

**Budowa warstwy klienta w architekturze typu klient-serwer zbudowanych z komponentów typu EE - klient desktopowy i internetowy. Zastosowanie komponentów opartych na technologii EJB 3.2. na podstawie**

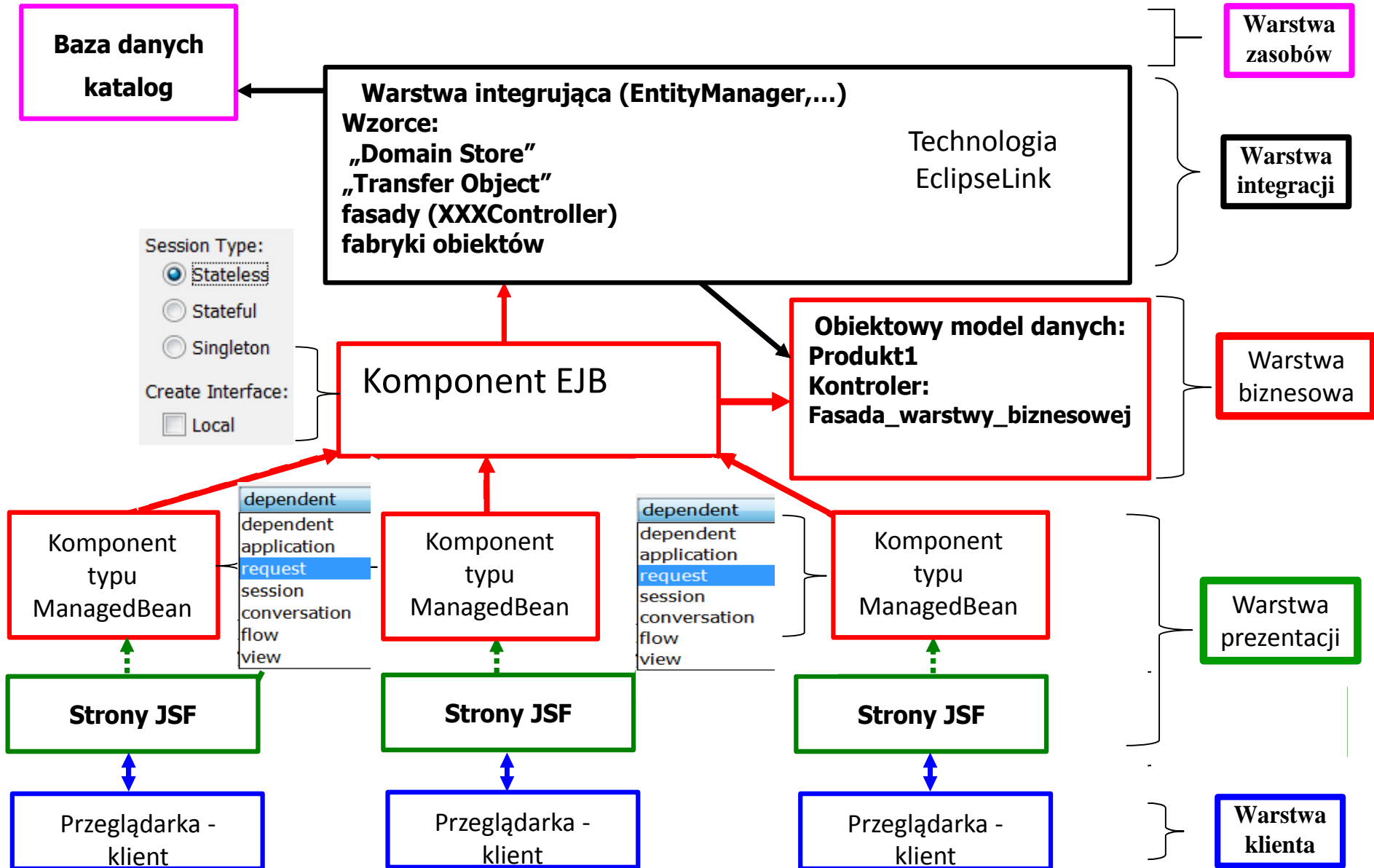
**<https://docs.oracle.com/javaee/7/JEETT.pdf>**

# **Programowanie komponentowe 5**

# 1. Aplikacja wielowarstwowa Java EE oparta na komponentach EJB typu Session Bean – rodzaj Stateless lub Singleton

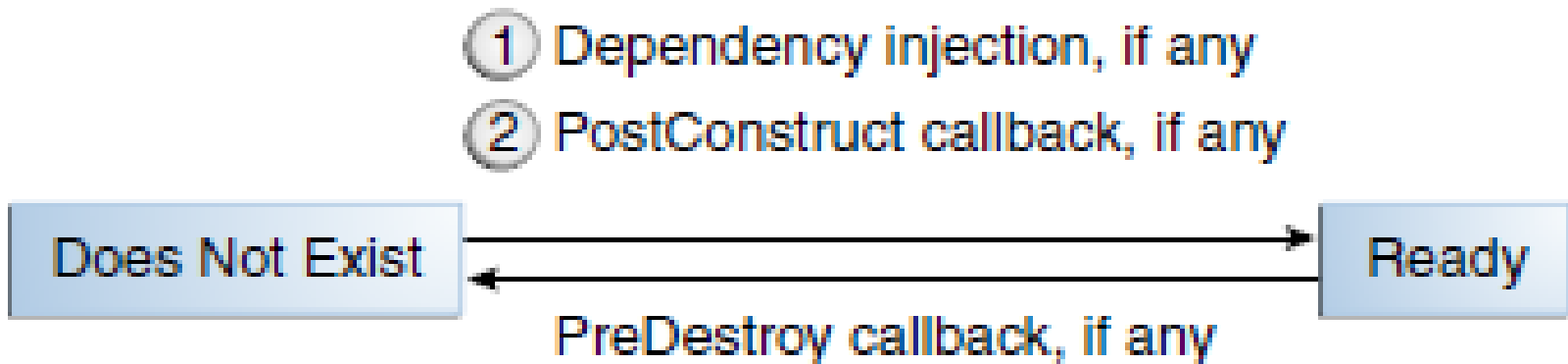
- 1.1. Warstwa internetowa oparta na komponentach Managed Bean o czasie życia **RequestScope**
- 1.2. Warstwa internetowa oparta na komponentach Managed Bean o czasie życia **SessionScope**

# Ad. 1.1. Architektura aplikacji pięciowarstwowej -Java EE 7.0 JavaServer Faces



# Cykl życia bezstanowych (Stateless) komponentów typu Session Bean – wykład 2

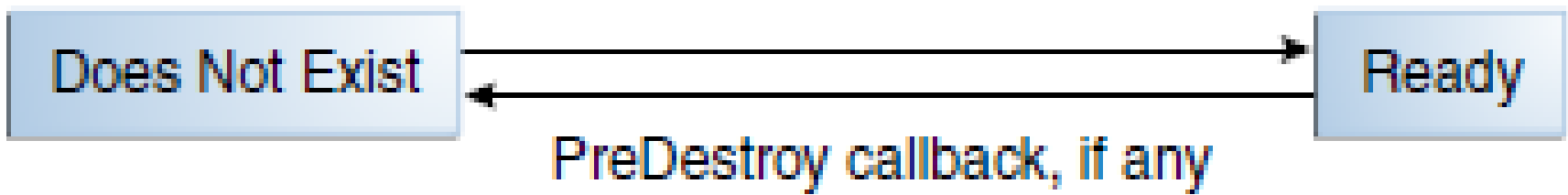
Ponieważ bezstanowy komponent typu Session Bean nigdy nie przechodzi w stan pasywny, cykl życia składa się tylko z dwóch stanów: **nonexistent and ready** podczas wywoływania metod przez warstwę klienta.



# Cykl życia komponentów Singleton typu Session Bean

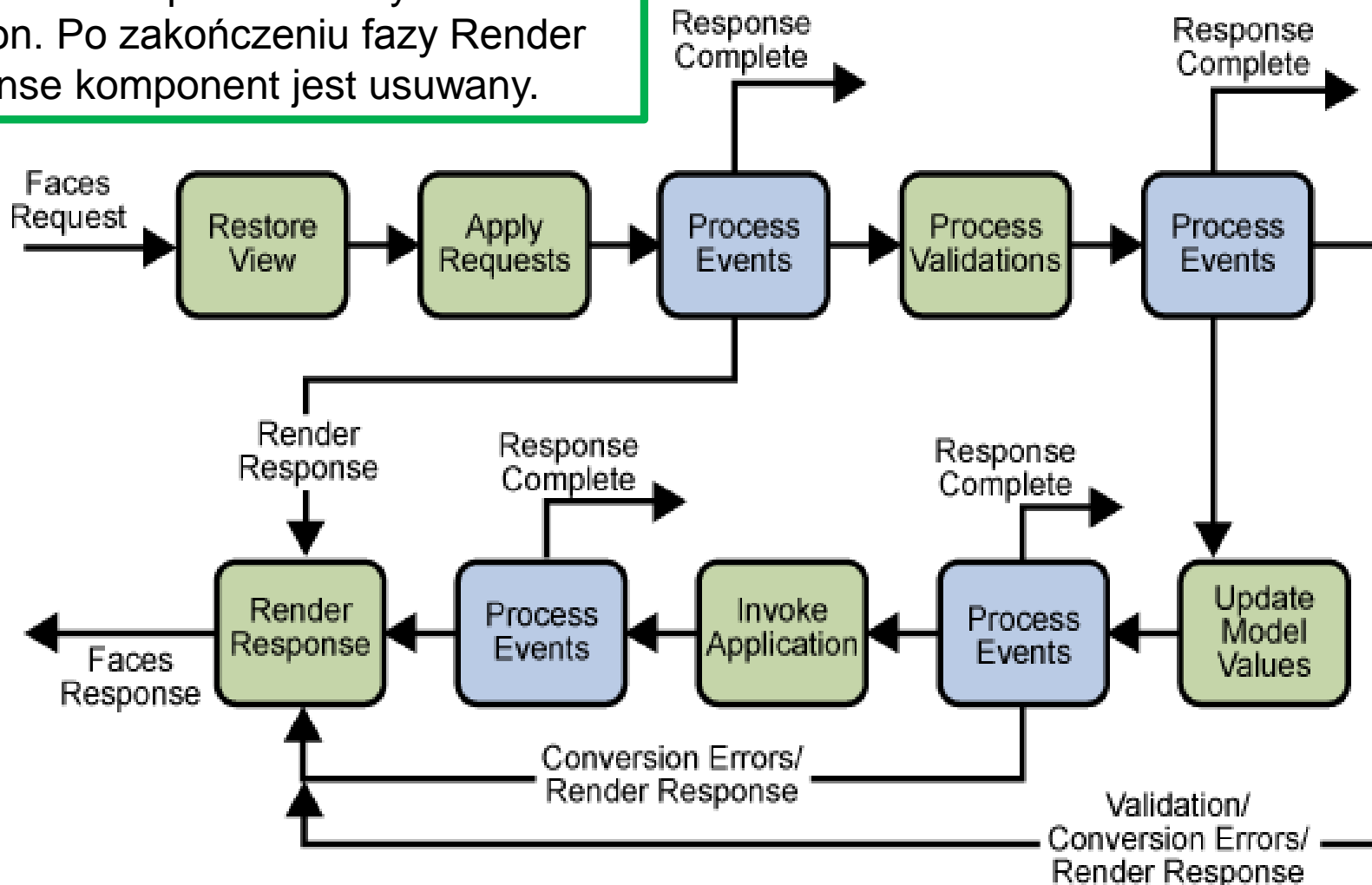
Ponieważ bezstanowy komponent typu Session Bean nigdy nie przechodzi w stan pasywny, cykl życia składa się tylko z dwóch stanów: **nonexistent and ready** podczas wywoływania metod przez warstwę klienta. Singleton startuje razem z aplikacją, jeśli ma adnotację **@Startup** i inicjuje aplikację za pomocą metody **@PostConstruct** (jeśli jest) i „czyści” za pomocą metody **@PreDestroy** (jeśli jest).

- 1 Dependency injection, if any
- 2 PostConstruct callback, if any



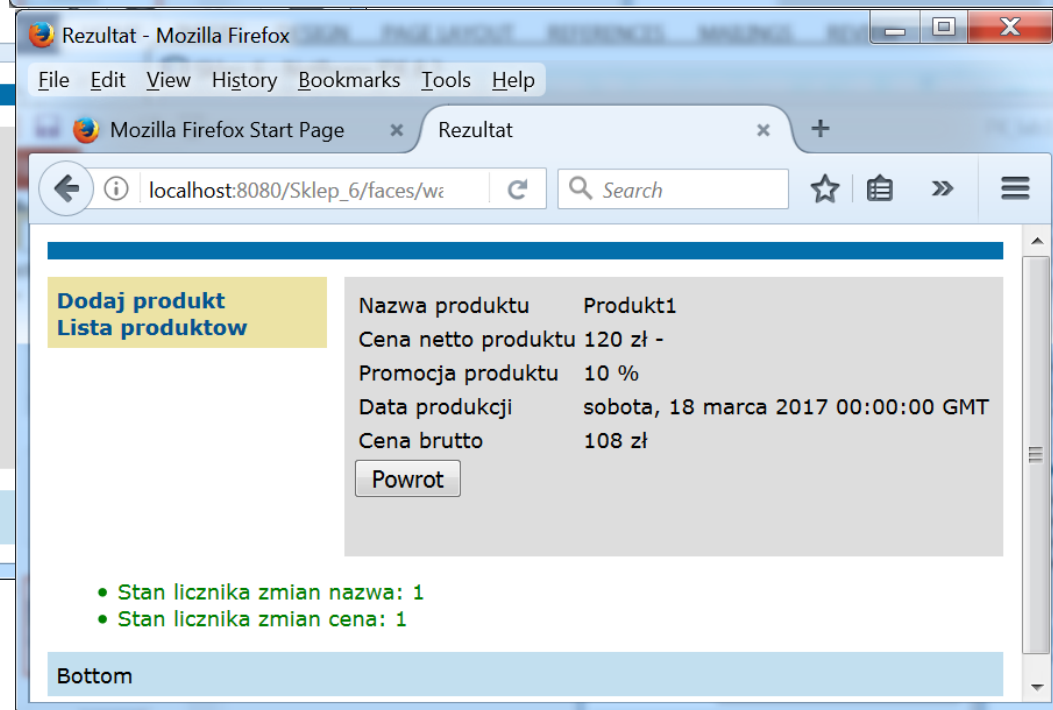
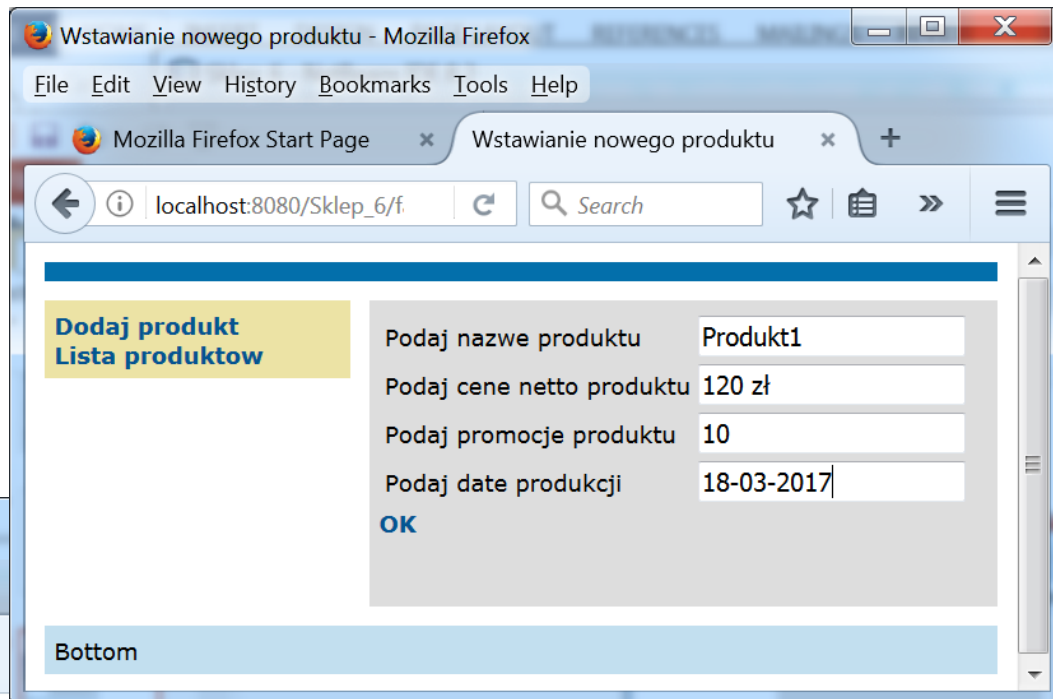
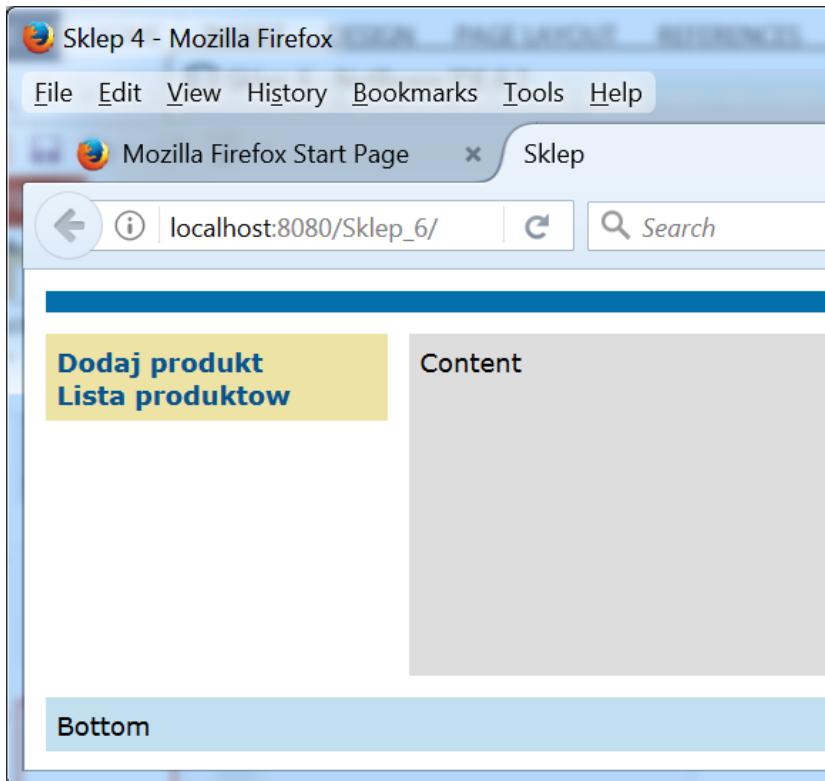
# Standard cyklu życia „Request-Response” dla JavaServer Faces

**RequestScoped:** podczas odebrania żądania tworzony jest nowy komponent typu **Managed\_produk**t – wszystkie jego dane są zaktualizowane podczas fazy Invoke Application. Po zakończeniu fazy Render Response komponent jest usuwany.



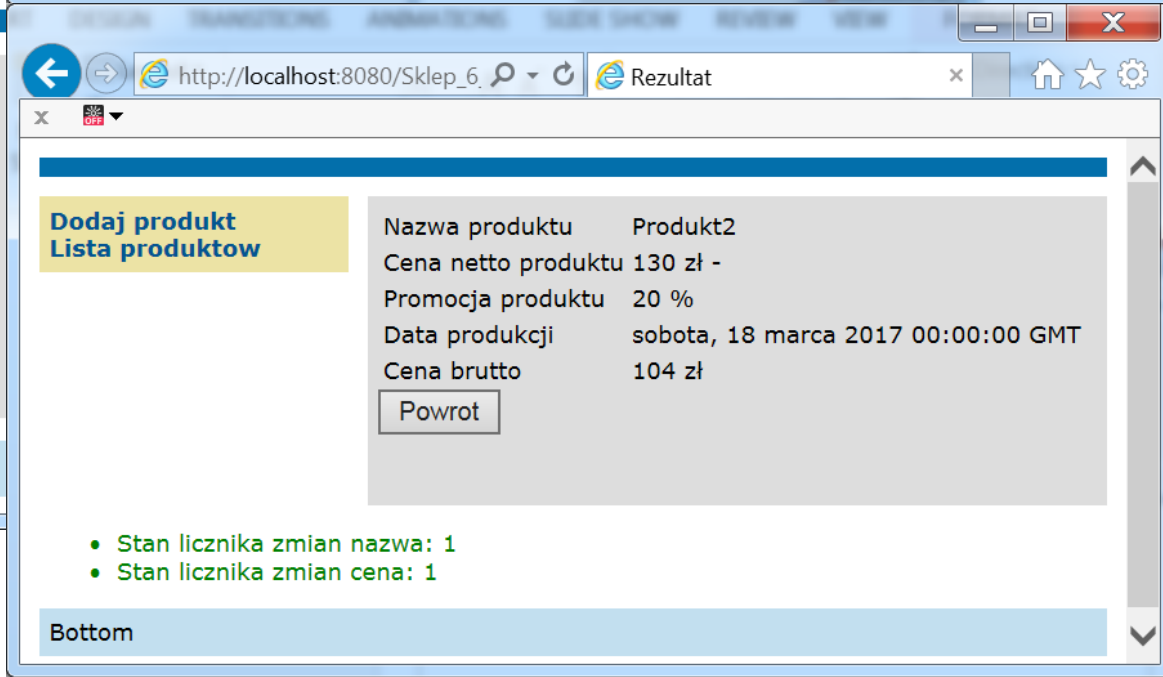
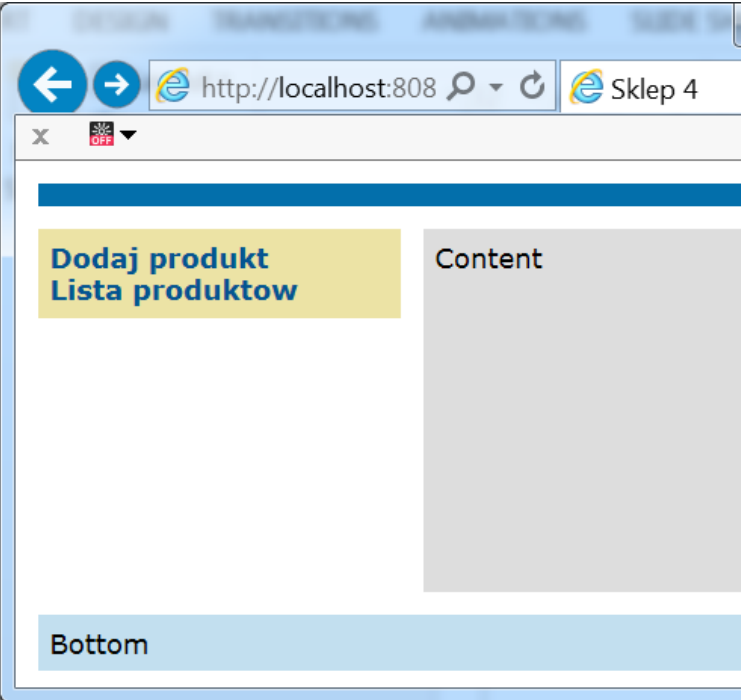
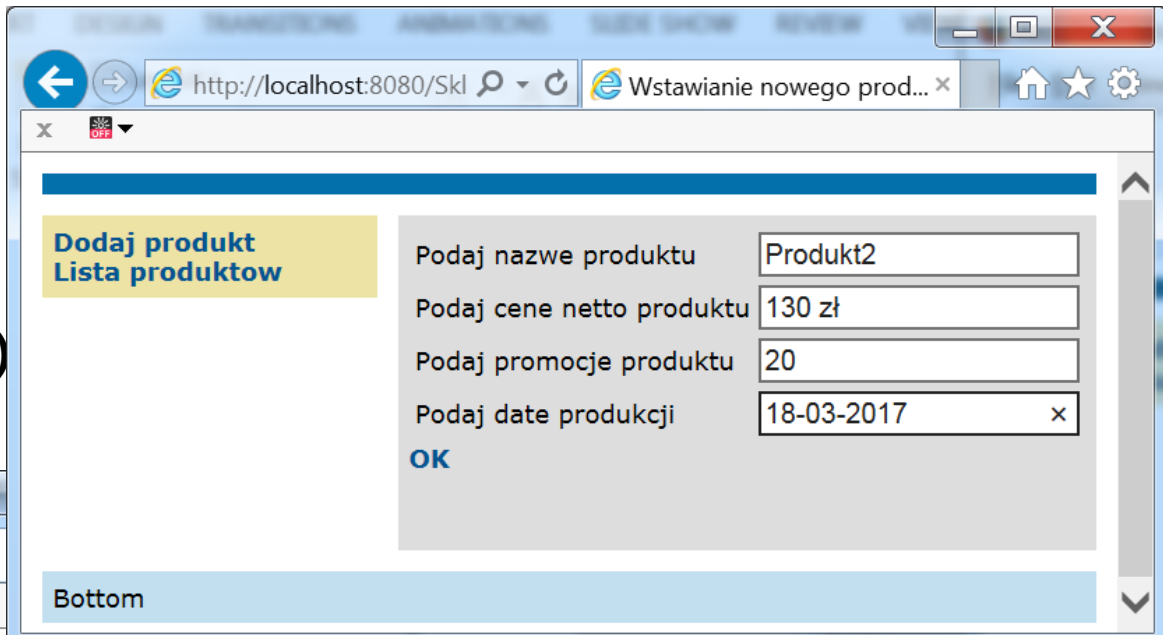
Uruchomienie aplikacji opartej na  
komponencie EJB typu Session,  
rodzaj Stateless - Clean and  
Build/Deploy/Run

Uruchomienie 1-ej instancji warstwy  
klienta aplikacji i wprowadzenie  
danych produktu – może być  
uruchomiona automatycznie w  
przeglądarce domyślnej.



