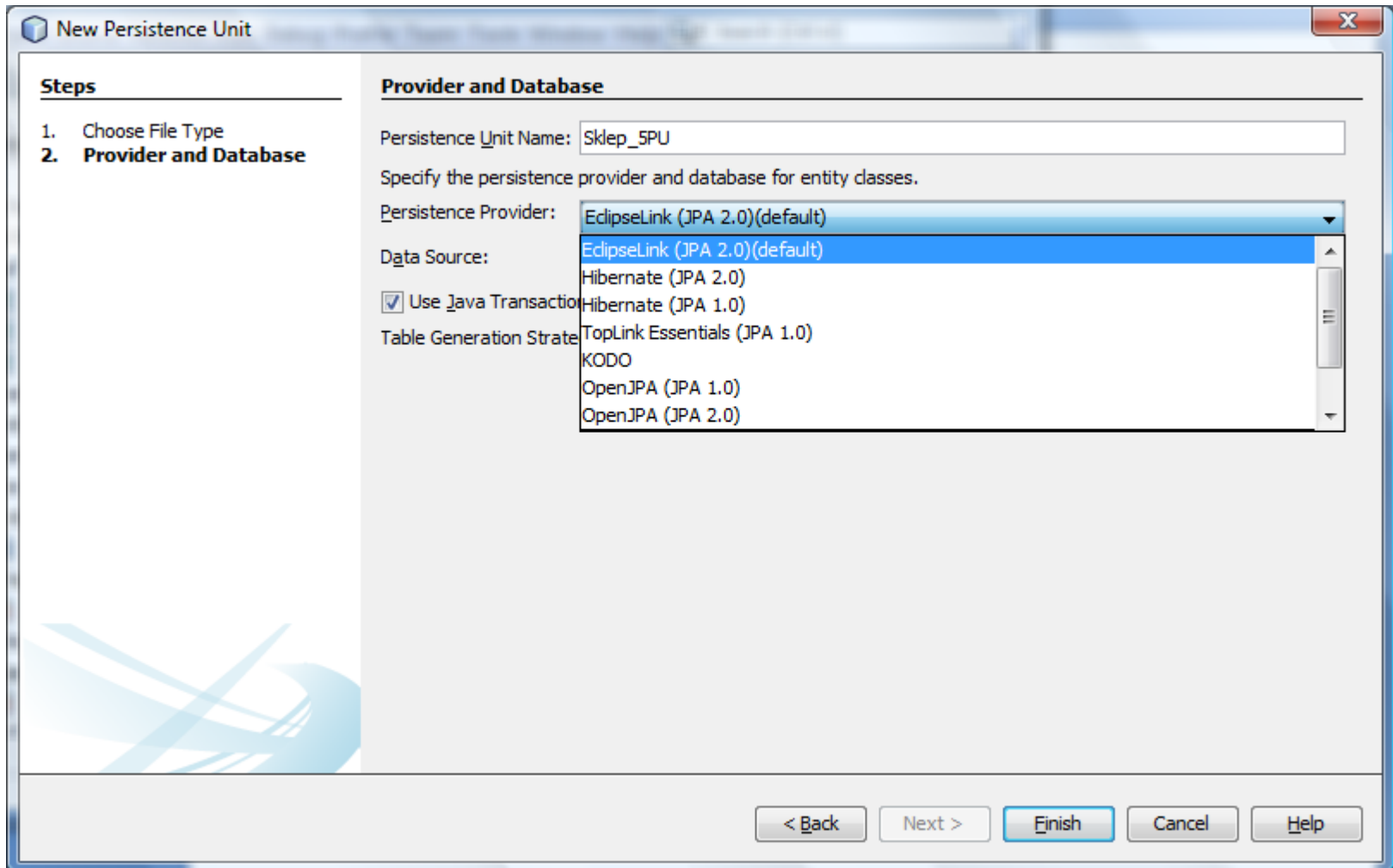
6.5. Dodanie pliku **persistence.xml** do projektu, potrzebnego w procesie utrwalania danych metodą ORM - **New/Other**

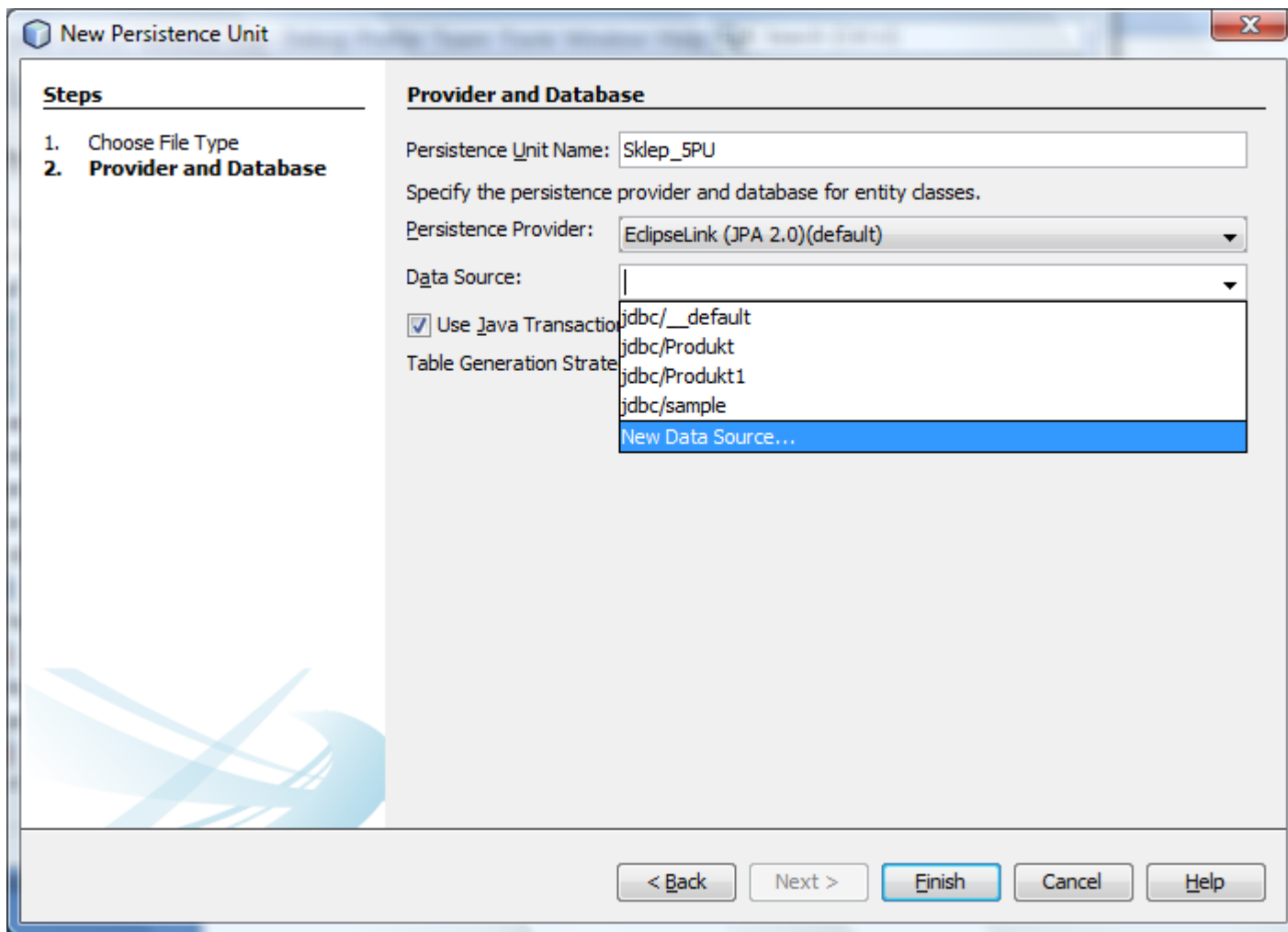


6.6. Wybór Persistence/Persistence Unit



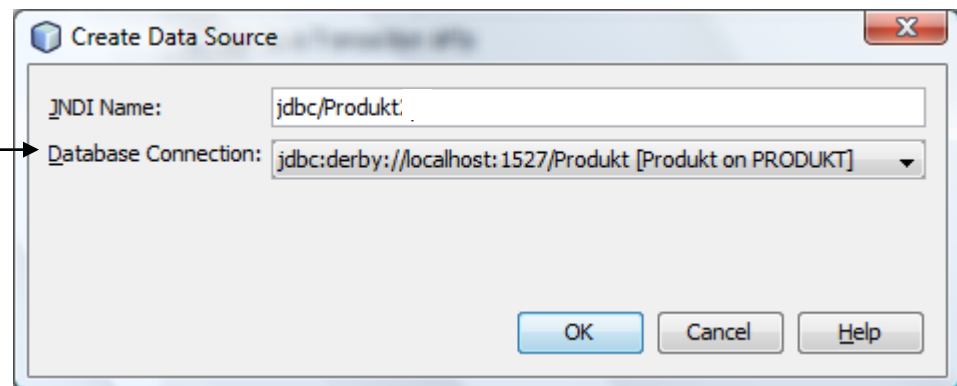
6.7. Wybór technologii EclipseLink do obsługi bazy danych metodą ORM



