МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та інформаційні технології управління»

ЗВІТ

з лабораторної роботи № 4

з дисципліни

«ПРОГРАМУВАННЯ ВЕБ-ЗАСТОСУВАНЬ JAVA EE»

ВИКОНАВ

Студент групи КН-36а

Кулик В.В.

ПЕРЕВІРИВ

Доцент каф. ПІІТУ

Бабкова Н. В.

Харків 2019

Тема

«ДОСЛІДЖЕННЯ ПРИНЦИПІВ СТВОРЕННЯ ВЕБ-ДОДАТКів ДЛЯ роботи із базами даних З використанням hibernate»

Завдання роботи

1. Розширити межі предметної області, додавши до її схеми ще одну сутність та ще один зв'язок. Внести необхідні зміни до структури БД. Створити необхідні класи для зберігання нових даних у додатку.
2. Додати до web-проекту можливості автоматичної сборки maven. У файл pom.xml додати залежності для роботи із БД. із сервлетами, із EL та JSTL. а також Hibernate.
3. Виконати маппинг класів для зберігання даних анотаціями JPA.
4. Переробити методи доступу до даних, використовуючи принципи доступу технології Hibernate. При цьому не використовувати мови запитів.
5. Допрацювати функціонал проекту, отриманого на Лабораторній №3, додавши веб-сторінки для заповнення нових таблиць. При розробці звернути увагу на контроль коректності даних, що вводяться.

Хід роботи

1. Була обрана предметна область “Магазин комп’ютерних комплектуючих”. Маємо чотири сутності (класи): Manufacturer – Product – Orders – Providers. Веб-застосунок дозволяє працівникам магазину контролювати замовлення від постачальника. У замовленнях відображається назва товару, його постачальник, кількість та остаточна вартість всього замовлення. Інтерфейс дозволяє редагувати та фільтрувати дані.

Нижче представлена структура проекту:

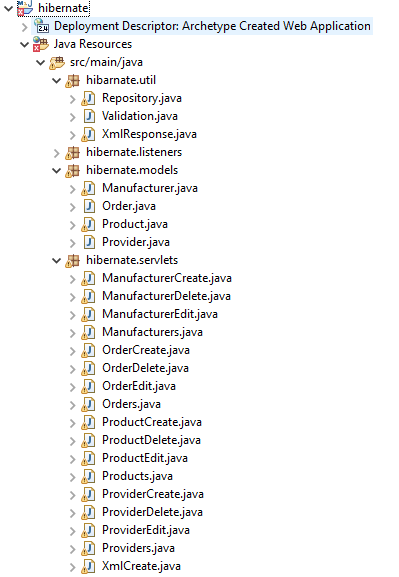


Рисунок 1 – Структура проекту

Нижче представлена структура pom файлу:



Рисунок 2 – Структура pow

Код маппованого классу Manufacturer з реалізацією методів Create, Delete, Edit:

package hibernate.models;

import java.sql.Connection;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.Set;

import javax.persistence.\*;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

@Entity

@Table(name="manufacturers")

public class Manufacturer {

private int id;

private String name;

public static SessionFactory factory;

private Set<Product> products;

public Manufacturer(String name) {

super();

this.name = name;

}

public Manufacturer() {

}

@Id

@GeneratedValue

@Column(name = "manufacturer\_id")

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

@Column(nullable = false, unique = true)

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@OneToMany(cascade = CascadeType.ALL, mappedBy = "manufacturer", orphanRemoval = true)

public Set<Product> getProducts() {

return this.products;

}

public void setProducts(Set<Product> products) {

this.products = products;

}

public boolean create() throws PersistenceException{

Session session = factory.openSession();

session.beginTransaction();

session.save(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean delete(){

Session session = factory.openSession();

session.beginTransaction();

session.delete(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean edit(){

Session session = factory.openSession();

session.beginTransaction();

session.update(this);

session.getTransaction().commit();

session.close();

return true;

}

}

Код маппованого классу Order з реалізацією методів Create, Delete, Edit:

package hibernate.models;

import javax.persistence.\*;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

@Entity

@Table(name = "orders")

public class Order {

private int id;

private Product product;

private Provider provider;

private int quantity;

public static SessionFactory factory;

public Order(Product product, Provider provider, int quantity) {

super();

this.product = product;

this.provider = provider;

this.quantity = quantity;

