JDBC Framework

Step 1: Creating pom.xml for including the required components

```
<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/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>SpringSetup</groupId>
  <artifactId>SpringSetup</artifactId>
  <packaging>war</packaging>
  <version>0.0.1-SNAPSHOT</version>
  <name>SpringSetup Maven Webapp</name>
  <url>http://maven.apache.org</url>

   <properties>

<org.springframework.version>5.1.6.RELEASE</org.springframework.version>
  </properties>

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


    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-core</artifactId>
      <version>${org.springframework.version}</version>
```

```
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-expression</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-beans</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context-support</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-orm</artifactId>
  <version>${org.springframework.version}</version>
</dependency>
```

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-web</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-webmvc</artifactId>
  <version>${org.springframework.version}</version>
</dependency>

<dependency>
   <groupId>log4j</groupId>
   <artifactId>log4j</artifactId>
   <version>1.2.16</version>
   <scope>runtime</scope>
 </dependency>


<dependency>
    <groupId>javassist</groupId>
    <artifactId>javassist</artifactId>
    <version>3.12.1.GA</version>
</dependency>



<dependency>
    <groupId>taglibs</groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
    <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>commons-dbcp</groupId>
  <artifactId>commons-dbcp</artifactId>
```

```
      <version>1.4</version>
    </dependency>

   <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.9</version>
    </dependency>

<dependency>
<groupId>javax.servlet</groupId>
<artifactId>jstl</artifactId>
<version>1.2</version>
</dependency>

<dependency>
<groupId>javax.servlet</groupId>
<artifactId>jstl</artifactId>
 <version>1.2</version>
</dependency>

<dependency>
<groupId>javax.persistence</groupId>
<artifactId>persistence-api</artifactId>
<version>1.0.2</version>
<scope>provided</scope>
 </dependency>


  </dependencies>
  <build>
    <finalName>SpringSetup</finalName>
  </build>
</project>
```

Step 2: Creating a table eproduct in the database and filling it with sample data

```
--
-- Table structure for table `eproduct`
--

DROP TABLE IF EXISTS `eproduct`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `eproduct` (
  `ID` bigint(20) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `date_added` timestamp NOT NULL DEFAULT
CURRENT_TIMESTAMP,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT
CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `eproduct`
--

LOCK TABLES `eproduct` WRITE;
/*!40000 ALTER TABLE `eproduct` DISABLE KEYS */;
INSERT INTO `eproduct` VALUES (1,'HP Laptop
ABC',21900.00,'2019-06-04 07:18:57'),(2,'Acer Laptop
ABC',23300.00,'2019-06-04 07:19:07'),(3,'Lenovo Laptop
ABC',33322.00,'2019-06-04 07:19:19');
/*!40000 ALTER TABLE `eproduct` ENABLE KEYS */;
UNLOCK TABLES;
```

Step 3: Creating an Entity class EProduct

```
package com.ecommerce.entity;

import java.math.BigDecimal;
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;


public class EProductEntity {

	private long ID;
	private String name;
	private BigDecimal price;
	private Date dateAdded;

	public long getID() {return this.ID; }
	public String getName() { return this.name;}
	public BigDecimal getPrice() { return this.price;}
	public Date getDateAdded() { return this.dateAdded;}

	public void setID(long id) { this.ID = id;}
	public void setName(String name) { this.name = name;}
	public void setPrice(BigDecimal price) { this.price = price;}
	public void setDateAdded(Date date) { this.dateAdded = date;}
}
```

Step 4: Creating a DAO class EProductDAO

```
package com.ecommerce.dao;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import com.ecommerce.entity.EProduct;

public class EProductDAO {

    JdbcTemplate template;

    public void setTemplate(JdbcTemplate template) {
      this.template = template;
    }

    public List<EProduct> getProducts(){
      return template.query("select * from eproduct",new
RowMapper<EProduct>(){
        public EProduct mapRow(ResultSet rs, int row) throws
SQLException {
            EProduct e=new EProduct();
          e.setId(rs.getInt(1));
          e.setName(rs.getString(2));
          e.setPrice(rs.getBigDecimal(3));
          e.setDateAdded(rs.getDate(4));
          return e;
        }
      });
    }
}
```

Step 5: Creating a Controller class MainController

