

Internet Engineering

Tomasz Babczyński, Zofia Kruczkiewicz
Tomasz Kubik

**Information systems modelling – UML and
service description languages**
Laboratory 4

Choose yourself and new technologies



HUMAN CAPITAL
HUMAN – BEST INVESTMENT!



Wrocław University of Technology

EUROPEAN
SOCIAL FUND





Design patterns used to build the Integration nad Resources Tiers

D.Alur, J.Crupi, D. Malks, Core J2EE. Desin Patterns

Outline of creating the Library Catalogue Java Application

1. Create a database in the Derby database system
2. Create Annotations in the object model of the Business Tier
3. You may add annotation to your own new classes and create the proper controllers – for higher assessment (5.0 or 5.5)
4. Create Session Beans for Entity classes





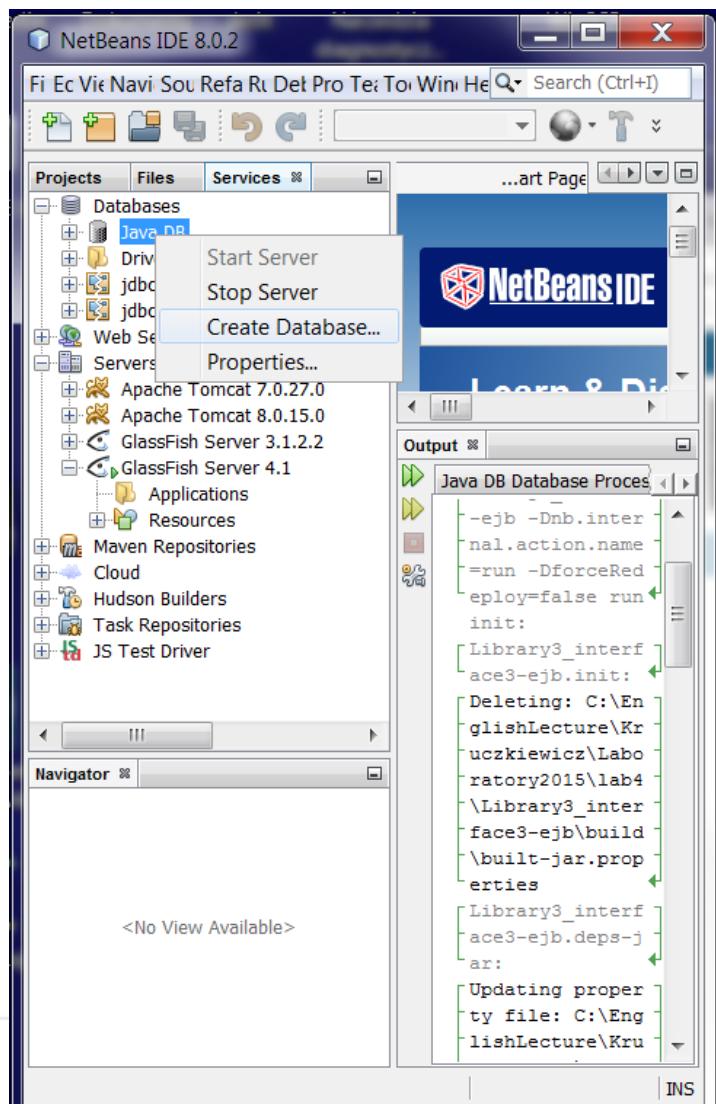
1. Create a database in the Derby database system

1. Select the Services tab and then expand the Databases node. Then right-click on Java DB item and select Create Database – (slide 2)
2. In the form for creating a database (slide 3), enter the data as follows: Database Name, User, Password and the Database Location (default or selected by the user). In the selected directory will be created a directory named database with an empty database (slide 4)
3. Check a directory where you have created from NetBeans an empty database in the system database Derby - - you can see the new catalogue Library1 (slide 4)
4. To connect to the database: Right-click the database connection node and choose the Connect item (slide 6) - **jdbc:derby://localhost:1527/Library2015:[Library2015 on LIBRARY2015] (jdbc:derby://localhost:1527/DataBase_Name:[User on DataBaseSchema])**.
5. Now expand the database connection node to browse the database until disconnect from the database. Then expand the database node to see the empty Library Database (slide 7) p – the Library2015 node is lacking.

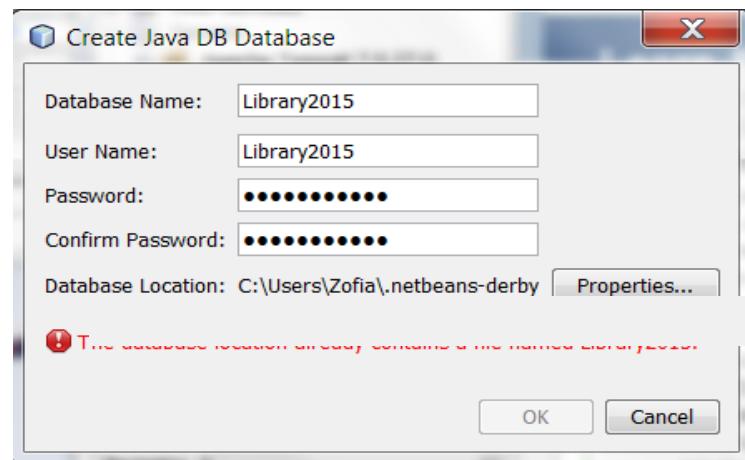




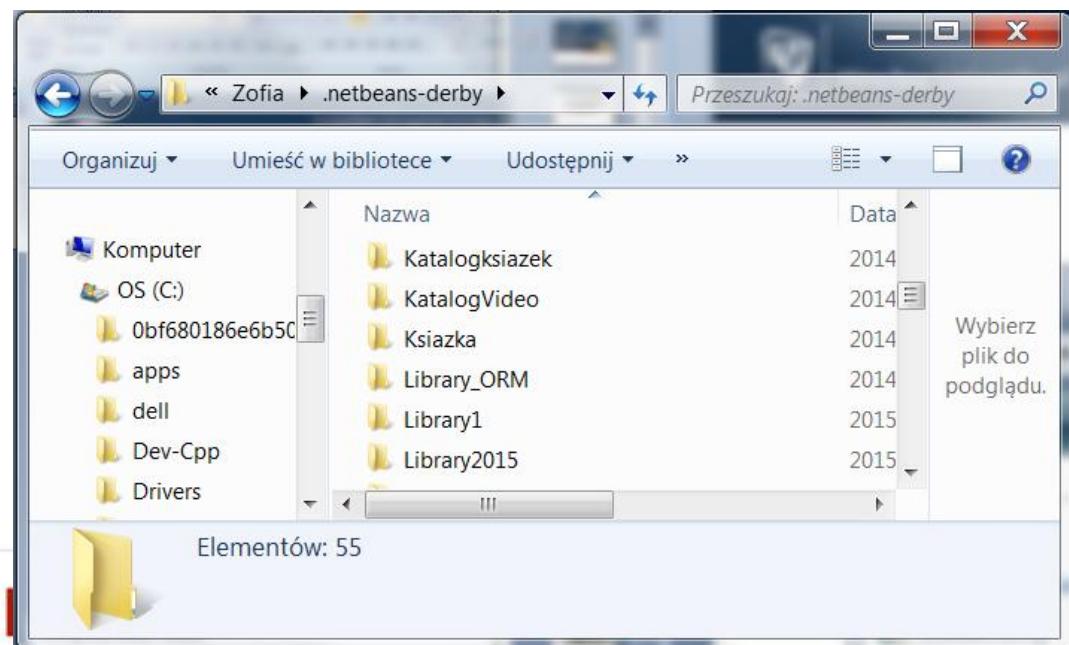
1)



2)



3)





4)

NetBeans IDE 8.0.2

File Edit View Navig Sour Refac Ru Deb Profi Tea Too Wind Help Search (Ctrl+I)

Projects Files Services

Databases Java DB Drivers

jdbc:derby://localhost:1527/Library2015 [Library2015 on LIBRARY2015]

jdbc:derby://localhost:1527/sample [app on APP]

Web Services

Servers

Apache Tomcat 7.0.27.0 Apache Tomcat 8.0.15.0 GlassFish Server 3.1.2.2 GlassFish Server 4.1 Applications Resources Maven Repositories Cloud Hudson Builders Task Repositories JS Test Driver

Navigator <No View Available>

INS

A context menu is open over the 'jdbc:derby://localhost:1527/Library2015' database entry. The menu options are: Connect..., Disconnect, Execute Command..., Refresh, Delete, Rename..., and Properties.

5)

NetBeans IDE 8.0.2

File Edit View Navig Sour Refac Ru Deb Profi Tea Too Wind Help Search (Ctrl+I)

Projects Files Services

Databases Java DB Drivers

jdbc:derby://localhost:1527/Library2015 [Library2015 on LIBRARY2015]

LIBRARY2015

Tables Views Procedures Other schemas

jdbc:derby://localhost:1527/sample [app on APP]

Web Services

Servers

Apache Tomcat 7.0.27.0 Apache Tomcat 8.0.15.0 GlassFish Server 3.1.2.2 GlassFish Server 4.1 Applications Resources

Navigator <No View Available>

INS

The 'LIBRARY2015' database node under 'Databases' is expanded, showing its sub-structure: Tables, Views, Procedures, and Other schemas.





3. Create Annotations in the object model

Mapping Entity Classes (Help of the NetBeans)

- An entity class is used to represent a table in a database, and the fields in an entity class correspond to columns in that table. In an entity class, you can use annotations to specify how fields in an entity class are mapped to the corresponding database columns and tables.
- If the name of the mapped database column is the same as the field or property name because they are mapped by default.

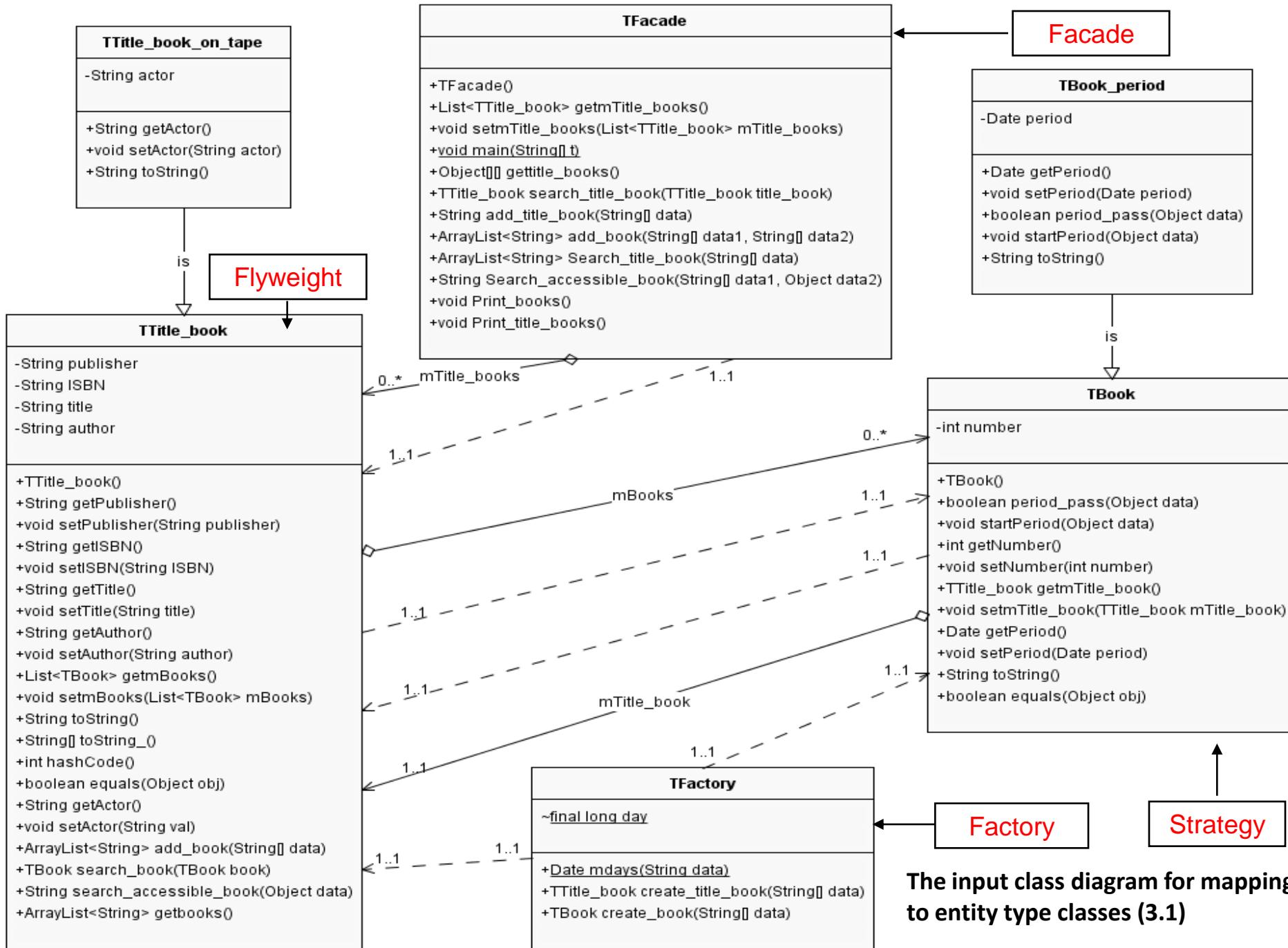
The following annotations are commonly used when mapping entity classes:

- @Entity defines the Entity class
- @Id Specifies the primary key property or field of an entity.
- @GeneratedValue Allows you to specify the strategy that automatically generates the values of primary keys. Used with @Id.
- @Column Specifies a mapped column for a persistent property or field.
- @ManyToMany Defines a many-valued association with many-to-many multiplicity.
- @ManyToOne Defines a single-valued association to another entity class that has many-to-one multiplicity.
- @OneToMany Defines a many-valued association with one-to-many multiplicity. For more on using annotations and annotation elements to map entities in an enterprise application, see the Java EE 5 Tutorial

Insert the annotations shown in the next slides to the TTitle_book, TTitle_book_on_tape, TBook, TBook_period

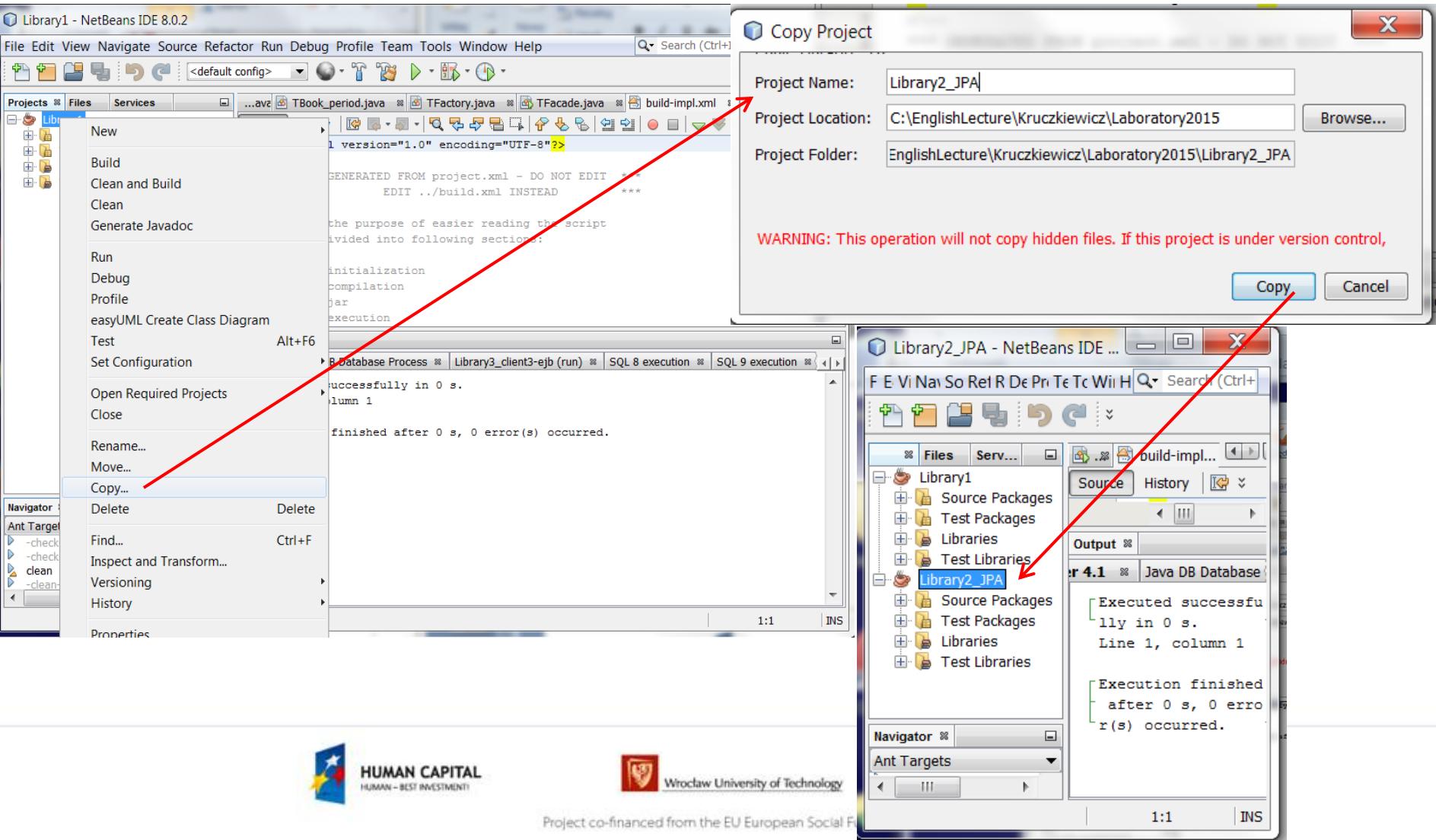
Click the Add Class and select all classes as the Entity classes (slide 5)







3.2. Creation the copy of the Library1 project (lab1) as the Library2_JPA project – right click the Library1 project item in the Projects tab, choose the Copy item; fill the name, select the location of new project and click Copy





3.3. Add libraries of EclipseLink (JPA 2.1) technologies to the Library2_JPA project – right click the Libraries folder, select the Add Library... item; choose the EclipseLink from GlassFish and click AddLibrary

The screenshot shows two NetBeans IDE windows. The left window is titled "Library2_JPA - NetBeans IDE 8.0.2" and displays a project structure with a "Libraries" node selected. A context menu is open, with the "Add Library..." option highlighted by a red arrow. The right window is titled "Add Library" and shows a list of "Available Libraries". The "EclipseLink (JPA 2.1)" and "EclipseLink from GlassFish" options are listed and highlighted with blue boxes. A second red arrow points to the "Add Library" button at the bottom of the dialog. Both windows show the "Source" tab selected. The right window also displays the contents of the "Library2_JPA2" project, which includes several Java files under "sub_business_tier" and "sub_business_tier.entities" packages, and a list of added libraries at the bottom.



Library2_JPA2 - NetBeans IDE 8.0.2

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

Search (Ctrl+I)

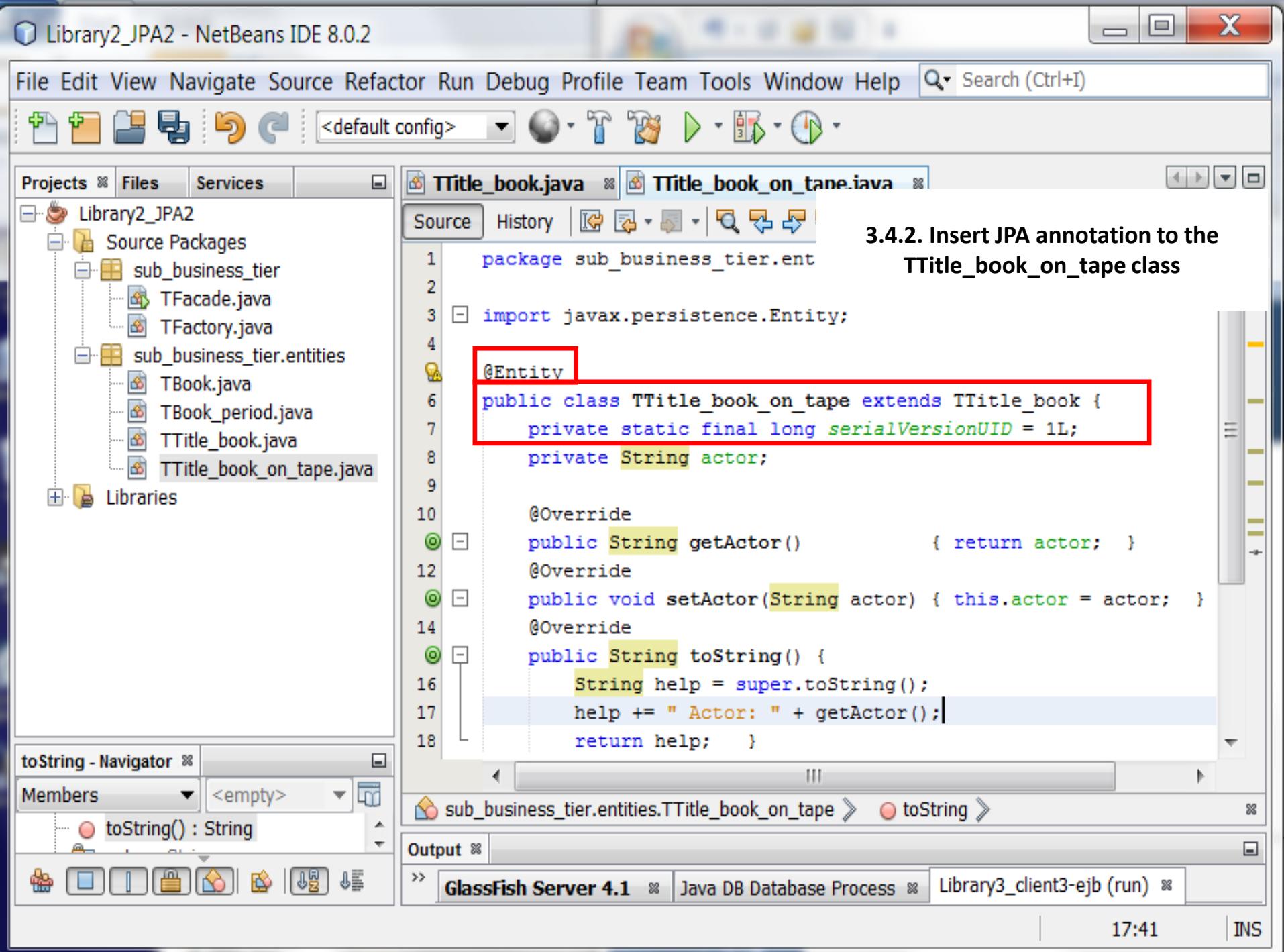
Projects Files Services

...ava TTtitle_book_on_tape.java TTtitle_book.java Book_form.java Card0.java

Source History

```
1 package sub_business tier.entities;
2
3 import java.io.Serializable;
4 import java.util.ArrayList;
5 import java.util.Collection;
6 import java.util.Iterator;
7 import java.util.List;
8 import java.util.Objects;
9 import javax.persistence.Entity;
10 import javax.persistence.GeneratedValue;
11 import javax.persistence.GenerationType;
12 import javax.persistence.Id;
13 import javax.persistence.OneToMany;
14 import javax.persistence.Transient;
15 import sub_business tier.TFactory;
16
17 @Entity
18 public class TTtitle_book implements Serializable {
19     private static final long serialVersionUID = 1L;
20     private String publisher;
21     private String ISBN;
22     private String title;
23     private String author;
24
25     @Id
26     @GeneratedValue(strategy = GenerationType.AUTO)
27     private Long id;
28     public Long getId() { return id; }
29     public void setId(Long val) { id = val; }
30
31     @OneToMany(mappedBy = "mTitle_book")
32     private Collection<TBook> books;
33     public Collection<TBook> getBooks() { return books; }
34     public void setBooks(Collection<TBook> val) { books = val; }
35
36     @Transient
37     List<TBook> mBooks;
38     public List<TBook> getmBooks() { return mBooks; }
39     public void setmBooks(List<TBook> mBooks) { this.mBooks = mBooks; }
40
41     public TTtitle_book() { mBooks = new ArrayList<>(); }
```

3.4. /3.4.1. Insert JPA annotation to the TTtitle_book class



Library2_JPA2 - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)

