Assignment 3

One of the popular courier companies started selling gift items and flowers. They want to encourage online shopping, and so they are providing discount offers to customers. Customers get a discount of 10% on shopping for more than 5000 rupees. Company management has decided to implement an application to provide discount of gift items and also to display details of the available gift items.

You as a member of the development team need to perform the following:

* Create a Spring-based web application with two pages. The first page will prompt users to enter their name and the second page will show a greeting.
* Create a function for ordering Gift Items. The gift item includes: Toys, Wallet, Key chains etc.
* Customers should get an offer of discount as mentioned above.
* Create a function named “FindGift” to display all the available list of gifts on the index page.
* Link of the listed items should be available to the Gift Search Page. The Gift Search page should display details like- Gift ID, Gift Name, Price, Colors etc on tabular form, on the basis of Gift ID.

**Không sử dụng cơ sở dữ liệu**

**com.controllers**

**GiftController.java**

package com.controllers;

import com.models.Gifts;

import com.models.Orders;

import com.models.Users;

import java.util.ArrayList;

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import java.util.List;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

@RequestMapping("/")

public class GiftController {

private static List<Gifts> giftItems = new ArrayList<>();

{

giftItems.add(new Gifts(1, "Stuffed Animal", 7000.00, "Red", "Toy"));

giftItems.add(new Gifts(2, "Leather Wallet", 3500.00, "Blue", "Wallet"));

giftItems.add(new Gifts(3, "Metal Keychain", 1000.00, "Yellow", "Key chain"));

giftItems.add(new Gifts(4, "Board Game", 4500.00, "White", "Toy"));

giftItems.add(new Gifts(5, "Scarameow Keychain", 1500.00, "Blue", "Key chain"));

}

public Gifts getGiftById(long id) {

for (Gifts gift : giftItems) {

if (gift.getGiftID() == id) {

return gift;

}

}

return null;

}

@ModelAttribute("gifts")

public Gifts setGift() {

return new Gifts();

}

@RequestMapping(value = "index", method = RequestMethod.GET)

public String redirectGreetingPage(ModelMap model) {

model.addAttribute("userName", new Users());

model.addAttribute("gifts", giftItems);

return "index";

}

@RequestMapping(value = "greeting", method = RequestMethod.POST)

public String greeting(@ModelAttribute("userName") Users user, ModelMap model) {

String name = user.getName();

model.addAttribute("name", name);

return "greeting";

}

@RequestMapping(value = "gift/{giftID}", method = RequestMethod.GET)

public String displayGiftDetails(@PathVariable("giftID") long giftID, ModelMap model) {

model.addAttribute("gift", getGiftById(giftID));

return "giftDetails";

}

@RequestMapping(value = "orders", method = RequestMethod.POST)

public String orderGifts(@RequestParam(required = false) List<Long> giftIds, ModelMap model) {

if (giftIds == null || giftIds.isEmpty()) {

model.addAttribute("checkGiftIdEmpty", true);

model.addAttribute("initName", new Users());

model.addAttribute("gifts", giftItems);

return "index";

}

double totalPrice = 0.0;

final double MINPRICEDISCOUNT = 5000.00;

final double PRICEDISCOUNT = 0.9; // Apply 10% discount

int quantity = 0;

List<Orders> orderList = new ArrayList<>();

for (long giftId : giftIds) {

for (Gifts giftItem : giftItems) {

if (giftItem.getGiftID() == giftId ) {

quantity += 1;

double discount = giftItem.getPrice() >= MINPRICEDISCOUNT ? PRICEDISCOUNT : 0.0;

totalPrice = calculateDiscountPrice(giftItem.getPrice(), MINPRICEDISCOUNT, PRICEDISCOUNT, model);

Orders order = new Orders(giftItem.getGiftID(), giftItem.getGiftName(), quantity, discount, giftItem.getPrice(), totalPrice, totalPrice, giftItem.getColors(), giftItem.getCategory());

orderList.add(order);

break;

}

}

}

model.addAttribute("orderedGifts", orderList);

model.addAttribute("totalPrice", totalPrice);

return "orders";

}

private double calculateDiscountPrice(double amount, double discountPrice, double discount, ModelMap model) {

if (amount >= discountPrice) {

model.addAttribute("discountApplied", true);

return amount \* discount;

}

return amount;

}

}

**com.models**

**Gifts.java**

package com.models;

public class Gifts {

private long giftID;

private String giftName;

private double price;

private String colors;

private String category;

/\*Getter and setter - constructor\*/

}

**Orders.java**package com.models;

public class Orders {

private Long giftOrderID;

private String giftOrderName;

private int giftOrderQuantity;

private double giftOrderDiscount;

private double giftOrderPrice;

private double discountedPrice;

private double totalPrice;

private String giftOrderColors;

private String giftOrderCategory;

/\*Getter and setter - constructor\*/

}

**Users.java**

package com.models;

public class Users {

private String name;

public String getName() {

return name != null && !name.isEmpty() ? name : "User";

