package com.nttdata.petstore.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Iterator;

import java.util.List;

import org.apache.log4j.Logger;

import com.nttdata.petstore.dbcon.ConnectionHolder;

import com.nttdata.petstore.dbcon.DBConnectionException;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.dbfw.DBHelper;

import com.nttdata.petstore.dbfw.ParamMapper;

import com.nttdata.petstore.domain.Cart;

import com.nttdata.petstore.domain.CartItem;

public class OrderDAO {

public static final Logger LOG = Logger.getLogger("ProductDAO.class");

// placing an order in the cart.

public Object placeOrder(final Cart shoppingCart) {

Integer place = null;

try {

place = insertNewOrder(shoppingCart);

} catch (DBFWException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

return place;

}

// inserting a new order in the list, a user want to buy.

public int insertNewOrder(final Cart shoppingCart) throws DBFWException {

boolean isUpdated = false;

int result = 0;

Connection con = null;

ConnectionHolder holder=null;

try {

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

List cartList = shoppingCart.getItemDetails();

Iterator iterator = cartList.iterator();

while (iterator.hasNext()) {

final CartItem cartItem = (CartItem) iterator.next();

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

// TODO Auto-generated method stub

preStmt.setInt(1, shoppingCart.getOrderId());

preStmt.setString(2, shoppingCart.getCustId());

preStmt.setInt(3, cartItem.getItem().getItemId());

preStmt.setInt(4, cartItem.getItem().getProductId());

preStmt.setInt(5, cartItem.getItem().getCategoryId());

preStmt.setInt(6, cartItem.getQuantity());

}

};

try {

result = DBHelper.executeUpdate(con,

SQLMapper.INSERT\_PURCHASE\_DETAILS, mapParam);

} catch (DBFWException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

throw new DBFWException(e.getMessage());

}

if (result > 1) {

isUpdated = true;

}

}

return result;

}

public Object getPurchaseDetails(final int orderId) throws DBFWException {

ResultSet resSet = null;

Connection con = null;

ConnectionHolder holder=null;

PreparedStatement preStmt = null;

List<Object> products;

try {

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

// TODO Auto-generated method stub

preStmt.setInt(1, orderId);

}

};

products = DBHelper.executeSelect(con, SQLMapper.SELECT\_NEW\_PURCHASE,

SQLMapper.RES\_MAPPER\_THREE, mapParam);

System.out.println(products);

return products;

}

// method for generating random OrderID for the new List of Purchases.

public int generateOrderID() throws DBFWException, PetStoreDAOException {

List orderId;

int order = 0;

int newOrder = 0;

Connection con = null;

ConnectionHolder holder=null;

try {

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

orderId = DBHelper.executeSelect(con, SQLMapper.SELECT\_ORDERID,

SQLMapper.RES\_MAPPER\_SEVEN);

Iterator iterate = orderId.iterator();

while (iterate.hasNext()) {

order = (Integer) iterate.next();

newOrder = order + 1;

System.out.println(order);

}

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

LOG.error("Throws a DBFW exception");

throw new PetStoreDAOException(e.getMessage());

}finally{

try {

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

LOG.error("Throws a DBFW exception");

//e.printStackTrace();

throw new PetStoreDAOException(e.getMessage());

}

return newOrder;

}

}

}

package com.nttdata.petstore.dao;

public class PetStoreDAOException extends Exception {

public PetStoreDAOException(String message) {

super(message);

// TODO Auto-generated constructor stub

}

public PetStoreDAOException(String message, Throwable cause) {

super(message, cause);

// TODO Auto-generated constructor stub

}

}------------------------------------------------------------

package com.nttdata.petstore.dao;

import static org.junit.Assert.\*;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

import org.junit.Test;

import com.nttdata.petstore.dbcon.DBConnectionException;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.domain.Category;

import com.nttdata.petstore.domain.Item;

import com.nttdata.petstore.domain.Product;

public class ProductDAOTest {

// @Test

// public void testGetCategories() {

// fail("Not yet implemented");

// }

@Test

public void testGetCatbyID() throws PetStoreDAOException, DBFWException {

//fail("Not yet implemented");

Category expectedDetails = new Category();

Category actItem = new Category();

expectedDetails.setCategoryId(31);

expectedDetails.setCategoryName("DOG");

expectedDetails.setCategoryDescription("German Shepherd");

ProductDAO data = new ProductDAO();

List actualList = null;

// try {

actualList = data.getCatbyID(31);

assertNotNull(actualList);

// } catch (DBFWException e) {

// // TODO Auto-generated catch block

// e.printStackTrace();