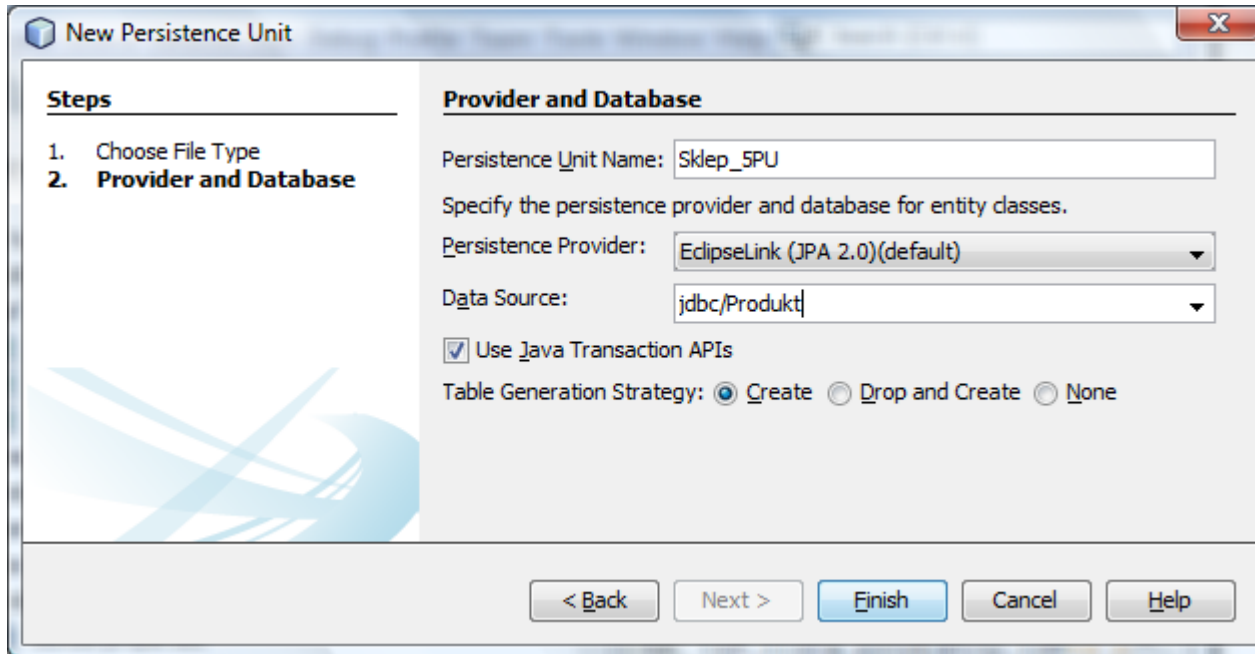


6.8.
Utworzenie
połączenia
opartego na
JNDI –
nadanie
nazwy
jdbc/Produkt

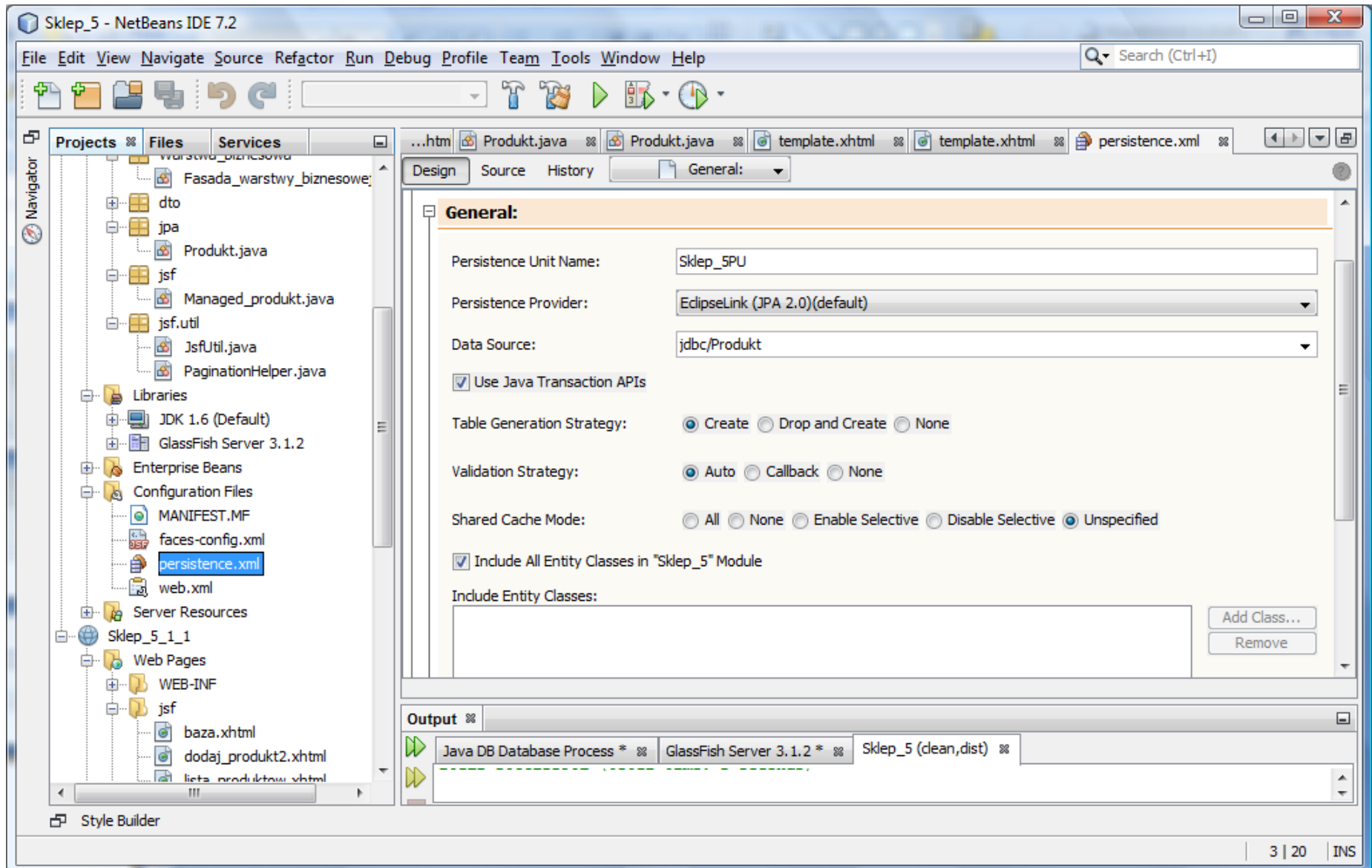
Wybór połączenia z
bazą danych



6.9. Gotowe połączenie z bazą danych - **Finish**



6.10. Pełna zawartość pliku **Persistence.xml**, pozwalająca na korzystanie z technologii ORM dla klas z adnotacją **@Entity** (tutaj klasa **Produkt**)

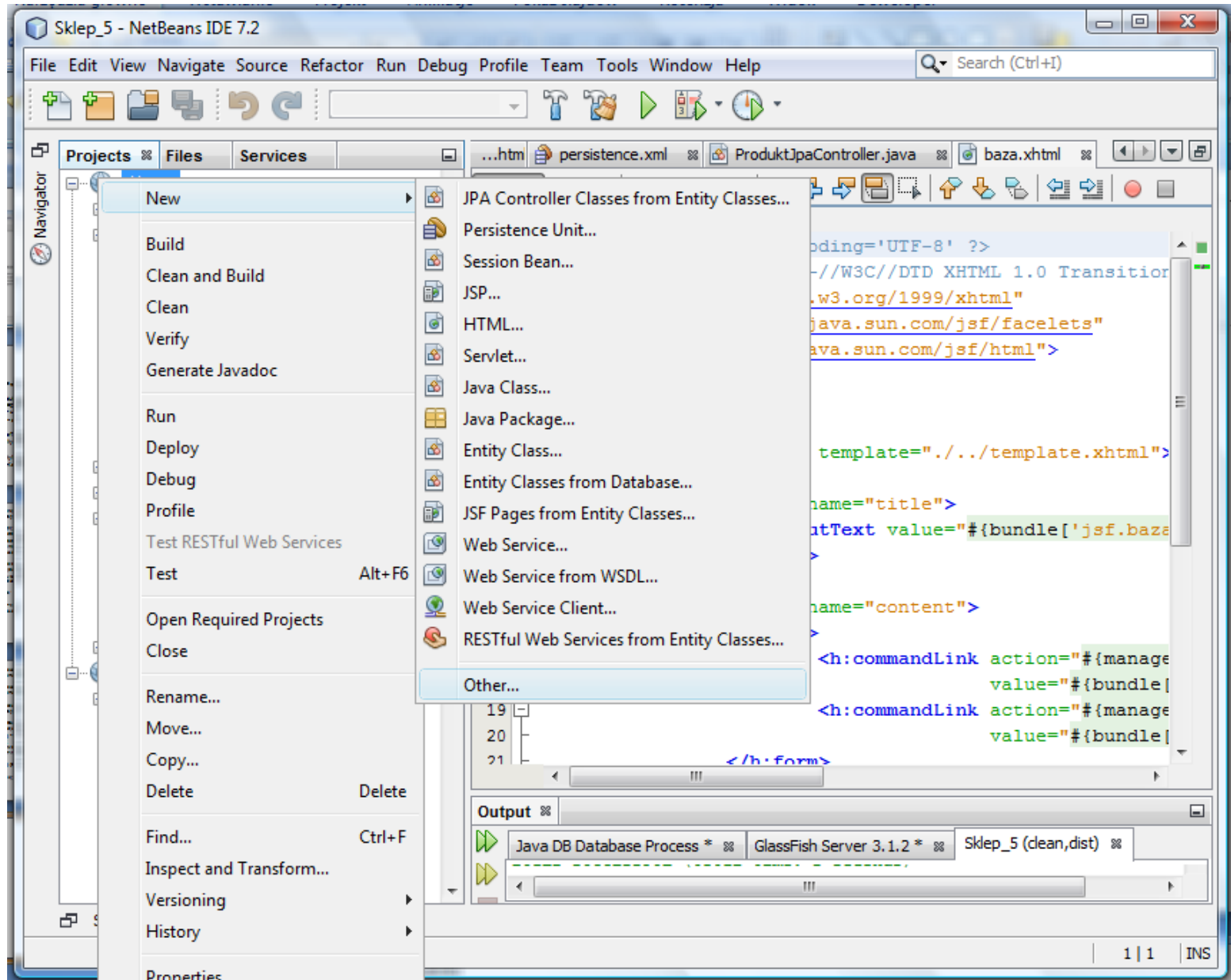


The screenshot displays the NetBeans IDE 7.2 interface. The 'Projects' pane on the left shows the project structure for 'Sklep_5_1_1', including a 'persistence.xml' file. The 'General' tab of the 'persistence.xml' file is selected, showing the following configuration:

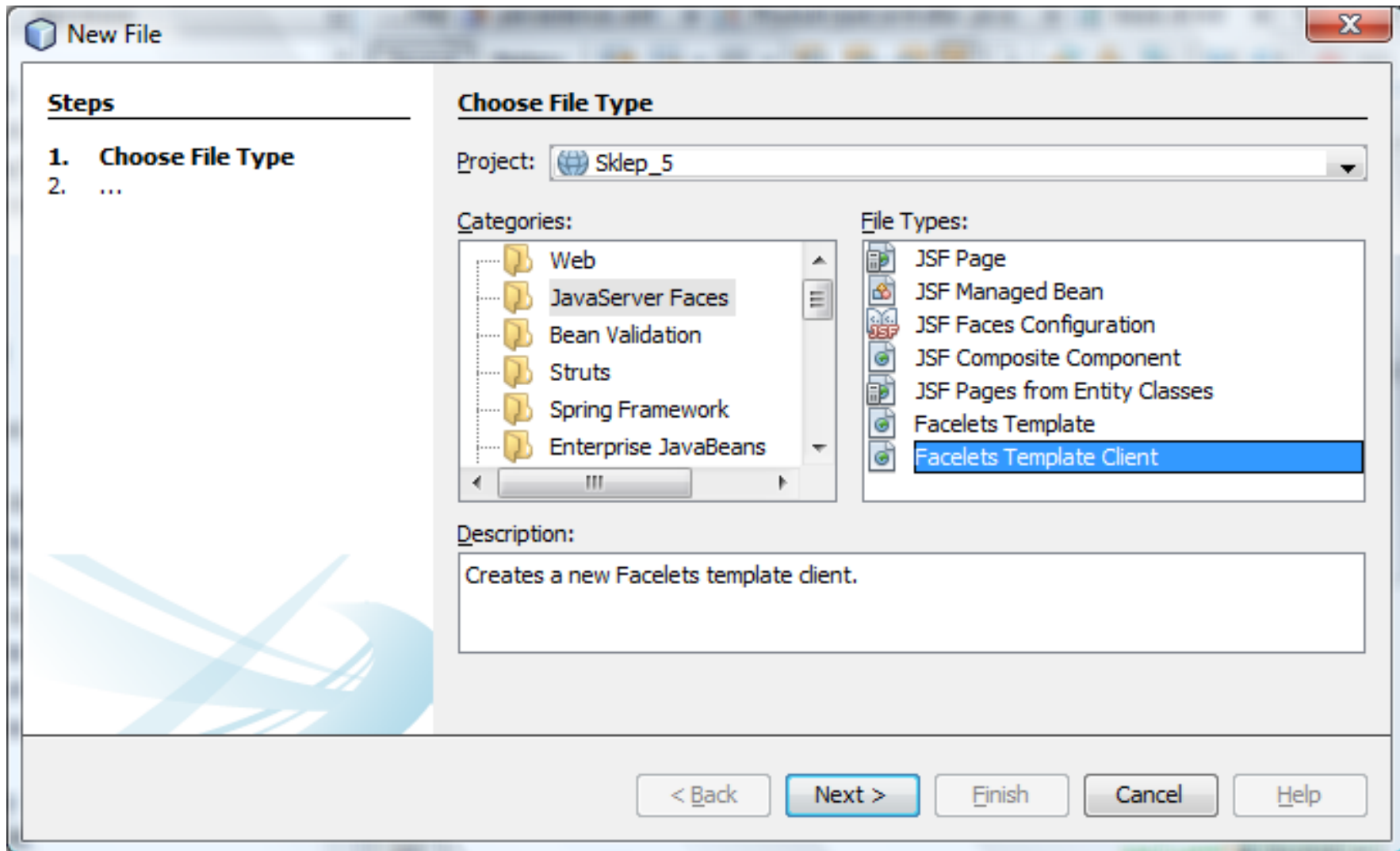
- Persistence Unit Name: Sklep_5PU
- Persistence Provider: EclipseLink (JPA 2.0)(default)
- Data Source: jdbc/Produkt
- Use Java Transaction APIs
- Table Generation Strategy: Create Drop and Create None
- Validation Strategy: Auto Callback None
- Shared Cache Mode: All None Enable Selective Disable Selective Unspecified
- Include All Entity Classes in "Sklep_5" Module
- Include Entity Classes: (Empty list)

The 'Output' window at the bottom shows the project name 'Sklep_5 (clean, dist)'.

6.11. Dodanie nowej strony do obsługi bazy danych – **New/Other**



6.12. Dodanie strony opartej na szablonie `template.xml` (1)



6.13. Dodanie strony **baza.xhtml** opartej na szablonie **template.xml** (2)

New Facelets Template Client

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

File Name:

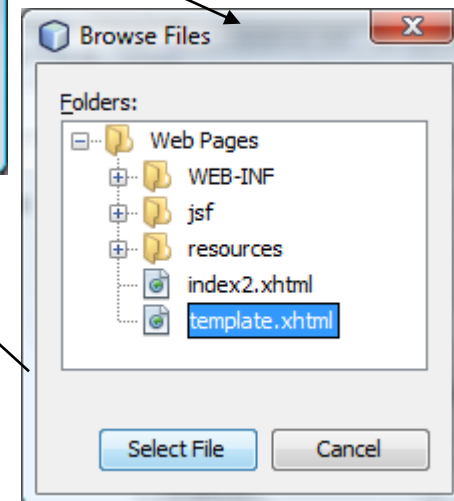
Project:

Folder:

Created File:

Template:

Generated Root Tag: <html>
 <ui:composition>



6.14. Zawartość wygenerowanego pliku [baza.xhtml](#)

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html">
<body>
    <ui:composition template=" ../template.xhtml">
        <ui:define name="title">
            <h:outputText value="#{bundle['jsf.baza.tytul']}"></h:outputText>
        </ui:define>
```

6.15. Modyfikacja zawartości strony **baza.xhtml** opartej na szablonie **template.xml**

```
<ui:define name="content">
  <h:form>
    <h:commandLink action="#{managed_produkt.zapisz}"
      value="#{bundle['jsf.baza.zapisz']}" /><br/>
    <h:commandLink action="#{managed_produkt.pobierz}"
      value="#{bundle['jsf.baza.pobierz']}" /><br/>
  </h:form>
</ui:define>
</ui:composition>
</body>
</html>
```

Pierwszy comandLink obsługuje **zapis** do bazy danych, a drugi **odczyt** z bazy danych

6.16. Uzupełnienie pliku **Bundle.properties**

jsf.baza.tytul=Utrwalanie danych

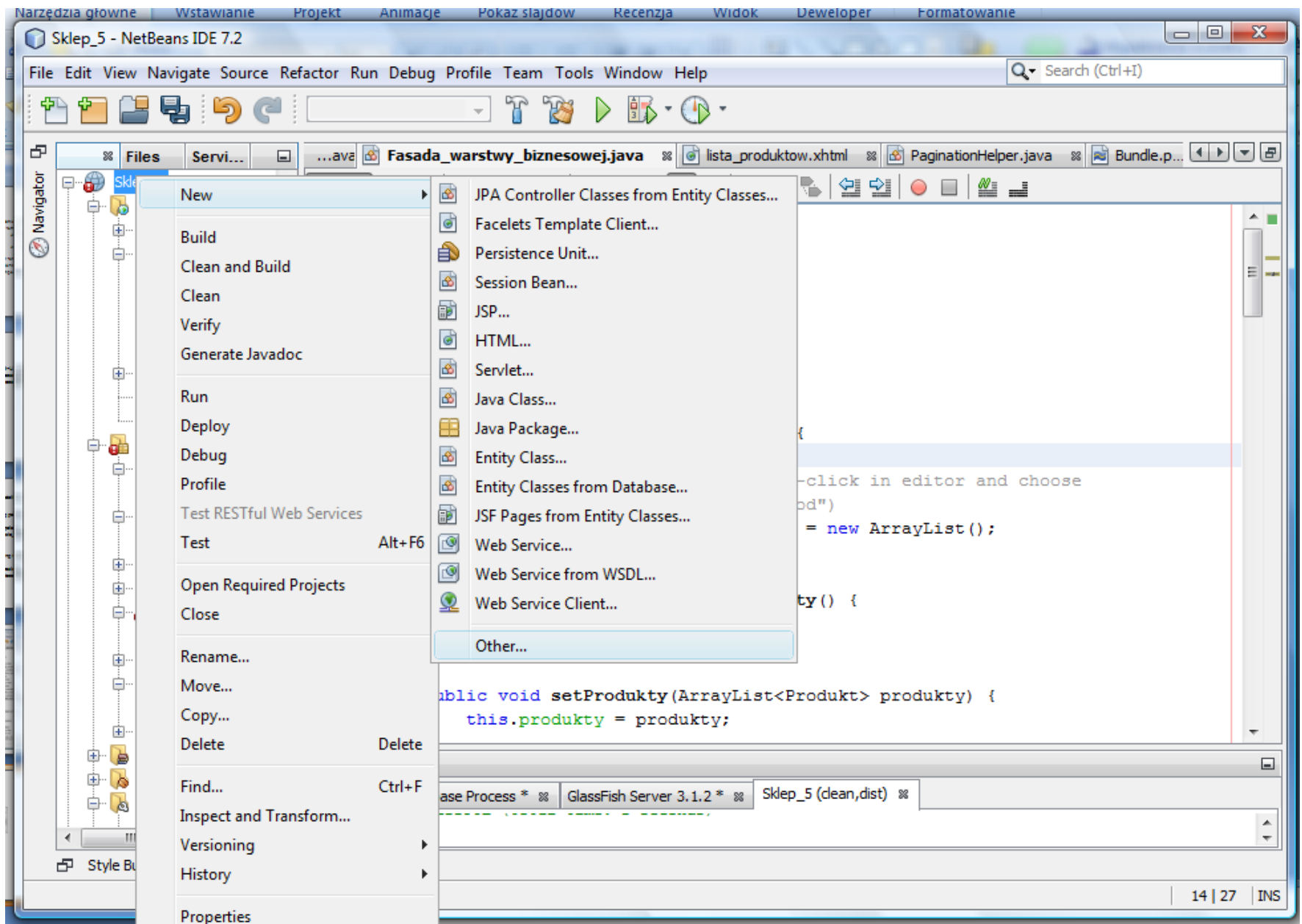
jsf.baza.zapisz=Zapisz do bazy danych

jsf.baza.pobierz=Pobierz z baza danych

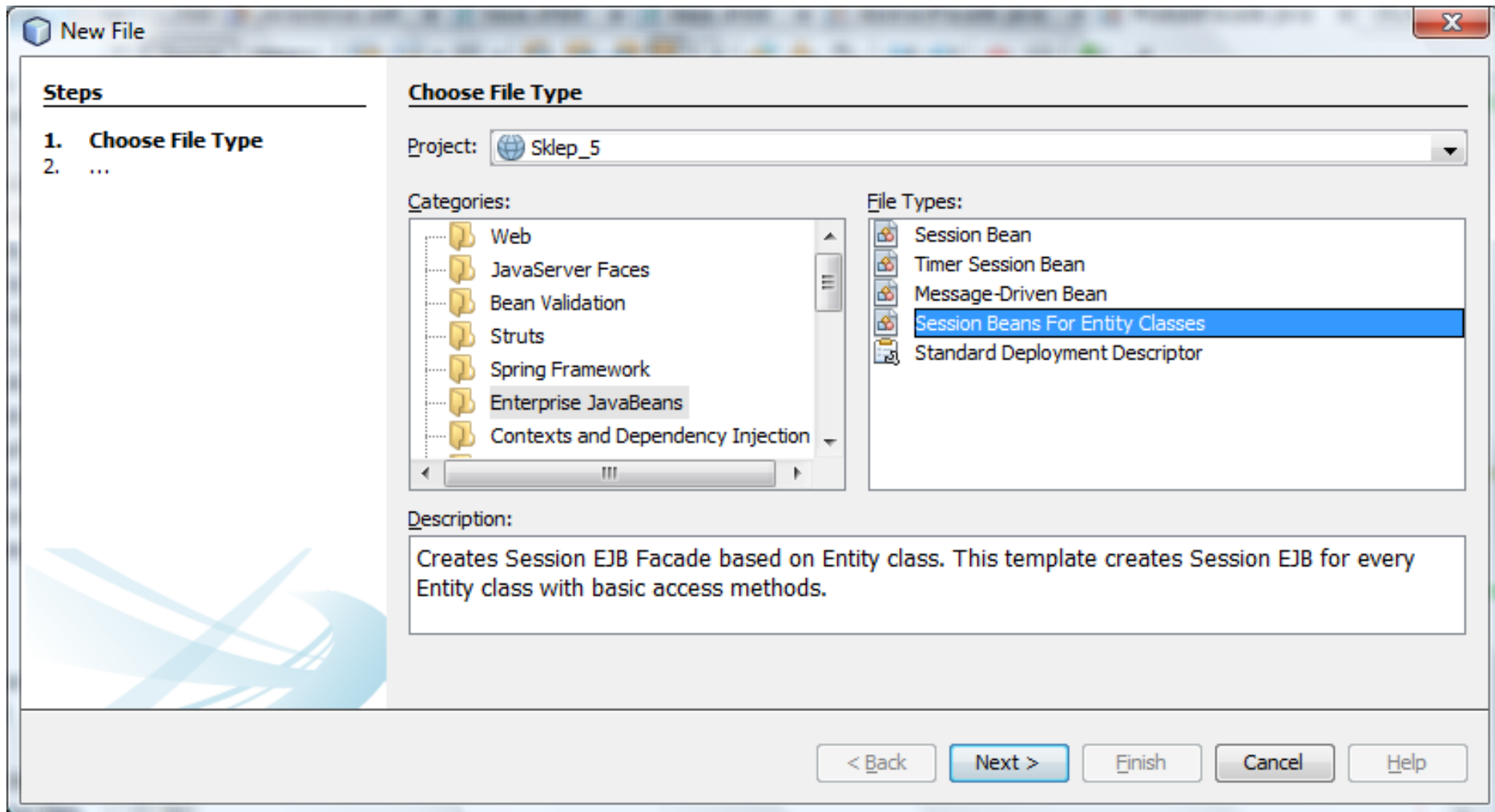
6.17. Uzupełnienie kodu pliku **template.xhtml** – bloku "left", czyli dodanie znacznika **h:link** do wywoływania strony **baza.xhtml**