}

public Order() {

}

@Column(name = "quantity", nullable = false)

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

@Id

@GeneratedValue

@Column(name = "id")

public int getId() {

return this.id;

}

public void setId(int id) {

this.id = id;

}

@ManyToOne

@JoinColumn(name = "product\_id")

public Product getProduct() {

return product;

}

public void setProduct(Product product) {

this.product = product;

}

@ManyToOne

@JoinColumn(name = "provider\_id")

public Provider getProvider() {

return provider;

}

public void setProvider(Provider provider) {

this.provider = provider;

}

public boolean create(){

Session session = factory.openSession();

session.beginTransaction();

session.save(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean delete(){

Session session = factory.openSession();

session.beginTransaction();

session.delete(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean edit(){

Session session = factory.openSession();

session.beginTransaction();

session.update(this);

session.getTransaction().commit();

session.close();

return true;

}

}

Код маппованого классу Product з реалізацією методів Create, Delete, Edit:

package hibernate.models;

import java.util.List;

import java.util.Set;

import javax.persistence.\*;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

@Entity

@Table(name = "products")

public class Product{

private int id;

private String name;

private double price;

private Manufacturer manufacturer;

private List<Order> orders;

@OneToMany(mappedBy="product")

public List<Order> getOrders() {

return orders;

}

public void setOrders(List<Order> orders) {

this.orders = orders;

}

public static SessionFactory factory;

public Product(String name, double price, Manufacturer manufacturer) {

super();

this.name = name;

this.price = price;

this.manufacturer = manufacturer;

}

public Product() {

}

@Id

@GeneratedValue(strategy=GenerationType.AUTO)

@Column(name = "id")

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

@Column(nullable = false, unique = true)

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Column(nullable = false)

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

@ManyToOne

@JoinColumn(name= "manufacturer\_id")

public Manufacturer getManufacturer() {

return manufacturer;

}

public void setManufacturer(Manufacturer manufacturer) {

this.manufacturer = manufacturer;

}

public boolean create() throws PersistenceException{

Session session = factory.openSession();

session.beginTransaction();

session.save(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean delete(){

Session session = factory.openSession();

session.beginTransaction();

session.delete(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean edit(){

Session session = factory.openSession();

session.beginTransaction();

session.update(this);

session.getTransaction().commit();

session.close();

return true;

}

}

Код маппованого классу Provider з реалізацією методів Create, Delete, Edit :

package hibernate.models;

import java.util.List;

import javax.persistence.\*;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

@Entity

@Table(name = "providers")

public class Provider {

private int id;

private String name;

private String email;

private List<Order> orders;

@OneToMany(mappedBy="provider")

public List<Order> getOrders() {

return orders;

}

public void setOrders(List<Order> orders) {

this.orders = orders;

}

public static SessionFactory factory;

public Provider(String name, String email) {

super();

this.name = name;

this.email = email;

}

public Provider() {

}

@Id

@GeneratedValue(strategy=GenerationType.AUTO)

@Column(name = "id")

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

@Column(nullable = false, unique = true)

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Column(nullable = false, unique = true)

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public boolean create() throws PersistenceException{

Session session = factory.openSession();

session.beginTransaction();

session.save(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean delete(){

Session session = factory.openSession();

session.beginTransaction();

session.delete(this);

session.getTransaction().commit();

session.close();

return true;

}

public boolean edit(){

Session session = factory.openSession();

session.beginTransaction();

session.update(this);

session.getTransaction().commit();

session.close();

return true;

}

}

Файл hibernate.cfg.xml:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 5.3//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-5.3.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hbm2ddl.auto"*>update</property>

<property name=*"dialect"*>org.hibernate.dialect.SQLiteDialect</property>

<property name=*"connection.url"*>jdbc:sqlite:e:\\Program Files\\SQLite\\shop.db</property>

<property name=*"connection.driver\_class"*>org.sqlite.JDBC</property>

<property name=*"hibernate.show\_sql"*>true</property>

<mapping class=*"hibernate.models.Manufacturer"*/>

<mapping class=*"hibernate.models.Product"*/>

<mapping class=*"hibernate.models.Provider"*/>

<mapping class=*"hibernate.models.Order"*/>

</session-factory>

</hibernate-configuration>

Код підключення до БД:

package hibernate.listeners;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.concurrent.CopyOnWriteArrayList;

import javax.servlet.ServletContext;

import javax.servlet.ServletContextEvent;

import javax.servlet.ServletContextListener;

import org.hibernate.SessionFactory;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import hibarnate.util.Repository;

import hibernate.models.Manufacturer;

import hibernate.models.Order;

import hibernate.models.Product;

import hibernate.models.Provider;

public class ContextListener implements ServletContextListener {

private SessionFactory factory;

public void contextInitialized(ServletContextEvent sce) {

final ServletContext ctx = sce.getServletContext();