}

public void setName(String name) {

this.name = name;

}

}

**Có sử dụng cơ sở dữ liệu**

**SQL**

USE master

GO

CREATE DATABASE A21131\_Assigment3

GO

USE A21131\_Assigment3

GO

CREATE TABLE Gifts(

GiftID int PRIMARY KEY IDENTITY(1,1),

GiftName nvarchar(255),

Price money,

Colors nvarchar(100),

Category nvarchar(100),

)

INSERT INTO Gifts(GiftName,Price,Colors,Category) VALUES ('Stuffed Animal', 7000.00, 'Red', 'Toy');

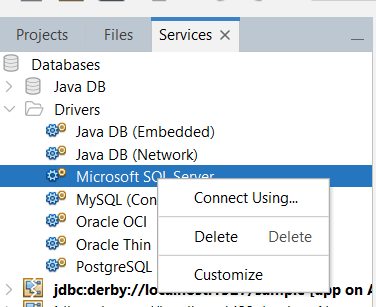
INSERT INTO Gifts(GiftName,Price,Colors,Category) VALUES ('Leather Wallet', 3500.00, 'Blue', 'Wallet');

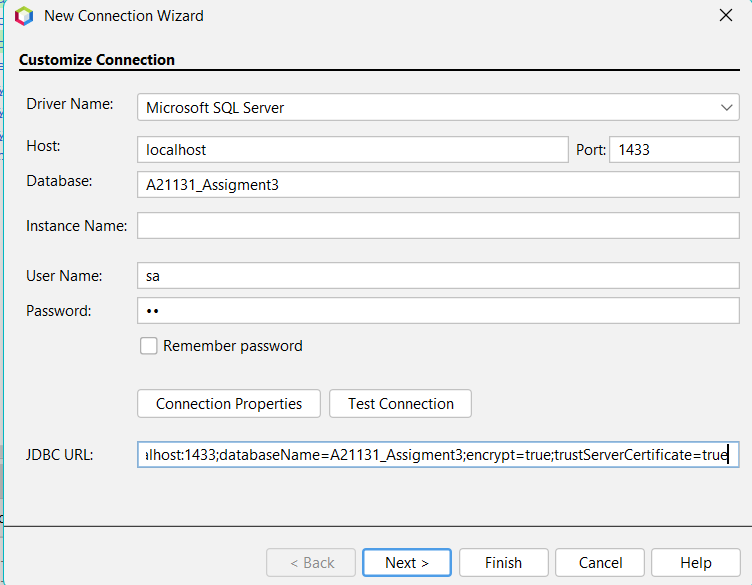
INSERT INTO Gifts(GiftName,Price,Colors,Category) VALUES ('Metal Keychain', 1000.00, 'Yellow', 'Key chain');

INSERT INTO Gifts(GiftName,Price,Colors,Category) VALUES ('Board Game', 4500.00, 'White', 'Toy');

INSERT INTO Gifts(GiftName,Price,Colors,Category) VALUES ('Scarameow Keychain', 1500.00, 'Blue', 'Key chain');

