

## Lab 03

# Thực hiện kết nối CSDL, truy xuất thông tin và hiển thị, cập nhật thông tin

### Mục tiêu

- Biết cách cấu hình JNDI DataSource để kết nối với cơ sở dữ liệu
- Thực hành lấy kết nối CSDL và hiển thị dữ liệu với ứng dụng web

### Phần I Bài tập step by step

---

#### Bài 1.1

- Tạo JNDI resource khai báo JDBC DataSource
- Cấu hình Java web application để truy cập JNDI resource đó

Sử dụng JNDI DataSource cho phép:

- Sử dụng database connection pooling cung cấp bởi container (web container hoặc EJB container)
- Database connection độc lập với ứng dụng web
- Nếu cấu hình JNDI trên web server thì cho phép các ứng dụng khác nhau cùng chia sẻ kết nối tới CSDL

#### Step 1: Tạo CSDL tên ProductDB

Câu lệnh SQL để tạo CSDL như sau:

```
CREATE DATABASE IF NOT EXISTS [ProductDB];
```

```
USE [ProductDB];
```

```
CREATE TABLE [Product] (
```

```
[product_id] int(11) NOT NULL,

[name] varchar(45) NOT NULL,

[quantity] int(11) NOT NULL,

[price] int(11) NOT NULL,

[category_id] int(11) NOT NULL,

PRIMARY KEY ([product_id]),

KEY [category_id_idx] ([category_id]),

CONSTRAINT [fk_category_id] FOREIGN KEY [category_id] REFERENCES Category (category_id) ON
DELETE NO ACTION ON UPDATE NO ACTION);

INSERT INTO [Product] VALUES

(1,'Tivi Sam sung 32in',300,10,1),

(2,'Tivi Toshiba 40in',600,20,1),

(3,'Tu lanh Sharp 300L',500,30,2),

(4,'Dieu hoa Daikin 12000BTU',800,16,8),

(5,'May giat Samsung 12kg',460,10,6),

(6,'Tivi Panasonic 32in',100,10,1);
```

Trong bài thực hành này, chúng ta sử dụng SQL Server để quản lý CSDL. Sinh viên có thể lựa chọn các hệ quản trị CSDL khác ví dụ như MySQL, PostgreSQL v.v.

Câu lệnh trên cho phép tạo CSDL tên ProductDB, 1 table tên Product và insert vào Product một vài bản ghi.

**STEP 2:** Mở Netbeans -> Tạo mới ứng dụng Web: JNDIDataSource -> Glashfish as Server -> Finish

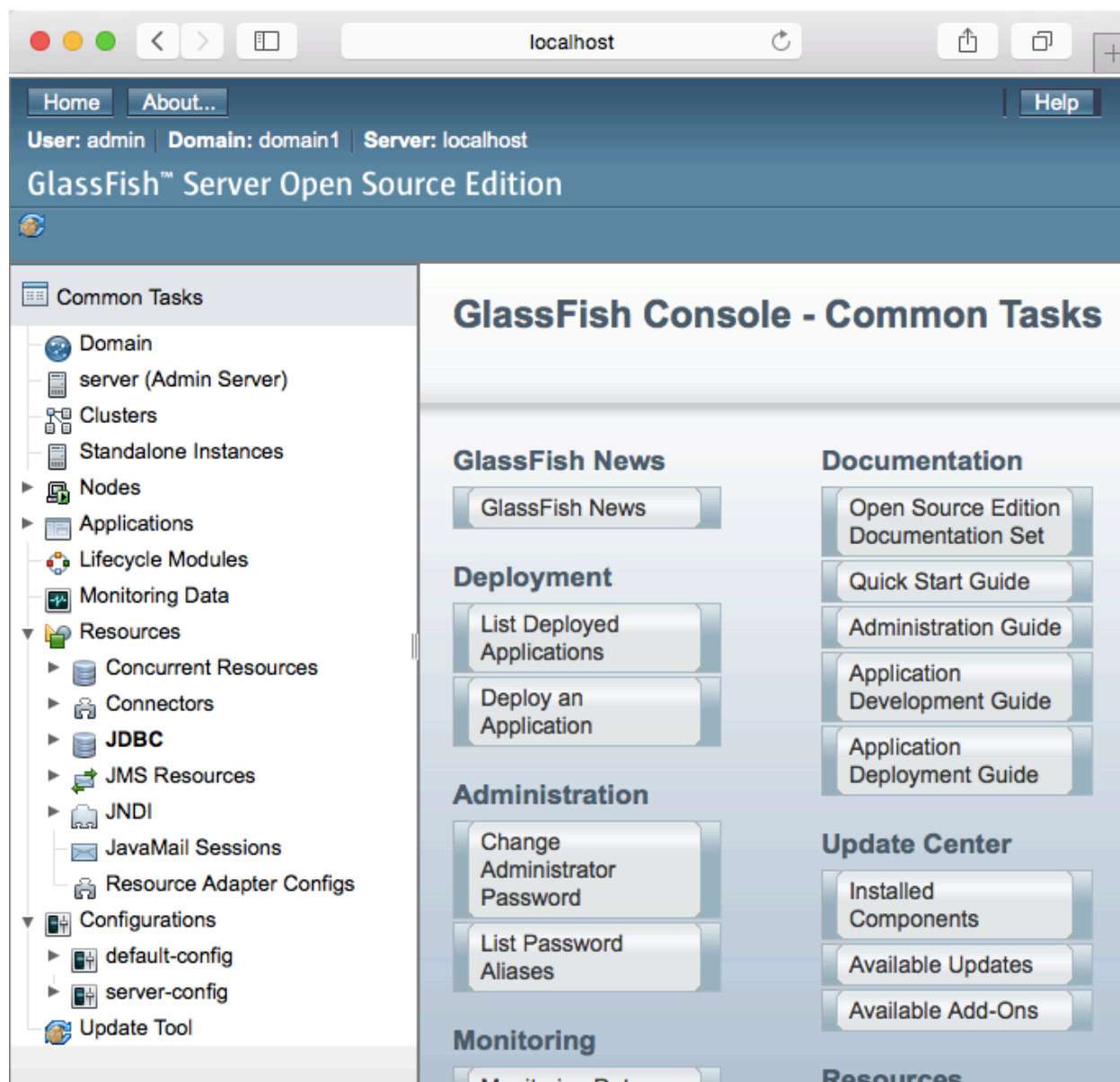
**STEP 3:** Tạo Connection Pool tới CSDL trên Glashfish

