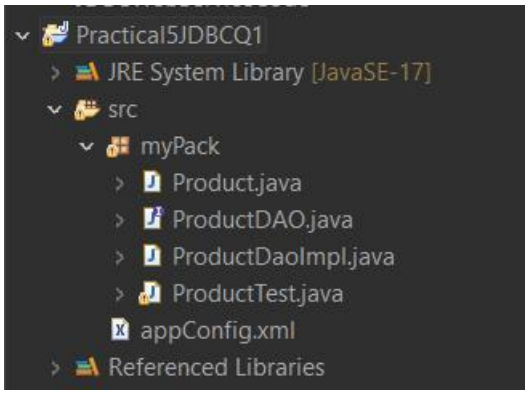
Practical No. 05
JDBC Data Access with Spring using Oracle/MySQL database

Q.2 Create a Spring application that interacts with a database using JdbcTemplate. Implement a DAO (Data Access Object) class that provides CRUD (Create, Read, Update, Delete) operations for a "Product" entity. The Product entity should have the following attributes: - id (int, primary key), name (String), price (double), quantity (int)

Implementation:



Code:

Product.java

```
package myPack;

public class Product {
int id;
String name;
double price;
int quantity;
@Override
public String toString() {
return "Product [id=" + id + ", name=" + name + ", price=" + price + ", quantity=" +
quantity + "]";
}
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public double getPrice() {
return price;
}
public void setPrice(double price) {
this.price = price;
```

```
}
public int getQuantity() {
return quantity;
}
public void setQuantity(int quantity) {
this.quantity = quantity;
}
public Product() {
super();
}
public Product(int id, String name, double price, int quantity) {
super();
this.id = id;
this.name = name;
this.price = price;
this.quantity = quantity;
}
}
```

ProductDAO.java (Interface)

```
package myPack;

import java.util.List;

public interface ProductDAO {
void create(Product product);
Product findById(int id);
List<Product> findAll();
void update(Product product);
void delete(int id);
}
```

ProductDaoImpl.java

```
package myPack;

import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;
public class ProductDaoImpl implements ProductDAO {
private final JdbcTemplate jdbcTemplate;

public ProductDaoImpl(JdbcTemplate jdbcTemplate) {
this.jdbcTemplate = jdbcTemplate;
}

@Override
public void create(Product product) {
String sql = "INSERT INTO product (name, price, quantity) VALUES (?, ?, ?)";
jdbcTemplate.update(sql, product.getName(), product.getPrice(),
product.getQuantity());
}
```

```
@SuppressWarnings("deprecation")
@Override
public Product findById(int id) {
String sql = "SELECT * FROM product WHERE id = ?";
return jdbcTemplate.queryForObject(sql, new Object[]{id}, new ProductMapper());
}

@Override
public List<Product> findAll() {
String sql = "SELECT * FROM product";
return jdbcTemplate.query(sql, new ProductMapper());
}

@Override
public void update(Product product) {
String sql = "UPDATE product SET name = ?, price = ?, quantity = ? WHERE id = ?";
jdbcTemplate.update(sql, product.getName(), product.getPrice(),
product.getQuantity(), product.getId());
}

@Override
public void delete(int id) {
String sql = "DELETE FROM product WHERE id = ?";
jdbcTemplate.update(sql, id);
}

private static class ProductMapper implements RowMapper<Product> {
@Override
public Product mapRow(ResultSet rs, int rowNum) throws SQLException {
Product product = new Product();
product.setId(rs.getInt("id"));
product.setName(rs.getString("name"));
product.setPrice(rs.getDouble("price"));
product.setQuantity(rs.getInt("quantity"));
return product;
}
}
}
```

appConfig.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
<bean id="ds" class="org.apache.commons.dbcp2.BasicDataSource">
<property name="driverClassName"
value="com.mysql.cj.jdbc.Driver"></property>
<property name="url" value="jdbc:mysql://localhost:3306/prodb"></property>
<property name="username" value="root"></property>
<property name="password" value=""></property>
</bean>
<!-- JdbcTemplate Configuration -->
```

```
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
<property name="dataSource" ref="ds" />
</bean>

<!-- ProductDao Configuration -->
<bean id="productDao" class="myPack.ProductDaoImpl">
<constructor-arg ref="jdbcTemplate" />
</bean>
</beans>
```

ProductTest.java

```
package myPack;

import java.util.List;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;


public class ProductTest {
public static void main(String[] args) {
ApplicationContext context = new
ClassPathXmlApplicationContext("appConfig.xml");
ProductDAO productDao = context.getBean("productDao", ProductDAO.class);

// Create a new product
Product newProduct = new Product();
newProduct.setName("Shirdi Mitha");
newProduct.setPrice(1.00);
newProduct.setQuantity(100);
productDao.create(newProduct);
System.out.println("New product created: " + newProduct);

// Find product by ID
int productId = 2;
Product foundProduct = productDao.findById(productId);
System.out.println("Product found by ID: " + foundProduct);

// Update product
foundProduct.setName("Updated Product");
foundProduct.setPrice(2.00);
foundProduct.setQuantity(50);
productDao.update(foundProduct);
System.out.println("Product updated: " + foundProduct);

// Find all products
List<Product> allProducts = productDao.findAll();
System.out.println("All products:");
for (Product product : allProducts) {
System.out.println(product);
}

// Delete product
productDao.delete(productId);
```

```
System.out.println("Product deleted with ID: " + productId);
}
}
```

Output:

```
New product created: Product [id=0, name=Shirdi Mitha, price=1.0, quantity=100]
Product found by ID: Product [id=2, name=Ambani Chaiwala, price=340.99, quantity=75]
Product updated: Product [id=2, name=Updated Product, price=2.0, quantity=50]
All products:
Product [id=2, name=Updated Product, price=2.0, quantity=50]
Product [id=3, name=Bhupendra Jogi Mobile Shop, price=1900.99, quantity=100]
Product [id=7, name=Dattu Sawant Builder, price=1224.99, quantity=50]
Product [id=8, name=Onkar MCA Billionarie Chocolate, price=1224.99, quantity=50]
Product [id=9, name=Ambani Chaiwala, price=340.99, quantity=75]
Product [id=10, name=Bhupendra Jogi Mobile Shop, price=1900.99, quantity=100]
Product [id=11, name=Shirdi Mitha, price=1.0, quantity=100]
Product [id=12, name=Shirdi Mitha, price=1.0, quantity=100]
Product [id=13, name=Shirdi Mitha, price=1.0, quantity=100]
Product deleted with ID: 2
```