Programowanie komponentowe 5,  
Zofia Kruczkiewicz

**Uruchomienie 2-ej instancji warstwy klienta aplikacji aplikacji i wprowadzenie danych kolejnego produktu (przez wprowadzenie adresu: [http://localhost:8080/Sklep\\_6\\_1](http://localhost:8080/Sklep_6_1))**





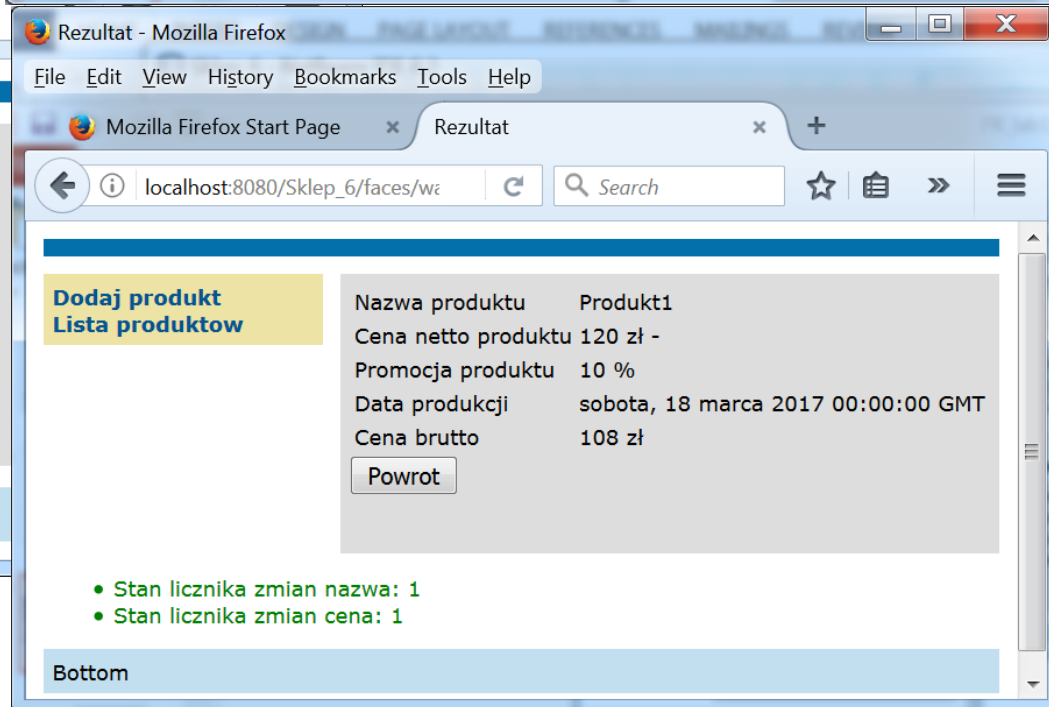
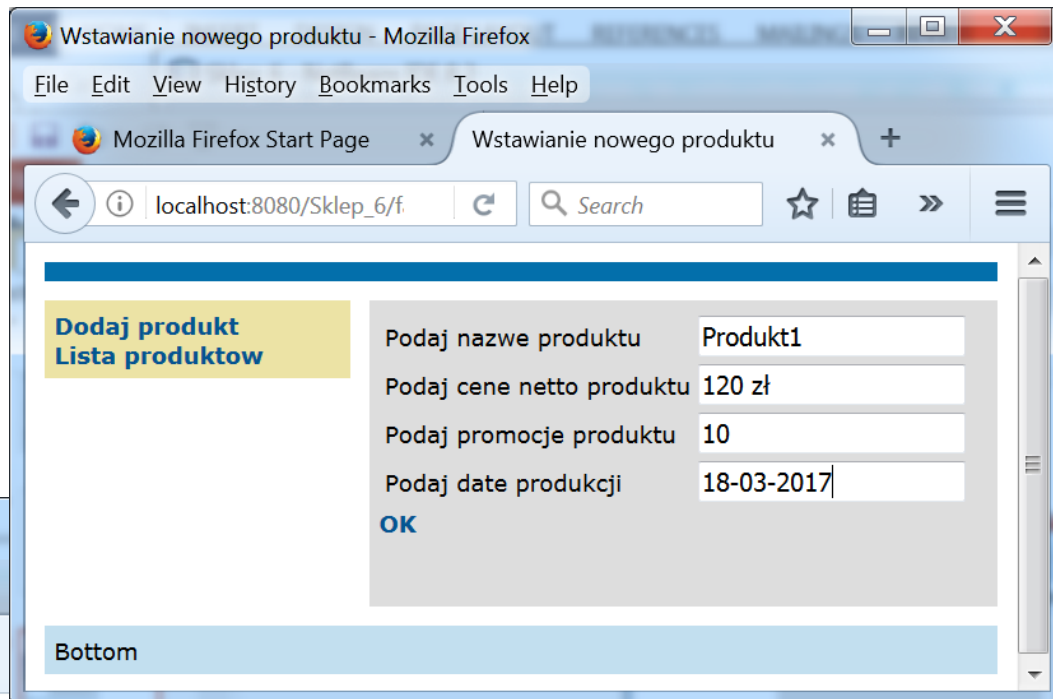
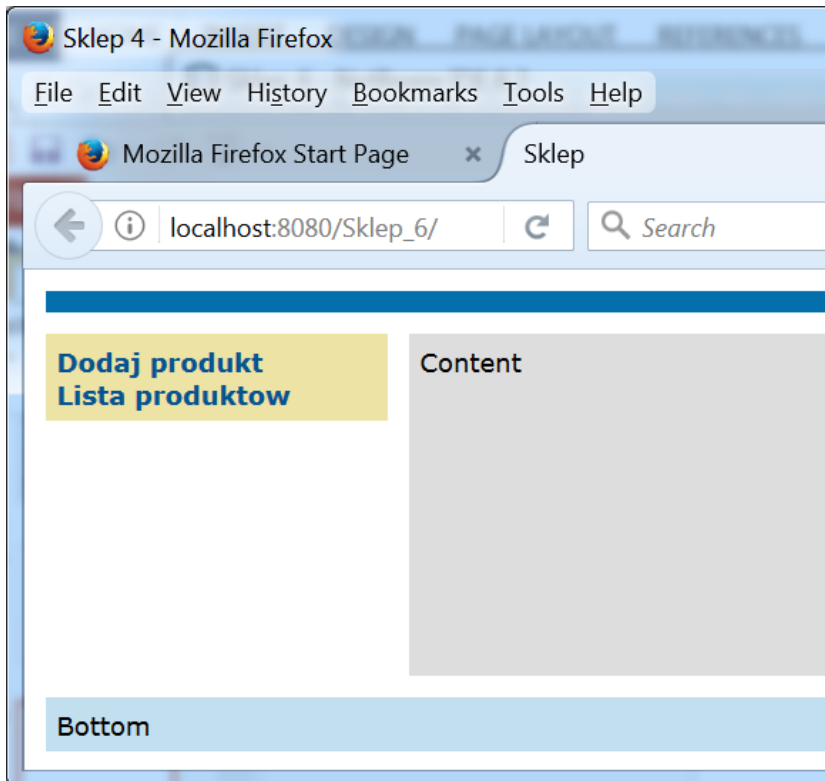
Po uruchomieniu formularza **Lista produktów** w obu instancjach klientów aplikacji widać, że korzystają **z tej samej instancji** komponentu typu **Fasada\_warstwy\_biznesowej\_ejb**, czyli tej samej instancji obiektu **Fasada\_warstwy\_biznesowej**.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

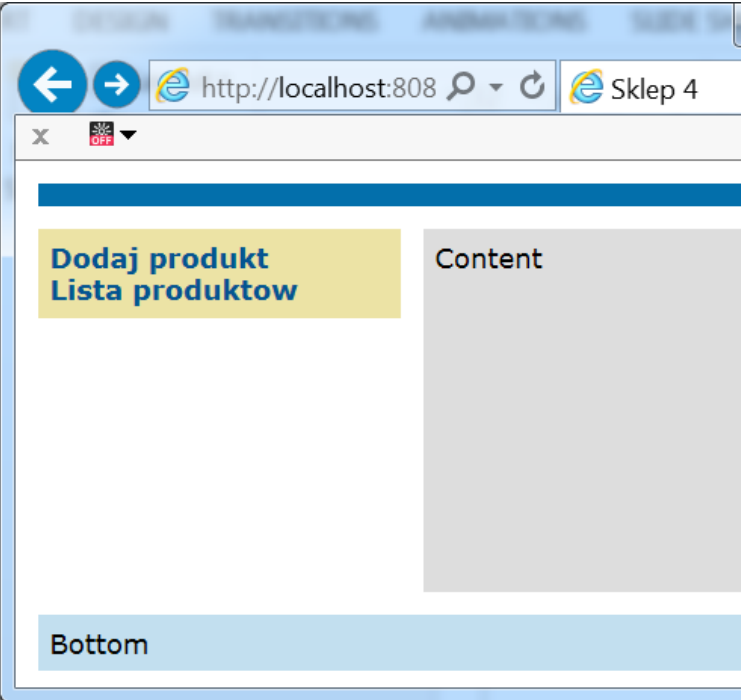
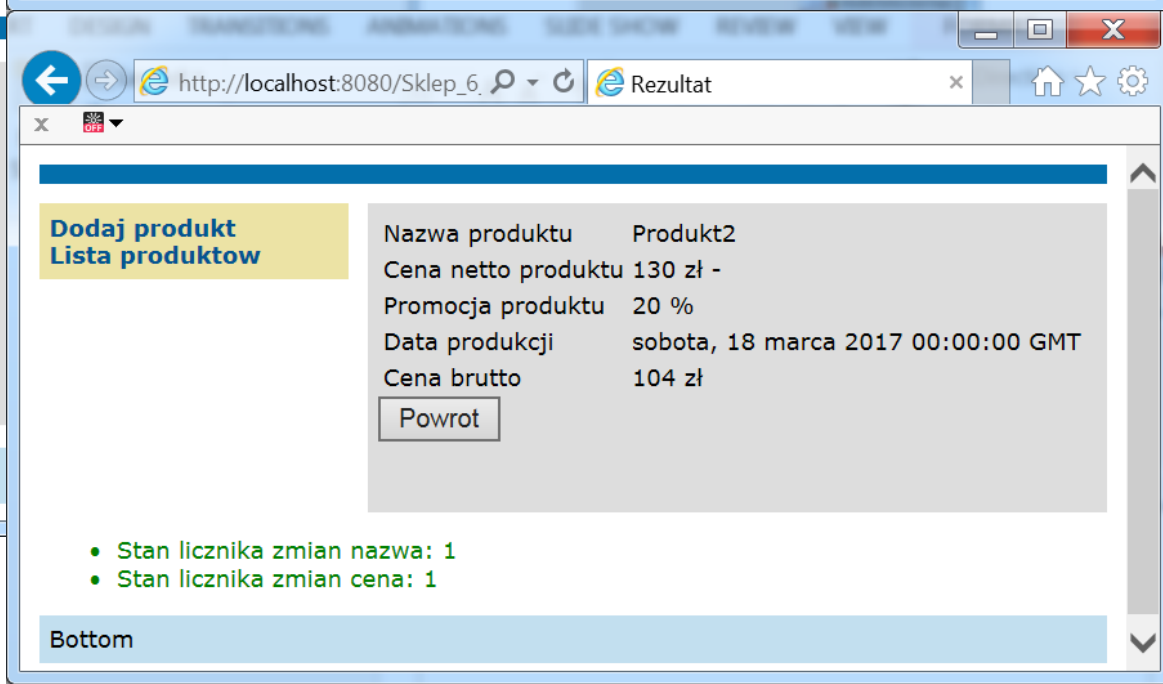
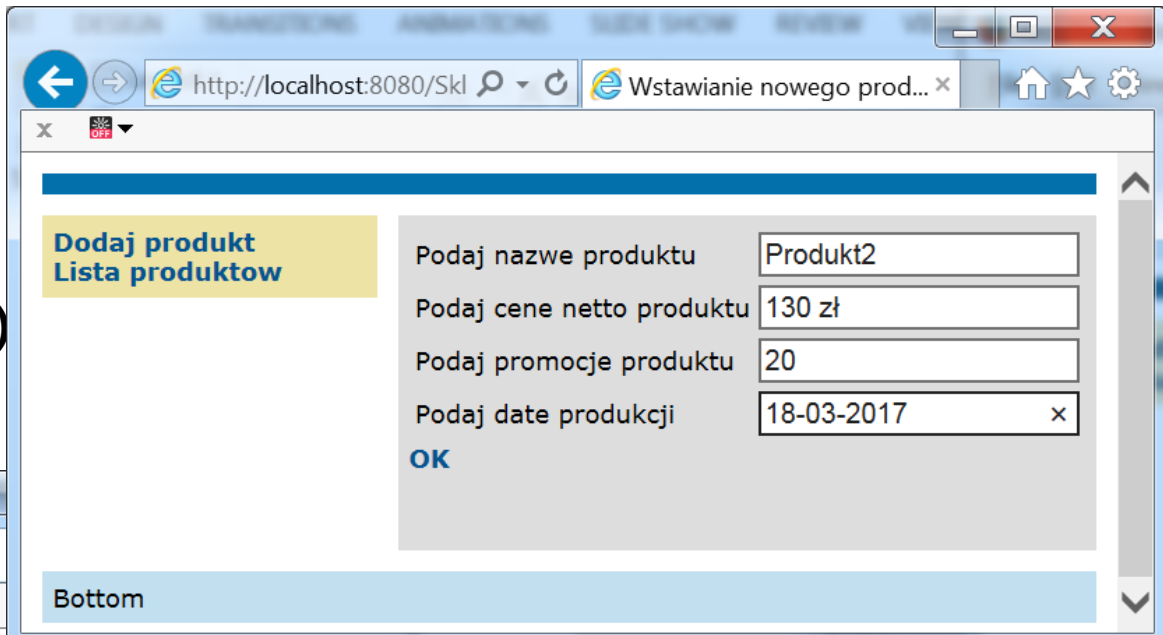
Uruchomienie aplikacji opartej na  
komponencie EJB typu **Session**,  
rodzaj **Stateless** - Clean and  
Build/Deploy/Run

Uruchomienie 1-ej instancji warstwy  
klienta aplikacji i wprowadzenie  
danych produktu – może być  
uruchomiona automatycznie w  
przeglądarce domyślnej.



Programowanie komponentowe 5,  
Zofia Kruczkiewicz

Uruchomienie 2-ej instancji warstwy klienta aplikacji aplikacji i wprowadzenie danych kolejnego produktu (prze wprowadzenie adresu: [http://localhost:8080/Sklep\\_6\\_1](http://localhost:8080/Sklep_6_1))

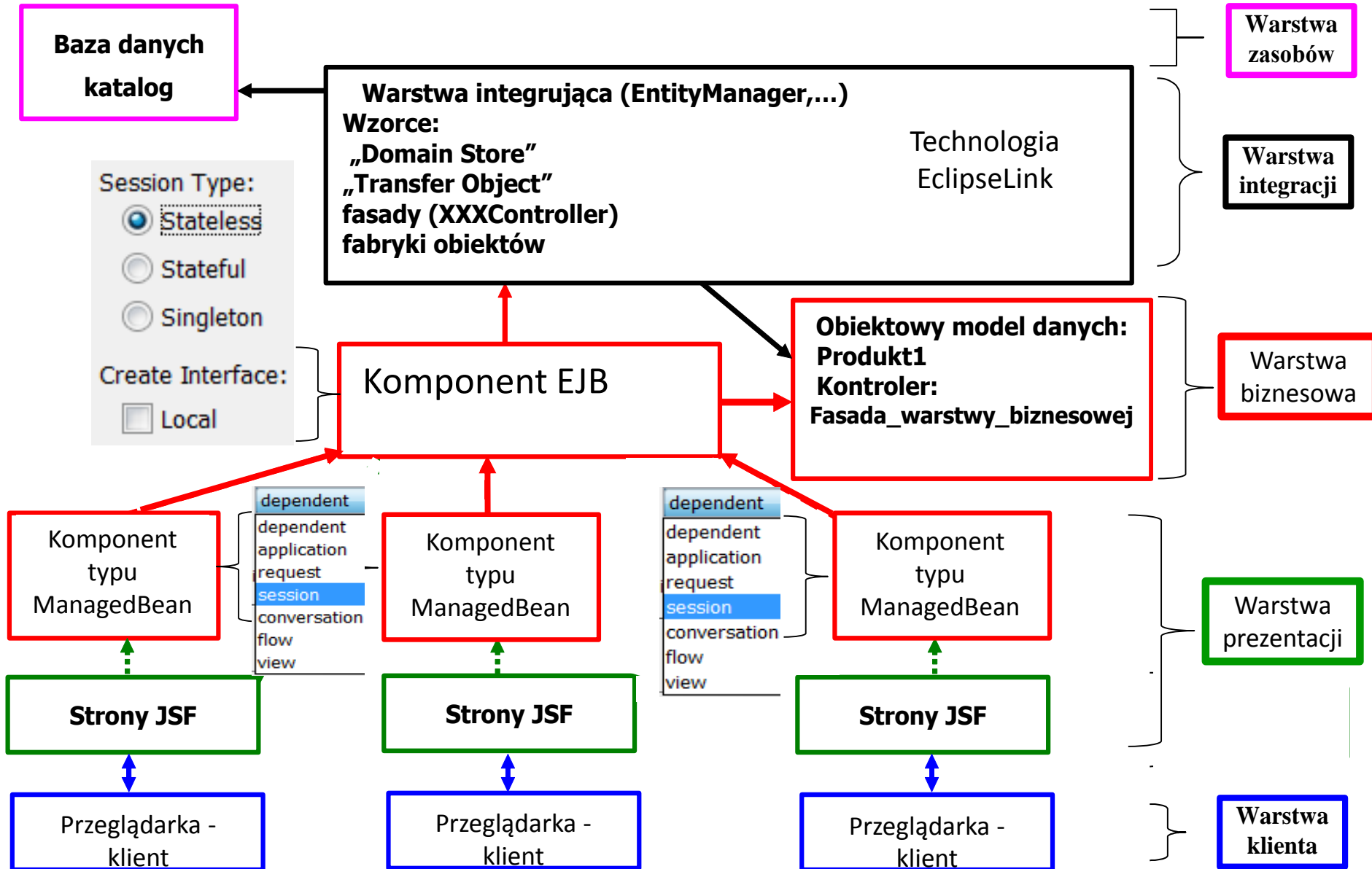


Po uruchomieniu formularza **Lista produktów** w obu instancjach klientów aplikacji widać, że korzystają **z tej samej instancji** komponentu typu **Fasada\_warstwy\_biznesowej\_ejb**, czyli tej samej instancji obiektu **Fasada\_warstwy\_biznesowej**.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

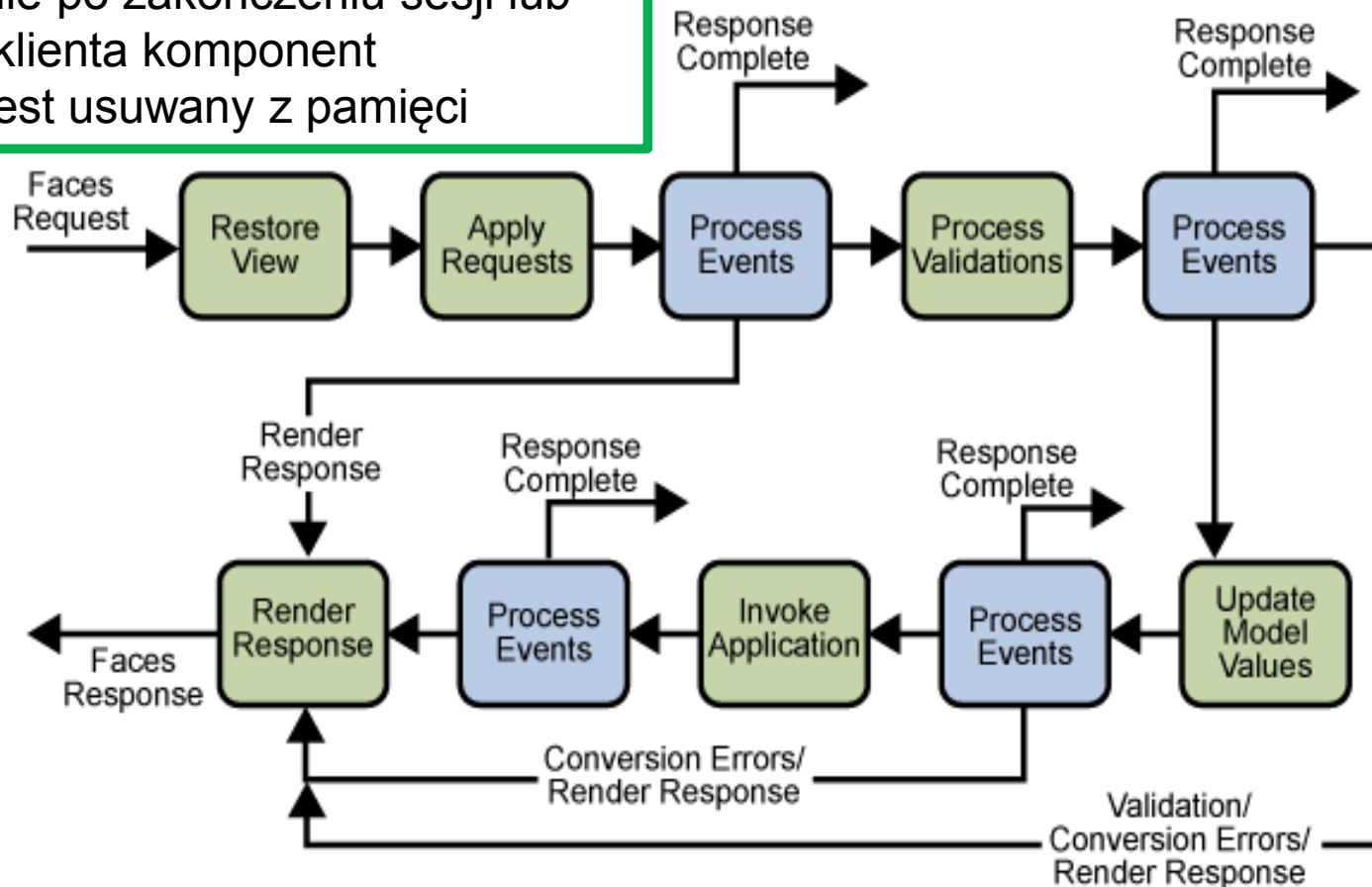
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

# Ad. 1.2. Architektura aplikacji pięciowarstwowej -Java EE 7.0 JavaServer Faces



# Standard cyklu życia „Request-Response” dla JavaServer Faces

**SessionScoped:** podczas uruchomienia instancji aplikacji klienta tworzony jest nowy komponent typu **Managed\_produk**t – wszystkie jego dane są zaktualizowane podczas fazy Invoke Application. Po zakończeniu fazy Render Response komponent pozostaje w pamięci. Jeśli nie wystąpią błędy, jedynie po zakończeniu sesji lub zamknięciu aplikacji klienta komponent **Managed\_produk**t jest usuwany z pamięci



## Tworzenie modelu komponentu dataTable w komponentcie typu Managed\_produkty na stronie lista\_produktyow.xhtml

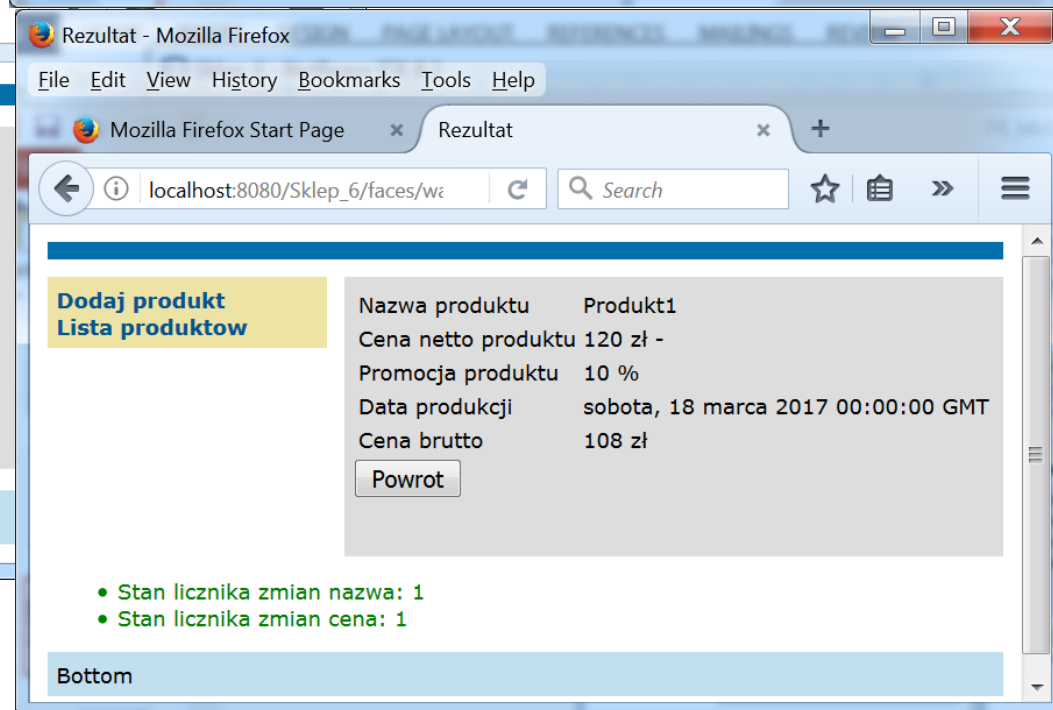
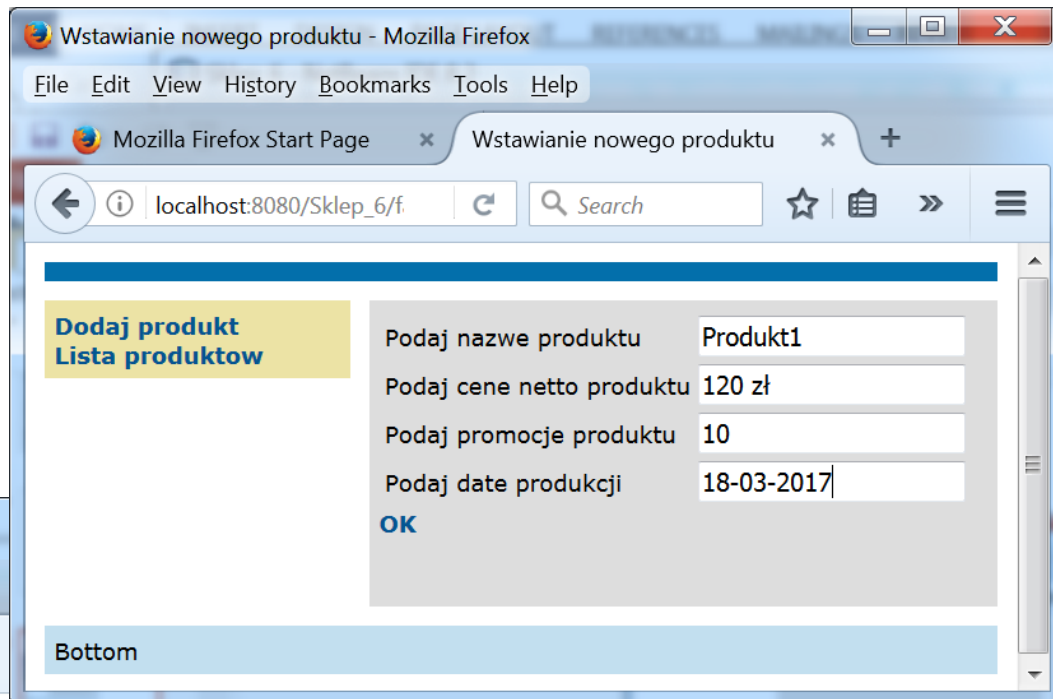
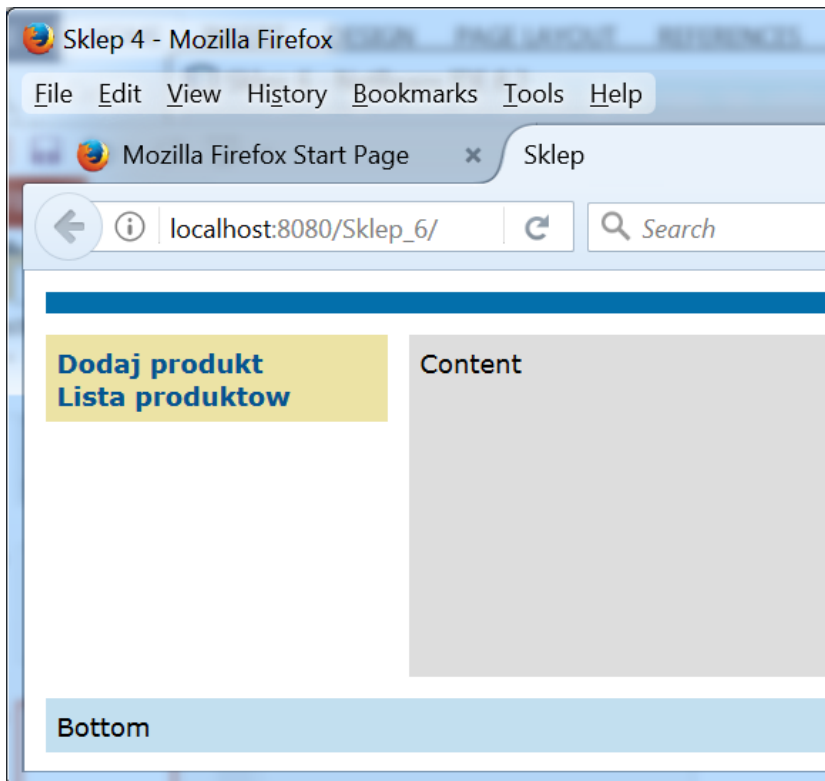
```
public DataModel utworz_DataModel() {  
    return new ListDataModel(fasada.items());  
}  
  
public DataModel getItems() {  
    if (items == null || stan==1) { ←  
        items = utworz_DataModel();  
    }  
    return items;  
}
```