**Kết nối database trên netbeans sau khi tạo database trong sql**

Chọn Services -> database -> Drives -> Micrsoft SQL Sever -> connect using -> 

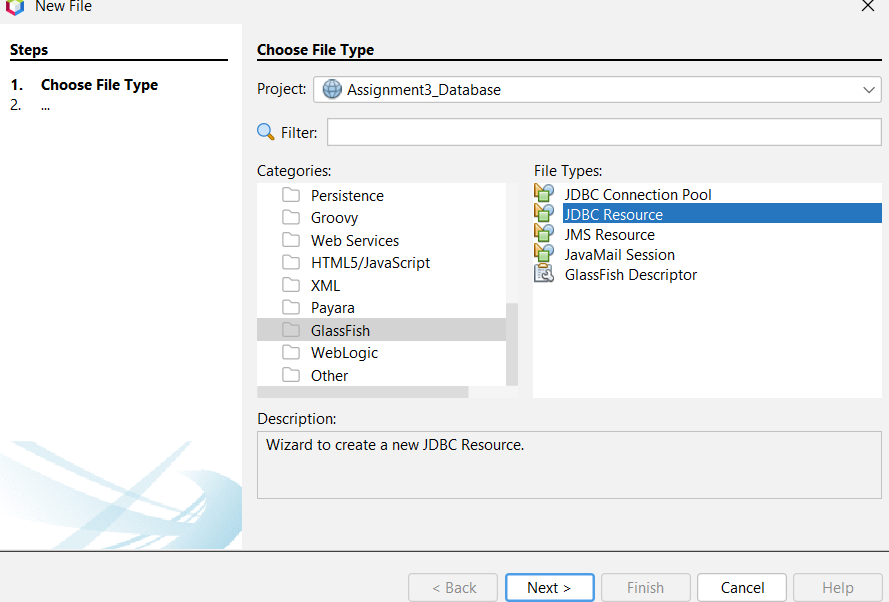
* 

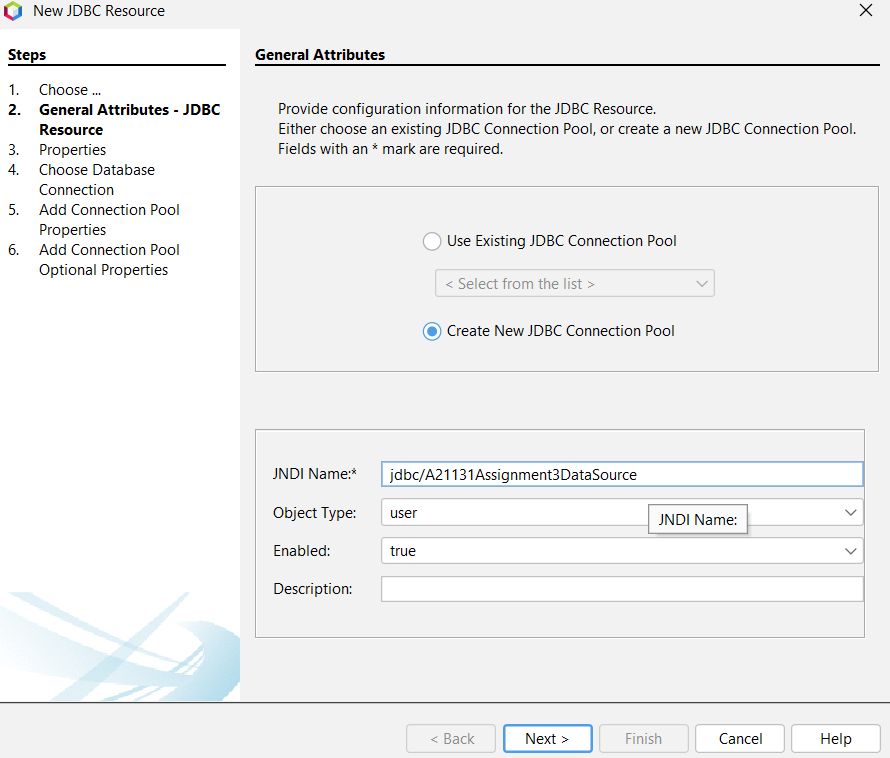
**JDBC URL:** “jdbc:sqlserver://localhost:1433;databaseName=A21131\_Assigment3;encrypt=true;trustServerCertificate=true”