Trước hết, phải chắc chắn rằng đã thêm thư viện sqljdbc4.jar vào thư mục cài đặt của glashfish: Netbeans/flashfish-4.1/flashfish/lib

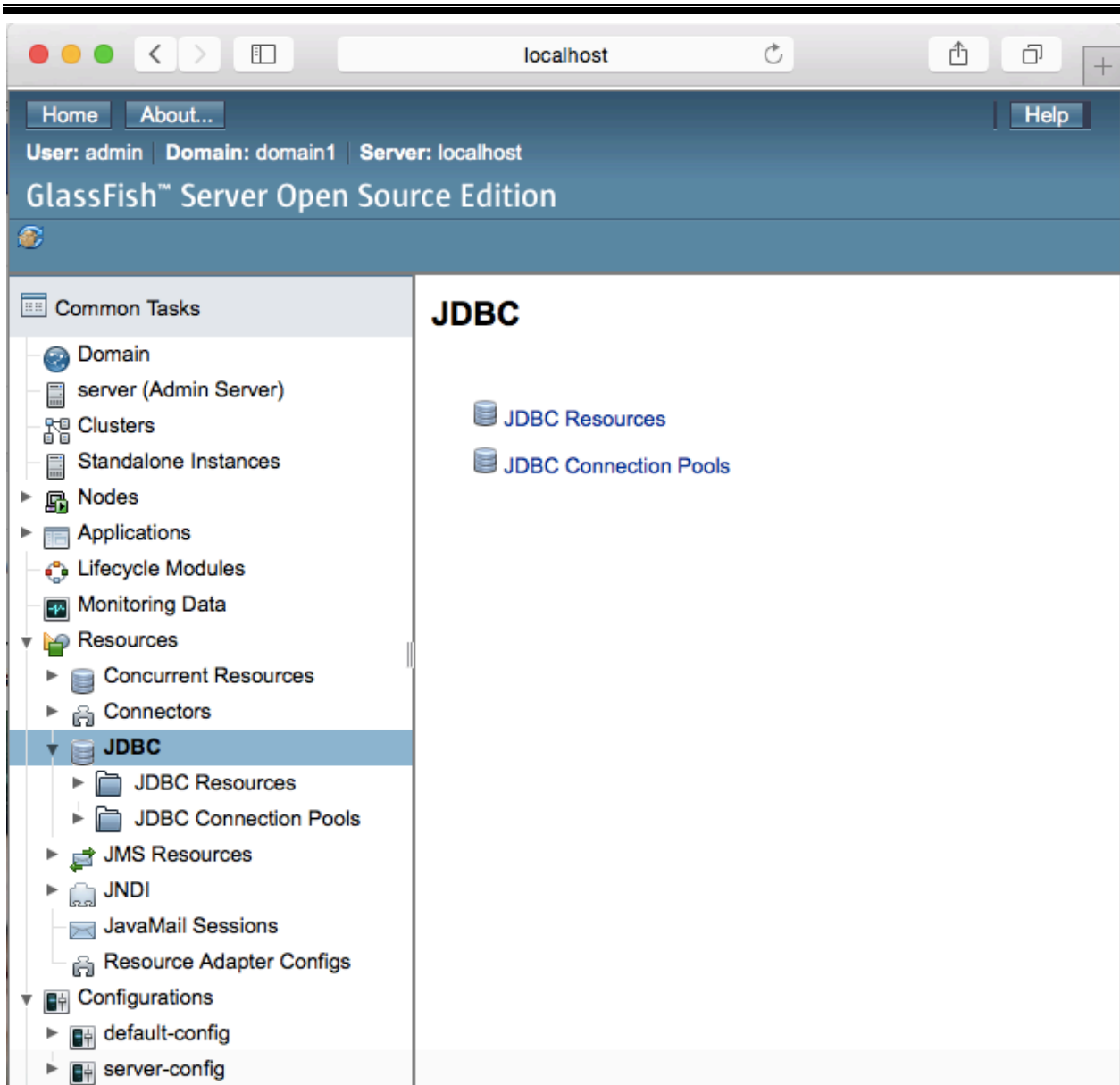
(hoặc C:/glashfish-4.1/glashfish/lib, nếu thư mục cài đặt trên ổ C)

Mở Netbeans -> Services -> Server -> GlashFish 4.1 -> Start

Sau khi GlashFish server khởi động, chuột phải vào GlashFish 4.1 -> View Domain Admin Console.



Resources -> JDBC -> JDBC Connection Pool -> New



The screenshot shows the GlassFish Server Open Source Edition web console. The browser address bar shows 'localhost'. The page title is 'GlassFish™ Server Open Source Edition'. The user is 'admin' on 'domain1' at 'localhost'. The main content area is titled 'New JDBC Connection Pool (Step 1 of 2)' and contains the following fields:

- Pool Name:** \* (required field) [Text input field]
- Resource Type:** [Dropdown menu] Must be specified if the datasource class implements the interface.
- Database Driver Vendor:** [Dropdown menu] Select or enter a database driver vendor
- Introspect:** ☐ **Enabled** If enabled, data source or driver implements enable introspection.