Przy czasie życia SessionScope obiektu typu Managed\_produkty atrybut items jest **równy null** tylko podczas obsługi pierwszego żądania. Badanie zmiennej stan pozwala na aktualizację modelu items: stan==1 oznacza, że  **dodano nowe dane w danej instancji aplikacji internetowej.**

Model elementów strony **rezultat2.xhtml** jest tworzony zawsze – albo pokazane są dane nowego produktu lub pojawia się napis o braku wstawienia nowego produktu (zmienna stan=0)

Uruchomienie aplikacji opartej na  
komponencie EJB typu Session,  
rodzaj Stateless - Clean and  
Build/Deploy/Run

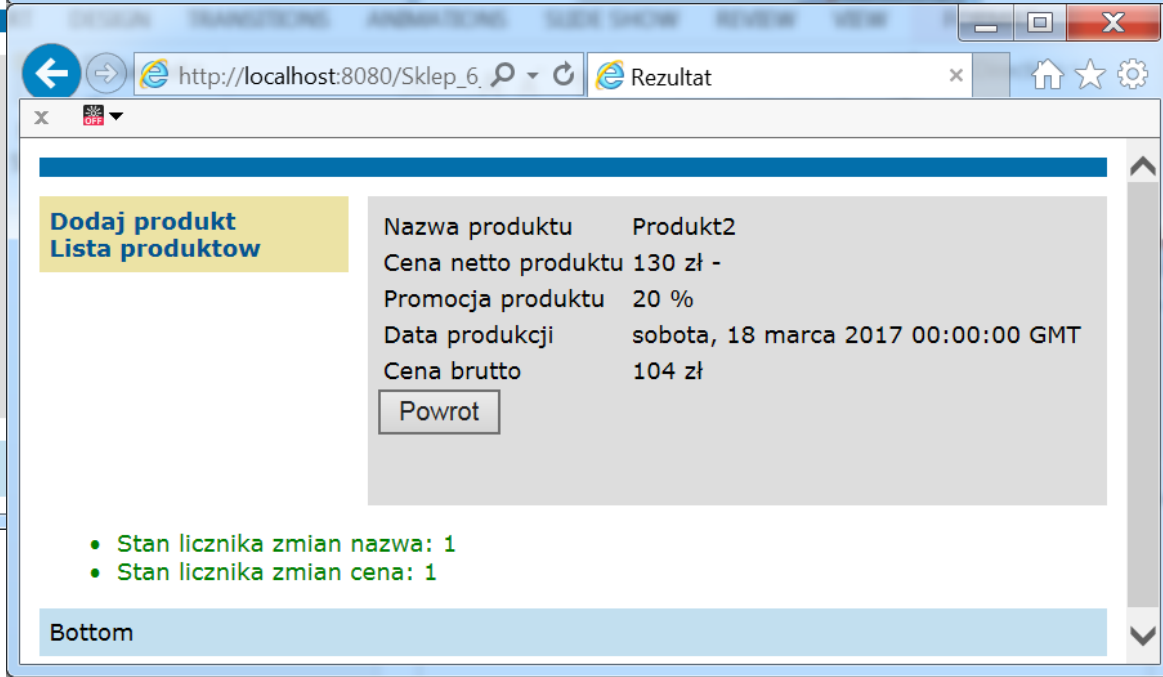
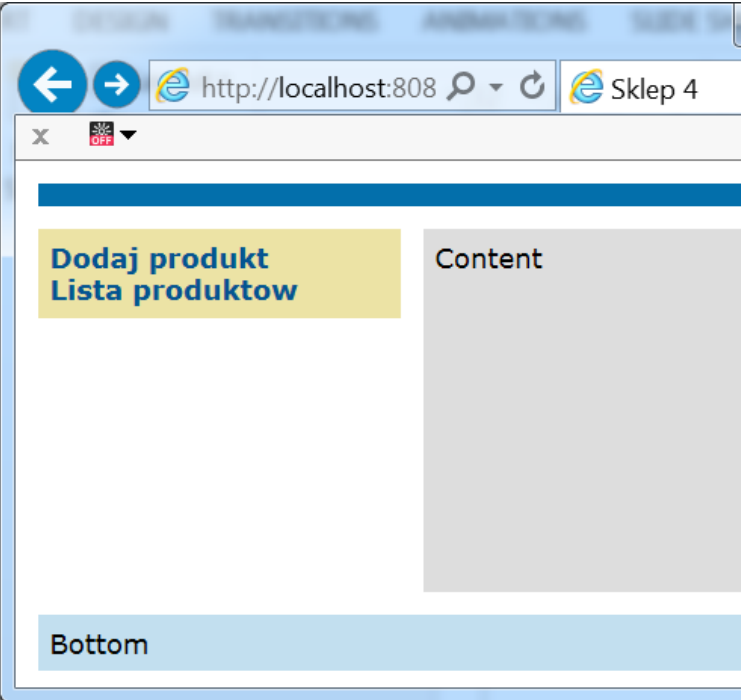
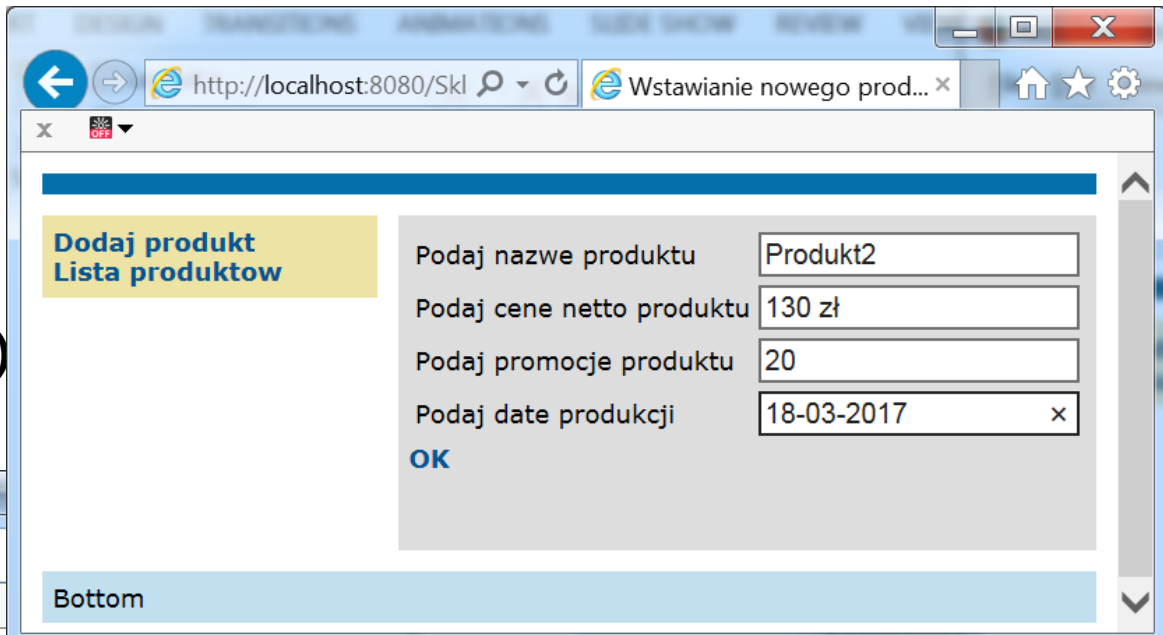
Uruchomienie 1-ej instancji warstwy  
klienta aplikacji i wprowadzenie  
danych produktu – może być  
uruchomiona automatycznie w  
przeglądarce domyślnej.



Programowanie komponentowe 5,  
Zofia Kruczkiewicz



**Uruchomienie 2-ej instancji warstwy klienta aplikacji aplikacji i wprowadzenie danych kolejnego produktu (przez wprowadzenie adresu: [http://localhost:8080/Sklep\\_6\\_1](http://localhost:8080/Sklep_6_1))**



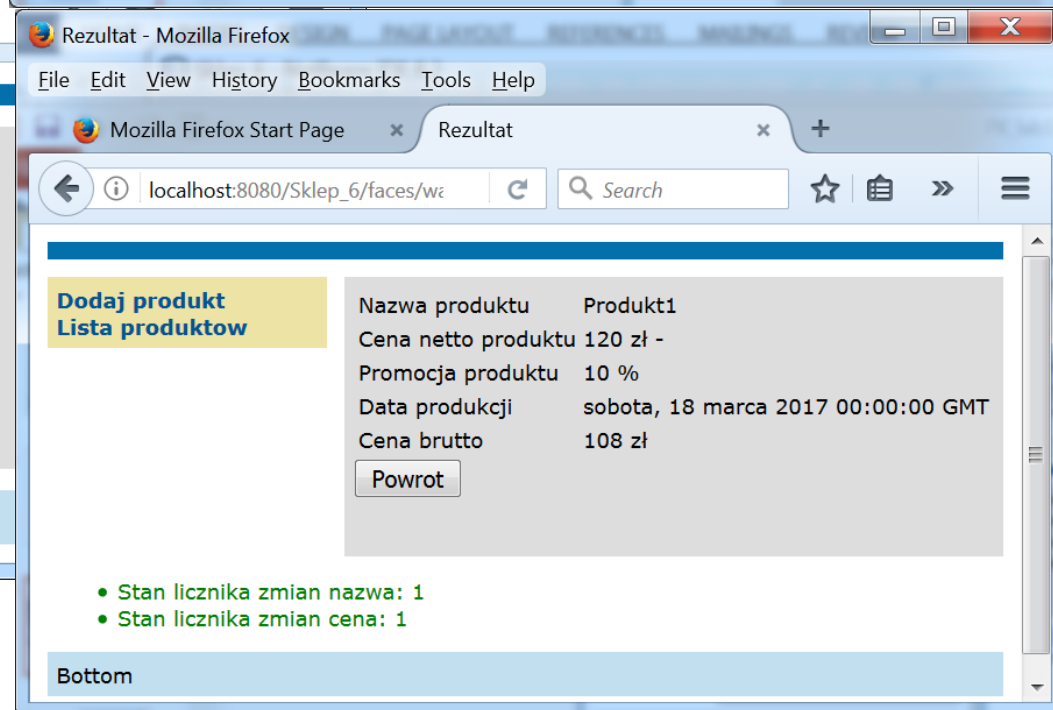
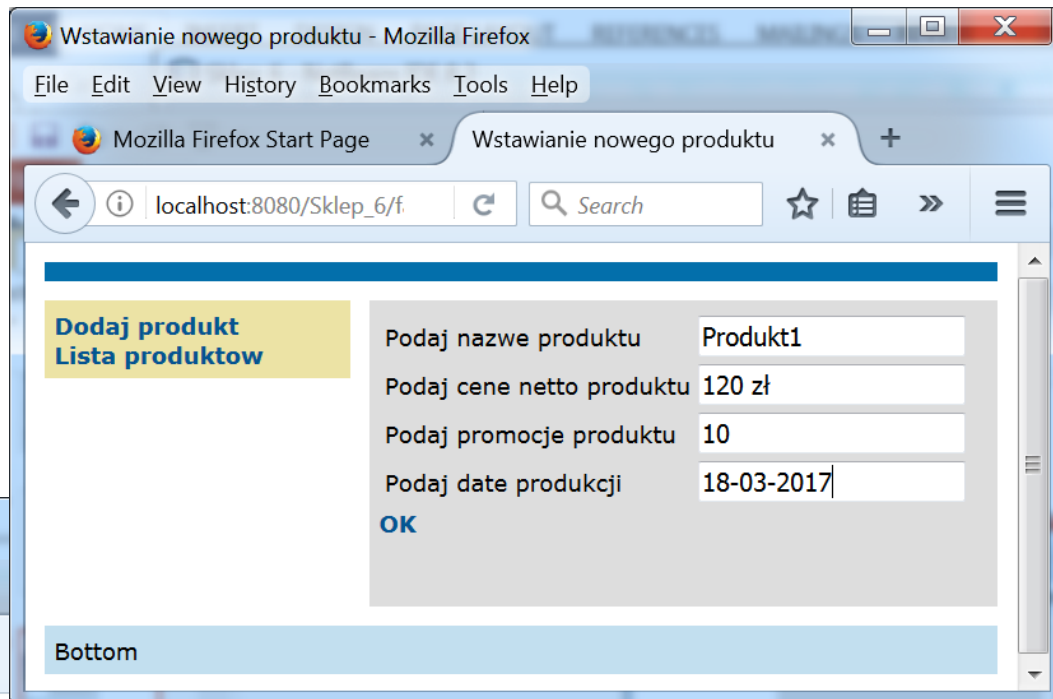
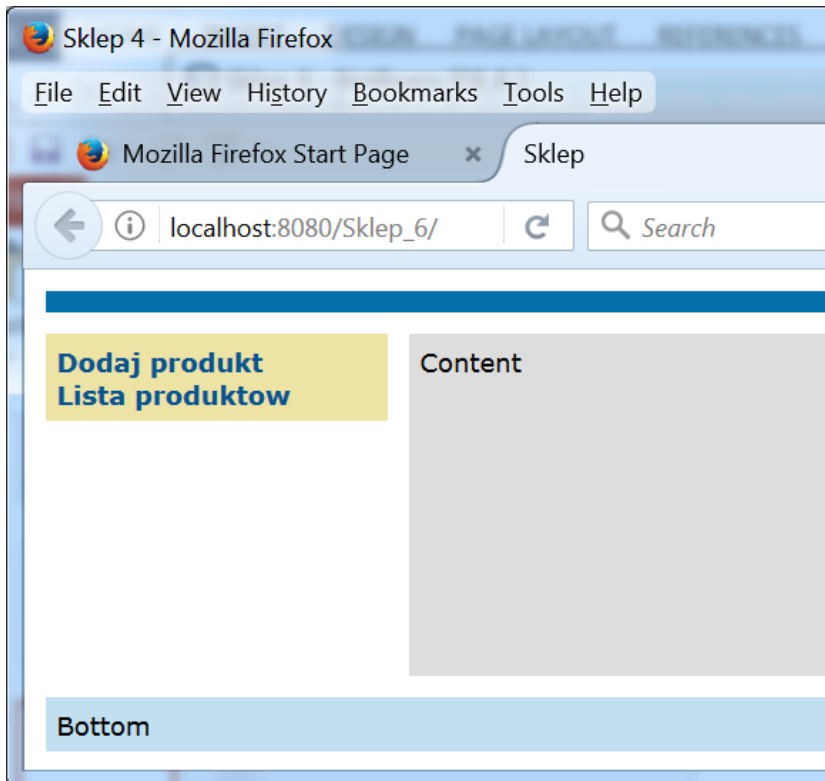
Po uruchomieniu formularza **Lista produktów** w obu instancjach klientów aplikacji widać, że korzystają **z tej samej instancji** komponentu typu **Fasada\_warstwy\_biznesowej\_ejb**, czyli tej samej instancji obiektu **Fasada\_warstwy\_biznesowej**.

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

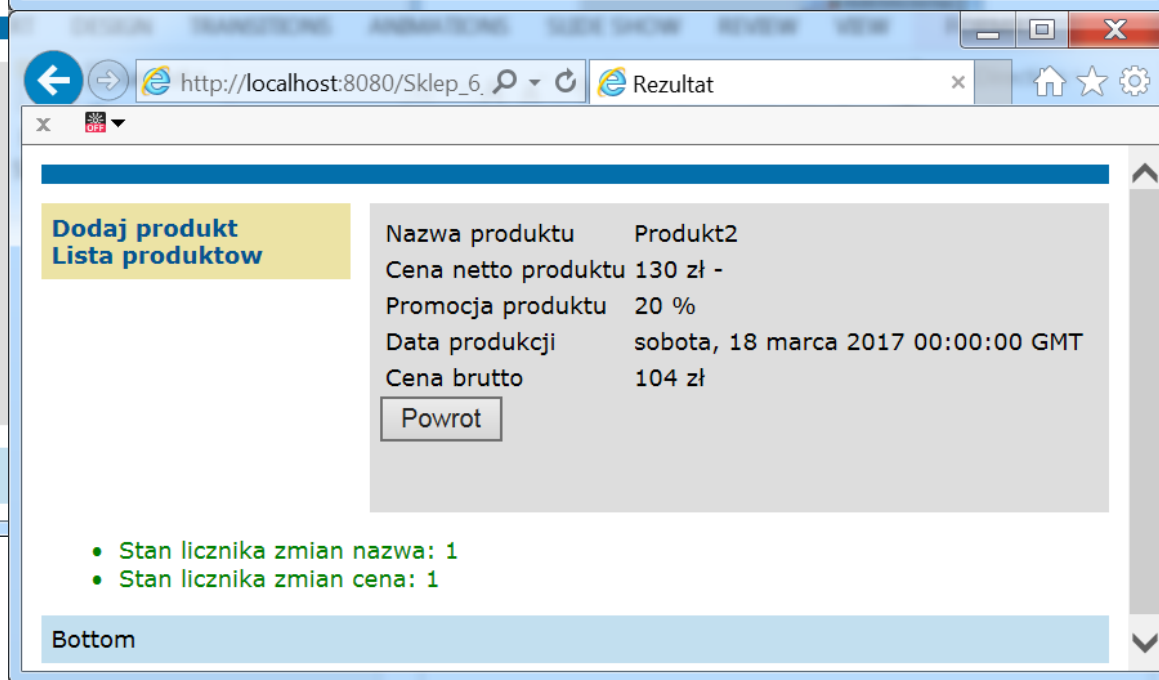
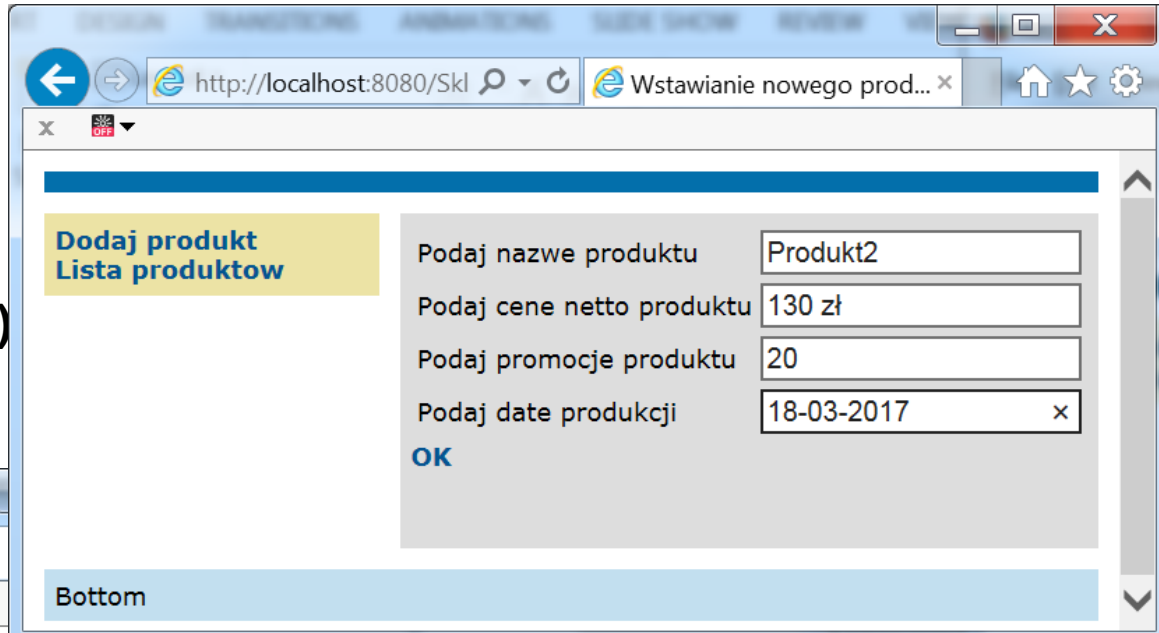
Uruchomienie aplikacji opartej na  
komponencie EJB typu Session,  
rodzaj Stateless - Clean and  
Build/Deploy/Run

Uruchomienie 1-ej instancji warstwy  
klienta aplikacji i wprowadzenie  
danych produktu – może być  
uruchomiona automatycznie w  
przeglądarce domyślnej.



Programowanie komponentowe 5,  
Zofia Kruczkiewicz

Uruchomienie 2-ej instancji warstwy klienta aplikacji aplikacji i wprowadzenie danych kolejnego produktu (przez wprowadzenie adresu: [http://localhost:8080/Sklep\\_6\\_1](http://localhost:8080/Sklep_6_1))



Po uruchomieniu formularza **Lista produktów** w obu instancjach klientów aplikacji widać, że korzystają **z tej samej instancji** komponentu typu **Fasada\_warstwy\_biznesowej\_ejb**, czyli tej samej instancji obiektu **Fasada\_warstwy\_biznesowej**.

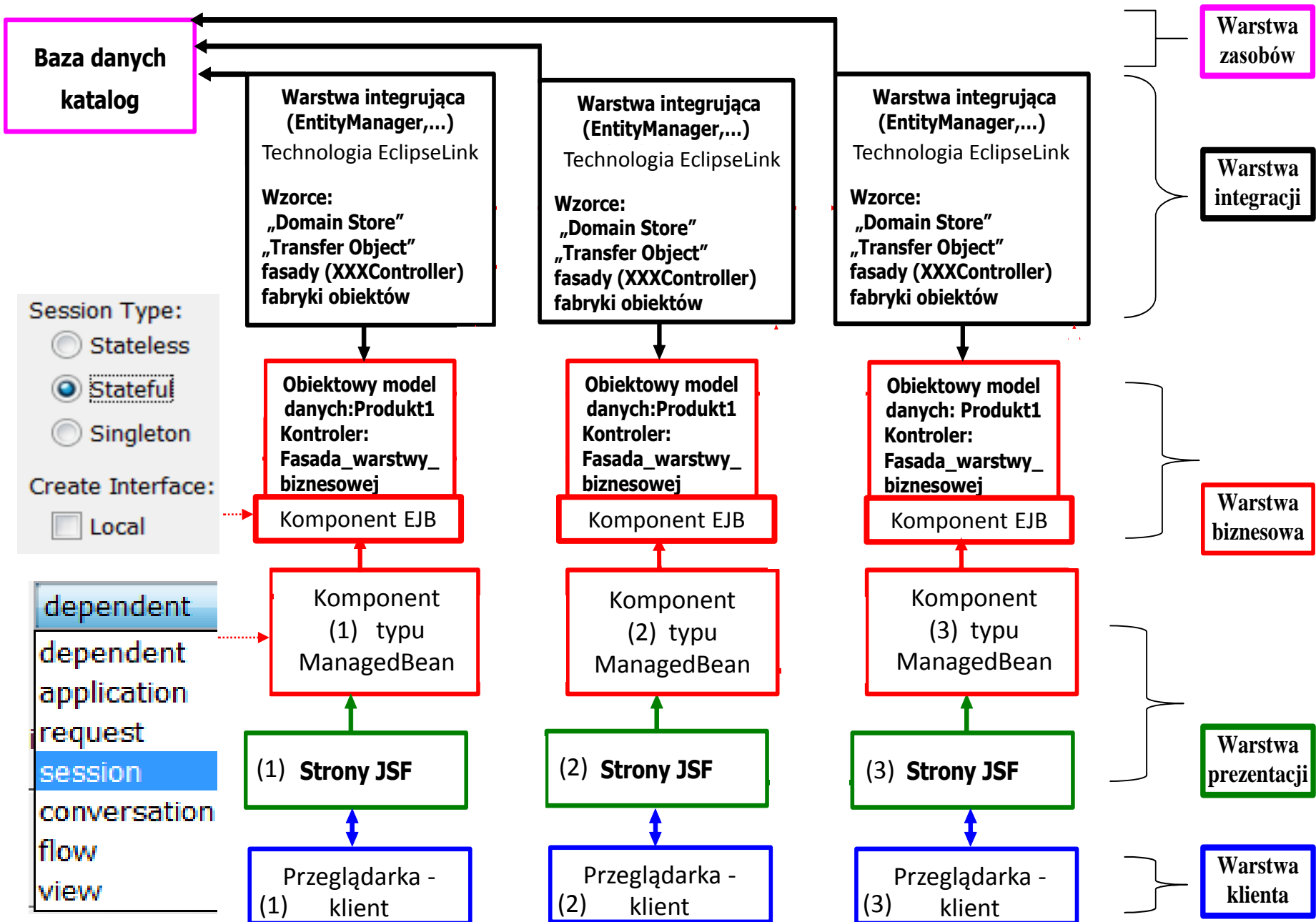
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	120.0	10	Sat Mar 18 01:00:00 CET 2017	108.0
2	Produkt2	130.0	20	Sat Mar 18 01:00:00 CET 2017	104.0