```
package com.ecommerce.controller;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.support.ClassPathXmlApplicationConte
xt;

import com.ecommerce.entity.EProduct;
import com.ecommerce.dao.EProductDAO;

@Controller
public class MainController {



        @Autowired
         EProductDAO eproductDAO;
        @RequestMapping(value = "/listProducts", method =
RequestMethod.GET)
         public String listProducts(ModelMap map)
         {
               List<EProduct> list= eproductDAO.getProducts();
            model.addAttribute("list",list);
            return "listProducts";
         }
        }
```

Step 6: Creating the dispatcher servlet main-servlet.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"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:jee="http://www.springframework.org/schema/jee"
    xmlns:lang="http://www.springframework.org/schema/lang"
    xmlns:p="http://www.springframework.org/schema/p"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xmlns:util="http://www.springframework.org/schema/util"

xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-2.5.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-
context-2.5.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-2.5.xsd">
    <context:annotation-config />
    <context:component-scan base-
package="com.ecommerce.controller" />
    <bean id="jspViewResolver"

class="org.springframework.web.servlet.view.InternalResourceViewR
esolver">
        <property name="viewClass"
            value="org.springframework.web.servlet.view.JstlView">
</property>
        <property name="prefix" value="/WEB-INF/view/"></property>
        <property name="suffix" value=".jsp"></property>
    </bean>
    <bean id="messageSource"

class="org.springframework.context.support.ReloadableResourceBu
```

```
ndleMessageSource">
        <property name="basename" value="classpath:messages">
</property>
        <property name="defaultEncoding" value="UTF-8"></property>
    </bean>

    <bean id="dataSource"
        class="org.apache.commons.dbcp.BasicDataSource" destroy-
method="close"
        p:driverClassName="${jdbc.driverClassName}"
        p:url="${jdbc.databaseurl}" p:username="${jdbc.username}"
        p:password="${jdbc.password}"></bean>



    <bean id="ds"
class="org.springframework.jdbc.datasource.DriverManagerDataSou
rce">
                <property name="driverClassName"
value="com.mysql.jdbc.Driver"></property>
                <property name="url"
value="jdbc:mysql://localhost:3306/ecommerce"></property>
                <property name="username" value="root"></property>
                <property name="password" value="master"></property>
        </bean>

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

        <bean id="dao" class="com.ecommerce.dao.EProductDAO">
                <property name="template" ref="jt"></property>
        </bean>


    <tx:annotation-driven />
    <bean id="transactionManager"
```

```
class="org.springframework.orm.hibernate3.HibernateTransactionMa
nager">
        <property name="sessionFactory" ref="sessionFactory">
</property>
    </bean>
</beans>
```

Step 7: Creating a view index.jsp

```
<html>
<body>
<h2>Spring Application</h2>

<a href="listProducts">List Products</a>
</body>
</html>
```

Step 8: Creating a view listProducts.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
   <%@ taglib uri="http://www.springframework.org/tags/form"
prefix="form"%>
   <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>List Products</title>
</head>
<body>
```

```
 <c:forEach var="product" items="${list}">
   ${product.name}, ${product.price}, ${product.dateAdded}<br>
 </c:forEach>

</body>
</html>
```

Step 9: Configuring web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
version="2.5"
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">

  <display-name>Archetype Created Web Application</display-
name>

    <welcome-file-list>
        <welcome-file>/WEB-INF/view/index.jsp</welcome-file>
    </welcome-file-list>
    <servlet>
        <servlet-name>main</servlet-name>
        <servlet-class>
            org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>main</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/main-servlet.xml</param-value>
    </context-param>
    <listener>
        <listener-class>
org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>
</web-app>
```