StandardServiceRegistry registry = new StandardServiceRegistryBuilder().configure().build();

factory = new MetadataSources(registry).buildMetadata().buildSessionFactory();

Manufacturer.factory = Provider.factory = Order.factory = Product.factory = factory;

Repository repo = new Repository(factory);

ctx.setAttribute("manufacturers", repo.getManufacturers());

ctx.setAttribute("products", repo.getProducts());

ctx.setAttribute("providers", repo.getProviders());

ctx.setAttribute("orders", repo.getOrders());

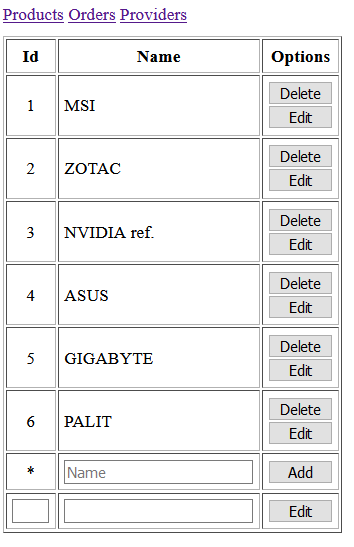
}

public void contextDestroyed(ServletContextEvent sce) {

}

}

Нижче представлена сторінка виробників, яка була додана за вимогою цієї лабораторної роботи:



Код відповідної JSP:

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Manufacturers</title>

<link rel=*"stylesheet"* href=*"styles/main.css"*>

<link rel=*"stylesheet"* href=*"styles/dropdown.css"*>

</head>

<body>

<div class=*"navigator"*>

<div class=*"dropdown"*>

<a href=*"/hibernate"*>Products</a>

<div class=*"dropdown-content"*>

<a href=*"Manufacturers"*>Manufacturers</a>

</div>

</div>

<a href=*"Orders"*>Orders</a>

<a href=*"Providers"*>Providers</a>

</div>

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

<tr>

<th>Id</th>

<th>Name</th>

<th>Options</th>

</tr>

<c:forEach var = *"manufacturer"* items=*"*${requestScope.manufacturers}*"*>

<tr>

<td class=*"id"*>${manufacturer.id}</td>

<td>${manufacturer.name}</td>

<td>

<form method=*"POST"* action=*"ManufacturerDelete"*>

<input hidden name=*"manufacturer"* value=*"*${manufacturer.id}*"*>

<input type=*"submit"* value=*"Delete"*>

</form>

<form method=*"GET"* action=*"ManufacturerEdit"*>

<input hidden name=*"manufacturer"* value=*"*${manufacturer.id}*"*>

<input type=*"submit"* value=*"Edit"*>

</form>

</td>

</tr>

</c:forEach>

<tr>

<form method=*"POST"* action=*"ManufacturerCreate"*>

<td class=*"id"*>\*</td>

<td><input type=*"text"* name=*"name"* placeholder=*"Name"*></td>

<td><input type=*"submit"* value=*"Add"*></td>

</form>

</tr>

<tr>

<form method=*"POST"* action=*"ManufacturerEdit"*>

<td class=*"id"*><input type=*"text"* readonly size=*"1"* name=*"manufacturer"* value=*"*${requestScope.manufacturer.id}*"*></td>

<td><input type=*"text"* name=*"name"* value=*"*${requestScope.manufacturer.name}*"*></td>

<td><input type=*"submit"* value=*"Edit"*></td>

</form>

</tr>

</table>

</body>

</html>

**Висновок**: на даній лабораторній роботі були вдосконалені навички роботи зі стеком технологій для створення веб-додатку на Java, за допомогою створення Servlet та jsp файлів. Також був досліджений алгоритм підключення бази даних SQLite через Hibernate, за допомогою якого були реалізовані CRUD-методи без використання мови запитів SQL. Також були отримані навички маппування класів.