// }

// Iterator iterate = actualList.iterator();

// if (iterate.hasNext()) {

// actItem = (Category) iterate.next();

// }

// assertTrue(expectedDetails.equals(actItem));

}

@Test

public void testGetProductList() {

// fail("Not yet implemented");

Product expecedDetail = new Product();

Product actItem = new Product();

expecedDetail.setProductId(81);

expecedDetail.setCategoryId(31);

expecedDetail.setProductDesc("bone biscuits");

expecedDetail.setProductName("Pedigree");

ProductDAO data = new ProductDAO();

List actualList = null;

try {

actualList = data.getProductList(31);

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

Iterator<Product> iterate = actualList.iterator();

if (iterate.hasNext()) {

actItem = iterate.next();

}

assertTrue(expecedDetail.equals(actItem));

}

@Test

public void testGetProduct() {

//fail("Not yet implemented");

Product expecedDetail = new Product();

Product actItem = new Product();

expecedDetail.setProductId(81);

expecedDetail.setCategoryId(31);

expecedDetail.setProductDesc("bone biscuits");

expecedDetail.setProductName("Pedigree");

ProductDAO data = new ProductDAO();

List actualList = null;

try {

actualList = data.getProduct(31, 81);

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

Iterator iterate = actualList.iterator();

while (iterate.hasNext()) {

actItem = (Product) iterate.next();

}

assertTrue(expecedDetail.equals(actItem));

}

//

@Test

public void testGetItemList() {

// fail("Not yet implemented");

Item expecedDetail = new Item();

Item actItem = new Item();

expecedDetail.setItemId(12);

expecedDetail.setProductId(81);

expecedDetail.setCategoryId(31);

expecedDetail.setItemName("Healthy Biscuits");

expecedDetail.setItemDescription("healthy food for dogs");

expecedDetail.setItemPrice(525);

List expect = new ArrayList();

expect.add(expecedDetail);

ProductDAO data = new ProductDAO();

List actualList = null;

try {

boolean isTrue;

actualList = data.getItemList(31, 81);

isTrue = expect.containsAll(actualList);

System.out.println(isTrue);

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

////

@Test

public void testGetItem() throws DBConnectionException {

//fail("Not yet implemented");

Item expecedDetail = new Item();

Item act = new Item();

expecedDetail.setItemId(12);

expecedDetail.setProductId(81);

expecedDetail.setCategoryId(31);

expecedDetail.setItemName("Healthy Biscuits");

expecedDetail.setItemDescription("healthy food for dogs");

expecedDetail.setItemPrice(525);

ProductDAO data = new ProductDAO();

List expect = new ArrayList();

expect.add(expecedDetail);

List actualList = null;

try {

actualList = data.getItem(31, 81, 12);

boolean isTrue;

//actualList = data.getItem(25895, 1234, 26);

isTrue = expect.containsAll(actualList);

System.out.println(isTrue);

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

package com.nttdata.petstore.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.List;

import org.apache.log4j.Logger;

import com.nttdata.petstore.dbcon.ConnectionHolder;

import com.nttdata.petstore.dbcon.DBConnectionException;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.dbfw.DBHelper;

import com.nttdata.petstore.dbfw.ParamMapper;

import com.nttdata.petstore.domain.Category;

import com.nttdata.petstore.domain.Item;

import com.nttdata.petstore.domain.Product;

public class ProductDAO {

public static final Logger LOG = Logger.getLogger("ProductDAO.class");

// return all the categories list

public List<Category> getCategories() throws DBConnectionException {

List<Category> categoryList = null;

Connection con = null;

ConnectionHolder holder=null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

LOG.debug("connecting to the dbhelper to execute select query");

categoryList = DBHelper.executeSelect(con,

SQLMapper.SELECT\_ALL\_CATEG, SQLMapper.RES\_MAPPER);

} catch (DBFWException e) {

// TODO Auto-generated catch block

LOG.error("Throws a DBFW exception");

e.printStackTrace();

}

return categoryList;

}

// returns a particular list matching the categId;

public List<Category> getCatbyID(final int categId) throws DBFWException, PetStoreDAOException {

Connection con = null;

List<Category> categoryList;

PreparedStatement prestmt = null;

ConnectionHolder holder=null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e1) {

// TODO Auto-generated catch block

//e1.printStackTrace();

throw new PetStoreDAOException(e1.getMessage());

}

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException {

// TODO Auto-generated method stub

try {

LOG.debug("setting the values");

preStmt.setInt(1, categId);

} catch (SQLException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

LOG.error("Throws a DBFW exception");

throw new DBFWException();