* Next -> chọn dbo -> next -> finish

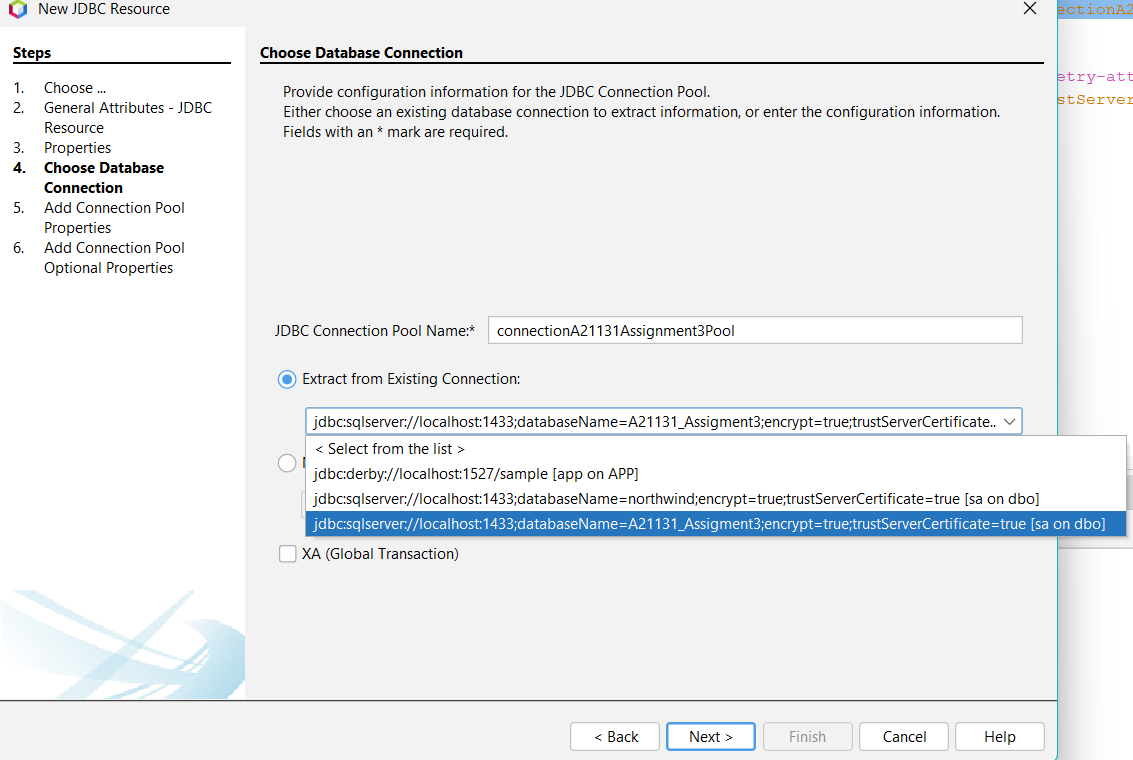
**Tạo glassfish resource**

Chuột phải Web Page -> New -> Other -> chọn như hình

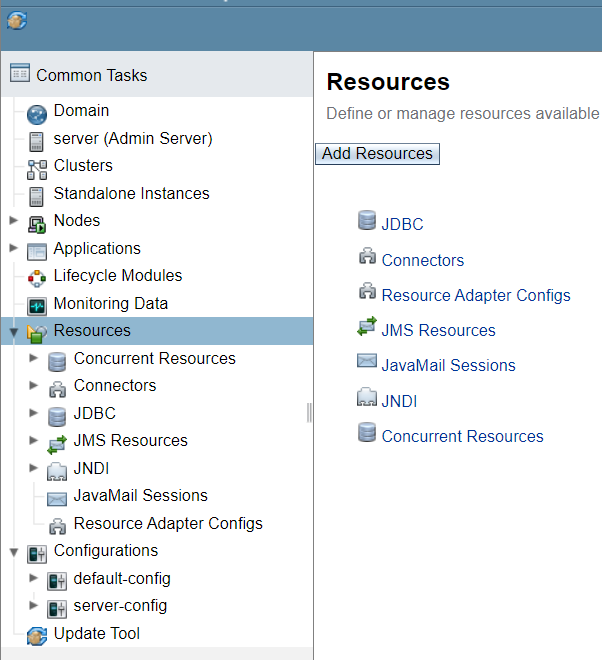




* Next -> next ->

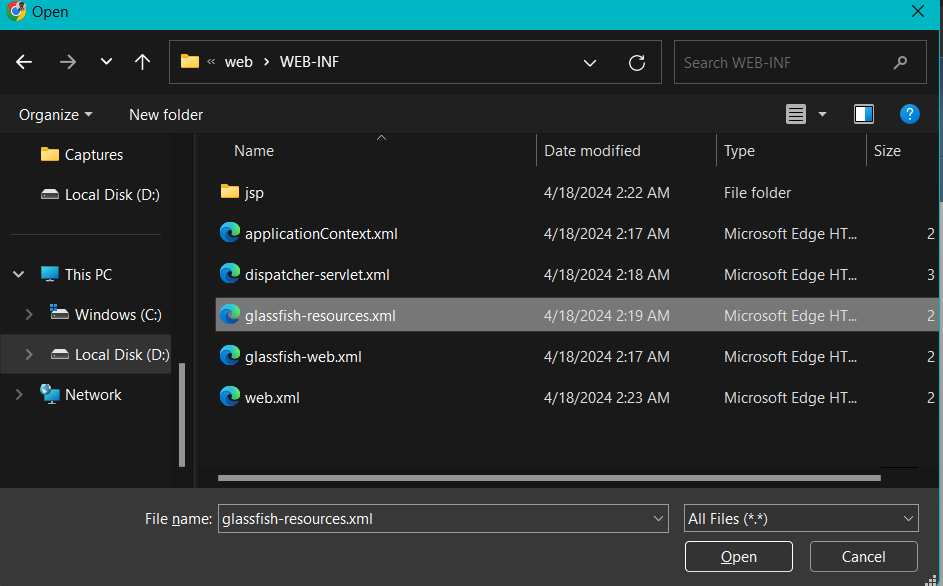


Sau khi tạo xong file glashfish resource -> truy cập localhost:4848 -> add resources