Projects Files Services

Source Packages

- sub_business_tier
 - TFacade.java
 - TFactory.java
- sub_business_tier.entities
 - TBook.java
 - TBook_period.java
 - TTitle_book.java
 - TTITLE_book_on_tape.java

Libraries

TTITLE_book_on_tape.java TBook.java

Source History

1 package sub_business_tier.entities;

2

3 import java.io.Serializable;

4 import java.util.Date;

5 import javax.persistence.Entity;

6 import javax.persistence.GeneratedValue;

7 import javax.persistence.GenerationType;

8 import javax.persistence.Id;

9 import javax.persistence.ManyToOne;

10

11 @Entity

12 public class TBook implements Serializable {

13

14 private static final long serialVersionUID = 1L;

15 private int number;

16 @ManyToOne

17 private TTitle_book mTitle_book;

18 public TTitle_book getmTitle_book() {

19 return mTitle_book; }

20 public void setmTitle_book(TTitle_book mTitle_book) {

21 this.mTitle_book = mTitle_book; }

22

23 @Id

24 @GeneratedValue(strategy = GenerationType.AUTO)

25 private Long id;

26 public void setId(Long id) { this.id = id; }

27 public Long getId() { return id; }

28

29 public TBook() { }

30

3.4.3. Insert JPA annotation to the TBook class

Library2_JPA2 - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)

Projects Files Services

Library2_JPA2

- Source Packages
 - sub_business_tier
 - TFacade.java
 - TFactory.java
 - sub_business_tier.entities
 - TBook.java
 - TBook_period.java
 - TTtitle_book.java
 - TTtitle_book_on_tape.java
- Libraries

...ava TBook.java TBook_period.java

Source History

```
1 package sub_business_tier.entities;
2
3 import java.util.Date;
4 import javax.persistence.Entity;
5 import javax.persistence.Temporal;
6 import sub_business_tier.TFactory;
7
8
9 @Entity
10 public class TBook_period extends TBook {
11
12     private static final long serialVersionUID = 1L;
13
14     @Temporal(javax.persistence.TemporalType.DATE)
15     private Date period;
16
17     @Override
18     public Date getPeriod() {
19         return period;
20     }
21
22     @Override
23     public void setPeriod(Date date) {
24         period = date;
25     }
26 }
```

period - Navigator

Members

- TBook_period :: TBook
 - getPeriod() : Date
 - period_pass(Object data) : boolean
 - setPeriod(Date date)
 - startPeriod(Object data)
 - toString() : String
 - period : Date
 - serialVersionUID : long

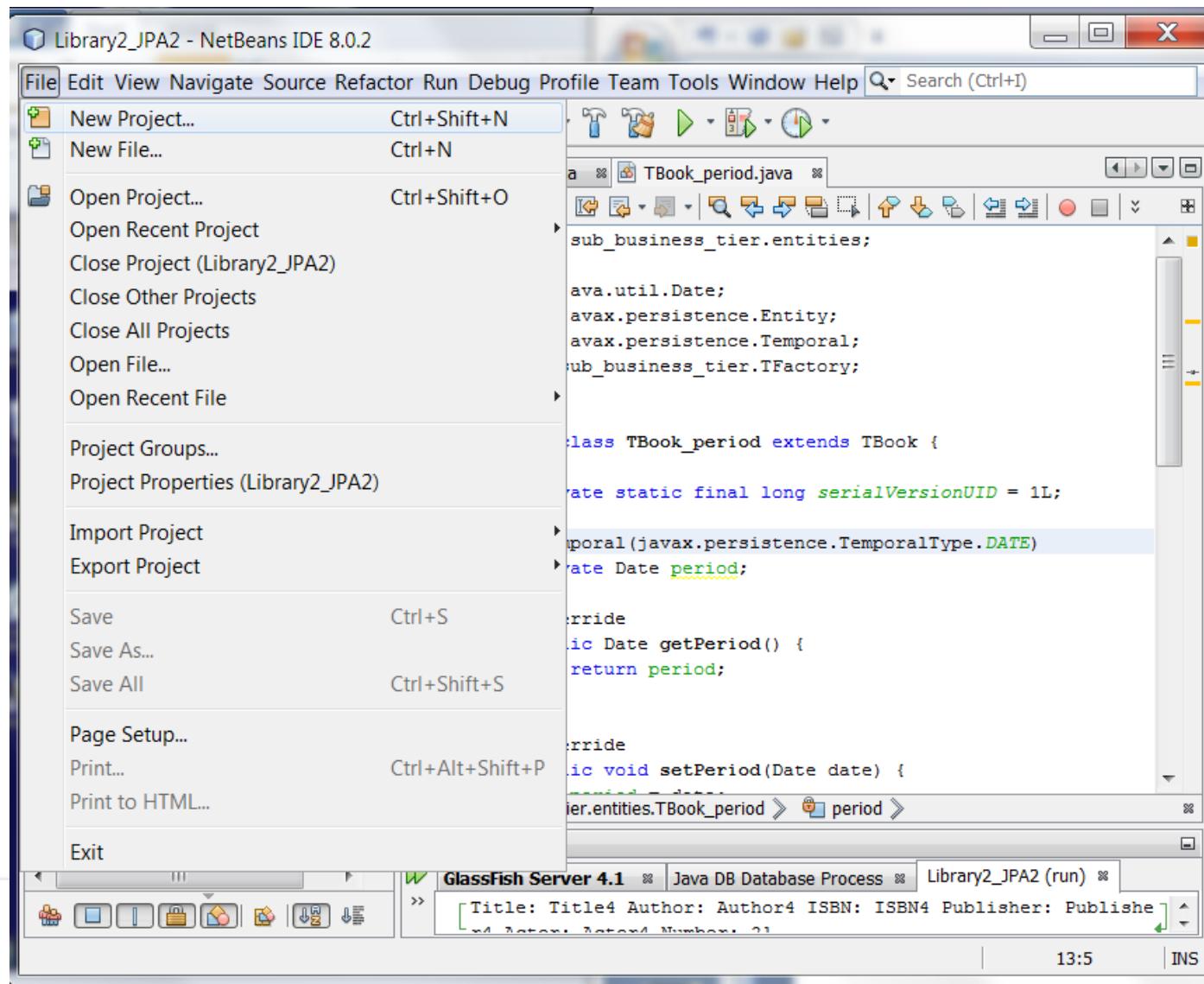
Output

>> GlassFish Server 4.1 Java DB Database Process Library3_client3-ejb

3.4.4. Insert JPA annotation to the TBook_period class



4. /4.1./4.1.1. Create the new Enterprise Application project – click File/New Project...





New Project

Steps

1. Choose Project
2. ...

Choose Project

Filter:

Categories:

- Java
- JavaFX
- UML
- Java Web
- Java EE
- HTML5

Projects:

- Enterprise Application
- Enterprise Application with Existing Sources
- EJB Module
- EJB Module with Existing Sources
- Enterprise Application Client
- Enterprise Application Client with Existing Sources

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 IDE-generated Ant build script to build and run your projects.

< Back Next > Finish Cancel Help

New Enterprise Application

Steps

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

Name and Location

Project Name:

Project Location:

Project Folder:

Use Dedicated Folder for Storing Libraries

Libraries Folder:

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

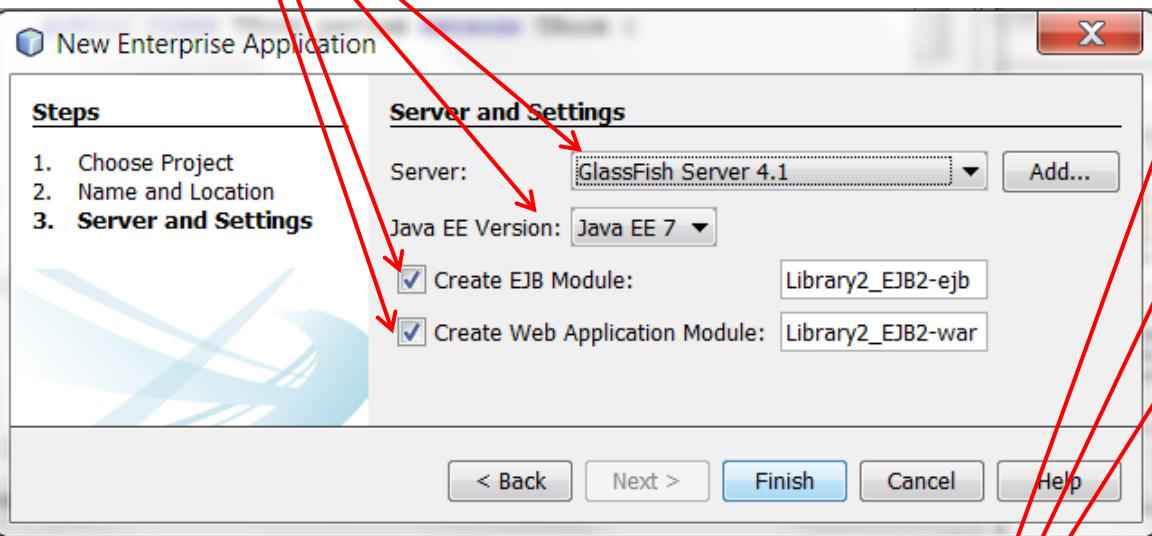
< Back Next > Finish Cancel Help

4.1.2. Select the Java EE/ Enterprise Application options

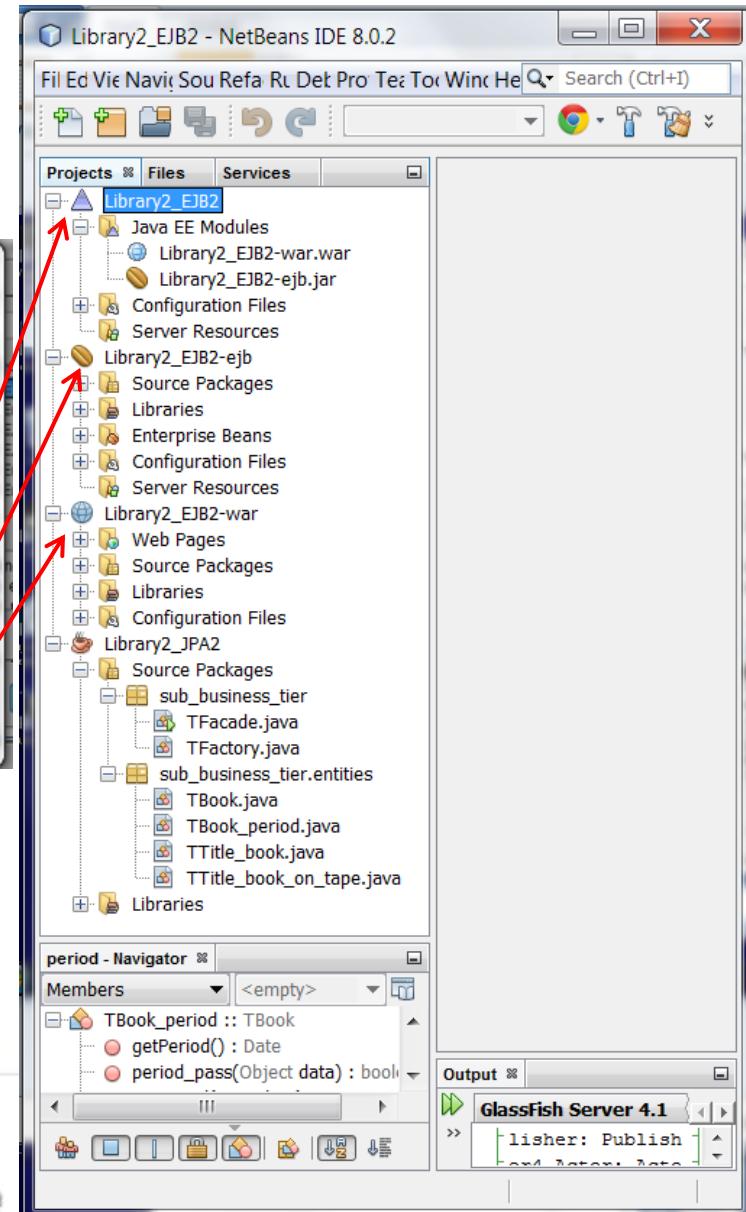
4.1.3. Fill the Library2_EJB2 name of Java EE project and select location of this project



4.1.4. Select the Application Server, JavaEE 7 platform and two types of modules: EJB modelu and Web Application module.

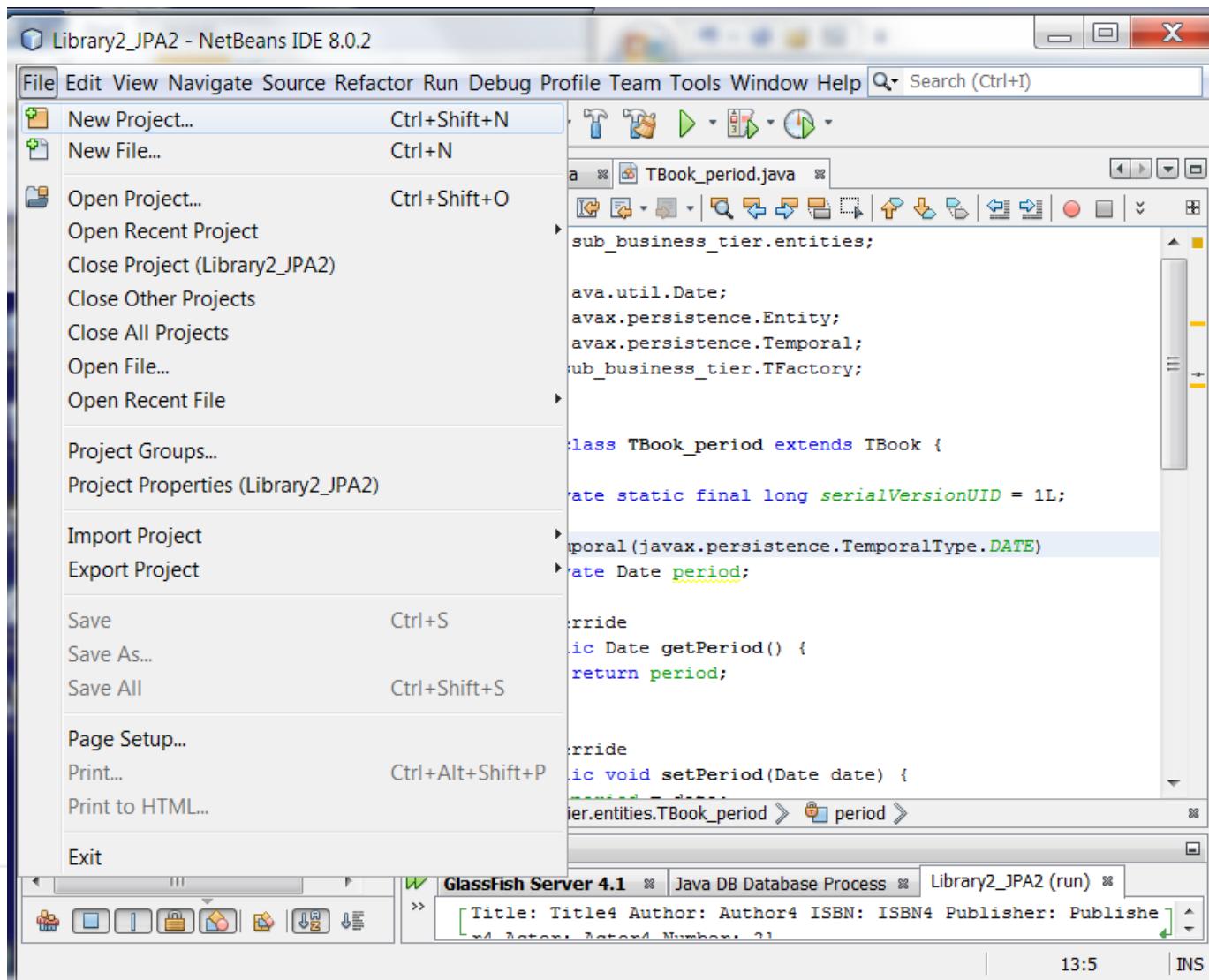


4.1.5. Result





4.2./4.2.1. Create the Java Class Library project as the interface for remote Session Bean in the Library2_EJB2-ejb module of the Library2_EJB2 Enterprise Application project – click File/New Project...





New Project

Steps

1. Choose Project
2. ...

Choose Project

Filter:

Categories:

- Java
- JavaFX
- UML
- Java Web
- Java_EE

Projects:

- Java Application
- Java Class Library
- Java Project with Existing Sources
- Java Free-Form Project

Description:

Creates a new Java SE library in a standard IDE project. A Java SE library does not contain a main class. Standard projects use an

< Back Next > Finish Cancel Help

4.2.2. Select the Java/ Java Class Library options and click Next

New Java Class Library

Steps

1. Choose Project
2. Name and Location

Name and Location

Project Name:

Project Location: C:\EnglishLecture\Kruczkiej\Laboratory2015\lab4

Project Folder: zkiewicz\Laboratory2015\lab4\Library2_interface2-ejb

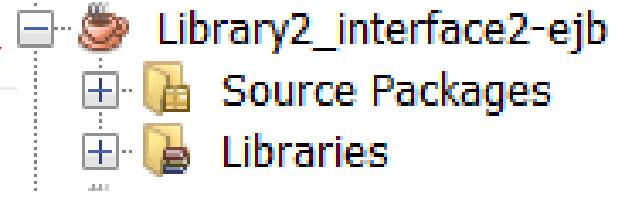
Use Dedicated Folder for Storing Libraries

Libraries Folder:

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

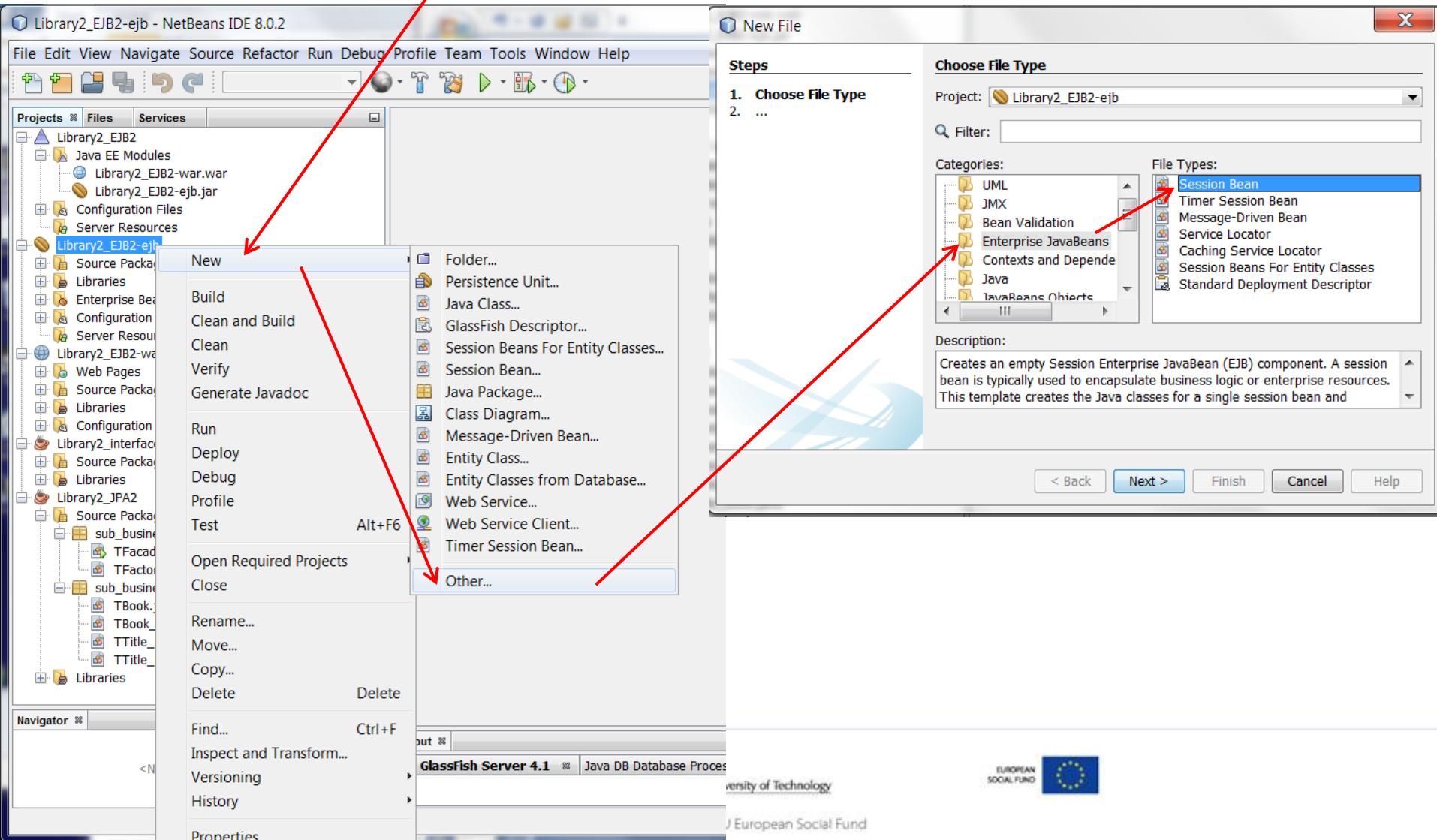
< Back Next > Finish Cancel Help

4.2.3. Fill the Library2_interface2-ejb name of Java Class Library project , select location of this project and click Finish. The result:



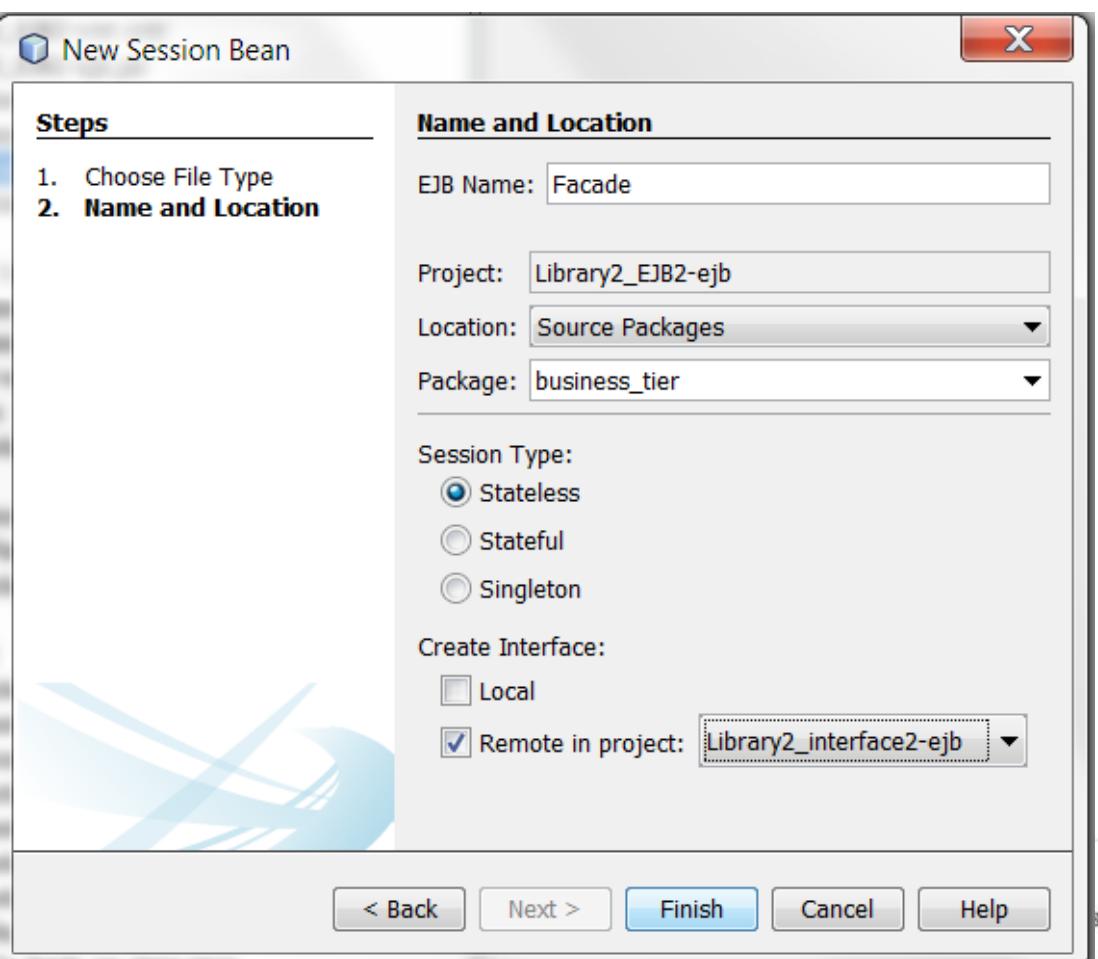


4.3./4.3.1. Create the remote Session Bean in the Enterprise Application project – right click the Library2_EJB2-ejb module in the Projects tab and select the New/Other items; select the Enterprise JavaBeans and Session Bean options; click Next