## 2. Aplikacja wielowarstwowa Java EE oparta na komponencie EJB typu Session Bean – rodzaj Stateful

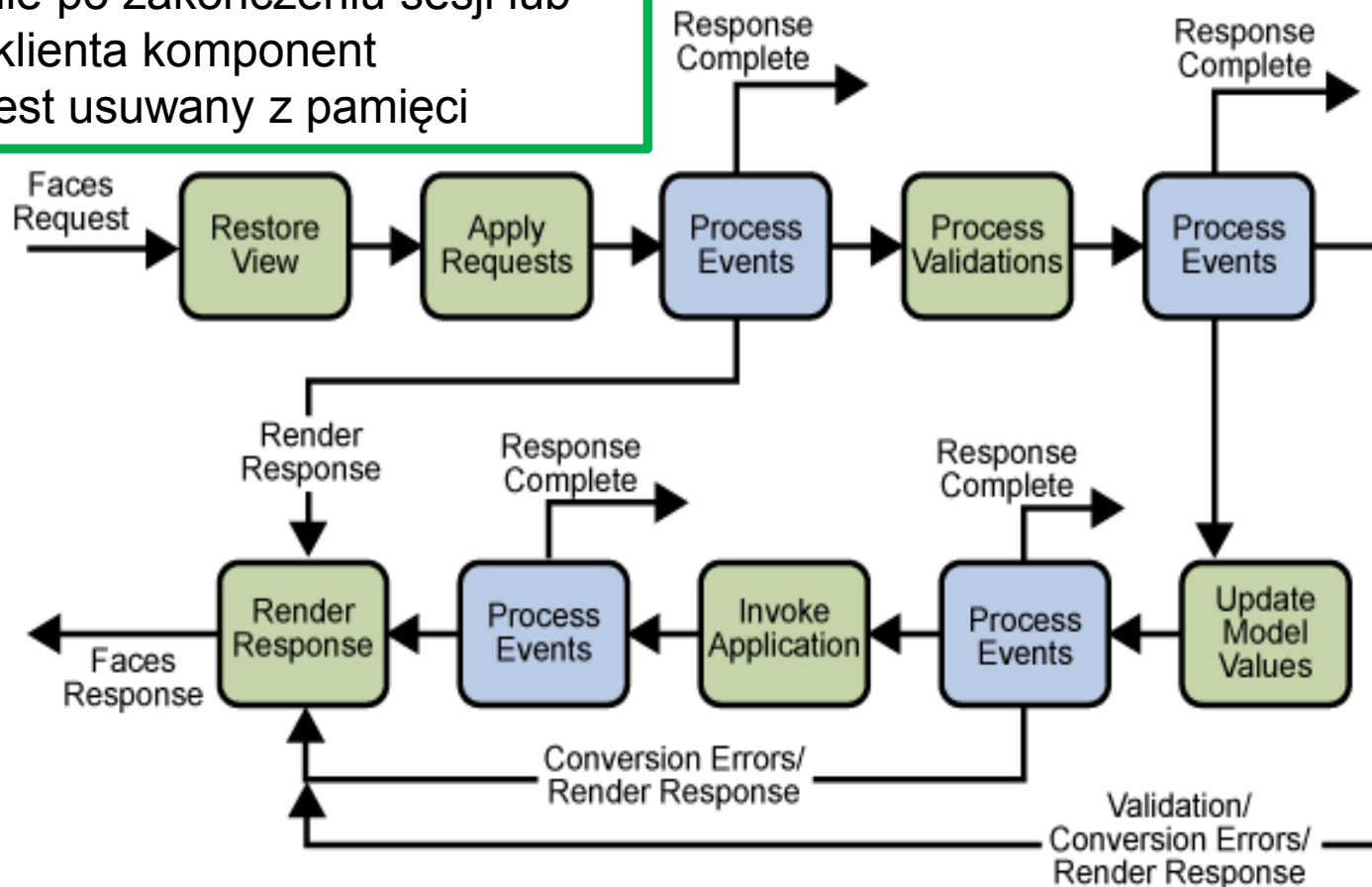
Warstwa internetowa oparta na komponencie Managed Bean o czasie życia **SessionScope**

# Architektura aplikacji pięciowarstwowej – Java EE 7.0 JavaServer Faces



# Standard cyklu życia „Request-Response” dla JavaServer Faces

**SessionScoped:** podczas uruchomienia instancji aplikacji klienta tworzony jest nowy komponent typu **Managed\_produk**t – wszystkie jego dane są zaktualizowane podczas fazy Invoke Application. Po zakończeniu fazy Render Response komponent pozostaje w pamięci. Jeśli nie wystąpią błędy, jedynie po zakończeniu sesji lub zamknięciu aplikacji klienta komponent **Managed\_produk**t jest usuwany z pamięci

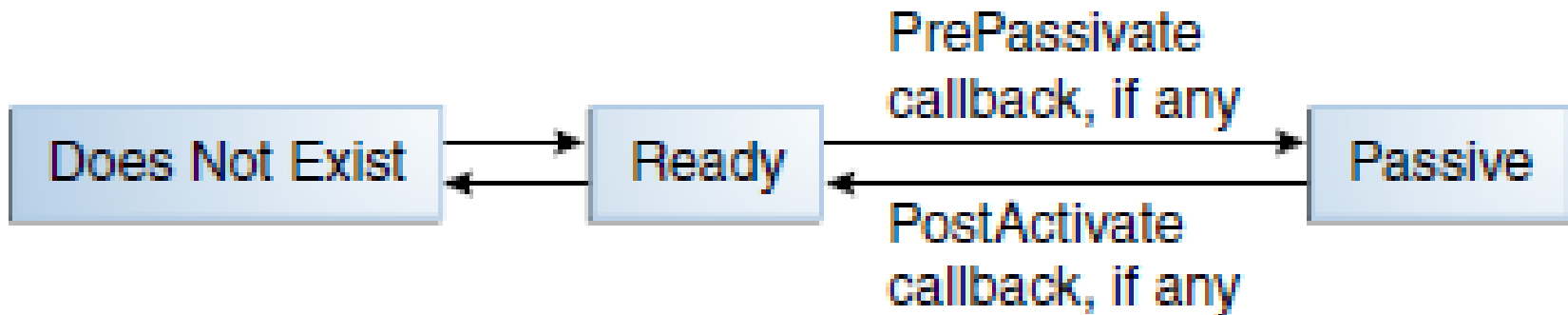




# Cykl życia stanowych (Stateful) komponentów typu Session Bean

## Zadania kontenera EJB:

1. Tworzy komponent
2. Dependency injection, jeśli są
3. Metoda PostConstruct callback, jeśli jest
4. Metoda Init, lub ejbCreate<METHOD>, jeśli są



1. Usuwa komponent
2. Metoda PreDestroy callback, jeśli jest

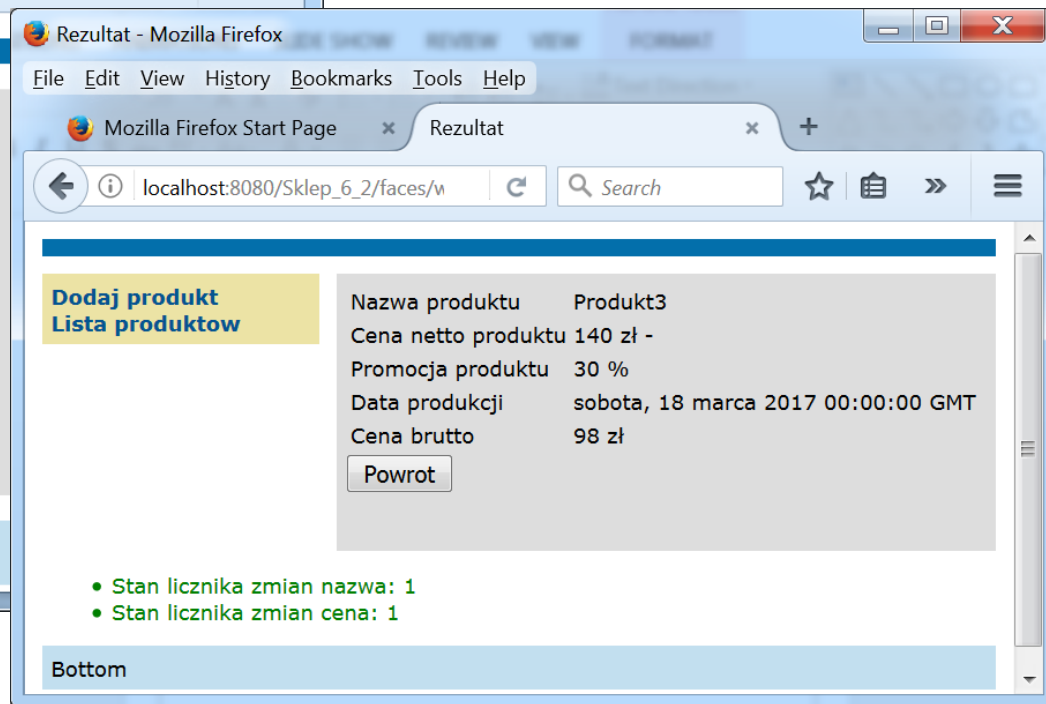
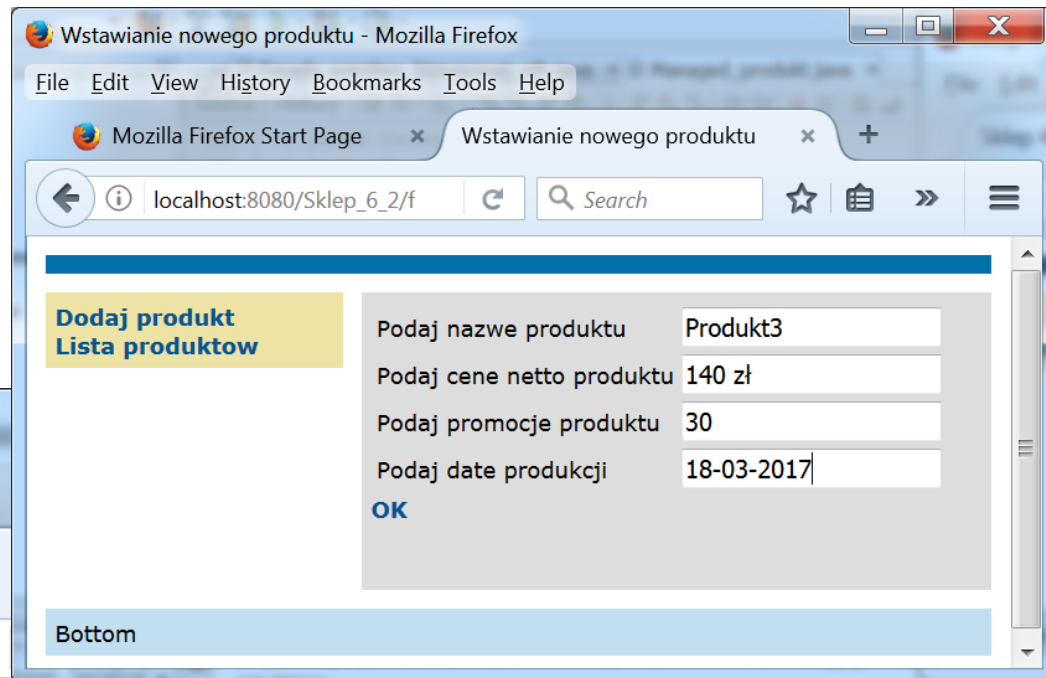
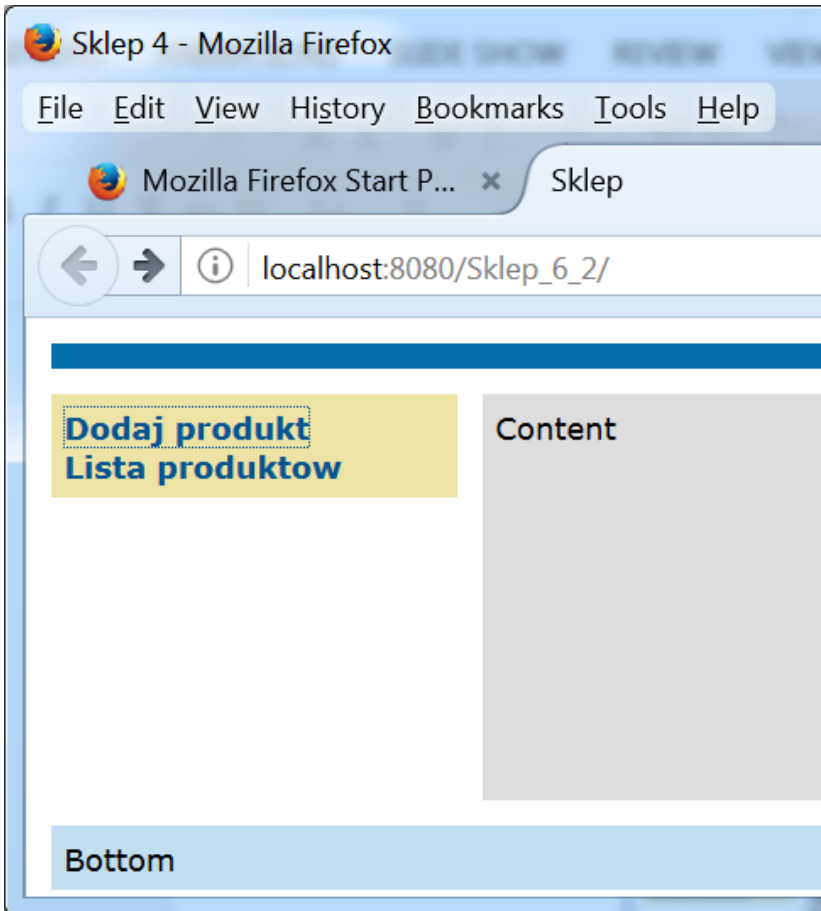
## Tworzenie modelu komponentu dataTable w kompoenencie typu Managed\_produk t na stronie lista\_produk tow.xhtml – podobnie jak w p.1.2

```
public DataModel utworz_DataModel() {  
    return new ListDataModel(fasada.items());  
}  
  
public DataModel getItems() { ←  
    if (items == null || stan==1) {  
        items = utworz_DataModel();  
        stan=0;  
    }  
    return items;  
}
```

Przy czasie życia SessionScope obiektu typu Managed\_produk t items jest **równy null** tylko podczas obsługi pierwszego żądania pierwszego. Badanie zmiennej stan pozwala na właściwą aktualizację modelu items: stan==1 oznacza, że **dodano nowe dane w danej instancji klienta internetowego**

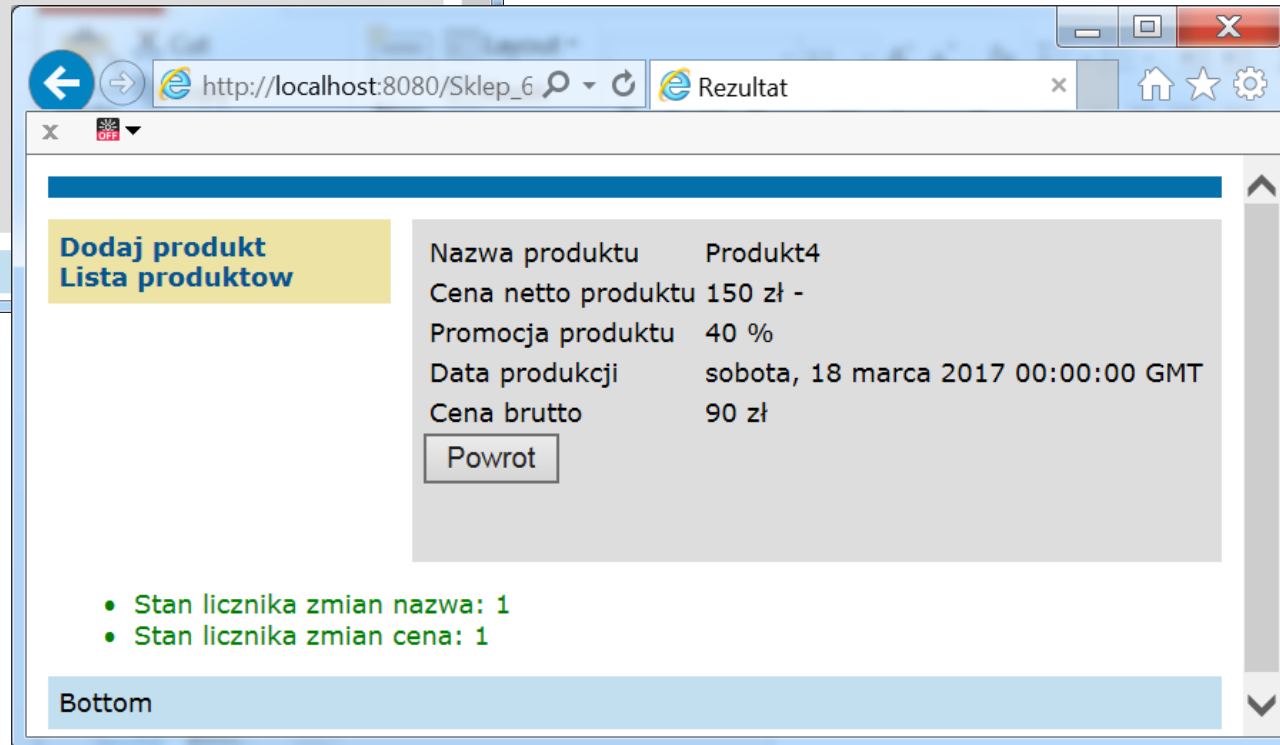
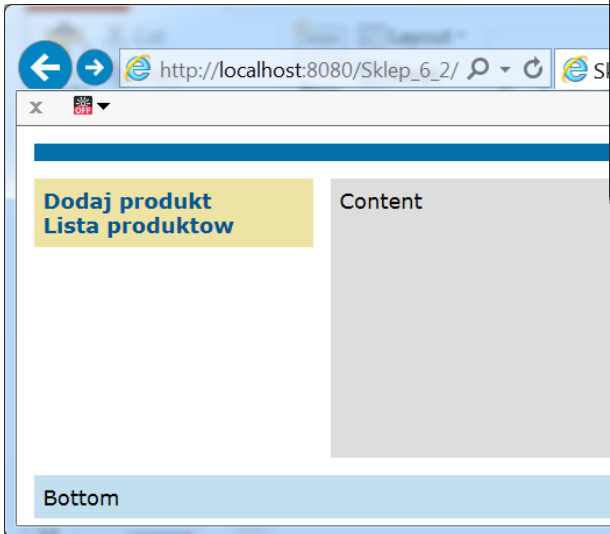
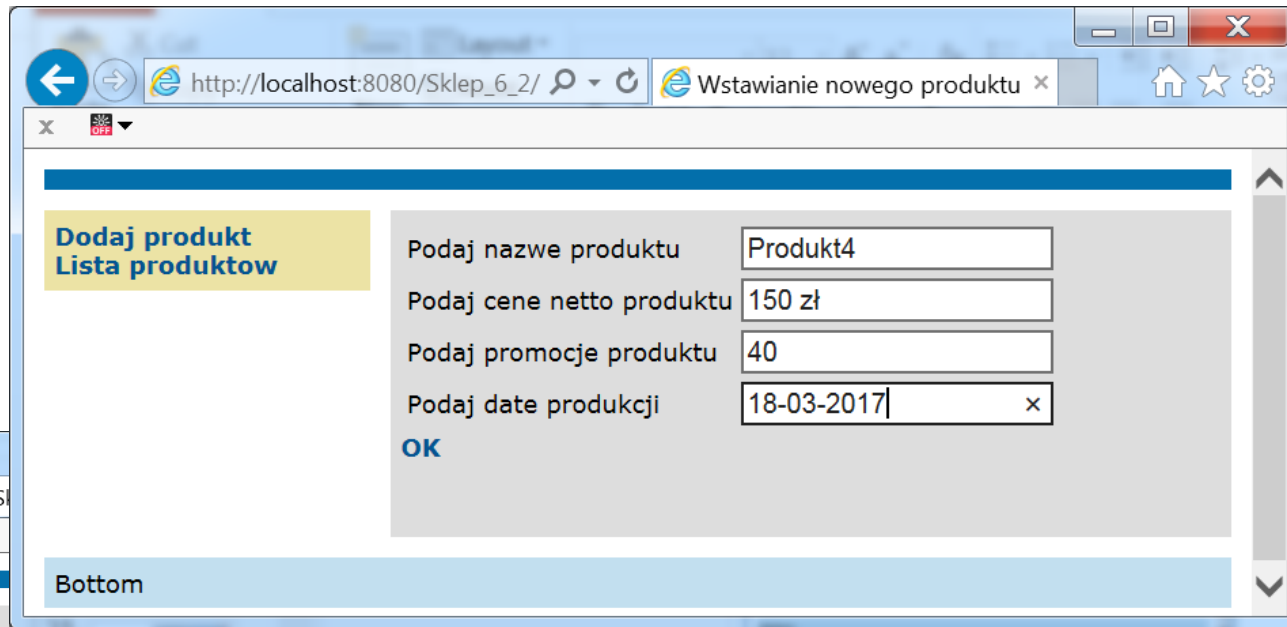
Model elementów strony **rezultat2.xhtml** jest tworzony zawsze – albo pokazane są dane nowego produktu lub pojawia się napis o braku wstawienia nowego produktu (zmienna stan=0)

Uruchomienie aplikacji opartej na  
komponencie EJB typu Session,  
rodzaj Stateful - Clean and  
Build/Deploy/Run  
Uruchomienie 1-ej instancji warstwy  
klienta aplikacji i wprowadzenie  
danych produktu

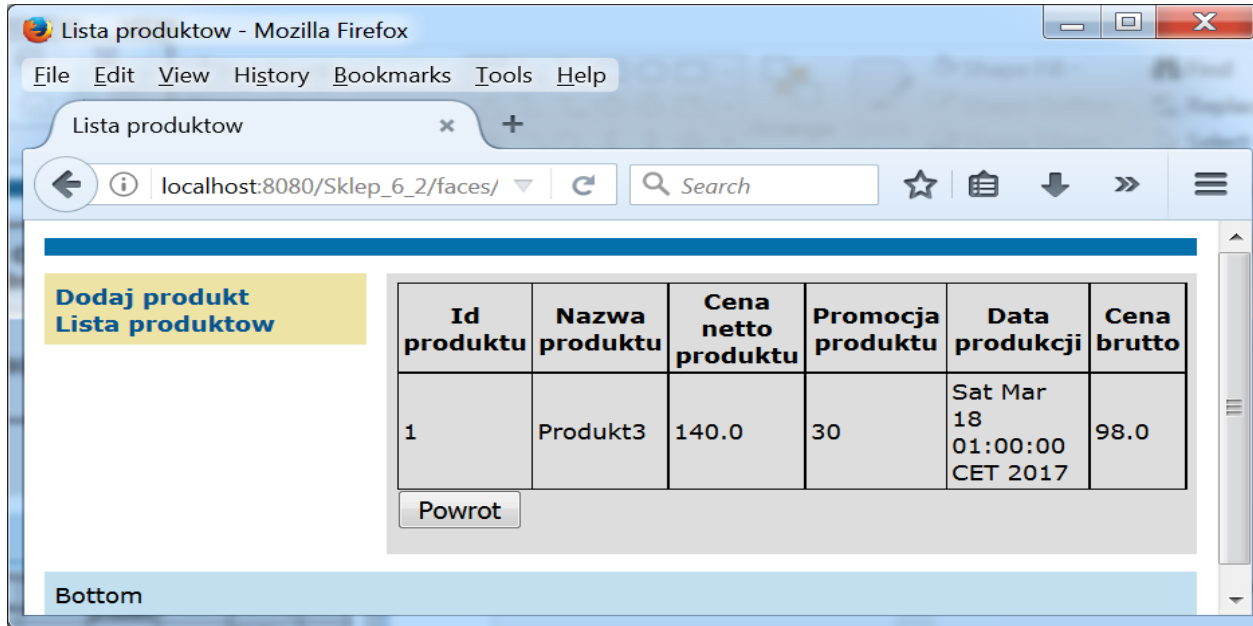


**Uruchomienie 2-ej instancji warstwy klienta aplikacji i wprowadzenie danych produktu (podobnie jak w p.1.16:**

**[http://localhost:8080/Sklep\\_6\\_2](http://localhost:8080/Sklep_6_2)**



Po uruchomieniu formularza **Lista produktów** w obu instancjach klientów aplikacji widać, że korzystają **z różnych instancji** komponentu typu **Fasada\_warstwy\_biznesowej\_ejb**, czyli różnych instancji obiektu **Fasada\_warstwy\_biznesowej**.



