# Web application with Mule

By Anirban Sen Chowdhary

There are often question raised on deploying and running Web application in Mule.

People often ask “*is it possible to deploy and run Web application in Mule* “ ??

And the answer is **Yes**

Before we start how, I like to have a overview on the similarities and difference between **Mule server** and **a Web application server**

**Similarities**

- They both allow you to run **multiple applications** simultaneously.
- They both provide an **application container.** What that means is, they both provide an environment in which an application can run, acting as an intermediary between application code and the operating system, and providing database access, easier communication over the network, memory management, lifecycle management, and other services.
- They both allow us to **manage** your applications at runtime.

**Source : MuleSoft**

**Differences**

Mule specializes in three things:-

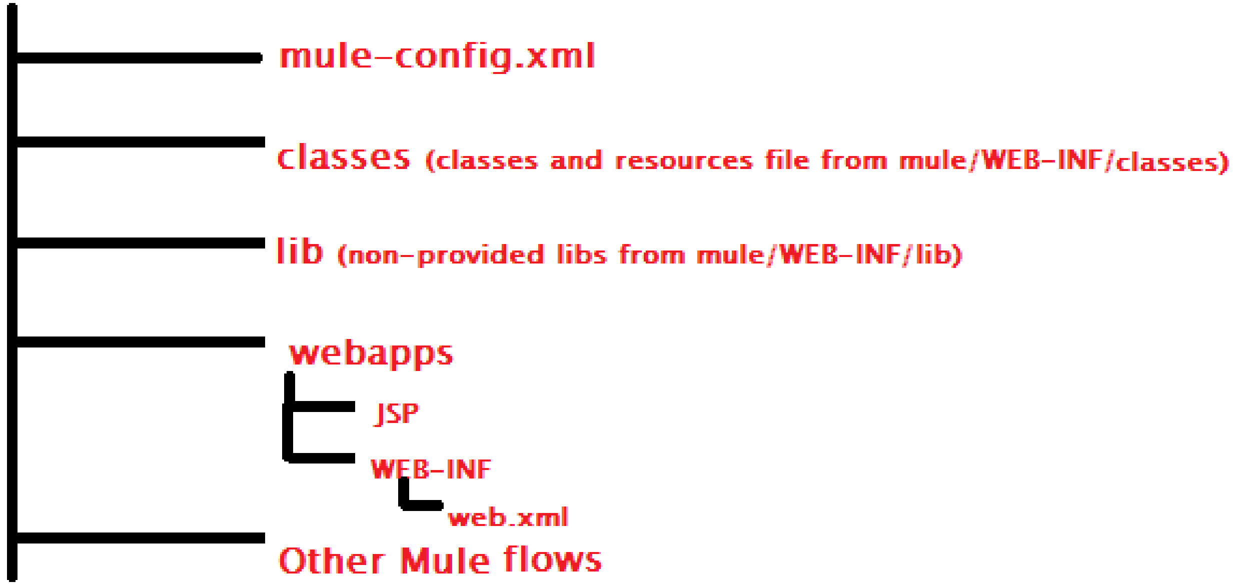
- It acts as a **platform for applications** that move data from one place to another and often transform that data along the way so that it is readable at the other end
- It **exports services** to other applications
- It provides **orchestrating services**

**Source : MuleSoft**

Now, the question is it possible to deploy a **Web application** in **Mule server** which can **interact** or **integrate** with **Mule application** running on the server ???

Yes .. It is possible …… Let’s see how ….

Mule can run JavaEE web-apps thanks to its embedded **Jetty container**. Let us consider our web-app is **mule.war**, the layout we want in our Mule application Zip is



mule-config.xml

classes (classes and resources file from mule/WEB-INF/classes)

lib (non-provided libs from mule/WEB-INF/lib)