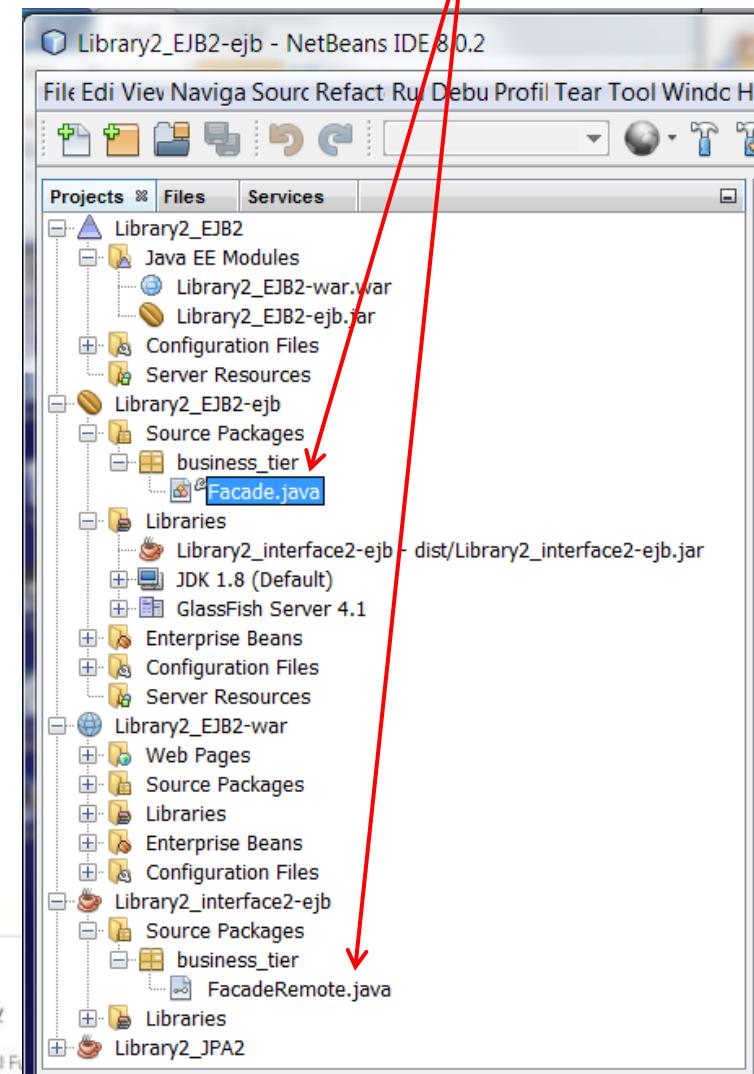




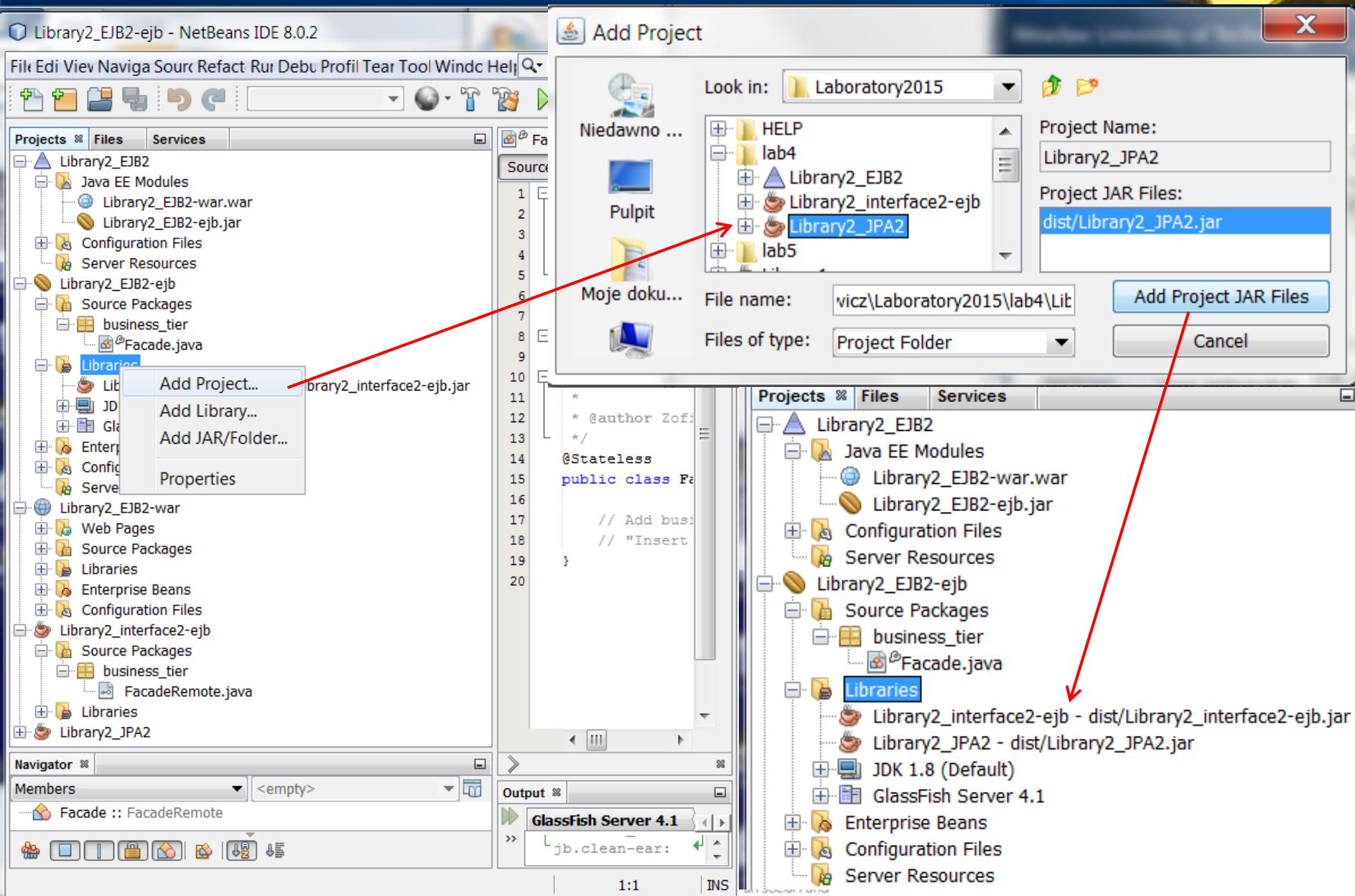
4.3.2. Fill the name of the remote Session Bean; create the new package (business_tier) in the Library2_EJB2-ejb Enterprise Application; select the the Stateless Session type; select the Remote and the Library2_interface2-ejb interface for the Facade as the remote Session Bean



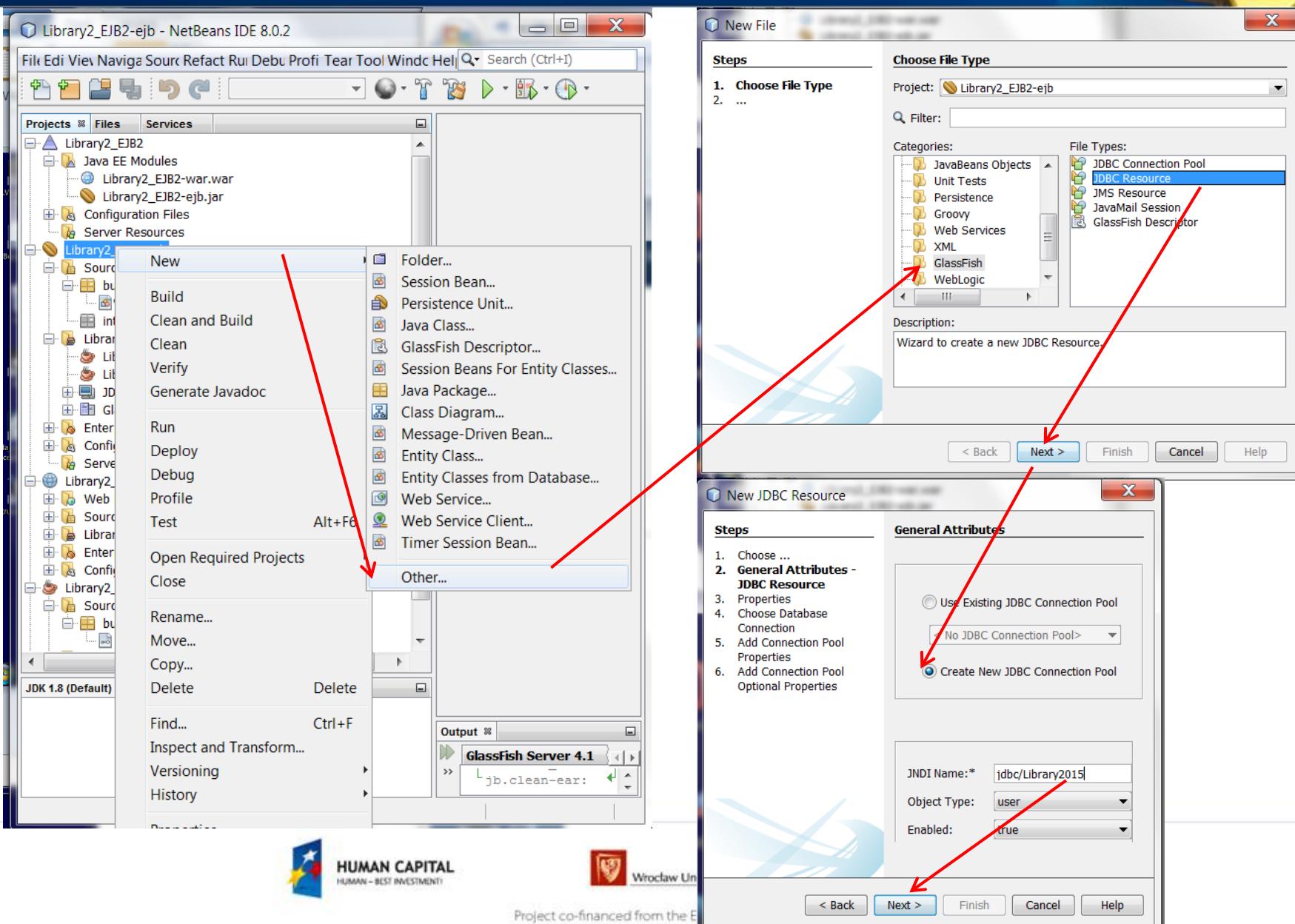
4.3.3. The result



4.4. Add the access to the business classes of the Library2_JPA2 Java Application Project (SE type) in the Library2_EJB2-ejb Enterprise Application – right click the Libraries folder and select the Add Project.. item



4.5./ 4.5.1 Create the JDBC resource of GlassFish for the access to the database by the Library2_EJB2-ejb Enterprise Application – right click the project item in the Projects tab, select: New/Other/GlassFish/JDBC Resource/Next/ Create New JDBC Connection Pool and fill the JNDI Name/Next





New JDBC Resource

Steps

1. Choose ...
2. General Attributes - JDBC Resource
3. Properties
- 4. Choose Database Connection**
5. Add Connection Pool Properties
6. Add Connection Pool Optional Properties

Choose Database Connection

Provide configuration information for the JDBC Connection Pool. Either choose an existing database connection to extract information, or enter the configuration information. Fields with an * mark are required.

JDBC Connection Pool Name: * Library2015_Pool

Extract from Existing Connection:
jdbc:derby://localhost:1527/Library2015 [Library2015 on LIBRARY2015]

New Configuration using Database:
< Select from the list >

XA (Global Transaction)

< Back Next > Finish Cancel Help

New JDBC Resource

Steps

1. Choose ...
2. General Attributes - JDBC Resource
3. Properties
4. Choose Database Connection
- 5. Add Connection Pool Properties**
6. Add Connection Pool Optional Properties

Add Connection Pool Properties

Enter the Datasource Classname, URL, and User to continue. Hit the Enter key to save values in the Properties table.

Datasource Classname: org.apache.derby.jdbc.ClientDataSource

Resource Type: javax.sql.DataSource

Description:

Properties:

Name	Value
URL	jdbc:derby://localhost:1527/Library2015
serverName	localhost
PortNumber	1527
DatabaseName	Library2015

< Back Next > Finish Cancel Help

4.5.2. Continuation

- 1) Go to the Choose Database Connection option: fill the JDBC Connection Pool Name, select the connection to Data source, created during the first step of this instruction (3 slide);
- 2) Add Connection Pool Properties
- 3) Add Connection Pool Optional Properties: click Finish

New JDBC Resource

Steps

1. Choose ...
2. General Attributes - JDBC Resource
3. Properties
4. Choose Database Connection
5. Add Connection Pool Properties
- 6. Add Connection Pool Optional Properties**

Specify Optional Properties for Connection Pool

Pool Settings

Steady Pool Size: 8

Max Pool Size: 32

Max Wait Time: 60000

Pool Resize Quantity: 2

Idle Timeout (secs): 300

Transaction Isolation

Transaction Isolation: JDBC Driver Default

Guarantee Isolation Level: true

Connection Validation

Connection Validation Required: true

Validation Method: auto-commit

Table Name: TAB_NAME

Fail All Connections: true

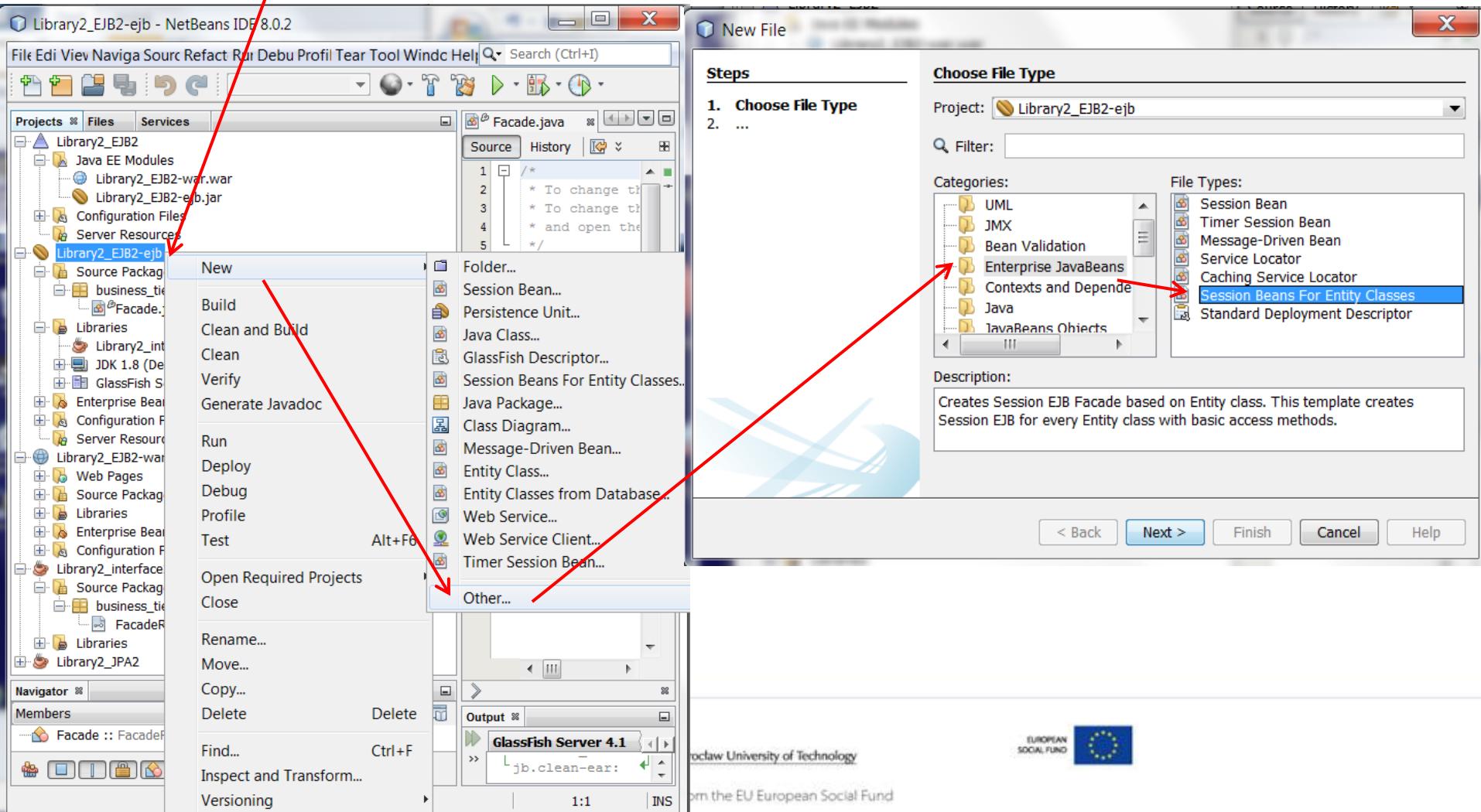
Non Transactional Connections: true

Allow Non Component Callers: false

< Back Next > Finish Cancel Help

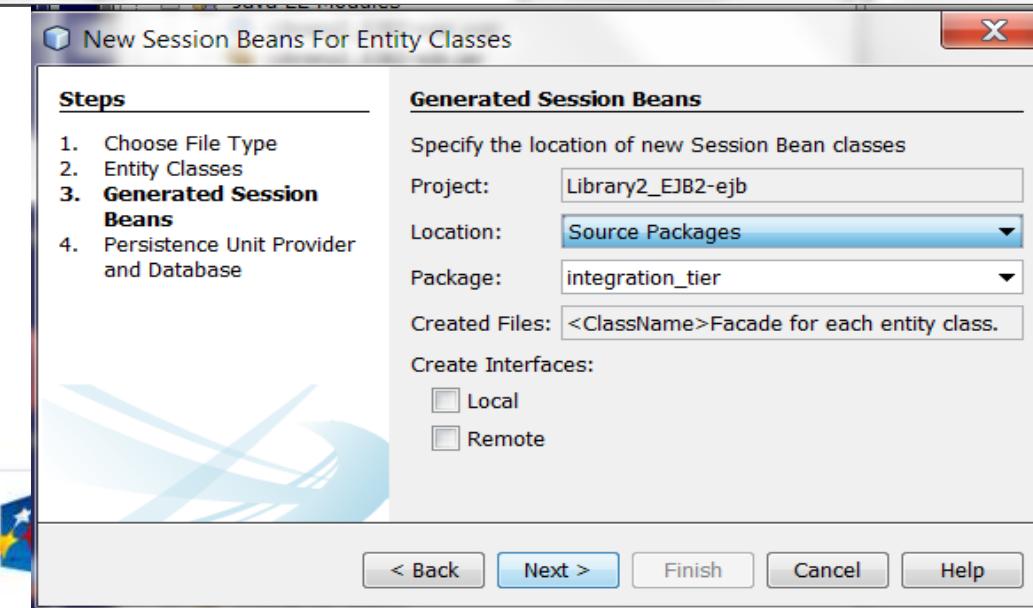
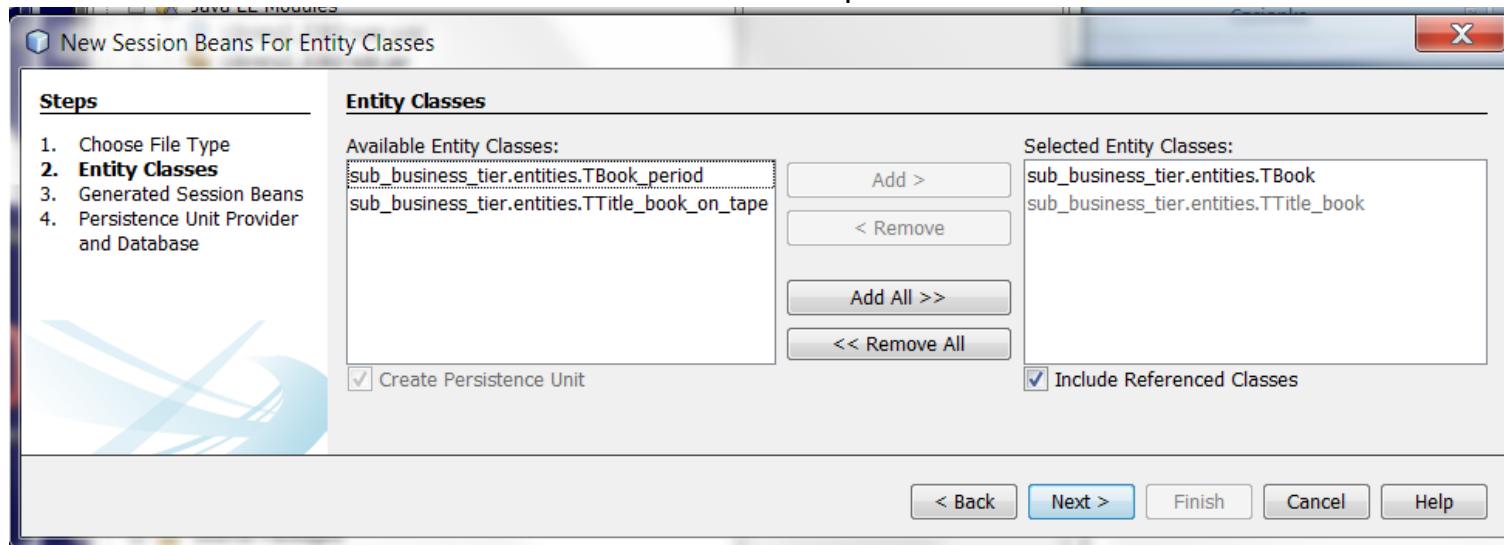


4.6./4.6.1. Create the local Session Bean for Entity Classes in the Library2_EJB2-ejb Enterprise Application project – right click the Library2_EJB2-ejb module in the Projects tab and select the New/Other items; select the Enterprise JavaBeans/Session Bean For Entity Classes options and click Next