```
<div id="left">  
    <h:link outcome="/faces/jsf/dodaj_produkt2"  
        value="Dodaj produkt"/> <br/>  
    <h:link outcome="/faces/jsf/lista_produktow"  
        value="Lista produktow"/><br/>  
    <h:link outcome="/faces/jsf/baza"  
        value="#{bundle['jsf.baza.tytul']}" />  
</div>
```

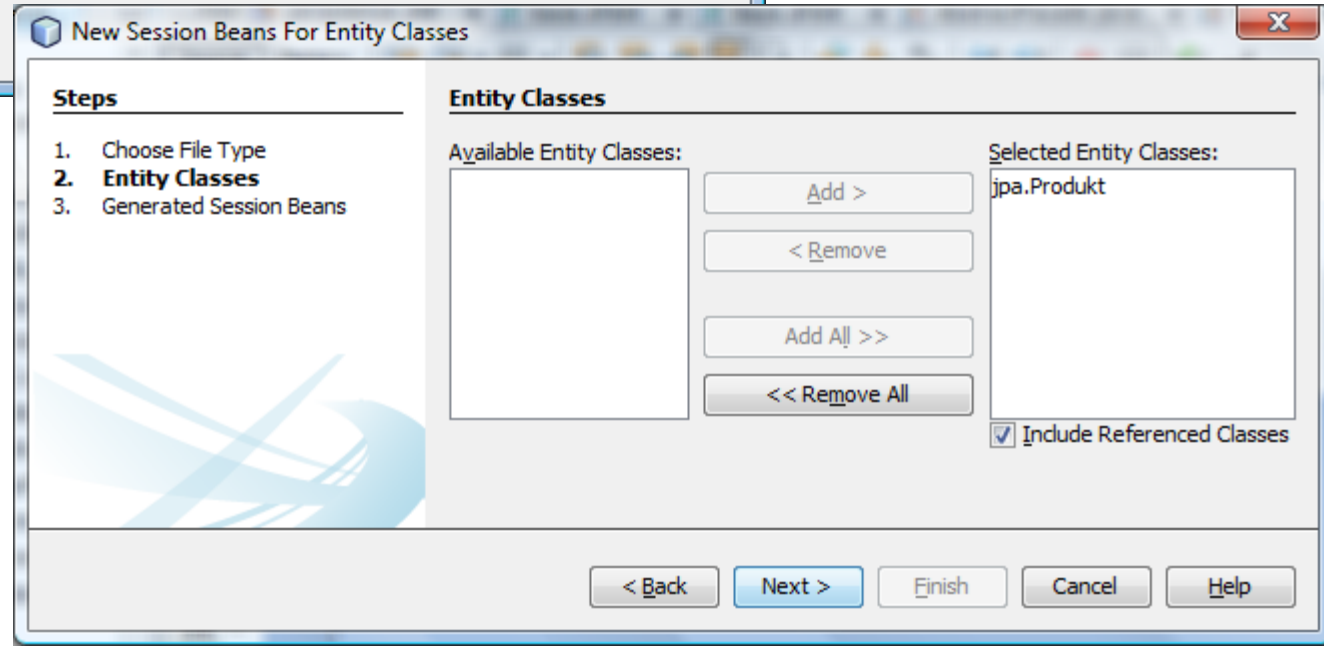
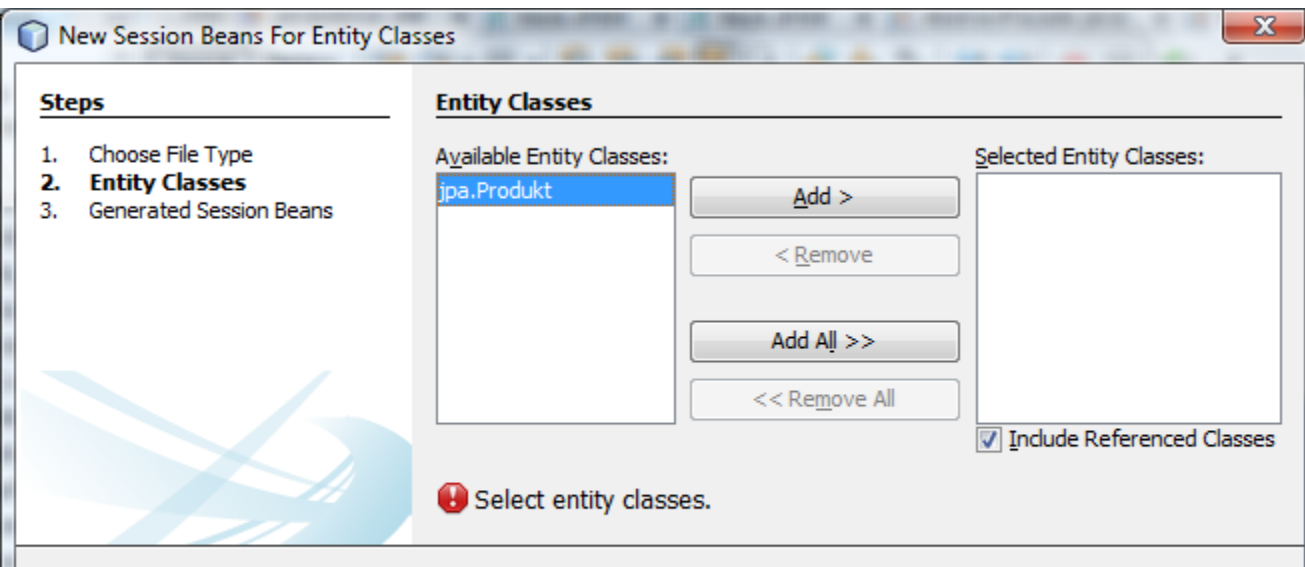
6.18. Dodanie klasy do utrwalania obiektów typu Produkt – **New/Other**



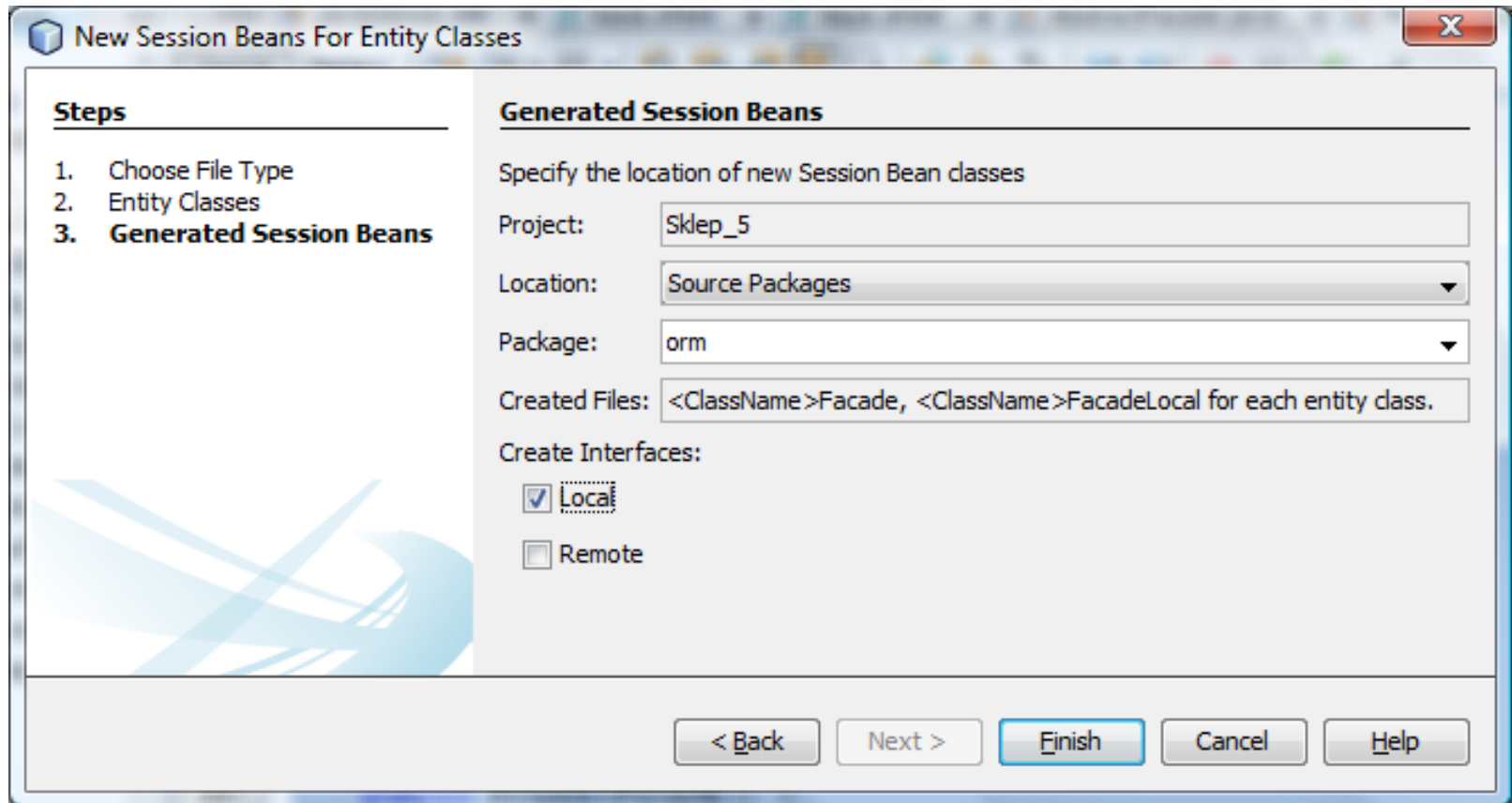
6.19. Utworzenie pliku wspierającego ORM klasy Produkt – **Enterprise JavaBeans/SessionBeans for Entity Classes**



6.20. Wskazanie klasy **Produkt** typu **Entity**



6.21. Wybór **Local** jako zasięg dla klasy typu EJB oraz wpisanie nazwy orm w polu Package



6.22. Widok wygenerowanego interfejsu **ProduktFacadeLocal**

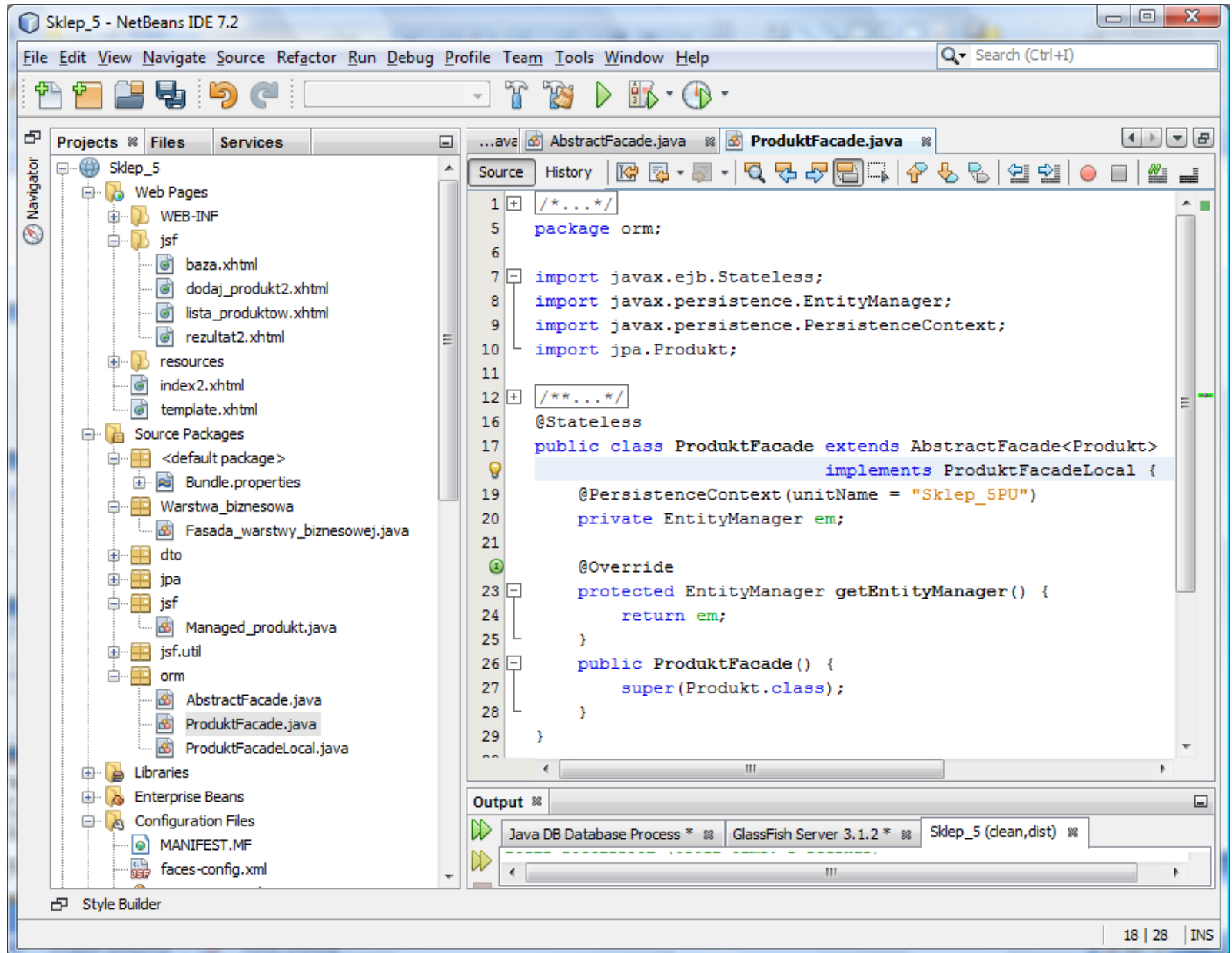
The screenshot displays the NetBeans IDE 7.2 interface. On the left, the Project Navigator shows a project named 'Sklep_5' with a package structure including 'Web Pages', 'WEB-INF', 'jsf', 'resources', 'Source Packages', 'dto', 'jpa', 'jsf', 'orm', and 'Libraries'. The 'orm' package contains the 'ProduktFacadeLocal.java' file, which is selected in the editor.

The main editor window shows the source code of the `ProduktFacadeLocal` interface. The code is as follows:

```
1  /*...*/
5  package orm;
6
7  import java.util.List;
8  import javax.ejb.Local;
9  import jpa.Produkt;
10
11 /*...*/
12 @Local
13
14 public interface ProduktFacadeLocal {
15     void create(Produkt produkt);
16     void edit(Produkt produkt);
17     void remove(Produkt produkt);
18     Produkt find(Object id);
19     List<Produkt> findAll();
20     List<Produkt> findRange(int[] range);
21     int count();
22 }
23
24
25
```

The `int count();` line is highlighted in blue. Below the editor, the Output window shows several tabs: 'Java DB Database Process *', 'GlassFish Server 3.1.2 *', and 'Sklep_5 (clean)'. The status bar at the bottom right indicates '23 | 17 | INS'.

6.23. Widok kontrolera ORM **ProduktFacade**, który implementuje interfejs **ProduktFacadeLocal** i dziedziczy po klasie abstrakcyjnej **AbstractFacade**



The screenshot displays the NetBeans IDE interface for the 'Sklep_5' project. The left sidebar shows the project structure, including source packages like 'orm'. The main editor window shows the code for 'ProduktFacade.java', which implements the 'ProduktFacadeLocal' interface by extending the 'AbstractFacade' class. The code includes package declarations, imports for 'javax.ejb.Stateless', 'javax.persistence.EntityManager', 'javax.persistence.PersistenceContext', and 'jpa.Produkt'. The class is annotated with '@Stateless' and has a constructor that injects an 'EntityManager'. It also overrides the 'getEntityManager()' method to return the injected 'EntityManager'.

```
1  /*...*/
5  package orm;
6
7  import javax.ejb.Stateless;
8  import javax.persistence.EntityManager;
9  import javax.persistence.PersistenceContext;
10 import jpa.Produkt;
11
12 /*...*/
16 @Stateless
17 public class ProduktFacade extends AbstractFacade<Produkt>
18     implements ProduktFacadeLocal {
19     @PersistenceContext(unitName = "Sklep_5PU")
20     private EntityManager em;
21
22     @Override
23     protected EntityManager getEntityManager() {
24         return em;
25     }
26     public ProduktFacade() {
27         super(Produkt.class);
28     }
29 }
```

The Output window at the bottom shows the status of the Java DB Database Process, GlassFish Server 3.1.2, and Sklep_5 (clean,dist).

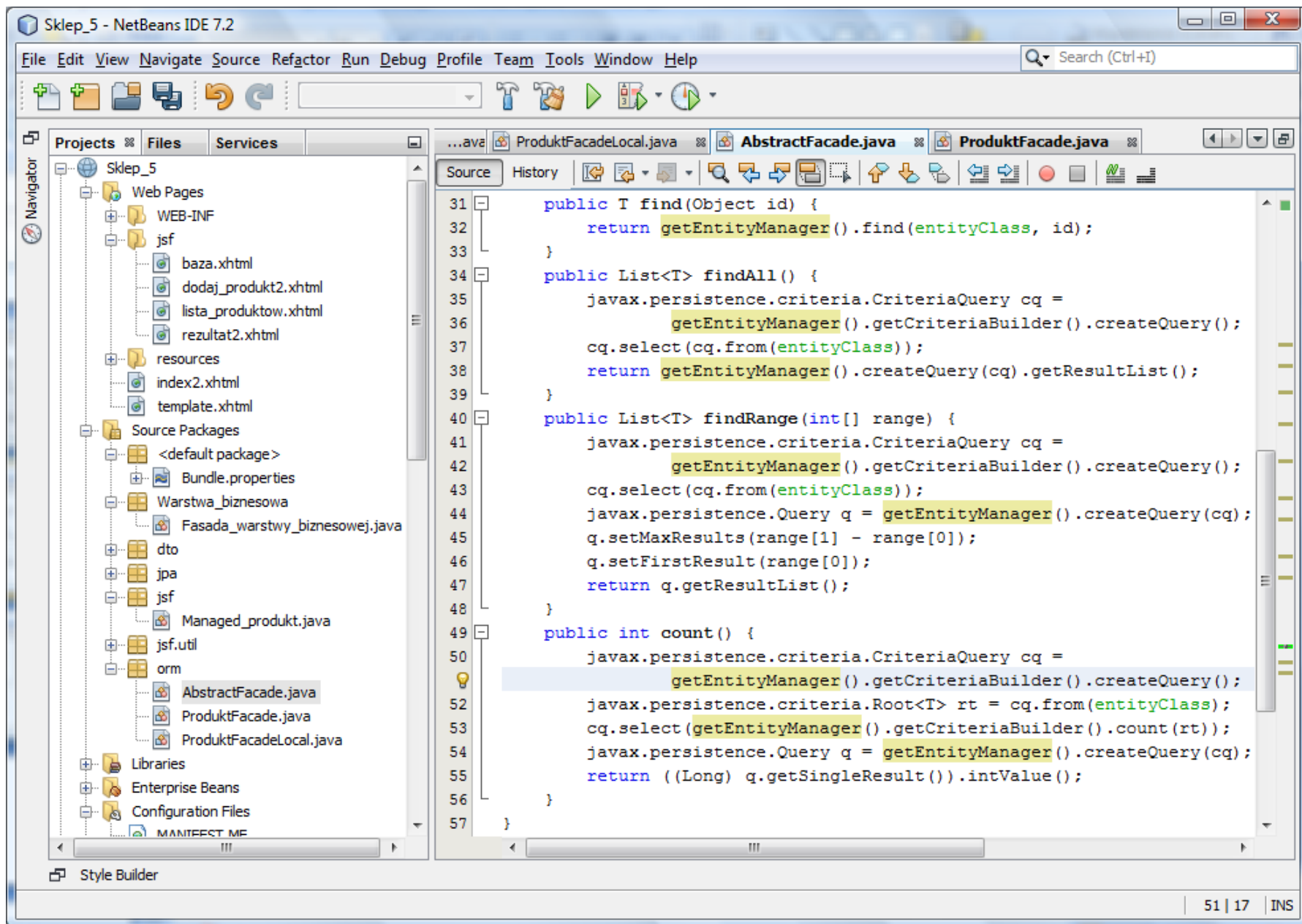
6.24. Wygenerowana klasa abstrakcyjna **AbstractFacade** (1)

The screenshot displays the NetBeans IDE interface for a project named "Sklep_5". The left-hand "Navigator" pane shows a project structure with a package named "orm" containing the file "AbstractFacade.java". The main editor window shows the source code of this class. The code is as follows:

```
1  /*...*/
5  package orm;
6
7  import java.util.List;
8  import javax.persistence.EntityManager;
9
10 /*...*/
11 public abstract class AbstractFacade<T> {
12     private Class<T> entityClass;
13
14     public AbstractFacade(Class<T> entityClass) {
15         this.entityClass = entityClass;
16     }
17     protected abstract EntityManager getEntityManager();
18     public void create(T entity) {
19         getEntityManager().persist(entity);
20     }
21     public void edit(T entity) {
22         getEntityManager().merge(entity);
23     }
24     public void remove(T entity) {
25         getEntityManager().remove(getEntityManager().merge(entity));
26     }
27 }
28
29
30
```

The "Output" window at the bottom shows the status of the IDE, including "Java DB Database Process", "GlassFish Server 3.1.2", and "Sklep_5 (clean, dist)".

6.25. Wygenerowana klasa abstrakcyjna **AbstractFacade** (2)



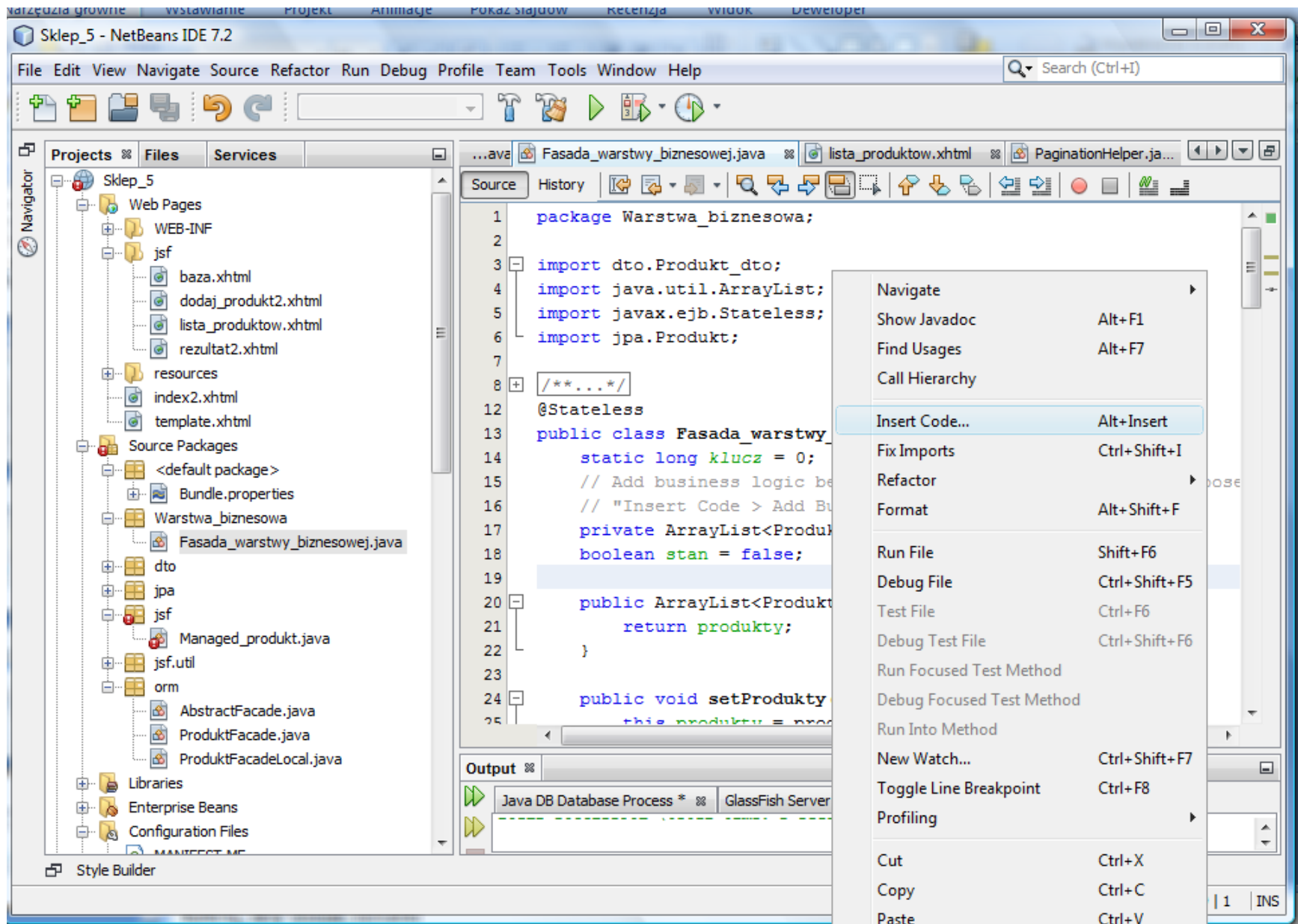
The screenshot displays the NetBeans IDE interface for a project named "Sklep_5". The left-hand "Navigator" pane shows the project structure, including source packages like "AbstractFacade.java", "ProduktFacade.java", and "ProduktFacadeLocal.java". The main editor window shows the source code of "AbstractFacade.java".

```
31 public T find(Object id) {
32     return getEntityManager().find(entityClass, id);
33 }
34 public List<T> findAll() {
35     javax.persistence.criteria.CriteriaQuery cq =
36         getEntityManager().getCriteriaBuilder().createQuery();
37     cq.select(cq.from(entityClass));
38     return getEntityManager().createQuery(cq).getResultList();
39 }
40 public List<T> findRange(int[] range) {
41     javax.persistence.criteria.CriteriaQuery cq =
42         getEntityManager().getCriteriaBuilder().createQuery();
43     cq.select(cq.from(entityClass));
44     javax.persistence.Query q = getEntityManager().createQuery(cq);
45     q.setMaxResults(range[1] - range[0]);
46     q.setFirstResult(range[0]);
47     return q.getResultList();
48 }
49 public int count() {
50     javax.persistence.criteria.CriteriaQuery cq =
51         getEntityManager().getCriteriaBuilder().createQuery();
52     javax.persistence.criteria.Root<T> rt = cq.from(entityClass);
53     cq.select(getEntityManager().getCriteriaBuilder().count(rt));
54     javax.persistence.Query q = getEntityManager().createQuery(cq);
55     return ((Long) q.getSingleResult()).intValue();
56 }
57 }
```

The code defines an abstract facade interface with four methods: `find`, `findAll`, `findRange`, and `count`. Each method delegates the call to `getEntityManager()` and uses JPA (Java Persistence API) queries to interact with the database. The `count` method returns the number of results as an integer.

At the bottom right of the IDE window, the page number "51 | 17" and the text "INS" are visible.

6.26. Utworzenie referencji do ziarna **ProduktFacade** w klasie **Fasada_warstwy_biznesowej** (1)



The screenshot shows the NetBeans IDE 7.2 interface. The main editor displays the source code of the `Fasada_warstwy_biznesowej` class. The code includes package declarations, imports for `dto.Produkt_dto`, `java.util.ArrayList`, `javax.ejb.Stateless`, and `jpa.Produkt`. The class is annotated with `@Stateless` and contains a `private ArrayList<Produkt> produkty` field and a `public void setProdukty` method. A context menu is open over the class name, listing various actions such as "Navigate", "Show Javadoc", "Find Usages", "Call Hierarchy", "Insert Code...", "Fix Imports", "Refactor", "Format", "Run File", "Debug File", "Test File", "Debug Test File", "Run Focused Test Method", "Debug Focused Test Method", "Run Into Method", "New Watch...", "Toggle Line Breakpoint", "Profiling", "Cut", "Copy", and "Paste". The "Insert Code..." option is highlighted.

```
1 package Warstwa_biznesowa;
2
3 import dto.Produkt_dto;
4 import java.util.ArrayList;
5 import javax.ejb.Stateless;
6 import jpa.Produkt;
7
8 /**...*/
9
10 @Stateless
11
12 public class Fasada_warstwy_biznesowej {
13     static long klucz = 0;
14     // Add business logic below
15     // "Insert Code > Add Business Logic"
16     private ArrayList<Produkt> produkty;
17     boolean stan = false;
18
19
20     public ArrayList<Produkt> getProdukty() {
21         return produkty;
22     }
23
24     public void setProdukty(Produkt[] produkty) {
25         this.produkty = new ArrayList<Produkt>(produkty.length);
26         for (Produkt produkt : produkty) {
27             this.produkty.add(produkt);
28         }
29     }
30 }
```


6.27. Utworzenie referencji do ziarna **ProduktFacade** w klasie **Fasada_warstwy_biznesowej** (2) – **Insert Code/Call Enterprise Bean** i wybór z listy ziarna typu **ProduktFacade** – pojawiła się referencja typu **ProduktFacadeLocal**

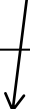
The screenshot displays the NetBeans IDE 7.2 interface. The main editor shows the source code of the class `Fasada_warstwy_biznesowej.java`. The code includes the following imports and class definition:

```
1 package Warstwa_biznesowa;
2
3 import dto.Produkt_dto;
4 import java.util.ArrayList;
5 import javax.ejb.EJB;
6 import javax.ejb.Stateless;
7 import jpa.Produkt;
8 import orm.ProduktFacadeLocal;
9
10 /**...*/
14 @Stateless
15 public class Fasada_warstwy_biznesowej {
16     @EJB
17     private ProduktFacadeLocal produktFacade;
18     boolean orm = false;
19
20     static long klucz = 0;
21     // Add business logic below. (Right-click in editor and c
22     // "Insert Code > Add Business Method")
23     private ArrayList<Produkt> produkty = new ArrayList();
24     boolean stan = false;
25 }
```

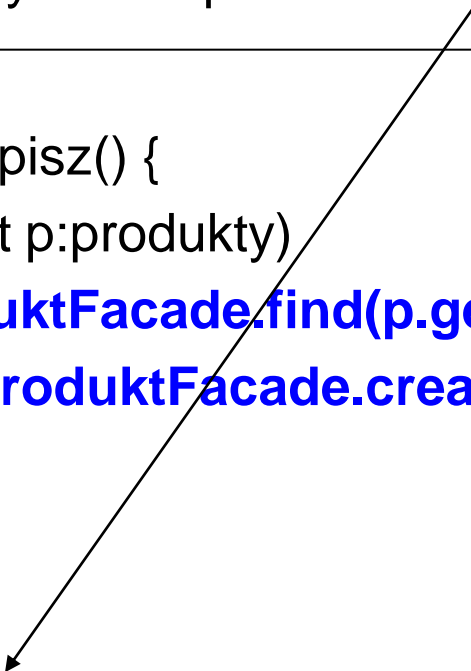
The `Call Enterprise Bean` dialog is open, showing the selection of the `ProduktFacade` bean from the `Fasada_warstwy_biznesowej` project. The `Reference Name` is set to `ProduktFacade`, and the `Referenced Interface` is set to `Local`.

The `Output` window at the bottom shows the status of the Java DB Database Process, GlassFish Server 3.1.2, and the Sklep_5 (clean,dist) process.

6.28. Metody w klasie **Fasada_wartwy_biznesowej** do utrwalania danych i do pobierania danych z danych Produkt.