webapps

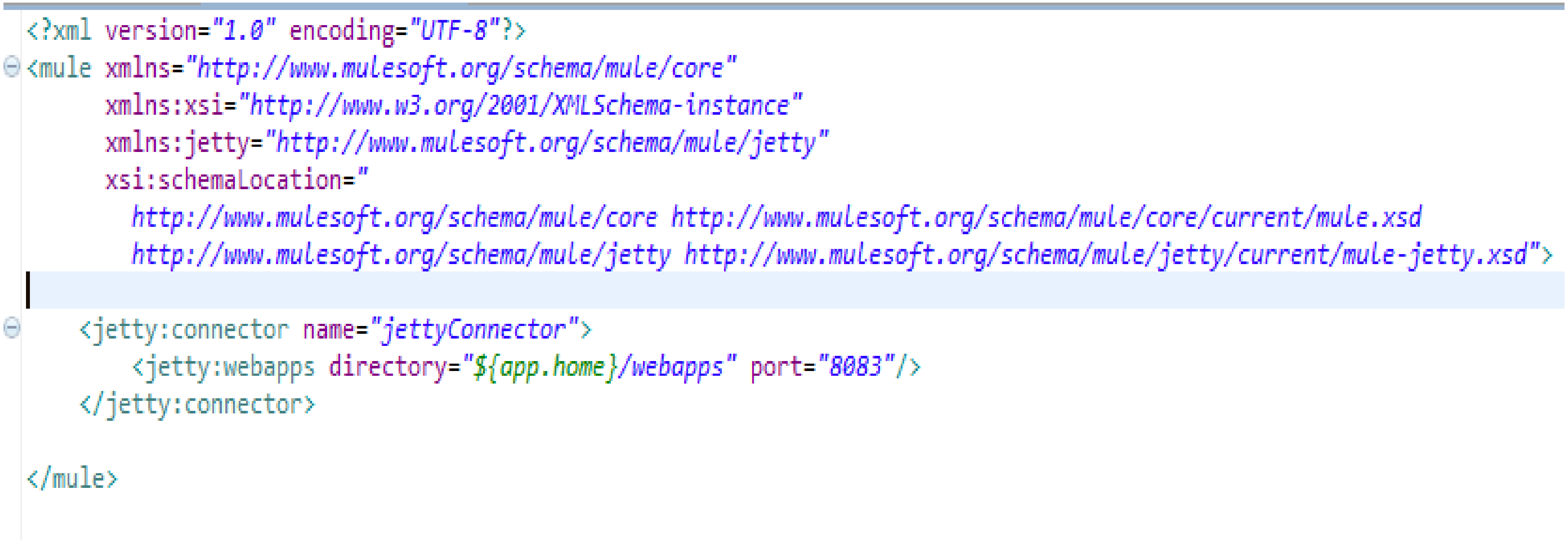
JSP

WEB-INF

web.xml

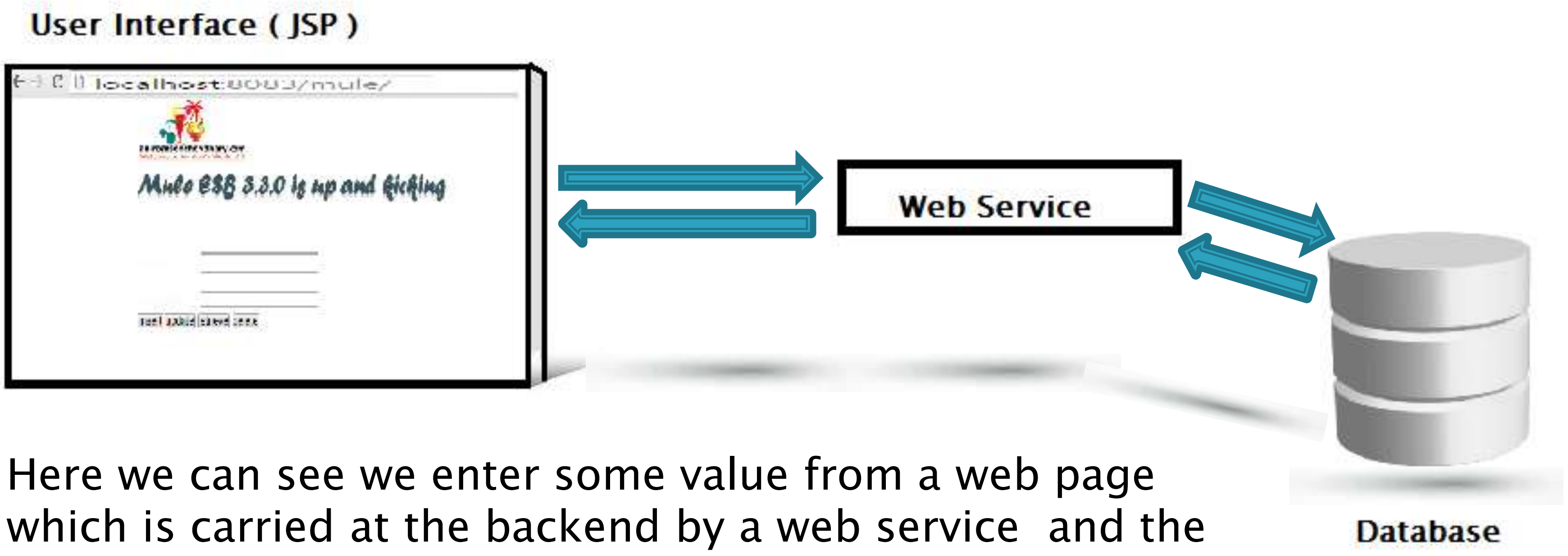
Other Mule flows

And our **mule-config.xml** will host the **jetty server**:



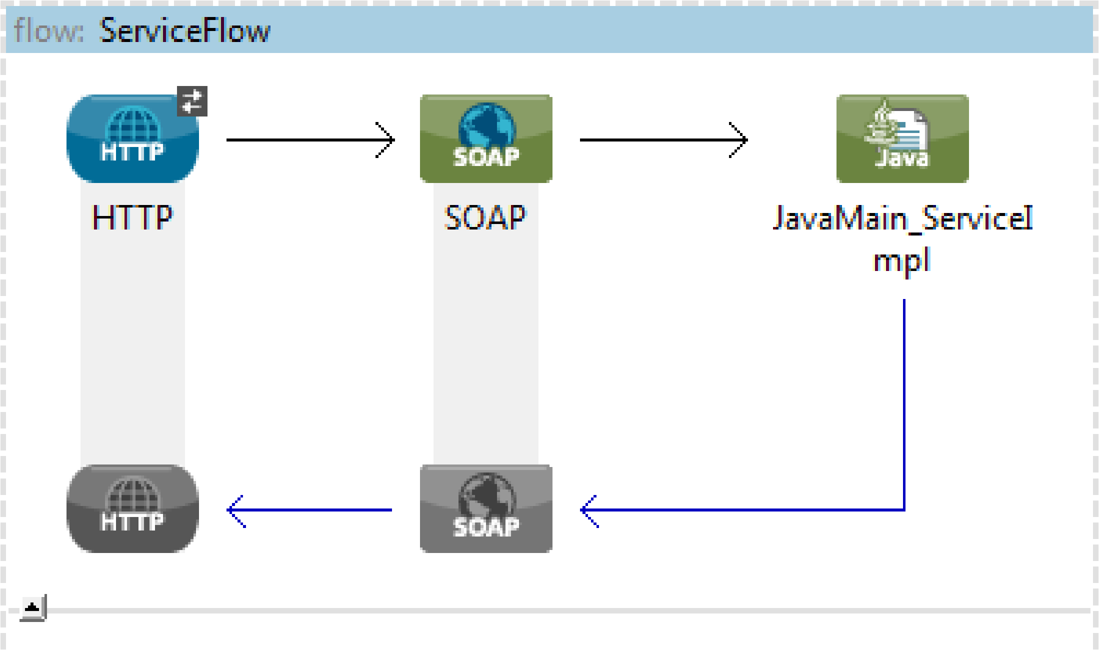
```
<?xml version="1.0" encoding="UTF-8"?>
<mule xmlns="http://www.mulesoft.org/schema/mule/core"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:jetty="http://www.mulesoft.org/schema/mule/jetty"
      xsi:schemaLocation="
        http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core/current/mule.xsd
        http://www.mulesoft.org/schema/mule/jetty http://www.mulesoft.org/schema/mule/jetty/current/mule-jetty.xsd">

    <jetty:connector name="jettyConnector">
        <jetty:webapps directory="${app.home}/webapps" port="8083"/>
    </jetty:connector>

</mule>
```

Now, let’s create our application such that the Web application can interact with our Mule :-

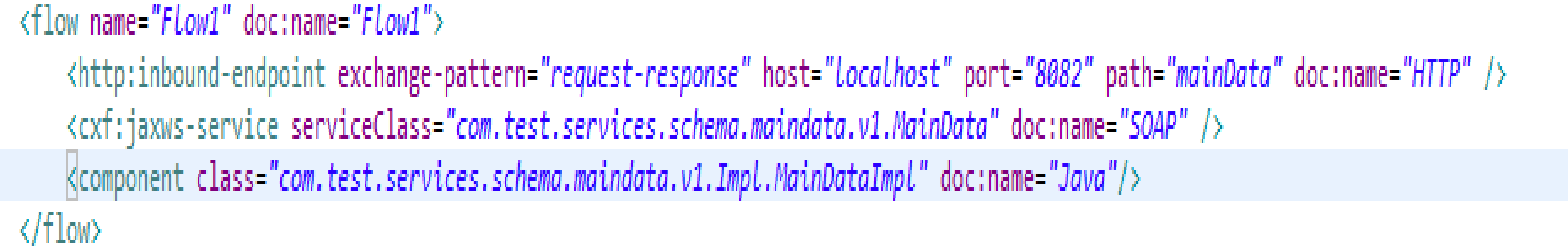


Here we can see we enter some value from a web page which is carried at the backend by a web service  and the data is stored/retrieve from Database

So, we will create another Mule flow that will expose the web service to carry the data backend :-