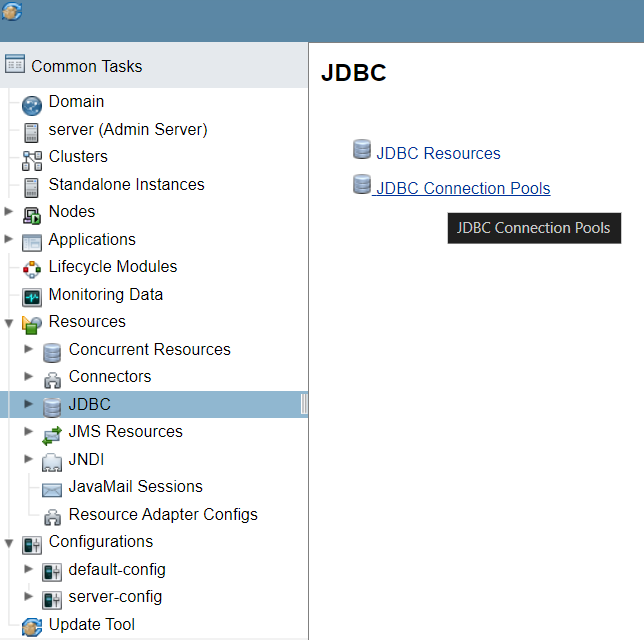


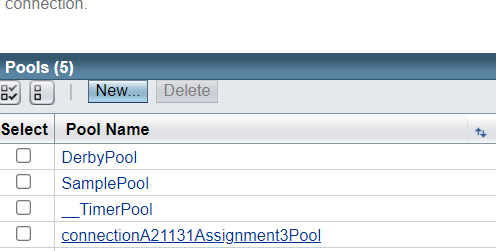
File resource nằm trong thư mục -> web -> web-inf

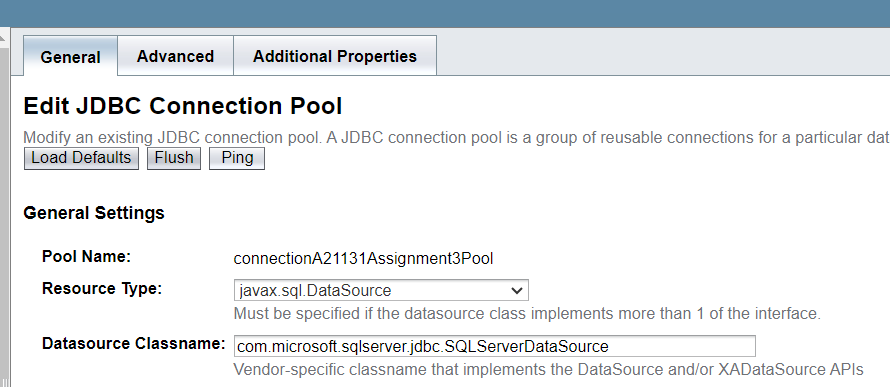
D:\A21131\Assignment3\_Database\web\WEB-INF