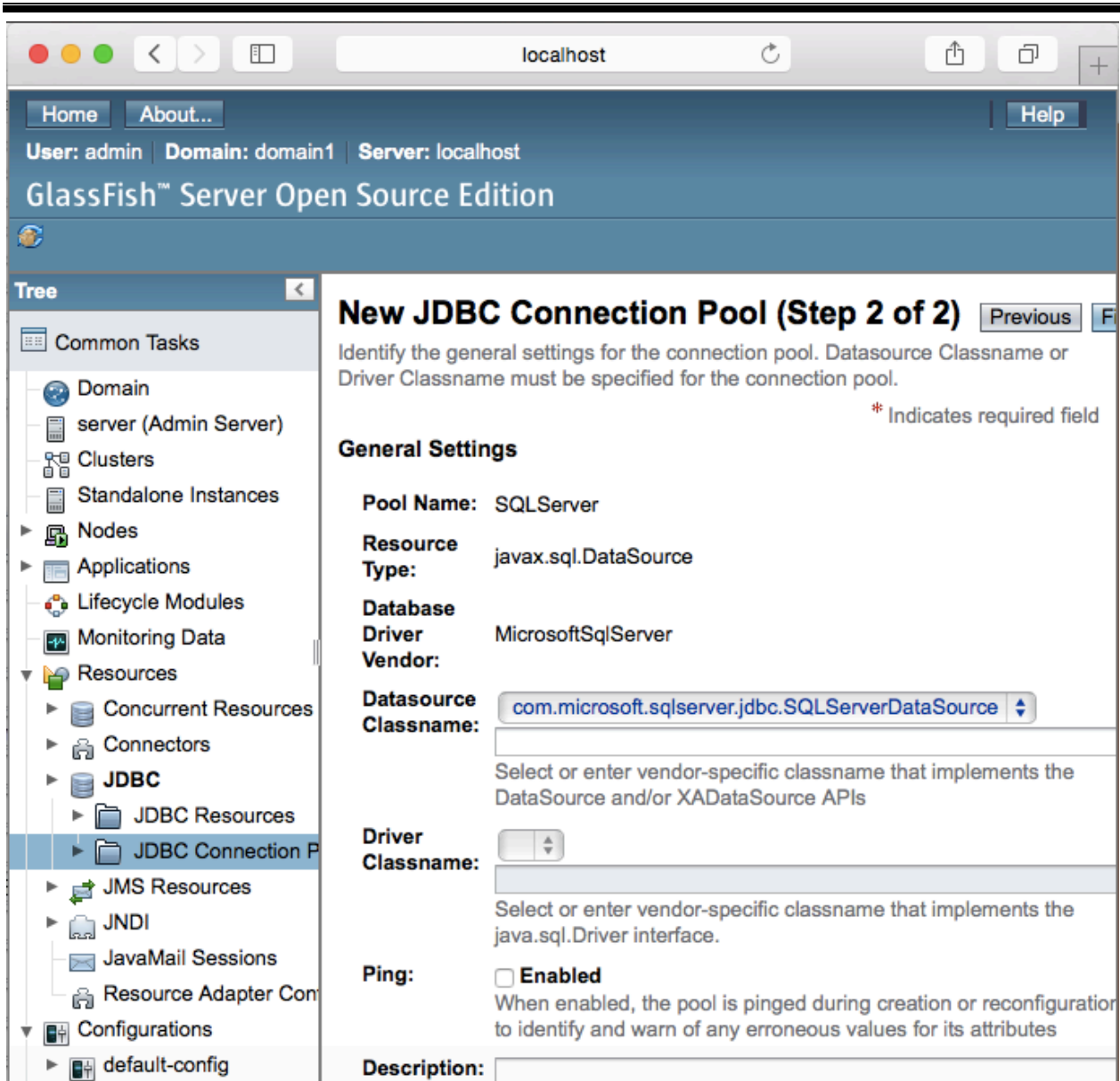
The left sidebar shows a tree view of the server configuration. The 'JDBC Connection Pools' folder is selected.

Pool Name: SQLServer

Resource Type: chọn javax.sql.DataSource

Database Driver Vendor: MicrosoftSqlServer

Click Next



Thêm các thông tin cho User, Password, ServerName, URL, DatabaseName

User: sa -> Thay bằng tên user đã cài đặt (vd: admin)

Password: 123456 -> Thay bằng password đã cài đặt

ServerName: localhost

DatabaseName: ProductDB

URL: jdbc:sqlserver://localhost:1433

Click Finish

Chọn Resources -> JDBC Resources -> Hiện lên 1 list các JDNI DataSource

The screenshot shows the GlassFish Server Open Source Edition administration console. The browser address bar shows 'localhost'. The page header includes 'Home', 'About...', and 'Help' links. Below the header, it displays 'User: admin | Domain: domain1 | Server: localhost'. The main title is 'GlassFish™ Server Open Source Edition'.

On the left is a 'Tree' navigation pane with the following structure:

- Common Tasks
- Domain
  - server (Admin Server)
  - Clusters
  - Standalone Instances
  - Nodes
  - Applications
  - Lifecycle Modules
  - Monitoring Data
  - Resources
    - Concurrent Resources
    - Connectors
    - JDBC
      - JDBC Resources (selected)
      - JDBC Connection Pools
    - JMS Resources
    - JNDI
    - JavaMail Sessions
    - Resource Adapter Configurations
  - Configurations
    - default-config

The main content area is titled 'JDBC Resources' and includes the text: 'JDBC resources provide applications with a means to connect to a database.'

Below this is a table titled 'Resources (4)' with the following columns: Select, JNDI Name, Logical JNDI Name, Enabled, and Connection Pool. The table contains four rows of resources:

Select	JNDI Name	Logical JNDI Name	Enabled	Connection Pool
<input type="checkbox"/>	<a href="#">jdbc/ProductDB</a>		✓	MySQLServer
<input type="checkbox"/>	<a href="#">jdbc/__TimerPool</a>		✓	__TimerPool
<input type="checkbox"/>	<a href="#">jdbc/__default</a>	java:comp/DefaultDataSource	✓	DerbyPool
<input type="checkbox"/>	<a href="#">jdbc/sample</a>		✓	SamplePool

## Chọn New

Home About... Help

User: admin Domain: domain1 Server: localhost

GlassFish™ Server Open Source Edition

**Tree**

- Common Tasks
- Domain
  - server (Admin Server)
  - Clusters
  - Standalone Instances
  - Nodes
  - Applications
  - Lifecycle Modules
  - Monitoring Data
  - Resources
    - Concurrent Resources
    - Connectors
    - JDBC
      - JDBC Resources
      - JDBC Connection Pools
      - JMS Resources
      - JNDI
      - JavaMail Sessions
      - Resource Adapter Configurations
  - Configurations
    - default-config

**New JDBC Resource** OK Cancel

Specify a unique JNDI name that identifies the JDBC resource you want to create. The name must contain only alphanumeric, underscore, dash, or dot characters.

JNDI Name: \*

Pool Name: DerbyPool

Use the [JDBC Connection Pools](#) page to create new pools

Description:

Status: ☒ Enabled

**Additional Properties (0)**

Add Property Delete Properties

Select	Name	Value	Description
No items found.			

JNDI Name: jdbc/ProductDB

Pool Name: SQLServer

Description: SQL Server Connection

Click OK.

Test Connection Pool bằng cách chọn SQLServer trên JDBC Connection Pool -> Ping



General
Advanced
Additional Properties

Ping Succeeded

### Edit JDBC Connection Pool

Modify an existing JDBC connection pool. A JDBC connection pool is a group of reusable connections for a particular database.

Load Defaults
Flush
Ping

#### General Settings

Pool Name:

Resource Type:    
Must be specified if the datasource class implements more than 1 of the interface.