```
public void zapisz() {  
    for (Produkt p:produkty)  
        { if(produktFacade.find(p.getId())==null)  
            produktFacade.create(p);  
        }  
}
```



```
public void pobierz() { //bazy usunięto w dniu 26.05.2013  
    List<Produkt> pom = produktFacade.findAll();  
    produkty.clear();  
    for(Produkt p:pom)  
        produkty.add(p);  
}
```

6.29. Metody w klasie **Managed_produk**t

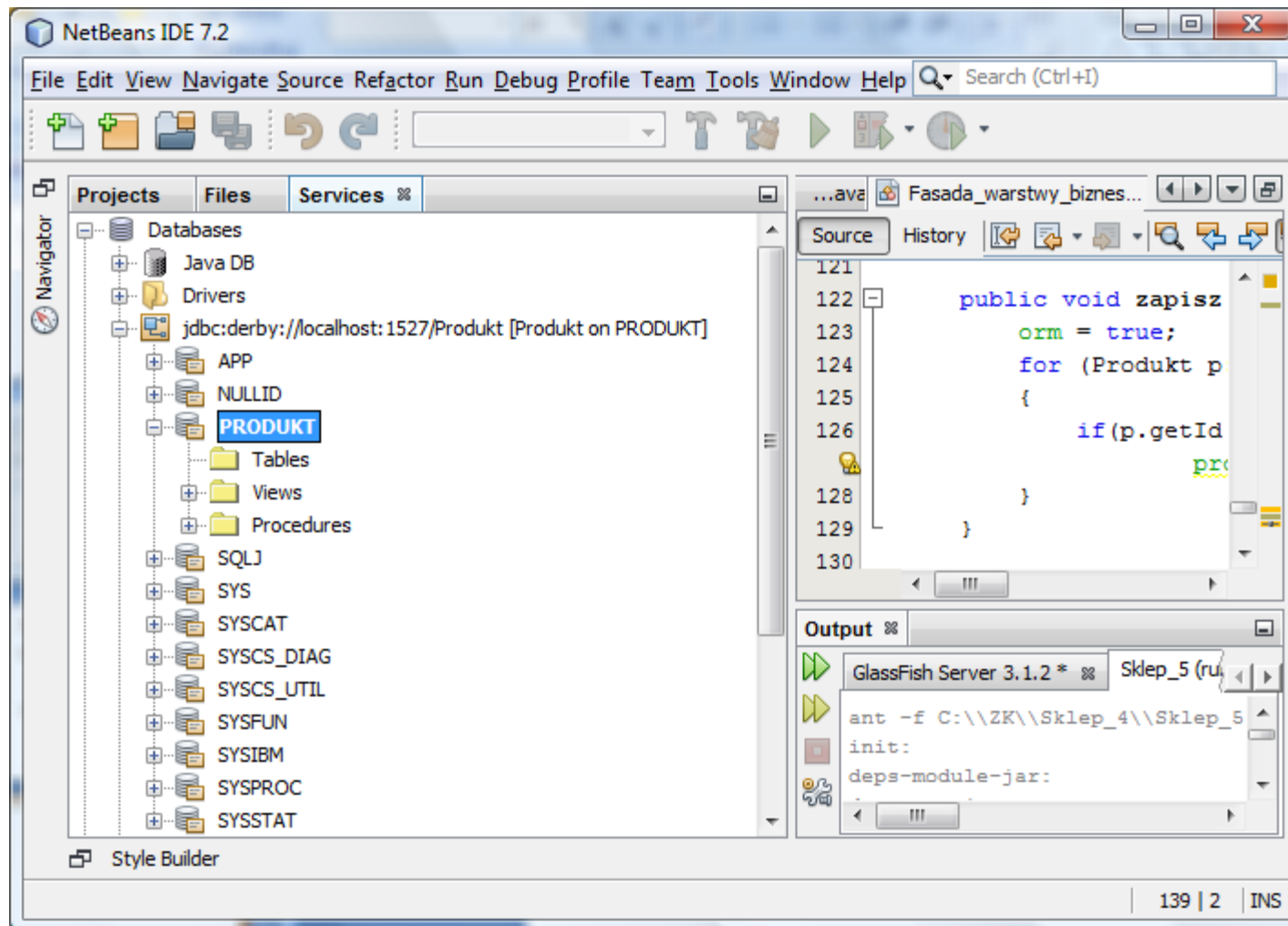
```
public String zapisz() {  
    fasada.zapisz();  
    return "/faces/index2";  
}
```