}

}

};

LOG.debug("connecting to the dbhelper for execute select");

categoryList = DBHelper.executeSelect(con, SQLMapper.SELECT\_CATEGORIES,

SQLMapper.RES\_MAPPER, mapParam);

return categoryList;

}

// return all the products as in the product table;

public List getProductList(final int categId) throws DBFWException {

Connection con = null;

ConnectionHolder holder=null;

List<Product> productList;

PreparedStatement prestmt = null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

LOG.debug("Inserting value");

preStmt.setInt(1, categId);

}

};

LOG.debug("connecting to the dbhelper to execute select query");

productList = DBHelper.executeSelect(con,

SQLMapper.SELECT\_PRODUCT\_LIST\_ON\_CATEG,

SQLMapper.RES\_MAPPER\_ONE, mapParam);

return productList;

}

// gets the specific product list matching the categId and the productId;

public List getProduct(final int categId, final int productId)

throws DBFWException {

// will return a single list;

Connection con = null;

ConnectionHolder holder=null;

List<Product> specificProductList;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

LOG.debug("Inserting the values");

preStmt.setInt(1, categId);

preStmt.setInt(2, productId);

}

};

LOG.debug("connecting to the DBhelper to execute select query");

specificProductList = DBHelper.executeSelect(con,

SQLMapper.SELECT\_SPECIFIC\_PRODUCT\_LIST\_ON\_CATEG,

SQLMapper.RES\_MAPPER\_ONE, mapParam);

return specificProductList;

}

// for the given productid and the categId get the item details;

public List getItemList(final int categId, final int prodId)

throws DBFWException {

Connection con = null;

ConnectionHolder holder=null;

List<Item> itemList;

try {

LOG.info("Establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

LOG.debug("Inserting the value");

preStmt.setInt(1, categId);

preStmt.setInt(2, prodId);

}

};

LOG.debug("connecting to the dbhelper for execute select");

itemList = DBHelper.executeSelect(con, SQLMapper.SELECT\_ITEMS\_ALL,

SQLMapper.RES\_MAPPER\_SIX, mapParam);

return itemList;

}

// for the given product id and the custid AND THE ITEM ID get the singular

// list;

public List getItem(final int categId, final int prodId, final int itemId)

throws DBFWException, DBConnectionException {

// return a single statement;

Connection con = null;

ConnectionHolder holder=null;

List<Item> specificItemList=null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

LOG.debug("Inserting values");

preStmt.setInt(1, categId);

preStmt.setInt(2, prodId);

preStmt.setInt(3, itemId);

}

};

LOG.debug("connecting to the dbhelper to execute select query");

specificItemList = DBHelper.executeSelect(con,

SQLMapper.SELECT\_ITEMS\_PARTICULAR, SQLMapper.RES\_MAPPER\_SIX,

mapParam);

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

try {

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

throw new DBConnectionException(e.getMessage());

}

}

return specificItemList;

}

}-----------------------------------------------------------

package com.nttdata.petstore.dao;

import java.sql.ResultSet;

import java.sql.SQLException;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.dbfw.ResultMapper;

import com.nttdata.petstore.domain.Category;

import com.nttdata.petstore.domain.Customer;

import com.nttdata.petstore.domain.Item;

import com.nttdata.petstore.domain.Product;

public class SQLMapper {

public static final String INSERT\_PRODUCT\_LINE\_DETAILS = "insert into Product\_Line\_Details\_89517 values (?,?,?,?,?,?)";

public static final String INSERT\_PURCHASE\_DETAILS = "insert into Purchase\_Detail\_89517 values(?,?,?,?,?,?)";

public static final String SELECT\_VALIDATE = "select \* from User\_89517 where CustID =? and password =?";

public static final String INSERT\_USER = "insert into User\_89517 values(?,?)";

public static final String INSERT\_CUSTOMER\_DETAILS = "insert into Customer\_89517 values(?,?,?,?,?,?,?)";

public static final String INSERT\_CREDIT\_INFO = "insert into CreditCard\_Info\_89517 values(?,?,?)";

public static final String SELECT\_PURCHASE = "select \* from Purchase\_Detail\_89517 where OrderID =?";

public static final String SELECT\_NEW\_PURCHASE = " select p.ItemID,p.ProdID,p.CategID,p.ItemName,p.ItemDesc,p.Price from Product\_Line\_Items\_89517 p inner join Purchase\_Detail\_89517 d on p.ItemID=d.ItemID where OrderID=?";

