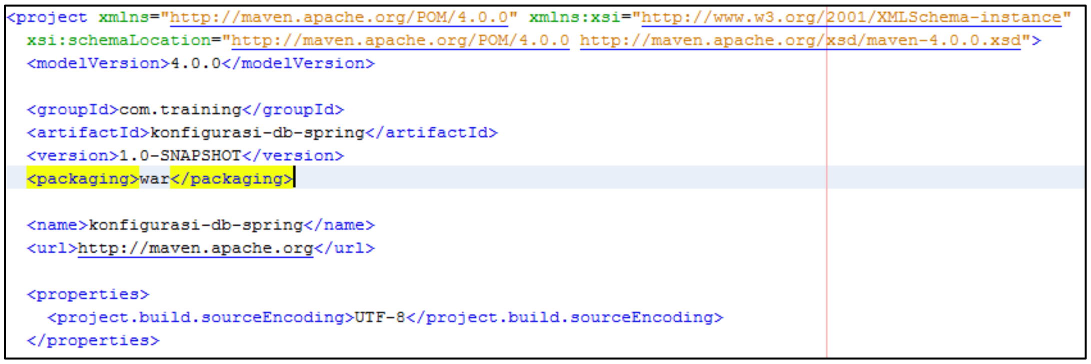
Konfigurasi Database Spring Framework

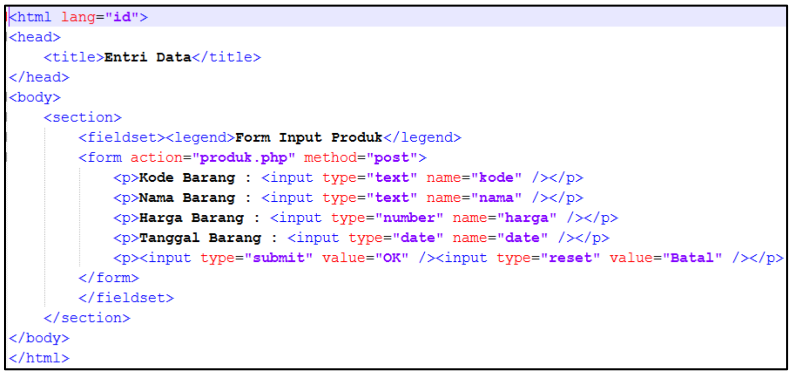
1. Konversi project menjadi aplikasi web



```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.training</groupId>
  <artifactId>konfigurasi-db-spring</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>

  <name>konfigurasi-db-spring</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>
```

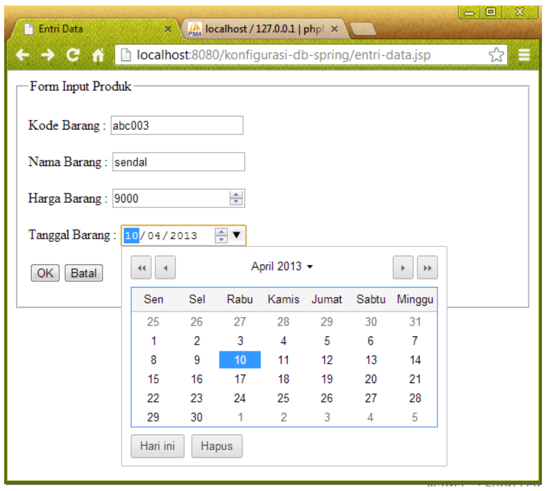
Gambar 1. Mengkonversi packaging

2. Buatlah form JSP untuk entri data produk



```html
<html lang="id">
<head>
    <title>Entri Data</title>
</head>
<body>
    <section>
        <fieldset><legend>Form Input Produk</legend>
        <form action="produk.php" method="post">
            <p>Kode Barang : <input type="text" name="kode" /></p>
            <p>Nama Barang : <input type="text" name="nama" /></p>
            <p>Harga Barang : <input type="number" name="harga" /></p>
            <p>Tanggal Barang : <input type="date" name="date" /></p>
            <p><input type="submit" value="OK" /><input type="reset" value="Batal" /></p>
        </form>
        </fieldset>
    </section>
</body>
</html>
```

Gambar 2. Script file entri-data.jsp



Entri Data
localhost / 127.0.0.1 | php
localhost:8080/konfigurasi-db-spring/entri-data.jsp