```
public String pobierz() {  
    fasada.pobierz();  
    getPagination().setPage(); //dodano w dniu 26.05.2013  
    recreateModel(); //dodano w dniu 26.05.2013  
    return "/faces/index2";  
}
```

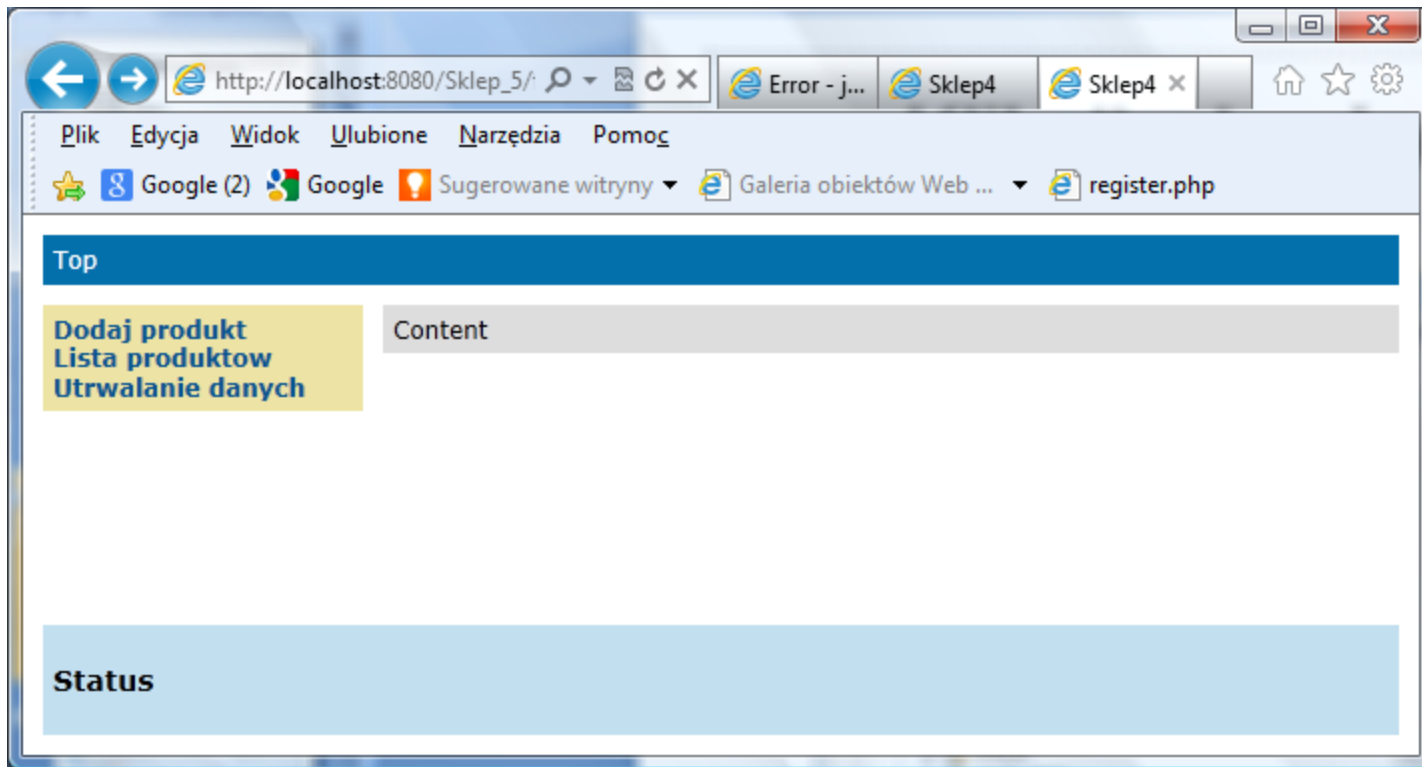
Aktualizacja liczby stron po odczytaniu danych z bazy danych

Aktualizacja modelu tabeli

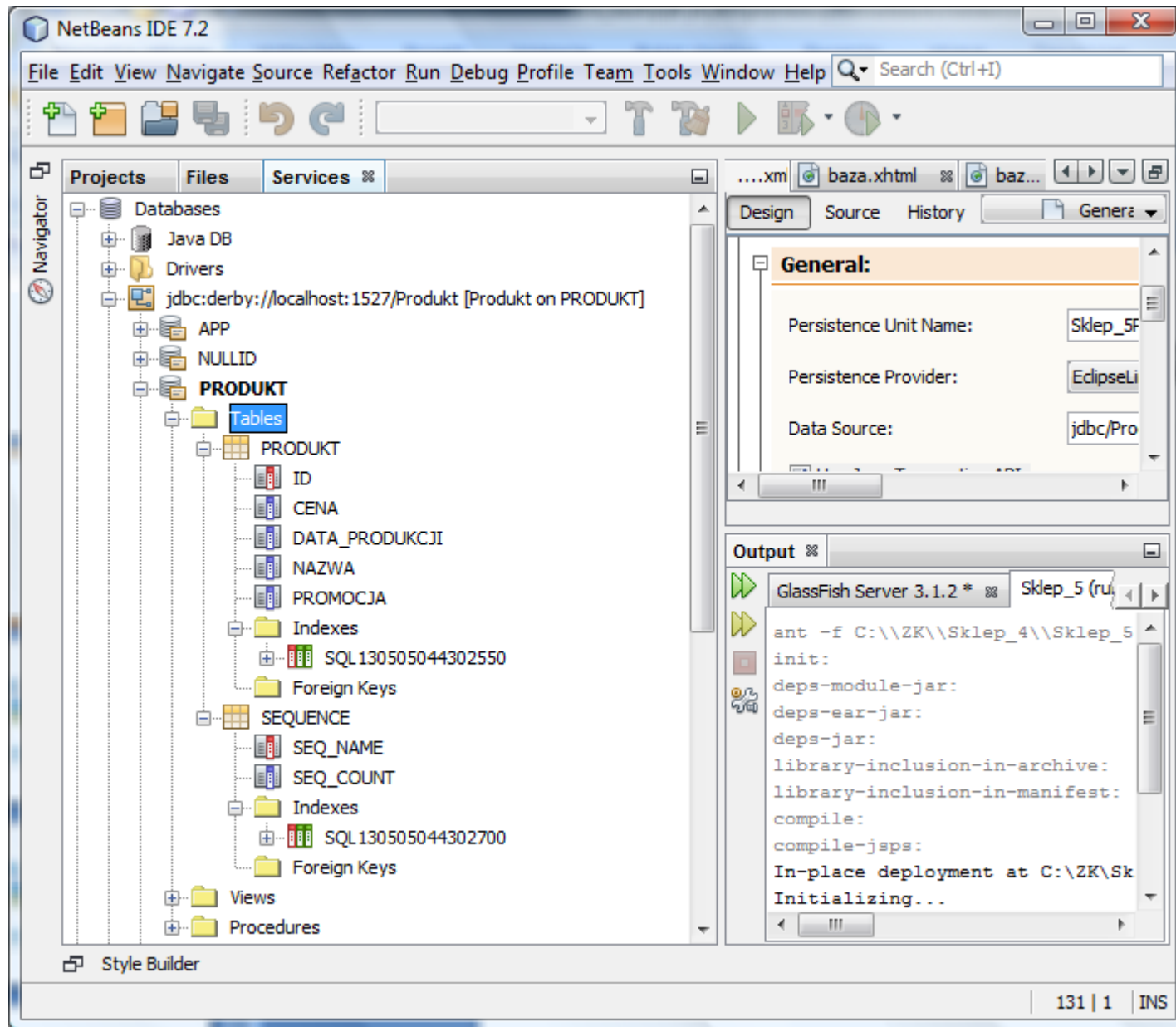
6.30 . Baza danych jest pusta przed uruchomieniem aplikacji



6.31. Uruchomienie aplikacji



6.32. Po uruchomieniu aplikacji tworzy się schemat bazy danych.



6.33. Prezentacja działania aplikacji

Top

Dodaj produkt
Lista produktów
Utrwalanie danych

1..3/4 [Następny 3](#)

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
3	Produkt2	220 zł	10 %	czwartek, 12-12-2013	198 zł	Rezultat Edycja Usun
4	Produkt1	110 zł	10 %	czwartek, 12-12-2013	99 zł	Rezultat Edycja Usun
5	Produkt3	330 zł	10 %	czwartek, 12-12-2013	297 zł	Rezultat Edycja Usun

Status

http://localhost:8080/Sklep_5/faces/jsf/lista_produkow.xhtml#

Dodaj produkt
Lista produktów
Utrwalanie danych

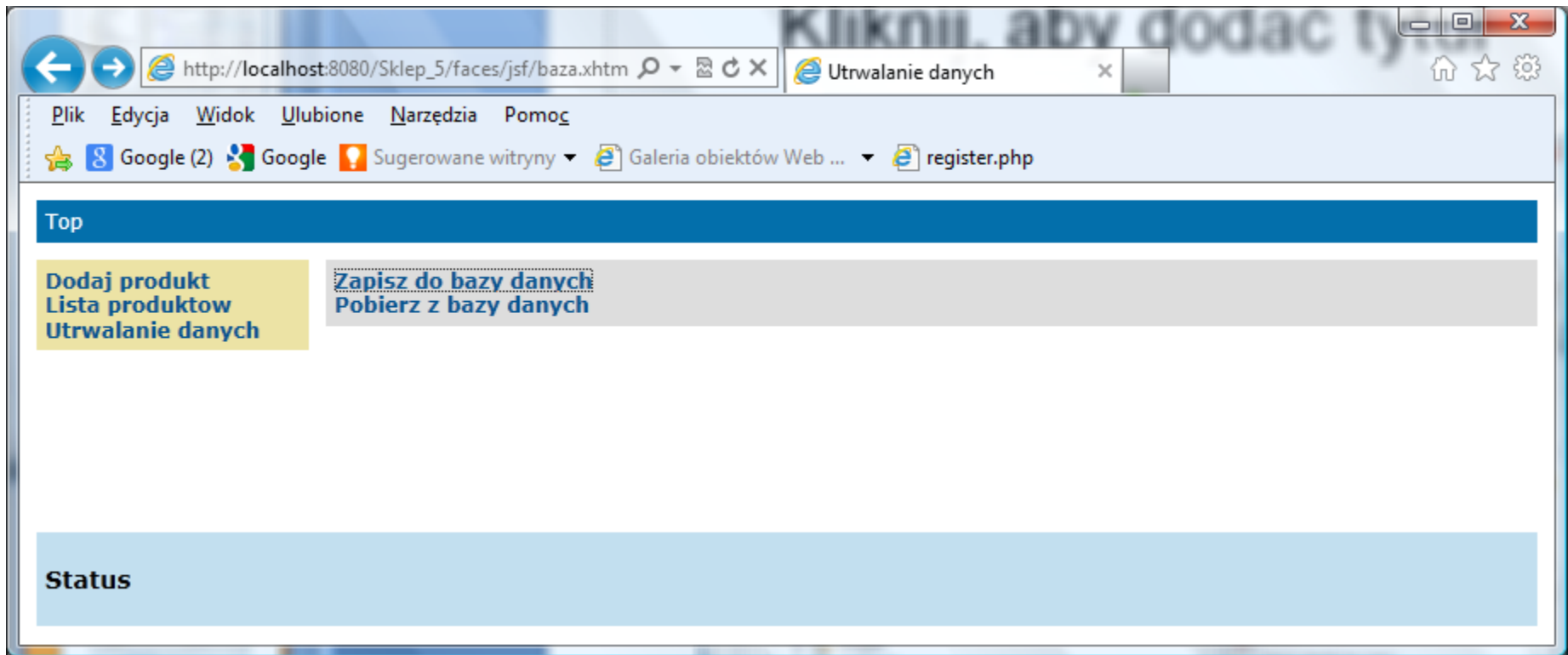
4..4/4 [Poprzedni 3](#)

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji produktu	Cena brutto produktu	
7	Produkt5	155 zł	10 %	czwartek, 12-12-2013	139,5 zł	Rezultat Edycja Usun

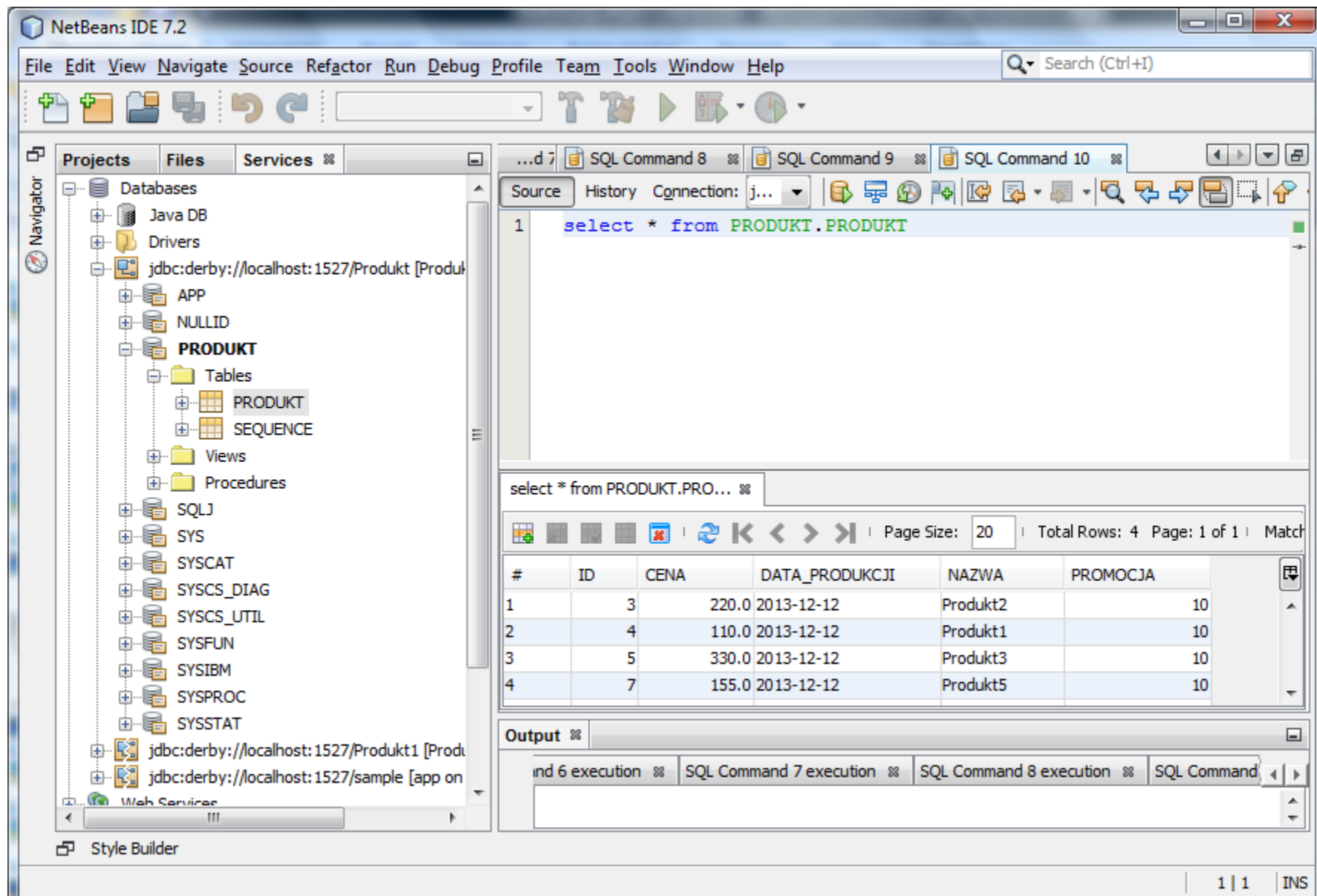
Status

http://localhost:8080/Sklep_5/faces/jsf/lista_produkow.xhtml#

6.34. Zapis do bazy danych



6.35. Prezentacja bazy danych – należy kliknąć prawym klawiszem myszy na tabelę **Produkt** i wybrać **View data**.



The screenshot shows the NetBeans IDE 7.2 interface. On the left, the Navigator pane displays a tree view of the database structure. The 'PRODUKT' table is highlighted. The main editor window shows an SQL Command window with the query: `select * from PRODUKT.PRODUKT`. Below the query, the results are displayed in a table format. The table has 6 columns: #, ID, CENA, DATA_PRODUKCJI, NAZWA, and PROMOCJA. The results show 4 rows of data.

#	ID	CENA	DATA_PRODUKCJI	NAZWA	PROMOCJA
1	3	220.0	2013-12-12	Produkt2	10
2	4	110.0	2013-12-12	Produkt1	10
3	5	330.0	2013-12-12	Produkt3	10
4	7	155.0	2013-12-12	Produkt5	10