Lista produktow - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Lista produktow

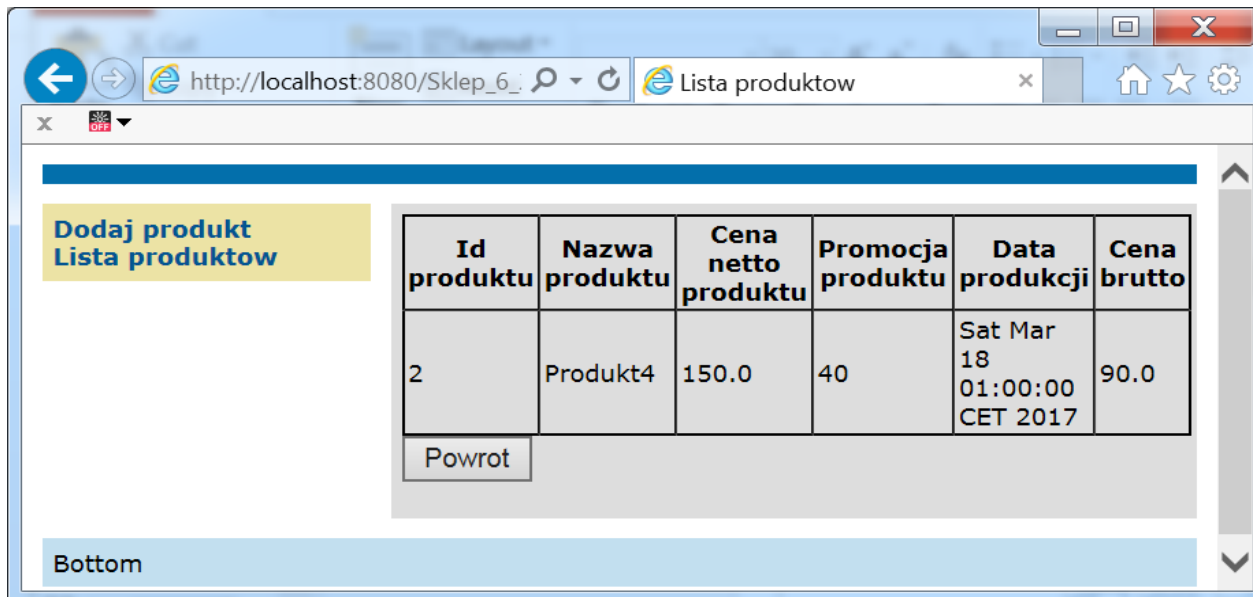
localhost:8080/Sklep\_6\_2/faces/ Search

**Dodaj produkt**  
**Lista produktow**

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt3	140.0	30	Sat Mar 18 01:00:00 CET 2017	98.0

Powrot

Bottom



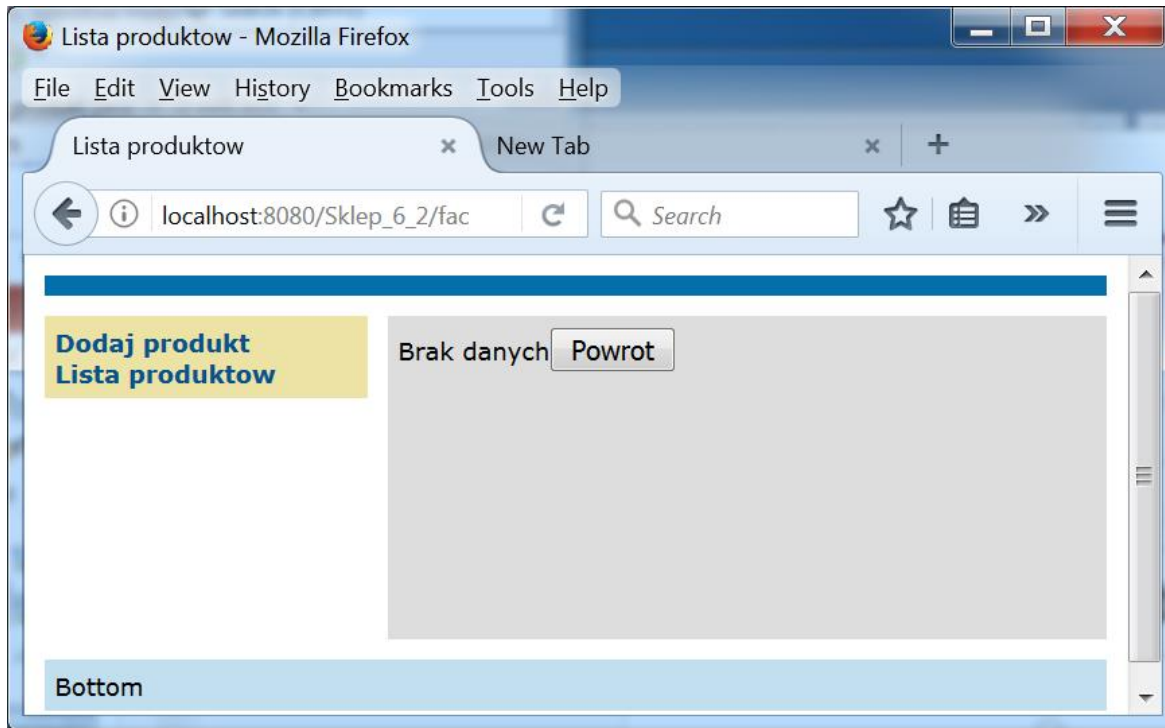
http://localhost:8080/Sklep\_6\_... Lista produktow

**Dodaj produkt**  
**Lista produktow**

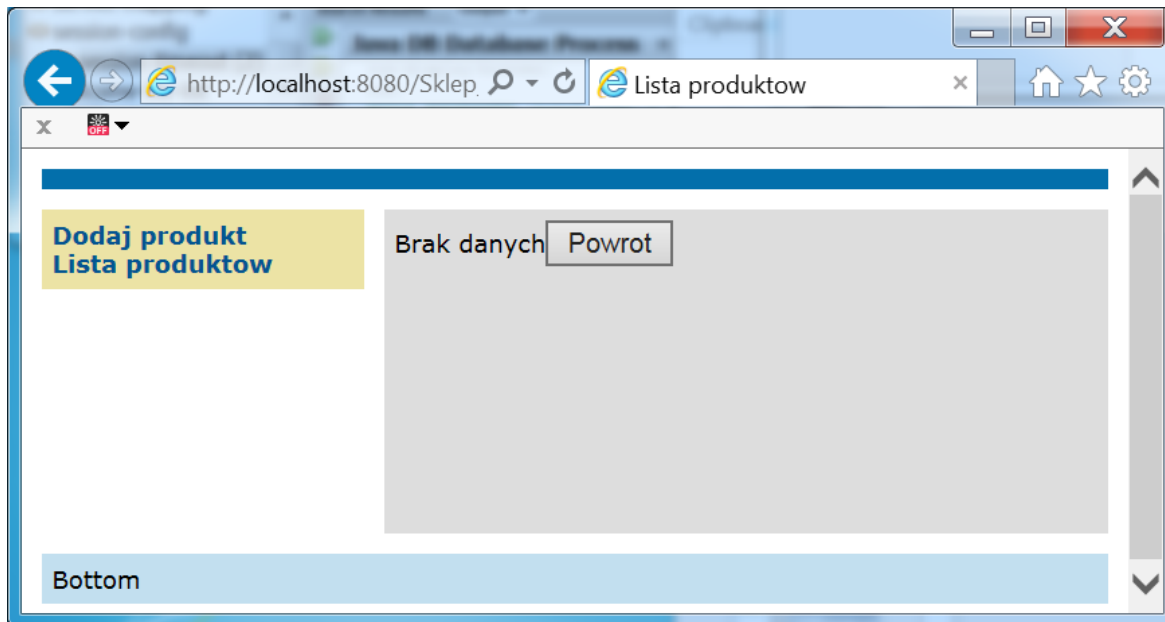
Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
2	Produkt4	150.0	40	Sat Mar 18 01:00:00 CET 2017	90.0

Powrot

Bottom



Po upływie 2 min próba wywołania strony Lista produktów za pomocą linku Lista produktów daje rezultat – Brak danych, ponieważ każdy z komponentów typu **Stateful** przeszedł w stan nieaktywny po upływie 2 min i nastąpiła utrata danych.

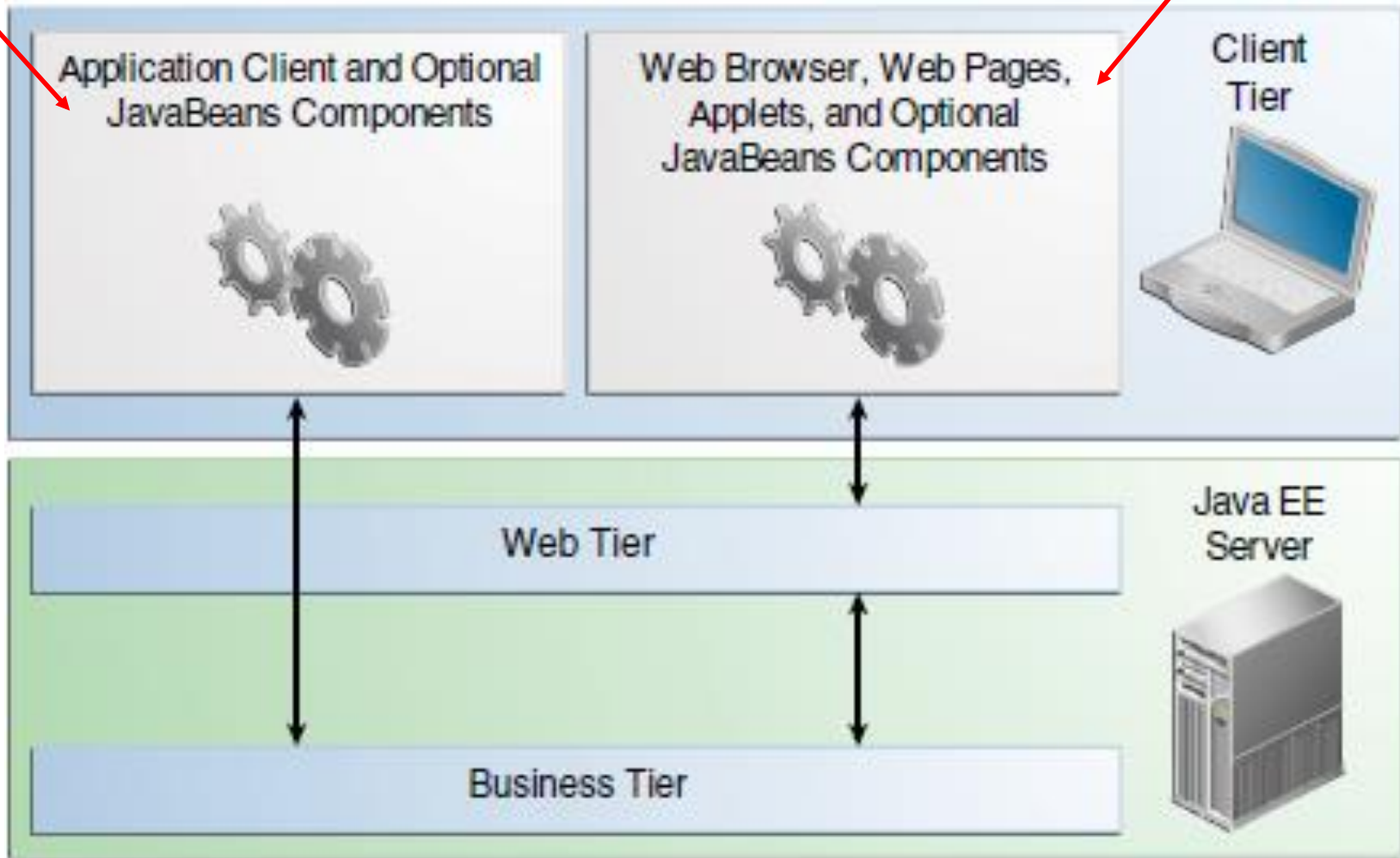


### **3. Wykonanie aplikacji wielowarstwowej na platformie Java EE z warstwą klienta zawierającą klienta internetowego i desktopowego**

# Komunikacja między warstwą klienta i serwerem aplikacji

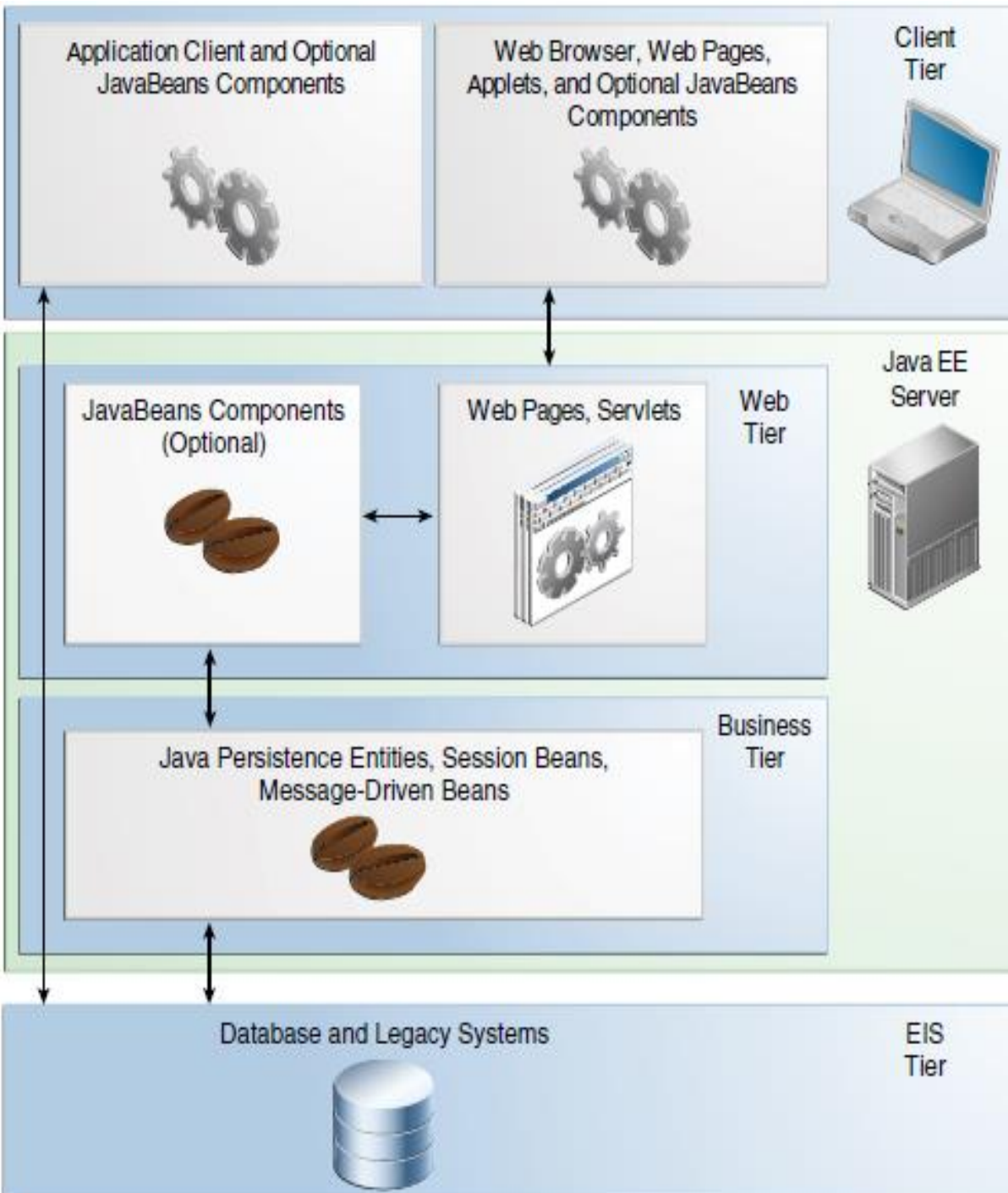
**Klient desktopowy**

**Klient internetowy**



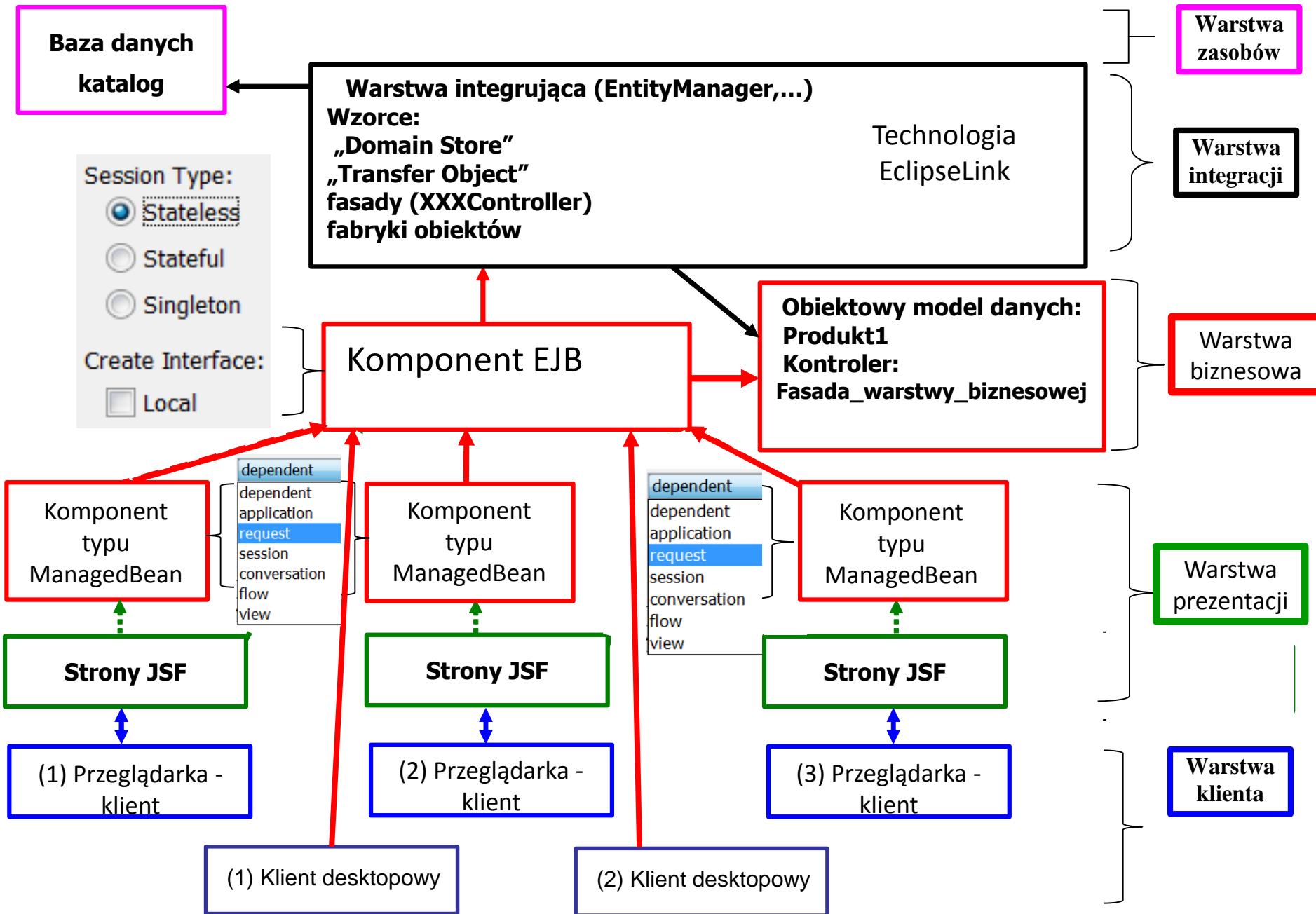


# Komponenty poszczególnych warstw aplikacji EE



Enterprise Information Systems

# Architektura aplikacji pięciowarstwowej -Java EE 7.0 JavaServer Faces



# 3.1. Wykonanie projektu typu Enterprise Application

The image shows a screenshot of the NetBeans IDE 8.2 interface. On the left is the 'File' menu, and on the right are two dialog boxes: 'New Project' and 'New Enterprise Application'.

**File Menu:**

- File
- Edit
- View
- Navigate
- Source
- Refactor
- Run
- Debug
- Project

**New Project Dialog:**

- Steps:**
  1. Choose Project
  2. ...
- Choose Project:**
  - Filter: [ ]
  - Categories: Java, JavaFX, Java Web, Java EE, HTML5/JavaScript, Java ME Embedded, Java Card, Maven
  - Projects: Enterprise Application (selected), Enterprise Application with Existing, EJB Module, EJB Module with Existing Sources, Enterprise Application Client, Enterprise Application Client with E
  - Description: Creates a new enterprise application in a standard project. You can also create an EJB module project and Web application project in the enterprise application. A standard project uses an
- Buttons: < Back, Next >, Finish, Cancel, Help

**New Enterprise Application Dialog:**

- Steps:**
  1. Choose Project
  2. Name and Location
  3. Server and Settings
- Name and Location:**
  - Project Name: SklepPK\_Lab2\_EE
  - Project Location: C:\Studia\Szkola\CalyPK\Laboratoria\lab2 [Browse...]
  - Project Folder: ..\Studia\Szkola\CalyPK\Laboratoria\lab2\SI
  - Use Dedicated Folder for Storing Libraries
  - Libraries Folder: [ ] [Browse...]
  - Text: Different users and projects can share the same compilation libraries (see Help for details).
- Buttons: < Back, Next >, Finish, Cancel, Help



## New Enterprise Application



### Steps

1. Choose Project
2. Name and Location
3. **Server and Settings**

### Server and Settings

Server:

GlassFish Server 4.1.1

Add...

Java EE Version:

Java EE 7

Create EJB Module:

SklepPK\_Lab2\_EE-ejb

Create Web Application Module:

SklepPK\_Lab2\_EE-war

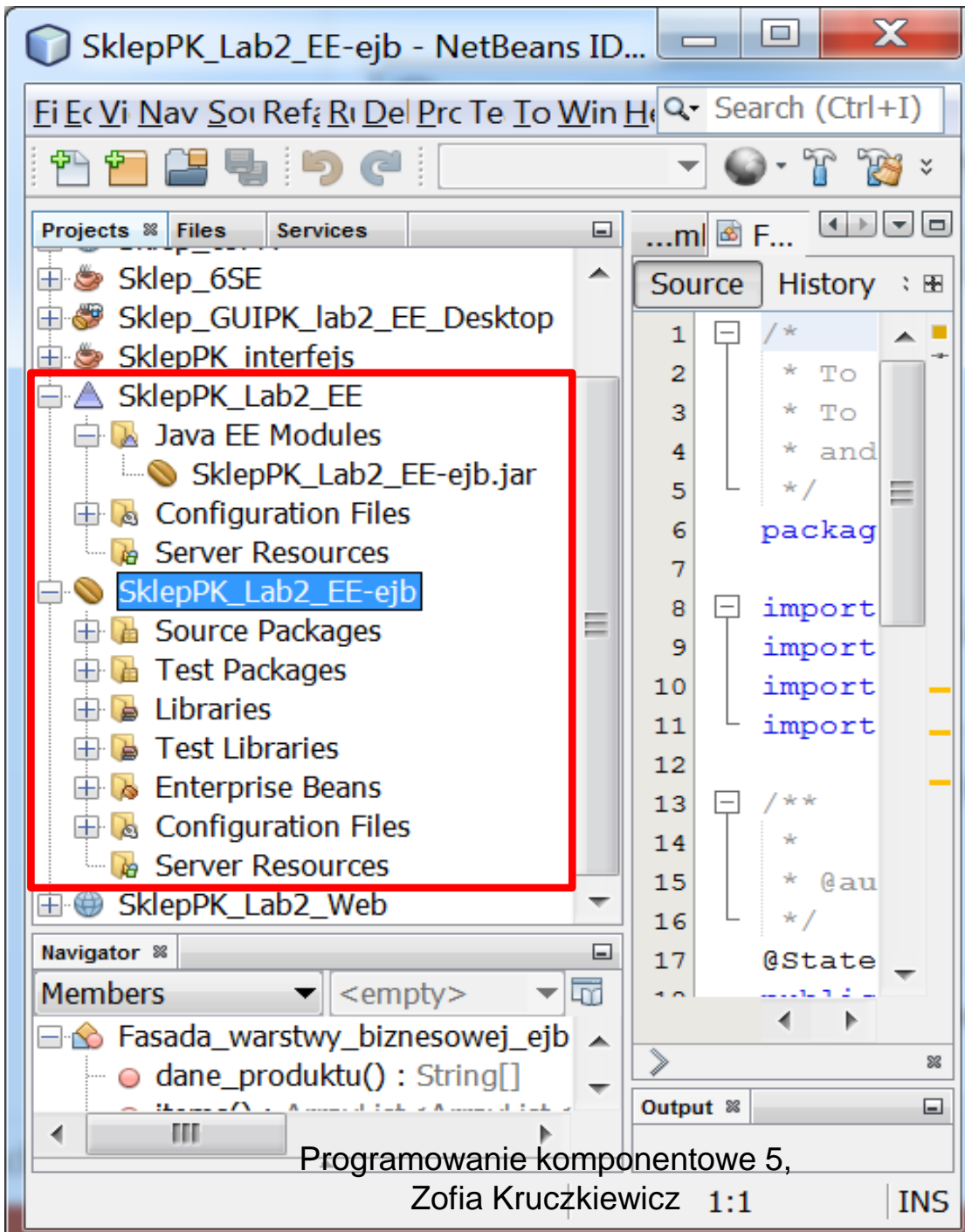
< Back

Next >

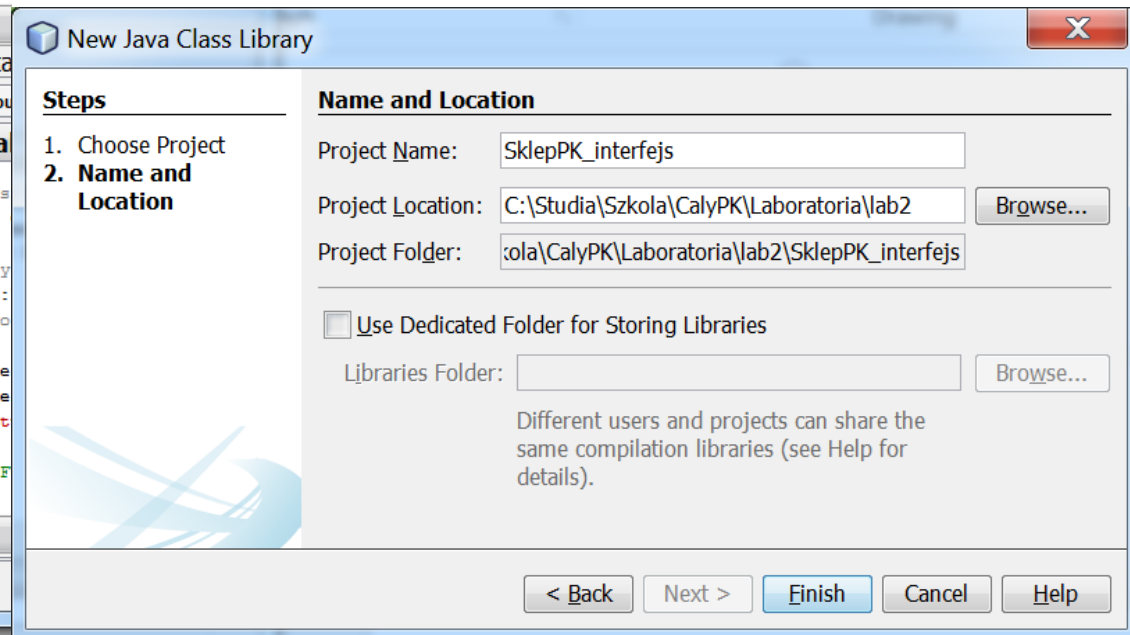
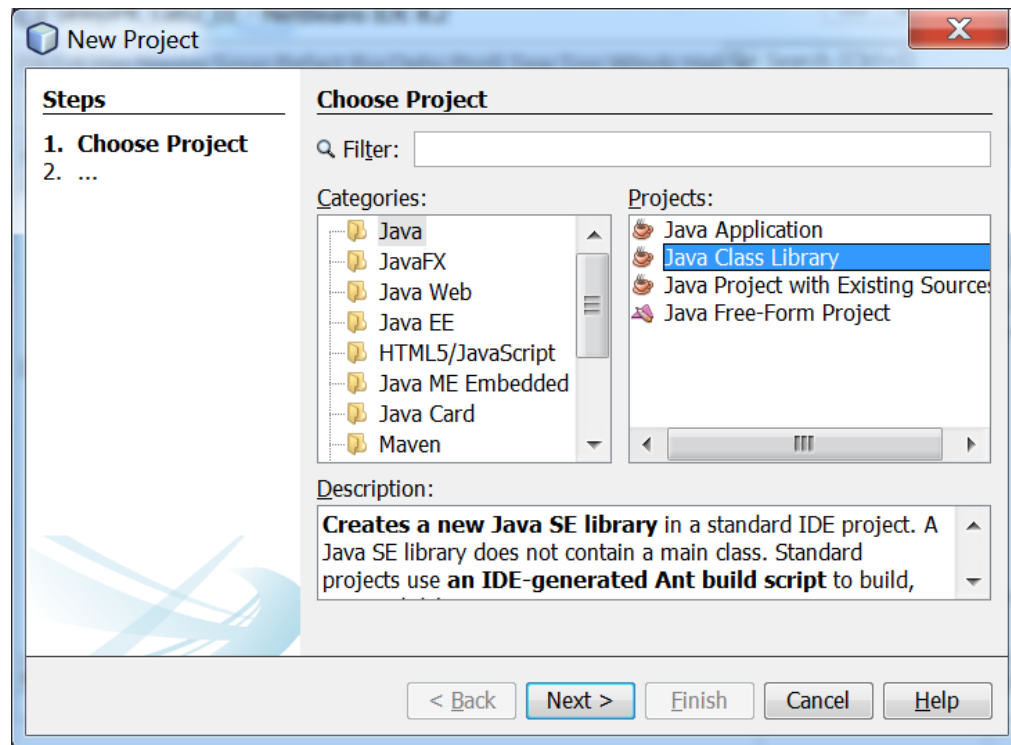
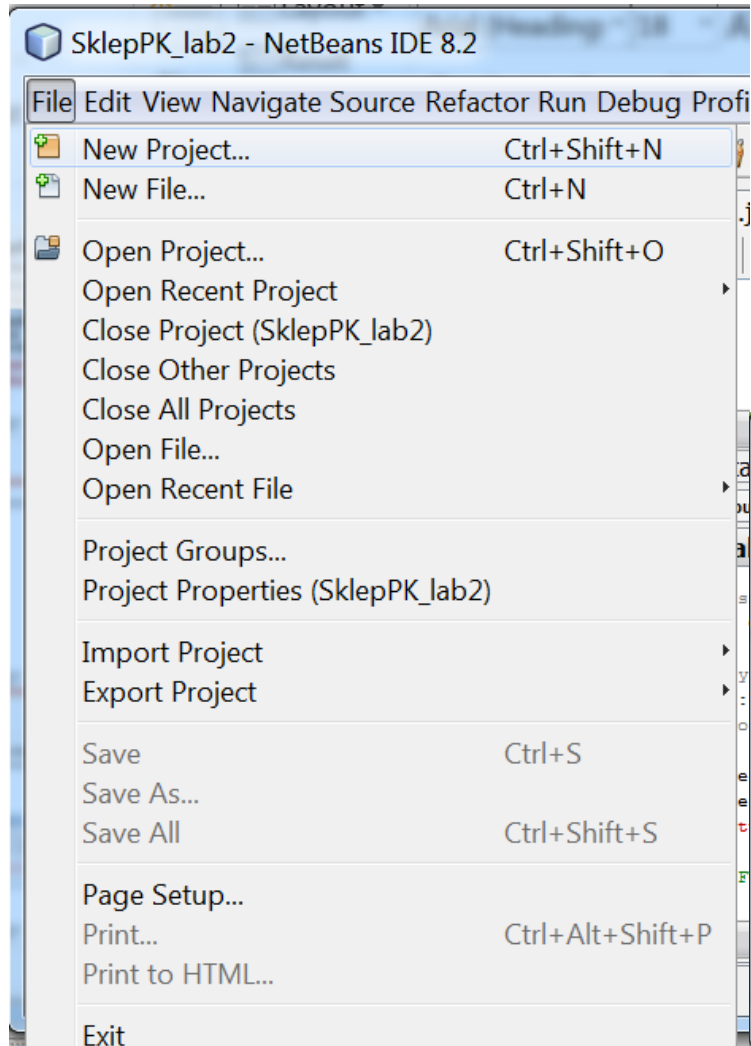
Finish