Form Input Produk

Kode Barang : abc003

Nama Barang : sendal

Harga Barang : 9000

Tanggal Barang : 10/04/2013

OK Batal

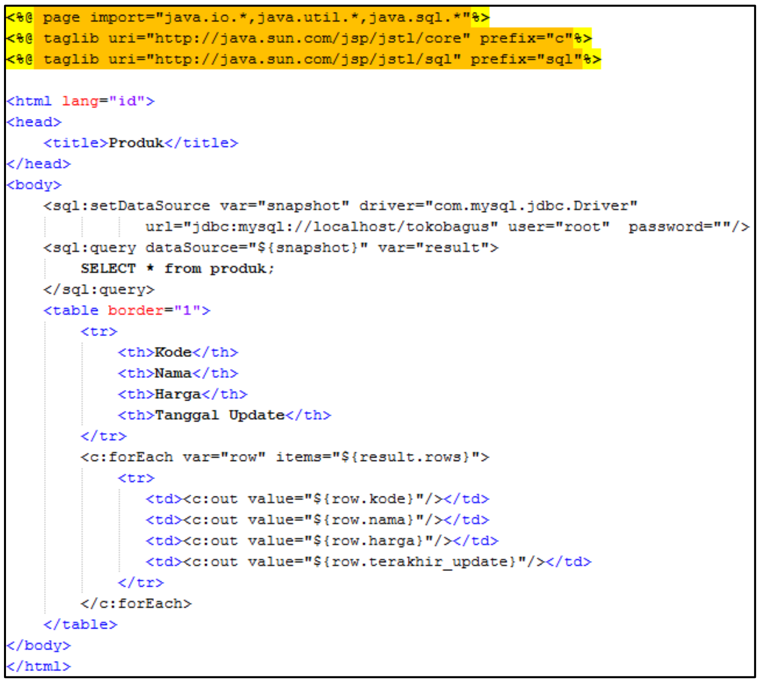
April 2013

Sen	Sel	Rabu	Kamis	Jumat	Sabtu	Minggu
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

Hari ini Hapus

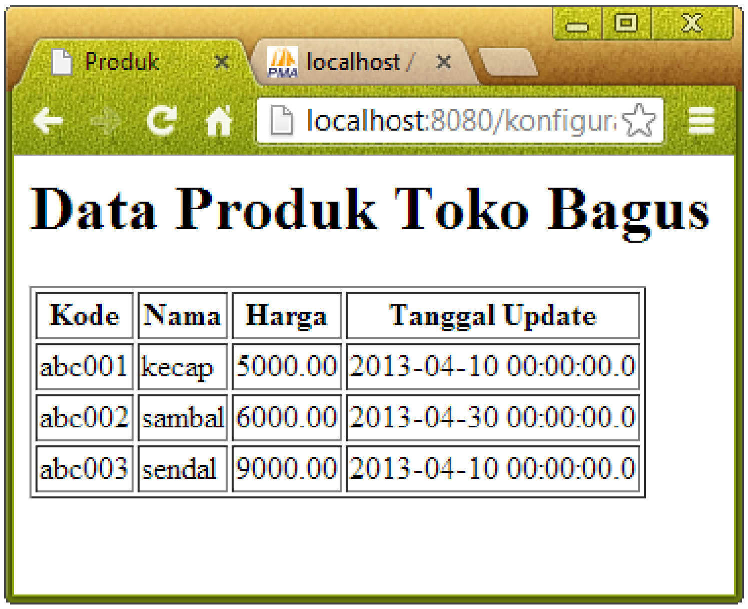
Gambar 3. Tampilan file entri-data.jsp

3. Buatlah halaman JSP untuk menampilkan data produk



```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>

<html lang="id">
<head>
    <title>Produk</title>
</head>
<body>
    <sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
                url="jdbc:mysql://localhost/tokobagus" user="root"  password=""/>
    <sql:query dataSource="${snapshot}" var="result">
        SELECT * from produk;
    </sql:query>
    <table border="1">
        <tr>
            <th>Kode</th>
            <th>Nama</th>
            <th>Harga</th>
            <th>Tanggal Update</th>
        </tr>
        <c:forEach var="row" items="${result.rows}">
            <tr>
               <td><c:out value="${row.kode}"/></td>
               <td><c:out value="${row.nama}"/></td>
               <td><c:out value="${row.harga}"/></td>
               <td><c:out value="${row.terakhir_update}"/></td>
            </tr>
        </c:forEach>
    </table>
</body>
</html>
```