Datasource Classname:    
Vendor-specific classname that implements the DataSource and/or XADataSource APIs

Driver Classname:    
Vendor-specific classname that implements the java.sql.Driver interface.

Ping: ☒ Enabled   
When enabled, the pool is pinged during creation or reconfiguration to identify and warn of any erroneous values for its attributes

Deployment Order:    
Specifies the loading order of the resource at server startup. Lower numbers are loaded first.

Description:

---

#### Pool Settings

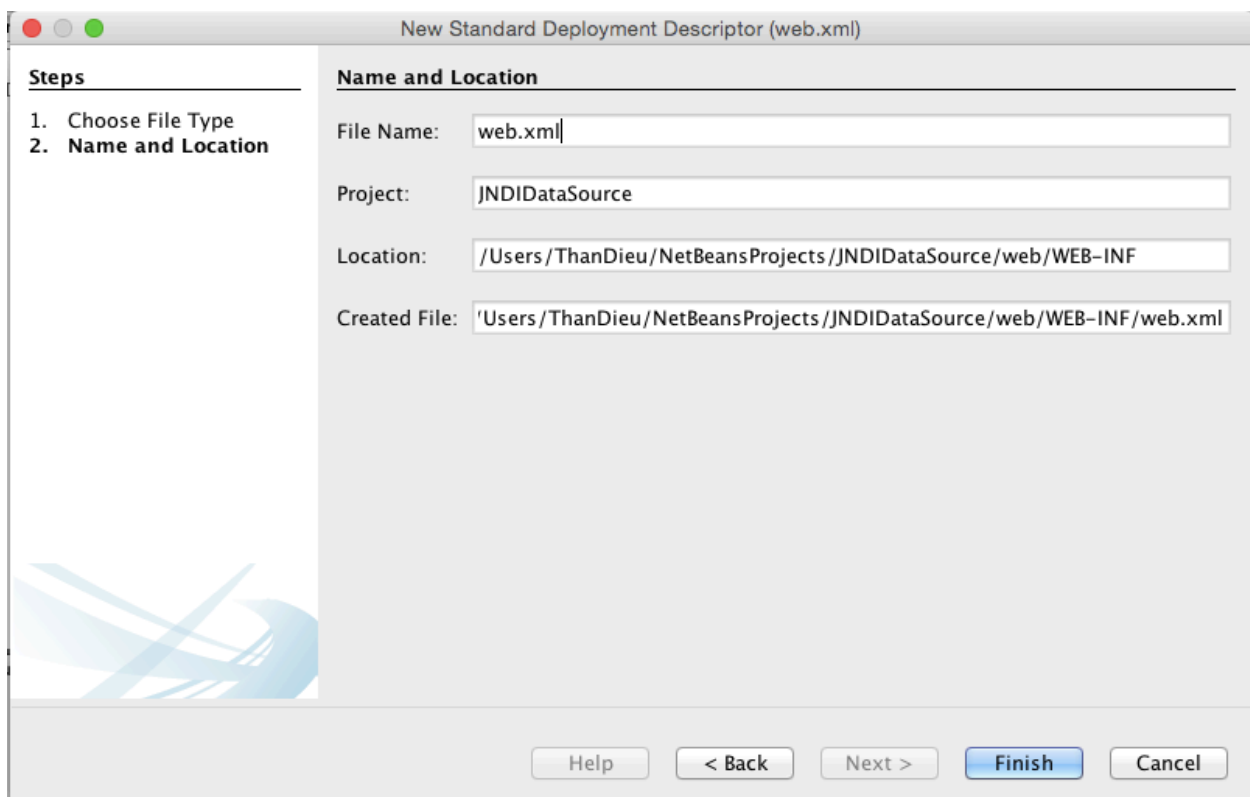
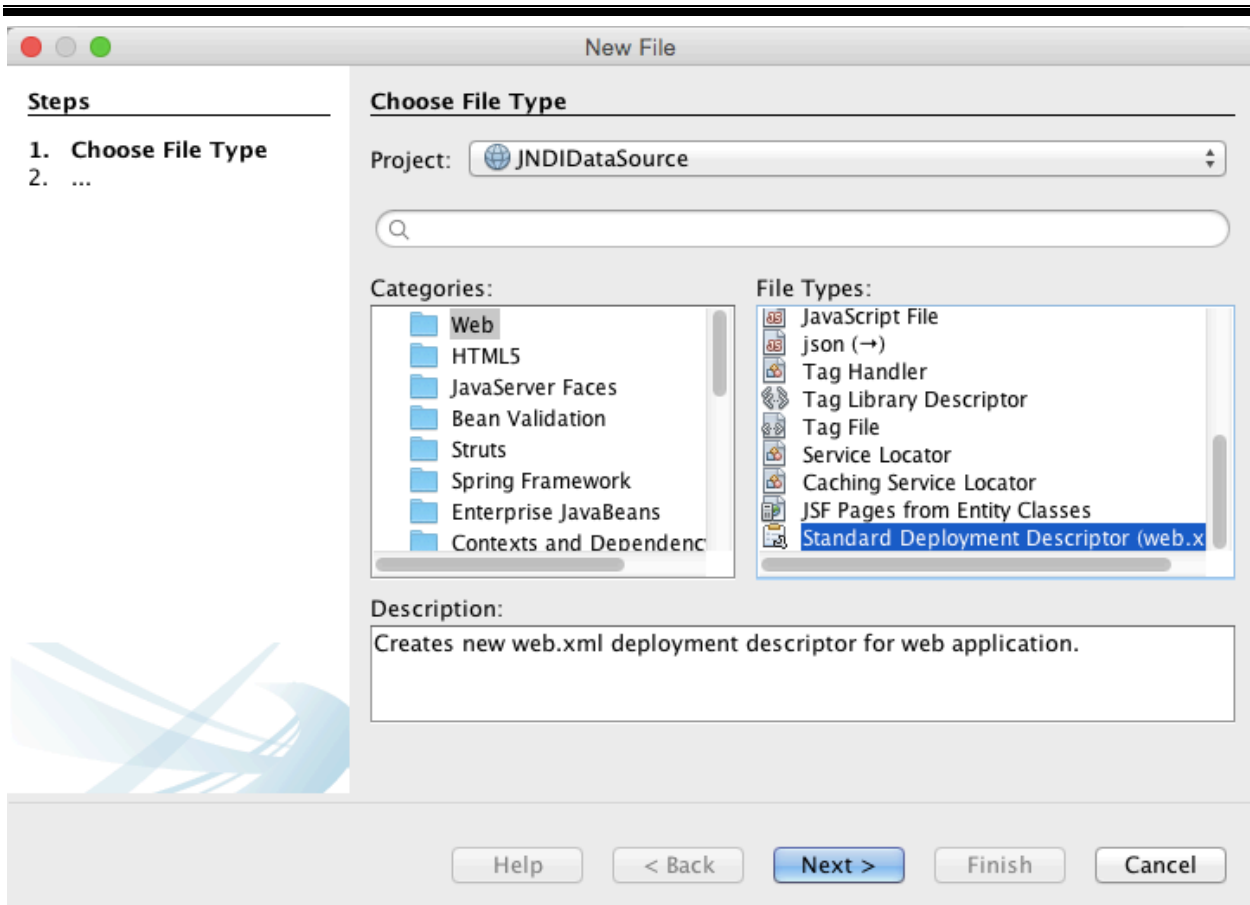
Restart lại GlashFish Server: Từ Netbeans, chọn Services -> Server -> Glashfish 4.1 -> Chuột phải -> Restart