public static final String SELECT\_PURCHASE\_DETAILS = "select s.ItemID, s.ProdID, s.CategID, ItemDesc, ItemName,Price from Purchase\_detail\_89517 s inner join Category\_Product\_89517 d on s.ProdID = d.ProdID where OrderId =?";

public static final String INSERT\_PLACE\_ORDER = "insert into Purchase\_Detail\_89517 values (?,?,?,?) where OrderID = ?";

public static final String SELECT\_ALL\_CATEG = "select \* from Categories\_89517";

public static final String SELECT\_CATEGORIES = "select \* from Categories\_89517 where CategID =?";

public static final String SELECT\_SEPCIFIC\_CATEGORIES = "select Category, ProdID, CategID, ProdDesc, ProdName from Categories\_89517 c inner join Category\_Proucts\_89517 e on c.CategID = e.CategID where CategID = ?";

public static final String SELECT\_PRODUCT\_LIST\_ON\_CATEG = "select ProdID, CategID, ProdName, ProdDesc from Category\_Products\_89517 where CategID =?";

public static final String SELECT\_SPECIFIC\_PRODUCT\_LIST\_ON\_CATEG = "select ProdID, CategID, ProdName, ProdDesc from Category\_Products\_89517 where CategID =? and ProdID=?";

public static final String SELECT\_ITEMS\_ALL = "select \* from Product\_Line\_Items\_89517 where CategID = ? and ProdID = ?";

public static final String SELECT\_ITEMS\_PARTICULAR = "select ItemID, ProdID, CategID, ItemName, ItemDesc, Price from Product\_Line\_Items\_89517 where CategID = ? and ProdID=? and ItemID =?";

public static final String SELECT\_ORDERID = "select max(OrderID) as ordernew from Purchase\_Detail\_89517";

public static final ResultMapper RES\_MAPPER = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

return new Category(resSet.getInt("CategID"),

resSet.getString("CategName"),

resSet.getString("CategDesc"));

}

};

public static final ResultMapper RES\_MAPPER\_ONE = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

return new Product(resSet.getInt("ProdID"),

resSet.getInt("CategID"), resSet.getString("ProdName"),

resSet.getString("ProdDesc"));

}

};

public static ResultMapper RES\_MAPPER\_TWO = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

Customer customer = new Customer();

String custid = resSet.getString("CustID");

String pass = resSet.getString("Password");

// TODO Auto-generated method stub

customer.setCustId(custid);

customer.setPasword(pass);

return customer;

}

};

public static ResultMapper RES\_MAPPER\_THREE = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

return new Item(resSet.getInt("ItemID"), resSet.getInt("ProdID"),

resSet.getInt("CategID"), resSet.getString("ItemName"),

resSet.getString("ItemDesc"), resSet.getInt("Price"));

}

};

public static ResultMapper RES\_MAPPER\_FOUR = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

return new Customer(resSet.getString("FirstName"),

resSet.getString("LastName"), resSet.getString("DOB"),

resSet.getString("Address"), resSet.getInt("ContactNo"));

}

};

public static ResultMapper RES\_MAPPER\_FIVE = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

return new Customer(resSet.getInt("CreditCardno"),

resSet.getString("Credit"), resSet.getString("ExpiryDt"));

}

};

public static ResultMapper RES\_MAPPER\_SIX = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

return new Item(resSet.getInt("ItemID"), resSet.getInt("ProdID"),

resSet.getInt("CategID"), resSet.getString("ItemName"),

resSet.getString("ItemDesc"), resSet.getInt("Price"));

}

};

public static ResultMapper RES\_MAPPER\_SEVEN = new ResultMapper() {

public Object mapRow(ResultSet resSet) throws DBFWException,

SQLException {

// TODO Auto-generated method stub

int orderNew = resSet.getInt("ordernew");

return orderNew;

}

};

}

package com.nttdata.petstore.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import org.apache.log4j.Logger;

import com.nttdata.petstore.dbcon.ConnectionHolder;

import com.nttdata.petstore.dbcon.DBConnectionException;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.dbfw.DBHelper;

import com.nttdata.petstore.dbfw.ParamMapper;

import com.nttdata.petstore.domain.Customer;

public class UserDAO {

public static final Logger LOG = Logger.getLogger("ProductDAO.class");

public boolean validateUser(String userId, String password)

throws PetStoreDAOException {

boolean isValidated = false;

Connection con = null;

ConnectionHolder holder=null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

// List list = new ArrayList();

// Customer customer = new Customer();

isValidated = DBHelper.validateUser(con, SQLMapper.SELECT\_VALIDATE,

SQLMapper.RES\_MAPPER\_TWO, userId, password);