Gambar 4. Script file data-entri.jsp

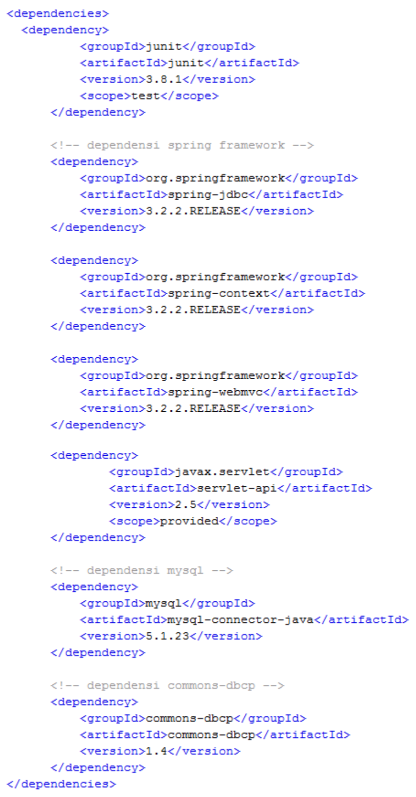


Data Produk Toko Bagus

Kode	Nama	Harga	Tanggal Update
abc001	kecap	5000.00	2013-04-10 00:00:00.0
abc002	sambal	6000.00	2013-04-30 00:00:00.0
abc003	sendal	9000.00	2013-04-10 00:00:00.0

Gambar 5. Tampilan file data-entri.jsp

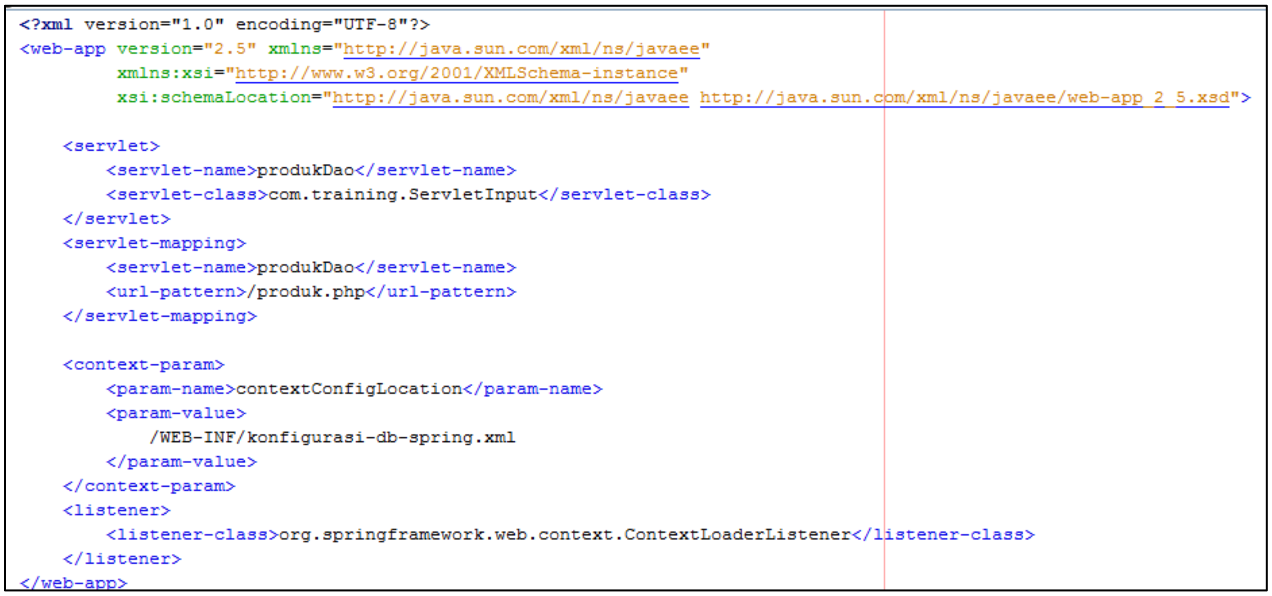
4. Tambahkan dependensi Spring MVC dan dependensi JSTL



```
<dependencies>
  <dependency>
          <groupId>junit</groupId>
          <artifactId>junit</artifactId>
          <version>3.8.1</version>
          <scope>test</scope>
      </dependency>

      <!-- dependensi spring framework -->
      <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-jdbc</artifactId>
          <version>3.2.2.RELEASE</version>
      </dependency>

      <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-context</artifactId>
          <version>3.2.2.RELEASE</version>
      </dependency>

      <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-webmvc</artifactId>
          <version>3.2.2.RELEASE</version>
      </dependency>

      <dependency>
              <groupId>javax.servlet</groupId>
              <artifactId>servlet-api</artifactId>
              <version>2.5</version>
              <scope>provided</scope>
      </dependency>

      <!-- dependensi mysql -->
      <dependency>
          <groupId>mysql</groupId>
          <artifactId>mysql-connector-java</artifactId>
          <version>5.1.23</version>
      </dependency>

      <!-- dependensi commons-dbcp -->
      <dependency>
          <groupId>commons-dbcp</groupId>
          <artifactId>commons-dbcp</artifactId>
          <version>1.4</version>
      </dependency>
</dependencies>
```