->sau khi chọn file xong nhấn ok, sau đó vô như hình để ping



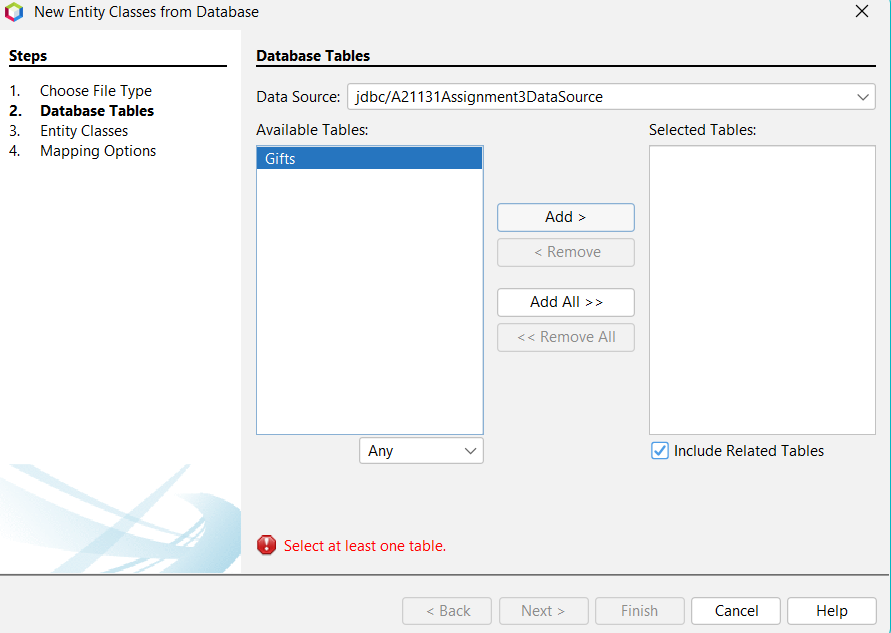




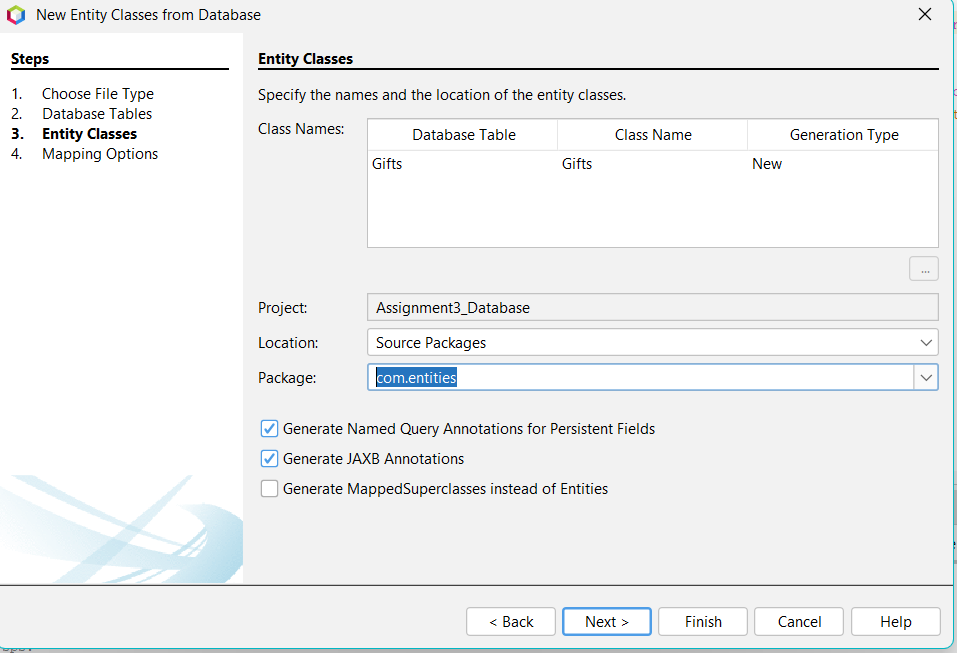
Sau khi ping thành công trở lại netbeans tạo entities và sessionbean

**com.entities**

chuột phải source packages -> new -> other -> Persistance -> Entity Classes from Database -> chọn như hình -> add -> next



Đặt tên như chỗ bôi đen trong hình dưới-> next -> finish



Sau khi tạo xong, vô file Gift.java thay đổi dòng “ private BigDecimal price; “ thành “ private Double price; “, **các chỗ báo đỏ BigDecimal trong file thì đổi thành Double tương tự**

**Orders.java**package com.models;

public class Orders {

private Integer giftOrderID;

private String giftOrderName;

private int giftOrderQuantity;

private double giftOrderDiscount;

private double giftOrderPrice;

private double discountedPrice;

private double totalPrice;

private String giftOrderColors;

private String giftOrderCategory;

/\*Getter and setter - constructor\*/

}

**Users.java**

package com.models;

public class Users {

private String name;

public String getName() {

return name != null && !name.isEmpty() ? name : "User";

}

public void setName(String name) {

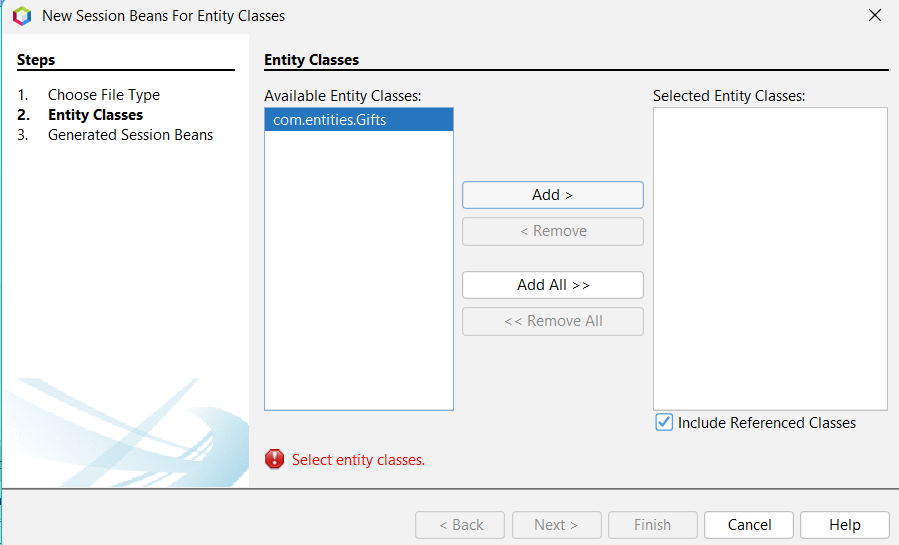
this.name = name;

