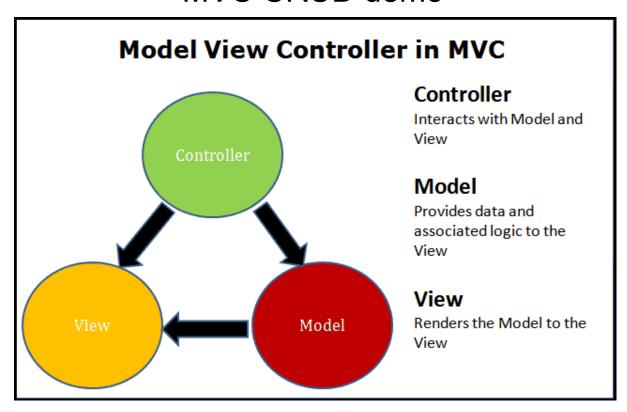
MVC CRUD demo



1.Tạo database với tên là MyDB với Table có tên Products cấu trúc như sau:

```
Connection: jdbc:derby://localhost:1527/myDB [sa on SA]

CREATE TABLE Products

ID int not null generated always as identity (start with 1, increment by 1)

ProName varchar(50),

ProDesc varchar(50),

Price double

Price double
```

2. Tạo Project có cấu trúc như sau:

```
💿 ...-/A 🗗 Product.java [-/A] 🗴 🔯 ProductDao.java [-/A] 🗴 🗗 ProductDaolmpl.java [-/A] 🗴 🗗 MVCcrud.java [-/A] 🗴 🗗 ProductCons
Files Projects Services
 ► $JDBCApps [0000000...]
                                         Source History 🔯 🔯 🔻 🔻 💆 🞝 🗗 📮 🔐 🤡 ڬ 💆 🔘 📗 🏙 🚅
 SoginApp [0000000...
                                               package com.example.entity;
 ▼ ⑤MVCcrud [0000000...]
   ▼ 🛅 Source Packages
                                             日/**
       ▼ Bocom.example.controller
            ProductService.java [-/A]
                                                * @author tuann
      ▼ ⊞<sub>®</sub>com.example.dao

■ DBConnection.java [-/A]

                                               public class Product {
      ▼ ⊞<sub>0</sub>com.example.entity
                                                    private Integer productId;
private String productName;
private String productDesc;
private Double productPrice;
            Product.java [-/A]
      ▼ Egcom.example.model
                                         10
            ProductDao.java [-/A]
ProductDaolmpl.java [-/A]
                                         11
                                         12
      ▼ ⊞<sub>3</sub>com.example.ui
                                         13
                                         14
                                                    public Product() {
            ProductConsole.java [-/A]
      ▼ <u>⊞</u>mvccrud
                                         16
            MVCcrud.java [-/A]
                                                    public Product(String productName, String productDesc, Double productPrice) {
   ► 🛅 Test Packages
                                         18
                                                         this.productName = productName;
this.productDesc = productDesc;

▼ Ightaries

                                         19
      Java DB Driver - derby.jar
                                                         this productPrice = productPrice;
      🕨 🚍 Java DB Driver - derbyclient.jai
                                         21
         Java DB Driver - derbynet.jar
                                         22
         JDK 1.8 (Default)
                                         23 E
24
                                                    public Integer getProductId() {
   Test Libraries
                                                         return productId;
```

3. Product entity(thực thể này sẽ mapping đến table trong CSDL):

```
package com.example.entity;
1
2
3
   日/**
4
5
     * @author tuann
6
7
     public class Product {
8
9
         private Integer productId;
         private String productName;
private String productDesc;
10
11
12
         private Double productPrice;
13
14
          public Product() {
15
16
17
          public Product(String productName, String productDesc, Double productPrice) {
18
              this.productName = productName;
19
              this.productDesc = productDesc;
20
              this.productPrice = productPrice;
21
22
23
          public Integer getProductId() {
24
              return productId;
25
26
          public void setProductId(Integer productId) {
27
28
              this.productId = productId;
30
          public String getProductName() {
31
32
              return productName;
33
34
          public void setProductName(String productName) {
35
   曱
36
              this.productName = productName;
37
38
```

4. DBConnection class(lớp kết nối đến CSDL):

```
1
     package com.example.dao;
2
3
   早/**
 4
 5
     * @author tuann
    L */
 6
 7
   pimport java.sql.Connection;
     import java.sql.DriverManager;
8
     import java.sql.SQLException;
10
     import java.util.logging.Level;
11
     import java.util.logging.Logger;
12
     public class DBConnection {
9
14
15
         public static Connection createConnection() {
             String dbUrl = "jdbc:derby://localhost:1527/myDB";
16
17
             Connection conn = null;
18
             try {
19
                  conn = DriverManager.getConnection(dbUrl, "sa", "sa");
20
             } catch (SQLException ex) {
                 Logger.getLogger(DBConnection.class.getName()).log(Level.SEVERE, null, ex);
21
22
23
             return conn;
24
25
26
     }
27
```