Cancel

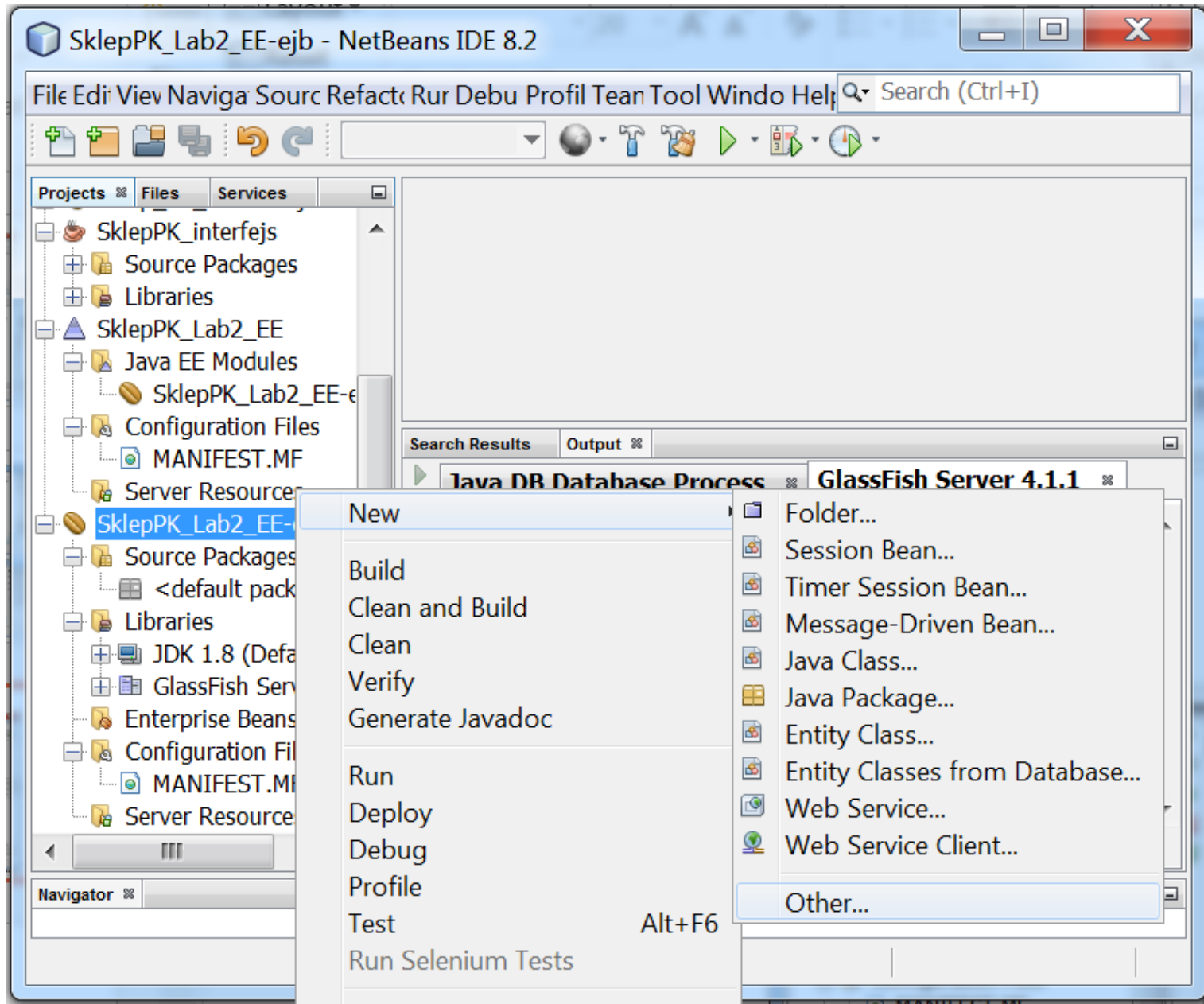
Help



### 3.2. Wykonanie projektu Java SE typu Java Class Library (SklepPK\_interfejs) w celu zdefiniowania w nim w kolejnych krokach **interfejsu komponentu EJB**.



### 3.3. Definicja komponentu EJB typu Session w module typu EJB (wybranie pozycji **New/Other**)



New File

**Steps**

1. Choose File Type
2. ...

**Choose File Type**

Project: SklepPK\_Lab2\_EE-ejb

Filter:

Categories:

- Bean Validation
- Enterprise JavaBeans
- Contexts and Dependence
- Java
- JavaBeans Objects
- Unit Tests

File Types:

- Session Bean
- Timer Session Bean
- Message-Driven Bean

Description:

Creates an empty Session Enterprise JavaBean. A session bean is typically used to encapsulate business logic and resources. This template creates a

< Back

New Session Bean

**Steps**

1. Choose File Type
2. Name and Location

**Name and Location**

EJB Name: Fasada\_warstwy\_biznesowej\_ejb

Project: SklepPK\_Lab2\_EE-ejb

Location: Source Packages

Package: warstwa\_biznesowa\_ejb

**Session Type:**

- Stateless
- Stateful
- Singleton

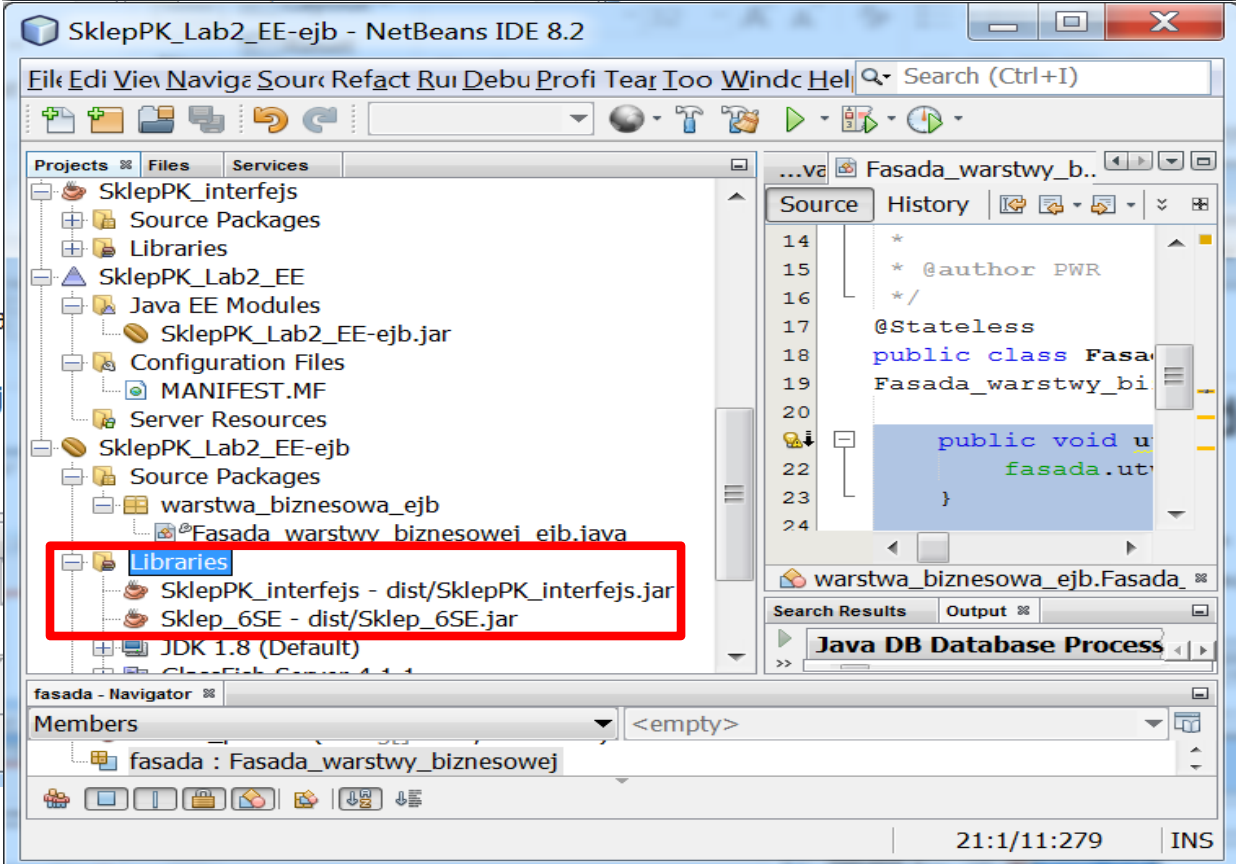
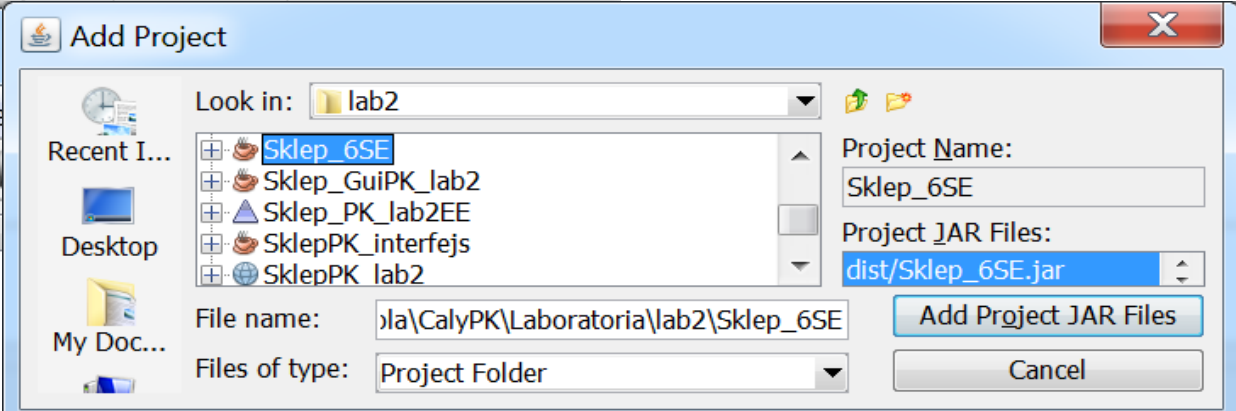
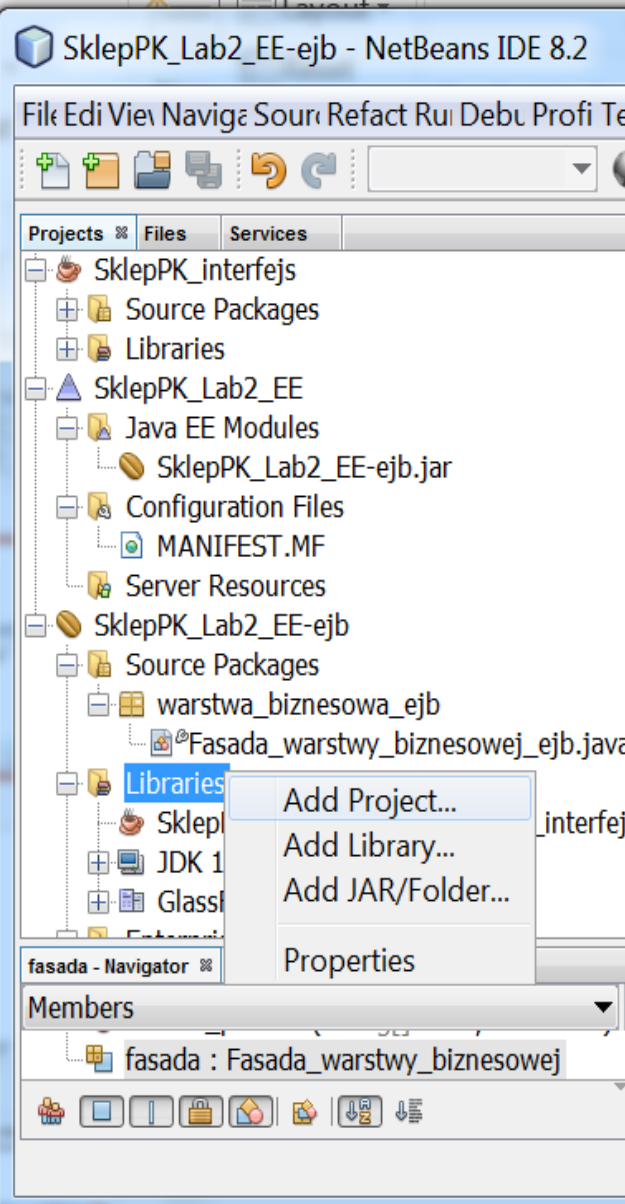
**Create Interface:**

- Local
- Remote in project: SklepPK\_interfejs

< Back   Next >   Finish   Cancel   Help



# 3.4. Dodanie projektu SE z logiką biznesową do modułu EJB



## 3.5. Zdefiniowanie w komponencie EJB typu Session Bean (Fasada\_warstwy\_biznesowej\_ejb) kodu zdalnego dostępu do logiki biznesowej

The screenshot shows the NetBeans IDE 8.2 interface. The main editor displays the source code for `Fasada_warstwy_biznesowej_ejb.java`. The code is as follows:

```
1 package warstwa_biznesowa_ejb;
2
3 import java.util.ArrayList;
4 import java.util.Date;
5 import javax.ejb.Stateless;
6 import warstwa_biznesowa.Fasada_warstwy_biznesowej;
7
8 @Stateless
9 public class Fasada_warstwy_biznesowej_ejb implements
10     Fasada_warstwy_biznesowej_ejbRemote {
11     Fasada_warstwy_biznesowej fasada = new Fasada_warstwy_biznesowej();
12
13     public void utworz_produkty(String dane[], Date data) {
14         fasada.utworz_produkty(dane, data);
15     }
16
17     public String[] dane_produkty() {
18         return fasada.dane_produkty();
19     }
20
21     public ArrayList<ArrayList<String>> items() {
22         return fasada.items();
23     }
24 }
```

The `items()` method is highlighted in blue. The IDE also shows a project tree on the left with the following structure:

- Enterprise Beans
- Configuration Files
- Server Resources
- SklepPK\_interfejs
- Source Packages
- Libraries
- SklepPK\_Lab2\_EE
  - Java EE Modules
    - SklepPK\_Lab2\_EE-ejb.jar
  - Configuration Files
  - MANIFEST.MF
  - Server Resources
  - SklepPK\_Lab2\_EE-ejb
    - Source Packages
      - warstwa\_biznesowa\_ejb
        - @Fasada\_warstwy\_biznesowej\_ejb.java
  - Libraries
    - SklepPK\_interfejs - dist/SklepPK\_interfejs.jar
    - Sklep\_6SE - dist/Sklep\_6SE.jar
    - JDK 1.8 (Default)
    - GlassFish Server 4.1.1
  - Enterprise Beans

The bottom of the IDE shows the `dane_produkty - Navigator` window with the following members:

- Members: `<empty>`
- Members list: `dane_produkty() : String[]`

The status bar at the bottom right shows the time `19:1` and the mode `INS`.

# Fasada\_warstwy\_biznesowej\_ejb - kod

```
package warstwa_biznesowa;
```

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

```
import java.util.Date;
```

```
import javax.ejb.Stateless;
```

```
import warstwa_biznesowa.Fasada_warstwy_biznesowej;
```

```
@Stateless
```

```
public class Fasada_warstwy_biznesowej_ejb implements  
        Fasada_warstwy_biznesowej_ejbRemote {
```

```
Fasada_warstwy_biznesowej fasada=new Fasada_warstwy_biznesowej();
```

```
public void utworz_produkt(String dane[], Date data) {
```

```
    fasada.utworz_produkt(dane, data);
```

```
}
```

```
public String[] dane_produktu() {
```

```
    return fasada.dane_produktu(); }
```

```
public ArrayList<ArrayList<String>> items() {
```

```
    return fasada.items(); }
```

```
}
```

# Kod interfejsu – dla zdalnego komponentu typu EJB

The screenshot displays the NetBeans IDE 8.2 interface. The main editor window shows the source code for the file `Fasada_warstwy_biznesowej_ejbRemote.java`. The code is as follows:

```
1 |  
2 | package warstwa_biznesowa_ejb;  
3 |  
4 | import java.util.ArrayList;  
5 | import java.util.Date;  
6 | import javax.ejb.Remote;  
7 |  
8 | @Remote  
9 | public interface Fasada_warstwy_biznesowej_ejbRemote {  
10 |     public void utworz_produkt(String dane[], Date data);  
11 |     public String[] dane_produktu();  
12 |     public ArrayList<ArrayList<String>> items();  
13 | }  
14 |
```

The code is enclosed in a red rectangular box. The IDE's Project Explorer on the left shows the project structure, including the package `warstwa_biznesowa_ejb` and the interface `Fasada_warstwy_biznesowej_ejbRemote`. The bottom status bar shows the current cursor position at line 1, column 1.

# Fasada\_warstwy\_biznesowej\_ejbRemote – kod interfejsu

```
package warstwa_biznesowa_ejb;
```

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

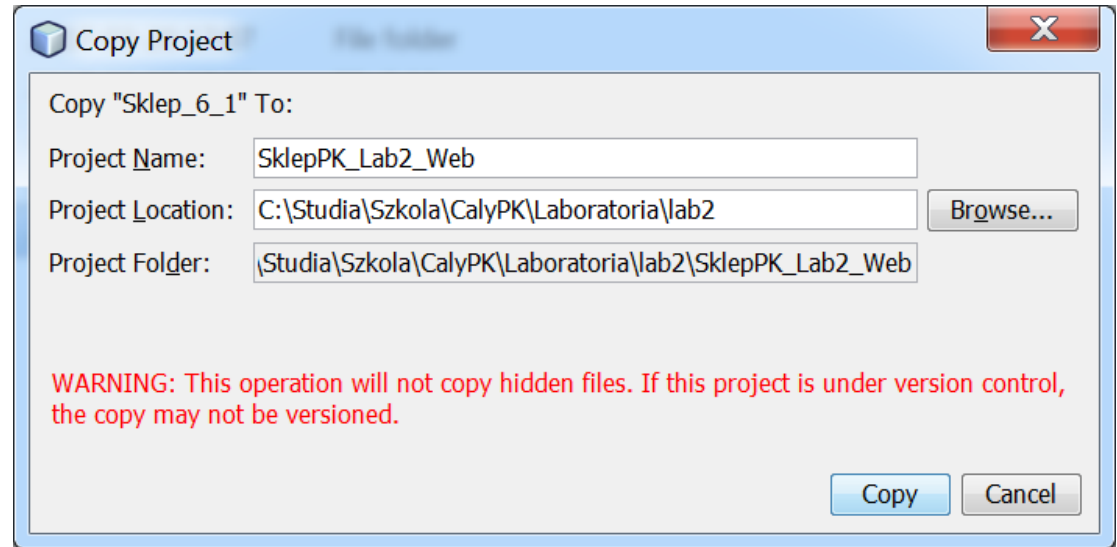
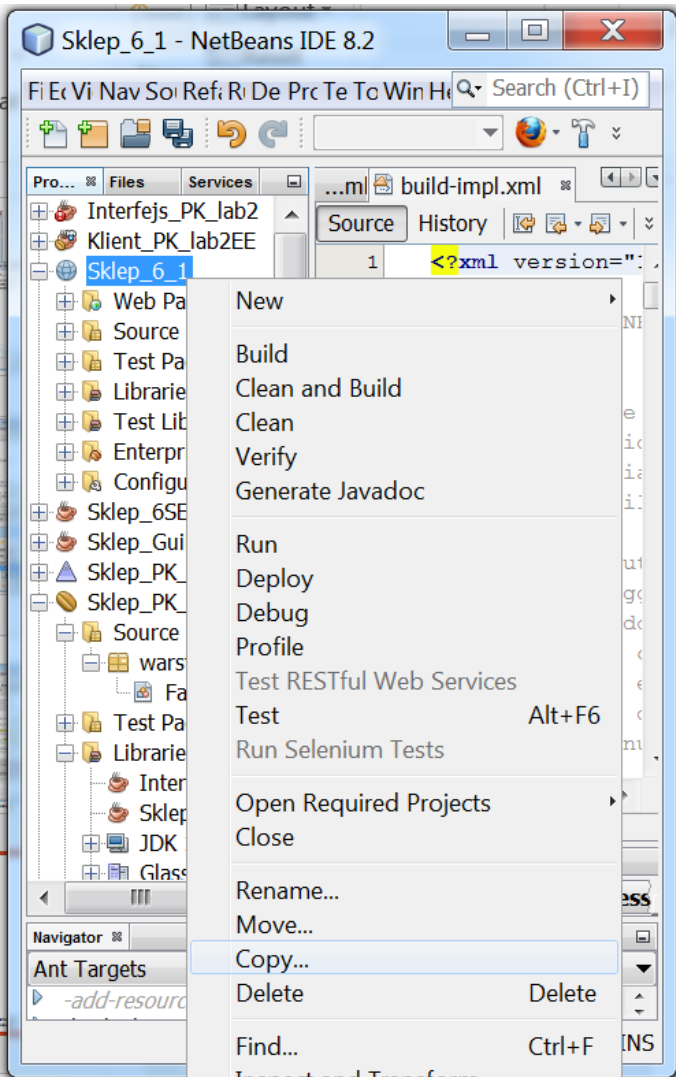
```
import java.util.Date;
```

```
import javax.ejb.Remote;
```