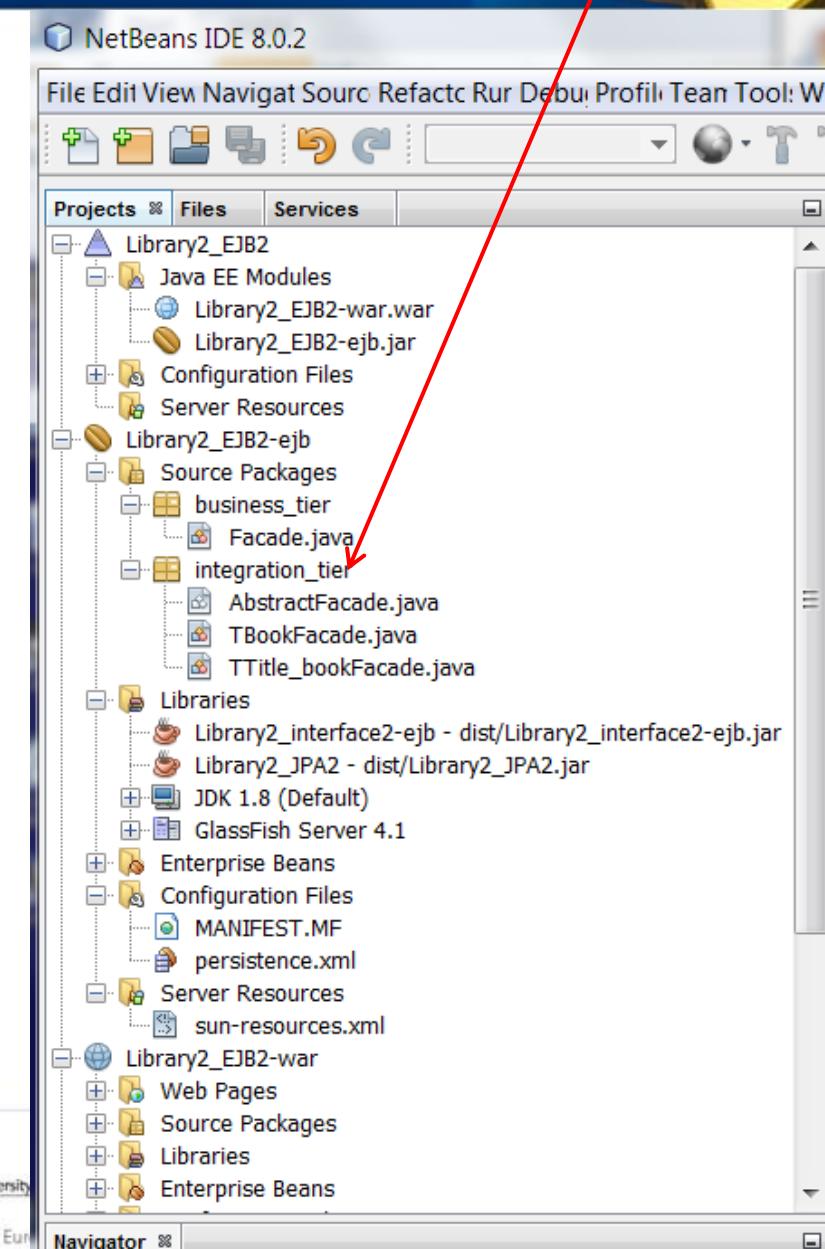
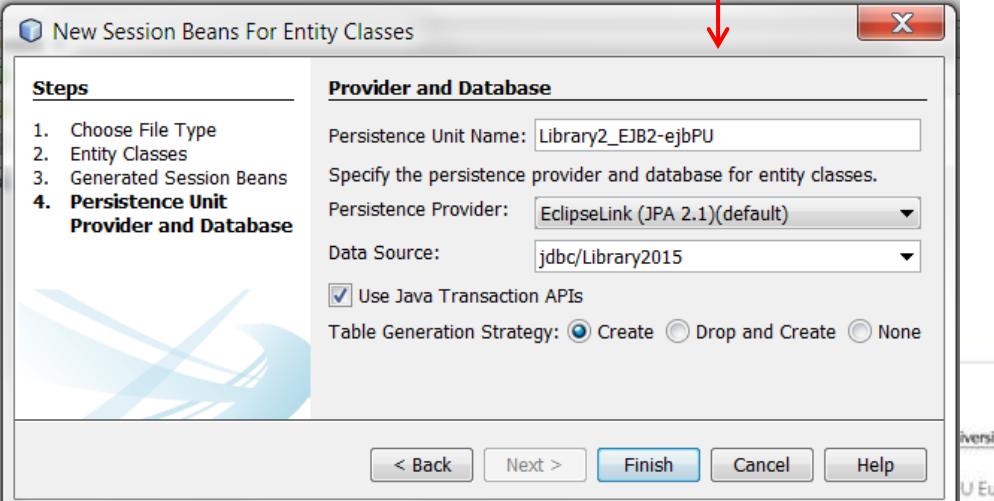
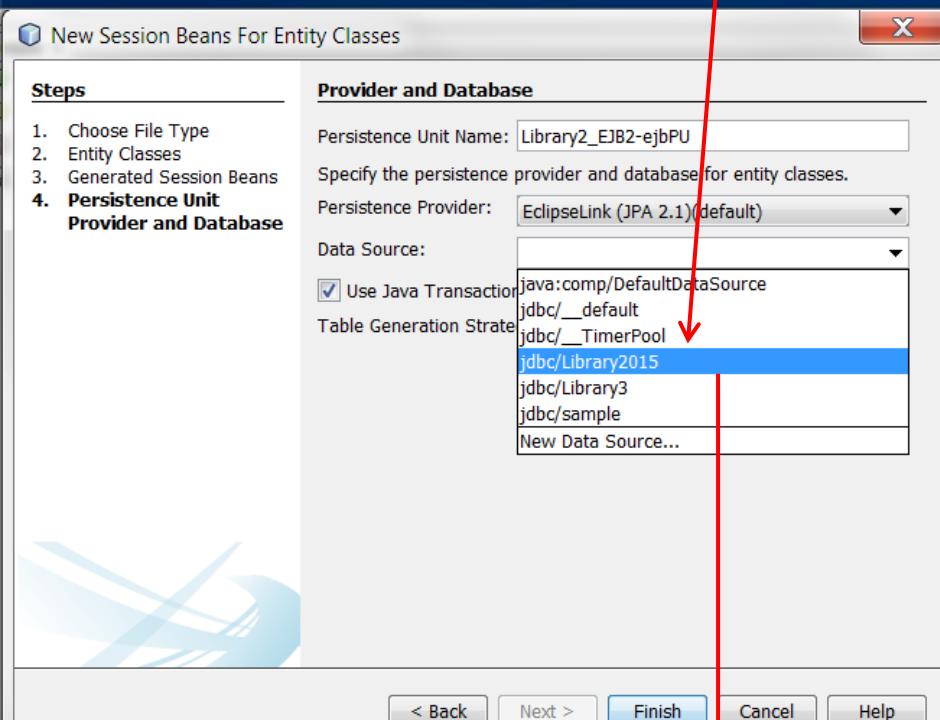


4.6.2. Continuation – select the TTitle_book and TBook entity classes and add to the right with the selected Include Referenced Classes option and click Next



4.6.3. Continuation – fill the field Package with the integration_tier name of the new package for created beans and click Next

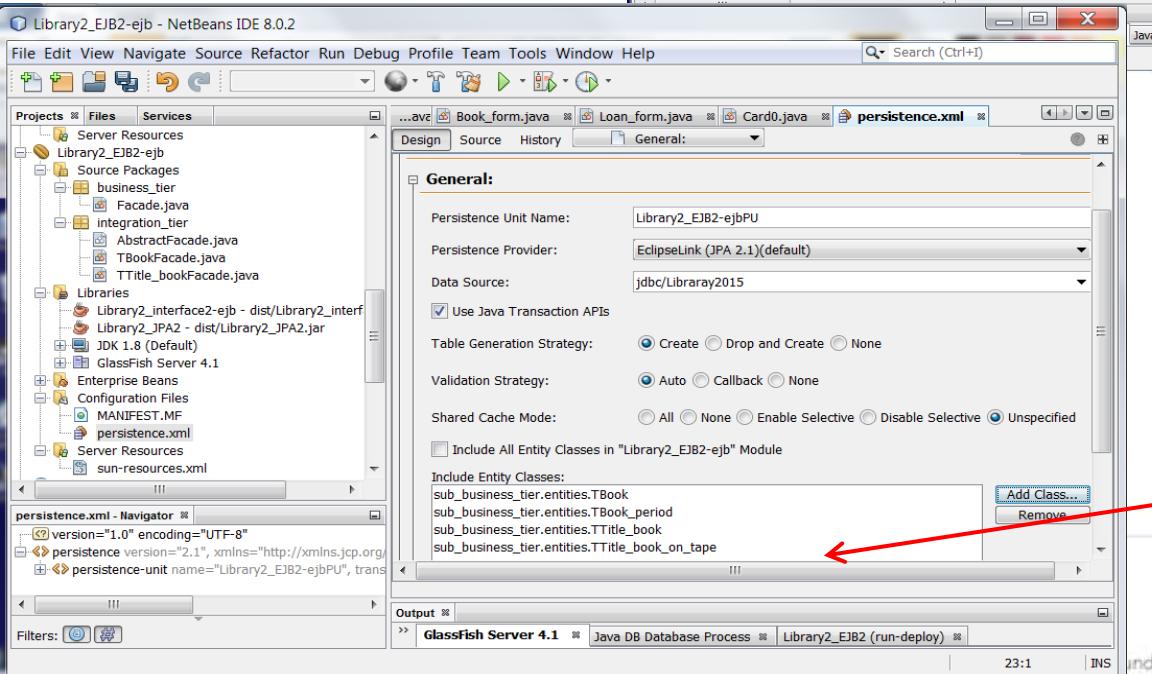
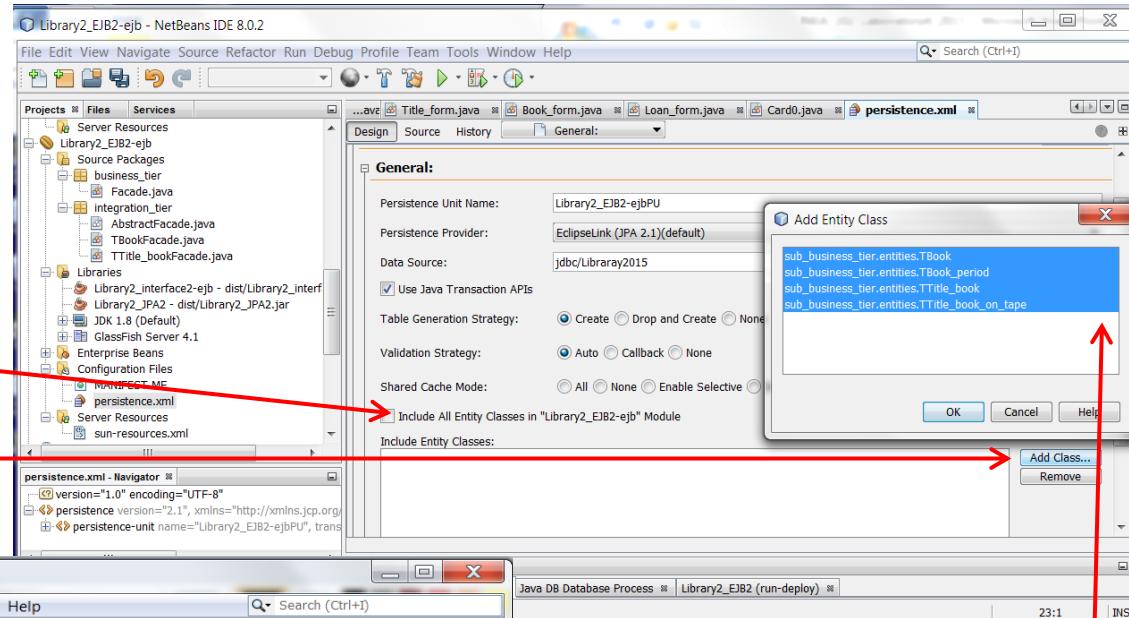
4.6.4. Continuation (Persistence Unit Provider and Database) – select the DataSource created during the 4.5 step and click Finish. The result is shown on the right.





4.7./ 4.7.1 Update the Include Entity Classes option in the created Persistence Unit in the 4.6 step

1. Unselect the **Include All Entity Classes** in „Library2_EJB2-ejb” Module.
2. Click the Add Class..



4.7.2 Continuation

1. Select all entity classes
2. The result



```
package integration_tier;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import sub_business_tier.entities.TTitle_book;
@Stateless
public class TTitle_bookFacade extends AbstractFacade<TTitle_book> {

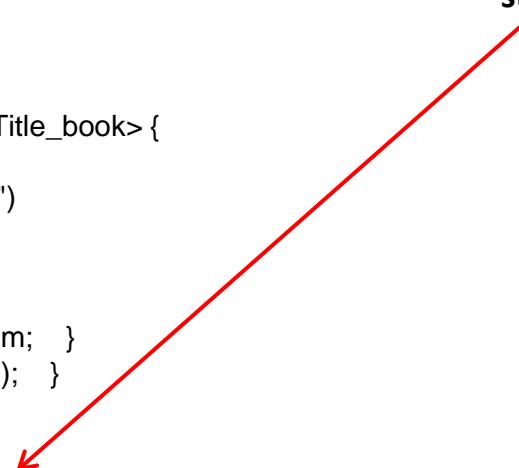
    @PersistenceContext(unitName = "Library2_EJB2-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() { return em; }
    public TTitle_bookFacade() { super(TTitle_book.class); }

    //Added code
    public TTitle_book[] getTTitle_books_() {
        return findAll().toArray(new TTitle_book[0]);
    }
    public void addTTitle_books(List<TTitle_book> titles) {
        TTitle_book newTTitle_book;
        Iterator<TTitle_book> it = titles.iterator();
        while (it.hasNext()) {
            newTTitle_book = it.next();
            if (newTTitle_book.getId() == null) {
                getEntityManager().persist(newTTitle_book);
            }
        }
    }
}
```

4.8./4.8.1. Update generated code of TTitle_bookFacade

Session Bean for TTitle_book and TTitle_book_on_tape entity classes, generated in the 4.6 steps



```

package integration_tier;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import sub_business_tier.entities.TBook;
import sub_business_tier.entities.TTitle_book;

@Stateless
public class TBookFacade extends AbstractFacade<TBook> {

    @PersistenceContext(unitName = "Library2_EJB2-ejbPU")
    private EntityManager em;

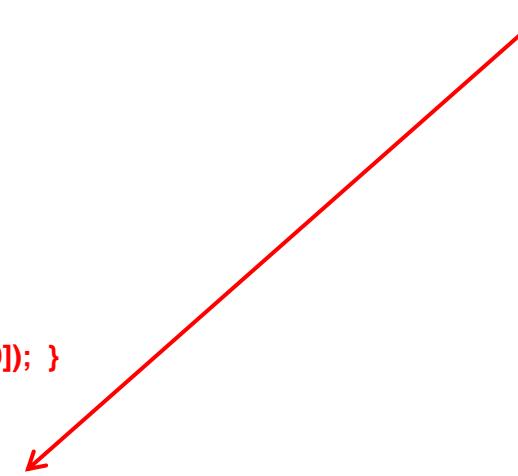
    @Override
    protected EntityManager getEntityManager() { return em; }
    public TBookFacade() { super(TBook.class); }

    //.....
    public TBook[] getTBooks_() { return findAll().toArray(new TBook[0]); }

    public void addTBooks(List<TTitle_book> titles) {
        TBook newBook;
        Iterator<TTitle_book> it = titles.iterator();
        while (it.hasNext()) {
            TTitle_book newTitle_book = it.next();
            if (newTitle_book.getId() == null) {
                continue;
            }
            Iterator<TBook> it_ = newTitle_book.getmBooks().iterator();
            while (it_.hasNext()) {
                newBook = it_.next();
                if (newBook.getId() == null) {
                    getEntityManager().persist(newBook);
                }
            }
        }
    }
}

```

4.8.2. Update generated code of TBookFacade Session Bean for TBook and TBook_period entity classes, generated in the 4.6 steps



**4.8.3. Right click and select the Fix Imports.. item. In the form of Fix Imports you must select the shown imports.**

The screenshot shows the NetBeans IDE interface with the following components:

- Left Panel:** Projects, Files, Services. The **Source** tab is selected, showing the **TBookFacade.java** file content.
- Middle Panel:** The code editor displays the **TBookFacade.java** file with several import statements underlined in red, indicating they are not resolved. The code includes methods like `getEntityManager()`, `getTBooks_()`, and `addTBooks(List<TTitle_book> titles)`.
- Right Panel:** A modal dialog titled "Fix Imports" is open, prompting the user to "Select the fully qualified name to use in the import statement." It lists three entries:
 - Import Statements:** `TTitle_book` (with dropdown options: `sub_business_tier.entities.TTitle_book`, `Iterator` (unchecked), and `List` (unchecked)).
 - Remove unused imports** checkbox is checked.
- Bottom Right:** A second modal dialog titled "Library2_EJB2 - NetBeans IDE 8.0.2" shows the project structure with the **TBookFacade.java** file highlighted in the Source Packages section.

Red arrows point from the text in the first section header to the "Fix Imports" dialog and the highlighted file in the second modal.

4.8.4. The result

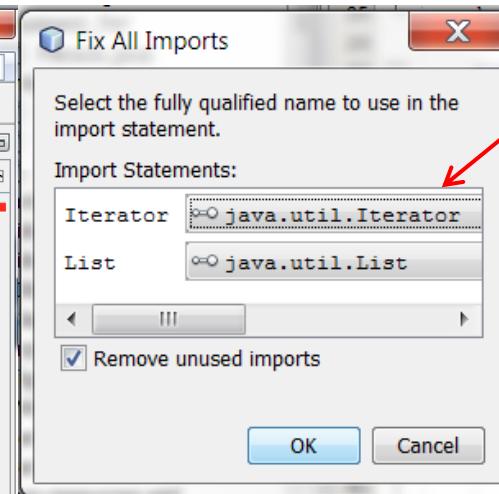


4.8.5. Right click and select the Fix Imports.. item. In the form of Fix Imports you must select the shown imports.

The screenshot shows the NetBeans IDE interface with the title bar "Library2_EJB2-ejb - NetBeans IDE 8.0.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Search (Ctrl+I). The toolbar has various icons for file operations. The left panel displays the project structure under "Projects" for "Library2_EJB2" with modules like "Java EE Modules" and "Server Resources". The main editor window shows Java code for "TTitle_bookFacade.java". The code includes imports for `java.util.Iterator` and `java.util.List`. A red arrow points from the text "In the form of Fix Imports you must select the shown imports." to the "Import Statements" section of the "Fix All Imports" dialog.

```
19     @PersistenceContext(unitName = "Library2_EJB2-ejbPU")
20     private EntityManager em;
21
22     @Override
23     protected EntityManager getEntityManager() {
24         return em;
25     }
26
27     public TTitle_bookFacade() {
28         super(TTitle_book.class);
29     }
30
31
32     public TTitle_book[] getTTTitle_books_() {
33         return findAll().toArray(new TTitle_book[0]);
34     }
35
36
37     public void addTTTitle_books(List<TTitle_book> titles) {
38         TTitle_book newTTitle_book=null;
39         Iterator<TTitle_book> it = titles.iterator();
40         while (it.hasNext()) {
41             newTTitle_book = it.next();
42             if (newTTitle_book.getId() == null) {
43                 getEntityManager().persist(newTTitle_book);
44             }
45         }
46     }

```



The screenshot shows the NetBeans IDE interface with the title bar "Library2_EJB2-ejb - NetBeans IDE 8.0.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Search (Ctrl+I). The toolbar has various icons for file operations. The left panel displays the project structure under "Projects" for "Library2_EJB2" with modules like "Java EE Modules" and "Server Resources". The main editor window shows Java code for "TTitle_bookFacade.java". The code now includes the fully qualified imports `java.util.Iterator` and `java.util.List`. A red arrow points from the text "The result" to the "TTitle_bookFacade.java" file in the project tree.

```
19     @PersistenceContext(unitName = "Library2_EJB2-ejbPU")
20     private EntityManager em;
21
22     @Override
23     protected EntityManager getEntityManager() {
24         return em;
25     }
26
27     public TTitle_bookFacade() {
28         super(TTitle_book.class);
29     }
30
31
32     public TTitle_book[] getTTTitle_books_() {
33         return findAll().toArray(new TTitle_book[0]);
34     }
35
36
37     public void addTTTitle_books(List<java.util.Iterator> titles) {
38         java.util.Iterator newTTitle_book=null;
39         Iterator<TTitle_book> it = titles.iterator();
40         while (it.hasNext()) {
41             newTTitle_book = it.next();
42             if (newTTitle_book.getId() == null) {
43                 getEntityManager().persist(newTTitle_book);
44             }
45         }
46     }

```

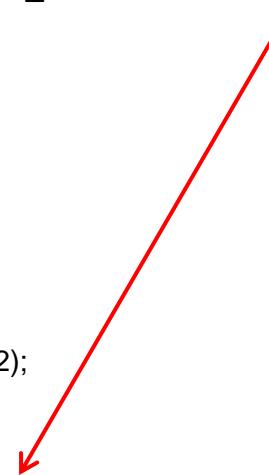
4.8.6. The result





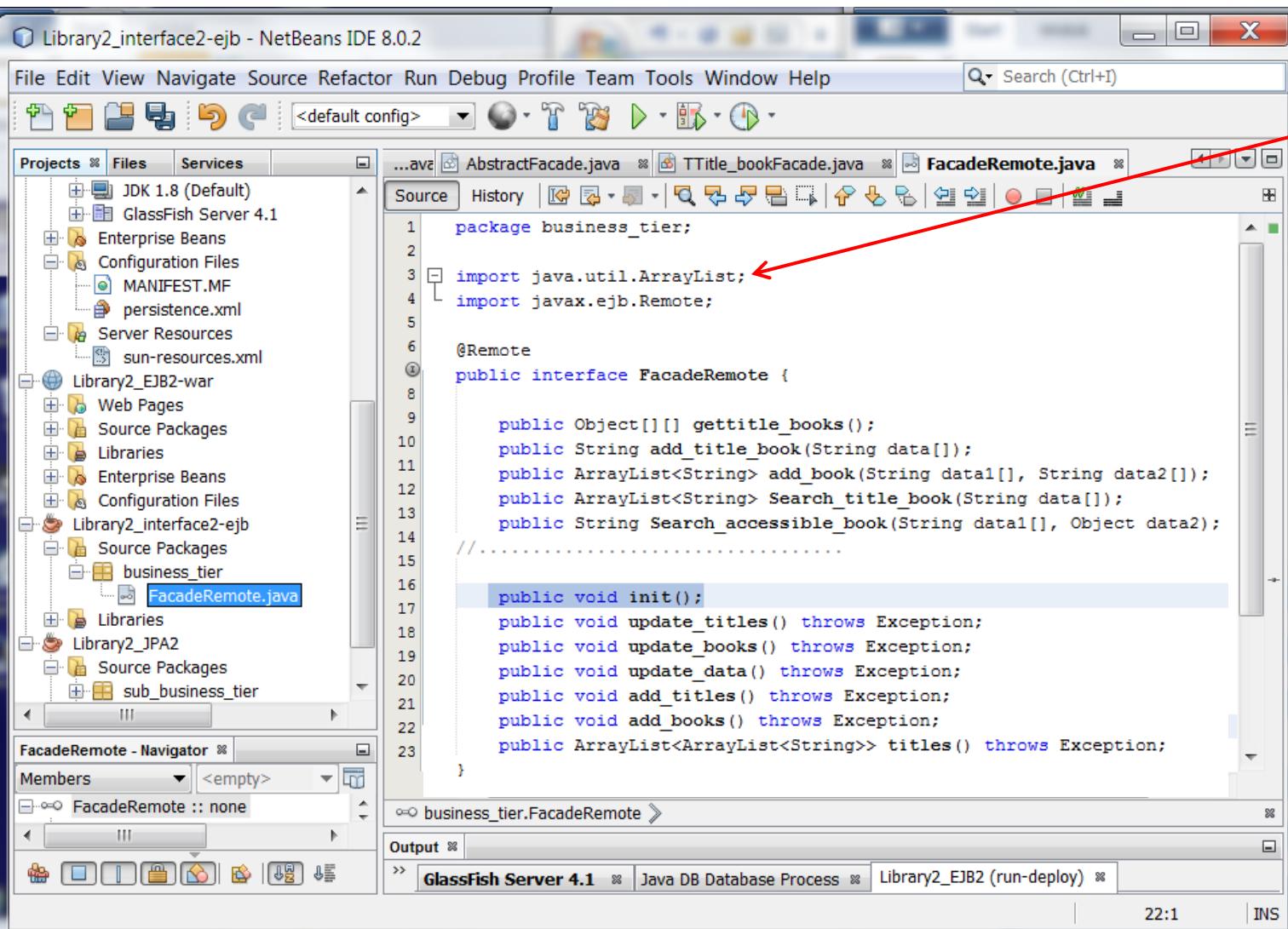
```
package business_tier;  
  
import javax.ejb.Remote;  
  
@Remote  
public interface FacadeRemote {  
    public Object[][] gettitle_books();  
  
    public String add_title_book(String data[]);  
  
    public ArrayList<String> add_book(String data1[], String data2[]);  
  
    public ArrayList<String> Search_title_book(String data[]);  
  
    public String Search_accessible_book(String data1[], Object data2);  
//.....  
    public void init();  
  
    public void update_titles() throws Exception;  
  
    public void update_books() throws Exception;  
  
    public void update_data() throws Exception;  
  
    public void add_titles() throws Exception;  
  
    public void add_books() throws Exception;  
  
    public ArrayList<ArrayList<String>> titles() throws Exception;  
}
```

**4.9./ 4.9.1 . Update code of the
FacadeRemote interface from the project of
lab3 – for accessing the database by using
the TTitle_bookFacade and TBookFacade**





4.9.2. After Fix
Imports.. process
- right click and select the Fix
Imports.. item. In the form of Fix
Imports... you must select the ArrayList import.



```
package business_tier;

import java.util.ArrayList;
import javax.ejb.Remote;

@Remote
public interface FacadeRemote {

    public Object[][] gettitle_books();
    public String add_title_book(String data[]);
    public ArrayList<String> add_book(String data1[], String data2[]);
    public ArrayList<String> Search_title_book(String data[]);
    public String Search_accessible_book(String data1[], Object data2);
    //.....
    public void init();
    public void update_titles() throws Exception;
    public void update_books() throws Exception;
    public void update_data() throws Exception;
    public void add_titles() throws Exception;
    public void add_books() throws Exception;
    public ArrayList<ArrayList<String>> titles() throws Exception;
}
```





4.10./ 4.10.1. Create the update of code of the Facade Session Bean - right click the code editor and select the Insert Code... item. In the Insert Code form select the Call Enterprise Bean... item and in its form select the TBookFacade, which is the local Enterprise Bean for persisting the TBook and TBook_period entities.

