NPM : 4510210039

Nama : Febri Damatraseta Fairuz

Materi : Spring MVC Validasi

Praktikum 6

Code

1. Menambahkan dependensi `hibernate-validator` pada pom.xml

```
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>4.3.1.Final</version>
</dependency>
```

Gambar 1. Dependensi hibernate-validator

2. Menggunakan annotation untuk validasi di domain object
   - @NotNull : tidak boleh null
   - @NotEmpty : mencegah string kosong "", " "
   - @Size : membatasi panjang string
   - @Min dan @Max : membatasi nilai minimum dan maksimum
   - @Past : tanggal harus sebelum waktu sekarang

```
private Integer id;

@NotNull
@NotEmpty
@Size(min= 1, max = 5)
private String kode;

@NotNull @NotEmpty
@Size(min = 5, max = 255)
private String nama;

@Min(100)
private BigDecimal harga;

private Date terakhirUpdate;
```

Gambar 2. Menambahkan annotation untuk validasi pada Produk.java

3. Mengaktifkan validasi di controller dan mengimplementasikan edit dan hapus produk.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
        xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd">

    <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource">
        <property name="driverClassName" value="com.mysql.jdbc.Driver" />
        <property name="url" value="jdbc:mysql://localhost/tokobagus" />
        <property name="username" value="root" />
        <property name="password" value="" />
    </bean>

    <context:component-scan base-package="com.training.springmvc" />
</beans>
```

Gambar 3. Membuat file koneksi ke database dan menampilkan record pada database

File ProdukDao.java

```
package com.training.springmvc;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.sql.DataSource;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;


@Repository
public class ProdukDao {

    @Autowired private DataSource dataSource;

    private String sqlInsert = "insert into produk (kode, nama,
harga, terakhir_update) " + "values (?,?,?,?)";
    private String sqlUpdate = "update produk set kode=?, nama=?,
harga=?, terakhir_update=? " + "where id = ?";
    private String sqlCariSemuaProduk = "select * from produk
order by kode";
    private String sqlCariById = "select * from produk where id =
?";
    private String sqlHapusById = "delete from produk where id =
?";

    public void simpan(Produk p) throws Exception {
        Connection c = dataSource.getConnection();

        if (p.getId() == null) {
```

```
            PreparedStatement              psInsert              =
c.prepareStatement(sqlInsert);
            psInsert.setString(1, p.getKode());
            psInsert.setString(2, p.getNama());
            psInsert.setBigDecimal(3, p.getHarga());
            psInsert.setDate(4,                                   new
java.sql.Date(p.getTerakhirUpdate().getTime()));

            psInsert.executeUpdate();
        } else {
            PreparedStatement              psUpdate              =
c.prepareStatement(sqlUpdate);
            psUpdate.setString(1, p.getKode());
            psUpdate.setString(2, p.getNama());
            psUpdate.setBigDecimal(3, p.getHarga());
            psUpdate.setDate(4,                                   new
java.sql.Date(p.getTerakhirUpdate().getTime()));
            psUpdate.setInt(5, p.getId());

            psUpdate.executeUpdate();
        }
        c.close();
    }

    public List<Produk> cariSemuaProduk() throws Exception {
        List<Produk> hasil = new ArrayList<Produk>();

        Connection c = dataSource.getConnection();
        PreparedStatement           psCariSemuaProduk           =
c.prepareStatement(sqlCariSemuaProduk);

        ResultSet rs = psCariSemuaProduk.executeQuery();
        while(rs.next()){
            Produk p = konversiResultSetKeProduk(rs);
            hasil.add(p);
        }

        c.close();
        return hasil;
    }

    public Produk cariById(Integer id) throws Exception{
        if(id == null){
            return null;
        }
        Connection c = dataSource.getConnection();
        PreparedStatement               psCariById              =
c.prepareStatement(sqlCariById);
        psCariById.setInt(1, id);
        ResultSet rs = psCariById.executeQuery();
        if(!rs.next()){
```

```
            return null;
        }
        Produk p = konversiResultSetKeProduk(rs);
        c.close();
        return p;
    }

    public Produk konversiResultSetKeProduk(ResultSet rs) throws
SQLException{
        Produk p = new Produk();
        p.setId(rs.getInt("id"));
        p.setKode(rs.getString("kode"));
        p.setNama(rs.getString("nama"));
        p.setHarga(rs.getBigDecimal("harga"));
        p.setTerakhirUpdate(rs.getDate("terakhir_update"));
        return p;
    }

    public void hapus(Integer id) throws Exception{
        if(id == null){
            return;
        }
        Connection c = dataSource.getConnection();
        PreparedStatement              psHapusById              =
c.prepareStatement(sqlHapusById);
        psHapusById.setInt(1, id);
        psHapusById.executeUpdate();
        c.close();
    }
}
```

4. Menampilkan pesan error bila validasi gagal