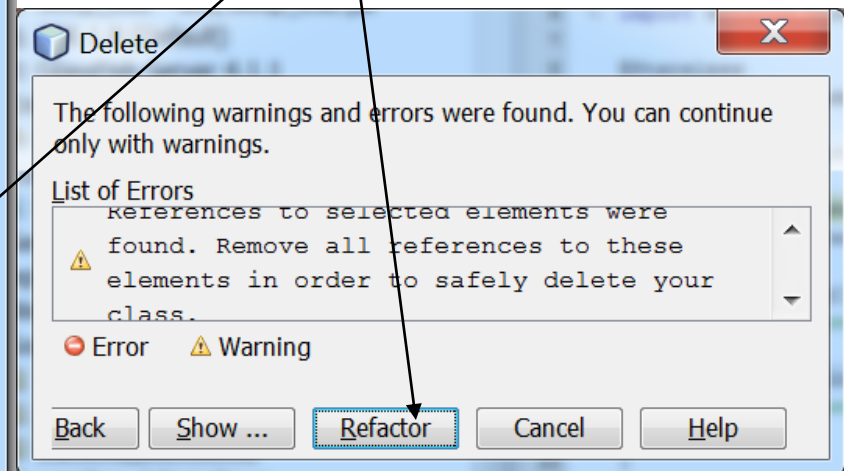
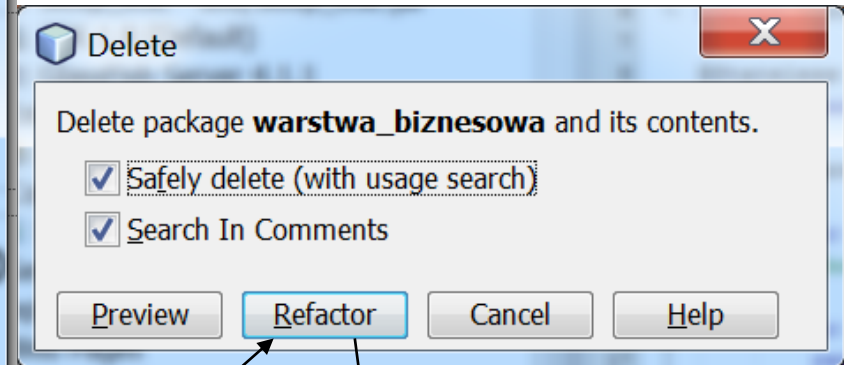
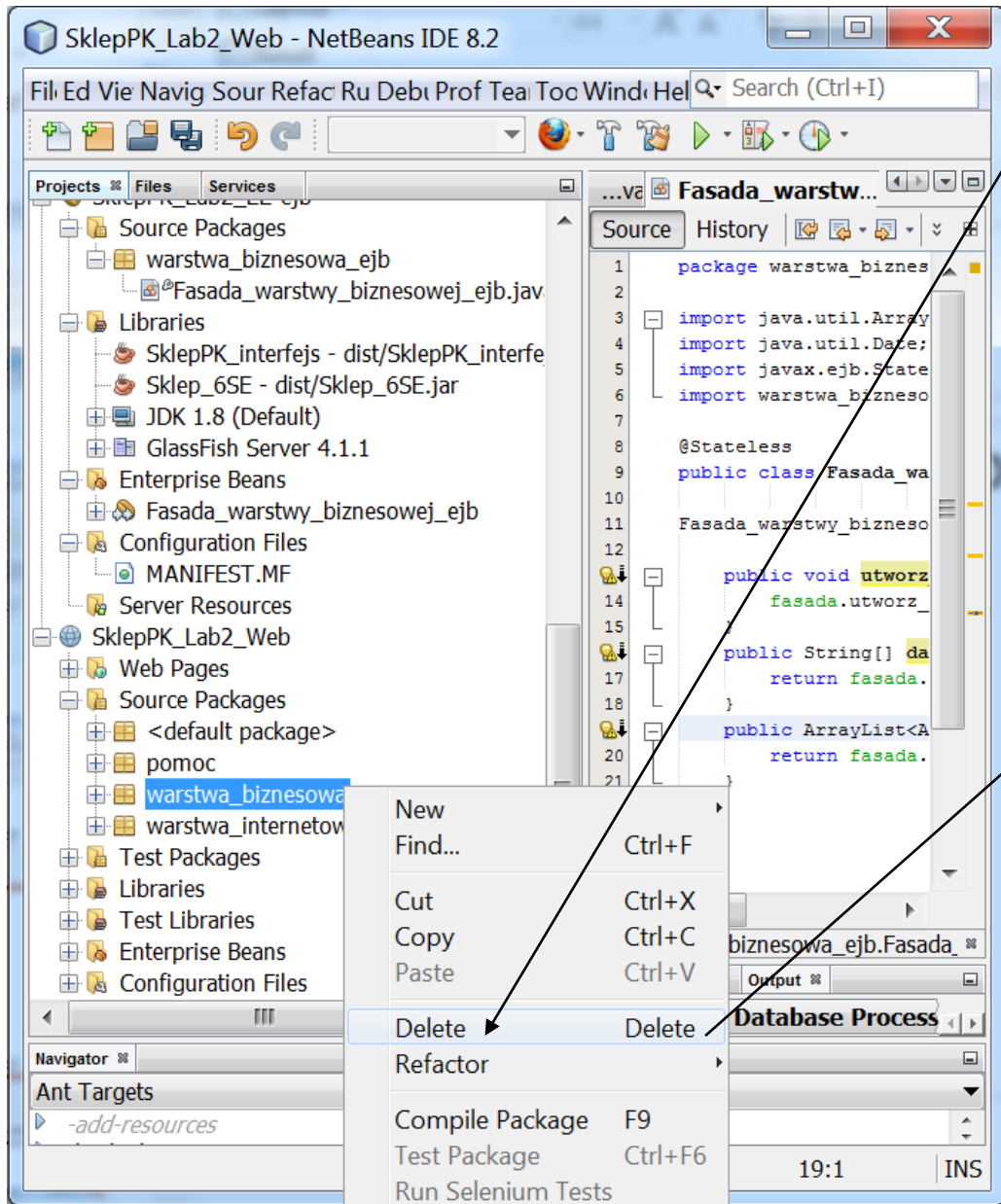
```
@Remote
```

```
public interface Fasada_warstwy_biznesowej_ejbRemote {  
    public void utworz_produkt(String dane[], Date data);  
    public String[] dane_produktu();  
    public ArrayList<ArrayList<String>> items();  
}
```

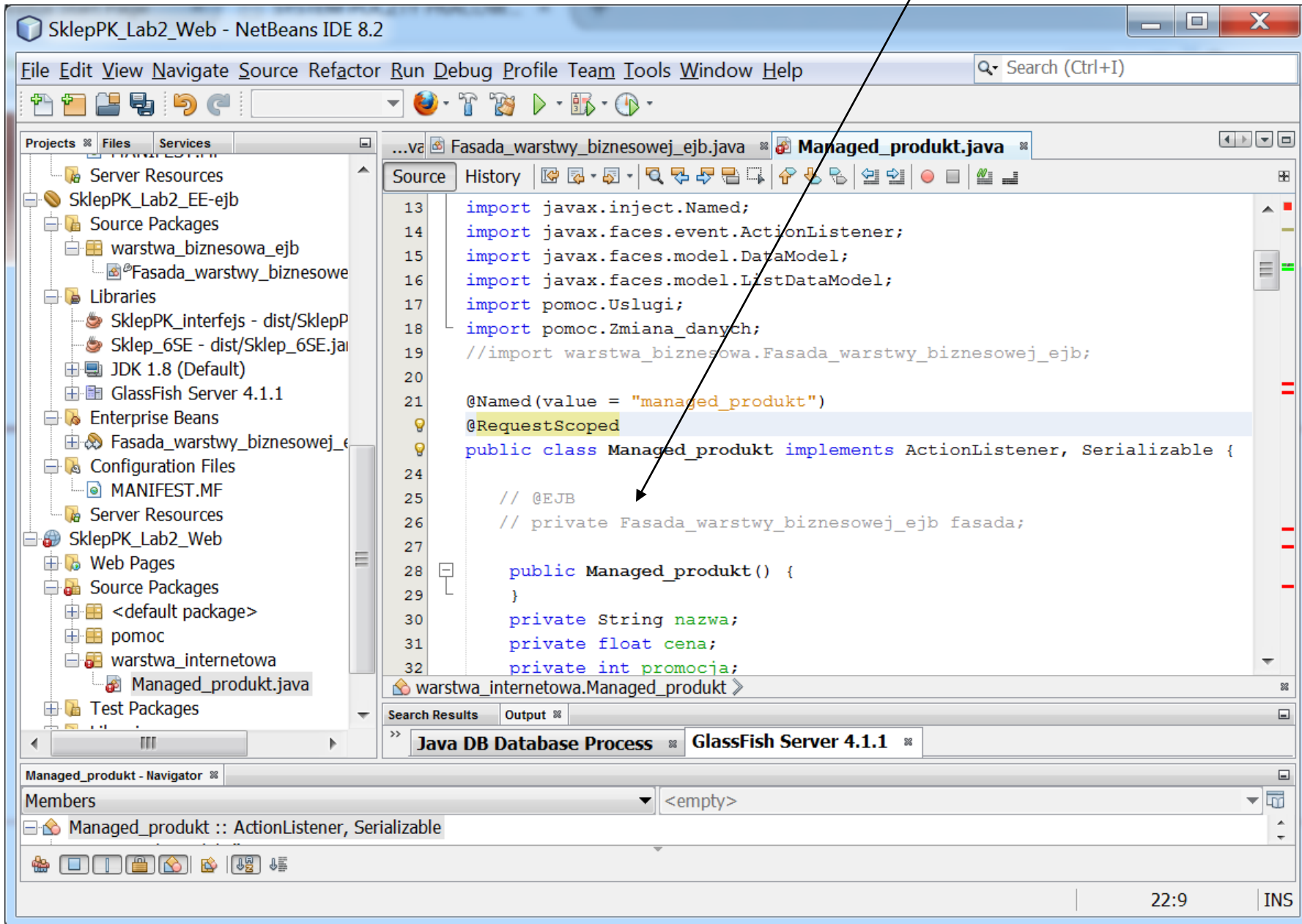
## 3.6. Dodanie do projektu EE warstwy klienta internetowego – kopia istniejącej aplikacji Java Web Application



### 3.6.1. W projekcie typu Java Web typu **Web Application** bezpiecznie pakiet **warstwa\_biznesowa** – **teraz komponent EJB jest zdefiniowany w module EJB**



3.6.2. W komponencie typu Managed Bean o nazwie **Managed\_produk**t należy usunąć z kodu dotychczasowe powiązanie z komponentem EJB.



The screenshot shows the NetBeans IDE 8.2 interface. The main editor displays the source code for `Managed_produk.java`. The code includes the following annotations and class definition:

```
13 import javax.inject.Named;
14 import javax.faces.event.ActionListener;
15 import javax.faces.model.DataModel;
16 import javax.faces.model.ListDataModel;
17 import pomoc.Uslugi;
18 import pomoc.Zmiana_danych;
19 //import warstwa_biznesowa.Fasada_warstwy_biznesowej_ejb;
20
21 @Named(value = "managed_produk")
22 @RequestScoped
23 public class Managed_produk implements ActionListener, Serializable {
24
25     // @EJB
26     // private Fasada_warstwy_biznesowej_ejb fasada;
27
28     public Managed_produk() {
29     }
30     private String nazwa;
31     private float cena;
32     private int promocja;
```

An arrow points from the text above to the `@EJB` annotation on line 25, which is currently commented out. The IDE also shows a project tree on the left, a search results panel at the bottom, and a status bar at the bottom right indicating the time is 22:9 and the mode is INS.



3.6.3. Należy teraz wstawić powiązanie z nowym komponentem typu EJB. W tym celu kliknąć prawym klawiszem myszy na kod klasy i wybrać pozycję **Insert Code...**

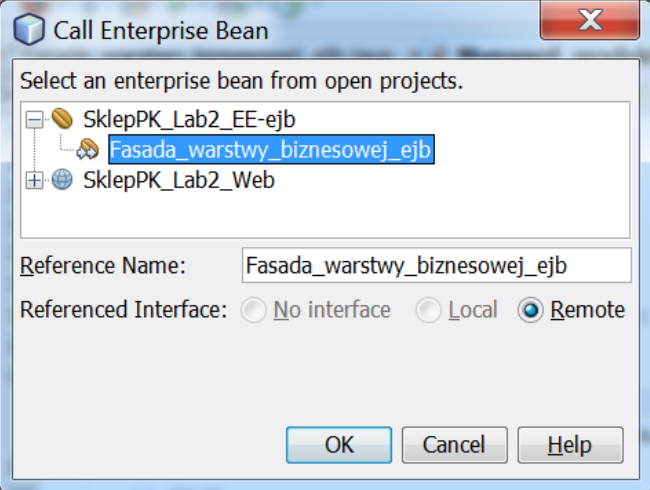
The screenshot shows the NetBeans IDE 8.2 interface. The main editor window displays the code for `Managed_produkkt.java`. A context menu is open over the code, listing various actions. The `Insert Code...` option is highlighted in blue. An arrow points from the text above to this option. The menu items and their shortcuts are:

- Navigate
- Show Javadoc (Alt+F1)
- Find Usages (Alt+F7)
- Call Hierarchy
- Insert Code...** (Alt+Insert)
- Fix Imports (Ctrl+Shift+I)
- Refactor
- Format (Alt+Shift+F)
- Run File (Shift+F6)
- Debug File (Ctrl+Shift+F5)
- Test File (Ctrl+F6)
- Debug Test File (Ctrl+Shift+F6)
- Run Focused Test Method
- Debug Focused Test Method
- Run Into Method
- New Watch... (Ctrl+Shift+F7)
- Toggle Line Breakpoint (Ctrl+F8)
- Profile

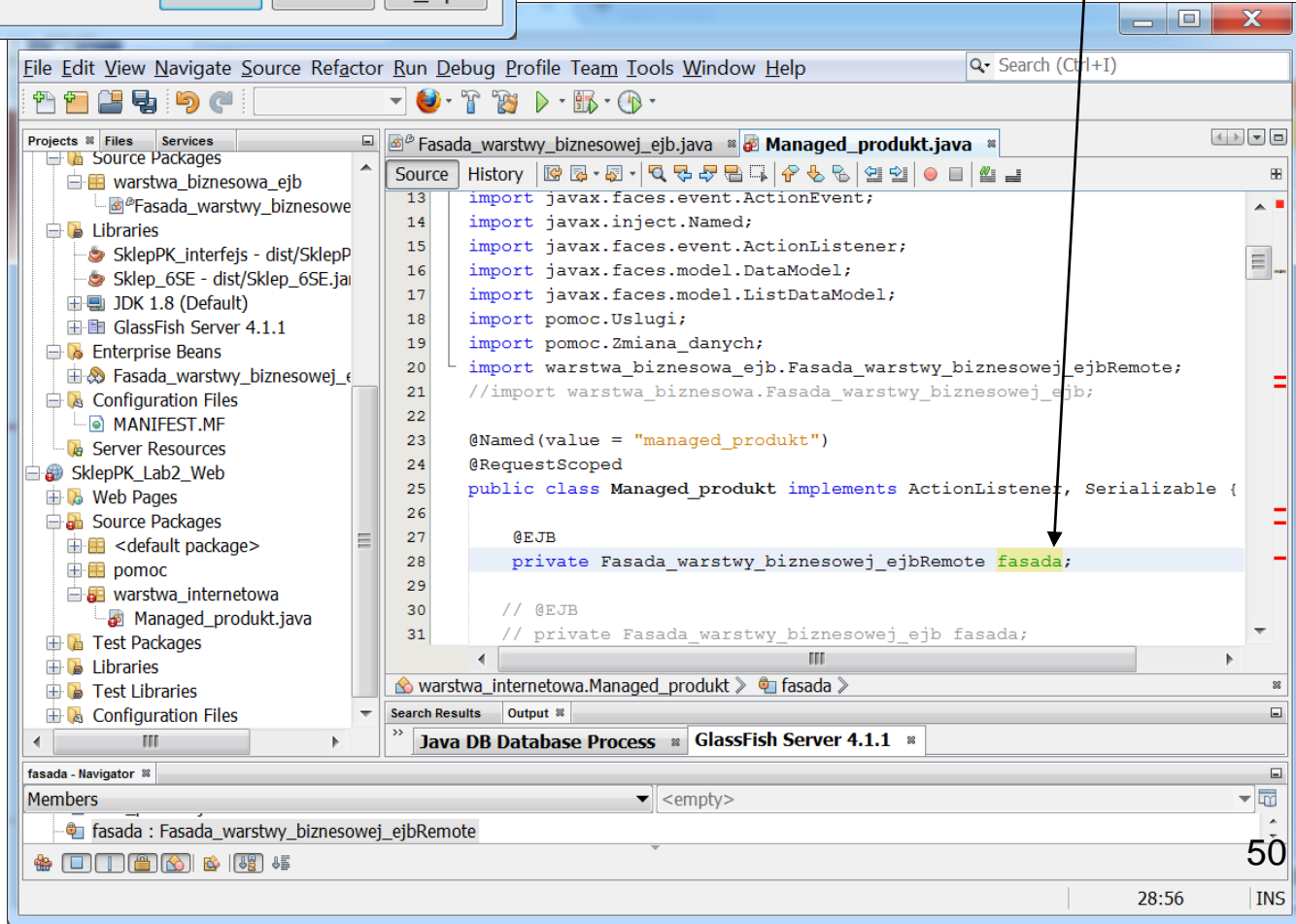
The code in the editor includes several imports and a comment:

```
13 import javax.inject.Named;
14 import javax.faces.event.ActionListener;
15 import javax.faces.model.DataModel;
16 import javax.faces.model.ListDataModel;
17 import pomoc.Uslugi;
18 import pomoc.Zmiana_danych;
19 //import warstwa_biznesowa.Fasada_warstwy_biznesowej_ejb;
```

- Generate
- Constructor...
- Logger...
- Getter...
- equals() and hashCode()...
- toString()...
- Delegate Method...
- Override Method...
- Add Property...
- Call Enterprise Bean...
- Use Database...
- Send JMS Message...
- Send E-mail...
- Call Web Service Operation...
- Generate REST Client...



3.6.3.cd. Należy wybrać pozycję **Call Enterprise Bean..** i w kolejnym formularzu należy wybrać w projekcie EJB **SklepPK\_Lab2\_EE-ejb** komponent EJB typu Session: **Fasada\_warstwy\_biznesowej\_ejbRemote** i nadać mu nazwę **fasada**.



### 3.6.4. Wynik zmian – w celu usunięcia błędów należy zmodyfikować metody dostępu do dodanego komponentu: **Fasada\_warstwy\_biznesowej\_ejbRemote**

SklepPK\_Lab2\_Web - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

Projects Files Services

Fasada\_warstwy\_biznesowe

- Test Packages
- Libraries
  - SklepPK\_interfejs - dist/SklepP
  - Sklep\_6SE - dist/Sklep\_6SE.jar
- JDK 1.8 (Default)
- GlassFish Server 4.1.1
- Test Libraries
- Enterprise Beans
- Fasada\_warstwy\_biznesowej\_e
- Configuration Files
- MANIFEST.MF
- Server Resources

SklepPK\_Lab2\_Web

- Web Pages
- Source Packages
  - <default package>
  - pomoc
  - warstwa\_internetowa
    - Managed\_produkkt.java
- Test Packages
- Libraries
- Test Libraries
- Configuration Files

Managed\_produkkt.java

```
18 import pomoc.Uslugi;
19 import pomoc.Zmiana_danych;
20 import warstwa_biznesowa_ejb.Fasada_warstwy_biznesowej_ejbRemote;
21 //import warstwa_biznesowa.Fasada_warstwy_biznesowej_ejb;
22
23 @Named(value = "managed_produkkt")
24 @RequestScoped
25 public class Managed_produkkt implements ActionListener, Serializable
26
27     @EJB
28     private Fasada_warstwy_biznesowej_ejbRemote fasada;
29     // @EJB
30     // private Fasada_warstwy_biznesowej_ejb fasada;
31     public Fasada_warstwy_biznesowej_ejbRemote getFasada() {
32         return fasada;
33     }
34     public void setFasada(Fasada_warstwy_biznesowej_ejbRemote fasada) {
35         this.fasada = fasada;
36     }
```

warstwa\_internetowa.Managed\_produkkt > setFasada >

Search Results Output

Java DB Database Process GlassFish Server 4.1.1

setFasada - Navigator

Members

<empty>

- setFasada(Fasada\_warstwy\_biznesowej\_ejbRemote fasada)

34:2 INS

### 3.7. Dodanie warstwy klienta desktopowego Java EE typu Enterprise Application Client. Pierwszym etapem jest utworzenie projektu na platformie Java SE

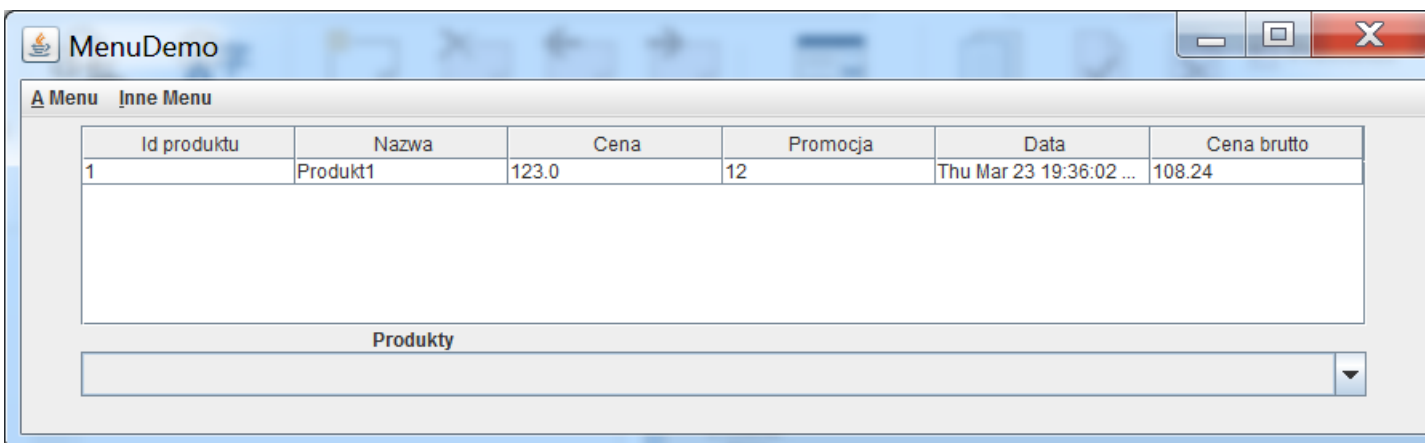
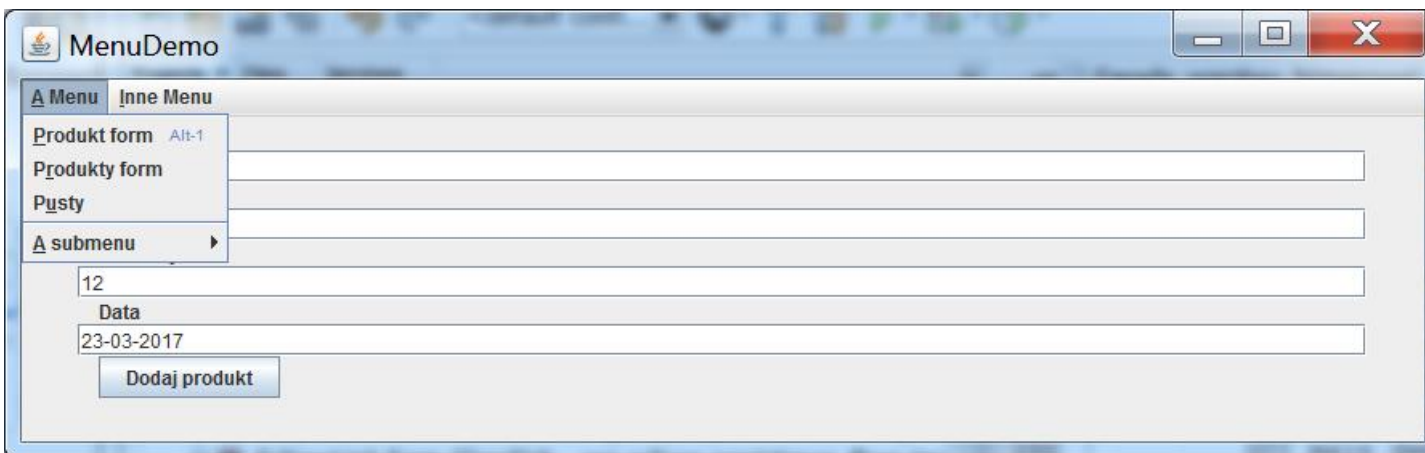
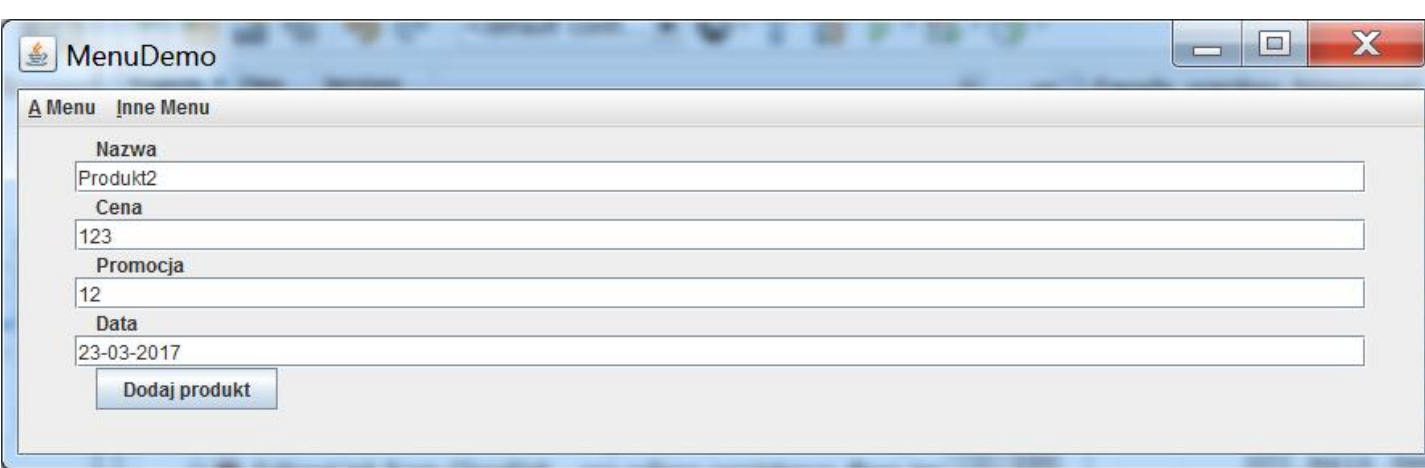
([http://zofia.kruczkiewicz.staff.iiar.pwr.wroc.pl/wyklady/pojava/Lab6\\_2016.pdf](http://zofia.kruczkiewicz.staff.iiar.pwr.wroc.pl/wyklady/pojava/Lab6_2016.pdf)).

Logika biznesowa dla projektu **Sklep\_GuiPK\_lab2\_SE** stanowiącego warstwę klienta w dwuwarstwowej aplikacji Java SE jest projekt **Sklep\_6SE** – z kodem logiki biznesowej (p.3.4 - połączonej podobnie jak z modułem EJB)

The screenshot shows an IDE interface with the following components:

- Project Explorer (Left):** Displays a project named **Sklep\_6SE** with sub-packages **warstwa\_biznesowa** (containing **Fasada\_warstwy\_biznesowej.java** and **Produkt1.java**) and **Libraries**. Below it is another project **Sklep\_GuiPK\_lab2\_SE** with sub-packages **sklep\_gui** (containing **GUI\_main.java**, **Produkt\_form.java**, **Produkty\_form.java**, and **Pusty\_form.java**) and **Libraries**. The file **Sklep\_6SE - dist/Sklep\_6SE.jar** is highlighted with a red box.
- Code Editor (Center):** Shows the source code of **GUI\_main.java**. The code includes a `createAndShowGUI()` method and a `main()` method. The `main()` method creates a `JFrame` and starts a `Runnable` thread that calls `createAndShowGUI()`.