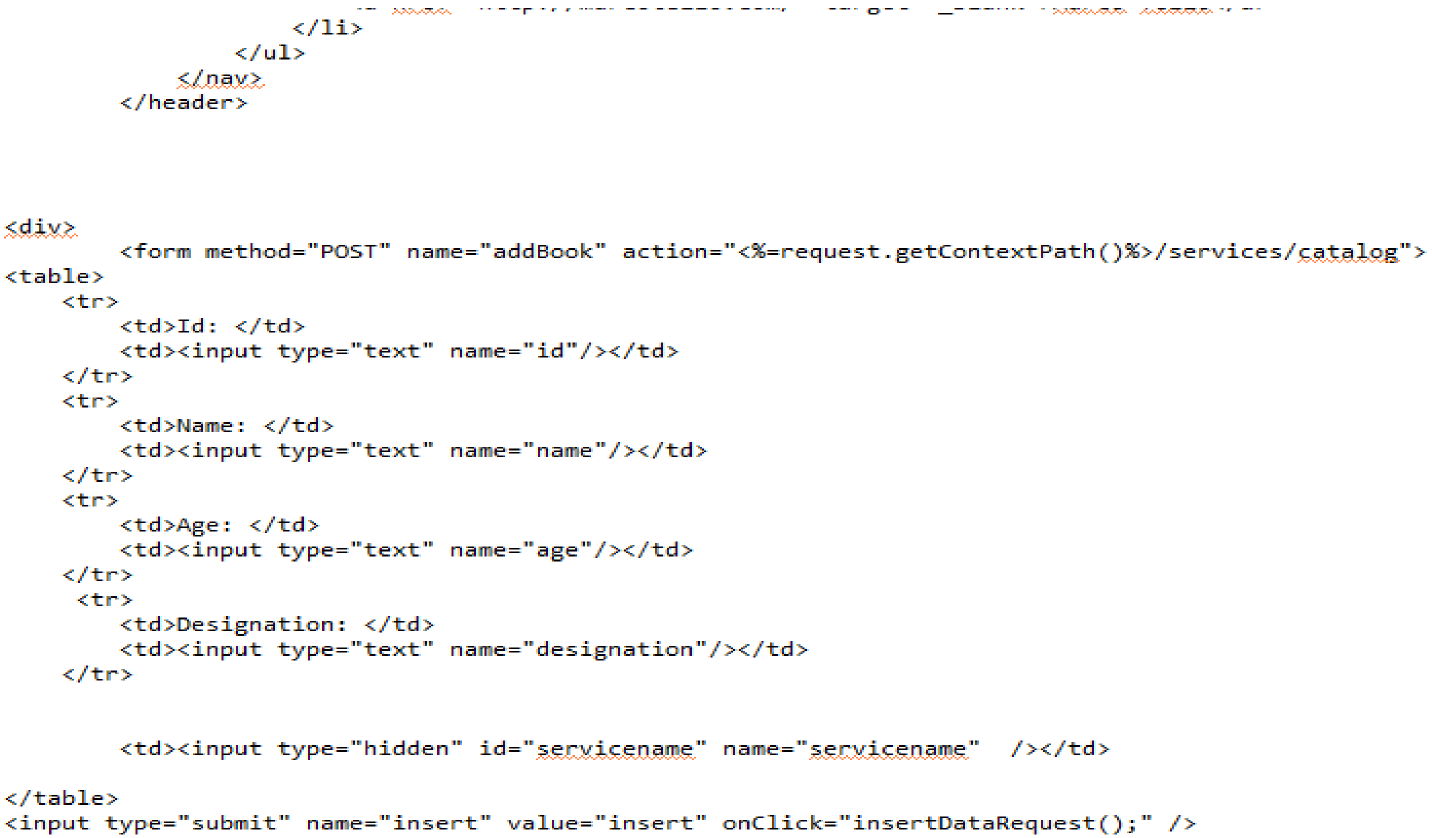


So, the Mule flow will be :-



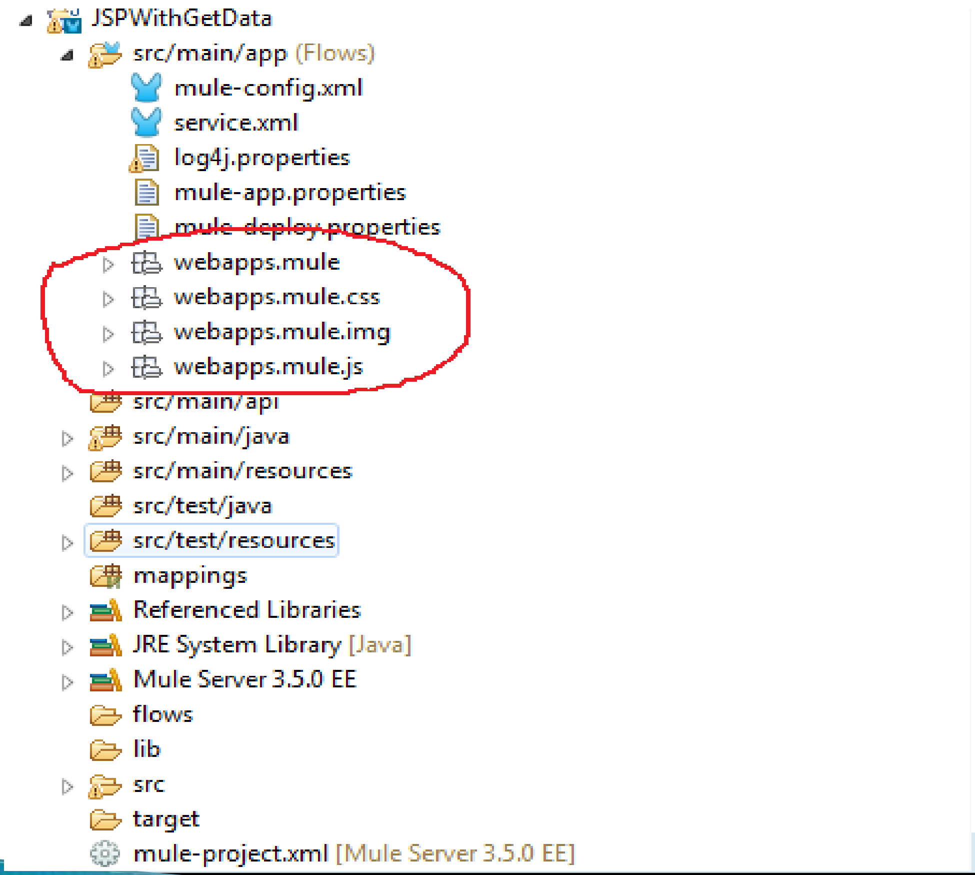
```
<flow name="Flow1" doc:name="Flow1">
    <http:inbound-endpoint exchange-pattern="request-response" host="localhost" port="8082" path="mainData" doc:name="HTTP" />
    <cxf:jaxws-service serviceClass="com.test.services.schema.maindata.v1.MainData" doc:name="SOAP" />
    <component class="com.test.services.schema.maindata.v1.Impl.MainDataImpl" doc:name="Java"/>
</flow>
```

And our User interface file will be :-



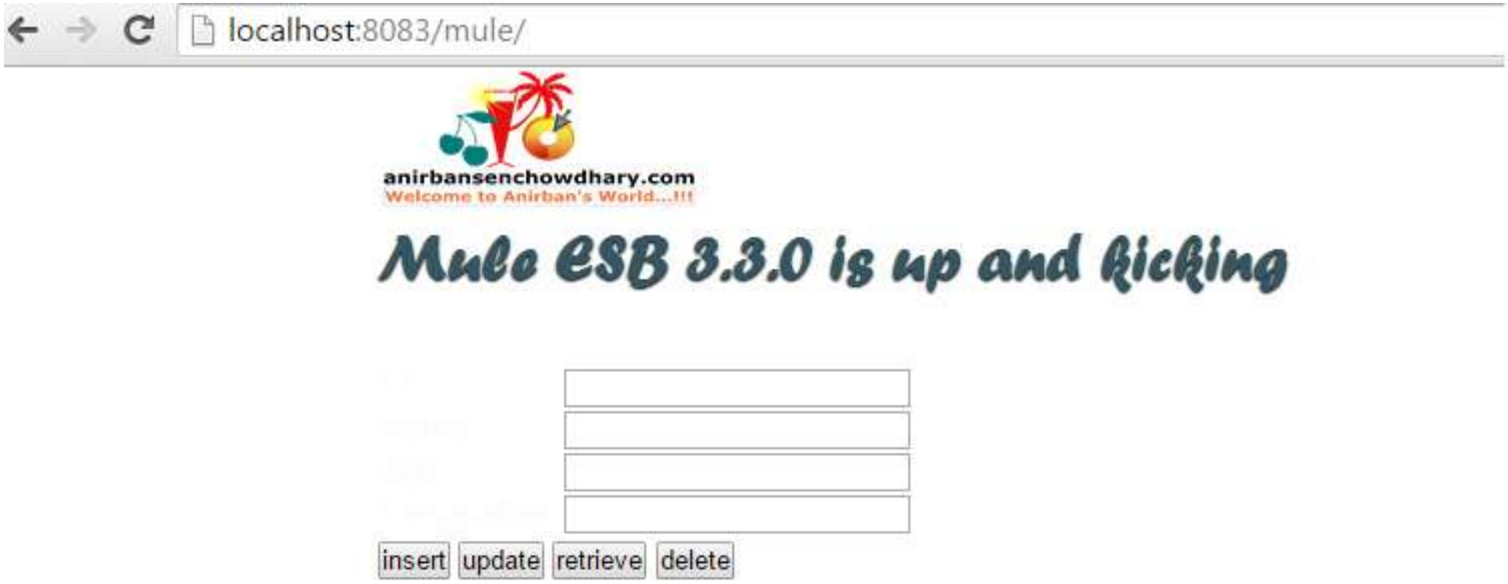
```
                    </li>
                </ul>
            </nav>
        </header>



<div>
        <form method="POST" name="addBook" action="<%=request.getContextPath()%>/services/catalog">
<table>
    <tr>
        <td>Id: </td>
        <td><input type="text" name="id"/></td>
    </tr>
    <tr>
        <td>Name: </td>
        <td><input type="text" name="name"/></td>
    </tr>
    <tr>
        <td>Age: </td>
        <td><input type="text" name="age"/></td>
    </tr>
     <tr>
        <td>Designation: </td>
        <td><input type="text" name="designation"/></td>
    </tr>


        <td><input type="hidden" id="servicename" name="servicename"  /></td>

</table>
<input type="submit" name="insert" value="insert" onClick="insertDataRequest();" />
```