Hết Step 3, chúng ta đã tạo 1 connection pool tới CSDL SQL Server và khai báo 1 JNDI reference tới connection đó. Trong step tiếp theo, chúng ta sẽ khai báo sử dụng JNDI này trong ứng dụng web.

Tương tự, có thể tạo một connection pool tới các hệ quản trị CSDL khác ví dụ MySQL Server. Lưu ý chọn chính xác Database Class Vendor, ví dụ của MySQL là MySQL. Sau đó điền URL (ví dụ của MySQL là: jdbc:mysql://localhost: 3306), user, pass, databaseName.

**STEP 4:** Thêm khai báo resource-reference vào file web.xml của ứng dụng.

Tạo mới file web.xml nếu chưa có, Project->New Others->Standard Deployment Descriptor



Thêm thông tin về resource sẽ sử dụng trong ứng dụng dùng thẻ `<resource-ref></resource-ref>`

```
<resource-ref>

    <description>DB Connection</description>

    <res-ref-name>jdbc/ProductDB</res-ref-name>

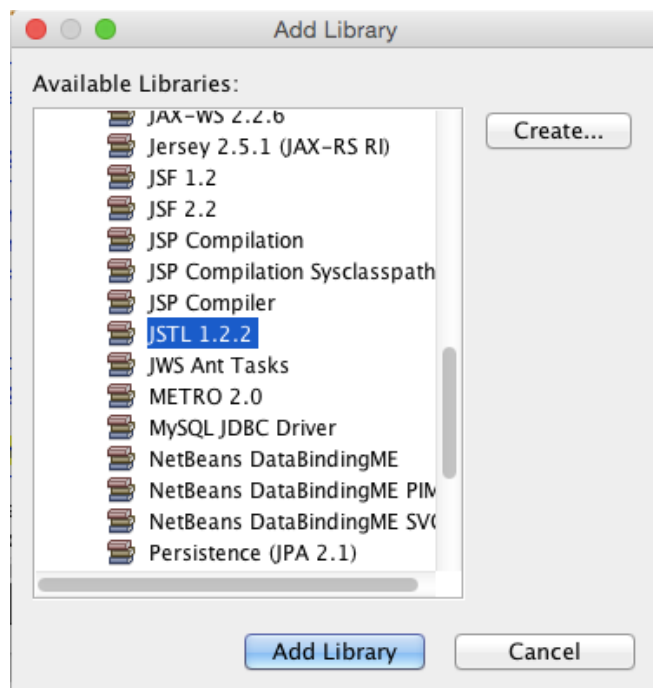
    <res-type>javax.sql.ConnectionPoolDataSource</res-type>

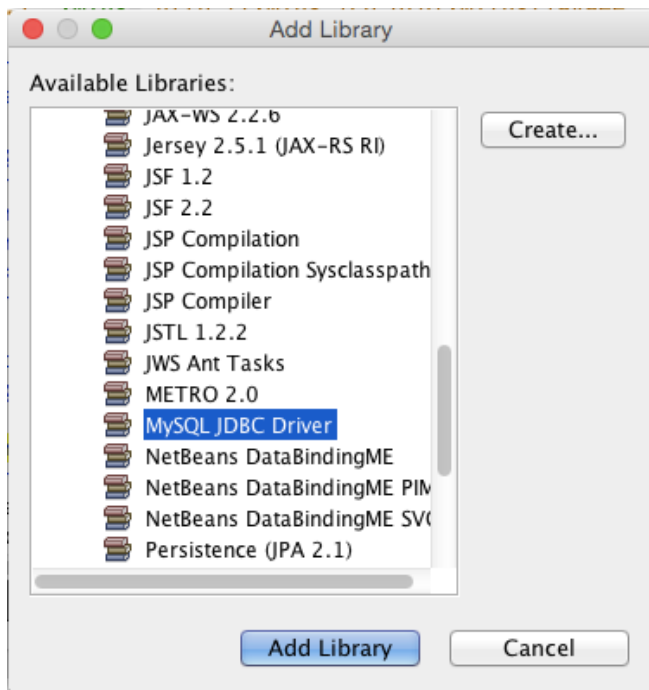
    <res-auth>Container</res-auth>

</resource-ref>
```

**STEP 5:** Sử dụng JNDI đã khai báo để tìm resource, kết nối tới CSDL và lấy dữ liệu về.

Thêm các thư viện liên quan vào project: jstl và jdbc-mysql, nếu dùng SQL Server thì add đường dẫn đến nơi chứa sqljdbc4.jar (download về máy nếu không có sẵn)





### 5.1. Có thể lấy dữ liệu trực tiếp bằng JSTL SQL với thẻ `query`

```
<sql:query var="listProducts" dataSource="jdbc/ProductDB">
    select name, quantity from Product;
</sql:query>
```

Thông tin khai báo `dataSource` chính là tên của JNDI resource.

Cần chú ý tên của JNDI resource tương ứng với CSDL đã config trên GlassFish.

Kết quả của câu lệnh SQL sẽ được trả về trong biến `listProducts`.