Gambar 6. Menambahkan dependencies

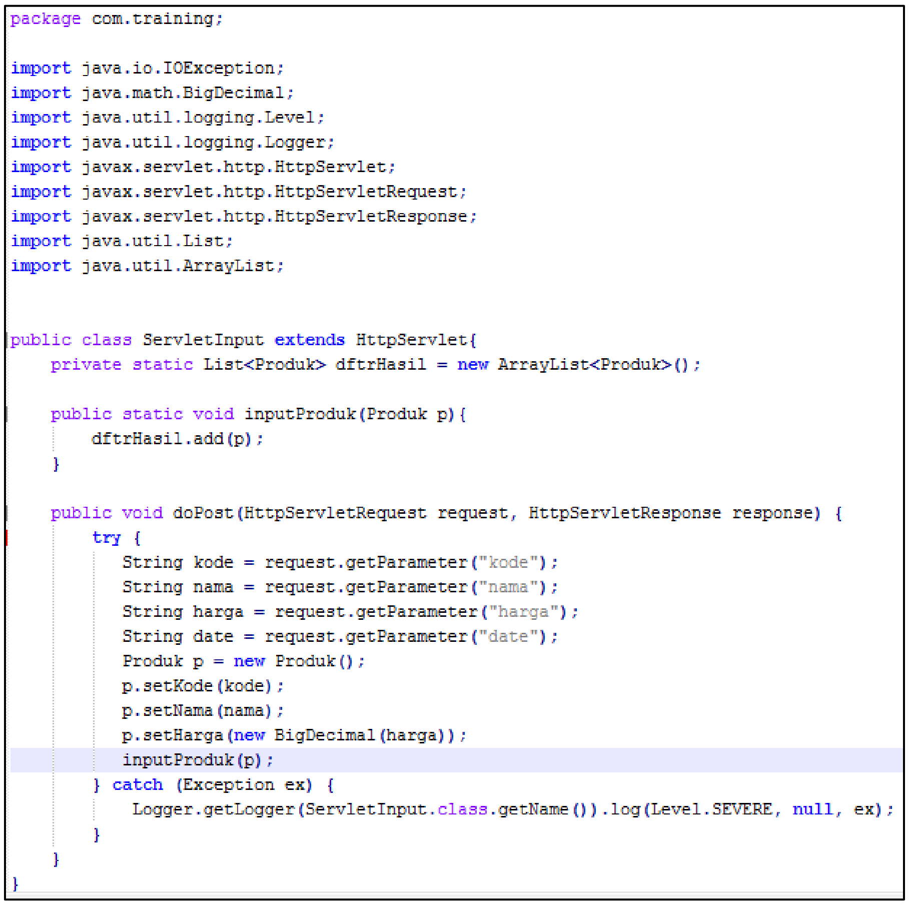
5. Inisialisasi konfigurasi Spring di web.xml



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

    <servlet>
        <servlet-name>produkDao</servlet-name>
        <servlet-class>com.training.ServletInput</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>produkDao</servlet-name>
        <url-pattern>/produk.php</url-pattern>
    </servlet-mapping>