5.ProductDao interface(interface này mô tả các phương thức sẽ triển khai trong lớp Biz)

```
package com.example.model;
 2
 3
   ₽/**
 4
 5
     * @author tuann
    L */
 6
   □ import com.example.entity.Product;
 7
 8
    import java.util.ArrayList;
 9
     public interface ProductDao {
 1
11
         public void createProduct(Product product);
 •
13
         public Product getProductById(int productId);
 1
15
         public ArrayList<Product> getAllProducts();
1
17
         public void updateProduct(Product product);
19
         public boolean deleteProduct(int productId);
₩.
21
22
     }
23
```

6. ProductDaoImpl class(implements các phương thức mô tả trong interface)

```
package com.example.model;
   2
3
4
5
6
7
8
9
       □ /**
           * @author tuann
        pimport com.example.dao.DBConnection;
          import com.example.entity.Product;
import java.sql.Connection;
import java.sql.PreparedStatement;
           import java.sql.ResultSet;
import java.sql.SQLException;
 11
12
           import java.sql.Statement;
import java.util.ArrayList;
 13
14
15
16
17
           import java.util.logging.Level;
import java.util.logging.Logger;
 18
19
           public class ProductDaoImpl implements ProductDao {
                private final Connection conn = DBConnection.createConnection();
private final String SQL_CREATE_PRODUCT = "INSERT INTO products (productName, productDesc,productPrice) VALUES (?, ?, ?)";
private final String SQL_GET_PRODUCT_BY_ID = "SELECT * FROM products WHERE Id=?";
private final String SQL_GET_ALL_PRODUCTS = "SELECT * FROM products";
private final String SQL_UPDATE_PRODUCT = "UPDATE productSET productName=?,
private final String SQL_DELETE_PRODUCT = "DELETE FROM products WHERE ID=?";
 21
22
 23
24
25
 26
27
 ②
29
                  public void createProduct(Product product) {
    try (PreparedStatement pstmt = conn.prepareStatement(SQL_CREATE_PRODUCT,
    Statement.RETURN_GENERATED_KEYS)) {
 30
31
                               pstmt.setString(1, product.getProductName());
pstmt.setString(2, product.getProductDesc());
pstmt.setDouble(3, product.getProductPrice());
pstmt.executeUpdate();
 32
 33
34
                                try (ResultSet generatedKeys = pstmt.getGeneratedKeys()) {
   if (generatedKeys.next()) {
      product.setProductId(generatedKeys.getInt(1));
}
 35
 36
 37
 38
                                       }
39
                       } catch (SQLException ex) {
40
41
                              Logger.getLogger(ProductDaoImpl.class.getName()).log(Level.SEVERE, null, ex);
43
                }
44
45
      ₽
                public Product getProductById(int productId) {
47
                       Product product = new Product();
try (PreparedStatement pstmt = conn.prepareStatement(SQL_GET_PRODUCT_BY_ID)) {
48
49
                               pstmt.setInt(1, productId);
                               try (ResultSet rs = pstmt.executeQuery()) {
   while (rs.next()) {
50
51
52
53
54
55
                                             product.setProductId(rs.getInt(1));
                                            product.setProductName(rs.getString(2));
product.setProductDesc(rs.getString(3));
                                             product.setProductPrice(rs.getDouble(4));
56
57
                                      }
58
                       } catch (SQLException ex) {
                              Logger.getLogger(ProductDaoImpl.class.getName()).log(Level.SEVERE, null, ex);
59
60
61
                        return product;
62
63
```

```
64
           @Override
           public ArrayList<Product> getAllProducts() {
               ArrayList<Product> allProducts = new ArrayList();
 66
               try (PreparedStatement pstmt = conn.prepareStatement(SQL_GET_ALL_PRODUCTS);
 67
 68
                       ResultSet rs = pstmt.executeQuery()) {
 69
                   while (rs.next()) {
 70
                       Product product = new Product();
 71
                       product.setProductId(rs.getInt(1));
 72
                       product.setProductName(rs.getString(2));
 73
                       product.setProductDesc(rs.getString(3));
 74
                       product.setProductPrice(rs.getDouble(4));
 75
                       allProducts.add(product);
 76
 77
               } catch (SQLException ex) {
 78
                   Logger.getLogger(ProductDaoImpl.class.getName()).log(Level.SEVERE, null, ex);
 79
 80
               return allProducts;
 81
 82
 83
          @Override
           public void updateProduct(Product product) {
 85
               try (PreparedStatement pstmt = conn.prepareStatement(SQL_UPDATE_PRODUCT)) {
                   pstmt.setString(1, product.getProductName());
 86
                   pstmt.setString(2, product.getProductDesc());
pstmt.setDouble(3, product.getProductPrice());
 87
 88
 89
                   pstmt.setInt(8, product.getProductId());
 90
                   pstmt.executeUpdate();
 91
               } catch (SQLException ex) {
 92
                   Logger.getLogger(ProductDaoImpl.class.getName()).log(Level.SEVERE, null, ex);
 93
 94
          }
 95
 96
          @Override
           public boolean deleteProduct(int productId) {
               try (PreparedStatement pstmt = conn.prepareStatement(SQL_DELETE_PRODUCT)) {
 98
 99
                   pstmt.setInt(1, productId);
                   pstmt.executeUpdate();
100
101
                   return true;
               } catch (SQLException ex) {
102
103
                   Logger.getLogger(ProductDaoImpl.class.getName()).log(Level.SEVERE, null, ex);
104
105
               return false;
106
107
      }
108
109
```