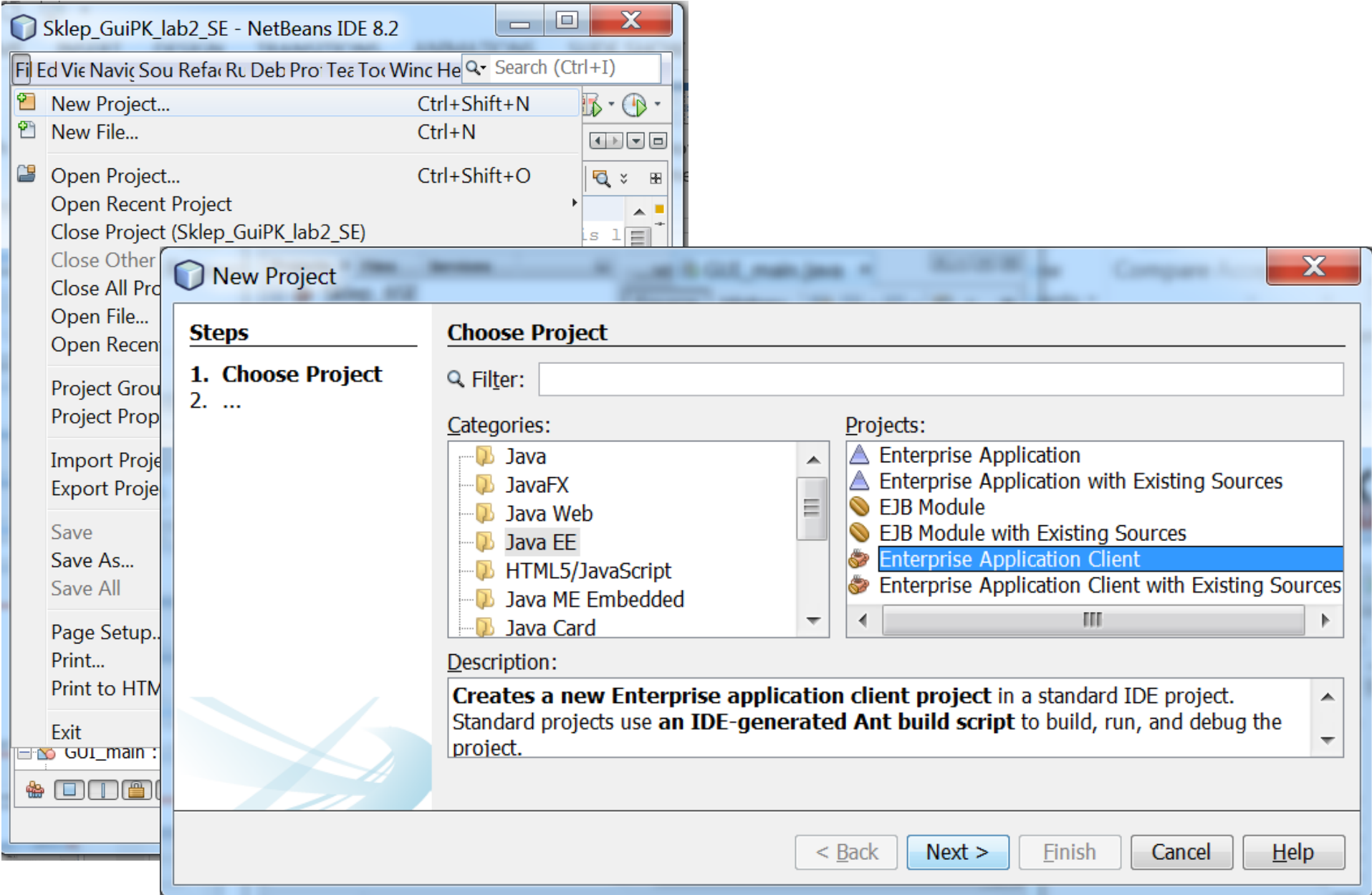
```
142     }
143     }
144
145     private static void createAndShowGUI() { //metoda tw
146         // obiekt typu JMenuBar utworzony w metodzie createMenuBar
147         JFrame frame = new JFrame("MenuDemo");
148         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
149         frame.setSize(900, 400);
150         GUI_main demo = new GUI_main();
151         frame.setJMenuBar(demo.createMenuBar()); //dodanie k
152         frame.setContentPane(demo.createContentPane());
153         frame.setVisible(true); //wyswietlenie okna
154     }
155
156     public static void main(String[] args) {
157         //utworzenie wątku zarządzającego zdarzeniami utworzonego GUI
158
159         java.awt.EventQueue.invokeLater(new Runnable() {
160             @Override
161             public void run() {
162                 createAndShowGUI();
163             }
164         });
165     }
166
```
- Taskbar (Bottom):** Shows running processes: **Java DB Database Process**, **GlassFish Server 4.1.1**, and **Sklep\_GuiPK\_lab2 (clean)**.
- Navigator (Bottom):** Shows the class **GUI\_main :: ActionListener**.



**3.7.1.  
Wykonanie i uruchomienie projektu Java SE o architekturze dwuwarstwowej.**

**Takie podejście ułatwia wdrożenie desktopowego klienta aplikacji EE.**

## 3.7.2. Drugi etap dodania warstwy klienta desktopowego Java EE typu **Enterprise Application Client**.



New Enterprise Application Client

**Steps**

1. Choose Project
- 2. Name and Location**
3. Server and Settings

**Name and Location**

Project Name: Sklep\_GUIPK\_lab2\_EE\_Desktop

Project Location: C:\Studia\Szkola\CalyPK\Laboratoria\lab2

Project Folder: Szkola\CalyPK\Laboratoria\lab2\Sklep\_GUIPK\_lab2\_EE\_Desktop

Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

New Enterprise Application Client

**Steps**

1. Choose Project
2. Name and Location
- 3. Server and Settings**

**Server and Settings**

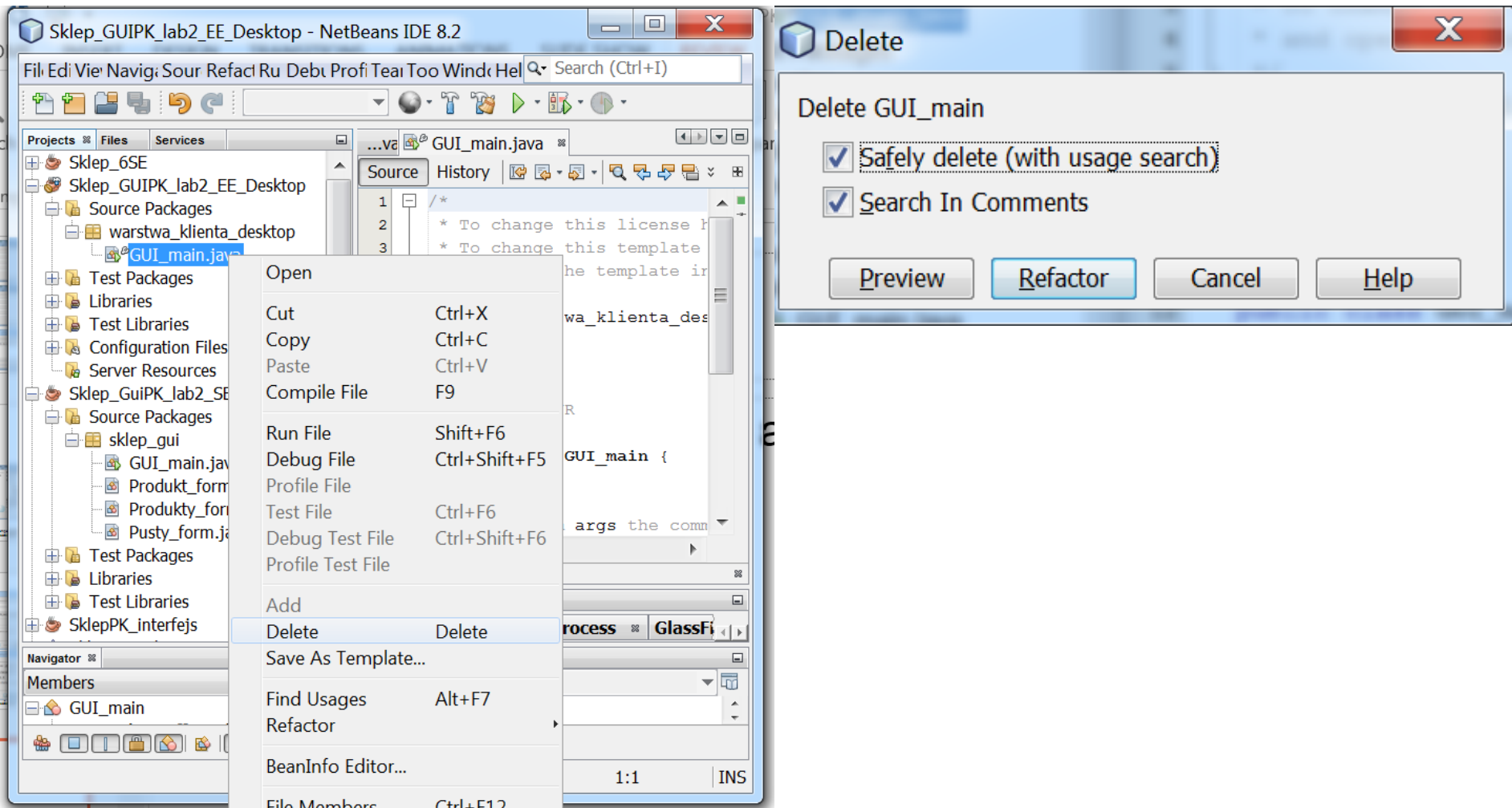
Add to Enterprise Application: <None>

Server: GlassFish Server 4.1.1

Java EE Version: Java EE 7

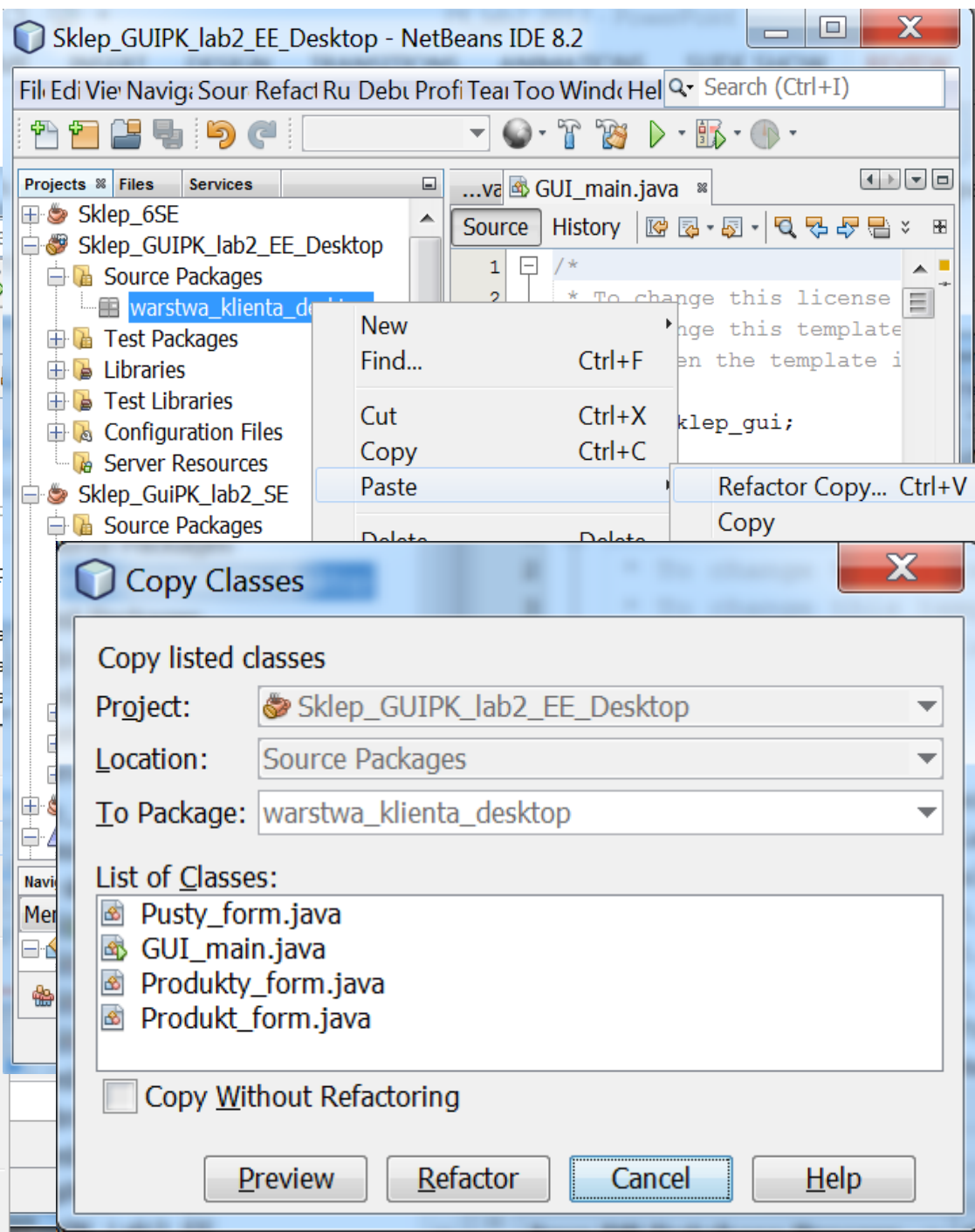
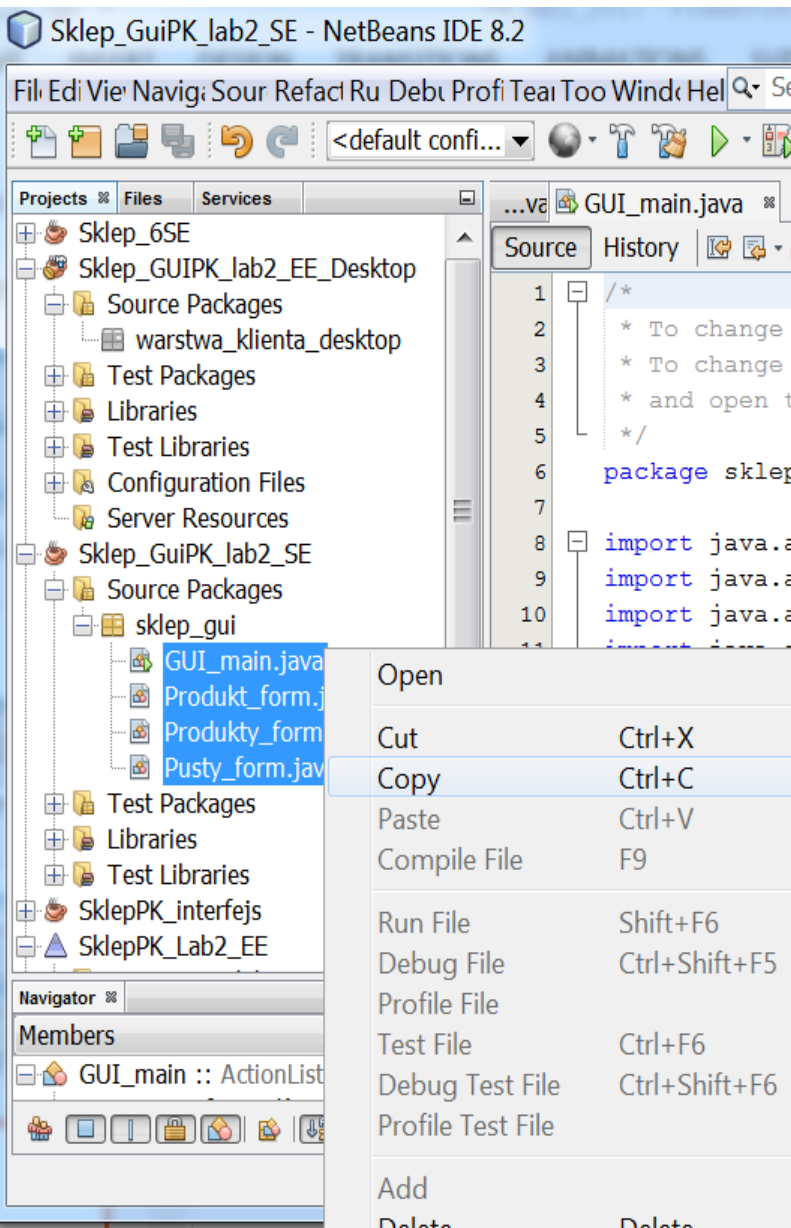
Main Class: warstwa\_klienta\_desktop.GUI\_main

### 3.7.3. Usunięcie pliku GUI\_main





### 3.7.4. Skopiowanie plików z formularzami z projektu klienta SE: Sklep\_GuiPK\_lab2\_SE.



### 3.7.5. Należy w pliku GUI\_main dodać połączenie z komponentem EJB typu Session: **Fasada\_warstwy\_biznesowej\_ejb**, umieszczonym w module EJB

The screenshot shows the NetBeans IDE 8.2 interface. The main editor window displays the source code of `GUI_main.java`. The code includes imports for `javax.swing.KeyStroke` and `warstwa_biznesowa.Fasada_warstwy_biznesowej`. The `GUI_main` class implements `ActionListener` and contains several static fields and methods. A context menu is open over the line `fasade = new Fasada_warstwy_biznesowej();`, listing various actions such as 'Navigate', 'Show Javadoc', 'Insert Code...', 'Run File', and 'Cut'. The 'Insert Code...' option is highlighted. The left sidebar shows the project structure for 'Sklep\_GUIPK\_lab2\_EE\_Desktop', with the 'warstwa\_klienta' package expanded to show 'GUI\_main.java'. The bottom status bar indicates the cursor is at line 39, column 7, with the 'INS' key highlighted.

```
19 import javax.swing.KeyStroke;
20 //import warstwa_biznesowa.Fasada_warstwy_biznesowej;
21 public class GUI_main implements ActionListener {
22
23     static JPanel cards; //panel, który posiada obiekt typu CardLayout, kt
24     //formularzy: Produkt_form do wprowadzania danych produktu
25     //oraz Produkty_form do wyświetlania danych o produktach
26
27     static CardLayout cl; //kolekcja paneli typu JPanel
28
29     //utworzenie 4 paneli reprezentujących formularze aplikacji
30
31     //pusty formularz
32     form(); //panel formularza do wsta
33     y_form();
34
35     form";
36     y_form";
37
38
39     fasade = new Fasada_warstwy_biznesowej();
40
41
42     getFacade() {
43
44
```

- Generate
- Constructor...
- Logger...
- Getter...
- Setter...
- Getter and Setter...
- toString()...
- Delegate Method...
- Override Method...
- Add Property...
- Call Enterprise Bean...**
- Use Database...
- Send JMS Message...
- Send E-mail...
- Call Web Service Operation...
- Generate REST Client...

### Call Enterprise Bean

Select an enterprise bean from open projects.

- SklepPK\_Lab2\_EE-ejb
  - Fasada\_warstwy\_biznesowej\_ejb**
  - SklepPK\_Lab2\_Web

Reference Name:

Referenced Interface:  No interface  Local  Remote

OK Cancel Help

The screenshot shows an IDE window with the following content:

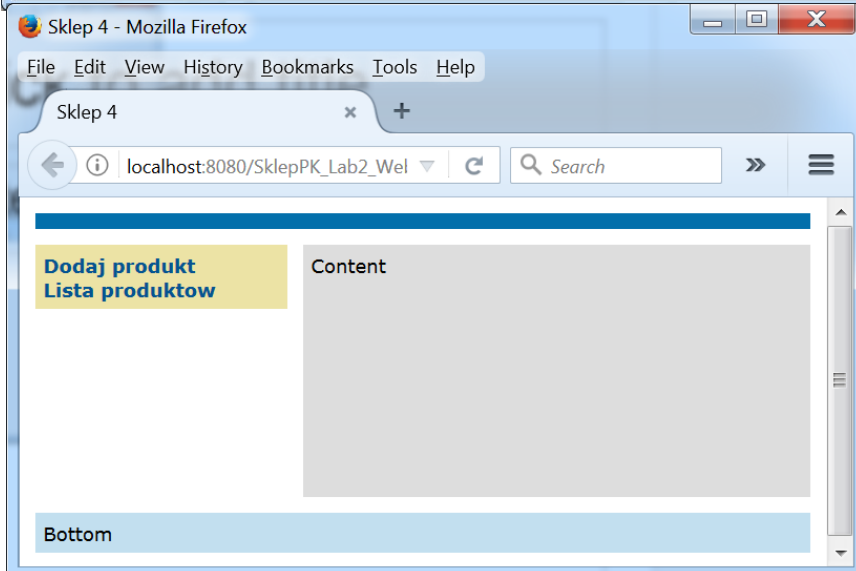
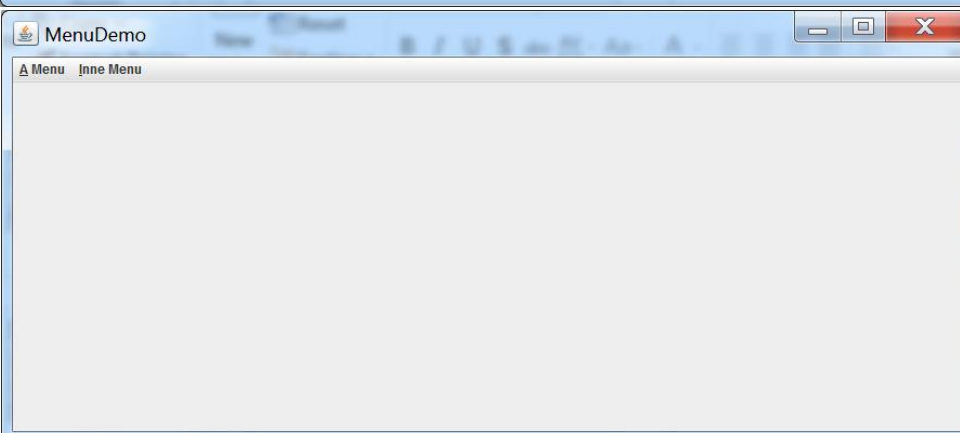
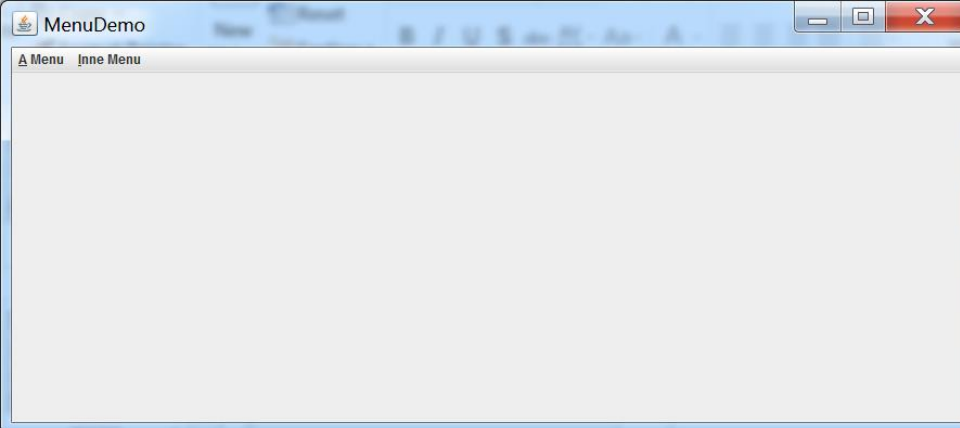
- Project Explorer:**
  - Sklep\_6SE
  - Sklep\_GUIPK\_lab2\_EE\_Desktop
    - Source Packages
      - warstwa\_klienta\_desktop
        - @GUI\_main.java
        - @Produkt\_form.java
        - @Produkty\_form.java
        - @Pusty\_form.java
    - Test Packages
    - Libraries
    - Test Libraries
    - Configuration Files
    - Server Resources
    - SklepPK\_interfejs
    - SklepPK\_Lab2\_EE
    - SklepPK\_Lab2\_EE-ejb
    - SklepPK\_Lab2\_Web

- Source Editor:**

```

25  @EJB
26  private static Fasada_warstwy_biznesowej_ejbRemote fasada;
27
28  static JPanel cards;           //panel, który posiada obiekt typu Ca
29  //formularzy: Produkt_form do wprowadzania danych produktu
30  //oraz Produkty_form do wyswietlania danych o produktach
31
32  static CardLayout cl;         //kolekcja paneli typu JPanel
33
34  //utworzenie 4 paneli reprezentujących formularze aplikacji
35  static Pusty_form card0 = new Pusty_form();           //pusty formu
36  static Produkt_form card1 = new Produkt_form();       //panel formu
37  static Produkty_form card2 = new Produkty_form();
38
39  final static String PUSTY = "Pusty";
40  final static String PRODUKT = "Produkt form";
41  final static String PRODUKTY = "Produkty form";
42
43  //static Fasada_warstwy_biznesowej facade = new Fasada_warstwy_bi
44  static public Fasada_warstwy_biznesowej_ejbRemote getFacade() {
45      return fasada;
46  }

```
- Bottom Panel:**
- warstwa\_klienta\_desktop.GUI\_main
- Search Results
- Output
- Java DB Database Process
- GlassFish Server 4.1.1
- Sklep\_GuiPK\_lab2\_Desktop (clean)
- Members View:**
- PRODUKTY - Navigator
- Members: <empty>
- PRODUKTY : String



### 3.8. Uruchomienie projektu.

Należy w podanej kolejności wykonać operacje **Clean and Build** na projektach składowych (w celu łatwiejszej lokalizacji błędów):

- 1) Sklep\_6SE
- 2) SklepPK\_Lab2\_EE-ejb
- 3) SklepPK\_Lab2\_Web
- 4) Sklep\_GUIPK\_lab2\_EE\_Desktop
- 5) SklepPK\_Lab2\_EE

Następnie, należy wykonać operację **Deploy** na projekcie **SklepPK\_Lab2\_EE**.

Teraz można uruchomić dowolną liczbę aplikacji klienckich za pomocą operacji **Run**:

- 1) SklepPK\_Lab2\_Web
- 2) Sklep\_GUIPK\_lab2\_EE\_Desktop

W przykładzie uruchomiono dwie instancje aplikacji desktopowej i jedną internetową.

### 3.8.1. Proces wprowadzania danych w uruchomionych aplikacjach klienckich

MenuDemo

A Menu Inne Menu

Nazwa  
Produkt1

Cena  
123

Promocja  
12

Data  
20-03-2017

Dodaj produkt

MenuDemo

A Menu Inne Menu

Nazwa  
Produkt2

Cena  
223

Promocja  
22

Data  
20-03-2017

Dodaj produkt

Wstawianie nowego produktu - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Wstawianie nowego produktu x +

localhost:8080/SklepPK\_Lab2 Search

**Dodaj produkt**  
**Lista produktów**

Podaj nazwe produktu Produkt3

Podaj cene netto produktu 330 zł

Podaj promocje produktu 33

Podaj date produkcji 18-03-2017

OK

Bottom

MenuDemo

A Menu Inne Menu

Id produktu	Nazwa	Cena	Promocja	Data	Cena brutto
1	Produkt1	123.0	12	Mon Mar 20 20:25:10 CET 2017	108.24
2	Produkt2	223.0	22	Mon Mar 20 20:26:18 CET 2017	173.93999
3	Produkt3	330.0	33	Sat Mar 18 01:00:00 CET 2017	221.09999

Produkty

[2, Produkt2, 223.0, 22, Mon Mar 20 20:26:18 CET 2017, 173.93999]

A Menu Inne Menu

Id produktu	Nazwa	Cena	Promocja	Data	Cena brutto
1	Produkt1	123.0	12	Mon Mar 20 20:25:10 CET 2017	108.24
2	Produkt2	223.0	22	Mon Mar 20 20:26:18 CET 2017	173.93999
3	Produkt3	330.0	33	Sat Mar 18 01:00:00 CET 2017	221.09999

Produkty

Lista produktow - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Lista produktow

localhost:8080/SklepPK\_Lab2\_Web/faces/warstwa\_interneto

Dodaj produkt  
Lista produktow

Id produktu	Nazwa produktu	Cena netto produktu	Promocja produktu	Data produkcji	Cena brutto
1	Produkt1	123.0	12	Mon Mar 20 20:25:10 CET 2017	108.24
2	Produkt2	223.0	22	Mon Mar 20 20:26:18 CET 2017	173.93999
3	Produkt3	330.0	33	Sat Mar 18 01:00:00 CET 2017	221.09999

Powrot

Bottom

3.8.2. Każda z aplikacji klienckich korzysta z danych umieszczonych w tym samym komponencie EJB typu Session – rodzaj Stateless.