The screenshot shows the NetBeans IDE interface with the following details:

- Title Bar:** Library2_EJB2-ejb - NetBeans IDE 8.0.2
- Menu Bar:** File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
- Toolbar:** Standard NetBeans toolbar icons.
- Projects Tab:** Shows the project structure with Library2_EJB2-war.war, Library2_EJB2-ejb.jar, Configuration Files, Server Resources, and Source Packages (business tier, integration tier). The business tier contains Facade.java, AbstractFacade.java, TBookFacade.java, and TTtitle_bookFacade.java.
- Code Editors:** FacadeRemote.java and Facade.java are open. Facade.java contains the following code:

```
package business_tier;  
import javax.ejb.Stateless;  
  
@Stateless  
public class Facade implements FacadeRemote {  
  
    // Add business logic below / Right-click in editor  
}
```
- Context Menu:** A context menu is open over the code editor, with the "Insert Code..." option highlighted.
- Call Enterprise Bean... Dialog:** A modal dialog titled "Call Enterprise Bean" is displayed, showing a list of enterprise beans from open projects: Library2_EJB2-war, Library2_EJB2-ejb, Facade, and TBookFacade. The TBookFacade entry is selected.
- Call Enterprise Bean Dialog Fields:** Reference Name: TBookFacade, Referenced Interface: No interface, Local (radio button selected), Remote (radio button unselected).

Red arrows point from the "Call Enterprise Bean..." option in the context menu to the "Call Enterprise Bean..." option in the "Insert Code..." submenu, and then to the TBookFacade entry in the "Call Enterprise Bean" dialog.



4.10.2. Create the update of code of the Facade Session Bean - right click the code editor and select the Insert Code... item. In the Insert Code... form select the Call Enterprise Bean... item and in its form select the TTtitle_bookFacade, which is the local Enterprise Bean for persisting the TTtitle_book and TTtitle_book_on_tape entities.

The screenshot shows the NetBeans IDE interface with the following components visible:

- Projects Tab:** Shows the project structure with modules like Library2_EJB2-war.war and Library2_EJB2-ejb.
- Code Editors:** Two editors are open: FacadeRemote.java and Facade.java. Facade.java contains Java code for a session bean.
- Right-click Context Menu:** A context menu is open over the code in Facade.java, with the "Insert Code..." option highlighted.
- Insert Code... Dialog:** This dialog lists various generation options, with "Call Enterprise Bean..." selected.
- Call Enterprise Bean Dialog:** This dialog allows selecting an enterprise bean from open projects. It shows "Library2_EJB2-war" and "Library2_EJB2-ejb" expanded, with "TTtitle_bookFacade" selected under "Facade".
- Bottom Buttons:** OK, Cancel, and Help buttons are at the bottom of the dialogs.

Red arrows point from the "Insert Code..." item in the context menu to the "Call Enterprise Bean..." item in the first dialog, and from the "TTtitle_bookFacade" entry in the second dialog back to the "Call Enterprise Bean..." item in the first dialog.

```
package business_tier;
import integration_tier.TBookFacade;
import javax.ejb.EJB;
import javax.ejb.Stateless;
@Stateless
public class Facade implements FacadeRemote {
    @EJB
    private TBookFacade tBookFacade;
```



4.10.3. Create the update of code of the Facade class in the Library2_JPA2 project from sub_business_tier – the update_data method for initialization the content of the mTitle_books collection and the content of each mBooks collection of each instance of the TTitle_book class by using data from database

```
public synchronized void update_data(TTitle_book titles[], TBook books[]) {  
    mTitle_books.clear();  
    for (TTitle_book t : titles) {  
        mTitle_books.add(t);  
    }  
    for (TTitle_book title : mTitle_books) {  
        for (TBook book : books) {  
            TTitle_book title1 = book.getmTitle_book();  
            if (title1 != null) {  
                if (title1.equals(title)) {  
                    title.getmBooks().add(book);  
                }  
            }  
        }  
    }  
}
```



```

package business_tier;

import integration_tier.TBookFacade;
import integration_tier.TTitle_bookFacade;
import java.util.ArrayList;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.ejb.Stateless;
import sub_business_tier.TFacade;
import sub_business_tier.entities.TBook;
import sub_business_tier.entities.TTitle_book;

@Stateless
public class Facade implements FacadeRemote {
    @EJB
    private TTitle_bookFacade tTitle_bookFacade;
    @EJB
    private TBookFacade tBookFacade;
    private TTitle_book titles[];
    private TBook books[];

    TFacade facade = new TFacade(); ←

    @PostConstruct
    @Override
    public void init() {
        try {
            update_data();
        } catch (Exception e) { }
    }

    @Override
    public Object[][] gettitle_books()
    {
        return facade.gettitle_books();
    }

    @Override
    public String add_title_book(String data[])
    {
        return facade.add_title_book(data);
    }

    @Override
    public ArrayList<String> add_book(String data1[], String data2[])
    {
        return facade.add_book(data1, data2);
    }

    @Override
    public ArrayList<String> Search_title_book(String data[])
    {
        return facade.Search_title_book(data);
    }

    @Override
    public String Search_accessible_book(String data1[], Object data2)
    {
        return facade.Search_accessible_book(data1, data2);
    }
}

```

4.10.4. The same part of code of the Facade remote session bean from the project of lab3

The new method for initial activities od Session Bean



4.10.5. Update code of the Facade remote session bean from the project of lab3 – for accessing the database by using the TTITLE_bookFacade and TBookFacade

```
//.....  
public void update_titles() throws Exception { titles = tTitle_bookFacade.getTTITLE_books_(); }  
  
public void update_books() throws Exception { books = tBookFacade.getTBooks_(); }  
  
public void update_data() throws Exception {  
    update_titles();  
    update_books();  
    facade.update_data(titles, books); }  
  
public void add_titles() throws Exception { tTitle_bookFacade.addTTITLE_books(facade.getmTitle_books());}  
}  
  
public void add_books() throws Exception { tBookFacade.addTBooks(facade.getmTitle_books()); }  
  
public ArrayList<ArrayList<String>> titles() throws Exception {  
    List<TTITLE_book> help1 = tTitle_bookFacade.findAll();  
    ArrayList<ArrayList<String>> help2;  
    help2 = new ArrayList();  
    for (TTITLE_book t : help1) {  
        ArrayList<String> help3 = new ArrayList();  
        help3.add(t.getPublisher());  
        help3.add(t.getISBN());  
        help3.add(t.getTitle());  
        help3.add(t.getAuthor());  
        help3.add(t.getActor());  
        help2.add(help3);  
    }  
    return help2;  
}
```



4.10.6. After Fix Imports.. process (right click and select the Fix Imports... item. In the form of Fix Imports... you must select the shown imports.

Library2_EJB2-ejb - NetBeans IDE 8.0.2

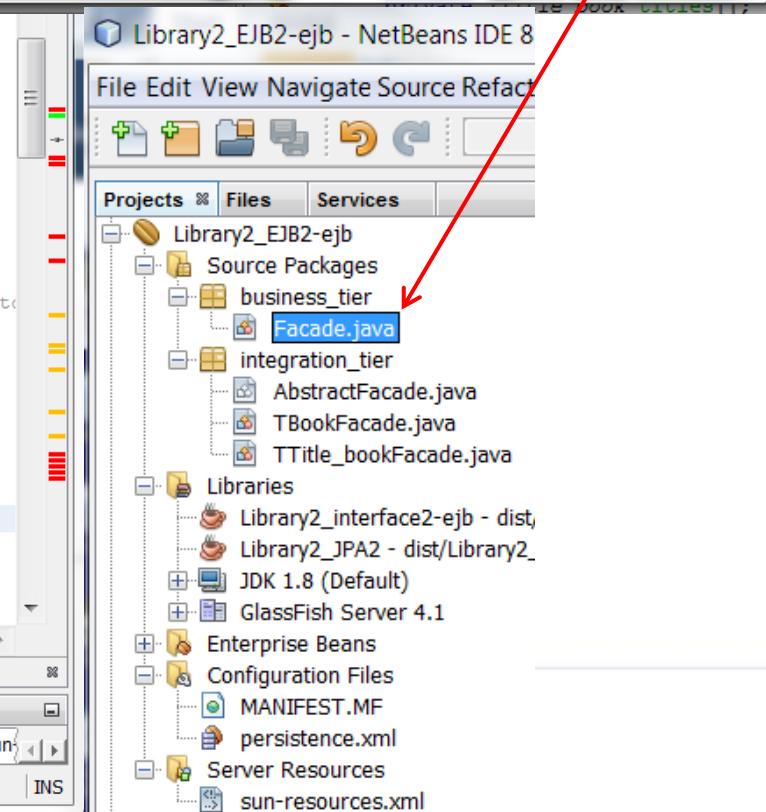
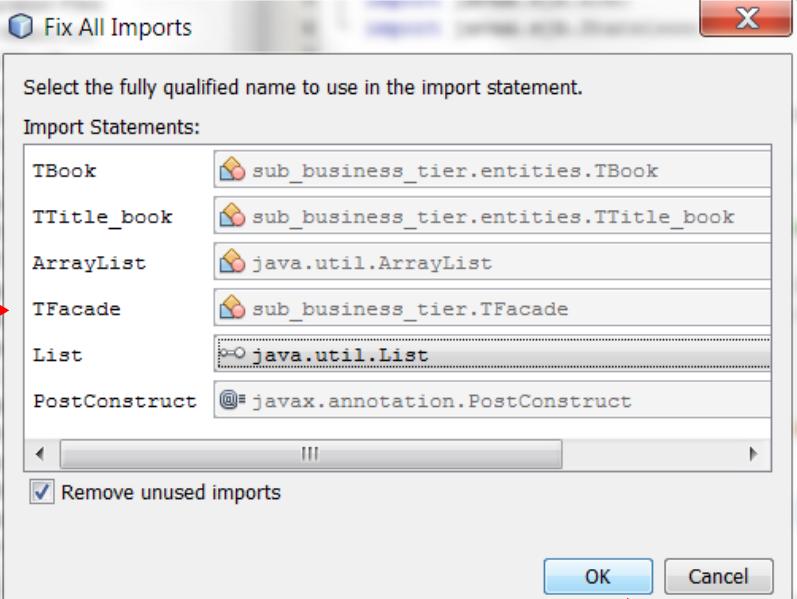
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)

Projects Files Services

Source Packages business tier Facade.java

```
4 import integration_tier.TTitle_bookFacade;
5 import javax.ejb.EJB;
6 import javax.ejb.Stateless;
7
8 @Stateless
9 public class Facade implements FacadeRemote {
10     @EJB
11     private TTitle_bookFacade tTitle_bookFacade;
12     @EJB
13     private TBookFacade tBookFacade;
14
15     // Add business logic below. (Right-click in editor)
16     // "Insert Code > Add Business Method"
17     @PostConstruct
18     public void init() {
19         try {
20             System.out.println("update1");
21             update_data();
22         } catch (Exception e) {
23
24         }
25     }
26     TFacade facade = new TFacade();
```

Output > GlassFish Server 4.1 > Java DB Database Process > Library2_EJB2 (run) 23:1 INS





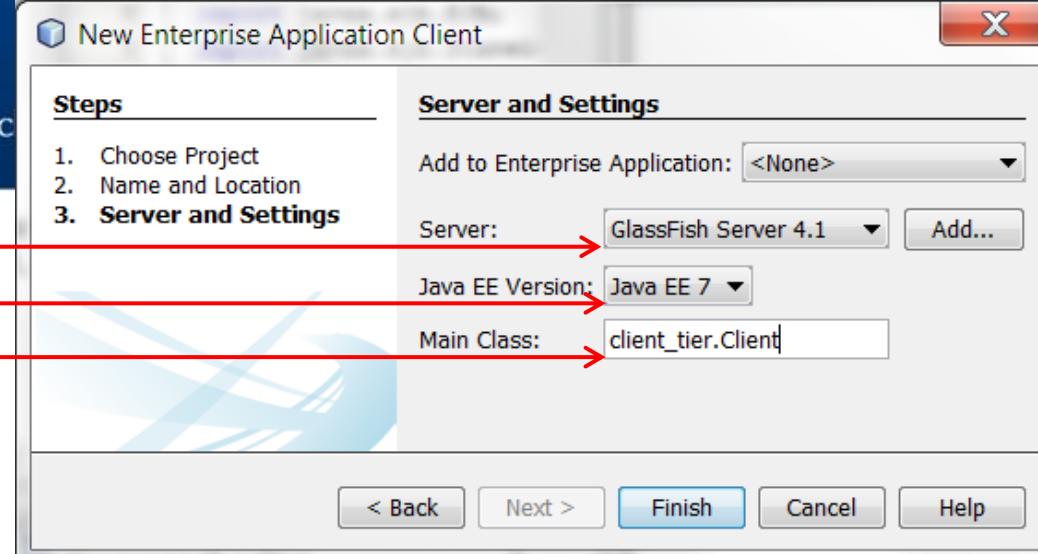
5./5.1.1. Create the new Enterprise Application Client (the based on code of the Library1_client1-ejb project of lab3) – click File/New Project.../Java EE/ Enterprise Application Client; fill name of project as Library2_Client2-ejb and select the location of the project (the same as the location of other projects)

The screenshot shows the NetBeans IDE 8.0.2 interface with three windows open:

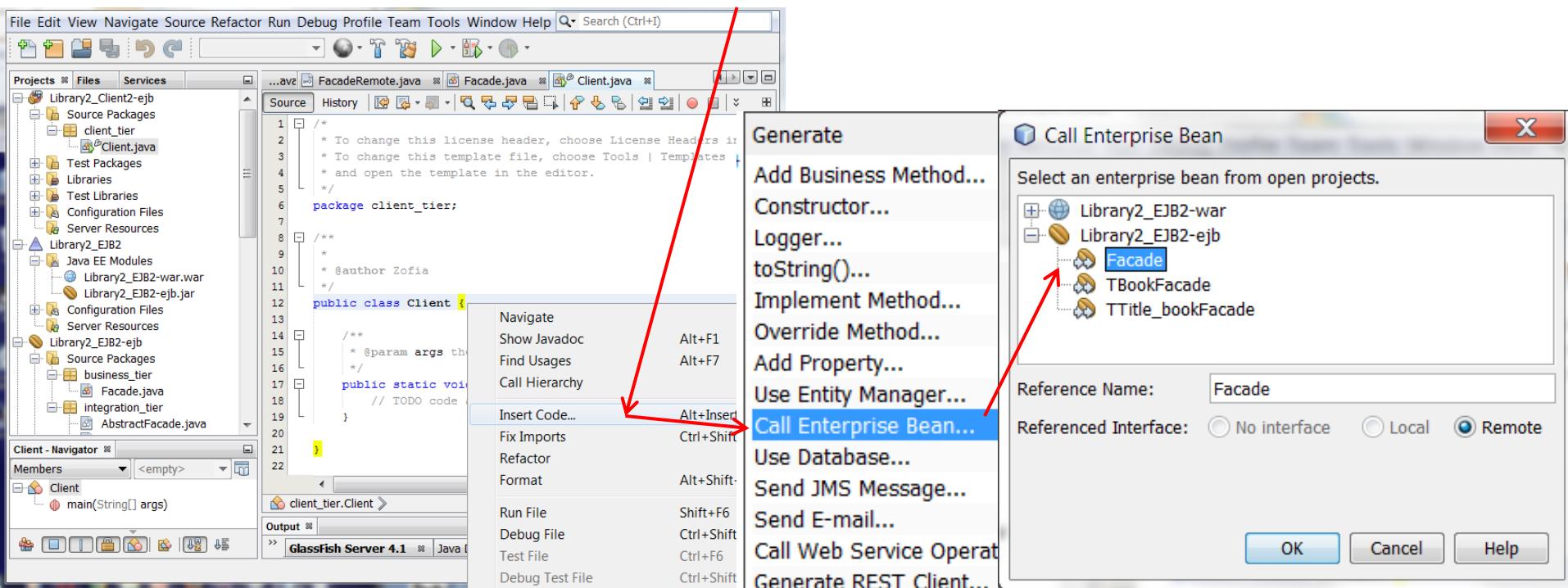
- NetBeans IDE 8.0.2 Window:** Shows the main menu bar with "File" selected. A red arrow points to the "New Project..." option under the "File" menu.
- Choose Project Window:** A dialog box titled "Choose Project". It displays a list of categories: Java, JavaFX, UML, Java Web, Java EE, HTML5, and Java ME Embedded. Under the Java EE category, "Enterprise Application Client" is selected and highlighted with a red arrow. A list of project types is shown on the right, with "Enterprise Application Client" also being highlighted.
- New Enterprise Application Client Window:** A dialog box titled "New Enterprise Application Client". It has two tabs: "Steps" and "Name and Location". The "Steps" tab shows the process: 1. Choose Project, 2. Name and Location, 3. Server and Settings. The "Name and Location" tab is active, showing fields for "Project Name" (set to "Library2_Client2-ejb"), "Project Location" (set to "C:\EnglishLecture\Kruczkiewicz\Laboratory2015\lab4"), and "Project Folder" (set to "e\Kruczkiewicz\Laboratory2015\lab4\Library2_Client"). Red arrows point from the "Project Name" field to the "Name and Location" tab and from the "Project Location" field to the "Name and Location" tab.



5.1.2. Continuation – select the application server GlassFish 4.1, Java EE 7 platform and fill the name of main class – the same package and name of class as in the SE client project.



5.1.3. Create code of the Client class - right click the code editor and select the Insert Code item. In the InsertCode form select the Call Enterprise Bean.. item and in its form select the Facade, which is the remote Enterprise Bean from the Library2_EJB2-ejb module





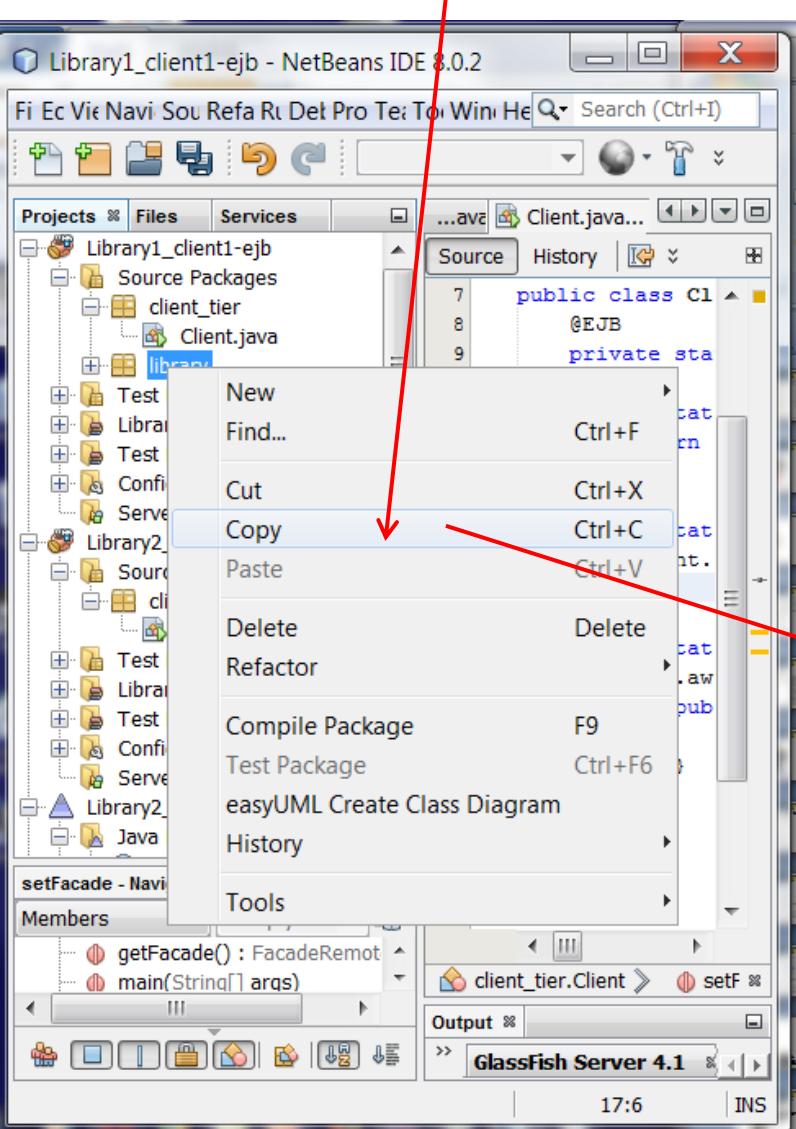
5.1.4. Create the getter and setter methods for the reference of the facade session bean - right click the code editor and select the Insert Code... item. In the Insert Code form select the Getter and Setter... item and in its form select the facade attribute of theClient class. Click Generate.

The screenshot shows the NetBeans IDE interface with the following details:

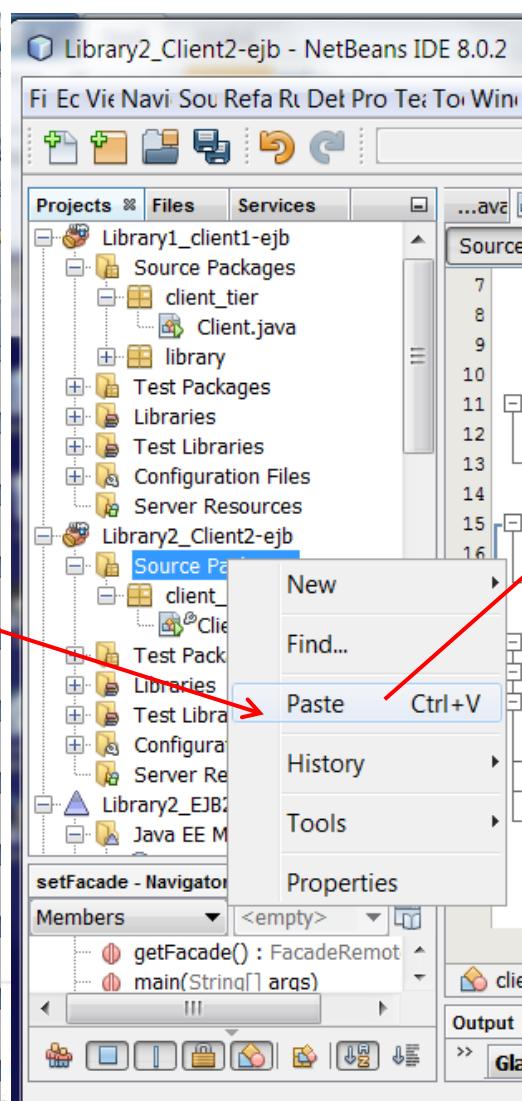
- Project Structure:** The left sidebar shows two projects: "Library2_Client2-ejb" and "Library2_EJB2".
- Code Editor:** The main window displays a Java file named "Client.java" with the following code:

```
10  /*
11  *
12  * @author Zofia
13  */
14
15  public class Client {
16      @EJB
17      private static FacadeRemote facade;
18
19      /**
20       * @param args the
21       */
22      public static void main(String[] args) {
23          // TODO code a
24      }
25  }
```
- Context Menu:** A context menu is open over the line containing the `facade` field, with the "Insert Code..." option highlighted by a red arrow.
- Insert Code Dialog:** A dialog titled "Generate" is open, listing various options like Constructor..., Logger..., and Getter... (which is also highlighted by a red arrow). The "Getter and Setter..." option is selected.
- Generate Getters and Setters Dialog:** This dialog is also open, showing a list of fields to generate getters and setters for. The "Client" checkbox is checked, and the "facade : FacadeRemote" checkbox is also checked and highlighted by a red arrow. There is an "Encapsulate Fields" checkbox at the bottom left.
- Bottom Status Bar:** The status bar at the bottom indicates "Project co-financed from the EU European Social Fund".