The folder structure of our Mule application will be :-



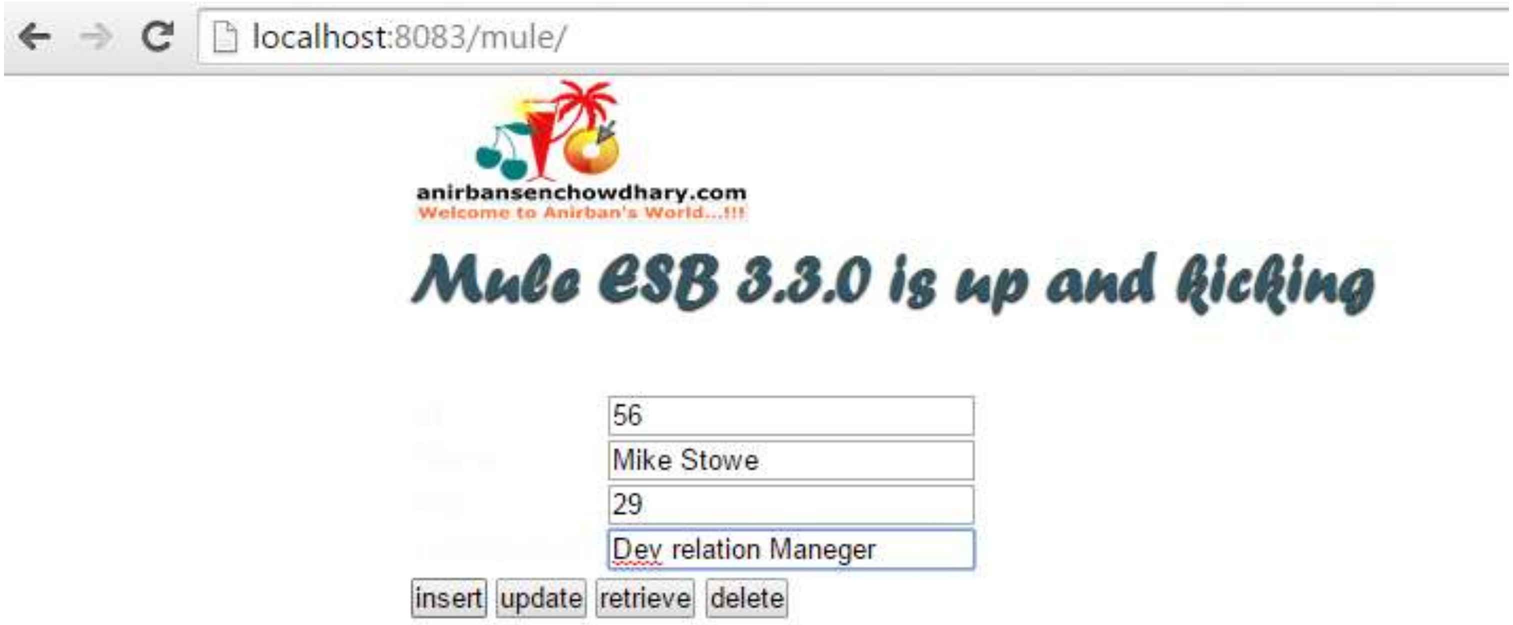
```
JSPWithGetData
    src/main/app (Flows)
        mule-config.xml
        service.xml
        log4j.properties
        mule-app.properties
        mule-deploy.properties
        webapps.mule
        webapps.mule.css
        webapps.mule.img
        webapps.mule.js
    src/main/api
    src/main/java
    src/main/resources
    src/test/java
    src/test/resources
    mappings
    Referenced Libraries
    JRE System Library [Java]
    Mule Server 3.5.0 EE
    flows
    lib
    src
    target
    mule-project.xml [Mule Server 3.5.0 EE]
```