File `index.jsp` để hiển thị dữ liệu lấy về từ CSDL như sau:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<sql:query var="listProducts" dataSource="jdbc/ProductDB">
    select name, quantity from Product;
</sql:query>
```

```
<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Users List</title>

</head>

<body>

    <div align="center">

        <table border="1" cellpadding="5">

            <caption><h2>List of products</h2></caption>

            <tr>

                <th>Name</th>

                <th>Quantity</th>

            </tr>

            <c:forEach var="p" items="${listProducts.rows}">

                <tr>

                    <td><c:out value="${p.name}" /></td>

                    <td><c:out value="${p.quantity}" /></td>

                </tr>

            </c:forEach>

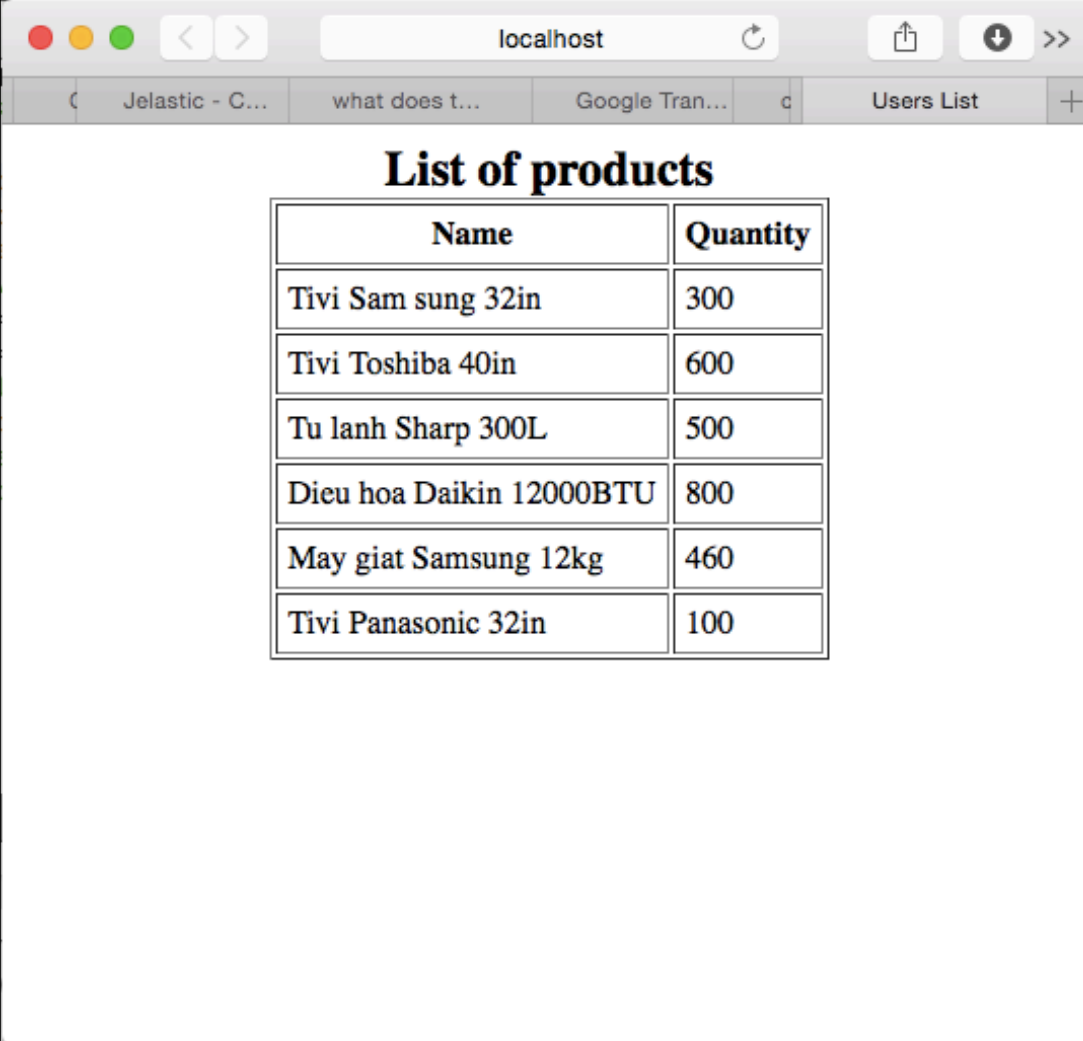
        </table>

    </div>

</body>

</html>
```

Build và Run project, kết quả như sau:

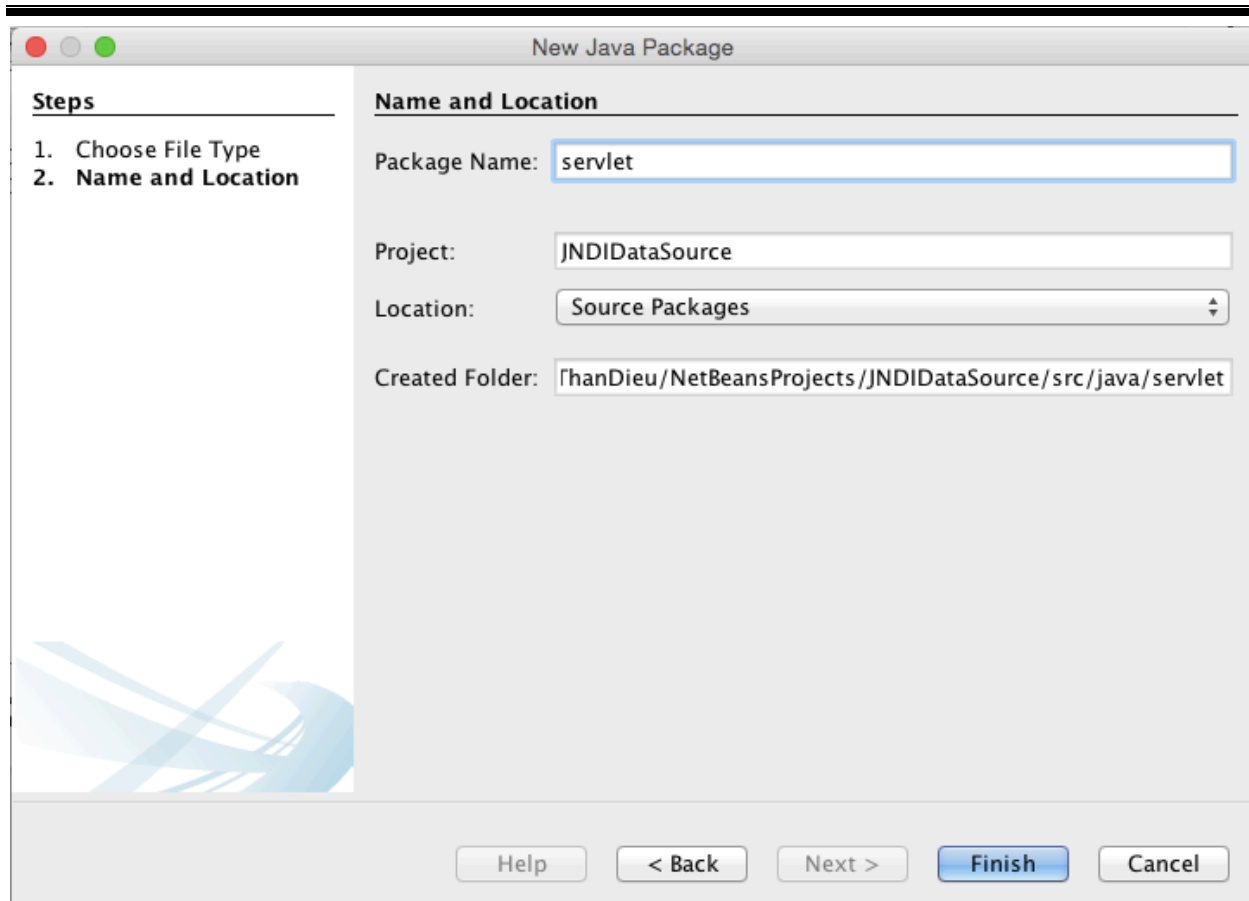