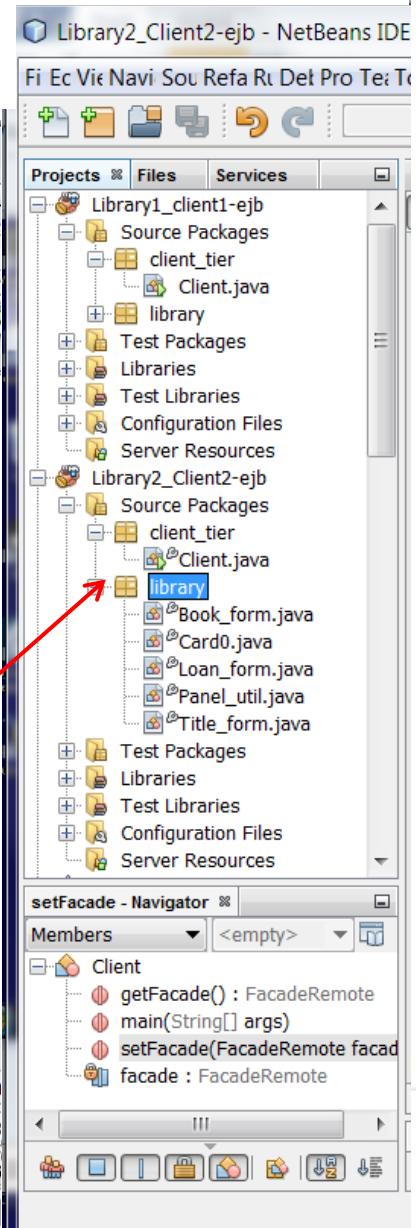
5.1.5. Create the copy of the library package from the Library1_client1-ejb project (lab3) – right click the library package item of this project and select the Copy item



5.1.6. Right click the Source Packages of the Library2_Client2-ejb project and select the Paste item



5.1.7. The result





```
package client_tier;

import business_tier.FacadeRemote;
import javax.ejb.EJB;
import library.Panel_util;

public class Client {
    @EJB
    private static FacadeRemote facade;

    public static FacadeRemote getFacade()
    { return facade; }

    public static void setFacade(FacadeRemote facade)
    { Client.facade = facade; }

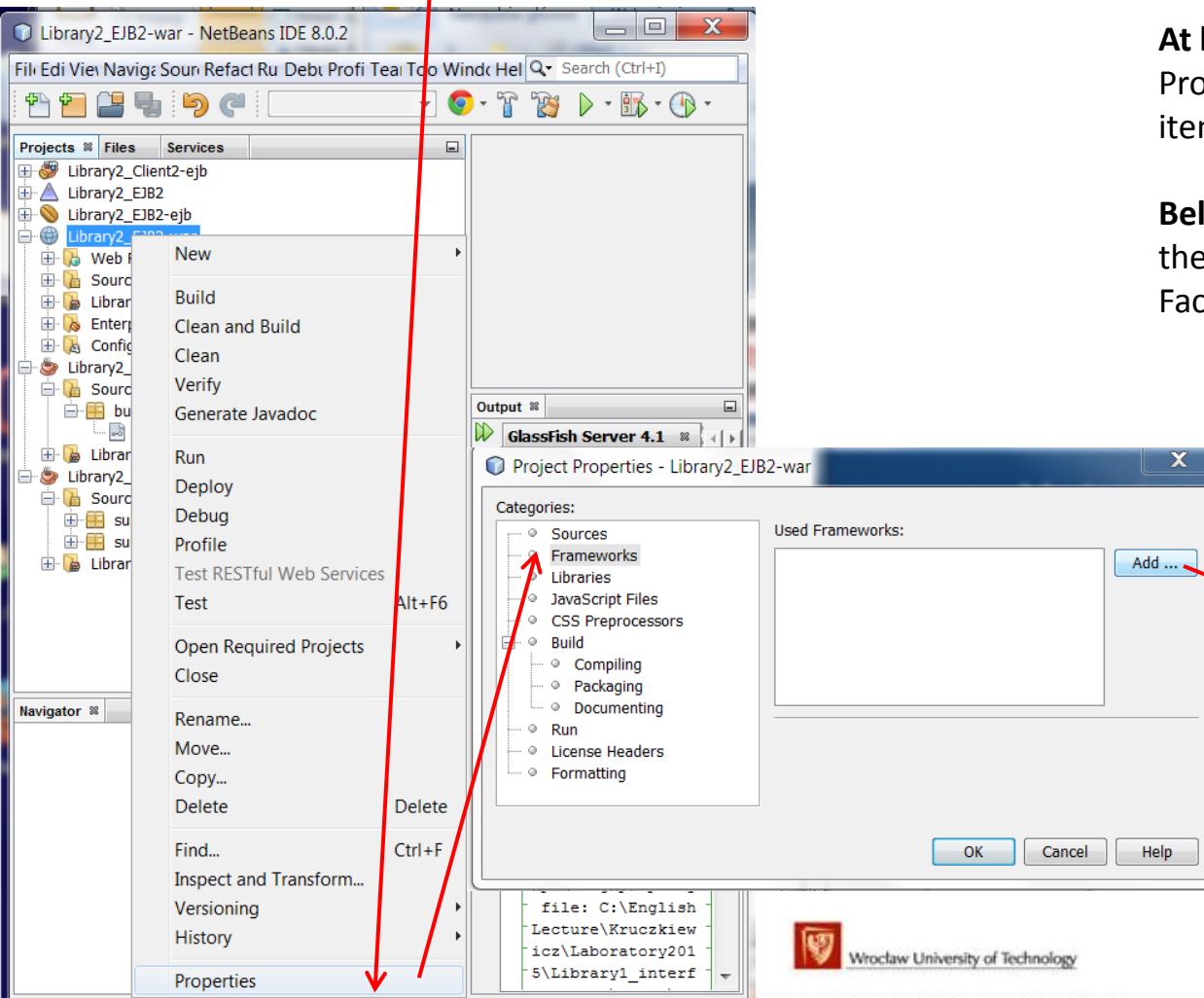
    public static void main(String[] args) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                Panel_util.createAndShowGUI();
            }
        });
    }
}
```

5.1.8. Code of the Client class
Library2_Client2-ejb project – the
same as the code of lab3 client
project



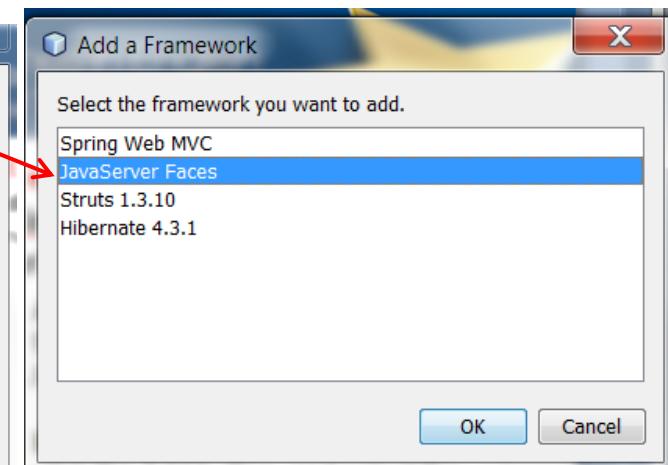


6. /6.1. Definition of the Web Client Tier (**Library2_EJB2-war**, prepared during creation of **Library2_EJB2** project) for inserting application data into the database and displaying the data from database – change the Web Framework to the JavaServer Faces type



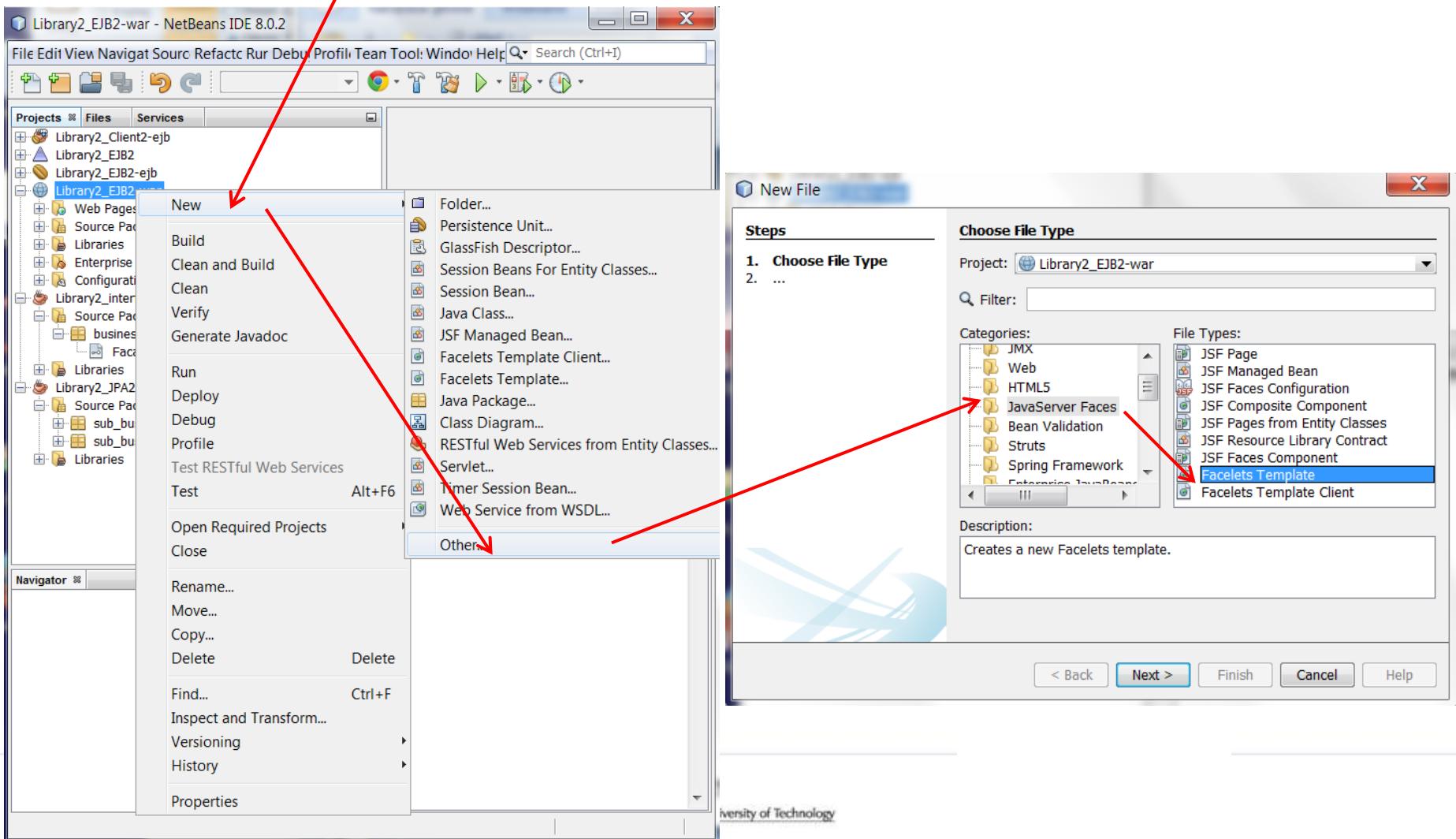
At left: Right click the name of project in the Projects tab, and choose the Properties item

Below: Choose the Framework item, click the Add button, and select the JavaServer Faces framework



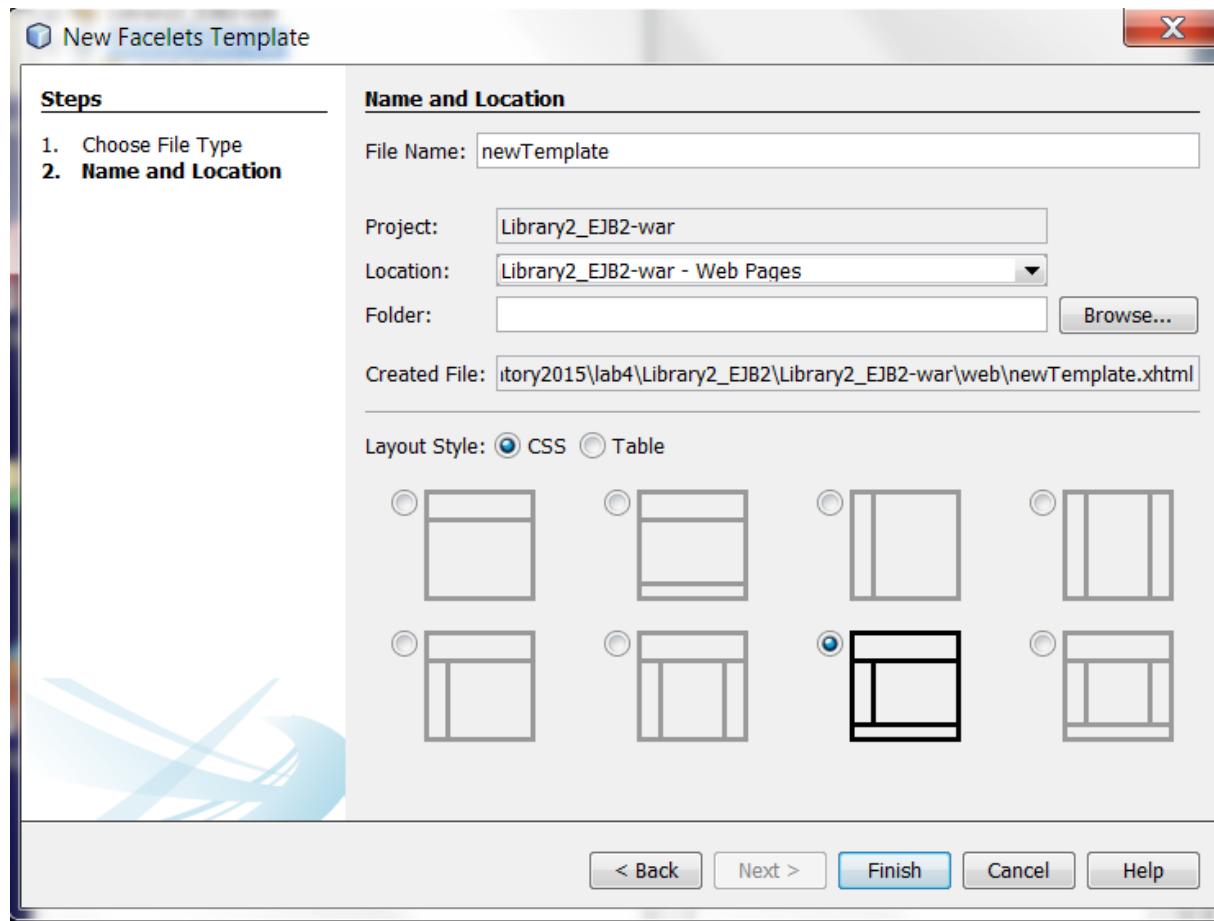


6.2. /6.2.1. Addition the template of web pages: Right click the name of project in the Projects tab, choose the New/Other items, then select JavaServer Faces File type, and the Facelets Template file. Click Next.





6.2.2. Below: choose the template and set the name of template file. Click Finish.





Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigate Source Projects Files Services newTemplate.xhtml

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/1999/xhtml"
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:h="http://java.sun.com/jsf/html">

    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <h:outputStylesheet name=".css/default.css"/>
        <h:outputStylesheet name=".css/cssLayout.css"/>
        <title>Facelets Template</title>
    </head>
    <body>
        <div id="top">
            <ui:insert name="top">Top</ui:insert>
        </div>
        <div>
            <div id="left">
                <ui:insert name="left">Left</ui:insert>
            </div>
            <div id="content" class="left_content">
                <ui:insert name="content">Content</ui:insert>
            </div>
        </div>
        <div id="bottom">
            <ui:insert name="bottom">Bottom</ui:insert>
        </div>
    </body>
</html>
```

Navigator CSS HTML html XHTML

GlassFish Server 4.1 Java DB Database Process Library1_client1-ejb (run) Library1_client1-ejb (r)

29:1 INS

6.3/6.3.1. Addition the template of web pages: Change the the content of attributes of the **h:head** tag and **html** tag



```
<?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">

    <h:head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <h:outputStylesheet name="css/default.css" />
        <h:outputStylesheet name="css/cssLayout.css"/>
        <title>Library</title>
    </h:head>

    <h:body>
        <div id="top">
            <ui:insert name="top">Top</ui:insert>
        </div>
        <div>
            <div id="left">
                <h:link outcome="/faces/presentation_tier_view/Store_data" value="Store data"/><br/>
                <h:link outcome="/faces/presentation_tier_view>Show_data" value="Show data"/>
            </div>
            <div id="content" class="left_content">
                <ui:insert name="content">Content</ui:insert>
            </div>
        </div>
        <div id="bottom">
            <ui:insert name="bottom">Bottom</ui:insert>
        </div>
    </h:body>
</html>
```

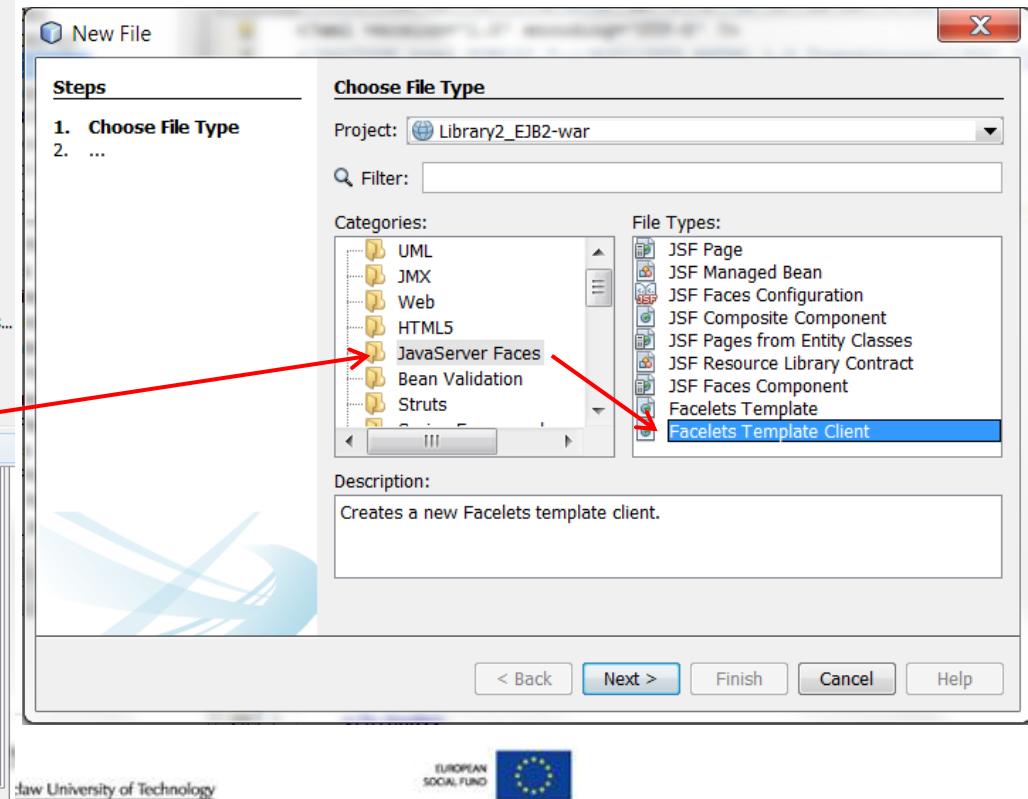
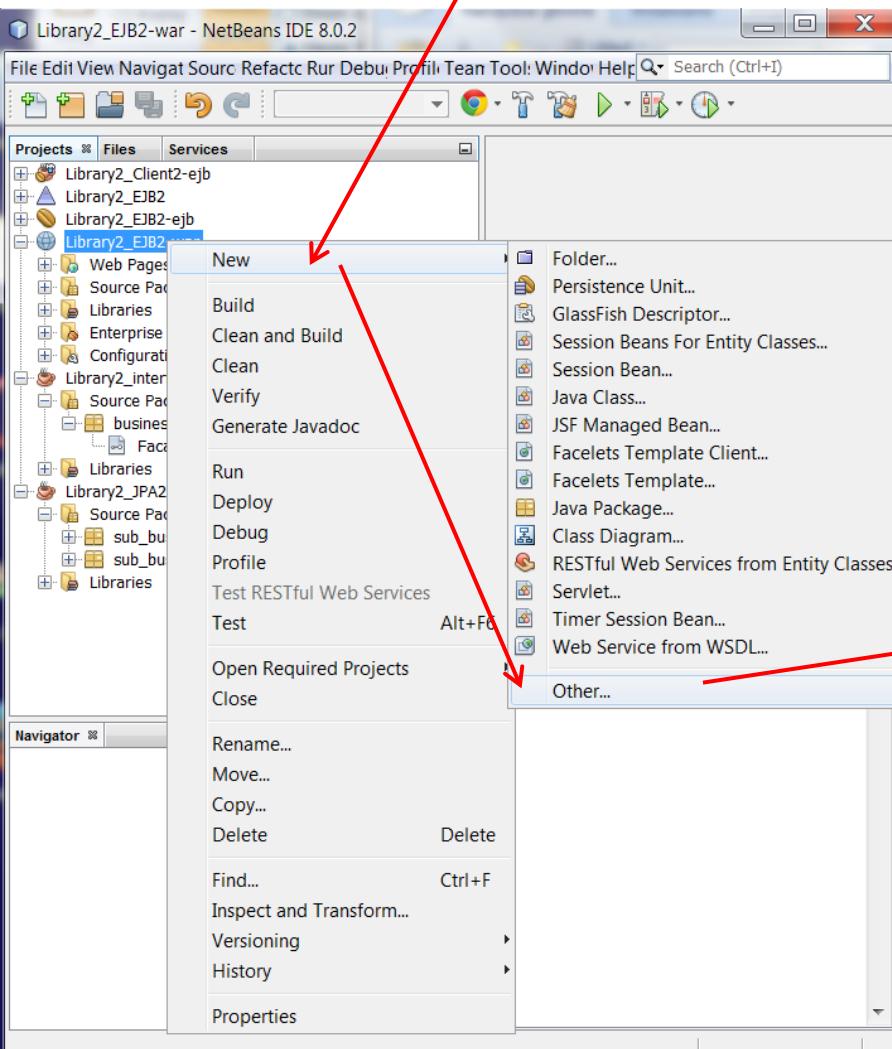
6.3.2. Addition the template of web pages:

Insert the new code into the div tag with id equals left – this will be a menu of all web pages based on this template





6.4./ 6.4.1. Creation the main JSF page (index2), based on the previous defined template – right click the Library2_EJB2-war item and select the New/Other... items. In the New File dialog choose the JavaServer Faces/Facelets Template Client options. Click Next.





New Facelets Template Client

Steps

1. Choose File Type
2. Name and Location

Name and Location

File Name: ←

Project: Library2_EJB2-war

Folder: Browse...

Created File: aboratory2015\lab4\Library2_EJB2\Library2_EJB2-war\web\index2.xhtml

Template: Browse...

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

Sections To Generate:

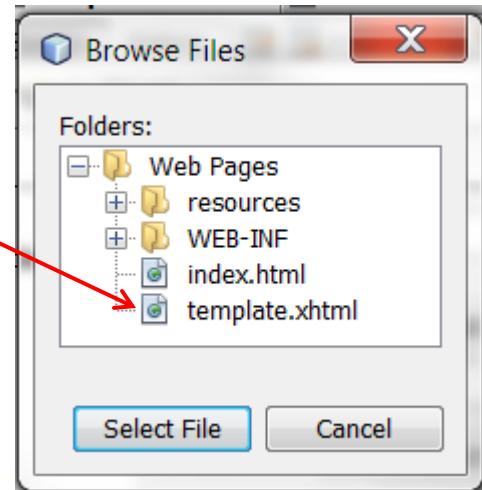
Select valid or non-empty template... ←

⚠ Select a template for which the client will be generated.

< Back Next > Finish Cancel Help

6.4.2. Creation the main JSF page (index2), based on the previous defined template – continuation

Below: choose the proper template – click Select File in the Browse Files dialog





6.4.3. Creation the main JSF page (index2), based on the previous defined template – continuation

Below: after choosing the proper template

New Facelets Template Client

Steps

1. Choose File Type
2. Name and Location

Name and Location

File Name: index2

Project: Library2_EJB2-war

Folder:

Created File: aboratory2015\lab4\Library2_EJB2\Library2_EJB2-war\web\index2.xhtml

Template: _EJB2/Library2_EJB2-war/web/template.xhtml

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

Sections To Generate:

<input checked="" type="checkbox"/>	top
<input checked="" type="checkbox"/>	left
<input checked="" type="checkbox"/>	content
<input checked="" type="checkbox"/>	bottom

< Back



6.4.4. Creation the main JSF page (index2), based on the previous defined template – the result

At left: the view of the unused JSF and JSP main pages (index.jsp and index.xhtml);

At right: after deleting unused pages - the index2.xhtml is the main JSF page

Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)

Projects Files Services

template.xhtml index.html index2.xhtml

Source History

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional
http://www.w3.org/1999/xhtml"
xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
<body>
    <ui:composition template="./template.xhtml">
        <ui:define name="top">
            top
        </ui:define>
        <ui:define name="left">
            left
        </ui:define>
        <ui:define name="content">
            content
        </ui:define>
        <ui:define name="bottom">
            bottom
        </ui:define>
    </ui:composition>
</body>
</html>
```