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
            /WEB-INF/konfigurasi-db-spring.xml
        </param-value>
    </context-param>
    <listener>
        <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>
</web-app>
```

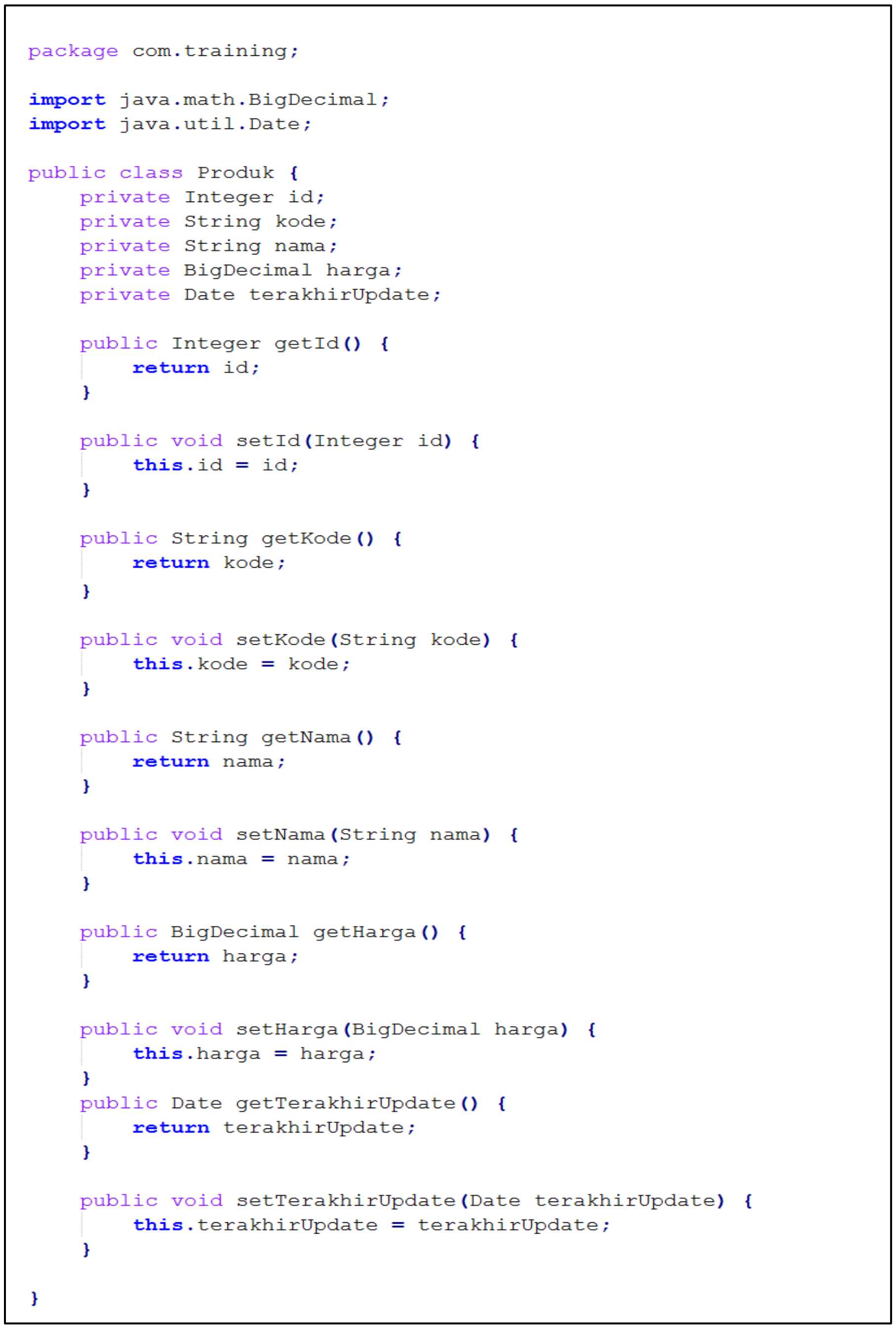
Gambar 7. Inisialisasi spring pada web.xml

6. Buatlah servlet untuk melakukan input record ke database dan menampilkan record dari database ke file JSP



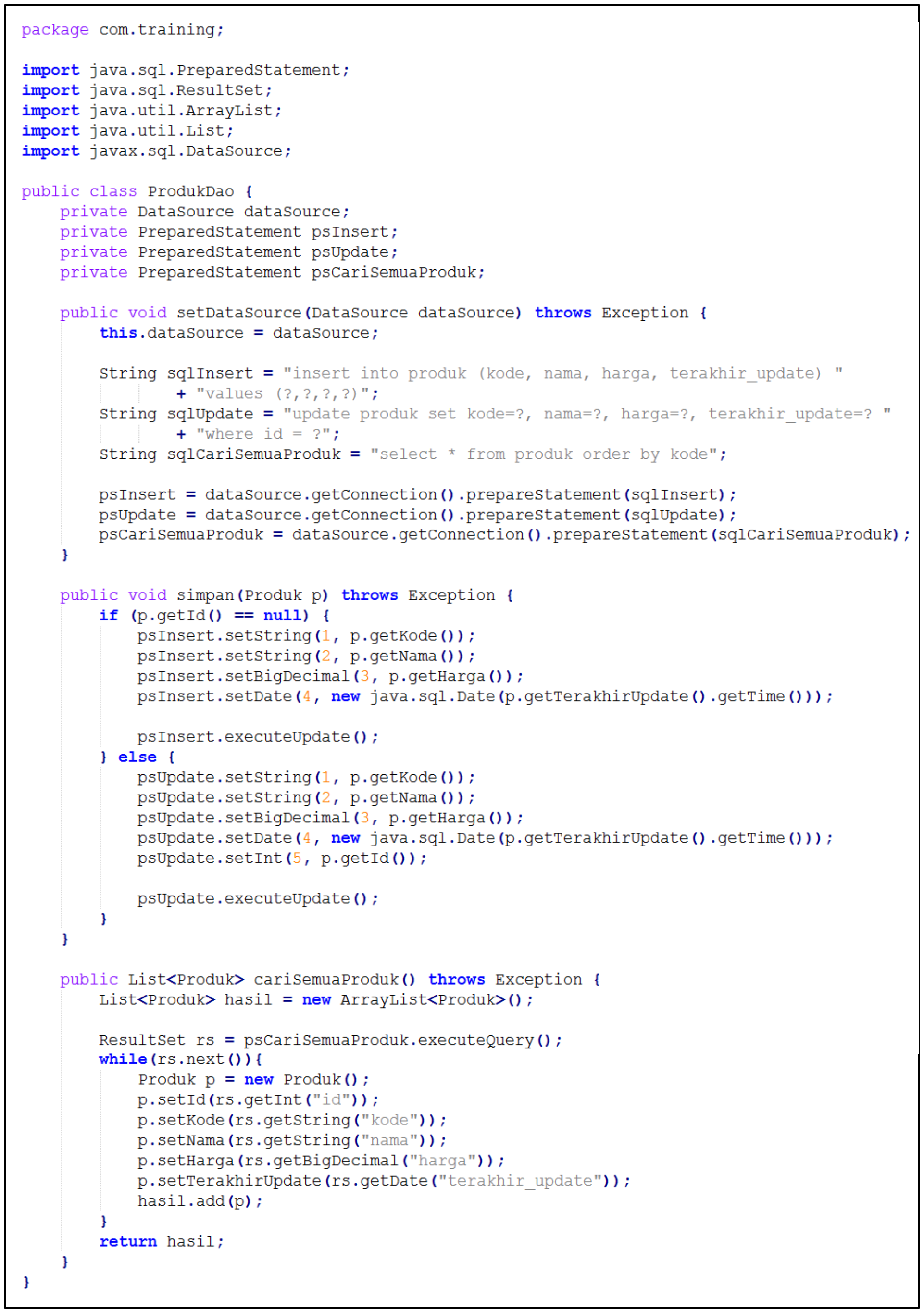
```java
package com.training;