}

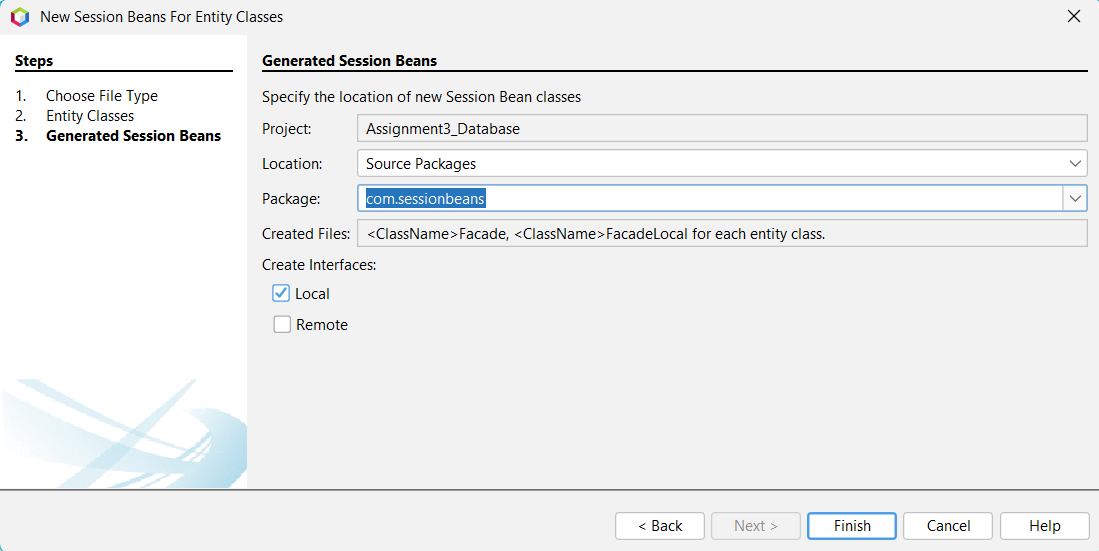
}

**com.sessionbeans**

chuột phải source packages -> new -> other -> Persistance -> Session Beans for Entity Classes -> chọn như hình



Chọn add -> next -> tick vô ô local -> đặt tên vô ô bôi đen như hình dưới -> finish



**com.controllers**

**GiftController.java**

package com.controllers;

import com.entities.Gifts;

import com.entities.Orders;

import com.entities.Users;

import com.sessionbeans.GiftsFacadeLocal;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.naming.Context;

import javax.naming.InitialContext;

import javax.naming.NamingException;

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

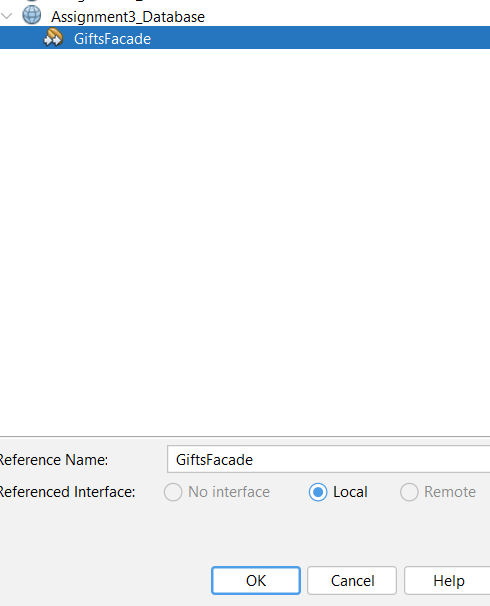
import org.springframework.web.bind.annotation.RequestParam;

@Controller

@RequestMapping("/")

public class GiftController {

GiftsFacadeLocal giftsFacade = lookupGiftsFacadeLocal(); **-> Chuột phải -> insert -> call enterprise bean -> chọn project đang làm -> hiện ra như hình -> chọn như hình -> ok**



List<Gifts> giftItems = giftsFacade.findAll();

@ModelAttribute("gifts")

public Gifts setGift() {

return new Gifts();

}

@RequestMapping(value = "index", method = RequestMethod.GET)

public String redirectGreetingPage(ModelMap model) {

model.addAttribute("userName", new Users());

model.addAttribute("gifts", giftItems);

return "index";

}

@RequestMapping(value = "greeting", method = RequestMethod.POST)

public String greeting(@ModelAttribute("userName") Users user, ModelMap model) {

String name = user.getName();

model.addAttribute("name", name);

return "greeting";

}

@RequestMapping(value = "gift/{giftID}", method = RequestMethod.GET)

public String displayGiftDetails(@PathVariable("giftID") Integer giftID, ModelMap model) {

model.addAttribute("gift", giftsFacade.find(giftID));

return "giftDetails";

}

@RequestMapping(value = "orders", method = RequestMethod.POST)

public String orderGifts(@RequestParam(required = false) List<Integer> giftIds, ModelMap model) {

if (giftIds == null || giftIds.isEmpty()) {

model.addAttribute("checkGiftIdEmpty", true);

model.addAttribute("initName", new Users());

model.addAttribute("gifts", giftItems);

return "index";

}

double totalPrice = 0.0;

final double MINPRICEDISCOUNT = 5000.00;

final double PRICEDISCOUNT = 0.9; // Apply 10% discount

int quantity = 0;

List<Orders> orderList = new ArrayList<>();

for (long giftId : giftIds) {

for (Gifts giftItem : giftItems) {

if (giftItem.getGiftID() == giftId ) {

quantity += 1;

double discount = giftItem.getPrice() >= MINPRICEDISCOUNT ? PRICEDISCOUNT : 0.0;

totalPrice = calculateDiscountPrice(giftItem.getPrice(), MINPRICEDISCOUNT, PRICEDISCOUNT, model);