Let’s test our application , and we will be hitting url http://localhost:8083/mule/ and we will get following :-



We can see our JSP page has appeared on the screen

Now let’s put some value and hit insert button, :-



Mule ESB 3.3.0 is up and kicking

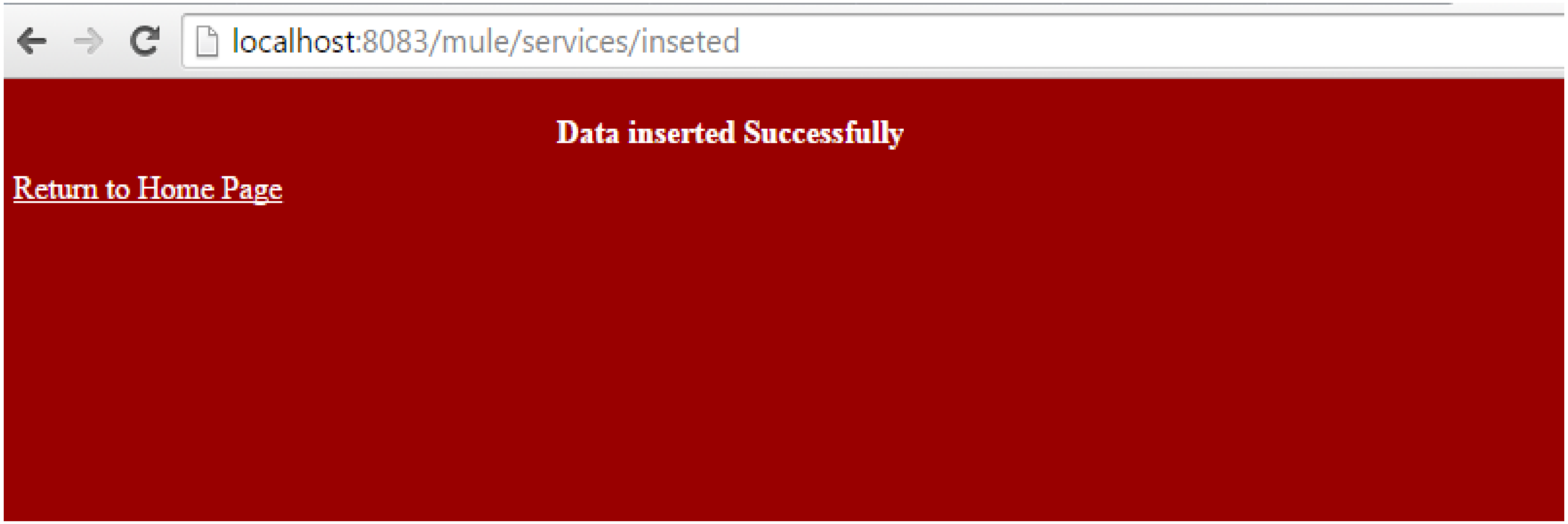
56

Mike Stowe

29

Dev relation Maneger

We will get the response back to the browser



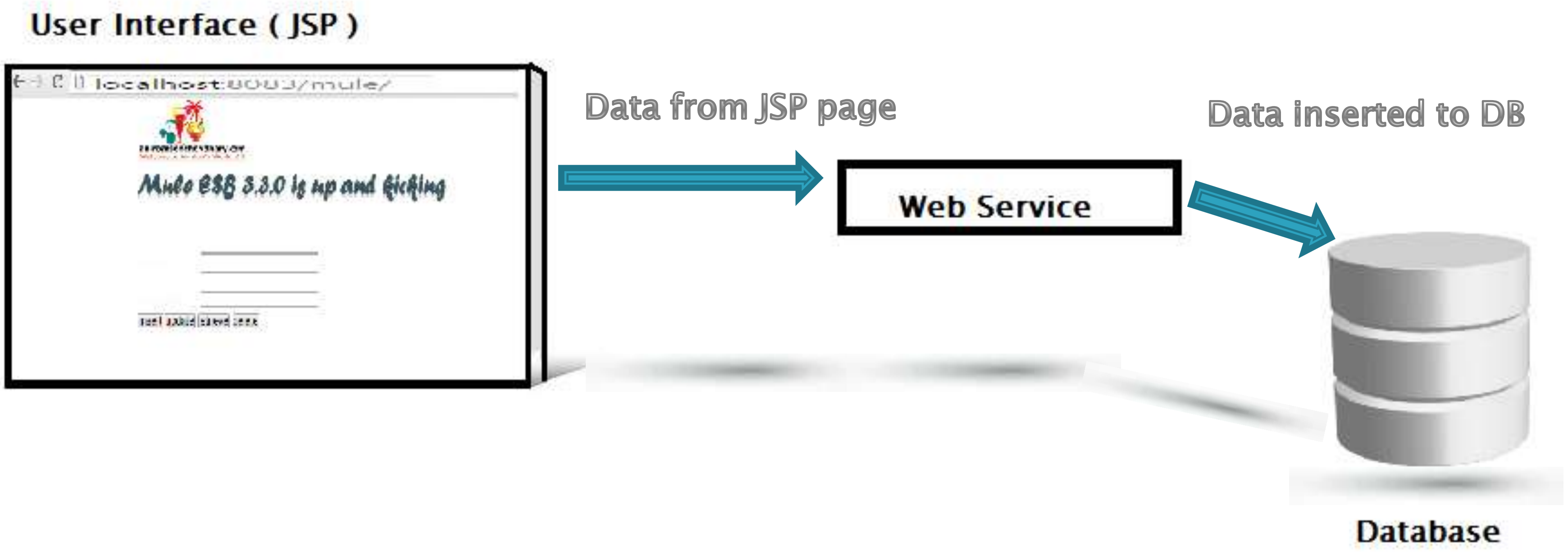
localhost:8083/mule/services/inseted

Data inserted Successfully

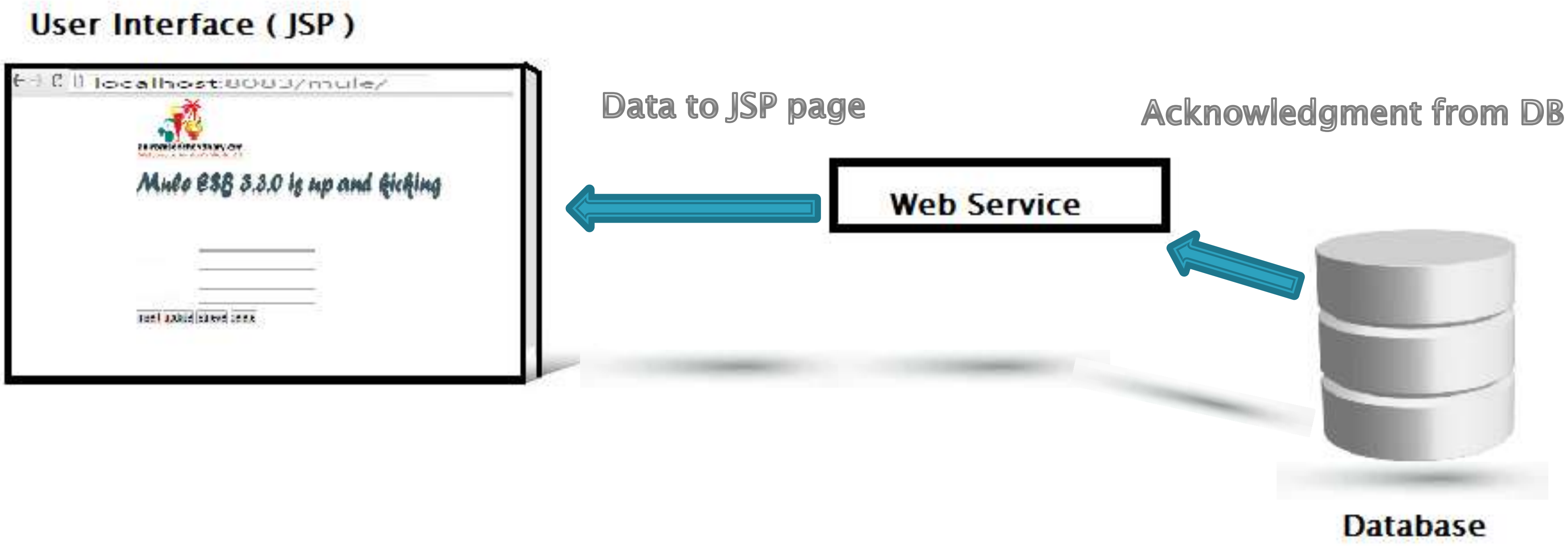
Return to Home Page

You can see easily that the backend web service exposed in Mule has inseted the data into the database and returned the response back to the **JSP** page

This is what happened behind …. When we entered the value in our web page (JSP) and hit the **insert** button, the data is carried into the database by the backend web service hosted in Mule



Now after the data has been inserted into database, the web service carries the response back to the web page (JSP)



So, we can see that Mule can host the web application using **Jetty** connector and we can interact with the web application with our other Mule flows easily, ad we have done here with a web service from our Mule flow which carried all the data to and from the web page .

Hope you liked this small tricks in Mule.
If you have any similar tricks in Mule, please do share it with all and let our **Mule** community grow
Happy coding ☺

# Thank You