Navigator

html body

GlassFish Server 4.1 Java DB Database Process Library1_client1-ejb (ru) 20:1 INS

Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Pr

Projects Files Services

index2.xhtml

Source History

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional
http://www.w3.org/1999/xhtml"
xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
<body>
    <ui:composition template="./template.xhtml">
        <ui:define name="top">
            top
        </ui:define>
        <ui:define name="left">
            left
        </ui:define>
        <ui:define name="content">
            content
        </ui:define>
        <ui:define name="bottom">
            bottom
        </ui:define>
    </ui:composition>
</body>
</html>
```

Output

GlassFish Server 4.1 Java DB Database Process Library1_client1-ejb (ru) 20:1 INS



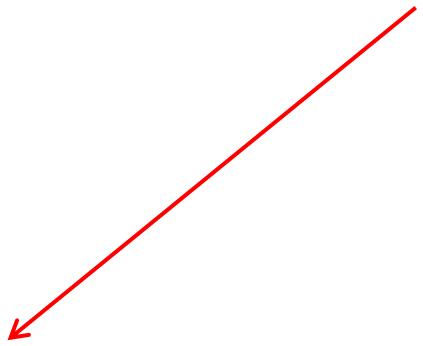
```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"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/faces"
      xmlns:h="http://java.sun.com/jsf/html">
```

```
<body>
    <ui:composition template=".//template.xhtml">
        <ui:define name="top">
            top
        </ui:define>
        <ui:define name="left">
            <h:link outcome="/faces/presentation_tier_view/Store_data" value="Store data"/><br/>
            <h:link outcome="/faces/presentation_tier_view>Show_data" value="Show data"/>
        </ui:define>
        <ui:define name="content">
            content
        </ui:define>
        <ui:define name="bottom">
            bottom
        </ui:define>
    </ui:composition>
</body>
</html>
```

6.4.5. Add the h:link tags in the left part of template





6.4.6. Set up the main web page as the index2.xhtml – in the web.xml file (the descriptor of the web module).

The screenshot shows the NetBeans IDE 8.0.2 interface with the project `Library2_EJB2-war` open. The left pane displays the project structure, including the `WEB-INF/web.xml` file under the `Web Pages` folder. The right pane shows the XML content of the `web.xml` file. A red arrow points to the line containing the `<welcome-file>` tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi="http://xmlns.jcp.org/xml/ns/javaee xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">
    <context-param>
        <param-name>javax.faces.PROJECT_STAGE</param-name>
        <param-value>Development</param-value>
    </context-param>
    <servlet>
        <servlet-name>Faces Servlet</servlet-name>
        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>/faces/*</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>faces/index2.xhtml</welcome-file>
    </welcome-file-list>
</web-app>
```



New Facelets Template Client

Steps

1. Choose File Type
2. Name and Location

Name and Location

File Name: Store_data

Project: Library2_EJB2-war

Folder: presentation_tier_view

Created File: _EJB2\Library2_EJB2-war\web\presentation_tier_view\Store_data.xhtml

Template: _EJB2\Library2_EJB2-war\web\template.xhtml

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

Sections To Generate:

<input type="checkbox"/>	top
<input type="checkbox"/>	left
<input checked="" type="checkbox"/>	content
<input type="checkbox"/>	bottom

< Back Cancel Help

New Facelets Template Client

Steps

1. Choose File Type
2. Name and Location

Name and Location

File Name: Show_data

Project: Library2_EJB2-war

Folder: presentation_tier_view

Created File: _EJB2\Library2_EJB2-war\web\presentation_tier_view>Show_data.xhtml

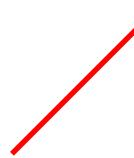
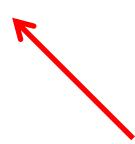
Template: _EJB2\Library2_EJB2-war\web\template.xhtml

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

Sections To Generate:

<input type="checkbox"/>	top
<input type="checkbox"/>	left
<input checked="" type="checkbox"/>	content
<input type="checkbox"/>	bottom

< Back Cancel Help



6.5. /6.5.1. Addition the two JSF pages, based on the previous defined template for **inserting data in the database** and **displaying the data from databases**





6.5.2. The new JSF page for inserting data in the database – Store_data.xhtml

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/1999/xhtml"
  xmlns:ui="http://java.sun.com/jsf/faces"
  xmlns:h="http://java.sun.com/jsf/html">

<body>
  <ui:composition template="../../template.xhtml">
    <ui:define name="content">
      <h:form>
        <h:commandButton action="#{managed_Bean1.store_data}" value="Store data"/><br/>
      </h:form>
    </ui:define>
  </ui:composition>
</body>
</html>
```

Tag for calling the store_data method from managed_Bean1 object (the Managed Bean type) – its definition is presented further part of this instruction





6.5.3. The definition of code of the Store_data.xhtml JSF page.

```
<?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="content">
<h:form>
    <h:commandButton action="#{managed_Bean1.store_data}"
                     value="Store data"/><br/>
</h:form>
</ui:define>
</ui:composition>
</body>
</html>
```



6.5.4. The code of the second JSF page for displaying data from the database , using the h: dataTable JSF component – the Show_data.xhtml page

Library2_EJB2-war - NetBeans IDE 8.0.2

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

Search (Ctrl+I)

Projects Files Services

Source History

SQL 1 [jdbc:derby://localhost:15...]

template.xhtml

index2.xhtml

web.xml

Show_data.xhtml

Show_data.xhtml

Configuration Files

Server Resources

Library2_EJB2

Java EE Modules

Library2_EJB2-war.war

Library2_EJB2-ejb.jar

Configuration Files

Server Resources

Library2_EJB2-ejb

Source Packages

Libraries

Enterprise Beans

Configuration Files

Server Resources

Library2_EJB2-war

Web Pages

WEB-INF

presentation_tier_view

Source

```
<?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"
      xmlns:f="http://java.sun.com/jsf/core">

    <body>
        <ui:composition template="../../template.xhtml">
            <ui:define name="content">
                <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="Lista produktów pusta" rendered="#{managed_Bean1.items.rowCount == 0}"/>
                        <h:panelGroup rendered="#{managed_Bean1.items.rowCount > 0}">
                            <h:dataTable value="#{managed_Bean1.items}" var="item" border="0"
                                         cellpadding="2" cellspacing="0" rowClasses="jsfcrud_odd_row,jsfcrud_even_row"
                                         rules="all" style="border:solid 1px">
                                <h:column>
                                    <f:facet name="header">
                                        <h:outputText value="Publisher"/>
                                    </f:facet>
                                    <h:outputText value="#{item.get(0)}"/>
                                </h:column>
```

Navigator

Classes

.jsfcrud_even_row

.jsfcrud_list_form

.jsfcrud_odd_row

Elements

SELECTOR

Ids

#messagePanel

HTML

body

Output

Java DB Database Process

GlassFish Server 4.1

SQL 1 execution

Library2_Client2-ejb (run)

Library2_Client2-ejb (run) running... 53:7 INS



6.5.5. The code of the second JSF page for displaying data from the database - continuation

Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)

Projects Files Services ...tm web.xml Store_data.xhtml Show_data.xhtml

Source History

```
<h:column>
    <f:facet name="header">
        <h:outputText value="ISBN"/>
    </f:facet>
    <h:outputText value="#{item.get(1)}"/>
</h:column>
<h:column>
    <f:facet name="header">
        <h:outputText value="Title"/>
    </f:facet>
    <h:outputText value="#{item.get(2)}"/>
</h:column>
<h:column>
    <f:facet name="header">
        <h:outputText value="Author"/>
    </f:facet>
    <h:outputText value="#{item.get(3)}"/>
</h:column>
<h:column>
    <f:facet name="header">
        <h:outputText value="Actor"/>
    </f:facet>
    <h:outputText value="#{item.get(4)}"/>
</h:column>
</h:DataTable>
</h:panelGroup>
</h:form>
</ui:define>
</ui:composition>
</body>
</html>
```

Navigator

- CSS
 - Classes
 - .jsfcrud_even_row
 - .jsfcrud_list_form
 - .jsfcrud_odd_row
 - Elements
 - SELECTOR
 - Ids
 - #messagePanel
 - HTML
 - html
 - body
 - XHTML

Output

 - Java DB Database Process
 - GlassFish Server 4.1
 - SQL 1 execution
 - Library2_Cl

Library2_Client2-ejb (run) running... 48:27 INS



6.5.6. The code of the second JSF page for displaying data from the database - continuation

```
<?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"
      xmlns:f="http://java.sun.com/jsf/core">
<body>
<ui:composition template=".//template.xhtml">
<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="Lista_produktow_pusta" rendered="#{managed_Bean1.items.rowCount == 0}"/>
  <h:panelGroup rendered="#{managed_Bean1.items.rowCount > 0}">
    <h:dataTable value="#{managed_Bean1.items}" var="item" border="0"
                 cellpadding="2" cellspacing="0" rowClasses="jsfcrud_odd_row,jsfcrud_even_row" rules="all" style="border:solid 1px">
      <h:column>
        <f:facet name="header"> <h:outputText value="Publisher"/> </f:facet>
        <h:outputText value="#{item.get(0)}"/>
      </h:column>
      <h:column>
        <f:facet name="header"> <h:outputText value="ISBN"/> </f:facet>
        <h:outputText value="#{item.get(1)}"/>
      </h:column>
```



6.5.7. The code of the second JSF page for displaying data from the database - continuation

```
<h:column>
    <f:facet name="header">
        <h:outputText value="Title"/>
    </f:facet>
    <h:outputText value="#{item.get(2)}"/>
</h:column>
<h:column>
    <f:facet name="header">
        <h:outputText value="Author"/>
    </f:facet>
    <h:outputText value="#{item.get(3)}"/>
</h:column>
<h:column>
    <f:facet name="header">
        <h:outputText value="Actor"/>
    </f:facet>
    <h:outputText value="#{item.get(4)}"/>
</h:column>
</h:dataTable>
</h:panelGroup>
</h:form>
</ui:define>
</ui:composition>
</body>
</html>
```



Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Tear Tools Window Help Search (Ctrl+I)

Projects Files Services

Source History

...tm web.xml Store_data.xhtml Show_data.xhtml

27 <h:column>

28 <f:facet name="header">

29 <h:outputText value="ISBN"/>

30 </f:facet>

31 <h:outputText value="#{item.get(1)}"

32 </h:column>

33 <h:column>

34 <f:facet name="header">

35 <h:outputText value="Title"/>

36 </f:facet>

37 <h:outputText value="#{item.get(2)}"

38 </h:column>

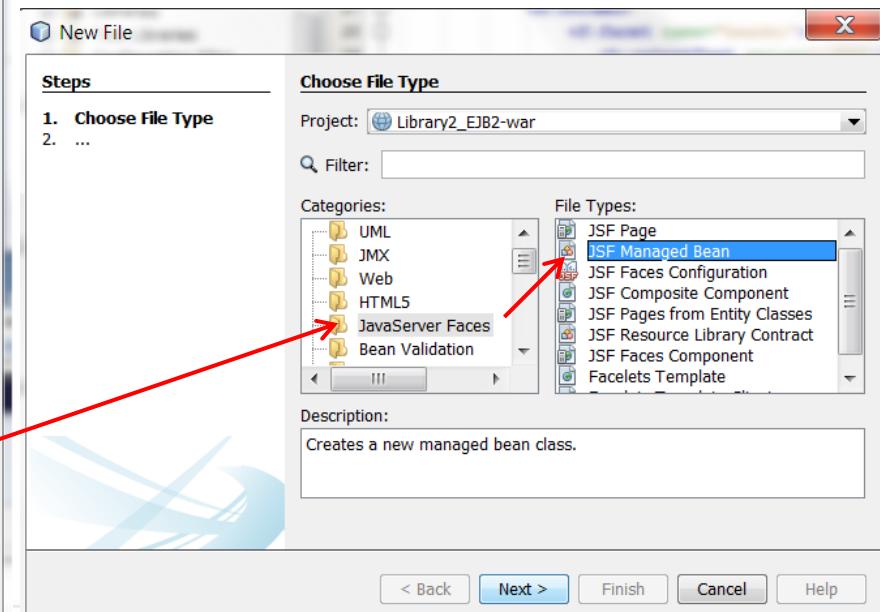
39 <h:column>

40 <f:facet name="header">

New Build Clean and Build Clean Verify Generate Javadoc Run Deploy Debug Profile Test RESTful Web Services Test Open Required Projects Close Rename... Move... Alt+F6

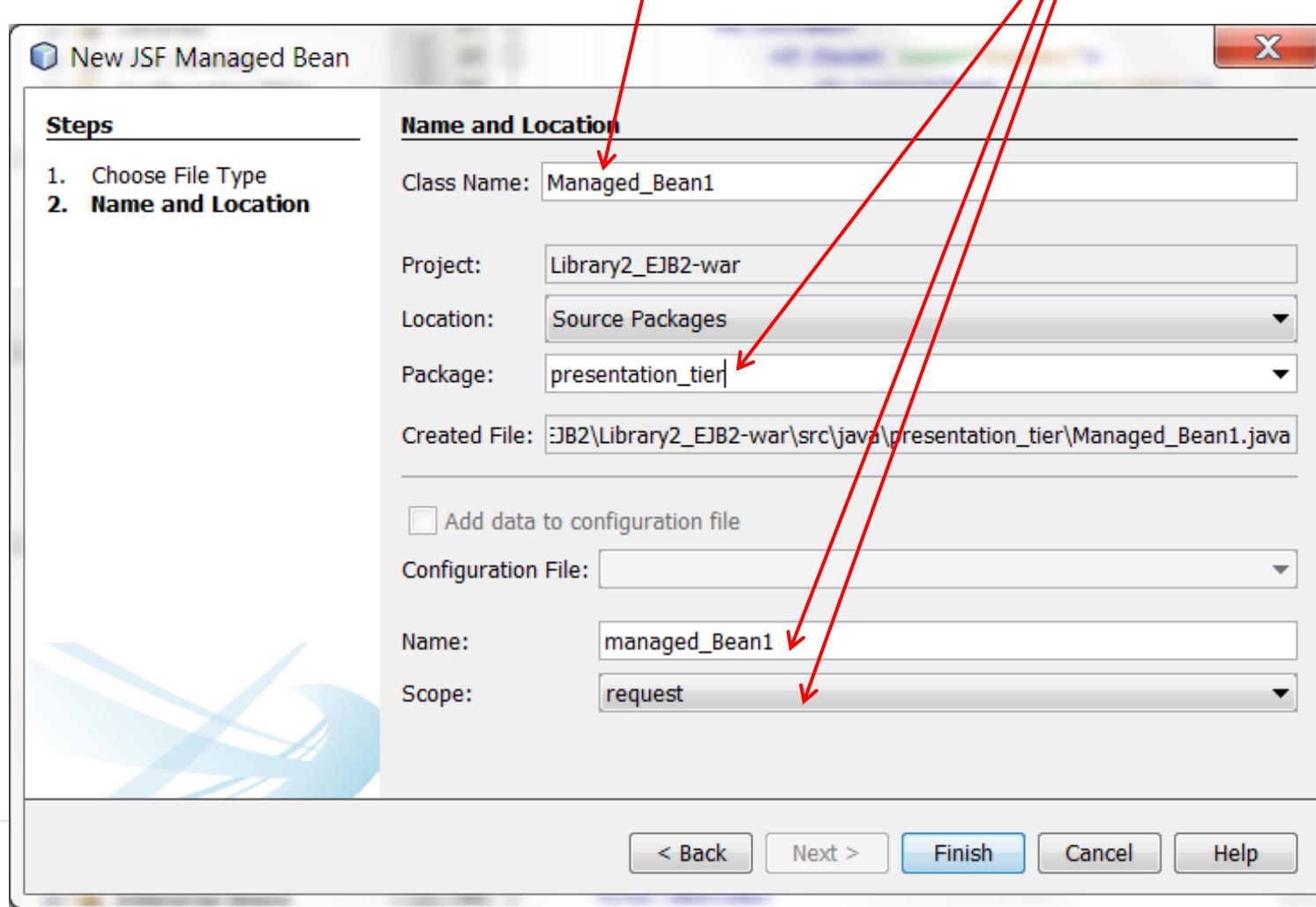
GlassFish Server 4.1 running... SQL 1 execution 48:27 INS

6.6. /6.6.1. Creation of the Managed Bean type of object as the Presentation Tier object based on JSF technology – right click the Library2_EJB2-war item in the Projects item, select the JavaServer Faces/JSF Managed Bean File Types. Click Next.





6.6.2. Creation the Managed Bean type of object as the Presentation Tier object based on JSF technology - continuation. Fill the Class Name field, name of package in the Package field, setup the scope of Managed Bean and fill its name in the name field.





6.6.3. Creation of the Managed Bean type of object as the Presentation Tier object based on JSF technology - the generated code of this class

Library2_EJB2-war - NetBeans IDE 8.0.2

File Edit View Navigator Sources Refactor Run Debug Profiles Team Tools Windows Help Search (Ctrl+I)

Projects Files Services

...tm Show_data.xhtml Managed_Bean1.java

Source History

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package presentation_tier;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;

/**
 *
 * @author Zofia
 */
@ManagedBean
@RequestScoped
public class Managed_Bean1 {

    /**
     * Creates a new instance of Managed_Bean1
     */
    public Managed_Bean1() {
    }

}
```

Navigator

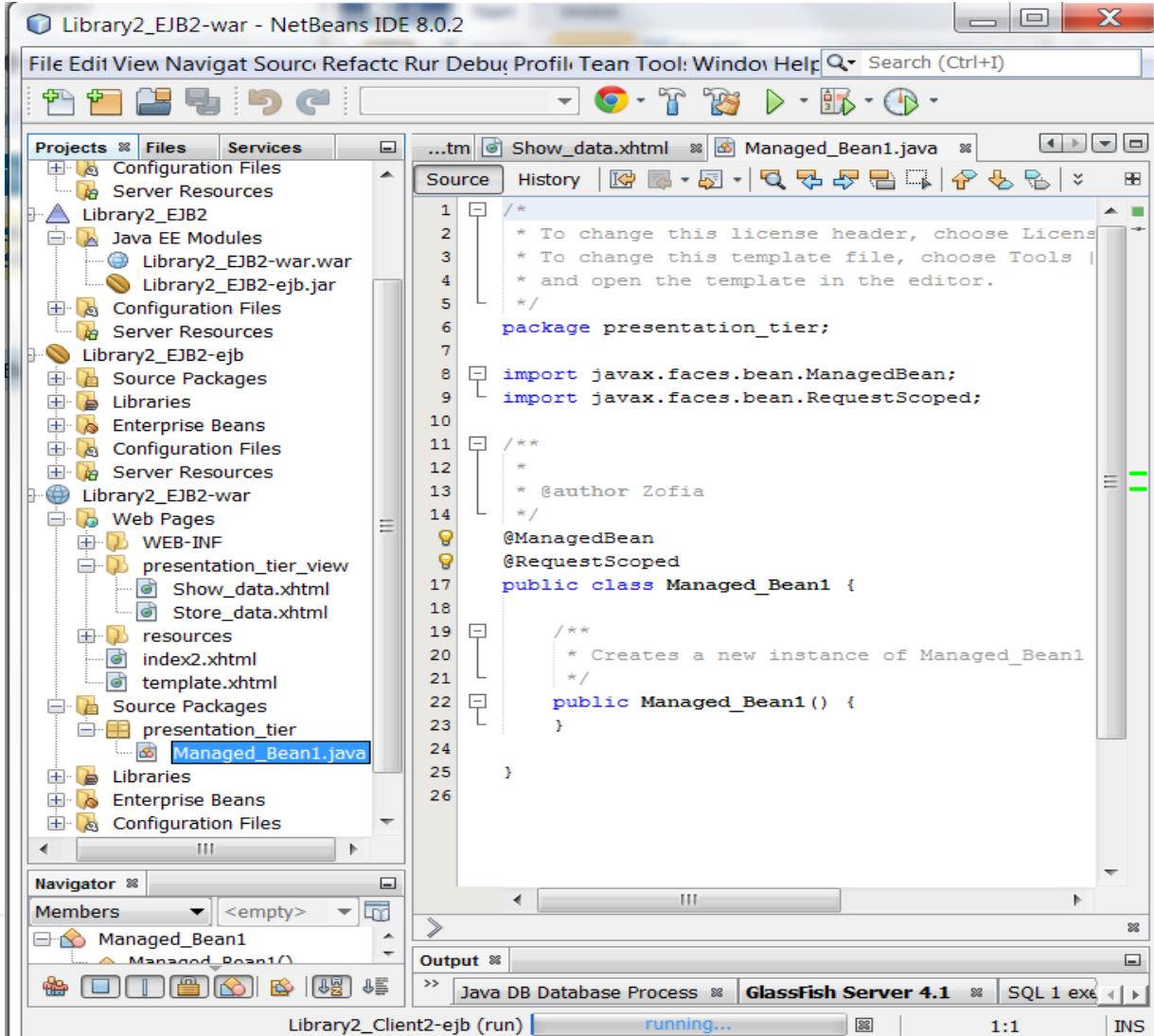
Members <empty>

Managed_Bean1

Output

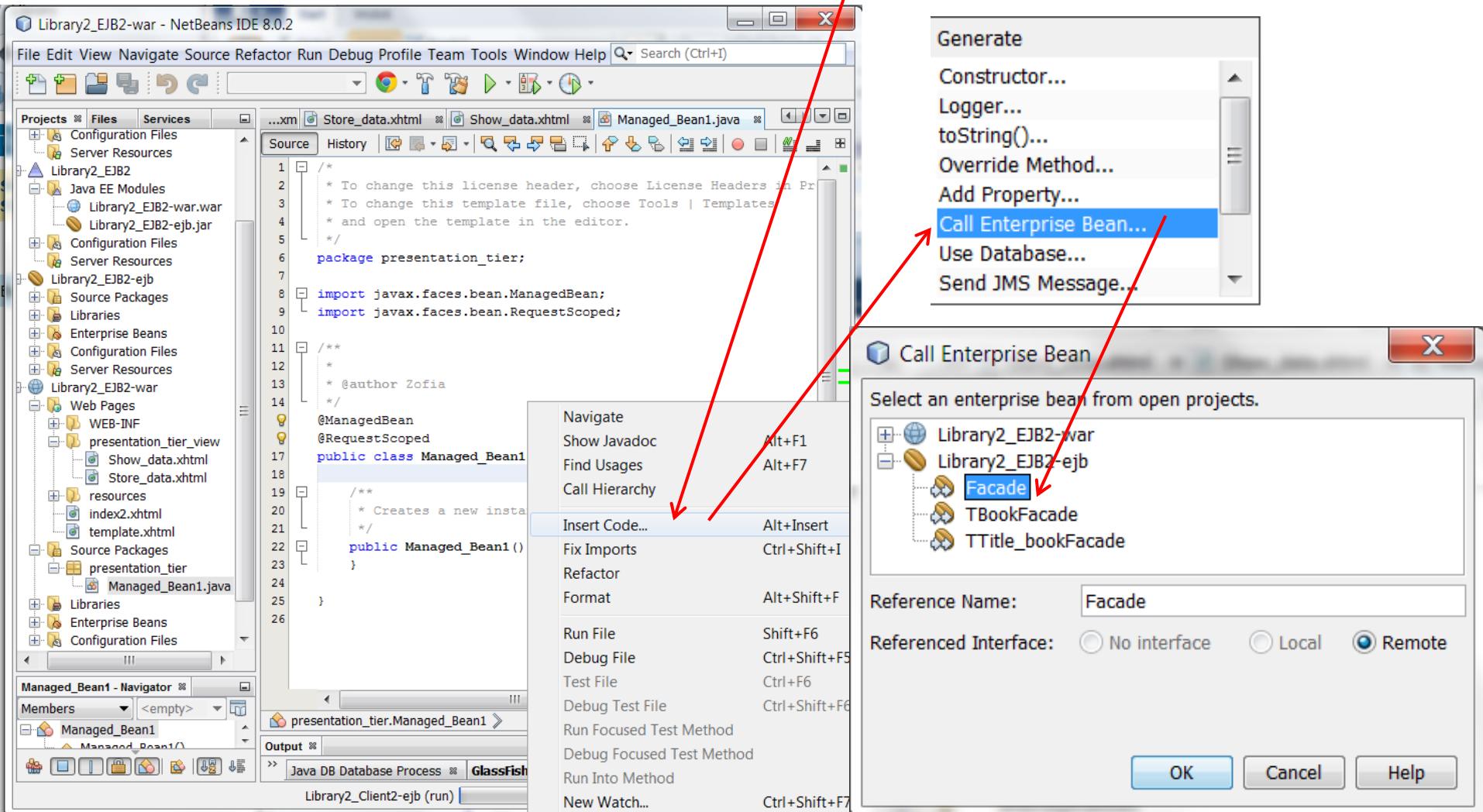
> Java DB Database Process GlassFish Server 4.1 SQL 1 exec

Library2_Client2-ejb (run) running... 1:1 INS





6.6.4. Creation the connection with the Facade remote session bean from the Library2_EJB2-ejb module by using the mechanism of Insert code (right click the code in the editor window, select the the Insert Code... item, and then select the Call Enterprise Bean, and at last select the Facade component from the list in the Call Enterprise dialog). Click OK..