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

try {

LOG.info("closing Connection");

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

return isValidated;

}

public Object registerUser(final Customer customerObject)

throws PetStoreDAOException, DBConnectionException {

boolean isInserted = false;

int rows = 0;

int rowsOne = 0;

int rowsTwo = 0;

Connection con = null;

ConnectionHolder holder=null;

try {

LOG.info("establishing Connection");

holder=ConnectionHolder.getInstance();

con=holder.getConnection();

ParamMapper mapParam = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

// TODO Auto-generated method stub

preStmt.setString(1, customerObject.getCustId());

preStmt.setString(2, customerObject.getPasword());

}

};

ParamMapper mapParamTwo = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

// TODO Auto-generated method stub

preStmt.setInt(1, customerObject.getCreditCardno());

preStmt.setString(2, customerObject.getCardType());

preStmt.setString(3, customerObject.getCardExpiryDate());

}

};

ParamMapper mapParamOne = new ParamMapper() {

public void mapParams(PreparedStatement preStmt)

throws DBFWException, SQLException {

// TODO Auto-generated method stub

preStmt.setString(1, customerObject.getCustId());

preStmt.setString(2, customerObject.getFirstName());

preStmt.setString(3, customerObject.getLastName());

preStmt.setString(4, customerObject.getDateOfBirth());

preStmt.setString(5, customerObject.getAddress());

preStmt.setInt(6, customerObject.getContactNumber());

preStmt.setInt(7, customerObject.getCreditCardno());

}

};

LOG.info("connecting to the dbhelper to execute update query");

rowsTwo = DBHelper.executeUpdate(con, SQLMapper.INSERT\_CREDIT\_INFO,

mapParamTwo);

rowsOne = DBHelper.executeUpdate(con,

SQLMapper.INSERT\_CUSTOMER\_DETAILS, mapParamOne);

rows = DBHelper.executeUpdate(con, SQLMapper.INSERT\_USER, mapParam);

if ((rows > 0) && (rowsOne > 0) && (rowsTwo > 0)) {

isInserted = true;

System.out.println(isInserted);

} else {

isInserted = false;

System.out.println(isInserted);

}

} catch (DBConnectionException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

throw new DBConnectionException(e.getMessage());

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

LOG.error("Throws a DBFW exception");

throw new DBConnectionException(e.getMessage());

}finally{

try {

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

}

}

return customerObject;

}

}-------------------------------------------

package com.nttdata.petstore.dao;

import static org.junit.Assert.\*;

import org.junit.Test;

public class UserDAOTest {

@Test

public void testValidateUser() throws PetStoreDAOException {

UserDAO dao1 = new UserDAO();

boolean isTrue;

isTrue = dao1.validateUser("2KL", "pass");

assertEquals(true, isTrue);

}

}

package com.nttdata.petstore.dao;

import static org.junit.Assert.\*;

import java.util.Iterator;

import java.util.List;

import org.junit.Test;

import com.nttdata.petstore.dbfw.DBFWException;

import com.nttdata.petstore.domain.Cart;

import com.nttdata.petstore.domain.CartItem;

import com.nttdata.petstore.domain.Item;

public class OrderDAOTest {

@Test

public void testPlaceOrder() {

//fail("Not yet implemented");

Item expectedDetail = new Item();

expectedDetail.setItemId(12);

expectedDetail.setProductId(81);

expectedDetail.setCategoryId(31);

expectedDetail.setItemName("Healthy Biscuits");

expectedDetail.setItemDescription("healthy food for dogs");

expectedDetail.setItemPrice(525);

OrderDAO data = new OrderDAO();

Object actualList=null;

try {

actualList = data.getPurchaseDetails(121);

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

assertTrue(expectedDetail.equals(actualList));

}

@Test

public void testInsertNewOrder() {

fail("Not yet implemented");

}

@Test

public void testGetPurchaseDetails() {

Item expectedDetail = new Item();

Item actItem = new Item();

expectedDetail.setItemId(12);

expectedDetail.setProductId(81);

expectedDetail.setCategoryId(31);

expectedDetail.setItemName("Healthy Biscuits");

expectedDetail.setItemDescription("healthy food for dogs");

expectedDetail.setItemPrice(525);

System.out.println(expectedDetail);

OrderDAO data = new OrderDAO();

List actualList;

try {

actualList = (List) data.getPurchaseDetails(121);

System.out.println(actualList);

Iterator iterate = actualList.iterator();

while (iterate.hasNext()) {

actItem = (Item) iterate.next();

}

assertTrue(expectedDetail.equals(actItem));

} catch (DBFWException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}