Orders order = new Orders(giftItem.getGiftID(), giftItem.getGiftName(), quantity, discount, giftItem.getPrice(), totalPrice, totalPrice, giftItem.getColors(), giftItem.getCategory());

orderList.add(order);

break;

}

}

}

model.addAttribute("orderedGifts", orderList);

model.addAttribute("totalPrice", totalPrice);

return "orders";

}

private double calculateDiscountPrice(double amount, double discountPrice, double discount, ModelMap model) {

if (amount >= discountPrice) {

model.addAttribute("discountApplied", true);

return amount \* discount;

}

return amount;

}

**Đoạn này được tạo ra tự động từ bước call enterprise bean trên ->** private GiftsFacadeLocal lookupGiftsFacadeLocal() {

try {

Context c = new InitialContext();

return (GiftsFacadeLocal) c.lookup("java:global/Assignment3\_Database/GiftsFacade!com.sessionbeans.GiftsFacadeLocal");

} catch (NamingException ne) {

Logger.getLogger(getClass().getName()).log(Level.SEVERE, "exception caught", ne);

throw new RuntimeException(ne);

}

}

}

**VIEW CHUNG CỦA CẢ HAI CÁCH TRÊN**

**jsp/index.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@taglib prefix="form" uri="http://www.springframework.org/tags/form" %>

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Welcome to Gift Shop</title>

</head>

<body>

<div class="container">

<h1>Enter your name: </h1>

<form:form action="greeting.htm" method="POST" modelAttribute="userName">

<table>

<tbody>

<tr>

<td>

<form:input path="name" type="text" name="name" id="name" placeholder="Enter Your Name"/>

</td>

<td>

<input type="submit" value="Submit"/>

</td>

</tr>

</tbody>

</table>

</form:form>

<h2>Available Gifts</h2>

<c:if test="${checkGiftIdEmpty}">

<div>

Please choose at least one product to continue order

</div>

</c:if>

<form:form action="orders.htm" method="POST" modelAttribute="orders">

<table>

<thead>

<tr>

<th></th>

<th>Name</th>

<th>Category</th>

<th>Price</th>

<th></th>

</tr>

</thead>

<tbody>

<c:forEach items="${gifts}" var="gift">

<tr>

<td><input type="checkbox" name="giftIds" value="${gift.giftID}"></td>

<td>${gift.giftName}</td>

<td>${gift.category}</td>

<td>${gift.price}</td>

<td><a href="gift/${gift.giftID}.htm">Detail</a></td>

</tr>

</c:forEach>

</tbody>

</table>

<button type="submit">Order Gifts</button>

</form:form>

</div>

</body>

</html>

**jsp/greeting.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Greeting</title>

</head>

<body>

<h1>Hello, ${userName.name} !</h1>

<p>Welcome to our gift shop.</p>

<a href="index.htm" class="btn btn-primary">View Available Gifts</a>

</body>

</html>

**jsp/giftDetails.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Gift Search Page</title>

</head>

<body>

<h1>This is your product with id = ${gift.giftID}</h1>

<c:if test="${gift.price >= 5000}">

<p>This product has a 10% discount.</p>

</c:if>

<table>

<tbody>

<tr>

<th>Gift ID</th>

<td>${gift.giftID}</td>

</tr>

<tr>

<th>Gift Name</th>

<td>${gift.giftName}</td>

</tr>

<tr>

<th>Price</th>

<td>${gift.price}</td>

</tr>

<tr>

<th>Category</th>

<td>${gift.category}</td>

</tr>

<tr>

<th>Color</th>

<td>${gift.colors}</td>

</tr>

</tbody>

</table>

<a href="<c:url value="/index.htm"/>">Back to View Gifts</a>

</div>

</body>

</html>

**jsp/orders.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Order Place</title>

</head>

<body>

<h1>Order Summary</h1>

<c:if test="${discountApplied}">

<div>

Congratulations! You received a 10% discount for your order exceeding 5000 Ruby.

</div>

</c:if>

<table>

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Color</th>

<th>Category</th>

<th>Price (Ruby)</th>

<th>Discount (%)</th>

<th>Discounted Price (Ruby)</th>

</tr>

</thead>

<tbody>

<c:forEach var="gift" items="${orderedGifts}">

<tr>

<td>${gift.giftOrderID}</td>

<td>${gift.giftOrderName}</td>

<td>${gift.giftOrderColors}</td>

<td>${gift.giftOrderCategory}</td>

<td>${gift.giftOrderPrice}</td>

<td>${gift.giftOrderDiscount}</td>

<td>${discountedPrice}</td>

</tr>

</c:forEach>

</tbody>

</table>

<p>Total Price: ${totalPrice} Ruby</p>

<div>

<a href="<c:url value="/index.htm"/>">Back to Gift List</a>

</div>

</body>

</html>

**Comment dòng cuối trong file web.xml nếu sử dụng database -> như hình bên dưới**  