```
<spring:form modelAttribute="produk">
    <table border="1">
        <tbody>
            <tr>
                <td>Kode</td>
                <td>
                    <spring:input path="kode"/>
                </td>
                <td>
                    <spring:errors path="kode" />
                </td>
            </tr>
            <tr>
                <td>Nama</td>
                <td>
                    <spring:input path="nama"/>
                </td>
                <td>
                    <spring:errors path="nama" />
                </td>
            </tr>
            <tr>
                <td>Harga</td>
                <td>
                    <spring:input path="harga"/>
                </td>
                <td>
                    <spring:errors path="harga" />
                </td>
            </tr>
            <tr>
                <td colspan="3"> <input type="submit" value="Simpan"> </td>
            </tr>
        </tbody>
    </table>
</spring:form>
```

Gambar 4. Menampilkan pesan gagal atau validasi

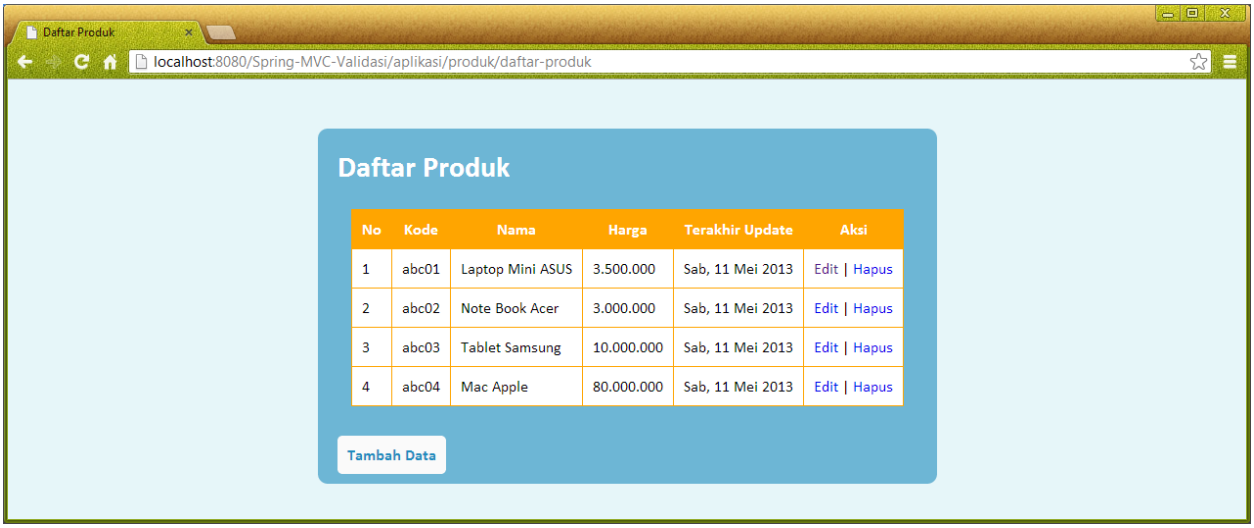
Display

Gambar 5. Daftar produk

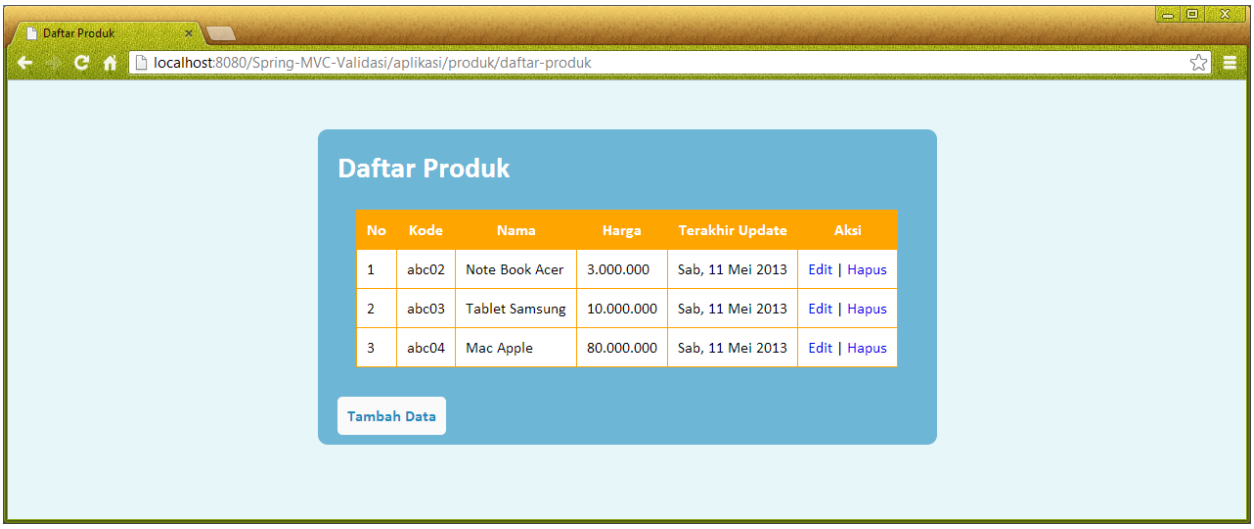
Gambar 6. Menampilkan validasi

Gambar 7. Menampilkan daftar produk

Gambar 8. Melakukan proses edit produk Asus



Gambar 9. Hasil edit berhasil



Gambar 10. Menghapus record produk Asus