The screenshot shows a web browser window with the address bar set to 'localhost'. The browser has several tabs open, including 'Jelastic - C...', 'what does t...', 'Google Tran...', and 'Users List'. The main content area displays a table titled 'List of products'.

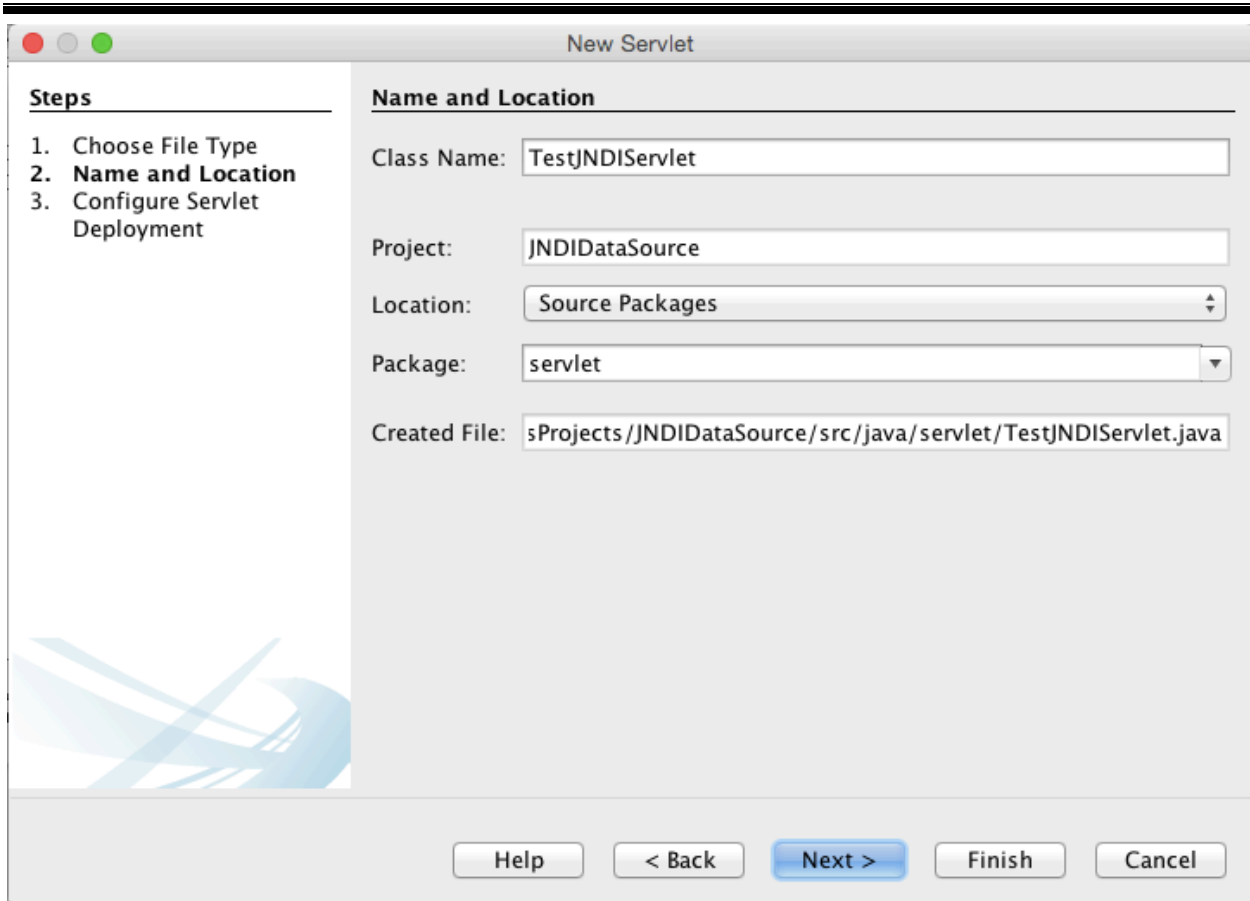
Name	Quantity
Tivi Sam sung 32in	300
Tivi Toshiba 40in	600
Tu lanh Sharp 300L	500
Dieu hoa Daikin 12000BTU	800
May giat Samsung 12kg	460
Tivi Panasonic 32in	100

**5.2.** Chúng ta cũng có thể sử dụng JNDI Resource trong các class Java. Tạo một Servlet, kết nối tới CSDL, hiển thị kết quả trả về của câu lệnh SQL.

Project->New Package: servlet



New Servlet: TestJNDIServlet.java



Trong file servlet này, việc tìm kiếm JNDI Resource có thể thực hiện theo 2 cách:

#### Cách 1: Lookup trực tiếp JNDI resource

```
Context initContext = new InitialContext();
Context envContext = (Context) initContext.lookup("java:comp/env");
DataSource ds = (DataSource) envContext.lookup("jdbc/ProductDB");
Connection conn = ds.getConnection();
```

#### Cách 2: Dùng resource injection (annotation @Resource)

```
@Resource(name = "jdbc/ProductDB")
private DataSource ds;
```

Sau đó trong thân method, lấy kết nối CSDL:

```
Connection conn = ds.getConnection();
```

File code TestJNDIServlet.java như sau:



## Cách 1:

```
package servlet;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.naming.Context;

import javax.naming.InitialContext;

import javax.naming.NamingException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.sql.DataSource;

@WebServlet("/listProducts")

public class TestJNDIServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,

        HttpServletResponse response) throws ServletException,

        IOException {
```

```

        PrintWriter writer = response.getWriter();

        try {

            Context initContext = new InitialContext();

            Context envContext = (Context)
initContext.lookup("java:comp/env");

            DataSource ds = (DataSource)
envContext.lookup("jdbc/ProductDB");

            Connection conn = ds.getConnection();

            Statement statement = conn.createStatement();

            String sql = "select name, quantity from Product";

            ResultSet rs = statement.executeQuery(sql);

            int count = 1;

            while (rs.next()) {

                writer.println(String.format("Product #%d: %-15s %s",
count++,

                                rs.getString("name"),
rs.getString("quantity")));

            }

            } catch (SQLException ex) {

                System.err.println(ex);

            } catch (NamingException ex) {

            }

            Logger.getLogger(TestJNDIServlet.class.getName()).log(Level.SEVERE,
null, ex);

        }

    }
}

```

Cách 2:

```
package servlet;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.sql.DataSource;

import javax.annotation.Resource;

@WebServlet("/listProducts")

public class TestJNDIServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    @Resource(name = "jdbc/ProductDB")

    private DataSource dataSource;

    protected void doGet(HttpServletRequest request,

        HttpServletResponse response) throws ServletException,

        IOException {

        PrintWriter writer = response.getWriter();

        try {

            Connection conn = dataSource.getConnection();

            Statement statement = conn.createStatement();
```

```
String sql = "select name, quantity from Product";

ResultSet rs = statement.executeQuery(sql);

int count = 1;

while (rs.next()) {

    writer.println(String.format("Product #%d: %-15s %s",
count++,

                                rs.getString("name"),
rs.getString("quantity")));

}

} catch (SQLException ex) {

    System.err.println(ex);

}

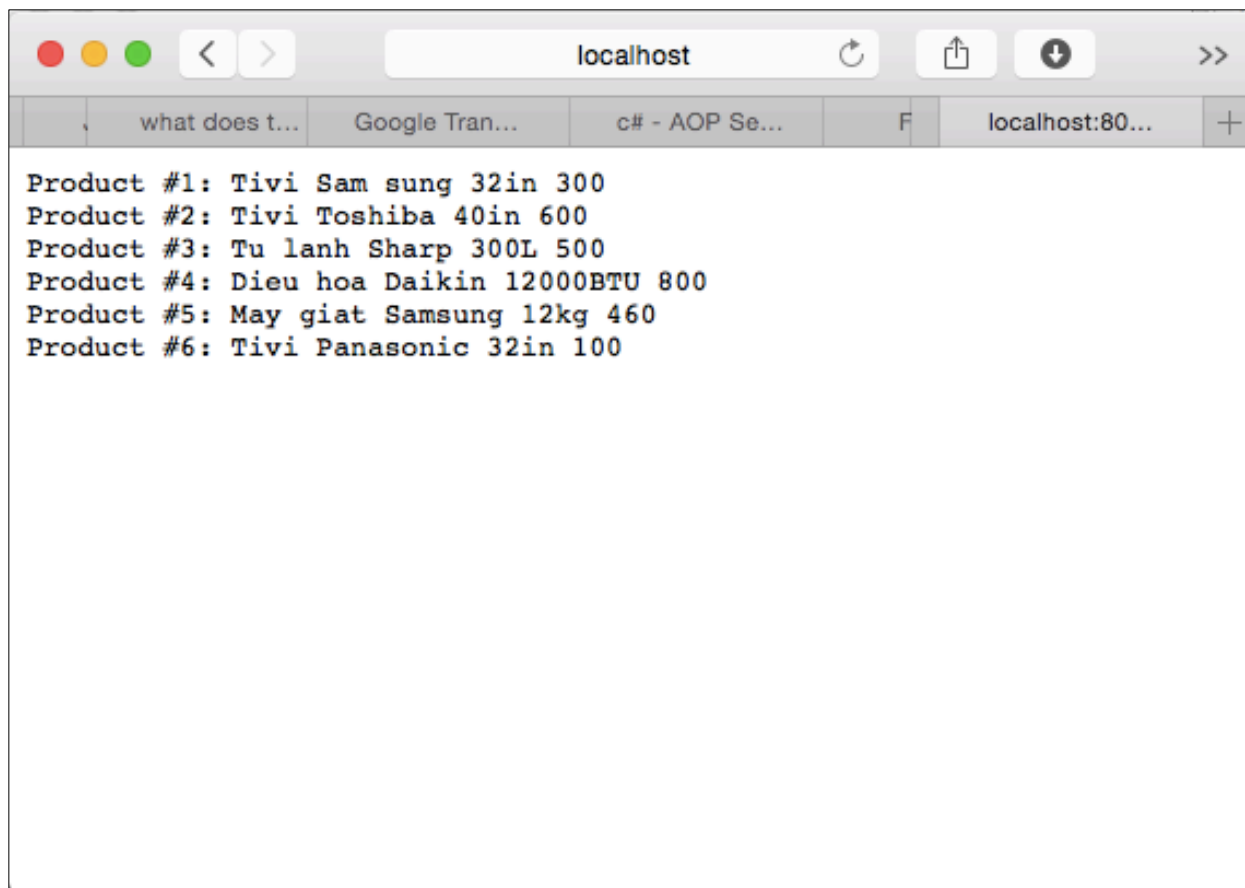
}

}
```

Build và Run project, gọi tới Servlet bằng url:

<http://localhost:8080/JNDIDataSource/listProducts>

Kết quả như sau:



## PHẦN 2: BÀI TẬP THỰC HÀNH

**Lập mới CSDL (new schema) trên MySQL Server hoặc trên SQLServer có tên là ClassManagement. Thêm bảng Student có các trường sau: StudentID (primary key), FirstName, LastName, Address, Telephone, Email, Password. Thêm một số dữ liệu cho bảng.**

**Tạo mới JNDI Data Source trên GlassFish cho CSDL ClassManagement.**

**Tạo connect tới ClassManagement thông qua lookup JNDI. Hiển thị thông tin sinh viên có trong bảng trên JSP.**

# HẾT