import java.io.IOException;
import java.math.BigDecimal;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.util.List;
import java.util.ArrayList;


public class ServletInput extends HttpServlet{
    private static List<Produk> dftrHasil = new ArrayList<Produk>();

    public static void inputProduk(Produk p){
        dftrHasil.add(p);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response) {
        try {
           String kode = request.getParameter("kode");
           String nama = request.getParameter("nama");
           String harga = request.getParameter("harga");
           String date = request.getParameter("date");
           Produk p = new Produk();
           p.setKode(kode);
           p.setNama(nama);
           p.setHarga(new BigDecimal(harga));
           inputProduk(p);
        } catch (Exception ex) {
            Logger.getLogger(ServletInput.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Gambar 8. ServletInput.java



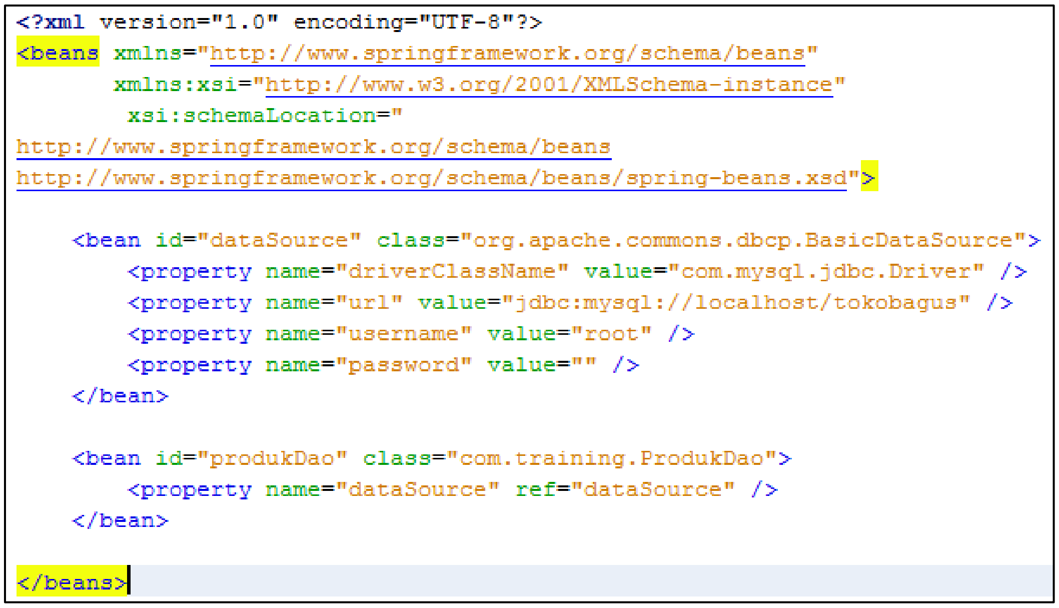
```
package com.training;

import java.math.BigDecimal;
import java.util.Date;

public class Produk {
    private Integer id;
    private String kode;
    private String nama;
    private BigDecimal harga;
    private Date terakhirUpdate;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getKode() {
        return kode;
    }

    public void setKode(String kode) {
        this.kode = kode;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public BigDecimal getHarga() {
        return harga;
    }

    public void setHarga(BigDecimal harga) {
        this.harga = harga;
    }
    public Date getTerakhirUpdate() {
        return terakhirUpdate;
    }

    public void setTerakhirUpdate(Date terakhirUpdate) {
        this.terakhirUpdate = terakhirUpdate;
    }

}
```

Gambar 9. Script Produk.java



```java
package com.training;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;
import javax.sql.DataSource;

public class ProdukDao {
    private DataSource dataSource;
    private PreparedStatement psInsert;
    private PreparedStatement psUpdate;
    private PreparedStatement psCariSemuaProduk;

    public void setDataSource(DataSource dataSource) throws Exception {
        this.dataSource = dataSource;

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

        psInsert = dataSource.getConnection().prepareStatement(sqlInsert);
        psUpdate = dataSource.getConnection().prepareStatement(sqlUpdate);
        psCariSemuaProduk = dataSource.getConnection().prepareStatement(sqlCariSemuaProduk);
    }

    public void simpan(Produk p) throws Exception {
        if (p.getId() == null) {
            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 {
            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();
        }
    }

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

        ResultSet rs = psCariSemuaProduk.executeQuery();
        while(rs.next()){
            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"));
            hasil.add(p);
        }
        return hasil;
    }
}
```

Gambar 10. Script ProdukDao.java



```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="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>

    <bean id="produkDao" class="com.training.ProdukDao">
        <property name="dataSource" ref="dataSource" />
    </bean>

</beans>
```

Gambar 11. Konfigurasi-db-spring.xml