7. ProductConsole class(Lớp "view" đóng vai trò tương tác với người dùng)

```
1
     package com.example.ui;
2
3
   ₽/**
4
   * @author tuann */
5
6
   pimport com.example.entity.Product;
7
     import java.io.IOException;
     import java.util.Scanner;
    import com.example.controller.ProductService;
10
   import java.util.ArrayList;
11
12
14
   * @author tuann */
15
16
     public class ProductConsole {
17
18
         private final Scanner sc;
19
         ProductService productService = new ProductService();
20
21
22
   巨
         public ProductConsole() {
             this.sc = new Scanner(System.in);
23
24
25
         private int menu() {
26
   曱
             System.out.println("---PRODUCT MENU---");
27
             System.out.println("1. Add product");
28
             System.out.println("2. Show all product");
29
30
             System.out.println("3. Remove product");
31
             System.out.println("0. Exit");
32
             int choice = readInt(0, 3);
33
             return choice;
34
35
```

```
public void start() {
               while (true) {
37
38
                    int choice = menu();
                   switch (choice) {
39
                        case 0:
40
41
                            System.exit(0);
42
                            break;
43
                        case 1:
44
                            addProduct();
45
                            break;
46
                        case 2:
                            showAll();
47
48
                            break;
49
                        case 3:
50
                            removeProduct();
51
                            break;
52
                        default:
53
                            throw new AssertionError();
54
55
               }
56
57
58
    巨
           private int readInt(int min, int max) {
               int choice;
59
               while (true) {
60
61
                   try {
                        choice = Integer.parseInt(sc.nextLine());
62
                        if (choice >= min && choice <= max) {</pre>
63
64
                            break;
65
                   } catch (NumberFormatException e) {
66
67
68
69
               return choice;
70
71
 72
          private double readDouble(int min, double max) {
 73
              double price;
              while (true) {
 74
 75
                  try {
                      price = Double.parseDouble(sc.nextLine());
 76
 77
                       if (price >= min && price <= max) {</pre>
 78
                          break;
 79
                  } catch (NumberFormatException e) {
 80
                      System.out.println("Invalid value. Try to enter a float value");
 81
 82
 83
 84
              return price;
 85
 86
 87
          private void addProduct() {
 88
              System.out.println("Enter product name: ");
 89
 90
              String name = sc.nextLine();
 91
              System.out.println("Enter product desc: ");
 92
              String desc = sc.nextLine();
              System.out.println("Enter product price: ");
 93
              double price = readDouble(0, Double.MAX_VALUE);
 94
              Product product = new Product(name, desc, price);
 95
 96
              productService.createProduct(product);
 97
 98
99
    早
          private void showAll() {
              System.out.println("--All product--");
100
              System.out.println("ID\tName\tDesc\tPrice");
101
102
              ArrayList<Product> allProducts = productService.getAllProducts();
103
104
              allProducts.forEach((product) -> {
                  System.out.println(product.getProductId() + "\t" + product.getProductName() + "\t"
105
                          + product.getProductDesc() + "\t" + product.getProductPrice());
106
              });
107
108
          }
109
```

```
110 👨
          private void removeProduct() {
111
              System.out.println("Enter ID of product");
              int id = readInt(0, Integer.MAX_VALUE);
112
              boolean result = productService.deleteProduct(id);
113
              if (result) {
114
                  System.out.println("Product was removed");
115
              } else {
116
                  System.out.println("Product not found");
117
118
119
      }
120
121
```

8. MVCcrud -> run application

```
package mvccrud;
 1
 2
   import java.io.IOException;
import com.example.ui.ProductConsole;
 3
 4
 5
 6
   ₽/**
 7
 8
     * @author tuann
 9
     public class MVCcrud {
10
11
12
          * @param args the command line arguments
13
14
           * @throws java.io.IOException
15
          public static void main(String[] args) throws IOException {
16
              ProductConsole pc = new ProductConsole();
17
              pc.start();
18
19
20
     }
21
22
```