6.6.5. The code of the Managed_Bean1 class with added reference of the Facade EJB by using the annotation and injecting mechanism.

The screenshot shows the NetBeans IDE interface with the title bar "Library2_EJB2-war - NetBeans IDE 8.0.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Tools, Window, Help, and Search (Ctrl+I). The toolbar has icons for file operations like New, Open, Save, and Build. The left pane is the Projects view, showing a Java EE Module named "Library2_EJB2-war" containing "Java EE Modules" (Library2_EJB2-war.war, Library2_EJB2-ejb.jar), "Configuration Files", and "Server Resources". Inside "Library2_EJB2-ejb", there are "Source Packages", "Libraries", "Enterprise Beans", and "Configuration Files". Inside "Library2_EJB2-war", there are "Web Pages" (WEB-INF, presentation tier_view, resources, Source Packages), "Libraries", "Enterprise Beans", and "Configuration Files". The central pane displays the code for "Managed_Bean1.java" under the "Source" tab:

```
1 package presentation_tier;
2
3 import business_tier.FacadeRemote;
4 import javax.ejb.EJB;
5 import javax.faces.bean.ManagedBean;
6 import javax.faces.bean.RequestScoped;
7
8 @ManagedBean
9 @RequestScoped
10 public class Managed_Bean1 {
11     @EJB
12     private FacadeRemote facade;
13
14     public Managed_Bean1() {
15     }
16
17 }
18
```

The code editor has syntax highlighting for Java and annotations. The bottom pane shows the Navigator with "Managed_Bean1" selected and the Output window showing "Java DB Database Process", "GlassFish Server 4.1", and "SQL 1 execute". The status bar at the bottom right shows the time as 14:5 and the mode as INS.



6.6.6. The code of the Managed_Bean1 class

```
package presentation_tier;

import business_tier.FacadeRemote;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;

@ManagedBean
@RequestScoped
public class Managed_Bean1 {
    @EJB
    private FacadeRemote facade;
    private DataModel items;
    public Managed_Bean1() { }

    public FacadeRemote getFacade()
    public void setFacade(FacadeRemote facade)
        { return façade; }
        { this.facade = façade; }

    public String store_data()
        try {
            facade.add_titles();
            facade.add_books();
        } catch (Exception e) { }
        return "/faces/index2";
}
```

The model of the dataTable component used on the Show_data.xhtml JSF page for displaying the data from the database

The **store_data** method for handling event of the h:commandButton component, used by the Store_data.xhtml page. This method calls two methods from the Facade session bean, which inserting data into the database by using ORM (JPA 2.1 controllers from Library2_EJB2-ejb module). As the response, it returns to the main index2.xhtml JSF page (p. 3-7, 24-25)



6.6.7. The code of the Managed_Bean1 class - continuation

```
public String show_data() {  
    create_DataModel();  
    return "/faces/presentation_tier_view>Show_data";  
}  
  
public DataModel create_DataModel() {  
    try{  
        return new ListDataModel( facade.titles());  
    }  
    catch(Exception e)  
    {  
        System.out.println("Blad");  
        return null;  }  
}  
  
public DataModel getItems() {  
    if (items == null) {  
        items = create_DataModel();    }  
    return items;  
}  
  
public void setItems(DataModel items) {  
    this.items = items;  }  
}
```

The **show_data** method for handling event as the displaying the actual data from the database. It is useful for Manageg_Bean1, if it has the Session Scope. As the response, it returns to the Show_data.xhtml JSF page.

The **create_DataModel** method for creation the new data model for h:DataTable component – this based on data returned from the titles method of the Facade session bean and they are read from database by using ORM mechanism (p. 24-25, 3-7)

The **getItems** method for creation the new data model for h:DataTable component by using binding mechanism – this based on data returned from the create_DataModel (at the previous slide).



6.6.9. The code of the Managed_Bean1 class – the result

The screenshot shows the NetBeans IDE interface with the title bar "Library2_EJB2-war - NetBeans IDE 8.0.2". The left sidebar displays the project structure for "Library2_EJB2-war" under "Java EE Modules". The "Managed_Bean1.java" file is open in the main editor window, showing Java code for a managed bean. The code includes imports for presentation tier classes and annotations like @ManagedBean and @RequestScoped. The bottom navigation bar shows tabs for "Java DB Database Process", "GlassFish Server 4.1", and "SQL 1 execute".

```
package presentation_tier;

import business_tier.FacadeRemote;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;

@ManagedBean
@RequestScoped
public class Managed_Bean1 {
    @EJB
    private FacadeRemote facade;
    private DataModel items;

    public Managed_Bean1() {
    }

    public FacadeRemote getFacade() {
        return facade;
    }

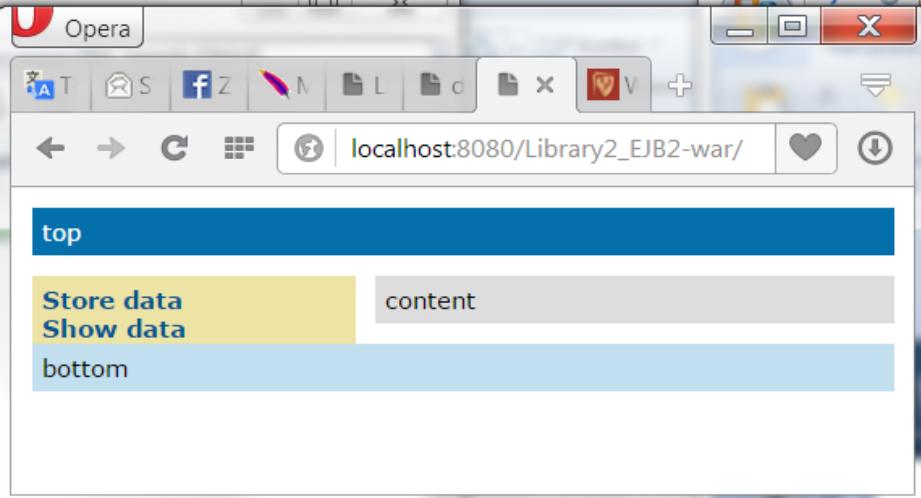
    public void setFacade(FacadeRemote facade) {
        this.facade = facade;
    }
}
```



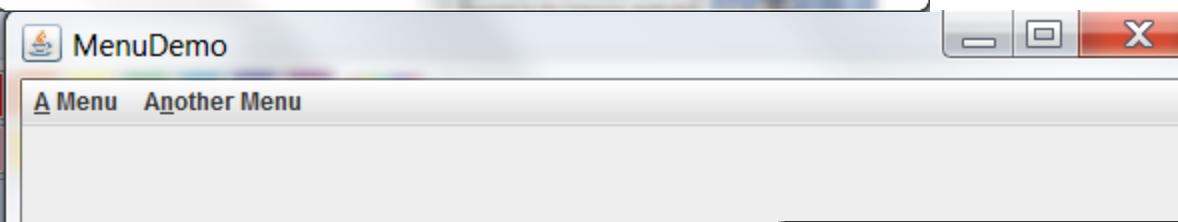
7.0. Running the program

1. Connect with the Library2015 database (5 slide), if you start first one.
2. If you start after introduction of some changes in your programs, you must undeploy all deployed programs: Library2_EJB2 , Library2_Client2-ejb and Library2_EJB2-war, accordingly to the 81 slide. After this development, your program will be executed properly, if you clean and build the following programs (necessary, if you make some changes or start first time):
 1. Library2_JPA2
 2. Library2_interface2-ejb
 3. Library2_EJB2-ejb
 4. Library2_Client2-ejb
 5. Library2_EJB2-war
3. Then you must **deploy** the Library2_EJB2 program if start first one.
4. Finally, you may **run** a few instances of Library2_Client2-ejb programs.
5. At last, insert the following url address: http://localhost:8080/Library2_EJB2-war/ in any browser and run a few web clients .
6. These programs (p.3 i 4) share the common data as titles and books.
7. In the **Service Tab** you may see, if your EE program deploy properly (Server item). The other useful information you may get from the **GlassFish output window tab**.
8. If you stop your above programs, you must undeploy them, accordingly to the 81 slide.





7.1. The view of the main web page (rendered by using the `index2.xhtml` JSF page) of the **Library2_EJB2-war project**



7.2. The view of the Enterprise Application Client (**Library2_Client2-ejb**) for processing of application data - with the same responsibilities as of the version of third laboratory.

The screenshot shows a window titled "MenuDemo" with a menu bar containing "A Menu" and "Another Menu". The main content area contains a form for adding book data:

Title	<input type="text" value="1"/>
Author	<input type="text" value="1"/>
ISBN	<input type="text" value="1"/>
Publisher	<input type="text" value="1"/>
Actor	<input type="text"/>

At the bottom left is a blue "Add title" button.

The screenshot shows a window titled "MenuDemo" with a menu bar containing "A Menu" and "Another Menu". The main content area contains a detailed form for adding book data:

Title	<input type="text" value="1"/>
Author	<input type="text" value="1"/>
ISBN	<input type="text" value="1"/>
Publisher	<input type="text" value="1"/>
Actor	<input type="text" value="1"/>

Below the form is a blue "Add title" button.



7.3. The view after choose the **Store data** link (the left part of page) of
the **Library2_EJB2-war** client

The screenshot shows a Opera browser window with the URL `localhost:8080/Library2_EJB2-war/faces/presentation_tier_view/Store_data.xhtml`. The page has a blue header bar with the word 'Top'. Below it is a yellow button labeled 'Store data' and 'Show data'. To the right of this button is a grey button labeled 'Store data'. The rest of the page is a light blue area labeled 'Bottom'.

7.4. The view after choosing the **Store data** button (the right part of page) –
i.e. after inserting the application data
into the database (p.7.2) by the
Library2_EJB2-war client

The screenshot shows a Opera browser window with the URL `localhost:8080/Library2_EJB2-war/faces/presentation_tier_view/Store_data.xhtml`. The page has a blue header bar with the word 'top'. Below it is a yellow button labeled 'Store data' and 'Show data'. To the right of this button is a grey button labeled 'content'. The rest of the page is a light blue area labeled 'bottom'.

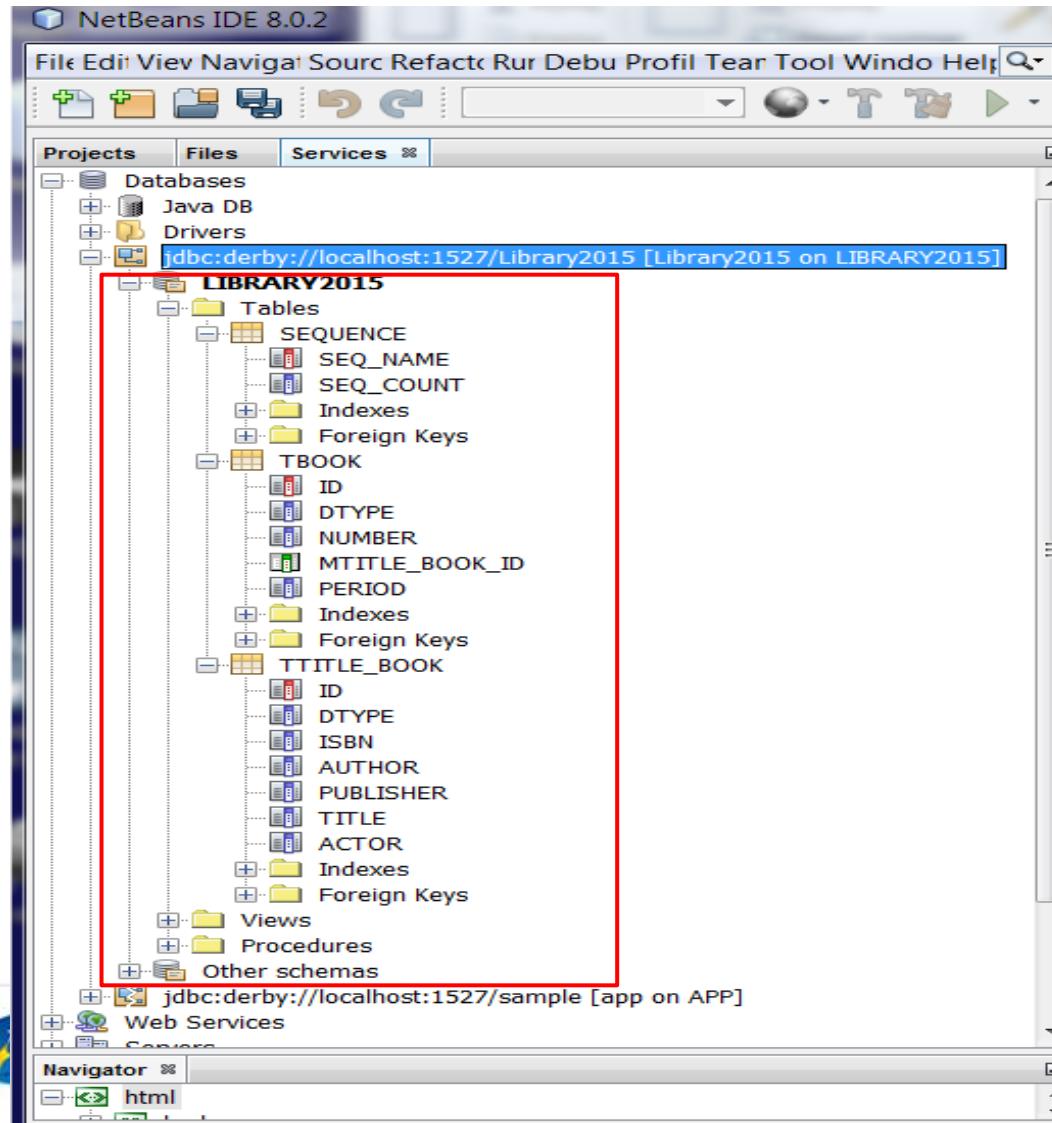
7.5. The view after choosing
the **Show data** button (the
left part of page) – i.e. after
getting data from the
database by the
Library2_EJB2-war client

The screenshot shows a Opera browser window with the URL `localhost:8080/Library2_EJB2-war/faces/presentation_tier_view>Show_data.xhtml`. The page has a blue header bar with the word 'Top'. Below it is a yellow button labeled 'Store data' and 'Show data'. The rest of the page is a light blue area labeled 'Bottom'. In the center, there is a table with the following data:

Publisher	ISBN	Title	Author	Actor
1	1	1	1	
1	1	1	1	1



7.6. The generated tables by using the ORM mechanism, accordingly to the annotation placed in the Entity classes (the instruction of the fourth laboratory)





7.7. The data stored in the database by Store data web page- at previous time they have been inserted by the Enterprise Application Client program (**Library2_Client2-ejb**) as the application data (p.7.2)

The screenshot shows the NetBeans IDE 8.0.2 interface. On the left, the Projects panel displays a database connection named 'LIBRARY2015' under 'Databases', which contains tables like 'SEQUENCE', 'TBOOK', and 'TTITLE_BOOK'. The 'Navigator' panel shows files for CSS, HTML, and XHTML. In the center, the Database browser shows two SQL queries. The top query is 'select * from LIBRARY2015.TTITLE_BOOK;'. The results show two rows:

#	ID	DTYPE	ISBN	AUTHOR	PUBLISHER	TITLE	ACTOR
1	1	TTTitle_book	1	1	1	1	<NULL>
2	2	TTTitle_book_on_tape	1	1	1	1	1

The bottom query is 'select * from LIBRARY2015...'. The Output window at the bottom right shows deployment logs for 'Library2_Client2-ejb (run) #2':

```
Copying 2 files to C:\EnglishLecture\Kruczkiewicz\Laboratory2015\lab4\Library2_Client2-ejb\dist\Library2_Client2-ejbClient
Warning: C:\EnglishLecture\Kruczkiewicz\Laboratory2015\lab4\Library2_Client2-ejb\dist\gfdeploy\Library2_Client2-ejb does not exist.
```



MenuDemo

A Menu Another Menu

Publisher	ISBN	Title	Author	Actor
1	1	1	1	
1	1	1	1	1
2	2	2	2	
2	2	2	2	2

Number of a book
4

Period of a book

Add book

Books

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 3 Period: Mon Feb 23 14:18:10 CET 2015

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 3 Period: Mon Feb 23 14:18:10 CET 2015

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 4

Opera

Tłumac SYSTEM Zofia K Maven List the dr inż. Z Face Witryna +

localhost:8080/Library2_EJB2-war/faces/presentation_tier_view>Show_data.xhtml#

Top

Store data Show data

Publisher	ISBN	Title	Author	Actor
1	1	1	1	
1	1	1	1	1
2	2	2	2	
2	2	2	2	2

Bottom

7.8. The **Library2_Client2-ejb** form to adding the new books of the selected title, as the application data.



MenuDemo

A Menu Another Menu

Publisher	ISBN	Title	Author	Actor
1	1	1	1	
1	1	1	1	1
2	2	2	2	
2	2	2	2	2

Number of a book
4

Period of a book

Add book

Books

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 3 Period: Mon Feb 23 14:18:10 CET 2015

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 3 Period: Mon Feb 23 14:18:10 CET 2015

Title: 1 Author: 1 ISBN: 1 Publisher: 1 Actor: 1 Number: 4

7.9. Restored data from database, after again opening EE application with two kinds of clients: **Library2_Client2-ejb** as the Enterprise Application client (below) and **Library2_EJB2-war** as the web client

Opera

Tłumac SYSTEM Zofia K Maven List the dr inż. Z Face Witryna +

localhost:8080/Library2_EJB2-war/faces/presentation_tier_view>Show_data.xhtml#

Top

Store data Show data

Publisher	ISBN	Title	Author	Actor
1	1	1	1	
1	1	1	1	1
2	2	2	2	
2	2	2	2	2

Bottom



7.10. The data stored in the database (titles) by Store data web page- at previous time they have been inserted by the Enterprise Application Client program (**Library2_Client2-ejb**) as the application data (p. 7.8)

The screenshot shows the NetBeans IDE interface with the following details:

- Projects:** Shows a connection to "LIBRARY2015" which contains "Tables" (SEQUENCE, TBOOK, TTITLE_BOOK), "Views", and "Procedures".
- Connections:** A connection named "...tm" is selected, showing the SQL query: `select * from LIBRARY2015.TTITLE_BOOK;`
- Results:** The results of the query are displayed in a table:

#	ID	DTYPE	ISBN	AUTHOR	PUBLISHER	TITLE	ACTOR
1	1	TTTitle_book	1	1	1	1	<NULL>
2	2	TTTitle_book_on_tape	1	1	1	1	1
3	51	TTTitle_book	2	2	2	2	<NULL>
4	52	TTTitle_book_on_tape	2	2	2	2	2
- Output:** The output pane shows the execution results:

```
Java DB Database Process  GlassFish Server 4.1  SQL 1 execution  Library2_Client2-ejb (run)  Library2_EJB2
Executed successfully in 0 s.
Line 1, column 1

Execution finished after 0 s, 0 error(s) occurred.
```



7.11. The data stored in the database (books) by Store data web page- at previous time they have been inserted by the Enterprise Application Client program (**Library2_Client2-ejb**) as the application data (p. 7.8)

The screenshot shows the NetBeans IDE interface with the following details:

- Projects:** Shows a database named **LIBRARY2015** under the **Databases** section, containing tables like **SEQUENCE**, **TBOOK**, and **TTITLE_BOOK**.
- SQL Editor:** Displays the query `select * from LIBRARY2015.TBOOK;`. The results pane shows the following data:

#	ID	DTYPE	NUMBER	MTITLE_BOOK_ID	PERIOD
1	3	TBook	1	1	<NULL>
2	54	TBook	4	2	<NULL>
3	53	TBook_period	3	2	2015-02-23

- Output:** Shows the execution results:

```
1 execution | Library2_Client2-ejb (run) | Library2_EJB2-war (run) | SQL 2 execut
Executed successfully in 0,031 s.
Line 1, column 1
Execution finished after 0,031 s, 0 error(s) occurred.
```



7.12. The data stored in the database by the Store data web page - at previous time they have been inserted by the Enterprise Application Client program (**Library2_Client2-ejb**) as the application data (p.7.8) - the view of auxiliary sequence table (to support the AUTO mechanism of generating the keys of persisted data during ORM mechanism)

The screenshot shows the NetBeans IDE 8.0.2 interface. On the left, the Projects panel displays a database connection named 'LIBRARY2015' containing tables like 'SEQUENCE', 'TBOOK', and 'TTITLE_BOOK'. The Navigator panel shows a file structure for 'Library2_EJB2-war' with CSS, HTML, and XHTML files. In the center, the Database browser window is open, showing the SQL query 'select * from LIBRARY2015."SEQUENCE"'; the results pane displays a single row with SEQ_NAME 'SEQ_GEN' and SEQ_COUNT '100'. Below this, the Output window shows the message 'Executed successfully in 0,046 s.' and 'Execution finished after 0,046 s, 0 error(s) occurred'.

#	SEQ_NAME	SEQ_COUNT
1	SEQ_GEN	100

```
select * from LIBRARY2015."SEQUENCE";
#   SEQ_NAME      SEQ_COUNT
1  SEQ_GEN       100
```

```
Executed successfully in 0,046 s.
Line 1, column 1

Execution finished after 0,046 s, 0 error(s) occurred
```



7.13. The closing or before the update of Enterprise application – after undeploy process of EE project components

The screenshot shows the NetBeans IDE 8.0.2 interface. A red arrow points from the 'Undeploy' option in the context menu of the 'Library2_Client2-ejb' application node in the Servers list to the 'Output' window.

Output Window Content:

```
Java DB Database Process  GlassFish Server 4.1  SQL 1 execution  Library2_Client2-ejb (run)  Library2_EJB2

Executed successfully in 0 s.
Line 1, column 1

Execution finished after 0 s, 0 error(s) occurred.
```

Code Editor Window Content:

```
...tm Managed_Bean1.java build-impl.xml SQL 2 [jdbc:derby://localhost:15...]
Connection: jdbc:derby://localhost:1527/Library2015 [Library2015 on LIBRARY2015]
1 select * from LIBRARY2015.TTITLE_BOOK;
2
```

Database Navigator Window Content:

#	ID	DTYPE	ISBN	AUTHOR	PUBLISHER	TITLE	ACTOR
1	1	TTTitle_book	1	1	1	1	<NULL>
2	2	TTTitle_book_on_tape	1	1	1	1	1
3	51	TTTitle_book	2	2	2	2	<NULL>
4	52	TTTitle_book_on_tape	2	2	2	2	2



5. You may add annotation to your own new classes and create the proper controllers – for higher assessment (5.0 or 5.5)

